ACM32H5xx 用户手册

基于 ARM Star-MC1 内核的 32 位微控制器

版本：V1.5

日期：2025-3-31

上海航芯电子科技股份有限公司

目录

1. 文档约定

1.1. 基本信息

本芯片是基于 ARMv8-M 架构 Star-MC1 内核的通用处理器芯片。芯片具体信息以数据手册为准。

1.2. 寄存器属性缩写表

寄存器说明中使用以下缩写词：

read/write (RW) :	可读写
read-only (RO) :	只读
write-only (WO) :	只写
read/clear write 0 (RCW0):	可读，写 0 清 0，写 1 无效
read/clear write 1 (RCW1):	可读，写 1 清 0，写 0 无效
read/clear by read (RCR):	可读，读该位后硬件自动清零该位，写无效
read/set (RS):	可读，也可将其置 1，写 0 无效
RSV:	保留

1.3. 术语

AHB:	高级高性能总线（advanced high-performance bus）
APB:	高级外设总线（advanced peripheral bus）
Word:	字，32 位数据
Half-word:	半字，16 位数据
Byte:	字节，8 位数据
NVR:	非易失区域（Non-Volatile Region）

2. 存储器及系统架构（SYSCFG）

2.1. 内核处理器

本芯片采用基于 ARMv8-M 架构的 Star MC1 处理器内核，支持 Cortex-M33 和 Cortex-M4F 指令集。系统频率最高可达 220MHz，支持单精度浮点运算（FPU）和 DSP 扩展，支持 MPU 存储保护功能。

内核处理器采用 Star-MC1。处理器包括两个总线接口分别称为 C-AHB 总线、S-AHB 总线：

- C-AHB 总线：用于访问代码区的指令或数据。
- S-AHB 总线：用于访问 SRAM 区、外部 RAM 区、外设区或厂商自定义系统区的指令或数据。
- 处理器配置了 16KB 的指令缓存（ICACHE）和 16KB 的数据缓存（DCACHE），可提高代码执行效率，同时支持 32KB 指令紧密耦合 RAM（ITCM）和 32KB 数据紧密耦合 RAM（DTCM）。

处理器的具体规格可参见 Star-MC1 的 User Guide。

2.2. 系统架构

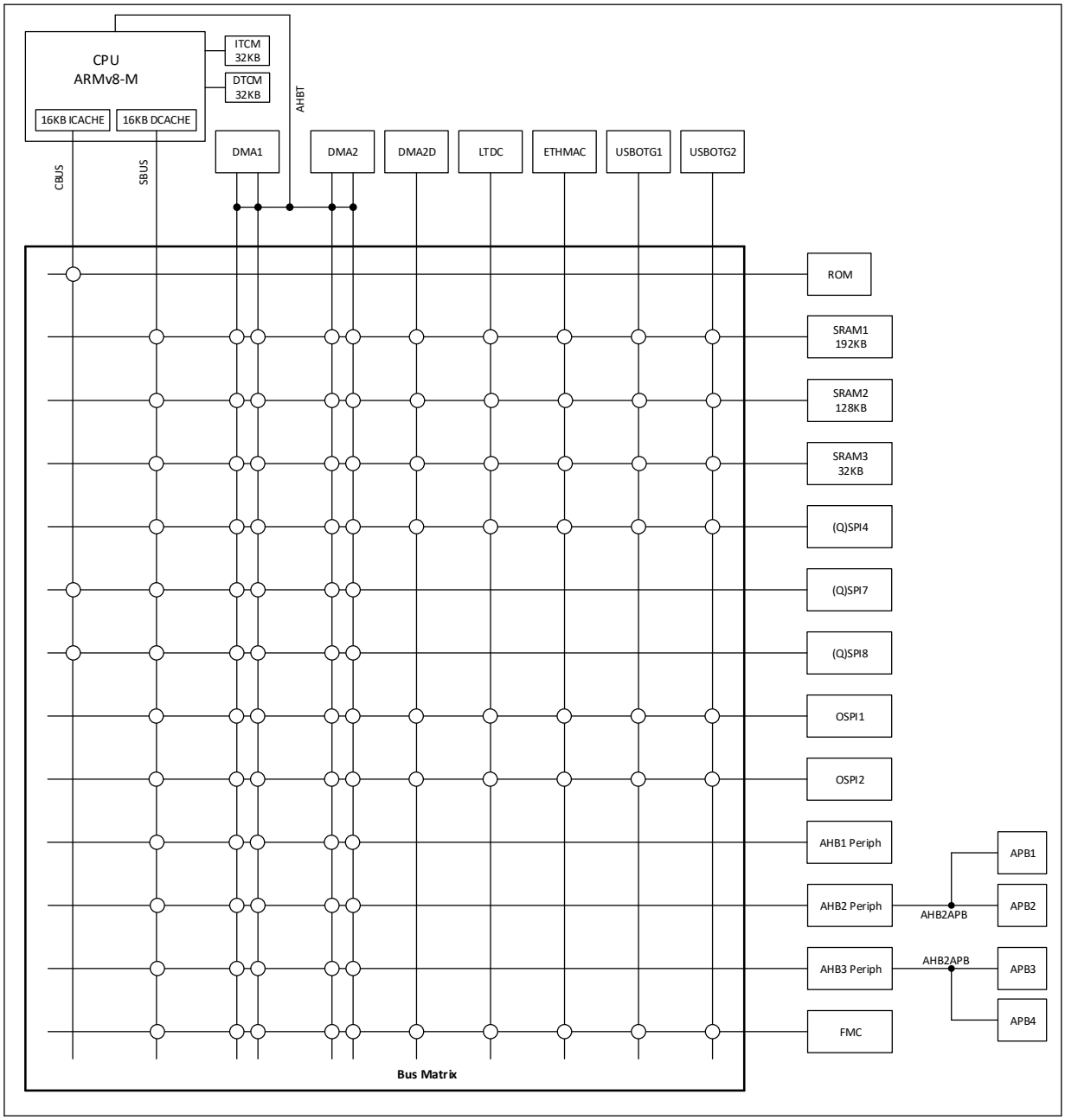
芯片采用 Multi-AHB 总线矩阵结构，该结构可实现系统中的多个主控总线和被控总线的并行访问。Multi-AHB 总线矩阵结构包括一个 AHB 互连矩阵、3 个 AHB 总线（AHB1\AHB2 和 AHB3）和 4 个 APB 总线（APB1、APB2、APB3 和 APB4）。

Multi-AHB 总线矩阵可连接：

- 10 条主控总线
  - CBUS（内核 Code 总线）
  - SBUS（内核 System 总线）
  - DMA1
  - DMA2
  - USBOTG1
  - USBOTG2
  - ETHMAC
  - SDMMC
  - LTDC
  - DMA2D
- 15 条被控总线
  - 内部 ROM
  - SRAM1（192KB）
  - SRAM2（128KB）
  - SRAM3（32KB）
  - ITCM（32KB）
  - DTCM（32KB）
  - (Q)SPI4
  - (Q)SPI7
  - (Q)SPI8

- OSPI1
- OSPI2
- AHB1 外设
- AHB2 外设
- AHB3 外设
- FMC

图 2-1 总线矩阵



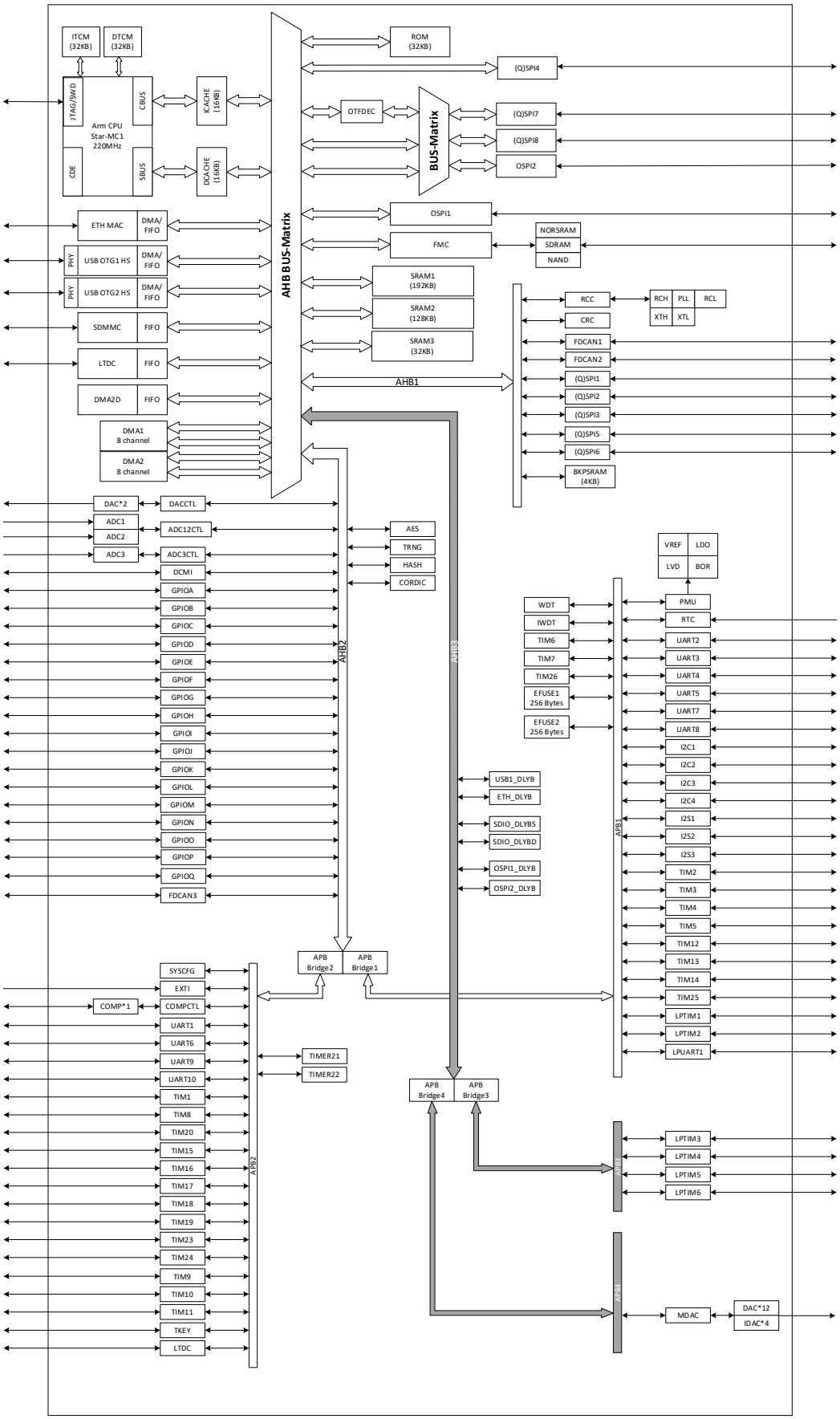
总线矩阵用于各主控总线间的访问仲裁管理。仲裁采用循环调度算法。

- CBUS 总线：用于 STAR-MC1 对内部代码区域的取指和数据访问
- SBUS 总线：用于 STAR-MC1 对 SRAM、外设、外部存储等区域的取指和数据访问
- DMA 总线：用于 DMA 对 FLASH、SRAM、外部存储区的数据访问
- AHBT 总线：CPU 提供 AHBT 总线，以允许 DMA 访问 ITCM 和 DTCM

- AHB2APB 总线桥：在 AHB 与 APB 总线间提供同步连接。
- 总线主设备外设：DMA2D/LTDC/ETHMAC/USBOTG1/USBOTG2 等主设备外设可对 SRAM、外部存储区的数据（通过 QSPI/OSPI/FMC）进行访问
- 在每次复位之后，所有的外设时钟处于关闭状态（除 SRAM1、SRAM2 和 SRAM3 外）。在用一个外设前，软件必须打开相应的时钟使能位。

系统架构图如下所示。

图 2-2 系统架构



注：AHB1、AHB2 和 AHB3 的时钟频率一致且等于系统频率，APB1、APB2、APB3 和 APB4 的时钟频率由系统频率分频而来，分频比可独立配置。

2.3. 存储器映射

内核处理器采用哈佛结构，可以使用相互独立的总线来读取指令和加载/存储数据。指令代码和数据都位于相同的存储器地址空间，但在不同的地址范围。程序存储器，数据存储器，寄存器和 I/O 端口都在同一个 4 GB 的地址空间之内。存储器中字节数据以小端方式排列。

处理器地址映射如下所示

图 2-3 存储器地址映射

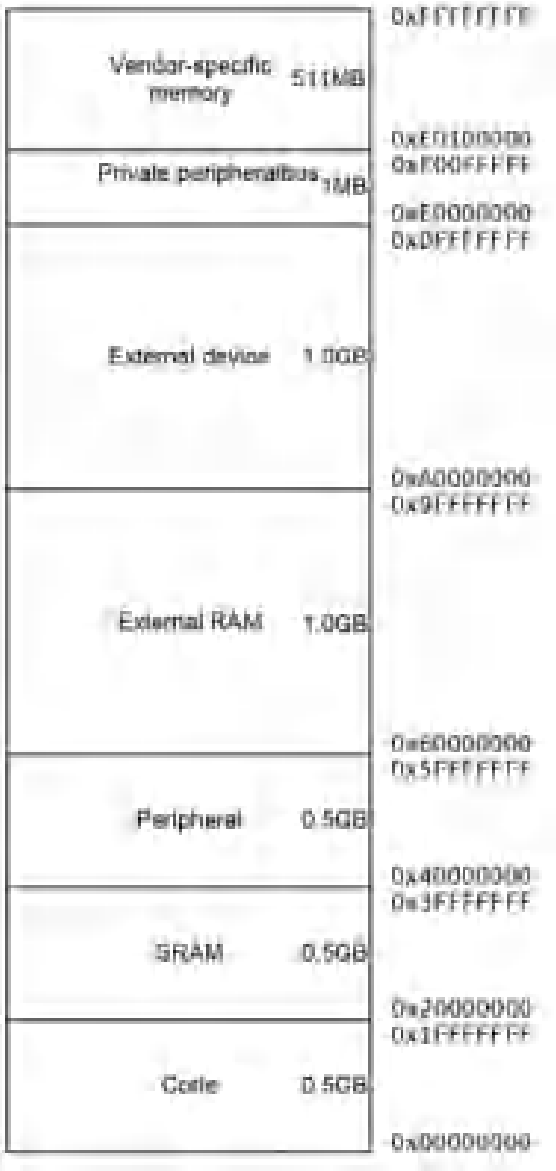


表 2-1 存储地址映射表

CODE	
ROM 或 ITCM	0x0000_0000——0x07FF_FFFF 0x1FF0_0000——0x1FF7_FFFF （ITCM：0x0000_0000——0x0000_7FFF）
(Q)SPI7-MEM	0x0800_0000——0x0FFF_FFFF
(Q)SPI8-MEM	0x1000_0000——0x17FF_FFFF

保留	0x1800_0000——0x1FEF_FFFF
保留	0x1FF8_0000——0x1FFF_FFFF
RAM	
DTCM	0x2000_0000——0x2000_7FFF
SRAM1	0x2000_8000——0x2003_7FFF
SRAM2	0x2003_8000——0x2005_7FFF
SRAM3	0x2005_8000——0x2005_FFFF
保留	0x2006_0000——0x3FFF_FFFF
APB1 Slave	
TIM2	0x4000_0000——0x4000_03FF
TIM3	0x4000_0400——0x4000_07FF
TIM4	0x4000_0800——0x4000_0BFF
TIM5	0x4000_0C00——0x4000_0FFF
TIM6	0x4000_1000——0x4000_13FF
TIM7	0x4000_1400——0x4000_17FF
TIM12	0x4000_1800——0x4000_1BFF
TIM13	0x4000_1C00——0x4000_1FFF
TIM14	0x4000_2000——0x4000_23FF
保留	0x4000_2400——0x4000_27FF
RTC	0x4000_2800——0x4000_2BFF
WDT	0x4000_2C00——0x4000_2FFF
IWDT	0x4000_3000——0x4000_33FF
I2S1	0x4000_3400——0x4000_37FF
I2S2	0x4000_3800——0x4000_3BFF
I2S3	0x4000_3C00——0x4000_3FFF
保留	0x4000_4000——0x4000_43FF
UART2	0x4000_4400——0x4000_47FF
UART3	0x4000_4800——0x4000_4BFF
UART4	0x4000_4C00——0x4000_4FFF
UART5	0x4000_5000——0x4000_53FF

I2C1	0x4000_5400——0x4000_57FF
I2C2	0x4000_5800——0x4000_5BFF
I2C3	0x4000_5C00——0x4000_5FFF
I2C4	0x4000_6000——0x4000_63FF
保留	0x4000_6400——0x4000_67FF
保留	0x4000_6800——0x4000_6BFF
保留	0x4000_6C00——0x4000_6FFF
PMU	0x4000_7000——0x4000_73FF
保留	0x4000_7400——0x4000_77FF
保留	0x4000_7800——0x4000_7BFF
LPTIM1	0x4000_7C00——0x4000_7FFF
LPUART1	0x4000_8000——0x4000_83FF
保留	0x4000_8400——0x4000_93FF
LPTIM2	0x4000_9400——0x4000_97FF
UART7	0x4000_9800——0x4000_9BFF
UART8	0x4000_9C00——0x4000_9FFF
TIM25	0x4000_A000——0x4000_A3FF
TIM26	0x4000_A400——0x4000_A7FF
保留	0x4000_A800——0x4000_DFFF
EFUSE1	0x4000_E000——0x4000_E7FF
EFUSE2	0x4000_E800——0x4000_EFFF
保留	0x4000_EA00——0x4000_FFFF
APB2 Slave	
SYSCFG	0x4001_0000——0x4001_00FF
保留	0x4001_0100——0x4001_01FF
COMP1	0x4001_0200——0x4001_02FF
保留	0x4001_0300——0x4001_03FF
EXTI	0x4001_0400——0x4001_07FF
保留	0x4001_0800——0x4001_08FF
保留	0x4001_0900——0x4001_09FF

保留	0x4001_0A00——0x4001_0BFF
TIM1	0x4001_2C00——0x4001_2FFF
保留	0x4001_3000——0x4001_33FF
TIM8	0x4001_3400——0x4001_37FF
UART1	0x4001_3800——0x4001_3BFF
UART6	0x4001_3C00——0x4001_3FFF
TIM15	0x4001_4000——0x4001_43FF
TIM16	0x4001_4400——0x4001_47FF
TIM17	0x4001_4800——0x4001_4BFF
保留	0x4001_4C00——0x4001_4FFF
TIM20	0x4001_5000——0x4001_53FF
保留	0x4001_5400——0x4001_57FF
保留	0x4001_5800——0x4001_63FF
TKEY	0x4001_6400——0x4001_67FF
LTDC	0x4001_6800——0x4001_6BFF
保留	0x4001_6C00——0x4001_7FFF
TIM9	0x4001_8000——0x4001_83FF
TIM10	0x4001_8400——0x4001_87FF
TIM11	0x4001_8800——0x4001_8BFF
保留	0x4001_8C00——0x4001_8FFF
TIM18	0x4001_9000——0x4001_93FF
TIM19	0x4001_9400——0x4001_97FF
保留	0x4001_9800——0x4001_9BFF
TIM21	0x4001_9C00——0x4001_9FFF
TIM22	0x4001_A000——0x4001_A3FF
TIM23	0x4001_A400——0x4001_A7FF
TIM24	0x4001_A800——0x4001_ABFF
保留	0x4001_AC00——0x4001_AFFF
UART9	0x4001_B000——0x4001_B3FF
UART10	0x4001_B400——0x4001_B7FF

保留	0x4001_B800——0x4001_FFFF
AHB1 Slave	
DMA1	0x4002_0000——0x4002_03FF
DMA2	0x4002_0400——0x4002_07FF
保留	0x4002_0800——0x4002_0BFF
保留	0x4002_0C00——0x4002_0FFF
RCC	0x4002_1000——0x4002_13FF
保留	0x4002_1400——0x4002_1FFF
FDCAN1	0x4002_2000——0x4002_23FF
FDCAN2	0x4002_2400——0x4002_27FF
保留	0x4002_2800——0x4002_2FFF
CRC	0x4002_3000——0x4002_33FF
保留	0x4002_3400——0x4002_7FFF
ETHMAC	0x4002_8000——0x4002_93FF
保留	0x4002_9400——0x4002_AFFF
DMA2D	0x4002_B000——0x4002_BBFF
保留	0x4002_BC00——0x4002_FFFF
(Q)SPI1	0x4003_0000——0x4003_03FF
(Q)SPI2	0x4003_0400——0x4003_07FF
(Q)SPI3	0x4003_0800——0x4003_0BFF
(Q)SPI4-REG	0x4003_0C00——0x4003_0FFF
(Q)SPI5	0x4003_1000——0x4003_13FF
(Q)SPI6	0x4003_1400——0x4003_17FF
保留	0x4003_1800——0x4003_63FF
BKP_SRAM	0x4003_6400——0x4003_73FF
保留	0x4003_7400——0x4003_FFFF
USB1_OTG_HS	0x4004_0000——0x4007_FFFF
USB2_OTG_HS	0x4008_0000——0x400B_FFFF
USB1_OTG_PHY_CFG	0x400C_0000——0x400C_03FF
USB2_OTG_PHY_CFG	0x400C_0400——0x400C_07FF

保留	0x400C_0800——0x47FF_FFFF
AHB2 Slave	
GPIOA	0x4800_0000——0x4800_03FF
GPIOB	0x4800_0400——0x4800_07FF
GPIOC	0x4800_0800——0x4800_0BFF
GPIOD	0x4800_0C00——0x4800_0FFF
GPIOE	0x4800_1000——0x4800_13FF
GPIOF	0x4800_1400——0x4800_17FF
GPIOG	0x4800_1800——0x4800_1BFF
GPIOH	0x4800_1C00——0x4800_1FFF
GPIOI	0x4800_2000——0x4800_23FF
GPIOJ	0x4800_2400——0x4800_27FF
GPIOK	0x4800_2800——0x4800_2BFF
GPIOL	0x4800_2C00——0x4800_2FFF
GPIOM	0x4800_3000——0x4800_33FF
GPION	0x4800_3400——0x4800_37FF
GPIOO	0x4800_3800——0x4800_3BFF
GPIOP	0x4800_3C00——0x4800_3FFF
GPIOQ	0x4800_4000——0x4800_43FF
保留	0x4800_4400——0x4FFF_FFFF
ADC1/ADC2	0x5000_0000——0x5000_03FF
ADC3	0x5000_0400——0x5000_07FF
DAC1	0x5000_0800——0x5000_0BFF
DAC2	0x5000_0C00——0x5000_0FFF
保留	0x5000_1000——0x5000_13FF
FDCAN3	0x5000_1400——0x5000_17FF
保留	0x5000_1800——0x5004_FFFF
DCMI	0x5005_0000——0x5005_03FF
保留	0x5005_0400——0x5005_FFFF
AES	0x5006_0000——0x5006_03FF

CORDIC	0x5006_0400——0x5006_07FF
HRNG	0x5006_0800——0x5006_0BFF
HASH	0x5006_0C00——0x5006_0FFF
AES_SPI1	0x5006_1000——0x5006_13FF
保留	0x5006_1400——0x5006_17FF
保留	0x5006_1800——0x5007_FFFF
保留	0x5008_0000——0x5009_FFFF
保留	0x500A_1000——0x50FF_FFFF
APB3 Slave	
LPTIM3	0x5100_0000——0x5100_03FF
LPTIM4	0x5100_0400——0x5100_07FF
LPTIM5	0x5100_0800——0x5100_0BFF
LPTIM6	0x5100_0C00——0x5100_0FFF
保留	0x5100_1000——0x5100_13FF
保留	0x5100_1400——0x51FF_FFFF
AHB3 Slave	
保留	0x5200_0000——0x5200_3FFF
保留	0x5200_4000——0x5200_4FFF
(Q)SPI7-REG	0x5200_5000——0x5200_53FF
(Q)SPI8-REG	0x5200_5400——0x5200_57FF
保留	0x5200_5800——0x5200_5FFF
保留	0x5200_6000——0x5200_67FF
USB1_DLYB-REG	0x5200_6800——0x5200_6BFF
ETH_DLYB-REG	0x5200_6C00——0x5200_6FFF
保留	0x5200_7000——0x520C_7FFF
SDIO	0x520C_8000——0x520C_9FFF
SDIO_DLYBS-REG	0x520C_A000——0x520C_A7FF
SDIO_DLYBD-REG	0x520C_A800——0x520C_AFFF
保留	0x520C_8C00——0x520C_8FFF
保留	0x520C_9000——0x520D_13FF

OSPI1-REG	0x520D_1400——0x520D_17FF
OSPI2-REG	0x520D_1800——0x520D_1BFF
保留	0x520D_1C00——0x520D_1FFF
OSPI1_DLYB-REG	0x520D_2000——0x520D_23FF
OSPI2_DLYB-REG	0x520D_2400——0x520D_27FF
保留	0x520D_2800——0x5FFF_FFFF
APB4 Slave	
保留	0x5300_0000——0x5300_03FF
保留	0x5300_0400——0x5300_07FF
保留	0x5300_0800——0x5300_0BFF
保留	0x5300_0C00——0x5300_0FFF
保留	0x5300_1000——0x5300_13FF
保留	0x5300_1400——0x5300_17FF
保留	0x5300_1800——0x5300_1BFF
MDAC	0x5300_1C00——0x5300_1FFF
保留	0x5300_2000——0x5300_23FF
保留	0x5300_2400——0x5300_27FF
保留	0x5300_2800——0x5300_2BFF
保留	0x5300_2C00——0x5300_2FFF
保留	0x5300_3000——0x5300_33FF
保留	0x5300_3400——0x5300_37FF
保留	0x5300_3800——0x5300_3BFF
保留	0x5300_3C00——0x53FF_FFFF
External memory	
FMC-NORSRAM	0x6000_0000——0x6FFF_FFFF
SDRAM1_SDRAM2	0x7000_0000——0x7FFF_FFFF
OSPI1-MEM	0x8000_0000——0x87FF_FFFF
OSPI2-MEM	0x8800_0000——0x8FFF_FFFF
(Q)SPI4-MEM	0x9000_0000——0x9FFF_FFFF
External Peripheral	

FMC-REG	0xA000_0000——0xA000_03FF
保留	0xA000_0400——0xAFFF_FFFF
保留	0xB000_0000——0xBFFF_FFFF
SDRAM1_NORSRAM	0xC000_0000——0xCFFF_FFFF
SDRAM2	0xD000_0000——0xDFFF_FFFF

2.3.1. SRAM

芯片内部集成 SRAM 的特性如下：

- 352KB 的系统 SRAM，分成：
  - 192KB SRAM1，带 ECC，支持双 bit 错误检测及单 bit 纠错
  - 128KB SRAM2
  - 32KB SRAM3，带 ECC，支持双 bit 错误检测及单 bit 纠错
  - 均支持字节、半字（16 位）以及字（32 位）访问
- 32KB ITCM（指令紧密耦合 RAM），可在最高系统时钟频率下以零等待周期寻址
- 32KB DTCM（数据紧密耦合 RAM），可在最高系统时钟频率下以零等待周期寻址
- 4KB 的备份 SRAM（BKPSRAM），带 ECC，支持双 bit 错误检测及单 bit 纠错

2.3.2. ITCM 和 DTCM

TCM=Tightly Coupled Memory，是一种高速内存，直接集成在 CPU 内核中，为降低访问延迟进行了优化，可在最高系统时钟频率下以零等待周期寻址。

TCM 分为 ITCM（指令紧密耦合 RAM）和 DTCM（数据紧密耦合 RAM）。

TCM 可供 CPU 和 DMA 访问，且不会占用 Cache 资源，支持字节、半字（16 位）以及字（32 位）访问。

某些对时间要求非常严格的代码，可以被放到 ITCM 中执行，这可以有效地提高运行速度。某些需要频繁存取的数据，也可以放到 DTCM 中以节省存取时间。

ITCM 被映射到地址 0x00000000。

DTCM 被映射到地址 0x20000000。

2.3.2.1. ITCM 控制寄存器(ITCM_CR)

地址：0xE001E010

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留

6:3	SZ	RO	0110	TCM 容量 0000：不支持 0011：4KB 0100：8KB 0101：16KB 0110：32KB 0111：64KB 1000：128KB 1001：256KB 1010：512KB 1011：1MB 1100：2MB 1101：4MB 1110：8MB 1111：16MB 其他：保留
2:1	RSV	-	-	保留
0	EN	RW	0	TCM 使能控制 0：禁止 1：使能

2.3.2.2. DTCM 控制寄存器(DTCM_CR)

地址：0xE001E014

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:3	SZ	RO	0110	TCM 容量 0000：不支持 0011：4KB 0100：8KB 0101：16KB 0110：32KB 0111：64KB 1000：128KB 1001：256KB 1010：512KB 1011：1MB 1100：2MB 1101：4MB 1110：8MB 1111：16MB 其他：保留
2:1	RSV	-	-	保留

0	EN	RW	0	TCM 使能控制 0：禁止 1：使能

2.3.3. EFUSE

芯片内置两块 EFUSE，每块容量 256Bytes。按 Byte 操作，只能写一次。

EFUSE1 用于芯片配置，EFUSE2 可供用户使用。

2.3.4. 内部叠封 SPI-FLASH

芯片内部叠封 SPI-FLASH，通过 SPI7 访问，支持内存映射模式。

其特性由具体封装型号决定。

2.3.5. 内部叠封 OSPI-PSRAM

芯片内部叠封 OSPI-PSRAM，通过 OSPI2 访问，支持内存映射模式。

其容量和特性由具体封装型号决定。

2.3.6. 内部叠封 SDRAM

芯片内部叠封 SDRAM，其容量和特性由具体封装型号决定。

2.4. 启动配置

芯片上电时总是从 ROM 启动，首先读取 EFUSE 中的 BOOT_DEVICE 启动设备字段、系统寄存器 WMR 的 BOOTDEVICE 标志位（锁存 BOOT1/2 引脚的状态），决定选用哪一组 SPI 端口 FLASH 作为外部启动设备。

然后读取 EFUSE 和 SPI-FLASH 中的 BOOT_MODE 启动模式字段、系统寄存器 WMR 的 BOOTMODE 标志位（锁存 BOOT0 引脚的状态），决定是进入 ROM bootloader 下载模式还是从 SPI-FLASH 启动。

芯片出厂前会通过 EFUSE 中的 BOOT_DEVICE 对外部启动设备进行配置，所以用户无需关心外部启动设备配置。

BOOT[2:0]引脚状态锁存：

复位后（上电复位、外部引脚复位）或从 Standby 模式退出时，在 SYSCLK 的第四个上升沿锁存 BOOT[2:0]引脚的值；BOOT0 保存到系统寄存器 WMR 的 BOOTMODE 位，BOOT1 和 BOOT2 保存到 BOOTDEVICE[1:0]位。

BOOT0 引脚为 BOOT 专用引脚，BOOT2 和 BOOT1 引脚在复位或退出待机模式后，用户可以使用 BOOT2 和 BOOT1 引脚上的复用功能。

表 2-2 外部启动设备选择

配置			SPI 端口 FLASH
EFUSE.BOOT_DEVICE	BOOT2	BOOT1	
0xF0	x	x	(Q)SPI8-1NOR Flash
0xE1	x	x	(Q)SPI8-0 NOR Flash
0xC3	x	x	(Q)SPI7-1 NOR Flash
0xD2	x	x	(Q)SPI7-0 NOR Flash
not 0xF0, 0xE1, 0xC3, 0xD2	0	0	(Q)SPI8-1NOR Flash
	0	1	(Q)SPI8-0 NOR Flash
	1	0	(Q)SPI7-1 NOR Flash
	1	1	(Q)SPI7-0 NOR Flash

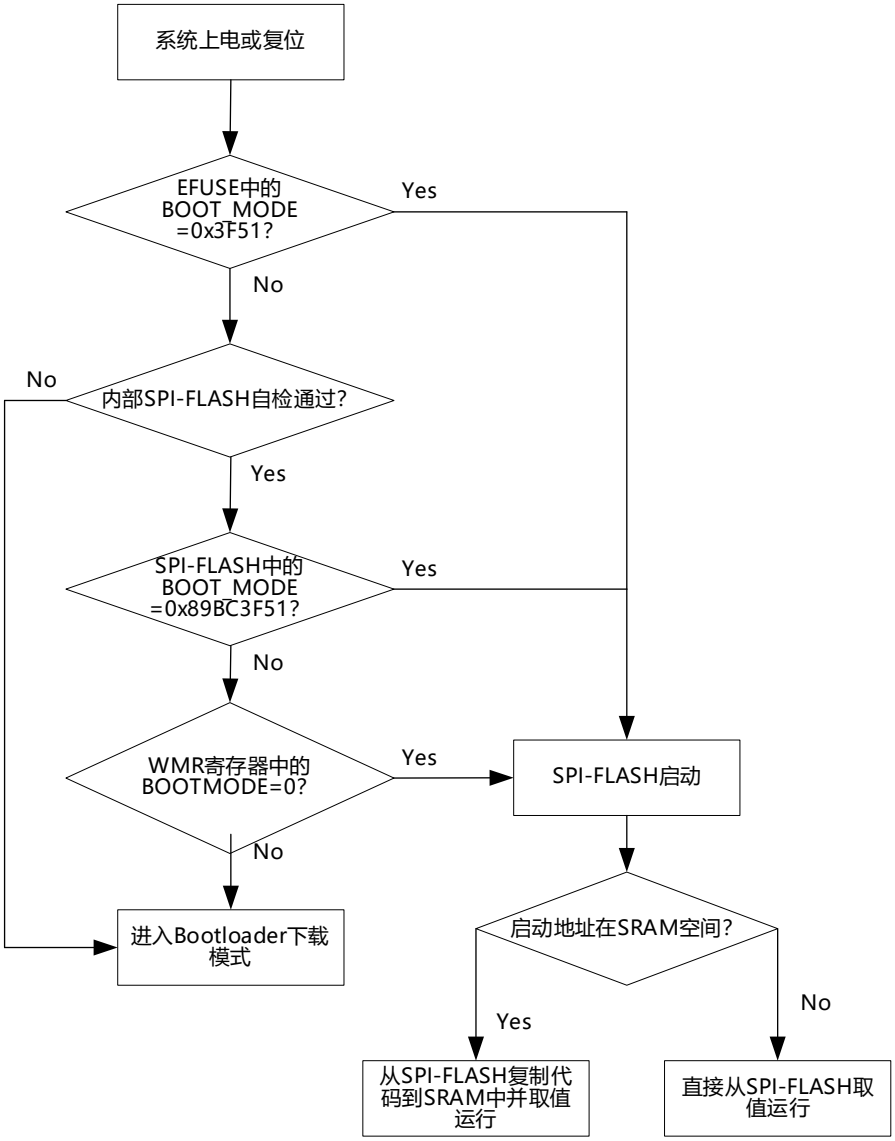
注：(Q)SPI7 NOR Flash 在芯片管脚上有 2 组：(Q)SPI7-0 和(Q)SPI7-1；(Q)SPI8 NOR Flash 在芯片管脚上有 2 组：(Q)SPI8-0 和(Q)SPI8-1

表 2-3 启动模式选择

启动模式选择			启动模式
EFUSE.BOOT_MODE	SPI-FLASH.BOOT_MODE	BOOT0	
0x3F51	x	x	从 SPI-FLASH 启动
Not 0x3F51	0x3F51	x	从 SPI-FLASH 启动
Not 0x3F51	Not 0x89BC3F51	0	从 SPI-FLASH 启动
Not 0x3F51	Not 0x89BC3F51	1	进入 ROM bootloader 下载模式

下图描述了芯片启动模式选择过程。

图 2-4 芯片启动模式选择



2.5. 设备唯一序列号

每颗芯片都包含唯一的 128 位（16 字节）的序列号。

序列号地址：EFUSE1 的偏移地址 0x58 开始。

2.6. 系统配置寄存器（SYSCFG）

2.6.1. 寄存器列表

SYSCFG 寄存器基地址：0x40010000

偏移	名称	复位值	描述
0x00	SYSCFG_SYSCR	0xc0000000	系统控制寄存器
0x04	SYSCFG_WMR	0x00000340	工作模式寄存器
0x08	SYSCFG_VER	0xfecf0130	版本寄存器
0x0C	保留		

0x10	SYSCFG_PHYCFG	0x00000011	USBPHY 模块控制寄存器
0x14	SYSCFG_DUMMY	0x00000000	Dummy 寄存器，可读写
0x18	SYSCFG_RAMECCIR	0x00000000	SRAM ECC 中断寄存器
0x1C	SYSCFG_RAMECCSR	0x00000000	SRAM ECC 状态寄存器
0x20	SYSCFG_RAMECCICR	0x00000000	SRAM ECC 中断清除寄存器
0x24	SYSCFG_GPIO5VOCR1	0x00000000	GPIO 5V 输出控制寄存器 1
0x28	SYSCFG_GPIO5VOCR2	0x00000000	GPIO 5V 输出控制寄存器 2

2.6.2. 系统控制寄存器(SYSCFG_SYSCR: 00h)

位域	名称	属性	复位值	描述
31:30	RSV	-	11	保留（复位值为 11）
29:21	RSV	-	-	保留
20	ETHMAC_RX_DLYSEL	RW	0	ETHMAC RX 时钟延时模块的时钟输入选择 0：HCLK 1：ETHMAC PHY RX 时钟
19	BOOT_PUN	RW	0	BOOT0 pin 脚上拉控制位： 0：使能上拉 1：禁止上拉
18:16	RSV	-	-	保留
15:13	EPIS	RW	0	以太网 PHY 接口选择 000：MII 001~011：保留 100：RMII 101~111：保留 注意：必须在 ETH 处于复位状态且在使能 ETH 时钟之前完成以太网 PHY 接口选择（EPIS）配置，否则需要重新复位 ETH 模块使该配置生效。
12	ETHMAC_TX_CLKGE	RW	0	以太网 MAC 控制器的 TX_CLK 时钟门控信号禁止位： 0：TX_CLK 时钟门控信号有效 1：TX_CLK 时钟门控信号禁止
11	BKPSRAM_LOCK	RW	0	Backup SRAM ECC 双 bit 错误检测锁定 0：Backup SRAM ECC 双 bit 错误检测输出从 TIM1/8/15/16/17 的 break 输入端断开 1：Backup SRAM ECC 双 bit 错误检测输出与 TIM1/8/15/16/17 的 break 输入端连接

10	SRAM3_LOCK	RW	0	SRAM3 ECC 双 bit 错误检测锁定 0：SRAM3 ECC 双 bit 错误检测输出从 TIM1/8/15/16/17 的 break 输入端断开 1：SRAM3 ECC 双 bit 错误检测输出与 TIM1/8/15/16/17 的 break 输入端连接
9	SRAM1_LOCK	RW	0	SRAM1 ECC 双 bit 错误检测锁定 0：SRAM1 ECC 双 bit 错误检测输出从 TIM1/8/15/16/17 的 break 输入端断开 1：SRAM1 ECC 双 bit 错误检测输出与 TIM1/8/15/16/17 的 break 输入端连接
8	RSV	-	-	保留
7	SDRAM_IO_SWP	RW	0	SDRAM IO 重映射 0：FMC 默认 IO 映射 1：SDRAM 专用 IO 映射
6	RSV	-	-	保留
5:4	FMC_SWP	RW	0	FMC 存储器映射地址交换，用于交换 SDRAM 存储区域和 NOR/PSRAM 存储区域或重映射 SDRAM2 存储区 00：不交换存储区映射 NOR/PSRAM：0x60000000~0x6FFFFFFF SDRAM1：0x70000000~0x7FFFFFFF 或 0xC0000000~0xCFFFFFFF SDRAM2：0xD0000000~0xDFFFFFFF 01：交换 NOR/PSRAM 存储区和 SDRAM 存储区 NOR/PSRAM：0xC0000000~0xCFFFFFFF SDRAM1：0x60000000~0x6FFFFFFF SDRAM2：0x70000000~0x7FFFFFFF 或 0xD0000000~0xDFFFFFFF 10：重映射 SDRAM2 存储区 NOR/PSRAM：0x60000000~0x6FFFFFFF SDRAM1：0xC0000000~0xCFFFFFFF SDRAM2：0x70000000~0x7FFFFFFF 或 0xD0000000~0xDFFFFFFF 11：保留
3	RSV	-	-	保留
2	LVD_LOCK	RW	0	LVD 检测错误锁定 0：LVD 检测错误输出从 TIM1/8/15/16/17 的 break 输入端断开 1：LVD 检测错误输出与 TIM1/8/15/16/17 的 break 输入端连接
1	RSV	-	-	保留

0	LOCKUP_LOCK	RW	0	Core LOCKUP 输出锁定 0： LOCKUP 输出从 TIM1/8/15/16/17 的 break 输入端断开 1： LOCKUP 输出与 TIM1/8/15/16/17 的 break 输入端连接

2.6.3. 工作模式寄存器(SYSCFG_WMR: 04h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:8	BOOTDEVICE	RO	11	复位后（上电复位、外部引脚复位）或从 Standby 模式退出时，锁存 BOOT2,BOOT1 引脚状态，当系统从片外 SPI-Flash 启动时，启动设备为： 00：(Q)SPI8-1（NOR Flash） 01：(Q)SPI8-0（NOR Flash） 10：(Q)SPI7-1（NOR Flash） 11：(Q)SPI7-0（NOR Flash）
7	RSV	-	-	保留
6	RTCREADY	RO	1	STANDBY 唤醒后，RTC 域是否准备完成 0：RTC 域未 ready，不能访问 1：RTC 域已 ready，可以访问
5:4	RSV	-	-	保留
3	BOOTMODE	RO	1	复位后（上电复位、外部引脚复位）或从 Standby 模式退出时，锁存 BOOT0 引脚状态： 0：指示系统从片外 SPI-Flash 启动 1：指示系统从 ROM 启动
2:0	RSV	-	-	保留

2.6.4. 版本寄存器(SYSCFG_VER: 08h)

位域	名称	属性	复位值	描述
31:0	VERSION	RO	0xFECF0130	芯片版本 15~8：0x01 代表 MCU 产品 7~4：0x3 代表芯片系列 3~0：0x0 代表芯片版本 V1 31~16 位为 15~0 位的取反值

2.6.5. USBPHY 配置寄存器(SYSCFG_PHYCFG: 10h)

位域	名称	属性	复位值	描述

31:9	RSV	-	-	保留
8	USB1_ULPI_DLYSEL	RW	0	USB1 ULPI 时钟延时模块的时钟输入选择 0：HCLK 1：ULPI_CLK
7:5	RSV	-	-	保留
4	USB2_PHY_RSTN	RW	1	USB2_PHY 软复位，写 0 复位，写 1 撤销
3:1	RSV	-	-	保留
0	USB1_PHY_RSTN	RW	1	USB1_PHY 软复位，写 0 复位，写 1 撤销

2.6.6. SRAM ECC 中断寄存器(SYSCFG_RAMECCIR: 18h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	BKPSRAM_DED_IE	RW	0	Backup SRAM ECC 双 bit 错误检测中断使能位 0：禁止 1：使能
8	BKPSRAM_SEC_IE	RW	0	Backup SRAM ECC 单 bit 错误纠正中断使能位 0：禁止 1：使能
7:6	RSV	-	-	保留
5	SRAM3_DED_IE	RW	0	SRAM3 ECC 双 bit 错误检测中断使能位 0：禁止 1：使能
4	SRAM3_SEC_IE	RW	0	SRAM3 ECC 单 bit 错误纠正中断使能位 0：禁止 1：使能
3:2	RSV	-	-	保留
1	SRAM1_DED_IE	RW	0	SRAM1 ECC 双 bit 错误检测中断使能位 0：禁止 1：使能
0	SRAM1_SEC_IE	RW	0	SRAM1 ECC 单 bit 错误纠正中断使能位 0：禁止 1：使能

2.6.7. SRAM ECC 状态寄存器(SYSCFG_RAMECCSR: 1Ch)

位域	名称	属性	复位值	描述

31:10	RSV	-	-	保留
9	BKPSRAM_DED_IF	RO	0	Backup SRAM ECC 双 bit 错误检测标志位，向 RAMRCCICR 寄存器中的 BKPSRAM_DED_IC 写 1，可将标志位清零 0：未发生双 bit 错误检测 1：发生双 bit 错误检测
8	BKPSRAM_SEC_IF	RO	0	Backup SRAM ECC 单 bit 错误纠正标志位，向 RAMRCCICR 寄存器中的 BKPSRAM_SEC_IC 写 1，可将标志位清零 0：未发生单 bit 错误纠正 1：发生单 bit 错误纠正
7:6	RSV	-	-	保留
5	SRAM3_DED_IF	RO	0	SRAM3 ECC 双 bit 错误检测标志位，向 RAMRCCICR 寄存器中的 SRAM3_DED_IC 写 1，可将标志位清零 0：未发生双 bit 错误检测 1：发生双 bit 错误检测
4	SRAM3_SEC_IF	RO	0	SRAM3 ECC 单 bit 错误纠正标志位，向 RAMRCCICR 寄存器中的 SRAM3_SEC_IC 写 1，可将标志位清零 0：未发生单 bit 错误纠正 1：发生单 bit 错误纠正
3:2	RSV	-	-	保留
1	SRAM1_DED_IF	RO	0	SRAM1 ECC 双 bit 错误检测标志位，向 RAMRCCICR 寄存器中的 SRAM1_DED_IC 写 1，可将标志位清零 0：未发生双 bit 错误检测 1：发生双 bit 错误检测
0	SRAM1_SEC_IF	RO	0	SRAM1 ECC 单 bit 错误纠正标志位，向 RAMRCCICR 寄存器中的 SRAM1_SEC_IC 写 1，可将标志位清零 0：未发生单 bit 错误纠正 1：发生单 bit 错误纠正

2.6.8. SRAM ECC 中断清除寄存器(SYSCFG_RAMECCICR: 20h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	BKPSRAM_DED_IC	WO	0	Backup SRAM ECC 双 bit 错误检测中断清除位 0：无影响 1：清除 BKPSRAM_DED_IF 标志位
8	BKPSRAM_SEC_IC	WO	0	Backup SRAM ECC 单 bit 错误纠正中断清除位 0：无影响 1：清除 BKPSRAM_SEC_IF 标志位
7:6	RSV	-	-	保留

5	SRAM3_DED_IC	WO	0	SRAM3 ECC 双 bit 错误检测中断清除位 0：无影响 1：清除 SRAM3_DED_IF 标志位
4	SRAM3_SEC_IC	WO	0	SRAM3 ECC 单 bit 错误纠正中断清除位 0：无影响 1：清除 SRAM3_SEC_IF 标志位
3:2	RSV	-	-	保留
1	SRAM1_DED_IC	WO	0	SRAM1 ECC 双 bit 错误检测中断清除位 0：无影响 1：清除 SRAM1_DED_IF 标志位
0	SRAM1_SEC_IC	WO	0	SRAM1 ECC 单 bit 错误纠正中断清除位 0：无影响 1：清除 SRAM1_SEC_IF 标志位

2.6.9. GPIO 5V 输出控制寄存器 1 (SYSCFG_GPIO5VOCR1: 24h)

位域	名称	属性	复位值	描述
31	PK13OE5V	RW	0	PK13 5V 输出使能位 0：禁止 1：使能
30	PK12OE5V	RW	0	PK12 5V 输出使能位 0：禁止 1：使能
29	PK11OE5V	RW	0	PK11 5V 输出使能位 0：禁止 1：使能
28	PK10OE5V	RW	0	PK10 5V 输出使能位 0：禁止 1：使能
27	PK9OE5V	RW	0	PK9 5V 输出使能位 0：禁止 1：使能
26	PK8OE5V	RW	0	PK8 5V 输出使能位 0：禁止 1：使能

25	PK7OE5V	RW	0	PK7 5V 输出使能位 0：禁止 1：使能
24	PK6OE5V	RW	0	PK6 5V 输出使能位 0：禁止 1：使能
23	PK5OE5V	RW	0	PK5 5V 输出使能位 0：禁止 1：使能
22	PK4OE5V	RW	0	PK4 5V 输出使能位 0：禁止 1：使能
21	PK3OE5V	RW	0	PK3 5V 输出使能位 0：禁止 1：使能
20	PK2OE5V	RW	0	PK2 5V 输出使能位 0：禁止 1：使能
19	PK1OE5V	RW	0	PK1 5V 输出使能位 0：禁止 1：使能
18	PK0OE5V	RW	0	PK0 5V 输出使能位 0：禁止 1：使能
17	PJ15OE5V	RW	0	PJ15 5V 输出使能位 0：禁止 1：使能
16	PJ14OE5V	RW	0	PJ14 5V 输出使能位 0：禁止 1：使能
15	PJ13OE5V	RW	0	PJ13 5V 输出使能位 0：禁止 1：使能
14	PJ12OE5V	RW	0	PJ12 5V 输出使能位 0：禁止 1：使能

13	PJ11OE5V	RW	0	PJ11 5V 输出使能位 0：禁止 1：使能
12	PJ10OE5V	RW	0	PJ10 5V 输出使能位 0：禁止 1：使能
11	PJ9OE5V	RW	0	PJ9 5V 输出使能位 0：禁止 1：使能
10	PJ8OE5V	RW	0	PJ8 5V 输出使能位 0：禁止 1：使能
9	PJ7OE5V	RW	0	PJ7 5V 输出使能位 0：禁止 1：使能
8	PJ6OE5V	RW	0	PJ6 5V 输出使能位 0：禁止 1：使能
7	PI14OE5V	RW	0	PI14 5V 输出使能位 0：禁止 1：使能
6	PH12OE5V	RW	0	PH12 5V 输出使能位 0：禁止 1：使能
5	PH11OE5V	RW	0	PH11 5V 输出使能位 0：禁止 1：使能
4	PH10OE5V	RW	0	PH10 5V 输出使能位 0：禁止 1：使能
3	PH9OE5V	RW	0	PH9 5V 输出使能位 0：禁止 1：使能
2	PH8OE5V	RW	0	PH8 5V 输出使能位 0：禁止 1：使能

1	PH7OE5V	RW	0	PH7 5V 输出使能位 0：禁止 1：使能
0	PH6OE5V	RW	0	PH6 5V 输出使能位 0：禁止 1：使能

2.6.10. GPIO 5V 输出控制寄存器 2 (SYSCFG_GPIO5VOCR2: 28h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	PP15OE5V	RW	0	PP15 5V 输出使能位 0：禁止 1：使能
16	PP14OE5V	RW	0	PP14 5V 输出使能位 0：禁止 1：使能
15	PP13OE5V	RW	0	PP13 5V 输出使能位 0：禁止 1：使能
14	PP12OE5V	RW	0	PP12 5V 输出使能位 0：禁止 1：使能
13	PP11OE5V	RW	0	PP11 5V 输出使能位 0：禁止 1：使能
12	PP10OE5V	RW	0	PP10 5V 输出使能位 0：禁止 1：使能
11	PO15OE5V	RW	0	PO15 5V 输出使能位 0：禁止 1：使能
10	PO14OE5V	RW	0	PO14 5V 输出使能位 0：禁止 1：使能

9	PO13OE5V	RW	0	PO13 5V 输出使能位 0：禁止 1：使能
8	PO12OE5V	RW	0	PO12 5V 输出使能位 0：禁止 1：使能
7	PO3OE5V	RW	0	PO3 5V 输出使能位 0：禁止 1：使能
6	PO2OE5V	RW	0	PO2 5V 输出使能位 0：禁止 1：使能
5	PO1OE5V	RW	0	PO1 5V 输出使能位 0：禁止 1：使能
4	PO0OE5V	RW	0	PO0 5V 输出使能位 0：禁止 1：使能
3	PM13OE5V	RW	0	PM13 5V 输出使能位 0：禁止 1：使能
2	PM12OE5V	RW	0	PM12 5V 输出使能位 0：禁止 1：使能
1	PM11OE5V	RW	0	PM11 5V 输出使能位 0：禁止 1：使能
0	PM10OE5V	RW	0	PM10 5V 输出使能位 0：禁止 1：使能

3. 电源管理单元（PMU）

电源管理单元用于管理芯片的电源供应及不同电源模式的切换。

3.1. PMU 主要特性

- 电源和电源域
  - VDD 电源域
  - 模拟电源域（VDDA）
  - USB 电源域（VDDUSB）
  - 内核域（Vcore）
  - 待机域（Vstandby）
- 系统电源稳压
  - 稳压器（内核域 LDO 和待机域 LDO）
- 外设电源稳压
  - eFuse 稳压器（LDO2P5）
- 电源监控
  - POR/PDR 监控器
  - BOR 监控器
  - LVD 监控器
- 电源管理
  - 工作模式
  - 电压调节控制
  - 低功耗模式

3.2. 供电电源

芯片的工作电压（VDD）2.97~3.60V。如下图所示：

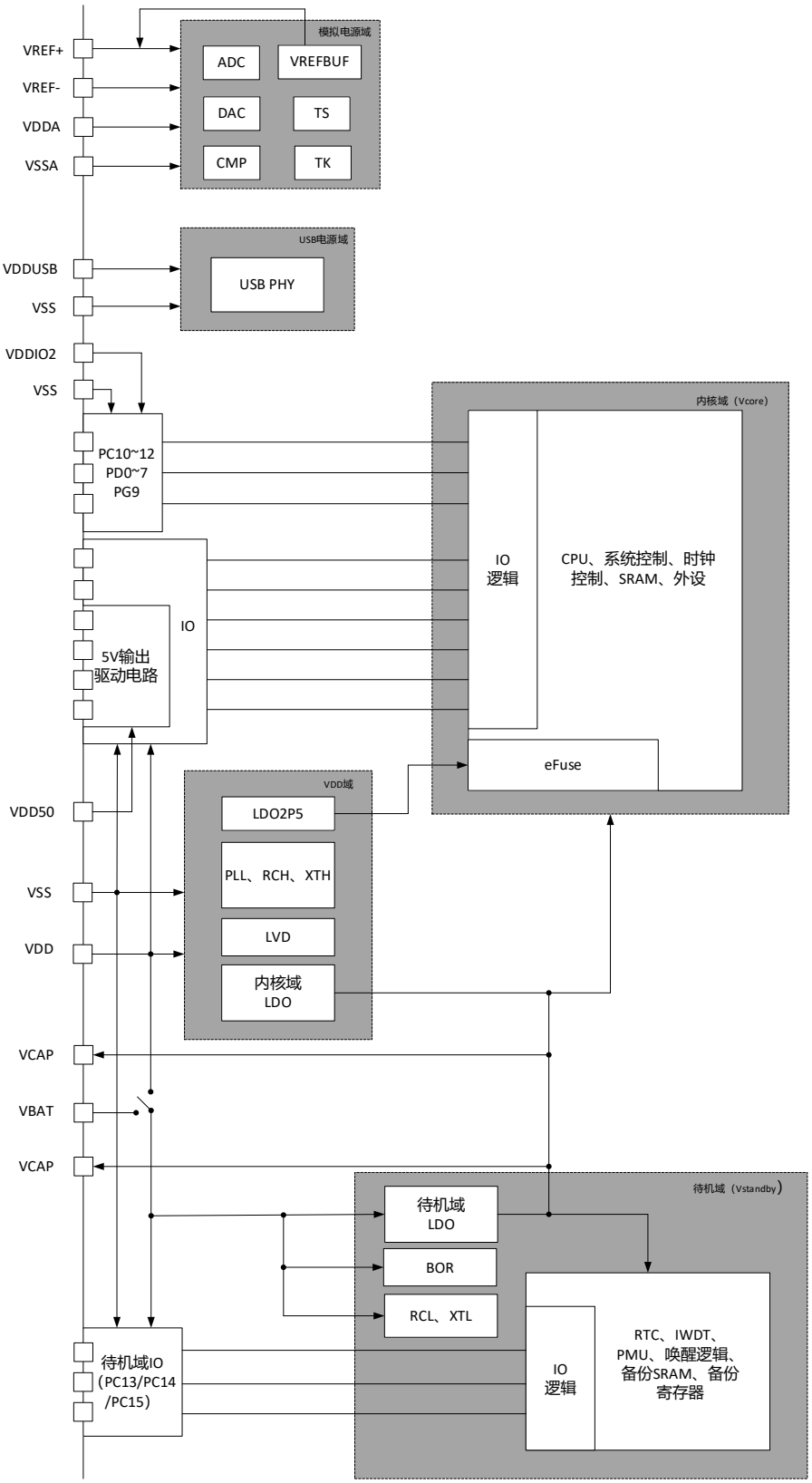
- VDD（2.97~3.60V）为 IO、稳压器、LVD、BOR、PLL、内部 RC 时钟等供电
- VDDA（2.97V~3.60V）为 ADC、DAC、TS、TKEY 和 COMP 等供电，当不使用这些外设时，最好将 VDDA 连接到 VDD 上
- VDDUSB（2.97V~3.60V）、VSSUSB 为高速收发器 USB OTG HS 供电；某些封装内部已将 VDDUSB 连接到 VDD；对于内部未连接在一起的封装，当不使用 USB 时，最好将 VDDUSB 连接到 VDD
- VDDIO2（1.62V~3.60V）为 PC[12:10] 、PD[7:0]、PG9 单独供电；当不使用这些 IO 时，VDDIO2 最好连接到 VDD
- VDD50 为 IO 5V 驱动输出电路提供供电
- VCAP 为内核域和待机域 LDO 输出，外接电容
- VBAT 为待机区供电；待机区电源由内部电源切换器来选择 VDD 供电或 VBAT 供电
- VREF+和 VREF-：VREF+为 ADC 和 DAC 的外部参考电压，同时也是内部电压参考缓冲器 VREFBUF 的输

出。

- VREF+可由外部提供、或内部 VREFBUF 提供
- VREF-连接到 VSSA
- 当 ADC 和 DAC 禁止时，VREF+可以接地

- VSSA，独立的模拟地
- VSS，芯片的公共地

图 3-1 系统电源框图



3.2.1. VDD 电源域

VDD 电压输入范围为 2.97V~3.6V，主要给 IO、LDO（内核域 LDO、待机域 LDO 和 eFuse LDO）、LVD、PLL、RCH 时钟（RCH 和 RCL）、晶振时钟（XTH 和 XTL）等供电。

另外，待机域 LDO、RCL、XTL、PC13~15，可通过内部电源切换电路选择 VDD 或 VBAT（电池）供电。如果没有外部电池的应用，建议将 VBAT 引脚通过 100nF 的外部陶瓷去耦电容连接到 VDD 引脚上。对于不带专用引脚的封装，VBAT 内部绑定到 VDD。

3.2.2. 模拟电源域

模拟电源域通过专用的 VDDA 和 VSSA 引脚供电，电压范围为 2.97V~3.6V ，给 ADC、DAC、VREFBUF、TS、CMP 和 TK 等模拟模块供电；独立的引脚供电可以单独过滤和屏蔽噪声，提高了 ADC、DAC 等模拟模块的转换效率。

3.2.3. USB 电源域

USB 收发器由单独的 VDDUSB 电源引脚供电；VDDUSB 电压范围为 2.97V~3.6V，完全独立于 VDD 和 VDDA。

当不使用 USB 时，需将 VDDUSB 连接到 VDD。

3.2.4. 内核域

Vcore 内核域电源通过稳压器提供。

■ 稳压器

稳压器提供 2 种不同的工作模式：主模式（MR）和关断模式。稳压器的工作模式将根据芯片系统工作模式（运行、停止和待机）进行使用。

- 运行模式/停止模式

稳压器工作在主模式（MR），为 Vcore 域提供全功率电压。稳压器输出电压可通过寄存器 PMU_CTRL0 的 MLDO_LV 位来调节。

- 待机模式

稳压器关闭且 Vcore 域掉电。

3.2.5. 待机域

待机域包括以下模块：

- 待机域 LDO、BOR
- RCL 时钟、XTL 低速晶振时钟
- PC13~PC15 I/O
- RTC、IWDT、PMU
- 唤醒逻辑
- 备份 SRAM
- 备份寄存器

数字电路的电源由待机域稳压器提供。待机域的数字电路主要包括 IWDT、RTC、PMU、唤醒逻辑、备份 SRAM 和备份寄存器。

待机域稳压器

待机域稳压器提供两种不同的工作模式：低功耗模式（LP）和关断模式。默认工作模式为关断模式，此时由内核域稳压器给待机域的数字电路供电。

当芯片进入低功耗待机模式时，内核域稳压器关闭，同时待机域稳压器开启（低功耗模式），待机域的数字电路由待机域稳压器供电。

当芯片从待机模式唤醒时，内核域稳压器开启，待内核域稳压器工作稳定后，待机域稳压器关闭，待机域的数字电路由内核域稳压器供电。

3.2.6. EFUSE 稳压器

EFUSE 存储模块的电源通过 EFUSE 稳压器（LDO2P5）提供，电压范围为 2.25V~2.75V。

3.3. 电源监控

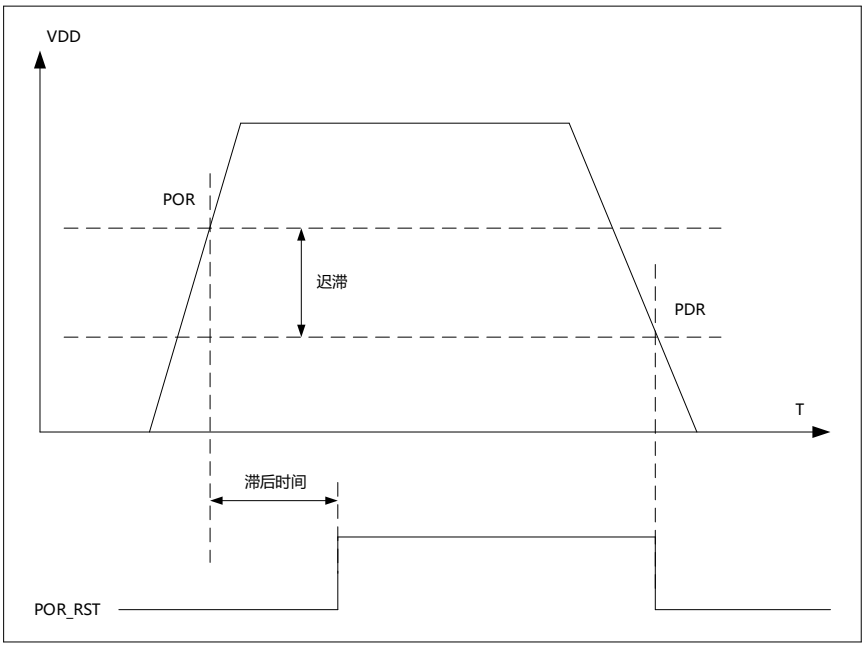
3.3.1. 上电复位(POR)/掉电复位(PDR)

内部集成 POR（上电复位）/PDR（掉电复位）电路，以确保芯片正常的启动操作。

当 VDD 电压低于 VPOR 阈值时，系统无需外部复位电路便会保持复位状态；当 VDD 电压高于 VPOR 阈值，系统便会退出复位状态，如图所示。

上电/掉电复位阈值的详细信息，请参见数据手册的电气特性部分。

图 3-2 上电复位/掉电复位波形图



3.3.2. 欠压复位(BOR)

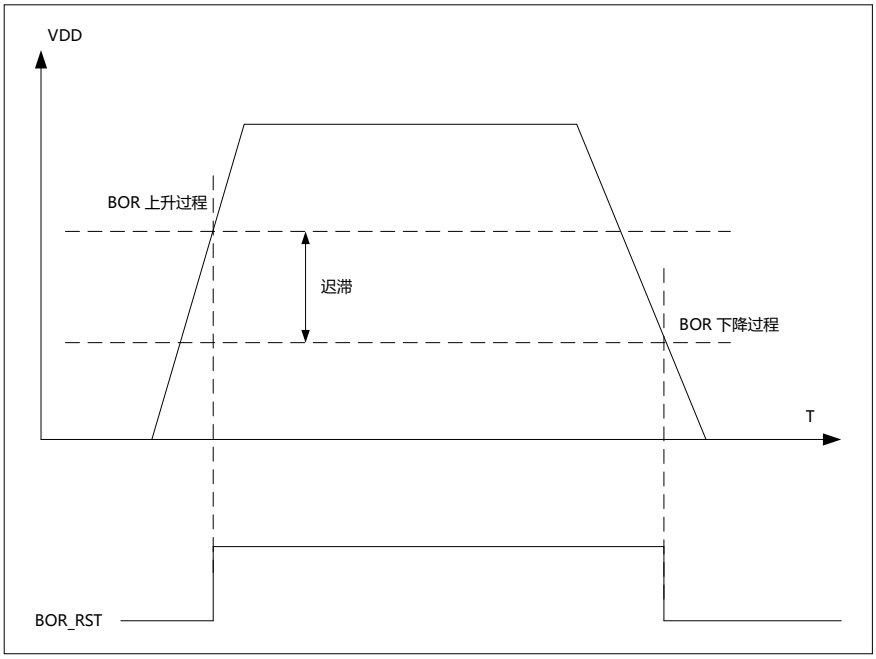
上电过程，欠电复位将芯片保持复位状态，直到 VDD 电源电压达到制定的 VBOR 阈值。

VBOR 阈值通过 PMU_CTRL2 寄存器的位 BOR_CFG 进行配置。默认情况下，BOR 关闭且 BOR 复位使能关闭。

有关欠电复位的相关详细信息，请参见产品数据手册的电气特性部分。

BOR 使能且 BOR 复位使能，当 VDD 电源电压降至所选 VBOR 阈值以下时，将产生系统复位。

图 3-3 欠电复位波形图



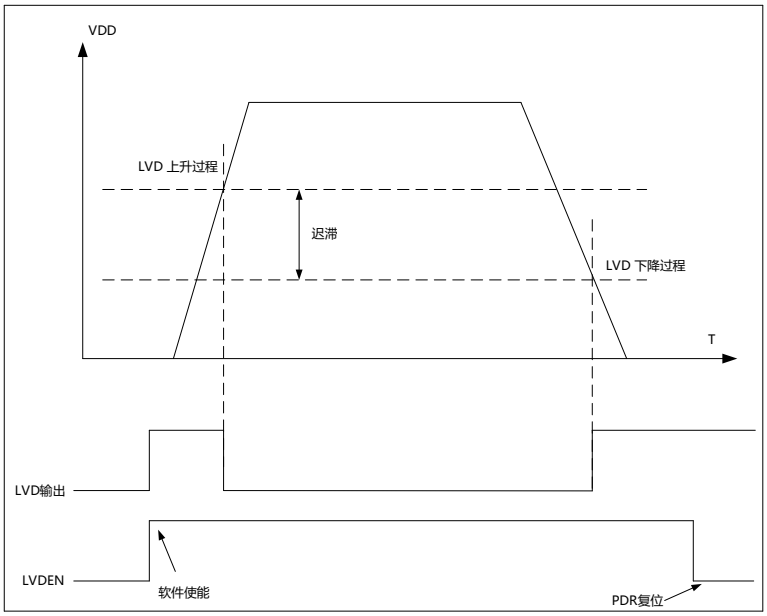
3.3.3. 电压检测模块(LVD)

LVD 用于检测 VDD 供电电压是否低于 LVD 检测阈值，检测阈值通过 PMU_CTRL1 寄存器的 LVD_SEL 位进行配置。

PMU_CTRL1 寄存器中提供了 LVDF 标志，用于指示 VDD 电压是大于还是小于 LVD 阈值。该事件内部连接到 EXTI，如果通过 EXTI 寄存器使能，则可以产生中断。该功能的用处之一就是可以在中断服务程序中执行紧急关闭系统的任务。

通过将 RCR 寄存器中的 LVDRSTEN 位置 1，当发生 LVD 事件时系统将会被复位。

图 3-4 LVD 波形图



3.4. 低功耗模式

芯片提供了多个低功耗模式，可在 CPU 不需要执行代码时节省功耗。用户可以根据应用选择合适的低功耗模式，在低功耗、短启动时间和可用唤醒源之间寻求最佳平衡。

芯片支持以下低功耗模式：

- 睡眠模式（SLEEP）：内核停止工作，外设保持工作
- 停止模式（STOP）：RCH 时钟可配置自动关闭；RCL/XTL 时钟下的外设可以工作；内核域 LDO 的输出电压范围可通过寄存器来调节
- 待机模式（STANDBY）：内核域断电，待机域 LDO 开启，待机域中的寄存器保持状态、外设可继续工作

3.4.1. SLEEP

CPU 休眠，外设不休眠，软件可以关闭各外设时钟。

SLEEP 模式	说明
进入模式	清除 NVIC 中断和事件，清除 Systick 中断 设置 SLEEPDEEP = 0 如果 SLEEPONEXIT=0，执行 WFI（等待中断）或 WFE（等待事件）指令时立即进入 SLEEP 模式 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 SLEEP 模式
退出模式	如果使用 WFI 指令或从 ISR 返回进入 SLEEP 模式，NVIC 响应的任何外设中断都能将系统从 SLEEP 模式中唤醒 如果使用 WFE 指令进入 SLEEP 模式，系统将在有事件发生时立即退出 SLEEP 模式： 如果 SEVONPEND=0，只有使能的中断或事件可以唤醒系统； 如果 SEVONPEND=1，所有的事件和中断（包括未使能的中断），都可以将系统唤醒

3.4.2. STOP

内核 LDO 工作在主模式下，可通过寄存器 PMU_CTRL0 的 MLDO_LV 调节输出电压电平；通过配置寄存器 PMU_CTRL0 的 RCHPDEN 控制进入 STOP 模式后，RCH 时钟是否自动关闭。

STOP0 模式	说明
进入模式	清除 NVIC 中断和事件，清除 Systick 中断 配置寄存器 PMU_CTRL0 的 LPMS=0 设置 SLEEPDEEP = 1 如果 SLEEPONEXIT=0，执行 WFI（等待中断）或 WFE（等待事件）指令时立即进入 STOP 模式 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 STOP 模式
退出模式	如果使用 WFI 指令或从 ISR 返回进入 STOP 模式，在 NVIC 中使能的任何 EXTI 中断都能将系统从 STOP 模式中唤醒 如果使用 WFE 指令进入 STOP 模式，系统将在有事件发生时立即退出 STOP 模式。 如果 SEVONPEND=0，在 NVIC 中使能的 EXTI 中断或 EXTI 事件可以唤醒系统； 如果 SEVONPEND=1，所有 EXTI 事件和 EXTI 中断（包括在 NVIC 中未使能的中断），都可以将系统唤醒 RSTN 管脚复位/BOR 复位/IWDT 复位(RCC_RCR 寄存器配置 IWDTRST_EN=1，即允许复位系统) /LVD 复位（RCC_RCR 寄存器配置 LVDRST_EN =1，即允许复位系统）可将系统退出 STOP 模式，唤醒后，程序重新从头开始执行，可通过查询复位标志位，来判别唤醒源

3.4.3. STANDBY

内核域掉电，待机域 LDO 开启，待机域外设保持工作。

PC13/PC14/PC15 通过 PMU_IOSEL 配置 I/O 的状态；WAKEUP 引脚通过寄存器 PMU_CTRL2 和 PMU_CTRL3 进行使能和控制。

STANDBY 模式	说明
进入模式	清除 NVIC 中断和事件，清除 Systick 中断 配置寄存器 PMU_CTRL0 的 LPMS=3’ b10x 设置 SLEEPDEEP = 1 如果 SLEEPONEXIT=0，执行 WFI（等待中断）或 WFE（等待事件）指令时立即进入 STANDBY 模式 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 STANDBY 模式
退出模式	WAKEUP 引脚的有效电平/RTC 中断/RSTN 管脚复位/IWDT 复位/BOR 复位可将系统从 STANDBY 模式中唤醒； 唤醒后，程序将重新从头开始执行

表 3-1 唤醒信号与 PIN 引脚对应关系

唤醒信号	PIN 引脚
WKUP1	PA0
WKUP2	PA2

WKUP3	PI8
WKUP4	PC13
WKUP5	PI11

3.4.4. 低功耗模式下各模块工作状态

下表为各个模块在不同功耗模式的汇总

表 3-2 各个模块在不同功耗模式下工作状态

模块	RUN	SLEEP	STOP	STANDBY
CPU	Y	-	-	X
SRAM1	Y	Y	Y	X
SRAM2	Y	Y	Y	X
ITCM/DTCM	Y	Y	Y	X
eFuse	Y	Y	Y	X
备份 SRAM	Y	Y	Y	Y
备份寄存器	Y	Y	Y	Y
UART/I2C/SPI/I2S	O	O	-	X
LPUART	O	O	O	X
Timer/WDT	O	O	-	X
LPTIM	O	O	O	X
UAC（算法）	O	O	-	X
IWDT	O	O	O	O
USB	O	O	O	X
PLL	O	O	-	X
RCH	O	O	-	X
RCL	O	O	O	O
XTH	O	O	-	X
XTL	O	O	O	O
ADC	O	O	-	X
DAC	O	O	-	X
CMP	O	O	O	X
LVD	O	O	O	X
MRLDO12	Y	Y	Y	X
LPLDO12	Y	Y	Y	Y
RTC	O	O	O	O
GPIO	Y	Y	Y	Y 注 2

注 1：Y：使能工作；O：可选择使能或禁止工作；-：停止工作；X：掉电

注 2：除 PC13/PC14/PC15 外的所有 GPIO 均掉电（浮空状态）；PC13/PC14/PC15 可通过 PMU_IOSEL 配置保持或运行状态；5 个 WAKEUP 引脚（WKUP1~WKUP5）可作为唤醒源

3.5. PMU 寄存器描述

3.5.1. 寄存器列表

PMU 寄存器基地址：0x40007000

偏移	名称	复位值	描述
0x00	PMU_CTRL0	0x03c04000	PMU 控制寄存器 0
0x04	PMU_CTRL1	0x00000000	PMU 控制寄存器 1
0x08	PMU_CTRL2	0x000a0000	PMU 控制寄存器 2
0x0C	PMU_CTRL3	0x00000000	PMU 控制寄存器 3
0x10	PMU_SR	0x00010000	PMU 状态寄存器
0x14	PMU_STCLR	0x00000000	PMU 状态清除寄存器
0x18	PMU_IOSEL	0x00000000	STANDBY 域 IO 复用寄存器
0xC0	TEST_ANATEST_SR	0x00000000	ANATEST 选择寄存器（客户不开放）
0xC4	TEST_LDOCAL	0x00004844	LDO 校准寄存器 （客户不开放）
0xC8	TEST_LDOCR	0x00000000	LDO 测试控制寄存器（客户不开放）

3.5.2. PMU 控制寄存器 0(PMU_CTRL0: 00h)

位域	名称	属性	复位值	描述
31	BKPRAMREN	RW	0	BKP SRAM 保持状态使能位 0：BKP SRAM 不进入保持状态 1：BKP SRAM 进入保持状态
30	BKPRAMSEN	RW	0	BKP SRAM 睡眠状态使能位 0：BKP SRAM 不进入睡眠状态 1：BKP SRAM 进入睡眠状态（数据丢失） 注：当 BKPRAMREN 和 BKPRAMSEN 同时置 1 时，BKP SRAM 会进入睡眠状态，及 BKP SRAM 进入睡眠模式的优先级较高
29:28	RSV	-	-	保留
27:16	STOP_WPT	RW	0x3C0	STOP 唤醒时间设置 从 STOP 模式唤醒所需的时钟周期数(若使能 RC64M_DIV_EN 位，以 RC64M 时钟 16 分频计数；否则以 RC64M 时钟计数)
15	RSV	-	-	保留

14:12	LPLDO12_LV	RW	100	STANDBY 模式下，LPLDO12 调整选择 100：调整为 1.1V（不变） 其他：保留
11:10	RSV	-	-	保留
9:8	MLDO12_LV	RW	00	STOP 模式下，MLDO12 电压是否自动调整 00：保持为 1.2V 01：调整为 1.0V 10：调整为 0.9V 11：调整为 0.8V
7	RTC_WE	RW	0	RTC 模块写使能 0: RTC 模块写禁止 1: RTC 模块写使能
6	RSV	-	-	保留
5	RCH_DIV_EN	RW	0	STOP 模式下，RCH 选择 16 分频输出 0：选择原始不分频输出 1：选择 16 分频输出
4	RCHPDEN	RW	0	STOP 模式下，RCH 是否自动关闭 0：RCH 不自动关闭 1：RCH 自动关闭 注：PLL、RCL、XTH、XTL 不会自动关闭，需软件控制
3:1	RSV	-	-	保留
0	LPMS	RW	0	低功耗模式选择位（内核进 DeepSleep 时有效）： 0：STOP 模式 1：STANDBY 模式

3.5.3. PMU 控制寄存器 1(PMU_CTRL1: 04h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	LVD_VALUE	RO	0	LVD 原始状态
14	LVD_FILTER	RO	0	LVD Filter 后的状态
13:12	RSV	-	-	保留

11:9	FLT_TIME	RW	000	LVD 滤波时间配置，滤波时间按 FiltClk 计算，FiltClk 由系统寄存器 CCR2 的 BIT15 进行控制。 000：1 个周期 001：2 个周期 010：4 个周期 011：16 个周期 100：64 个周期 101：256 个周期 110：1024 个周期 111：4095 个周期
8	LVD_FLTEN	RW	0	数字滤波使能配置 0：禁止数字滤波 1：使能数字滤波 如果在 STOP 模式用 LVD 唤醒，需要禁止数字滤波或者使用 RCL 时钟滤波
7:4	RSV	-	-	保留
3:1	LVD_SEL	RW	0000	LVD 阈值电压选择，下列电压值为触发报警的典型值，解除报警电压值为所列电压值+100mV。 000：1.71 001：2.01 010：2.23 011 ： 2.43 100：2.51 101：2.73 110 ： 2.80 111 ： 2.90 其它：预留
0	LVDEN	RW	0	LVD 使能控制 0：禁止 LVD 1：使能 LVD 注：中断功能在 EXTI 中设置，复用功能在 RCR 中设置。滤波后为高就产生复位。 使能后，LVD 模拟需要 2us 的稳定时间，在使能 2us 内可能误报警。

3.5.4. 电源控制寄存器 2(PMU_CTRL2: 08h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:26	BOR_CFG	RW	00	BOR 电压选择 00：2.0V 到 2.1V 电压范围的复位阈值 01：2.2V 到 2.3V 电压范围的复位阈值 10：2.49V 到 2.61V 电压范围的复位阈值 11：2.77V 到 2.90V 电压范围的复位阈值
25	RSV	-	-	保留

24	BOR_EN	RW	0	BOR 使能 0：禁止 BOR 1：使能 BOR 注: BOR 使能后，1us 后输出稳定，可以设置 BOR 复位使能（BORSTN_EN）。关闭 BOR 前，请先关闭 BORRST_EN。
23:21	RSV	-	-	保留
20	BORRST_EN	RW	0	BOR 复位使能 0：BOR 复位不使能 1：BOR 复位使能
19:16	STDBY_WPT	RW	0xa	STANDBY 模式唤醒等待寄存器 0000：无效，不等待 0001：等待 1 个 RC32K 0010：等待 2 个 RC32K … 1110：等待 14 个 RC32K 1111：等待 15 个 RC32K
15:13	RSV	-	-	保留
12:8	WUXFILEN	RW	0	WKUPx(x=5~1)管脚滤波使能，使用 RTCCLK 滤波，滤波 1 拍毛刺 0：WKUPx 管脚滤波不使能 1：WKUPx 管脚滤波使能
7:5	RSV	-	-	保留
4:0	EWUPX	RW	0	WKUPx(x=5~1)管脚唤醒功能使能 0：WKUPx 管脚唤醒功能不使能 1：WKUPx 管脚唤醒功能使能，滤波功能由 WUxFIL 位选择。

3.5.5. 电源控制寄存器 3 (PMU_CTRL3: 0Ch)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	WUPOLX	RW	0	WKUPx(x=5~1)管脚唤醒极性选择 0：WKUPx 管脚高电平唤醒 1：WKUPx 管脚低电平唤醒

3.5.6. PMU 状态寄存器(PMU_SR: 10h)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	BORN	R	1	BORN 的原始状态

15	BORWUF	RO	0	BOR 唤醒标志。通过写 CBORWUF 位（PMU_STCLR 寄存器）清除 0：没有 BOR 唤醒事件 1：BOR 发生唤醒事件 BORWUF 不清除，会引起 STANDBY 模式立刻唤醒
14	IWDTWUF	RO	0	IWDT 唤醒标志。通过写 CIWDTWUF 位（PMU_STCLR 寄存器）清除 0：没有 IWDT 唤醒事件 1：IWDT 发生唤醒事件 IWDTWUF 不清除，会引起 STANDBY 模式立刻唤醒
13	RSTWUF	RO	0	RSTN 唤醒标志。通过写 CRSTWUF 位（PMU_STCLR 寄存器）清除 0：没有 RSTN 唤醒事件 1：RSTN 发生唤醒事件 RSTWUF 不清除，会引起 STANDBY 模式立刻唤醒
12	RTCWUF	RO	0	RTC 唤醒标志。通过写 CRTCWUF 位（PMU_STCLR 寄存器）清除 0：没有 RTC 唤醒事件 1：RTC 发生唤醒事件 RTCWUF 即使不清除，也不会引起 STANDBY 模式唤醒
11:9	RSV	-	-	保留
8	SBF	RO	0	STANDBY 标志。当芯片进入 STANDBY 模式时，硬件自动设置，只能被 POR 或者写 CSBF 位（PMU_STCLR 寄存器）清除 0：芯片未进入过 STANDBY 模式 1：芯片进入过 STANDBY 模式
7:5	RSV	-	-	保留
4:0	WUPFX	RO	0	WKUPx(x=5~1)唤醒标志。WKUPx 管脚检测到一个唤醒事件，可以通过写 CWUFx 位（PMU_STCLR 寄存器）清除 0：没有 WKUPx 唤醒事件 1：WKUPx 管脚发生唤醒事件 注：WUPFx 标志如果为高，进入 STANDBY 后会立刻唤醒

3.5.7. PMU 状态清除寄存器(PMU_STCLR: 14h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	CBORWUF	WO	0	清除唤醒标志 BORWUF 标志。读始终为 0，写 1 清除唤醒标志 0：没有作用 1：清除唤醒标志

14	CIWDTWUF	WO	0	清除唤醒标志 IWDTWUF 标志。读始终为 0，写 1 清除唤醒标志 0：没有作用 1：清除唤醒标志
13	CRSTWUF	WO	0	清除唤醒标志 RSTWUF 标志。读始终为 0，写 1 清除唤醒标志 0：没有作用 1：清除唤醒标志
12	CRTCWUF	WO	0	清除唤醒标志 RTCWUF 标志。读始终为 0，写 1 清除唤醒标志 0：没有作用 1：清除唤醒标志
11:9	RSV	-	-	保留
8	CSBF	WO	0	清除 STANDBY 标志。读始终为 0，写 1 清除 STANDBY 标志 0：没有作用 1：清除 STANDBY 标志
7:5	RSV	-	-	保留
4:0	CWUFX	WO	0	清除唤醒标志 WUPFx(x=5~1)。读始终为 0，写 1 清除唤醒标志 0：没有作用 1：清除唤醒标志

3.5.8. STANDBY 域 IO 复用寄存器(PMU_IOSEL: 18h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	PI8_VALUE	RW	0	PI8_Value，具体见 PI8_SEL，注：读取的值为之前写入的值。
10	PC15_VALUE	RW	0	PC15_Value，具体见 PC15_SEL，注：读取的值为之前写入的值。
9	PC14_VALUE	RW	0	PC14_Value，具体见 PC14_SEL，注：读取的值为之前写入的值。
8	PC13_VALUE	RW	0	PC13_Value，具体见 PC13_SEL，注：读取的值为之前写入的值。
7:6	PI8_SEL	RW	0	PI8 管脚功能选择 00：主区 GPIO 功能；STANDBY 模式下为输入状态 01：输出 PI8_Value，推挽输出 10：RTC 输入，作为时间戳功能（Tamper） 11：输出关闭

5:4	PC15_SEL	RW	0	PC15 管脚功能选择 00：主区 GPIO 功能；STANDBY 模式下为输入状态 01：输出 PC15_Value，推挽输出 其它：输出关闭 注：需要选择 XTL_OUT，请配置为模拟端口
3:2	PC14_SEL	RW	0	PC14 管脚功能选择 00：主区 GPIO 功能；STANDBY 模式下为输入状态 01：输出 PC14_Value，推挽输出 其它：输出关闭 注：需要选择 XTL_IN，请配置为模拟端口
1:0	PC13_SEL	RW	0	PC13 管脚功能选择 00：主区 GPIO 功能；STANDBY 模式下为输入状态 01：RTC Fout，选择 RTC 的 FSEL_OUT 输出；同时 PC13_Value 作为输出模式选择位： 0：开漏输出 1：推挽输出 10：RTC 输入，作为时间戳功能（Tamper） 11：输出 PC13_Value，推挽输出

注：若 PMU_IOSEL 寄存器配置成非 GPIO 功能，则以 PMU_IOSEL 寄存器为最高优先级，此时 GPIO 寄存器对 PC13、PC14、PC15、PI8 的配置将会忽略。

3.5.9. ANATEST 选择寄存器(TEST_ANATEST_SR: C0h) (客户不开放)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	ANATEST_SEL	RW	000	模拟 ANATEST 信号选择 ANA_SEL<2:0>定义如下 000：浮空 111：VBG_LP(LPBGR_TRIM) 110：VRF_LP 101：VDD25 100：VREF1V 011：VDD12(LDO12_TRIM) 010：VDD12_RET(LPLDO12_TRIM) 001：VREFINT

3.5.10. LDO 校准寄存器(TEST_LDOCAL: C4h) (客户不开放)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14:12	LPLDO12_TRIM	RW	100	LPLDO12 电压 TRIM 默认 100：1.1v
11:8	MLDO12_TRIM	RW	1000	MLDO12 电压 TRIM
7	RSV	-	-	保留
6:4	LPBGR_TRIM	RW	100	LPBGR 的 TRIM 值
3	RSV	-	-	保留
2:0	VREF_TRIM	RW	100	基准电压 Trim

3.5.11. LDO 测试控制寄存器(TEST_LDOCR: C8h) (客户不开放)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	TEST_LPLDO12_EN	RW	0	软件模式下，LPLDO12 使能 0：不使能 1：使能
8:6	RSV	-	-	保留
5:4	TEST_MLDO12_LV	RW	0	MLDO12 电压选择 00：保持为 1.2V 01：调整为 1.0V 10：调整为 0.9V 11：调整为 0.8V
3:1	RSV	-	-	保留
0	LDO_TEST_EN	RW	0	LDO 测试模式使能位 0：禁止 1：使能

4. 复位和时钟控制(RCC)

RCC 模块管理整个芯片的时钟和复位信号的生成。

4.1. 复位

RCC 支持三种类型的复位：

- 电源复位
- 系统复位
- 待机域复位

4.1.1. 电源复位

复位源	描述
上电复位（POR）	当芯片供电电压 VDD 高于阈值电压时，产生上电复位；复位整个芯片
掉电复位（PDR）	当芯片供电电压 VDD 低于阈值电压时，产生掉电复位；复位整个芯片
内核域 LDO 上电复位（POR12）	当芯片冷启动时，内核域 LDO 上电产生该复位；复位内核域和待机域； 当芯片从 STANDBY 模式唤醒时，内核域 LDO 上电产生该复位；复位内核域，不复位待机域
欠压复位（BOR）	当芯片供电电压 VDD 低于可编程的阈值电压时，产生欠压复位；复位内核域和 IWDT 除待机域中的 IWDT 模块外，不复位待机域的其他模块； 芯片处于 STANDBY 低功耗模式时，BOR 复位唤醒后，复位情况同 POR12

注：电源域的划分详见 PMU 章节

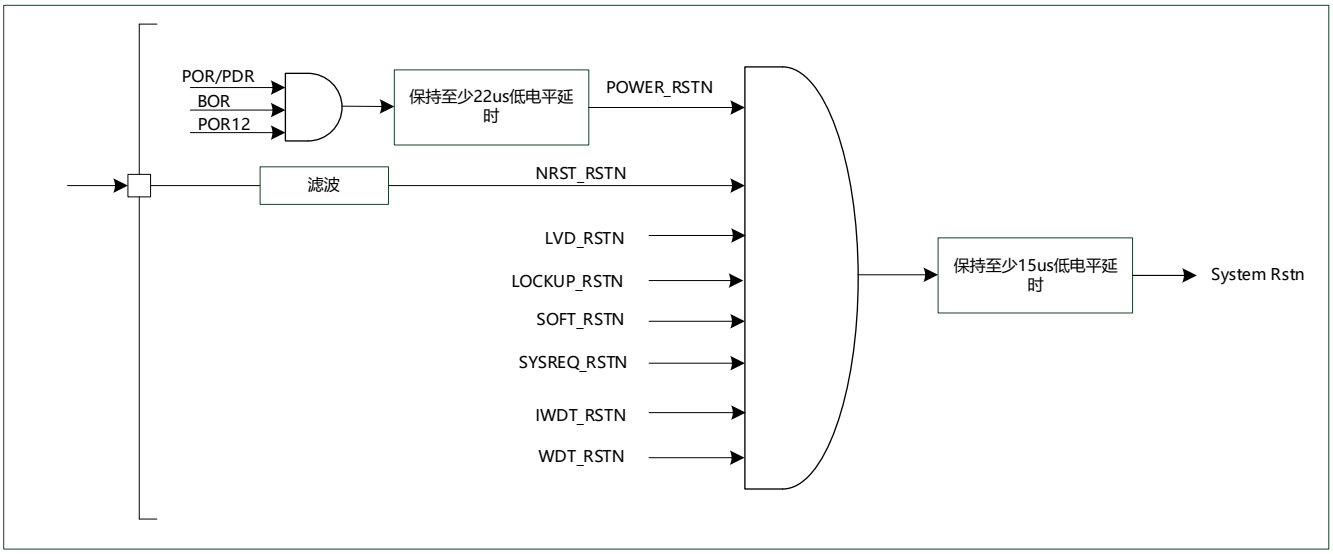
4.1.2. 系统复位

复位源	描述
外部管脚复位（NRST_RSTN）	复位信号来自于外部复位引脚，低电平有效（最大滤除脉宽 80ns）；复位内核域（包括 RCC 寄存器和 JTAG）； 不复位待机域； 芯片处于 STANDBY 低功耗模式时，NRST 复位唤醒后，复位情况同 POR12
LVD_RSTN	低压检测复位（使能寄存器 RCC_RCR 中的 LVDRSTEN 位）；复位内核域，不复位 RCC 寄存器； 不复位待机域；
LOCKUP_RSTN	CPU LOCKUP 复位（使能寄存器 RCC_RCR 中的 LOCKRSTEN 位）；复位内核域，不复位 RCC 寄存器； 不复位待机域；

SOFT_RSTN	软件复位；复位内核域，不复位 RCC 寄存器； 不复位待机域；
SYSREQ_RSTN	CPU SYSREQ 复位；复位内核域，不复位 RCC 寄存器； 不复位待机域；
IWDT_RSTN	独立看门狗复位（使能寄存器 RCC_RCR 中的 IWDTRSTEN 位）；复位内核域和 IWDT 模块，不复位 RCC 寄存器； 除待机域的 IWDT 模块外，不复位待机域的其他模块； 芯片处于 STANDBY 低功耗模式时，IWDT 复位唤醒后，除复位 IWDT 模块外，复位情况同 POR12
WDT_RSTN	系统看门狗复位（使能寄存器 RCC_RCR 中的 WDTRSTEN 位）；复位内核域，不复位 RCC 寄存器； 不复位待机域；

注：电源域的划分详见 PMU 章节

图 4-1 系统复位电路



4.1.3. 待机域复位

复位源	描述
待机域软复位	待机域软复位可由设置 STANDBY 电源域控制寄存器（RCC STDBY_CTRL）中的 STDBYRST 位产生 复位待机域（除待机域的 IWDT 模块、备份 SRAM 及 RCC STDBY_CTRL 寄存器）

注：电源域的划分详见 PMU 章节

4.2. 时钟

RCC 有多种时钟源可供选择:

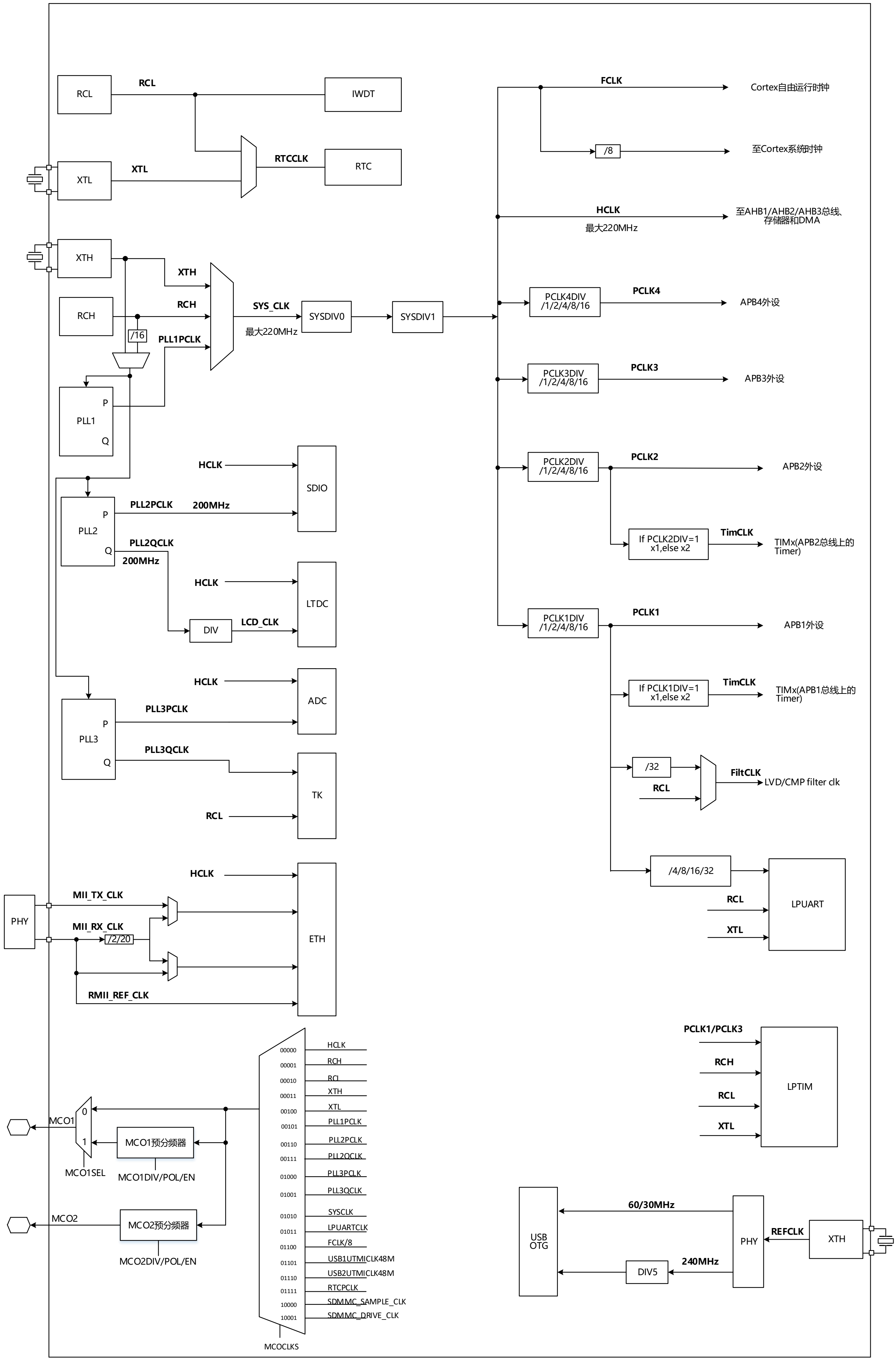
- 内部振荡器：
  - RCH（内部高速 RC 振荡器）时钟：64MHz
  - RCL（内部低速 RC 振荡器）时钟：32KHz
- 外部晶体振荡器：
  - XTH（外部高速晶体振荡器）时钟：4~32MHz
  - XTL（外部低速晶体振荡器）时钟：32.768KHz
- PLL（锁相环）时钟：
  - PLL1：主 PLL，用于为系统及部分外设提供高速时钟
  - PLL2 和 PLL3：专用 PLL，用于生成精确时钟，为特定外设提供时钟

对于每个时钟源来说，在未使用时都可单独打开或者关闭，以降低功耗。

除以下外设外，其他所有外设时钟均由系统时钟（SYSCLK）提供：

- 由专用的 PLL2 提供 SDMMC 时钟（200MHz）
- 由专用的 PLL2 提供像素时钟 LCD_CLK
- 由专用的 PLL3 提供 ADC 时钟
- 由专用的 PLL3 提供 TKEY 时钟
- 由内部 PHY 提供 USB OTG HS 时钟（60MHz）及 USB OTG FS 时钟（48MHz）
- 由外部 PHY 提供的以太网 MAC 时钟（TX、RX 和 RMII）

图 4-2 时钟结构图


RCL
IWDT
XTL
RTCCLK
RTC
XTH
RCH
/16
SYS_CLK
最大220MHz
SYSDIV0
SYSDIV1
PLL1PCLK
PLL1
P
Q
FCLK
Cortex自由运行时钟
/8
至Cortex系统时钟
HCLK
最大220MHz
至AHB1/AHB2/AHB3总线、存储器和DMA
PCLK4DIV /1/2/4/8/16
PCLK4
APB4外设
PCLK3DIV /1/2/4/8/16
PCLK3
APB3外设
PCLK2DIV /1/2/4/8/16
PCLK2
APB2外设
If PCLK2DIV=1 x1,else x2
TimCLK
TIMx(APB2总线上的Timer)
PCLK1DIV /1/2/4/8/16
PCLK1
APB1外设
If PCLK1DIV=1 x1,else x2
TimCLK
TIMx(APB1总线上的Timer)
/32
FiltCLK
LVD/CMP filter clk
/4/8/16/32
LPUART
SDIO
PLL2PCLK
200MHz
PLL2
PLL2QCLK
LTDC
DIV
LCD_CLK
ADC
PLL3PCLK
PLL3
PLL3QCLK
TK
ETH
PHY
MII_TX_CLK
MII_RX_CLK
/2/20
RMII_REF_CLK
PCLK1/PCLK3
LPTIM
MCO1
MCO1预分频器
MCO1SEL
MCO1DIV/POL/EN
MCO2
MCO2预分频器
MCO2DIV/POL/EN
00000 HCLK
00001 RCH
00010 RCL
00011 XTH
00100 XTL
00101 PLL1PCLK
00110 PLL2PCLK
00111 PLL2QCLK
01000 PLL3PCLK
01001 PLL3QCLK
01010 SYSCLK
01011 LPUARTCLK
01100 FCLK/8
01101 USB1UTMICLK48M
01110 USB2UTMICLK48M
01111 RTCPCLK
10000 SDMMC_SAMPLE_CLK
10001 SDMMC_DRIVE_CLK
MCOCLKS
USB OTG
60/30MHz
240MHz
DIV5
REFCLK

注 1：SYSDIV0/SYSDIV1/PCLK1DIV/PCLK2DIV/PCLK3DIV/PCLK4DIV 等分频配置详见 CCR2 寄存器

注 2：AHB1/AHB2/AHB3/APB1/APB2/APB3/APB4 外设详见系统架构图或外设存储空间映射表

注 3：有关内部和外部时钟源特性的所有详细信息，请参见器件数据手册的电气特性部分

AHB1/AHB2/AHB3 的时钟频率与系统频率一致最高 220MHz，APB1/APB2/APB3/APB4 的时钟频率由系统频率分频而来，分频比可独立配置（当不分频时，APB1/APB2/APB3/APB4 的时钟频率最高也可达 220MHz)。

RCC 通过 FCLK 时钟的 8 分频作为 Cortex 系统定时器（SysTick)的外部时钟；SysTick 可使用此时钟作为时钟源，也可使用 FCLK 作为时钟源，具体可在 SysTick 控制和状态寄存器中配置。

定时器时钟频率分配由硬件按以下 2 种情况自动设置：

- 如果相应的 APB 预分频系数是 1，定时器的时钟频率与所在 APB 总线频率一致。
- 否则，定时器的时钟频率被设为与其相连的 APB 总线频率的 2 倍。

4.2.1. RCH 时钟

RCH 时钟由内部 64MHz RC 振荡器产生，可直接用作系统时钟，或者用作 PLL 输入（RCH 的 16 分频模式)。RCH 时钟可由寄存器 RCC_RCHCR 的 RCHEN 位来启动和关闭。

不同芯片的 RC 振荡器频率不同，因此会对每个器件进行出厂校准，校准精度达到 TA=25℃时 1.5%的精度。

系统复位时，工厂校准值被装载到寄存器 RCC_RCHCR 的 RCHTRIM 位。

如果用户的应用基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。可以通过寄存器 RCC_RCHCR 的 RCHTRIM 位来调整 RCH 的频率。

寄存器 RCC_RCHCR 的 RCHRDY 位用来指示 RCH 振荡器是否稳定。如果在时钟中断寄存器 RCC_CIR 中允许产生中断，将会产生相应的中断。

RCH 时钟还可作为备份时钟源使用，以防 XTH 晶振发生故障。

4.2.2. RCL 时钟

RCL 时钟由内部 RC 振荡器产生，可作为低功耗时钟源在停机和待机模式下保持运行，供独立看门狗(IWDG)、RTC 和 PMU 使用。时钟频率在 32KHz 左右。

RCL 时钟硬件固定使能，不能由软件关闭；但在 STANDBY 低功耗模式下，可配置寄存器 RCC_STDBYCTRL 中的 RCLEN 位是否关闭。

寄存器 RCC_STDBYCTRL 中的 RCLRDY 标志指示 RCL 振荡器是否稳定。如果在时钟中断寄存器 RCC_CIR 中允许产生中断，将会产生相应的中断。

不同芯片的 RCL 振荡器频率不同，因此会对每个器件进行出厂校准，校准精度达到 TA=25℃时±3%的精度。

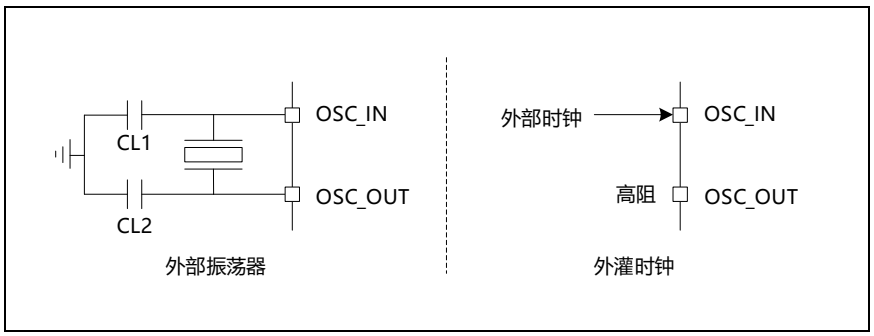
系统复位时，工厂校准值被装载到寄存器 RCC_STDBYCTRL 的 RCLTRIM 位。

如果用户的应用基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。可以通过寄存器 RCC_STDBYCTRL 的 RCLTRIM 位来调整 RCL 的频率。

4.2.3. XTH 时钟

外部高速晶体振荡器 XTH 支持 4~32MHz，可直接用作系统时钟，或者用作 PLL 输入；XTH 有 2 个时钟源：

- XTH 外部晶振/陶瓷谐振器
- XTH 外灌输入时钟



■ XTH 外部晶振/陶瓷谐振器

将寄存器 RCC_XTHCR 中的 XTHEN 位置 1 且 XTHBYP 置 0，XTH 外部晶体振荡器将被启动。在寄存器 RCC_XTHCR 中的 XTHRDY 标志指示外部高速晶体振荡器是否稳定。如果在时钟中断寄存器 RCC_CIR 中允许产生中断，将会产生相应的中断。

外部振荡器和负载电容尽可能地靠近振荡器的引脚，以减小输出失真和起振稳定时间。

负载电容值必须根据不同的振荡器做相应调整。

■ XTH 外灌输入时钟

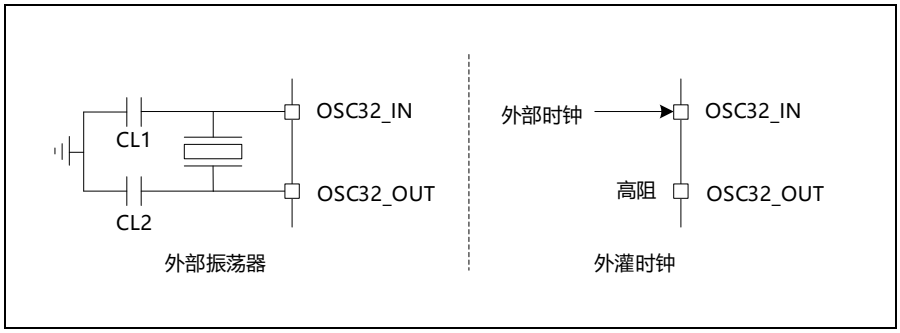
外灌时钟时，将寄存器 RCC_XTHCR 中的 XTHBYP 和 XTHEN 位置 1 进行选择。必须使用占空比约为 50%的外部时钟信号（方波、正弦波或三角波）来驱动振荡器输入引脚 OSC_IN，同时输出引脚 OSC_OUT 保持高阻状态 。

4.2.4. XTL 时钟

外部低速振荡器 XTL 是一个 32.768KHz 的低速外部晶体或陶瓷谐振器，它为实时时钟或者其他定时功能提供一个低功耗且精确的时钟源。

XTL 有 2 个时钟源：

- XTL 外部晶振/陶瓷谐振器
- XTL 外灌输入时钟



■ XTL 外部晶振/陶瓷谐振器

将寄存器 RCC_STDBYCTRL 中的 XTLEN 位置 1 且 XTLBYP 置 0，XTL 外部晶体振荡器将被启动。在寄存器 RCC_STDBYCTRL 中的 XTLRDY 标志指示外部低速晶体振荡器是否稳定。如果在时钟中断寄存器 RCC_CIR 中允许产生中断，将会产生相应的中断。

■ XTL 外灌输入时钟

外灌时钟时，将寄存器 RCC_STDBYCTRL 中的 XTLBYP 和 XTLEN 位置 1 进行选择。必须使用占空比约为 50%的外部时钟信号（方波、正弦波或三角波）来驱动振荡器输入引脚 OSC32_IN，同时输出引脚 OSC32_OUT 保持高阻状态 。

4.2.5. PLL 时钟

内置 PLL（锁相环时钟）可以用来倍频 RCH 或 XTH 的输出时钟。

RCC 具有三个 PLL：

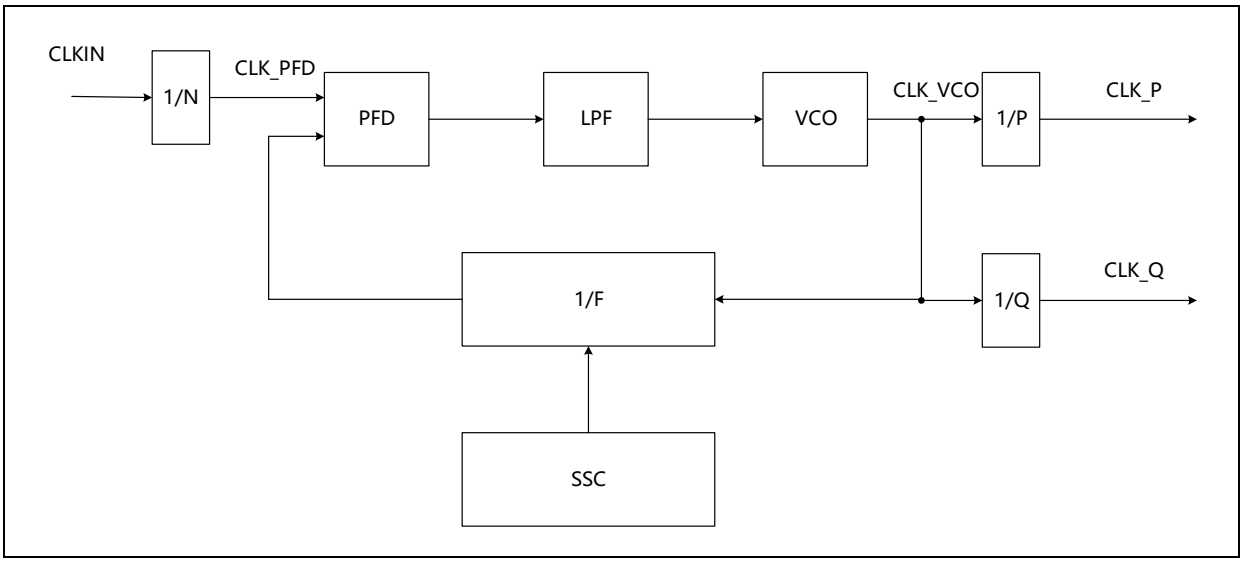
- PLL1：主 PLL，用于为系统及部分外设提供高速时钟
- PLL2 和 PLL3：专用 PLL，用于生成精确时钟，为特定外设提供时钟

PLL1 和 PLL2 具有以下特性：

- 时钟输入（CLKIN）频率范围：1MHz~132MHz
- PFD 输入（CLK_PFD）频率范围：1MHz~2MHz
- VCO 输出（CLK_VCO）频率范围：100MHz~550MHz
- 时钟输出（CLK_P）频率范围：30MHz~275MHz
- 时钟输出（CLK_Q）频率范围：16MHz~550MHz
- 支持扩频时钟（SSC，Spread Spectrum Clocking）功能

PLL1/PLL2 功能框图如下：

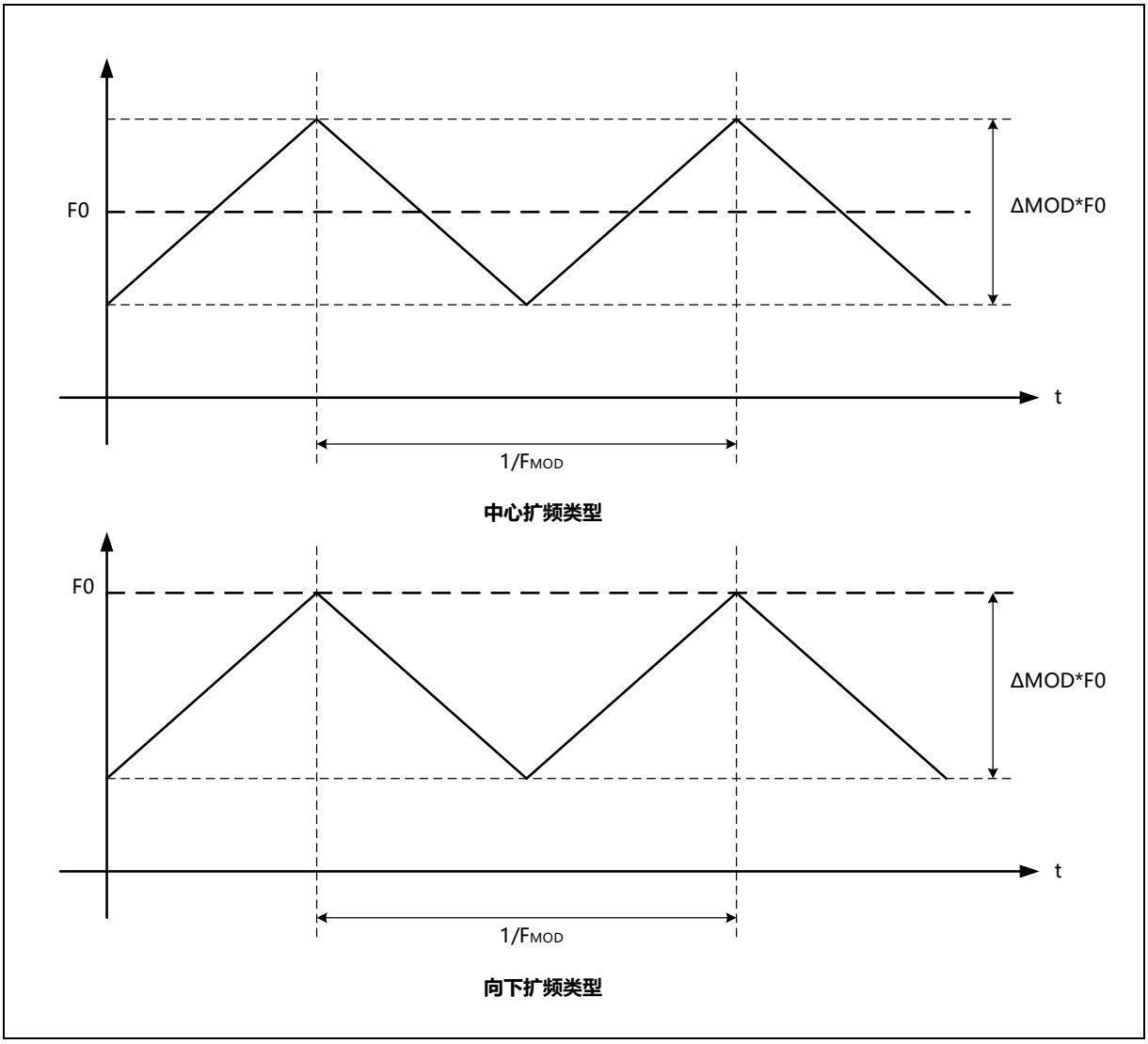
图 4-3 PLL1/PLL2 功能框图



■ 时钟扩频（SSC）

使能时钟扩频（SSC）功能后，PLL1/PLL2 将工作在小数模式下。

时钟扩频类型分为中心扩频和向下扩频两种类型，如下图所示



图中，FMOD 为调制频率，即频率扩展周期率。一般调制频率较低，可实现频率平滑调制，减小调制后的时钟抖动；FMOD 计算公式如下：

FMOD=FCLK_PFD /(2* PLLxSSCPER)

调制频率小于 10KHz，且 PLLxSSCPER 必须大于 48。

▵MOD 为扩频比，即频率抖动范围；较大的扩频比可以加强 EMI 的衰减程度，但也可能会高于系统最大额定频率或低于平均频率而对系统造成影响，因此，要保证扩频比在 0.25%~2%范围内。

▵MOD 计算公式如下：

▵MOD =(PLLxSSCPER *PLLxSSCSTP)/(2^15-1)/ PLLxF

注 1：(PLLxSSCPER *PLLxSSCSTP)/(2^15-1)≤1，且 PLLxSSCSTP 小于 682

注 2：PLLXSSCPER 尽量设置大一些，否则扩频后频率变化比较小

■ PLL3 具有以下特性：

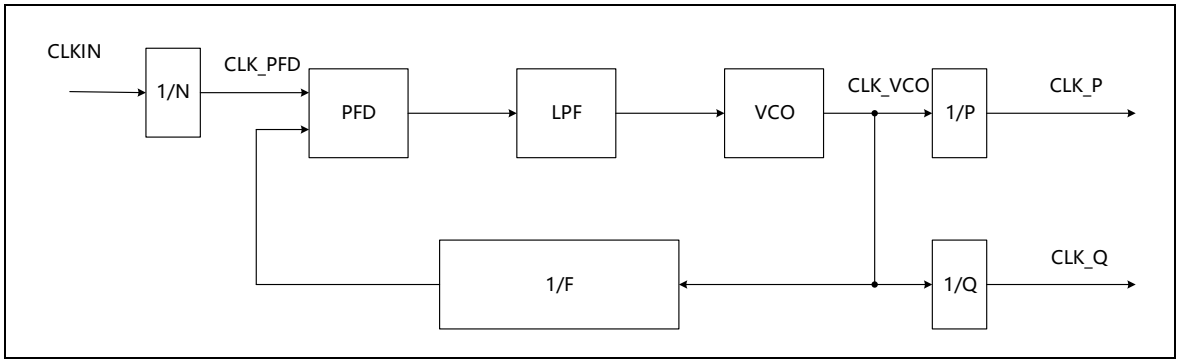
- PFD 输入（CLK_PFD）频率范围：1MHz~50MHz
- VCO 输出（CLK_VCO）频率范围：200MHz~500MHz

➢ 时钟输出（CLK_P）频率范围：25MHz~500MHz

➢ 时钟输出（CLK_Q）频率范围：25MHz~500MHz

PLL3 功能框图：

图 4-4 PLL3 功能框图



■ PLL 配置流程

- 首先使能 PLLx（将寄存器 RCC_PLLxCR 的 PLLxEN 位置 1）
- 若使用 PLL1 或 PLL2 的 SSC 功能，需先配置寄存器 RCC_PLLxSCR，并使能 SSC 功能（将寄存器 RCC_PLLxSCR 的 PLLxSSCEN 位置 1）
- 退出 PLLx 休眠模式（配置寄存器 RCC_PLLxCR 的 PLLxSLEEP 位为 0）
- 等待寄存器 RCC_PLLxCR 的 PLLxLOCK 位或 PLLxFREERUN 位置 1；
- 若需改变 PLLx 输出频率，可重新配置寄存器 RCC_PLLxCFR 的 PLLxQ、PLLxP、PLLxN（仅 PLL1 和 PLL2 支持）和 PLLxF 位
- 将寄存器 RCC_PLLxCR 的 PLLxUPDATEEN 位置 1
- 等待寄存器 RCC_PLLxCR 的 PLLxLOCK 位或 PLLxFREERUN 位置 1（当 RCH 作为 PLL 的时钟源时，由于 RC 振荡器 Jitter 可能偏大，会导致 PLLxLOCK 位不可靠，此时可查询 PLLxFREERUN 位来判断 PLLx 是否正常工作，PLLxFREERUN 位跟 PLLxRUNDLY 有关，PLLx 等待时间的设置可参见数据手册的电气特性部分）

如果在时钟中断寄存器 RCC_CIR 中允许产生中断，将会产生相应的中断（寄存器 RCC_PLLxCR 的 PLLxLOCKSEL 位选择 PLLx 中断源是 PLLxLOCK 还是 PLLxFREERUN）。

4.2.6. 系统时钟

系统时钟（SYSCLK）有 3 种时钟源：RCH、XTH、PLL1PCLK。在系统复位后，默认系统时钟为 RCH。在直接使用 RCH 或通过 PLL1PCLK 使用 RCH 为时钟源作为系统时钟时，RCH 时钟源不能停止。

根据工作模式不同，采用不同时钟源方案，通过配置时钟控制寄存器 RCC_CCR1 的 SYSCLKSEL 位来选择系统时钟的来源。切换时钟时，请保证目标时钟源已经使能，并且切换完成后才能关闭原时钟源。

表 4-1 系统时钟源

SYSCLKSEL	系统时钟源
00/01	RCH
10	XTH

11	PLL1PCLK

4.2.7. 时钟安全系统

■ XTH 时钟停振检测

XTH 时钟停振检测功能可通过软件激活。激活后，XTH 时钟停振检测器将在 XTH 振荡器启动并稳定后使能；当 XTH 时钟禁止时，XTH 时钟停振检测器也将禁止；

当停振检测器检测到 XTH 停振时，系统会自动切换到 RCH 时钟，XTH 振荡器被禁止，随后停振检测器检测使能位将由硬件清零；一个时钟故障事件将发送到高级控制定时器（TIM1、TIM8、TIM20）的断路输入，同时产生一个中断标志位，并自动生成 NMI。

如果 XTH 时钟作为 PLL 的时钟源，则在发生故障时，PLL 也会被禁止。

进入 STOP 模式前，需将 XTH 禁止，否则，虽然 XTH 可在 STOP 模式下工作，并在发生停振时产生 NMI，但仅能唤醒 CPU，却唤醒不了系统。

■ XTL 时钟停振检测

XTL 时钟停振检测功能可通过软件激活。激活后，XTL 时钟停振检测器将在 XTL 振荡器启动并稳定后使能；

当停振检测器检测到 XTL 停振时，会产生一个中断标志位，软件必须先禁止停振检测器检模块（XTHSDEN=0），随后可禁止 XTL 时钟模块及清除中断标志位；

XTL 停振检测模块可通过 EXTI 唤醒 STOP，唤醒后，如果已使能中断，则立刻进入中断服务程序；

XTL 停振检测模块可唤醒 STANDBY，唤醒后，会产生中断标志位；软件必须先禁止停振检测器检模块（XTHSDEN=0）；如果随后使能中断，则立刻进入中断服务程序。

4.2.8. RTC 时钟

通过设置寄存器 RCC_STDBYCTRL 中的 RTCSEL 位，RTC 时钟源可由 RCL 或 XTL 时钟提供。

在配置 RTC 时钟源时，必须将寄存器 RCC_STDBYCTRL 中的 RTCEN 位（RTC 时钟使能位）置 1。

4.2.9. 看门狗时钟

RCL 时钟作为独立看门狗（IWDG)的时钟源，在使用独立看门狗前，须软件使能 RCL 时钟。

4.2.10. 时钟输出

芯片有两个时钟输出（MCO）引脚可供使用，分别为 MCO1 和 MCO2。可以为每个输出选择一个时钟源。所选时钟可通过可配置的预分频器进行分频。

MCO1 和 MCO2 输出通过寄存器 RCC_CLKOCR 进行控制。

GPIO 端口对应于各个 MCO 引脚，必须在复用功能模式下进行编程（仅 PA8 对应的 MCO1，在芯片复位后默认为 MCO1 输出，详见芯片数据手册中的“复用功能映射”表)。

为 MCO 输出提供的时钟不能超出最大引脚速度（详细信息请参见产品数据手册)。

4.3. RCC 寄存器描述

4.3.1. 寄存器列表

RCC 寄存器基地址：0x40021000

偏移	名称	复位值	描述
0x00	RCC_RCR	0x80000004	复位控制寄存器
0x04	RCC_RSR	0x00000600	复位源状态寄存器
0x08	RCC_AHB1RSTR	0x00f0fe63	AHB1 外设模块复位控制寄存器
0x0C	RCC_AHB2RSTR	0x989fffff	AHB2 外设模块复位控制寄存器
0x10	RCC_AHB3RSTR	0x00001313	AHB3 外设模块复位控制寄存器
0x14	RCC_APB1RSTR1	0xc9ffc9ff	APB1 外设模块复位控制寄存器 1
0x18	RCC_APB1RSTR2	0x000000fe	APB1 外设模块复位控制寄存器 2
0x1C	RCC_APB2RSTR	0xffecbf55	APB2 外设模块复位控制寄存器
0x20	RCC_APB3RSTR	0x0000000f	APB3 外设模块复位控制寄存器
0x24	RCC_APB4RSTR	0x0003f1ff	APB4 外设模块复位控制寄存器
0x28	RCC_CCR1	0x00000000	时钟控制寄存器 1
0x2C	RCC_CCR2	0x83000013	时钟控制寄存器 2
0x30	RCC_PERCFGR	0x03000000	外设模块时钟配置寄存器
0x34	RCC_CIR	0x00000005	时钟中断寄存器
0x38	RCC_AHB1CKENR	0x10000000	AHB1 外设模块时钟使能寄存器
0x3C	RCC_AHB2CKENR	0x00000000	AHB2 外设模块时钟使能寄存器
0x40	RCC_AHB3CKENR	0x00000000	AHB3 外设模块时钟使能寄存器
0x44	RCC_APB1CKENR1	0x00000000	APB1 外设模块时钟使能寄存器 1
0x48	RCC_APB1CKENR2	0x000000c0	APB1 外设模块时钟使能寄存器 2
0x4C	RCC_APB2CKENR	0x00000001	APB2 外设模块时钟使能寄存器
0x50	RCC_APB3CKENR	0x00000000	APB3 外设模块时钟使能寄存器
0x54	RCC_APB4CKENR	0x00000000	APB4 外设模块时钟使能寄存器
0x58	RCC_RCHCR	0x00000011	RCH 控制寄存器
0x5C	RCC_XTHCR	0x0000000c	XTH 控制寄存器
0x60	RCC_PLL1CR	0x08200000	PLL1 模块控制寄存器
0x64	RCC_PLL1CFR	0x000041b8	PLL1 模块配置寄存器
0x68	RCC_PLL1SCR	0x00000000	PLL1 模块扩频控制寄存器
0x6C	RCC_PLL2CR	0x08200000	PLL2 模块控制寄存器
0x70	RCC_PLL2CFR	0x000041b8	PLL2 模块配置寄存器
0x74	RCC_PLL2SCR	0x00000000	PLL2 模块扩频控制寄存器

0x78	RCC_PLL3CR	0x08200000	PLL3 模块控制寄存器
0x7C	RCC_PLL3CFR	0x0010006d	PLL3 模块配置寄存器
0x84	RCC_CLKOCR	0x3f8013e0	时钟输出控制寄存器
0x88	RCC_DCKCFG	0x00000000	外设专用时钟配置寄存器
0x8C	RCC_STDBYCTRL	0x00c08338	STANDBY 电源域控制寄存器

4.3.2. 复位控制寄存器(RCC_RCR: 00h)

位域	名称	属性	复位值	描述
31	SRST	W	1	软件复位 SOFT RST： 写 0 产生系统复位，下一个时钟自动回 1
30:16	RSV	-	-	保留
15:8	IWDTRSTDIS	WO	0	IWDT reset 禁止码值： 该位写入 0xB5，同时 bit2 写入 1’ b0，可将 IWDT 复位系统功能禁止
7:4	RSV	-	-	保留
3	LOCKRSTEN	RW	0	CPU 发生 LOCKUP 后，是否复位系统 0：不允许其复位系统逻辑 1：允许 LOCKUP 复位系统逻辑
2	IWDTRSTEN	RW	1	IWDT reset 使能，使能后允许独立看门狗信号复位系统 0：不允许其复位系统逻辑 1：允许看门狗 reset 复位系统逻辑
1	WDTRSTEN	RW	0	WDT reset 使能，使能后允许看门狗信号复位系统 0：不允许其复位系统逻辑 1：允许看门狗 reset 复位系统逻辑
0	LVDRSTEN	RW	0	低压 LVD reset 使能，使能后允许低电压检测电路产生的信号复位系统 0：不允许其复位系统逻辑 1：允许 VDL RESET 复位系统逻辑

4.3.3. 复位源状态寄存器(RCC_RSR: 04h)

复位状态寄存器，记录引起系统复位的原因。需要通过 RSTFlag_Clr 清除，或者上电复位清除。

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	RSTFLAGCLR	WO	0	写 1 清除所有复位标志

15:11	RSV	-	-	保留
10	PWRRSTF	RO	1	PWR 复位标志： 0:上一次复位不是由 POR/BOR 复位设置引起 1:上一次复位由 POR/BOR 复位设置引起 注：此标志在 STANDBY 区域，清除会有延迟。
9	POR12RSTF	RO	1	POR12 复位标志： 0:上一次复位不是由 POR12 复位设置引起 1:上一次复位由 POR12 复位设置引起
8	SRSTF	RO	0	SOFT reset status： 0:上一次复位不是由系统软复位设置引起 1:上一次复位由系统软复位设置引起
7:6	RSV	-	-	保留
5	RSTNF	RO	0	RSTN Reset 标志： 0:上一次复位不是由复位管脚引起 1:上一次复位由复位管脚引起
4	SYSREQRSTF	RO	0	CPU SYSREQ Reset 标志： 0:上一次复位不是由 SYSREQ 引起 1:上一次复位由 SYSREQ 引起
3	LOCKUPRSTF	RO	0	LOCKUP Reset 标志： 0:上一次复位不是由 LOCKUP 引起 1:上一次复位由 LOCKUP 引起
2	IWDTRSTF	RO	0	IWDT Reset 标志： 0:上一次复位不是由独立看门狗模块引起 1:上一次复位由独立看门狗模块引起 注：此标志在 STANDBY 区域，清除会有延迟。
1	WDTRSTF	RO	0	WDT Reset 标志： 0:上一次复位不是由看门狗模块引起 1:上一次复位由看门狗模块引起
0	LVDRSTF	RO	0	LVD Reset 标志： 0：上一次复位不是由低压检测模块引起 1：上一次复位由低电压检测模块引起

4.3.4. AHB1 外设模块复位控制寄存器(RCC_AHB1RSTR: 08h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留

23	FDCAN2RST	RW	1	FDCAN2 模块复位 0：复位 1：不复位
22	FDCAN1RST	RW	1	FDCAN1 模块复位 0：复位 1：不复位
21	USB2CRST	RW	1	USB OTG2 控制器复位 0：复位 1：不复位
20	USB1CRST	RW	1	USB OTG1 控制器复位 0：复位 1：不复位
19:16	RSV	-	-	保留
15	SPI6RST	RW	1	SPI6 模块复位 0：复位 1：不复位
14	SPI5RST	RW	1	SPI5 模块复位 0：复位 1：不复位
13	SPI4RST	RW	1	SPI4 模块复位 0：复位 1：不复位
12	SPI3RST	RW	1	SPI3 模块复位 0：复位 1：不复位
11	SPI2RST	RW	1	SPI2 模块复位 0：复位 1：不复位
10	SPI1RST	RW	1	SPI1 模块复位 0：复位 1：不复位
9	DMA2DRST	RW	1	DMA2D 模块复位 0：复位 1：不复位
8:7	RSV	-	-	保留

6	ETHRST	RW	1	Ethernet MAC 模块复位 0：复位 1：不复位
5	CRCRST	RW	1	CRC 模块复位 0：复位 1：不复位
4:2	RSV	-	-	保留
1	DMA2RST	RW	1	DMA2 模块复位 0：复位 1：不复位
0	DMA1RST	RW	1	DMA1 模块复位 0：复位 1：不复位

4.3.5. AHB2 外设模块复位控制寄存器(RCC_AHB2RSTR: 0Ch)

位域	名称	属性	复位值	描述
31	THMRST	RW	1	热敏头打印控制器模块复位 0：复位 1：不复位
30:29	RSV	-	-	保留
28	FDCAN3RST	RW	1	FDCAN3 模块复位 0：复位 1：不复位
27	UACRST	RW	1	UAC 模块复位 0：复位 1：不复位
26:24	RSV	-	-	保留
23	DCMIRST	RW	1	DCMI 模块复位 0：复位 1：不复位
22:21	RSV	-	-	保留
20	DAC2RST	RW	1	DAC2 模块复位 0：复位 1：不复位

19	DAC1RST	RW	1	DAC1 模块复位 0：复位 1：不复位
18	ADC3RST	RW	1	ADC3 模块复位 0：复位 1：不复位
17	ADC12RST	RW	1	ADC1/ADC2 模块复位 0：复位 1：不复位
16	GPIOQRST	RW	1	GPIOQ 模块复位 0：复位 1：不复位
15	GPIOPRST	RW	1	GPIOP 模块复位 0：复位 1：不复位
14	GPIOORST	RW	1	GPIOO 模块复位 0：复位 1：不复位
13	GPIONRST	RW	1	GPION 模块复位 0：复位 1：不复位
12	GPIOMRST	RW	1	GPIOM 模块复位 0：复位 1：不复位
11	GPIOLRST	RW	1	GPIOL 模块复位 0：复位 1：不复位
10	GPIOKRST	RW	1	GPIOK 模块复位 0：复位 1：不复位
9	GPIOJRST	RW	1	GPIOJ 模块复位 0：复位 1：不复位
8	GPIOIRST	RW	1	GPIOI 模块复位 0：复位 1：不复位

7	GPIOHRST	RW	1	GPIOH 模块复位 0：复位 1：不复位
6	GPIOGRST	RW	1	GPIOG 模块复位 0：复位 1：不复位
5	GPIOFRST	RW	1	GPIOF 模块复位 0：复位 1：不复位
4	GPIOERST	RW	1	GPIOE 模块复位 0：复位 1：不复位
3	GPIODRST	RW	1	GPIOD 模块复位 0：复位 1：不复位
2	GPIOCRST	RW	1	GPIOC 模块复位 0：复位 1：不复位
1	GPIOBRST	RW	1	GPIOB 模块复位 0：复位 1：不复位
0	GPIOARST	RW	1	GPIOA 模块复位 0：复位 1：不复位

4.3.6. AHB3 外设模块复位控制寄存器(RCC_AHB3RSTR: 10h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	FMCRST	RW	1	FMC 模块复位 0：复位 1：不复位
11:10	RSV	-	-	保留
9	OSPI2RST	RW	1	OSPI2-MEM 模块复位 0：复位 1：不复位

8	OSPI1RST	RW	1	OSPI1-MEM 模块复位 0：复位 1：不复位
7:5	RSV	-	-	保留
4	SDMMCRST	RW	1	SDMMC 和 SDMMC-DLYB 模块复位 0：复位 1：不复位
3:2	RSV	-	-	保留
1	(Q)SPI8RST	RW	1	(Q)SPI8 模块复位 0：复位 1：不复位
0	(Q)SPI7RST	RW	1	(Q)SPI7 模块复位 0：复位 1：不复位

4.3.7. APB1 外设模块复位控制寄存器 1(RCC_APB1RSTR1: 14h)

位域	名称	属性	复位值	描述
31	LPUART1RST	RW	1	LPUART1 模块复位 0：复位 1：不复位
30	LPTIM1RST	RW	1	LPTIM1 模块复位 0：复位 1：不复位
29:28	RSV	-	-	保留
27	PMURST	RW	1	PMU 模块复位 0：复位 1：不复位
26:25	RSV	-	-	保留
24	I2C4RST	RW	1	I2C4 模块复位 0：复位 1：不复位
23	I2C3RST	RW	1	I2C3 模块复位 0：复位 1：不复位

22	I2C2RST	RW	1	I2C2 模块复位 0：复位 1：不复位
21	I2C1RST	RW	1	I2C1 模块复位 0：复位 1：不复位
20	UART5RST	RW	1	UART5 模块复位 0：复位 1：不复位
19	UART4RST	RW	1	UART4 模块复位 0：复位 1：不复位
18	UART3RST	RW	1	UART3 模块复位 0：复位 1：不复位
17	UART2RST	RW	1	UART2 模块复位 0：复位 1：不复位
16	I2S3RST	RW	1	I2S3 模块复位 0：复位 1：不复位
15	I2S2RST	RW	1	I2S2 模块复位 0：复位 1：不复位
14	I2S1RST	RW	1	I2S1 模块复位 0：复位 1：不复位
13:12	RSV	-	-	保留
11	WDTRST	RW	1	WDT 模块复位 0：复位 1：不复位
10:9	RSV	-	-	保留
8	TIM14RST	RW	1	TIM14 模块复位 0：复位 1：不复位

7	TIM13RST	RW	1	TIM13 模块复位 0：复位 1：不复位
6	TIM12RST	RW	1	TIM12 模块复位 0：复位 1：不复位
5	TIM7RST	RW	1	TIM7 模块复位 0：复位 1：不复位
4	TIM6RST	RW	1	TIM6 模块复位 0：复位 1：不复位
3	TIM5RST	RW	1	TIM5 模块复位 0：复位 1：不复位
2	TIM4RST	RW	1	TIM4 模块复位 0：复位 1：不复位
1	TIM3RST	RW	1	TIM3 模块复位 0：复位 1：不复位
0	TIM2RST	RW	1	TIM2 模块复位 0：复位 1：不复位

4.3.8. APB1 外设模块复位控制寄存器 2(RCC_APB1RSTR2: 18h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	EFUSE2RST	RW	1	EFUSE2 模块复位 0：复位 1：不复位
6	EFUSE1RST	RW	1	EFUSE1 模块复位 0：复位 1：不复位

5	TIM26RST	RW	1	TIM26 模块复位 0：复位 1：不复位
4	TIM25RST	RW	1	TIM25 模块复位 0：复位 1：不复位
3	UART8RST	RW	1	UART8 模块复位 0：复位 1：不复位
2	UART7RST	RW	1	UART7 模块复位 0：复位 1：不复位
1	LPTIM2RST	RW	1	LPTIM2 模块复位 0：复位 1：不复位
0	RSV	-	-	保留

4.3.9. APB2 外设模块复位控制寄存器(RCC_APB2RSTR: 1Ch)

位域	名称	属性	复位值	描述
31	UART10RST	RW	1	UART10 模块复位 0：复位 1：不复位
30	UART9RST	RW	1	UART9 模块复位 0：复位 1：不复位
29	TIM24RST	RW	1	TIM24 模块复位 0：复位 1：不复位
28	TIM23RST	RW	1	TIM23 模块复位 0：复位 1：不复位
27	TIM22RST	RW	1	TIM22 模块复位 0：复位 1：不复位

26	TIM21RST	RW	1	TIM21 模块复位 0：复位 1：不复位
25	TIM19RST	RW	1	TIM19 模块复位 0：复位 1：不复位
24	TIM18RST	RW	1	TIM18 模块复位 0：复位 1：不复位
23	TIM11RST	RW	1	TIM11 模块复位 0：复位 1：不复位
22	TIM10RST	RW	1	TIM10 模块复位 0：复位 1：不复位
21	TIM9RST	RW	1	TIM9 模块复位 0：复位 1：不复位
20	RSV	-	-	保留
19	LTDCRST	RW	1	LTDC 模块复位 0：复位 1：不复位
18	TKEYRST	RW	1	TKEY 模块复位 0：复位 1：不复位
17:16	RSV	-	-	保留
15	TIM20RST	RW	1	TIM20 模块复位 0：复位 1：不复位
14	RSV	-	-	保留
13	TIM17RST	RW	1	TIM17 模块复位 0：复位 1：不复位
12	TIM16RST	RW	1	TIM16 模块复位 0：复位 1：不复位

11	TIM15RST	RW	1	TIM15 模块复位 0：复位 1：不复位
10	UART6RST	RW	1	UART6 模块复位 0：复位 1：不复位
9	UART1RST	RW	1	UART1 模块复位 0：复位 1：不复位
8	TIM8RST	RW	1	TIM8 模块复位 0：复位 1：不复位
7	RSV	RW	-	-
6	TIM1RST	RW	1	TIM1 模块复位 0：复位 1：不复位
5	RSV	-	-	保留
4	EXTIRST	RW	1	EXTI 模块复位 0：复位 1：不复位
3	RSV	-	-	保留
2	CMP1RST	RW	1	CMP1 模块复位 0：复位 1：不复位
1	RSV	-	-	保留
0	SYSCFGRST	RW	1	SYSCFG 复位 0：复位 1：不复位

4.3.10. APB3 外设模块复位控制寄存器(RCC_APB3RSTR: 20h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留

3	LPTIM6RST	RW	1	LPTIM6 模块复位 0：复位 1：不复位
2	LPTIM5RST	RW	1	LPTIM5 模块复位 0：复位 1：不复位
1	LPTIM4RST	RW	1	LPTIM4 模块复位 0：复位 1：不复位
0	LPTIM3RST	RW	1	LPTIM3 模块复位 0：复位 1：不复位

4.3.11. APB4 外设模块复位控制寄存器(RCC_APB4RSTR: 24h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	DPWM6RST	RW	1	直流电机 PWM 控制器模块 6 复位 0：复位 1：不复位
16	DPWM5RST	RW	1	直流电机 PWM 控制器模块 5 复位 0：复位 1：不复位
15	SPWM6RST	RW	1	步进电机 PWM 控制器模块 6 复位 0：复位 1：不复位
14	SPWM5RST	RW	1	步进电机 PWM 控制器模块 5 复位 0：复位 1：不复位
13	SPWM4RST	RW	1	步进电机 PWM 控制器模块 4 复位 0：复位 1：不复位
12	SPWM3RST	RW	1	步进电机 PWM 控制器模块 3 复位 0：复位 1：不复位
11:9	RSV	-	-	保留

8	MDACRST	RW	1	多通道 DAC 模块复位 0：复位 1：不复位
7	LPTRST	RW	1	LPT 模块复位 0：复位 1：不复位
6	PNDLRST	RW	1	打印机针头控制器模块复位 0：复位 1：不复位
5	DPWM4RST	RW	1	直流电机 PWM 控制器模块 4 复位 0：复位 1：不复位
4	DPWM3RST	RW	1	直流电机 PWM 控制器模块 3 复位 0：复位 1：不复位
3	DPWM2RST	RW	1	直流电机 PWM 控制器模块 2 复位 0：复位 1：不复位
2	DPWM1RST	RW	1	直流电机 PWM 控制器模块 1 复位 0：复位 1：不复位
1	SPWM2RST	RW	1	步进电机 PWM 控制器模块 2 复位 0：复位 1：不复位
0	SPWM1RST	RW	1	步进电机 PWM 控制器模块 1 复位 0：复位 1：不复位

4.3.12. 时钟控制寄存器 1(RCC_CCR1: 28h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留

1:0	SYSCLKSEL	RW	00	系统时钟 SYS_CLK 选择： 00：来自 RCH 01：来自 RCH 10：来自 XTH 11：来自 PLL1PCLK

4.3.13. 时钟控制寄存器 2(RCC_CCR2: 2Ch)

位域	名称	属性	复位值	描述
31	DIVDONE	RO	1	SYSDIV 分频设置计数 256 个时钟完成，可以更新分频值 0：计数 256 个时钟未完成 1：计数 256 个时钟完成
30:24	HRNGSDIV	RW	0x03	HRNG SlowClk 分频选择，产生 HRNG_SCLK 0000000:不分频 0000001:2 分频 0000010:3 分频 0000011:4 分频 … 1111100:125 分频 1111101:126 分频 1111110:127 分频 1111111:128 分频
23:21	RSV	-	-	保留
20	FLTCLKSEL	RW	0	FLTCLK 选择，LVD 和 COMP 的滤波时钟选择 0：PCLK1 的 32 分频 1：RCL
19:17	PCLK4DIV	RW	000	PCLK4 分频设置 0xx: 不分频 100：2 分频 101：4 分频 110：8 分频 111: 16 分频

16:14	PCLK3DIV	RW	000	PCLK3 分频设置 0xx: 不分频 100：2 分频 101：4 分频 110：8 分频 111: 16 分频
13:11	PCLK2DIV	RW	000	PCLK2 分频设置 0xx: 不分频 100：2 分频 101：4 分频 110：8 分频 111: 16 分频
10:8	PCLK1DIV	RW	000	PCLK1 分频设置 0xx: 不分频 100：2 分频 101：4 分频 110：8 分频 111: 16 分频
7:4	SYSDIV1	RW	0001	系统时钟第二级分频选择，产生 HCLK 0000:不分频 0001:2 分频 0010:3 分频 0011:4 分频 … 1100:13 分频 1101:14 分频 1110:15 分频 1111:16 分频

3:0	SYSDIV0	RW	0011	系统时钟第一级分频选择，产生 SYS_CLK_DIV0 0000:不分频 0001:2 分频 0010:3 分频 0011:4 分频 … 1100:13 分频 1101:14 分频 1110:15 分频 1111:16 分频 注：为了防止频率突变导致电源不稳定，更改系统时钟和算法时钟分频设置需要逐级变化，每次更改变化后保证 256 个时钟后（读取 DIVDONE 为 1）再更新分频值。

4.3.14. 外设模块时钟配置寄存器(RCC_PERCFGR: 30h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:26	LPUART1CKS	RW	00	LPUART1 时钟选择 00: RCL 01: XTL 10: PCLK1 分频，由 LPUARTDIV 确定
25:24	LPUART1DIV	RW	11	LPUART1 使用 PCLK1 分频选择 00:4 分频 01:8 分频 10:16 分频 11:32 分频
23:22	LPTIM6CKS	RW	0	LPTIM6 时钟选择 00：PCLK3 01：RCL 10：RCH 11：XTL
21:20	LPTIM345CKS	RW	0	LPTIM3/4/5 时钟选择 00：PCLK3 01：RCL 10：RCH 11：XTL

19:18	LPTIM2CKS	RW	0	LPTIM2 时钟选择 00：PCLK1 01：RCL 10：RCH 11：XTL
17:16	LPTIM1CKS	RW	0	LPTIM1 时钟选择 00：PCLK1 01：RCL 10：RCH 11：XTL
15:12	RSV	-	-	保留
11	SDMMCCKS	RW	0	SDMMC 时钟选择 0:HCLK 1:PLL2PCLK
10	RSV	-	-	保留
9:8	SDMMCSCKS	RW	0	SD/SDIO/MMC 采样时钟选择 00：SD/SDIO/MMC 控制器时钟（经过延时单元） 01：SD/SDIO/MMC 卡 1 的外部驱动器的反馈时钟 10：SD/SDIO/MMC 卡 2 的外部驱动器的反馈时钟 11：SD/SDIO/MMC 控制器时钟（未经过延时单元）
7:0	RSV	-	-	保留

4.3.15. 时钟中断寄存器(RCC_CIR: 34h)

位域	名称	属性	复位值	描述
31	XTLSDF	RO	0	XTL 停振中断标志 当 XTL 被检测到停振后硬件置位。软件设置 XTLSDIC 清除 0：无 XTL 停振中断 1：产生 XTL 停振中断
30	RSV	-	-	保留
29	XTLSDIC	WO	0	XTL 停振中断清除。 软件写 1 清除中断 XTLSDIF 0：无影响 1：清除 XTL 停振中断标志

28	XTLSDIE	RW	0	XTL 停振中断使能 0：禁止 XTL 停振中断 1：使能 XTL 停振中断
27	XTHSDF	RO	0	XTH 停振中断标志 当 XTH 被检测到停振后硬件置位。软件设置 XTHSDIC 清除 0：无 XTH 停振中断 1：产生 XTH 停振中断
26	RSV	-	-	保留
25	XTHSDIC	WO	0	XTH 停振中断清除。 软件写 1 清除中断 XTHSDIF 0：无影响 1：清除 XTH 停振中断标志
24	XTHSDIE	RW	0	XTH 停振中断使能 0：禁止 XTH 停振中断 1：使能 XTH 停振中断
23	RSV	-	-	保留
22	PLL3LOCKIC	WO	0	PLL3 LOCK 中断清除。 软件写 1 清除中断 PLL3LOCKIF 0：无影响 1：清除 PLL3 LOCK 中断标志
21	PLL2LOCKIC	WO	0	PLL2 LOCK 中断清除。 软件写 1 清除中断 PLL2LOCKIF 0：无影响 1：清除 PLL2 LOCK 中断标志
20	PLL1LOCKIC	WO	0	PLL1 LOCK 中断清除。 软件写 1 清除中断 PLL1LOCKIF 0：无影响 1：清除 PLL1 LOCK 中断标志
19	XTHRDYIC	WO	0	XTH 时钟稳定中断清除。 软件写 1 清除中断 XTHRDYIF 0：无影响 1：清除 XTH 时钟稳定中断标志
18	RCHRDYIC	WO	0	RCH 时钟稳定中断清除。 软件写 1 清除中断 RCHRDYIF 0：无影响 1：清除 RCH 时钟稳定中断标志

17	XTLRDYIC	WO	0	XTL 时钟稳定中断清除。 软件写 1 清除中断 XTLRDYIF 0：无影响 1：清除 XTL 时钟稳定中断标志
16	RCLRDYIC	WO	0	RCL 时钟稳定中断清除。 软件写 1 清除中断 RCLRDYIF 0：无影响 1：清除 RCL 时钟稳定中断标志
15	RSV	-	-	保留
14	PLL3LOCKIE	RW	0	PLL3 LOCK 中断使能 0：禁止 PLL3 LOCK 中断 1：使能 PLL3 LOCK 中断
13	PLL2LOCKIE	RW	0	PLL2 LOCK 中断使能 0：禁止 PLL2 LOCK 中断 1：使能 PLL2 LOCK 中断
12	PLL1LOCKIE	RW	0	PLL1 LOCK 中断使能 0：禁止 PLL1 LOCK 中断 1：使能 PLL1 LOCK 中断
11	XTHRDYIE	RW	0	XTH 时钟稳定中断使能 0：禁止 XTH 时钟稳定中断 1：使能 XTH 时钟稳定中断
10	RCHRDYIE	RW	0	RCH 时钟稳定中断使能 0：禁止 RCH 时钟稳定中断 1：使能 RCH 时钟稳定中断
9	XTLRDYIE	RW	0	XTL 时钟稳定中断使能 0：禁止 XTL 时钟稳定中断 1：使能 XTL 时钟稳定中断
8	RCLRDYIE	RW	0	RCL 时钟稳定中断使能 0：禁止 RCL 时钟稳定中断 1：使能 RCL 时钟稳定中断
7	RSV	-	-	保留
6	PLL3LOCKIF	RO	0	PLL3 LOCK 中断标志 当 PLL3 模块 LOCK，PLL3LOCK(或 PLL3FREERUN)标志有效后硬件置位。软件设置 PLL3LOCKIC 清除 0：无 PLL3 LOCK 中断 1：产生 PLL3 LOCK 中断

5	PLL2LOCKIF	RO	0	PLL2 LOCK 中断标志 当 PLL2 模块 LOCK，PLL2LOCK(或 PLL2FREERUN)标志有效后硬件置位。软件设置 PLL2LOCKIC 清除 0：无 PLL2 LOCK 中断 1：产生 PLL2 LOCK 中断
4	PLL1LOCKIF	RO	0	PLL1 LOCK 中断标志 当 PLL1 模块 LOCK，PLL1LOCK(或 PLL1FREERUN)标志有效后硬件置位。软件设置 PLL1LOCKIC 清除 0：无 PLL1 LOCK 中断 1：产生 PLL1 LOCK 中断
3	XTHRDYIF	RO	0	XTH 时钟稳定中断标志 当 XTH 时钟稳定，XTHRDY 标志有效后硬件置位。软件设置 XTHRDYIC 清除 0：无 XTH 时钟稳定中断 1：产生 XTH 时钟稳定中断
2	RCHRDYIF	RO	1	RCH 时钟稳定中断标志 当 RCH 时钟稳定，RCHRDY 标志有效后硬件置位。软件设置 RCHRDYIC 清除 0：无 RCH 时钟稳定中断 1：产生 RCH 时钟稳定中断
1	XTLRDYIF	RO	0	XTL 时钟稳定中断标志 当 XTL 时钟稳定，XTLRDY 标志有效后硬件置位。软件设置 XTLRDYIC 清除 0：无 XTL 时钟稳定中断 1：产生 XTL 时钟稳定中断
0	RCLRDYIF	RO	1	RCL 时钟稳定中断标志 当 RCL 时钟稳定，RCLRDY 标志有效后硬件置位。软件设置 RCLRDYIC 清除 0：无 RCL 时钟稳定中断 1：产生 RCL 时钟稳定中断

4.3.16. AHB1 外设模块时钟使能寄存器(RCC_AHB1CKENR: 38h)

位域	名称	属性	复位值	描述
31	SRAM3CKEN	RW	0	SRAM3 在 SLEEP 模式下时钟使能 0：SLEEP 模式下不使能 1：SLEEP 模式下使能
30	SRAM2CKEN	RW	0	SRAM2 在 SLEEP 模式下时钟使能 0：SLEEP 模式下不使能 1：SLEEP 模式下使能

29	SRAM1CKEN	RW	0	SRAM1 在 SLEEP 模式下时钟使能 0：SLEEP 模式下不使能 1：SLEEP 模式下使能
28	ROMCKEN	RW	1	ROM 模块时钟使能 0：不使能 1：使能
27	BKPSRAMCKEN	RW	0	BKPSRAM 时钟使能 0：不使能 1：使能
26:24	RSV	-	-	保留
23	FDCAN2CKEN	RW	0	FDCAN2 模块时钟使能 0：不使能 1：使能
22	FDCAN1CKEN	RW	0	FDCAN1 模块时钟使能 0：不使能 1：使能
21	USB2CCKEN	RW	0	USB OTG2 控制器时钟使能 0：不使能 1：使能
20	USB1CCKEN	RW	0	USB OTG1 控制器时钟使能 0：不使能 1：使能
19:16	RSV	-	-	保留
15	SPI6CKEN	RW	0	SPI6 模块时钟使能 0：不使能 1：使能
14	SPI5CKEN	RW	0	SPI5 模块时钟使能 0：不使能 1：使能
13	SPI4CKEN	RW	0	SPI4 模块时钟使能 0：不使能 1：使能
12	SPI3CKEN	RW	0	SPI3 模块时钟使能 0：不使能 1：使能

11	SPI2CKEN	RW	0	SPI2 模块时钟使能 0：不使能 1：使能
10	SPI1CKEN	RW	0	SPI1 模块时钟使能 0：不使能 1：使能
9	DMA2DCKEN	RW	0	DMA2D 模块时钟使能 0：不使能 1：使能
8	ETHRXCKEN	RW	0	Ethernet 接收时钟使能 0：不使能 1：使能
7	ETHTXCKEN	RW	0	Ethernet 发送时钟使能 0：不使能 1：使能
6	ETHMACCKEN	RW	0	Ethernet MAC 总线接口时钟使能 0：不使能 1：使能
5	CRCCKEN	RW	0	CRC 模块时钟使能 0：不使能 1：使能
4:2	RSV	-	-	保留
1	DMA2CKEN	RW	0	DMA2 模块时钟使能 0：不使能 1：使能
0	DMA1CKEN	RW	0	DMA1 模块时钟使能 0：不使能 1：使能

4.3.17. AHB2 外设模块时钟使能寄存器(RCC_AHB2CKENR: 3Ch)

位域	名称	属性	复位值	描述
31	THMCKEN	RW	0	热敏头打印控制器模块时钟使能 0：不使能 1：使能
30:29	RSV	-	-	保留

28	FDCAN3CKEN	RW	0	FDCAN3 模块时钟使能 0：不使能 1：使能
27	CORDICCKEN	RW	0	CORDIC 模块时钟使能 0：不使能 1：使能
26	HRNGCKEN	RW	0	HRENG 模块时钟使能 0：不使能 1：使能
25	AESCKEN	RW	0	AES 模块时钟使能 0：不使能 1：使能
24	AESPI1CKEN	RW	0	AES_SPI1 模块时钟使能 0：不使能 1：使能
23	DCMICKEN	RW	0	DCMI 模块时钟使能 0：不使能 1：使能
22:21	RSV	-	-	保留
20	DAC2CKEN	RW	0	DAC2 模块时钟使能 0：不使能 1：使能
19	DAC1CKEN	RW	0	DAC1 模块时钟使能 0：不使能 1：使能
18	ADC3CKEN	RW	0	ADC3 模块时钟使能 0：不使能 1：使能
17	ADC12CKEN	RW	0	ADC1/ADC2 模块时钟使能 0：不使能 1：使能
16	GPIOQCKEN	RW	0	GPIOQ 模块时钟使能 0：不使能 1：使能

15	GPIOPCKEN	RW	0	GPIOP 模块时钟使能 0：不使能 1：使能
14	GPIOOCKEN	RW	0	GPIOO 模块时钟使能 0：不使能 1：使能
13	GPIONCKEN	RW	0	GPION 模块时钟使能 0：不使能 1：使能
12	GPIOMCKEN	RW	0	GPIOM 模块时钟使能 0：不使能 1：使能
11	GPIOLCKEN	RW	0	GPIOL 模块时钟使能 0：不使能 1：使能
10	GPIOKCKEN	RW	0	GPIOK 模块时钟使能 0：不使能 1：使能
9	GPIOJCKEN	RW	0	GPIOJ 模块时钟使能 0：不使能 1：使能
8	GPIOICKEN	RW	0	GPIOI 模块时钟使能 0：不使能 1：使能
7	GPIOHCKEN	RW	0	GPIOH 模块时钟使能 0：不使能 1：使能
6	GPIOGCKEN	RW	0	GPIOG 模块时钟使能 0：不使能 1：使能
5	GPIOFCKEN	RW	0	GPIOF 模块时钟使能 0：不使能 1：使能
4	GPIOECKEN	RW	0	GPIOE 模块时钟使能 0：不使能 1：使能

3	GPIODCKEN	RW	0	GPIOD 模块时钟使能 0：不使能 1：使能
2	GPIOCCKEN	RW	0	GPIOC 模块时钟使能 0：不使能 1：使能
1	GPIOBCKEN	RW	0	GPIOB 模块时钟使能 0：不使能 1：使能
0	GPIOACKEN	RW	0	GPIOA 模块时钟使能 0：不使能 1：使能

4.3.18. AHB3 外设模块时钟使能寄存器(RCC_AHB3CKENR: 40h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	FMCCKEN	RW	0	FMC 模块时钟使能 0：不使能 1：使能
11:10	RSV	-	-	保留
9	OSPI2CKEN	RW	0	OSPI2-MEM 模块时钟使能 0：不使能 1：使能
8	OSPI1CKEN	RW	0	OSPI1-MEM 模块时钟使能 0：不使能 1：使能
7:5	RSV	-	-	保留
4	SDMMCCKEN	RW	0	SDMMC 和 SDMMC-DLYB 模块时钟使能 0：不使能 1：使能
3:2	RSV	-	-	保留
1	(Q)SPI8CKEN	RW	0	(Q)SPI8 模块时钟使能 0：不使能 1：使能

0	(Q)SPI7CKEN	RW	0	(Q)SPI7 模块时钟使能 0：不使能 1：使能

4.3.19. APB1 外设模块时钟使能寄存器 1(RCC_APB1CKENR1: 44h)

位域	名称	属性	复位值	描述
31	LPUART1CKEN	RW	0	LPUART1 模块时钟使能 0：不使能 1：使能
30	LPTIM1CKEN	RW	0	LPTIM1 模块时钟使能 0：不使能 1：使能
29:28	RSV	-	-	保留
27	PMUCKEN	RW	0	PMU 模块时钟使能 0：不使能 1：使能
26:25	RSV	-	-	保留
24	I2C4CKEN	RW	0	I2C4 模块时钟使能 0：不使能 1：使能
23	I2C3CKEN	RW	0	I2C3 模块时钟使能 0：不使能 1：使能
22	I2C2CKEN	RW	0	I2C2 模块时钟使能 0：不使能 1：使能
21	I2C1CKEN	RW	0	I2C1 模块时钟使能 0：不使能 1：使能
20	UART5CKEN	RW	0	UART5 模块时钟使能 0：不使能 1：使能
19	UART4CKEN	RW	0	UART4 模块时钟使能 0：不使能 1：使能

18	UART3CKEN	RW	0	UART3 模块时钟使能 0：不使能 1：使能
17	UART2CKEN	RW	0	UART2 模块时钟使能 0：不使能 1：使能
16	I2S3CKEN	RW	0	I2S3 模块时钟使能 0：不使能 1：使能
15	I2S2CKEN	RW	0	I2S2 模块时钟使能 0：不使能 1：使能
14	I2S1CKEN	RW	0	I2S1 模块时钟使能 0：不使能 1：使能
13:12	RSV	-	-	保留
11	WDTCKEN	RW	0	WDT 模块时钟使能 0：不使能 1：使能
10	RTCCKEN	RW	0	RTC 模块总线时钟使能 0：不使能 1：使能
9	RSV	-	-	保留
8	TIM14CKEN	RW	0	TIM14 模块时钟使能 0：不使能 1：使能
7	TIM13CKEN	RW	0	TIM13 模块时钟使能 0：不使能 1：使能
6	TIM12CKEN	RW	0	TIM12 模块时钟使能 0：不使能 1：使能
5	TIM7CKEN	RW	0	TIM7 模块时钟使能 0：不使能 1：使能

4	TIM6CKEN	RW	0	TIM6 模块时钟使能 0：不使能 1：使能
3	TIM5CKEN	RW	0	TIM5 模块时钟使能 0：不使能 1：使能
2	TIM4CKEN	RW	0	TIM4 模块时钟使能 0：不使能 1：使能
1	TIM3CKEN	RW	0	TIM3 模块时钟使能 0：不使能 1：使能
0	TIM2CKEN	RW	0	TIM2 模块时钟使能 0：不使能 1：使能

4.3.20. APB1 外设模块时钟使能寄存器 2(RCC_APB1CKENR2: 48h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	EFUSE2CKEN	RW	1	EFUSE2 模块时钟使能 0：不使能 1：使能
6	EFUSE1CKEN	RW	1	EFUSE1 模块时钟使能 0：不使能 1：使能
5	TIM26CKEN	RW	0	TIM26 模块时钟使能 0：不使能 1：使能
4	TIM25CKEN	RW	0	TIM25 模块时钟使能 0：不使能 1：使能
3	UART8CKEN	RW	0	UART8 模块时钟使能 0：不使能 1：使能

2	UART7CKEN	RW	0	UART7 模块时钟使能 0：不使能 1：使能
1	LPTIM2CKEN	RW	0	LPTIM2 模块时钟使能 0：不使能 1：使能
0	RSV	-	-	保留

4.3.21. APB2 外设模块时钟使能寄存器(RCC_APB2CKENR: 4Ch)

位域	名称	属性	复位值	描述
31	UART10CKEN	RW	0	UART10 模块时钟使能 0：不使能 1：使能
30	UART9CKEN	RW	0	UART9 模块时钟使能 0：不使能 1：使能
29	TIM24CKEN	RW	0	TIM28 模块时钟使能 0：不使能 1：使能
28	TIM23CKEN	RW	0	TIM25 模块时钟使能 0：不使能 1：使能
27	TIM22CKEN	RW	0	TIM22 模块时钟使能 0：不使能 1：使能
26	TIM21CKEN	RW	0	TIM21 模块时钟使能 0：不使能 1：使能
25	TIM19CKEN	RW	0	TIM19 模块时钟使能 0：不使能 1：使能
24	TIM18CKEN	RW	0	TIM18 模块时钟使能 0：不使能 1：使能

23	TIM11CKEN	RW	0	TIM11 模块时钟使能 0：不使能 1：使能
22	TIM10CKEN	RW	0	TIM10 模块时钟使能 0：不使能 1：使能
21	TIM9CKEN	RW	0	TIM9 模块时钟使能 0：不使能 1：使能
20	RSV	-	-	保留
19	LTDCCKEN	RW	0	LTDC 模块时钟使能 0：不使能 1：使能
18	TKEYCKEN	RW	0	TKEY 模块时钟使能 0：不使能 1：使能
17:16	RSV	-	-	保留
15	TIM20CKEN	RW	0	TIM20 模块时钟使能 0：不使能 1：使能
14	RSV	-	-	保留
13	TIM17CKEN	RW	0	TIM17 模块时钟使能 0：不使能 1：使能
12	TIM16CKEN	RW	0	TIM16 模块时钟使能 0：不使能 1：使能
11	TIM15CKEN	RW	0	TIM15 模块时钟使能 0：不使能 1：使能
10	UART6CKEN	RW	0	UART6 模块时钟使能 0：不使能 1：使能
9	UART1CKEN	RW	0	UART1 模块时钟使能 0：不使能 1：使能

8	TIM8CKEN	RW	0	TIM8 模块时钟使能 0：不使能 1：使能
7	RSV	RW	-	-
6	TIM1CKEN	RW	0	TIM1 模块时钟使能 0：不使能 1：使能
5	RSV	-	-	保留
4	EXTICKEN	RW	0	EXTI 模块时钟使能 0：不使能 1：使能
3	RSV	-	-	保留
2	CMP1CKEN	RW	0	CMP1 模块时钟使能 0：不使能 1：使能
1	RSV	-	-	保留
0	SYSCFGCKEN	RW	1	SYSCFG 模块时钟使能 0：不使能 1：使能

4.3.22. APB3 外设模块时钟使能寄存器(RCC_APB3CKENR: 50h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	LPTIM6CKEN	RW	0	LPTIM6 模块时钟使能 0：不使能 1：使能
2	LPTIM5CKEN	RW	0	LPTIM5 模块时钟使能 0：不使能 1：使能
1	LPTIM4CKEN	RW	0	LPTIM4 模块时钟使能 0：不使能 1：使能

0	LPTIM3CKEN	RW	0	LPTIM3 模块时钟使能 0：不使能 1：使能

4.3.23. APB4 外设模块时钟使能寄存器(RCC_APB4CKENR: 54h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	DPWM6CKEN	RW	0	直流电机 PWM 控制器模块 6 时钟使能 0：不使能 1：使能
16	DPWM5CKEN	RW	0	直流电机 PWM 控制器模块 5 时钟使能 0：不使能 1：使能
15	SPWM6CKEN	RW	0	步进电机 PWM 控制器模块 6 时钟使能 0：不使能 1：使能
14	SPWM5CKEN	RW	0	步进电机 PWM 控制器模块 5 时钟使能 0：不使能 1：使能
13	SPWM4CKEN	RW	0	步进电机 PWM 控制器模块 4 时钟使能 0：不使能 1：使能
12	SPWM3CKEN	RW	0	步进电机 PWM 控制器模块 3 时钟使能 0：不使能 1：使能
11:9	RSV	-	-	保留
8	MDACCKEN	RW	0	多通道 DAC 模块时钟使能 0：不使能 1：使能
7	LPTCKEN	RW	0	LPT 模块时钟使能 0：不使能 1：使能
6	PNDLCKEN	RW	0	打印机针头控制器模块时钟使能 0：不使能 1：使能

5	DPWM4CKEN	RW	0	直流电机 PWM 控制器模块 4 时钟使能 0：不使能 1：使能
4	DPWM3CKEN	RW	0	直流电机 PWM 控制器模块 3 时钟使能 0：不使能 1：使能
3	DPWM2CKEN	RW	0	直流电机 PWM 控制器模块 2 时钟使能 0：不使能 1：使能
2	DPWM1CKEN	RW	0	直流电机 PWM 控制器模块 1 时钟使能 0：不使能 1：使能
1	SPWM2CKEN	RW	0	步进电机 PWM 控制器模块 2 时钟使能 0：不使能 1：使能
0	SPWM1CKEN	RW	0	步进电机 PWM 控制器模块 1 时钟使能 0：不使能 1：使能

4.3.24. RCH 模块控制寄存器(RCC_RCHCR: 58h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	RCHPRDY	RO	0	内部 RCHP 时钟稳定标志 0：RCHP 时钟未稳定 1：RCHP 时钟稳定，时钟有效
23:22	RSV	-	-	保留
21	RCHSEL	RW	0	高速 RC 时钟选择位 0：选择 RCH 模块时钟 1：选择 RCHP 模块时钟
20	RCHPEN	RW	0	RCHP 模块使能 0：禁止 1：使能 当 XTH 发生停振且 RCHSEL=1 时，RCHP 使能位将硬件置 1
19:17	RSV	-	-	保留

16:15	RCHTRIMH	RW	0x0	RCH 时钟修调的粗调 TRIM 值 RCHTRIMH 值越大 RCH 频率越高，step 为 23%
14:8	RCHTRIML	RW	0x0	RCH 时钟修调的细调 TRIM 值 RCHTRIML 值越大 RCH 频率越高，step 为 0.5%
7:5	RSV	-	-	保留
4	RCHRDY	RO	1	内部 RCH 时钟稳定标志 0：RCH 时钟未稳定 1：RCH 时钟稳定，时钟有效
3	RCHDIV	RW	0	选择 16 分频后的输出 0：不分频输出 1：选择 16 分频后输出。如果系统时钟为 RCH，此位置 1 后，系统时钟会变为原来的 1/16。
2:1	RSV	-	-	保留
0	RCHEN	RW	1	RCH 模块使能 0：禁止 1：使能 当 XTH 发生停振时，RCH 使能位将硬件置 1

4.3.25. XTHCR 模块控制寄存器(RCC_XTHCR: 5Ch)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	XTHSDEN	RS	0	XTH 振荡器停振检测使能位： 0：禁止 1：使能 该位不能软件清零，软件仅能置 1（RCC 复位或从 STANDBY 唤醒时清零）； 当 XTHSDEN 置 1 时，停振检测将在 XTH 振荡器已使能且稳定后（XTHRDY 置 1）开启； 当检测到 XTH 停振时，XTHSDEN 由硬件清零
7:5	RSV	-	-	保留
4	XTHRDY	RO	0	XTH 振荡器时钟稳定标志，输出时钟有效。 0：XTH 时钟未稳定 1：XTH 时钟稳定，时钟有效

3:2	XTHRDYTIME	RW	11	XTH 稳定时间设置，以 XTH 时钟计数 00: 1024 个时钟 01: 4096 个时钟 10: 16384 个时钟 11: 32768 个时钟
1	XTHBYP	RW	0	XTH 振荡器旁路使能 0：禁止 XTL 振荡器旁路模式 1：使能 XTL 振荡器旁路模式
0	XTHEN	RW	0	XTH 模块使能 当 XTH 停振时，使能位硬件清零 0：禁止 1：使能

4.3.26. PLL1 模块控制寄存器(RCC_PLL1CR: 60h)

位域	名称	属性	复位值	描述
31	PLL1LOCKSEL	RW	0	PLL1 LOCK 信号选择，用于产生 PLL1 LOCK 中断 0：选择 PLL1LOCK（模拟输出） 1：选择 PLL1FREERUN（数字计数）
30	PLL1FREERUN	RO	0	PLL1 工作状态寄存器。该位即时地反映 PLL1 的工作状态，当 PLL1 进入 SLEEP 模式或重新配置频率后，硬件自动清除该位；当 PLL1 退出 SLEEP 模式且配置生效后，等待 PLL1LOCKDLY 设置时间后，硬件自动置位。 1：PLL1 正常工作。 0：PLL1 正常不正常。
29	PLL1LOCK	RO	0	PLL1 锁定状态位。检测模拟 PLL1 锁定信号的上升沿。 1：PLL1 已经锁定。 0：PLL1 未锁定。
28:23	PLL1LOCKDLY	RW	0x10	PLL1 锁定等待时钟周期数。PLL1 重新重新配置频率后，至少需要 200us(@Fclkvco=100MHz)/ 300us(@Fclkvco=432MHz)的时间才能锁定。该控制位设定的锁定时间计算如下： PLL1LOCKDLY * 512 * （1/HCLK） HCLK 为 4M 时，1 代表等待约 128us。
22	PLL1UPDATEEN	RW	0	PLL1 配置更新控制位，下一个周期自动回 0。重新改写 PLL1P、PLL1Q、PLL1N 和 PLL1F、PLL1SRCSEL 后，需要该该比特置位新的配置才能生效。复位时间 60* （1/HCLK） 1：PLL1 配置更新。 0：PLL1 配置不更新。

21	PLL1SLEEP	RW	1	PLL1 休眠控制位（PLL1 处于复位状态） 1：PLL 进入休眠。 0：PLL 不进入休眠。 注：先使能 PLL1，再退出休眠。
20:6	RSV	-	-	保留
5	PLL1QCLKEN	RW	0	PLL1QCLK 时钟输出使能位 0：禁止 1：使能
4	PLL1PCLKEN	RW	0	PLL1PCLK 时钟输出使能位 0：禁止 1：使能
3:2	RSV	-	-	保留
1	PLL1SRCSEL	RW	0	PLL1 时钟源选择 0：选择片内 RCH（需设置 16 分频模式） 1：选择片外 XTH
0	PLL1EN	RW	0	PLL1 模块使能 0：不使能，PLL1 模块处于 power down 状态 1：使能，PLL1 模块使能 如果 PLL1 用作系统时钟且 PLL1 的时钟源选择 XTH，当 XTH 发生停振时，PLL1 使能位将硬件清零

4.3.27. PLL1 模块配置寄存器(RCC_PLL1CFR: 64h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:24	PLL1Q	RW	00	PLL1QCLK 输出分频控制字段。PLL1Q 含有输出分频控制字段。对应分频值为 PLL1Q 0000：1 0001：1 0010：2 …… 1111：15
23:22	RSV	-	-	保留

21:20	PLL1P	RW	00	PLL1PCLK 输出分频控制字段。PLL1P 含有输出分频控制字段。对应分频值为 2*(PLL1P+1) 00：2 01：4 10：6 11：8
19:18	RSV	-	-	保留
17:12	PLL1N	RW	0x04	降频因子分频器字段。PLL1N 含有 PLL 输入时钟的分频因子，该值的实际作用是参考频率的分频因子。对应分频值为 PLL1N 000000：1 000001：1 000010：2 000011：3 …… 111111：63
11:9	RSV	-	-	保留
8:0	PLL1F	RW	0x1B8	增频因子分频器字段。PLL1F 含有 PLL 反馈回路中分频器的分频因子，该值的实际作用是参考频率的倍频因子，对应倍频值为 PLL1F。 000000000~000110010：50 000110011：51 …… 111111111：511

4.3.28. PLL1 模块扩频控制寄存器(RCC_PLL1SCR: 68h)

位域	名称	属性	复位值	描述
31:17	PLL1SSCSTP	RW	0	PLL1 扩频调制步长配置位。 扩频调制步长用来确定扩频比（即频率抖动范围）： (PLL1SSCPER *PLL1SSCSTP)/(215-1)/ PLL1F 其中，(PLL1SSCPER *PLL1SSCSTP)/(215-1)≤1，扩频率的范围为：0.25%~2%
16:4	PLL1SSCPER	RW	0	PLL1 扩频调制比选择位： 扩频调制比确定 PLL 时钟频率扩展调制周期： Fmod=Fclkpfd/(2*PLL1 SSCPER)，调制频率最大为 10KHz
3:2	RSV	-	-	保留
1	PLL1SSCMD	RW	0	PLL1 扩频类型选择位 0：中心扩频 1：向下扩频

0	PLL1SSCEN	RW	0	PLL1 扩频功能使能位 0：禁止 1：使能

4.3.29. PLL2 模块控制寄存器(RCC_PLL2CR: 6Ch)

位域	名称	属性	复位值	描述
31	PLL2LOCKSEL	RW	0	PLL2 LOCK 信号选择，用于产生 PLL2 LOCK 中断 0：选择 PLL2LOCK（模拟输出） 1：选择 PLL2FREERUN（数字计数）
30	PLL2FREERUN	RO	0	PLL2 工作状态寄存器。该位即时地反映 PLL2 的工作状态，当 PLL2 进入 SLEEP 模式或重新配置频率后，硬件自动清除该位；当 PLL2 退出 SLEEP 模式且配置生效后，等待 PLL2LOCKDLY 设置时间后，硬件自动置位。 1：PLL2 正常工作。 0：PLL2 正常不正常。
29	PLL2LOCK	RO	0	PLL2 锁定状态位。检测模拟 PLL2 锁定信号的上升沿。 1：PLL2 已经锁定。 0：PLL2 未锁定。
28:23	PLL2LOCKDLY	RW	0x10	PLL2 锁定等待时钟周期数。PLL2 重新重新配置频率后，至少需要 200us(@Fclkvco=100MHz)/ 300us(@Fclkvco=432MHz)的时间才能锁定。该控制位设定的锁定时间计算如下： PLL2LOCKDLY * 512 * （1/HCLK） HCLK 为 4M 时，1 代表等待约 128us。
22	PLL2UPDATEEN	RW	0	PLL2 配置更新控制位，下一个周期自动回 0。重新改写 PLL2P、PLL2Q、PLL2N 和 PLL2F、PLL2SRCSEL 后，需要该该比特置位新的配置才能生效。复位时间 60* （1/HCLK） 1：PLL2 配置更新。 0：PLL2 配置不更新。
21	PLL2SLEEP	RW	1	PLL2 休眠控制位（PLL2 处于复位状态） 1：PLL 进入休眠。 0：PLL 不进入休眠。 注：先使能 PLL2，再退出休眠。
20:6	RSV	-	-	保留
5	PLL2QCLKEN	RW	0	PLL2QCLK 时钟输出使能位 0：禁止 1：使能

4	PLL2PCLKEN	RW	0	PLL2PCLK 时钟输出使能位 0：禁止 1：使能
3:2	RSV	-	-	保留
1	PLL2SRCSEL	RW	0	PLL2 时钟源选择 0：选择片内 RCH（需设置 16 分频模式） 1：选择片外 XTH
0	PLL2EN	RW	0	PLL2 模块使能 0：不使能，PLL2 模块处于 power down 状态 1：使能，PLL2 模块使能 如果 PLL2 的时钟源选择 XTH，当 XTH 发生停振时，PLL2 使能位将硬件清零

4.3.30. PLL2 模块配置寄存器(RCC_PLL2CFR: 70h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:24	PLL2Q	RW	00	PLL2QCLK 输出分频控制字段。PLL2Q 含有输出分频控制字段。对应分频值为 PLL2Q 0000：1 0001：1 0010：2 …… 1111：15
23:22	RSV	-	-	保留
21:20	PLL2P	RW	00	PLL2PCLK 输出分频控制字段。PLL2P 含有输出分频控制字段。对应分频值为 2*(PLL2P+1) 00：2 01：4 10：6 11：8
19:18	RSV	-	-	保留

17:12	PLL2N	RW	0x04	降频因子分频器字段。PLL2N 含有 PLL 输入时钟的分频因子，该值的实际作用是参考频率的分频因子。对应分频值为 PLL2N 000000：1 000001：1 000010：2 000011：3 …… 111111：63
11:9	RSV	-	-	保留
8:0	PLL2F	RW	0x1B8	增频因子分频器字段。PLL2F 含有 PLL 反馈回路中分频器的分频因子，该值的实际作用是参考频率的倍频因子，对应倍频值为 PLL2F。 000000000~000110010：50 000110011：51 …… 111111111：511

4.3.31. PLL2 模块扩频控制寄存器(RCC_PLL2SCR: 74h)

位域	名称	属性	复位值	描述
31:17	PLL2SSCSTP	RW	0	PLL2 扩频调制步长配置位。 扩频调制步长用来确定扩频比（即频率抖动范围）： (PLL2SSCPER *PLL2SSCSTP)/(215-1)/ PLL2F 其中，(PLL2SSCPER *PLL2SSCSTP)/(215-1)≤1，扩频率的范围为：0.25%~2%
16:4	PLL2SSCPER	RW	0	PLL2 扩频调制比选择位： 扩频调制比确定 PLL 时钟频率扩展调制周期： Fmod=Fclkpfd/(2* PLL2SSCPER)，调制频率最大为 10KHz
3:2	RSV	-	-	保留
1	PLL2SSCMD	RW	0	PLL2 扩频模式选择位 0：中心模式 1：下降模式
0	PLL2SSCEN	RW	0	PLL2 扩频功能使能位 0：禁止 1：使能

4.3.32. PLL3 模块控制寄存器(RCC_PLL3CR: 78h)

位域	名称	属性	复位值	描述
31	PLL3LOCKSEL	RW	0	PLL3 LOCK 信号选择，用于产生 PLL3 LOCK 中断 0：选择 PLL3LOCK（模拟输出） 1：选择 PLL3FREERUN（数字计数）
30	PLL3FREERUN	RO	0	PLL3 工作状态寄存器。该位即时地反映 PLL3 的工作状态，当 PLL3 进入 SLEEP 模式或重新配置频率后，硬件自动清除该位；当 PLL3 退出 SLEEP 模式且配置生效后，等待 PLL3LOCKDLY 设置时间后，硬件自动置位。 1：PLL3 正常工作。 0：PLL3 正常不正常。
29	PLL3LOCK	RO	0	PLL3 锁定状态位。检测模拟 PLL3 锁定信号的上升沿。 1：PLL3 已经锁定。 0：PLL3 未锁定。
28:23	PLL3LOCKDLY	RW	0x10	PLL3 锁定等待时钟周期数。PLL3 重新重新配置频率后，至少需要 100us 的时间才能锁定。该控制位设定的锁定时间计算如下： PLL3LOCKDLY * 512 * （1/HCLK） HCLK 为 32M 时，1 代表等待约 16us。
22	PLL3UPDATEEN	RW	0	PLL3 配置更新控制位，下一个周期自动回 0。重新改写 PLL3P、PLL3Q、PLL3N 和 PLL3F、PLL3SRCSEL 后，需要该该比特置位新的配置才能生效。复位时间 60* （1/HCLK） 1：PLL3 配置更新。 0：PLL3 配置不更新。
21	PLL3SLEEP	RW	1	PLL3 休眠控制位（PLL3 处于复位状态） 1：PLL 进入休眠。 0：PLL 不进入休眠。 注：先使能 PLL3，再退出休眠。
20:6	RSV	-	-	保留
5	PLL3QCLKEN	RW	0	PLL3QCLK 时钟输出使能位 0：禁止 1：使能
4	PLL3PCLKEN	RW	0	PLL3PCLK 时钟输出使能位 0：禁止 1：使能
3:2	RSV	-	-	保留
1	PLL3SRCSEL	RW	0	PLL3 时钟源选择 0：选择片内 RCH（需设置 16 分频模式） 1：选择片外 XTH

0	PLL3EN	RW	0	PLL3 模块使能 0：不使能，PLL3 模块处于 power down 状态 1：使能，PLL3 模块使能 如果 PLL3 的时钟源选择 XTH，当 XTH 发生停振时，PLL3 使能位将硬件清零

4.3.33. PLL3 模块配置寄存器(RCC_PLL3CFR: 7Ch)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25:24	PLL3Q	RW	00	PLL3QCLK 输出分频控制字段。PLL3Q 含有输出分频控制字段。对应分频值为 2**PLL3Q 00：1 01：2 10：4 11：8
23:22	RSV	-	-	保留
21:20	PLL3P	RW	01	PLL3PCLK 输出分频控制字段。PLL3P 含有输出分频控制字段。对应分频值为 2**PLL3P 00：1 01：2 10：4 11：8
19:18	RSV	-	-	保留
17:12	PLL3N	RW	0x0	降频因子分频器字段。PLL3N 含有 PLL 输入时钟的分频因子，该值的实际作用是参考频率的分频因子。对应分频值为 PLL3N+1 000000：1 000001：2 000010：3 000011：4 …… 111111：64
11:7	RSV	-	-	保留

6:0	PLL3F	RW	0x6d	增频因子分频器字段。PLL3F 含有 PLL 反馈回路中分频器的分频因子，该值的实际作用是参考频率的倍频因子，对应倍频值为 PLL3F+1。 0x00：1 倍频 0x01：2 倍频 …… 0x18：25 倍频 …… 0x7F：128 倍频

4.3.34. 时钟输出控制寄存器(RCC_CLKOCR: 84h)

位域	名称	属性	复位值	描述
31	MCO2EN	RW	0	MCO2 预分频器输出使能 0：不使能（输出电平由 MCO2POL 位决定） 1：使能
30	MCO2POL	RW	0	MCO2 预分频器时钟输出极性选择 0：原极性（停止时为 0） 1：反极性（停止时为 1）
29:24	MCO2DIV	RW	0x3F	MCO2 预分频器分频系数： 分频数为寄存器值 + 1
23	MCO1EN	RW	1	MCO1 预分频器输出使能 0：不使能（输出电平由 MCO1POL 位决定） 1：使能
22	MCO1POL	RW	0	MCO1 预分频器时钟输出极性选择 0：原极性（停止时为 0） 1：反极性（停止时为 1）
21:6	MCO1DIV	RW	0x004F	MCO1 预分频器分频系数： 分频数为寄存器值 + 1
5	MCO1SEL	RW	1	MCO1 输出选择 0：旁路 MCO1 预分频器(直接来自 MCOCLKS 选择信号) 1：MCO1 预分频器输出

4:0	MCOCLKS	RW	0000	MCO1/MCO2 时钟源选择 00000：HCLK 00001：RCH 00010：RCL 00011：XTH 00100：XTL 00101：PLL1PCLK 00110：PLL2PCLK 00111：PLL2QCLK 01000：PLL3PCLK 01001：PLL3QCLK 01010：SYSCLK 01011：LPUART1CLK 01100：FCLK/8 01101：USB1UTMICLK48M 01110：USB2UTMICLK48M 01111：RTCPCLK 10000：SDMMC_SAMPLE_CLK 10001：SDMMC_DRIVE_CLK

4.3.35. 外设专用时钟配置寄存器(RCC_DCKCFG: 88h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	LCDDIV	RW	00	LCD_CLK 像素时钟分频系数配置位 00：2 分频 01：4 分频 10：8 分频 11：16 分频

4.3.36. STANDBY 电源域控制寄存器(RCC_STDBYCTRL: 8Ch)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23	STDBYRST	RW	1	STANDBY 电源域软复位，写 0 后下一拍自动变 1 0：复位 STANDBY 域 1：不复位 STANDBY 域

22	RTCEN	RW	1	RTCCLK 使能 0：不使能 1：使能
21:20	RTCSEL	RW	00	RTCCLK 选择 00：RCL 时钟 01：XTL 时钟 1x：没有时钟
19:16	RCLDIS	WO	0	RCL 时钟关闭码值： 该位写入 0xA，同时 bit8 写入 1’ b0，可将 RCL 时钟关闭
15:10	RCLTRIM	RW	0x20	RCL 模块的 TRIM 值 TRIM 值增大，RCL 时钟增加。
9	RCLRDY	RO	1	RCL 时钟 ready。RCL 时钟稳定标志。在 RCLEN 位清除后，RCLRDY 在额外 1 个 RCL CLK 后变低 0：RCL 时钟未稳定 1：RCL 时钟稳定，时钟有效
8	RCLEN	RW	1	RCL 时钟使能 0：RCL 不使能 1：RCL 使能
7	RSV	RO	0	
6	XTLSDEN	RS	0	XTL 振荡器停振检测使能位： 0：禁止 1：使能 当 XTLSDEN 置 1 时，停振检测将在 XTL 振荡器已使能且稳定后（XTLRDY 置 1）开启； 当检测到 XTL 停振时，XTHSDEN 才能由软件清零
5:3	XTLDRV	RW	111	XTL 振荡器驱动能力选择 XTLDRV[2]决定功耗： 0：正常功耗 1：低功耗 XTLDRV[1:0[决定功耗： 00：XTL 低驱动 … 11：XTL 高驱动
2	XTLBYP	RW	0	XTL 振荡器旁路使能 0：禁止 XTL 振荡器旁路模式 1：使能 XTL 振荡器旁路模式

1	XTLRDY	RO	0	XTL 振荡器 ready。XTL 振荡器时钟稳定标志，输出时钟有效。在 XTLEN 位清除后，XTLRDY 在额外 6 个 XTLCLK 后变低 0：XTL 时钟未稳定 1：XTL 时钟稳定，时钟有效
0	XTLEN	RW	0	XTL 振荡器使能 0：XTL 不使能 1：XTL 使能

注 1：进行写操作后，需要等待至少 1 个 AHB 周期后才能读取

5. 嵌套向量中断控制器（NVIC）

5.1. 概述

嵌套向量中断控制器(NVIC) 是内核处理器的一个重要组成部分。它与 CPU 处理器内核紧密耦合，实现低中断延迟以及对新到中断的有效处理，外部中断信号连接到 NVIC，NVIC 将对这些中断进行优先级排序。

所有的 NVIC 寄存器只能采用字传输。任何试图读/写半字或字节的结果都是不可预知的。

NVIC 寄存器都是小端格式。访问处理器要正确处理处理器的大小端配置。

(关于 NVIC 更详细的内容可查看 ARM China STAR Processor User Guild 官方文档)

5.2. 主要特性

- 支持 142 路可屏蔽向量中断
- 16 个可编程中断优先级
- 可嵌套中断支持
- 中断可屏蔽
- 电平触发和边沿触发

5.3. 中断源

表 5-1 NVIC 中断源

NVIC 位置	优先级	优先级类型	名称	说明	地址
-	-	-	-	保留	0x000
-	-3	固定	Reset	复位	0x004
-	-2	固定	NMI	不可屏蔽中断	0x008
-	-1	固定	HardFault	所有类型的错误	0x00C
-	0	可设置	MemManage	存储器管理	0x010
-	1	可设置	BusFault	预取故障，存储器访问故障	0x014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x018
-	-	-	-	保留	0x01C-0x02B
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x02C
-	4	可设置	Debug Monitor	调试监控器	0x030
-	-	-	-	保留	0x034
-	5	可设置	PendSV	可挂起的系统服务	0x038
-	6	可设置	SysTick	系统节拍定时器	0x03C
0	7	可设置	WDT		0x040
1	8	可设置	LVD	连到 EXTI 线 16 的电源电压检测（LVD）中断	0x044
2	9	可设置	RTC_XTLSD	RTC 全局中断/XTL 振荡器停振检测中断	0x048

NVIC 位置	优先级	优先级类型	名称	说明	地址
3	10	可设置	保留		0x04C
4	11	可设置	保留		0x050
5	12	可设置	保留		0x054
6	13	可设置	CLKRDY_INT	时钟 Ready 中断	0x058
7	14	可设置	EXTI0	EXTI 线 0 中断	0x05C
8	15	可设置	EXTI1	EXTI 线 1 中断	0x060
9	16	可设置	EXTI2	EXTI 线 2 中断	0x064
10	17	可设置	EXTI3	EXTI 线 3 中断	0x068
11	18	可设置	EXTI4	EXTI 线 4 中断	0x06C
12	19	可设置	DMA1	DMA1 全局中断	0x070
13	20	可设置	DMA2	DMA2 全局中断	0x074
14	21	可设置	ADC1_2	ADC1 和 ADC2 全局中断	0x078
15	22	可设置	ADC3	ADC3 全局中断	0x07C
16	23	可设置	DAC1	DAC1 全局中断	0x080
17	24	可设置	COMP1	连到 EXTI 线 19 的 COMP1 中断	0x084
18	25	可设置	USB1	USB1 全局中断	0x088
19	26	可设置	FDCAN1		0x08C
20	27	可设置	FDCAN2		0x090
21	28	可设置	EXTI9_5	EXTI 线 9~5 中断	0x094
22	29	可设置	TIM1_BRK_UP_TRG_COM		0x098
23	30	可设置	TIM1_CC	捕获/比较中断	0x09C
24	31	可设置	TIM2		0x0A0
25	32	可设置	TIM3		0x0A4
26	33	可设置	TIM6		0x0A8
27	34	可设置	TIM7		0x0AC
28	35	可设置	TIM8_BRK_UP_TRG_COM		0x0B0
29	36	可设置	TIM8_CC	TIM8 捕获/比较中断	0x0B4
30	37	可设置	TIM15		0x0B8
31	38	可设置	TIM16		0x0BC
32	39	可设置	TIM17		0x0C0
33	40	可设置	I2C1		0x0C4
34	41	可设置	I2C2		0x0C8
35	42	可设置	(Q)SPI1		0x0CC
36	43	可设置	(Q)SPI2		0x0D0
37	44	可设置	(Q)SPI3		0x0D4

NVIC 位置	优先级	优先级类型	名称	说明	地址
38	45	可设置	I2S1		0x0D8
39	46	可设置	I2S2		0x0DC
40	47	可设置	UART1		0x0E0
41	48	可设置	UART2		0x0E4
42	49	可设置	UART3		0x0E8
43	50	可设置	UART4		0x0EC
44	51	可设置	EXTI15_10	EXTI 线 15~10 中断	0x0F0
45	52	可设置	USB1_WKUP	连到 EXTI 线 23 的 USB1 唤醒中断	0x0F4
46	53	可设置	LPUART1		0x0F8
47	54	可设置	LPTIM1	LPTIM1 全局中断	0x0FC
48	55	可设置	USB2_WKUP	连到 EXTI 线 24 的 USB2 唤醒中断	0x100
49	56	可设置	AES		0x104
50	57	可设置	FPU_INT		0x108
51	58	可设置	USB2	USB2 全局中断	0x10C
52	59	可设置	DCMI		0x110
53	60	可设置	TIM4		0x114
54	61	可设置	保留		0x118
55	62	可设置	IWDT_WKUP	连到 EXTI 线 18 的 IWDT 唤醒中断	0x11C
56	63	可设置	LTDC_INT	LTDC 全局中断	0x120
57	64	可设置	LTDC_ERR	LTDC 全局错误中断	0x124
58	65	可设置	DMA2D		0x128
59	66	可设置	LPTIM2	LPTIM2 全局中断	0x12C
60	67	可设置	LPTIM3	LPTIM3 全局中断	0x130
61	68	可设置	LPTIM4	LPTIM4 全局中断	0x134
62	69	可设置	LPTIM5	LPTIM5 全局中断	0x138
63	70	可设置	LPTIM6	LPTIM6 全局中断	0x13C
64	71	可设置	AES_SPI1		0x140
65	72	可设置	I2S3		0x144
66	73	可设置	(Q)SPI4		0x148
67	74	可设置	(Q)SPI5		0x14C
68	75	可设置	(Q)SPI6		0x150
69	76	可设置	I2C3		0x154
70	77	可设置	I2C4		0x158
71	78	可设置	FDCAN3		0x15C
72	79	可设置	保留		0x160

NVIC 位置	优先级	优先级类型	名称	说明	地址
73	80	可设置	ETH	ETHMAC 全局中断	0x164
74	81	可设置	ETH_WKUP	连到 EXTI 线 33 的 ETH 唤醒中断	0x168
75	82	可设置	SDMMC		0x16C
76	83	可设置	UART5		0x170
77	84	可设置	UART6		0x174
78	85	可设置	UART7		0x178
79	86	可设置	UART8		0x17C
80	87	可设置	UART9		0x180
81	88	可设置	UART10		0x184
82	89	可设置	DAC2	DAC2 全局中断	0x188
83	90	可设置	TIM5		0x18C
84	91	可设置	TIM9		0x190
85	92	可设置	TIM10		0x194
86	93	可设置	TIM11		0x198
87	94	可设置	TIM12		0x19C
88	95	可设置	TIM13		0x1A0
89	96	可设置	TIM14		0x1A4
90	97	可设置	TIM18		0x1A8
91	98	可设置	TIM19		0x1AC
92	99	可设置	TIM20_BRK_UP_TRG_COM		0x1B0
93	100	可设置	TIM20_CC	TIM20 捕获/比较中断	0x1B4
94	101	可设置	TIM21		0x1B8
95	102	可设置	TIM22		0x1BC
96	103	可设置	TIM23		0x1C0
97	104	可设置	TIM24		0x1C4
98	105	可设置	TIM25		0x1C8
99	106	可设置	TIM26		0x1CC
100	107	可设置	(Q)SPI7		0x1D0
101	108	可设置	(Q)SPI8		0x1D4
102	109	可设置	OSPI1		0x1D8
103	110	可设置	OSPI2		0x1DC
104	111	可设置	保留		0x1E0
105	112	可设置	TKEY		0x1E4
106	113	可设置	LPT	行式打印机并口模块全局中断	0x1E8
107	114	可设置	保留		0x1EC

NVIC 位置	优先级	优先级类型	名称	说明	地址
108	115	可设置	OTG1_HS_EP_OUT	USB1 OTG HS 端点输出全局中断	0x1F0
109	116	可设置	OTG1_HS_EP_IN	USB1 OTG HS 端点输入全局中断	0x1F4
110	117	可设置	OTG2_HS_EP_OUT	USB2 OTG HS 端点输出全局中断	0x1F8
111	118	可设置	OTG2_HS_EP_IN	USB2 OTG HS 端点输入全局中断	0x1FC
112	119	可设置	NDL	针式打印机控制器全局中断	0x200
113	120	可设置	THM	热敏打印机控制器全局中断	0x204
114	121	可设置	STM1_PWM	步进电机 PWM 控制器 1 全局中断	0x208
115	122	可设置	STM2_PWM	步进电机 PWM 控制器 2 全局中断	0x20C
116	123	可设置	DCM1_PWM	直流电机 PWM 控制器 1 全局中断	0x210
117	124	可设置	DCM2_PWM	直流电机 PWM 控制器 2 全局中断	0x214
118	125	可设置	DCM3_PWM	直流电机 PWM 控制器 3 全局中断	0x218
119	126	可设置	DCM4_PWM	直流电机 PWM 控制器 4 全局中断	0x21C
120	127	可设置	STM3_PWM	步进电机 PWM 控制器 3 全局中断	0x220
121	128	可设置	STM4_PWM	步进电机 PWM 控制器 4 全局中断	0x224
122	129	可设置	STM5_PWM	步进电机 PWM 控制器 5 全局中断	0x228
123	130	可设置	STM6_PWM	步进电机 PWM 控制器 6 全局中断	0x22C
124	131	可设置	DCM5_PWM	直流电机 PWM 控制器 5 全局中断	0x230
125	132	可设置	DCM6_PWM	直流电机 PWM 控制器 6 全局中断	0x234
126	133	可设置	NAND	NAND 存储控制器中断	0x238
127	134	可设置	BCH	BCH 中断	0x23C
128	135	可设置	SDRAM	SDRAM 控制器全局中断	0x240
129	136	可设置	DMA1_CH0	DMA1 通道 0 全局中断	0x244
130	137	可设置	DMA1_CH1	DMA1 通道 1 局中断	0x248
131	138	可设置	DMA1_CH2	DMA1 通道 2 全局中断	0x24C
132	139	可设置	DMA1_CH3	DMA1 通道 3 全局中断	0x250
133	140	可设置	DMA1_CH4	DMA1 通道 4 全局中断	0x254
134	141	可设置	DMA1_CH5	DMA1 通道 5 全局中断	0x258
135	142	可设置	DMA1_CH6	DMA1 通道 6 全局中断	0x25C
136	143	可设置	DMA1_CH7	DMA1 通道 7 全局中断	0x260
137	144	可设置	DMA2_CH0	DMA2 通道 0 全局中断	0x264
138	145	可设置	DMA2_CH1	DMA2 通道 1 局中断	0x268
139	146	可设置	DMA2_CH2	DMA2 通道 2 全局中断	0x26C
140	147	可设置	DMA2_CH3	DMA2 通道 3 全局中断	0x270
141	148	可设置	DMA2_CH4	DMA2 通道 4 全局中断	0x274
142	149	可设置	DMA2_CH5	DMA2 通道 5 全局中断	0x278

NVIC 位置	优先级	优先级类型	名称	说明	地址
143	150	可设置	DMA2_CH6	DMA2 通道 6 全局中断	0x27C
144	151	可设置	DMA2_CH7	DMA2 通道 7 全局中断	0x280
145	152	可设置	SRAM1_SEC	SRAM1 ECC 单 bit 错误纠正中断	0x284
146	153	可设置	SRAM1_DED	SRAM1 ECC 双 bit 错误检测中断	0x288
147	154	可设置	SRAM3_SEC	SRAM3 ECC 单 bit 错误纠正中断	0x28C
148	155	可设置	SRAM3_DED	SRAM3 ECC 双 bit 错误检测中断	0x290
149	156	可设置	BKPSRAM_SEC	Backup SRAM ECC 单 bit 错误纠正中断	0x294
150	157	可设置	BKPSRAM_DED	Backup SRAM ECC 双 bit 错误检测中断	0x298

6. 外部中断/事件控制器（EXTI）

6.1. 概述

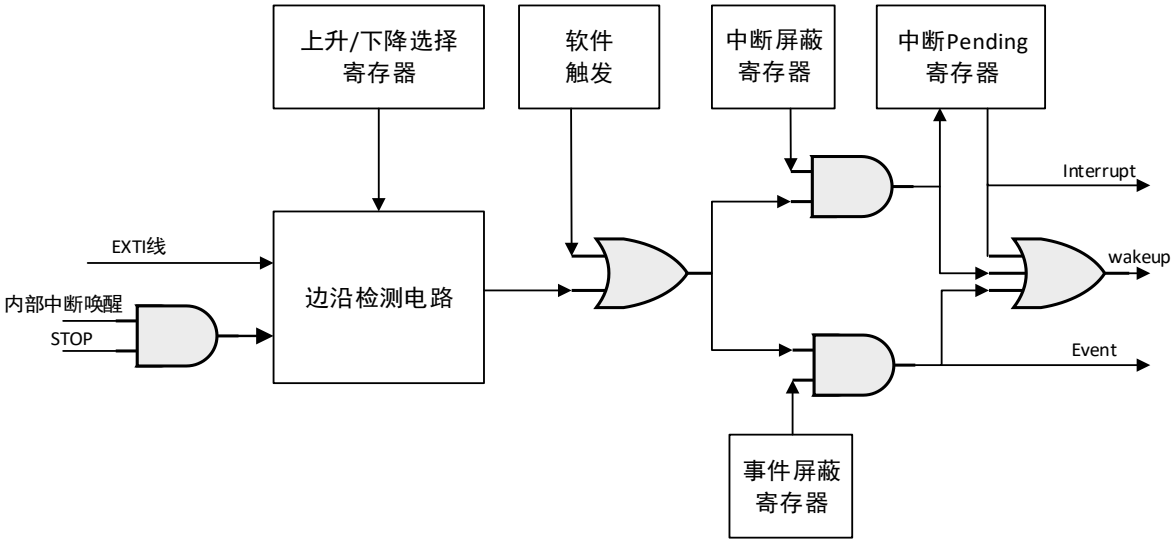
EXTI 包含 31 个相互独立的边沿检测电路并且可以向处理器产生中断请求或事件唤醒。EXTI 提供 3 种触发类型，其中请求源 0~15 为 GPIO 管脚可支持上升沿触发、下降沿触发和任意沿触发，其他请求源默认使用上升沿触发。EXTI 中每个边沿检测电路都可以分别配置或屏蔽。挂起寄存器保持着状态线的中断请求。

6.2. 主要特性

- EXTI 支持多达 31 个软件的中断/事件请求
- 每个中断/事件都有独立的触发和屏蔽
- 每个中断线都有专用的状态位

6.3. 结构框图

图 6-1 框图



6.4. 触发源

EXTI 触发源包括来自 I/O 管脚的 16 根线以及来自内部模块的 15 根线，包括 LVD、 RTC、 LPUART、IWDT（唤醒）、COMP、LPTIM 和 USB_Wakeup。 通过配置 EXTICR 寄存器，所有的 GPIO 管脚都可以被选作 EXTI 的触发源。

表 6-1 EXTI 触发源

EXTI 序号	对应源
0	PA0~ PQ0
1	PA1~PQ1

2	PA2~PQ2
3	PA3~PQ3
4	PA4~PQ4
5	PA5~PQ5
6	PA6~PQ6
7	PA7~PQ7
8	PA8~PQ8
9	PA9~PQ9
10	PA10~PQ10
11	PA11~PQ11
12	PA12~PQ12
13	PA13~PQ13
14	PA14~PQ14
15	PA15~PQ15
16	LVD
17	RTC(中断)/XTL 停振
18	IWDT
19	COMP1
20	Reserved
21	Reserved
22	Reserved
23	USB1_Wakeup
24	USB2_Wakeup
25	LPTIM1_Wakeup（中断）
26	LPTIM2_Wakeup（中断）
27	LPTIM3_Wakeup（中断）
28	LPTIM4_Wakeup（中断）
29	LPTIM5_Wakeup（中断）
30	LPTIM6_Wakeup（中断）
31	LPUART1_Wakeup（中断）

32	Reserved
33	ETH_Wakeup
34	TKEY
35~63	Reserved

6.5. 唤醒管理

唤醒事件可以通过下述配置产生：

- 在 SLEEP 模式下，在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时在 Core 的系统控制寄存器中使能 SEVONPEND 位。当 CPU 从 WFE 恢复后，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位(在 NVIC 中断清除挂起寄存器中)。
- 在 SLEEP 模式下，在外设的控制寄存器使能一个中断，同时在 NVIC 中使能。当 CPU 从 SLEEP 恢复后，通过中断处理函数处理相应中断。
- 在 STOP 模式下，配置一个外部或内部 EXTI 线为事件模式(EENR)，当 CPU 从 WFE 恢复后，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。
- 在 STOP 模式下，使能 EXTI 的 NVIC 中断(INT5)，根据外部 EXTI 线的边沿检测需求设置 2 个触发寄存器(RTENR/FTENR)，同时在中断屏蔽寄存器的相应位写 ‘1’ 允许中断请求(IENR)。当外部中断线上发生了期待的边沿时，将产生一个 EXTI 中断请求，对应的挂起位(PDR)也随之被置 ‘1’。

6.6. 功能描述

6.6.1. EXTI 中断

EXTI 具有 31 个触发源，每个触发源都可以触发 EXTI 中断。

■ EXTI 中断配置流程：

- 配置 EXTICR1、EXTICR2，选择 GPIO 引脚作为触发源；
- 配置 RTENR、FTENR 选择上升沿触发、下降沿触发或任意沿触发；
- 配置 IENR，使能所需的中断线；
- 当线上发生了需要的边沿时，PDR 寄存器相应的位会置位，进入 EXTI 中断。

注：通过在寄存器 EXTI_SWIER 对应位写 ‘1’，也可以通过软件产生中断

6.6.2. EXTI 中断/事件唤醒

EXTI 可以通过中断或者事件唤醒 MCU。

■ 硬件中断/事件源选择

通过下面的过程来配置 31 个线路做为中断/事件源：

- 配置 EXTICR1、EXTICR2，选择 GPIO 引脚作为触发源。
- 配置 31 个中断线的使能位 IENR 或配置 31 个事件线的使能位 EENR。
- 配置 RTENR、FTENR 选择上升沿触发、下降沿触发。

- 如果使用中断唤醒，需要配置 NVIC 控制寄存器中相应的使能位和屏蔽位，使得 31 个中断线中的请求可以被正确地响应。

6.7. EXTI 寄存器描述

6.7.1. 寄存器列表

EXTI 寄存器基地址：0x40010400

偏移	名称	复位值	描述
0x0000	IENR1	0xfe000000	中断使能寄存器 1
0x0004	IENR2	0x00000001	中断使能寄存器 2
0x0008	EENR1	0x00000000	事件使能寄存器 1
0x000C	EENR2	0x00000000	事件使能寄存器 2
0x0010	RTENR1	0xfe000000	上升沿触发使能寄存器 1
0x0014	RTENR2	0x00000001	上升沿触发使能寄存器 2
0x0018	FTENR1	0x00000000	下降沿触发使能寄存器 1
0x001C	FTENR2	0x00000000	下降沿触发使能寄存器 2
0x0020	SWIER1	0x00000000	软件中断事件寄存器 1
0x0024	SWIER2	0x00000000	软件中断事件寄存器 2
0x0028	PDR1	0x00000000	中断挂起寄存器 1
0x002C	PDR2	0x00000000	中断挂起寄存器 2
0x0030	EXTICR1	0x00000000	外部 I/O 选择寄存器 1
0x0034	EXTICR2	0x00000000	外部 I/O 选择寄存器 2
0x0038	EXTICR3	0x00000000	外部 I/O 选择寄存器 3

6.7.2. 中断使能寄存器(IENR1: 00h)

位域	名称	属性	复位值	描述
31:0	INTENX	RW	0xFE000000	EXTIx(x=0~31)上的中断使能 0：禁止来自线 x 上的中断请求 1：使能来自线 x 上的中断请求

6.7.3. 中断使能寄存器(IENR2: 04h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留

2:0	INTENX	RW	0x01	EXTIx(x=32~34)上的中断使能 0：禁止来自线 x 上的中断请求 1：使能来自线 x 上的中断请求

6.7.4. 事件使能寄存器(EENR1: 08h)

位域	名称	属性	复位值	描述
31:0	EVENX	RW	0x0	EXTIx(x=0~31)上的事件使能 0：禁止来自线 x 上的事件请求 1：使能来自线 x 上的事件请求

6.7.5. 事件使能寄存器(EENR2: 0Ch)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	EVENX	RW	0x0	EXTIx(x=32~34)上的事件使能 0：禁止来自线 x 上的事件请求 1：使能来自线 x 上的事件请求

6.7.6. 上升沿触发使能寄存器(RTENR1: 10h)

位域	名称	属性	复位值	描述
31:0	RTENX	RW	0xFE000000	EXTIx(x=0~31)上的上升沿触发使能 0：禁止来自线 x 上的上升沿触发 1：使能来自线 x 上的上升沿触发

注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

6.7.7. 上升沿触发使能寄存器(RTENR2: 14h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	RTENX	RW	0x01	EXTIx(x=32~34)上的上升沿触发使能 0：禁止来自线 x 上的上升沿触发 1：使能来自线 x 上的上升沿触发

注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

6.7.8. 下降沿触发使能寄存器(FTENR1: 18h)

位域	名称	属性	复位值	描述
31:0	FTENX	RW	0x0	EXTIx(x=0~31)上的下降沿触发使能 0：禁止来自线 x 上的下降沿触发 1：使能来自线 x 上的下降沿触发

注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

6.7.9. 下降沿触发使能寄存器(FTENR2: 1Ch)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	FTENX	RW	0x0	EXTIx(x=32~34)上的下降沿触发使能 0：禁止来自线 x 上的下降沿触发 1：使能来自线 x 上的下降沿触发

注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

6.7.10. 软件中断事件寄存器(SWIER1: 20h)

位域	名称	属性	复位值	描述
31:0	SWIEX	WO	0x0	EXTIx(x=0~31)上的软件中断 当该位为‘0’时，写‘1’将设置 PDR 中相应的挂起位。如果在 IENR 中允许产生该中断，则此时将产生一个中断

6.7.11. 软件中断事件寄存器(SWIER2: 24h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	SWIEX	WO	0x0	EXTIx(x=32~34)上的软件中断 当该位为‘0’时，写‘1’将设置 PDR 中相应的挂起位。如果在 IENR 中允许产生该中断，则此时将产生一个中断

6.7.12. 中断挂起寄存器(PDR1: 28h)

位域	名称	属性	复位值	描述
31:0	PDX	RC_W1	0x0	EXTIx(x=0~31)上的挂起位 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写‘1’可以清除它

6.7.13. 中断挂起寄存器(PDR2: 2Ch)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	PDX	RC_W1	0x0	EXTIx(x=32~34)上的挂起位 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写‘1’可以清除它

6.7.14. 外部中断配置寄存器(1 EXTICR1: 30h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留

29:0	EXTIX[4:0]	RW	0000	EXTIx(x=5~0)源选择 00000: PA[x]引脚 00001: PB[x]引脚 00010: PC[x]引脚 00011: PD[x]引脚 00100: PE[x]引脚 00101: PF[x]引脚 00110: PG[x]引脚 00111: PH[x]引脚 01000: PI[x]引脚 01001: PJ[x]引脚 01010: PK[x]引脚 01011: PL[x]引脚 01100: PM[x]引脚 01101: PN[x]引脚 01110: PO[x]引脚 01111: PP[x]引脚 10000: PQ[x]引脚

6.7.15. 外部中断配置寄存器(2 EXTICR2: 34h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留

29:0	EXTIX[4:0]	RW	0000	EXTIx(x=11~6)源选择 00000: PA[x]引脚 00001: PB[x]引脚 00010: PC[x]引脚 00011: PD[x]引脚 00100: PE[x]引脚 00101: PF[x]引脚 00110: PG[x]引脚 00111: PH[x]引脚 01000: PI[x]引脚 01001: PJ[x]引脚 01010: PK[x]引脚 01011: PL[x]引脚 01100: PM[x]引脚 01101: PN[x]引脚 01110: PO[x]引脚 01111: PP[x]引脚 10000: PQ[x]引脚

6.7.16. 外部中断配置寄存器(3 EXTICR3: 38h)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留

19:0	EXTIX[4:0]	RW	0000	EXTIx(x=15~12)源选择 00000: PA[x]引脚 00001: PB[x]引脚 00010: PC[x]引脚 00011: PD[x]引脚 00100: PE[x]引脚 00101: PF[x]引脚 00110: PG[x]引脚 00111: PH[x]引脚 01000: PI[x]引脚 01001: PJ[x]引脚 01010: PK[x]引脚 01011: PL[x]引脚 01100: PM[x]引脚 01101: PN[x]引脚 01110: PO[x]引脚 01111: PP[x]引脚 10000: PQ[x]引脚

7. DMA 控制器（DMA）

7.1. 概述

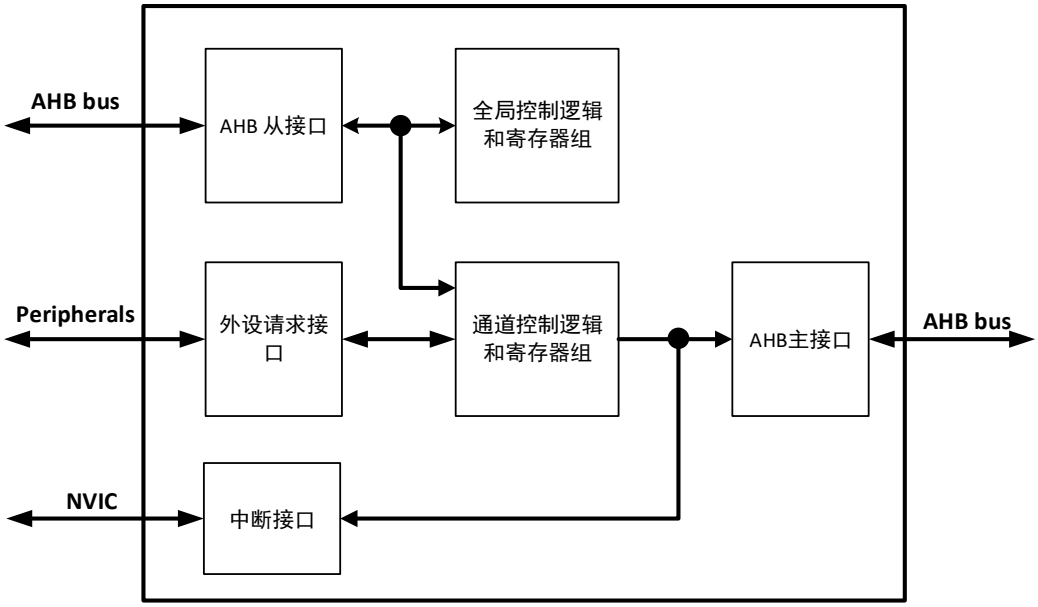
DMA 控制器提供了一种硬件的数据传输方式，无需 CPU 的介入，可以处理外设和存储器之间或者存储器和存储器之间的传输数据。因无 CPU 介入，从而使 CPU 可以专注在处理其他系统功能上。DMA 控制器有 8 通道，每个通道都可以处理一个或多个外设的存储器访问请求，并且每个通道都有各自独立的 FIFO。DMA 控制器内部包含了仲裁器，用来仲裁多个 DMA 请求的优先级。

7.2. 主要特性

- 每个 DMA 控制器含有 8 个 DMA 通道，每个通道都可独立配置
- 支持外设到存储器、存储器到外设、存储器到存储器的数据传输
- 每个通道连接的硬件 DMA 请求可有软件配置
- 支持硬件优先级（通道 0 拥有最高优先级，通道 7 拥有最低优先级）
- 支持源地址/目标地址递增或不变
- 每个通道有 16 bytes 的内部 FIFO
- 支持字节/半字/字的传输，源和目标的数据宽度不相等时，DMA 自动封装/解封必要的传输数据来优化带宽
- 支持 Burst 功能（每次外设请求一次，DMA 需要传输的数目）
- 支持链表功能，可以实现循环传输、双缓冲传输
- 支持大端模式和小端模式
- 支持 DMA 中断功能
- 原始中断：内部中断信号，不管对应通道的中断是否使能，都会置位
- 屏蔽后中断：原始中断和通道中断使能进行逻辑与的结果
- 可控的传输数目，最少支持 1，最多支持 65535

7.3. 功能说明

7.3.1. 框图



DMA 包含 AHB 从接口、AHB 主接口、外设请求接口、中断接口、全局控制逻辑和全局寄存器组、通道逻辑和通道寄存器组。

- AHB 从接口：CPU 通过此接口读写 DMA 内部的寄存器组。
- AHB 主接口：此接口用于从源存储器中读取数据，也用于向目标存储器中写入数据。
- 外设请求接口：此接口负责响应外设请求。
- 中断接口:此接口用户产生各种 DMA 的中断，发送给 NVIC 模块。
- 全局控制逻辑和寄存器组: 该模块包括全局的控制逻辑和全局的寄存器组。
- 通道控制逻辑和寄存器组: 该模块包括各通道的控制逻辑和各通道的寄存器组。

7.3.2. 通道

DMA 包含 8 个通道，每个通道支持一个单向的数据流，通道内嵌 16 字节的 FIFO。通过设置各自通道寄存器来控制此单向数据流的属性。

7.3.3. 外设请求

DMA1 和 DMA2 各自有 256 个外设请求，每个 DMA 内部的通道共享此 256 个外设请求。

表 7-1 目标外设和源外设请求号

DMA1 请求					
请求号	**请求源**	**请求号**	**请求源**	**请求号**	**请求源**
REQ0	ADC1	REQ64	STM4_PWM	REQ128	UART5_RX
REQ1	SPI1_TX	REQ65	STM5_PWM	REQ129	UART6TX
REQ2	SPI1_RX	REQ66	STM6_PWM	REQ130	UART6_RX

DMA1 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ3	SPI2_TX	REQ67	LPT	REQ131	UART7_TX
REQ4	SPI2_RX	REQ68	TIM5_UP	REQ132	UART7_RX
REQ5	UART1_TX	REQ69	TIM5_CH1	REQ133	UART8_TX
REQ6	UART1_RX	REQ70	TIM5_CH2	REQ134	UART8_RX
REQ7	UART2_TX	REQ71	TIM5_CH3	REQ135	UART9_TX
REQ8	UART2_RX	REQ72	TIM5_CH4	REQ136	UART9_RX
REQ9	I2C1_TX	REQ73	TIM5_TRIG	REQ137	UART10_TX
REQ10	I2C1_RX	REQ74	TIM5_COM	REQ138	UART10_RX
REQ11	I2C2_TX	REQ75	TIM25_UP	REQ139	FDCAN1_TX
REQ12	I2C2_RX	REQ76	TIM25_CH1	REQ140	FDCAN1_RX
REQ13	TIM1_CH1	REQ77	TIM25_CH2	REQ141	FDCAN2_TX
REQ14	TIM1_CH2	REQ78	TIM25_TRIG	REQ142	FDCAN2_RX
REQ15	TIM1_CH3	REQ79	TIM25_COM	REQ143	FDCAN3_TX
REQ16	TIM1_CH4	REQ80	TIM20_UP	REQ144	FDCAN3_RX
REQ17	TIM1_UP	REQ81	TIM20_CH1	REQ145	SPI6_RX
REQ18	TIM1_TRIG	REQ82	TIM20_CH2	REQ146	SPI6_TX
REQ19	TIM1_COM	REQ83	TIM20_CH3	REQ147	
REQ20	TIM2_CH1	REQ84	TIM20_CH4	REQ148	
REQ21	TIM2_CH2	REQ85	TIM20_CH5	REQ149	
REQ22	TIM2_CH3	REQ86	TIM20_CH6	REQ150	
REQ23	TIM2_CH4	REQ87	TIM20_TRIG	REQ151	
REQ24	TIM2_UP	REQ88	TIM20_COM	REQ152	
REQ25	TIM2_TRIG	REQ89	TIM18_UP	REQ153	
REQ26	ADC2	REQ90	TIM18_CH1	REQ154	
REQ27	ADC3	REQ91	TIM18_TRIG	REQ155	
REQ28	UART3_TX	REQ92	TIM18_COM	REQ156	
REQ29	UART3_RX	REQ93	TIM19_UP	REQ157	
REQ30	LPUART_TX	REQ94	TIM19_CH1	REQ158	

DMA1 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ31	LPUART_RX	REQ95	TIM19_TRIG	REQ159	
REQ32	TIM15_CH1	REQ96	TIM19_COM	REQ160	
REQ33	TIM15_CH2	REQ97	TIM21_UP	REQ161	
REQ34	TIM15_UP	REQ98	TIM22_UP	REQ162	
REQ35	TIM15_TRIG	REQ99	TIM23_UP	REQ163	
REQ36	TIM15_COM	REQ100	TIM23_CH1	REQ164	
REQ37	I2S1_TX	REQ101	TIM23_CH2	REQ165	
REQ38	I2S1_RX	REQ102	TIM23_CH3	REQ166	
REQ39	DAC1_CH1	REQ103	TIM23_CH4	REQ167	
REQ40	DAC1_CH2	REQ104	TIM23_TRIG	REQ168	
REQ41	I2S2_TX	REQ105	TIM23_COM	REQ169	
REQ42	I2S2_RX	REQ106	TIM24_UP	REQ170	
REQ43	I2S3_TX	REQ107	TIM24_CH1	REQ171	
REQ44	I2S3_RX	REQ108	TIM24_CH2	REQ172	
REQ45	UART4_TX	REQ109	TIM24_CH3	REQ173	
REQ46	UART4_RX	REQ110	TIM24_CH4	REQ174	
REQ47	SPI3_TX	REQ111	TIM24_TRIG	REQ175	
REQ48	SPI3_RX	REQ112	TIM24_COM	REQ176	
REQ49	TIM4_CH1	REQ113	SPI4_RX	REQ177	
REQ50	TIM4_CH2	REQ114	SPI4_TX	REQ178	
REQ51	TIM4_CH3	REQ115	SPI5_RX	REQ179	
REQ52	TIM4_CH4	REQ116	SPI5_TX	REQ180	
REQ53	TIM4_UP	REQ117	DAC2_CH1	REQ181	
REQ54	TIM4_TRIG	REQ118	DAC2_CH2	REQ182	
REQ55	NDL	REQ119	OSPI1_RX	REQ183	
REQ56	STM1_PWM	REQ120	OSPI1_TX	REQ184	
REQ57	STM2_PWM	REQ121	OSPI2_RX	REQ185	
REQ58	DCMI	REQ122	OSPI2_TX	REQ186	

DMA1 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ59	SPI7_RX	REQ123	I2C3_TX	REQ187	
REQ60	SPI7_TX	REQ124	I2C3_RX	REQ188	
REQ61	SPI8_RX	REQ125	I2C4_TX	REQ189	
REQ62	SPI8_TX	REQ126	I2C4_RX	REQ190	
REQ63	STM3_PWM	REQ127	UART5_TX	REQ191	

DMA2 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ0	ADC1	REQ64	STM4_PWM	REQ128	UART5_RX
REQ1	SPI1_TX	REQ65	STM5_PWM	REQ129	UART6TX
REQ2	SPI1_RX	REQ66	STM6_PWM	REQ130	UART6_RX
REQ3	SPI2_TX	REQ67	LPT	REQ131	UART7_TX
REQ4	SPI2_RX	REQ68	TIM5_UP	REQ132	UART7_RX
REQ5	UART1_TX	REQ69	TIM5_CH1	REQ133	UART8_TX
REQ6	UART1_RX	REQ70	TIM5_CH2	REQ134	UART8_RX
REQ7	UART2_TX	REQ71	TIM5_CH3	REQ135	UART9_TX
REQ8	UART2_RX	REQ72	TIM5_CH4	REQ136	UART9_RX
REQ9	I2C1_TX	REQ73	TIM5_TRIG	REQ137	UART10_TX
REQ10	I2C1_RX	REQ74	TIM5_COM	REQ138	UART10_RX
REQ11	I2C2_TX	REQ75	TIM25_UP	REQ139	FDCAN1_TX
REQ12	I2C2_RX	REQ76	TIM25_CH1	REQ140	FDCAN1_RX
REQ13	TIM8_CH1	REQ77	TIM25_CH2	REQ141	FDCAN2_TX
REQ14	TIM8_CH2	REQ78	TIM25_TRIG	REQ142	FDCAN2_RX
REQ15	TIM8_CH3	REQ79	TIM25_COM	REQ143	FDCAN3_TX
REQ16	TIM8_CH4	REQ80	TIM20_UP	REQ144	FDCAN3_RX
REQ17	TIM8_UP	REQ81	TIM20_CH1	REQ145	SPI6_RX
REQ18	TIM8_TRIG	REQ82	TIM20_CH2	REQ146	SPI6_TX
REQ19	TIM8_COM	REQ83	TIM20_CH3	REQ147	

DMA2 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ20	TIM3_CH1	REQ84	TIM20_CH4	REQ148	
REQ21	TIM3_CH2	REQ85	TIM20_CH5	REQ149	
REQ22	TIM3_CH3	REQ86	TIM20_CH6	REQ150	
REQ23	TIM3_CH4	REQ87	TIM20_TRIG	REQ151	
REQ24	TIM3_UP	REQ88	TIM20_COM	REQ152	
REQ25	TIM3_TRIG	REQ89	TIM18_UP	REQ153	
REQ26	ADC2	REQ90	TIM18_CH1	REQ154	
REQ27	ADC3	REQ91	TIM18_TRIG	REQ155	
REQ28	UART3_TX	REQ92	TIM18_COM	REQ156	
REQ29	UART3_RX	REQ93	TIM19_UP	REQ157	
REQ30	LPUART_TX	REQ94	TIM19_CH1	REQ158	
REQ31	LPUART_RX	REQ95	TIM19_TRIG	REQ159	
REQ32	TIM16_CH1	REQ96	TIM19_COM	REQ160	
REQ33	TIM16_UP	REQ97	TIM21_UP	REQ161	
REQ34	TIM17_CH1	REQ98	TIM22_UP	REQ162	
REQ35	TIM17_UP	REQ99	TIM23_UP	REQ163	
REQ36	TIM6_UP	REQ100	TIM23_CH1	REQ164	
REQ37	I2S1_TX	REQ101	TIM23_CH2	REQ165	
REQ38	I2S1_RX	REQ102	TIM23_CH3	REQ166	
REQ39	DAC1_CH1	REQ103	TIM23_CH4	REQ167	
REQ40	DAC1_CH2	REQ104	TIM23_TRIG	REQ168	
REQ41	I2S2_TX	REQ105	TIM23_COM	REQ169	
REQ42	I2S2_RX	REQ106	TIM24_UP	REQ170	
REQ43	I2S3_TX	REQ107	TIM24_CH1	REQ171	
REQ44	I2S3_RX	REQ108	TIM24_CH2	REQ172	
REQ45	UART4_TX	REQ109	TIM24_CH3	REQ173	
REQ46	UART4_RX	REQ110	TIM24_CH4	REQ174	
REQ47	SPI3_TX	REQ111	TIM24_TRIG	REQ175	

DMA2 请求					
请求号	请求源	请求号	请求源	请求号	请求源
REQ48	SPI3_RX	REQ112	TIM24_COM	REQ176	
REQ49	TIM7_UP	REQ113	SPI4_RX	REQ177	
REQ50	保留	REQ114	SPI4_TX	REQ178	
REQ51	保留	REQ115	SPI5_RX	REQ179	
REQ52	保留	REQ116	SPI5_TX	REQ180	
REQ53	保留	REQ117	DAC2_CH1	REQ181	
REQ54	保留	REQ118	DAC2_CH2	REQ182	
REQ55	NDL	REQ119	OSPI1_RX	REQ183	
REQ56	STM1_PWM	REQ120	OSPI1_TX	REQ184	
REQ57	STM2_PWM	REQ121	OSPI2_RX	REQ185	
REQ58	DCMI	REQ122	OSPI2_TX	REQ186	
REQ59	SPI7_RX	REQ123	I2C3_TX	REQ187	
REQ60	SPI7_TX	REQ124	I2C3_RX	REQ188	
REQ61	SPI8_RX	REQ125	I2C4_TX	REQ189	
REQ62	SPI8_TX	REQ126	I2C4_RX	REQ190	
REQ63	STM3_PWM	REQ127	UART5_TX	REQ191	

7.3.4. 仲裁器

DMA 通道优先级是固定的，通道 0 为最高优先级，通道 7 为最低优先级。高优先级和低优先级同时请求时，仲裁器优先响应高优先级通道的请求。如果 DMA 正在进行一个低优先级通道的传输数据，此时一个高优先级通道的请求发生，DMA 将会等待低优先级通道的数据传输完毕，之后再处理高优先级通道的请求。

7.3.5. 传输位宽

传输位宽是指 DMA 发起一次 AHB 总线传输时获取的有效数据宽度。

传输位宽支持字节/半字/字三种位宽，源位宽由 DMA_Cx_CTRL.SWIDTH 决定，目标位宽由 DMA_Cx_CTRL.DWIDTH 决定。

源位宽的设置需要与源外设（或源存储器）的有效数据位宽一致。

目标位宽的设置需要与目标外设（或目标存储器）的有效数据位宽一致。

7.3.6. 突发传输

突发传输分为源突发传输和目标突发传输。

源突发传输是指源外设向 DMA 发送一次硬件请求时，DMA 需要向源外设发起 AHB 总线读传输的次数。源突发传输由 DMA_Cx_CTRL.SBSIZE 决定。在外设到存储器模式（P2M）下，建议将源突发传输设置为源外设 FIFO 深度的一半。

目标突发传输是指目标外设向 DMA 发送一次硬件请求时，DMA 需要向目标外设发起 AHB 总线写传输的次数。目标突发传输由 DMA_Cx_CTRL.DBSIZE 决定。在存储器到外设模式（M2P）下，建议将目标突发传输设置为目标外设 FIFO 深度的一半。

在存储器到存储器模式（M2M）下，源突发传输会影响 DMA 从源存储器读取数据时发起的 AHB 传输类型，进而影响读取数据的效率。目标突发传输会影响 DMA 从源存储器读取数据时发起的 AHB 传输类型，进而影响读取数据的效率。在存储器到存储器模式（M2M）下，建议将源突发传输和目标突发传输都设置为 16 或者更大，这样传输效率最高。

7.3.7. 源、目标

从 DMA 的角度看，源传输和目标传输可以在整个 4 GB 区域（地址在 0x0000 0000 和 0xFFFF FFFF 之间）都可以寻址外设和存储器。但是考虑的 AHB 地址矩阵、外设和存储器的特性，DMA 有部分空间是无法正确寻址的，包括：AHB 地址矩阵未定义的空间、ROM 空间。

源地址必须与源位宽对齐，目标地址必须与目标位宽对齐。

7.3.8. 传输模式

7.3.8.1. 外设到存储器（P2M）

外设到存储器的传输（P2M），该模式的基本步骤:

1）等待源外设请求信号。

- 源外设准备好数据以后，会向 DMA 发送一个高电平的 REQ 请求信号。源外设需要准备的数据量（单位字节）等于 DMAC_Cx_CTRL.SWIDTH 与 DMAC_Cx_CTRL.SBSIZE 的乘积。
- DMA 在 AHB 总线空闲并且此通道具有最高优先级的请求时，向该外设发送 CLR 信号，通知源外设准备数据传输。

2）AHB 主接口从源地址获取源数据，存放在通道的 FIFO 中。

- AHB 主接口自动会发起 AHB 总线传输从源地址读取数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.SWIDTH 决定，AHB 总线传输读取数据的次数等于 DMAC_Cx_CTRL.SBSIZE。
- 每次 AHB 总线传输的有效数据量将会被暂存到通道内置的 FIFO 中，当通道内置 FIFO 满时，将暂停从源地址读取数据。

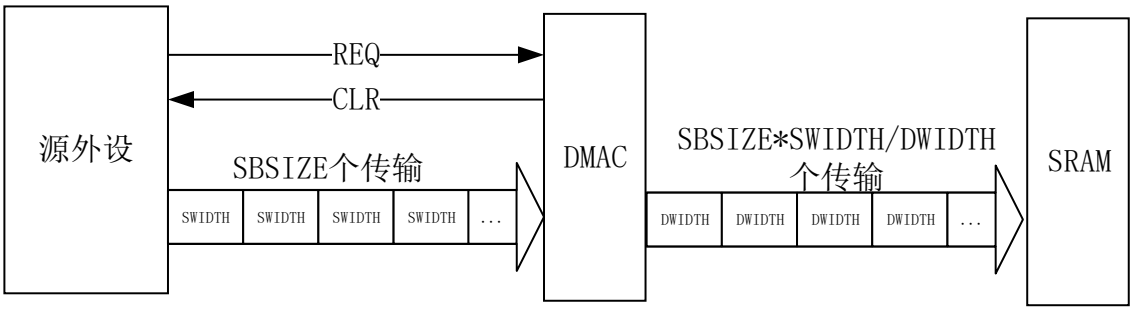
3）AHB 主接口将通道的 FIFO 中数据写入到目标地址。

- 当通道 FIFO 存放的数据量（单位字节）大于或等于 DMAC_Cx_CTRL.DWIDTH 所指定的数据量时，AHB 主接口自动会发起 AHB 总线传输向目标地址写入数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.DWIDTH 决定，AHB 总线传输写入数据的次数等于 SBSIZE*SWIDTH/DWIDTH。

4）更新传输数目 TransferSize。

- 传输数目 TransferSize 分为源传输数目 SrcTransferSize 和目标传输数目 DstTransferSize。
- 源传输数目 SrcTransferSize 代表 AHB 主接口从源地址读取数据总共需要发起的 AHB 总线传输数目。每次完成上述 3 步骤后，源传输数目 SrcTransferSize 需要减少 SBSIZE。

- 目标传输数目 DstTransferSize 代表 AHB 主接口向目标地址写入数据总共需要发起的 AHB 总线传输数目。每次完成上述 3 步骤后，目标传输数目 DstTransferSize 需要减少 SBSIZE*SWIDTH/DWIDTH。
- 当传输数据 TransferSize 减少到 0 时，代表传输完成，硬件自动关闭此通道。



7.3.8.2. 储存器到外设（M2P）

存储器到外设的传输（M2P），该模式的基本步骤:

1）AHB 主接口从源地址获取源数据，存放在通道的 FIFO 中。

- AHB 主接口自动会发起 AHB 总线传输从源地址读取数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.SWIDTH 决定，AHB 总线传输读取数据的次数等于 DMAC_Cx_CTRL.SBSIZE。
- 每次 AHB 总线传输的有效数据量将会被暂存到通道内置的 FIFO 中，当通道内置 FIFO 满时，将暂停从源地址读取数据。

2）等待目标外设请求信号。

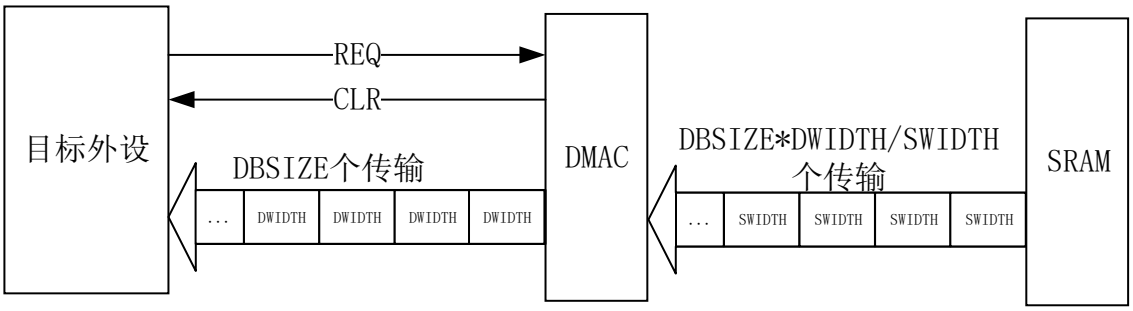
- 目标外设准备好足够的空间接受数据以后，会向 DMA 发送一个高电平的 REQ 请求信号。目标外设需要准备接受的数据量（单位字节）等于 DMAC_Cx_CTRL.DWIDTH 与 DMAC_Cx_CTRL.DBSIZE 的乘积。
- DMA 在 AHB 总线空闲并且此通道具有最高优先级的请求时，向该外设发送 CLR 信号，通知目标外设准备数据传输。

3）AHB 主接口将通道的 FIFO 中数据写入到目标地址。

- 当通道 FIFO 存放的数据量（单位字节）大于或等于 DMAC_Cx_CTRL.DWIDTH 所指定的数据量时，AHB 主接口自动会发起 AHB 总线传输向目标地址写入数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.DWIDTH 决定，AHB 总线传输写入数据的次数等于 SBSIZE*SWIDTH/DWIDTH。

4）更新传输数目 TransferSize。

- 传输数目 TransferSize 分为源传输数目 SrcTransferSize 和目标传输数目 DstTransferSize。
- 源传输数目 SrcTransferSize 代表 AHB 主接口从源地址读取数据总共需要发起的 AHB 总线传输数目。每次完成上述 3 步骤后，源传输数目 SrcTransferSize 需要减少 SBSIZE。
- 目标传输数目 DstTransferSize 代表 AHB 主接口向目标地址写入数据总共需要发起的 AHB 总线传输数目。每次完成上述 3 步骤后，目标传输数目 DstTransferSize 需要减少 SBSIZE*SWIDTH/DWIDTH。
- 当传输数据 TransferSize 减少到 0 时，代表传输完成，硬件自动关闭此通道。



7.3.8.3. 存储器到存储器（M2M）

存储器到存储器的传输（M2M），该模式的基本步骤:

1）AHB 主接口从源地址获取源数据，存放在通道的 FIFO 中。

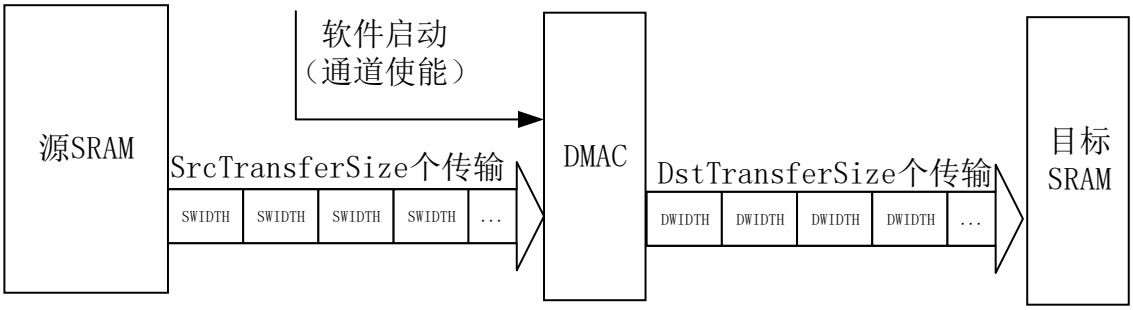
- AHB 主接口自动会发起 AHB 总线传输从源地址读取数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.SWIDTH 决定，AHB 总线传输读取数据的次数等于 DMAC_Cx_CTRL.SBSIZE。
- 每次 AHB 总线传输的有效数据量将会被暂存到通道内置的 FIFO 中，当通道内置 FIFO 满时，将暂停从源地址读取数据。

2）AHB 主接口将通道的 FIFO 中数据写入到目标地址。

- 当通道 FIFO 存放的数据量（单位字节）大于或等于 DMAC_Cx_CTRL.DWIDTH 所指定的数据量时，AHB 主接口自动会发起 AHB 总线传输向目标地址写入数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.DWIDTH 决定，AHB 总线传输写入数据的次数等于 SBSIZE*SWIDTH/DWIDTH。

3）更新传输数目 TransferSize。

- 传输数目 TransferSize 分为源传输数目 SrcTransferSize 和目标传输数目 DstTransferSize。
- 源传输数目 SrcTransferSize 代表 AHB 主接口从源地址读取数据总共需要发起的 AHB 总线传输数目。每次完成上述 2 步骤后，源传输数目 SrcTransferSize 需要减少 SBSIZE。
- 目标传输数目 DstTransferSize 代表 AHB 主接口向目标地址写入数据总共需要发起的 AHB 总线传输数目。每次完成上述 2 步骤后，目标传输数目 DstTransferSize 需要减少 SBSIZE*SWIDTH/DWIDTH。
- 当传输数据 TransferSize 减少到 0 时，代表传输完成，硬件自动关闭此通道。



7.3.9. 指针递增

根 DMA_Cx_CTRL.SI 和 DMA_Cx_CTRL.DI 的状态，通道的源地址和目标地址在每次传输后可以自动增加或者保持常量。

对于单个寄存器访问，可以禁止递增模式。

如果使能了源递增模式，则每次传输后，源地址变为上一次传输的源地址递增 1（源位宽是字节）、2（源位宽

是半字)、4（源位宽是字)。

如果使能了目标递增模式，则每次传输后，目标地址变为上一次传输的目标地址递增 1（目标位宽是字节)、2（目标位宽是半字)、4（目标位宽是字)。

7.3.10. 链表模式

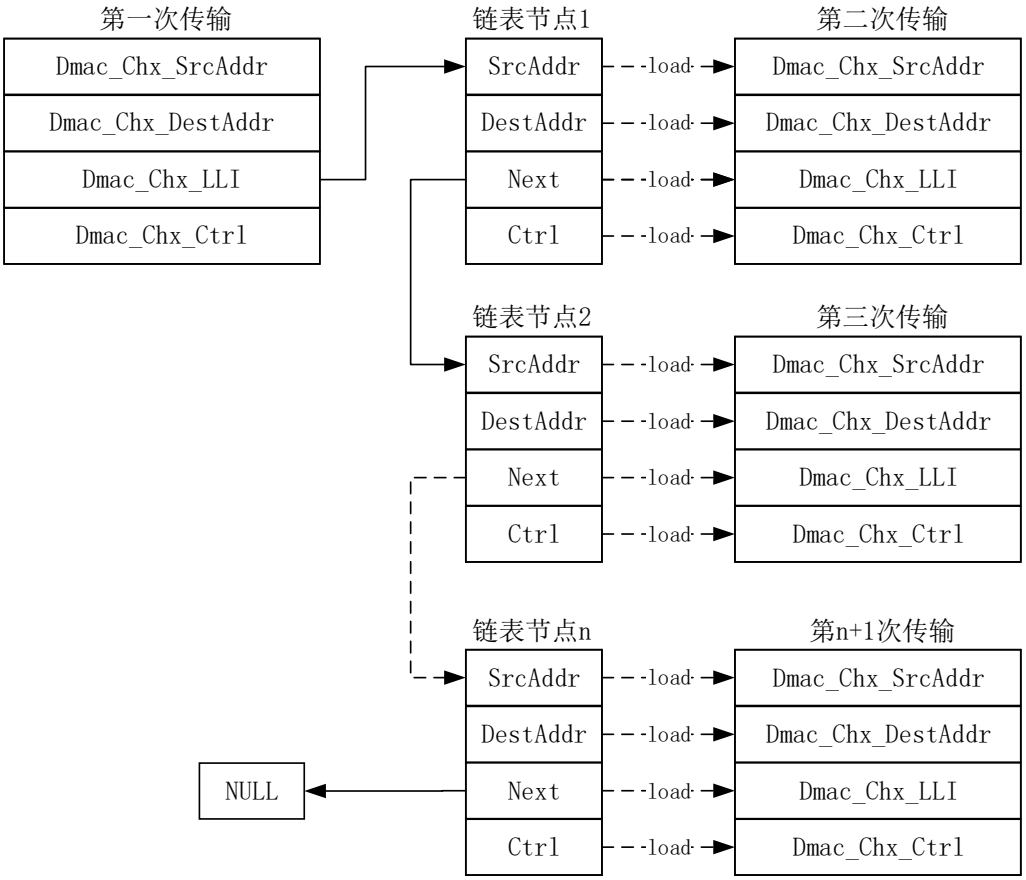
一个 DMA 链表是由一个或多个链表节点组成，存放在存储器中。该链表的每个链表节点都是一个链表结构体。一个链表节点（链表结构体）由 4 个 32 比特的数据项组成，这 4 个数据项按照一定的顺序来排列。

在传输完成后，如果通道 DMA_Cx_LLI.LLI 的值不为 0，则 DMA 将会从(DMA_Cx_LLI.LLI<<2)地址处取 4 个字。第一个字（SrcAddr）加载到通道的 DMA_Cx_SRC_ADDR 寄存器；第二个字（DestAddr）加载到通道的 DMA_Cx_DEST_ADDR 寄存器；第三个字（Next）加载到通道的 DMA_Cx_LLI 寄存器；第四个字（Ctrl）加载到通道的 DMA_Cx_CTRL 寄存器。(注意：通道的 DMA_Cx_CONFIG 寄存器保持不变)。

在传输完成后，如果通道 DMA_Cx_LLI.LLI 的值等于 0，则将通道关闭，硬件将 DMA_Cx_CONFIG.EN 清零。

顺序	链表结构体数据项	加载的寄存器
1	SrcAddr	DMA_Cx_SRC_ADDR
2	DestAddr	DMA_Cx_DEST_ADDR
3	Next	DMA_Cx_LLI
4	Ctrl	DMA_Cx_CTRL

如下图所示，每个链表节点的 Next 指向下一个链表节点的地址，最后一个链表节点的 Next 指向 NULL。如果最后一个链表节点的 Next 指向第一个链表节点，则构成一个单向循环链表。当只有一个链表结构体，且 Next 指向自身，则构成一个自循环链表。普通 DMA 传输（非链表模式）因 DMA_Cx_LLI.LLI 指向 NULL，可以理解为一个单节点的链表传输。



7.3.11. 循环模式

用链表模式可以很容易实现循环模式功能。

1）初始化一个链表结构体，该链表结构体的 Next 指向自身。

2）初次配置该通道时，将 DMA_Cx_LLI.LLI 也指向该链表结构体。

7.3.12. 双缓冲模式

用链表模式可以很容易实现双缓冲模式功能。

初始化两个链表结构体，第一个链表结构体的 SrcAddr(DestAddr)指向缓冲区 1，第二个链表结构体的 SrcAddr(DestAddr)指向缓冲区 2。

第一个链表结构体的 Next 指向第二个链表结构体。

初次配置该通道时，将 DMA_Cx_LLI.LLI 也指向第一个链表结构体。

7.3.13. 可编程端模式、数据宽度、封装/解封、字节序

DMA 支持可编程的大小端模式，数据宽度。DMA 模块可配置为双 AHB 主机和单 AHB 主机模式，当 DMA 模块支持双 AHB 主机模式时，可以配置源和目标为不同的大小端模式。本项目不支持双 AHB 主机模式，所以源和目标的大小端模式一定相同。

7.3.13.1. 源小端模式、目标小端模式

源端模式	目标端模式	源位宽	目标位宽	源地址/源数据	目标地址/目标数据

Little	Little	8	8	0/0x21 1/0x43 2/0x65 3/0x87	0/0x21 1/0x43 2/0x65 3/0x87
Little	Little	8	16	0/0x21 1/0x43 2/0x65 3/0x87	0/0x4321 2/0x8765
Little	Little	8	32	0/0x21 1/0x43 2/0x65 3/0x87	0/0x87654321
Little	Little	16	8	0/0x4321 2/0x8765	0/0x21 1/0x43 2/0x65 3/0x87
Little	Little	16	16	0/0x4321 2/0x8765	0/0x4321 2/0x8765
Little	Little	16	32	0/0x4321 2/0x8765	0/0x87654321
Little	Little	32	8	0/0x87654321	0/0x21 1/0x43 2/0x65 3/0x87
Little	Little	32	16	0/0x87654321	0/0x4321 2/0x8765
Little	Little	32	32	0/0x87654321	0/0x87654321

7.3.13.2. 源大端模式、目标大端模式

源端模式	目标端模式	源位宽	目标位宽	源地址/源数据	目标地址/目标数据
Big	Big	8	8	3/0x12 2/0x34 1/0x56 0/0x78	3/0x12 2/0x34 1/0x56 0/0x78
Big	Big	8	16	3/0x12 2/0x34 1/0x56 0/0x78	0/0x1234 2/0x5678

Big	Big	8	32	3/0x12 2/0x34 1/0x56 0/0x78	0/0x12345678
Big	Big	16	8	2/0x1234 0/0x5678	3/0x12 2/0x34 1/0x56 0/0x78
Big	Big	16	16	2/0x1234 0/0x5678	0/0x1234 2/0x5678
Big	Big	16	32	0/0x1234 2/0x5678	0/0x12345678
Big	Big	32	8	0/0x12345678	3/0x12 2/0x34 1/0x56 0/0x78
Big	Big	32	16	0/0x12345678	0/0x1234 2/0x5678
Big	Big	32	32	0/0x12345678	0/0x12345678

7.3.14. 关闭 DMA 通道

通道传输完成或者通道传输错误发生时，由硬件自动关闭通道。

DMA 传输过程中，禁止通道（DMA_Cx_CONFIG.EN 置位），此方式关闭通道，会丢失通道 FIFO 中的数据。

如果想要不丢失 FIFO 中的数据，那么可以用如下方式关闭通道：

1）暂停 DMA 传输（DMA_Cx_CONFIG.Halt 置位）。Halt 位可以阻止 DMA 响应后续通道的请求。

2）轮询 DMA_Cx_CONFIG.Active 状态位，等待其为 0。Active 位代表通道内 FIFO 的数据都已经传输完毕。

3）禁止通道（DMA_Cx_CONFIG.EN 清零）。

7.3.15. 暂停 DMA 通道

将 DMA_Cx_CONFIG.HALT 寄存器置位可以暂停通道。当 DMA_Cx_CONFIG.HALT 置位时，DMA 仲裁器将忽略该通道的后续传输请求；如果 DMA_Cx_CONFIG.HALT 被清零，那么 DMA 仲裁器将继续处理该通道的传输请求。

7.3.16. 中断

7.3.16.1. 半传输完成

当传输数目达到总传输数目一半时，代表半传输完成。

如果 DMA_Cx_CTRL.ITC 置位，那么硬件会在半传输完成时自动将 DMA_RAW_INT_TC_STATUS.HFTC 置位。

如果 DMA_Cx_CTRL.ITC 置位并且 DMA_Cx_CONFIG.IHFTC 置位，那么硬件会在半传输完成时自动将 DMA_INT_TC_STATUS.HFTC 置位。

7.3.16.2. 传输完成

当传输数目等于总传输数目时，代表传输完成。

如果 DMA_Cx_CTRL.ITC 置位，那么硬件会在传输完成时自动将 DMA_RAW_INT_TC_STATUS.TC 置位。

如果 DMA_Cx_CTRL.ITC 置位并且 DMA_Cx_CONFIG.ITC 置位，那么硬件会在传输完成时自动将 DMA_INT_TC_STATUS.TC 置位。

如果 DMA_Cx_LLI.LLI 等于 0，传输完成时，硬件自动关闭通道，并将 DMA_Cx_CONFIG.EN 清零。

如果 DMA_Cx_LLI.LLI 不等于 0，传输完成时，硬件不会关闭通道，DMA_Cx_CONFIG.EN 将保持置位状态。

7.3.16.3. 传输错误

当 DMA 传输过程中，AHB 总线回复错误的响应时，会出现传输错误。一般访问一个不存在的地址，或者访问一个受保护的地址时，会出现传输错误。

出现传输错误时硬件自动将 DMA_Cx_CONFIG.EN 清零，同时将 DMA_RAW_INT_ERR_STATUS.ERR 置位。如果 DMA_Cx_CONFIG.IE 置位，那么硬件会自动将 DMA_INT_ERR_STATUS.ERR 置位。

7.3.16.4. 链表模式下的中断模式

链表模式下有两种中断模式：

1）每个链表节点传输都产生中断。

a）使能 DMA_Cx_CONFIG.ITC。

b）使能每个链表节点的 DMA_Cx_CTRL.ITC（或 DMA_Cx_CTRL.IHFTC）。

2）所有链表节点传输都完成后，产生一次中断。

a）使能 DMA_Cx_CONFIG.ITC。

b）使能最后一个链表节点的 DMA_Cx_CTRL.ITC（或 DMA_Cx_CTRL.IHFTC），禁止其他链表节点的 DMA_Cx_CTRL.ITC（或 DMA_Cx_CTRL.IHFTC）。

任何时候，只要 DMA_Cx_CONFIG.IE 置位，当传输错误发生时，都会立刻产生传输错误中断。

7.4. 配置流程

7.4.1.1. DMA 初始化

1）使能 DMA（DMA_CONFIG.EN）。

2）禁止通道（DMA_Cx_CONFIG.EN）。

3）配置流控制和传输类型（DMA_Cx_CONFIG. FLOWCTRL）。

4）配置源外设 ID（DMA_Cx_CONFIG.SRCID）。

5）配置目标外设 ID（DMA_Cx_CONFIG.DESTID）。

6）配置源地址位宽（DMA_Cx_CTRL.SWIDTH）。

7）配置目标地址位宽（DMA_Cx_CTRL.SWIDTH）。

8）配置源地址递增使能位（DMA_Cx_CTRL.SI）。

9）配置目标地址递增使能位（DMA_Cx_CTRL.DI）。

10）配置源突发传输数量（DMA_Cx_CTRL.SBSIZE）。

11）配置目标突发传输数量（DMA_Cx_CTRL.DBSIZE）。

12）配置原始中断位（DMA_Cx_CTRL.ITC），可实现 DMA 中断或标志位查询操作。

13）配置中断使能位（DMA_Cx_CONFIG.IHFTC、DMA_Cx_CONFIG.ITC、DMA_Cx_CONFIG.IE），可实现 DMA 各种中断操作。

7.4.1.2. DMA 传输

1）配置源地址（DMA_Cx_SRC_ADDR）。

2）配置目标地址（DMA_Cx_DEST_ADDR）。

3）配置通道链接表地址（DMA_Cx_LLI.LLI）。

4）配置传输长度（DMA_Cx_CTRL.TRANSFERSIZE）。

5）使能通道（DMA_Cx_CONFIG.EN）。

7.5. DMA 寄存器描述

7.5.1. 寄存器列表

DMA1 寄存器基地址：0x40020000

DMA2 寄存器基地址：0x40020400

偏移	名称	复位值	描述
0x00	DMA_INT_STATUS	0x00000000	中断状态寄存器
0x04	DMA_INT_TC_STATUS	0x00000000	传输完成中断寄存器
0x08	DMA_INT_TC_CLR	0x00000000	传输完成中断清除寄存器
0x0C	DMA_INT_ERR_STATUS	0x00000000	传输错误中断寄存器
0x10	DMA_INT_ERR_CLR	0x00000000	传输错误中断清除寄存器

0x14	DMA_RAW_INT_TC_STATUS	0x00000000	传输完成原始中断寄存器
0x18	DMA_RAW_INT_ERR_STATUS	0x00000000	传输错误原始中断寄存器
0x1C	DMA_EN_CH_STATUS	0x00000000	通道使能状态寄存器
0x30	DMA_CONFIG	0x00000000	DMA 配置寄存器
0x100+(x*0x20)	DMA_Cx_SRC_ADDR	0x00000000	源通道 x 地址寄存器（x=0~7）
0x104+(x*0x20)	DMA_Cx_DEST_ADDR	0x00000000	目标通道 x 地址寄存器（x=0~7）
0x108+(x*0x20)	DMA_Cx_LLI	0x00000000	通道 x 链接表寄存器（x=0~7）
0x10C+(x*0x20)	DMA_Cx_CTRL	0x00000000	通道 x 控制寄存器（x=0~7）
0x110+(x*0x20)	DMA_Cx_CONFIG	0x00000000	通道 x 配置寄存器（x=0~7）

7.5.2. 中断状态寄存器(DMA_INT_STATUS: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	INT	RO	0	中断状态寄存器。 1：表示对应的通道产生中断，当 DMA 控制器传输完成（DMA_INT_TC_STATUS）或传输错误(DMA_INT_ERR_STATUS)时触发该中断。 0：没有产生中断

7.5.3. 传输完成中断寄存器(DMA_INT_TC_STATUS: 04h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	HFTC	RO	0	半传输完成中断状态 1：对应的通道半传输完成中断 0：没有半传输完成中断 如果禁止了该中断，可以查询 DMA_RAW_INT_TC_STATUS 寄存器。
7:0	TC	RO	0	传输完成中断状态 1：对应的通道传输完成中断 0：没有传输完成中断 如果禁止了该中断，可以查询 DMA_RAW_INT_TC_STATUS 寄存器。

7.5.4. 传输完成中断清除寄存器(DMA_INT_TC_CLR: 08h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留

15:8	HFTC	WO	0	半传输完成中断状态清楚 1：写 1 将清除相应通道的半传输完成状态。 0：无动作
7:0	TC	WO	0	传输完成中断状态清除 1：写 1 将清除相应通道的传输完成状态。 0：无动作

7.5.5. 传输错误中断寄存器(DMA_INT_ERR_STATUS: 0Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	ERR	RO	0	传输错误中断状态 1：对应的通道传输错误中断。 0：没有传输错误中断。 如果禁止了该中断，可以查询 DMA_RAW_INT_ERR_STATUS 寄存器。

7.5.6. 传输错误中断清除寄存器(DMA_INT_ERR_CLR: 10h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	ERR	WO	0	传输错误中断状态清除。 1：写 1 将清除相应通道的传输错误状态 0：无动作

7.5.7. 传输完成原始中断寄存器(DMA_RAW_INT_TC_STATUS: 14h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	HFTC	RO	0	半传输完成原始中断状态。 1：对应的通道半传输完成 0：没有半传输完成
7:0	TC	RO	0	传输完成原始中断状态。 1：对应的通道传输完成 0：没有传输完成

7.5.8. 传输错误原始中断寄存器(DMA_RAW_INT_ERR_STATUS: 18h)

位域	名称	属性	复位值	描述

31:8	RSV	-	-	保留
7:0	ERR	RO	0	传输错误中断状态。 1：对应的通道传输错误 0：没有错误

7.5.9. 通道使能状态寄存器(DMA_EN_CH_STATUS: 1Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EN	RO	0	1：对应的通道已使能 0：对应的通道未使能

7.5.10. DMA 配置寄存器(DMA_CONFIG: 30h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	M2ENDIAN	RW	0	MASTER2 字节存储次序配置 1：大端模式 0：小端模式
1	M1ENDIAN	RW	0	MASTER1 字节存储次序配置 1：大端模式 0：小端模式
0	EN	RW	0	DMA 使能 1：使能 DMA 0：禁止 DMA

7.5.11. 源通道 x 地址寄存器(DMA_Cx_SRC_ADDR: 100h+(x*20h), x=0~7)

位域	名称	属性	复位值	描述
31:0	SRCADDR	RW	0	源通道地址寄存器

注意：当通道使能打开时，不允许改变该寄存器的值。

7.5.12. 目标通道 x 地址寄存器(DMA_Cx_DEST_ADDR: 104h+(x*20h), x=0~7)

位域	名称	属性	复位值	描述
31:0	DESTADDR	RW	0	目标通道地址寄存器

注意：当通道使能打开时，不允许改变该寄存器的值。

7.5.13. 通道 x 链接表寄存器(DMA_Cx_LLI: 108h+(x*20h), x=0~7)

位域	名称	属性	复位值	描述
31:2	LLI	RW	0	通道链接表，表示下一个 32 位的链接表地址 Addr[31:2]，其中 Addr[1:0]为 0.
1	RSV	-	-	保留
0	LM	RW	0	下一个 LLI 的 MASTER 号 1：MASTER2 0：MASTER1

7.5.14. 通道 x 控制寄存器(DMA_Cx_CTRL: 10Ch+(x*20h), x=0~7)

位域	名称	属性	复位值	描述
31	ITC	RW	0	当前传输是否产生原始中断。 1：使能。 0：禁止。
30	RSV	-	-	保留
29:28	DIORDD	RW	0	00:不增不减 01:递增 10:递减 11:不增不减 目标地址递增递减使能位，每个 AHB 总线传输后目标地址将加上/加上/不变一个递增量
27:26	SIORSD	RW	0	00:不增不减 01:递增 10:递减 11:不增不减 源地址递增递减使能位，每个 AHB 总线传输后目标地址将加上/加上/不变一个递增量
27:25	DWIDTH	RW	0	目标传输位宽。源和目标位宽可以不同，硬件会自动打包、解包数据。 DWIDTH 对应的位宽如下： 000：字节（8 位） 001：半字（16 位） 010：字（32 位） 其它：保留

24:22	SWIDTH	RW	0	源传输位宽。源和目标位宽可以不同，硬件会自动打包、解包数据。 SWIDTH 对应的位宽如下： 000：字节（8 位） 001：半字（16 位） 010：字（32 位） 其它：保留
21:19	DBSIZE	RW	0	目标突发传输大小。 000：1 001：4 010：8 011：16 100：32 101：64 110：128 111：256
18:16	SBSIZE	RW	0	源突发传输大小。 000：1 001：4 010：8 011：16 100：32 101：64 110：128 111：256
15:0	TRANSFERSIZE	RW	0	总传输数目 写入此寄存器时以源外设（或源存储器）为准，代表 DMA 总共需要从源外设（或源存储器）发起的 AHB 总线读传输的次数。 读出此寄存器时以目标外设（或目标存储器为准），代表 DMA 还需向目标外设（或目标存储器）发起的 AHB 总线写传输的次数。 注意，读此寄存器并不能准确地获取当前传输的进度，因为读此寄存器的时候 DMA 可能又传输了很多次。

注意：当通道使能打开时，不允许改变该寄存器的值。在通道使能打开前将各个通道寄存器配置好，打开使能后，配置好的寄存器即生效。

7.5.15. 通道 x 配置寄存器(DMA_Cx_CONFIG: 110h+(x*20h), x=0~7)

位域	名称	属性	复位值	描述
31:24	SRCID	RW	0	源外设 ID。当源是存储器时，该值无效。
23:16	DESTID	RW	0	目标外设 ID。当目标是存储器时，该值无效。
15:12	RSV	-	-	保留
11	LOCK	RW	0	锁定 AHB 总线

10	DMST	RW	0	目标 AHB MASTER 选择 1：MASTER 2 0：MASTER 1
9	SMST	RW	0	源 AHB MASTER 选择 1：MASTER 2 0：MASTER 1
8	HALT	RW	0	1：中止当前 DMA 传输，中止后通道 FIFO 中的内容会被清除。 0：正常
7	ACTIVE	RO	0	1：通道 FIFO 中有数据 0：通道 FIFO 中没有数据
6	IHFTC	RW	0	半传输完成中断使能 1：使能该通道的半传输完成中断 0：屏蔽该通道的半传输完成中断
5	ITC	RW	0	传输完成中断使能 1：使能该通道的传输完成中断 0：屏蔽该通道的传输完成中断
4	IE	RW	0	传输错误中断使能 1：使能该通道的传输错误中断 0：屏蔽该通道的传输错误中断
3:1	FLOWCTRL	RW	0	流控制器和传输类型。 FLOWCTRL 传输方向 控制器 000 存储器到存储器 DMA 001 存储器到外设 DMA 010 外设到存储器 DMA 011 外设到外设 DMA
0	EN	RW	0	通道使能 1：通道使能，传输完成后硬件自动清零。软件写 0 无效。 0：通道关闭 可以读取 DMA_EN_CH_STATUS 寄存器知道通道使能的情况。

8. 一次性可编程存储器（EFUSE）

8.1. 概述

EFUSE 为可一次编程的非易失性存储器。

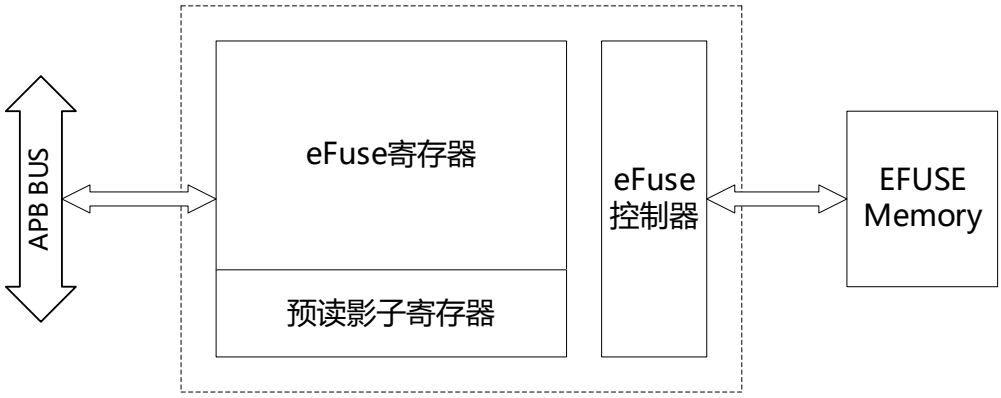
本芯片内置两个 EFUSE 模块，每个模块 256 字节，按字节进行读和编程操作。

8.2. 主要特性

- 两个 EFUSE 模块，每个模块 256 字节
- 编程时 AVDD 保持高电平，但累计时间不能超过 1 秒；在读模式或空闲模式下，AVDD 保持低电平或悬空
- 编程上电建立时间 10us
- 编程脉冲宽度 4us
- 编程时间计数器的时钟源为 RCH，不支持 RCH 的 16 分频模式

8.3. 结构框图

图 8-1 EFUSE 控制器结构框图



8.4. 功能说明

EFUSE 控制器支持硬件预读和 2 种软件操作模式：

- 编程模式
- 读取模式

8.4.1. 硬件预读

上电或 EFUSE 控制器被复位后，EFUSE 控制器会硬件自动读取 EFUSE MEMORY 中的前 16 个字节数据，并将数据存放在相应的影子寄存器（EFUSE_DSRn）中，同时置起预读完成标志位（寄存器 EFUSE_SR 的 bit31）。

当判断预读完成后，才可以进行编程或读取等软件操作。

8.4.2. 编程模式

EFUSE 存储单元初始值为 0，经过编程后变为 1。

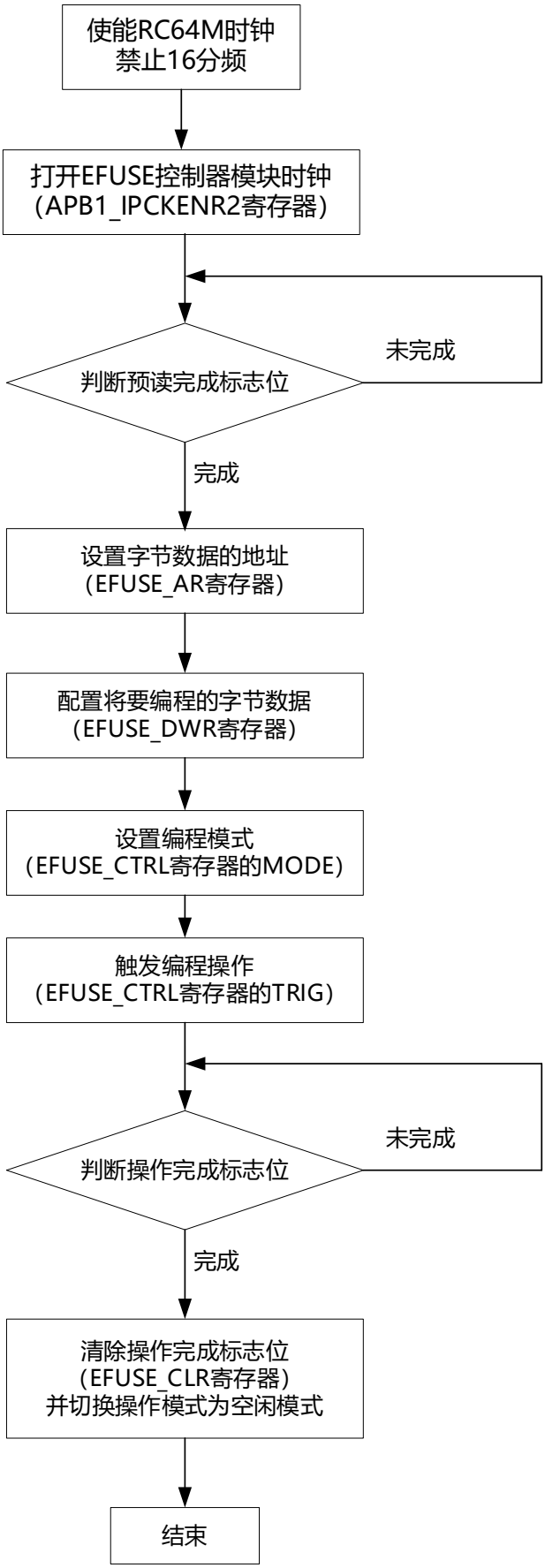
在编程模式（配置 EFUSE_CTRL 寄存器 bit4 为 1）下，可一次性编程 8 位，编程数据由（EFUSE_DWR）设定。

8.4.3. 读取模式

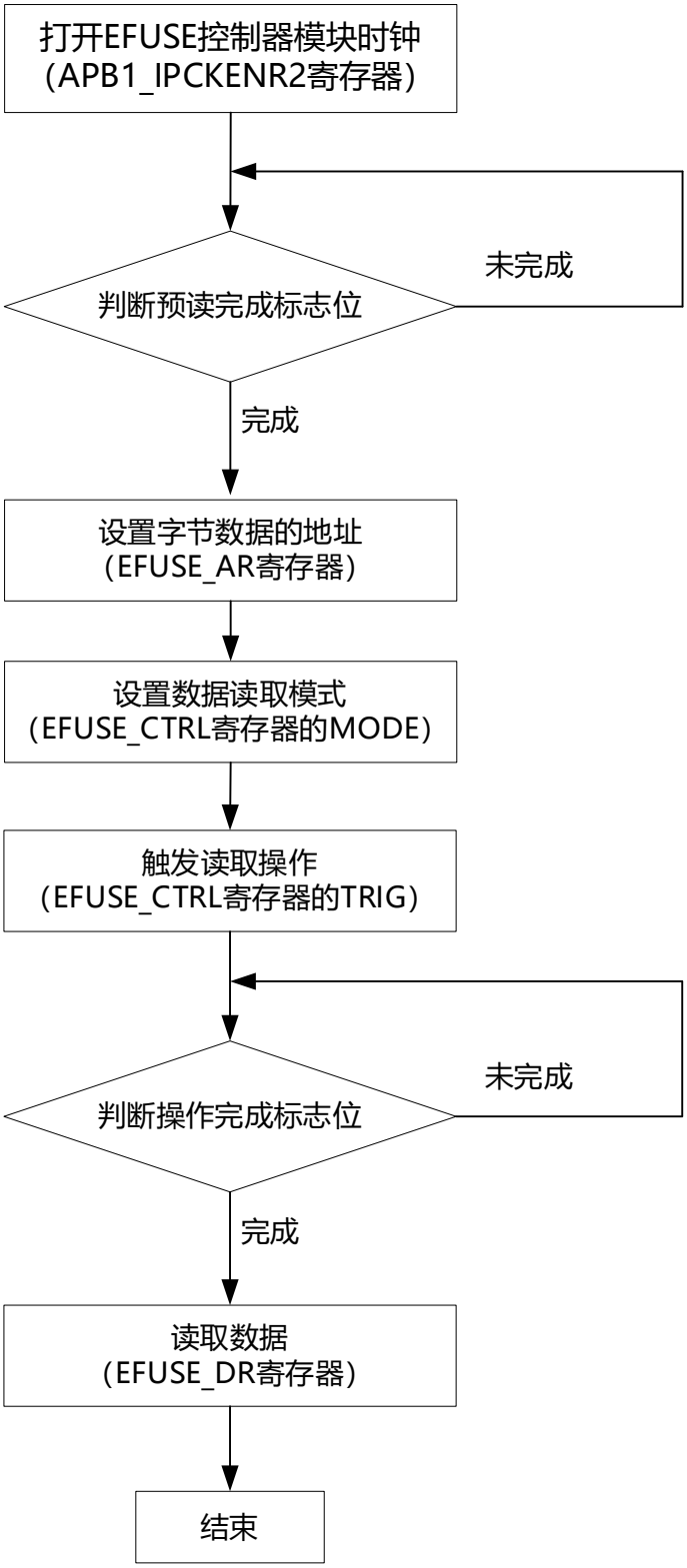
在读取模式（配置 EFUSE_CTRL 寄存器 bit4 为 0）下可一次读取 8 位数据，并暂存在数据读取寄存器中（EFUSE_DR)；

8.5. 配置流程

8.5.1. 编程操作



8.5.2. 读取操作



8.6. EFUSE 寄存器描述

8.6.1. 寄存器列表

EFUSE1 寄存器基地址：0x4000E000

EFUSE2 寄存器基地址：0x4000E800

偏移	名称	复位值	描述
0x00	EFUSE_WP	0x00000000	编程保护寄存器
0x04	EFUSE_CTRL	0x00000000	控制寄存器
0x08	EFUSE_AR	0x00000000	地址寄存器
0x0C	EFUSE_DWR	0x00000000	字节数据写入寄存器
0x10	EFUSE_SR	0x80000000	状态寄存器
0x14	EFUSE_CLR	0x00000000	状态清零寄存器
0x18	EFUSE_DR	0x00000000	数据读取寄存器
0x1C	EFUSE_DSDP	0x00000000	预读数据影子寄存器读保护
0x20	EFUSE_BYTEWP	0x00000000	字节编程保护寄存器
0x24	EFUSE_PGCFG	0x5a0a0a1f	编程参数配置寄存器
0x400+(x*4)	EFUSE_DSRx	0x00000000	预读数据影子寄存器 x（x=0~15）

8.6.2. 写保护寄存器(EFUSE_WP: 00h)

位域	名称	属性	复位值	描述
31:0	WP	RW	0x0	EFUSE 编程保护： 0xEF59A6CB：EFUSE 禁止被写入 其他：EFUSE 允许被写入

注 1：芯片上电或系统复位后只能配置一次

注 2：EFUSE 模块软复位（RCC_APB1RSTR2 寄存器的 bit7 和 bit6）不复位该寄存器

8.6.3. 控制寄存器(EFUSE_CTRL: 04h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	MODE	RW	0x0	EFUSE 操作模式选择位 0：读取模式 1：编程模式
3:1	RSV	-	-	保留
0	TRIG	RW	0x0	EFUSE 编程/读操作触发位 0：无效 1：触发使能 写 1 触发编程或读操作，硬件自动清零

8.6.4. 地址寄存器(EFUSE_AR: 0x08)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	ADDR	RW	0x0	指定 EFUSE 操作地址

8.6.5. 字节数据写入寄存器(EFUSE_DWR: 0x0C)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	WDATA	RW	0x0	在字节编程模式下，指定将要编程的字节数据

8.6.6. 状态寄存器(EFUSE_SR: 0x10)

位域	名称	属性	复位值	描述
31	PREREAD_DONE	RO	0x1	EFUSE 硬件预读操作完成标志位 0：预读操作未完成 1：预读操作完成
30:2	RSV	-	-	保留
1	UNPG	RO	0x0	EFUSE 编程操作未执行且终止标志位 0：未触发编程操作 1：触发编程操作，但未执行并终止编程 以下情况，会终止编程操作： 1）编写全 0 数据 2）对已编程的地址再编写相同的数据 3）开启字节编程保护功能后（寄存器 EFUSE_BYTEWP），对已编程非全 0 数据的地址再进行编写 4）开启写保护功能后（寄存器 EFUSE_WP），触发编程操作（寄存器 EFUSE_CTRL 的 bit0 置 1）
0	DONE	RO	0x0	EFUSE 操作完成标志位 0：读取/编程操作未完成或未执行 1：读取/编程操作完成

8.6.7. 状态清零寄存器(EFUSE_CLR: 0x14)

位域	名称	属性	复位值	描述

31:2	RSV	-	-	保留
1	CUNPG	WO	0x0	清零 EFUSE 编程操作未执行且终止标志位 写 1 将 FUSE_SR 寄存器中的 UNPG 标志位清零
0	CDONE	WO	0x0	清零 EFUSE 读取/编程完成标志位 写 1 将 FUSE_SR 寄存器中的 DONE 标志位清零

8.6.8. 数据读取寄存器(EFUSE_DR: 0x18)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DATA	RO	0x0	读取模式下，EFUSE 的输出数据

8.6.9. 预读数据影子寄存器(读保护 EFUSE_DSDP: 0x1C)

位域	名称	属性	复位值	描述
31:0	DP	RW	0x0	禁止读取预读数据影子寄存器的数值： 0xACF56B49：禁止读取 其他：允许读取

注 1：芯片上电或系统复位后只能配置一次

注 2：EFUSE 模块软复位（RCC_APB1RSTR2 寄存器的 bit7 和 bit6）不复位该寄存器

8.6.10. 字节编程保护寄存器(EFUSE_BYTEWP: 0x20)

位域	名称	属性	复位值	描述
31:0	BYTEWP	RW	0x0	EFUSE 字节编程保护： 0xC98E1A6D：对已编程非全 0 数据的地址再编程时将被禁止 其他：EFUSE 允许被编程

注 1：芯片上电或系统复位后只能配置一次

注 2：EFUSE 模块软复位（RCC_APB1RSTR2 寄存器的 bit7 和 bit6）不复位该寄存器

8.6.11. 编程参数配置寄存器(EFUSE_PGCFG: 0x24)

位域	名称	属性	复位值	描述

31:24	AVDD_SP	RW	0x5a	AVDD 上电建立时间配置，时间计算如下： (AVDD_SP+1)*(8/RCH) RCH 为 64MHz，AVDD_SP 单位时间为 0.125us 注：建立时间要求≥10us
23:16	AVDD_HD	RW	0x0a	AVDD 下电保持时间配置，时间计算如下： (AVDD_HD+1)*(8/RCH) RCH 为 64MHz，AVDD_HD 单位时间为 0.125us 注：保持时间要求≥1us
15:8	PGWT	RW	0x0a	编程完成后等待时间配置，时间计算如下： (PGWT+1)*(8/RCH) RCH 为 64MHz 的 16 分频（即 4MHz）时，PGWT 单位时间为 2us 注：等待时间要求≥1us
7:0	PGT	RW	0x1f	编程时间配置，即烧录脉冲宽度，时间计算如下： (PGT+1)*(8/RCH) RCH 为 64MHz 的 16 分频（即 4MHz）时，PGT 单位时间为 2us 注：编程时间范围：3us≤PGT≤5us，典型值为 4us

注 1：芯片上电或系统复位后只能配置一次

注 2：EFUSE 模块软复位（RCC_APB1RSTR2 寄存器的 bit7 和 bit6）不复位该寄存器

8.6.12. 预读数据影子寄存器 x(EFUSE_DSRx: 400h+(x*4), x=0~15)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DATA	RO	0x0	EFUSE 预读取的数据影子寄存值

9. 延迟模块（DLYB)

9.1. 概述

延迟模块（DLYB，Delay Block）对输入时钟进行相位偏移编程，并输出时钟。输出时钟用于其他外设（如SDIO、OSPI 等接口）接收的数据进行计时。

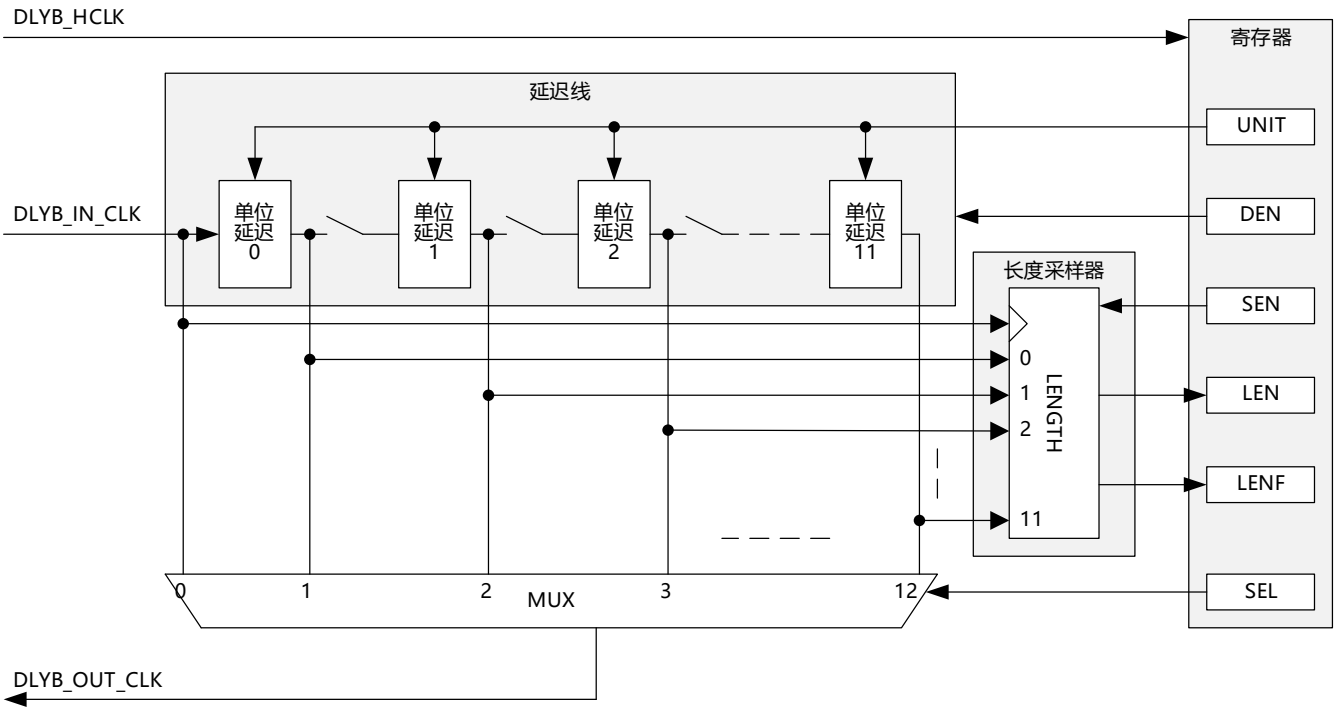
模块的延迟与电压和温度有关，可能需要重新配置应用程序并确定输出时钟与接收数据之间的相位关系。

9.2. 主要特性

- 输入时钟频率范围为 25 MHz 到 208 MHz
- 多达 12 个偏移相位

9.3. 结构框图

图 9-1 DLYB 结构框图



延迟模块包括以下部分：

- 寄存器接口模块
- 以单位延迟计数的延迟线
- 延迟线长度采样
- 输出时钟选择复用器

9.4. 功能描述

通过寄存器 DLYB_CR.DEN 使能延迟模块。通过 DLYB_CR.SEN 使能长度采样器。

总的延迟时间以“单位延迟”为单位递增。

单位延迟的时间可通过寄存器 DLYB_CFGR.UNIT 配置。

输出时钟相位通过寄存器 DLYB_CFGR.SEL 选择。

只有在输出时钟禁止（即 DLYB_CR.SEN=1）的情况下，才能配置 DLYB_CFGR.UNIT 和 DLYB_CFGR. SEL。

延迟线长度可以通过长度采样器来配置。

通过配置延迟线长度超过一个时钟周期，可以让输出时钟周期相位偏移超过一个输入时钟周期。

如果输出时钟延迟小于一个输入时钟周期，可以减小延迟线长度。这允许更小的单位延迟，得到更高的精度。

一旦延迟线长度被配置，可以通过输出时钟选择器选择输出时钟相位偏移。

延迟模块（DLYB）控制逻辑关系如下表所示

模块使能 (DEN)	采样器使能 时钟输出禁止 (SEN)	输出时钟相位选择 (SEL)	延迟线长度 (LEN)	延迟线长度有效标志位 (LENF)	单位延迟数值 (UNIT)	输出时钟
0	0	无效	无效	无效	无效	输出时钟=输入时钟
x	1	可以修改	可访问	可访问	可以修改	禁止
1	0	不可修改	无效	无效	不可修改	输出时钟=输入时钟经过延迟移相后的时钟

9.4.1. 延迟线长度配置步骤

LEN 用于确定与输入时钟周期相关的延迟线长度。

延迟线长度应当配置为覆盖一个完整的输入时钟周期。

尽管延迟线有 12 个延迟单元，下面的过程描述返回 0~10 之间的长度，作为延迟输出值上边界，是为了确保延迟在一个完整的输入时钟周期内被校准。

在整个配置期间，时钟输入必须一直存在。

要将延迟线长度配置为一个输入时钟周期，步骤如下：

- 设置 DEN=1，使能延迟模块
- 设置 SEN=1，使能长度采样器
- 设置 SEL=12，使能所有延迟单元
- 配置 UNIT 值，从 0~63 循环，直到延迟线长度达到了需要的配置：
  - 更新 UNIT 值并等待长度标志 LENF 为 1
  - 读取 LEN

- 如果（LEN[10:0]>0）且（LEN[11]或 LEN[10]=0），则延迟线长度可以覆盖一个输入时钟周期，此时退出循环

- 确定需要多少个单位延迟（N）才能覆盖一个输入时钟周期：
  - N 从 10 到 0 循环，直到 LEN[N]=1，则覆盖输入时钟周期的单位延迟数量=N，此时退出循环
- 通过将 SEN 清零，可禁止长度采样

如果输出时钟延迟小于一个输入时钟周期，则延迟线长度可以小于一个输入时钟周期。这允许一个更小的单位延迟，提供更高的精度。

9.4.2. 输出时钟相位配置步骤

当将延迟线长度配置为一个输入时钟周期时，可以在覆盖一个输入时钟周期的多个单延迟之间选择输出时钟相位，步骤如下：

- 设置 SEN=1，禁止输出时钟
- 配置 SEL 选择期望的输出时钟相位值
- 设置 SEN=0，使能输出时钟

9.5. DLYB 寄存器描述

9.5.1. 寄存器列表

USB1DLYB-REG 寄存器基地址：0x52006800

ETHDLYBREG 寄存器基地址：0x52006C00

SDIO-DLYBS-REG 寄存器基地址：0x520CA000

SDIO-DLYBD-REG 寄存器基地址：0x520CA800

OSPI1-DLYB-REG 寄存器基地址：0x520D2000

OSPI2-DLYB-REG 寄存器基地址：0x520D2400

偏移	名称	复位值	描述
0x00	DLYB_CR	0x00000000	DLYB 控制寄存器
0x04	DLYB_CFGR	0x00000000	DLYB 配置寄存器

9.5.2. DLYB 控制寄存器(DLYB_CR: 00h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	SEN	RW	0	长度采样器使能位 0：禁止长度采样器，禁止访问寄存器 UNIT 和 SEL，使能输出时钟 1：使能长度采样器，允许访问寄存器 UNIT 和 SEL，禁止输出时钟

0	DEN	RW	0	延迟模块使能位 0：禁止 1：使能

9.5.3. DLYB 配置寄存器(DLYB_CFGR: 04h)

位域	名称	属性	复位值	描述
31	LENF	RO	0	长度有效标志位，该标志位表示在 UNIT 位更改之后，LEN 域中包含的延迟线长度值是否有效 0：LEN 中的长度值无效 1：LEN 中的长度值有效
30:28	RSV	-	-	保留
27:16	LEN	RO	0x0	延迟线长度值，这些位反应了在输入时钟上升沿采样到的 12 个延迟单元的值。 只有 LENF=1 时，延迟线长度值 LEN 才有效
15:14	RSV	-	-	保留
13:8	UNIT	RW	0x0	定义每个单位延迟的延迟步数，仅当 SEN=1 时才能修改写入 单位延迟=初始延迟+UNIT×延迟步长
7:4	RSV	-	-	保留
3:0	SEL	RW	0x0	选择输出时钟的相位，仅当 SEN=1 时修改写入 输出时钟相位=输入时钟+SEL×单位延迟

10. 通用输入输出口（GPIO）

10.1. 概述

每组 GPIO 包含 16 个通用数据输入输出接口，这些管脚可以与其他功能管脚共享，这取决于芯片的配置。通过这些数据接口，可以配置任意数目的管脚作为中断信号输入。

复用功能（AF）的备用引脚，极大提高了端口利用的灵活性。GPIO 引脚通过配置相关的寄存器可以用作复用功能输入/输出引脚。

每个 GPIO 引脚可以独立配置为输出（推挽或开漏）、输入、外设复用功能或模拟模式。每个 GPIO 引脚可以独立配置为上拉、下拉或浮空。

10.2. 主要特性

- 所有输入/输出引脚方向都可以通过软件进行配置
- 支持施密特触发器输入
- 每个引脚具有弱上/下拉功能
- 支持推挽/开漏输出
- 支持置位/清零输出功能，可按位操作
- 支持模拟输入/输出配置
- 所有 GPIO 引脚可复用为 EXTI，且边沿可配置
- 支持复用功能输入/输出配置
- 支持端口配置锁定

10.3. 功能描述

在复位后，除 Wakeup 唤醒引脚、Debug 调试引脚、MCO 功能引脚为数字模式。大部分 GPIO 配置为模拟模式（数字功能失效，上下拉电阻失效）。

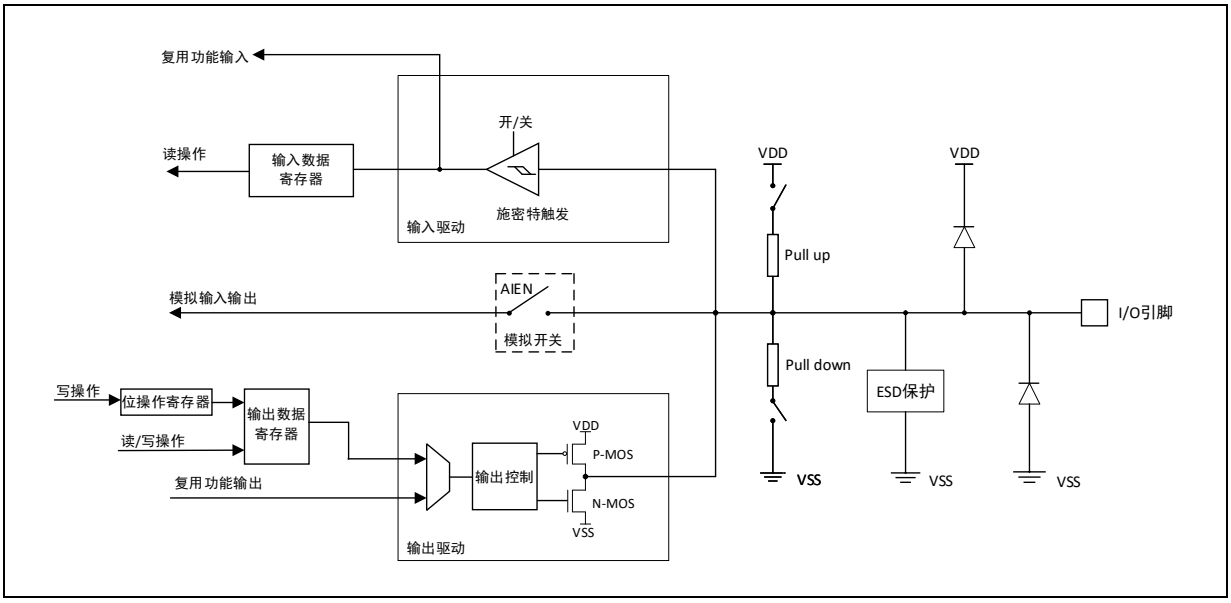
根据数据手册中列出的每个 I/O 端口的特性，可将 GPIO 端口配置为多种模式：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟
- 具有上拉或下拉功能的开漏输出
- 具有上拉或下拉功能的推挽输出
- 具有上拉或下拉功能的开漏复用功能
- 具有上拉或下拉功能的推挽复用功能

每个 I/O 端口位均可自由编程，但 I/O 端口寄存器必须按 32 位字进行访问。GPIOx_BSC 寄存器旨在实现对 GPIOx_ODATA 寄存器进行原子读取/修改访问。

10.3.1. 结构框图

图 10-1 3V 兼容的 GPIO 结构（TC）



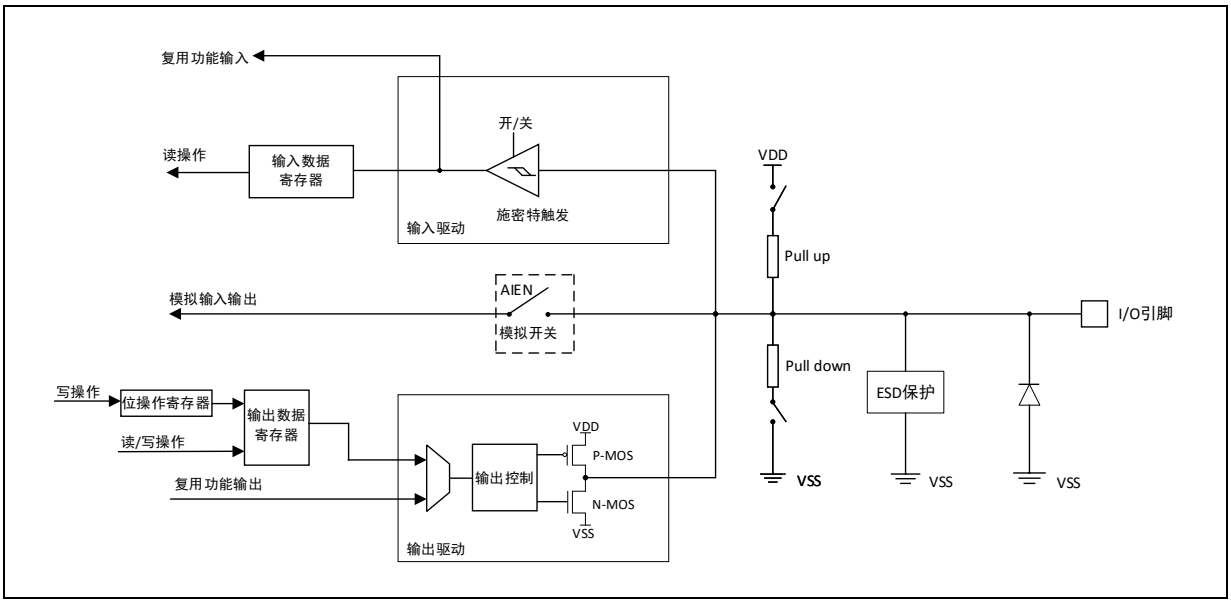
TC 管脚的特点：

使能内部上拉电阻，外部电压大于 VDD：就会发生电流倒灌进 VDD。

禁止内部上拉电阻，配置为输入模式或 OD 输出模式，外部电压大于 VDD+0.3V：会发生电流倒灌进 VDD。

禁止内部上拉电阻，配置为 PP 输出模式，且输出高电平，外部电压大于 VDD：就会发生电流倒灌进 VDD。

图 10-2 5V 容限的 GPIO 结构（FT）



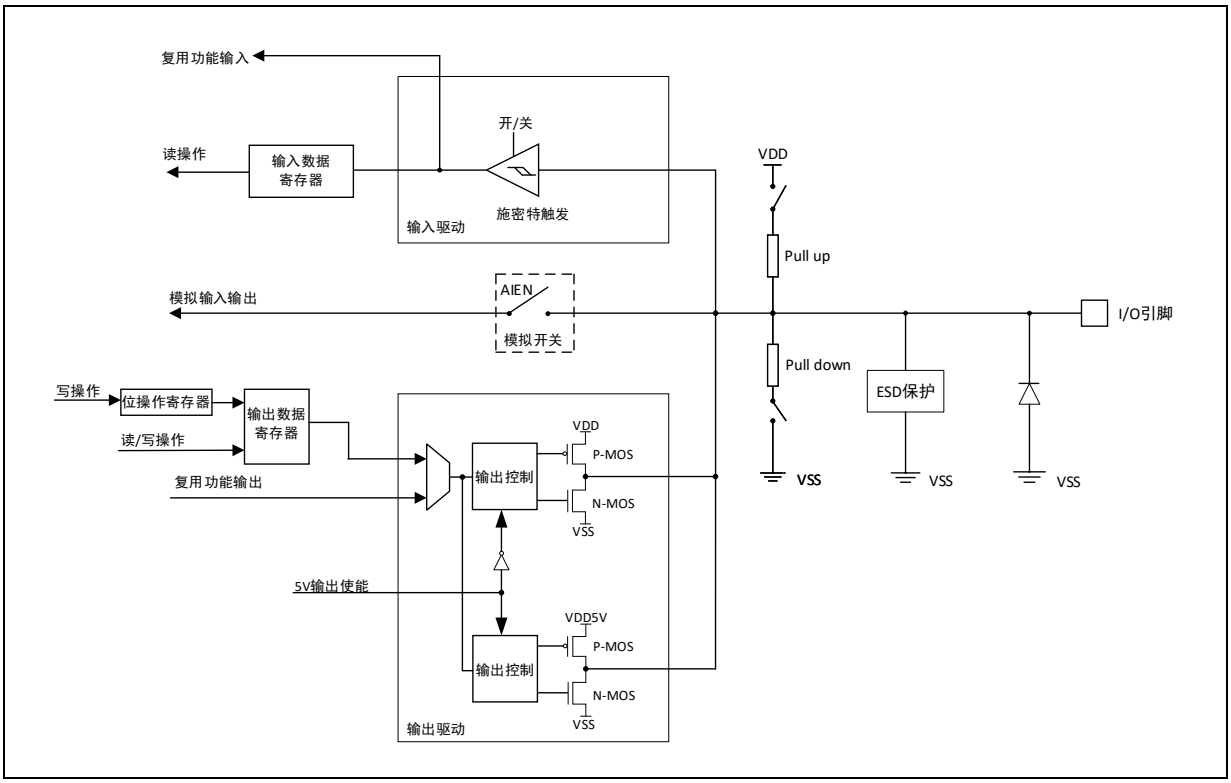
FT 管脚的特点：

使能内部上拉电阻，外部电压大于 VDD：就会发生电流倒灌进 VDD。

禁止内部上拉电阻，**配置为输入模式或 OD 输出模式**，即使外部电压大于 VDD：也不会有电流倒灌进 VDD。

禁止内部上拉电阻，配置为 PP 输出模式，且输出高电平，外部电压大于 VDD：就会发生电流倒灌进 VDD。

图 10-3 带 5V 输出驱动的 5V 容限 GPIO 结构（FT_5）



FT_5 管脚的特点：

5V 输出禁止时，特性与 FT 管脚完全相同。

VDD50 接 5V 电源，配置为输出模式，且 5V 输出使能时，引脚可输出 5V 电平。

输出 5V 电平时，如果使能内部上拉电阻，就会发生电流倒灌进 VDD。

5V 输出使能时，PP 和 OD 配置无效，均为 PP 输出。

10.3.2. 输入功能

当 GPIO 配置为输入功能时：

- 输出缓冲器被禁止
- 输入缓冲器被打开
- 施密特触发器可配置是否使能（寄存器 GPIOx_SMIT）
- 上拉和下拉电阻可配置是否打开（寄存器 GPIOx_PUPD）
- I/O 引脚的状态会不断采集并保存在 GPIO_IDATA 寄存器中。

10.3.3. 输出功能

当 GPIO 配置为输出功能时：

- 输出缓冲器被打开，可选择推挽输出或开漏输出
  - 开漏模式（OD）：输出控制寄存器设置为 0 时，相应引脚输出低电平；输出控制寄存器设置为 1，相应管脚处于高阻状态
  - 推挽模式（PP）：输出控制寄存器设置为 0 时，相应引脚输出低电平；输出控制寄存器设置为 1，相应引脚输出高电平

- 输入缓冲器被打开
- 施密特触发器可配置是否使能（寄存器 GPIOx_SMIT）
- 上拉和下拉电阻可配置是否打开（寄存器 GPIOx_PUPD）
- 输出驱动能力可配置选择（寄存器 GPIOx_DS0 和 GPIOx_DS1）
- 可通过配置 GPIO_SET、GPIO_CLR 控制 I/O 引脚状态或者通过 GPIO_ODATA 控制 I/O 引脚状态。
- 对输入数据寄存器的读访问可获取 I/O 状态（寄存器 GPIOx_IDATA）

10.3.4. 5V 输出驱动

对于支持 5V 输出驱动的 GPIO（类型为 FT_5，通过 VDD50 引脚供电），可通过 SYSCFG 模块的 GPIO5VOCR1 和 GPIO5VOCR2 寄存器配置，使能 5V 输出驱动。其他配置同上一节。

10.3.5. 模拟功能

当 GPIO 配置为模拟功能时：

- 输出缓冲器被禁止
- 输入缓存器被禁止
- 施密特触发器功能无效（无论配置是否使能）
- 模拟开关断开时，上拉和下拉电阻需要软件配置关闭（寄存器 GPIOx_PUPD）；模拟开关闭合时，上拉和下拉电阻无效。
- ADC 慢速通道（除 ADC12_INP1,ADC12_INP3,ADC3_INP2,ADC3_INP3 之外的其他 ADC 通道）、COMP 输入通道和 TKEY 通道通过模拟开关与引脚连接，因此需软件配置模拟开关闭合（寄存器 GPIOx_AIEN 对应 bit 置 1）；其他模拟通道直连引脚，需软件配置模拟开关断开（寄存器 GPIOx_AIEN 对应 bit 清 0）
- 读取输入数据寄存器时数值为 0
-

10.3.6. 复用功能

当端口配置为 AFIO 时，GPIO 引脚用作外设复用功能；使用复用功能前，先对 AFIO 配置寄存器和输出类型寄存器进行操作，复用输入或输出由外设决定。

- 输出缓冲器可配置推挽输出或开漏输出（寄存器 GPIOx_OTYP）
- 输出缓冲器由来自外设的信号驱动
- 输入缓冲器被打开
- 施密特触发器可配置是否使能（寄存器 GPIOx_SMIT）
- 上拉和下拉电阻可配置是否打开（寄存器 GPIOx_PUPD）
- 输出驱动能力可配置选择（寄存器 GPIOx_DS0 和 GPIOx_DS1）
- 对输入数据寄存器的读访问可获取 I/O 状态（寄存器 GPIOx_IDATA）

每个 I/O 引脚都有一个复用器，该复用器一次仅允许一个外设的复用功能（AF）连接到 I/O 引脚。这可以确保共用同一个 I/O 引脚的外设之间不会发生冲突。

该复用器采用 16 路复用功能输入（AF0 到 AF15），可通过 GPIOx_AF0（针对引脚 0 到 7）和 GPIOx_AF1（针对引脚 8 到 15）寄存器对这些输入进行配置：

- 复位后，复用器默认选择复用功能 0 (AF0)。通过寄存器 GPIOx_MD 配置复用模式
- 芯片数据手册中详细说明了每个引脚的特定复用功能分配

10.3.7. 外部中断/事件线

所有端口都有外部中断能力，可以在 EXTI 模块中配置，端口必须配置成输入模式。

10.3.8. GPIO 锁定功能

锁定机制用于冻结 IO 配置以防止被意外更改。当在一个端口位上执行了锁定序列（详见 GPIOx_LOCK 寄存器描述），在下一次复位之前，不能再更改端口的配置。

被锁定的寄存器包括：GPIOx_MD、GPIOx_OTYP、GPIOx_PUPD、GPIOx_AF0、GPIOx_AF1、GPIOx_ST、GPIOx_SMIT。

10.4. GPIO 寄存器描述

10.4.1. 寄存器列表

GPIOA 寄存器基地址：0x48000000

GPIOB 寄存器基地址：0x48000400

GPIOC 寄存器基地址：0x48000800

GPIOD 寄存器基地址：0x48000C00

GPIOE 寄存器基地址：0x48001000

GPIOF 寄存器基地址：0x48001400

GPIOG 寄存器基地址：0x48001800

GPIOH 寄存器基地址：0x48001C00

GPIOI 寄存器基地址：0x48002000

GPIOJ 寄存器基地址：0x48002400

GPIOK 寄存器基地址：0x48002800

GPIOL 寄存器基地址：0x48002C00

GPIOM 寄存器基地址：0x48003000

GPION 寄存器基地址：0x48003400

GPIOO 寄存器基地址：0x48003800

GPIOP 寄存器基地址：0x48003C00

GPIOQ 寄存器基地址：0x48004000

偏移	名称	复位值	描述
0x00	GPIOx_MD		GPIO 模式寄存器
0x04	GPIOx_OTYP		GPIO 输出类型寄存器

0x08	GPIOx_PUPD		GPIO 上下拉寄存器
0x0C	GPIOx_IDATA		GPIO 输入引脚映射寄存器
0x10	GPIOx_ODATA		GPIO 输出引脚映射寄存器
0x14	GPIOx_BSC		GPIO 输出置位/清零寄存器
0x18	GPIOx_AF0		GPIO 复用功能配置寄存器 0
0x1C	GPIOx_AF1		GPIO 复用功能配置寄存器 1
0x20	GPIOx_DS0		GPIO 驱动能力配置寄存器 0
0x24	GPIOx_DS1		GPIO 驱动能力配置寄存器 1
0x28	GPIOx_SMIT		GPIO 施密特使能寄存器
0x2C	GPIOx_LOCK		GPIO 配置锁定寄存器
0x30	GPIOx_AIEN		GPIO 模拟开关配置寄存器

10.4.2. GPIO 模式寄存器(GPIOx_MD: 00h)

位域	名称	属性	复位值	描述
31:0	MDY[15:0][1:0]	RW	GPIOA: 0xABFEFFFF GPIOB: 0xFFFFFEBF GPIOC: 0xFFFFFFFF GPIOD: 0xFFFFFFFF GPIOE: 0xFFC3FFFF GPIOF: 0xFFFFFFFF GPIOG: 0xFFFFFFFF GPIOH: 0xFFFFFFFF GPIOI: 0xFFFFFFFF GPIOJ: 0xFFFFFFFF GPIOK: 0xFFFFFFFF GPIOL: 0xFFFFFFFF GPIOM: 0xFFFFFFFF GPION: 0xFFFFFFFF GPIOO: 0xFFFFFFFF GPIOP: 0xFFFFFFFF GPIOQ: 0xFFFFFFFF	GPIOx 的 PINy(y=15 to 0)模式配置位： 00：输入模式 01：通用输出模式 10：复用功能模式 11：模拟功能模式（复位状态）

10.4.3. GPIO 输出类型寄存器(GPIOx_OTYP: 04h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	-

15:0	OTYP	RW		GPIOx 的 PINy(y=15 to 0)输出类型配置位： 0：输出推挽模式 1：输出开漏模式

10.4.4. GPIO 上下拉寄存器(GPIOx_PUPD: 08h)

位域	名称	属性	复位值	描述
31:0	PUPDY[15:0][1:0]	RW	GPIOA: 0x64100000 GPIOB: 0x01000180 GPIOC~D: 0x00000000 GPIOE: 0x00140000 GPIOF~I: 0x00000000 GPIOJ: 0xAA000000 GPIOK: 0x02AAAA80 GPIOL: 0x00000000 GPIOM: 0x0AA00000 GPION~O: 0x00000000 GPIOP: 0xAAA00000 GPIOQ: 0x00000000	GPIOx 的 PINy(y=15 to 0)上拉/下拉配置位： 00：无上/下拉 01：上拉 10：下拉 11：保留

10.4.5. GPIO 输入引脚映射寄存器（GPIOx_IDATA: 0Ch)

位域	名称	属性	复位值	描述
31:16	REV	-	-	-
15:0	GPIO_IDATA	RO	0x00000000	GPIOx 的 PINy(y=15 to 0)输入引脚映射寄存器： 当 GPIO 方向为输入有效，读获得外部引脚值；此寄存器为只读寄存器。

10.4.6. GPIO 输出引脚映射寄存器(GPIOx_ODATA: 10h)

位域	名称	属性	复位值	描述
31:16	REV	-	-	-
15:0	GPIO_ODATA	RW	0x00000000	GPIOx 的 PINy(y=15 to 0)输出引脚映射寄存器： 当 GPIO 方向为输出有效，写直接写至外部引脚，读获得外部引脚值。

10.4.7. GPIO 输出置位/清零寄存器(GPIOx_BSC: 14h)

位域	名称	属性	复位值	描述
31:16	GPIO_CLR	WO	0x00000000	GPIOx 的 PINy(y=15 to 0)输出清零寄存器： 0：无效操作； 1：当 IO 为输出时，IO 清零。 注：如果同时设置了 GPIO_SET 和 GPIO_CLR 的对应位，GPIO_SET 位起作用
15:0	GPIO_SET	WO	0x00000000	GPIOx 的 PINy(y=15 to 0)输出置位寄存器： 0：无效操作； 1：当 IO 为输出时，IO 置位。

10.4.8. GPIO 复用功能配置寄存器 0(GPIOx_AF0: 18h)

位域	名称	属性	复位值	描述
31:0	GPIO_AFSEL0Y[7:0][3:0]	RW	0x00000000	GPIOx 的 PINy(y=7 to 0)复用功能配置位 0000~1111：AF0~AF15

10.4.9. GPIO 复用功能配置寄存器 1(GPIOx_AF1: 1Ch)

位域	名称	属性	复位值	描述
31:0	GPIO_AFSEL1Y[15:8][3:0]	RW	0x00000000	GPIOx 的 PINy(y=15 to 8)复用功能配置位 0000~1111：AF0~AF15

10.4.10. GPIO 驱动能力配置寄存器 0(GPIOx_DS0: 20h)

位域	名称	属性	复位值	描述

31:0	GPIO_DS0Y[7:0][3:0]	RW	0x00000000	GPIOx 的 PINy(y=7 to 0) 驱动能力配置位(典型值) 0000：2mA 0001：4mA 0010：6mA 0011：8mA 0100：10mA 0101：12mA 0110：14mA 0111：16mA 1xxx：2mA 大驱动 IO 的驱动能力配置位(典型值) 0000：11mA 0001：14mA 0010：17mA 0011：20mA 0100：23mA 0101：26mA 0110：29mA 0111：32mA 1xxx：11mA

10.4.11. GPIO 驱动能力配置寄存器 1(GPIOx_DS1: 24h)

位域	名称	属性	复位值	描述

31:0	GPIO_DS1Y[15:8][3:0]	RW	0x00000000	GPIOx 的 PINy(y=15 to 8) 驱动能力配置位(典型值) 0000：2mA 0001：4mA 0010：6mA 0011：8mA 0100：10mA 0101：12mA 0110：14mA 0111：16mA 1xxx：2mA 大驱动 IO 的驱动能力配置位(典型值) 0000：11mA 0001：14mA 0010：17mA 0011：20mA 0100：23mA 0101：26mA 0110：29mA 0111：32mA 1xxx：11mA

10.4.12. GPIO 施密特使能寄存器(GPIOx_SMIT: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	-
15:0	GPIO_SMTEN	RW	0x00000000	GPIOx 的 PINy(y=15 to 0)施密特输入使能位： 0：禁止 1：使能

10.4.13. GPIO 配置锁定寄存器(GPIOx_LOCK: 2Ch)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	-

16	LOCK_KEY	RW	0x0	锁定键。该位可随时读出，它只可通过锁定键写入序列修改。 0：端口配置锁定键位无效 1：端口配置锁定键位有效； GPIOx_LOCK 寄存器被锁住，直到下次系统复位或 GPIOx 模块外设复位。 锁定键的写入序列： 写 1 -> 写 0 -> 写 1 -> 读 0 -> 读 1 最后一个读可省略，但可以用来确认锁定键已被激活。 注：在进行锁定键的写入序列时，不能改变 LOCK_EN[15:0]的值；锁定键写入序列中的任何错误都将不能激活锁定键。
15:0	LOCK_EN	RW	0x00000000	GPIOx 的 PINy(y=15 to 0)配置锁定使能位： 0：禁止 1：使能 注：当 LOCK_KEY=0 时该位可写入

10.4.14. GPIO 模拟开关配置寄存器(GPIOx_AIEN: 30h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	-
15:0	GPIO_AIEN	RW	0x00000000	GPIOx 的 PINy(y=15 to 0)模拟开关配置位（当 GPIOx 为模拟功能模式时有效）： 0：开关打开 1：开关闭合

11. 定时器的分类

通用 MCU 的定时器包括以下几种类型：

- 基本定时器，包含基本的自动装载和可编程预分频的计数器；
- 通用定时器，在基本定时器基础上，加入输入/输出通道控制、触发控制和编码功能；
- 高级定时器，在通用定时器基础上，加入输出互补、刹车控制功能。

表 11-1 定时器的类型

功能	高级定时器 TIM1/8/20	通用定时器 TIM2/3/4/5/23/24	通用定时器 TIM9/12	通用定时器 TIM10/11/13/14	通用定时器 TIM15/25	通用定时器 TIM16/17/18/19	基本定时器 TIM6/7/21/22
预分频器分辨率	16 位	16 位	16 位	16 位	16 位	16 位	16 位
计数器分辨率	16 位	TIM2/5/23/24 为 32 位 TIM3/4 为 16 位	16 位	16 位	16 位	16 位	16 位
计数模式	向上， 向下， 中央对齐	向上， 向下， 中央对齐	向上	向上	向上	向上	向上
重复计数器	●	×	●	●	●	●	×
捕获/比较通道	4	4	2	1	2	1	0
互补输出和死区插入	支持，4 个通道	×	×	×	支持，1 个通道	支持，1 个通道	×
刹车功能	●	×			●	●	×
单脉冲(OPM)模式	●	●	●	●	●	●	×
编码模式	支持，编码模式 1/2/3	支持，编码模式 1/2/3	×	×	×	×	×
外部时钟模式 2	●	●	×	×	×	×	×
外部事件清除 OCxREF	●	×	×	×	×	×	×
6 步 PWM	●	×	×	×	×	×	×
强制输出模式	●	●	●	●	●	●	×
输入 XOR 功能	●	●	×	×	×	×	×
DMA 请求产生	●	●	×	×	●	●	●

●代表支持该功能。×代表不支持该功能。

12. 高级定时器（TIM1/8/20）

12.1. 概述

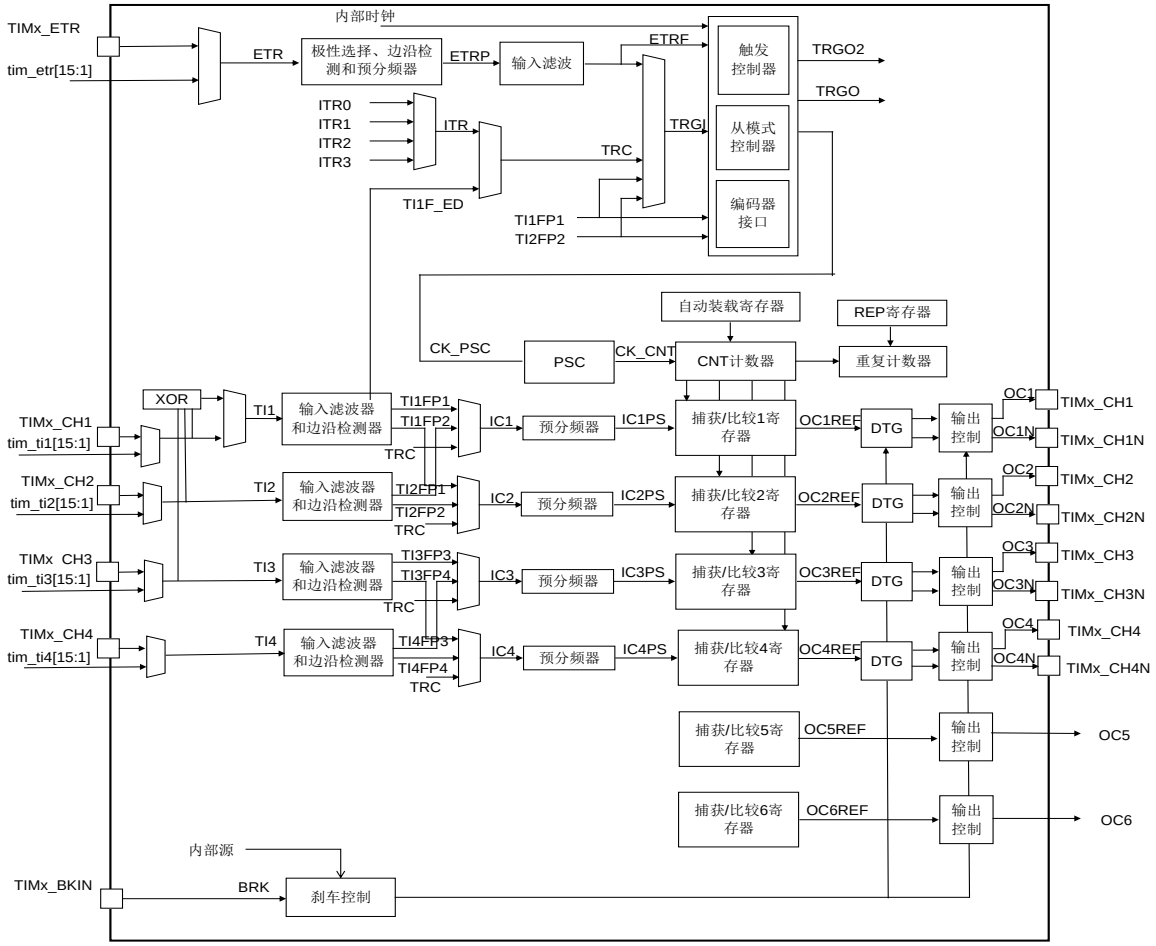
高级控制定时器（TIM1/8/20）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

12.2. 主要特性

- 16 位向上、向下、向上/下自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边沿或中间对齐模式)
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 支持针对定位的增量(正交)编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

12.3. 结构框图

图 12-1 高级控制定时器框图



12.4. TIMx 输入映射

表 12-1 TIMx 内部触发输入（ITRx）

	TIM1 源	TIM8 源
ITR0	-	tim1_trgo
ITR1	tim2_trgo	tim2_trgo
ITR2	tim3_trgo	tim3_trgo
ITR3	tim4_trgo	tim4_trgo
ITR4	-	-
ITR5	tim8_trgo	-
ITR6	tim15_trgo	tim15_trgo
ITR7	tim16_oc1	tim16_oc1
ITR8	tim17_oc1	tim17_oc1
保留	-	-

表 12-2 TIMx ETR 输入

ETRSEL[3:0]	TIM1 源	TIM8 源
0000	tim1_etr(IO)	tim8_etr(IO)
0001	comp1_out	comp1_out
0010	comp2_out	comp2_out
0011	comp3_out	comp3_out
0100	comp4_out	comp4_out
0101	adc1_awd	adc2_awd
0110	adc2_awd	-
保留	-	-

表 12-3 TIMx 通道 1 输入

TI1SEL[3:0]	TIM1 源	TIM8 源
0000	tim1_ch1	tim8_ch1
0001	comp1_out	comp1_out
0010	comp2_out	comp2_out
0011	comp3_out	comp3_out
0100	comp4_out	comp4_out
保留	-	-

表 12-4 TIMx 通道 2 输入

TI2SEL[3:0]	TIM1 源	TIM8 源
0000	tim1_ch2	Tim8_ch2
保留	-	-

表 12-5 TIMx 通道 3 输入

TI3SEL[3:0]	TIM1 源	TIM8 源
0000	tim1_ch3	Tim8_ch3
保留	-	-

表 12-6 TIM1 通道 4 输入

TI4SEL[3:0]	TIM1 源	TIM8 源
0000	Tim1_ch4	Tim8_ch4
保留	-	-

表 12-7 TIM1 OCREF_CLR 输入

OCRSEL[2:0]	TIM1 源	TIM8 源
0000	comp1_out	comp1_out
0001	comp2_out	comp2_out
0010	comp3_out	comp3_out
0011	comp4_out	comp4_out
保留	-	-

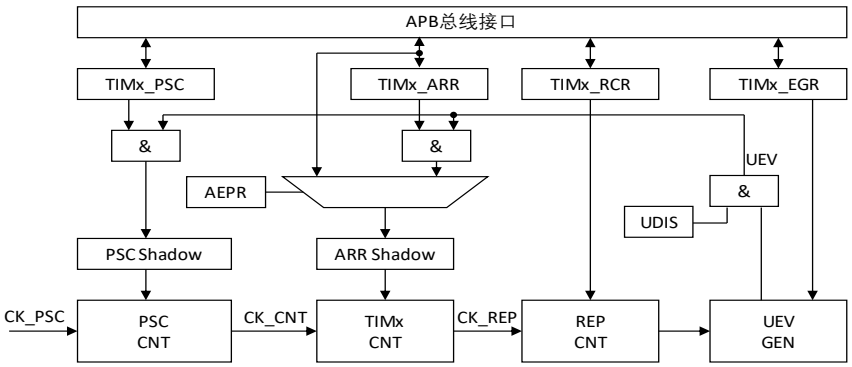
12.5. 功能描述

12.5.1. 计数单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)
- 重复次数寄存器（TIMx_RCR)

图 12-2 计数单元的结构



如上图所示，自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

高级定时器支持三种计数模式：

- 向上计数模式

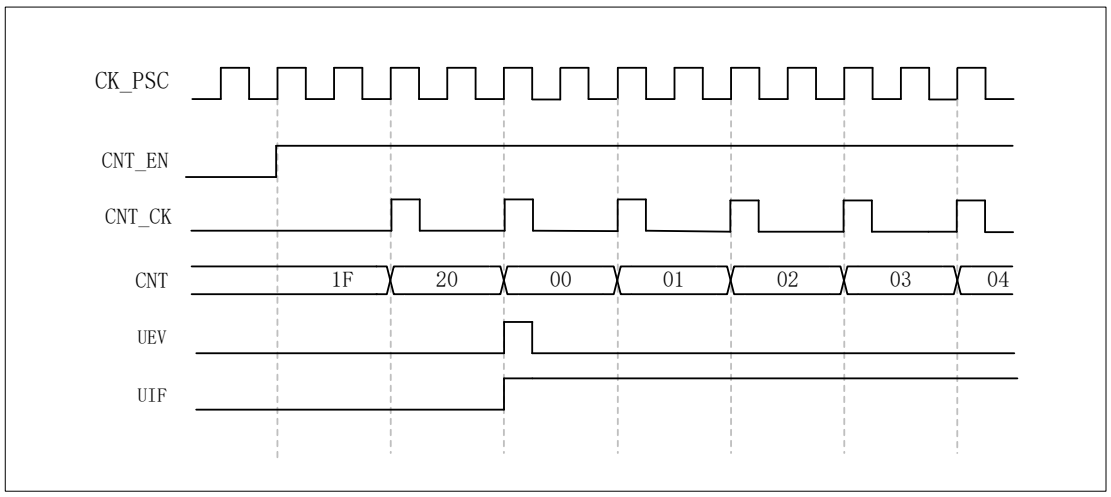
在向上计数模式中，计数器从 0 计数到自动重载载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件，如果 UDIS=0，就会产生一次更新事件。更新事件也可以由 UG 位置位产生（通过 EGR 寄存器软件置位或通过硬件从模式方式)。当更新事件产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件产生的更新事件才产生更新中断和更新 DMA 请求。

发生更新事件时，将更新所有寄存器且将更新标志(TIMx_SR 寄存器中的 UIF 位) 置 1 (取 决于 URS 位)：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容。
- 使用预装载值 (TIMx_ARR) 更新自动重载影子寄存器。
- 预分频器的缓冲区中将重新装载预装载值(TIMx_PSC 寄存器的内容) 。

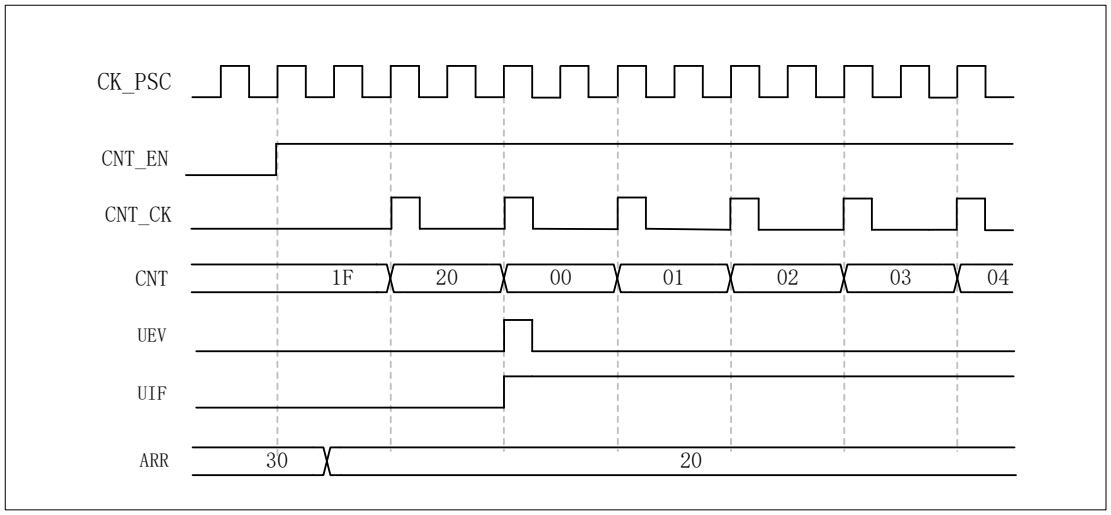
下图示例说明当 TIMx_ARR=0x20 时二分频下计数器的行为示意图。

图 12-3 向上计数模式更新事件



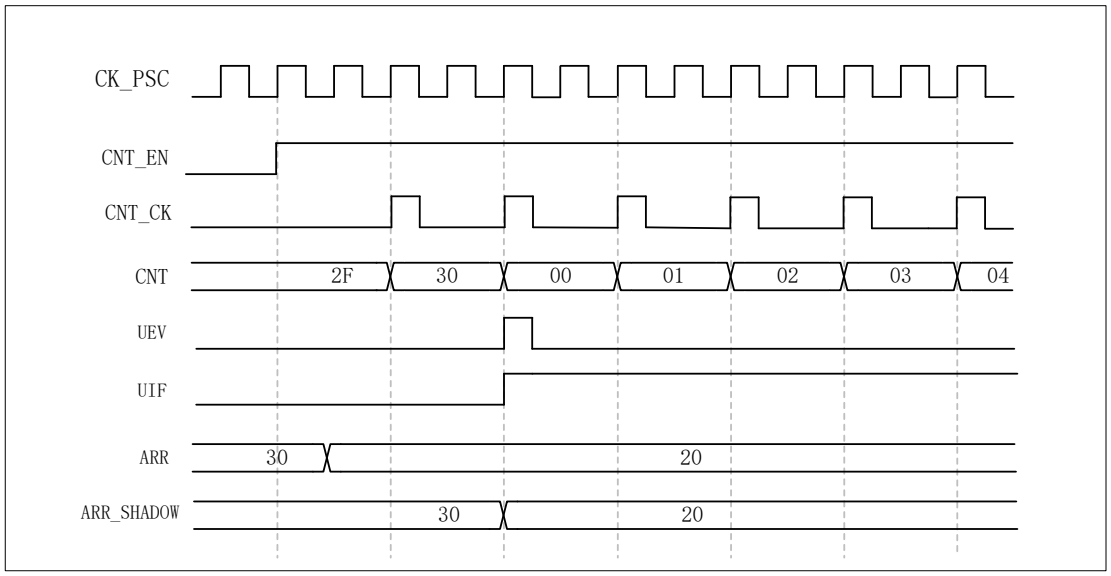
下图为 ARPE=0 时计数器的行为示意图：

图 12-4 向上计数模式（禁止自动重装载预装载）



下图为 ARPE=1 时计数器的行为示意图：

图 12-5 向上计数（使能自动重装载预装载）



● 向下计数模式

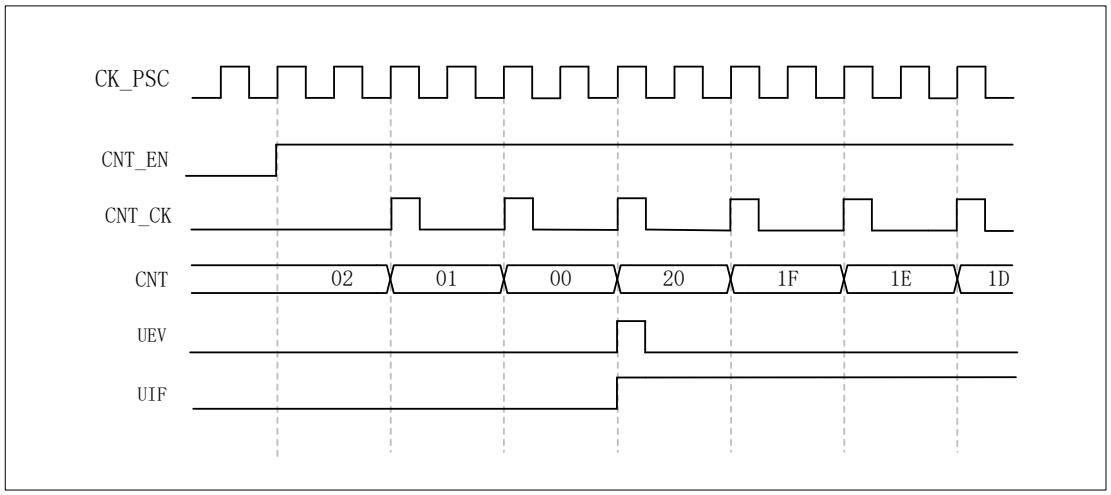
在向下模式中，计数器从自动重载值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动重载值重新开始并且产生一个计数器向下溢出事件，如果 UDIS=0，就会产生一次更新事件。更新事件也可以由 UG bit 置位产生（通过 EGR 寄存器软件置位或通过硬件从模式方式)。当更新事件产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件产生的更新事件才产生更新中断和更新 DMA 请求。

发生更新事件时，将更新所有寄存器且将更新标志(TIMx_SR 寄存器中的 UIF 位) 置 1 (取决于 URS 位)：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容。
- 使用预装载值 (TIMx_ARR) 更新自动重载影子寄存器。
- 预分频器的缓冲区中将重新装载预装载值(TIMx_PSC 寄存器的内容) 。

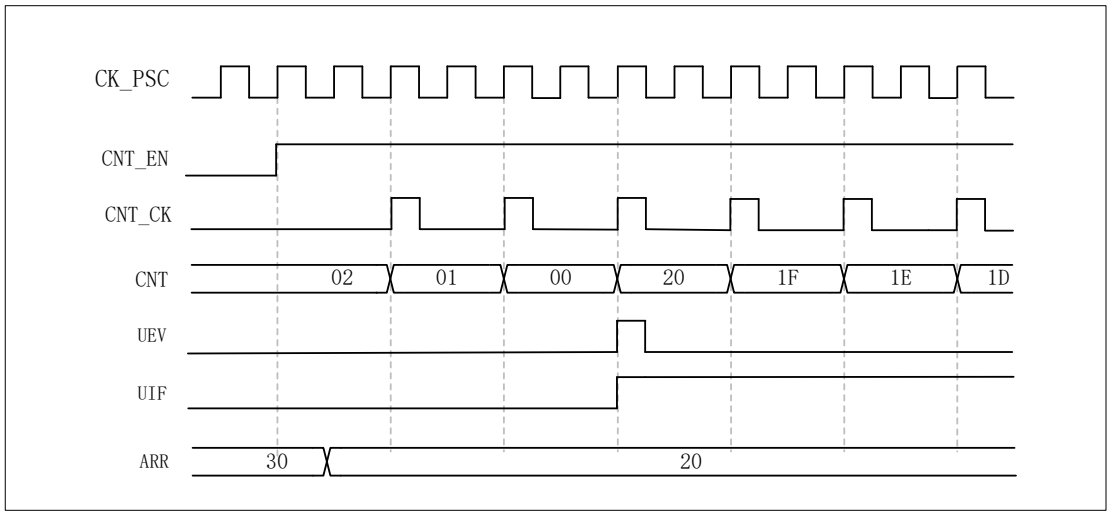
下图示例说明当 TIMx_ARR=0x20 时二分频下计数器的行为示意图。

图 12-6 向下计数模式更新事件



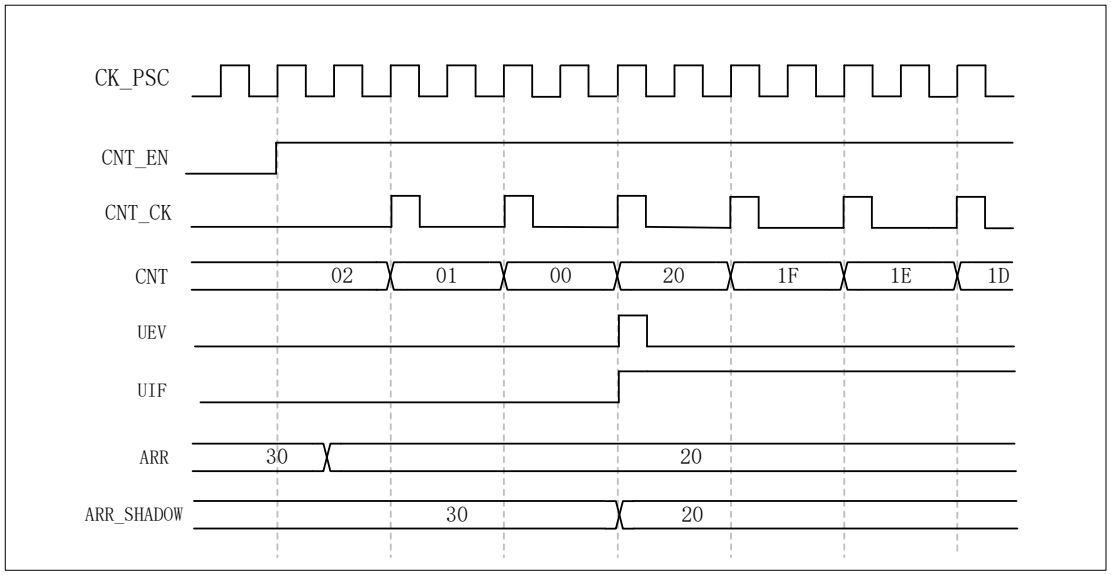
下图为 ARPE=0 时计数器的行为示意图：

图 12-7 向下计数模式（禁止自动重装载预装载）



下图为 ARPE=1 时计数器的行为示意图：

图 12-8 向下计数模式（使能自动重装载预装载）



● 中央对齐模式

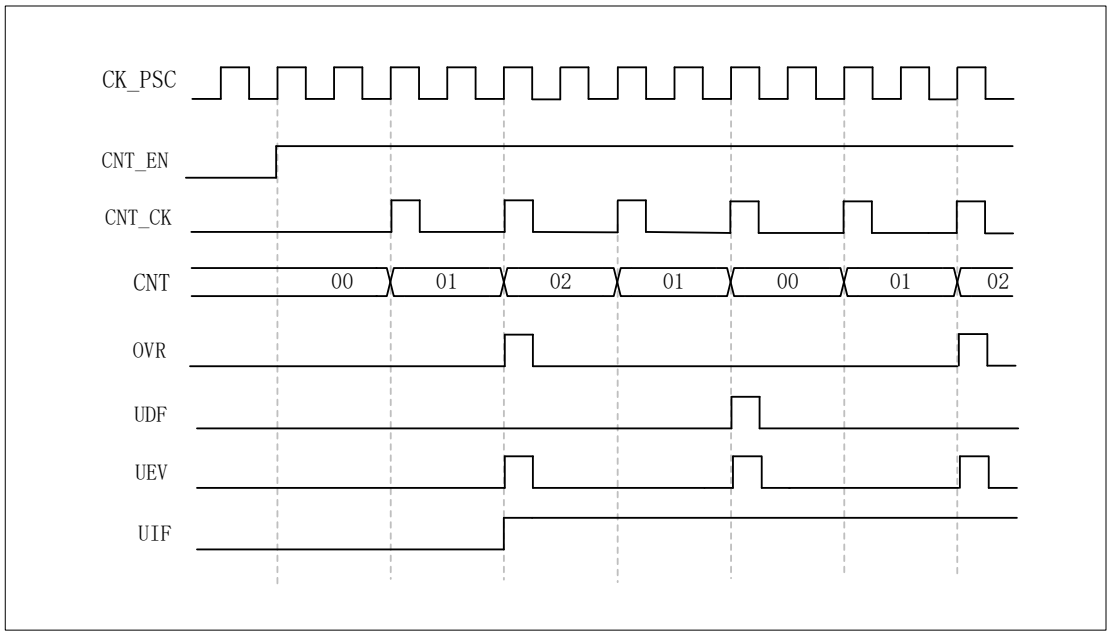
在中央对齐模式下，计数器交替的从 0 开始向上计数到 ARR-1，产生一次上溢出事件，然后再从 ARR 向下计数到 1，然后产生下溢出事件，然后又从 0 开始计数。向上计数时，定时器模块在计数器计数到自动加载值减一则产生一个上溢事件；向下计数时，定时器模块在计数器计数到 1 时产生一个下溢事件。无论上溢还是下溢，当 UDIS=0 时，都会产生一次更新事件。当更新事件（UEV/UG/硬件从模式方式）产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件（UEV）产生的更新事件才产生更新中断和更新 DMA 请求。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时， 中央对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在下溢出情况下置 1，即：计数器递减计数溢出(中心对齐模式 1，CMS = “01”）、计数器递增计数溢出(中心对齐模式 2，CMS = “10”)以及计数器递增/递减计数溢出(中心对齐模式 3，CMS = “11”)。

在中央计数模式中，TIMx_CR1 寄存器中的计数方向控制位 DIR 为只读，指示计数方向。计数方向被硬件自动更新。

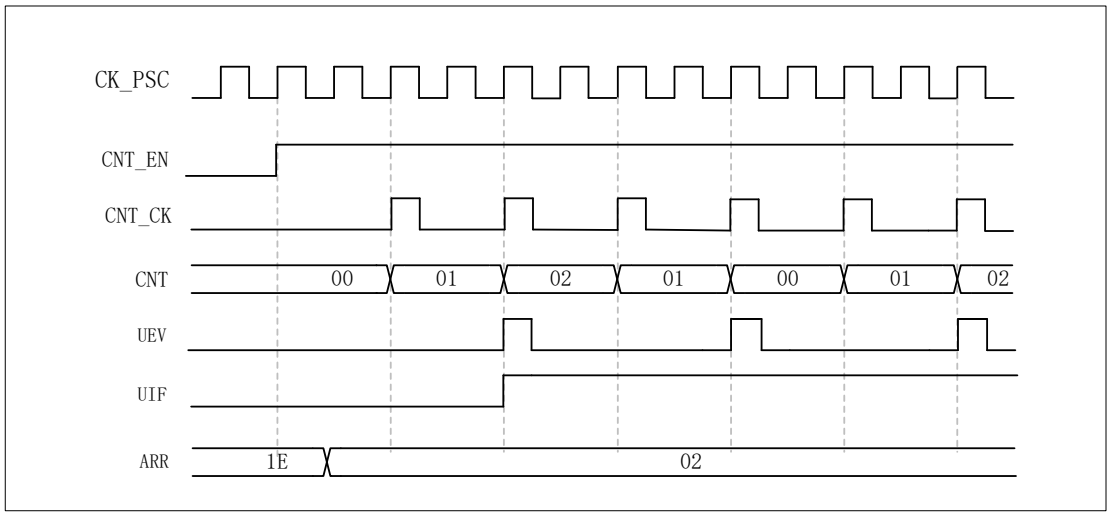
下图示例说明当 TIMx_ARR=0x02 时二分频下计数器的行为示意图。

图 12-9 中央对齐模式更新事件



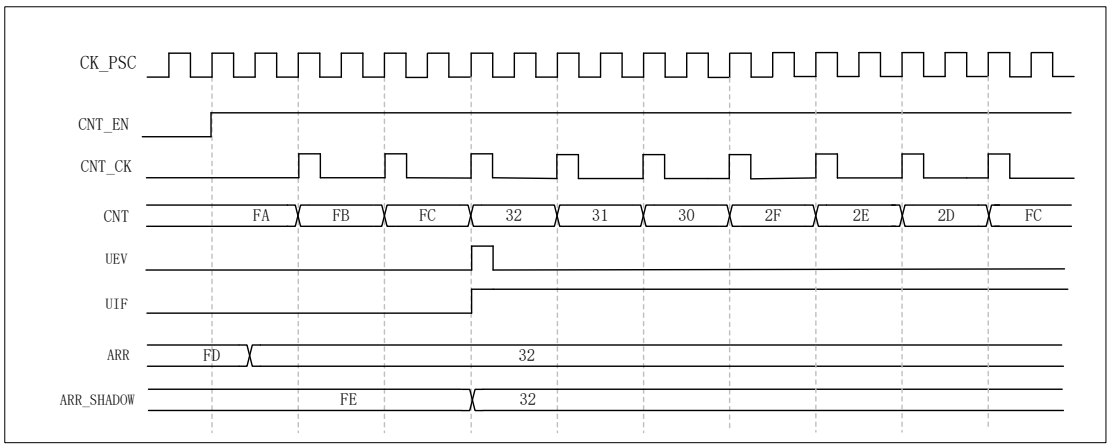
下图为 ARPE=0 时计数器的行为示意图：

图 12-10 中央对齐模式（禁止自动重装载预装载）



下图为 ARPE=1 时计数器的行为示意图：

图 12-11 中央对齐模式（使能自动重装载预装载）

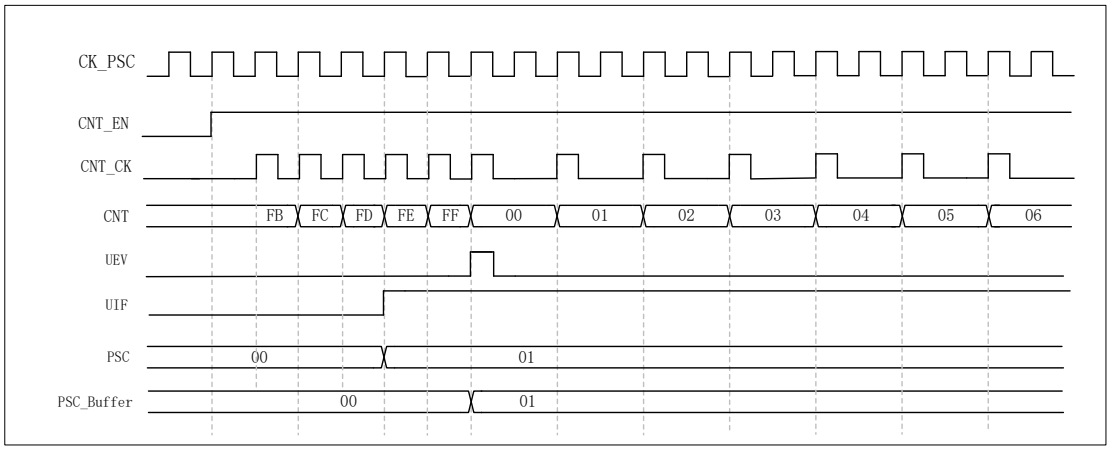


12.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器：PSC_Buffer，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 12-12 预分频更改示意图



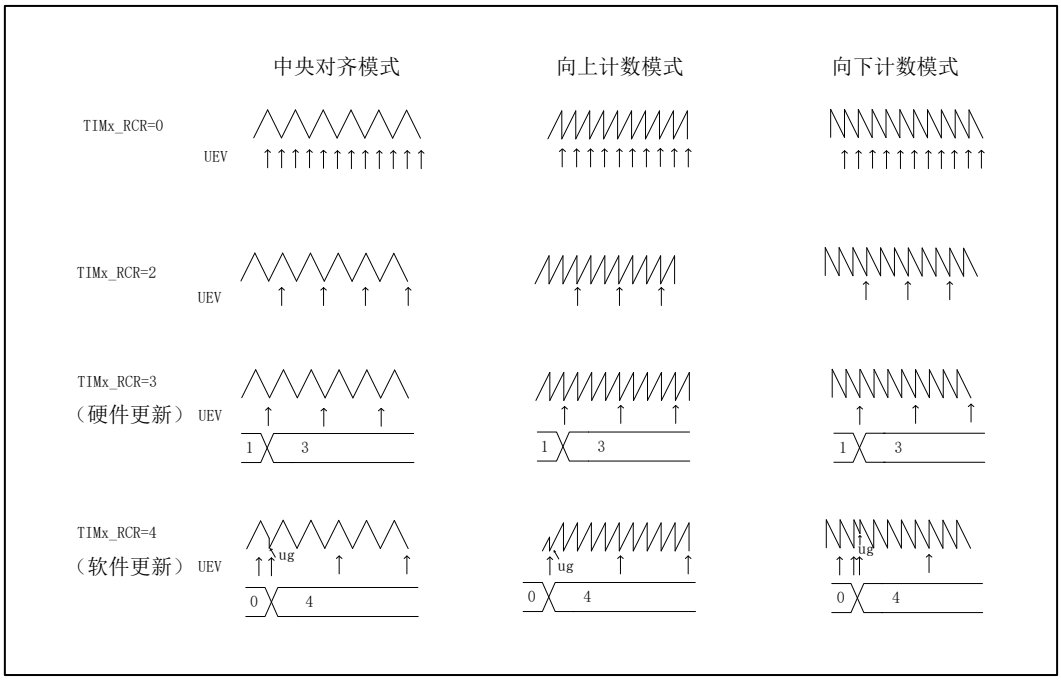
12.5.3. 重复计数器

重复计数器是一个 8 位的递减计数器。重复计数器为 0 时所发生的溢出事件，实际上更新事件 UEV 只能在重复计数器计数达到 0 时产生。重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。尽管这使得最大重复次数不超过 32768 个 PWM 周期，但在每个 PWM 周期内可更新占空比两次。当在中心对齐模式下，每个 PWM 周期仅刷新一次比较寄存器时，由于模式的对称性，最大分辨率为 2xTck。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

图 12-13 重复计数器溢出事件



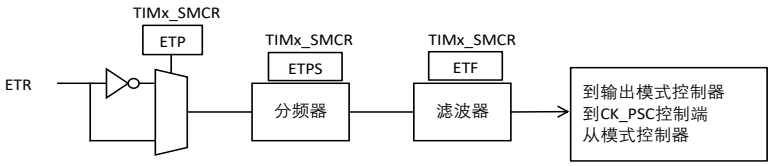
12.5.4. 外部触发输入

定时器具有一个外部触发输入 ETR，它可用作：

- 外部时钟(外部时钟模式 2)
- 用于从模式的触发信号(请参见 TIMx_SMCR->TS 位)
- 用于逐周期电流调节的 PWM 复位输入

下图介绍了 ETR 输入的调节过程。输入极性通过 TIMx_SMCR 寄存器中的 ETP 位定义，触发信号可通过 ETPS[1:0] 比特编程的分频比进行预分频，然后通过 ETF[3:0] 比特进行数字滤波。

图 12-14 ETR 触发滤波



ETR 输入来自多个源：输入引脚（默认配置）、比较器输出和模拟看门狗等。使用 TIMx_AF1->ETRSEL[3:0]比特进行选择。

12.5.5. 时钟源选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚

- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。

图 12-15 外部时钟模式 1

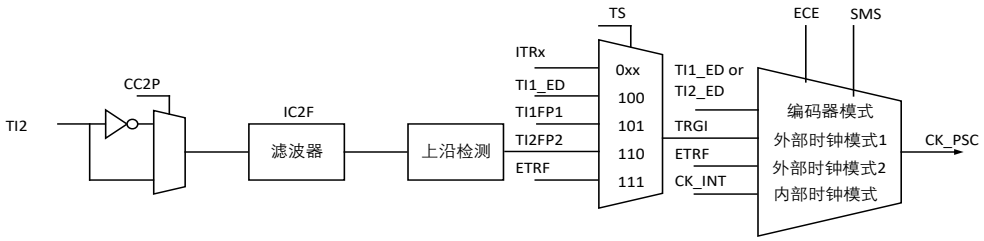
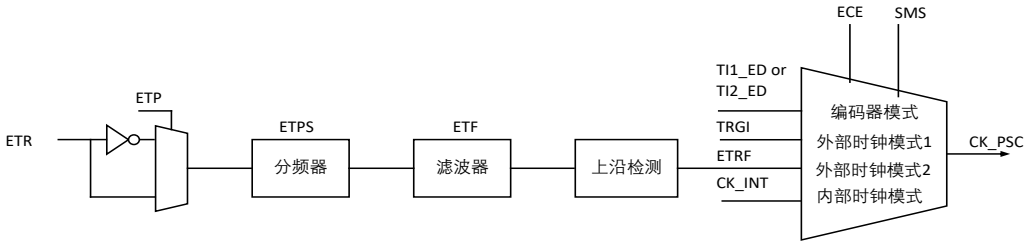


图 12-16 外部时钟模式 2



但如果按功能划分如以上 2 张图示所示，并按照定时器的从模式控制寄存器 TIMx_SMCR 的 ECE 和 SMS 的控制，应该分为以下几种模式：

- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有多个信号源，并由 TIMx_SMCR.TS[4:0]寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 1（ITR1）
  - TS=010，内部触发 2（ITR2）
  - TS=011，内部触发 3（ITR3）
  - TS=100，TI1 的边沿检测器（TI1F_ED）
  - TS=101，滤波后的定时器输入 1（TI1FP1）
  - TS=110，滤波后的定时器输入 2（TI2FP2）
  - TS=111，外部触发输入（ETRF）
  - ...
- 外部时钟模式 2：SMS=xxx，ECE=1，CK_PSC 来自 ETRF
- 编码器模式
  - 编码器模式 1：SMS=001，ECE=0，CK_PSC 来自 TI1FP1 的上下沿
  - 编码器模式 2：SMS=010，ECE=0，CK_PSC 来自 TI2FP2 的上下沿
  - 编码器模式 3：SMS=011，ECE=0，CK_PSC 来自 TI1FP1 和 TI2FP2 的上下沿

12.5.6. 编码器模式

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。在这个模式下，计数器依照增量编码器的

速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

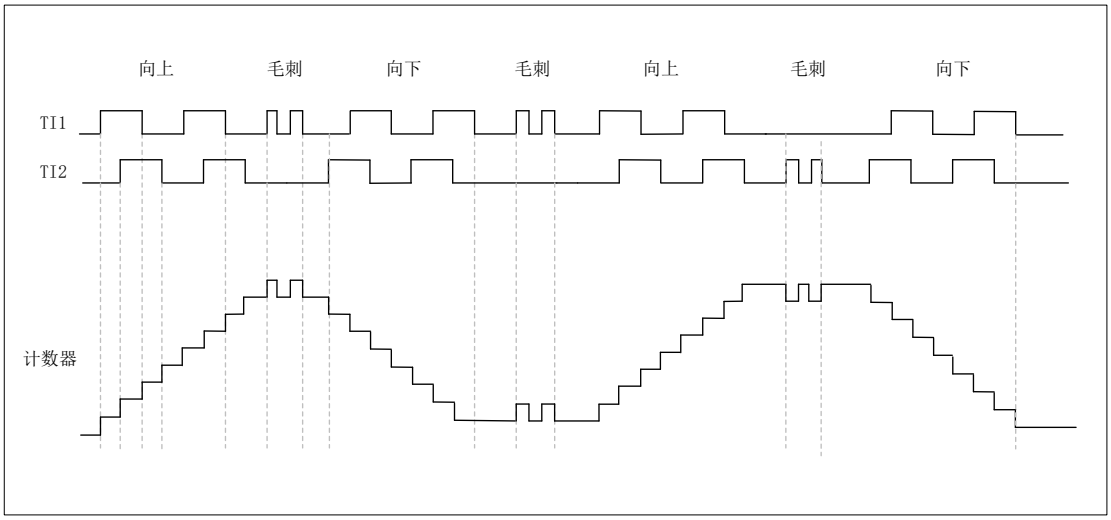
表 12-8 计数器方向和编码器信号的关系

有效边沿	相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1		TI2FP2	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
TI1 和 TI2 都计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。

图 12-17 编码器模式下计数器操作实例



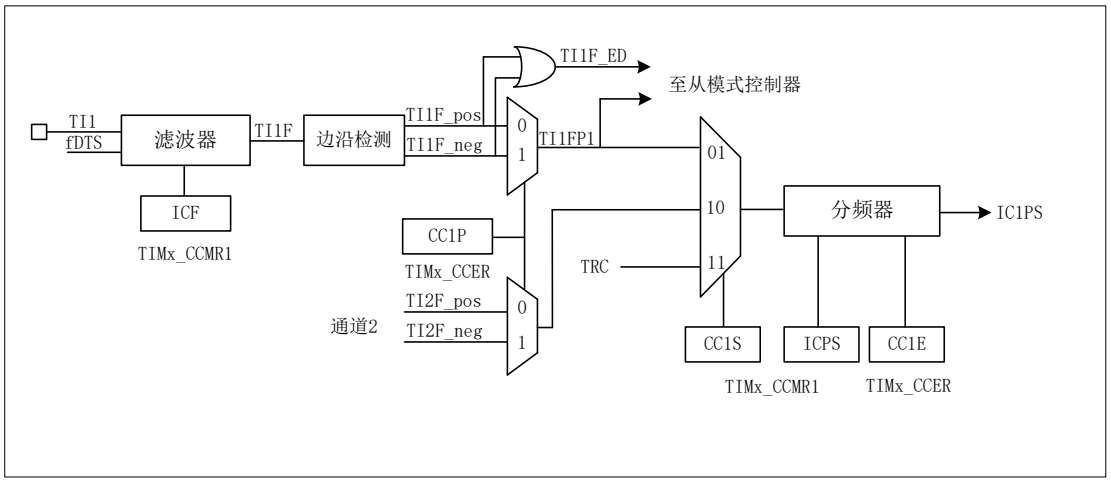
12.5.7. 捕获/比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 4 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄

存器(ICxPS)。

图 12-18 捕获/比较通道（如：通道 1 输入部分）



注：fDTS（Frequency-Discrimination-Thresholds）指死区发生器和数字滤波器所用的采样时钟频率。

捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 12-19 捕获/比较通道 1 的主电路

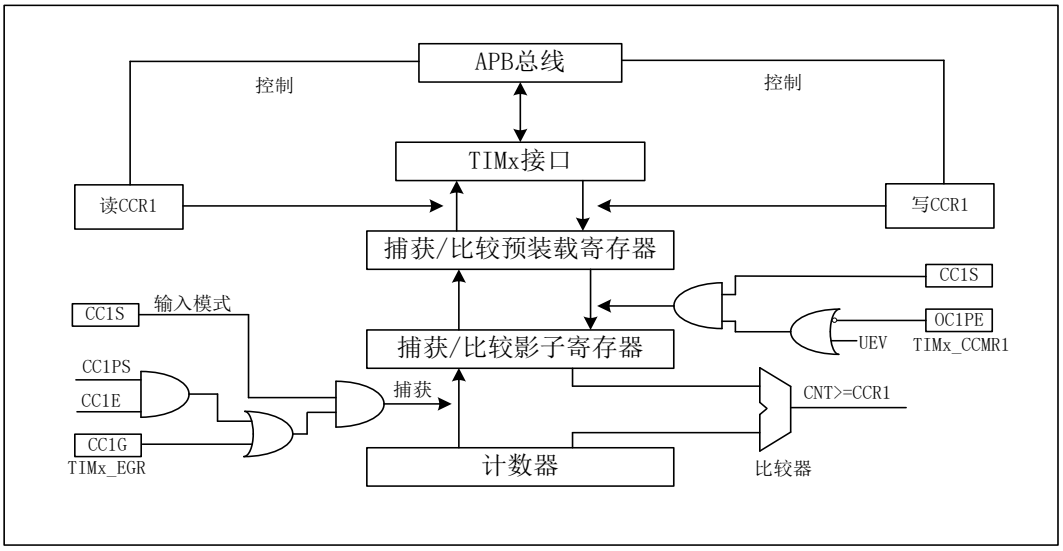


图 12-20 捕获/比较通道的输出部分(通道 1 至 3)

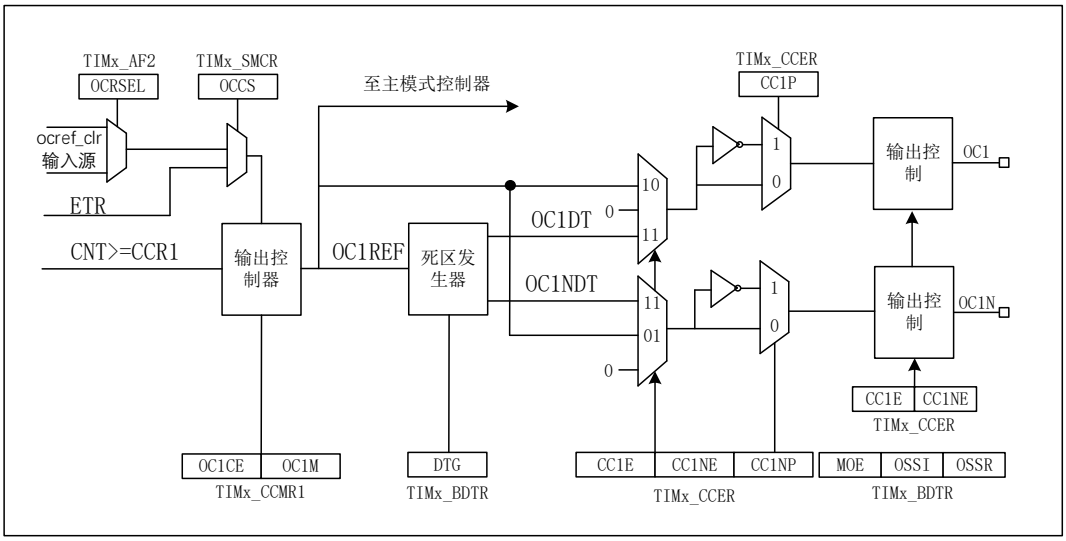
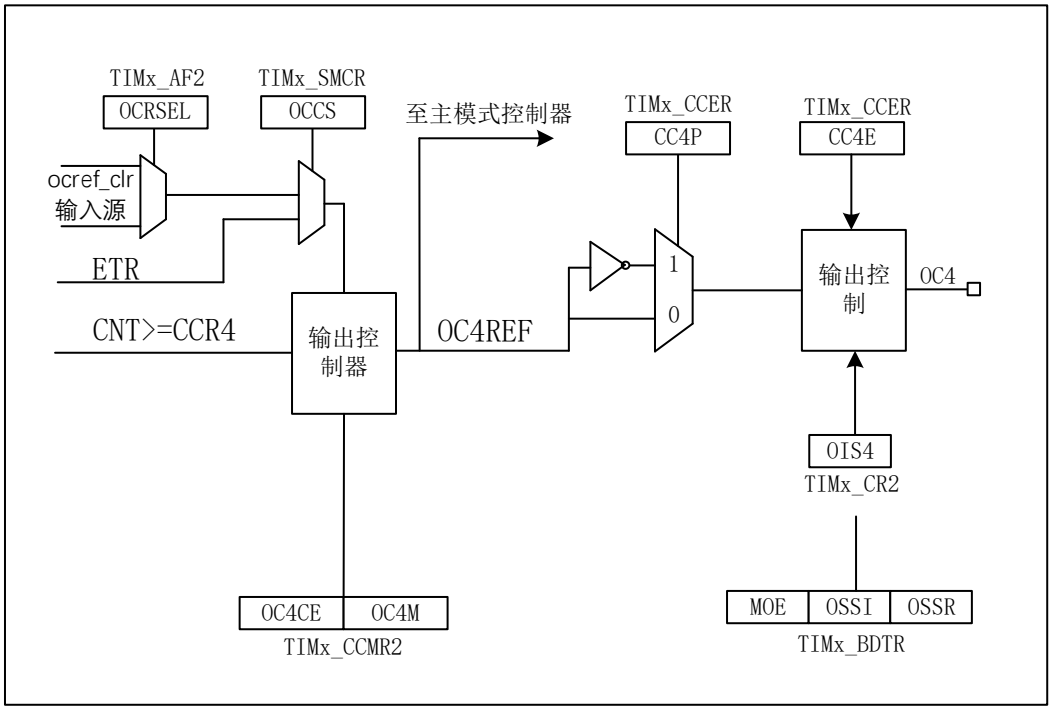


图 12-21 捕获/比较通道的输出部分(通道 4)



12.5.7.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

6）如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

12.5.7.2. PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

1）两个 ICx 信号被映射至同一个 TIx 输入。

2）这 2 个 ICx 信号为边沿有效，但是极性相反。

3）其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。 例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

4）选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。

5）选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。

6）选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。

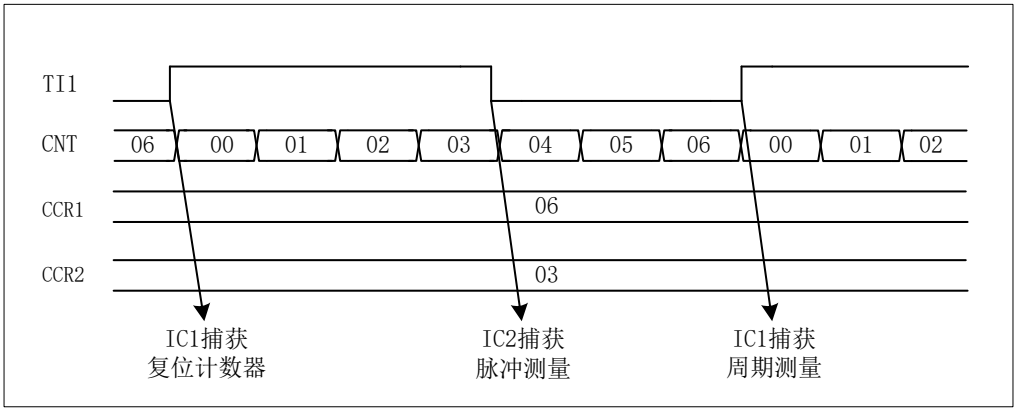
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

8）选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。

9）配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。

10）使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

图 12-22 PWM 输入模式时序



12.5.7.3. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

12.5.7.4. 输出比较模式

此项功能是用来控制一个输出波形，或者指示经过一段给定的时间。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

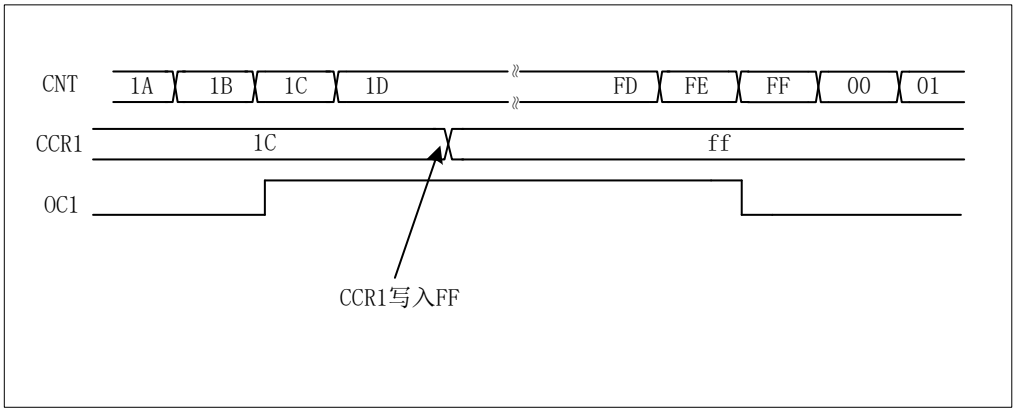
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 12-23 输出比较模式，翻转 OC1



12.5.7.5. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。 OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

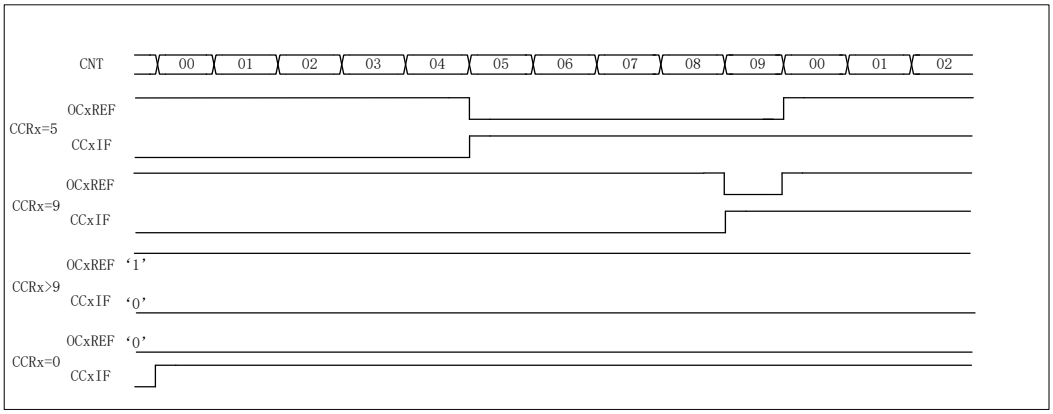
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ PWM 边沿对齐模式

● 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=9 时边沿对齐的 PWM 波形实例。

图 12-24 边沿对齐的 PWM 波形（ARR=9）



● 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

■ PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

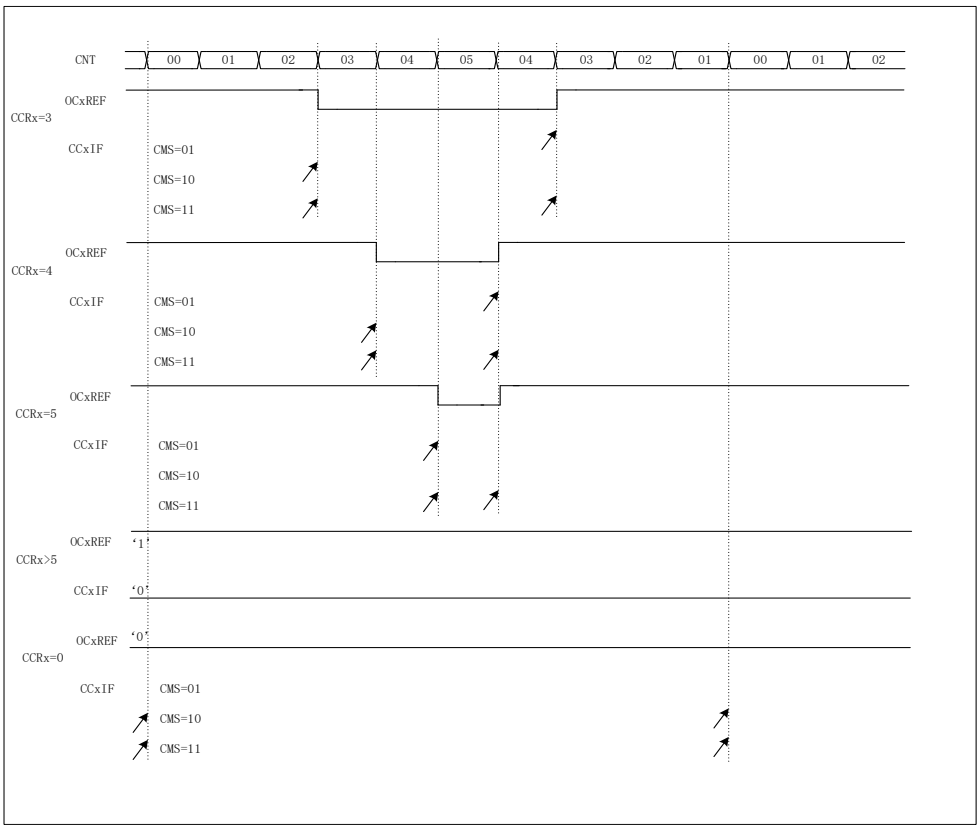
下图给出了一些中央对齐的 PWM 波形的例子。

1）TIMx_ARR=5

2）PWM 模式 1

3）TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图 12-25 中央对齐的 PWM 波形（ARR=5）



使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

12.5.7.6. 互补输出和死区插入

高级控制定时器能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。 这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

配置 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性(主输出 OCx 或互补输出 OCxN)。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，见表“带刹车功能的互补输出通道 OCx 和 OCxN 的控制位”。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

- OCx 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。
- OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。

如果延迟大于当前有效的输出宽度(OCx 或者 OCxN)，则不会产生相应的脉冲。 下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。(假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1)

图 12-26 带死区插入的互补输出

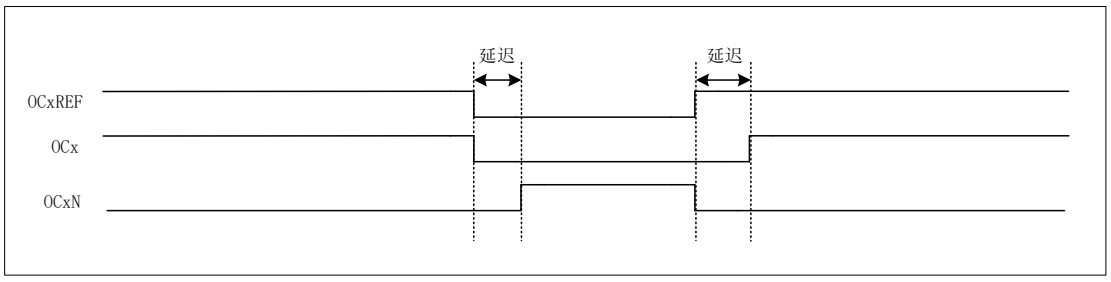


图 12-27 死区波形延迟大于负脉冲

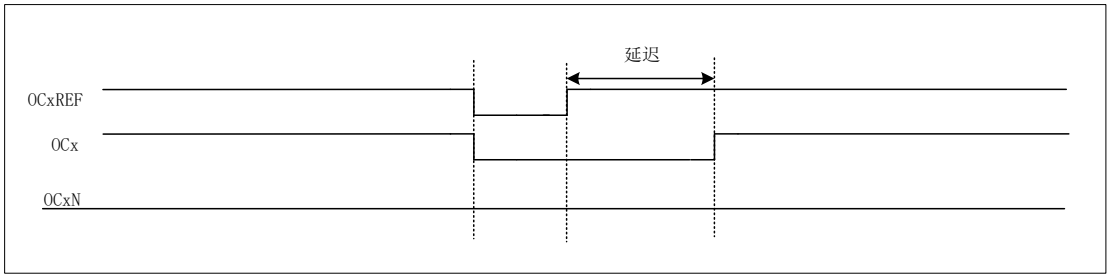
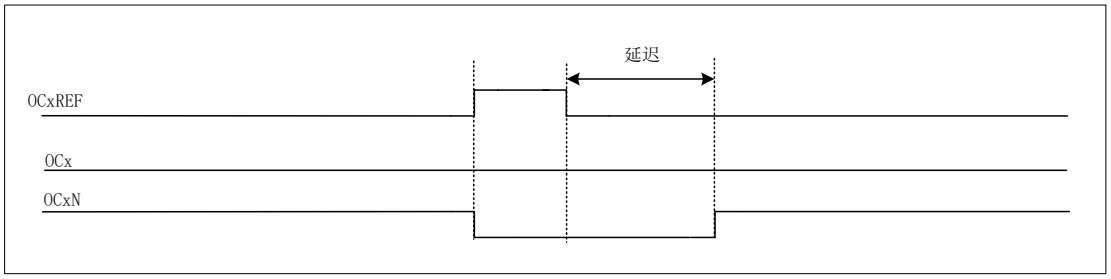


图 12-28 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。

重定向 OCxREF 到 OCx 或 OCxN

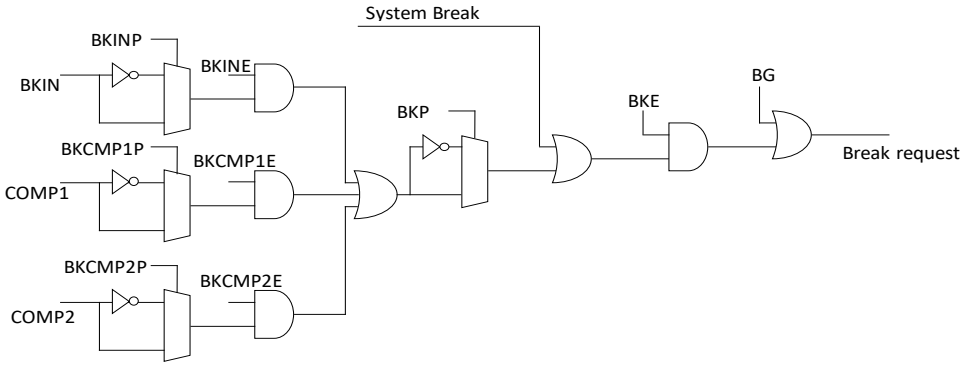
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注： 当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

12.5.7.7. 使用刹车功能

当使用刹车功能时，依据相应的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。刹车源可以选择刹车输入引脚，也可以选择比较器输出。另外，系统也可以产生刹车请求，如图所示。

图 12-29 刹车



系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的，在实际信号(作用在输出端)和同步控制位(在 TIMx_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。

- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时

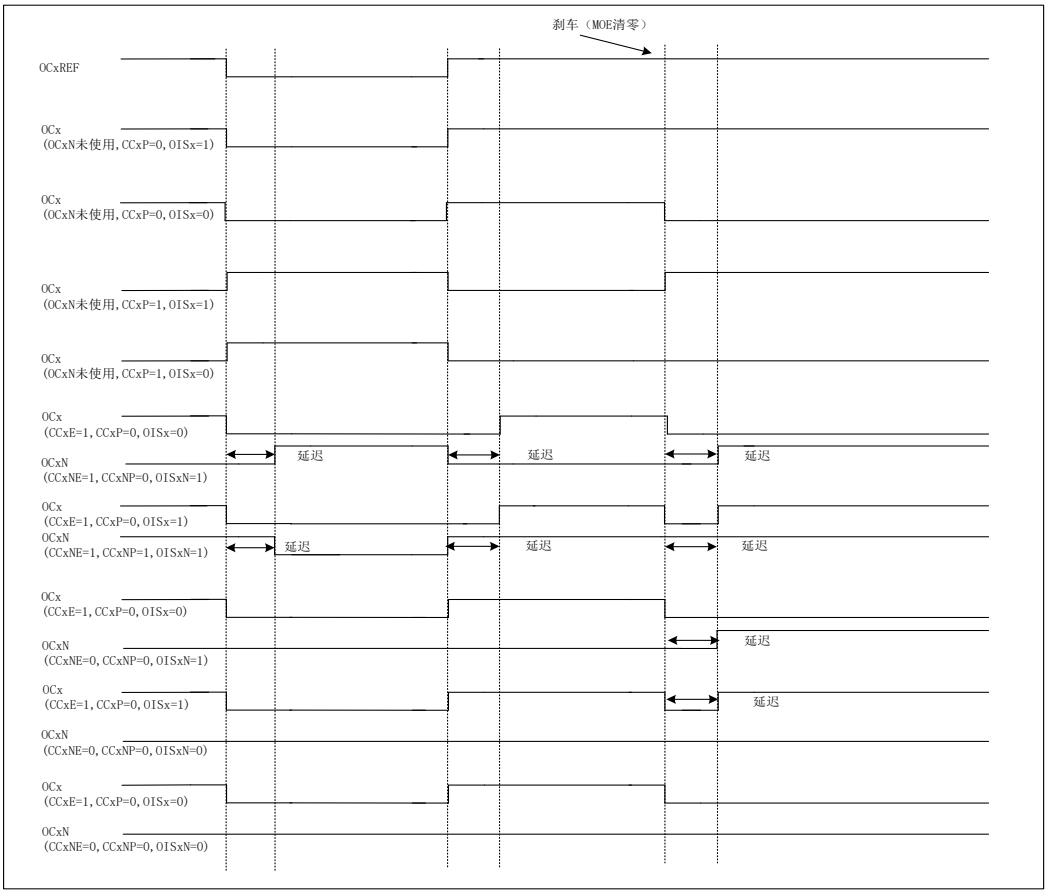
器释放使能输出，否则使能输出始终为高。

- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’ 1’ 时，则产生一个中断。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’ 1’ ；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。 除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性)。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位为 0，写成非 0 值后在复位之前都不能再修改它。下图显示响应刹车的输出实例。

图 12-30 响应刹车的输出



12.5.7.8. 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为’1’，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。

该功能只能用于输出比较和 PWM 模式，而不能用于强制模式。 例如，

OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

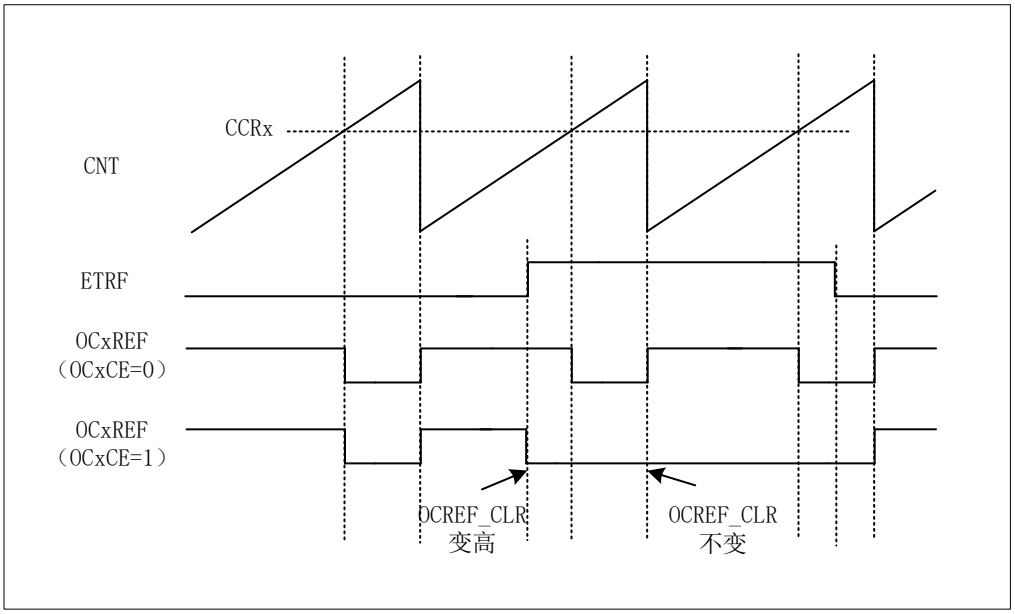
1）外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。

2）必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。

3）外部触发极性(ETP)和外部触发滤波器(ETF)可以根据需要配置。

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作。在这个例子中，定时器 TIMx 被置于 PWM 模式。

图 12-31 清除 TIMx 的 OCxREF



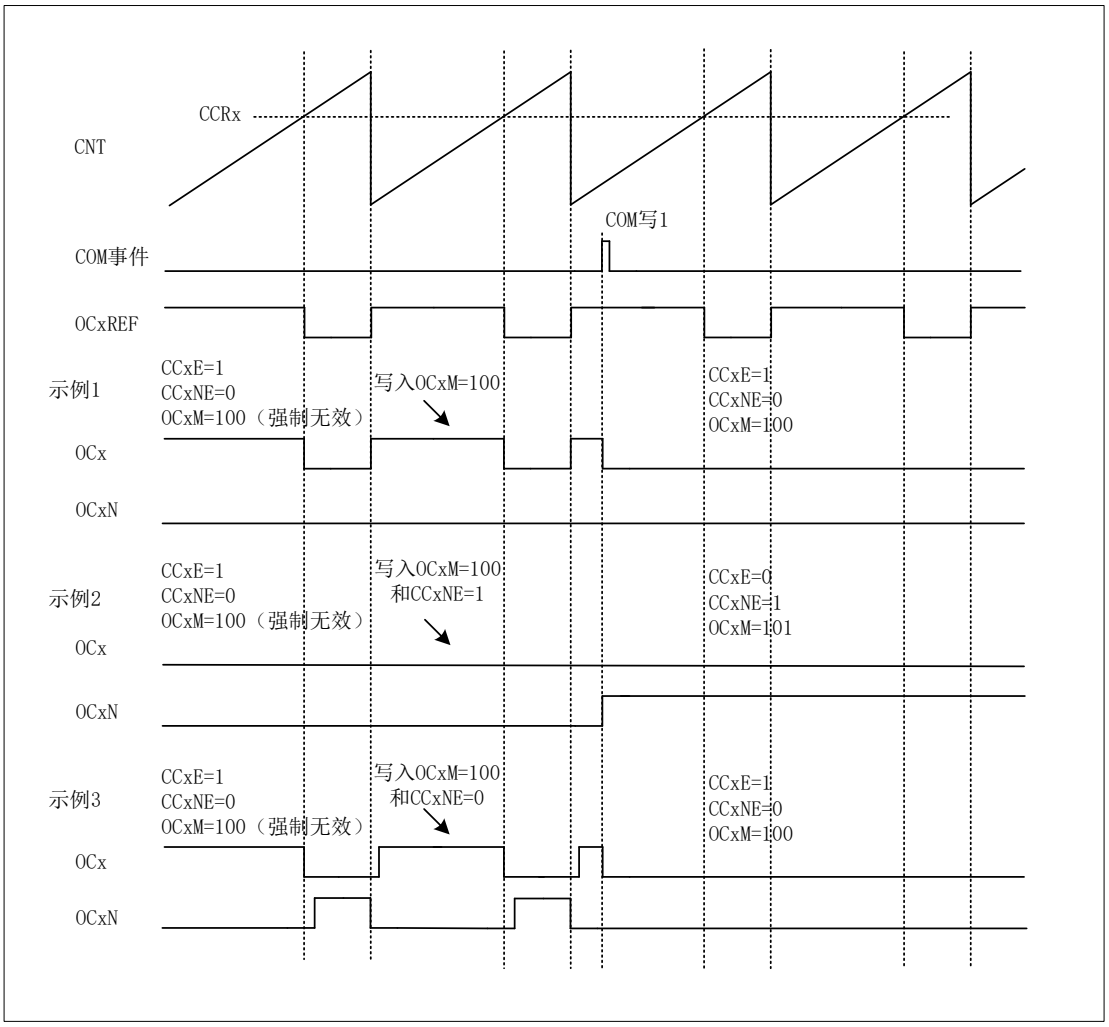
12.5.7.9. 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。

COM 可以通过设置 TIMx_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。 当发生 COM 事件时会设置一个标志位(TIMx_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

下图显示当发生 COM 事件时，三种不同配置下 OCx 和 OCxN 输出。

图 12-32 产生六步 PWM，使用 COM 的例子(OSSR=1)



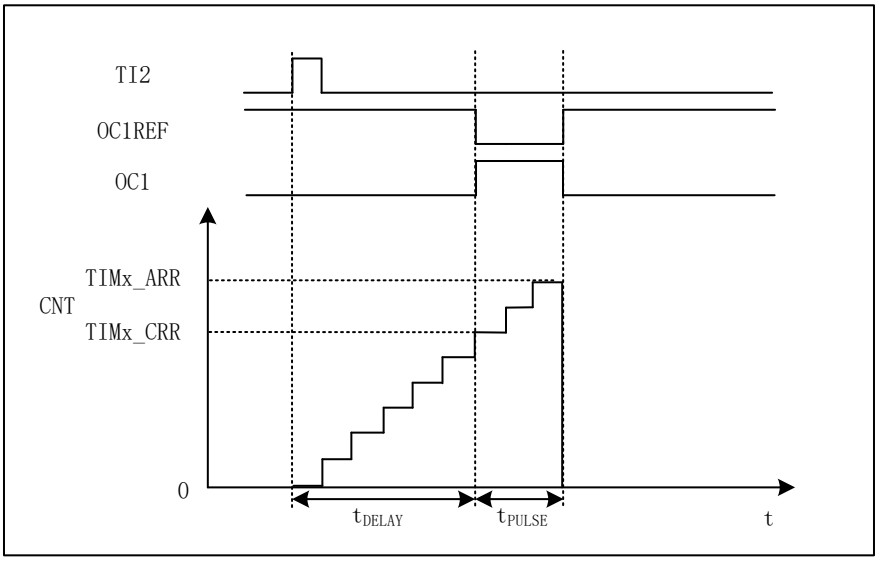
12.5.7.10. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 12-33 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

12.5.7.11. 与霍尔传感器的接口

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。 异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下面给出了此特性用于连接霍尔传感器的例子。

使用高级控制定时器产生 PWM 信号驱动马达时，可以用另一个通用定时器作为“接口定时器”来连接霍尔传感器，见图“霍尔传感器接口的实例”，3 个定时器输入脚(CC1、CC2、CC3)通过一个异或门连接到 TI1 输入通道(通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器从新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以(通过触发一个 COM 事件)用于改变高级定时器各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。因此“接口定时器”通道必须编程为在一个指定的延时(输出比较或 PWM 模式)之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器。 举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。

1）置 TIMx_CR2 寄存器的 TI1S 位为’ 1’ ，配置三个定时器输入逻辑或到 TI1 输入，

2）时基编程：置 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。

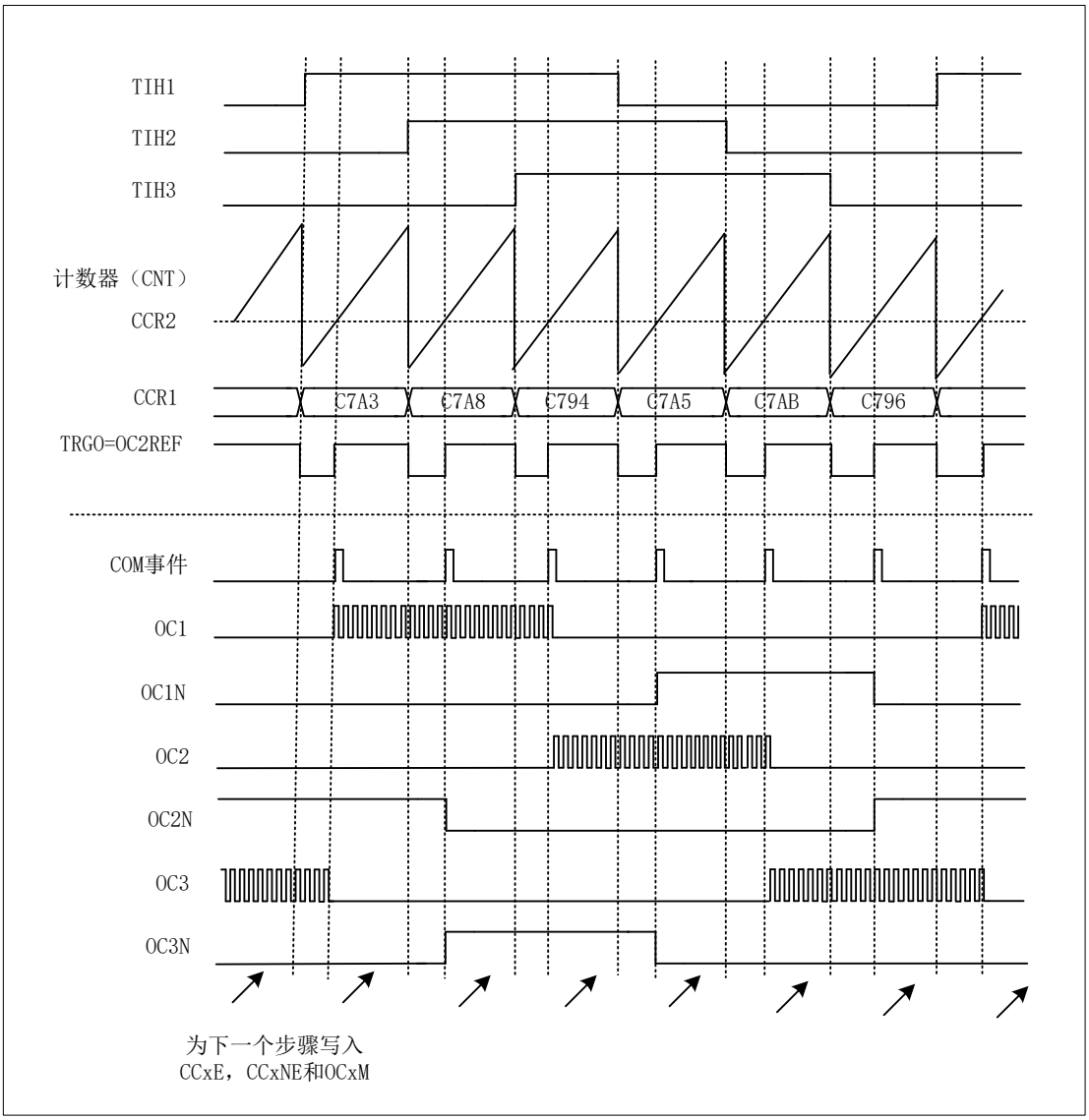
3）设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。

4）设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。

5）选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的(TIMx_CR2 寄存器中 CCPC=1)，同时触发输入控制 COM 事件(TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入下一步的 PWM 控制位(CCxE、OCxM)，这可以在处理 OC2REF 上升沿的中断子程序里实现。 下图显示了这个实例。

图 12-34 霍尔传感器接口的实例



12.5.8. 定时器从模式

TIMx 定时器能够在从模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

12.5.8.1. 复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR， TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

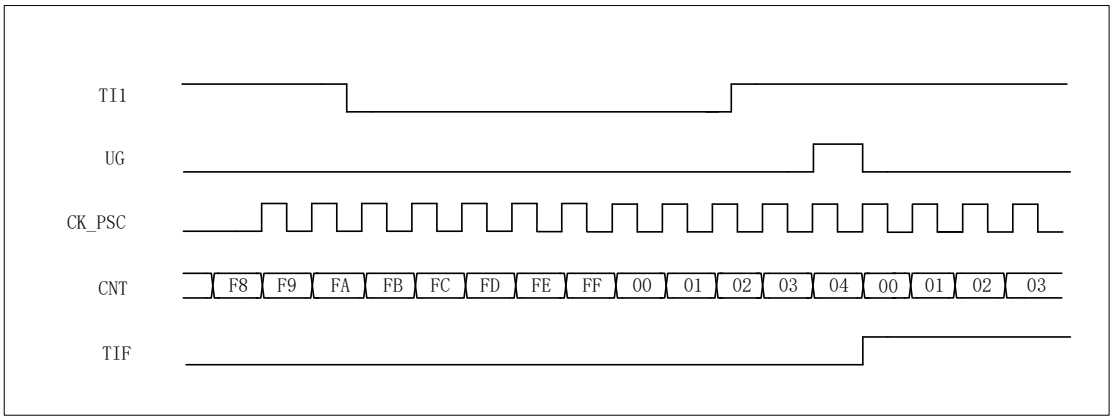
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

2）置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。 下图显示当自动重装载寄存器 TIMx_ARR=0xFF 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 12-35 复位模式下的控制电路



12.5.8.2. 门控模式

按照选中的输入端电平使能计数器。

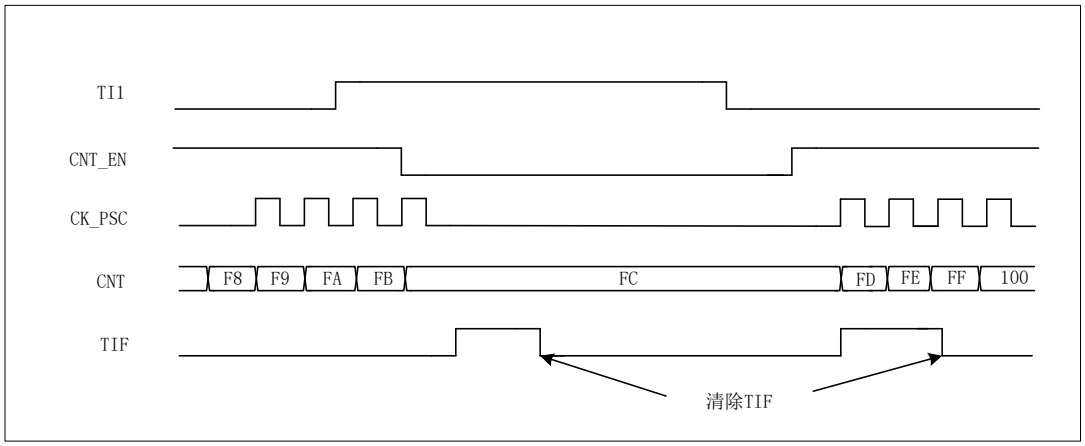
在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图 12-36 门控模式下的控制电路



12.5.8.3. 触发模式

输入端上选中的事件使能计数器。

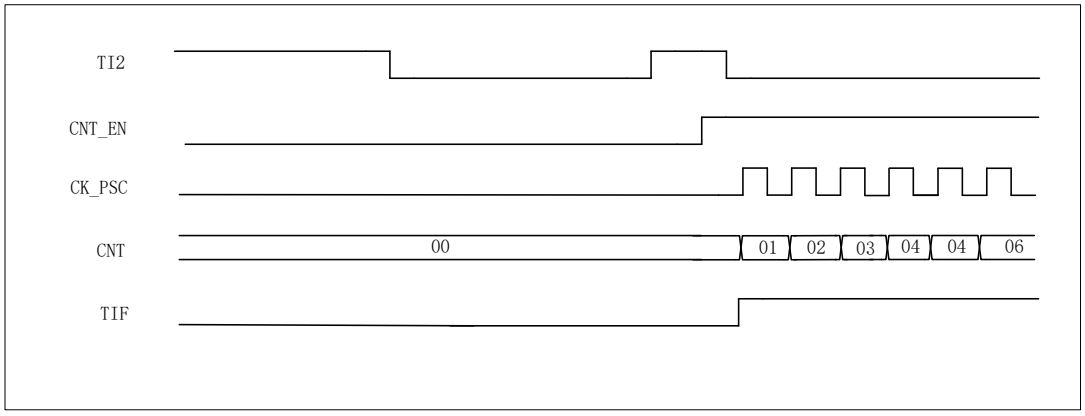
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 12-37 触发模式下的控制电路



12.5.8.4. 外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1）通过 TIMx_SMCR 寄存器配置外部触发输入电路： ⼀ ETF=0000：没有滤波 ⼀ ETPS=00：不用预分频器 ⼀ ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。

2）按如下配置通道 1，检测 TI 的上升沿： ⼀ IC1F=0000：没有滤波 ⼀ 触发操作中不使用捕获预分频器，不

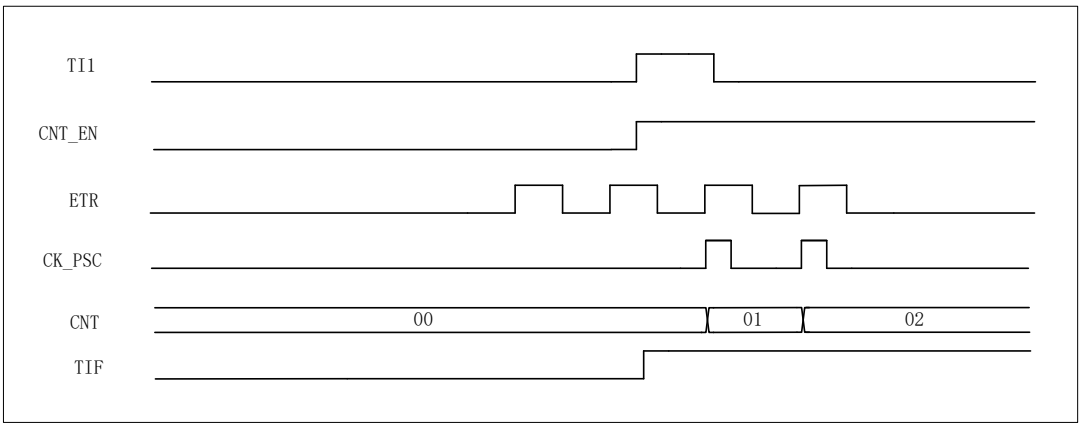
需要配置 — 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源 — 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

3）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

图 12-38 外部时钟模式 2+触发模式控制电路



12.5.9. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

非 Burst DMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMx 会给 DMA 发送请求,DMA 控制器开始根据配置搬移数据，等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

12.5.10. 定时器调试模式

定时器在调试时依然在运行。

12.6. TIM1/8/20 寄存器描述

12.6.1. 寄存器列表

TIM1 寄存器基地址：0x40012C00

TIM8 寄存器基地址：0x40013400

TIM20 寄存器基地址：0x40015000

表 12-9 高级控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	TIMx_SMCR	TIMx 从模式控制寄存器
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	TIMx_CCMR2	TIMx 捕获/比较模式寄存器 2
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	TIMx_RCR	TIMx 重复计数寄存器
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	TIMx_CCR2	TIMx 捕获比较寄存器 2
0x3C	TIMx_CCR3	TIMx 捕获比较寄存器 3
0x40	TIMx_CCR4	TIMx 捕获比较寄存器 4
0x44	TIMx_BDTR	TIMx 刹车和死区控制寄存器
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x54	TIMx_CCMR3	TIMx 捕获/比较模式寄存器 3
0x58	TIMx_CCR5	TIMx 捕获比较寄存器 5
0x5C	TIMx_CCR6	TIMx 捕获比较寄存器 6
0x60	TIMx_AF1	TIMx 复用功能选择寄存器 1

0x64	TIMx_AF2	TIMx 复用功能选择寄存器 2
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

12.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	OC_SEL	RW	0x0	输出比较选择 0：比较输出不经过寄存器锁存一拍对外输出 1：比较输出经过寄存器锁存一拍对外输出 该位对客户不开放
13:10	BKF	RW	0x0	刹车滤波器（Break filter) 这几位定义了用于刹车输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟(tDTS)与定时器时钟（CK_INT）的分频比例 00:tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器

6:5	CMS	RW	0x0	计数模式 00:边沿对齐模式，计数器根据方向位（DIR）向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0x0	方向控制位 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求
1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器 1：使能计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

12.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述

31:24	RSV	-	-	保留，始终读为 0
23:20	MMS2	RW	0x0	主模式选择 2 (Master mode selection 2) 这些位可选择将发送到 ADC 以实现同步的信息（TRGO2）。这些位的组合如下： 0000： 复位——TIMx_EGR 寄存器中的 UG 位用作触发输出（TRGO2）。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO2 上的信号相比实际复位会有延迟。 0001： 使能——计数器使能信号 CNT_EN 用作触发输出（TRGO2）。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。在门控模式下，计数器使能信号是 CEN 控制位和的触发输入信号的逻辑与产生。当计数器使能信号由触发输入控制时，TRGO2 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的 说明）。 0010： 更新——选择更新事件作为触发输出（TRGO2）。例如，主定时器可用作从定时器的预分频器。 0011： 比较脉冲——只要发生捕获或比较匹配事件时，CC1IF 标志置 1（即使已为高），触发输出（TRGO2）都会发送一个正脉冲。 0100： 比较——OC1REF 信号用作触发输出（TRGO2） 0101： 比较——OC2REF 信号用作触发输出（TRGO2） 0110： 比较——OC3REF 信号用作触发输出（TRGO2） 0111： 比较——OC4REF 信号用作触发输出（TRGO2） 1000： 比较——OC5REF 信号用作触发输出（TRGO2） 1001： 比较——OC6REF 信号用作触发输出（TRGO2） 1010： 比较脉冲——OC4REF 上升沿或下降沿时， TRGO2 上生成脉冲 1011： 比较脉冲——OC6REF 上升沿或下降沿时， TRGO2 上生成脉冲 1100： 比较脉冲——OC4REF 或 OC6REF 上升沿时， TRGO2 上生成脉冲 1101： 比较脉冲——OC4REF 上升沿或 OC6REF 下降沿时， TRGO2 上生成脉冲 1110： 比较脉冲——OC5REF 或 OC6REF 上升沿时， TRGO2 上生成脉冲 1111： 比较脉冲——OC5REF 上升沿或 OC6REF 下降沿时， TRGO2 上生成脉冲 注： 必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触 发信号时，不得实时更改从定时器或 ADC 的时钟。
19	RSV	-	-	保留，始终读为 0
18	OIS6	RW	0x0	输出空闲状态 6(OC6 输出)。参见 OIS1 位。
17	RSV	-	-	保留，始终读为 0
16	OIS5	RW	0x0	输出空闲状态 5(OC5 输出)。参见 OIS1 位。
15	OIS4N	RW	0x0	输出空闲状态 4(OC4N 输出)。参见 OIS1N 位。
14	OIS4	RW	0x0	输出空闲状态 4(OC4 输出)。参见 OIS1 位。
13	OIS3N	RW	0x0	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位。
12	OIS3	RW	0x0	输出空闲状态 3(OC3 输出)。参见 OIS1 位。
11	OIS2N	RW	0x0	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
10	OIS2	RW	0x0	输出空闲状态 2(OC2 输出)。参见 OIS1 位。

9	OIS1N	RW	0x0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0 1：当 MOE=0 时，死区后 OC1N=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
8	OIS1	RW	0x0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
7	TI1S	RW	0x0	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入
6:4	MMS	RW	0x0	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0x0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求 1：当发生更新事件时，送出 CCx 的 DMA 请求
2	CCUS	RW	0x0	捕获/比较控制更新选择 (Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0
0	CCPC	RW	0x0	捕获/比较预装载控制位 (Capture/compare preloaded control) 0：CCxE，CCxNE 和 OCxM 位不是预装载的 1：CCxE，CCxNE 和 OCxM 位是预装载的 设置该位后，它们只在设置了 COM 位后被更新

12.6.4. 从模式控制寄存器(TIMx_SMCR: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留，始终读为 0
21:20	TS[4:3]	RW	0x0	
19:16	RSV	-	-	保留，始终读为 0
15	ETP	RW	0x0	外部触发极性（External trigger polarity） 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效 1：ETR 被反相，低电平或下降沿有效
14	ECE	RW	0x0	外部时钟使能位（External clock enable） 该位启用外部时钟模式 2 0：禁止外部时钟模式 2 1：使能外部时钟模式 2 计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是’111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS	RW	0x0	外部触发预分频（External trigger prescaler） 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频 01：ETRP 频率除以 2 10：ETRP 频率除以 4 11：ETRP 频率除以 8

11:8	ETF	RW	0x0	外部触发滤波（External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
7	MSM	RW	0x0	主/从模式（Master/slave mode) 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	RW	0x0	触发选择（Trigger selection) 这 5 位选择用于同步计数器的触发输入 00000：内部触发 0(ITR0) 00100：TI1 的边沿检测器(TI1F_ED) 00001：内部触发 1(ITR1) 00101：滤波后的定时器输入 1(TI1FP1) 00010：内部触发 2(ITR2) 00110：滤波后的定时器输入 2(TI2FP2) 00011：内部触发 3(ITR3) 00111：外部触发输入(ETRF) 01000: 内部触发 4(ITR4) 01001: 内部触发 5(ITR5) 01010: 内部触发 6(ITR6) 01011: 内部触发 7(ITR7) 01100: 内部触发 8(ITR8) 01101: 内部触发 9(ITR9) 01110: 内部触发 10(ITR10) 其它：保留 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。 ITRx 请参看 TIMx 输入映射章节

3	OCCS	RW	0x0	OCREF 清零选择（OCREF clear selection） 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 COMP1 或 COMP2 输出，具体取决于 TIM1_OR1.OCREF_CLR 1： OCREF_CLR_INT 连接到 ETRF
2:0	SMS	RW	0x0	从模式选择（Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表 12-10 TIM1 内部触发连接

从定时器	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	TIM15	保留	TIM3	TIM17 OC1

12.6.5. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	TDE	RW	0x0	允许触发 DMA 请求（Trigger DMA request enable) 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13	COMDE	RW	0x0	允许 COM 的 DMA 请求（COM DMA request enable) 0：禁止 COM 的 DMA 请求； 1：允许 COM 的 DMA 请求。
12	CC4DE	RW	0x0	允许捕获/比较 4 的 DMA 请求（Capture/Compare 4 DMA request enable) 0：禁止捕获/比较 4 的 DMA 请求； 1：允许捕获/比较 4 的 DMA 请求。

11	CC3DE	RW	0x0	允许捕获/比较 3 的 DMA 请求（Capture/Compare 3 DMA request enable) 0：禁止捕获/比较 3 的 DMA 请求； 1：允许捕获/比较 3 的 DMA 请求。
10	CC2DE	RW	0x0	允许捕获/比较 2 的 DMA 请求（Capture/Compare 2 DMA request enable) 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。
9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable) 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	BIE	RW	0x0	允许刹车中断（Break interrupt enable) 0：禁止刹车中断； 1：允许刹车中断。
6	TIE	RW	0x0	触发中断使能（Trigger interrupt enable) 0：禁止触发中断； 1：使能触发中断。
5	COMIE	RW	0x0	允许 COM 中断（COM interrupt enable) 0：禁止 COM 中断； 1：允许 COM 中断。
4	CC4IE	RW	0x0	允许捕获/比较 4 中断（Capture/Compare 4 interrupt enable) 0：禁止捕获/比较 4 中断； 1：允许捕获/比较 4 中断。
3	CC3IE	RW	0x0	允许捕获/比较 3 中断（Capture/Compare 3 interrupt enable) 0：禁止捕获/比较 3 中断； 1：允许捕获/比较 3 中断。
2	CC2IE	RW	0x0	允许捕获/比较 2 中断（Capture/Compare 2 interrupt enable) 0：禁止捕获/比较 2 中断； 1：允许捕获/比较 2 中断。
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable) 0：禁止更新中断； 1：允许更新中断。

12.6.6. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留，始终读为 0

17	CC6IF	W0C	0x0	捕获/比较 6 中断标记（Capture/Compare 6 interrupt flag）参考 CC1IF 描述。
16	CC5IF	W0C	0x0	捕获/比较 5 中断标记（Capture/Compare 5 interrupt flag）参考 CC1IF 描述。
15:13	RSV	-	-	保留，始终读为 0
12	CC4OF	W0C	0x0	捕获/比较 4 重复捕获标记（Capture/Compare 4 overcapture flag）参见 CC1OF 描述。
11	CC3OF	W0C	0x0	捕获/比较 3 重复捕获标记（Capture/Compare 3 overcapture flag）参见 CC1OF 描述。
10	CC2OF	W0C	0x0	捕获/比较 2 重复捕获标记（Capture/Compare 2 overcapture flag）参见 CC1OF 描述。
9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为‘1’。
8	RSV	-	-	位 8 保留，始终读为 0
7	BIF	W0C	0x0	刹车中断标记（Break interrupt flag） 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，则该位可由软件清‘0’。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
6	TIF	W0C	0x0	触发器中断标记（Trigger interrupt flag） 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置‘1’。它由软件清‘0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	W0C	0x0	COM 中断标记（COM interrupt flag） 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置‘1’。它由软件清‘0’。 0：无 COM 事件产生； 1：COM 中断等待响应。
4	CC4IF	W0C	0x0	捕获/比较 4 中断标记（Capture/Compare 4 interrupt flag）参考 CC1IF 描述。
3	CC3IF	W0C	0x0	捕获/比较 3 中断标记（Capture/Compare 3 interrupt flag）参考 CC1IF 描述。
2	CC2IF	W0C	0x0	捕获/比较 2 中断标记（Capture/Compare 2 interrupt flag）参考 CC1IF 描述。

1	CC1IF	W0C	0x0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’ 。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’ ，它由软件清’0’或通过读 TIMx_CCR1 清’0’ 。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置’1’ 。它由软件清’0’ 。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’ ： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

12.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	位 31:8 保留，始终读为 0
7	BG	WO	0x0	产生刹车事件 (Break generation) 该位由软件置’1’ ，用于产生一个刹车事件，由硬件自动清’0’ 。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
6	TG	WO	0x0	产生触发事件 (Trigger generation) 该位由软件置’1’ ，用于产生一个触发事件，由硬件自动清’0’ 。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	WO	0x0	捕获/比较事件，产生控制更新 (Capture/Compare control update generation) 该位由软件置’1’ ，由硬件自动清’0’ 。 0：无动作； 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
4	CC4G	WO	0x0	产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 CC1G 描述。

3	CC3G	WO	0x0	产生捕获/比较 3 事件（Capture/Compare 3 generation）参考 CC1G 描述。
2	CC2G	WO	0x0	产生捕获/比较 2 事件（Capture/Compare 2 generation）参考 CC1G 描述。
1	CC1G	WO	0x0	产生捕获/比较 1 事件（Capture/Compare 1 generation） 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

12.6.8. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	OC2CE	RW	0x0	输出比较 2 清 0 使能（Output Compare 2 clear enable）
14:12	OC2M	RW		输出比较 2 模式（Output Compare 2 mode）
11	OC2PE	RW	0x0	输出比较 2 预装载使能（Output Compare 2 preload enable）
10	OC2FE	RW	0x0	输出比较 2 快速使能（Output Compare 2 fast enable）

9:8	CC2S	RW	0x0	捕获/比较 2 选择。(Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0x0	输出比较 1 清’0’使能 (Output Compare 1 clear enable) 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
6:4	OC1M	RW	0x0	输出比较 1 模式 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。

2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:12	IC2F	RW	0x0	输入捕获 2 滤波器（Input capture 2 filter）
11:10	IC2PSC	RW	0x0	输入/捕获 2 预分频器（Input capture 2 prescaler）
9:8	CC2S	RW	0x0	捕获/比较 2 选择（Capture/Compare 2 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。

7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

12.6.9. 捕获/比较模式寄存器 2(TIMx_CCMR2: 1Ch)

输出比较模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	OC4CE	RW	0x0	输出比较 4 清 0 使能（Output compare 4 clear enable）
14:12	OC4M	RW	0x0	输出比较 4 模式（Output compare 4 mode）
11	OC4PE	RW	0x0	输出比较 4 预装载使能（Output compare 4 preload enable）
10	OC4FE	RW	0x0	输出比较 4 快速使能（Output compare 4 fast enable）

9:8	CC4S	RW	0x0	捕获/比较 4 选择（Capture/Compare 4 selection） 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0x0	输出比较 3 清 0 使能（Output compare 3 clear enable）
6:4	OC3M	RW	0x0	输出比较 3 模式（Output compare 3 mode）
3	OC3PE	RW	0x0	输出比较 3 预装载使能（Output compare 3 preload enable）
2	OC3FE	RW	0x0	输出比较 3 快速使能（Output compare 3 fast enable）
1:0	CC3S	RW	0x0	捕获/比较 3 选择（Capture/Compare 3 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:12	IC4F	RW	0x0	输入捕获 4 滤波器（Input capture 4 filter）
11:10	IC4PSC	RW	0x0	输入/捕获 4 预分频器（Input capture 4 prescaler）
9:8	CC4S	RW	0x0	捕获/比较 4 选择（Capture/Compare 4 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F	RW	0x0	输入捕获 3 滤波器（Input capture 3 filter）
3:2	IC3PSC	RW	0x0	输入/捕获 3 预分频器（Input capture 3 prescaler）

1:0	CC3S	RW	0x0	捕获/比较 3 选择（Capture/compare 3 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

12.6.10. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留，始终读为 0
21	CC6P	RW	0x0	输入/捕获 6 输出极性（Capture/Compare 6 output polarity）参考 CC5P 的描述。
20	CC6E	RW	0x0	输入/捕获 6 输出使能（Capture/Compare 6 output enable）参考 CC5E 的描述。
19:18	RSV	-	-	保留，始终读为 0
17	CC5P	RW	0x0	输入/捕获 5 输出极性（Capture/Compare 5 output polarity）参考 CC5P 的描述。
16	CC5E	RW	0x0	输入/捕获 5 输出使能（Capture/Compare 5 output enable）参考 CC5E 的描述。
15	CC4NP	RW	0x0	输入/捕获 4 互补输出极性（Capture/Compare 4 complementary output polarity）参考 CC1NP 的描述。
14	CC4NE	RW	0x0	输入/捕获 4 互补输出使能（Capture/Compare 4 complementary output enable）参考 CC1NE 的描述。
13	CC4P	RW	0x0	输入/捕获 4 输出极性（Capture/Compare 4 output polarity）参考 CC1P 的描述。
12	CC4E	RW	0x0	输入/捕获 4 输出使能（Capture/Compare 4 output enable）参考 CC1E 的描述。
11	CC3NP	RW	0x0	输入/捕获 3 互补输出极性（Capture/Compare 3 complementary output polarity）参考 CC1NP 的描述。
10	CC3NE	RW	0x0	输入/捕获 3 互补输出使能（Capture/Compare 3 complementary output enable）参考 CC1NE 的描述。
9	CC3P	RW	0x0	输入/捕获 3 输出极性（Capture/Compare 3 output polarity）参考 CC1P 的描述。
8	CC3E	RW	0x0	输入/捕获 3 输出使能（Capture/Compare 3 output enable）参考 CC1E 的描述。
7	CC2NP	RW	0x0	输入/捕获 2 互补输出极性（Capture/Compare 2 complementary output polarity）参考 CC1NP 的描述。
6	CC2NE	RW	0x0	输入/捕获 2 互补输出使能（Capture/Compare 2 complementary output enable）参考 CC1NE 的描述。
5	CC2P	RW	0x0	输入/捕获 2 输出极性（Capture/Compare 2 output polarity）参考 CC1P 的描述。

4	CC2E	RW	0x0	输入/捕获 2 输出使能（Capture/Compare 2 output enable）参考 CC1E 的描述。
3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效； 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别（TIMx_BDTR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00 (通道配置为输出）则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效； 1：OC1 低电平有效。 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表 12-11 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止（与定时器断开） OCx=0，OCx_EN=0	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0
		0	0	1	输出禁止（与定时器断开） OCx=0，OCx_EN=0	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP OCx_EN=1	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
		1	0	1	关闭状态(输出使能且为无效电平) OCx=CCxP，OCx_EN=1	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP， OCxN_EN=1
		1	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP， OCxN_EN=1	关闭状态(输出使能且为无效电平) OCxN=CCxNP，OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
0	0	X	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	0		0	1	输出禁止（与定时器断开） 异步地：OCx=CCxP，OCx_EN=0，OCxN=CCxNP，OCxN_EN=0， 若时钟存在：经过一个死区时间后，OCx=OISx，OCxN=OISx， 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		1	0		
	0		1	1		
	1		0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	1		0	1	关闭状态（输出使能且为无效电平） 异步地：OCx=CCxP，OCx_EN=1，OCxN=CCxNP，OCxN_EN=1， 若时钟存在：经过一个死区时间后 OCx=OISx，OCxN=OISxN， 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		1	0		
	1		1	1		

12.6.11. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value）

12.6.12. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

12.6.13. 自动重装载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

12.6.14. 重复计数寄存器(TIMx_RCR: 30h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:8	RSV	-	-	位 15:8 保留，始终读为 0
7:0	REP	RW	0x0	重复计数器的值（Repetition counter value） 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；

12.6.15. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

12.6.16. 捕获/比较寄存器 2(TIMx_CCR2: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR2	RW	0x0	捕获/比较通道 2 的值（Capture/Compare 2 value） 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

12.6.17. 捕获/比较寄存器 3(TIMx_CCR3: 3Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR3	RW	0x0	捕获/比较通道 3 的值（Capture/Compare 3 value） 若 CC3 通道配置为输出： CCR3 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。

12.6.18. 捕获/比较寄存器 4(TIMx_CCR4: 40h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0

15:0	CCR4	RW	0x0	捕获/比较通道 4 的值（Capture/Compare 4 value） 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值(预装载值)。 如果在 TIMx_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。

12.6.19. 刹车和死区寄存器(TIMx_BDTR: 44h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	MOE	RW	0x0	主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。
14	AOE	RW	0x0	自动输出使能（Automatic output enable） 0：MOE 只能被软件置’1’； 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。
13	BKP	RW	0x0	刹车输入极性（Break polarity） 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0x0	刹车功能使能（Break enable） 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0x0	运行模式下“关闭状态”选择（Off-state selection for Run mode） 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。

10	OSSI	RW	0x0	空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	0x0	锁定设置（Lock configuration）该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位，TIMx_AF1 所有位，TIMx_AF2 所有位，TIMx_CR1 的 BKF 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	0x0	死区发生器设置（Dead-time generator setup） 这些位定义了插入互补输出之间的死区持续时间。 假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg，Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg，Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg，Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位。

12.6.20. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0

12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIM2_CR1 - 如果 DBL=7，DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0
4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

12.6.21. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses） 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

12.6.22. 捕获/比较模式寄存器 3(TIMx_CCMR3: 54h)

通道 5 和通道 6 只能配置为输出。

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留，始终读为 0

24	OC6M[3]			
23:17	RSV	-	-	保留，始终读为 0
16	OC5M[3]			
15	OC6CE	RW	0x0	输出比较 2 清 0 使能（Output Compare 2 clear enable）
14:12	OC6M	RW	0x0	输出比较 2 模式（Output Compare 2 mode）
11	OC6PE	RW	0x0	输出比较 2 预装载使能（Output Compare 2 preload enable）
10	OC6FE	RW	0x0	输出比较 2 快速使能（Output Compare 2 fast enable）
9:8	RSV	-	-	保留，始终读为 0
7	OC5CE	RW	0x0	输出比较 1 清’0’使能（Output Compare 1 clear enable） 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
6:4	OC5M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC5PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。

2	OC5FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	RSV	-	-	保留，始终读为 0

12.6.23. 捕获/比较寄存器 5(TIMx_CCR5: 58h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR5	RW	0x0	捕获/比较通道 5 的值（Capture/Compare 1 value） 若 CC5 通道配置为输出： CCR5 包含了装入当前捕获/比较 5 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC5PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 5 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC5 端口上产生输出信号。 若 CC5 通道配置为输入： CCR5 包含了由上一次输入捕获 5 事件(IC5)传输的计数器值。

12.6.24. 捕获/比较寄存器 6(TIMx_CCR6: 5Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR6	RW	0x0	捕获/比较通道 6 的值（Capture/Compare 6 value） 若 CC6 通道配置为输出： CCR6 包含了装入当前捕获/比较 6 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC6PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 6 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC6 端口上产生输出信号。 若 CC6 通道配置为输入： CCR6 包含了由上一次输入捕获 6 事件(IC6)传输的计数器值。

12.6.25. 复用功能选择寄存器 1(TIMx_AF1: 60h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留，始终读为 0
17:16	ETRSEL[3:2]			
15:14	ETRSEL	RW	0x0	ETR 输入源选择 0000：GPIO 0001：COMP1 0010：COMP2 0011：AWD 0100：etr4 0101：etr5 0110：etr6 0111：etr7 1000：etr8 1001：etr9 1010：etr10 1011：etr11 1100：etr12 1101：etr13 1110：etr14 1111：etr15 输入源请参看 TIMx 输入映射章节
13	BKCMP4P	RW	0x0	比较器 4 输入极性控制 0：输入高电平有效 1：输入低电平有效
12	BKCMP3P	RW	0x0	比较器 3 输入极性控制 0：输入高电平有效 1：输入低电平有效
11	BKCMP2P	RW	0x0	比较器 2 输入极性控制 0：输入高电平有效 1：输入低电平有效
10	BKCMP1P	RW	0x0	比较器 1 输入极性控制 0：输入高电平有效 1：输入低电平有效
9	BKINP	RW	0x0	刹车输入极性控制 0：输入高电平有效 1：输入低电平有效
8:5	RSV	-	-	保留，始终为 0。
4	BKCMP4E	RW	0x0	比较器 4 输入使能控制 0：禁止 1：使能

3	BKCMP3E	RW	0x0	比较器 3 输入使能控制 0：禁止 1：使能
2	BKCMP2E	RW	0x0	比较器 2 输入使能控制 0：禁止 1：使能
1	BKCMP1E	RW	0x0	比较器 1 输入使能控制 0：禁止 1：使能
0	BKINE	RW	0x0	刹车输入使能控制 0：禁止 1：使能

12.6.26. 复用功能选择寄存器 2(TIMx_AF2: 64h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留，始终读为 0
18:16	OCRSEL[2:0]	RW	0x0	ocref_clr 源选择 这些位选择 ocref_clr 输入源。 000：tim_ocref_clr0 001：tim_ocref_clr1 010：tim_ocref_clr2 011：tim_ocref_clr3 100：tim_ocref_clr4 101：tim_ocref_clr5 110：tim_ocref_clr6 111：tim_ocref_clr7 输入源请参看 TIMx 输入映射章节
15:0	RSV	-	-	保留，始终为 0。

12.6.27. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留，始终为 0。

27:24	TI4SEL	RW	0x0	TI4 输入选择 0000：TIMx_CH4 0001：tim_ti4_in1 0010：tim_ti4_in2 0011：tim_ti4_in3 … 1111：tim_ti4_in15 输入源请参看 TIMx 输入映射章节
23:20	-	-	-	保留，始终为 0。
19:16	TI3SEL	RW	0x0	TI3 输入选择 0000：TIMx_CH3 0001：tim_ti3_in1 0010：tim_ti3_in2 0011：tim_ti3_in3 … 1111：tim_ti3_in15 输入源请参看 TIMx 输入映射章节
15:12	RSV	-	-	保留，始终为 0。
11:8	TI2SEL	RW	0x0	TI2 输入选择 0000：TIM_CH2 0001：COMP2 0010：tim_ti2_in2 0011：tim_ti2_in3 … 1111：tim_ti2_in15 输入源请参看 TIMx 输入映射章节
7:4	-	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：TIM_CH1 0001：COMP1 0010：tim_ti1_in2 0011：tim_ti1_in3 … 1111：tim_ti1_in15 输入源请参看 TIMx 输入映射章节

12.6.28. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0

6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5	COMBE	RW	0x0	COM 事件的 DMA 请求类型 0：Single； 1：Burst；
4	CC4BE	RW	0x0	捕获/比较 4 事件的 DMA 请求类型 0：Single； 1：Burst；
3	CC3BE	RW	0x0	捕获/比较 3 事件的 DMA 请求类型 0：Single； 1：Burst；
2	CC2BE	RW	0x0	捕获/比较 2 事件的 DMA 请求类型 0：Single； 1：Burst；
1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；
0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

13. 通用定时器（TIM2/3/4/5/23/24）

13.1. 概述

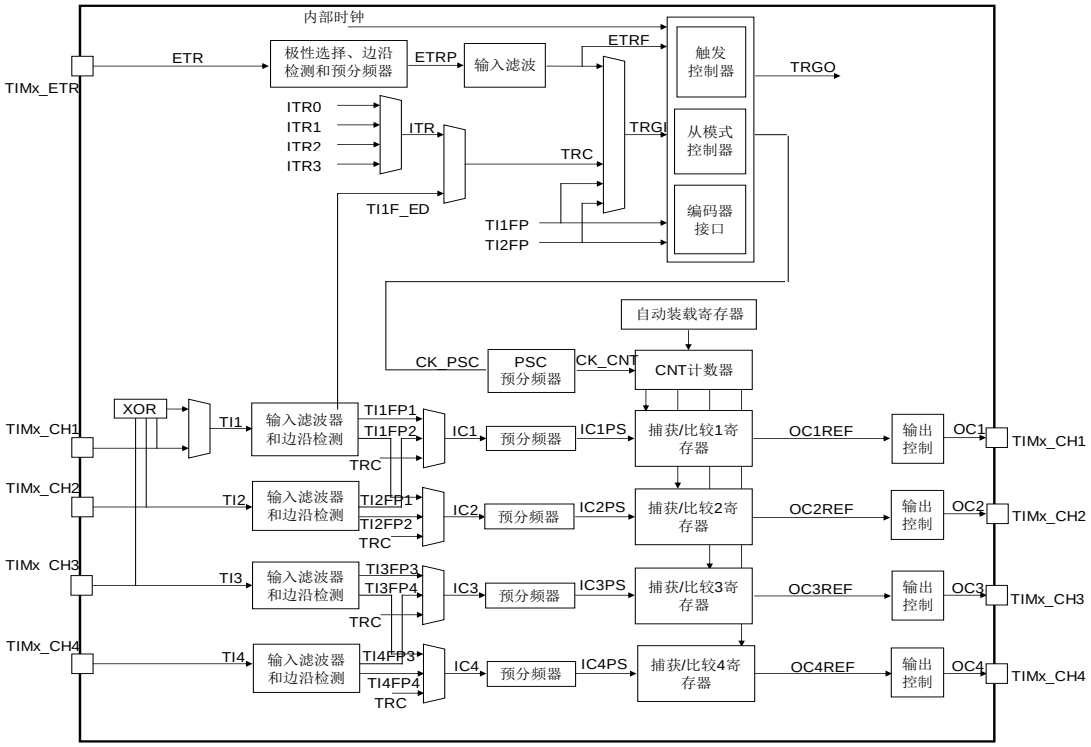
通用定时器（TIM2/3/4/5/23/24）由一个 16/32 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM 等)。高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

13.2. 主要特性

- 16/32（TIM3/4: 16 位，TIM2/5/23/24: 32 位）位向上、向下、向上/下自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘或中间对齐模式)
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
- 支持针对定位的增量(正交)编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

13.3. 结构框图

图 13-1 通用定时器框图



13.4. TIMx 输入映射

表 13-1 TIMx 内部触发输入（ITRx）

	TIM2 源	TIM3 源	TIM4 源
ITR0	tim1_trgo	tim1_trgo	tim1_trgo
ITR1	-	tim2_trgo	tim2_trgo
ITR2	tim3_trgo	-	tim3_trgo
ITR3	tim4_trgo	tim4_trgo	-
ITR4	-	-	-
ITR5	tim8_trgo	tim8_trgo	tim8_trgo
ITR6	tim15_trgo	tim15_trgo	tim15_trgo
ITR7	tim16_oc1	tim16_oc1	tim16_oc1
ITR8	tim17_oc1	tim17_oc1	tim17_oc1
保留	-	-	-

表 13-2 TIMx ETR 输入

ETRSEL[3:0]	TIM2 源	TIM3 源	TIM4 源
0000	tim2_etr(IO)	tim3_etr(IO)	tim4_etr(IO)
0001	comp1_out	comp1_out	comp1_out

0010	comp2_out	comp2_out	comp2_out
0011	comp3_out	comp3_out	comp3_out
0100	comp4_out	comp4_out	comp4_out
0101	tim3_etr(IO)	tim2_etr(IO)	tim3_etr(IO)
0110	tim4_etr(IO)	tim4_etr(IO)	-
0111	-	adc2_awd	-
保留	-	-	-

表 13-3 TIMx 通道 1 输入

TI1SEL[3:0]	TIM2 源	TIM3 源	TIM4 源
0000	tim2_ch1	tim3_ch1	tim4_ch1
0001	comp1_out	comp1_out	comp1_out
0010	comp2_out	comp2_out	comp2_out
0011	comp3_out	comp3_out	comp3_out
0100	comp4_out	comp4_out	comp4_out
保留	-	-	-

表 13-4 TIMx 通道 2 输入

TI2SEL[3:0]	TIM2 源	TIM3 源	TIM4 源
0000	tim2_ch2	tim3_ch2	tim4_ch2
0001	comp1_out	comp1_out	comp1_out
0010	comp2_out	comp2_out	comp2_out
0011	comp3_out	comp3_out	comp3_out
0100	comp4_out	comp4_out	comp4_out
保留	-	-	-

表 13-5 TIMx 通道 3 输入

TI3SEL[3:0]	TIM2 源	TIM3 源	TIM4 源
0000	tim2_ch3	tim3_ch3	tim4_ch3
0001	comp4_out	comp3_out	-
保留	-	-	-

表 13-6 TIM1 通道 4 输入

TI4SEL[3:0]	TIM2 源	TIM3 源	TIM4 源
0000	tim2_ch4	tim3_ch4	tim4_ch4

0001	comp1_out	-	-
0010	comp2_out	-	-
保留	-	-	-

表 13-7 TIM1 OCREF_CLR 输入

OCRSEL[2:0]	TIM2 源	TIM3 源	TIM4 源
0000	comp1_out	comp1_out	comp1_out
0001	comp2_out	comp2_out	comp2_out
0010	comp3_out	comp3_out	comp3_out
0011	comp4_out	comp4_out	comp4_out
保留	-	-	-

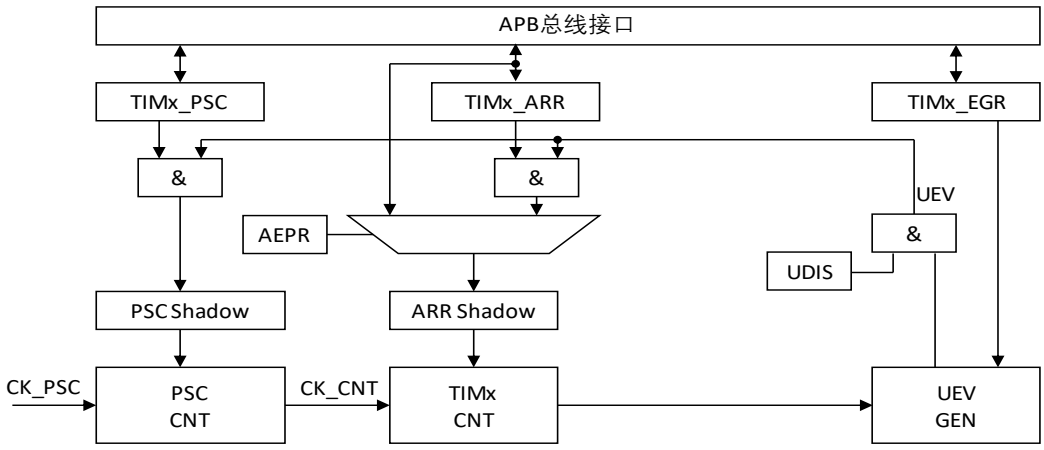
13.5. 功能描述

13.5.1. 计数单元

可编程通用定时器的主要部分是一个 16/32 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)

图 13-2 计数单元的结构



自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

通用定时器支持三种计数模式：

● 向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

● 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

● 中央对齐模式

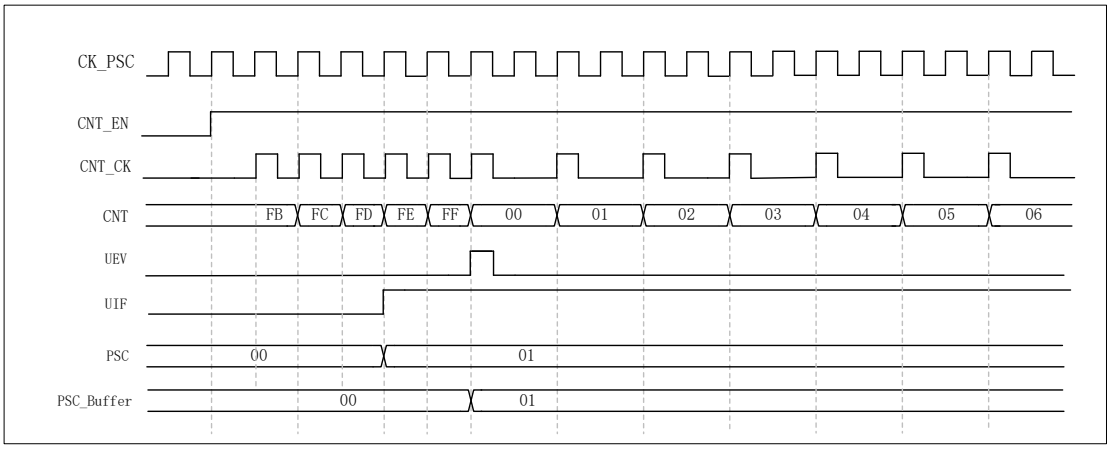
在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。向上计数模式中，定时器模块在计数器计数到自动加载值-1 产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMx_CR1 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。计数方向被硬件自动更新。

13.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 13-3 当预分频器的参数从 1 变到 2 时，计数器的时序图



13.5.3. 时钟源选择

计数器时钟可由下列时钟源提供：

● 内部时钟(CK_INT)

● 外部时钟模式 1：外部输入引脚

● 外部时钟模式 2：外部触发输入 ETR

● 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。

图 13-4 外部时钟模式 1

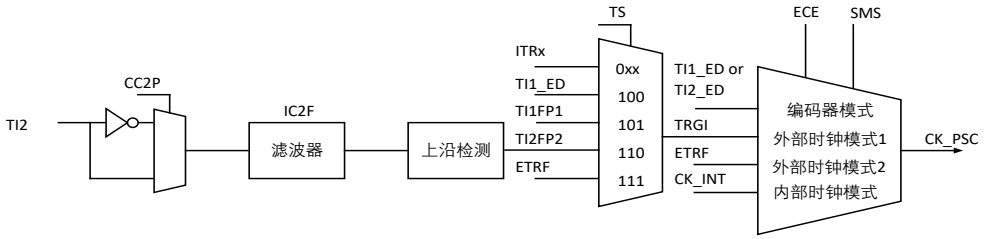
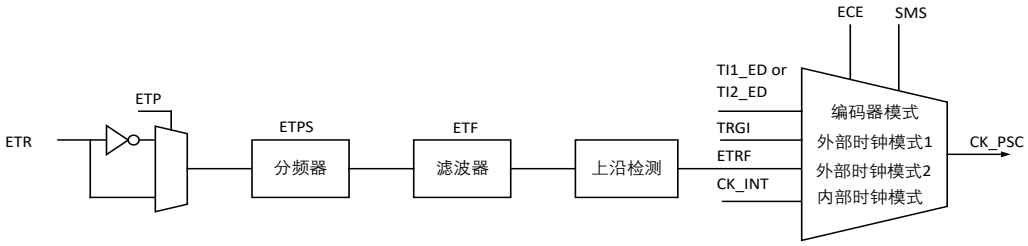


图 13-5 外部时钟模式 2



但如果按功能划分如以上 2 张图示所示，并按照定时器的从模式控制寄存器 TIMx_SMCR 的 ECE 和 SMS 的控制，应该分为以下几种模式：

- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有八个信号源，并由 TIMx_SMCR.TS 寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 1（ITR1）
  - TS=010，内部触发 2（ITR2）
  - TS=011，内部触发 3（ITR3）
  - TS=100，TI1 的边沿检测器（TI1F_ED）
  - TS=101，滤波后的定时器输入 1（TI1FP1）
  - TS=110，滤波后的定时器输入 2（TI2FP2）
  - TS=111，外部触发输入（ETRF）
- 外部时钟模式 2：SMS=xxx，ECE=1，CK_PSC 来自 ETRF
- 编码器模式
  - 编码器模式 1：SMS=001，ECE=0，CK_PSC 来自 TI1FP1 的上下沿
  - 编码器模式 2：SMS=010，ECE=0，CK_PSC 来自 TI2FP2 的上下沿
  - 编码器模式 3：SMS=011，ECE=0，CK_PSC 来自 TI1FP1 和 TI2FP2 的上下沿

13.5.4. 编码器模式

从“时钟源选择”章节可知，编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。表“计数器方向和编码器信号的关系”列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

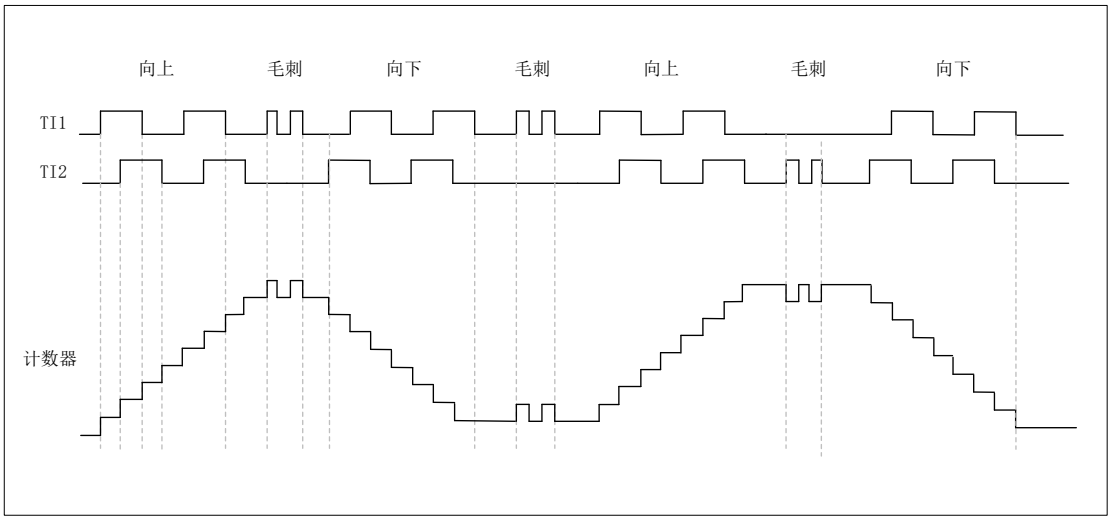
表 13-8 计数器方向和编码器信号的关系

有效边沿	相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1		TI2FP2	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
TI1 和 TI2 都计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。

图 13-6 编码器模式下计数器操作实例

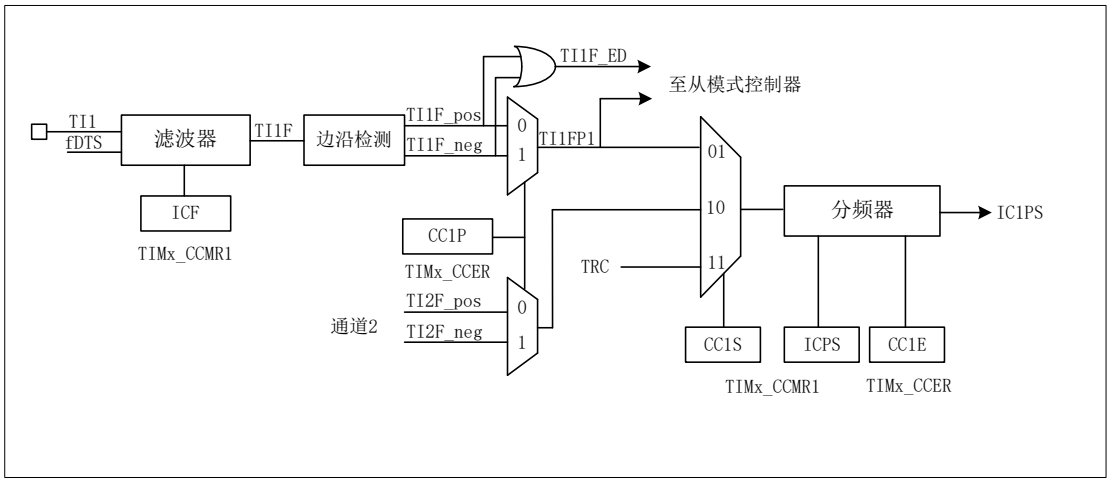


13.5.5. 捕获比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 4 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

图 13-7 捕获/比较通道(如：通道 1 输入部分)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 13-8 捕获/比较通道 1 的主电路

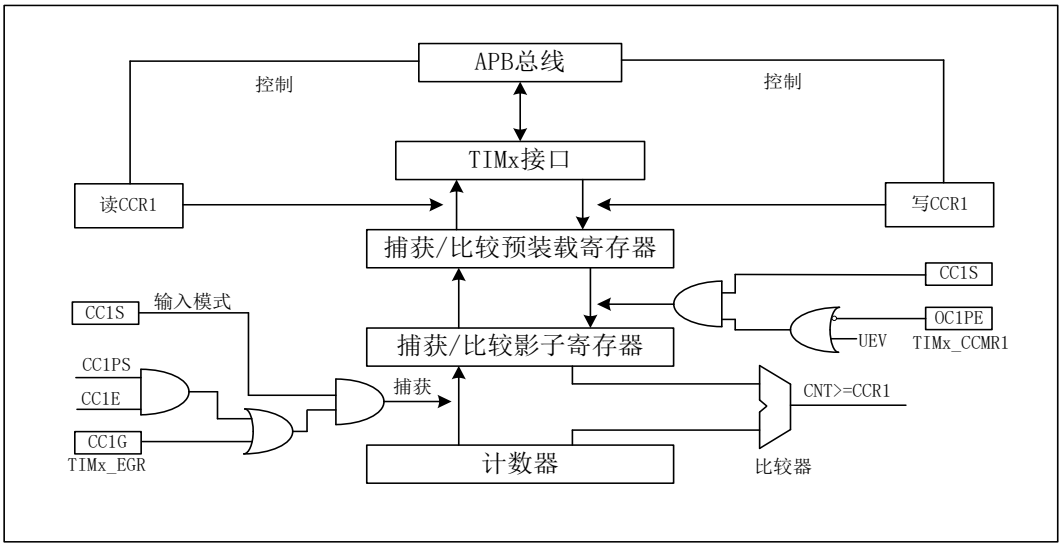
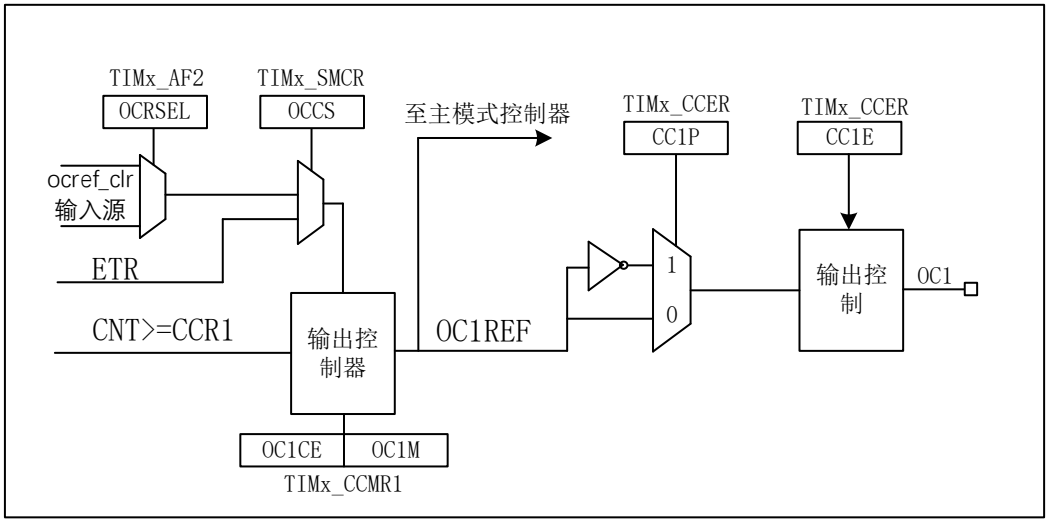


图 13-9 捕获/比较通道的输出部分(通道 1)



13.5.5.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

13.5.5.2. PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

1）两个 ICx 信号被映射至同一个 TIx 输入。

2）这 2 个 ICx 信号为边沿有效，但是极性相反。

3）其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。 例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

4）选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。

5）选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。

6）选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。

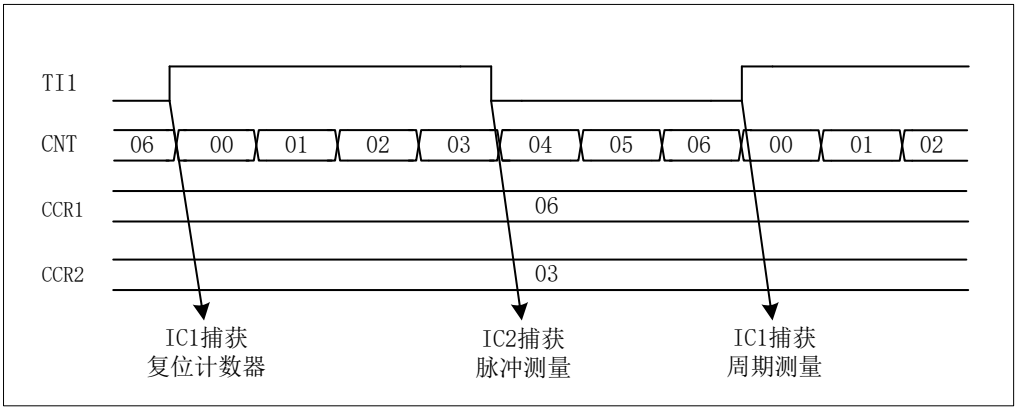
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

8）选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。

9）配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。

10）使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

图 13-10 PWM 输入模式时序



13.5.5.3. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

13.5.5.4. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

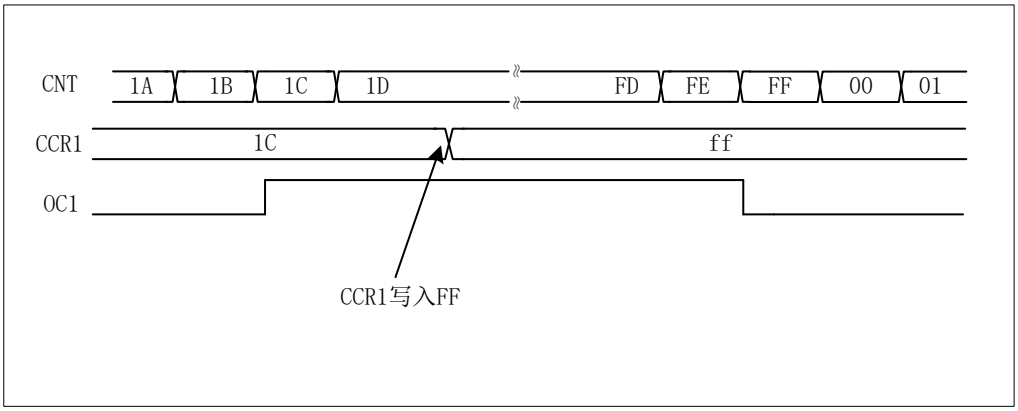
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 13-11 输出比较模式，翻转 OC1



13.5.5.5. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

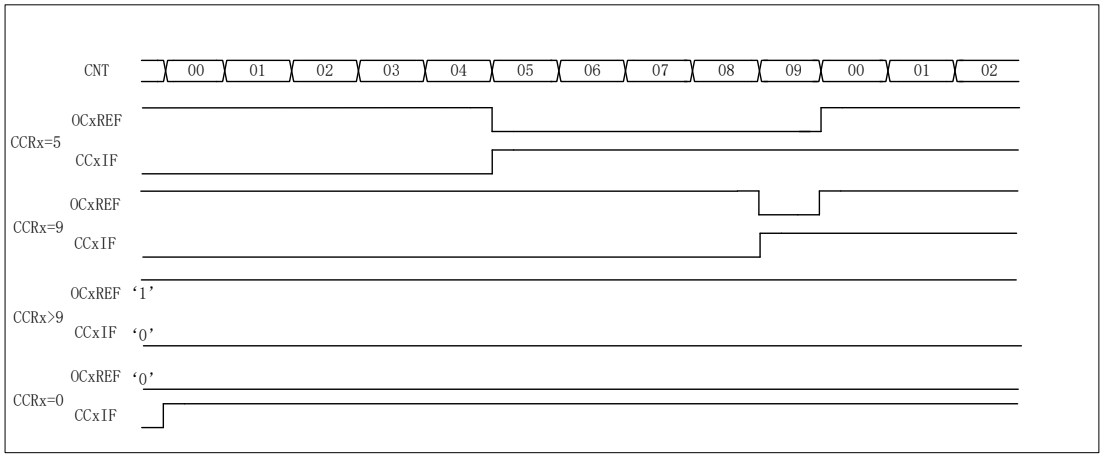
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ PWM 边沿对齐模式

● 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=9 时边沿对齐的 PWM 波形实例。

图 13-12 边沿对齐的 PWM 波形（ARR=9）



● 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

■ PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

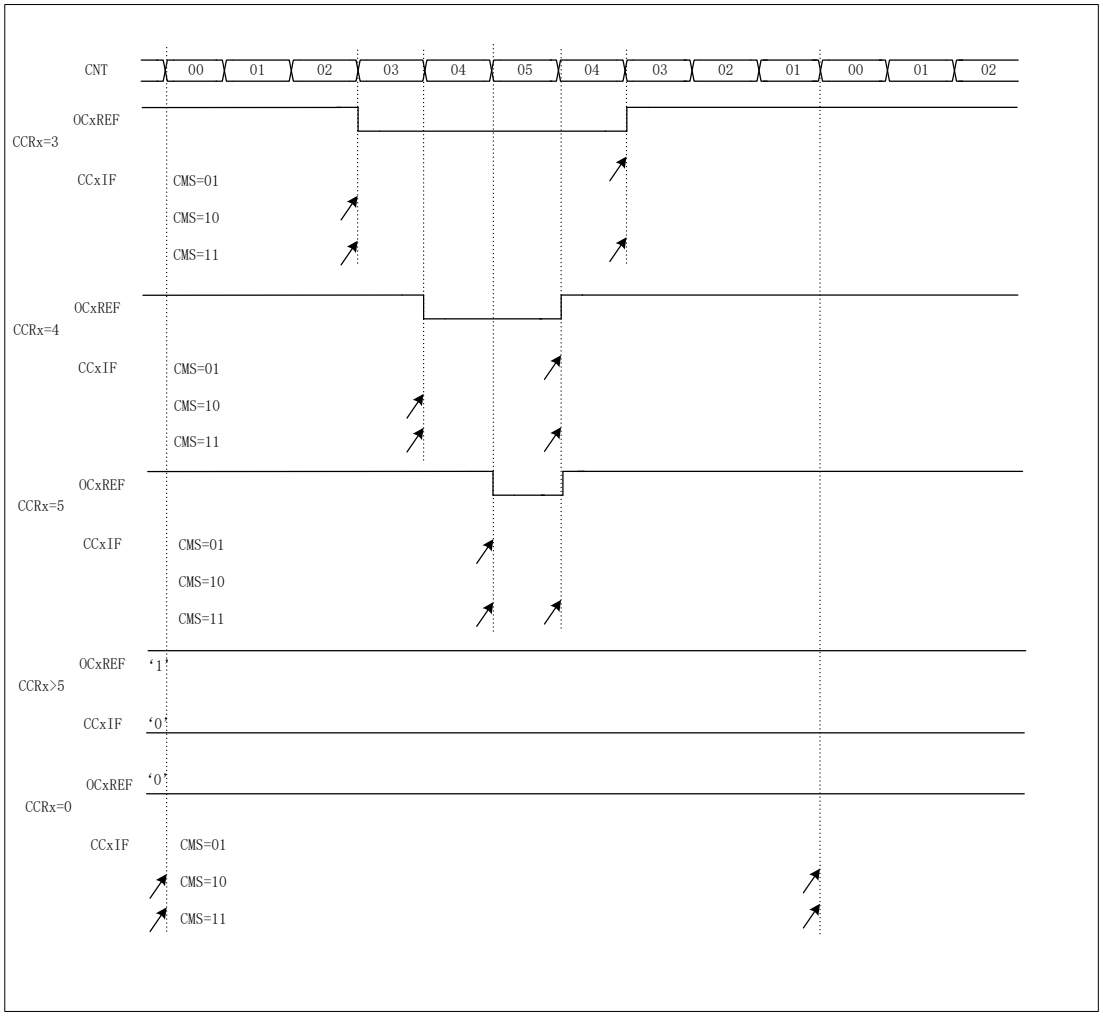
下图给出了一些中央对齐的 PWM 波形的例子

1）TIMx_ARR=5

2）PWM 模式 1

3）TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图 13-13 中央对齐的 PWM 波形（ARR=5）



使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
- 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
- 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

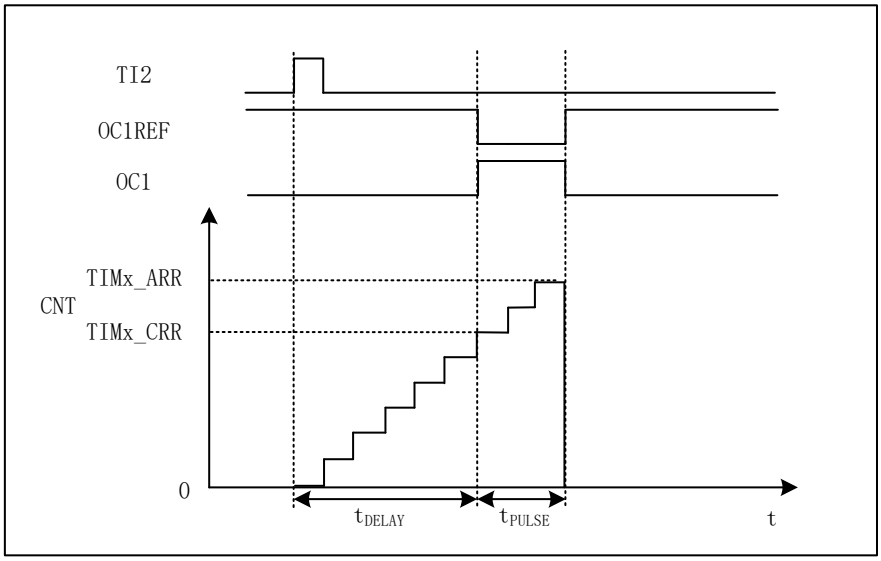
13.5.5.6. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 13-14 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

13.5.5.7. 与霍尔传感器的接口

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。 异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下面给出了此特性用于连接霍尔传感器的例子。

使用高级控制定时器产生 PWM 信号驱动马达时，可以用另一个通用定时器作为“接口定时器”来连接霍尔传感器，见图“霍尔传感器接口的实例”，3 个定时器输入脚(CC1、CC2、CC3)通过一个异或门连接到 TI1 输入通道(通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器从新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以(通过触发一个 COM 事件)用于改变高级定时器各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。因此“接口定时器”通道必须编程为在一个指定的延时(输出比较或 PWM 模式)之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器。 举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。

1）置 TIMx_CR2 寄存器的 TI1S 位为’ 1’ ，配置三个定时器输入逻辑或到 TI1 输入，

2）时基编程：置 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。

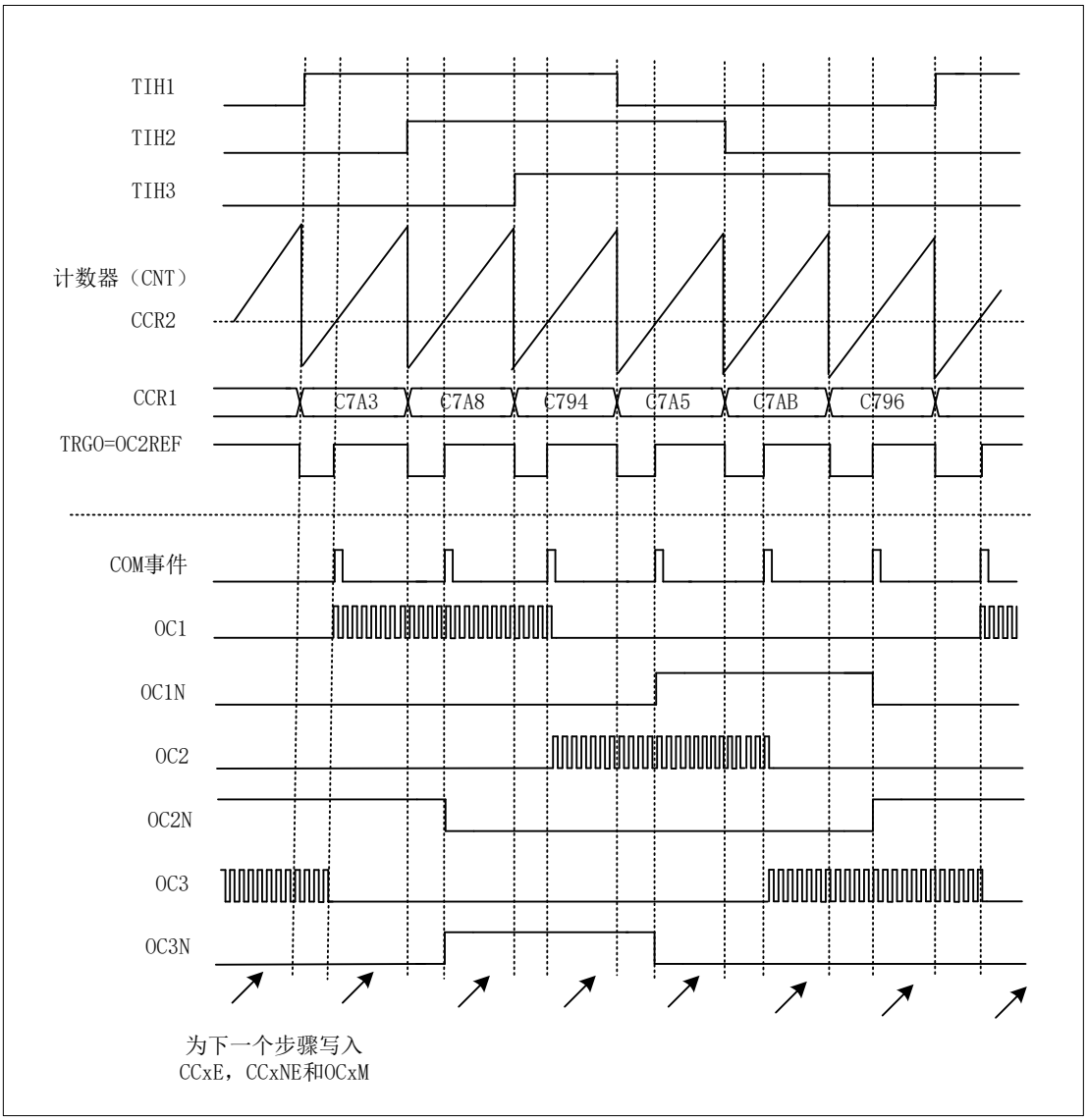
3）设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。

4）设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。

5）选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的(TIMx_CR2 寄存器中 CCPC=1)，同时触发输入控制 COM 事件(TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入下一步的 PWM 控制位(CCxE、OCxM)，这可以在处理 OC2REF 上升沿的中断子程序里实现。 下图显示了这个实例

图 13-15 霍尔传感器接口的实例



13.5.6. 定时器互连

TIMx 定时器能够在从模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

13.5.6.1. 复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR， TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

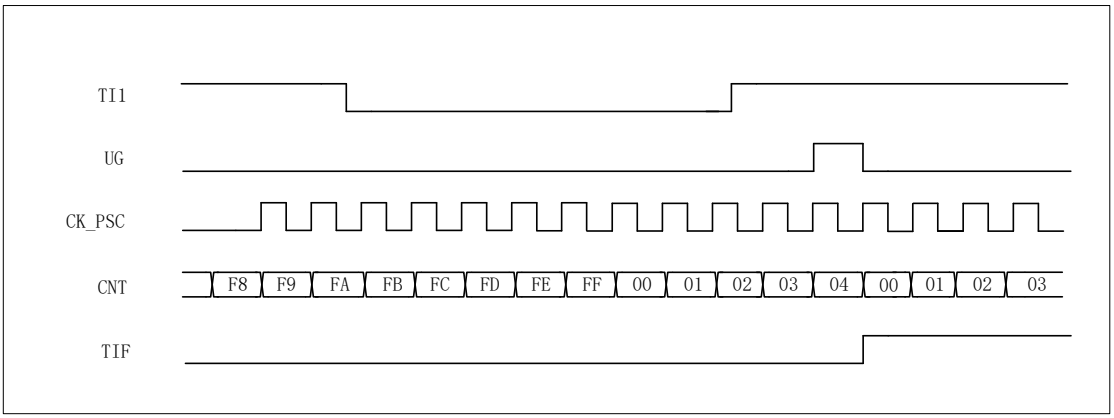
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

2）置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。 下图显示当自动重装载寄存器 TIMx_ARR=0xFF 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 13-16 复位模式下的控制电路



13.5.6.2. 门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

1）配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

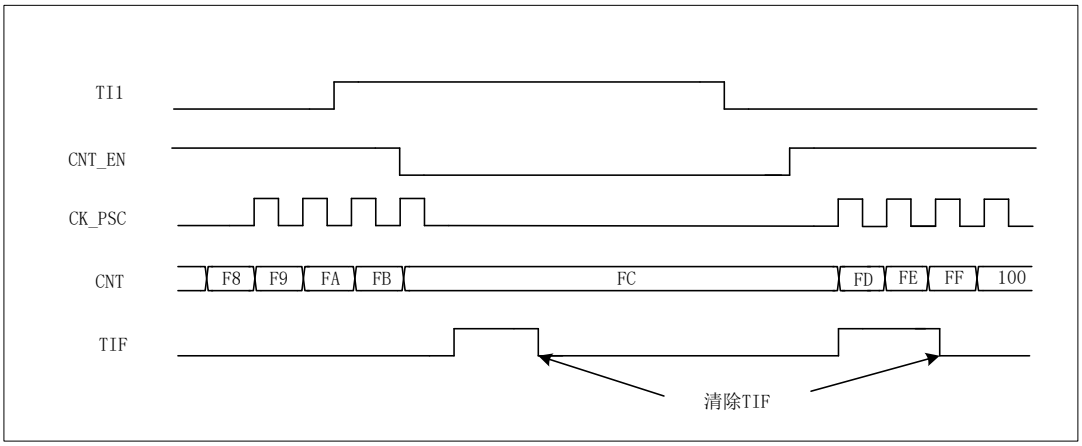
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图 13-17 门控模式下的控制电路



13.5.6.3. 触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

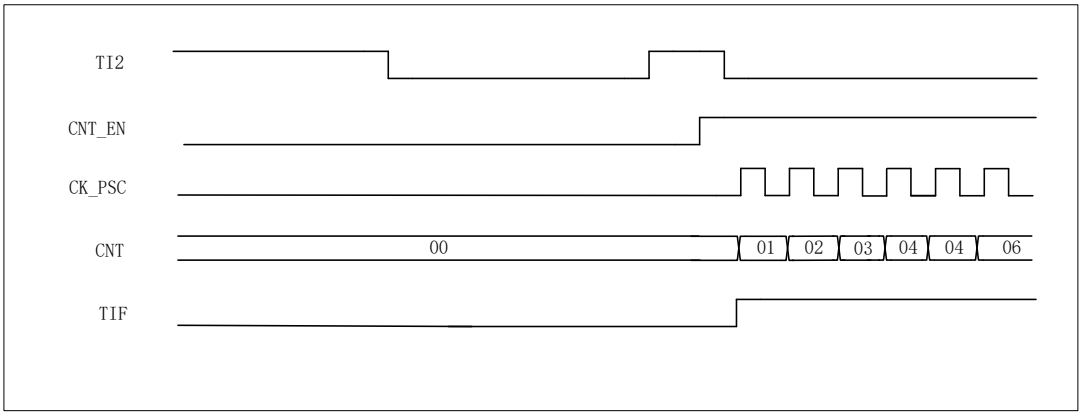
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

2）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 13-18 触发模式下的控制电路



13.5.6.4. 外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1）通过 TIMx_SMCR 寄存器配置外部触发输入电路： ⎼ ETF=0000：没有滤波 ⎼ ETPS=00：不用预分频器 ⎼ ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。

2）按如下配置通道 1，检测 TI 的上升沿： ⎼ IC1F=0000：没有滤波 ⎼ 触发操作中不使用捕获预分频器，不

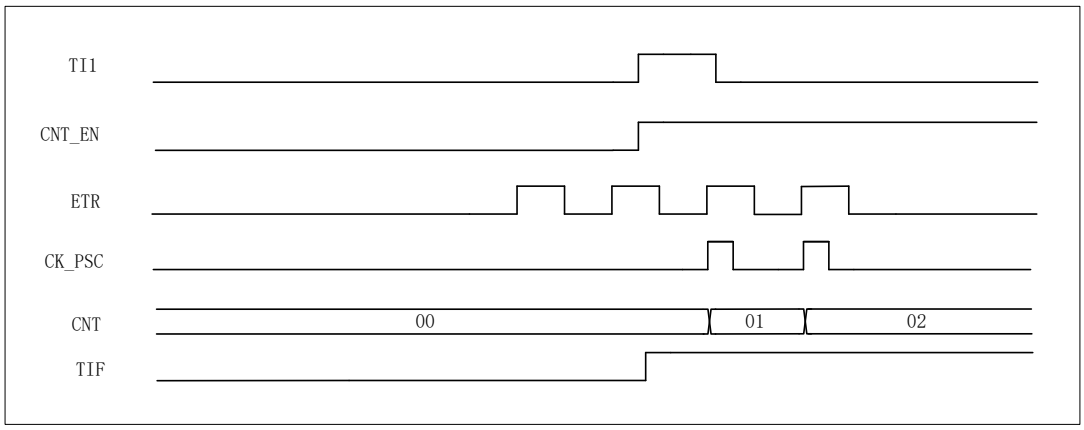
需要配置 — 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源 — 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

3）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

图 13-19 外部时钟模式 2+触发模式控制电路



13.5.7. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

SingleDMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMx 会给 DMA 发送请求,等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

13.5.8. 定时器调试模式

定时器在调试时依然在运行。

13.6. TIM2/3/4/5/23/24 寄存器描述

13.6.1. 寄存器列表

TIM2 寄存器基地址：0x40000000

TIM3 寄存器基地址：0x40000400

TIM4 寄存器基地址：0x40000800

TIM5 寄存器基地址：0x40000C00

TIM23 寄存器基地址：0x4001A400

TIM24 寄存器基地址：0x4001A800

表 13-9 高级控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	TIMx_SMCR	TIMx 从模式控制寄存器
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	TIMx_CCMR2	TIMx 捕获/比较模式寄存器 2
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	-	保留
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	TIMx_CCR2	TIMx 捕获比较寄存器 2
0x3C	TIMx_CCR3	TIMx 捕获比较寄存器 3
0x40	TIMx_CCR4	TIMx 捕获比较寄存器 4
0x44	-	保留
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x60	TIMx_AF1	TIMx 复用功能选择寄存器
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

13.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟(tDTS)与定时器时钟（CK_INT）的分频比例。 00: tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:5	CMS	RW	0x0	计数模式 00:边沿对齐模式，计数器根据方向位（DIR）向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0x0	方向控制位 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。

1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

13.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留。
7	TI1S	RW	0x0	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
6:4	MMS	RW	0x0	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0x0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。

2	CCUS	RW	0x0	捕获/比较控制更新选择（Capture/compare control update selection） 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0
0	CCPC	RW	0x0	捕获/比较预装载控制位（Capture/compare preloaded control） 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的； 设置该位后，它们只在设置了 COM 位后被更新。

13.6.4. 从模式控制寄存器(TIMx_SMCR: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留，始终读为 0
21:20	TS[4:3]	RW	0x0	触发选择信号
19:16	RSV	-	-	保留，始终读为 0
15	ETP	RW	0x0	外部触发极性（External trigger polarity） 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
14	ECE	RW	0x0	外部时钟使能位（External clock enable） 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。 计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是’ 111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS	RW	0x0	外部触发预分频（External trigger prescaler） 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。

11:8	ETF	RW	0x0	外部触发滤波（External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
7	MSM	RW	0x0	主/从模式（Master/slave mode) 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	RW	0x0	触发选择（Trigger selection），和 TS[4:3]组成 5 位的 TS 信号。 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00100：TI1 的边沿检测器(TI1F_ED) 00001：内部触发 1(ITR1) 00101：滤波后的定时器输入 1(TI1FP1) 00010：内部触发 2(ITR2) 00110：滤波后的定时器输入 2(TI2FP2) 00011：内部触发 3(ITR3) 00111：外部触发输入(ETRF) 01000: 内部触发 4(ITR4) 01001: 内部触发 5(ITR5) 01010: 内部触发 6(ITR6) 01011: 内部触发 7(ITR7) 01100: 内部触发 8(ITR8) 01101: 内部触发 9(ITR9) 01110: 内部触发 10(ITR10) 其它：保留 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。 内部触发源 ITRx 详情输入源请参看 TIMx 输入映射章节

3	OCCS	RW	0x0	OCREF 清零选择（OCREF clear selection） 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 COMPx 输出，具体取决于 TIMx_AF2. OCRSEL 1： OCREF_CLR_INT 连接到 ETRF
2:0	SMS	RW	0x0	从模式选择（Slave mode selection） 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

13.6.5. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	TDE	RW	0x0	允许触发 DMA 请求（Trigger DMA request enable） 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13	COMDE	RW	0x0	允许 COM 的 DMA 请求（COM DMA request enable）0：禁止 COM 的 DMA 请求； 1：允许 COM 的 DMA 请求。
12	CC4DE	RW	0x0	允许捕获/比较 4 的 DMA 请求（Capture/Compare 4 DMA request enable） 0：禁止捕获/比较 4 的 DMA 请求； 1：允许捕获/比较 4 的 DMA 请求。
11	CC3DE	RW	0x0	允许捕获/比较 3 的 DMA 请求（Capture/Compare 3 DMA request enable） 0：禁止捕获/比较 3 的 DMA 请求； 1：允许捕获/比较 3 的 DMA 请求。
10	CC2DE	RW	0x0	允许捕获/比较 2 的 DMA 请求（Capture/Compare 2 DMA request enable） 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。

9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable) 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	RSV	-	-	保留，始终读为 0
6	TIE	RW	0x0	触发中断使能（Trigger interrupt enable) 0：禁止触发中断； 1：使能触发中断。
5	COMIE	RW	0x0	允许 COM 中断（COM interrupt enable) 0：禁止 COM 中断； 1：允许 COM 中断。
4	CC4IE	RW	0x0	允许捕获/比较 4 中断（Capture/Compare 4 interrupt enable) 0：禁止捕获/比较 4 中断； 1：允许捕获/比较 4 中断。
3	CC3IE	RW	0x0	允许捕获/比较 3 中断（Capture/Compare 3 interrupt enable) 0：禁止捕获/比较 3 中断； 1：允许捕获/比较 3 中断。
2	CC2IE	RW	0x0	允许捕获/比较 2 中断（Capture/Compare 2 interrupt enable) 0：禁止捕获/比较 2 中断； 1：允许捕获/比较 2 中断。
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable) 0：禁止更新中断； 1：允许更新中断。

13.6.6. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	保留，始终读为 0
12	CC4OF	W0C	0x0	捕获/比较 4 重复捕获标记（Capture/Compare 4 overcapture flag）参见 CC1OF 描述。
11	CC3OF	W0C	0x0	捕获/比较 3 重复捕获标记（Capture/Compare 3 overcapture flag）参见 CC1OF 描述。
10	CC2OF	W0C	0x0	捕获/比较 2 重复捕获标记（Capture/Compare 2 overcapture flag）参见 CC1OF 描述。

9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
8:7	RSV	-	-	保留，始终读为 0
6	TIF	W0C	0x0	触发器中断标记（Trigger interrupt flag） 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	W0C	0x0	COM 中断标记（COM interrupt flag） 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置’1’。它由软件清’0’。 0：无 COM 事件产生； 1：COM 中断等待响应。
4	CC4IF	W0C	0x0	捕获/比较 4 中断标记（Capture/Compare 4 interrupt flag）参考 CC1IF 描述。
3	CC3IF	W0C	0x0	捕获/比较 3 中断标记（Capture/Compare 3 interrupt flag）参考 CC1IF 描述。
2	CC2IF	W0C	0x0	捕获/比较 2 中断标记（Capture/Compare 2 interrupt flag）参考 CC1IF 描述。
1	CC1IF	W0C	0x0	捕获/比较 1 中断标记（Capture/Compare 1 interrupt flag） 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

13.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TG	WO	0x0	产生触发事件（Trigger generation） 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	WO	0x0	捕获/比较事件，产生控制更新（Capture/Compare control update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
4	CC4G	WO	0x0	产生捕获/比较 4 事件（Capture/Compare 4 generation）参考 CC1G 描述。
3	CC3G	WO	0x0	产生捕获/比较 3 事件（Capture/Compare 3 generation）参考 CC1G 描述。
2	CC2G	WO	0x0	产生捕获/比较 2 事件（Capture/Compare 2 generation）参考 CC1G 描述。
1	CC1G	WO	0x0	产生捕获/比较 1 事件（Capture/Compare 1 generation） 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

13.6.8. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义

输出比较模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	OC2CE	RW	0x0	输出比较 2 清 0 使能（Output Compare 2 clear enable）

14:12	OC2M	RW	0x0	输出比较 2 模式（Output Compare 2 mode）
11	OC2PE	RW	0x0	输出比较 2 预装载使能（Output Compare 2 preload enable）
10	OC2FE	RW	0x0	输出比较 2 快速使能（Output Compare 2 fast enable）
9:8	CC2S	RW	0x0	捕获/比较 2 选择。(Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0x0	输出比较 1 清’0’使能（Output Compare 1 clear enable） 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
6:4	OC1M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。

2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
15:12	IC2F	RW	0x0	输入捕获 2 滤波器（Input capture 2 filter）
11:10	IC2PSC	RW	0x0	输入/捕获 2 预分频器（Input capture 2 prescaler）
9:8	CC2S	RW	0x0	捕获/比较 2 选择（Capture/Compare 2 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。

7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

13.6.9. 捕获/比较模式寄存器 2(TIMx_CCMR2: 1Ch)

输出比较模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	OC4CE	RW	0x0	输出比较 4 清 0 使能（Output compare 4 clear enable）
14:12	OC4M	RW	0x0	输出比较 4 模式（Output compare 4 mode）
11	OC4PE	RW	0x0	输出比较 4 预装载使能（Output compare 4 preload enable）
10	OC4FE	RW	0x0	输出比较 4 快速使能（Output compare 4 fast enable）

9:8	CC4S	RW	0x0	捕获/比较 4 选择（Capture/Compare 4 selection） 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0x0	输出比较 3 清 0 使能（Output compare 3 clear enable）
6:4	OC3M	RW	0x0	输出比较 3 模式（Output compare 3 mode）
3	OC3PE	RW	0x0	输出比较 3 预装载使能（Output compare 3 preload enable）
2	OC3FE	RW	0x0	输出比较 3 快速使能（Output compare 3 fast enable）
1:0	CC3S	RW	0x0	捕获/比较 3 选择（Capture/Compare 3 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:12	IC4F	RW	0x0	输入捕获 4 滤波器（Input capture 4 filter）
11:10	IC4PSC	RW	0x0	输入/捕获 4 预分频器（Input capture 4 prescaler）
9:8	CC4S	RW	0x0	捕获/比较 4 选择（Capture/Compare 4 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F	RW	0x0	输入捕获 3 滤波器（Input capture 3 filter）
3:2	IC3PSC	RW	0x0	输入/捕获 3 预分频器（Input capture 3 prescaler）
1:0	CC3S	RW	0x0	捕获/比较 3 选择（Capture/compare 3 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

13.6.10. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	CC4NP	RW	0x0	输入/捕获 4 互补输出极性（Capture/Compare 4 complementary output polarity）参考 CC1NP 的描述。
14	RSV	-	-	保留，始终读为 0
13	CC4P	RW	0x0	输入/捕获 4 输出极性（Capture/Compare 4 output polarity）参考 CC1P 的描述。
12	CC4E	RW	0x0	输入/捕获 4 输出使能（Capture/Compare 4 output enable）参考 CC1E 的描述。
11	CC3NP	RW	0x0	输入/捕获 3 互补输出极性（Capture/Compare 3 complementary output polarity）参考 CC1NP 的描述。
10	CC3NE	RW	0x0	输入/捕获 3 互补输出使能（Capture/Compare 3 complementary output enable）参考 CC1NE 的描述。
9	CC3P	RW	0x0	输入/捕获 3 输出极性（Capture/Compare 3 output polarity）参考 CC1P 的描述。
8	CC3E	RW	0x0	输入/捕获 3 输出使能（Capture/Compare 3 output enable）参考 CC1E 的描述。
7	CC2NP	RW	0x0	输入/捕获 2 互补输出极性（Capture/Compare 2 complementary output polarity）参考 CC1NP 的描述。
6	CC2NE	RW	0x0	输入/捕获 2 互补输出使能（Capture/Compare 2 complementary output enable）参考 CC1NE 的描述。
5	CC2P	RW	0x0	输入/捕获 2 输出极性（Capture/Compare 2 output polarity）参考 CC1P 的描述。
4	CC2E	RW	0x0	输入/捕获 2 输出使能（Capture/Compare 2 output enable）参考 CC1E 的描述。
3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效； 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别（TIMx_BDTR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00 (通道配置为输出）则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。

1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 1：捕获使能

13.6.11. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value），16 位定时器

位域	名称	属性	复位值	描述
31:0	CNT	RW	0x0	计数器的值（Counter value), 32 位定时器

13.6.12. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0

15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

13.6.13. 自动加载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

13.6.14. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0。16 位定时器。
15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

位域	名称	属性	复位值	描述
31:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。 32 位定时器。

13.6.15. 捕获/比较寄存器 2(TIMx_CCR2: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0。16 位定时器

15:0	CCR2	RW	0x0	捕获/比较通道 2 的值（Capture/Compare 2 value） 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

位域	名称	属性	复位值	描述
31:0	CCR2	RW	0x0	捕获/比较通道 2 的值（Capture/Compare 2 value） 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。 32 位定时器。

13.6.16. 捕获/比较寄存器 3(TIMx_CCR3: 3Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0。16 位定时器。
15:0	CCR3	RW	0x0	捕获/比较通道 3 的值（Capture/Compare 3 value） 若 CC3 通道配置为输出： CCR3 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。

位域	名称	属性	复位值	描述
31:0	CCR3	RW	0x0	捕获/比较通道 3 的值（Capture/Compare 3 value） 若 CC3 通道配置为输出： CCR3 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。 32 位定时器。

13.6.17. 捕获/比较寄存器 4(TIMx_CCR4: 40h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0。16 位定时器。

15:0	CCR4	RW	0x0	捕获/比较通道 4 的值（Capture/Compare 4 value） 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值(预装载值)。 如果在 TIMx_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。

位域	名称	属性	复位值	描述
31:0	CCR4	RW	0x0	捕获/比较通道 4 的值（Capture/Compare 4 value） 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值(预装载值)。 如果在 TIMx_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。 32 位定时器。

13.6.18. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0
12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIMx_CR1 - 如果 DBL=7，DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0

4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

13.6.19. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它的最大值取决于 TIMx_DCR 寄存器中定义的 DBL。

13.6.20. 复用功能选择寄存器 1(TIMx_AF1: 60h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留，始终读为 0
17:14	ETRSEL	RW	0000	ETR 输入源选择 0000：ETR0 0001：ETR1 0010：ETR2 0011：ETR3 0100：ETR4 0101：ETR5 0110：ETR6 0111：ETR7 其他：保留 输入源请参看 TIMx 输入映射章节
13:0	RSV	-	-	保留，始终为 0。

13.6.21. 复用功能选择寄存器 2(TIMx_AF2: 64h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留，始终读为 0

18:16	OCRSEL[2:0]	RW	0x0	ocref_clr 源选择 这些位选择 ocref_clr 输入源。 000：tim_ocref_clr0 001：tim_ocref_clr1 010：tim_ocref_clr2 011：tim_ocref_clr3 100：tim_ocref_clr4 101：tim_ocref_clr5 110：tim_ocref_clr6 111：tim_ocref_clr7 输入源请参看 TIMx 输入映射章节
15:0	RSV	-	-	保留，始终为 0。

13.6.22. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留，始终为 0。
27:24	TI4SEL	RW	0x0	TI4 输入选择 0000：tim_ti4_in0 0001：tim_ti4_in1 0010：tim_ti4_in2 0011：tim_ti4_in3 … 1111：tim_ti4_in15 输入源请参看 TIMx 输入映射章节
23:20	RSV	-	-	保留，始终为 0。
19:16	TI3SEL	RW	0x0	TI3 输入选择 0000：tim_ti3_in0 0001：tim_ti3_in1 0010：tim_ti3_in2 0011：tim_ti3_in3 … 1111：tim_ti3_in15 输入源请参看 TIMx 输入映射章节
15:12	RSV	-	-	保留，始终为 0。

11:8	TI2SEL	RW	0x0	TI2 输入选择 0000：tim_ti2_in0 0001：tim_ti2_in1 0010：tim_ti2_in2 0011：tim_ti2_in3 … 1111：tim_ti2_in15 输入源请参看 TIMx 输入映射章节
7:4	RSV	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：tim_ti1_in0 0001：tim_ti1_in1 0010：tim_ti1_in2 0011：tim_ti1_in3 … 1111：tim_ti1_in15 输入源请参看 TIMx 输入映射章节

13.6.23. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5	COMBE	RW	0x0	COM 事件的 DMA 请求类型 0：Single； 1：Burst；
4	CC4BE	RW	0x0	捕获/比较 4 事件的 DMA 请求类型 0：Single； 1：Burst；
3	CC3BE	RW	0x0	捕获/比较 3 事件的 DMA 请求类型 0：Single； 1：Burst；
2	CC2BE	RW	0x0	捕获/比较 2 事件的 DMA 请求类型 0：Single； 1：Burst；
1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；

0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

14. 基本定时器（TIM6/7/21/22）

14.1. 概述

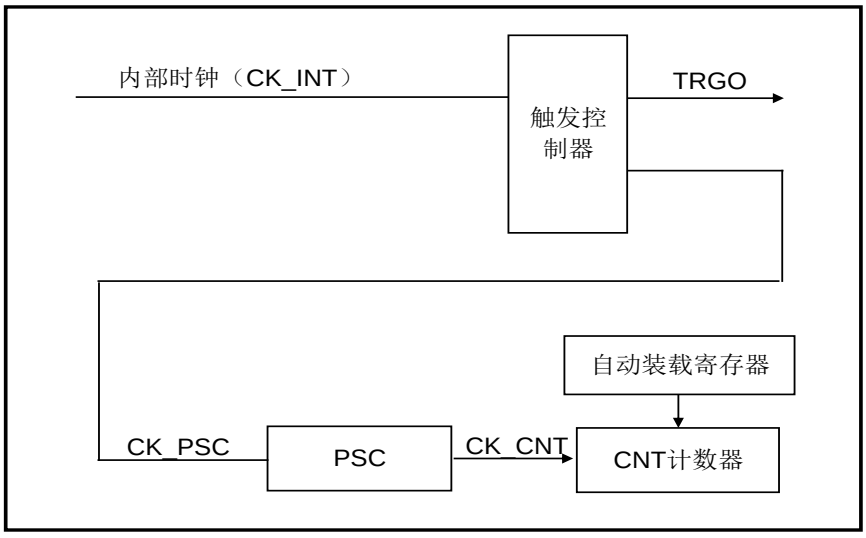
基本定时器（TIM6/7/21/22）包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。它们可以作为通用定时器提供时间基准。

14.2. 主要特性

- 16 位自动重装载累加计数器
- 16 位可编程(可实时修改)预分频器，用于对输入的时钟按系数为 1～65536 之间的任意数值分频
- 在更新事件(计数器溢出)时产生中断/DMA 请求

14.3. 结构框图

图 14-1 基本定时器框图



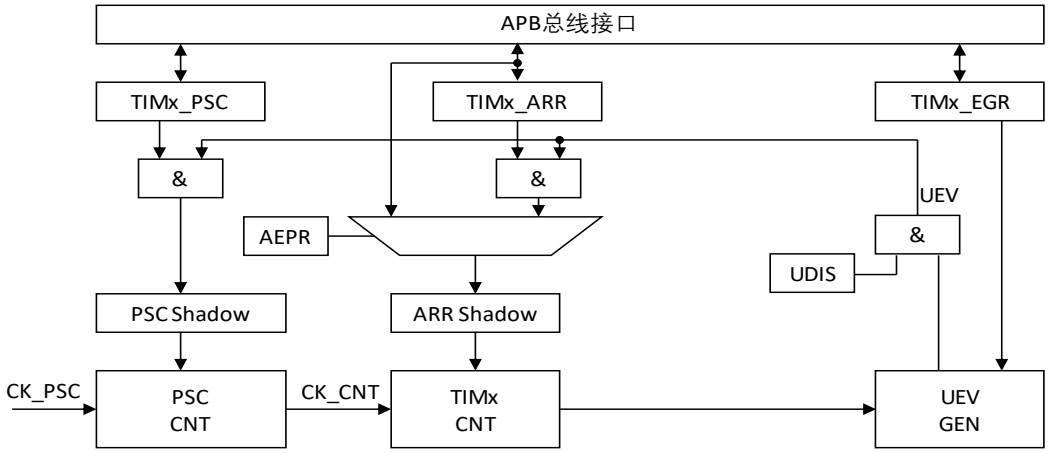
14.4. 功能描述

14.4.1. 计数单元

基本定时器定时器的主要部分是一个 16 位向上计数器和与其相关的自动装载寄存器。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)

图 14-2 计数单元的结构



自动重装载寄存器是预加载的，每次读写自动重装载寄存器时，实际上是通过读写预加载寄存器实现。根据 TIMx_CR1 寄存器中的自动重装载预加载使能位(ARPE)，写入预加载寄存器的内容能够立即或在每次更新事件时，传送到它的影子寄存器。当 TIMx_CR1 寄存器的 UDIS 位为’0’，则每当计数器达到溢出值时，硬件发出更新事件；软件也可以产生更新事件；关于更新事件的产生，随后会有详细的介绍。 计数器由预分频输出 CK_CNT 驱动，设置 TIMx_CR1 寄存器中的计数器使能位(CEN)使能计数器计数。注意：实际的设置计数器使能信号 CNT_EN 相对于 CEN 滞后一个时钟周期。

基本定时器 TIM6 支持一种计数模式：

- 向上计数模式

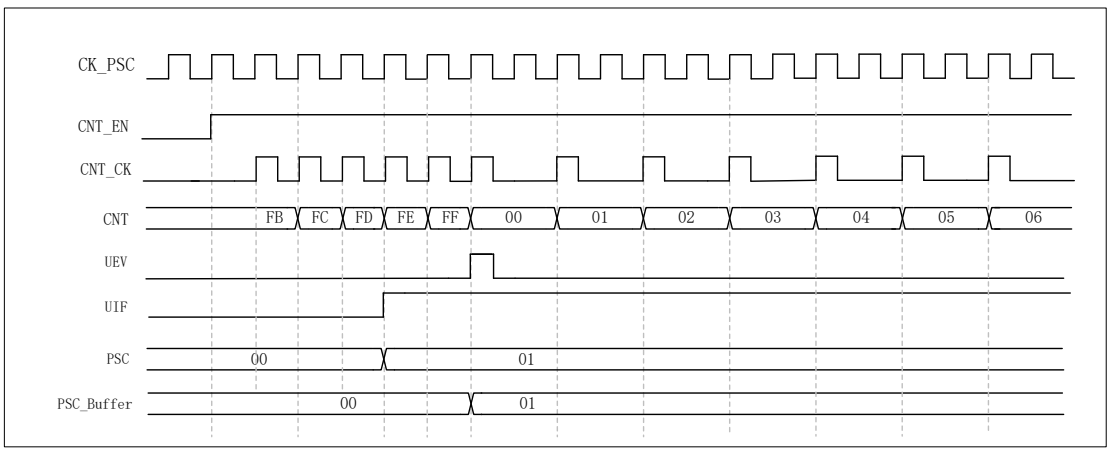
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

14.4.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

和给出了在预分频器运行时，更改计数器参数的例子。

图 14-3 当预分频器的参数从 1 变到 2 时，计数器的时序图



14.4.3. 时钟源

计数器的时钟由内部时钟(CK_INT)提供。TIMx_CR1 寄存器的 CEN 位和 TIMx_EGR 寄存器的 UG 位是实际的控制位，(除了 UG 位被自动清除外)只能通过软件改变它们。一旦置 CEN 位为’1’，内部时钟即向预分频器提供时钟。

14.4.4. 定时器调试模式

定时器在调试时依然在运行。

14.5. TIM6/7/21/22 寄存器描述

14.5.1. 寄存器列表

TIM6 寄存器基地址：0x40001000

TIM7 寄存器基地址：0x40001400

TIM21 寄存器基地址：0x40019C00

TIM22 寄存器基地址：0x4001A000

表 14-1 基本定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	-	保留
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器

14.5.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:4	RSV	-	-	保留，始终读为 0

3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

14.5.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0

6:4	MMS	RW	0x0	主模式选择（Master mode selection） 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3:0	RSV	-	-	保留，始终读为 0

14.5.4. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留，始终读为 0
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable）0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7:1	RSV	-	-	保留，始终读为 0
0	UIE	RW	0x0	允许更新中断（Update interrupt enable） 0：禁止更新中断； 1：允许更新中断。

14.5.5. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:1	RSV	-	-	保留，始终读为 0

0	UIF	RW	0x0	更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

14.5.6. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:1	RSV	-	-	保留，始终读为 0
0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

14.5.7. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value）

14.5.8. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

14.5.9. 自动重装载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

15. 通用定时器（TIM9/12）

15.1. 概述

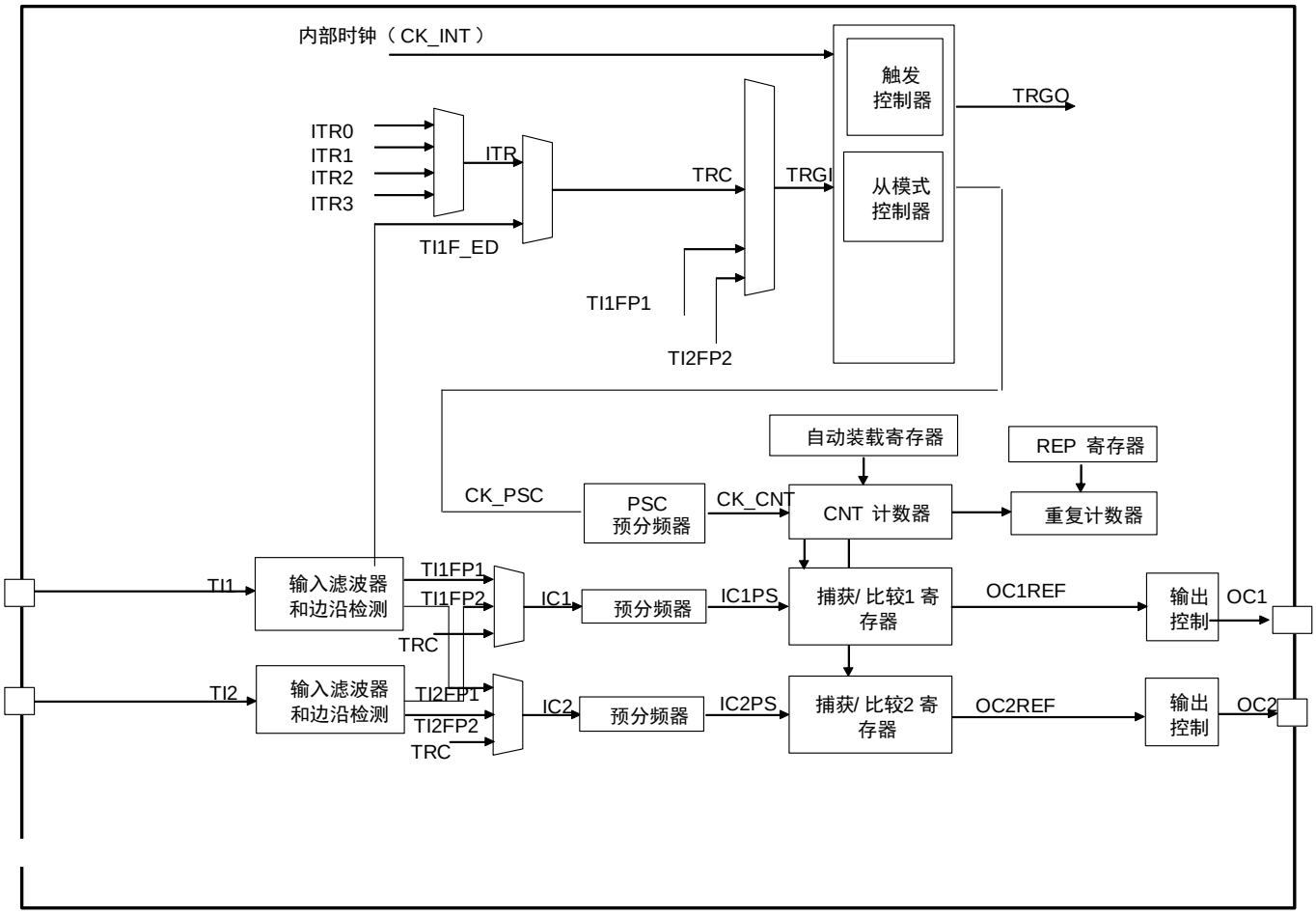
通用定时器（TIM9/12）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM 等)。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

15.2. 主要特性

- 16 位向上、向下、向上/下自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 2 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘或中间对齐模式)
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较

15.3. 结构框图

图 15-1 通用定时器框图



15.4. TIMx 输入映射

表 15-1 TIMx 内部触发输入（ITRx）

	TIM9 分配的 ITR	TIM12 分配的 ITR
ITR0	tim2_trgo	tim4_trgo
ITR1	tim3_trgo	tim5_trgo
ITR2	tim10_oc1	tim13_oc1
ITR3	tim11_oc1	tim14_oc1
ITR4	-	-
ITR5		-
ITR6		-
ITR7		-
ITR8		-
保留	-	-

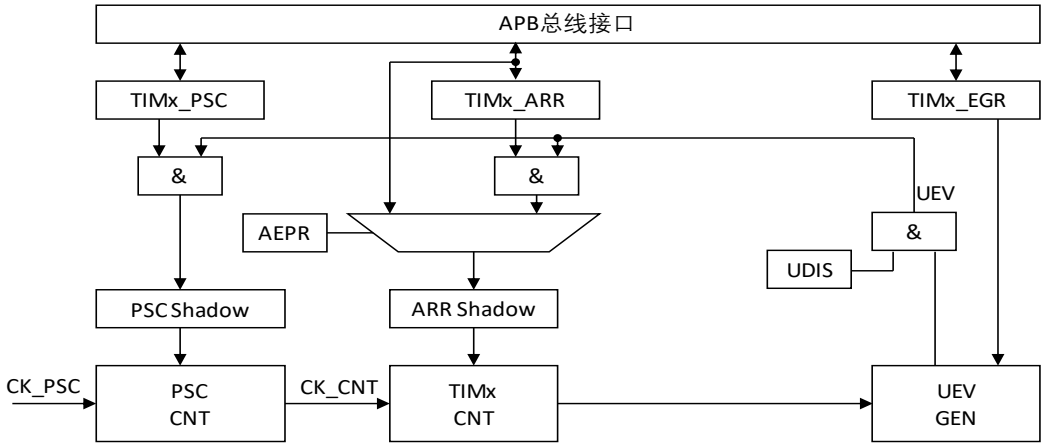
15.5. 功能描述

15.5.1. 计数单元

可编程通用定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)

图 15-2 计数单元的结构



自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

通用定时器支持三种计数模式：

- 向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

- 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

- 中央对齐模式

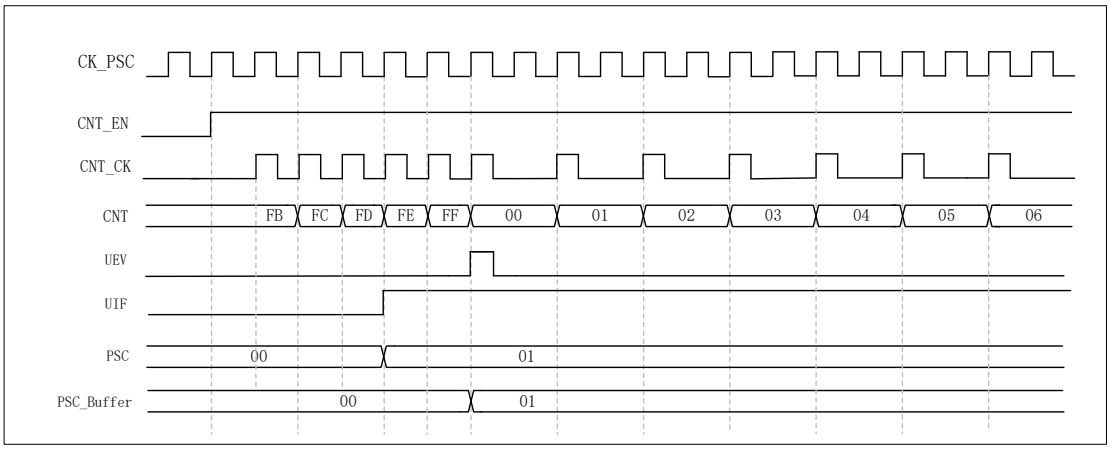
在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。向上计数模式中，定时器模块在计数器计数到自动加载值-1 产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMx_CR1 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。计数方向被硬件自动更新。

15.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 15-3 当预分频器的参数从 1 变到 2 时，计数器的时序图

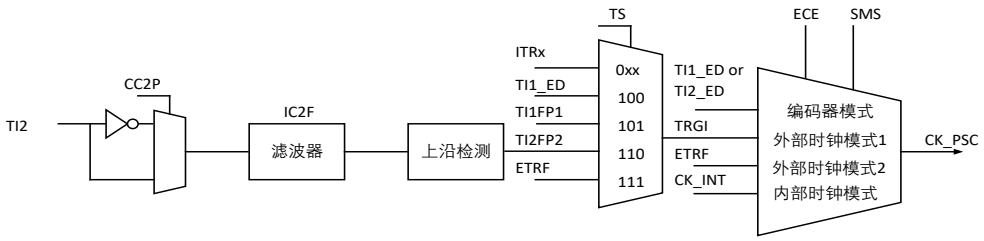


15.5.3. 时钟源选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚
- 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。

图 15-4 外部时钟模式 1



但如果按功能划分如以上 2 张图示所示，并按照定时器的从模式控制寄存器 TIMx_SMCR 的 ECE 和 SMS 的控制，应该分为以下几种模式：

- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式只要 CEN 位被写成 ‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有八个信号源，并由 TIMx_SMCR.TS 寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 1（ITR1）

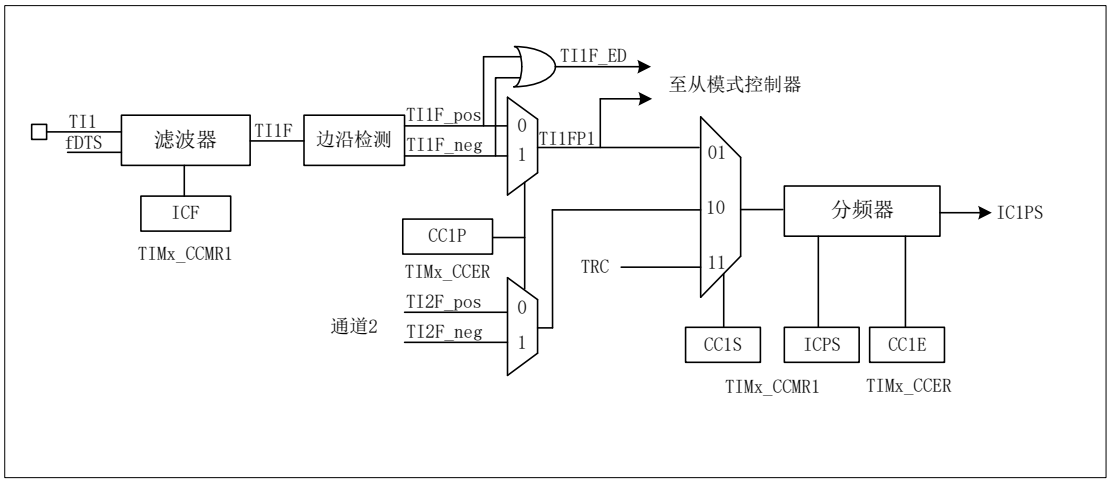
- TS=010，内部触发 2（ITR2）
- TS=011，内部触发 3（ITR3）
- TS=100，TI1 的边沿检测器（TI1F_ED）
- TS=101，滤波后的定时器输入 1（TI1FP1）
- TS=110，滤波后的定时器输入 2（TI2FP2）
- TS=111，外部触发输入（ETRF）

15.5.4. 捕获比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 4 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

图 15-5 捕获/比较通道(如：通道 1 输入部分)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 15-6 捕获/比较通道 1 的主电路

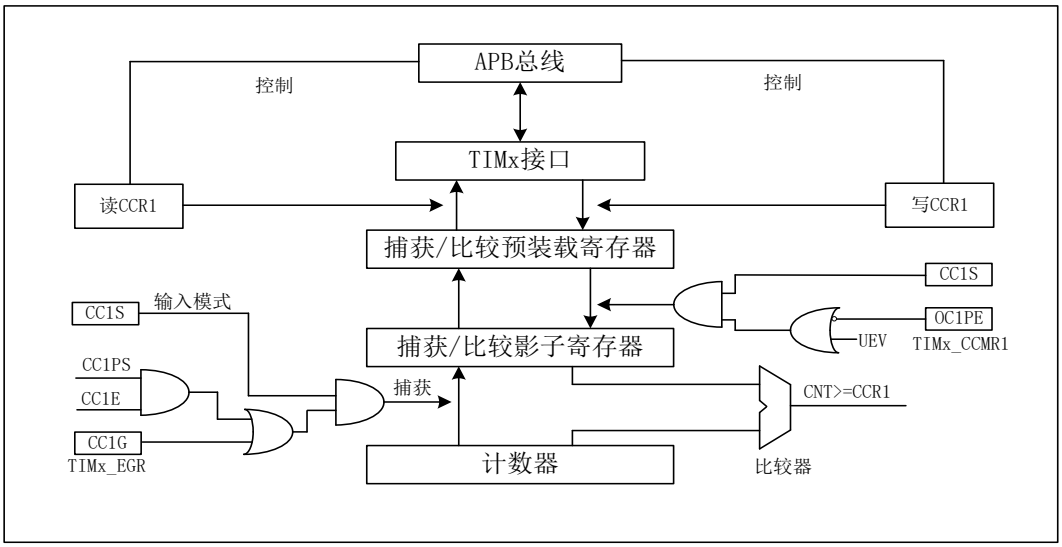
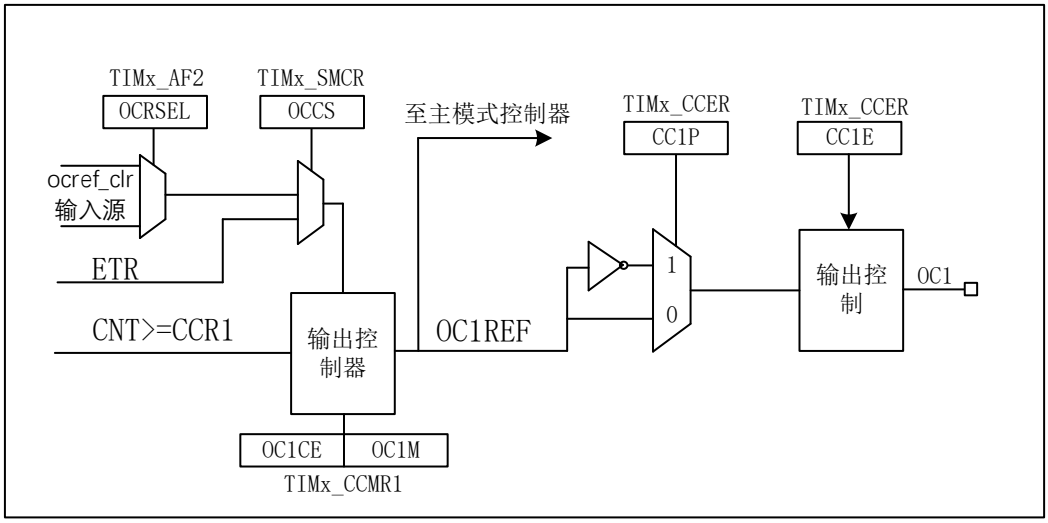


图 15-7 捕获/比较通道的输出部分(通道 1)



15.5.4.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

15.5.4.2. PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

1）两个 ICx 信号被映射至同一个 TIx 输入。

2）这 2 个 ICx 信号为边沿有效，但是极性相反。

3）其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。 例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

4）选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。

5）选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。

6）选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。

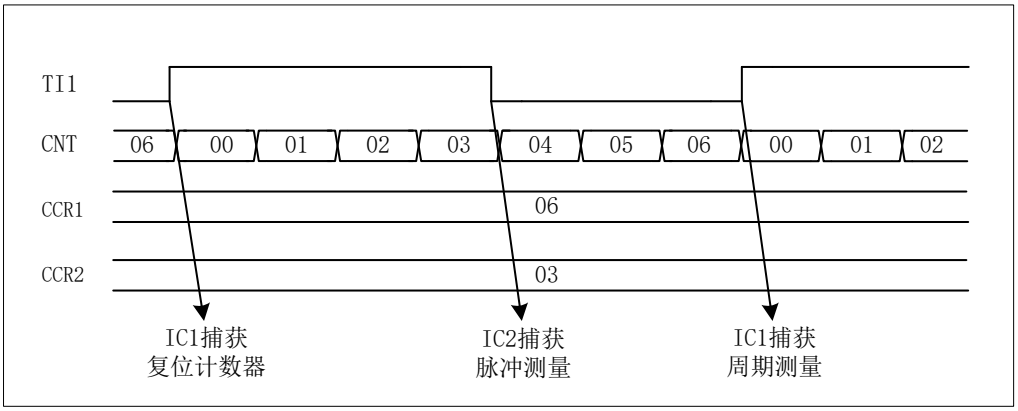
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

8）选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。

9）配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。

10）使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

图 15-8 PWM 输入模式时序



15.5.4.3. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

15.5.4.4. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

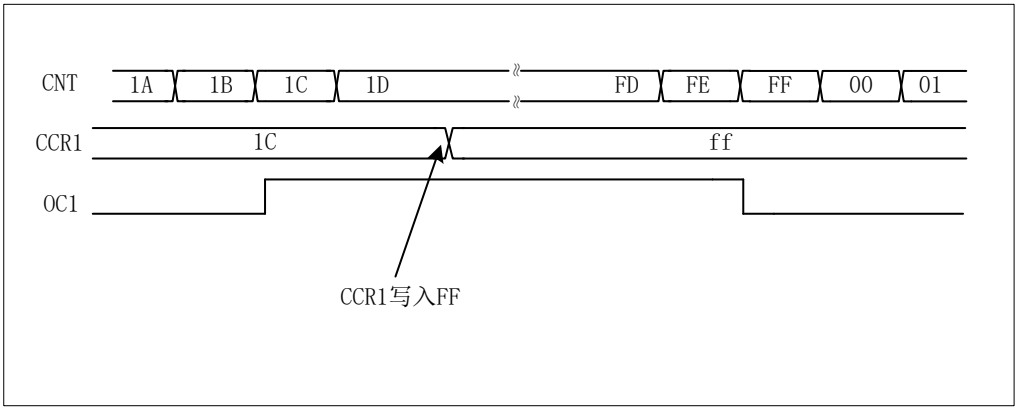
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 15-9 输出比较模式，翻转 OC1



15.5.4.5. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

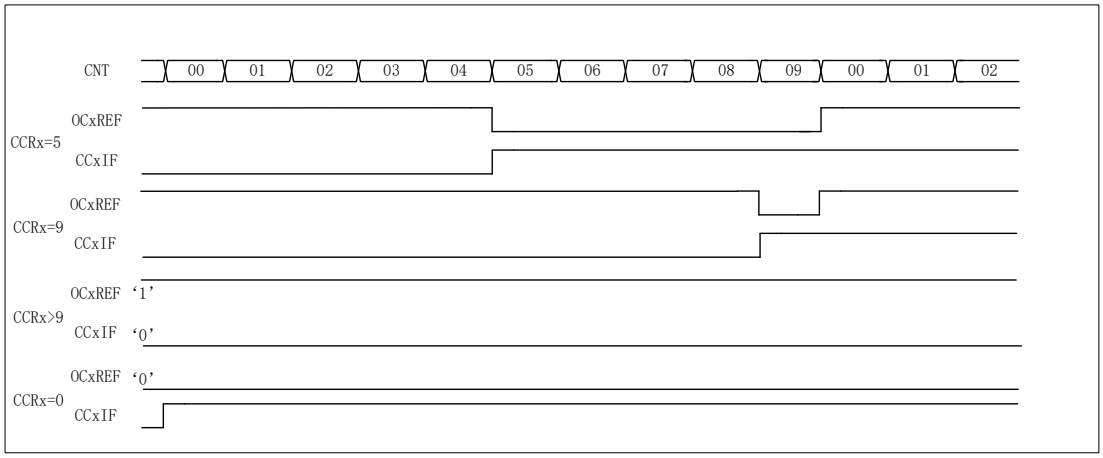
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ PWM 边沿对齐模式

● 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=9 时边沿对齐的 PWM 波形实例。

图 15-10 边沿对齐的 PWM 波形（ARR=9）



● 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

■ PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

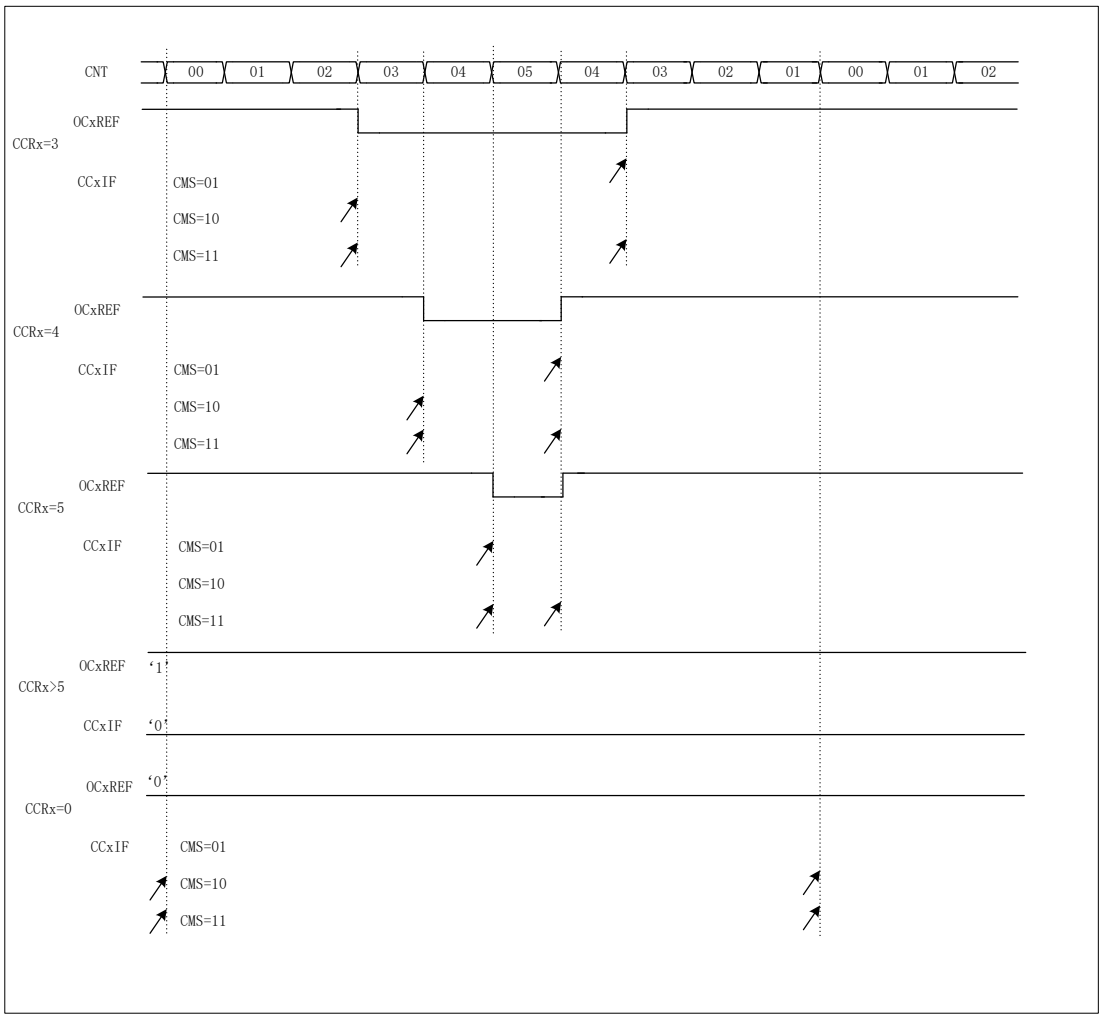
下图给出了一些中央对齐的 PWM 波形的例子

1）TIMx_ARR=5

2）PWM 模式 1

3）TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图 15-11 中央对齐的 PWM 波形（ARR=5）



使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
- 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
- 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

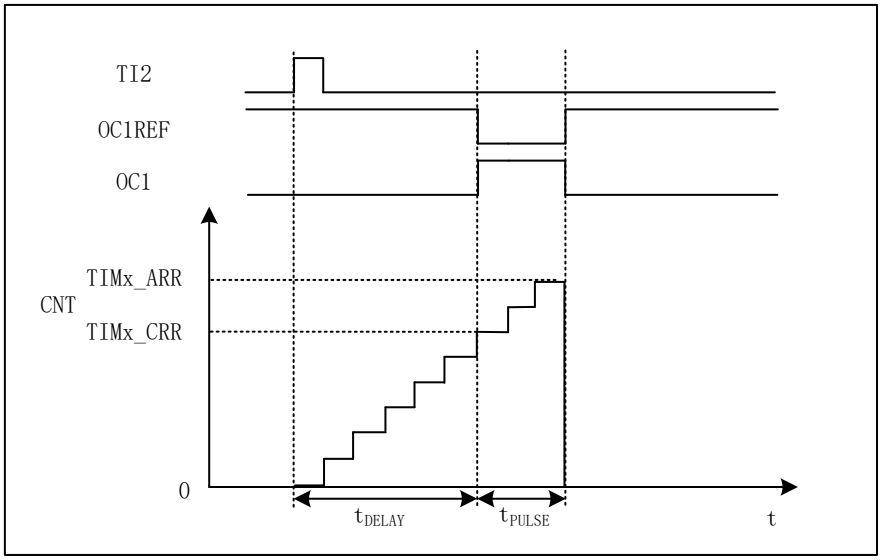
15.5.4.6. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 15-12 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

15.5.5. 定时器互连

TIMx 定时器能够在从模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

15.5.5.1. 复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR， TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

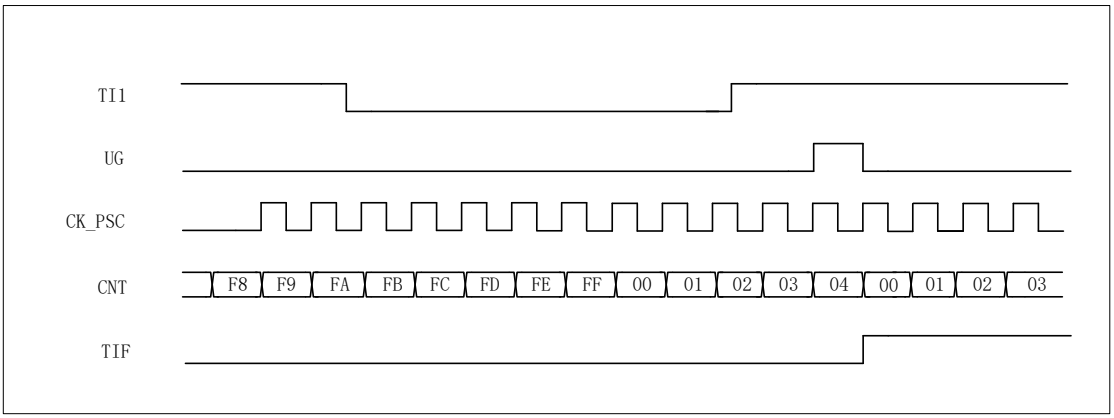
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

2）置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。 下图显示当自动重装载寄存器 TIMx_ARR=0xFF 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 15-13 复位模式下的控制电路



15.5.5.2. 门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

1）配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

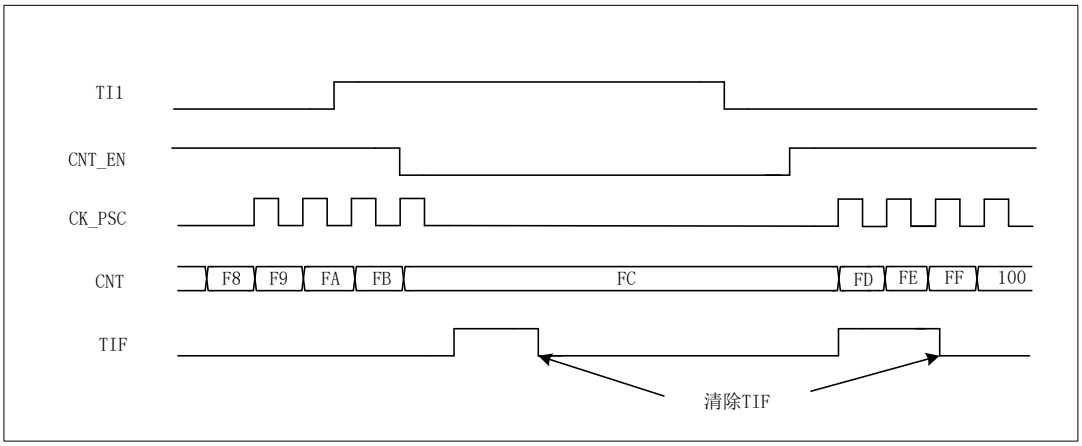
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图 15-14 门控模式下的控制电路



15.5.5.3. 触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

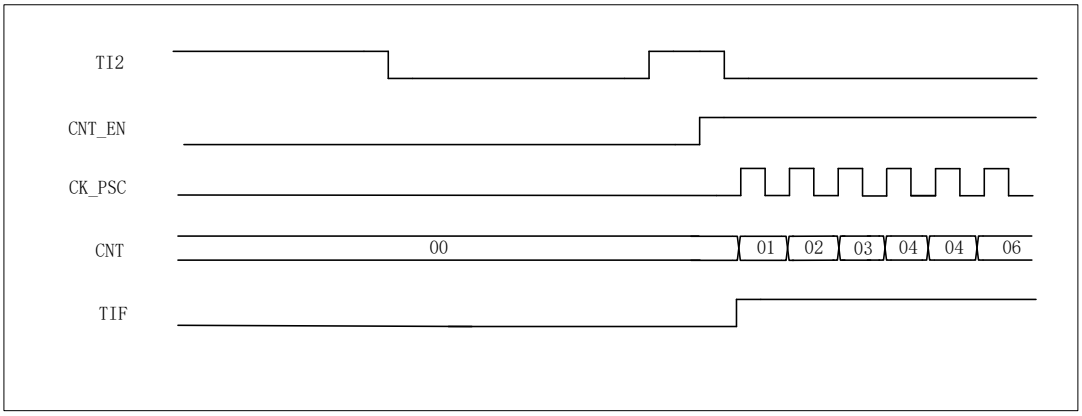
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

2）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 15-15 触发模式下的控制电路



15.5.6. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

SingleDMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMx 会给 DMA 发送请求,等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

15.5.7. 定时器调试模式

定时器在调试时依然在运行。

15.6. TIM9/12 寄存器描述

15.6.1. 寄存器列表

TIM9 寄存器基地址：0x40018000

TIM12 寄存器基地址：0x40001800

表 15-2 通用控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	TIMx_SMCR	TIMx 从模式控制寄存器
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	-	
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	-	保留
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	TIMx_CCR2	TIMx 捕获比较寄存器 2
0x3C	-	保留
0x40	-	保留
0x44	-	保留
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x60	-	保留
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

15.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述

31:10	RSV	-	-	保留，始终读为 0
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟(tDTS)与定时器时钟（CK_INT）的分频比例。 00: tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:5	CMS	RW	0x0	计数模式 00:边沿对齐模式，计数器根据方向位（DIR）向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0x0	方向控制位 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。

1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

15.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留。
7	TI1S	RW	0x0	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
6:4	MMS	RW	0x0	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0x0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。

2	CCUS	RW	0x0	捕获/比较控制更新选择（Capture/compare control update selection） 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0
0	CCPC	RW	0x0	捕获/比较预装载控制位（Capture/compare preloaded control） 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的； 设置该位后，它们只在设置了 COM 位后被更新。

15.6.4. 从模式控制寄存器(TIMx_SMCR: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留，始终读为 0
21:20	TS[4:3]	RW	0x0	触发选择信号
19:8	RSV	-	-	保留，始终读为 0
7	MSM	RW	0x0	主/从模式（Master/slave mode） 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	RW	0x0	触发选择（Trigger selection），和 TS[4:3]组成 5 位的 TS 信号。 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00100：TI1 的边沿检测器(TI1F_ED) 00001：内部触发 1(ITR1) 00101：滤波后的定时器输入 1(TI1FP1) 00010：内部触发 2(ITR2) 00110：滤波后的定时器输入 2(TI2FP2) 00011：内部触发 3(ITR3) 00111：外部触发输入(ETRF) 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 01010：内部触发 6(ITR6) 01011：内部触发 7(ITR7) 01100：内部触发 8(ITR8) 01101：内部触发 9(ITR9) 01110：内部触发 10(ITR10) 其它：保留 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。 内部触发源 ITRx 详情输入源请参看 TIMx 输入映射章节

3	OCCS	RW	0x0	OCREF 清零选择（OCREF clear selection） 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 COMPx 输出，具体取决于 TIMx_AF2. OCRSEL 1： OCREF_CLR_INT 连接到 ETRF
2:0	SMS	RW	0x0	从模式选择（Slave mode selection） 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

15.6.5. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	TDE	RW	0x0	允许触发 DMA 请求（Trigger DMA request enable） 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13：11	RSV	-	-	保留
10	CC2DE	RW	0x0	允许捕获/比较 2 的 DMA 请求（Capture/Compare 2 DMA request enable） 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。
9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable） 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable）0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	RSV	-	-	保留，始终读为 0

6	TIE	RW	0x0	触发中断使能（Trigger interrupt enable） 0：禁止触发中断； 1：使能触发中断。
5:3	RSV	-	-	保留
2	CC2IE	RW	0x0	允许捕获/比较 2 中断（Capture/Compare 2 interrupt enable） 0：禁止捕获/比较 2 中断； 1：允许捕获/比较 2 中断。
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable） 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable） 0：禁止更新中断； 1：允许更新中断。

15.6.6. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留，始终读为 0
10	CC2OF	W0C	0x0	捕获/比较 2 重复捕获标记（Capture/Compare 2 overcapture flag）参见 CC1OF 描述。
9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
8:7	RSV	-	-	保留，始终读为 0
6	TIF	W0C	0x0	触发器中断标记（Trigger interrupt flag） 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5：3	RSV	-	-	保留
2	CC2IF	W0C	0x0	捕获/比较 2 中断标记（Capture/Compare 2 interrupt flag）参考 CC1IF 描述。

1	CC1IF	W0C	0x0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’ 。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’ ，它由软件清’0’或通过读 TIMx_CCR1 清’0’ 。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置’1’ 。它由软件清’0’ 。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’ ： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

15.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TG	WO	0x0	产生触发事件 (Trigger generation) 该位由软件置’1’ ，用于产生一个触发事件，由硬件自动清’0’ 。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5：3	RSV	-	-	保留
2	CC2G	WO	0x0	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。
1	CC1G	WO	0x0	产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置’1’ ，用于产生一个捕获/比较事件，由硬件自动清’0’ 。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。

0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

15.6.8. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义

输出比较模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	OC2CE	RW	0x0	输出比较 2 清 0 使能（Output Compare 2 clear enable）
14:12	OC2M	RW	0x0	输出比较 2 模式（Output Compare 2 mode）
11	OC2PE	RW	0x0	输出比较 2 预装载使能（Output Compare 2 preload enable）
10	OC2FE	RW	0x0	输出比较 2 快速使能（Output Compare 2 fast enable）
9:8	CC2S	RW	0x0	捕获/比较 2 选择。(Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0x0	输出比较 1 清’0’使能（Output Compare 1 clear enable） 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。

6:4	OC1M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
15:12	IC2F	RW	0x0	输入捕获 2 滤波器（Input capture 2 filter）
11:10	IC2PSC	RW	0x0	输入/捕获 2 预分频器（Input capture 2 prescaler）
9:8	CC2S	RW	0x0	捕获/比较 2 选择（Capture/Compare 2 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

15.6.9. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	CC2NP	RW	0x0	输入/捕获 2 互补输出极性（Capture/Compare 2 complementary output polarity）参考 CC1NP 的描述。
6	CC2NE	RW	0x0	输入/捕获 2 互补输出使能（Capture/Compare 2 complementary output enable）参考 CC1NE 的描述。
5	CC2P	RW	0x0	输入/捕获 2 输出极性（Capture/Compare 2 output polarity）参考 CC1P 的描述。
4	CC2E	RW	0x0	输入/捕获 2 输出使能（Capture/Compare 2 output enable）参考 CC1E 的描述。
3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效; 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别（TIMx_BDTR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00 (通道配置为输出）则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。

0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 1：捕获使能

15.6.10. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value)

15.6.11. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value) 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

15.6.12. 自动加载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value) ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

15.6.13. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0

15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

15.6.14. 捕获/比较寄存器 2(TIMx_CCR2: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR2	RW	0x0	捕获/比较通道 2 的值（Capture/Compare 2 value） 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

15.6.15. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0
12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIMx_CR1 - 如果 DBL=7，DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0

4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

15.6.16. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它的最大值取决于 TIMx_DCR 寄存器中定义的 DBL。

15.6.17. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留，始终为 0。
11:8	TI2SEL	RW	0x0	TI2 输入选择 0000：tim_ti2_in0 其他值：保留
7:4	RSV	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：tim_ti1_in0 其他值：保留

15.6.18. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5:3	RSV	-	-	保留
2	CC2BE	RW	0x0	捕获/比较 2 事件的 DMA 请求类型 0：Single； 1：Burst；

1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；
0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

16. 通用定时器（TIM10/11/13/14）

16.1. 概述

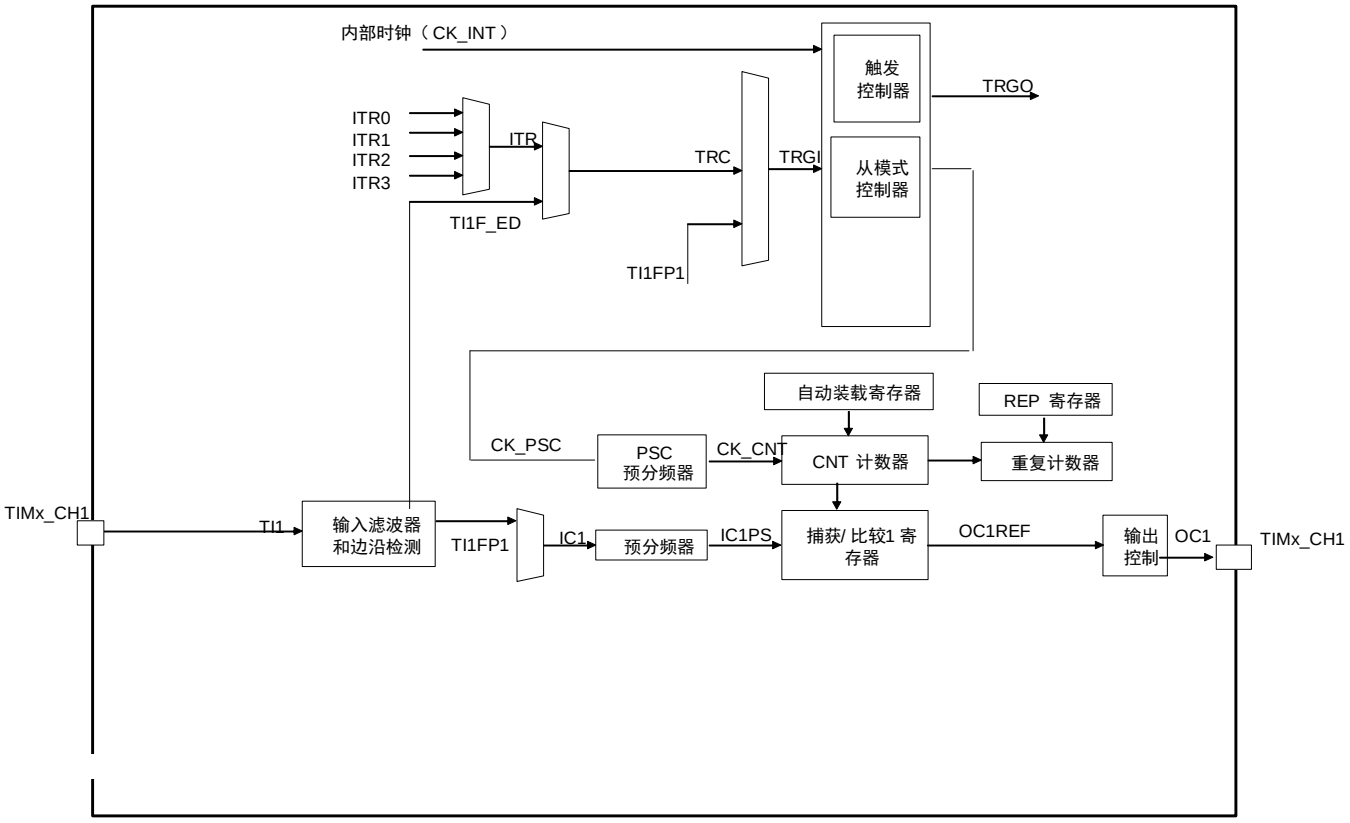
通用定时器（TIM10/11/13/14）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含产生输出波形(输出比较、PWM 等)。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

16.2. 主要特性

- 16 位向上、向下、向上/下自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 1 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘或中间对齐模式)
  - 单脉冲模式输出
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较

16.3. 结构框图

图 16-1 通用定时器框图



16.4. TIMx 输入映射

不支持。

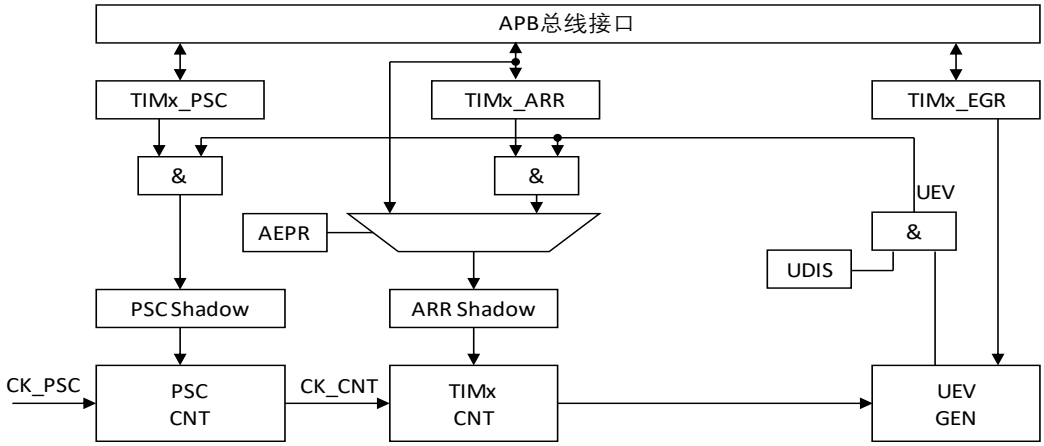
16.5. 功能描述

16.5.1. 计数单元

可编程通用定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)

图 16-2 计数单元的结构



自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

通用定时器支持三种计数模式：

- 向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

- 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

- 中央对齐模式

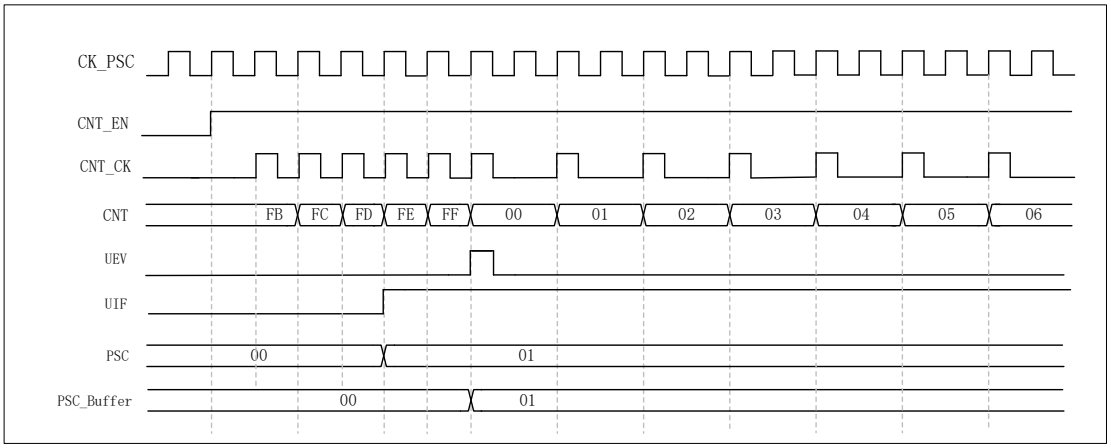
在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。向上计数模式中，定时器模块在计数器计数到自动加载值-1 产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMx_CR1 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。计数方向被硬件自动更新。

16.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 16-3 当预分频器的参数从 1 变到 2 时，计数器的时序图

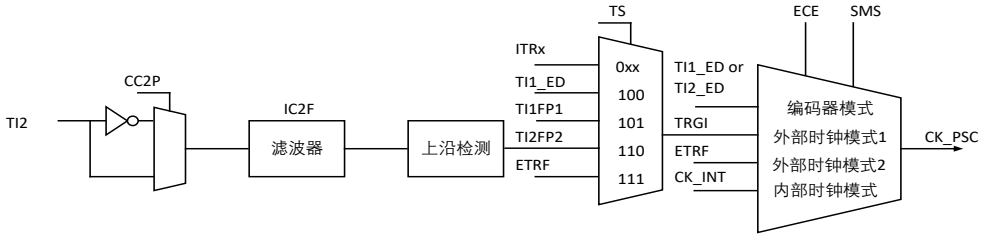


16.5.3. 时钟源选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚

图 16-4 外部时钟模式 1



但如果按功能划分如以上 2 张图示所示，并按照定时器的从模式控制寄存器 TIMx_SMCR 的 ECE 和 SMS 的控制，应该分为以下几种模式：

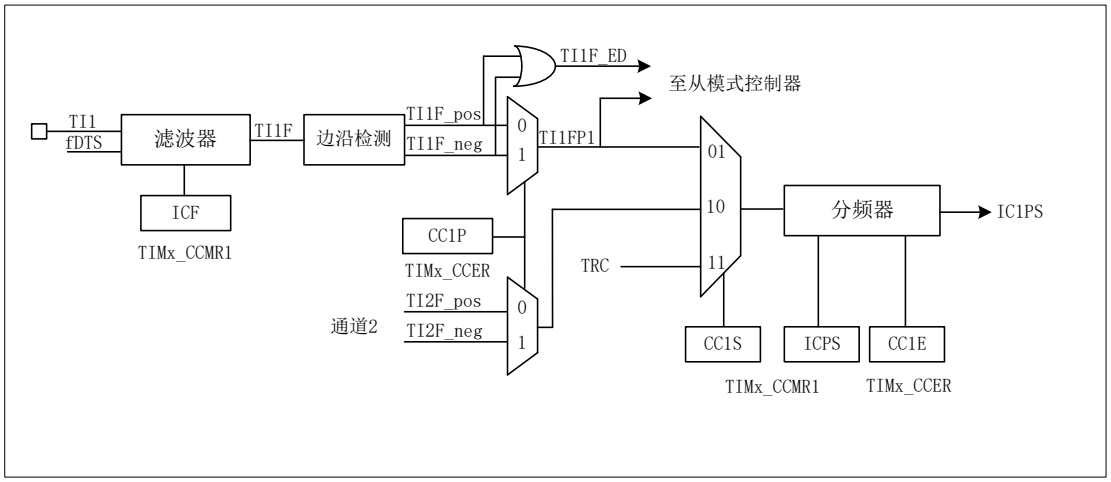
- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有八个信号源，并由 TIMx_SMCR.TS 寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 1（ITR1）
  - TS=010，内部触发 2（ITR2）
  - TS=011，内部触发 3（ITR3）
  - TS=100，TI1 的边沿检测器（TI1F_ED）
  - TS=101，滤波后的定时器输入 1（TI1FP1）
  - TS=110，滤波后的定时器输入 2（TI2FP2）
  - TS=111，外部触发输入（ETRF）

16.5.4. 捕获比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 4 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

图 16-5 捕获/比较通道(如：通道 1 输入部分)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 16-6 捕获/比较通道 1 的主电路

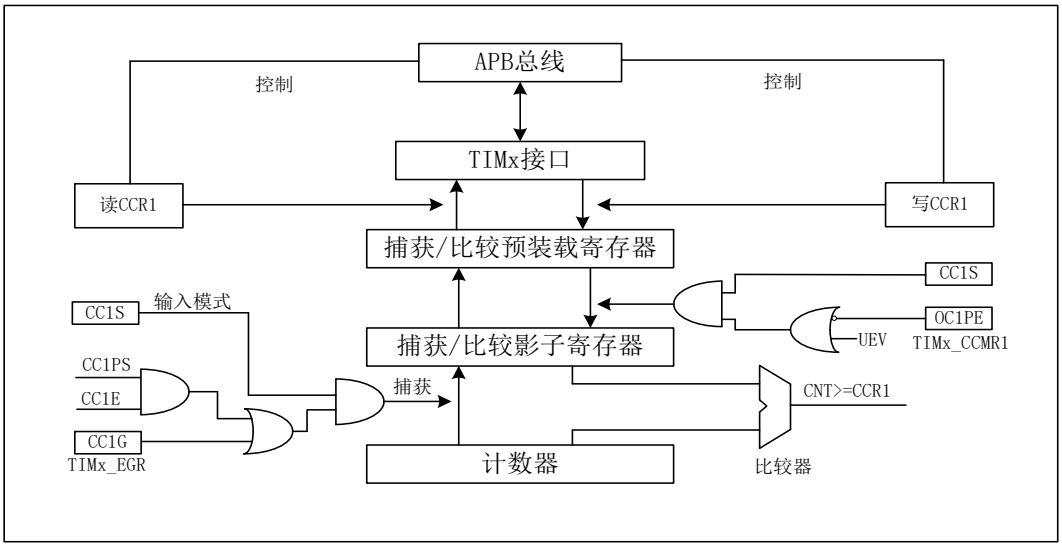
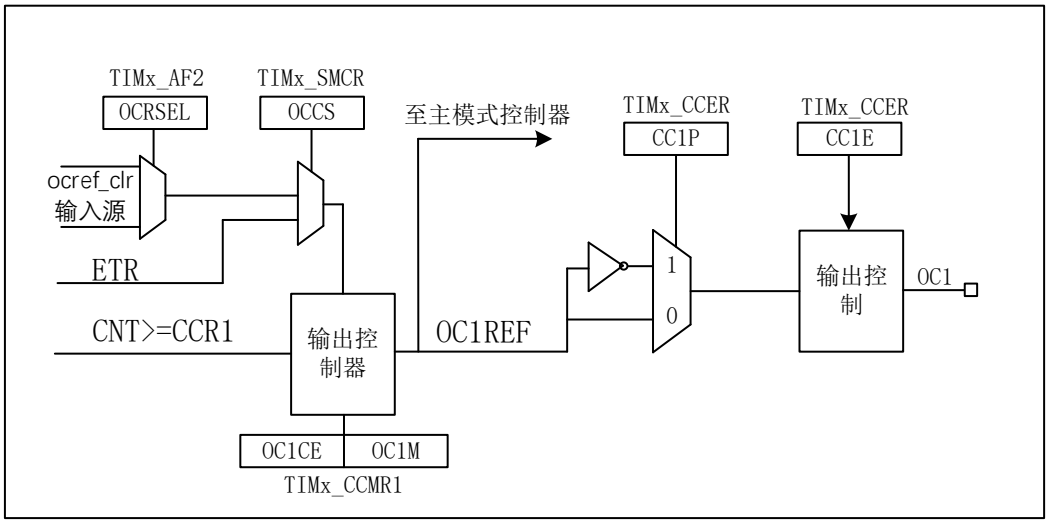


图 16-7 捕获/比较通道的输出部分(通道 1)



16.5.4.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

16.5.4.2. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

16.5.4.3. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

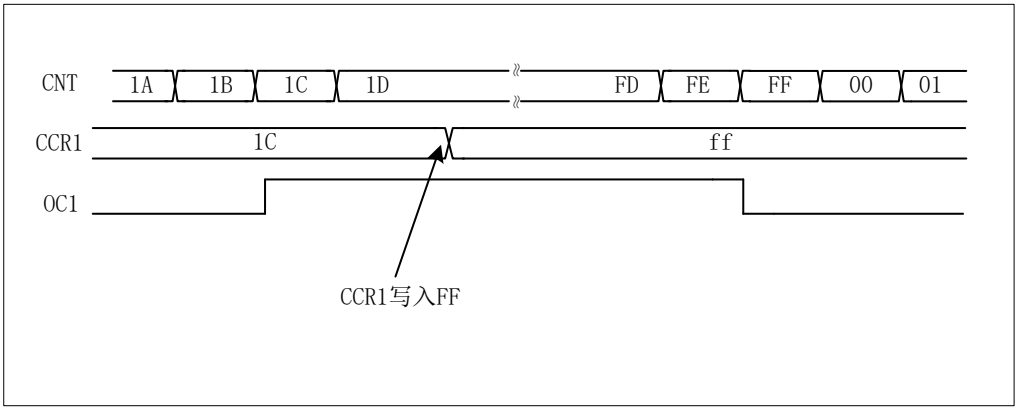
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 16-8 输出比较模式，翻转 OC1



16.5.4.4. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

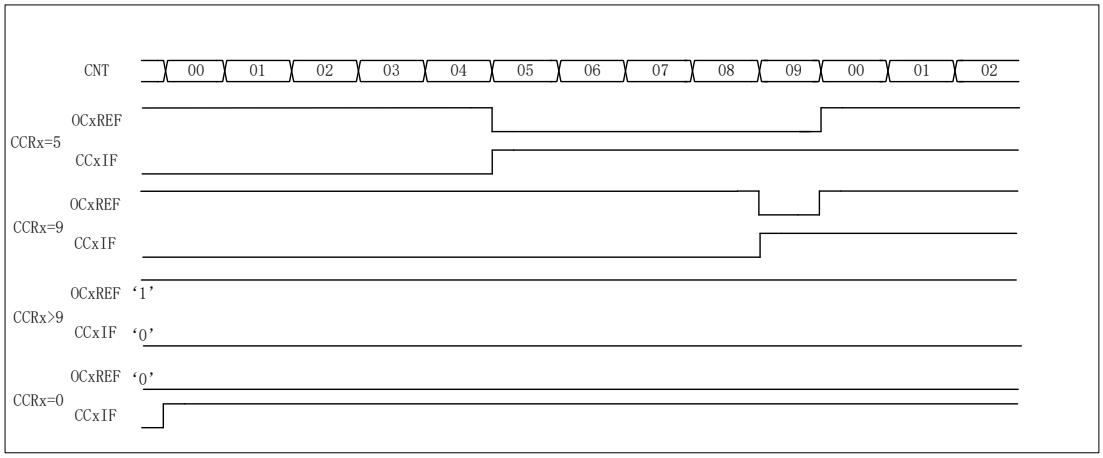
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ PWM 边沿对齐模式

● 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=9 时边沿对齐的 PWM 波形实例。

图 16-9 边沿对齐的 PWM 波形（ARR=9）



● 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

■ PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

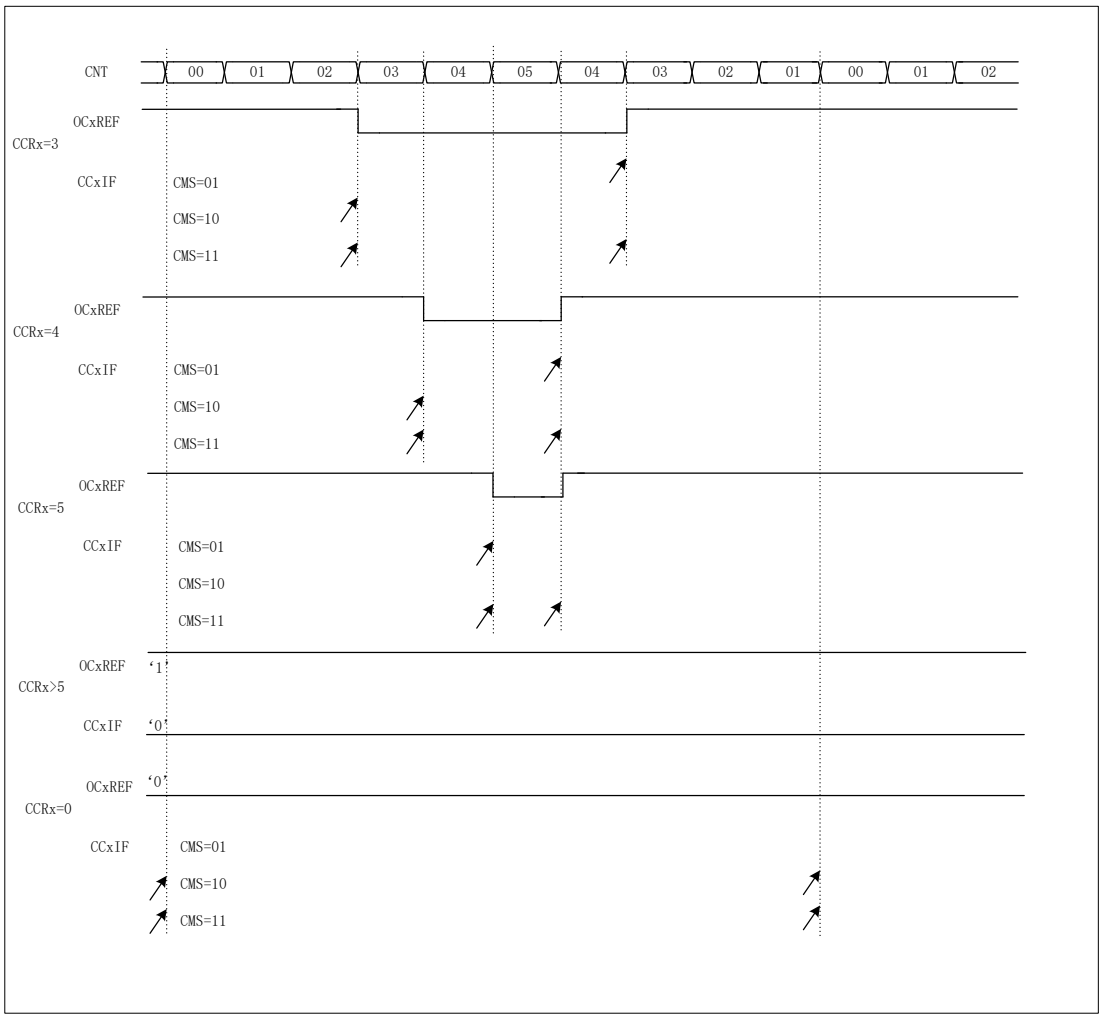
下图给出了一些中央对齐的 PWM 波形的例子

1）TIMx_ARR=5

2）PWM 模式 1

3）TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图 16-10 中央对齐的 PWM 波形（ARR=5）



使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
- 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
- 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

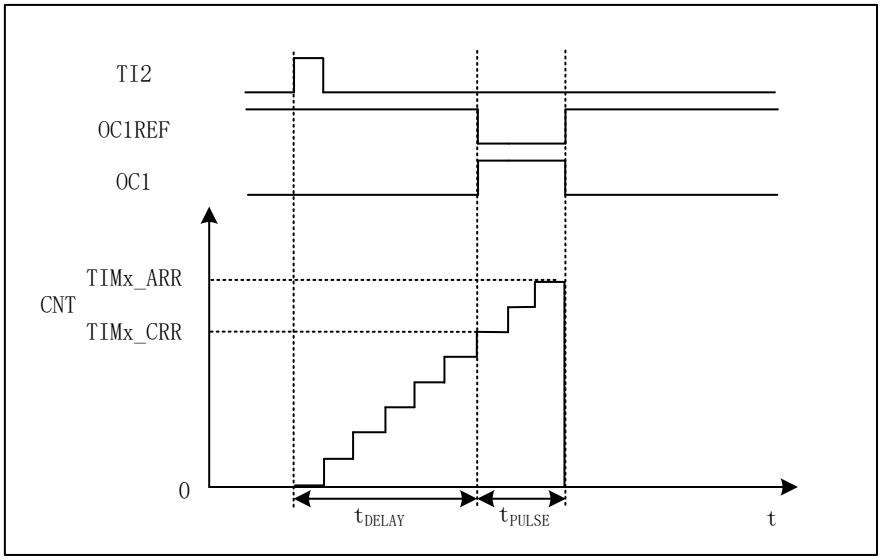
16.5.4.5. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 16-11 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

16.5.5. 定时器互连

TIMx 定时器能够在从模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

16.5.5.1. 复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR， TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

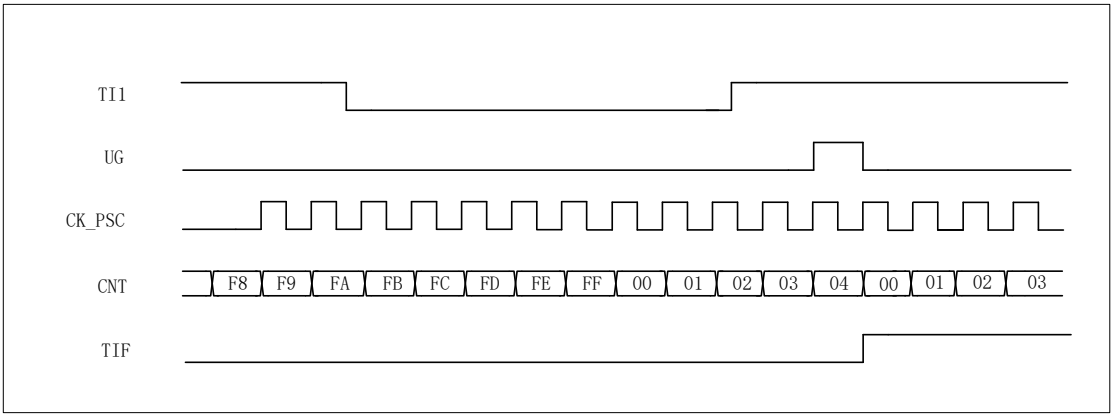
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

2）置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。 下图显示当自动重装载寄存器 TIMx_ARR=0xFF 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 16-12 复位模式下的控制电路



16.5.5.2. 门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

1）配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

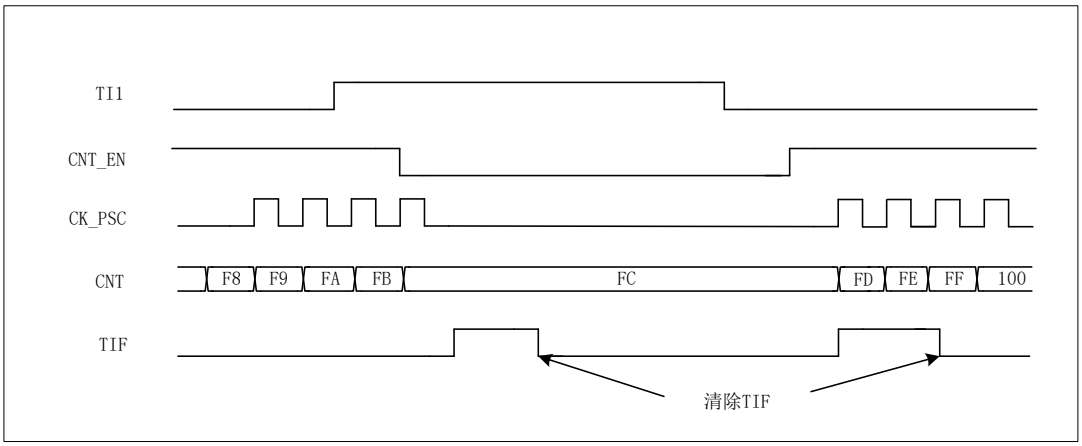
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图 16-13 门控模式下的控制电路



16.5.5.3. 触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

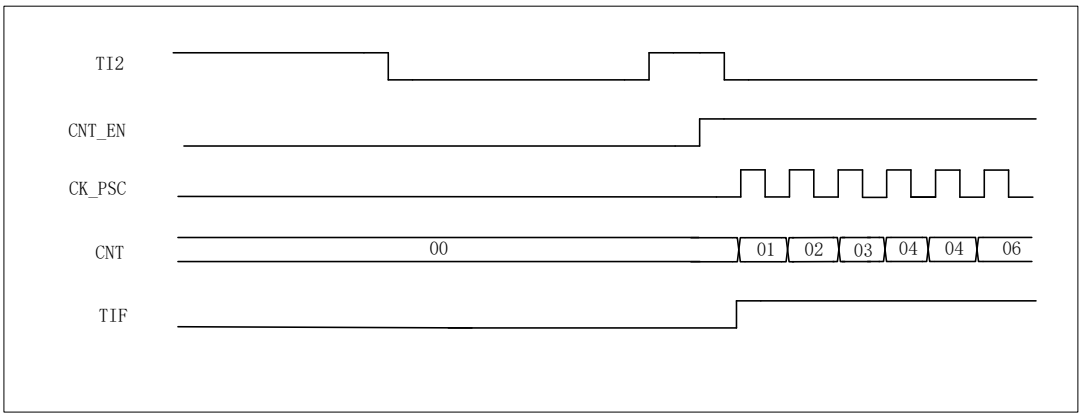
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

2）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 16-14 触发模式下的控制电路



16.5.6. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

SingleDMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事

件发生，TIMx 会给 DMA 发送请求,等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

16.5.7. 定时器调试模式

定时器在调试时依然在运行。

16.6. TIM10/11/13/14 寄存器描述

16.6.1. 寄存器列表

TIM10 寄存器基地址：0x40018400

TIM11 寄存器基地址：0x40018800

TIM13 寄存器基地址：0x40001C00

TIM14 寄存器基地址：0x40002000

表 16-1 通用控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	TIMx_SMCR	TIMx 从模式控制寄存器
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	-	
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器

0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	-	保留
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	-	保留
0x3C	-	保留
0x40	-	保留
0x44	-	保留
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x60	-	保留
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

16.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟(tDTS)与定时器时钟（CK_INT）的分频比例。 00: tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:5	CMS	RW	0x0	计数模式 00:边沿对齐模式，计数器根据方向位（DIR）向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。

4	DIR	RW	0x0	方向控制位 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

16.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留。
7	TI1S	RW	0x0	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。

6:4	MMS	RW	0x0	主模式选择（Master mode selection） 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0x0	捕获/比较的 DMA 选择（Capture/compare DMA selection） 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0x0	捕获/比较控制更新选择（Capture/compare control update selection） 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0
0	CCPC	RW	0x0	捕获/比较预装载控制位（Capture/compare preloaded control） 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的； 设置该位后，它们只在设置了 COM 位后被更新。

16.6.4. 从模式控制寄存器(TIMx_SMCR: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留，始终读为 0
21:20	TS[4:3]	RW	0x0	触发选择信号
19:8	RSV	-	-	保留，始终读为 0
7	MSM	RW	0x0	主/从模式（Master/slave mode） 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。

6:4	TS[2:0]	RW	0x0	触发选择（Trigger selection），和 TS[4:3]组成 5 位的 TS 信号。 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00100：TI1 的边沿检测器(TI1F_ED) 00001：内部触发 1(ITR1) 00101：滤波后的定时器输入 1(TI1FP1) 00010：内部触发 2(ITR2) 00110：滤波后的定时器输入 2(TI2FP2) 00011：内部触发 3(ITR3) 00111：外部触发输入(ETRF) 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 01010：内部触发 6(ITR6) 01011：内部触发 7(ITR7) 01100：内部触发 8(ITR8) 01101：内部触发 9(ITR9) 01110：内部触发 10(ITR10) 其它：保留 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。 内部触发源 ITRx 详情输入源请参看 TIMx 输入映射章节
3	OCCS	RW	0x0	OCREF 清零选择（OCREF clear selection） 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 COMPx 输出，具体取决于 TIMx_AF2. OCRSEL 1： OCREF_CLR_INT 连接到 ETRF
2:0	SMS	RW	0x0	从模式选择（Slave mode selection） 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

16.6.5. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	TDE	RW	0x0	允许触发 DMA 请求（Trigger DMA request enable) 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13：10	RSV	-	-	保留
9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable) 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	RSV	-	-	保留，始终读为 0
6	TIE	RW	0x0	触发中断使能（Trigger interrupt enable) 0：禁止触发中断； 1：使能触发中断。
5：2	RSV	RW	0x0	保留
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable) 0：禁止更新中断； 1：允许更新中断。

16.6.6. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’ 1’ 。
8:7	RSV	-	-	保留，始终读为 0

6	TIF	W0C	0x0	触发器中断标记（Trigger interrupt flag） 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5：2	RSV	-	-	保留
1	CC1IF	W0C	0x0	捕获/比较 1 中断标记（Capture/Compare 1 interrupt flag） 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

16.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TG	WO	0x0	产生触发事件（Trigger generation） 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5：2	RSV	-	-	保留

1	CC1G	WO	0x0	产生捕获/比较 1 事件（Capture/Compare 1 generation） 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

16.6.8. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义

输出比较模式：

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	OC1CE	RW	0x0	输出比较 1 清’0’使能（Output Compare 1 clear enable） 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。

6:4	OC1M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

16.6.9. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留，始终读为 0

3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效； 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别（TIMx_BDTR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00 (通道配置为输出）则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 1：捕获使能

16.6.10. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value)

16.6.11. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

16.6.12. 自动加载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

16.6.13. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

16.6.14. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0

12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIMx_CR1 - 如果 DBL=7，DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0
4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

16.6.15. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses） 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它的最大值取决于 TIMx_DCR 寄存器中定义的 DBL。

16.6.16. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：tim_ti1_in0 其他值：保留

16.6.17. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5:2	RSV	-	-	保留
1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；
0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

17. 通用定时器（TIM15/25）

17.1. 概述

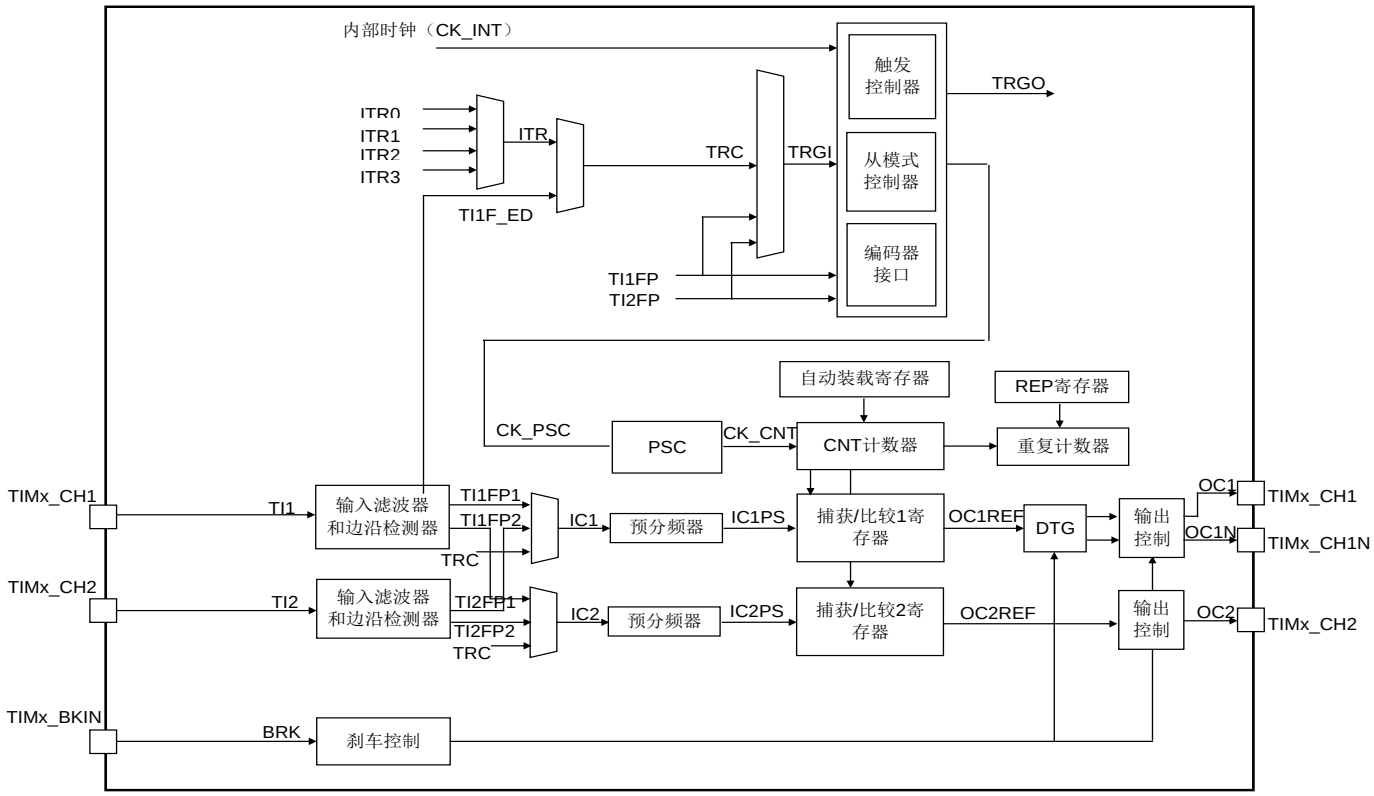
通用定时器由（TIM15/25）一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。 使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

17.2. 主要特性

- 16 位向上自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 2 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 支持针对定位的增量(正交)编码器
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 触发输入作为外部时钟

17.3. 结构框图

图 17-1 通用定时器框图



17.4. TIMx 输入映射

表 17-1 TIMx 内部触发输入（ITRx）

	TIM15 源
ITR0	tim1_trgo
ITR1	tim2_trgo
ITR2	tim3_trgo
ITR3	tim4_trgo
ITR4	-
ITR5	tim8_trgo
ITR6	-
ITR7	tim16_oc1
ITR8	tim17_oc1
保留	-

表 17-2 TIMx 通道 1 输入

TI1SEL[3:0]	TIM15 源

0000	tim15_ch1
0001	comp1_out
0010	comp2_out
0011	CLKOUT
0100	RX1
保留	-

表 17-3 TIMx 通道 2 输入

TI2SEL[3:0]	TIM15 源
0000	tim15_ch2
0001	comp2_out
0010	comp3_out
0011	CLKOUT
0100	RX2
保留	-

表 17-4 TIM1 OCREF_CLR 输入

OCRSEL[2:0]	TIM15 源
0000	comp1_out
0001	comp2_out
0010	comp3_out
0011	comp4_out
保留	-

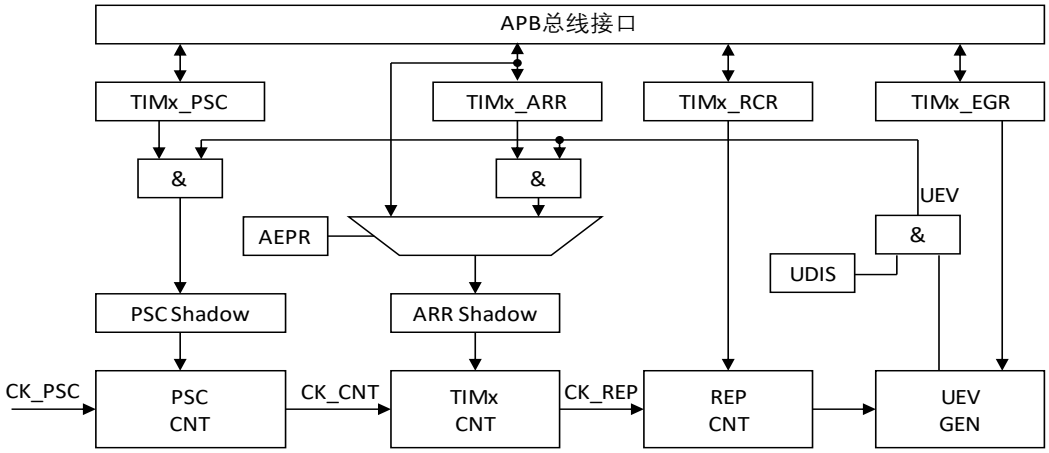
17.5. 功能描述

17.5.1. 计数单元

可编程定时器 TIM15/25 的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)
- 重复次数寄存器（TIMx_RCR)

图 17-2 计数单元的结构



自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

通用定时器 TIM15/25 支持一种计数模式：

- 向上计数模式

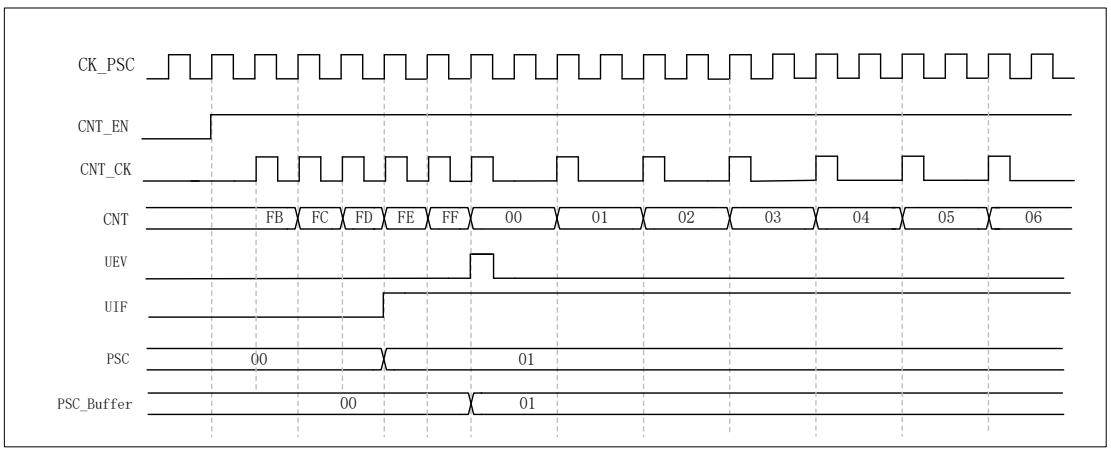
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

17.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 17-3 当预分频器的参数从 1 变到 2 时，计数器的时序图



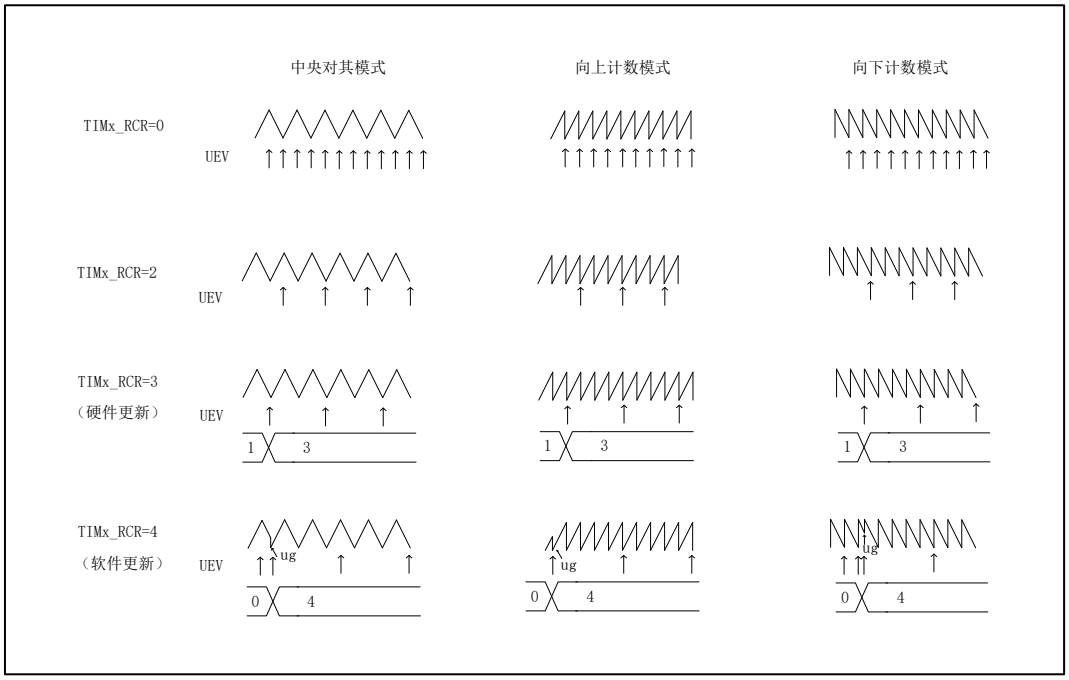
17.5.3. 重复计数器

重复计数器是一个 8 位的递减计数器，更新事件 UEV 只能在重复计数器计数达到 0 时产生。重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

图 17-4 重复计数器溢出事件



17.5.4. 编码器模式

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 17-5 计数器方向和编码器信号的关系

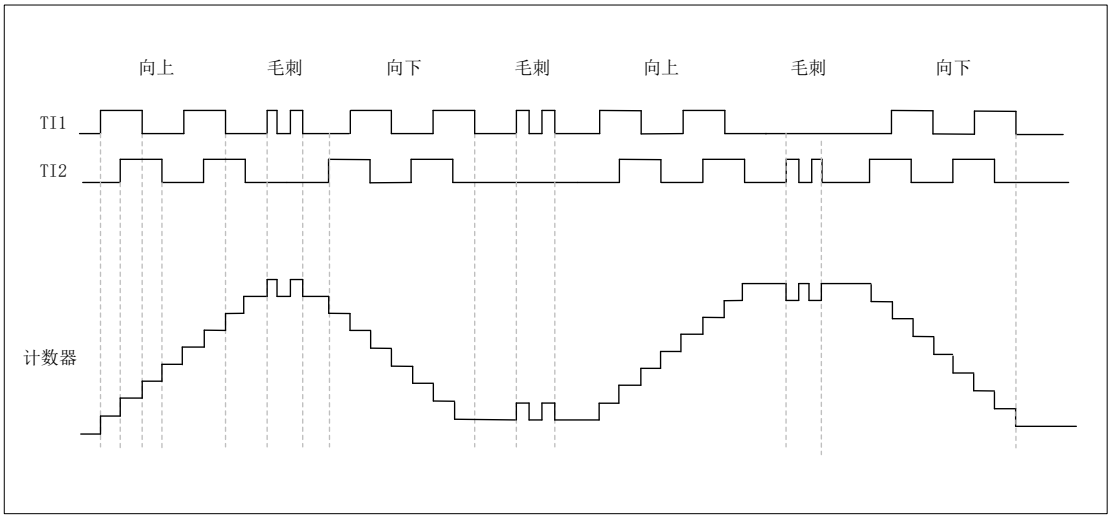
有效边沿	相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1		TI2FP2	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
TI1 和 TI2 都计数	高	向下计数	向上计数	向上计数	向下计数

有效边沿	相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1		TI2FP2	
		上升	下降	上升	下降
	低	向上计数	向下计数	向下计数	向上计数

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。

图 17-5 编码器模式下计数器操作实例



17.5.5. 时钟源选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚
- 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。

图 17-6 时钟源

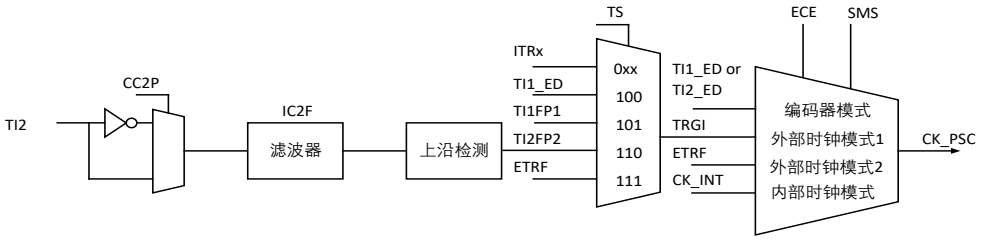


图 17-7 外部时钟模式 1

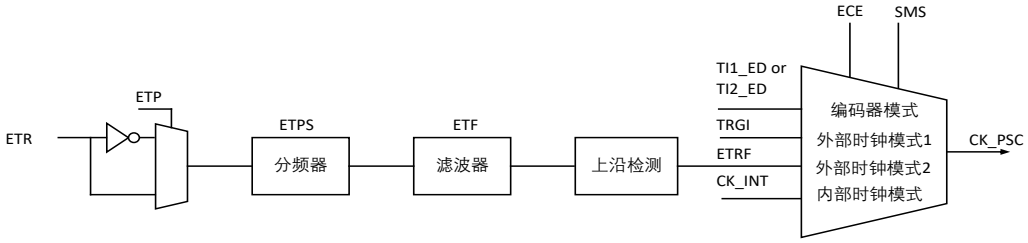


图 17-8 外部时钟模式 2

但如果按功能划分如以上 2 张图示所示，并按照定时器的从模式控制寄存器 TIMx_SMCR 的 ECE 和 SMS 的控制，应该分为以下几种模式：

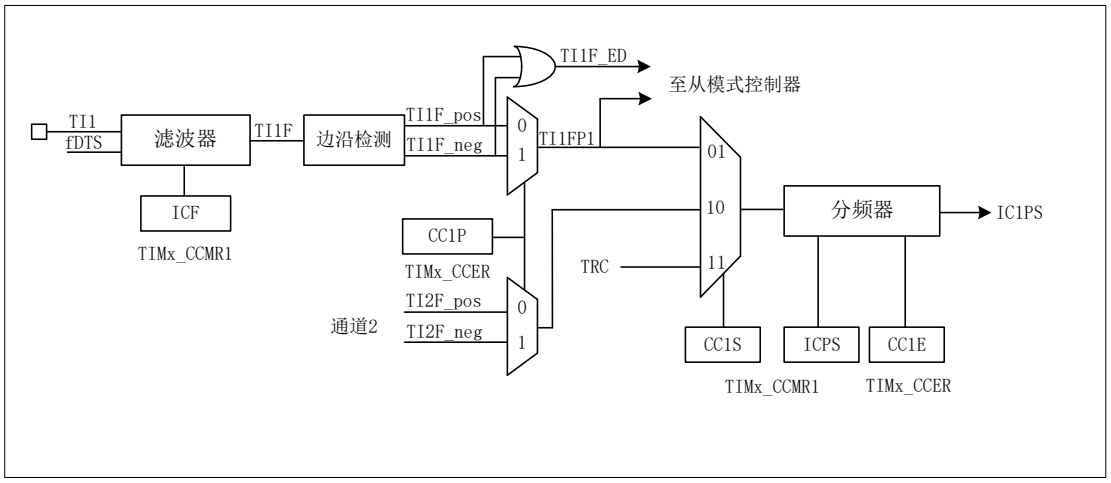
- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有八个信号源，并由 TIMx_SMCR.TS 寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 0（ITR1）
  - TS=010，内部触发 0（ITR2）
  - TS=011，内部触发 0（ITR3）
  - TS=100，TI1 的边沿检测器（TI1F_ED）
  - TS=101，滤波后的定时器输入 1（TI1FP1）
  - TS=110，滤波后的定时器输入 2（TI2FP2）
  - TS=111，外部触发输入（ETRF）
- 外部时钟模式 2：SMS=xxx，ECE=1，CK_PSC 来自 ETRF

17.5.6. 捕获比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 4 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

图 17-9 捕获/比较通道(如：通道 1 输入部分)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 17-10 捕获/比较通道 1 的主电路

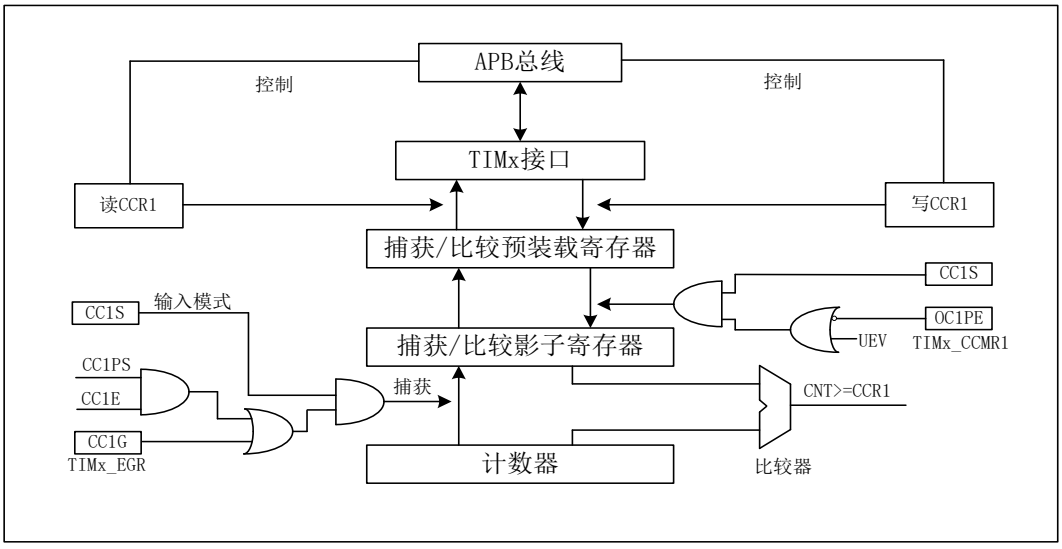


图 17-11 捕获/比较通道的输出部分(通道 1)

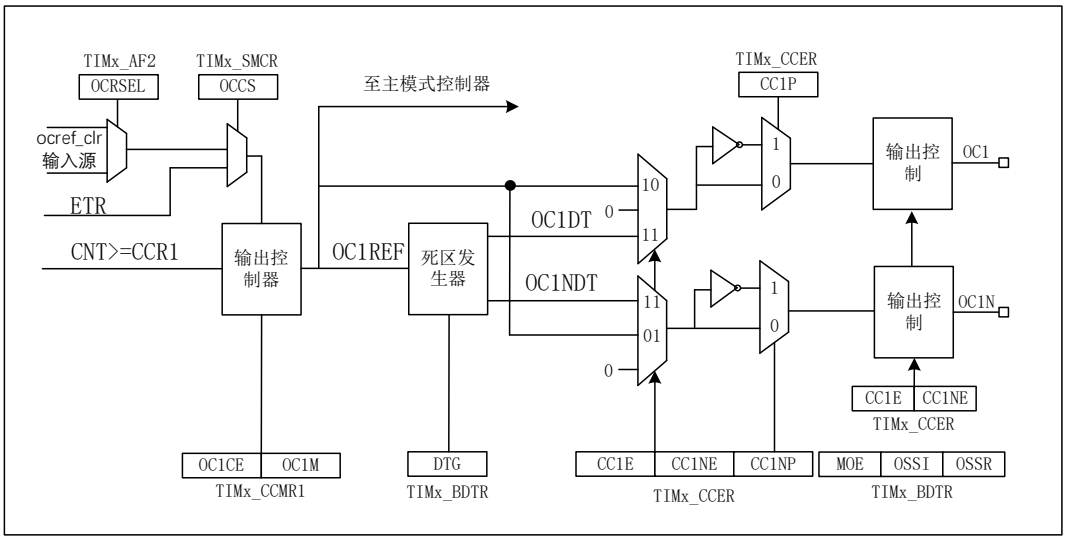
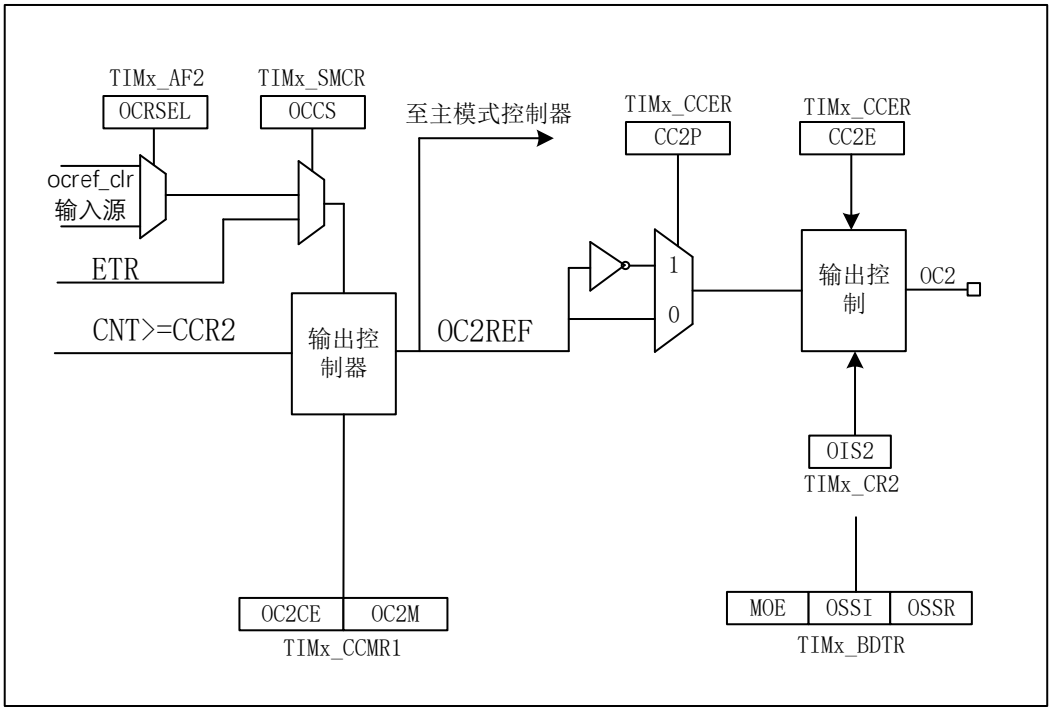


图 17-12 捕获/比较通道的输出部分(通道 2)



17.5.6.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

6）如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读

取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

17.5.6.2. PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

1）两个 ICx 信号被映射至同一个 TIx 输入。

2）这 2 个 ICx 信号为边沿有效，但是极性相反。

3）其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。 例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

4）选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。

5）选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。

6）选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。

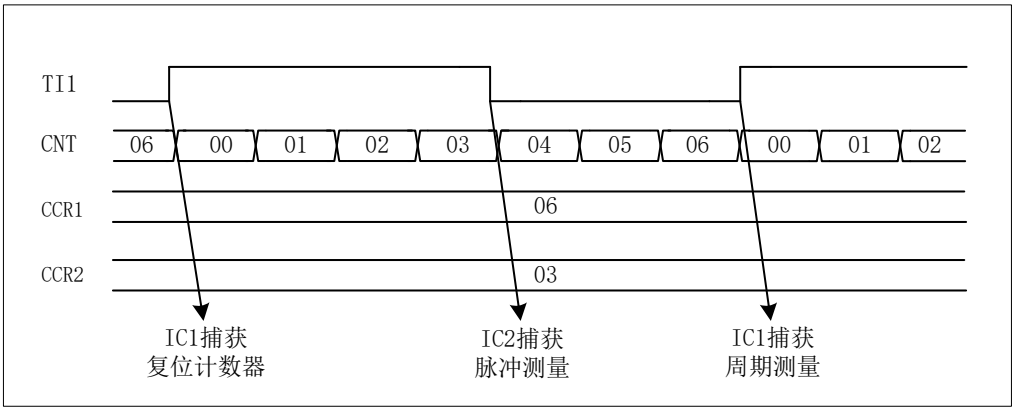
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

8）选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。

9）配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。

10）使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

图 17-13 PWM 输入模式时序



17.5.6.3. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

17.5.6.4. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

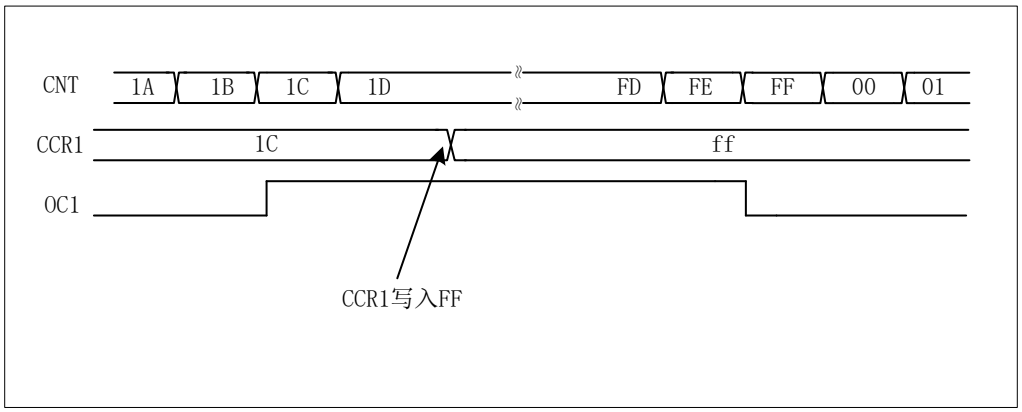
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 17-14 输出比较模式，翻转 OC1



17.5.6.5. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。 OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

17.5.6.6. 互补输出和死区插入

高级控制定时器能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。 这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

配置 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性(主输出 OCx 或互补输出 OCxN)。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，见表“带刹车功能的互补输出通道 OCx 和 OCxN 的控制位”。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

- OCx 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。
- OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。

如果延迟大于当前有效的输出宽度(OCx 或者 OCxN)，则不会产生相应的脉冲。 下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。(假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1)

图 17-15 带死区插入的互补输出

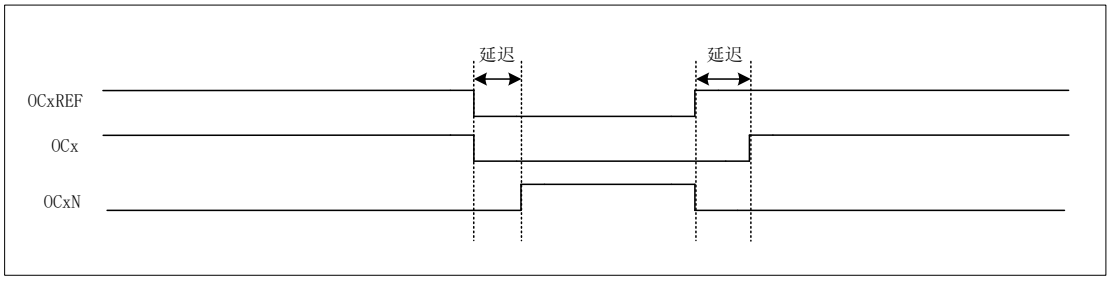


图 17-16 死区波形延迟大于负脉冲

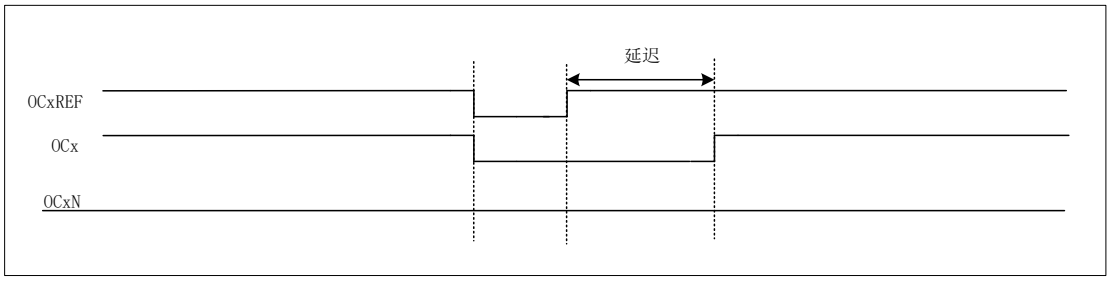
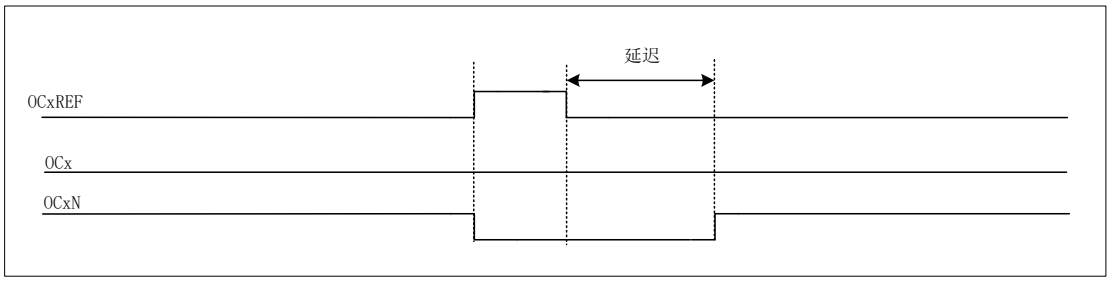


图 17-17 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。

重定向 OCxREF 到 OCx 或 OCxN

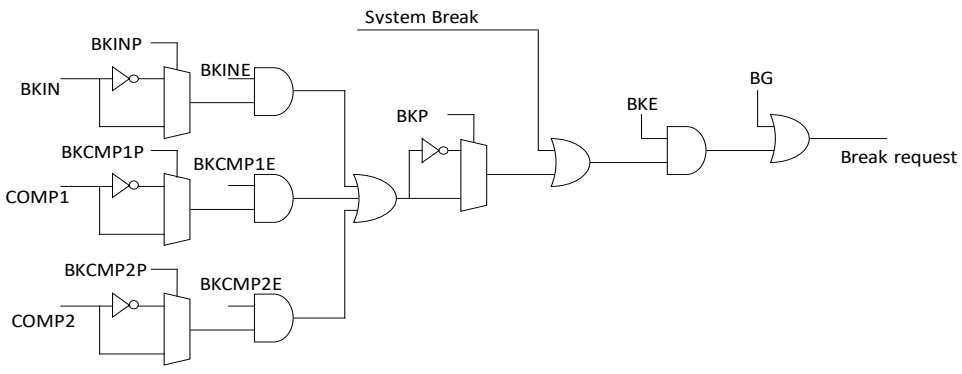
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注： 当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

17.5.6.7. 使用刹车功能

当使用刹车功能时，依据相应的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。刹车源可以选择刹车输入引脚，也可以选择比较器输出。另外，系统也可以产生刹车请求，如图所示。

图 17-18 刹车



刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生。

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和

BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的，在实际信号(作用在输出端)和同步控制位(在 TIMx_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

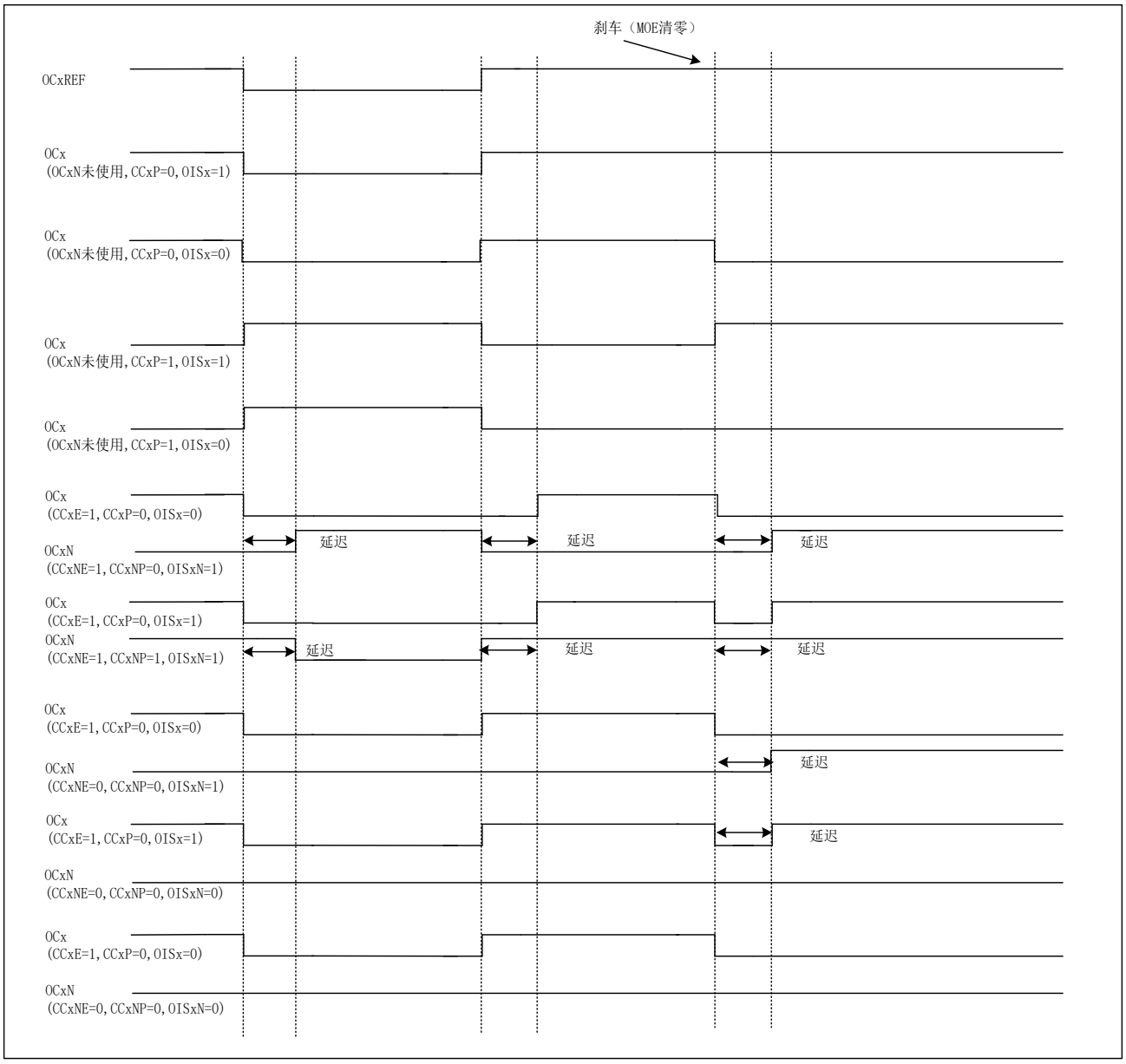
当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’ 1’ 时，则产生一个中断。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’ 1’ ；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。 除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性)。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 TIM1 和 TIM8 刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。 下图显示响应刹车的输出实例。

图 17-19 响应刹车的输出



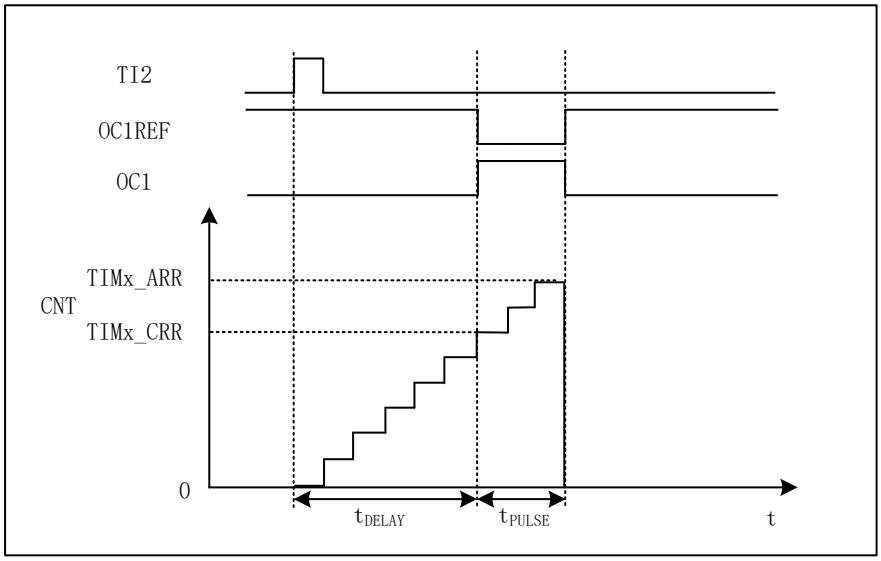
17.5.6.8. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 17-20 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

17.5.7. 定时器互连

TIMx 定时器能够在从模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

17.5.7.1. 复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR， TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

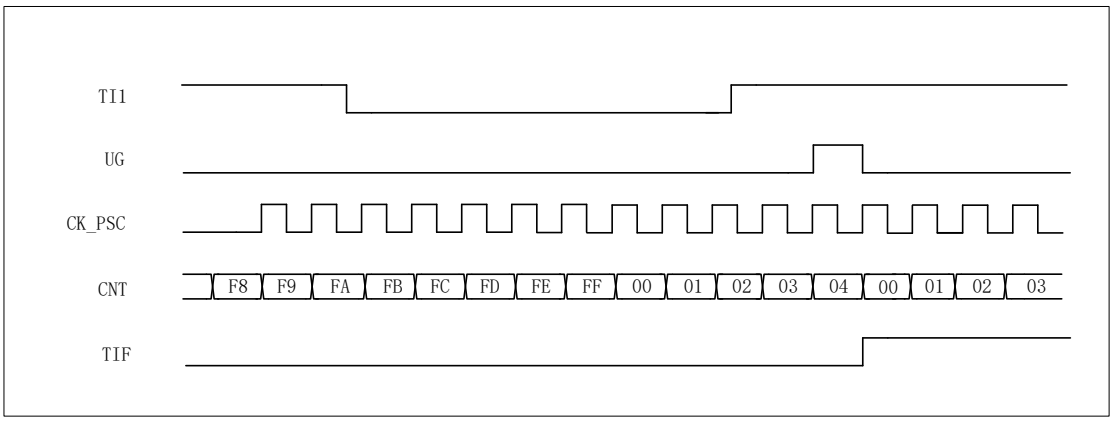
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

2）置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。 下图显示当自动重装载寄存器 TIMx_ARR=0xFF 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 17-21 复位模式下的控制电路



17.5.7.2. 门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

1）配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

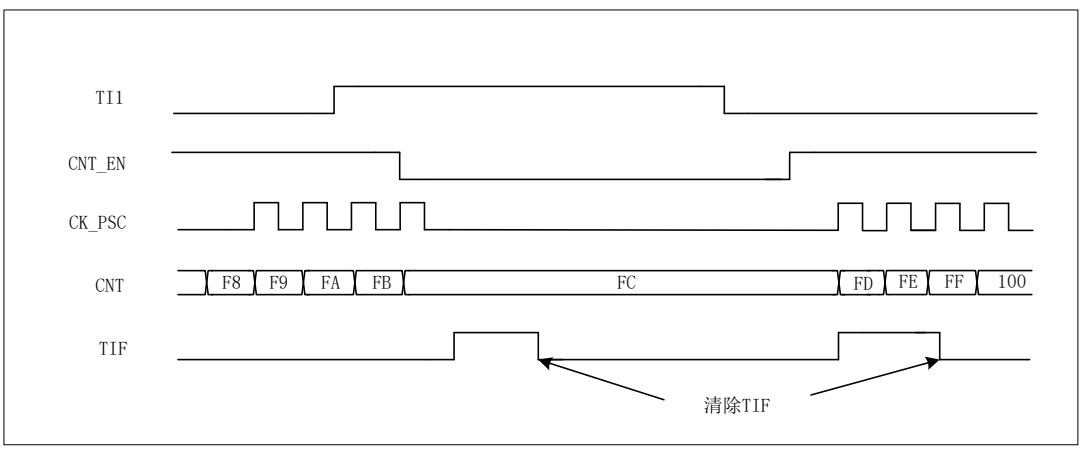
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

3）置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图 17-22 门控模式下的控制电路



17.5.7.3. 触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

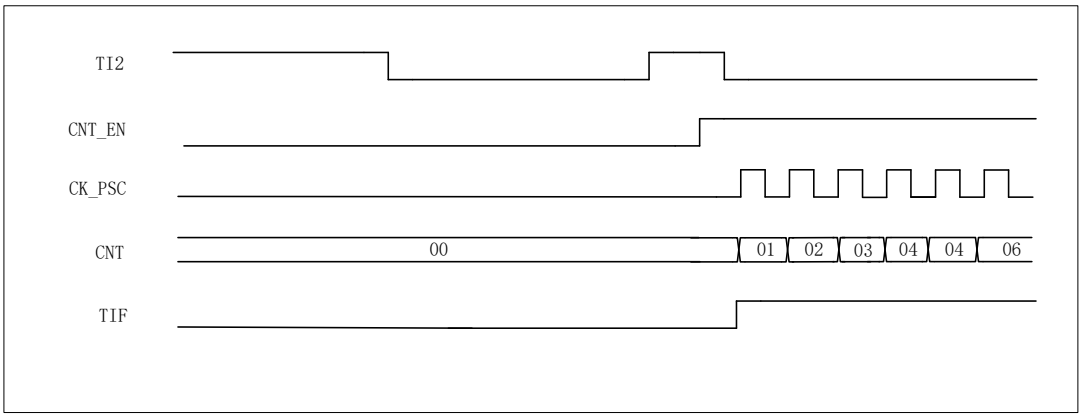
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

2）置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 17-23 触发模式下的控制电路



17.5.8. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

SingleDMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMx 会给 DMA 发送请求,等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

17.5.9. 定时器调试模式

定时器在调试时依然在运行。

17.6. TIM15/25 寄存器描述

17.6.1. 寄存器列表

TIM15 寄存器基地址：0x40014000

TIM25 寄存器基地址：0x4000A000

表 17-6 高级控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	TIMx_SMCR	TIMx 从模式控制寄存器
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	-	保留
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	TIMx_RCR	TIMx 重复计数寄存器
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	TIMx_CCR2	TIMx 捕获比较寄存器 2
0x3C	-	保留
0x40	-	保留
0x44	TIMx_BDTR	TIMx 刹车和死区控制寄存器
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x60	TIMx_AF1	TIMx 复用功能选择寄存器 1
0x64	TIMx_AF2	TIMx 复用功能选择寄存器 2
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

17.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，读始终为 0。
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟与定时器时钟（CK_INT）的分频比例。 00:tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:4	RSV	-	-	保留，读始终为 0。
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

17.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留，始终读为 0
10	OIS2	RW	0x0	输出空闲状态 2(OC2 输出)。参见 OIS1 位。
9	OIS1N	RW	0x0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
8	OIS1	RW	0x0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
7	RSV	-	-	保留，始终读为 0
6:4	MMS	RW	0x0	主模式选择（Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能– 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0x0	捕获/比较的 DMA 选择（Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0x0	捕获/比较控制更新选择（Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0

0	CCPC	RW	0x0	捕获/比较预装载控制位（Capture/compare preloaded control） 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的； 设置该位后，它们只在设置了 COM 位后被更新。

17.6.4. 从模式控制寄存器(TIMx_SMCR: 08h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	MSM	RW	0x0	主/从模式（Master/slave mode） 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS	RW	0x0	触发选择（Trigger selection） 这 3 位选择用于同步计数器的触发输入。 000：保留 100：TI1 的边沿检测器(TI1F_ED) 001：内部触发 1(ITR1) 101：滤波后的定时器输入 1(TI1FP1) 010：内部触发 2(ITR2) 110：滤波后的定时器输入 2(TI2FP2) 011：内部触发 3(ITR3) 111：保留 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	RSV	-	-	保留，始终读为 0

2:0	SMS	RW	0x0	从模式选择（Slave mode selection） 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表 17-7 TIM15 内部触发连接

从定时器	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM15	保留	TIM3	TIM16 OC1	TIM17 OC1

17.6.5. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，始终读为 0
14	TDE	RW	0x0	允许触发 DMA 请求（Trigger DMA request enable） 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13:11	RSV	-	-	保留，始终读为 0
10	CC2DE	RW	0x0	允许捕获/比较 2 的 DMA 请求（Capture/Compare 2 DMA request enable） 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。
9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable） 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable）0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。

7	BIE	RW	0x0	允许刹车中断（Break interrupt enable） 0：禁止刹车中断； 1：允许刹车中断。
6	TIE	RW	0x0	触发中断使能（Trigger interrupt enable） 0：禁止触发中断； 1：使能触发中断。
5	COMIE	RW	0x0	允许 COM 中断（COM interrupt enable） 0：禁止 COM 中断； 1：允许 COM 中断。
4:3	RSV	-	-	保留，始终读为 0
2	CC2IE	RW	0x0	允许捕获/比较 2 中断（Capture/Compare 2 interrupt enable） 0：禁止捕获/比较 2 中断； 1：允许捕获/比较 2 中断。
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable） 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable） 0：禁止更新中断； 1：允许更新中断。

17.6.6. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留，始终读为 0
10	CC2OF	W0C	0x0	捕获/比较 2 重复捕获标记（Capture/Compare 2 overcapture flag）参见 CC1OF 描述。
9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为‘1’。
8	RSV	-	-	位 8 保留，始终读为 0
7	BIF	W0C	0x0	刹车中断标记（Break interrupt flag） 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，则该位可由软件清‘0’。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。

6	TIF	W0C	0x0	触发器中断标记（Trigger interrupt flag） 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	W0C	0x0	COM 中断标记（COM interrupt flag） 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置’1’。它由软件清’0’。 0：无 COM 事件产生； 1：COM 中断等待响应。
4:3	RSV	-	-	保留，始终读为 0
2	CC2IF	W0C	0x0	捕获/比较 2 中断标记（Capture/Compare 2 interrupt flag）参考 CC1IF 描述。
1	CC1IF	W0C	0x0	捕获/比较 1 中断标记（Capture/Compare 1 interrupt flag） 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

17.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	位 15:8 保留，始终读为 0

7	BG	WO	0x0	产生刹车事件（Break generation） 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
6	TG	WO	0x0	产生触发事件（Trigger generation） 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	WO	0x0	捕获/比较事件，产生控制更新（Capture/Compare control update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
4:3	RSV	-	-	保留，始终读为 0
2	CC2G	WO	0x0	产生捕获/比较 2 事件（Capture/Compare 2 generation）参考 CC1G 描述。
1	CC1G	WO	0x0	产生捕获/比较 1 事件（Capture/Compare 1 generation） 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	WO	0x0	产生更新事件（Update generation） 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

17.6.8. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留，读始终为 0。
14:12	OC2M	RW	0x0	输出比较 2 模式（Output Compare 2 mode）

11	OC2PE	RW	0x0	输出比较 2 预装载使能（Output Compare 2 preload enable）
10	OC2FE	RW	0x0	输出比较 2 快速使能（Output Compare 2 fast enable）
9:8	CC2S	RW	0x0	捕获/比较 2 选择。(Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。 此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	RSV	-	-	保留，读始终为 0。
6:4	OC1M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。

2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:12	IC2F	RW	0x0	输入捕获 2 滤波器（Input capture 2 filter）
11:10	IC2PSC	RW	0x0	输入/捕获 2 预分频器（Input capture 2 prescaler）
9:8	CC2S	RW	0x0	捕获/比较 2 选择（Capture/Compare 2 selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。

7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

17.6.9. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	CC2NP	RW	0x0	输入/捕获 2 互补输出极性（Capture/Compare 2 complementary output polarity）参考 CC1NP 的描述。
6	RSV	-	-	保留，始终读为 0
5	CC2P	RW	0x0	输入/捕获 2 输出极性（Capture/Compare 2 output polarity）参考 CC1P 的描述。

4	CC2E	RW	0x0	输入/捕获 2 输出使能（Capture/Compare 2 output enable）参考 CC1E 的描述。
3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效； 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效； 1：OC1 低电平有效。 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表 17-8 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	OCx 输出状态	OCxN 输出状态

1	X	0	0	0	输出禁止（与定时器断开） OCx=0，OCx_EN=0	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0
		0	0	1	输出禁止（与定时器断开） OCx=0，OCx_EN=0	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP OCx_EN=1	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
		1	0	1	关闭状态(输出使能且为无效电平) OCx=CCxP，OCx_EN=1	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP， OCxN_EN=1
		1	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP， OCxN_EN=1	关闭状态(输出使能且为无效电平) OCxN=CCxNP，OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
0	0	X	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	0		0	1	输出禁止（与定时器断开） 异步地：OCx=CCxP，OCx_EN=0，OCxN=CCxNP，OCxN_EN=0， 若时钟存在：经过一个死区时间后，OCx=OISx，OCxN=OISx， 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		1	0		
	0		1	1		
	1		0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	1		0	1	关闭状态（输出使能且为无效电平） 异步地：OCx=CCxP，OCx_EN=1，OCxN=CCxNP，OCxN_EN=1， 若时钟存在：经过一个死区时间后 OCx=OISx，OCxN=OISxN， 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		1	0		
	1		1	1		

17.6.10. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value）

17.6.11. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’ 或被工作在复位模式的从控制器清’0’

17.6.12. 自动重装载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

17.6.13. 重复计数寄存器(TIMx_RCR: 30h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	位 15:8 保留，始终读为 0
7:0	REP	RW	0x0	重复计数器的值（Repetition counter value） 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；

17.6.14. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0

15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

17.6.15. 捕获/比较寄存器 2(TIMx_CCR2: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CCR2	RW	0x0	捕获/比较通道 2 的值（Capture/Compare 2 value） 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

17.6.16. 刹车和死区寄存器(TIMx_BDTR: 44h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	MOE	RW	0x0	主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。
14	AOE	RW	0x0	自动输出使能（Automatic output enable） 0：MOE 只能被软件置’1’； 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。
13	BKP	RW	0x0	刹车输入极性（Break polarity） 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。

12	BKE	RW	0x0	刹车功能使能（Break enable） 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0x0	运行模式下“关闭状态”选择（Off-state selection for Run mode） 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0x0	空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	0x0	锁定设置（Lock configuration）该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位，TIMx_AF1 所有位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	0x0	死区发生器设置（Dead-time generator setup） 这些位定义了插入互补输出之间的死区持续时间。 假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg，Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg，Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg，Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位。

17.6.17. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0
12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIM2_CR1 - 如果 DBL=7，DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0
4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

17.6.18. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses） 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

17.6.19. 复用功能选择寄存器(TIMx_AF1: 60h)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留，始终为 0。
13	BKCMP4P	RW	0x0	比较器 4 输入极性控制 0：输入高电平有效 1：输入低电平有效
12	BKCMP3P	RW	0x0	比较器 3 输入极性控制 0：输入高电平有效 1：输入低电平有效
11	BKCMP2P	RW	0x0	比较器 2 输入极性控制 0：输入高电平有效 1：输入低电平有效
10	BKCMP1P	RW	0x0	比较器 1 输入极性控制 0：输入高电平有效 1：输入低电平有效
9	BKINP	RW	0x0	刹车输入极性控制 0：输入高电平有效 1：输入低电平有效
8:5	RSV	-	-	保留，始终为 0。
4	BKCMP4E	RW	0x0	比较器 4 输入使能控制 0：禁止 1：使能
3	BKCMP3E	RW	0x0	比较器 3 输入使能控制 0：禁止 1：使能
2	BKCMP2E	RW	0x0	比较器 2 输入使能控制 0：禁止 1：使能
1	BKCMP1E	RW	0x0	比较器 1 输入使能控制 0：禁止 1：使能
0	BKINE	RW	0x0	刹车输入使能控制 0：禁止 1：使能

17.6.20. 复用功能选择寄存器 2(TIMx_AF2: 64h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留，始终读为 0

18:16	OCRSEL[2:0]	RW	0x0	ocref_clr 源选择 这些位选择 ocref_clr 输入源。 000：tim_ocref_clr0 001：tim_ocref_clr1 010：tim_ocref_clr2 011：tim_ocref_clr3 100：tim_ocref_clr4 101：tim_ocref_clr5 110：tim_ocref_clr6 111：tim_ocref_clr7 输入源请参看 TIMx 输入映射章节
15:0	RSV	-	-	保留，始终为 0。

17.6.21. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留，始终为 0。
11:8	TI2SEL	RW	0x0	TI2 输入选择 0000：tim_ti2_in0 0001：tim_ti2_in1 0010：tim_ti2_in2 0011：tim_ti2_in3 … 1111：tim_ti2_in15 输入源请参看 TIMx 输入映射章节
7:4	RSV	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：tim_ti1_in0 0001：tim_ti1_in1 0010：tim_ti1_in2 0011：tim_ti1_in3 … 1111：tim_ti1_in15 输入源请参看 TIMx 输入映射章节

17.6.22. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0

6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5	COMBE	RW	0x0	COM 事件的 DMA 请求类型 0：Single； 1：Burst；
4:3	RSV	-	-	保留，始终读为 0
2	CC2BE	RW	0x0	捕获/比较 2 事件的 DMA 请求类型 0：Single； 1：Burst；
1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；
0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

18. 通用定时器（TIM16/17/18/19）

18.1. 概述

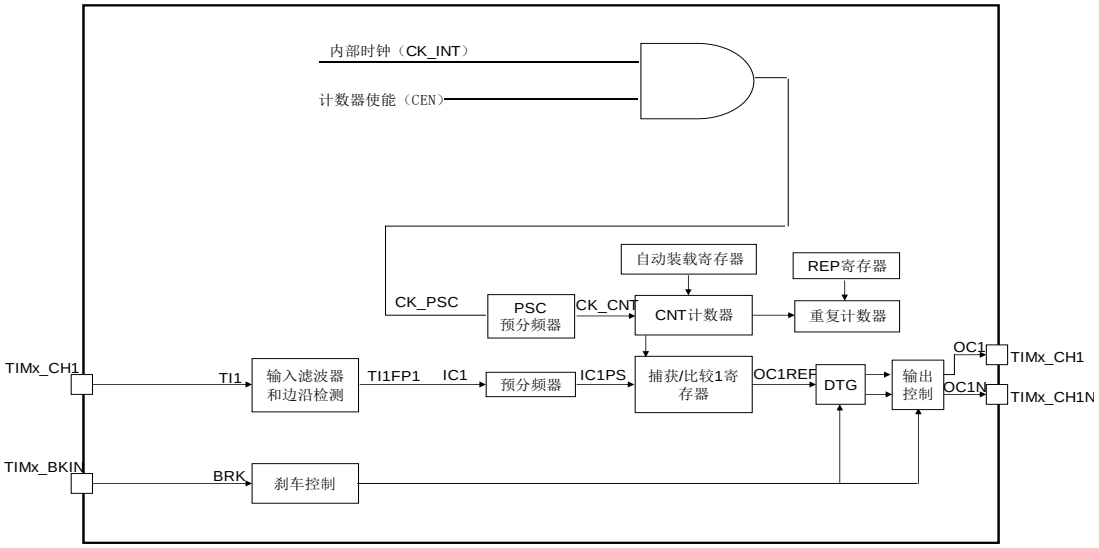
通用定时器（TIM16/17/18/19）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。 使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

18.2. 主要特性

- 16 位向上自动装载计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 1 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 触发输入作为外部时钟

18.3. 结构框图

图 18-1 通用定时器框图



18.4. TIMx 输入映射

表 18-1 TIMx 通道 1 输入

TI1SEL[3:0]	TIM16 源	TIM17 源
0000	tim16_ch1	tim17_ch1
0001	CLKOUT	CLKOUT
0010	XTH_DIV32	USB_SOF
0011	RTC_WAKEUP	RTC_WAKEUP
0100	RCL	RCL
保留	-	-

表 18-2 TIM1 OCREF_CLR 输入

OCRSEL[2:0]	TIM16 源	TIM17 源
0000	comp1_out	comp1_out
0001	comp2_out	comp2_out
0010	comp3_out	comp3_out
0011	comp4_out	comp4_out
保留	-	-

18.5. 功能描述

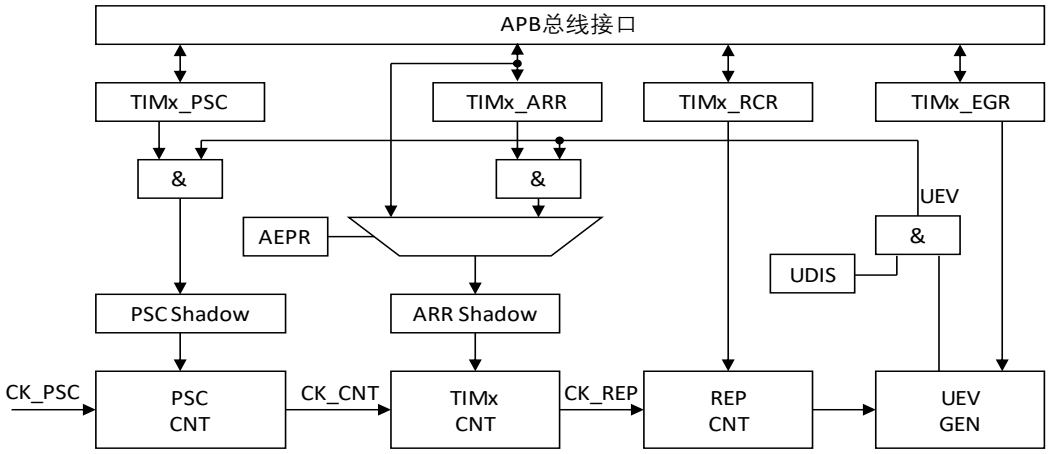
18.5.1. 计数单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计

数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。 计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。 时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 自动装载寄存器（TIMx_ARR)
- 预分频器寄存器（TIMx_PSC)
- 重复次数寄存器（TIMx_RCR)

图 18-2 计数单元的结构



如上图所示，自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。 计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

通用定时器 TIM16/17 支持一种计数模式：

- 向上计数模式

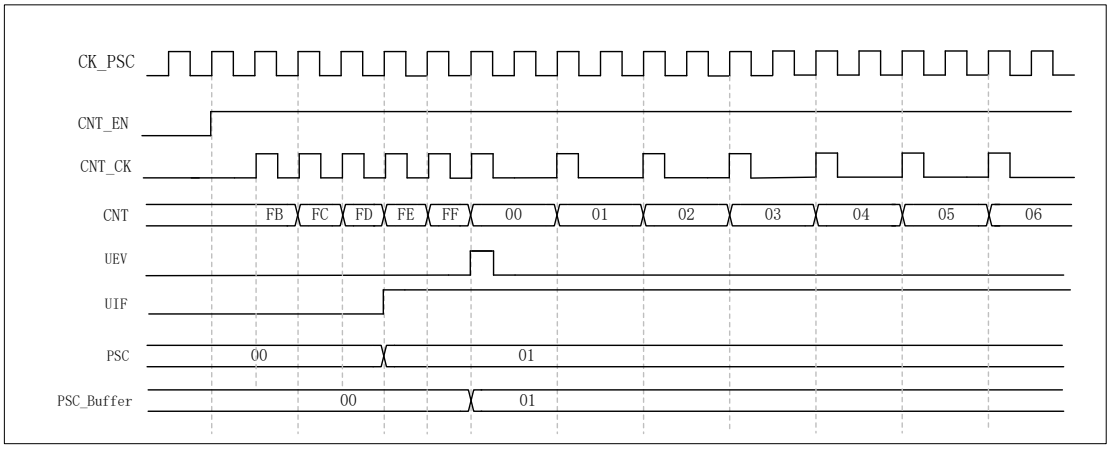
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

18.5.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

图 18-3 当预分频器的参数从 1 变到 2 时，计数器的时序图



18.5.3. 时钟源选择

计数器的时钟由内部时钟(CK_INT)提供。TIMx_CR1 寄存器的 CEN 位和 TIMx_EGR 寄存器的 UG 位是实际的控制位，(除了 UG 位被自动清除外)只能通过软件改变它们。一旦置 CEN 位为’1’，内部时钟即向预分频器提供时钟。

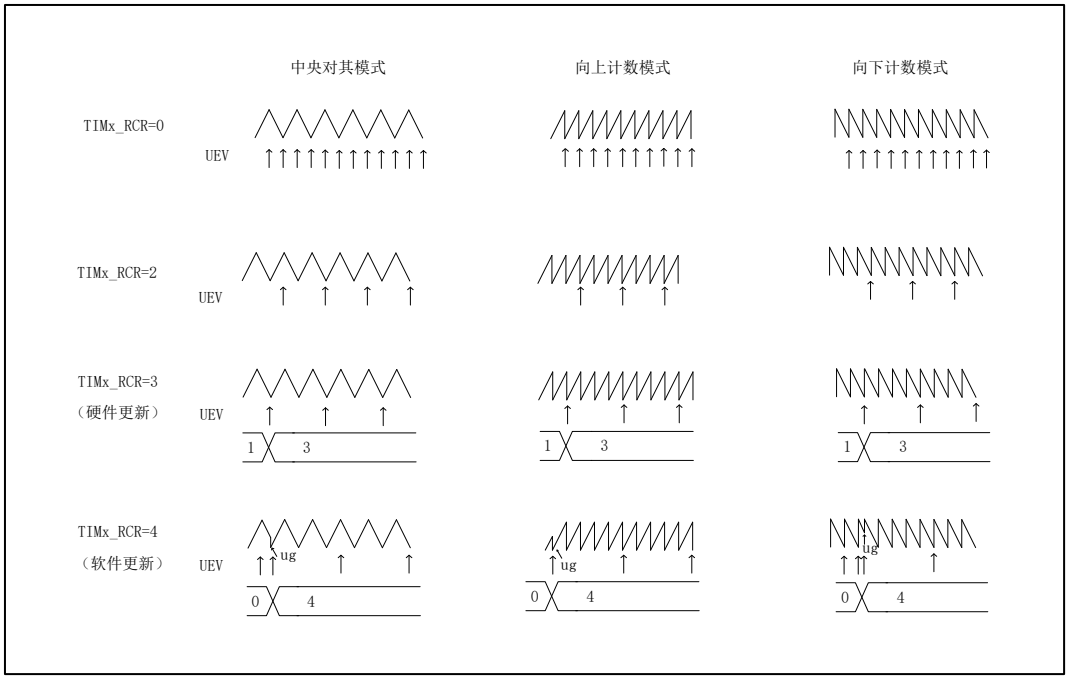
18.5.4. 重复计数器

重复计数器是一个 8 位的递减计数器，更新事件 UEV 只能在重复计数器计数达到 0 时产生。重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

图 18-4 重复计数器溢出事件

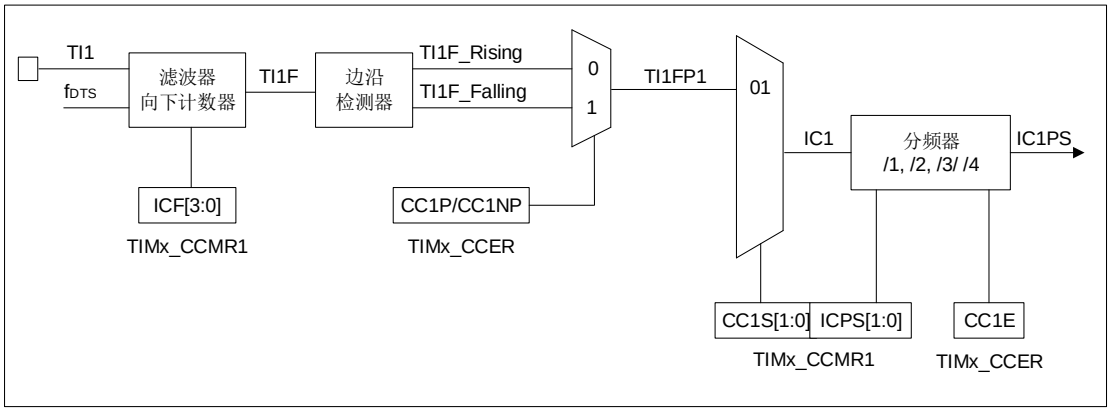


18.5.5. 捕获比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。以下 3 张图示是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

图 18-5 捕获/比较通道(如：通道 1 输入部分)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。 在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。 在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

图 18-6 捕获/比较通道 1 的主电路

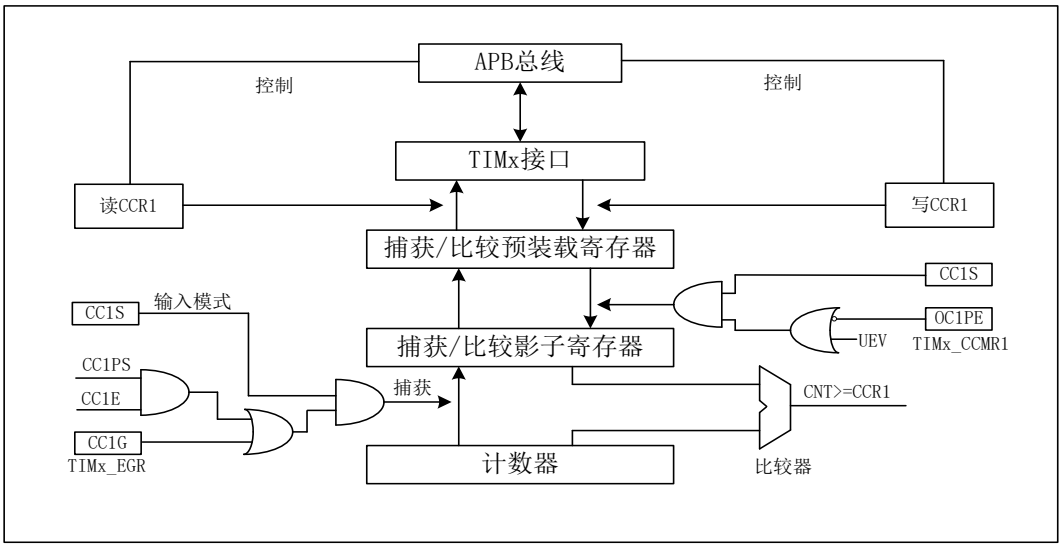
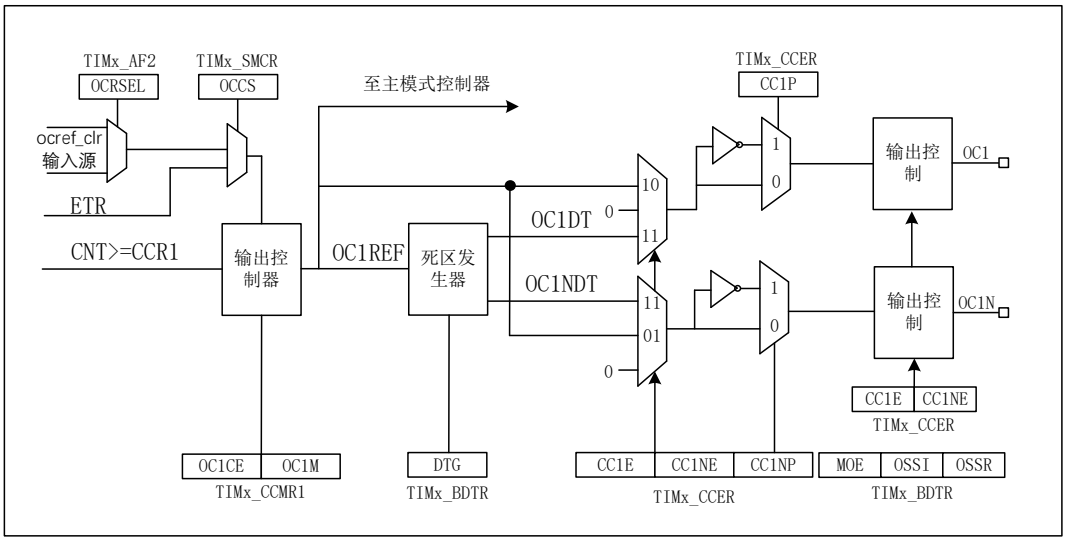


图 18-7 捕获/比较通道的输出部分(通道 1)



18.5.5.1. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

1）选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。

2）根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

3）选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。

4）配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。

5）设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。

6）如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

1）产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。

2）CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

3）如设置了 CC1IE 位，则会产生一个中断。

4）如设置了 CC1DE 位，则还会产生一个 DMA 请求。 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

5）为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

18.5.5.2. 强制输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。 置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

1）置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

18.5.5.3. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1）将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。

2）设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。

3）若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。

4）若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1）选择计数器时钟(内部，外部，预分频器)。

2）将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3）如果要产生一个中断请求，设置 CCxIE 位。

4）选择输出模式，例如：

a）要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011

b）置 OCxPE = 0 禁用预装载寄存器

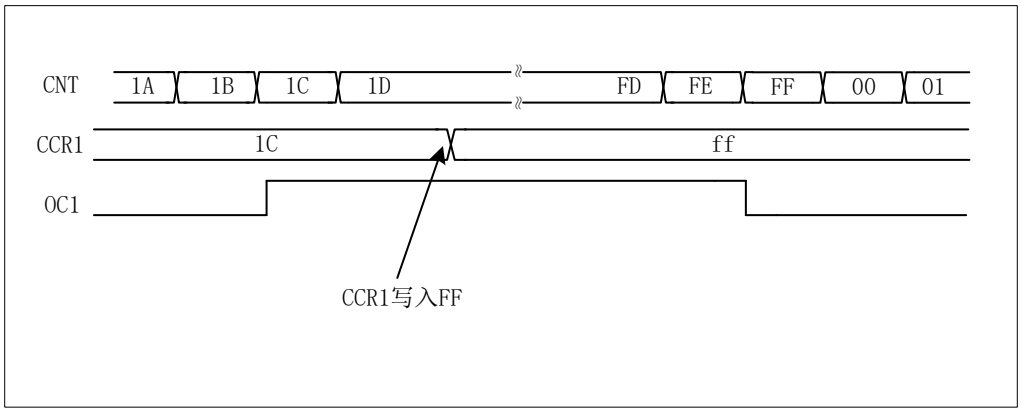
c）置 CCxP = 0 选择极性为高电平有效

d）置 CCxE = 1 使能输出。

5）设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。

图 18-8 输出比较模式，翻转 OC1



18.5.5.4. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。 OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

18.5.5.5. 互补输出和死区插入

高级控制定时器能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。 这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

配置 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性(主输出 OCx 或互补输出 OCxN)。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见表 18-4。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

- OCx 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。
- OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。

如果延迟大于当前有效的输出宽度(OCx 或者 OCxN)，则不会产生相应的脉冲。 下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。(假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1)

图 18-9 带死区插入的互补输出

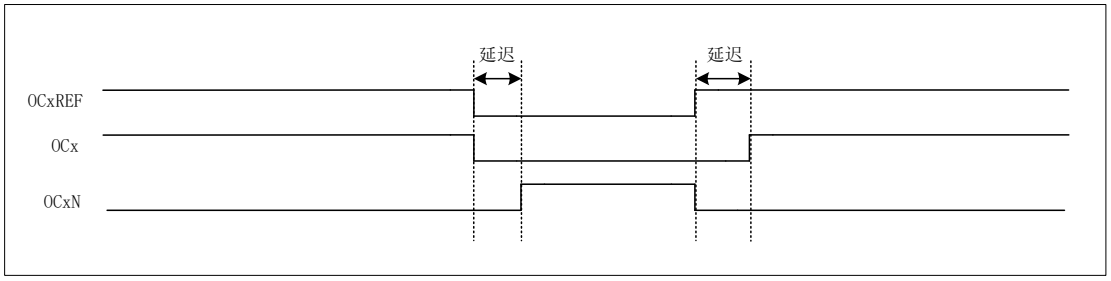


图 18-10 死区波形延迟大于负脉冲

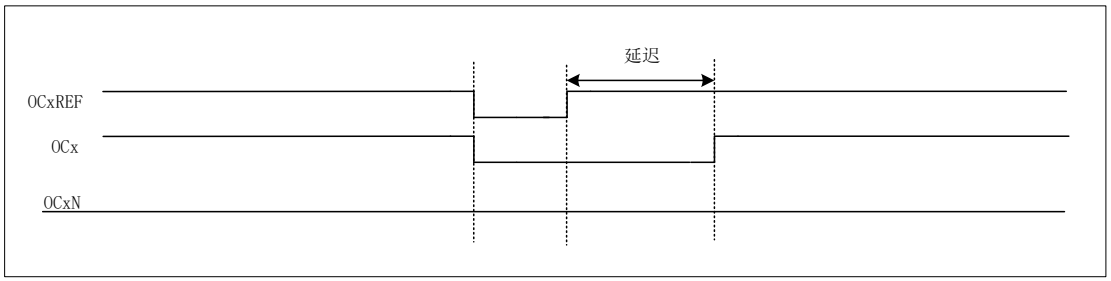
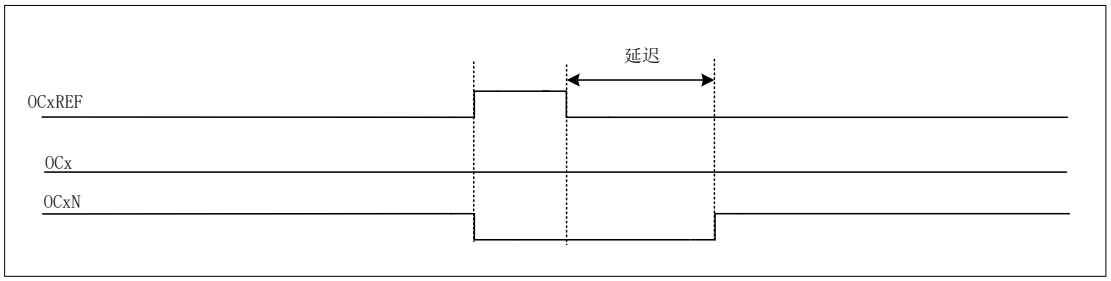


图 18-11 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。

重定向 OCxREF 到 OCx 或 OCxN

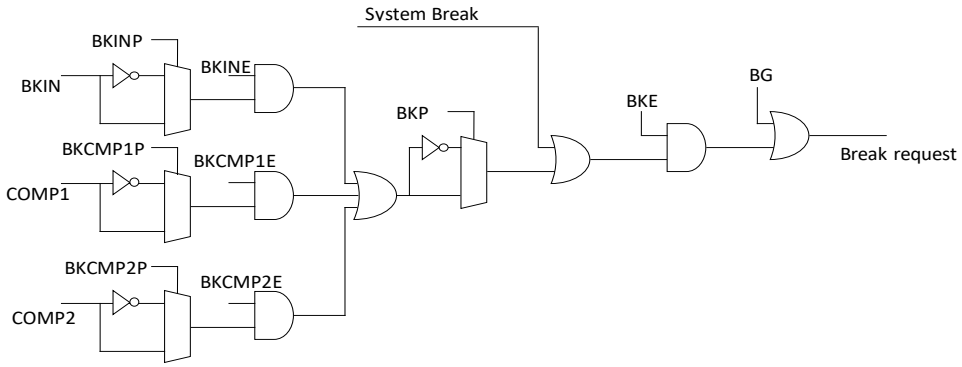
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注： 当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

18.5.5.6. 使用刹车功能

当使用刹车功能时，依据相应的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。刹车源可以选择刹车输入引脚，也可以选择比较器输出。另外，系统也可以产生刹车请求，如图所示。

图 18-12 刹车



刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生。

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的，在实际信号(作用在输出端)和同步控制位(在 TIMx_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

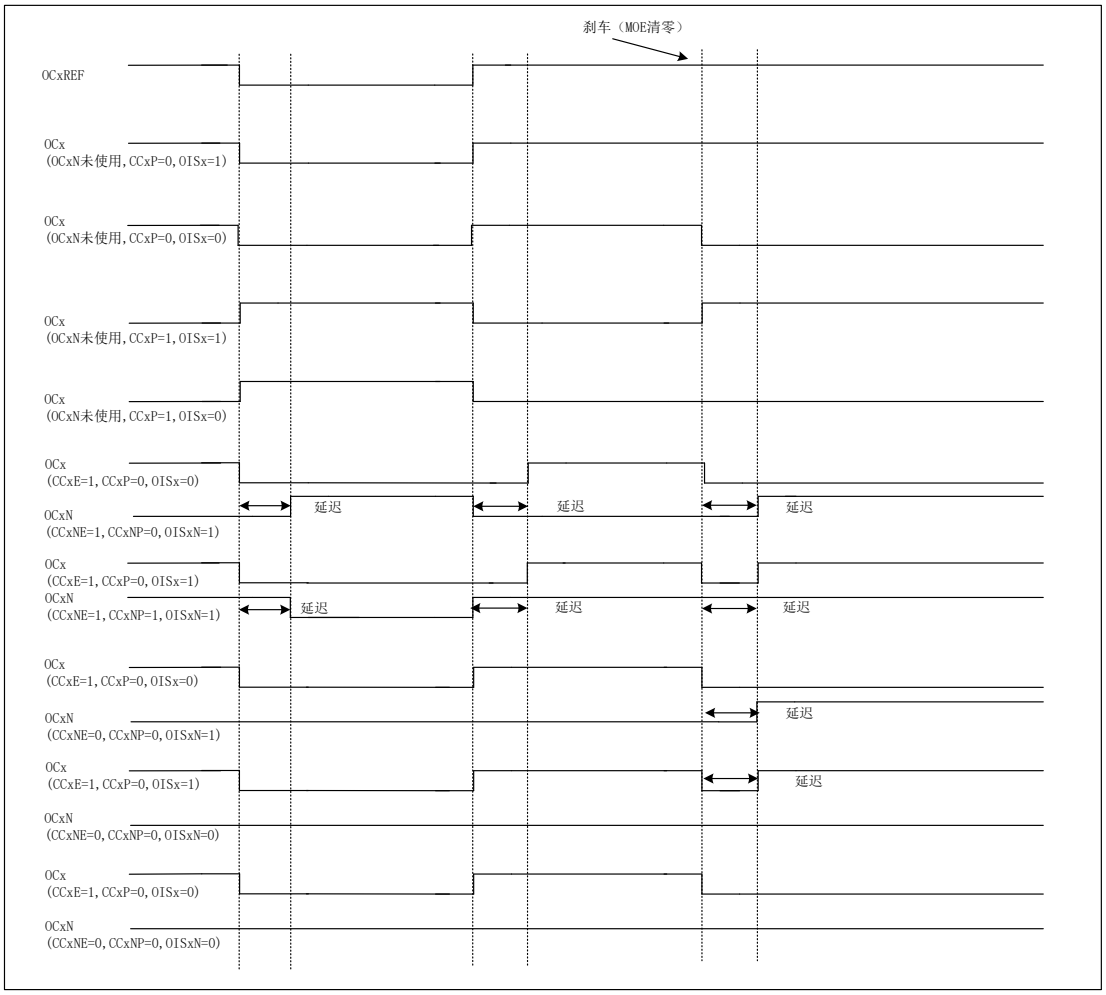
当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’1’时，则产生一个中断。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。 除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性)。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 13.4.18 节 TIM1 和 TIM8 刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。 下图显示响应刹车的输出实例。

图 18-13 响应刹车的输出



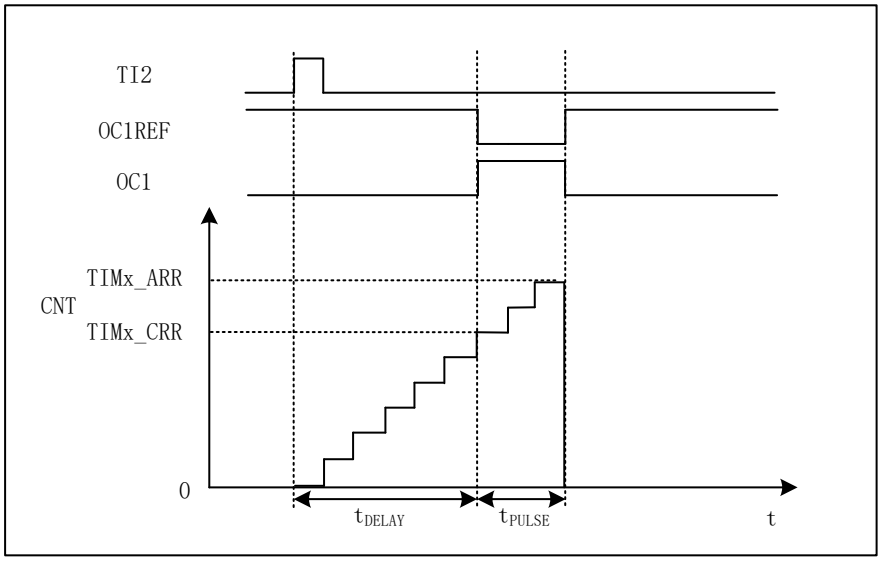
18.5.5.7. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。 可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

图 18-14 单脉冲模式的例子



例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。 假定 TI2FP2 作为触发 1:

1）置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

2）置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

3）置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

4）置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- tDELAY 由 TIMx_CCR1 寄存器中的值定义。
- tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。 因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

18.5.6. 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。DMA 访问定时器有两种方式：非 burst 和 burst 方式。

SingleDMA 访问：

先配置 TIMx_DBER 中对应的 single 位，使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMx 会给 DMA 发送请求,等待 DMA 发送清除信号后一次传输完成。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

Burst DMA 访问：

有三个跟定时器 DMA 模式相关的寄存器：TIMx_DCR、TIMx_DBER 和 TIMx_DMAR。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。使用 burst DMA 访问时，先配置 TIMx_DBER 中对应的 burst 位，TIMx_DCR 中的 DBA 和 DBL。当中断事件发生，TIMx 会给 DMA 发送请求。DMA 配置成 M2P 模

式，PADDR 是 TIMx_DMAR 寄存器地址，DMA 就会访问 TIMx_DMAR 寄存器。实际上，TIMx_DMAR 寄存器只是一个缓冲，定时器会将 TIMx_DMAR 映射到一个内部寄存器，这个内部寄存器由 TIMx_DCR 寄存器中的 DBA 来指定,例如 DBA=2,则内部寄存器为 TIMx_SMCR 寄存器。如果 TIMx_DCR 寄存器的 DBL 比特值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMx_DCR 寄存器的 DBL 比特值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMx_DMAR 寄存器的访问会映射到访问定时器的 DBA+0x4, DBA+0x8, DBA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送（DBL+1）次请求。

如果再来 1 次 DMA 请求事件，TIMx 将会重复上面的过程。

18.5.7. 定时器调试模式

定时器在调试时依然在运行。

18.6. TIM16/17/18/19 寄存器描述

18.6.1. 寄存器列表

TIM16 寄存器基地址：0x40014400

TIM17 寄存器基地址：0x40014800

TIM18 寄存器基地址：0x40019000

TIM19 寄存器基地址：0x40019400

表 18-3 高级控制定时器的寄存器映射

偏移	名称	描述
0x00	TIMx_CR1	TIMx 控制寄存器 1
0x04	TIMx_CR2	TIMx 控制寄存器 2
0x08	-	保留
0x0C	TIMx_DIER	TIMx DMA/中断使能寄存器
0x10	TIMx_SR	TIMx 状态寄存器
0x14	TIMx_EGR	TIMx 事件产生寄存器
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1
0x1C	-	保留
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器
0x24	TIMx_CNT	TIMx 计数器
0x28	TIMx_PSC	TIMx 预分频器
0x2C	TIMx_ARR	TIMx 自动装载寄存器
0x30	TIMx_RCR	TIMx 重复计数寄存器
0x34	TIMx_CCR1	TIMx 捕获比较寄存器 1
0x38	-	保留

0x3C	-	保留
0x40	-	保留
0x44	TIMx_BDTR	TIMx 刹车和死区控制寄存器
0x48	TIMx_DCR	TIMx DMA 控制寄存器
0x4C	TIMx_DMAR	TIMx 连续模式的 DMA 地址
0x60	TIMx_AF1	TIMx 复用功能选择寄存器 1
0x64	TIMx_AF2	TIMx 复用功能选择寄存器 2
0x68	TIMx_TISEL	TIMx 输入选择寄存器
0x6C	TIMx_DBER	TIMx DMA 请求类型选择寄存器

18.6.2. 控制寄存器 1(TIMx_CR1: 00h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，读始终为 0。
9:8	CKD	RW	0x0	时钟分频因子 死区发生器和数字滤波器所用的采样时钟与定时器时钟（CK_INT）的分频比例。 00:tDTS=tCK_INT 01: tDTS=2 x tCK_INT 10: tDTS=4 x tCK_INT 11:保留
7	ARPE	RW	0x0	自动重装载预装载允许位 0:TIMx_ARR 寄存器没有缓冲 1:TIMx_ARR 寄存器被装入缓冲器
6:4	RSV	-	-	保留，读始终为 0。
3	OPM	RW	0x0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0x0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。

1	UDIS	RW	0x0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

18.6.3. 控制寄存器 2(TIMx_CR2: 04h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9	OIS1N	RW	0x0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
8	OIS1	RW	0x0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
7:4	RSV	-	-	保留，始终读为 0
3	CCDS	RW	0x0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0x0	捕获/比较控制更新选择 (Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	RSV	-	-	保留，始终读为 0
0	CCPC	RW	0x0	捕获/比较预装载控制位 (Capture/compare preloaded control) 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的； 设置该位后，它们只在设置了 COM 位后被更新。

18.6.4. DMA/中断使能寄存器(TIMx_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9	CC1DE	RW	0x0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	0x0	允许更新的 DMA 请求（Update DMA request enable) 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	BIE	RW	0x0	允许刹车中断（Break interrupt enable) 0：禁止刹车中断； 1：允许刹车中断。
6	RSV	-	-	保留，始终读为 0
5	COMIE	RW	0x0	允许 COM 中断（COM interrupt enable) 0：禁止 COM 中断； 1：允许 COM 中断。
4:2	RSV	-	-	保留，始终读为 0
1	CC1IE	RW	0x0	允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断。
0	UIE	RW	0x0	允许更新中断（Update interrupt enable) 0：禁止更新中断； 1：允许更新中断。

18.6.5. 状态寄存器(TIMx_SR: 10h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留，始终读为 0
9	CC1OF	W0C	0x0	捕获/比较 1 重复捕获标记（Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
8	RSV	-	-	位 8 保留，始终读为 0
7	BIF	W0C	0x0	刹车中断标记（Break interrupt flag) 一旦刹车输入有效，由硬件对该位置’1’。如果刹车输入无效，则该位可由软件清’0’。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。

6	RSV	-	-	保留，始终读为 0
5	COMIF	W0C	0x0	COM 中断标记（COM interrupt flag） 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置’1’。它由软件清’0’。 0：无 COM 事件产生； 1：COM 中断等待响应。
4:2	RSV	-	-	保留，始终读为 0
1	CC1IF	W0C	0x0	捕获/比较 1 中断标记（Capture/Compare 1 interrupt flag） 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	W0C	0x0	更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

18.6.6. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留，始终读为 0
7	BG	WO	0x0	产生刹车事件（Break generation） 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
6	RSV	-	-	保留，始终读为 0

5	COMG	WO	0x0	捕获/比较事件，产生控制更新（Capture/Compare control update generation) 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
4:2	RSV	-	-	保留，始终读为 0
1	CC1G	WO	0x0	产生捕获/比较 1 事件（Capture/Compare 1 generation) 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	WO	0x0	产生更新事件（Update generation) 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’（但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

18.6.7. 捕获/比较模式寄存器 1(TIMx_CCMR1: 18h)

通道可用于输入(捕获模式)或输出(比较模式)，通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0

6:4	OC1M	RW	0x0	输出比较 1 模式（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0x0	输出比较 1 预装载使能（Output Compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0x0	输出比较 1 快速使能（Output Compare 1 fast enable） 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S	RW	0x0	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 其它：保留 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式：

位域	名称	属性	复位值	描述
15:8	RSV	-	-	保留，始终读为 0
7:4	IC1F	RW	0x0	输入捕获 1 滤波器（Input capture 1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC	RW	0x0	输入/捕获 1 预分频器（Input capture 1 prescaler） 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	0x0	捕获/比较 1 选择（Capture/Compare 1 Selection） 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 其它：保留 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

18.6.8. 捕获/比较使能寄存器(TIMx_CCER: 20h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留，始终读为 0

3	CC1NP	RW	0x0	输入/捕获 1 互补输出极性（Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效； 1：OC1N 低电平有效。 CC1 通道配置为输入：该位与 CC1P 结合使用以定义 TI1FP1 和 TI2FP1 的极性。参考 CC1P 的描述。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0x0	输入/捕获 1 互补输出使能（Capture/Compare 1 complementary output enable) 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	0x0	输入/捕获 1 输出 极性（Capture/Compare 1 output polarity) CC1 通道配置为输出： 0：OC1 高电平有效； 1：OC1 低电平有效。 CC1 通道配置为输入： CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的有效极性，用于触发或捕获操作。 00：不反相/上升沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿，在门控模式或编码器模式下触发操作，TIxFP1 不反相。 01：反向/下降沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的下降沿，在门控模式或编码器模式下触发操作，TIxFP1 反相。 10：保留，不使用此配置。 11：不反相/双边沿。在复位、外部时钟或触发模式下，捕获或触发发生在 TIxFP1 的上升沿和下降沿，在门控模式下触发操作，TIxFP1 不反相（此配置不得在编码器模式下使用） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0x0	输入/捕获 1 输出使能（Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表 18-4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止（与定时器断开） OCx=0，OCx_EN=0	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0

		0	0	1	输出禁止（与定时器断开） OCx=0，OCx_EN=0	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP OCx_EN=1	输出禁止（与定时器断开） OCxN=0，OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
		1	0	1	关闭状态(输出使能且为无效电平) OCx=CCxP，OCx_EN=1	OCxREF + 极性， OCxN=OCxREF ⊕ CCxNP， OCxN_EN=1
		1	1	0	OCxREF + 极性， OCx=OCxREF ⊕ CCxP， OCxN_EN=1	关闭状态(输出使能且为无效电平) OCxN=CCxNP，OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区， OCxN_EN=1
0	0	X	0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	0		0	1	输出禁止（与定时器断开） 异步地：OCx=CCxP，OCx_EN=0，OCxN=CCxNP，OCxN_EN=0， 若时钟存在：经过一个死区时间后，OCx=OISx，OCxN=OISx， 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		1	0		
	0		1	1		
	1		0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP，OCxN_EN=0
	1		0	1	关闭状态（输出使能且为无效电平） 异步地：OCx=CCxP，OCx_EN=1，OCxN=CCxNP，OCxN_EN=1， 若时钟存在：经过一个死区时间后 OCx=OISx，OCxN=OISxN， 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		1	0		
	1		1	1		

18.6.9. 计数器(TIMx_CNT: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	CNT	RW	0x0	计数器的值（Counter value）

18.6.10. 预分频器(TIMx_PSC: 28h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	PSC	RW	0x0	预分频器的值（Prescaler value） 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’

18.6.11. 自动重装载寄存器(TIMx_ARR: 2Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	ARR	RW	0x0	自动重装载的值（Prescaler value） ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

18.6.12. 重复计数寄存器(TIMx_RCR: 30h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:8	RSV	-	-	位 15:8 保留，始终读为 0
7:0	REP	RW	0x0	重复计数器的值（Repetition counter value） 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；

18.6.13. 捕获/比较寄存器 1(TIMx_CCR1: 34h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0

15:0	CCR1	RW	0x0	捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

18.6.14. 刹车和死区寄存器(TIMx_BDTR: 44h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15	MOE	RW	0x0	主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。
14	AOE	RW	0x0	自动输出使能（Automatic output enable） 0：MOE 只能被软件置’1’； 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。
13	BKP	RW	0x0	刹车输入极性（Break polarity） 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0x0	刹车功能使能（Break enable） 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0x0	运行模式下“关闭状态”选择（Off-state selection for Run mode） 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。

10	OSSI	RW	0x0	空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	0x0	锁定设置（Lock configuration）该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	0x0	死区发生器设置（Dead-time generator setup） 这些位定义了插入互补输出之间的死区持续时间。 假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg，Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg，Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg，Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位。

18.6.15. DMA 控制寄存器(TIMx_DCR: 48h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	位 31:13 保留，始终读为 0

12:8	DBL	RW	0x0	DMA 连续传送长度（DMA burst length） 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIM2_CR1 - 如果 DBL=7，DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出：(TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中 (TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节， 以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	RSV	-	-	位 7:5 保留，始终读为 0
4:0	DBA	RW	0x0	这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ...

18.6.16. 连续模式的 DMA 地址(TIMx_DMAR: 4Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留，始终读为 0
15:0	DMAB	RW	0x0	DMA 连续传送寄存器（DMA register for burst accesses） 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作：TIMx_CR1 地址 + (DBA + DMA 索引)*4，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

18.6.17. 复用功能选择寄存器(TIMx_AF1: 60h)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留，始终为 0。

13	BKCMP4P	RW	0x0	比较器 4 输入极性控制 0：输入高电平有效 1：输入低电平有效
12	BKCMP3P	RW	0x0	比较器 3 输入极性控制 0：输入高电平有效 1：输入低电平有效
11	BKCMP2P	RW	0x0	比较器 2 输入极性控制 0：输入高电平有效 1：输入低电平有效
10	BKCMP1P	RW	0x0	比较器 1 输入极性控制 0：输入高电平有效 1：输入低电平有效
9	BKINP	RW	0x0	刹车输入极性控制 0：输入高电平有效 1：输入低电平有效
8:5	RSV	-	-	保留，始终为 0。
4	BKCMP4E	RW	0x0	比较器 4 输入使能控制 0：禁止 1：使能
3	BKCMP3E	RW	0x0	比较器 3 输入使能控制 0：禁止 1：使能
2	BKCMP2E	RW	0x0	比较器 2 输入使能控制 0：禁止 1：使能
1	BKCMP1E	RW	0x0	比较器 1 输入使能控制 0：禁止 1：使能
0	BKINE	RW	0x0	刹车输入使能控制 0：禁止 1：使能

18.6.18. 复用功能选择寄存器 2(TIMx_AF2: 64h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留，始终读为 0

18:16	OCRSEL[2:0]	RW	0x0	ocref_clr 源选择 这些位选择 ocref_clr 输入源。 000：tim_ocref_clr0 001：tim_ocref_clr1 010：tim_ocref_clr2 011：tim_ocref_clr3 100：tim_ocref_clr4 101：tim_ocref_clr5 110：tim_ocref_clr6 111：tim_ocref_clr7 输入源请参看 TIMx 输入映射章节
15:0	RSV	-	-	保留，始终为 0。

18.6.19. 输入选择寄存器(TIMx_TISEL: 68h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留，始终为 0。
3:0	TI1SEL	RW	0x0	TI1 输入选择 0000：tim_ti1_in0 0001：tim_ti1_in1 0010：tim_ti1_in2 0011：tim_ti1_in3 … 1111：tim_ti1_in15 输入源请参看 TIMx 输入映射章节

18.6.20. DMA 请求类型选择寄存器(TIMx_DBER: 6Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留，始终读为 0
6	TBE	RW	0x0	触发事件的 DMA 请求类型 0：Single； 1：Burst；
5	COMBE	RW	0x0	COM 事件的 DMA 请求类型 0：Single； 1：Burst；
4:2	RSV	-	-	保留，始终读为 0
1	CC1BE	RW	0x0	捕获/比较 1 事件的 DMA 请求类型 0：Single； 1：Burst；

0	UBE	RW	0x0	更新事件的 DMA 请求类型 0：Single； 1：Burst；

19. 64 位定时器（TIM26）

19.1. 概述

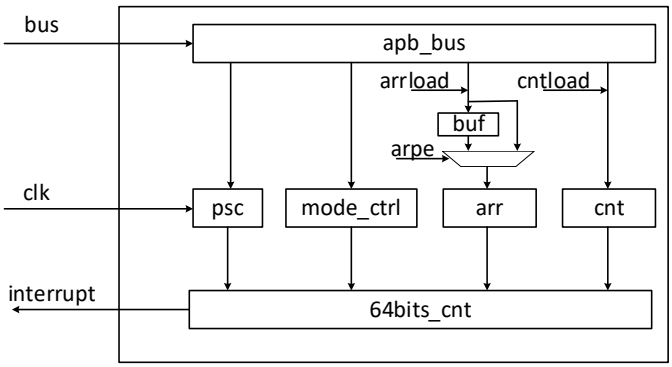
64 位定时器 TIM26 由一个 64 位的自动装载计数器组成，它由一个可编程的预分频器驱动。使用定时器预分频器和系统时钟控制预分频器，实现计数时钟频率。

19.2. 主要特性

- 64 位可编程向上计数器
- 6 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1 ~ 64 之间的任意数值

19.2.1. 结构框图

图 19-1 64 位定时器结构框图



19.2.2. 功能描述

19.2.2.1. 计数单元

工作模式：

- 自由计数模式：计数器从 0 或者 CNT 给定值开始计数到 2^64-1 时返回 0 重新计数
- 循环计数模式（中断模式）：计数器从 0 或者 CNT 给定值开始计数到自动重装载值后产生中断，返回 0 开始重新计数。

19.2.2.2. 预分频器

预分频器可以将计数器的时钟频率按 1 到 64 之间的任意值分频。它是基于一个(在 PSC 寄存器中的)6 位寄存器控制的 6 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新时被采用。

19.3. TIM 寄存器描述

TIM26 寄存器基地址：0x4000A400

偏移	名称	复位值	描述
0x00	TIM64B_CTRL	0x0	控制寄存器
0x04	TIM64B_SR	0x0	状态寄存器
0x08	TIM64B_ARRL	0x0	自动重装载寄存器低位(ARR[31:0])
0x0C	TIM64B_ARRH	0x0	自动重装载寄存器高位(ARR [63:32])
0x10	TIM64B_CNTL	0x0	计数器低位(CNT[31:0])
0x14	TIM64B_CNTH	0x0	计数器高位(CNT [63:32])

19.3.1. 控制寄存器（TIM64B_CTRL: 00h）

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留，始终读为 0。
11	CNTLOAD	WO	0x0	CNT 值加载位 置一将更新 TIM_CNT[63:0]值，该位硬件自动清零。注：每次写完 TIM_CNT 后，需更新一次 TIM_CNT[63:0]。
10	ARRLOAD	WO	0x0	ARR 值加载位 置一将更新 TIM_ARR[63:0]值，该位硬件自动清零。 注：每次写完 TIM_ARR 后，需更新一次 TIM_ARR[63:0]。
9	ARPE	RW	0x0	自动重装载预装载允许位 0:TIM_ARR[63:0]寄存器没有缓冲 1:TIM_ARR[63:0]寄存器被装入缓冲器，当下一次更新事件来临后更新。
8	IE	RW	0x0	中断使能（interrupt enable) 0：中断不使能； 1：中断使能。
7:2	PSC	RW	0x0	预分频器的值（Prescaler value) 计数器的时钟频率等于 fclk/(PSC[7:2]+1)。 每次当更新事件产生时，装入预分频器并生效
1	CMOD	RW	0x0	计数器模式(cnt mode) 0：自由计数模式； 1：循环计数模式。
0	CEN	RW	0x0	使能计数器 0：禁止计数器； 1：使能计数器。

19.3.2. 状态寄存器（TIM64B_SR: 04h）

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留，始终读为 0。

8	UF	RW	0x0	更新事件标记（Update flag） 0：无更新事件产生； 1：计数器更新事件发生。 该位由硬件置‘1’，软件写“1”清零，仅在循环计数模式下产生。
7:0	RSV	-	-	保留，始终读为 0。

19.3.3. 自动重装载寄存器低 32 位（TIM64B_ARRL: 08h）

位域	名称	属性	复位值	描述
31:0	ARRL	RW	0x0	自动重装载的值（Prescaler value） ARR [31:0]：ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。 注：读此位，其值为已装载的 ARR 值

19.3.4. 自动重装载寄存器高 32 位（TIM64B_ARRH: 0Ch）

位域	名称	属性	复位值	描述
31:0	ARRH	RW	0x0	自动重装载的值（Prescaler value） ARR[63:32]：ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。 注：读此位，其值为已装载的 ARR 值

19.3.5. 计数器低 32 位(TIM64B_CNTL: 10h)

位域	名称	属性	复位值	描述
31:0	CNTL	RW	0x0	计数器的值（Counter value） CNT[31:0]

注：当需要读取当前 64 位计数器的值时，先读取 CNT[31:0]的值，再读取 CNT[63:32]的值。

19.3.6. 计数器高 32 位(TIM64B_CNTH: 14h)

位域	名称	属性	复位值	描述
31:0	CNTH	RW	0x0	计数器的值（Counter value） CNT[63:32]

20. 低功耗定时器（LPTIM）

20.1. 概述

LPTIM 是一个 16 位定时器。由于 LPTIM 的时钟源具有多样性，因此 LPTIM 能够在所有电源模式（Standby 和 Powerdown 模式下除外）下保持运行状态。即使没有内部时钟源，LPTIM 也能运行，鉴于这一点，可将其用作“脉冲计数器”，这种脉冲计数器在某些应用中十分有用。此外，LPTIM 还能将系统从低功耗模式唤醒，因此非常适合实现“超时功能”，而且功耗极低。

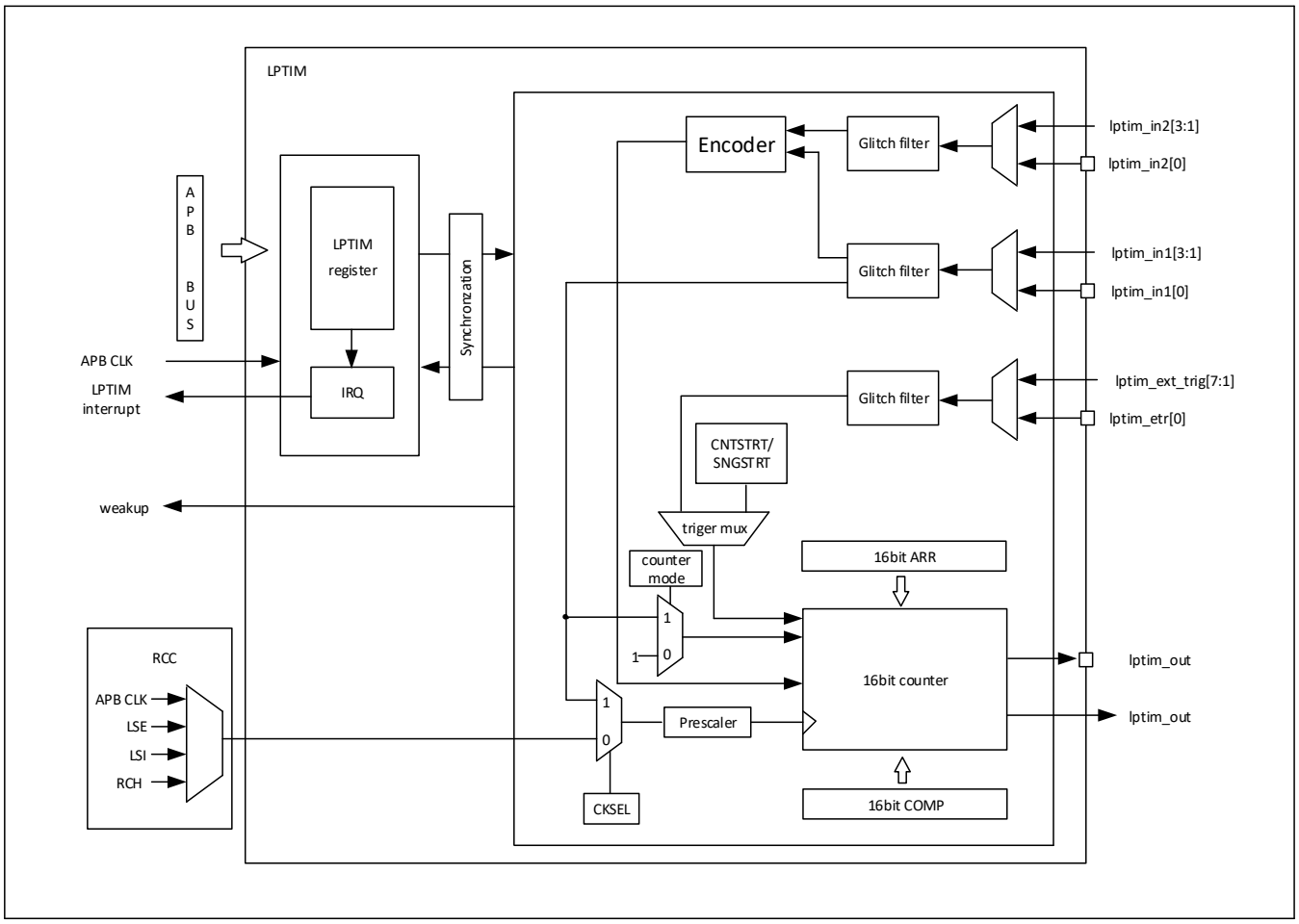
LPTIM 引入了一个灵活的时钟方案，该方案能够提供所需的功能和性能，同时还能最大程度地降低功耗。

20.2. 主要特性

- 16 位递增计数器
- 3 位预分频器，可采用 8 种分频系数（1、2、4、8、16、32、64 和 128）
- 可选时钟
  - 内部时钟源：PCLK、RC32K、RC64M、XTL 时钟
  - LPTIM 输入的外部时钟源（在没有低功耗振荡器运行的情况下工作，由脉冲计数器作为外部时钟源）
- 16 位 ARR 自动重载寄存器
- 16 位比较寄存器
- 连续/单次模式
- 可选软件/硬件输入触发
- 可编程数字干扰滤波器
- 可配置输出：脉冲和 PWM
- 可配置 I/O 极性
- 编码器模式
- 重复计数器

20.3. 结构框图

图 20-1 LPTIM 结构框图



lptim_out 是内部 LPTIM 输出信号，可以连接到内部外设。

20.4. 功能说明

20.4.1. 触发映射

表 20-1 LPTIM1 输入 1 连接

LPTIM_IN1_MUX	LPTIM1 输入 1 连接位置
LPTIM_IN1_MUX0	用作 LPTIM1_IN1 复用功能的 GPIO 引脚

表 20-2 LPTIM1 输入 2 连接

LPTIM_IN2_MUX	LPTIM1 输入 1 连接位置
LPTIM_IN2_MUX0	用作 LPTIM1_IN2 复用功能的 GPIO 引脚

表 20-3 外部触发连接

TRIGSEL	External trigger

LPTIM_EXT_TRIG0	GPIO

20.4.2. 复位和时钟

LPTIM 可通过多个时钟源提供时钟。它可以由内部时钟信号提供时钟，内部时钟信号可通过复位和时钟控制器(RCC)在 PCLK、RC32K、RC64M 和 XTL 时钟源中进行选择。此外 LPTIM 还可通过注入到其外部 Input1 上的外部时钟信号提供时钟。当通过外部时钟源提供时钟时，LPTIM 可以在下述两种可能配置中的其中一种配置下运行：

- 第一种配置是，LPTIM 通过外部信号提供时钟，但是同时，内部时钟信号从 PCLK 或任何其他嵌入式振荡器（包括 RC32K、RC64M、XTL）提供给 LPTIM。
- 第二种配置是，LPTIM 仅由外部时钟源通过外部 Input1 提供时钟。此配置可在进入低功耗模式后所有内置振荡器关闭时，用于实现超时功能或脉冲计数器功能。

对 CKSEL 和 COUNTMODE 位进行编程，可控制 LPTIM 使用外部时钟源还是内部时钟源。

当使用外部时钟源时，可使用 CKPOL 位选择外部时钟信号的有效边沿。如果上升沿和下降沿均为有效边沿，则还应提供内部时钟信号（第一种配置）。在这种情况下，内部时钟信号频率应至少为外部时钟信号频率的 4 倍。

20.4.3. 干扰滤波器

LPTIM 输入、外部（映射到 GPIO）或内部（在芯片级上映射到其他嵌入式外设，例如嵌入式比较器）由数字滤波器保护，避免任何毛刺和噪声干扰在 LPTIM 内部传播，从而防止产生意外计数或触发。在激活数字滤波器之前，首先应向 LPTIM 提供内部时钟源，这是保证滤波器正常工作的必要条件。

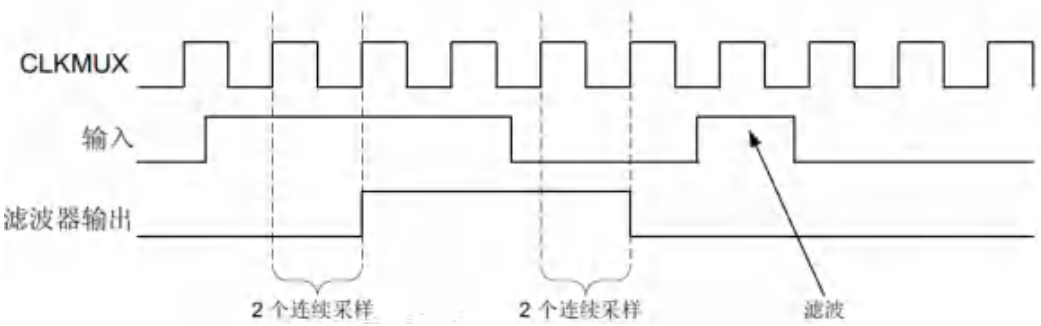
数字滤波器分为两组：

- 第一组数字滤波器保护 LPTIM 外部输入。数字滤波器的敏感性由 CKFLT 位控制。
- 第二组数字滤波器保护 LPTIM 内部触发输入。数字滤波器的敏感性由 TRGFLT 位控制。

注：数字滤波器的敏感性以组为单位进行控制。无法单独配置同一组内各个数字滤波器的敏感性。

滤波器的敏感性会影响相同的连续采样的数量，在其中一个 LPTIM 输入上检测到此类连续采样时，才能将某信号电平变化视为有效切换。下图给出了编程 2 个连续采样时，干扰滤波器行为的示例。

图 20-2 干扰滤波器时序图



注：不提供内部时钟信号时，必须通过将 CKFLT 和 TRGFLT 位设为 0 来停用数字滤波器。在这种情况下，可使用外部模拟滤波器来防止 LPTIM 外部输入产生干扰。

20.4.4. 预分频器

LPTIM16 位计数器前面有一个可配置的 2 次幂预分频器。预分频器的分频比由 PRESC[2：0]3 位字段控制。下表列出了所有可能的分频比：

表 20-4 预分频系数

PRESC	分频系数
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

使能匹配中断情况下，不建议使用预分频功能。否则当计数时钟较低时，中断程序退出后会因为中断标志位再次产生从而立即又进入中断程序，直到计数时钟加 1 后才能退出中断。

20.4.5. 触发多路复用器

LPTIM 计数器可通过软件启动，也可以在 8 个触发输入之一上检测到有效边沿后启动。

TRIGEN[1:0]用于确定 LPTIM 触发源：

- TRIGEN[1:0]等于“00”时，LPTIM 计数器会在通过软件将 CNTSTRT 位或 SNGSTRT 位其中之一置 1 后立即启动。TRIGEN[1:0]的其余三个可能的值用于配置触发输入使用的有效边沿。LPTIM 计数器会在检测到有效边沿后立即启动。
- TRIGEN[1:0]不等于“00”时，TRIGSEL[2:0]用于选择使用 8 个触发输入中的哪一个来启动计数器。

外部触发信号视为 LPTIM 的异步信号。因此，检测到触发信号后，由于同步问题，需要延迟两个计数器时钟周期，定时器才能开始运行。

如果在定时器已启动时发生新的触发事件，则此事件将被忽略（除非已使能超时功能）。

注：必须使能定时器，才能将 SNGSTRT/CNTSTRT 位置 1。当定时器禁止时，对这些位执行的任何写操作都将被硬件丢弃。

20.4.6. 工作模式

LPTIM 支持以下两种工作模式：

- 连续模式：定时器自由运行，由触发事件启动并且直到被禁止才会停止
- 单触发模式：定时器由触发事件启动，并在 LPTIM 产生更新事件时（或在达到无重复计数器的产品的 ARR 值时）停止。

1. 单触发模式

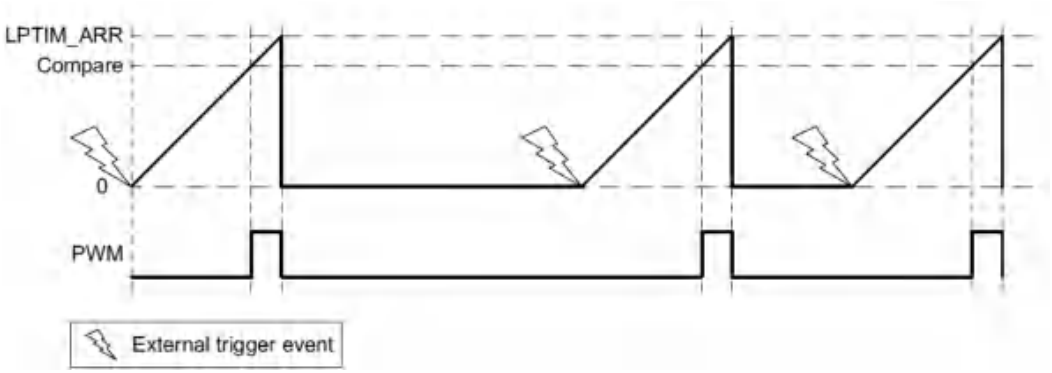
要启用单触发模式，必须将 SNGSTRT 位置 1。

- 重复计数器功能不可用

新的触发事件将重新启动计时器。在计数器启动之后且计数器达到 ARR 之前发生的任何触发事件都将被丢弃。

选择外部触发时，在 SNGSTRT 位置 1 后以及计数器寄存器停止后（包含零值）到达的每个外部触发事件都将为计数器启动新的单触发计数周期，如图所示。

图 20-3 LPTIM 输出波形，单触发模式

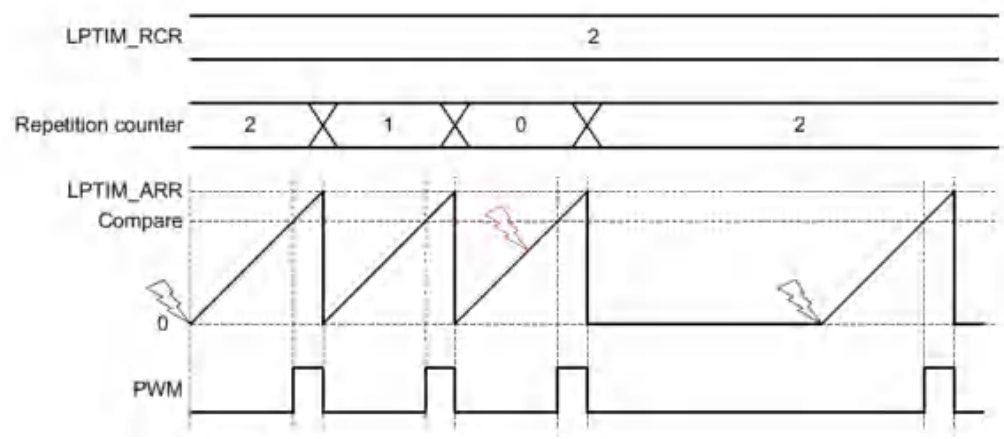


- 重复计数器功能可用

新的触发事件将重新启动计时器。在计数器启动之后和下一个 LPTIM 更新事件之前发生的任何触发事件都将被丢弃。

如果选择了外部触发，则每个外部触发事件在 SNGSTRT 位置 1 并且重复计数器停止之后（在更新事件之后）到达，并且如果重复寄存器的内容不为零，则重新加载重复计数器使用重复寄存器已经包含的值，并开始一个新的单次计数周期，如图所示。

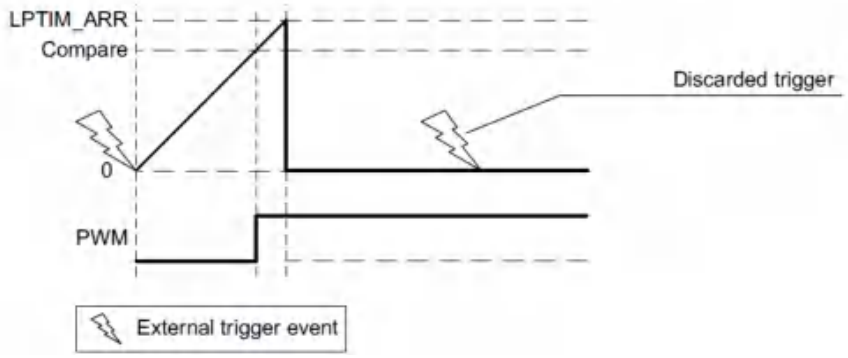
图 20-4 LPTIM 输出波形，单触发模式，当重复寄存器不为 0（PRELOAD=1）



已激活一次设置模式：

LPTIMx_CFGR 寄存器中的 WAVE 位域置 1 时，将激活置 1 一次模式。在这种情况下，计数器仅会在第一个触发事件后启动一次，任何后续触发事件都将被丢弃，如图所示。

图 20-5 LPTIM 输出波形，单触发模式，一次激活模式（WAVE=1）



若通过软件启动（TRIGEN[1:0]=“00”），将 SNGSTRT 置 1 会使计数器进行单触发计数。要使能连续计数，必须将 CNTSTRT 位置 1。

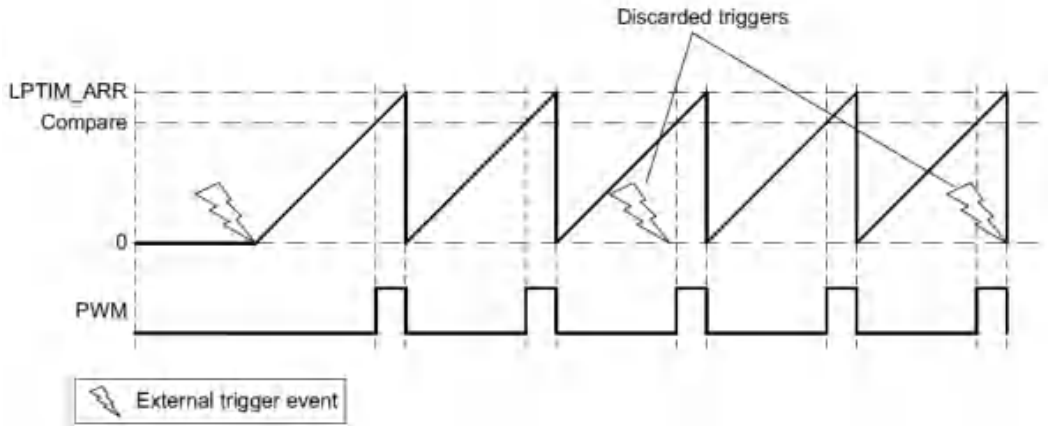
2. 连续模式

要启用连续计数，必须将 CNTSTRT 位置 1。

如果选择了外部触发，则在设置 CNTSTRT 之后到达的外部触发事件将启动计数器进行连续计数。任何后续的外部触发事件都将被丢弃，如图所示。

若通过软件启动（TRIGEN[1:0] =“00”），将 CNTSTRT 置 1 会使计数器开始连续计数。

图 20-6 LPTIM 输出波形，重复触发模式



SNGSTRT 和 CNTSTRT 位只能在定时器使能时（ENABLE 位置 1）置 1。可以“实时的”从单次模式切换为连续模式。

如果先前选择了连续模式，则设置 SNGSTRT 会将 LPTIM 切换为单发模式。一旦生成 LPTIM 更新事件（或对于没有重复计数器的产品，当达到 ARR 时），计数器（如果处于活动状态）将停止。

如果之前选择的是单触发模式，则将 CNTSTRT 置 1 会使 LPTIM 切换为连续模式。计数器（激活时）将在达到 ARR 后立即重新启动。

20.4.7. 超时功能

若在一个选定的触发输入上检测到有效边沿，则可用于复位 LPTIM 计数器。该功能通过 TIMOUT 位进行控制。

第一个触发事件将启动计时器，任何连续的触发事件将复位 LPTIM 计数器和重复计数器，并且计时器将重新启动。

可实现低功耗超时功能。超时值对应于比较值；如果在预期的时间帧内未发生触发，MCU 将由比较匹配事件唤醒。

20.4.8. 生成波形

两个 16 位寄存器，LPTIMx_ARR（自动重载寄存器）和 LPTIMx_CMP（比较寄存器）用于在 LPTIM 输出上生成多个不同的波形

定时器可生成以下波形：

● PWM 模式

若 LPTIMx_CMP 寄存器与 LPTIMx_CNT 寄存器匹配（相等），则会立即将 LPTIM 输出置 1。若 LPTIMx_ARR 寄存器与 LPTIMx_CNT 寄存器匹配（相等），则会立即将 LPTIM 输出复位。

立即更新寄存器模式下，若动态更改的 LPTIMx_CMP 值小于当前 LPTIMx_CNT 值，则当前周期内将无法匹配，LPTIM 输出保持不变，直到 LPTIMx_CNT 值与 LPTIMx_ARR 值匹配，LPTIM 输出复位。

立即更新寄存器模式下，若动态更改的 LPTIMx_ARR 值小于当前 LPTIMx_CNT 值，则当前周期内将无法匹配，LPTIM 输出保持不变，直到 LPTIM_CNT 计数到 0xFFFF 溢出回到 0，下个周期 LPTIMx_CMP 值与 LPTIMx_CNT 值匹配时，LPTIM 输出置 1。

● 单脉冲模式

对于第一个脉冲，输出波形与 PWM 模式输出波形类似，随后输出将永久复位。

● 置 1 一次模式

除输出保持最后一个信号电平外（取决于配置的输出极性），输出波形与单脉冲模式输出波形类似。

上述模式要求 LPTIMx_ARR 寄存器的值严格大于 LPTIMx_CMP 寄存器的值。

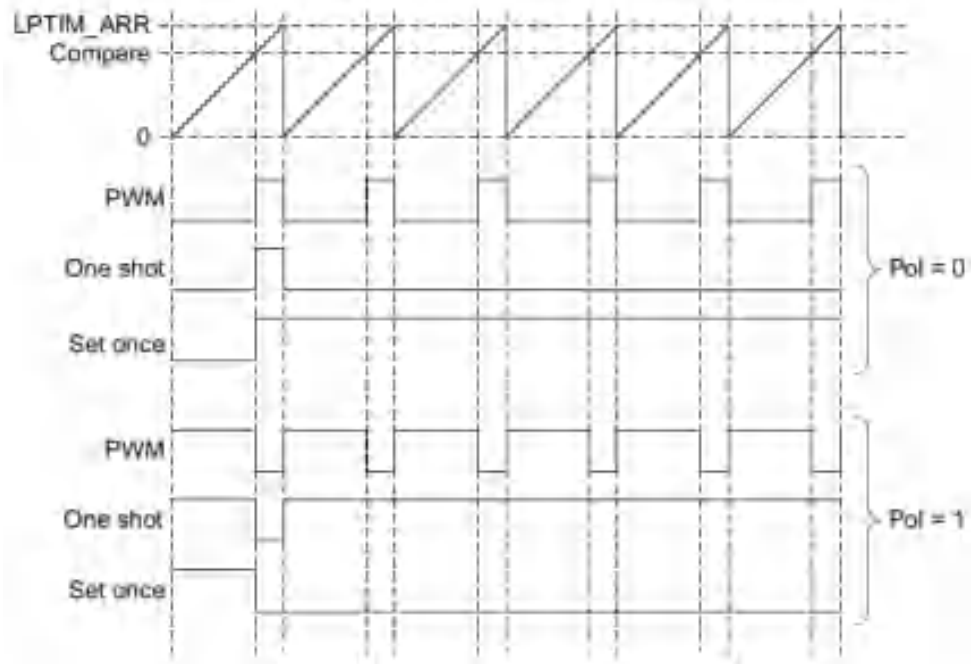
LPTIM 输出波形可通过 WAVE 位配置，具体如下：

- 若将 WAVE 位复位为 0，则会强制 LPTIM 生成 PWM 波形或单脉冲波形，具体取决于将哪个位（CNTSTRT 或 SNGSTRT）置 1。
- 若将 WAVE 位置 1，则会强制 LPTIM 生成置 1 一次模式波形。

WAVPOL 位控制 LPTIM 输出极性。更改立即生效，因此输出默认值将在极性重新配置后立即更改，甚至会在定时器使能前进行更改。

生成的信号的频率高达 LPTIM 时钟频率 2 分频。图 197 给出了可能在 LPTIM 输出上生成的三种波形。此外，此图还显示了通过 WAVPOL 位更改极性所产生的效果。

图 20-7 生成波形



20.4.9. 寄存器更新

LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器在 APB 总线写操作后会立即更新，或如果计时器已启动则与下一个 LPTIM 更新事件同步（如果没有重复计数器的产品已经启动计时器，则在当前时间段结束时）。

PRELOAD 位控制 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器的更新方式：

- 当 PRELOAD 位复位为 0 时，LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器会在写访问后立即更新。
- 当 PRELOAD 位置 1 时，若定时器已启动，LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器会在下一个 LPTIM 更新事件中更新（或在无重复计数器产品的当前阶段结束时）。

APB 总线和 LPTIM 使用的时钟不同，因此在 APB 写操作后，需要经过一定的延迟，写入值才能用于计数器比较器。在此延迟期间，必须避免向这些寄存器执行其他写操作。

LPTIMx_ISR 寄存器中的 ARROK 标志和 CMPOK 标志分别指示 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器的写操作已完成。

向 LPTIMx_ARR 寄存器和 LPTIMx_CMP 寄存器执行写操作后，只有在前一次写操作完成后，才能对同一寄存器执行新的写操作。在 ARROK 标志或 CMPOK 标志置 1 前执行连续的写操作将造成无法预知的结果。

20.4.10. 计数器模式

LPTIM 计数器可用于对 LPTIM Input1 上的外部事件进行计数，也可用于对内部时钟周期进行计数。CKSEL 位和 COUNTMODE 位用于控制将使用哪些源更新计数器。

若使用 LPTIM 对 Input1 上的外部事件进行计数，计数器可在上升沿、下降沿或两种边沿进行更新，具体取决于写入 CKPOL[1:0]位的值。

根据 CKSEL 和 COUNTMODE 值，可选择以下计数模式：

- CKSEL = 0：LPTIM 由内部时钟源提供时钟
  - COUNTMODE = 0

LPTIM 配置为由内部时钟源提供时钟，而 LPTIM 计数器配置为在每个内部时钟脉冲之后进行更新。

  - COUNTMODE = 1

LPTIM 外部 Input1 通过提供给 LPTIM 的内部时钟采样。因此，为了不丢失任何事件，外部 Input1 信号变化

的频率决不应超过提供给 LPTIM 的内部时钟的频率。此外，提供给 LPTIM 的内部时钟一定不能预先分频（PRESC[2：0] = 000）。

- CKSEL = 1：LPTIM 由外部时钟源提供时钟
  - COUNTMODE 值不相关。

在这种配置下，LPTIM 无需内部时钟源（已使能干扰滤波器时除外）。注入到 LPTIM 外部 Input1 的信号用作 LPTIM 的系统时钟。此配置适合未使能任何内置振荡器的工作模式。对于这种配置，LPTIM 计数器可以在 input1 时钟信号的上升沿或下降沿进行更新，但不可在上升沿和下降沿均更新。

由于注入到 LPTIM 外部 Input1 的信号也可用于 LPTIM 的时钟，计数器递增计数前存在一些初始延时（使能 LPTIM 后）。更确切地说，LPTIM 外部 Input1 的前五个有效边沿将丢失（使能 LPTIM 后）。

20.4.11. 定时器使能

LPTIMx_CR 寄存器的 ENABLE 位用于使能/禁止 LPTIM。将 ENABLE 位置 1 后，需要延迟两个计数器时钟周期，才能真正使能 LPTIM。

LPTIMx_CFGR 和 LPTIMx_IER 寄存器必须在禁止 LPTIM 后才能修改。

20.4.12. 定时器计数器复位

为了将 LPTIM_CNT 寄存器的内容重置为零，实现了两种重置机制：

同步复位机制：同步复位由 LPTIM_CR 寄存器中的 COUNTRST 位控制。将 COUNTRST 位字段设置为“1”后，复位信号将在 LPTIM 内核时钟域中传播。因此，重要的是要注意，在考虑复位之前，将经过 LPTIM 内核逻辑的几个时钟脉冲。这将使 LPTIM 计数器在触发复位到生效之间产生一些额外的计数。由于 COUNTRST 位位于 APB 时钟域中，而 LPTIM 计数器位于 LPTIM 内核时钟域中，向 COUNTRST 位写入 1 时，内核时钟需要 3 个时钟周期的延迟才能同步 APB 时钟域发出的复位信号。

异步复位机制：异步复位由位于 LPTIM_CR 寄存器中的 RSTARE 位控制。当该位置 1 时，对 LPTIM_CNT 寄存器的任何读访问都将其内容复位为零。异步复位应在没有提供 LPTIM 内核时钟的时间范围内触发。例如，当 LPTIM Input1 用作外部时钟源时，仅当有足够的保险保证在 LPTIM Input1 上不会发生翻转时，才应应用异步复位。

应该注意的是，为了可靠地读取 LPTIM_CNT 寄存器的内容，必须执行两个连续的读取访问并进行比较。当两个读取访问的值相等时，可以认为读取访问是可靠的。不幸的是，启用异步复位后，将无法读取两次 LPTIM_CNT 寄存器。

注：LPTIM 内部没有机制可以防止开发人员同时使用两种复位机制。因此，开发人员应确保应用程序上具备不能同时使用这两种复位机制的方法，如互斥机制。

20.4.13. 编码器模式

此模式用于处理来自正交编码器的信号，此正交编码器用于检测旋转元件的角度位置。编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 LPTIMx_ARR 寄存器中编程的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 LPTIMx_ARR。通过两个外部输入信号 Input1 和 Input2 生成时钟信号作为 LPTIM 计数器时钟。这两个信号间的相位确定计数方向。

仅当 LPTIM 由内部时钟源提供时钟时才可使用编码器模式。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟频率 4 分频。必须满足此条件才能确保 LPTIM 正常工作。

方向更改通过 LPTIM_ISR 寄存器中的两个 Down 和 Up 标志发出信号。使能 DOWNIE 位、UPIE 位，两个方向的改变事件都能产生中断。

要激活编码器模式，必须将 ENC 位置 1。LPTIM 必须首先配置为连续模式。

激活编码器模式后，将根据增量编码器的速度和方向自动修改 LPTIM 计数器。因此，其内容始终代表编码器的位置。计数方向由递增和递减标志指示，对应于所连传感器的旋转方向。

根据使用 CKPOL[1:0]位配置的边沿敏感性，可得几种不同的计数方案。下表汇总了可能的组合（假设 Input1 和 Input2 不同时切换）。

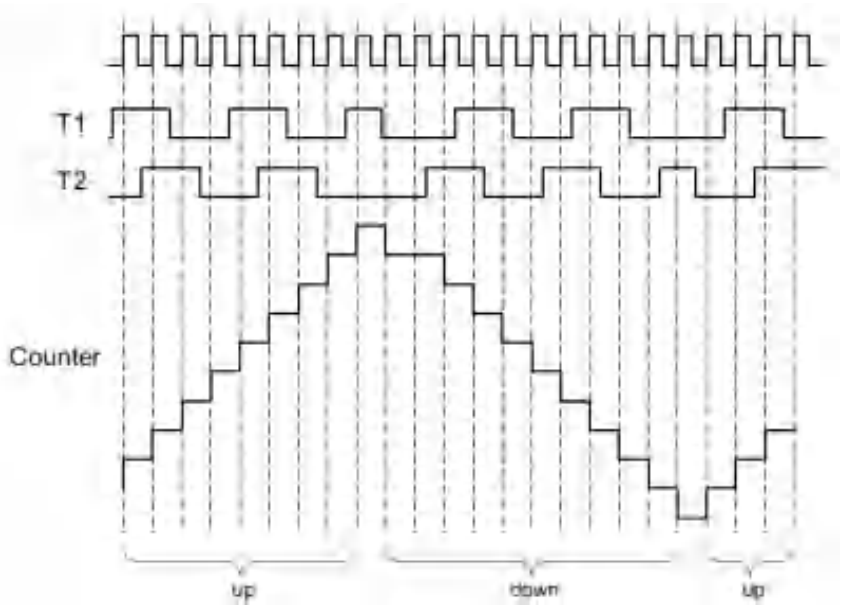
表 20-5 编码器计数方案

有效边沿	相反信号的电平 input1 对应 input2 input2 对应 input1	Input1 信号		Input2 信号	
		上升	下降	上升	下降
上升沿	高	递减	不计数	递增	不计数
	低	递增	不计数	递减	不计数
下降沿	高	不计数	递增	不计数	递减
	低	不计数	递减	不计数	递增
上升沿 下降沿	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

下图所示为编码器模式下配置了两种边沿敏感性的计数序列。

注意：在此模式下，LPTIM 必须由内部时钟源提供时钟，因此 CKSEL 位必须保持其复位值 0。另外，预分频器分频比必须等于其复位值 1（PRESC[2:0]位必须为“000”）。

图 20-8 编码器模式计数序列



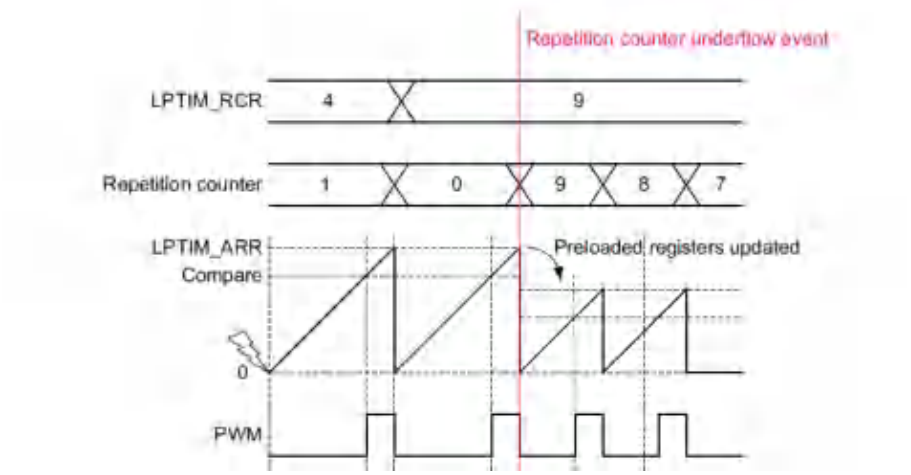
20.4.14. 重复计数器

LPTIM 具有一个重复计数器，每次 LPTIM 计数器溢出事件发生时，该计数器就会递减 1。当重复计数器到达零并且 LPTIM 计数器溢出时，将生成重复计数器下溢事件。在每个重复计数器下溢事件之后，向重复计数器加载属于重复寄存器 LPTIM_RCR 的 REP[7：0]位字段的内容。

当重复计数为 0 时，每个 LPTIM 计数器溢出都会产生重复下溢事件。

写入 REP[7：0]位域的值并不能立即加载到重复计数器中，仅当下一次重复下溢事件发生，写入的 REP[7：0]数值才会加载到重复计数器中。此行为在下图进行了描述。

图 20-9 重复计数模式 LPTIM_RCR 不为 0（PRELOAD=1）



重复计数器下溢事件与 LPTIM 预加载的寄存器更新有系统地关联（有关更多信息，请参见“寄存器更新”部分）。

重复计数器下溢事件通过映射到 LPTIM_ISR 寄存器中的更新事件（REPUE）标志发送给软件。置位时，如果设置了对应的映射到 LPTIM_IER 寄存器的更新事件中断使能（REPUEIE）控制位，则 REPUE 标志可以触发 LPTIM 中断。

重复寄存器 LPTIM_RCR 位于 APB 总线接口时钟域中，重复计数器本身位于 LPTIM 内核时钟域中。每次将新值写入 LPTIM_RCR 寄存器时，该新内容就会从 APB 总线接口时钟域传播到 LPTIM 内核时钟域，以便在重复计数器下溢事件发生后立即将新写入的值加载到重复计数器。新写入内容的同步延迟是四个 APB 时钟周期加上三个 LPTIM 内核时钟周期，并且经过 LPTIM_ISR 寄存器中的 REPOK 标志来发出信号。当 LPTIM 内核时钟周期相对较慢时例如，当 XTL 时钟源为 LPTIM 内核提供时钟时，可能需要很长时间才能通过软件对 REPOK 标志进行轮询，以检测 LPTIM_RCR 寄存器内容的同步已完成。因此，如果设置了 REPOK 标志，则 LPTIM_IER 寄存器中的 REPOKIE 控制位将被置位，从而产生中断。

注：对 LPTIM_RCR 寄存器进行写操作后，只有在完成前一个写操作后，才能对同一个寄存器执行新的写操作。REPOK 标志置 1 之前的任何连续写操作都将导致不可预测的结果。

注意：使用重复计数器时，必须将 LPTIM_CFGR 寄存器中的 PRELOAD 位置 1，否则可能会发生不可预测的行为。

20.5. LPTIM 低功耗模式

表 20-6 LPTIM 低功耗模式

模式	描述
Sleep	LPTIM 中断可退出 Sleep 模式。
Stop	LPTIM 时钟源选择 RC32K 时，LPTIM 中断可退出 Stop 模式。
Standby	LPTIM 断电不工作。

20.6. LPTIM 中断

若以下事件通过 LPTIMx_IER 寄存器使能，则这些事件会生成中断/唤醒事件：

- 与设定的数值比较后匹配

- 自动重载匹配（编码器模式下无论哪种方向）
- 外部触发事件
- 自动重载寄存器写操作完成
- 比较寄存器写操作完成
- 方向变化（编码器模式），可编程（递增/递减/同时递增和递减）。
- 重复计数器下溢事件
- 重复寄存器更新完成

注：只要 LPTIMx_IER 寄存器（中断使能寄存器）中的位在 LPTIMx_ISR 寄存器（状态寄存器）中相应标志置 1 后置 1，就不会触发中断。

表 20-7 中断事件

中断事件	说明
比较匹配	当计数器寄存器（LPTIM_CNT）的内容与比较寄存器（LPTIM_CMP）的内容匹配时，生成中断标志。
自动重载匹配	当计数器寄存器（LPTIM_CNT）的内容与自动重新加载寄存器（LPTIM_ARR)的内容匹配时，生成中断标志。
外部触发事件	当检测到外部触发事件时，会生成中断标志。
自动重载寄存器写操作完成	当对 LPTIM_ARR 寄存器的写操作完成时，生成中断标志。
比较寄存器写操作完成	当对 LPTIM_CMP 寄存器的写操作完成时，生成中断标志。
方向更改	用于编码器模式。嵌入两个中断标志以发出方向更改的信号： – UP 标志发出递增计数方向更改的信号 – DOWN 标志发出递减计数方向更改的信号
重复计数器下溢事件	重复计数器下溢事件发生时，生成中断标志。
重复寄存器更新完成	当对 LPTIM_RCR 中 REP[7：0]的写操作完成时，生成中断标志。

20.7. LPTIM 寄存器描述

20.7.1. 寄存器列表

LPTIM1 寄存器基地址：0x40007C00

LPTIM2 寄存器基地址：0x40009400

LPTIM3 寄存器基地址：0x51000000

LPTIM4 寄存器基地址：0x51000400

LPTIM5 寄存器基地址：0x51000800

LPTIM6 寄存器基地址：0x51000C00

偏移	名称	复位值	描述

0x00	LPTIM_ISR	0x00000000	中断和状态寄存器
0x04	LPTIM_ICR	0x00000000	中断清零寄存器
0x08	LPTIM_IER	0x00000000	中断使能寄存器
0x0C	LPTIM_CFGR1	0x00000000	配置寄存器 1
0x10	LPTIM_CR	0x00000000	控制寄存器
0x14	LPTIM_CMP	0x00000000	比较寄存器
0x18	LPTIM_ARR	0x00000001	自动重载寄存器
0x1C	LPTIM_CNT	0x00000000	计数器寄存器
0x24	LPTIM_CFGR2	0x00000000	配置寄存器 2
0x28	LPTIM_RCR	0x00000000	重复寄存器

20.7.2. 中断和状态寄存器(LPTIM_ISR: 00h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	REPOK	R	0	重复寄存器更新成功 0：重复寄存器未更新。 1：重复寄存器更新成功。 当 LPTIM_RCR 寄存器 REP 的 APB 总线写操作已成功完成，此位由硬件置位。
7	REPUE	R	0	重复计数器下溢事件 0：重复计数器未发生下溢事件。 1：重复计数器发生下溢事件。 当重复计数器下溢事件发生时，此位由硬件置 1。 注：如果 LPTIM 不支持重复计数器功能，该位保留。
6	DOWN	R	0	计数方向从递增变为递减 (Counter direction change up to down) 0：计数方向未从递增变为递减 1：计数方向从递增变为递减 在编码器模式下，计数方向从递增变为递减时，此位由硬件置 1。
5	UP	R	0	计数方向从递减变为递增 (Counter direction change down to up) 0：计数方向未从递减变为递增 1：计数方向从递减变为递增 在编码器模式下，计数方向从递减变为递增时，此位由硬件置 1。
4	ARROK	R	0	自动重载寄存器更新成功 (Autoreload register update OK) 0：自动重载寄存器未更新 1：自动重载寄存器更新成功 当对 LPTIM_ARR 寄存器的 APB 总线写操作完成时，此位由硬件置 1。

3	CMPOK	R	0	比较寄存器更新成功（Compare register update OK） 0：比较寄存器未更新 1：比较寄存器更新成功 当对 LPTIM_CMP 寄存器的 APB 总线写操作完成时，此位由硬件置 1。
2	EXTTRIG	R	0	外部触发边沿事件（External trigger edge event） 0：未发生外部触发边沿事件 1：发生外部触发边沿事件 外部触发输入上产生有效边沿时，此位由硬件置 1。如果此时定时器已启动，而忽略触发事件，则不会将此标志置 1。
1	ARRM	R	0	自动重载匹配（Autoreload match） 0：LPTIM_CNT 值与 LPTIM_ARR 值不匹配 1：LPTIM_CNT 值与 LPTIM_ARR 值匹配 当 LPTIM_CNT 值与 LPTIM_ARR 值相等时，此位由硬件置 1。
0	CMPM	R	0	比较匹配（Compare match） 0：LPTIM_CNT 值与 LPTIM_CMP 值不匹配 1：LPTIM_CNT 值与 LPTIM_CMP 值匹配 当 LPTIM_CNT 值与 LPTIM_CMP 值相等时，此位由硬件置 1。

20.7.3. 中断清零寄存器(LPTIM_ICR: 04h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	REPOKCF	W	0	REPOK 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 REPOK 标志 注：如果 LPTIM 不支持重复计数器功能，该位保留。
7	REPUECF	W	0	REPUE 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 REPUE 标志 注：如果 LPTIM 不支持重复计数器功能，该位保留。
6	DOWNCF	W	0	DOWN 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 DOWN 标志
5	UPCF	W	0	UP 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 UP 标志

4	ARROKCF	W	0	ARROK 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 ARROK 标志
3	CMPOKCF	W	0	CMPOK 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 CMPOK 标志
2	EXTTRIGCF	W	0	EXTTRIG 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 EXTTRIG 标志
1	ARRMCF	W	0	ARRM 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 ARRM 标志
0	CMPMCF	W	0	CMPM 标志清零 0：无效操作 1：清除 LPTIM_ISR 中 CMPM 标志

20.7.4. 中断使能寄存器(LPTIM_IER: 08h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	REPOKIE	RW	0	重复寄存器更新成功中断使能 0：禁止 1：使能 注：如果 LPTIM 不支持重复计数器功能，该位无意义。
7	REPUEIE	RW	0	重复计数器下溢事件中断使能 0：禁止 1：使能 注：如果 LPTIM 不支持重复计数器功能，该位无意义。
6	DOWNIE	RW	0	方向变为递减中断使能（Direction change to down Interrupt Enable) 0：禁止 1：使能
5	UPIE	RW	0	方向变为递增中断使能（Direction change to UP Interrupt Enable) 0：禁止 1：使能
4	ARROKIE	RW	0	自动重载寄存器更新成功中断使能（Autoreload register update OK Interrupt Enable)

				0：禁止 1：使能
3	CMPOKIE	RW	0	比较寄存器更新成功中断使能（Compare register update OK Interrupt Enable) 0：禁止 1：使能
2	EXTTRIGIE	RW	0	外部触发有效边沿中断使能（External trigger valid edge Interrupt Enable) 0：禁止 1：使能
1	ARRMIE	RW	0	自动重载匹配中断使能（Autoreload match Interrupt Enable) 0：禁止 1：使能
0	CMPMIE	RW	0	比较匹配中断使能（Compare match Interrupt Enable) 0：禁止 1：使能

注：LPTIM_IER 寄存器必须在 LPTIM 禁止（LPTIM_CR 的 ENABLE 位为 0）时才能修改。

20.7.5. 配置寄存器 1 LPTIM_CFGR1: 0Ch)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	REP_LODMOD	RW	0	重复寄存器值加载模式（REP load mode） REP_LODMOD 位控制 LPTIM_RCR 寄存器的更新方式 0：寄存器在每次 APB 总线写访问后更新 1：寄存器在当前 LPTIM 重复计数器递减为零时更新 注：此位为零时，重复寄存器值在每次 APB 总线写访问后 4 个内部计数 CLK 后有效
24	ENC	RW	0	编码器模式使能（Encoder mode enable) ENC 位控制编码器模式 0：禁止编码器模式 1：使能编码器模式
23	COUNTMODE	RW	0	计数器模式使能（counter mode enabled) COUNTMODE 位用于选择 LPTIM 使用哪个时钟源来为计数器提供时钟： 0：计数器在每个内部时钟脉冲后递增 1：计数器在 LPTIM 外部 Input1 上的每个有效时钟脉冲后递增

22	PRELOAD	RW	0	寄存器更新模式（Registers update mode) PRELOAD 位控制 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器的更新方式 0：寄存器在每次 APB 总线写访问后更新 1：寄存器在当前 LPTIM 周期结束时更新
21	WAVPOL	RW	0	波形极性（Waveform shape polarity) WAVEPOL 位控制输出极性 0： LPTIM_CNT 小于 LPTIM_CMP 时，输出低电平，LPTIM_CNT 等于 LPTIM_CMP 时，输出高电平。 1： LPTIM_CNT 小于 LPTIM_CMP 时，输出高电平，LPTIM_CNT 等于 LPTIM_CMP 时，输出低电平。
20	WAVE	RW	0	波形（Waveform shape) WAVE 位控制输出波形 0： 停用置 1 一次模式和 PWM/单脉冲波形 1： 激活置 1 一次模式
19	TIMEOUT	RW	0	超时使能(timeout enable) TIMOUT 位控制超时功能 0：定时器已启动后，外部触发事件不会清零计数器 1：定时器已启动后，外部触发事件将会清零计数器
18:17	TRIGEN	RW	00	触发使能和极性（Trigger enable and polarity) TRIGEN 位控制 LPTIM 计数器是否由外部触发信号启动。如果已选择由外部触发信号启动，触发有 效边沿的配置有以下三种： 00：软件触发（由软件启动计数） 01：上升沿为有效边沿 10：下降沿为有效边沿 11：上升沿和下降沿均为有效边沿
16:13	TRIGSEL	RW	000	触发源选择器（Trigger selection) TRIGSEL 位用于选择作为 LPTIM 触发事件的触发源，可用触发源包括以下 8 种： 0000： GPIO 0001： lptim_ext_trig1 ... 1111： lptim_ext_trig15
12	RSV	-	-	保留

11:9	PRESC	RW	000	时钟预分频器（Clock prescaler） PRESC 位配置预分频器的分频系数。分频系数可从以下分频系数中选择： 000： 1 001： 2 010： 4 011： 8 100： 16 101： 32 110： 64 111： 128
8	RSV	-	-	保留
7:6	TRGFLT	RW	00	触发信号的可配置数字滤波器（Configurable digital filter for trigger） TRGFLT 值用于设置连续相同采样的数量，若在内部触发信号电平发生变化时检测到此类连续采样，才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能。 00：任何触发信号有效电平变化均视为有效触发。 01：触发信号有效电平变化必须至少稳定 2 个时钟周期，才能将其视为有效触发。 10：触发信号有效电平变化必须至少稳定 4 个时钟周期，才能将其视为有效触发。 11：触发信号有效电平变化必须至少稳定 8 个时钟周期，才能将其视为有效触发。
5	RSV	-	-	保留
4:3	CKFLT	RW	00	外部时钟的可配置数字滤波器（Configurable digital filter for external clock） CKFLT 值用于设置连续相同采样的数量，若在外部时钟信号电平发生变化时检测到此类连续采样， 才会将此电平变化视为有效电平切换。必须存在内部时钟源才能使用此功能。 00：任何外部时钟信号电平变化均视为有效切换。 01：外部时钟信号电平变化必须至少稳定 2 个时钟周期，才能将其视为有效切换。 10：外部时钟信号电平变化必须至少稳定 4 个时钟周期，才能将其视为有效切换。 11：外部时钟信号电平变化必须至少稳定 8 个时钟周期，才能将其视为有效切换。

2:1	CKPOL	RW	00	时钟极性（Clock Polarity） 如果 LPTIM 由外部时钟源提供时钟： 当 LPTIM 由外部时钟源提供时钟时， CKPOL 位用于配置计数所使用的有效边沿： 00：上升沿为用于计数的有效边沿。 01：下降沿为用于计数的有效边沿。 10：上升沿和下降沿均为有效边沿。当外部时钟信号的上升沿和下降沿均视为有效边沿时，LPTIM 还必须由内部时钟源提供时钟，且内部时钟源频率至少等于外部时钟频率的四倍。 11：不允许 如果将 LPTIM 配置为编码器模式（ENC 位置 1）： 00：编码器子模式 1 激活。 01：编码器子模式 2 激活。 10：编码器子模式 3 激活。 有关编码器模式子模式的更多详细信息，请参见第 38.4.14 节：编码器模式。
0	CKSEL	RW	0	时钟选择器（Clock selector） CKSEL 位选择 LPTIM 将使用的时钟源： 0： LPTIM 由内部时钟源（APB 时钟或任意内置振荡器）提供时钟。 1： LPTIM 由外部时钟源通过 LPTIM 外部 Input1 提供时钟。

注：LPTIM_CFGR1 寄存器必须在 LPTIM 禁止（LPTIM_CR 的 ENABLE 位为 0）时才能修改。

20.7.6. 控制寄存器(LPTIM_CR: 10h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	RSTARE	WO	0	读操作后复位使能（Reset after read enable） 此位由软件置 1 和清零。当 RSTARE 置 1 时，对 LPTIM_CNT 寄存器的任何读取访问都将异步复位 LPTIM_CNT 寄存器的内容。 注意： 只能对该位域执行写操作。这意味着该位不能被读回以验证已写入的值。例如，如果该位设置为 1，即使使能了“读操作后复位”功能（由于该位域以前已写入 1），尝试读取它也将返回 0。要关闭“读操作后复位”或确保已经关闭，应（通过将其编程为 0）将该位复位，即使其已经为 0。
3	COUNTRST	WO	0	计数器复位（Counter reset） 此位通过软件置 1，通过硬件清零。当设置为 1 时，该位将触发 LPTIM_CNT 计数器寄存器的同步复位。由于该复位的同步性质，它仅在 3 个 LPTimer 内核时钟周期（LPTimer 内核时钟可能不同于 APB 时钟）的同步延迟之后发生。 注意： COUNTRST 在已由硬件清零之前，不得由软件置“1”。因此，软件应在尝试将 COUNTRST 位置“1”之前检查其是否已清零。

2	CNTSTRT	WO	0	定时器以连续模式启动（Timer start in Continuous mode） 此位通过软件置 1，通过硬件清零。 若通过软件启动（TRIGEN[1:0] = “00”），将此位置 1 会使 LPTIM 以连续模式启动。 如果禁止软件启动（TRIGEN[1:0] 不等于“00”），将此位置 1 会使定时器在检测到外部触发信号后立即以连续模式启动。 如果在进行单脉冲模式计数时将此位置 1，则在 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次 匹配时定时器不会停止， LPTIM 计数器将继续以连续模式计数。 只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。
1	SNGSTRT	WO	0	LPTIM 以单脉冲模式启动（LPTIM start in Single mode） 此位通过软件置 1，通过硬件清零。 若通过软件启动（TRIGEN[1:0] = “00”），将此位置 1 会使 LPTIM 以单脉冲模式启动。 如果禁止软件启动（TRIGEN[1:0] 不等于“00”），将此位置 1 会使 LPTIM 在检测到外部触发信号后立即以单脉冲模式启动。 如果在 LPTIM 处于连续计数模式时将此位置 1， LPTIM 将在 LPTIM_ARR 寄存器和 LPTIM_CNT 寄存器下一次匹配时停止。 只有在使能 LPTIM 时才能将此位置 1。此位由硬件自动复位。
0	ENABLE	RW	0	LPTIM 使能（LPTIM enable） ENABLE 位由软件置 1 和清零。 0：禁止 LPTIM 1：使能 LPTIM

20.7.7. 比较寄存器(LPTIM_CMP: 14h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CMP	RW	0x0000	比较值（Compare value） CMP 为 LPTIM 所使用的比较值。 注意： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_CMP 寄存器。

20.7.8. 自动重载寄存器(LPTIM_ARR: 18h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留

15:0	ARR	RW	0x0001	自动重载值（Auto reload value） ARR 为 LPTIM 的自动重载值。 此值必须严格大于 CMP[15:0] 的值。 注意： 必须在 LPTIM 已使能时（ENABLE 位置 1）才能修改 LPTIM_ARR 寄存器。

20.7.9. 计数器寄存器(LPTIM_CNT: 1Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CNT	R	0x0000	计数器值（Counter value） 当 LPTIM 通过异步时钟运行时，读取 LPTIM_CNT 寄存器会返回不可靠的值。因此在这种情况下，必须连续执行读访问两次，并验证两次返回的值是否相同。当两次连续读访问的值相同时，可认为读访问可靠。

20.7.10. 配置寄存器 2 LPTIM_CFGR2: 24h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:4	IN2SEL	RW	00	LPTIM 输入 2 选择（LPTIM input 2 selection） IN2SEL 位控制 LPTIM 输入 2 多路复用器，它将 LPTIM 输入 2 连接到可用输入之一。 0000:GPIO 0001: lptim_in2_mux1 ... 1111: lptim_in2_mux15
3:0	IN1SEL	RW	00	LPTIM 输入 1 选择（LPTIM input 1 selection） IN1SEL 位控制 LPTIM 输入 1 多路复用器，它将 LPTIM 输入 1 连接到可用输入之一。 0000: GPIO 0001: lptim_in1_mux1 ... 1111: lptim_in1_mux15

注：LPTIM_CFGR2 寄存器必须在 LPTIM 使能（LPTIM_CR 的 ENABLE 位为 1）时才能修改。

20.7.11. 重复寄存器(LPTIM_RCR: 28h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
7:0	REP	RW	0x00	重复寄存器值 REP 是 LPTIM 的重复值。 在下一个重复下溢事件发生之前，写入 REP[7:0]对重复计数器的内容没有影响。 注意：LPTIM_RCR 寄存器只能在 LPTIM 使能（ENABLE 位设置为 1）时修改。

21. 系统看门狗（WDT）

21.1. 概述

系统看门狗（WDT）是一个定时器电路，通常被用来监测由于外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。

看门狗模块采用 32 位的递减计数器，从一个可编程的加载值减到零。当计数器计数减为 0，如果看门狗模式设为复位，则看门狗模块输出复位信号复位系统；如果看门狗模式设为中断，则触发看门狗中断，如果在设定的清除时间限定内软件仍未清除看门狗中断，则产生复位信号复位系统。

用户可以通过设置看门狗使能位来停止/启动计数器。使能看门狗定时器后，应用程序需要在看门狗产生复位之前进行喂狗。否则 WDT 会产生复位信号复位系统。

WDT 最适合那些要求看门狗在精确计时范围内起作用的应用程序。

21.2. 主要特性

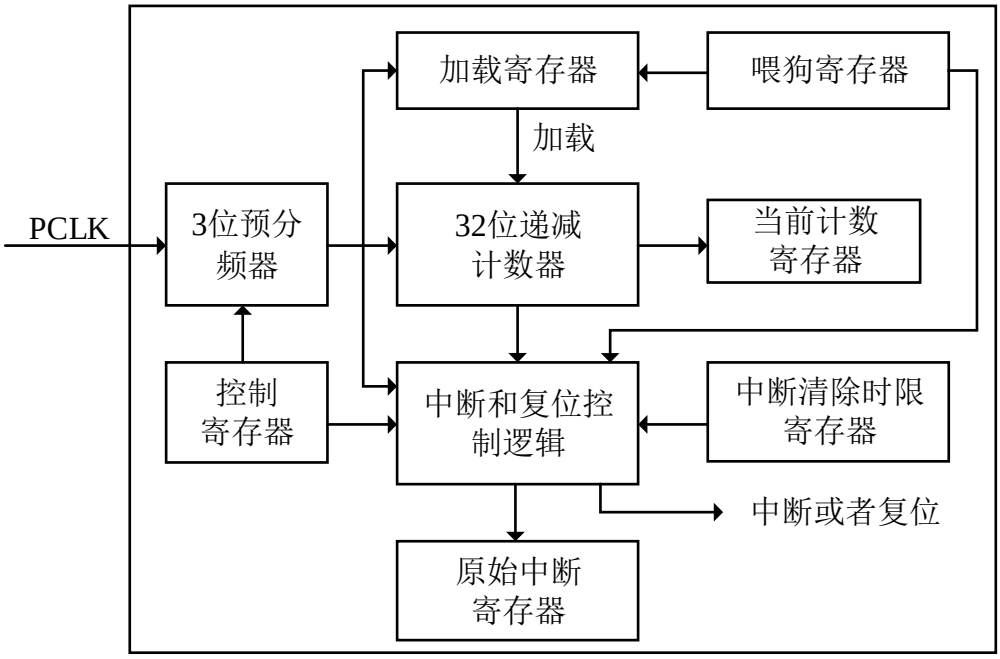
- 32 位的递减计数器
- 可编程预分频
- 可编程装载值
- 可编程中断清除时限
- 条件复位
  - 看门狗模式为复位时，当递减计数器等于 0，则产生复位
  - 看门狗模式为中断时，当递减计数器等于 0，经过中断清除时限后，产生复位。
- 如果启动了看门狗并允许中断，当递减计数器等于 0 时产生中断，在中断清除时限内，可以进行喂狗以避免 WDT 复位。

21.3. 功能描述

加载寄存器（WDT_LOAD）写入需要向下计数的加载值，计数的时钟来自于 PCLK 经过 3 位预分频器后的时钟信号。当看门狗启动时，32 位的递减计数器会从加载值减到 0。看门狗可以产生复位或者中断，可以通过控制寄存器的看门狗模式（WDT_CTRL.MODE）来选择。在产生中断时，如果在清除时限寄存器（WDT_INTCLRTIME）设定的时间内未清除看门狗中断，则产生复位信号复位系统。喂狗寄存器（WDT_FEED）中写入特征值 0xAA55A55A 使计数器从加载寄存器加载装载值，加载动作会清除中断。读取当前计数寄存器(WDT_COUNT)可以查看看门狗定时器的当前计数值。原始中断寄存器（WDT_RIS）保存着看门狗的原始中断状态位。

21.3.1. 结构框图

图 21-1 WDT 结构图



21.3.2. 时钟分频

WDT 时钟由 APB 时钟经预分频后提供，预分频是由一个 3 位的预分频器实现。可以设置 WDT_CTRL.DIVISOR 来实现 1/2/4/8/16/32/64/128 分频。

计算看门狗超时的公式如下：

$$T_{WDT} = T_{PCLK} \times 2^{DIVISOR} \times (LOAD + 1)$$

其中：

T_{WDT}：WDT 的超时时间

T_{PCLK}：APB 时钟的周期

21.3.3. 启动看门狗

系统复位后，看门狗总是处于关闭状态。设置 WDT_CTRL.EN 位能够开启看门狗。

21.3.4. 递减计数器

32 位的递减计数器在看门狗使能(WDT_CTRL.EN 置位)后，会进行递减计数。递减到 0 时会产生复位或者中断。在产生复位之前或者中断清除时限内进行了喂狗，则递减计数器会装载加载寄存器(WDT_LOAD)里的值，然后以加载值进行递减计数。

21.3.5. 中断模式

当计数器从 WDT_LOAD 寄存器设定的值计数减为 0，如果看门狗模式设为中断(WDT_CTRL.MODE 置位)并且使能中断(WDT_CTRL.INTEN 置位)，则触发看门狗中断。在产生中断时，如果在清除时限寄存器（WDT_INTCLTTIME）设定的时间内未清除看门狗中断，则产生复位信号，复位系统。清除看门狗中断需要在中断服务程序中进行喂狗，喂狗后才会退出中断服务程序。

当计数器从 WDT_LOAD 寄存器设定的值计数减为 0，如果看门狗模式设为中断(WDT_CTRL.MODE 置位)，但中断未使能(WDT_CTRL.INTEN 复位)，则会产生原始中断。如果在中断清除时限内未进行喂狗，也会产生复位信号，复位系统。

注：如果需要看门狗复位信号复位系统，需要将 RCC 模块的 RCC_RCR. WDTRSTEN 置位。

计算中断清除时限的公式如下：

$$T_{INTCLRT} = T_{PCLK} \times 2^{DIVISOR} \times (INTCLRT + 1)$$

其中：

$T_{INTCLRT}$：WDT 中断清除时限

T_{PCLK}：APB 时钟的周期

21.3.6. 复位模式

当计数器从 WDT_LOAD 寄存器设定的值计数减为 0，如果看门狗模式设为复位(WDT_CTRL.MODE 复位)，则看门狗模块输出复位信号，复位系统。

注：如果需要看门狗复位信号复位系统，需要将 RCC 模块的 RCC_RCR. WDTRSTEN 置位。

21.4. 配置流程

21.4.1. 中断模式

1）配置加载寄存器(WDT_LOAD)；

2）配置定时器时钟预分频值(WDT_CTRL.DIVISOR)；

3）配置看门狗工作模式(WDT_CTRL.MODE = 1)；

4）配置中断使能位(WDT_CTRL. INTEN = 1)；

5）配置中断清除时限寄存器(WDT_ INTCLRTIME)；

6）使能看门狗定时器(WDT_CTRL.EN = 1)。

注：如果没有使能中断，则在计数到 0 溢出，原始中断会产生，经过中断清除时限寄存器值个计数后，将产生一个系统复位。工作模式为中断但没有使能中断，计数到 0 溢出，原始中断还是会产生的。产生了原始中断后就会去计数清除时限。清除时限计数到 0，就会产生一个系统复位。

21.4.2. 复位模式

1）配置加载寄存器(WDT_LOAD)；

2）配置定时器时钟预分频值(WDT_CTRL.DIVISOR)；

3）配置看门狗工作模式(WDT_CTRL.MODE = 0)；

4）配置中断使能位(WDT_CTRL.INTEN = 0)；

5）使能看门狗定时器(WDT_CTRL.EN = 1)。

注：计数器计数到 0 溢出，将产生一个系统复位。

21.5. WDT 寄存器描述

21.5.1. 寄存器列表

WDT 寄存器基地址：0x40002C00

偏移	名称	复位值	描述
0x00	WDT_LOAD	0xFFFFFFFF	加载寄存器
0x04	WDT_COUNT	0xFFFFFFFF	当前计数寄存器
0x08	WDT_CTRL	0x00000006	控制寄存器
0x0C	WDT_FEED	0x00000000	喂狗寄存器
0x10	WDT_INTCLRTIME	0x00001000	中断清除时限寄存器
0x14	WDT_RIS	0x00000000	原始中断状态寄存器

21.5.2. 加载寄存器(WDT_LOAD: 00h)

位域	名称	属性	复位值	描述
31:0	LOAD	RW	0xFFFFFFFF	加载值

21.5.3. 当前计数寄存器(WDT_COUNT: 04h)

位域	名称	属性	复位值	描述
31:0	COUNT	RO	0xFFFFFFFF	当前计数值。

21.5.4. 控制寄存器(WDT_CTRL: 08h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	EN	RW	0	看门狗使能 0：禁止看门狗 1：使能看门狗
6	MODE	RW	0	看门狗模式 0：复位 1：中断
5	RSV	-	-	保留

4	INTEN	RW	0	中断使能 0：禁止中断 1：使能中断
3	RSV	-	-	保留
2:0	DIVISOR	RW	110	看门狗时钟预分频： 000: 不分频 001: 2 分频 010: 4 分频 011: 8 分频 100: 16 分频 101: 32 分频 110: 64 分频 111: 128 分频

21.5.5. 喂狗寄存器(WDT_FEED: 0Ch)

位域	名称	属性	复位值	描述
31:0	FEED	WO	0x00000000	喂狗 写入 0xAA55A55A，计数器从装载寄存器加载装载值，加载动作会清除 WDT 中断。

21.5.6. 中断清除时限寄存器(WDT_INTCLRTIME: 10h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	INTCLRT	RW	0x1000	中断清除时限 中断产生后（即使中断未使能），在这个时限内没有去喂狗，WDT 会产生复位信号。时限的单位是看门狗工作时钟。

21.5.7. 原始中断状态寄存器(WDT_RIS: 14h)

位域	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	WDTRIS	RO	0	原始中断状态标志位。

22. 独立看门狗（IWDT）

22.1. 概述

本芯片内置两个看门狗外设（独立看门狗和系统看门狗），具有安全性高、定时准确及使用灵活的优点。两个看门狗外设均可用于检测并解决由软件错误导致的故障；当计数器达到给定的超时值时，触发一个中断（仅适用于系统看门狗）或产生系统复位。

独立看门狗（IWDT）由专用低速时钟（RCL）驱动，因此即便在主时钟发生故障时仍然保持工作状态。系统看门狗（WDT）时钟由 APB 时钟经预分频后提供，检测应用程序非正常的过迟的操作。

IWDT 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。WDT 最适合那些要求看门狗在精确计时起作用的应用程序。

关于系统看门狗的详情，请参看“系统看门狗（WDT）”章节。

22.2. 主要特性

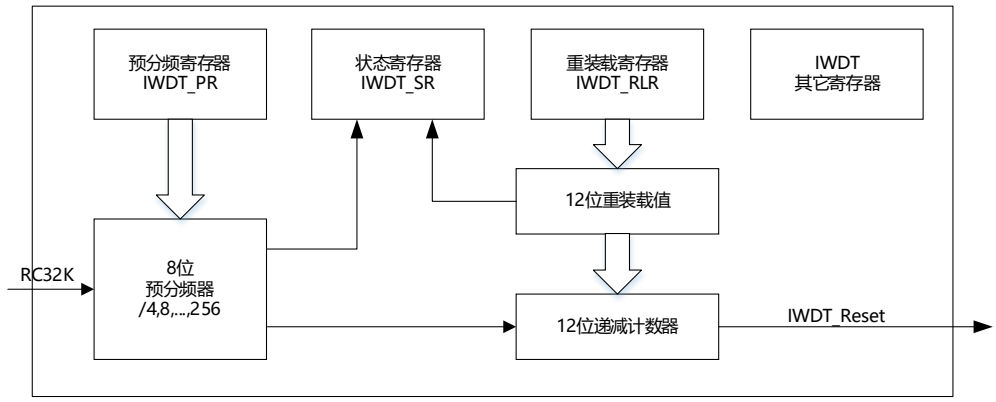
- 自由运行的 12 位向下计数器
- 看门狗被激活后，计数器计数至 0x00 时产生复位
- 当递减计数器在窗口外被重新装载，则产生复位
- 可编程预分频因子和可编程装载值
- 时钟由独立的 RC32K 时钟提供(可在停止和待机模式下工作)
- 可作为 STOP 模式唤醒源

22.3. 功能描述

22.3.1. 结构框图

结构框图如下：

图 22-1 IWDT 结构框图



在命令寄存器(IWDT_CMDR)中写入 0xCCCC，开始启用独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号(IWDT_RESET)。无论何时，只要在控制寄存器 IWDT_CMD 中写入 0xAAAA，IWDT_RLR 中的值就会被重新加载到计数器，从而避免产生看门狗复位。

22.3.2. 时钟

独立看门狗（IWDT）由其专用低速时钟（RC32K）驱动，因此即便在主时钟发生故障时仍然保持工作状态。IWDT 和 RC32K 都由 VDD 供电，在低功耗模式 STOP 或者 STANDBY 都能正常工作，可以让系统在异常状态唤醒。RC32K 通过设置 RCC_STDBY_CTRL 的 RC32EN 为高，使能 RC32K 时钟

22.3.3. 预分频

IWDT 时钟通过预分频器再进行计数，具体预分频设置由 IWDT_PR 寄存器确定。超时值计算为：

tIWDT=(RL[11:0]+1)*tRC32K*预分频系数

下表为根据不同预分频系数的超时时间。

表 22-1 独立看门狗超时时间（RC32K 输入时钟）

预分频系数	PR[2:0]	最短时间(ms) RL[11:0]=0x000	最长时间（ms） RL[11:0]=0xFFF
1/4	0	0.125	512
1/8	1	0.25	1024
1/16	2	0.5	2048
1/32	3	1	4096
1/64	4	2	8192
1/128	5	4	16384
1/256	6 或 7	8	32768

注：这些时间均针对 32kHz 时钟给出。实际上，MCU 内部的 RC 频率会在 30kHz 到 45kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。

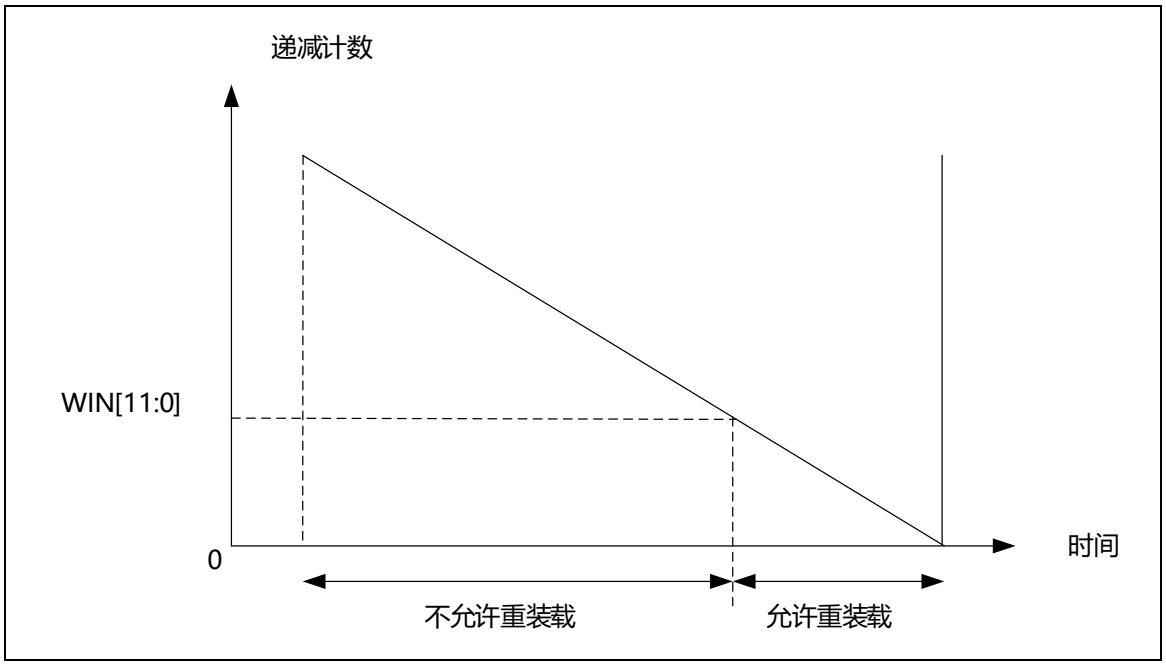
22.3.4. 寄存器访问保护

IWDT_PR、IWDT_RLR、IWDT_WINR 和 IWDT_WUTR 寄存器具有写保护功能。要修改这几个寄存器的值，必须先向 IWDT_CMDR 寄存器中写入 0x5555 打开写保护。写保护打开后可以对多个受保护寄存器进行修改。直到再次往该寄存器写入一个不同的值来中断操作流程，此时寄存器将重新被保护。重装载操作(即写入 0xAAAA)也会启动写保护功能。状态寄存器指示 IWDT_PR、IWDT_RLR、IWDT_WINR 和 IWDT_WUTR 寄存器是否正在被更新。

22.3.5. 窗口选项

独立看门狗定时器通过配置窗口寄存器 IWDT_WINR 中的 WIN[11:0]位用来设定窗口值。当计数值大于 WIN[11:0]中的值，重装载操作(IWDT_CMDR 写入 0xAAAA)会引起复位；只有当计数器的值小于窗口值，重装载向下计数器可以避免复位。开启窗口功能后，超时时间计算不会改变。下图为窗口功能示意图。

图 22-2 窗口区域示意图



IWDT_WINR 的默认值为 0xFFF，与重装载值 IWDT_RLR 一样，因此默认不开启窗口功能。只有当 IWDT_WINR 设置为小于 IWDT_RLR 时，窗口功能才会开启。

配置 IWDT_WINR 会自动发起计数器变为为重装载值 PL。

22.3.6. 唤醒功能

独立看门狗定时器有独立唤醒功能，开启后可以将 MCU 从 STOP 模式唤醒，防止产生复位。IWDT_WUTR 寄存器可以设置唤醒值，当递减计数到预设唤醒值后，会产生一个预分频周期的 WAKEUP 信号，通过 EXTI 模块唤醒系统。IWDT_WUTR 设置需要小于 IWDT_RLR，否则唤醒功能无法工作。此功能默认不开启，对 IWDT_CMD 写入 0x6666 开启；开启后对 IWDT_CMD 写入 0x9999 关闭。

唤醒时间计算公式： tIWDT_WU=(RL[11:0]- WUT [11:0]+1)*tRC32K*预分频系数

其中预分频系数参见预分频章节。

为了在 STOP 模式唤醒，还需要配置 EXTI 使能这一功能，步骤如下

1）设置 EXTI_IERN 的 bit19 为 1，使能 IWDT 唤醒功能

2）设置 EXTI_RTENR 的 bit19 为 1，使能上升沿唤醒

3）进入 STOP，等待 IWDT 唤醒

4）IWDT 唤醒后，EXTI_PDR 的 bit19 为高，软件查询后写 1 清除。

22.3.7. 低功耗

IWDT 由 VDD 供电，在低功耗模式 STOP 或者 STANDBY 都能正常工作，可以让系统唤醒（STOP）或者异常状态恢复（STANDBY&STOP）。

STOP 模式唤醒功能参见上一章节（唤醒功能）。

STANDBY 模式下，IWDT 超时复位会把系统从低功耗模式唤醒。IWDT 唤醒后，PMU_SR 寄存器的 IWDT 唤醒标志位(IWDTWUF)会置 1；PMU_STCLR 寄存器的 CIWDTWUF 位会清除 IWDTWUF 位。

22.3.8. IWDT 复位

IWDT 计数器超时产生系统复位，只对主区有效，对 STANDBY 区域无效，也不复位 EFC 控制器、RCC 寄存器。

为了使能 IWDT 复位，首先要配置 RCC 模块的相关位，具体步骤如下

1）设置 RCC_STDBY_CTRL 的 RC32EN 为高，使能 RC32K 时钟

2）设置 RCC_RCR 的的 IWDTRST_EN 为高，使能独立看门狗信号复位系统

IWDT 的配置参见配置流程。

22.4. 配置流程

1）使能 IWDT

WDT_CMDR 写入 0xCCCC。

2）禁止写保护

WDT_CMDR 写入 0x5555。

3）配置预分频值

IWDT_PR 写入预分频值。

4）配置重装载值

等待 IWDT_SR 的 PVU 为 0，IWDT_RLR 写入重装载值。

5）配置唤醒值

如果启用唤醒功能：

6）等待 IWDT_SR 的 RVU 为 0，IWDT_WUTR 写入唤醒值，配置唤醒模式（EXTI）。

如果未启用唤醒功能：

7）等待 IWDT_SR 的 RVU 为 0，IWDT_WUTR 写入 0xFFF。

配置默认窗口值

8）如果未启用窗口功能：

等待 IWDT_SR 的 WVU 为 0，IWDT_WINR 写入 0xFFF。

9）等待所有配置完成

等待 IWDT_SR 为 0。

10）使能唤醒功能

如果启用唤醒功能：

11）IWDT_CMDR 写入 0x6666。

配置窗口值

12）如果启用窗口功能

IWDT_WINR 写入窗口值。

13）重装载

IWDT_CMDR 写入 0xAAAA。

14）如果使能了窗口功能，请在计数器的值小于窗口值时喂狗，参见窗口选项。

等待所有配置完成

15）等待 IWDT_SR 为 0。

22.5. IWDT 寄存器描述

22.5.1. 寄存器列表

IWDT 寄存器基地址：0x40003000

偏移	名称	复位值	描述
0x00	IWDT_CMDR	0x00000000	命令寄存器
0x04	IWDT_PR	0x00000000	预分频寄存器
0x08	IWDT_RLR	0x00000fff	重装载寄存器
0x0C	IWDT_SR	0x00000000	状态寄存器
0x10	IWDT_WINR	0x00000fff	窗口寄存器
0x14	IWDT_WUTR	0x00000fff	唤醒值设置

22.5.2. 命令寄存器(IWDT_CMDR: 00h)

位域	名称	属性	复位值	描述
31:0	CMD	WO	0x00000000	命令 0xAAAA：喂狗，重装载计数器。 0x5555：禁止写保护。IWDT_PR、IWDT_RLR 和 IWDT_WINR 三个寄存器具有写保护，在写寄存器之前需取消写保护。 0xCCCC：使能 IWDT 0xEF01ABCD：禁止 IWDT。 0x6666：使能唤醒功能。 0x9999：禁止唤醒功能。

注：请在启动独立看门狗后再配置寄存器。

22.5.3. 预分频寄存器(IWDT_PR: 04h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留

2:0	PR	R/W	000	预分频因子 000: 4 分频 001: 8 分频 010: 16 分频 011: 32 分频 100: 64 分频 101: 128 分频 110: 256 分频 111: 256 分频 IWDT_SR 寄存器的 PVU 位为 0 时，才允许对 PR 进行读写操作。

注：IWDT_PR 具有写保护功能，配置 IWDT_PR 前，需禁止写保护（向 IWDT_CMDR 寄存器中写入 0x5555)。

22.5.4. 重装载寄存器(IWDT_RLR: 08h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	PL	R/W	0xFFF	重装载值 当向 IWDT_CMDR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。独立看门狗超时周期可通过此重装载值和时钟预分频值来计算。 IWDT_SR 寄存器的 RVU 位为 0 时，才允许对 PL 进行读写操作。

注：IWDT_RLR 具有写保护功能，配置 IWDT_RL 前，需禁止写保护（向 IWDT_CMDR 寄存器中写入 0x5555)。

22.5.5. 状态寄存器(IWDT_SR: 0Ch)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
4	RLF	RO	0	喂狗完成状态 0：喂狗完成 1：喂狗进行中 进入 STOP 模式前，需要等待喂狗完成。
3	WTU	RO	0	唤醒值更新状态 0：唤醒值更新完成 1：唤醒值更新进行中
2	WVU	RO	0	窗口值更新状态 0：窗口值更新完成 1：窗口值更新进行中
1	RVU	RO	0	重装载值更新状态 0：重装载值更新完成 1：重装载值更新进行中

0	PVU	RO	0	预分频值更新状态 0：预分频值更新完成 1：预分频值更新进行中

22.5.6. 窗口寄存器(IWDT_WINR: 10h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	WIN	R/W	0xFFF	窗口值 定义窗口值的上限值，用于与向下递减计数器进行比较。当计数值大于 WIN 值，重装载操作会引起复位。当计数值小于等于 WIN 值，重装载操作不会引起复位。 IWDT_SR 寄存器的 WVU 位为 0 时，才允许对 WIN 进行读写操作。

注：IWDT_WINR 具有写保护功能，配置 IWDT_WINR 前，需禁止写保护（向 IWDT_CMDR 寄存器中写入 0x5555)。

22.5.7. 唤醒寄存器(IWDT_WUTR: 14h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	WUT	R/W	0xFFF	唤醒值 定义唤醒值，用于定义唤醒信号产生的计数值，产生一个预分频周期的 WakeUp 信号。 IWDT_SR 寄存器的 WTU 位为 0 时，才允许对 WUT 进行读写操作。

注：IWDT_WUTR 具有写保护功能，配置 IWDT_WUTR 前，需禁止写保护（向 IWDT_CMDR 寄存器中写入 0x5555)。

23. 实时时钟（RTC）

23.1. 概述

实时时钟（RTC）是一个独立的 BCD 定时器/计数器。RTC 提供一个日历时钟、一个可编程闹钟中断，以及一个具有中断功能的周期性可编程唤醒标志。RTC 还包含用于唤醒低功耗模式的自动唤醒单元。

RTC 提供二进码十进数格式（BCD)的秒、分、时、日、周、月、年，并自动将月份的天数补偿为 28、29（闰年)、30 和 31 天。

RTC 使用数字校准功能对晶振精度的偏差进行补偿。 上电复位后，所有 RTC 寄存器都会受到保护，以防止可能的非正常写访问。 无论器件状态如何（运行模式、低功耗模式或处于复位状态)，只要电源电压保持在工作范 围内，RTC 便不会停止工作。

23.2. 主要特性

- 包含秒、分钟、小时（24 小时制)、星期几、日期、月份和年份的日历。
- 自动闰年调整
- 数字校准功能：通过调整最小时间单位（最大可调精度 0.95ppm）来进行日历校准，调校后理论精度+/-0.477ppm
- 自动唤醒单元，可周期性地生成标志以触发自动唤醒中断，
- 16 位可编程自动重载递减定时器
- 闹钟功能，可通过任意日历字段的组合驱动闹钟。
- RTC 计时器部分不复位
- 入侵检测：
  - 2 个带可配置滤波的入侵检查
  - 入侵事件可配置为上升沿和下降沿，并可配置清除备份寄存器
  - 入侵事件可产生时间戳
- 16 个 32 位（共 64 字节）通用备份寄存器，能够在省电模式下保存数据。当有外部侵入事件发生时，备份寄存器可复位（可配置)。
- RTC 计数时钟可在 XTL 和 RC32K 中选择
- RTC_OUT 可在多个信号选择：秒方波、秒计数进位、闹钟匹配、周期唤醒
- 可屏蔽中断/事件：
  - 闹钟
  - 周期唤醒
  - 侵入事件

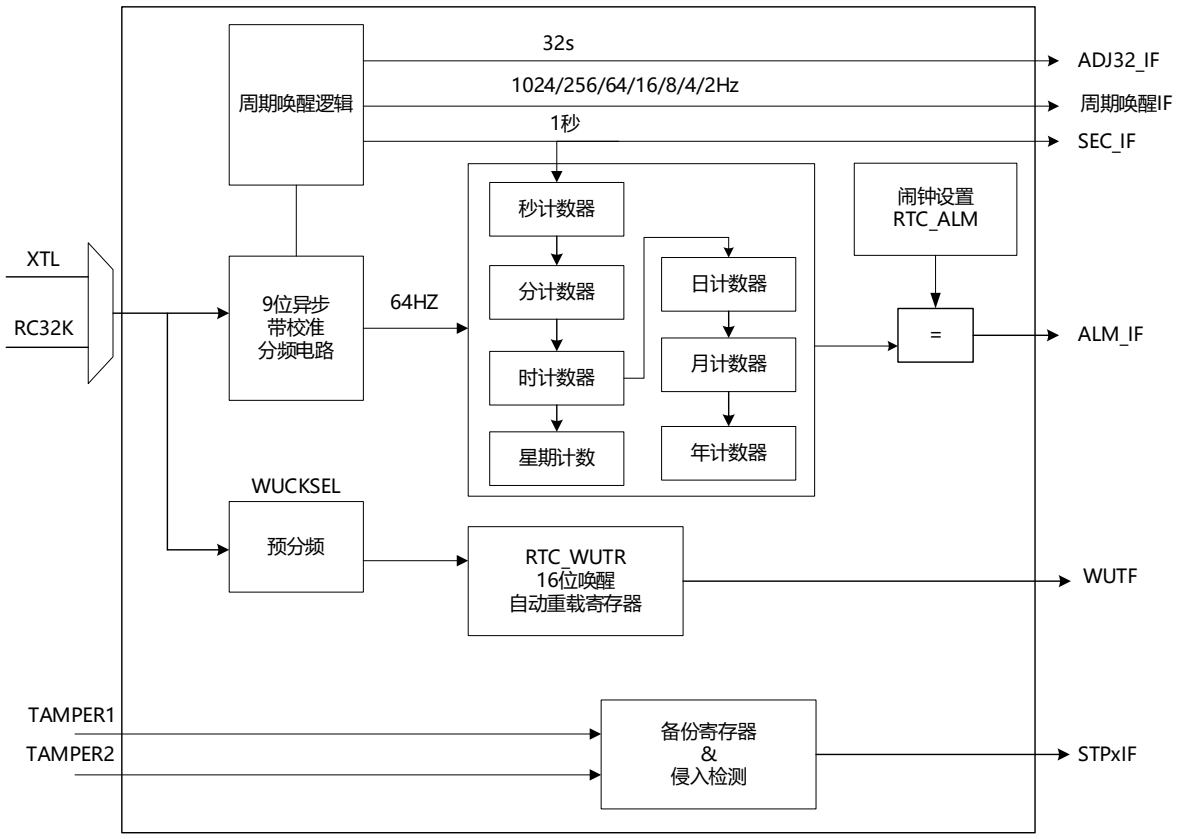
23.3. 功能描述

实时时钟的时钟源可配置为外部低速晶振 XTL 和内部低速 RC32K；默认使用内部低速 RC32K。所有软件写入和读取的日期时间值都为 BCD 码，无须十六进制转换为十进制，如 27 日直接写入 0x27 等。

23.3.1. 结构框图

整体框图如下：

图 23-1 RTC 内部结构框图



23.3.2. 复位过程

RTC 的写保护寄存器 RTC_WP、中断使能寄存器 RTC_IE、中断标志寄存器 RTC_SR、闹钟寄存器 RTL_ALARM、控制寄存器 RTC_CR、时间戳寄存器 RTC_CLKSTAMPx、日历戳寄存器 RTC_CALSTAMPx、唤醒定时寄存器 RTC_WURT 和备份寄存器 RTC_BAKUPx 只受上电复位和 STANDBY 电源域软复位（STDBY_CTRL.STDBYRST）复位，其他复位源不能复位这几个控制寄存器。RTC 其它寄存器没有复位控制，不受任何复位影响，上电状态不定，上电后需要初始化。

当以下事件中之一发生时，产生备份区（包括 RTC）复位。

1）软件复位，备份区域复位可由设置 STANDBY 电源域软复位（STDBY_CTRL 的 STDBYRST）位产生。

2）在 VDD 掉电的前提下，VDD 上电将引发备份区域复位。

23.3.3. 时间和日历

■ RTC 寄存器写保护

上电复位后，可通过 PMU 控制寄存器（PMU_CTRL0）的 RTC_WE 位保护 RTC 寄存器以防止非正常的写访问,必须将 RTC_WE 位置 1 才能使能 RTC 寄存器的写访问。

上电复位后，时间和日历寄存器（包括秒、分、时、日、周、月、年和亚秒计数器）受到写保护。通过向写保护寄存器（RTC_WP）写入一个密钥来使能对 RTC 寄存器的写操作。解锁时间和日历寄存器的写保护，需要执行以下步骤： 将 0xCA53CA53 写入 RTC_WP 寄存器。

写入一个错误的关键字会再次激活写保护。保护机制不受系统复位影响。

■ RTC 时间设置

解除 RTC 的写保护后，可以写入新的时间和日历，写入后后续计数就从新写入的时间和日历开始。软件应在秒中断发生后再去设置时间，计时更加准确。当软件写入秒时间时，硬件自动清除秒内计数器。

■ RTC 时间读取

有两种读取方式

● 方式 1：任意时刻读取方式

1）读出分，时，周，日，月，年计数寄存器值；

2）读出秒计数寄存器值；

3）再次读出秒计数寄存器值；

4）判断两次秒的读出值是否相同，不同重新从第一步开始， 相同读取结束。

● 方式 2：中断读取方式

1）在 RTC 周期中断服务中读取秒，分， 时，周，日， 月，年计数寄存器值。因为中断发生后到下次数据改变至少 1s 的时间。

23.3.4. 闹钟功能

闹钟功能可以设定一个星期内，任意几天（周一到周日）定时唤醒；也可以设定一个月内，任意某一天定时唤醒，可以精确到秒级别。软件先设置好闹钟寄存器 RTC_ALARM，包括闹钟的秒 ALMSEC、闹钟的分 ALMMIN、闹钟的时 ALMHOUR、闹钟星期/日数值 ALMWEEK/ALMDAY、闹钟星期/日模式 ALM_WDS。当 RTC 日历中的秒计数器，分计数器、小时计数器、周计数器（闹钟星期模式适用）、日计数器（闹钟日模式）的数值分别与闹钟寄存器的对应值相等时，产生闹钟中断。

RTC_CR 中的闹钟星期/天数值功能屏蔽位 ALM_MSKD、闹钟时数值屏蔽位 ALM_MSKH、闹钟分数值屏蔽位 ALM_MSKM 分别忽略闹钟星期/天、时和分的比较功能，在某一屏蔽位设为 1 后闹钟就不比较该位屏蔽的数值，只需要当非屏蔽数值相等时，就产生闹钟中断。比如 ALM_MSKD 为 1，则当秒计数器、分计数器、小时计数器的数值分别与闹钟秒 ALMSEC、闹钟的分 ALMMIN、闹钟的时 ALMHOUR 值相等时，就产生闹钟中断，不需要周计数器或日计数器的数值与闹钟的闹钟星期/日数值 ALMWEEK/ALMDAY 值相等，可以作为每日闹钟功能。

23.3.5. 时钟误差补偿

由于计数器采用 32.768KHz 的时钟计数，如果需要对每秒精度进行补偿时，只能按照 32.768KHz 的整数周期补偿，则每秒补偿的最小单位为 （1/32768） *106=30.5ppm，无法满足高精度的要求。

那么要在 32.768KHz 的计数时钟下实现精度较高的时钟补偿时，需要在算法上做调整，将最大补偿周期扩大 32 倍。则在只能补偿的最小单位为 30.5ppm 的情况下，平均到每秒的补偿单位变为 30.5ppm/32=0.96ppm，最大调校范围为+/-(511*30.517us/32s)=+/-487ppm。满足了精度较高的时钟补偿要求。而且补偿发生在每 32 秒内比较均匀的范围内。

调校值由 10bit 寄存器组成，其中最高位为符号位，表示计数值增减，其余 9bit 表示增减的绝对值。为了提高每秒的平均精度，避免较大的秒间跃变，采取将 32s 调校值平均分配到每秒内的做法，其实现方法如下：

除了最高符号位，其余 9bit 可分为高 4bit 的公共值和 5bit 私有值，其中公共值表示 32s 内每秒都要调整的值，私有值表示 32s 内部分秒需要加减 1。

Bit9	Bit[8:5]	Bit[4:0]
Sign	Common Value (C)	Differential Value (D)

调校值公式可表示为：Correction = (C*32 + D)*30.517/32000000

假设只使时钟增加 0.952ppm，即 32s 周期只增加一个 30.5us，调校值写为 0_0000_00001，所以公共值为 0，私有值为 1，只需要对 32s 内的一个秒周期加 1 即可；假设增加 487ppm，即 32s 周期内增加 511 个 30.5us，调校值写为 0_1111_11111，公共值为 15，私有值为 31，表示 32s 中每秒都要加 15，同时其中还有 31s 需要额外加 1。

调校值举例：

ppm	ADJUST	Common	Differential	Expression
0.953	0_0000_00001	0	1	1*30.517/32000000
-125.88	1_0100_00100	4	4	(4*32+4)*30.517/32000000
32.42	0_0001_00010	1	2	(1*32+2)*30.517/32000000
487	0_1111_11111	15	31	(15*32+31)*30.517/32000000

通过以上方式，可以得到平滑调整的秒周期，秒和秒之间最大只差 30.5ppm，可以避免集中式调整引入的秒周期抖动。

23.3.6. RTC 输出

RTC 模块可以从 RTC Fout 输出内部的频率信号或者闹钟信号到管脚 PC13 上用于调试和校准，具体输出信号由 RTC_CR 寄存器的 FSEL 选择。

为了使 RTC Fout 输出到 PC13，需要进行如下设置

1）设置 PMU 的 STANDBY 域 IO 复用寄存器 PMU_IOSEL 的 PC1_SEL[1:0]为 2’ b01 选择 RTC Fout

2）设置 PMU_IOSEL 的 PC13_Value 选择输出驱动模式，0 作为 OD 输出，1 为 push-pull 输出。

23.3.7. 周期中断唤醒

RTC 模块提供多种周期唤醒，除了包括 1 秒、1 分、1 小时、1 天日历唤醒外，还提供 1024HZ、256Hz、64Hz、16Hz、8Hz、4Hz、2Hz 和 32s 等多种唤醒方式。

此外，RTC 模块提供一个可任意设置的定时唤醒定时器（WUTIMER），由 16 位可编程自动重载递减定时器组成。可通过 RTC_CR 寄存器中的 WUTE 位来使能此唤醒定时器功能。

唤醒定时器的时钟输入可以是：

- 2、 4、 8 或 16 分频的 RTC 时钟 (RTCCLK)。当 RTCCLK 为 XTL (32.768 kHz) 时，可配置的唤醒中断周期介于 122 µs 和 32 s 之间，且分辨率低至 61 µs。
- 0.5Hz、1Hz、2Hz 信号，可配置的唤醒中断周期为 0.5s 和 36 小时之间。

完成初始化后，定时器开始从 RTC_WUTR 递减计数。在低功耗模式下使能唤醒功能时，递减计数保持有效。此外，当计数器计数到 0 时，RTC_SR 寄存器的 WUTF 标志会置 1，并且唤醒寄存器会使用其重载值（RTC_WUTR 寄存器值）自动重载。之后必须用软件清零 WUTF 标志。

通过将 RTC_IE 寄存器中的 WUTIE 位置 1 来使能周期性唤醒中断时，它会使器件退出低功耗模式。

要配置或更改唤醒定时器的自动重载值（RTC_WUTR 中的 WUT[15:0]），需要按照以下顺序操作：

1）清零 RTC_CR 中的 WUTE 以禁止唤醒定时器。

2）轮询 RTC_SR 中的 WUTWF，直到该位置 1，以确保可以访问唤醒自动重载定时器和 WUCKSEL[2:0] 位。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。

3）编程唤醒自动重载值 WUT[15:0]，并选择唤醒时钟（RTC_CR 中的 WUCKSEL[2:0] 位）。将 RTC_CR 寄存器中的 WUTE 位置 1 以再次使能定时器。唤醒定时器重新开始递减数。由于时钟同步的缘故，在 WUTE 置 1 后， WUTWF 位也会清零，但需要花费 2 个 RTCCLK 时钟周期

如果已通过 RTC_CR 寄存器的 FSEL 位选择 WUT 周期性唤醒连接到 RTC Fout 输出，则会通过相应的管脚输出 WUT 唤醒信号。

23.3.8. 侵入检测和时间戳

RTC 支持两个外部 IO 侵入事件检测，侵入检查通过 TAMPxEN 使能对应 IO 检测功能。通过设置 RTC_CR 的 TAMP1EN 和 TAMP2EN 分别使能两个 IO 的侵入检测功能。当 TAMPER 引脚上的信号从 0 变成 1（上升沿）或者从 1 变成 0（下降沿）， 会产生一个侵入检测事件。侵入检测事件将所有数据备份寄存器内容清除，设置 RTC_CR 的 TAMPxFCLR 为高使能下降沿清除备份寄存器，设置 RTC_CR 的 TAMPxRCLR 使能为高使能下降沿清除备份寄存器，两者可以同时使能。

侵入 IO 有上升沿相应的标志，可用于产生中断或者供软件查询。下降沿侵入事件标志为 RTC_SR 的 STP2FIF；上升沿侵入事件标志为 RTC_SR 的 STP2RIF。设置 RTC_IE 寄存器的 STPxFIE 位为’1’，当检测到侵入事件下降沿就会产生一个中断；设置 RTC_IE 寄存器的 STPxRE 位为’1’，当检测到侵入事件上升沿就会产生一个中断。

注： 当内核电源断开时（进入 STANDBY 模式），侵入检测功能仍然有效。为了避免不必要的复位数据备份寄存器，TAMPER 引脚应该在片外连接到正确的电平。

两个侵入 IO 各有一组独立 STAMP 寄存器组（RTC_CLKSTAMPx 和 RTC_CALSTAMPx），通过设置 RTC_CR 的 TSxEDGE 选择记录上升沿（TSxEDGE=0）或者下降沿（TSxEDGE=1）时间戳，当侵入 IO 出现选择的上升沿或下降沿时，RTC 会自动记录当前时间到 STAMP 寄存器组中。侵入检查时间戳在相应标志寄存位（下升沿标志为 STP2FIF，上升沿标志为 STP2RIF）为 0 的情况下记录事件发生时间，如果对应标志已经为 1，则忽略相应事件。因此如果有多次事件发生，时间戳仅记录第一次事件发生的时间，除非软件在事件发生后清除了标志位。

为了防止 TAMPER 引脚的毛刺导致误触发侵入事件，建议使能外部 IO 的输入数字滤波，通过 RTC_CR 的 TAMPxFLTEN 使能某个通道滤波功能，设置 TAMPxFLT 选择滤波周期。滤波时钟可以选择 RTC 时钟或者 64HZ 时钟，滤波时钟两个 IO 使用同一种配置。

RTC 支持两个 TAMPER 输入，其中 TAMPER1 来自 PC13，TAMPER2 来自 PA0，IO 设置如下。

TAMPER1 的 IO 设置，设置 PMU 的 STANDBY 域 IO 复用寄存器 PMU_IOSEL 的 PC1_SEL[1:0]为 10 选择侵入检测功能。

TAMPER1 的 IO 设置，设置 PA0 为模拟功能。

23.3.9. 备份寄存器

备份寄存器 RTC_BAKUP 是 16 个 32 位的寄存器，可用来存储 64 个字节的用户应用程序数据。它们处在备份域 里，只有当 VDD 电源被切断，备份寄存器数值才会丢失。当系统在待机模式下被唤醒，或系统复位或 BOR 复位时，它们也不会被复位。

复位后，对备份寄存器和 RTC 的访问被禁止，并且备份域被保护以防止可能存在的意外的写操 作。执行以下操作可以使能对备份寄存器和 RTC 的访问。

1）通过设置寄存器 RCC_APB1_IPCKENR 的 PMUCKEN 和 RTCCKEN 位来打开 PMU 和 RTC 模块的时钟

2）PMU 控制寄存器（PMU_CTRL0）的 RTC_WE 位来使能对后备寄存器和 RTC 的访问。

备份寄存器处于备份域中。待机模式唤醒或系统复位操作都不会影响这些寄存器。只有当被检测到有侵入事件和备份域复位时，这些寄存器才会复位。

23.3.10. RTC 低功耗

低功耗模式对 RTC 的作用

模式	说明
睡眠	无影响 RTC 中断可使芯片退出睡眠模式
停止	RTC 寄存器内容保持，当 RTC 时钟源为 XTL 或 RC3K 时，RTC 保持工作状态。RTC 闹钟、RTC 入侵事件、RTC 时间戳事件、和 RTC 唤醒会使器件退出停机模式
待机	RTC 寄存器内容保持，当 RTC 时钟源为 XTL 或 RC3K 时，RTC 保持工作状态。RTC 闹钟、RTC 入侵事件、RTC 时间戳事件、和 RTC 唤醒会使器件退出停机模式

23.3.11. RTC 中断

RTC 的所有中断标志 RTC_SR 都可以产生系统中断、STOP 模式唤醒和 STANDBY 模式唤醒，通过 RTC_IE 的使能选择某几个中断标志。RTC_IE 的使能位对中断、STOP 唤醒和 STANDBY 唤醒都有效。RTC 模块产生一个全局中断信号 RTC_INT，为所有使能的中断标志相或。RTC_INT 在系统级连接到不同模块作为不同功能，连接到 NVIC 的中断源 2 作为中断功能，连接到 EXTI 的 EXTI 线 17 作为 STOP 唤醒，连接到 PMU 作为 STANDBY 唤醒。

■ 中断模式配置

1）配置 NVIC 的 RTC 通道（INT2）并将其使能

2）使能 RTC 的某一功能并使能相应中断

■ STOP 唤醒配置

1）将 EXTI 线 17 配置为中断模式使能或事件模式使能，然后选择上升沿有效

2）使能 RTC 的某一功能并使能相应中断

■ STANDBY 模式唤醒

1）RTC_INT 作为 STANDBY 模式不需要进行系统级设置。

2）使能 RTC 的某一功能并使能相应中断

中断控制位

中断事件	事件标志	启用 控制位	退出 睡眠模式	退出 停止模式	退出 待机模式
闹钟	ALM_IF	ALM_IE	是	是	是
TAMP1 输入检测	STP1RIF STP1FIF	STP1RIE STP1FIE	是	是	是
TAMP2 输入检测	STP2RIF STP2FIF	STP2RIE STP2FIE	是	是	是
唤醒定时器中断	WUTF	WUTIE	是	是	是
周期唤醒	xHZ_IF SEC_IF	xHZ_IE SEC_IE	是	是	是

23.4. 配置流程

23.4.1. RTC 模块使能流程

1）使能 PMU 模块时钟 RCC. APB1ENR.PMUCKEN=1 和 RTC 模块总线时钟 RCC. APB1ENR.RTCCKEN=1;

2）RTC 模块写使能 PMU.CTL0.RTC_WE=1;

3）选择 RTCCLK 时钟源 RCC.STDBYCTL.RTCSEL，当时钟源选择 RC32K 时，需要使能 RC32K 时钟 RCC. STDBYCTL.RC32EN=1；当选择外部低速晶振 XTL 时，需要使能 XTL 振荡器 RCC. STDBYCTL.XTLEN=1;

23.4.2. RTC 时间设置流程

1）等待秒中断时间发生

2）解除写保护，将 0xCA53CA53 写入 RTC_WP 寄存器

3）连续写入秒、分、时、日 、月、年、星期寄存器

4）读出时间寄存器进行校验

5）使能写保护

23.4.3. RTC 时间读取流程

有两种读取方式

● 方式 1：任意时刻读取方式

1）读出分，时，周，日，月，年计数寄存器值;

2）读出秒计数寄存器值;

3）再次读出秒计数寄存器值;

4）判断两次秒的读出值是否相同，不同重新从第一步开始， 相同读取结束。

● 方式 2：中断读取方式

1）在 RTC 周期中断服务中读取秒，分， 时，周，日， 月，年计数寄存器值。因为中断发生后到下次数据改变至少 1s 的时间。

23.4.4. RTC 闹钟设置流程

1）设值 RTC_CR 的 ALM_EN，关闭闹钟

2）设置 RTC_ALARM 的 ALM_WDS，选择星期模式（0）还是日模式（1）。

3）对于星期模式，设置 RTC_ALARM 的 ALMWEEK[6:0]选择每周几闹钟有效，其中 bit0 为周日，bit1 为 周一，bit6 为周六，可以多选。对于日模式，设置 ALMDAY 为日的 BCD 格式

4）设置 RTC_ALARM 的秒闹钟 ALMSEC，分闹钟 ALMMIN，时闹钟 ALMHOUR，

5）设置 RTC_CR 的闹钟屏蔽位：ALM_MSKD、ALM_MSKH、ALM_MSKM。

6）设置 RTC_IE 的 ALM_IE，闹钟中断使能;

7）设定 ALM_EN 为 1，闹钟许可;

8）当发生闹钟事件，RTC_SR 的 ALM_IF 闹钟标志变为高，产生 RTC 中断，进入 RTC 中断处理闹钟事件。

23.4.5. 唤醒定时器设置流程

1）清零 RTC_CR 中的 WUTE 以禁止唤醒定时器。

2）轮询 RTC_SR 中的 WUTWF，直到该位置 1，以确保可以访问唤醒自动重载定时器和 WUCKSEL[2:0]位。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。

3）编程唤醒自动重载值 WUT[15:0]，并选择唤醒时钟（RTC_CR 中的 WUCKSEL[2:0] 位）。将 RTC_CR 寄存器中的 WUTE 位置 1 以再次使能定时器。唤醒定时器重新开始递减数。由于时钟同步的缘故，在 WUTE 置 1 后， WUTWF 位也会清零，但需要花费 2 个 RTCCLK 时钟周期

4）设置 RTC_IE 的 WUTIE 为高，使能定时唤醒中断。

23.4.6. 侵入检测设置流程

1）设置 RTC_CR 的 TAMP1EN 和 TAMP2EN 为 0，关闭侵入检测功能

2）分别设置 TAMPER 输入的 IO 功能。TAMPER1 的 IO 设置，设置 PMU 的 STANDBY 域 IO 复用寄存器 PMU_IOSEL 的 PC1_SEL[1:0]为 10 选择侵入检测功能。TAMPER1 的 IO 设置，设置 PA0 为模拟功能。

3）设置侵入事件清除备份寄存器功能。设置 RTC_CR 的 TAMPxFCLR 为高使能下降沿清除备份寄存器，设置 RTC_CR 的 TAMPxRCLR 使能为高使能下降沿清除备份寄存器，两者可以同时使能。

4）选择侵入事件的时间戳功能的对应的边沿，设置 RTC_CR 的 TSxEDGE 为 0，选择上升沿，为 1 选择下降沿时间戳。

5）设置侵入信号的 IO 滤波功能，通过 RTC_CR 的 TAMPxFLTEN 使能某个通道滤波功能，设置 TAMPxFLT 选择滤波周期。

6）设置中断使能，设置 RTC_IE 寄存器的 STPxFIE 位为’1’，使能侵入事件下降沿中断；设置 RTC_IE 寄存器的 STPxRE 位为’1’，使能侵入事件上升沿中断。

7）设置 RTC_CR 的 TAMP1EN 和 TAMP2EN 为 1，打开侵入检测功能

8）检测到下降沿侵入事件，RTC_SR 的 STP2FIF 标志变高；检测到上升沿侵入事件，RTC_SR 的 STP2RIF 标志变高，产生 RTC 中断，进入 RTC 中断处理侵入事件。

23.4.7. RTC 中断和唤醒设置流程

RTC 模块产生一个全局中断信号 RTC_INT，为所有使能的中断标志相或。RTC_INT 在系统级连接到不同模块作为不同功能，连接到 NVIC 的中断源 2 作为中断功能，连接到 EXTI 的 EXTI 线 17 作为 STOP 唤醒，连接到 PMU 作为 STANDBY 唤醒。

■ 中断模式配置

1）配置 NVIC 的 RTC 通道（INT2）并将其使能

2）使能 RTC 的某一功能并使能相应中断

■ STOP 唤醒配置

1）将 EXTI 线 17 配置为中断模式使能或事件模式使能，然后选择上升沿有效

2）使能 RTC 的某一功能并使能相应中断

■ STANDBY 模式唤醒

1）RTC_INT 作为 STANDBY 模式不需要进行系统级设置。

2）使能 RTC 的某一功能并使能相应中断

23.5. RTC 寄存器描述

23.5.1. 寄存器列表

RTC 寄存器基地址：0x40002800

偏移	名称	复位值	描述
0x00	RTC_WP	0x00000000	写保护寄存器
0x04	RTC_IE	0x00000000	中断使能寄存器
0x08	RTC_SR	0x01000000	中断标志寄存器
0x0C	RTC_SEC	0x00000000	秒计数寄存器
0x10	RTC_MIN	0x00000000	分计数寄存器
0x14	RTC_HOUR	0x00000000	时计数寄存器
0x18	RTC_DAY	0x00000000	日计数寄存器
0x1C	RTC_WEEK	0x00000000	周计数寄存器
0x20	RTC_MONTH	0x00000000	月计数寄存器
0x24	RTC_YEAR	0x00000000	年计数寄存器
0x28	RTC_ALARM	0x00000000	闹钟寄存器
0x2C	RTC_CR	0x00000000	控制寄存器
0x30	RTC_ADJUST	0x00000000	时钟误差补偿寄存器
0x44	RTC_CLKSTAMP1	0x00000000	时间戳寄存器 1
0x48	RTC_CALSTAMP1	0x00000000	日历戳寄存器 1
0x4C	RTC_CLKSTAMP2	0x00000000	时间戳寄存器 2
0x50	RTC_CALSTAMP2	0x00000000	日历戳寄存器 2
0x54	RTC_WUTR	0x0000ffff	唤醒定时器寄存器
0x70~0xAC	RTC_BAKUP0~15	0x00000000	备份寄存器 0~15

23.5.2. 写保护寄存器(RTC_WP: 00h)

位域	名称	属性	复位值	描述
31:0	WE	RW	0	RTC 写使能命令， 当 CPU 向 RTC_WP 写入 0x CA53CA53 时，允许 CPU 向 RTC 的时间/日期寄存器写入初值，这时 WE 读为 1；当 CPU 向 RTC_WP 写入不为 0x CA53CA53 的任意值时恢复写保护,这时读 WE 为 0。

23.5.3. 中断使能寄存器(RTC_IE: 04h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留

17	WUTIE	RW	0	唤醒定时器中断使能 0：禁止唤醒定时器中断 1：使能唤醒定时器中断
16	STP2RIE	RW	0	RTC STAMP2 上升沿事件中断使能 0：禁止中断 1：使能中断
15	STP2FIE	RW	0	RTC STAMP2 下降沿事件中断使能 0：禁止中断 1：使能中断
14	STP1RIE	RW	0	RTC STAMP1 上升沿事件中断使能 0：禁止中断 1：使能中断
13	STP1FIE	RW	0	RTC STAMP1 下降沿事件中断使能 0：禁止中断 1：使能中断
12	ADJ32_IE	RW	0	32 秒中断使能。 1：中断使能打开 0：中断使能禁止
11	ALM_IE	RW	0	闹钟中断使能。 1：中断使能打开 0：中断使能禁止
10	1KHZ_IE	RW	0	1khz 中断使能。 1：中断使能打开 0：中断使能禁止
9	256HZ_IE	RW	0	256hz 中断使能。 1：中断使能打开 0：中断使能禁止
8	64HZ_IE	RW	0	64hz 中断使能。 1：中断使能打开 0：中断使能禁止
7	16HZ_IE	RW	0	16hz 中断使能。 1：中断使能打开 0：中断使能禁止
6	8HZ_IE	RW	0	8hz 中断使能。 1：中断使能打开 0：中断使能禁止
5	4HZ_IE	RW	0	4hz 中断使能。 1：中断使能打开 0：中断使能禁止
4	2HZ_IE	RW	0	2hz 中断使能。 1：中断使能打开 0：中断使能禁止

3	SEC_IE	RW	0	秒中断使能。 1：中断使能打开 0：中断使能禁止
2	MIN_IE	RW	0	分中断使能。 1：中断使能打开 0：中断使能禁止
1	HOUR_IE	RW	0	小时中断使能。 1：中断使能打开 0：中断使能禁止
0	DATE_IE	RW	0	天中断使能。 1：中断使能打开 0：中断使能禁止

23.5.4. 中断标志寄存器(RTC_SR: 08h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	WUTWF	RO	1	唤醒定时器写标志 此位在 RTC_CR 中的 WUTE 位清零后，并在 2 个 RTCCLK 周期后由硬件置 1，在 WUTE 位置 1 后并在 2 个 RTCCLK 周期后清零。当 WUTE 位清零且 WUTWF 位置 1 时，可更改唤醒定时器的值。 0：不允许更新唤醒定时器配置 1：允许更新唤醒定时器配置
23:18	RSV	-	-	保留
17	WUTF	RCW1	0	唤醒定时器标志（Wakeup timer flag） 当唤醒自动重载计数器计数到 0 时，由硬件将此标志置 1。写 1 清零 1：中断置位 0：无中断产生
16	STP2RIF	RCW1	0	RTC TAMP2 上升沿事件中断标志。写 1 清零 1：中断置位 0：无中断产生 此寄存器为 1 的情况下时间戳不再记录新的上升沿事件
15	STP2FIF	RCW1	0	RTC TAMP2 下降沿事件中断标志。写 1 清零 1：中断置位 0：无中断产生 此寄存器为 1 的情况下时间戳不再记录新的下降沿事件
14	STP1RIF	RCW1	0	RTC TAMP1 上升沿事件中断标志。写 1 清零 1：中断置位 0：无中断产生 此寄存器为 1 的情况下时间戳不再记录新的上升沿事件

13	STP1FIF	RCW1	0	RTC TAMP1 下降沿事件中断标志。写 1 清零 1：中断置位 0：无中断产生 此寄存器为 1 的情况下时间戳不再记录新的下降沿事件
12	ADJ32_IF	RCW1	0	32S 中断标志。写 1 清零 1：中断置位 0：无中断产生
11	ALM_IF	RCW1	0	闹钟中断标志。写 1 清零 1：中断置位 0：无中断产生
10	1KHZ_IF	RCW1	0	1khz 中断标志。写 1 清零 1：中断置位 0：无中断产生
9	256HZ_IF	RCW1	0	256hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
8	64HZ_IF	RCW1	0	64hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
7	16HZ_IF	RCW1	0	16hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
6	8HZ_IF	RCW1	0	8hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
5	4HZ_IF	RCW1	0	4hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
4	2HZ_IF	RCW1	0	2hz 中断标志。写 1 清零 1：中断置位 0：无中断产生
3	SEC_IF	RCW1	0	秒中断标志。写 1 清零 1：中断置位 0：无中断产生
2	MIN_IF	RCW1	0	分中断标志。写 1 清零 1：中断置位 0：无中断产生
1	HOUR_IF	RCW1	0	小时中断标志。写 1 清零 1：中断置位 0：无中断产生

0	DATE_IF	RCW1	0	天中断标志。写 1 清零 1：中断置位 0：无中断产生

23.5.5. 秒计数寄存器(RTC_SEC: 0Ch)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:0	BCDSEC	RW	X	秒时间数值，BCD 格式。

备注：系统复位或 RTC 域复位后，年月日时分秒寄存器的值不确定。

23.5.6. 分钟计数寄存器(RTC_MIN: 10h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:0	BCDMIN	RW	X	分钟时间数值，BCD 格式。

23.5.7. 小时计数寄存器(RTC_HOUR: 14h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5:0	BCDHOUR	RW	X	小时数值，BCD 格式。

23.5.8. 日计数寄存器(RTC_DAY: 18h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5:0	BCDDATE	RW	X	天数值，BCD 格式。

23.5.9. 周计数寄存器(RTC_WEEK: 1Ch)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	BCDWEEK	RW	X	周数值，BCD 格式。

23.5.10. 月计数寄存器(RTC_MONTH: 20h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	BCDMONTH	RW	X	月数值，BCD 格式。

23.5.11. 年计数寄存器(RTC_YEAR: 24h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	BCDYEAR	RW	X	年数值，BCD 格式。

23.5.12. 闹钟寄存器(RTC_ALARM: 28h)

位域	名称	属性	复位值	描述
31	ALM_WDS	RW	0	闹钟星期/天选择 0：选择闹钟星期模式 1：选择闹钟日模式
30:24	ALMWEEK/ALMDAY	RW	00	闹钟的星期数值/日数值 当为星期模式时： b24:b30 分别对应周日:周六，对应为置 ‘1’ 时，代表每周该日闹钟有效。 如， b24=1， b30=1 代表周日和周六闹钟设定有效 当为日模式时： [29:24]：闹钟的日 BCD 格式
23:22	RSV	-	-	保留
21:16	ALMHOUR	RW	00	闹钟的小时 BCD 格式。
15	RSV	-	-	保留
14:8	ALMMIN	RW	00	闹钟的分 BCD 格式。
7	RSV	-	-	保留
6:0	ALMSEC	RW	00	闹钟的秒 BCD 格式

23.5.13. 控制寄存器(RTC_CR: 2Ch)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留

26:24	WUCKSEL	RW	0	唤醒时钟选择 000：选择 RTCCLK/16(RTCCLK 16 分频) 001：选择 RTCCLK/8(RTCCLK 8 分频) 010：选择 RTCCLK/4(RTCCLK 4 分频) 011：选择 RTCCLK/2(RTCCLK 2 分频) 100：选择 1Hz 101：选择 0.5Hz 其它：选择 2Hz
23	WUTE	RW	0	唤醒定时器使能 0：禁止唤醒定时器 1：使能唤醒定时器
22	TAMPFLTCLK	RW	0	侵入信号滤波时钟选择 0: RTCCLK 1: 512 个 RTCCLK（64Hz）
21	TS2EDGE	RW	0	TAMP2 记录 STAMP2 边沿选择，只影响 STAMP2 时间戳记录的边沿选择，不影响 TAMP2 的 STP2FIF 和 STP2RIF 标志产生。 0：STAMP2 选择记录 TAMP2 上升沿 1：STAMP2 选择记录 TAMP2 下降沿
20:19	TAMP2FLT	RW	00	侵入信号 2 滤波周期 00: 1 个 RTCCLK 或 64Hz 01: 2 个 RTCCLK 或 64Hz 10: 4 个 RTCCLK 或 64Hz 11: 8 个 RTCCLK 或 64Hz
18	TAMP2FLTEN	RW	0	侵入信号 2 滤波使能 0：滤波不使能 1：滤波使能
17	TAMP2FCLR	RW	0	侵入 2 下降沿清除备份寄存器 0：不清除备份寄存器 1：清除备份寄存器
16	TAMP2RCLR	RW	0	侵入 2 上升沿清除备份寄存器 0：不清除备份寄存器 1：清除备份寄存器
15	TS1EDGE	RW	0	TAMP1 记录 STAMP1 边沿选择，只影响 STAMP1 时间戳记录的边沿选择，不影响 TAMP1 的 STP1FIF 和 STP1RIF 标志产生。 0：STAMP1 选择记录 TAMP1 上升沿 1：STAMP1 选择记录 TAMP1 下降沿
14:13	TAMP1FLT	RW	00	侵入信号 1 滤波周期 00: 1 个 RTCCLK 或 64Hz 01: 2 个 RTCCLK 或 64Hz 10: 4 个 RTCCLK 或 64Hz 11: 8 个 RTCCLK 或 64Hz

12	TAMP1FLTEN	RW	0	侵入信号 1 滤波使能 0：滤波不使能 1：滤波使能
11	ALM_MSKD	RW	0	闹钟星期/天数值功能屏蔽位 0：闹钟不屏蔽星期/天数值比较 1：闹钟屏蔽星期/天数值比较
10	ALM_MSKH	RW	0	闹钟时数值屏蔽位 0：闹钟不屏蔽时数值比较 1：闹钟屏蔽时数值比较
9	ALM_MSKM	RW	0	闹钟分数字屏蔽位 0：闹钟不屏蔽分数值比较 1：闹钟屏蔽分数值比较
8	TAMP1FCLR	RW	0	侵入 1 下降沿清除备份寄存器 0：不清除备份寄存器 1：清除备份寄存器
7	TAMP1RCLR	RW	0	侵入 1 上升沿清除备份寄存器 0：不清除备份寄存器 1：清除备份寄存器
6	TAMP2EN	RW	0	侵入 2 时间戳功能使能位。 1：打开时间戳 0：关闭时间戳
5	TAMP1EN	RW	0	侵入 1 时间戳功能使能位。 1：打开时间戳 0：关闭时间戳
4	ALM_EN	RW	0	闹钟功能使能 0：不使能闹钟 1：使能闹钟

3:0	FSEL	RW	0000	频率输出选择信号： 4’ b0000：保留 4’ b0001：保留 4’ b0010：输出秒计数器进位信号，高电平宽度 1s 4’ b0011：输出分计数器进位信号，高电平宽度 1s 4’ b0100：输出小时计数器进位信号，高电平宽度 1s 4’ b0101：输出天计数器进位信号，高电平宽度 1s 4’ b0110：输出闹钟匹配信号 4’ b0111：输出 32 秒方波信号 4’ b1000：输出 WUT 唤醒信号 4’ b1001：反向输出秒计数器进位信号 4’ b1010：反向输出分计数器进位信号 4’ b1011：反向输出小时计数器进位信号 4’ b1100：反向输出天计数器进位信号 4’ b1101：反向输出闹钟匹配信号 4’ b1110：反向输出 WUT 唤醒信号 4’ b1111：输出 RTC 内部秒时标方波

23.5.14. 时钟误差补偿寄存器(RTC_ ADJUST: 30h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	ADJSIGN	RW	X	补偿调整方向 0：增加 1：减少
8:0	ADJVALUE	RW	XX	补偿调整数值

23.5.15. 时间戳寄存器 1 (RTC_CLKSTAMP1: 44h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21:16	HRSTP1	RO	0	检测到 TAMP1 后存储 BCD 小时寄存器的值。
15	RSV	-	-	保留
14:8	MINSTP1	RO	0	检测到 TAMP1 后存储 BCD 分寄存器的值。
7	RSV	-	-	保留
6:0	SECSTP1	RO	0	检测到 TAMP1 后存储 BCD 秒寄存器的值

23.5.16. 日历戳寄存器 1 (RTC_CALSTAMP1: 48h)

位域	名称	属性	复位值	描述
31:24	YEARSTP1	RO	0	检测到 TAMP1 后存储 BCD 年寄存器的值。
23:21	RSV	-	-	保留
20:16	MONSTP1	RO	0	检测到 TAMP1 后存储 BCD 月寄存器的值。
15:11	RSV	-	-	保留
10:8	WKSTP1	RO	0	检测到 TAMP1 后存储 BCD 周寄存器的值。
7:6	RSV	-	-	保留
5:0	DAYSTP1	RO	0	检测到 TAMP1 后存储 BCD 日寄存器的值

23.5.17. 时间戳寄存器 2 (RTC_CLKSTAMP2: 4Ch)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
21:16	HRSTP2	RO	0	检测到 TAMP2 后存储 BCD 小时寄存器的值。
15	RSV	-	-	保留
14:8	MINSTP2	RO	0	检测到 TAMP2 后存储 BCD 分寄存器的值。
7	RSV	-	-	保留
6:0	SECSTP2	RO	0	检测到 TAMP2 后存储 BCD 秒寄存器的值

23.5.18. 日历戳寄存器 2 (RTC_CALSTAMP2: 50h)

位域	名称	属性	复位值	描述
31:24	YEARSTP2	RO	0	检测到 TAMP1 后存储 BCD 年寄存器的值。
23:21	RSV	-	-	保留
20:16	MONSTP2	RO	0	检测到 TAMP2 后存储 BCD 月寄存器的值。
15:11	RSV	-	-	保留
10:8	WKSTP2	RO	0	检测到 TAMP2 后存储 BCD 周寄存器的值。
7:6	RSV	-	-	保留
5:0	DAYSTP2	RO	0	检测到 TAMP2 后存储 BCD 日寄存器的值

23.5.19. 唤醒定时器寄存器(RTC_WUTR: 54h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留

15:0	WUT	RW	0xFFFF	唤醒自动重载值位（Wakeup auto-reload） 当使能唤醒定时器时（WUTE 置 1），每 (WUT[15:0] + 1) 个 ck_wut 周期将 WUTF 标志置 1 一次。 ck_wut 周期通过 RTC_CR 寄存器的 WUCKSEL[2:0] 位进行选择

23.5.20. 备份寄存器 0~15 (RTC_BAKUP0~15: 70~ACh)

位域	名称	属性	复位值	描述
31:0	BAKUP	RW	00	备份寄存器 这些位可以被用来写入用户数据。 注意：BAKUP 寄存器不会被系统复位、BOR 复位、从待机模式唤醒所复位。 它们可以由备份域复位来复位或(如果侵入检测引脚 TAMPER 功能被开启时)由侵入引脚事件复位。

24. 外部存储器控制器（FMC）

24.1. 概述

FMC 包括以下三个存储控制器：NOR Flash/PSRAM 存储控制器、NAND 存储控制器和 SDRAM 存储控制器。FMC 用来访问各种片外存储器，通过配置寄存器，可以把 AMBA 协议转换为专用的片外存储器通信协议，并且满足外部存储器器件的访问时间要求。

所有外部存储器共享地址、数据和控制信号，但有各自的片选信号。FMC 一次只能访问一个外部器件。

FMC 模块划分为许多个子 Bank，每个 Bank 支持特定的存储器类型，用户可以通过对 Bank 的寄存器配置来控制外部存储器。部分存储器的 BANK 地址可以重新映射。

三个存储控制器的技术细节，请参考对应章节。

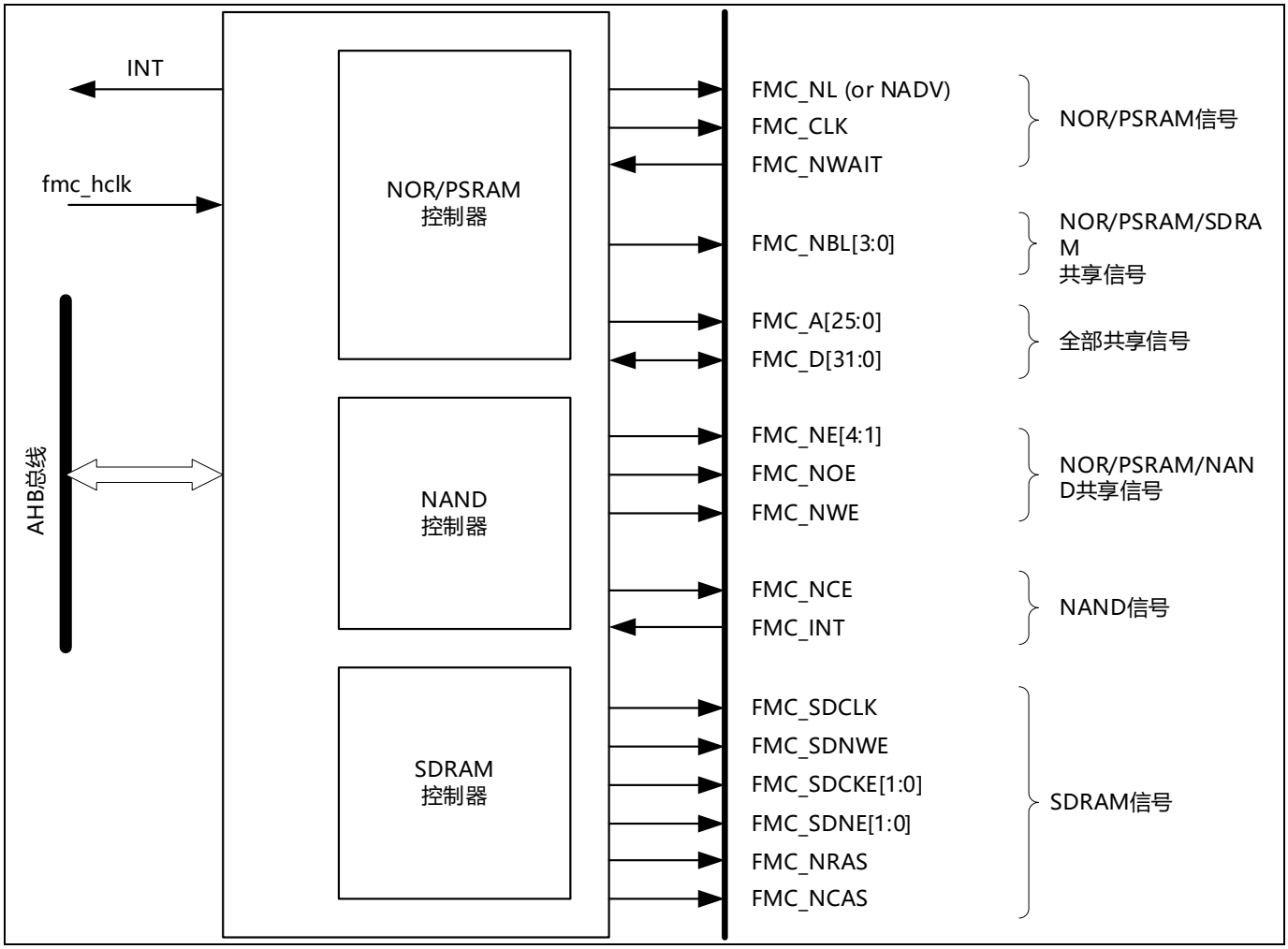
24.2. 主要特性

- 片外存储器类型，包括：
  - 静态随机访问存储器（SRAM）
  - NOR Flash
  - PSRAM（4 个存储区域）
  - 8 位 NAND Flash
  - SDRAM
- AMBA 协议与各种片外存储器协议转换
- 时序参数可编程可以满足用户特定需求
- 具有 8 位、16 位或 32 位宽的数据总线
- 每个存储 BANK 有独立的片选控制
- 每个存储 BANK 可独立配置，支持独立的读写时序
- 写使能和字节通道选择输出，可配合 PSRAM、SRAM 和 SDRAM 器件使用
- NOR Flash 和 PSRAM 支持地址总线和数据总线的复用。
- 外部异步等待控制
- 当 AMBA 总线宽度与外部存储器数据宽度不同时，会自动分割操作。

24.3. 功能描述

24.3.1. 结构框图

图 24-1FMC 结构框图



注：NAND 的 CLE 连接到 FMC_A[16]，NAND 的 ALE 连接到 FMC_A[17]，NAND 的 REN 连接到 FMC_NOE，NAND 的 CEN[0]连接到 FMC_NE[2]，NAND 的 RBN 连接到 FMC_INT。

24.3.2. AHB 接口

CPU 和其它 AHB 总线主设备可通过该 AHB 从设备接口访问外部存储器。

AHB 事务会转换为外部器件协议。尤其是当所选外部存储器的宽度为 16 位或 8 位时，AHB 中的 32 位宽事务将被划分成多个连续的 16 或 8 位访问。连续访问之间，FMC 片选不会翻转。

在数据传输的过程中，AHB 数据宽度和存储器数据宽度可能不相同。为了保证数据传输的一致性，FMC 读写访问需要遵从一下规范：

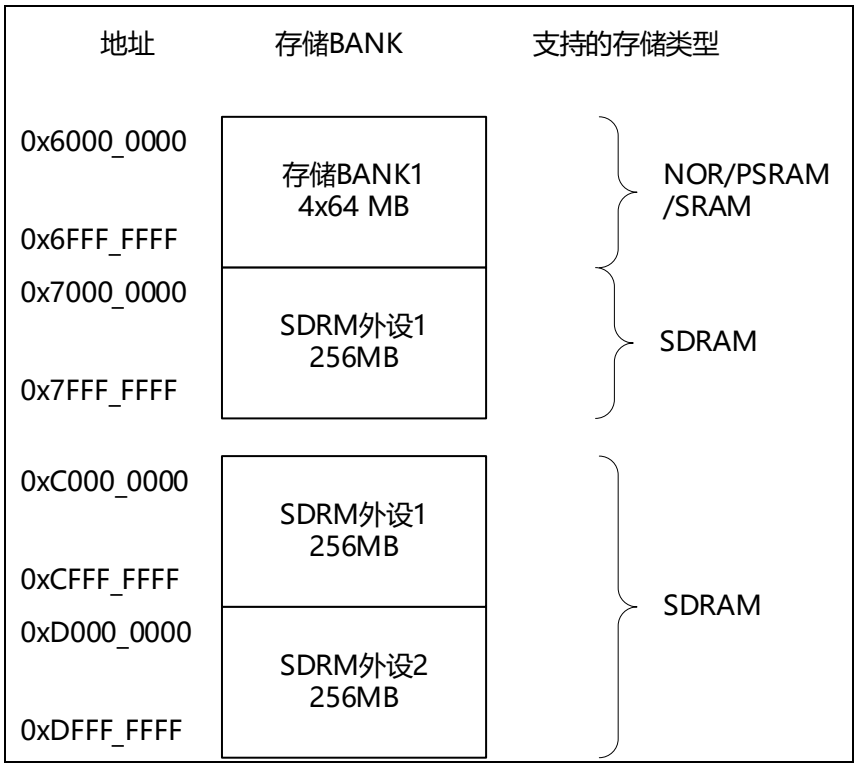
- AHB 访问宽度等于存储器宽度，则没有问题；
- AHB 访问宽度大于存储器宽度，则自动将 AHB 访问分割成几个连续的存储器数据宽度的传输；
- AHB 访问宽度小于存储器宽度。如果外部存储设备具有字节选择功能，如 SRAM、ROM、PSRAM、SDRAM，则可通过它的字节通道 NBL[3:0]来访问对应的字节。否则禁止写操作，只允许读操作。

24.3.3. 外部器件地址映射

FMC 外部存储器被划分为固定大小的存储 Bank，每个存储 Bank 的大小为 256 MB（请参见下图）：

- 存储 BANK1 可连接多达 4 个 NOR Flash 或 PSRAM 设备。此存储区域被划分为如下 4 个 NOR/PSRAM 子区域，带 4 个专用片选信号：
  - 存储 BANK1 NOR/PSRAM 1
  - 存储 BANK1 NOR/PSRAM 2
  - 存储 BANK1 NOR/PSRAM 3
  - 存储 BANK1 NOR/PSRAM 4
- 存储 BANK 2 用于 SDRAM 器件，具体是 SDRAM 存储 BANK1 还是 SDRAM 存储 BANK2 取决于 FMC_SWP 位配置。
- 存储 BANK3 和 4 用于连接 SDRAM 器件（每个存储 BANK 对应 1 个 SDRAM 外设）。
- NAND 不支持地址映射访问，读写访问通过 NAND 控制器的数据寄存器发起
- 对于每个存储区域，所要使用的存储器类型可由用户应用程序通过配置寄存器配置。

图 24-2 存储映射（默认映射）



FMC 存储区域映射可通过 系统寄存器中的 FMC_SWP[1:0] 位修改。 下表显示了用于交换 NOR/PSRAM 存储区域与 SDRAM 存储区域或者重映射 SDRAM 存储区域 2 的配置，从而允许在两个不同的地址映射中访问 SDRAM 存储区域。

表 24-1 FMC 存储映射表

偏移	名称	复位值	描述
0x60000000-0x6fffffff	NOR/PSRAM 区域	SDRAM 外设 1	NOR/PSRAM 区域
0x70000000-0x7fffffff	SDRAM 外设 1	SDRAM 外设 2	SDRAM 外设 2

0xc0000000-0xcfffffff	SDRAM 外设 1	NOR/PSRAM 区域	SDRAM 外设 1
0xd0000000-0xdfffffff	SDRAM 外设 2	SDRAM 外设 2	SDRAM 外设 2

24.3.3.1. NOR/PSRAM 地址映射

HADDR[27:26] 位用于从下表中所示的四个存储区域之中选择其中一个存储区域。

表 24-2 NOR/PSRAM 地址映射

HADDR[27:26]	选择的存储区域
00	NOR/PSRAM 1
01	NOR/PSRAM 2
10	NOR/PSRAM 3
11	NOR/PSRAM 4

HADDR[25:0] 位包含外部存储器地址。由于 ADDR 为字节地址，而存储器按字寻址，所以根据存储器数据宽度不同，实际向存储器发送的地址也将有所不同，如下表所示。

表 24-3 NOR/PSRAM 存储地址表

存储数据宽度	Core 发出的数据地址	FMC 对应有效地址	最大存储容量
8 位	HADDR[25:0]	FMC_A[25:0]	256Mb
16 位	HADDR[25:1] >> 1	FMC_A[24:0]	
32 位	HADDR[25:2] >> 2	FMC_A[23:0]	

24.3.3.2. SDRAM 地址映射

两个 SDRAM 存储外设在不用的 FMC_SWP 模式下有不同选择，详见 FMC 存储映射表。下表为在同一个 SDRAM 外设内的 13 位行地址和 11 位列地址的配置映射：

表 24-4 SDRM 存储映射表

存储数据宽度	内部 bank	行地址	列地址	最大存储容量
8 位	HADDR[25:24]	HADDR[23:11]	HADDR[10:0]	64 Mbytes: 4 x 8K x 2K
16 位	HADDR[26:25]	HADDR[24:12]	HADDR[11:1]	128 Mbytes: 4 x 8K x 2K x 2
32 位	HADDR[27:26]	HADDR[25:13]	HADDR[12:2]	256 Mbytes: 4 x 8K x 2K x 4

上表给出了最大配置下的地址映射，对于其它情况，HADDR[27:0] 位将转换为外部 SDRAM 地址，具体取决于 SDRAM 控制器配置：

- 数据大小：8、16 或 32 位
- 行大小：11、12 或 13 位

- 列大小：8、9、10 或 11 位
- 内部 BANK 数量：2 或 4 Bank

25. NOR Flash/PSRAM 控制器（FMC_NORSRAM）

25.1. 概述

NOR Flash/PSRAM 控制器，用来访问各种片外存储器。通过配置寄存器，可以把 AMBA 协议转换为专用的片外存储器通信协议，包括 SRAM，PSRAM，ROM、NorFlash 和 8080-LCD。用户还可以调整配置寄存器中的时间参数来提高通信效率。

25.2. 主要特性

- 支持片外存储器类型：
  - SRAM
  - PSRAM
  - ROM
  - NOR Flash
  - 8080-LCD
- AMBA 协议与各种片外存储器协议转换
- 时序参数可编程可以满足用户特定需求
- 对于部分存储器类型支持独立的读写时序
- 支持 8 位，或 16 位总线带宽
- NOR Flash 和 PSRAM 支持地址总线和数据总线的复用
- 提供写使能和字节选择信号
- 当 AMBA 总线宽度与外部存储器数据宽度不同时，会自动分割操作
- 支持同步模式、Burst 突发模式

25.3. 功能说明

NORFlash/PSRAM 控制器控制可以支持 NOR Flash、PSRAM、SRAM、ROM、CRAM(Cellular RAM)等外部存储器。

FMC 对每个存储块输出一个唯一的片选信号 NE[4:1]，所有其它的(地址、数据和控制)信号则是共享的。在同步方式中， FMC 向选中的外部设备产生时钟(CLK)，该时钟的频率是 HCLK 时钟的整除因子。每个存储块的大小固定为 64M 字节。每个存储块都有专门的寄存器控制。

可编程的存储器参数包括访问时序见下表:

表 25-1 可编程的 NOR/PSRAM 访问参数

参数	功能	访问方式	单位	最小	最大
地址建立时间	地址建立阶段的时间	异步	AHB 时钟周期(HCLK)	1	16
地址保持时间	地址保持阶段的时间	异步，复用 I/O	AHB 时钟周期(HCLK)	2	16
数据建立时间	数据建立阶段的时间	异步	AHB 时钟周期(HCLK)	2	256
总线恢复时间	总线恢复阶段的时间	异步或同步读	AHB 时钟周期(HCLK)	1	16

存储器访问的时钟周期(CLK)与 AHB 时钟周期的比例	时钟分频因子	同步	AHB 时钟周期(HCLK)	1	16
突发模式下产生第一个数据所需的时钟数目	数据产生时间	同步	存储器时钟周期(CLK)	2	17

25.3.1. 外部存储器接口信号

表 25-2 NOR Flash 接口信号描述

FMC 引脚	传输方向	模式	功能描述
FMC_CLK	输出	同步	同步时钟信号
非复用 FMC_A[25:0]	输出	异步/同步	地址总线信号
复用 FMC_A[25:16]			
FMC_D[15:0]	输入/输出	异步/同步（复用）	地址/数据总线
	输入/输出	异步/同步（非复用）	数据总线
FMC_NE	输出	异步/同步	片选
FMC_NOE	输出	异步/同步	输出使能（读使能）
FMC_NWE	输出	异步/同步	写使能
FMC_NWAIT	输入	异步/同步	等待输入信号
FMC_NL（NADV）	输出	异步/同步	地址有效

表 25-3 PSRAM 非复用接口信号描述

FMC 引脚	传输方向	模式	功能描述
FMC_CLK	输出	同步	同步时钟信号
FMC_A[25:0]	输出	异步/同步	地址总线
FMC_D[15:0]	输入/输出	异步/同步	数据总线
FMC_NE	输出	异步/同步	片选
FMC_NOE	输出	异步/同步	输出使能（读使能）
FMC_NWE	输出	异步/同步	写使能
FMC_NWAIT	输入	异步/同步	等待输入信号
FMC_NL（NADV）	输出	异步/同步	锁存（地址有效）使能
FMC_NBL[1]	输出	异步/同步	高字节使能
FMC_NBL[0]	输出	异步/同步	低字节使能

25.3.2. 支持的存储器和事务

下表列出了当 NOR， PSRAM 和 SRAM 存储器数据总线为 16 位时，所支持的设备类型、 访问模式和传输的示例。

表 25-4 NOR Flash/PSRAM 支持的传输

存储器类型	访问模式	读/写	AHB 传输宽度	存储器传输宽度	注释
NorFlash	异步	R	8	16	
	异步	R	16	16	
	异步	W	16	16	
	异步	R	32	16	分 2 次 FMC 访问
	异步	W	32	16	分 2 次 FMC 访问
	同步	R	16	16	
	同步	R	32	16	
PSRAM	异步	R	8	16	
	异步	W	8	16	使用字节信号 FMC_NBL[1:0]
	异步	R	16	16	
	异步	W	16	16	
	异步	R	32	16	分 2 次 FMC 访问
	异步	W	32	16	分 2 次 FMC 访问
	同步	R	16	16	
	同步	R	32	16	
	同步	W	8	16	使用字节信号 FMC_NBL[1:0]
	同步	W	16	16	
	同步	W	32	16	分 2 次 FMC 访问
SRAM/ROM	异步	R	8	8	
	异步	R	8	16	
	异步	R	16	8	分 2 次 FMC 访问
	异步	R	16	16	
	异步	R	32	8	分 4 次 FMC 访问
	异步	R	32	16	分 2 次 FMC 访问
	异步	W	8	8	
	异步	W	8	16	使用字节信号 FMC_NBL[1:0]
	异步	W	16	8	
	异步	W	16	16	
	异步	W	32	8	
	异步	W	32	16	

25.3.3. 通用时序

在异步模式下，所有控制器输出信号在内部 AHB 总线时钟（HCLK）的上升沿改变。

在同步模式下，所有控制器输出数据在外部存储器时钟（FMC_CLK）的下降沿改变。

25.3.4. 异步事务

FMC 为 SRAM、 ROM、 PSRAM、 NOR Flash 等外部静态存储器提供可编程的时序参数以及多种时序模型以满足不同的需求。

异步静态存储器(NOR 闪存和 PSRAM)

- 所有信号由内部时钟 HCLK 保持同步，但该时钟不会输出到存储器；
- FMC 始终在片选信号 NE 失效前对数据线采样，这样能够保证符合存储器的数据保持时序(片选失效至数据失效的间隔，通常最小为 0ns)；
- 当设置了扩展模式，可以在读和写时混合使用模式 A、 B、 C 和 D(例如，允许以模式 A 进行读，而以模式 B 进行写)。

表 25-5 NOR/PSRAM 控制时序参数

参数	功能	访问模式	单位	最大值	最小值
CLKDIV	同步时钟分频比	同步	HCLK	2	16
DLAT	数据延迟	异步	FMC_CLK	2	17
BUSLAT	总线延迟	异步/同步读	HCLK	1	16
DSET	数据建立时间	异步	HCLK	2	256
AHLD	地址保持时间	异步（复用）	HCLK	2	16
ASET	地址建立时间	异步	HCLK	1	16

FMC 模块 NOR Flash/PSRAM 控制器可以提供多种时序模型。用户可以通过修改 FMC 时序模型表。NOR/PSRAM 控制时序参数中列出的参数来使之适合不同类型外部存储器的时序以及满足用户的要求。当将寄存器 FMC_SNCTL 位 EXMODEN 置 1 使能扩展模式后，可以通过寄存器 FMC_SNTCFG 和 FMC_SNWTCFG 将读写配置成独立的时序。

1）模式 A 与模式 1 的不同之处：模式 A NOE 的切换、与独立的读取和写入时序

2）模式 2 与模式 1 的不同之处：模式 2 有 NADV 信号，没有模式 1 的 NBL[1:0]信号

3）模式 B 与模式 1 的不同之处：模式 B 有 NADV 信号，没有模式 1 的 NBL[1:0]信号，NWE 的切换、与独立的读取和写入时序

4）模式 C 与模式 1 的不同之处：模式 C 有 NADV 信号，没有模式 1 的 NBL[1:0]信号，NOE 的切换、与独立的读取和写入时序

5）模式 D 与模式 1 的不同之处：模式 C 有 NADV 信号，没有模式 1 的 NBL[1:0]信号，NADV 变化后 NOE 的切换、与独立的读取和写入时序

6）复用模式与模式 D 的不同之处在于数据总线上驱动低地址字节

表 25-6 FMC 时序模型

时序模型	拓展模式	模式描述	写时序参数	读时序参数

异步	模式 1	0	SRAM/PSRAM/CRAM	DSET ASET	DSET ASET
	模式 2	0	Nor Flash	DSET ASET	DSET ASET
	模式 A	1	SRAM/PSRAM/CRAM 在数据阶段时 FMC_NOE 翻转	WDSET WASET	DSET ASET
	模式 B	1	Nor Flash	WDSET WASET	DSET ASET
	模式 C	1	Nor Flash 在数据阶段时 FMC_NOE 翻转	WDSET WASET	DSET ASET
	模式 D	1	有地址保持功能	WDSET WAHLD WASET	DSET AHLD ASET
	模式 AM	0	Nor Flash 数据/地址复用	DSET AHLD ASET BUSLAT	DSET AHLD ASET BUSLAT
同步	模式 E	0	NOR/PSRAM/CRAM 同步读， /PSRAM/CRAM 同步写	DLAT CLKDIV	DLAT CLKDIV
	模式 SM	0	Nor Flash 数据/地址复用	DLAT CLKDIV	DLAT CLKDIV

25.3.4.1. 模式 1-SRAM/CRAM

图 25-3 模式 1 读访问

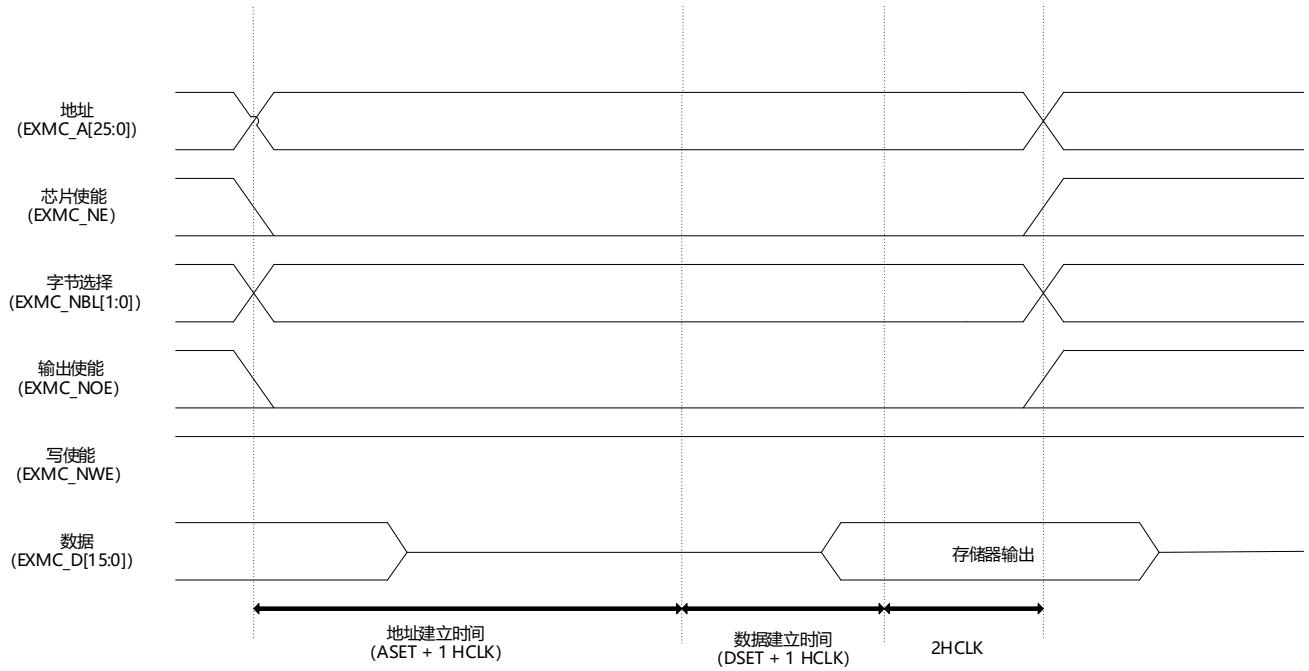


图 25-4 模式 1 写访问

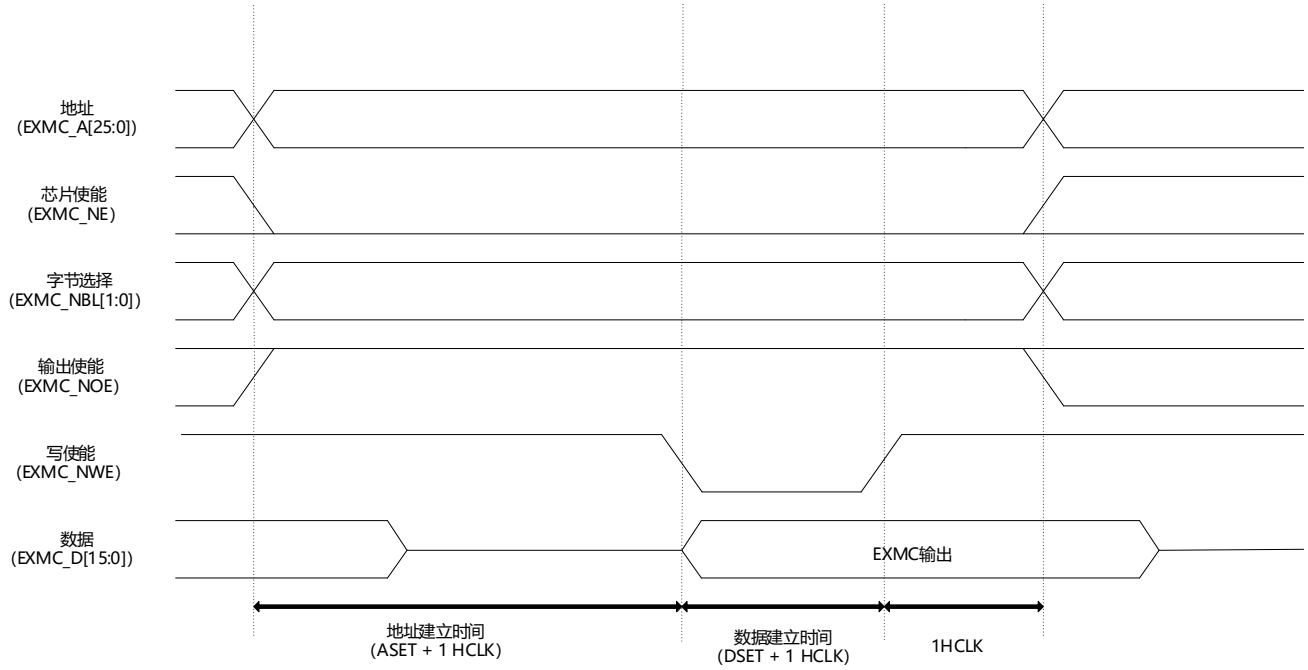


表 25-7 模式 1 相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31:20	RSV	0x000
19	SYNCWR	0x0
18:16	CPS	0x0
15	ASYNCWAIT	取决于存储器
14	EXMODEN	0x0
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	无影响
5:4	NRW	取决于存储器
3:2	NRTP	取决于存储器，除了 0x2（Nor Flash）
1	NRMUX	0x0
0	NRBKEN	0x1

FMC_SNTCFG		
31:30	RSV	0x0
29:28	ASYNCMOD	无影响
27:24	DLAT	无影响
23:20	CKDIV	无影响
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（写操作为 DSET+1HCLK 时钟周期，读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	无影响
3:0	ASET	取决于存储器与用户

25.3.4.2. 模式 A - SRAM/PSRAM（CRAM） OE 翻转

图 25-5 模式 A 写访问

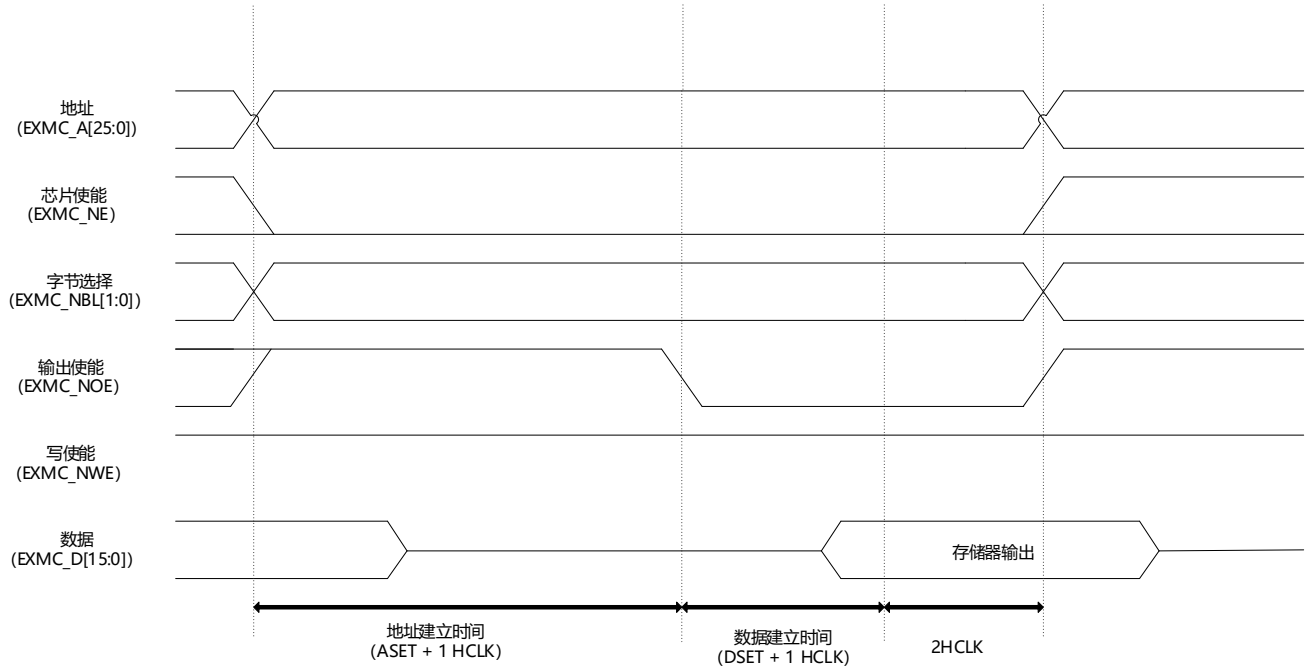
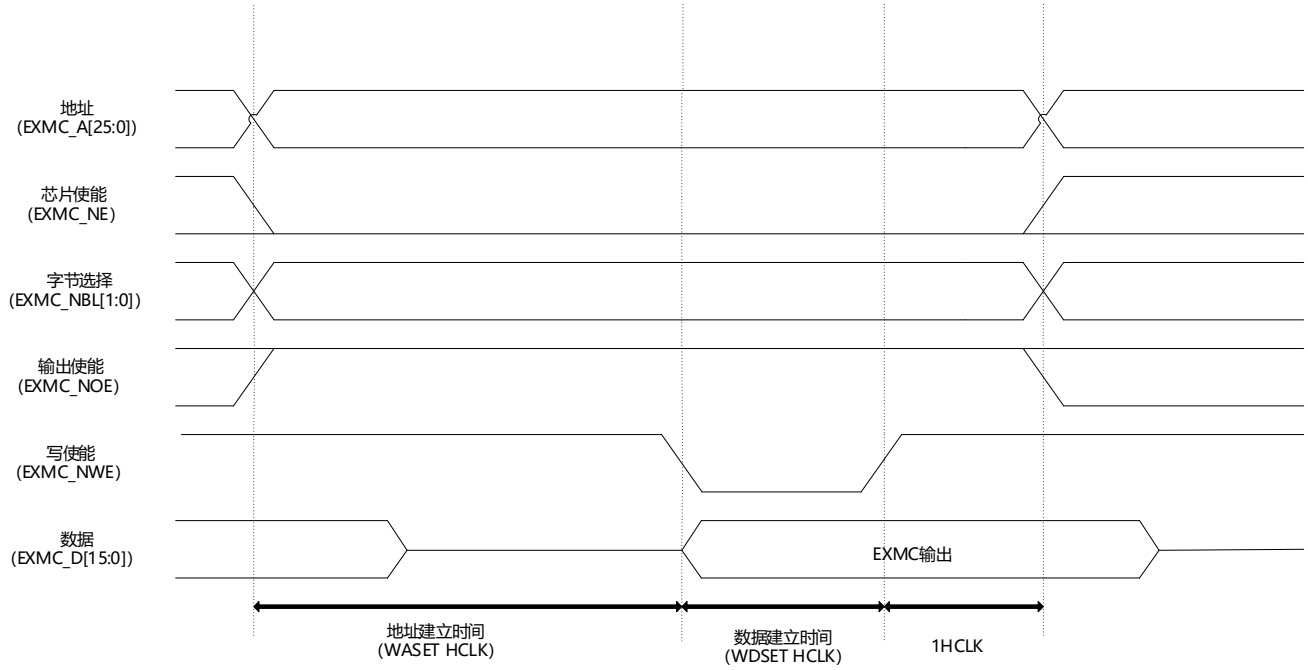


图 25-6 模式 A 读访问



模式 A 和模式 1 写时序的区别在于，当读和写访问具有相同的时序配置时， 模式 A 的写时序是独立于读时序的。

表 25-8 模式 A 相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31-20	RSV	0x000
19	SYNCWR	0x0
18:16	CPS	0x0
15	ASYNCWTEN	取决于存储器
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	无影响
5:4	NRW	取决于存储器
3:2	NRTP	取决于存储器，除了 0x2（Nor Flash）

1	NRMUX	0x0
0	NRBKEN	0x1
FMC_SNTCFG（读）		
31:30	RSV	0x0
29:28	ASYNCMOD	0x0
27:24	DLAT	无影响
23:20	CKDIV	无影响
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	无影响
3:0	ASET	取决于存储器与用户
FMC_SNWTCFG（写）		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 B：0x1
27:20	RSV	0xFF
19:16	WBUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	WDSET	取决于存储器与用户（写操作为 DSET+1HCLK 时钟周期）
7:4	WAHLD	0x0
3:0	WASET	取决于存储器与用户

25.3.4.3. 模式 2/B-Nor Flash

图 25-7 模式 2 读访问

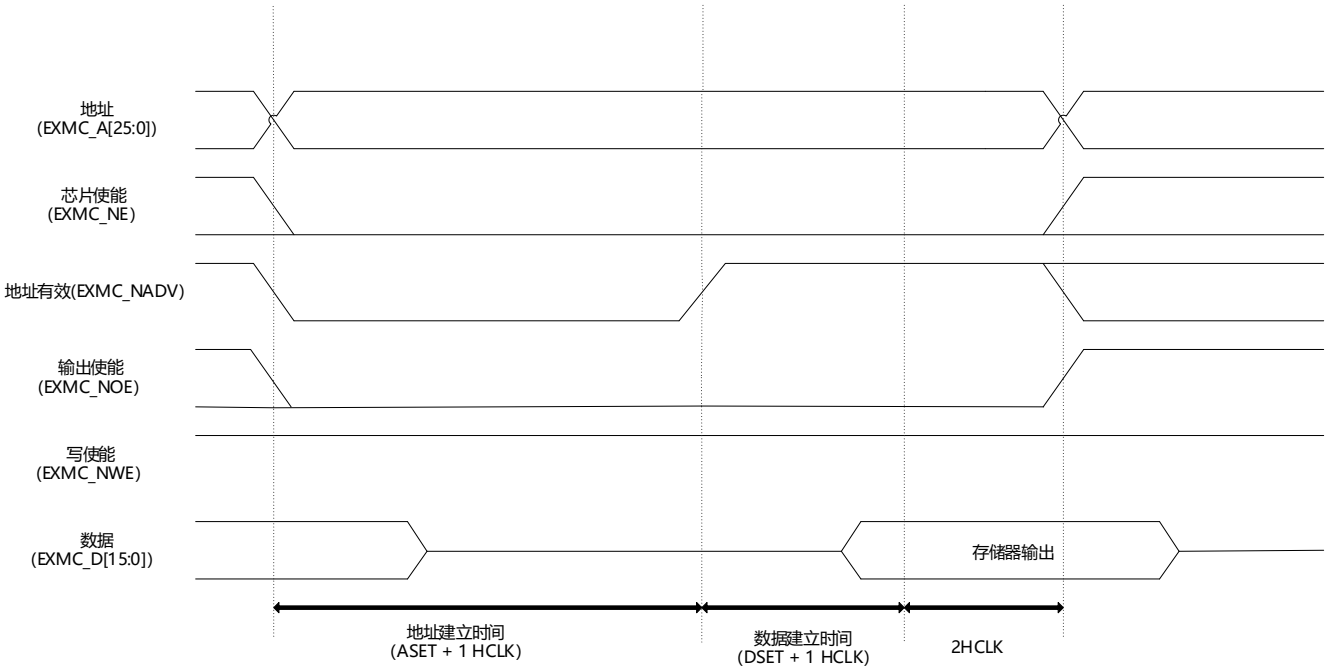


图 25-8 模式 2 写访问

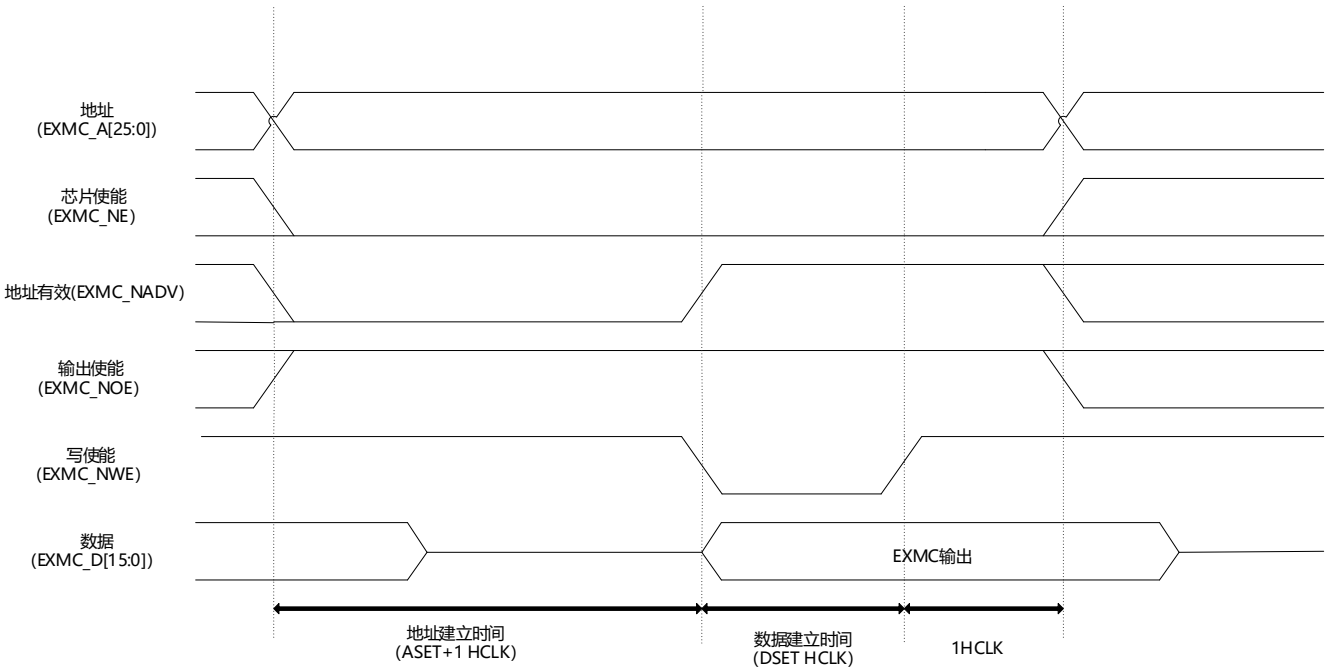
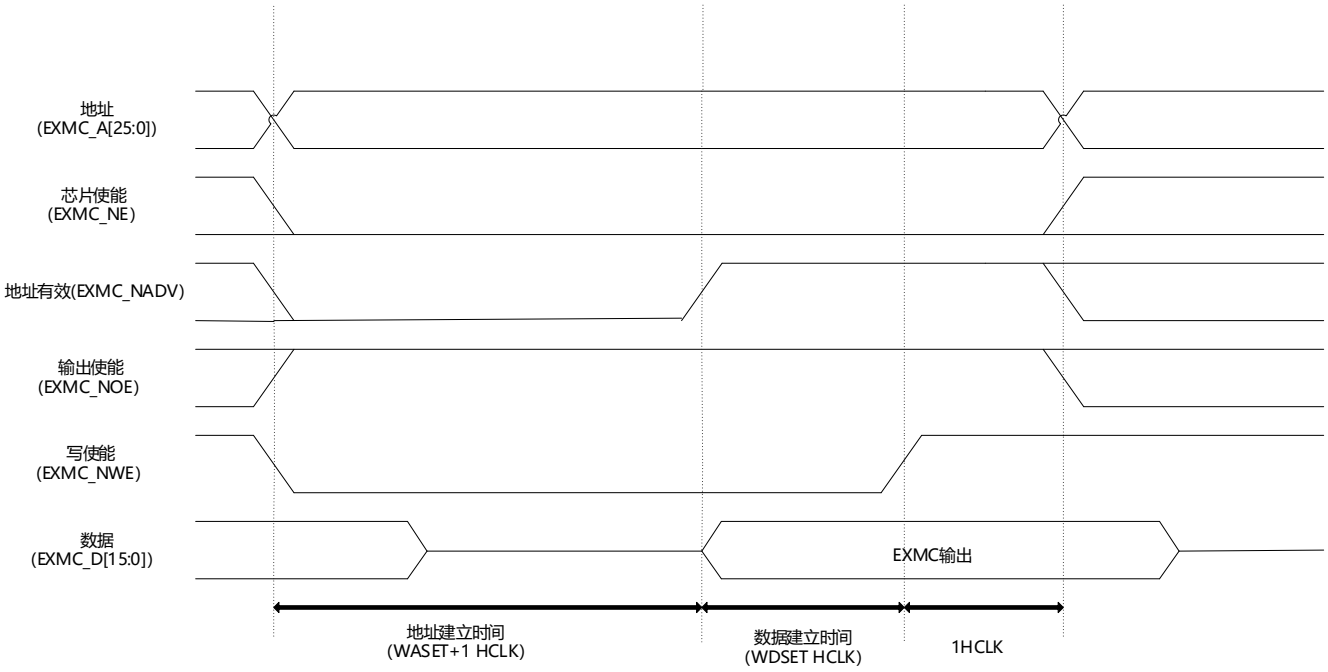


图 25-9 模式 B 写访问



模式 2/B 相关寄存器配置：

表 25-9 模式 2/B 相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL（模式 2，模式 B)		
31:20	RSV	0x000
19	SYNCWR	0x0
18:16	CPS	0x0

15	ASYNCWTEN	取决于存储器
14	EXMODEN	模式 2：0x0；模式 B：0x1
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	0x1
5:4	NRW	取决于存储器
3:2	NRTP	Nor Flash：0x2
1	NRMUX	0x0
0	NRBKEN	0x1
FMC_SNTCFG（模式 2 读/写操作；模式 B 读操作）		
31:30	RSV	0x0
29:28	ASYNCMOD	模式 B：0x1
27:24	DLAT	无影响
23:20	CKDIV	无影响
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	0x0
3:0	ASET	取决于存储器与用户
FMC_SNWTCFG（模式 B 写操作）		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 B：0x1
27:20	RSV	0xFF
19:16	WBUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	WDSET	取决于存储器与用户（写操作为 DSET+1HCLK 时钟周期）
7:4	WAHLD	0x0
3:0	WASET	取决于存储器与用户

25.3.4.4. 模式 C-NOR Flash OE 翻转

图 25-10 模式 C 读访问

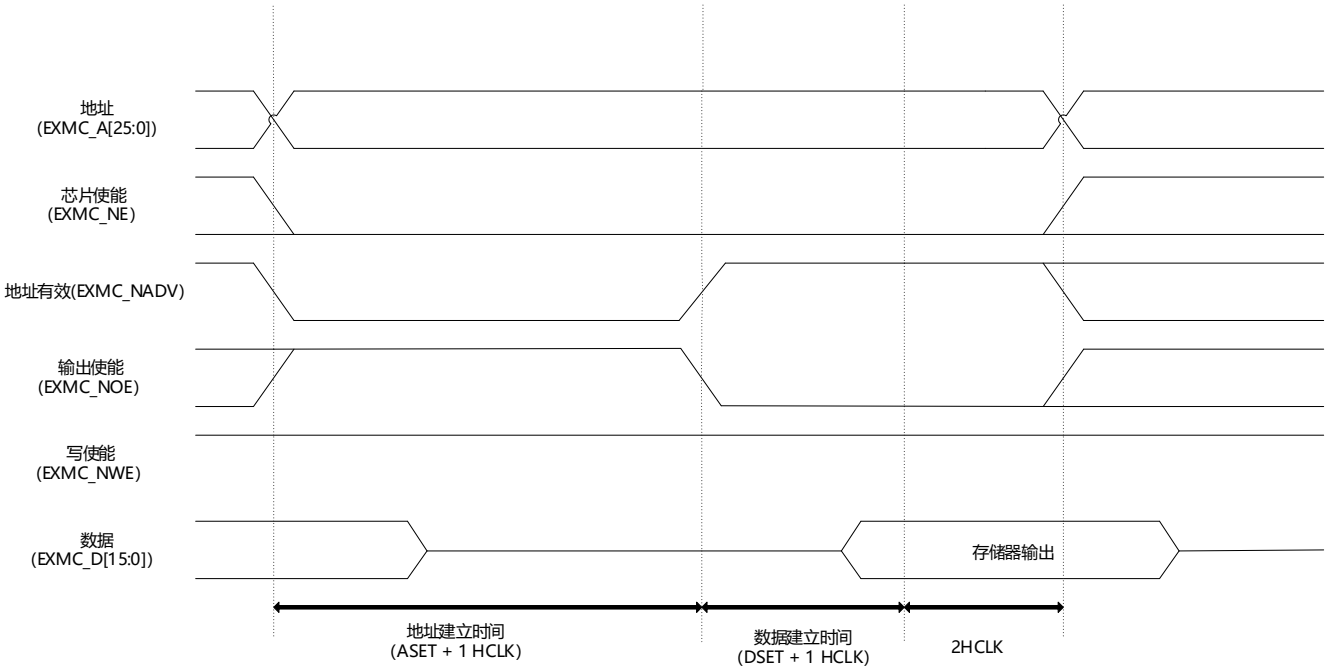
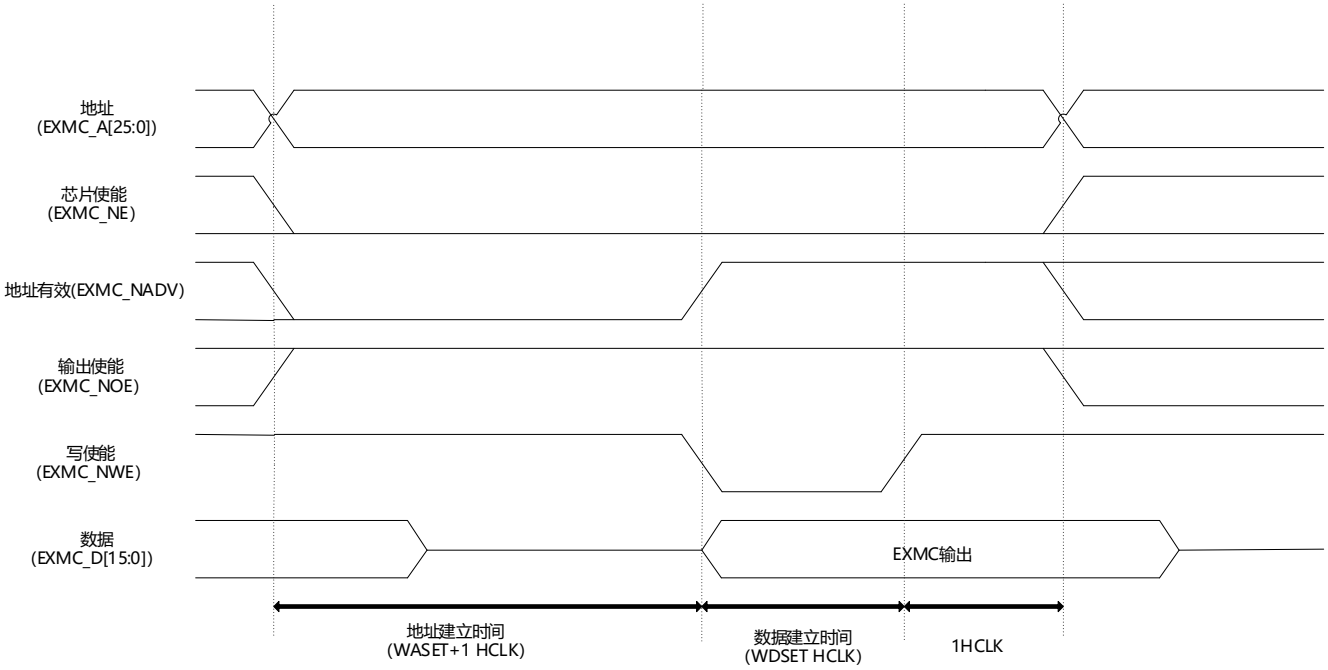


图 25-11 模式 C 写访问



模式 C 和模式 1 写时序的区别在于，当读和写访问具有相同的时序配置时， 模式 C 的写时序是独立于读时序的。

表 25-10 模式 C 相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		

31:20	RSV	0x000
19	SYNCWR	0x0
18:16	CPS	0x0
15	ASYNCWTEN	取决于存储器
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	0x1
5:4	NRW	取决于存储器
3:2	NRTP	Nor Flash：0x2
1	NRMUX	0x0
0	NRBKEN	0x1
FMC_SNTCFG		
31:30	RSV	0x0
29:28	ASYNCMOD	模式 C：0x2
27:24	DLAT	0x0
23:20	CKDIV	0x0
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	0x0
3:0	ASET	取决于存储器与用户
FMC_SNWTCFG		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 C：0x2
27:20	RSV	0xFF
19:16	WBUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	WDSET	取决于存储器与用户（写操作为 DSET+1HCLK 时钟周期）
7:4	WAHLD	0x0
3:0	WASET	取决于存储器与用户

25.3.4.5. 模式 D -带地址扩展的异步操作

图 25-12 模式 D 读访问

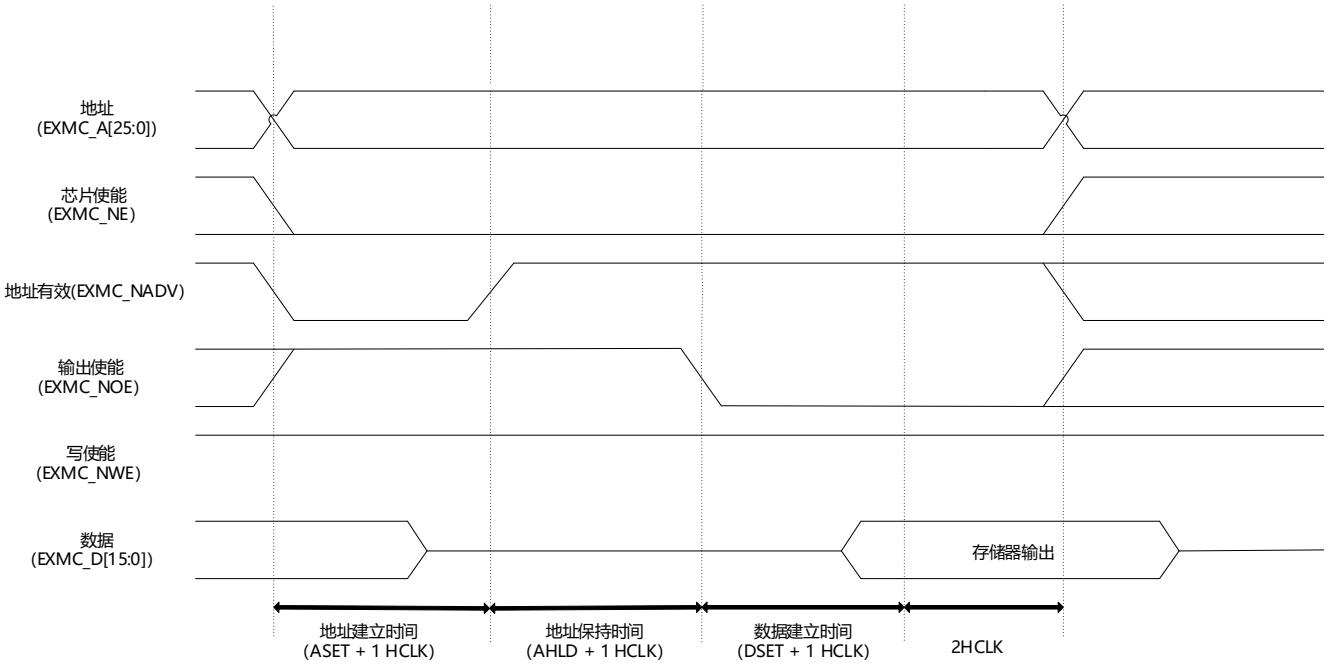


图 25-13 模式 D 写访问

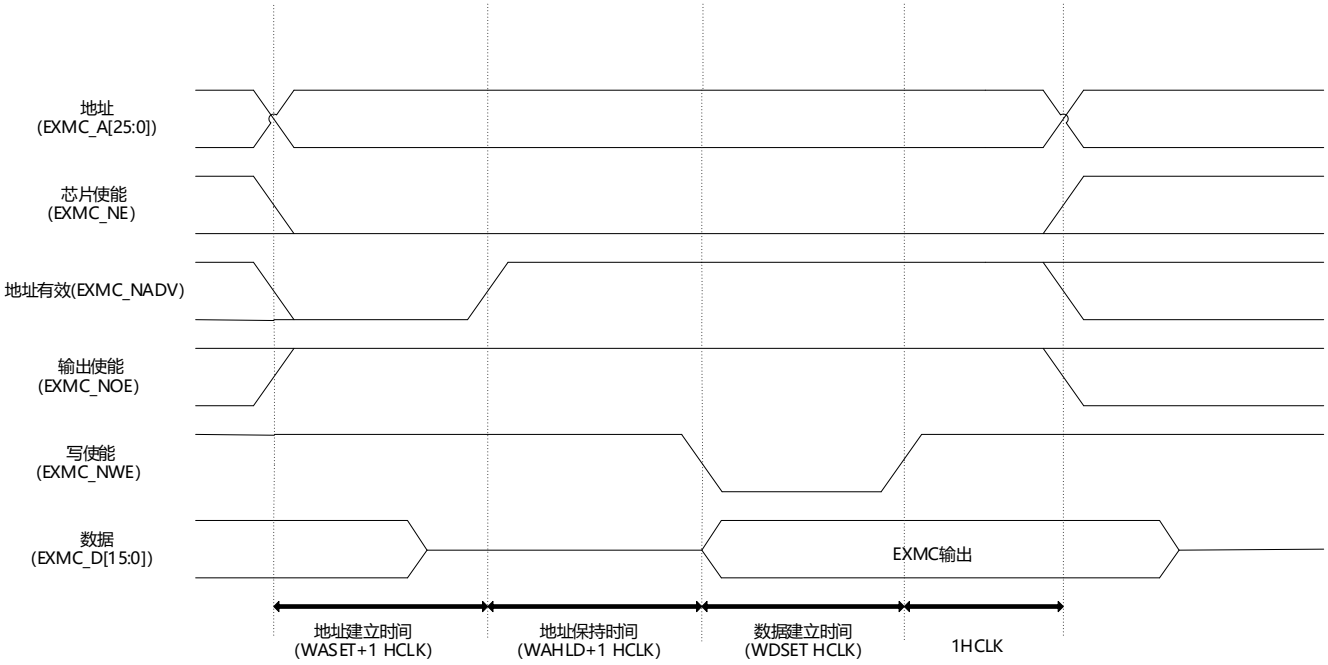


表 25-11 模式 D 相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31:20	RSV	0x000

19	SYNCWR	0x0
18:16	CPS	0x0
15	ASYNCWTEN	取决于存储器
14	EXMODEN	0x1
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	取决于存储器
5:4	NRW	取决于存储器
3:2	NRTP	取决于存储器
1	NRMUX	0x0
0	NRBKEN	0x1
FMC_SNTCFG		
31:30	RSV	0x0
29:28	ASYNCMOD	模式 D：0x3
27:24	DLAT	无关
23:20	CKDIV	无影响
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	取决于存储器与用户
3:0	ASET	取决于存储器与用户
FMC_SNWTCFG		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 D：0x3
27:20	RSV	0xFF
19:16	WBUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	WDSET	取决于存储器与用户（写操作为 DSET+1HCLK 时钟周期）
7:4	WAHLD	取决于存储器与用户
3:0	WASET	取决于存储器与用户

25.3.4.6. 模式 AM - NOR Flash 地址/数据总线复用

图 25-14 复用模式读访问

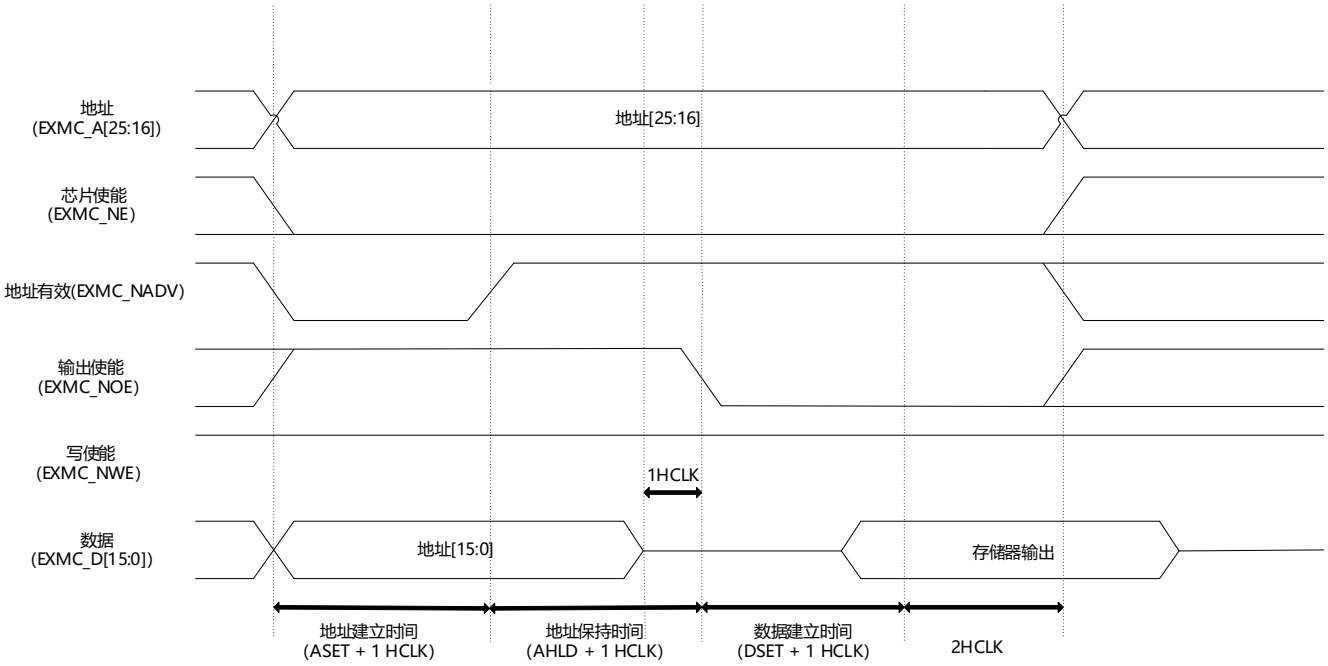


图 25-15 复用模式写访问

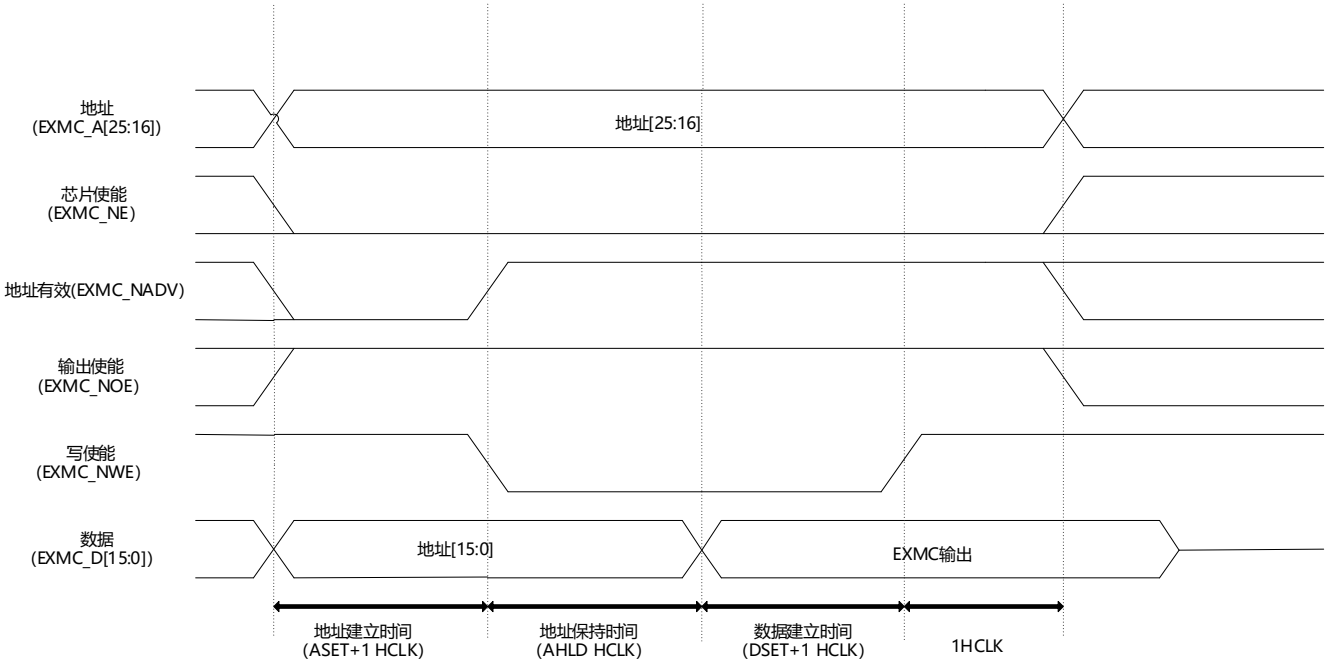


表 25-12 复用模式相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31:20	RSV	0x000

19	SYNCWR	0x0
18:16	CPS	0x0
15	ASYNCWTEN	取决于存储器
14	EXMODEN	0x0
13	NRWTEN	0x0
12	WREN	取决于用户
11	NRWTCFG	无影响
10	WRAPEN	0x0
9	NRWTPOL	仅当位 15 为 1 时有效
8	SBRSTEN	0x0
7	RSV	0x1
6	NREN	0x1
5:4	NRW	取决于存储器
3:2	NRTP	0x2：Nor Flash
1	NRMUX	0x1
0	NRBKEN	0x1
FMC_SNTCFG		
31:30	RSV	0x0
29:28	ASYNCMOD	0x0
27:24	DLAT	无影响
23:20	CKDIV	无影响
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	取决于存储器与用户（写操作为 DSET+2HCLK 时钟周期，读操作为 DSET+3HCLK 时钟周期）
7:4	AHLD	取决于存储器与用户
3:0	ASET	取决于存储器与用户

异步通信的等待时序：

等待功能由寄存器 FMC_SNCTL 位 ASYNCWAIT 控制。在访问外部存储器期间，若使能异步等待功能（ASYNCWAIT=1），数据建立时间将会根据 FMC_NWAIT 的有效信号而自动延长。

延长时间的计算如下：

1）若存储器等待信号与 FMC_NOE/ FMC_NWE 信号对齐：TDATA_SETUP≥ maxTWAIT_ASSERTION+4HCLK

2）若存储器等待信号与 FMC_NE 信号对齐：如果 maxTWAIT_ASSERTION≥TADDRES_PHASE+THOLD_PHASE ，则 TDATA_SETUP ≥ (maxTWAIT_ASSERTION -TADDRES_PHASE-THOLD_PHASE)+4HCLK 否则 TDATA_SETUP ≥4HCLK 。

图 25-16 异步等待有效时的读时序

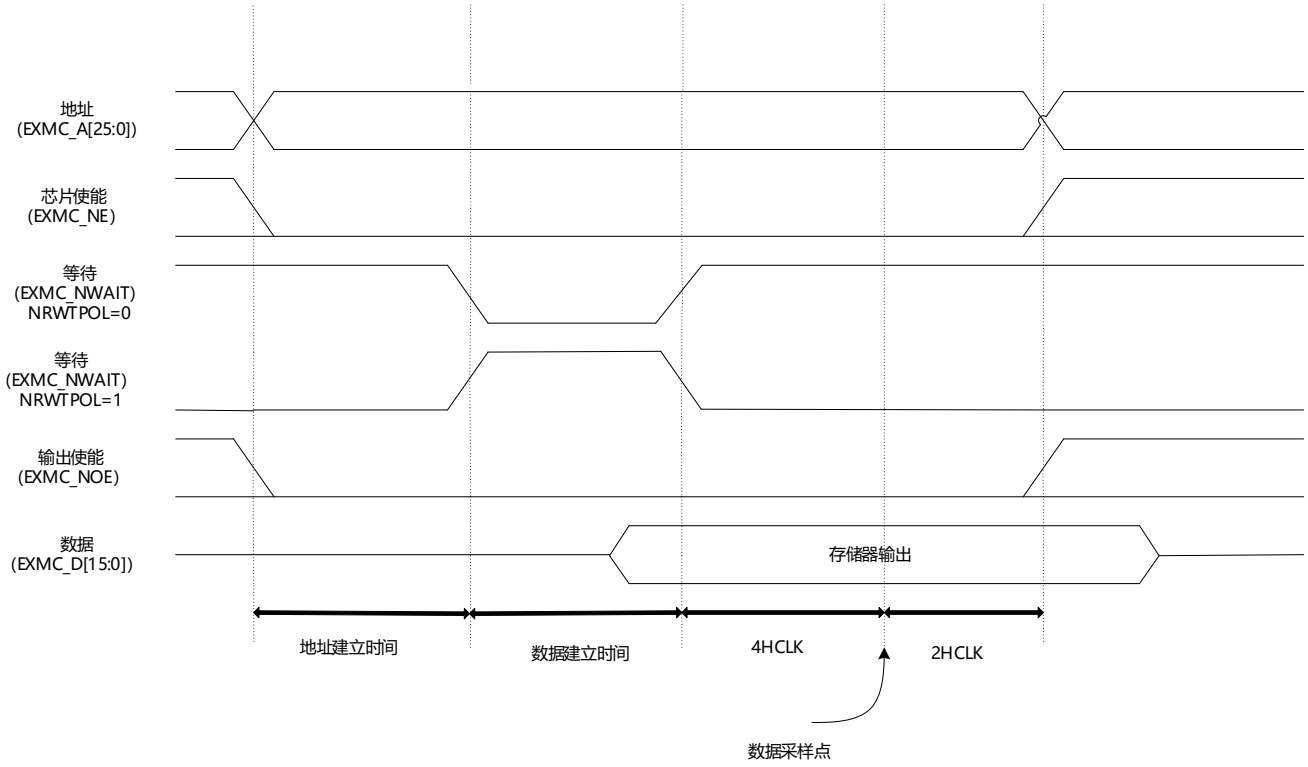
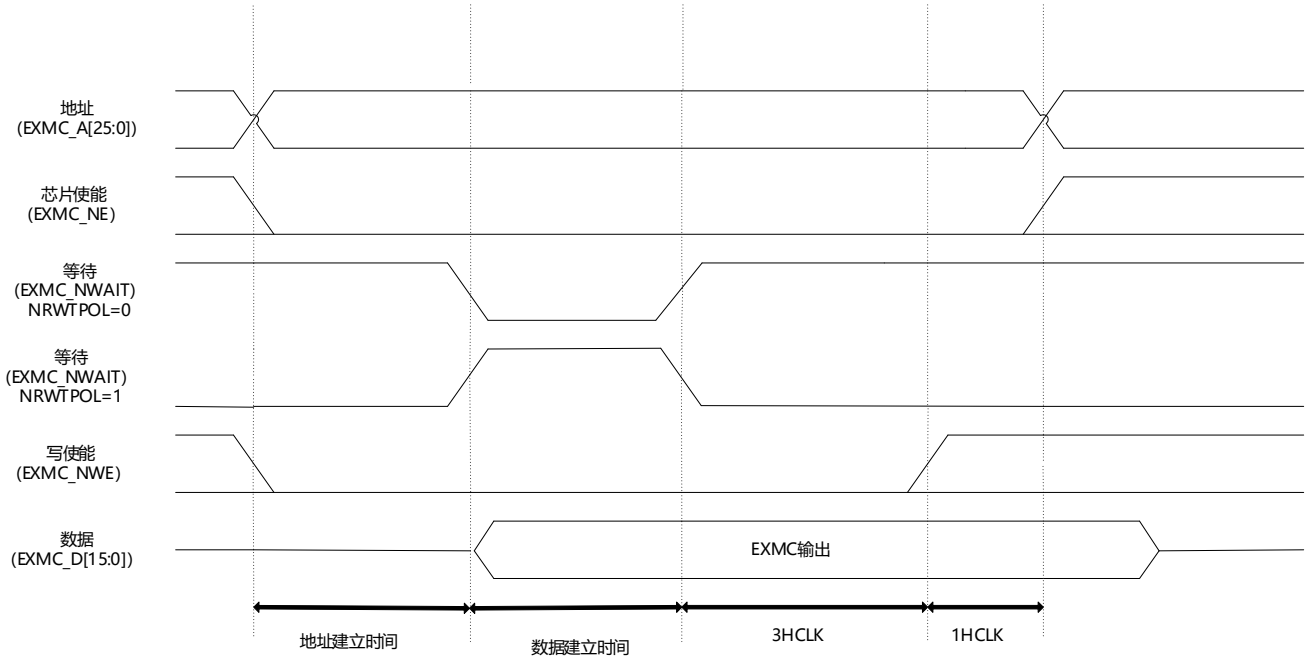


图 25-17 异步等待有效时的写时序



25.3.5. 同步事务

存储器时钟（FMC_CLK）与系统时钟（HCLK）关系如下：

FMC_CLK= HCLK / (CKDIV+1)

其中 CKDIV 是同步时钟分频比，通过配置寄存器 FMC_SNTCFG 中的 CKDIV 位来设置不同的值。

1）数据延迟与 NOR Flash 延迟：

数据延迟 DLAT 是指在采样数据之前需要等待的 FMC_CLK 周期数。它和 NOR 闪存延迟的关系如下所述。

- NOR 闪存延迟不包含 FMC_NADV，二者之间的关系为：

NOR 闪存延迟=DLAT+2

- NOR 闪存延迟包含 FMC_NADV，二者之间的关系为：

NOR 闪存延迟=DLAT+3

2）数据等待：

用户需要保证 FMC_NWAIT 信号与外部设备一致。该信号通过寄存器 FMC_SNCTL 来设置，由位 NRWTEN 来使能，位 NRWTCFG 决定 FMC_NWAIT 信号是等待状态同时有效，或者比等待状态提前一个时钟周期有效，位 NRWTPOL 设置 FMC_NWAIT 信号极性。

在 NOR Flash 的同步突发模式中， 当寄存器 FMC_SNCTL 的位 NRWTEN 置为 1， 则在数据延迟之后会检测 FMC_NWAIT 信号。 如果检测到 FMC_NWAIT 有效， 就会插入等待时钟， 直到 FMC_NWAIT 变为无效。

- FMC_NWAIT 有效极性：
  - NRWTPOL= 1， FMC_NWAIT 高电平有效；
  - NRWTPOL= 0， FMC_NWAIT 低电平有效。
- 在同步突发模式中， FMC_NWAIT 信号有两种配置：
  - NRWTCFG = 1， FMC_NWAIT 信号有效时， 当前时钟周期数据无效；
  - NRWTCFG = 0， FMC_NWAIT 信号有效时， 下一个时钟周期数据无效，这是复位后的默认配置。

在 FMC_NWAIT 信号有效的等待周期内， FMC 会持续的给存储器发送时钟信号，保持片选和输出使能有效，并且忽视总线上的无效数据。

3）CRAM 页边界突发传输的自动分组：

CRAM1.5 中禁止突发传输跨越页边界， FMC 遇到边界会进行传输的自动分组。为了保证正确的突发分组操作， 用户需要在寄存器 FMC_SNCTL 位 CPS 中需要设定 CRAM 的页大小。

4）模式 SM - 单次突发传输：

对于同步突发传输，如果 AHB 需要的数据为 16 位，则 FMC 会执行一次长度为 1 的成组传输；如果 AHB 需要的数据为 32 位，则 FMC 会把这次传输分成 2 次 16 位的传输，即执行一次长度为 2 的突发传输。对于其他的配置，请参考表 1-6. FMC 的 NOR Flash/PSRAM 支持的所有传输。

同步复用突发读时序 – NOR， PSRAM（CRAM）：

图 25-18 同步复用突发传输读时序

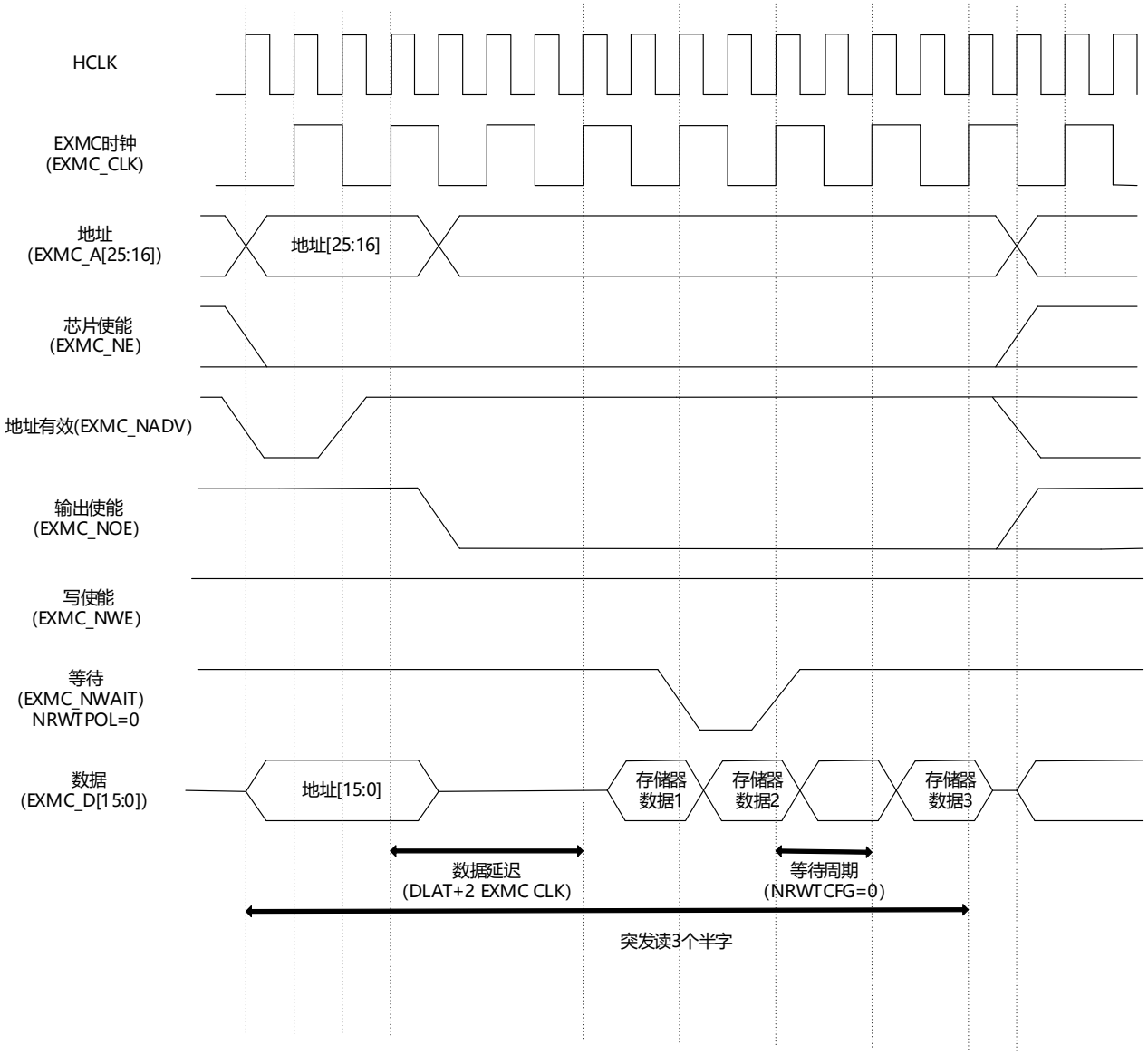


表 25-13 同步复用模式读时序相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31:20	RSV	0x000
19	SYNCWR	无影响
18:16	CPS	0x0
15	ASYNCWTEN	0x0
14	EXMODEN	0x0
13	NRWTEN	取决于存储器
12	WREN	无影响
11	NRWTCFG	取决于存储器
10	WRAPEN	0x0
9	NRWTPOL	取决于存储器

8	SBRSTEN	0x1，突发读使能
7	RSV	0x1
6	NREN	取决于存储器
5:4	NRW	0x1
3:2	NRTP	取决于存储器，0x1/0x2
1	NRMUX	0x1，取决于存储器与用户
0	NRBKEN	0x1
FMC_SNTCFG（读）		
31:30	RSV	0x0
29:28	ASYNCMOD	0x0
27:24	DLAT	数据延迟
23:20	CKDIV	上图设置：0x1，FMC_CLK=2HCLk
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	无影响
7:4	AHLD	无影响
3:0	ASET	无影响

模式 SM – 同步复用突发写时序– PSRAM（CRAM）：

图 25-19 同步复用突发传输写时序

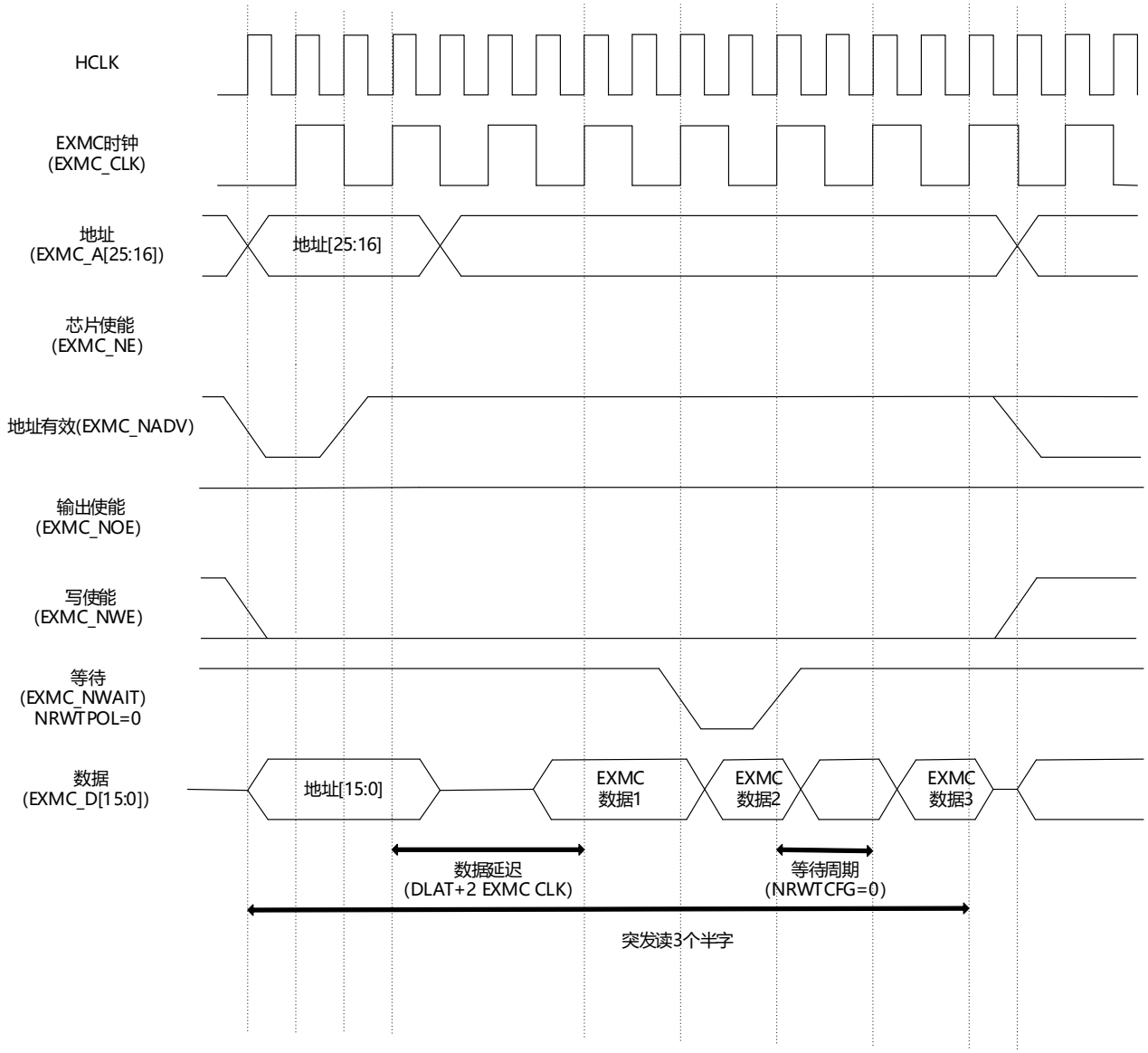


表 25-14 同步复用模式写时序相关寄存器配置

位/位域	位名	参考设定值
FMC_SNCTL		
31:20	RSV	0x000
19	SYNCWR	0x1，同步写使能
18:16	CPS	0x0
15	ASYNCWAIT	0x0
14	EXMODEN	0x0
13	NRWTEN	取决于存储器
12	WREN	0x1
11	NRWTCFG	0x0（这里必须为 0）
10	WRAPEN	0x0
9	NRWTPOL	取决于存储器

8	SBRSTEN	无影响
7	RSV	0x1
6	NREN	取决于存储器
5:4	NRW	0x1
3:2	NRTP	0x1
1	NRMUX	0x1，取决于用户
0	NRBKEN	0x1
FMC_SNTCFG（写）		
31:30	RSV	0x0
29:28	ASYNCMOD	0x0
27:24	DLAT	数据延迟
23:20	CKDIV	上图设置：0x1，FMC_CLK=2HCLk
19:16	BUSLAT	从 FMC_NE 上升沿到下降沿的时间
15:8	DSET	无影响
7:4	AHLD	无影响
3:0	ASET	无影响

25.4. 配置流程

25.4.1. Nor 闪存设置流程

1）将 Nor Flash 对应 IO 口的外设复用功能配置到 FMC 功能；

2）参考《异步事务》与《同步事务》章节，选择相应的时序模型，进行相关寄存器配置；

3）进行 Nor Flash 访问。

25.4.2. PSRAM 设置流程

1）将 PSRAM 对应 IO 口的外设复用功能配置到 FMC 功能；

2）参考《异步事务》与《同步事务》章节，选择相应的同步或异步时序模型，进行相关寄存器配置；

3）进行 PSRAM 访问。

25.4.3. SRAM 设置流程

1）将 SRAM 对应 IO 口的外设复用功能配置到 FMC 功能；

2）参考《异步事务》与《同步事务》章节，选择相应的同步或异步时序模型，进行相关寄存器配置；

3）进行 SRAM 访问。

25.4.4. LCD8080 设置流程

1）将 LCD8080 对应 IO 口的外设复用功能配置到 FMC 功能；

2）参考《异步事务》与《同步事务》章节，选择相应的同步或异步时序模型，进行相关寄存器配置；

3）进行 LCD8080 访问。

25.5. FMC_NORSRAM 寄存器描述

25.5.1. 寄存器列表

FMC_NORSRAM 模块寄存器基地址：0xA0000000

偏移	名称	复位值	描述
0x00	FMC_SNCTL1	0x000030da	NORSRAM 控制寄存器 1
0x08	FMC_SNCTL2	0x000030da	NORSRAM 控制寄存器 2
0x10	FMC_SNCTL3	0x000030da	NORSRAM 控制寄存器 3
0x18	FMC_SNCTL4	0x000030da	NORSRAM 控制寄存器 4
0x04	FMC_SNTCFG1	0x0fffffff	NORSRAM 时序配置寄存器 1
0x0C	FMC_SNTCFG2	0x0fffffff	NORSRAM 时序配置寄存器 2
0x14	FMC_SNTCFG3	0x0fffffff	NORSRAM 时序配置寄存器 3
0x1C	FMC_SNTCFG4	0x0fffffff	NORSRAM 时序配置寄存器 4
0x104	FMC_SNWTCFG1	0x0fffffff	NORSRAM 写时序寄存器 1
0x10C	FMC_SNWTCFG2	0x0fffffff	NORSRAM 写时序寄存器 2
0x114	FMC_SNWTCFG3	0x0fffffff	NORSRAM 写时序寄存器 3
0x11C	FMC_SNWTCFG4	0x0fffffff	NORSRAM 写时序寄存器 4

25.5.2. NORSRAM 控制寄存器 1~4(FMC_SNCTLx: 00h+8*(x-1), x=1~4)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19	SYNCWR	RW	0	选择写操作模式 0：异步写操作 1：同步写操作
18:16	CPS	RW	000	CRAM 页大小 000： 页边界自动突发分割 001： 128 字节 010： 256 字节 011： 512 字节 100： 1024 字节 其他：保留

15	ASYNCWAIT	RW	0	异步等待功能使能位 0： 禁用异步等待功能 1： 使能异步等待功能
14	EXMODEN	RW	0	扩展模式使能 0： 禁用扩展模式 1： 使能扩展模式
13	NRWTEN	RW	1	NWAIT 信号使能 对于存储器的突发模式访问，该位使能/禁用在 NWAIT 信号中插入等待状态功能。 0： 禁用 NWAIT 信号 1： 使能 NWAIT 信号
12	WREN	RW	1	写操作使能 0： 禁止 FMC 对外部存储器的写操作，否则产生一个 AHB 错误 1： 允许 FMC 对外部存储器的写操作（复位缺省值）
11	NRWTCFG	RW	0	NWAIT 信号配置， 只在同步模式有效 0： NWAIT 信号在等待状态前的一个数据周期有效 1： NWAIT 信号在等待状态期间有效
10	WRAPEN	RW	0	非对齐突发模式使能 0： 禁止非对齐突发操作 1： 允许非对齐突发操作
9	NRWTPOL	RW	0	NWAIT 信号极性 0： NWAIT 低电平有效 1： NWAIT 高电平有效
8	SBRSTEN	RW	0	同步突发模式使能 0： 禁止同步突发模式 1： 使能同步突发模式
7	RSV	RW	1	保留
6	NREN	RW	1	NOR 闪存访问使能 0：禁止 NOR Flash 访问 1：允许 NOR Flash 访问
5:4	NRW	RW	01	存储器数据宽度 00： 8 位 01： 16 位（复位缺省值） 10/11：保留
3:2	NRTP	RW	10	存储器类型 00： SRAM 01： PSRAM（CRAM） 10： NOR Flash（复位之后的默认值） 11：保留
1	NRMUX	RW	1	数据线/地址线复用 0：禁用地址/数据复用功能 1：允许地址/数据复用功能

0	NRBKEN	RW	00	存储块使能 0：禁用对应的存储器块 1：使能对应的存储器块

25.5.3. NORSRAM 时序配置寄存器 1~4(FMC_SNTCFGx: 04h+8*(x-1), x=1~4)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:28	ASYNCMOD	RW	00	异步访问模式 该位域仅在当 FMC_SNCTL 寄存器的 EXMODEN 位为 1 时有效。 00： 模式 A 01： 模式 B 10： 模式 C 11： 模式 D
27:24	DLAT	RW	1111	NOR Flash 数据延时，仅在同步模式有效 0x0： 第一组突发传输时，数据延迟时间为 2 个 FMC_CLK 时钟周期 0x1： 第一组突发传输时，数据延迟时间为 3 个 FMC_CLK 时钟周期 …… 0xF： 第一组突发传输时，数据延迟时间为 17 个 FMC_CLK 时钟周期
23:20	CKDIV	RW	1111	同步模式时钟分频比，仅在同步模式有效 0x0：保留 0x1： FMC_CLK 周期=2 个 HCLK 周期 …… 0xF： FMC_CLK 周期=16 个 HCLK 周期
19:16	BUSLAT	RW	1111	总线延迟时间 在复用读模式中使用，避免总线冲突，是总线恢复到高阻态的最小时间。 0x0： 总线延迟=1 个 HCLK 周期 0x1： 总线延迟=2 个 HCLK 周期 …… 0xF： 总线延迟=16 个 HCLK 周期
15:8	DSET	RW	11111111	异步数据建立时间 该位域仅在异步模式有效。 0x00：保留 0x01： 数据建立时间=2 个 HCLK 周期 …… 0xFF： 数据建立时间=256 个 HCLK 周期

7:4	AHLD	RW	1111	异步地址保持时间 该位域设置地址保持时间， 仅在模式 D 与复用模式有效。 0x0：保留 0x1： 地址保持时间=2 个 HCLK …… 0xF： 地址保持时间=16 个 HCLK
3:0	ASET	RW	1111	异步地址建立时间 该位域设置地址建立时间。 注意： 该位域仅在 SRAM， ROM， NOR Flash 的异步模式有效。 0x0： 地址建立时间= 1 个 HCLK …… 0xF： 地址建立时间= 16 个 HCLK

25.5.4. NORSRAM 写时序寄存器 1~4(FMC_SNWTCFGx: 104h+8*(x-1), x=1~4)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:28	WASYNCMOD	RW	00	异步访问模式 该位域仅在当 FMC_SNCTL 寄存器的 EXMODEN 位为 1 时有效。 00： 模式 A 01： 模式 B 10： 模式 C 11： 模式 D
27:24	DLAT	RW	1111	NOR Flash 数据延时，仅在同步模式有效 0x0： 第一组突发传输时，数据延迟时间为 2 个 FMC_CLK 时钟周期 0x1： 第一组突发传输时，数据延迟时间为 3 个 FMC_CLK 时钟周期 …… 0xF： 第一组突发传输时，数据延迟时间为 17 个 FMC_CLK 时钟周期
27:20	RSV	RW	1111	保留
19:16	WBUSLAT	RW	1111	总线延迟时间 在复用读模式中使用，避免总线冲突，是总线恢复到高阻态的最小时间。 0x0： 总线延迟=1 个 HCLK 周期 0x1： 总线延迟=2 个 HCLK 周期 …… 0xF： 总线延迟=16 个 HCLK 周期
15:8	WDSET	RW	11111111	异步数据建立时间 该位域仅在异步模式有效。 0x00：保留 0x01： 数据建立时间=2 个 HCLK 周期 …… 0xFF： 数据建立时间=256 个 HCLK 周期

7:4	WAHLD	RW	1111	异步地址保持时间 该位域设置地址保持时间， 仅在模式 D 与复用模式有效。 0x0：保留 0x1： 地址保持时间=2 个 HCLK …… 0xF： 地址保持时间=16 个 HCLK
3:0	WASET	RW	1111	异步地址建立时间 该位域设置地址建立时间。 注意： 该位域仅在 SRAM， ROM， NOR Flash 的异步模式有效。 0x0： 地址建立时间= 1 个 HCLK …… 0xF： 地址建立时间= 16 个 HCLK

26. SDRAM 控制器（FMC_SDRAM）

26.1. 概述

SDRAM 控制器的主要功能把 AHB 总线操作转换为 SDRAM 接口协议，用于连接同步 DRAM (SDRAM/Mobile LPSDR SDRAM) 存储器，支持 SDRAM 的读写操作和刷新功能。

26.2. 主要特性

- 两个 SDRAM 存储区域，可独立配置
- 16 位和 32 位数据总线宽度
- 13 位行地址，11 位列地址，4 个内部存储区域：4x16Mx32bit (256 MB)、4x16Mx16bit(128 MB)
- 支持字、半字和字节访问
- SDRAM 时钟可以是 HCLK/2 或 HCLK/3
- 自动进行和 Bank 边界管理
- 多 Bank 乒乓访问
- 可编程时序参数
- 支持自动刷新操作，可编程刷新速率
- 自刷新模式
- 掉电模式
- 通过软件进行 SDRAM 上电初始化
- CAS 延迟 1,2,3
- 读 FIFO 可缓存，支持 8 行 x 32 位深度（ 8 x14 位地址标记）

26.3. SDRAM 外部存储器接口信号

启动时，必须通过用户应用程序对用于连接 FMC SDRAM 控制器与外部 SDRAM 设备的 SDRAM I/O 引脚进行配置。应用程序未使用的 SDRAM 控制器 I/O 引脚可用于其它用途。

表 26-1 SDRM 信号

SDRAM 信号	I/O 类型	说明	复用功能
SDCLK	O	SDRAM 时钟	-
SDCKE[1:0]	O	SDCKE0：SDRAM 存储区域 1 时钟使能 SDCKE1：SDRAM 存储区域 2 时钟使能	-
SDNE[1:0]	O	SDNE0：SDRAM 存储区域 1 芯片使能 SDNE1：SDRAM 存储区域 2 芯片使能	-
A[12:0]	O	地址	FMC_A[12:0]
D[31:0]	I/O	双向数据总线	FMC_D[31:0]
BA[1:0]	O	存储区域地址	FMC_A[15:14]
NRAS	O	行地址选通	-
NCAS	O	列地址选通	-

SDNWE	O	写入使能	-
NBL[3:0]	O	写访问的输出字节屏蔽（存储器信号名称： DQM[3:0]）	FMC_NBL[3:0]

26.4. 功能描述

所有 SDRAM 控制器输出（信号、地址和数据）在存储器时钟（FMC_SDCLK）的下降沿上变化。

26.4.1. SDRAM 初始化

初始化序列通过软件进行管理。如果使用了两个存储区域，则必须将 FMC_SDRCMD 寄存器中的目标存储区域位 CTD1 和 CTD2 置 1，同时为存储区域 1 和存储区域 2 生成初始化序列。

1）将存储器件的特性编程到 FMC_SDRCRx 寄存器中。 SDRAM 时钟频率、 RBURST 和 RPIPE 特性必须编程到 FMC_SDRCR1 寄存器中。

2）将存储器设备的时序编程到 FMC_SDRTRx 寄存器中。TRP 和 TRC 时序必须编程到 FMC_SDRTR1 寄存器中。

3）将 CMD 位置为“001”并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）以开始为存储器提供时钟信号（ SDCKE 驱动为高电平）。

4）等待指定延迟周期。典型延迟为 100 μs（有关上电后所需延迟的信息，请参见 SDRAM 数据手册）。

5）将 CMD 位置为“010”并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）以发送“全部预充电”命令。

6）将 CMD 位置为“011”并配置 FMC_SDRCMD 寄存器中的目标存储区位（ CTD1 和/或 CTD2）和连续自动刷新命令（NARF）的数量。请参见 SDRAM 数据手册了解应发出的自动刷新命令个数（通常为 8 个）。

7）配置 MRD 字段，将 CMD 位置为“100”并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）以发送“加载模式寄存器”命令并对 SDRAM 设备进行编程。尤其突发长度（BL）必须置“1”且必须选择 CAS 延迟。如果两个 SDRAM 存储区域的模式寄存器不同，则此步骤必须重复两次，每个存储区域各一次且目标存储区域位相应置 1。对于移动 SDRAM 器件， MRD 字段还用于配置扩展模式寄存器，同时发出加载模式寄存器命令。

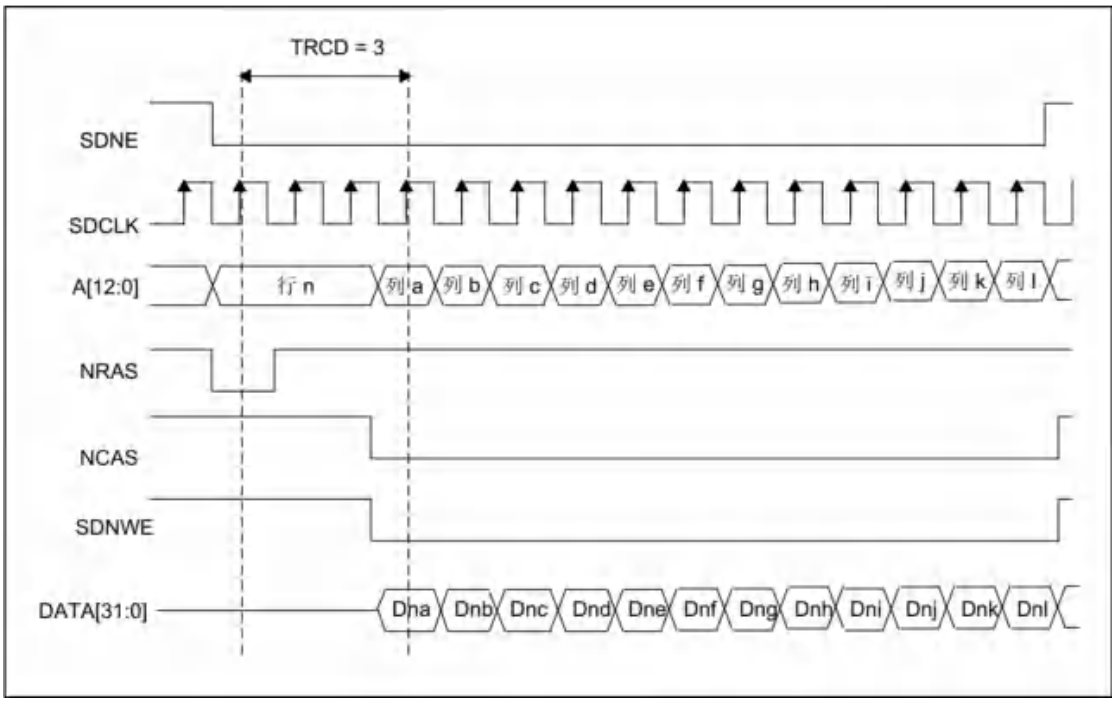
8）编程 FMC_SDRART 寄存器中的刷新速率。刷新速率对应于刷新周期之间的延迟，值必须与 SDRAM 设备相适应。

9）在这一阶段， SDRAM 设备已做好接受命令的准备。如果进行 SDRAM 访问期间发生系统复位，则数据总线仍可能由 SDRAM 设备驱动。因此，必须在复位后重新初始化 SDRAM 设备。

26.4.2. SDRAM 写访问

SDRAM 控制器可接收单次的和突发的写请求，并将其视为单次存储器访问。在这两种情况下， SDRAM 控制器都会跟踪各 Bank 的有效行，以能够对不同的 Bank 进行连续的写访问（多 Bank）。执行任何写访问前，必须将 FMC_SDRCRx 寄存器中的 WP 位清零，禁止 SDRAM 存储区域的写保护。

图 26-1 SDRAM 写波形



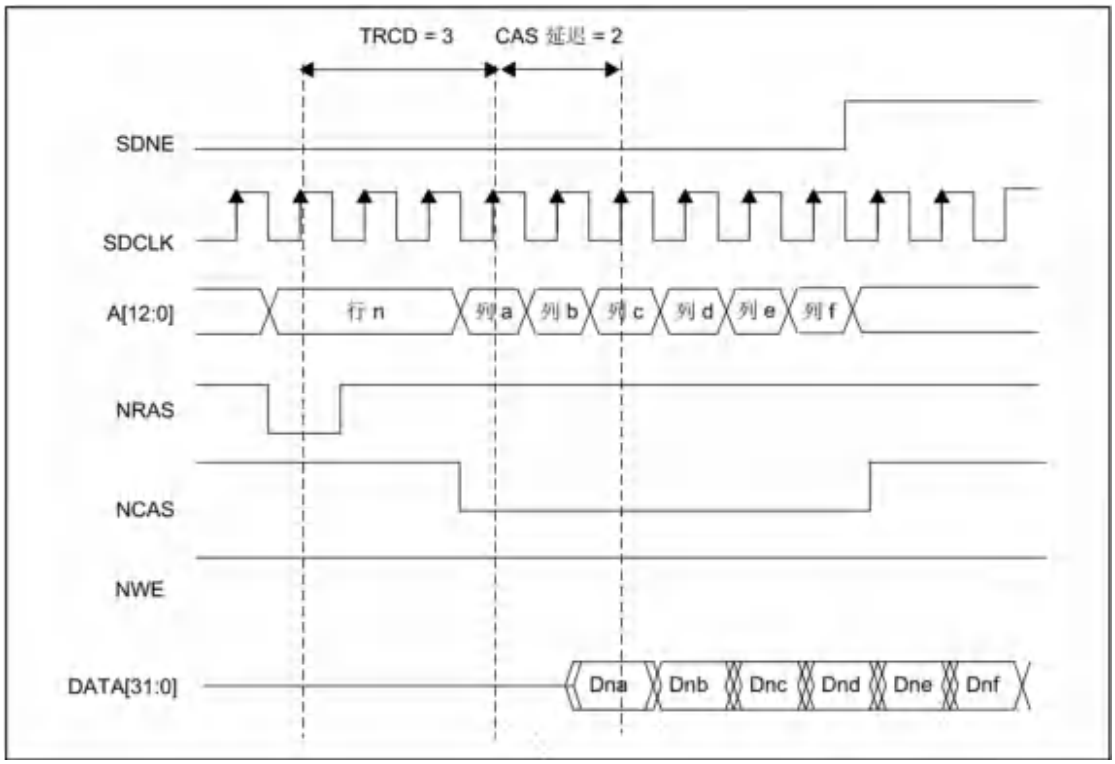
SDRAM 控制器始终会检查下一个访问。

- 如果下一个访问发生在同一行或在其它有效行，则直接执行写操作。
- 如果下一个访问指向一个无效行，则 SDRAM 控制器将生成预充电命令、激活该新行并初始化写命令。

26.4.3. SDRAM 读访问

SDRAM 控制器可接收单次的和突发的读请求，并将其视为单次存储器访问。在这两种情况下， SDRAM 控制器都会跟踪各 Bank 的有效行，以能够对不同的 Bank 进行连续的读访问（多 Bank 乒乓访问）。

图 26-2 SDRAM 读波形



FMC SDRAM 控制器具有可缓存的读取 FIFO（8 行 x 32 位），用于存储在 CAS 延迟周期（最多 3 个存储器

时钟周期，在 FMC_SDRCRx 中配置）和 RPIPE 延迟（设为 2 个 hclk 时钟周期时，在 FMC_SDRCR1 中配置）期间提前读取的数据，通过软件设置。建议软件依据的公式如下：

缓存数据个数 CAS atency + 1 + (RPIPE DIV2)。

必须将 FMC_SDRCR1 寄存器中的 RBURST 位置 1 才能使能缓存功能。

示例

- CAS=3， RPIPE= 2xhclk。这种情况下， FIFO 中将存储 5 个未提交的数据（CAS 延迟期间读取的 4 个数据和 RPIPE 延迟期间读取的 1 个数据）。
- CAS=3， RPIPE= 1xhclk。这种情况下，FIFO 中将存储 4 个未提交的数据（CAS 延迟期间读取的 4 个数据）

读 FIFO 的每行都具有 14 位地址标记用于标识自身内容：11 位用于表示列地址， 2 位表示 bank 地址，1 位用于选择 SDRAM 设备

在突发读取事务期间，如果提前到达行末尾，则提前读取的数据（未提交）不存储到读 FIFO。对于单次读访问，数据将正确存储到读 FIFO。

每当出现读请求时， SDRAM 控制器将检查：

- 地址与地址标记之一是否匹配，如果找到匹配，则直接从 FIFO 读取数据并清空相应地址标记/行内容，同时压缩 FIFO 中的剩余数据以避免空行。
- 否则，向存储器发送新的读命令并用新数据更新 FIFO。如果 FIFO 已满，则较早的数据将丢失

图 26-3 FIFO 未命中读操作（CAS=2， RPIPE=0）

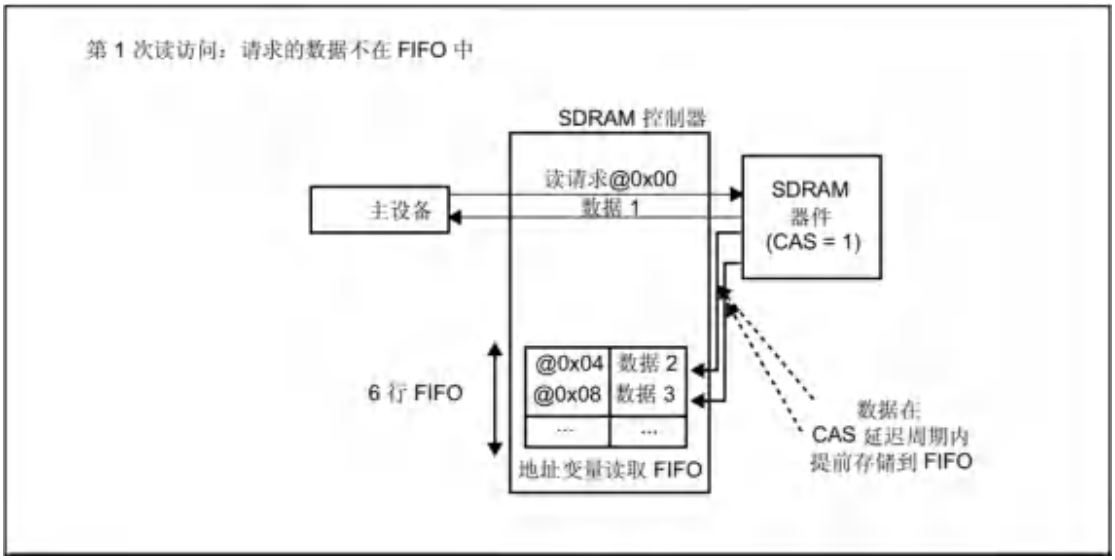
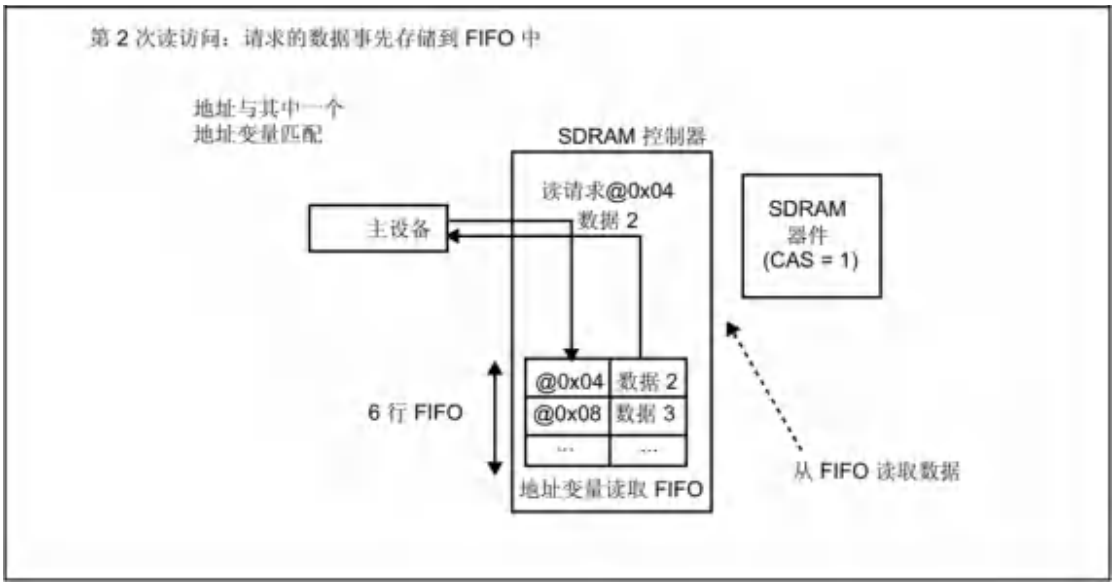


图 26-4 FIFO 命中读操作



写访问或预充电命令期间，读 FIFO 将刷新并做好填充新数据的准备。

接到第一个读请求后，如果当前访问还未进行到行边界，则 SDRAM 控制器将在 CAS 延迟周期和 RPIPE 延迟（如果已配置）期间接受下一个读访问。这将通过递增存储器地址来实现。必须满足以下条件：

- FMC_SDRCR1 寄存器中的 RBURST 控制位必须置“1”。

地址管理取决于下一个 AHB 请求：

- 下一个请求是连续的（突发访问）。

这种情况下， SDRAM 控制器将递增地址。

- 下一个请求不连续
  - 如果新的读请求指向与上一请求相同的行或另一个有效行，则新地址将传送给存储器，同时主设备在 CAS 延迟周期停止工作，等待从存储器获取新数据。
  - 如果新的读请求指向无效行，则 SDRAM 控制器生成预充电命令、激活该新行并初始化读命令。

如果 RBURST 位置 0，则不使用读 FIFO。

26.4.4. 行和 Bank 边界管理

当读/ 写访问跨越了行边界时，如果下一个读/ 写访问是连续的，并且当前访问已执行到行边界，则 SDRAM 控制器将执行以下操作：

1）对有效行进行预充电

2）激活新行

3）启动读/写命令

对于各种列和数据总线宽度配置，都支持在行边界自动激活下一行。

SDRAM 控制器可根据需要在以下命令之间插入附加时钟周期：

- 在预充电和激活命令之间插入以匹配 TRP 参数（仅当下一个访问指向同一存储区域中的其它行时）
- 在激活和读命令之间插入以匹配 TRCD 参数

这些参数在 FMC_SDRTRx 寄存器中定义。

有关跨越行边界读取和突发写访问的信息，请参见下图。

图 26-5 跨行边界的读访问

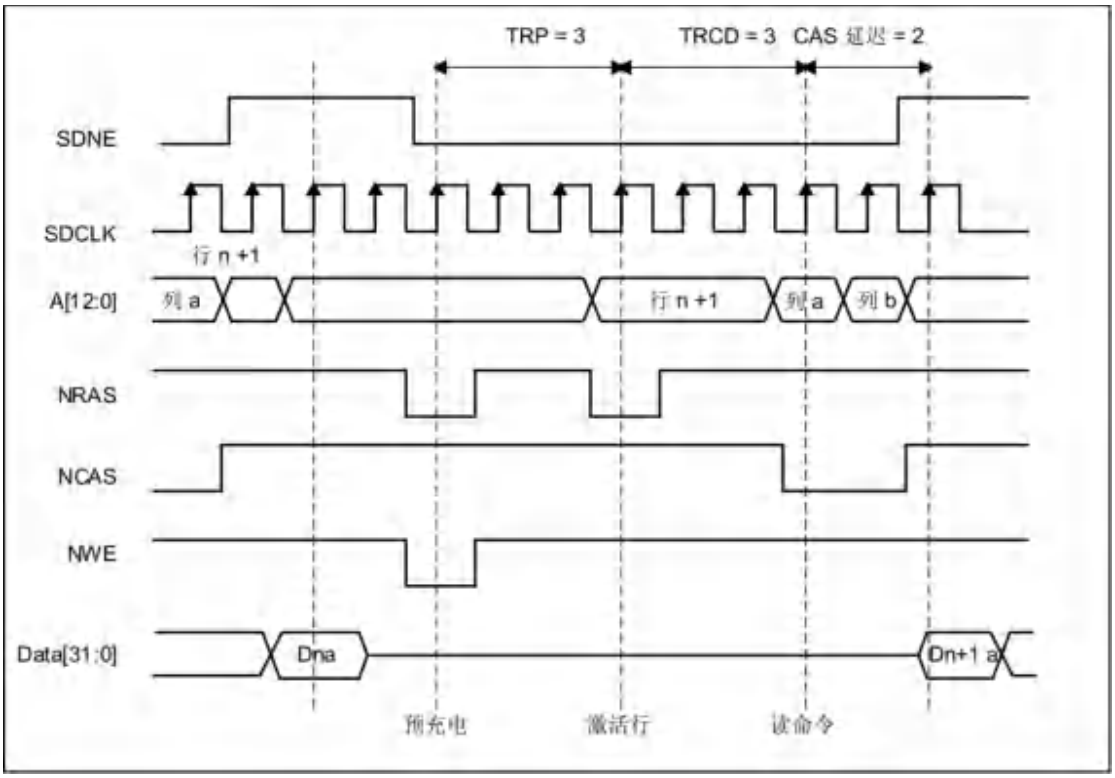
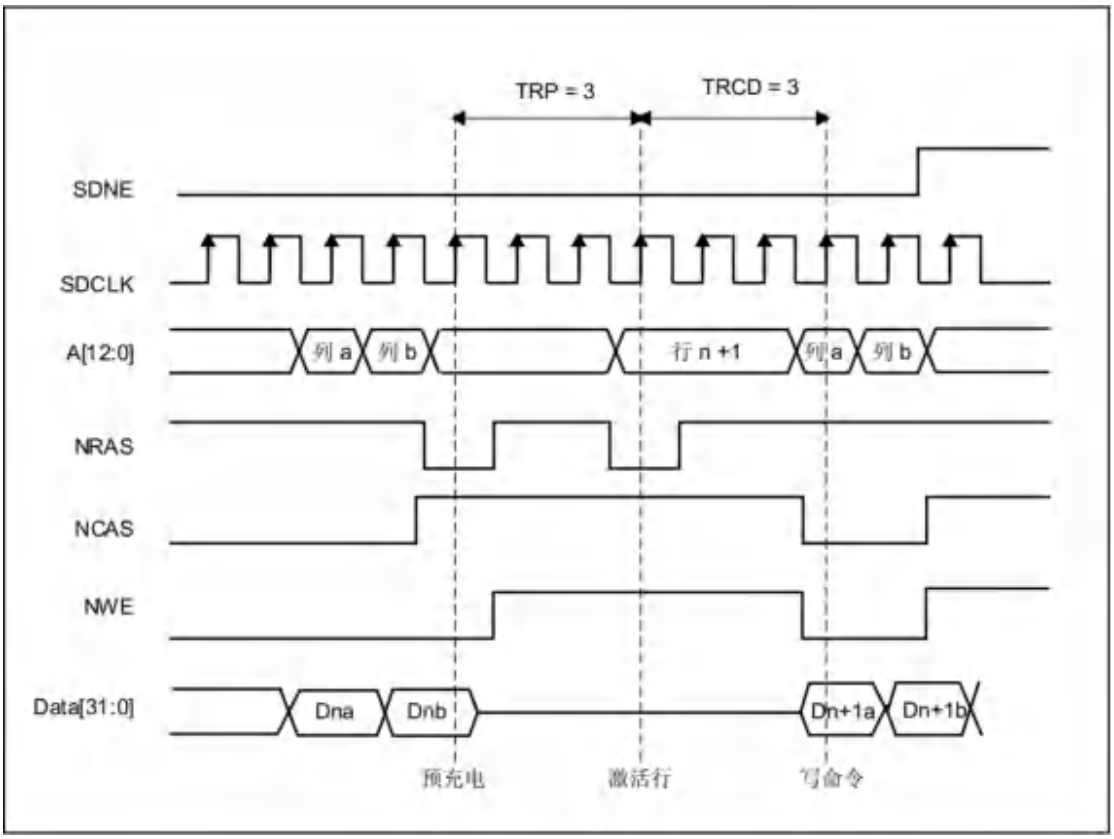


图 26-6 跨行边界的写访问



如果下一个访问是连续的并且当前访问跨越了 Bank 边界，则 SDRAM 控制器将激活下一个 Bank 的第一个行并发出新的读/写命令。可能存在以下两种情况：

- 如果当前 Bank 不是最后一个存储区域，则必须对新 Bank 中的激活行进行预充电。对于各种行/列和数据总线宽度配置，都支持在存储区域边界自动激活下一行。
- 如果当前 Bank 是最后一个 Bank，则仅在对 13 位行、11 位列、4 个内部 Bank 和 32 位数据总线的 SDRAM 设备寻址时，支持自动激活下一行。否则，将违反 SDRAM 地址范围并生成从错误。

- 对于 13 位行地址、 11 位列地址、 4 个内部 Bank 和总线宽度为 32 位的 SDRAM 存储器， SDRAM 控制器将通过第二个 SDRAM 设备继续进行读/ 写操作（假设第二个 SDRAM 设备已初始化）：

a）SDRAM 控制器将激活第一行（在对有效行预充电之后，假设第一个内部 Bank 中存在有效行）并发出新的读/写命令。

b）如果第一行已激活，则 SDRAM 控制器将仅发出读/写命令。

26.4.5. SDRAM 刷新周期

自动刷新命令用于刷新 SDRAM 设备。 SDRAM 控制器会定期发送自动刷新命令。它使用一个内部计数器装载 FMC_SDRART 寄存器中的 COUNT 值。该值定义刷新周期之间的存储器时钟周期个数（刷新速率）。该计数器的值达到零时将生成一个内部脉冲。

如果存在进行中的存储器访问，则会延迟自动刷新请求。不过，在存储器访问和自动刷新请求同时出现时，则优先处理自动刷新请求。

如果在自动刷新期间访问存储器，则会缓存访问请求并在自动刷新完成后进行处理。

如果在上一个自动刷新请求尚未完成的情况下又出现了新的自动刷新请求，则状态寄存器中的 RFE（刷新错误）位将置 1。该位如果已使能（ RFEIE = “1”），将生成中断。

如果 SDRAM 的行不是空闲状态（并非所有行都已关闭），则 SDRAM 控制器将生成 PALL（全部预充电）命令，然后再进行自动刷新。

如果由 FMC_SDRCMD 命令模式寄存器（模式位 = “011”）生成自动刷新命令，则必须先发出 PALL 命令（模式位 = “010”）。

26.4.6. 低功耗模式

可使用两种低功耗模式：

- 自刷新模式：由 SDRAM 设备自身执行自动刷新循环以保留数据，无需外部时钟。
- 掉电模式：由 SDRAM 控制器执行自动刷新循环。

26.4.6.1. 自刷新模式

通过将 CMD 位置为 “101” 并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）来选择该模式。

SDRAM 时钟在 TRAS 延迟后停止运行，而内部刷新定时器只有在满足以下条件之一时才停止计数：

- 向两个设备都发出了自刷新命令。
- 其中一个设备未激活（ SDRAM 存储区域未初始化）。

进入自刷新模式前， SDRAM 控制器会自动发送 PALL 命令。

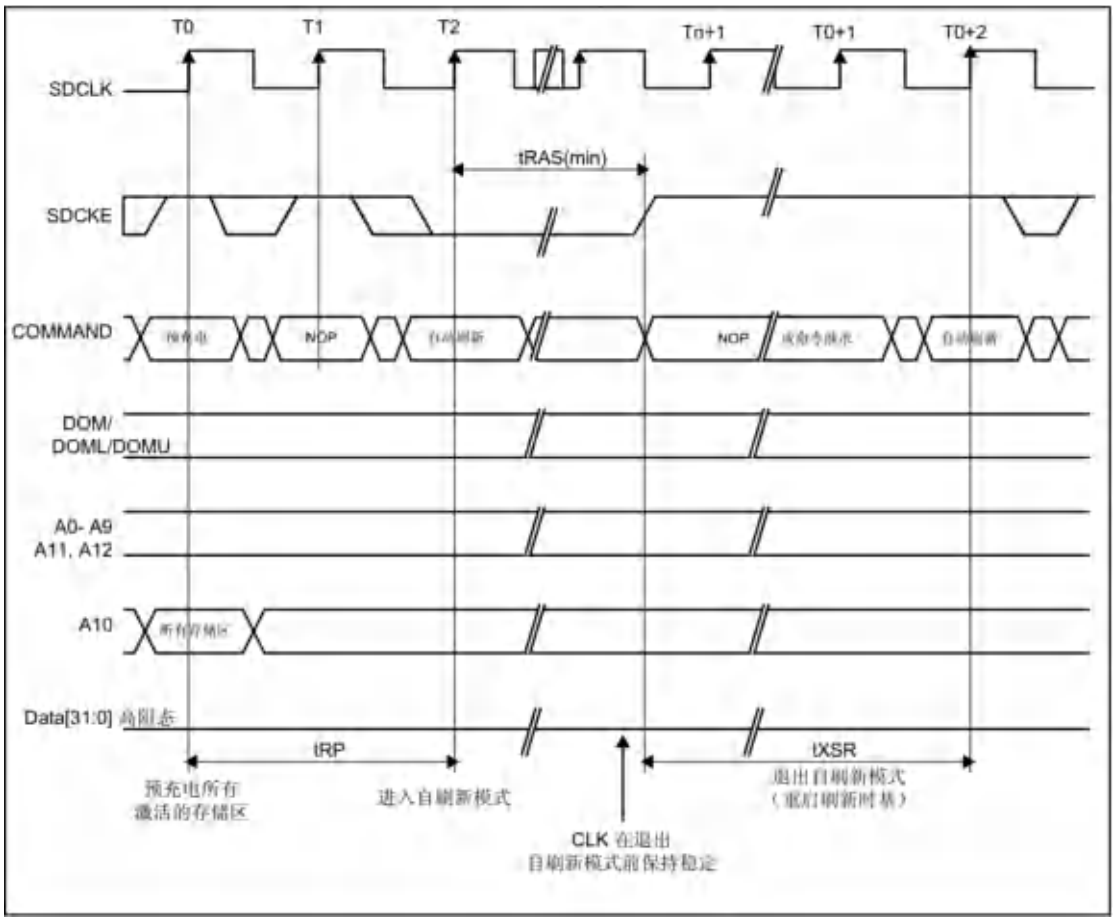
在自刷新模式下，除保持低电平的 SDCKE 外， SDRAM 设备的所有输入都无效。

SDRAM 设备必须处于自刷新模式最短为 TRAS 时间，且能够在更长的时间内始终处于自刷新模式。为保证这一最短时长，在自刷新激活后的 TRAS 延迟期间， BUSY 状态标志将保持高电平。

SDRAM 控制器会在有 SDRAM 设备被选定后立即生成一个命令序列以退出自刷新模式。存储器访问完成后，选定的设备将保持正常模式。

要退出自刷新模式，必须将 CMD 位置为 “000” （正常模式）并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）。

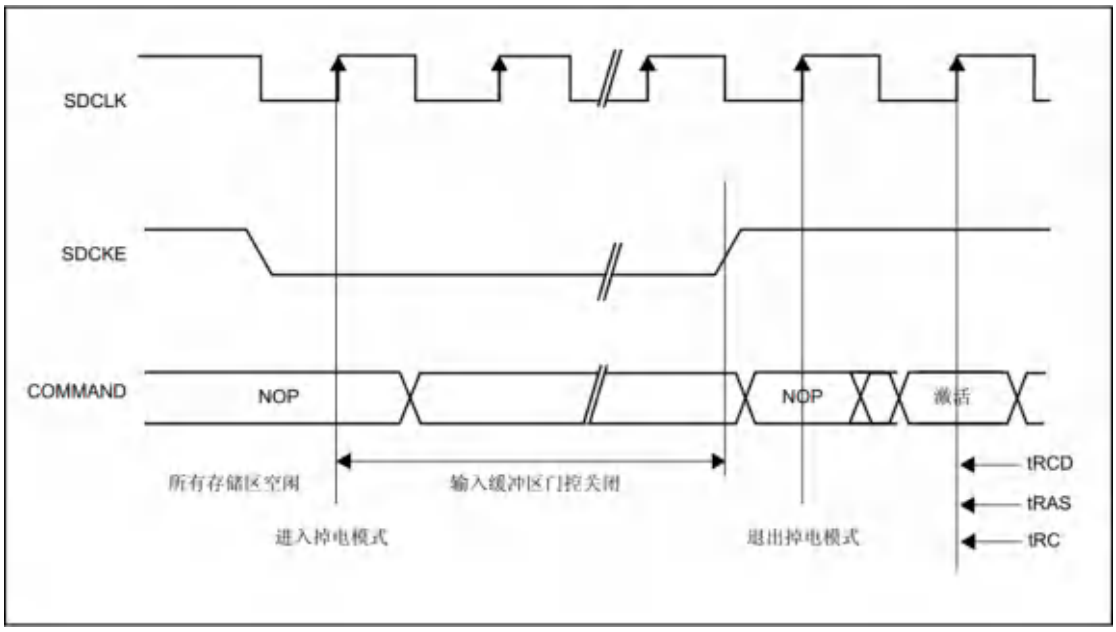
图 26-7 自刷新模式



26.4.6.2. 掉电模式

通过将 CMD 位置为“110”并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）来选择该模式。

图 26-8 掉电模式



SDRAM 控制器会在有 SDRAM 设备被选定后立即退出掉电模式。存储器访问完成后，选定的 SDRAM 设备将保持正常模式。

在掉电模式期间，将禁用 SDRAM 设备的所有输入/输出缓冲区，只有保持低电平的 SDCKE 除外。

SDRAM 设备保持掉电模式的时间不会长于刷新周期，并且自身无法执行自刷新循环。因此， SDRAM 控制器通过以下操作执行刷新：

1）退出掉电模式并将 SDCKE 驱动为高电平

2）生成 PALL 命令（前提是在掉电模式下存在激活行）

3）生成自动刷新命令

4）再次将 SDCKE 驱动为低电平以返回掉电模式

要退出掉电模式，必须将 CMD 位置为“000”（正常模式）并配置 FMC_SDRCMD 寄存器中的目标存储区域位（ CTD1 和/或 CTD2）。

26.5. FMC_SDRAM 控制器寄存器描述

26.5.1. 寄存器列表

FMC_SDRAM 寄存器基地址：0xA0000140

偏移	名称	复位值	描述
0x00	FMC_SDRCR1	0x00390000	SDRAM 控制寄存器 1
0x04	FMC_SDRCR2	0x00390000	SDRAM 控制寄存器 2
0x08	FMC_SDRTR1	0x0fffffff	SDRAM 时序寄存器 1
0x0c	FMC_SDRTR2	0x0fffffff	SDRAM 时序寄存器 2
0x10	FMC_SDRCMD	0x00000000	SDRAM 命令模式寄存器
0x14	FMC_SDRART	0x00000000	SDRAM 刷新定时器寄存器
0x18	FMC_SDRSR	0x00000000	SDRAM 状态寄存器
0x1c	FMC_SDRSMA1	0x00000000	SDRAM 采样微调寄存器 1
0x20	FMC_SDRSMA2	0x00000000	SDRAM 采样微调寄存器 2

26.5.2. 控制寄存器 1,2(FMC_SDRCRx: 00h+4*(x–1), x=1,2)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留
21	RBUSRLS2	RW	1	RBURST 有效时，读 FIFO 是否提前释放总线。FMC 的 IO 在片外存储器、SDRAM 和 NAND 之间共用。在 SDRAM 预读期间，有片外存储器和 NAND 的访问会影响预读，因为只有系统不使用片外存储器和 NAND 时，可以使能 BUSRLS 位。 0：读 FIFO 不提早释放总线 1：读 FIFO 提早释放总线 注： SDRCR2 寄存器中的相应位为只读位

20	WBUSRLS	RW	1	写期间，提早一个时钟释放总线。。FMC 的 IO 在片外存储器、SDRAM 和 NAND 之间共用。在 SDRAM 预读期间，有片外存储器和 NAND 的访问会影响预读，因为只有系统不使用片外存储器和 NAND 时，可以使能 BUSRLS 位。 0：不提早释放 1：提早一个时钟释放总线 注： SDRCR2 寄存器中的相应位为只读位
19	RBUSRLS	RW	1	RBURST 有效时，预读期间是否提前释放总线。FMC 的 IO 在片外存储器、SDRAM 和 NAND 之间共用。在 SDRAM 预读期间，有片外存储器和 NAND 的访问会影响预读，因为只有系统不使用片外存储器和 NAND 时，可以使能 BUSRLS 位。 0：预读期间不提早释放总线 1：预读期间提早释放总线 注： SDRCR2 寄存器中的相应位为只读位
18:16	BDEPTH	RW	1	Burst read 缓存深度，不能为 0，7 为最大值缓存数 7 个数据 在 RBURST 为 1 时有效。
15	W2R_DLY	RW	0	写转读连续操作加 SDCLK 延迟使能。在 CAS 为 1/2，写转读连续操作时候是否加 1/2 个 SDCLK NOP 操作。 注： SDRCR2 寄存器中的相应位为只读位。
14:13	RPIPE	RW	0	读管道 (Read pipe) 这些位可定义在 CAS 延时后延后多少个 hclk 时钟周期读取数据。 00：无 hclk 时钟周期延迟 01：一个 hclk 时钟周期延迟 10：两个 hclk 时钟周期延迟 11：备用 注： SDRCR2 寄存器中的相应位为只读位。
12	RBURST	RW	0	Burst read 此位可使能 Burst Read 模式。 SDRAM 控制器预期在 CAS 延迟期间接受下一个读命令并将数据存储在读 FIFO 中。 0：不将单次读请求作为突发请求管理 1：始终将单次读请求作为突发请求管理 注： SDRCR2 寄存器中的相应位为只读位
11:10	CLKDIV	RW	0	SDRAM 时钟配置 这些位用于定义两个 SDRAM 存储区域的 SDRAM 时钟周期以及在更改频率前禁止时钟。此时，必须重新初始化 SDRAM。 00：禁止 SDCLK 时钟 01： SDCLK 周期 = 4 个 hclk 周期 10： SDCLK 周期 = 2 个 hclk 周期 11： SDCLK 周期 = 3 个 hclk 周期 注： SDRCR2 寄存器中的相应位为只读位。

9	WP	RW	0	写保护（Write protection） 该位可使能对 SDRAM 存储区域的写模式访问。 0：禁止写保护，允许写访问 1：使能写保护，忽略写访问
8:7	CAS	RW	0	CAS 延迟（CAS Latency） 该位可设置 SDRAM CAS 延迟，按存储器时钟周期计 00：保留。 01： 1 个周期 10： 2 个周期 11： 3 个周期
6	NB	RW	0	内部 Bank 数量（Number of internal banks） 该位可设置内部 Bank 数量。 0： 2 个内部 Bank 1： 4 个内部 Bank
5:4	MWID	RW	0	存储器数据总线宽度（Memory data bus width）。 这些位定义存储器件宽度。 00：8 位 01： 16 位 10： 32 位 11： 保留。
3:2	NRA	RW	0	行地址位数（Number of row address bits） 这些位定义行地址的位数。 00： 11 位 01： 12 位 10： 13 位 11： 保留
1:0	NCA	RW	0	列地址位数（Number of column address bits） 这些位定义列地址的位数。 00： 8 位 01： 9 位 10： 10 位 11： 11 位

注： 修改 RBURST 或 RPIPE 设置或者禁止 SDCLK 时钟之前，用户必须先发送 PALL 命令以确保先完成正在进行的操作

26.5.3. 时序寄存器 1,2(FMC_SDRTRx: 08h+4*(x–1), x = 1,2)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留

27:24	TRCD	RW	1111	行到列延迟（Row to column delay） 这些位定义激活命令与读/写命令之间的延迟，按存储器时钟周期数计。 0000： 0 个周期 0001： 2 个周期 .... 1111： 16 个周期
23:20	TRP	RW	1111	行预充电延迟（Row precharge delay） 这些位定义预充电命令与其它命令之间的延迟，按存储器时钟周期数计。仅在 FMC_SDRTR1 寄存器中配置 TRP 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRP。 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期 注： FMC_SDRTR2 寄存器中的相应位为无关位。
19:16	TWR	RW	1111	恢复延迟（Recovery delay） 这些位定义写命令和预充电命令之间的延迟，按存储器时钟周期数计。 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期 注： TWR 必须设置成与 SDRAM 数据手册中定义的写恢复时间（tWR）相匹配，保证： TWR>=TRAS - TRCD，并且 TWR >=TRC - TRCD – TRP 例如： TRAS= 4 个周期， TRCD= 2 个周期。因此， TWR >= 2 个周期。 TWR 必须设置为 0x1。 如果使用了两个 SDRAM 设备，则必须为 SDRTR1 和 SDRTR2 配置相同的 TWR 时 序（对应于较慢的 SDRAM 设备）。
15:12	TRC	RW	1111	行循环延迟（Row cycle delay） 这些位定义刷新命令和激活命令之间的延迟，以及两个相邻刷新命令之间的延迟，以存储器时钟周期数表示。在同一个内部 bank 上两个使能命令之间的延迟。 仅在 FMC_SDRTR1 寄存器中配置 TRC 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRC。 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期 注： TRC 必须与 SDRAM 设备数据手册中定义的 TRC 和 TRFC（自动刷新周期）时序相匹配。 注： SDRTR2 寄存器中的相应位为无关位。

11:8	TRAS	RW	1111	自刷新时间（Self refresh time） 这些位定义最短的自刷新周期，按存储器时钟周期数计。 这些位指定了使能命令与预充电命令之间延迟多少 SDRAM 时钟周期单元 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期
7:4	TXSR	RW	1111	退出自刷新延迟（Exit Self-refresh delay） 这些位定义从发出自刷新命令到发出激活命令之间的延迟，按存储器时钟周期数计。 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期 注： 如果使用了两个 SDRAM 设备，则必须为 SDRTR1 和 SDRTR2 配置相同的 TXSR 时序（对应于较慢的 SDRAM 设备）。
3:0	TMRD	RW	1111	加载模式寄存器到激活（Load Mode Register to Active） 这些位定义加载模式寄存器命令和激活或刷新命令之间的延迟，按存储器时钟周期计。 0000： 1 个周期 0001： 2 个周期 .... 1111： 16 个周期

注： 如果连接了两个 SDRAM 设备，对于命令模式寄存器同时访问这两个设备（加载模式寄存器命令和自刷新命令）的情况，将按 FMC_SDRTR1 寄存器中为存储区域 1 配置的时序参数（ TMRD ，TRAS 和 TXSR 时序）发出访问命令。

仅在 FMC_SDRTR1 寄存器中配置 TRP 和 TRC 时序。如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRP 和 TRC 时序。

26.5.4. 命令模式寄存器（FMC_SDRCMD: 10h)

该寄存器包含访问 SDRAM 设备时所发出的命令。该寄存器用于初始化 SDRAM 设备、激活自刷新模式和掉电模式。写入 CMD 字段后，将根据 CTD1 和 CTD2 命令位向单个或全部两个 SDRAM 存储区域发送命令。该寄存器为两个 SDRAM 存储区域所共用。

位域	名称	属性	复位值	描述
31:23	RSV	-	-	保留
21:9	MRD	RW	0	模式寄存器定义（Mode Register definition） 此 13 位域定义 SDRAM 模式寄存器内容。可通过加载模式寄存器命令对模式寄存器进行编程。 MRD [12:0] 位还用于编程 Mobile SDRAM 的扩展模式寄存器。

8:5	NARF	RW	0	自动刷新次数 (Number of Auto-refresh) 这些位定义 CMD = “011” 时所发出的连续自动刷新命令个数。 0000： 1 个自动刷新周期 0001： 2 个自动刷新周期 .... 1110： 15 个自动刷新周期 1111： 16 个自动刷新周期
4	CTD1	RW	0	命令目标存储器 1 (Command Target Device 1) 该位指示是否向 SDRAM 存储器 1 发送命令。 0：命令不发送到 SDRAM 存储区域 1 1：命令会发送到 SDRAM 存储区域 1
3	CTD2	RW	0	命令目标存储器 2 (Command Target Device 2) 该位指示是否向 SDRAM 存储器 2 发送命令。 0：命令不发送到 SDRAM 存储区域 2 1：命令会发送到 SDRAM 存储区域 2
2:0	CMD	RW	0	命令(Command) 这些位定义发送到 SDRAM 存储器的命令。 000：正常模式 001：时钟配置使能 010：预充电所有存储区域命令(PALL) 011：自动刷新命令(Auto-refresh) 100：加载模式寄存器 101：自刷新命令(Self-refresh) 110：掉电命令 111：保留

注： 命令发出后，至少一个命令目标存储区域位（CTD1 或 CTD2）必须置 1，否则该命令将被忽略。

注： 如果使用两个 SDRAM 存储区域，则必须通过将 CTD1 和 CTD2 位置 1 来向两个器件同时发送自刷新和 PALL 命令，否则该命令将被忽略。

注： 如果只使用一个 SDRAM 存储区域，并通过将其关联的 CTD 位置 1 来发出命令，则未使用存储区域的另一个 CTD 位必须保持为 0。

26.5.5. 刷新定时器寄存器（FMC_SDRART: 14h）

该寄存器通过配置刷新定时器计数值来设置刷新循环之间的刷新速率，按 SDCLK 时钟周期数计。

SDRAM 刷新速率=(SDRAM 刷新周期 /行数)

COUNT= SDRAM 刷新速率× SDRAM 时钟频率-20

示例

SDRAM 刷新速率 = 64 ms /8196=7.81μs

其中 64 ms 是 SDRAM 的刷新周期。

7.81μs ×60MHz = 468.6

如果考虑读请求后出现内部刷新请求，则必须将刷新速率增加 20 个 SDRAM 时钟周期（如上所示）以获得重充足的裕量。其对应的 COUNT 值为“0000111000000” (448)。

该 13 位域将加载到使用 SDRAM 时钟递减的定时器。此定时器在计数到零时生成刷新脉冲。 COUNT 值必须设置为至少 41 个 SDRAM 时钟周期。

一旦完成对 FMC_SDRART 寄存器的编程，定时器就开始计数。如果寄存器中编程的值为“0”，则不会执行刷新。切不可在初始化后重新编程该寄存器以避免刷新速率被修改。

每当生成刷新脉冲时，都会重新将该 13 位 COUNT 字段加载到计数器中。

如果存在进行中的存储器访问，则会延迟自动刷新请求。不过，在存储器访问和自动刷新请求同时出现时，则优先处理自动刷新请求。如果在刷新期间访问存储器，则会缓存访问请求并在刷新完成后进行处理。

此寄存器为 SDRAM 存储区域 1 和存储区域 2 所共用。

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	RFEIE	RW	0	RFE 中断使能 0：禁止中断 1： RFE = 1 时生成中断
13:1	COUNT	RW	0	刷新定时器计数 (Refresh Timer Count) 该 13 位域定义 SDRAM 设备的刷新速率，以存储器时钟周期数表示。该字段必须设置为至少 41 个 SDRAM 时钟周期。 刷新速率 = (COUNT + 1) x SDRAM 频率时钟 COUNT = （ SDRAM 刷新周期/行数） – 20
0	CRFE	RW	0	清除刷新错误标志 (Clear Refresh error flag) 该位用于清除状态寄存器中的刷新错误标志 (RFE)。 0：无影响 1：清除刷新错误标志

注： 所编程的 COUNT 值不可等于以下时序之和： TWR+TRP+TRC+TRCD+4 个存储器时钟周期

26.5.6. 状态寄存器 (FMC_SDRSR: 18h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
5	BUSY	RO	0	该位指定 SDRAM 控制器处理上次命令的状态。 0：SDRAM 控制器准备好接收新命令 1：SDRAM 控制器 BUSY，不能接收新命令
4:3	STA2	RO	0	存储区域 2 的工作状态 (Status for Device 2) 这些位定义 SDRAM 存储区域 2 的工作状态。 00：正常模式 01：自刷新模式 10：掉电模式

2:1	STA1	RO	0	存储区域 1 的工作状态(Status for Device 1) 这些位定义 SDRAM 存储区域 1 的工作状态。 00：正常模式 01：自刷新模式 10：掉电模式
0	RFE	RO	0	刷新错误标志 (Refresh error flag) 0：未检测到刷新错误 1：检测到刷新错误 在 RFEIE = 1 且 RFE = 1 时会生成中断

26.5.7. 采样微调寄存器 1(FMC_SDRSMA1: 1Ch)

位域	名称	属性	复位值	描述
31:0	SMA1	RW	0x0000	对 SDR_IN[31:0]输入采样调节位，每一个 bit 对应一个输入线，优先级低于 SMA2. 0: 对采样沿处的数据进行采样，受 RPIPE 控制 1：对采样沿前一个 HCLK 处的数据进行采样

26.5.8. 采样微调寄存器 2(FMC_SDRSMA2: 20h)

位域	名称	属性	复位值	描述
31:0	SMA2	RW	0x0000	对 SDR_IN[31:0]输入采样调节位，每一个 bit 对应一个输入线，优先级高于 SMA1。 0: 对采样沿处的数据进行采样，受 RPIPE 控制 1：对采样沿前半个 HCLK 处的数据进行采样

27. NAND Flash 控制器（FMC_NAND）

27.1. 概述

NAND Flash 控制器实现将系统 CPU 总线转换成符合外部 NAND Flash 存储器协议的写地址、写命令、写数据、读数据等基本操作。

27.2. 主要特性

- AHB 总线便于系统集成，同时实现高速的读写操作。
- 标准的 SDR 接口，符合 ONFi 标准。
- SDR 接口时序可配置，支持多种型号的 NAND Flash 存储器。
- 支持大小端数据可配置。
- NAND Flash 接口支持 8 比特。
- BCH 算法支持
  - 硬件即时编/译码。
  - 每页可纠正 8 个比特的错误（512 字节数据+3 字节信息位+13 字节 ECC 校验位）
  - 支持不流水线模式。
  - 支持自动纠错。

27.3. 功能描述

27.3.1. 外部存储接口信号

下表列出了 NAND Flash 信号。

表 27-1 8 位 NAND Flash

FMC 信号名称	I/O	功能
A[17]	O	NAND Flash 地址锁存使能(ALE)信号
A[16]	O	NAND Flash 命令锁存使能(CLE)信号
D[7:0]	I/O	8 位复用双向地址/数据总线
NCE	O	片选
NOE(NRE)	O	输出使能（读取使能）
NWE	O	写入使能
NWAIT/INT	I	输入 NAND Flash 就绪/繁忙信号

27.3.2. NAND Flash 支持的存储器和事务

下表为所支持的设备、访问模式和事务。

表 27-2 支持的存储器和事务

设备	模式	R/W	总线宽度	存储器 数据大小	访问次数
8 位 NAND	异步	R	8	8	1 次
	异步	W	8	8	1 次
	异步	R	16	8	2 次
	异步	W	16	8	2 次
	异步	R	32	8	4 次
	异步	W	32	8	4 次

27.3.3. 数据通道访问配置

总线类型	ENDIAN	AHB[31:24]	AHB[23:16]	AHB[15:8]	AHB[7:0]
Byte	0 或 1	——	——	——	D[7:0]
Half-Word 拆成两次 Flash 操作	0（小端）	——	——	——	D[7:0] (1)
		——	——	D[7:0] (2)	——
	1（大端）	——	——	D[7:0] (1)	——
		——	——	——	D[7:0] (2)
Word 拆成四次 Flash 操作	0（小端）	——	——	——	D[7:0] (1)
		——	——	D[7:0] (2)	——
		——	D[7:0] (3)	——	——
		D[7:0] (4)	——	——	——
	1（大端）	D[7:0] (1)	——	——	——
		——	D[7:0] (2)	——	——
		——	——	D[7:0] (3)	——
		——	——	——	D[7:0] (4)

27.3.4. NAND Flash 的时序图

NAND Flash 控制器通过 NFMWST 配置 NAND 接口时序，时间参数周期都使用 FMC_CLK 计算；其中 TWP 和 TWH 定义写时序，TRP 和 TREH 定义读时序，TWHR 和 TRHW 定义读写切换等待时序，TADL 为地址周期到数据数据等待时序。

图 27-1 NAND 写操作时序

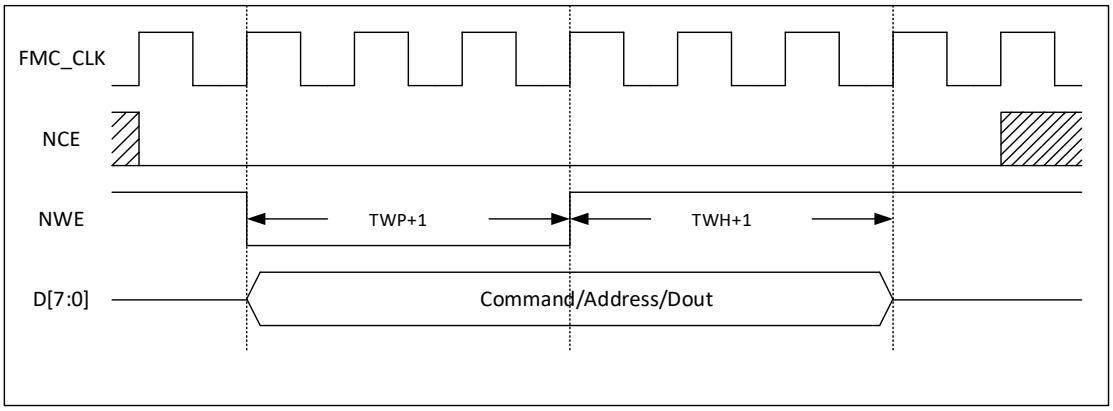


图 27-2 NAND 读时序（EDO=0）

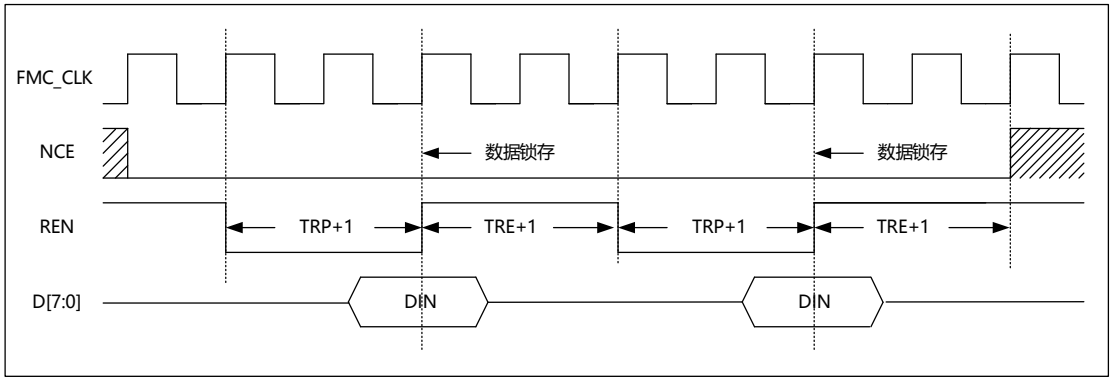


图 27-3 NAND 读时序（EDO=1）

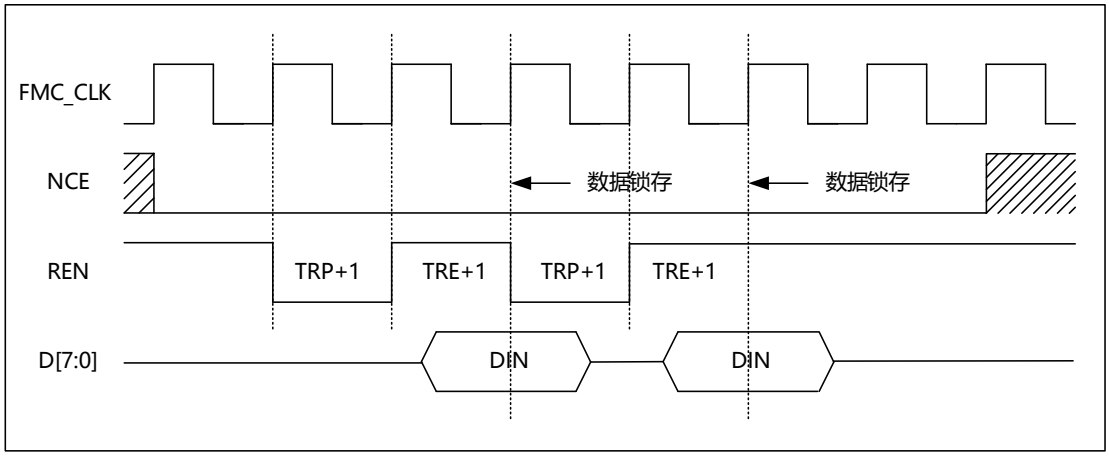


图 27-4 TWHR 时序

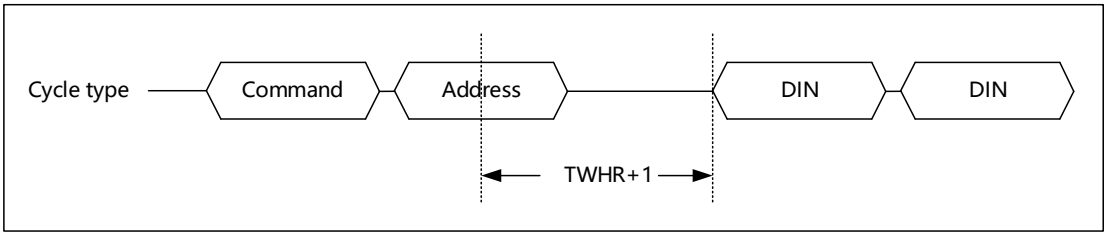


图 27-5 TRHW 时序

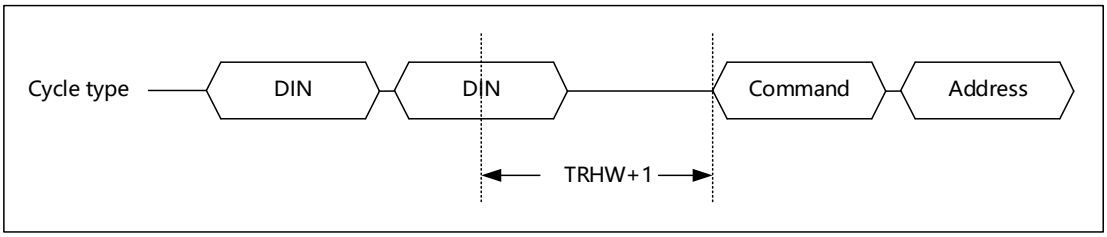
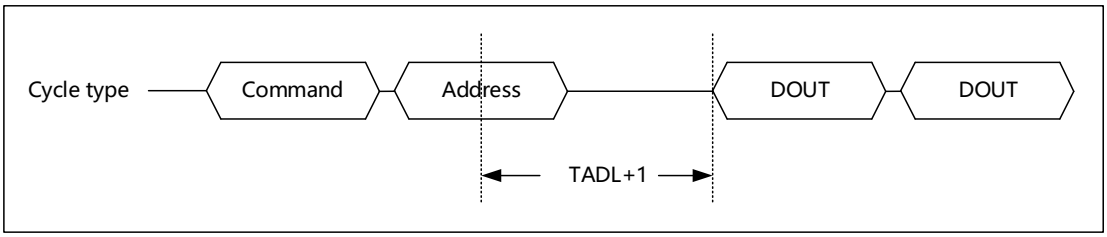


图 27-6 TADL 时序



27.3.5. NAND Flash 操作

NAND Flash 器件的命令锁存使能（CLE）信号和地址锁存使能（ALE）信号由 FMC 控制器驱动。这意味着要向 NAND Flash 发送命令或地址，CPU 必须对 NAND Flash 控制器的特定寄存器执行操作，完成 NAND Flash 的读写。其中，NFMCMD 为 NFM 命令通道寄存器，对其进行写操作会拉高 CLE 信号发起 Command 序列操作；NFMADDR 为 NFM 地址通道寄存器，对其进行写操作会拉高 ALE 信号发起 Address 序列操作。NFMECCDATA 和 NFMNECCDATA 为数据通道寄存器，对其进行操作发现数据的读写操作。

■ 对 NAND Flash 设备进行的读写取操作需要执行以下步骤：

1）设置 NFMWST 设置各种时序周期。

2）根据应用，设置 NFMCTRL 的 ENDIAN 和 EDO_EN

3）设置 NFMCTRL 的 FCE 为 0，设置片选 NCE 为 0

4）对 NFMCMD 进行写操作，发送 Command

5）对 NFMADDR 进行写操作，发送 Address。每次写发送 8 位地址，多次写直到发送所有地址完成。

6）如果需要纠错功能，对 NFMECCDATA 进行读或者写操作，发起 NAND Flash 的读或者写操作；同时，数据会发送 BCH 模块进行纠错计算。如果不需要纠错功能，对 NFMECCDATA 进行读或者写操作，只发起发起 NAND Flash 的读或者写操作，数据不会给 BCH 模块。对 NFMECCDATA（或 NFMECCDATA）进行多次操作，直到需要的 NAND Flash 读写完成，通常是一个 page。

27.3.6. BCH 操作

BCH 模块完成纠错功能，CPU 通过 NFMECCDATA 寄存器对 NAND Flash 操作时，会把数据发送到 BCH 模块。BCH 模块会完成纠错或者产生校验码。BCH 操作流程如下。

27.3.6.1. BCH 编码流程

1）设置 BCH 配置寄存器 BCH_CONFIG

2）设置 BCH 控制寄存器 BCH_CTRL 中的 Mode 位，将 BCH 设置为编码模式。

3）设置 BCH 状态寄存器 BCH_STATUS 中的 ENCODE_CLR 位，复位 BCH 编码通道。

4）通过 ECC 通道写入 BCH 算法规定的数据。

5）读 BCH 校验码寄存器。

27.3.6.2. BCH 译码自动纠错模式流程

1）设置 BCH 配置寄存器 BCH_CONFIG

2）设置 BCH 控制寄存器 BCH_CTRL 中的 Mode 位，将 BCH 设置为译码模式。

3）设置 BCH 控制寄存器 BCH_CTRL 中的 Auto_Correct 位，使能自动纠错功能。

4）设置 BCH 页计数寄存器 BCH_PAGENUM 为 0。

5）设置 BCH 纠错基地址寄存器 BCH_BASEADDR。

6）通过 ECC 通道读出 BCH 算法规定的数据。

7）等待 SNYDROME_DONE 置位，置位后清零。

8）重复步骤 5，直到数据全部读出。

9）根据 BCH_PAGENUM 判断纠错是否完成。

27.3.6.3. BCH 译码手动纠错模式流程

1）设置 BCH 配置寄存器 BCH_CONFIG

2）设置 BCH 控制寄存器 BCH_CTRL 中的 Mode 位，将 BCH 设置为译码模式。

3）设置 BCH 控制寄存器 BCH_CTRL 中的 Auto_Correct 位，使能手动纠错功能。

4）通过 ECC 通道读出 BCH 算法规定的数据。

5）等待 CORRECT_DONE 置位。

6）读出 BCH 状态寄存器 BCH_STATUS 中的 ERN 位，得出错误的比特数。

7）将错误地址指针寄存器 BCH_ERRADDRPTR 指向的原数据与错误向量异或得到正确的数据。

27.4. FMC_NAND 寄存器描述

27.4.1. 寄存器列表

FMC_NAND 寄存器基地址：0xA0000200

偏移	名称	复位值	描述
0x00	FMC_NFMCTRL	0x0000000f	NFM 控制寄存器
0x04	FMC_NFMWST	0x00000000	NFM 等待寄存器
0x08	FMC_NFMSTATUS	0x00000000	NFM 状态寄存器
0x10	FMC_BCH_CONFIG	0x00000000	BCH 配置寄存器
0x14	FMC_BCH_CTRL	0x00000000	BCH 控制寄存器
0x18	FMC_BCH_STATUS	0x00000000	BCH 状态寄存器
0x1c	FMC_BCH_CODE	0x00000000	BCH 校验码寄存器
0x20	FMC_BCH_ERRADR	0x00000000	BCH 错误地址寄存器
0x24	FMC_BCH_ERRVEC	0x00000000	BCH 错误向量寄存器

0x28	FMC_BCH_BASEADDR	0x00000000	BCH SRAM 基地址寄存器
0x2C	FMC_BCH_CODEPTR	0x00000000	BCH 校验码指针寄存器
0x30	FMC_BCH_ERRADDRPTR	0x00000000	BCH 错误地址指针寄存器
0x34	FMC_BCH_ERRVECPTR	0x00000000	BCH 错误向量指针寄存器
0x38	FMC_BCH_PAGENUM	0x00000000	BCH 页计数器。
0x3C	FMC_BCH_ADDRLATCH	0x00000000	BCH 地址锁存寄存器
0x40	FMC_NFMCMD	0x00000000	NFM 命令通道寄存器
0x44	FMC_NFMADDR	0x00000000	NFM 地址通道寄存器
0x48	FMC_NFMECCDATA	0x00000000	NFM ECC 数据通道寄存器
0x4C	FMC_NFMNECCDATA	0x00000000	NFM 非 ECC 数据通道寄存器

27.4.2. NFM 控制寄存器(FMC_NFMCTRL: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	EDO_EN	RW	0	EDO 功能使能 0：不使能 EDO 功能 1：使能 EDO 功能
6	RBNINTEN	RW	0	RBN 管脚上升沿状态中断使能 0: POS_RBN 中断不使能 1: POS_RBN 中断使能
5	ENDIAN	RW	0	NFC 数据大小端模式选择，具体选择参见 7.1 1：大端模式 0：小端模式
4	FWP	RW	0	写保护使能。 1：FWP 是 1，禁止写保护。 0：FWP 是 0，使能写保护。
3:0	FCE	RW	1111	FCE 管脚寄存器。 写该比特位将决定 FCE 管脚的输出数据。

27.4.3. NFM 等待寄存器(FMC_NFMWST: 04h)

位域	名称	属性	复位值	描述
31:27	TADL	RW	0	ALE to data out start CLK Cycle，tADL 为设置值加 1
26:21	TRHW	RW	0	REN High to WEN Low CLK Cycle，tRHW 为设置值加 1

20:16	TWHR	RW	0	WEN High to REN Low CLK Cycle，tWHR 为设置值加 1
15:12	TREH	RW	0	REN High hold time CLK Cycle，tREH 为设置值加 1
11:8	TRP	RW	0	REN Pulse width CLK Cycle，tRP 为设置值加 1
7:4	TWH	RW	0	WEN High hold time CLK Cycle，tWH 为设置值加 1
3:0	TWP	RW	0	WEN pulse width CLK Cycle，tWP 为设置值加 1

27.4.4. NFM 状态寄存器(FMC_NFMSTATUS: 08h)

位域	名称	属性	复位值	描述
31:1	RSV	-	-	保留
7:4	POS_RBN	RO	0	RBN 管脚上升沿状态。 1：发生上升沿事件，写 1 清除该标志位。 0：未发生。
3:0	RBN	RO	0	RBN 管脚状态位，读此位反应 RBN 管脚的状态。 0 = NAND flash 存储器忙 1 = NAND flash 存储器准备好

27.4.5. BCH 配置寄存器(FMC_BCH_CONFIG: 10h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	BCH_TYPE	RW	0	BCH 类型选择位。 000：512 字节数据+3 字节信息位+13 字节 ECC 校验位，可纠错 8 比特。 其它：保留位。

27.4.6. BCH 控制寄存器(FMC_BCH_CTRL: 14h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
5	AUTO_CORRECT	RW	0	自动纠错使能位。当使能自动纠错功能时，将自动启用流水线模式。手动模式时，不会启用流水线模式。 1：自动纠错模式。 0：手动纠错模式。
4	DECODE_INT_ENABLE	RW	0	译码中断使能。 1：使能译码完成中断。 0：禁止译码完成中断。

3	RESET_CHANNEL	RW	0	复位 ECC 数据通道。 向 RESET_HAND 位写 1，将复位 ECC 数据通道。读该比特总是返回 0。 1：复位 ECC 数据通道。 0：无效。
2	RSV	-	-	保留
1	IGALL1	RW	0	忽略全 1 译码使能。IGALL1 比特位决定当数据全是“FF”时，ECC 模块译码的结果。IGALL1 设置为 1 时，译码错误标志位将不会置位。 1：当数据全是“FF”时，忽略 ECC 模块译码的结果。 0：当数据全是“FF”时，不忽然模块译码的结果。
0	MODE	RW	0	BCH 模式控制位。 1：BCH 译码。 0：BCH 编码。

27.4.7. BCH 状态寄存器(FMC_BCH_STATUS: 18h)

位域	名称	属性	复位值	描述
31:24	CEN	RO	0	校验码错误比特数。仅在 DECODE_DONE 标志位置位时，该比特数据有效。
23:16	DEN	RO	0	数据错误比特数。仅在 DECODE_DONE 标志位置位时，该比特数据有效。
15:8	ERN	RO	0	错误比特总数。仅在 DECODE_DONE 标志位置位时，该比特数据有效。
7	RSV	-	-	保留
6	DATA_ALL0	RO	0	数据全 0 标志位。译码时读出的数据全 0 时，该比特位置位。该标志位置位后，下次译码完成后会自动清零。 1：数据全 0。 0：数据不是全 0。
5	DATA_ALL1	RO	0	数据全 1 标志位。译码时读出的数据全 1 时，该比特位置位。该标志位置位后，下次译码完成后会自动清零。 1：数据全 1。 0：数据不是全 1.
4	BCH_FAIL	RO	0	BCH 错误标志位。在译码阶段，当发生超过 BCH 纠错的时候，该比特置位。 1：BCH 错误。写 1 清除该标志位。 0：BCH 没有错误。
3	CORRECT_DONE	RO	0	译码阶段纠错完成。 1：纠错完成，写 1 清除该标志位。 0：纠错没有完成。
2	BM_DONE	RO	0	译码阶段错误多项式求解完成。 1：错误多译式求解完成。写 1 清除该标志位。 0：错误多项式求解没有完成。

1	SNYDROME_DONE	RO	0	译码阶段半随机式求解完成。 1：半随式求解完成，写 1 清除该标志位。 0：半随式求解没有完成。
0	ENCODE_CLR	WC	0	编码器可以即时地对数据进行编码，输入数据结束后，编码立即完成，不需要判断状态。向该位写 1 可以清除校验码寄存器，写 0 无效。 开始新的编码前需要向该位写 1。读该比特总是返回 0。

27.4.8. BCH 校验码寄存器(FMC_BCH_CODE: 1Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留位。
7:0	BCH_CODE	RO	0	BCH 校验码，FIFO 接口。

27.4.9. BCH 错误地址寄存器(FMC_BCH_ERRADR: 20h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留位。
7:0	BCH_ERRADR	RO	0	BCH 错误地址，FIFO 接口。

27.4.10. BCH 错误向量寄存器(FMC_BCH_ERRVEC: 24h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留位。
7:0	BCH_ERRVEC	RO	0	BCH 错误向量，FIFO 接口。

27.4.11. BCH 纠错基地址寄存器(FMC_BCH_BASEADDR: 28h)

位域	名称	属性	复位值	描述
31:0	BCH_BASEADDR0	RW	0	SRAM 基地址 0

注：BCH 模块只能访问 128KB 的 SRAM 即 0x2003_8000——0x2005_7FFF

27.4.12. BCH 校验码指针寄存器(FMC_BCH_CODEPTR: 2Ch)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留位
3:0	CODE_PTR	RW	0	BCH 校验位指针寄存器。

27.4.13. BCH 错误地址指针寄存器(FMC_BCH_ERRADDRPTR: 30h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留位
5:0	ADDR_PTR	RW	0	BCH 错误地址指针寄存器。（最大写入值为 59）

27.4.14. BCH 错误向量指针寄存器(FMC_BCH_ERRVECPTR: 34h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留位
5:0	VECTOR_PTR	RW	0	BCH 错误向量指针寄存器。（最大写入值为 59）

27.4.15. BCH 页记数寄存器(FMC_BCH_PAGENUM: 38h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留位
7:0	PAGE_NUM	RW	0	页记数寄存器。

27.4.16. BCH 地址锁存寄存器(FMC_BCH_ADDRLATCH: 3Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留位
7:0	BCH_ADDRLATCH	RO	0	BCH 纠错地址锁存寄存器。在自动纠错功能打开模式下，开始计算 SIBM 时锁存住该页的基地址，当纠错完成后，自动清零。

27.4.17. NFM 命令通道寄存器(FMC_NFMCMD: 40h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7: 0	NFMCMD	WO	0	NFM 命令通道寄存器

27.4.18. NFM 地址通道寄存器(FMC_NFMADDR: 44h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留

7: 0	NFMADDR	WO	0	NFM 地址通道寄存器

27.4.19. NFM ECC 数据通道寄存器(FMC_NFMECCDATA: 48h)

位域	名称	属性	复位值	描述
31: 0	NFMECCDATA	RW	0	NFM ECC 数据通道寄存器 此寄存器可以按照字、半字和字节访问，具体选择参见 7.1

27.4.20. NFM 非 ECC 数据通道寄存器(FMC_NFMNECCDATA: 4Ch)

位域	名称	属性	复位值	描述
31: 0	NFMNECCDATA	RW	0	NFM 非 ECC 数据通道寄存器 此寄存器可以按照字、半字和字节访问，具体选择参见 7.1

28. 安全数字输入/输出多媒体卡接口（SDMMC）

28.1. 概述

安全数字输入/输出多媒体卡接口(SDMMC)定义了 SD 卡、SD I/O 卡、多媒体卡(MMC) 和 CE-ATA 卡 Host Controller，提供 AHB 系统总线与 SD 存储卡、SD I/O 卡、MMC 和 CE-ATA 设备之间的数据传输。

所支持的 SD 存储卡和 SD I/O 卡系统规格书可以通过 SD 卡协会网站（www.sdcard.org）获取。

所支持的多媒体卡（MMC）系统规格书可以通过多媒体卡协会网站（www.jedec.org）获取，由 JEDEC 固态技术协会出版。

所支持的 CE-ATA 系统规格书可以通过 CE-ATA 工作组网站（www.ce-ata.org）获取。

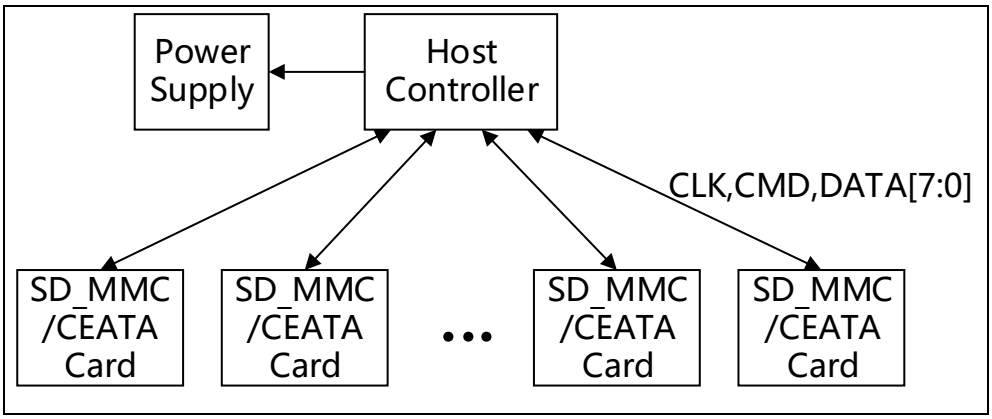
28.2. 主要特性

- MMC：完全兼容多媒体卡系统规范版本 5.0。卡支持三种不同数据总线模式： 1 位（默认）、 4 位和 8 位。完全兼容先前版本的多媒体卡(向后兼容性)
- SD 卡：完全兼容 SD 存储卡规范版本 3.0 和 3.01
- SDI/O：完全兼容 SDIO 卡规范版本 3.0。有两种不同的数据总线模式： 1 位(默认)和 4 位。
- CE-ATA：完全兼容 CE-ATA 数字协议版本 1.1
- 对于 8 位模式，数据传输高达 208 MB/s(取决于允许的最大 I/O 速度)
- 数据和命令输出使能信号，控制外部双向驱动程序
- 完成信号使能和失能(CE-ATA)
- 内置 FIFO，宽度为 32bit，深度为 16，支持当 over-run 和 under-run 时停止时钟
- 支持可编程的波特率。支持最多四种时钟分配率以满足在不同波特率下通信的要求
- 支持时钟控制开关
- 支持卡检测
- 支持卡写保护
- 支持 block size 从 1 到 65536
- 支持 DMA 传输

当前版本的 SDMMC 每次只支持一个 SD/SDIO/MMC4.41/CE-ATA 卡，但支持多个 MMC4.1 或之前版本的卡。

28.3. 总线拓扑

28.3.1. 拓扑框图



28.3.2. 总线描述

SDMMC 接口包含以下信号线：

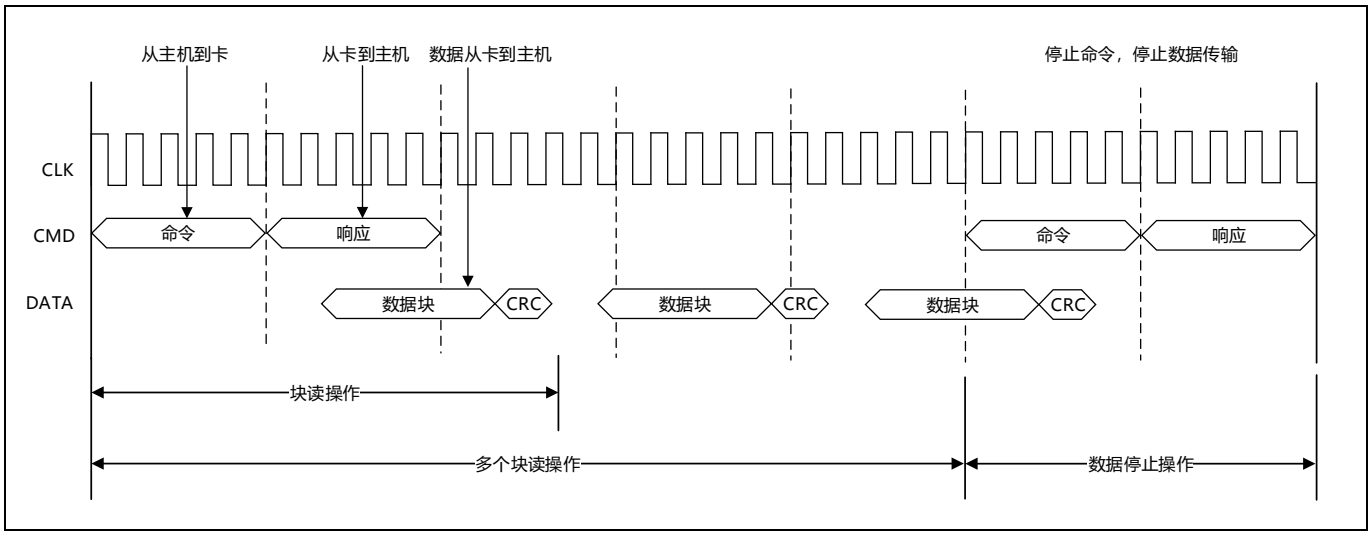
- CLK - 主机到卡的时钟信号
- CMD - 双向命令和响应信号
- DATA - 双向数据信号
- VDD，VSS1，VSS2 - 电源和地

SD_MMC_CEATA 协议是基于命令和数据的比特流，这些比特流由一个开始位发起，终止于一个结束位。主机控制器提供了一个参考时钟，并且是唯一可以初始化传输事务的主机。

- 命令(Command) - 从主机发送到卡。在 CMD 线上串行发送的一个令牌，用于启动一个操作。
- 响应(Response) - 响应是一个令牌，它作为对先前接收命令的应答。从已寻址的卡发送到主机。响应在 CMD 线上以串行方式传输。并不是所有的命令都期望得到卡的响应。
- 数据(Data) - 传输的数据。可以从主机转移到卡上，反之亦然。数据在 DATA 线上以串行方式传输。并不是所有的命令都需要传输数据。

下图展示了一次典型的 Multi-Block 读操作。图中的时钟只是示意性的，并不表示在一次传输中实际有这么多个时钟。

图 28-1 Multi-Block 读操作



下图展示了一次典型的 Multi-Block 写操作。图中的时钟只是示意性的，并不表示在一次传输中实际有这么多个时钟。

图 28-2 Multi-Block 写操作

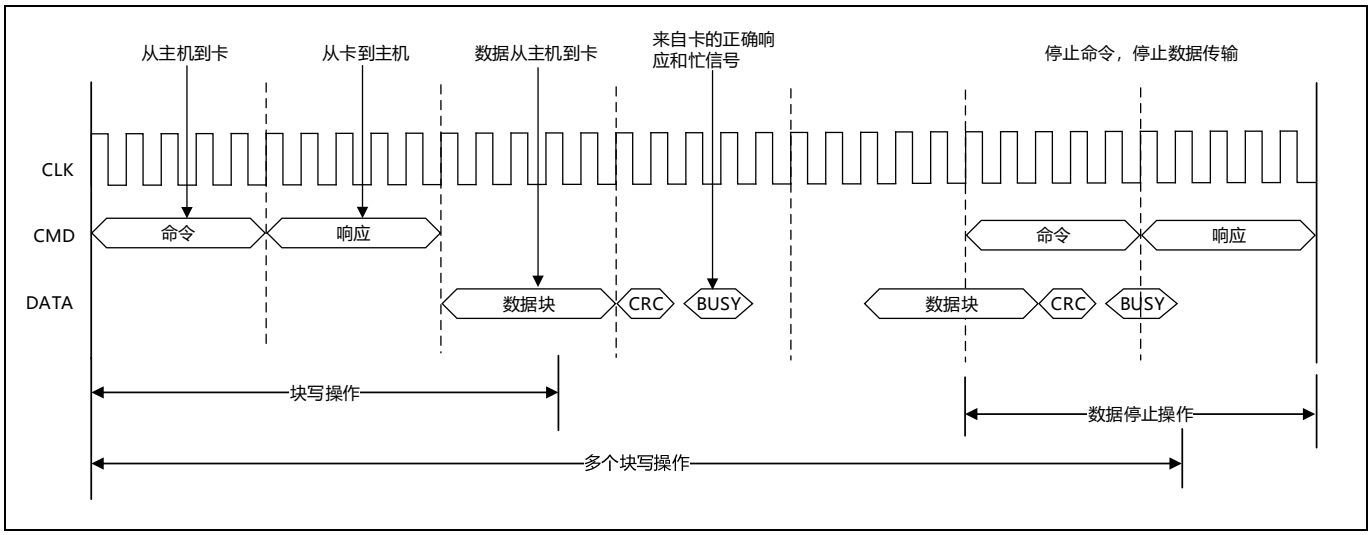


表 28-1 命令格式

位的位置	宽度	值	说明
47	1	0	起始位
46	1	1	传输位
[45:40]	6	-	命令索引
[39:8]	32	-	参数
[7:1]	7	-	CRC7
0	1	1	结束位

表 28-2 短响应格式

位的位置	宽度	值	说明

47	1	0	起始位
46	1	0	传输位
[45:40]	6	-	命令索引
[39:8]	32	-	参数
[7:1]	7	-	CRC7
0	1	1	结束位

表 28-3 长响应格式

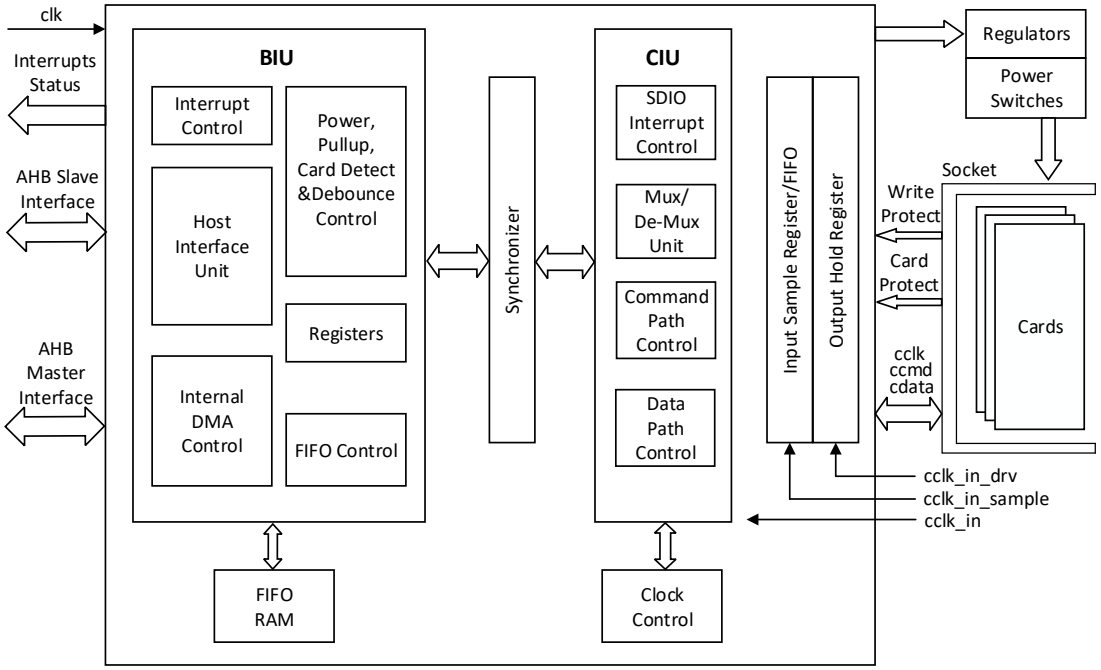
位的位置	宽度	值	说明
135	1	0	起始位
134	1	0	传输位
[133:128]	6	-	保留
[127:1]	127	-	CID 或 CSD
0	1	-	结束位

28.4. 功能描述

28.4.1. 结构框图

SDMMC 由以下主要模块组成。

- Bus Interface Unit (BIU) – 提供 AMBA AHB 和 DMA 接口，用于寄存器和数据读写。
- Card Interface Unit (CIU) – 负责 SD_MMC_CEATA 协议，并提供时钟管理。



28.4.2. 总线接口单元

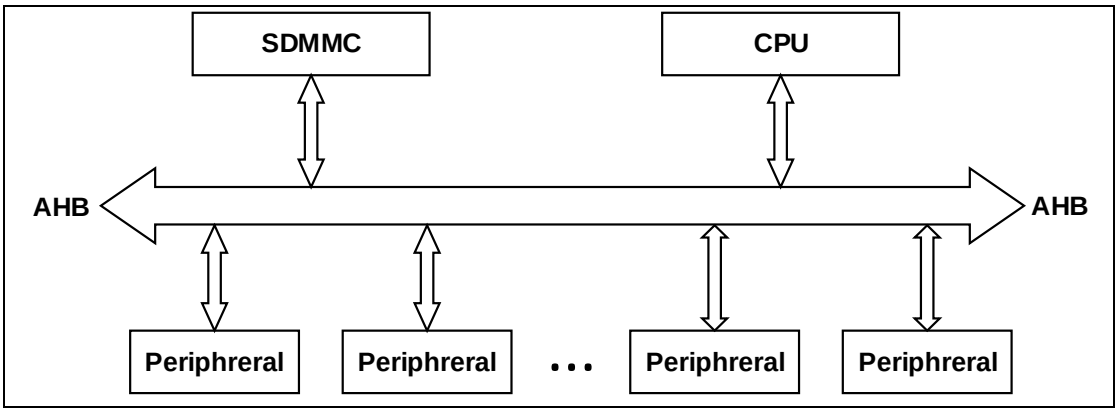
总线接口单元(BIU)提供以下功能：

- 主机接口(Host Interface)
- DMA 接口(DMA Interface)
- 中断控制(Interrupt Control)
- 寄存器访问(Register Access)
- FIFO 访问(FIFO Access)
- 电源/上拉控制和卡检测(Power/Pullup Control and Card Detection)

28.4.2.1. 主机接口(Host Interface)

HIU(Host Interface Unit)是一个 AHB 从接口，提供了 SDMMC 和 CPU 总线之间的接口。

图 28-3 SDMMC 在 AHB 系统



28.4.2.2. DMA 接口(DMA Interface)

SDMMC 的 DMA 接口为 IDMA(Internal Direct Memory Access)。是一个内部的 DMA 接口。

28.4.2.3. 中断控制(Interrupt Control)

中断控制单元产生一个依赖于原始中断状态、中断屏蔽寄存器和全局中断使能位的中断。一旦检测到中断条件，它就在原始中断状态寄存器中设置相应的中断位。原始中断状态位一直保持，直到软件通过向原始中断状态寄存器的对应位写入 1 来清除该位。

中断端口是高有效，电平敏感的。只有当原始中断状态寄存器中的任何位处于活动状态，相应的中断掩码位为 1，全局中断使能位为 1。中断端口才处于活动状态。

上电时，全局中断使能位被重置为 0，中断屏蔽寄存器为全 0。

表 28-4 中断状态

位	中断	描述
17:16	SDIO Interrupt	来自于 SDIO 卡的中断。Bit[17]对应于卡[1]，Bit[16]对应于卡[0]
15	End Bit Error (read)/Write no CRC(EBE)	在读操作中，结束位错误。或者是在写操作中，没有数据 CRC

14	Auto Command Done(ACD)	Stop/Abort 命令由卡单元自动发送，而不是由主机发起 建议：在此中断后的数据传输完成中断是否产生取决于数据传输是否正确完成。软件通常不需要为非 CE-ATA 访问启用该功能。对于 CE-ATA 访问，如果软件在控制寄存器中设置了 send_auto_stop_ccsd 位，那么软件应该启用该位
13	Start Bit Error(SBE)/Busy Clear Interrupt(BCI)	从卡读数据时，数据起始位错误。在 4 位模式下，如果 DAT[0]表示起始位，即为 0。其他任何数据位不为 0，则此错误被置位。 当数据写入卡时，忙清除中断。这个中断是在最后一个数据块写入卡后，卡驱动的忙绿完成后产生的
12	Hardware Locked write Error(HLE)	在硬件锁期间，试图写入一个被锁的寄存器
11	FIFO Underrun/Overrun Error(FRUN)	FIFO 满了，但是 Host 要继续往 FIFO 里搬数据；或者 FIFO 空了，但是 Host 要继续从 FIFO 中取数据
10	Data Starvation by Host Timeout(HTO)	如果 FIFO 满了，还要继续从卡读数据；或者 FIFO 空了，还要继续向卡写数据，那此时就会停掉输出时钟，以避免丢失数据。在超时寄存器输出时钟时间内还没有解决问题，中断就会产生
9	Data Read Timeout(DRTO)/ Boot Data Start(BDS)	在正常功能模式下：读数据超时 读数据超时时此中断产生。读数据超时时，数据传输完成中断也会产生。 在启动模式下：收到启动确认 当该中断产生时，表明 SDMMC 已经开始接收来自于卡的启动数据。对该寄存器写 1 清除中断。
8	Response Timeout(RTO)/ Boot Ack Received(BAR)	在正常功能模式下：响应超时 响应超时时此中断产生。如果发生了响应超时，命令结束中断也会产生。如果命令涉及数据传输，当响应超时时，SDMMC 不会尝试数据传输 在启动模式下：收到启动确认 当设置了 expect_boot_ack 时，在接收到启动确认确认时，产生中断。对该寄存器写 1 清除中断。
7	Data CRC Error(DCRC)	从卡收到的数据 CRC 与 CIU 本地产生的 CRC 不匹配。如果写 CRC 状态被 Host 不正确的采样也会产生该中断。
6	Response CRC Error(RCRC)	从卡收到的响应 CRC 与内部 CIU 本地产生的 CRC 不匹配
5	Receive FIFO Data Request(RXDR)	从卡读操作中，FIFO count 大于等于 RX_Level 建议：在 DMA 模式中，这一位的使能不要打开
4	Transmit FIFO Data Request(TXDR)	往卡写操作中，FIFO count 小于等于 TX_Level 建议：在 DMA 模式中，这一位的使能不要打开
3	Data Transfer Over(DTO)	表明数据传输完成。数据传输完成后，即使有起始位错误或者 CRC 错误，中断也会产生。 建议：在非 DMA 模式下，当数据传输完成后，Host 应该去把在 FIFO 剩下的数据搬出去；在 DMA 模式下，把剩余数据搬出去是 DMA 的责任
2	Command Done(CD)	命令发送出去后，收到了卡的响应，即使响应错误或者 CRC 错误，这一位中断也会置位。当响应超时时或者 CCSD 发送给 CE-ATA 设备时，这一位也会置位
1	Response Error(RE)	如果发生下列情况之一，则接收到的响应出现错误 响应的起始位不为 0 命令索引不匹配 结束位不为 1

0	Card-Detect(CDT)	当有一张或者卡插入或者拔出时，该中断产生。软件需要读取卡检测寄存器(CDETECT，0x50)来确定当前卡的状态。

28.4.2.4. 寄存器访问(Register Access)

寄存器访问单元是总线接口单元(BIU)的一部分，它提供对寄存器的读写访问。所有寄存器都位于 BIU 时钟域中。当通过设置 start_bit (CMD 寄存器的位[31])将命令发送到卡片时，CIU 操作所需的所有相关寄存器都被转移到 CIU 模块。在这段时间内，从 BIU 转移到 CIU 的寄存器不应该被写入。再次写入这些寄存器之前，软件应该等待硬件清除开始位。

寄存器单元具有硬件锁定特性，以防止对寄存器的非法写入。锁是必要的，因为主机和卡时钟域是不同的，这样可以避免产生亚稳态，锁还可以防止非法软件操作。

一旦通过设置 CMD 寄存器的 start_bit 发出命令启动，以下寄存器不能被重新编程，直到该命令被卡接口单元(CIU)接受：

- CMD - Command
- CMDARG - Command Argument
- BYTCNT - Byte Count
- BLKSIZ - Block Size
- CLKDIV - Clock Divider
- CLKENA - Clock Enable
- CLKSRC - Clock Source
- TMOUT - Timeout
- CTYPE - Card Type

一旦 CIU 接受命令，硬件将重置 start_bit。如果主机在这个锁定时间内试图写入这些寄存器中的任何一个，那么写入将被忽略，硬件锁错误位将被置位在原始中断状态寄存器中。此外，如果中断被启用，并且硬件锁错误中断没有被屏蔽，那么此中断将被发送到主机。

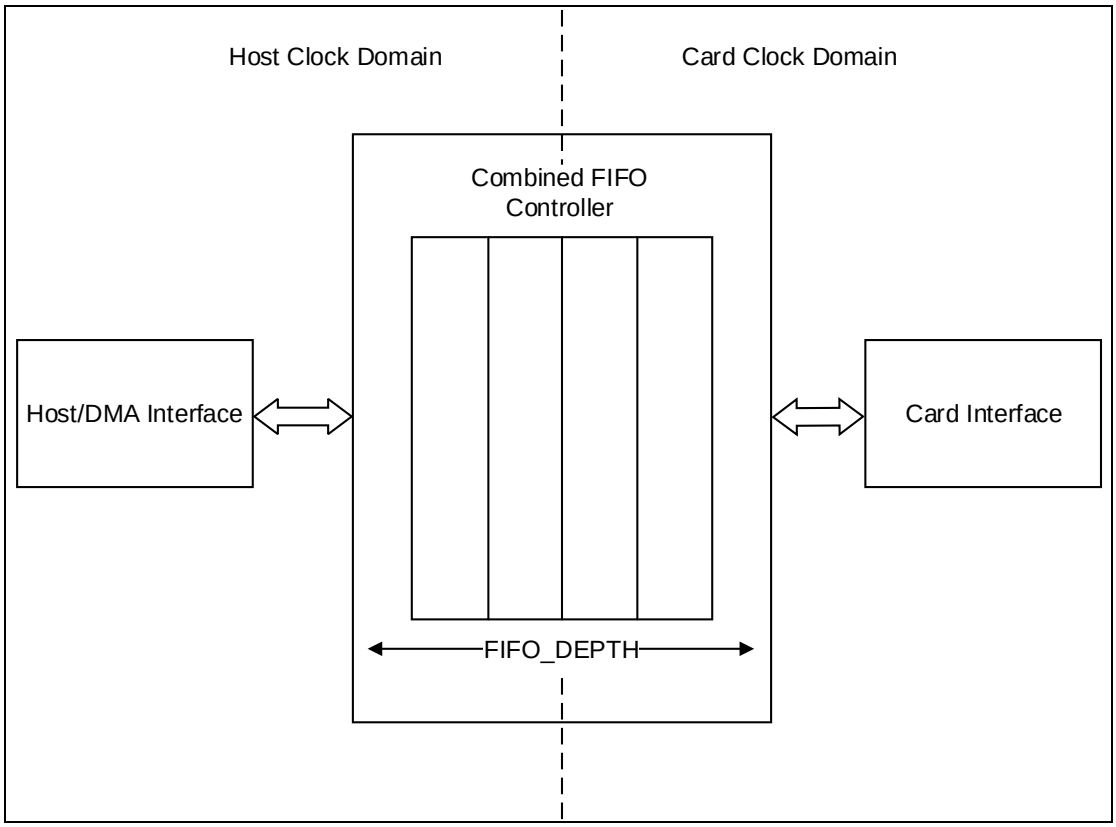
28.4.2.5. FIFO 访问(FIFO Access)

FIFO 控制器将内部 FIFO 接口到主机/DMA 接口和卡控制器单元。

由于对卡的写和读传输不会同时发生，为了节省面积，读写操作使用一个共享的 FIFO。在 cmd_taken 握手期间，对命令寄存器的读/写位[10]进行采样并存储。这个内部位定义了 SDMMC 是处于读模式还是写模式。

FIFO 使用双时钟同步读和同步写双端口 RAM。其中一个端口连接到主机时钟 clk。第二个端口连接到卡时钟 cclk_in。

图 28-4 发送和接收 FIFO



28.4.2.6. 电源/上拉控制和卡检测(Power/Pullup Control and Card Detection)

该控制单元控制电源和 MMC 开路漏极上拉。每张卡的电源可以有选择地打开或关闭。此外，还有两个 4bit 卡电压控制信号，可以控制两个稳压器的电压。

控制寄存器有一个 enable_OD_pullup 位，用于在 MMC 初始化期间打开开路漏极模式的上拉。此外，SDMMC 提供了一个 8 位通用输入端口 gp_in 和一个 16 位通用输出端口 gp_out。

卡片检测单元在卡片检测信号中寻找卡片插入或卡片移除的任何变化。它过滤掉与机械插入或移除相关的抖动，并对主机产生一个中断。我们可以编程 debounce 过滤器的值.

上电时，控制器应读入 card_detect 端口并将值存储在内存中。在接收到卡检测中断时，它应该再次读取 card_detect 端口。并且前一个卡检测状态 XOR，以找出哪张卡已经中断。如果同时拔出或插入多个卡，则只有一个卡检测中断。异或值表示哪些卡片被插拔。内存应该用新的卡片检测值进行更新。

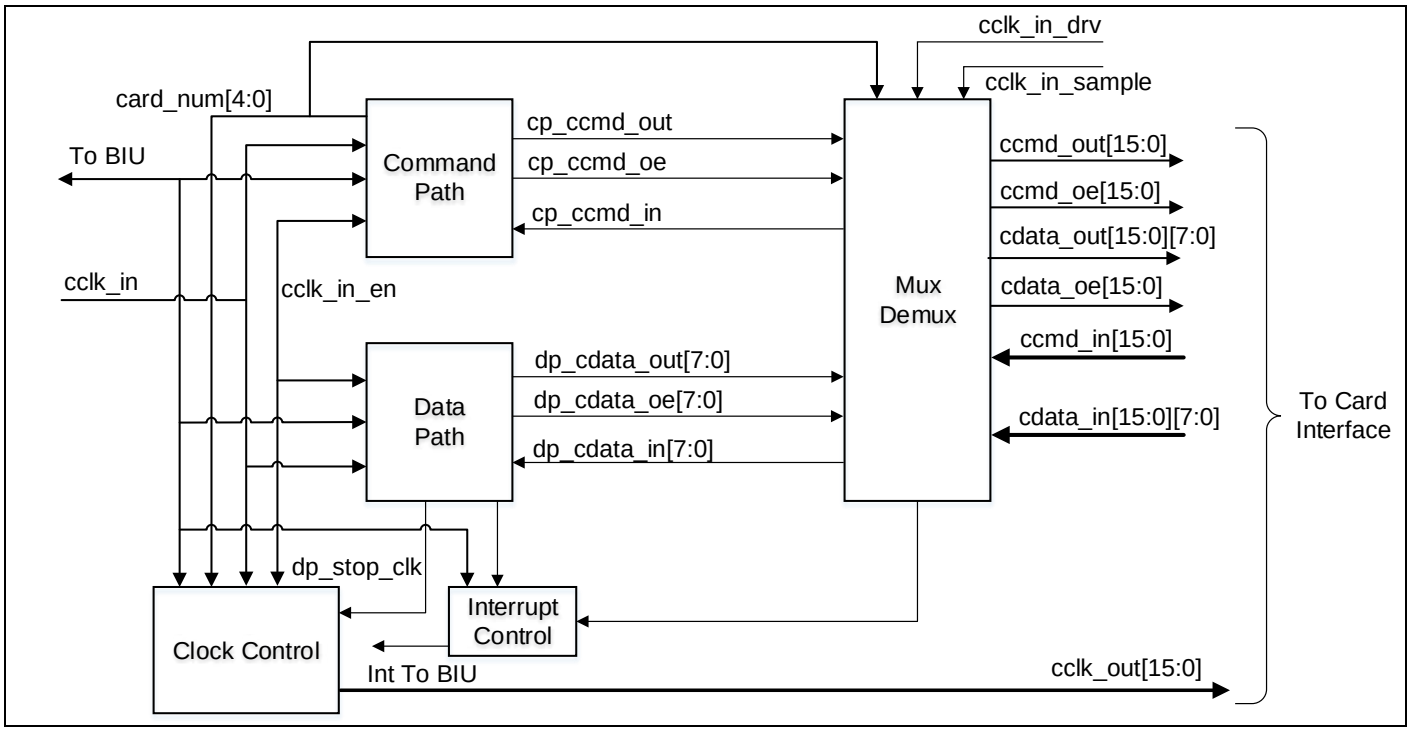
28.4.3. 卡接口单元

卡接口单元 CIU (Card Interface Unit) 连接 BIU (Bus Interface Unit)和 SD_MMC_CEATA 卡设备。主机将命令参数写入 BIU 控制寄存器，然后这些参数将传递给 CIU。CIU 根据 SD_MMC_CEATA 协议在选定的卡总线上生成 SD_MMC_CEATA 命令和数据传输。控制寄存器值还决定命令和数据传输是否以 CE-ATA 设备为目标。SDMMC 相应地控制命令和数据路径。

CIU 由以下主要功能块组成：

- 命令路径(Command Path)
- 数据路径(Data Path)
- SDIO 中断控制(SDIO Interrupt Control)
- 时钟控制(Clock Control)
- Mux/Demux Unit

- 错误检测(Error Detection)



28.4.3.1. 命令路径(Command Path)

命令路径的功能如下：

- 加载时钟参数(Load clock parameters)
- 加载卡命令参数(Load card command parameters)
- 发送命令到卡总线(Send command to card bus – ccmd_out line)
- 接收来自卡总线的响应(Receive response from card bus – ccmd_in line)
- 向 BIU 发送响应(Send response to BIU)
- 在命令线上驱动 P 位(Drive the P-bit on command line)

1. 加载卡命令参数(Load Command Parameters)

命令路径中加载了以下命令或响应:

- 来自 BIU 的新命令 - 当使用 start_cmd 时，则在命令寄存器中设置 start_cmd 位。
- 内部生成自动停机命令 - 当数据路径结束时，将加载停止命令请求。
- 带相对地址的中断请求响应 - 当命令路径正在等待来自 MMC 卡的 IRQ 响应，并且 BIU 发出“发送 IRQ 响应”请求时，则在控制寄存器中设置 send_irq_response 位。

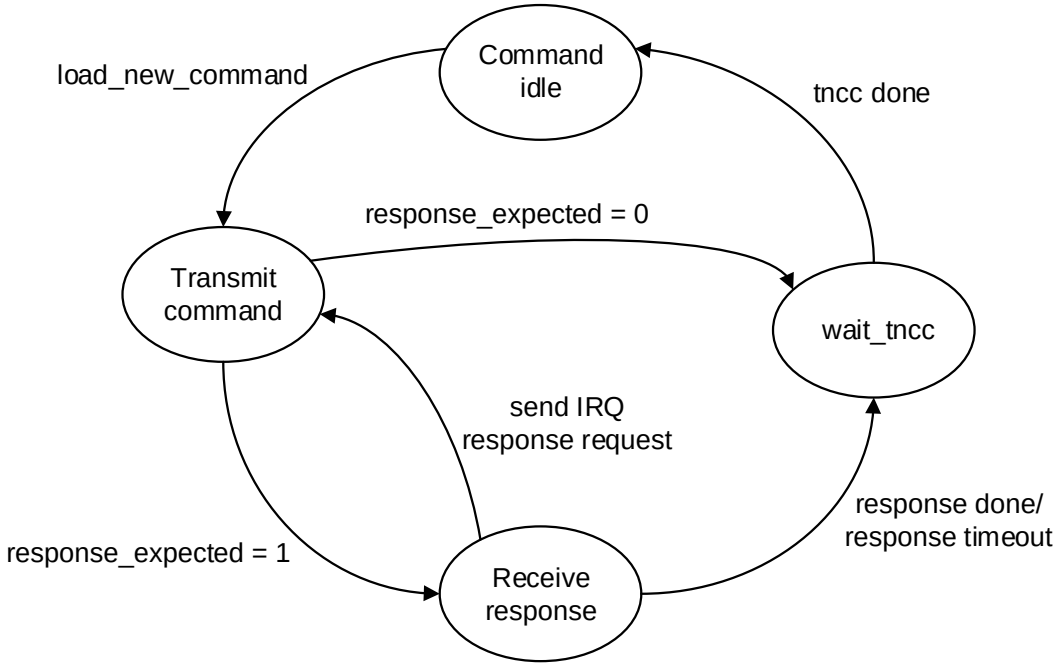
从 BIU 在命令路径中加载新命令依赖于以下命令寄存器位设置：

- update_clock_registers_only - 如果在命令寄存器中设置了该位，则命令路径只更新时钟使能寄存器、时钟频器和时钟源寄存器。如果没有设置该位，则命令路径加载命令、命令参数和超时寄存器，然后开始处理新命令。
- wait_prvdata_complete - 如果设置了该位，命令路径将在以下条件之一加载新命令:

- 如果数据路径是空闲的(也就是说，没有正在进行的数据传输)，或者正在进行开放式的数据传输(byte_count = 0)
- 当前数据传输完成后，如果正在进行预定义的数据传输

2. 发送命令和接收响应(Send Command and Receive Response)

一旦在命令路径中加载了一个新命令，则 update_clock_registers_only 位被取消设置，命令路径状态机就会在 SD_MMC 总线上发送一个命令。SD_MMC 命令路径状态机如下图：



命令路径状态机根据命令寄存器位值执行以下功能：

- send_initialization - 在发送命令之前发送 80 个时钟的初始化序列
- response_expected - 命令期望响应。命令发出后，命令路径状态机接收到一个 48 位或 136 位的响应，并且发送给 BIU。如果卡响应的起始位在超时寄存器中编程的时钟数内没有收到，那么响应超时和命令完成位在原始中断状态寄存器中被设置成一个到 BIU 的信号。如果未设置预期响应位，则命令路径发送一个命令并向 BIU 发出响应信号，也就是说，命令完成位在原始中断状态寄存器中设置。
- response_length - 如果设置了该位，则接收到一个 136 位响应，如果不设置，则收到为 48 位响应。
- check_response_crc - 如果设置了该位，命令路径将比较响应中收到的 CRC7 和内部生成的 CRC7。如果两者不匹配，则向 BIU 发送响应 CRC 错误信号，也就是说，响应 CRC 错误位在原始中断状态寄存器中设置。

3. 发送响应给 BIU(Send Response to BIU)

如果在命令寄存器中设置了 response_expected 位，则接收到的响应被发送到 BIU。在短响应时更新 Response0 寄存器，在长响应时更新 Response3、Response2、Response1 和 Response0 寄存器，在此之后设置命令完成位。如果响应是由 CIU 发送的 auto_stop 命令，则响应保存在 Response1 寄存器中，在 Response1 寄存器之后设置自动命令完成位。

此外，命令路径检查以下内容：

- 传输位= 0
- 命令索引匹配发送命令的命令索引
- 接收卡响应的结束位= 1

136 位响应或者 check_response_crc 位未设置的情况，是不检查命令索引的。对于 136 位响应和保留 CRC 48 位响应，命令索引保留为 111111。

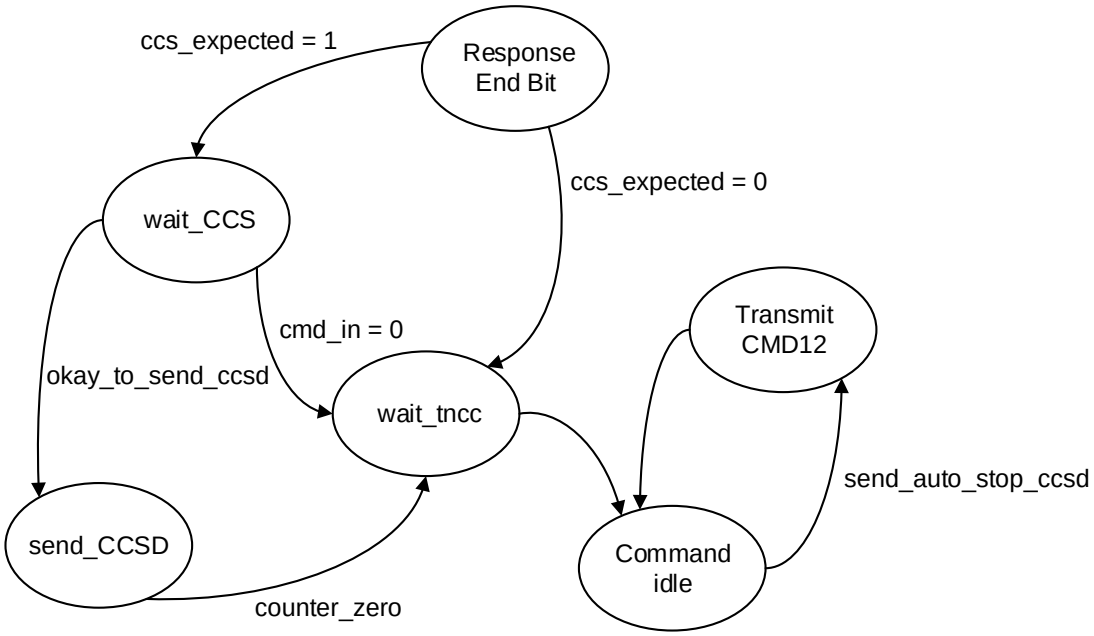
4. 在命令线上驱动 P 位(Drive the P-bit on Command Line)

如果没有预期的响应，命令路径在 CMD 线上两个命令之间驱动 P 位= 1。如果预期有响应，则在收到响应之后和下一个命令开始之前驱动 P 位;这可以通过设置 ccmd_out 和 ccmd_out_en 来完成。访问 CE-ATA 设备时，对于需要 Command Completion Signal 的命令，只有在 CE-ATA 设备中中断被禁用(nIEN=1)时，才会在响应后驱动 p 位;即清除 ccs_expected 位。如果命令需要 Command Completion Signal，那么只有在收到 Command Completion Signal 才驱动 P 位。

在初始化期间，软件应该设置 ccmd_od_pullup_en 位，它表示开漏模式，在此期间控制器只在命令总线上驱动 0 或高阻抗(Z)。硬件 1 永远不会在开漏模式下驱动。

5. 轮询命令完成信号(Polling Command Completion Signal)

CE-ATA 设备产生 Command Completion Signal，通知主控制器正常的 ATA 命令完成或 ATA 命令终止。在收到卡的响应后，命令状态机根据命令寄存器位值执行下图所示的功能。



以下描述了一些细节：

- Response_end_bit - 接收来自设备的响应结束位。如果设置了 “cmd_compl_expected bit” ，则进入 wait_CCS 状态。
- wait_CCS - FSM 等待 CE-ATA 设备的 CCS (Command Completion Signal)信号。在等待 CCS 的过程中可能会发生以下事件：
  - 软件设置 send_ccsd 位，表示不等待 Command Completion Signal，并在命令行上发送 Command Completion Signal Disable 模式
  - 在 CMD 线上接 Command Completion Signal
- send_ccsd - 在 CMD 线上发送 Command Completion Signal Disable 模式(5'b00001)

6. 命令完成信号侦测和中断(Command Completion Signal Detection and Interrupt)

如果在命令寄存器中设置了 ccs_expected 位，则通过在 RINTSTS 寄存器中设置 Data Transfer Over (DTO) 位来表示来自 CE-ATA 设备的命令完成信号(CCS)。如果这个中断没有被屏蔽，SDMMC 生成一个数据传输 (DTO) 中断。

对于 RW_MULTIPLE_BLOCK 命令，如果 CE-ATA 设备中断被关闭(nIEN=1)，即命令寄存器中的 ccs_expected 位被清除，则说明设备上没有任何 CCS。当数据传输结束时，即当传输了编程的字节数时，在

RINTSTS 寄存器中设置数据传输位

7. 命令完成信号超时(Command Completion Signal Timeout)

如果命令期望来自设备的 CCS 即如果命令寄存器中设置了 ccs_expected 位，命令状态机将等待 CCS 并保持 wait_CCS 状态。当 CE-ATA 设备发 CCS 失败时，主机软件需要建立一个超时机制，释放命令路径和数据路径。SDMMC 没有实现硬件定时器，维护软件定时器是主机软件的责任。

在 CCS 超时的情况下，主机应该通过在 CTRL 寄存器中设置 send_ccsd 位来发出一个 CCSD。SMMC 控制器命令状态机向 CE-ATA 设备发送 CCSD，并退出到空闲状态。在发送 CCSD 后，主机还应该向 CE-ATA 设备发送一个 CMD12，以中止未执行的 ATA 命令。

8. 命令完成信号禁用(Command Completion Signal Disable)

如果在 CTRL 寄存器中设置了 send_ccsd 位，则 SDMMC 在 CMD 线上发送一个命令完成信号禁用(CCSD)模式。主机可以在等待 CCS 或 CCS 超时后发送 CCSD。

在发送 CCSD 模式后，SDMMC 控制器在 RINTSTS 中设置命令完成(CD)位，并且如果命令完成中断没有被屏蔽，也会向主机生成一个中断。

如果在 CTRL 寄存器中设置了 send_auto_stop_ccsd 位，SDMMC 在发送 CCSD 模式后发送一个内部生成的停止命令(CMD12)。SDMMC 控制器在 RINTSTS 寄存器中设置自动命令完成位。

28.4.3.2. 数据路径(Data Path)

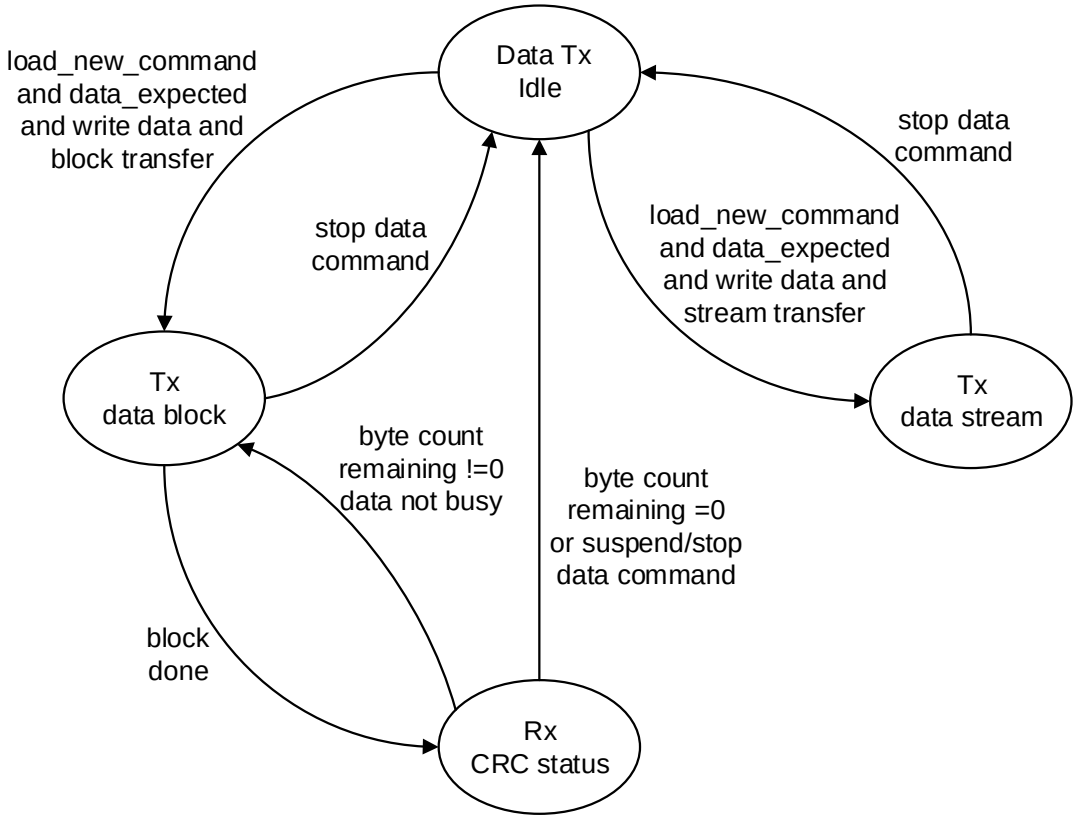
写数据传输时，数据路径块弹出数据 FIFO ，在 cdata_out 上传输数据;读数据传输时，数据路径块接收 cdata_in 上的数据，在 FIFO 上推入数据。当数据传输命令没有执行时，数据路径加载新的数据参数—即预期数据、读/写数据传输、流/块传输、块大小、字节数、卡类型、超时寄存器。

如果在命令寄存器中设置了 data_expected 位，则新命令是一个数据传输命令，数据路径将启动以下其中之一：

- 读/写位= 1 时数据发送
- 读/写位= 0 时数据接收

1. 数据发送

数据发送状态机，如图所示。



在接收到数据写命令的响应两个时钟后，启动数据传输。即使命令路径检测到响应错误或响应 CRC 错误，也会发生这种情况。如果由于响应超时而没有收到卡的响应，则数据不会被传输。根据命令寄存器中 TRANSFER_MODE 位的值，数据发送状态机将数据以流或块的形式放在卡数据总线上。

■ 数据流发送

如果在命令寄存器中的 TRANSFER_MODE 位设置为 1，它是一个流写入数据传输。数据路径将数据从 FIFO 弹出并以流的形式传输到卡数据总线。如果 FIFO 变为空，则卡时钟停止并在 FIFO 中数据可用时重新启动传输。

如果 BYTE_COUNT 寄存器编程为 0，则是一个开放的流写入数据传输。在此数据传输过程中，数据路径以流的形式持续传输数据，直到主机软件发出停止命令。当停止命令的结束位和数据的结束位在两个时钟上匹配时，流数据传输终止。

如果 BYTE_COUNT 寄存器编程非零值并且命令寄存器中 SEND_AUTO_STOP 被设置，当停止命令的结束位出现在流写传输的最后一个字节匹配之后时，停止命令在内部生成并加载到命令路径中。如果主机在所有数据字节被传输到卡总线之前发出停止命令，这个数据传输也可以终止。

■ 单个数据块发送

如果命令寄存器中的 TRANSFER_MODE 位设置为 0，并且 BYTE_COUNT 寄存器的值等于 BLOCK_SIZE 寄存器的值，则发生单块写数据传输。数据发送状态机在单个块中发送数据，其中字节数等于块大小，包括内部生成的 CRC16。

如果所选卡片 CTYPE 寄存器位设置为 1 位，4 位，或 8 位数据传输。数据通过 1，4 或 8 条数据线传输。CRC16 通过 1，4 或 8 条数据线单独生成和传输。

传输单个数据块后，数据发送状态机接收来自卡的 CRC 状态，并向 BIU 发出数据传输信号，当在 RINTSTS 寄存器中数据传输结束位置 1 时，就会发生这种情况。

如果从卡接收到 Negative CRC 状态，数据路径通过在 RINTSTS 寄存器中设置数据 CRC 错误位向 BIU 发出数据 CRC 错误信号。

此外，如果 CRC 状态的开始位在数据块结束后没有在两个时钟内被接收，则通过在 RINTSTS 寄存器中设置

write-no-CRC 位向 BIU 发出 CRC 状态错误开始位的信号。

■ 多个数据块发送

如果命令寄存器中的 TRANSFER_MODE 位设置为 0 并且 BYTE_COUNT 寄存器中的值不等于 BLOCK_SIZE 寄存器的值，则发生多块写数据传输。数据发送状态机以块的形式发送数据，其中块中的字节数等于块的大小，包括内部生成的 CRC16。

如果所选卡片 CTYPE 寄存器位设置为 1 位，4 位，或 8 位数据传输。数据通过 1，4 或 8 条数据线传输。CRC16 通过 1，4 或 8 条数据线单独生成并且传输。

传输一个数据块后，数据发送状态机接收卡的 CRC 状态。如果剩余的 BYTE_COUNT 变为 0，数据路径向 BIU 发送数据传输完成信号。当 RINTSTS 寄存器中数据传输位时置位时，就会发生这种情况。

如果剩余数据字节大于 0，则数据路径状态机开始传输另一个数据块。

如果从卡接收到 Negative CRC 状态，数据路径通过在 RINTSTS 寄存器中设置数据 CRC 错误位向 BIU 发出数据 CRC 错误信号，并继续数据传输，直到所有字节传输完毕。

此外，如果 CRC 状态的开始位在数据块结束后没有在两个时钟内被接收，则通过在 RINTSTS 寄存器中设置 write-no-CRC 位向 BIU 发出 CRC 状态错误开始位的信号

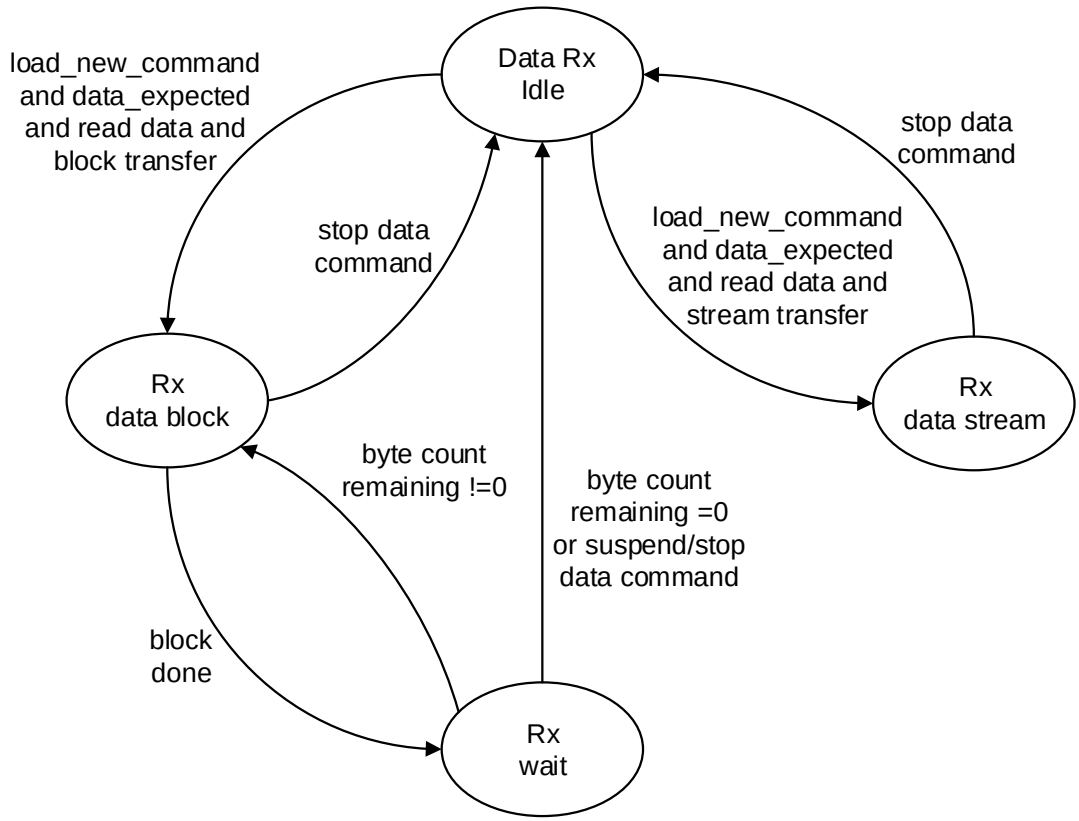
如果在命令寄存器中设置了 SEND_AUTO_STOP 位，则停止命令是在最后一个数据块的传输期间内部生成的，其中没有向卡传输额外的字节。STOP 命令的结束位可能与最后一个数据块中的 CRC 状态的结束位不完全匹配。

如果块大小小于 4,16 或 32（ 分别对应卡数据宽度 1 位，4 位，8 位)， 发送状态机终止数据传输当所有数据传输完成，在这段时间产生的内部停止命令并加载在命令路径。

如果 BYTE_COUNT 为 0，BLOCK SIZE 大于 0 ，则这是一个开放的块传输。用于这种类型的数据传输的数据发送状态机继续进行块写数据传输，直到主机软件发出停止或中止命令。

2. 数据接收

数据接收状态机，如图所示。



在数据读取命令的结束位两个时钟周期后接收数据，即使命令路径检测到响应错误或响应 CRC 错误。如果由于响应超时而没有收到卡的响应，则 BIU 没有收到数据传输完成的信号。如果 SDMMC 发送的命令是对卡的非法操作，就会发生这种情况，这会阻止卡开始读取数据传输。

■ 数据流接收

如果命令寄存器中的 TRANSFER_MODE 位等于 1，则发生流读数据传输，此时数据路径从卡接收数据并将其推送到 FIFO。如果 FIFO 已满，卡片时钟停止。一旦 FIFO 不再满，卡时钟重新启动。

如果 BYTE_COUNT 寄存器等于 0，则发生开放式流读数据传输。在这种类型的数据传输过程中，数据路径不断地接收数据流，直到主机软件发出停止命令。流数据传输在停止命令的结束位两个时钟周期后终止。

如果 BYTE_COUNT 寄存器是非零值并且 SEND_AUTO_STOP 在命令寄存器中被设置，停止命令在内部生成并加载到命令路径中，在该路径中，停止命令的结束位出现在接收到流数据传输的最后一个字节之后。如果主机在从卡接收到所有数据字节之前发出停止或中止命令，则此数据传输可以终止。

■ 单个数据块接收

如果命令寄存器中的 TRANSFER_MODE 位设置为 0，并且 BYTE_COUNT 寄存器的值等于 BLOCK_SIZE 寄存器的值，就会发生单块读数据传输。当在数据超时之前接收到一个起始位时，将接收块大小的数据和 CRC16，并使用内部生成的 CRC16 检查。

当 CRC16 不匹配时，数据路径向 BIU 发出数据 CRC 错误信号。如果接收到的结束位不是 1，则 BIU 收到结束位错误。

如果所选卡片 CTYPE 寄存器位设置为 1 位，4 位，或 8 位数据传输。数据通过 1，4 或 8 条数据线接收。CRC16 通过 1，4 或 8 条数据线分别生成和检查。

■ 多个数据块接收

如果命令寄存器中的 TRANSFER_MODE 位设置为 0，并且 BYTE_COUNT 寄存器的值不等于 BLOCK_SIZE 寄存器的值，则它是一个多块读取数据传输。数据接收状态机以块的形式接收数据，其中块中的字节数等于块的大小，包括内部生成的 CRC16。

如果所选卡片 CTYPE 寄存器位设置为 1 位，4 位，或 8 位数据传输。数据通过 1，4 或 8 条数据线接收。CRC16 通过 1，4 或 8 条数据线单独生成和检查。当接收到一个数据块后，如果剩余的 BYTE_COUNT 变为 0，则数据路径向 BIU 发送数据传输结束信号。

如果剩余的数据字节大于 0，则数据路径状态机将导致另一个数据块被接收。如果接收到的数据块的 CRC16 与内部生成的 CRC16 不匹配，则会向 BIU 发送数据 CRC 错误，数据接收会进行进一步的数据传输，直到所有字节都传输完。此外，如果接收的数据块的结束位不是 1，数据路径上的数据向 CIU 发出结束位错误信号，数据接收状态机终止数据接收，等待数据超时，并向 BIU 发送数据传输完成信号。

如果在命令寄存器中设置了 SEND_AUTO_STOP 位，则停止命令是在最后一个数据块传输时内部生成的，其中没有从卡传输额外的字节。停止命令的结束位可能与最后一个数据块的结束位不完全匹配。

如果对于 1 位、4 位或 8 位数据传输模式，向卡传输数据所需的块大小分别小于 4、16 或 32 字节，当传输所有数据时，数据接收状态机终止数据传输，此时将在命令路径中加载内部生成的停止命令，之后从卡接收的数据被数据路径忽略。

如果 BYTE_COUNT 为 0，则 BLOCK_SIZE 必须大于 0，这是一个开放式块传输。对于这种类型的数据传输，数据接收状态机继续块读取数据传输，直到主机软件发出停止或中止命令

3. 自动停止

SDMMC 内部生成一个停止命令，并在命令寄存器中设置了 SEND_AUTO_STOP 位时加载到命令路径中。auto stop 命令有助于通过 MMC 卡的流读或写和 SD 卡的多块读或写来传输准确的数据字节数。

以下列出了 auto stop 命令的条件：

- 字节数大于 0 的 MMC 卡的流读取
- 字节数大于 0 的 MMC 卡的流写入
- 字节数大于 0 的 SD 卡的多块读取
- 字节数大于 0 的 SD 卡的多块写入

每当发出自动停止命令时，主机软件不应该向 SDMMC 发出新命令，直到 SDMMC 发出自动停止命令并完成数据传输。

28.4.3.3. SDIO 中断控制(SDIO Interrupt Control)

通过置位中断信号两个时钟周期，向 BIU 报告 SD 卡的中断。SDIO 卡通过设置 cdata_in 低来发出中断信号。所选卡的中断周期由中断控制状态机决定。对于非活动或非选定的卡，中断周期总是有效的，对于选定的卡，1 位数据模式总是有效的。对于选定卡的中断期在以下条件下有效：

- 卡是空闲的
- 正在进行非数据传输命令
- 在两个数据块之间的数据块结束位之后的第 3 个时钟开始
- 从最后一个数据的结束位后的两个时钟，直到下一个数据传输命令的结束位

在以下情况下，当卡数据宽度为 4 位时，SDMMC 不会对所选卡的 SDIO 中断进行采样。由于 SDIO 中断是电平触发的，它在一个进一步的中断周期中采样时，主机不会从卡上丢失任何 SDIO 中断。

1）读/写 resume - CIU 将 resume 命令视为正常的数据传输命令。resume 命令期间的 SDIO 中断的处理与其他数据命令类似。根据 SDIO 规范，对于普通数据命令，中断周期在数据命令的命令结束位之后结束;对于 resume 命令，在响应结束位之后结束。对于 resume 命令，SDMMC 在 resume 命令结束位之后停止中断采样周期，而不是在 resume 命令的响应结束位之后停止。

2）读传输时挂起 - 如果主机挂起了读数据传输，则主机在 SDMMC 中设置 abort_read_data 位来复位数据状态机。在 CIU 中，对 SDIO 中断进行处理，以便于在主机设置了 abort_read_data 位之后开始中断采样。在这种情况下，SDMMC 在从 suspend 命令的响应到设置 abort_read_data 位之间不采样 SDIO 中断，并在设置 abort_read_data 位之后开始采样。

28.4.3.4. 时钟控制(Clock Control)

时钟控制块提供 SD_MMC_CEATA 卡所需的不同时钟频率。cclk_in 信号是时钟控制块的时钟分配器的源时钟(cclk_in >=卡的最大工作频率)。这个源时钟(cclk_in)用于产生不同的卡时钟频率。每个卡时钟可以有不同的时钟频率，因为 SD 卡可以是低速或全速 SD 卡，SDMMC 为每个卡提供一个时钟信号(cclk_out)，这允许每个卡以不同的时钟频率工作。

卡的时钟频率取决于以下时钟控制寄存器：

- 时钟分频寄存器 - 内部时钟分频器用于产生卡所需的不同时钟频率。一个配置参数决定了时钟分割器的数量(1-4)。每个时钟分频器的分频因子可以通过写入时钟分频器寄存器来编程。时钟分频寄存器是一个 8 位值，提供从 1 到 510 的时钟分频器因子。0 表示时钟分频器旁路，1 表示 2 分频，2 表示 4 分频，以此类推。
- 时钟源寄存器 - 通过对时钟源寄存器进行编程，从四个时钟分频器中选择一个时钟用于卡 cclk_out。
- 时钟控制寄存器 - 每个卡的 cclk_out 可以在以下条件下启用或禁用:
  - clk_enable - 如果时钟控制寄存器中卡的 clk_enable 位被编程(设置为 1)或禁用(设置为 0)，则卡的 clk_out 被启用或禁用。
  - Low-power mode - 通过将时钟控制寄存器的低功耗模式位设置为 1，可以使卡的低功耗模式生效。如果开启低功耗模式以节省卡电量，当卡空闲至少 8 个时钟周期时，禁用 cclk_out 信号。当加载新命令并且命令路径变为非空闲状态时，cclk_out 信号将启用。

此外，当内部 FIFO 是满的，读卡操作(无法从卡接收更多的数据)或者当内部 FIFO 是空的，写卡操作(没有数据可用来发送到卡)，所选卡的 cclk_out 是禁用的。这是为了避免产生 FIFO 的上溢和下溢。

28.4.3.5. Mux/Demux Unit

在 SD_MMC_CEATA 模式下，SDMMC 和每个卡之间运行着一个单独的总线。在这种模式下，当控制器发出命令或数据时，Demux 逻辑只向所选卡发送命令或数据， 未被选中的卡会忽略这些命令和数据。卡片数由命令寄存器中设置的 CARD_NUM 值选择。来自所选卡的响应或数据输入被多路复用并发送到 SDMMC。

cclk_in 信号在这个块中将所有输出注册到 SD_MMC_CEATA 卡。如果设置了 use_hold_reg 可编程位，那么所有输出都由 cclk_in_drv 时钟重新注册。cclk_in_drv 是 cclk_in 的延迟，用于保证 SDR12 和 SDR25 速度模式下的 SD_MMC_CEATA 卡的高保持时间要求。

对于 DDR50, SDR50 和 SDR104 速度模式，use_hold_reg 位可能被编程为 0 或 1。如果 cclk_in_drv 相移为 0，则 use_hold_reg 位不设置，SD_MMC_CEATA 卡的输出不被 cclk_in_drv 时钟注册。如果 cclk_in_drv 相移非 0，那么 use_hold_reg 位被设置，SD_MMC_CEATA 卡的输出被 cclk_in_drv 时钟注册。

为了保证卡在这些模式下工作时满足保持时间要求，需要在所有卡的输出- ccmd_out、ccmd_out_en、cdata_out 和 cdata_out_en 上添加缓冲或延迟线，以满足保持时间要求。

cclk_in_sample 信号对来自 SD_MMC_CEATA 卡的所有输入进行采样。cclk_in_sample 是 cclk_in 的延迟，与 cclk_out 相匹配，因此由 SD_MMC_CEATA 卡驱动的信号在采样时将满足 setup 和 hold 要求。

28.4.3.6. 错误检测(Error Detection)

1) 响应

- 响应超时 - 在超时寄存器中写入的那么多个时钟周期内，没有接收到应答的起始位。
- 响应 CRC 错误 – 响应的 CRC7 与内部产生的 CRC7 不匹配。
- 响应错误 - 应答的起始位不是 0，或者命令索引与发送的命令索引不匹配，或者应答结束位不是 1。

2) 数据发送

- 无 CRC 状态 - 在一次写数据传输时，在数据块的结束位被发出两个时钟周期后，没有收到 CRC 的起始位，数据路径做出如下操作：
  - 向 BIU 发送 CRC 状态错误信号
  - 终止进一步的数据传输
  - 向 BIU 发送输出传输结束信号
- Negative CRC - 如果写数据块后接收到的 CRC 状态为 Negative(即不 010)，向 BIU 发出数据 CRC 错误信号，并停止进一步的数据传输
- 由于 FIFO 空导致的数据缺失 - 如果在写数据传输期间 FIFO 变为空，或者如果卡时钟停止并且 FIFO 在数据超时的时钟下保持着空状态。则数据缺失错误信号会发送给 BIU。数据路径继续等待 FIFO 中的数据。

3) 数据接收

- 数据超时 - 在读取数据传输期间，如果数据开始位在超时寄存器中编程的时钟数之前未被接收，数据路径执行以下操作：
  - 向 BIU 发送数据超时错误
  - 终止进一步的数据传输
  - 向 BIU 发送输出传输结束信号
- 数据起始位错误 - 在 4 位或 8 位的读数据传输过程中，如果发生 SBE，驱动程序必须为 SD/MMC 卡发出 CMD12，为 SDIO 卡发出 CMD52，以便脱离错误状态。收到 CMD 完成后，驱动程序必须复位 IDMAC 和 CIU(如果需要)以清除错误。一般来说，FIFO 复位必须在发出任何数据传输命令之前完成。寄存器 CTRL

bit[2:0]控制这些复位：

- CTRL[2] = DMA_RESET
- CTRL[1] = FIFO_RESET
- CTRL[0] = CONTROLLER_RESET/CIU RESET

- 数据 CRC 错误 - 在读取数据块的传输过程中，如果接收到的 CRC16 与内部生成的 CRC16 不匹配，数据路径信号数据 CRC 错误到 BIU ，并停止进一步的数据传输。
- 数据结束位错误 - 在读数据传输过程中，如果接收到的数据结束位不为 1，则数据路径向 BIU 发出结束位错误信号，终止数据的继续传输，并向 BIU 发出数据传输完成的信号。
- 有 FIFO 满导致的数据缺失 - 在读取数据传输期间，当 FIFO 满时，卡时钟停止。如果 FIFO 在数据超时时钟里保持着满状态，数据缺失错误信号就会发向 BIU(RINTSTS 寄存器中主机数据缺失超时位会置位)，数据路径继续等待 FIFO 开始清空。

28.4.4. IDMAC

Internal Direct Memory Access Controller(IDMAC)有一个控制和状态寄存器(CSR)和一个传输/接收引擎，它将数据从主机内存传输到设备端口，反之亦然。控制器利用描述符以最小的主机 CPU 干预有效地将数据从源移动到目标。可以编程控制器中断主机 CPU 的情况：例如卡的数据发送和接收传输完成，以及其他正常或错误的条件。

IDMAC 和 Host 驱动通过一个数据结构进行通信。CSR 地址 0x80 到 0x98 为主机编程所保留。

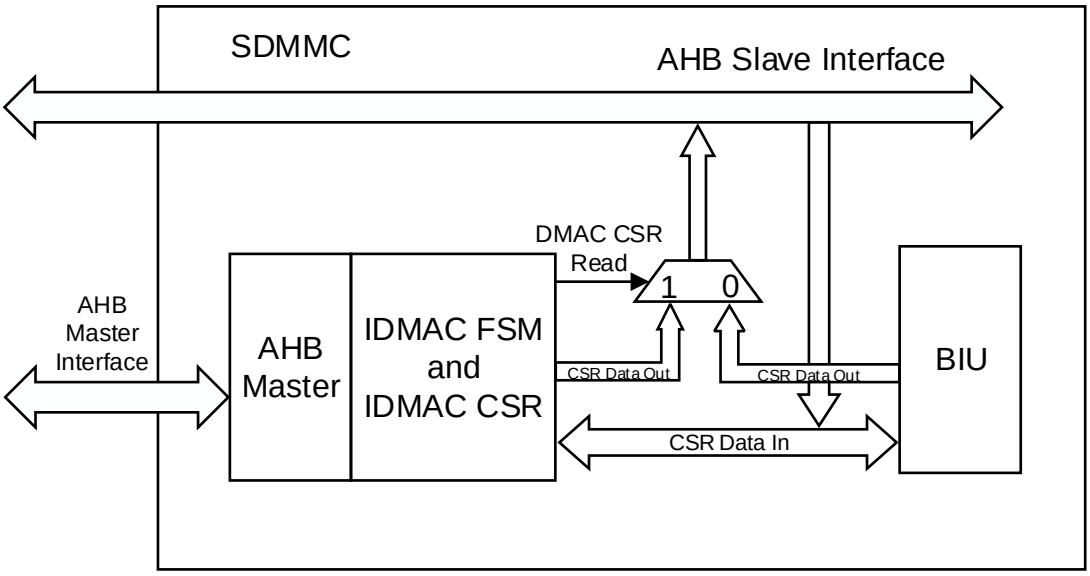
IDMAC 将从卡接收到的数据传输到主机内存中的数据缓冲区，并将主机内存中数据缓冲区的发送数据传输到 FIFO ，在主机内存中的描述符充当指向这些缓冲区的指针。

数据缓冲区驻留在主机的物理内存空间中，由完整数据或部分数据组成。缓冲区只包含数据，而缓冲区状态是在描述符中维护的。数据链接是指跨多个数据缓冲区的数据。但是，一个描述符不能跨多个数据。

接收和传输都使用单个描述符。列表的基址被写入描述符列表基址寄存器(DBADDR @0x88)。描述符列表是向前链接的。最后一个描述符可以指向第一个描述符以创建一个环结构。描述符列表驻留在主机的物理内存地址空间中。每个描述符最多可以指向两个数据缓冲区。

28.4.4.1. IDMAC 控制状态寄存器访问(IDMAC CSR Access)

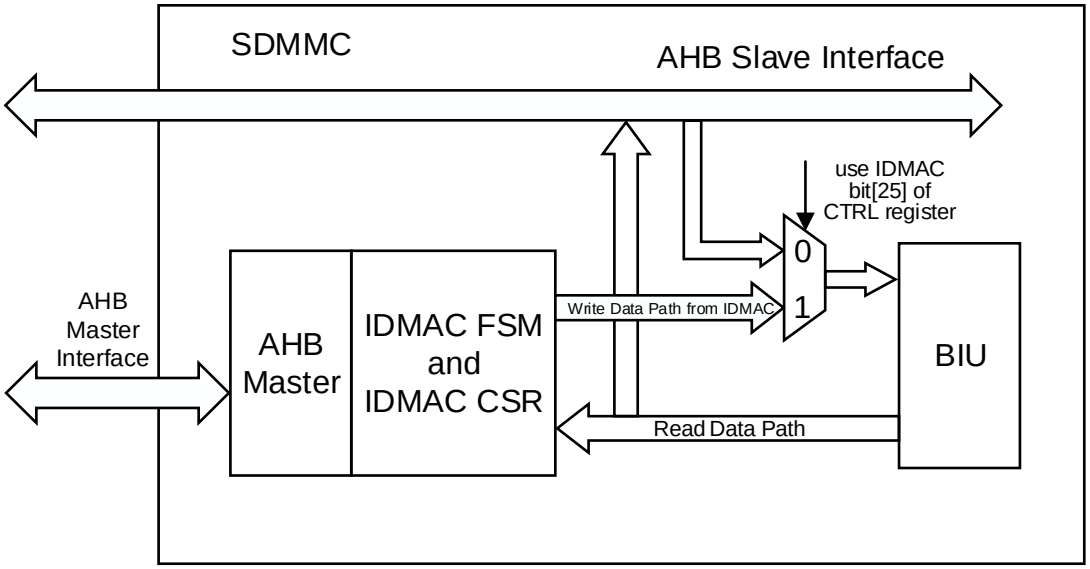
1. IDMAC 控制状态寄存器路径访问(IDMAC CSR Path Access)



当引入 IDMAC 时，一个额外的 CSR 空间驻留在 IDMAC 中，它控制 IDMAC 的功能。主机除了访问 BIU

中现有的控制寄存器，还访问新的 CSR 空间。IDMAC CSR 主要包含描述符信息。

2. IDMAC 数据路径访问(IDMAC Data Path Access)



通过在 BIU 的 CTRL 寄存器中编程 bit[25] ，可以使能或禁止 IDMAC 操作。如果 IDMAC 存在但被禁用，这允许数据传输通过访问 AMBA 总线上的从接口。当 IDMAC 使能后，不能通过 slave 接口访问 FIFO。

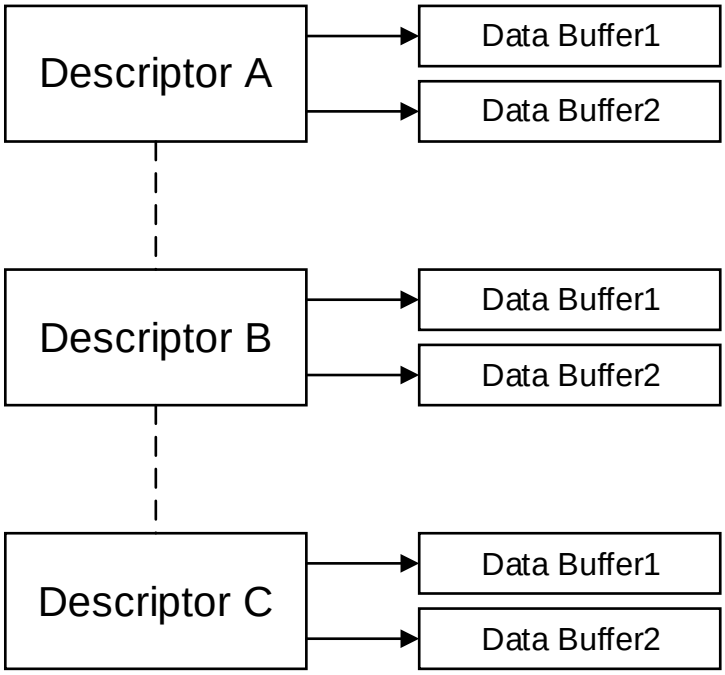
28.4.4.2. 描述符(Descriptors)

IDMAC 使用以下类型的描述符结构:

双缓冲区结构 - 两个描述符之间的距离由总线模式寄存器 (BMOD@0x80)的描述符跳过长度(DSL)字段中编程的跳过长度值决定。

链式结构 - 每个描述符指向一个唯一的缓冲区和下一个描述符。

图 28-5 双缓冲区描述符结构



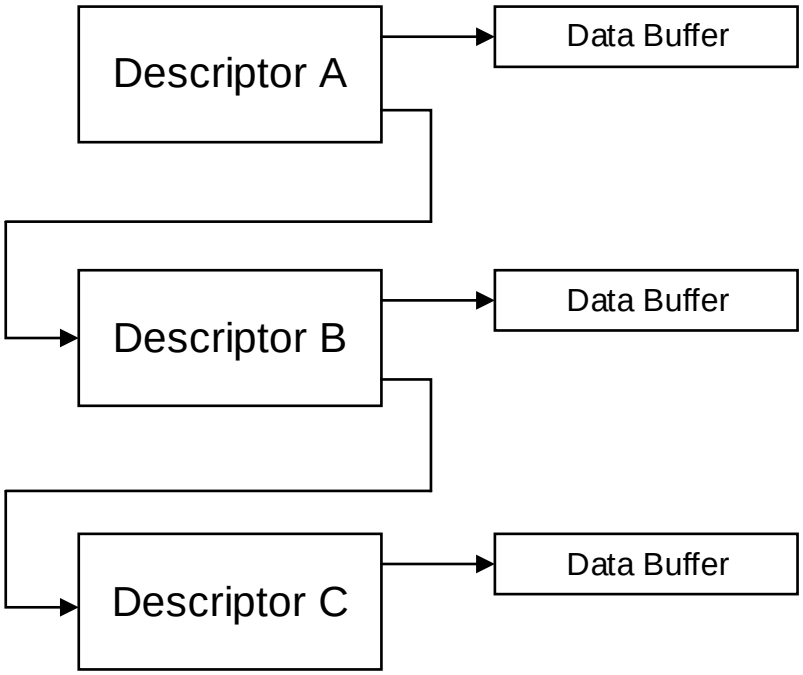
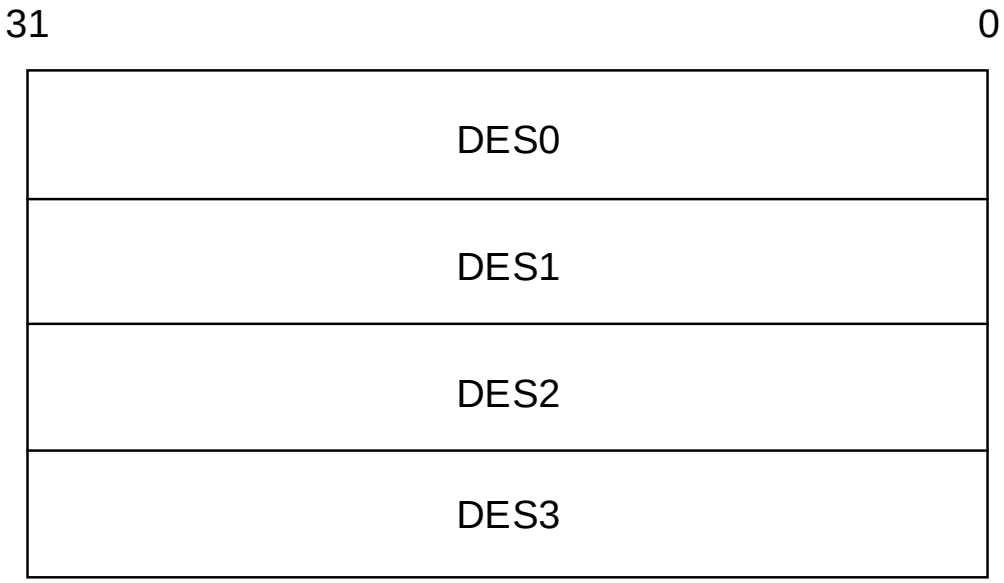


图 28-6 描述符格式



描述符的内部格式。描述符地址必须与用于 32 位 AHB 数据总线的总线宽度对齐。

DES0 包含控制和状态信息

表 28-5 DES0

位域	名称	描述
31	OWN	该位置 1 时，表示 IDMAC 拥有描述符。该位复位时，表示 Host 拥有描述符。IDMAC 在数据传输完成后清除此位。

30	CARD ERROR SUMMARY (CES)	这些错误位指示传输到卡或者来自于卡的状态。这些位存在于 RINTSTS。 表示以下位的逻辑或： EBE：End Bit Error RTO：Response Timeout RCRC：Response CRC SBE：Start Bit Error DRTO：Data Read Timeout DCRC：Data CRC for Receive RE：Response Error
29:26	RESERVED	保留
5	END OF RING (ER)	当设置此位时，表示描述符列表达到了它的最终描述符。IDMAC 返回列表的基地址，创建一个描述符环。这只适用于双缓冲区描述符结构。
4	SECOND ADDRESS CHAINED (CH)	当设置此位时，表示描述符中的第二个地址是下一个描述符的地址，而不是第二个缓冲区地址。当设置该位时，BS2 (DES1[25:13])应为全 0。
3	FIRST DESCRIPTOR (FD)	当设置此位时，表示该描述符包含数据的第一个缓冲区。如果第一个缓冲区的大小为 0，则下一个描述符包含数据的开始。
2	LAST DESCRIPTOR (LD)	这个位与 DMA 传输的最后一个块相关联。当设置时，该位表示该描述符指向的缓冲区是数据的最后一个缓冲区。描述符完成后，剩余的字节计数为 0。换句话说，在设置了 LD 位的描述符完成之后，剩余的字节计数应该是 0。
1	DISABLE INTERRUPT ON COMPLETION (DIC)	当设置此位时，将阻止 IDMAC 状态寄存器(IDSTS)中以该描述符指向的缓冲区结尾的数据的 TI/RI 位的设置。
0	RESERVED	保留

DES1 包含缓冲区大小信息。

表 28-6 DES1

位域	名称	描述
31:26	RESERVED	保留
25:13	BUFFER 2 SIZE (BS2)	这些位表示第二个数据缓冲区字节大小。缓冲区大小必须是 4 字节的倍数，这取决于总线宽度为 32 位。在缓冲区大小不是 4 或倍数的情况下，产生的行为是未定义的。如果该字段为 0，DMA 将忽略此缓冲区，并在双缓冲区结构的情况下继续处理下一个缓冲区。该字段对于链式结构无效，即设置了 DES0[4]时无效。
12:0	BUFFER 1 SIZE (BS1)	数据缓冲区字节大小，必须是 4 字节的倍数，这取决于总线宽度为 32 位。在缓冲区大小不是 4 的倍数的情况下，产生的行为是未定义的。这个字段不应该是零。

DES2 包含指向数据缓冲区的地址指针。

表 28-7 DES2

位域	名称	描述

31:0	BUFFER ADDRESS POINTER 1	当使用双缓冲区结构时，这些位表示第一个数据缓冲区的物理地址。对于链描述符，这些位表示数据缓冲区的物理地址。

如果当前描述符不是链式描述符结构中的最后一个描述符，则 DES3 包含指向下一个描述符的地址指针，或者双缓冲区结构的第二个缓冲区地址。

表 28-8 DES3

位域	名称	描述
31:0	BUFFER ADDRESS POINTER 2/NEXT DESCRIPTOR ADDRESS	当使用双缓冲区结构时，这些位表示第二个缓冲区的物理地址。如果第二地址链位(DES0[4])被设置，那么这个地址包含了指向下一个描述符所在的物理内存的指针。如果这不是最后一个描述符，则下一个描述符地址指针必须与总线宽度对齐(DES3[1:0] = 0 对应总线宽度为 32，内部 LBS 被忽略)

28.4.4.3. 初始化(Initialization)

IDMAC 初始化过程如下:

1）写入到 IDMAC 总线模式寄存器 BMOD 来设置主机总线访问参数。

2）写 IDMAC 中断使能寄存器 IDINTEN 以屏蔽不必要的中断。

3）软件驱动程序创建传输或接收描述符列表。然后写 IDMAC 描述符列表基址寄存器 DBADDR，为了向 IDMAC 提供列表的起始地址。

4）IDMAC 引擎试图从描述符列表中获取描述符。

28.4.4.4. 发送(Transmission)

IDMAC 传输过程如下:

1）主机设置用于发送的描述符(DES0-DES3)，并设置 OWN 位(DES0[31])。主机还准备数据缓冲区。

2）主机在 BIU 的 CMD 寄存器中编程写数据命令。

3）主机还将编写所需的发送阈值水平。

4）IDMAC 决定在第 2 步中需要进行写数据传输

5）IDMAC 引擎获取描述符并检查 OWN 位。如果没有设置 OWN 位，则意味着主机拥有描述符。在这种情况下，IDMAC 进入暂停状态，并在 IDSTS 寄存器中设置描述符不可用中断。在这种情况下，主机需要通过向轮询请求寄存器写入任何值来释放 IDMAC。

6）然后，它将等待命令完成(CD)位，并且没有来自 BIU 的指示传输完成的错误。

7）IDMAC 引擎现在将等待来自 BIU 的 DMA 接口请求。此请求将根据编程的发送阈值生成。对于不能使用突发访问的最后一个字节的数据，在 AHB 主接口上执行 SINGLE 传输。

8）IDMAC 从主机内存中的数据缓冲区中获取传输数据，并将数据传输到 FIFO 以传输到卡。

9）当数据跨越多个描述符时，IDMAC 将获取下一个描述符，并继续对下一个描述符进行操作。描述符中的 Last Descriptor 位表示数据是否跨越多个描述符。

10）当数据传输完成时，通过设置传输中断(如果启用)，状态信息在 IDSTS 寄存器中更新。另外，IDMAC 通过对 DES0 执行写操作来清除 OWN 位。

28.4.4.5. 接收(Reception)

主机设置用于接收的描述符(DES0-DES3)，并设置 OWN 位(DES0[31])。

主机在 BIU 的 CMD 寄存器中编程读数据命令。

主机将编写所需的接收阈值级别。

IDMAC 决定在第 2 步中需要进行读数据传输。

IDMAC 引擎获取描述符并检查 OWN 位。如果没有设置 OWN 位，则意味着主机拥有描述符。在这种情况下，IDMAC 进入暂停状态，并在 IDSTS 寄存器中设置描述符不可用中断。在这种情况下，主机需要通过向轮询请求寄存器写入任何值来释放 IDMAC。

然后，它将等待命令完成(CD)位，并且没有来自 BIU 的指示传输完成的错误。

IDMAC 引擎现在将等待来自 BIU 的 DMA 接口请求。此请求将根据编程的接收阈值生成。对于不能使用突发访问的最后一个字节的数据，在 AHB 上执行 SINGLE 传输。

IDMAC 从 FIFO 获取数据并传输到主机内存。

当数据跨越多个描述符时，IDMAC 将获取下一个描述符，并继续对下一个描述符进行操作。描述符中的 Last Descriptor 位表示数据是否跨越多个描述符。

当数据接收完成时，通过设置接收中断(如果启用)，状态信息在 IDSTS 寄存器中更新。另外，IDMAC 通过对 DES0 执行写操作来清除 OWN 位。

28.4.4.6. 中断(Interrupts)

中断可以作为各种事件的结果产生。IDSTS 寄存器包含所有可能导致中断的位。IDINTEN 寄存器为每个可能导致中断的事件都包含一个使能位。

在 IDSTS 寄存器中概述了两组汇总中断:正常中断和异常中断。中断通过将 1 写入相应的位位置来清除。当清除组内所有启用的中断时，相应的汇总位也会清除。

中断不排队，如果中断事件发生在驱动程序响应它之前，不会产生额外的中断。例如，Receive Interrupt-IDSTS[1]表示有一个或多个数据被传输到 Host 缓冲区。

当多个事件同时发生时，中断只会产生一次。驱动程序必须扫描 IDSTS 寄存器来查找中断原因。

28.5. SDMMC 寄存器描述

28.5.1. 寄存器列表

SDMMC 寄存器基地址：0x520C8000

偏移	名称	复位值	描述
0x00	SDMMC_CTRL	0x00000000	SDMMC 控制寄存器
0x04	SDMMC_PWREN	0x00000000	SDMMC 电源使能寄存器
0x08	SDMMC_CLKDIV	0x00000000	SDMMC 时钟分频寄存器
0x0c	SDMMC_CLKSRC	0x00000000	SDMMC 时钟源寄存器
0x10	SDMMC_CLKENA	0x00000000	SDMMC 时钟使能寄存器
0x14	SDMMC_TMOUT	0x3fffffc0040	SDMMC 超时寄存器
0x18	SDMMC_CTYPE	0x00000000	SDMMC 卡类型寄存器
0x1c	SDMMC_BLKSIZ	0x00000200	SDMMC 块大小寄存器
0x20	SDMMC_BYTCNT	0x00000200	SDMMC 字节个数寄存器
0x24	SDMMC_INTMASK	0x00000000	SDMMC 中断屏蔽寄存器

0x28	SDMMC_CMDARG	0x00000000	SDMMC 命令参数寄存器
0x2c	SDMMC_CMD	0x20000000	SDMMC 命令寄存器
0x30	SDMMC_RESP0	0x00000000	SDMMC 响应 0 寄存器
0x34	SDMMC_RESP1	0x00000000	SDMMC 响应 1 寄存器
0x38	SDMMC_RESP2	0x00000000	SDMMC 响应 2 寄存器
0x3c	SDMMC_RESP3	0x00000000	SDMMC 响应 3 寄存器
0x40	SDMMC_MINTSTS	0x00000000	SDMMC 屏蔽中断状态寄存器
0x44	SDMMC_RINTSTS	0x00000000	SDMMC 原始中断状态寄存器
0x48	SDMMC_STATUS	0x00000406	SDMMC 状态寄存器
0x4c	SDMMC_FIFOTH	0x000f0000	SDMMC FIFO 阈值寄存器
0x5c	SDMMC_TCBCNT	0x00000000	SDMMC 转移的卡字节计数寄存器
0x60	SDMMC_TBBCNT	0x00000000	SDMMC 转移的 FIFO 字节计数寄存器
0x64	SDMMC_DEBNCE	0x00ffffff	SDMMC 去抖动寄存器
0x74	SDMMC_UHS_REG	0x00000000	
0x78	SDMMC_RSTN	0x00000003	SDMMC 硬件复位寄存器
0x80	SDMMC_BMOD	0x00000000	SDMMC 总线模式寄存器
0x84	SDMMC_PLDMND	0x00000000	SDMMC 轮询需求寄存器
0x88	SDMMC_DBADDR	0x00000000	SDMMC 描述符列表基地址寄存器
0x8c	SDMMC_IDSTS	0x00000000	SDMMC IDMA 状态寄存器
0x90	SDMMC_IDINTEN	0x00000000	SDMMC IDMA 中断使能寄存器
0x94	SDMMC_DSCADDR	0x00000000	SDMMC 当前主机描述符地址寄存器
0x98	SDMMC_BUFADDR	0x00000000	SDMMC 当前主机缓冲区地址寄存器
0x100	SDMMC_CTHCTL	0x00000000	SDMMC 卡阈值控制寄存器
0x10c	SDMMC_EMMC_DDR	0x00000000	SDMMC EMMC DDR 寄存器
0x110	SDMMC_ENA_SHIFT	0x00000000	SDMMC 相位移动控制寄存器
0x200	SDMMC_DATA	0x00000000	SDMMC FIFO 通道访问寄存器

28.5.2. 控制寄存器(SDMMC_CTRL: 00h)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	USE_INTERNAL_DMAC	RW	0	数据传输是否使用 IDMA 0：不使用 IDMA，使用 AHB 从机 1：使用 IDMA
24:12	RSV	-	-	保留

11	CEATA_DEV_INT_STS	RW	0	CE-ATA 设备中断状态 0： 在 CE-ATA 设备上不使能中断(ATA 控制寄存器中 nIEN = 1) 1：在 CE-ATA 设备上使能中断(ATA 控制寄存器中 nIEN = 0) 在 CE-ATA 设备上电复位或其他复位后，软件应该适当地写入这个位。在复位后，CE-ATA 设备中断通常是禁止的，如果主机需要使能 CE-ATA 设备中断，则软件需要设置此位。
10	SEND_AUTO_STP_CCSD	RW	0	发送自动停止信号设置位 0：如果 SDMMC 没有复位该位,则清除此位。 1：将 CCSD 发送到 CE-ATA 设备后，再发送内部生成的 STOP。
9	SEND_CCSD	RW	0	发送命令完成信号禁用(CCSD)设置位 0： 如果 SDMMC 没有复位该位,则清除此位。 1： 发送命令完成信号禁用(CCSD)给 CE_ATA 设备。
8	ABORT_READ_DATA	RW	0	中止读数据设置位 0： 无变化 1： 中止读数据
7	SEND_IRQ_RESPONSE	RW	0	发送中断请求响应设置位 0： 无变化 1： 发送自动中断请求响应
6	READ_WAIT	RW	0	读等待设置位 0： 清除读等待 1： 设置读等待 置位时，发送读等待信号给 SDIO 卡。
5	RSV	-	-	保留
4	INT_ENABLE	RW	0	全局中断使能位 0： 禁止中断 1： 使能中断
3	RSV	-	-	保留
2	DMA_RESET	RW	0	DMA 复位 0： 无改变 1： 复位 IDMA 接口控制逻辑 为了复位 DMA 接口，软件需要设置此位为 1。该位在两个 AHB 时钟后自动清除。
1	FIFO_RESET	RW	0	FIFO 复位 0： 无变化 1： 复位数据 FIFO 为了复位 FIFO，软件需要设置此位为 1。该位在复位操作后自动清除。

0	CONTROLLER_RESET	RW	0	控制器复位 0： 无变化 1： 复位 SDMMC 控制器 为了复位控制器，软件需要设置此位为 1。该位在两个 AHB 时钟和两个 cclk_in 时钟周期后自动清除。 复位模块： BIU/CIU 接口 CIU 和状态寄存器 控制寄存器中的 ABORT_READ_DATA，SEND_IRQ_RESPONSE，READ_WAIT 位 命令寄存器中的 START_CMD 位 注：不影响 DMA 接口、FIFO 和主机中断。

28.5.3. 电源使能寄存器(SDMMC_PWREN: 04h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	POWER_EN	RW	0	电源开关，bit[0]控制卡 0，bit[1]控制卡 1。 0： 电源关闭 1： 电源使能 控制卡的电源。一旦打开卡的电源，在尝试初始化卡之前，软件需要等待稳压器/开关上升时间。

28.5.4. 时钟分频寄存器(SDMMC_CLKDIV: 08h)

位域	名称	属性	复位值	描述
31:24	CLK_DIV3	RW	0	时钟分频器 3 分频 参考 CLK_DIV0 描述
23:16	CLK_DIV2	RW	0	时钟分频器 2 分频 参考 CLK_DIV0 描述
15:8	CLK_DIV1	RW	0	时钟分频器 1 分频 参考 CLK_DIV0 描述
7:0	CLK_DIV0	RW	0	时钟分频器 0 分频 CLK0 的分频率为 2*n。比如，CLK_DIV0=0，分频率为 2*0=0，没有分频，输出 card 时钟为原频率；CLK_DIV0=1，分频率为 2*1=2，分频率为 2，输出 card 时钟为原频率的一半，依次类推。

28.5.5. 时钟源寄存器 (SDMMC_CLKSRC: 0Ch)

位域	名称	属性	复位值	描述

31:0	CLK_SOURCE	RW	0	卡的时钟来源。Bit[1:0]控制卡 0,bit[31:30]控制卡 15。 00：CLK_DIV0 01：CLK_DIV1 10：CLK_DIV2 11：CLK_DIV3

28.5.6. 时钟使能寄存器（SDMMC_CLKENA: 10h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	CLK_LOW_POWER	RW	0	时钟低功耗控制，bit[16]控制卡 0，bit[17]控制卡 1。 0：非低功耗模式 1：低功耗模式 当卡处于 IDLE 状态时，时钟停止 注：适用于 MMC 和 SD 存储卡，对于 SDIO 卡，如果必须检测中断，则不应停止时钟。
15:2	RSV	-	-	保留
1:0	CLK_ENABLE	RW	0	卡对应的时钟输出使能，bit[0]控制卡 0，bit[1]控制卡 1。 0：禁止时钟 1：使能时钟

28.5.7. 超时寄存器(SDMMC_TMOUT: 14h)

位域	名称	属性	复位值	描述
31:18	DATA_TIMEOUT	RW	0xFFFFFF	数据超时设置 卡读数据超时时间，以卡输出时钟 cclk_out 来计数。
7:0	RESPONSE_TIMEOUT	RW	0x40	响应超时设置 响应的超时时间，以卡输出时钟 cclk_out 来计数

28.5.8. 卡类型寄存器(SDMMC_CTYPE: 18h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	WIDTH1	RW	0	卡位宽设置位 1，bit[16]控制卡 0，bit[17]控制卡 1。 0：非 8 位模式 1：8 位模式 注：width1 寄存器比 width0 寄存器优先级高。当同为 1 时，controller 处于 8 位模式。
15:2	RSV	-	-	保留

1:0	WIDTH0	RW	0	卡位宽设置位 0，bit[0]控制卡 0，bit[1]控制卡 1。 0：1 位模式 1：4 位模式

28.5.9. 块大小寄存器(SDMMC_BLKSIZ: 1Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	BLOCK_SIZE	RW	0x200	块大小设置位 一个 BLOCK 包所含的字节个数

28.5.10. 字节个数寄存器(SDMMC_BYTCNT: 20h)

位域	名称	属性	复位值	描述
31:0	BYTE_COUNT	RW	0x200	待传输数据的字节个数 BYTE_COUNT 应该是 BLOCK_SIZE 的整数倍。BYTE_COUNT 为 0 表示未定义传输数据的数量，这时候应该由 HOST 发送 STOP 命令来停止数据传输。

28.5.11. 中断屏蔽寄存器(SDMMC_INTMASK: 24h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	SDIO_INT_MASK	RW	0	SDIO 中断屏蔽。bit[16]控制卡 0，bit[17]控制卡 1。 0：屏蔽 SDIO 中断 1：使能 SDIO 中断

15:0	INT_MASK	RW	0	屏蔽各个位的中断，将相应位清 0 表示屏蔽中断，置 1 表示使能中断。 15：结束位错误（读）/无 CRC（写） 14：自动结束命令 13：起始位错误/忙清除中断 12：硬件锁定写错误 11：FIFO 下溢/上溢错误 10：数据缺失超时/电压切换中断 9：读数据超时 8：响应超时 7：数据 CRC 错误 6：响应 CRC 错误 5：接收数据 FIFO 请求 4：发送数据 FIFO 请求 3：数据传输完成 2：命令结束 1：响应错误 0：卡检测

28.5.12. 命令参数寄存器(SDMMC_CMDARG: 28h)

位域	名称	属性	复位值	描述
31:0	CMD_ARG	RW	0	命令参数

28.5.13. 命令寄存器(SDMMC_CMD: 2Ch)

位域	名称	属性	复位值	描述
31	START_CMD	RW	0	开始(START)命令 一旦命令被 CIU 接收，该位清除。当该位置 1 时，HOST 不应该尝试写任何命令寄存器，如果写了的话，RAW INTERRUPT 寄存器中的 HARD LOCK ERROR INTERRUPT 被置 1。 一旦 START 命令发送并且收到来自卡的响应，RAW INTERRUPT 寄存器中的 COMMAND DONE 位被置 1。
30	RSV	-	-	保留
29	USE_HOLD_REG	RW	1	使用保持(HOLD)寄存器 0：CMD 和 DATA 发送到卡，不经过保持寄存器 1：CMD 和 DATA 发送到卡，经过保持寄存器

28	VOLT_SWITCH	RW	0	电压切换 0：无电压切换 1：电压切换使能，对于 CMD11 来说，该位必须值 1.
27	BOOT_MODE	RW	0	启动模式 0：强制启动操作 1：备用启动操作
26	DIS_BOOT	RW	0	禁用启动 当软件将该位与 START_CMD 一起设置时，CIU 终止启动操作。禁止将 DIS_BOOT 和 ENA_BOOT 同时设置。
25	EXPECT_BOOT_ACK	RW	0	期望启动确认 当软件将该位与 ENA_BOOT 一起设置时，CIU 期望从卡得到 0-1-0 的启动确认。
24	ENA_BOOT	RW	0	使能启动 只有在强制启动模式下才应该设置此位。当软件将该位与 START_CMD 一起设置时，CIU 通过设置 CMD 线低来响应卡的启动序列。禁止将 DIS_BOOT 和 ENA_BOOT 同时设置。
23	CCS_EXPECTED	RW	0	0：在 CE-ATA 设备中没有使能中断，或命令不期望从设备获得 CCS 1：CE-ATA 设备使能了中断，RW_BLK 命令期望来自 CE-ATA 设备的 CCS
22	READ_CEATA_DEVICE	RW	0	0：主机不对 CE-ATA 设备执行读访问 1：主机对 CE-ATA 设备执行读访问
21	UPDATE_CLK	RW	0	0：正常命令序列 1：不发送命令，仅仅更新时钟相关寄存器的值。时钟相关寄存器包括以下寄存器：CLKDIV, CLKSRC, CLKENA
20:16	CARD_NUM	RW	0	正在使用的卡号 表示被访问卡的物理槽位号
15	SEND_INI_SEQ	RW	0	发送初始化序列设置位 0：在发送命令之前不发送初始化序列(80 个卡时钟) 1：在发送命令之前发送初始化序列(80 个卡时钟) 上电后，在向卡发送任何命令之前，必须将 80 个时钟发送到卡进行初始化。该位应该在发送第一个命令到卡时设置，以便控制器在发送命令给卡之前，初始化时钟。在启动模式下不应设置此位。
14	STOP_ABORT_CMD	RW	0	0：无 STOP 或者 ABORT 命令用于停止正在进行的当前数据传输 1：STOP 或者 ABORT 命令用于停止正在进行的当前数据传输
13	WAIT_PRVDATA_COMPLETE	RW	0	0：立刻发送命令，即使此前的数据传输尚未完成。 1：等待此前的数据传输完成后再发送命令。
12	SEND_AUTO_STOP	RW	0	发送自动 STOP 0：数据传输完成后不发送 STOP 命令 1：数据传输完成后发送 STOP 命令

11	TRANSFER_MODE	RW	0	传输模式设置 0：块(BLOCK)数据传输命令 1：流(STREAM)数据传输命令
10	READ/WRITE	RW	0	读写设置位 0：从卡读数据 1：向卡写数据
9	DATA_EXPECTED	RW	0	数据期望 0：不期望与卡数据传输 1：期望与卡数据传输
8	CHECK_RESP_CRC	RW	0	检查响应 CRC 0：不检查响应 CRC 1：检查响应 CRC
7	RESP_LENGTH	RW	0	响应长度 0：期望来自卡的短响应(48 位) 1：期望来自卡的长响应(136 位)
6	RESP_EXPECT	RW	0	响应期望 0：不期望得到响应 1：期望得到响应
5:0	CMD_INDEX	RW	0	命令索引

28.5.14. 响应 0 寄存器(SDMMC_RESP0: 30h)

位域	名称	属性	复位值	描述
31:0	RESPONSE0	R	0	响应的[31:0]位

28.5.15. 响应 1 寄存器(SDMMC_RESP1: 34h)

位域	名称	属性	复位值	描述
31:0	RESPONSE1	R	0	响应的[63:32]位

28.5.16. 响应 2 寄存器(SDMMC_RESP2: 38h)

位域	名称	属性	复位值	描述
31:0	RESPONSE2	R	0	响应的[95:64]位

28.5.17. 响应 3 寄存器(SDMMC_RESP3: 3Ch)

位域	名称	属性	复位值	描述
31:0	RESPONSE3	R	0	响应的[127:96]位

28.5.18. 屏蔽中断状态寄存器(SDMMC_MINTSTS: 40h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	SDIO_INT	RW	0	bit[16]控制卡 0，bit[17]控制卡 1。 0：无来自于 SDIO 卡中断 1：有来自于 SDIO 卡中断 只有当对应位没有被中断屏蔽寄存器屏蔽时，中断位才有效。
15:0	INT_STATUS	RW	0	中断状态 只有在中断屏蔽寄存器中设置了相应位时，中断才能被启用 15：结束位错误(读)/无 CRC(写) 14：自动结束命令 13：起始位错误/忙清除中断 12：硬件锁定写错误 11：FIFO 下溢/上溢错误 10：数据缺失超时/电压切换中断 9：读数据超时 8：响应超时 7：数据 CRC 错误 6：响应 CRC 错误 5：接收数据 FIFO 请求 4：发送数据 FIFO 请求 3：数据传输完成 2：命令结束 1：响应错误 0：卡检测

28.5.19. 原始中断状态寄存器(SDMMC_RINTSTS: 44h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留

17:16	RAW_SDIO_INT	RW	0	bit[16]控制卡 0，bit[17]控制卡 1。 0：无来自于 SDIO 卡中断 1：有来自于 SDIO 卡中断 该中断位不受中断屏蔽寄存器影响，始终有效。写 1 清，写 0 无效。
15:0	RAW_INT_STATUS	RW	0	原始中断状态 15：结束位错误(读)/无 CRC(写) 14：自动结束命令 13：起始位错误/忙清除中断 12：硬件锁定写错误 11：FIFO 下溢/上溢错误 10：数据缺失超时/电压切换中断 9：读数据超时 8：响应超时 7：数据 CRC 错误 6：响应 CRC 错误 5：接收数据 FIFO 请求 4：发送数据 FIFO 请求 3：数据传输完成 2：命令结束 1：响应错误 0：卡检测 该中断位不受中断屏蔽寄存器影响，始终有效。写 1 清，写 0 无效。

28.5.20. 状态寄存器(SDMMC_STATUS: 48h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:17	FIFO_COUNT	R	0	FIFO 计数
16:11	RESP_INDEX	R	0	前一个响应的索引
10	DATA_STATE_MC_BUSY	R	1	DATA 发送或者接收状态机忙 0：状态机非 BUSY 1：状态机 BUSY
9	DATA_BUSY	R	X	DATA[0]的取反 0：卡数据非 BUSY 1：卡数据 BUSY
8	DATA_3_STATUS	R	X	DATA[3]的原始值，检查卡是否存在 0：卡不存在 1：卡存在

7:4	CMD_FSM_STATUS	R	0	命令状态机状态 0：IDLE 1：发送初始化序列 2：发送命令起始位 3：发送命令的传输方向位 4：发送命令的索引和参数 5：发送命令的 CRC7 6：发送命令结束位 7：收到响应起始位 8：收到中断响应 9：收到响应的传输方向位 10：收到响应的索引 11：收到响应的数据 12：收到响应的 CRC7 13：收到响应的结束位 14：命令路径等待 NCC 15：等待：从命令到响应的等待阶段
3	FIFO_FULL	R	0	FIFO 已满
2	FIFO_EMPTY	R	1	FIFO 已空
1	FIFO_TX_LEVEL	R	1	FIFO 的数据 COUNT 小于等于发送的 LEVEL 值(TX_LEVEL)
0	FIFO_RX_LEVEL	R	0	FIFO 的数据 COUNT 大于等于发送的 LEVEL 值(RX_LEVEL)

28.5.21. FIFO 阈值寄存器(SDMMC_FIFOTH: 4Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	RX_LEVEL	RW	0x00F	从卡读数据时 FIFO 的 LEVEL 值 当 FIFO 的数 COUNT 于等于这个值时，DMA/FIFO 请求信号起来。在整个传输的末端，不管 LEVEL 值的大小，DMA/FIFO 请求信号都会起来使得剩下的数据能被传出去。 在非 DMA 模式下，当 RXDR 中断使能，会产生 RXDR 中断以代替 DMA 请求。在整个传输的末端，当 LEVEL 值比剩下的数据个数大的时候，不会产生中断。当看到数据 Transfer Done 中断时，HOST 应该去读剩下的未完成的数据。 在 DMA 模式下，在整个传输的末端，即使剩下的数据个数比 LEVEL 值小，DMA 请求信号也会起来以进行一次单次传输，在数据 Transfer Done 中断起来之前传输完所有的数据。
15:12	RSV	-	-	保留

11:0	TX_LEVEL	RW	0	向卡写数据时 FIFO 的 LEVEL 值 当 FIFO 的数据 COUNT 小于等于这个值时，DMA/FIFO 请求信号起来。如果使能了中断，则产生中断。在整个传输的末端，不管 LEVEL 值的大小，DMA/FIFO 请求信号或者中断都产生。 在非 DMA 模式下，当 TXDR 中断使能，会产生 TXDR 中断以代替 DMA 请求。在整个传输的末端，在最后一次中断产生时，HOST 要把要求传输数量中尚未完成的数据填充到 FIFO 中去（不是在 FIFO 满之前或者传输完成之后，因为 FIFO 可能并不是空的）。 在 DMA 模式下，在整个传输的末端，如果最后一个传输少于一个 burst size，DMA controller 执行一个 single cycles 直到要求数量的数据传输完成。

28.5.22. 转移的卡字节计数寄存器(SDMMC_TCBCNT: 5Ch)

位域	名称	属性	复位值	描述
31:0	TRANS_CARD_BYTE_CNT	R	0	由 HOST CIU 模块发往卡的数据的字节数。 该寄存器仅在数据传输完成后读取，在数据传输期间读取该寄存器返回值为 0。

28.5.23. 转移的 FIFO 字节计数寄存器(SDMMC_TBBCNT: 60h)

位域	名称	属性	复位值	描述
31:0	TRANS_CARD_BYTE_CNT	R	0	在 HOST/DMA 内存与 BIU 的 FIFO 之间传输数据的字节数

28.5.24. 去抖动寄存器(SDMMC_DEBNCE: 64h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:0	DEBOUNCE_COUNT	RW	0xFFFFFF	Debounce 时钟周期数，用于卡检测防抖处理。使用系统时钟计数。去抖动典型时间是 5-25ms

28.5.25. UHS 寄存器(SDMMC_UHS_REG: 74h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	DDR_REG	RW	0	DDR 模式。bit[16]控制卡 0，bit[17]控制卡 15。 0：非 DDR 模式 1：DDR 模式

15:0	RSV	-	-	保留

28.5.26. 硬件复位寄存器(SDMMC_RSTN: 78h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	CARD_RSTN	RW	0x3	硬件复位。bit[0]控制卡 0，bit[1]控制卡 1。 0：复位 1：正常

28.5.27. 总线模式寄存器(SDMMC_BMOD: 80h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:8	PBL	RW	0	可编程突发长度(Programmable Burst Length) 这些位表示在一个 IDMAC 事务中执行的最大节拍数 000：1 001：4 010：8 011：16 100：32 101：64 110：128 111：256
7	DE	RW	0	IDMA 使能位 0：IDMA 不使能 1：IDMA 使能
6:2	DSL	RW	0	描述符跳过长度(Descriptor Skip Length) 该位指定了在两个未连接的描述符之间跳转的字(32 位)的数量。只适用于双缓冲结构
1	FB	RW	0	固定突发长度(Fixed Burst) 该位控制 AHB 主接口是否执行固定突发传输 0：AHB 在突发传输时使用 SINGLE 和 INCR 1：AHB 在突发传输时使用 SINGLE、INCR4、INCR8 或 INCR16

0	SWR	RW	0	软件复位(Software Reset) 当置位时，DMA 控制器复位其所有内部寄存器。1 个 AHB 时钟周期后自动清除

28.5.28. 轮询需求寄存器(SDMMC_PLDMND: 84h)

位域	名称	属性	复位值	描述
31:0	PD	W	0	轮询需求(Poll Demand) 如果描述符的 OWN 位没有设置，则 FSM 进入 Suspend 状态。主机需要将任意值写入该寄存器，让 IDMAC FSM 恢复正常的描述符获取操作。

28.5.29. 描述符列表基地址寄存器(SDMMC_DBADDR: 88h)

位域	名称	属性	复位值	描述
31:0	SDL	RW	0	描述符列表的起始处(Start of Descriptor List) 此字段包含第一个描述符的基地址。IDMA 会忽略 32 位总线宽度的 LSB 位(1:0)，并在内部将这些位视为全零值。因此，这些 LSB 位为只读 (RO)。

28.5.30. IDMA 状态寄存器(SDMMC_IDSTS: 8Ch)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16:13	FSM_STATUS	R	0	IDMAC 状态机当前状态 0：DMA_IDLE 1：DMA_SUSPEND 2：DESC_RD 3：DESC_CHK 4：DMA_RD_REQ_WAIT 5：DMA_WR_REQ_WAIT 6：DMA_RD 7：DMA_WR 8：DESC_CLOSE

12:10	FBE_CODE	R	0	致命总线错误码(Fatal Bus Error Code) 此字段指示导致总线错误的错误类型 001：在发送过程中收到 HOST 中止 010：在接收过程中收到 HOST 中止 其他：保留 仅对致命总线错误位 IDSTS[2]有效。该字段不会产生中断
9	AIS	RW	0	异常中断汇总(Abnormal Interrupt Summary) 异常中断汇总位的值是以下位的逻辑或运算结果： IDSTS[2]-Fatal Bus Interrupt IDSTS[4]-DU Interrupt 只有未屏蔽的位会影响异常中断汇总位。它是黏着位。每当导致 AIS 置 1 的对应位被清零时，必须同时将该位也清零（通过向此位写入 1）。
8	NIS	RW	0	正常中断汇总(Normal Interrupt Summary) 正常中断汇总位的值是以下位的逻辑或运算结果： IDSTS[0]-Transmit Interrupt IDSTS[1]-Receive Interrupt 只有未屏蔽位会影响正常中断汇总位。它是黏着位。每当导致 NIS 置 1 的对应位被清零时，必须同时将该位也清零（通过向此位写入 1）。
7:6	RSV	-	-	保留
5	CES	RW	0	卡错误汇总(Card Error Summary) 卡错误汇总位的值是以下位的逻辑或运算结果： EBE-END Bit Error RTO-Response Timeout/Boot Ack Timeout RCRC-Response CRC SBE-Start Bit Error DRTO-Data Read Timeout/BDS timeout DCRC-Data CRC for Receive RE-Response Error 写 1 清除该位
4	DU	RW	0	描述符不可用中断(Descriptor Unavailable Interrupt) 当描述符由于 OWN=0 (DES0[31] =0)而不可用时设置该位。写 1 会清除。
3	RSV	-	-	保留
2	FBE	RW	0	致命总线错误中断(Fatal Bus Error Interrupt) 此位指示发送总线错误。当这个位被设置时，DMA 禁用所有的总线访问。写 1 会清除。
1	RI	RW	0	接收中断(Receive Interrupt) 此位指示描述符的数据接收已完成，写 1 清除。

0	TI	RW	0	发送中断(Transmit Interrupt) 此位指示描述符的数据发送已完成，写 1 清除。

28.5.31. IDMA 中断使能寄存器(SDMMC_IDINTEN: 90h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	AIS_EN	RW	0	异常中断汇总使能(Abnormal Interrupt Summary Enable) 当置 1 时，使能异常中断汇总。 当置 0 时，禁止异常中断汇总。 该位使能以下位： IDINTEN[2]-Fatal Bus Interrupt IDINTEN[4]-DU Interrupt
8	NIS_EN	RW	0	正常中断汇总(Normal Interrupt Summary Enable) 当置 1 时，使能正常中断汇总。 当置 0 时，禁止正常中断汇总。 该位使能以下位： IDINTEN[0]-Transmit Interrupt IDINTEN[1]-Receive Interrupt
7:6	RSV	-	-	保留
5	CES_EN	RW	0	卡错误汇总中断使能(Card Error Summary Enable) 0：禁止中断 1：使能中断
4	DU_EN	RW	0	描述符不可用中断使能(Descriptor Unavailable Interrupt Enable) 0：禁止中断 1：使能中断
3	RSV	-	-	保留
2	FBE_EN	RW	0	致命总线错误中断使能(Fatal Bus Error Interrupt Enable) 0：禁止中断 1：使能中断
1	RI_EN	RW	0	接收中断使能(Receive Interrupt Enable) 0：禁止中断 1：使能中断
0	TI_EN	RW	0	发送中断使能(Transmit Interrupt Enable) 0：禁止中断 1：使能中断

28.5.32. 当前主机描述符地址寄存器(SDMMC_DSCADDR: 94h)

位域	名称	属性	复位值	描述
31:0	HDA	R	0	主机描述符地址指针(Host Descriptor Address Pointer) 这个寄存器指向 IDMAC 读取的当前描述符的起始地址。指针在操作过程中被 IDMAC 更新。此字段在复位时清除。

28.5.33. 当前缓冲区地址寄存器(SDMMC_BUFADDR: 98h)

位域	名称	属性	复位值	描述
31:0	HBA	R	0	主机缓冲区地址指针(Host Buffer Address Pointer) 这个寄存器指向正在被 IDMAC 访问的当前数据缓冲区地址。指针在操作过程中被 IDMAC 更新。此字段在复位时清除。

28.5.34. 卡阈值控制寄存器(SDMMC_CTHCTL: 100h)

位域	名称	属性	复位值	描述
31:23	RSV	-	-	保留
22:16	CARD_THR	RW	0	卡阈值大小
15:2	RSV	-	-	保留
1	BUSY_CLR_INT_EN	RW	0	繁忙清除中断使能(Busy Clear Interrupt Enable) 0：禁止 1：使能
0	CARD_RD_THR_EN	RW	0	卡读阈值使能(Card Read Threshold Enable) 0：禁止 1：使能

28.5.35. EMMC DDR 寄存器(SDMMC_EMMC_DDR: 10Ch)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留位。此字段必须为 0

1:0	HALF_START_BIT	RW	0	控制 SDMMC 内部基于起始位持续时间的起始位检测机制。bit[0]控制 0，bit[1]控制卡 1。对于 eMMC4.5，起始位可以是: 0：一个完整的时钟周期 1：少于一个完整的时钟周期 对于 eMMC4.5 及以上版本的卡，该位需要置 1。对于 SD 卡应用，该位需要置 0。在 DDR 模式下，此字段被忽略。

28.5.36. 相位移动寄存器(SDMMC_ENA_SHIFT: 110h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3:0	EN_SHIFT	RW	0	控制在设计中提供的相移量。bit[1:0]控制卡 0，bit[3:2]控制卡 1。 00：默认相移 01：移相到下一个上升沿 10：移相到下一个下降沿 11：保留

28.5.37. FIFO 通道访问寄存器(SDMMC_DATA: 200Ch)

位域	名称	属性	复位值	描述
31:0	DATA	RW	0	FIFO 数据访问通道，读写该寄存器即可以将数据读出或写入内部 FIFO。

29. 串行外设接口（SPI）

29.1. 概述

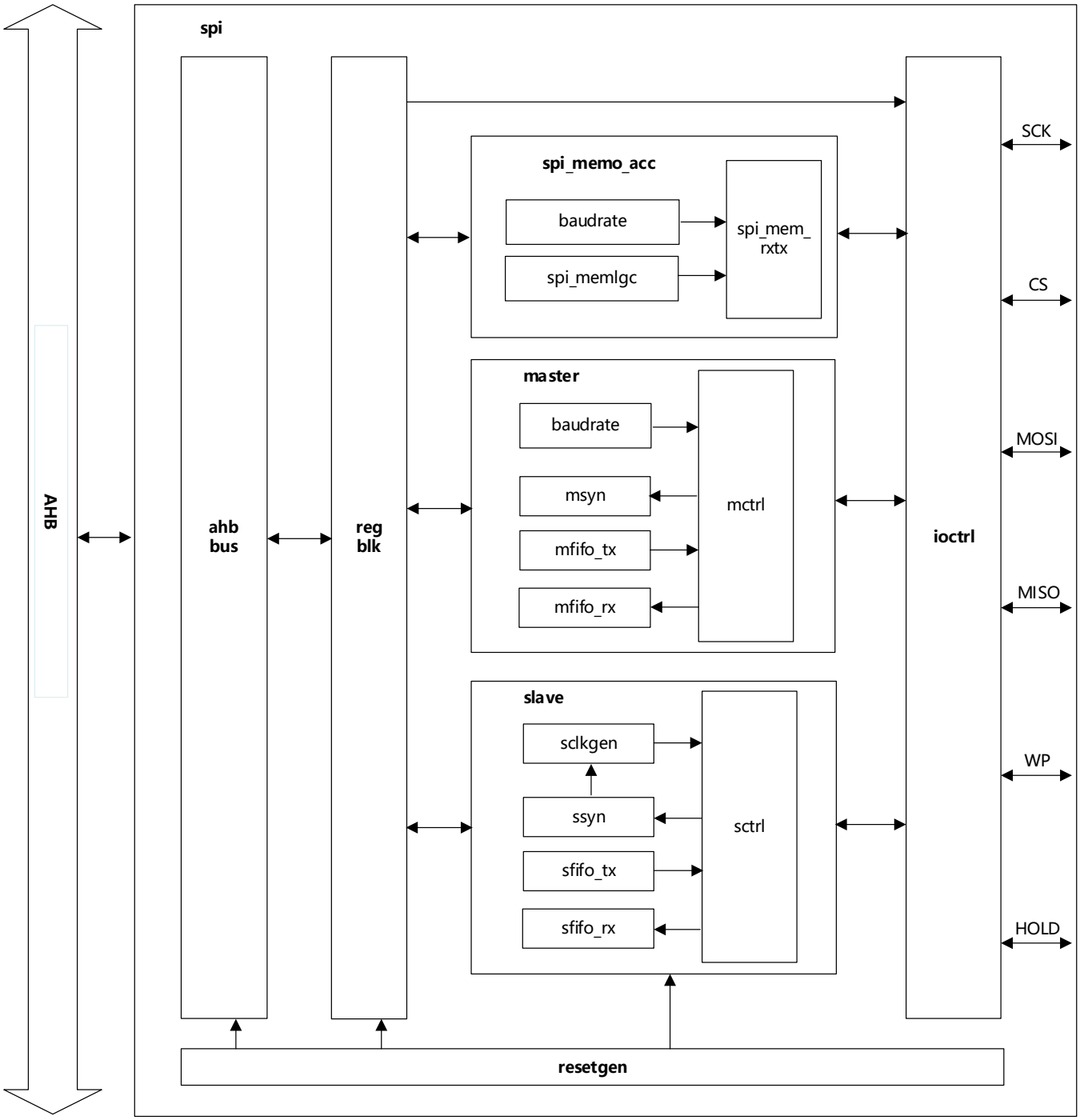
串行外设接口（SPI）模块用于微控制器（MCU）与满足 SPI 外设之间进行全双工、全同步、串行通讯，该接口可配置为主模式或从模式，在配置为主器件时，它为外部从器件提供通信时钟（SCK）。从器件选择信号（CS）可以由主器件提供，也可以选择由从器件提供。SPI 主/从模式均支持标准 SPI 模式（1 线模式）、DSPI 模式（2 线模式）、QSPI 模式（4 线模式）。

29.2. 要特性

- 支持 SPI 主/从模式
- 支持全双工、半双工同步传输
- 可通过两级分频因子来配置宽范围波特率
- 主模式下最大频率可达内核频率（HCLK）的一半
- 从模式下最大频率可达内核频率（HCLK）的一半
- 可编程的时钟相位和极性
- 支持标准 SPI 模式（1 线模式）、DSPI 模式（2 线模式）、QSPI 模式（4 线模式）
- SPI 从模式支持软件 CS 控制功能
- 支持从机 SCK、CS 滤毛刺功能
- 可调节的主器件接收器采样时间
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 具有 DMA 功能的两个 16x 8 位的内置 Rx 和 Tx FIFO
- 可编程的传输数据量
- 支持内存映射模式（SPI4、SPI7、SPI8 支持）
- 内存映射模式允许 8、16 和 32 位数据访问

29.3. 功能描述

29.3.1. 结构框图



如上图所示，ahb_bus 为 ahb 总线 slave 接口，reg_blk 为配置寄存器组，resetgen 产生整个 spi 模块的复位信号。spi_memo_acc 模块为内存映射模式模块，通过硬件自动完成命令、地址以及其他参数的发送，实现对 spi 存储器的读写。master 模块为 FIFO 模式模块。spi 配置为主机时，master 产生时钟、片选信号和控制数据的收发。spi 配置为从机时，slave 解析片选信号和控制数据的收发。ioctrl 控制 io 的输出和输入。

29.3.2. SPI 的功能模式

SPI 可以工作在以下两种功能模式下工作：

- FIFO 模式：为了和 SPI 内存映射模式进行区别，将通常的 SPI 数据传输方式，即使用 SPI 寄存器来执行 SPI 总线上数据传输的操作称为 FIFO 模式。

- 内存映射模式：通过 SPI 模块作为主机将外部读写设备（如 SPI Nor FLASH，SPI PSRAM 等）直接映射到 MCU 的地址空间，从而可以通过直接地址访问的方式来读写该设备，SPI 总线上的信号传输均由该 SPI 模块内部自动转换实现。

29.3.3. SPI 接口信号

SPI 模块有六个 I/O 引脚，用于与外部器件进行 SPI 通信。 该模块支持一线模式和多线传输模式（二线和四线模式），接口 IO 定义为：IO0 (MOSI)，IO1 (MISO)，IO2 (WP)，IO3 (HOLD) 。

SCK：时钟信号 。主器件输出时钟信号，从器件输入时钟信号，SCK 的频率可通过寄存器 SPI_BAUD 来配置，详见波特率设置寄存器(SPI_BAUD：04h)。

CS：CS 管脚都是由主机控制，对于主机来说，它用于片选某个外设（当有多个从设备挂载在 SPI 总线上时）；对于从机来说，当 CS 被片选中之后，从机的状态机才进入工作状态，从而开始接收主机发送过来的 SCK 信号。当 SPI 从机支持并开启软件 CS 功能后，也可以省去 CS 管脚。详见 SPI 从模式 CS 功能。

IO0（MOSI）：在单线模式中主器件为串行输出，从器件为串行输入。在双线/四线模式中为双向 IO。

IO1（MISO）：在单线模式中主器件为串行输入，从器件为串行输出。在双线/四线模式中为双向 IO。

IO2（WP）：在单线/双线模式中为串行输入。在四线模式中为双向 IO。

IO3（HOLD）：在单线/双线模式中为串行输入。在四线模式中为双向 IO。

三种模式下的引脚复用关系如下表所示：

模式/信号	SCK	CS	MOSI	MISO	WP	HOLD
标准 SPI	SCK	CS	MOSI	MISO		
DSPI	SCK	CS	IO0(MOSI)	IO1(MISO)		
QSPI	SCK	CS	IO0(MOSI)	IO1(MISO)	IO2(WP)	IO3(HOLD)

29.3.4. SPI 通信格式

SPI 通信过程中，可执行接收和发送操作。串行时钟（SCK）对数据线上的信息的移位和采样进行同步。通信格式取决于时钟相位、时钟极性和数据帧格式。为了能够在彼此间进行通信，主器件和从器件必须遵循相同的通信格式并且必须正确同步。

■ 时钟相位和极性控制

通过 CTL.CPOL 和 CTL.CPHA 位，可以用软件选择四种可能的时序关系。

时钟极性（CPOL）表示时钟（SCK）不传输数据（空闲）时的电平状态。CPOL 复位，表示 SCK 空闲时为低电平。CPOL 置位，表示 SCK 空闲时为高电平。

时钟相位（CPHA）表示锁存数据位的时钟边沿。CPHA 复位，表示在 SCK 的第一个边沿锁存数据位（CPOL 复位时为上升沿，CPOL 置位时为下降沿）。CPHA 置位，表示在 SCK 的第二个边沿锁存数据位（CPOL 复位时为下降沿，CPOL 置位时为上升沿）。

时钟极性（CPOL）和时钟相位（CPHA）的组合用于选择数据捕获时钟边沿。

注： 禁止在传输过程或 CS 为低期间切换 CPOL/CPHA。

SCK 的空闲状态必须与 SPI_CTL[3]寄存器中选择的极性相对应（如果 CPOL=1，则上拉 SCK；如果 CPOL=0，则下拉 SCK）。

■ 数据帧格式

SPI 移位寄存器可设置为以 MSB 在前或 LSB 在前的方式移出数据，具体取决于 SPI_CTL.LSB 位的值。

29.3.5. SCK 波特率设置

SCK 波特率的设置，通过 BAUD 寄存器实现。主器件需要设置波特率，从器件不需要设置波特率。

SCK 波特率计算公式：SPI_CLK = fHCLK / (DIV1 * (DIV2+1))。

注：DIV1 必须是 2 到 254 之间的偶数（包括 2 和 254）、DIV2：0-255。

例如：在 fHCLK 为 64MHz 时，如需设置 SPI 的时钟频率为 8MHz，可设置 DIV1 为 8，DIV2 为 0.

29.3.6. 接口模式

SPI 接口类型，通过 SPI_CTL.X_Mode 进行配置，配置项如下表所示。

X_Mode[1: 0]	SPI 类型
00	1 线模式（标准 SPI）
01	2 线模式（DSPI）
10	4 线模式（QSPI）

29.3.7. 1 线模式

FIFO 模式时：

当 CTL. X_Mode 为 0b’ 00 时，SPI 进入 1 线模式（标准 SPI）。

一线模式支持全双工、半双工。

主器件通过 IO0（MOSI）发送数据，IO1（MISO）接收数据。

从器件通过 IO0（MOSI）接收数据，IO1（MISO）发送数据。

CTL.IO_MODE 控制 IO 输入输出方向的方法。

CTL.IO_MODE 复位，表示软件手动控制 IO 输入输出方向，软件通过 SPI_OUT_EN 寄存器控制 MOSI、MISO、WP、HOLD 四个引脚的输入输出方向。

CTL.IO_MODE 置位，表示硬件自动控制 IO 输入输出方向，不受 SPI_OUT_EN 寄存器控制影响。

当硬件自动控制 IO 输入输出方向时：

- IO0：主输出，从输入。主模式接收数据（半双工）时，该信号电平受 TX_CTL.Dummy 控制。
- IO1：主输入，从输出。从模式接收数据（半双工）时，该信号电平受 TX_CTL.Dummy 控制。
- IO2：主输入，从输入。IO 电平受 IO 的上拉/下拉电阻配置控制。
- IO3：主输入，从输入。IO 电平受 IO 的上拉/下拉电阻配置控制。

一线模式时，IO2、IO3 可配置为 GPIO 使用。

内存映射模式时，硬件自动控制 IO 方向，不受 CTL .IO_MODE、SPI_OUT_EN 影响。

时序图：

图 29-1 SPI 一线模式时序图（MSB）

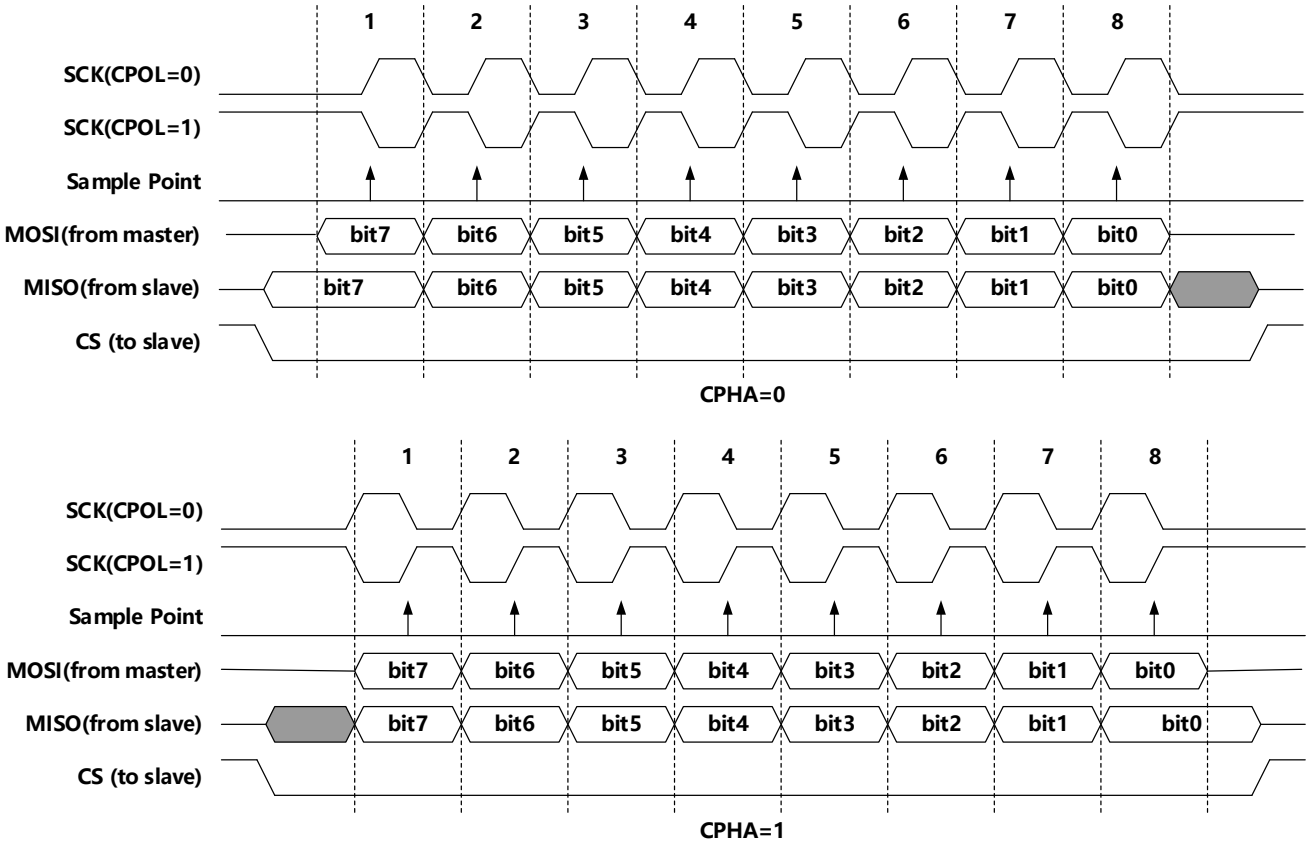
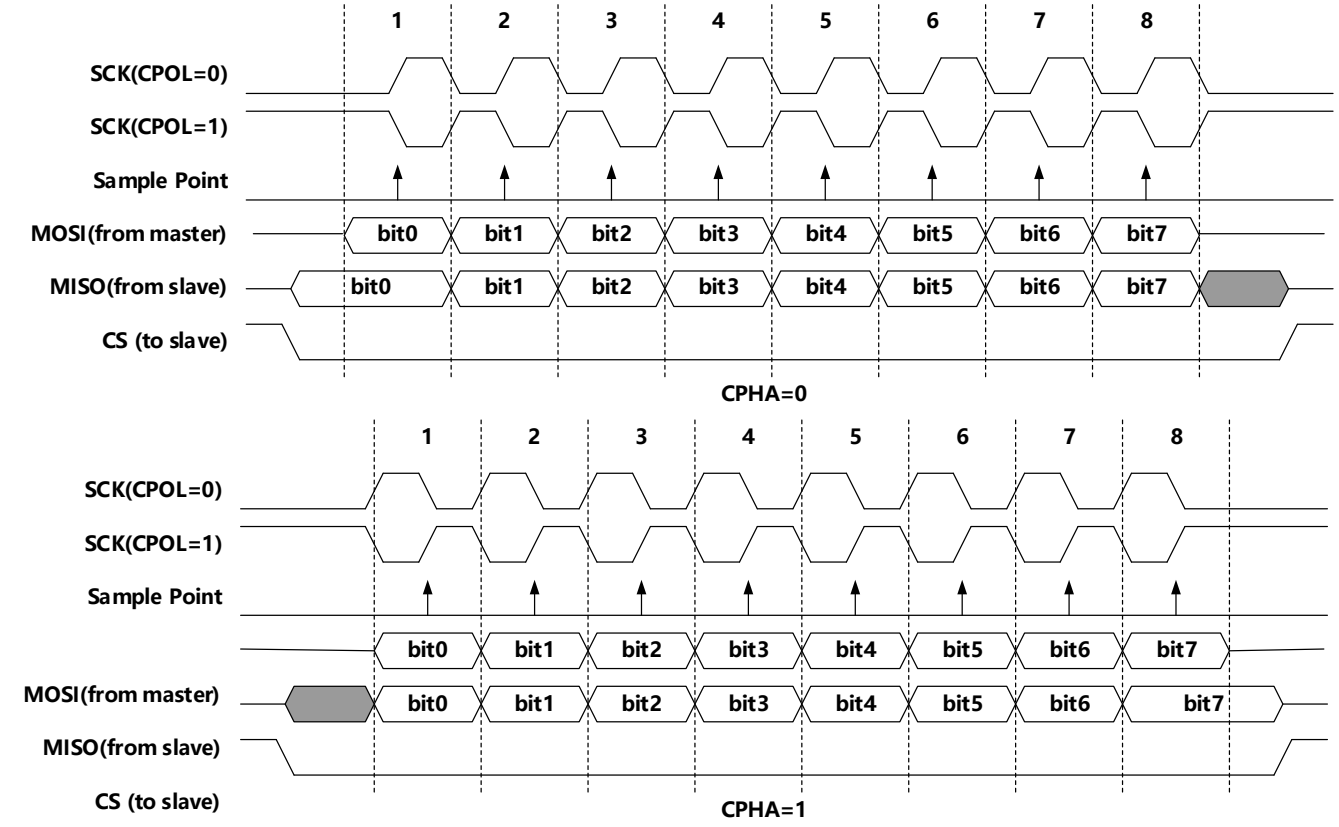


图 29-2 SPI 一线模式时序图（LSB）



29.3.8. 2 线模式

FIFO 模式时：

当 CTL. X_Mode 为 0b’ 01 时，SPI 进入双线模式（DSPI)。

双线模式支持半双工，不支持全双工。主器件、从器件通过 IO0/IO1 进行发送/接收数据。

CTL .IO_MODE 控制 IO 输入输出方向的方法。

CTL.IO_MODE 复位，表示软件手动控制 IO 输入输出方向，软件通过 SPI_OUT_EN 寄存器控制 MOSI、MISO、WP、HOLD 四个引脚的输入输出方向。

CTL.IO_MODE 置位，表示硬件自动控制 IO 输入输出方向，不受 SPI_OUT_EN 寄存器控制影响。

当硬件自动控制 IO 输入输出方向时：

- IO0：发送时自动切换为输出，接收时自动切换为输入。
- IO1：发送时自动切换为输出，接收时自动切换为输入。
- IO2：输入。IO 电平受 IO 的上拉/下拉电阻配置控制。
- IO3：输入。IO 电平受 IO 的上拉/下拉电阻配置控制。

双线模式时，IO2、IO3 可配置为 GPIO 使用。

内存映射模式时，硬件自动控制 IO 方向，不受 CTL .IO_MODE、SPI_OUT_EN 影响。

时序图：

图 29-3 SPI 二线模式时序图（MSB)

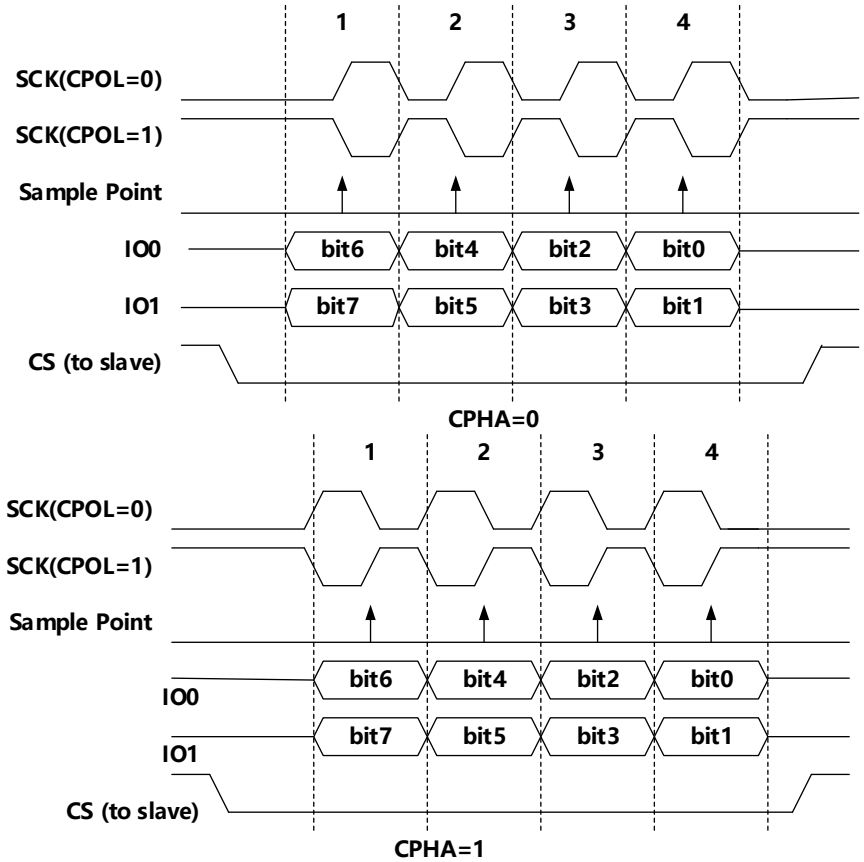
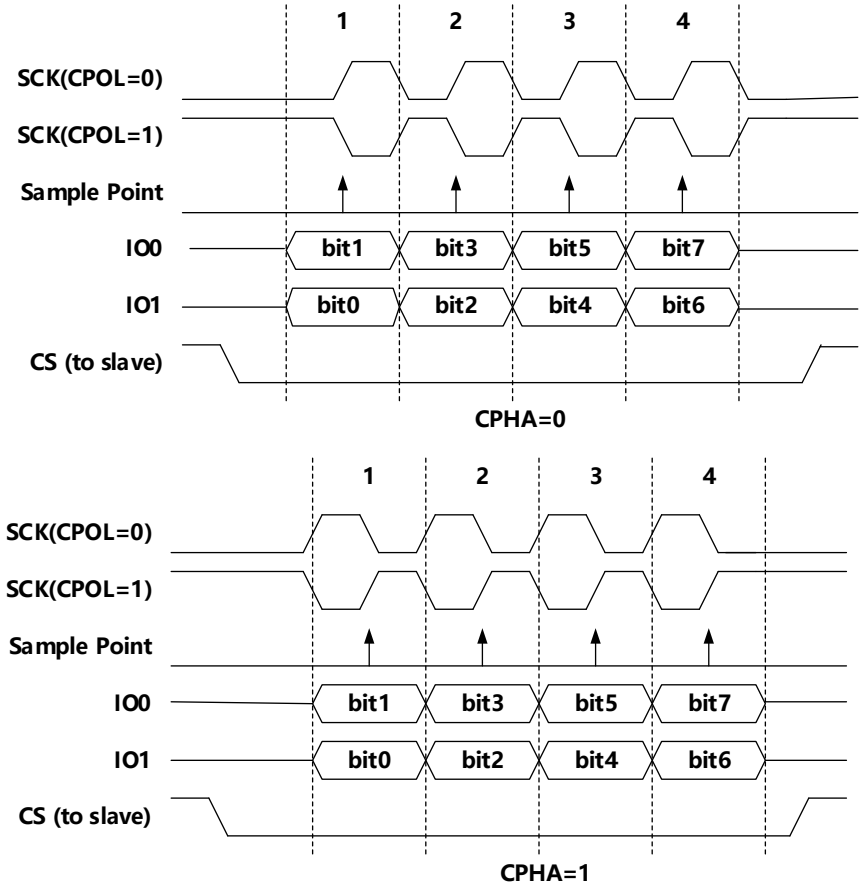


图 29-4 SPI 二线模式时序图（LSB）



29.3.9. 4 线模式

FIFO 模式时：

当 CTL. X_Mode 为 0b’ 10 时，SPI 进入四线模式（QSPI）。

四线模式支持半双工，不支持全双工。主器件、从器件通过 IO0/IO1/IO2/IO3 信号进行发送/接收数据。

CTL .IO_MODE 控制 IO 输入输出方向的方法。

CTL.IO_MODE 复位，表示软件手动控制 IO 输入输出方向，软件通过 SPI_OUT_EN 寄存器控制 MOSI、MISO、WP、HOLD 四个引脚的输入输出方向。

CTL.IO_MODE 置位，表示硬件自动控制 IO 输入输出方向，不受 SPI_OUT_EN 寄存器控制影响。

当硬件自动控制 IO 输入输出方向时：

- IO0：发送时自动切换到输出，接收时自动切换到输入。
- IO1：发送时自动切换到输出，接收时自动切换到输入。
- IO2：发送时自动切换到输出，接收时自动切换到输入。
- IO3：发送时自动切换到输出，接收时自动切换到输入。

内存映射模式时，硬件自动控制 IO 方向，不受 CTL .IO_MODE、SPI_OUT_EN 影响。

时序图：

图 29-5 SPI 四线模式时序图（MSB）

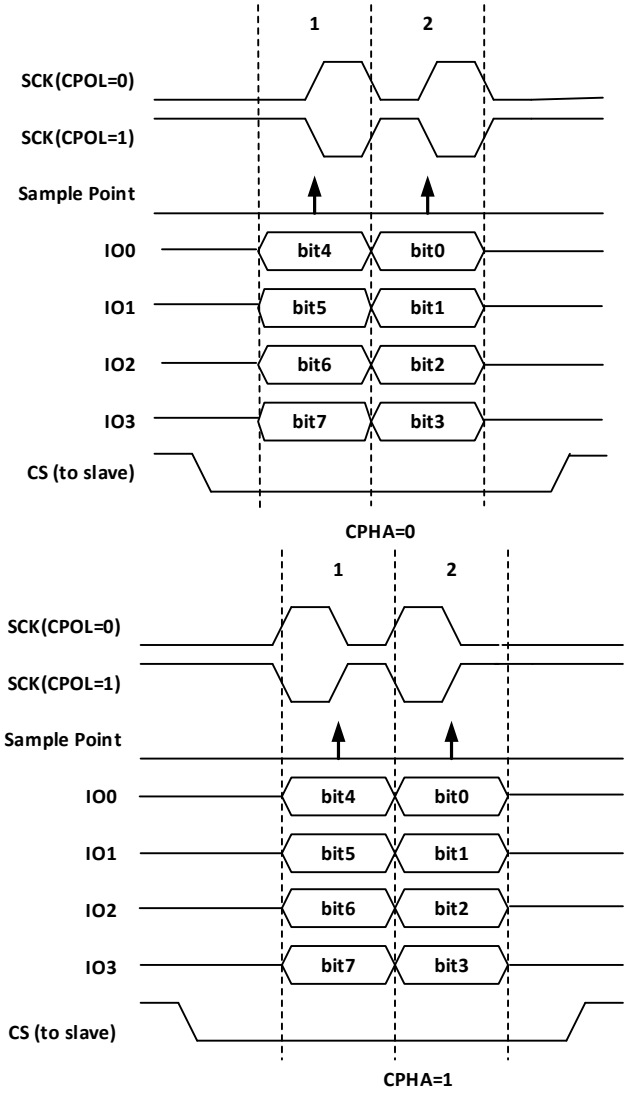
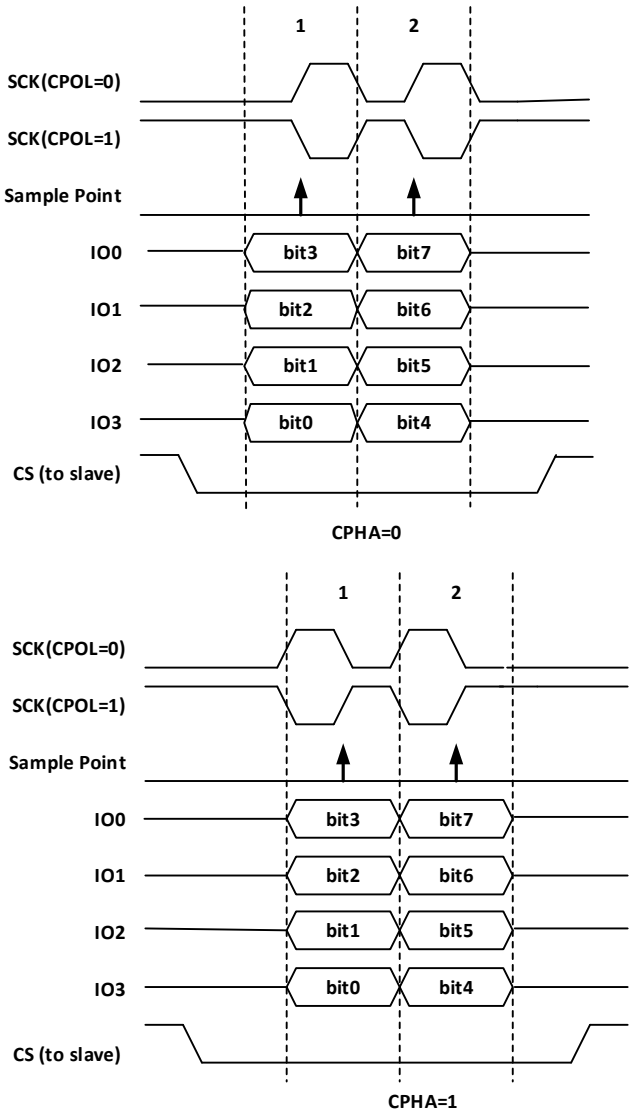


图 29-6 SPI 四线模式时序图（LSB）



29.3.10. 通信类型

■ 全双工

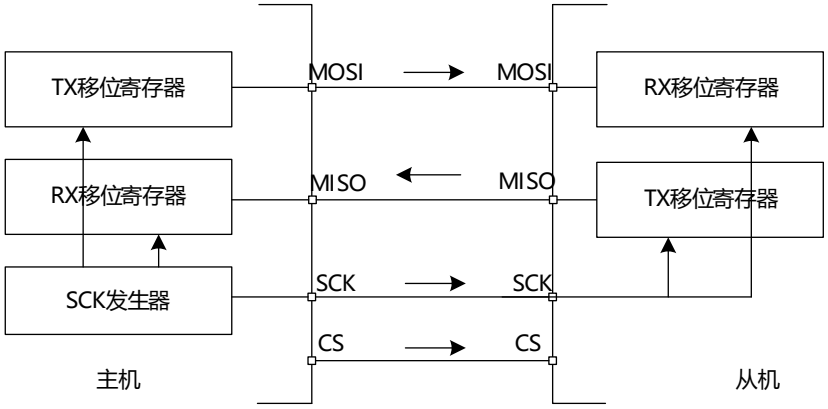
一线模式时，仅 FIFO 模式下 SPI 支持全双工通信，而双线模式、四线模式不支持全双工通信。

当 TX_CTL.TX_EN 和 RX_CTL.RX_EN 同时使能时，SPI 进入全双工通信。

主器件的 TX 移位寄存器通过 MOSI 与从器件的 RX 移位寄存器连接，主器件的 RX 移位寄存器通过 MISO 与从器件的 TX 移位寄存器连接。

主器件提供同步通信的时钟 SCK，数据随 SCK 时钟边沿同步移位。主器件通过 MOSI 线将待发送的数据发送给从器件，同时通过 MISO 线从从器件接收数据。当数据帧传输完成时（所有位均移出），主器件和从器件之间完成数据交换。

图 29-7 SPI 全双工通信



■ 半双工

一线模式、双线模式、四线模式时，SPI 都支持半双工通信。

当 TX_CTL.TX_EN 和 RX_CTL.RX_EN 仅有一个使能时，SPI 进入半双工通信。

半双工，所有 IO 口依然被占用。

主发从收时，使用 SCK、MOSI、CS 三个信号线。无论主器件，还是从器件，MISO 都默认为输入方向。

主收从收时，使用 SCK、MISO、CS 三个信号线。无论主器件，还是从器件，MOSI 都默认为输入方向。

图 29-8 半双工主发从收

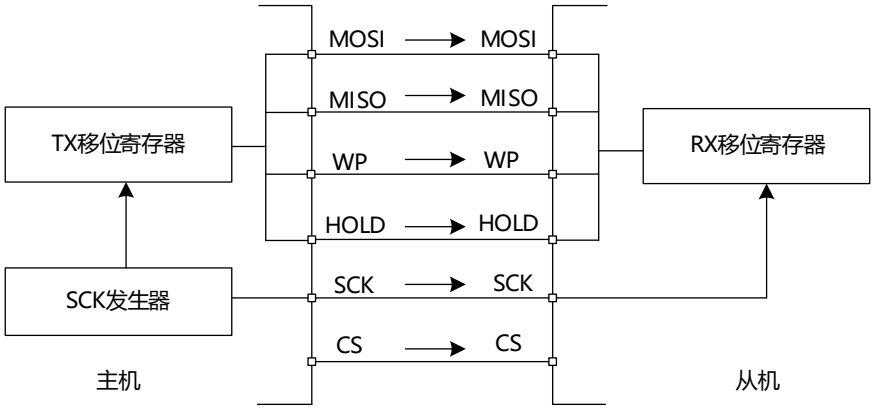
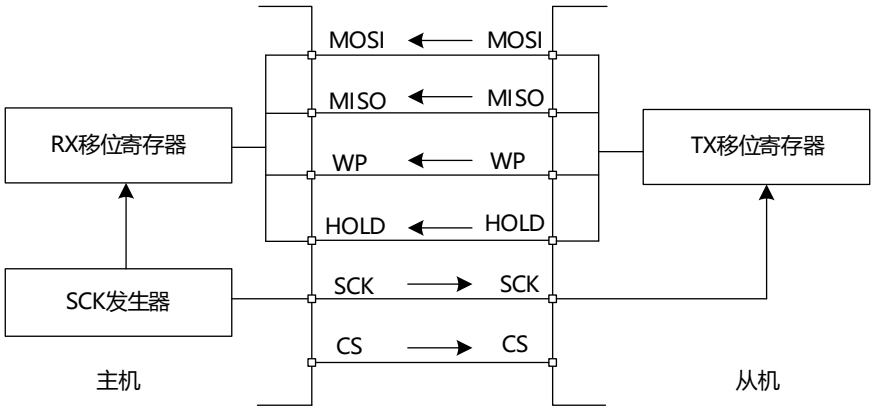


图 29-9 半双工主收从发



29.3.11. SPI 多片选功能

该 SPI 模块支持多片选功能，作为主机时，可通过 SPI_CS.SPI_CS[4：0]对应比特位选择使用 CS0-CS4 片选信号来选择从机，CS1 和 CS2 是通过 SPI 管脚 IO1 和 IO2 映射输出的，仅在一线模式下使用，通过 SPI_CS.CSMAP_EN 位开启 CS1 和 CS2 片选功能；CS0/CS3/CS4 有独立管脚，可在任何时候使用；作为从机时，CS0/CS3/CS4 均可作为 CS 信号输入管脚，无需其它操作。

29.3.12. SPI 从机 NCS 功能

当 GPIO 资源紧张时，通过软件 CS 功能，可以从硬件方面为从器件节约一个 CS 信号线。软件 CS 模式时，从器件的片选不再受主机的 CS 管脚影响，而是通过软件的方式控制从机片选。

使能 CTL. SWCS_EN，SPI 进入 NCS 模式。复位 CTL. SWCS，从机片选有效，开始发送/接收数据。使能 CTL. SWCS，从器件片选无效，停止发送/接收数据。

NCS 的配置必须在 SPI 模块初始化配置完成之后进行。如果先配置 NCS 功能，从器件立即处于工作状态，对 SCK 进行检测。之后再配置时钟极性时，SCK 可能会产生一个高低电平的翻转，从而让从器件误判为一个 SCK 有效边沿，造成从器件时序错乱。

该功能也可实现一主多从控制，利用软件进行选择从机，从而进行数据传输。

29.3.13. 采样移位

采样移位仅适用于主机接收数据的场景。当从机发送的数据延后于主机 SCK 采样边沿，按正常时序采样，主机会在数据输出前进行采样，最终导致采样错误。设置 RX_CTL. SSHIFT，可以将采样时间延后 0-3 个 HCLK 时钟周期，从而匹配数据与采样的时序，确保有效采样。

该功能一般用于 SCK 较高频率下，需检测出 SPI 主机和外设/从机传输的电气特性延迟，根据需要设置。如果不确定其传输延迟，当 SCK 在较快频率下（如 SCK 为 100MHz）无法正确传输，可设置波特率值，使 SCK 频率变低，若能正确传输，则可配置该功能，在较快 SCK 频率下分别尝试 1-3 延迟，调配出最佳延迟。该功能配置位在 RX_CTL. SSHIFT 位进行设置。下图为二分频 RX_CTL. SSHIFT 设置为不同值时的时序图。

图 29-10 移位采样示意图（CPOL=0，CPHA=0，MSB，RX_CTL. SSHIFT=1）

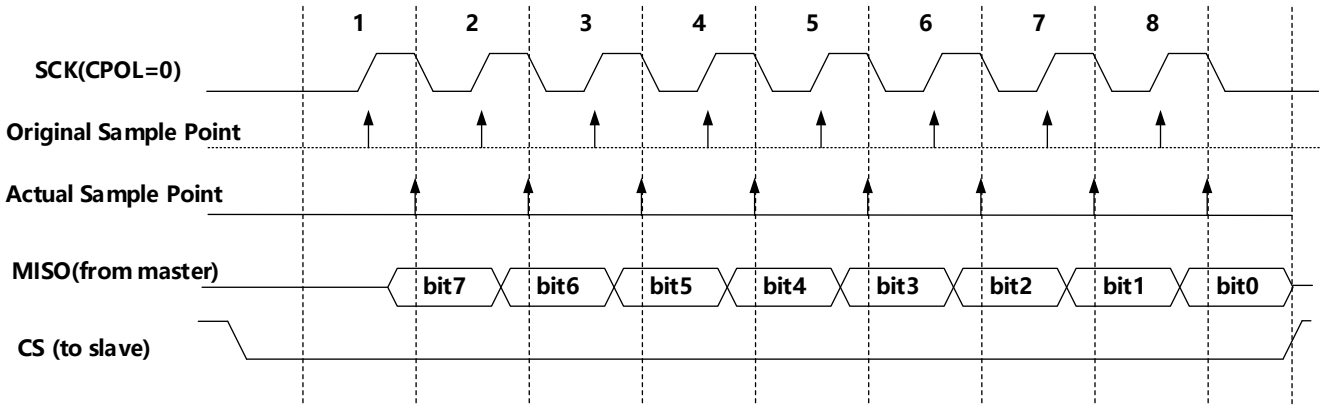
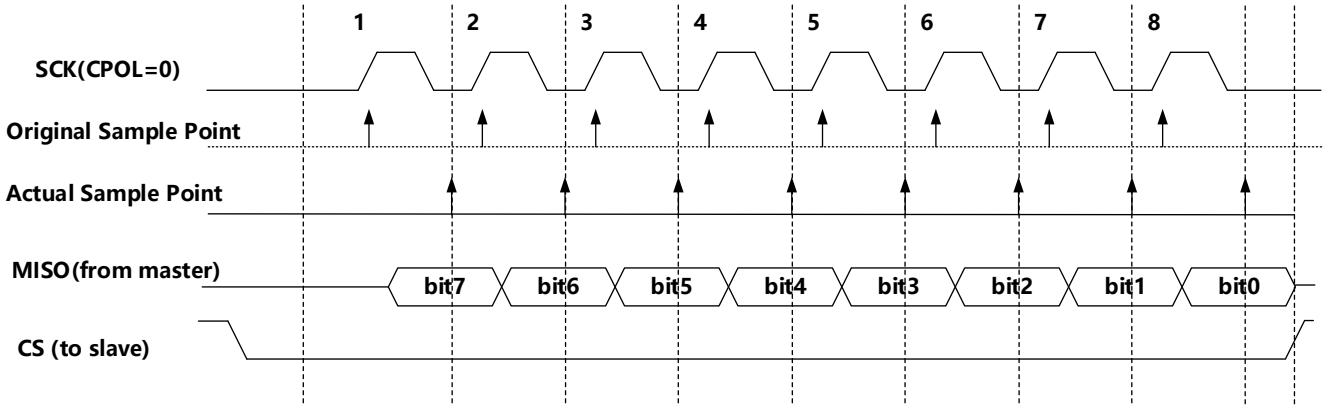


图 29-11 移位采样示意图（CPOL=0，CPHA=0，MSB，RX_CTL. SSHIFT=2）



主机推迟 1.5/2.5 个 hclk 采样指对推迟 2/3 个 hclk 采样点的前半个 hclk 处提前进行数据采样，实际采样点依然为推迟 2/3 个 hclk 采样点，因此主机推迟 1.5/2.5 个 hclk 周期开始采样的效率与主机推迟 2/3 个 hclk 周期开始采样的效率相同。当 MSDA_EN 使能时，采样数据为微调后的数据；MSDA_EN 不使能则为 IO 输入数据。

29.3.14. SPI 从机滤毛刺功能

硬件滤波功能指对从器件的 CS 和 SCK 进行滤波，有效防止 CS 和 SCK 电气特性干扰产生的毛刺。硬件滤波仅适用于从器件。

使能 CTL. CS_FILTER，开启从器件硬件滤波功能。

硬件滤波功能开启后，对 SPI 传输速率有一定的约束，从器件的 HCLK 频率应大于等于 SCK 频率的 12 倍。

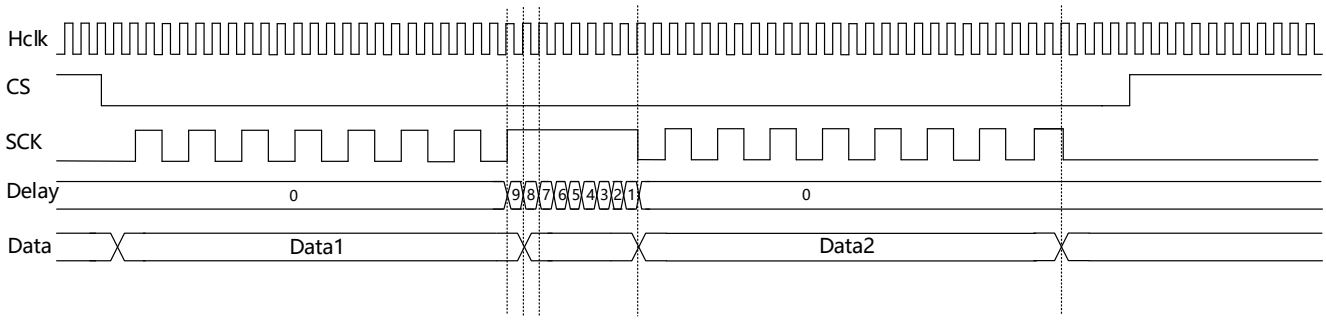
29.3.15. 传输等待功能

该功能仅在主模式下由 SPI_TXDelay 寄存器来决定，目的是为了在不降低 SCK 频率的条件下，降低 SPI 传输的速率。当通过 DMA 来传输数据或 CPU 与 SPI FIFO 之间数据交换速率较快时，SCK 处于连续不间断发送，速率较快，当设置 SPI_TXDelay 的值，将会有一个 32 位的计数器在每一字节传输结束后开始计数，当到达设定值时下一字节数数据开始传输。

在发送等待的过程中，SCK 电平将按照最后 1Bit 数据采样沿值停止，等待下一字节数据传输。

下图为 TX_Delay 配置为 9 时的传输时序图。

图 29-12 SPI 发送时序图（CPOL=0/CPHA=0）



在主机发送时，如果 CPU 写入数据速率较慢，而 SPI 传输数据速率较快，会出现和传输等待功能类似的情况，中间空闲部分处于等待 FIFO 中写入数据，SCK 电平将按照最后 1Bit 数据采样沿值停止；主机接收时，如果 CPU 读 FIFO 数据速率较慢，而 SPI 传输数据速率较快，导致 RX_FIFO 满，也将会出现和传输等待功能类似的情况，中间空闲部分处于等待 CPU 读走 FIFO 中数据，SCK 电平将按照最后 1Bit 数据采样沿值停止；

29.3.16. 批量传输

该 SPI 模块支持批量传输和无限数据的收发。

当需要批量传输时，在主或从收发开始前（CS 拉低前），需要先配置 SPI_BATCH.Batch_Number，所配置值为此次传输的数据量，当完成设置数据量传输后，SPI 才会退出至空闲状态，否则会处于等待状态；

当需要无限数据收发时，配置 SPI_BATCH.Batch_Number 为零。作为主机，当发送开始后，TX_FIFO 中不空立即开始传输，TX_FIFO 变空后，传输等待；接收时，RX_FIFO 不满，一直处于接收状态并不断发送 SCK，RX_FIFO 满后，传输等待。当全双工传输时（仅一线），由主发控制主收。

该 SPI 模块作为从机，该寄存器设置值只会影响从机 batch 中断产生，不影响数据传输。设置不为零时，完成传输会置起相应的 RX_BATCH_DONE/TX_BATCH_DONE/BATCH_DONE 中断状态，可供软件处理；设置为零时，不影响数据传输，相应的 RX_BATCH_DONE/TX_BATCH_DONE/BATCH_DONE 中断将无效。

29.3.17. SPI 中断和状态

中断和状态	中断使能位	描述
INSTR_SEND_DONE	无	指令已发送记忆标志位
RX_BATCH_DONE	RX_BATCH_DONE_EN	接收模式下批量传输完成标志位。
TX_ BATCH_DONE	TX_BATCH_DONE_EN	发送模式下批量传输完成标志位。
RX_FIFO_FULL_OVERFLOW	RX_FIFO_FULL_OVERFLOW_EN	从机接收 FIFO 写入溢出标志位。
RX_FIFO_EMPTY_OVER_FLOW	RX_FIFO_EMPTY_OVERFLOW_EN	从机接收 FIFO 读出溢出标志位。
RX_FIFO_NOT_EMPTY	RX_FIFO_NOT_EMPTY_EN	接收 FIFO 非空标志位。
CS_POS_Flg	CS_POS_EN	CS 管脚电平上升沿事件标志位。
RX_FIFO_HALF_FULL	RX_FIFO_HALF_FULL_EN	接收 FIFO 半满标志位。
RX_FIFO_HALF_EMPTY	RX_FIFO_HALF_EMPTY_EN	接收 FIFO 半空标志位。
TX_FIFO_HALF_FULL	TX_FIFO_HALF_FULL_EN	发送 FIFO 半满标志位。
TX_FIFO_HALF_EMPTY	TX_FIFO_HALF_EMPTY_EN	发送 FIFO 半空标志位。
RX_FIFO_FULL	RX_FIFO_FULL_EN	接收 FIFO 满标志位。
RX_FIFO_EMPTY	RX_FIFO_EMPTY_EN	接收 FIFO 空标志位。
TX_FIFO_FULL	TX_FIFO_FULL_EN	发送 FIFO 满标志位。
TX_FIFO_EMPTY	TX_FIFO_EMPTY_EN	发送 FIFO 空标志位。
BATCH_DONE	BATCH_DONE_EN	批量传输完成标志位。
BUSY	无	SPI 忙标志位。

详细描述见状态寄存器。

注明：SPI 模块作为主机时，批量传输完成后，BATCH_DONE 置位，SPI 模块不会再发送/接收数据。

SPI 模块作为从机发送模式时，批量传送完成后，BATCH_DONE 置位，如主机继续读取数据，从机重新开始一次新的计数，SPI 模块会继续发送数据，优先发送 FIFO 中的数据，FIFO 中数据发送为空后，发送 dummy byte。

SPI 模块作为从机接收模式时，批量传送完成后，BATCH_DONE 置位，如主机继续发送数据，从机重新开始一次新的计数，并将数据写入 FIFO 中。

在全双工模式下时，BATCH_DONE 表示批量发送和接收完成。从机双工模式下，批量发送和接收完成后，如

果主机继续传输，从机从新开始一次新的计数。从机双工模式下，批量传输完成后主机停止发送和接收。

● BUSY：

BUSY 标志由硬件置 1 和清 0（对此操作写没有任何作用）。

作为从机发送时，当 TX_FIFO 非空或者移位寄存器中有数据正在发送时，该位置一，当 TX_FIFO 中数据发送完成并且移位寄存器中数据全部发送完成后，该位清零，若从机在发送过程中，主机停止向从机发送 SCK，则从机 BUSY 将一直置一，直到 CS 被拉高后才会清除；作为从机接收时，表示 SPI 正在接收数据。在主模式下，正在发送数据或者接收数据时该位置一，当 FIFO 空且总线上无数据则清零。

如果软件要关闭 SPI 或 SPI 其他操作，可以使用 BUSY 标志检测传输是否结束以避免破坏最后一个字节传输。

● BATCH_DONE：

该位在主从模式下均有效，表示 SPI_BATCH 设定数据量的数据全部发送或接收完成，该标志位置一，若打开该位对应中断使能，将发起中断，写一清除该标志位。若使用无限数据传输该标志位及相关标志位将无效。

● RX_FIFO_FULL_OVERFLOW（从机接收 FIFO 写入溢出）：

从机接受数据时，RX_FIFO 中的数据未被系统读走或系统读取速率比接受速率慢时，导致 RX_FIFO 的数据量到达 16 字节，使得 FIFO 满状态，当下一字节写入时，FIFO 仍未有数据读走时，发生满溢出（FULL_OVERFLOW）事件。

● RX_FIFO_EMPTY_OVERFLOW(从机接收 FIFO 读出溢出标志位)：

从机 RX_FIFO 中无数据，系统读取 RX_FIFO 将会发生该溢出事件。

● INSTR_SEND_DONE（指令已发送记忆标志位）：

● 该标志位为内存映射模式下的只发送一次指令的状态标志，为一时，表示已经发送过读写指令，下次读写操作将不会再发送指令；为零时代表未发送过读写指令，下次读写操作先发送读写指令。

29.3.18. SPI 的 DMA 传输

仅在 FIFO 模式下，SPI 的 TX 和 RX 支持 DMA 功能，有其对应的 DMA 请求号。

设置 SPI_TX_CTL.TX_DMA_REQ_EN 位使能 SPI 的 DMA 发送，设置 SPI_RX_CTL.RX_DMA_REQ_EN 位使能 SPI 的 DMA 接收。设置 SPI_TX_CTL.TX_DMA_Level 位域，值为 TX DMA 请求 level。当 TX FIFO 中的数据小于等于此值时，TX DMA 请求有效。设置 SPI_RX_CTL.RX_DMA_Level 位域，值为 RX DMA 请求 level。当 RX FIFO 中的数据大于等于此值时，RX DMA 请求有效。

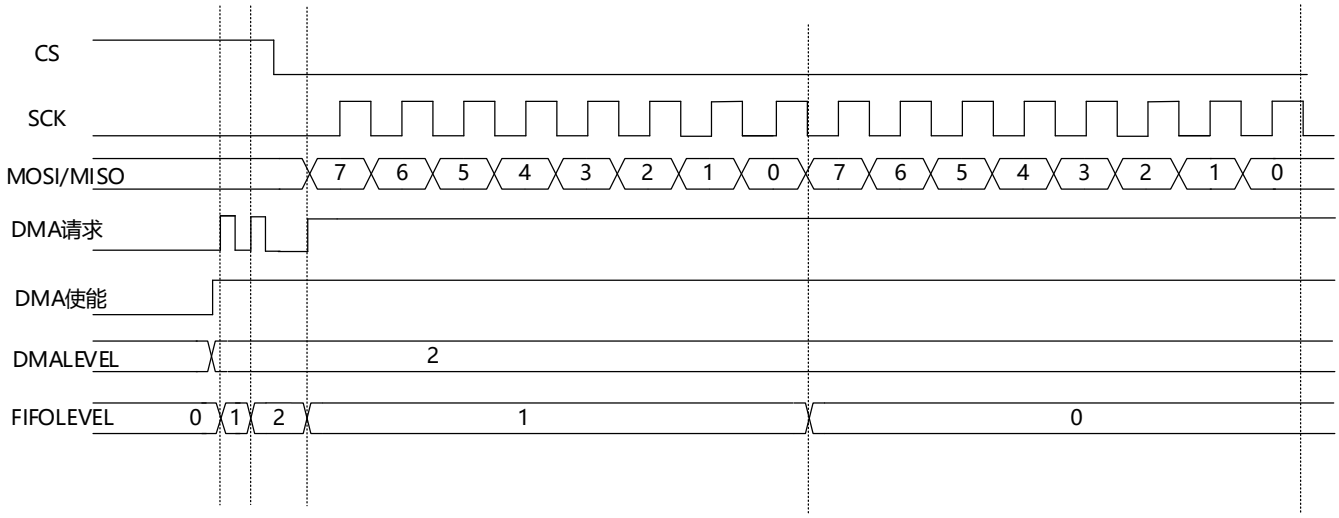
当使用 SPI_TX DMA 请求功能时，TX_FIFO 的 DMA_LEVEL 值设置的越大，SPI 的传输效率越高。例如设置 TX_DMA_Level 为 16，SPI 会一直发起 DMA 请求直至 TX_FIFO 写满，而后移位寄存器每读走一字节数据，TX_DMA 请求就发送一次，使得 SPI 发送无等待数据写入的延迟。反之，设置 TX_DMA_Level 为 1 时，SPI 初始化后发起一次请求，将数据写入 TX_FIFO 后，等 SPI 启动，移位寄存器将数据读走后，才会再次发起一次请求，如果 SPI 是四线较高频率传输，可能 SPI 发送会等待 DMA 搬运数据至 TX_FIFO 中，使得 SPI 传输效率降低。

当使用 SPI_RX DMA 请求功能时，RX_FIFO 的 DMA_LEVEL 值设置的越小，SPI 的传输效率越高。例如设置 RX_DMA_Level 为 1，当 RX_FIFO 接收一字节数据后，RX_DMA 请求立即发起，DMA 搬走数据，RX_FIFO 不会到达满状态。若 RX_DMA_Level 设置为 16，RX_FIFO 满以后才会发起 DMA 请求，若传输速率很高的情况下，若主机收数据，将会等待数据读走一字节再传输；若从机接收数据，可能会导致数据溢出，丢失数据，导致 SPI 传输效率降低。

使用 DMA 进行数据搬运时，先进行 DMA 设置，打开对应 DMA 使能，通道使能，设置 SPI 对应的 DMA 请求号等参数，再进行 SPI 初始化，SPI_TX_CTL/SPI_RX_CTL 设置对应的参数，打开 SPI DMA 使能后，根据 LEVEL 值 DMA 请求由硬件置起，DMA 开始数据搬运。如下图为 SPI_TX 的 DMA 传输，设置 DMA Transfer 为 2，搬运两次数据，SPI 初始化后，设置 SPI 的 TX_DMA_Level 为 2，设置 SPI_BATCH 为 2，打开 DMA 请求使能后 DMA 请求立即置一，传输两次后，FIFO 中有两个字节数据，拉低 CS，传输开始，当 TX 移位寄存器读走一笔数据后，DMA 再次置一，DMA 不再响应该请求。SPI 传输可以通过 BATCH_DONE 中断判断传输结

束或检测 BUSY 信号判断是否发送完成。

图 29-13 SPI TX_DMA 请求传输示意图



29.3.19. SPI Dummy 字节

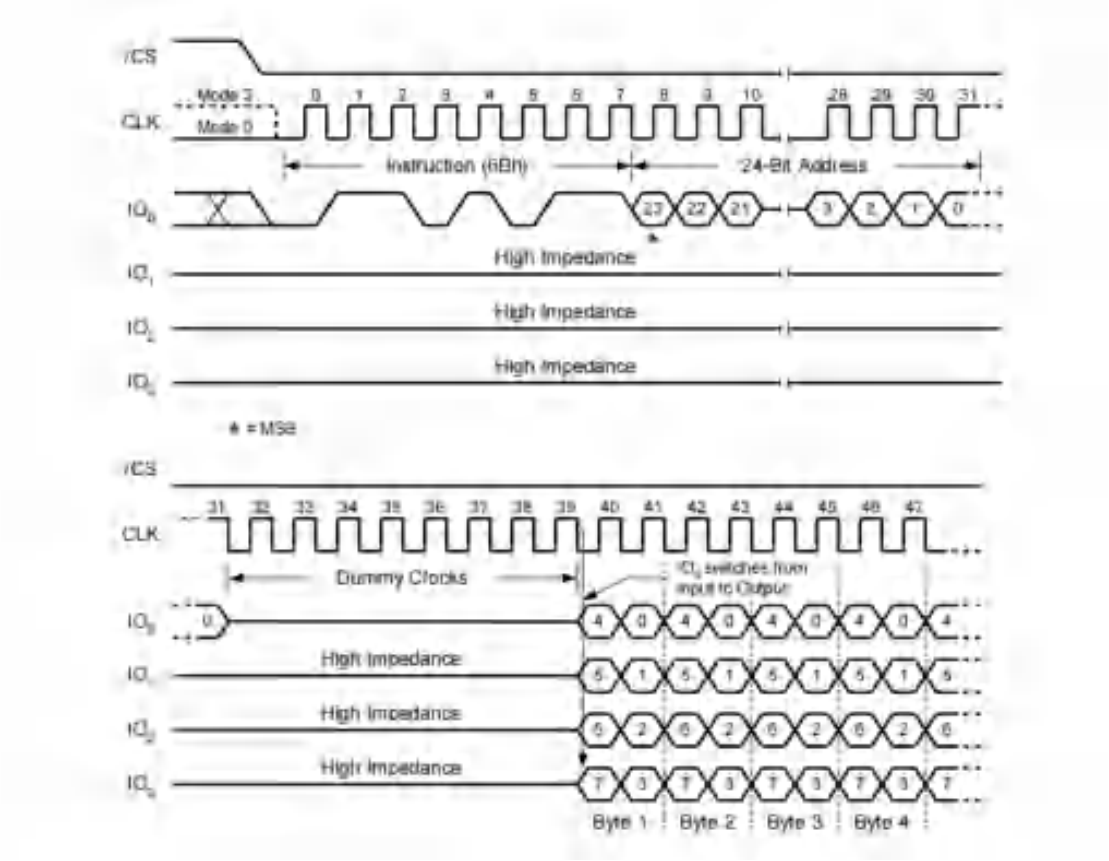
SPI 工作在主模式，当仅处于接收模式下，MOSI 数据线的状态由 Dummy 控制位决定。当 SPI 引擎会将 Dummy 数据按比特移出。

SPI 工作在从模式，当 SPI_TX_CTL.TX_MODE 为 0，SPI 发送 FIFO 为空时，FIFO 指针需要从 CPU 时钟域同步到 SPI 时钟域，1 线和 2 线模式需要一个字节进行同步，4 线模式则需要 2 个字节进行同步，当主设备发起传输时，主设备需要多接收 1 到 2 个 Dummy 字节；当 SPI_TX_CTL.TX_MODE 为 1，SPI 发送 FIFO 为空时，FIFO 指针不需要从 CPU 时钟域同步到 SPI 时钟域，当主设备发起传输时，从机中的 FIFO 有效数据可以立即发送出去，主设备不会多接收 Dummy 字节。

SPI 工作在从模式，当仅处于接收模式下，MISO 数据线的状态由 Dummy 控制位决定。SPI 引擎会将 Dummy 数据按比特移出。

29.3.20. SPI 访问存储器命令序列

SPI 通过命令与 FLASH 通信每条命令包括指令、地址、（空指令、交替字节）数据这几个阶段。 CS 在每条指令开始前下降，在每条指令完成后再次上升。下图为四线模式下的读命令示例。



● 指令阶段

这一阶段，FIFO 模式下，指令将写入 FIFO 中进行发送；在内存映射模式下，将在 SPI_CMD.Rd_Cmd/ SPI_CMD.Wr_Cmd 中配置的一条 8 位指令发送到 FLASH，通过 SPI_MEMO_ACC.Instr_Mode 可配置一、二、四线发送。

● 地址阶段

在地址阶段，将 1-4 字节发送到 FLASH，指示操作地址。在 FIFO 模式下待发送的地址字节数写入 FIFO 中进行发送。在内存映射模式下，在 SPI_MEMO_ACC.Addr_width 可配置地址长度，然后硬件通过对 AHB 总线地址进行译码直接给出地址并发送至 FLASH，通过 SPI_MEMO_ACC.Addr_Mode 可配置一、二、四线发送。

相应关系见下表：

Addr_width[1: 0]	**地址宽度**
00	8bit
01	16bit
10	24bit
11	32bit

● 空指令阶段

在 FIFO 模式下待发送的空指令字节写入 FIFO 中进行发送，空指令字节数以及发送模式根据存储器命令来进行配置。在内存映射模式下，由 SPI_MEMO_ACC.Rd_Db_EN/SPI_MEMO_ACC.Wr_Db_EN 位来控制是否需要发送读或写空指令字节，SPI_MEMO_ACC.Dummy_Cycle_Size 寄存器控制空指令长度，长度为 1-8 个 SCK，空指令周期间，信号线处于无驱动状态。

● 交替字节阶段

在 FIFO 模式下待发送的交替字节写入 FIFO 中进行发送，交替字节以及发送模式根据存储器命令来进行配置。在内存映射模式下，由 SPI_MEMO_ACC.Rd_Ab_EN/SPI_MEMO_ACC.Wr_Db_EN 位来控制是否需要发送读或写交替字节，SPI_MEMO_ACC.Alter_Byte_Size 寄存器控制交替字节长度，SPI_MEMO_ACC.Alter_Byte_Mode 寄存器可配置一、二、四线发送交替字节，SPI_ALTER_BYTE.Alter_Byte 配置需要发送的交替字节。

Alter_Byte_Size[1: 0]	交替字节宽度	交替字节（Alter_Byte[31:0]）
00	8bit	Alter_Byte[7:0]
01	16bit	Alter_Byte[15:0]
10	24bit	Alter_Byte[23:0]
11	32bit	Alter_Byte[31:0]

- 数据阶段

在数据阶段，可从 FLASH 接收或向其发送任意数量的字节。

在 FIFO 写入模式下，发送到 FLASH 的数据写入 TX_FIFO 中。在 FIFO 读取模式下，通过读取 RX_FIFO 获得从 FLASH 接收的数据。在内存映射模式下，写入或读取的数据通过 AHB 总线直接进行读写。

数据阶段可一次发送/接收 1 位 （在单线 SPI 模式中通过 IO0）、 2 位 （在双线 SPI 模式中通过 IO0/IO1）或 4 位（在四线 SPI 模式中通过 IO0/IO1/IO2/IO3）。这可通过寄存器 SPI_MEMO_ACC.Data_Mode 字段进行配置。

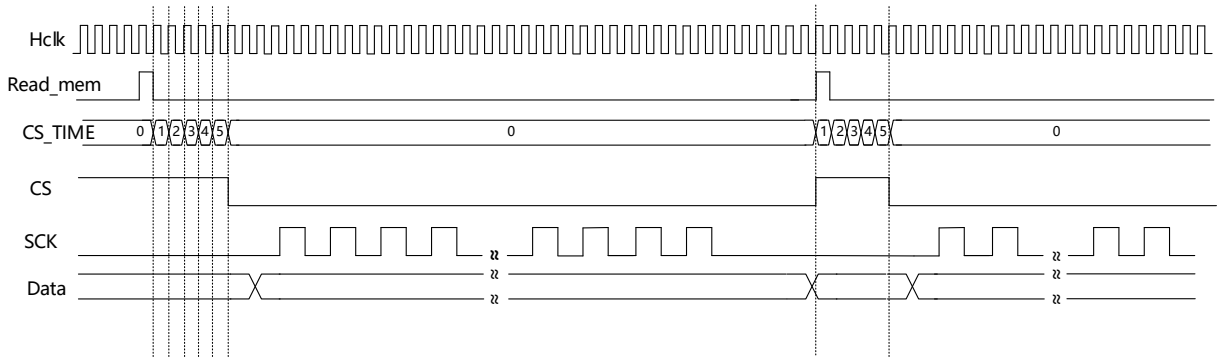
29.3.21. 内存映射模式

该 SPI 接口模块支持对满足 SPI 协议的存储器的快速读写功能，支持一线、二线、四线数据读写。当该功能开启时 MCU 可以如读写普通存储器一样通过 AHB 总线直接快速读写支持 SPI 协议的 sram 或者 nor_flash，其通过 SPI 接口当做 MCU 的附属存储器并映射到指定地址。该模式为内存映射模式。

内存映射模式的控制位由 SPI_MEMO_ACC 寄存器给出，SPI_CMD/SPI_ALTER_BYTE 为快速读写时所需的发送的参数。存储器读写序列由 29.3.20 节给出，对该序列按需配置即可实现对存储器的快速读写功能。

内存映射模式与 FIFO 模式有一部分共用寄存器，如下：

- SPI_BAUD
- SPI_CTL.CS_TIME：是指当 CS 拉高以后多少个 HCLK 之后拉低 CS 信号，该功能是为了满足存储器的在 CS 高后一些复位等功能处理时间。如下图 SPI_CTL.CS_TIME 为 5。



- SPI_CTL.LSB/SPI_CTL.CPHA/SPI_CTL.CPOL 这三位内存映射模式与 FIFO 模式共用，其用法与 FIFO 模式一致。
- SPI_RX_CTL.MSDA_EN/SPI_RX_CTL[19:11]仅在内存映射模式下使用。
- SPI_CTL.SSHIFT[2:0]在 FIFO 模式和内存映射模式下共用该寄存器，其用法与 FIFO 模式一致。

29.3.21.1. 参数配置

存储器读写序列参数配置参看 29.3.20 节中各参数介绍，详情请参见寄存器说明。需要用户自行对应存储器时序关系进行配置，以达到预期的时序。

29.3.21.2. XIP 启动

XIP(Executed In Place)本地执行，是指 CPU 直接读取存储器进行程序执行，而不用将程序先读到 RAM 中，CPU 再读取执行的方式，减少了内核从存储器将程序拷贝到 RAM 的时间。但因该方式读取程序速度较慢，会降低 CPU 运行的速率，可配合 Cache 预取来加快运行速率。

内存映射模式就是一个存储器控制器，系统通过总线读取设定的存储地址空间，就可直接读写到存储器的数据，CPU 直接运行。

29.3.21.3. 读写连续模式

打开连续模式使能位即 SPI_MEMO_ACC.Con_Mode_EN 位之后，当条件满足时，硬件会自动开启连读功能来停止发送命令、地址等，直接读写数据，可提高数据读写速率。连续的条件为：

- 读写地址为较上一次为连续值，即没有地址跳变。
- 较上一次没有进行过读写切换操作。

除了设置寄存器外，连读模式对用户操作没有影响，当连读模式启动时，用户操作等同于读取普通读取 spi 存储器。

29.3.21.4. 采样微调功能

采样微调功能指对主机数据输入信号线进行提前采样，当使能 RX_CTL. MSDA_EN 位后，配置 bit[11] - bit[14]或 bit[16] - bit[19]可以将对应设置的信号线提前半个 hclk 或一个 hclk 进行采样，将该信号线数据提前采样点一个或半个 hclk 处进行采样并保存，在采样点到来后将数据整体存入缓存 buffer 中。

该功能针对于多线模式下，线延迟不一致，使得采样点处不能正确采样全部信号线的数据，可使用该功能对延迟小的线上的数据进行提前采样进行微调，达到采样正确的效果。

eg：该功能使能：

当 SSHIFT[3:0]设置为推迟 2HCLK 时，设置提前半个 HCLK 采样，则该信号线实际采样点为推迟 1.5HCLK 采样；设置提前一个 HCLK 采样，则该信号线实际采样点为推迟 1HCLK 采样。

当 SSHIFT[3:0]设置为推迟 2.5HCLK 时，设置提前半个 HCLK 采样，则该信号线实际采样点为推迟 1.5HCLK 采样；设置提前一个 HCLK 采样，则该信号线实际采样点为推迟 2HCLK 采样。

该功能仅在内存映射模式下有效。

29.3.21.5. CS 低超时拉高功能

CS 低超时拉高功能使能是指当读写操作执行后，CS 将被拉低，并且保持为低，长时间无读写操作或者长时间为连续模式且地址连续导致 CS 一直处于拉低状态，内部有 CS 拉低计时器，当拉低时开始计数，超时后将会在空闲状态拉高 CS，当下次发生读写操作时，CS 重新拉低，并执行读写操作。

该功能通过 SPI_MEMO_ACC.CS_Tout_EN 位使能，在 SPI_CS_TOUT_VAL .CS_Tout_Val 设置超时时间，当 SPI_MEMO_ACC.CS_Tout_EN 使能位关闭时，内部超时计数器不工作。

29.3.21.6. 仅发送一次指令功能

仅发送一次指令功能是指读写操作发生后，将会发送一次指令，在后续的续写操作将不会再发送指令，无论 CS 拉高后再次拉低执行操作，也将从地址阶段开始发送，不发送指令。

该功能通过 SPI_MEMO_ACC.Send_Instr_Once_EN 位使能，在第一次读写操作发生后，发送一次指令，并会将 SPI_STATUS.INSTR_SEND_DONE 标志位置一，并保持为一，后续操作将不会发送指令。若需要发送指令可通过对 SPI_MEMO_ACC.Once_instr_clr 写一清除 SPI_STATUS.INSTR_SEND_DONE 标志位，下一次不连续操作将会重新发送指令。

29.4. 配置流程

29.4.1. SPI 主模式发送

■ 初始阶段

1）配置 SPI 控制寄存器：X_Mode、LSB_first、CPOL、CPHA、Mst_mode 比特位

2）配置波特率寄存器

3）配置发送等待寄存器

4）配置 SPI_OUT_EN 寄存器切换管脚或配置 SPI_CTL 的 IO_MODE 位为 1 硬件自动切换管脚

■ 发送阶段

1）设置 SPI 发送控制寄存器 SPI_TX_CTRL 中的 TX_EN 比特位

2）设置 SPI_BATCH 寄存器

3）通过 SPI 从设备选择寄存器，将 SPI 配置成选择状态

4）当发送 FIFO 非满时，写入待发送的数据，若 SPI_BATCH 寄存器不为零，则直到 BATCH 个数据全部发完；若 SPI_BATCH 寄存器为零，FIFO 中有数据则一直发送

5）若 SPI_BATCH 寄存器不为零，等待 SPI 状态寄存器 BATCH_DONE 状态位置位；若 SPI_BATCH 寄存器为零，跳过此步骤

6）若 SPI_BATCH 寄存器不为零，再次发送数据重复 2-5，直至 SPI 数据全部发送完成；若 SPI_BATCH 寄存器为零，跳过此步骤

7）当所有数据写入 FIFO 后，等待 SPI_STATUS.BUSY 状态位为零

8）清除 SPI_TX_CTRL 中的 TX_EN 位

9）清除 SPI 从设备选择寄存器，结束发送

29.4.2. SPI 主模式接收

■ 初始阶段

1）配置 SPI 控制寄存器：X_Mode、LSB_first、CPOL、CPHA、Mst_mode 比特位

2）配置波特率寄存器

3）配置发送等待寄存器

4）配置管脚输出使能寄存器

5）配置 SPI_OUT_EN 寄存器切换管脚或配置 SPI_CTL.IO_MODE 硬件自动切换管脚

■ 接收阶段

1）设置 SPI 接收控制寄存器 SPI_RX_CTRL 中的 RX_EN 等比特位

2）设置 SPI_BATCH 寄存器

3）通过 SPI 从设备选择寄存器，将 SPI 配置成选择状态

4）当接收 FIFO 非空时，读取接收 FIFO 中的数据

5）若 SPI_BATCH 寄存器不为零，等待 SPI 状态寄存器 BATCH_DONE 状态位置位；若 SPI_BATCH 寄存器为零，则跳过此步骤。

6）若 SPI_BATCH 寄存器不为零，再次接收数据重复 2-5，直至 SPI 数据全部接收完成；若 SPI_BATCH 寄存器为零，则跳过此步骤。

7）若 SPI_BATCH 寄存器为零，读取完所有数据

8）清除 SPI_RX_CTRL 中的 RX_EN 位

9）清除 SPI 从设备选择寄存器，结束接收

10）若 SPI_BATCH 寄存器为零的情况下完成接收，复位 RX_FIFO

29.4.3. SPI 从模式发送

■ 初始阶段

1）配置 SPI_CTL->X_Mode、LSB_first、CPOL、CPHA、SLAVE_EN 等比特位

2）配置管脚输出使能寄存器

3）配置 SPI_OUT_EN 寄存器切换管脚或配置 SPI_CTL.IO_MODE 硬件自动切换管脚

■ 发送阶段

1）设置 SPI 发送控制寄存器 SPI_TX_CTRL 中的 TX_EN、TX_MODE 比特位

2）设置 SPI_BATCH 寄存器

3）当发送 TX_FIFO 非满时，写入待发送的数据（TX_MODE 等于零，可在第五步写入数据；TX_MODE 等于一，须在 CS 低之前写入数据）

4）设置 SPI_CTL->SWCS_EN、SWCS 位（若需要）

5）当发送 TX_FIFO 非满时，写入待发送的数据，若 SPI_BATCH 寄存器不为零，直到 BATCH 个数据全部发完；若 SPI_BATCH 寄存器为零，发送完需要的数据后等待 SPI_STATUS.BUSY 状态位为零

6）清除 SPI_TX_CTRL 中的 TX_EN 位

7）结束发送

29.4.4. SPI 从模式接受

■ 初始阶段

1）配置 SPI_CTL->X_Mode、LSB_first、CPOL、CPHA、SLAVE_EN 等比特位

2）配置管脚输出使能寄存器

3）配置 SPI_OUT_EN 寄存器切换管脚或配置 SPI_CTL.IO_MODE 硬件自动切换管脚

■ 接收阶段

1）设置 SPI 接收控制寄存器 SPI_RX_CTRL 中的 RX_EN 比特位

2）设置 SPI_BATCH 寄存器

3）设置 SPI_CTL->SWCS_EN、SWCS 位（若需要）

4）当发送 RX_FIFO 非空时，读取数据，若 SPI_BATCH 寄存器不为零，则等到 BATCH 个数据全部收完；若 SPI_BATCH 寄存器为零，读取完成所有数据后等待 SPI_STATUS.BUSY 状态位为零

5）清除 SPI_RX_CTRL 中的 RX_EN 位

6）结束发送

29.4.5. SPI 存储器内存映射模式读写

AHB 总线可以直接对 SPI 储存器地址进行读写操作，硬件会自动开启对 SPI 存储器的通信，包括发送命令、地址和数据以及接收存储器发回的数据，写操作设备将 AHB 总线解析为 SPI 总线数据发送至存储器，读操作设备在通信完成后将把读取的数值通过 AHB 总线返回 MCU。

对于不同的 spi 存储器件，如有需要需按照器件的各自要求设置其状态寄存器。关于 spi 存储器的设置要求，请参照其各自 spec。其使用方法及设置步骤如下：

1）设置 SPI_CTL 寄存器的 mst_mode 位，将 SPI 接口置于主模式

2）通过 SPI_CTL 的 CPOL 和 CPHA 设置 SPI 工作模式

3）通过 SPI_BAUD 设置时钟波特率

5）在 SPI_CMD 寄存器的 Rd_Cmd/Wr_Cmd 位中写入需要发送的读命令

6）设置 SPI_ALTER_BYTE.Alter_Byte 寄存器（如果需要）

7）设置 SPI_MOME_ACC 寄存器的各个参数，参考 29.3.20 节各阶段配置

8）使能 SPI_ACC_EN 位，开启 SPI_ACC_EN 位后，硬件自动控制 SPI 接口的输入与输出，即完成读取 SPI 存储设备的准备设置

29.5. SPI 寄存器描述

29.5.1. 寄存器列表

SPI1 寄存器基地址：0x40030000

SPI2 寄存器基地址：0x40030400

SPI3 寄存器基地址：0x40030800

SPI4 寄存器基地址：0x40030C00

SPI5 寄存器基地址：0x40031000

SPI6 寄存器基地址：0x40031400

SPI7 寄存器基地址：0x52005000

SPI8 寄存器基地址：0x52005400

SPI4-MEM 内存映射空间地址：0x90000000~0x9FFFFFFF

SPI7-MEM 内存映射空间地址：0x08000000~0x0FFFFFFF

SPI8-MEM 内存映射空间地址：0x10000000~0x17FFFFFF

偏移	名称	复位值	描述
0x00	SPI_DAT	0x00000000	数据寄存器
0x04	SPI_BAUD	0x00000002	波特率设置寄存器
0x08	SPI_CTL	0x00002800	控制寄存器
0x0C	SPI_TX_CTL	0x00000000	发送控制寄存器
0x10	SPI_RX_CTL	0x00000000	接收控制寄存器
0x14	SPI_IE	0x00000000	中断控制寄存器
0x18	SPI_STATUS	0x00000154	状态寄存器

0x1C	SPI_TXDELAY	0x00000000	发送等待寄存器
0x20	SPI_BATCH	0x00000000	批量数据个数寄存器
0x24	SPI_CS	0x00000000	从设备选择寄存器
0x28	SPI_OUT_EN	0x00000000	管脚输出使能
0x2C	SPI_MEMO_ACC	0x00040000	取值控制寄存器
0x30	SPI_CMD	0x00000000	取值命令寄存器
0x34	SPI_ALTER_BYTE	0x00000000	取值交替字节寄存器
0x38	SPI_CS_TOUT_VAL	0x00000000	CS 低超时计数值

29.5.2. 数据寄存器(SPI_DAT: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DAT	RW	0x0	数据寄存器 写该寄存器，数据填入 TX_FIFO；读该寄存器，获取 RX_FIFO 数据

29.5.3. 波特率设置寄存器(SPI_BAUD: 04h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DIV2	RW	0x0	SPI 串行时钟二级分频因子
7:0	DIV1	RW	0x2	SPI 串行时钟一级分频因子。该分频因子必须是 2 到 254 之间的偶数（包括 2 和 254）。Bit[0]返回值总是为 0。

SPI 串行时钟计算公式：SPI_CLK = FHCLK / (DIV1 * (DIV2+1))。

29.5.4. 控制寄存器(SPI_CTL: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21	SWCS_EN	RW	0x0	从机 CS 软件模式选择 1：CS 软件模式，CS 电平取决于 SWCS 位 0：CS 硬件模式，CS 电平取决于 CS 引脚
20	SWCS	RW	0x0	CS 软件模式下 CS 引脚选择 该位只在 SWCS_EN 位为‘1’时有意义。它决定了 CS 上的电平，在 CS 引脚上的 I/O 操作无效。
19	RSV	-	-	保留
18:11	CS_TIME	RW	0x05	存储映射模式下 CS 高电平持续周期 （CS 高电平持续周期为寄存器值加 1，为 0 时 CS 持续一个系统时钟周期）

10	CS_FILTER	RW	0x0	从机 CS 滤毛刺选择位 1：开启从机 CS 滤毛刺功能 0：不开启从机 CS 滤毛刺功能
9	CS_RST	RW	0x0	从机 CS 复位选择位 1：CS 无效时不复位从机内部比特计数 0：CS 无效时复位从机内部比特计数 注：此寄存器不向客户开放
8	SLAVE_EN	RW	0x0	从机收发逻辑使能位 1：使能 SPI 从机功能 0：不使能 SPI 从机功能 注：默认 SPI 从模式时不使能 SPI 从机逻辑不影响 FIFO 读写，SPI 从模式初始化时，该位需要置 0，待从机 SPI 初始化完成后使能该位
7	IO_MODE	RW	0x0	IO 方向模式选择位 1：硬件自动切换 0：软件切换 注：仅 FIFO 模式下使用
6:5	X_MODE	RW	0x0	多线模式控制位 00：1X 模式 01：2X 模式 10：4X 模式 11：保留 注：FIFO 模式和内存映射模式均有效
4	LSB	RW	0x0	MSB/LSB 在前选择位 1：SPI 总线传输中 LSB 在前 0：SPI 总线传输中 MSB 在
3	CPOL	RW	0x0	时钟极性控制位 1：SCLK 低电平有效，空闲状态下为高 0：SCLK 高电平有效，空闲状态下为低
2	CPHA	RW	0x0	时钟相位控制位 1：在时钟 SCLK 的偶边沿采样数据 0：在时钟 SCLK 的奇边沿采样数据
1	SFILTER	RW	0x0	从机时钟滤毛刺选择位 1：开启从机时钟滤毛刺功能 0：不开启从机时钟滤毛刺功能 注：开启从机时钟滤毛刺功能后，48M 主频下从机 SPI 最高频率为 4M
0	MST_MODE	RW	0x0	主从模式选择位 1：SPI 工作在主模式下 0：SPI 工作在从模式下

29.5.5. 发送控制寄存器(SPI_TX_CTL: 0Ch)

位域	名称	属性	复位值	描述

31:17	RSV	-	-	保留
15:8	DUMMY	RW	0x0	无效字节寄存器
7:4	TX_DMA_LEVEL	RW	0x0	TX DMA 请求 level 当 TX FIFO 中的数据小于等于此值时，TX DMA 请求有效
3	TX_DMA_REQ_EN	RW	0x0	TX DMA 请求使能 其值为 1，且 FIFO 中的数据小于等于 TX_DMA_Level 时，发出 DMA 请求
2	TX_MODE	RW	0x0	从机发送数据选择位 1：发送 FIFO 中数据，不发送 Dummy 0：先发送 Dummy，然后发送 FIFO 中数据 置 1 不发送 Dummy 时，必须保证 SPI 作为从机发送数据时 FIFO 有数据，该位值必须在 CS 无效或 SLAVE_EN 为 0 时修改 注：该位为 0 时，四线模式下会发送两个 byte 的 Dummy 数据，在一线或二线模式下，只发送一个 byte 的 Dummy 数据。在 mode0/2 模式下，一次传输中发送的第一个 Dummy 值为零
1	TX_FIFO_RESET	RW	0x0	TX_FIFO 复位控制位 1：写 1 复位发送 FIFO 指针 0：无影响 写 1 复位有效，直到写 0 复位才会撤销
0	TX_EN	RW	0x0	发送使能位 1：TX 方向使能 0：TX 方向禁止

29.5.6. 接收控制寄存器(SPI_RX_CTL: 10h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	SSHIFT[2]	RW	0x0	详见 SSHIFT[1:0]
23:20	RSV	-	-	保留
19	MSDHA_X3	RW	0x0	spi_hold_in half 调节位(X3) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
18	MSDHA_X2	RW	0x0	spi_wp_in half 调节位(X2) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
17	MSDHA_X1	RW	0x0	spi_miso_in half 调节位(X1) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
16	MSDHA_X0	RW	0x0	spi_mosi_in half 调节位(X0) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
15	RSV	-	-	保留

14	MSDA_X3	RW	0x0	spi_hold_in 调节位(X3) 0：对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
13	MSDA_X2	RW	0x0	spi_wp_in 调节位(X2) 0：对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
12	MSDA_X1	RW	0x0	spi_miso_in 调节位(X1) 0：对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
11	MSDA_X0	RW	0x0	spi_mosi_in 调节位(X0) 0：对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
10	MSDA_EN	RW	0x0	主机采样数据调节使能位(master sample data adjust) 0：不使能 1：使能 注：当同一根线同时设置了提前半个 HCLK 和一个 HCLK 时，提前半个 HCLK 将作为采样点，优先级更高 仅 MEM 模式下有效
9:8	SSHIFT[1:0]	RW	0x0	主机采样移位控制位 000:不发生移位 001:主机推迟 1 个 hclk 周期开始采集数据 010:主机推迟 1.5 个 hclk 周期开始采集数据 011:主机推迟 2 个 hclk 周期开始采集数据 100:主机推迟 2.5 个 hclk 周期开始采集数据 101:主机推迟 3 个 hclk 周期开始采集数据 其他：保留
7:4	RX_DMA_LEVEL	RW	0x0	RX DMA 请求 level。当 RX FIFO 中的数据大于等于此值时，RX DMA 请求有效。
3	RX_DMA_REQ_EN	RW	0x0	RX DMA 请求使能 其值为 1，且当 FIFO 中的数据大于等于 RX_DMA_Level 时，发出 DMA 请求
2	RSV	-	-	保留
1	RX_FIFO_RESET	RW	0x0	RX_FIFO 复位控制位 1：写 1 复位接收 FIFO 指针 0：无影响 写 1 复位有效，直到写 0 复位才会撤销
0	RX_EN	RW	0x0	接收使能位 1：RX 方向使能 0：RX 方向禁止

29.5.7. 中断控制寄存器(SPI_IE: 14h)

位域	名称	属性	复位值	描述

31:16	RSV	-	-	保留
15	RX_BATCH_DONE_EN	RW	0x0	接收批量传输完成中断使能位 1：中断使能 0：中断禁止
14	TX_BATCH_DONE_EN	RW	0x0	发送批量传输完成中断使能位 1：中断使能 0：中断禁止
13	RX_FIFO_FULL_OVERFLOW_EN	RW	0x0	从机接收 FIFO 写溢出中断使能位 1：中断使能 0：中断禁止
12	RX_FIFO_EMPTY_OVERFLOW_EN	RW	0x0	从机接收 FIFO 读溢出中断使能位 1：中断使能 0：中断禁止
11	RX_FIFO_NOT_EMPTY_EN	RW	0x0	接收 FIFO 非空中断使能位 1：中断使能 0：中断禁止
10	CS_POS_EN	RW	0x0	CS 管脚电平上升沿事件中断使能位 1：中断使能 0：中断禁止
9	RX_FIFO_HALF_FULL_EN	RW	0x0	接收 FIFO 半满中断使能位 1：中断使能 0：中断禁止
8	RX_FIFO_HALF_EMPTY_EN	RW	0x0	接收 FIFO 半空中断使能位 1：中断使能 0：中断禁止
7	TX_FIFO_HALF_FULL_EN	RW	0x0	发送 FIFO 半满中断使能位 1：中断使能 0：中断禁止
6	TX_FIFO_HALF_EMPTY_EN	RW	0x0	发送 FIFO 半空中断使能位 1：中断使能 0：中断禁止
5	RX_FIFO_FULL_EN	RW	0x0	接收 FIFO 满中断使能位 1：中断使能 0：中断禁止
4	RX_FIFO_EMPTY_EN	RW	0x0	接收 FIFO 空中断使能位 1：中断使能 0：中断禁止
3	TX_FIFO_FULL_EN	RW	0x0	发送 FIFO 满中断使能位 1：中断使能 0：中断禁止

2	TX_FIFO_EMPTY_EN	RW	0x0	发送 FIFO 空中断使能位 1：中断使能 0：中断禁止
1	BATCH_DONE_EN	RW	0x0	批量完成中断使能位 1：中断使能 0：中断禁止
0	RSV	-	-	保留

29.5.8. 状态寄存器(SPI_STATUS: 18h)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	INSTR_SEND_DONE	RW	0x0	指令已发送记忆标志位 写 SPI_MEMO_ACC.Once_instr_clr 清除该标志位
15	RX_BATCH_DONE	RO	0x0	接收模式下批量传输完成标志位 写第 1 位 BATCH _DONE 清除该位 注：全双工模式下与 TX_ BATCH_DONE 同时产生
14	TX_ BATCH_DONE	RO	0x0	发送模式下批量传输完成标志位 写第 1 位 BATCH _DONE 清除该位
13	RX_FIFO_FULL_OVERFLOW	RO	0x0	从机接收 FIFO 写入溢出标志位 1：发生从机接收 FIFO 写入溢出 0：未发生从机接收 FIFO 写入溢出
12	RX_FIFO_EMPTY_OVERFLOW	RO	0x0	从机接收 FIFO 读出溢出标志位 1：发生从机接收 FIFO 读出溢出 0：未发生从机接收 FIFO 读出溢出
11	RX_FIFO_NOT_EMPTY	RO	0x0	接收 FIFO 非空标志位
10	CS_POS_FLG	RO	0x0	CS 管脚电平上升沿事件标志位 1:发生管脚电平上升沿事件 0：未发生事件 写 1 清除该标志位，该位用于从模式，CS 信号拉高，表示从机选中取消，结束传输
9	RX_FIFO_HALF_FULL	RO	0x0	接收 FIFO 半满标志位 1：接收 FIFO 中的字节数大于等于 8 0：接收 FIFO 中的字节数小于 8
8	RX_FIFO_HALF_EMPTY	RO	0x1	接收 FIFO 半空标志位 1：接收 FIFO 的剩余空间大于等于 8 0：接收 FIFO 中的字节数小于等于 8
7	TX_FIFO_HALF_FULL	RO	0x0	发送 FIFO 半满标志位 1：发送 FIFO 中的字节数大于等于 8 0：发送 FIFO 中的字节数小于 8

6	TX_FIFO_HALF_EMPTY	RO	0x1	发送 FIFO 半空标志位 1：发送 FIFO 的剩余空间大于等于 8 0：发送 FIFO 中的字节数小于等于 8
5	RX_FIFO_FULL	RO	0x0	接收 FIFO 满标志位 1：接收 FIFO 中满 0：接收 FIFO 未满 注：从机接收数据时，为防止数据接收溢出，可通过查询接收 FIFO 非空和接收 FIFO 半满来读数据
4	RX_FIFO_EMPTY	RO	0x1	接收 FIFO 空标志位 1：接收 FIFO 空 0：接收 FIFO 非空
3	TX_FIFO_FULL	RO	0x0	发送 FIFO 满标志位 1：发送 FIFO 中满 0：发送 FIFO 未满
2	TX_FIFO_EMPTY	RO	0x1	发送 FIFO 空标志位 1：发送 FIFO 空 0：发送 FIFO 非空 注:从机发送数据时，为从机连续发送有效数据，可通过查询发送 FIFO 半空或发送 FIFO 满标志来往 FIFO 中写数据
1	BATCH_DONE	RW	0x0	批量传输完成标志位 写 1 则清除该标志位 1：传输完成 0：传输未完成 该状态发送模式和接收模式均会产生
0	BUSY	RO	0	SPI 忙标志位 1：SPI 设备忙 0：SPI 设备空闲 硬件清 0 和置 1

29.5.9. 发送等待寄存器(SPI_TXDelay: 1Ch)

位域	名称	属性	复位值	描述
31:0	SPI_TDY	RW	0x0	SPI 每发送一个字节需要等待的时间 该控制位只在主模式下有效

29.5.10. 批量传输数据个数寄存器(SPI_BATCH : 20h)

位域	名称	属性	复位值	描述
31:0	BATCH_NUMBER	RW	0x0	批量传输量 0x0：不使用批量传输功能 其他：启用批量传输，并传输设置值数据量的数据

29.5.11. 从设备选择寄存器(SPI_CS: 24h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	CSMAP_EN	RW	0	CS 映射使能 1：使能； 0：不使能（CS1/CS2 将不可用）； CS1/CS2 映射至 IO2/IO3 管脚，仅在标准 SPI 模式下(一线模式)可用。
7:5	RSV	-	-	保留
4:0	SPI_CS	WO	0x00	片选信号 00001：CS0 控制信号（SPI_CS） 00010：CS1 控制信号（标准 spi 模式下，CS1 映射至输出 SPI_IO2，仅一线模式） 00100：CS2 控制信号（标准 spi 模式下，CS2 映射至输出 SPI_IO3，仅一线模式） 01000：CS3 控制信号（SPI_CS3） 10000：CS4 控制信号（SPI_CS4） 主模式下，置 1 将使设备选择信号 SPI_CS 变低，此位应该在配置的最后一步写入，写此位后数据传输立即开始，如果在传输的不同阶段需要改变其它寄存器的配置，那么需要重新写此位（不管此位变或不变）来触发下一次传输；从模式下，写此位无效，读 SPI_CS[0]将会获得此时 CS 电平的状态。 注：仅在 FIFO 模式下有效，内存映射模式下仅 CS0 有效。

注：从机 SPI_CLK 极性必须在 CS 有效之前的一个 SPI_CLK 周期初始化完成。

29.5.12. 管脚输出方向(SPI_OUT_EN: 28h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	SPI_HOLD_EN	RW	0x0	管脚输出使能位 1：输出模式 0：输入模式 在 IO 自动切换的时候，该位无效
2	SPI_WP_EN	RW	0x0	管脚输出使能位 1：输出模式 0：输入模式 在 IO 自动切换的时候，该位无效
1	SPI_MISO_EN	RW	0x0	管脚输出使能位 1：输出模式 0：输入模式 在 IO 自动切换的时候，该位无效
0	SPI_MOSI_EN	RW	0x0	管脚输出使能位 1：输出模式 0：输入模式 在 IO 自动切换的时候，该位无效

29.5.13. SPI 取值控制寄存器(SPI_MEMO_ACC: 2Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留位。
27	ONCE_INSTR_CLR	WO	0x0	清除已发送一次指令记忆标志位
26:25	DATA_MODE	RW	0x0	数据模式： 00：单线传输数据 01：双线传输数据 10：四线传输数据 11：保留
24:23	ALTER_BYTE_MODE	RW	0x0	交替字节模式： 00：单线传输交替字节 01：双线传输交替字节 10：四线传输交替字节 11：保留
22:21	ADDR_MODE	RW	0x0	地址模式 00：单线传输地址 01：双线传输地址 10：四线传输地址 11：保留
20:19	INSTR_MODE	RW	0x0	指令模式 00：单线传输指令 01：双线传输指令 10：四线传输指令 11：保留
18:17	ADDR_WIDTH	RW	0x2	发送地址的位数等于该寄存器值。 00 : 8 bit 01 : 16bit 10 : 24bit 11 : 32bit
16:15	RSV	-	-	保留
14:12	DUMMY_CYCLE_SIZE	RW	0x0	空指令周期长度 000：一个 SCK 周期 001：两个 SCK 周期 ... 111：八个 SCK 周期
11	RD_DB_EN	RW	0x0	读操作空指令字节使能位 1：使能空指令字节 0：不使能空指令字节
10	WR_DB_EN	RW	0x0	写操作空指令字节使能位 1：使能空指令字节 0：不使能空指令字节

9	RSV	-	-	保留
8:7	ALTER_BYTE_SIZE	RW	0x0	交替字节长度 00：8 位交替字节 01：16 位交替字节 10：24 位交替字节 11：32 位交替字节
6	RD_AB_EN	RW	0x0	读操作交替字节使能位 1：使能交替字节 0：不使能交替字节
5	WR_AB_EN	RW	0x0	写操作交替字节使能位 1：使能交替字节 0：不使能交替字节
4	SEND_INSTR_ONCE_EN	RW	0x0	仅发送一次指令使能 0：不使能 1：使能
3	CON_MODE_EN	RW	0x0	连续模式使能位。 1:使能连续模式。 0:不使能连续模式。
2	RSV	-	-	保留
1	CS_TOUT_EN	RW	0x0	CS 低超时拉高功能使能 1：使能 0：不使能 注：CS 拉低 timeout 后当前读写操作结束后拉高 CS
0	SPI_ACC_EN	RW	0x0	SPI 快速访问存储器功能块使能。 1:使能。 0:不使能，作为普通 spi 接口。

29.5.14. SPI 取值命令寄存器(SPI_CMD: 30h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留位
15:8	WR_CMD	RW	0x0	存放写 SPI 存储器的指令
7:0	RD_CMD	RW	0x0	存放读 SPI 存储器的指令

29.5.15. SPI 取值交替字节寄存器(SPI_ALTER_BYTE: 34h)

位域	名称	属性	复位值	描述
31:0	ALTER_BYTE	RW	0x0	存放要在地址后立即发送到外部 SPI 设备的可选数据。

29.5.16. CS 低超时计数值(SPI_CS_TOUT_VAL: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CS_TOUT_VAL	RW	0X0	CS 低超时拉高计数值 当 CS 拉低后到强制拉高 CS 的时间值 注：该功能不建议在写操作时使用，因为当 CS 拉高后，部分存储器编程使能将会关闭，无法继续写入。

30. 八线 SPI 接口（OSPI）

30.1. 概述

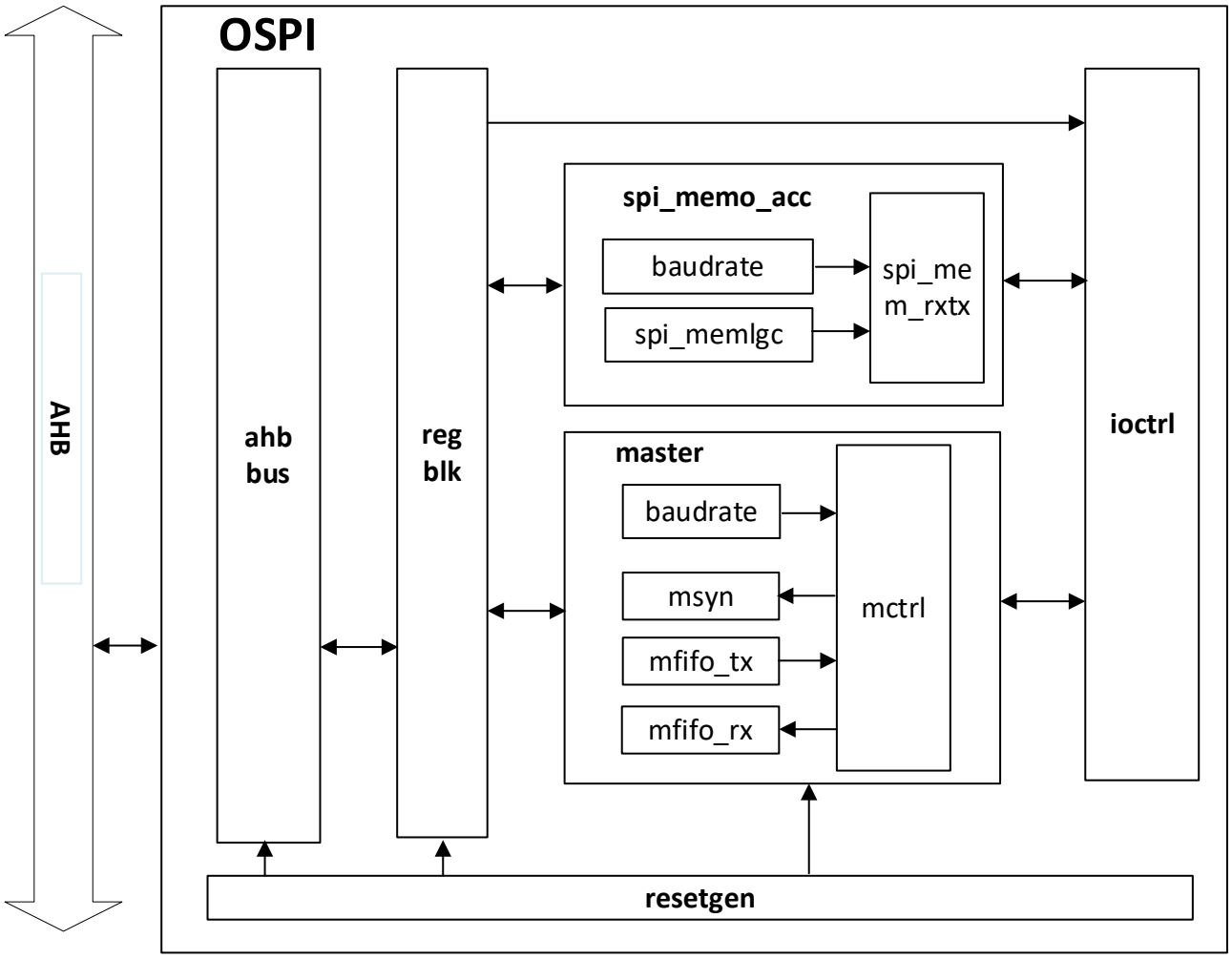
OSPI 接口模块用于微控制器（MCU）与满足 OSPI 外设之间进行全双工、全同步、串行通讯；OSPI 接口 IP 可以工作在查询或中断方式下。

30.2. 主要特性

- 功能模式：支持间接模式和内存映射模式
  - 间接模式：使用 OSPI 寄存器进行读写操作
  - 内存映射模式：通过 OSPI 作为主机将外部存储设备（如 OSPI PSRAM 等）直接映射到 MCU 的地址空间，从而可以通过直接地址访问的方式来读写该设备，OSPI 总线上的信号传输均由该 OSPI 模块内部自动转换实现
- 在内存映射模式下支持读写操作
- 可通过两级分频因子来配置宽范围波特率；
- 支持 Mode0/1/2 /3 四种传输协议（除八线 DTR 模式）
- 支持 OSPI 一线、二线、四线、八线传输
- 支持 SDR 和 DTR 模式(仅八线)
- 支持 Xccela OPI、APM OPI、HyperBus™、xOSPI 等协议
- 间接模式支持 DMA 访问
- 发送/接收数据 FIFO 大小为 16 个字节

30.3. 功能描述

30.3.1. 结构框图



如上图所示，ahb_bus 为 ahb 总线 slave 接口，reg_blk 为配置寄存器组，resetgen 产生整个 spi 模块的复位信号。spi_memo_acc1 通过硬件自动完成命令、地址以及其他参数的发送，实现对 spi 存储器的读写。spi 配置为主机时，master 产生时钟、片选信号和控制数据的收发。spi 配置为从机时。ioctrl 控制 io 的输出和输入。

30.3.2. 时序图

OSPI 模块支持一线模式和多线传输模式（二线、四线模式和八线模式）下的 IO 定义为：IO0 (MOSI)，IO1 (MISO)，IO2 (WP)，IO3 (HOLD)，IO4，IO5，IO6，IO7。

时序图如下：

30.3.2.1. 一线模式时序

图 30-1 OSPI 一线模式时序图（MSB）

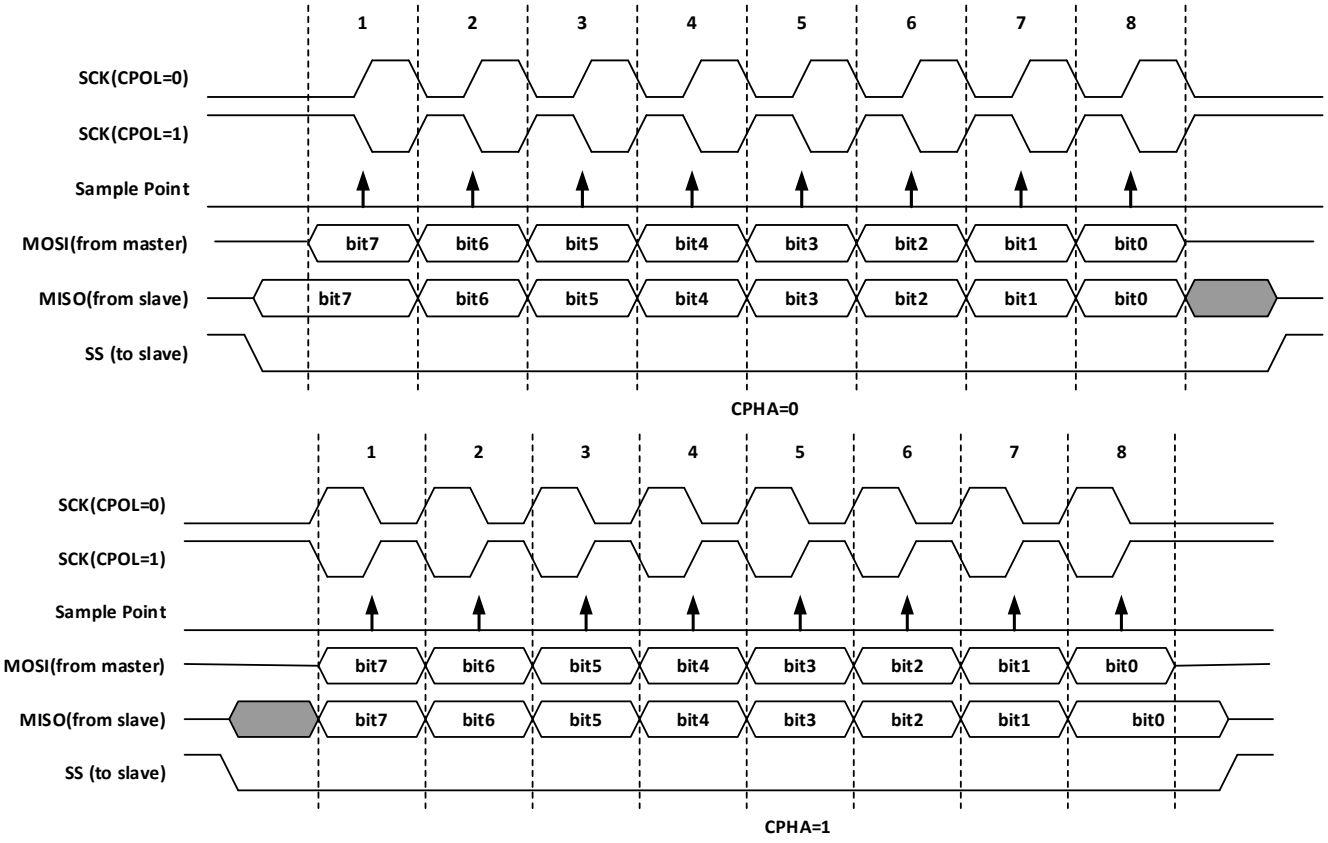
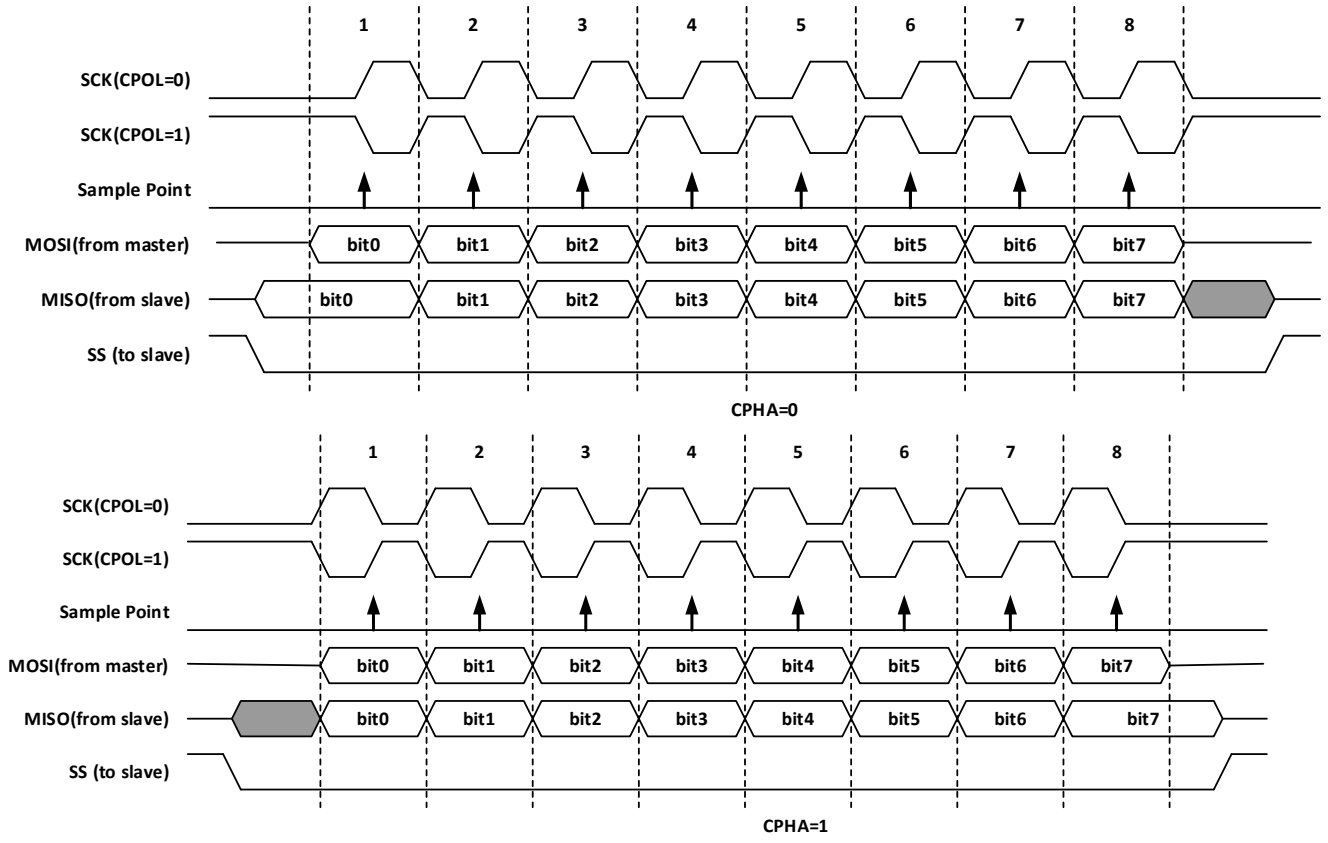


图 30-2 OSPI 一线模式时序图（LSB）



30.3.2.2. 二线模式时序

图 30-3 OSPI 二线模式时序图（MSB）

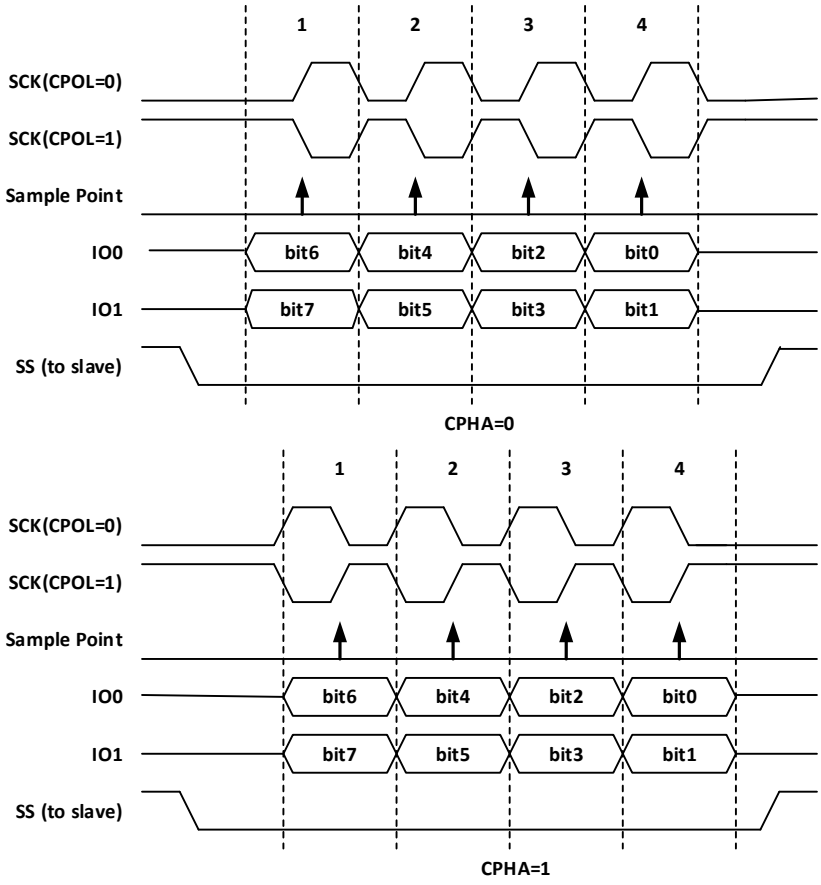
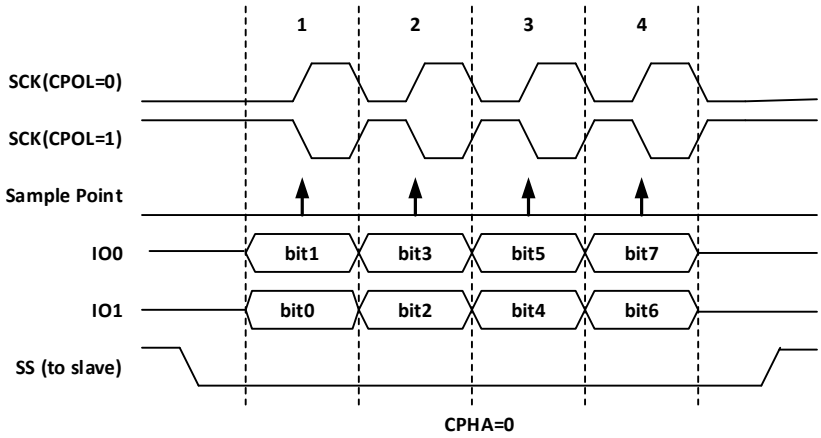
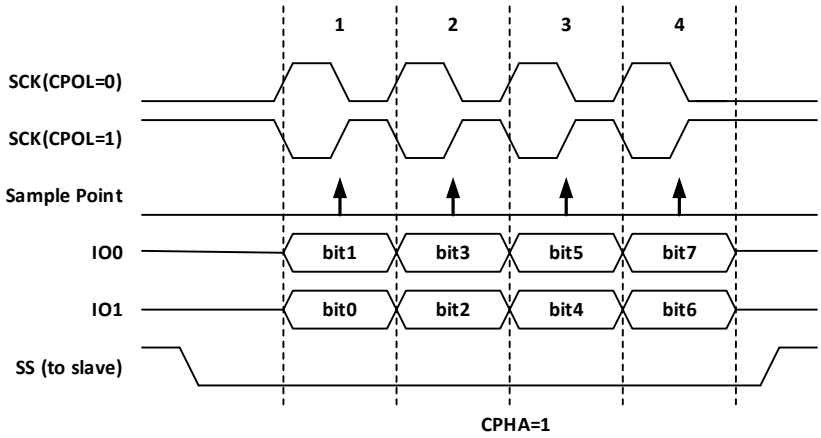


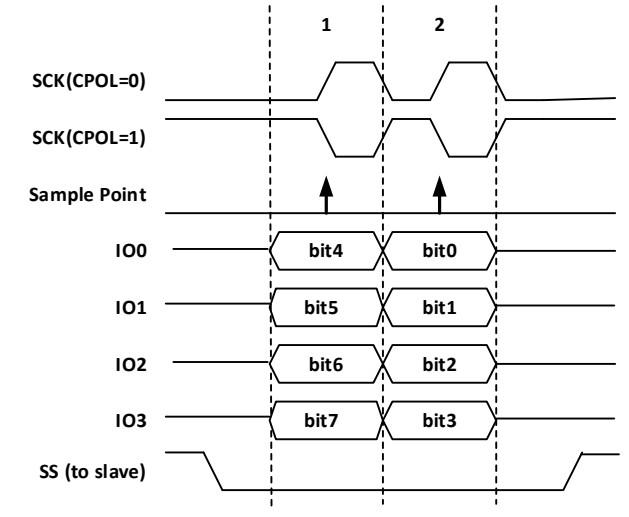
图 30-4 OSPI 二线模式时序图（LSB）





30.3.2.3. 四线模式时序

图 30-5 OSPI 四线模式时序图（MSB）



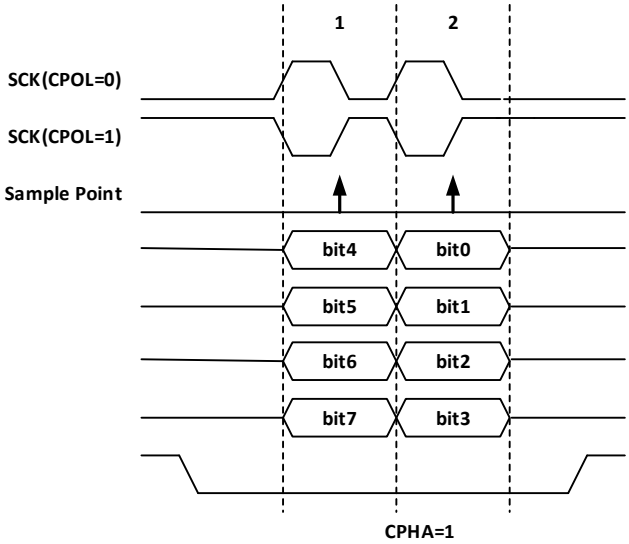
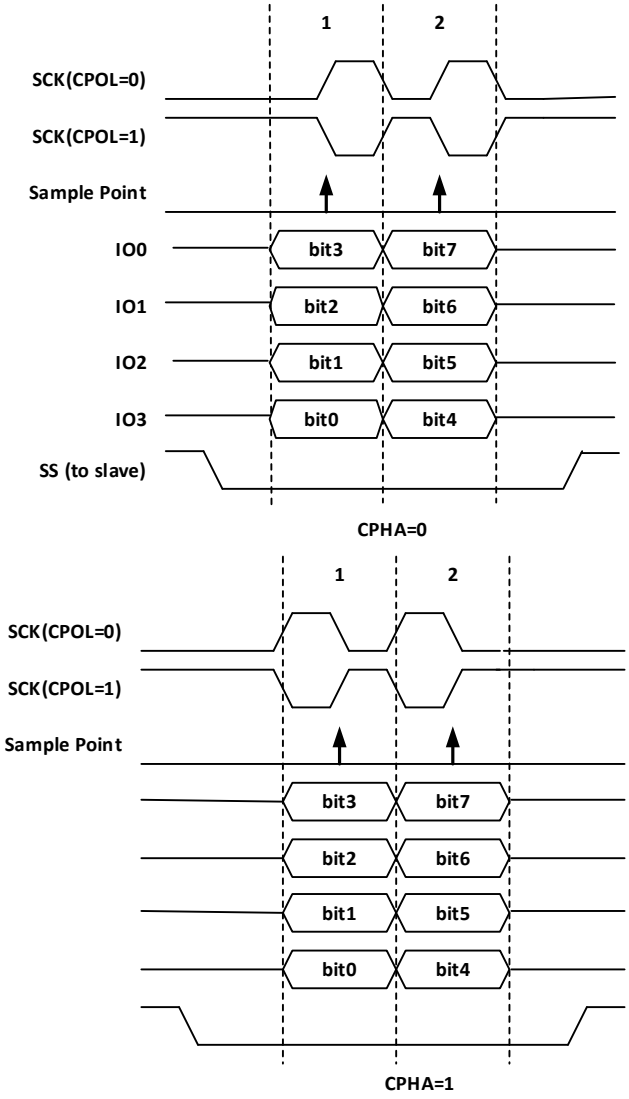


图 30-6 OSPI 四线模式时序图（LSB）



30.3.2.4. 八线模式时序

图 30-7 OSPI 八线模式时序图（MSB）

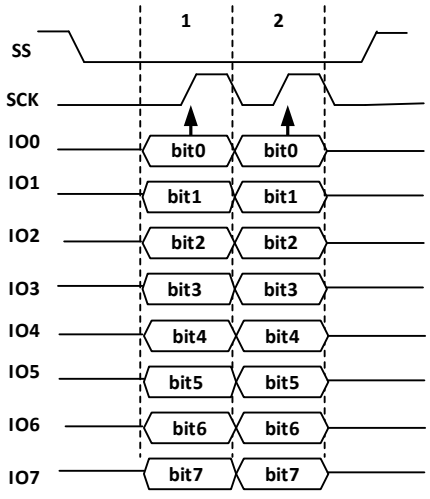
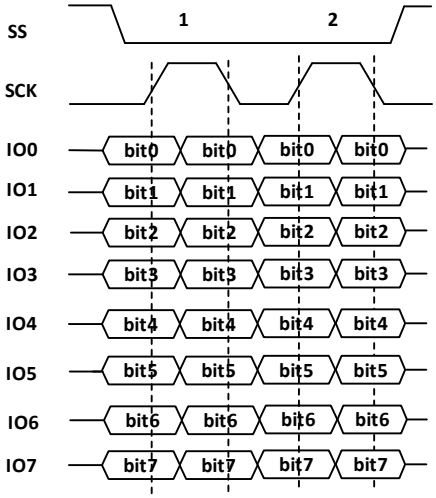


图 30-8 OSPI 八线双沿模式时序图（MSB）



30.3.3. 时钟波特率设置

OSPI 的时钟波特率设置，通过设置 OSPI_BAUD 寄存器来完成。

根据 HCLK 频率和需要设定的 OSPI 时钟频率，计算并设置 OSPI_BAUD 的 DIV2 和 DIV1。

注：分频因子 DIV1 必须是 2 到 254 之间的偶数（包括 2 和 254）。

OSPI 串行时钟计算公式:OSPI_CLK = FHCLK / (DIV1 * (DIV2+1))。

例如：在 FHCLK 为 64MHz 时，如需设置 OSPI 的时钟频率为 8MHz，可设置 DIV1 为 8，DIV2 为 0.

30.3.4. DMA 请求

每个 OSPI 接口的 TX 和 RX 都支持 DMA 功能，都有其对应的 DMA 请求号。

设置 OSPI_TX_CTL.TX_DMA_REQ_EN 位使能 OSPI 的 DMA 发送，设置 OSPI_RX_CTL.RX_DMA_REQ_EN 位使能 OSPI 的 DMA 接收。

同时还可以用 OSPI 的 FIFO 功能来发起 DMA 请求。

设置 OSPI_TX_CTL.TX_DMA_Level 位域，值为 TX DMA 请求 level。当 TX FIFO 中的数据小于等于此值时，TX DMA 请求有效。设置 OSPI_RX_CTL.RX_DMA_Level 位域，值为 RX DMA 请求 level。当 RX FIFO 中的数据大于等于此值时，RX DMA 请求有效。

30.3.5. 存储器读取模式

其中 OSPI 控制器接口模块支持对满足 OSPI 协议的存储器的快速读取功能，当该功能开启时 MCU 可以如读取普通存储器一样通过内存地址直接访问 OSPI 协议的 NOR Flash 和 OSPI PSRAM 等。

基本原理是在设定 OSPI 的基本参数如时钟波特率、工作模式、线宽模式之后，在 OSPI_CMD.Rd_Cmd 位中写入需要发送的读命令，OSPI_ALTER_BYTE 中写入交替字节（如果需要）OSPI_MEMO_ACC1 中进行配置相应的参数；最后使能 OSPI_MEMO_ACC1.OSPI_Acc_EN 位，开启 OSPI_Acc_EN 位后，硬件自动控制 OSPI 接口的输入与输出，即完成读取 OSPI 存储设备的准备设置。

AHB 总线可以直接对储存器地址进行读操作，硬件会自动开启对 OSPI 存储器的通信，包括发送命令和地址和接收存储器发回的数据，在通信完成后 OSPI 接口将把读取的数值通过 AHB 总线返回 MCU。

例如：在开启存储器读取模式后，OSPI flash 中的数据将被映射到 memory 开始的地址中，可以从该地址直接读取数据。

30.4. 配置流程

30.4.1. OSPI 主模式发送

■ 初始阶段

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode 等比特位。

2）配置波特率寄存器。

3）配置发送等待寄存器。

4）配置 OSPI_OUT_EN 寄存器切换管脚或配置 OSPI_CTL 的 IO_MODE 位为 1 硬件自动切换管脚。

■ 发送阶段

1）设置 OSPI 发送控制寄存器 OSPI_TX_CTRL 中的 TX_EN 比特位。

2）设置 OSPI_BATCH 寄存器。

3）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

4）当发送 FIFO 非满时，写入待发送的数据，直到 BATCH 个数据全部发完。

5）等待 OSPI 状态寄存器 BATCH_DONE 状态位置位。

6）再次发送数据重复 2-5，直至 OSPI 数据全部发送完成。

7）清除 OSPI_TX_CTRL 中的 TX_EN 位。

8）清除 OSPI 从设备选择寄存器，结束发送。

■ OSPI DTR 写 PSRAM

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode、存储器类型等比特位。

2）配置波特率寄存器。

3）设置 OSPI_BATCH 寄存器，设置 OSPI 发送控制寄存器 OSPI_TX_CTRL 中的 TX_EN 等比特位。

4）向 FIFO 中填入发送的指令地址及空周期字节数据。

5）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

6）等待发送完成。

7）再次设置 OSPI_BATCH 寄存器，再次拉低一下 CS 启动传输。

8）向 FIFO 中填入发送的数据。

9）等待 OSPI 状态寄存器 BATCH_DONE 状态位置位或 TX_BUSY 状态位为零，拉高 CS。

10）存储器为 Xccela OPI 协议存储时，直接结束发送；存储器为 APM OPI 协议存储时，等待状态 APM_DUMY_DONE 状态位置一，再清除后结束发送(TX_EN 关闭)。

■ Hyperbus 协议存储器和 xOSPI 协议存储器发送

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode、存储器类型等比特位。

2）配置波特率寄存器。

3）设置 OSPI_BATCH 寄存器，设置 OSPI 发送控制寄存器 OSPI_TX_CTRL 中的 TX_EN 等比特位。

4）向 FIFO 中填入发送的 CA 数据。

5）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

6）等待发送完成。

7）读取状态寄存器中的 RWDS 位，以配置 Latency Count，RWDS=1，则 LC=7；RWDS=，则 LC=3（根据存储器的 LC 数来设置，并非值 3/7）。

8）设置 OSPI_BATCH 寄存器等于 LC*2；写入 FIFO 对应数量的 dummy 数据，再次拉低一下 CS 启动传输，这两次操作可调换先后顺序。

9）等待发送完成。

10）设置 OSPI_BATCH 寄存器为发送的数据长度、DQSE(按需求)、RWDS_OE，再次拉低一下 CS 启动传输，FIFO 中写入数据。

11）等待发送完成，结束发送。

30.4.2. OSPI 主模式接收

■ 初始阶段

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode 比特位。

2）配置波特率寄存器。

3）配置发送等待寄存器。

4）配置管脚输出使能寄存器。

5）配置 OSPI_OUT_EN 寄存器切换管脚或配置 OSPI_CTL.IO_MODE 硬件自动切换管脚

■ 接收阶段

1）设置 OSPI 发送控制寄存器 OSPI_RX_CTRL 中的 RX_EN 比特位。

2）设置 OSPI_BATCH 寄存器。

3）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

4）当接收 FIFO 非空时，读取接收 FIFO 中的数据，直到 BATCH 个数据全部收完。

5）等待 OSPI 状态寄存器 BATCH_DONE 状态位置位。

6）再次接收数据重复 2-5，直至 OSPI 数据全部接收完成。

7）清除 OSPI_RX_CTRL 中的 RX_EN 位。

8）清除 OSPI 从设备选择寄存器，结束接收。

■ OSPI DTR 读 PSRAM

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode、存储器类型等比特位。

2）配置波特率寄存器。

3）设置 OSPI_BATCH 寄存器，设置 OSPI 发送控制寄存器 OSPI_TX_CTRL 中的 TX_EN 等比特位。

4）向 FIFO 中填入发送的指令地址及空周期字节数据。

5）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

6）等待发送完成。关闭 OSPI_TX_CTRL 中的 TX_EN 比特位。

7）设置 OSPI 发送控制寄存器 OSPI_RX_CTRL 中的 RX_EN、移位采样比特位。

8）设置 OSPI_BATCH 寄存器。

9）通过 OSPI 从设备选择寄存器，再次启动 OSPI 接受数据。

10）等待 OSPI 状态寄存器 BATCH_DONE 状态位置位，拉高 CS。

10）存储器为 Xccela OPI 协议存储时，直接结束传输；存储器为 APM OPI 协议存储时，等待状态 APM_DUMY_DONE 状态位置一，再清除后结束传输。

■ **Hyperbus 协议存储器和 xOSPI 协议存储器接收**

1）配置 OSPI 控制寄存器。X_Mode、LSB_first、CPOL、CPHA、Mst_mode、存储器类型等比特位。

2）配置波特率寄存器。

3）设置 OSPI_BATCH 寄存器，设置 OSPI 发送控制寄存器 OSPI_TX_CTRL 中的 TX_EN 等比特位。

4）向 FIFO 中填入发送的 CA 数据。

5）通过 OSPI 从设备选择寄存器，将 OSPI 配置成选择状态。

6）等待发送完成。

7）读取状态寄存器中的 RWDS 位，以配置 Latency Count，RWDS=1，则 LC=7；RWDS=0，则 LC=3，（根据存储器的 LC 数来设置，并非值 3/7）。

8）设置 OSPI_BATCH 寄存器等于 LC*2；写入 FIFO 对应的 dummy 数据，再次拉低一下 CS 启动传输，这两次操作可调换先后顺序。

9）等待发送完成。关闭 OSPI_TX_CTRL 中的 TX_EN 比特位。

10）设置 OSPI 发送控制寄存器 OSPI_RX_CTRL 中的 RX_EN、移位采样比特位。

11）设置 OSPI_BATCH 寄存器。

12）通过 OSPI 从设备选择寄存器，再次启动 OSPI 接受数据。

13）等待 OSPI 状态寄存器 BATCH_DONE 状态位置位，拉高 CS。

等待发送完成，结束发送。

30.4.3. OSPI 主模式接收时 Dummy 控制位

OSPI 工作在主模式，当仅处于接收模式下，MOSI 数据线的状态由 Dummy 控制位决定。当 OSPI 引擎会将 Dummy 数据按比特移出。

30.4.4. OSPI 存储器读取

该 OSPI 接口模块支持对满足 OSPI 协议的存储器的快速读写功能，当该功能开启时 MCU 可以读写存储器，一般可快速读写支持 OSPI 协议的 sram 或者 nor_flash 和 OSPI 的 PSRAM 等。其使用方法及设置步骤如下：

对于不同的 spi 存储器件，如有需要需按照器件的各自要求设置其状态寄存器。关于 spi 存储器的设置要求，请参照其各自 spec。

进行读操作前，需要进行一些基本设定，通过 OSPI_BAUD 寄存器设置时钟波特率；通过 OSPI_CTL 寄存器的 CPOL 位和 CPHA 位，并设置 OSPI 工作模式 mst_mode 位等；通过 OSPI_CMD 或 OSPI_MOME_ACC2 寄存器设置需要发送的读写命令（Hperbus 模式下由硬件自动控制），再通过 OSPI_MOME_ACC 寄存器设置相应的发送配置，最后使能 OSPI_ACC_EN 位，开启 OSPI_ACC_EN 位后，硬件自动控制 OSPI 接口的输入与输出，即完成读取 OSPI 存储设备的准备设置。

AHB 总线可以直接对储存器地址进行读操作，硬件会自动开启对 OSPI 存储器的通信，包括发送命令和地址和接收存储器发回的数据，在通信完成后 OSPI 接口将把读取的数值通过 AHB 总线返回 MCU。

30.4.5. OSPI 存储器读取（连读）

启动连读使能位即 OSPI_MEMO_ACC1 寄存器的 Con_Mode_EN 位之后，当条件满足时，硬件会自动开启连读功能停止发送命令以及地址以减少读操作用时。连读的条件为：

读写地址为连续值，即没有地址跳变。

没有进行过读写操作切换。

除了设置寄存器外，连读模式对用户操作没有影响，当连读模式启动时，用户操作等同于读取普通读取 spi 存储器。

30.4.6. OSPI SRAM 写入

同时 OSPI 接口模块支持对满足 OSPI 协议的 SRAM 的写功能，其使用方法如下：

首先需要设置 OSPI_CTL 寄存器的 mst_mode 位，将 OSPI 接口置于主模式；然后通过 OSPI_BAUD 设置时钟波特率，通过 OSPI_CTL 的 CPOL 和 CPHA 设置 OSPI 工作模式；再后设置 OSPI_CTL 的 X_mode 位和 OSPI_OUT_EN 寄存器设定通信 IO。以上为 OSPI 主模块发送准备设置。

之后通过 OSPI_CMD 寄存器的 Wr_Cmd 位设置需要发送的写命令，再通过 OSPI_MOME_ACC 寄存器设置相应的发送配置，最后使能 OSPI_ACC_EN 位，开启 OSPI_ACC_EN 位后，硬件自动控制 OSPI 接口的输入与输出，即完成写 OSPI sram 的准备设置。

完成上述配置后，软件就可以直接通过 memory 地址访问的方式来读写 OSPI sram 空间了。

30.5. OSPI 寄存器描述

30.5.1. 寄存器列表

OSPI1 寄存器基地址：0x520D1400

OSPI2 寄存器基地址：0x520D1800

OSPI1-MEM 内存映射空间地址：0x80000000~0x87FFFFFF

OSPI2-MEM 内存映射空间地址：0x88000000~0x8FFFFFFF

偏移	名称	复位值	描述
0x00	OSPI_TX_DAT	0x00000000	发送数据寄存器
0x00	OSPI_RX_DAT	0x00000000	接收数据寄存器
0x04	OSPI_BAUD	0x00000002	波特率设置寄存器
0x08	OSPI_CTL	0x00002800	控制寄存器
0x0C	OSPI_TX_CTL	0x00010000	发送控制寄存器
0x10	OSPI_RX_CTL	0x00000000	接收控制寄存器
0x14	OSPI_IE	0x00000000	中断控制寄存器
0x18	OSPI_STATUS	0x00000154	状态寄存器
0x1C	OSPI_TXDelay	0x00000000	发送等待寄存器
0x20	OSPI_BATCH	0x00000000	批量数据个数寄存器
0x24	OSPI_CS	0x00000001	从设备选择寄存器
0x28	OSPI_OUT_EN	0x00000000	管脚输出使能
0x2C	OSPI_MEMO_ACC1	0x00040000	OSPI 取值控制寄存器 1

0x30	OSPI_CMD	0x00000000	OSPI 取值命令寄存器
0x34	OSPI_ALTER_BYTE	0x00000000	OSPI 取值交替字节寄存器
0x38	OSPI_CS_TOUT_VAL	0x00000000	CS 低超时计数值
0x3C	OSPI_MEMO_ACC2	0x00000000	OSPI 取值控制寄存器 2

30.5.2. 发送数据寄存器(OSPI_TX_DAT: 00h)

位域	名称	属性	复位值	描述
31:8	TX_DAT	WO	0x0	发送数据寄存器 当 FW_MODE 为字节写时，这些比特位写值无效；当 FW_MODE 不为字节写时，FIFO 以字或半字的方式写入，再以字节进行保存。例如，字方式写一次，FIFO 写指针增加四。 注：此功能主要用于 DTR 模式
7:0	TX_DAT	WO	0x0	发送数据寄存器 该寄存器只写，读操作返回 OSPI_RX_DAT 的值。

注：当发送字节(batch_number)等于 2 时，FIFO 中应预先填入 2 字节数据，再拉低 CS；当发送字节(batch_number)大于等于四时，建议开启 FW_EN 功能，保证传输过程中软件写入的速率。

30.5.3. 接收数据寄存器(OSPI_RX_DAT: 00h)

位域	名称	属性	复位值	描述
31:8	RX_DAT	RO	0x0	接收数据寄存器 当 FR_MODE 为字节读时，这些比特位写值无效；当 FR_MODE 不为字节读时，FIFO 以字或半字的方式读取数据。 注：此功能主要用于 DTR 模式
7:0	RX_DAT	RO	0x0	接收数据寄存器 该寄存器只读，写操作改变 OSPI_TX_DAT 的值。

30.5.4. 波特率设置寄存器(OSPI_BAUD: 04h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DIV2	RW	0x0	OSPI 串行时钟二级分频因子。
7:0	DIV1	RW	0x2	OSPI 串行时钟一级分频因子。该分频因子必须是 2 到 254 之间的偶数（包括 2 和 254）。Bit[0]返回值总是为 0。

OSPI 串行时钟:OSPI_CLK = FHCLK / (DIV1 * (DIV2+1))

30.5.5. 控制寄存器(OSPI_CTL: 08h)

位域	名称	属性	复位值	描述
31:30	FR_MODE	RW	0x0	FIFO 读模式 00：FIFO 字节读 01：FIFO 半字读 10：FIFO 字读 11：保留 注：当使用 DTR 模式时，需要读取的数据量超过 16 字节时应使用半字或字读的方式来读取数据，并且使用满或半满状态判断 FIFO 空间。低于 16 字节时使用常规字节读取方式读取数据。
29:28	FW_MODE	RW	0x0	FIFO 写模式 00：FIFO 字节写 01：FIFO 半字写 10：FIFO 字写 11：保留 注：写 FIFO 时应通过读 FIFO 半空状态来写数据，以防止数据丢失。当使用 DTR 模式时，不可使用字节写。
27	DMCTRL	RW	0x0	数据掩码控制位(Data Mask Control) 1：屏蔽偶地址数据 0：屏蔽奇地址数据
26	DM_EN	RW	0x0	数据掩码使能位 1：数据掩码使能位 0：不使能 使能该位后，将屏蔽 DMCTRL 所设置的数据。 注：仅在 FIFO 模式下使用，且 TX_FIFO 为空时才能使能。
25:23	APMD_CLK	RW	0x0	APM OPI 协议传输结束后输出的 DUMMY CLOCK 数 000：无时钟 CLOCK 001：一个时钟 CLOCK ... 111：七个时钟 CLOCK
22:21	MEM_MODE	RW	0x0	存储器类型 00：Xccela OPI 协议存储器 01：APM OPI 协议存储器 10：Hyperbus 协议存储器 11：xOSPI 协议存储器 注：此配置在 DTR 模式下有效，仅支持存储器在 DTR 模式下进行访问。
20:19	-	RW	0x0	保留

18:11	CS_TIME	RW	0x05	存储映射模式下 CS 高电平持续周期 （CS 高电平持续周期为寄存器值加 1，为 0 时 CS 持续一个系统时钟周期） 在 APM OPI 协议存储器模式下，该位不能为零，且 CS 高电平持续时间为 tAPMD_CLK+tCS_TIME。
10	-	RW	0x0	保留
9	DQSOE	RW	0x0	数据选通输出使能位 1：DQS 输出使能（data mask 输出使能） 0：DQS 输出禁止(data mask 始终为零) 注：仅在八线模式、双倍速率传输时生效。 在 FIFO 模式下，用于发送数据时，DQS/DM 信号输出使能，在发送完指令地址等阶段字节后，发送有效数据前打开该输出使能。 在 MEM 模式下，作为 DQS/DM 信号使能，若需要使用 DQS/DM 信号，初始化时打开使能。
8	DTRM	RW	0x0	双倍传输速率模式 1：双倍传输速率模式下传输(DTR)； 0：单倍传输速率模式下传输(SDR)； 注：仅在八线模式下生效
7	IO_MODE	RW	0x0	IO 方向模式选择位 1：硬件自动切换； 0：软件切换；
6:5	X_MODE	RW	0x0	多线模式控制位 00：1X 模式； 01：2X 模式； 10：4X 模式； 11：8X 模式； 注：在 MEM 访问模式下，八线仅支持 8S-8S-8S 和 8D-8D-8D 传输。
4	LSB_FIRST	RW	0x0	MSB/LSB 在前选择位 1：OSPI 总线传输中 LSB 在前； 0：OSPI 总线传输中 MSB 在前；
3	CPOL	RW	0x0	时钟极性控制位 1：SCLK 低电平有效。空闲状态下为高； 0：SCLK 高电平有效。空闲状态下为低； 注：八线 DTR 模式下只支持 CPOL=0；CPHA=0 的模式。
2	CPHA	RW	0x0	时钟相位控制位 1：在时钟 SCLK 的偶边沿采样数据。 0：在时钟 SCLK 的奇边沿采样数据。
1	RSV	-	-	保留

0	MST_MODE	RW	0x0	主模式选择位 1：主模式使能； 0：主模式不使能；

30.5.6. 发送控制寄存器(OSPI_TX_CTL: 0Ch)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:16	OUTDLY	RW	0x01	输出延迟 在 DTR 模式通信时相对于 SCK 边沿的输出延迟，以满足保持（hold）要求。 00：输出无延迟 01：延迟二分之一个 HCLK 周期 10：延迟一个 HCLK 周期 11：延迟两个 HCLK 周期 注：仅在 DTR 模式下有效；二分频下只能延迟 1/2 个 HCLK；四分频下 1/2 或 1 个 HCLK 周期。
15:8	DUMMY	RW	0x0	无效字节寄存器
7:4	TX_DMA_LEVEL	RW	0x0	TX DMA 请求 level。当 TX FIFO 中的数据小于等于此值时，TX DMA 请求有效。
3	TX_DMA_REQ_EN	RW	0x0	TX DMA 请求使能。 其值为 1，且 FIFO 中的数据小于等于 TX_DMA_Level 时，发出 DMA 请求。
2	RSV	RW	0x0	-
1	TX_FIFO_RESET	RW	0x0	TX_FIFO 复位控制位 1：写 1 复位发送 FIFO 指针； 0：无影响； 写 1 复位有效，直到写 0 复位才会撤销。
0	TX_EN	RW	0x0	发送使能位 1：TX 方向使能； 0：TX 方向禁止；

30.5.7. 接收控制寄存器(OSPI_RX_CTL: 10h)

位域	名称	属性	复位值	描述

31:28	SSHIFT[3:0]	RW	0x0	主机采样移位控制位 0000:不发生移位 0001:主机推迟 1 个 hclk 周期开始采集数据 0010:主机推迟 1.5 个 hclk 周期开始采集数据 0011:主机推迟 2 个 hclk 周期开始采集数据 0100:主机推迟 2.5 个 hclk 周期开始采集数据 0101:主机推迟 3 个 hclk 周期开始采集数据 0110:主机推迟 3.5 个 hclk 周期开始采集数据 0111:主机推迟 4 个 hclk 周期开始采集数据 1000：主机推迟 4.5 个 hclk 周期开始采集数据 1001：主机推迟 5 个 hclk 周期开始采集数据 1010：主机推迟 5.5 个 hclk 周期开始采集数据 1011：主机推迟 6 个 hclk 周期开始采集数据 1100：主机推迟 6.5 个 hclk 周期开始采集数据 1101:主机推迟 7 个 hclk 周期开始采集数据 其他：保留 注：在 DTR 接收模式下，此控制位必须根据存储器数据有效时间来配置。当 DQS_SAMP_EN 为一时，该位应根据 dqs 的采样点来调整采样的颗粒度。 注：主机推迟 1.5/2.5...个 hclk 周期开始采样的效率与主机推迟 2/3...个 hclk 周期开始采样的效率相同。
27	RSV	-	-	保留
26	MSDHA_X7	RW	0x0	spi_io7_in half 调节位(X7) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
25	MSDHA_X6	RW	0x0	spi_io6_in half 调节位(X6) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
24	MSDHA_X5	RW	0x0	spi_io5_in half 调节位(X5) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
23	MSDHA_X4	RW	0x0	spi_io4_in half 调节位(X4) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
22	MSDHA_X3	RW	0x0	spi_hold_in half 调节位(X3) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
21	MSDA_X2	RW	0x0	spi_wp_in half 调节位(X2) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样

20	MSDHA_X1	RW	0x0	spi_miso_in half 调节位(X1) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
19	MSDHA_X0	RW	0x0	spi_mosi_in half 调节位(X0) 0: 对采样沿处的数据进行采样 1：对采样沿前半个 HCLK 处的数据进行采样
18	MSDA_X7	RW	0x0	spi_io7_in 调节位(X7) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
17	MSDA_X6	RW	0x0	spi_io6_in 调节位(X6) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
16	MSDA_X5	RW	0x0	spi_io5_in 调节位(X5) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
15	MSDA_X4	RW	0x0	spi_io4_in 调节位(X4) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
14	MSDA_X3	RW	0x0	spi_hold_in 调节位(X3) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
13	MSDA_X2	RW	0x0	spi_wp_in 调节位(X2) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
12	MSDA_X1	RW	0x0	spi_miso_in 调节位(X1) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样
11	MSDA_X0	RW	0x0	spi_mosi_in 调节位(X0) 0: 对采样沿处的数据进行采样 1：对采样沿前一个 HCLK 处的数据进行采样

10	MSDA_EN	RW	0x0	主机移位采样数据调节使能位(master sample data adjust) 0：不使能 1：使能 其值为 1 时，可通过设置 bit[11] - bit[18]与 bit[19] - bit[26]对多线模式下的数据信号进行单独调节，如在主机采样沿到来时，可选择在该采样沿前一个或半个 HCLK 处的数据进行提前采样。 注：当同一根线同时设置了提前半个 HCLK 和一个 HCLK 时，提前半个 HCLK 将作为采样点，优先级更高。 仅 MEM 模式下有效；其值为 0 时，bit[11] - bit[18]与 bit[19] - bit[26]无效。 eg：该功能使能： 当 SSHIFT[3:0]设置为推迟 2HCLK 时，设置提前半个 HCLK 采样，则该信号线实际采样点为推迟 1.5HCLK 采样；设置提前一个 HCLK 采样，则该信号线实际采样点为推迟 1HCLK 采样。 当 SSHIFT[3:0]设置为推迟 2.5HCLK 时，设置提前半个 HCLK 采样，则该信号线实际采样点为推迟 1.5HCLK 采样；设置提前一个 HCLK 采样，则该信号线实际采样点为推迟 2HCLK 采样。
9	-	-	-	保留
8	DQS_SAMP_EN	RW	0x0	DQS 采样使能 1：使用 DQS 采样 0：DQS 采样禁止 该位使能后，接收数据时，数据将会使用 DQS 作为时钟对输入数据进行采样，上升沿将数据存入 BUFF1 中，下降沿将数据存入 BUFF2 中，通过移位采样功能调节采样颗粒度对两个 BUFF 中的数据进行采样。
7:4	RX_DMA_LEVEL	RW	0x0	RX DMA 请求 level。当 RX FIFO 中的数据大于等于此值时，RX DMA 请求有效。
3	RX_DMA_REQ_EN	RW	0x0	RX DMA 请求使能。 其值为 1，且当 FIFO 中的数据大于等于 RX_DMA_Level 时，发出 DMA 请求。
2	RSV	-	-	保留
1	RX_FIFO_RESET	RW	0x0	RX_FIFO 复位控制位 1：写 1 复位接收 FIFO 指针； 0：无影响； 写 1 复位有效，直到写 0 复位才会撤销
0	RX_EN	RW	0x0	接收使能位 1：RX 方向使能； 0：RX 方向禁止；

30.5.8. 中断控制寄存器(OSPI_IE: 14h)

位域	名称	属性	复位值	描述

31:16	RSV	-	-	保留
15	RX_BATCH_DONE_EN	RW	0x0	接收批量传输完成中断使能位 1：中断使能； 0：中断禁止；
14	TX_BATCH_DONE_EN	RW	0x0	发送批量传输完成中断使能位 1：中断使能； 0：中断禁止；
13:12	RSV	RW	-	-
11	RX_FIFO_NOT_EMPTY_EN	RW	0x0	接收 FIFO 非空中断使能位 1：中断使能； 0：中断禁止；
10	RSV	RW	-	-
9	RX_FIFO_HALF_FULL_EN	RW	0x0	接收 FIFO 半满中断使能位 1：中断使能； 0：中断禁止；
8	RX_FIFO_HALF_EMPTY_EN	RW	0x0	接收 FIFO 半空中断使能位 1：中断使能； 0：中断禁止；
7	TX_FIFO_HALF_FULL_EN	RW	0x0	发送 FIFO 半满中断使能位 1：中断使能； 0：中断禁止；
6	TX_FIFO_HALF_EMPTY_EN	RW	0x0	发送 FIFO 半空中断使能位 1：中断使能； 0：中断禁止；
5	RX_FIFO_FULL_EN	RW	0x0	接收 FIFO 满中断使能位 1：中断使能； 0：中断禁止；
4	RX_FIFO_EMPTY_EN	RW	0x0	接收 FIFO 空中断使能位 1：中断使能； 0：中断禁止；
3	TX_FIFO_FULL_EN	RW	0x0	发送 FIFO 满中断使能位 1：中断使能； 0：中断禁止；
2	TX_FIFO_EMPTY_EN	RW	0x0	发送 FIFO 空中断使能位 1：中断使能； 0：中断禁止；

1	BATCH_DONE_EN	RW	0x0	批量完成中断使能位 1：中断使能； 0：中断禁止；
0	RSV	-	-	保留

30.5.9. 状态寄存器(OSPI_STATUS: 18h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18	INSTR_SEND_DONE	RO	0x0	指令已发送记忆标志位 该标志位为 MEM 模式下的只发送一次指令的状态标志，为一时，表示已经发送过读写指令，下次读写将不会发送指令；为零时代表未发送过读写指令，下次读写先发送读写指令。
17	RWDS	RO	0x0	Hyperbus 或 xOSPI 协议下的 RWDS 输入信号，用来配置 Latency Count 周期 RWDS=0：LC=3。 RWDS=1：LC=7。 根据具体情况定义 LC，该周期数为 CMD/ADDR 阶段结束后到数据开始前的周期数。
16	APM_DUMY_DONE	RW	0x0	APM OPI 协议存储器 dummy cycle 结束 注：该位由硬件置一，写一该位清零。
15	RX_BATCH_DONE	RO	0x0	接收模式下批量传输完成标志位 写第 1 位 BATCH _DONE 清除该位 注：全双工模式下与 TX_ BATCH_DONE 同时产生
14	TX_ BATCH_DONE	RO	0x0	发送模式下批量传输完成标志位 写第 1 位 BATCH _DONE 清除该位
13:12	RSV	-	-	-
11	RX_FIFO_NOT_EMPTY	RO	0x0	接收 FIFO 非空标志位 OSPI_CTL ->Mst_mod 未配置为一时，读此位为 1。
10	RSV	-	-	-
9	RX_FIFO_HALF_FULL	RO	0x0	接收 FIFO 半满标志位 1：接收 FIFO 中的字节数大于等于 8； 0：接收 FIFO 中的字节数小于 8；
8	RX_FIFO_HALF_EMPTY	RO	0x1	接收 FIFO 半空标志位 1：接收 FIFO 的剩余空间大于等于 8； 0：接收 FIFO 中的字节数小于等于 8； OSPI_CTL ->Mst_mod 未配置为一时，读此位为零。

7	TX_FIFO_HALF_FULL	RO	0x0	发送 FIFO 半满标志位 1：发送 FIFO 中的字节数大于等于 8； 0：发送 FIFO 中的字节数小于 8；
6	TX_FIFO_HALF_EMPTY	RO	0x1	发送 FIFO 半空标志位 1：发送 FIFO 的剩余空间大于等于 8； 0：发送 FIFO 中的字节数小于等于 8； OSPI_CTL ->Mst_mod 未配置为一时，读此位为零。
5	RX_FIFO_FULL	RO	0x0	接收 FIFO 满标志位 1：接收 FIFO 中满； 0：接收 FIFO 未满； 注：从机接收数据时，为防止数据接收溢出，可通过查询接收 FIFO 非空和接收 FIFO 半满来读数据。
4	RX_FIFO_EMPTY	RO	0x1	接收 FIFO 空标志位 1：接收 FIFO 空； 0：接收 FIFO 非空； OSPI_CTL ->Mst_mod 未配置为一时，读此位为零。
3	TX_FIFO_FULL	RO	0x0	发送 FIFO 满标志位 1：发送 FIFO 中满； 0：发送 FIFO 未满；
2	TX_FIFO_EMPTY	RO	0x1	发送 FIFO 空标志位 1：发送 FIFO 空； 0：发送 FIFO 非空； 注:从机发送数据时，为从机连续发送有效数据，可通过查询发送 FIFO 半空或发送 FIFO 满标志来往 FIFO 中写数据。 OSPI_CTL ->Mst_mod 未配置为一时，读此位为零。
1	BATCH_DONE	RW	0x0	批量传输完成标志位。写 1 则清除该标志位 1：传输完成； 0：传输未完成； 该状态发送模式和接收模式均会产生
0	TX_BUSY	RO	0	OSPI 忙于发送标志位 1：作从机时，OSPI 正在发送数据或送状态下发送 FIFO 不为空。作主机时，OSPI 主机正在发送数据； 0：OSPI 空闲； 硬件清 0 和置 1 从机模式下，FIFO 非空，OSPI 正在发送数据，TX_BUSY 将置 1

注明：OSPI 模块作为主机时，批量传输完成后，BATCH_DONE 置位，OSPI 模块不会再发送/接收数据。

OSPI 模块作为从机发送模式时，批量传送完成后，BATCH_DONE 置位，如主机继续读取数据，从机重新开始一次新的计数，OSPI 模块会继续发送数据，优先发送 FIFO 中的数据，FIFO 中数据发送为空后，发送 dummy byte。

OSPI 模块作为从机接收模式时,批量传送完成后，BATCH_DONE 置位，如主机继续发送数据，从机重新开始一次新的计数，并将数据写入 FIFO 中。

在全双工模式下时，BATCH_DONE 表示批量发送和接收完成。从机双工模式下，批量发送和接收完成后，如果主机继续传输，从机从新开始一次新的计数。从机双工模式下，批量传输完成后主机停止发送和接收。

TX_BUSY 标志由硬件置 1 和清 0（对此操作写没有任何作用），TX_BUSY 用于 OSPI 通信状态。TX_BUSY 置 1 时，表示 OSPI 正在发送数据。如果软件要关闭 OSPI 或 OSPI 其他操作，可以使用 TX_BUSY 标志检测传输是否结束以避免破坏最后一个字节传输。

30.5.10. 发送等待寄存器(OSPI_TXDelay: 1Ch)

位域	名称	属性	复位值	描述
31:0	OSPI_TDY	RW	0x0	OSPI 每发送一个字节需要等待 HCLK 计数。该控制位只在主模式下有效。

注：八线和八线 DTR 模式下不支持该功能。

在发送等待的过程中，CLK 将按照 CPOL 设定值停止。

30.5.11. 批量传输数据个数寄存器(OSPI_BATCH: 20h)

位域	名称	属性	复位值	描述
31:0	BATCH_NUMBER	RW	0x0	该寄存器用来存储 OSPI 总线上即将传输的数据字节个数。

注：DTR 模式下，此寄存器应设置为偶数。

在 DTR 模式下，一个时钟将会传输两个字节数据，因此对软件和硬件的配合非常重要， 在发送时，若未及时将数据写入 FIFO，将会导致发送 DUMMY 数据，同时 DUMMY 数据会被硬件计为有效数据，Batch_number 将会将 DUMMY 数计入计数器值。

30.5.12. 从设备选择寄存器(OSPI_CS: 24h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	CSMAP_EN	W/R	0	CS 映射使能 1：使能； 0：不使能（CS1/CS2 将不可用）； CS1/CS2 映射至 IO2/IO3 管脚，仅在标准 OSPI 模式下(一线模式)可用。
2:0	OSPI_CS	WO	0x1	置 1 将使设备选择信号 OSPI_CS[2:0]变低。主模式下此位应该在配置的最后一步写入，写此位后数据传输立即开始。如果主模式在传输的不同阶段需要改变其它寄存器的配置，即么需要重新写此位（不管此位变或不变）来触发下一次传输。 001：CS0 控制信号（OSPI_CS） 010：CS1 控制信号（标准 spi 模式下，CS1 映射至输出 OSPI_IO2） 100：CS2 控制信号（标准 spi 模式下，CS2 映射至输出 OSPI_IO3） 注：此位不可读，若读将会得到不确定值。

30.5.13. 管脚输出方向(OSPI_OUT_EN: 28h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	OSPI_IO7_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式；0 在 IO 自动切换的时候，该位无效
6	OSPI_IO6_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式；0 在 IO 自动切换的时候，该位无效
5	OSPI_IO5_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式；0 在 IO 自动切换的时候，该位无效
4	OSPI_IO4_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式；0 在 IO 自动切换的时候，该位无效
3	OSPI_HOLD_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式；0 在 IO 自动切换的时候，该位无效
2	OSPI_WP_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式； 在 IO 自动切换的时候，该位无效
1	OSPI_MISO_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式； 在 IO 自动切换的时候，该位无效
0	OSPI_MOSI_EN	RW	0x0	管脚输出使能位 1：输出模式； 0：输入模式； 在 IO 自动切换的时候，该位无效

30.5.14. 取值控制寄存器 1(OSPI_MEMO_ACC1: 2Ch)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留位。
29	ONCE_INSTR_CLR	WO	0x0	清除已发送一次指令记忆标志位
28	RSV	RW	0x0	该位需保持为零
27	HYPER_BT	RW	0x0	Hyperbus Burst Type (CA[45]) 表明突发是 Linear 的还是 Wrapped 的 1：Linear 突发 0：wrapped 突发
26:25	DATA_MODE	RW	0x0	数据模式： 00：单线传输数据 01：双线传输数据 10：四线传输数据 11：保留 当控制寄存器中配置为八线模式时，此寄存器无效。
24:23	ALTER_BYTE_MODE	RW	0x0	交替字节模式： 00：单线传输交替字节 01：双线传输交替字节 10：四线传输交替字节 11：保留 当控制寄存器中配置为八线模式时，此寄存器无效。
22:21	ADDR_MODE	RW	0x0	地址模式 00：单线传输地址 01：双线传输地址 10：四线传输地址 11：保留 当控制寄存器中配置为八线模式时，此寄存器无效。
20:19	INSTR_MODE	RW	0x0	指令模式 00：单线传输指令 01：双线传输指令 10：四线传输指令 11：保留 当控制寄存器中配置为八线模式时，此寄存器无效。

18:17	ADDR_WIDTH	RW	0x2	发送地址的位数等于该寄存器值。 00 : 8 bit 01 : 16bit 10 : 24bit 11 : 32bit Hyperbus 模式时，地址位数必须配置为 32bits.
16:12	DUMMY_CYCLE_SIZE	RW	0x0	空指令周期长度 00000：一个 SCK 周期 00001：两个 SCK 周期 ... 11111：三十二个 SCK 周期 Hyperbus 或 xOSPI 模式时该位无效，由 Hxspi_LC0/Hxspi_LC1 配置。
11	RD_DB_EN	RW	0x0	读操作空指令字节使能位 1：使能空指令字节 0：不使能空指令字节 Hyperbus 模式时该位按需置一。
10	WR_DB_EN	RW	0x0	写操作空指令字节使能位 1：使能空指令字节 0：不使能空指令字节 Hyperbus 模式时该位按需置一。
9	RSV	RW	0x0	保留
8:7	ALTER_BYTE_SIZE	RW	0x0	交替字节长度 00：8 位交替字节 01：16 位交替字节 10：24 位交替字节 11：32 位交替字节 当作为八线模式传输时，此寄存器无效。
6	RD_AB_EN	RW	0x0	读操作交替字节使能位 1：使能交替字节 0：不使能交替字节 当作为八线模式传输时，此寄存器无效。
5	WR_AB_EN	RW	0x0	写操作交替字节使能位 1：使能交替字节 0：不使能交替字节 当作为八线模式传输时，此寄存器无效。

4	SEND_INSTR_ONCE_EN	RW	0x0	仅发送一次指令使能 0：不使能 1：使能 注：该位使能后，只会发送一次读写指令，并会将 OSPI_STATUS.INSTR_SEND_DONE 标志位置一，若需要重新发送一次指令，软件对 Once_instr_clr 位写一，清除已发送一次指令记忆标志，即可在下一次读写操作中发送一次指令。
3	CON_MODE_EN	RW	0x0	连续模式使能位。 1:使能连续模式。 0:不使能连续模式。
2	RSV	RW	0x0	保留
1	CS_TOUT_EN	RW	0x0	CS 低超时拉高功能使能 1：使能 0：不使能 注：CS 拉低 timeout 后当前读写操作结束后拉高 CS
0	OSPI_ACC_EN	RW	0x0	OSPI 快速访问存储器功能块使能。 1:使能。 0:不使能，作为普通 spi 接口。

30.5.15. 取值命令寄存器(OSPI_CMD: 30h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留位。
15:8	WR_CMD	RW	0x0	存放写 OSPI 存储器的指令。 Hyperbus 模式下自动控制。
7:0	RD_CMD	RW	0x0	存放读 OSPI 存储器的指令。 Hyperbus 模式下自动控制。

30.5.16. 取值交替字节寄存器(OSPI_ALTER_BYTE: 34h)

位域	名称	属性	复位值	描述
31:0	ALTER_BYTE	RW	0x0	存放要在地址后立即发送到外部 OSPI 设备的可选数据。

30.5.17. CS 低超时计数值(OSPI_CS_TOUT_VAL: 38h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留

15:0	CS_TOUT__VAL	RW	0X0	CS 低超时拉高计数值 当 CS 拉低后到强制拉高 CS 的时间值 注：该功能不建议在写存储器时使用

30.5.18. 取值控制寄存器 2(OSPI_MEMO_ACC2: 3Ch)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14:12	WRPS	RW	0x0	回卷大小（Wrap Size） 该配置大小对应设置的存储器回卷长度 000：存储器已禁用回卷模式或 OSPI 控制器不使用回卷读写 001：保留 010：外部存储器支持 16 字节的回卷大小 011：外部存储器支持 32 字节的回卷大小 100：外部存储器支持 64 字节的回卷大小 101-111：保留，必须保持复位值
11:8	BL	RW	0x0	突发长度（Burst Length） 000：读写不跨边界或存储器无边界，按存储器 Linear Burst 或 Wrap Burst 格式读写。 001：16 byte 010：32 byte 011：64 byte 100：128 byte 101：256 byte 110：512 byte 111：1K byte 其他：保留（禁止写入） 该值应与存储器 Burst Length 或跨越边界保持一致，连续模式下读写时，地址若跨越该长度空间，将会重新发送一次命令地址，以保证存储器数据是连续的。 注：该位一般用在 Linear Burst 传输跨边界使用，在传输前，根据存储器类型打开存储器 Linear Burst 功能；若在 Wrap Burst 或 Hybrid Wrap 传输类型下使用，会降低传输效率，速度变慢。 例：PSRAM Burst Length 为 1K Byte，Linear Burst 读写完 1K 数据后需要重新发起命令地址。
7:4	HXOSPI_LC1	RW	0x0	Hyperbus 或 xOSPI 模式下，RWDS 为一时的 LC 周期数 0000：一个 SCK 周期 0001：两个 SCK 周期 ... 1111：十六个 SCK 周期 该周期数为 CA[47:0]阶段结束后到数据开始前的周期数。

3:0	HXOSPI_LC0	RW	0x0	Hyperbus 或 xOSPI 模式下，RWDS 为零时的 LC 周期数 0000：一个 SCK 周期 0001：两个 SCK 周期 ... 1111：十六个 SCK 周期 该周期数为 CA[47:0]阶段结束后到数据开始前的周期数。

31. 通用异步收发器（UART）

31.1. 概述

通用异步收发器（UART）能够灵活地与外部设备进行全双工数据交换。芯片上集成了 10 个 UART 串口模块。UART 部分除了支持常用的 UART 功能外，还可以支持 LIN 总线、IrDA SIR、智能卡（主模式）、单线半双工模式、多机通信、RS485 等功能。

31.2. 主要特性

31.2.1. UART 的基本功能特性

- 全双工异步通信
- 支持 CTS，RTS 流控制
- 16 字节的硬件收发 FIFO（其中 UART3 和 UART6 支持 64 字节的接收 FIFO）
- 波特率支持整数和小数分频
- 总线空闲检测
- 可编程字长（5~9bit）
- 可编程校验奇偶或 0/1 校验
- 可编程停止位个数

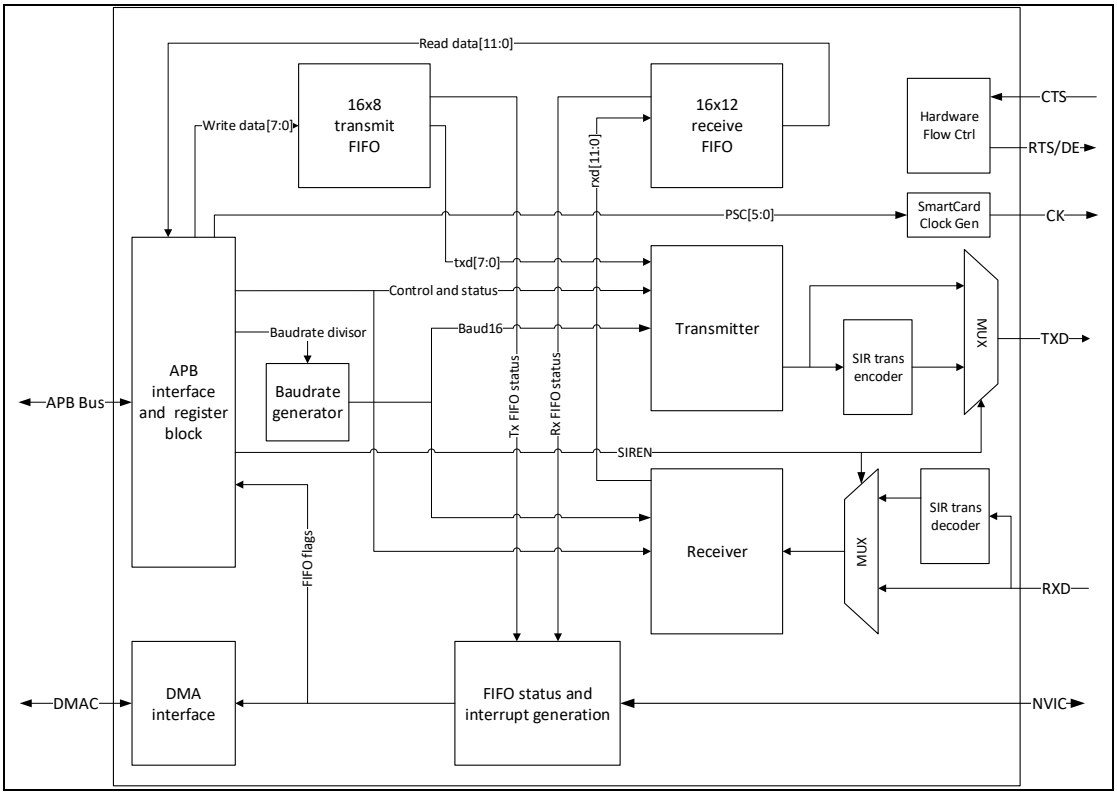
31.2.2. UART 的扩展功能特性

- 支持多机通信功能（非地址匹配节点则进入静默模式）
- 支持 RS485 通信（硬件实现 RS485 收发器 DE 控制）
- 支持 7816 主机模式
- 支持单线模式-只使用 TX 脚进行数据收发
- 支持波特率自适应功能
- 支持波特率计数功能
- 支持 LIN
- 支持 IrDA
- 支持同步模式

31.3. 功能描述

31.3.1. 结构框图

图 31-1 UART 结构框图



31.3.2. UART 信号

■ UART 全双工通信

➢ TX：数据发送

要发送的数据通过 UART 的 TX 管脚进行发送，当没有数据要发送时，TX 管脚为高电平。在 UART 单线模式或智能卡（主机）模式下，TX 被用来同时发送和接收数据，因此当 UART 处在这两种模式下时，为单工模式。

➢ RX：数据接收

UART 通过 RX 管脚来接收数据（除单线模式和智能卡模式外）

■ RS232 硬件流控模式

➢ CTS：（明确是否可以发送）

当该信号为高电平时，将在当前数据传输结束后阻断下一次的数据发送。

➢ RTS：（请求对方发送）

当该信号为低电平时，表明 UART 已准备好接收数据

■ RS485 通信

➢ DE：驱动输出使能

该信号用于控制 RS485 收发器的输入/输出使能。

■ **智能卡模式**

➢ CK：时钟信号

当 UART 处在智能卡（主机）模式下， CK 脚输出时钟信号提供给卡片。

31.3.3. UART 帧字符结构

通过配置 UART_CR3. WLEN，可以将 UART 的字长配置成 5，6，7，8，9 bit（详见寄存器 UART_CR3）。

31.3.4. UART FIFO 及触发阈值

UART FIFO 分为发送 FIFO（以下称为 TXFIFO）和接收 FIFO（以下称为 RXFIFO），均含有 16 个字节。通过将 UART_CR3.FEN 置 1，使能 UART FIFO 功能。

当开启了 UART FIFO 功能后，UART 的 TXI 和 RXI 中断不再是发送和接收一个字节就产生，而是会根据设置的 FIFO 触发阈值而产生。TXFIFO 的触发阈值配置见 UART_CR3. TXIFLSEL，RXFIFO 的触发阈值见 UART_CR3. RXIFLSEL。

需要注意的是，中断产生不是单纯由 FIFO 中的数据数量决定，触发仅产生于特定操作行为时数据数量到达中断触发点的瞬间。对于 RXFIFO，仅产生于接收数据时 RXFIFO 中数据数量到达中断触发点的瞬间，读取时到达触发点不会产生中断；对于 TXFIFO，中断仅产生于发送时 TXFIFO 中数据数量到达中断触发点的瞬间，往 RXFIFO 中写入数据达到触发点时也不会产生中断。

31.3.5. UART 波特率

串口设置中，首先配置波特率，通过设置分频因子寄存器来完成。波特率寄存器 UART_BRR，其中 UART_IBRD 位域设置波特率计算值的整数部分，UART_IBAUD = (integer(FPCLK/(16*BAUD))), integer 为取整操作；UART_FBRD 位域设置波特率计算值的小数部分，UART_FBAUD = (integer(badf*64 + 0.5))，integer 为取整操作，badf 为 FPCLK/(16*BAUD)小数部分。

31.3.6. 串口设置

在寄存器 UART_CR3 中：

1）SPS 位选择校验模式（选择奇偶校验还是 0/1 校验）

2）WLEN 位域选择字宽（支持 5~9bit）

3）FEN 位配置 FIFO(是否使用 FIFO ）

4）STP2 位配置停止位的个数

5）EPS 位配置具体的校验方式（选择奇检验还是偶校验、选择强制 0 校验还是强制 1 校验，和 SPS 位有关）

6）PEN 位配置校验使能

31.3.7. 总线空闲（IDLEI）检测

● 总线空闲检测有两种方式实现，一种是通过空闲帧 IDLEI 标志实现，一种是通过比特计时功能实现。

● 空闲帧 IDLE

接收数据时，检测到上一个字符的停止位时，内部定时器开始工作，如果定时器的时长大于一个字符的时长，就会置位 IDLEI，并可以进入中断。

字符长度包括起始位、数据位、校验位和停止位长度。

这种方式时长固定，且检测时长较短。

● 比特计时功能

使能 UART_BCNT.AUTO_START_EN 后，接收数据时，检测到上一个字符的停止位时，内部定时器开始工作，如果定时器的时长大于 UART_BCNT.BCNT_VALUE，就会置位 BCNTI，并可以进入中断。

这种方式时长可配置。

31.3.8. UART 数据发送

在配置好 UART 后，往 UART_DR 中写数据，该数据将立即通过 TX 管脚以配置的波特率发送出去。

31.3.9. UART 数据接收

当 UART 收到一个字符后，数据将被保存在 UART_DR 寄存器中，UART_FR 寄存器中的 RXFE 将被清零。如果未启 FIFO 功能，则一次只能接收一个数据，当 UART 再次接收到一个数据后，将产生 Overrun 错误标识。当开启了 FIFO 功能后，将可以接收 16 个数据，直到 FIFO 填满。当 RXFIFO 满了之后，还有新数据进来，也会产生 Overrun 错误。

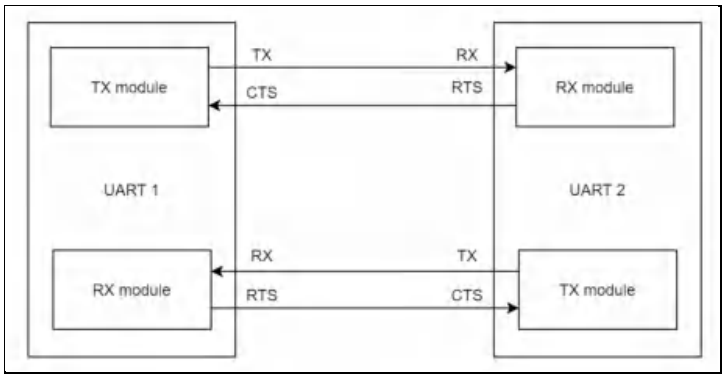
开启 FIFO 并配合 IDLEI 标识完成数据包的接收

当开启了 FIFO 功能后，那么 RXI 中断的产生将根据 RXFIFO 的触发阈值而产生。当接收到一包数据的最后几个字节时，若数据量小于 RXFIFO 触发阈值，那么将不会产生 RXI 中断，此时软件通过中断方式接收时将收不到最后几个储存在 RXFIFO 中的数据。此时可以配合 UART 的 IDELI 中断，在 IDELI 中断处理过程中将 RXFIFO 中的数据全部读取完，完成整包数据的接收。

31.3.10. CTS 和 RTS 流控功能

硬件流控功能的通过 CTS 和 RTS 引脚来实现。通过配置控制寄存器 UART_CR1 的 CTSEn 和 RTSEn 位来使能硬件流控功能。

图 31-2 两个 UART 之间的硬件流控连接示意图



控制流功能不影响 UART 的正常使用，没用使用需求时请禁止使用该功能。

CTS 为 UART 输入端口，低电平有效，表示 UART 可以发送数据。如果 CTS 输入状态为 1，用在当前传输结束时阻止数据发送。写 UART_DR 寄存器时，数据只会保存在发送 FIFO 中不会被发出，为 0 时开始发送。

RTS 为 UART 输出端口，低电平有效，表示 UART 已经准备好可以接收数据，当接收 FIFO 中数据个数大于 RXIFLSEL 寄存器所设的中断触发点个数时，RTS 输出状态会被置位，表示不能再接收更多数据。

31.3.11. 通过 DMA 进行 UART 数据收发

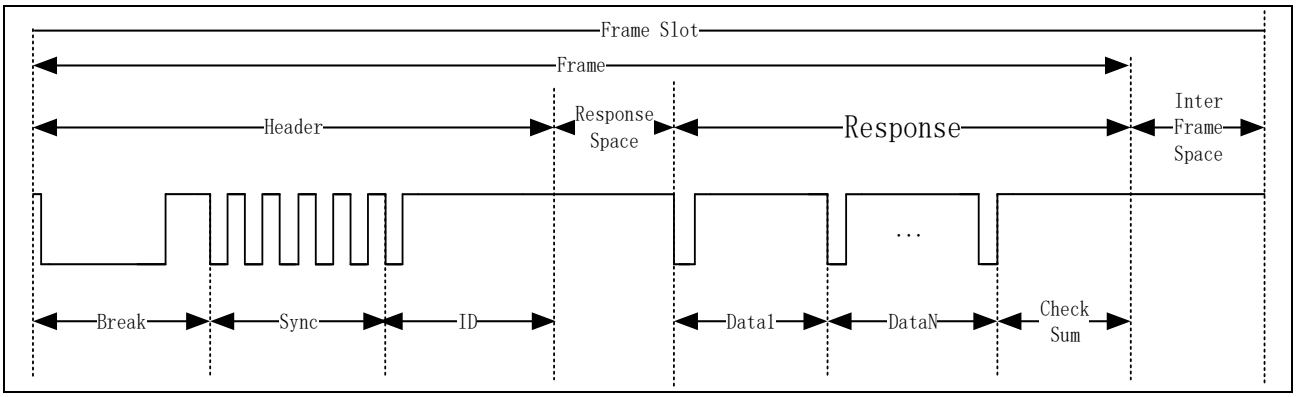
每个 UART 接口的 TX 和 RX 都支持 DMA 功能，都有其对应的 DMA 请求号。

设置 UART_CR1.TXDMAE 位使能 DMA 发送，设置 RXDMAE 位使能 DMA 接收。

另外可以通过设置 UART_CR1.DMAONERR 来选择在 UART 发生接收错误（PE、FE、BE、OE）时是否产生 DMA 请求。

31.3.12. LIN 总线功能

图 31-3 LIN 帧格式



上图是 LIN 总线的基本数据帧格式图，其中包含间隔场、同步场、标识符场、数据场和校验场。其中除了间隔场外，其他场的数据格式都和普通带 1 个起始位和 1 个停止位的 UART 数据格式一样。间隔场包含一个连续不少于 13 个 bit 的低电平信号。见下图所示。

图 31-4 LIN 间隔场格式

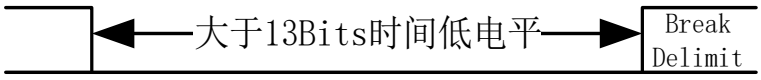
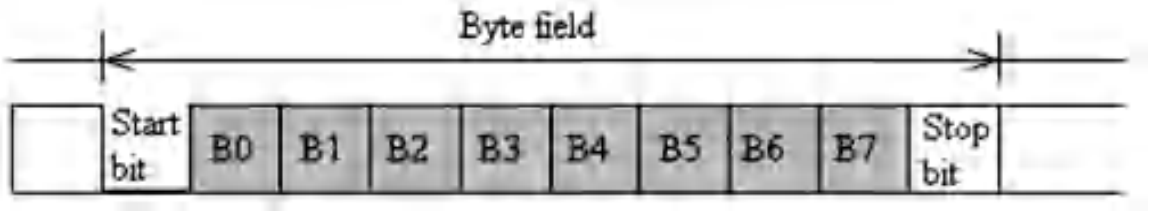


图 31-5 LIN 字符格式



设置 UART_BCNT.BCNT_VALUE 位域可以改变 LIN 总线模式间隔场的 Break 信号长度，使能其中的 BCNT_START 位开始计时，然后使能 UART_CR3.BRK 位发送间隔场。

在发送同步场时，可以发送一个 0x55，发送过程与普通 UART 发送过程相同。

在发送标识符场、数据场和校验场时，LIN 发送过程与普通 UART 发送过程相同。

从模式接收时，需使用间隔场 Break 信号的检测功能，可以通过直接轮询访问 UART_ISR.LBDI 位,或通过使能 UART_IE.LBDI 中断，在中断服务函数中检测 Break 信号来实现。

同时 LIN 总线还支持在从机进入 STOP 模式时，通过 EXTI 模块唤醒，实现低功耗应用。

31.3.13. IrDA SIR 功能

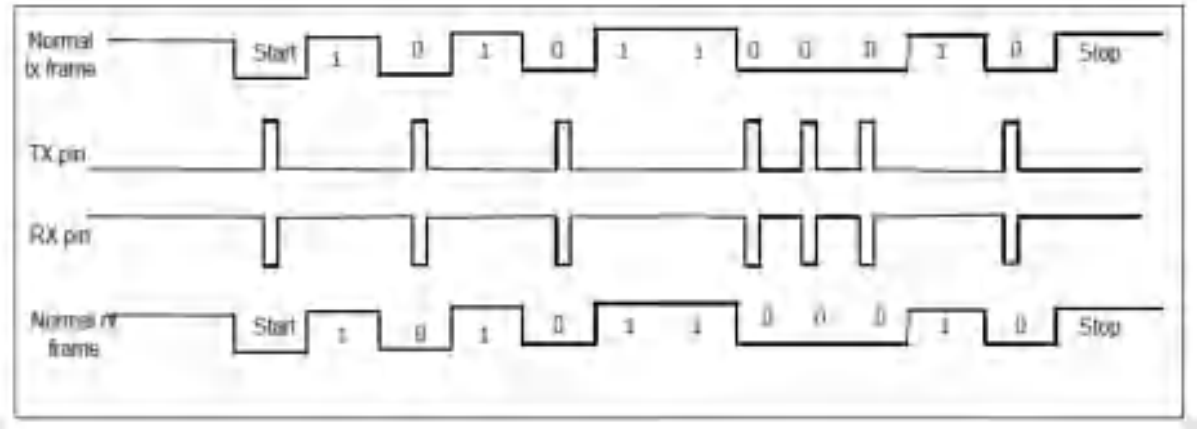
使能 UART_CR1.SIREN 位，即可打开 IrDA SIR 红外功能。

如若使用 IrDA SIR Low Power 模式，需要使能 UART_CR1.SIRLP 位。

在 IrDA 模式下，UART 数据帧由 SIR 发送编码器进行调制，调制后的信号经由红外 LED 进行发送，经解调后将数据发送至 UART 接收器。对于编码器而言，波特率应小于 115200。在 IrDA 模式下，TX 引脚电平与 RX 引脚不同。TX 引脚通常为低电平，RX 引脚通常为高电平。IrDA 引脚电平保持稳定代表逻辑 ‘1’ ，红外光源脉冲(RTZ 信号)代表逻辑 ‘0’ 。其脉冲宽度通常占一个位时间的 3/16。

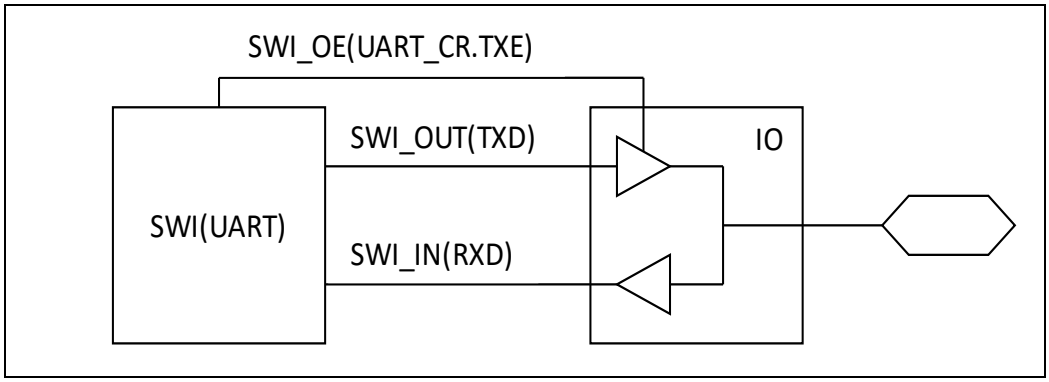
启用 Low Power 模式时，高电平脉冲宽度为为最高波特率 115200 时的一个位宽的 3/16。

图 31-6 IrDA SIR 数据调制解调



31.3.14. 单线半双工通信

图 31-7 UART 单线半双工模式框图



UART 可以工作在单线半双工模式，通过设置使能 UART_CR2.HDSEL 位开启。开启单线半双工模式时，发送和接收都是通过 TX 引脚完成。

发送时，设置 UART_CR1.TXE 为 1，开启 TX 引脚发送，此时需禁用 RXE。

接收时，设置 UART_CR1.TXE 为 0，开启 TX 引脚接收，此时需启用 RXE。

31.3.15. 智能卡主模式

UART 支持智能卡模式，置位 UART_CR2.SCEN，打开 UART 的智能卡模式。在智能卡模式中，UART 固定采用 1.5 位停止位，此时 UART_CR3.STP2 配置无效。

智能卡模式中，需要如下配置：

- 需要置位 UART_CR2.SCEN
- 需要置位 UART_CR2.HDSEL，因为接收发送都是通过 TX 管脚完成。
- 需要置位 UART_CR3.PEN，使能校验位。
- 需要置位 UART_CR2.CLKEN。可以从 CK 管脚上输出时钟，由 UART 时钟根据 UART_GTPR.PSC 分频得到，这个功能比较独立，在非智能卡模式也可以使用。
- 如果需要支持重发机制，则最好关闭发送 FIFO，在发送时检测到 FE 错误时，需要软件配合重新发送上一个字节。
- 如果发送时需要额外保护时间，需要配置 UART_GTPR.GT。
- 如果需要实现接收超时功能（BWT 和 CWT），可以配置 UART_BCNT.AUTO_START_EN 位，并配置合适大

小的 UART_BCNT.BCNT_VALUE。

发送模式

- 配置 UART_GTPR.GT 以增加额外保护时间。发送一个字符需要 12+UART_GTPR.GT 个 ETU 时间，然后才会将发送完成标志 TXI 置位。
- 在开始位、数据位和校验位期间，发送的输出使能被置位，可以驱动总线。在停止位期间，UART 释放总线，并在校验位结束后延迟一个 ETU 时间，采样总线状态，如果检测到低电平，将帧错误标志 FE 在 11 个 ETU 处置位。注意 FE 标志无需等待额外保护时间，先于发送完成标志 TXI 至少一个 ETU 时间。

接收模式

- 接收时，如果检测到校验错误，会根据 UART_CR2.NACK 位来决定是否在停止位期间拉低总线，以告诉发送器发生了校验错误。
- 如果检测到校验错误，会在 9.5 个 ETU 处置位 PE。如果置位了 UART_CR2.NACK 位，则会在从第 10.5 个 ETU 处拉低总线，并维持 1 个 ETU 时间。

31.3.16. 多机通信

多机通信网络支持一主多从模式，某个 UART 设备可以为主，它的 TX 输出和其他 UART 从设备的 RX 输入相连接；UART 从设备各自的 TX 输出逻辑与在一起，并且和主设备的 RX 输入相连接。

■ 未被寻址的设备可启动静默模式，在静默模式中：

- 任何接收状态位都不会被置位，包括 PE、FE、BE、LBDI。
- 所有接收中断被禁止。
- UART_CR2.RWU 位被置 1。

■ 根据 UART_CR2.WAKE 位状态，UART 可以用两种方法进入或退出静默模式。

- 如果 WAKE = 0：进行空闲总线检测。
  - 当 UART_CR2.RWU 被软件写 1 时，UART 进入静默模式。当检测到空闲帧时，它被唤醒。然后 UART_CR2.RWU 被硬件清零，但是 UART_ISR.IDLEI 不置位。
  - RWU 可以在任何时候被软件写 0，并退出静默模式。
  - 空闲帧（IDLEI）是从接收字符的停止位开始计数，等待一个完整字符时间，如果没有新的数据在总线上传输，则置位。
- 如果 WAKE = 1：进行地址标记检测。
  - 如果接收到的字节与它的编程地址不匹配时，硬件置位 UART_CR2.RWU，UART 进入静默模式。接收的地址字节不会写入到接收 FIFO 中，也不会产生中断，因为此时已经进入了静默模式。
  - 当接收到的字节与接收器内的编程地址匹配时，硬件清零 UART_CR2.RWU，UART 退出静默模式。接收的地址字节被正常接收，因为 RWU 已被清零。
  - 在静默模式时，当接收缓冲器不包含数据时，UART_CR2.RWU 位可以被软件写 0 或 1。否则该次软件写操作被忽略。

■ 地址标记检测

- 在这个模式下，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。在一个地址字节中，目标地址被放在低位。
- 如果字宽 WLEN 等于 8 比特或者 9 比特，则支持 7 位地址模式。其他字宽（5、6、7 比特）不支持 7 位地址模式。所有字宽都支持 4 位地址模式。UART_CR2.ADDM7 决定 7 位地址模式还是 4 位地址模式。
- 接收到的地址同 UART_CR2.ADDR 做比较。

31.3.17. 波特率自适应

- UART 可根据接收一个字符检测并自动设置波特率寄存器。自动波特率检测在以下两种情况下非常有用：
  - 事先不知道系统的通信速度。
  - 系统正在使用精度相对较低的时钟源且该机制允许在不测量时钟偏差的情况下获得正确的波特率。
- 波特率自适应功能需发送 0x7F 字符, UART 会测量下降沿到下降沿的时间（8 个比特位时间）。
- 可以被检测的波特率范围为 UART 模块工作时钟频率的 16~65536 分频。
- 置位 UART_CR2.ABREN 位使能波特率自适应功能。之后 UART 将等待 RX 线路上的传递的字符。通过 UART_ISR.ABRI 位来指示自动波特率操作完成。
- 在波特率自适应完成后，UART_BRR 波特率寄存器值将被自动更新，
- 需要软件将 UART_CR2.ABREN 位清零，否则会一直处于波特率自适应模式。
- 在波特率自适应模式中，UART 接收到的数据将会被丢弃。

31.3.18. RS485 的控制器支持

RS485 属于半双工总线，所以 RS485 收发器需要进行发送和接收的方向转换。一种方式是可以使用 MCU 的某个通用 GPIO 口连接到 RS485 收发器的方向控制脚（以下简称 DE Pin），由软件来控制 RS485 收发器的收发方向。如果 MCU 自带 RS485 功能，则可以使用 MCU 的 RS485 专用引脚（DE 引脚）来自动控制收发器的收发方向。

UART RTS 和 DE 共用一个管脚，DE 功能优先。所有 RTS 引脚都可以作为 DE 功能。

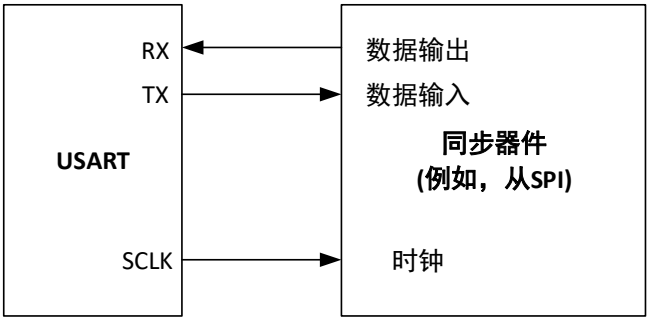
根据用户的硬件电路，DE 信号可能需要高有效，或者低有效，这可以通过 UART_BCNT.DEP 来选择 DE 信号的极性。

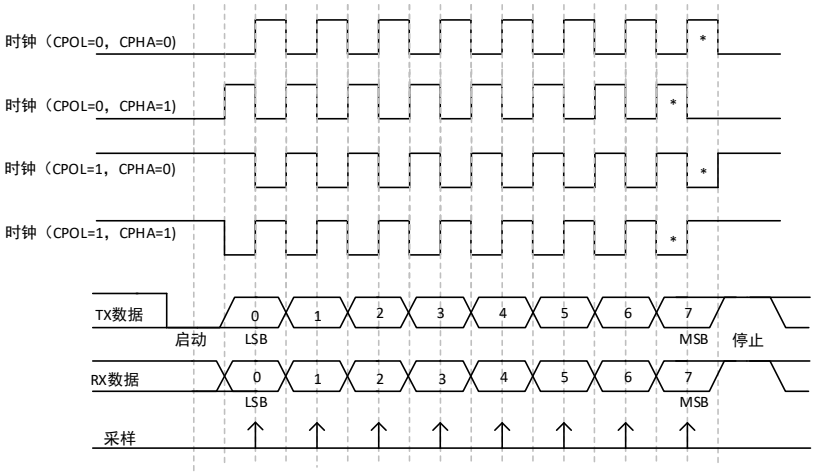
例如，RS485 收发器的 DE 脚高电平代表发送方向，低电平代表接收方向。当选择 DE Pin 高有效时，在 UART 发送第一个字符的起始位之前的一段时间（通过 UART_BCNT.BCNT_VALUE.DEAT 配置），会先将 DE Pin 拉高，在发送最后一个字符的停止位之后的一段时间（通过 UART_BCNT.BCNT_VALUE.DEDT 配置），再将 DE Pin 拉低，让 RS485 收发器处于接收状态。

31.3.19. 同步模式

- 通过将 CR2.CLKEN 位写入 1 来选择同步模式。在同步模式下，必须将以下位清零：
  - CR1.SIREN、CR2. HDSEL 和 CR2.SCEN。
- 通过 UART，用于可以在主模式下控制双向同步串行通信。SCLK 引脚是 UART 发送器时钟的输出。在起始位或停止位期间，不会向 SCLK 引脚发送时钟脉冲。在最后一个有效数据位（地址标记）期间，将会（也可能不会）生成时钟脉冲，这取决于 CR2.LBCL 位的状态。通过 CR2.CPOL 用户可以选择时钟极性；通过 CR2.CPHA 用户可以选择外部时钟相位。
- 在空闲状态、报头模式和发送断路期间，外部 SCLK 时钟处于未激活状态。
- UART 发送器在同步模式下的工作方式与异步模式下完全相同。但是由于 SCLK 与 TX 同步（根据 CPOL 和 CPHA），因此 TX 上的数据是同步的。
- 在此模式下，UART 接受器的工作方式与异步模式下的不同。如果 RE=1，则数据在 SCLK 上采样（上升或下降沿，取决于 CPOL 和 CPHA），而不会进行任何过采样。此时必须保证建立时间和保持时间（取决于波特率：1/16 位时间）符合要求。
- SCLK 引脚可与 TX 引脚结合使用。因此，仅当使能发送器（TXE=1）且正在发送数据时，才会提供时钟。这意味着，没有发送数据的情况下无法接受同步数据。
- 只有在 TXE=0 且 RXE=0 时，才能修改 LBCL、CPOL 和 CPHA 位。

- 建议按照相同指令将 TXE 和 RXE 位置 1，以尽量缩短接受器的建立时间和保持时间。
- UART 只支持主模式：它不能接受或发送与输入时钟相关的数据（SCLK 始终为输出）
- 同步模式的 SCLK 和智能卡的 CK 共用一个管脚。





本时序图以8位数据为例

*LBCL位控制最后一个数据时钟脉冲

31.4. UART 中断

中断名	描述	标识位清除方式
ABRI	波特率自适应完成中断	写 1 清 0
IDLEI	IDLE 中断	写 1 清 0
BCNTI	Bit Count Timeout 中断	写 1 清 0
LBDI	LIN Break Detection 中断	写 1 清 0
OEI	overrun 中断	写 1 清 0
BEI	break error 中断	写 1 清 0
PEI	奇偶校验错误中断	写 1 清 0
FEI	帧格式错误中断	写 1 清 0
TXI	发送中断	写 1 清 0 或写 DR
RXI	接收中断	写 1 清 0 或读 DR

31.5. 配置流程

31.5.1. 串口设置

串口设置中，首先配置波特率，通过设置分频因子寄存器来完成。波特率寄存器 UART_BRR，其中 UART_IBRD 位域设置波特率计算值的整数部分，UART_IBAUD = (integer(FPCLK/(16*BAUD))), integer 为取整操作；UART_FBRD 位域设置波特率计算值的小数部分，UART_FBAUD = (integer(badf*64 + 0.5))，integer 为取整操作，badf 为 FPCLK/(16*BAUD)小数部分。

在寄存器 UART_CR3 中：

1）SPS 位选择校验模式（选择奇偶校验还是 0/1 校验）

2）WLEN 位域选择字宽（支持 5~9bit）

3）FEN 位配置 FIFO(是否使用 FIFO ）

4）STP2 位配置停止位的个数

5）EPS 位配置具体的校验方式（选择奇检验还是偶校验、选择强制 0 校验还是强制 1 校验，和 SPS 位有关）

6）PEN 位配置校验使能

31.5.2. 串口的发送和接收

1）配置波特率（UART_BRR）

2）配置 FIFO(是否使用 FIFO ）（UART_CR3.FEN）

3）配置控制寄存器(奇偶校验位等)（UART_CR3.PEN/EPS/SPS）

4）配置中断使能寄存器(是否使用中断)（UART_IE）

5）使能 UART（UART_CR1.TXE/RXE/UARTEN）

31.5.3. LIN 硬件功能支持

- LIN 作为从机，检测 Break 同步间隔帧有两种方式：
  - 直接轮询访问 UART_ISR.LBDI 位
  - 使能 UART_IE.LBDI 位，然后触发中断，在中断查看 UART_ISR.LBDI 位
- LIN 检测 Break 同步间隔帧的最小长度由一个当前设置的标准数据帧决定，LBDI 信号会在 RXD 数据变为高时被置为有效
  - 1 位起始位
  - N 位数据位（由 UART_CR3.WLEN 决定，建议 8 位）
  - 校验位（0 位或 1 位校验位，由 UART_CR3.PEN 决定）
  - 停止位（1 位或 2 位停止位，由 UART_CR3.STP2 决定）
- LIN 作为主机发送 Break 同步间隔帧流程：
  - 配置 UART_BCNT 寄存器的 BCNT_VALUE 为 13
  - 同步使能 UART_CR3.BRK 位和 UART_BCNT. BCNT_START 位
  - 轮询访问 UARTISR.BCNTI 位，等待置 1；或者使能中断，等待 BCNTI 中断
  - 清零 UART_CR3.BRK 位

31.5.4. IrDA SIR 功能使用流程

1）配置 UART_GTPR.PSC，设置 IrDA 低功耗波特率为 115200

2）使能 UART_CR1.SIREN 位，即可打开 IrDA SIR 红外功能

3）如若使用 IrDA SIR Low Power 模式发送数据，需要使能 UART_CR1.SIRLP 位

4）UART 模块在 IrDA SIR 模式时，总是采用低功耗波特率来采样接收的数据。

31.5.5. 单线模式功能使用流程

1）使能 UART_CR2.HDSEL 位

2）单线模式中发送数据和接收数据都使用 TX 管脚。

3）单线模式在发送时需要开启 TXE 同时禁止 RXE；接收时开启 RXE

31.5.6. 智能卡功能使用流程

1）置位 UART_CR2.SCEN

2）置位 UART_CR2.HDSEL，设置 TX 引脚单线半双工

3）置位 UART_CR3.PEN，使能校验位

4）通过 UART_GTPR.PSC 设置 CK 分频

5）置位 UART_CR2.CLKEN，从 CK 管脚上输出时钟

6）如果需要支持重发机制，则关闭发送 FIFO，在发送时检测到 FE 错误时，需要软件配合重新发送上一个字节

7）如果发送时需要额外保护时间，需要配置 UART_GTPR.GT

8）如果需要实现接收超时功能（BWT 和 CWT），可以配置 UART_BCNT.AUTO_START_EN 位，并配置合适大小的 UART_BCNT.BCNT_VALUE

9）发送时，配置 UART_GTPR.GT 以增加额外保护时间

10）接收时，使能 UART_CR2.NACK 位以检测校验错误

31.5.7. 多机通信功能使用流程

1）多机通信网络主设备的 TX 输出和其他 UART 从设备的 RX 输入相连接；从设备各自的 TX 输出逻辑与在一起后，和主设备的 RX 输入相连接

2）设备可通过设置 UART_CR2.RWU 进入静默模式

3）根据需求设置 UART_CR2.WAKE 位，UART 可以用两种方法进入或退出静默模式

- 如果 WAKE = 0：进行空闲总线检测
- 如果 WAKE = 1：进行地址标记检测

4）如果使用地址标记检测

5）设置 UART_CR2.ADDM7 决定 7 位地址模式还是 4 位地址模式

6）设置 UART_CR2.ADDR 节点地址

31.5.8. 波特率自适应功能使用流程

1）置位 UART_CR2.ABREN 位使能波特率自适应功能

2）UART 等待 RX 线路上的主机传递的 0x7F 字符。通过 UART_ISR.ABRI 位来指示自动波特率操作完成。也可 UART_IE. ABRI 产生相应的中断

3）在波特率自适应完成后，UART_BRR 波特率寄存器值将被自动更新，需要软件将 UART_CR2.ABREN 位清零，否则会一直处于波特率自适应模式

4）在波特率自适应模式中，UART 接收到的数据将会被丢弃

31.5.9. RS485 的 DE 控制功能使用流程

1）将 UART_BCNT.DEM 位置 1，使能 DE 信号控制功能

2）设置 UART_BCNT.DEAT 位域值，配置 DE 信号与 START 位之间的时间

3）设置 UART_BCNT. DEDT 位域值，配置发送的消息中最后一个字符的停止位与 DE 信号之间的时间

4）通过 UART_BCNT. DEP 位配置 DE 的极性

31.5.10. 同步模式

- 通过将 CR2.CLKEN 位写入 1 来选择同步模式。在同步模式下，必须将以下位清零：
  - CR1.SIREN、CR2. HDSEL 和 CR2.SCEN。
- 通过 UART，用于可以在主模式下控制双向同步串行通信。SCLK 引脚是 UART 发送器时钟的输出。在起始位或停止位期间，不会向 SCLK 引脚发送时钟脉冲。在最后一个有效数据位（地址标记）期间，将会（也可能不会）生成时钟脉冲，这取决于 CR2.LBCL 位的状态。通过 CR2.CPOL 用户可以选择时钟极性；通过 CR2.CPHA 用户可以选择外部时钟相位。
- 在空闲状态、报头模式和发送断路期间，外部 SCLK 时钟处于未激活状态。
- UART 发送器在同步模式下的工作方式与异步模式下完全相同。但是由于 SCLK 与 TX 同步（根据 CPOL 和 CPHA），因此 TX 上的数据是同步的。
- 在此模式下，UART 接受器的工作方式与异步模式下的不同。如果 RE=1，则数据在 SCLK 上采样（上升或下降沿，取决于 CPOL 和 CPHA），而不会进行任何过采样。此时必须保证建立时间和保持时间（取决于波特率：1/16 位时间）符合要求。
- SCLK 引脚可与 TX 引脚结合使用。因此，仅当使能发送器（TXE=1）且正在发送数据时，才会提供时钟。这意味着，没有发送数据的情况下无法接受同步数据。
- 只有在 TXE=0 且 RXE=0 时，才能修改 LBCL、CPOL 和 CPHA 位。
- 建议按照相同指令将 TXE 和 RXE 位置 1，以尽量缩短接受器的建立时间和保持时间。
- UART 只支持主模式：它不能接受或发送与输入时钟相关的数据（SCLK 始终为输出）

同步模式的 SCLK 和智能卡的 CK 共用一个管脚。

31.6. UART 寄存器描述

31.6.1. 寄存器列表

UART1 寄存器基地址：0x40013800

UART2 寄存器基地址：0x40004400

UART3 寄存器基地址：0x40004800

UART4 寄存器基地址：0x40004C00

UART5 寄存器基地址：0x40005000

UART6 寄存器基地址：0x40013C00

UART7 寄存器基地址：0x40009800

UART8 寄存器基地址：0x40009C00

UART9 寄存器基地址：0x4001B000

UART10 寄存器基地址：0x4001B400

偏移	名称	复位值	描述
0x00	UART_DR	0x00000000	数据寄存器
0x04	UART_FR	0x00000090	标志寄存器
0x08	UART_BRR	0x00000000	波特率寄存器
0x0C	UART_IE	0x00000000	中断使能寄存器
0x10	UART_ISR	0x00000000	中断和状态寄存器
0x14	UART_CR1	0x00000300	控制寄存器 1
0x18	UART_CR2	0x00000000	控制寄存器 2
0x1C	UART_CR3	0x00004800	控制寄存器 3
0x20	UART_GTPR	0x00000000	保护时间和预分频寄存器
0x24	UART_BCNT	0x0000000d	比特计时寄存器

31.6.2. 数据寄存器(UART_DR: 00h)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	OE	RO	0	Overrun 错误
11	BE	RO	0	Break 错误
10	PE	RO	0	奇偶校验错误
9	FE	RO	0	帧格式错误
8:0	DATA	RW	0	发送或接收的数据 注： 使能 FIFO 功能时，若 FIFO 已满，有新的数据进入，FIFO 中原有数据不会被覆盖，新的数据会被直接丢失 禁止 FIFO 功能时，若 UART_DR 寄存器中有数据还未发送，写 UART_DR 寄存器，新的发送数据会被直接丢失，不会覆盖 UART_DR 寄存器中原有未发送数据

31.6.3. 标志位寄存器(UART_FR: 04h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留

9	BUSY	RO	0	发送忙标志 FIFO 开启时： 0：发送 FIFO 为空并且所有位都从移位寄存器中移出 1：发送 FIFO 有数据 FIFO 关闭时： 0：移位寄存器空 1：移位寄存器未空
8	CTS	RO	0	CTS 输入管脚状态 0：CTS 输入高电平 1：CTS 输入低电平
7	TXFE	RO	1	发送 FIFO/UART_DR 寄存器空状态位 0：如果使能 FIFO 表示发送 FIFO 非空；如果禁止 FIFO 表示 UART_DR 寄存器有数据 1：如果使能 FIFO 表示发送 FIFO 为空；如果禁止 FIFO 表示 UART_DR 寄存器无数据
6	RXFF	RO	0	接收 FIFO/UART_DR 寄存器满状态位 0：如果使能 FIFO 表示接收 FIFO 非满；如果禁止 FIFO 表示 UART_DR 寄存器非满 1：如果使能 FIFO 表示接收 FIFO 为满；如果禁止 FIFO 表示 UART_DR 寄存器为满
5	TXFF	RO	0	发送 FIFO/UART_DR 寄存器满状态位 0：如果使能 FIFO 表示发送 FIFO 非满；如果禁止 FIFO 表示 UART_DR 寄存器非满 1：如果使能 FIFO 表示发送 FIFO 为满；如果禁止 FIFO 表示 UART_DR 寄存器为满
4	RXFE	RO	1	接收 FIFO/UART_DR 寄存器空状态位 0：如果使能 FIFO 表示接收 FIFO 非空；如果禁止 FIFO 表示 UART_DR 寄存器有数据 1：如果使能 FIFO 表示接收 FIFO 为空；如果禁止 FIFO 表示 UART_DR 寄存器无数据
3	OE	RO	0	Overrun 错误 0: 无错误 1：有错误 注：OE 只在接收时有效 开启 FIFO 功能时，当接收 FIFO 已满后再接收到数据时被置位 关闭 FIFO 功能时，当 UART_DR 寄存器中有数据后再接收到数据时被置位 写 1 清 0
2	BE	RO	0	Break 错误 0: 无错误 1：有错误 当接收数据持续为低超过传输 1 个 word 的时间时被置位
1	PE	RO	0	奇偶或 0/1 校验位错误 0: 无错误 1：有错误

0	FE	RO	0	帧格式错误 0：无错误 1：有错误 当停止位错误时被置位

31.6.4. 波特率寄存器(UART_BRR: 08h)

位域	名称	属性	复位值	描述
31:22	RSV	RO	0x00	保留
21:6	UART_IBAUD	RW	0x0000	波特率分频整数因子 UART_IBAUD = (integer(Fsys/(16*BAUD))), integer 为取整操作
5:0	UART_FBAUD	RW	0x00	波特率分频分数因子 UART_FBAUD =(integer(badf*64 + 0.5))，integer 为取整操作，badf 为 Fsys/(16*BAUD)小数部分 注：若 FBAUD 计算结果大于等于 64，将本寄存器写 0，将原本写入 UART_IBRD 的值加 1

31.6.5. 中断使能寄存器(UART_IE: 0Ch)

位域	名称	属性	复位值	描述
31:16	RSV	RO	0	保留
15	TCI	RW	0	发送完成中断使能位 0：禁止 1：使能
14	ABRI	RW	0	波特率自适应完成中断使能位 0：禁止 1：使能
13	IDLEI	RW	0	IDLE 中断使能位 0：禁止 1：使能
12	BCNTI	RW	0	Bit Count Timeout 中断使能位 0：禁止 1：使能
11	LBDI	RW	0	LIN Break Detection 中断使能位 0：禁止 1：使能
10	OEI	RW	0	overrun 中断使能位 0：禁止 1：使能
9	BEI	RW	0	break error 中断使能位 0：禁止 1：使能

8	PEI	RW	0	奇偶校验错误中断使能位 0：禁止 1：使能
7	FEI	RW	0	帧格式错误中断使能位 0：禁止 1：使能
6	RSV	RO	0	保留
5	TXI	RW	0	发送中断使能位 0：禁止 1：使能
4	RXI	RW	0	接收中断使能位 0：禁止 1：使能
3:0	RSV	RO	0	保留

31.6.6. 中断和状态寄存器(UART_ISR: 10h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	TCI	RCW1	0	发送完成原始中断 0：未完成 1：完成 写 1 清 0
14	ABRI	RCW1	0	波特率自适应功能完成原始中断 0：未完成 1：完成 写 1 清 0
13	IDLEI	RCW1	0	监测总线空闲原始中断 0：没有检测到空闲总线 1：检测到空闲总线 写 1 清 0
12	BCNTI	RCW1	0	Bit Count Timeout 原始中断 0：无中断 1：有中断 写 1 清 0
11	LBDI	RCW1	0	LIN Break Detection 原始中断 0：无中断 1：有中断 写 1 清 0

10	OEI	RCW1	0	overrun 原始中断 0：无中断 1：有中断 当 OE 标志产生时此位被置位 写 1 清 0
9	BEI	RCW1	0	break error 原始中断 0：无中断 1：有中断 当 BE 标志产生时此位被置位 写 1 清 0
8	PEI	RCW1	0	奇偶校验错误原始中断 0：无中断 1：有中断 当 PE 标志产生时此位被置位 写 1 清 0
7	FEI	RCW1	0	帧格式错误原始中断 0：无中断 1：有中断 当 FE 标志产生时此位被置位 写 1 清 0
6	RSV	-	-	保留
5	TXI	RCW1	0	发送原始中断 0：无中断 1：有中断 注： ● 使能 FIFO 功能时，当发送时发送 FIFO 中的数据个数到达 TXIFLSEL 寄存器所设的中断触发点时此位被置位。可以通过以下 3 种方式清除该中断： ➢ 向本寄存器位写 1 清 0 ➢ 向 FIFO 中填入数据使 FIFO 中的数据数量大于 TXIFLSEL 寄存器所设的中断触发个数 ➢ 重新设置 TXIFLSEL 寄存器使中断触发个数由大变小（如由 8 个触发变为 2 个触发） ● 禁止 FIFO 功能时，当数据从 UART_DR 寄存器进入发送移位寄存器时此位被置位。可以通过以下 2 种方式清楚该中断： ➢ 向本寄存器位写 1 清 0 ➢ 发送未完成时，写新的数据到 UART_DR 寄存器

4	RXI	RCW1	0	接收原始中断 0：无中断 1：有中断 注： ● 使能 FIFO 功能时，接收时当接收 FIFO 中的数据个数到达 RXIFLSEL 寄存器所设的中断触发点时此位被置位。可以通过以下 3 种方式清除该中断： ➢ 向本寄存器位写 1 清 0 ➢ 将接收 FIFO 中的数据读走使 FIFO 中的数据数量小于 RXIFLSEL 寄存器所设的中断触发点个数 ➢ 重新设置 RXIFLSEL 寄存器使中断触发个数由小变大（如由 2 个触发变为 8 个触发） ● 禁止 FIFO 功能时，当 UART_DR 寄存器有新的数据进入时此位被置位。可以通过以下 2 种方式清除该中断： ➢ 向本寄存器位写 1 清 0 ➢ 读 UART_DR 寄存器
3:0	RSV	-	-	保留

31.6.7. 控制寄存器 1(UART_CR1: 14h)

位域	名称	属性	复位值	描述
16	ENDIAN_SEL	RW	0	大小端选择位 0：小端传输，即 LSB 先发，MSB 后发 1：大端传输
15	CTSEN	RW	0	CTS 流控制使能位 0：禁止 1：使能
14	RTSEN	RW	0	RTS 流控制使能位 0：禁止 1：使能
13:12	RSV	RO	0	保留
11	RTS	RW	0	RTS 输出管脚状态 0：RTS 输出高电平 1：RTS 输出低电平
10	RSV	RW	0	保留
9	RXE	RW	1	接收使能位 0：禁止 1：使能
8	TXE	RW	1	发送使能位 0：禁止 1：使能
7:6	RSV	RO	0	保留

5	DMAONERR	RW	0	发生接收错误（PE、FE、BE、OE）时，DMA 接收请求不置位 0：不使能 1：使能
4	TXDMAE	RW	0	发送 DMA 使能 0：不使能 1：使能
3	RXDMAE	RW	0	接收 DMA 使能 0：不使能 1：使能
2	SIRLP	RW	0	IrDA SIR 低功耗模式使能位： 0：禁止 1：使能
1	SIREN	RW	0	IrDA SIR ENDEC 模块使能位： 0：禁止 1：使能
0	UARTEN	RW	0	UART 使能位 0：禁止 1：使能

31.6.8. 控制寄存器 2(UART_CR2: 18h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18	ADDM7	RW	0	是否选择 7 位地址模式 0：4 位地址模式 1：7 位地址模式
17:11	ADDR	RW	0	本设备的 UART 节点地址 该位域给出本设备 UART 节点的地址
10	CPOL	RW	0	时钟极性 同步模式下选择 SCLK 引脚上时钟输出的极性。它与 CPHA 位结合使用可获得所需的时钟/数据关系。 0：空闲时 SCLK 引脚为低电平 1：空闲时 SCLK 引脚为高电平
9	CPHA	RW	0	时钟相位 同步模式下选择 SCLK 引脚上时钟输出的相位。它与 CPOL 位结合使用可获得所需的时钟/数据关系。 0：在时钟第一个变化沿捕获数据 1：在时钟第二个变化沿捕获数据

8	LBCL	RW	0	最后一个位时钟脉冲 同步模式下选择与发送的最后一个数据位关联的时钟脉冲是否必须在 SCLK 引脚输出。 0：最后一个数据位的时钟脉冲不在 SCLK 引脚上输出。 1：最后一个数据位的时钟脉冲在 SCLK 引脚上输出。
7	CLKEN	RW	0	SCLK/CK 时钟使能 0：禁止 SCLK/CK 引脚 1：使能 SCLK/CK 引脚
6	NACK	RW	0	智能卡 NACK 使能 0：校验错误出现时，不发送 NACK 1：校验错误出现时，发送 NACK
5	SCEN	RW	0	智能卡模式使能 0：禁止智能卡模式 1：使能智能卡模式
4	ABREN	RW	0	使能波特率自适应功能 0：禁止波特率自适应功能 1：使能波特率自适应功能
3	WAKE	RW	0	静默模式唤醒方法选择 0：被空闲总线唤醒 1：被地址标记唤醒
2	RWU	RW	0	接收唤醒 该位用来决定是否把 UART 进入静默模式，由软件设置或清除。当唤醒序列到来时，硬件也会将其清零 0：接收器处于正常工作模式 1：接收器处于静默模式
1	RSV	RO	0	保留
0	HDSEL	RW	0	单线半双工模式选择 0：全双工模式 1：单线半双工模式 单线半双工模式： RX 不再使用 TX 输出在发送时输出，其他处于接收模式

31.6.9. 控制寄存器 3(UART_CR3: 1Ch)

位域	名称	属性	复位值	描述
15:13	RXIFLSEL	RW	010	接收中断的触发点选择位: 000: 1/8（接收 FIFO 收到 2 个数据） 001: 1/4（接收 FIFO 收到 4 个数据） 010: 1/2（接收 FIFO 收到 8 个数据） 011: 3/4（接收 FIFO 收到 12 个数据） 100: 7/8（接收 FIFO 收到 14 个数据） 101: 接收 FIFO 变非空（接收 FIFO 收到 1 个数据）

12:10	TXIFLSEL	RW	010	发送中断的触发点选择位: 000: 1/8 （发送到 FIFO 中剩余 2 个数据） 001: 1/4（发送到 FIFO 中剩余 4 个数据） 010: 1/2（发送到 FIFO 中剩余 8 个数据） 011: 3/4（发送到 FIFO 中剩余 12 个数据） 100: 7/8（发送到 FIFO 中剩余 14 个数据） 101: 发送 FIFO 变空（发送 FIFO 为空时不会产生中断，从仅剩 1 个数据到完全为空变化时才会产生中断；产生中断不代表最后一个数据发送完成，仅代表最后一个数据从 FIFO 中移除并开始发送） 注：中断产生不是单纯由 FIFO 中的数据数量决定，触发仅产生于特定操作行为时数据数量到达中断触发点的瞬间。对于接收 FIFO，仅产生于接收数据时 FIFO 中数据数量到达中断触发点的瞬间，读取时到达触发点不会产生中断；对于发送 FIFO，中断仅产生于发送时 FIFO 中数据数量到达中断触发点的瞬间，往发送 FIFO 中写入数据达到触发点时也不会产生中断
9	SPS	RW	0	校验模式选择位 0：奇/偶校验 1：0/1 校验
8:6	WLEN	RW	0	字宽选择位 000：5bits 001：6bits 010：7bits 011：8bits 1xx：9bits
5	FEN	RW	0	FIFO 使能位 0：禁止 FIFO 1：使能 FIFO
4	RSV	RO	0	保留
3	STP2	RW	0	停止位数选择位: 0：1 位停止位 1：2 位停止位
2	EPS	RW	0	0/1 校验或者奇/偶校验选择位（取决于 SPS） 0：奇/偶校验选择奇校验，或 0/1 校验选择校验位强制为 1 1：奇/偶校验选择偶校验，或 0/1 校验选择校验位强制为 0
1	PEN	RW	0	校验使能位: 0：禁止奇/偶校验或 0/1 校验 1：使能奇/偶校验或 0/1 校验
0	BRK	RW	0	BREAK 发送使能位 0：禁止 1：使能

31.6.10. 保护时间和预分频寄存器(UART_GTPR: 20h)

位域	名称	属性	复位值	描述

31:12	RSV	RO	0	保留
11:8	GT	RW	0x00	智能卡模式下使用。保护时间，以 etu 为单位
7:0	PSC	RW	0x00	该位段在 IrDA 或智能卡模式下使用 IrDA 模式下： PSC =integer(Fsys/(16*ILPBAUD))，integer 为取整操作，ILPBAUD 是低功耗模式的目标波特率，正常为 115200，范围要求在 88750~132500 之间 智能卡模式下（低 5 位有效）：对源时钟进行分频 0：源时钟进行 2 分频 1：源时钟进行 4 分频 2：源时钟进行 6 分频 31：源时钟进行 64 分频

31.6.11. 比特计时寄存器(UART_BCNT: 24h)

位域	名称	属性	复位值	描述
31:28	RSV	-	0	保留
27	DEM	RW	0	使能 RS485 的 DE 功能 0：禁止 1：使能
26	DEP	RW	0	DE 信号极性选择 0：高电平有效 1：低电平有效
25	AUTO_START_EN	RW	0	硬件自动计时控制 0：禁止 1：使能 在接收数据，如果使能此功能，在接收到 STOP 位时自动启动一次比特计时
24	BCNT_START	WO	0	比特计时开始。
23:0	BCNT_VALUE	RW	0xd	DEM 不使能时： 比特计时初值 DEM 使能时： [7:4]是 DEAT [3:0]是 DEDT

DE 和 RTS 共用一个管脚，DE 优先。

32. 低功耗串口（LPUART）

32.1. 概述

低功耗 UART（LPUART）是一个低功耗的 UART 模块，通讯可以使用独立的时钟源。只需要使用 32.768KHz 的时钟就可以使用 9600 波特率通讯。更高的波特率可以通过外部选择更快的时钟源实现。

当 LPUART 使用 PCLK 作为时钟时，可以达到更高的波特率。即使设备处于低功耗模式，LPUART 也可以在极低能耗的情况下等待传入的 UART 帧。LPUART 包括所有必要的硬件支持，以最小的功耗实现异步串行通信。DMA（直接存储器访问）可用于数据传输/接收。

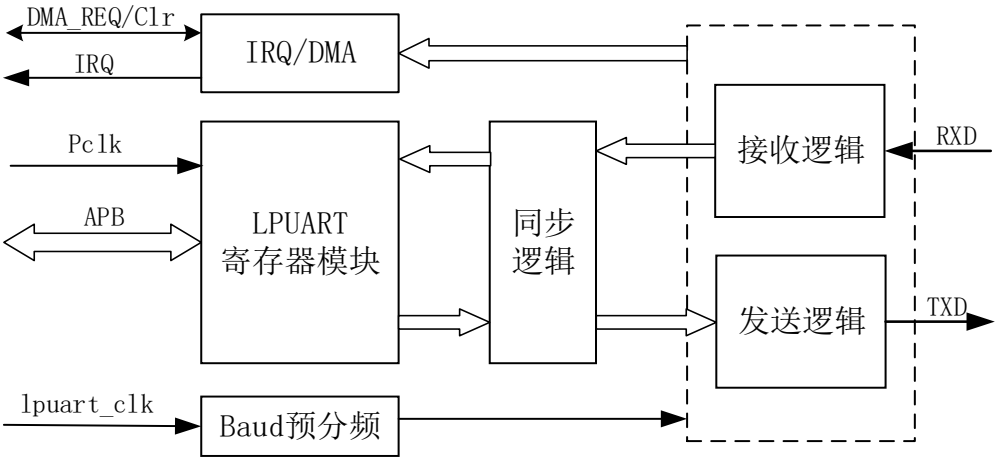
32.2. 主要特性

- 寄存器和通讯时钟独立
- LPUART 一共支持三种时钟，分别为：RCL、XTL、PCLK 分频（分频系数由 LPUARTDIV 确定）
- 32.768KHz 时钟，最大支持 9600 波特率
- 可编程数据字长（7 位或 8 位）
- 可使用 PCLK 的分频时钟作为工作时钟
- 奇/偶校验、0/1 校验或者无校验可配置
- 可配置停止位（1 或 2 个停止位）
- STOP 模式下唤醒系统：起始位、收到 1 字节或者收到字节匹配
- 支持 DMA 工作
- 总线空闲检测
-

32.3. LPUART 功能描述

LPUART 主要分为寄存器接口、接收逻辑、发送逻辑、中断/唤醒/DMA 逻辑等组成。

整体框图如下：



由 LPUART 的整体框图可知，LPUART 模块的接口有如下几种类型：

1）DMA_REQ/Clr 接口：用于利用 DMA 进行接收或发送的通信。

2）IRQ 接口：作为 LPUART 的唤醒源和 LPUART 传输相关的中断；

3）PCLK 接口：APB 总线的工作时钟；

4）APB 总线接口：用于配置串口相关的寄存器；

5）Lpuart_clk 接口：作为总线的工作时钟；

6）RXD/TXD 接口：数据通信接口。

由 LPUART 的整体框图可知，LPUART 模块有如下几个子功能块：

1）IRQ/DMA：具有使用 DMA 方式通信的功能；包含唤醒功能以及中断功能；

2）LPUART 寄存器模块：串口相关参数的寄存器配置；

3）Baud 预分频：波特率分频设置；

4）接收逻辑：用于数据的接收；

5）发送逻辑：用于数据的发送。

LPUART 与 UART 主要区别在于功耗方面，LPUART 是低功耗的通信协议，更适用于低功耗应用场景。

32.3.1. 时钟

LPUART 一共支持三种时钟，分别为：

- RCL
- XTL
- PCLK 分频，分频系数由 LPUARTDIV 确定

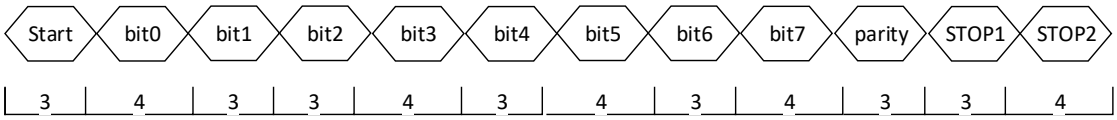
通过 RCC_CCR2 寄存器进行设置，当选择 RCL 或 XTL 为时钟源时，最大波特率支持 9600，LPUART 提供低功耗支持。当选择 PCLK 分频作为时钟源，支持更高波特率，LPUART 提供高性能支持。

32.3.2. 波特率

波特率设置分为整数部分和小数部分，其中整数部分至少为 3 分频（设置值为 2），设置整数分频；小数部分按照比特设置从 Start 到 STOP 分析每个比特是否增加 1。由于 LPUART 工作在 32.768K 频率下时，通讯时钟不是 9600 的整数倍，故必须采用这种小数分频的方式，如采用 3-4 分频循环的方式实现。

下图为 32.768K 频率下，使用 9600 波特率，8 位数据，校验位，和 2 个停止位的示意图。波特率整数部分（IBAUD）配置为 3 分频（每个比特 3-4 个时钟，设置值为 2），接收采样（RXSAM）设为第 2 个时钟（设置值为 1）；小数寄存器配置位 12’ b1001_0101_0010。其中接收采样在每个 bit 的中间时钟。

图 32-1 LPUART BIT 分布图



波特率小数部分的加权平均加上整数部分，应该和工作时钟除以波特率相等；并且从一个字节看小数的 0-1 分布应该均匀，比如要设置为 4’ b0101,而不是 4’ b0011。上面的例子中，工作时钟除以波特率为 32768/9600=3.413；波特率小数部分为 5/12=0.417,加上整数部分 3 则波特率加权平均值为 3.417。

如果外部选择更高的时钟，则 LPUART 可以实现更快的波特率。

- 波特率计算公式为
  - 波特率整数部分=（工作频率/需要波特率）的整数部分
  - 小数部分相求加权平均=工作频率/需要波特率-波特率整数部分
- 如要设为 115200，系统时钟为 64M，经过 32 分频产生 LPUART 工作时钟， 则：
  - 波特率应为 2000000/115200=17.361

➢ 整数部分可以设置为 17 分频（寄存器值 16）

小数加权平均为 0.361,如采用 Start+8 位数据+校验+1 位 STOP 共 11 位，则小数部分在 11 比特中需要 11*0.361≈4 个 1，可以设置为 12’ b100100100100。

32.3.3. 数据位

LPUART 收发控制器可以通过配置线控寄存器 LPUART_LCR 位域 WLEN 选择数据位宽：当 LPUART_LCR.WLEN 为 0 时，数据位宽为 8bits；当 LPUART_LCR.WLEN 为 1 时，数据位宽为 7bits。

32.3.4. 校验位

LPUART 收发控制器可以通过配置线控寄存器 LPUART_LCR 位域 SPS 选择校验模式：当 LPUART_LCR.SPS 为 0 时，校验模式为奇/偶校验；当 LPUART_LCR.SPS 为 1 时，校验模式为 0/1 校验。

通过配置线控寄存器 LPUART_LCR 位域 EPS 选择奇/偶校验中的校验模式，或在 0/1 校验中选择校验位值：当 LPUART_LCR.EPS 为 0 时，校验模式为奇校验或将 0/1 校验选择校验位强制为 1；当 LPUART_LCR.EPS 为 1 时，校验模式为偶校验，或 0/1 校验选择校验位强制为 0。

通过配置 LPUART_LCR 位域 PEN 使能或禁止校验功能：当 LPUART_LCR.PEN 为 0 时，禁止校验功能；当 LPUART_LCR.PEN 为 1 时，使能校验功能。

LPUART_LCR.PEN=1	LPUART_LCR.SPS=0	LPUART_LCR.SPS=1
LPUART_LCR.EPS=0	奇校验	校验位强制为 1
LPUART_LCR.EPS=1	偶校验	校验位强制为 0

32.3.5. 停止位

LPUART 收发控制器可以通过配置线控寄存器 LPUART_LCR 位域 STP2 选择停止位数：当 LPUART_LCR.STP2 为 0 时，具有 1 位停止位；当 LPUART_LCR.STP2 为 1 时，具有 2 位停止位。

32.3.6. 数据极性

发送数据的极性由线控寄存器 LPUART_LCR 位域 TXPOL 控制。当 TXPOL 为 0 时，发送数据不取反；当 TXPOL 为 1 时：发送数据取反。

接收数据的极性由线控寄存器 LPUART_LCR 位域 RXPOL 控制。当 RXPOL 为 0 时，接收数据不取反；当 RXPOL 为 1 时：接收数据取反。

32.3.7. 地址

设置地址寄存器 LPUART_ADDR，写入匹配地址，接到到 1byte 字节数据必须等于 LPUART_ADDR 中的匹配地址时才能有效唤醒。

32.3.8. STOP 唤醒

在 STOP 模式系统时钟（PCLK）停止情况下，LPUART 收发逻辑可以独立工作在 32.768K 时钟下，此时可以选择接收唤醒方式，通过设置 RXWKS[6:5]选择三种唤醒中的一种：

- START 位检测唤醒：当接收到 START 位时唤醒；
- 1byte 数据接收完成：当接收到 1byte 的字节时唤醒；

- 接收数据匹配成功：若选择此唤醒模式，则还需要设置地址寄存器 LPUART_ADDR，写入匹配地址，当接到到 1byte 字节数据且数据等于 LPUART_ADDR 中的匹配地址时唤醒，即使采用地址匹配方式，收到的数据也会放入数据寄存器。
- 通过设置 WKCK 位，可以选择接收完 1 字节，是否检查校验位和 STOP 位，再触发唤醒/中断。

当 WKCK 位为 0 时，接收完 1 字节，不检查校验位和 STOP 位，直接触发唤醒/中断。

当 WKCK 位为 1 时，接收完 1 字节，检查校验位和 STOP 位都正确，才触发唤醒/中断。

32.3.9. DMA 请求

LPUART 可以利用 DMA 连续通信

- LPUART DMA 接收初始化

根据 DMA 初始化配置流程，配置以下参数：

1）流控制和传输类型为外设到存储器。

2）源外设为 LPUART_RX。

3）目标外设为存储器。

4）源地址位宽为字节。

5）根据应用需要，目标地址位宽可为字节，可为半字，可为字。

6）禁止源地址递增。

7）根据应用需要，禁止/使能目标地址位递增。

8）源突发传输数量为 1。

9）目标突发传输数量为 1。

10）使能原始中断。

11）根据应用需要，配置中断使能位。

- LPUART DMA 接收

根据 DMA 传输配置流程，配置以下参数：

1）源地址为 LPUART_RXDR 寄存器地址。

2）根据应用需要，目标地址为合法的存储器地址。

3）配置通道链表地址为 NULL。

4）使能通道

32.3.10. 中断

LPUART 可以产生以下几种类型的中断：

	中断类型	中断使能位	描述
(lp)uart irq	start	LPUART_IE.STARTIE	起始位检测中断
	match	LPUART_IE.MATCHIE	地址匹配中断
	rxov	LPUART_IE.RXOVIE	接收 overrun 中断
	ferr	LPUART_IE.FEIE	接收帧格式错误(STOP 位)中断

	perr	LPUART_IE.PEIE	接收校验错误中断
	tx_if	LPUART_IE.TXEIE	发送 buffer 空中断
	tc_if	LPUART_IE.TCIE	数据发送完成中断
	rx_if	LPUART_IE.RXIE	字节接收完成中断

32.4. 配置流程

32.4.1. 初始化

1）配置时钟源（RCC_PERCFGR. LPUART1CLKSEL 和 RCC_PERCFGR. LPUART1DIV）。

2）禁止发送（LPUART_CR. TX_EN）、禁止接收（LPUART_CR. RX_EN）。

3）配置波特率（LPUART_IBAUD 和 LPUART_FBAUD）。

4）配置数据位宽（LPUART_LCR.WLEN）。

5）配置停止位位数（LPUART_LCR.STP2）。

6）配置校验位（LPUART_LCR.EPS、LPUART_LCR.SPS、LPUART_LCR.PEN）。

7）配置唤醒方式（LPUART_LCR.WKCK、LPUART_LCR.RXWKS、LPUART_ADDR）。

8）配置 IO 极性（LPUART_LCR.TXPOL、LPUART_LCR.RXPOL）。

9）禁止中断（LPUART_IE）。

10）复位标志位（LPUART_SR）。

11）使能发送（LPUART_CR. TX_EN）、使能接收（LPUART_CR. RX_EN）。

32.4.2. 发送

1）等待发送缓冲区空（LPUART_SR.TXEIF）置位时，向发送数据寄存器（LPUART_TXDR）写入发送数据。可以设置 LPUART_IE.TXEIE 位使能这一中断。

2）重复第 1 步骤，直到发送数据数据为 0。复位发送完成标志位（LPUART_SR.TCIF）。

3）等待发送完成标志位置位（LPUART_SR.TCIF）。

32.4.3. 接收

1）复位接收缓冲区满标志（读 LPUART_RXDR，并放弃所读数据，复位 LPUART_SR.RXF），开始接收数据。

2）等待接收缓冲区满标志（LPUART_SR.RXF）置位时，从接收数据寄存器（LPUART_RXDR）中读取接收数据。

3）重复第 2 步骤，直到接收数据全部完成。

32.5. LPUART 寄存器描述

32.5.1. 寄存器列表

LPUART 寄存器基地址： 0x40008000

偏移	名称	复位值	描述
0x00	LPUART_RXDR	0x00000000	接收数据寄存器
0x04	LPUART_TXDR	0x00000000	发送数据寄存器
0x08	LPUART_LCR	0x00030080	线控制寄存器
0x0C	LPUART_CR	0x00000000	控制寄存器
0x10	LPUART_IBAUD	0x00000102	波特率整数部分
0x14	LPUART_FBAUD	0x00000000	波特率整数部分
0x18	LPUART_IE	0x00000000	中断使能寄存器
0x1C	LPUART_SR	0x00000404	状态寄存器
0x20	LPUART_ADDR	0x00000000	地址寄存器

32.5.2. 接收数据寄存器(LPUART_RXDR: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RXDATA	RO	0	接收的数据

32.5.3. 发送数据寄存器(LPUART_TXDR: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	TXDATA	WO	0	发送的数据

32.5.4. 线控寄存器(LPUART_LCR: 08h)

位域	名称	属性	复位值	描述
31:16	BCNT_VALUE	RW	11	接受数据时，检测到上一个字符的停止位时，内部定时器计时的时长
15	AUTO_START_EN	RW	0	硬件自动计时控制 0：禁止 1：使能 在接受数据，如果使能此功能，在接收到 STOP 位时自动启动一次比特计时。
14:10	RSV	-	-	保留

9	TXPOL	RW	0	发送数据极性是否取反 0: 不取反 1：取反
8	RXPOL	RW	0	接收数据极性是否取反 0: 不取反 1：取反
7	WKCK	RW	1	接收唤醒校验选择 0：接收完 1 字节，不检查校验位和 STOP 位，直接触发唤醒/中断 1：接收完 1 字节，检查校验位和 STOP 位都正确，才触发唤醒/中断
6:5	RXWKS	RW	00	接收唤醒选择，STOP 模式用于唤醒选择 00：START 位检测唤醒 01：1byte 数据接收完成 10：接收数据匹配成功 11：无唤醒 STOP 模式下用于唤醒
4	WLEN	RW	0	字宽选择位 0：8bits 1：7bits
3	STP2	RW	0	停止位数选择位: 0：1 位停止位 1：2 位停止位
2	EPS	RW	0	0/1 校验或者奇/偶校验选择位（取决于 SPS） 0：奇/偶校验选择奇校验，或 0/1 校验选择校验位强制为 1 1：奇/偶校验选择偶校验，或 0/1 校验选择校验位强制为 0
1	SPS	RW	0	校验模式选择位 0：奇/偶校验 1：0/1 校验
0	PEN	RW	0	校验使能位: 0：禁止奇/偶校验或 0/1 校验 1：使能奇/偶校验或 0/1 校验

32.5.5. 控制寄存器(LPUART_CR: 0Ch)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
2	DMA_EN	RW	0	使能 DMA 功能
1	TX_EN	RW	0	发送使能位 0：禁止 1：使能

0	RX_EN	RW	0	接收使能位： 0：禁止 1：使能

32.5.6. 波特率整数部分(LPUART_IBAUD: 10h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
15:8	RXSAM	RW	0x1	接收采样点设置，一般设为 IBAUD>>1, 可以略微调整。
7:0	IBAUD	RW	0x2	波特率分频整数因子-1，设置范围为 2~254 IBAUD = (integer(Fsys/BAUD))-1, integer 为取整操作 最小值为 2，即 3 分频；最大值为 254，即 256 分频。 结合小数部分，波特率最大为 3~4 分频之间，最小值为 255~256 之间。

32.5.7. 波特率小数部分(LPUART_FBAUD: 14h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	FBAUD	RW	0x000	波特率小数部分，设置 1 字节内的每一个 bit 是否调整。从低位到高位对应为 Start 到 STOP。 0：不调整 1：增加一个时钟。 这个比较麻烦，需要增加一张常用波特率表 小数部分的加权平均为 (Fsys/BAUD) -1- IBAUD

32.5.8. 中断使能寄存器(LPUART_IE: 18h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	IDLEIE	R/W	0	IDLE 中断使能位 0：禁止 1：使能
16	BCNTIE	R/W	0	Bit Count Timeout 中断使能位 0：禁止 1：使能
15:11	RSV	-	-	保留

9	STARTIE	RW	0	起始位检测中断使能位 0：禁止 1：使能
8	MATCHIE	RW	0	地址匹配中断使能位 0：禁止 1：使能
7:6	RSV	-	-	保留
5	RXOVIE	RW	0	接收 overrun 中断使能位 0：禁止 1：使能
4	FEIE	RW	0	接收帧格式错误(STOP 位)中断使能位 0：禁止 1：使能
3	PEIE	RW	0	接收校验错误中断使能位 0：禁止 1：使能
2	TXEIE	RW	0	发送 buffer 空中断使能 0：禁止 1：使能
1	TCIE	RW	0	数据发送完成中断使能位 0：禁止 1：使能
0	RXIE	RW	0	字节接收完成中断使能位 0：禁止 1：使能

32.5.9. 状态寄存器(LPUART_SR: 1Ch)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	IDLEIF	RO	0	监测总线空闲原始中断 0：没有检测到空闲总线 1：检测到空闲总线 写 1 清 0
16	BCNTIF	RO	0	Bit Count Timeout 原始中断 0：无中断 1：有中断 写 1 清 0
15:10	RSV	-	-	保留
10	TXE	RO	1	发送缓存空标志。

9	STARTIF	RO	0	起始位检测中断标志，写 1 清零
8	MATCHIF	RO	0	地址匹配中断标志，表示接收缓冲区内的数据与地址寄存器相同，写 1 清零 可以设置 WKCK 位选择是否检查校验和 STOP 位
7	TXOVF	RO	0	TXDR 溢出错误，软件写 1 清零 当 TXDR 满时，软件又向 TXDR 写入新数据，该位置 1
6	RXF	RO	0	接收缓冲满，读 RXDR 清零，不产生中断 注：不管校验是否正确
5	RXOVIF	RO	0	接收 overrun 中断标志，写 1 清零 0：无中断 1：有中断 注：不管校验是否正确
4	FEIF	RO	0	帧格式错误（STOP）中断标志，写 1 清零 0：无中断 1：有中断
3	PEIF	RO	0	校验错误中断标志，写 1 清零 0：无中断 1：有中断
2	TXEIF	RO	1	发送 buffer 空中断标志，写入 TXDR 后清零 0：无中断产生 1：发送 buffer 空
1	TCIF	RO	0	数据发送完成中断标志，TXDR 为空并且当前字节发送完成。写 1 清 0：无中断 1：有中断
0	RXIF	RO	0	字节接收完成中断标志，写 1 或读取 RXDR 时清零 1：接收完一帧数据后中断产生 0：无中断产生

32.5.10. 地址寄存器(LPUART_ADDR: 20h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	ADDR	R/W	0	匹配地址设置

33. 内部集成电路总线接口（I2C）

33.1. 概述

I2C 总线是连接微控制器和其他集成电路芯片之间的串行总线。它有两根线，一根是时钟信号线 SCL，另一根是数据信号线 SDA。芯片上的 I2C 接口模块通过数据引脚 SDA 和时钟引脚 SCL 连接到 I2C 总线上，控制所有 I2C 总线规定的时序。I2C 模块可配置成主模式（支持多主机功能）或从模式，它支持标准、快速和快速增强三种速率，同时兼容 SMBus2.0。

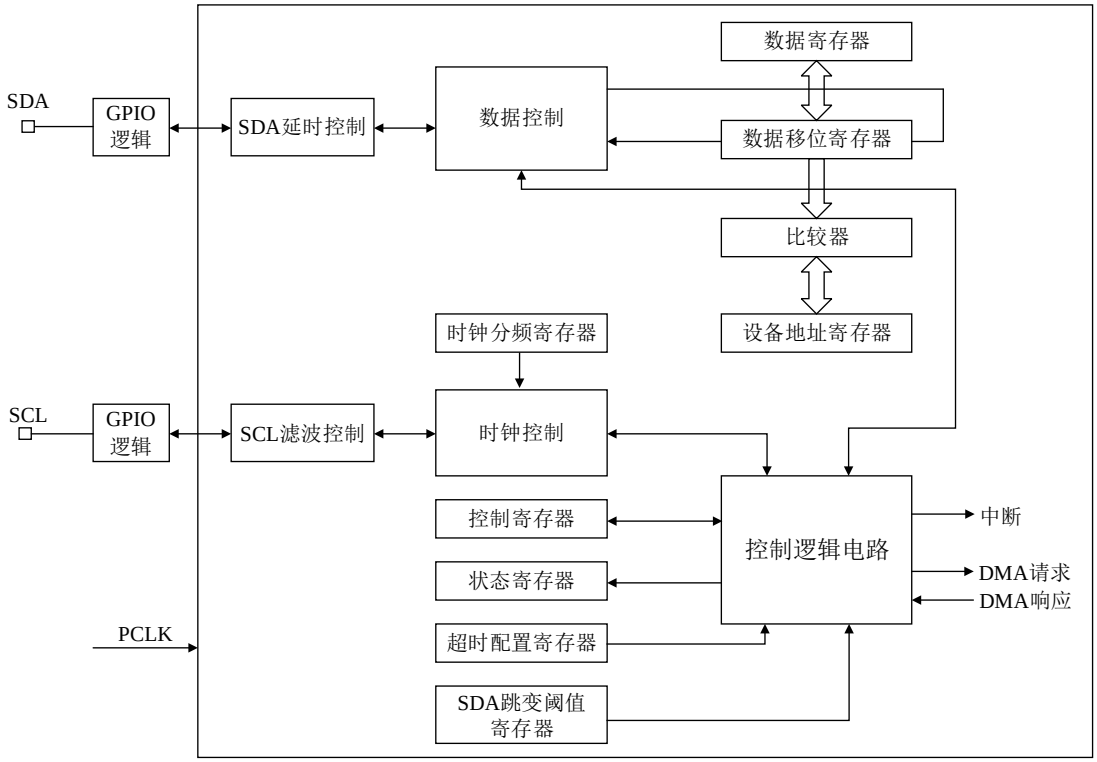
根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

33.2. 主要特性

- 支持主模式和从模式
- 支持多主机模式，支持仲裁机制
- I2C 主设备功能：
  - 生成时钟
  - 起始位和停止位生成
- I2C 从设备功能：
  - 可编程的 I2C 从设备地址
  - 支持 7bit 设备地址，支持多个从设备地址
  - 可编程的 NACK/ACK 回复
- 支持不同的通讯速度
  - 标准（高至 100KHz）
  - 快速（高至 400KHz）
  - 快速增强（高至 1MHz）
- 支持从机拉时钟功能
- 支持 DMA 收发数据
- 兼容 SMBus2.0

33.3. 结构框图

图 33-1 I2C 结构框图



33.4. 功能描述

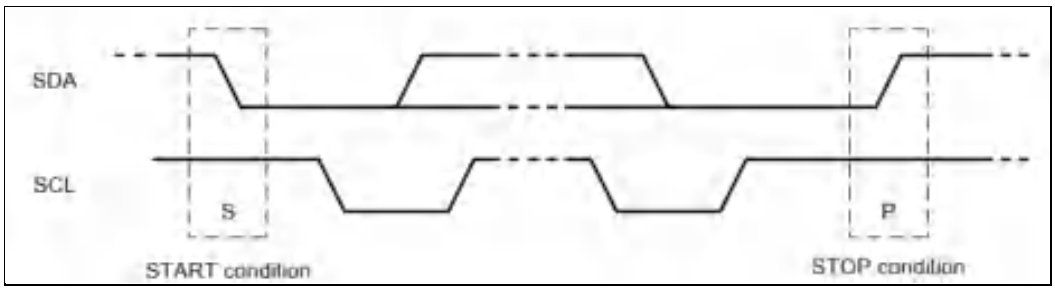
I2C 模块将数据从串行转换成并行（接收），或并行转换成串行（发送）。I2C 模块通过数据引脚 SDA 和时钟引脚 SCL 连接到 I2C 总线。它可以连接到标准（高至 100KHz）、快速（高至 400KHz）或快速增强（高至 1MHz）的 I2C 总线上。

该接口也可通过数据引脚 SDA 和时钟引脚 SCL 连接到 SMBus。如果支持 SMBus，还可使用一个 GPIO 作为 SMBus 报警引脚。

33.4.1. 开始和停止条件

所有的数据传输起始于一个 START 结束于一个 STOP。

图 33-2 I2C 起始和停止条件



START 条件定义为：在 SCL 为高时，SDA 线上出现一个从高到低的电平转换。

STOP 条件定义为：在 SCL 为高时，SDA 线上出现一个从低到高的电平转换。

33.4.2. 模式选择

I2C 模块默认为从模式，设置 I2C_CR.MASTER 为 1 时设备变成主模式。

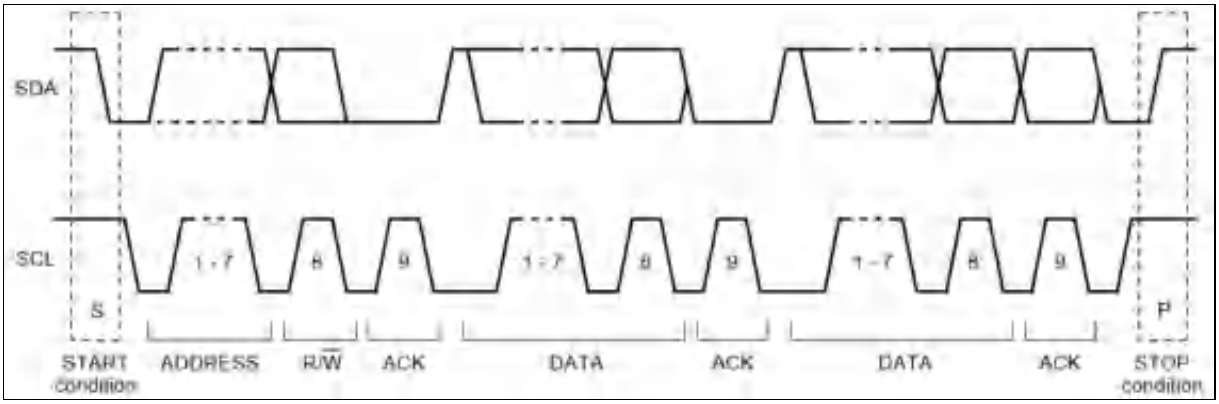
主模式时，I2C 接口启动数据传输并产生时钟信号，并可以发出停止条件信号停止传输。

从模式时，I2C 接口能识别设置的设备地址（7 位）。

数据和地址按 8 位/字节进行传输，高位(MSB)在前。跟在起始条件后的是地址。地址只在主模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位（ACK）给发送器。参考下图，一个完整的 I2C 数据传输：

图 33-3 I2C 总线传输

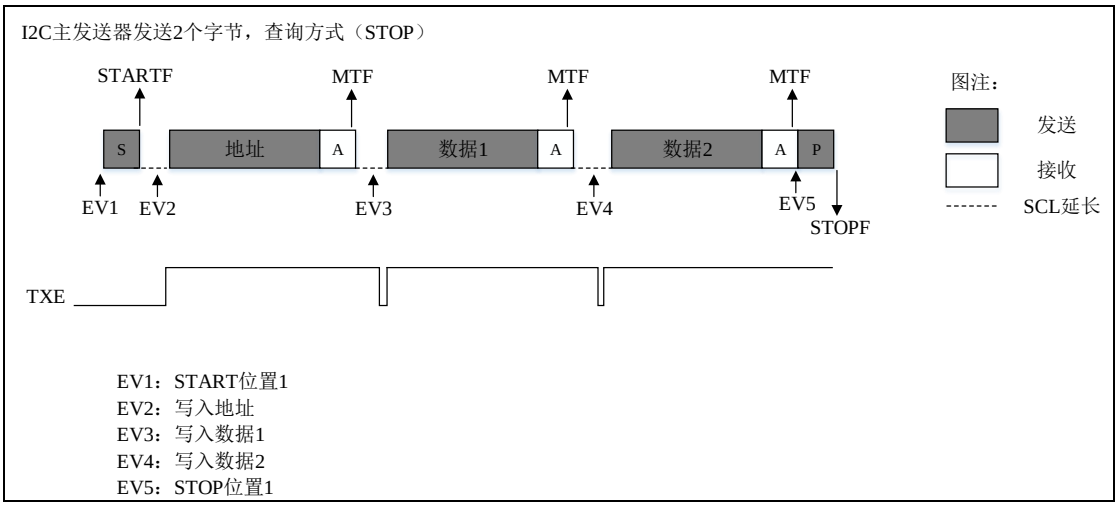


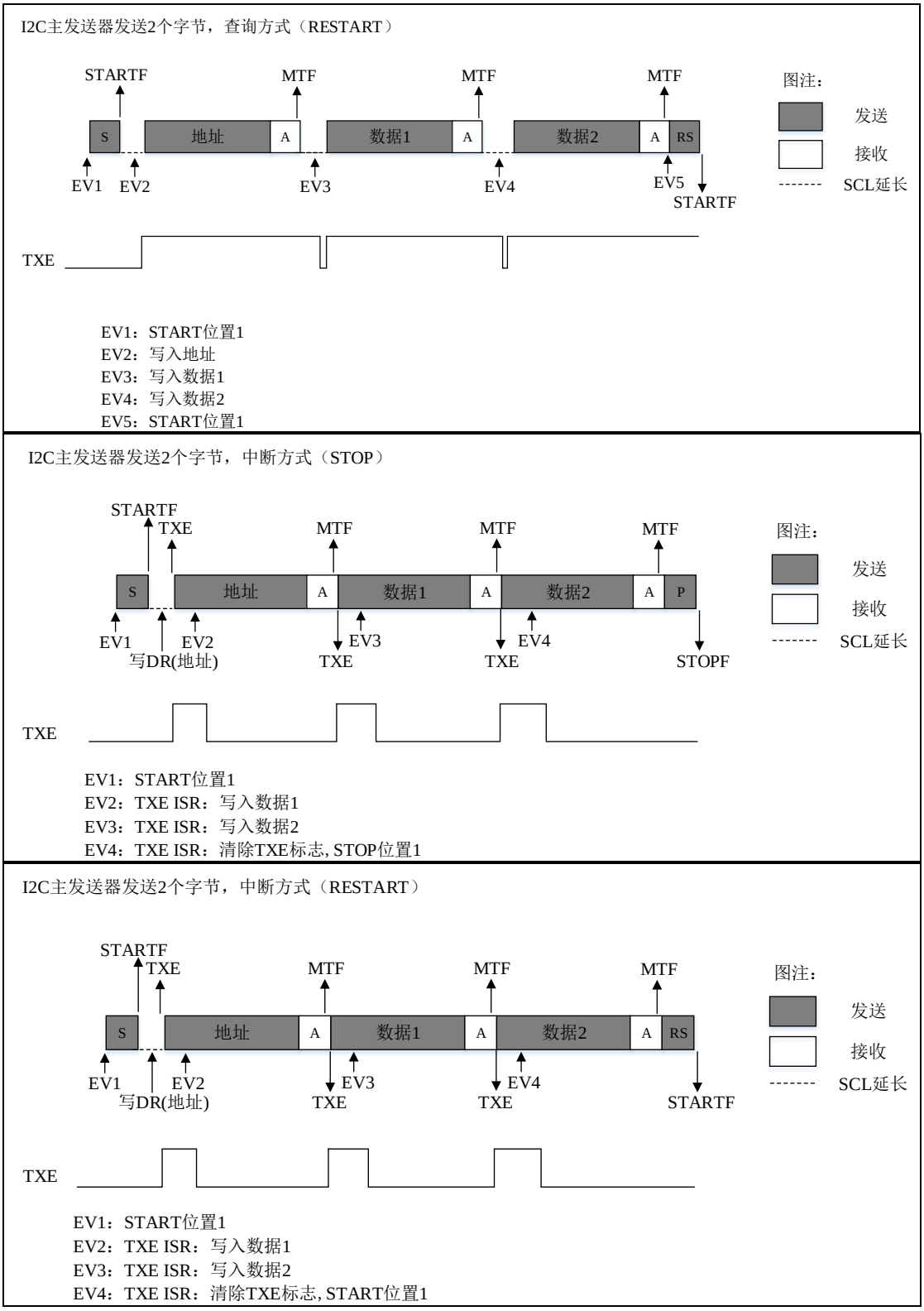
33.4.3. 从机地址 SLAVE_ADDR1/2/3

I2C 接口包含三个从机地址。SLAVE_ADDR1 默认使能，软件配置 ADDR1[7:1]后，从机地址 1 生效。SLAVE_ADDR2 和 SLAVE_ADDR3 包含使能位。软件配置 ADDR2[7:1]和 ADDR3[7:1]后，需要将使能位 ADDR2_EN 和 ADDR3_EN 置 1 从机地址 2 和 3 才可以生效。

33.4.4. 主模式发送

图 33-4 I2C 主模式发送序列图





在主模式时，I2C 接口启动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并以停止条件结束。当写控制寄存器的 START 位为 1 在总线上发起一次起始条件，设备就进入了主模式传输。

以下是主模式的操作顺序：

- 使能设备，配置时钟分频寄存器和主模式。
- 配置控制寄存器的 START 位为 1，产生起始条件。
- 写入数据寄存器，开始发送数据。

■ SCL 主时钟生成

I2C_CLK_DIV 位用于生成 SCL 时钟的高电平和低电平。正常情况下。高电平和低电平的长度应该相等。由于主器件和从器件可能会延长 SCL 线。内部计数器在产生 SCL 时钟时会检查是 SCL 拉低信号是否有效。如果 SCL 拉低信号有效，则延长 SCL 时钟低电平，直到 SCL 拉低信号消失。

■ 启动条件

当 I2C 总线空闲时，START 位置 1 后，I2C 接口会生成一个起始位。

注：在主设备下，START 位置 1 后，接口会在当前字节传输结束后生成一个重复起始位。

起始位发出之后，STARTF 位会由硬件置 1。然后把从设备地址写入数据寄存器。如果没有写入的动作，这时 I2C 接口保持 SCL 为低以等待设备地址被写进数据寄存器。

■ 从地址传输

接下来从地址会通过内部移位寄存器发送到 SDA 线上。在 7 位寻址模式下，会发送一个地址字节。地址字节被发出后，主设备会根据发送的从地址字节读写方向位来决定是进入发送模式还是接收模式。

要进入发送模式，将要发送的从地址字节中的 R/W 位置 0。

要进入接收模式，将要发送的从地址字节中的 R/W 位置 1。

■ 主发送器

发送了地址后，主设备通过内部移位寄存器将字节从数据寄存器加载到 SDA 线上，并将 I2C_SR.TXE 置 1 表示数据寄存器数据已被取走，软件需要更新数据寄存器来清除 I2C_SR.TXE 标志。

收到应答脉冲后确认新的数据已经发送到数据寄存器。如果在上一个数据发送结束之前新数据仍然没有被写进数据寄存器，即 I2C_SR.TXE 仍然为 1，这时 I2C 接口保持 SCL 为低以等待新的数据被写进数据寄存器。

主设备设置 STOP 位产生停止条件。

图 33-5 7 位主发送器的传送图

说明：S=Start(起始条件)， P=Stop(停止条件)，A=响应，NA=非响应

:主机到从机　　:从机到主机

■ 主接收器

发送了地址后，TX_RX_FLAG 会置 1，软件需要清除 TX_RX_FLAG 来产生 SCL 时钟。I2C 接口从 SDA 线接收数据字节，并通过内部移位寄存器存储到数据寄存器，产生数据寄存器非空标志 I2C_SR.RXNE，软件需要读出数据寄存器的值来清除 I2C_SR.RXNE 标志。

若当前字节接收完成（I2C_SR.RXNE=1），不去读取数据清除 I2C_SR.RXNE，这时 I2C 接口保持 SCL 为低以等待数据寄存器的值被读出（数据寄存器和移位寄存器均为满）。

读出数据后，寄存器非空标志 I2C_SR.RXNE 清 0，主机开始一次新的传输。

主机针对自从设备接收的最后一个字节发送 NACK。在接收到此 NACK 之后，从设备会释放对 SCL 和 SDA 线的控制。随后，主设备可发送一个停止位/重复起始位。

1）为了在最后一个接收数据字节后生成非应答脉冲，必须在读取倒数第二个数据字节后（倒数第二个 RXNE 事件之后）立即将 TACK 位置 1。

2）要生成停止位/重复起始位，软件必须在读取倒数第二个数据字节后（倒数第二个 RXNE 事件之后）将 STOP/START 位置 1。

3）在只接收单个字节的情况下，需要在 TX_RX_FLAG 标志清零之后生成 TACK 位和停止位。

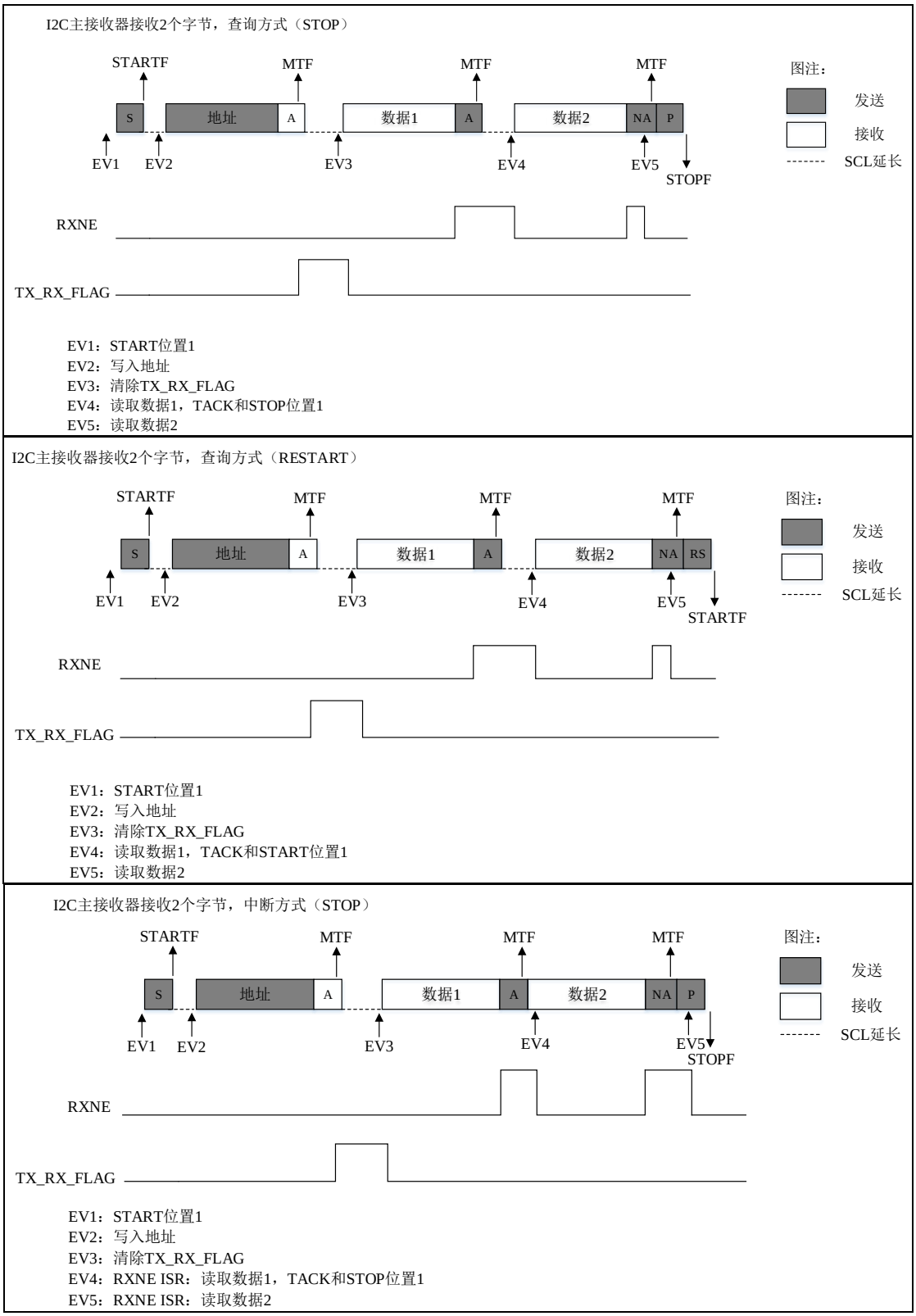
图 33-6 7 位主接收器的传送图

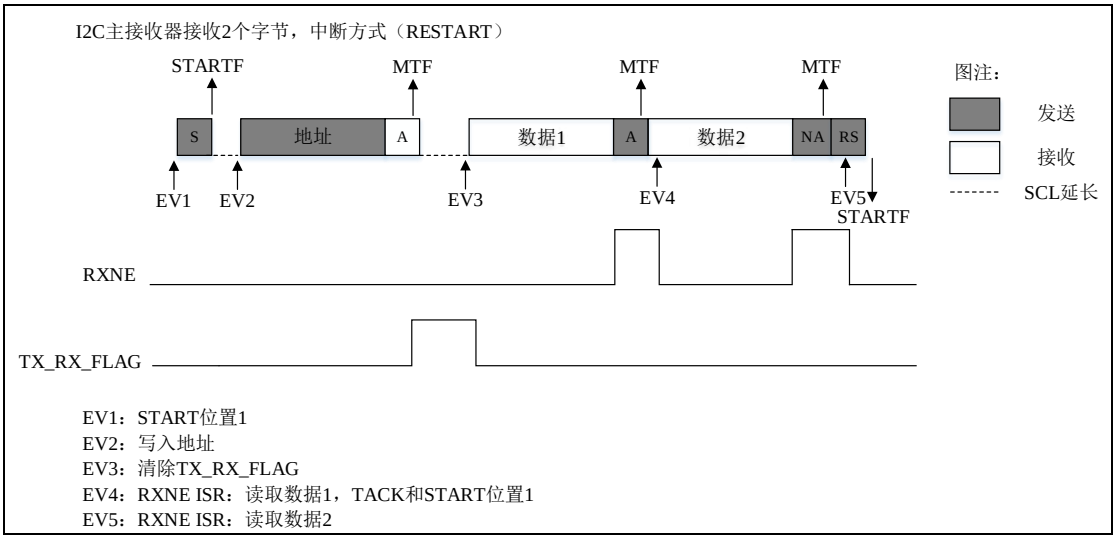
说明：S=Start(起始条件)， P=Stop(停止条件)，A=响应，NA=非响应

:主机到从机 :从机到主机

33.4.5. 主模式接收

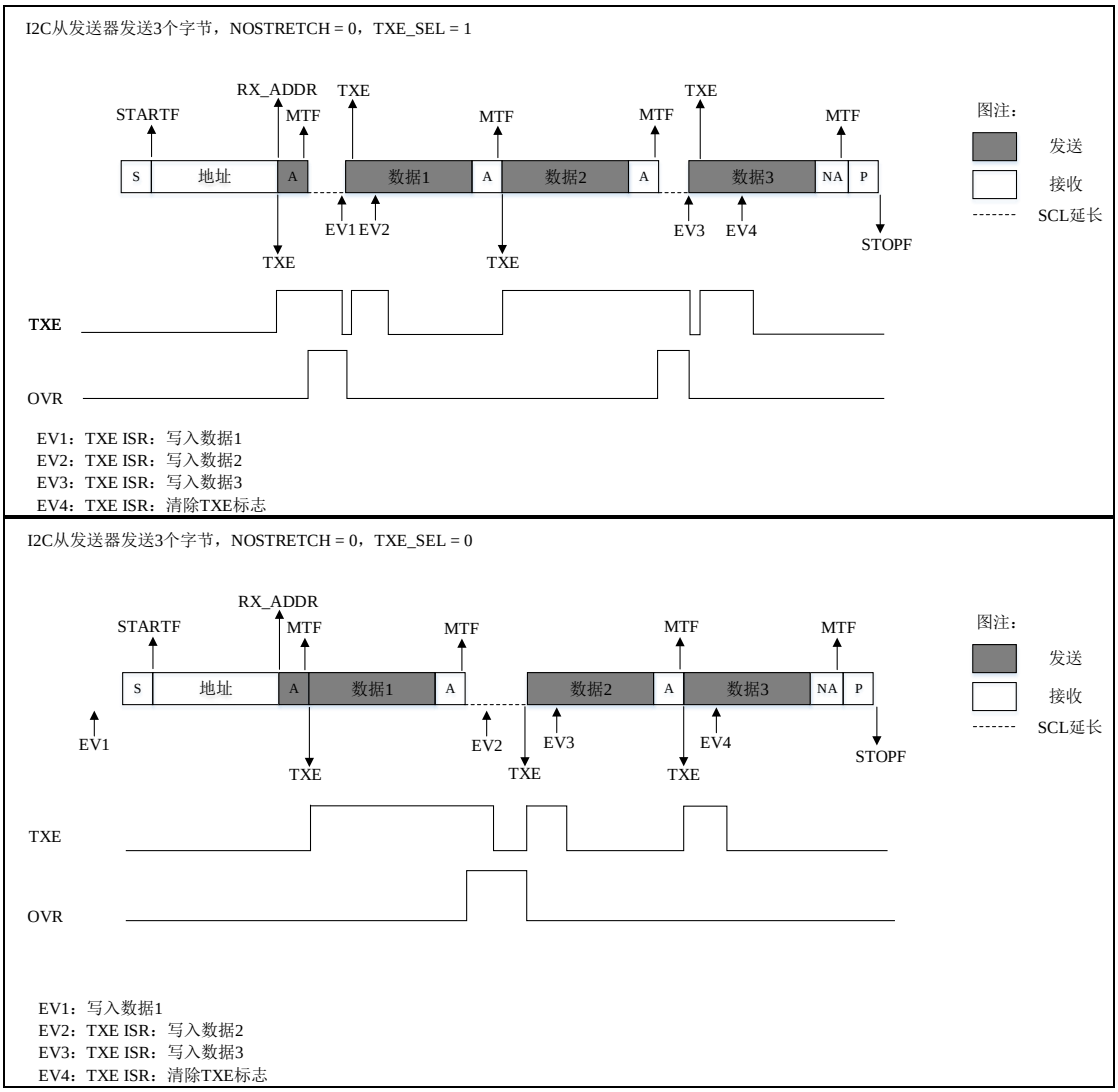
图 33-7 I2C 主模式接收序列图

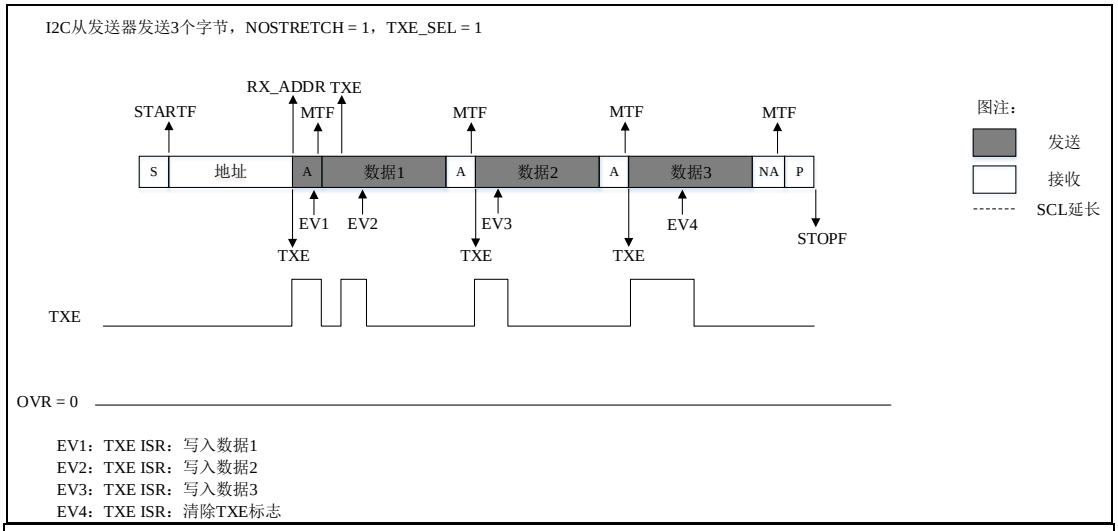


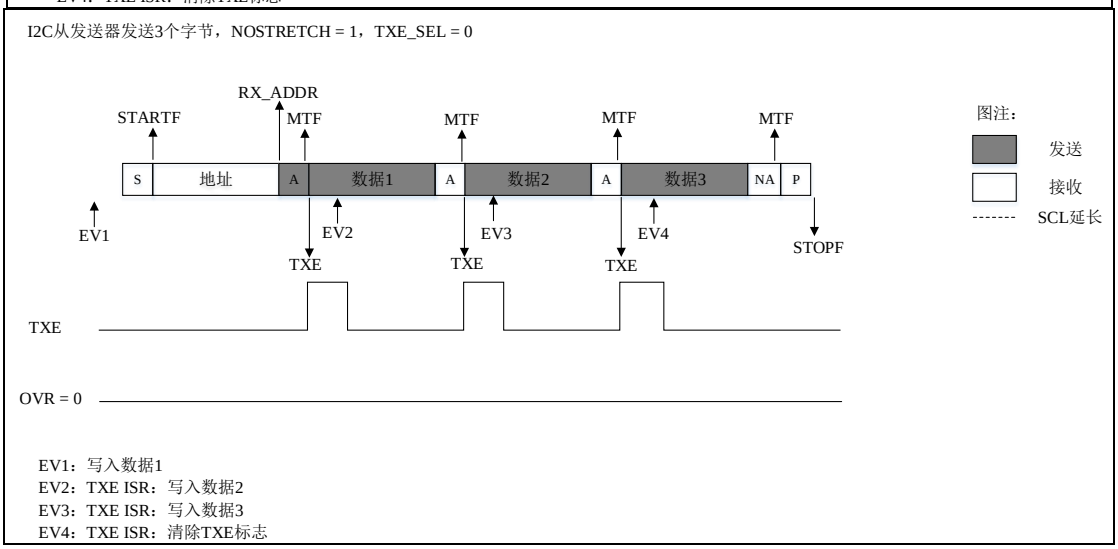


33.4.6. 从模式发送

图 33-8 I2C 从模式发送序列图







一旦检测到起始条件，在 SDA 线上接收到的地址被送到移位寄存器。然后与芯片自己的设备地址相比较，如果地址不匹配 I2C 将其忽略并等待另一个起始条件。如果地址匹配，TACK 需为 0，产生 ACK 应答。此控制器还会检测当前操作是发送还是接收（I2C_SR.SRW 位），I2C 接口进行如下操作：

■ 从发送器

发送器将字节从数据寄存器加载到内部移位寄存器发送到 SDA 线上，并将状态寄存器 TXE 置 1 表示数据寄存器数据已被取走，软件需要更新数据寄存器来清除 TXE 标志。

当收到应答脉冲后，如果在下一个数据的时钟到来之前该数据仍然没有被写进数据寄存器，即 TXE 仍然为 1，则从机上溢/下溢状态位（OVR）被置 1（数据寄存器和移位寄存器均为空）。此时如果 NOSTRETCH 为 1，则从机不会延长 SCL 时钟，当主机发起新的读时序时，从机数据寄存器的数据将不会再次发送给主机。此时如果 NOSTRETCH 为 0 且收到 ACK 应答，则 I2C 接口保持 SCL 为低以等待新的数据被写进数据寄存器。如果收到 NACK 应答，则 I2C 接口会释放对 SCL 和 SDA 的控制。

图 33-9 7 位从发送器的传送图

说明：S=Start(起始条件)， P=Stop(停止条件)，A=响应，NA=非响应

:主机到从机

:从机到主机

■ 从接收器

在接收到数据后，从接收器将通过内部移位寄存器从 SDA 线接收到的字节存储到数据寄存器，并产生数据寄存

器非空标志 RXNE，软件需要读出数据寄存器的值来清除 RXNE 标志。

I2C 接口在接收到每个字节后都产生一个应答脉冲。

如果在下一个数据接收结束之前数据寄存器的值未被读出，即 RXNE 仍然为 1，则从机上溢/下溢状态位被置 1（数据寄存器和移位寄存器均为满）。此时如果 NOSTRETCH 为 0，这时 I2C 接口保持 SCL 为低以等待数据寄存器的值被读出；否则主机将继续传输数据，新收到的数据将被不会被写到数据寄存器中，原来的数据会保留。

图 33-10 7 位从接收器的传送图

说明：S=Start(起始条件)， P=Stop(停止条件)，A=响应，NA=非响应

:主机到从机

:从机到主机

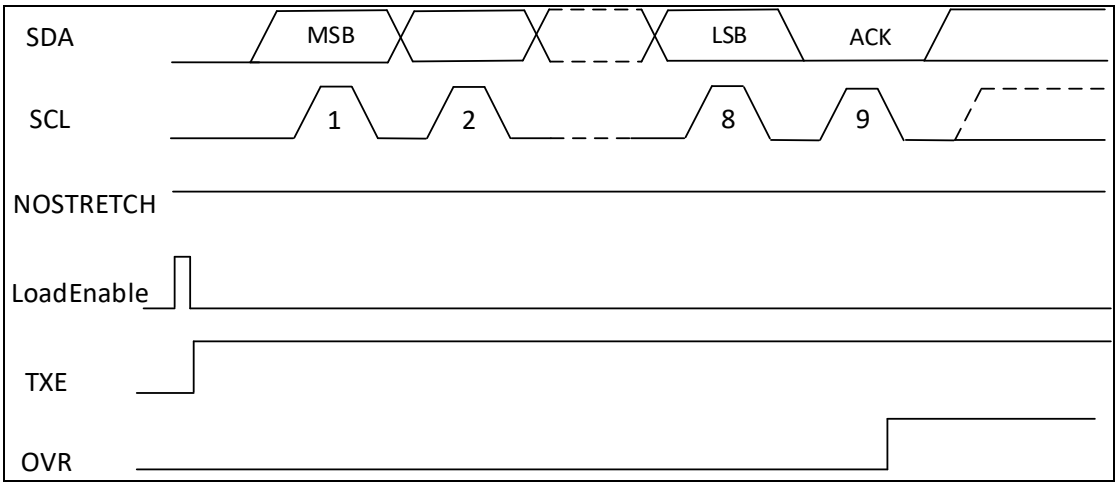
■ 关闭从通信

在传输完最后一个数据字节后，主设备发出一个停止条件，I2C 接口检测到这一条件时释放 SCL 和 SDA 线。

■ 时钟延长

- 发送模式：当数据寄存器里的数据没有被更新时且主机应答为 ACK，把 SCL 拉低以等待新的数据写入。当主机应答为 NACK 时，SCL 不会被拉低
- 接收模式：当数据寄存器里的数据没有被读走时，把 SCL 拉低以等待旧的数据被读走。
- SCL 拉低功能可以通过 I2C_CR 的 NOSTRETCH 禁止。

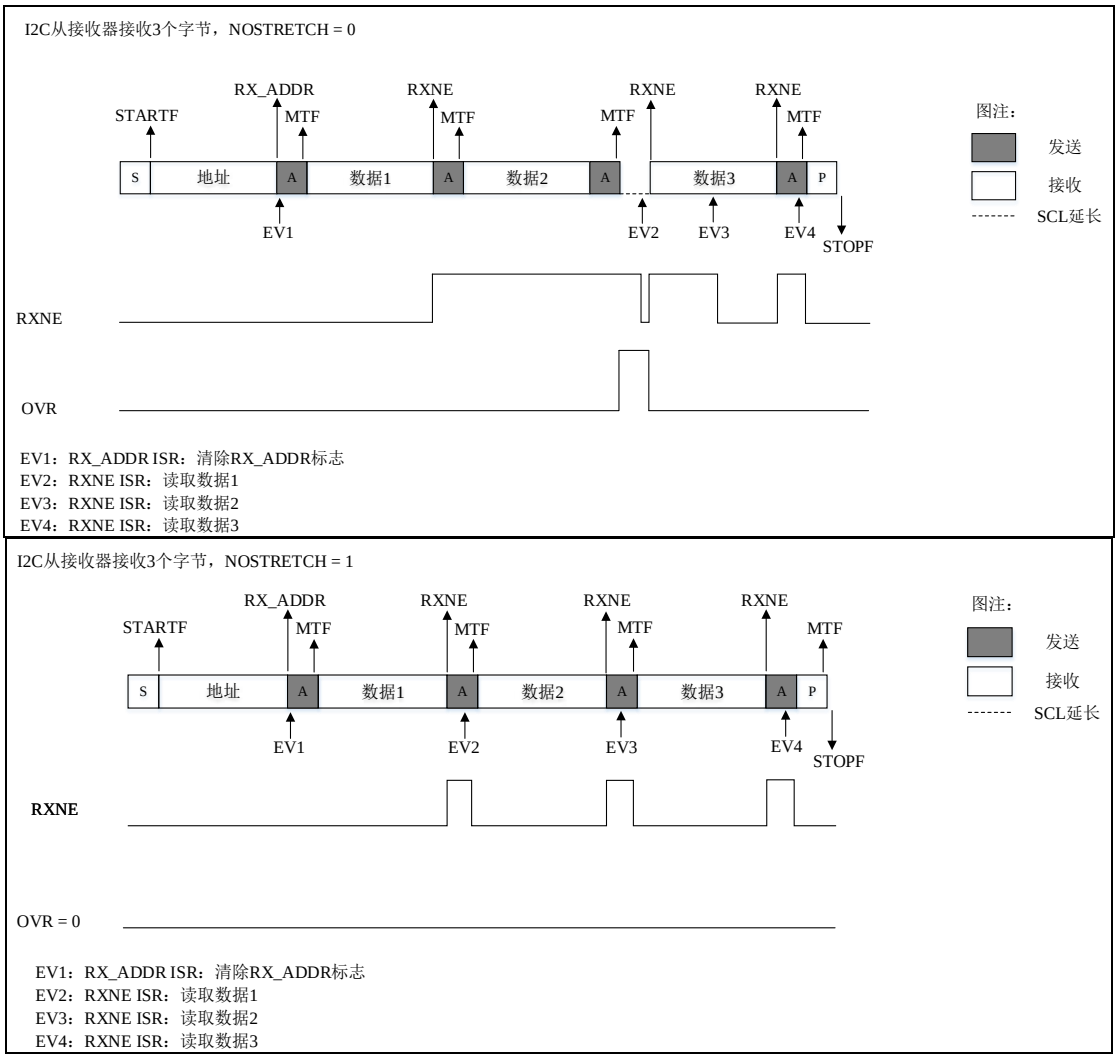
图 33-11 从机发送模式时钟延长时序



虚线部分 SCL 为低，主机无法发送 SCL。

33.4.7. 从模式接收

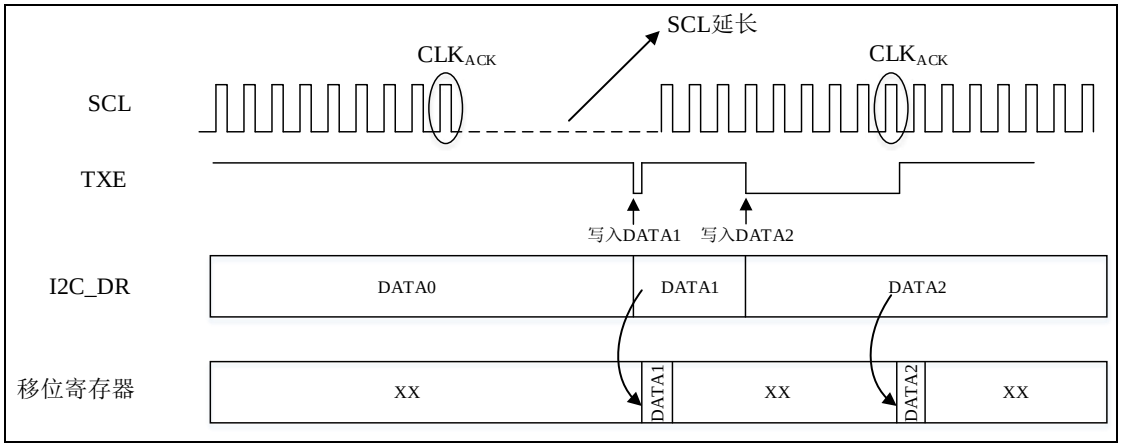
图 33-12 I2C 从模式接收时序



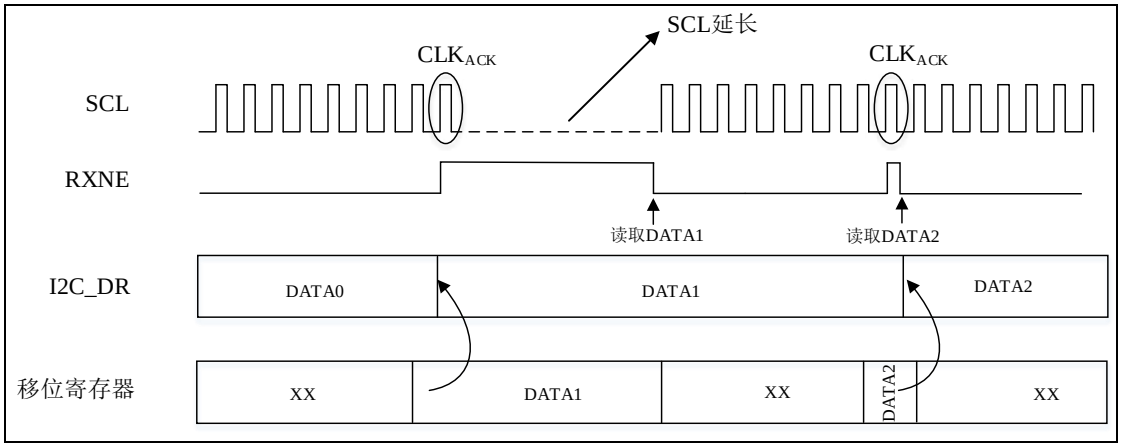
33.4.8. 时钟 SCL 延长

■ 主机主动延长 SCL

➢ 主发送模式下， 如果在上一次数据传输结束之前 TXE 位已置 1 但数据字节尚未写入数据寄存器，主机会延长 SCL 时钟，直到新的数据被写进数据寄存器。



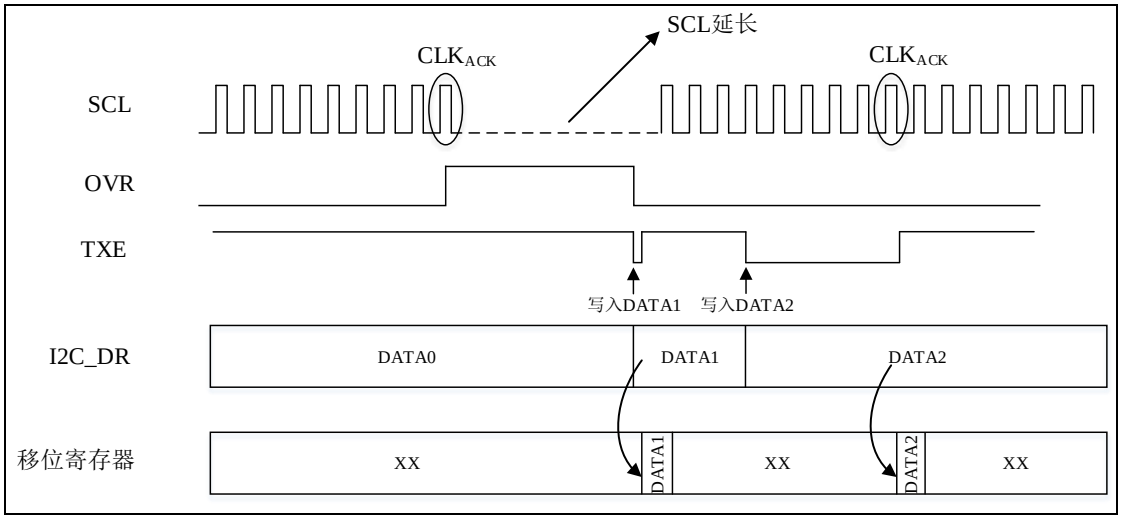
➢ 主接收模式下，主机接收到一个数据后（RXNE=1），如果不读取数据寄存器清除 RXNE，这时 I2C 保持 SCL 为低以等待数据寄存器的数据被读出。



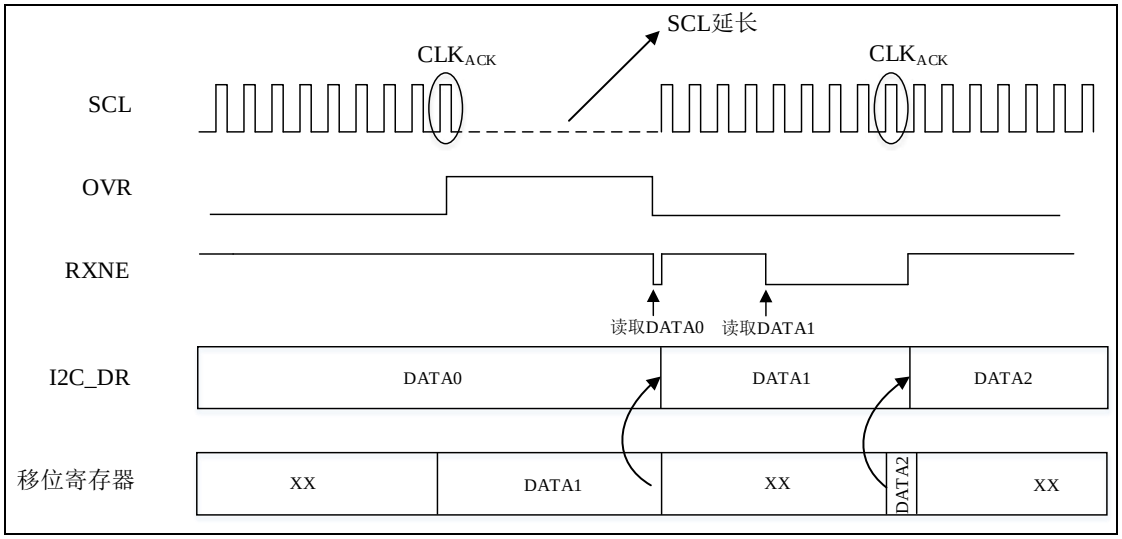
■ SCL 被从机延长(需满足 NOSTRTCH=0):

当设置 I2C_CR. NOSTRETCH 为 0 时，I2C 从机支持拉低并延长 SCL；否则从机不会拉低延长 SCL。

- 从机发送模式下，如果在下一个数据的时钟到来之前 TXE 位已置 1 但数据尚未写入数据寄存器，则 OVR 置 1。此时，如果收到的是 ACK 信号，则从机会拉低并延长 SCL 时钟，直到新的数据被写进数据寄存器。



- 从机接收模式下，如果在下一个数据接收完成之前 RXNE 位已置 1 但数据寄存器中的数据尚未读取，则 OVR 置 1。从机会拉低并延长 SCL 时钟，直到数据寄存器中的数据被读出。



33.4.9. 错误条件

以下错误条件可能导致通信失败。

■ 主模式仲裁丢失（MARLO）

当主机在 SDA 线上发送高电平但在 SCL 的上升沿却采样到低电平时，会检测主机仲裁丢失。在这种情况下：

- I2C_SR 寄存器中的 MARLO 位会由硬件置 1，如果 I2C_CR 寄存器中的 MARLO_INT_EN 位置 1，将生成中断。
- I2C_CR 寄存器中的 MAROL_SEL 位如果置 1，主机仲裁丢失时会自动切换成从机。

■ 从机上溢/下溢（OVR）

I2C 接口在接收数据时，从模式中可能出现上溢错误。I2C 从机已经接收到一个字节（RXNE=1），但是收到下一个字节之前 I2C_DR 中的数据未被取走，在这种情况下：

- I2C_SR 寄存器中的 OVR 位会由硬件置 1。如果 I2C_CR 寄存器中的 OVR_INT_EN 位置 1，将生成中断。
- 会丢失接收的最后一个字节（NOSTRETCH=1）
- I2C 接口保持 SCL 为低以等待 I2C_DR 寄存器的值被读出（NOSTRETCH=0）

I2C 接口在发送数据时，从模式中可能出现下溢错误。当收到上一个字节的应答脉冲后，下一个字节的时钟信号到来之前，从机还未把下一个要发送的字节写进 I2C_DR（TXE=1）。在这种情况下：

- I2C_SR 寄存器中的 OVR 会由硬件置 1，如果 I2C_CR 寄存器中的 OVR_INT_EN 位置 1，将生成中断。
- 当主机发起新时序时，从机数据寄存器中的数据不会再次发送给主机（NOSTRETCH=1）。
- 当主机发起新时序时，I2C 接口保持 SCL 为低以等待新的数据被写进 I2C_DR（NOSTRETCH=0）。

■ 超时错误（TIMEOUT）

仅当支持 SMBus 时功能时，才涉及本节内容。

满足以下任何条件均会出现超时错误：

- SCL 的低电平持续时间达到 TIMEOUTA[11:0]位中定义的时间，这用于检测 SMBus 超时。
- EXT_MODE=1 且主器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0]位中定义的时间（SMBus tLOW:MEXT 参数）
- EXT_MODE=0 且从器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0]位中定义的时间（SMBus tLOW:SEXT 参数）

检测到超时错误时，I2C_SR 寄存器中的 TIMEOUTAF 或者 TIMEOUTBF 将由硬件置 1。如果 I2C_TIMEOUT 寄存器中的 TOUA_INT_EN 或者 TOUTB_INT_EN 位置 1，将生成中断。

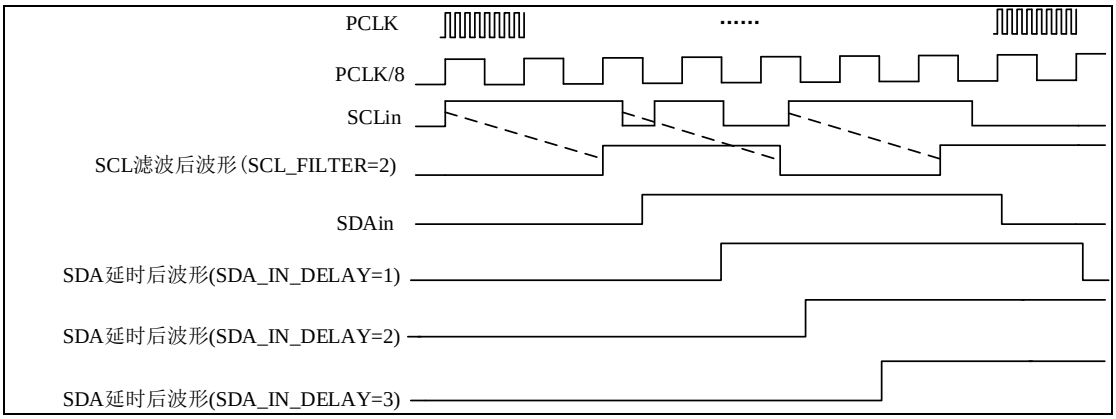
33.4.10. SCL 总线滤波算法

当 I2C_FILTER 寄存器的 SCL_FILTER 值为 0 时，表示 SCL 没有滤波功能。不为 0 时滤波时间为 Tcntc*SCL_FILTER。其中 Tcntc 为 PCLK 的 8 分频时钟周期。

例：

SCL_FILTER=2 时，SCL 必须采样到连续两个 Tcntc 宽度的高电平才能输出高电平，宽度小于两个 Tcntc 的脉冲被认为是干扰毛刺而被过滤掉。

图 33-13 滤波算法示意图



注：

在使用过程中，设置 I2C_FILTER.SDA_IN_DELAY 与 I2C_FILTER.SCL_FILTER 的值相同，SDA 信号与 SCL 信号在经过滤波后，将保持输入时的相位。

SCL 滤波功能会滤除滤低于 Tcntc*SCL_FILTER 时间的毛刺。

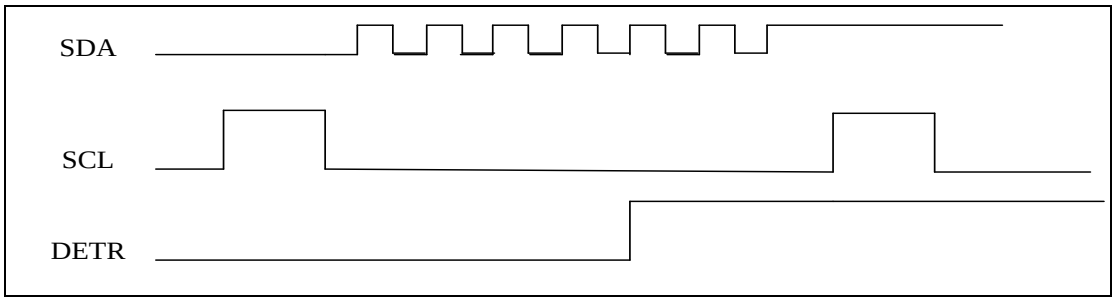
33.4.11. SCL 为低时检测 SDA 的跳变

当 I2C_DET 寄存器的值为 0 时，表示不使能检测功能，为其他值时表示使能检测功能，并且当 SCL 为低电平时检测到 SDA 的上升沿跳变次数大于等于 I2C_DET 寄存器的值时，把 I2C_SR.DETR 位置 1。

例：

当 I2C_DET=5 时，在 SCL 低电平时，内部计数器开始工作，当检测到 SDA 上升沿跳变累积大于等于 5 时，I2C_SR.DETR=1；当 SCL 变成高电平时，内部计数器清零，等待 SCL 下个低电平到来时再开始计数。如下图所示：

图 33-14 SDA 跳变检测图



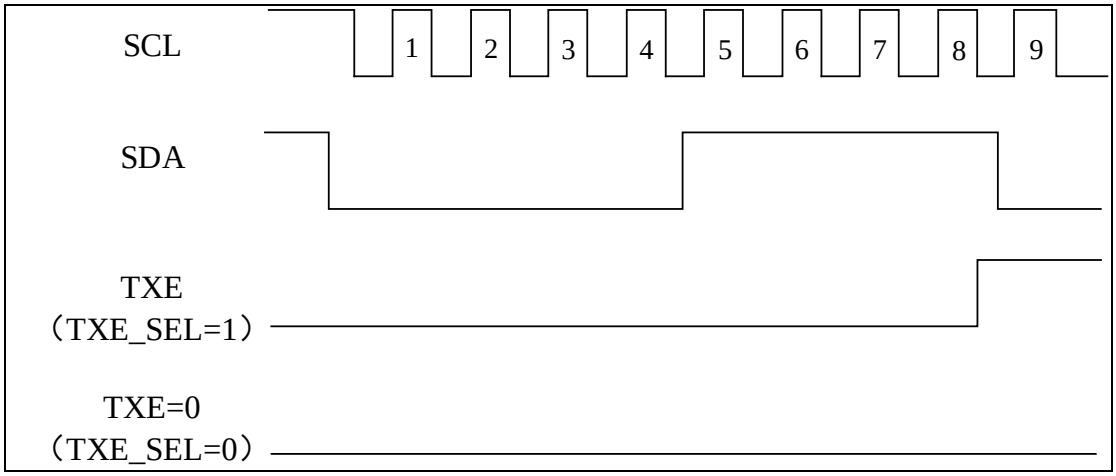
33.4.12. TXE_SEL

当主模式接收从模式发送时：

TXE_SEL 为 1，当从机匹配了主机发送的地址，TXE 置 1。

TXE_SEL 为 0，当从机匹配了主机发送的地址，TXE 为 0。

如下图，从机匹配了主机发送的 7 比特地址（7’ b0000111）时，TXE 在不同 TXE_SEL 下的状态变化。



33.4.13. SMBus 的功能特性

系统管理总线（SMBus）是一个双线制接口，各器件可通过它在彼此之间或者与系统的其余部分进行通信。它以 I2C 的工作原理为基础。SMBus 可针对系统和电源管理相关的任务提供控制总线。系统可使用 SMBus 与设备进行消息传递，而无需切换各个控制线。

系统管理总线规范涉及三类器件。从器件，用于接收或响应命令。主器件，用于发出命令、生成时钟和终止传输。主机，专用的主器件，可提供连接系统 CPU 的主接口。主机必须具有主-从设备功能，并且必须支持 SMBus 主机通知协议，系统中只允许存在一个主机。

■ SMBus 与 I2C 的相似之处

- 双线制总线协议（1 个时钟总线，1 个数据总线） + 可选 SMBus 报警线
- 主从通信，主器件提供时钟
- 多主器件功能
- SMBus 数据格式与 I2C 7 位地址格式相似

■ SMBus 与 I2C 之间的差异

SMBus	I2C
最大速度 100KHz	最大速度 400KHz
最小速度 10KHz	无最小时钟速度
35ms 时钟低电平超时	无超时
逻辑电平固定	逻辑电平取决于 VDD
地址类型不同（保留、动态等）	7 位、10 位和广播从模式地址类型
总线协议不同（快速命令、过程调用等）	无总线协议

■ SMBus 应用用途

通过系统管理总线，器件可以提供制造商信息、告诉系统它的型号/部件号、保存暂停事件的状态、报告不同的错误类型、接受控制参数并返回其状态。SMBus 可针对系统和电源管理相关的任务提供控制总线。

■ 器件标识

系统管理总线中作为从器件的任何器件均具有一个唯一地址，被称为从地址。有关保留的从地址列表的信息，请参见 SMBus 规范版本 2.0 (http://smbus.org/specs/)。

■ 总线协议

SMBus 规范支持多达 9 类总线协议。有关这些协议和 SMBus 地址类型的详细信息，请参见 SMBus 规范版

本 2.0 (http://smbus.org/specs/)。这些协议应通过用户软件实施。

■ 超时错误

SMBus 和 I2C 之间存在一些定时规范方面的差异。SMBus 定义了一个时钟低电平超时，tTIMEOUT 为 35ms。另外，SMBus 还指定 tLOW:SEXT 作为从器件的累积时钟低电平延长时间。SMBus 指定 tLOW:MEXT 作为主器件的累积时钟低电平延长时间。有关这些超时的详细信息，请参见 SMBus 规范版本 2.0(http://smbus.org/specs/)。

超时

该外设内置了硬件定时器，以便符合 SMBus 规范第 2.0 版本中定义的 3 个超时。

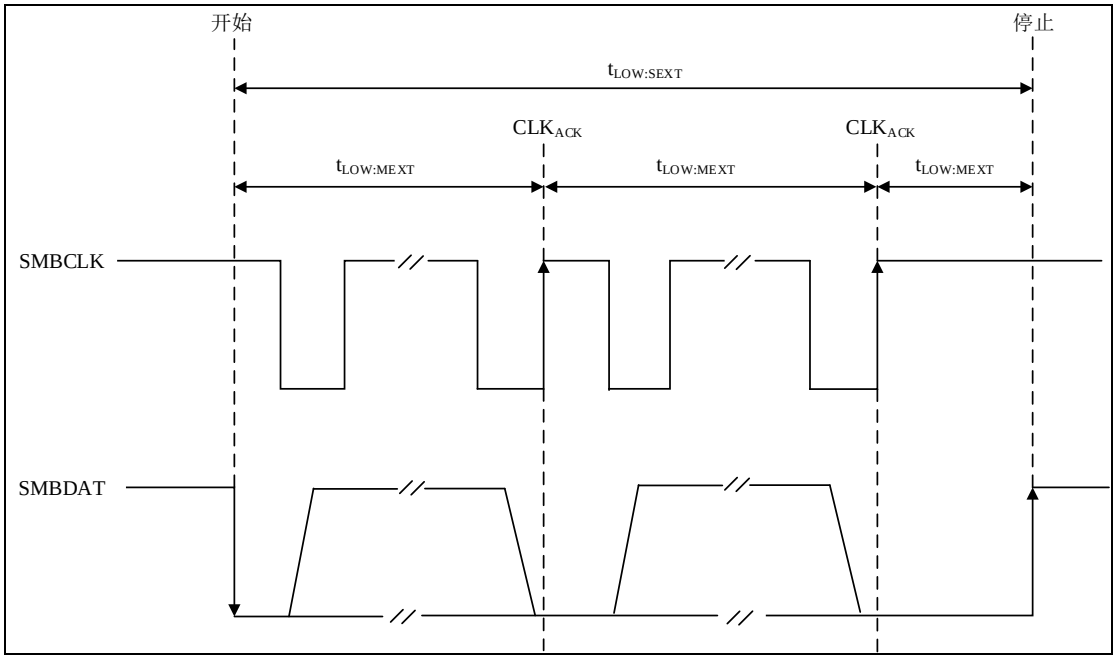
表 33-1.SMBus 超时规范

符号	参数	限值		单位
		最大值	最小值	
tTIMEOUT	检测时钟低电平超时	25	35	ms
tLOW:SEXT	累积时钟低电平延长时间（从器件）	-	25	ms
tLOW:MEXT	累积时钟低电平延长时间（主器件）	-	10	ms

注：

1）tLOW:SEXT 是一段累积时间，即给定从器件在一条消息的最初开始到停止期间时钟信号可延展的时间。其他从器件或主器件也可能延长时钟，进而导致时钟低电平总延长时间超过 tLOW:SEXT。因此，测量该参数时该器件应该是全速主器件寻址的唯一器件。

2）tLOW:MEXT 是一段累积时间，即主器件在消息的每个字节（定义为 START 到 ACK、ACK 到 ACK 或 ACK 到 STOP）内时钟信号可延展的时间。从器件或其它主器件也可能延长时钟，进而导致时钟低电平总时间超过 tLOW:MEXT（针对给定字节）。因此，测量该参数时该全速主器件只寻址一个从器件。



33.4.14. SMBus 初始化

仅当支持 SMBus 功能时，才涉及本节内容。除了 I2C 的初始化之外，还需要进行一些其他特定的初始化和软件功能的实现，才能实现 SMBus 通信。

■ 接收的命令和数据应答控制（从模式）

SMBus 接收器必须能够对接收到的每个命令或数据进行否定回答。可以使用 I2C_CR 的 TACK 进行应答控制。

■ 特定地址（从模式）

必须要时必须配置特定的 SMBus 地址。

在地址寄存器配置 SMBus 器件默认地址（0b1100001）

在地址寄存器配置 SMBus 主机地址（0b0001000）

在地址寄存器器配置报警响应地址（0b0001100）

■ 数据包错误校验

SMBus 的 PEC 可以通过硬件 CRC 模块和软件实现。

■ 超时检测

I2C 超时配置寄存器可以配置 tTIMEOUT 的时长，以 PCLK 的 2048 倍为计时长度。TIMOUTEN 置 1 使能定时器，当时钟低电平超过 TIMEOUTA 时，TIMEOUTAF 置 1。

I2C 超时配置寄存器可以配置 tLOW:SEXT 或 tLOW:MEXT 的时长，通过 EXT_MODE 来选择。以 PCLK 的 2048 倍为计时长度。EXTEN 使能定时器。当从器件的累积时钟低电平延长时间或主器件的累积时钟低电平延长时间超过 TIMEOUTB 时，TIMEOUTBF 置 1。

33.4.15. SMBus I2C_TIMEOUT 寄存器配置示例

将 tTIMEOUT 的最大持续时间配置为 25ms:

表 33-2 不同 PCLK 频率下的 TIMEOUTA 设置示例

FPCLK	TIMEOUTA[11:0]	TIMOUTEN 位	tTIMEOUT
30MHz	0x16E	1	367×2048×33.33ns=25ms
40MHz	0x1E8	1	489×2048×25ns=25ms
60MHz	0x2DA	1	733×2048×16.66ns=25ms

将 tLOW:SEXT 和 tLOW:MEXT 的最大持续时间配置为 8ms:

表 33-3 不同 PCLK 频率下的 TIMEOUTB 设置示例

FPCLK	TIMEOUTB[11:0]	EXTEN 位	tLOW:EXT
30MHz	0x75	1	118×2048×33.33ns=8ms
40MHz	0x9C	1	158×2048×25ns=8ms
60MHz	0xEA	1	235×2048×16.66ns=8ms

33.4.16. DMA 请求

DMA 请求(DMA_EN=1)仅用于数据传输。发送时 TXE=1 或接收时 RXNE=1，则产生 DMA 请求。结束当前字节传输之前，必须发出 DMA 请求。使用 DMA 模式时，从机发送模式下 TXE_SEL 必须设置为 1。不可以使能 TXE 和 RXNE 中断，不可以使用软件清除 TXE 和 RXNE 状态。当传输的数据量达到相应 DMA 通道编程设定的值时，DMA 控制器会生成一个传输完成 TC 中断（如果已使能）:

- 主发送器：在 TC 中断服务程序中，禁止 DMA 请求，然后在等到 MTF 标志后设置 STOP 位。
- 主接收器：在 TC 中断服务程序中，禁止 DMA 请求，设置 TACK 和 STOP 位。

■ 利用 DMA 发送

通过设置 I2C_CR 寄存器中的 DMA_EN 位可以激活 DMA 模式进行发送。只要 TXE 位被置位，数据将由 DMA 从预置的存储区装载进 I2C_DR 寄存器。为 I2C 分配一个 DMA 通道，须执行以下步骤（x 是通道号）：

1）设置 DMAC_CONFIG 的 EN 位，使能 DMA。

2）在 DMAC_Cx_SRC_ADDR 设置存储区地址。数据在每次 TXE 事件后从这个存储区传送至 I2C_DR。

3）在 DMAC_Cx_DSET_ADDR 设置 I2C_DR 寄存器地址。数据在每次 TXE 事件后从存储器传送至这个地址。

4）在 DMAC_Cx_CONFIG 寄存器中配置传输类型、源/目标外设 ID、传输完成中断。

5）在 DMAC_Cx_CTRL 寄存器中配置全局中断使能、源/目标地址递增使能与禁止、源/目标传输位宽、源/目标 burst size、传输长度。

6）通过设置 DMAC_Cx_CONFIG 寄存器 EN 位激活通道。

■ 利用 DMA 接收

通过设置 I2C_CR 寄存器中的 DMA_EN 位可以激活 DMA 模式进行接收。每次接收到数据时，RXNE 置位，将由 DMA 把 I2C_DR 寄存器的数据传送到设置的存储区。设置 DMA 通道进行 I2C 接收，须执行以下步骤（x 为通道号）：

1）设置 DMAC_CONFIG 的 EN 位，使能 DMA。

2）在 DMAC_Cx_SRC_ADDR 设置 I2C_DR 寄存器地址。数据在每次 RXNE 事件后从此地址传送到存储区。

3）在 DMAC_Cx_DSET_ADDR 设置存储区地址。数据在每次 RXNE 事件后 I2C_DR 寄存器传送到此存储区。

4）在 DMAC_Cx_CONFIG 寄存器中配置传输类型、源/目标外设 ID、传输完成中断。

5）在 DMAC_Cx_CTRL 寄存器中配置全局中断使能、源/目标地址递增使能与禁止、源/目标传输位宽、源/目标 burst size、传输长度。

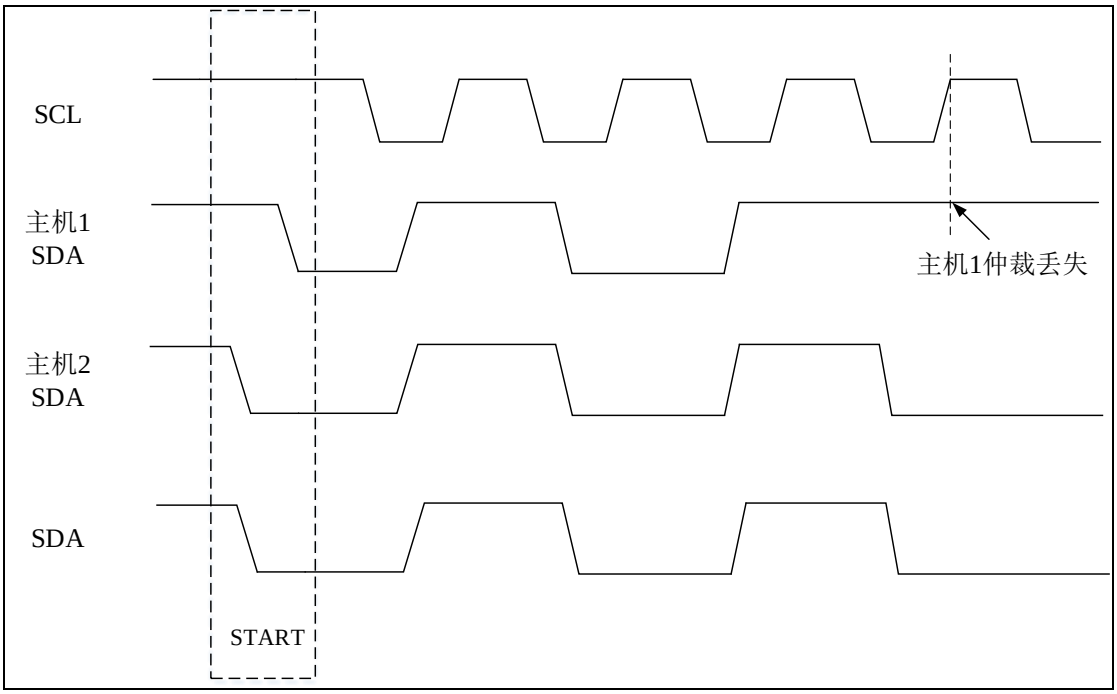
6）通过设置 DMAC_Cx_CONFIG 寄存器 EN 位激活通道。

33.4.17. I2C 的仲裁机制

仲裁逐位进行。在每一位的仲裁期间，当 SCL 为高时，每个主机都检查 SDA 电平是否和自己发送的相同。理论上，如果两个主机所传输的内容完全相同，那么它们能成功完成传输而不出现错误。如果一个主机发送高电平但检测到 SDA 电平为低，则认为自己仲裁丢失。丢失总裁的主机需要关闭自己的 SDA 输出驱动，而另一个主机则继续完成自己的传输。

主机仲裁丢失时，状态寄存器的 MARLO 位置 1。如果 MARLO_IN_EN 为 1，则会产生主仲裁丢失中断。

图 33-15 I2C 仲裁时序



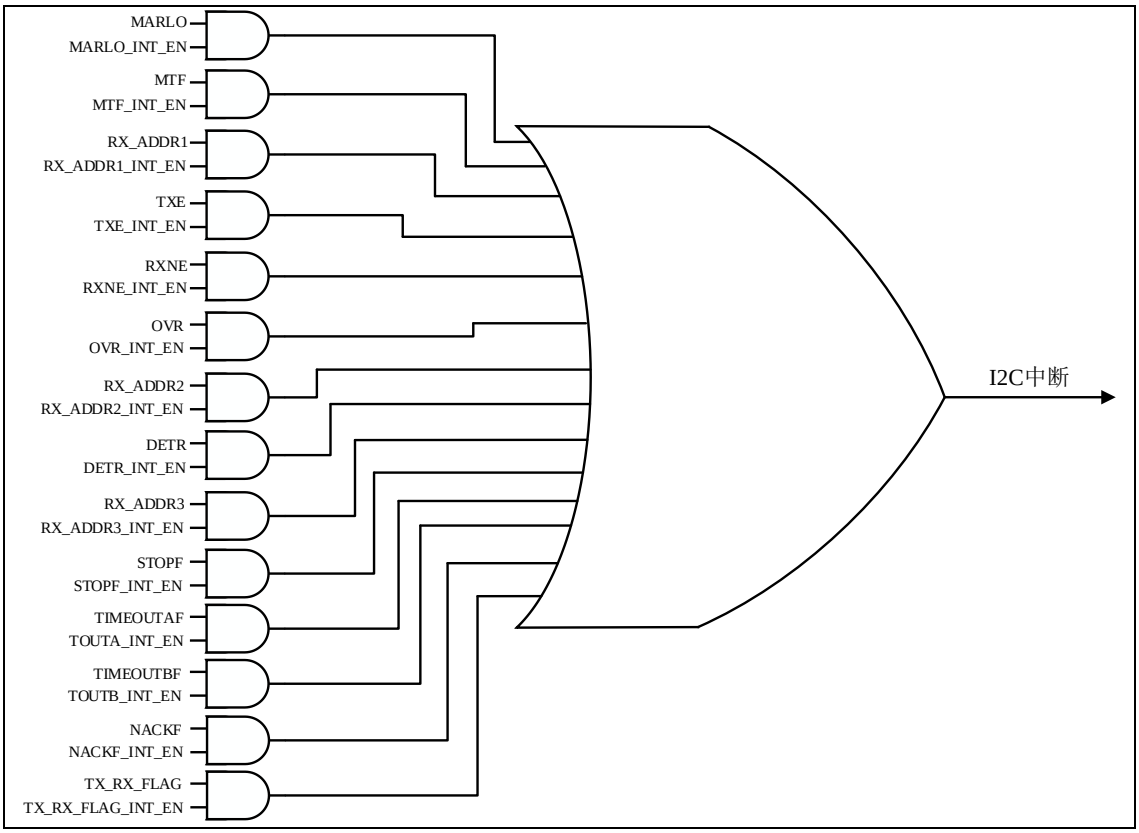
33.5. 中断及中断标志

下表列出了 I2C 中断请求列表。

表 33-4 I2C 中断请求表

中断事件	事件标志	使能控制位
主模式仲裁丢失	MARLO	MARLO_INT_EN
字节传输完成	MTF	MTF_INT_EN
从设备地址 1 匹配	RX_ADDR1	RX_ADDR1_INT_EN
数据寄存器数据被发送器取走	TXE	TXE_INT_EN
接收时数据寄存器非空	RXNE	RXNE_INT_EN
从机上溢/下溢	OVR	OVR_INT_EN
从设备地址 2 匹配	RX_ADDR2	RX_ADDR2_INT_EN
SDA 跳变状态检测	DETR	DETR_INT_EN
从设备地址 3 匹配	RX_ADDR3	RX_ADDR3_INT_EN
STOP 条件检测	STOPF	STOPF_INT_EN
TIMEOUTA 超时	TIMEOUTFAF	TOUTA_INT_EN
TIMEOUTB 超时	TIMEOUTFBF	TOUTB_INT_EN
接收到 NACKF	NACKF	NACKF_INT_EN
主机发送转接收	TX_RX_FLAG	TX_RX_FLAG_INT_EN

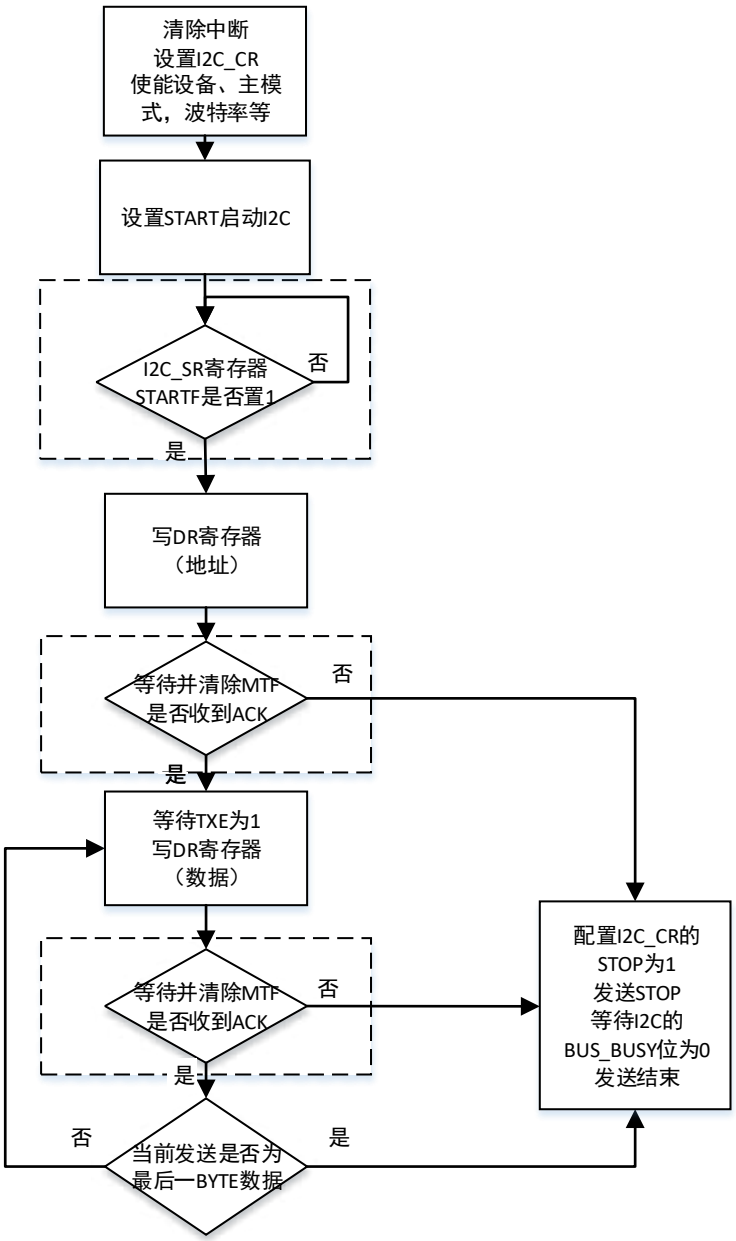
图 33-16 I2C 中断映射图图



33.6. 配置流程

33.6.1. 主发送器

图 33-17 主机发送流程图（虚线步骤可跳过，下同）



1）清除中断标志

2）写 I2C_CLK_DIV 寄存器的值确定 I2C 传输频率

3）写 I2C_CR 寄存器的 TX、MEN、MASTER 和 START 为 1，发起 START 条件

4）等待 I2C_SR.STARTF 标志为 1(也可不等待)。等待 STARTF 标志为 1 表明 START 条件已经发出，START 置 1 后，等不等待 STARTF 都可以填写 DR 发送地址。等待到 STARTF 再填 DR 会使填 DR 的动作滞后，START 条件之后的 SCL 低电平会长一点。如果确定 START 条件肯定可以正常发出，可以不等待 STARTF，否则等待 STARTF

5）把 I2C 要访问的 SLAVE 的 7 位地址写入 I2C_DR 寄存器中。(如若直接写数据或 START 之前数据已写入，STARTF 将被清 0)

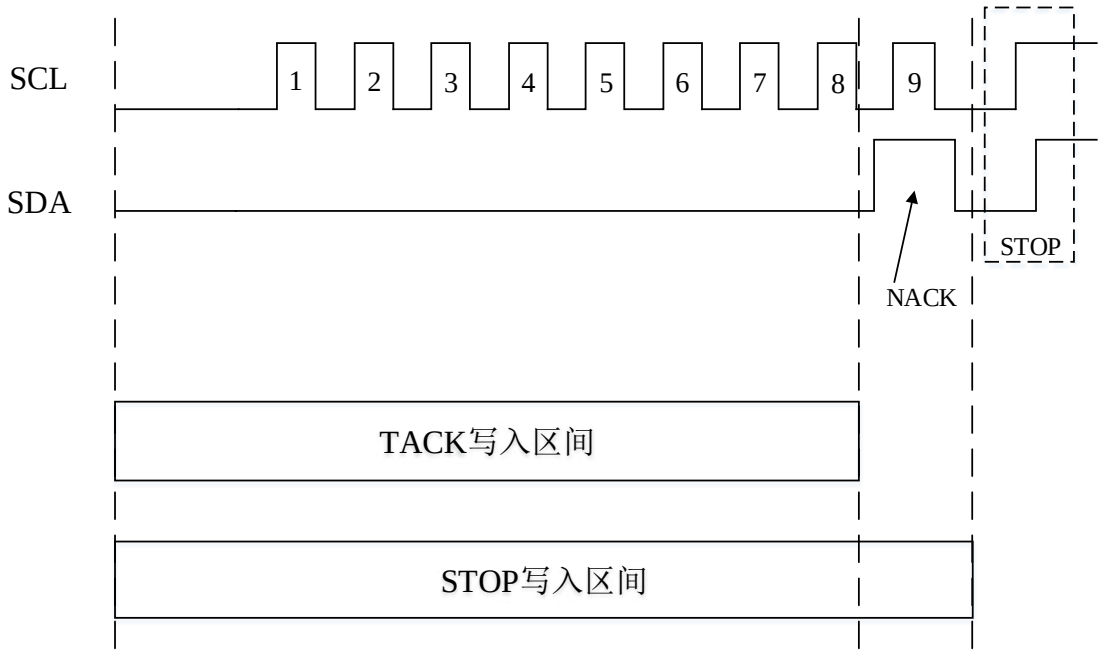
6）等待 I2C_SR.MTF 标志并清除 MTF，判断是否收到 ACK 后，表示从机正确。如果收到 NACK，软件写 I2C_CR.STOP 位为 1，结束发送并释放总线，软件等待 I2C_SR.BUS_BUSY 为 0 后退出

7）等待 I2C_SR.TXE 为 1，往 I2C_DR 寄存器写入要发送的字节，同时硬件会清除 TXE 位。若当前数据发送完成后，新的数据没有写入，主机不会产生新的发送时序

8）等待 I2C_SR.MTF 标志并清除 MTF，判断是否收到 ACK，如果收到 ACK 表示从机正确。如果收到 NACK，软件写 I2C_CR.STOP 位为 1，结束发送并释放总线，软件等待 I2C_SR.BUS_BUSY 为 0 后退出

9）重复 7 - 8 操作。在 8 操作中如确定回 ACK，可跳过

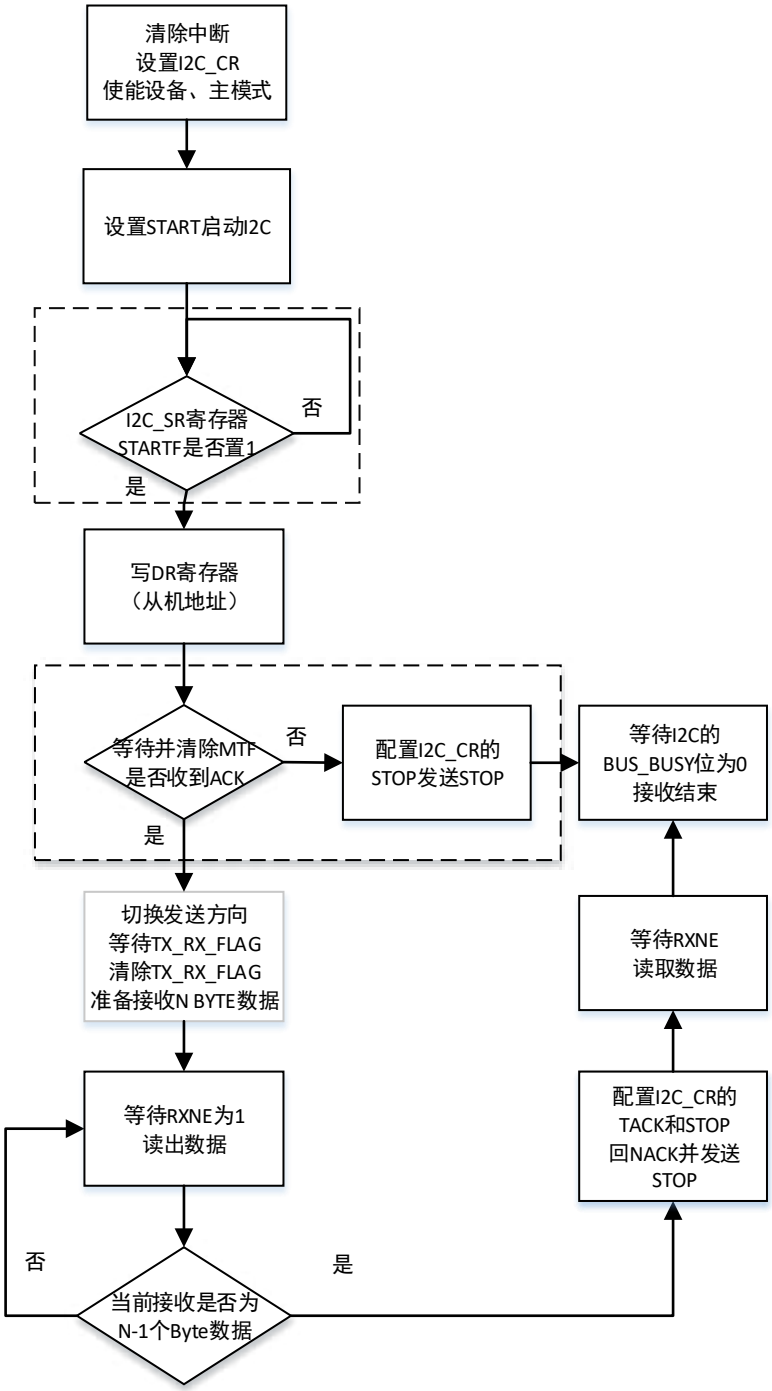
10）等到倒数第二个字节发送完成（MTF=1），向 I2C_DR 写完最后一个字节后。可以写 I2C_CR.STOP 为 1 结束发送。STOP 也可以在最后一个字节（MTF=1）发送完成后写入



I2C_CR.TX_AUTO_EN 使能后，I2C_CR.TX 位可以不配置，主默认作为发送器，地址字节最后 1bit，为 1 则为接收器，为 0 则为发送器。

33.6.2. 主接收器

图 33-18 主机接收流程图



1）清除中断标志

2）写 I2C_CLK_DIV 寄存器的值确定 I2C 传输频率

3）写 I2C_CR 寄存器的 MEN、MASTER 和 START 为 1，发起 START 条件

4）等待 I2C_SR.SRARTF 标志为 1(也可不等待)，把 I2C 要访问的 SLAVE 的 7 位地址和 1bit 的 1 写入 I2C_DR 寄存器中，表示作为接收器

5）等待 I2C_SR.MTF 标志并清除 MTF，判断是否收到 ACK，如果收到 ACK 表示从机正确。如果收到 NACK，软件写 I2C_CR 的 STOP 位，结束发送并释放总线，软件等待 I2C_SR.BUS_BUSY 位为 0 后退出。或者收到 NACK 后，软件写 I2C_CR.START 位为 1，重新发起一次新的操作。

6）切换发送为接收后，软件需要查看 I2C_SR.TX_RX_FLAG，若为 1，清除该状态。若该状态不清除。硬件会

停止接收时序

7）读到 I2C_SR.RXNE 为 1 时，处理器需读取 I2C_DR 寄存器中接收到的字节，同时硬件会清除 RXNE 位。等待下一个字节的接收结束。数据接收完成后若软件不读取数据，硬件将不会产生 SCL 时钟接收新的数据。

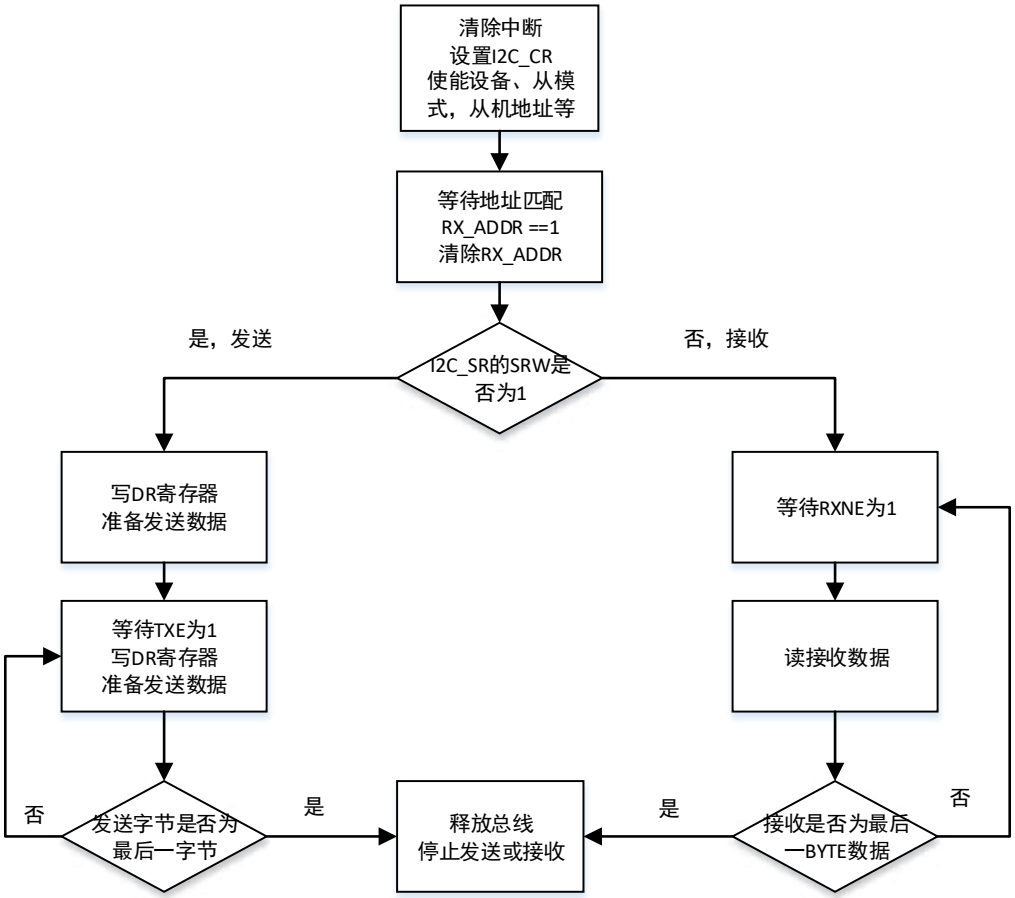
8）重复 7 操作

9）当倒数第二个字节接收完成，且发送完 ACK 信号后(即倒数第二个 RXNE 标志)，主机写 I2C_CR 的 TACK 和 STOP 为 1，表示下一个要接收的字节为最后一个字节

10）最后一个字节接收完成后硬件发出 NACK 信号并产生 STOP 条件，软件等待 I2C_SR.BUS_BUSY 为 0 后退出

33.6.3. 从发送器

图 33-19 从机发送接收流程图



1）清除中断标志

2）向 I2C_SLAVE_ADDR1 寄存器或 I2C_SLAVE_ADDR2_3 寄存器写入 7 位地址作为自己在从机状态下被寻址的地址

3）写 I2C_CR.MEN 为 1，使能 I2C 模块

4）等待 RX_ADDR1、RX_ADDR2（ADDR2_EN = 1）或 RX_ADDR3（ADDR3_EN = 1）标志是否有效。地址匹配无效则重复 4

5）地址匹配有效，判断 SRW 位是否为 1。为 0 表示从接收，为 1 表示从发送

6）写 I2C_CR.TX 为 1，切换到发送器，写第一个要发送的数据给 I2C_DR

7）等待 I2C_SR.TXE 为 1 时，向 I2C_DR 寄存器中写入即将要发送的数据，同时硬件会清除 TXE 位

8）重复 7，当收到主机发来的 STOP 后，I2C 模块释放总线。软件等待 I2C_SR 的 BUS_BUSY 为 0 后退出

9）注：I2C_CR.TX_AUTO_EN 使能后，I2C_CR 位的 TX 位可以不配置，从默认作为接收器，到地址字节最后 1bit，如果是 1 则为发送器，为 0 则为接收器

33.6.4. 从接收器

1）清除中断标志

2）向 I2C_SLAVE_ADDR1 寄存器或 I2C_SLAVE_ADDR2_3 寄存器写入 7 位地址作为自己在从机状态下被寻址的地址

3）写 I2C_CR 寄存器的 MEN 为 1，使能 I2C 模块

4）等待 RX_ADDR1、RX_ADDR2（ADDR2_EN = 1）或 RX_ADDR3（ADDR3_EN = 1）标志是否有效。地址匹配无效则重复 4

5）地址匹配有效，判断 I2C_SR.SRW 位是否为 1。为 0 表示从接收，为 1 表示从发送

6）等待 I2C_SR.RXNE 为 1 时，读取 I2C_DR 寄存器中接收到的字节，同时硬件会清除 RXNE 位

7）重复 6，当收到主机发来的 STOP 后，I2C 模块释放总线。软件等待 I2C_SR.BUS_BUSY 为 0 后退出

33.7. I2C 寄存器描述

33.7.1. 寄存器列表

I2C1 寄存器基地址：0x40005400

I2C2 寄存器基地址：0x40005800

I2C3 寄存器基地址：0x40005C00

I2C4 寄存器基地址：0x40006000

偏移	名称	复位值	描述
0x00	I2C_SLAVE_ADDR1	0x00000000	设备地址寄存器 1
0x04	I2C_CLK_DIV	0x00000000	时钟分频寄存器
0x08	I2C_CR	0x0C028000	控制寄存器
0x0C	I2C_SR	0x00000001	状态寄存器
0x10	I2C_DR	0x00000000	数据寄存器
0x14	I2C_SLAVE_ADDR2_3	0x00000000	设备地址寄存器 2_3
0x18	I2C_DET	0x00000000	SDA 跳变阈值寄存器
0x1C	I2C_FILTER	0x00000000	滤波寄存器
0x24	I2C_TIMEOUT	0x00000000	超时配置寄存器

33.7.2. 设备地址寄存器 1 (I2C_SLAVE_ADDR1: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	ADDR1	RW	0x0	地址的 7~1 位
0	RSV	-	-	保留

33.7.3. 时钟分频寄存器(I2C_CLK_DIV: 04h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:12	DLY_TIME	RW	0x0	从机拉时钟结束时延迟释放时钟线 SCL 的时间，以 APB 时钟计数。 注：当 I2C_CR 的 STRETCH_FIXEN 置 1 时，DLY_TIME 生效。
11:0	I2C_CLK_DIV	RW	0x0	I2C 时钟分频值，只在主模式时设置 Fscl = (Fpclk) / (4*(I2C_CLK_DIV +1)) 注： 1、Fpclk 为 APB 时钟频率，和系统时钟频率一致 2、I2C_CLK_DIV 的值必须大于 2

33.7.4. 控制寄存器(I2C_CR: 08h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27	MULTI_MA_EN	RW	0x1	多主机使能 0：禁止 1：使能 在总线上有多个主机时，需要将该位置 1。
26	HIGH_WAIT_EN	RW	0x1	主机时钟高等待使能 0：禁止 1：使能 该位置 1 时，如果主机内部时钟为高高电平，当 IO 上检测到的时钟为低电平，还未升高时，内部需要等待 IO 上的时钟变成高电平。
25	STRETCH_FIX_EN	RW	0x0	从机拉时钟修正使能 0：修正禁止 1：修正使能 该位置 1 时，从机拉时钟状态结束时，时钟线 SCL 延迟释放，延迟释放时间由 I2C_CLK_DIV 的 DLY_TIME 控制。
24	SU_HD_FIX_EN	RW	0x0	set-up 和 hold 时序修正使能（起始条件 START 和结束条件 STOP） 0：修正禁止 1：修正使能
23	MARLO_SEL	RW	0x0	主机仲裁丢失后行为选择 0：不自动切换成从机，兼容以前产品 1：自动切换成从机 注：主机仲裁丢失，自动切换成从机后，将无法在本次传输中对自身地址进行应答。但是能够在赢得仲裁的主机发起新的寻址时，对自身地址进行应答。
22	TX_RX_FLAG_INT_EN	RW	0x0	TX_RX_FLAG 中断使能 0：TX_RX_FLAG=1 中断不使能 1：TX_RX_FLAG=1 中断使能

21	NACKF_INT_EN	RW	0x0	接收到否定应答标志中断使能 0：NACKF=1 中断不使能 1：NACKF=1 中断使能
20	STOPF_INT_EN	RW	0x0	STOPF 中断使能 0：STOPF=1 中断不使能 1：STOPF=1 中断使能
19	RX_ADDR3_INT_EN	RW	0x0	ADDR3 地址匹配中断使能 0：RX_ADDR3=1 中断不使能 1：RX_ADDR3=1 中断使能
18	DMA_EN	RW	0x0	DMA 功能使能 0：不使能 1：使能，使用 TXE 和 RXNE 分别产生发送和接收请求。此时不能打开 TXE 和 RXNE 中断，不能用软件清除这两个状态
17	TXE_SEL	RW	0x1	从模式发送在从机地址匹配后，TXE 是否变高 0：不变高 1：变高，DMA 模式一定要选择
16	MARLO_INT_EN	RW	0x0	主仲裁丢失中断使能 0：MARLO =1 中断不使能 1：MARLO =1 中断使能
15	TX_AUTO_EN	RW	0x1	SDA 数据线方向自动切换。 0：不使能自动切换功能 1：使能自动切换功能 此位设置为 1，根据地址字节的 RW 位自动切换 SDA 数据线的传输方向
14	RSV	-	-	保留
13	DETR_INT_EN	RW	0x0	SDA 跳变检测中断使能 0：DETR=1 中断不使能 1：DETR=1 中断使能
12	RX_ADDR2_INT_EN	RW	0x0	ADDR2 地址匹配中断使能 0：RX_ADDR2=1 中断不使能 1：RX_ADDR2=1 中断使能
11	OVR_INT_EN	RW	0x0	从机上溢/下溢中断使能 0：OVR=1 中断不使能 1：OVR=1 中断使能
10	RXNE_INT_EN	RW	0x0	接收数据中断使能 0：RXNE=1 中断不使能 1：RXNE=1 中断使能
9	TXE_INT_EN	RW	0x0	发送数据中断使能 0：TXE=1 中断不使能 1：TXE=1 中断使能
8	RX_ADDR1_INT_EN	RW	0x0	ADDR1 地址匹配中断使能 0：RX_ADDR1=1 中断不使能 1：RX_ADDR1=1 中断使能

7	MTF_INT_EN	RW	0x0	字节传输完成中断使能 0：MTF=1 中断不使能 1：MTF=1 中断使能
6	TACK	RW	0x0	传输应答位 0：接收一字节后，在应答周期产生 ACK 1：接收一字节后，在应答周期产生 NACK 注:TACK 必须在应答周期前写入
5	STOP	RW	0x0	结束条件产生位 0：发送完当前字节不产生结束条件 1：主设备在发送完当前字节后，将产生结束条件。产生结束条件后，硬件自动清 0
4	START	RW	0x0	起始条件产生位 0：主模式下不产生起始条件 1：主模式下产生起始条件 注：空闲时刻和 NACK/ACK 应答后才可产生起始条件，起始条件产生后，硬件自动清 0，I2C_SR 的 STARTF 位置 1
3	TX	RW	0x0	发送接收选择位 0：设备作为接收器 1：设备作为发送器 当作为从设备时，处理器应该查询 I2C_SR 的 SRW 位，判断是作为发送器还是接收器，然后设置与之匹配的 TX 位 TX_AUTO_EN 位使能后 TX 位设置无效
2	MASTER	RW	0x0	主从设备选择位 0：从模式 1：主模式
1	NOSTRETCH	RW	0x0	从模式禁止时钟延长 0：使能时钟延长 1：禁止时钟延长
0	MEN	RW	0x0	设备使能位 0：设备不使能 1：设备使能

注：TX_AUTO_EN 自动使能后 I2C 模块 SDA 方向由硬件自动切换，无需软件配置。主机模式下默认为发送器，根据第一个发送的地址的 RW 位自动切换发送器或接收器，从机模式下默认为接收器，根据第一个接收的地址的 RW 位自动切换发送器或接收器。

33.7.5. 状态寄存器(I2C_SR: 0Ch)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留位
17	NACKF	RO	0x0	接收到否定应答（NACKF）标志 传输完字节后接收到 NACK 时，该标志位由硬件置 1，写 1 清 0 注：主机发送模式下收到 NACK 停止发送，等待软件处理事件。

16	TIMEOUTBF	RO	0x0	TIMEOUTB 超时标志，表示 SMBus EXT 超时 写 1 清 0
15	TIMEOUTAF	RO	0x0	TIMEOUTA 超时标志，表示 SMBus SCL Timeout 超时 写 1 清 0
14	RX_ADDR3	RO	0x0	从设备地址 3 匹配状态位 0：设备地址 3 和接收到的地址不相等 1：设备地址 3 和接收到的地址相等 写 1 清 0 注：当地址匹配时，SRW 位表示了地址字节的 RW 位
13	DETR	RO	0x0	SDA 跳变检测状态位 0：当 SCL 为低时 SDA 上升沿跳变次数小于 I2C_DET 所设置的检测阈值 1：当 SCL 为低时 SDA 上升沿跳变次数大于等于 I2C_DET 所设置的检测阈值 写 1 清 0
12	RX_ADDR2	RO	0x0	从设备地址 2 匹配状态位 0：设备地址 2 和接收到的地址不相等 1：设备地址 2 和接收到的地址相等 写 1 清 0 注：当地址匹配时，SRW 位表示了地址字节的 RW 位
11	OVR	RO	0x0	从机上溢/下溢状态位 0：未发生上溢/下溢 1：发生上溢/下溢 从机收到的字节尚未读取，并收到新的字节时 OVR 置 1 从机发送过程中从机需要发送一个新的字节但尚未向 DR 寄存器写入数据时 OVR 置 1
10	RXNE	RO	0x0	接收数据时数据寄存器状态位 0：接收时数据寄存器空 1：接收时数据寄存器非空 硬件置位，通过读数据寄存器 I2C_DR 可以清除该位，写 1 清 0 注：主机接收模式下收到数据后必须读取 DR 寄存器数据否则主机不会产生新的读时序
9	TXE	RO	0x0	发送数据时数据寄存器状态位 0：I2C_DR 数据未被发送器取走 1：I2C_DR 数据被发送器取走 硬件置位，写 1 清 0，写 DR 清 0 注：主机发送模式下，写 DR 寄存器后主机才会发送数据
8	RX_ADDR1	RO	0x0	从设备地址 1 匹配状态位 0：设备地址 1 和接收到的地址不相等 1：设备地址 1 和接收到的地址相等 写 1 清 0 注：当地址匹配时，SRW 位表示了地址字节的 RW 位

7	MTF	RO	0x0	字节传输完成状态位 0：字节传输未完成 1：字节传输完成 当一个字节数据（包括地址）正在传输时，该位为 0；在一个字节传输完后，在第 9 个 SCL 时钟下降沿（应答周期）MTF 被置为 1。写 1 清 0
6	MARLO	RO	0x0	主模式仲裁丢失 0：没有检测到仲裁丢失 1：检测到仲裁丢失 主发送 SDA 为高，但接收到 SDA 为低时认为丢失仲裁，需要软件处理
5	TX_RX_FLAG	RO	0x0	主机发送转接收状态位 0：主机未由发送状态转为接收状态 1：主机由发送状态转为接收状态 硬件置 1，写 1 清 0 注：当 TX_RX_FALG 状态置 1 后，硬件停止 I2C 时钟，需软件清该状态。从模式下该状态无效
4	BUS_BUSY	RO	0x0	总线忙碌状态位 0：总线上无数据通信（检测到总线上的结束条件，此位清 0） 1：总线上正在进行数据通信（检测到总线上的起始条件，此位置 1）
3	SRW	RO	0x0	从机读写状态指示位 0：作为从设备接收器 1：作为从设备发送器 当地址匹配后，SRW 指示地址字节中的 RW 位，该位仅在如下条件有效：一个完整的传输已经发生，没有其他传输被初始化；并且 I2C 被配置为从模式，且从地址匹配。当接收到停止条件或一个新的起始条件，该位自动清除
2	STOPF	RO	0x0	STOP 条件位检测位，主从都会产生 0：未检测到停止位 1：检测到停止位 只能写 1 清 0
1	STARTF	RO	0x0	主机起始条件发送状态位 0：起始条件未发送 1：起始条件已发送 写 1 清 0 注：I2C_DR 寄存器有数据将自动启动发送时序并清除该位，主机模式下，SDA 方向不由硬件切换并处于接收状态时 STARTF 也将被清除
0	RACK	RO	0x1	NACK 检测位 0：最近的应答周期检测到总线上的 ACK 1：最近的应答周期检测到总线上的 NACK 只有 START 条件将清除 RACK 位

33.7.6. 数据寄存器(I2C_DR: 10h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留

7:0	I2CDR	RW	0x0	I2C 数据寄存器

33.7.7. 设备地址寄存器 2_3 (I2C_SLAVE_ADDR2_3: 14h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:9	ADDR3	RW	0x0	地址的 7~1 位
8	ADDR3_EN	RW	0	0：SLAVE_ADDR3 地址匹配不使能 1：SLAVE_ADDR3 地址匹配使能
7:1	ADDR2	RW	0x0	地址的 7~1 位
0	ADDR2_EN	RW	0	0：SLAVE_ADDR2 地址匹配不使能 1：SLAVE_ADDR2 地址匹配使能

33.7.8. SDA 跳变阈值寄存器（I2C_DET: 18h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	DETCNT	RW	0x0	当 SCL 线为低时，SDA 线上跳变次数阈值。 0：没有检测功能 其他值：SDA 的上升沿跳变次数阈值

33.7.9. 滤波寄存器(I2C_FILTER: 1Ch)

位域	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12:8	SDA_IN_DELAY	RW	0x0	SDA 输入延时设置位，SCL 滤波功能使能后，SDA_IN_DELAY 与 SCL_FITER 的值设置相同，SDA 信号与 SCL 信号在经过滤波后，将保持输入时的相位。
7:5	RSV	-	-	保留
4:0	SCL_FILTER	RW	0x0	滤波 0 算法 SCL 滤波值设置位。 滤的毛刺的最大宽度为 Tfclk*8* SCL_FILTER。Tfclk 为系统时钟周期

33.7.10. 超时配置寄存器(I2C_TIMEOUT: 24h)

位域	名称	属性	复位值	描述
31	EXTEN	RW	0x0	TimeoutB 定时使能，超出后产生 TIMEOUTB 标志
30	TOUTB_INT_EN	RW	0x0	TIMEOUTBF 标志中断使能
29	EXT_MODE	RW	0x0	EXT 计数（TimeoutB）模式选择 0：计数 SEXT 模式 1：计数 MEXT 模式
28	RSV	-	-	保留

27:16	TIMEOUTB	RW	0x0	SMBus EXT 时间设置，以 PCLK 的 2048 倍为计时长度
15	TIMOUTEN	RW	0x0	TimeoutA 定时使能，超出后产生 TIMEOUTA 标志
14	TOUTA_INT_EN	RW	0x0	TIMEOUTAF 标志中断使能
13:12	RSV	-	-	保留
11:0	TIMEOUTA	RW	0x0	SMBus Timeout 时间设置，以 PCLK 的 2048 倍为计时长度

34. 片上音频接口（I2S）

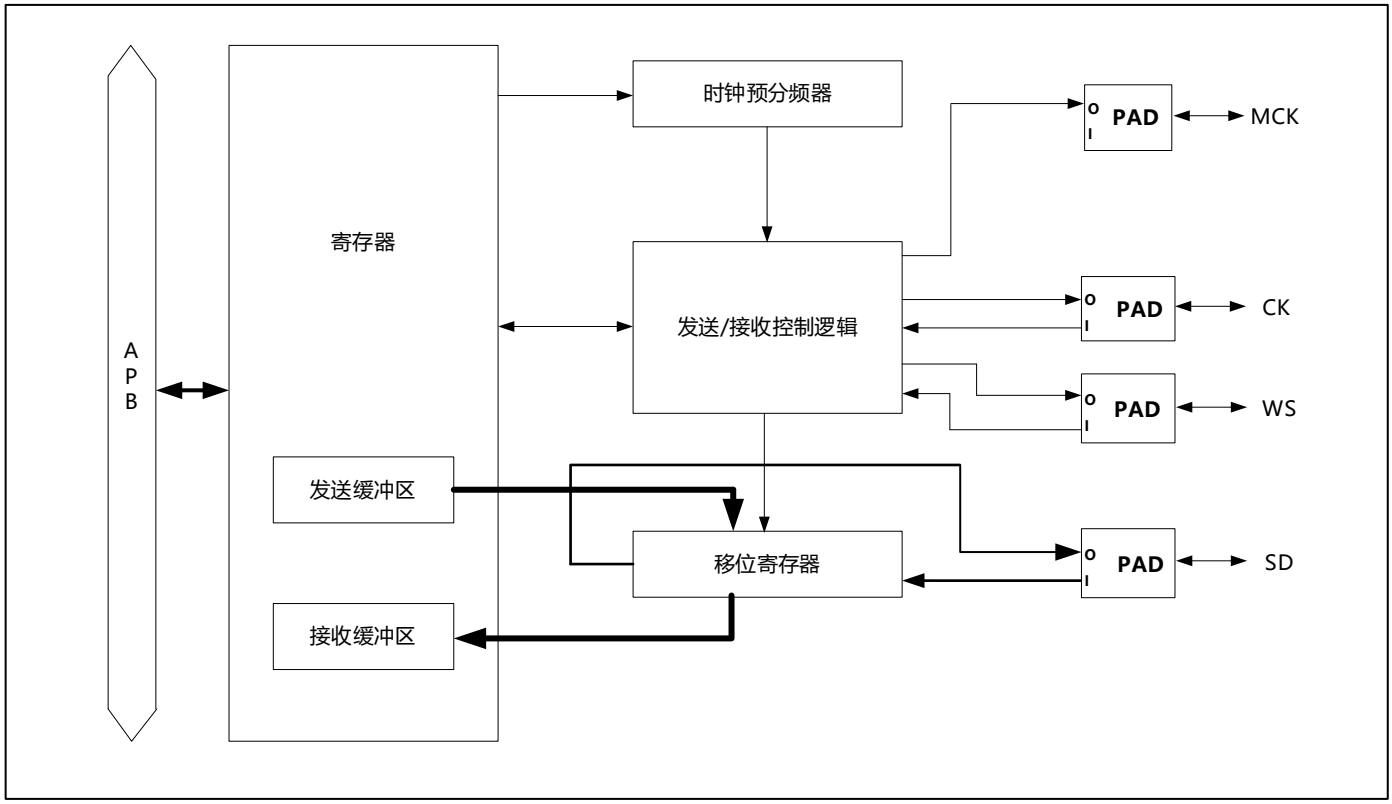
34.1. 概述

片上音频接口（Inter-IC Sound，缩写为 I2S）模块可以通过 I2S 音频协议与外部设备进行通信。I2S 接口支持四种音频标准，分别是 I2S 飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准。它可以在四种模式下运行，包括主机发送模式，主机接收模式，从机发送模式和从机接收模式。

34.2. 主要特性

- 具有发送和接收功能的主从操作
- 单工通信
- 支持四种 I2S 音频标准：飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准
- 数据长度可以为 16 位，24 位和 32 位
- 通道长度为 16 位或 32 位
- 32 位缓冲区用于发送和接收
- 9 位可编程预分频器，可实现精确的音频采样频率
- 可编程空闲状态时钟极性
- 可以输出主时钟（MCK）
- 发送和接收支持 DMA 功能

34.3. 结构框图



I2S 采用 APB 协议接口。发送缓冲区、接收缓冲区均为 32 位的数据寄存器，内部根据 I2S_CR.DTLEN[1:0]和 I2S_CR.CHLEN 控制位自动调节有效数据。在主模式下，移位寄存器通过发送/接收控制逻辑进行数据收发，收

发频率通过 I2S_PR.DIV 和 I2S_PR.OF 控制并生成主时钟（MCK）和串行时钟（CK）供从机使用；在从模式下，根据主机提供的 CK 控制移位寄存器发送/接收逻辑进行数据收发。MCU 通过读发送/接收缓冲区空满状态进行数据读写。

I2S 信号管脚：

- SD：串行数据输入、输出，用来发送和接收数据；
- WS：字选择，主模式下作为数据控制信号输出，从模式下作为输入；
- CK：串行时钟，主模式下作为时钟信号输出，从模式下作为输入。
- MCK：主时钟，在 I2S 配置为主模式，寄存器 I2S_PR.MCKOE 位为 1 时，作为输出主时钟信号使用。

34.4. 功能描述

34.4.1. 时钟

I2S 比特率用来确定 I2S 数据线上的数据流和 I2S 时钟信号频率。

I2S 接口时钟（CK）是通过 I2S_PR 寄存器的 DIV 位，OF 位和 MCKOE 位以及 I2S_CR 寄存器的 CHLEN 位来配置的。I2S 的时钟源是外设时钟（PCLK）。I2S 比特率可以通过表 1-1 I2S 比特率计算公式所示的公式计算。

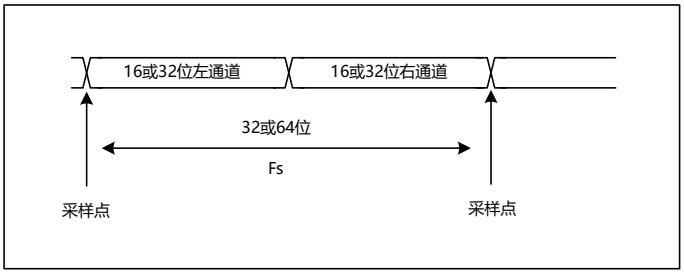
表 34-1 I2S 比特率计算公式

MCKOE	CHLEN	公式
0	0	PCLK / (DIV * 2 + OF)
0	1	PCLK / (DIV * 2 + OF)
1	0	PCLK / (8 * (DIV * 2 + OF))
1	1	PCLK / (4 * (DIV * 2 + OF))

音频采样率(Fs)和 I2S 比特率的关系由如下公式定义：

Fs = I2S 比特率 / (通道长度 * 通道数)

下图为音频采样频率 Fs 定义：



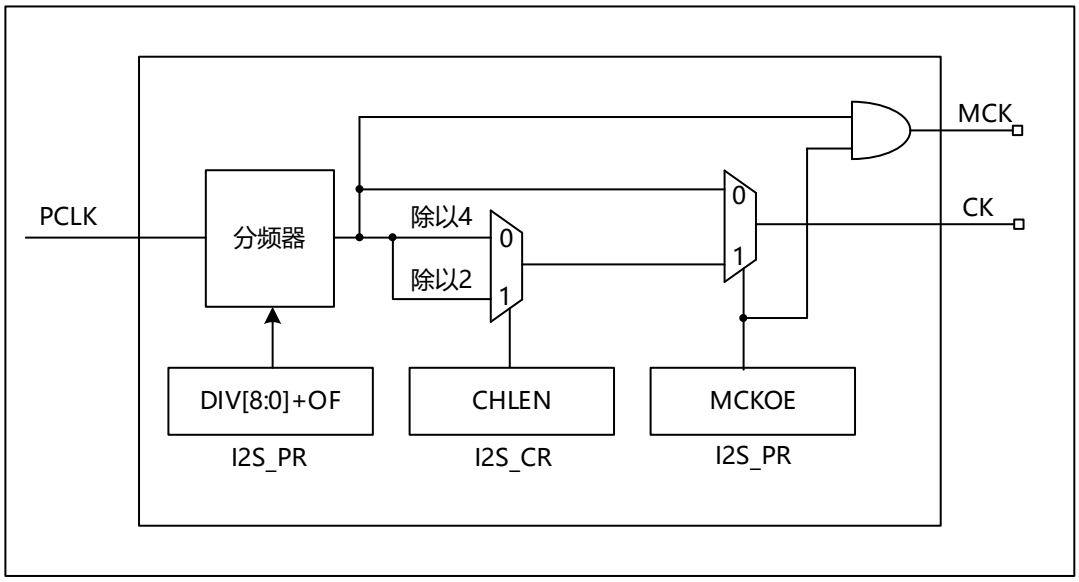
为了得到期望的音频采样率，时钟生成器需要按下表音频采样频率计算公式所列的公式进行配置。

表 34-2 Fs 采样频率计算公式

MCKOE	CHLEN	公式
0	0	PCLK / (32 * (DIV * 2 + OF))
0	1	PCLK / (64 * (DIV * 2 + OF))
1	0	PCLK / (256 * (DIV * 2 + OF))

1	1	PCLK / (256 * (DIV * 2 + OF))

下图为 I2S 时钟发生器结构框图，其中 DIV/OF/MCKOE 仅用于主模式，从模式下无效。



34.4.2. 音频标准

I2S 音频标准是通过设置 I2S_CR 寄存器中的 I2SSTD 位来选择的，可以选择四种音频标准：I2S 飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准。除 PCM 之外的所有标准都是两个通道(左通道和右通道)的音频数据分时复用 I2S 接口的，并通过 I2S_WS 信号来区分当前数据属于哪个通道。对于 PCM 标准，I2S_WS 信号表示帧同步信息。

数据长度（原始数据位数）和通道长度（通道实际发送的数据位数）可以通过 I2SCR 寄存器中的 DTLEN 位和 CHLEN 位来设置。

由于通道长度必须大于或等于数据长度，所以有四种数据包类型可供选择。它们分别是：

- 16 位数据打包成 16 位数据帧格式
- 16 位数据打包成 32 位数据帧格式(16 位 LSB 数据补零)
- 24 位数据打包成 32 位数据帧格式(8 位 LSB 数据补零)
- 32 位数据打包成 32 位数据帧格式。用于发送和接收的数据缓冲区都是 32 位宽度

对于所有标准和数据包类型来说，数据的最高有效位总是最先被发送的(MSB first)。对于所有基于两通道分时复用的标准来说，总是先发送左通道，然后是右通道。

34.4.3. I2S 飞利浦标准

对于 I2S 飞利浦标准，I2S_CR.CKPL 可配置空闲 CK 电平，WS 信号空闲默认为高电平，I2S_WS 和 I2S_SD 在 I2S_CK 的下降沿变化，有效数据从第二个时钟下降沿开始，主或从可在上升沿处采集有效数据。具体各类配置情况的时序图如下所示。

图 34-1 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=0)

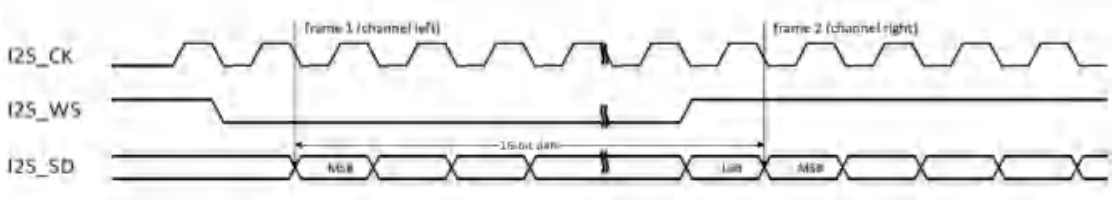


图 34-2 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=1)

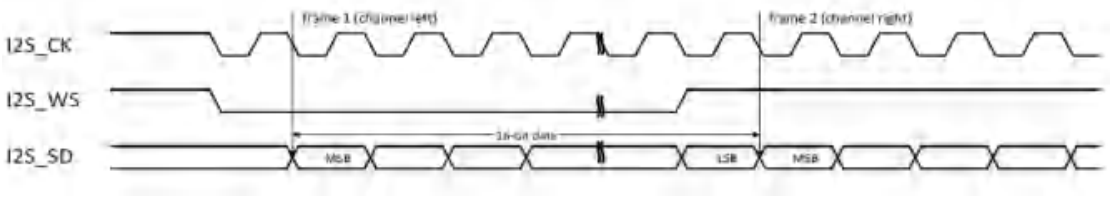


图 34-3 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=0)

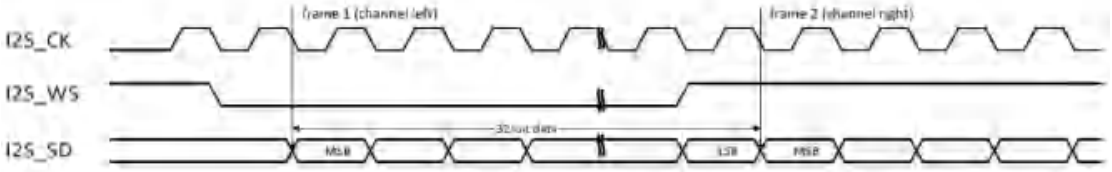
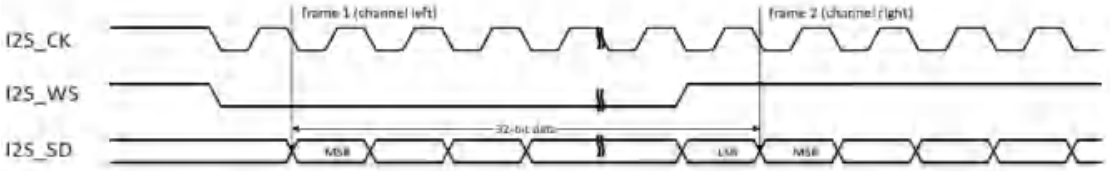


图 34-4 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=1)



当 24 位数据打包成 32 位数据帧的帧格式时，为了将该 24 位数据扩展成 32 位数据，剩下的 8 位被硬件强制填充为 0x00。

图 34-5 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

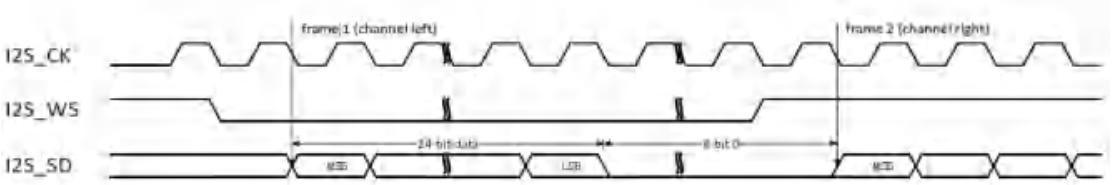
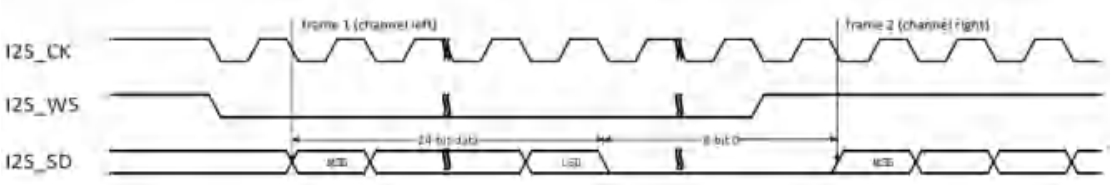


图 34-6 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=1)



当 16 位数据打包成 32 位数据帧时，为了将该 16 位数据扩展成 32 位数据，剩下的 16 位被硬件强制填充为 0x0000。

图 34-7 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

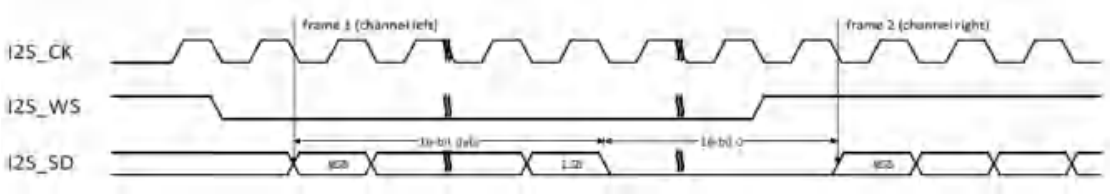
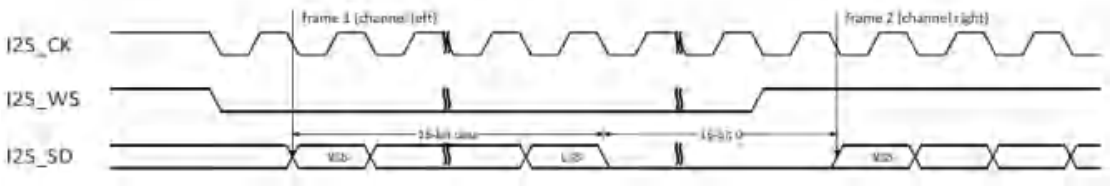


图 34-8 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



34.4.4. MSB 对齐标准

对于 MSB 对齐标准，I2S_CR.CKPL 可配置空闲 CK 电平，WS 信号空闲默认为低电平，I2S_WS 和 I2S_SD 在 I2S_CK 的下降沿变化。SPI_DATA 寄存器的处理方式与 I2S 飞利浦标准完全相同，部分处理方式参看 I2S 飞利浦标准。各个配置情况的时序图如下所示。

图 34-9 MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=0)

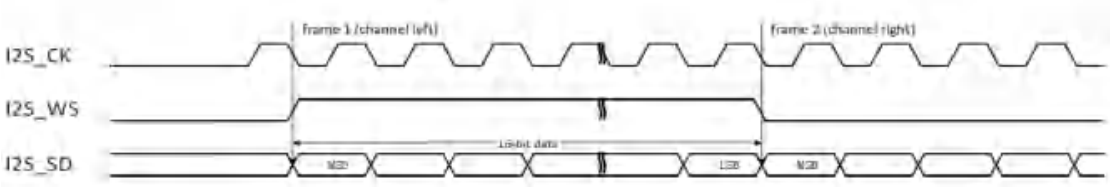


图 34-10 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=1)

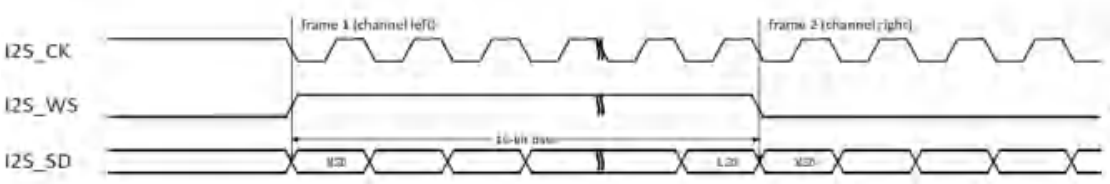


图 34-11 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=0)

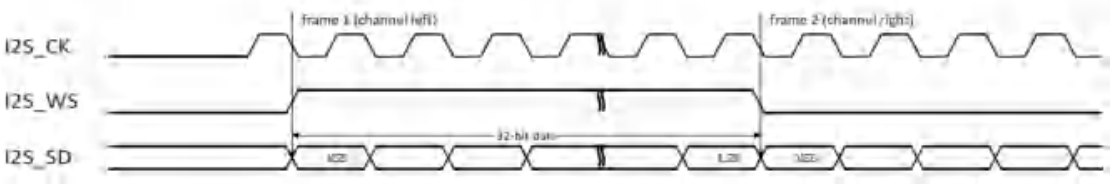


图 34-12 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=1)

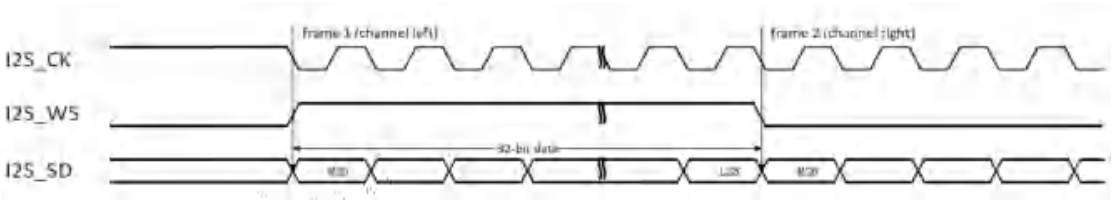


图 34-13 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

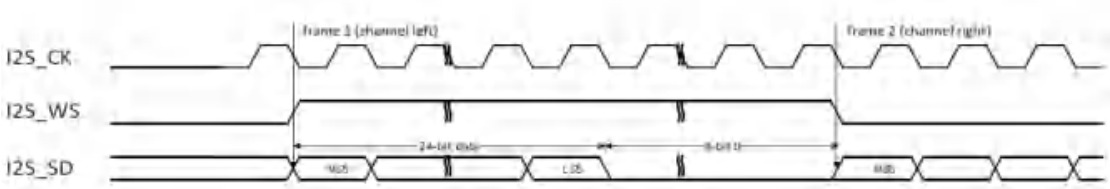


图 34-14 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)

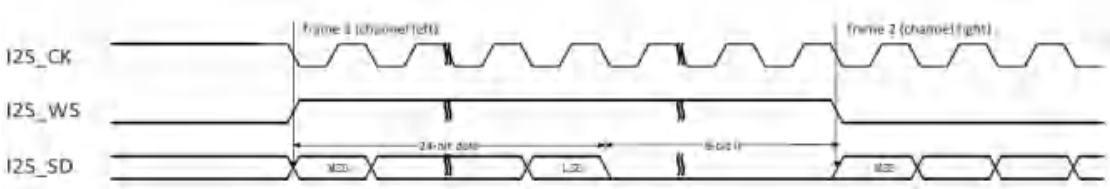


图 34-15 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

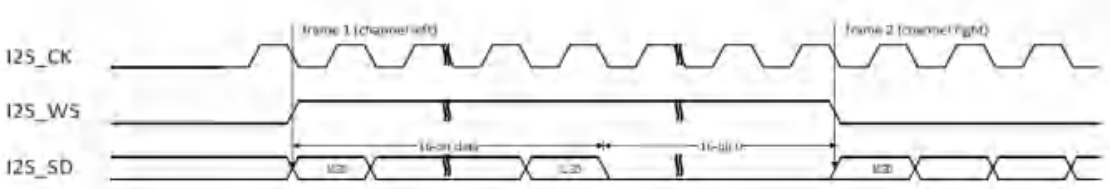
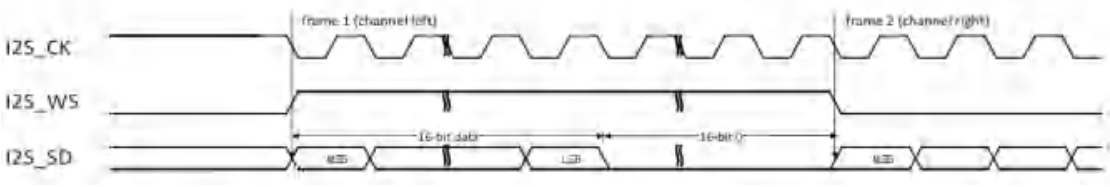


图 34-16 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



34.4.5. LSB 对齐标准

对于 LSB 对齐标准，I2S_CR.CKPL 可配置空闲 CK 电平，WS 信号空闲默认为低电平，I2S_WS 和 I2S_SD 在 I2S_CK 的下降沿变化。在通道长度与数据长度相同的情况下，LSB 对齐标准和 MSB 对齐标准是完全相同的。对于通道长度大于数据长度的情况，LSB 对齐标准的有效数据与最低位对齐，而 MSB 对齐标准的有效数据与最高位对齐。通道长度大于数据长度的各种配置情况时序图如下所示。当 24 位数据打包成 32 位数据帧时，为了将该 24 位数据扩展成 32 位数据，剩下的 8 位被硬件强制填充为 0x00。

图 34-17 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

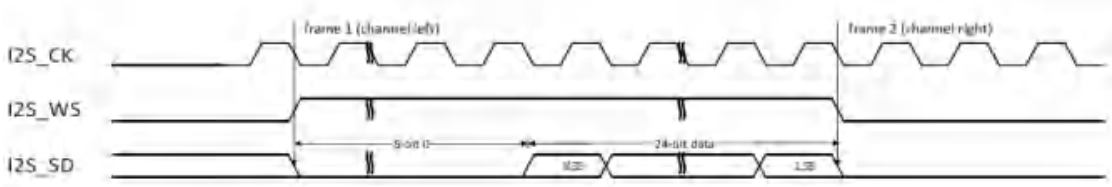
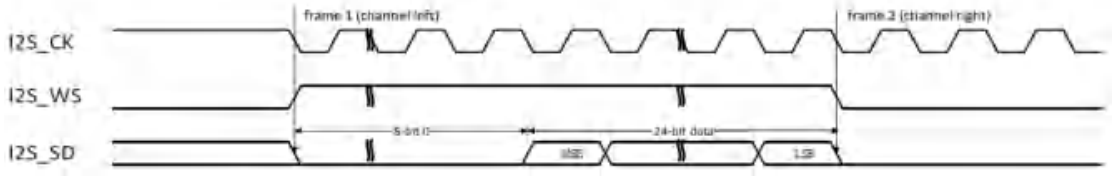


图 34-18 LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)



当 16 位数据打包成 32 位数据帧时，为了将该 16 位数据扩展成 32 位数据，剩下的 16 位被硬件强制填充为 0x0000。

图 34-19 LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

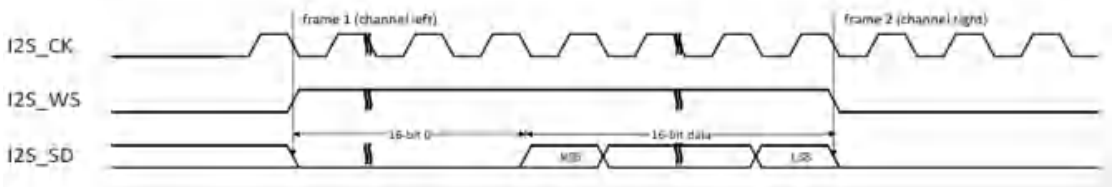
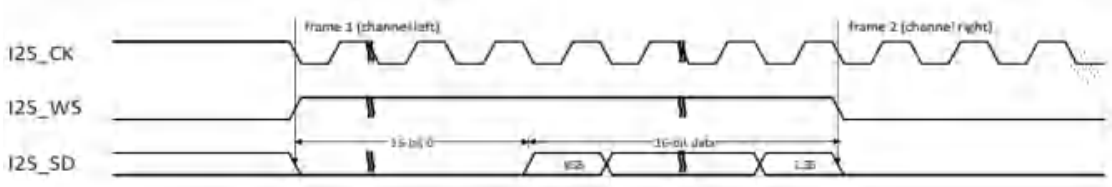


图 34-20 LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



34.4.6. PCM 对齐标准

对于 PCM 标准，I2S_CR.CKPL 可配置空闲 CK 电平，WS 信号空闲默认为低电平，I2S_WS 和 I2S_SD 在 I2S_CK 的上升沿变化，I2S_WS 信号表示帧同步信息。可以通过 SPI_I2SCTL 寄存器的 PCMSMOD 位来选择短帧同步模式和长帧同步模式，短帧指 I2S_WS 只保持一位数据长度的时间，而长帧保持 13 位数据长度的时间，帧头与数据的关系请参看时序图。SPI_DATA 寄存器的处理方式与 I2S 飞利浦标准完全相同。短帧同步模式的各种配置情况时序图如下所示。

图 34-21 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)

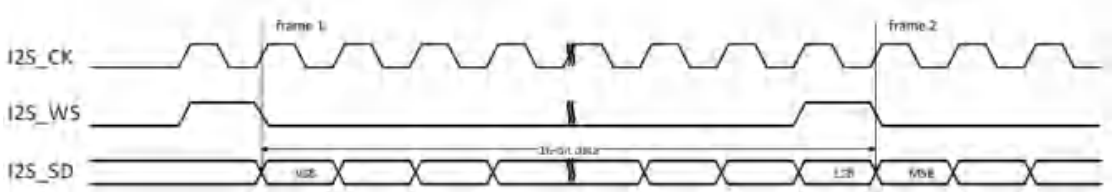


图 34-22 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)

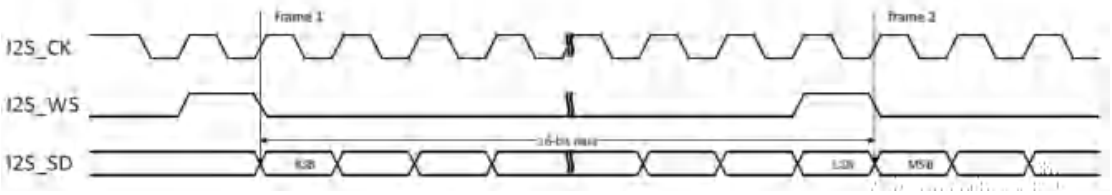


图 34-23 PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)

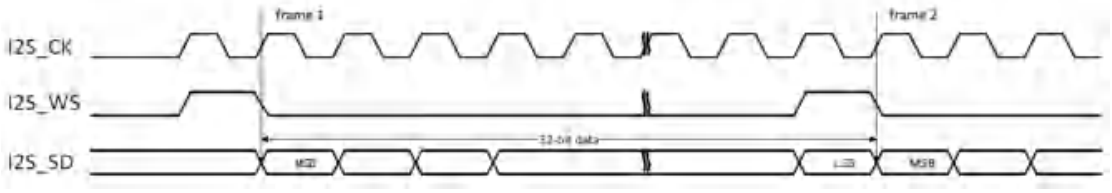


图 34-24 PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)

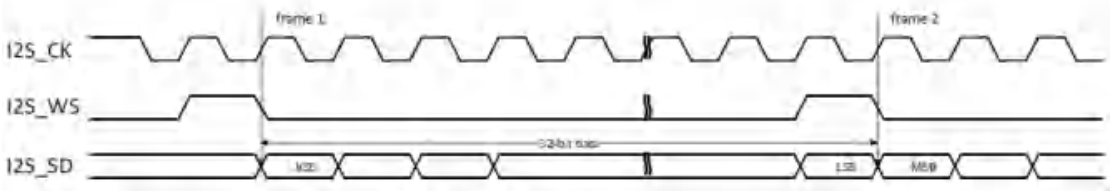


图 34-25 PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

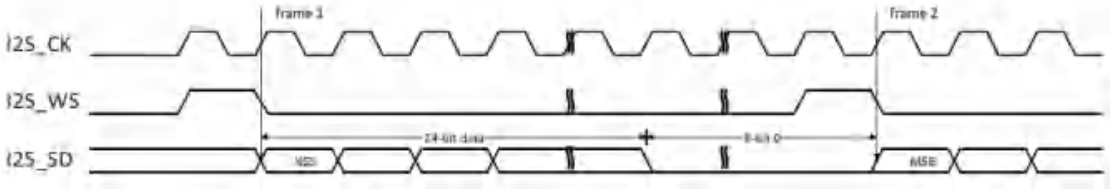


图 34-26 PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

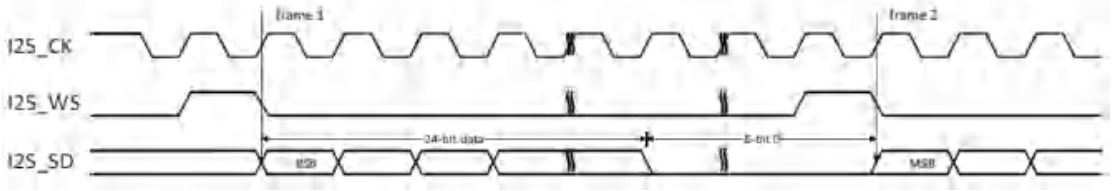


图 34-27 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)

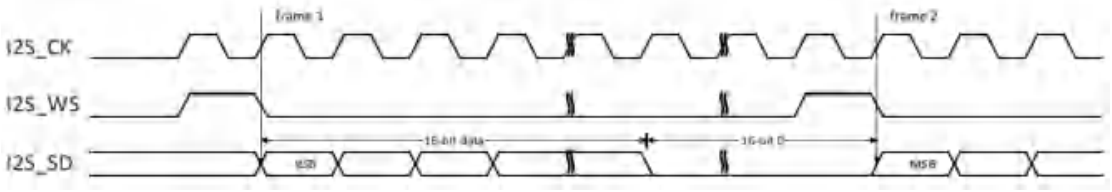
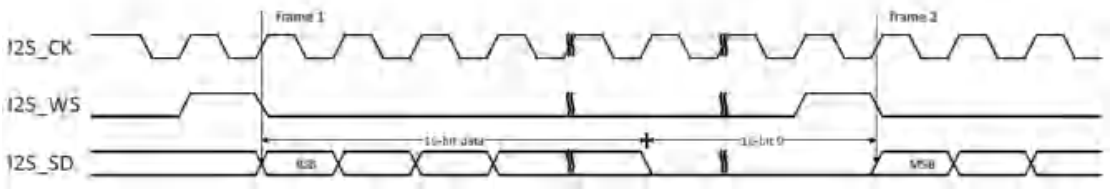


图 34-28 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)



长帧同步模式的各种配置情况时序图如下所示。

图 34-29 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)

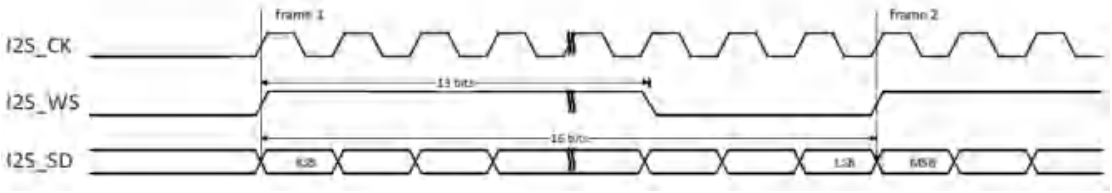


图 34-30 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)

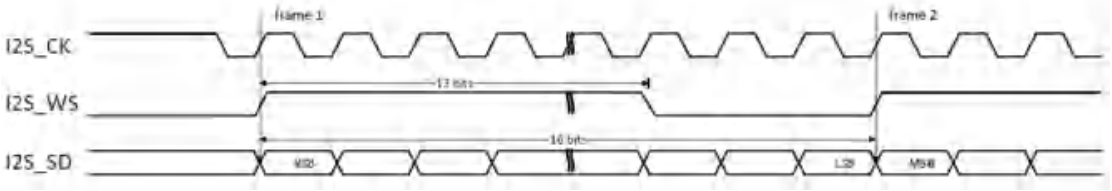


图 34-31 PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)

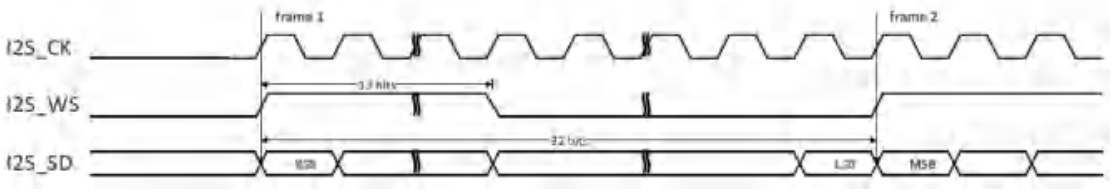


图 34-32 PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)

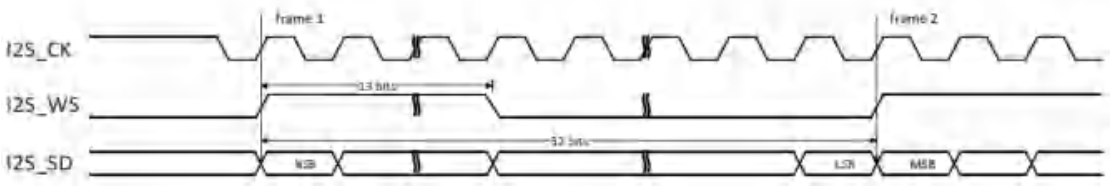


图 34-33 PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

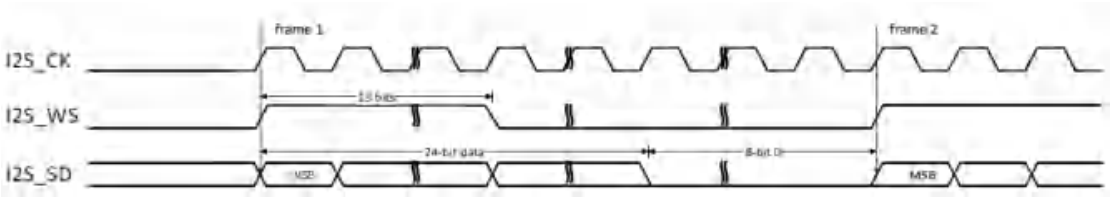


图 34-34 PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

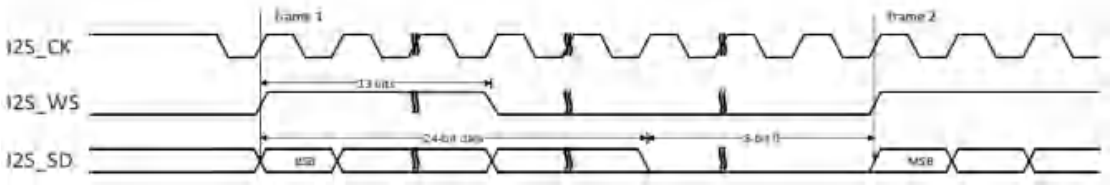


图 34-35 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)

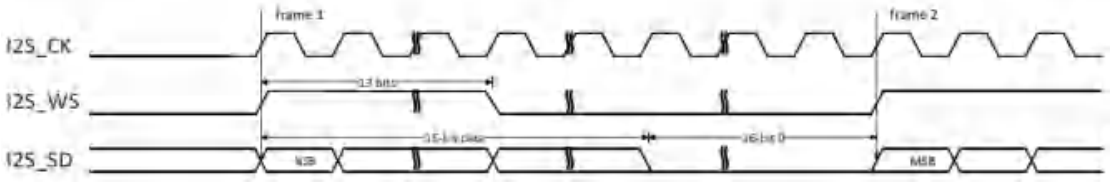
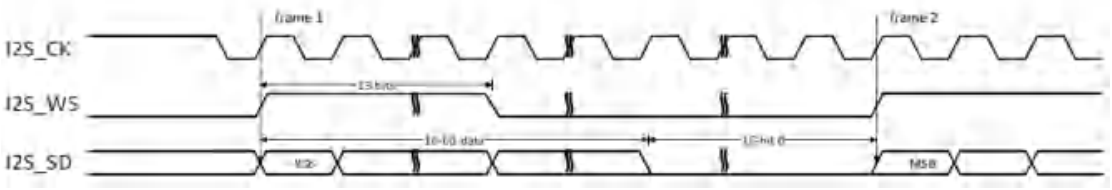


图 34-36 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)



34.4.7. 运行模式

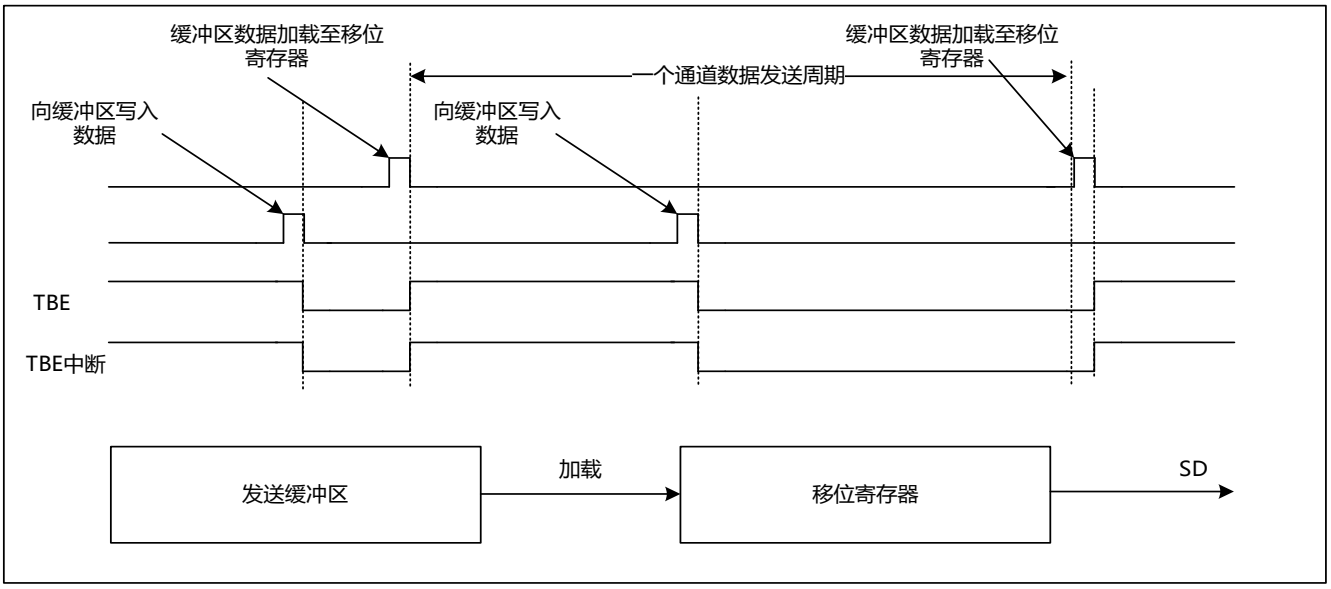
I2S 运行方式通过 I2S_CR.MODE/I2S_CR.TEN/I2S_CR.REN 位进行配置，可配置为从机发送模式、从机接收模式、主机发送模式、主机接收模式、主机全双工模式和从机全双工模式六种运行模式。

当配置为主机发送模式、主机接收模式时，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 I2S_PR.MCKOE 位来选择输出或者不输出主时钟（MCK)。

34.4.8. 主发模式

TXE 标志位被用来控制发送流程。如前文所述，TXE 标志位表示发送缓冲区空，此时，如果 I2S_IE 寄存器的 TXEIE 位为 1，将产生中断。首先，发送缓冲区为空(TXE 为 1)，且移位寄存器中没有发送序列。先配置 I2S_CR.START 位，当 32 位数据被写入 I2S_DR 寄存器时(TXE 变为 0)，数据立即从发送缓冲区装载到移位寄存器中(TXE 变为 1)。此时，发送序列开始。

数据是并行地装载到 32 位移位寄存器中的，然后串行地从 I2S_SD 引脚发出（高位先发)。下一个数据应该在 TXE 为 1 时写入 I2S_DR 寄存器。数据写入 I2S_DR 寄存器之后，TXE 变为 0。当前发送序列结束时，发送缓冲区的数据会自动装载到移位寄存器中，然后 TXE 标志变回 1。为了保证连续的音频数据发送，下一个将要发送的数据必须在当前发送序列结束之前写入 I2S_DR 寄存器。

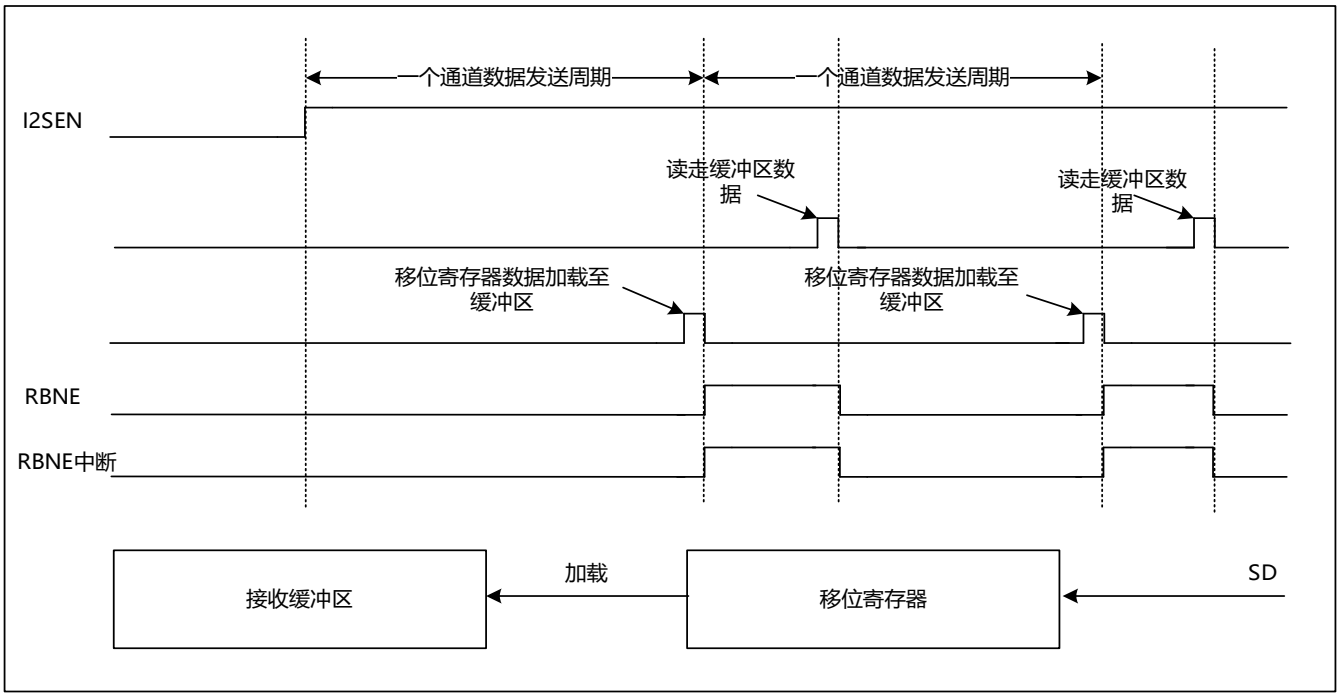


对于除 PCM 标准外的所有标准，I2SCH 标志用来区别当前传输数据所属的通道。I2SCH 标志在每次 TXE 标志由 0 变 1 的时候更新。刚开始 I2SCH 标志为 0，表示左通道的数据应该被写入 I2S_DR 寄存器。

为了关闭 I2S，I2SE 位必须在 MSUSP 标志为 1 之后清零。

34.4.9. 主收模式

RXNE 标志被用来控制接收序列。如前文所述，RXNE 标志表示接收缓冲区非空，如果 I2S_IE 寄存器的 RXNEIE 位为 1，将产生中断。I2S_CR 寄存器的 I2SE 位被置 1，当 I2S_CR.START 置位后，接收流程立即开始。首先，接收缓冲区为空(RXNE 为 0)。当一个接收流程结束时，接收到的数据将从移位寄存器装载到接收缓冲区(RXNE 变为 1)。当 RXNE 为 1 时，用户应该将数据从 I2S_DR 寄存器中读走。读操作完成后，RXNE 变为 0。必须在下一次接收结束之前读走 I2S_DR 寄存器中的数据。



对于除 PCM 之外的所有标准来说，I2SCH 标志用来区分当前传输数据所属的通道。I2SCH 标志在每次 RXNE 标志由 0 变 1 时更新。

34.4.9.1. 主模式批量接受数据

当在主机接受模式时，可以通过配置 I2S_RSIZE.RSIZE 来控制需要接受的数据量，当为零时，数据将会无限接

收，当接收数据寄存器空时，立即发送时钟并接收数据。当不为零时，将会接收 I2S_RSIZE.RSIZE 帧数据后停止接收，不再发送时钟。

34.4.9.2. 主模式接收数据停止

当在主机接受模式时，可通过 I2S_CR.STOP 置位来停止接收，当 mcu 从 I2S_RXDR 中读走倒数第二笔数据后立马打开 I2S_CR.STOP 位，在最后一笔数据接受完后，设备自动停止接收，不再发送时钟。

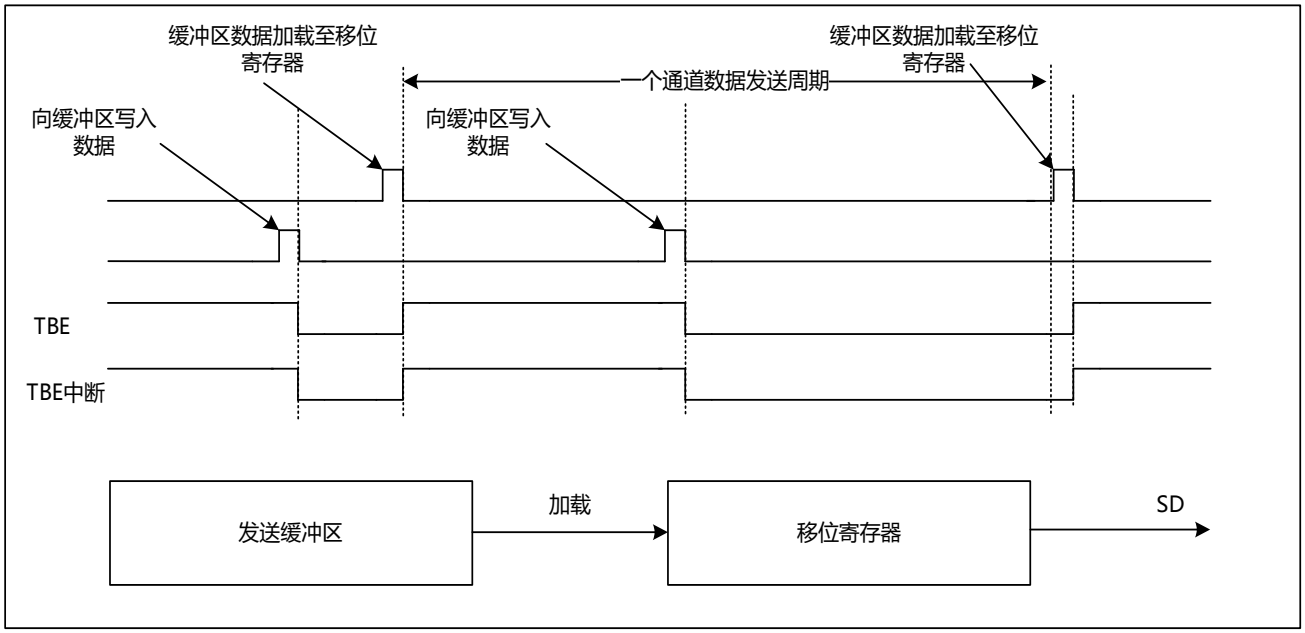
34.4.10. 从发模式

从机发送流程和主机发送流程相似，不同之处如下：

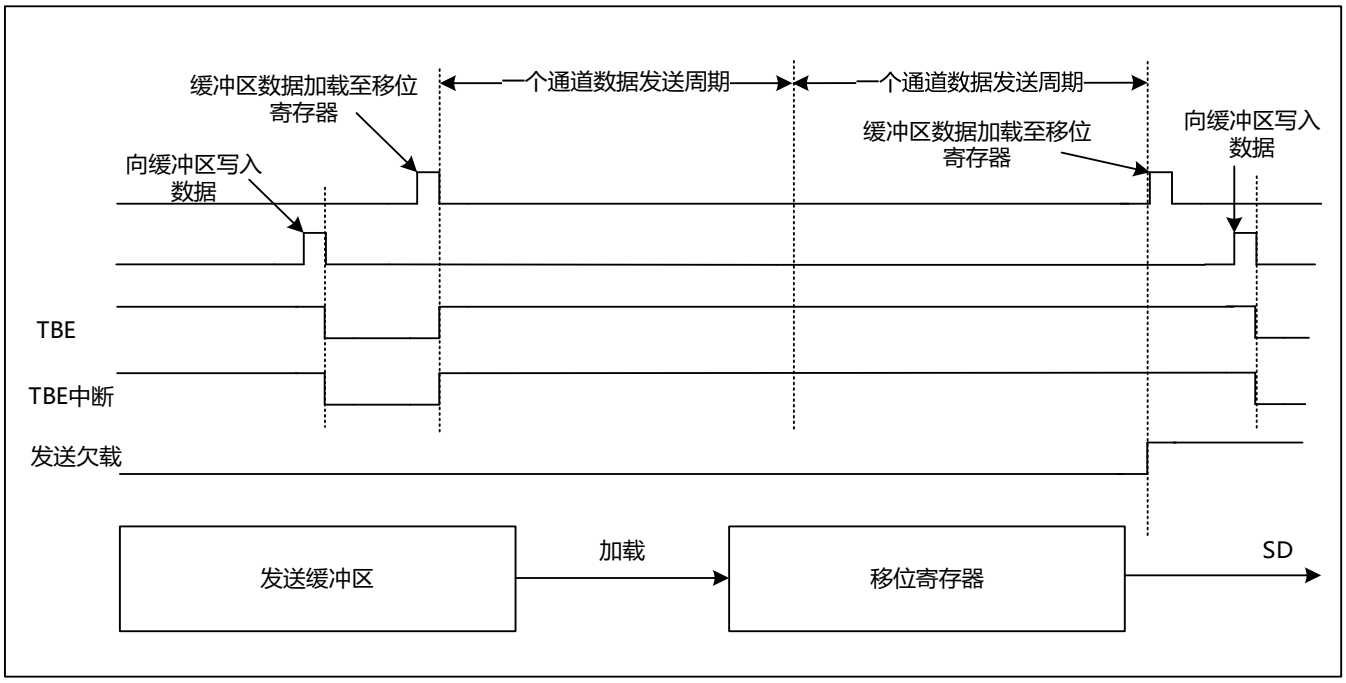
在从机模式下，从机需要在外部主机开始通讯之前使能。当外部主机开始发送时钟信号且 I2S_WS 信号请求传输数据时，发送流程开始。数据需要在外部主机发起通讯之前写入 I2S_TXDR 寄存器。为了确保音频数据的连续传输，必须在当前发送序列结束之前将下一个待发送的数据写入 I2S_TXDR 寄存器，否则会产生发送欠载错误。此时 UDR 标志会置 1，如果 I2S_IE 寄存器的 ERRIE 位为 1，将会产生中断。这种情况下，必须先关闭 I2S 再打开 I2S 来恢复通讯。从机模式下，I2SCH 标志是根据外部主机发送的 I2S_WS 信号而变化的。

为关闭 I2S，必须在 TXE 标志变为 1 且 SVTC 标志变为 1 之后，才能清除 EN 位。

下图为在当前发送序列结束之前将下一个待发送的数据写入 I2S_TXDR 寄存器，未发生发送欠载错误的示意图，与主发流程相同。



下图为在当前发送序列结束之前未将下一个待发送的数据写入 I2S_DR 寄存器，发生发送欠载错误的示意图。

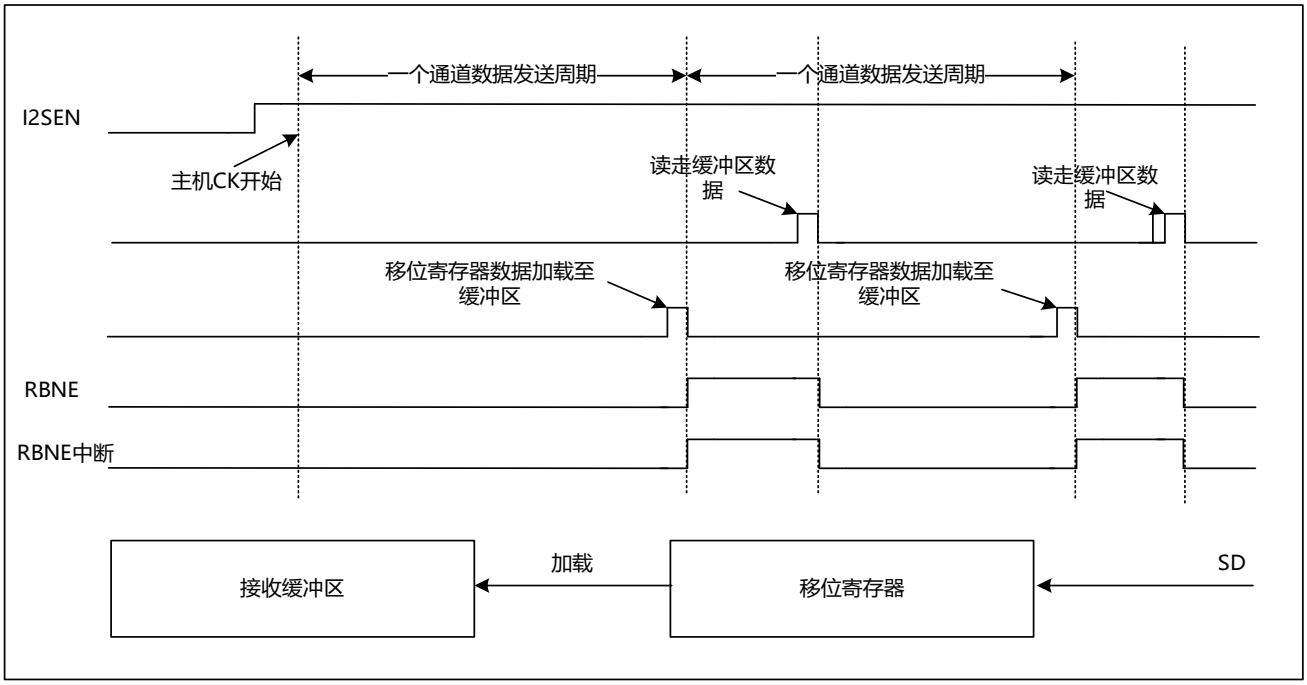


34.4.11. 从收模式

从机接收流程与主机接收流程类似，不同之处如下。

在从机模式下，从机需要在外部主机开始通讯之前使能。当外部主机开始发送时钟信号且 I2S_WS 信号指示数据开始时，接收流程开始。从机模式下，I2SCH 标志是根据外部主机发送的 I2S_WS 信号而变化的。

为了关闭 I2S，必须在收到最后一个 RXNE 之后立即清除 I2SE 位。



34.4.12. 状态标志

- 接收缓冲区非空（RXNE）

主或从模式的接收缓冲区接收到数据后置一，读接收缓冲区将复位该标志位。使能对应的中断使能或 DMA 使能，相应的中断或 DMA 请求将产生或清除。

- 发送缓冲区空（TXE）

主或从模式的发送缓冲区写入数据后清零，数据载入移位寄存器将置一。使能对应的中断使能或 DMA 使能，相应的中断或 DMA 请求将产生或清除。

● I2S 通道标志（I2SCH）

该标志位表示下一个将要传输的数据是属于左通道还是右通道，但在 PCM 模式下，该位无意义。

当在从机发送模式下发生发送欠载错误（UDR）时，该位将不可靠。

在接收模式下，此标志指示已接收的数据所属的通道。注意，如果发生错误（例如 OVR），此标志将失去意义，若需恢复，需要重新将模块复位后才可恢复。

● 发送欠载错误标志（UDR）

仅在从发送模式下，如果在软件尚未将任何值加载到 I2S_TXDR 之前出现第一个数据发送时钟，此标志将置 1。如果 I2S_DIER.ERRIE 使能，中断置起，读状态寄存器将清除标志。

● 接收过载错误标志（OVR）

主或从机尚未从 I2S_RXDR 中读取上一个数据又接收到新数据，此标志将置 1，传入的数据将丢失。如果 I2S_DIER.ERRIE 使能，中断置起。对 I2S_RXDR 寄存器执行读操作将返回先前正确接收的数据。

对于主机，此标志将置 1 后，将不会再发送 CK 直到读走 I2S_RXDR 后继续发送，因此从机发送的数据会丢失一个。对于从机，此标志将置 1 后，主器件后续发送的所有其它数据都将丢失。

要将 OVR 位清零，应首先对 I2S_RXDR 寄存器执行读操作，然后再读 SPI_SR 寄存器即可清除该标志。

● 主机发送挂起标志（MSUSP）

此位为一表示发送移位寄存器中数据发送完毕且发送缓冲区为空。设备进入挂起等待状态。

● 从机有效数据发送完成标志（SVTC）

此位为一表示发送移位寄存器中数据发送完毕且发送缓冲区为空。该位可作为从机发送结束标志来关闭从机。

● 帧错误(FE)

仅当 I2S 配置为从模式时，此标志才可由硬件置 1。如果外部主器件没有按照从器件期望的那样改变 WS 信号，则此标志将置 1。

如果从机在传输的过程中出现帧错误，会出现一到两帧数据错误后，硬件自动纠正之后正确传输。

34.4.13. 中断

中断事件	中断标志	使能位
接收缓冲区非空中断	RXNE	RXNEIE
发送缓冲区空中断	TXE	TXEIE
错误中断	OVR	ERRIE
	UDR	
	FE	

34.4.14. DMA

DMA 方式可有效的提高音频数据传输效率，可在发送完当前帧之前就向 TX 缓冲区数据写入下一通道传输数据或在下一通道数据接收完之前读走 RX 缓冲区中数据，通过 I2S_CR 中的 TXDMATEN/RXDMAREN 位开启使能，DMA 置位和复位与中断方式一致，时序可参看四种运行模式示意图下的 TXE、RXNE 中断。

34.5. 配置流程

34.5.1. 主发模式

1）配置 I2S_PR.DIV[8:0]位、OF 位和 MCKOE 位，定义 I2S 的比特率和选择是否需要提供 I2S_MCK 信号。

2）配置 I2S_CR.CKPL 位，定义空闲状态的时钟极性。

3）配置 I2S_CR.I2SSTD 位、PCMMOD 位、MODE 位、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

4）配置 I2S_IER.TXEIE 位、ERRIE 位和 I2S_CR.TXDMAEN 位，选择中断源和 DMA 功能。此步骤可选。

5）将 I2S_CR.I2SE 位置 1，启动 I2S。

6）先置位 I2S_CR.START，再等待 I2S_SR.TXE 置位，向 I2S_TXDR 地址写入音频数据，I2S 立即开始传输。

7）通过中断方式等待传输完成，当所有数据发送完成，等待 I2S_SR.MSUSP 置位，I2S_CR.I2SE 位复位，关闭 I2S。

34.5.2. 主收模式

1）配置 I2S_PR.DIV[7:0]位、OF 位和 MCKOE 位，定义 I2S 的比特率和选择是否需要提供 I2S_MCK 信号。

2）配置 I2S_CR.CKPL 位，定义空闲状态的时钟极性。

3）配置 I2S_CR.STD 位、PCMMOD 位、TEN 位、REN、MODE、CKPL、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

4）配置 I2S_IER.RXNEIE 位、ERRIE 位和 I2S_CR.RXDMAEN 位，选择中断源和 DMA 功能。此步骤可选。

5）将 I2S_CR.I2SE 位置 1，启动 I2S，置位 I2S_CR.START 后传输立即开始。

6）根据 RXNE 标志位进行数据读取。

7）当收完最后一个数据后 I2S_CR.I2SE 位置 0，关闭 I2S。

34.5.3. 从发模式

1）配置 I2S_CR.CKPL 位，定义空闲状态的时钟极性。

2）配置 I2S_CR.STD 位、PCMMOD 位、TEN 位、REN、MODE、CKPL、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

3）配置 I2S_IER.TBEIE 位、ERRIE 位和 I2S_CR.TXDMAEN 位，选择中断源和 DMA 功能。此步骤可选。

4）将 I2S_CR.I2SE 位置 1，启动 I2S，该位应在主设备使能前打开。

5）等待 I2S_SR. TXE 置位，向 I2S_DR 地址写入音频数据，应在主设备 CK 发出前写入，否则会出现发送欠载。

6）通过 SVTC 中断方式等待传输完成，当所有数据发送完成，I2S_CR.I2SE 位置 0，关闭 I2S。

34.5.4. 从收模式

1）配置 I2S_CR.CKPL 位，定义空闲状态的时钟极性。

2）配置 I2S_CR.STD 位、PCMMOD 位、TEN 位、REN、MODE、CKPL、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

3）配置 I2S_IER.RXNEIE 位、ERRIE 位和 I2S_CR.RXDMAEN 位，选择中断源和 DMA 功能。此步骤可选。

4）将 I2S_CR.I2SE 位置 1，启动 I2S，该位应在主设备使能前打开。

5）根据 RXNE 标志位进行数据读取。

6）当最后一比数据读完后，I2S_CR.I2SE 位置 0，关闭 I2S。

34.6. I2S 寄存器描述

34.6.1. 寄存器列表

I2S1 寄存器基地址：0x40003400

I2S2 寄存器基地址：0x40003800

I2S3 寄存器基地址：0x40003C00

偏移	名称	复位值	描述
0x00	I2S_TXDR	0x00000000	发送数据寄存器
0x04	I2S_RXDR	0x00000000	接收数据寄存器
0x08	I2S_CR	0x00000000	控制寄存器
0x0C	I2S_PR	0x00000000	时钟预分频寄存器
0x10	I2S_IER	0x00000000	中断使能寄存器
0x14	I2S_SR	0x00000002	状态寄存器
0x18	RSV	0x00000000	保留
0x1C	I2S_RSIZE	0x00000000	数据量寄存器

34.6.2. 发送数据寄存器(I2S_TXDR: 00h)

位域	名称	属性	复位值	描述
31 :0	TXDR	RW	0x0	发送数据寄存器 硬件有两个缓冲区：发送缓冲区和接收缓冲区。向 I2S_TXDR 写数据将会把数据存入发送缓冲区。

34.6.3. 接收数据寄存器(I2S_RXDR: 04h)

位域	名称	属性	复位值	描述
31 :0	RXDR	RW	0x0	数据寄存器 硬件有两个缓冲区：发送缓冲区和接收缓冲区。从 I2S_RXDR 读数据，将从接收缓冲区获得数据。

34.6.4. 控制寄存器(I2S_CR: 08h)

位域	名称	属性	复位值	描述
31: 17	RSV	-	-	保留

16	TXDMAEN	RW	0x0	发送缓冲区 DMA 使能 0：发送缓冲区 DMA 禁止 1：发送缓冲区 DMA 使能。当 I2S_SR 中的 TBE 置位时，将会在相应的 DMA 通道上产生一个 DMA 请求。
15	RXDMAEN	RW	0x0	接收缓冲区 DMA 使能 0：接收缓冲区 DMA 禁止 1：接收缓冲区 DMA 使能。当 I2S_SR 中的 RBNE 置位时，将会在相应的 DMA 通道上产生一个 DMA 请求。
14	STOP	RW	0x0	停止使能 1：停止 0：运行 下一帧数据传输完成后停止传输。 仅在主模式下有效。 主要用于主机接收数据。
13	START	WO	0x0	启动传输（start transfer） 1：启动传输 0：不启动传输 该位仅在主模式下 I2SE 使能后有效，由软件置一，硬件清零；从模式下 I2SE 使能后即可传输。
12	IOSWP	RW	0x0	交换 MISO 和 MOSI 引脚的功能 0：不交换 1：交换 MOSI 和 MISO 引脚功能 该位置 1 时， MISO 和 MOSI 引脚复用功能互换。 原 MISO 引脚变为 MOSI 引脚，原 MOSI 引脚变为 MISO 引脚。
11	EN	RW	0x0	I2S 使能 0：I2S 禁止； 1：I2S 使能；
10	REN	RW	0x0	接收使能位 1：接受使能 0：接收不使能 注：若使用全双工模式时，TEN 和 REN 需要同时写入寄存器。
9	TEN	RW	0x0	发送使能位 1：发送使能 0：发送不使能 注：若使用全双工模式时，TEN 和 REN 需要同时写入寄存器。
8	MODE	RW	0x0	主从模式位 1：主机模式 0：从机模式 当 I2S 模式关闭时配置该位。
7	PCMMODE	RW	0x0	PCM 帧同步模式 0：短帧同步 1：长帧同步 只有在 PCM 标准下，该位才有意义。 当 I2S 模式关闭时配置该位。

6	RSV	-	-	保留
5: 4	STD	RW	0x0	I2S 标准选择 00：I2S 飞利浦标准 01：MSB 对齐标准 10：LSB 对齐标准 11：PCM 标准 当 I2S 模式关闭时配置该位。
3	CKPL	RW	0x0	空闲状态时钟极性 0：I2S_CK 空闲状态为低电平 1：I2S_CK 空闲状态为高电平 当 I2S 模式关闭时配置该位。
2:1	DTLEN	RW	0x0	数据长度 00：16 位 01：24 位 10：32 位 11：保留 当 I2S 模式关闭时配置该位。
0	CHLEN	RW	0x0	通道长度 0：16 位 1：32 位 通道长度必须大于或等于数据长度。 当 I2S 模式关闭时配置该位。

34.6.5. 时钟预分频寄存器(I2S_PR: 0Ch)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	MCKOE	RW	0x0	I2S_MCK 输出使能 0：I2S_MCK 输出禁止 1：I2S_MCK 输出使能 当 I2S 模式关闭时配置该位。
9	OF	RW	0x0	预分频器的奇系数 0：实际分频系数为 DIV * 2 1：实际分频系数为 DIV * 2 + 1 当 I2S 模式关闭时配置该位。
8:0	DIV	RW	0x0	预分频器的分频系数 实际分频系数是 DIV * 2 + OF。 DIV 不能为 0. 当 I2S 模式关闭时配置该位。

34.6.6. I2S 中断使能寄存器(I2S_IER: 10h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	SVTCIE	RW	0x0	从机有效数据发送完成中断使能 0：中断禁止 1：中断使能
9	MSUSPIE	RW	0x0	主机发送挂起中断使能 0：中断禁止 1：中断使能
8:6	RSV	-	-	保留
5	ERRIE	RW	0x0	错误中断使能 0：错误中断禁止 1：错误中断使能。当 FE、OVR 位或者 UDR 位置 1 时，产生中断。
4:2	RSV	-	-	保留
1	TXEIE	RW	0x0	发送缓冲区空中断使能 0：TBE 中断禁止 1：TBE 中断使能。当 TBE 置位时，产生中断。
0	RXNEIE	RW	0x0	接收缓冲区非空中断使能 0：RBNE 中断禁止. 1：RBNE 中断使能。当 RBNE 置位时，产生中断。

34.6.7. 状态寄存器(I2S_SR: 14h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	SVTC	RW	0x0	从机有效数据发送完成标志 0：有效数据发送未完成 1：有效数据发送完成 此位为一表示发送移位寄存器中数据发送完毕且发送缓冲区为空。 该位可作为从机发送结束标志来关闭从机。 该位硬件置一，软件对该位写 0 清除中断
9	MSUSP	RW	0x0	主机发送挂起标志 0：传输中 1：挂起 此位为一表示发送移位寄存器中数据发送完毕且发送缓冲区为空。设备进入挂起等待状态； 该位硬件置一，软件对该位写 0 清除中断
8:6	RSV	-	-	保留

5	FE	RC W0	0x0	帧错误： 0：没有 I2S 帧错误发生 1：I2S 帧错误发生 该位由硬件置位，可以通过该位写 0 清除。
4	OVR	RO	0x0	接收过载错误标志 0：没有接收过载错误发生 1：接收过载错误发生 该位由硬件置位，软件序列清零。软件序列为：先读 I2S_DR 寄存器，然后通过该位写 0 清除。
3	UDR	RO	0x0	发送欠载错误标志 0：无发送欠载错误发生 1：发送欠载错误发生 该位由硬件置位，通过该位写 0 清除。
2	CH	RO	0x0	I2S 通道标志 0：下一个将要发送或接收的数据属于左通道 1：下一个要发送或接收的数据属于右通道 该位由硬件置位和清除。 I2S PCM 模式下该位没有意义。
1	TXE	RO	0x1	发送缓冲区空 0：发送缓冲区非空 1：发送缓冲区空
0	RXNE	RO	0x0	接收缓冲区非空 0：接收缓冲区空 1：接收缓冲区非空

34.6.8. 数据量寄存器（I2S_RSIZE: 1Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	RSIZE	RW	0x0	主机接收数据量值 0x0：该功能不使能； 非 0：主模式接收 RSIZE 帧数据。 例：设置该位为零，I2S 设备可接收无限数据量数据；当设置该位为 100；当接收到 100 帧数据后，不再发送 CK，CK 恢复到空闲状态。

35. 控制器局域网络（FDCAN）

35.1. 概述

FDCAN 模块是一个支持 CAN2.0B 和 CAN FD 功能的模块。

CAN 协议传输是以数据帧为基本单元。CAN 协议存在两种数据帧，标准帧和扩展帧。

CAN2.0B 最多支持 8 字节的数据帧，CANFD 最多能支持 64 字节的数据帧。

所有的 CAN 总线上的节点都是平等的，CAN 总线是一个多主的总线。但是 CAN 总线上同时只能有一个节点在发送数据。数据帧的识别是通过消息 ID 实现的，在一个 CAN 总线上，某一特定的 ID 只能由一个设备发送。CAN 总线上的所有 CAN 设备会接受 CAN 总线上的所有信息，但是 CAN 设备需要根据消息 ID 筛选自己需要的信息，可以由硬件的过滤组来实现过滤功能。

消息的 ID 也可以用于总线仲裁。低优先级 CAN ID 的设备在遇到高优先级 CAN ID 设备发送数据时候，自动停止发送，然后等到总线空闲的时候尝试重新发送数据。

CAN2.0B 定义的波特率最高是 1MHz，而 FD 标准的波特率没有限制。FD 标准定义了一个速率切换的功能。FD 可以让数据部分以较高的速率传输，而帧头部分继续以较低的速率传输。

CAN FD 标准是一个向下兼容的标准，每个 CAN FD 节点都能够接受和发送 CAN2.0B 的数据帧。

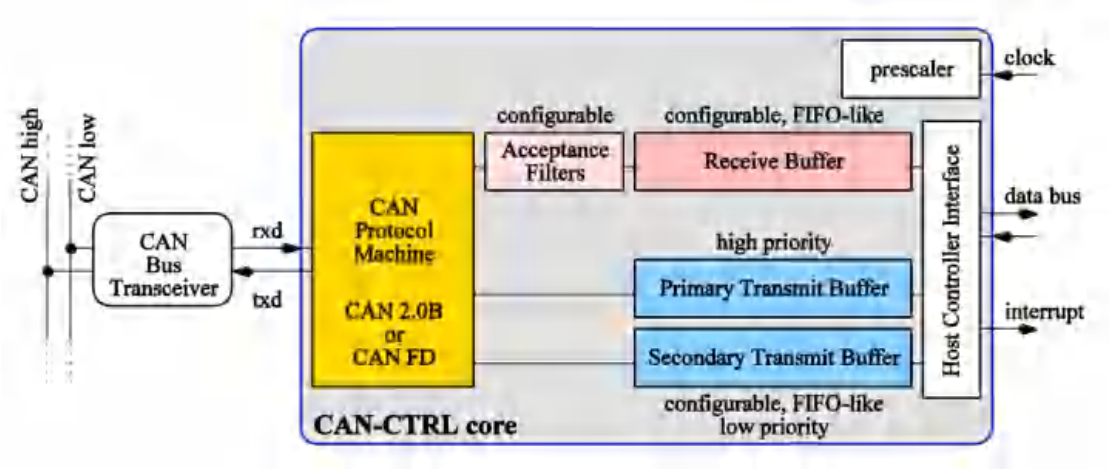
35.2. 主要特性

- 支持 CAN2.0B 和 FDCAN。
- CAN2.0B 最多支持 1M 的波特率。
- FDCAN 的波特率可以自由设置。
- 可编程的分频器（1 分频到 256 分频）
- 接受 FIFO 可以接受 16 帧数据。
- 发送 FIFO 分为主优先级发送缓冲（PTB）和次优先级发送缓冲（STB）
- STB 支持 FIFO 模式和优先级模式
- 支持 16 组 29 比特的过滤组
- 支持 Single Shot 发送模式
- 支持监听和回环模式
- 支持低功耗模式
- 支持时间戳功能，支持 ISO11898-4 时间戳和 CiA 603 时间戳
- 支持 AUTOSAR 和 SAE J1939

35.3. 功能描述

35.3.1. 结构框图

图 35-1 FDCAN 结构框图



35.3.2. 帧格式

图 35-2 CAN2.0 和 FDCAN 帧格式

表 35-1 CAN Bit 缩写词

Abbreviation	Description	Comment
ID	IDentifier	
RTR	Remote Transmission Request	Remote or Data frame
SRR	Substitute Remote Request	
RRS	Remote Request Substitution	
IDE	IDentifier Extension	Standard or Extended frame
DLC	Data Length Code	Number of payload bytes
EDL	Extended Data Length	CAN 2.0 or CAN FD frame
FDF	FD Format indicator (=EDL)	CAN 2.0 or CAN FD frame
BRS	Bit Rate Switch	
ESI	Error State Indicator	
r1, r0, res	Reserved bits	

CAN FD 支持两种标准，ISO 标准 Bosch 标准。Bosch 标准采用 EDL 字符，而 ISO 采用 FDF 字符。ISO 标准将填充计数放在 CRC 域，而 Bosch 标准中填充计数不算在 CRC 域。另外，ISO 标准和 Bosch 标准的 CRC 初始值是不一样的，所有两个标准是不兼容的。

35.3.3. 接受缓冲 RB

CAN 控制器接受到的 CAN 消息与过滤组的 ID 进行比较，如果通过了就被存放到接受缓冲 RB 中。RB 最多可以存放 16 帧 CAN 数据帧。CPU 通过寄存器接口，永远读到的是最早接受的消息帧。

35.3.4. 发送缓冲 TB

CAN 控制器具有 2 个发送缓冲，主发送缓冲 PTB 和次优先级发送缓冲 STB。PTB 具有更高的优先级，但是只能存放一帧数据。STB 具有低优先级，但是共有 16 个槽（或称作 slot），因此可以存放最多 16 个数据帧。STB 可以设置为 FIFO 模式和优先级模式。STB 可以一次传输一帧也可以一次传输所有帧。在 STB 的 FIFO 模式中，STB 中最早写入的数据帧被优先发送；在优先级模式中，具有最高优先级 ID 的消息将被优先发送。

TTCAN 模式下，STB 中的每个槽都被定位了。每个槽可以被标识为满或者空。TTCAN 总的发送是由触发器触发后发送的。每个触发器包括一个指向特定槽的指针。

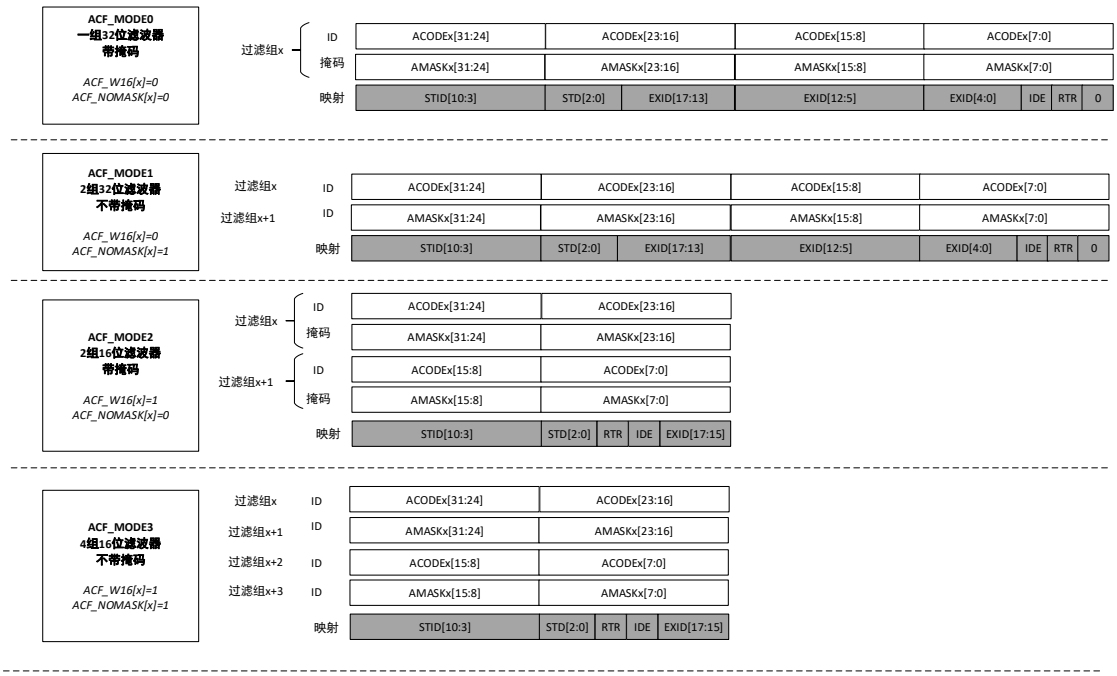
35.3.5. 过滤器

CAN 控制器提供了 16 组接受过滤组，用于过滤接受到的数据，从而降低 CPU 负荷，筛选器支持标准格式 11 位 ID 或者扩展格式 29 位 ID。每组过滤组都都有一个 32 位 ID CODE 寄存器和一个 32 位 ID MASK 寄存器，ID CODE 寄存器用于比较接受到的 ID，而 ID MASK 寄存器用于选择比较的 ID 位。如果对应的 ID MASK 位为 1，则不比较该位的 ID CODE。

接受到 CAN 消息只要通过 16 组接受过滤组中的任意一组，则数据就会被接受，并存储到接收缓存 RB 中，否则数据将被丢弃。

每组过滤器通过 ACFCR.AE 位使能/禁止。ID MASK 和 ID CODE 存器器通过 ACFCR.SELMASK 位设定，ACFCR.SELMASK 等于 0 时，则选择 ID CODE 寄存器，否则选择 ID MASK 寄存器。ID CODE 寄存器和 ID MASK 寄存器只能在 CR.RESET 等于 1 时设定。

图 35-3 过滤器



35.3.6. Single Shot 模式

不需要使用自动重新发送功能时，可以通过寄存器设定为 Single Shot（单次发送）模式。CR.TPSS 用于设定 PTB 的单次发送模式，CR.TSSS 位用于设定 STB 的单次发送模式。数据成功发送时单次发送和正常发送模式没有区别。但是数据发送失败时会出现下列情况：

- TPIF 置位，对应的 BUF SLOT 数据会被清除。
- 有错误发生时，KOER 更新，硬件将 BEIF 置位。
- 仲裁失败，硬件将 ALIF 置位。
- 单次发送模式下，发送完成不能单独依靠 TPIF 来判断，需要配合 BEIF 和 ALIF 标志一起判断。

35.3.7. 取消数据发送

- 可以通过 TPA 或者 TSA 取消已请求但还没有被执行的数据发送。取消数据发送会出现以下几种情况：
- 仲裁中
- 节点仲裁失败，则取消数据发送。
- 节点仲裁成功，则继续发送。
- 数据发送中
- 成功发送数据且收到 ACK，对应的标志和状态正常置位。数据发送不取消。
- 成功发送数据但没有收到 ACK，数据发送取消，错误计数器增加。
- TSALL=1 设定的发送数据，正在发送的 STB SLOT 数据正常发送，没有开始发送的 STB SLOT 被取消。
- 取消数据发送的结果有以下两种情况。
- TPA 释放 PTB，且使 TPE=0。
- TSA 释放一个 STB SLOT 或者全部 STB SLOT 取决是 TSONE 还是 TSALL 使能的发送。
-

35.3.8. 错误处理

- CAN 控制器一方面可以自动处理部分错误，比如自动重发数据或者丢弃接收到含有错误的帧，另一方面通过中断将错误向 CPU 报告。
- CAN 节点有以下三种错误状态：
- 错误主动：节点检测到错误时自动发送主动错误标志。
- 错误被动：节点检测到错误时自动发送被动错误标志。
- 节点关闭：关闭状态下此节点不再影响整个 CAN 网络。
- CAN 控制器提供 TECNT 和 RECNT 两个计数器用于计数错误。TECNT 和 RECNT 计数器按照 CAN2.0B 协议规定的规则进行增减。另外提供可编程的 CAN 错误警告 LIMIT 寄存器用于产生错误中断。
- CAN 通信过程中有以下 5 种错误类型，错误类型可以通过 EALCAP 寄存器的 KOER 位识别：
- 位错误。
- 形式错误。
- 填充错误。
- 应答错误。
- CRC 错误
-

35.3.9. 节点关闭

- 当发送错误数大于 255 时，CAN 节点自动进入节点关闭状态从而不参与 CAN 通信，直到返回到错误主动状态。可以通过 CR 寄存器的 BUSOFF 位确认 CAN 节点关闭状态。BUSOFF 被置位的同时 EIF 中断产生。
- CAN 从节点关闭状态状态恢复到错误主动状态有以下两种方法：
- 上电复位。
- 接收到连续 128 个 11 位的隐形位序列（恢复序列）。
- 节点关闭状态下，TECNT 值保持不变，RECNT 用于计数恢复序列。从节点关闭状态恢复后，TECNT 和 RECNT 被复位为 0。
- 节点关闭标志 BUSOFF 置位的同时，CR 寄存器的 RESET 位也被置位。
-

35.3.10. 仲裁失败位置捕捉

- CAN 控制器能够精确捕捉到仲裁失败位的位置并反映到 ALC 寄存器中。ALC 寄存器中保存着最近一次仲裁失败位的位置，如果节点赢得仲裁，则 ALC 位不更新。
- ALC 值定义如下：
- SOF 位后，第一个 ID 数据位 ALC 为 0，第二个 ID 数据位 ALC 为 1，依次类推。因为仲裁只发生在仲裁场内，所以 ALC 的最大值为 31。比如一个标准格式远程帧和一个扩展帧仲裁，扩展帧在 IDE 位失败，则 ALC=12。
-

35.3.11. 回环模式

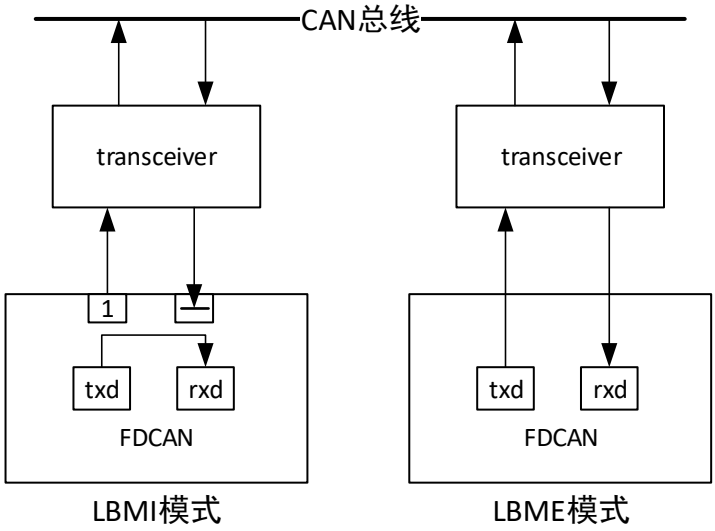
- CAN 控制器支持以下两种回环模式：

- 内部回环。
- 外部回环。
- 两种回环模式都可以接收自己发出的数据帧，主要用于测试用途。

内部回环模式，模块内部将接收数据线连接到发送数据线，并且发送数据不输出。内部回环模式下，节点会生成自应答信号以避免 ACK 错误。

外部回环模式保持和收发器的连接因此发送的数据仍能出现在 CAN 总线上，在收发器的帮助下，CAN 能收到自己发送的数据。外部回环模式可以通过 RCTRL 寄存器的 SACK 位来决定是否生成自应答信号 SACK=0 时，不生成自应答信号，SACK=1 时，生成自应答信号。

- 外部回环模式，SACK=0 时，会出现以下两种情况：
- 其它节点也收到本节点发送的数据帧并发送应答信号，该情况下本节点能够成功收发数据。
- 如果没有其它节点返回应答信号，则会产生应答错误，会重新发送数据并增加错误计数器。此时推荐采用单次发送模式。
- 从回环模式返回到正常模式时，除了清除模式位以外，还需要软件复位 CAN 控制器。



35.3.12. 静默模式

- 静默模式可以用来监控 CAN 网络数据。在静默模式下，可以从 CAN 总线接收数据，不向总线发送任何数据。将 CR 寄存器中的 LOM 置 1，使 CAN 总线控制器进入静默模式，将其清 0 可以离开静默模式。
- 外部回环模式可以和静默模式组合成外部回环静默模式，此时 CAN 可以认为一个安静的接收者，但在有必要的时候可以发送数据。外部回环静默模式下，帧包含自应答信号允许被发送，但是该节点不会产生错误标志和过载帧。
-

35.3.13. AUTOSAR 和 SAE J1939

本 IP 硬件上可以支持 AUTOSAR 和 SAE J1939。

35.3.14. Time-Triggered CAN

本 IP 支持时间戳功能，可以使用在 ISO11898-4 协议的 Time-Triggered CAN 通信（TTCAN）。TTCAN 最基本的要求是有一个定时器，用于给接受到的信息打上时间戳或者定时发送。在 CAN 网络中，有一个 CAN 设备节点是定时器主机，定时器主机发送时间参考消息。两个时间参考消息之间的时间就是最基本的循环周期。

TTCAN 系统定义了时间窗口，时间窗口内，设备节点可以发送数据。

存在三中时间窗口类型：

- 排他时间窗口。只能有一个节点发送消息。
- 自由时间窗口。暂未使用，为以后扩展使用。
- 仲裁时间窗口。多个节点同时发送，然后进行仲裁。

如果接受到一帧数据，那么将此时定时器的值作为此数据帧的时间戳。对于发送，CAN 设备可以由硬件触发器启动一个预定好的消息。CAN 设备自动检测时间参考消息，然后开始周期定时器。本 IP 中的定时器是一个 16 位的定时器，定时器的一个周期是一个位时间。 除了硬件触发器支持消息发送，本 IP 还提供了一个看门狗触发器，用来检测丢失的参考消息。硬件的部分支持意味着，主机软件需要在每一个时间窗口内准备好需要执行的操作。比如，主机需要准备好下一个需要发送的消息数据，并且定义好下一个要触发的时间点。换句话说，消息矩阵的传输是需要主机软件配合参与的。

本章从以下 5 个部分介绍 TTCAN 功能。

35.3.14.1. TTCAN 模式下的 TBUF 行为

TTTBM=1 时，PTB 和 STB Slot 一样组成 TB Slot，通过 TBPTR 寄存器指定发送 BUF，其中 TBPTR=0 时，指向 PTB，TBPTR=1 时，指向 STB Slot1，依此类推。主机可以通过 TPE 和 TPF 寄存器来标记发送 BUF Slot。此时 TBSEL 和 TSNEXT 寄存器无任何意义从而可以被忽略。

TTTBM=1 时，PTB 不具有任何特殊的属性，和 STB Slot 一样，传送完成标志也采用 TSIF。

TTCAN 模式时，发送 BUF 没有 FIFO 模式和优先级仲裁模式，同时也只有一个选定的 Slot 可以发送数据。

TTCAN 模式下，传输开始需要采用时间触发方式，TPE、TSONE、TSALL、TPSS 和 TPA 被固定 0 且被忽略。

TTTBM=0 时，组合使用事件驱动通信和接受时间戳功能。在该模式下，PTB 和 STB 的功能和 TTEN=0 时一致，因此 PTB 始终具有最高的优先级，而 STB 可以工作在 FIFO 模式或仲裁模式。

35.3.14.2. TTCAN 功能描述

上电后，Time Master 需要根据 ISO11898-4 协议进行初始化。一个 CAN 网络中，最多可以有 8 个潜在的 Time Master。每个 Time Master 都有自己的参考消息 ID。这些潜在的 Time Master 根据自己的优先级发送各自的参考消息，

TTEN=1 后，16 位的计数器开始工作，当参考消息被成功接受 Time Master 成功发送参考消息时，CAN 控制器将 Sync_Mark 拷贝给 Ref_Mark，Ref_Mark 将 cycle time 设置为 0。成功接受参考消息置位 RIF 标志而成功发送参考消息置位 TPIF 标志或者 TSIF 标志。此时主机需要准备下一个动作的触发条件。

触发条件可以是接受触发。该触发仅触发中断可用于检测期待的消息没有被收到。触发条件也可以是发送触发，该触发开始发送通过 TTPTR 寄存器指定的 TBUF Slot 里的数据。如果选定的 TBUF Slot 被标记为空，则不开始发送，但置位中断标志。

35.3.14.3. TTCAN 时序

CAN 控制器支持 ISO11898-4 level 1，包括的一个 16 位计数器工作在为 PRESC、SEG_1、SEG_2 定义的位时间下。如果 TTEN=1，则有一个额外的预分频器 T_PRESC。一帧数据的 SOF 时，计数器的值为 Sync_Mark。如果该帧数据为参考消息，则将 Sync_Mark 拷贝给 Ref_Mark。Cycle time 等于计数器的值减去 Ref_Mark。该时间用作为接受消息的时间戳或者发送消息的触发时间基准。

35.3.14.4. TTCAN 触发方式

通过 TTYPE 寄存器定义 TTCAN 的触发方式，TTPTR 寄存器指定发送 Slot，而 TT_TRIG 指定触发器的周期时间。包含以下 5 种触发方式：

- 立即触发
  - 通过写 TT_TRIG 的高位，启动触发器。此模式下，TTPTR 选定的 TBUF Slot 内的数据会立即发送。TTIF 不置位。
- 时间触发
  - 时间触发方式仅通过置位 TTIF 标志产生中断，并无其他功能。如果一个节点期待在特定的时间窗口收到期待的数据，则可以使用时间触发方式。如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其他动作。
- 单次发送触发
  - 单次发送触发方式用于在执行时间窗口内发送数据。此时，忽略 TSSS 位。
  - 通过 TEW 位设置 ISO11898-4 规定的最多 16 个 cycle time 的 Tick，设置为 1~16。如果在规定的发送使能时间窗口内数据没有开始发送，则帧被丢弃。对应的发送 BUF Slot 被标记为空，并且置位 AIF，对应的发送 BUF 内的数据不会被改写，因为可以通过置位 TPF 再次发送。
  - 如果 TT_TRIG 值小于实际的周期时间，则 TEIF 置位且无其他动作。
- 发送开始触发
  - 发送开始触发方式用于仲裁时间窗口内，参与仲裁。TSSS 用于决定是否自动重发或者单次发送模式。如果指定的消息没有被成功发送，可以使用发送停止触发来停止该发送。
  - 如果 TT_TRIG 值小于实际的周期时间，则 TEIF 置位且无其他动作。
- 发送停止触发
  - 发送停止触发方式用于停止通过发送开始触发方式已经开始的发送。如果发送被停止，则发送帧被舍弃，置位 AIF 并将选定的 TBUF Slot 标记为空，但 TBUF Slot 内的时间不会被改写，可以通过置位 TPF 就可以再次发送。
  - 如果 TT_TRIG 值小于实际的周期时间，则 TEIF 置位且无其他动作。

35.3.14.5. TTCAN 触发看门时间

TTCAN 触发看门时间功能类似于看门狗功能，在 TTTBM=1 时使用。用来看门从上次成功接到参考消息开始的时间。参考消息可以在周期 cycle time 中或者一个事件后被接受，应用程序应该根据具体情况设定合适的看门时间。

如果 cycle count 等于 TT_WTRIG，则置位 WTIF。通过 WTIE 写 0，关闭看门触发。如果 TT_WTRIG 比实际的 cycle time 小，则 TEIF 值位。

35.3.15. CiA 603 Time-Stamping

CiA 603 具有一个自由运行的定时器，定时器的时钟并不是 CAN 的位时间。精度必须是 10us 或者 1us。本 IP 需要外一个外部自由运行的定时器，并连接到本 IP 的 Time_in 端口。

CiA 603 定义了至少 16 位时间戳的方法，此芯片 CAN 控制器支持 64 位时间戳，可以与 TTCAN 一起使用或者独立使用。

CiA 603 有一个自由运行的计时器，它计算时钟周期而不是 CAN 位次。CAN 控制器外部有自由运行定时器，

计数值连接到 CAN 控制器；计数器的时钟（默认 100Mhz）和 CAN 控制器的通信时钟异步和系统时钟也是异步，通信时钟域的事件来触发从 PTPC 计数器来的计数值作为时间戳，再到总线的时钟域，最后回到 can_clk 时钟域。

在 SOF 或 EOF 位的采样点处获取时间戳，其中帧被取为有效的。这可以通过配置位 TIMEPOS 来选择。ACK 后的七个隐形位分隔符形成 EOF。

在许多系统中，广泛使用的基于软件的时间戳依赖于接受和传输中断，因此，建议在 EOF 上加时间戳。

CiA 603 应该支持 AUTOSAR 的时间戳和时间同步，用于 AUTOSAR CAN 网络中的一个节点是时间主站。时间主站发送同步消息。SYNC 消息的时间戳由时间主机和所有时间获取从机。从命令 SYNC 消息事件到发出 SYNC 消息的事件之间的时间差实际传输的 SYNC 消息将在后续消息中由时间。因此此 IP 只支持传输帧 TTS 的一个时间戳，但所有接受帧的单独时间戳。为传输帧生成时间戳可以使用 TBUFTTSEN 为每帧单独启用或禁止。

CiA 定义了读取定时器和修改定时器的规则，CAN-CTRL 不包括定时器，但使用外部定时器。CAN-CTRL 只包括时间戳机制，寄存器用于存储 TTS 和用于存储每个帧的 RTS 的内存。

寄存器位 TIMEEN 启动或禁止时间戳，如果禁用，TTS 和 RTS 无效。

35.4. CAN 寄存器描述

35.4.1. 寄存器列表

FDCAN1 寄存器基地址：0x40022000

FDCAN2 寄存器基地址：0x40022400

FDCAN3 寄存器基地址：0x50001400

偏移	名称	复位值	描述
0x00~0x4F	FDCAN_RBUF	0x00000000	接受缓存
0x50~0x97	FDCAN_TBUF	0x00000000	发送缓存
0x98	FDCAN_TTSL	0x00000000	时间戳低位寄存器
0x9C	FDCAN_TTSH	0x00000000	时间戳高位寄存器
0xA0	FDCAN_CR	0x00900080	控制寄存器
0xA4	FDCAN_IR	0x000000fe	中断寄存器
0xA8	FDCAN_LIMIT	0x0000001b	阈值寄存器
0xAC	FDCAN_SBTR	0x01020203	慢位时间寄存器
0xB0	FDCAN_FBTR	0x01020203	快位时间寄存器
0xB4	FDCAN_TDC	0x00000000	发送补偿寄存器
0xB8	FDCAN_ECC	0x00000000	错误寄存器
0xBC	FDCAN_ACFCR	0x00010000	过滤组控制寄存器
0xC0	FDCAN_ACFMODE	0x00000000	过滤组模式寄存器
0xC4	FDCAN_ACODR	0x00000000	过滤组代码寄存器
0xC8	FDCAN_TIMCFG	0x00000002	时间戳控制寄存器
0xCC	FDCAN_TTCFG	0x00009000	TTCAN 控制寄存器
0xD0	FDCAN_REFMSG	0x00000000	TTCAN 参考消息寄存器
0xD4	FDCAN_TRIGCFG	0x00000000	时间控制寄存器
0xD8	FDCAN_TTTRIG	0x00000000	时间触发寄存器
0xDC	FDCAN_WTRIG	0x00000000	Watch 触发寄存器

35.4.2. 接受缓存(FDCAN_RBUF: 0h)

位域	名称	属性	复位值	描述
31:0	RBUF	RO	0	发送时间戳[31:0]

Receive Buffer Registers RBuf-Standard Format (r-0)

Address	Bit position								Function
-	7	6	5	4	3	2	1	0	-
RBUF	ID(7:0)								Identifier
RBUF+1	-					ID(10:8)			Identifier
RBUF+2	-								Identifier
RBUF+3	ESI	-							Identifier
RBUF+4	IDE=0	RTR	FDF	BRS	DLC(3:0)				Control
RBUF+5	KOER			TX	-				Status
RBUF+6	CYCLE_TIME(7:0)								TTCAN
RBUF+7	CYCLE_TIME(15:8)								TTCAN
RBUF+8	d1(7:0)								Data B1
RBUF+9	d2(7:0)								Data B2
-	-								-
RBUF+71	d64(7:0)								Data B64
RBUF+72	RTS(7:0)								CiA 603
-	-								-
RBUF+79	RTS(63:56)								CiA 603

Receive Buffer Registers RBUF-Extended Format (r-0)

Address	Bit position								Function
-	7	6	5	4	3	2	1	0	-
RBUF	ID(7:0)								Identifier
RBUF+1	ID(15:8)								Identifier
RBUF+2	ID(23:16)								Identifier
RBUF+3	ESI	-		ID(28:24)					Identifier

RBUF+4	IDE=1	RTR	FDF	BRS	DLC(3:0)	Control
RBUF+5	KOER			TX	-	Status
RBUF+6	CYCLE TIME(7:0)					TTCAN
RBUF+7	CYCLE_TIME(15:8)					TTCAN
RBUF+8	d1(7:0)					Data B1
RBUF+9	d2(7:0)					Data B2
-	-					-
RBUF+71	d64(7:0)					Data B64
RBUF+72	RTS(7:0)					CiA 603
-	-					-
RBUF+79	RTS(63:56)					CiA 603

35.4.3. 发送缓存(FDCAN_TBUF: 50h)

位域	名称	属性	复位值	描述
31:0	TBUF	RO	0	发送时间戳[31:0]

Transmit Buffer Registers TBUF-Standard Format (rw)

Address	Bit position								Function
-	7	6	5	4	3	2	1	0	-
TBUF	ID(7:0)								Identifier
TBUF+1	-					ID(10:8)			Identifier
TBUF+2	-								Identifier
TBUF+3	TTSEN	-							Identifier
TBUF+4	IDE=0	RTR	FDF	BRS	DLC(3:0)				Control
TBUF+8	d1(7:0)								Data B1
TBUF+9	d2(7:0)								Data B2
-	-								-
TBUF+71	d64(7:0)								Data B64

Transmit Buffer Registers TBUF-Extended Format (rw)

Address	Bit position								Function
-	7	6	5	4	3	2	1	0	-
TBUF	ID(7:0)								Identifier
TBUF+1	ID(15:8)								Identifier
TBUF+2	ID(23:16)								Identifier
TBUF+3	TTSEN	-		ID(28:24)					Identifier
TBUF+4	IDE=1	RTR	FDF	BRS	DLC(3:0)				Control
TBUF+8	d1(7:0)								Data B1
TBUF+9	d2(7:0)								Data B2
-	-								-
TBUF+71	d64(7:0)								Data B64

35.4.4. 时间戳低位寄存器（FDCAN_TTSL: 98h)

位域	名称	属性	复位值	描述
31:0	TTS	RO	0	发送时间戳[31:0]

35.4.5. 时间戳高位寄存器(FDCAN_TTSH: 9Ch)

位域	名称	属性	复位值	描述
31:0	TTS	RO	0	发送时间戳[63:31]

35.4.6. 控制寄存器(FDCAN_CR: A0h)

位域	名称	属性	复位值	描述

31	SACK	R/W	0	自我应答 0：禁止 1：在 LBME=1 时，使能自我应答功能。
30	ROM	R/W	0	接受 FIFO 溢出模式 0：丢弃最早接受的数据帧 1：丢弃最新接受的数据帧
29	ROV	RO	0	接受 FIFO 溢出 0：没有溢出 1：溢出
28	RREL	R/W	0	释放接受 FIFO 0：不释放 1：释放接受 FIFO 的一个槽
27	RBALL	R/W	0	接受缓冲模式 0：正常模式 1：接受全部帧，包括有错误的数据帧
26	RSV	-	-	保留
25:24	RSTAT	RO	0	接受缓冲的状态 00：空 01：大于 empty 并小于 AFWL 10：大于等于 AFWL 但是并没有慢 11：已经满了。
23	FD_ISO	R/W	1	FD ISO 模式标准 0：Bosch CAN FD 模式 1：ISO CAN FD 模式
22	TSNEXT	R/W	0	STB Next 0：不发送 1：STB 槽填满了，选择下一个 FIFO 槽。 当 CPU 向 TBUF 中填好数据后，需要将 TSNEXT 置 1。
21	TSMODE	R/W	0	STB 操作模式 0：FIFO 模式 1：优先级模式
20	TTTBM	R/W	1	TTCAN 发送缓冲模式
19:18	RSV	-	-	保留
17:16	TSSTAT	RO	0	STB 已经存放的的数据帧数目

15	TBSEL	R/W	0	发送缓冲区选择位 0：选择 PTB 1：选择 STB 写入 TBUF 的数据被存在哪个缓冲区。
14	LOM	R/W	0	监听模式 0：禁止 1：使能
13	STBY	R/W	0	传输低功耗模式 0：禁止 1：使能 此位被接到外部的 stby 管脚，用于控制收发器进入低功耗模式。
12	TPE	R/W	0	使能 PTB 传输 0：不传输 1：传输 PTB 中的数据帧。
11	TPA	R/W	0	中断 PTB 传输 0：不中断 1：中断 PTB 中已经被请求过但是没有实际开始帧传输。
10	TSONE	R/W	0	传输 STB 中的一帧 0：不传输 1：传输 STB 中的最早请求的一帧。
9	TSALL	R/W	0	传输 STB 中的全部帧 0：不传输 1：传输在 STB 中的全部帧。
8	TSA	R/W	0	中断 STB 传输 0：不中断 1：中断 STB 的传输，已经被请求但是还没有真正开始传输。 对于 FSONE 传输，只中断一帧。 对于 TSALL 传输，所有帧都中断。
7	RESET	R/W	1	复位 0：不复位 IP 1：复位 IP 只有在 RESET=1 的情况下才能修改 BTR、PSCR、ACFENR、ACFCR 寄存器。 IP 进入 BusOff 以后会自动将 RESET 设置为 1。 将 RESET 设置为 0 后 11 个位时间后才能参与 CAN 总线通信。

6	LBME	R/W	0	外部环回模式 0：禁止 1：使能
5	LBMI	R/W	0	内部环回模式 0：禁止 1：使能
4	TPSS	R/W	0	PTB 处于 Single Shot 模式 0：禁止 1：使能
3	TSSS	R/W	0	STB 处于 Single Shot 模式 0：禁止 1：使能
2	RACTIVE	RO	0	接受激活状态位 0：IP 正在接受数据。 1：IP 没有接受数据。
1	TACTIVE	RO	0	发送激活状态位 0：IP 正在发送数据。 1：IP 没有发送数据。
0	BUSOFF	RO	0	Bus Off 状态位 0：IP 处于“Bus On”状态 1：IP 处于“Bus Off”状态

35.4.7. 中断使能和标志寄存器(FDCAN_IR: A4h)

位域	名称	属性	复位值	描述
23	EWARN	RO	0	错误报警标志 1：RECNT 或 h 者 TECNT 大于等于 EWL 0：没达到报警阈值 对本寄存器中的中断标志位写 1 清零
22	EPASS	RO	0	Error Passive 模式状态位
21	EPIE	R/W	0	Error Passive 中断使能 0：禁止 1：使能
20	EPIF	RO	0	Error Passive 中断标志

19	ALIE	R/W	0	仲裁失败中断使能 0：禁止 1：使能
18	ALIF	RO	0	仲裁失败中断标志
17	BEIE	R/W	0	Bus Error 中断使能 0：禁止 1：使能
16	BEIF	RO	0	Bus Error 中断标志
15	RIF	RO	0	接受中断标志
14	ROIF	RO	0	接受 FIFO 溢出中断标志
13	RFIF	RO	0	接受 FIFO 满中断标志
12	RAFIF	RO	0	接受 FIFO 即将满中断标志
11	TPIF	RO	0	PTB 中断标志
10	TSIF	RO	0	STB 中断标志
9	EIF	RO	0	错误中断标志
8	AIF	RO	0	Abort 中断标志
7	RIE	R/W	1	接受中断使能 0：禁止 1：使能
6	ROIE	R/W	1	接受 FIFO 溢出中断使能 0：禁止 1：使能
5	RFIE	R/W	1	接受 FIFO 满中断使能 0：禁止 1：使能
4	RAFIE	R/W	1	接受 FIFO 即将满中断使能 0：禁止 1：使能
3	TPIE	R/W	1	PTB 中断使能 0：禁止 1：使能
2	TSIE	R/W	1	STB 中断使能 0：禁止 1：使能

1	EIE	R/W	1	错误中断使能 0：禁止 1：使能
0	TSFF	RO	0	STB 满标志 0：STB 未满 1：STB 已满

注：对本寄存器中的中断标志位写 1 清零

35.4.8. 阈值寄存器(FDCAN_LIMIT: A8h)

位域	名称	属性	复位值	描述
7:4	AFWL	R/W	0x1	接受 FIFO 快满报警阈值 AFWL_i 与 RB 中的帧数目进行比较。
3:0	EWL	R/W	0xB	错误报警阈值 Limit = (EWL + 1)*8。

35.4.9. 慢位时间寄存器(FDCAN_SBTR: ACh)

位域	名称	属性	复位值	描述
31:24	S_PRESC	R/W	1	Slow 分频值 0：1 分频 1：2 分频 0xFF:256 分频
23	RSV	-	-	保留
22:16	S_SJW	R/W	0x2	Slow SJW 值 （S_SJW +1） TQ
15	RSV	-	-	保留
14:8	S_SEG_2	R/W	0x2	Slow Seg2 值 （S_Seg_2 + 1）TQ
7:0	S_SEG_1	R/W	0x3	Slow Seg1 值 采样点是位开始后的（S_Seg_1 + 2）TQ

35.4.10. 快位时间寄存器(FDCAN_FBTR: B0h)

位域	名称	属性	复位值	描述
31:24	F_PRESC	R/W	1	Fast 分频值 0：1 分频 1：2 分频 0xFF:256 分频
23:20	RSV	-	-	保留
19:16	F_SJW	R/W	0x2	Fast SJW 值 （F_SJW +1） TQ
15:12	RSV	-	-	保留
11:8	F_SEG_2	R/W	0x2	Fast Seg2 值 （F_Seg_2 + 1）TQ
7:5	RSV	-	-	保留
4:0	F_SEG_1	R/W	0x3	Fast Seg1 值 采样点是位开始后的（F_Seg_1 + 2）TQ

35.4.11. 发送延迟补偿寄存器(FDCAN_TDC: B4h)

位域	名称	属性	复位值	描述
7	TDCEN	R/W	0	发送延迟补偿功能使能
6:0	SSPOFF	R/W	0	STB 采样点偏移

35.4.12. 错误寄存器(FDCAN_ECC: B8h)

位域	名称	属性	复位值	描述
31:24	TECNT	RO	0	发送错误计数值
23:16	RECNT	RO	0	接受错误计数值
15:8	RSV	-	-	保留

7:5	KOER	RO	0	错误码 000：No Error 001：Bit Error 010：Form Error 011：Stuff Error 100：Ack Error 101：Crc Error 110：Other Error
4:0	ALC	RO	0	冲裁丢失位置

35.4.13. 过滤组使能寄存器（FDCAN_ACFCR: BCh)

位域	名称	属性	复位值	描述
31:16	AE	R/W	0x0001	接受过滤组使能位 接受过滤组 0 默认使能。
15:8	RSV	-	-	保留
7:6	RSV	-	-	保留
5	SELMASK	R/W	0	接受过滤组 MASK 寄存器选择 0：选择 Acceptance Code 寄存器 1：选择 Acceptance Mask 寄存器
4	RSV	-	-	保留
3:0	ACFADR	R/W	0	接受过滤组地址选择

35.4.14. 过滤组模式寄存器（FDCAN_ACFMODE: C0h)

位域	名称	属性	复位值	描述
31:16	ACF_W16	R/W	0x0000	过滤组宽度 0：32bit 1：16bit
15:0	ACF_NOMASK	R/W	0x0000	接受过滤组屏蔽模式 0：MASK 使能 1：MASK 无效

35.4.15. 过滤组控制寄存器(FDCAN_ACODR: C4h)

位域	名称	属性	复位值	描述
31:0	ACODE/AMASK	R/W	0	接受过滤组的 Code 值或者 Mask 值

35.4.16. 时间戳控制寄存器（FDCAN_TIMCFG: C8h)

位域	名称	属性	复位值	描述
15:10	RSV	-	-	保留
1	TIMEPOS	R/W	1	时间戳位置 0：SOF 1：EOF
0	TIMEEN	R/W	0	时间戳功能使能 0：禁止 1：使能

35.4.17. TTCAN 配置寄存器(FDCAN_TTCFG: CCh)

位域	名称	属性	复位值	描述
17	RFMSGIE	R/W	0	参考消息触发中断使能
16	RFMSGIF	R/W	0	参考消息触发中断标志
15	WTIE	R/W	1	看门狗触发中断使能
14	WTIF	R/W	0	看门狗触发中断标志
13	TEIF	R/W	0	触发错误中断标志
12	TTIE	R/W	1	时间触发中断使能
11	TTIF	R/W	0	时间触发中断标志

10:9	T_PRESC	R/W	0	时间分频值 0：1 分频 1：2 分频 2：4 分频 4：8 分频
8	TTEN	R/W	0	使能时间触发功能 0：禁止 1：使能
7	TBE	R/W	0	将发送槽设置为空 0：无操作 1：由 TBPTR 设置的槽被标识为空
6	TBF	R/W	0	将发送槽设置为满 0：无操作 1：由 TBPTR 设置的槽被标识为满
5:0	TBPTR	R/W	0	指向 TB 信息槽 0：指向 PTB 其他：指向 STB 的槽

35.4.18. 参考消息寄存器(FDCAN_REFMSG: D0h)

位域	名称	属性	复位值	描述
31	REF_IDE	R/W	0	参考信息 IDE 位
30:29	RSV	-	-	保留
28:0	REF_ID	R/W	0	参考信息 ID 值

35.4.19. 时间控制寄存器(FDCAN_TRIGCFG: D4h)

位域	名称	属性	复位值	描述
15:12	TEW	R/W	0	发送使能窗口值
11	RSV	-	-	保留

10:8	TTYPE	R/W	0	触发类型 000：立即触发 001：时间触发 010：单槽发送触发 011：发送开始触发 100：发送结束触发
7:6	RSV	-	-	保留
5:0	TTPTR	R/W	0	发送触发 TB 槽位置

35.4.20. 时间触发寄存器(FDCAN_TTTRIG: D8h)

位域	名称	属性	复位值	描述
15:0	TT_TRIG	R/W	0	触发时间值

35.4.21. Watch 触发寄存器(FDCAN_WTRIG: DCh)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	TT_WTRIG	R/W	0	看门狗触发时间

36. USBPHY 控制器（USBPHYC）

36.1. 概述

USBPHYC 支持对高速 USB PHY 的配置和状态的观察。

PHY 中有一个 6 引脚寄存器配置接口，用于访问一组控制寄存器，这些寄存器用于控制 PHY 的运行，也可用于调试。

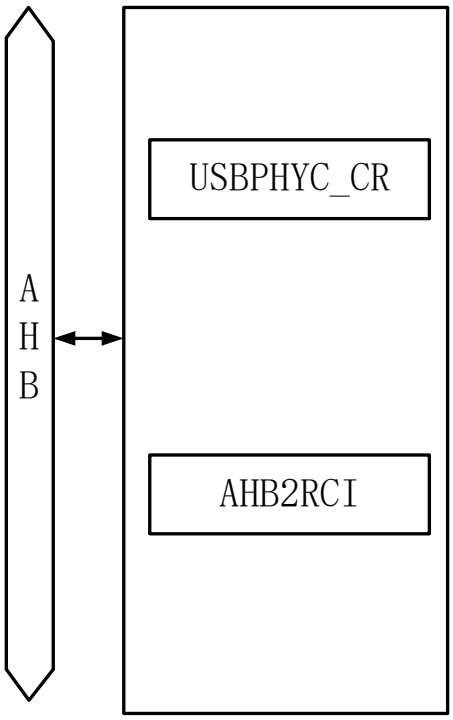
某些功能可根据客户要求通过寄存器配置接口进行调整。

由于 PHY 在默认设置下可以正常工作，因此无需设置其他值。

36.2. 主要特性

- 支持对高速 USB PHY 的配置；
- 支持对 PHY 状态的观察；

36.3. 结构框图



36.4. USBPHYC 寄存器描述

36.4.1. 寄存器列表

USB1_OTG_PHY_CFG 寄存器基地址：0x400C0000

USB2_OTG_PHY_CFG 寄存器基地址：0x400C0400

偏移	名称	复位值	描述
0x00	USBPHYC_TR0	0x00000080	USBPHYC 调试寄存器 0

0x04	USBPHYC_TR1	0x000000f8	USBPHYC 调试寄存器 1
0x08	USBPHYC_TR2	0x00000047	USBPHYC 调试寄存器 2
0x0C	USBPHYC_TR3	0x00000001	USBPHYC 调试寄存器 3
0x10	USBPHYC_TR4	0x00000000	USBPHYC 调试寄存器 4
0x14	USBPHYC_TR5	0x00000001	USBPHYC 调试寄存器 5
0x18	USBPHYC_TR6	0x00000000	USBPHYC 调试寄存器 6
0x1C	USBPHYC_TR7	0x00000000	USBPHYC 调试寄存器 7
0x28	USBPHYC_TR8	0x000000f0	USBPHYC 调试寄存器 8
0x2c	USBPHYC_TR9	0x00000000	USBPHYC 调试寄存器 9
0x80	USBPHYC _CR	0x00000010	USBPHYC 控制寄存器

36.4.2. 调试寄存器 0(USBPHYC_TR0: 00h)

位域	名称	属性	复位值	描述
31 :8	RSV	-	-	保留
7:5	ADJ_RES_HS	RW	0x4	微调 45 ohm 终端电阻 (HS) 00X：40 ohm 01X：44 ohm 10X：45 ohm 11X：50 ohm
4	RSV	-	-	保留
3	FS_SE_PD	RW	0x0	PHY 模拟输入掉电信号。 当 FS_SE_PD =1.PHY 不支持恢复和所有 VBUS 检测，ID 检测无效
2	RSV	-	-	保留
1:0	CLK_MODE	RW	0x0	RX 的时钟门控信号 00：低功耗模式 01：最低功耗模式 1X：正常功耗模式

36.4.3. 调试寄存器 1(USBPHYC_TR1: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:4	ADJ_PW_HS	RW	0xf	保留

3:0	ADJ_VREF_SQ	RW	0x8	静噪检测阈值电压控制位。 Chirp mode 步长：5~8mV； Work mode 步长：5~8mV ADJ_VREF_SQ：Chirp mode：Work mode 0000 ：155mV ：70mV …… 1000 ：205mV ：120mV …… 1111 ：245mV ：160mV

36.4.4. 调试寄存器 2(USBPHYC_TR2: 08h)

位域	名称	属性	复位值	描述
31: 7	RSV	-	-	保留
6:4	ADJ_VSW_HS	RW	0x4	输出眼形调整控制位。步长：20mV ADJ_VSW_HS：Eye shape 000 ：320mV …… 100 ：400mV …… 111 ：460mV
3:0	ADJ_IREF_RES	RW	0x7	内部偏置电流调整控制位。 步长：3.2uA ADJ_IREF_RES：Bias Current 0001 ：125uA …… 0111 ：100uA …… 1111 ：78uA

36.4.5. 调试寄存器 3(USBPHYC_TR3: 0Ch)

位域	名称	属性	复位值	描述
31: 7	RSV	-	-	保留

6:4	ADJ_TXCLK_PHASE	RW	0x0	TX 时钟相位选择，（MCLK 为 PH2） 000：PH2 001：PH0 01X：PH1 100：~PH2 101：~PH0 11X：~PH1
3:0	ADJ_PLL	RW	0x1	PLL 调整信号 ADJ_PLL [3:2]：FVCO 00：Low 01：Normal 10：High 11：Higher ADJ_PLL [1:0]: KVCO 00：Low 01：Normal 10：High 11：Higher

36.4.6. 控制寄存器(USBPHYC _CR: 80h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	REFCLK_EN	R	0x0	输入时钟使能。 当“REFCLK_EN”为“LOW”时，表示 USB PHY 不需要输入参考时钟。 REFCLK_EN 由 SUSPENDM 和 PLL_EN 控制。 SUSPENDM：PLL_EN：REFCLK_EN 0 ：0 ：0 0 ：1 ：1 1 ：X ：1
6	TEST_BIST	R	0x0	仅在 TEST=1 时有效； 置位时表示 PHY 功能正常。
5	PLL_EN	RW	0x0	使能内部 PLL
4	REFCLK_MODE	RW	0x1	选择参考时钟输入 0：输入时钟 25M 1：输入时钟 12M

3	TEST	RW	0x0	在测试模式下启动 PHY 自检，在正常操作中必须拉低
2:1	RSV	-	-	保留
0	OTG_SUSPENDM	RW	0x0	OTG 功能启用

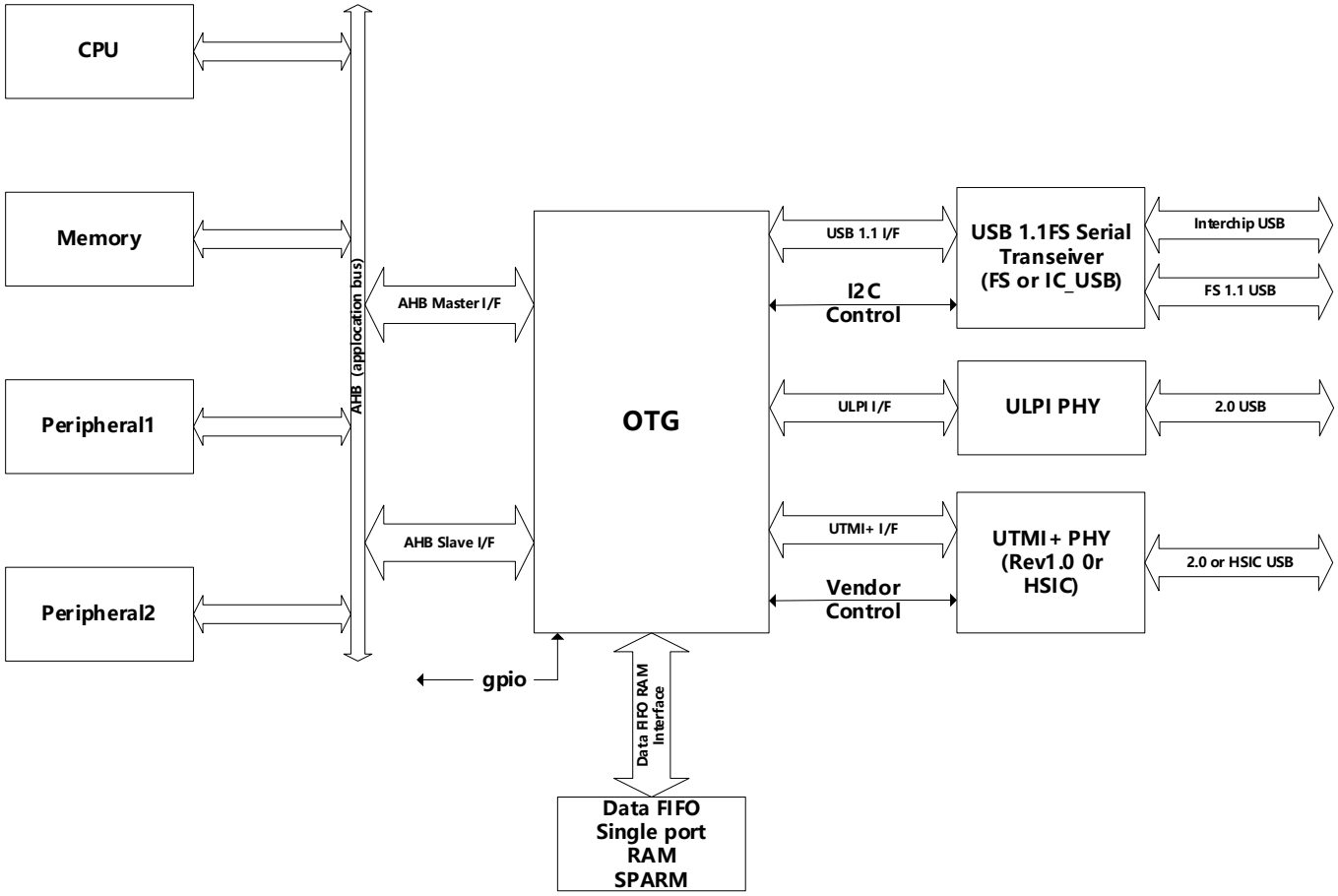
37. 高速 USB OTG (OTG_HS)

37.1. 概述

OTG_HS 是一种双角色设备（dual-role device，DRD）控制器，同时支持从机和主机功能，完全符合 USB 2.0 规范的 On-The-Go 补充标准。可以将其配置为仅主机或仅从机控制器，完全符合 USB 2.0 规范。主机模式下，支持高速（HS，480Mb/s）传输；从机模式下，支持高速（HS，480Mb/s）、低速（LS，1.5Mb/s）传输。OTG_HS 同时支持主机协商协议（HNP）和会话请求协议（SRP）。OTG 模式下唯一需要的外部器件是提供 V_{BUS} 的电荷泵。

37.2. 系统级框图

图 37-1USB OTG 系统框图



37.3. 主要特性

主要特性可分为三类：通用特性、主机模式特性和从机模式特性。

37.3.1. 通用特性

OTG_HS 接口的通用特性如下：

- 经 USB_IF 认证符合通用串行总线 2.0 版本规范
- 软件可配置为 OTGv1.3 和 OTGv2.0 工作模式
- OTG HS 支持以下 PHY 接口：

- 片上高速 PHY（UTMI 8/16）
- 片外高速 PHY 的 ULPI 接口

- 模块内嵌的 PHY 完全支持定义在 USB2.0 规范 OTG 补充标准 2.0 版中的 OTG 协议
  - 支持 A-B 器件识别（ID 线）
  - 支持主机协商协议（HNP）和会话请求协议（SRP）
  - 允许主机关闭 V_{BUS} 以在 OTG 应用中节省电池电量
  - 支持通过内部比较器对 VBUS 电平采取 OTG 监控
  - 支持主机到从机的角色动态切换
- 可通过软件配置为以下角色：
  - 具有 SRP 功能的 USB HS 从机（B 器件）
  - 具有 SRP 功能的 USB HS/LS 主机（A 器件）
  - USB On-The-Go 全速双角色设备
- 支持 HS SOF 和 LS Keep-alive 令牌
  - SOF 脉冲可通过 PAD 输出
  - SOF 脉冲通过内部连接到定时器（TIMx）
  - 可配置的帧周期
  - 可配置的帧结束中断
- OTG_HS 内嵌 DMA，可软件配置 AHB 的突发（burst）类型。
- 具有省电功能，例如在 USB 挂起期间停止系统时钟、关闭数字模块时钟、对 PHY 和 DFIFO 电源加以管理。
- 具有 8 KB 专用 RAM，采用先进的 FIFO 管理：
  - 可将 RAM 空间划分为不同 FIFO，以便灵活高效地使用 RAM
  - 每个 FIFO 可存储多个数据包
  - 动态分配存储区
  - FIFO 大小可配置为非 2 的幂次方值，以便连续使用存储单元
- 一帧之内可以无需要应用程序干预，以达到最大 USB 带宽。
- 它支持电池充电规范第 1.2 版中介绍的充电端口检测（仅限 FS PHY 收发器）

37.3.2. 主机模式特性

OTG_HS 接口在主机模式下具有以下特性：

- 通过外部电荷泵生成 VBUS 电压。
- 多达 16 个主机通道（又称之为管道）：每个通道都可以动态实现重新配置，可支持任何类型的 USB 传输。
- 内置硬件调度器可：
  - 在周期性硬件队列中存储多达 16 个中断加同步传输请求
  - 在非周期性硬件队列中存储多达 16 个控制加批量传输请求
  - 管理一个共享 RxFIFO、一个周期性传输 TxFIFO 和一个非周期性传输 TxFIFO，以高效使用 USB 数据 RAM。

37.3.3. 从机模式特性

OTG_HS 接口在从机模式下具有以下特性：

- 1 个双向控制端点 0
- 15 个 IN 端点，可配置为支持批量传输、中断传输或同步传输
- 15 个 OUT 端点，可配置为支持批量传输、中断传输或同步传输
- 管理一个共享 Rx FIFO 和一个 Tx-OUT FIFO，以高效使用 USB 数据 RAM
- 管理多达 16 个专用 Tx-IN FIFO（每个配置为 IN 的端点一个），以降低应用程序负荷
- 支持软断开功能。

37.4. OTG_HS 功能说明

37.4.1. USB OTG 引脚信号

信号名称	信号类型	说明
OTG_HS/FS_DP	输入/输出	USB OTGD+数据线
OTG_HS/FS_DM	输入/输出	USB OTGD-数据线
OTG_HS/FS_ID	输入	USB OTG ID
OTG_HS/FS_VBUS	双向	USB OTG 总线电源
OTG_HS_SOF	输入	USB OTG 帧起始
OTG_HS_ULPI_CK	输入	USB OTG ULPI 时钟
OTG_HS_ULPI_DIR	输入	USB OTG ULPI 数据总线方向控制
OTG_HS_ULPI_STP	输出	USB OTG ULPI 数据流停止
OTG_HS_ULPI_NXT	输入	USB OTG ULPI 下一数据流请求
OTG_HS_ULPI_D0~7	输入/输出	USB OTG ULPI 8 位双向数据总线

37.4.2. OTG 模块

USB OTG 通过复位和时钟控制模块接收 48 MHz 的时钟。USB 时钟用于在全速通信时驱动 48 MHz 时钟域，必须在配置 OTG 模块前使能。

CPU 通过 AHB 外设总线对 OTG 模块寄存器进行读写操作，通过“OTG_HS 中断”章节中所述的 USB OTG 中断线接收 USB 事件通知。

CPU 通过向特定的 OTG 单元（压栈寄存器）写入 32 位字来向 USB 提交数据。数据随即自动存储到 USB 数据 RAM 中配置的数据发送 FIFO 中。每个 IN 端点（从机模式）或 OUT 通道（主机模式）都有一个 Tx FIFO 压栈寄存器。

CPU 从特定的 OTG 地址（出栈寄存器）读取 32 位字，以读取来自 USB 的数据。数据随即从 8KB USB 数据 RAM 内配置的共享 Rx FIFO 中弹出。每个 OUT 端点（从机模式）或 IN 通道（主机模式）都有一个 Rx FIFO 出栈寄存器。

USB 协议层通过串行接口引擎（SIE）驱动，并通过片上物理层（PHY）中的收发器模块进行数据的串行通信，

也可以通过外部 HS PHY 进行数据的串行通信。

37.4.3. 高速 OTG PHY

USB OTG_HS 内核包含一个片上 HS PHY 和一个 ULPI 接口以连接外部 HS PHY。

37.5. OTG 双角色设备（DRD)

37.5.1. ID 线检测

采取主机还是从机（默认设置）角色取决于 ID 输入引脚的电平。插入 USB 电缆时可根据是 MicroA 还是 MicroB 插头连接到 micro-AB 插座来确定 ID 线状态。

- 如果 USB 电缆的 B 端连入，其 ID 线悬空，则由于设备在 ID 线上的集成上拉电阻，设备将检测到 ID 高电平并确认采取默认的从机角色。在此配置中，OTG_HS 符合 USB2.0 On-The-Go 2.0 版补充标准中第 4.2.4 节 ID 引脚中所述的 FSM 标准。
- 如果 USB 电缆的 A 端连入，其 ID 线接地，则 OTG_FS/OTG_HS 将发出 ID 线状态更改中断（GINTSTS 寄存器中的 ConIDStsChng 位）以告知应用程序初始化主机，并自动切换为主机角色。在此配置中， OTG_HS 符合 USB2.0 On-The-Go 2.0 版补充标准中第 4.2.4 节 ID 引脚中所述的 FSM 标准。

37.5.2. HNP 双角色设备

全局 USB 配置寄存器中的 HNP 使能位（GUSBCFG 寄存器中的 HNPCap 位）可使 OTG_HS 模块根据主机协商协议（HNP）动态切换角色，例如从 A 主机切换为 A 器件（反之亦然），或者从 B 器件切换为 B 主机（反之亦然）。通过全局 OTG 控制和状态寄存器中的连接器 ID 状态位（GOTGCTL 寄存器中的 ConIDSts 位），以及全局中断和状态寄存器中的当前工作模式位（GINTSTS 中的 CurMod 位）的组合值，可定义设备当前状态。

37.5.3. SRP 双角色设备

全局 USB 配置寄存器中的 SRP 使能位（GUSBCFG 中的 SRPCap 位）可使 OTG_HS 模块关闭 V_{BUS} 供电，为 A 器件节省电能。注意，无论 OTG_HS 采取主机角色还是从机角色，A 器件将始终负责 V_{BUS} 的提供。

37.6. USB 从机

本节介绍了 OTG_HS 在 USB 设备模式下所具有的功能。在以下情形下，OTG_HS 用作 USB 从机：

- OTG B 设备
  - OTG B 器件插入 USB 电缆 B 端时的默认状态
- OTG A 设备
  - HNP 将 OTG_HS 切换到从机角色后的 OTG A 器件状态
- B 设备
  - 如果 ID 线存在、功能正常被连接到 USB 电缆的 B 端，并且全局 USB 配置寄存器中的 HNP 功能位（GUSBCFG 中的 HNPCap 位）清零。
- 仅从机
  - 将全局 USB 配置寄存器（GUSBCFG）中的强制从机模式位（ForceDevtMode ）置 1，强制 OTG_HS 内核工作在仅从机模式。这种情况下，即使 USB 连接器上的 ID 线可用，也会忽略 ID 线。

注： 要在 B 设备或仅从机配置情形下构建总线供电的设备方案，需要添加一个外部调压器，用于从 V_{BUS} 生成所需电源。

37.6.1. 支持 SRP 功能的 USB 设备

全局 USB 配置寄存器中的 SRP 功能位（GUSBCFG 中的 SRPCap 位）可使 OTG_HS 支持会话请求协议(SRP)。这样一来，远程 A 器件便可以在 USB 会话挂起时，通过关闭 V_{BUS} 来节省电能。

37.6.2. USB 从机状态

● 供电状态

VBUS 输入检测到 B 会话有效电压，就会使 USB 设备进入供电状态（请参见 USB2.0 第 9.1 部分）。然后， OTG_FS/OTG_HS 自动连接 DP 上拉电阻，发出全速设备与主机相连的信号并生成会话请求中断（ GINTSTS 中的 SessReqInt 位），指示进入供电状态。

此外， VBUS 输入还可确保主机在 USB 操作期间提供有效的 VBUS 电平。如果检测到 VBUS 降至 B 会话有效电压以下（例如，因电源干扰或主机端口关闭引发）， OTG_HS 将自动断开连接并生成检测到会话结束中断（ GOTGINT 中的 SesEndDet 位），指示 OTG_HS 已退出供电状态。

供电状态下， OTG_HS 期望收到来自主机的复位信号。其它 USB 操作则无法执行。收到复位信号后，立即生成检测到复位中断（ GINTSTS 中的 USBRst）。复位信号结束后，将生成枚举完成中断（ GINTSTS 中的 EnumDone 位）， OTG_HS 随即进入默认状态。

● 软断开

供电状态可借助软断开功能通过软件退出。将设备控制寄存器中的软断开位（DCTL 中的 SftDiscon 位）置 1 即可移除 DP 上拉电阻，此时尽管没有从主机端口实际拔出 USB 电缆，但主机端仍会发生设备断开检测中断。

● 默认状态

默认状态下， OTG_HS 期望从主机收到 SET_ADDRESS 命令。其它 USB 操作则无法执行。当 USB 上解码出有效 SET_ADDRESS 命令时，应用程序会将相应的数值写入设备配置寄存器中的设备地址字段（DCFG 中的 DevAddr 位）。 OTG_HS 随即进入地址状态，并准备好以所配置的 USB 地址对主机事务进行应答。

● 挂起状态

OTG_HS 设备持续监视 USB 活动。在 USB 空闲时间达到 3 ms 后，将发出早期挂起中断（GINTSTS 中的 ErlySusp 位），并在 3 ms 后由挂起中断（ GINTSTS 中的 USBSusp 位）确认设备进入挂起状态。然后，设备状态寄存器中的设备挂起位（DSTS 中的 SuspSts 位）自动置 1， OTG_HS 随即进入挂起状态。

可通过设备本身退出挂起状态。这种情况下，应用程序会将设备控制寄存器中的远程唤醒信号位（ DCTL 中的 RmtWkUpSig 位）置 1，并在 1 ms 到 15 ms 后将其清零。

但若设备检测到主机发出的恢复信号时，将生成恢复中断（ GINTSTS 中的 WkUpInt 位），设备挂起位自动清零。

37.6.3. USB 设备端点

OTG_HS 模块实现了以下 USB 端点：

● 控制端点 0：

- 双向且仅处理控制消息
- 使用一组单独的寄存器来处理 IN 和 OUT 事务
- 专用控制（DIEPCTL0/DOEPCTL0）寄存器、传输配置（DIEPTSIZ0/DOEPTSIZ0）寄存器和状态中断(DIEPINT0/DOEPINT0）寄存器。控制和传输大小寄存器中可用的位组与其它端点中稍有不同

● 16 个 IN 端点

- 每个端点都可配置为支持同步传输、批量传输或中断传输类型

- 每个端点都有专用控制（DIEPCTLx）寄存器、传输配置（DIEPTSIZx）寄存器和状态中断（DIEPINTx）寄存器
- 设备 IN 端点通用中断屏蔽寄存器（DIEPMSK）可用于使能/禁止所有 IN 端点（包括 EP0）上的同一类端点中断源。
- 支持未完成的同步 IN 传输中断（ GINTSTS 中的 incompISOIN 位），该中断将在当前帧中至少有一个同步 IN 端点上的传输未完成时触发。该中断和周期帧结束中断（GINTSTS/EOPF）一起触发。

● 16 个 OUT 端点

- 每个端点都可配置为支持同步传输、批量传输或中断传输类型
- 每个端点都有专用控制（DOEPCTLx）寄存器、传输配置（DOEPTSIZx）寄存器和状态中断（DOEPINTx）寄存器
- 设备 OUT 端点通用中断屏蔽寄存器（DOEPMSK）可用于使能/禁止所有 OUT 端点（包括 EP0）上的同一类端点中断源
- 支持未完成的同步 OUT 传输中断（GINTSTS 中的 incompISOOUT 位），该中断将在当前帧中至少有一个同步 OUT 端点上的传输未完成时触发。该中断和周期帧结束中断（GINTSTS/EOPF）一起触发。

● 端点控制

应用程序可通过设备端点 x IN/OUT 控制寄存器（DIEPCTLx/DOEPCTLx）对端点采取以下控制：

- 端点使能/禁止
- 在当前配置下激活端点
- 设置 USB 传输类型（同步、批量和中断）
- 设置支持的数据包大小
- 设置与 IN 端点相关的 Tx FIFO 编号
- 设置希望收到的或发送时要使用到的 data0/data1 PID（仅限批量/中断传输）
- 设置接收或发送事务时所对应的奇/偶帧（仅限同步传输）
- 可以设置 NAK 位，从而不论此时 FIFO 的状态如何，都对主机的请求回复 NAK
- 可以设置 STALL 位，使得主机对该端点的令牌都被硬件回复 STALL
- 可以将 OUT 端点设置为侦听模式，即对接收到的数据不进行 CRC 检查

● 端点传输

设备端点 x 传输大小寄存器（DIEPTSIZx/DOEPTSIZx）允许应用程序对传输大小参数进行编程并读取传输状态。必须在端点控制寄存器中的端点使能位置 1 之前完成对此寄存器的设置。使能端点后，这些字段立即变为只读状态，同时 OTG_HS 模块根据当前传输状态对这些字段进行更新。可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输的数据包个数

● 端点状态/中断

设备端点 x 中断寄存器（DIEPINTx/DOPEPINTx）指示端点在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的 OUT 端点中断位或 IN 端点中断位（分别为 GINTSTS 中的 OEPInt 位或 GINTSTS 中的 IEPInt 位）置 1 时，应用程序必须读取这些寄存器以获得详细信息。在应用程序读取这些寄存器之前，必须先读取设备全体端点中断（DAINT）寄存器，以获取设备端点 x 中断寄存器的端点编号。应用程序必须将此寄存器中的相应位清零，才能将 DAINT 和 GINTSTS 寄存器中的相应位清零。

模块提供以下状态检查和中断产生功能：

- 传输完成中断，指示应用程序（AHB）和 USB 端均已完成数据传输

- Setup 阶段已完成（仅针对控制传输类型的 OUT 端点）
- 相关的发送 FIFO 为半空或全空状态（ IN 端点）
- NAK 应答已发送到主机（仅针对同步传输的 IN 端点）
- Tx FIFO 为空时接收到 IN 令牌（仅针对批量和中断传输类型的 IN 端点）
- 尚未使能端点时接收到 OUT 令牌
- 检测到串扰错误
- 应用程序关闭端点生效
- 应用程序对端点设置 NAK 生效（仅针对同步传输类型的 IN 端点）
- 接收到 3 个以上连续 setup 数据包（仅针对控制类型的 OUT 端点）
- 检测到超时状况（仅针对控制传输类型的 IN 端点）
- 同步传输类型的数据包未产生中断而丢失

37.7. USB 主机

本节介绍了 OTG_HS 在 USB 主机模式下所具有的功能。在以下情形下， OTG_HS 用作 USB 主机：

- OTG A 主机
  - OTG A 器件在插入 USB 电缆 A 端时的默认状态
- OTG B 主机
  - OTG B 器件被 HNP 切换为主机角色后的状态
- A 主机
  - 如果 ID 线有连接，器件与 USB 电缆的 A 端相连，全局 USB 配置寄存器中的 HNP 功能位（ GUSBCFG 中的 HNPCap 位）清零。 DP/DM 线上的集成下拉电阻自动使能。
- 仅作主机
  - 1.14.4 全局 USB 配置寄存器中的强制模式位（ GUSBCFG 中的 ForceHstMode 位）将强制 OTG_HS 模块作为 USB 仅作主机运行。这种情况下，即使 USB 连接器上存在 ID 线，模块也会忽略 ID 线上的电平。 DP/DM 线上的集成下拉电阻自动使能。

注： 微控制器不能输出 5 V 以提供 VBUS。为此，必须在微控制器以外添加电荷泵或电源开关（如果应用电路板提供 5 V 电源）来驱动 5 V VBUS 线。外部电荷泵可通过任何 GPIO 输出驱动。 OTG A 主机、 A 器件和仅作主机配置都需要使用电荷泵。

37.7.1. 支持 SRP 功能的 USB 主机

全局 USB 配置寄存器中的 SRP 使能位（ GUSBCFG 中的 SRPCap 位）可提供 SRP 支持。使能 SRP 功能后，主机可在 USB 会话挂起时通过关闭 VBUS 电源来节省电能。

37.7.2. USB 主机状态

- 给主机端口供电

微控制器不能输出 5 V 以提供 VBUS。为此，必须在微控制器以外添加电荷泵或电源开关（如果应用电路板提供 5 V 电源）来驱动 5 V VBUS 线。外部电荷泵可通过任何 GPIO 输出驱动，或者通过连接至外部 PMIC（电源管理 IC）的 I2C 接口驱动。当应用程序确定控制外部器件输出 VBUS 时，还必须将主机端口控制和状态寄存器中的端口电源位（ HPRT 中的 PrtPwr 位）置 1。

- VBUS 有效

使能 HNP 或 SRP 后，应将 VBUS 感应引脚连接到 VBUS。 VBUS 输入可确保电荷泵在 USB 操作期间提供有效的 VBUS 电平。如果 VBUS 电压意外降至 VBUS 有效阈值（4.4 V）以下，将通过会话结束检测位（GOTGINT 中的 SesEndDet 位）触发 OTG 中断。之后应用必须断开 VBUS 电源并使端口电源位清零。

当 HNP 和 SRP 均禁止时，无需将 VBUS 感应引脚连接到 VBUS。

电荷泵过流标志也可用来防止电气损坏。将电荷泵的过流标志输出连接到任意 GPIO 输入，然后将其配置为出现有效电平时生成端口中断。过流 ISR 必须立即关闭 VBUS 并清零端口电源位。

● 主机检测设备连接

如果使能 SRP 或 HNP，即使可以随时连接 USB 外设或 B 器件，但是 OTG_HS 也只有在 VBUS 有效后（ 5V）才能检测到设备的连接。当 VBUS 处于有效电平且已连接远程 B 器件时，OTG_HS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（HPRT 中的 PrtConnDet 位）触发。

在 HNP 和 SRP 同时关闭的情况下， USB 设备或 B 器件将在连接后立即被检测到。 OTG_HS 模块将发出主机端口中断信号，该中断由主机端口控制和状态寄存器中的设备连接位（ OTG_HPRT 中的 PrtConnDet 位）触发。

● 主机检测设备断开

设备断开事件将触发断开连接检测中断（GINTSTS 中的 DisconnInt 位）。

● 主机枚举

检测到设备连接后，若又有新的设备连接进来，主机必须通过向新的设备发送 USB 复位和配置命令来启动枚举过程。

开始驱动 USB 复位前，应用程序必须等待去抖动完成位（ GOTGINT 中的 DbnceDone 位）触发 OTG 中断，这表示由于在 DP (FS) 或 DM (LS) 上连接上拉电阻而发生电气抖动之后，总线恢复稳定状态。

应用程序通过将主机端口控制和状态寄存器中的端口复位位（HPRT 中的 PrtRst 位）置 1，并保持最少 10 ms，最多 20 ms，来在 USB 总线上发出 USB 复位信号（单端零）。应用程序计算这个过程的持续时间，然后将端口复位位清零。

USB 复位序列完成后，端口使能/禁止更改位（HPRT 中的 PrtEnChng 位）立即触发主机端口中断，进而向应用程序发出通知，指示可从主机端口控制和状态寄存器中的端口速度字段（ HPRT 中的 PrtSpd）读取枚举的设备速度，以及主机已经开始驱动 SOF (FS）或 Keep-alive 令牌（LS)。此时主机已就绪，可通过对设备发送命令来完成对设备的枚举。

● 主机挂起

应用程序通过将主机端口控制和状态寄存器中的端口挂起位（HPRT 中的 PrtSusp）置 1 来挂起 USB 活动。OTG_HS 模块停止发送 SOF 并进入挂起状态。

可由远程设备的自主活动（远程唤醒）使总线退出挂起状态。这种情况下，远程唤醒信号将触发远程唤醒中断（ GINTSTS 中的 WkUpInt 位），硬件把主机端口控制和状态寄存器中的端口恢复位（ HPRT 中的 PrtRes 位）自动置位，并通过 USB 自动驱动恢复信号。应用程序必须为恢复窗口定时，然后将端口恢复位清零以退出挂起状态并重新发送 SOF。

如果由主机发起退出挂起状态，则应用程序必须将端口恢复位置 1 以启动主机端口上的恢复信号，为恢复窗口定时并最终将端口恢复位清零。

37.7.3. 主机通道

OTG_HS 内核实现了 16 主机通道。每个主机通道均可用于 USB 主机传输（ USB 管道)。主机最多能同时处理 16 个传输请求。如果应用程序有 8 个以上的传输请求挂起，则在通道从之前任务释放后（即，接收到传输完成和通道停止中断后），主机控制器驱动器 (HCD) 必须为未处理的传输请求重新对通道进行分配。

每个主机通道都可配置为支持输入/输出以及周期性/非周期性事务。每个主机通道都使用专用控制 (HCCHARx) 寄存器、传输配置 (HCTSIZx) 寄存器状态中断 (HCINTx) 寄存器以及和其中断屏蔽寄存器 (HCINTMSKx)。

● 主机通道控制

应用程序可通过主机通道 x 特性寄存器（HCCHARx）对主机通道作以下控制：

- 通道使能/禁止
- 设置目标 USB 设备的 HS/FS/LS 速度
- 设置目标 USB 设备的地址
- 设置与该通道通信的目标 USB 设备上的端点的编号
- 设置该通道上的传输方向： IN/OUT
- 设置该通道上的 USB 传输的类型：控制/批量/中断/同步
- 设置与该通道通信的设备端点的最大包长
- 设置要进行周期传输的帧：奇帧/偶帧

● 主机通道传输

主机通道传输大小寄存器（HCTSIZx）允许应用程序对传输大小参数进行编程并读取传输状态。必须在主机通道特性寄存器中的通道使能位置 1 之前完成对此寄存器的设置。使能端点后，数据包计数字段立即变为只读状态，同时 OTG_HS 模块根据当前传输状态对该字段进行更新。

可对以下传输参数进行编程：

- 以字节为单位的传输大小
- 构成整个传输大小的数据包个数
- 初始数据 PID

● 主机通道状态/中断

主机通道 x 中断寄存器（HCINTx）指示通道在出现 USB 和 AHB 相关事件时的状态。当模块中断寄存器中的主机通道中断位（GINTSTS 中的 HCINT 位）置 1 时，应用程序必须读取这些寄存器以获得详细信息。在读取这些寄存器之前，应用程序必须先读取主机全体通道中断（HAINT）寄存器，以获取主机通道 x 中断寄存器的通道编号。应用程序必须将该寄存器中的相应位清零，才能将 HAINT 和 GINTSTS 寄存器中的对应位清零。HCINTMSKx 寄存器还提供每个通道各中断源的屏蔽位。

主机模块提供以下状态检查和中断产生功能：

- 传输完成中断，指示应用程序（AHB）和 USB 端均已完成数据传输
- 通道因传输完成、 USB 事务错误或应用程序发出禁止命令而停止
- 相关的发送 FIFO 为半空或全空状态（ IN 端点）
- 接收到 ACK 响应
- 接收到 NAK 响应
- 接收到 STALL 响应
- 由于 CRC 校验失败、超时、位填充错误和错误的 EOP 导致 USB 事务错误
- 串扰错误
- 帧上溢
- 数据同步错误

37.7.4. 主机调度器

主机模块内置硬件调度器，可自主对应用程序发出的 USB 事务请求重新排序和管理。每一帧开始时，主机都先执行周期性（同步和中断）事务，然后执行非周期性（控制和批量）事务，以符合 USB 规范对同步和中断

传输高优先级的保证。

主机通过请求队列（一个周期性请求队列和一个非周期请求队列）处理 USB 事务。每个请求队列最多可存储 8 个条目。每个条目代表一个应用程序发起但还未得到响应的 USB 事务请求，并存储了执行该 USB 事务所用到的 IN 或 OUT 通道的编号，以及其它相关信息。USB 事务请求在队列中的写入顺序决定了事务在 USB 接口上的执行顺序。

每一帧开始时，主机都先处理周期性请求队列，然后处理非周期性请求队列。如果当前帧结束时，计划在当前帧执行的同步或中断类型的 USB 传输事务请求仍处于挂起状态，则主机将发出未完成周期性传输中断（GINTSTS 中的 incomplP 位）。OTG_FS/OTG_HS 模块全面负责对周期性和非周期性请求队列的管理。周期性发送 FIFO 和队列状态寄存器（HPTXSTS ）与非周期性发送 FIFO 和队列状态寄存器（GNPTXSTS ）都为只读寄存器，应用程序可使用它们来读取各请求队列的状态。其中包括：

- 周期性（非周期性）请求队列中当前可用的空闲条目数（最多 8 个）
- 周期性（非周期性） Tx FIFO （ OUT 事务）中当前可用的空闲空间
- IN/OUT 令牌、主机通道编号和其它状态信息

由于每个请求队列最多可存储 8 个 USB 事务请求，因此应用程序可以把主机 USB 事务请求提前发送给调度器；实际的通信最晚会在调度器处理完已挂起的 8 个周期事务和 8 个非周期事务完成之后出现在 USB 总线上。

要向主机调度器（队列）发出事务请求，应用程序必须读取 HNPTXSTS 寄存器中的 PTxQSpcAvail 位或 GNPTXSTS 寄存器中的 NPTxQSpcAvail 位，确保周期性（非周期性）请求队列中至少有一个可用空间来存储当前请求。

37.8. SOF 触发

表 37-1 SOF 触发输出与 TIM ITR 的连接

	TIM2	TIM23	TIM5	TIM24
USB1	ITR5	ITR5	ITR7	ITR7
USB2	ITR6	ITR6	ITR8	ITR8

OTG_HS 模块在主机和设备模式下都可以监视、跟踪和配置 SOF 帧并且还具备 SOF 脉冲输出连接功能。

此功能尤其适用于自适应音频时钟生成，其中音频设备需要与 PC 提供的同步音频数据流实现同步，或者主机需要根据音频设备的要求调整数据帧率。

37.8.1. 主机 SOF

主机模式下，可以在主机帧间隔寄存器（HFIR）中对所产生的两个连续 SOF (HS/FS) 或 Keep-alive (LS) 令牌期间所出现的 PHY 时钟数进行编程，进而应用程序可对 SOF 帧周期进行控制。帧开始（GINTSTS 中的 SOF 位）时都将生成中断。当前帧编号和出现下一个 SOF 前剩余的时间应用程序在主机帧编号寄存器 (HFNUM) 中能够进行跟踪。

SOF 令牌发出的同时会产生 SOF 脉冲信号，并且宽度为 12 个系统时钟周期。此外， SOF 脉冲信号还在内部与定时器的输入触发相连，因此可通过 SOF 脉冲触发输入捕获功能、输出比较功能和定时器。

37.8.2. 设备 SOF

在设备模式下，USB 每次接收到 SOF 令牌时，都将触发帧开始中断（GINTSTS 中的 SOF 位）。相应的帧编号可从设备状态寄存器（DSTS 中的 FNSOF 位）读取。还可以生成宽度为 12 个系统时钟周期的 SOF 脉冲信号。此外，SOF 脉冲信号还在内部与 TIM 的输入触发相连，因此可通过 SOF 脉冲触发输入捕获功能、输出比

较功能和定时器。

周期性帧结束中断（GINTSTS/EOPF）用于在经过了 80%、85%、90%或 95%的帧间隔时间时通知应用程序，具体取决于设备配置寄存器中的周期性帧间隔字段（DCFG 中的 PerFrInt 位）。此功能可用于确定该帧的所有同步通信是否完成。

37.9. 电源选项

OTG PHY 的功耗由通用模块配置寄存器中的两个或三个位控制，具体取决于 OTG 支持的版本。

- PHY 掉电 (OTG_GCCFG/PWRDWN)

用于开启/关闭 PHY 的全速收发器模块。先置位后才允许后续的 USB 操作。

- VBUS 检测使能 (OTG_GCCFG/VBDEN)

用于开启/关闭与 OTG 操作关联的 VBUS 感应比较器。

USB 会话没有开始或设备未连接时，可以在 USB 挂起状态下使用功率降低技术。

- 停止 PHY 时钟（ OTG_PCGCCTL 中的 STPPCLK 位)

将时钟门控控制寄存器中的停止 PHY 时钟位置 1 时， OTG 全速模块的大多数 48 MHz 内部时钟域均由时钟门控关闭。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗

还会关掉收发器的大部分单元，只有负责检测异步恢复事件或远程唤醒事件的部分还保持工作状态。

- HCLK 门控（ OTG_PCGCCTL 中的 GATEHCLK 位)

将时钟门控控制寄存器中的 Gate HCLK 位置 1 时， OTG_HS 模块内部的大多数系统时钟域均由时钟门控关闭。只有寄存器读取和写入接口保持活动状态。即使应用程序仍提供时钟输入，也会节省掉模块由于时钟信号翻转带来的动态功耗。

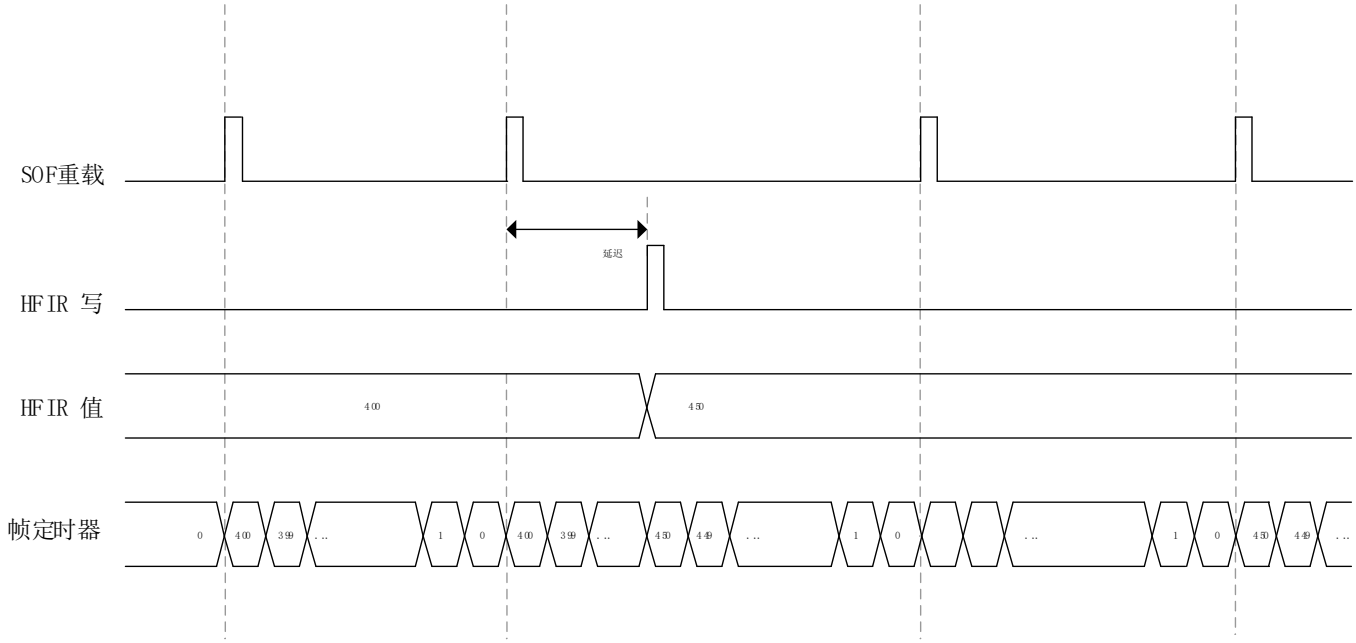
- USB 系统停止

当 OTG_HS 处于 USB 挂起状态时，应用程序可通过将 USB 系统中的所有时钟源全部关闭来显著降低总功耗。 USB 系统停止可通过以下方式激活：首先将停止 PHY 时钟位置 1，然后在电源控制系统模块（PWR）中将系统配置为深度睡眠模式。OTG_HS 模块通过对 USB 上的远程唤醒（作为主机）或恢复（作为设备）信号进行异步检测，自动重新激活系统时钟和 USB 时钟。为了节省动态功耗，只在 USB 数据 FIFO 被 OTG_HS 模块访问时为其提供时钟。

37.10. 动态更新 HFIR 寄存器

主机模式下，USB 模块具有对微帧周期进行动态微调的功能，能够将外部设备与 micro-SOF 帧进行同步。

如果 HFIR 寄存器在当前 micro-SOF 帧内发生更改，则将在下一个帧中对 SOF 周期进行相应修正，具体说明请参见下图。



37.11. USB 数据 FIFO

USB 系统具有 4KB 专用 RAM，采用复杂的 FIFO 控制机制。OTG_HS 模块中的数据包 FIFO 控制器模块将 RAM 空间划分为多个 Tx-FIFO（USB 传输前，应用程序将数据压入其中进行短暂存储）和单个 Rx FIFO（从 USB 接收到的数据被应用程序读取之前，在其中进行短暂存储）。RAM 中所构建的 FIFO 的数量与组织方式取决于设备的角色。设备模式下，为每个激活的 IN 端点配置一个 Tx FIFO。FIFO 的大小均由软件配置，以更好地满足应用要求。

37.11.1. 设备 FIFO 架构

● 设备 Rx FIFO

OTG 设备使用单个接收 FIFO 接收发送到所有 OUT 端点的数据。只要 Rx FIFO 中有空余空间，收到的数据包就挨个填入 Rx FIFO。除了有效数据外，接收到的数据包状态（包含 OUT 端点目标编号、字节数、数据 PID 和对所接收数据的验证）也由模块进行存储。没有可用空间时，设备会回复主机事务 NACK 应答并在被寻址的端点上触发中断。接收 FIFO 的大小在接收 FIFO 大小寄存器（GRXFSIZ）中配置。单个接收 FIFO 架构使得 USB 设备更高效地填充接收 RAM 缓冲区：

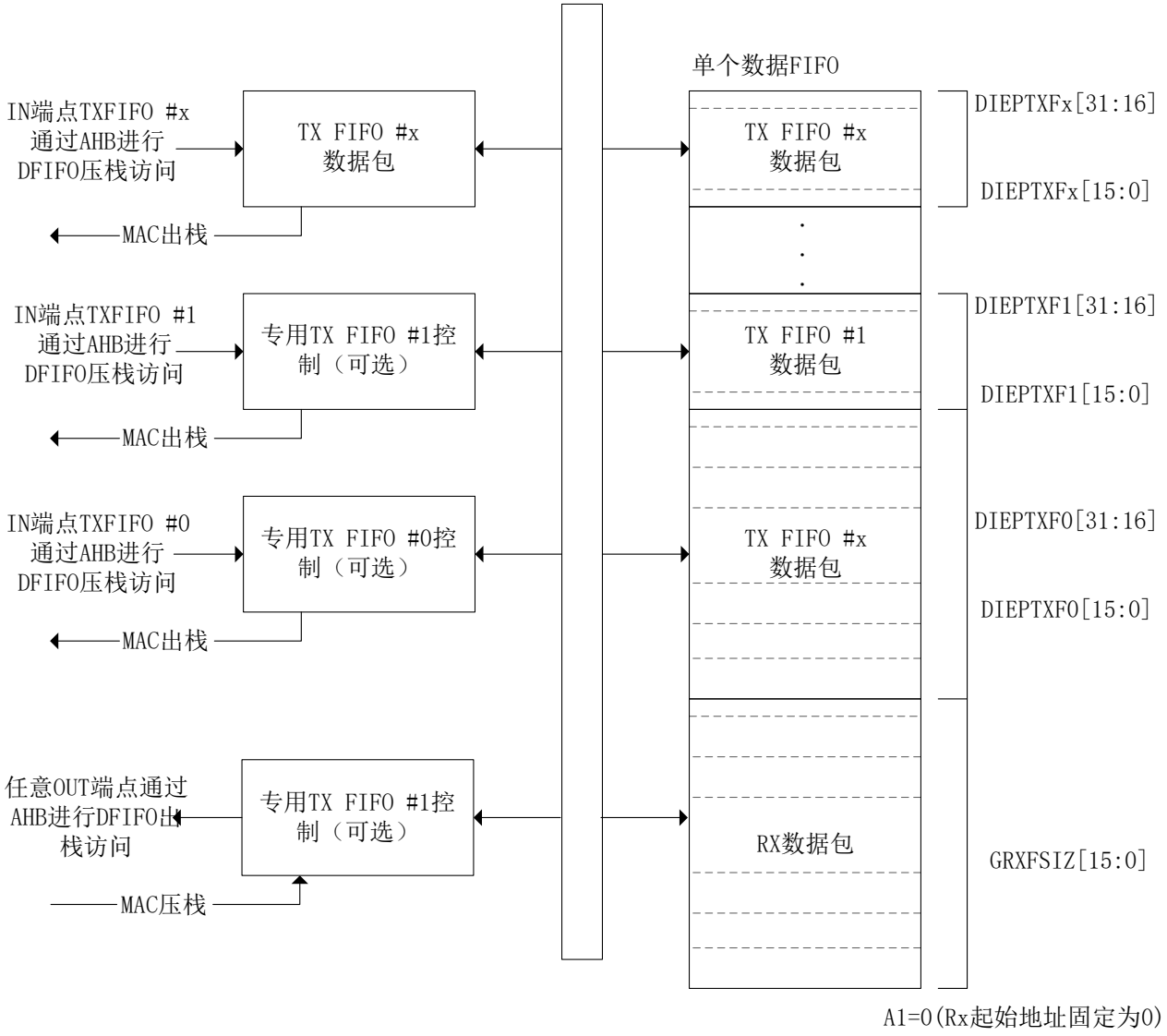
- 所有 OUT 端点共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_HS 模块可将主机发出的任何 OUT 通信序列填充到接收 FIFO，直到没有多余空闲空间

只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会一直接收 Rx FIFO 非空中断（GINTSTS 中的 RxFLvl 位）。应用程序从接收状态读取和出栈寄存器（GRXSTSP）中读取数据包信息，最后通过读取与端点相关的出栈地址从接收 FIFO 读出相应数据。

● 设备 Tx FIFO

模块为各个 IN 端点提供了专用的 FIFO。应用程序通过端点 0 发送 FIFO 大小寄存器（DIEPTXF0）为 IN 端点 0 配置 FIFO 大小；通过设备 IN 端点发送 FIFOx 寄存器（DIEPTXFx）为 IN 端点 x 配置 FIFO 大小。

下图为设备模式下的 FIFO 地址映射和 AHB FIFO 访问映射



37.11.2. 主机 FIFO 架构

● 主机 Rx FIFO

主机使用一个接收 FIFO 处理所有周期和非周期事务。 FIFO 用作接收缓冲区以保存从 USB 接收到的数据（接收到的数据包的数据部分），直至这些数据传输到系统存储器。只要 FIFO 中有空间，来自设备 IN 端点的数据包就接收进来并挨个存储。接收到的每个数据包的状态（包含主机目标通道、字节数、数据 PID 和对所接收数据的校验）也存储在 FIFO 中。接收 FIFO 的大小在接收 FIFO 大小寄存器 (GRXFSIZ) 中配置。单个接收 FIFO 架构使得 USB 主机高效地填充接收数据缓冲区：

- 所有 IN 配置主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_HS 模块可将主机发出的任何 IN 通信序列带来的接收数据填充到接收 FIFO，直到没有多余空闲空间

只要至少有一个数据包在 Rx FIFO 中可供读取，应用程序就会接收 Rx FIFO 非空中断。应用程序从接收状态读取和出栈寄存器中读取数据包信息。

● 主机 Tx FIFO

主机使用一个发送 FIFO 处理所有非周期（控制和批量） OUT 事务，使用另一个发送 FIFO 处理所有周期（同步和中断） OUT 事务。 FIFO 用作发送缓冲区以保存要通过 USB 发送的数据（发送数据包）。周期（非周期） Tx FIFO 的大小在主机周期（非周期）发送 FIFO 大小（HPTXFSIZ/GNPTXFSIZ）寄存器中配置。

两个 Tx FIFO 按优先级实施操作，周期性通信的优先级较高，因此在 USB 一帧的时间内首先进行周期性通信。帧起始时，内置的主机调度器先处理周期请求队列，再处理非周期请求队列。

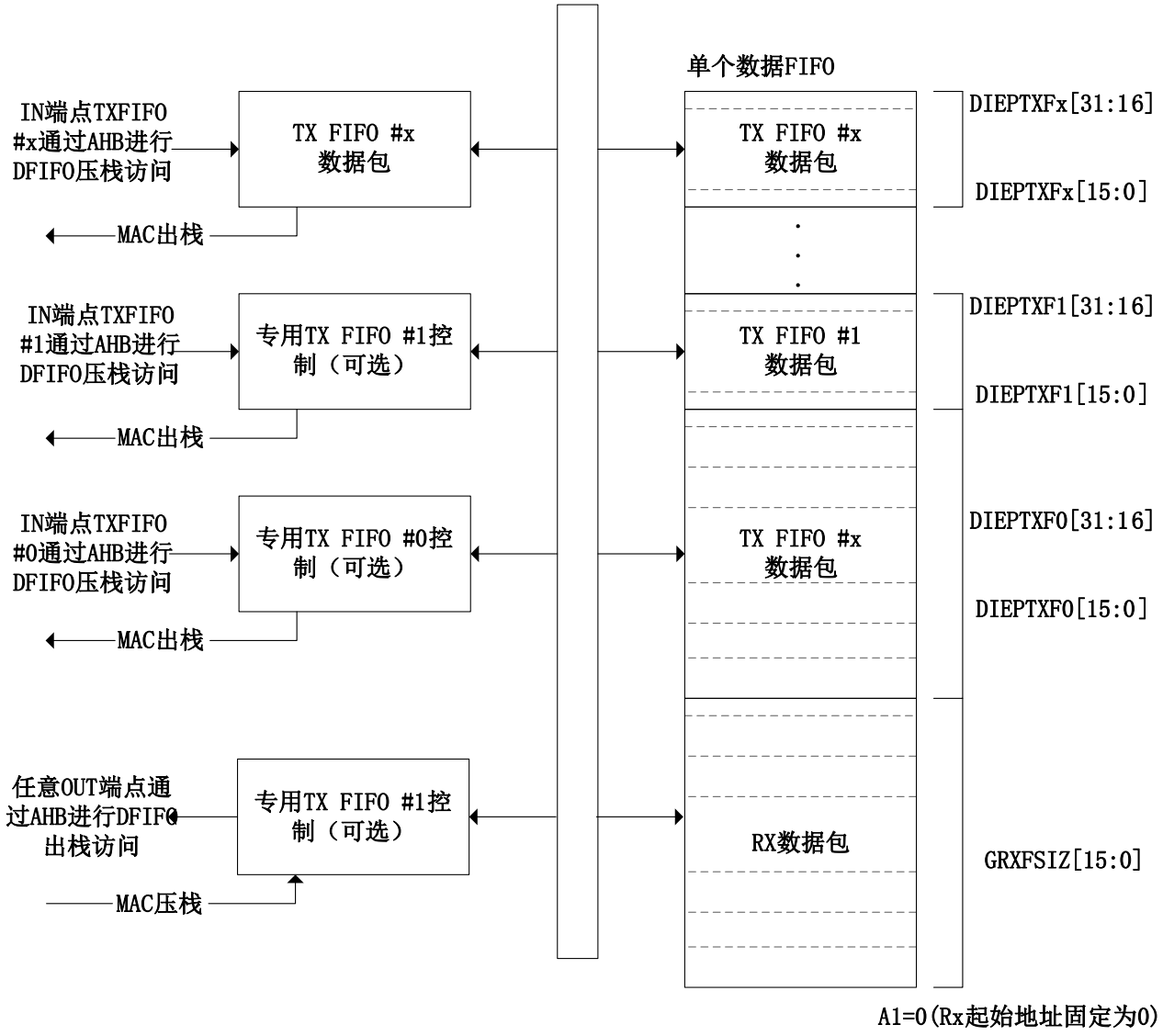
两个发送 FIFO 的架构使得 USB 主机能够对周期和非周期发送数据缓冲区分别进行优化管理：

- 配置为支持周期（非周期） OUT 事务的所有主机通道共享同一个 RAM 缓冲区（共享 FIFO）
- OTG_HS 模块可将主机发出的任何 OUT 通信填充到周期性（非周期性）发送 FIFO，直到没有多余空间。

只要周期性 Tx FIFO 为半空或全空， OTG_FS/OTG_HS 模块就会发出周期性 Tx FIFO 空中断（ GINTSTS 中的 PTxFEmp 位），具体取决于 AHB 配置寄存器中的周期性 Tx FIFO 空门限位（ GAHBCFG 中的 PTxFEmpLvl 位）的值。只要周期性 Tx FIFO 和周期性请求队列中均存在空闲空间，应用程序便可提前写入发送数据。可通过读取主机周期性发送 FIFO 和队列状态寄存器 (HPTXSTS) 来了解二者的可用空间。

只要非周期性 Tx FIFO 为半空或全空， OTG_FS/OTG_HS 模块就会发出非周期性 Tx FIFO 空中断（ GINTSTS 中的 NPTxFEmp 位），具体取决于 AHB 配置寄存器中的非周期性 Tx FIFO 空门限位（ GAHBCFG 中的 NPTxFEmpLvl 位）。只要非周期性 Tx FIFO 和非周期性请求队列中均存在空闲空间，应用程序便可写入发送数据。可通过读取主机非周期性发送 FIFO 和队列状态寄存器 (GNPTXSTS) 来了解二者的可用空间。

下图为主机模式下的 FIFO 地址映射和 AHB FIFO 访问映射：



37.11.3. FIFO RAM 分配

■ 设备模式

● 接收 FIFO RAM 分配：

应用程序应为 SETUP 数据包分配 RAM：

- 接收 FIFO 中必须保留 10 个位置以在控制端点上接收 SETUP 数据包。OTG 模块不会向这些为 SETUP 数据包保留的位置写入任何其它数据。
- 将会为全局 OUT NAK 分配一个位置。
- 状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小/4）+1 的空间。如果使能了多个同步端点，则为接收连续数据包分配的空间必须至少为（最大数据包大小/4）的两倍+1。通常，推荐的空间为（最大数据包/4+1）的两倍，这样当上一个数据包向 CPU 传送时， USB 可同时接收后续的数据包。
- 传输完成状态信息和该端点收到的最后一个数据包会一起被推入 FIFO。推荐为每个 OUT 端点分配一个位置存储该端点上的传输状态信息。器件 RxFIFO = （4 * 控制端点数量 + 6）+ （(所使用的最大 USB 数据包/4） + 用于状态信息的 1） + （ 2 *OUT 端点数量）+ 用于全局 NAK 的 1

例如：周期性 USB 数据包的 MPS 是 1024 个字节，非周期性 USB 数据包的 MPS 是 512 个字节。有三个 OUT 端点、三个 IN 端点、一个控制端点和三个主机通道。则器件 RxFIFO = （4 * 1 + 6）+ （(1024/4）+1）+ （2 * 4） + 1 = 276

● 发送 FIFO RAM 分配：

各个 IN 端点发送 FIFO 所需的最小 RAM 空间为该特定 IN 端点的最大数据包大小。

注： 为发送 IN 端点 FIFO 分配的空间越多， USB 的性能就越高。

■ 主机模式

● 接收 FIFO RAM 分配：

状态信息随各个接收数据包写入 FIFO。因此，必须至少为接收数据包分配（最大数据包大小 / 4） + 1 的空间。如果使能了多个同步通道，则为接收连续数据包分配的空间必须至少为（最大数据包大小 / 4）的两倍 + 1。通常，推荐的空间为（最大数据包/4 + 1）的两倍，这样当上一个数据包向 CPU 传送时， USB 可同时接收后续的数据包。

传输完成状态信息和主机通道中的最后一个数据包会一起被推入 FIFO。因此，必须为此分配一个位置。主机 RxFIFO = (（所用的最大 USB 数据包/4） + 用于状态信息的 1） + 用于传输完成的 1。

例如：主机 RxFIFO = (（ 1024/4） + 1） + 1 = 258

● 发送 FIFO RAM 分配：

主机非周期性发送 FIFO 所需的最小 RAM 为所支持的所有非周期性 OUT 通道上传输的最大数据包的大小。通常，推荐的空间为最大数据包大小的两倍，这样当 USB 正在发送当前数据包的同时，AHB 可以往发送 FIFO 填入下一个数据包。非周期性 TxFIFO = 所用的最大非周期性 USB 数据包/4。

例如：非周期性 TxFIFO = （ 512/4） = 128

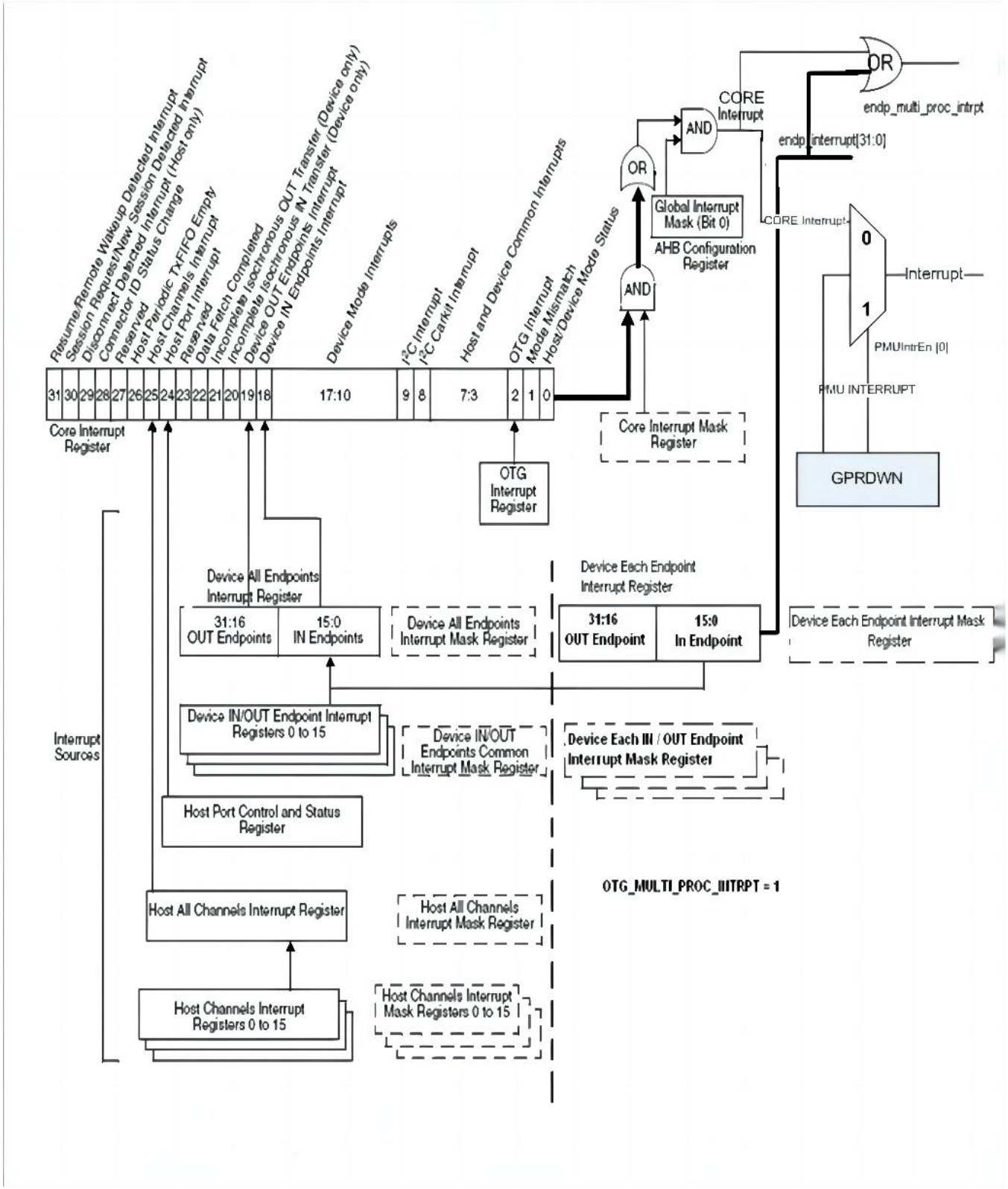
主机周期性发送 FIFO 所需的最小 RAM 为所支持的所有周期性 OUT 通道上传输的最大数据包的大小。如果至少有一个同步 OUT 端点，则空间必须至少为该通道中最大数据包大小的两倍。主机周期性 TxFIFO = 所用的最大周期性 USB 数据包/4。

例如：主机周期性 TxFIFO = （ 1024/4） = 256

注： 为非周期性发送 FIFO 分配的空间越多， USB 的性能就越高。

37.12. OTG_HS 中断

当 OTG_HS 控制器在一种模式下（设备模式或主机模式）工作时，应用程序不得以另一种角色模式访问寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器（GINTSTS 寄存器中的 MMIS 位）中反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。



37.13. OTG_HS 控制和状态寄存器

应用程序通过 AHB 从接口对控制和状态寄存器（CSR）进行读写操作，以此来控制 OTG_HS 模块。这些都是 32 位寄存器，其地址按 32 位对齐，因此只能以 32 位的方式访问。

CSR 分为以下几类：

- 模块全局寄存器
- 主机模式寄存器
- 主机全局寄存器
- 主机端口 CSR
- 主机通道相关寄存器
- 设备模式寄存器
- 设备全局寄存器
- 设备端点相关寄存器
- 电源和时钟门控寄存器
- 数据 FIFO (DFIFO) 访问寄存器

只有模块全局寄存器、电源和时钟门控寄存器、数据 FIFO 访问寄存器以及主机端口控制和状态寄存器，可在主机模式和设备模式下进行访问。当 OTG_HS 控制器在一种模式下（设备模式或主机模式）工作时，应用程序不得以另一种角色模式访问寄存器。如果发生了非法访问，将会产生模式不匹配中断并在模块中断寄存器（GINTSTS 寄存器中的 ModeMis 位）中反映。当模块从一种角色模式切换到另一种角色模式时，新工作模式下的寄存器必须重新编程为上电复位后的状态。

37.14. OTG_HS 编程模型

37.14.1. 模块初始化

应用程序必须执行模块初始化序列。如果上电期间连接电缆，则 GINTSTS 中的当前工作模式位（GINTSTS.CurMod）将指示模式。连接“A 型”插头后，OTG_HS 控制器进入主机模式；连接“B 型”插头后， OTG_FS 控制器进入设备模式。

本节介绍了 OTG_HS 控制器在上电后的初始化过程。无论是以主机模式还是设备模式工作，应用程序都必须遵循初始化序列。根据模块配置对所有模块全局寄存器进行初始化：

1）在 GAHBCFG/GINTSTS 寄存器中编程以下字段：

- 全局中断屏蔽位 GAHBCFG.GlblIntrMsk = 1
- RxFIFO 非空（GINTSTS.RxFLvl）
- 周期性 TxFIFO 空门限(GINTSTS.PTxFEmp)

2）在 GUSBCFG 寄存器中编程以下字段：

- HNP 功能位(GUSBCFG.HNPCap)
- SRP 功能位(GUSBCFG.SRPCap)
- OTG_HS 超时校准字段(GUSBCFG.TOutCal)
- USB 周转时间字段(GUSBCFG.USBTrdTim)

3）软件必须使能 GINTMSK 寄存器中的以下位：

- OTG 中断屏蔽(GINTMSK.OTGIntMsk)

- 模式不匹配中断屏蔽(GINTMSK.ModeMisMsk)

4）通过读取 GINTSTS.CurMod，软件可确定 OTG_HS 控制器是在主机模式还是设备模式下工作。

37.14.2. 主机初始化

要将模块作为主机进行初始化，应用程序必须执行以下步骤：

1）编程 GINTMSK 寄存器中的主机通道中断屏蔽位 HChIntMsk 以打开中断。

2）编程 HCFG 寄存器以选择主机模式。

3）将 HPRT 中的 PrtPwr 位编程为 1，给 USB 总线提供 VBUS。

4）等待 HPRT 中的 PrtConnDet 中断。这表示某设备已连接到主机端口。

5）将 HPRT 中的 PrtRst 位编程为 1，在 USB 总线上发出复位信号。

6）至少等待 10ms，以便完成复位过程。

7）将 HPRT 中的 PrtRst 位编程为 0，

8）等待 HPRT 中的 PrtEnChng 中断。

9）读取 HPRT 中的 PrtSpd 位以获取枚举速度。

10）使用所选 PHY 时钟 ，相应地设置 HFIR 寄存器。

11）根据步骤 9 中检测到的设备速度编程 HCFG 寄存器中的 FSLSPclkSel 字段。如果 FSLSPclkSel 发生更改，则必须执行端口复位。

12）编程 GRXFSIZ 寄存器以选择接收 FIFO 的大小。

13）编程 GNPTXFSIZ 寄存器，以选择用于非周期性通信事务的非周期性发送 FIFO 的大小和起始地址。

14）编程 HPTXFSIZ 寄存器，以选择用于周期性通信事务的周期性发送 FIFO 的大小和起始地址。

要与设备通信，系统软件必须初始化并使能至少一个通道。

37.14.3. 设备初始化

上电期间或者从主机模式切换为设备模式后，应用程序必须执行下列步骤来将模块作为设备进行初始化。

1）在 DCFG 寄存器中编程以下字段：

- 设备速度（DCFG.DevSpd）
- 非零长度状态 OUT 握手信号（DCFG.NZStsOUTHShk）

2）编程 GINTMSK 寄存器以使能以下中断：

- USB 复位（GINTMSK.USBRstMsk）
- 枚举完成（GINTMSK.EnumDoneMsk）
- 早期挂起（GINTMSK.ErlySuspMsk）
- USB 挂起（GINTMSK.USBSuspMsk）
- SOF（GINTMSK.SofMsk）

3）等待 GINTSTS 中的 USBRst 中断。这表示已在 USB 上检测到复位信号，复位过程自接收到此中断后约持续 10ms。

4）等待 GINTSTS 中的 EnumDone 中断。此中断指示 USB 上复位过程结束。接收到此中断时，应用程序必须读取 DSTS 寄存器中的 EnumSpd 位以确定枚举速度。

此时，设备已准备好接受 SOF 数据包并在控制端点 0 上执行控制传输。

37.14.4. DMA 模式

OTG 主机使用 AHB 主接口来获取发送数据包数据（AHB 到 USB）和接收数据更新（USB 到 AHB）。AHB 主接口使用经过编程的 DMA 地址（主机模式下的 HCDMAx 寄存器和外设模式下的 DIEPDMAx/DOEPDMAx 寄存器）来访问数据缓冲区。

37.14.5. 主机编程模型

■ 通道初始化

应用程序必须初始化一个或多个通道，之后才能与所连接的设备通信。要初始化和使能通道，应用程序必须执行以下步骤：

1）编程 GINTMSK 寄存器以取消对以下位的中断屏蔽：

- 通道中断(GINTMSK.HChIntMsk)
- 用于 OUT 事务的非周期性发送 FIFO 为空（在流水线事务级别工作且数据包计数字段编程值大于 1 时适用）。
- 用于 OUT 事务的非周期性发送 FIFO 为半空（在流水线事务级别工作且数据包计数字段编程值大于 1 时适用）。

2）编程 HAINTMSK 寄存器以使能所选通道中断。

3）编程 HCINTMSK 寄存器，以使能主机通道中断寄存器中反映的和通信事务有关的中断。

4）编程所选通道的 HCTSIZx 寄存器，指定以字节为单位的总传输大小和包括短数据包在内的预期数据包个数。应用程序必须使用初始数据 PID（用于第一个 OUT 事务或预期从第一个 IN 事务获取）编程 PID 字段。

5）编程所选通道的 HCCHARx 寄存器，指定设备的端点特性，例如类型、速度、方向等。（仅当应用程序准备好发送或接收数据包时，才能通过将通道使能位置 1 来使能通道）。

6）使用集线器地址和端口地址在 HCSPLTx 寄存器中对所选通道进行编程（仅分离事务）。

7）使用缓冲区起始地址在 HCDMAx 寄存器中对所选通道进行编程（仅 DMA 事务）。

■ 通道的停止

应用程序可以通过编程 HCCHARx 寄存器将 ChDis 和 ChEna 位置 1 来禁止任何通道。这会使 OTG_HS 主机清空之前在该通道上发出的请求（如果有）并生成通道停止中断。应用程序在将通道重新分配给其它通信事务之前，必须等待 HCINTx 中的 ChHltd 中断。OTG_HS 主机不会中断已在 USB 上启动的通信事务。

要禁止某个在 DMA 模式下工作的通道，应用程序无需检查请求队列中是否有可用空间。OTG_HS 主机检查是否有空间，在仲裁给要禁止的通道时，写入通道禁止的请求。同时，当 HCCHARx 中的 ChDis 位置 1 时，将从请求队列中丢弃所有已发出的请求。

禁止通道前，应用程序必须确保非周期性请求队列（禁止非周期性通道时）或周期性请求队列（禁止周期性通道时）中至少有一个空闲空间。应用程序可以在请求队列已满时（禁止通道之前），通过编程 HCCHARx 寄存器将 ChDis 位置 1 和将 ChEna 位清零，清空已发出的请求。

出现以下任一情况时，应用程序将禁止通道：

1）IN 或 OUT 通道的 HCINTx 中接收到 STALL、TXERR、BBERR 或 DTERR 中断。应用程序在接收到通道停止信号之前，必须能够接收相同通道的其它中断（DTERR、Nak、Data、TXERR）。

2）在非周期性 IN 传输或高带宽中断 IN 传输期间 HCINTx 中接收到 XFRC 中断。

3）接收到 GINTSTS 中的 DISCINT（断开设备连接）中断。（应用程序将禁止所有已使能的通道）。

4）应用程序在传输正常完成之前将其终止。

■ Ping 协议

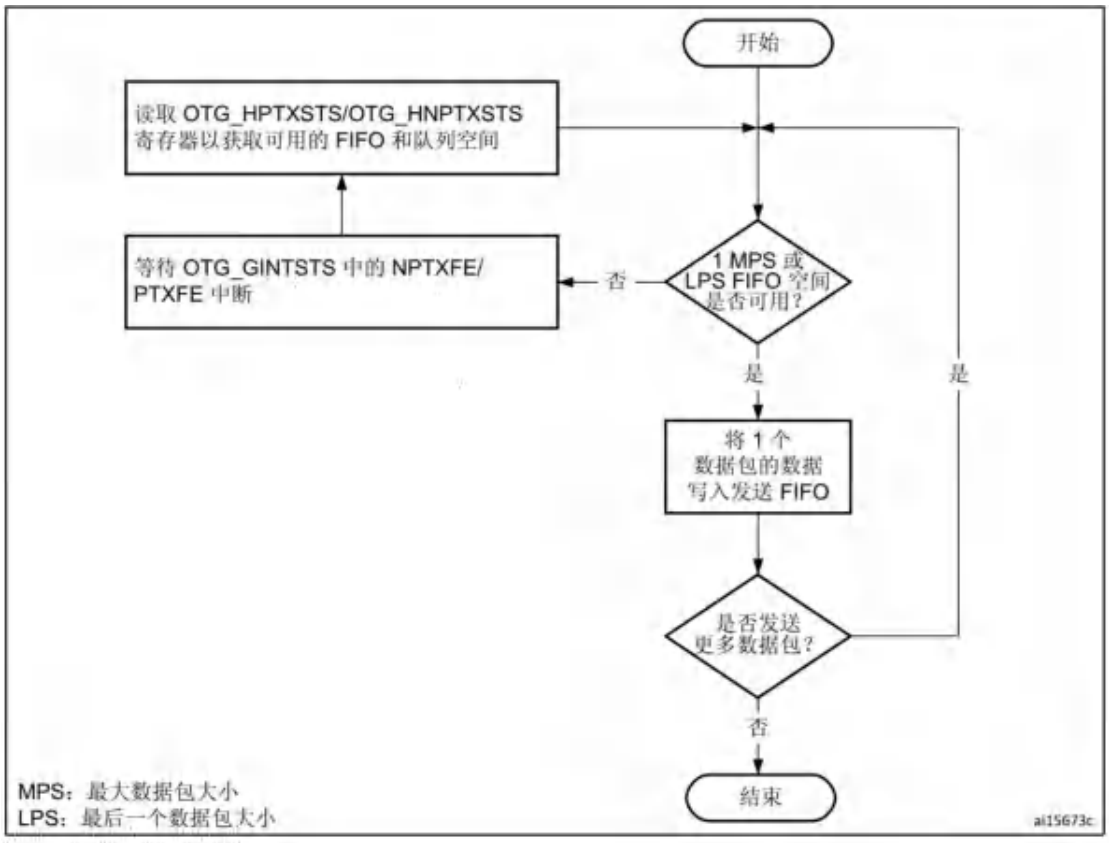
当 OTG_HS 主机工作在高速模式下时，如果应用程序要与高速批量或控制（数据和状态阶段）OUT 端点进行通信，则必须使用 ping 协议。当应用程序接收到 NAK/NYET/TXERR 中断时，也必须使用 ping 协议。当 OTG_HS 主机接收到上述其中一个响应时，它不会在该端点上继续执行任何通信事务，而是丢弃所有已发出或已获取的 OUT 请求（从请求队列中），然后清空发送 FIFO 中的相应数据。仅对从模式有效。在从模式下，应用程序可通过以下两种方式发送 ping 令牌：在使能通道之前，将 HCTSIZx 中的 DOPING 位置 1，或者在通道已使能之后，对 HCTSIZx 寄存器执行写操作时将 DOPING 位置 1。这将使能 OTG_HS 主机将 ping 请求写入到请求队列中。应用程序必须等待设备对 ping 令牌的响应（NAK、ACK 或 TXERR 中断），然后才能继续执行通信事务或发送另一个 ping 令牌。仅当应用程序从设备的 OUT 端点接收到对 ping 令牌的 ACK 响应后，才能继续执行数据通信事务。在 DMA 模式下工作时，对于批量/控制 OUT 事务，应用程序不需要在收到 NAK/NYET 回复时将 HCTSIZx 中的 DOPING 位置 1。OTG_HS 主机会自动将 HCTSIZx 中的 DOPING 位置 1，然后在批量/控制 OUT 传输时发出 ping 令牌。OTG_HS 主机会持续发送 ping 令牌，直至收到 ACK 为止，随后自动切换到数据通信事务。

■ 操作模型

应用程序必须初始化一个通道，之后才能与所连接的设备通信。本节介绍了针对不同 USB 事务类型要执行的操作序列。

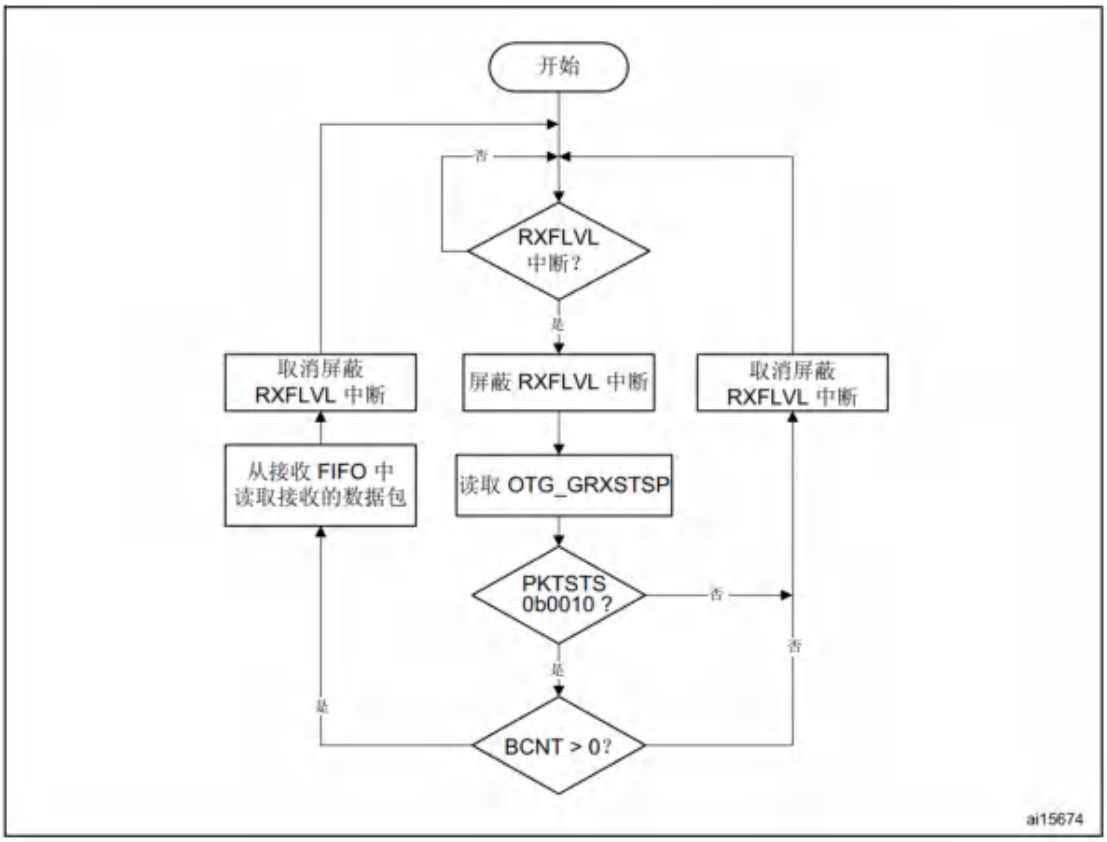
● 写入发送 FIFO

应用程序对数据包执行最后一个 DWORD 写操作的同时，OTG_HS 主机自动向周期性/非周期性请求队列写入一个条目（OUT 请求）。因此开始向发送 FIFO 写入数据之前，应用程序必须确保周期性/非周期性请求队列中至少有一个空闲空间。应用程序必须始终以 DWORD 形式向发送 FIFO 写入数据。如果数据包大小不是 DWORD 的整数倍，则应用程序必须将数据包填充到 DWORD 的整数倍。OTG_HS 主机根据设定的最大数据包大小和传输大小确定实际数据包大小。发送 FIFO 写任务：



● 读取接收 FIFO

应用程序必须忽略除 IN 数据包(bx0010)以外的所有数据包状态。 接收 FIFO 读任务：



● 批量和控制传输类型的 OUT/SETUP 通信事务

图“正常批量/控制 OUT/SETUP”显示了典型的批量或控制 OUT/ SETUP 流水线事务级操作。请参见通道 1 (ch_1)。该通道发送了两个批量 OUT 数据包。控制 SETUP 事务的工作方式相同，只不过只包含一个数据包。假设：

- 应用程序尝试发送两个最大数据包大小的数据包（传输大小=1024 字节）。
- 非周期性发送 FIFO 可存储两个数据包（HS 模式下为 1KB）。
- 非周期性请求队列深度=4。

● 正常批量和控制传输类型的 OUT/SETUP 操作

（通道 1）中的操作顺序如下：

1）初始化通道 1

2）写入通道 1 的第一个数据包

3）在应用执行最后一次 DWORD 写操作时，模块将向非周期性请求队列写入一个请求条目

4）只要非周期性队列非空，模块即会尝试在当前帧内发送一个 OUT 令牌

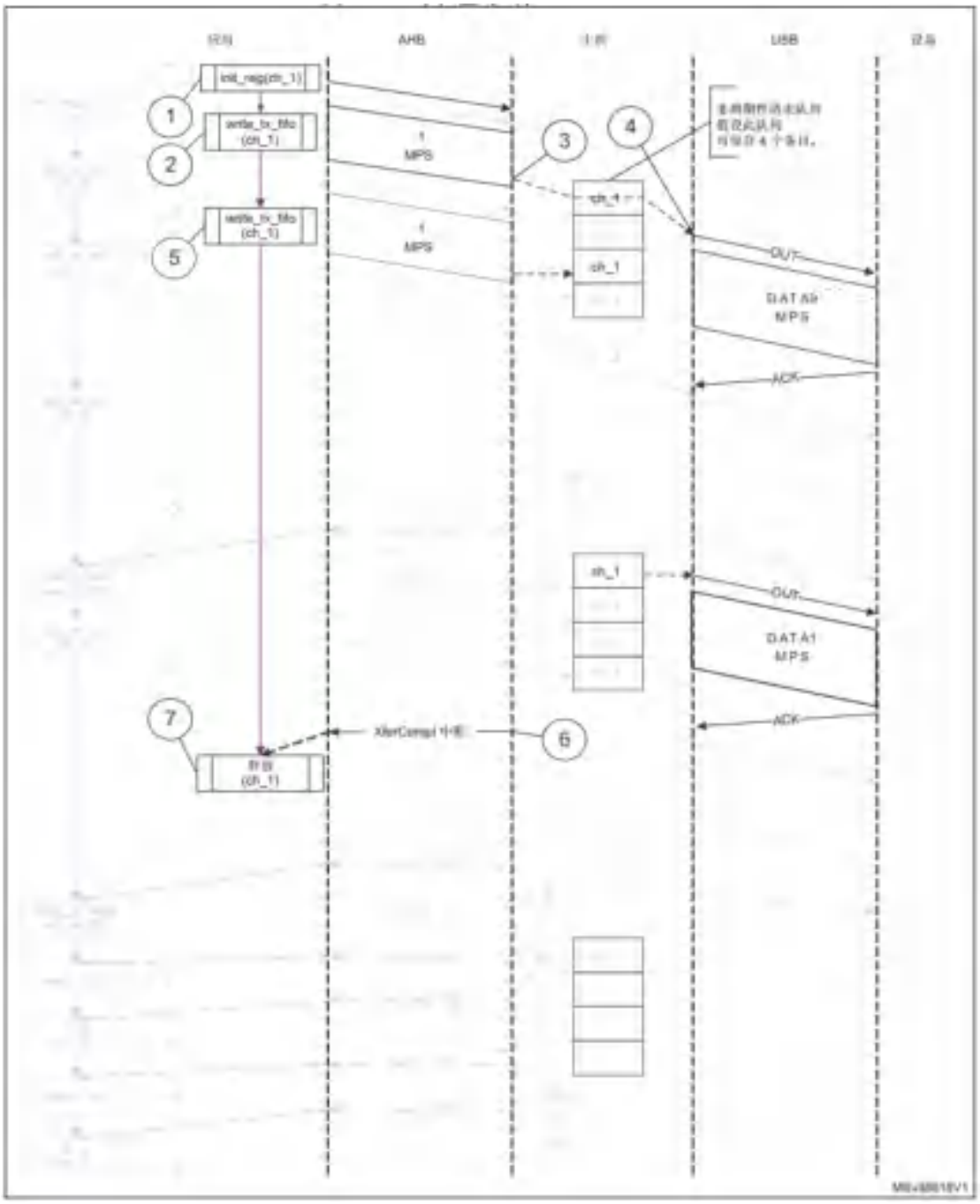
5）写入通道 1 的第二个（最后一个）数据包

6）成功完成最后一个事务后，模块立即生成 XFRC 中断

7）为了响应 XFRC 中断，将释放通道以供其它传输操作使用

8）处理非 ACK 响应

下图为正常批量/控制 OUT/SETUP：



1.灰显元素与此图的上下文不相关。

以下代码示例说明了批量和控制 OUT/SETUP 传输类型的通信事务的通道相关中断服务程序。

● 批量/控制 OUT/SETUP 和批量/控制 IN 事务的中断服务程序

1) 批量/控制 OUT/SETUP

当发送 FIFO 和请求队列中有可用空间时，应用程序会将数据包写入发送 FIFO。应用程序可利用 OTG_GINTSTS 中的 NPTXFE 中断确定发送 FIFO 空间。

2) 批量/控制 IN

当请求队列空间可用时，应用程序会写入请求，直到接收到 XFRC 中断。

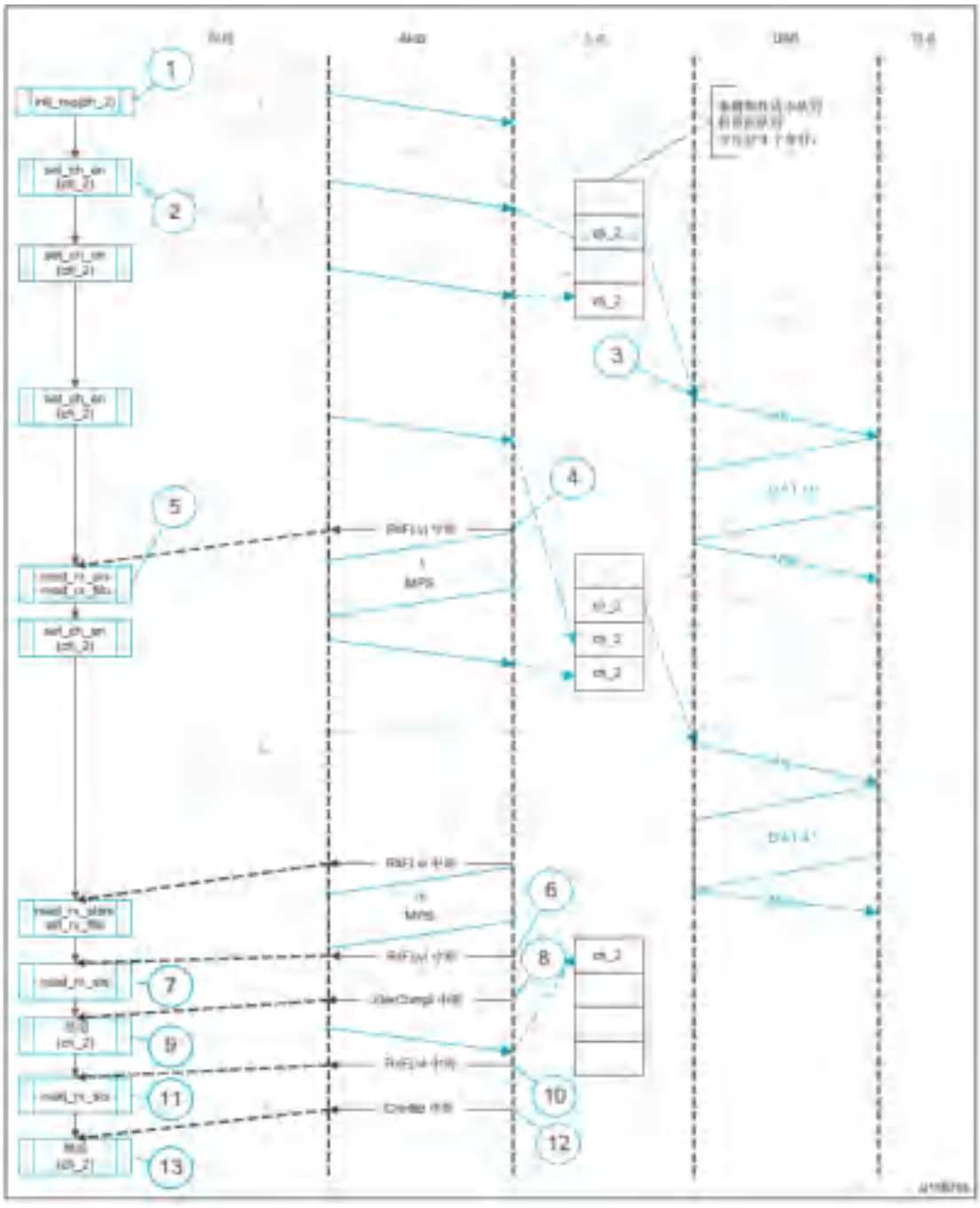
● 批量和控制 IN 事务

图 750 显示了典型的批量或控制传输类型的 IN 流水线事务级操作。请参见通道 2(ch_2)。假设：

- 应用程序要接收两个最大数据包大小的数据包（传输大小=1024 字节）。
- 接收 FIFO 可以包含至少一个最大数据包大小的数据包和每个数据包的两个状态字（HS 对应 520 字节）。

➢ 非周期性请求队列深度=4。

图 37-2 批量/控制 IN 事务



1.灰显元素与此图的上下文不相关。

操作顺序如下：

1）初始化通道 2。

2）将 HCCHAR2 中的 CHENA 位置 1，以向非周期性请求队列写入 IN 请求。

3）模块尝试在完成当前 OUT 事务后发送 IN 令牌。

4）接收到的数据包写入接收 FIFO 后，模块立即生成 RXFLVL 中断。

5）为了响应 RXFLVL 中断，将屏蔽 RXFLVL 中断并读取接收到的数据包状态，以此确定接收的字节数，然后相应地读取接收 FIFO。之后使能 RXFLVL 中断。

6）模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。

7）应用程序必须读取接收数据包状态，并在不是 IN 数据包（GRXSTSR 中的 PKTSTS≠0b0010）时忽略它。

8）读取接收数据包状态后，模块立即生成 XFRC 中断。

9）为了响应 XFRC 中断，将禁止通道并停止针对其它请求向 HCCHAR2 写入数据。向 HCCHAR2 寄存器写入数据时，模块立即向非周期性请求队列写入通道禁止请求。

10）停止状态写入接收 FIFO 后，模块立即生成 RXFLVL 中断。

11）读取并忽略接收数据包状态。

12）只要停止状态从接收 FIFO 弹出，模块立即生成 CHH 中断。

13）为了响应 CHH 中断，将释放通道以供其它传输操作使用。

14）处理非 ACK 响应

● 控制事务

控制传输的建立、数据和状态阶段必须作为三个独立的传输过程来执行。建立、数据和状态阶段的 OUT 事务与上文所述的批量 OUT 事务的执行方式类似。数据或状态阶段的 IN 事务与上文所述的批量 IN 事务的执行方式类似。在所有这三个阶段，应用程序都会将 HCCHAR1 中的 EPTYP 字段设置为控制传输类型。在建立阶段期间，应用程序会将 HCTSIZ1 中的 PID 字段设置为 SETUP。

● 中断 OUT 事务

图 751 显示了典型的中断 OUT 操作。假设：

- 应用程序尝试从奇数帧开始，每帧发送一个数据包（高达 1 个最大数据包大小）（传输大小=1024 字节）
- 周期性发送 FIFO 可存储一个数据包（1KB)
- 周期性请求队列深度=4

操作顺序如下：

1）初始化和使能通道 1。应用程序必须将 HCCHAR1 中的 ODDFRM 位置 1。

2）写入通道 1 的第一个数据包。

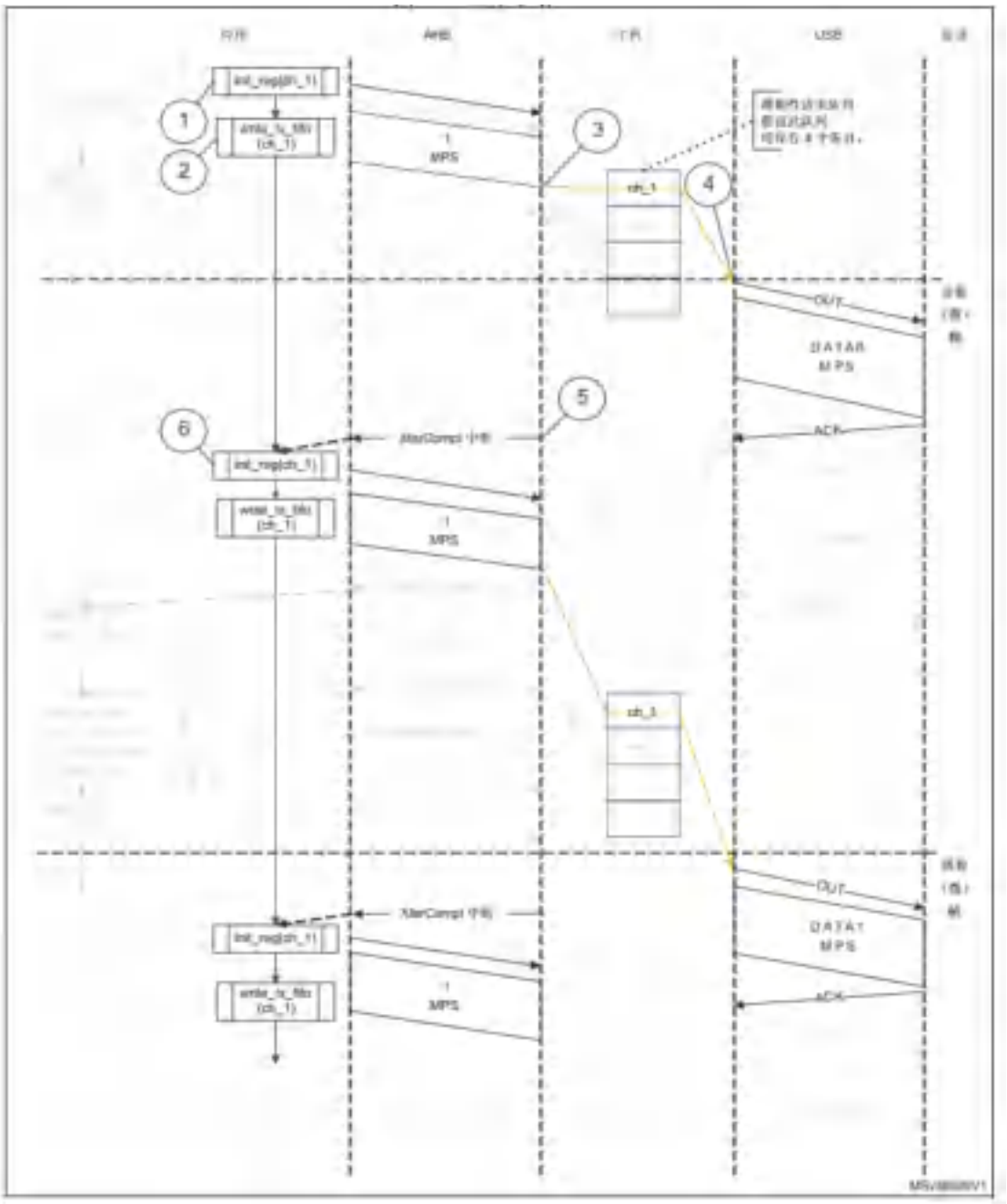
3）应用程序在执行每个数据包的最后一次 DWORD 写操作时， OTG_HS 主机向周期性请求队列写入一个请求条目。

4）OTG_HS 主机尝试在下一（奇数）帧发送 OUT 令牌。

5）成功发送最后一个数据包后， OTG_HS 主机立即生成 XFRC 中断。

6）为了响应 XFRC 中断，将重新初始化通道以供下一传输操作使用。

图 37-3 正常中断 OUT



1. 灰显元素与此图的上下文不相关。

● 中断 OUT/IN 事务的中断服务程序

1) 中断 OUT

应用程序利用 OTG_GINTSTS 中的 NPTXFE 中断确定发送 FIFO 空间。

2) 中断 IN

● 中断 IN 事务

假设：

- 应用程序要从奇数帧开始，每帧接收一个数据包（最大 1 个最大数据包大小）（传输大小=1024 字节）。
- 接收 FIFO 可以保存至少一个最大数据包大小的数据包和每个数据包的两个状态字（1031 字节）。
- 周期性请求队列深度=4。

正常中断 IN 操作

操作顺序如下：

1）初始化通道 2。应用程序必须将 HCCHAR2 中的 ODDFRM 位置 1。

2）将 HCCHAR2 中的 CHENA 位置 1，以将 IN 请求写入周期性请求队列。

3）只要 CHENA 置位，对于每次 HCCHAR2 寄存器写操作，OTG_HS 主机都会将一个 IN 请求写入周期性请求队列。

4）OTG_HS 主机尝试在下一奇数帧发送 IN 令牌。

5）接收到 IN 数据包并写入接收 FIFO 后， OTG_HS 主机便会立即生成 RXFLVL 中断。

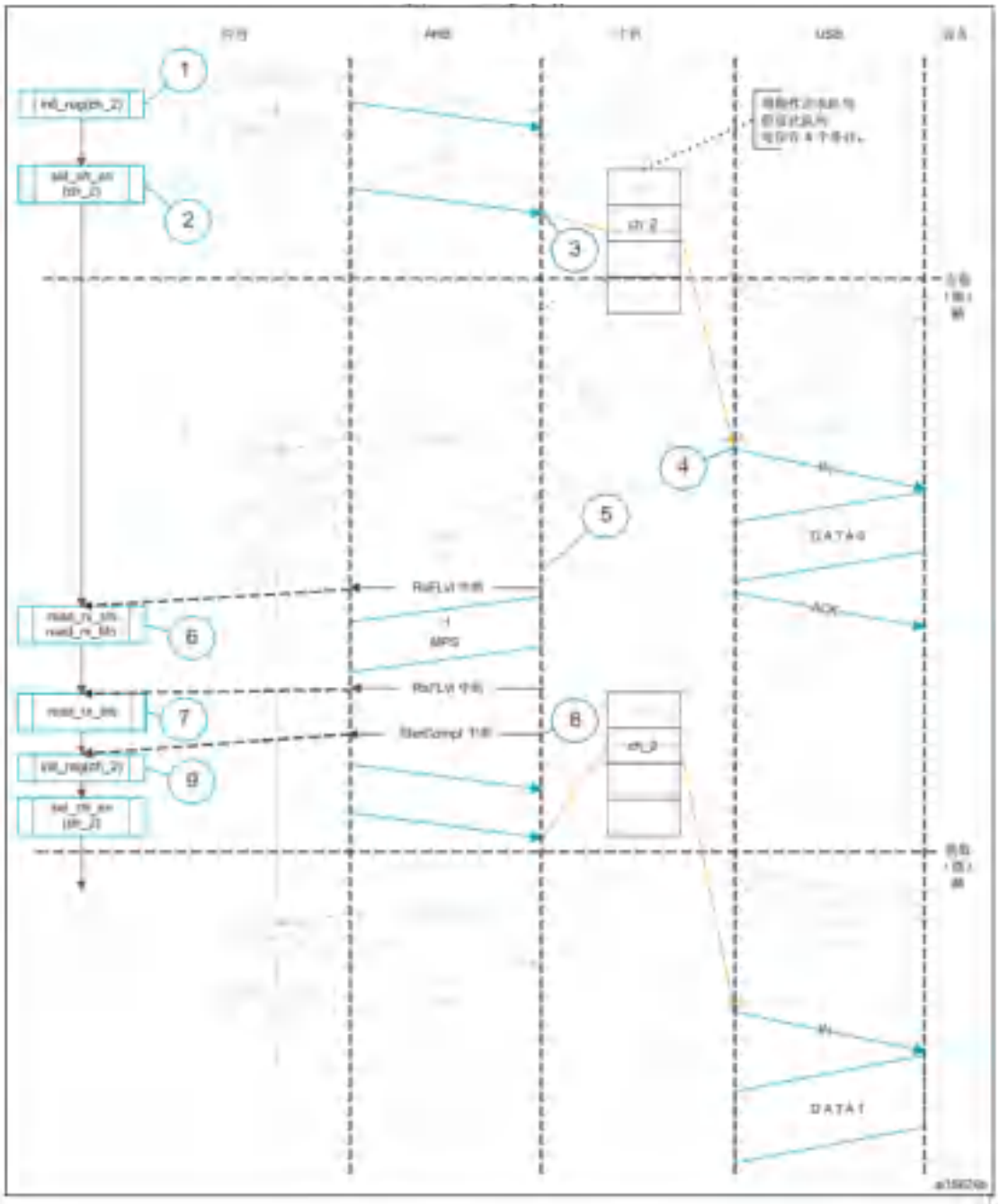
6）为响应 RXFLVL 中断，读取接收到的数据包状态以确定接收的字节数，然后相应地读取接收 FIFO。应用程序必须在读取接收 FIFO 前屏蔽 RXFLVL 中断，并且在读取整个数据包后使能。

7）模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。应用程序必须读取接收数据包状态，并在不是 IN 数据包（GRXSTSR 中的 PKTSTS≠0b0010）时忽略它。

8）读取接收数据包状态后，模块立即生成 XFRC 中断。

9）为响应 XFRC 中断，将读取 HCTSIZ2 中的 PKTCNT 字段。如果 HCTSIZ2 中的 PKTCNT 位不等于 0，则在重新初始化通道以进行下次传输前（如果存在），禁止该通道。如果 HCTSIZ2 中的 PKTCNT 位等于 0，则重新初始化通道以进行下次传输。此时，应用程序必须复位 HCCHAR2 中的 ODDFRM 位。

图 37-4 正常中断 IN



1. 灰显元素与此图的上下文不相关。

● 同步 OUT 事务

图 752 中显示了典型的同步 OUT 操作。假设：

- 应用程序尝试从奇数帧开始，每帧发送一个数据包（最大 1 个最大数据包大小）。(传输大小 = 1024 字节)。
- 周期性发送 FIFO 可存储一个数据包 (1 KB)。
- 周期性请求队列深度 = 4。

操作顺序如下：

1）初始化和使能通道 1。应用程序必须将 OTG_HCCHAR1 中的 ODDFRM 位置 1。

2）写入通道 1 的第一个数据包。

3）应用程序在执行每个数据包的最后一次 DWORD 写操作时， OTG_HS 主机向周期性请求队列写入一个请求条目。

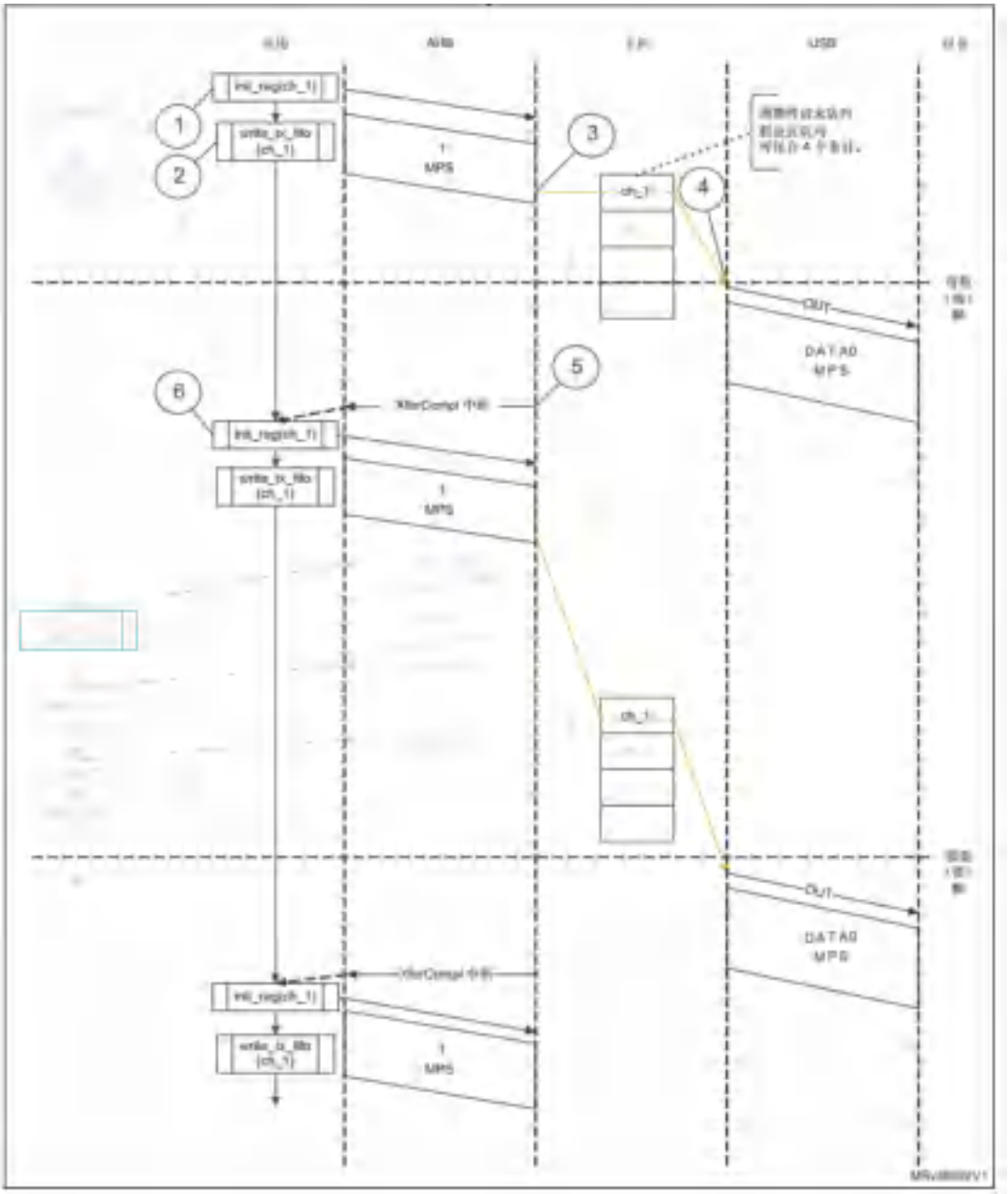
4）OTG_HS 主机尝试在下一（奇数）帧发送 OUT 令牌。

5）成功发送最后一个数据包后， OTG_HS 主机立即生成 XFRC 中断。

6）为了响应 XFRC 中断，将重新初始化通道以供下一传输操作使用。

7）处理非 ACK 响应。

图 37-5 同步 OUT 事务



1. 灰显元素与此图的上下文不相关。

● 同步 IN 事务

假设：

- 应用程序尝试从下一个奇数帧开始，每帧接收一个数据包（最大 1 个最大数据包大小）（传输大小 = 1024 字节）。
- 接收 FIFO 可以保存至少一个最大数据包大小的数据包和每个数据包的两个状态字（ 1031 字节）。
- 周期性请求队列深度 = 4。

操作顺序如下：

1）初始化通道 2。应用程序必须将 HCCHAR2 中的 ODDFRM 位置 1。

2）将 HCCHAR2 中的 CHENA 位置 1，以将 IN 请求写入周期性请求队列。

3）只要 CHENA 置位，对于每次 HCCHAR2 寄存器写操作，OTG_HS 主机都会将一个 IN 请求写入周期性请求队列。

4）OTG_HS 主机尝试在下一（奇数）帧发送 IN 令牌。

5）接收到 IN 数据包并写入接收 FIFO 后， OTG_HS 主机便会立即生成 RXFLVL 中断。

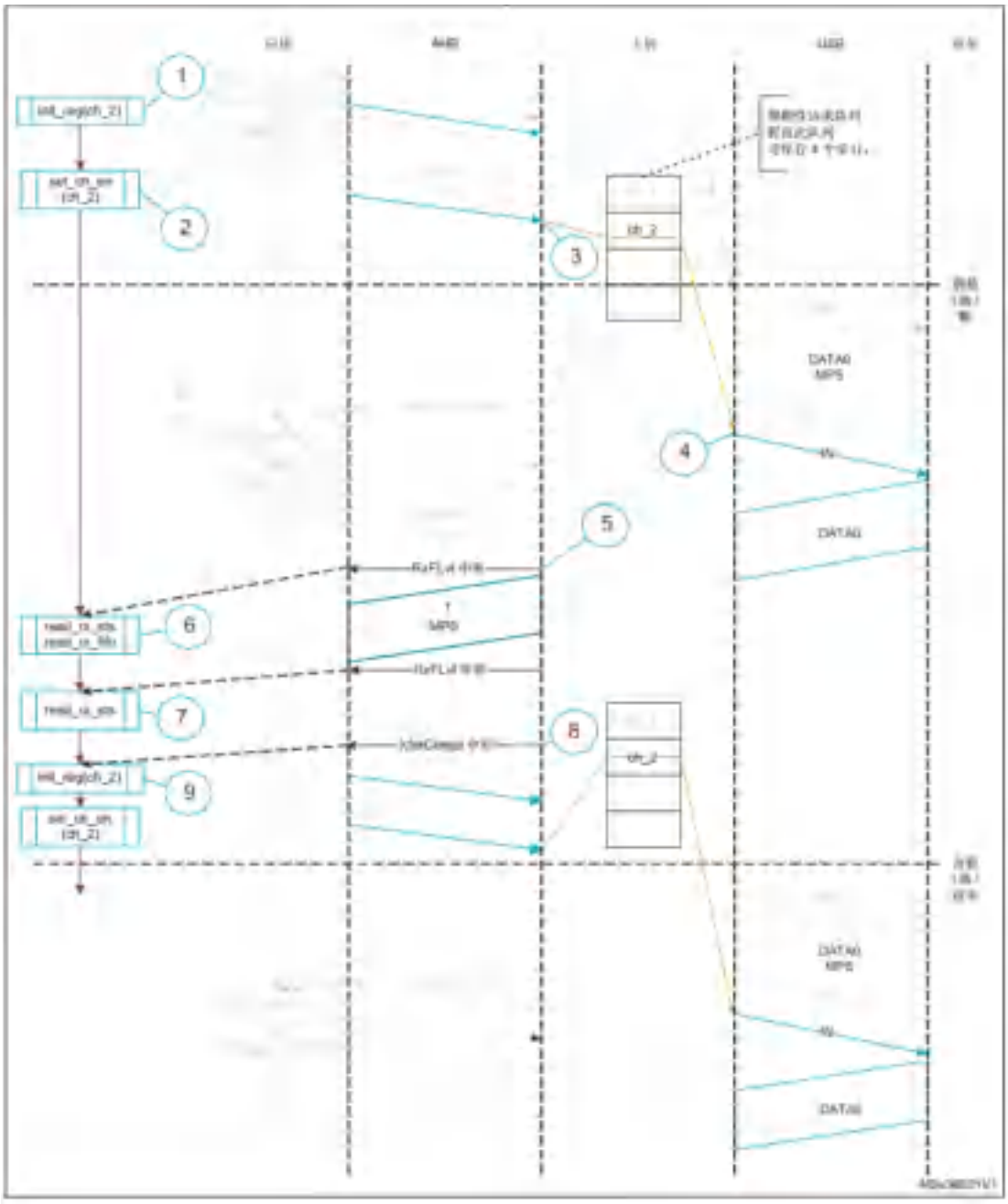
6）为响应 RXFLVL 中断，读取接收到的数据包状态以确定接收的字节数，然后相应地读取接收 FIFO。应用程序必须在读取接收 FIFO 前屏蔽 RXFLVL 中断，并且在读取整个数据包后使能。

7）模块针对接收 FIFO 中的传输完成状态条目生成 RXFLVL 中断。应用程序必须读取接收数据包状态，并在不是 IN 数据包（GRXSTSR 中的 PKTSTS 位 ≠ 0b0010）时忽略它。

8）读取接收数据包状态后，模块立即生成 XFRC 中断。

9）为响应 XFRC 中断，将读取 HCTSIZ2 中的 PKTCNT 字段。如果在 HCTSIZ2 中的 PKTCNT≠0，则在重新初始化通道以进行下次传输前（如果存在），禁止该通道。如果 HCTSIZ2 中的 PKTCNT=0，则重新初始化通道以进行下次传输。此时，应用程序必须复位 HCCHAR2 中的 ODDFRM 位。

图 37-6 同步 IN 事务



1. 灰显元素与此图的上下文不相关。

● 选择队列深度

请谨慎选择周期性和非周期性请求队列深度，以与要访问的周期性/非周期性端点数量相匹配。非周期性请求队列深度会影响非周期性传输的性能。队列越深（加上足够的 FIFO 大小），模块就更能对非周期性传输进行流水线处理。如果队列大小太小，则仅在队列空间释放时模块才能放入新请求。要按调度计划执行周期性传输，模块的周期性请求队列深度至关重要。根据一个微帧内要安排的周期性传输次数，选择周期性队列深度。如果周期性请求队列深度小于一个微帧内要安排的周期性传输数，则会发生帧溢出情况。

● 处理串扰情况

OTG_HS 控制器处理两种情况的 babble：数据包 babble 和端口 babble。如果设备发送的数据量超过通道的最大数据包大小，则会发生数据包串扰。如果模块从 EOF2（非常接近下一帧的 SOF）时刻还在从设备接收数据，则发生端口串扰。当 OTG_HS 控制器检测到数据包 babble 时，它停止向 Rx 缓冲区中写入数据并等待数据包结束信号（EOP）。当该控制器检测到 EOP 时，它会清空 Rx 缓冲区中的已写入数据并对应用程序生成串扰中断。

当 OTG_HS 控制器检测到端口 babble 时，它会清空 RxFIFO 并禁止端口。模块随后将生成“端口已禁止”

中断（GINTSTS 中的 HPRTINT 位和 HPRT 中的 PENCHNG 位）。在收到该中断时，应用程序必须通过检查 HPRT 中的 POCA 位，来确定该中断并非由过流（“端口已禁止”中断的另一个原因）所致，然后执行软复位。检测到端口串扰情况后，模块不再发送任何其它令牌。

● DMA 模式下的批量和控制 OUT/SETUP 事务

操作顺序如下：

1）按照通道初始化一节中的说明初始化并使能通道 1。

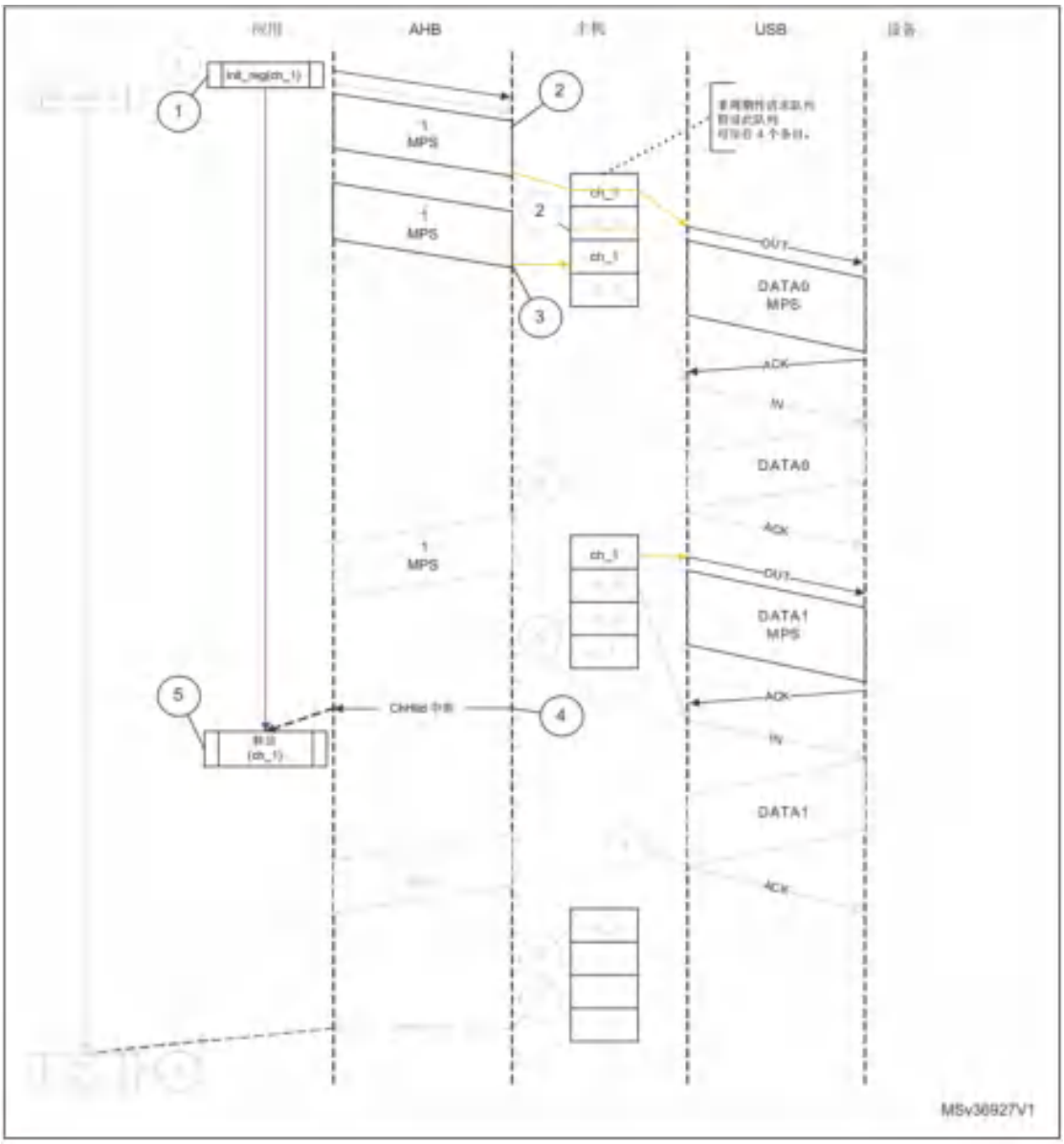
2）使能通道后， OTG_HS 主机即开始获取第一个数据包以发送。对于内部 DMA 模式，OTG_HS 主机使用经过编程的 DMA 地址来获取数据包。

3）获取第二个（最后一个）数据包的最后一个 DWORD 后， OTG_HS 主机会在内部屏蔽通道 1，不参与仲裁。

4）最后一个数据包发送出去后， OTG_HS 主机立即生成一个 CHH 中断。

5）为了响应 CHH 中断，将释放通道以供其它传输操作使用。

图 37-7 正常批量/控制 OUT/SETUP 事务 – DMA



● 使用内部 DMA 处理 NAK 和 NYET：

1) OTG_HS 主机发送批量传输类型的 OUT 事务。

2) 设备回复 NAK 或 NYET。

3) 如果应用程序没有屏蔽 NAK 或 NYET 中断，模块将为应用程序生成相应的中断。应用程序并不需要处理这些中断，因为模块会负责调整缓冲区指针和重新初始化通道，而无需应用程序干预。

4) 模块自动发出一个 ping 令牌。

5) 当设备返回 ACK 后，模块会继续传输。应用程序也可以选择使用这些中断（在这种情况下，NAK 或 NYET 中断将被应用程序屏蔽）。

当主机接收到 NAK 或 NYET 时， 模块不会生成单独的中断。

● DMA 模式下的批量和控制 IN 事务

操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能所使用的通道（通道 x）。

2) 当通道接收到来自仲裁器的授权后（以循环方式执行仲裁），OTG_HS 会立即向请求队列写入一个 IN 请求。

3) 当正确接收到最后一个字节后，OTG_HS 主机便开始将接收到的数据写入到系统存储器中。

4) 当接收到最后一个数据包后，OTG_HS 会将一个内部标志置 1，以便从请求队列中删除任何额外的 IN 请求。

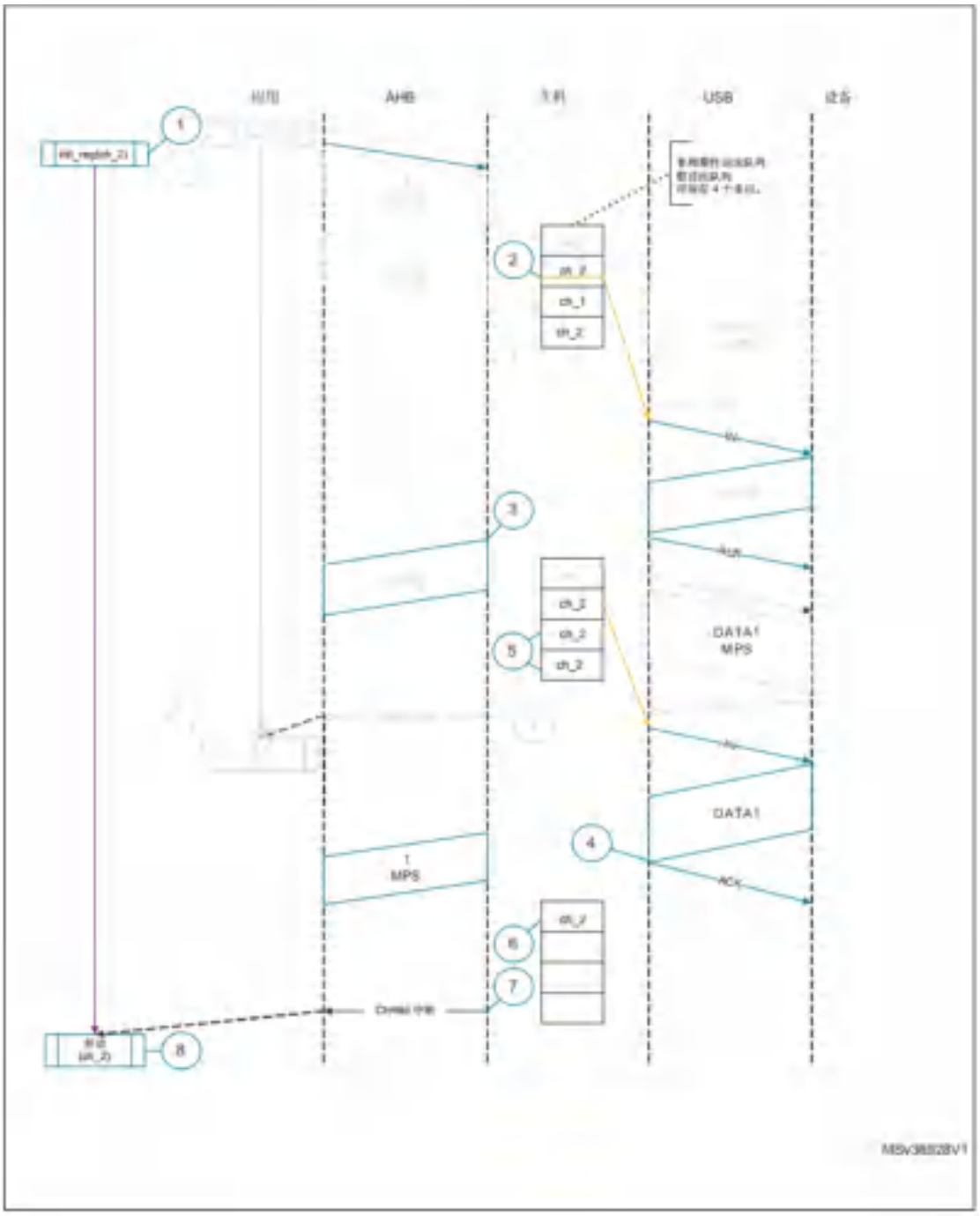
5) OTG_HS 主机会清空额外请求。

6) 向请求队列写入最后一个禁止通道的请求。此时，会在内部屏蔽通道 2，不参与仲裁。

7) 当禁止请求到达队列顶端时，OTG_HS 主机即会生成 CHH 中断。

8) 为了响应 CHH 中断，将释放通道以供其它传输操作使用。

图 37-8 正常批量/控制 IN 事务 – DMA



● DMA 模式下的中断 OUT 事务

1) 按照通道初始化一节中的说明初始化并使能通道 x。

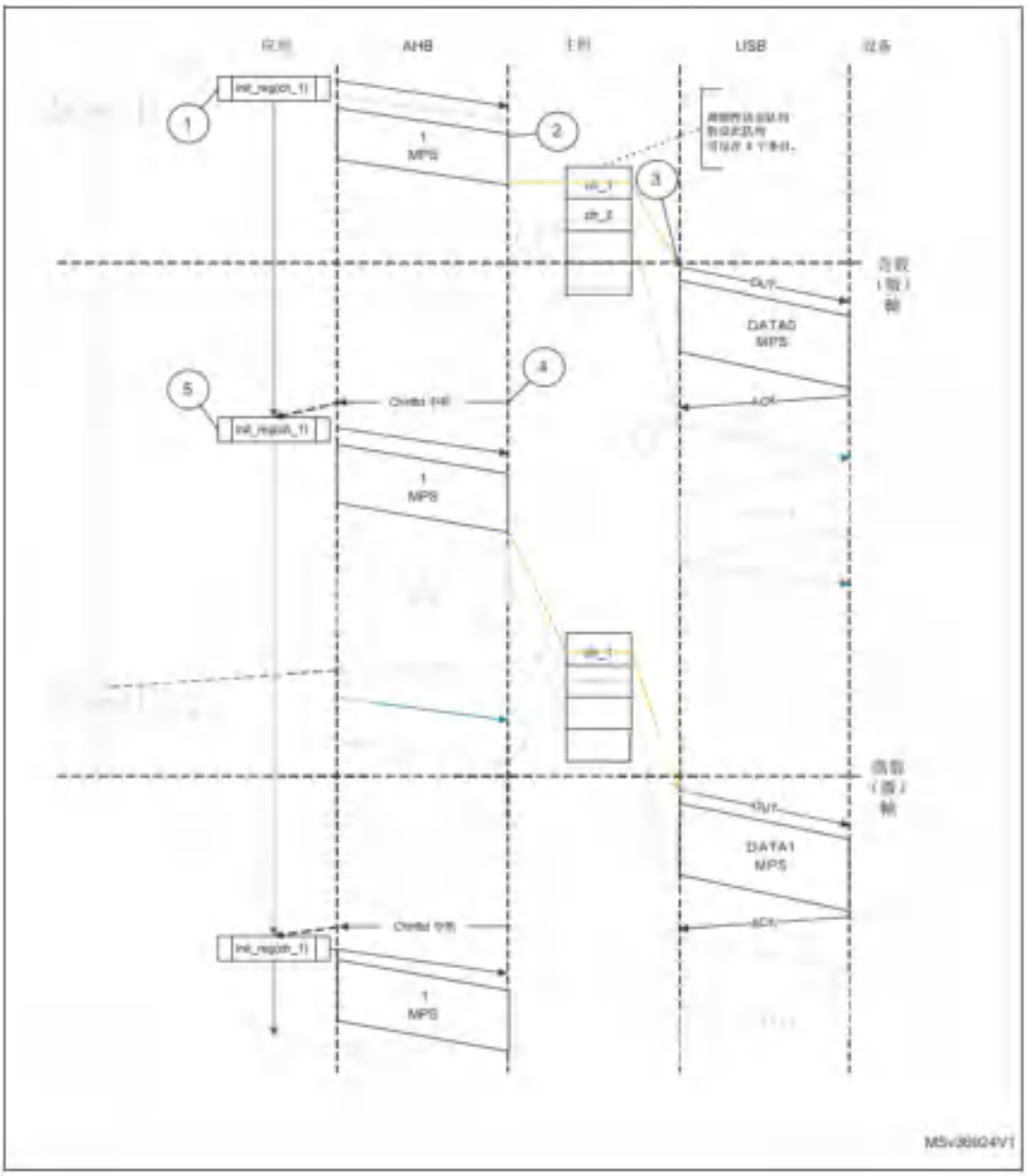
2) 使能通道后，OTG_HS 主机开始获取第一个数据包以发送，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。在高带宽传输模式下，OTG_HS 主机会继续获取下一个数据包（直至数据包数量达到 MC 字段中指定的数值），然后才能切换到下一个通道。

3) OTG_HS 主机尝试在下一个奇数帧/微帧的起始位置发送 OUT 令牌。

4) 成功发送数据包后，OTG_HS 主机将生成一个 CHH 中断。

5) 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 37-9 正常中断 OUT 事务 - DMA 模式



● DMA 模式下的中断 IN 事务

通道 x 的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x。

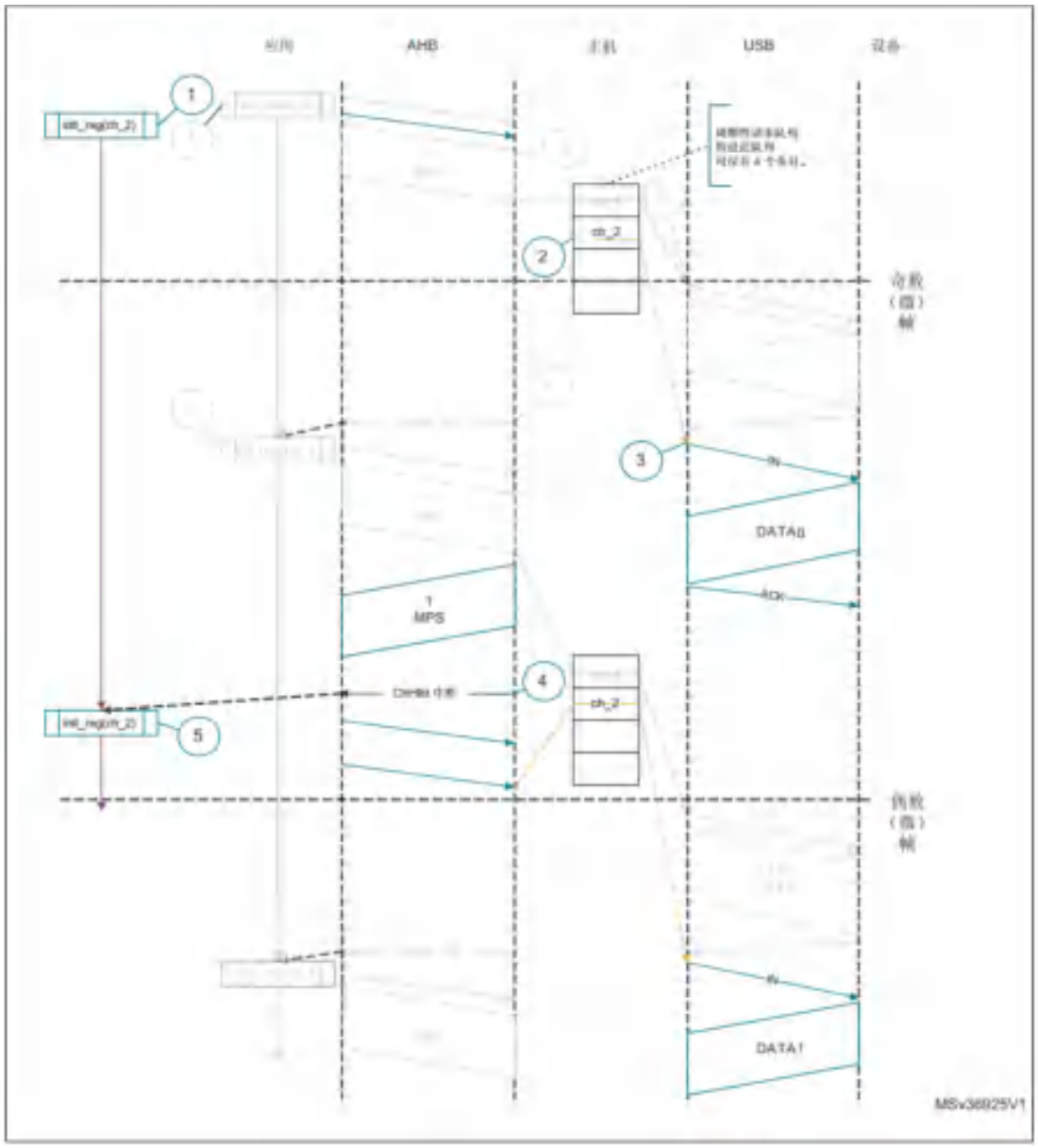
2) 当通道获取了来自仲裁器的授权后（以循环方式执行仲裁）， OTG_HS 主机会立即向请求队列写入一个 IN 请求。在高带宽传输模式下， OTG_HS 主机将执行连续写入操作，直到写入的数据包个数达到 MC。

3) OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 IN 令牌。

4) 接收到 IN 数据包并写入接收 FIFO 后， OTG_HS 主机便会立即生成 CHH 中断。

5) 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 37-10 正常中断 IN 事务 - DMA 模式



● DMA 模式下的同步 OUT 事务

1) 按照通道初始化一节中的说明初始化并使能通道 x。

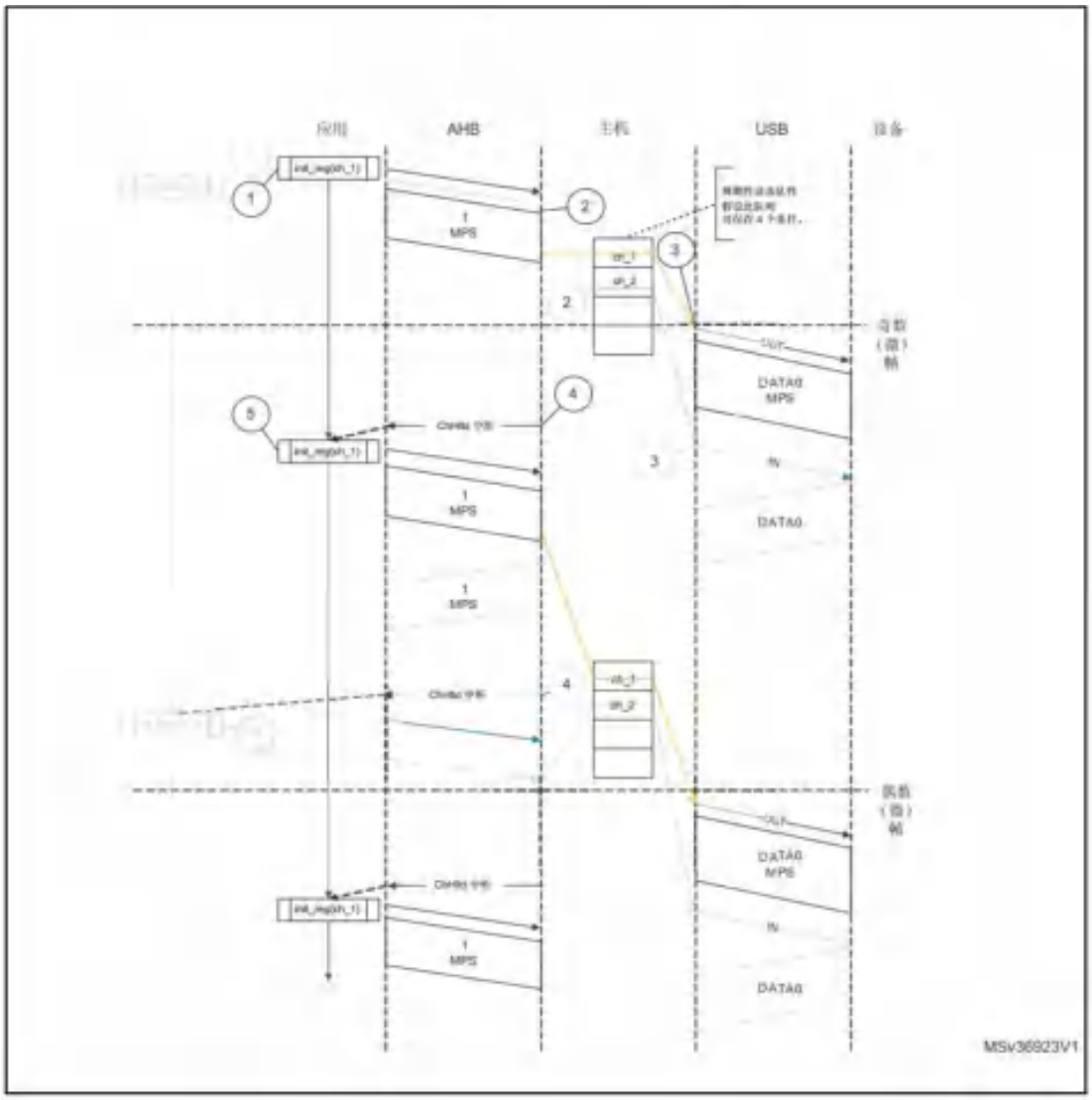
2) 使能通道后，OTG_HS 主机开始获取第一个数据包，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。在高带宽传输模式下，OTG_HS 主机会继续获取下一个数据包（直到写入的数据包个数达到 MC），然后才能切换到下一个通道。

3) OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 OUT 令牌。

4) 成功发送数据包后，OTG_HS 主机将生成一个 CHH 中断。

5) 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 37-11 正常同步 OUT 事务 - DMA 模式



● DMA 模式下的同步 IN 事务

通道 x 的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x。

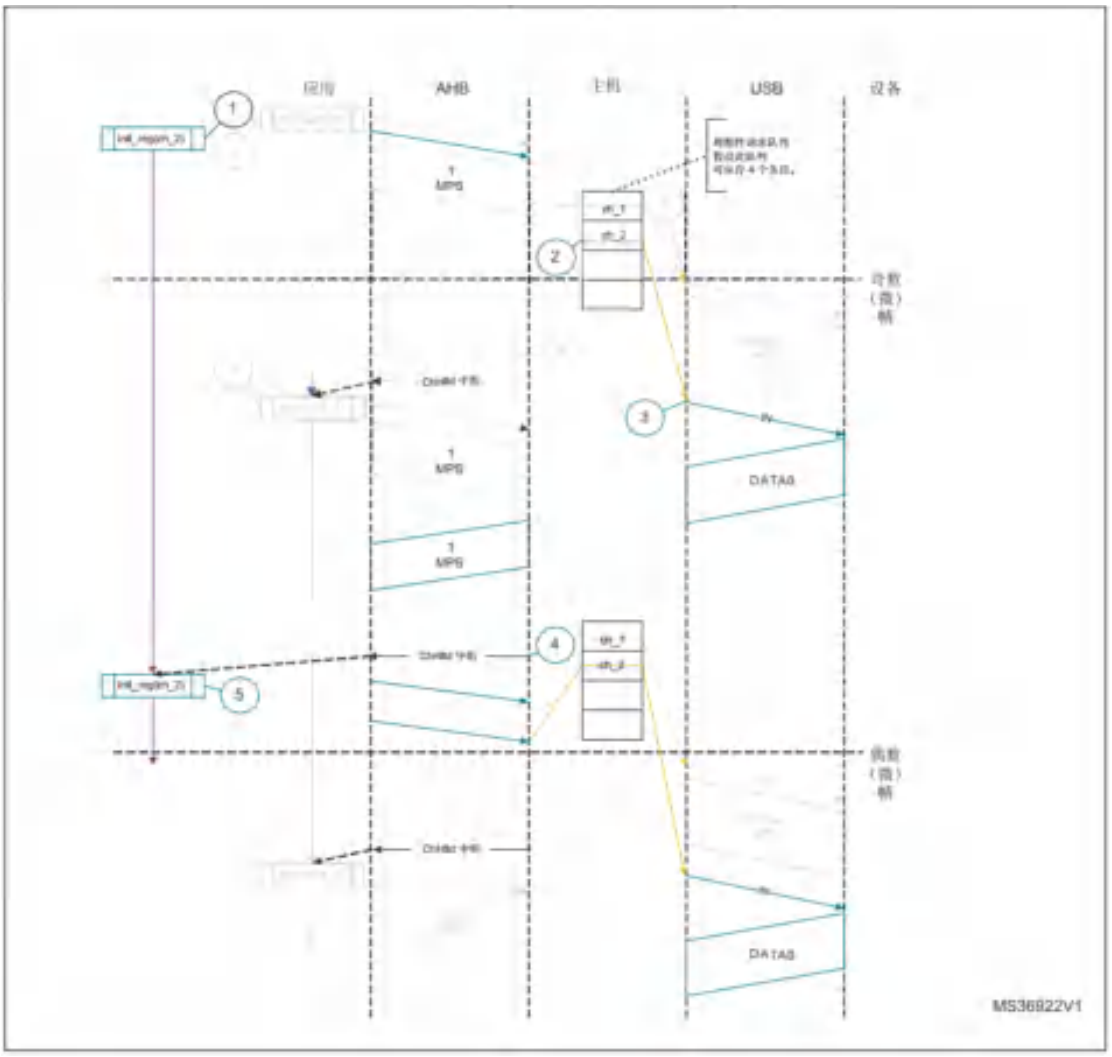
2) 当通道获取了来自仲裁器的授权后（以循环方式执行仲裁），OTG_HS 主机会立即向请求队列写入一个 IN 请求。在高带宽传输模式下，OTG_HS 主机将执行连续写入操作，直到写入的数据包个数达到 MC。

3) OTG_HS 主机尝试在下一个（奇数）帧/微帧的起始位置发送 IN 令牌。

4) 接收到 IN 数据包并写入接收 FIFO 后，OTG_HS 主机便会立即生成 CHH 中断。

5) 为了响应 CHH 中断，将重新初始化通道以供下一传输操作使用。

图 37-12 正常同步 IN 事务 - DMA 模式



● DMA 模式下的批量和控制 OUT/SETUP 分离事务

（通道 x）中的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x 以用于启动分离。

2) 使能通道后，OTG_HS 主机开始获取第一个数据包以发送，并在获取数据包的最后一个 DWORD 后写入 OUT 请求。

3) 成功发送启动分离后，OTG_HS 主机将生成 CHH 中断。

4) 作为对 CHH 中断的响应，将 HCSPLT1 中的 COMPLSPLT 位置 1，以发送完成分离。

5) 成功发送完成分离后，OTG_HS 主机将生成 CHH 中断。

6) 为了响应 CHH 中断，将会释放通道。

● DMA 模式下的批量/控制 IN 分离事务

通道 x 的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x。

2) 获取了来自仲裁器的授权后，OTG_HS 主机将会向非周期性请求队列写入启动分离请求。写入该请求后，OTG_HS 主机将在内部屏蔽通道 x，不参与仲裁。

3) 发送 IN 令牌后，OTG_HS 主机立即生成 CHH 中断。

4) 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1 并重新使能通道，以发送完成分离令牌。这将使能通道 x，并让它参与后续仲裁。

5) 接收到来自仲裁器的授权后，OTG_HS 主机将会向非周期性请求队列写入完成分离请求。

6) 成功接收到数据包后，OTG_HS 主机将开始向系统存储器写入数据包。

7) 只要接收到的数据包写入系统存储器，OTG_HS 主机即会生成一个 CHH 中断。

8) 为了响应 CHH 中断，将会释放通道。

● DMA 模式下的中断 OUT 分离事务

（通道 x）中的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 1 以用于启动分离。应用程序必须将 HCCHAR1 中的 ODDFRM 位置 1。

2) OTG_HS 主机开始读取数据包。

3) OTG_HS 主机尝试发送启动分离事务。

4) 成功发送启动分离后，OTG_HS 主机将生成 CHH 中断。

5) 作为对 CHH 中断的响应，将 HCSPLT1 中的 COMPLSPLT 位置 1，以发送完成分离。

6) 成功结束完成分离事务后，OTG_HS 主机将生成 CHH 中断。

7) 为了响应 CHH 中断，将会释放通道。

● DMA 模式下的中断 IN 分离事务

（通道 x）中的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x 以用于启动分离。

2) 当通道 x 接收到来自仲裁器的授权后， OTG_HS 主机会立即向请求队列写入一个 IN 请求。

3) OTG_HS 主机尝试在下一个奇数微帧的起始位置发送启动分离 IN 令牌。

4) 成功发送启动分离 IN 令牌后，OTG_HS 主机将生成 CHH 中断。

5) 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1，以发送完成分离。

6) 成功接收到数据包后，OTG_HS 主机即会开始向系统存储器写入数据。

7) 将接收到的数据传输到系统存储器后，OTG_HS 主机会生成 CHH 中断。

8) 为了响应 CHH 中断，将释放或重新初始化通道以供下一启动分离使用。

● DMA 模式下的同步 OUT 分离事务

通道 x 的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x 以用于启动分离（开始）。应用程序必须将 HCCHAR1 中的 ODDFRM 位置 1。编程 MPS 字段。

2) OTG_HS 主机开始读取数据包。

3) 成功发送启动分离后（开始），OTG_HS 主机将生成 CHH 中断。

4) 为了响应 CHH 中断，将重新初始化寄存器以发送启动分离（结束）。

5) 成功发送启动分离后（结束），OTG_HS 主机将生成 CHH 中断。

6) 为了响应 CHH 中断，将会释放通道。

● DMA 模式下的同步 IN 分离事务

通道 x 的操作顺序如下：

1) 按照通道初始化一节中的说明初始化并使能通道 x 以用于启动分离。

2) 当通道 x 接收到来自仲裁器的授权后，OTG_HS 主机会立即向请求队列写入一个 IN 请求。

3) OTG_HS 主机尝试在下一个奇数微帧的起始位置发送启动分离 IN 令牌。

4) 成功发送启动分离 IN 令牌后，OTG_HS 主机将生成 CHH 中断。

5) 作为对 CHH 中断的响应，将 HCSPLT2 中的 COMPLSPLT 位置 1，以发送完成分离。

6) 成功接收到数据包后，OTG_HS 主机即会开始向系统存储器写入数据。

将接收到的数据传输到系统存储器后，OTG_HS 主机会生成 CHH 中断。为了响应 CHH 中断，将释放或重新初始化通道以供下一启动分离使用。

37.14.6. 设备编程模型

■ USB 复位时的端点初始化

1) 为所有 OUT 端点将 NAK 位置 1

- DOEPCTLx 中， SNAK = 1（对于所有 OUT 端点）

2) 使能以下中断位

- DAINTMSK 中， INEP0 = 1（控制 0 IN 端点）
- DAINTMSK 中， OUTEP0 = 1（控制 0 OUT 端点）
- DOEPMSK 中的 STUPM = 1
- DOEPMSK 中的 XFRCM = 1
- DIEPMSK 中的 XFRCM = 1
- DIEPMSK 中的 TOM = 1

3) 为每个 FIFO 设置数据 FIFO RAM

- 对 GRXFSIZ 寄存器进行编程，以能够接收控制传输的 OUT 数据和设置数据。如果未使能阈值，则该寄存器必须至少等于控制端点 0 的 1 个最大数据包大小 + 2 个字（用于控制 OUT 数据包的状态） + 10 个字（用于 SETUP 数据包）。
- 对 DIEPTXF0 寄存器进行编程（取决于所选的 FIFO 编号），以能够发送控制 IN 数据。该寄存器至少必须等于控制端点 0 的 1 个最大数据包大小。

4) 对端点相关寄存器中的以下字段进行编程，以使控制 OUT 端点 0 接收 SETUP 数据包

- DOEPTSIZ0 中的 STUPCNT = 3（接收最多 3 个连续的 SETUP 数据包）

5) 在 DMA 模式下的 USB OTG_HS，DOEPDMA0 寄存器应具有一个有效的存储器地址，以存储接收到的任何 SETUP 数据包。

此时，接收 SETUP 数据包所需的所有初始化工作便已完成。

■ 枚举完成时的端点初始化

1) 在枚举完成中断（GINTSTS 中的 ENUMDNE）中，读取 DSTS 寄存器以确定设备的枚举速度。

2) 对 DIEPCTL0 中的 MPSIZ 字段进行编程以设置最大数据包大小。该步骤配置控制端点 0。控制端点的最大数据包大小取决于枚举速度。

3) 对于 DMA 模式下的 USB OTG_HS，编程 DOEPCTL0 寄存器来使能控制 OUT 端点 0，以接收 SETUP 数据包。

此时，设备已准备好接收 SOF 数据包并配置为在控制端点 0 上执行控制传输。

■ 收到 SetAddress 命令时的端点初始化

本节介绍了应用程序在 SETUP 数据包中接收到 SetAddress 命令时必须执行的操作。

1) 使用在 SetAddress 命令中接收到的设备地址来对 DCFG 寄存器进行编程

2) 对模块进行编程以发出状态阶段的 IN 数据包

■ 收到 SetConfiguration/SetInterface 命令时的端点初始化

本节介绍了应用程序在 SETUP 包中接收 SetConfiguration 或 SetInterface 命令时必须执行的操作。

1) 接收到 SetConfiguration 命令时，应用程序必须对端点寄存器进行编程，以使用新配置中有效端点的特性来配置这些端点寄存器。

2) 接收到 SetInterface 命令时，应用程序必须对命令指定的端点的端点寄存器进行编程。

3) 在先前配置或其它设置中有效的端点在新的配置或其它设置中无效。必须停用这些无效端点。

4) 使用 DAINTMSK 寄存器使能有效端点的中断，屏蔽无效端点的中断。

5) 为每个 FIFO 设置数据 FIFO RAM。

6) 配置完所有必需的端点后，应用程序必须对模块进行编程以发送状态阶段的 IN 数据包。

此时，设备模块已可以接收和发送任何类型的数据包。

■ 端点激活

本节介绍激活设备端点或者将现有设备端点配置为新类型所需的步骤。

1) 在 DIEPCTLx 寄存器（对于 IN 或双向端点）或 DOEPCTLx 寄存器（对于 OUT 或双向端点）的以下字段中，对所需端点的特性进行编程。

- 最大数据包大小
- USB 活动端点位置 1
- 端点初始数据同步位（对于中断和批量端点）
- 端点类型
- Tx FIFO 编号

2) 激活端点后，模块便开始解码发送到该端点的令牌，并在收到的令牌有效的情况下回复有效握手信号。

■ 端点停用

本节介绍停用现有端点所需的步骤。

1) 在要停用的端点中，将 DIEPCTLx 寄存器（对于 IN 或双向端点）或

DOEPCTLx 寄存器（对于 OUT 或双向端点）中的 USB 活动端点位清零。

2) 停用端点后，模块便会忽略发送到该端点的令牌，从而导致 USB 超时。

注： 应用程序必须满足以下条件才能设置设备模块以处理通信：必须将 GINTMSK 寄存器中的 NPTXFEM 和 RXFLVLM 清零。

■ 操作模型

SETUP 和 OUT 数据传输：

本节介绍了数据 OUT 传输和 SETUP 事务期间的内部数据流和应用程序操作步骤。

● 数据包读取

本节介绍如何从接收 FIFO 读取数据包（ OUT 数据和 SETUP 数据包）。

1) 捕获到 RXFLVL 中断（GINTSTS 寄存器）时，应用程序必须读取接收状态弹出寄存器（GRXSTSP)。

2) 应用程序可以通过写入 RXFLVLM = 0（在 GINTMSK 中）来屏蔽 RXFLVL 中断（在 GINTSTS 中），直到它把数据包从接收 FIFO 中读取出来。

3) 如果已接收数据包的字节计数不是 0，则从接收数据 FIFO 中弹出这些数据并存储在存储器中。如果接收到的数据包字节计数为 0，则不会从接收数据 FIFO 中弹出任何数据。

4) 从接收 FIFO 读出的数据包状态有以下几种状态：

a) 全局 OUT NAK：

PKTSTS = 全局 OUT NAK， BCNT = 0x000， EPNUM = (0x0)， DPID = (0b00)。

这些数据表示全局 OUT NAK 位已生效。

b) SETUP 数据包：

PKTSTS = SETUP， BCNT = 0x008， EPNUM = 控制端点编号， DPID = DATA0。

这些数据表示指定端点上收到的 SETUP 数据包现在可从接收 FIFO 中读取。

c) 建立阶段完成：

PKTSTS = 建立阶段完成， BCNT = 0x0， EPNUM = 控制端点编号，

DPID = (0b00)。

这些数据表示指定端点的建立阶段完成并且数据阶段已启动。在此状态条目从接收 FIFO 中弹出后，模块将在该控制 OUT 端点上产生建立中断。

d) OUT 数据包：

PKTSTS = DataOUT， BCNT = 接收的 OUT 数据包的大小 (0 ≤ BCNT ≤ 1 024)，EPNUM = 收到数据包的端点编号， DPID = 实际数据 PID。

e) 数据传输完成：

PKTSTS = OUT 数据传输完成， BCNT = 0x0， EPNUM = 完成数据传输的 OUT

端点编号， DPID = (0b00)。

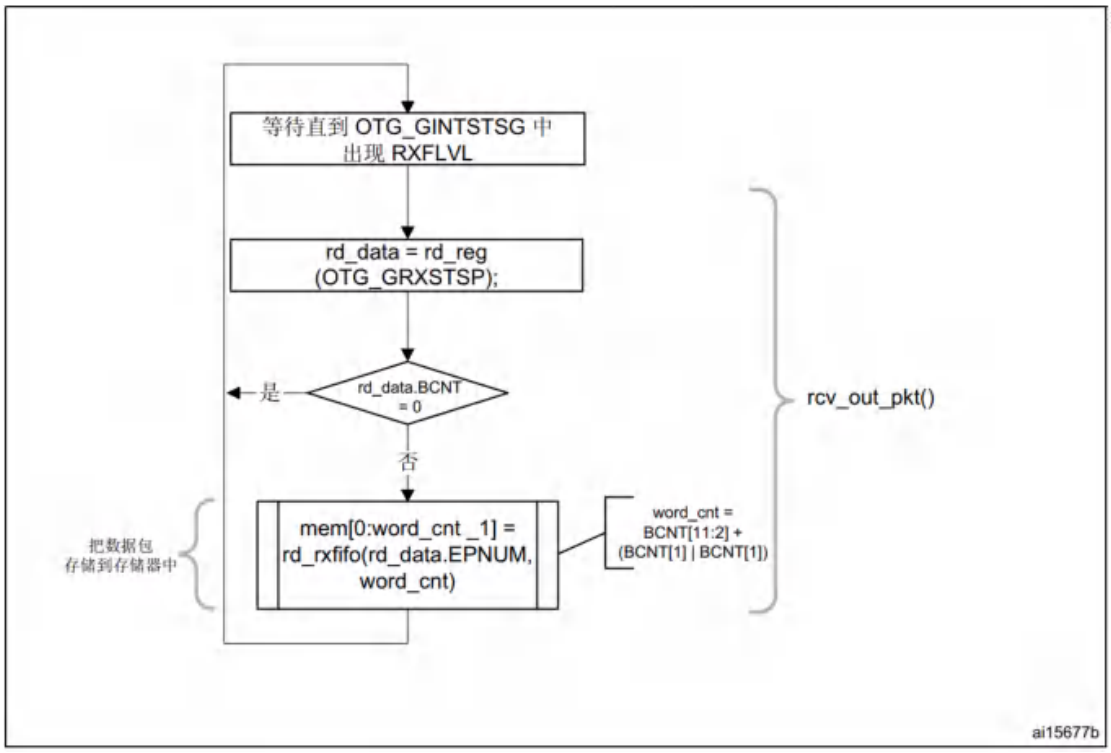
这些数据表示指定 OUT 端点的 OUT 数据传输完成。在此状态条目从接收 FIFO 中弹出后，模块将在指定的 OUT 端点上引发“传输完成”中断。

5) 从接收 FIFO 中弹出数据后，必须使能 RXFLVL 中断 (GINTSTS)。

6) 每次应用程序检测到 GINTSTS 中的 RXFLVL 中断时，都将重复步骤 1 到 5。读取空的接收 FIFO 可能导致未定义的模块行为。

图 761 提供了上述过程的流程图。

图 37-13 接收 FIFO 数据包读取



■ SETUP 事务

本节介绍了模块处理 SETUP 数据包的方式以及应用程序处理 SETUP 事务的顺序。

● 应用程序要求

1) 要接收 SETUP 数据包，必须将控制 OUT 端点中的 STUPCNT 字段（DOEPTSIZx）编程为非零值。如果应用程序将 STUPCNT 字段编程为非零值，模块会接收 SETUP 数据包并将其写入接收 FIFO，而不考虑 NAK 状态和 DOEPCTLx 中的 EPENA 位设置。控制端点每收到一个 SETUP 数据包后，STUPCNT 字段都会递减。如果在接收 SETUP 数据包之前，未将 STUPCNT 字段编程为适当值，模块仍能接收 SETUP 数据包并使 STUPCNT 字段递减，但应用程序可能无法确定在控制传输的建立阶段中接收的 SETUP 数据包正确数量。

- 在 DOEPTSIZx 中， STUPCNT = 3

2)应用程序必须始终在接收数据 FIFO 中分配一些额外空间，以便能够在控制端点上接收连续的最多三个 SETUP 数据包。

- 预留空间 10 个字。第一个 SETUP 数据包需要 3 个字，“建立阶段完成”状态双字需要 1 个字，还需要 6 个字以存储两个额外的 SETUP 数据包。
- 每个 SETUP 数据包需要 3 个字以存储 8 个字节的 SETUP 数据和 4 个字节的 SETUP 状态。模块将在接收 FIFO 中保留这些空间。
- 这段 FIFO 仅用于存储 SETUP 包，绝对不会将该空间用于数据包。

3) 应用程序必须从接收 FIFO 中读取 SETUP 数据包的 2 个字。

4) 应用程序必须从接收 FIFO 中读取并丢弃“建立阶段完成”状态字。

● 内部数据流

1) 接收到 SETUP 数据包时，模块会将接收到的数据写入接收 FIFO，而不会检查接收 FIFO 中的可用空间，且不考虑端点的 NAK 和 STALL 位设置。

- 模块会在内部将接收到 SETUP 数据包的控制 IN/OUT 端点的 IN NAK 和 OUT NAK 位置 1。

2) USB 上接收到的每个 SETUP 数据包，模块会将 3 个字的数据写入接收 FIFO，并且将 STUPCNT 字段递减 1。

- 第一个字包含由模块所使用的内部控制信息
- 第二个字包含 SETUP 命令的前 4 个字节
- 第三个字包含 SETUP 命令的最后 4 个字节

3) 当建立阶段结束，数据 IN/OUT 阶段开始时，模块会将一个状态条目（“建立阶段完成”字）写入接收 FIFO，指示建立阶段完成。

4) 在 AHB 端， SETUP 数据包被应用程序读取。

5) 当应用程序从接收 FIFO 中弹出“建立阶段完成”字时，模块将使用 STUP 中断（DOEPINTx）来中断应用程序，指示其可以处理接收到的 SETUP 数据包。

6) 模块会将控制 OUT 端点的端点使能位清零。

● 应用程序编程顺序

1) 对 DOEPTSIZx 寄存器进行编程。

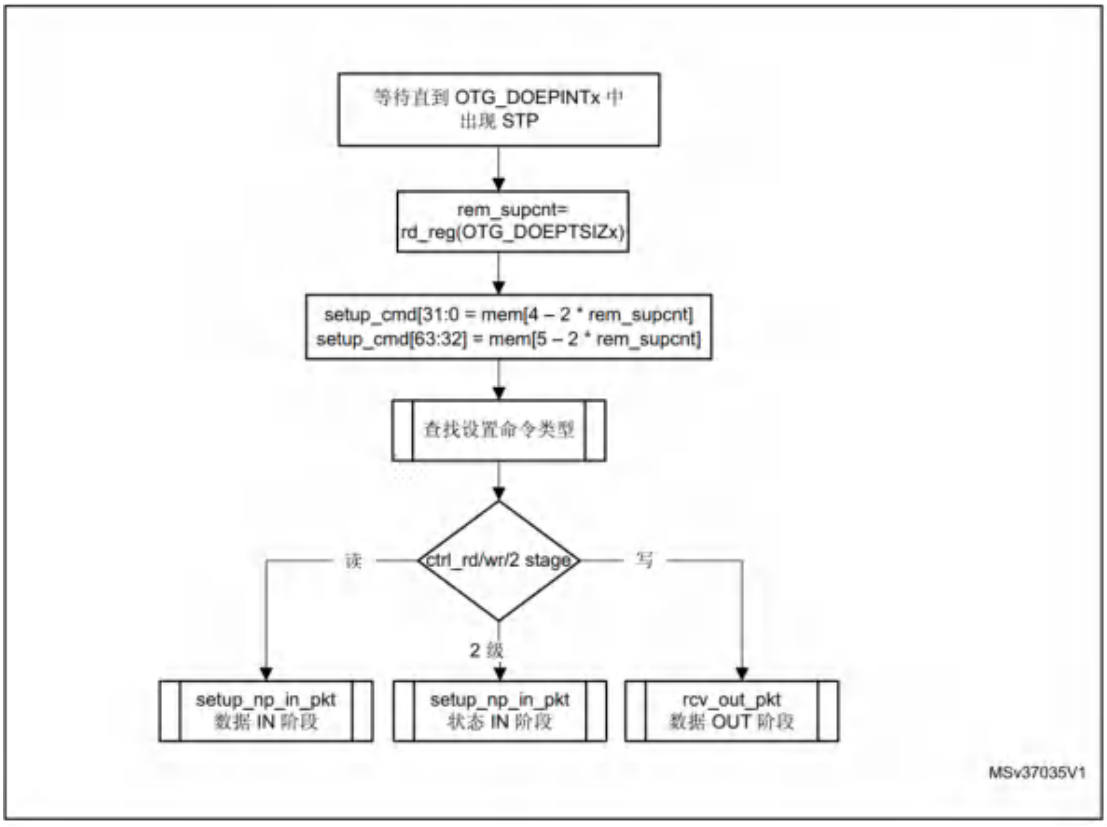
- STUPCNT = 3

2) 等待 RXFLVL 中断（GINTSTS）并且从接收 FIFO 中读取数据包。

3) STUP 中断的触发（DOEPINTx）表示 SETUP 数据传输成功完成。

- 发生该中断时， 应用程序必须读取 DOEPTSIZx 寄存器以确定接收的 SETUP 数据包数量并处理最后接收的 SETUP 数据包。

图 37-14 处理 SETUP 数据包



● 处理三个以上连续的 SETUP 数据包

根据 USB 2.0 规范，在 SETUP 数据包错误中，主机通常不会向同一个端点发送 3 个以上连续的 SETUP 数据包。但是， USB 2.0 规范并未限制主机可以向同一个端点发送的连续 SETUP 数据包数量。出现这种情况时， OTG_HS 控制器将生成中断（DOEPINTx 中的 B2BSTUP）。

● 将全局 OUT NAK 置 1

内部数据流：

1) 如果应用程序将全局 OUT NAK（DCTL 中的 SGONAK 位）置 1，模块将停止向接收 FIFO 中写入 SETUP 数据包以外的数据。无论接收 FIFO 中可用空间大小如何，设备都会对主机发送的非同步 OUT 令牌回复 NAK 握手，而对同步 OUT 数据包直接予以忽略。

2) 模块将全局 OUT NAK 写入接收 FIFO。应用程序必须为此留出足够空间。

3) 当应用程序从接收 FIFO 中弹出全局 OUT NAK 字时，模块会将 GONAKEFF 中断（GINTSTS）置 1。

4) 应用程序检测到该中断后，会认为模块处于全局 OUT NAK 模式。应用程序可以通过将 DCTL 中的 SGONAK 位清零来清除该中断。

应用程序编程顺序：

1) 要停止接收任何类型的数据到接收 FIFO 中，应用程序必须通过编程以下字段以将全局 OUT NAK 位置 1。

- 在 DCTL 中， SGONAK = 1

2) 等待 GINTSTS 中的 GONAKEFF 中断。一旦被触发，该中断表示模块已停止接收 SETUP 数据包以外的任何类型数据。

3) 如果应用程序已将 DCTL 中的 SGONAK 位置 1，则在模块引发 GONAKEFF 中断（GINTSTS）之前，应用程序可以接收有效 OUT 数据包。

4) 应用程序可通过对 GINTMSK 寄存器中的 GONAKEFFM 位执行写操作来暂时屏蔽此中断。

- 在 GINTMSK 寄存器中， GONAKEFFM = 0

5) 当应用程序准备退出全局 OUT NAK 模式时，必须将 DCTL 中的 SGONAK 位清零。此操作还会清除 GONAKEFF 中断（GINTSTS）。

- 在 DCTL 中， CGONAK = 1

6) 如果应用程序在之前已屏蔽此中断，则必须按以下方式使能该中断：

- 在 GINTMSK 中， GONAKEFFM = 1

● 禁止 OUT 端点

应用程序必须使用以下顺序禁止已使能的 OUT 端点。

应用程序编程顺序：

1) 禁止任何 OUT 端点前，应用程序必须在模块中使能全局 OUT NAK 模式。

- 在 DCTL 中， SGONAK = 1

2) 等待 GONAKEFF 中断（GINTSTS）

3) 通过编程以下字段来禁止 OUT 端点：

- 在 DOEPCTLx 中， EPDIS = 1
- 在 DOEPCTLx 中， SNAK = 1

4) 等待 EPDISD 中断（DOEPINTx），该中断表示已完全禁止 OUT 端点。引发 EPDISD 中断时，模块还会将以下位清零：

- 在 DOEPCTLx 中， EPDIS = 0
- 在 DOEPCTLx 中， EPENA = 0

5) 应用程序必须将全局 OUT NAK 位清零，以开始从其它未禁止的 OUT 端点接收数据。

- 在 DCTL 中， SGONAK = 0

● 通用非同步 OUT 数据传输

本节介绍一种常规非同步 OUT 数据传输（控制、批量或中断）。

应用程序要求：

1) 建立 OUT 传输前，应用程序必须在存储器中分配一个缓冲区，以容纳要作为 OUT 传输的一部分而接收的所有数据。

2) 对于 OUT 传输，端点的传输大小寄存器中的传输大小字段必须是端点的最大数据包大小的倍数（且以字对齐）。

- 传输大小[EPNUM] = n × (MPSIZ[EPNUM] + 4 − (MPSIZ[EPNUM] mod 4))
- 数据包计数[EPNUM] = n
- n > 0

3) 发生 OUT 端点中断时，应用程序必须读取端点的传输大小寄存器以计算存储器中有效数据量。接收的有效数据量可能小于编程的传输大小。

- 存储器中的有效数据量 = 应用程序设置的初始传输量 − 模块更新后的剩余传输量
- 接收到 USB 数据包数 = 应用程序设置的初始数据包数 − 模块更新后的剩余数据包数

内部数据流：

1) 应用程序必须在端点相关寄存器中设置传输大小和数据包计数字段，将 NAK 位清零，并使能端点来接收数据。

2) NAK 位清零后，模块便开始接收数据并将数据写入接收 FIFO（只要接收 FIFO 中有空间）。对于 USB 上接收的每个数据包，数据包及其状态都会写入接收 FIFO。写入接收 FIFO 的每个数据包（数据量达到最大数据包大小的数据包或短数据包）都会使该端点的数据包计数字段递减 1。

- 收到的数据包若 CRC 无效，则自动被从接收 FIFO 中清除。
- 在 USB 上为数据包回复 ACK 后，模块将丢弃主机因无法检测到 ACK 而重新发送的非同步 OUT 数据包。应用程序不会在具有相同数据 PID 的相同端点上检测到多个连续的 OUT 数据包。在这种情况下，数据包计数不会递减。
- 如果接收 FIFO 中没有空间，则会忽略同步或非同步数据包并且不会将它们写入接收 FIFO。此外，非同步 OUT 令牌将会收到 NAK 握手应答。
- 在上述所有三种情况中，数据包计数都不会递减，因为没有任何数据写入接收 FIFO。

3) 当数据包计数变为 0 或者在端点上接收到短数据包时，该端点的 NAK 位将置 1。 NAK 位置 1 后，将忽略同步或非同步数据包并且不会将它们写入接收 FIFO，同时非同步 OUT 令牌会收到 NAK 握手应答。

4) 在数据写入接收 FIFO 后，应用程序将从接收 FIFO 中读取数据并将数据写入外部存储器，一次一个数据包，逐个端点过来。

5) 在 AHB 上向外部存储器写入完每个数据包后，端点的传输大小都会自动减去该数据包的大小。

6) 在以下情况时， OUT 端点的 OUT 数据传输完成状态将写入接收 FIFO：

- 传输大小为 0 并且数据包计数为 0
- 写入接收 FIFO 的最后一个 OUT 数据包是短数据包
- （ 0 ≤ 数据包大小 < 最大数据包大小）

7) 当应用程序弹出此状态条目（ OUT 数据传输完成），并生成该端点的传输完成中断，同时清零端点使能位。

应用程序编程顺序：

1) 使用传输大小和相应数据包个数对 DOEPTSIZx 寄存器进行编程。

2) 使用端点特性对 DOEPCTLx 寄存器进行编程，并将 EPENA 和 CNAK 位置 1。

- 在 DOEPCTLx 中， EPENA = 1

➢ 在 DOEPCTLx 中， CNAK = 1

3) 等待 RXFLVL 中断（在 GINTSTS 中）并且从接收 FIFO 中读走数据包。

➢ 此步骤可重复多次，具体取决于传输大小。

4) 触发 XFRC 中断（DOEPINTx），以表示非同步 OUT 数据传输成功完成。

5) 读取 DOEPTSIZx 寄存器，以确定有效数据量。

● 通用同步 OUT 数据传输

本节介绍常规的同步 OUT 数据传输。

应用程序要求：

1) 非同步 OUT 数据传输的所有应用程序要求均适用于同步 OUT 数据传输。

2) 对于同步 OUT 数据传输中的传输大小和数据包计数字段，必须始终将其设置为单个帧中可接收的最大数据包大小的数据包数目。同步类型的 OUT 数据传输事务必须在一个帧内完成。

3) 在周期性帧结束（GINTSTS 中的 EOPF 中断）之前，应用程序必须从接收 FIFO 中读取所有同步 OUT 数据包（数据条目和状态条目）。

4) 要接收下一帧中的数据，必须在 EOPF(GINTSTS)之后 SOF(GINTSTS)之前使能一个同步 OUT 端点。

内部数据流：

1) 同步 OUT 端点的内部数据流与非同步 OUT 端点的基本相同，但稍有差异。

2) 同步 OUT 端点通过将端点使能位置 1 并将 NAK 位清零来使能时，必须相应地将偶数/奇数帧位置 1。仅当符合以下条件时，模块才会在同步 OUT 端点上接收特定帧中的数据：

➢ EONUM （在 DOEPCTLx 中） = FNSOF[0]（在 DSTS 中）

3) 当应用程序从接收 FIFO 中完整地读取一个同步 OUT 数据包（数据和状态）后，模块会根据从接收 FIFO 中读取的最后一个同步 OUT 数据包的数据 PID 更新 DOEPTSIZx 中的 RXDPID 字段。

应用程序编程顺序：

1) 使用传输大小和相应数据包个数对 DOEPTSIZx 寄存器进行编程

2) 使用端点特性对 DOEPCTLx 寄存器进行编程，并将端点使能位、清除 NAK 位和奇数/偶数帧位置 1。

➢ EPENA = 1

➢ CNAK = 1

➢ EONUM = （ 0：偶数/1：奇数）

3) 等待 RXFLVL 中断（在 GINTSTS 中）并且从接收 FIFO 中读走数据包

➢ 此步骤可重复多次，具体取决于传输大小。

4) XFRC 中断（在 DOEPINTx 中）表示同步 OUT 数据传输完成。该中断不一定意味着存储器中的数据是有效的。

5) 对于同步 OUT 传输，应用程序可能并不总会检测到该中断。而是可以检测到

➢ GINTSTS 中的 INCOMPISOOUT 中断。

6) 读取 DOEPTSIZx 寄存器以确定接收的传输大小以及帧中所接收数据的有效性。仅当符合以下条件之一时，应用程序才必须将存储器中接收的数据视为有效数据：

➢ RXDPID = DATA0（在 DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 1

➢ RXDPID = DATA1（在 DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 2

➢ RXDPID = D2（在 DOEPTSIZx 中）并且接收该有效数据的 USB 数据包数量 = 3

接收该有效数据的 USB 数据包数量 = 应用程序编程的初始数据包个数 - 模块更新后的剩余数据包个数，应

用程序可将无效数据包丢弃。

● 不完整的同步 OUT 数据传输

本节介绍了同步 OUT 数据包出现丢包时应用程序编程顺序。

内部数据流：

1) 对于同步 OUT 端点，可能并不总是触发 XFRC 中断（在 DOEPINTx 中）。如果模块丢弃同步 OUT 数据包，则在以下情况下，应用程序可能无法检测到 XFRC 中断（DOEPINTx）：

- 在接收 FIFO 无法容纳完整的 ISO OUT 数据包时，模块将丢弃接收到的 ISO OUT 数据
- 接收到的同步 OUT 数据包存在 CRC 错误
- 模块接收到的同步 OUT 令牌损坏
- 应用程序从接收 FIFO 中读取数据的速度非常缓慢

2) 如果模块在所有同步 OUT 端点的传输完成前检测到周期性帧结束，将触发未完成同步 OUT 数据中断（GINTSTS 中的 INCOMPISOOUT），指示至少有一个同步 OUT 端点上未触发 XFRC 中断（在 DOEPINTx 中）。此时，未完成传输的端点仍保持使能，但在 USB 的该端点上，没有进行中的有效传输。

应用程序编程顺序：

1) 硬件触发 INCOMPISOOUT 中断（GINTSTS）表示当前帧中至少有一个同步 OUT 端点具有未完成的传输。

2) 如果因未从端点完全读取同步 OUT 数据而发生这种情况，应用程序必须确保首先从接收 FIFO 读取走所有同步 OUT 数据（包括数据条目和状态条目），然后再继续处理。

- 从接收 FIFO 读取所有数据后，应用程序即可检测到 XFRC 中断（DOEPINTx）。

在此情况下，应用程序必须重新使能端点以接收下一个帧中的同步 OUT 数据。

3) 当应用程序接收到 INCOMPISOOUT 中断（在 GINTSTS 中）时，应用程序必须读取所有同步 OUT 端点的控制寄存器（DOEPCTLx），以确定哪些端点在当前微帧中具有不完整的传输。同时满足以下两个条件时，表示端点传输未完成：

- EONUM 位（在 DOEPCTLx 中） = FNSOF[0]（在 DSTS 中）
- EPENA = 1（在 DOEPCTLx 中）

4) 在检测到 SOF 中断（在 GINTSTS 中）前，必须执行完成上一步操作，以确保当前帧编号未发生更改。

5) 对于具有未完成传输的同步 OUT 端点，应用程序必须丢弃存储器中的数据，并通过将 DOEPCTLx 中的 EPDIS 位置 1 来禁止该端点。

6) 等待 EPDISD 中断（在 DOEPINTx 中），并使能该端点，以便接收下一个帧中的新数据。

- 由于模块可能需要一些时间才能禁止端点，因此应用程序在接收到无效同步数据后，可能无法接收下一个帧中的数据。

● 停止非同步 OUT 端点

本节介绍应用程序如何才能停止非同步端点。

1) 将模块置于全局 OUT NAK 模式。

2) 禁止所需的端点

- 禁止端点时，请设置 STALL = 1（在 DOEPCTL 中），而不是将 DOEPCTL 中的 SNAK 位置 1。STALL 位的优先级始终高于 NAK 位。

3) 当应用程序不再需要端点回复 STALL 握手信号时，必须将 STALL 位（在 DOEPCTLx 中）清零。

4) 如果应用程序由于收到主机的 SetFeature.Endpoint Halt 或 ClearFeature.Endpoint Halt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点上的状态阶段传输前，将 STALL 位置 1 或清零。

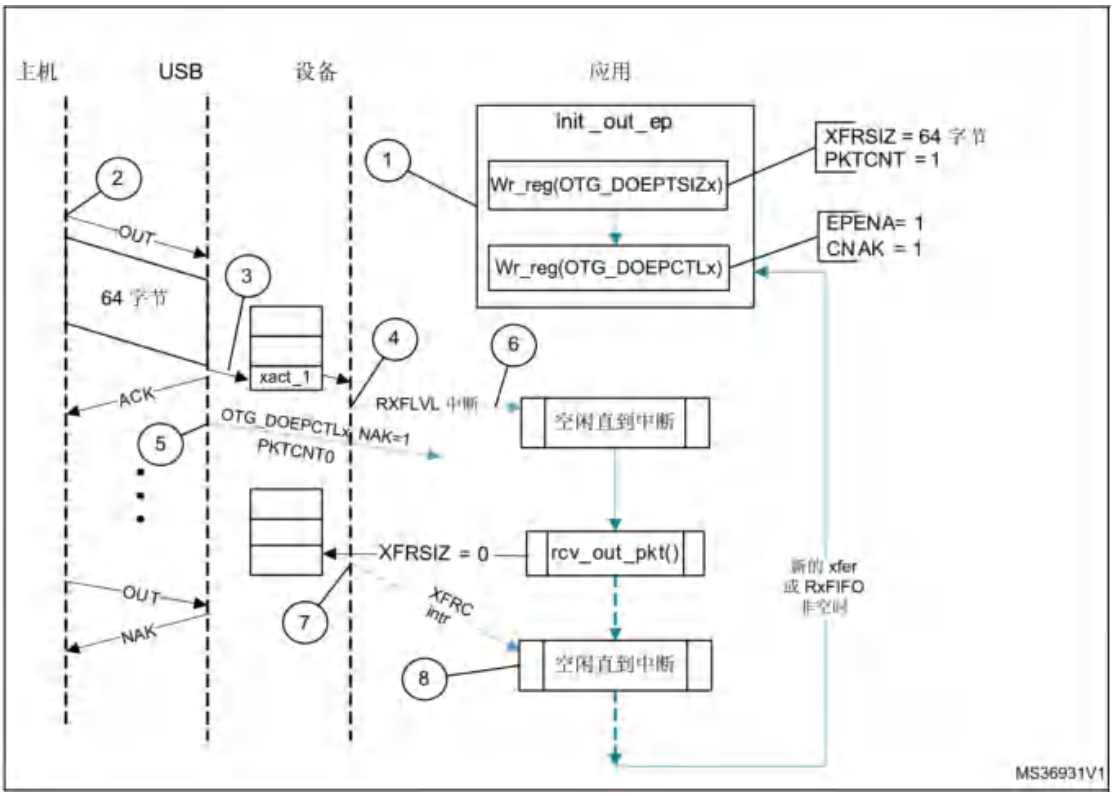
■ 示例

本节介绍并描述了一些基本的传输类型和情景。

● 批量 OUT 事务

图 763 描述了将单个批量 OUT 数据包从 USB 接收到 AHB 中的过程，并介绍了这一过程中涉及到的事件。

图 37-15 批量 OUT 事务



接收到 SetConfiguration/SetInterface 命令后，应用程序将初始化所有 OUT 端点：将 CNAK 和 EPENA 置 1（在 DOEPCTLx 中），并将 DOEPTSIZx 寄存器中的 XFRSIZ 和 PKTCNT 设置成合适的值。

1) 主机尝试将数据（ OUT 令牌）发送到端点。

2) 当模块在 USB 上接收到 OUT 令牌时，模块会将数据包存储在 Rx FIFO 中，因为其中有可用空间。

3) 在 Rx FIFO 中写入完整数据包后，模块随后会触发 RXFLVL 中断（在 GINTSTS 中）。

4) 接收到 PKTCNT 所指定数量的 USB 数据包后，模块在内部将该端点的 NAK 位置 1，以避免其再接收任何数据包。

5) 应用程序处理中断并从 Rx FIFO 中读取数据。

6) 应用程序读取完所有数据后（相当于 XFRSIZ），模块将生成 XFRC 中断（在 DOEPINTx 中）。

7) 应用程序处理中断并通过 XFRC 中断位（在 DOEPINTx 中）的触发得知本次传输完成。

■ IN 数据传输

● 数据包写入

本节介绍在已使能专用发送 FIFO 的情况下应用程序如何将数据包写入端点 FIFO。

1) 应用程序可以选择轮询模式或中断模式。

- 在轮询模式下， 应用程序通过读取 DTXFSTSx 寄存器来监视端点发送数据 FIFO 的状态，从而确定该数据 FIFO 中是否有足够空间。
- 在中断模式中，应用程序等待 TXFE 中断（在 DIEPINTx 中），然后读取 DTXFSTSx 寄存器，以确定数据

FIFO 中是否有足够空间。

- 要写入单个非零长度的数据包，数据 FIFO 中必须有足够的空间来容纳整个数据包。
- 要写入零长度的数据包，应用程序不能查看 FIFO 空间。

2) 如果使用上述方法之一，当应用程序确定有足够空间来写入发送数据包时，应用程序必须首先对端点控制寄存器进行相应写操作，然后再将数据写入数据 FIFO。通常，应用程序必须对 DIEPCTLx 寄存器执行读 - 修改 - 写操作，以避免在将端点使能位置 1 的同时，修改寄存器中的其它内容。

如果有足够空间，应用程序可将同一端点的多个数据包写入发送 FIFO。对于周期性 IN 端点，应用程序只能一次写入一个微帧内的多个数据包。只有先前一个微帧的通信事务传输完成之后，应用程序才会写入下一个微帧内要发送的所有数据包。

● 将 IN 端点 NAK 置 1

内部数据流：

1) 当应用程序将特定端点的 IN NAK 置 1 时，模块将停止端点上的数据发送，而不考虑端点发送 FIFO 中的数据是否可用。

2) 非同步端点收到 IN 令牌，回复 NAK 握手应答

- 同步端点收到 IN 令牌，返回零长度数据包

3) 模块在 DIEPINTx 中触发 INEPNE（IN 端点 NAK 有效）中断以响应 DIEPCTLx 中的 SNAK 位。

4) 应用程序检测到该中断后，便会认为端点处于 IN NAK 模式。应用程序可以通过将 DIEPCTLx 中的 CNAK 位置 1 来清除该中断。

应用程序编程顺序：

1) 要在特定 IN 端点上停止发送任何数据，应用程序必须将 IN NAK 位置 1。要将该位置 1，必须编程以下字段。

- 在 DIEPCTLx 中， SNAK = 1

2) 等待 DIEPINTx 中的 INEPNE 中断触发。该中断表示模块已在端点上停止发送数据。

3) 在应用程序将 NAK 位置 1 但“NAK 有效”中断尚未触发时，模块可以在端点上发送有效 IN 数据。

4) 应用程序可通过写入 DIEPMSK 中的 INEPNEM 位来临时屏蔽该中断。

- 在 DIEPMSK 中， INEPNEM = 0

5) 要退出端点 NAK 模式，应用程序必须将 DIEPCTLx 中的 NAK 状态位（NAKSTS）清零。此操作还会将 INEPNE 中断位（在 DIEPINTx 中）清零。

- 在 DIEPCTLx 中， CNAK = 1

6) 如果应用程序已将该中断屏蔽，则必须按以下方式使能：

- 在 DIEPMSK 中， INEPNEM = 1

● 禁止 IN 端点

使用以下顺序来禁止先前已使能的特定 IN 端点。

应用程序编程顺序：

1) 应用程序必须先停止在 AHB 上写入数据，之后才能禁止 IN 端点。

2) 应用程序必须将端点设置为 NAK 模式。

- 在 DIEPCTLx 中， SNAK = 1

3) 等待 DIEPINTx 中的 INEPNE 中断。

4) 将必须禁止的端点的 DIEPCTLx 寄存器中的以下位置 1。

- 在 DIEPCTLx 中， EPDIS = 1
- 在 DIEPCTLx 中， SNAK = 1

5) DIEPINTx 中的 EPDISD 中断的触发表示模块已完全禁止指定的端点。在触发中断的同时，模块还会将以下位清零：

- 在 DIEPCTLx 中， EPENA = 0
- 在 DIEPCTLx 中， EPDIS = 0

6) 应用程序必须读取周期性 IN EP 的 DIEPTSIZx 寄存器，以计算该端点在 USB 上发送的数据量。

7) 应用程序必须通过将 GRSTCTL 寄存器中的以下字段置 1，来清空端点发送 FIFO 中的数据：

- TXFNUM（在 GRSTCTL 中） = 端点发送 FIFO 编号
- TXFFLSH（在 GRSTCTL 中） = 1

应用程序必须轮询 GRSTCTL 寄存器，直至模块将 TXFFLSH 位清零，这表示 FIFO 清空操作结束。要在该端点上发送新数据，应用程序可以稍后重新使能该端点。

● 通用非周期性 IN 数据传输

应用程序要求：

1) 建立 IN 传输前，应用程序必须确保组成一次 IN 传输的每个数据包都可以容纳在单个缓冲区中。

2) 对于 IN 传输，端点传输大小寄存器中的传输大小字段表示本次传输的有效数据量，它由多个最大数据包大小和单个短数据包组成。该短数据包在传输结束时发送。

- 要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包：

传输大小[EPNUM] = x × MPSIZ[EPNUM] + sp

如果 (sp > 0)，数据包计数[EPNUM] = x + 1。

否则，数据包计数[EPNUM] = x

- 要发送单个零长度数据包：

传输大小[EPNUM] = 0

数据包计数[EPNUM] = 1

- 要发送多个最大数据包大小的数据包并在传输结束时外加一个零长度数据包，应用程序必须将传输拆分为两个部分。第一部分发送最大数据包大小的数据包，第二部分仅发送零长度数据包。

第一次传输：传输大小[EPNUM]=x×MPSIZ[epnum]；数据包计数= n；

第二次传输：传输大小[EPNUM]=0；数据包计数=1；

3) 使能某个端点进行数据传输后，模块会更新传输大小寄存器。在 IN 传输结束时，应用程序必须读取传输大小寄存器，以确定送入发送 FIFO 中的数据已有多少通过 USB 发送出去。

4) 送入发送 FIFO 中的数据量 = 应用程序编程的初始传输大小 - 模块更新后的最终传输大小。

- 通过 USB 已经发送的数据量 = (应用程序编程的初始数据包计数 - 模块更新后的最终数据包计数) × MPSIZ[EPNUM]
- 要通过 USB 发送的剩余数据量 = (应用程序编程的初始传输大小 - 已通过 USB 发送的数据量)

内部数据流：

1) 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。

2) 应用程序还必须向该端点的发送 FIFO 写入必需的数据。

3) 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。向 FIFO 写入数据后，“FIFO 中的数据

包数”计数会递增（这是一个 3 位计数，由模块在内部进行维护，每个 IN 端点发送 FIFO 对应一个。在 IN 端点 FIFO 中，模块所维护的最大数据包数始终为八个）。对于零长度数据包，每个 FIFO 均另有一个单独的标志， FIFO 中没有任何数据。

4) 当数据写入发送 FIFO 后，模块会在接收到 IN 令牌时将这些数据送出。每个非同步 IN 数据包发送出去并收到回复的 ACK 握手信号后，该端点的数据包计数都会递减 1，直到数据包计数变 0 为止。发生超时时，数据包计数不会递减。

5) 对于零长度数据包（由内部零长度标志指示），模块会针对 IN 令牌发出一个零长度数据包，并递减数据包计数字段的值。

6) 如果接收到 IN 令牌的端点对应的 FIFO 中无数据，且该端点的数据包计数字段为零，则模块会针对该端点生成一个“Tx FIFO 为空时接收到 IN 令牌” (ITTXFE) 中断（前提是该端点的 NAK 位未置 1）。模块在该非同步端点上回复 NAK 握手信号。

7) 模块会在内部使 FIFO 指针重新返回到开头，并且不会生成超时中断。

8) 当传输大小为 0 且数据包计数为 0 时，将生成该端点的传输完成 (XFRC) 中断，同时将端点使能清零。

应用程序编程顺序：

1) 使用传输大小和相应数据包计数对 DIEPTSIZx 寄存器进行编程。

2) 使用端点特性对 DIEPCTLx 寄存器进行编程，并将 CNAK 和 EPENA（端点使能）位置 1。

3) 发送非零长度数据包时，应用程序必须轮询 DTXFSTSx 寄存器（其中 x 为与该端点相关联的 FIFO 编号），以确定数据 FIFO 中是否有足够的空间。写入数据前，应用程序可以选择使用 TXFE（在 DIEPINTx 中）。

● 通用周期性 IN 数据传输

本节介绍典型的周期性 IN 数据传输。

应用程序要求：

1) 第 2515 页的通用非周期性 IN 数据传输的应用程序要求 1、 2、 3 和 4 对周期性 IN 数据传输同样适用（只是对要求 2 稍加修改）。

- 应用程序只能发送若干个最大数据包大小的数据包或若干个最大数据包大小的包，外加传输结束时的一个短数据包。要发送多个最大数据包大小的数据包并在传输结束时外加一个短数据包，必须满足以下条件：

传输大小[EPNUM] = x × MPSIZ[EPNUM] + sp

（其中 x 是 ≥ 0 的整数，且 0 ≤ sp < MPSIZ[EPNUM]）

如果 (sp > 0)，数据包计数[EPNUM] = x + 1

否则，数据包计数[EPNUM] = x;

MCNT[EPNUM] = 数据包计数[EPNUM]

- 应用程序无法在传输结束时发送零长度数据包。应用程序可以单独发送一个零长度数据包。要发送单个零长度数据包：

传输大小[EPNUM] = 0

数据包计数[EPNUM] = 1

MCNT[EPNUM] = 数据包计数[EPNUM]

2) 应用程序一次只能安排一帧的数据传输。

- (MCNT − 1) × MPSIZ ≤ XFERSIZ ≤ MCNT × MPSIZ
- PKTCNT = MCNT（在 DIEPTSIZx 中）
- 如果 XFERSIZ < MCNT × MPSIZ，则传输的最后一个数据包为短数据包。

- 注意： MCNT 在 OTG_DIEPTSIZx 中， MPSIZ 在 DIEPCTLx 中， PKTCNT 在 DIEPTSIZx 中，而 XFERSIZ 在 DIEPTSIZx 中

3) 接收到 IN 令牌前，应用程序必须将要在帧中发送的完整数据写入到发送 FIFO 中。在接收到 IN 令牌时，即使发送 FIFO 中该帧要发送的数据只差 1 个双字未写进来，模块也会执行 FIFO 为空时的操作。当发送 FIFO 为空时：

- 同步端点上将回复零长度数据包
- 中断端点上将回复 NAK 握手信号

内部数据流：

1) 应用程序必须在特定端点的寄存器中设置传输大小和数据包计数字段，并使能该端点来发送数据。

2) 应用程序还必须向与该端点相关联的发送 FIFO 写入必需的数据。

3) 应用程序每向发送 FIFO 写入一个数据包，该端点的传输大小便会自动减去该数据包大小。应用程序持续从存储器获取数据来写入发送 FIFO，直到该端点的传输大小变为 0。

4) 当周期性端点接收到 IN 令牌时，模块将开始发送 FIFO 中的数据（如果 FIFO 中有数据）。如果 FIFO 中没有该帧要发送的数据的完整数据包，则模块将为该端点生成一个“TxFIFO 为空时接收到 IN 令牌”中断。

- 同步端点上将回复零长度数据包
- 中断端点上将回复 NAK 握手信号

5) 端点的数据包计数会在下列情况下递减 1：

- 对于同步端点，发送一个零长度或非零长度的数据包时
- 对于中断端点，在发送 ACK 握手信号时递减
- 当传输大小和数据包计数均为 0 时，将生成该端点的传输完成中断，同时将端点使能位清零。

6) 在“周期性帧间隔”（由 DCFG 中的 PFIVL 位控制）内，当模块发现任何在当前帧内应为空的同步 IN 端点 FIFO 中的数据还未发送完成时，都会在 GINTSTS 中生成一个 IISOIXFR 中断。

应用程序编程顺序：

1) 使用端点特性对 DIEPCTLx 寄存器进行编程，并将 CNAK 和 EPENA 位置 1。

2) 将需要在下一帧中发送的数据写入发送 FIFO。

3) 硬件触发 ITTXFE 中断（在 DIEPINTx 中）表示应用程序尚未将需要发送的全部数据写入发送 FIFO。

4) 如果在检测到中断前已使能中断端点，则将忽略该中断。如果中断端点未使能，则使能此端点，以便数据能够在收到下一次 IN 令牌时发送出去。

5) 硬件触发 XFRC 中断（在 DIEPINTx 中）时如果 DIEPINTx 中未产生 ITTXFE 中断，则表示成功完成同步 IN 传输。读取 DIEPTSIZx 寄存器时始终得到传输大小 = 0 且数据包计数 = 0，表示所有数据都已通过 USB 发送完毕。

6) 无论是否产生 ITTXFE 中断（在 DIEPINTx 中），只要触发 XFRC 中断（在 DIEPINTx 中）， 即表示中断 IN 传输成功完成。读取 DIEPTSIZx 寄存器时始终得到传输大小 = 0 且数据包计数 = 0，表示所有数据都已通过 USB 发送完毕。

7) 如果在未产生任何前述中断的情况下在 GINTSTS 中触发未完成同步 IN 传输（IISOIXFR）中断， 则表示模块在当前帧中至少未收到 1 个周期性 IN 令牌。

● 未完成同步 IN 数据传输

本节介绍应用程序针对未完成同步 IN 数据传输必须执行的操作。

内部数据流：

1) 符合下列条件之一时，即认为同步 IN 传输未完成：

a) 模块在至少一个同步 IN 端点上接收到损坏的同步 IN 令牌。此时，应用程序检测到未完成同步 IN 传输中断（ GINTSTS 中的 IISOIXFR 位)。

b) 应用程序向发送 FIFO 写入数据的速度过慢，在将完整数据写入 FIFO 之前便接收到 IN 令牌。此时，应用程序在 DIEPINTx 中检测到“Tx FIFO 为空时接收到 IN 令牌”中断。应用程序可忽略此中断，因为最终将在周期性帧结束时产生一个未完成同步 IN 传输中断（GINTSTS 中的 IISOIXFR 位)。模块会通过 USB 发送一个零长度数据包来响应接收到的 IN 令牌。

2) 应用程序必须尽快停止向发送 FIFO 写入数据。

3) 应用程序必须将端点的 NAK 位和禁止位置 1。

4) 模块会禁止该端点，将禁止位清零并触发端点的“端点禁止”中断。

应用程序编程顺序：

1) 应用程序可以在任何同步 IN 端点上忽略 DIEPINTx 中的“Tx FIFO 为空时接收到 IN 令牌”中断，因为最终这将产生一个未完成同步 IN 传输中断（在 OTG_GINTSTS 中)。

2) 硬件触发未完成同步 IN 传输中断（在 GINTSTS 中）表示在至少一个同步 IN 端点上存在未完成的同步 IN 传输。

3) 应用程序必须读取所有同步 IN 端点的“端点控制”寄存器来检测存在未完成 IN 数据传输的端点。

4) 应用程序必须停止向与这些端点相关联的“周期性发送 FIFO”写入数据。

5) 对 DIEPCTLx 寄存器中的下列字段进行编程以禁止端点：

- 在 DIEPCTLx 中， SNAK = 1
- 在 DIEPCTLx 中， EPDIS = 1

6) 硬件触发 DIEPINTx 中的“端点禁止”中断表示模块已禁止该端点。

- 此时，应用程序必须清空相关联的发送 FIFO 中的数据，或者通过在下一微帧中使能新传输的端点来覆盖 FIFO 中的现有数据。要刷新数据，应用程序必须使用 GRSTCTL 寄存器。

● 停止非同步 IN 端点

本节介绍应用程序如何才能停止非同步端点。

应用程序编程顺序：

1) 禁止要停止的 IN 端点。同时将 STALL 位置 1。

2) DIEPCTLx 中的 EPDIS = 1（当端点已使能时）

- DIEPCTLx 中的 STALL = 1
- STALL 位的优先级始终高于 NAK 位

3) 硬件触发“端点禁止”中断（在 DIEPINTx 中)，应用程序获知模块已禁止指定的端点。

4) 应用程序必须根据端点类型清空非周期性或周期性发送 FIFO。对于非周期性端点，应用程序必须重新使能另一个无需停止的非周期性端点来发送数据。

5) 当应用程序准备好结束该端点的 STALL 握手信号时，必须将 DIEPCTLx 的 STALL 位清零。

6) 如果应用程序因收到来自主机的 SetFeature.Endpoint Halt 命令或 ClearFeature.EndpointHalt 命令来设置或清除端点的 STALL 状态，则必须在该控制端点的状态阶段传输之前将 STALL 位置 1 或清零。

特例：停止控制 OUT 端点

如果在控制传输的数据阶段，主机发送的 IN/OUT 令牌数超过 SETUP 数据包指定的值，则模块必须对这些多余的 IN/OUT 令牌回复 STALL。在这种情况下，在控制传输的数据阶段，当模块传输完 SETUP 数据包指定的数据量后，应用程序必须使能 DIEPINTx 中的 ITTXFE 中断和 DOEPINTx 中的 OTEPDIS 中断。随后，当应用程序收到此中断时，必须将相应端点控制寄存器中的 STALL 位置 1 并清除此中断。

37.14.7. 最坏情况下的响应时间

当 OTG_FS/OTG_HS 控制器作为设备使用时，对于任何跟随在同步 OUT 之后的令牌，都存在一个最坏响应时间。这个最坏情况响应时间随 AHB 时钟频率的不同而异。

模块寄存器位于 AHB 域内，在更新这些寄存器值之前，模块不会接受新令牌。对于任何跟随在同步 OUT 之后的令牌，最坏的情况是：由于同步事务不需要握手信号，所以下一个令牌可能很快就到达。当 AHB 与 PHY 时钟频率相同时，这个最坏情况值为 7 个 PHY 时钟。AHB 时钟越快，此值越小。

如果发生这种最坏情况，模块将以 NAK 响应批量/中断令牌，并丢弃同步和 SETUP 令牌。对于 SETUP，主机会将这种情况视为超时，并尝试重新发送 SETUP 数据包。对于同步传输，会产生未完成同步 IN 传输中断 (IISOIXFR) 和未完成同步 OUT 传输中断 (IISOOXFR)，来通知应用程序同步 IN/OUT 数据包被丢弃。

■ 选择 GUSBCFG 中的 TRDT 的值

TRDT 的值 (GUSBCFG) 是指在接收到 IN 令牌后以 PHY 时钟计算的时间长度， MAC 将在这段时间内获取 FIFO 状态并从 PFC 模块获取第一个数据。这段时间包含 PHY 和 AHB 时钟之间的同步延迟。最坏情况延迟是当 AHB 时钟与 PHY 时钟相等时的延迟。这种情况下的延迟为 5 个时钟周期。

MAC 接收到 IN 令牌后，此信息（接收到的令牌）将由 PFC （ PFC 以 AHB 时钟运行）同步到 AHB 时钟。随后， PFC 从 SPRAM 中读取数据并将它们写入双时钟源缓冲区。 MAC 再从源缓冲区读出数据（ 4 个深度)。

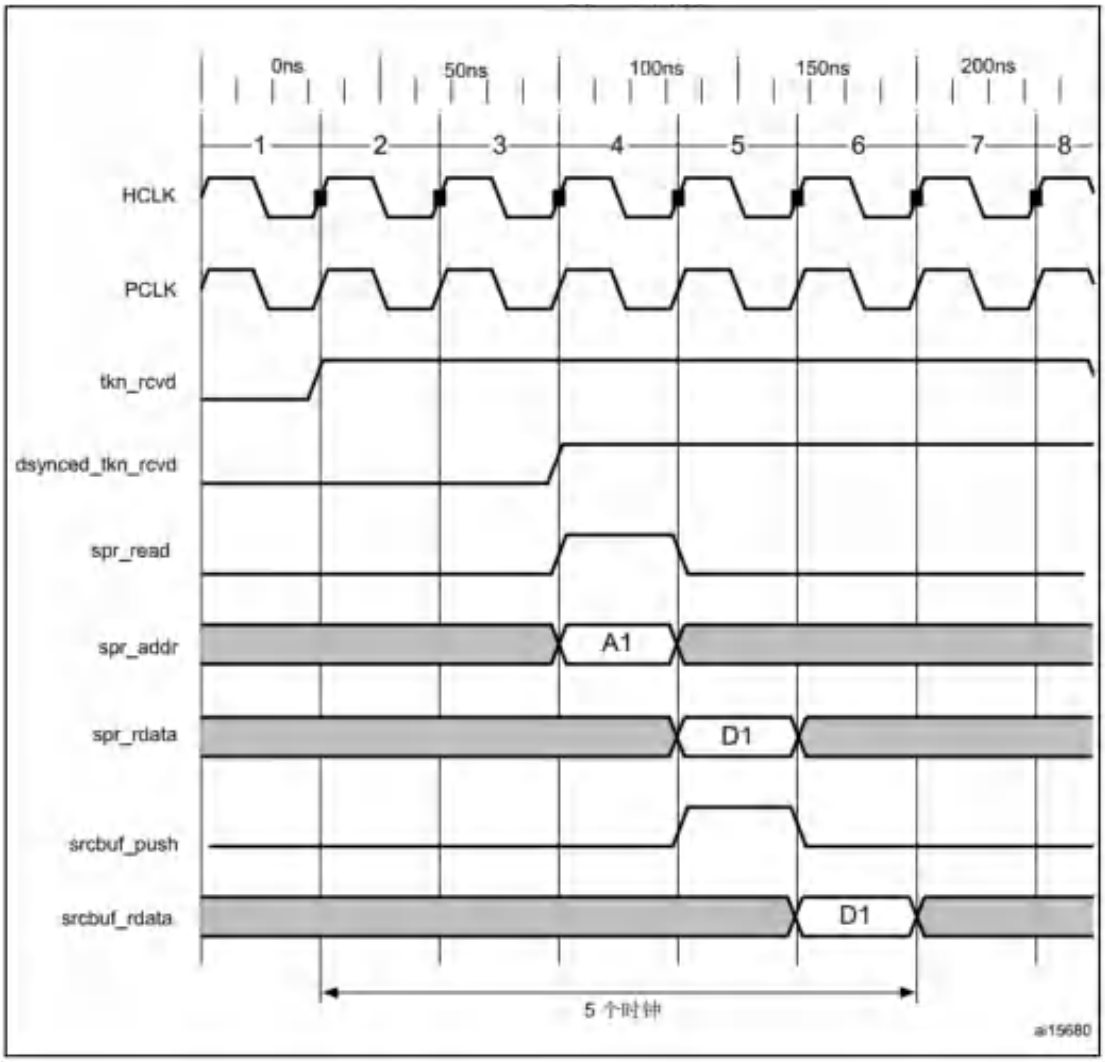
如果 AHB 的运行频率高于 PHY，应用程序可以使用较小的 TRDT 值（在 GUSBCFG 中)。

图 764 具有以下信号：

- tkn_rcvd：接收到令牌信息（从 MAC 到 PFC)
- dynced_tkn_rcvd：双倍同步 tkn_rcvd（从 PCLK 到 HCLK 域)
- spr_read：读取到 SPRAM
- spr_addr：寻址到 SPRAM
- spr_rdata：从 SPRAM 中读取数据
- srcbuf_push：压入源缓冲区
- srcbuf_rdata：从源缓冲区读取数据。 MAC 看到的数据

要计算 TRDT 的值，请参见表 487： TRDT 值 (HS)。

图 37-16 TRDT 最大时序情况



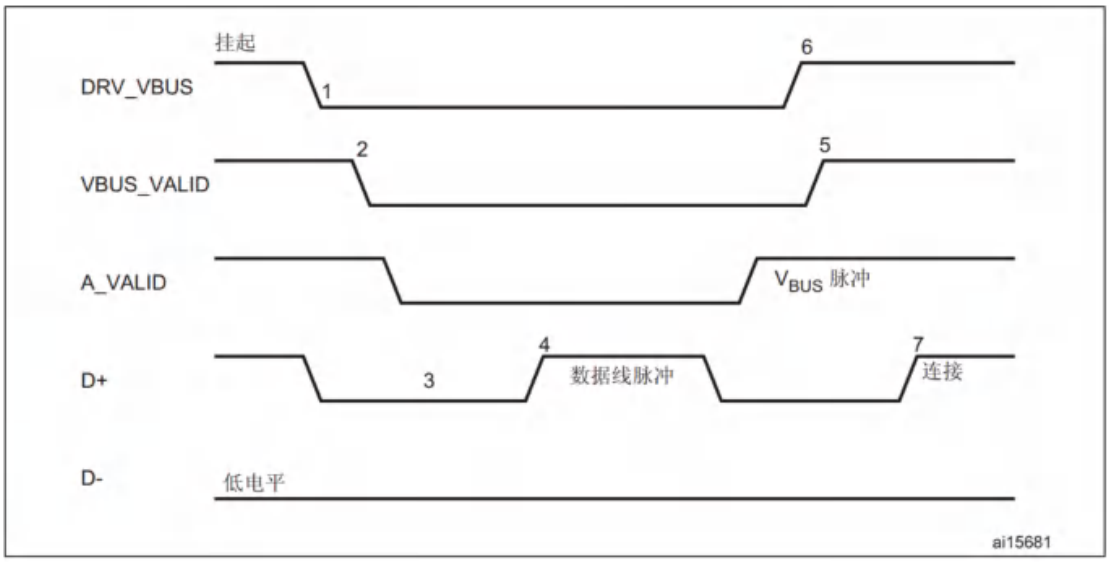
37.14.8. OTG 编程模型

OTG_HS 控制器是一种支持 HNP 和 SRP 的 OTG 设备。该模块接口与“A 型”插头连接时，该模块称为 A 器件。该模块接口与“B 型”插头连接时，该模块称为 B 器件。在主机模式下， OTG_HS 控制器将关闭 VBUS 以节省电能。 B 器件可以借助 SRP 请求 A 器件打开 VBUS 电源。设备必须同时执行数据线脉冲和 VBUS 脉冲，但主机可以检测数据线脉冲或者 VBUS 脉冲中的一个用于 SRP。 B 器件可以借助 HNP 协商并切换到主机角色。在协商模式下执行 HNP 后， B 器件会挂起总线并恢复到设备角色。

A 器件会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 使能位置 1。这将使能 OTG_HS 控制器在 A 器件模式下检测到 SRP。

图 37-17 A 器件 SRP



1. DRV_VBUS = 发送到 PHY 的 VBUS 驱动信号

VBUS_VALID = 来自 PHY 的 VBUS 有效信号

A_VALID = 发送到 PHY 的 A 器件 VBUS 电平信号

D+ = 正向数据线

D- = 负向数据线

以下几点引用并描述了图 765 所示的信号过程：

1) 为节省电能，在总线空闲时，应用程序会通过在主机端口控制和状态寄存器中写入端口挂起位和端口电源位来挂起设备并关闭端口电源。

2) PHY 通过禁止 VBUS_VALID 信号来指示端口电源已关闭。

3) 当 VBUS 电源关闭时，设备必须检测到至少 2 ms 的 SE0 信号才可以启动 SRP。

4) 要启动 SRP，设备需要打开其数据线的上拉电阻并维持 5 到 10 ms。 OTG_HS 控制器会检测到数据线脉冲。

5) 设备会驱动 VBUS 到 A 器件会话有效电平（最小 2)0 V）以上，以产生 VBUS 脉冲。OTG_HS 控制器检测到 SRP 时将中断应用程序。在全局中断状态寄存器中，检测到会话请求位（GINTSTS 中的 SRQINT 位）将置 1。

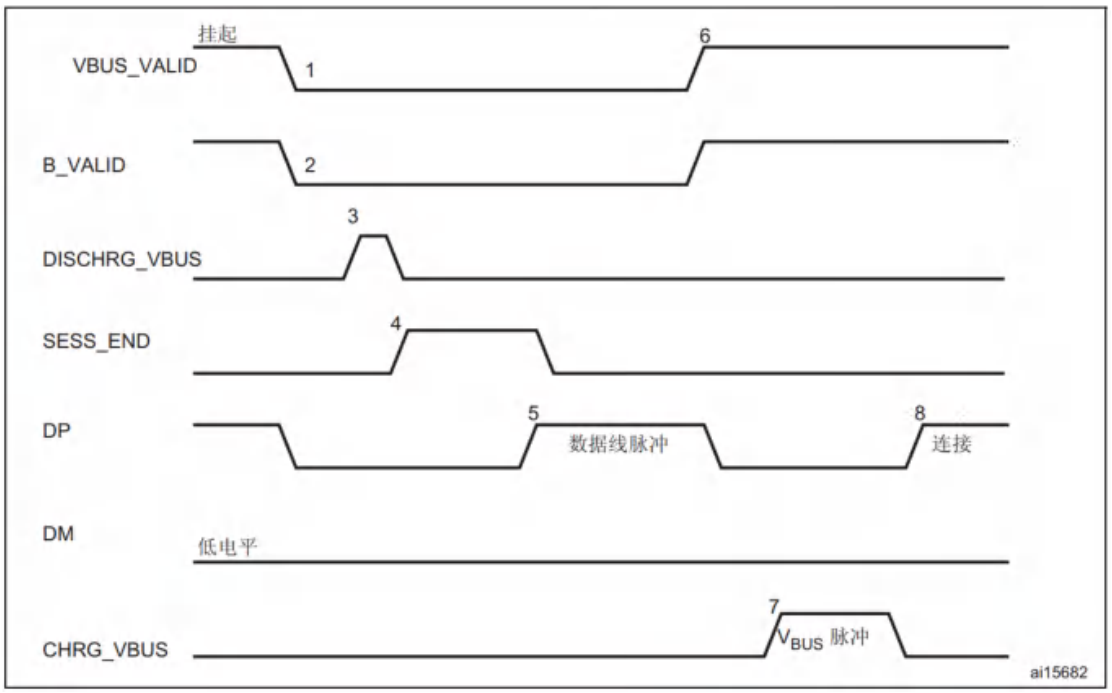
6) 应用程序必须响应“检测到会话请求”中断，并通过在主机端口控制和状态寄存器中写入端口电源位来打开端口电源位。 PHY 通过输出 VBUS_VALID 信号来指示端口已通电。

7) 当 USB 通电后，设备将连接，从而完成 SRP 过程。

B 器件会话请求协议

应用程序必须将模块 USB 配置寄存器中的 SRP 使能位置 1。这将使能 OTG_HS 控制器在 B 器件模式下启动 SRP。 OTG_HS 控制器可以借助 SRP 向主机请求新会话。

图 37-18 B 器件 SRP



1. VBUS_VALID = 来自 PHY 的 VBUS 有效信号

B_VALID = 发送到 PHY 的 B 器件有效会话

DISCHRG_VBUS = 发送到 PHY 的放电信号

SESS_END = 发送到 PHY 的会话结束信号

CHRG_VBUS = 发送到 PHY 的 VBUS 充电信号

DP = 正向数据线

DM = 负向数据线

以下几点引用并描述了图 766 所示的信号过程：

1) 为节省电能，在总线空闲时主机会挂起并关闭端口电源。

OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后， OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。OTG_HS 控制器会通知 PHY 对 VBUS 进行放电。

2) PHY 指示会话结束。这是 SRP 的初始条件。启动 SRP 之前， OTG_HS 控制器需要 2 ms 的 SE0 信号。

对于 USB 1.1 全速串行收发器，应用程序必须在 BSVLD 位（位于 OTG_GOTGCTL）被禁止后，等待 VBUS 放电至 0.2 V。此放电时间可从收发器供应商处获取，该值可能因收发器而异。

3) OTG_HS 模块通知 PHY 对 VBUS 加速放电。

4) 应用程序通过将 OTG 控制和状态寄存器中的会话请求位置 1 来启动 SRP。 OTG_HS 控制器先执行数据线脉冲，随后执行 VBUS 脉冲。

5) 主机会从数据线脉冲或 VBUS 脉冲检测到 SRP，然后打开 VBUS。 PHY 指示 VBUS 对设备上电。

6) OTG_HS 控制器执行 VBUS 脉冲。

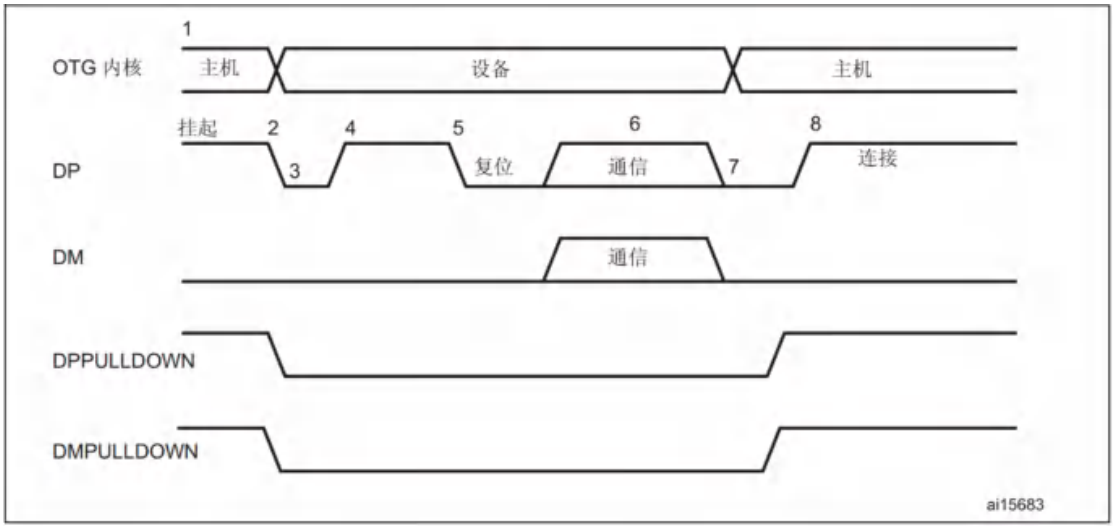
主机通过打开 VBUS 启动一个新会话，指示 SRP 已成功。 OTG_HS 控制器通过将 OTG 中断状态寄存器中的会话请求成功状态更改位置 1 来中断应用程序。应用程序读取 OTG 控制和状态寄存器中的会话请求成功位。

7) 当 USB 通电时， OTG_HS 控制器将进行连接，并完成 SRP 过程。

A 器件主机协商协议

通过 HNP 可以将 USB 主机角色从 A 器件切换到 B 器件。应用程序必须将模块 USB 配置寄存器中的 HNP 使能位置 1，以使能 OTG_HS 控制器在 A 器件模式下执行 HNP 功能。

图 37-19 A 器件 HNP



1. DPPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DP 线的下拉电阻。

DMPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DM 线的下拉电阻。

以下几点引用并描述了图 767 所示的信号过程：

1) OTG_HS 控制器向 B 器件发送 SetFeature b_hnp_enable 描述符以使能 HNP 支持。 B 器件的 ACK 响应表示 B 器件支持 HNP。应用程序必须将 OTG 控制和状态寄存器中的主机设置 HNP 使能位置 1，以向 OTG_FS/ OTG_HS 控制器指示 B 器件支持 HNP。

2) 应用程序不再使用总线时，会通过在主机端口控制和状态寄存器中写入端口挂起位来挂起总线。

3) B 器件检测到 USB 挂起时，将断开连接，指示 HNP 的初始条件。 B 器件只有在切换到主机角色时才会启动 HNP，否则总线会保持挂起状态。

OTG_HS 控制器将 OTG 中断状态寄存器中检测到的主机协商中断位置 1，指示 HNP 的启动。

OTG_HS 控制器将禁止 PHY 中 DP 线和 DM 线的下拉，以指示设备角色。PHY 使能 OTG_DP 上拉电阻，以告知 B 器件 A 器件的连接。应用程序必须读取 OTG 控制和状态寄存器中的当前模式位，以确定设备工作模式。

4) B 器件检测到连接，发出 USB 复位信号，并枚举 OTG_HS 进行数据通信。

5) B 器件继续担当主机角色，发起数据通信，并在结束时挂起总线。

OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后， OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。

6) 在协商模式下， OTG_HS 控制器会检测到总线挂起，连接断开，然后切换回主机角色。OTG_HS 控制器将使能 PHY 中 DM 和 DM 下拉电阻，以指示其承担主机角色。

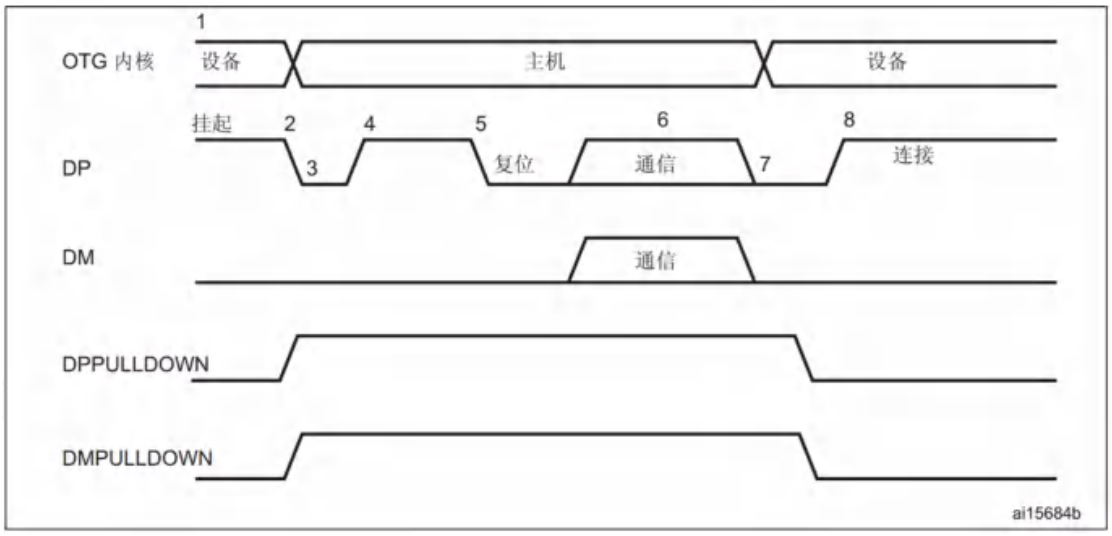
7) OTG_HS 控制器将 OTG 中断状态寄存器中的连接器 ID 状态更改中断置 1。应用程序必须读取 OTG 控制和状态寄存器中的连接器 ID 状态位，以确定 OTG_HS 控制器在 A 器件模式下工作。这向应用程序表示 HNP 已经完成。应用程序必须读取 OTG 控制和状态寄存器中的当前模式位，以确定主机工作模式。

8) 连接 B 器件， 完成 HNP 过程。

B 器件主机协商协议

通过 HNP 可以将 USB 主机角色从 B 器件切换到 A 器件。应用程序必须将模块 USB 配置寄存器中的 HNP 功能位置 1，以使能 OTG_HS 控制器作为 B 器件的 HNP 功能。

图 37-20 B 器件 HNP



1. DPPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DP 线的下拉电阻。

DMPULLDOWN = 从模块发送到 PHY 的信号，用于使能/禁止 PHY 内部 DM 线的下拉电阻。

以下几点引用并描述了图 768 所示的信号过程：

1) A 器件发出一个 SetFeature b_hnp_enable 描述符以使能 HNP 支持。 OTG_HS 控制器的 ACK 响应表示它支持 HNP。应用程序必须将 OTG 控制和状态寄存器中的设备 HNP 使能位置 1，以指示支持 HNP。

应用程序将 OTG 控制和状态寄存器中的 HNP 请求位置 1，以向 OTG_HS 控制器指示启动 HNP。

2) A 器件不再使用总线时，会通过在主机端口控制和状态寄存器中写入端口挂起位来挂起总线。

OTG_HS 控制器会在总线空闲 3 ms 后，将模块中断寄存器中的早期挂起位置 1。随后， OTG_HS 控制器会将模块中断寄存器中的 USB 挂起位置 1。OTG_HS 控制器会断开连接， A 器件在总线上检测到 SE0，指示 HNP 即将开始。OTG_HS 控制器将使能 PHY 中的 DP 和 DM 下拉电阻，以指示其承担主机角色。

在检测到 SE0 的 3 ms 内， A 器件会通过激活 OTG_DP 上拉电阻进行响应。 OTG_HS 控制器检测到此信号时认为有设备接入。OTG_HS 控制器将 OTG 中断状态寄存器中的主机协商成功状态更改中断位置 1，指示 HNP 的状态。应用程序必须读取 OTG 控制和状态寄存器中的主机协商成功位，以确定主机协商成功。应用程序必须读取模块中断寄存器 (OTG_GINTSTS) 中的当前模式位，以确定主机工作模式。

3) 应用程序将复位位（ OTG_HS_HPRT 中的 PRST 位）置 1，同时 OTG_HS 控制器发出 USB 复位信号，并枚举 A 器件进行数据通信。

4) OTG_HS 控制器继续保持发起通信的主机角色，在通信结束后，会通过将主机端口控制和状态寄存器中的端口挂起位置 1 来挂起总线。

5) 在协商模式下，当 A 器件检测到挂起时，会断开连接，然后切换回主机角色。 OTG_HS 控制器将禁止 PHY 中 DP 线和 DM 线的下拉，以指示其假设的设备角色。

6) 应用程序必须读取模块中断寄存器 (OTG_GINTSTS) 中的当前模式位，以确定主机工作模式。

7) OTG_HS 控制器进行连接，完成 HNP 过程

37.15. USB_OTH_HS 寄存器描述

USB1_OTG_HS 寄存器基地址：0x40040000

USB2_OTG_HS 寄存器基地址：0x40080000

37.15.1. 全局寄存器列表

偏移	名称	复位值	描述
0x0	OTG_GOTGCTL		控制和状态寄存器
0x4	OTG_GOTGINT		中断寄存器
0x8	OTG_GAHBCFG		AHB 配置寄存器
0xC	OTG_GUSBCFG		USB 配置寄存器
0x10	OTG_GRSTCTL		复位寄存器
0x14	OTG_GINTSTS		中断寄存器
0x18	OTG_GINTMSK		中断屏蔽寄存器
0x1C	OTG_GRXSTSR		接收状态调试读取寄存器
0x20	OTG_GRXSTSP		接收状态读取和出栈寄存器
0x24	OTG_GRXFSIZ		接收 FIFO 大小寄存器
0x28	OTG_GNPTXFSIZ		非周期性发送 FIFO 大小寄存器
0x2C	OTG_GNPTXSTS		非周期性发送 FIFO/队列状态寄存器
0x38	OTG_GGPIO		通用 IO 寄存器
0x3C	OTG_GUID		用户 ID 寄存器
0x4C	OTG_GHWCFG3		用户硬件配置寄存器 3
0x54	OTG_GLPMCFG		LPM 配置寄存器
0x5C	OTG_GDFIFOCFG		全局 DFIFO 配置寄存器
0x100	OTG_HPTXFSIZ		主机周期性发送 FIFO 大小寄存器
0x104+(x-1)*0x4	OTG_DIEPTXFx		设备 IN 端点发送 FIFO 大小寄存器 x（x=1...15，其中 x=FIFO 编号）

37.15.1.1. 控制和状态寄存器(OTG_GOTGCTL: 00h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21	CURMOD	RO	0x0	当前操作模式（Current Mode of Operation） 指示当前模式。 1'b0: 设备模式 1'b1: 主机模式 注： 在设备模式和主机模式均可访问。
20	OTGVER	RW	0x0	OTG 版本（OTGVer） 选择 OTG 版本。 0： OTG 版本 1.3。在此版本中，模块支持通过数据线脉冲和 Vbus 脉冲实现 SRP。 1： OTG 版本 2.0。在此版本中，模块仅支持通过数据线脉冲实现 SRP。

19	BSESVLD	RO	0x0	B 会话有效 (B-session valid) 指示设备模式下收发器的状态。 0： B 会话无效。 1： B 会话有效。 在 OTG 模式下，该位可用于确定设备处于连接状态还是断开状态。 注： 仅可在设备模式下访问。
18	ASESVLD	RO	0x0	A 会话有效 (A-session valid) 指示主机模式下收发器的状态。 0： A 会话无效 1： A 会话有效 注： 仅可在主机模式下访问
17	DBNCTIME	RO	0x0	长/短去抖动时间 (Long/short debounce time) 指示已检测到连接的去抖动时间。 0：长去抖动时间，用于物理连接 (100ms + 2.5 µs) 1：短去抖动时间，用于软连接 (2.5 µs) 注： 仅可在主机模式下访问。
16	CONIDSTS	RO	0x1	连接器 ID 状态 (Connector ID status) 指示发生连接事件时的连接器 ID 状态。 0： OTG_FS/OTG_HS 控制器处于 A 器件模式 1： OTG_FS/OTG_HS 控制器处于 B 器件模式 注： 在设备模式和主机模式均可访问。
15	DBNCEFLTRBYPASS	RW	0x0	模式：主机和设备 去抖滤波器旁路（Debounce Filter Bypass） 启用时绕过 avalid、bvalid、vbusvalid、sessend、iddig 信号的去抖过滤器。 注意：该寄存器位仅在内核中存在去抖滤波器时才有效。 0x0: 禁止去抖过滤器旁路 0x1: 使能去抖过滤器旁路
14:13	RSV	-	-	保留
12	EHEN	RW	0x0	嵌入式主机启用 Embedded Host Enable (EHEn) 用于在 OTG A 器件状态机和嵌入式主机状态机之间选择。 0：选择 OTG A 器件状态机 1：选择嵌入式主机状态机

11	DEVHNPEN	RW	0x0	模式：支持 HNP 的 OTG 设备 已启用 HNP 的设备 (DevHNPEn) 使能设备 HNP 特性 (Device HNP enabled) 从所连 USB 主机处成功接收到 SetFeature.SetHNPEnable 命令时，应用程序将该位置 1。 0：应用不使能 HNP 1：应用使能 HNP 注： 仅可在设备模式下访问
10	HSTSETHNPEN	RW	0x0	模式：支持 HNP 的 OTG 主机 主机设置 HNP 启用 (HstSetHNPEn) 使用 SetFeature.SetHNPEnable 命令在所连接设备上成功使能 HNP 后，应用程序将该 位置 1。 0：主机未对设备设置了 HNP 使能 1：主机已对设备设置了 HNP 使能 注： 仅可在主机模式下访问。
9	HNPREQ	RW	0x0	模式：支持 HNP 的 OTG 设备 HNP 请求 (HNP request) 应用程序将该位置 1 时，将对所连接 USB 主机发起 HNP 请求。当 OTG_GOTGINT 寄存器中的主机协商成功状态更改位（OTG_GOTGINT 中的 HNSSCHG 位）置 1 时，应用程序可 通过写 0 将该位清零。 HNSSCHG 位清零时，模块会将该位清零。 0：无 HNP 请求 1： HNP 请求 注： 仅可在设备模式下访问。
8	HSTNEGSCS	RO	0x0	模式：支持 HNP 的设备 主机协商成功 (HstNegScs) 当主机协商成功时，模块会将该位置 1。当该寄存器中的 HNP 请求位 (HNPRQ) 置 1 时，模块会将该位清零。 0：主机协商失败 1：主机协商成功 注： 仅可在设备模式下访问
7	BVALIDOVVAL	RW	0x0	B 器件会话有效覆盖值 (B-peripheral session valid override value) 此位用于在 BVALOEN 位置 1 时设置 Bvalid 信号的覆盖值。 0： BVALOEN = 1 时， Bvalid 值为 “0” 1： BVALOEN = 1 时， Bvalid 值为 “1” 注： 仅可在设备模式下访问。

6	BVALIDOVEN	RW	0x0	B 器件会话有效覆盖使能 (B-peripheral session valid override enable) 该位用于使能/禁止软件通过 BVALOVAL 位来覆盖 Bvalid 信号。 0：禁止覆盖，模块使用来自所选 PHY 的 Bvalid 信号 1：从 PHY 内部接收的 Bvalid 由 BVALOVAL 位的值覆盖 注： 仅可在设备模式下访问。
5	AVALIDOVVAL	RW	0x0	A 器件会话有效覆盖值 (A-peripheral session valid override value) 此位用于在 AVALOEN 位置 1 时设置 Avalid 信号的覆盖值。 0： AVALOEN = 1 时， Avalid 值为 “0” 1： AVALOEN = 1 时， Avalid 值为 “1” 注： 仅可在主机模式下访问。
4	AVALIDOVEN	RW	0x0	A 器件会话有效覆盖使能 (A-peripheral session valid override enable) 该位用于使能/禁止软件通过 AVALOVAL 位来覆盖 Avalid 信号。 0：禁止覆盖，模块使用来自所选 PHY 的 Avalid 信号 1：从 PHY 内部接收的 Avalid 由 AVALOVAL 位的值覆盖 注： 仅可在主机模式下访问。
3	VBVALIDOVVAL	RW	0x0	VBUS 有效覆盖值 (VBUS valid override value) 此位用于在 VBVALOEN 位置 1 时设置 vbusvalid 信号的覆盖值。 0： VBVALOEN = 1 时， vbusvalid 值为 “0” 1： VBVALOEN = 1 时， vbusvalid 值为 “1” 注： 仅可在主机模式下访问。
2	VBVALIDOVEN	RW	0x0	VBUS 有效覆盖使能 (VBUS valid override enable) 该位用于使能/禁止软件通过 VBVALOVAL 位来覆盖 vbusvalid 信号。 0：禁止覆盖，模块使用来自所选 PHY 的 vbusvalid 信号 1：从 PHY 内部接收的 vbusvalid 由 VBVALOVAL 位的值覆盖 注： 仅可在主机模式下访问。
1	SESREQ	RW	0x0	会话请求 (Session request) 应用程序将该位置 1 时，将在 USB 上发起会话请求。当 OTG_GOTGINT 寄存器中的主机协 商成功状态更改位（OTG_GOTGINT 中的 HNSSCHG 位）置 1 时，应用程序可通过写 0 将该位清零。 HNSSCHG 位清零时，模块会将该位清零。 如要使用 USB 1.1 全速串行收发器接口来启动会话请求，则应用程序必须在该寄存器中的 B 会话有效位（OTG_GOTGCTL 中的 BSVLD 位）清零后，等待 VBUS 放电至 0.2 V。该放电时间因 PHY 不同而异，可从 PHY 供应商处了解相关信息。 0：无会话请求 1：会话请求 注： 仅可在设备模式下访问。

0	SESREQSCS	RO	0x0	会话请求成功（Session request success） 当会话请求成功时，模块会将该位置 1。 0：会话请求失败 1：会话请求成功 注： 仅可在设备模式下访问。

37.15.1.2. 中断寄存器(OTG_GOTGINT: 04h)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19	DBNCEDONE	RCW1	0x0	去抖动完成（Debounce done） 模块会在设备连接后且去抖动结束时将该位置 1。应用程序会在看见该中断后，开始驱动 USB 复位。该位仅在 OTG_GUSBCFG 寄存器中的 HNP 功能位或 SRP 功能位（分别为 OTG_GUSBCFG 中的 HNPCAP 位或 SRPCAP 位）置 1 时有效。 注： 仅可在主机模式下访问。
18	ADEVTOUTCHG	RCW1	0x0	A 器件超时更改（A-device timeout change） 模块将该位置 1 时，指示 A 器件在等待 B 器件连接时超时。 注： 在设备模式和主机模式均可访问。
17	HSTNEGDET	RCW1	0x0	检测到主机协商（Host negotiation detected） 当检测到 USB 上的主机协商请求时，模块会将该位置 1。 注： 在设备模式和主机模式均可访问。
16:10	RSV	-	-	保留
9	HSTNEGSUCSTSCHNG	RCW1	0x0	主机协商成功状态更改（Host negotiation success status change） 模块将在 USB 主机协商请求成功或失败时将此位置 1。应用程序必须读取 OTG_GOTGCTL 寄存器中的主机协商成功位（ OTG_GOTGCTL 中的 HNGSCS 位）来检查请求成功还是 失败。 注： 在设备模式和主机模式均可访问。
8	SESREQSUCSTSCHNG	RCW1	0x0	会话请求成功状态更改（Session request success status change） 模块将在会话请求成功或失败时将此位置 1。应用程序必须读取 OTG_GOTGCTL 寄存器中的 会话请求成功位（OTG_GOTGCTL 中的 SRQSCS 位）来检查请求成功还是失败。 注： 在设备模式和主机模式均可访问
7:3	RSV	-	-	保留

2	SESENDDET	RCW1	0x0	检测到会话结束 (Session end detected) 模块将该位置 1 时，指示 VBUS < 0.8 V 时， VBUS 上的电压不再适用于 B 器件会话。 注： 在设备模式和主机模式均可访问。
1:0	RSV	-	-	保留

37.15.1.3. AHB 配置寄存器(OTG_GAHBCFG: 08h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23	AHBSINGLE	RW	0x0	AHB 单点支持 (AHBSingle) 当内核以 DMA 模式运行时，该位被编程为支持传输中剩余数据的单次传输。 1'b0: 传输中的剩余数据使用 INCR 突发大小发送。 1'b1: 传输中的剩余数据使用单突发大小发送。 注意：如果启用此功能，AHB RETRY 和 SPLIT 传输仍具有 INCR 突发类型。当连接到内核的 AHB 从站不支持 INCR 突发（以及总线中不使用拆分和重试事务）时，请启用此功能。 0x0 (INCRBURST): 传输中的剩余数据使用 INCR 突发大小发送 0x1 (SINGLEBURST): 传输中的剩余数据使用单突发大小发送 注： 在设备模式和主机模式均可访问。
22	NOTIALLDMAWRIT	RW	0x0	通知所有 DMA 写入事务 (NotiAllDmaWrit) 对该位进行编程，以启用与通道/端点相对应的所有 DMA 写入事务的系统 DMA 完成功能。只有当 GAHBCFG.RemMemSupp 设置为 1 时，该位才有效。 0x1 (ALLTRANS): 对于 AHB 接口上的所有 DMA 写入事务，内核都会发出 int_dma_req，同时还会发出 int_dma_done、chep_last_transact 和 chep_number 信号信息。内核等待所有 DMA 写入事务的 sys_dma_done 信号，以完成特定通道/端点的传输。 0x0 (LASTTRANS) ： 内核仅对与特定通道/端点相对应的 DMA 写入传输的最后一个事务发出 int_dma_req 信号。同样，内核只有在 DMA 写入事务完成特定通道/端点的传输时，才会等待 sys_dma_done 信号。 注： 在设备模式和主机模式均可访问。

21	REMMEMSUPP	RW	0x0	远程内存支持（RemMemSupp） 对该位进行编程，以启用等待系统 DMA 完成信号的 DMA 写入传输功能。 GAHBCFG.RemMemSupp=1 当 DMA 开始对外部存储器进行写入传输时，会发出 int_dma_req 输出信号。当内核完成传输时，它会发出 int_dma_done 信号，标志着控制器完成了 DMA 写入。然后，内核等待系统发出 sys_dma_done 信号，继续完成与特定通道/端点相对应的数据传输。 GAHBCFG.RemMemSupp=0 int_dma_req 和 int_dma_done 信号不会被断言，只要 DMA 写入传输在内核边界完成，内核就会继续断言 XferComp 中断，而不会等待 sys_dma_done 信号来完成数据传输。 0x0 (DISABLED): 远程内存支持功能已禁用 0x1 (ENABLED): 已启用远程内存支持功能 注： 在设备模式和主机模式均可访问。
20:9	RSV	-	-	保留
8	PTXFEMPLVL	RW	0x0	周期性 Tx FIFO 空门限 (Periodic Tx FIFO empty level) 指示 GINTSTS 寄存器中的周期性 Tx FIFO 空中断位。GINTSTS 中的 PTXFE 位）何时触发。 0：GINTSTS.PTxFEmp 中断表示周期性 Tx FIFO 为半空状态 1：GINTSTS.PTxFEmp 中断表示周期性 Tx FIFO 为全空状态 注： 仅可在主机模式下访问。

7	NPTXFEMPLVL	RW	0x0	非周期性 TxFIFO 空电平 (NPTxFEmpLvl) 该位仅在从机模式下使用。在主机模式和设备共享 FIFO 模式下，该位指示何时触发核心中断寄存器 (GINTSTS.NPTxFEmp) 中的非周期性 TxFIFO 空中断位。 在设备模式下使用专用 FIFO 时，该位指示何时触发 IN 端点发送 FIFO 空中断（DIEPINTn.TxFEmp)。 主机模式和共享 FIFO 与设备模式： 1'b0: GINTSTS.NPTxFEmp 中断指示非定期 TxFIFO 已空一半。 1'b1: GINTSTS.NPTxFEmp 中断指示非定期 TxFIFO 已完全清空。 设备模式下的专用 FIFO： 1'b0:DIEPINTn.TxFEmp 中断指示 IN 端点 TxFIFO 已空一半。 1'b1: DIEPINTn.TxFEmp 中断指示 IN 端点 TxFIFO 已完全清空。 0x0 (HALFEMPTY): DIEPINTn.TxFEmp 中断表示非周期性 TxFIFO 已空一半或 IN 端点 TxFIFO 已空一半。 0x1 (EMPTY): GINTSTS.NPTxFEmp 中断表示非定期 TxFIFO 全部清空或 IN 端点 TxFIFO 全部清空。 注： 在设备模式和主机模式均可访问。
6	RSV	-	-	保留
5	DMAEN	RW	0x0	DMA Enable (DMAEn) 为 USB OTG HS 使能 DMA (DMA enabled for USB OTG HS) 0：模块不使用 DMA 进行数据收发 1：模块使用 DMA 进行数据收发 注： 在设备模式和主机模式均可访问。

4:1	HBSTLEN	OTG_ARCHITECTURE == 0 ? RO:RW	0x0	突发长度/类型（HBstLen） 该字段用于外部和内部 DMA 模式. 在外部 DMA 模式下，这些位出现在 dma_burst[3:0] 端口上，可由外部封装器用于将外部 DMA 控制器接口连接到 Synopsys DW_ahb_dmac 或 ARM PrimeCell。 External DMA Mode defines the 以 32 位字为单位的 DMA burst 长度： 4'b0000: 1 word 4'b0001: 4 words 4'b0010: 8 words 4'b0011: 16 words 4'b0100: 32 words 4'b0101: 64 words 4'b0110: 128 words 4'b0111: 256 words Others: Reserved 内部 DMA 模式 AHB 主控 burst 类型： 4'b0000 Single 4'b0001 INCR 4'b0011 INCR4 4'b0101 INCR8 4'b0111 INCR16 Others: Reserved 0x0 (WORD1ORSINGLE): 1 word or single 0x1 (WORD4ORINCR): 4 words or INCR 0x2 (WORD8): 8 words 0x3 (WORD16ORINCR4): 16 words or INCR4 0x4 (WORD32): 32 words 0x5 (WORD64ORINCR8): 64 words or INCR8 0x6 (WORD128): 128 words 0x7 (WORD256ORINCR16): 256 words or INCR16 0x8 (WORDX): Others reserved 注： 在设备模式和主机模式均可访问。
0	GLBLINTRMSK	RW	0x0	全局中断屏蔽（Global interrupt mask） 该位用于屏蔽全局中断或使能全局中断。中断状态寄存器由模块进行更新，与此位的设置无关。 0：对应用程序屏蔽中断。 1：不对应用程序屏蔽中断。 注： 在设备模式和主机模式均可访问。

37.15.1.4. USB 配置寄存器(OTG_GUSBCFG: 0Ch)

位域	名称	属性	复位值	描述
31	CORRUPTTXPKT	WO	0x0	损坏的发送的数据包（CorruptTxPkt） 该位仅供调试使用。应用程序应始终将 1'b0 写入该位。 0x0 (NODEBUG)：正常模式 0x1 (DEBUG)：调试模式 注： 在设备模式和主机模式均可访问。
30	FORCEDEVMODE	RW	0x0	强制设备模式（Force device mode） 向该位写入 1 时，可将模块强制为设备模式，而无需考虑 OTG_ID 输入引脚。 0：正常模式 1：强制设备模式 将强制位置 1 后，应用程序必须等待至少 25 ms 后更改方可生效。 注： 在设备模式和主机模式均可访问
29	FORCEHSTMODE	RW	0x0	强制主机模式（Force host mode） 向该位写入 1 时，可将模块强制为主机模式，而无需考虑 OTG_ID 输入引脚。 0：正常模式 1：强制主机模式 将强制位置 1 后，应用程序必须等待至少 25 ms 后更改方可生效。 注： 在设备模式和主机模式均可访问
28	TXENDDELAY	RW	0x0	发送端延迟（TxEndDelay） 向该位写入 1'b1 可使控制器在远程唤醒过程中遵循 UTMI+ 规范 1.05 第 4.1.5 节规定的 opmode 信号 TxEndDelay 时序。 1'b0：正常模式 1'b1：发送端延迟 注： 仅可在设备模式下访问。
27	RSV	-	-	保留

26	IC_USBCAP	{OTG_ENABLE_IC_USB == 1 & OTG_FSPHY_INTERFACE != 0 ? RW:ROO}	0x0	IC_USB 功能 (IC_USBCap) 应用程序使用该位控制内核的 IC_USB 功能。1'b0: 未选择 IC_USB PHY 接口。 1'b1: 已选择 IC_USB PHY 接口。 只有当 OTG_ENABLE_IC_USB=1 和 OTG_FSPHY_INTERFACE!=0 时，该位才可写入。 当 OTG_ENABLE_IC_USB = 1 时，复位值取决于配置参数 OTG_SELECT_IC_USB。在所有其他情况下，该位被设置为 1'b0，且只读。 注： 在设备模式和主机模式均可访问
25:23	RSV	-	-	保留
22	TERMSELDLPULSE	OTG_HSPHY_INTERFACE!= 0 ? RW:ROO	0x0	USB OTG HS 的 TermSel DLine 脉冲选择 (TermSel DLine pulsing selection for USB OTG HS) 该位选择 utmi_termselect 来驱动 SRP（会话请求协议）期间的数据线脉冲。 0：使用 utmi_txvalid 驱动数据线脉冲（默认） 1：使用 utmi_termsel 驱动数据线脉冲 注： 仅可在设备模式下访问。
21:16	RSV	-	-	保留
15	PHYLPWRCLKSEL	RW	0x0	USB OTG HS 的 PHY 低功耗时钟选择 (PHY Low-power clock select for USB OTG HS) 该位用于选择 480 MHz 或 48 MHz（低功耗） PHY 模式。在 FS 和 LS 模式下， PHY 通常可 使用 48 MHz 时钟运行，从而节约功耗。 0： 480 MHz 内部 PLL 时钟 1： 48 MHz 外部时钟 在 480 MHz 模式下， UTMI 接口以 60 MHz 或 30MHz 运行，具体取决于选择的是 8 位还是 16 位数据宽度。在 48 MHz 模式下， UTMI 接口在 FS 和 LS 模式下以 48 MHz 运行。 注： 在设备模式和主机模式均可访问
14	RSV	-	-	保留

13:10	USBTRDTIM	RW	0x5	USB 周转时间 (USB turnaround time) 以 PHY 时钟数为单位设置周转时间。 要计算 TRDT 的值，请使用如下公式： TRDT = 4 × AHB 时钟 + 1 个 PHY 时钟 例如： 如果 AHB 时钟频率 = 72 MHz（PHY 时钟频率 = 48 MHz），则 TRDT 设置为 9。 如果 AHB 时钟频率 = 48 MHz（PHY 时钟频率 = 48 MHz），则 TRDT 设置为 5。 注： 仅可在设备模式下访问。
9	HNPCAP	OTG_MODE == 0 ? RW:ROO	0x0	HNP 功能 (HNP-capable) 应用程序使用该位控制 OTG_FS/OTG_HS 控制器的 HNP 功能。 0：不使能 HNP 功能。 1：使能 HNP 功能。 注： 在设备模式和主机模式均可访问。
8	SRPCAP	OTG_MODE !=2 & OTG_MODE != 4 ? RW:ROO	0x0	SRP 功能 (SRP-capable) 应用程序使用该位控制 OTG_FS/OTG_HS 控制器的 SRP 功能。如果模块作为无 SRP 功 能的 B 器件，则无法请求连接的 A 器件（主机）激活 VBUS 并开始会话。 0：不使能 SRP 功能。 1：使能 SRP 功能。 注： 在设备模式和主机模式均可访问。
7	RSV	-	-	保留
6	PHYSEL	OTG_HSPHY_INTERFACE!= 0 & OTG_FSPHY_INTERFACE!= 0 ? RW:ROO	0x0	USB2.0 高速 PHY 或 USB 1.1 全速串行收发器选择 (PHYSel) 应用程序利用该位选择高速 UTMI+ 或 ULPI PHY，或全速收发器。 1'b0: USB 2.0 高速 UTMI+ 或 ULPI PHY 1'b1: USB 1.1 全速串行收发器 如果未选择 USB 1.1 全速串行收发器接口，则该位始终为 0，只允许写入。 如果未选择高速 PHY 接口，则该位始终为 1，只允许写访问。 如果选择了两种接口类型（参数值不为零），应用程序将使用该位选择哪个接口处于活动状态，并进行读写访问。 注： 在设备模式和主机模式均可访问。

5	FSINTF	OTG_FSPHY_INTERFACE!= 0 ? RW:ROO	0x0	全速串行接口选择（FSIntf） 应用程序利用该位选择单向或双向 USB 1.1 全速串行收发器接口。 1'b0: 6-pin 单向全速串行接口 1'b1: 3-pin 双向全速串行接口 如果未选择 USB 1.1 全速串行收发器接口，则该位始终为 0，只允许写访问。如果选择了 USB 1.1 FS 接口，则应用程序可通过设置该位在 3 引脚和 6 引脚接口之间进行选择，访问方式为读写。 注：若要支持新的 4 针双向接口，则需要选择 6 针单向 FS 串行模式，并添加外部控制将其转换为 4 针接口。 注： 在设备模式和主机模式均可访问。 Volatile: true
4	ULPI_UTMI_SEL	(OTG_HSPHY_INTERFACE) == 3) ? RW : ((OTG_HSPHY_INTERFACE == 2) & (OTG_FSPHY_INTERFACE == 3))? RW:RO	0x0	ULPI or UTMI+ 选择（ULPI_UTMI_Sel) 应用程序使用该位选择 UTMI+ 接口或 ULPI 接口。 1'b0: UTMI+ 接口 1'b1: ULPI 接口 注： 在设备模式和主机模式均可访问。 Volatile: true
3	PHYIF	OTG_HSPHY_DWIDTH==2 ? RW:RO	0x0	PHY 接口（PHYIf) 应用程序使用该位配置内核，以支持具有 8 位或 16 位接口的 UTMI+ PHY。选择 ULPI PHY 时，必须将其设置为 8 位模式。 1'b0: 8 bits 模式 1'b1: 16 bits 模式 只有选择了 UTMI+ 和 ULPI，该位才可写入。否则，该位将返回配置时所选开机接口的值。 注： 在设备模式和主机模式均可访问。

2:0	TOUTCAL	RW	0x0	HS/FS 超时校准（TOutCal） 应用程序在该字段中编入的 PHY 时钟数被添加到内核中的高速/全速包间超时持续时间中，以考虑 PHY 带来的任何额外延迟。这可能是必要的，因为 PHY 在生成线路状态条件时引入的延迟可能因 PHY 而异。 高速运行的 USB 标准超时值为 736 至 816（含）比特倍。全速运行的 USB 标准超时值为 16 至 18（含）位次。应用程序必须根据枚举速度对该字段进行编程。 每个 PHY 时钟增加的比特次数如下： 高速运行： One 30-MHz PHY clock = 16 bit times One 60-MHz PHY clock = 8 bit times 全速运行： One 30-MHz PHY clock = 0.4 bit times One 60-MHz PHY clock = 0.2 bit times One 48-MHz PHY clock = 0.25 bit times 注： 在设备模式和主机模式均可访问。

37.15.1.5. 复位寄存器(OTG_GRSTCTL: 10h)

位域	名称	属性	复位值	描述
31	AHBIDLE	RO	0x1	AHB 主器件空闲（AHB master idle） 指示 AHB 主器件状态机处于空闲情况。 注： 在设备模式和主机模式均可访问。
30	DMAREQ	RO	0x0	DMA 请求信号（DMAReq） 该位指示 DMA 请求正在进行，用于调试。 注： 在设备模式和主机模式均可访问。
29:11	RSV	-	-	保留

10:6	TXFNUM	RW	0x0	Tx FIFO 编号 (Tx FIFO number) 使用 Tx FIFO 刷新位进行刷新的 FIFO 编号。只有在模块将 Tx FIFO 刷新位清零后，方可更 改此字段。 00000： – 主机模式下刷新非周期性 Tx FIFO – 设备模式下刷新 Tx FIFO 0 00001： – 主机模式下刷新周期性 Tx FIFO – 设备模式下刷新 Tx FIFO 1 00010：设备模式下刷新 Tx FIFO 2 ... 01111：设备模式下刷新 Tx FIFO 15 10000：在设备模式或主机模式下刷新所有的发送 FIFO。 注： 在设备模式和主机模式均可访问。
5	TXFFLSH	RS	0x0	Tx FIFO 刷新 (Tx FIFO flush) 此位选择性地刷新一个或所有的发送 FIFO，但当模块处理通信事务时无法执行该操作。 只有在确认模块当前未对 Tx FIFO 执行读写操作后，应用程序方可对此位执行写操作。使用 以下寄存器进行确认： 读 — NAK 有效中断可确保模块当前未对 FIFO 执行读操作 写 — OTG_GRSTCTL 中的 AHBIDL 位可确保模块当前未对 FIFO 执行任何写操作。 通常建议在重新配置 FIFO 时进行刷新。还建议在设备端点禁止期间进行 FIFO 刷新。应用程 序必须等待模块将此位清零，才能执行任意操作。该位需要八个时钟来清零（使用较慢的 phy_clk 或 hclk 时钟）。 注： 在设备模式和主机模式均可访问。
4	RXFFLSH	RS	0x0	Rx FIFO 刷新 (Rx FIFO flush) 应用程序可使用此位刷新整个 Rx FIFO，但必须首先确保模块当前未在处理事务。 只有在确认模块当前未对 Rx FIFO 执行读写操作后，应用程序方可对此位执行写操作。 应用程序必须等到此位清零后，方可执行其它操作。通常需要等待 8 个时钟周期（以 PHY 或 AHB 时钟中最慢的为准）。 注： 在设备模式和主机模式均可访问。
3	RSV	-	-	保留

2	FRMCNTRRST	RS	0x0	主机帧计数器复位（Host frame counter reset) 应用程序对该位执行写操作时，模块中的帧数计数器复位。帧计数器复位后，由模块发送的 下一个 SOF 的帧号为 0。 当应用程序向此位写入“1”后，可能无法回读该值，原因是模块将在几个时钟周期内将此位 清零。 注： 仅可在主机模式下访问。
1	PIUFSSFTRST	RS	0x0	PIU FS 专用控制器软复位（PIUFSSftRst) 部分软复位（Partial soft reset)，复位内部状态机，但保留枚举信息。可用于恢复一些特定的 PHY 错误。 注： 在设备模式和主机模式均可访问。
0	CSFTRST	RS	0x0	模块软复位（Core soft reset) 按如下所述将 HCLK 和 PHY 时钟域复位： 除以下各位外，将各个中断和所有 CSR 寄存器位清零： – OTG_PCGCCTL 中的 GAYEHCLK 位 – OTG_PCGCCTL 中的 STPPCLK 位 – OTG_HCFG 中的 FSLSPCS 位 – OTG_DCFG 中的 DSPD 位 – OTG_DCTL 中的 SDIS 位 – OTG_GCCFG 寄存器 – OTG_GPWRDN 寄存器 将所有模块状态机（AHB 从器件除外）复位至空闲状态，并清空所有发送 FIFO 和接收 FIFO。 在 AHB 传输的最后数据阶段结束后，尽快终止 AHB 主器件上的所有事务。立即终止 USB 上 的所有事务。 应用程序可在需要复位模块时随时对该位执行写操作。该位为自清零位，模块将在其中所有 必要逻辑复位后将该位清零，该过程需要若干个时钟的时间，具体取决于模块的当前状态。 该位一旦清零，软件必须等待至少 3 个 PHY 时钟后才可以访问 PHY 域（同步延迟)。此 外，软件还必须在确定该寄存器中的位 31 置 1（AHB 主器件空闲）后方可开始运行。 软件复位通常在两种情况下使用，一是软件开发期间，二是用户动态更改以上所列 USB 配置 寄存器中的 PHY 选择位后。用户更改 PHY 时，将为 PHY 选择相应的时钟并用于 PHY 域 中。一旦选择了新的时钟，则必须复位 PHY 域，才能保证正常运行。 注： 在设备模式和主机模式均可访问。

37.15.1.6. 中断寄存器(OTG_GINTSTS: 14h)

位域	名称	属性	复位值	描述
31	WKUPINT	RCW1	0x0	检测到恢复/远程唤醒中断（Resume/remote wakeup detected interrupt） 挂起（L2）或 LPM(L1）状态期间的唤醒中断。 – 挂起（L2）期间： 在设备模式下，当 USB 总线上检测到恢复信号时，将触发该中断。在主机模式下，当 USB 上检测到远程唤醒时，将触发该中断。 – LPM(L1）期间： USB 上检测到主机发起的恢复信号或设备发起的远程唤醒时，将触发该中断。 注： 在设备模式和主机模式均可访问。
30	SESSREQINT	RCW1	0x0	检测到会话请求/新会话中断（Session request/new session detected interrupt） 在主机模式下，当检测到来自设备的会话请求时，将触发该中断。在设备模式下，当 VBUS 在 B 外设设备的有效范围内时，将触发该中断。 注： 在设备模式和主机模式均可访问。
29	DISCONNINT	RCW1	0x0	检测到断开连接中断（Disconnect detected interrupt） 当检测到设备断开连接时触发该中断。 注： 仅可在主机模式下访问。
28	CONIDSTSCHNG	RCW1	0x0	连接器 ID 线状态更改（Connector ID status change） 当连接器 ID 线状态发生更改时，模块将该位置 1。 注： 在设备模式和主机模式均可访问。
27	LPM_INT	RCW1	0x0	LPM 中断（LPM interrupt）在设备模式下，当设备接收到 LPM 事务并以非 ERROR 应答时，将触发该中断。在主机模式下，当设备以非 ERROR 应答 LPM 事务或主机模块完成所编程次数（OTG_GLPMCFG 中的 RETRYCNT 位）的 LPM 事务时，将触发该中断。只有 OTG_GLPMCFG 中的 LPMCAP 位置 1 时，该字段才有效。 注： 在设备模式和主机模式均可访问。
26	PTXFEMP	RO	0x1	周期性 Tx FIFO 为空（Periodic Tx FIFO empty） 当周期性发送 FIFO 为半空或全空状态，且周期性请求队列中存在可写入至少一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 GAHBCFG 寄存器中的周期性 Tx FIFO 空门限位（GAHBCFG 中的 PTXFELVL 位）决定。 注： 仅可在主机模式下访问
25	HCHINT	RO	0x0	主机通道中断（Host channels interrupt） 模块将该位置 1 时，指示模块中一个通道上存在挂起的中断（在主机模式下）。应用程序必须读取 OTG_HAINT 寄存器，以确定发生中断的通道的准确编号，然后读取相应的 OTG_HCINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将 OTG_HCINTx 寄存器的相应状态位清零，之后才能将该位清零。 注： 仅可在主机模式下访问。

24	PRTINT	RO	0x0	主机端口中断（Host port interrupt） 模块将该位置 1 时，指示主机模式下 OTG_FS/OTG_HS 控制器端口的状态发生变化。应用程序必须读取 OTG_HPRT 寄存器，以确定引发此中断的确切事件。应用程序必须先将 OTG_HPRT 寄存器的相应状态位清零，之后才能将该位清零。 注： 仅可在主机模式下访问。
23	RESETDET	RCW1	0x0	检测到复位中断（Reset detected interrupt） 在设备模式下，若设备处于挂起状态，则在部分断电模式下的 USB 上检测到复位时，将触发该中断。 注： 仅可在设备模式下访问。
22	FETSUSP	RCW1	0x0	数据读取暂停（FetSusp） USB OTG HS 的数据获取挂起（Data fetch suspended for USB OTG HS），该中断仅在 DMA 模式下有效。 该中断指示，模块因 TxFIFO 空间或请求队列空间不可用而停止为 IN 端点获取数据。应用程序进行端点不匹配处理时会用到该中断。例如，在检测到端点不匹配后，应用程序将执行以下操作： – 将全局非周期性 IN NAK 握手信号置 1 – 禁止 IN 端点 – 清空 FIFO – 根据 IN 令牌序列学习队列确定令牌序列 – 重新使能端点 如果全局非周期性 IN NAK 已清零但模块尚未为 IN 端点获取数据，同时又已接收到 IN 令牌，则清零全局非周期性 IN NAK 握手信号：模块将产生“FIFO 为空时接收到 IN 令牌”中断。然后，OTG 将 NAK 响应发送到主机。为避免这种情况的发生，应用程序可以检查 OTG_GINTSTS 中的 FetSusp 中断，该中断可确保在 FIFO 存满后再将全局 NAK 握手信号清零。或者，应用程序可以在将全局 IN NAK 握手信号清零时屏蔽“当 FIFO 为空时接收到 IN 令牌”中断。 注： 仅可在设备模式下访问。
21	INCOMPLP	RCW1	0x0	未完成周期性传输（Incomplete periodic transfer） 注： 仅可在主机模式下访问。 在主机模式下，如果存在仍处于挂起状态的未完成周期性事务，而这些事务计划在当前帧期间完成，则模块会将该中断位置 1。 INCOMPISOOUT： 未完成 OUT 同步传输（Incomplete isochronous OUT transfer） 注： 仅可在设备模式下访问。 在设备模式下，模块将该中断置 1 时，指示当前帧中至少有一个同步 OUT 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断（EOPF）位一同触发。
20	INCOMPISOIN	RCW1	0x0	未完成 IN 同步传输（Incomplete isochronous IN transfer） 模块将该中断置 1 时，指示当前帧中至少有一个同步 IN 端点上的传输未完成。该中断随该寄存器中的周期性帧结束中断（EOPF）位一同触发。 注： 仅可在设备模式下访问。

19	OEPINT	RO	0x0	OUT 端点中断（OUT endpoint interrupt） 模块将该位置 1 时，指示模块中一个 OUT 端点上存在挂起的中断（在设备模式下）。应用程序必须读取 OTG_DAINT 寄存器，以确定发生中断的 OUT 端点的准确编号，然后读取相应的 OTG_DOEPINTx 寄存器，以确定引发中断的确切原因。应用程序必须先将相应 OTG_DOEPINTx 寄存器的相应状态位清零，之后才能将该位清零。 注： 仅可在设备模式下访问。
18	IEPINT	RO	0x0	IN 端点中断（IN endpoint interrupt） 模块将该位置 1 时，指示模块中一个 IN 端点上存在挂起的中断（在设备模式下）。应用程序必须读取 OTG_DAINT 寄存器，以确定发生中断的 IN 端点的准确编号，然后读取相应 的 OTG_DIEPINTx 寄 存 器，以 确 定 引 发 中 断 的 确 切 原 因。应 用 程 序 必 须 先 将 相 应 OTG_DIEPINTx 寄存器的相应状态位清零，之后才能将该位清零。 注： 仅可在设备模式下访问。
17	EPMIS	RCW1	0x0	端点不匹配中断（EPMis） 注：该中断仅在共享 FIFO 运行时有效。表示已收到非周期性端点的 IN 令牌，但非周期性传输 FIFO 顶部存在另一个端点的数据，且应用程序编程的 IN 端点不匹配计数已过期。 0x0 (INACTIVE)：无活动 0x1 (ACTIVE): 端点不匹配中断 注： 仅可在设备模式下访问。
16	RSV	-	-	保留
15	EOPF	RCW1	0x0	周期性帧结束中断（End of periodic frame interrupt） 指示当前帧已达到 OTG_DCFG 寄存器中周期性帧间隔字段（ OTG_DCFG 中的 PFIVL 位） 所指定的周期。 注： 仅可在设备模式下访问
14	ISOOUTDROP	RCW1	0x0	丢弃同步 OUT 数据包中断（Isochronous OUT packet dropped interrupt） 如果由于 Rx FIFO 空间不足，无法容纳同步 OUT 端点的最大数据包，从而导致模块无法向 Rx FIFO 写入同步 OUT 数据包，模块会将该位置 1。 注： 仅可在设备模式下访问。
13	ENUMDONE	RCW1	0x0	枚举完成（Enumeration done） 模块将该位置 1 时，指示速度枚举已完成。应用程序必须读取 OTG_DSTS 寄存器来获取枚举速度。 注： 仅可在设备模式下访问。
12	USBRST	RCW1	0x0	USB 复位（USB reset） 模块将该位置 1 时，指示在 USB 上检测到复位信号。 注： 仅可在设备模式下访问。
11	USBSUSP	RCW1	0x0	USB 挂起（USB suspend） 模块将该位置 1 时，指示在 USB 上检测到挂起状态。当数据线上的空闲状态保持一段额外的时间后，模块进入挂起状态。 注： 仅可在设备模式下访问。

10	ERLYSUSP	RCW1	0x0	早期挂起（Early suspend） 模块将该位置 1 时，指示已检测到 USB 处于空闲状态的时间达到 3 ms。 注： 仅可在设备模式下访问。
9:8	RSV	-	-	保留
7	GOUTNAKEFF	RO	0x0	全局 OUT NAK 有效（Global OUT NAK effective） 指示 OTG_DCTL 寄存器中由应用程序设置的“将全局 OUT NAK 置 1”位（OTG_DCTL 中的 SGONAK 位）已在模块中生效。通过写入 OTG_DCTL 寄存器中的“将全局 OUT NAK 清零”位（OTG_DCTL 中的 CGONAK 位），可将该位清零。 注： 仅可在设备模式下访问。
6	GINNAKEFF	RO	0x0	全局非周期性 IN NAK 有效（Global IN nonperiodic NAK effective） 指示 OTG_DCTL 寄存器中由应用程序设置的“将全局非周期性 IN NAK 置 1”位（OTG_DCTL 中的 SGINAK 位）已在模块中生效。也就是说，模块已对应用程序设置的全局 IN NAK 位进行采样，结果已生效。通过清零 OTG_DCTL 寄存器中的“将全局非周期性 IN NAK 清零”位（OTG_DCTL 中的 CGINAK 位），可将该位清零。 此中断不一定表示 USB 上已发送了一个 NAK 握手信号。 STALL 位优先级高于 NAK 位。 注： 仅可在设备模式下访问。
5	NPTXFEMP	RO	0x1	非周期性 Tx FIFO 为空（Non-periodic Tx FIFO empty） 当非周期性 Tx FIFO 为半空或全空状态，且非周期性发送请求队列中至少存在可写入一个条目的空间时，将触发该中断。该 FIFO 为半空状态还是全空状态由 GAHBCFG 寄存器中的非周期性 Tx FIFO 空门限位（GAHBCFG 中的 TXFELVL 位）决定。 注： 在设备模式和主机模式均可访问。
4	RXFLVL	RO	0x0	Rx FIFO 非空（Rx FIFO non-empty） 指示 Rx FIFO 中至少有一个数据包等待读取。 注： 在主机模式和设备模式均可访问。
3	SOF	RCW1	0x0	帧起始（Start of frame） 在主机模式下，模块将该位置 1 时，指示 USB 上已发送一个 SOF (FS) 或 Keep-Alive (LS) 信号。应用程序必须将此位置 1 才可清除该中断。 在设备模式下，模块将该位置 1 时，指示 USB 上已接收到一个 SOF 令牌。应用程序可通过读取 OTG_DSTS 寄存器来获得当前的帧编号。只有在模块以 FS 模式运行时，才会出现此中断。 注： 注意：如果上电复位后立即读取，该寄存器可能返回“1”。如果寄存器位在上电复位后立即读取“1”，则不表示已发送 SOF（主机模式下）或已接收 SOF（设备模式下）。只有在主机和设备之间建立有效连接后，此中断的读取值才有效。如果此位在上电复位后置 1，应用程序可把该位清零。 注： 在主机模式和设备模式均可访问。

2	OTGINT	RO	0x0	OTG 中断 (OTG interrupt) 模块将该位置 1 时，指示出现 OTG 协议事件。应用程序必须读取 OTG 中断状态（OTG_GOTGINT）寄存器，以确定引发此中断的确切事件。应用程序必须先将 OTG_GOTGINT 寄存器的相应状态位清零，之后才能将该位清零。 注： 在主机模式和设备模式均可访问。
1	MODEMIS	RCW1	0x0	模式不匹配中断 (Mode mismatch interrupt) 当应用程序尝试访问以下寄存器时，模块将该位置 1： – 模块运行在设备模式下访问主机模式寄存器 – 模块运行在主机模式下访问设备模式寄存器 寄存器访问在 AHB 上以 OKAY 响应结束，但该访问在内部被模块忽略并且不会影响模块运行。 注： 在主机模式和设备模式均可访问。
0	CURMOD	RO	0x0	当前工作模式 (Current mode of operation) 指示当前模式。 0：设备模式 1：主机模式 注： 在主机模式和设备模式均可访问。

37.15.1.7. 中断屏蔽寄存器(OTG_GINTMSK: 18h)

位域	名称	属性	复位值	描述
31	WKUPINTMSK	RW	0x0	检测到恢复/远程唤醒中断屏蔽 (Resume/remote wakeup detected interrupt mask) 0：屏蔽中断 1：使能中断 注： 在主机模式和设备模式均可访问
30	SESSREQINTMSK	RW	0x0	检测到会话请求/ 新会话中断屏蔽 (Session request/new session detected interrupt mask) 0：屏蔽中断 1：使能中断 注： 在主机模式和设备模式均可访问
29	DISCONNINTMSK	RW	0x0	检测到断开连接中断屏蔽 (Disconnect detected interrupt mask) 0：屏蔽中断 1：使能中断 注： 在主机模式和设备模式均可访问

28	CONIDSTSCHNGMSK	RW	0x1	连接器 ID 状态更改屏蔽（Connector ID status change mask） 0：屏蔽中断 1：使能中断 注： 在主机模式和设备模式均可访问
27	LPM_INTMSK	RW	0x0	LPM 中断屏蔽（LPM interrupt mask） 0：屏蔽中断 1：使能中断 注： 在主机模式和设备模式均可访问
26	PTXFEMPMSK	RW	0x0	周期性 Tx FIFO 空屏蔽（Periodic Tx FIFO empty mask） 0：屏蔽中断 1：使能中断 注： 仅可在主机模式下访问
25	HCHINTMSK	RW	0x0	主机通道中断屏蔽（Host channels interrupt mask） 0：屏蔽中断 1：使能中断 注： 仅可在主机模式下访问
24	PRTINTMSK	RW	0x0	主机端口中断屏蔽（Host port interrupt mask） 0：屏蔽中断 1：使能中断 注： 仅可在主机模式下访问
23	RESETDETMSK	RW	0x0	检测到复位中断屏蔽（Reset detected interrupt mask） 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
22	FETSUSPMSK	RW	0x0	USB OTG HS 的数据获取挂起屏蔽（Data fetch suspended mask for USB OTG HS） 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
21	INCOMPLPMSK	RW	0x0	未完成周期性传输中断屏蔽（Incomplete periodic transfer mask） 0：屏蔽中断 1：使能中断 注： 仅可在主机模式下访问。 IISOOXFRM： 未完成 OUT 同步传输中断屏蔽（Incomplete isochronous OUT transfer mask） 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问

20	RSV	-	-	保留
19	OEPINTMSK	RW	0x0	OUT 端点中断屏蔽（OUT endpoints interrupt mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
18	IEPINTMSK	RW	0x0	IN 端点中断屏蔽（IN endpoints interrupt mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
17	EPMISMSK	RW	0x0	端点不匹配中断屏蔽（EPMisMsk) 0x0 (MASK): 端点不匹配中断屏蔽 0x1 (NOMASK): 端点不匹配中断不屏蔽 注： 仅可在设备模式下访问
16	RSV	-	-	保留
15	EOPFMSK	RW	0x0	周期性帧结束中断屏蔽（End of periodic frame interrupt mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
14	ISOOUTDROPMSK	RW	0x0	丢弃同步 OUT 数据包中断屏蔽（Isochronous OUT packet dropped interrupt mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
13	ENUMDONEMSK	RW	0x0	枚举完成中断屏蔽（Enumeration done mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
12	USBRSTMSK	RW	0x0	USB 复位中断屏蔽（USB reset mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
11	USBSUSPMSK	RW	0x0	USB 挂起中断屏蔽（USB suspend mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问

10	ERLYSUSPMSK	RW	0x0	早期挂起中断屏蔽（Early suspend mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
9:8	RSV	-	-	保留
7	GOUTNAKEFFMSK	RW	0x0	全局 OUT NAK 生效中断屏蔽（Global OUT NAK effective mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
6	GINNAKEFFMSK	RW	0x0	全局非周期性 IN NAK 生效中断屏蔽（Global non-periodic IN NAK effective mask) 0：屏蔽中断 1：使能中断 注： 仅可在设备模式下访问
5	NPTXFEMPMSK	RW	0x0	非周期性 Tx FIFO 空中断屏蔽（Non-periodic Tx FIFO empty mask) 0：屏蔽中断 1：使能中断 注：在主机模式和设备模式均可访问
4	RXFLVLMSK	RW	0x0	接收 FIFO 非空中断屏蔽（Receive FIFO non-empty mask) 0：屏蔽中断 1：使能中断 注： 在设备模式和主机模式均可访问
3	SOFMSK	RW	0x0	帧起始中断屏蔽（Start of frame mask) 0：屏蔽中断 1：使能中断 注： 在设备模式和主机模式均可访问。
2	OTGINTMSK	RW	0x0	OTG 中断屏蔽（OTG interrupt mask) 0：屏蔽中断 1：使能中断 注： 在设备模式和主机模式均可访问。
1	MODEMISMSK	RW	0x0	模式不匹配中断屏蔽（Mode mismatch interrupt mask) 0：屏蔽中断 1：使能中断 注： 在设备模式和主机模式均可访问。
0	RSV	-	-	保留

37.15.1.8. 接收状态调试读取寄存器(OTG_GRXSTSR: 1Ch)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24:21	FN	RO	0x0	帧编号 (Frame number) 这是在 USB 上接收的数据包帧编号的 4 个最低有效位。只有当支持同步 OUT 端点时才支持此字段。 注： 仅可在设备模式下访问
20:17	PKTSTS	RO	0x0	数据包状态 (Packet status)，指示接收的数据包的状态。 在主机模式下： 0010：接收到 IN 数据包 0011： IN 传输完成（触发中断） 0101：数据同步错误（触发中断） 0111：暂停通道（触发中断） 其它值：保留 Reset:4'b0 在设备模式下： 0001：全局 OUT NAK（触发中断） 0010：接收到 OUT 数据包 0011： OUT 传输完成（触发中断） 0100： SETUP 事务完成（触发中断） 0110：接收到 SETUP 数据包 其它值：保留 Reset:4'h0 0x1 (OUTNAK)：设备模式下的全局 OUT NAK（触发中断） 0x2 (INOUTDPRX)：在主机模式下接收 IN 数据包，在设备模式下接收 OUT 数据包 0x3 (INOUTTRCOM)：在主机和设备模式下完成 IN 或 OUT 传输（触发中断） 0x4 (DSETUPCOM)：在设备模式下完成 SETUP 事务（触发中断） 0x5 (DTTOG)：主机模式下的数据切换错误（触发中断） 0x6 (DSETUPRX)：在设备模式下接收到 SETUP 数据包 0x7 (CHHALT)：通道在主机模式下停止（触发中断） Volatile: true
16:15	DPID	RO	0x0	数据 PID (DPID) 在主机模式下，表示接收到的数据包的数据 PID。在设备模式下，指示接收到的 OUT 数据包的数据 PID。 数据 PID (Data PID) 指示接收的数据包的数据 PID 类型： 00： DATA0 10： DATA1 01： DATA2 11： MDATA

14:4	BCNT	RO	0x0	字节数（BCnt） 在主机模式下，表示接收到的 IN 数据包的字节数。 在设备模式下，表示接收到的数据包的字节数。
3:0	CHNUM	RO	0x0	通道编号（Channel number） 指示当前接收的数据包所属的通道编号。 注： 仅可在主机模式下访问 端点编号（Endpoint number） 指示当前接收的数据包所属的端点编号。 注： 仅可在设备模式下访问

37.15.1.9. 接收状态读取和出栈寄存器(OTG_GRXSTSP: 20h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24:21	FN	RO	0x0	帧编号（Frame number） 这是在 USB 上接收数据包的（微）帧编号的最小有效 4 位。该字段仅在支持等时 OUT 端点时才受支持。 注： 仅可在设备模式下访问
20:17	PKTSTS	RO	0x0	数据包状态（PktSts），表示接收到的数据包的状态。 在主机模式， 4'b0010: 收到的 IN 数据包 4'b0011: IN 传输完成（触发中断） 4'b0101: 数据切换错误（触发中断） 4'b0111: 通道停止（触发中断） 其它值：保留 Reset:4'b0 在设备模式 4'b0001: 全局输出 NAK（触发中断） 4'b0010: 收到输出数据包 4'b0011: 输出传输完成（触发中断） 4'b0100:SETUP 交易完成（触发中断） 4'b0110: 收到 SETUP 数据包 其它值：保留 Reset:4'b0 0x1 (OUTNAK): 设备模式下的全局 OUT NAK（触发中断） 0x2 (INOUTDPRX): 在主机模式下接收 IN 数据包，在设备模式下接收 OUT 数据包 0x3 (INOUTTRCOM): 在主机和设备模式下完成 IN 或 OUT 传输（触发中断） 0x4 (DSETUPCOM): 在设备模式下完成 SETUP 事务（触发中断） 0x5 (DTTOG): 主机模式下的数据切换错误（触发中断） Volatile: true

16:15	DPID	RO	0x0	数据 PID (DPID) 在主机模式， 指示接收到的数据包的数据 PID。在设备模式下，指示接收到的 OUT 数据包的数据 PID。 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA
14:4	BCNT	RO	0x0	字节数（BCnt) 在主机模式下，表示接收到的 IN 数据包的字节数。 在设备模式下，表示接收到的数据包的字节数。
3:0	CHNUM	RO	0x0	通道编号（ChNum) 注： 仅可在主机模式下访问 指示当前接收的数据包所属的通道编号。 端点编号（EPNum) 注： 仅可在设备模式下访问 指示当前接收的数据包所属的端点编号。 0x0 (CHEP0): Channel or EndPoint 0 0x1 (CHEP1): Channel or EndPoint 1 0x2 (CHEP2): Channel or EndPoint 2 0x3 (CHEP3): Channel or EndPoint 3 0x4 (CHEP4): Channel or EndPoint 4 0x5 (CHEP5): Channel or EndPoint 5 0x6 (CHEP6): Channel or EndPoint 6 0x7 (CHEP7): Channel or EndPoint 7 0x8 (CHEP8): Channel or EndPoint 8 0x9 (CHEP9): Channel or EndPoint 9 0xa (CHEP10): Channel or EndPoint 10 0xb (CHEP11): Channel or EndPoint 11 0xc (CHEP12): Channel or EndPoint 12 0xd (CHEP13): Channel or EndPoint 13 0xe (CHEP14): Channel or EndPoint 14 0xf (CHEP15): Channel or EndPoint 15 Volatile: true

37.15.1.10. 接收 FIFO 大小寄存器(OTG_GRXFSIZ: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留

15:0	RXFDEP	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x8000	接收 FIFO 深度（RxFIFO Depth） 该值以 32 位字为单位。最小值为 16 ， 最大值为 32,768 。该寄存器的上电复位值在配置期间指定为最大 Rx 数据 FIFO 深度。 注： 在设备模式和主机模式均可访问。 如果在 coreConsultant 中选择了“启用动态 FIFO 大小”，则会对这些触发器进行优化，并在读取时返回开机值。如果在 coreConsultant 中选择了启用动态 FIFO 大小，则可以在该字段中写入新值。编程值不得超过开机值。

37.15.1.11. 非周期性发送 FIFO 大小寄存器(OTG_GNPTXFSIZ: 28h)

位域	名称	属性	复位值	描述
31:16	NPTXFDEP	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x8000	非周期性 TxFIFO 深度（NPTxFDep） 对于主机模式，该字段始终有效。 对于设备模式，该字段仅在 OTG_EN_DED_TX_FIFO=0 时有效。 该值以 32 位字为单位。最小值为 16 ， 最大值为 32,768 。 在 coreConsultant 配置期间，该字段属性由 "启用动态 FIFO 大小？"（OTG_DFIFO_DYNAMIC）决定： OTG_DFIFO_DYNAMIC = 0：这些触发器已优化，读数将返回上电值。OTG_DFIFO_DYNAMIC = 1：应用程序可在该字段中写入新值。编程值不得超过 coreConsultant 中设置的开机值。该字段的开机复位值在 coreConsultant 配置期间指定为最大 IN 端点 FIFO 0 深度（参数 OTG_TX_DINEP_DFIFO_DEPTH_0）。注： 仅可在主机模式下访问
15:0	NPTXFSTADDR	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x8000	非周期性传输 RAM 起始地址（NPTxFStAddr） 对于主机模式，该字段始终有效。 该字段包含非定期发送 FIFO RAM 的内存起始地址。 该字段在 coreConsultant 配置期间由 "启用动态 FIFO 大小？"(OTG_DFIFO_DYNAMIC)决定：OTG_DFIFO_DYNAMIC = 0 这些触发器经过优化，读取时会返回上电值。 OTG_DFIFO_DYNAMIC = 1 应用程序可在该字段中写入新值。编程值不得超过 coreConsultant 中设置的开机值。 编程值不得超过 coreConsultant 中设置的开机值。 该字段的上电复位值在 coreConsultant 配置期间由最大 Rx 数据 FIFO 深度（参数 OTG_RX_DFIFO_DEPTH）指定。

37.15.1.12. 非周期性发送 FIFO/队列状态寄存器(OTG_GNPTXSTS: 2Ch)

位域	名称	属性	复位值	描述

31	RSV	-	-	保留
30:24	NPTXQTOP	RO	0x0	非周期性发送请求队列顶部（Top of the non-periodic transmit request queue) 非周期性发送请求队列中 MAC 目前正在处理的条目。 位 30:27：通道/端点编号（Channel/endpoint number) 位 26:25： 00： IN/OUT 令牌 01：长度为零的发送数据包（设备 IN/主机 OUT） 11：通道停止命令 位 24：结束（所选通道/端点的最后一个条目）
23:16	NPTXQSPCAVAIL	RO	0x8	非周期性发送请求队列可用空间（Non-periodic transmit request queue space available) 指示非周期性发送请求队列中的可用空闲空间大小。该队列既包含 IN 请求，又包含 OUT 请求。 0：非周期性发送请求队列已满 1： 1 个位置可用 2： 2 个位置可用 n： n 个位置可用（0<= n<=8) 其它值：保留
15:0	NPTXFSPCAVAIL	RO	0x8000	非周期性 Tx FIFO 可用空间（Non-periodic Tx FIFO space available) 指示非周期性 Tx FIFO 中的可用空闲空间大小。以 32 位字为单位。 0：非周期性 TxFIFO 已满 1： 1 个字可用 2： 2 个字可用 n： n 个字可用（其中， 0 n 512） 其它值：保留

37.15.1.13. 通用 IO 寄存器(OTG_GGPIO: 38h)

位域	名称	属性	复位值	描述
31:16	GPO	RW	0x0	通用输出（GPO) 该字段从内核 gp_o[15:0]输出。应用程序可对该字段进行编程，以确定 gp_o[15:0]输出上的相应值。
15:0	GPI	RO	0x0	通用输入（GPI) 该字段的读取值反映了 gp_i[15:0] 内核输入值。

37.15.1.14. 用户 ID 寄存器(OTG_GUID: 3Ch)

位域	名称	属性	复位值	描述
31:0	GUID	RW	0x1235678	用户 ID（UserID） 应用程序可编程的 ID 字段。Reset： 可配置

37.15.1.15. 用户硬件配置寄存器 3(OTG_GHWCFG3: 4Ch)

位域	名称	属性	复位值	描述
31	DESCDMA	RO	0x0	DFIFO 深度（DfifoDepth - EP_LOC_CNT） 该值以 32 位字为单位。最小值为 32，最大值为 32,768。 注：该字段使用 OTG_DFIFO_DEPTH 参数配置。
30	DESCDMAENABLED	RO	0x0	Scatter/Gather DMA 配置 1'b0：非动态配置 1'b1：动态配置 注：该字段使用 OTG_EN_DESC_DMA 参数配置。
29:26	INEPS	RO	0xf	设备模式 IN 端点（包括控制端点）的数量（INEps） 0：1 个 IN 端点 1：2 个 IN 端点 . 15：16 个 IN 端点
25	DEDFIFOMODE	RO	0x1	为设备 IN 端点启用专用传输 FIFO（DedFifoMode） 1'b0：未启用专用传输 FIFO 操作。 1'b1：启用专用传输 FIFO 操作。
24	SESSENDFLTR	RO	0x1	会话结束过滤器（SessEndFltr） 1'b0：无过滤器 1'b1：有过滤器
23	BVALIDFLTR	RO	0x1	b_valid 过滤器（BValidFltr） 1'b0：无过滤器 1'b1：有过滤器
22	AVALIDFLTR	RO	0x1	a_valid 过滤器（AValidFltr） 1'b0：无过滤器 1'b1 ：有过滤器
21	VBUSVALIDFLTR	RO	0x1	VBUS 有效过滤器（VBusValidFltr） 1'b0：无滤波器 1'b1：有滤波器

20	IDDGFLTR	RO	0x1	IDDIG 过滤器（IddgFltr） 1'b0：无滤波器 1'b1：有滤波器
19:16	NUMCTLEPS	RO	0x0	端点 0 以外的设备模式控制端点数量（NumCtlEps） 范围：0-15 0x0 (ENDPT0): End point 0 0x1 (ENDPT1): End point 1 0x2 (ENDPT2): End point 2 0x3 (ENDPT3): End point 3 0x4 (ENDPT4): End point 4 0x5 (ENDPT5): End point 5 0x6 (ENDPT6): End point 6 0x7 (ENDPT7): End point 7 0x8 (ENDPT8): End point 8 0x9 (ENDPT9): End point 9 0xa (ENDPT10): End point 10 0xb (ENDPT11): End point 11 0xc (ENDPT12): End point 12 0xd (ENDPT13): End point 13 0xe (ENDPT14): End point 14 0xf (ENDPT15): End point 15
15:14	PHYDATAWIDTH	RO	0x2	UTMI+ PHY/ULPI 转内部 UTMI+ 封装器数据宽度（PhyDataWidth） 使用 ULPI PHY 时，内部封装器将 ULPI 转换为 UTMI+。 2'b00：8 位 2'b01：16 位 2'b10：8/16 位，软件可选 其他：保留
13	ENHANCEDLPMSUPT	RO	0x1	增强型 LPM 支持（EnhancedLPMSupt） 该位表示控制器支持以下行为： 基于 FIFO 状态的 L1 输入行为 TX FIFO 即使 ISOC IN TX FIFO 不为空，也接受 L1 请求。如果非定期 TX FIFO 不为空，则拒绝 L1 请求。确保控制器处于 L1 状态时，应用程序可以刷新 TX FIFO。 RX FIFO 即使 RX FIFO（周期性和非周期性通用）不是空的，也接受 L1 请求。接受 L1 请求，但延迟 SLEEPM 断言，直到 RX SINK 缓冲器为空。 如果任何控制端点上正在进行控制传输，则防止 L1 输入。即使 PHY 时钟处于门控状态，也能清空 TxFIFO。 0x1 (ENABLED)： 启用增强型 LPM 支持

12	ACGSUPT	RO	0x0	主动时钟门控 该位表示控制器在无 USB 和 AHB 流量期间支持动态（开关）功率降低。 1'b0：未启用主动时钟门控。 1'b1：启用主动时钟门控。
11	IPGISOCSUPT	RO	0x1	包间空隙 ISOC OUT 最坏情况支持（ipgisocSupt) 该位表示控制器支持 ISOC OUT 令牌后任何令牌的 Rx 后 Rx 包间间隙（IPG）最坏情况（32 位时间），符合 UTMI 规范。如果不支持该功能，当 ISOC OUT 令牌之后的任何令牌具有最坏情况 IPG 时，控制器将检测不到被跟随的令牌。在不支持此功能的情况下，控制器的最坏 IPG 取决于 AHB 和 PHY 时钟频率。 0x0(DISABLED)：已禁用数据包间隙 ISOC OUT 最坏情况支持 0x1(ENABLED)：已启用数据包间隙 ISOC OUT 最坏情况支持（默认值） Volatile: true
10	SERVINTFLOW	RO	0x0	服务间隔流 该位表示控制器支持 ISOC IN EP 基于服务间隔的调度流。 0x0(DISABLED)：不支持服务间隔流 0x1(ENABLED)：支持服务间隔流 Volatile: true
9	ENHANCEDLPMSUPT1	RO	0x1	增强型 LPM 支持（EnhancedLPMSupt1） 该位表示控制器支持基于 FIFO 状态的 L1 输入。即使批量/中断 TxFIFO 不为空，也接受 L1 请求。 0x0(DISABLED)：即使非定期（批量/中断）TxFIFO 不为空，也拒绝 L1 请求。 0x1(ENABLED)：即使非定期（批量/中断）TxFIFO 不为空，仍接受 L1 请求。
8	RSV	-	-	保留
7	EXTENDEDHIBERNATION	RO	0x0	扩展休眠功能 1'b0：未启用扩展休眠功能 1'b1：启用扩展休眠功能
6	HIBERNATION	RO	0x0	休眠功能 1'b0：未启用休眠功能 1'b1：启用休眠功能
5	AHBFREQ	RO	0x1	小于 60 MHz 的最小 AHB 频率（AhbFreq) 0x0(DISABLED)：最低 AHB 频率超过 60 MHz 0x1(ENABLED)：最低 AHB 频率低于 60 MHz
4	PARTIALPWRDN	RO	0x1	部分断电（PartialPwrDn) 1'b0：未启用部分断电功能 1'b1：启用部分断电

3:0	NUMDEVPERIOEPS	RO	0x0	设备模式周期性 IN 端点数量（NumDevPerioEps） 范围： 0-15

37.15.1.16. LPM 配置寄存器(OTG_GLPMCFG: 54h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29	LPM_RESTORESLPSTS	RW	0x0	LPM 恢复休眠状态（LPM_RestoreSlpSts）当应用对内核进行电源关断（部分掉电/休眠）时，应用需要对该位进行编程，以恢复内核的 LPM 状态。 在恢复过程中，应用程序需要根据从主机接收到的 BESL 值，决定进入浅睡眠（仅时钟门控）还是深睡眠（电源门控），对该位进行编程。 1'b0：应用程序根据主机提供的 BESL 值将内核置于浅睡眠状态 1'b1：应用程序根据主机提供的 BESL 值将内核置于深度休眠状态
28	LPM_ENBESL	RW	0x0	LPM Enable BESL (LPM_EnBESL) 使能尽力服务延迟 (Enable best effort service latency) 该位用于使能 LPM 勘误表中定义的 BESL 功能： 0：模块按照以下文档中的说明工作： 基于 USB 2.0 规范的 USB 2.0 链路层电源管理补充工程变更通知， 2007 年 7 月 16 日 1：模块按照 LPM 勘误表中的说明工作： USB 2.0 ECN 的勘误表：链路层电源管理 (LPM) - 2007 年 7 月 注： 本文档只考虑（ LPM 勘误表中说明的）更新行为，因此 ENBESL 位应由应用程序软件置 1。
27:25	LPM_RETRYCNT_STS	RO	0x0	LPM 重试计数状态 (LPM retry count status) 当前 LPM 序列仍要发送的 LPM 主机重试次数 注： 仅可在主机模式下访问。
24	SNDLPM	RW	0x0	发送 LPM 事务 (Send LPM transaction) 当应用程序软件将该位置 1 时，将发送包含两个令牌 EXT 和 LPM 的 LPM 事务。从设备接收到有效响应（ STALL、 NYET 或 ACK）或模块发送完编程的 LPM 重试次数后，硬件即将该位清零。 注： 只有当主机连接到本地端口时，该位才可置 1。 注：仅可在主机模式下访问。

23:21	LPM_RETRY_CNT	RW	0x0	LPM 重试计数 (LPM retry count) 主机模式： 如果设备响应为 ERROR，主机将执行的额外 LPM 重试次数，直到收到有效的设备响应 STALL/NYET/ACK。 设备模式： LPM 接受控制（LPM_Accept_Ctrl) LPM_Accept_Ctrl[1]： 应用程序可使用该位在单次控制传输的多个阶段之间拒绝 LPM 令牌（NYET)
20:17	LPM_CHNL_INDX	RW	0x0	LPM 通道索引 (LPM Channel Index) 主机模式： 向本地设备发送 LPM 事务时，必须应用到 LPM 事务的通道编号。基于 LPM 通道编号，模块可自动向 LPM 事务插入相应通道中编程的设备地址和端点号。 设备模式： LPM 接受控制（LPM_Accept_Ctrl） LPM_Accept_Ctrl[0] (LPM_Chnl_Indx[3])：即使 INTR 端点 TxFIFO 中存在数据，应用程序也可以使用该位接受 LPM 令牌。该位仅适用于专用 TX FIFO 配置（OTG_EN_DED_TX_FIFO=1)。 1'b0：拒绝（NYET）LPM 令牌，当 INTR 端点的 txfifo 中存在数据时。 1'b1：接受（ACK） LPM 令牌，当 txfifo 中存在 INTR 端点的数据时。 Values: 0x0 (CH0): Channel 0 0x1 (CH1): Channel 1 0x2 (CH2): Channel 2 0x3 (CH3): Channel 3 0x4 (CH4): Channel 4 0x5 (CH5): Channel 5 0x6 (CH6): Channel 6 0x7 (CH7): Channel 7 0x8 (CH8): Channel 8 0x9 (CH9): Channel 9 0xa (CH10): Channel 10 0xb (CH11): Channel 11 0xc (CH12): Channel 12 0xd (CH13): Channel 13 0xe (CH14): Channel 14 0xf (CH15): Channel15
16	L1RESUMEOK	RO	0x0	睡眠状态恢复正常 (Sleep State Resume OK) 表示设备或主机可以从睡眠状态启动恢复。该位在 LPM 睡眠 (L1) 状态下有效。它在睡眠模式下经 50 us (TL1Residency) 延时后置 1。该位在 SLPSTS 为 0 时复位。 1：应用程序或主机可以从睡眠状态启动恢复 0：应用程序或主机无法从睡眠状态启动恢复

15	SLPSTS	RO	0x0	端口睡眠状态（Port sleep status） 设备模式： 只要 USB 总线上存在睡眠条件，该位就会置 1。当 ACK 响应发送给 LPM 事务且 TL1TokenRetry 定时器过期时，模块将进入睡眠模式。要停止 PHY 时钟，应用程序必须将 OTG_PCGCCTL 中的 STPPCLK 位置 1，这将触发 PHY 挂起输入信号。应用程序必须依靠 LPMRSP 中的 SLPSTS 而不是 ACK 来确认切换到睡眠状态。 出现以下情况时，模块会退出睡眠状态： – USB 线上有活动时 – 当应用程序对 DCTL 中的 RWUSIG 位执行写操作或者复位或软断开设备时。 主机模式： 模块通过来自设备的 ACK 响应成功将 LPM 事务发送给本地端口后，主机将切换到睡眠（L1）状态。此位的读取值反映该端口的当前睡眠状态。在以下事件后，模块会将该位清零： – 模块检测到远程 L1 唤醒信号， – 应用程序将 OTG_HPRT 寄存器中的 PRST 位或 PRES 位置 1，或者 – 应用程序将模块中断寄存器中的“检测到 L1 恢复/远程唤醒中断”位或“检测到断开连接中断”位（分别为 GINTSTS 中的 WKUPINT 或 DISCINT 位）置 1。 0：模块未处于 L1 1：模块处于 L1
14:13	COREL1RES	RO	0x0	LPM 响应（LPM response） 设备模式： 这两位反映模块对所接收的 LPM 事务的响应。 主机模式： 从本地设备接收、用于 LPM 事务的握手响应 11： ACK 10： NYET 01： STALL 00： ERROR（无握手响应）

12:8	HIRD_THRES	RW	0x0	BESL/HIRD 阈值（HIRD_Thres） 设备模式： 注意：当与控制器（主机或设备）相关的 ULPI/UTMI PHY 不支持 LPM 休眠功能时钟门控时，建议将 HIRD_Thres[12] 位保持在禁用状态（= 1'b0）。这样可确保 ULPI 封装器处于启用模式，并能检测到任何唤醒事件。 EnBESL = 1'b0：当 HIRD 值大于或等于该字段 HIRD_Thres[3:0] 中定义的值且 HIRD_Thres[4] 设置为 1b1 时，内核将 PHY 置于 L1 深度低功耗模式（通过内核断言 L1SuspendM） EnBESL = 1'b1： 当 BESL 值大于或等于该字段 BESL_Thres[3:0] 中定义的值且 BESL_Thres[4] 设置为 1'b1 时，内核将 PHY 置于 L1 深度低功耗模式（通过内核断言 L1SuspendM） DCTL.DeepSleepBESLReject = 1'b1： 在设备启动的恢复中，内核希望主机在与 HIRD_Thres[3:0]中指定的 L1 退出时间相对应的 BESL 值内恢复对设备的服务。当接收到的 LPM 令牌中的 HIRD 大于 HIRD 阈值时，设备发送 NYET 响应。 主机模式： 当 HIRD_Thres[4]设置为 1'b1 时，内核将 PHY 置于 L1 深度低功耗模式（通过内核断言 L1SuspendM）。HIRD_Thres[3:0] 指定主机 (TL1HubDrvResume2) 在检测到设备启动恢复时在 USB 总线上反映恢复信号的时间。 要区分深度睡眠和浅度睡眠，需要进行 HIRD 大于或等于 HIRD 阈值的比较。为了区分 LPM 令牌的 NYET 或 ACK 响应，使用了 HIRD 大于 HIRD 阈值比较法。 当 EnBESL = 1'b0 时，在主机模式下，HIRD_Thres 的编程值不应大于 4'b1100，因为这将超过 TL1HubDrvResume2 的最大值。 当 EnBESL = 1'b1 时，在主机模式下，HIRD_Thres 的编程值不应大于 4'b0110，因为这将超过 TL1HubDrvResume2 的最大值。 有关其他详细信息，请参阅数据库中的 "GLPMCFG.HIRD_Thres 寄存器字段的其他详细信息 "部分。

7	ENBLSLPM	RW	0x0	utmi_sleep_n (EnblSlpM) 用于 ULPI 接口：当处于 L1 状态时，应用程序使用该位控制 PHY 的 utmi_sleep_n 断言。对于主机，该位仅在本地设备模式下有效。 1'b0：内核的 utmi_sleep_n 断言未传输到外部 PHY。 1'b1：内核的 utmi_sleep_n 断言传输到外部 PHY。 注：配置 ULPI 接口时，启用该位将导致写入 ULPI 功能控制寄存器的第 7 位。Synopsys ULPI PHY 支持写入该位，并且在 L1 状态下，当无法断言 utmi_l1_suspend_n 时，会断言 SleepM。 当与控制器（主机或设备）相关的 ULPI/ UTMI PHY 不支持 LPM 睡眠功能时钟门控时，建议将 EnblSlpM 位保持在禁用状态（= 1'b0）。这可确保 ULPI 封装处于启用模式，并能检测到任何唤醒事件。 对于所有其他接口： 应用程序使用该位控制 PHY 在 L1 状态下的 utmi_sleep_n。对于主机，该位仅在本地设备模式下有效。 1'b0：内核的 utmi_sleep_n 断言不会传输到外部 PHY。 1'b1：当 utmi_l1_suspend_n 不能被断言时，来自内核的 utmi_sleep_n 断言将被传输到外部 PHY。 注： 在主机模式和设备模式均可访问
6	BREMOTEWAKE	OTG_MODE!=3 & OTG_MODE!=4 ? RW:RO	0x0	远程唤醒 (bRemoteWake) 主机模式： 将在 LPM 事务的 wIndex 字段中发送的远程唤醒值。 设备模式： 此字段为只读。当向 LPM 事务发送 ACK/NYET/STALL 响应时，该字段将随接收到的 LPM 令牌 bRemoteWake bmAttribute 更新。 1'b0：远程唤醒已禁用 1'b1：在主机或设备模式下，该字段的值为远程唤醒值 注： 在主机模式和设备模式均可访问

5:2	HIRD	OTG_MODE!=3 & OTG_MODE!=4 ? RW:RO	0x0	主机启动的恢复持续时间（HIRD） EnBESL = 1'b0 主机启动的恢复时间。 主机模式： 将在 LPM 事务中发送的 HIRD 值。该值也用于主机启动恢复的持续时间 TL1HubDrvResume1。 设备模式： 此字段为只读字段，在向 LPM 事务发送 ACK/NYET/STALL 响应时，使用 Received LPM Token HIRD bmAttribute（接收到的 LPM 令牌 HIRD bmAttribute）进行更新。 If HIRD[3:0], 4'b0000, THIRD(us) = 50 4'b0001, THIRD(us) = 125 4'b0010, THIRD(us) = 200 4'b0011, THIRD(us) = 275 4'b0100, THIRD(us) = 350 4'b0101, THIRD(us) = 425 4'b0110, THIRD(us) = 500 4'b0111, THIRD(us) = 575 4'b1000, THIRD(us) = 650 4'b1001, THIRD(us) = 725 4'b1010, THIRD(us) = 800 4'b1011, THIRD(us) = 875 4'b1100, THIRD(us) = 950 4'b1101, THIRD(us) = 1025 4'b1110, THIRD(us) = 1100 4'b1111, THIRD(us) = 1175 EnBESL = 1'b1 最佳服务延迟（BESL）。 主机模式： 将在 LPM 事务中发送的 BESL 值。该值还用于启动持续时间为 TL1HubDrvResume1 的主机启动恢复。 设备模式： 当向 LPM 事务发送 ACK/NYET/STALL 响应时，该字段用接收到的 LPM 令牌 BESL bmAttribute 更新。 If BESL[3:0], 4'b0000, TBESL(us) = 125 4'b0001, TBESL(us) = 150 4'b0010, TBESL(us) = 200 4'b0011, TBESL(us) = 300 4'b0100, TBESL(us) = 400 4'b0101, TBESL(us) = 500 4'b0110, TBESL(us) = 1000

				4'b0111, TBESL(us) = 2000 4'b1000, TBESL(us) = 3000 4'b1001, TBESL(us) = 4000 4'b1010, TBESL(us) = 5000 4'b1011, TBESL(us) = 6000 4'b1100, TBESL(us) = 7000 4'b1101, TBESL(us) = 8000 4'b1110, TBESL(us) = 9000 4'b1111, TBESL(us) = 10000
1	APPL1RES	RW	0x0	由应用程序编程的 LPM 响应（AppL1Res） 设备应用软件预编程的 LPM 令牌握手响应。响应取决于 GLPMCFG.LPMCap。如果 GLPMCFG.LPMCap 为 1'b0，则内核作为不支持 LPM 的设备运行，不响应任何 LPM 事务。如果 GLPMCFG.LPMCap 为 1'b1，则内核响应如下： 1: ACK 尽管 ACK 已预先编程，但只有在 LPM 事务成功时，内核才会响应 ACK。LPM 交易成功的条件是： - EXT 令牌和 LPM 令牌中都没有 PID/CRC5 错误（否则为 ERROR） - 在 LPM 事务中收到有效的 bLinkState = 0001B (L1)（否则为 STALL）。 - 传输队列中没有待处理数据（否则为 NYET） 0: NYET 当出现以下情况时，预编程软件位将被覆盖，以响应 LPM 令牌： - 收到的 bLinkState 不是 L1（STALL 响应） - 由于损坏（ERROR 响应），在 LPM 标记数据包中检测到错误。 0x0 (NYET_RESP)：当检测到 LPM 令牌数据包因损坏而出错时，内核将以 NYET 响应 0x1 (ACK_RESP)：内核仅在 LPM 交易成功时响应 ACK。 注： 仅可在设备模式下访问
0	LPMCAP	RW	0x0	LPM-Capable (LPMCap) 应用程序使用该位控制控制器的 LPM 功能。如果内核作为不支持 LPM 的主机运行，则无法请求所连接的设备/集线器激活 LPM 模式。如果内核作为不支持 LPM 的设备运行，则无法响应任何 LPM 事务。如果 GLPMCFG.LPMCap 为 1'b0，则软件不得设置 GLPMCFG 寄存器中的任何其余字段，这些字段应保持其复位值。 1'b0: 未启用 LPM 功能。 1'b1: 启用 LPM 功能。

37.15.1.17. 全局 DFIFO 配置寄存器(OTG_GDFIFOCFG: 5Ch)

位域	名称	属性	复位值	描述
31:16	EPINFOBASEADDR	RW	0x7fe0	EPInfoBaseAddr 该字段提供 EP 信息控制器的起始地址。
15:0	GDFIFOCFG	RW	0x8000	GDFIFOCfg 该字段用于对 DFIFO 大小进行动态编程。只有当应用程序向该寄存器编程一个非零值时，该值才会生效。编程值必须符合 "FIFO RAM 分配 "中描述的准则。如果 FIFO 大小编程错误，内核没有任何纠正逻辑。

37.15.1.18. 主机周期性发送 FIFO 大小寄存器(OTG_HPTXFSIZ: 100h)

位域	名称	属性	复位值	描述
31:16	PTXFSIZE	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x8000	主机周期性传输 Tx FIFO 深度 (Host periodic Tx FIFO depth) 以 32 位字为单位。 最小值为 16
15:0	PTXFSTADDR	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x0	主机周期性传输 Tx FIFO 起始地址 (Host periodic Tx FIFO start address) 此字段用于配置周期性发送 FIFO RAM 的存储器起始地址。

37.15.1.19. 设备 IN 端点发送 FIFO 大小寄存器 (OTG_DIEPTXFx: 104h+(x-1)*04h, x=1~15)

位域	名称	属性	复位值	描述
31:16	INEPNTXFDEP	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x8000	IN 端点 Tx FIFO 深度 (IN endpoint Tx FIFO depth) 以 32 位字为单位。 最小值为 16
15:0	INEPNTXFSTADDR	OTG_DFIFO_DYNAMIC == 1 ? RW:RO	0x0	IN 端点发送 FIFOx RAM 起始地址 (IN endpoint FIFOx transmit RAM start address) 此字段包含 IN 端点发送 FIFOx 的存储器起始地址。该地址必须与 32 位存储器位置对齐。

37.15.2. 主机模式寄存器列表

偏移	名称	复位值	描述
0x400	OTG_HCFG		主机配置寄存器
0x404	OTG_HFIR		主机帧间隔寄存器

0x408	OTG_HFNUM		主机帧编号/帧剩余时间寄存器
0x410	OTG_HPTXSTS		主机周期性发送 FIFO/队列状态寄存器
0x414	OTG_HAINT		主机全体通道中断寄存器
0x418	OTG_HAINTMSK		主机全体通道中断屏蔽寄存器
0x440	OTG_HPRT		主机端口控制和状态寄存器
0x500+(x*0x20)	OTG_HCCHARx		主机通道 x 特性寄存器（x=0~15，其中 x=通道编号）
0x504+(x*0x20)	OTG_HCSPLTx		主机通道 x 分离控制寄存器（x=0~15，其中 x=通道编号）
0x508+(x*0x20)	OTG_HCINTx		主机通道 x 中断寄存器（x=0~15，其中 x=通道编号）
0x50c+(x*0x20)	OTG_HCINTMSKx		主机通道 x 中断屏蔽寄存器（x=0~15，其中 x=通道编号）
0x510+(x*0x20)	OTG_HCTSIZx		主机通道 x 传输大小寄存器（x=0~15，其中 x=通道编号）
0x514+(x*0x20)	OTG_HCDMAx		主机通道 x DMA 地址寄存器（x=0~15，其中 x=通道编号）

37.15.2.1. 主机配置寄存器(OTG_HCFG: 400h)

位域	名称	属性	复位值	描述
31	MODECHTIMEN	RW	0x0	模式更改就绪定时器（ModeChTimEn） 该位用于启用/禁用主机内核在复位结束时等待 200 PHY 时钟周期，以便在挂起或 LPM 之后将 PHY 的 opmode 信号更改为 00。 1'b0：主机内核等待 200 个 PHY 时钟周期或在恢复结束时等待线态为 SE0，才能将操作模式从 2'b10 更改为 2'b00 1'b1：主机内核只等待 SE0 的线状态恢复结束，将运行模式从 2'b10 更改为 2'b00
30:16	RSV	-	-	保留
15:8	RESVALID	RW	0x2	恢复验证周期（ResValid） 该字段仅在设置 HCFG.Ena32KHzS 时有效。它将控制内核从挂起状态恢复时的恢复周期。当设置该值时，内核将计算 "ResValid "时钟周期数，以检测有效恢复。
7	ENA32KHZS	RW	0x0	32 KHz 暂停模式（Ena32KHzS） 只有在选择 FS PHY 接口时才能设置该位。否则，该位必须设置为 0。当选择 FS PHY 接口并设置该位时，内核会将挂起期间的 PHY 时钟从 48 MHz 切换为 32 KHz。 1'b0：禁用 32 KHz 暂停模式 1'b1：启用 32 KHz 暂停模式
6:3	RSV	-	-	保留
2	FSLSSUPP	RW	0x0	仅支持 FS 和 LS (FS- and LS-only support) 应用程序使用此位控制模块的枚举速度。使用此位，应用程序可使模块工作为 FS 主机，即使所连接的设备支持 HS 通信也是如此。请勿在初始编程后更改此字段。 1：仅限 FS/LS，即使所连接设备可支持 HS（只读）

1:0	FSLSPCLKSEL	RW	0x0	FS/LS PHY 时钟选择（FSLSPclkSel） 当模块处于 FS 主机模式时： • 2'b00：PHY 时钟运行在 30/60 MHz • 2'b01：PHY 时钟运行在 48 MHz • 其他：保留 当模块处于 LS 主机模式时： • 2'b00：PHY 时钟运行在 30/60 MHz。 当未选择 UTMI+/ULPI PHY 低功耗模式时，使用 30/60 MHz。 • 2'b01：PHY 时钟运行在 48 MHz。选择 UTMI + PHY 低功耗模式时，如果 PHY 在 LS 模式下提供 48 MHz 时钟，请使用 48MHz。 • 2'b10：PHY 时钟以 6 MHz 运行。在 USB 1.1 FS 模式下，选择 UTMI + PHY 低功耗模式时使用 6 MHz，并且 PHY 在 LS 模式下提供 6 MHz 时钟。如果在 LS 模式下选择 6 MHz 时钟，则必须进行软复位。 • 2'b11：保留 注： • 当模块处于 FS 模式时，内部和外部时钟具有相同的频率。 • 当模块处于 LS 模式时， - 如果 FSLSPclkSel = 2'b00：内部和外部时钟具有相同的频率 - 如果 FSLSPclkSel = 2'b10：内部时钟是外部 48 MHz 时钟的 8 分频

37.15.2.2. 主机帧间隔寄存器(OTG_HFIR: 404h)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	HFIRRLDCTRL	RW	0x0	重载控制（Reload control） 该位允许在运行期间动态重载 HFIR 寄存器。 0：无法动态重载 HFIR 1：运行期间可动态重载 HFIR。 该位需要在初始配置期间编程，并且不得在运行期间更改其值。
15:0	FRINT	RW	0xea60	帧间隔（FrInt） 应用程序为该字段编程的值指定两个连续 SOF（FS）或微 SOF（HS）或 Keep-Alive 标记（HS）之间的间隔。该字段包含构成所需帧间隔的 PHY 时钟数。当 PHY 时钟频率为 60 MHz 时，该字段中设置的默认值适用于 FS 操作。只有在主机端口控制和状态寄存器（HPRT.PrtEnaPort）的端口启用位被设置后，应用程序才能向该寄存器写入数值。如果没有编程值，内核将根据主机配置寄存器（HCFG.FSLSPclkSel）的 FS/LS PHY 时钟选择字段中指定的 PHY 时钟计算值。初始配置后，请勿更改该字段的值。 125 s * (PHY clock frequency for HS) 1 ms * (PHY clock frequency for FS/LS)

37.15.2.3. 主机帧编号/帧剩余时间寄存器(OTG_HFNUM: 408h)

位域	名称	属性	复位值	描述
31:16	FRREM	RO	0x0	帧剩余时间（Frame time remaining） 指示当前帧的剩余时间（以 PHY 时钟数为单位）。每过去 1 个 PHY 时钟，此字段递减 1。 当值达到零时，此字段将重新装载帧间隔寄存器中的值，并由模块在 USB 上发送一个新 SOF。
15:0	FRNUM	RO	0x3fff	帧编号（Frame number） 当在 USB 上发送 1 个新 SOF 时此字段的值将递增 1，当达到 0x3FFF 时会清零。

37.15.2.4. 主机周期性发送 FIFO/队列状态寄存器(OTG_HPTXSTS: 410h)

位域	名称	属性	复位值	描述
31:24	PTXQTOP	RO	0x0	周期性传输发送请求队列顶部（Top of the periodic transmit request queue） 指示周期性传输 Tx 请求队列中 MAC 当前正在处理的项。该寄存器用于调试。 位 31：奇数/偶数帧（Odd/Even frame） 0：以偶数帧发送 1：以奇数帧发送 位 30:27：通道/端点编号（Channel/endpoint number） 位 26:25：类型（Type） 00： IN/OUT 01：零长度数据包 11：禁止通道命令 位 24：结束（所选通道/端点的最后一个条目）
23:16	PTXQSPCAVAIL	RO	0x8	周期性传输发送请求队列可用空间（Periodic transmit request queue space available） 指示可供写入的周期性传输发送请求队列的空闲位置的数量。该队列既包含 IN 请求，又包含 OUT 请求。 00：周期性发送请求队列已满 01： 1 个位置可用 10： 2 个位置可用 bxn： n 个位置可用（0 n 16） 其它值：保留

15:0	PTXFSPCAVAIL	RO	0x8000	周期性传输发送数据 FIFO 可用空间 (Periodic transmit data FIFO space available) 指示可供写入的周期性传输 Tx FIFO 的空闲位置的数量。以 32 位字为单位。 0000：周期性 Tx FIFO 已满 0001： 1 个字可用 0010： 2 个字可用 bxn： n 个字可用（其中 0<=n<= 32768） 其它值：保留

37.15.2.5. 主机全体通道中断寄存器(OTG_HAINT: 414h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	HAINT	RO	0x0	通道中断 (Channel interrupts) 每个通道一位：通道 0 对应位 0，通道 15 对应位 15

37.15.2.6. 主机全体通道中断屏蔽寄存器(OTG_HAINTMSK:418h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	HAINTMSK	RW	0x0	通道中断屏蔽 (Channel interrupt mask) 0：屏蔽中断 1：使能中断 每个通道一位：通道 0 对应位 0，通道 15 对应位 15

37.15.2.7. 主机端口控制和状态寄存器(OTG_HPRT: 440h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18:17	PRTSPD	RO	0x0	端口速度 (Port speed) 指示连接到该端口的设备的速度。 01：全速 10：低速 11：保留 00：高速

16:13	PRTTSTCTL	RW	0x0	端口测试控制（Port test control） 应用程序向该字段写入一个非零值，以将端口置于测试模式，同时端口上会产生对应模式的信号。 0000：禁止测试模式 0001： Test_J 模式 0010： Test_K 模式 0011： Test_SE0_NAK 模式 0100： Test_Packet 模式 0101： Test_Force_Enable 其它值：保留
12	PRTPWR	RW	0x0	端口电源（Port power） 应用程序使用该字段控制该端口的电源，且发生过流情况时，模块会将该位清零。 0：掉电 1：上电
11:10	PRTLNSTS	RO	0x0	端口线状态（Port line status） 指示 USB 数据线的当前逻辑电平 位 10： OTG_FS_DP 的逻辑电平 位 11： OTG_FS_DM 的逻辑电平
9	RSV	-	-	保留
8	PRTRST	RW	0x0	端口复位（Port reset） 应用程序将该位置 1 时，会在该端口上启动复位序列。应用程序必须为复位周期定时，并在复位序列完成后将该位清零。 0：端口未处于复位状态 1：端口处于复位状态 应用程序必须将该位置 1 并最少保持 10 ms，以在端口上启动复位。除所需的最少持续时间之外，在将该位清零前，应用程序可将该位置 1 的状态再保持 10 ms，即使 USB 标准并没有设置最大限制。 高速： 50 ms 全速/低速： 10 ms
7	PRTSUSP	RS	0x0	端口挂起（Port suspend） 应用程序将此位置 1 以将此端口置于挂起模式。只有此位置 1 时，模块才会停止发送 SOF。要停止 PHY 时钟，应用程序必须将端口时钟停止位置 1，这会使能 PHY 的挂起输入引脚。 此位的读取值反映该端口的当前挂起状态。检测到远程唤醒信号，或者应用程序将此寄存器中的端口复位位或端口恢复位置 1 后，模块可将此位清零；或应用程序将模块中断寄存器中的检测到恢复/ 远程唤醒中断位或检测到断开连接中断位（分别为 GINTSTS 中的 WKUINT 或 DISCINT）置 1，模块也可将此位清零。 0：端口未处于挂起模式 1：端口处于挂起模式

6	PRTRES	RW	0x0	端口恢复（Port resume） 应用程序将此位置 1 以在该端口上驱动恢复信号。模块会持续驱动恢复信号直到应用程序将此位清零。 如模块中断寄存器中的检测到端口恢复/ 远程唤醒中断位（GINTSTS 中的 WKUINT 位）指示，如果模块检测到 USB 远程唤醒序列，则开始驱动恢复信号，而无需应用程序进行干预；如果模块检测到断开连接的情况，则将此位清零。此位的读取值指示当前模块是否 正在驱动恢复信号。 0：不驱动恢复信号 1：驱动恢复信号 当 LPM 已使能且模块处于 L1 状态时，此位的影响如下： 1. 应用程序将此位置 1 以在该端口上驱动恢复信号。 2. 模块继续驱动恢复信号，直至达到 OTG_GLPMCFG 寄存器的 BESLTHRS[3:0] 字段预定 的时间。 3. 如模块中断寄存器中的“检测到端口 L1 恢复/远程 L1 唤醒中断”位（GINTSTS 中的 WKUPINT 位）指示，如果模块检测到 USB 远程唤醒序列，则开始驱动恢复信号，而无需应用程序进行干预，并在恢复结束时，将此位清零。此位可以由模块和应用程序置 1 和清零。即使主机未连接任何设备，此位也可由模块清零。
5	PRTOVRCURRCHNG	RCW1	0x0	端口过流变化（Port overcurrent change） 该寄存器中端口过流激活位（位 4）状态发生变化时，模块将此位置 1。
4	PRTOVRCURRACT	RO	0x0	端口过流激活（Port overcurrent active） 此位指示端口的过流状况。 0：无过流状况 1：有过流状况
3	PRTENCHNG	RCW1	0x0	端口使能/禁止变化（Port enable/disable change） 该寄存器中的端口使能位 2 的状态发生变化时，模块将此位置 1。
2	PRTENA	RCW1	0x0	端口使能（Port enable） 端口执行复位序列后，只能由模块使能，并且可以由过流状况、断开连接状况或应用程序将此位清零来禁止。应用程序无法通过对寄存器执行写操作将此位置 1。只能将此位清零来禁止端口。对此位的操作不会触发应用程序的任何中断。 0：禁止端口 1：使能端口
1	PRTCONNDET	RCW1	0x0	检测到端口连接（Port connect detected） 当检测到设备连接时，模块将此位置 1，以使用模块中断寄存器中的主机端口中断位（GINTSTS 中的 HChIntMsk 位）触发应用程序的中断。应用程序必须将此位置 1 才可清除该中断。
0	PRTCONNSTS	RO	0x0	端口连接状态（Port connect status） 0：端口未连接设备 1：端口已连接设备 0x0

37.15.2.8. 主机通道 x 特性寄存器(OTG_HCCHARx: 500h+(x*20h), x=1~15)

位域	名称	属性	复位值	描述
31	CHENA	RS	0x0	通道使能（Channel enable） 此字段由应用程序软件置 1，并由 OTG 主机硬件清零。 0：禁止通道 1：使能通道
30	CHDIS	RS	0x0	禁止通道（Channel disable） 应用程序将此位置 1 以停止通过通道发送/接收数据，即使通过该通道的传输还未完成，停止操作仍然生效。应用程序必须等待禁止通道的中断以确认通道已经被禁止。
29	ODDFRM	RW	0x0	奇数帧（Odd frame） 此字段由应用程序置位或复位，以分别指示 OTG 主机必须传输奇数帧或偶数帧。此字段只适用于周期性（同步和中断）事务。 0：偶数帧 1：奇数帧
28:22	DEVADDR	RW	0x0	设备地址（Device Address） 此字段用于指定要与该主机通信的特定设备。
21:20	EC	RW	0x0	多重计数（Multicount） 此字段向主机指示该周期性端点每帧必须执行的事务数。该字段不用于非周期性传输。 00：保留。对该字段的操作会产生不明确的结果 01： 1 个事务 10：该端点每帧需要发出 2 个事务 11：该端点每帧需要发出 3 个事务 注： 此字段至少须置为 01。
19:18	EPTYPE	RW	0x0	端点类型（Endpoint type） 指示选择的传输类型。 00：控制 01：同步 10：批量 11：中断
17	LSPDDEV	RW	0x0	低速设备（Low-speed device） 此字段由应用程序置 1，表示此通道正在与一个低速设备进行通信。
16	RSV	-	-	保留
15	EPDIR	RW	0x0	端点方向（Endpoint direction） 指示通信事务的方向是输入还是输出。 0： OUT 1： IN

14:11	EPNUM	RW	0x0	端点编号（Endpoint number） 指示要与该主机通道通信的 USB 设备的端点号。
10:0	MPS	RW	0x0	最大数据包大小（Maximum packet size） 指示与该主机通道通信的设备端点的最大数据包大小。

37.15.2.9. 主机通道 x 分离控制寄存器(OTG_HCSPLTx: 504h+(x*20h), x=1~15)

位域	名称	属性	复位值	描述
31	SPLTENA	RW	0x0	分离使能（Split enable） 应用程序将此位置 1 时，指示允许该通道执行分离通信事务。
30:17	RSV	-	-	保留
16	COMPSPLT	RW	0x0	执行完全分离（Do complete split） 应用程序将此位置 1 时，可请求 OTG 主机执行完全分离通信事务。
15:14	XACTPOS	RW	0x0	事务位置（Transaction position） 此字段用于决定随各 OUT 事务发送全部、第一个、中间还是最后一个有效负载。 11：全部。指此事务的全部数据有效负载（小于或等于 188 字节） 10：起始。指此事务的第一个数据有效负载（大于 188 字节） 00：中间。指此事务的中间有效负载（大于 188 字节） 01：结尾。指此事务的最后一个有效负载（大于 188 字节）
13:7	HUBADDR	RW	0x0	集线器地址（Hub address） 此字段存储事务转发器的集线器设备地址。
6:0	PRTADDR	RW	0x0	端口地址（Port address） 此字段是接收方事务转发器的端口编号。

37.15.2.10. 主机通道 x 中断寄存器(OTG_HCINTx: 508h+(x*20h), x=1~15)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	DATATGLERR	RCW1	0x0	数据同步错误（Data toggle error）
9	FRMOVRUN	RCW1	0x0	帧溢出（Frame overrun）
8	BBLERR	RCW1	0x0	串扰错误（Babble error）

7	XACTERR	RCW1	0x0	通信事务错误（Transaction error） 指示 USB 上发生下列错误之一： - CRC 校验失败 - 超时 - 位填充错误 - 错误的 EOP
6	NYET	RCW1	0x0	USB OTG HS 的收到尚未就绪响应中断（Not yet ready response received interrupt for USB OTG HS）
5	ACK	RCW1	0x0	收到/发出 ACK 响应中断（ACK response received/transmitted interrupt）
4	NAK	RCW1	0x0	收到 NAK 响应中断（NAK response received interrupt）
3	STALL	RCW1	0x0	收到 STALL 响应中断（ STALL response received interrupt）
2	AHBERR	RCW1	0x0	USB OTG HS 的 AHB 错误（AHB error for USB OTG HS） 仅当处于内部 DMA 模式下且 AHB 读/写操作期间发生 AHB 错误时才生成此错误。应用程序可读取相应的 DMA 通道地址寄存器来获取错误地址。
1	CHHLTD	RCW1	0x0	通道停止（Channel halted） 因任意 USB 事务错误或为响应应用程序的禁止请求而导致传输非正常结束。
0	XFERCOMPL	RCW1	0x0	传输完成（Transfer completed） 未出现任何错误，正常完成传输。

37.15.2.11. 主机通道 x 中断屏蔽寄存器(OTG_HCINTMSKx: 50Ch+(x*20h), x=1~15)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	DATATGLERRMSK	RW	0x0	数据翻转错误屏蔽（Data toggle error mask） 0：屏蔽中断 1：解除屏蔽中断
9	FRMOVRUNMSK	RW	0x0	帧溢出屏蔽（Frame overrun mask） 0：屏蔽中断 1：解除屏蔽中断
8	BBLERRMSK	RW	0x0	串扰错误屏蔽（Babble error mask） 0：屏蔽中断 1：解除屏蔽中断
7	XACTERRMSK	RW	0x0	通信事务错误屏蔽（Transaction error mask） 0：屏蔽中断 1：解除屏蔽中断

6	NYETMSK	RW	0x0	响应接收中断屏蔽（response received interrupt mask） 0：屏蔽中断 1：解除屏蔽中断
5	ACKMSK	RW	0x0	接收/发送 ACK 响应中断屏蔽（ACK response received/transmitted interrupt mask） 0：屏蔽中断 1：解除屏蔽中断
4	NAKMSK	RW	0x0	NAK 响应接收中断屏蔽（NAK response received interrupt mask） 0：屏蔽中断 1：解除屏蔽中断
3	STALLMSK	RW	0x0	STALL 响应接收中断屏蔽（STALL response received interrupt mask） 0：屏蔽中断 1：解除屏蔽中断
2	AHBERRMSK	RW	0x0	AHB 错误（AHB error） 0：屏蔽中断 1：解除屏蔽中断
1	CHHLTDMSK	RW	0x0	通道停止中断屏蔽（Channel halted mask） 0：屏蔽中断 1：解除屏蔽中断
0	XFERCOMPLMSK	RW	0x0	传输完成中断屏蔽（Transfer completed mask） 0：屏蔽中断 1：解除屏蔽中断

37.15.2.12. 主机通道 x 传输大小寄存器(OTG_HCTSIZx: 510h+(x*20h), x=1~15)

位域	名称	属性	复位值	描述
31	DOPNG	RW	0x0	DoPng 该位仅用于 OUT 传输。将该字段设置为 1 将指示主机执行 PING 协议。 注意：不要为 IN 传输设置该位。如果为 IN 传输设置了该位，则会禁用通道。 0x0 (NOPING)： 无 ping 协议 0x1 (PING)： Ping 协议

30:29	PID	RW	0x0	数据 PID (Data PID) 应用程序在此字段设置数据通信的初始同步 PID。后续传输的数据 PID 由主机硬件维护。 00： DATA0 01： DATA2 10： DATA1 11： SETUP（控制传输） /MDATA（非控制传输）
28:19	PKTCNT	RW	0x0	数据包计数（Packet count) 应用程序在此字段中设置将要发送（OUT）或接收（IN）的数据包数。 主机每成功发送或接收一个 OUT/IN 数据包便递减一次计数值。此值达到 0 后，将中断应用程序来指示操作正常完成。
18:0	XFERSIZE	RW	0x0	传输大小（Transfer size) 对于 OUT 操作，此字段为传输期间主机发送的数据字节数。 对于 IN 操作，此字段为应用程序保留给传输的缓冲区大小。对于 IN 事务（周期性和非周期性），应用程序会将此字段编程为最大数据包大小的整数倍。

37.15.2.13. 主机通道 xDMA 地址寄存器(OTG_HCDMAx: 514h+(x*20h), x=1~15)

位域	名称	属性	复位值	描述

31:0	DMAADDR	RW	0x0	DMA 地址（DMA address) 此字段存储主机从设备端点获取数据或往设备端点发送数据所用 DMA 传输的存储器的地址。每次进行 AHB 传输，该寄存器都会递增。 In Buffer DMA Mode: [31:0]: DMA 地址（DMAAddr)，该字段保存外部存储器的起始地址，端点数据必须从该地址获取或存储到该地址。每个 AHB 事务都会递增该寄存器。 复位：如果未编程，则为 X，因为寄存器位于 SPRAM 中。 In Scatter-Gather DMA (DescDMA) Mode for Non-Isochronous: [31:9]: DMA 地址（DMAAddr)，起始地址必须为 512 字节对齐。 该字段为 512 字节页面的起始地址。列表中的第一个描述符应位于该地址。第一个描述符可能已就绪，也可能尚未就绪。内核从 CTD 值开始处理列表。 [8:3]: 当前传输描述（CTD) 该值以描述符数量为单位。数值范围为 0 至 63。 0 - 1 个描述符。 63 - 64 个描述符。 该字段表示当前在列表中处理的描述符。该字段由应用程序和内核共同更新。例如，如果应用程序在编程 CTD=5 后启用通道，那么内核将开始处理第六个描述符。该地址通过在 DMAAddr 中添加（8 字节*5=) 40（十进制）的值获得。 复位值： 6'h0 [2:0]：保留 In Scatter-Gather DMA (DescDMA) Mode for Isochronous: [31:N]: DMA 地址（DMAAddr) 起始地址必须为 512 字节对齐。 该字段保存 2*(nTD+1) 字节位置的地址，其中等时描述符所在位置的 N 基于 nTD，如下所示： [31:N]：基地址 [N-1:3]：偏移 [2:0]：000 对于 HS ISOC，如果 nTD 是： 7, N=6 15, N=7 31, N=8 63, N=9 127, N=10 255, N=11 对于 FS ISOC，如果 nTD 是： 1, N=4 3, N=5 7, N=6 15, N=7 31, N=8 63, N=9

				[N-1:3]: 当前传输描述（CTD） 等时传输的 CTD 基于当前帧/（微）帧值。需要由应用程序设置为零。 复位值： (N+1:3)'h0 [2:0]: 保留

37.15.3. 从机模式寄存器列表

偏移	名称	复位值	描述
0x800	OTG_DCFG		设备配置寄存器
0x804	OTG_DCTL		设备控制寄存器
0x808	OTG_DSTS		设备状态寄存器
0x810	OTG_DIEPMSK		设备 IN 端点通用中断屏蔽寄存器
0x814	OTG_DOEPMSK		设备 OUT 端点通用中断屏蔽寄存器
0x818	OTG_DAINT		设备全体端点中断寄存器
0x81c	OTG_DAINTMSK		设备全体端点中断屏蔽寄存器
0x828	OTG_DVBUSDIS		设备 VBUS 放电时间寄存器
0x82c	OTG_DVBUSPULSE		设备 VBUS 脉冲时间寄存器
0x830	OTG_DTHRCTL		设备阈值控制寄存器
0x834	OTG_DIEPEMPMSK		设备 IN 端点 FIFO 空中断屏蔽寄存器
0x838	OTG_DEACHINT		设备单个端点中断寄存器
0x83c	OTG_DEACHINTMSK		设备单个端点中断屏蔽寄存器
0x840+(x*0x4)	OTG_DIEPEACHMSKx		设备单个 IN 端点 x 中断寄存器（x=0~15，其中 x=端点编号）
0x880+(x*0x4)	OTG_DOEPEACHMSKx		设备单个 OUT 端点 x 中断寄存器（x=0~15，其中 x=端点编号）
0x900+(x*0x20)	OTG_DIEPCTLx		设备 IN 端点 x 控制寄存器（x=0~15，其中 x=端点编号）
0x908+(x*0x20)	OTG_DIEPINTx		设备 IN 端点 x 中断寄存器（x=0~15，其中 x=端点编号）
0x910	OTG_DIEPTSIZ0		设备 IN 端点 0 传输大小寄存器
0x910+(x*0x20)	OTG_DIEPTSIZx		设备 IN 端点 1 传输大小寄存器（x=1~15，其中 x=端点编号）
0x914+(x*0x20)	OTG_DIEPDMAx		设备 IN 端点 x DMA 地址寄存器（x=0~15，其中 x=端点编号）
0x918+(x*0x20)	OTG_DTXFSTSx		设备 IN 端点发送 FIFO 状态寄存器（x=0~15，其中 x=端点编号）
0xb00	OTG_DOEPCTL0		设备 OUT 端点 0 控制寄存器
0xb00+(x*0x20)	OTG_DOEPCTLx		设备 OUT 端点 x 控制寄存器（x=1~15，其中 x=端点编号）
0xb08 +(x*0x20)	OTG_DOEPINTx		设备 OUT 端点 x 中断寄存器（x=0~15，其中 x=端点编号）
0xb10	OTG_DOEPTSIZ0		设备 OUT 端点 0 传输大小寄存器
0xb10+(x*0x20)	OTG_DOEPTSIZx		设备 OUT 端点 x 传输大小寄存器（x=1~15，其中 x=端点编号）
0xb14+(x*0x20)	OTG_DOEPDMAx		设备 OUT 端点 xDMA 地址寄存器（x=0~15，其中 x=端点编号）

37.15.3.1. 设备配置寄存器(OTG_DCFG: 800h)

位域	名称	属性	复位值	描述
31:26	RESVALID	RW	0x2	恢复验证周期（ResValid） 该字段仅在设置 DCFG.Ena32KHzSusp 时有效。它控制内核从挂起状态恢复时的恢复周期。当设置该位时，内核会计算 ResValid 的时钟周期数，以检测有效的恢复。
25:24	PERSCHINTVL	RW	0x0	周期性调度间隔（Periodic schedule interval） 此字段指定内部 DMA 引擎必须为获取周期性 IN 端点数据分配 的时间。根据周期性端点数量的不同，必须将此值指定为 25%、 50% 或 75% 的（微）帧。 – 只要存在活动的周期性端点，内部 DMA 引擎便会为获取周期性 IN 端点数据分配指定的时间 – 没有任何活动周期性端点时，内部 DMA 引擎将仅用于非周期性端点并忽略 此字段 – 经过一（微）帧中指定的时间后， DMA 切换为获取非周期性端点上的传输 数据 00： 25%（微）帧 01： 50%（微）帧 10： 75%（微）帧 11：保留
23:18	RSV	-	-	保留
17	IPGISOCSUPT	RW	0x1	支持最坏情况下的数据包间隙 ISOC OUT (ipgisocSupt) 该位表示控制器支持 ISOC OUT 令牌后任何令牌的最坏情况 Rx 后 Rx 包间间隙（IPG）（32 位次），符合 UTMI 规范。如果不支持该功能，当 ISOC OUT 令牌之后的任何令牌具有最坏情况 IPG 时，控制器将检测不到被跟随的令牌。在不支持此功能的情况下，控制器的最坏 IPG 取决于 AHB 和 PHY 时钟频率。 0x0（DISABLED）：最坏情况数据包间隙 ISOC OUT 支持已禁用 0x1（ENABLED）：最坏情况数据包间隙 ISOC OUT 支持已启用
16	RSV	-	-	保留
15	ERRATICINTMSK	RW	0x0	不定错误中断屏蔽（Erratic error interrupt mask） 1：发送不定错误时不触发早期挂起中断 0：发生不定错误时生成早期挂起中断
14	XCVRDLY	RW	0x0	收发器延时（Transceiver delay） 使能或禁止器件啁啾期间 ULPI 时序的延迟。 0：禁止延时（使用默认时序） 1：使能默认时序的延时（对于一些 ULPI PHY，必须要使能延时）
13	RSV	-	-	保留

12:11	PERFRINT	RW	0x0	周期性帧间隔（Periodic frame interval） 指示一帧内必须使用周期性帧中断通知应用程序的时间点。此功能可用于确定该帧的所有同步通信是否完成。 00： 80% 帧间隔 01： 85% 帧间隔 10： 90% 帧间隔 11： 95% 帧间隔
10:4	DEVADDR	RW	0x0	设备地址（Device Address） 应用程序必须在执行每个 SetAddress 控制命令后根据命令参数对该字段进行设置。
3	ENA32KHZSUSP	RW	0x0	启用 32 KHz 暂停模式（Ena32KHzSusp） 只有选择 FS PHY 接口时才能设置该位。否则，需要将该位设置为零。如果选择了 FS PHY 接口并设置了该位，则必须将挂起期间的 PHY 时钟从 48 MHz 切换为 32 KHz。 0x0（DISABLED）：未选择 USB 1.1 全速串行收发器 0x0（ENABLED）： 选择 USB 1.1 全速串行收发器接口
2	NZSTSOUTHSHK	RW	0x0	非零长度状态 OUT 握手信号（Non-zero-length status OUT handshake） 在控制传输状态阶段的 OUT 事务期间，当模块收到非零长度数据包后，应用程序可以使用此字段选择要发送的握手信号。 1：收到非零长度状态 OUT 事务时，回复 STALL 握手信号，收到的 OUT 数据包不发送给应用程序。 0：将收到的 OUT 数据包（零长度或非零长度）发送给应用程序，并基于设备端点控制寄存器中端点的 NAK 和 STALL 位回复握手信号。
1:0	DEVSPD	RW	0x0	设备速度（Device speed） 指示应用程序要求模块进行枚举所采用的速度，或应用程序支持的最大速度。但是，实际总线速度只有在完成 chirp 序列后才能确定，同时此速度基于与模块连接的 USB 主机的速度。 0x0 (USBHS20)：高速 USB 2.0 PHY 时钟为 30 MHz 或 60 MHz 0x1 (USBFS20)： 全速 USB 2.0 PHY 时钟为 30 MHz 或 60 MHz 0x2 (USBLS116)： 低速 USB 1.1 收发器时钟为 6 MHz 0x3 (USBFS1148)： 全速 USB 1.1 收发器时钟为 48 MHz

37.15.3.2. 设备控制寄存器(OTG_DCTL: 804h)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留

19	SERVINT	RW	0x0	异步 IN 端点基于服务间隔的调度 该字段用于为 ISOC IN EP 启用基于服务间隔的调度功能。 注意： 该位仅适用于设备模式和使用散射/收集 DMA 模式时。该功能不应与 DCTL.IgnrFrmNum 一起启用。 Scatter/Gather DMA 模式(GAHBCFG.DMAEn=1,DCFG.DescDMA=1): 启用该位后，ISOC IN 描述符结构中的帧号字段将被解释为服务间隔的最后一帧。在 Scatter/Gather DMA 模式下，如果启用该位，控制器将在服务间隔的最后一帧刷新待处理数据包。 0x0（DISABLED）：控制器行为取决于 DCTL.IgnrFrmNum 字段。 0x1（ENABLED）：Scatter/Gather DMA 模式： 控制器可以在服务间隔的任何帧中传输数据包。 Volatile: true
18	DEEPSLEEPBESLREJECT	RW	0x0	深度睡眠 BESL 拒绝（Deep sleep BESL reject) 模块拒绝 BESL 值大于编程的 BESL 阈值的 LPM 请求。将为 BESL 值大于 BESL 阈值的 LPM 令牌发送 NYET 响应。默认情况下，禁用深度睡眠 BESL 拒绝功能。
17	RSV	-	-	保留
16	NAKONBBLE	RW	0x0	Babble 错误时的 NAK（NakOnBble） 自动设置 NAK（NakOnBble）。核心会为接收到 babble 信息的端点自动设置 NAK。 0x0 (DISABLED): 禁用在 Babble 的 Error 0x1 (ENABLED): 使能在 Babble 的 Error

15	IGNRFRMNUM	RW	0x0	异步端点忽略帧号功能（IgnrFrmNum） 该字段也用于控制周期性传输中断（PTI）功能。 注意：当内核工作在阈值模式时，请勿将 IgnrFrmNum 位编程为 1'b1。 从站模式（GAHBCFG.DMAEn=0）： 该位在从站模式下无效，不应编程为 1。Scatter/Gather DMA Mode (GAHBCFG.DMAEn=1,DCFG.DescDMA=1): 注意：启用 Scatter/Gather DMA 模式时，此功能不适用于高速、高带宽传输。启用该位后，每个描述符只能有一个数据包。 0：内核只在预定传输的帧号中传输数据包。 1：核心忽略帧号，在数据包准备就绪时立即发送数据包。 在 Scatter/Gather DMA 模式下，如果启用该位，则在收到已结束帧的 ISOC IN 令牌时不会刷新数据包。 Non-Scatter/Gather DMA 模式，即缓冲 DMA 模式 (GAHBCFG.DMAEn=1, DCFG.DescDMA=0): 禁用 Scatter/Gather DMA 模式时，应用程序使用该字段启用周期性传输中断（PTI）模式。应用程序可对多个（微）帧的周期性端点传输进行编程。 0：禁用周期性传输中断功能，应用程序需要每（微）帧为周期性端点编程传输 1：启用周期性传输中断功能，应用程序可为周期性端点的多个（微）帧传输编程。 在 PTI 模式下，多个（微）帧的传输完成后，应用程序将收到传输完成中断。 0x0 (DISABLED): Scatter/Gather DMA 模式： 内核只在预定传输的帧号中传输数据包。 Non-Scatter/Gather DMA 模式： 禁用周期性传输中断功能。 0x1 (ENABLED): Scatter/Gather DMA 模式： 内核忽略帧号，在数据包准备就绪时立即发送数据包。 Non-Scatter/Gather DMA 模式： 启用周期性传输中断功能。
14:12	RSV	-	-	保留
11	PWRONPRGDONE	RW	0x0	上电编程完成（Power-on programming done） 应用程序使用此位指示寄存器从掉电模式唤醒后已完成编程。
10	CGOUTNAK	WO	0x0	将全局 OUT NAK 清零（Clear global OUT NAK） 对此位执行写操作会将全局 OUT NAK 清零。
9	SGOUTNAK	WO	0x0	将全局 OUT NAK 置 1 (Set global OUT NAK) 对此位执行写操作会将全局 OUT NAK 置 1。 应用程序使用此位在所有 OUT 端点发送 NAK 握手信号。应用程序只有确定模块中断寄存器中的全局 OUT NAK 有效位（GINTSTS 中 GONAKEFF 位）已清零时，才可以将此位置 1。
8	CGNPINNAK	WO	0x0	将全局 IN NAK 清零（Clear global IN NAK） 对此位执行写操作会将全局 IN NAK 清零。

7	SGNPINNAK	WO	0x0	将全局 IN NAK 置 1 (Set global IN NAK) 对此字段执行写操作会将全局非周期性 IN NAK 置 1。应用程序使用此位使所有非周期性 IN 端点发送 NAK 握手信号。 应用程序只有确定模块中断寄存器中的全局 IN NAK 有效位（GINTSTS 中的 GINAKEFF 位）已清零时，才可以将此位置 1。
6:4	TSTCTL	RW	0x0	测试控制 (Test control) 000：禁止测试模式 001： Test_J 模式 010： Test_K 模式 011： Test_SE0_NAK 模式 100： Test_Packet 模式 101： Test_Force_Enable 其它值：保留
3	GOUTNAKSTS	RO	0x0	全局 OUT NAK 状态 (Global OUT NAK status) 0：将根据 FIFO 状态和 NAK 和 STALL 位设置发送握手信号。 1：无论 Rx FIFO 中是否还有空闲空间都不接收数据。除 SETUP 事务之外，对所有收到的数据包回复 NAK 握手信号。所有同步类型的 OUT 数据包都将被丢弃。
2	GNPINNAKSTS	RO	0x0	全局 IN NAK 状态 (Global IN NAK status) 0：根据发送 FIFO 中是否有待发送的数据回复握手信号。 1：使所有非周期性 IN 端点回复 NAK 握手信号，无需考虑发送 FIFO 中是否有待发送的数据。
1	SFTDISCON	RW	0x1	软断开 (Soft disconnect) 应用程序使用该位向 USB OTG 模块发出执行软断开的信号。该位置 1 时，主机不会看到设备已连接，且该设备也不会接收 USB 上的信号。在应用程序将此位清零之前，模块会保持断开状态。 0：正常工作。此位在软断开之后清零，会使主机收到设备已连接的事件。重新连接设备之后， USB 主机会重新启动设备枚举。 1：模块使 USB 主机收到设备断开连接的事件。
0	RMTWKUPSIG	RW	0x0	发送远程唤醒信号 (Remote wakeup signaling) 应用程序将此位置 1 时，模块会启动远程发送信号，以唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出挂起状态。根据 USB 2.0 规范，应用程序必须在将此位置 1 之后的 1 ms 到 15 ms 内将其清零。如果 LPM 已使能并且模块处于 L1（睡眠）状态，则当应用程序将此位置 1 时，模块会启动 L1 远程信号来唤醒 USB 主机。应用程序必须将此位置 1 以使模块退出睡眠状态。按照 LPM 规范中的规定，在应用程序将此位置 1 后经过 50 µs (TL1DevDrvResume)，硬件会自动将此位清零。当前一个 LPM 事务中的 bRemoteWake 为零时（请参见 GLPMCFG 寄存器中的 REMWAKE 位），应用程序不得将此位置 1。

37.15.3.3. 设备状态寄存器(OTG_DSTS: 808h)

位域	名称	属性	复位值	描述

31:24	RSV	-	-	保留
23:22	DEVLNSTS	RO	0x0	设备线状态（Device line status） 指示 USB 数据线的当前逻辑电平。 位 [23]： D+ 的逻辑电平 位 [22]： D- 的逻辑电平
21:8	SOFFN	RO	0x0	接收 SOF 的帧编号（Frame number of the received SOF）
7:4	RSV	-	-	保留
3	ERRTICERR	RO	0x0	不定错误（Erratic error） 模块将该位置 1 以报告任何不定错误。 由于不定错误， OTG_FS/OTG_HS 控制器会进入挂起状态，并且会以 GINTSTS 寄存器的早期挂起位（GINTSTS 中的 ESUSP 位）为应用程序生成一个中断。如果早期挂起中断是由不定错误触发，则应用程序只能执行软断开以恢复通信。
2:1	ENUMSPD	RO	0x1	枚举速度（Enumerated speed） 指示 OTG_HS 控制器通过 chirp 序列检测速度后被枚举成的速度。 01：保留 10：保留 11：全速（ PHY 时钟运行频率为 48 MHz） 其它值：保留
0	SUSPSTS	RO	0x0	挂起状态（Suspend status） 在设备模式下，只要在 USB 上检测到挂起状态，该位就会置 1。当 USB 数据线上的空闲状态保持 3 ms，模块便会进入挂起状态。出现以下情况时，模块会退出挂起状态： – USB 数据线上有活动 – 应用程序对 OTG_DCTL 寄存器的远程唤醒信号位（ OTG_DCTL 中的 RWUSIG 位）执行写操作。

37.15.3.4. 设备 IN 端点通用中断屏蔽寄存器(OTG_DIEPMSK: 80Ch)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	NAKMSK	RW	0x0	NAK 中断屏蔽（NAK interrupt mask） 0：屏蔽中断 1：解除屏蔽中断
12:9	RSV	-	-	保留
8	TXFIFOUNDRNMSK	RW	0x0	USB OTG HS 的 FIFO 下溢屏蔽（FIFO underrun mask for USB OTG HS） 0：屏蔽中断 1：使能中断

7	RSV	-	-	保留
6	INEPNAKEFFMSK	RW	0x0	IN 端点 NAK 有效中断屏蔽（IN endpoint NAK effective mask) 0：屏蔽中断 1：使能中断
5	INTKNEPMISMSK	RW	0x0	EP 不匹配时接收到 IN 令牌中断屏蔽（IN token received with EP mismatch mask) 0：屏蔽中断 1：使能中断
4	INTKNTXFEMPMSK	RW	0x0	Tx FIFO 为空时接收到 IN 令牌中断屏蔽（IN token received when Tx FIFO empty mask) 0：屏蔽中断 1：使能中断
3	TIMEOUTMSK	RW	0x0	超时中断屏蔽（非同步端点） (Timeout condition mask (Non-isochronous endpoints)) 0：屏蔽中断 1：使能中断
2	AHBERRMSK	RW	0x0	AHB 错误屏蔽（AHBErrMsk) 0x0 (MASK): 屏蔽 AHB 错误中断 0x1 (NOMASK): 不屏蔽 AHB 错误中断
1	EPDISBLDMSK	RW	0x0	端点禁止中断屏蔽（Endpoint disabled interrupt mask) 0：屏蔽中断 1：解除屏蔽中断
0	XFERCOMPLMSK	RW	0x0	传输完成中断屏蔽（Transfer completed interrupt mask) 0：屏蔽中断 1：解除屏蔽中断

37.15.3.5. 设备 OUT 端点通用中断屏蔽寄存器(OTG_DOEPMSK: 810h)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	NYETMSK	RW	0x0	NYET 中断屏蔽（NYET interrupt mask) 0：屏蔽中断 1：解除屏蔽中断
13	NAKMSK	RW	0x0	NAK 中断屏蔽（NAKMsk) 0：屏蔽中断 1：解除屏蔽中断

12	BBLEERRMSK	RW	0x0	Babble Error 中断屏蔽（BbleErrMsk） 0：屏蔽中断 1：解除屏蔽中断
11:9	RSV	-	-	保留
8	OUTPKTERRMSK	RW	0x0	OUT 数据包错误屏蔽（OutPktErrMsk） 0：屏蔽中断 1：解除屏蔽中断
7	RSV	-	-	保留
6	BACK2BACKSETUP	RW	0x0	接收到连续的 SETUP 数据包中断屏蔽（Back-to-back SETUP packets received mask）。仅适用于控制 OUT 端点。 0：屏蔽中断 1：解除屏蔽中断
5	STSPHSERCVDMSK	RW	0x0	状态阶段接收屏蔽（StsPhseRcvdMsk） 仅适用于控制 OUT 端点。 0：屏蔽状态阶段接收 1：解除屏蔽状态阶段接收
4	OUTTKNEPDISMSK	RW	0x0	端点禁止时接收到 OUT 令牌中断屏蔽（OUT token received when endpoint disabledmask）。仅适用于控制 OUT 端点。 0：屏蔽中断 1：解除屏蔽中断
3	SETUPMSK	RW	0x0	SETUP 阶段完成中断屏蔽（SETUP phase done mask）。仅适用于控制端点。 0：屏蔽中断 1：解除屏蔽中断
2	AHBERRMSK	RW	0x0	AHB Error (AHBErrMsk) 0：屏蔽 AHB Error 中断 1：解除屏蔽 AHB Error 中断
1	EPDISBLDMSK	RW	0x0	端点禁止中断屏蔽（Endpoint disabled interrupt mask） 0：屏蔽中断· 1：解除屏蔽中断
0	XFERCOMPLMSK	RW	0x0	传输完成中断屏蔽（Transfer completed interrupt mask） 0：屏蔽中断 1：解除屏蔽中断

37.15.3.6. 设备全体端点中断寄存器(OTG_DAINT: 814h)

位域	名称	属性	复位值	描述

31:16	OUTEPINT	RO	0x0	OUT 端点中断 每个 OUT 端点对应一个比特。比特 16 对应 OUT 端点 0，比特 19 对应 OUT 端点 3。 0x0: 无中断 0x1: OUT 端点中断
15:0	INEPINT0	RO	0x0	IN 端点中断 每个 IN 端点对应一个比特。比特 0 对应 IN 端点 0，比特 3 对应 IN 端点 3。 0x0: 无中断 0x1: OUT 端点中断

37.15.3.7. 设备全体端点中断屏蔽寄存器(OTG_DAINTMSK: 818h)

位域	名称	属性	复位值	描述
31	OUTEPMSK15	RW	0x0	OUT 端点 15 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
30	OUTEPMSK14	RW	0x0	OUT 端点 14 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
29	OUTEPMSK13	RW	0x0	OUT 端点 13 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
28	OUTEPMSK12	RW	0x0	OUT 端点 12 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
27	OUTEPMSK11	RW	0x0	OUT 端点 11 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
26	OUTEPMSK10	RW	0x0	OUT 端点 10 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
25	OUTEPMSK9	RW	0x0	OUT 端点 9 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
24	OUTEPMSK8	RW	0x0	OUT 端点 8 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断

23	OUTEPMSK7	RW	0x0	OUT 端点 7 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
22	OUTEPMSK6	RW	0x0	OUT 端点 6 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
21	OUTEPMSK5	RW	0x0	OUT 端点 5 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
20	OUTEPMSK4	RW	0x0	OUT 端点 4 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
19	OUTEPMSK3	RW	0x0	OUT 端点 3 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
18	OUTEPMSK2	RW	0x0	OUT 端点 2 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
17	OUTEPMSK1	RW	0x0	OUT 端点 1 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
16	OUTEPMSK0	RW	0x0	OUT 端点 0 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
15	INEPMSK15	RW	0x0	IN 端点 15 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
14	INEPMSK14	RW	0x0	IN 端点 14 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
13	INEPMSK13	RW	0x0	IN 端点 13 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
12	INEPMSK12	RW	0x0	IN 端点 12 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断

11	INEPMSK11	RW	0x0	IN 端点 11 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
10	INEPMSK10	RW	0x0	IN 端点 10 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
9	INEPMSK9	RW	0x0	IN 端点 9 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
8	INEPMSK8	RW	0x0	IN 端点 8 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
7	INEPMSK7	RW	0x0	IN 端点 7 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
6	INEPMSK6	RW	0x0	IN 端点 6 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
5	INEPMSK5	RW	0x0	IN 端点 5 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
4	INEPMSK4	RW	0x0	IN 端点 4 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
3	INEPMSK3	RW	0x0	IN 端点 3 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
2	INEPMSK2	RW	0x0	IN 端点 2 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
1	INEPMSK1	RW	0x0	IN 端点 1 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断
0	INEPMSK0	RW	0x0	IN 端点 0 中断屏蔽 0x0: 屏蔽中断 0x1: 禁止屏蔽中断

37.15.3.8. 设备 VBUS 放电时间寄存器(OTG_DVBUSDIS: 828h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	DVBUSDIS	RW	0x17d7	设备 VBUS 放电时间 (Device VBUS discharge time) 指定 SRP 期间 VBUS 发出脉冲之后的 VBUS 放电时间。该时间值等于：VBUS 放电时间（ PHY 时钟数） /1024。 该值可基于不同的 VBUS 负载根据需要进行调整。

37.15.3.9. 设备 VBUS 脉冲时间寄存器(OTG_DVBUSPULSE: 82Ch)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DVBUSPULSE	RW	0x5b8	设备 VBUS 脉冲时间 (Device VBUS pulsing time) 指定 SRP 期间的 VBUS 脉冲时间。 VBUS 脉冲时间（PHY 时钟数） /1024

37.15.3.10. 设备阈值控制寄存器(OTG_DTHRCTL: 830h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27	ARBPRKEN	OTG_ARCHITECTURE == 0 ? RO:RW	0x1	仲裁器驻留使能 (Arbiter parking enable) 该位用于控制内部 DMA 仲裁对 IN 端点的驻留。当阈值使能且该位置 1 后， DMA 会对在 USB 上收到令牌的 IN 端点驻留其仲裁。采用这种方式可以避免出现下溢情况。默认情况下使能驻留。
26	RSV	-	-	保留
25:17	RXTHRLEN	OTG_ARCHITECTURE == 0 ? RO:RW	0x8	接收阈值长度 (Receive threshold length) 该字段以双字为单位指定接收阈值的大小。该字段还指定模块在 AHB 上启动传输之前在 USB 上接收的数据量。阈值长度最小值为八个双字。推荐 RXTHRLEN 的值和设定的 AHB 批量传输长度（GAHBCFG 中的 HBSTLEN ）相同。
16	RXTHREN	OTG_ARCHITECTURE == 0 ? RO:RW	0x0	接收阈值使能 (Receive threshold enable) 该位置 1 时，模块会使能接收方向的阈值。
15:13	RSV	-	-	保留

12:11	AHBTHRRATIO	OTG_ARCHITECTURE == 0 ? RO:RW	0x0	AHB 门限比率（AHBThrRatio） 这些比特仅定义发送路径的 AHB 门限和 MAC 门限之间的比率。AHB 门限始终小于或等于 USB 门限，因为这不会增加开销。AHB 和 MAC 门限必须使用 DWORD 对齐。应用程序需要对 TxThrLen 和 AHBThrRatio 进行编程，以使 AHB 门限值 DWORD 对齐。如果 AHB 门限值不是 DWORD 对齐，内核可能无法正常运行。在编程 TxThrLen 和 AHBThrRatio 时，应用程序必须确保最小 AHB 阈值不低于 8DWORDS，以满足 USB 的周转时间要求。 2'b00: AHB 阈值= MAC 阈值 2'b01: AHB 阈值= MAC 阈值/ 2 2'b10: AHB 阈值= MAC 阈值/ 4 2'b11: AHB 阈值= MAC 阈值/ 8
10:2	TXTHRLEN	OTG_ARCHITECTURE == 0 ? RO:RW	0x8	发送阈值长度（Transmit threshold length） 该字段以双字为单位指定发送阈值的大小。该字段指定模块在 USB 上启动传输之前相应端点发送 FIFO 中的数据量（单位为字节）。阈值长度最小值为八个双字。该字段控制同步和非同步 IN 端点阈值。推荐 TXTHRLEN 的值和设定的 AHB 批量传输长度（GAHBCFG 中的 HBSTLEN 位）相同。
1	ISOTHREN	OTG_ARCHITECTURE == 0 ? RO:RW	0x0	ISO IN 端点阈值使能（ISO IN endpoint threshold enable） 该位置 1 时，模块会使能同步 IN 端点的阈值。
0	NONISOTHREN	OTG_ARCHITECTURE == 0 ? RO:RW	0x0	非同步 IN 端点阈值使能（Nonisochronous IN endpoints enable） 该位置 1 时，模块会使能非同步 IN 端点的阈值。

37.15.3.11. 设备 IN 端点 FIFO 空中断屏蔽寄存器(OTG_DIEPEMPMSK: 834h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	INEPTXFEMPMSK	RW	0x0	IN EP Tx FIFO 空中断屏蔽位（IN EP Tx FIFO empty interrupt mask bits） 这些位用作 OTG_DIEPINTx 的屏蔽位。 每个位对应一个 IN 端点的 TXFE 中断： IN 端点 0 对应位 0，而 IN 端点 3 对应位 3。 0：屏蔽中断 1：解除屏蔽中断

37.15.3.12. 设备单个端点中断寄存器(OTG_DEACHINT: 838h)

位域	名称	属性	复位值	描述

31:16	ECHOUTEPINT	RO	0x0	每个 OUT 端点中断位 (EchOutEPInt) 每个 OUT 端点一位： 位 16 用于输出端点 0，位 31 用于输出端点 15 0x1 (OEP0_INT): OUT EP0 中断 0x2 (OEP1_INT): OUT EP1 中断 0x4 (OEP2_INT): OUT EP2 中断 0x8 (OEP3_INT): OUT EP3 中断 0x10 (OEP4_INT): OUT EP4 中断 0x20 (OEP5_INT): OUT EP5 中断 0x40 (OEP6_INT): OUT EP6 中断 0x80 (OEP7_INT): OUT EP7 中断 0x100 (OEP8_INT): OUT EP8 中断 0x200 (OEP9_INT): OUT EP9 中断 0x400 (OEP10_INT): OUT EP10 中断 0x800 (OEP11_INT): OUT EP11 中断 0x1000 (OEP12_INT): OUT EP12 中断 0x2000 (OEP13_INT): OUT EP13 中断 0x4000 (OEP14_INT): OUT EP14 中断 0x8000 (OEP15_INT): OUT EP15 中断
15:0	ECHINEPINT	RO	0x0	每个 IN 端点中断位 (EchInEpInt) 每个 IN 端点一位： 位 0 代表 IN 端点 0，位 15 代表端点 15 0x1 (IEP0_INT): IN EP0 中断 0x2 (IEP1_INT): IN EP1 中断 0x4 (IEP2_INT): IN EP2 中断 0x8 (IEP3_INT): IN EP3 中断 0x10 (IEP4_INT): IN EP4 中断 0x20 (IEP5_INT): IN EP5 中断 0x40 (IEP6_INT): IN EP6 中断 0x80 (IEP7_INT): IN EP7 中断 0x100 (IEP8_INT): IN EP8 中断 0x200 (IEP9_INT): IN EP9 中断 0x400 (IEP10_INT): IN EP10 中断 0x800 (IEP11_INT): IN EP11 中断 0x1000 (IEP12_INT): IN EP12 中断 0x2000 (IEP13_INT): IN EP13 中断 0x4000 (IEP14_INT): IN EP14 中断 0x8000 (IEP15_INT): IN EP15 中断

37.15.3.13. 设备单个端点中断屏蔽寄存器(OTG_DEACHINTMSK: 83Ch)

位域	名称	属性	复位值	描述
31	ECHOUTEPMSK15	RW	0x0	OUT EP15 中断屏蔽位 0x1 ：(OEP15_INT_MASK)
30	ECHOUTEPMSK14	RW	0x0	OUT EP14 中断屏蔽位 0x1 ：(OEP14_INT_MASK)
29	ECHOUTEPMSK13	RW	0x0	OUT EP13 中断屏蔽位 0x1 ：(OEP13_INT_MASK)
28	ECHOUTEPMSK12	RW	0x0	OUT EP12 中断屏蔽位 0x1 ：(OEP12_INT_MASK)
27	ECHOUTEPMSK11	RW	0x0	OUT EP11 中断屏蔽位 0x1 ：(OEP11_INT_MASK)
26	ECHOUTEPMSK10	RW	0x0	OUT EP10 中断屏蔽位 0x1 ：(OEP10_INT_MASK)
25	ECHOUTEPMSK9	RW	0x0	OUT EP9 中断屏蔽位 0x1 ：(OEP9_INT_MASK)
24	ECHOUTEPMSK8	RW	0x0	OUT EP8 中断屏蔽位 0x1 ：(OEP8_INT_MASK)
23	ECHOUTEPMSK7	RW	0x0	OUT EP7 中断屏蔽位 0x1 ：(OEP7_INT_MASK)
22	ECHOUTEPMSK6	RW	0x0	OUT EP6 中断屏蔽位 0x1 ：(OEP6_INT_MASK)
21	ECHOUTEPMSK5	RW	0x0	OUT EP5 中断屏蔽位 0x1 ：(OEP5_INT_MASK)
20	ECHOUTEPMSK4	RW	0x0	OUT EP4 中断屏蔽位 0x1 ：(OEP4_INT_MASK)
19	ECHOUTEPMSK3	RW	0x0	OUT EP3 中断屏蔽位 0x1 ：(OEP3_INT_MASK)
18	ECHOUTEPMSK2	RW	0x0	OUT EP2 中断屏蔽位 0x1 ：(OEP2_INT_MASK)
17	ECHOUTEPMSK1	RW	0x0	OUT EP1 中断屏蔽位 0x1 ：(OEP1_INT_MASK)
16	ECHOUTEPMSK0	RW	0x0	OUT EP0 中断屏蔽位 0x1 ：(OEP0_INT_MASK)

15	ECHINEPMSK15	RW	0x0	IN EP15 中断屏蔽位 0x1 (IEP15_INT_MASK)
14	ECHINEPMSK14	RW	0x0	IN EP14 中断屏蔽位 0x1 (IEP14_INT_MASK)
13	ECHINEPMSK13	RW	0x0	IN EP13 中断屏蔽位 0x1 (IEP13_INT_MASK)
12	ECHINEPMSK12	RW	0x0	IN EP12 中断屏蔽位 0x1 (IEP12_INT_MASK)
11	ECHINEPMSK11	RW	0x0	IN EP11 中断屏蔽位 0x1 (IEP11_INT_MASK)
10	ECHINEPMSK10	RW	0x0	IN EP10 中断屏蔽位 0x1 (IEP10_INT_MASK)
9	ECHINEPMSK9	RW	0x0	IN EP9 中断屏蔽位 0x1 (IEP9_INT_MASK)
8	ECHINEPMSK8	RW	0x0	IN EP8 中断屏蔽位 0x1 (IEP8_INT_MASK)
7	ECHINEPMSK7	RW	0x0	IN EP7 中断屏蔽位 0x1 (IEP7_INT_MASK)
6	ECHINEPMSK6	RW	0x0	IN EP6 中断屏蔽位 0x1 (IEP6_INT_MASK)
5	ECHINEPMSK5	RW	0x0	IN EP5 中断屏蔽位 0x1 (IEP5_INT_MASK)
4	ECHINEPMSK4	RW	0x0	IN EP4 中断屏蔽位 0x1 (IEP4_INT_MASK)
3	ECHINEPMSK3	RW	0x0	IN EP3 中断屏蔽位 0x1 (IEP3_INT_MASK)
2	ECHINEPMSK2	RW	0x0	IN EP2 中断屏蔽位 0x1 (IEP2_INT_MASK)
1	ECHINEPMSK1	RW	0x0	IN EP1 中断屏蔽位 0x1 (IEP1_INT_MASK)
0	ECHINEPMSK0	RW	0x0	IN EP0 中断屏蔽位 0x1 (IEP0_INT_MASK)

37.15.3.14. 设备单个 IN 端点 x 中断寄存器(OTG_DIEPEACHMSKx: 840h+x*04h, x=0...15)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	NAKMSK	RW	0x0	NAK 中断屏蔽（NAKMsk) 0x0 (MASK)：屏蔽中断 0x1 (NOMASK)：不屏蔽中断
12:9	RSV	-	-	保留
8	TXFIFOUNDRNMSK	RW	0x0	Fifo Underrun 中断屏蔽（TxfifoUndrnMsk) 0x0 (MASK)：屏蔽中断 0x1 (NOMASK)：不屏蔽中断
7	RSV	-	-	保留
6	INEPNAKEFFMSK	RW	0x0	IN 端点 NAK 有效屏蔽（INEPNakEffMsk) 0x0 (MASK): 屏蔽 IN 端点 NAK 有效 0x1 (NOMASK): 不屏蔽 IN 端点 NAK 有效
5	INTKNEPMISMSK	RW	0x0	收到的 IN 令牌 EP 不匹配屏蔽（INTknEPMisMsk) 0x0 (MASK): 屏蔽 IN 令牌 EP 不匹配 0x1 (NOMASK)：不屏蔽 IN 令牌 EP 不匹配
4	INTKNTXFEMPMSK	RW	0x0	TxFIFO 为空时屏蔽收到的 IN 令牌 (INTknTXFEmpMsk) 0x0 (MASK)：TxFIFO 空时屏蔽接收到的 IN 令牌 0x1 (NOMASK)：TxFIFO 空时，不接收 IN 令牌掩码
3	TIMEOUTMSK	RW	0x0	屏蔽超时状态（TimeOUTMsk)（非同步端点） 0x0 (MASK): 屏蔽超时状态 0x1 (NOMASK)：不屏蔽超时状态
2	AHBERRMSK	RW	0x0	AHB Error 屏蔽（AHBErrMsk) 0x0 (MASK): 屏蔽 AHB Error 0x1 (NOMASK)：不屏蔽 AHB Error
1	EPDISBLDMSK	RW	0x0	端点禁用中断屏蔽（EPDisbldMsk) 0x0 (MASK)：屏蔽端点禁用中断 0x1 （NOMASK)：不屏蔽端点禁用中断
0	XFERCOMPLMSK	RW	0x0	传输完成中断屏蔽（XferComplMsk) 0x0 (MASK)：屏蔽传输完成中断 0x1 (NOMASK)：不屏蔽传输完成中断

37.15.3.15. 设备单个 OUT 端点 x 中断寄存器(OTG_DOEPEACHMSKx: 880h+x*04h, x=0...15)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	NYETMSK	RW	0x0	NYET 中断屏蔽（NYETMsk） 0x0 (MASK): 屏蔽 NYET 中断 0x1 (NOMASK): 不屏蔽 NYET 中断
13	NAKMSK	RW	0x0	NAK 中断屏蔽（NAKMsk） 0x0 (MASK): 屏蔽 NAK 中断 0x1 (NOMASK): 不屏蔽 NAK 中断
12	BBLEERRMSK	RW	0x0	Babble 错误中断屏蔽（BbleErrMsk） 0x0 (MASK): 屏蔽 Babble 错误中断 0x1 (NOMASK): 不屏蔽 Babble 错误中断
11:9	RSV	-	-	保留
8	OUTPKTERRMSK	RW	0x0	OUT 包错误屏蔽（OutPktErrMsk） 0x0 (MASK): 屏蔽 OUT 包错误 0x1 (NOMASK): 不屏蔽 OUT 包错误
7	RSV	-	-	保留
6	BACK2BACKSETUP	RW	0x0	背对背 SETUP 数据包接收屏蔽（Back2BackSETup） 仅适用于控制 OUT 端点。Values: 0x0 (MASK): 屏蔽收到的背对背 SETUP 数据包 0x1 (NOMASK): 不屏蔽收到的背对背 SETUP 数据包
5	STSPHSRCVDMSK	RW	0x0	状态相位接收屏蔽（StsPhsRcvdMsk） 仅适用于控制 OUT 端点。 0x0 (MASK)：屏蔽状态相位接收 0x1（NOMASK)：不屏蔽状态相位接收
4	OUTTKNEPDISMSK	RW	0x0	端点禁用时收到的 OUT 令牌屏蔽（OUTTknEPdisMsk） 仅适用于控制 OUT 端点。 0x0 (MASK)： 屏蔽端点禁用时收到的 OUT 令牌 0x1（NOMASK)：不屏蔽端点禁用时收到的 OUT 令牌
3	SETUPMSK	RW	0x0	设置相位完成屏蔽（SetUPMsk） 仅适用于控制端点。 0x0 (MASK)：屏蔽设置阶段完成 0x1（NOMASK)：不屏蔽设置阶段完成

2	AHBERRMSK	RW	0x0	AHB Error (AHBErrMsk) 0x0 (MASK): 屏蔽 AHB Error 0x1 (NOMASK): 屏蔽 AHB Error
1	EPDISBLDMSK	RW	0x0	端点禁用中断屏蔽（EPDisbldMsk) 0x0 (MASK)：屏蔽端点禁用中断 0x1（NOMASK)：不屏蔽端点禁用中断
0	XFERCOMPLMSK	RW	0x0	传输完成中断屏蔽（XferComplMsk) 0x0 (MASK)：屏蔽传输完成中断 0x1（NOMASK)：不屏蔽传输完成中断

37.15.3.16. 设备控制 IN 端点 x 控制寄存器(OTG_DIEPCTLx: 900h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31	EPENA	RS	0x0	端点使能（Endpoint Enable） 适用于 IN 和 OUT 端点。 应用程序设置该位以开始在端点上传输数据。 在此端点上设置以下任何中断之前，内核会清除该位： – setup 阶段完成 – 端点禁用 – 传输完成 0x0: 无动作 0x1: 使能端点
30	EPDIS	RS	0x0	端点禁用（Endpoint Disable） 应用程序设置该位以停止在端点上发送/接收数据，甚至在该端点的传输完成之前。 在将端点视为禁用之前，应用程序必须等待端点禁用中断。 内核在设置端点禁用中断之前清除该位。 仅当已为此端点设置了端点启用时，应用程序才必须设置该位。 0x0：无动作 0x1: 禁止端点
29	SETD1PID	WO	0x0	设置奇数帧（SetD1PID ） 仅适用于同步 IN 和 OUT 端点。 写入此字段会将偶数/奇数帧（EONUM）字段设置为奇数帧。 0x0: 禁止设置 DATA1 PID 0x1: 使能设置 DATA1 PID

28	SETD0PID	WO	0x0	SD0PID：设置 DATA0 PID 仅适用于中断/批量 IN 端点。 写入该字段将该寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。 SEVNFRM：设置偶数帧 仅适用于同步 IN 端点。 写入此字段会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。 0x0：禁止设置 DATA0 PID 0x1：使能设置 DATA0 PID
27	SNAK	WO	0x0	设置 NAK（Set NAK） 写入该位会设置端点的 NAK 位。 使用该位，应用程序可以控制端点上 NAK 握手的传输。 内核还可以在传输完成中断时或在端点接收到 SETUP 后为 OUT 端点设置该位。 0x0: 不设置 NAK 0x1: 设置 NAK
26	CNAK	WO	0x0	Clear NAK (CNAK) 将 NAK 清零 (Clear NAK) 对此位进行写操作会将端点的 NAK 位清零。
25:22	TXFNUM	RW	0x0	Tx FIFO 编号 (Tx FIFO number) 这些位用于指定与此端点相关联的 FIFO 编号。必须为每个有效的 IN 端点设置单独的 FIFO 编号。此字段仅针对 IN 端点有效。
21	STALL	RS	0x0	STALL 握手 (STALL handshake) 仅适用于非控制、非同步 IN 端点（访问类型为 rw）。 应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。只有应用程序能够将此位清零，而模块则不能。 仅适用于控制端点（访问类型为 rs） 此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。
20	RSV	-	-	保留
19:18	EPTYPE	RW	0x0	端点类型 (Endpoint type) 以下是这个逻辑端点支持的传输类型。 00：控制 01：同步 10：批量 11：中断

17	NAKSTS	RO	0x0	NAK 状态 (NAK status) 它指示以下结果： 0：模块根据 FIFO 状态回复非 NAK 握手。 1：模块在此端点上回复 NAK 握手。 当应用程序或模块将此位置 1 时： 对于非同步 IN 端点：即使 Tx FIFO 中存在可用数据，模块也会停止通过 IN 端点发送任何数据。 对于同步 IN 端点：即使 Tx FIFO 中存在可用数据，模块也会发送长度为零的数据包。 无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。
16	DPID	RO	0x0	端点数据 PID（Endpoint Data PID） 仅适用于中断/批量 IN 和 OUT 端点。 包含要在此端点上接收或传输的数据包的 PID。 在端点被激活后，应用程序必须对要在该端点接收或发送的第一个数据包的 PID 进行编程。 应用程序使用该寄存器的 SetD1PID 和 SetD0PID 字段来编程 DATA0 或 DATA1 PID。 1'b0: DATA0 1'b1: DATA1 该字段适用于 Scatter/Gather DMA 模式和非 Scatter/Gather DMA 模式。 偶/奇（微）帧（EO_FrNum）在非分散/聚集 DMA 模式下： 仅适用于同步 IN 和 OUT 端点。 指示核心为此端点发送/接收同步数据的（微）帧号。 应用程序必须使用该寄存器中的 SetEvnFr 和 SetOddFr 字段来编程它打算在其中发送/接收此端点的同步数据的偶数/奇数（微）帧号。 1'b0: Even (micro)frame 1'b1: Odd (micro)frame 当 Scatter/Gather DMA 模式使能时，该字段被保留。 发送数据的帧号在发送描述符结构中提供。 在接收描述符结构中更新接收数据的帧。 0x0: DATA0 or Even Frame 0x1: DATA1 or Odd Frame
15	USBACTEP	RW	0x0	USB 活动端点 (USB active endpoint) 指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。
14:11	RSV	-	-	保留
10:0	MPS	RW	0x0	最大数据包大小 (Maximum packet size) 应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。以字节为单位

37.15.3.17. 设备 IN 端点 x 中断寄存器(OTG_DIEPINTx: 908h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留

14	NYETINTRPT	RCW1	0x0	NYET 中断（NYETIntrpt） 当为非等时 OUT 端点传输 NYET 响应时，内核会产生该中断。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): NYET 中断，写 1 清
13	NAKINTRPT	RCW1	0x0	NAK 中断（NAKInterrupt） USB OTG HS 的 NAK 输入（NAK input for USB OTG HS） 当设备发出或收到 NAK 时，模块将生成该中断。 如果是同步 IN 端点，由于 Tx FIFO 中无数据可发而发送长度为零的数据包时也会生成该中断。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
12	BBLEERR	RCW1	0x0	Babble 错误中断（Babble error interrupt） 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
11	PKTDRPSTS	RCW1	0x0	数据包丢弃状态（Packet dropped status） 该位用于向应用程序指示有 ISOC OUT 数据包被丢弃。该位 没有相应的中断屏蔽位，也不会生成中断。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
10	RSV	-	-	保留
9	BNAINTR	RCW1	0x0	缓冲区不可用中断（Buffer not available interrupt） 当所访问的描述符没有准备好供模块处理时（例如主机繁忙或 DMA 已完成），模块将生成该中断。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
8	TXFIFOUNDRN	RCW1	0x0	发送 FIFO 下溢（TxfifoUndrn）(Transmit Fifo Underrun (TxfifoUndrn)) 当模块检测到该端点的发送 FIFO 下溢时，将生成该中断。相关性：该中断仅在使能了阈值时有效。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
7	TXFEMP	RO	0x1	发送 FIFO 为空（Transmit FIFO empty） 当此端点的 Tx FIFO 为半空或全空时，此中断被置位。 Tx FIFO 为半空还是全空状态由 GAHBCFG 寄存器中的 Tx FIFO 空门限位（GAHBCFG 中的 TXFELVL 位）决定。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断

6	INEPNAKEFF	RCW1	0x0	IN 端点 NAK 有效 (IN endpoint NAK effective) 当应用程序通过向 OTG_DIEPCTLx 中的 CNAK 位写入数据来将 IN 端点 NAK 清零时，此位可被清零。 该中断指示模块已对（由应用程序或模块）置 1 的 NAK 采样，结果已生效。该中断指示由应用程序置 1 的 IN 端点 NAK 位已在模块中起作用。 此中断不一定表示 USB 上已发送了一个 NAK 握手信号。 STALL 位优先级高于 NAK 位。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
5	INTKNEPMIS	RCW1		收到 EP 不匹配的 IN 令牌（INTknEPMis） 仅适用于非周期性 IN 端点。 表示非周期性 TxFIFO 顶部的数据不属于接收 IN 令牌的端点。该中断在收到 IN 标记的端点上断言。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
4	INTKNTXFEMP	RCW1	0x0	Tx FIFO 为空时接收到 IN 令牌 (IN token received when Tx FIFO is empty) 仅适用于非周期性 IN 端点。 当和该端点相对应的 Tx FIFO（周期性/非周期性）为空时，接收到 IN 令牌，从而产生中断。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断
3	TIMEOUT	RCW1	0x0	超时条件 (Timeout condition) 仅适用于控制 IN 端点。指示该端点对最近收到的 IN 令牌响应超时。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
2	AHBERR	RCW1		AHB error (AHBErr) 适用于 IN 和 OUT 端点。 只有在内部 DMA 模式下，当 AHB 读/写过程中出现 AHB 错误时，才会产生该错误。应用程序可通过读取相应的端点 DMA 地址寄存器来获取错误地址。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
1	EPDISBLD	RCW1	0x0	端点禁止中断 (Endpoint disabled interrupt) 此位指示该端点已经由应用程序禁止掉。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清
0	XFERCOMPL	RCW1	0x0	传输完成中断 (Transfer completed interrupt) 此字段指示在此端点上设置的传输已经在 USB 和 AHB 上传输完成。 0x0 (INACTIVE): 无中断 0x1 (ACTIVE): 中断，写 1 清

37.15.3.18. 设备 IN 端点 0 传输大小寄存器(OTG_DIEPTSIZ0: 910h)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:19	PKTCNT	RW	0x0	数据包计数（Packet count） 指示端点 0 的一次数据传输包含的数据包个数。 每次从 Tx FIFO 读取数据包（最大大小数据包或短数据包）时，此字段将递减。
18:7	RSV	-	-	保留
6:0	XFERSIZE	RW	0x0	传输大小（Transfer Size） 指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次向 Tx FIFO 写入来自外部存储器的数据包时，模块会使此字段递减。

37.15.3.19. 设备 IN 端点 x 传输大小寄存器(OTG_DIEPTSIZx: 910h+x*20h, x=1...15)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留
30:29	MC	RW		多重计数（Multi count） 对于周期性 IN 端点，此字段指示在 USB 上每帧必须发送的数据包数。模块使用此字段计算 同步 IN 端点的数据 PID。 01： 1 个数据包 10： 2 个数据包 11： 3 个数据包
28:19	PKTCNT	RW	0x0	数据包计数（Packet count） 指示该端点上的一次数据传输包含的数据包个数。 每次从 Tx FIFO 读取数据包（最大大小数据包或短数据包）时，此字段将递减。
18:0	XFERSIZE	RW	0x0	传输大小（Transfer size） 此字段包含当前端点的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次向 Tx FIFO 写入来自外部存储器的数据包时，模块会使此字段递减。

37.15.3.20. 设备 IN 端点 xDMA 地址寄存器(OTG_DIEPDMAx: 914h+x*20h, x=0...15)

位域	名称	属性	复位值	描述

31:0	DMAADDR	RW		DMA 地址（DMA address） 该字段保存外部存储器中的起始地址，必须从该地址获取端点的数据。每次进行 AHB 传输，该寄存器都会递增。

37.15.3.21. 设备 IN 端点 x 发送 FIFO 状态寄存器(OTG_DTXFSTSx: 918h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	INEPTXFSPCAVAIL	RO	0x8000	IN 端点 Tx FIFO 可用空间（IN endpoint Tx FIFO space available） 指示端点 Tx FIFO 中的可用空闲空间大小。 以 32 位字为单位： 0x0：端点 Tx FIFO 已满 0x1： 1 个字可用 0x2： 2 个字可用 0xn： n 个字可用 其它值：保留

37.15.3.22. 设备控制 OUT 端点 0 控制寄存器(OTG_DOEPCTL0: B00h)

位域	名称	属性	复位值	描述
31	EPENA	RS	0x0	端点使能（Endpoint Enable） 当 Scatter/Gather DMA 模式使能时，对于 OUT 端点，该位指示描述符结构和用于接收数据的数据缓冲区已设置。 当 Scatter/Gather DMA 模式被禁用时（例如基于缓冲区指针的 DMA 模式），该位指示应用程序已分配内存以开始从 USB 接收数据。 在此端点上触发以下任一中断之前，模块会将此位清零： – SETUP 阶段完成 – 端点禁止 – 传输完成 注意：在 DMA 模式下，该位必须设置模块才能将 SETUP 数据包传输到内存中，并且不会在 SETUP 数据包的传输完成中断时被清除。 0x0: 无动作 0x1: 使能端点
30	EPDIS	RO	0x0	端点禁止（Endpoint Disable） 应用程序无法禁用控制 OUT 端点 0。 0x0: 无端点禁止
29:28	RSV	-	-	保留

27	SNAK	WO	0x0	将 NAK 置 1 (Set NAK) 对此位进行写操作会将端点的 NAK 位置 1。 通过此位，应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后，模块也可以将 OUT 端点的这个位置 1。 0x0: 无动作 0x1: NAK 置 1
26	CNAK	WO	0x0	将 NAK 清零 (Clear NAK) 对此位进行写操作会将端点的 NAK 位清零。 0x0: 无动作 0x1: NAK 清零
25:22	RSV	-	-	保留
21	STALL	RS	0x0	STALL 握手 (STALL handshake) 此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。 0x0: 无 Stall 0x1: Stall 握手
20	SNP	RW	0x0	RESERVED Values: 0x0 (RESERVED0): Reserved 0 0x1 (RESERVED1): Reserved 1
19:18	EPTYPE	RO	0x0	端点类型 (Endpoint type) 硬件固定为二进制 00，表示端点为控制传输类型。 0x0：端点控制 0
17	NAKSTS	RO	0x0	NAK 状态 (NAK status) 指示以下结果： 0：模块根据 FIFO 状态回复非 NAK 握手。 1：模块在此端点上回复 NAK 握手。 当应用程序或模块将此位置 1 时，即使 Rx FIFO 中存在空间可继续容纳收到的数据包，模块 也会停止接收数据。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。 0x0: 模块正在根据 FIFO 状态发送非 NAK 握手 0x1: 模块正在此端点上传输 NAK 握手
16	RSV	-	-	保留
15	USBACTEP	RO	0x1	USB 活动端点（USB Active Endpoint） 此位总是置 1，指示在所有配置和接口中控制端点 0 始终处于激活状态。 0x1: USB 活动端点 0
14:2	RSV	-	-	保留

1:0	MPS	RO	0x0	最大数据包大小（Maximum packet size） 控制 OUT 端点 0 的最大数据包大小与在控制 IN 端点 0 中进行编程的值相同。 00： 64 字节 01： 32 字节 10： 16 字节 11： 8 字节

37.15.3.23. 设备控制 OUT 端点 x 控制寄存器(OTG_DOEPCTLx: B00h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31	EPENA	RS	0x0	端点使能（Endpoint enable） 适用于 IN 和 OUT 端点。 当 Scatter/Gather DMA 模式使能时： 对于 IN 端点，该位指示描述符结构和数据缓冲区已准备好发送。 对于 OUT 端点，它指示描述符结构和接收数据的数据缓冲区已设置。 当 Scatter/Gather DMA 模式使能时，例如基于缓冲区指针的 DMA 模式： 对于 IN 端点，该位指示数据已准备好在端点上传输。 对于 OUT 端点，该位指示应用程序已分配内存以开始从 USB 接收数据。 在此端点上触发以下任一中断之前，模块会将此位清零： – SETUP 阶段完成 – 端点禁止 – 传输完成 注意：对于 DMA 模式下的控制端点，必须设置该位，控制器才能将 SETUP 数据包传输到存储器。 该位不会在 SETUP 数据包的传输完成中断时被清除。 0x0: 无动作 0x1: 使能端点
30	EPDIS	RS	0x0	端点禁止（Endpoint Disable） 适用于 IN 和 OUT 端点。 应用程序设置该位以停止在端点上发送/接收数据，甚至在该端点的传输完成之前。在将端点视为禁用之前，应用程序必须等待端点禁用中断。 内核在设置 Endpoint Disabled 中断之前清除该位。 仅当已为此端点设置了端点启用时，应用程序才必须设置该位。 0x0: 无动作 0x1: 禁止端点

29	SETD1PID	WO	0x0	设置 DATA1 PID (Set DATA1 PID) 仅适用于中断和批量 IN 和 OUT 端点。 写入该字段会将该寄存器中的端点数据 PID (DPID) 字段设置为 DATA1。 该字段适用于分散 - 聚集 DMA 模式和非分散 - 聚集 DMA 模式。 设置奇数（微）帧（SetOddFr） 仅适用于同步 IN 和 OUT 端点。 写入此字段会将偶数和奇数（微）帧（EO_FrNum）字段设置为奇数（微）帧。 0x0：禁用设置 DATA1 PID 或不强制奇数帧 0x1：将端点数据 PID 设置为 DATA1 或将 EO_FrNum 字段设置为奇数（微）帧
28	SETD0PID	WO	0x0	设置 DATA0 PID (Set DATA0 PID) 仅适用于中断/批量 OUT 端点。 对此字段进行写操作会将此寄存器中的端点数据 PID (DPID) 字段设置为 DATA0。 0x0: 禁用设置 DATA0 PID 或不强制偶数帧 0x1: 将端点数据 PID 设置为 DATA0 或将 EO_FrNum 字段设置为奇数（微）帧 SEVNFRM： 设置偶数帧 (Set even frame) 仅适用于同步 OUT 端点。 对此字段进行写操作会将偶数/奇数帧 (EONUM) 字段设置为偶数帧。
27	SNAK	WO	0x0	将 NAK 置 1 (Set NAK) 对此位进行写操作会将端点的 NAK 位置 1。 通过此位，应用程序可以控制端点上 NAK 握手信号的发送。发生传输完成中断时或端点上接收到 SETUP 后，模块也可以将 OUT 端点的这个位置 1。 0x0 (INACTIVE): NAK 未置位 0x1 (ACTIVE): NAK 置位
26	CNAK	WO	0x0	将 NAK 清零 (Clear NAK) 对此位进行写操作会将端点的 NAK 位清零。
25:22	RSV	-	-	保留
21	STALL	RS	0x0	STALL 握手 (STALL handshake) 仅适用于非控制、非同步 OUT 端点（访问类型为 rw）。 应用程序将此位置 1 使得设备对来自 USB 主机的所有令牌都回复 STALL。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。只有应用程序能够将此位清零，而模块则不能。 仅适用于控制端点（访问类型为 rs）。 此端点接收到 SETUP 令牌时，应用程序只能将此位置 1，而模块会将其清零。如果 NAK 位、全局 IN NAK 或全局 OUT NAK 与此位同时置 1，则 STALL 位优先。无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。 0x0 (INACTIVE): STALL All non-active tokens 0x1 (ACTIVE): STALL All Active Tokens
20	SNP	RW	0x0	RESERVED 0x0 (RESERVED0): Reserved 0 0x1 (RESERVED1): Reserved 1

19:18	EPTYPE	RW	0x0	端点类型（Endpoint type） 以下是这个逻辑端点支持的传输类型。 00：控制 01：同步 10：批量 11：中断
17	NAKSTS	RO	0x0	NAK 状态（NAK status） 指示以下结果： 当应用程序或模块将此位置 1 时： 即使 Rx FIFO 存在空间可容纳传入数据包，模块也会停止在 OUT 端点上接收任何数据。 无论此位如何设置，模块总是通过 ACK 握手响应 SETUP 数据包。 0x0 (NONNAK): Core 根据 FIFO 状态传输非 NAK 握手信号 0x1 (NAK): Core 正在向该端点发送 NAK 握手信号
16	DPID	RO	0x0	EONUM： 偶数/奇数帧（Even/odd frame） 仅适用于同步 IN 和 OUT 端点。 指示模块为此端点发送/接收同步的数据所在的帧的编号。应用程序必须通过此寄存器中的 SEVNFRM 和 SODDFRM 字段对偶数/奇数帧编号进行编程，以便此端点发送/接收同步 数据。 0：偶数帧 1：奇数帧 DPID： 端点数据 PID (Endpoint data PID) 仅适用于中断/批量 OUT 端点。 包含此端点上将要接收或发送的数据包的 PID。端点激活后，应用程序必须对要在此端点上接收或发送的首个数据包的 PID 进行编程。应用程序使用 SD0 PID 寄存器字段对 DATA0 或 DATA1 PID 进行编程。 0： DATA0 1： DATA1 0x0 (INACTIVE): 端点数据 PID 未激活 0x1 (ACTIVE): 端点数据 PID 激活
15	USBACTEP	RW	0x0	USB 活动端点（USB active endpoint） 指示此端点在当前配置和接口中是否激活。检测到 USB 复位后，模块会为所有端点（端点 0 除外）将此位清零。接收到 SetConfiguration 和 SetInterface 命令后，应用程序必须相应地对端点寄存器进行编程并将此位置 1。 0x0 (DISABLED): 无动作 0x1 (ENABLED): USB 端点激活
14:11	RSV	-	-	保留

10:0	MPS	RW	0x0	最大数据包大小（Maximum packet size） 应用程序必须将此字段编程为当前逻辑端点的最大数据包大小。此值以字节为单位。

37.15.3.24. 设备 OUT 端点 x 中断寄存器(OTG_DOEPINTx: B08h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	STUPPKTRCVD	RCW1	0x0	收到设置数据包 仅适用于缓冲 DMA 模式下的控制输出端点该位由控制器设置，表示该缓冲区保存 8 字节的设置数据。每个缓冲区只有一个设置数据包。收到设置数据包后，控制器会关闭缓冲区并禁用相应的端点。应用程序必须重新启用端点，才能接收控制传输的任何 OUT 数据，并重新编程缓冲区起始地址。 注意：由于上述行为，控制器可以接收任意数量的背靠背设置数据包，每个设置数据包使用一个缓冲区。 1'b0: 未收到 Setup 数据包 1'b1: 收到 Setup 数据包
14	NYETINTRPT	RCW1	0x0	NYET 中断（NYETIntrpt） 当为非同步 OUT 端点传输 NYET 响应时，core 会产生该中断。 0x0 (INACTIVE): 无 NYET 中断 0x1 (ACTIVE): NYET 中断
13	NAKINTRPT	RCW1	0x0	NAK 中断（NAKInterrupt） 当设备发出或收到 NAK 时，模块将生成该中断。如果是同步 IN 端点，由于 Tx FIFO 中无数据可发而发送长度为零的数据包时也会生成该中断。 0x0: 无 NAK 中断 0x1: NAK 中断
12	BBLEERR	RCW1	0x0	Babble 错误中断（Babble error interrupt） 当收到端点的 babble 时，模块会生成此中断。 0x0：无 BbleErr 中断 0x1:BbleErr 中断
11	PKTDRPSTS	RCW1	0x0	数据包丢弃状态（Packet dropped status） 该位用于向应用程序指示有 ISOC OUT 数据包被丢弃。该位没有相应的中断屏蔽位，也不会生成中断。 0x0: 无中断 0x1: 包丢弃状态
10	RSV	-	-	保留

9	BNAINTR	RCW1	0x0	缓冲区不可用中断（Buffer not available interrupt） 该位仅在 Scatter/Gather DMA 模式使能时有效。 当访问的描述符尚未准备好供内核处理时，内核会生成此中断，例如主机忙或 DMA 完成。 0x0: 无 BNA 中断 0x1: BNA 中断
8	OUTPKTERR	RCW1	0x0	OUT 包错误（OUT Packet Error） 仅适用于 OUT 端点 此中断仅在使能阈值时有效。 当模块检测到非同步 OUT 包的溢出或 CRC 错误时，此中断被置位。 0x0: 无 OUT 包 Error 0x1: OUT 包 Error
7	RSV	-	-	保留
6	BACK2BACKSETUP	RCW1	0x0	接收到连续的 SETUP 数据包 仅适用于控制 OUT 端点。 此位指示该端点已接收到三个以上的连续 SETUP 数据包。 0x0：没有收到 Back-to-Back SETUP 包 0x1：收到 Back-to-Back SETUP 包
5	STSPHSERCVD	RCW1	0x0	为控制写入接收的状态相位（StsPhseRcvd） 该中断仅对控制输出端点有效。 只有在内核将主机在控制写入传输的数据阶段发送的所有数据传输到系统内存缓冲区后，才会产生该中断。 该中断向应用程序指示主机已从数据阶段切换到控制写传输的状态阶段。应用程序在解码数据阶段后，可使用该中断来 ACK 或 STALL 状态阶段。这仅适用于 Gather DMA 模式. 0x0 (INACTIVE): 未收到控制写入的状态相位 0x1 (ACTIVE): 未收到控制写入的状态相位
4	OUTTKNEPDIS	RCW1	0x0	端点禁止时接收到 OUT 令牌（OUT token received when endpoint disabled) 仅适用于控制 OUT 端点。 指示在尚未使能端点时接收到 OUT 令牌， 从而产生中断。 0x0: 端点禁止时没有收到 OUT 令牌 0x1: 端点禁止时收到 OUT 令牌
3	SETUP	RCW1	0x0	SETUP 阶段完成（SETUP phase done) 仅适用于控制 OUT 端点。 指示控制端点的 SETUP 阶段已完成，当前控制传输中不再接收到连续的 SETUP 数据包。 在此中断上，应用程序可以对接收到的 SETUP 数据包进行解码。 0x0: SETUP 阶段未完成 0x1: SETUP 阶段完成

2	AHBERR	RCW1	0x0	AHB error（AHBErr） 适用于 IN 和 OUT 端点。 只有在内部 DMA 模式下，当 AHB 读/写过程中出现 AHB 错误时，才会产生该错误。应用程序可通过读取相应的端点 DMA 地址寄存器来获取错误地址。0x0 (INACTIVE): 无 AHB Error 中断 0x1 (ACTIVE): AHB Error 中断
1	EPDISBLD	RCW1	0x0	端点禁止中断（Endpoint disabled interrupt） 适用于 IN 和 OUT 端点。 此位指示该端点已经由应用程序禁止掉。 0x0: 无端点禁止中断 0x1: 端点禁止中断
0	XFERCOMPL	RCW1	0x0	传输完成中断（Transfer completed interrupt） 适用于 IN 和 OUT 端点。 当 Scatter/Gather DMA 模式被启用时 对于 IN 端点，该字段指示来自描述符的请求数据从外部系统存储器移动到内部 FIFO。 对于 OUT 端点，该字段表示来自内部 FIFO 的请求数据被移动到外部系统存储器。 只有当相应的端点描述符关闭并且相应描述符的 IOC 位被置位时才会产生此中断。 注意：在 DMA 模式下，必须设置该位，以便内核将 SETUP 数据包传输到内存中。 当 Scatter/Gather DMA 模式被禁用时，该字段表示该端点的 AHB 和 USB 上的编程传输已完成。 0x0: 无传输完成中断 0x1: 传输完成中断

37.15.3.25. 设备 OUT 端点 0 传输大小寄存器(OTG_DOEPTSIZ0: B10h)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留
30:29	SUPCNT	RW	0x0	SETUP 数据包计数（SETUP packet count） 此字段指定端点能连续接收的 SETUP 数据包数量。 01： 1 个数据包 10： 2 个数据包 11： 3 个数据包
28:20	RSV	-	-	保留
19	PKTCNT	RW	0x0	数据包计数（Packet count） 每向 Rx FIFO 写入一个数据包，此字段递减
18:7	RSV	-	-	保留

6:0	XFERSIZE	RW	0x0	指示端点 0 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从 Rx FIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。

37.15.3.26. 设备 OUT 端点 x 传输大小寄存器(OTG_DOEPTSIZx: B10h+x*20h, x=1...15)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留
30:29	RXDPID	RO	0x0	接收到的数据 PID (Received data PID) 仅适用于同步 OUT 端点。 这是此端点收到的上一个数据包的 PID。 00： DATA0 01： DATA2 10： DATA1 11： MDATA SETUP 数据包计数 (SETUP packet count) 此字段指定端点能连续接收的 SETUP 数据包数量。 01： 1 个数据包 10： 2 个数据包 11： 3 个数据包
28:19	PKTCNT	RW	0x0	数据包计数 (Packet count) 每向 Rx FIFO 写入一个数据包，此字段递减。
18:0	XFERSIZE	RW	0x0	传输大小 (Transfer size) 指示端点 x 的一次数据传输包含的数据量，以字节为单位。仅当应用程序传输完这些数据后，模块才会中断该应用程序。传输大小可以设置为端点的最大数据包大小，以在每个数据包结束时中断。 每次从 Rx FIFO 读取数据包并将其写入外部存储器时，模块会使此字段递减。

37.15.3.27. 设备 OUT 端点 x DMA 地址寄存器(OTG_DOEPDMAx: B14h+x*20h, x=0...15)

位域	名称	属性	复位值	描述
31:0	DMAADDR	RW		保存外部存储器的起始地址，用于存储或获取端点数据。 注意：对于控制端点，该字段存储控制 OUT 数据包以及 SETUP 事务数据包。 当连续接收到三个以上的 SETUP 数据包时，内存中的 SETUP 数据包将被覆盖。 该寄存器在每次 AHB 事务时递增。应用程序只能提供一个 DWORD 对齐的地址。

37.15.4. 电源和时钟门控控制寄存器列表

偏移	名称	复位值	描述
0xe00	OTG_PCGCCTL		电源和时钟门控控制寄存器

37.15.4.1. 电源和时钟门控控制寄存器(OTG_PCGCCTL: E00h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	RESETAFTERSUSP	RW	0x0	挂起后复位 适用于部分掉电模式 在部分掉电工作模式下，如果主机需要在挂起后复位，则需要在移除钳位前在主机模式下设置该位。如果未设置该位，则主机会在挂起后发出复位。 该位不适用于设备模式和非部分掉电模式。在休眠模式下，需要在 PCGCCTL.EssRegRestored 设置之前在 RESTORE_POINT 设置该位。在这种情况下，需要将 PCGCCTL.restore_mode 设置为 wait_restore。 0x0 (DISABLED)：在仅主机模式下，主机在挂起后发出恢复指令. 0x1 (ENABLED): 在仅主机模式下，如果主机需要在挂起后发出复位命令，则主机在移除钳位前设置该位
7	L1SUSPENDED	RO	0x0	L1 深度睡眠 在 L1 状态下时，此位表示 PHY 处于深度睡眠状态。 0x0: 非深度睡眠 0x1: 深度睡眠
6	PHYSLEEP	RO	0x0	PHY 处于睡眠状态 (PHY in Sleep) 此位指示 PHY 处于睡眠状态。 0x0: PHY 不处于睡眠状态 0x1: PHY 处于睡眠状态
5	ENBL_L1GATING	RW	0x0	使能睡眠时钟门控 (Enable Sleep clock gating) 此位置 1 时，如果模块无法触发 utmi_l1_suspend_n，则会在睡眠状态下使能模块内部时钟门控。当此位不置 1 时，不会在睡眠状态下使能 PHY 时钟门控。 0x0: PHY 时钟在休眠状态下没有门控 0x1: 模块内部时钟门控在休眠状态下启用
4	RSV	-	-	保留

3	RSTPDWNMODULE	RW	0x0	复位掉电模块（RstPdwnModule） 该位仅在部分掉电模式下有效。 应用程序在关闭电源时设置该位。 在电源接通且 PHY 时钟启动后，应用程序清零该位。 注意：只有将该位设置为 1b0 时，才能对所有内核寄存器进行 RW 设置。 0x0 (ON): 电源已接通 0x1 (OFF): 电源已关闭
2	PWRCLMP	RW	0x0	电源 Clamp (PwrClmp) 该位仅在部分掉电模式下有效 应用程序在关闭电源前设置该位，以钳位开机模块和关机模块之间的信号。 在电源接通前，应清零该位以禁用 Clamp。 0x0 (DISABLED): 清除该位可在接通电源前禁用 Clamp 功能。 0x1 (ENABLED): 仅在部分掉电模式下，设置该位可在关闭电源前 Clamp 开机模块和关机模块之间的信号
1	GATEHCLK	RW	0x0	门控 HCLK (Gate HCLK) 当 USB 通信挂起或会话无效时，应用程序会将此位置 1，以停止对除 AHB 总线从接口、主接口和唤醒逻辑之外的模块提供时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。 0x0: 当 USB 恢复或新会话开始时清除该位 0x1: 当 USB 挂起或会话无效时，设置该位以将 hclk 门控到模块
0	STOPPCLK	RW	0x0	停止 PHY 时钟 (Stop PHY clock) 当 USB 通信挂起、会话无效或设备断开连接时，应用程序将此位置 1 以停止 PHY 时钟。当 USB 恢复通信或新会话启动时，应用程序将此位清零。 0x0: 禁止停止 Pclk 0x1: 使能停止 Pclk

38. 以太网 MAC 控制器（ETH）

38.1. 概述

以太网(ETH)：通过 DMA 控制器进行介质访问控制(MAC)

借助以太网外设，器件可以通过以太网按照 IEEE 802.3-2002 标准发送和接收数据。

以太网提供了可配置、灵活的外设，用以满足客户的各种应用需求。它支持与外部物理层(PHY) 相连的两个工业标准接口：默认情况下使用的介质独立接口 (MII)（在 IEEE 802.3 规范中定义）和简化介质独立接口 (RMII)。它有多种应用领域，例如交换机和网络接口卡。除了 IEEE 802.3 规范中定义的默认接口外，以太网外设还支持多个与 PHY 相连的工业标准接口。它符合以下标准：

- IEEE 802.3-2008，用于以太网 MAC 和介质独立接口 (MII)
- IEEE 1588-2008，用于精密网络时钟同步协议 (PTP)
- AMBA 2.0，用于 AHB 主端口和 AHB 从端口
- RMII 联盟的 RMII 规范第 1.2 版

38.2. 主要特性

以太网外设嵌入了针对直接存储器接口的专用 DMA、介质访问控制器 (MAC) 和支持多种格式的 PHY 接口模块。

38.2.1. MAC 内核特性

- 支持外部 PHY 接口实现 10Mb/s 和 100 Mb/s 数据传输速率
- 通过符合 IEEE 802.3 的 MII 接口与外部快速以太网 PHY 进行通信
- 支持全双工和半双工操作
  - 支持适用于半双工操作的 CSMA/CD 协议
  - 支持适用于全双工操作的 IEEE 802.3x 流量控制
  - 全双工操作时可以将接收的暂停控制帧转发到用户应用程序
  - 半双工操作时提供背压流量控制
  - 全双工操作中如果流量控制输入信号消失，将自动发送零时间片暂停帧
- 报头和帧起始数据 (SFD) 在发送路径中插入、在接收路径中删除
- 可逐帧控制 CRC 和 pad 自动生成
- 接收帧时可自动去除 pad/CRC
- 可编程帧长度，支持高达 16 KB 的巨型帧
- 可编程帧间隔（ 40-96 位时间，以 8 为步长）
- 支持多种灵活的地址过滤模式：
  - 高达四个 48 位完美 (DA) 地址过滤器，对每个字节进行掩码操作
  - 高达三次 48 位 SA 地址比较检查，对每个字节进行掩码操作
  - 64 位 Hash 滤波器（可选），适用于多播和单播 (DA) 地址
  - 可传送所有多播地址帧
  - 支持混合模式，因此可传送所有帧，无需为网络监视进行过滤

- 传送所有传入数据包时（每次过滤时）均附有一份状态报告

- 为发送和接收数据包分别返回 32 位状态
- 支持对接收帧进行 IEEE 802.1Q VLAN 变量检测
- 为应用程序提供单独的发送、接收和控制接口
- 支持通过 RMON/MIB 计数器（RFC2819/RFC2665）进行强制网络统计
- 使用 MDIO 接口配置和管理 PHY 设备
- 检测 LAN 唤醒帧和 AMD Magic Packet™ 帧
- 在接收功能中支持对接收到的由以太网帧封装的 IPv4 和 TCP 数据包进行校验和卸载
- 在增强型接收功能中支持检查 IPv4 头校验和以及在 IPv4 或 IPv6 数据包中封装的 TCP、UDP 或 ICMP 校验和
- 支持以太网帧时间戳（请参见 IEEE 1588-2008）。每个帧的发送或接收状态下给出 64 位时间戳
- 两组 FIFO：一个具有可编程阈值功能的 2 KB 发送 FIFO 和一个具有可配置阈值（默认为 64 个字节）功能的 2 KB 接收 FIFO
- 接收 FIFO 进行多帧存储时，通过在 EOF 传输后向接收 FIFO 插入接收状态矢量，从而使得接收 FIFO 无需存储这些帧的接收状态
- 在存储转发模式下，可以在接收时过滤所有的错误帧，此时不会将这些错误帧转发给应用程序
- 可以转发过小的好帧
- 为接收 FIFO 中丢失或损坏的帧（由于溢出）生成脉冲，借此支持数据统计
- 向 MAC 内核发送数据时支持存储转发机制
- 根据接收 FIFO 填充（阈值可配置）级别自动生成要发送至 MAC 内核的暂停帧控制或背压信号
- 发送时处理冲突帧的自动重新发送
- 丢弃延迟冲突、过度冲突、过度延迟和下溢条件下的帧
- 通过软件控制刷新 Tx FIFO
- 计算 IPv4 头校验和与 TCP、 UDP 或 ICMP 校验和并将其插入在存储转发模式下发送的帧中
- 支持调试时通过 MII 进行内部回送

38.2.2. DMA 特性

- AHB 从接口中支持所有 AHB 突发类型
- 软件可以在 AHB 主接口中选择 AHB 突发类型（固定或不确定突发）
- 可以从 AHB 主端口选择地址对齐突发
- 通过帧定界符优化以数据包为导向的 DMA 传输
- 支持对数据缓冲区进行字节对齐寻址
- 双缓冲区（环）或链表（链接）描述符链接
- 采用描述符架构可以在 CPU 几乎不干预的情况传输大型数据块
- 每个描述符可传输高达 8 KB 的数据
- 报告正常工作和传输错误时的综合状态
- 可为发送和接收 DMA 引擎单独编程突发大小，以充分利用主总线
- 可编程中断选项，适用于不同的工作条件

- 按帧控制发送/接收完成中断
- 接收引擎和发送引擎间采用循环调度仲裁或固定优先级仲裁
- 启动/停止模式
- 当前 Tx/Rx 缓冲区指针作为状态寄存器
- 当前 Tx/Rx 描述符指针作为状态寄存器

38.2.3. PTP 特性

- 接收帧和发送帧时间戳
- 粗略校准法和精细校准法
- 系统时间大于目标时间时触发中断
- 输出秒脉冲（产品复用功能输出）

38.2.4. 总线接口特性

■ AHB 主接口

AHB 主接口的特性如下：

- 通过 AHB 与应用进行接口
- 小端模式
- AHB 主端口上的 32 位数据
- 拆分、重试和错误 AHB 响应
- AHB 1K 边界突发拆分
- 可通过软件选择 AHB 突发类型（固定突发、不确定突发或二者混合突发）

AHB 主接口不会生成以下响应：

- 回卷突发
- 锁定或受保护的传输

■ AHB 从接口

AHB 从接口支持以下特性：

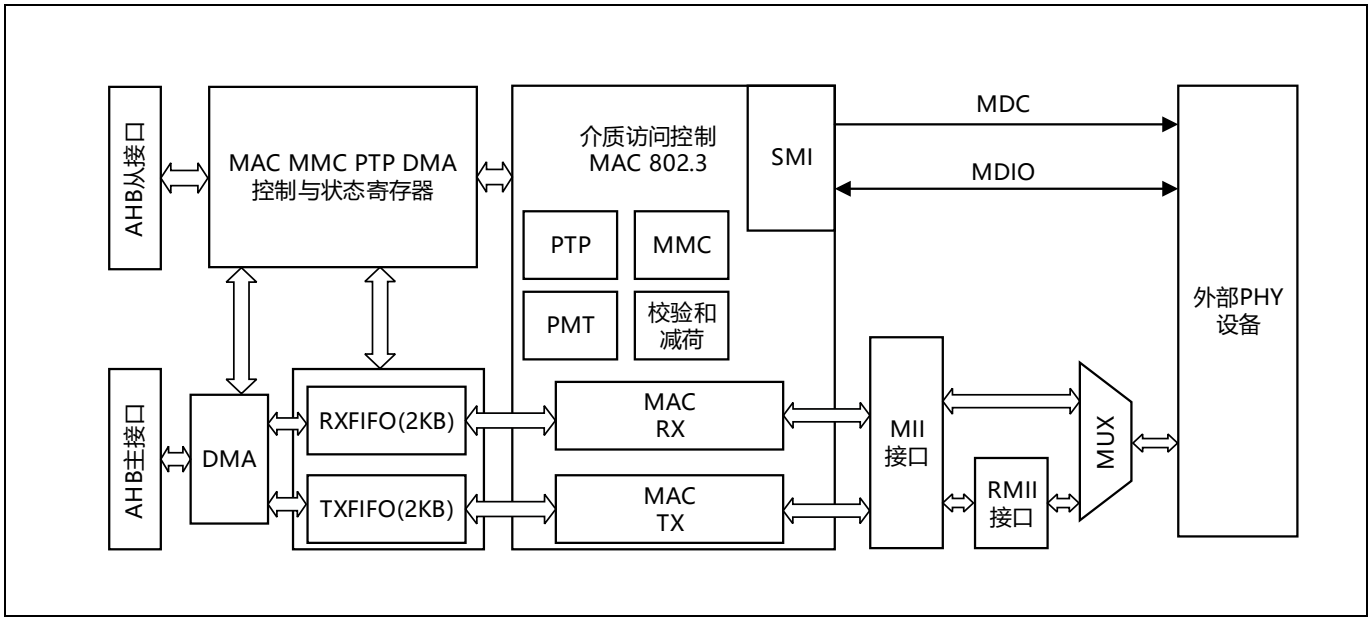
- 通过 AHB 与应用进行接口
- 小端模式
- AHB 从接口（ 32 位），用于 CSR 访问
- 所有 AHB 突发类型

AHB 从接口不会生成以下响应：

- 拆分
- 重试
- 错误

38.3. 功能描述

38.3.1. 结构框图



38.3.2. 以太网引脚

MII 信号	RMII 信号	说明
ETH_MII_TX_CLK		MII 数据发送时钟，25MHz
ETH_MII_TXD0	ETH_RMII_TXD0	MII/RMII 数据发送
ETH_MII_TXD1	ETH_RMII_TXD1	MII/RMII 数据发送
ETH_MII_TXD2		MII 数据发送
ETH_MII_TXD3		MII 数据发送
ETH_MII_TX_ER		MII 发送错误
ETH_MII_TX_EN	ETH_RMII_TX_EN	MII/RMII 发送使能
ETH_MII_RX_CLK	ETH_RMII_REF_CLK	MII 数据接收时钟，25MHz RMII 参考时钟，50MHz
ETH_MII_RXD0	ETH_RMII_RXD0	MII/RMII 数据接收
ETH_MII_RXD1	ETH_RMII_RXD1	MII/RMII 数据接收
ETH_MII_RXD2		MII 数据接收
ETH_MII_RXD3		MII 数据接收
ETH_MII_RX_ER		MII 接收错误
ETH_MII_RX_DV	ETH_RMII_CRS_DV	MII 接收数据有效 RMII 载波侦听/接收数据有效
ETH_MII_CRS		MII 载波侦听
ETH_MII_COL		MII 冲突检测

ETH_MDC	SMI 接口时钟
ETH_MDIO	SMI 接口数据
ETH_PPS_OUT	IEEE1588 对时的脉冲

38.3.3. 以太网功能说明：SMI、MII 和 RMII

以太网外设包括带专用 DMA 控制器的 MAC 802.3（介质访问控制）。它支持介质独立接口(MII) 和简化介质独立接口（RMII），并通过软件在两个接口间进行切换。

DMA 控制器通过 AHB 主从接口与内核和存储器相连。 AHB 主接口用于控制数据传输，而 AHB 从接口则用于访问“控制和状态寄存器” (CSR) 的空间。

在进行数据发送时，首先将数据由系统存储器以 DMA 的方式送至发送 FIFO (Tx FIFO) 进行缓冲，再通过 MAC 内核发送。同样，接收 FIFO (Rx FIFO) 则存储通过线路接收的以太网帧，直到这些帧通过 DMA 传送到系统存储器。

以太网外设还包括用于与外部 PHY 通信的 SMI。通过一组配置寄存器，用户可以为 MAC 控制器和 DMA 控制器选择所需模式和功能。

38.3.3.1. 站管理接口：SMI

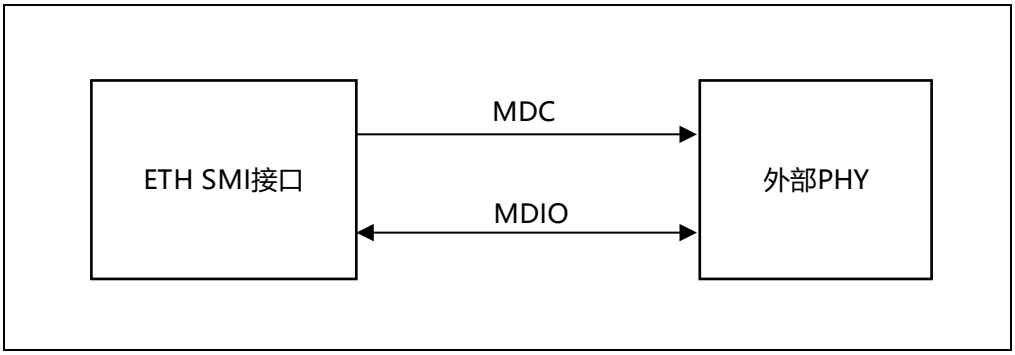
站管理接口 (SMI) 允许应用程序通过 2 线时钟和数据线访问任意 PHY 寄存器。该接口支持访问多达 32 个 PHY。

应用程序可以从 32 个 PHY 中选择一个 PHY，然后从任意 PHY 包含的 32 个寄存器中选择一个寄存器，发送控制数据或接收状态信息。任意给定时间内只能对一个 PHY 中的一个寄存器进行寻址。

MDC 时钟线和 MDIO 数据线在微控制器中均用作复用功能 I/O：

- MDC：周期性时钟，提供以最大频率 2.5 MHz 传输数据时的参考时序。 MDC 的最短高电平时间和最短低电平时间必须均为 160 ns。 MDC 的最小周期必须为 400 ns。在空闲状态下， SMI 管理接口将 MDC 时钟信号驱动为低电平。
- MDIO：数据输入/输出比特流，用于通过 MDC 时钟信号向/从 PHY 设备同步传输状态信息

图 38-1 SMI 接口信号



1. SMI 帧格式

下表中给出了与读操作或写操作有关的帧结构，位传输顺序必须从左到右。

表 38-1 管理帧格式

类型	管理帧格式							
	报头(32 位)	起始	操作	PADDR	RADDR	TA	数据(16 位)	空闲

读取	1...1	01	10	ppppp	rrrrr	Z0	ddddddddddddddd	Z
写入	1...1	01	01	ppppp	rrrrr	10	ddddddddddddddd	Z

管理帧包括八个字段：

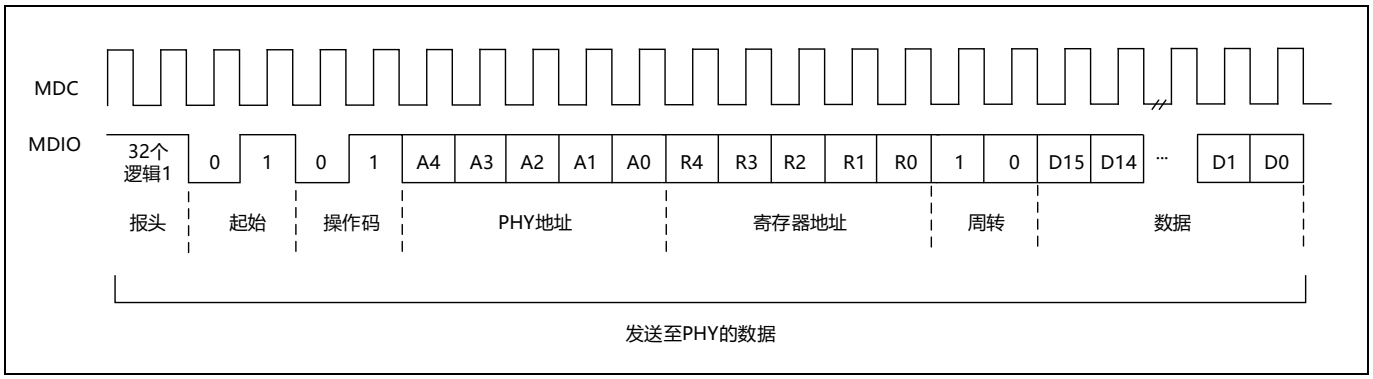
- 报头：每个事务（读取或写入）均可通过报头字段启动，报头字段对应于 MDIO 线上 32 个连续的逻辑“1”位以及 MDC 上的 32 个周期。该字段用于与 PHY 设备建立同步。
- 起始：帧起始由 <01> 模式定义，用于验证线路从默认逻辑“ 1”状态变为逻辑“ 0”状态，然后再从逻辑“0”状态变为逻辑“1”状态。
- 操作：定义正在发生的事务（读取或写入）的类型。
- PADDR： PHY 地址有 5 位，可构成 32 个唯一 PHY 地址。最先发送和接收地址的 MSB 位。
- RADDR：寄存器地址有 5 位，从而可在所选 PHY 设备中对 32 个不同的寄存器进行寻址。最先发送和接收地址的 MSB 位。
- TA：周转字段在 RADDR 和 DATA 字段间定义了一个 2 位模式，以避免在读取事务期间出现竞争现象。读取事务时， MAC 控制器将 TA 的 2 个位驱动为 MDIO 线上的高阻态。 PHY 设备必须将 TA 的第一位驱动为高阻态，将 TA 的第二位驱动为“0”。写入事务时， MAC 控制器针对 TA 字段驱动 <10> 模式。 PHY 设备必须将 TA 的 2 个位驱动为高阻态。
- 数据：数据字段为 16 位。最先发送和接收的位必须为 ETH_MACMIIDR 寄存器的位 15。
- 空闲： MDIO 线驱动为高阻态。三态驱动器必须禁止， PHY 的上拉电阻使线路保持逻辑“1”状态。

2. SMI 写操作

当应用程序将 MII 写入位和繁忙位（在以太网 MAC MII 地址寄存器 (ETH_MACMIIAR) 中）置 1 时，SMI 将通过传输 PHY 地址、 PHY 中的寄存器地址以及写入数据（在以太网 MACMII 数据寄存器 (ETH_MACMIIDR) 中）来触发对 PHY 寄存器进行写操作。事务进行期间，应用程序不应更改 MII 地址寄存器的内容或 MII 数据寄存器。在此期间对 MII 地址寄存器或 MII 数据寄存器执行的写操作将会忽略（繁忙位处于高电平状态），事务将无错完成。写操作完成后， SMI 将通过复位繁忙位进行指示。

下图显示了写操作的帧格式。

图 38-2 MDIO 时序和帧结构-写周期

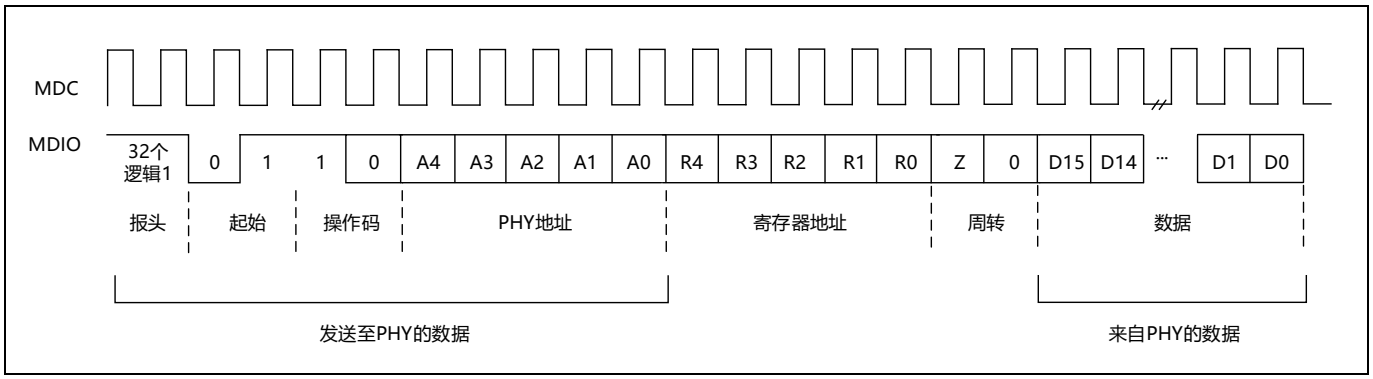


3. SMI 读操作

当用户将以太网 MAC MII 地址寄存器 (ETH_MACMIIAR) 中的 MII 繁忙位置 1、 MII 写入位清零时，SMI 将通过传输 PHY 地址和 PHY 中的寄存器地址在 PHY 寄存器中触发读操作。事务进行期间，应用程序不应更改 MII 地址寄存器的内容或 MII 数据寄存器。在此期间对 MII 地址寄存器或 MII 数据寄存器执行的写操作将会忽略（繁忙位处于高电平状态），事务将无错完成。读操作完成后， SMI 将复位繁忙位，然后用从 PHY 中读取的数据更新 MII 数据寄存器。

下图显示了读操作的帧格式。

图 38-3 MDIO 时序和帧结构-读周期



4. SMI 时钟选择

MAC 启动管理写/读操作。 SMI 时钟是一个分频时钟，其时钟源为应用时钟（ AHB 时钟)。

分频系数取决于 MII 地址寄存器中设置的时钟范围。

下表显示了如何设置时钟范围。

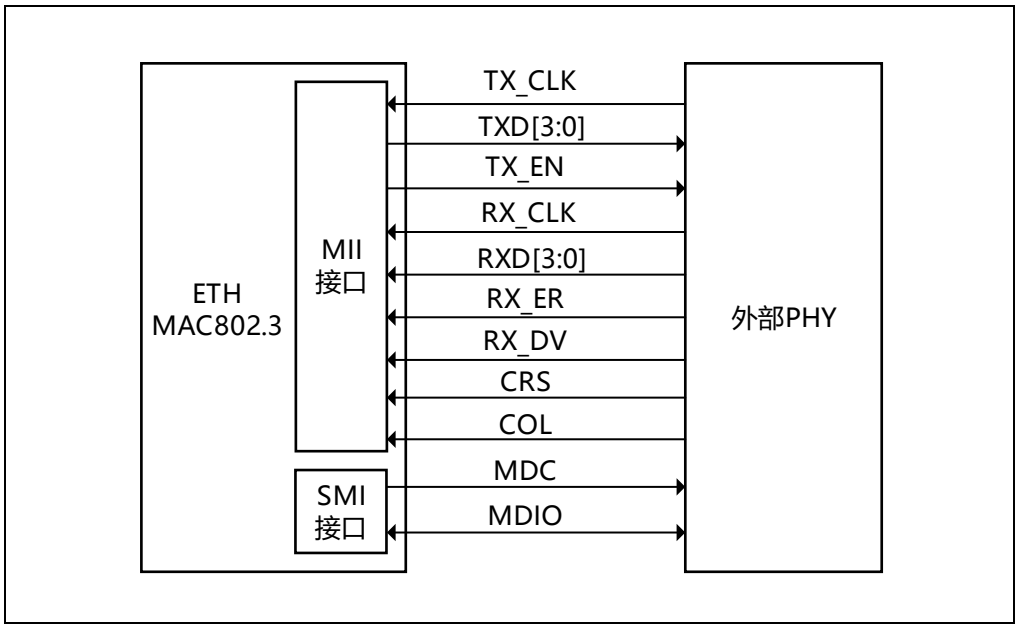
表 38-2 时钟范围

选择	HCLK 时钟	MDC 时钟
000	60-100MHz	AHB 时钟/42
001	100-150MHz	AHB 时钟/62
010	20-35MHz	AHB 时钟/16
011	35-60MHz	AHB 时钟/26
100	150-250MHz	AHB 时钟/102
101	250-300MHz	AHB 时钟/124
110,111	保留	-

38.3.3.2. 介质独立接口：MII

介质独立接口 (MII) 定义了 10 Mbit/s 和 100 Mbit/s 的数据传输速率下 MAC 子层与 PHY 之间的互连。

图 38-4 介质独立接口信号



- MII_TX_CLK：连续时钟信号。该信号提供进行 TX 数据传输时的参考时序。标称频率为：速率为 10 Mb/s 时为 2.5 MHz；速率为 100 Mb/s 时为 25 MHz。
- MII_RX_CLK：连续时钟信号。该信号提供进行 RX 数据传输时的参考时序。标称频率为：速率为 10 Mb/s 时为 2.5 MHz；速率为 100 Mb/s 时为 25 MHz。
- MII_TX_EN：发送使能信号。该信号表示 MAC 当前正针对 MII 发送半字节。该信号必须与报头的前半字节进行同步（MII_TX_CLK），并在所有待发送的半字节均发送到 MII 时必须保持同步。
- MII_TXD[3:0]：数据发送信号。该信号是 4 个一组的数据信号，由 MAC 子层同步驱动，在 MII_TX_EN 信号有效时才为有效信号（有效数据）。 MII_TXD[0] 为最低有效位，MII_TXD[3] 为最高有效位。禁止 MII_TX_EN 时，发送数据不会对 PHY 产生任何影响。
- MII_CRS：载波侦听信号。当发送或接收介质处于非空闲状态时，由 PHY 使能该信号。发送和接收介质均处于空闲状态时，由 PHY 禁止该信号。 PHY 必须确保 MII_CRS 信号在冲突条件下保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。
- MII_COL：冲突检测信号。检测到介质上存在冲突后， PHY 必须立即使能冲突检测信号，并且只要存在冲突条件，冲突检测信号必须保持有效状态。该信号无需与 TX 和 RX 时钟保持同步。在全双工模式下，该信号没意义。
- MII_RXD[3:0]：数据接收信号。该信号是 4 个一组的数据信号，由 PHY 同步驱动，在 MII_RX_DV 信号有效时才为有效信号（有效数据）。 MII_RXD[0] 为最低有效位，MII_RXD[3] 为最高有效位。当 MII_RX_EN 禁止、 MII_RX_ER 使能时，特定的 MII_RXD[3:0] 值用于传输来自 PHY 的特定信息（请参见表 1-4）。
- MII_RX_DV：接收数据有效信号。该信号表示 PHY 当前正针对 MII 接收已恢复并解码的半字节。该信号必须与恢复帧的头半字节进行同步（MII_RX_CLK），并且一直保持同步到恢复帧的最后半字节。该信号必须在最后半字节随后的第一个时钟周期之前禁止。为了正确地接收帧， MII_RX_DV 信号必须在时间范围上涵盖要接收的帧，其开始时间不得迟于 SFD 字段出现的时间。
- MII_RX_ER：接收错误信号。该信号必须保持一个或多个周期（MII_RX_CLK），从而向 MAC 子层指示在帧的某处检测到错误。该错误条件必须通过 MII_RX_DV 验证，如表 1-4 所示。

表 38-3 TX 接口信号编码

MII_TX_EN	MII_TXD[3:0]	说明
0	0000-1111	正常帧间

1	0000-1111	正常数据发送

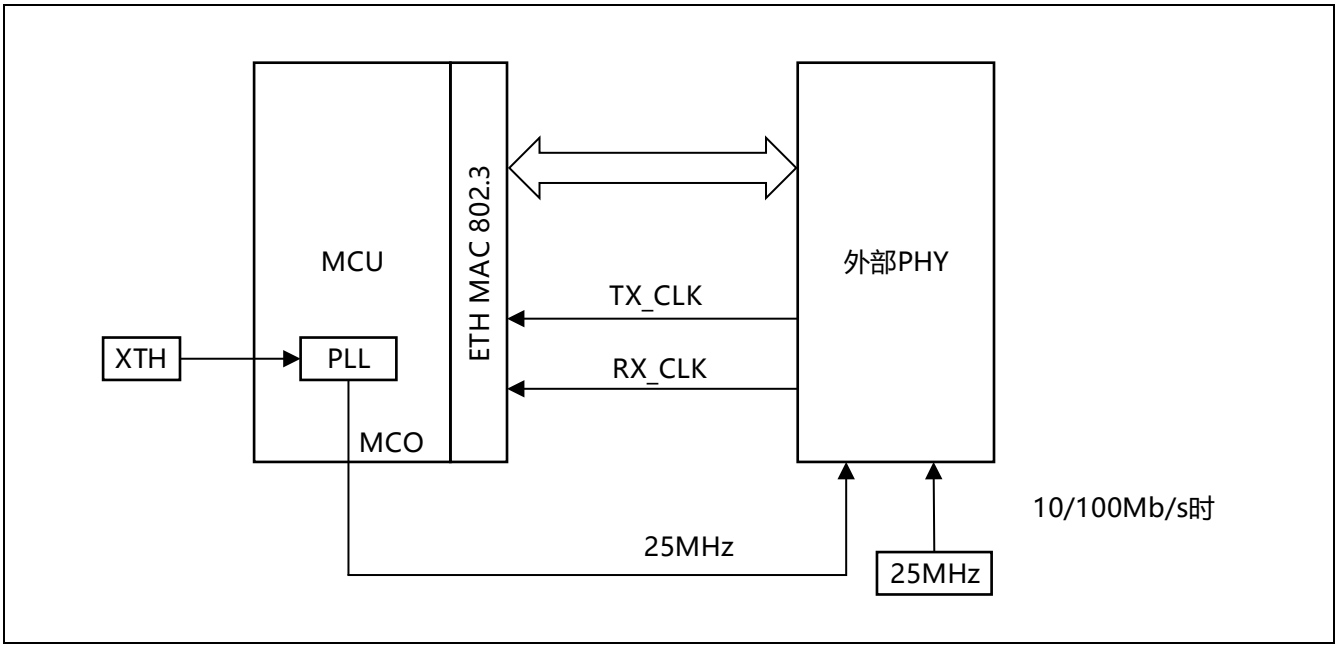
表 38-4 RX 接口信号编码

MII_RX_DV	MII_RX_ERR	MII_RXD[3:0]	说明
0	0	0000-1111	正常帧间
0	1	0000	正常帧间
0	1	0001-1101	保留
0	1	1110	错误载波检测
0	1	1111	保留
1	0	0000-1111	正常数据接收
1	1	0000-1111	数据接收出现错误

1. MII 时钟源

要生成 TX_CLK 和 RX_CLK 时钟信号，必须向外部 PHY 提供 25MHz 时钟，如图 1-5 所示。除了使用外部 25 MHz 石英晶体提供该时钟，还可以通过微控制器的 MCO 引脚输出该信号。这种情况下，必须对 PLL 倍频进行配置，以通过 25 MHz 外部石英晶体在 MCO 引脚上获得所需频率。

图 38-5 MII 时钟源



38.3.3.3. 精简介质独立接口：RMII

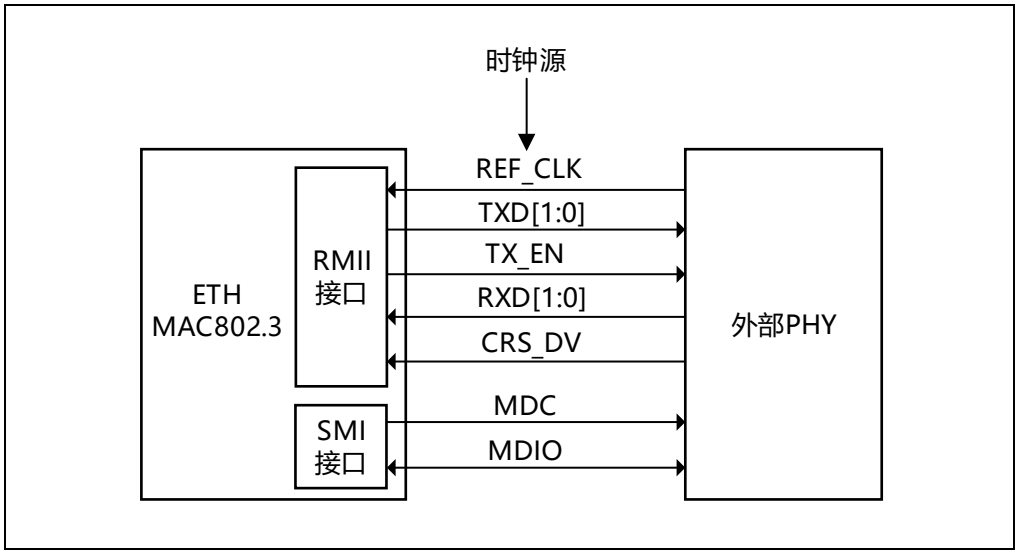
精简介质独立接口（RMII）规范降低了 10/100 Mb/s 下微控制器以太网外设与外部 PHY 间的引脚数。根据 IEEE 802.3u 标准， MII 包括 16 个数据和控制信号的引脚。 RMII 规范将引脚数减少为 7 个（引脚数减少 62.5%）。

RMII 是 MAC 和 PHY 之间的实例化对象。这有助于将 MAC 的 MII 转换为 RMII。RMII 具有以下特性：

- 支持 10 Mb/s 和 100 Mb/s 的运行速率
- 参考时钟必须是 50 MHz

- 相同的参考时钟必须从外部提供给 MAC 和外部以太网 PHY
- 它提供了独立的 2 位宽（双位）的发送和接收数据路径

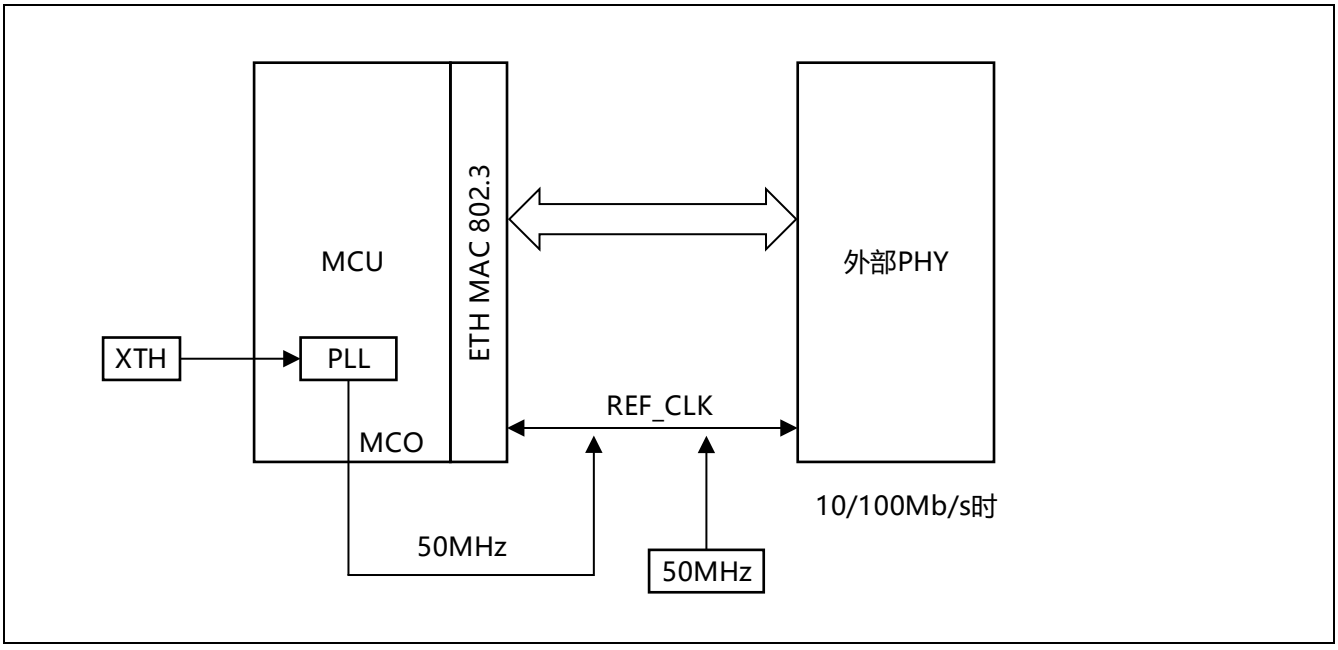
图 38-6 精简介质独立接口信号



1. RMII 时钟源

使用外部 50 MHz 时钟驱动 PHY 或使用嵌入式 PLL 生成 50 MHz 频率信号来驱动 PHY。

图 38-7 RMII 时钟源



38.3.3.4. MII/RMII 选择

使用系统控制寄存器中的位 15 EPIS 选择 MII 或 RMII 模式。以太网控制器处于复位模式或使能时钟前，应用程序必须设置 MII/RMII 模式。

38.3.4. 以太网功能说明：MAC 802.3

适用于局域网（LAN）的 IEEE 802.3 国际标准将 CSMA/CD（带有冲突检测的载波侦听多路访问）用作访问方法。

以太网外设包括一个带介质独立接口（MII）的 MAC 802.3（介质访问控制）控制器和一个专用 DMA 控制器。

MAC 模块对以下系列的系统使用 LAN CSMA/CD 子层：数据速率为 10 Mb/s 和 100 Mb/s 的基带系统和宽带系统。支持半双工和全双工工作模式。冲突检测访问方法仅适用于半双工工作模式。支持 MAC 控制帧子层。

MAC 子层执行以下与数据链路控制步骤相关的功能：

- 数据封装（发送和接收）
  - 组帧（帧边界定界、帧同步）
  - 寻址（处理源地址和目标地址）
  - 错误检测
- 介质访问管理
  - 介质分配（冲突避免）
  - 竞争解决（冲突处理）

MAC 子层主要有两个工作模式：

- 半双工模式：站点使用 CSMA/CD 算法争用物理介质。
- 全双工模式：满足以下条件时，无需解决竞争问题（CSMA/CD 算法不是必需的）便可同时发送和接收数据：
  - 物理介质能够支持同步发送和接收
  - 正好有 2 个站点与 LAN 相连
  - 两个站均配置为全双工工作模式

38.3.4.1. MAC 802.3 帧格式

正如 IEEE 802.3-2002 标准规定， MAC 块使用 MAC 子层和可选 MAC 控制子层

(10/100 Mb/s)。

为使用 CSMA/CD MAC 的数据通信系统指定了两个帧格式：

- 基本 MAC 帧格式
- 标记 MAC 帧格式（扩展了基本 MAC 帧格式）

图 1-9 和图 1-10 介绍了包含以下字段的帧结构（未标记和标记）：

- 报头： 7 字节字段，用于进行同步（ PLS 电路）

十六进制值： 55-55-55-55-55-55-55

位模式： 01010101 01010101 01010101 01010101 01010101 01010101 01010101（从右到左进行位传输）

- 起始帧定界符（SFD)： 1 字节字段，用于指示帧的开始。

十六进制值： D5

位模式： 11010101（从右到左进行位传输）

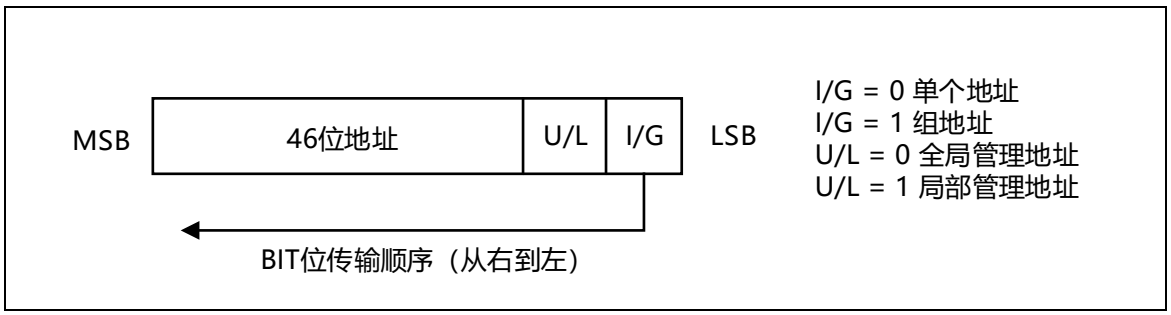
- 目标地址字段和源地址字段： 6 字节字段，指示目标站和源站地址，具体如下（请参见图 478)：
  - 每个地址的长度为 48 位
  - 目标地址字段中的第一个 LSB 位 (I/G) 用于指示单个地址 (I/G = 0) 或组地址 (I/G =1)。一个组地址可以标识“无”、一个或多个，或所有连接至 LAN 的站。在源地址中，第一位保留并复位为 0。

- 第二位（U/L）用于区分局部（U/L = 1）或全局（U/L = 0）管理地址。对于广播地址，该位同样为 1。
- 每个地址字段的各个字节必须最先发送最低有效位。

地址指定基于以下类型：

- 单个地址：与网络中的特殊站关联的物理地址。
- 组地址。与给定网络中一个或多个站关联的多目标地址。有两种多播地址：
  - 多播组地址：与一组逻辑相关站关联的地址。
  - 广播地址：一个特殊的预定义多播地址（目标地址字段中全为“1”），该地址总是表示给定 LAN 上的所有站。

图 38-8 地址字段格式



- QTag 前缀：在源地址字段和 MAC 客户端长度/类型字段中插入的 4 字节字段。该字段是对基本帧（未标记）的扩展，用于获得标记的 MAC 帧。未标记的 MAC 帧不包括该字段。扩展的标记如下：
  - 2 字节常量长度/类型字段值（符合“类型”解析，大于 0x0600），等于 802.1Q 变量协议类型的值（十六进制 0x8100）。该常量字段用于区分标记和未标记的 MAC 帧。
  - 包含变量控制信息的 2 字节字段可以再分为：一个 3 位用户优先级、一个 1 位标准格式指示符（CFI）和一个 12 位 VLAN 标识符。标记 MAC 帧的长度通过 QTag 前缀扩展 4 个字节。
- MAC 客户端长度/类型： 2 字节字段，具有不同含义（互斥），具体取决于其值：
  - 如果该值小于或等于 MaxValidFrame (0d1500)，则该字段表示 802.3 帧的后续数据字段中所包含的 MAC 客户端数据字节的数量（长度解析）。
  - 如果该值大于或等于 MinTypeValue（十进制 0d1536， 0x0600），则该字段表示与以太网帧相关的 MAC 客户端协议的种类（类型解析）。

无论长度/类型字段的解析结果为何，如果数据字段的长度小于协议正确运行所需的最小长度，则将在数据字段之后、 FCS（帧检查序列）字段之前添加一个 PAD 字段。发送和接收长度/类型字段时，高位字节在前。

对于在 maxValidLength 到 minTypeValue 范围内（不包括边界）的长度/类型字段值，未指定 MAC 子层的行为： MAC 子层可能传递、也可能不传递这些值。

- 数据和 PAD 字段： n 字节数据字段。其数据完全透明，这意味着数据字段中可能出现任意顺序的字节值。 PAD 的大小（如果存在）由数据字段的大小决定。数据和 PAD 字段的最大和最小长度为：
  - 最大长度 = 1500 字节
  - 无标记的 MAC 帧的最小长度 = 46 字节
  - 带标记的 MAC 帧的最小长度 = 42 字节

当数据字段的长度小于要求的最小长度时，将添加 PAD 字段以匹配最小长度（带标记的帧为 42 字节，无标记的帧为 46 字节）。

- 帧检查序列：包含循环冗余校验（CRC）值的 4 字节字段。 CRC 计算基于下列字段：源地址、目标地址、QTag 前缀、长度/类型、 LLC 数据和 PAD（即，除报头、 SFD 字段以外的所有字段）。生成的多项式如下：

$$G(x)=x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

帧的 CRC 值按如下方式计算：

- 帧的前 2 位是互补位
- 帧的 n 个位是（n – 1）次多项式 M(x) 的系数。目标地址的第一位对应于 x^{n-1}项，数据字段的最后一位对应于 x0 项
- M(x) 乘以x^{32}再除以 G(x)，得到 31 次的余数 R(x)
- 将 R(x) 的系数视为一个 32 位序列
- 对位序列进行互补，结果为 CRC
- 32 位的 CRC 值放置在帧检查序列中。首先发送x^{32}项，最后发送x^{0}项

图 38-9 MAC 帧格式

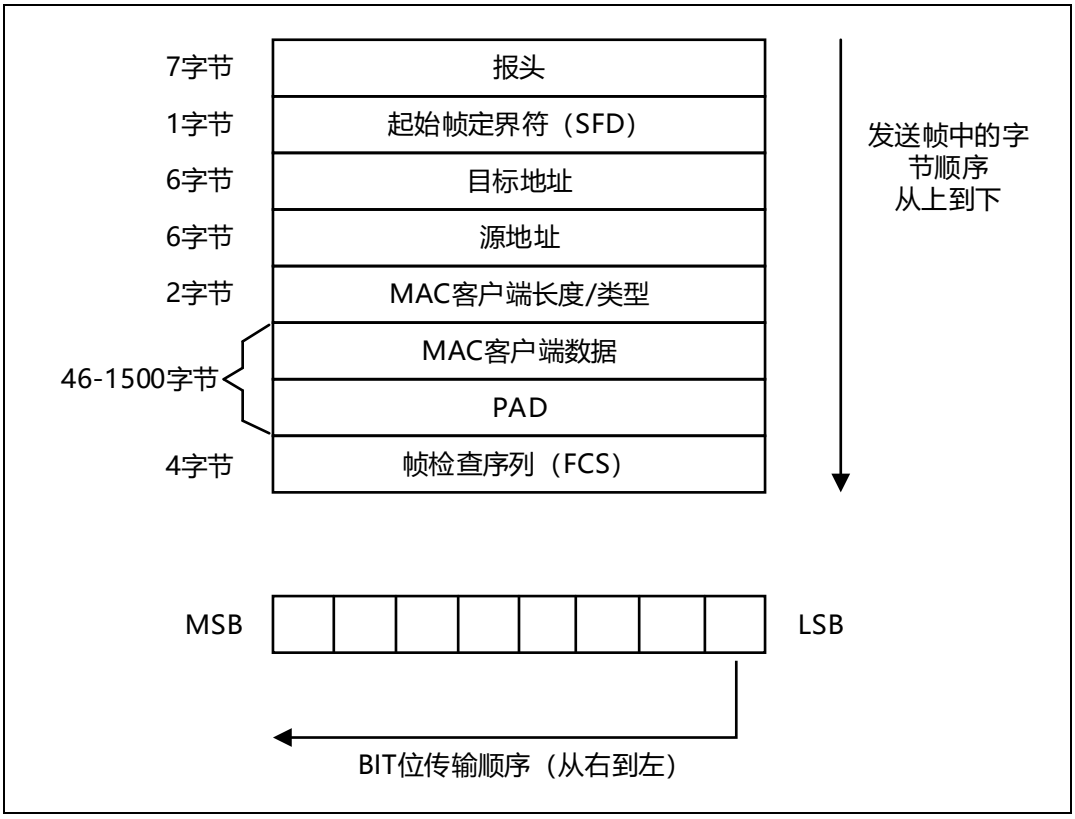
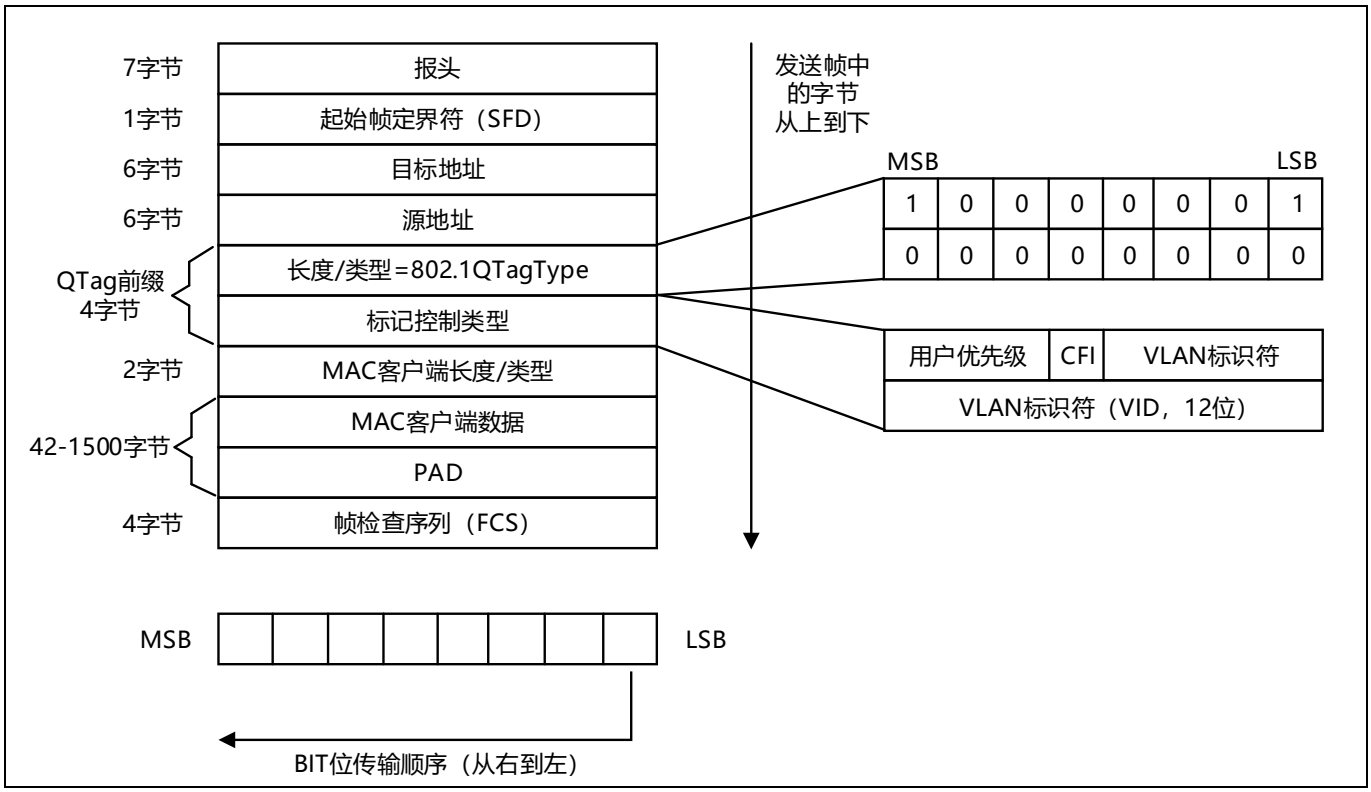


图 38-10 带标记的 MAC 帧格式



发送 MAC 帧的每个字节（除 FSC 字段外）时，低位在前。

通过以下条件之一定义无效 MAC 帧：

- 帧长度与长度 / 类型字段指定的预期值不一致。如果长度 / 类型字段包含类型值，则认为帧长度与此字段一致（没有无效帧）
- 帧长度不是字节的整数倍（额外位）
- 根据传入帧计算出的 CRC 值与包含的 FCS 不匹配。

38.3.4.2. MAC 帧发送

DMA 控制发送路径的所有事务。从系统存储器读取的以太网帧由 DMA 推入 FIFO。然后将帧弹出并传输到 MAC 内核。帧传输结束时，从 MAC 内核获取发送状态并传回 DMA。发送 FIFO 的深度为 2 KB。 FIFO 填充级别将指示给 DMA，以便 DMA 可通过 AHB 接口在所需的系统存储器突发中启动数据获取。来自 AHB 主接口的数据将推入 FIFO。

检测到 SOF 时， MAC 接受数据并开始向 MII 发送。在应用程序启动发送后，向 MII 发送帧数据所需的时间是可变的，具体取决于 IFG 延迟、发送报头/SFD 的时间以及半双工模式的任意回退延迟等延迟因素。EOF 传输到 MAC 内核后，内核将完成正常的发送，然后将发送的状态返回给 DMA。如果在发送过程中发生常规冲突（在半双工模式下）， MAC 内核将使发送状态有效，然后接受并丢弃所有后续数据，直至收到下一 SOF。检测到来自 MAC 的重试请求（在状态中）时，应从 SOF 重新发送同一帧。如果发送期间未连续提供数据， MAC 将发出下溢状态。在帧的正常传输期间，如果 MAC 在未获得前一帧的 EOF 的情况下接收到 SOF，则将忽略该 SOF 并将新的帧视为前一帧的延续。

向 MAC 内核弹出数据有两种操作模式：

- 在阈值模式下，只要 FIFO 中的字节数超过配置的阈值（或在超过阈值前写入帧结束），数据就准备好弹出并转发到 MAC 内核。使用 ETH_DMAOMR 的 TTC 位配置阈值。
- 在存储转发模式下，仅当在 FIFO 中存储完整的帧后，才会向 MAC 内核弹出帧。如果 Tx FIFO 的大小小于要发送的以太网帧，则在 Tx FIFO 接近填满时向 MAC 内核弹出帧。

应用可通过将 FTF 位（ ETH_DMAOMR 寄存器 [20]）置 1 来清空发送 FIFO 的所有内容。此位自行清零，并将 FIFO 指针初始化为默认状态。如果向 MAC 内核传输帧时将 FTF 位置 1，则传输将停止，因为此时 FIFO 被视为空。因此， MAC 发送器将出现下溢事件并且相应的状态字将转发给 DMA。

1. 自动 CRC 和 pad 生成

当从应用程序接收的字节数少于 60（ DA+SA+LT+数据）时，会向发送帧附加零，使数据长度正好为 46 字节，以满足 IEEE 802.3 的最小数据字段要求。可将 MAC 编程为不附加任何填充值。计算帧检查序列 (FCS) 字段的循环冗余校验 (CRC) 并将其附加到正在发送的数据。如果将 MAC 编程为不将 CRC 值附加到以太网帧的末尾，则不发送计算出的 CRC。此规则有一种例外情况，即，将 MAC 编程为向小于 60 字节（ DA+SA+LT+数据）的帧附加填充时， CRC 将附加在填充帧的末尾。

CRC 发生器计算以太网帧的 FCS 字段的 32 位 CRC。编码由下面的多项式定义。

$$G(x)=x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

2. 发送协议

MAC 控制以太网帧的发送操作。它执行下列功能以满足 IEEE 802.3/802.3z 规范。包括：

- 生成报头和 SFD
- 在半双工模式下生成阻塞信号
- 控制 Jabber 超时
- 控制半双工模式下的流量（背压）
- 生成发送帧状态
- 包含符合 IEEE 1588 的时间戳快照逻辑

当请求发送新的帧时， MAC 将发送报头和 SFD，紧接着发送数据。报头定义为 0b10101010 模式的 7 个字节， SFD 定义为 0b10101011 模式的 1 个字节。冲突窗口定义为 1 个时隙（对于 10/100 Mb/s 以太网，为 512 个位时间）。阻塞信号生成仅适用于半双工模式，不适用于全双工模式。

在 MII 模式下，如果在开始传输帧到 CRC 字段结束之间的任何时间发生冲突， MAC 将在 MII 上发送 0x5555 5555 的 32 位阻塞信号，通知所有其它站已发生冲突。如果在报头发送阶段发生冲突， MAC 将完成报头和 SFD 的发送，然后发送阻塞信号。

系统使用一个 jabber 定时器，用于在传输的字节超过 2048 字节（默认值）时切断以太网帧的发送。在半双工模式下， MAC 使用延迟机制进行流量控制（背压）。当应用程序请求停止接收帧时，如果已使能发送流量控制，则只要 MAC 检测到接收帧，就会发送一个 32 字节的 JAM 信号。这会导致冲突并使远程站回退。应用程序通过将 ETH_MACFCR 寄存器中的 BPA 位（位 0）置 1 来请求流量控制。如果应用程序请求发送帧，则即使激活背压功能，也将按调度计划发送。请注意，如果背压功能长时间保持激活（发生的连续冲突事件超过 16 个），则远程站将由于冲突过多而中止发送。如果针对发送帧使能 IEEE 1588 时间戳功能，则将 SFD 置于发送 MII 总线时，此模块会获取系统时间的快照。

3. 发送调度程序

MAC 负责调度 MII 上的帧发送。它可保持两个发送帧之间的帧间隔，并在半双工模式下遵循截断二进制指数回退算法。在满足 IFG 和回退延迟条件后， MAC 使能发送。它可确保两个发送帧之间的空闲期段，即配置的帧间隔（ ETH_MACCR 寄存器中的 IFG 位）。如果要发送的帧在配置的 IFG 时间之前到达，则 MII 会等待来自 MAC 的使能信号，然后再开始发送。只要 MII 的载波信号进入无效状态， MAC 就会启动其 IFG 计数器。在编程的 IFG 值的末尾， MAC 将以全双工模式使能发送。在半双工模式下，当 IFG 配置为 96 个位时间时，MAC 将遵循 IEEE 802.3 规范第 4.2.3.2.1 节中指定的顺从规则。如果在 IFG 间隔的前三分之二时间内（对于所有 IFG 值都为 64 位时间）检测到载波， MAC 将复位其 IFG 计数器。如果在 IFG 间隔的后三分之一时间内检测到载波， MAC 将继续执行 IFG 计数并在 IFG 间隔结束后使能发送器。 MAC 在半双工模式下工作时，实施截断二进制指数回退算法。

■ 单数据包发送操作

发送操作的常规事件序列如下：

1）如果系统有要传输的数据，则 DMA 控制器将通过 AHB 主接口从存储器中获取这些数据并将其转发给 FIFO。它将继续接收数据直到传输帧结束。

2）当超过阈值或 FIFO 接收到完整的数据包时，数据会被弹出并被传输到 MAC 内核。 DMA 继续从 FIFO 传输数据，直到将完整的数据包传输到 MAC 为止。帧传输完成后，来自 MAC 的状态将通知 DMA 控制器。

■ 发送操作——缓冲区中有两个数据包

1）由于 DMA 必须先更新描述符状态才能将其释放给主机，因此一个发送 FIFO 内最多只能有两个帧。仅当 OSF（对第二个帧起作用）位置 1 时， DMA 才会获取第二个帧并将其置于 FIFO 中。如果此位未置 1，则仅当 MAC 完全处理该帧且 DMA 已释放描述符后，DMA 才会从存储器获取下一帧。

2）如果 OSF 位已置 1，则 DMA 将第一个帧传输到 FIFO 后将立即开始获取第二个帧。不会等待状态更新。同时，在发送第一个帧时， FIFO 接收到第二个帧。第一帧的数据传输完成后，来自 MAC 的状态会通知 DMA。如果 DMA 已向 FIFO 发送第二个数据包，则第二次发送必须等待第一个数据包的状态，才能继续下一帧。

4. 冲突期间的重新发送

在半双工模式下，向 MAC 传输帧时，可能在 MAC 线接口上发生冲突事件。 MAC 甚至会在接收到帧结束之前就给出状态来指示重试。然后将使能重新发送并再次将帧从 FIFO 中弹出。当超过 96 个字节弹向 MAC 内核后， FIFO 控制器将释放该空间，使 DMA 可推入更多数据。这意味着超过阈值后或 MAC 内核指示延迟冲突事件时，无法重新发送。

5. 发送 FIFO 刷新操作

MAC 为软件提供了一个控制权，使其可通过操作模式寄存器中的位 20 来清空发送 FIFO。

清空操作是立即操作，即使 Tx FIFO 正在向 MAC 内核传输帧， Tx FIFO 和相应的指针也会清零到初始状态。这将导致 MAC 发送器中生成下溢事件，并且帧发送将中止。此类帧的状态将同时标记下溢和帧清空事件（ TDES0 位 13 和 1）。清空操作期间，没有数据从应用程序 (DMA) 传输到 FIFO。根据清空的帧数（包括局部帧），将相应数量的传输发送状态字传输到应用程序。完全清空的帧的帧清空状态位 (TDES0 13) 将置 1。当应用程序 (DMA) 接受所有已清空的帧的状态字后，清空操作完成。发送 FIFO 清空控制寄存器位随后将清零。此时，将接受来自应用程序 (DMA) 的新帧。清空操作完成后，将丢弃所有为发送提交的数据，除非数据以 SOF 标记开头。

6. 发送状态字

在向 MAC 内核传输以太网帧结束时以及内核完成帧的发送后，发送状态将提供给应用程序。发送状态的详细说明与 TDES0 中的位 [23:0] 相同。如果使能 IEEE 1588 时间戳功能，将返回特定帧的 64 位时间戳以及发送状态。

7. 发送校验和减荷

通信协议（例如 TCP 和 UDP）将实施校验和字段，这有助于确定通过网络发送的数据的完整性。由于以太网最广泛的用途是通过 IP 数据报封装 TCP 和 UDP，因此以太网控制器具有发送校验和减荷功能，该功能支持校验和计算、发送路径中的校验和插入以及接收路径中的错误检测。本节将介绍发送帧的校验和减荷功能的操作。

注意： 通过完整的帧来计算 TCP、 UDP 或 ICMP 的校验和，然后将其插入相应的报头字段。由于此要求，仅当发送 FIFO 配置为存储转发模式（即， ETH_ETH_DMAOMR 寄存器中的 TSF 位置 1）时，才使能此功能。如果内核配置为阈值（直通）模式，则将绕过发送校验和减荷。必须确保发送 FIFO 足够深，使其能存储一个完整的帧，以便将该帧传输到 MAC 内核发送器。如果 FIFO 的深度小于输入以太网帧的太小，则将绕过

有效负载（TCP/UDP/ICMP）校验和插入功能，并且仅修改帧的 IPv4 报头校验和，即使在存储转发模式下也是如此。

发送校验和减荷支持两种类型的校验和计算和插入。可通过将 CIC 位（ TDES1 中的位 28:27）置 1 来控制每个帧的校验和。

有关 IPv4、 TCP、 UDP、 ICMP、 IPv6 和 ICMPv6 数据包头规范的信息，请分别参见 IETF 规范 RFC 791、 RFC 793、 RFC 768、 RFC 792、 RFC 2460 和 RFC 4443。

■ IP 报头校验和

在 IPv4 数据报中，报头字段的完整性由 16 位头校验和字段（ IPv4 数据报的第十一个字节和第十二个字节）指示。当以太网帧的类型字段值为 0x0800 且 IP 数据报的版本字段值为 0x4 时，校验和减荷将检测到 IPv4 数据报。计算期间，将忽略输入帧的校验和字段并将其替换为计算出的值。 IPv6 报头没有校验和字段；因此，校验和减荷不修改 IPv6 报头字段。此 IP 报头校验和计算的结果由发送状态中的 IP 报头错误状态位（位 16）指示。只要以太网类型字段的值和 IP 报头版本字段的值不一致，或当以太网帧没有足够的数据（如 IP 报头长度字段所指示）时，此状态位将置 1。换言之，在以下情况

下发生 IP 报头错误时，此位将置 1：

- 对于 IPv4 数据报：
  - 接收的以太网类型为 0x0800，但 IP 报头的版本字段不等于 0x4
  - IPv4 报头长度字段指示小于 0x5（ 20 字节）的值
  - 总的帧长度小于 IPv4 报头长度字段给定的值
- 对于 IPv6 数据报：

以太网类型为 0x86DD，但 IP 报头版本字段不等于 0x6

帧在 IPv6 报头（ 40 字节）之前结束，或已完全接收到扩展报头（如扩展报头中相应的报头长度字段中给定）。如果以太网类型字段指示 IPv4 有效负载，即使校验和减荷检测到 IP 报头错误，也会插入 IPv4 报头校验和。

■ TCP/UDP/ICMP 校验和

TCP/UDP/ICMP 校验和对 IPv4 或 IPv6 报头（包括扩展报头）进行处理，并确定封装的有效负载是 TCP、UDP 还是 ICMP。

请注意：

- 对于非 TCP、非 UDP 或非 ICMP/ICMPv6 的有效负载，将绕过此校验和且不会对帧进行修改。
- 将绕过分段的 IP 帧（ IPv4 或 IPv6）、带安全功能（例如，验证报头或封装的安全有效负载）的 IP 帧和带路由报头的 IPv6 帧，并且校验和不对这些帧进行处理。

计算 TCP、 UDP 或 ICMP 有效负载的校验和，然后将其插入报头中相应的字段。它可工作在以下两种模式：

- 在第一种模式下， TCP、 UDP 或 ICMPv6 伪报头并未包含在校验和计算中，并假定其存在于输入帧的校验和字段中。校验和字段包含在校验和计算中，然后替换为最终计算出的校验和。
- 在第二种模式下，将忽略校验和字段， TCP、 UDP 或 ICMPv6 伪报头数据包含在校验和计算中，并使用最终计算出的值覆盖校验和字段。

请注意：对于 ICMP-over-IPv4 数据包，由于没有为其定义伪报头，因此在这两种模式下 ICMP 数据包中的校验和字段都必须始终为 0x0000。如果不等于 0x0000，可能向数据包插入不正确的校验和。

此操作的结果由发送状态向量中的有效负载校验和错误状态位（位 12）指示。当检测到下列情况之一时，有效负载校验和错误状态位置 1：

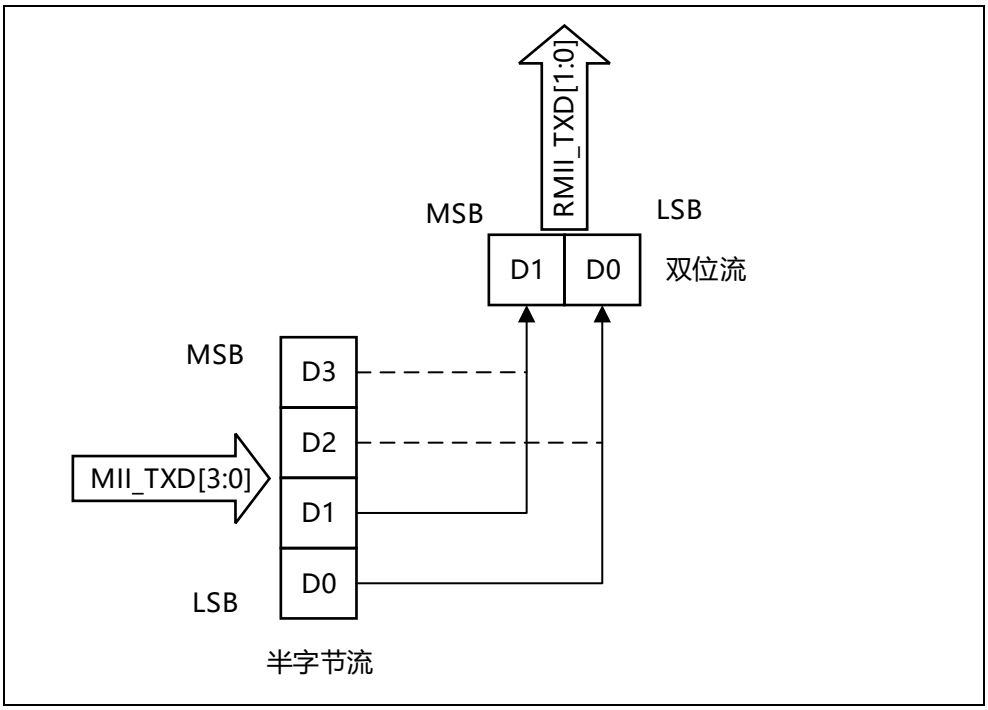
- 在存储转发模式下，帧已转发到 MAC 发送器，但帧结束未写入 FIFO
- 在接收到 IP 报头中的有效负载长度字段指示的字节数前，数据包已结束。

当数据包的长度大于指示的有效负载长度时，将字节作为填充字节忽略，并且不报告错误。检测到第一种类型的错误时，不修改 TCP、 UDP 或 ICMP 报头。对于第二种错误类型，计算的校验和仍将插入相应的报头字段。

8. MII/RMII 发送位序

来自 MII 的每个半字节都在 RMII 上发送，一次发送双位，双位的发送顺序如图 1-11 所示。首先发送位序较低的位（ D1 和 D0），再发送位序较高的位（ D2 和 D3）。

图 38-11 发送位序



9. MII/RMII 发送时序图

图 38-12 无冲突发送

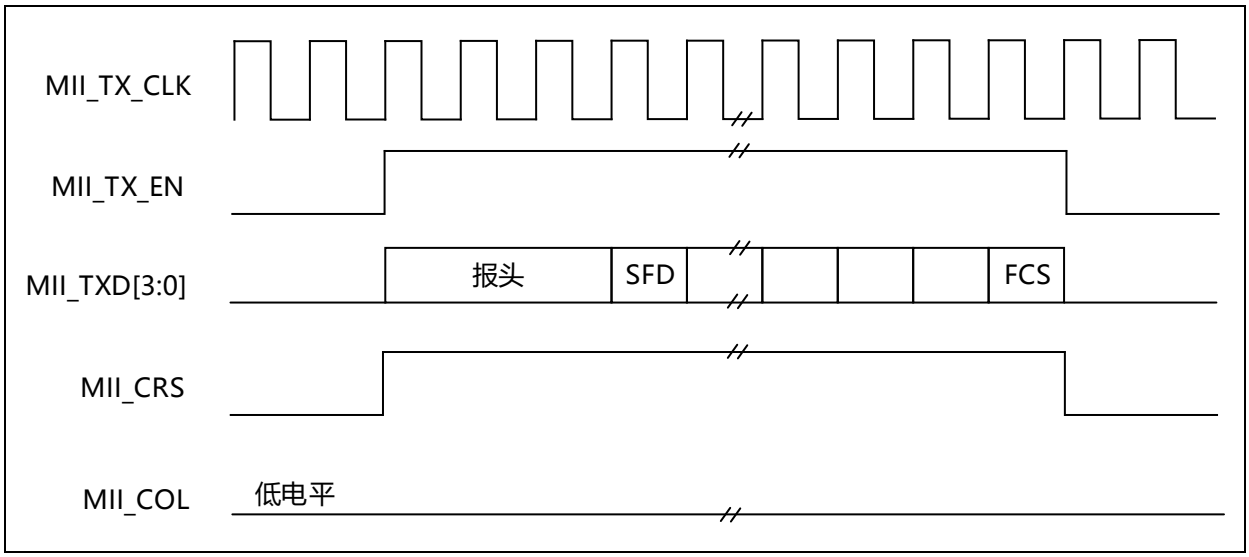


图 38-13 有冲突发送

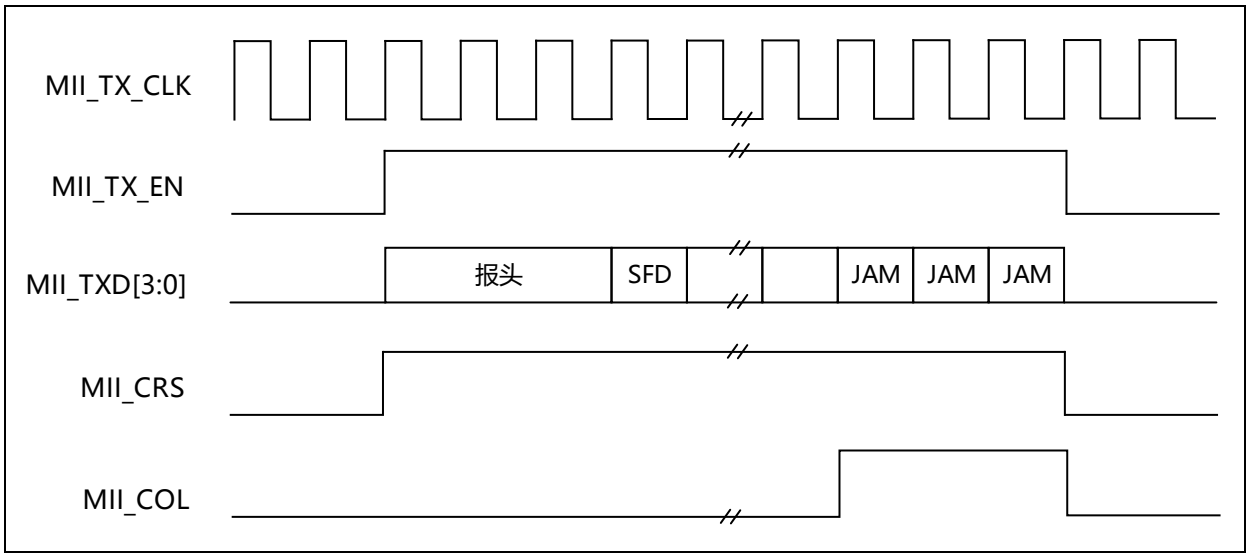
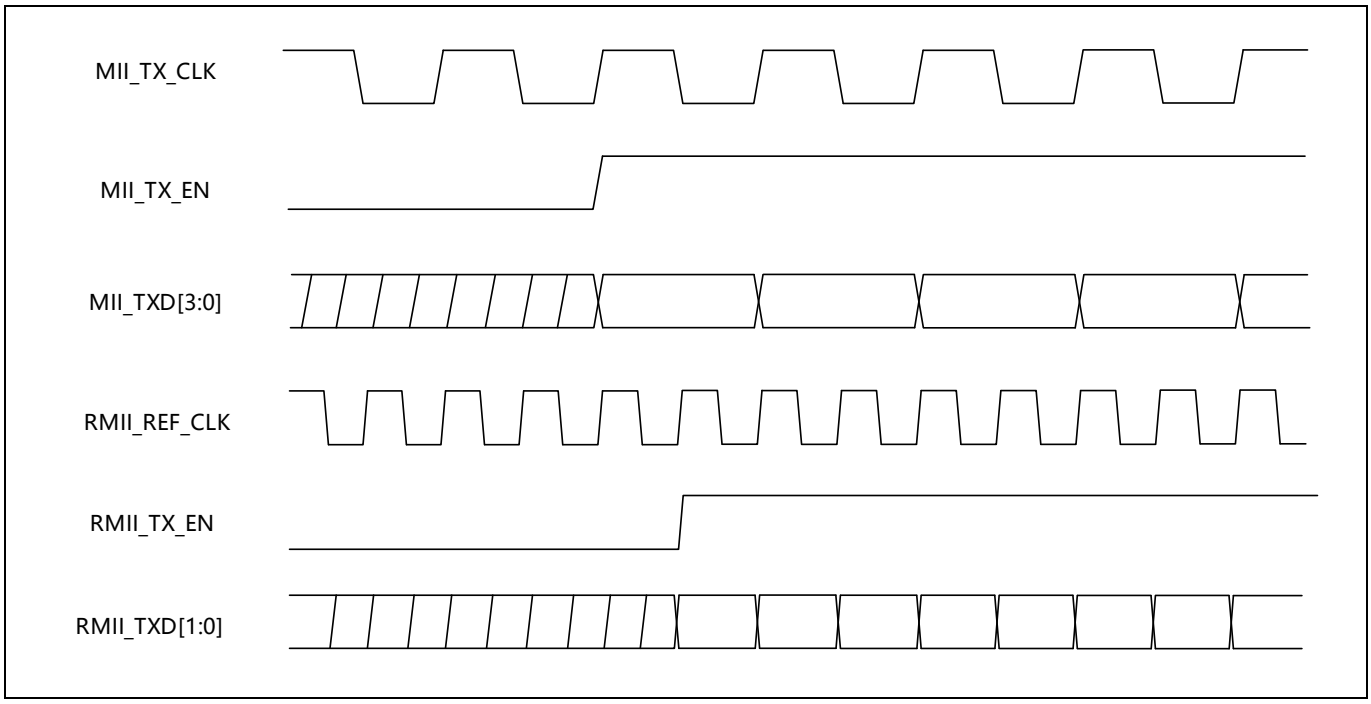


图 38-14 MII 和 RMII 模式下的帧发送



38.3.4.3. MAC 帧接收

MAC 接收的帧将推入 Rx FIFO。此 FIFO 的状态（填充级别）一旦超过配置的接收阈值（ ETH_DMAOMR 寄存器中的 RTC)，就会将其指示给 DMA，这样 DMA 可向 AHB 接口发起预配置的突发传输。

在默认直通模式下，当 FIFO 接收到 64 个字节（使用 ETH_DMAOMR 寄存器中的 RTC 位配置）或完整的数据包时，数据将弹出，其可用性将通知给 DMA。 DMA 向 AHB 接口发起传输后，数据传输将从 FIFO 持续进行，直到传输完整个数据包。完成 EOF 帧的传输后，状态字将弹出并发送到 DMA 控制器。

在 Rx FIFO 存储转发模式（通过 ETH_DMAOMR 寄存器中的 RSF 位配置）下，仅在帧完全写入接收 FIFO 后才可读出帧。在此模式下，将丢弃所有错误帧（如果内核配置为执行此操作)，这样只会读出有效帧并将其转发到应用程序。在直通模式下，某些错误帧不会被丢弃，因为在帧结束时接收到错误状态，而此时已从 FIFO 读出帧开始。

当 MAC 在 MII 上检测到 SFD 时，将启动接收操作。 MAC 内核将去除报头和 SFD，然后再继续处理帧。

检查报头字段以进行过滤， FCS 字段用于验证帧的 CRC。如果帧未通过地址滤波器，则在内核中丢弃该帧。

1. 接收协议

去除接收的帧的报头和 SFD。检测到 SFD 后， MAC 开始向接收 FIFO 发送以太网帧数据，从 SFD 后面的第一个字节（目标地址）开始发送。如果使能 IEEE 1588 时间戳功能，则在 MII 上检测到任何帧的 SFD 时，都将获取系统时间的快照。除非 MAC 滤出并丢弃帧，否则此时间戳将传递给应用程序。

如果接收的帧长度/ 类型字段小于 0x600 并且为 MAC 编程了自动去除 CRC/pad 选项，则 MAC 将向接收 FIFO 发送帧数据（数据量不超过长度/类型字段中指定的数量），然后开始丢弃字节（包括 FCS 字段）。如果长度/ 类型字段大于或等于 0x600，则不管编程的自动 CRC 去除选项的值如何， MAC 都会向 Rx FIFO 发送所有接收到的以太网帧数据。默认情况下，使能 MAC 看门狗定时器，即，超过 2048 个字节（ DA + SA + LT + 数据 + pad + FCS）的帧会被切断。可通过对 MAC 配置寄存器中的看门狗禁止 (WD) 位编程来禁止此功能。但是，即使禁止看门狗定时器，仍将切断大于 16 KB 的帧并给出看门狗超时状态。

2. 接收 CRC：自动 CRC 和 pad 去除

MAC 将检查接收帧中的任何 CRC 错误。它将计算接收的帧（包括目标地址字段到 FCS 字段）的 32 位 CRC。编码由下面的多项式定义。

$$G(x)=x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

不管是否自动去除 pad/CRC， MAC 都将接收整个帧来计算所接收帧的 CRC 校验。

3. 接收校验和减荷

对接收的以太网帧中的 IPv4 和 IPv6 帧进行检测和处理，以确保数据完整性。可通过将 ETH_MACCR 寄存器中的 IPCO 位置 1 来使能接收校验和减荷。 MAC 通过检查所接收的以太网帧的类型字段中是否存在值 0x0800 或 0x86DD，来识别 IPv4 或 IPv6 帧。此识别方法也适用于带 VLAN 标记的帧。接收校验和减荷将计算 IPv4 报头校验和，并检查其是否与接收的 IPv4 报头校验和匹配。如果指示的有效负载类型（以太网类型字段）与 IP 报头版本之间有任何不匹配，或接收的帧的字节数小于 IPv4 报头的长度字段中指示的数量（ IPv4 或 IPv6 报头中的可用字节数少于 20）， IP 报头错误位将置 1。接收校验和减荷还可以识别接收的 IP 数据报（ IPv4 或 IPv6）中的 TCP、 UDP 或 ICMP 有效负载，并正确计算此类有效负载的校验和，如 TCP、 UDP 或 ICMP 规范中所定义。它包括用于校验和计算的 TCP/UDP/ICMPv6 伪报头字节，并检查接收的校验和字段是否与计算的值匹配。此操作的结果由接收状态字中的有效负载校验和错误位给出。如果 TCP、 UDP 或 ICMP 有效负载的长度与 IP 报头中给出的预期有效负载长度不匹配，此状态位也将置 1。如 TCP/UDP/ICMP 校验和中所述，接收校验和减荷将绕过分段 IP 数据报的有效负载、带安全功能的 IP 数据报、 IPv6 路由报头以及除 TCP、 UDP 或 ICMP 以外的有效负载。此信息在接收状态中给出（无论是否绕过校验和），如 RDES0：接收描述符字 0 一节所述。在此配置中，内核不会向接收的以太网帧附加任何有效负载校验和字节。

如 RDES0：接收描述符字 0 中所述，某些寄存器位的含义会发生变化，如表 1-5 所示。

表 38-5 帧状态

位 18：以太网帧	位 27：报头校验和错误	位 28：有效负载校验和错误	帧状态
0	0	0	帧为 IEEE 802.3 帧（长度字段值小于 0x0600）
1	0	0	IPv4/IPv6 类型帧，在其中未检测到校验和错误
1	0	1	IPv4/IPv6 类型帧，在其中检测到有效负载校验和错误（参见 PCE 的说明）

1	1	0	IPv4/IPv6 类型帧，在其中检测到 IP 报头校验和错误（参见 IPCO HCE 的说明）
1	1	1	IPv4/IPv6 类型帧，在其中检测到 PCE 和 IPCO HCE
0	0	1	IPv4/IPv6 类型帧，其中不存在 IP HCE，并且由于不支持的有效负载而绕过有效负载检查
0	1	1	既不是 IPv4 也不是 IPv6 的类型帧（校验和减荷完全绕过校验和的检验）
0	1	0	保留

4. 接收帧控制器

如果复位 MAC CSR 帧过滤寄存器中的 RA 位，则 MAC 将根据目标/源地址执行帧过滤（如果应用程序决定不接收任何不良帧，如矮帧、 CRC 错误帧等，则仍需要执行另一等级的过滤）。检测到过滤失败时，帧将被丢弃且不会传输到应用程序。当过滤参数动态改变时，如果（DA-SA）过滤失败，则剩余的帧将被丢弃并且接收状态字将立即更新（零帧长度位、 CRC 错误位和矮帧错误位将置 1），指示过滤失败。在以太网掉电模式下，所有接收到的帧都将被丢弃且不会转发给应用程序。

5. 接收流量控制

MAC 将检测接收暂停帧并暂停帧发送，暂停时间为接收的暂停帧内指定的延迟（仅限全双工模式）。可通过 ETH_MACFCR 中的 RFCE 位使能或禁止暂停帧检测功能。使能接收流量控制后，将开始监视接收帧的目标地址是否与控制帧的多播地址（0x0180 C200 0001）匹配。如果检测到匹配（接收的帧的目标地址与保留的控制帧的目标地址匹配）， MAC 将根据 ETH_MACFFR 中的 PCF 位来决定是否将接收的控制帧传输到应用程序。

MAC 还将对接收控制帧的类型、操作码和暂停定时器字段进行解码。如果状态的字节计数指示 64 个字节，并且不存在任何 CRC 错误，则 MAC 发送器将暂停任何数据帧的发送，暂停时间为解码的暂停时间值乘以时隙（对于 10/100 Mb/s 模式，均为 64 字节时间）。同时，如果检测到另一个零暂停时间值的暂停帧， MAC 将复位暂停时间并管理新的暂停请求。

如果接收的控制帧与类型字段（0x8808）、操作码（0x00001）以及字节长度（ 64 字节）均不匹配，或存在 CRC 错误，则 MAC 不会生成暂停。

如果暂停帧具有多播目标地址， MAC 将根据地址匹配来过滤帧。

对于具有单播目标地址的暂停帧， MAC 将根据 DA 是否与 MAC 地址 0 寄存器的内容匹配以及 ETH_MACFCR 中的 UPDF 位是否置 1（检测具有单播目标地址的暂停帧）来进行过滤。PCF 寄存器位（ ETH_MACFFR 中的位 [7:6]）可对控制帧的过滤以及地址过滤进行控制。

6. 接收操作多帧处理

由于接收数据后状态立即可用，因此只要 FIFO 未满，就可以向其中存储帧。

7. 错误处理

如果在从 MAC 接收 EOF 数据之前 Rx FIFO 已满，则将声明上溢并丢弃整个帧，（ ETH_DMAMFBOCR 寄存器）中的上溢计数器将递增。由于上溢，状态将指示这是一个部分帧。如果（使用 ETH_DMAOMR 中的 FEF 和 FUGF 位）使能相应功能， Rx FIFO 可过滤错误帧和过小帧。

如果将接收 FIFO 配置为在存储转发模式下工作，则可过滤并丢弃所有错误帧。

在直通模式下，如果在从 Rx FIFO 读取帧的 SOF 时，该帧的状态和长度可用，则可丢弃整个错误帧。 DMA

可通过使能接收帧清空位来清空正在从 FIFO 读取的错误帧。然后到应用(DMA) 的数据传输将停止，其余帧将从内部读取并丢弃。如果可用，则可以启动下一帧传输。

8. 接收状态字

以太网帧接收结束时， MAC 向应用（DMA）输出接收状态。接收状态的详细说明与 RDES0 中的位 [31:0] 相同，请参见 RDES0：接收描述符字 0。

9. 帧长度接口

对于开关应用，应用和 MAC 之间的数据发送和接收以传输完整帧的形式进行。为了向出站端口传输帧，应用层应知道从入站端口接收的帧的长度。在每次帧接收结束时， MAC 内核在状态内提供每个接收的帧的长度。

注意： 对于由于上溢而写入 Rx FIFO 的部分帧，将指定值为 0 的帧长度。

10. MII/RMII 接收位序

每个半字节都从接收自 RMII 的双位发送到 MII，半字节发送顺序如图 1-5 所示。先接收位序较低的位（ D0 和 D1），再接收位序较高的位（ D2 和 D3）。

图 38-15 接收位序

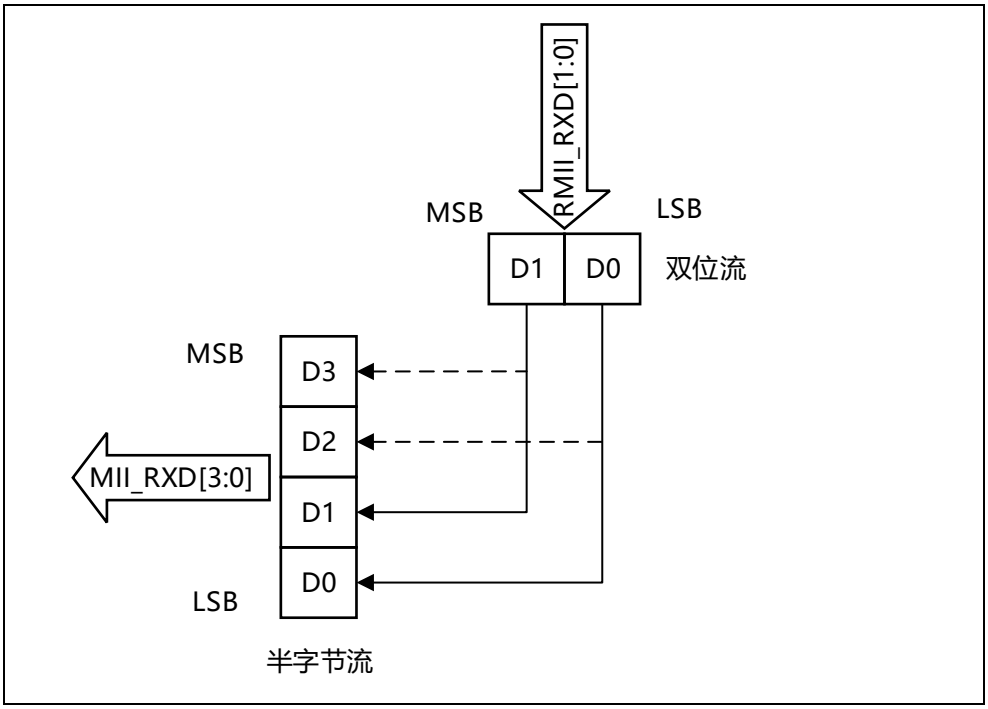


图 38-16 无错误接收

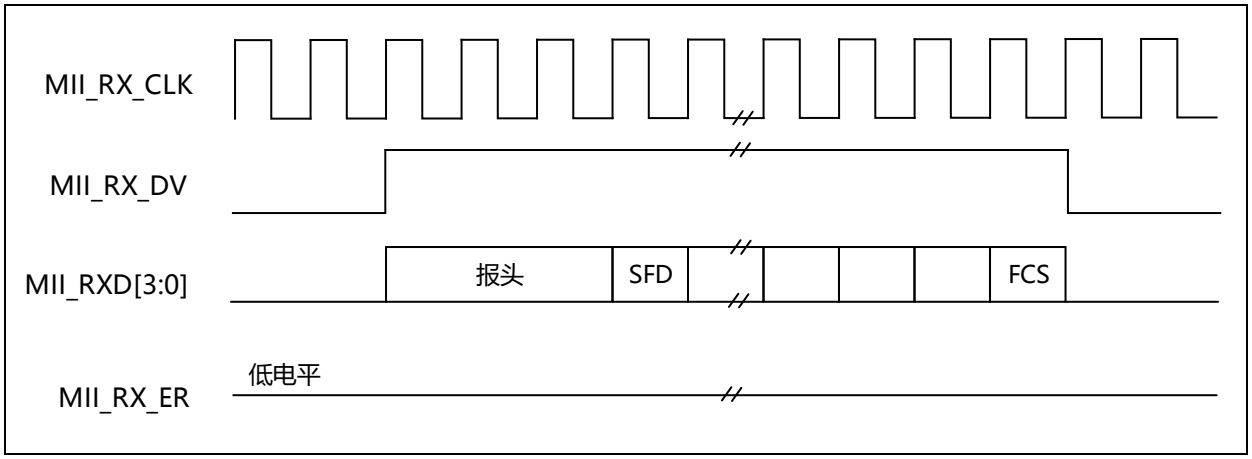
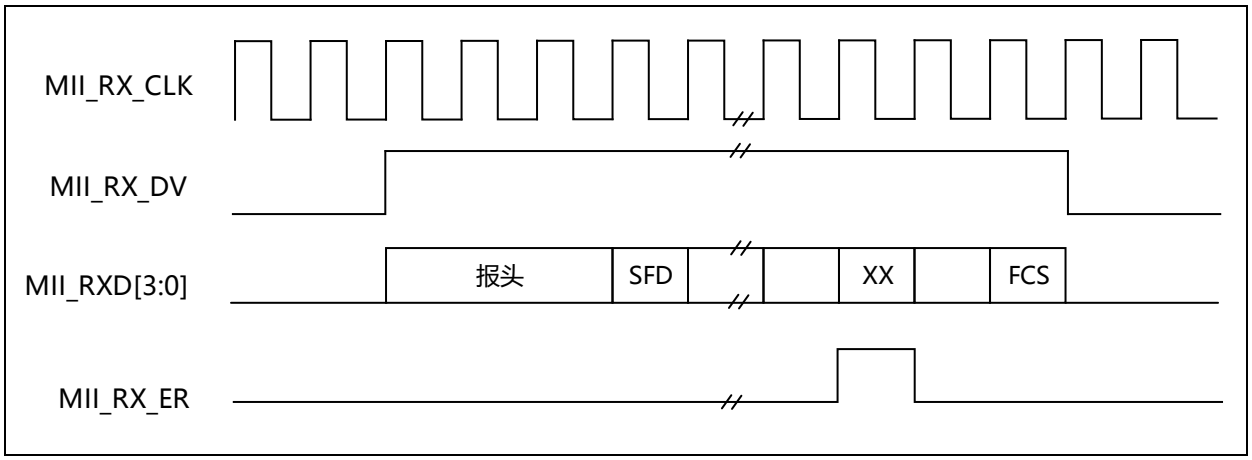


图 38-17 有错误接收



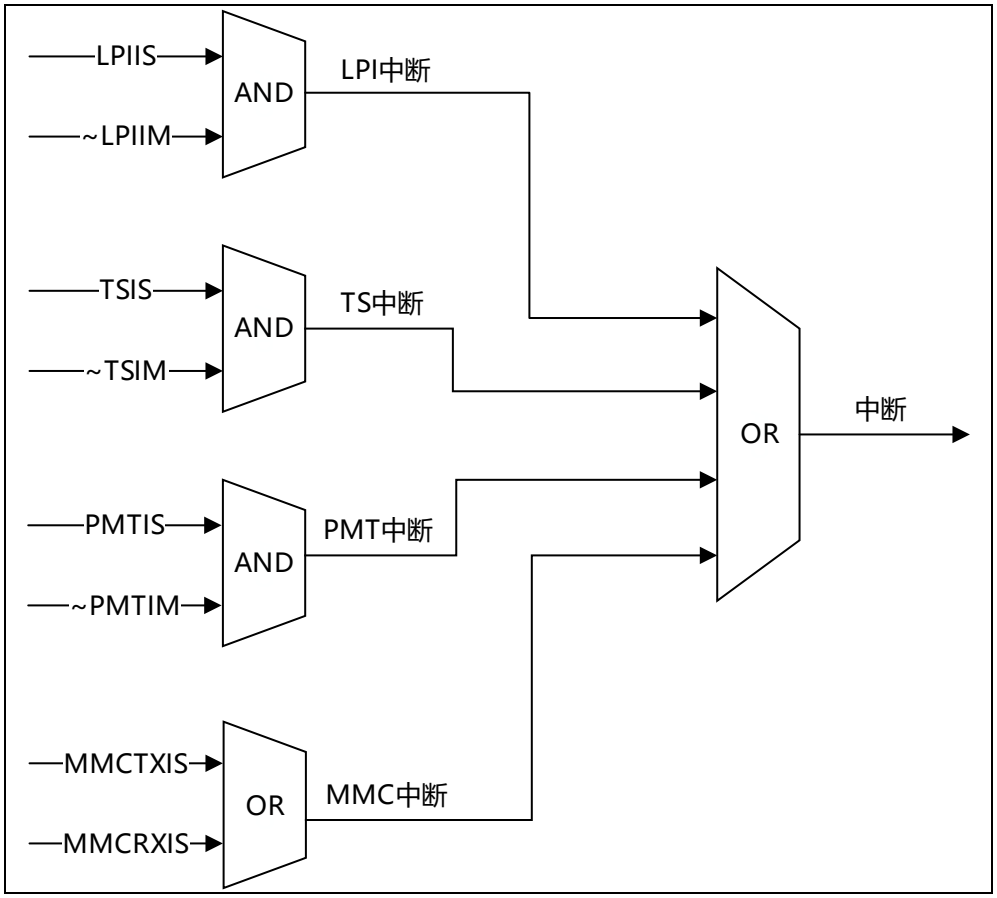
38.3.4.4. MAC 中断

MAC 内核可由于各种事件而生成中断。

ETH_MACISR 寄存器描述了可导致 MAC 内核生成中断的事件。可通过将中断屏蔽寄存器中相应的屏蔽位置 1 来阻止各事件生成中断。

中断寄存器位仅指示报告事件的模块。必须读取相应的状态寄存器和其它寄存器才能清除中断。例如，如果中断寄存器的位 3 设置为高电平，则指示在掉电模式下接收到魔术数据包或网络唤醒帧。必须读取 ETH_MACPMTCSR 寄存器才能清除此中断事件

图 38-18 MAC 内核中断屏蔽方案



38.3.4.5. MAC 过滤

1. 地址过滤

地址过滤将检查所有接收的帧的目标地址和源地址并相应报告地址过滤状态。地址检查基于应用程序选择的不同参数（帧过滤寄存器）。 还可以识别过滤的帧：多播帧或广播帧。

地址过滤使用站的物理（MAC）地址和多播散列表进行地址检查。

2. 单播目标地址过滤

MAC 支持多达 4 个用于单播完美过滤的 MAC 地址。如果选择完美过滤（复位帧过滤寄存器中的 HU 位）， MAC 会将接收的单播地址的所有 48 位与编程的 MAC 地址进行比较来确定是否匹配。默认情况下，始终使能 MacAddr0，其它地址 MacAddr1—MacAddr3 则通过单独的使能位进行选择。将其它地址 (MacAddr1—MacAddr3) 的各个字节与接收的相应 DA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节。这有助于 DA 的组地址过滤。在散列过滤模式（ HU 位置 1）下， MAC 将使用 64 位散列表执行对单播地址的不完美过滤。对于散列过滤， MAC 将使用接收的目标地址的 6 个高 CRC 位来索引散列表的内容。值为 000000 时，选取所选寄存器中的位 0；值为 111111 时，选取散列表寄存器中的位 63。如果相应位（由 6 位 CRC 指示）已置 1，将认为单播帧已通过散列过滤，否则认为帧未能通过散列过滤。

3. 散列或完美地址过滤

可通过将帧过滤寄存器中的 HPF 位置 1 并将相应的 HU 或 HM 位置 1，来将 DA 过滤配置为在其 DA 与散列过滤或完美过滤匹配时允许帧通过。此配置适用于单播帧和多播帧。如果 HPF 位复位，则只有一种过滤方式（散列或完美）应用于接收的帧。

4. 广播地址过滤

在默认模式下， MAC 不过滤任何广播帧。但是，如果将帧过滤寄存器中的 DBF 位置 1 来将 MAC 编程为拒绝所有广播帧，则会丢弃任何广播帧。

5. 单播源地址过滤

MAC 还可以根据接收的帧的源地址字段来执行完美过滤。默认情况下， MAC 将 SA 字段与 SA 寄存器中编程的值进行比较。可通过将相应寄存器中的位 30 置 1，来将 MAC 地址寄存器 [1:3] 配置为包含 SA 而不是 DA 进行比较。带 SA 的组地址过滤也受到支持。如果帧过滤寄存器中的 SAF 位置 1，则 MAC 将丢弃未通过 SA 过滤的帧。否则， SA 过滤的结果将通过接收状态字中的状态位给出（参见 RDES0：接收描述符字 0)。

SAF 位置 1 时，对 SA 过滤和 DA 过滤的结果进行与运算，以决定是否需要转发帧。这意味着任何一个过滤未通过都将丢弃帧。两个过滤必须都通过，才能将帧转发到应用。

6. 反向过滤操作

对于目标地址和源地址过滤，可在最终输出时选择反转过滤匹配结果。这分别由帧过滤寄存器中的 DAIF 和 SAIF 位控制。 DAIF 位同时适用于单播和多播 DA 帧。在此模式下，将反转单播/多播目标地址过滤的结果。类似地，当 SAIF 位置 1 时，将反转单播 SA 过滤的结果。表 1-6 和表 1-7 按照接收帧的类型汇总了目标地址和源地址过滤。

表 38-6 目标地址过滤

帧类型	PM	HPF	HU	DAIF	HM	PAM	DBF	DA 过滤操作
广播	1	X	X	X	X	X	X	通过
	0	X	X	X	X	X	0	通过
	0	X	X	X	X	X	1	不通过

单播	1	X	X	X	X	X	X	通过所有帧
	0	X	0	0	X	X	X	完美/组过滤匹配时通过
	0	X	0	1	X	X	X	完美/组过滤匹配时不通过
	0	0	1	0	X	X	X	散列过滤匹配时通过
	0	0	1	1	X	X	X	散列过滤匹配时不通过
	0	1	1	0	X	X	X	散列或完美/组过滤匹配时通过
	0	1	1	1	X	X	X	散列或完美/组过滤匹配时不通过
多播	1	X	X	X	X	X	X	通过所有帧
	X	X	X	X	X	1	X	通过所有帧
	0	X	X	0	0	0	X	完美/组过滤匹配时通过，并在 PCF=0X 时丢弃暂停控制帧
	0	0	X	0	1	0	X	散列过滤匹配时通过，并在 PCF=0X 时丢弃暂停控制帧
	0	1	X	0	1	0	X	散列或完美/组过滤匹配时通过，并在 PCF=0X 时丢弃暂停控制帧
	0	X	X	1	0	0	X	完美/组过滤匹配时不通过，并在 PCF=0X 时丢弃暂停控制帧
	0	0	X	1	1	0	X	散列过滤匹配时不通过，并在 PCF=0X 时丢弃暂停控制帧
	0	1	X	1	1	0	X	散列或完美/组过滤匹配时不通过，并在 PCF=0X 时丢弃暂停控制帧

表 38-7 源地址过滤

帧类型	PM	SAIF	SAF	SA 过滤操作
单播	1	X	X	通过所有帧
	0	0	0	完美/组过滤匹配时通过，但不丢弃未通过的帧
	0	1	0	完美/组过滤匹配时不通过，但不丢弃帧
	0	0	1	完美/组过滤器匹配时通过并将不通过的帧丢弃
	0	1	1	完美/组过滤器匹配时不通过并将不通过的帧丢弃

38.3.4.6. MAC 回送模式

MAC 支持对发送到其接收器的帧进行回送。默认情况下，禁止 MAC 回送功能，可通过编程 MAC ETH_MACCR 寄存器中的 Loopback 位使能该功能。

38.3.4.7. MAC 管理计数器：MMC

MAC 管理计数器（MMC）保持一组寄存器来收集有关已接收帧和已发送帧的统计信息。其中包括一个用于控制各寄存器行为的控制寄存器、两个包含当前中断状态的 32 位寄存器（接收和发送），以及两个包含中断寄存器掩码的 32 位寄存器（接收和发送）。这些寄存器均可从应用程序中访问。每个寄存器均为 32 位宽。

以太网寄存器说明介绍了各个计数器，并列出了各统计计数器的地址。该地址用于对所需发送/接收计数器进行读/写访问。

接收 MMC 计数器针对通过地址过滤的帧进行更新。已丢弃帧的统计信息不会进行更新，除非丢弃的帧为小于 6 字节的矮帧（ DA 字节不能完全接收）。

1. 好的发送帧与接收帧

如果帧成功发送，则将发送的帧视为“好帧”。换句话说，如果帧发送过程没有因为以下错

误而中止，则认为发送的帧是好帧：

- Jabber 超时
- 无载波/载波丢失
- 延迟冲突
- 帧下溢
- 过度延迟
- 过度冲突

如果不存在以下错误，则认为接收帧是“好帧”：

- CRC 错误
- 矮帧（短于 64 字节）
- 对齐错误（仅限 10/100 Mb/s）
- 长度错误（仅限非类型帧）
- 超出范围（仅限非类型帧，超过最大大小）
- MII_RXER 输入错误
- 最大帧大小取决于帧类型，如下：
- 无标记帧的最大大小 = 1518
- VLAN 帧的最大大小 = 1522

38.3.4.8. 电源管理：PMT

本部分介绍 MAC 支持的电源管理（PMT）机制。 PMT 支持接收网络（远程）唤醒帧和魔术数据包帧。PMT 可为 MAC 接收到的唤醒帧和魔术数据包帧生成中断。可通过远程唤醒帧使能位以及魔术数据包使能位使能 PMT 模块。这些使能位（ WFE 和 MPE）位于 ETH_MACPMTCSR 寄存器中，可由应用编程。在 PMT 中使能掉电模式时， MAC 将丢弃所有接收到的帧并且不会将这些帧转发给应用。仅当接收到魔术数据包或远程唤醒帧且使能相应检测时， MAC 才会退出掉电模式。

1. 远程唤醒帧过滤寄存器

有八个唤醒帧过滤寄存器。要对每个寄存器执行写操作，需要逐个值加载唤醒帧过滤寄存器。连续加载唤醒帧过滤寄存器八次，便可对唤醒帧过滤寄存器加载所需的值。读操作与写操作相同。要读取八个值，必须读唤醒帧过滤寄存器八次后，才能到达最后一个寄存器。每次读/写操作都会将唤醒帧过滤寄存器指向下一个过滤寄存器。

图 38-19 唤醒帧过滤寄存器

唤醒帧过滤寄存器0	过滤器0字节屏蔽							
唤醒帧过滤寄存器1	过滤器1字节屏蔽							
唤醒帧过滤寄存器2	过滤器2字节屏蔽							
唤醒帧过滤寄存器3	过滤器3字节屏蔽							
唤醒帧过滤寄存器4	RSV	过滤器3命令	RSV	过滤器2命令	RSV	过滤器1命令	RSV	过滤器0命令
唤醒帧过滤寄存器5	过滤器3偏移		过滤器2偏移		过滤器1偏移		过滤器0偏移	
唤醒帧过滤寄存器6	过滤器1 CRC-16				过滤器0 CRC-16			
唤醒帧过滤寄存器7	过滤器3 CRC-16				过滤器2 CRC-16			

● 过滤器 i 字节屏蔽

该寄存器定义过滤器 i（ 0、 1、 2 和 3）检测帧的哪些字节来确定帧是否为唤醒帧。MSB（第 31 位）必须为零。位 j [30:0] 为字节屏蔽。如果将字节屏蔽的位 j（字节数）置 1，则传入帧的过滤器 i 偏移 + j 由 CRC 模块处理；否则将忽略过滤器 i 偏移 + j。

● 过滤器 i 命令

该 4 位命令控制过滤器 i 操作。位 3 指定地址类型，定义模式的目标地址类型。当该位置 1 时，模式只适用于多播帧。当该位复位时，模式只适用于单播帧。位 2 和位 1 为保留位。位 0 为过滤器 i 的使能位；如果位 0 未置 1，则将禁止过滤器 i。

● 过滤器 i 偏移

该寄存器定义过滤器 i 要检测的帧的偏移（在帧范围内）。该 8 位模式偏移是要检测的过滤器 i 第一个字节的偏移。允许的最小值为 12，表示帧的第 13 个字节（偏移值 0 表示帧的第一个字节）。

● 过滤器 i CRC-16

该寄存器包含根据模式计算的 CRC_16 值，以及对唤醒过滤寄存器模块编程的字节屏蔽。

2. 远程唤醒帧检测

当 MAC 处于睡眠模式并已使能 ETH_MACPMTCSR 寄存器中的远程唤醒位时， MAC 可在接收到远程唤醒帧后恢复正常工作。应用可通过连续对唤醒帧过滤寄存器地址执行写操作，来写入全部八个唤醒过滤寄存器。应用可通过向 ETH_MACPMTCSR 寄存器中的位 2 写入 1 来使能远程唤醒。 PMT 支持四种可编程过滤器，这些过滤器提供不同的接收帧模式。如果传入帧通过过滤命令的地址过滤，并且过滤器 CRC-16 与传入的检测模式相匹配，则可接收唤醒帧。 Filter_offset（最小值 12，表示帧的第 13 个字节）确定要检测的帧的偏移。过滤器字节屏蔽确定必须检测帧的哪些字节。必须将字节屏蔽的第 31 位置 0。只需要检查唤醒帧是否存在长度错误、 FCS 错误、帧尾错误、 MII 错误、冲突，确保其不是矮帧。即使唤醒帧长度超过 512 字节，但只要帧的 CRC 值有效，帧便有效。 ETH_MACPMTCSR 寄存器中的唤醒帧检测会针对每个接收到的远程唤醒帧进行更新。此外，还会生成 PMT 中断（如果已使能）来指示已接收到远程唤醒帧。

3. 魔术数据包检测

魔术数据包帧基于一种特殊的方法，这种方法使用 AMD 公司的魔术数据包技术对网络上处于睡眠模式下的器

件上电。 MAC 接收称为魔术数据包的特定信息包，此信息包的目标地址是网络上的节点。只对目标地址为器件或多播地址的魔术数据包进行检查，以确定这些数据包是否满足唤醒要求。对通过地址过滤（单播或多播）的魔术数据包进行检测，确定其是否符合远程唤醒数据格式，即 6 个字节的数据所有位为全 ‘1’ 的数据包加上重复出现 16 次的 MAC 地址。应用通过向 ETH_MACPMTCSR 寄存器中的位 1 写入 1 来使能魔术数据包唤醒。 PMT 模块持续监视目标地址为对应于指定魔术数据包模式的节点的每个帧。检查每个接收帧的目标地址和源地址字段之后是否为 0xFFFF FFFF FFFF 形式。之后， PMT 模块检查帧是否存在重复 16 次且没有任何断开或中断的 MAC 地址。如果重复 16 次的地址中存在中断，则会再次扫描传入帧中是否存在 0xFFFF FFFF FFFF 形式。这 16 次重复可位于帧中的任何位置，但必须在同步数据流后面 (0xFFFF FFFF FFFF)。此外，只要检测到重复 16 次的 MAC 地址，器件就可以接收多播帧。如果节点的 MAC 地址为 0x0011 2233 4455，则 MAC 扫描的数据序列为：

目标地址源地址 .....FFFF FFFF FFFF

0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455

0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455

0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455

0011 2233 4455 0011 2233 4455 0011 2233 4455 0011 2233 4455

...CRC

ETH_MACPMTCSR 寄存器中的魔术数据包检测会针对接收到的魔术数据包进行更新。此

外，还会生成 PMT 中断（如果已使能）来指示已接收到魔术数据包。

4. 掉电期间的系统的注意事项

如果使能 EXTI 中断线 19，以太网 PMT 模块能够在系统处于停止模式时检测帧。

MAC 接收器状态机在掉电模式期间应保持使能状态。这意味着，由于 ETH_MACCR 寄存器中的 RE 位与魔术数据包/网络唤醒帧检测有关，因此必须始终将该位置 1。但是，在掉电模式期间，应通过将 ETH_MACCR 寄存器中的 TE 位清零来关闭发送器状态机。此外，在掉电模式期间，应禁止以太网 DMA，原因是此时无需将魔术数据包/ 网络唤醒帧复制到 SRAM。要禁止以太网 DMA，需将 ETH_DMAOMR 寄存器中的 ST 位和 SR 位（分别对应发送 DMA 和接收 DMA）清零。

推荐的掉电和唤醒顺序如下：

1）禁止发送 DMA，并等待所有之前的帧发送完成。接收到发送中断 ETH_DMASR 寄存器[0] 时，便可检测到这些帧发送。

2）通过将 ETH_MACCR 配置寄存器中的 RE 位和 TE 位清零来禁止 MAC 发送器和 MAC 接收器。

3）等待接收 DMA 清空 Rx FIFO 中的所有帧。

4）禁止接收 DMA。

5）配置并使能 EXTI 中断线 19，以生成一个事件或中断。

6）如果将 EXTI 中断线 19 配置为生成中断，还必须正确配置 ETH_WKUP_IRQ 处理程序功能，此功能用于清零 EXTI 中断线 19 的挂起位。

7）通过将 ETH_MACPMTCSR 寄存器中的 MFE/WFE 位置 1 来使能魔术数据包/网络唤醒帧检测。

8）通过将 ETH_MACPMTCSR 寄存器中的 PD 位置 1 来使能 MAC 掉电模式。

9）通过将 ETH_MACCR 寄存器中的 RE 位置 1 来使能 MAC 接收器。

10）进入系统的停止模式

11）接收到有效唤醒帧时，以太网外设会退出掉电模式。

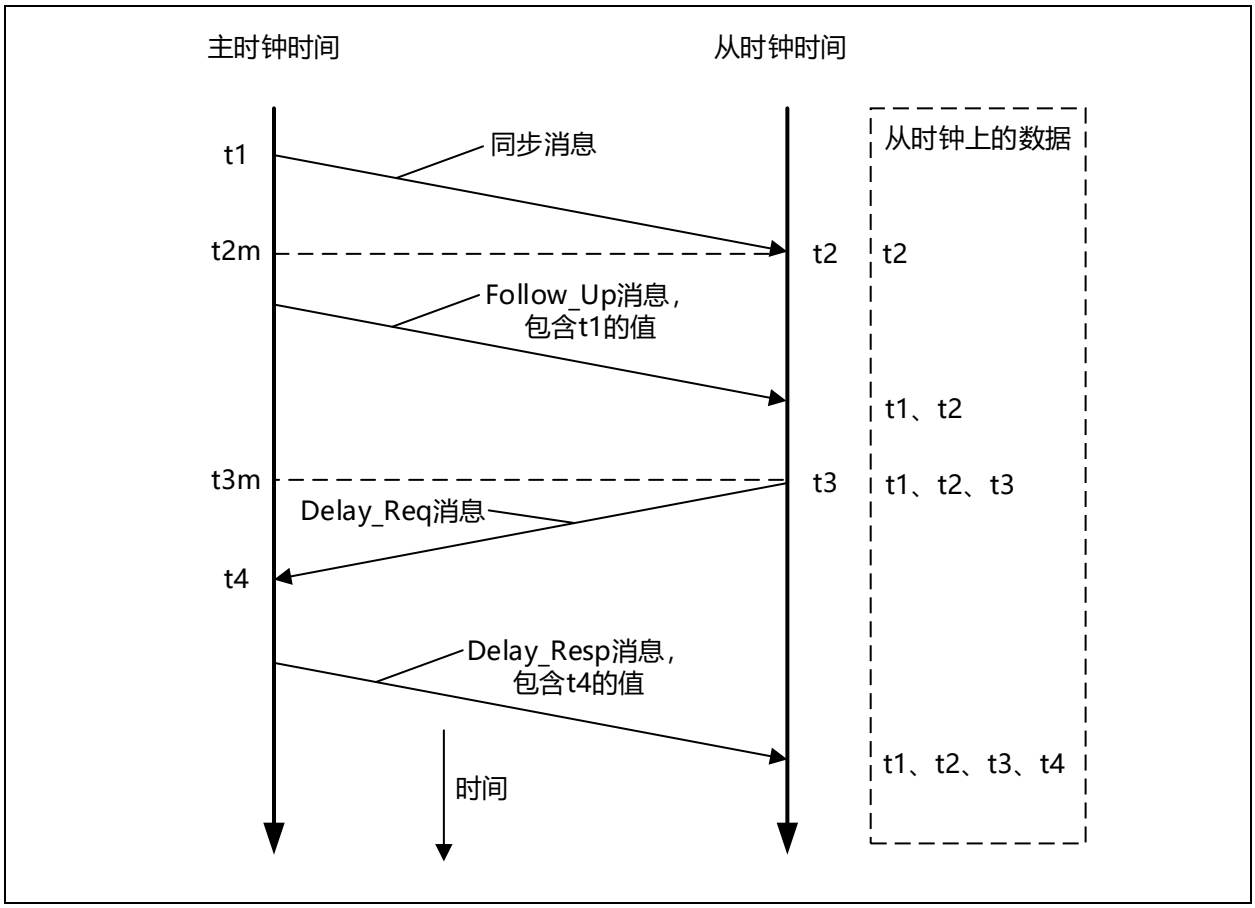
12）读取 ETH_MACPMTCSR 以清零电源管理事件标志，使能 MAC 发送器状态机并接收和发送 DMA。

13）配置系统时钟：使能 HSE 并设置时钟。

38.3.4.9. 精密时间协议(IEEE 1588 PTP)

IEEE 1588 标准定义了一种协议，此协议支持使用网络通信、局域计算和分布式对象等技术实现的测量和控制系统中的精密时钟同步。该协议适用于利用支持多播消息传送的局域网（包括但不限于以太网）进行通信的系统。该协议用于对非均匀系统进行同步，这类系统包含固有精度、分辨率和稳定性都不断变化的时钟。该协议支持亚秒范围的系统级同步精度，并且需要极少的网络和本地时钟计算资源。这种基于消息的协议称为精密时间协议（PTP），它通过 UDP/IP 传送。系统或网络归类为主节点和从节点，用于分配时序/时钟信息。该协议通过交换 PTP 消息来同步从节点与主节点， 图 1-20 中对此有详细说明。

图 38-20 网络时间同步



1）主节点向其所有节点广播 PTP 同步消息。同步消息包含主节点的参考时间信息。消息离开主节点系统的时间为 t1。对于以太网端口，必须通过 MII 接口来捕获该时间。

2）从节点接收同步消息并利用其参考时序捕获准确时间 t2。

3）主节点随后会向从节点发送一个包含 t1 信息的 Follow_up 消息以备后用。

4）从节点向主节点发送一个 Delay_Req 消息，标出该帧离开 MII 的准确时间 t3。

5）主节点接收该消息，捕获消息进入其系统的准确时间 t4。

6）主节点将 Delay_Resp 消息中的 t4 信息发送给从节点。

7）从节点使用 t1、 t2、 t3 和 t4 这四个值来同步其本地参考时序与主节点的参考时序。

大多数协议都在 UDP 层之上通过软件实现。但如上所述，要通过 MII 接口来捕获特定 PTP 包进入或离开以太网端口时的准确时间，需要硬件支持。必须捕获该时序信息并将其返回给软件，以实现高精度的 PTP。

1. 参考时序源

根据 IEEE 1588 规范的定义，要获取时间快照，内核需要一个 64 位格式的参考时间（分成两个 32 位通道，高 32 位表示时间的秒数，低 32 位表示时间的纳秒数）。

PTP 参考时钟输入用于在内部生成参考时间（亦称作系统时间）以及捕获时间戳。该参考时钟的频率必须大于等于时间戳计数器的分辨率。主节点与各从节点之间的同步精度目标约为 100 ns。

系统时间校准方法一节中介绍了系统时间的生成、更新与修改。

精度取决于 PTP 参考时钟输入周期、振荡器的特性（漂移）以及同步过程的频率。

由于从 Tx 和 Rx 时钟输入域到 PTP 参考时钟域的同步， 时间戳锁存值的不确定度为 1 个参考时钟周期。如果加上分辨率导致的不确定度，则会增加一半的时间戳周期。

2. 使用 PTP 功能发送帧

在 MII 上输出帧的 SFD 时，将捕获时间戳。对于需要捕获时间戳的帧，可按帧为单位进行控制。换句话说，可对每个发送帧进行标记以指示是否有必要捕获该帧的时间戳。无需处理发送帧即可识别 PTP 帧。可通过发送描述符中的控制位执行帧控制。捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的发送状态一同发送回相应的发送描述符内，从而自动将时间戳与特定 PTP 帧相连。 64 位时间戳信息会写回 TDES2 和 TDES3 字段，其中 TDES2 会保持时间戳的 32 个最低有效位。

3. 使用 PTP 功能接收帧

使能 IEEE 1588 时间戳功能时，以太网 MAC 将捕获 MII 上接收到的所有帧的时间戳。 MAC 会在完成帧接收过程时提供时间戳。捕获到的时间戳返回给应用的方式与提供帧状态的方式相同。时间戳会随帧的接收状态一同发送回相应的接收描述符内。 64 位时间戳信息会写回 RDES2 和 RDES3 字段，其中 RDES2 会保持时间戳的 32 个最低有效位。

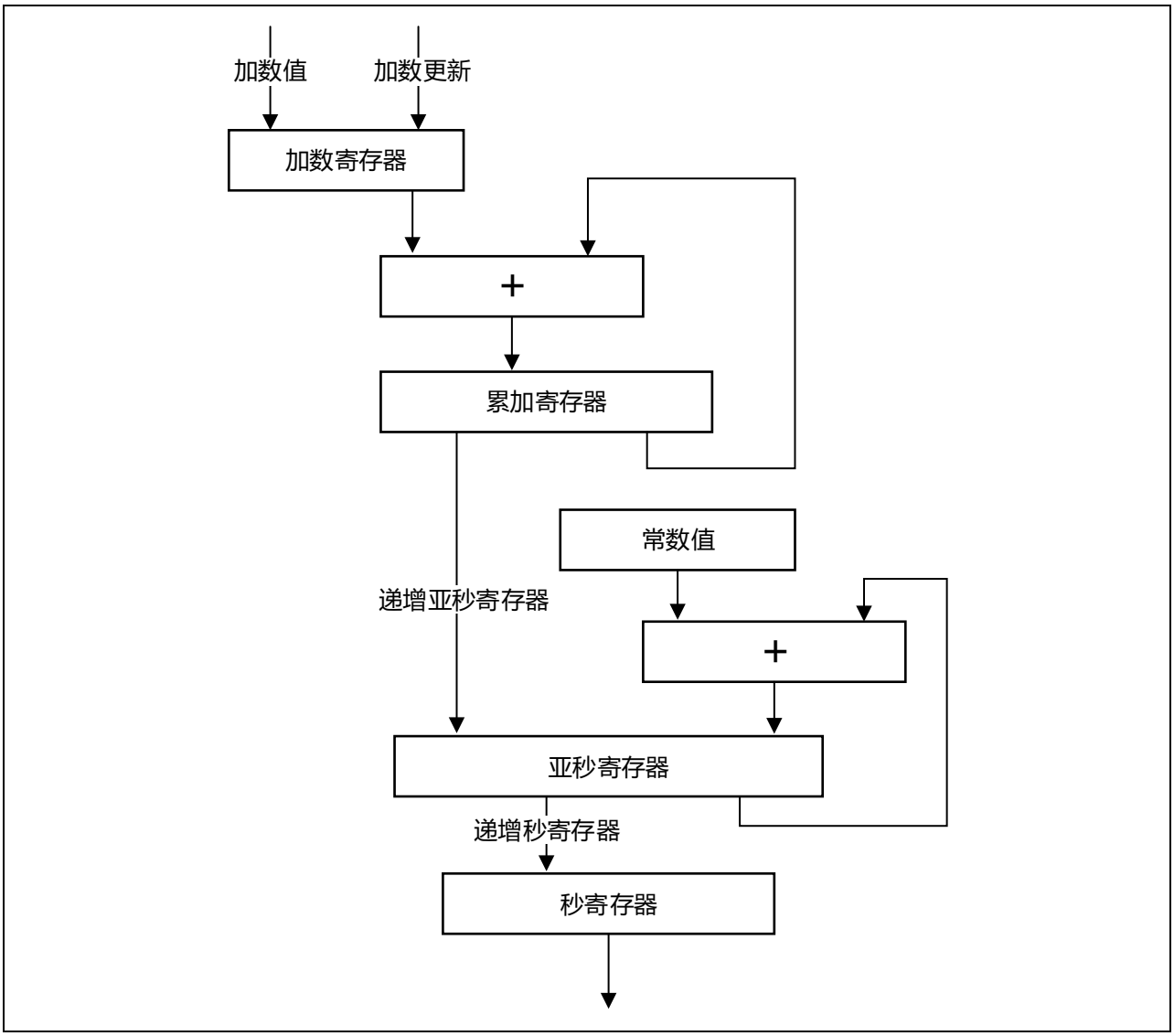
4. 系统时间校准方法

使用 PTP 输入参考时钟 HCLK 更新 64 位 PTP 时间。该 PTP 时间可用作时钟源，以获取 MII 上发送或接收的以太网帧的快照（时间戳）。可使用粗略校准方法或精密校准方法对系统时间定时器进行初始化或校准。

使用粗略校准方法时，初始值或偏移值会写入时间戳更新寄存器。对于初始化，会将时间戳更新寄存器中的值写入系统时间计数器；对于系统时间校准，会将偏移值（ 时间戳更新寄存器）加到系统时间中或从系统时间中减去。

使用精密校准方法时，从时钟（参考时钟）频率相对于主时钟（如 IEEE 1588 中定义）的偏移会在一段时间内进行校准，而不像粗略校准方法中那样，在单个时钟周期内进行校准。校准时间越长，越有助于保持线性时间，并且不会导致各 PTP 同步消息间隔之间的参考时间发生剧烈变化（或者大型抖动）。在此方法中，会使用一个累加器对加数寄存器中的内容求和，如图 1-21 所示。累加器生成的算数进位将用作使系统时间计数器递增的脉冲。累加器和加数寄存器均为 32 位寄存器。此处，累加器用作高精度频率乘法器或除法器。 图 1-21 给出了该算法。

图 38-21 使用精密校准方法更新系统时间



系统时间更新逻辑需要使用 50 MHz 的时钟频率，以达到 20 ns 的精度。分频是指参考时钟频率与所需时钟频率的比率。因此，如果我们说参考时钟 (HCLK) 的频率为 66MHz，则通过计算可知分频比为 66 MHz/50 MHz = 1.32。故此，寄存器中设置的默认加数值为2^{32}/1.32，相当于 0xC1F0 7C1F。

如果参考时钟向下偏移，例如降至 65 MHz，则分频比为 65/50 或 1.3，加数寄存器中要设置的值为 2^{32}/1.30，相当于 0xC4EC 4EC4。如果时钟向上偏移，例如升至 67 MHz，则必须将加数寄存器设置为 0xBF0 B7672。当时钟偏移为零时，应编程的默认加法值为 0xC1F07C1F (2^{32}/1.32)。

在图 1-21 中，用于使亚秒寄存器递增的常数值为 0d43。这可使系统时间的精度达到 20 ns

（换句话说，增量步长时间为 20 ns)。

软件必须根据同步消息对频率偏移进行计算，并相应更新加数寄存器。首先，使用加数寄存器中的 FreqCompensationValue0 设置从时钟。该值的计算公式如下：

FreqCompensationValue0 = 2^{32} / FreqDivisionRatio

如果起初假定 MasterToSlaveDelay 对于连续的同步消息是相同的，则必须应用下述算法。数个同步周期之后，频率会锁定。从时钟随后可确定 MasterToSlaveDelay 的精确值，并使用新值重新与主时钟同步。

该算法如下：

- 在 MasterSyncTime (n) 时刻，主时钟向从时钟发送同步消息。从时钟在其本地时钟为 SlaveClockTime (n) 时接收到该消息，并用如下公式计算 MasterClockTime (n)：MasterClockTime (n) = MasterSyncTime (n) + MasterToSlaveDelay (n)

- 当前同步周期的主时钟计数 MasterClockCount (n) 的计算公式如下：MasterClockCount (n) = MasterClockTime (n) - MasterClockTime (n - 1)（假定 MasterToSlaveDelay 对于同步周期 n 和 n - 1 是相同的）
- 当前同步周期的从时钟计数 SlaveClockCount (n) 的计算公式如下：SlaveClockCount (n) = SlaveClockTime (n) - SlaveClockTime (n - 1)
- 当前同步周期的主从时钟计数差值 SlaveClockCount (n) 的计算公式如下：ClockDiffCount (n) = MasterClockCount (n) - SlaveClockCount (n)
- 从时钟的分频系数 FreqScaleFactor (n) 的计算公式如下：FreqScaleFactor (n) = (MasterClockCount (n) + ClockDiffCount (n)) / SlaveClockCount (n)
- 加数寄存器的频率补偿值 FreqCompensationValue (n) 的计算公式如下：FreqCompensationValue (n) = FreqScaleFactor (n) × FreqCompensationValue (n - 1)

理论上，该算法可在一个同步周期内实现锁定；但是，由于网络传播延迟和工作条件会不断变化，因此可能需要多个周期。

该算法可进行自校准：如果出于某些原因导致最初通过主时钟设置的从时钟不正确，则该算法会花费更多同步周期将其校准。

5. 系统时间生成初始化的编程步骤

通过将时间戳控制寄存器（ETH_PTPTSCR）中的位 0 置 1 来使能时间戳功能。但之后必须对时间戳计数器进行初始化才能启动时间戳操作。适当的顺序如下：

1）通过将 MACIMR 寄存器中的位 9 置 1 以屏蔽时间戳触发中断。

2）编程时间戳寄存器位 0 以使能时间戳。

3）根据 PTP 时钟频率编程亚秒递增计数器。

4）如果使用精密校准方法，编程时间戳加数寄存器并将时间戳控制寄存器的位 5 置 1（加数寄存器更新）。

5）轮询时间戳控制寄存器，直到位 5 清零。

6）要选择精密校准方法（根据需要），编程时间戳控制寄存器位 1。

7）用适当的时间值编程时间戳高位更新寄存器和时间戳低位更新寄存器。

8）将时间戳控制寄存器位 2 置 1（时间戳初始化）。

9）用时间戳更新寄存器中写入的值初始化时间戳计数器后，时间戳计数器便开始运行。

10）使能 MAC 接收器和发送器以使时间戳功能正常运行。

注意： 如果将 ETH_PTPTSCR 寄存器中的位 0 清零来禁止时间戳操作，必须重复执行以上步骤以重新启动时间戳操作

6. 使用粗略校准方法更新系统时间的编程步骤

要在一个过程（粗略校准方法）中同步或更新系统时间，请按照以下步骤执行：

1）将偏移（正偏移或负偏移）写入时间戳更新高位和低位寄存器。

2）将时间戳控制寄存器中的位 3 (TU) 置 1。

3）当 TU 位清零时，时间戳更新寄存器中的值将加到系统时间中或者从系统时间中减去。

7. 使用精密校准方法更新系统时间的编程步骤

要同步或更新系统时间以减少系统时间抖动（精密校准方法），请按照以下步骤执行：

1）借助系统时间校准方法一节中介绍的算法，计算想要增加或减少系统时间增量时采用的速率。

2）使用新值更新加数寄存器，并将时间戳控制寄存器的位 5 置 1。

3）等待加数寄存器中的新值生效。这可以通过在系统时间达到目标值后，激活时间戳触发中断来实现。

4）在目标时间高位和低位寄存器中编程所需目标时间。通过将 ETH_MACIMR 寄存器中的位 9 清零以取消屏蔽时间戳中断。

5）将时间戳控制寄存器位 4 (TITE) 置 1。

6）当该寄存器产生中断时，读取 ETH_MACSR 寄存器。

7）用旧值重新编程时间戳加数寄存器并再次将 ETH_PTPSCR 位 5 置 1。

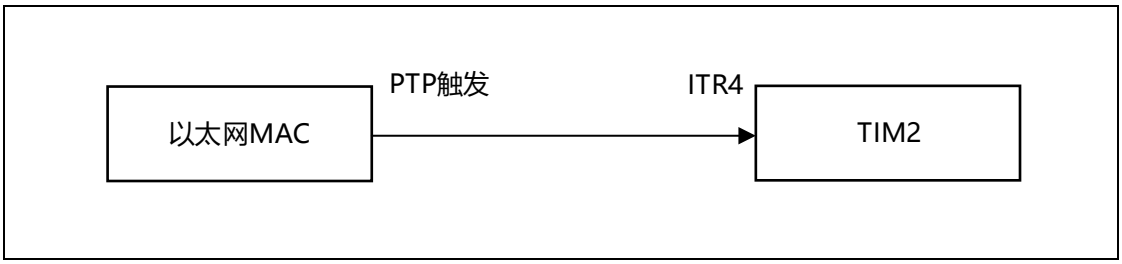
8. 与 TIM2 内部相连的 PTP 触发

当系统时间大于目标时间时， MAC 会提供触发中断。使用中断会引起已知延迟，并导致命令执行时间出现不确定性。

为避免这种不确定性，可在系统时间大于目标时间时将 PTP 触发输出信号置为高电平。该信号内部连接到 TIM2 输入触发。通过该信号，由已同步的 PTP 系统时间触发的输入捕获功能、输出比较功能以及定时器的波形均可使用。这时不会引起不确定性，因为定时器的时钟以及 PTP 参考时钟 (HCLK) 是同步的。

该 PTP 触发信号会连接到可用软件选择的 TIM2 ITR4 输入。该连接可通过 TIM2 从模式寄存器 (TIM2_SMCR) 中的 TS[3:0] 进行选择。 图 1-22 显示了该连接。

图 38-22. PTP 触发输出与 TIM2 ITR4 的连接



9. PTP 每秒脉冲数输出信号

该 PTP 脉冲输出用于检查网络中所有节点之间的同步性。要能够测试本地从时钟与主参考

时钟之间的差值，可为两个时钟提供每秒脉冲数 (PPS) 输出信号（根据需要，可将该信号连接到示波器）。这样便可测量两个信号间的偏差。 PPS 输出的脉冲宽度为 125 ms。

PPS 输出可通过 TIM2 从模式寄存器 (TIM2_SMCR) 中的位 TS[3:0] 进行选择。

PPS 输出的默认频率为 1 Hz。可使用 PPSFREQ[3:0]（位于 ETH_PTPPPSCR 中）将 PPS 输出的频率设置为 $2^{PPSFREQ}$Hz。

如果设置为 1 Hz， 则使用二进制翻转（ TSR=0， ETH_PTPTSCR 寄存器中的位 9）时，PPS 脉冲宽度为 125 ms；使用数字翻转 (TSR=1) 时， PPS 脉冲宽度为 100 ms。如果设置为 2 Hz 或更高频率，则使用二进制翻转时， PPS 输出的占空比为 50%。

使用数字翻转 (TSR=1) 时，建议不要使用频率为 1 Hz 以外的 PPS 输出，因为波形可能不规则（尽管其平均频率可能在任意一秒窗口期间始终正确）。

图 38-23 PPS 输出

38.3.5. 以太网功能说明：DMA 控制器操作

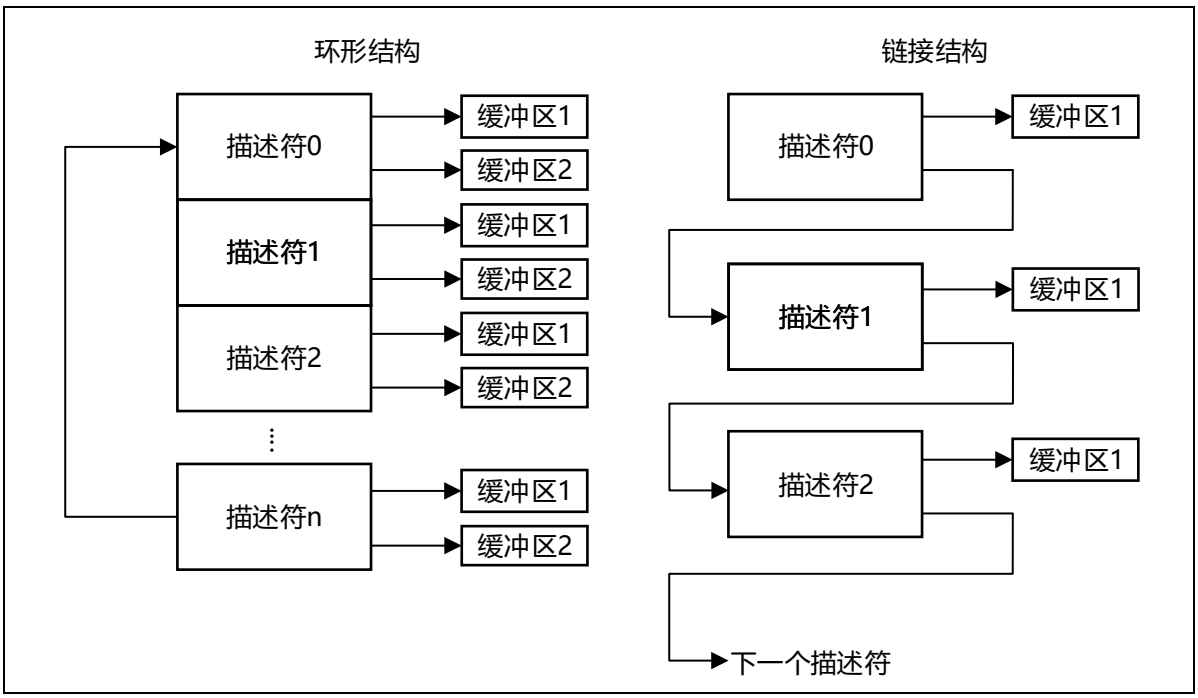
DMA 具有独立的发送和接收引擎以及相应的 CSR（控制和状态寄存器）空间。发送引擎将数据从系统存储器传送到 Tx FIFO，而接收引擎将数据从 Rx FIFO 传送到系统存储器。 DMA 可以在 CPU 完全不干预的情况下，通过描述符有效地将数据从源传送到目标。 DMA 专为面向包的数据传送（如以太网中的帧）而设计。该控制器经过编程后，可在完成帧发送和接收传送操作时以及其它正常/错误条件下产生 CPU 中断。 DMA 和应用程序通过以下两种数据结构进行通信：

- 控制和状态寄存器（CSR）
- 描述符列表和数据缓冲区。

第 1.4 节详细介绍了控制和状态寄存器。 还有相应的段落部分详细介绍了描述符。

DMA 既可将接收到的数据帧传送到存储器中的接收缓冲区，也可以传送存储器的发送缓冲区中的数据帧。位于存储器中的描述符用作指向这些缓冲区的指针。共有两个描述符列表：一个用于接收，一个用于发送。两个列表的基址分别写入 DMA 寄存器 3 和寄存器 4。描述符列表是一种前向链表（无论是隐式还是显式）。最后一个描述符会指回第一个描述符以构成环形结构。可通过配置接收和发送描述符（ RDES1[14] 和 TDES0[20]）中链接的第二个地址来完成描述符的显式链接。描述符列表位于主机的物理存储空间。每个描述符最多可指向两个缓冲区。这样便能使用两个物理寻址的缓冲区替代存储器中两个连续的缓冲区。数据缓冲区位于主机的物理存储空间，通常由整个帧或部分帧组成，但不会超过单个帧。缓冲区中仅包含数据。其状态保存在描述符中。数据链接是指跨越多个数据缓冲区的帧。但是单个描述符不能跨越多个帧。检测到帧结束时，DMA 会跳到下一个帧缓冲区。可以使能或禁止数据链接。描述符的环形结构与链接结构如图 1-24 所示。

图 38-24 描述符环形结构与链接结构



1. 使用 DMA 进行传送的初始化

MAC 的初始化步骤如下：

1）对 ETH_DMABMR 执行写操作以设置总线访问参数。

2）对 ETH_DMAIER 寄存器执行写操作以屏蔽不必要的中断源。

3）软件驱动器创建发送和接收描述符列表。随后对 ETH_DMARDLAR 和 ETH_DMATDLAR 寄存器执行写操作，为 DMA 提供各列表的起始地址。

4）对 MAC 寄存器 ETH_MACFFR，ETH_MACHTHR，ETH_MACHTLR 执行写操作以选择所需的过滤选项。

5）对 MAC ETH_MACCR 寄存器执行写操作以配置和使能发送与接收工作模式。可根据自动协商结果将 DM 位置 1（读取自 PHY）。

6）对 ETH_DMAOMR 寄存器执行写操作，将位 13 和位 1 置 1 以启动发送和接收。

7）发送与接收引擎进入运行状态，并尝试从相应描述符列表中获取描述符。这两个引擎随后开始处理接收和发送操作。发送和接收处理过程彼此独立，可单独进行启动或停止。

2. 主机总线突发访问

DMA 会尝试在 AHB 主接口上执行固定长度的突发传送（如果已使用 ETH_DMABMR 中的 FB 位进行相应配置）。突发的最大长度由 PBL 字段（ETH_DMABMR [13:8]）指示和限制。对于要读取的 16 个字节，接收和发送描述符始终采用可能的最大突发大小（由 PBL 限制）进行访问。

仅当发送 FIFO 中的空间足以容纳配置的突发或帧结束之前的字节数时（当帧短于配置的突发长度时），发送 DMA 才会启动数据传送。 DMA 会向 AHB 主接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、 INCR8、 INCR16 和单独事务的最佳搭配来传送数据。否则（非固定长度突发）会使用 INCR（未定义长度）与单独事务传送数据。

仅当接收 FIFO 中有足够的数据用于配置的突发时，或者在接收 FIFO 中检测到帧结束时（当帧短于配置的突发长度时），接收 DMA 才会启动数据传送。 DMA 会向 AHB 主接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、 INCR16 和单独事务的最佳搭配来传送数据。如果在 AHB 接口上的固定突发结束前已到达帧结束，将执行空传送以完成固定长度的突发传送。否则（复位 ETH_DMABMR 中的 FB 位）会使用 INCR（未定义长度）和单独事务传送数据。

当 AHB 接口配置为地址对齐的节拍时，两个 DMA 引擎会确保 AHB 启动的第一次突发传送小于或者等于已配置 PBL 的大小。这样，后续的所有节拍都会从与已配置 PBL 对齐的地址开始。由于 AHB 接口不支持多于 INCR16 的传送， DMA 只能对齐节拍最大为 16（对于 PBL >16）的地址。

3. 主机数据缓冲区对齐

发送和接收数据缓冲区在起始地址对齐方面没有任何限制。在我们的系统（ 32 位寻址空间）中，缓冲区的起始地址可与四个字节中的任意一个对齐。但是， DMA 始终在地址与总线宽度对齐时启动传输，并且在不需要的字节通道上传输空数据。这通常发生在以太网帧的开始或结束传送期间。软件驱动程序应根据缓冲区的起始地地址和帧的大小丢弃空数据。

● 缓冲区读操作示例：

如果发送缓冲区地址为 0x0000 0FF2，并且需要传送 15 个字节，则 DMA 将从地址 0x0000 0FF0 读取 5 个全字，但在将数据传送到发送 FIFO 时，将丢弃或忽略额外的字节（前两个字节）。同样地，还将忽略最后一次传送的最后 3 个字节。 DMA 始终可确保向发送 FIFO 传送的是全 32 位数据项，除非是帧结束。

● 缓冲区写操作示例：

如果接收缓冲区地址为 0x0000 0FF2，并且需要传送接收到的帧的 16 个字节，则 DMA 将从地址 0x0000 0FF0 开始写入 5 个全 32 位数据项。但是，第一次传送的前 2 个字节与第三次传送的最后 2 个字节将包含空数据。

4. 缓冲区大小计算

DMA 不会更新发送和接收描述符中的大小字段。DMA 只更新描述符的状态字段（xDES0)。必须用驱动程序计算大小。发送 DMA 会向 MAC 内核传送准确的字节数（由 TDES1 中的缓冲区大小字段指示)。如果将描述符标记为第一个描述符（将 TDES0 中的 FS 位置 1)，则 DMA 会将缓冲区的第一次传送标记为帧起始。如果将描述符标记为最后一个描述符（ TDES0 中的 LS 位)，则 DMA 会将数据缓冲区的最后一次传送标记为帧结束。接收 DMA 持续将数据传送至缓冲区，直到缓冲区已满或接收到帧结束为止。当描述符的 FS 位置 1 时，如果未将描述符标记为最后一个描述符（ RDES0 中的 LS 位)，则该描述符对应的缓冲区会填满，并且缓冲区中的有效数据量将通过缓冲区大小字段减去数据缓冲指针偏移得到的结果表示。当数据缓冲区指针与数据总线宽度对齐时，偏移为零。如果将描述符标记为最后一个描述符，则缓冲区不会填满（由 RDES1 中的缓冲区大小指示)。要计算该缓冲区中最终的有效数据量，驱动程序必须读取帧长度（ RDES0[29:16] 中的 FL 位)，并减去该帧中先前缓冲区的缓冲区大小之和。接收 DMA 始终使用新的描述符传送下一个帧的帧起始。

注意：即使当接收缓冲区的起始地址与系统数据总线宽度未对齐时，系统也应分配一个大小与系统总线宽度对齐的接收缓冲区。例如，如果系统分配一个起始地址为 0x1000、大小为 1024 字节（1 KB）的接收缓冲区，则软件可将接收描述符中的缓冲区起始地址编程为具有 0x1002 偏移。接收 DMA 将帧写入该缓冲区，其中前两个单元（ 0x1000 和 0x1001）中为空数据。实际帧从单元 0x1002 开始写入。因此，尽管已将缓冲区大小编程为 1024 字节，但由于存在起始偏移地址，因此该缓冲区的实际有用空间为 1022 字节。

5. DMA 仲裁器

DMA 内的仲裁器会在发送和接收通道对 AHB 主接口进行的访问之间进行仲裁。可以使用两类仲裁：循环调度和固定优先级。如果选择循环调度仲裁（复位 ETH_DMABMR 中的 DA 位)，仲裁器会在发送和接收 DMA 同时请求访问时，按照 ETH_DMABMR 中的 PM 位设置的比率分配数据总线。当 DA 位置 1 时，接收 DMA 进行数据访问的优先级始终高于发送 DMA。将 ETH_DMABMR 中的 TXPR 位置 1，则发送 DMA 的优先级会高于接收 DMA。

TXPR	PM[1]	PM[0]	DA	优先方案
0	X	X	1	接收优先级始终高于发送
0	0	0	0	发送和接收拥有相同的优先级。在请求同时产生时，接收首先获得访问权限
0	0	1	0	接收的优先级高于发送，比率为 2:1
0	1	0	0	接收的优先级高于发送，比率为 3:1
0	1	1	0	接收的优先级高于发送，比率为 4:1
1	X	X	1	发送优先级始终高于接收
1	0	0	0	发送和接收拥有相同的优先级。在请求同时产生时，发送首先获得访问权限
1	0	1	0	发送的优先级高于接收，比率为 2:1
1	1	0	0	发送的优先级高于接收，比率为 3:1
1	1	1	0	发送的优先级高于接收，比率为 4:1

6. 对 DMA 的错误响应

对于由 DMA 通道发起的任何数据传送，如果从机给出错误响应，则相应 DMA 将停止所有操作并更新状态寄存器（ ETH_DMASR 寄存器）中的错误位和致命总线错误位。此外，该 DMA 控制器只能在软复位或硬复位外设以及重新初始化 DMA 之后才能恢复操作。

7. Tx DMA 配置

■ TxDMA 操作：默认（非 OSF）模式

发送 DMA 引擎在默认模式下的操作顺序如下：

1）在为数据缓冲区设置好相应的以太网帧数据后，用户配置发送描述符（TDES0-TDES3)并将 OWN 位(TDES0[31]）置 1。

2）ST 位（ ETH_DMAOMR 寄存器 [13]）置 1 后， DMA 会立即进入运行状态。

3）在运行状态下， DMA 会为需要传送的帧轮询发送描述符列表。轮询启动后， DMA 会以连续的描述符环形顺序或链接顺序持续进行。如果 DMA 检测到标记为 CPU 所有的描述符，或者发生错误条件，则会挂起传送并将发送缓冲区不可用位（ ETH_DMASR 寄存器 [2]）与正常中断汇总使能位（ ETH_DMASR 寄存器 [16]）置 1。发送引擎继续执行步骤 9。

4）如果获得的描述符标记为 DMA 所有（TDES0[31] 置 1），则 DMA 将根据取得的描述符对发送数据缓冲区地址进行解码。

5）DMA 从存储器中获取发送数据并传送这些数据。

6）如果以太网帧保存在多个描述符的数据缓冲区中，则 DMA 将关闭中间描述符并获取下一个描述符。重复执行步骤 3、 4 和 5，直到完成以太网帧数据的传送。

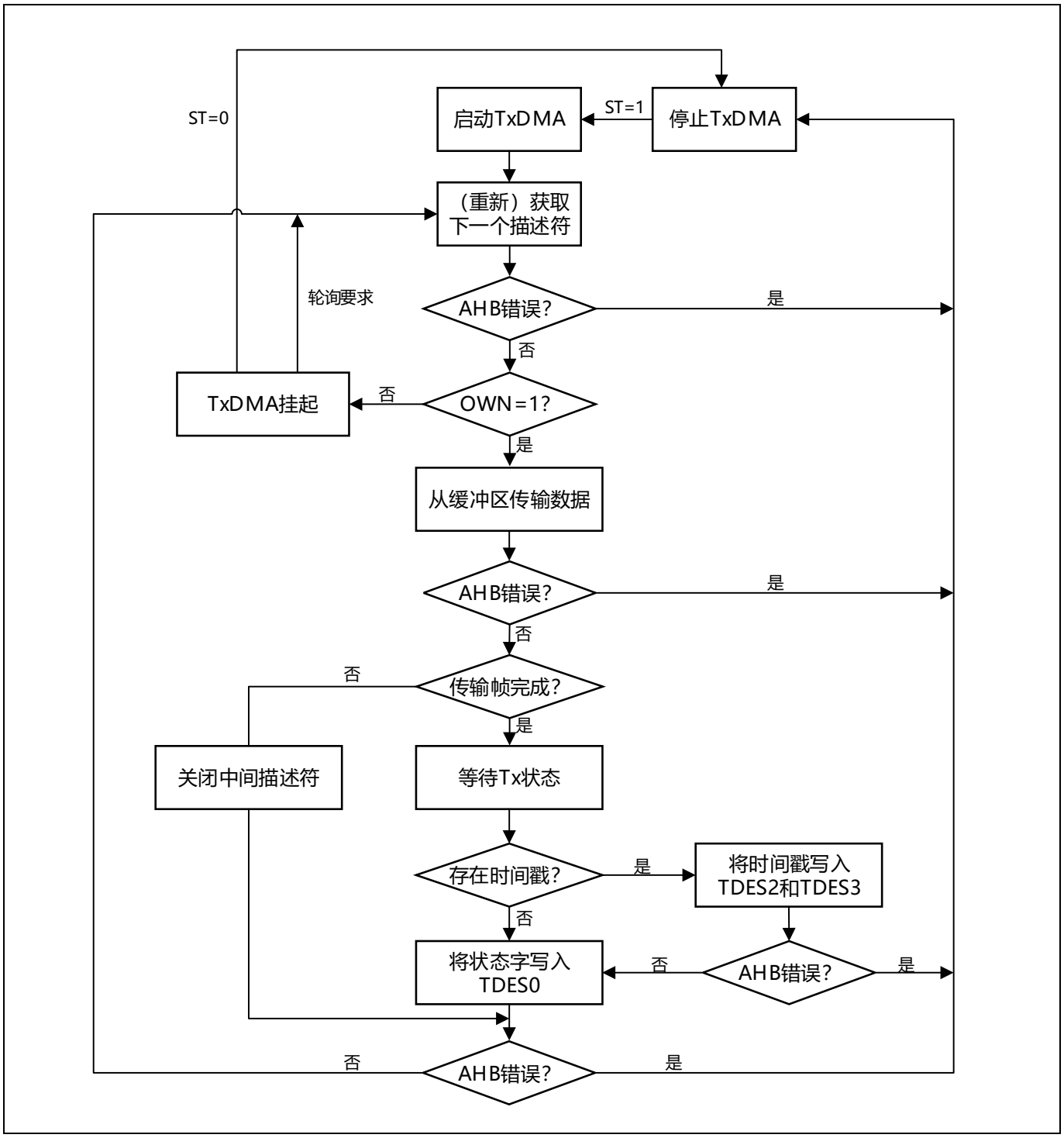
7）完成帧传送后，如果已为帧使能 IEEE 1588 时间戳（按发送状态中的指示），则时间戳值将写入包含帧结束缓冲区的发送描述符（ TDES2 和 TDES3）随后，状态信息将写入该发送描述符（TDES0)。由于此步骤中会清零 OWN 位，因此该描述符现在由 CPU 所有。如果没有为该帧使能时间戳， DMA 不会更改 TDES2 和 TDES3 的内容。

8）发送完在其最后一个描述符中将完成时中断（TDES0[30]）置 1 的帧后，发送中断（ ETH_DMASR 寄存器 [0]）将置 1。随后， DMA 引擎返回步骤 3。

9）在挂起状态下， DMA 会在接收到发送轮询要求时尝试重新获取描述符（因此返回步骤 3），并将下溢中断状态位清零。如果用户通过清除 ST 位（ ETH_DMAOMR 寄存器 [13]）来停止 DMA，DMA 会进入停止状态。

图 1-25 给出了默认模式下 TxDMA 发送过程的流程图。

图 38-25 默认模式下的 TxDMA 操作



■ TxDMA 操作：OSF 模式

在运行状态下，发送过程无需关闭第一个帧的状态描述符便可同时获取两个帧（如果 ETH_DMAOMR 寄存器中的 OSF 位置 1）。发送过程完成第一个帧的传送后，会立即轮询发第二个帧的发送描述符列表。如果第二个帧有效，发送过程会在写入第一个帧的状态信息前发送此帧。

在 OSF 模式下，运行状态发送 DMA 按照以下顺序进行操作：

1）DMA 的操作过程如默认模式下 TxDMA 的步骤 1—6 中所述。

2）无需关闭前一帧的最后一个描述符， DMA 便可获取下一个描述符。

3）如果 DMA 拥有所需描述符，便会对该描述符中的发送缓冲区地址进行解码。如果 DMA 未拥有描述符，则会进入挂起模式并跳到步骤 7。

4）DMA 从存储器中取得发送帧并发送该帧，直到帧数据发送结束，如果该帧拆分到多个描述符中，则会同时关闭中间描述符。

5）DMA 等待前一个帧的发送状态和时间戳。当状态可用时，如果捕获到时间戳（由状态位指示）， DMA 会将这些时间戳写入 TDES2 和 TDES3。之后， OWN 位清零， DMA 将状态写入相应的 TDES0，进而关闭描述符。如果没有为前一个帧使能时间戳， DMA 不会更改 TDES2 和 TDES3 的内容。

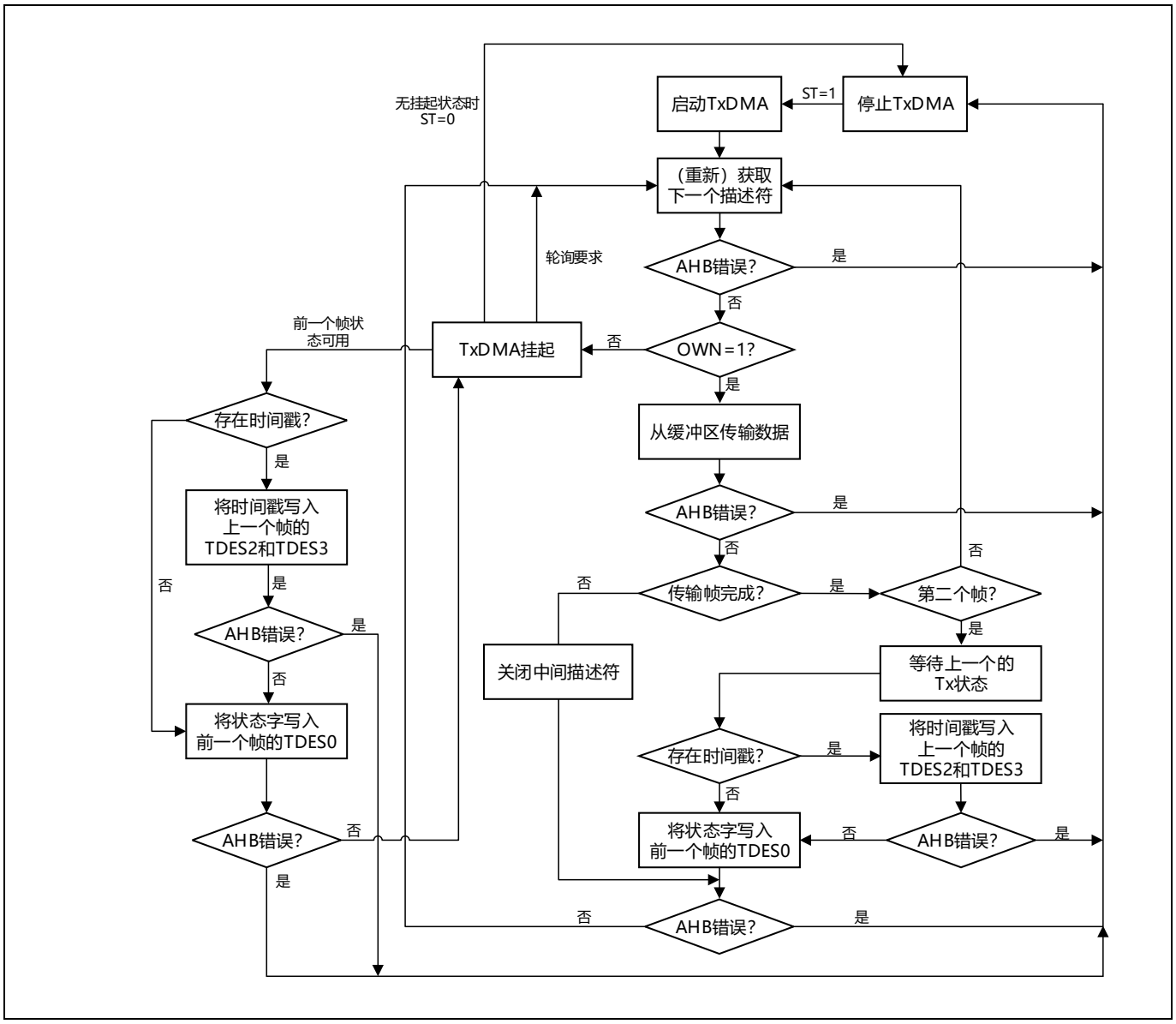
6）如果已使能时间戳，发送中断将置 1， DMA 获取下一个描述符，然后继续执行步骤 3（状态正常时）。如果上一个发送状态显示下溢错误，则 DMA 会进入挂起模式（步骤 7）。

7）在挂起模式下，如果 DMA 接收到挂起状态和时间戳，则会将时间戳（如果已为当前帧使能时间戳）写入 TDES2 和 TDES3，随后将状态写入相应的 TDES0。之后，将相关中断置 1 并返回挂起模式。如果没有挂起状态，用户通过清除 ST 位（ ETH_DMAOMR 寄存器 [13]）来停止 DMA，DMA 会进入停止状态。

8）只有在接收到发送轮询要求（ ETH_DMATPDR 寄存器）后， DMA 才会退出挂起模式并进入运行状态（转到步骤 1 或步骤 2，具体取决于挂起状态）。

图 1-26 显示了 OSF 模式下的基本流程图。

图 38-26 OSF 模式下的 TxDMA 操作



■ 发送帧处理

发送 DMA 期望数据缓冲区包含完整的以太网帧，不包括报头、填充字节和 FCS 字段。DA、 SA 和类型/长

度字段包含有效数据。如果发送描述符指示 MAC 内核必须禁止插入 CRC 或填充字节，则缓冲区必须具有包括 CRC 字节的完整以太网帧（不包括报头）。

帧可以采用数据链接形式，也可以跨越多个缓冲区。帧必须由第一个描述符（TDES0[28]）和最后一个描述符(TDES0[29]）定界。

传送启动时，必须将第一个描述符的 TDES0[28] 置 1。之后，帧数据便从存储器缓冲区传送到发送 FIFO。与此同时，如果当前帧的最后一个描述符(TDES0[29]）清零，发送过程将尝试获取下一个描述符。发送过程期望 TDES0[28] 在此描述符中清零。如果 TDES0[29] 清零，则指示中间缓冲区。如果 TDES0[29] 置 1，则指示帧的最后一个缓冲区。

当帧的最后一个缓冲区完成传送后， DMA 会将最终状态信息写回描述符的发送描述符 0 (TDES0) 字，该描述符在发送描述符 0 (TDES0[29]）中将末段置 1。此时，如果完成时中断（TDES0[30]）置 1，则发送中断（ ETH_DMASR 寄存器 [0] 中）将置 1，并获取下一个描述符，然后重复执行该过程。

实际的帧传送过程在发送 FIFO 达到可编程的发送阈值时（ ETH_DMAOMR 寄存器 [16:14]），或者 FIFO 中包含完整的帧时才会启动。此外，还可以选择存储转发模式（ ETH_DMAOMR 寄存器 [21]）。 DMA 完成帧的传送后会释放描述符（清零 OWN 位 TDES0[31]）。

■ 发送轮询挂起

可通过以下任意条件暂停发送轮询：

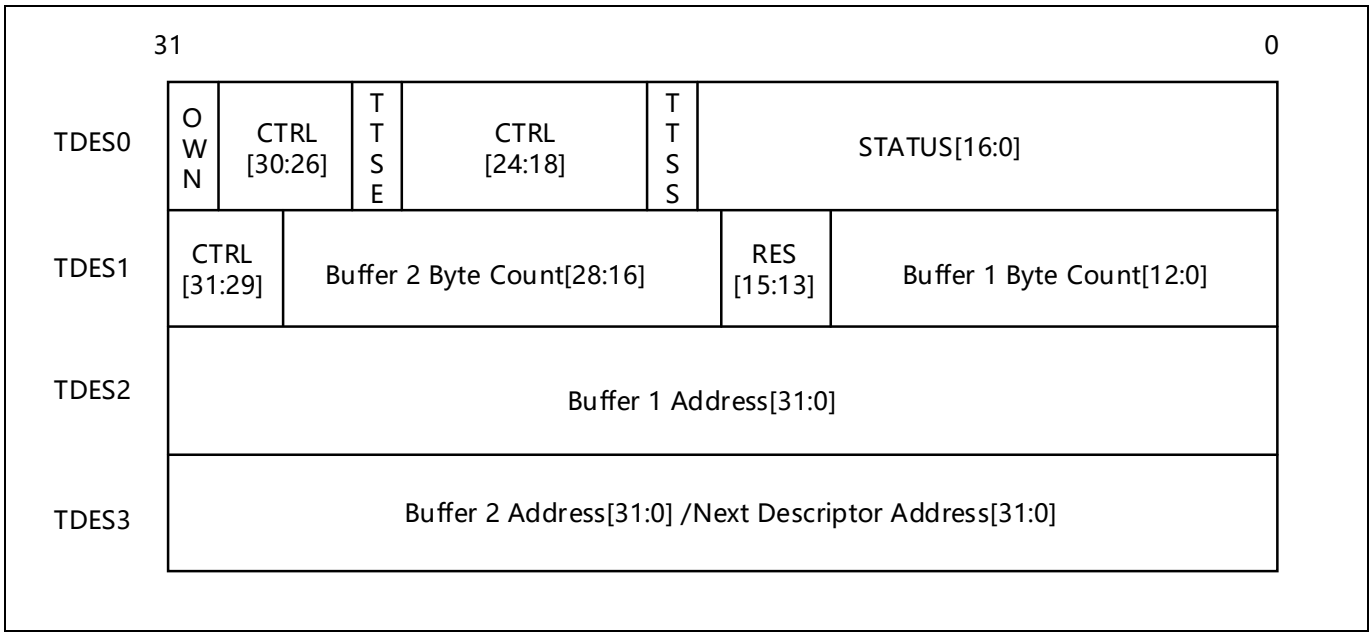
- DMA 检测到 CPU 所有的描述符 (TDES0[31]=0)，并且发送缓冲区不可用标志置 1（ ETH_DMASR 寄存器 [2])。如要恢复，驱动程序必须将描述符的所有权交给 DMA，然后发出轮询查询要求命令。
- 检测到由下溢导致的传送错误时，将中止帧的传送。相应的发送描述符 0 (TDES0) 位将置 1。
- 如果发生第二个条件，异常中断汇总位（ ETH_DMASR 寄存器 [15] 中）与发送下溢位（ ETH_DMASR 寄存器 [5] 中）都将置 1，并且信息将写入发送描述符 0，从而导致挂起。如果 DMA 由于第一个条件而进入挂起状态，则正常中断汇总位（ ETH_DMASR 寄存器 [16]）与发送缓冲区不可用位（ ETH_DMASR 寄存器 [2]）均置 1。
- 两种情况下，发送列表中的位置都会保留。保留的位置是 DMA 关闭的最后一个描述符后面的描述符位置。纠正挂起原因后，驱动程序必须明确发出发送轮询要求命令。

■ 常规 Tx DMA 描述符

常规发送描述符结构由四个 32 位字组成，如图 1-28 所示。下文给出了 TDES0、 TDES1、TDES2 和 TDES3 的位说明。

请注意，如果时间戳已激活（ ETH_PTPTSCR 位 0， TE=1）或 IPv4 校验和减荷已激活（ ETH_MACCR 位 10， IPCO=1），则必须使用增强的描述符。

图 38-27 常规发送描述符



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OWN	IC	LS	FS	DC	DP	TTSE	CRCR	CIC		TER	TCH	VIC		TTSS	IHE	ES	JT	FF	IPE	LCA	NC	LCO	EC	VF					ED	UF	DB
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- TDES0：发送描述符字 0

应用软件必须在描述符初始化期间编程控制位 [30:18] 以及 OWN 位 [31]。当 DMA 更新描述符（或将其写回）时，将复位所有控制位与 OWN 位，并且只报告状态信息。

位域	名称
31	OWN： 所有位 (OWN BIT)该位置 1 时，表示描述符为 DMA 所有。该位复位时，表示描述符为 CPU 所有。 DMA 可在完成帧发送时或在描述符分配的缓冲区全部读取完成后将该位清零。将属于同一个帧的所有后续描述符的所有位置 1 后，必须将帧的第一个描述符的所有位置 1。这避免了获取描述符合驱动程序设置所有为位之间可能存在的竞争。
30	IC： 完成时中断 (INTERRUPT ON COMPLETION)该位置 1 时，当前帧发送完毕后，发送中断位（ETH_DMASR 寄存器位 0）置 1。仅当设置了 LS 位（TDES0[29]）时，此位才有效。
29	LS： 末段 (LAST SEGMENT)该位置 1 时，指示缓冲区中包含帧的末段。
28	FS： 首段 (FIRST SEGMENT)该位置 1 时，指示缓冲区中包含帧的首段。
27	DC： 禁止 CRC (DISABLE CRC)该位置 1 时， MAC 不会将循环冗余校验 (CRC) 附加到所发送帧的末尾。该位仅在首段位(TDES0[28]) 置 1 的情况下有效。
26	DP： 禁止 PAD (DISABLE PAD)该位置 1 时， MAC 不会自动为不足 64 字节的帧添加补位项。该位复位时， DMA 会自动为不足 64 字节的帧添加补位项和 CRC，是否添加 CRC 字段与 DC (TDES0[27]) 位状态无关。该位仅在首段位 (TDES0[28]) 置 1 的情况下有效。
25	TTSE： 发送时间戳使能 (TRANSMIT TIMESTAMP ENABLE)TTSE 位置 1 且 TSE 置 1（ ETH_PTPTSCR 位为 0）时，将针对描述符所描述的发送帧激活 IEEE1588 硬件时间戳功能。该字段仅在首段控制位 (TDES0[28]) 置 1 的情况下有效。

24	CRCR： CRCR 替换控制（CRC REPLACEMENT CONTROL）该位置 1 时，MAC 会用重新计算的 CRC 字节替换传输数据帧的最后 4 个字节。应用应确保 CRC 字节存在于从传输缓冲区传输的帧中。
23:22	CIC： 校验和插入控制（CHECKSUM INSERTION CONTROL)这些位控制校验和的计算与插入。位编码如下所示：00：禁止插入校验和 01：仅使能 IP 报头校验和的计算与插入 10：使能 IP 报头校验和以及有效负载校验和的计算与插入，但不会在硬件中计算伪报头校验和 11：使能 IP 报头校验和以及有效负载校验和的计算与插入，并在硬件中计算伪报头校验和。
21	TER： 环发送结束（TRANSMIT END OF RING)该位置 1 时，表示描述符列表已到达其最后一个描述符。 DMA 会返回描述符列表的基址，并形成一个描述符环。
20	TCH： 链接的第二个地址（SECOND ADDRESS CHAINED)该位置 1 时，表示描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址。TDES0[20] 置 1 时， TBS2 (TDES1[28:16]) 为“无关”值。 TDES0[21] 优先级高于 TDES0[20]。
19:18	VIC： VLAN 插入控制（VLAN INSERTION CONTROL）这些位请求 MAC 在发送数据帧之前执行 VLAN 标记或取消标记操作。如果数据帧针对 VLAN 标记做了修改，MAC 会重新计算并替换 CRC 字节。位编码如下所示：00：不添加 VLAN 标记 01：在发送前从数据帧中删除 VLAN 标记，此选项应只与 VLAN 数据帧一起使用 10：使用在 VALN 标记包含或替换寄存器中编程的标记值插入 VLAN 标记 11：使用在 VALN 标记包含或替换寄存器中编程的标记值替换数据帧中的 VLAN 标记，此选项应只与 VLAN 数据包一起使用
17	TTSS： 发送时间戳状态（TRANSMIT TIMESTAMP STATUS)此字段用作状态位，指示已为所描述的发送帧捕获到时间戳。该位置 1 时， TDES2 和 TDES3 将保存为发送帧捕获到的时间戳值。该字段仅在描述符末段控制位（TDES0[29]）置 1 的情况下有效。注意，使能增强的描述符时（ ETH_DMABMR 中的 EDFE=1）， TTSS=1 指示 TDES6 和 TDES7 中存有时间戳值。
16	IHE： IP 报头错误（IP HEADER ERROR)该位置 1 时，指示 MAC 发送器在 IP 数据报头中检测到错误。发送器将对照从应用程序接收的报头字节数检查 IPV4 数据包的报头长度，如果出现不匹配，则指示错误状态。对于 IPV6 帧，如果主报头长度不是 40 个字节，则报告一个报头错误。此外， IPV4 或 IPV6 帧的以太网长度/类型字段值必须与随数据包接收的 IP 报头版本匹配。对于 IPV4 帧，如果报头长度字段的值小于 0X5，也会指示一个错误状态。
15	ES： 错误汇总（ERROR SUMMARY)指示以下位的逻辑或运算结果：TDES0[14]：JABBER 超时(JABBER TIMEOUT)TDES0[13]：帧刷新（FRAME FLUSH)TDES0[11]：载波丢失（LOSS OF CARRIER)TDES0[10]：无载波（NO CARRIER)TDES0[9]：延迟冲突（LATE COLLISION)TDES0[8]：过度冲突（EXCESSIVE COLLISION)TDES0[2]：过度延迟（EXCESSIVE DEFERRAL)TDES0[1]：下溢错误（UNDERFLOW ERROR)TDES0[16]： IP 报头错误（IP HEADER ERROR)TDES0[12]： IP 有效负载错误（IP PAYLOAD ERROR)
14	JT： JABBER 超时（JABBER TIMEOUT)该位置 1 时，指示 MAC 发送器经历了 JABBER 超时。只有在 MAC 配置寄存器的 JD 位未置 1 时，该位才会置 1。
13	FF： 帧已刷新（FRAME FLUSHED)该位置 1 时，指示 DMA/MTL 已依照 CPU 发出的软件刷新命令刷新帧。
12	IPE： IP 有效负载错误（IP PAYLOAD ERROR)该位置 1 时，指示 MAC 发送器在 TCP、 UDP 或 ICMP IP 数据报有效负载中检测到错误。发送器将对照从应用程序接收的 TCP、 UDP 或 ICMP IP 数据包实际字节数检查 IPV4 或 IPV6 报头中接收的有效负载长度，如果出现不匹配，则指示错误状态。
11	LCA： 载波丢失（LOSS OF CARRIER)该位置 1 时，指示帧发送期间丢失载波（即，帧发送期间有一个或多个发送时钟周期的 MII_CRS 信号无效）。该位仅对在 MAC 处于半双工模式时实现无冲突发送的帧有效。
10	NC： 无载波（NO CARRIER)该位置 1 时，指示在发送期间未由 PHY 触发载波侦听信号。
9	LCO： 延迟冲突（LATE COLLISION)该位置 1 时，指示帧发送过程因冲突窗口（ MII 模式下为 64 个字节时间，含报头）后出现冲突而中止。如果下溢错误位置 1，则该位无效。

8	EC： 过度冲突 (EXCESSIVE COLLISION)该位置 1 时，指示尝试发送当前帧时因出现 16 个连续冲突而中止发送。如果 MAC 配置寄存器中的 RD（禁止重试）位置 1，则此位会在出现首个冲突后置 1 并且帧发送过程将中止。
7	VF： VLAN 帧 (VLAN FRAME)该位置 1 时，指示所发送帧为 VLAN 类型的帧。
6:3	CC 冲突计数 (COLLISION COUNT)此 4 位计数器值指示发送帧之前出现的冲突个数。过度冲突位 (TDES0[8]) 置 1 时，该计数无效。
2	ED 过度延迟 (EXCESSIVE DEFERRAL)该位置 1 时，如果 MAC 控制寄存器中的延迟检查 (DC) 位为高电平，则此位指示因出现超过 24288 个位时间的过度延迟而中止发送。
1	UF： 下溢错误 (UNDERFLOW ERROR)该位置 1 时，指示 MAC 因 RAM 存储器的数据未及时到达而中止了帧发送。下溢错误表示 DMA 在帧发送期间遇到发送缓冲区为空的情况。发送过程将进入挂起状态并将发送下溢（寄存器 5[5]）和发送中断（寄存器 5[0]）位置 1。
0	DB： 延迟位 (DEFERRED BIT)该位置 1 时，指示 MAC 在发送前因存在载波而延迟。该位仅在半双工模式下有效。

● TDES1：发送描述符字 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAIC			TBS2													RESERVED			TBS1												
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:29	SAIC： SA 插入控制（SA INSERTION CONTROL）这些位请求 MAC 用 MAC 地址 0 寄存器中给出的值添加或替换以太网数据帧中的源地址字段。如果在帧中修改了源地址字段，MAC 会自动重新计算并替换 CRC 字节。位 31 指定用于源地址插入或替换的 MAC 地址寄存器（1 或 0）值。下面对位[30:29]的值进行了说明：00：不包含源地址 01：包含或插入源地址。为实现可靠发送，应用程序必须提供不包含源地址的帧 10：替换源地址。为实现可靠发送，应用程序必须提供包含源地址的帧 11：保留 这些位在首段控制位（TDES0[28]）置 1 的情况下有效。
28:16	TBS2： 发送缓冲区 2 大小 (TRANSMIT BUFFER 2 SIZE)这些位以字节为单位指示第二个数据缓冲区的大小。如果 TDES0[20] 位置 1，则该字段无效
15:13	保留
12:0	TBS1： 发送缓冲区 1 大小 (TRANSMIT BUFFER 1 SIZE)这些位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0， DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 TCH (TDES0[20]) 的值。

● TDES2：发送描述符字 2

TDES2 包含描述符第一个缓冲区的地址指针。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBAP1																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	TBAP1： 发送缓冲区 1 地址指针(TRANSMIT BUFFER 1 ADDRESS POINTER)。这些位将指示缓冲区 1 的物理地址。缓冲区地址对齐方式不限。有关缓冲区地址对齐方式的详细信息，请参见第 40 页的主机数据缓冲区对齐。

● TDES3：发送描述符字 3

TDES3 包含描述符中第二个缓冲区或下一个描述符的地址指针。

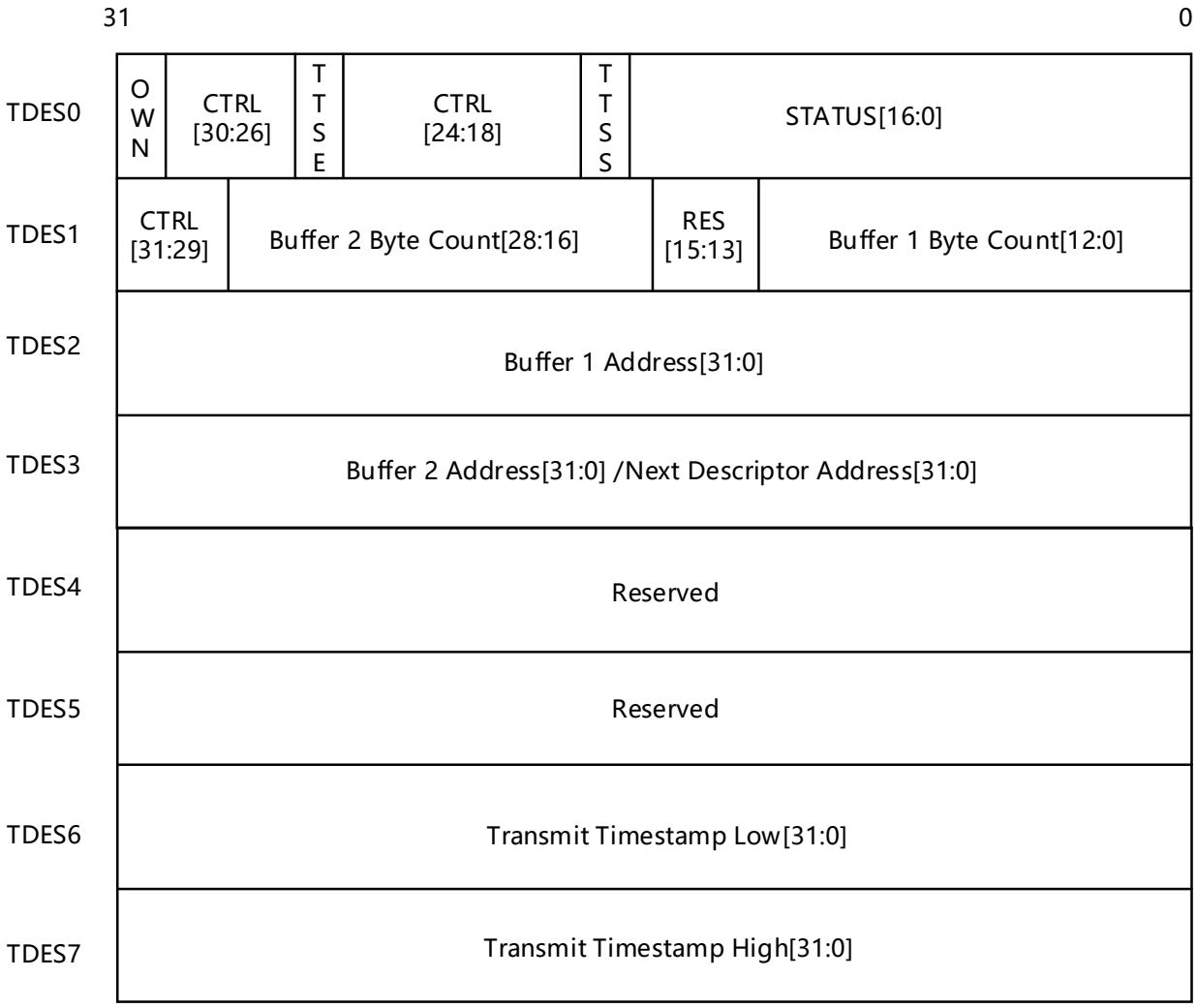
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBAP2 / NDA																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	TBAP2：发送缓冲区 2 地址指针（下一个描述符地址） (TRANSMIT BUFFER 2 ADDRESS POINTER (NEXT DESCRIPTOR ADDRESS))使用描述符环结构时，这些位将指示缓冲区 2 的物理地址。如果链接第二个地址位 (TDES0 [20]) 置 1，则该地址包含下一个描述符所在物理寄存器的指针。只有在 TDES0 [20] 位置 1 时，缓冲区地址指针才必须与总线宽度相符。(将在内部忽略 LSB)

■ 增强 Tx DMA 描述符

如果时间戳已激活（TE=1， ETH_PTPTSCR 位 0）或 IPv4 校验和减荷已激活（IPCO=1，ETH_MACCR 位 10），则必须使用增强的描述符（通过 ETHDMABMR 位 7 EDFE=1 使能）。增强的描述符由 8 个 32 位字组成，即描述符正常大小的两倍。 TDES0、 TDES1、 TDES2 和 TDES3 的定义与常规发送描述符相同（请参见常规 Tx DMA 描述符）。 TDES6 和 TDES7 包含时间戳。 TDES4、 TDES5、 TDES6 和 TDES7 的定义如下。选择增强的描述符模式时，需要利用软件为每个描述符分配 32 个字节（ 8 个双字）的内存。如果未使用时间戳或 IPv4 校验和减荷，则可以禁止增强的描述符格式，而软件可使用默认大小为 16 字节的常规描述符。

图 38-28 增强的发送描述符



● TDES4：发送描述符字 4

保留

● TDES5：发送描述符字 5

保留

● TDES6：发送描述符字 6

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTSL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	TTSL： 发送帧时间戳低位 (TRANSMIT FRAME TIMESTAMP LOW)DMA 使用为相应发送帧所捕获的时间戳的 32 个最低有效位更新该字段。只有在描述符的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

● TDES7：发送描述符字 7

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTSH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	TTSH： 发送帧时间戳高位 (TRANSMIT FRAME TIMESTAMP HIGH)DMA 使用为相应发送帧所捕获的时间戳的 32 个最高有效位更新该字段。只有在描述符的末段控制位 (LS) 置 1 时，该字段才包含时间戳。

8. Rx DMA 配置

图 1-29 所示为接收 DMA 引擎的接收序列，具体说明如下：

1）CPU 设置接收描述符 (RDES0-RDES3) 并将 OWN 位 (RDES0[31]) 置 1。

2）SR 位（ ETH_DMAOMR 寄存器 [1]）置 1 后， DMA 即进入运行状态。在运行状态下，DMA 轮询接收描述符列表，尝试获取空闲描述符。如果获得的描述符不空闲（由 CPU 所拥有），则 DMA 进入挂起状态并跳转到第 9 步。

3）DMA 将所获取描述符中的接收数据缓冲区地址解码。

4）处理传入帧并将其放入所获取描述符的数据缓冲区。

5）当缓冲区已满或帧传输完成时，接收引擎将获取下一个描述符。

6）如果当前的帧传输完成， DMA 将继续执行第 7 步。如果 DMA 未拥有下一个获取的描述符且帧传输尚未完成（尚未传输 EOF），则 DMA 会将 RDES0 中的描述符错误位置 1（除非禁止刷新）。 DMA 关闭当前描述符（将 OWN 位清零）并通过在 RDES1 值中将末段位 (LS) 清零来将其标记为中间描述符（如果禁止刷新，则将其标记为最后一个描述符），随后继续执行第 8 步。如果 DMA 已拥有下一个描述符，但当前的帧传输尚未完成，则 DMA 将关闭当前描述符作为中间值并返回第 4 步。

7）如果已使能 IEEE 1588 时间戳功能， DMA 会将时间戳（如果可用）写入当前描述符的 RDES2 和 RDES3。 DMA 随后获取所接收帧的状态并将该状态字写入当前描述符的 RDES0，同时 OWN 位清零且末段位置 1。

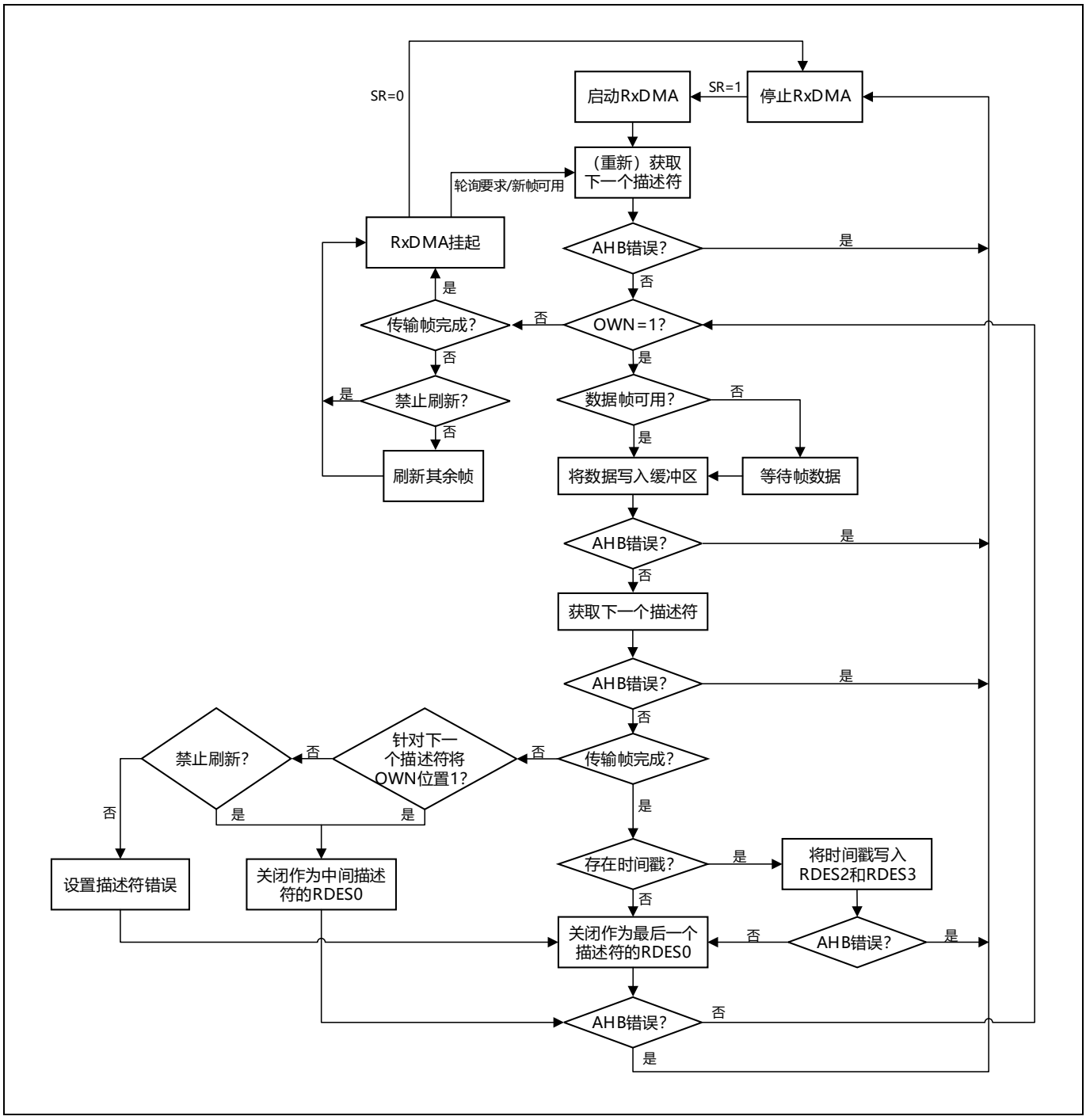
8）接收引擎检查最新描述符的 OWN 位。如果 CPU 拥有该描述符（ OWN 位为 0），则接收缓冲区不可用位（ ETH_DMASR 寄存器 [7]）置 1 且 DMA 接收引擎进入挂起状态（第 9 步）。如果 DMA 拥有该描述符，引擎将返回第 4 步并等待下一个帧。

9）在接收引擎进入挂起状态前，将从接收 FIFO 中刷新部分帧（可使用 ETH_DMAOMR 寄存器的位 24 控制刷新）。

10）如果用户清除了 ETH_DMAOMR 寄存器的 SR 位，接收 DMA 会进入停止状态。否则当接收 DMA 收到接收轮询要求命令或者可以从接收 FIFO 获得下一个帧的起点时，接收 DMA 将退出挂起状态。引擎继续执行第 2 步并重新获取下一个描述符。

直到完成时间戳回写并准备将状态回写到描述符时， DMA 才会确认接收状态。如果软件已通过 CSR 使能时间戳功能，则当无法获得帧的有效时间戳值时（例如，在写入时间戳前接收 FIFO 已满）， DMA 会向 RDES2 和 RDES3 的所有位写入 1。否则（即，未使能时间戳功能）， RDES2 和 RDES3 保持不变。

图 38-29 DMA 接收操作



■ 接收描述符获取

接收引擎始终会尝试为即将传入的帧获取一个额外的描述符。只要满足以下任一条件，即尝试获取描述符：

- DMA 进入运行状态后，接收启动/停止位（ ETH_DMAOMR 寄存器 [1]）立即置 1。
- 在当前传输的帧结束前，当前描述符的数据缓冲区已满。
- 控制器已完成数据帧接收，但当前的接收描述符尚未关闭。
- 接收过程因缓冲区由 CPU 所拥有（RDES0[31] = 0）而挂起，并且接收到新帧。
- 已发出接收轮询要求命令。

■ 接收帧处理

只有当帧通过地址过滤且其大小大于或等于为接收 FIFO 所设置的可配置阈值字节数时，或者在存储转发模式下将整个帧写入 FIFO 时， MAC 才会将接收的帧传输到存储器。如果帧未通过地址过滤，将自行进入 MAC 块中（除非接收所有位 ETH_MACFFR [31]置 1）。不足 64 字节的帧由于会发生冲突或过早结束，因而可从接收 FIFO 中清除。接收 64（可配置阈值）个字节后， DMA 块开始将帧数据传输到当前描述符所指定的接收缓冲区中。 DMA AHB 接口准备好接收数据传输后，如果 DMA 当前未从存储器获取发送数据，则将第一个描述符位 (RDES0[9]) 置 1，以分隔该帧。数据缓冲区已满或者帧的末段已传输到接收缓冲区时， OWN (RDES0[31]) 位将复位为 0，此时将释放描述符。如果帧包含在单个描述符中，则最后一个描述符位 (RDES0[8]) 和第一个描述符位 (RDES0[9]) 均置 1。 DMA 获取下一个描述符，将最后一个描述符位 (RDES0[8]) 置 1，并释放前一个帧描述符中的 RDES0 状态位。然后， DMA 将接收中断位（ ETH_DMASR 寄存器 [6]）置 1。这一过程将重复执行，直至 DMA 遇到被标记为由 CPU 所有的描述符。如果出现这种情况，接收过程将接收缓冲区不可用位（ ETH_DMASR 寄存器 [7]）置 1，随后进入挂起状态。但其在接收列表中的位置仍然保留。

■ 接收过程挂起

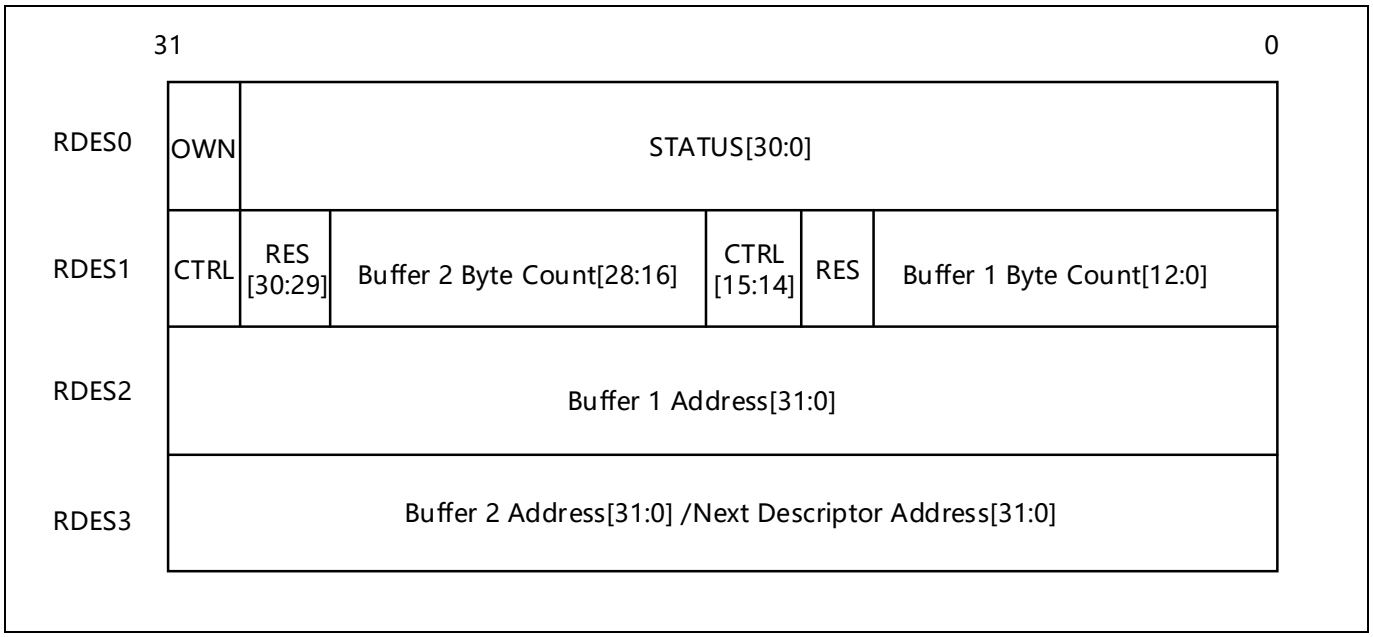
如果在接收过程处于挂起状态时有新的接收帧到达，则 DMA 将重新获取存储器中的当前描述符。如果该描述符现在由 DMA 所拥有，接收过程将重新进入运行状态并开始接收帧。如果该描述符仍由主机所拥有，则默认情况下， DMA 将丢弃 Rx FIFO 顶部的当前帧并将丢失帧计数器递增。如果 Rx FIFO 中存储了多个帧，将重复执行该过程。将 DMA 工作模式寄存器的位 24 (DFRF) 置 1 后，可避免丢弃或刷新 Rx FIFO 顶部的帧。在这种情况下，接收过程将接收缓冲区不可用状态位置 1 并返回到挂起状态。

■ 常规 Rx DMA 描述符

常规接收描述符结构由四个 32 位字（16 字节）组成，如图 1-30 所示。下文将介绍 RDES0、RDES1、RDES2 和 RDES3 的各个位。

请注意，如果已激活时间戳功能（ ETH_PTPTSCR 位 0 TSE=1）或 IPv4 校验和减荷（ ETH_MACCR 位 10 IPCO=1），则必须使用增强的描述符。

图 38-30 常规 RxDMA 描述符结构



● RDES0：接收描述符字 0

REDS0 包含接收帧状态、帧长度和描述符所有关系信息。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OWN	AFM	FL														ES	DE	SAF	LE	OE	VLAN	FS	LS	IPHCE/TSV	LCO	FT	RWT	RE	DE	CE	PCE/ESA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31	OWN： 所有关系位 (OWN BIT)该位置 1 时，指示描述符由 MAC 子系统的 DMA 所拥有。该位复位时，指示描述符由主机所拥有。 DMA 在帧接收完成或此描述符的关联缓冲区已满时将该位清零。
30	AFM： 目标地址过滤失败 (DESTINATION ADDRESS FILTER FAIL)该位置 1 时，指示帧未能通过 MAC 内核中的 DA 过滤。
29:16	FL： 帧长度 (FRAME LENGTH)这些位指示传输到主机存储器中的接收帧字节长度（含 CRC）。只有在最后一个描述符位(RDES0[8]) 置 1 且描述符错误位 (RDES0[14]) 复位时，该字段才有效。在最后一个描述符位 (RDES0[8]) 置 1 时，该字段有效。当最后一个描述符位和错误汇总位均未置 1 时，该字段指示已为当前帧传输的累计字节数。
15	ES： 错误汇总 (ERROR SUMMARY)指示以下位的逻辑或运算结果：RDES0[0]：有效负载校验和错误 (PAYLOAD CHECKSUM ERROR)RDES0[1]：CRC 错误 (CRC ERROR)RDES0[3]：接收错误 (RECEIVE ERROR)RDES0[4]：看门狗超时 (WATCHDOG TIMEOUT)RDES0[6]：延迟冲突 (LATE COLLISION)RDES0[7]：巨帧 (GIANT FRAME)（当 RDES0[7] 指示 IPV4 报头校验和错误时，该位不适用。）RDES0[11]：上溢错误 (OVERFLOW ERROR)RDES0[14]：描述符错误 (DESCRIPTOR ERROR)。只有最后一个描述符 (RDES0[8]) 置 1 时，该字段才有效。
14	DE： 描述符错误 (DESCRIPTOR ERROR)该位置 1 时，指示某个帧因超过当前描述符缓冲区的大小而被截断以及 DMA 未拥有下一个描述符。帧被截断。只有最后一个描述符 (RDES0[8]) 置 1 时，该字段才有效。
13	SAF： 源地址过滤失败 (SOURCE ADDRESS FILTER FAIL)该位置 1 时，指示帧的 SA 字段未能通过 MAC 内核中的 SA 过滤。
12	LE 长度错误 (LENGTH ERROR)该位置 1 时，指示接收帧的实际长度与长度/ 类型字段的值不符。该字段仅在帧类型位(RDES0[5]) 复位后有效。
11	OE： 上溢错误 (OVERFLOW ERROR)该位置 1 时，指示接收帧因缓冲区上溢而损坏。
10	VLAN： VLAN 标记 (VLAN TAG)该位置 1 时，指示描述符所指向的帧是由 MAC 内核标记的 VLAN 帧。
9	FS： 第一个描述符 (FIRST DESCRIPTOR)该位置 1 时，指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0，则第二个缓冲区将包含帧的帧头。如果第二个缓冲区的大小为 0，则下一个描述符将包含帧的帧头。
8	LS： 最后一个描述符 (LAST DESCRIPTOR)该位置 1 时，指示此描述符指向的缓冲区为帧的最后一个缓冲区。
7	IPHCE/TSV： IPV 报头校验和错误/时间戳有效 (IPV HEADER CHECKSUM ERROR/TIMESTAMP VALID)IPHCE 位置 1 时，指示 IPV4 或 IPV6 报头中存在错误。导致此错误的原因可能是：以太网类型字段与 IP 报头版本字段值不一致，或者与 IPV4 中报头的校验和不匹配，或者以太网帧缺少所需的 IP 报头字节数。如表 168 中所指出的那样，该位可具有特殊的含义。使能增强的描述符格式（ EDFE=1， ETH_DMABMR 的位 7）后，该位具有 TSV 功能（否则其为 IPHCE）。 TSV 位置 1 时，表示将在描述符字 6 (RDES6) 和 7 (RDES7) 中写入时间戳快照。只有最后一个描述符位 (RDES0[8]) 置 1 时， TSV 才有效。
6	LCO： 延迟冲突 (LATE COLLISION)该位置 1 时，表示在以半双工模式接收帧时发生了延迟冲突。

5	FT： 帧类型 (FRAME TYPE)该位置 1 时，表示接收帧为以太网类型帧 （ LT 字段大于或等于 0X0600）。该位复位时，表示接收的帧为 IEEE802.3 帧。该位不适用于少于 14 个字节的矮帧。使用正常描述符格式(ETH_DMABMR EDFE=0) 时， FT 可具有特殊的含义，如表 168 中所指出的那样。
4	RWT： 接收看门狗超时 (RECEIVE WATCHDOG TIMEOUT)该位置 1 时，表示接收看门狗计时器在接收当前帧时超时，且当前帧在看门狗超时后被截断了
3	RE： 接收错误 (RECEIVE ERROR)该位置 1 时，表示在帧接收期间，当发出 RX_DV 信号时，会发出 RX_ERR 信号。
2	DBE： DRIBBLE 位错误 (DRIBBLE BIT ERROR)该位置 1 时，表示接收的帧具有非整数倍数的字节（奇数半字节）。该位仅在 MII 模式下有效。
1	CE： CRC 错误 (CRC ERROR)该位置 1 时，表示接收的帧发生循环冗余校验 (CRC) 错误。只有最后一个描述符 (RDES0[8])置 1 时，该字段才有效。
0	PCE/ESA：有效负载校验和错误(PAYLOAD CHECKSUM ERROR)/扩展状态可用(EXTENDED STATE AVAILABLE)该位置 1 时，表示由内核计算的 TCP、 UDP 或 ICMP 校验和与接收到的封装 TCP、 UDP 或 ICMP 段校验和字段不匹配。当接收到的有效负载字节数与接收到的以太网帧中封装 IPV4 或 IPV6 数据图的长度字段中所指示的值不匹配时，该位也会被置 1。如表 168 中所指出的那样，该位可具有特殊的含义。使能增强的描述符格式 （ EDFE=1， ETH_DMABMR 的位 7）后，该位具有 ESA 功能（否则其为 PCE）。 ESA 置 1 时，表示描述符字 4 (RDES4) 中存在扩展状态。只有最后一个描述符位 (RDES0[8]) 置 1 时， ESA 才有效。

位 5、 7 和 0 用于指示表 1-8 中讨论的情况。

表 38-8 接收描述符 0-位 5、7 和 0 的编码(仅正常描述符格式,EDFE=0)

位 5：帧类型 (Frame type)	位 7：IPC 校验和错误 (IPC checksum error)	位 0：有效负载校验和错误 (payload checksum error)	帧状态
0	0	0	IEEE 802.3 类型帧（长度字段小于 0x0600）
1	0	0	IPv4/IPv6 类型帧，未检测到校验和错误
1	0	1	IPv4/IPv6 类型帧，检测到有效负载校验和错误（请参见 PCE 的相关说明）
1	1	0	IPv4/IPv6 类型帧，检测到 IP 报头校验和错误（请参见 IPCCE 的相关说明）
1	1	1	IPv4/IPv6 类型帧，同时检测到 IP 报头和有效负载校验和错误
0	0	1	IPv4/IPv6 类型帧，无 IP 报头校验和错误，有效负载检查因有效负载不受支持而被绕过
0	1	1	既不是 IPv4 也不是 IPv6 的类型帧（校验和减荷引擎完全绕过校验和）
0	1	0	保留

- RDES1：接收描述符字 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIC	RES		RBS2													RER	RCH	RES	RBS1												
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31	DIC:完成时禁止中断 (DISABLE INTERRUPT ON COMPLETION)置 1 时，该位会阻止状态寄存器的 RS 位 (CSR5[6]) 置 1，使得接收的帧在由该描述符所指示的缓冲区中结束，从而会因该帧的 RS 而禁止引发主机的中断。
30:29	保留
28:16	RBS2： 接收缓冲区 2 大小 (RECEIVE BUFFER 2 SIZE)这些位以字节为单位指示第二个数据缓冲区的大小。即使 RDES3（缓冲区 2 地址指针）的值未与总线宽度对齐，缓冲区大小也必须为 4、 8 或 16 的倍数，具体取决于总线宽度（分别为 32、 64 或 128）。如果缓冲区大小不是 4、 8 或 16 的倍数，则不会定义产生的行为。如果将 RDES1[14] 置 1，则该字段无效。
15	RER： 接收环结束 (RECEIVE END OF RING)该位置 1 时，表示描述符列表已到达其最后一个描述符。 DMA 会返回描述符列表的基址，并形成一个描述符环。
14	RCH： 链接的第二个地址 (SECOND ADDRESS CHAINED)该位置 1 时，表示描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址。该位置 1 时， RBS2 (RDES1[28:16]) 为“无关”值。 RDES1[15] 优先级高于 RDES1[14]。
13	保留
12:0	RBS1： 接收缓冲区 1 大小 (RECEIVE BUFFER 1 SIZE)以字节为单位表示第一个数据缓冲区的大小。即使 RDES2（缓冲区 1 地址指针）的值未对齐，缓冲区大小也必须为 4、 8 或 16 的倍数，具体取决于总线宽度（ 32、 64 或 128）。如果缓冲区大小不是 4、 8 或 16 的倍数，则不会定义产生的行为。如果该字段为 0，则 DMA 会忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 RCH（位 14）的值

● RDES2：接收描述符字 2

RDES2 包含描述符中第一个数据缓冲区的地址指针。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBAP1																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	RBAP1：接收缓冲区 1 地址指针 (RECEIVE BUFFER 1 ADDRESS POINTER)这些位会指示缓冲区 1 的物理地址。在缓冲区地址对齐方面没有任何限制，但以下情况除外：当使用 RDES2 值存储帧起点时， DMA 会使用所配置的值生成其地址。请注意，在传输帧开始期间， DMA 将在 RDES2[3/2/1:0] 位为 0 的情况下执行写操作，但帧数据会根据实际的缓冲区地址指针进行移位。当地址指针指向存储有帧中间部分或最后部分的缓冲区时， DMA 将忽略 RDES2[3/2/1:0]（与总线宽度 128/64/32 相对应）。

● RDES3：接收描述符字 3

RDES3 包含描述符中第二个数据缓冲区或下一个描述符的地址指针。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBAP2 / NDA																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	RBAP2： 接收缓冲区 2 地址指针（下一个描述符地址） (RECEIVE BUFFER 2 ADDRESS POINTER (NEXT DESCRIPTOR ADDRESS))这些位会指示当使用描述符环结构时缓冲区 2 的物理地址。如果链接的第二个地址 (RDES1 [14]) 位置 1，则该地址包含指向下一个描述符所在物理寄存器的指针。如果 RDES1 [14] 置 1，则缓冲区（下一个描述符）地址指针必须是总线宽度对齐形式（ RDES3[3、 2 或 1:0] = 0，与总线宽度 128、 64 或 32 相对应， LSB 被内部忽略。）然而，当 RDES1 [14] 复位时， RDES3 值无任何限制，但以下情况除外：当使用 RDES3 值存储帧起点时， DMA 会使用所配置的值生成其缓冲区地址。当地址指针指向存储有帧中间部分或最后部分的缓冲区时， DMA 将忽略 RDES3[3、 2 或 1:0]（与总线宽度 128、 64 或 32 相对应）。

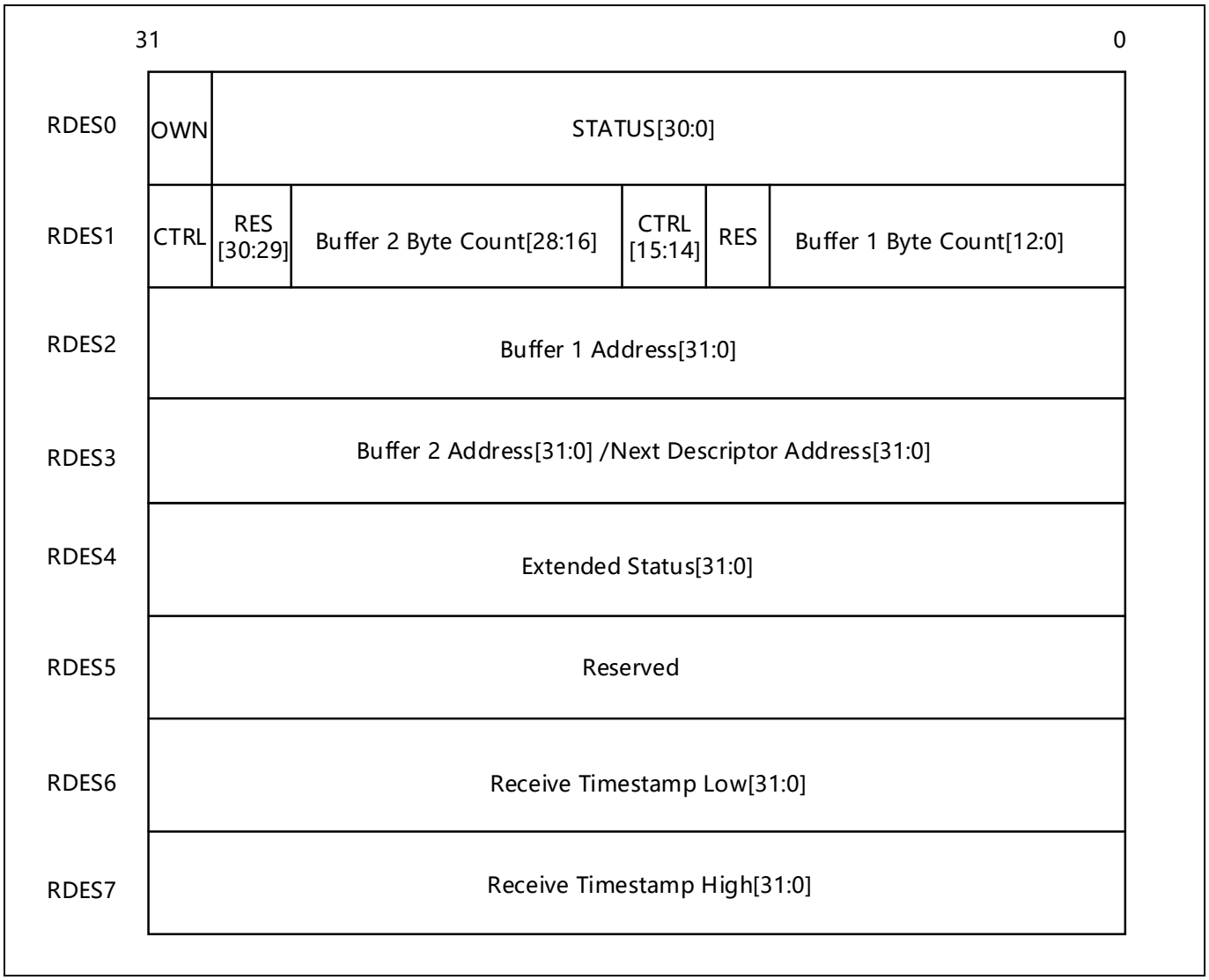
■ 增强 Rx DMA 描述符

如果时间戳已激活（TSE=1， ETH_PTPTSCR 位 0）或 IPv4 校验和减荷已激活（IPCO=1，ETH_MACCR 位 10），则必须使用增强的描述符（通过 ETHDMABMR 位 7 EDFE=1 使能）。

增强的描述符由 8 个 32 位字组成，即描述符正常大小的两倍。 RDES0、 RDES1、 RDES2 和 RDES3 的定义与常规接收描述符相同（请参见常规 Rx DMA 描述符）。 RDES4 包含扩展状态，而 RDES6 和 RDES7 包含时间戳。 RDES4、 RDES5、 RDES6 和 RDES7 的定义如下。

选择增强的描述符模式时，需要利用软件为每个描述符分配 32 个字节（ 8 个双字）内存。如果未使用时间戳或 IPv4 校验和减荷，则可以禁止增强的描述符格式，而软件可使用默认大小为 16 字节的常规描述符。

图 38-31 增强的接收描述符字段格式



● RDES4：接收描述符字 4

仅当存在与 IPv4 校验和或时间戳（由 RDES0 中的位 0 加以指示）相关的状态时，如下所示的扩展状态才有效。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES				L3L4FUM		L4 FM	L3 FM	RES									TP	PV	PFT	PMT				IPV6 PR	IPV4 PR	IPCB	IPPE	IPHE	IPPT		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:28	保留

27:26	L3L4FUM： 第 3 层和第 4 层过滤器编号匹配（LAYER 3 AND LAYER 4 FILTER NUMBER MATCHED）这些位表示与接收到的数据包匹配的第 3 层和第 4 层过滤器的编号： 00：过滤器 0 01：过滤器 1 10：过滤器 2 11：过滤器 3 此字段仅在位 24 或位 25 设置为高时有效。当多个过滤器匹配时，这些位仅给出最低过滤器编号
25	L4FM： 第 4 层过滤器匹配（LAYER 4 FILTER MATCH）该位置 1 时，表示接收到的数据帧与使能的第 4 层端口号字段之一匹配。只有在下列条件之一为真时，才会给出该状态： ● 第 3 层字段未使能，所有使能的第 4 层字段均匹配 ● 所有已使能的第 3 层和第 4 层过滤器字段均匹配 ● 如果有多个过滤器匹配，该位会给出由位[27:26]指示的过滤器的第 4 层过滤状态。
24	L3FM： 第 3 层过滤器匹配（LAYER 3 FILTER MATCH）该位置 1 时，表示接收到的数据帧与使能的第 3 层 IP 地址字段之一匹配。只有在下列条件之一为真时，才会给出该状态： ● 所有使能的第 3 层字段均匹配，所有使能的第 4 层字段均被绕过 ● 所有使能的过滤器字段均匹配 如果有多个过滤器匹配，该位会给出由位[27:26]指示的过滤器的第 3 层过滤状态。
23:15	保留
14	TP： 时间戳丢弃（TIMESTAMP DROPPED）该位置 1 时，表示此帧的时间戳已被捕获，但在接收 FIFO 中由于溢出而被丢弃。
13	PV： PTP 版本 (PTP VERSION)置 1 时，表示接收的 PTP 消息使用 IEEE 1588 的版本 2 格式。清零时，将使用版本 1 格式。仅当消息类型为非零时才有效
12	PFT： PTP 帧类型 (PTP FRAME TYPE)该位置 1 时，表示 PTP 消息直接通过以太网发送。该位清零且消息类型为非零时，表示 PTP 消息通过 UDP-IPV4 或 UDP-IPV6 发送。可通过位 6 或位 7 获取有关 IPV4 或 IPV6 的信息。
11:8	PMT： PTP 消息类型 (PTP MESSAGE TYPE)将对这些位进行编码，以给出所接收消息的类型。0000：未接收到任何 PTP 消息 0001： SYNC（所有时钟类型）0010： FOLLOW_UP（所有时钟类型）0011： DELAY_REQ（所有时钟类型）0100： DELAY_RESP（所有时钟类型）0101：PDELAY_REQ（在点对点透明时钟中）或 ANNOUNCE（在普通时钟或边界时钟中）0110：PDELAY_RESP（在点对点透明时钟中）或 MANAGEMENT（在普通时钟或边界时钟中）0111：PDELAY_RESP_FOLLOW_UP（在点对点透明时钟中）或 SIGNALING（在普通时钟或边界时钟中）— 1XXX - 保留
7	IPV6PR： 接收到 IPV6 数据包 (IPV6 PACKET RECEIVED)该位置 1 时，表示接收到的数据包为 IPV6 数据包。
6	IPV4PR： 接收到 IPV4 数据包 (IPV4 PACKET RECEIVED)该位置 1 时，表示接收到的数据包为 IPV4 数据包
5	IPCB： 绕过 IP 校验和 (IP CHECKSUM BYPASSED)该位置 1 时，表示绕过校验和减荷引擎。
4	IPPE： IP 有效负载错误 (IP PAYLOAD ERROR)该位置 1 时，表示由内核计算的 16 位 IP 有效负载校验和（即 TCP、 UDP 或 ICMP 校验和）与接收段中对应的校验和字段不匹配。当 TCP、UDP 或 ICMP 段长度与 IP 报头字段中的有效负载长度值不匹配时，它同样会置 1。
3	IPHE： IP 报头错误 (IP HEADER ERROR)该位置 1 时，表示由内核计算的 16 位 IPV4 报头校验和与接收的校验和字节不匹配，或 IP 数据报版本与以太网类型值不一致。

2:0	IPPT： IP 有效负载字节 (IP PAYLOAD TYPE)如果 IPV4 校验和减荷被激活 （ IPCO=1，ETH_MACCR 位 10） ，则这些位表示 IP 数据报中封装的有效负载类型。当 IP 报头出错或存在分段的 IP 时，这些位为 “00”。000：未知，或未处理 IP 有效负载 001：UDP010：TCP011：ICMP1XX：保留

● RDES5：接收描述符字 5

保留

● RDES6：接收描述符字 6

下表介绍了当接收描述符关闭且使能时间戳时对 RDES6 而言具有不同含义的字段。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTSL																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	RTSL： 接收帧时间戳低位 (RECEIVE FRAME TIMESTAMP LOW)DMA 使用为对应接收帧捕获的时间戳的 32 最低有效位更新该字段。 DMA 仅为最后描述符状态位 (RDES0[8]) 指示的接收帧的最后描述符更新该字段。当该字段和 RDES7 中的 RTSH 字段均显示 1 时，必须视为时间戳已损坏。

● RDES7：接收描述符字 7

下表介绍了当接收描述符关闭且使能时间戳时对 RDES7 而言具有不同含义的字段。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTSH																															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称
31:0	RTSH： 接收帧时间戳高位 (RECEIVE FRAME TIMESTAMP HIGH)DMA 使用为对应接收帧捕获的时间戳的 32 最高有效位更新该字段。 DMA 仅为最后描述符状态位 (RDES0[8]) 指示的接收帧的最后描述符更新该字段。当该字段和 RDES7 的 RTSL 字段都显示 1 时，必须将时间戳视为已损坏。

9. DMA 中断

中断可以作为各种事件的结果而生成。 ETH_DMASR 寄存器包含可能引起中断的所有位。ETH_DMAIER 寄存器对可能引起中断的每一个事件都包含一个使能位。

如 ETH_DMASR 寄存器中所描述，存在两组中断：正常中断和异常中断。通过向对应位位置写入 1 来清除中断。当清除了组内所有使能的中断时，对应的汇总位亦被清零。如果 MAC 内核是引发中断的原因，则 ETH_DMASR 寄存器中的任何 LPIS、TSTS 、PMTS 或 MMCS 位将置为高电平。

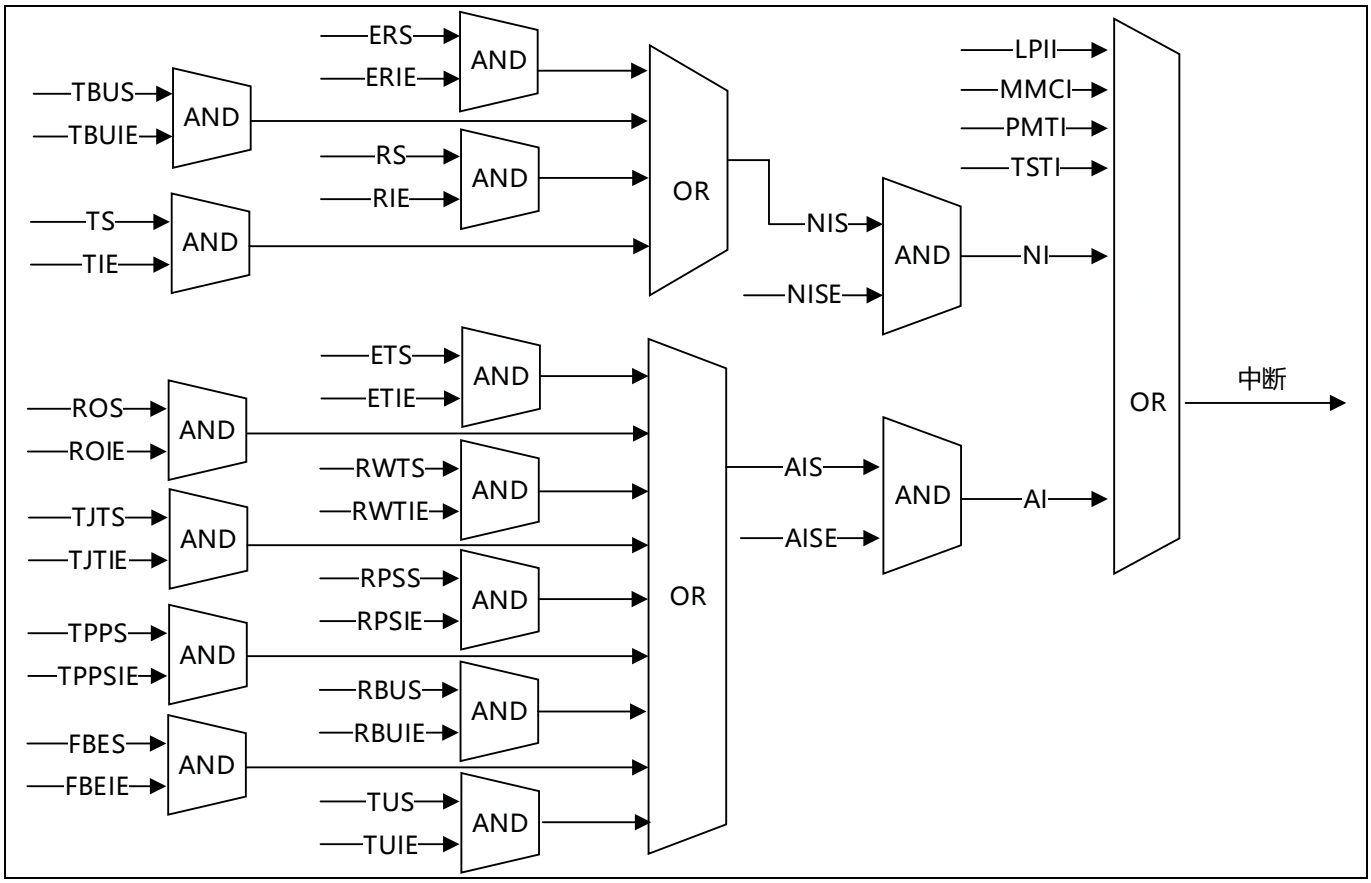
中断不会排队积压，如果中断事件在驱动程序对其做出响应之前发生，则不会再生成中断。例如，接收中断位（ ETH_DMASR 寄存器 [6]）指示一个或多个帧被传输到缓冲器。驱动程序必须扫描所有描述符，从 DMA

拥有的最后记录的位置到第一个位置。

对于同时发生的多个事件，一个中断只生成一次。驱动程序必须扫描 ETH_DMASR 寄存器，查找中断的原因。中断不会再次生成，除非在驱动程序已将 ETH_DMASR 寄存器中的适当位清零后发生新的中断事件。例如，控制器生成接收中断（ ETH_DMASR 寄存器[6]）而驱动程序开始读取 ETH_DMASR 寄存器。然后，发生接收缓冲器不可用（ ETH_DMASR 寄存器[7]）事件。驱动程序清除接收中断。即使是这时，也会由于活动或挂起的接收缓冲器不可用中断而生成一个新的中断。

可使用看门狗定时器（请参见 ETH_DMARIWTR 寄存器）灵活控制 RS 位(ETH_DMASR)寄存器)。当此看门狗定时器使用非零值编程时，只要 RxDMA 完成将接收的帧传输到系统存储器并且未触发接收状态（因未在相应接收描述符 (RDES1[31]) 使能接收状态），看门狗定时器即会被激活。当定时器按照编程值到时时，如果使能了 ETH_DMAIER 寄存器中的相应 RIE，则 RS 位会置 1 并会引发中断。当由于为该描述符将其使能而将一帧传输到存储器并且 RS 置 1 时，会在到时前禁止定时器。

图 38-32 中断方案



38.3.6. 以太网中断

以太网控制器有两个中断向量：一个专用于正常以太网操作，另一个仅当映射到 EXTI 线路 33 时用于以太网唤醒事件（LPI 中断和 PMT 中断）。

如 MAC 中断和 DMA 中断章节所列，第一个以太网向量为 MAC 和 DMA 生成的中断而保留。

第二个向量为发生唤醒事件时 LPI 和 PMT 生成的中断而保留。唤醒事件对 EXTI 线路 33 的映射使微控制器退出低功耗模式并生成一个中断。

当映射到 EXTI 线路 33 的以太网唤醒事件发生、使能了 MAC LPI 或 PMT 中断并且还使能了带上升沿检测的 EXTI 线路 33 中断时，会生成这两个中断。

注意： 读取 PMT 控制和状态寄存器会自动将接收的唤醒帧和接收的魔术数据包 PMT 中断标志清零。但是，由于用于这些标志的寄存器位于 CLK_RX 域，因此在固件能发现此更新前可能有显著的延迟。当 RX 时钟很慢（在 10 Mbit 模式）和当 AHB 总线为高频时，该延迟会特别长。由于从 PMT 到 CPU 的中断请求

基于 CLK_RX 域中的相同寄存器，所以即使在读取 PMT_CSR 之后， CPU 也可能错误地第二次调用中断例程。因此，可能需要固件轮询接收的唤醒帧和接收的魔术数据包位，并仅在发现它们都为‘0’时退出中断服务程序。

38.4. ETH 寄存器描述

38.4.1. 寄存器列表

ETH 寄存器基地址：0x40028000

偏移	名称	复位值	描述
0x0000	ETH_MACCR	0x00008000	MAC 配置寄存器
0x0004	ETH_MACFFR	0x00000000	MAC 帧过滤寄存器
0x0008	ETH_MACHTHR	0x00000000	MAC 散列表高位寄存器
0x000C	ETH_MACHTLR	0x00000000	MAC 散列表低位寄存器
0x0010	ETH_MACMIIAR	0x00000000	MAC MII 地址寄存器
0x0014	ETH_MACMIIDR	0x00000000	MAC MII 数据寄存器
0x0018	ETH_MACFCR	0x00000000	MAC 流控制寄存器
0x001C	ETH_MACVLANTR	0x00000000	MAC VLAN 标记寄存器
0x0024	ETH_MACDBGR	0x00000000	MAC 调试寄存器
0x0028	ETH_MACRWUFF	0x00000000	MAC 远程唤醒帧过滤寄存器
0x002C	ETH_MACPMTCSR	0x00000000	MAC PMT 控制和状态寄存器
0x0030	ETH_MACLPICSR	0x00000000	MAC LPI 控制和状态寄存器
0x0034	ETH_MACLPITCR	0x03e80000	MAC LPI 定时器控制寄存器
0x0038	ETH_MACISR	0x00000000	MAC 中断状态寄存器
0x003C	ETH_MACIMR	0x00000000	MAC 中断屏蔽寄存器
0x0040	ETH_MACA0HR	0x8000ffff	MAC 地址 0 高位寄存器
0x0044	ETH_MACA0LR	0xffffffff	MAC 地址 0 低位寄存器
0x0048	ETH_MACA1HR	0x0000ffff	MAC 地址 1 高位寄存器
0x004C	ETH_MACA1LR	0xffffffff	MAC 地址 1 低位寄存器
0x0050	ETH_MACA2HR	0x0000ffff	MAC 地址 2 高位寄存器
0x0054	ETH_MACA2LR	0xffffffff	MAC 地址 2 低位寄存器
0x0058	ETH_MACA3HR	0x0000ffff	MAC 地址 3 高位寄存器
0x005C	ETH_MACA3LR	0xffffffff	MAC 地址 3 低位寄存器
0x00DC	ETH_MACWTR	0x00000000	MAC 看门狗超时寄存器
0x0100	ETH_MMCCR	0x00000000	MMC 控制寄存器
0x0104	ETH_MMCRIR	0x00000000	MMC 接收中断寄存器
0x0108	ETH_MMCTIR	0x00000000	MMC 发送中断寄存器
0x010C	ETH_MMCRIMR	0x00000000	MMC 接收中断屏蔽寄存器
0x0110	ETH_MMCTIMR	0x00000000	MMC 发送中断屏蔽寄存器
0x014C	ETH_MMCTGFSCCR	0x00000000	MMC 在单个冲突后发送的良好帧计数器寄存器

0x0150	ETH_MMCTGFMCCR	0x00000000	MMC 在多个冲突后发送的良好帧计数器寄存器
0x0168	ETH_MMCTGFCR	0x00000000	MMC 发送的良好帧计数器寄存器
0x0194	ETH_MMCRFCECR	0x00000000	MMC 接收带有 CRC 错误帧计数器寄存器
0x0198	ETH_MMCRFAECR	0x00000000	MMC 接收带有对齐错误帧计数器寄存器
0x01C4	ETH_MMCRGUFCR	0x00000000	MMC 接收的良好单播帧计数器寄存器
0x0400	ETH_MACL3L4C0R	0x00000000	MAC L3 和 L4 控制 0 寄存器
0x0404	ETH_MACL4A0R	0x00000000	MAC L4 地址滤波器 0 寄存器
0x0410	ETH_MACL3A00R	0x00000000	MAC L3 地址 0 滤波器 0 寄存器
0x0414	ETH_MACL3A10R	0x00000000	MAC L3 地址 1 滤波器 0 寄存器
0x0418	ETH_MACL3A20R	0x00000000	MAC L3 地址 2 滤波器 0 寄存器
0x041C	ETH_MACL3A30R	0x00000000	MAC L3 地址 3 滤波器 0 寄存器
0x0430	ETH_MACL3L4C1R	0x00000000	MAC L3 和 L4 控制 1 寄存器
0x0434	ETH_MACL4A1R	0x00000000	MAC L4 地址滤波器 1 寄存器
0x0440	ETH_MACL3A01R	0x00000000	MAC L3 地址 0 滤波器 1 寄存器
0x0444	ETH_MACL3A11R	0x00000000	MAC L3 地址 1 滤波器 1 寄存器
0x0448	ETH_MACL3A21R	0x00000000	MAC L3 地址 2 滤波器 1 寄存器
0x044C	ETH_MACL3A31R	0x00000000	MAC L3 地址 3 滤波器 1 寄存器
0x0584	ETH_MACVTIRR	0x00000000	VLAN 标记包含或替换寄存器
0x0588	ETH_MACVHTR	0x00000000	VLAN 散列表寄存器
0x0700	ETH_PTPTSCR	0x00002000	PTP 时间戳控制寄存器
0x0704	ETH_PTPSSIR	0x00000000	PTP 亚秒递增寄存器
0x0708	ETH_PTPTSHR	0x00000000	PTP 时间戳高位寄存器
0x070C	ETH_PTPTSLR	0x00000000	PTP 时间戳低位寄存器
0x0710	ETH_PTPTSHUR	0x00000000	PTP 时间戳高位更新寄存器
0x0714	ETH_PTPTSLUR	0x00000000	PTP 时间戳低位更新寄存器
0x0718	ETH_PTPTSAR	0x00000000	PTP 时间戳加数寄存器
0x071C	ETH_PTPTTHR	0x00000000	PTP 目标时间高位寄存器
0x0720	ETH_PTPTTLR	0x00000000	PTP 目标时间低位寄存器
0x0728	ETH_PTPTSSR	0x00000000	PTP 时间戳状态寄存器
0x072C	ETH_PTPPPSCR	0x00000000	PTP PPS 控制寄存器
0x0730	ETH_PTPATSNR	0x00000000	PTP 辅助时间戳纳秒寄存器
0x0734	ETH_PTPATSSR	0x00000000	PTP 辅助时间戳秒寄存器
0x0760	ETH_PTPPPSIR	0x00000000	PTP PPS 间隔寄存器
0x0764	ETH_PTPPPSWR	0x00000000	PTP PPS 宽度寄存器
0x1000	ETH_DMABMR	0x00020100	DMA 总线模式寄存器
0x1004	ETH_DMATPDR	0x00000000	DMA 发送轮询要求寄存器

0x1008	ETH_DMARPDR	0x00000000	DMA 接收轮询要求寄存器
0x100C	ETH_DMARDLAR	0x00000000	DMA 接收描述符列表地址寄存器
0x1010	ETH_DMATDLAR	0x00000000	DMA 发送描述符列表地址寄存器
0x1014	ETH_DMASR	0x00000000	DMA 状态寄存器
0x1018	ETH_DMAOMR	0x00000000	DMA 工作模式寄存器
0x101C	ETH_DMAIER	0x00000000	DMA 中断使能寄存器
0x1020	ETH_DMAMFBOCR	0x00000000	DMA 丢失帧和缓冲区上溢计数器寄存器
0x1024	ETH_DMARIWTR	0x00000000	DMA 接收中断看门狗定时器寄存器
0x1048	ETH_DMACHTDR	0x00000000	DMA 当前主机发送描述符寄存器
0x104C	ETH_DMACHRDR	0x00000000	DMA 当前主机接收描述符寄存器
0x1050	ETH_DMACHTBAR	0x00000000	DMA 当前主机发送缓冲区地址寄存器

38.4.2. MAC 配置寄存器(ETH_MACCR: 00h)

位域	名称	属性	复位值	描述
31	RSV	-	0	保留，必须保持复位值
30:28	SAIRC	RW	0	源地址插入或替换控制 (Source Address Insertion or Replacement Control) 该字段控制对所有已发送数据包的源地址插入或替换。位 30 指定用于源地址插入或替换的 MAC 地址寄存器（0 或 1）（基于位 [29:28] 的值）： 10： – 如果位 30 置 0，则 MAC 会将 MAC 地址 0 寄存器（MAC 寄存器 192 和 193）的内容插入到所有已发送数据包的 SA 字段中。 – 如果位 30 置 1，且选择了 Enable MAC Address Register 1（使能 MAC 地址寄存器 1）选项，则 MAC 会将 MAC 地址 1 寄存器（MAC 寄存器 194 和 195）的内容插入到所有已发送数据包的 SA 字段中。 11： – 如果位 30 置 0，则 MAC 会替换所有已发送数据包 SA 字段中的 MAC 地址 0 寄存器（MAC 寄存器 192 和 193）的内容。 – 如果位 30 置 1，且使能了 MAC 地址寄存器 1，则 MAC 会替换所有已发送数据包 SA 字段中的 MAC 地址 1 寄存器（MAC 寄存器 194 和 195）的内容。 注：对此字段进行的更改仅在数据包起始时生效。如果在发送数据包时对该寄存器字段进行写操作，则只有后续数据包可使用更新值，即，当前数据包不使用更新值。
27	S2KP	RW	0	2K 数据包的 IEEE 802.3as 支持(IEEE 802.3as Support for 2K Packets) 0： JE 位未置 1 时， MAC 将所有接收到的大小超过 1,518 字节（对于带标记的数据包，为 1,522 字节）的数据包视为大型数据包。 1： MAC 将所有长度未超过 2,000 字节的数据包视为正常数据包。 JE 位未置 1 时， MAC 将所有接收到的大小超过 2K 字节的数据包视为大型数据包。 注：JE 位置 1 时，将该位置 1 对大型数据包状态无影响
26	RSV	-	0	保留，必须保持复位值
25	CSTF	RW	0	类型帧的 CRC 去除(CRC Stripping for Type Frames) 该位置 1 时，在将帧转发到应用之前，将去除并丢弃所有以太网类型（类型字段大于 0x0600）帧的最后 4 个字节（FCS）。
24	RSV	-	0	保留，必须保持复位值
23	WD	RW	0	禁止看门狗(Watchdog Disable) 0：MAC 允许接收的帧不超过 2048 字节并会截断超过此限制接收的任何字节。 1：MAC 禁止接收器上的看门狗定时器并可接收多达 16383 字节的帧。
22	JD	RW	0	禁止 Jabber (Jabber Disable) 0：如果发送期间应用程序发送超过 2048 字节的数据（如果 JE 位置 1，则是 10240 字节数据），MAC 会截断发送器。 1：MAC 禁止发送器上的 jabber 定时器并可发送多达 16384 字节的帧。

21	RSV	-	0	保留，必须保持复位值
20	JE	RW	0	巨型数据包使能(Jumbo Packet Enable) 该位置 1 时， MAC 允许 9,018 字节的巨型数据包（对于带 VLAN 标记的数据包，为 9,022 字节），且不会在 Rx 数据包状态中报告大型数据包错误。
19:17	IFG	RW	0	帧间间隙(Interframe Gap) 000： 96 位时间 001： 88 位时间 010： 80 位时间 … 111： 40 位时间 注：在半双工模式下，最小 IFG 仅能配置为 64 位时间 (IFG = 100)。不考虑更低的值
16	CSD	RW	0	禁止载波侦听(Carrier Sense Disable) 0： MAC 发送器会生成载波侦听错误，甚至中止发送。 1：会使 MAC 发送器忽略半双工模式下帧发送期间的 MII CRS 信号。不会因在此发送期间载波丢失或无载波而生成错误。
15	RSV	-	1	保留，必须保持复位值
14	FES	RW	0	快速以太网速度(Fast Ethernet Speed) 0：10 Mb/s 1：100 Mb/s
13	ROD	RW	0	禁止接收自身(Receive Own Disable) 0：MAC 接收发送时 PHY 提供的所有包。 1：MAC 禁止在半双工模式下接收帧。 如果 MAC 在全双工模式下工作，该位不适用。
12	LM	RW	0	回送模式(Loopback Mode) 当该位置 1 时， MAC 在 MII 回送模式下工作。回送正确工作需要 MII 接收时钟输入(RX_CLK)，因为发送时钟不是内部回送的。
11	DM	RW	0	双工模式 (Duplex Mode) 当该位置 1 时， MAC 在全双工模式下工作，此时它可以同时发送和接收。
10	IPCO	RW	0	IPv4 校验和减荷(IPv4 Checksum Offload) 0：禁止接收器中的校验和减荷功能并始终将相应的 PCE 和 IPHCE 状态位清零。 1：使能接收的帧有效载荷的 TCP/UDP/ICMP 标头的 IPv4 校验和检查。
9	DR	RW	0	禁止重试(Disable Retry) 0：MAC 会尝试根据 BL 的设置进行重试。 1：MAC 仅尝试一次发送。当在 MII 模式下发生冲突时， MAC 会忽略当前帧发送并以发送帧状态下过度冲突错误报告帧中止。 注：该位仅在半双工模式下适用。

8	RSV	-	0	保留，必须保持复位值
7	APCS	RW	0	自动 Pad/CRC 去除(Automatic pad/CRC Stripping) 0：MAC 会不加修改地传送所有传入的帧。 1：MAC 仅在长度字段值小于或等于 1536 字节时去除传入帧上的 Pad/FCS 字段。将所有已接收帧传送给应用程序时，如果长度字段大于或等于 1536 字节，将都不去除 Pad 或 FCS 字段。
6:5	BL	RW	0	后退限制(Back-off Limit) 后退限制决定发生冲突后重试期间 MAC 在重新安排一次发送之前等待的随机整数 (r) 个时间片延迟（对于 1000 Mb/s 为 4 096 位时间，对于 10/100 Mb/s 为 512 位时间） 00： k = min (n, 10) 01： k = min (n, 8) 10： k = min (n, 4) 11： k = min (n, 1) 其中 n = 重新发送尝试的次数。随机整数 r 的取值范围为 $0 \le r < 2^k$。
4	DC	RW	0	检查延迟(Deferral Check) 0：禁止延迟检查功能， MAC 发生延迟，直到 CRS 信号变为无效信号。 1：在 MAC 中使能了检查延迟功能。当发送状态机在 10/100 Mb/s 模式下延迟超过 24288 位时间时， MAC 将指示帧中止状态，同时在发送帧状态中将过度延迟错误位置 1。当发送器准备好发送但因 MII 上有有效 CRS（载波侦听）信号而被阻止时延迟开始。延迟时间是非累积性的。如果发送器延迟 10 000 位时间，然后发送、冲突、后退，然后在完成后退后不得不再次延迟，则延迟定时器复位为 0 并重新开始。 注：该位仅在半双工模式下适用。
3	TE	RW	0	发送器使能(Transmitter Enable) 0：在当前帧的发送完成后禁止 MAC 发送状态机，并且不再发送任何帧。 1：使能 MAC 的发送状态机，以便在 MII 接口上进行发送。
2	RE	RW	0	接收器使能(Receiver Enable) 0：MAC 接收状态机会在完成当前数据包的接收后被禁止。 Rx 状态机不会再从 MII 接口接收任何数据包。 1：使能 MAC 的接收状态机，以便从 MII 接口接收数据包。
1:0	PRELEN	RW	0	发送数据包的报头长度(Preamble Length for Transmit packets) 这些位控制被添加到每个 Tx 数据包的开头的报头字节数。仅当 MAC 工作在全双工模式下时，才可缩短报头。 00：7 字节报头 01：5 字节报头 10：3 字节报头 11：保留

38.4.3. MAC 帧过滤寄存器(ETH_MACFFR: 04h)

位域	名称	属性	复位值	描述
31	RA	RW	0	接收所有(Receive All) 0：MAC 接收器传送给应用程序的只是通过了 SA/DA 地址过滤的那些帧。 1：MAC 接收器将所有接收的帧传送到应用程序，不管它们是否已通过地址过滤。 SA/DA 过滤的结果在接收状态字的相应位更新（通过或失败）。
30:22	RSV	-	0	保留，必须保持复位值
21	DNTU	RW	0	丢弃非 TCP/UDP over IP 数据包(Drop Non-TCP/UDP over IP Packets) 0：MAC 会转发所有非 TCP/UDP over IP 数据包。 1：MAC 会丢弃非 TCP/UDP over IP 数据包。MAC 仅转发由第 4 层过滤器处理的数据包。 如果未选择 Enable Layer 3 and Layer 4 Packet Filter（使能第 3 层和第 4 层数据包过滤器）选项，该位将保留（处于 RO 状态，且具有默认值）。
20	IPFE	RW	0	第 3 层和第 4 层过滤器使能(Layer 3 and Layer 4 Filter Enable) 0：无论第 3 层和第 4 层字段的匹配状态如何， MAC 都会转发所有数据包。 如果未选择 Enable Layer 3 and Layer 4 Packet Filter（使能第 3 层和第 4 层数据包过滤器）选项，该位将保留（处于 RO 状态，且具有默认值）。 1： MAC 会丢弃与使能的第 3 层和第 4 层过滤器不匹配的数据包。如果未使能第 3 层或第 4 层过滤器以进行匹配，则该位无任何影响。
19:17	RSV	-	0	必须保持复位值
16	VTFE	RW	0	VLAN 标记过滤器使能(VLAN Tag Filter Enable) 0：无论 VLAN 标记的匹配状态如何， MAC 都会转发所有数据包。 1：MAC 将丢弃与 VLAN 标记不匹配的带 VLAN 标记的数据包。
15:11	RSV	-	0	保留，必须保持复位值
10	HPF	RW	0	散列或完美过滤器(Hash or Perfect Filter) 0：HU 或 HM 位置 1， 则只有在数据包与散列过滤器匹配时才会通过。 1：如果数据包与完美过滤或散列过滤（通过 HM 或 HU 位设置）匹配，则地址过滤器会通过此数据包。
9	SAF	RW	0	源地址过滤器使能(Source Address Filter Enable) 0：MAC 会根据 SA 地址比较结果，通过更新 Rx 状态的 SAF 位来将接收到的数据包转发到应用程序。 1：MAC 会将所接收数据包的 SA 字段与使能的 SA 寄存器中编程的值进行比较。如果比较结果为失败， MAC 会丢弃数据包。
8	SAIF	RW	0	源地址过反向(SA Inverse Filtering) 0：如果数据包的 SA 与 SA 寄存器中编程的值不匹配，则会将其标记为 SA 地址过滤失败。 1：地址检查块在反向过滤模式下进行 SA 地址比较。如果数据包的 SA 与 SA 寄存器中编程的值匹配，则会将其标记为 SA 地址过滤失败。

7:6	PCF	RW	0	通过控制帧(Pass Control Packets) 这些位控制所有控制数据包（包括单播和多播暂停数据包）的转发。 00： MAC 会过滤掉所有控制数据包，阻止它们到达应用程序。 01：即使所有控制数据包均未通过地址过滤， MAC 也仍将它们转发给应用程序（暂停数据包除外）。 10：即使所有控制数据包均未通过地址过滤， MAC 也仍将它们转发给应用程序。 11： MAC 转发通过地址过滤的控制数据包。
5	DBF	RW	0	禁止广播帧(Disable Broadcast Farmes) 0：地址过滤模块会通过所有接收到的广播数据帧。 1：地址过滤模块会阻止所有传入的广播数据帧。此外，它还会覆盖所有其他过滤设置。
4	PAM	RW	0	通过所有多播(Pass All Multicast) 0：多播数据包的过滤取决于 HM 位 1：指示通过接收到的带多播目标地址（目标地址字段的第一位是“1”）的所有数据包。
3	DAIF	RW	0	DA 反向过滤(DA Inverse Filtering) 0：执行数据包的正常过滤。 1：地址检查块在反向过滤模式下对单播和多播数据包均进行 DA 地址比较。
2	HM	RW	0	散列多播(Hash Multicast) 0：MAC 对多播数据包执行完美目标地址过滤，即，将 DA 字段与 DA 寄存器中编程的值进行比较。 1：MAC 根据散列表对接收到的多播数据包执行目标地址过滤。 如果未选择 Enable Address Filter Hash Table（使能地址过滤散列表）选项，该位将保留（且处于 RO 状态）。
1	HU	RW	0	散列单播(Hash Unicast) 0：MAC 对单播数据包执行完美目标地址过滤，即，将 DA 字段与 DA 寄存器中编程的值进行比较。 1：MAC 根据散列表对单播数据包执行目标地址过滤。 如果未选择 Enable Address Filter Hash Table（使能地址过滤散列表）选项，该位将保留（且处于 RO 状态）。
0	PM	RW	0	混合模式(Promiscuous Mode) 当该位置 1 时，不论目标或源地址为何，地址过滤模块都会通过所有传入的数据包。 PM 置 1 时，始终将接收状态字的 SA 或 DA 过滤失败状态位清零。

38.4.4. MAC 散列表高位寄存器(ETH_MACHTHR: 08h)

位域	名称	属性	复位值	描述

31:0	HTH	R/W	0	散列表高位(Hash Table High) 该字段包含散列表的高 32 位。

38.4.5. MAC 散列表低位寄存器(ETH_MACHTLR: 0Ch)

位域	名称	属性	复位值	描述
31:0	HTL	R/W	0	散列表低位(Hash Table Low) 该字段包含散列表的低 32 位。

38.4.6. MAC MII 地址寄存器(ETH_MACMIIAR: 10h)

位域	名称	属性	复位值	描述
31:16	RSV	-	0	保留，必须保持复位值
15:11	PA	RW	0	PHY 地址(PHY Address) 该字段指示正在访问 32 个可能的 PHY 器件中的哪一个。
10:6	MR	RW	0	MII 寄存器(MII Register) 这些位在所选 PHY 器件中选择需要 MII 寄存器。

5:2	CR	RW	0	CSR 时钟范围(Clock Range) CSR 时钟范围选择根据设计中使用的 CSR 时钟频率确定 MDC 时钟的频率： 0000： CSR 时钟 = 60-100 MHz； MDC 时钟 = CSR 时钟/42 0001： CSR 时钟 = 100-150 MHz； MDC 时钟 = CSR 时钟/62 0010： CSR 时钟 = 20-35 MHz； MDC 时钟 = CSR 时钟/16 0011： CSR 时钟 = 35-60 MHz； MDC 时钟 = CSR 时钟/26 0100： CSR 时钟 = 150-250 MHz； MDC 时钟 = CSR 时钟/102 0101： CSR 时钟 = 250-300 MHz； MDC 时钟 = CSR 时钟/124 0110， 0111：保留 适用于各个值的建议 CSR 时钟频率范围（bit5 = 0 时）可确保 MDC 时钟大致介于 1.0 MHz 到 2.5 MHz 频率范围之间。 Bit5 置 1 时，可以实现高于 2.5 MHz 频率限制（在 IEEE 802.3 中规定）的 MDC 时钟频率，并编程较低值的时钟分频器。例如，当 CSR 时钟频率为 100 MHz，并且将这些位编程为 1010 时，所得到的 MDC 时钟为 12.5 MHz，此值高于 IEEE 802.3 中规定的范围。仅当接口芯片支持更快的 MDC 时钟时才编程以下值： 1000： CSR 时钟/4 1001： CSR 时钟/6 1010： CSR 时钟/8 1011： CSR 时钟/10 1100： CSR 时钟/12 1101： CSR 时钟/14 1110： CSR 时钟/16 1111： CSR 时钟/18
1	MW	RW	0	MII 写(MII Write) 0：表示会启动一个读操作，将数据放入 MII 数据寄存器。 1：告知 PHY，将要启动一个使用 MII 数据寄存器的写操作。
0	MB	RW	0	MII 忙碌(MII Busy) 向 ETH_MACMIIAR 和 ETH_MACMIIDR 写入前，此位应读取逻辑 0。向 ETH_MACMIIAR 写入过程中，此位也必须复位为 0。在 PHY 寄存器访问过程中，此位由应用程序设为 0b1，指示读或写访问正在进行中。在对 PHY 进行写操作过程中， ETH_MACMIIDR （ MII 数据）应始终保持有效，直到 MAC 将此位清零。在对 PHY 进行读操作过程中， ETH_MACMIIDR 始终无效，直到 MAC 将此位清零。在此位清零后，才可以向 ETH_MACMIIAR （ MII 地址）写入。

38.4.7. MAC MII 数据寄存器(ETH_MACMIIDR: 14h)

位域	名称	属性	复位值	描述
31:16	RSV	-	0	保留，必须保持复位值

15:0	MD	RW	0	MII 数据(MII Data) 其中包含在某次管理读操作之后从 PHY 中读取的 16 位数据值，或在某次管理写操作之前要写入 PHY 的 16 位数据值。

38.4.8. MAC 流控制寄存器(ETH_MACFCR: 18h)

位域	名称	属性	复位值	描述
31:16	PT	R/W	0	暂停时间(Pause Time) 此字段保留发送控制帧中暂停时间字段要使用的值。
15:4	RSV	-	0	保留，必须保持复位值
3	UPFD	R/W	0	单播暂停帧检测(Unicast Pause Frame Detect) 0：MAC 仅检测具有 802.3x 标准中指定的唯一多播地址的暂停帧。 1：MAC 除了检测具有唯一多播地址的暂停帧外，还会检测具有 ETH_MACA0HR 和 ETH_MACA0LR 寄存器所指定的站单播地址的暂停帧。
2	RFCE	R/W	0	接收流控制使能(Receive Flow Control Enable) 0：禁止暂停帧的解码功能 1：MAC 对接收到的暂停帧进行解码，并禁止其在指定时间（暂停时间）内发送。
1	TFCE	R/W	0	发送流控制使能(Transmit Flow Control Enable) 在全双工模式下，当此位置 1 时， MAC 将使能流控制操作来发送暂停帧。当此位复位时，将禁止 MAC 中的流控制操作， MAC 不会传送任何暂停帧。 在半双工模式下，当此位置 1 时， MAC 将使能背压操作。当此位复位时，将禁止背压功能。
0	FCB/BPA	R/W	0	流控制忙碌或背压激活(Flow Control Busy/Back Pressure Activate) 此位在全双工模式下启动暂停控制帧，在半双工模式下， TFCE 位置 1 时，会激活背压功能。全双工模式下，向流控制寄存器写入数据前此位应读为 0。要启动暂停控制帧，应用程序必须将此位置 1。在控制帧发送过程中，此位保持置 1 以指示帧发送正在进行中。当暂停控制帧发送完成后， MAC 会将此位复位为 0。将此位清零后，才可以对流控制寄存器执行写操作。在半双工模式下，当此位置 1（ TFCE 也置 1）时， MAC 内核将置位背压功能。在背压操作期间，当 MAC 接收到新帧时，发送器会开始发送一个导致冲突的 JAM 模式。当 MAC 配置为全双工模式时，将自动禁止 BPA。

38.4.9. MAC VLAN 标记寄存器(ETH_MACVLANTR: 1Ch)

位域	名称	属性	复位值	描述
31:20	RSV	-	0	保留，必须保持复位值

19	VTHM	R/W	0	VLAN 标志散列表匹配使能 0：不执行 VLAN 散列匹配操作。 1：则使用 VLAN 标记的 CRC 的四个最高有效位来索引 VLAN Hash Table Register 的内容。VLAN 散列表寄存器的值 1 对应于索引，表示数据包与 VLAN 散列表匹配。ETV 置 1 时使用 12 位 VLAN 标识符（VID）的 CRC 进行比较。ETV 复位时，使用 16 位 VLAN 标记的 CRC 进行比较。
18	ESVL	R/W	0	使能 S-VLAN 该位置 1 时，MAC 发送器和接收器会将 S-VLAN 数据包（类型=0x88A8）视为带 VLAN 标记的有效数据包。
17	VTIM	R/W	0	VLAN 标记反向匹配使能(VLAN Tag Inverse Match Enable) 0：使能 VLAN 标记完美匹配。含有匹配 VLAN 标记的数据包被标记为匹配。 1：使能 VLAN 标记反向匹配。不含匹配 VLAN 标记的数据包被标记为匹配。
16	ETV	R/W	0	使能 12 位 VLAN 标记比较(Enable 12-Bit VLAN Tag Comparison) 0：所接收 VLAN 数据包的第 15 个和第 16 个字节的所有 16 位都用于比较和 VLAN 散列过滤。 1：使用 12 位 VLAN 标识符进行比较和过滤，而不使用完整的 16 位 VLAN 标记。VLAN 标记的位 [11:0] 与所接收带 VLAN 标记的数据包中的相应字段进行比较。类似地，使能时，只有所接收数据包中的 12 位 VLAN 标记用于基于散列的 VLAN 过滤。
15:0	VL	R/W	0	用于接收数据包的 VLAN 标记标识符(VLAN Tag Identifier for Receive Packets) 该字段包含用于识别 VLAN 数据包的 802.1Q VLAN 标记。该 VLAN 标记标识符将与接收到的数据包的第 15 个和第 16 个字节进行比较，以识别 VLAN 数据包。下面对该字段的各个位进行了说明： 位 [15:13]：用户优先级 位 12：标准格式指示符（CFI）或丢弃合法指示符（DEI） 位 [11:0]： VLAN 标记的 VLAN 标识符（VID）字段 ETV 位置 1 时，仅使用 VID 进行比较。如果该字段（ ETV 置 1 时为 [11:0]）全部为零，则 MAC 不会检查用于 VLAN 标记比较的第 15 个和第 16 个字节，而是将类型字段值为 0x8100 或 0x88A8 的所有数据包均声明为 VLAN 数据包。

38.4.10. MAC 调试寄存器(ETH_MACDBGR: 24h)

位域	名称	属性	复位值	描述
31:26	RSV	-	0	保留，必须保持复位值
25	TFF	RO	0	Tx FIFO 已满(Tx FIFO Full) 当它为高电平时，表示 Tx FIFO 已满，因此无法再接受发送的帧。
24	TFNE	RO	0	Tx FIFO 非空(Tx FIFO Not Empty) 当它为高电平时，表示 TxFIFO 不为空，还留有一些数据要发送。

23	RSV	-	0	保留，必须保持复位值
22	TFWA	RO	0	Tx FIFO 写有效(Tx FIFO Write Active) 当它为高电平时，表示 Tx FIFO 写控制器有效且正在将数据传输到 Tx FIFO
21:20	TFRS	RO	0	Tx FIFO 读状态(Tx FIFO Read Status) 此字段指示 TxFIFO 读控制器的状态： 00：空闲状态 01：读状态（将数据传输到 MAC 发送器） 10：等待来自 MAC 发送器的 TxStatus 11：写入收到的 TxStatus 或刷新 TxFIFO
19	MTP	RO	0	MAC 发送器处于暂停(MAC Transmitter in Pause) 当它为高电平时，表示 MAC 发送器处于暂停条件（仅在全双工模式下），因此不会安排发送任何帧。
18:17	MTFCS	RO	0	MAC 发送帧控制器状态(MAC Transmit Frame Controller Status) 此字段指示 MAC 发送帧控制器的状态： 00：空闲 01：等待前一个帧的状态或 IFG/回退阶段结束 10：生成并发送暂停控制帧（在全双工模式下） 11：传输要发送的输入帧
16	MMTEA	RO	0	MAC MII 发送引擎有效(MAC MII Transmit Engine Active) 当它为高电平时，表示 MAC MII 发送引擎正在主动发送数据而未处于空闲状态。
15:10	RSV	-	0	保留，必须保持复位值
9:8	RFFL	RO	0	Rx FIFO 填充级别(Rx FIFO Fill Level) 它指示 Rx FIFO 填充级别的状态： 00： Rx FIFO 为空 01： Rx FIFO 填充级别低于流控制取消激活阈值 10： Rx FIFO 填充级别高于流控制激活阈值 11： Rx FIFO 已满
7	RSV	-	0	保留，必须保持复位值
6:5	RFRCS	RO	0	Rx FIFO 读控制器状态(Rx FIFO Read Controller Status) 它指示 Rx FIFO 读控制器的状态： 00：空闲状态 01：读取帧数据 10：读取帧状态（或时间戳） 11：刷新帧数据和状态
4	RFWCS	RO	0	Rx FIFO 写控制器状态(Rx FIFO Write Controller Active) 当它为高电平时，表示 Rx FIFO 写控制器有效并正在将收到的帧传输到 FIFO。

3	RSV	-	0	保留，必须保持复位值
2:1	MRFFCS	RO	0	MAC 接收帧 FIFO 控制器状态(MAC Receive Frame FIFO Controller Status) 当这些位为高电平时，表示 MAC 接收帧控制器模块的各个小 FIFO 读和写控制器的有效状态。 位 1 表示小 FIFO 读控制器 位 0 表示小 FIFO 写控制器
0	MMRPES	RO	0	MAC MII 接收协议引擎状态(MAC MII Receive Protocol Engine Status) 当它为高电平时，表示 MAC MII 接收协议引擎正在主动接收数据而未处于空闲状态。

38.4.11. MAC 远程唤醒帧过滤寄存器(ETH_MACRWUFF: 28h)

有八个唤醒帧过滤寄存器。要对每个寄存器执行写操作，需要逐个值加载唤醒帧过滤寄存器。连续加载唤醒帧过滤寄存器八次，便可对唤醒帧过滤寄存器加载所需的值。读操作与写操作相同。要读取八个值，必须读唤醒帧过滤寄存器八次后，才能到达最后一个寄存器。每次读 写操作都会将唤醒帧过滤寄存器指向下一个过滤寄存器。有关详细信息，请参见远程唤醒帧过滤寄存器章节。

名称	描述							
唤醒帧过滤寄存器 0	过滤器 0 字节屏蔽							
唤醒帧过滤寄存器 1	过滤器 1 字节屏蔽							
唤醒帧过滤寄存器 2	过滤器 2 字节屏蔽							
唤醒帧过滤寄存器 3	过滤器 3 字节屏蔽							
唤醒帧过滤寄存器 4	RSVD	过滤器 3 命令	RSVD	过滤器 2 命令	RSVD	过滤器 1 命令	RSVD	过滤器 0 命令
唤醒帧过滤寄存器 5	过滤器 3 偏移		过滤器 2 偏移		过滤器 1 偏移		过滤器 0 偏移	
唤醒帧过滤寄存器 6	过滤器 1 CRC-16				过滤器 0 CRC-16			
唤醒帧过滤寄存器 7	过滤器 3 CRC-16				过滤器 2 CRC-16			

38.4.12. MAC PMT 控制和状态寄存器(ETH_MACPMTCSR: 2Ch)

位域	名称	属性	复位值	描述
31	RWFFPR	RW	0	远程唤醒帧过滤器寄存器指针复位（Remote Wakeup Frame Filter Register Pointer Reset) 当此位置 1 时，会将远程唤醒帧过滤寄存器指针复位为 3’ b000。它会在 1 个时钟周期后自动清零。
30:27	RSV	-	-	保留，必须保持复位值

26:24	RWFP	R	0	远程唤醒 FIFO 指针(Remote Wakeup FIFO Pointer) 该字段给出了远程唤醒帧过滤寄存器指针的当前值(0 到 7)。如果该指针的值等于 7，则对远程唤醒帧过滤寄存器进行写操作时，远程唤醒帧过滤寄存器的内容将被传输到 eth_mii_rx_clk 域。
23:10	RSV	-	-	保留，必须保持复位值
9	GU	RW	0	全局单播（Global Unicast) 当此位置 1 时，它会将 MAC (DAF) 地址确认所过滤的任意单播包使能为唤醒帧。
8:7	RSV	-	0	保留，必须保持复位值
6	WFR	RW	0	接收到唤醒帧（Wakeup Frame Received) 当此位置 1 时，表示因接收唤醒帧而生成了电源管理事件。通过对此寄存器执行读操作可将此位清零。
5	MPR	RW	0	接收到魔术数据包（Magic Packet Received) 当此位置 1 时，表示因接收魔术数据包而生成了电源管理事件。通过对此寄存器执行读操作可将此位清零。
4:3	RSV	-	0	保留，必须保持复位值
2	WFE	RW	0	唤醒帧使能(Wakeup Frame Enable) 当此位置 1 时，会因为接收到唤醒帧而使能电源管理事件的生成。
1	MPE	RW	0	魔术数据包使能（Magic Packet Enable) 当此位置 1 时，会因为接收到魔术数据包而使能电源管理事件的生成。
0	PD	RW	0	掉电（Power Down) 当此位置 1 时，所有接收到的帧都将丢弃。接收到魔术数据包或唤醒帧时，此位会自动清零，同时禁止掉电模式。此位清零后收到的帧会转发到应用程序。只有魔术数据包使能(Magic Packet Enable）或唤醒帧使能（Wakeup Frame Enable）位置 1 时，才可以将此位置 1。

38.4.13. MAC LPI 控制和状态寄存器(ETH_MACLPICSR: 30h)

位域	名称	属性	复位值	描述
31:20	RSV	-	0	保留，必须保持复位值
19	LPITXA	RW	0	LPI Tx 自动化(LPI Tx Automate) 该位控制 MAC 在发送端进入或退出 LPI 模式时的行为。 如果 LPITXA 和 LPIEN 位均置 1，则只有在已发送所有未完成的数据包(在内核中)和挂起数据包(在应用程序接口中)之后，MAC 才会进入 LPI 模式。应用程序发送任何数据包以供传送，或应用程序发出 Tx FIFO 刷新命令时，MAC 将退出 LPI 模式。此外，MAC 退出 LPI 模式时会自动将 LPIEN 位清零。当 MAC 处于 LPI 模式时，如果 ETH_DMAOMR 的 BIT20 Tx FIFO 刷新置为有效时，则 MAC 会退出 LPI 模式。 该位为 0 时，LPIEN 位直接控制 MAC 进入或退出 LPI 模式时的行为。

18	RSV	-	0	保留，必须保持复位值
17	PLS	RW	0	PHY 链路状态(PHY Link Status) 该位指示 PHY 的链路状态。只有在链路接通状态(OKAY)至少保持 LPI LS 定时器所指示的时间时，MAC 发送器才会将 LPI 模式置为有效。 该位置 1 时，将链路视为连通状态(UP)；该位复位时，将链路视为断开状态。
16	LPIEN	RW	0	LPI 使能(LPI Enable) 该位置 1 时，会命令 MAC 发送器进入 LPI 状态。该位复位时，会命令 MAC 退出 LPI 状态并恢复正常发送。 当 LPITXA 位置 1 时且由于新数据包到达等待发送而使 MAC 退出 LPI 状态时，该位清零。
15:10	RSV	-	0	保留，必须保持复位值
9	RLPIST	RO	0	接收 LPI 状态(Receive LPI State) 该位置 1 时，表示 MAC 正在通过 MII 接口接收 LPI 模式。
8	TLPIST	RO	0	发送 LPI 状态(Transmit LPI State) 该位置 1 时，表示 MAC 正在通过 MII 接口发送 LPI 模式。
7:4	RSV	-	0	保留，必须保持复位值
3	RLPIEX	RO	0	接收 LPI 退出(Receive LPI Exit) 该位置 1 时，表示 MAC 接收器已停止通过 MII 接口接收 LPI 模式，并且退出 LPI 状态和恢复正常接收。通过对此寄存器执行读操作可将此位清零。
2	RLPIEN	RO	0	接收 LPI 进入(Receive LPI Entry) 该位置 1 时，表示 MAC 接收器已经接收到 LPI 模式并进入 LPI 状态。通过对此寄存器执行读操作可将此位清零。
1	TLPIEX	RO	0	发送 LPI 退出(Transmit LPI Exit) 该位置 1 时，表示应用程序将 LPIEN 位清零，且 LPI TW 定时器到期后，MAC 发送器退出了 LPI 状态。通过对此寄存器执行读操作可将此位清零。
0	TLPIEN	RO	0	发送 LPI 进入(Transmit LPI Entry) 该位置 1 时，表示由于 LPIEN 位置 1 而使 MAC 发送器进入了 LPI 状态。通过对此寄存器执行读操作可将此位清零。

38.4.14. MAC LPI 定时器控制寄存器(ETH_MACLPITCR: 34h)

位域	名称	属性	复位值	描述
31:26	RSV	-	0	保留，必须保持复位值
25:16	LST	RW	0x3E8	LPI LS 定时器(LPI LS Timer) 该字段指定在可以将 LPI 模式发送到 PHY 端之前，来自 PHY 的链路状态(OKAY)应持续的最短时间(以毫秒为单位)。即使 LPIEN 位置 1，MAC 也不会发送 LPI 模式，除非 LPI LS 定时器达到编程端计数。LPI LS 定时器的默认值 1000(1 秒)

15:0	TWT	RW	0	LPI TW 定时器(LPI TW Timer) 该字段指定 MAC 停止向 PHY 发送 LPI 模式之后至少需要等待多长时间才能恢复正常发送(以微秒为单位)。TLPIEX 状态位在该定时器到期后置 1。

38.4.15. MAC 中断状态寄存器(ETH_MACISR: 38h)

位域	名称	属性	复位值	描述
31:11	RSV	-	0	保留，必须保持复位值
10	LPIIS	R	0	LPI 中断状态(LPI Interrupt Status) 使能节能型以太网功能时，会针对 MAC 发送器或接收器时进入还是退出 LPI 状态来设置该位。对 ETH_MACLPICSR 寄存器的 TLPIEN 位进行读操作时，该位清零。
9	TIS	R	0	时间戳中断状态（Timestamp Interrupt Status) 如果使能了时间戳功能，则当以下任一条件为真时，该位会置 1： – 系统时间值等于或超出目标时间高位和低位寄存器中指定的值。 – 秒寄存器发生上溢。 – 发生目标时间错误，即，编程的目标时间已结束。 可通过读取 ETH_PTPTSSR 寄存器清零该标志。
8:7	RSV	-	0	保留，必须保持复位值
6	MMCTIS	R	0	MMC 发送中断状态（MMC Transmit Interrupt Status) 只要 ETH_MMCTIR 寄存器中生成中断，此位就为高电平。当此中断寄存器(ETH_MMCTIR)中的所有位都清零时，此位清零
5	MMCRIS	R	0	MMC 接收中断状态（MMC Receive Interrupt Status) 只要 ETH_MMCRIR 寄存器中生成中断，此位就为高电平。当此中断寄存器(ETH_MMCRIR)中的所有位都清零时，此位清零。
4	MMCIS	R	0	MMC 中断状态（MMC Interrupt Status) 只要位 6:5 中的任何一个位被设置为高电平，此位即会被设置为高电平。仅当这两个位均为低电平时，此位才被清零。
3	PMTS	R	0	PMT 中断状态（PMT Interrupt Status) 只要在掉电模式下接收到魔术数据包或局域网唤醒帧，此位即被置 1（详见 ETH_MACPMTCSR 寄存器中的位 5 和 6)。当对 ETH_MACPMTCSR 寄存器位 [6:5]执行读取操作而被清零时，此位也将会清零。
2:0	RSV	-	0	保留，必须保持复位值

38.4.16. MAC 中断屏蔽寄存器(ETH_MACIMR: 3Ch)

位域	名称	属性	复位值	描述
31:11	RSV	-	0	保留，必须保持复位值

10	LPIIM	RW	0	LPI 中断屏蔽(LPI Interrupt Mask) 置 1 后，该位会屏蔽以太网 MAC 中断状态寄存器(ETH_MACISR)中 LPI 中断状态生成的中断信号。
9	TIM	RW	0	时间戳中断屏蔽（Timestamp Interrupt Mask) 置 1 后，该位会屏蔽以太网 MAC 中断状态寄存器(ETH_MACISR)中时间戳中断状态位生成的中断信号。
8:4	RSV	-	0	保留，必须保持复位值
3	PIM			PMT 中断屏蔽（PMT Interrupt Mask) 置 1 后，该位会屏蔽以太网 MAC 中断状态寄存器(ETH_MACISR)中 PMT 中断状态位生成的中断信号。
2:0	RSV	-	0	保留，必须保持复位值

38.4.17. MAC 地址 0 高位寄存器(ETH_MACA0HR: 40h)

MAC 地址 0 高位寄存器可保存站的第一个 6 字节 MAC 地址的高 16 位。在 MII 接口上接收到的第一个 DA 字节与 MAC 地址低位寄存器的 LS 字节（位 [7:0]）相对应。例如，如果在 MII 上接收到 0x112233445566（第一列通道 0 中的 0x11）作为目标地址，则 MAC 地址 0 寄存器 [47:0] 会与 0x665544332211 进行比较。

位域	名称	属性	复位值	描述
31	AE	RO	1	地址使能（Address Enable) 该位始终置 1。
30:16	RSV	-	0	保留，必须保持复位值
15:0	MACA0H	RW	0xFFFF	MAC 地址 0 高位 [47:32] (MAC Address0 High [47:32]) 此字段包含 6 字节 MAC 地址 0 的高 16 位 (47:32)。 MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中。

38.4.18. MAC 地址 0 低位寄存器(ETH_MACA0LR: 44h)

位域	名称	属性	复位值	描述
31:0	MACA0L	RW	0xFFFFFFFF	MAC 地址 0 低位 [31:0] (MAC Address0 Low [31:0]) 此字段包含 6 字节 MAC 地址 0 的低 32 位。 MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制（暂停）帧中。

38.4.19. MAC 地址 1 高位寄存器(ETH_MACA1HR: 48h)

位域	名称	属性	复位值	描述
31	AE	RW	0	地址使能（Address Enable） 0：地址过滤器会忽略用于过滤的地址 1：地址过滤器使用 MAC 地址 1 实现完美过滤。
30	SA	RW	0	源地址（Source Address） 0：将使用 MAC 地址 1 [47:0] 与所接收帧的 DA 字段进行比较。 1：将使用 MAC 地址 1 [47:0] 与所接收帧的 SA 字段进行比较。
29:24	MBC	RW	0	屏蔽字节控制（Mask Byte Control） 这些位是用于比较每个 MAC 地址字节的屏蔽控制位。当将其设为高电平时， MAC 不会将所接收到的 DA 或 SA 的相应字节与 MAC 地址 1 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示： 位 29：ETH_MACA1HR [15:8] 位 28：ETH_MACA1HR [7:0] 位 27：ETH_MACA1LR [31:24] … 位 24：ETH_MACA1LR [7:0] 可通过屏蔽地址的一个或多个字节来过滤一组地址（被称为组地址过滤）。
23:16	RSV	-	0	保留，必须保持复位值
15:0	MACA1H	RW	0xFFFF	MAC 地址 1 高位 [47:32] (MAC Address1 High [47:32]) 此字段包含第二个 6 字节 MAC 地址的高 16 位 (47:32)。

38.4.20. MAC 地址 1 低位寄存器(ETH_MACA1LR: 4Ch)

位域	名称	属性	复位值	描述
31:0	MACA1L	RW	0xFFFFFFFF	MAC 地址 1 低位 [31:0] (MAC Address1 Low [31:0]) 此字段包含 6 字节 MAC 地址 1 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

38.4.21. MAC 地址 2 高位寄存器(ETH_MACA2HR: 50h)

位域	名称	属性	复位值	描述
31	AE	RW	0	地址使能（Address Enable） 0：地址过滤器会忽略用于过滤的地址 1：地址过滤器使用 MAC 地址 2 实现完美过滤。

30	SA	RW	0	源地址（Source Address） 0：将使用 MAC 地址 2 [47:0] 与所接收帧的 DA 字段进行比较。 1：将使用 MAC 地址 2 [47:0] 与所接收帧的 SA 字段进行比较。
29:24	MBC	RW	0	屏蔽字节控制（Mask Byte Control） 这些位是用于比较每个 MAC 地址字节的屏蔽控制位。当将其设为高电平时， MAC 不会将所接收到的 DA 或 SA 的相应字节与 MAC 地址 2 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示： 位 29：ETH_MACA2HR [15:8] 位 28：ETH_MACA2HR [7:0] 位 27：ETH_MACA2LR [31:24] … 位 24：ETH_MACA2LR [7:0] 可通过屏蔽地址的一个或多个字节来过滤一组地址（被称为组地址过滤）。
23:16	RSV	-	0	保留，必须保持复位值
15:0	MACA2H	RW	0xFFFF	MAC 地址 2 高位 [47:32] (MAC Address2 High [47:32]) 此字段包含第三个 6 字节 MAC 地址的高 16 位 (47:32)。

38.4.22. MAC 地址 2 低位寄存器(ETH_MACA2LR: 54h)

位域	名称	属性	复位值	描述
31:0	MACA2L	RW	0xFFFFFFFF	MAC 地址 2 低位 [31:0] (MAC Address2 Low [31:0]) 此字段包含 6 字节 MAC 地址 2 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

38.4.23. MAC 地址 3 高位寄存器(ETH_MACA3HR: 58h)

位域	名称	属性	复位值	描述
31	AE	RW	0	地址使能（Address Enable） 0：地址过滤器会忽略用于过滤的地址 1：地址过滤器使用 MAC 地址 3 实现完美过滤。
30	SA	RW	0	源地址（Source Address） 0：将使用 MAC 地址 3 [47:0] 与所接收帧的 DA 字段进行比较。 1：将使用 MAC 地址 3 [47:0] 与所接收帧的 SA 字段进行比较。

29:24	MBC	RW	0	屏蔽字节控制（Mask Byte Control） 这些位是用于比较每个 MAC 地址字节的屏蔽控制位。当将其设为高电平时， MAC 不会将所接收到的 DA 或 SA 的相应字节与 MAC 地址 3 寄存器的内容进行比较。每个位都用于控制字节的屏蔽，如下所示： 位 29：ETH_MACA3HR [15:8] 位 28：ETH_MACA3HR [7:0] 位 27：ETH_MACA3LR [31:24] … 位 24：ETH_MACA3LR [7:0] 可通过屏蔽地址的一个或多个字节来过滤一组地址（被称为组地址过滤）。
23:16	RSV	-	0	保留，必须保持复位值
15:0	MACA3H	RW	0xFFFF	MAC 地址 3 高位 [47:32] (MAC Address3 High [47:32]) 此字段包含第四个 6 字节 MAC 地址的高 16 位 (47:32)。

38.4.24. MAC 地址 3 低位寄存器(ETH_MACA3LR: 5Ch)

位域	名称	属性	复位值	描述
31:0	MACA3L	RW	0xFFFFFFFF	MAC 地址 3 低位 [31:0] (MAC Address3 Low [31:0]) 此字段包含 6 字节 MAC 地址 3 的低 32 位。如果在初始化过程之后，应用程序未加载此字段的内容，则该内容将不会被定义。

38.4.25. MAC 看门狗超时寄存器(ETH_MACWTR: DCh)

位域	名称	属性	复位值	描述
31:17	RSV	-	0	保留，必须保持复位值
16	PWE	RW	0	可编程看门狗使能(Programmable Watchdog Enable) 当该位置 1 且 ETH_MACCR 寄存器的 WD 位复位时，WTO 字段用作已接收数据包的看门狗超时。当该位清零时，通过设置 ETH_MACCR 寄存器的 WD 和 JE 位来控制已接收数据包的看门狗超时。
15:14	RSV	-	0	保留，必须保持复位值
13:0	WTO	RW	0	看门狗超时(Watchdog Timeout) 当 PWE 位置 1 且 ETH_MACCR 寄存器的 WD 位复位时，该字段用作已接收数据包的看门狗超时。如果接收的数据包的长度超过该字段的值，则此数据包会被终止并被声明为错误数据包。 注：当 PWE 位置 1 时，该字段的值应大于 1522(0x05F2)。否则，IEEE 802.3 指定的带标记的有效数据包会被声明为错误数据包，然后被丢弃。

38.4.26. MMC 控制寄存器(ETH_MMCCR: 100h)

该寄存器配置 MMC 工作模式。CNTRST 位的优先级高于 CNTPRST 位。因此，软件尝试在同一写周期内将这两个位置 1 时，所有计数器都将清零，并且 CNTPRST 位不会置 1。

位域	名称	属性	复位值	描述
31:9	RSV	-	0	保留，必须保持复位值
8	UCDBC	RW	0	针对所丢弃广播数据包更新 MMC 计数器 (Update MMC Counters for Dropped Broadcast Packets) 0：不会针对丢弃的广播数据包更新 MMC 计数器。 1：MAC 会针对因 ETH_MACFFR 寄存器的 DBF 位置 1 而被丢弃的广播数据包更新所有相关 MMC 计数器。
7:6	RSV	-	0	保留，必须保持复位值
5	CNTPRSTLVL	RW	0	全-半预设置 (Full-Half Preset) 0：CNTPRST 位置 1 时，所有 MMC 计数器均预设为几乎一半值。所有八位字节计数器均预设为 0x7FFF_F800（半个 2 KB），所有数据包计数器均预设为 0x7FFF_FFF0（半个 16）。 1：CNTPRST 位置 1 时，所有 MMC 计数器均预设为几乎全值。所有八位字节计数器均预设为 0xFFFF_F800（全 2 KB），所有数据包计数器均预设为 0xFFFF_FFF0（全 16）。
4	CNTPRST	RW	0	计数器预设 (Counters Preset) 该位置 1 时，所有计数器均初始化或预设为几乎全值或几乎一半值，具体取决于 CNTPRSTLVL 位。该位在 1 个时钟周期后自动清零。该位与 CNTPRSTLVL 位一起用于调试和测试中断的有效情况，因为 MMC 计数器变为半-全或全值。
3	CNTFREEZ	RW	0	MMC 计数器冻结 (MMC Counter Freeze) 该位置 1 时，会将所有 MMC 计数器冻结为当前值。该位复位为 0 之前，收到或发送任何数据包 MMC 计数器都不会更新。在此模式下，“读取时复位”位置 1 时，如果读取任何 MMC 计数器，则该计数器也会清零。
2	RSTONRD	RW	0	读取时复位 (Reset on Read) 该位置 1 时，读取 MMC 计数器后，该计数器会复位为零（复位后自清零）。读取最低有效字节通道（位 [7:0]）后，计数器会清零。
1	CNTSTOPRO	RW	0	计数器停止翻转 (Counter Stop Rollover) 该位置 1 时，计数器达到其最大值后，不会返回到零。
0	CNTRST	RW	0	计数器复位 (Counters Reset) 该位置 1 时，所有计数器都将复位。该位在 1 个时钟周期后自动清零。

38.4.27. MMC 接收中断寄存器(ETH_MMCRIR: 104h)

该寄存器保存所有接收统计计数器生成的中断。

MMC 接收中断寄存器保存在发生以下情况时生成的中断：

接收统计计数器达到其最大值的一半（对于 32 位计数器为 0x8000_0000，对于 16 位计数器为 0x8000）。

接收统计计数器超过其最大值（对于 32 位计数器为 0xFFFF_FFFF，对于 16 位计数器为 0xFFFF）。

计数器停止翻转置 1 时，中断将置为有效，但计数器保持为全 1。 MMC 接收中断寄存器为 32 位寄存器。当读取引发中断的各个 MMC 计数器时，会将中断位清零。必须读取相应计数器的最低有效字节通道（位 [7:0]），才能将中断位清零。

位域	名称	属性	复位值	描述
31:18	RSV	-	0	保留，必须保持复位值
17	RGUFIS	R	0	接收的良好单播帧中断状态（Received Good Unicast Frames Interrupt Status） 当接收到的良好单播帧计数器达到其最大值的一半或最大值时，该位置 1。
16:7	RSV	-	0	保留，必须保持复位值
6	RFAEIS	R	0	接收的帧对齐错误中断状态（Received Frames Alignment Error Interrupt Status） 当存在对齐错误的接收帧计数器达到其最大值的一半或最大值时，该位置 1。
5	RFCEIS	R	0	接收的帧 CRC 错误中断状态（Received Frames CRC Error Counter Interrupt Status） 当存在 CRC 错误的接收帧计数器达到其最大值的一半或最大值时，该位置 1。
4:0	RSV	-	0	保留，必须保持复位值

38.4.28. MMC 发送中断寄存器(ETH_MMCTIR: 108h)

该寄存器保存所有发送统计计数器生成的中断。

MMC 发送中断寄存器保存发送统计计数器达到其最大值的一半（对于 32 位计数器为 0x8000_0000，对于 16 位计数器为 0x8000）时生成的中断，以及发送统计计数器超过最大值（对于 32 位计数器为 0xFFFF_FFFF，对于 16 位计数器为 0xFFFF）时生成的中断。

计数器停止翻转置 1 时，中断将置为有效，但计数器保持为全 1。

MMC 发送中断寄存器为 32 位寄存器。当读取引发中断的各个 MMC 计数器时，会将中断位清零。

必须读取相应计数器的最低有效字节通道（位 [7:0]），才能将中断位清零。

位域	名称	属性	复位值	描述
31:22	RSV	-	0	保留，必须保持复位值
21	TGFIS	R	0	发送良好帧中断状态（Transmit Good Frames Interrupt Status） 当发送的良好帧计数器达到其最大值的一半或最大值时，该位置 1。
20:16	RSV	-	0	保留，必须保持复位值
15	TMCGFCIS	R	0	发送多次冲突良好帧计数器中断状态（Transmit Multiple Collision Good Frames Counter Interrupt Status） 当发送的多次冲突良好帧计数器达到最大值的一半或最大值时，该位置 1。

14	TSCGFCIS	R	0	发送单次冲突良好帧计数器中断状态（Transmit Single Collision Good Frames Counter Interrupt Status) 当发送的单次冲突良好帧计数器达到最大值的一半或最大值时，该位置 1。
13:0	RSV	-	0	保留，必须保持复位值

38.4.29. MMC 接收中断屏蔽寄存器(ETH_MMCRIMR: 10Ch)

MMC 接收中断屏蔽寄存器用于保存接收统计计数器达到其最大值的一半或最大值时所生成中断的屏蔽。

位域	名称	属性	复位值	描述
31:18	RSV	-	0	保留，必须保持复位值
17	RGUFIM	RW	0	接收的良好单播帧中断屏蔽（Received Good Unicast Frames Interrupt Mask) 当接收到的良好单播帧计数器达到其最大值的一半或最大值时，将该位置 1 会屏蔽中断。
16:7	RSV	-	0	保留，必须保持复位值
6	RFAEIM	RW	0	接收的帧对齐错误中断屏蔽（Received Frames Alignment Error Interrupt Mask) 当存在对齐错误的接收帧计数器达到其最大值的一半或最大值时，将该位置 1 会屏蔽中断。
5	RFCEIM	RW	0	接收的帧 CRC 错误中断屏蔽（Received Frames CRC Error Interrupt Mask) 当存在 CRC 错误的接收帧计数器达到其最大值的一半时，将该位置 1 会屏蔽中断。
4:0	RSV	-	0	保留，必须保持复位值

38.4.30. MMC 发送中断屏蔽寄存器(ETH_MMCTIMR: 110h)

MMC 发送中断屏蔽寄存器用于保存发送统计计数器达到其最大值的一半或最大值时所生成中断的屏蔽。

位域	名称	属性	复位值	描述
31:22	RSV	-	0	保留，必须保持复位值
21	TGFIM	RW	0	发送的良好帧中断屏蔽（Transmit Good Frames Interrupt Mask) 当发送的良好帧计数器达到其最大值的一半或最大值时，将该位置 1 会屏蔽中断。
20:16	RSV	-	0	保留，必须保持复位值
15	TMCGFCIM			发送多次冲突良好帧计数器中断状态（Transmit Multiple Collision Good Frames Counter Interrupt Mask) 当发送的多次冲突良好帧计数器达到最大值的一半或最大值时，将该位置 1 会屏蔽中断。

14	TSCGFCIM			发送单次冲突良好帧计数器中断状态（Transmit Single Collision Good Frames Counter Interrupt Mask) 当发送的单次冲突良好帧计数器达到最大值的一半或最大值时，将该位置 1 会屏蔽中断。
13:0	RSV	-	0	保留，必须保持复位值

38.4.31. MMC 在单个冲突后发送的良好帧计数器寄存器(ETH_MMCTGFSCCR: 14Ch)

该寄存器包含在半双工模式下于单个冲突后成功发送的帧的数量。

位域	名称	属性	复位值	描述
31:0	TGFSCC	RO	0	单个冲突后发送的良好帧计数器（Transmitted Good Frames Single Collision Counter) 单个冲突后发送的良好帧数量。

38.4.32. MMC 在多个冲突后发送的良好帧计数器寄存器(ETH_MMCTGFMCCR: 150h)

位域	名称	属性	复位值	描述
31:0	TGFMCC	RO	0	多个冲突后发送的良好帧计数器（Transmitted Good Frames Multiple Collisions Counter) 多个冲突后发送的良好帧数量。

38.4.33. MMC 发送的良好帧计数器寄存器(ETH_MMCTGFCR: 168h)

位域	名称	属性	复位值	描述
31:0	TGFC	RO	0	发送的良好帧计数器（Transmitted Good Frames Counter) 发送的良好帧数量。

38.4.34. MMC 接收带有 CRC 错误帧计数器寄存器(ETH_MMCRFCECR: 194h)

位域	名称	属性	复位值	描述
31:0	RFCEC	RO	0	接收的帧 CRC 错误计数器（Received Frames CRC Error Counter) 接收到带有 CRC 错误的帧数量。

38.4.35. MMC 接收带有对齐错误帧计数器寄存器(ETH_MMCRFAECR: 198h)

位域	名称	属性	复位值	描述
31:0	RFAEC	RO	0	接收的帧对齐错误计数器（Received Frames Alignment Error Counter） 接收到带有对齐错误的帧数量。

38.4.36. MMC 接收的良好单播帧计数器寄存器(ETH_MMCRGUFCR: 1C4h)

位域	名称	属性	复位值	描述
31:0	RGUFC	RW	0	接收的良好单播帧计数器（Received Good Unicast Frames Counter） 接收到的良好单播帧数量。

38.4.37. MAC L3 和 L4 控制 0 寄存器(ETH_MACL3L4C0R: 400h)

位域	名称	属性	复位值	描述
31:22	RSV	-	0	保留，必须保持复位值
21	L4DPIM0	RW	0	第 4 层目标端口反向匹配使能(Layer 4 Destination Port Inverse Match Enable) 该位置 1 时，将使能第 4 层目标端口号字段以进行反向匹配。该位复位时，将使能第 4 层目标端口号字段以进行完美匹配。 只有在 L4DPM0 位置为高电平时，该位才有效。
20	L4DPM0	RW	0	第 4 层目标端口匹配使能(Layer 4 Destination Port Match Enable) 该位置 1 时，将使能第 4 层目标端口字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 4 层目标端口字段
19	L4SPIM0	RW	0	第 4 层源端口反向匹配使能(Layer 4 Source Port Inverse Match Enable) 该位置 1 时，将使能第 4 层源端口号字段以进行反向匹配。该位复位时，将使能第 4 层源端口号字段以进行完美匹配。 只有在 L4SPM0 位置为高电平时，该位才有效。
18	L4SPM0	RW	0	第 4 层源端口反向匹配使能(Layer 4 Source Port Match Enable) 该位置 1 时，将使能第 4 层源端口字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 4 层源端口字段
17	RSV	-	0	保留，必须保持复位值
16	L4PEN0	RW	0	第 4 层协议使能 该位置 1 时，将使用 UDP 数据包的源端口号和目标端口号字段进行匹配。该位复位时，将使用 TCP 数据包的源端口号和目标端口号字段进行匹配。 只有当 L4SPM0 或 L4DPM0 位置 1 时，才能完成第 4 层匹配。

15:11	L3HDBM0	RW	0	第 3 层 IP DA 高位匹配(Layer 3 IP DA Higher Bits Match) IPv4 数据包： 该字段包含在 IPv4 数据包中匹配的 IP 目标地址的高位数目。下面对该字段的各个值进行了说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 31：屏蔽除了 MSB 之外的所有位 IPv6 数据包： 该字段的位[12:11]对应于 L3HSBM0 的位[6:5]，表示在 IPv6 数据包中被屏蔽的 IP 源地址或目标地址的低位数目。下面 L3HDBM0[1:0]和 L3HSBM0 位的各个值进行了说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 127：屏蔽除了 MSB 之外的所有位 只有在 L3DAM0 或 L3SAM0 位置 1 时，该字段才有效。
10:6	L3HSBM0	RW	0	第 3 层 IP SA 高位匹配(Layer 3 IP SA Higher Bits Match) IPv4 数据包： 该字段包含为在 IPv4 数据包中匹配而被屏蔽的 IP 源地址的低位数目。下面对该字段的各个值进行说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 31：屏蔽除了 MSB 之外的所有位 IPv6 数据包： 该字段包含 L3HSBM0 的位[4:0]。这些位表示 IPv6 数据包中匹配的 IP 源地址或目标地址的高位数目。只有在 L3DAM0 或 L3SAM0 位置为高电平时，该字段才有效。
5	L3DAIM0	RW	0	第 3 层 IP DA 反向匹配使能(Layer 3 IP Inverse Match Enable) 该位置 1 时，将使能第 3 层 IP 目标地址字段以进行反向匹配。该位复位时。将使能第 3 层 IP 目标地址字段已进行完美匹配。 只有在 L3DAM0 位置为高电平时，该位才有效。
4	L3DAM0			第 3 层 IP DA 匹配使能(Layer 3 IP DA Match Enable) 该位置 1 时，将使能第 3 层 IP 目标地址字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 3 层 IP 目标地址字段。 注：LP3EN0 位置 1 时，应将该位或 L3SAM0 位置 1，这样可以检查 IPv6 DA 或 SA 以进行过滤。

3	L3SAIM0			第 3 层 IP SA 反向匹配使能(Layer 3 IP SA Inverse Match Enable) 该位置 1 时，将使能第 3 层 IP 源地址字段进行反向匹配。该位复位时，将使能第 3 层 IP 源地址字段进行完美匹配。 只有在 L3SAM0 位置 1 时，该位才有效。
2	L3SAM0	RW	0	第 3 层 IP SA 匹配使能(Layer 3 IP SA Match Enable) 该位置 1 时，将使能第 3 层 IP 源地址字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 3 层 IP 源地址字段 注：L3PEN0 位置 1 时，应将该位或 L3DAM0 位置 1。这样可以检查 IPv6 SA 或 DA 已进行过滤。
1	RSV	-	0	保留，必须保持复位值
0	L3PEN0	RW	0	第 3 层协议使能(Layer 3 Protocol Enable) 该位置 1 时，将为 IPv6 数据包使能第 3 层 IP 源地址或目标地址匹配。该位复位时，将为 IPv4 数据包使能第 3 层 IP 源地址或目标地址匹配。 只有当 L3SAM0 或 L3DAM0 位置 1 时，才能完成第 3 层匹配。

38.4.38. MAC L4 地址过滤器 0 寄存器(ETH_MACL4A0R: 404h)

位域	名称	属性	复位值	描述
31:16	L4DP0	RW	0	第 4 层目标端口号字段(Layer 4 Destination Port Number Field) ETH_MACL3L4C0R 寄存器中的 L4PEN0 位复位且 L4DPM0 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 TCP 目标端口号字段进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L4PEN0 和 L4DPM0 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 UDP 目标端口号字段进行匹配的值。
15:0	L4SP0	RW	0	第 4 层源端口号字段(Layer 4 Source Port Number Field) ETH_MACL3L4C0R 寄存器中的 L4PEN0 位复位且 L4DPM0 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 TCP 源端口号字段进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L4PEN0 和 L4DPM0 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 UDP 源端口号字段进行匹配的值。

38.4.39. MAC L3 地址 0 过滤器 0 寄存器(ETH_MACL3A00R: 410h)

位域	名称	属性	复位值	描述
31:0	L3A00	RW	0	第 3 层地址地址 0 字段(Layer 3 Address 0 Field) ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3SAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[31:0]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3DAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[31:0]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 位复位且 L3SAM0 位置 1 时，该字段包含要与 IPv4 数据包中 IP 的源地址字段进行匹配的值。

38.4.40. MAC L3 地址 1 过滤器 0 寄存器(ETH_MACL3A10R: 414h)

位域	名称	属性	复位值	描述
31:0	L3A10	RW	0	第 3 层地址地址 1 字段(Layer 3 Address 1 Field) ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3SAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[63:32]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3DAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[63:32]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 位复位且 L3DAM0 位置 1 时，该字段包含要与 IPv4 数据包中 IP 的目标地址字段进行匹配的值。

38.4.41. MAC L3 地址 2 过滤器 0 寄存器(ETH_MACL3A20R: 418h)

位域	名称	属性	复位值	描述
31:0	L3A20	RW	0	第 3 层地址地址 2 字段(Layer 3 Address 2 Field) ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3SAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[95:64]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3DAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[95:64]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 位复位时，不适用该字段。

38.4.42. MAC L3 地址 3 过滤器 0 寄存器(ETH_MACL3A30R: 41Ch)

位域	名称	属性	复位值	描述
31:0	L3A30	RW	0	第 3 层地址地址 3 字段(Layer 3 Address 3 Field) ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3SAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[127:96]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 和 L3DAM0 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[127:96]进行匹配的值。 ETH_MACL3L4C0R 寄存器中的 L3PEN0 位复位时，不使用该字段。

38.4.43. MAC L3 和 L4 控制 1 寄存器(ETH_MACL3L4C1R: 430h)

位域	名称	属性	复位值	描述
31:22	RSV	-	0	保留，必须保持复位值

21	L4DPIM1	RW	0	第 4 层目标端口反向匹配使能(Layer 4 Destination Port Inverse Match Enable) 该位置 1 时，将使能第 4 层目标端口号字段以进行反向匹配。该位复位时，将使能第 4 层目标端口号字段以进行完美匹配。 只有在 L4DPM1 位置为高电平时，该位才有效。
20	L4DPM1	RW	0	第 4 层目标端口匹配使能(Layer 4 Destination Port Match Enable) 该位置 1 时，将使能第 4 层目标端口字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 4 层目标端口字段
19	L4SPIM1	RW	0	第 4 层源端口反向匹配使能(Layer 4 Source Port Inverse Match Enable) 该位置 1 时，将使能第 4 层源端口号字段以进行反向匹配。该位复位时，将使能第 4 层源端口号字段以进行完美匹配。 只有在 L4SPM1 位置为高电平时，该位才有效。
18	L4SPM1	RW	0	第 4 层源端口反向匹配使能(Layer 4 Source Port Match Enable) 该位置 1 时，将使能第 4 层源端口字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 4 层源端口字段
17	RSV	-	0	保留，必须保持复位值
16	L4PEN1	RW	0	第 4 层协议使能 该位置 1 时，将使用 UDP 数据包的源端口号和目标端口号字段进行匹配。该位复位时，将使用 TCP 数据包的源端口号和目标端口号字段进行匹配。 只有当 L4SPM1 或 L4DPM1 位置 1 时，才能完成第 4 层匹配。
15:11	L3HDBM1	RW	0	第 3 层 IP DA 高位匹配(Layer 3 IP DA Higher Bits Match) IPv4 数据包： 该字段包含在 IPv4 数据包中匹配的 IP 目标地址的高位数目。下面对该字段的各个值进行了说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 31：屏蔽除了 MSB 之外的所有位 IPv6 数据包： 该字段的位[12:11]对应于 L3HSBM1 的位[6:5]，表示在 IPv6 数据包中被屏蔽的 IP 源地址或目标地址的低位数目。下面 L3HDBM1[1:0]和 L3HSBM1 位的各个值进行了说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 127：屏蔽除了 MSB 之外的所有位 只有在 L3DAM1 或 L3SAM1 位置 1 时，该字段才有效。

10:6	L3HSBM1	RW	0	第 3 层 IP SA 高位匹配(Layer 3 IP SA Higher Bits Match) IPv4 数据包： 该字段包含为在 IPv4 数据包中匹配而被屏蔽的 IP 源地址的低位数目。下面对该字段的各个值进行说明： 0：未屏蔽任何位 1：屏蔽 LSB[0] 2：屏蔽 LSB[1:0] … 31：屏蔽除了 MSB 之外的所有位 IPv6 数据包： 该字段包含 L3HSBM1 的位[4:0]。这些位表示 IPv6 数据包中匹配的 IP 源地址或目标地址的高位数目。只有在 L3DAM1 或 L3SAM1 位置为高电平时，该字段才有效。
5	L3DAIM1	RW	0	第 3 层 IP DA 反向匹配使能(Layer 3 IP Inverse Match Enable) 该位置 1 时，将使能第 3 层 IP 目标地址字段以进行反向匹配。该位复位时。将使能第 3 层 IP 目标地址字段已进行完美匹配。 只有在 L3DAM1 位置为高电平时，该位才有效。
4	L3DAM1			第 3 层 IP DA 匹配使能(Layer 3 IP DA Match Enable) 该位置 1 时，将使能第 3 层 IP 目标地址字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 3 层 IP 目标地址字段。 注：LP3EN1 位置 1 时，应将该位或 L3SAM1 位置 1，这样可以检查 IPv6 DA 或 SA 以进行过滤。
3	L3SAIM1			第 3 层 IP SA 反向匹配使能(Layer 3 IP SA Inverse Match Enable) 该位置 1 时，将使能第 3 层 IP 源地址字段进行反向匹配。该位复位时，将使能第 3 层 IP 源地址字段进行完美匹配。 只有在 L3SAM1 位置 1 时，该位才有效。
2	L3SAM1	RW	0	第 3 层 IP SA 匹配使能(Layer 3 IP SA Match Enable) 该位置 1 时，将使能第 3 层 IP 源地址字段以进行匹配。该位复位时，MAC 将忽略用于匹配的第 3 层 IP 源地址字段 注：L3PEN1 位置 1 时，应将该位或 L3DAM1 位置 1。这样可以检查 IPv6 SA 或 DA 已进行过滤。
1	RSV	-	0	保留，必须保持复位值
0	L3PEN1	RW	0	第 3 层协议使能(Layer 3 Protocol Enable) 该位置 1 时，将为 IPv6 数据包使能第 3 层 IP 源地址或目标地址匹配。该位复位时，将为 IPv4 数据包使能第 3 层 IP 源地址或目标地址匹配。 只有当 L3SAM1 或 L3DAM1 位置 1 时，才能完成第 3 层匹配。

38.4.44. MAC L4 地址过滤器 1 寄存器(ETH_MACL4A1R: 434h)

位域	名称	属性	复位值	描述

31:16	L4DP1	RW	0	第 4 层目标端口号字段(Layer 4 Destination Port Number Field) ETH_MACL3L4C1R 寄存器中的 L4PEN1 位复位且 L4DPM1 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 TCP 目标端口号字段进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L4PEN1 和 L4DPM1 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 UDP 目标端口号字段进行匹配的值。
15:0	L4SP1	RW	0	第 4 层源端口号字段(Layer 4 Source Port Number Field) ETH_MACL3L4C1R 寄存器中的 L4PEN1 位复位且 L4DPM1 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 TCP 源端口号字段进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L4PEN1 和 L4DPM1 位置 1 时，该字段包含要与 IPv4 或 IPv6 数据包中的 UDP 源端口号字段进行匹配的值。

38.4.45. MAC L3 地址 0 过滤器 1 寄存器(ETH_MACL3A01R: 440h)

位域	名称	属性	复位值	描述
31:0	L3A01	RW	0	第 3 层地址 0 字段(Layer 3 Address 0 Field) ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3SAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[31:0]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3DAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[31:0]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 位复位且 L3SAM1 位置 1 时，该字段包含要与 IPv4 数据包中 IP 的源地址字段进行匹配的值。

38.4.46. MAC L3 地址 1 过滤器 1 寄存器(ETH_MACL3A11R: 444h)

位域	名称	属性	复位值	描述
31:0	L3A11	RW	0	第 3 层地址 1 字段(Layer 3 Address 1 Field) ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3SAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[63:32]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3DAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[63:32]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 位复位且 L3SAM1 位置 1 时，该字段包含要与 IPv4 数据包中 IP 的源地址字段进行匹配的值。

38.4.47. MAC L3 地址 2 过滤器 1 寄存器(ETH_MACL3A21R: 448h)

位域	名称	属性	复位值	描述

31:0	L3A21	RW	0	第 3 层地址地址 2 字段(Layer 3 Address 2 Field) ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3SAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[95:64]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3DAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[95:64]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 位复位时，不适用该字段。

38.4.48. MAC L3 地址 3 过滤器 1 寄存器(ETH_MACL3A31R: 44Ch)

位域	名称	属性	复位值	描述
31:0	L3A31	RW	0	第 3 层地址地址 3 字段(Layer 3 Address 3 Field) ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3SAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 源地址字段的位[127:96]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 和 L3DAM1 位置 1 时，该字段包含要与 IPv6 数据包中 IP 目标地址字段的位[127:96]进行匹配的值。 ETH_MACL3L4C1R 寄存器中的 L3PEN1 位复位时，不使用该字段。

38.4.49. MAC VLAN 标记包含或替换寄存器(ETH_MACVTIRR: 584h)

位域	名称	属性	复位值	描述
31:20	RSV	-	0	保留，必须保持复位值
19	CSVL	RW	0	C-VLAN 或 S-VLAN(C-VLAN or S-VLAN) 该位置 1 时，会在所发送数据包的第 13 个和第 14 个字节中插入或替换 S-VLAN 类型(0x88A8)。该位复位时，会在所发送数据包的第 13 个和第 14 个字节中插入或替换 C-VLAN 类型(0x8100)
18	VLP	RW	0	VLAN 优先级控制(VLAN Priority Control) 该位置 1 时，使用控制位[17:16]进行 VLAN 删除、插入或替换。该位复位时，将会忽略位[17:16]
17:16	VLC	RW	0	发送数据包中的 VLAN 标记控制(VLAN Tag Control in Transmit Packets) 00：不进行 VLAN 标记删除、插入和替换 01：VLAN 标记删除。MAC 将删除所有带 VLAN 标记的已发送数据包的 VLAN 类型(字节 13 和 14)以及 VLAN 标记(字节 15 和 16) 10：VLAN 标记插入。在字节 13 和 14 中插入类型值(0x8100 或 0x88a8)后，MAC 会在数据包的字节 15 和 16 中插入 VLT。无论已发送数据包是否已具有 VLAN 标记，都会对它们全部进行该操作。 11：VLAN 标记替换。MAC 替换所有 VLAN 类型已发送数据包的字节 15 和 16 中的 VLT(字节 13 和 14 的值为 0x8100 或 0x88a8)

15:0	VLT	RW	0	发送数据包的 VLAN 标记(VLAN Tag for Transmit Packets) 该字段包含要插入或替换的 VLAN 标记的值。只有在发送线无效或处于初始化阶段时，才能更改该值。 在 VLAN 标记中，位[15:13]为用户优先级字段，位 12 为 CFI/DEI 字段，位[11:0]为 VID 字段。下面对该字段的各个位进行了说明： 位[15:13]：用户优先级 位 12：标准格式指示符(CFI)或丢弃合法指示符(DEI) 位[11:0]：VLAN 标记的 VLAN 标识符(VID)字段。

38.4.50. MAC VLAN 散列表寄存器(ETH_MACVHTR: 588h)

位域	名称	属性	复位值	描述
31:16	RSV	-	0	保留，必须保持复位值
15:0	VLHT	RW	0	VLAN 散列表(VLAN Hash Table) 该字段包含 16 位 VLAN 散列表

38.4.51. PTP 时间戳控制寄存器(ETH_PTPTSCR: 700h)

该寄存器控制着接收器内系统时间的生成以及为 PTP 数据包加盖时间戳。

位域	名称	属性	复位值	描述
31:29	RSV	-	0	保留，必须保持复位值
28	ASEN3	RW	0	辅助快照 3 使能(Auxiliary Snapshot 3 Enable) 该位控制辅助快照触发 3 的捕获。该位置 1 时，会使能 ptp_aux_trig_i[3]输入上的事件辅助快照。该位复位时，此输入上的事件将被忽略。
27	ASEN2	RW	0	辅助快照 2 使能(Auxiliary Snapshot 2 Enable) 该位控制辅助快照触发 2 的捕获。该位置 1 时，会使能 ptp_aux_trig_i[2]输入上的事件辅助快照。该位复位时，此输入上的事件将被忽略。
26	ASEN1	RW	0	辅助快照 1 使能(Auxiliary Snapshot 1 Enable) 该位控制辅助快照触发 1 的捕获。该位置 1 时，会使能 ptp_aux_trig_i[1]输入上的事件辅助快照。该位复位时，此输入上的事件将被忽略。
25	ASEN0	RW	0	辅助快照 0 使能(Auxiliary Snapshot 0 Enable) 该位控制辅助快照触发 0 的捕获。该位置 1 时，会使能 ptp_aux_trig_i[0]输入上的事件辅助快照。该位复位时，此输入上的事件将被忽略。
24	ASFC	RW	0	辅助快照 FIFO 清除(Auxiliary Snapshot FIFO Clear) 该位置 1 时，会复位辅助快照 FIFO 的指针。当指针复位并且 FIFO 为空时，该位清零。该位为高电平时，辅助快照存储在 FIFO 中。

23:19	RSV	-	0	保留，必须保持复位值
18	EMAFPFF	RW	0	使能 MAC 地址来过滤 PTP 帧 (Enable MAC Address For PTP Frame Filtering) 该位被置起后，当通过以太网直接发送 PTP 时，目的 MAC 地址（与任意一个 MAC 地址寄存器匹配）将用于过滤 PTP 帧。
17:16	SPPFTS	RW	0	选择需要拍摄快照的 PTP 数据包 (Select PTP Packets For Taking Snapshot) 00：普通时钟 01：边界时钟 10：端对端透明时钟 11：点对点透明时钟 这些位与位 15 和位 14 共同决定需要拍摄快照的 PTP 数据包类型组。时间戳快照对寄存器位的相依性表格中给出了相应编码。
15	ESFMRTM	RW	0	使能与主节点相关的消息的快照拍摄 (Enable Snapshot For Message Relevant To Master) 0：对与从节点相关的消息进行拍照。 1：对与主节点相关的消息进行拍照。
14	ETSFEM	RW	0	使能事件消息的时间戳拍照 (Enable Timestamp Snapshot For Event Messages) 0：时间戳拍照适用于除了 Announce、 Management 和 Signaling 以外的所有消息。 1：时间戳拍照仅用于事件消息(SYNC、Delay_Req、 Pdelay_Req 或 Pdelay_Resp)。
13	EPPFSIP4U	RW	1	使能处理经由 IPv4-UDP 发送的 PTP 帧 (Enable Processing of PTP Frames Sent over IPv4-UDP) 0：MAC 将忽略经由 UDP-IPv4 数据包发送的 PTP。 1：MAC 接收器将处理经由 IPv4 数据包封装在 UDP 内的 PTP 数据包。
12	EPPFSIP6U	RW	0	使能处理经由 IPv6-UDP 发送的 PTP 帧(Enable Processing of PTP Frames Sent over IPv6-UDP) 0：MAC 将忽略经由 UDP-IPv6 数据包发送的 PTP。 1：MAC 接收器将处理经由 IPv6 数据包封装在 UDP 内的 PTP 数据包。
11	EPPEF	RW	0	使能处理以太网帧上的 PTP(Enable Processing of PTP over Ethernet Frames) 0：MAC 会忽略以太网上的 PTP。 1：MAC 接收器将处理直接封装在以太网帧内的 PTP 数据包。
10	TSPTPPSV2E	RW	0	监听版本 2 格式时间戳 PTP 数据包使能 (Timestamp PTP Packet Snooping for Version2 format Enable) 0：使用版本 1 格式对 PTP 数据包进行监听。 1：使用版本 2 格式对 PTP 数据包进行监听。

9	TSR	RW	0	时间戳亚秒翻转：数字或二进制翻转控制（Timestamp Subsecond Rollover: digital or binary rollover control) 0：亚秒寄存器的翻转值将达到 0x7FFF FFFF。 1：如果亚秒计数器达到值 0x3B9A C9FF（十进制 999 999 999）并递增了时间戳（高位）秒数，则时间戳低位寄存器将翻转。 亚秒增量必须根据 PTP 参考时钟频率和此位的值正确进行编程。
8	TSARFE	RW	0	所有接收帧的时间戳快照使能(Timestamp Snapshot for All Received Frames Enable) 当此位置 1 时，时间戳快照会针对内核所接收的全部帧处于使能状态。
7:6	RSV	-	0	保留，必须保持复位值
5	TARU	RW	0	时间戳加数寄存器更新（Timestamp Addend Register Update) 当此位置 1 时，时间戳加数寄存器的内容会被更新到 PTP 进行精密校准。更新结束时此位会清零。此寄存器位必须在读为零后，才可以将其置 1。
4	TITE	RW	0	时间戳中断触发使能（Timestamp Interrupt Trigger Enable) 当此位置 1 时，如果系统时间大于在目标时间寄存器中写入的值，将产生时间戳中断。产生时间戳触发中断时，此位会清零。
3	TU	RW	0	时间戳更新（Timestamp Update) 该位被置起后，系统时间将根据系统时间-秒更新寄存器和系统时间-纳秒更新寄存器的设定值进行更新（即加或减）。在被更新之前，该位应该为 0。 硬件完成更新后，该位被复位。
2	TI	RW	0	时间戳初始化（Timestamp Initialize) 该位被置起后，将根据系统时间-秒更新寄存器和系统时间-纳秒更新寄存器的设定值来初始化（即覆盖）系统时间。在被更新之前，该位应该为 0。 完成初始化后，该位被复位。
1	TFCU	RW	0	时间戳精密更新或粗略更新(Timestamp Fine or Coarse Update) 0：使用粗略方法更新系统时间戳。 1：使用精密方法更新系统时间戳。
0	TE	RW	0	时间戳使能（Timestamp Enable) 该位被置起后，将会发送帧和接收帧添加时间戳。被关闭后，不再为发送帧和接收帧添加时间戳，时间戳生成器也会被挂起。被使能后，需要初始化时间戳（系统时间）。在接收端，只有当该位被置起后， MAC 才会处理 1588 帧。

表 1 指示根据 ETH_PTPTSCR 寄存器中的[17:14]字段选择拍摄快照的 PTP 消息。

表 38-9.时间戳快照对寄存器位的相依性

SPPFTS Bits 17:16	ESFMRTM Bit 15	ETSFEM Bit14	PTP 消息
00	x	0	SYNC,Follow_Up,Delay_Req,Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req

01	x	0	SYNC,Follow_Up,Delay_Req,Delay_Resp, Pdelay_Req,Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC,Pdelay_Req,Pdelay_Resp
01	1	1	Delay_Req,Pdelay_Req,Pdelay_Resp
10	x	x	SYNC,Delay_Req
11	x	x	Pdelay_Req,Pdelay_Resp

38.4.52. PTP 亚秒递增寄存器(ETH_PTPSSIR: 704h)

在粗略更新模式下，每个 HCLK 时间周期都会将该寄存器的值添加到系统时间。在精细更新模式下，每当累加器产生溢出时，就会向系统时间添加该寄存器的值。

位域	名称	属性	复位值	描述
31:8	RSV	-	0	保留，必须保持复位值
7:0	SSIV	RW	0	亚秒递增值（Sub-Second Increment Value) 该位域的设定值会在每个时钟周期(HCLK)与亚秒寄存器的值累加。例如，如果 PTP 时钟是 50 MHz (周期 20 ns)，当系统时间纳秒寄存器精度为 1 ns[通过在以太网 PTP 时间戳控制寄存器(ETH_PTPTSCR)中设置 9(TSR)]时，则需要将这些位的值设为 20 (0x14)。当 TSR 被清除后，纳秒寄存器分辨精度为 ~0.465ns。此时，需要将这些位的值设为 43 (0x2B)，即 20ns/0.465。

38.4.53. PTP 时间戳高位寄存器(ETH_PTPTSHR: 708h)

位域	名称	属性	复位值	描述
31:0	TS	RO	0	时间戳秒（Timestamp Second) 这些位域的值表示的是由 MAC 维护的当前系统时间的秒值。

38.4.54. PTP 时间戳低位寄存器(ETH_PTPTSLR: 70Ch)

位域	名称	属性	复位值	描述
31	RSV	-	0	保留，必须保持复位值
30:0	TSS	RO	0	时间戳亚秒（Timestamp Sub Seconds） 这个位域的值表示的是亚秒时间，精度为 0.46 ns。 将以太网 PTP 时间戳控制寄存器(ETH_PTPTSCR)）的位 9(TSR)置起后，每个位代表 1ns，所编程的值不能超过 0x3B9A_C9FF。

38.4.55. PTP 时间戳高位更新寄存器(ETH_PTPTSHUR: 710h)

位域	名称	属性	复位值	描述
31:0	TSU	RW	0	时间戳秒更新（Timestamp Second Update） 这个位域表示的是将被初始化或者将被添加到系统时间的秒时间值。

38.4.56. PTP 时间戳低位更新寄存器(ETH_PTPTSLUR: 714h)

位域	名称	属性	复位值	描述
31	AST	RW	0	时间的加与减（Add or Subtract Time） 该位被置起后，时间值将与更新寄存器的值相减。被复位后，时间值将与更新寄存器的值相加。
30:0	TSSU	R	0	时间戳亚秒（Timestamp Sub Seconds Update） 这个位域的值表示的是亚秒时间，精度为 0.46 ns。 将以太网 PTP 时间戳控制寄存器（ETH_PTPTSCR）的位 9(TSR)置起后，每个位代表 1ns，所编程的值不能超过 0x3B9A_C9FF。

38.4.57. PTP 时间戳加数寄存器(ETH_PTPTSAR: 718h)

位域	名称	属性	复位值	描述
31:0	TAR	RW	0	时间戳加数寄存器（Timestamp Addend Register） 这个位域表示的是为了实现时间同步，将要被添加到累加器的 32 位时间值。

38.4.58. PTP 目标时间高位寄存器(ETH_PTPTTHR: 71Ch)

位域	名称	属性	复位值	描述
31:0	TTSR	RW	0	目标时间秒寄存器（Target Time Seconds Register） 该寄存器存放着秒时间值。当时间戳值等于或超过这两个目标时间戳寄存器的值时， MAC 会根据 PPS 控制寄存器的位[6：5]，开始或中断 PPS 信号输出，并产生中断（如果已使能）

38.4.59. PTP 目标时间低位寄存器(ETH_PTPTTLR: 720h)

位域	名称	属性	复位值	描述

31	TTRB	RW	0	目标时间寄存器忙(Target Time Register Busy) 将 ETH_MACPPSCR 寄存器中的 PPSCMD0 字段编程为 010 或 011 时，MAC 会将该位置 1。将 PPSCMD0 字段编程为 010 或 011 时，会命令 MAC 将目标时间寄存器同步到 PTP 时钟域。在将目标时间寄存器同步到 PTP 时钟域之后，MAC 会将该位清零。当该位为 1 时，应用程序不得更新目标时间寄存器。否则，先前编程的时间同步会损坏。
30:0	TTLR	RW	0	目标时间戳低寄存器（Target Timestamp Low Register) 该寄存器存放着纳秒时间值(有符号的)。当时间戳的值等于这两个目标时间戳寄存器的值时， MAC 会根据 PPS 控制寄存器的 TRGTMODSEL0(位[6：5])，开始或中断 PPS 信号输出，并产生中断（如果已使能）当以太网 PTP 时间戳控制寄存器(ETH_PTPTSCR)的位 9(TSSR)被置起时，该位域的值不能超过 0x3B9A_C9FF。PPS 信号输出的实际开始时间或中断时间可能会有高达 1 个亚秒增量值的误差。

38.4.60. PTP 时间戳状态寄存器(ETH_PTPTSSR: 728h)

位域	名称	属性	复位值	描述
31:28	RSV	-	0	保留，必须保持复位值
27:25	ATSNS	RO	0	辅助时间戳快照数(Number of Auxiliary Timestamp Snapshots) 该字段指示 FIFO 中可用的快照数。值等于 FIFO 的深度 4 即表示 FIFO 已满。当辅助快照 FIFO 清除位置 1 时，这些位清零。
24	ATSTM	RO	0	辅助时间戳快照触发未命中(Auxiliary Timestamp Snapshot Trigger Missed) 辅助时间戳快照 FIFO 已满并且设置了外部触发时，该位置 1。这表示最新快照未存储在 FIFO 中。
23:20	RSV	-	0	保留，必须保持复位值
19:16	ATSTI	RO	0	辅助时间戳快照触发标识符(Auxiliary Timestamp Snapshot Trigger Identifier) 这些位标识辅助快照寄存器中的时间戳所适用的辅助触发输入。多个位同时置 1 时，即表示相应的辅助触发采用相同时钟进行采样。只有辅助快照数超过一个时，这些位才适用。为每个触发分配一个位： 位 16：辅助触发 0 位 17：辅助触发 1 位 18：辅助触发 2 位 19：辅助触发 3 软件可以读取该寄存器以找到在获取时间戳时设置的触发。
15:4	RSV	-	0	保留，必须保持复位值
3	TTTE	RO	0	时间戳目标时间错误(Timestamp Target Time Error) 在 ETH_PTPTTHR 和 ETH_PTPTTLR 寄存器中编程的最新目标时间结束后，该位置 1。应用程序读取该位时会将其清零。
2	ATTS	RO	0	辅助时间戳触发快照(Auxiliary Timestamp Trigger Snapshot) 将辅助快照写入 FIFO 时，该位置为高电平。

1	TTTR	RO	0	达到时间戳目标时间（Timestamp Target Time Reached） 被置起后，该位表示系统时间大于或等于目标时间秒寄存器和目标时间纳秒寄存器的设定值。
0	TSO	RO	0	时间戳秒溢出（Timestamp Seconds Overflow） 被置起后，该位表示时间戳秒值（支持 V2 格式）已经溢出并超过了 32’ hFFFF_FFFF

38.4.61. PTP PPS 控制寄存器(ETH_PTPPPSCR: 72Ch)

该寄存器控制 PPS 输出的频率。

位域	名称	属性	复位值	描述
31:7	RSV	-	0	保留，必须保持复位值
6:5	TRGTMODSEL	RW	0	PPS 输出的目标时间寄存器模式（Target Time Register Mode for PPS Output） 该字段指示 PPS 输出信号的目标时间寄存器模式： 00：仅为生成中断事件而编程目标时间寄存器。 01：保留 10：为生成中断事件以及开始或停止 PPS 输出信号生成而编程目标时间寄存器。 11：仅为开始或停止 PPS 输出信号生成而编程目标时间寄存器。未将任何中断置为有效。
4	PPSEN	RW	0	灵活 PPS 输出模式使能（Flexible PPS Output Mode Enable） 0：位 [3:0] 用作 PPSCTRL（固定 PPS 模式）。 1：位 [3:0] 用作 PPSCMD。

3:0	PPSCTRL	RW	0	PPS 输出频率控制 (PPS Output Frequency Control) 该字段控制 PPS 输出 (ptp_pps_o) 信号的频率。 PPSCTRL 的默认值为 0000， PPS 输出为每秒 1 个脉冲（宽度为 clk_ptp_i)。当 PPSCTRL 为其他值时， PPS 输出则为所生成的以下频率的时钟： 0001：二进制翻转为 2 Hz，数字翻转为 1 Hz。 0010：二进制翻转为 4 Hz，数字翻转为 2 Hz。 0011：二进制翻转为 8 Hz，数字翻转为 4 Hz。 0100：二进制翻转为 16 Hz，数字翻转为 8 Hz。 … 1111：二进制翻转为 32.768 KHz，数字翻转为 16.384 KHz。 注： 对于这些频率，在二进制翻转模式下， PPS 输出 (ptp_pps_o) 的占空比为 50%。在数字翻转模式下， PPS 输出频率为平均数。实际时钟的频率不同，每秒同步一次。例如： – PPSCTRL = 0010 时， PPS (2 Hz) 为如下序列： 第一个时钟的占空比为 50%，周期为 537 ms 第二个时钟的周期为 463 ms（268 ms 低电平， 195 ms 高电平） – PPSCTRL = 0011 时， PPS (4 Hz) 为如下序列： 前三个时钟的占空比为 50%，周期为 268 ms 第四个时钟的周期为 195 ms（134 ms 低电平， 61 ms 高电平） 此行为是因为在 ETH_PTPTSLR 寄存器中于数字翻转模式下对位进行非线性翻转而引起。

或 3:0	PPSCMD	RW	0	灵活 PPS 输出（ptp_pps_o[0]）控制（Flexible PPS Output (ptp_pps_o) Control） 为这些位编程非零值将命令 MAC 发起事件。当命令被传送或同步到 PTP 时钟域时，这些位将自动清零。软件应确保仅在这些位为“全零”时才进行编程。下面对 PPSCMD0 的值进行了说明： 0000：无命令 0001：启动单脉冲 该命令会生成单脉冲，该脉冲在目标时间寄存器中定义的起始点上升，持续时间为 PPS 宽度寄存器中所定义。 0010：启动脉冲串 该命令会生成脉冲串，该脉冲串在目标时间寄存器中定义的起始点上升，持续时间为 PPS 宽度寄存器中所定义，并以 PPS 间隔寄存器中定义的间隔进行重复。除非通过“经过一段时间后停止脉冲串”或“立即停止脉冲串”命令停止 PPS 脉冲串，否则默认情况下， PPS 脉冲串自由运行。 0011：取消启动 如果系统时间未超过编程的启动时间，该命令将取消启动单脉冲和启动脉冲串命令。0100：经过一段时间后停止脉冲串 目标时间寄存器中编程的时间结束后，该命令将停止由启动脉冲串命令（PPSCMD = 0010）启动的脉冲串。 0101：立即停止脉冲串 该命令会立即停止由启动脉冲串命令（PPSCMD = 0010）启动的脉冲串。 0110：取消停止脉冲串 如果编程的停止时间尚未结束，该命令将取消经过一段时间后停止脉冲串命令。 PPS 脉冲串在该命令成功执行后将自由运行。 0111-1111：保留

38.4.62. PTP 辅助时间戳纳秒寄存器(ETH_PTPATSNR: 730h)

位域	名称	属性	复位值	描述
31	RSV	-	0	保留，必须保持复位值
30:0	AUSTSLO	RO	0	辅助时间戳低位(Auxilliary Timestamp Low) 包含辅助时间戳的低 31 位(纳秒字段)

38.4.63. PTP 辅助时间戳秒寄存器(ETH_PTPATSSR: 734h)

位域	名称	属性	复位值	描述
31	RSV	-	0	保留，必须保持复位值
30:0	AUSTSHI	RO	0	辅助时间戳高位(Auxiliary Timestamp High) 包含辅助时间戳的低 32 位(秒字段)

38.4.64. PTP PPS 间隔寄存器(ETH_PTPPPSIR: 760h)

位域	名称	属性	复位值	描述
31:0	PPSINT	RW	0	PPS 输出信号间隔（PPS Output Signal Interval） 这些位存储 PPS 信号输出的上升沿之间的间隔。间隔以亚秒增量值的单位进行存储。 需要编程一个小于所需间隔的值。例如，如果 PTP 参考时钟为 50MHz（周期为 20ns），并且 PPS 信号输出的上升沿之间的预期间隔为 100ns（即 5 个单位的亚秒增量值），则应在该寄存器中编程值 4（5-1）。

38.4.65. PTP PPS 宽度寄存器(ETH_PTPPPSWR: 764h)

位域	名称	属性	复位值	描述
31:0	PPSWIDTH	RW	0	PPS 输出信号宽度（PPS Output Signal Width） 这些位存储 PPS 信号输出的上升沿和下降沿之间的宽度。宽度以亚秒增量值的单位数进行存储。 需要编程一个小于所需间隔的值。例如，如果 PTP 的参考时钟为 50MHz（周期为 20ns），并且 PPS 信号输出的上升沿和对应的下降沿之间的宽度为 80ns（即 4 个单位的亚秒增量值），则应该在该寄存器中编程 3（4-1）。 注：该寄存器中编程的值必须小于 ETH_PTPPPSIR 寄存器中编程的值。

38.4.66. DMA 总线模式寄存器(ETH_DMABMR: 1000h)

总线模式寄存器为 DMA 建立总线工作模式。

位域	名称	属性	复位值	描述
31	RIB	RW	0	重建 INCRx 突发(Rebuild INCRx Burst) 该位置 1 且 AHB 主设备获得了 SPLIT、RETRY 或提前突发终止(EBT)响应时，AHB 主接口将基于 INCRx 和 SINGLE 传输重新构建已发起突发传输的挂起节拍。默认情况下，AHB 主接口基于未指定(INCR)突发重建 EBT 的挂起节拍。
30:28	RSV	-	0	保留，必须保持复位值
27	TXPR	RW	0	发送优先级(Transmit Priority) 该位置 1 时，表示在系统测总线仲裁期间，Tx DMA 的优先级高于 Rx DMA。
26	MB	RW	0	混合突发（Mixed Burst） 该位置为高电平且 FB 位置为低电平时，如果突发长度大于 16，AHB 主设备会执行未定义的突发传输（INCR）。如果突发长度小于等于 16，AHB 主设备会执行固定突发传输（INCRx 和 SINGLE）。
25	AAB	RW	0	地址对齐的节拍（Address-Aligned Beats） 当该位设置为高电平并且 FB 位等于 1 时， AHB 接口会生成与起始地址 LS 位对齐的所有突发。如果 FB 位等于 0，则第一个突发（访问数据缓冲器的起始地址）不对齐，但后续的突发与地址对齐。

24	EPM	RW	0	8xPBL 模式 (8xPBL mode) 当设置为高电平时，该位会将编程的 PBL 值（位 [22:17] 和位 [13:8]）乘以八倍。因此，DMA 根据 PBL 值以最大 8、 16、 32、 64 、128 和 256 个节拍传输数据。
23	USP	RW	0	使用单独的 PBL (Use Separate PBL) 设置为高电平时，它会配置 RxDMA，将位 [22:17] 中配置的值用作 PBL，而位 [13:8] 中的 PBL 值仅适用于 TxDMA 操作。当该位清零时，位 [13:8] 中的 PBL 值同时适用于两个 DMA 引擎。
22:17	RDP	RW	1	Rx DMA PBL 这些位指示要在一个 RxDMA 事务中传输的最大节拍数。这是在单个块读/写操作中使用的最大值。 RxDMA 每次在主机总线上开始突发传输时，始终尝试按 RDP 中指定的方式进行突发。允许使用值 1、 2、 4、 8、 16 和 32 对 RDP 进行编程。任何其它值都会产生未定义的行为。仅在 USP 设置为高电平时这些位才有效且适用。
16	FB	RW	0	固定突发 (Fixed Burst) 该位控制 AHB 主接口是否执行固定突发传输。置 1 时， AHB 在正常突发传输开始期间仅使用 SINGLE、 INCR4、 INCR8 或 INCR16。复位时，AHB 使用 SINGLE 和 INCR 突发传输操作。
15:14	PM	RW	0	Rx Tx 优先级比 (Rx Tx priority ratio) RxDMA 请求优先于 TxDMA 请求，优先级比如下： 00： 1:1 01： 2:1 10： 3:1 11： 4:1 这仅在 DA 位清零时有效。
13:8	PBL	RW	1	可编程突发长度 (Programmable Burst Length) 这些位定义了一次 DMA 操作中所传输的最大节拍数。这个最大值用于单次读写操作。 DMA 每次在主机总线上进行突发传输时，总会尝试按照 PBL 的设定值进行突发传输。 PBL 允许设置为 1, 2, 4, 8, 16 和 32。除此以外的其它值均会导致意外情况发生。当 USP 置起时，PBL 值仅适用于 DMA 发送操作。 如果要传输的节拍数量大于 32，需按以下步骤操作： 1. 设置 PBLx8 模式. 2. 设置 PBL. 例如，如果待传输的最大值为 64，那么首先需要将 PBLx8 置 1，再将 PBL 设置为 8. PBL 的值有以下限制： 可能的最大传输数会受到 MTL 层的发送 FIFO 和接收 FIFO 大小的限制，以及 DMA 上的数据总线宽度的限制。FIFO 的限制： FIFO 所支持的最大传输次数是 FIFO 深度的一半，除非另有规定。

7	EDFE	RW	0	EDFE：增强描述符格式使能（Enhanced Descriptor Format Enable） 该位置 1 时，使能增强描述符格式，并将描述符大小增加至 32 字节（ 8 个 DWORD）。如果已激活时间戳功能（ ETH_PTPTSCR 位 0 TSE=1）或 IPv4 校验和减荷（ ETH_MACCR 位 10 IPCO=1），则必须使用此增强描述符。
6:2	DSL	RW	0	描述符跳过长度（Descriptor Skip Length） 该位指定两个未链接描述符之间跳过的字数。地址从当前描述符结束处开始跳到下一个描述符起始处。当 DSL 值等于零时，在环形模式下， DMA 会将描述符表视为连续的。
1	DA	RW	0	DMA 仲裁（DMA Arbitration） 0：循环调度， Rx:Tx 优先级比在位[15:14] 中给出 1： Rx 优先于 Tx
0	SWR	RW	0	软件复位（Software Reset） 当该位置 1 时， MAC DMA 控制器会复位所有 MAC 子系统的内部寄存器和逻辑。在所有内核时钟域完成复位操作后，该位自动清零。重新编程任何内核寄存器之前，在该位中读取 0 值。

38.4.67. DMA 发送轮询要求寄存器(ETH_DMATPDR: 1004h)

应用程序使用此寄存器来指示 DMA 轮询发送描述符列表。发送轮询要求寄存器使能发送 DMA 来检查当前描述符是否为 DMA 所有。如果 TxDMA 处于挂起模式，则发出发送轮询要求命令将其唤醒。如果发送帧中出现下溢错误或发送 DMA 所拥有的描述符不可用，则 TxDMA 会进入挂起模式。用户可以随时发出此命令，当该命令开始重新获取主机存储器的当前描述符后， TxDMA 即会对其进行复位。

位域	名称	属性	复位值	描述
31:0	TPD	R	0	发送轮询要求（Transmit Poll Demand） 向这些位写入任何值时， DMA 都会读取 ETH_DMACHTDR 寄存器指向的当前描述符。如果该描述符不可用（由主机所有），则发送会返回到挂起状态，并将 ETH_DMASR 寄存器位 2 进行置位。如果该描述符可用，则发送会继续进行。

38.4.68. DMA 接收轮询要求寄存器(ETH_DMARPDR: 1008h)

应用程序使用此寄存器来指示 DMA 轮询接收描述符列表。接收轮询要求寄存器会使能接收 DMA 来检查新描述符。此命令用于将 RxDMA 从挂起状态唤醒。仅当 RxDMA 所拥有的描述符不可用时，它才会进入挂起状态。

位域	名称	属性	复位值	描述
31:0	RPD	R	0	接收轮询要求（Receive Poll Demand） 向这些位写入任何值时， DMA 都会读取 ETH_DMACHRDR 寄存器指向的当前描述符。如果该描述符不可用（由主机所有），则接收会返回到挂起状态，且不会将 ETH_DMASR 寄存器位 7 进行置位。如果该描述符可用，接收 DMA 将返回到活动状态。

38.4.69. DMA 接收描述符列表地址寄存器(ETH_DMARDLAR: 100Ch)

接收描述符列表地址寄存器会指向接收描述符列表的起始处。描述符列表位于主机的物理存储器空间，且必须为字对齐。 DMA 会将描述符列表的对应 LSB 位置为低电平，进而在内部将其转换为总线宽度对齐地址。仅当接收停止时，才允许对 ETH_DMARDLAR 寄存器执行写操作。停止后，必须先对 ETH_DMARDLAR 寄存器执行写操作，然后才能发出接收启动命令。

位域	名称	属性	复位值	描述
31:0	SRL	RW	0	接收列表的起始处（Start of Receive List） 此字段包含接收描述符列表中的首个描述符的基址。 DMA 会在内部将 LSB 位 [1:0]（对应于 32 位的总线宽度）忽略，并将其值均视为零。因此，这些 LSB 位为只读。

38.4.70. DMA 发送描述符列表地址寄存器(ETH_DMATDLAR: 1010h)

发送描述符列表地址寄存器会指向发送描述符列表的起始处。描述符列表位于主机的物理存储器空间，且必须为字对齐。 DMA 会将描述符列表的对应 LSB 位置为低电平，进而在内部将其转换为总线宽度对齐地址。仅当发送停止时，才允许对 ETH_DMATDLAR 寄存器执行写操作。发送停止后，可以先对 ETH_DMATDLAR 寄存器执行写操作，然后再发出发送启动命令。

位域	名称	属性	复位值	描述
31:0	STL	RW	0	发送列表的起始处（Start of Transmit List） 此字段包含发送描述符列表中的首个描述符的基址。 DMA 会在内部将 LSB 位 [1:0]（对应于 32 位的总线宽度）忽略，并将其值均视为零。因此，这些 LSB 位为只读。

38.4.71. DMA 状态寄存器(ETH_DMASR: 1014h)

状态寄存器包含所有 DMA 向应用程序报告的状态位。软件驱动程序通常在中断服务例程或轮询期间读取 ETH_DMASR 寄存器。此寄存器中的大部分字段会导致主机中断。读取时 ETH_DMASR 寄存器位不会清零。向 ETH_DMASR 寄存器 [16:0] 中的（未保留）位写入 1 会将其清零，写入 0 则不起作用。通过屏蔽 ETH_DMAIER 寄存器中的位，可以屏蔽各个对应字段（位 [16:0]）。

位域	名称	属性	复位值	描述
31	RSV	-	0	保留，必须保持复位值
30	LPIS	R	0	LPI 状态(LPI Status) 此位指示 MAC 的 LPI 逻辑中的中断事件。软件必须读取 MAC 内核中对应的寄存器，以获取产生中断的具体原因，然后将其源清零，才能将此位复位为 0。
29	TSTS	R	0	时间戳触发状态（Timestamp Trigger Status） 此位指示 MAC 内核的时间戳发生器块中发生的中断事件。软件必须读取 MAC 内核的状态寄存器，将其源（位 9）清零，才能将此位复位为 0。当此位处于高电平时，使能后会产生中断。

28	PMTS	R	0	PMT 状态（PMT Status） 此位指示 MAC 内核的 PMT 中发生的事件。软件必须读取 MAC 内核中对应的寄存器，以获取产生中断的具体原因，然后将其源清零，才能将此位复位为 0。当此位处于高电平时，使能后会产生中断
27	MMCS	R	0	MMC 状态（MMC Status） 此位反映 MAC 内核的 MMC 中发生的事件。软件必须读取 MAC 内核中对应的寄存器，以获取产生中断的具体原因，然后将中断源清零，才能将此位置 0。当此位处于高电平时，使能后会产生中断
26	RSV	-	0	保留，必须保持复位值
25:23	EBS	R	0	错误位状态（Error Bits Status） 这些位表示的是造成总线错误的错误类型。仅适用于当位 13 (FBES)被置起的情况。该位域不会产生中断 000：在 Rx DMA 传输写数据时出错 011：在 Tx DMA 传输读数据时出错 100：在 Rx DMA 描述符写访问时出错 101：在 Tx DMA 描述符写访问时出错 110：在 Rx DMA 描述符读访问时出错 111：在 Tx DMA 描述符读访问时出错 注意： 001 和 010 表示保留。
22:20	TPS	R	0	送过程状态（Transmit Process State） 这些位指示发送 DMA FSM 的状态。此字段不会产生中断。 000： 停止；发出复位或停止发送命令 001： 运行；正在提取发送描述符 010： 运行；正在等待状态信息 011： 运行；正在读取主机内存缓存数据并将其排队到发送缓存(Tx FIFO) 100： 时间戳写状态 101： 保留 110： 暂停；发送描述符不可用或发送缓冲器下溢 111： 运行；正在关闭发送描述符
19:17	RPS	R	0	接收过程状态（Receive Process State） 这些位指示接收 DMA FSM 的状态。此字段不会产生中断。 000： 停止；发出复位或停止发送命令 001： 运行；正在提取接收描述符 010： 保留备用 011： 运行；正在等待接收数据包 100： 暂停；接收描述符不可用 101： 运行；正在关闭描述符 110： 时间戳写状态 111： 运行；将接收缓冲器的数据包转发到主机内存

16	NIS	R	0	正常中断汇总（Normal Interrupt Summary） 当使能 ETH_DMAIER 寄存器中对应的中断位时，正常中断汇总位的值是以下位的逻辑或运算结果： ETH_DMASR [0]：发送中断（Transmit interrupt） ETH_DMASR [2]：发送缓冲区不可用（Transmit buffer unavailable） ETH_DMASR [6]：接收中断（Receive interrupt） ETH_DMASR [14]：提前接收中断（Early receive interrupt） 只有未屏蔽的位会影响正常中断汇总位。它是黏着位，每当导致 NIS 位置 1 的对应位被清零时，必须同时将它也清零（通过向此位写入 1）。
15	AIS	R	0	异常中断汇总（Abnormal Interrupt Summary） 当使能 ETH_DMAIER 寄存器中对应的中断位时，异常中断汇总位的值是以下位的逻辑或运算结果： ETH_DMASR [1]：发送过程停止（Transmit process stopped） ETH_DMASR [3]：发送 jabber 超时（Transmit jabber timeout） ETH_DMASR [4]：接收 FIFO 上溢（Receive FIFO overflow） ETH_DMASR [5]：发送下溢（Transmit underflow） ETH_DMASR [7]：接收缓冲区不可用（Receive buffer unavailable） ETH_DMASR [8]：接收过程停止（Receive process stopped） ETH_DMASR [9]：接收看门狗超时（Receive watchdog timeout） ETH_DMASR [10]：提前发送中断（Early transmit interrupt） ETH_DMASR [13]：致命总线错误（Fatal bus error） 只有未屏蔽的位会影响异常中断汇总位。它是黏着位，每当导致 AIS 位置 1 的对应位被清零时，必须同时将它也清零。
14	ERS	R	0	提前接收状态（Early Receive Status） 该位表示 DMA 填充了数据包的第一个数据缓存。当软件向该位写 1 或者置起该寄存器的位 6(RS)时 （以先发生的为准），即可清除该位。
13	FBES	R	0	致命总线错误状态（Fatal Bus Error Status） 该位表示发生了位[25：23]所描述的总线错误。该位被置起后，对应的 DMA 将关闭全部的总线访问。
12:11	RSV	-	0	保留，必须保持复位值
10	ETS	R	0	提前发送状态（Early Transmit Status） 此位指示要发送的帧已完全传输到发送 FIFO。
9	RWTS	R	0	接收看门狗超时状态（Receive Watchdog Timeout Status） 该位被置起后，该位表示在接收看门狗定时器在接收当前帧时到期，并且在看门狗发生超时后，当前帧被截断。
8	RPSS	R	0	接收过程停止状态（Receive Process Stopped Status） 接收过程进入停止状态时，会对此位进行置位。

7	RBUS	R	0	接收缓冲区不可用状态（Receive Buffer Unavailable Status） 该位表示主机占用了接收列表中的下一个描述符，使得 DMA 无法获取。因此接收流程暂停。主机需要更改描述符的所有权，并释放接收轮询请求指令才能恢复接收流程。如果未释放接收轮询请求指令，那么 DMA 在收到一下个输入帧的时候，会恢复接收流程。只有在 DMA 占用了前一个接收描述符的情况下，该位才会置起。
6	RS	R	0	接收状态（Receive Status） 该位表示帧接收已完成。接收完成后，将在最后一个描述符中复位 RDES1(完成后关闭中断)的位 31，具体的帧状态信息会更新到描述符中。 接收流程仍处于运行状态
5	TUS	R	0	发送下溢状态（Transmit Underflow Status） 此位指示在帧发送期间，发送缓冲区发生下溢。发送会进入挂起状态，且下溢错误 TDES0[1]置 1。
4	ROS	R	0	接收上溢状态（Receive Overflow Status） 此位指示在帧接收期间，接收缓冲区发生 上溢。如果部分帧已传输到应用程序，则 RDES0[11] 中的上溢状态位置 1。
3	TJTS	R	0	发送 jabber 超时状态（Transmit Jabber Timeout Status） 该位表示当帧大小超过 2048 字节（若 Jumbo 帧被使能，则为 10240 字节）时会发生发送 Jabber 定时器超时。Jabber 发生超时后，发送流程中止并进入停止状态。这将导致发送 Jabber 超时标志位 TDES0[14]被置起。
2	TBUS	R	0	发送缓冲区不可用状态（Transmit Buffer Unavailable Status） 该位表示主机占用了发送列表中的下一个描述符，使得 DMA 无法获取，因此，发送流程暂停。位[22： 20]解释了发送流程状态。如果需要恢复发送流程，可以通过设置 TDES0[31]更改描述符的所有权并释放发送轮询请求指令。
1	TPSS	R	0	发送过程停止状态（Transmit Process Stopped Status） 当发送停止时，此位置 1。
0	TS	R	0	发送状态（Transmit Status） 该位表示帧发送已完成。发送完成后， TDES0 的位 Bit 31(OWN)会复位，具体的帧状态信息会更新到描述符。

38.4.72. DMA 工作模式寄存器(ETH_DMAOMR: 1018h)

工作模式寄存器将建立发送和接收工作模式及命令。作为 DMA 初始化过程的一部分，应将 ETH_DMAOMR 寄存器作为最后的 CSR 进行写操作。

位域	名称	属性	复位值	描述
31:27	RSV	-	0	保留，必须保持复位值

26	DTCEFD	RW	0	禁止丢弃 TCP/IP 校验和错误帧（Dropping of TCP/IP Checksum Error Frames Disable) 0：如果 FEF 位进行了复位，则会丢弃所有错误帧。 1：如果帧中仅存在由接收校验和减荷引擎检测出来的错误，则内核不会丢弃它。这类帧在 MAC 接收到的以太网帧中没有任何错误（包括 FCS 错误），而仅在封装的有效负载中有错误。
25	RSF	RW	0	接收存储并转发（Receive Store and Forward) 0：Rx FIFO 在直通模式下工作，具体取决于 RTC 位指定的阈值。 1：向 Rx FIFO 写入完整帧后，可从中读取一个帧，同时忽略 RTC 位。
24	DFRF	RW	0	禁止刷新接收帧（Disable Flushing of Received Frames) 当此位置 1 时， RxDMA 不会因为接收描述符/缓冲区不可用而刷新任何帧（通常情况下，当此位清零时会这样做）
23:22	RSV	-	0	保留，必须保持复位值
21	TSF	RW	0	发送存储并转发（Transmit Store and Forward) 当此位置 1 时，如果发送 FIFO 中有一个完整帧，则发送会启动。当此位置 1 时，会忽略由 ETH_DMAOMR 寄存器位 [16:14] 指定的 TTC 值。 该位清零后， ETH_DMAOMR 寄存器位 [16:14] 指定的 TTC 值才会有效。 该位只有在已停止传输时才能更改。
20	FTF	RW	0	刷新发送 FIFO (Flush Transmit FIFO) 当此位置 1 时，发送 FIFO 控制器逻辑电路被恢复至默认值，发送 FIFO 中的所有数据要么丢失，要么被清空。在完成清空动作后，该位即被清除。在该位被清除之前，不允许向工作模式寄存器进行写操作。 MAC 发送器已接受的数据是不会被清空的，会被安排发送，并导致数据下溢和超短帧传输。
19:17	RSV	-	0	保留，必须保持复位值
16:14	TTC	RW	0	发送阈值控制（Transmit Threshold Control) 这三个位用于控制发送 FIFO 的阈值级别。当发送 FIFO 中的帧大小大于阈值时启动发送。此外，还会发送长度小于阈值的全帧。这些位只有在 TSF 位（位 21）清零后才能使用。 000: 64 001: 128 010: 192 011: 256 100: 40 101: 32 110: 24 111: 16

13	ST	RW	0	启动/停止发送(Start/Stop Transmission) 0：在发送完当前帧后，发送流程进入停止状态。保存发送列表中的下一个描述符位置，并在重新开始发送时，将该描述符位置作为当前位置。若要修改列表地址，需要在该位被复位时向发送描述符列表地址寄存器编写一个新值。该位被重新置起时，这个写入的新值会生效。只有在当前帧已发送完成或者发送流程进入暂停状态时，停止发送指令才会生效。 1：发送流程处于运行状态， DMA 会检查当前位置的发送列表，确定待发送的帧。 DMA 要么从当前列表位置获取描述符（即发送描述符列表地址寄存器所设定的发送列表基地址），要么从之前发送流程中止的位置获取描述符。如果 DMA 未占用当前描述符，发送流程进入暂停状态，并且状态寄存器的位 2（发送缓冲不可用）会被置起。发送开始指令仅在发送停止的时候才有效。如果在未设置发送描述符列表地址寄存器之前就发出了发送开始指令，则 DMA 将会出现意想不到的后果。
12:8	RSV	-	0	保留，必须保持复位值
7	FEF	RW	0	转发错误帧（Forward Error Frames) 0：Rx FIFO 会丢弃带有错误状态（ CRC 错误、冲突错误、巨帧、看门狗超时、上溢）的帧。不过，如果某个帧的起始字节（写）指针已传输到读控制器端（阈值模式下），则不会丢弃该帧。如果 AHB 总线上未传输（输出）帧的起始字节，则 Rx FIFO 会丢弃此错误帧。 1：除短错误帧之外的所有帧都会转发到 DMA。
6	FUGF	RW	0	转发过小的好帧（Forward Undersized Good Frame) 0：Rx FIFO 会丢弃所有不足 64 字节的帧，除非因接收阈值下限更低（例如 RTC = 01）而导致此类帧已传输。 1：Rx FIFO 会转发包含 pad 字节和 CRC 的过小帧（无错误但长度不足 64 字节的帧）。
5	DGF	RW	0	丢弃巨型帧(Drop Giant Frame) 当此位置 1 时，MAC 会丢弃接收 FIFO 中收到的巨型帧。
4:3	RTC	RW	0	接收阀值控制（Receive Threshold Control) 这两个位用于控制接收 FIFO 的阈值级别。当接收 FIFO 中的帧大小大于阈值时启动 DMA 传输（请求）。此外，长度小于阈值的全帧会自动传输。 注意：如果配置的接收 FIFO 大小为 128 字节，则不适合使用 11 作为阈值。 注意：这些位只有在 RSF 位为 0 时才有效，而当 RSF 位置 1 后会忽略这些位 00： 64 01： 32 10： 96 11： 128
2	OSF	RW	0	处理第二个帧（Operate on Second Frame) 该位置 1 时会命令 DMA 处理第二个发送数据帧，即使尚未获得首个帧的状态。

1	SR	RW	0	启动/停止接收（Start/Stop Receive) 0：在发送完当前帧后，接收 DMA 操作停止。保存接收列表中的下一个描述符位置，并在重新开始接收时，将该描述符位置作为当前位置。只有在接收流程进入运行状态（正等着接收数据包）或者暂停状态时，停止接收指令才会生效 1：接收流程处于运行状态， DMA 会尝试从接收列表获取描述符并处理输入的帧。 DMA 要么从当前列表位置获取描述符（即接收描述符列表地址寄存器所设定的地址），要么从之前接收流程中止的位置获取描述符。如果 DMA 未占用当前描述符，接收流程进入暂停状态，并且状态寄存器的位 7（接收缓冲不可用）会被置起。接收开始指令仅在接收停止的时候才有效。如果在未设置接收描述符列表地址寄存器之前就发出了接收开始指令，则 DMA 将会出现意想不到的后果。
0	RSV	-	0	保留，必须保持复位值

38.4.73. DMA 中断使能寄存器(ETH_DMAIER: 101Ch)

中断使能寄存器可使能 ETH_DMASR 报告的中断。将一个位置 1 可使能相应的中断。完成硬件或软件复位之后，会禁止所有中断。

位域	名称	属性	复位值	描述
31:17	RSV	-	0	保留，必须保持复位值
16	NISE	RW	0	正常中断汇总使能（Normal Interrupt Summary Enable) 该位置 1 时，会使能一个正常中断。该位清零时，会禁止一个正常中断。该位可使能以下位： ETH_DMASR [0]：发送中断（Transmit interrupt) ETH_DMASR [2]：发送缓冲区不可用（Transmit buffer unavailable) ETH_DMASR [6]：接收中断（Receive interrupt) ETH_DMASR [14]：提前接收中断（Early receive interrupt)
15	AISE	RW	0	异常中断汇总使能（Abnormal Interrupt Summary Enable) 该位置 1 时，会使能一个异常中断。该位清零时，会禁止一个异常中断。该位可使能以下位： ETH_DMASR [1]：发送过程停止（Transmit process stopped) ETH_DMASR [3]：发送 jabber 超时（Transmit jabber timeout) ETH_DMASR [4]：接收上溢（Receive overflow) ETH_DMASR [5]：发送下溢（Transmit underflow) ETH_DMASR [7]：接收缓冲区不可用（Receive buffer unavailable) ETH_DMASR [8]：接收过程停止（Receive process stopped) ETH_DMASR [9]：接收看门狗超时（Receive watchdog timeout) ETH_DMASR [10]：提前发送中断（Early transmit interrupt) ETH_DMASR [13]：致命总线错误（Fatal bus error)
14	ERIE	RW	0	提前接收中断使能（Early Receive Interrupt Enable) 当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能提前接收中断。该位清零时，会禁止提前接收中断。

13	FBEIE	RW	0	致命总线错误中断使能（Fatal Bus Error Interrupt Enable） 当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能致命总线错误中断。该位清零时，会禁止致命总线使能中断
12:11	RSV	-	0	保留，必须保持复位值
10	ETIE	RW	0	提前发送中断使能（Early Transmit Interrupt Enable） 当该位通过异常中断汇总使能位（ ETH_DMAIER 寄存器 [15]）置 1 时，可使能提前发送中断。该位清零时，会禁止提前发送中断。
9	RWTIE	RW	0	接收看门狗超时中断使能（Receive Watchdog Timeout Interrupt Enable） 当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收看门狗超时中断。该位清零时，会禁止接收看门狗超时中断。
8	RPSIE	RW	0	接收过程停止中断使能（Receive Process Stopped Interrupt Enable） 当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收停止中断。该位清零时，会禁止接收停止中断。
7	RBUIE	RW	0	接收缓冲区不可用中断使能（Receive Buffer Unavailable Interrupt Enable） 当该位通过异常中断汇总使能位（ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收缓冲区不可用中断。该位清零时，会禁止接收缓冲区不可用中断。
6	RIE	RW	0	接收中断使能（Receive Interrupt Enable） 当该位通过正常中断汇总使能位（ ETH_DMAIER 寄存器 [16]）置 1 时，可使能接收中断。该位清零时，会禁止接收中断。
5	TUIE	RW	0	下溢中断使能（Underflow Interrupt Enable） 当该位通过异常中断汇总使能位（ ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送下溢中断。该位清零时，会禁止下溢中断。
4	ROIE	RW	0	上溢中断使能（Overflow Interrupt Enable） 当该位通过异常中断汇总使能位（ ETH_DMAIER 寄存器 [15]）置 1 时，可使能接收上溢中断。该位清零时，会禁止上溢中断。
3	TJTIE	RW	0	发送 jabber 超时中断使能（Transmit Jabber Timeout Interrupt Enable） 当该位通过异常中断汇总使能位（ ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送 jabber 超时中断。该位清零时，会禁止发送 jabber 超时中断。
2	TBUIE	RW	0	发送缓冲区不可用中断使能（Transmit Buffer Unavailable Interrupt Enable） 当该位通过正常中断汇总使能位（ETH_DMAIER 寄存器 [16]）置 1 时，可使能发送缓冲区不可用中断。该位清零时，会禁止发送缓冲区不可用中断。
1	TPSIE	RW	0	发送过程停止中断使能（Transmit Process Stopped Interrupt Enable） 当该位通过异常中断汇总使能位（ ETH_DMAIER 寄存器 [15]）置 1 时，可使能发送停止中断。该位清零时，会禁止发送停止中断。
0	TIE	RW	0	发送中断使能（Transmit Interrupt Enable） 当该位通过正常中断汇总使能位（ ETH_DMAIER 寄存器 [16]）置 1 时，可使能发送中断。该位清零时，会禁止发送中断。

只有 DMA 状态寄存器的 TSTS 或 PMTS 位已置位但相应的中断均未带标记时，或者 NIS/AIS 状态位已置位并且相应的中断使能位（NISE/AISE）已使能时，才会生成以太网中断。

38.4.74. DMA 丢失帧和缓冲区上溢计数器寄存器(ETH_DMAMFBOCR: 1020h)

DMA 可维护两个计数器在接收过程中对丢失帧数目进行计数。该寄存器会报告计数器的当前值。计数器用于进行诊断。位 [15:0] 指示因主机缓冲区不可用（无可用的接收描述符）而丢失的帧。位 [27:17] 指示因 Rx FIFO 存在上溢情况以及帧长度过短（不足 64 字节的好帧）而丢失的帧。

位域	名称	属性	复位值	描述
31:29	RSV	-	0	保留，必须保持复位值
28	OBFOC	R	0	FIFO 溢出计数器溢出位 (Overflow Bit for FIFO Overflow Counter) 每一次溢出帧计数器(位[27:17])发生溢出时，该位就会被置起，即接收 FIFO 发生溢出，溢出帧计数器达到最大值。在这种情况下，溢出帧计数器被重置为全 0，该位表示已发生翻转。
27:17	OFC	R	0	溢出帧计数器 (Overflow Frame Counter) 这些位域表示应用程序所丢失的帧数目。
16	OBMFC	R	0	丢失帧计数器溢出位 (Overflow Bit for Missed Frame Counter) 每当丢失帧计数器（位[15:0]）发生溢出时，该位被置起，即 DMA 会忽略由于主机接收缓存不可用而传入的帧，且丢失帧计数器达到最大值。在这种情况下，丢失帧计数器被重置为全 0，该位代表翻转已发生。
15:0	MFC	R	0	丢失帧计数器 (Missed Frame Counter) 这些位域表示由于主机接收缓存不可用而被控制器丢失的帧数目。每当 DMA 忽略一个传入的帧时，该计数器就会递增一次。

38.4.75. DMA 接收中断看门狗定时器寄存器(ETH_DMARIWTR: 1024h)

接收中断看门狗定时器寄存器指示来自 DMA 的接收中断 (RI) 的看门狗超时。向该寄存器中写入非零值时，会针对 ETH_DMACSR 寄存器的 RI 位使能看门狗定时器。

位域	名称	属性	复位值	描述
31:8	RSV	-	0	保留，必须保持复位值
7:0	RIWTC	RW	0	接收中断看门狗定时器计数 (Receive Interrupt Watchdog Timer Count) 指示 HCLK 时钟周期数乘以 256 之后的时间值，即看门狗定时器的设定时间值。看门狗定时器会在 RxDMA 结束帧传输之后由编程设定的值触发，但相应的 RS 状态位因相应描述符中的 RDES1[31] 已置 1 而未置 1。当看门狗定时器计时结束时， RS 位置 1 且定时器停止。由于 RS 根据每个接收帧的 RDES1[31] 进行自动设置而使 RS 位设为高电平时，看门狗定时器复位。

38.4.76. DMA 当前主机发送描述符寄存器(ETH_DMACHTDR: 1048h)

当前主机发送描述符寄存器会指向 DMA 所读取的当前发送描述符的起始地址。

位域	名称	属性	复位值	描述

31:0	HTDAP	R	0	主机发送描述符地址指针 (Host Transmit Descriptor Address Pointer) 复位时清零。该指针在运行期间由 DMA 更新。

38.4.77. DMA 当前主机接收描述符寄存器(ETH_DMACHRDR: 104Ch)

当前主机接收描述符寄存器会指向 DMA 所读取的当前接收描述符的起始地址。

位域	名称	属性	复位值	描述
31:0	HRDAP	R	0	主机接收描述符地址指针 (Host Receive Descriptor Address Pointer) 复位时清零。该指针在运行期间由 DMA 更新。

38.4.78. DMA 当前主机发送缓冲区地址寄存器(ETH_DMACHTBAR: 1050h)

当前主机发送缓冲区地址寄存器会指向 DMA 所读取的当前发送缓冲区地址。

位域	名称	属性	复位值	描述
31:0	HTBAP	R	0	主机发送缓冲区地址指针 (Host Transmit Buffer Address Pointer) 复位时清零。该指针在运行期间由 DMA 更新。

38.4.79. DMA 当前主机接收缓冲区地址寄存器(ETH_DMACHRBAR: 1054h)

当前主机接收缓冲区地址寄存器会指向 DMA 所读取的当前接收缓冲区地址。

位域	名称	属性	复位值	描述
31:0	HRBAP	R	0	主机接收缓冲区地址指针 (Host Receive Buffer Address Pointer) 复位时清零。该指针在运行期间由 DMA 更新。

39. 数字摄像头接口（DCMI）

39.1. 概述

数字摄像头接口（DCMI）是一个同步并行接口，能够接受外部 8 位、10 位、12 位或 14 位 CMOS 摄像头模块发出的高速数据流。可支持不同的数据格式，YCbCr4:2:2/RGB565 逐行视频和压缩数据（JPEG）。

此接口适用于黑白摄像头、X24 和 X5 摄像头，并假定所有将预处理（如调整大小）都在摄像头模块中执行。

39.2. 主要特性

- 8 位、10 位、12 位或 14 位并行接口。
- 内嵌码/外部行同步和帧同步
- 连续模式或快照模式
- 裁剪功能
- 支持以下数据格式
  - 8/10/12/14 位逐行视频：单色或原始拜尔格式
  - YCbCr 4:2:2 逐行视频
  - RGB565 逐行视频
  - 压缩数据：JPEG

39.3. 功能描述

数字摄像头接口是一个同步并行接口，可接受高速（高达 54MB/s）数据流。该接口包含多达 14 条数据线（D13-D0）和一条像素时钟线（PIXCLK）。像素时钟的极性可以编程，因此可以在像素时钟的上升沿或下降沿捕获数据。

这些数据被放到 32 位数据寄存器（DCMI_DR）中，然后通过通用 DMA 进行传输。图像缓冲区由 DMA 管理，而不是由摄像头接口管理。

从摄像头接受的数据可以按行/帧来组织（原始 YUB/RGB/拜尔模式），也可以是一系列 JPEG 图像。要使能 JPEG 图像接收，必须将 JPEG 位置 1。

数据流可由可选的 HSYNC（水平同步）信号和 VSYNC（垂直同步）信号硬件同步，或者通过数据流中嵌入的同步码同步。

39.3.1. 结构框图

图 39-1 DCMI 框图

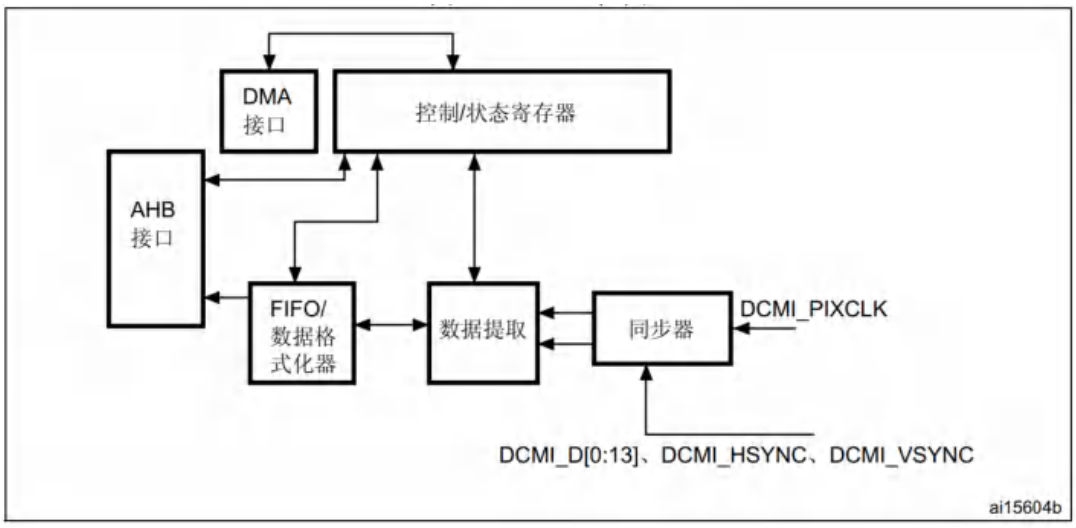
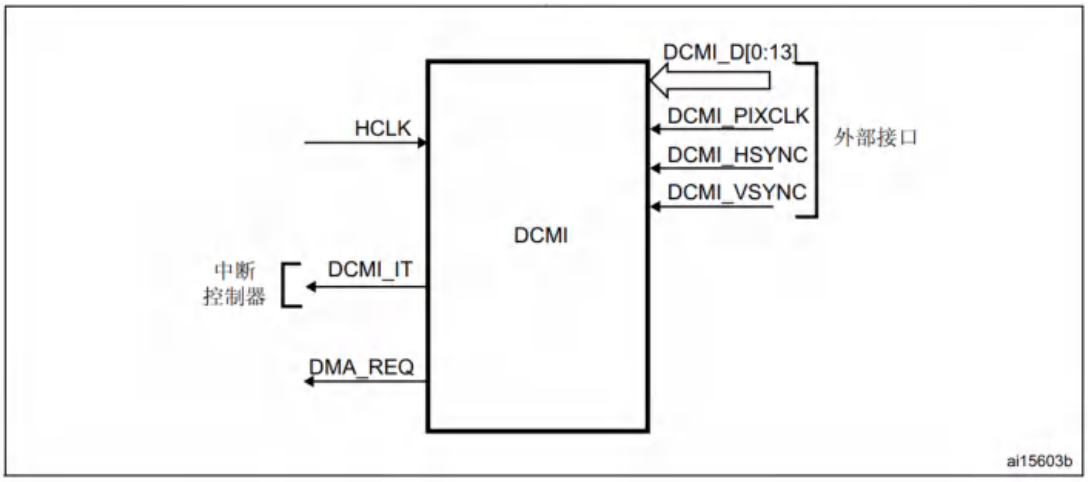


图 39-2 顶级框图



39.3.2. DCMI 时钟

数字摄像头接口使用 PIXCLK 和 HCLK 这两个时钟域。伴随 PIXCLK 产生的信号稳定后，将在 HCLK 上升沿时对这些信号进行采样。HCLK 域中的一个使能信号用于指示摄像头发出的数据已经稳定，可进行采样。PIXCLK 的最大周期必须大于 2.5 个 HCLK 周期。

39.3.3. DMA 接口

当 DCMI_CR 寄存器中的 Capture 位置 1 时，激活 DMA 接口。摄像头接口每次在其寄存器中收到一个完整的 32 位数据块时，都将触发一个 DMA 请求。

39.3.4. DCMI 物理接口

该接口由 11/13/15/17 个输入信号组成，仅支持从模式。根据 DCMI_CR 寄存器中的 EDM[1:0]位的设置，摄像头接口可以捕获 8 位、10 位、12 位或者 14 位数据。如果使用的位数少于 4，则必须将未使用的输入引脚接

地。

表 39-1 DCMI 信号

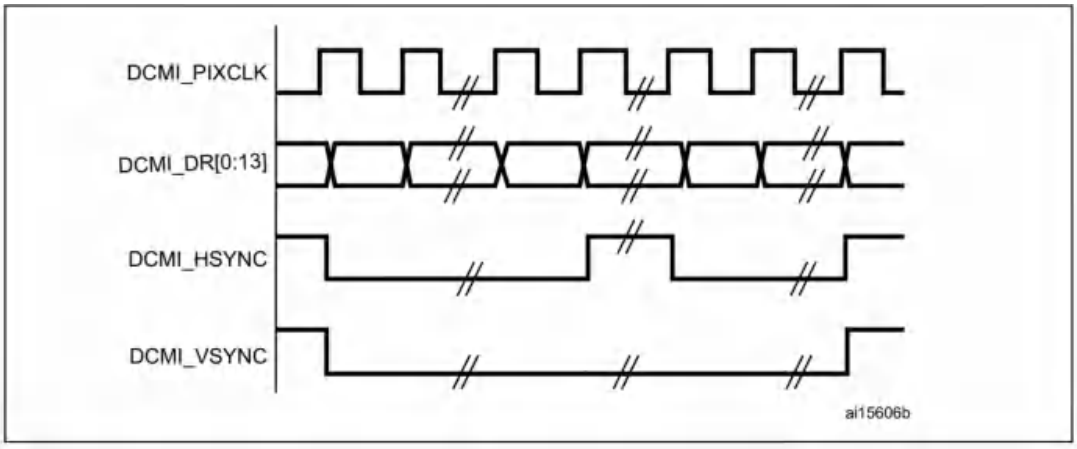
信号名称		信号说明
8 位 10 位 12 位 14 位	D[0..7] D[0..9] D[0..11] D[0..13]	Data
PIXCLK		像素时钟
HSYNC		水平同步/数据有效
VSYNC		垂直同步

数据与 PIXCLK 将保持同步，并根据像素时钟的极性在像素时钟上升沿/下降沿发生变化。

HSYNC 信号指示行的开始/结束

VSYNC 信号指示帧的开始/结束

图 39-3 DCMI 信号波形



1）DCMI_PIXCLK 的捕获沿为下降沿，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。

2）DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

39.3.4.1. 8 位数据

当 DCMI_CR 中的 EDM[1:0]编程为“00”时，接口将捕获其输入（D[0:7]）的 8 个 LSB，并将其存储为 8 位数据。D[13:8]输入则忽略，在此情况下，要捕获 32 位字，摄像头接口需要花费 4 个 PIXCLK。

捕获的第一个数据字节放置在 32 位字的 LSB 位置，捕获的第四个数据字节放置在 32 位字的 MSB 位置。

表 39-2 捕获的数据字节在 32 位字(宽 8 位)中的位置排布

字节地址	31:24	23:16	15:8	7:0
0	$D_{n+3}[7:0]$	$D_{n+2}[7:0]$	$D_{n+1}[7:0]$	$D_n[7:0]$
4	$D_{n+7}[7:0]$	$D_{n+6}[7:0]$	$D_{n+5}[7:0]$	$D_{n+4}[7:0]$

39.3.4.2. 10 位数据

当 DCMI_CR 中的 EDM[1:0]编程为“01”时，摄像头接口将捕获其输入 D[0:9]的 10 位数据，并将其存储为 16 位字的 10 个最低有效位。DCMI_DR 寄存器中的其余最高有效位（11 到 15）将清零。因此，在此情况下，每两个像素时钟会生成一个 32 位数据字。

捕获的第一个数据字节放置在 32 位字的 LSB 位置，捕获的第四个数据字节放置在 32 位字的 MSB 位置。

表 39-3 捕获的数据字节在 32 位字(宽 10)中的位置排布

字节地址	31:26	25:16	15:10	9:0
0	0	$D_{n+1}[9:0]$	0	$D_n[9:0]$
4	0	$D_{n+3}[9:0]$	0	$D_{n+2}[9:0]$

39.3.4.3. 12 位数据位

当 DCMI_CR 中的 EDM[1:0]编程为“10”时，摄像头接口将捕获其输入 D[0:11]的 12 位数据，并将其存储为 16 位字的 12 个最低有效位。DCMI_DR 寄存器中的其余最高有效位（13 到 15）将清零。因此，在此情况下，每两个像素时钟会生成一个 32 位数据字。

捕获的第一个数据字节放置在 32 位字的 LSB 位置，捕获的第四个数据字节放置在 32 位字的 MSB 位置。

表 39-4 捕获的数据字节在 32 位字(宽 12)中的位置排布

字节地址	31:28	27:16	15:12	11:0
0	0	$D_{n+1}[11:0]$	0	$D_n[11:0]$
4	0	$D_{n+3}[11:0]$	0	$D_{n+2}[11:0]$

39.3.4.4. 14 位数据位

当 DCMI_CR 中的 EDM[1:0]编程为“11”时，摄像头接口将捕获其输入 D[0:13]的 14 位数据，并将其存储为 16 位字的 14 个最低有效位。DCMI_DR 寄存器中的其余最高有效位（14 到 15）将清零。因此，在此情况下，每两个像素时钟会生成一个 32 位数据字。

捕获的第一个数据字节放置在 32 位字的 LSB 位置，捕获的第四个数据字节放置在 32 位字的 MSB 位置。

表 39-5 捕获的数据字节在 32 位字(宽 14)中的位置排布

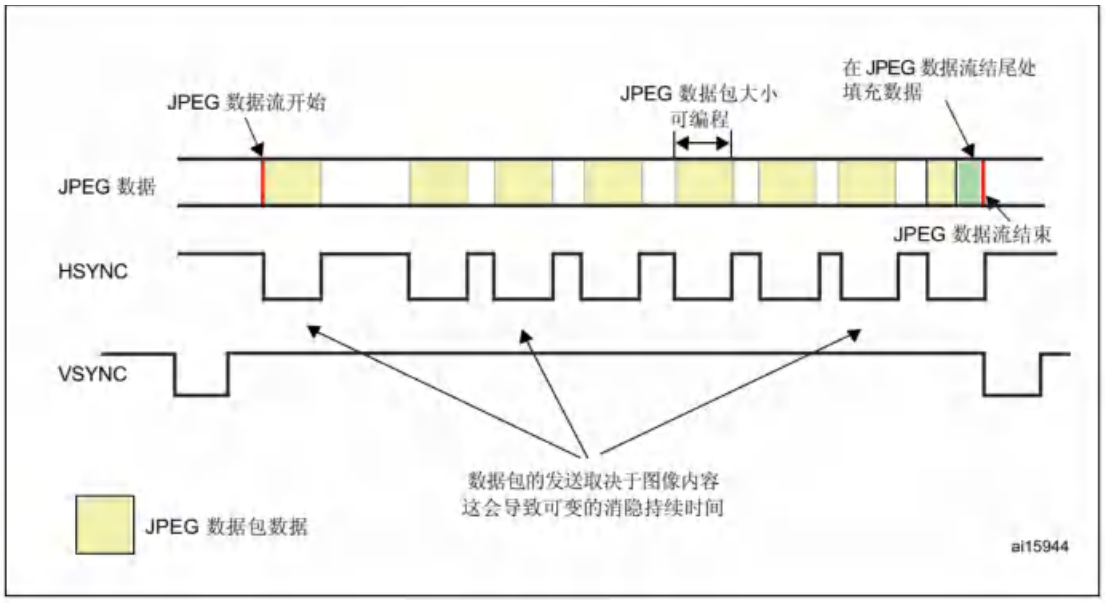
字节地址	31:30	29:16	15:14	13:0
0	0	$D_{n+1}[13:0]$	0	$D_n[13:0]$
4	0	$D_{n+3}[13:0]$	0	$D_{n+2}[13:0]$

39.3.5. 同步

数字摄像头接口支持内嵌码同步或硬件（HSYNC 和 VSYNC）同步。使用内嵌码同步时，由数字摄像头模块确保 0x00 和 0xFF 值仅用于同步（不同于数据中）。只有 8 位并行数据接口宽度支持内嵌码同步码。

对于压缩数据，DCMI 仅支持硬件同步模式。在这种情况下，VSYNC 指示图像的开始/结束，HSYNC 则用作“数据有效”信号。

图 39-4 时序图



39.3.5.1. 硬件同步模式

硬件同步模式下将使用两个同步信号（HSYNC/VSYNC）。根据摄像头模块/模式的不同，可能在水平/垂直同步期间内发送数据。由于系统会忽略 HSYNC/VSYNC 信号有效电平期间内接受的所有数据，HSYNC/VSYNC 信号相当于消隐信号。

为了正确地将图像传输到 DMA/RAM 缓冲区，数据传输将与 VSYNC 信号同步。选择硬件同步模式并启动捕获时，数据传输将与 VSYNC 信号的无效电平同步（开始下一帧时）。之后传输便可以连续执行，由 DMA 将连续帧传输到多个连续的缓冲区或一个具有循环特性的缓冲区。为了允许 DMA 管理连续帧，每一帧结束时都将激活 VSIF（垂直同步中断标志）。

39.3.5.2. 内嵌码同步模式

在此同步模式下，将使用数据流中嵌入的 32 位码来同步数据流。这些码使用数据中不再使用的值 0x00/0xFF。共有 4 种同步码类型，均采用 0xFF0000XY 格式。只有 8 位并行数据接口支持内嵌码同步。

注：在隔行扫描模式下，摄像头模块具有 8 个此类代码，因此，摄像头接口不支持隔行扫描模式（否则每帧数据的一般都会被丢弃）。

■ 模式 2

四个内嵌码可表示以下事件

- 帧开始 FS
- 帧结束 FE
- 行开始 LS
- 行结束 LE

4 个同步码采用的格式 0xFF0000XY 中的 XY 值可编程。

将 0xFF 编程为“帧结束”意味着所有除此之外的同步码都视为有效的帧结束同步码。在此模式下，一旦使能摄像头接口，将在首次出现帧结束 FE 同步码并且后接帧开始同步码之后开始捕获帧。

■ 模式 1

摄像头模式 1 是另一种编码。此模式与 ITU656 兼容。

这些同步码表示另一组事件：

- SAV（有效行） -行开始
- EAV（有效行） -行结束
- SAV（消隐）-帧间 消隐期内的行开始
- EAV（消隐）-帧间消隐期内的行结束

可以通过对同步码进行如下编程来支持此模式：

- FS <= 0xFF
- FE <= 0xFF
- LS <= SAV
- LE <= EAV

此外还针对帧/行开始和帧/行结束同步码实现了非屏蔽位功能。这样可以仅使用同步码种未被屏蔽的位进行比较。因此，可以选择一个位用于同步码的比较，来检测帧/行起始和帧/行结束。这意味着可以多个同步码表示帧/行的开始和结束。它们仅在未被屏蔽的的位相同即可。

示例：

FS = 0xA5

FS 的非屏蔽码=0x20

这种情况下，只需要比较数据码的第 4 位开检测是否是 FS 信号。

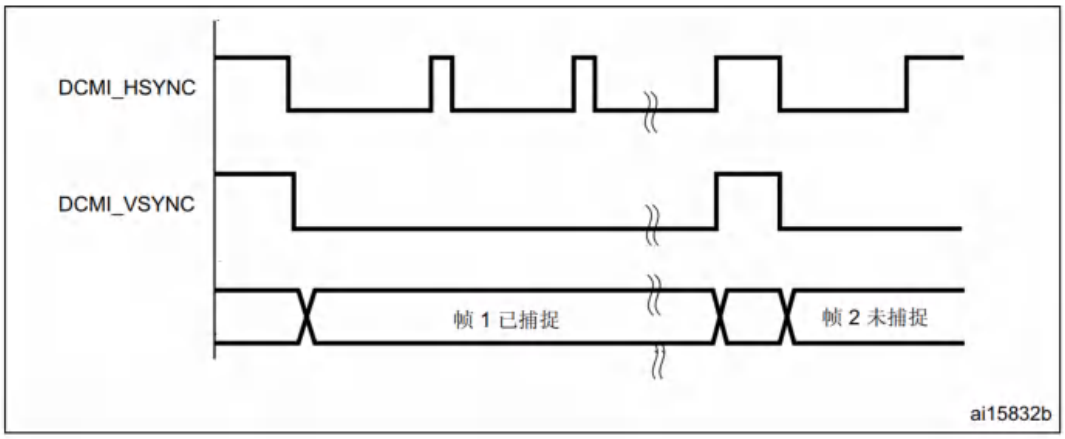
39.3.6. 捕获模式

此接口支持两种类型的捕获：快照（单帧）和连续采集。

39.3.6.1. 快照模式（单帧）

此模式下只捕获单帧（DCMI_CR 寄存器 CM=1）。在 DCMI_CR 种的 Capture 位置 1 后，该接口将等待系统检测帧开始，然后再对数据进行采样。收到完整的第一帧后，将自动禁止摄像头接口（DCMI_CR 中的 Capture 位清零）。如果使能相应中断，将生成中断 IT_FRAME

图 39-5 快照模式下的帧捕获波形



1）此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。

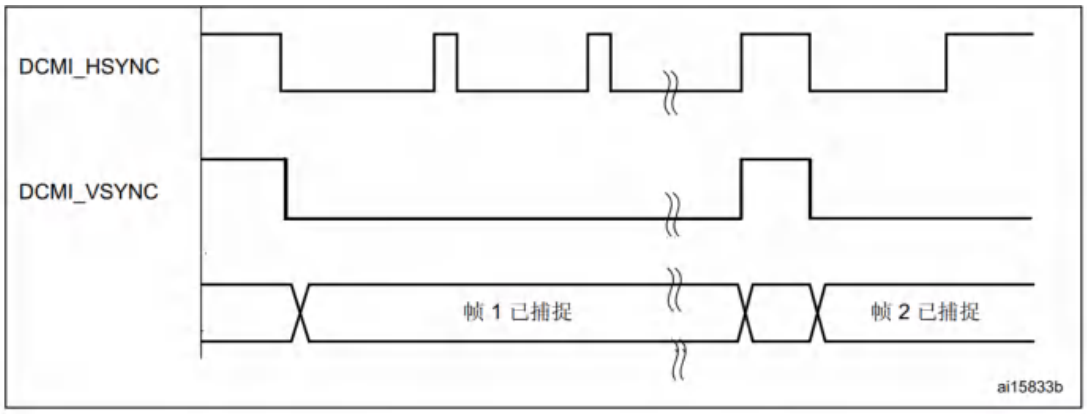
2）DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

39.3.6.2. 连续采集模式

在此模式下（DCMI_CR 中的 CM=0），一旦 DCMI_CR 中的 Capture 位置 1，将在下一个 VSYNC 或内嵌同步码帧开始同步码启动采集过程，具体取决于同步模式。该过程一直持续到 DCMI_CR 中的 Capture 位清零。Capture 位清零后，采集过程将持续到当前帧结束。

在连续采集模式下，可以通过配置 DCMI_CR 中的 FCRC 位来选择采集所有图形，或每 2 帧采集一次图片，或每四帧采集一个图片，以此降低帧捕获率。

图 39-6 连续采集模式下的帧捕获波形



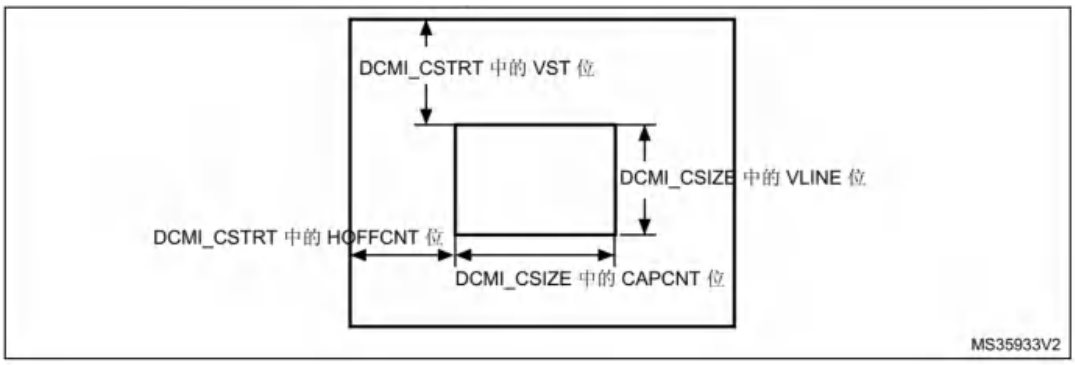
1）此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。

2）DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

39.3.7. 裁剪功能

摄像头接口可以使用裁剪功能从收到的图像中选择一个矩形窗口。起始（左上角）坐标和窗口大小（用像素时钟数表示的水平尺寸以及用行数表示的垂直尺寸）由两个 32 位寄存器 DCMI_CWSTRT 和 DCMI_CWSIZE 指定。窗口大小用像素时钟数（水平尺寸）和行数（垂直尺寸）表示。

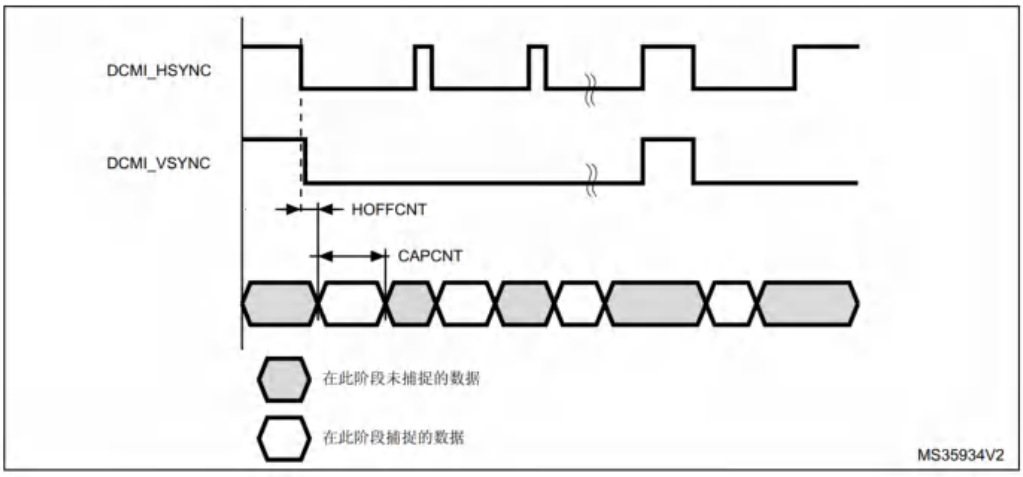
图 39-7 裁剪后窗口的坐标和大小



这些寄存器将捕获窗口的起点坐标指定为某一行（从 0 开始）和像素时钟数（从 0 开始），窗口大小则指定为行数和像素时钟数。CAPCNT 只能是 4 的倍数，才能通过 DMA 正确传输数据。

如果在捕获 DCMI_CWSIZE 寄存器中指定行数完成之前，VSYNC 信号已有效，那么捕获将停止，并且在中断使能时生成 IT_FRAME 中断。

图 39-8 数据捕获波形



1）此例中，DCMI_HSYNC 和 DCMI_VSYNC 的有效状态为 1。

2）DCMI_HSYNC 和 DCMI_VSYNC 的状态可同时发生更改。

39.3.8. JPEG 格式

要允许接受 JPEG 图像，必须将 DCMI_CR 寄存器中的 JPEG 位置 1。JPEG 图像不按行和帧存储，因此 VSYNC 信号用于启动捕获过程，而 HSYNC 则用作数据使能信号。行中包含的字节数可能不是 4 的倍数，因此处理此类情况时应十分谨慎，因此需要每次从捕获的数据形成一个 32 位字时，才生成一个 DMA 请求。检测到帧结束并且尚未凑成 32 位字时，将使用 0 进行填充，并触发一个 DMA 请求。

39.4. 数据格式说明

39.4.1. 数据格式

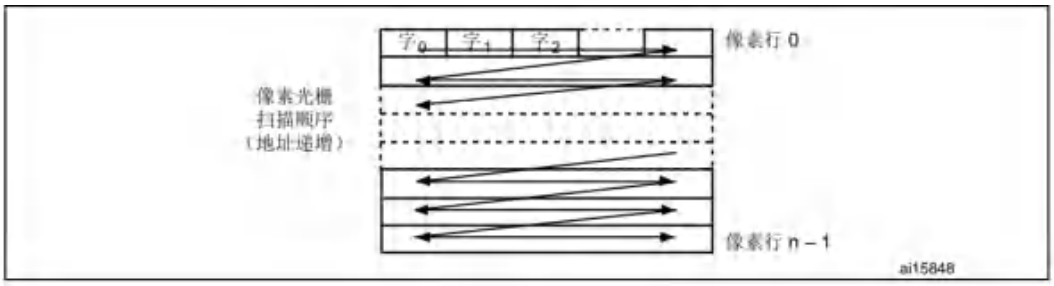
支持三种类型的数据：

- 8 位逐行视频：单色或原始拜尔格式
- YCbCr 4:2:2 逐行视频
- RGB565 逐行视频。采用 16 位（5 位蓝色，5 位红色，6 位绿色）编码的像素需要两个时钟周期传输。

压缩数据：JPEG

对于 B&W、YCbCr 或 RGB 数据，最大输入大小为 2048*2048 像素。JPEG 压缩模式无限制。对于单色、RGB 和 YCbCr，帧缓冲区以光栅模式存储。使用 32 位字，仅支持小端对齐格式。

图 39-9 像素光栅扫描顺序



39.4.2. 单色格式

特性：

- 光栅格式
- 每个像素 8 位

表 39-4 单色逐行视频格式的数据存储

字节地址	31:24	23:16	15:8	7:0
0	n + 3	n + 2	n + 1	n
4	n + 7	n + 6	n + 5	n + 4

39.4.3. RGB 格式

特性：

- 光栅格式
- RGB
- 一个缓冲区交替存储 RGB 信号：BRGBRGBRG
- 对显示输出的优化

RGB 平面格式与标准的 OS 帧缓冲显示格式兼容。

仅支持 16BPP（每个像素 16 位）：RGB565（每 32 位字表示 2 个像素）

不支持 24BPP（托盘化格式）和灰度格式。像素按照光栅扫描顺序进行存储，即从顶部像素行到底部像素行，从像素行的左侧到右侧。像素分量 R（红色）、G（绿色）和 B（蓝色）。所有分量的空间分辨率都相同。一帧数据中各个分量交替间隔存储。

表 39-5 以 RGB 逐行视频格式存储数据

字节地址	31:27	26:21	20:16	15:11	10:5	4:0
0	红色 n+1	绿色 n+1	蓝色 n+1	红色 n	绿色 n	蓝色 n
4	红色 n+4	绿色 n+3	蓝色 n+3	红色 n+2	绿色 n+2	蓝色 n+2

39.4.4. YCbCr 格式

特性：

- 光栅格式
- YCbCr 4:2:2
- 一个缓冲区交替存储 Y、Cb 和 Cr。CbYCrYCbYCr 等。

像素分量包括 Y（亮度）、Cb 和 Cr（蓝色色度和红色色度）。每个分量都采用 8 位进行编码。亮度和色度交替存储在一起。

表 39-6 YCbCr 逐行视频格式下的数据存储

字节地址	31:24	23:16	15:8	7:0
0	Y n+1	Cr n	Y n	Cb n
4	Y n+3	Cr n+2	Y n+2	Cb n+2

39.4.5. YCbCr 格式-仅含 Y 分量

特性：

- 光栅格式
- YCbCr 4:2:2
- 缓冲区仅包含 Y 分量信息-单色图形

像素分量包括 Y（亮度）、Cb 和 Cr（蓝色色度和红色色度）。在此模式下，将丢弃色度信息。仅存储每个像素采用 8 位进行编码的亮度分量。

结果为单色图形，其分辨率与原始 YCbCr 数据的分辨率相同。

表 39-7 YCbCr 逐行视频格式下的数据存储--Y 分量提取模式

字节地址	31:24	23:16	15:8	7:0
0	Y n+3	Y n+2	Y n+1	Y n
4	Y n+7	Y n+6	Y n+5	Y n+4

39.5. 信号描述

SYMBOL	TYPE	DESCRIPTION

D[0:13]	Input	数据输入
HSYNC	Input	水平同步（行同步）输入
VSYNC	Input	垂直同步（场同步）输入
PIXCLK	Input	像素时钟输入

39.6. DCMI 中断

有五种中断源。所有中断都可以通过软件屏蔽。全局中断（IT_DCMI）是所有单个中断的逻辑或运算所得结果。

中断名称	中断事件
IT_LINE	表示行结束
IT_FRAME	表示帧捕获结束
IT_OVR	表示接受的数据发生溢出错误
IT_VSYNC	表示同步帧
IT_ERR	表示内嵌码同步帧检测期间检测到错误
IT_DCMI	以上中断的逻辑或结果

39.7. DCMI 寄存器描述

39.7.1. 寄存器列表

DCMI 寄存器基地址：0x50050000

偏移	名称	复位值	描述
0x00	DCMI_CR	0x00000000	控制寄存器
0x04	DCMI_SR	0x00000000	状态寄存器
0x08	DCMI_RIS	0x00000001	原始中断状态寄存器
0x0C	DCMI_IER	0x00000000	中断使能寄存器
0x10	DCMI_MIS	0x00000000	屏蔽中断状态寄存器
0x14	DCMI_ICR	0x00000000	中断清零寄存器
0x18	DCMI_ESCR	0x00000000	内嵌同步码寄存器
0x1C	DCMI_ESUR	0x00000000	内嵌码同步取消屏蔽寄存器
0x20	DCMI_CWSTRT	0x00000000	裁剪窗口起点
0x24	DCMI_CWSIZE	0x00000000	裁剪窗口大小
0x28	DCMI_DR	0x00000000	数据寄存器

39.7.2. 模式寄存器(DCMI_CR: 00h)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	ENABLE	R/W	0	DCMI 使能 0：禁止 DCMI 1：使能 DCMI
13	RSV	-	-	保留
12	BURST4	R/W	0	使能 BURST4 传输模式
11:10	EDM	R/W	0	扩展数据模式 00：接口每个像素时钟捕获 8 位数据 01：接口每个像素时钟捕获 10 位数据 10：接口每个像素时钟捕获 12 位数据 11：接口每个像素时钟捕获 14 位数据
9:8	FCRC	R/W	0	帧捕获率控制。仅在连续采集模式下有效，快照模式下无效。 00：捕获所有帧 01：每隔一帧捕获一次（带宽降低 50%） 10：每隔三帧捕获一次（捕获降低 75%） 11：保留

7	VSPOL	R/W	0	垂直同步极性 0：VSYNC 低电平有效 1：VSYNC 高电平有效
6	HSPOL	R/W	0	水平同步极性 0：HSYNC 低电平有效 1：HSYNC 高电平有效
5	PCKPOL	R/W	0	像素时钟极性 0：下降沿有效 1：上升沿有效
4	ESS	R/W	0	内嵌码同步选择 0：硬件同步：数据捕获（帧/行开始/停止）由 HSYNC/VSYNC 信号同步 1：内嵌码同步：数据捕获由数据流中嵌入的同步码同步。 注意：仅对 8 位并行数据有效。ESS 位等于 1，将忽略 HSPOL/VSPOL。JPEG 模式下禁止此位。
3	JPEG	R/W	0	JPEG 格式 0：未经压缩的视频格式 1：此位用于 JPEG 数据传输。HSYNC 信号用作数据使能信号。此模式下无法裁剪和内嵌码同步功能（ESS 位）
2	CROP	R/W	0	裁剪功能 0：捕获完整图像。这种情况下，图像帧包含的字节总数应该为 4 的倍数。 1：仅捕获裁剪寄存器所指定的窗口中的数据。如果窗口大小超过图片大小，则仅捕获图片大小。
1	CM	R/W	0	捕获模式 0：连续采集模式-收到的数据将通过 DMA 传输到目标存储区。缓冲区位置和模式（线性或循环缓冲区）由系统 DMA 控制。 1：快照模式（单帧）-一旦激活，接口将等待帧开始，然后通过 DMA 传输单帧。帧结束将自动复位 CAPTURE 位。
0	CAPTURE	R/W	0	使能捕获 0：禁止捕获 1：使能捕获 摄像头接口等待第一帧开始，然后生成一个 DMA 请求将收到的数据传输到目标存储器中。 在快照模式下，收到的第一帧结束时将自动使 CAPTURE 位清零。 在连续采集模式下，如果在执行捕获操作时通过软件将此位清零，则帧结束后此位的清零才生效。 注意：使能此位之前，应对 DMA 控制器和所有 DCMI 配置寄存器进行适当的编程。

39.7.3. 控制寄存器(DCMI_SR: 04h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	FNE	RO	0	FIFO 非空 0：FIFO 为空 1：FIFO 包含有效数据
1	VSYNC	RO	0	此位指示编程了适当极性的 VSYNC 引脚的状态。 使用内嵌码同步时，此位含义如下： 0：有效帧 1：在帧之间同步。 如果使用内嵌码同步，则仅当 DCMI_CR 的 CAPTURE 位置 1 时，此位才有意义。
0	HSYNC	RO	0	此位指示编程了适当极性的 HSYNC 引脚的状态。 使用内嵌码同步时，此位含义如下： 0：有效行 1：在行之间同步。 如果使用内嵌码同步，则仅当 DCMI_CR 的 CAPTURE 位置 1 时，此位才有意义。

39.7.4. 原始中断状态寄存器(DCMI_RIS: 08h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	LINE_RIS	RO	0	行原始中断状态 当 HSYNC 信号从无效状态更改为有效状态时，此位置 1。即使行无效也会变成高电平。 如果使用内嵌码同步，则仅当 DCMI_CR 的 CAPTURE 位置 1 时，此位才置。向 DCMI_ICR 的 LINE_ISC 位写入 1 可清零此位。
3	VSYNC_RIS	RO	0	VSYNC 原始中断状态 当 VSYNC 信号从无效状态更改为有效状态时，此位置 1。即使行无效也会变成高电平。 如果使用内嵌码同步，则仅当 DCMI_CR 的 CAPTURE 位置 1 时，此位才置。向 DCMI_ICR 的 VSYNC_ISC 位写入 1 可清零此位。

2	ERR_RIS	RO	0	同步错误原始中断状态 0：未检测到同步错误 1：未按正确顺序接受内嵌码同步字符 此位仅在内嵌码同步模式下有效。向 DCMI_ICR 的 ERR_ISC 位写入 1 可清零此位。
1	OVR_RIS	RO	0	溢出原始中断状态 0：未发生数据缓冲区溢出 1：发生数据缓冲区溢出，数据 FIFO 损坏。 向 DCMI_ICR 的 OVR_ISC 位写入 1 可清零此位。
0	FRAME_RIS	RO	1	捕获完成原始中断状态 0：没有新的捕获数据 1：已捕获一帧 对一帧数据或裁剪窗口内的数据捕获完毕后，此位置 1。 如果捕获裁剪窗口，则将在裁剪窗口的最后一行结束时将此位置 1。即使捕获的帧位空（例如，窗口超过帧范围），此位也将置 1。 向 DCMI_ICR 的 FRAME_ISC 位写入 1 可清零此位。

39.7.5. 中断使能寄存器(DCMI_IER: 0Ch)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	LINE_IE	R/W	0	行中断使能 0：接受到行时不生成中断 1：完全接受到行时生成一个中断
3	VSYNC_IE	R/W	0	VSYNC 中断使能 0：不生产中断 1：VSYNC 每次从无效电平变为有效时都生成一个中断。
2	ERR_IE	R/W	0	同步错误中断使能 0：不生成中断 1：如果未按正确顺序接受码同步代码，则生成一个中断。 注：此位仅适用于内嵌码同步模式。
1	OVR_IE	R/W	0	溢出中断使能 0：不生成中断 1：如果 DMA 无法在接受到新数据（32 位）之前传输上一个数据，则生成一个中断。

0	FRAME_IE	R/W	0	捕获完成中断使能 0：不生成中断 1：接受完成一帧数据或裁剪窗口内的数据，则生成一个中断。

39.7.6. 屏蔽中断状态寄存器(DCMI_MIS: 10h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	LINE_MIS	RO	0	行屏蔽中断状态 0：无中断 1：有中断
3	VSYNC_MIS	RO	0	VSYNC 屏蔽中断状态 0：无中断 1：有中断
2	ERR_MIS	RO	0	同步错误屏蔽中断使能 0：无中断 1：有中断
1	OVR_MIS	RO	0	溢出屏蔽中断使能 0：无中断 1：有中断
0	FRAME_MIS	RO	0	捕获完成屏蔽中断使能 0：禁止。 1：使能。

39.7.7. 中断清零寄存器(DCMI_ICR: 14h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	LINE_ISC	WO	0	对此位写 1 将清除 DCMI_RIS 寄存器的 LINE_RIS 位。
3	VSYNC_ISC	WO	0	对此位写 1 将清除 DCMI_RIS 寄存器的 VSYNC_RIS 位。
2	ERR_ISC	WO	0	对此位写 1 将清除 DCMI_RIS 寄存器的 ERR_RIS 位。
1	OVR_ISC	WO	0	对此位写 1 将清除 DCMI_RIS 寄存器的 OVR_RIS 位。

0	FRAME_ISC	WO	0	对此位写 1 将清除 DCMI_RIS 寄存器的 FRAME_RIS 位。

39.7.8. 内嵌同步码寄存器(DCMI_ESCR: 18h)

位域	名称	属性	复位值	描述
31:24	FEC	R/W	0	帧结束分隔码 此字节指定帧结束分隔码。由 4 个字节组成，0xFF 0x00 0x00 FEC。如果将 FEC 编程为 0xFF，所有未使用的其他代码（0xFF0000XY）都将视为帧结束分隔符。
23:16	LEC	R/W	0	行结束分隔码 此字节指定行结束分隔码。由 4 个字节组成，0xFF 0x00 0x00 LEC。
15:8	LSC	R/W	0	行开始分隔码 此字节指定行开始分隔码。由 4 个字节组成，0xFF 0x00 0x00 LSC。
7:0	FSC	R/W	0	帧开始分隔码 此字节指定帧开始分隔码。由 4 个字节组成，0xFF 0x00 0x00 FSC。如果将 FSC 编程为 0xFF，将检测不到任何帧开始分隔符。但在 FEC 代码后第一次出现 LSC 时将视为帧分隔符的开始。

39.7.9. 内嵌同步码寄存器(DCMI_ESUR: 1Ch)

位域	名称	属性	复位值	描述
31:24	FEU	R/W	0	帧结束分隔符取消屏蔽 此字节指定帧结束分隔码的屏蔽。 0：将帧结束分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 FEC 字节中的相应位。 1：将帧结束分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 FEC 字节中的相应位。
23:16	LEU	R/W	0	行结束分隔符取消屏蔽 此字节指定行结束分隔码的屏蔽。 0：将行结束分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 LEC 字节中的相应位。 1：将行结束分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 LEC 字节中的相应位。

15:8	LSU	R/W	0	行开始分隔符取消屏蔽 此字节指定行开始分隔码的屏蔽。 0：将行开始分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 LSC 字节中的相应位。 1：将行结束分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 LSC 字节中的相应位。
7:0	FSU	R/W	0	帧开始分隔码取消屏蔽 此字节指定帧开始分隔码的屏蔽。 0：将帧开始分隔符与收到的数据进行比较时，将屏蔽 DCMI_ESCR 中 FSC 字节中的相应位。 1：将帧开始分隔符与收到的数据进行比较时，将比较 DCMI_ESCR 中 FSC 字节中的相应位。

39.7.10. 裁剪窗口起点寄存器(DCMI_CWSTRT: 20h)

位域	名称	属性	复位值	描述
31:29	RSV	--	--	保留
28:16	VST	R/W	0	窗口开始的行数。 图像捕获从此行开始，对之前的数据不予捕获。
15:14	RSV	--	--	保留
13:0	HOFFCNT	R/W	0	窗口开始的水平方向位移。 窗口行内，每行在捕获数据前需空出的像素时钟个数。

39.7.11. 裁剪大小寄存器(DCMI_CWSIZE: 24h)

位域	名称	属性	复位值	描述
31:29	RSV	--	--	保留
28:16	VLINE	R/W	0	垂直行计数 窗口内包含的行数
15:14	RSV	--	--	保留
13:0	CAPCNT	R/W	0	捕获计数 窗口内要捕获的像素时钟数

39.7.12. 数据寄存器(DCMI_DR: 28h)

位域	名称	属性	复位值	描述
31:0	DR	RO	0	捕获到的数据

40. LCD-TFT 控制器（LTDC）

40.1. 概述

LCD-TFT（液晶显示器- 薄膜晶体管） 显示器控制器提供并行数字 RGB（红色、绿色、蓝色）以及水平同步、垂直同步、像素时钟和数据使能信号，这些信号直接输出到不同 LCD 和 TFT 面板的接口。

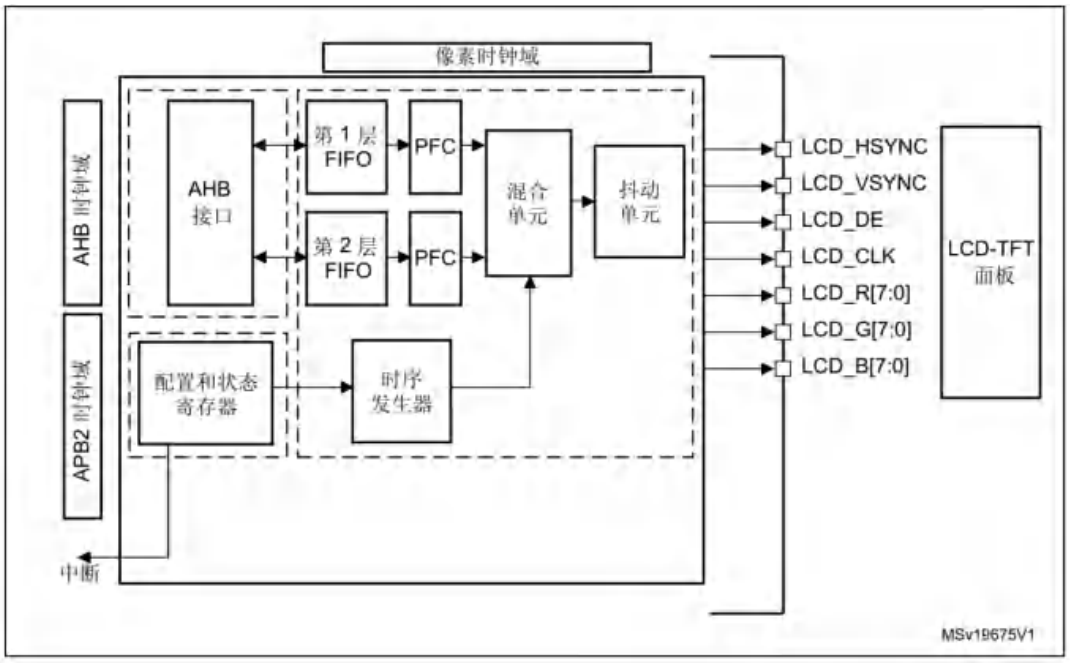
40.2. 主要特性

- 24 位 RGB 并行像素输出，每像素 8 位（RGB888）
- 2 个带有专用 FIFO 的显示层（FIFO 深度 256x32 位）
- 查色表（CLUT），每层高达 256 种颜色（256x24 位）
- 支持高达 XGA（1024x768）的分辨率
- 可针对不同显示面板编程时序
- 可编程背景色
- 可编程 HSyn、VSync 和数据使能信号的极性
- 每层有多达 8 个输入颜色格式可供选择
  - ARGB8888
  - RGB888
  - RGB565
  - ARGB1555
  - ARGB4444
  - L8（8 位 Luminance 或 CLUT）
  - AL44（4 位 alpha + 4 位 Luminance）
  - AL88（8 位 alpha + 8 位 Luminance）
- 每通道的低位采用伪随机抖动输出
  - 红色、绿色、蓝色的抖动宽度为 2 位。
- 使用 alpha 值（每像素或常数）在两层之间灵活混合
- 色健（透明颜色）
- 可编程窗口位置和大小
- 支持薄膜晶体管（TFT）彩色显示器
- AHB 主接口支持 16 个字的突发
- 高达 4 个可编程中断事件

40.3. 功能描述

40.3.1. LTDC 框图

图 40-1 LTDC 框图



层 FIFO：每层一个 256x32 位 FIFO。

PFC：执行像素格式转换的格式转换器，从层的所选输入像素格式转换为字。

AHB 接口：用于将数据从存储器传输到 FIFO。

40.3.2. LTDC 复位和时钟

LCD-TFT 控制器外设使用 3 个时钟域：

- AHB 时钟域（HCLK）：用于将将数据从存储器传输到 FIFO 层。
- APB 时钟域（PCLK）：用于配置寄存器
- 像素时钟域（LCD_CLK）：用于生成 LCD-TFT 接口信号。LCD_CLK 输出应按照面板要求配置。LCD_CLK 通过 PLLSAI 进行配置。

通过将 RCC_APB2RSTR 寄存器中的相应位置 1 可将 LCD 控制器复位，这将复位三个时钟域。

40.3.3. LCDC 引脚和信号接口

表 40-1 LCD-TFT 引脚和信号接口

LCD-TFT 信号	I/O	说明
LCD_CLK	O	时钟输出
LCD_HSYNC	O	水平同步
LCD_VSYNC	O	垂直同步
LCD_DE	O	数据使能
LCD_R[7:0]	O	数据：8 位红色数据
LCD_G[7:0]	O	数据：8 位绿色数据
LCD_B[7:0]	O	数据：8 位蓝色数据

必须通过用户程序配置 LCD-TFT 控制器引脚。未使用的引脚可用于其他功能。

对于高达 24 位（RGB888）的 LTDC 输出，如果使用低于 8bpp 的像素深度将 RGB565 或者 RGB666 输出到 16 位或者 18 位显示器，则 RGB 显示数据线必须连接到 LCD-TFT 控制器 RGB 数据线的 MSB。

40.4. LTDC 可编程参数

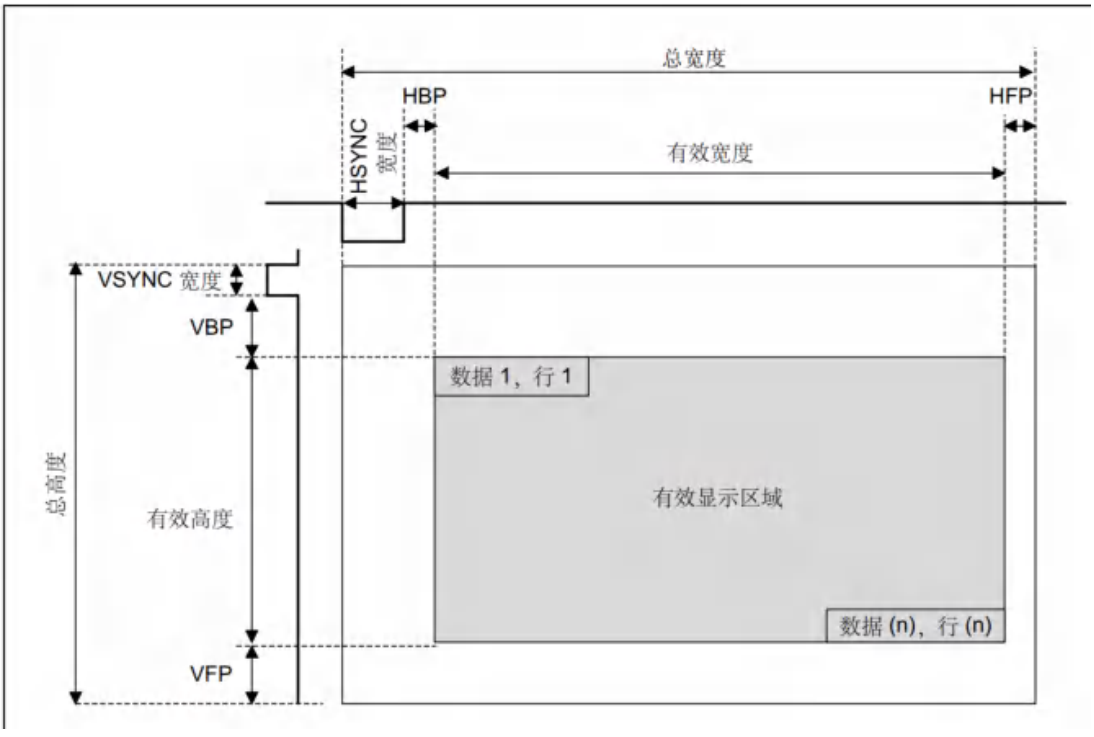
LCD-TFT 控制器提供灵活的可配置参数。其可通过 LTDC_GCR 寄存器使能或禁止。

40.4.1. LTDC 全局配置参数

40.4.1.1. 同步时序

同步时序发生器模块生成的可配置时序参数，该模块生产水平和垂直同步时序面板信号、像素时钟和数据使能信号。

图 40-2 LCD-TFT 同步时序



HBP 和 HFP 分别为水平后沿周期和水平前沿周期。

VBP 和 VFP 分别为垂直后沿周期和垂直前沿周期。

LCD-TFT 可编程同步时序包括：

- HSYNC 和 VSYNC 宽度：水平和垂直同步宽度，通过编程 LTDC_SSCR 寄存器中的 HSYNC Width-1 和 VSYNC Width-1 的值进行配置。
- HBP 和 VBP ：水平和垂直同步后沿宽度，通过编程 LTDC_BPCR 寄存器中的累加值 HSYNC Width + HBP -1 和累加值 VSYNC Width + VBP -1 进行配置。
- 有效宽度和有效高度：有效宽度和有效高度通过编程 LTDC_AWCR 寄存器中的累加值 HSYNC Width + HBP + Active Width -1 和累加值 VSYNC Width + VBP + Active Width -1 进行配置（仅支持最高 1024x768）。
- 总宽度：总宽度通过编程 LTDC_TWCR 寄存器中的累加值 HSYNC Width + HBP + Active Width + HFP - 1 进行配置。HFP 为水平前沿周期。
- 总高度：总高度通过编程 LTDC_TWCR 寄存器中的累加值 VSYNC Height + VBP + Active Height + VFP - 1 进行配置。VFP 为水平前沿周期。

使能 LTDC 时，产生时序以 X/Y=0/0 位置作为垂直同步区域中的第一个水平同步像素，随后是后沿、有效数据显示区域和前沿。

禁止 LTDC 时，时序发生器模块复位为 X=总宽带-1、Y=总高度-1，并在垂直同步阶段和 FIFO 刷新前保持上一个像素。因此，仅连续输出消隐数据。

同步时序配置示例：

TFT-LCD 时序(应从面板数据手册中提取)：

- 水平和垂直同步宽度：0x8 像素，0x4 行
- 水平和垂直后沿：0x7 像素，0x2 行
- 有效宽度和有效高度：0x280 像素，0x1E0 行(640x480)
- 水平前沿：0x6 像素
- 垂直前沿：0x2 行

LTDC 时序寄存器中编程的值将为：

- LTDC_SSCR 寄存器：将编程为 0x00070001。(HSW[11:0]为 0x7 且 VSH[10:0]为 0x3)
- LTDC_BPCR 寄存器：将编程为 0x000E0005。(AHBP[11:0]为 0xE(0x8+0x6)且 AVBP[10:0]为 0x5(0x4+0x1))
- LTDC_AWCR 寄存器：将编程为 0x028E01E5。(AAW[11:0]为 0x28E(0x8 + 0x7 + 0x27F)且 AAH[10:0]为 0x1E5(0x4 + 0x2 + 0x1DF))
- LTDC_TWCR 寄存器：将编程为 0x000000294。(TOTALW[11:0] 0x294(0x8 + 0x7+ 0x280 + 0x5)
- LTDC_THCR 寄存器：将编程为 0x0000001E7。(TOTALH[10:0] 为 0x1E7(0x4 + 0x2 + 0x1E0 + 1))

1. 可编程极性

水平和垂直同步、数据使能和像素时钟输出信号的极性可通过 LTDC_GCR 寄存器编程为高电平有效或低电平有效。

2. 背景色

恒定的背景色（RGB888）可通过 LTDC_BCCR 寄存器编程。它用于与底层混合。

3. 抖动

使用 LFSR 的伪随机抖动技术用于向各个像素颜色通道值（R、G 或 B）添加小的随机值（阈值）。从而当 18 位显示器上显示 24 位数据时，可在某些情况下 MSB 进行舍入操作。因此，抖动技术用于对各帧中不同的数据进行舍入操作。

伪随机抖动技术的过程是将 LSB 与阈值比较，并在 LSB 部分 >= 阈值时，仅向 MSB 部分加 1。一旦应用抖动技术，通常会减少 LSB。

添加的伪随机值得宽度为每个颜色通道 2 位；红色 2 位、绿色 2 位及蓝色 2 位。

使能 LCD-TFT 控制器后，LFSR 以第一个有效像素开始运行，并且即使在消隐周期内核抖动关闭时也保持运行状态。如果禁止 LTDC，LFSR 将复位。

那通过 LTDC_GCR 寄存器实时开启和关闭抖动。

4. 重载影子寄存器

一些配置寄存器执行影子操作。对活动寄存器执行写操作时，或在 LTDC_SRCR 寄存器配置阶段之后得垂直消隐周期开始时，可将影子寄存器立即重载到活动寄存器中。如果选择了立即重载配置，则只应在所有新寄存器完成写操作后激活重载。

不应在重载完成前再次修改影子寄存器。读取影子寄存器将返回实际有效值。新写入得只能在重载发生后读取。

如果在 LTDC_IER 寄存器中相应使能，则可产生寄存器重载中断。

40.4.1.2. 层可编程参数

最多可单独使能、禁止和配置两个层。层显示顺序固定，即自上而下。如果使能两个层，则层 2 为顶部显示窗口。

1. 窗口

可为每个层定位和调整大小，各个层必须位于有效显示区域内。

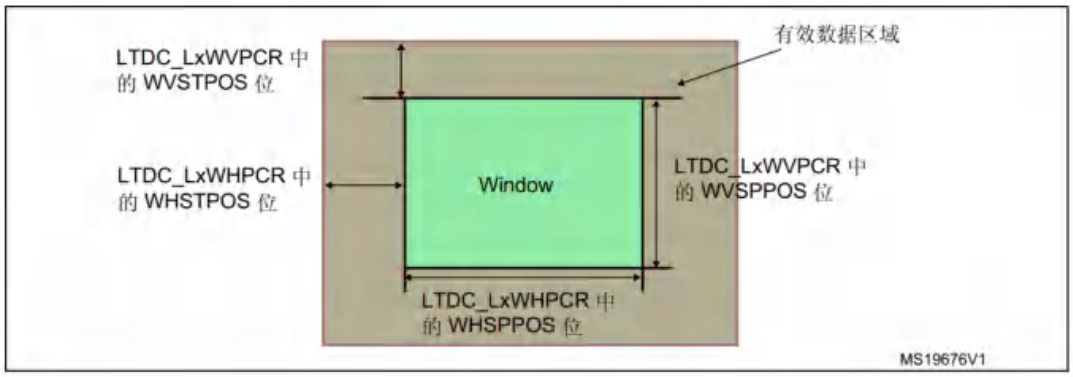
窗口位置和大小通过左上和右下的 XY 位置以及包含同步、后沿大小和有效数据区域的内部时序发生器配置。

可编程层位置和大小定义了一行中的第一个/最后一个可见像素和窗口中的第一个/最后一个可见行。它允许显示完整的图像帧，也允许只显示图像帧的一部分。

层中的第一个和最后一个可见像素通过 LTDC_LxWHPCR 寄存器中的 WHSTPOS[11:0]和 WHSPPOS[11:0]进行设置。

层中的第一个和最后一个可见行通过 LTDC_LxWVPCR 寄存器中的 WVSTPOS[10:0]和 WVSPPOS[10:0]进行设置。

图 40-3 层窗口可编程参数



2. 像素输入格式

可编程像素格式用于层的帧缓冲区中存储的数据。可通过 LTDC_LxPFCR 寄存器为每个层多达 8 个输入像素格式。

像素数据从帧缓冲区中读取，随后按照以下方式转换为内部 8888（ARGB）格式。宽度低于 8 位的分量通过位重复扩展到 8 位。所选位范围多次拼接，直至其超过 8 位。在得到的向量中 ，选择高 8 位。例如：5 位 RGB565 红色通道将变为 43210432。

表 40-2 像素数据映射与颜色格式的关系

ARGB8888			
@+3 A_x[7:0]	@+2 R_x[7:0]	@+1 G_x[7:0]	@ B_x[7:0]
@+7 A_{x+1}[7:0]	@+6 R_{x+1}[7:0]	@+5 G_{x+1}[7:0]	@+4 B_{x+1}[7:0]
RGB888			
@+3 B_{x+1}[7:0]	@+2 R_x[7:0]	@+1 G_x[7:0]	@ B_x[7:0]
@+7 G_{x+2}[7:0]	@+6 B_{x+2}[7:0]	@+5 R_{x+1}[7:0]	@+4 G_{x+1}[7:0]
RGB565			
@+3 R_{x+1}[4:0] G_{x+1}[5:3]	@+2 G_{x+1}[2:0] B_{x+1}[4:0]	@+1 R_x[4:0] G_x[5:3]	@ G_x[2:0] B_x[4:0]
@+7 R_{x+3}[4:0] G_{x+3}[5:3]	@+6 G_{x+3}[2:0] B_{x+3}[4:0]	@+5 R_{x+2}[4:0] G_{x+2}[5:3]	@+4 G_{x+2}[2:0] B_{x+2}[4:0]

ARGB1555			
@+3 $A_{x+1}[0]R_{x+1}[4:0]$ $G_{x+1}[4:3]$	@+2 $G_{x+1}[2:0]$ $B_{x+1}[4:0]$	@+1 $A_x[0]$ $R_x[4:0]$ $G_x[4:3]$	@ $G_x[2:0]$ $B_x[4:0]$
@+7 $A_{x+3}[0]R_{x+3}[4:0]$ $G_{x+3}[4:3]$	@+6 $G_{x+3}[2:0]$ $B_{x+3}[4:0]$	@+5 $A_{x+2}[0]R_{x+2}[4:0]G_{x+2}[4:3]$	@+4 $G_{x+2}[2:0]$ $B_{x+2}[4:0]$
ARGB4444			
@+3 $A_{x+1}[3:0]R_{x+1}[3:0]$	@+2 $G_{x+1}[3:0]$ $B_{x+1}[3:0]$	@+1 $A_x[3:0]$ $R_x[3:0]$	@ $G_x[3:0]$ $B_x[3:0]$
@+7 $A_{x+3}[3:0]R_{x+3}[3:0]$	@+6 $G_{x+3}[3:0]$ $B_{x+3}[3:0]$	@+5 $A_{x+2}[3:0]R_{x+2}[3:0]$	@+4 $G_{x+2}[3:0]$ $B_{x+2}[3:0]$
ARGB8888			
L8			
@+3 $L_{x+3}[7:0]$	@+2 $L_{x+2}[7:0]$	@+1 $L_{x+1}[7:0]$	@ $L_x[7:0]$
@+7 $L_{x+7}[7:0]$	@+6 $L_{x+6}[7:0]$	@+5 $L_{x+5}[7:0]$	@+4 $L_{x+4}[7:0]$
AL44			
@+3 $A_{x+3}[3:0]$ $L_{x+3}[3:0]$	@+2 $A_{x+2}[3:0]$ $L_{x+2}[3:0]$	@+1 $A_{x+1}[3:0]$ $L_{x+1}[3:0]$	@ $A_x[3:0]$ $L_x[3:0]$
@+7 $A_{x+7}[3:0]$ $L_{x+7}[3:0]$	@+6 $A_{x+6}[3:0]$ $L_{x+6}[3:0]$	@+5 $A_{x+5}[3:0]$ $L_{x+5}[3:0]$	@+4 $A_{x+4}[3:0]$ $L_{x+4}[3:0]$
AL88			
@+3 $A_{x+1}[7:0]$	@+2 $L_{x+1}[7:0]$	@+1 $A_x[7:0]$	@ $L_x[7:0]$
@+7 $A_{x+3}[7:0]$	@+6 $L_{x+3}[7:0]$	@+5 $A_{x+2}[7:0]$	@+4 $L_{x+2}[7:0]$

3. 查色表（CLUT）

可在运行时通过 LTDC_LxCR 寄存器为每个层使能 CLUT，CLUT 仅在使用 L8、AL44 和 AL88 输入像素格式时适用于索引色。

首先，CLUT 必须加载用于替换相应像素（索引色）的 RGB 值。每个颜色（RGB 值）在 CLUT 内都有自己对应得地址。

R、G 和 B 值及其各自得地址均通过 LTDC_LxCLUTWR 寄存器编程。

- 在使用 L8 和 AL88 输入像素格式时，CLUT 必须加载 256 个颜色。各颜色得地址在 LTDC_LxCLUTWR 寄存器的 CLUTADD 位中配置。
- 在使用 AL44 输入像素格式时，CLUT 必须仅加载 16 个颜色。各颜色的地址必须通过 4 位 L 通道重复为 8 位进行填充，具体如下：
  - L0（索引色 0），地址 0x00 处。
  - L1，地址 0x11 处。
  - L15，地址 0xFF 处。

4. 颜色帧缓冲区地址

每个层的颜色帧缓冲区都有一个起始地址，该地址通过 LTDC_LxCFBAR 寄存器进行配置。当使能某个层时，将从颜色帧缓冲区中获取该数据。

5. 颜色帧缓冲区长度

每层均设置颜色帧缓冲区的总行长（单位为字节）和行数，二者可分别通过 LTDC_LxCFBLR 和 LTDC_LxCFBLNR 寄存器进行配置。

行长和行数的设置用于阻止超过帧缓存结尾的数据被预取到层对应的 FIFO 中。

- 如果设置为低于所需字节，则会产生 FIFO 下溢中断。
- 如果设置为高于实际所需字节，则将丢弃从 FIFO 中读取的无用数据。无用数据不会显示。

6. 颜色帧缓冲区间隔

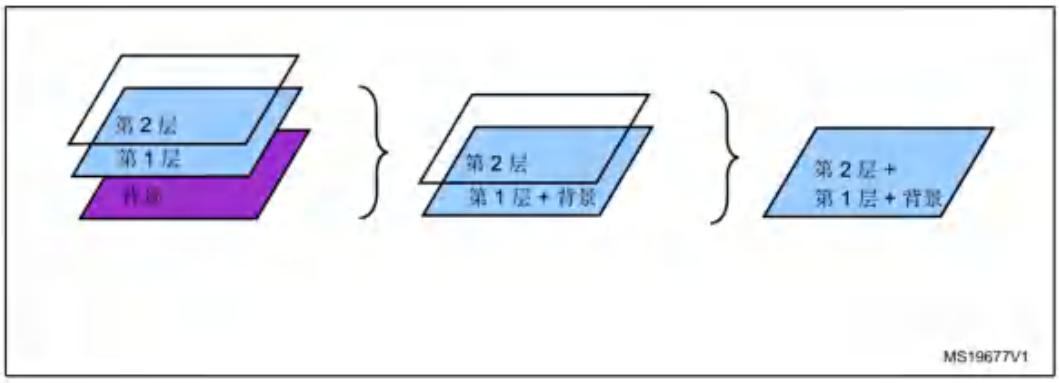
每层的颜色帧缓冲区具有可配置间距，此间距是一行的开始与下一行开始的距离（以字节为单位）。它通过 LTDC_LxCFBLR 寄存器配置。

7. 层混合

混合操作始终有效，两层可按照 LTDC_LxBFCR 寄存器中配置的混合系数进行混合。

混合顺序固定，即由下至上。如果使能了两层，首先第 1 层将与背景色混合，随后第 2 层与第 1 层和背景的混合颜色结果再次混合。

图 40-4 两层与背景混合



8. 默认颜色

每层可具有 ARGB 格式的默认颜色，该颜色在定义的层窗口外使用或在层禁止时使用。

默认颜色通过 LTDC_LxDCCR 寄存器配置。

始终在两层间执行混合操作，即便其中一层禁止也是如此。要避免层禁止时显示默认颜色，需要将 LTDC_LxBFCR 寄存器中此层的混合系数设置为其复位值。

9. 色健

色健（RGB）可配置为代表透明像素。

使能色健后，当前像素（格式转换后，混合前的像素）将与色健进行比较。如果当前像素与编程的 RGB 值相匹配，则该像素的所有通道 ARGB 均设置为 0。

运行时，可配置色健值并用其替换像素 RGB 值。

色健通过 LTDC_LxCKCR 寄存器配置。

40.5. LTDC 中断

LTDC 提供四个可屏蔽中断，这些中断经逻辑或运算后产生两个中断向量。

中断源可通过 LTDC_IER 寄存器单独使能或禁止。将相应的屏蔽位置 1 可使能相应中断。

发生如下事件时会产生两个中断：

- 行中断：达到编程的行时产生。行中断的位置在 LTDC_LIPCR 寄存器中编程。
- 寄存器重载中断：在垂直消隐周期内执行影子寄存器重载时产生。
- FIFO 下溢中断：从空层 FIFO 中请求像素时产生
- 传输错误中断：数据传输期间出现 AHB 总线错误时产生。

这些中断事件与 NVIC 控制器相连，如下图所示：

图 40-5 中断事件

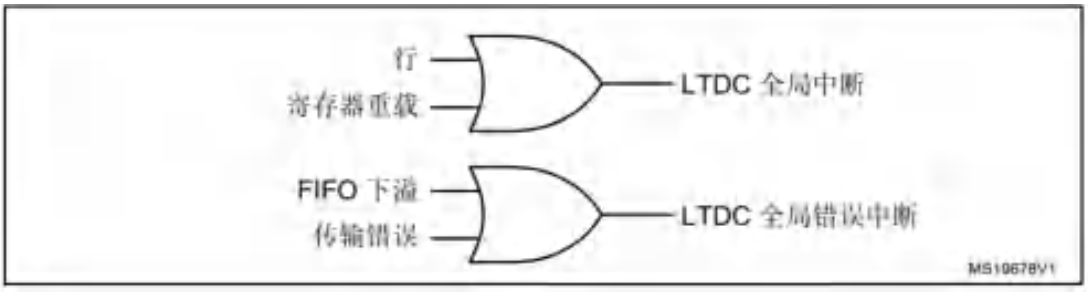


表 40-3 LTDC 中断请求

中断事件	事件标志	使能控制位
行	LIF	LIE
寄存器重载	RRIF	RRIEN
FIFO 下溢	FUIF	FUIE
传输错误	TERRIF	TERRIE

40.6. 配置流程

- 在 RCC 寄存器中使能 LTDC 时钟
- 按照面板数据表配置所需像素时钟
- 按照面板数据表配置同步时序 VSYNC 、HSYNC、垂直和水平后沿、有效数据区域以及 前沿时序。
- 配置 LTDC_GCR 寄存器中的同步信号和时钟极性
- 必要时，配置 LTDC_BCCR 寄存器中的背景色
- 配置 LTDC_IER 和 LTDC_LIPCR 寄存器中的所需中断。

- 通过执行以下编程操作配置 1/2 层的参数
  - 编程 LTDC_LxWHPCR 和 LTDC_LxWVPCR 寄存器中的层窗口的水平和垂直位置。层窗口必须位于有效数据区域。
  - 编程 LTDC_LxPFCR 寄存器中的像素输入格式
  - 编程 LTDC_LxCFBAR 寄存器中的颜色帧起始地址
  - 编程 LTDC_LxCFBLR 寄存器中的颜色帧缓冲区的行长和间距
  - 编程 LTDC_LxCFBLNR 寄存器中的颜色帧缓冲区的行数
  - 必要时，在 LTDC_LxCLUTWR 寄存器中为 CLUT 加载 RGB 值及其地址
  - 必要时，分别在 LTDC_LxDCCR 和 LTDC_LxBFCR 寄存器中配置默认颜色和混合系数。
- 使能 LTDC_LxCR 寄存器中的第 1/2 层，必要时使能 CLUT。
- 必要时，可分别在 LTDC_GCR 和 LTDC_LxCKCR 寄存器中使能抖动和色键。也可以实时使能这两个功能。
- 通过 LTDC_SRCR 寄存器将影子寄存器重载到活动寄存器中。
- 使能 LTDC_GCR 寄存器中的 LCD-TFT 控制器。
- 除 CLUT 外，所有层参数均可实时修改。新配置必须通过配置 LTDC_SRCR 寄存器立即重载或在垂直消隐周期内重载。

注意：所有层的寄存器均执行影子操作。一旦对某个寄存器执行写操作，便不应在重载完成前再次进行修改。因此，如果在尚未重载时对同一寄存器执行新的写操作，则将覆盖之前的配置。

40.7. LTDC 寄存器描述

40.7.1. 寄存器列表

LTDC 寄存器基地址：0x40016800

偏移	名称	复位值	描述
0x08	LTDC_SSCR	0x00000000	同步大小配置寄存器
0x0C	LTDC_BPCR	0x00000000	后沿配置寄存器
0x10	LTDC_AWCR	0x00000000	有效宽度配置寄存器
0x14	LTDC_TWCR	0x00000000	总宽度配置寄存器
0x18	LTDC_GCR	0x00000000	全局控制寄存器
0x24	LTDC_SRCR	0x00000000	影子重载配置寄存器
0x2C	LTDC_BCCR	0x00000000	背景色配置寄存器
0x34	LTDC_IER	0x00000000	中断使能寄存器
0x38	LTDC_ISR	0x00000000	中断状态寄存器
0x3C	LTDC_ICR	0x00000000	中断清零寄存器
0x40	LTDC_LIPCR	0x00000000	行中断位置配置寄存器
0x44	LTDC_CPSR	0x00000000	当前位置状态寄存器
0x48	LTDC_CDSR	0x00000000	当前显示状态寄存器
0x84	LTDC_L1CR	0x00000000	第 1 层控制寄存器

0x88	LTDC_L1WHPCR	0x00000000	第 1 层窗口水平位置配置寄存器
0x8C	LTDC_L1WVPCR	0x00000000	第 1 层窗口垂直位置配置寄存器
0x90	LTDC_L1CKCR	0x00000000	第 1 层色键配置寄存器
0x94	LTDC_L1PFCR	0x00000000	第 1 层像素格式配置寄存器
0x98	LTDC_L1CACR	0x000000ff	第 1 层常数 Alpha 配置寄存器
0x9C	LTDC_L1DCCR	0x00000000	第 1 层默认颜色配置寄存器
0xA0	LTDC_L1BFCR	0x00000607	第 1 层混合系数配置寄存器
0xAC	LTDC_L1CFBAR	0x00000000	第 1 层颜色帧缓冲区地址寄存器
0xB0	LTDC_L1CFBLR	0x00000000	第 1 层颜色帧缓冲区长度寄存器
0xB4	LTDC_L1CFBLNR	0x00000000	第 1 层颜色帧缓冲区行数寄存器
0xC4	LTDC_L1CLUTWR	0x00000000	第 1 层 CLUT 写寄存器
0x104	LTDC_L2CR	0x00000000	第 2 层控制寄存器
0x108	LTDC_L2WHPCR	0x00000000	第 2 层窗口水平位置配置寄存器
0x10C	LTDC_L2WVPCR	0x00000000	第 2 层窗口垂直位置配置寄存器
0x110	LTDC_L2CKCR	0x00000000	第 2 层色键配置寄存器
0x114	LTDC_L2PFCR	0x00000000	第 2 层像素格式配置寄存器
0x118	LTDC_L2CACR	0x00000000	第 2 层常数 Alpha 配置寄存器
0x11C	LTDC_L2DCCR	0x00000000	第 2 层默认颜色配置寄存器
0x120	LTDC_L2BFCR	0x00000000	第 2 层混合系数配置寄存器
0x12C	LTDC_L2CFBAR	0x00000000	第 2 层颜色帧缓冲区地址寄存器
0x130	LTDC_L2CFBLR	0x00000000	第 2 层颜色帧缓冲区长度寄存器
0x134	LTDC_L2CFBLNR	0x00000000	第 2 层颜色帧缓冲区行数寄存器
0x144	LTDC_L2CLUTWR	0x00000000	第 2 层 CLUT 写寄存器

40.7.2. 同步大小配置寄存器(LTDC_SSCR: 08h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	HSW	RW	0	水平同步宽度（以像素时钟周期为单位） 这些位定义水平同步像素减 1。
15:11	RSV	-	-	保留
10:0	VSH	RW	0	垂直同步高度（以水平扫描行为单位） 这些位定义垂直同步高度减 1。它代表水平同步行的数量。

40.7.3. 后沿配置寄存器(LTDC_BPCR: 0Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	AHBP	RW	0	累加水平后沿（以像素时钟周期为单位） 这些位定义累加水平后沿宽度。 水平后沿是水平同步信号变为无效到下一扫描行的有效显示开始之间的间隔。
15:11	RSV	-	-	保留
10:0	AVBP	RW	0	累加垂直后沿（以水平扫描行为单位） 这些位定义累加垂直后沿宽度。 垂直后沿是帧开始到下一帧的首个有效扫描行开始所包含的水平扫描行的数量。

40.7.4. 有效宽度配置寄存器(LTDC_AWCR: 10h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	AAW	RW	0	累加有效宽度（以像素时钟周期为单位） 这些位定义累加有效宽度。 有效宽度是面板扫描行的有效显示区中的像素。支持最大的有效宽度是 0x400。
15:11	RSV	-	-	保留
10:0	AAH	RW	0	累加有效高度（以水平扫描行为单位） 这些位定义累加高度。 有效高度是面板中的有效行数。支持最大的有效高度是 0x300。

40.7.5. 总宽度配置寄存器(LTDC_TWCR: 14h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	TOTALW	RW	0	总宽度（以像素时钟周期为单位） 这些位定义累加总宽度。

15:11	RSV	-	-	保留
10:0	TOTALH	RW	0	总高度（以水平扫描行为单位） 这些位定义累加高度。

40.7.6. 总宽度配置寄存器(LTDC_GCR: 18h)

位域	名称	属性	复位值	描述
31	HSPOL	RW	0	水平同步极性 0：水平同步极性低电平有效 1：水平同步极性高电平有效
30	VSPOL	RW	0	垂直同步极性 0：垂直同步低电平有效 1：垂直同步高电平有效
29	DEPOL	RW	0	数据使能极性 0：数据使能极性低电平有效 1：数据使能极性高电平有效
28	PCPOL	RW	0	像素时钟极性 0：输入像素时钟 1：反相输入像素时钟
27:18	RSV	-	-	保留
17	AHB_LOCK	RW	0	AHB 总线锁定
16	DEN	RW	0	抖动使能 0：禁止抖动 1：使能抖动
15	RSV	-	-	保留
14:12	DRW	RW	0	抖动红色宽度
11	RSV	-	-	保留
10:8	DGW	RW	0	抖动绿色宽度
7	RSV	RW	0	保留
6:4	DBW	RW	0	抖动蓝色宽度
3:1	RSV	-	-	保留

0	LTDCEN	RW	0	LCD-TFT 控制器使能位 0：禁止 LTDC 1：使能 LTDC

40.7.7. 影子重载配置寄存器(LTDC_SRCR: 24h)

位域	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	VBR	RW	0	垂直消隐重载 此位由软件置 1，只有重载后硬件清零。 0：无影响 1：影子寄存器在垂直消隐周期内重载
0	IMR	RW	0	立即重载 0：无影响 1：影子寄存器立即重载

40.7.8. 背景色配置寄存器(LTDC_BCCR: 2Ch)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:16	BCRED	RW	0	背景红色值
15:8	BCGREEN	RW	0	背景绿色值
7:0	BCBLUE	RW	0	背景蓝色值

40.7.9. 中断使能寄存器(LTDC_IER: 34h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	RRIE	RW	0	寄存器重载中断使能
2	TERRIE	RW	0	传输错误中断使能
1	FUIE	RW	0	FIFO 下溢中断使能
0	LIE	RW	0	行中断使能

40.7.10. 中断状态寄存器(LTDC_ISR: 38h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	RRIF	RO	0	寄存器重载中断标志
2	TERRIF	RO	0	传输错误中断标志
1	FUIF	RO	0	FIFO 下溢中断标志
0	LIF	RO	0	行中断标志

40.7.11. 中断清零寄存器(LTDC_ICR: 3Ch)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	CRRIF	WO	0	寄存器重载中断清零标志
2	CTERRIF	WO	0	传输错误中断清零标志
1	CFUIF	WO	0	FIFO 下溢中断清零标志
0	CLIF	WO	0	行中断清零标志

40.7.12. 行中断位置配置寄存器(LTDC_LIPCR: 40h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
10:0	LIPOS	RW	0	行中断位置

40.7.13. 当前位置状态寄存器(LTDC_CPSR: 44h)

位域	名称	属性	复位值	描述
31:16	CXPOS	RO	0	当前 X 位置
15:0	CYPOS	RO	0	当前 Y 位置

40.7.14. 当前显示状态寄存器(LTDC_CDSR: 48h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	HSYNCS	RO	0	水平同步显示状态 0：低电平有效 1：高电平有效
2	VSYNCS	RO	0	垂直同步显示状态 0：低电平有效 1：高电平有效
1	HDES	RO	0	水平数据使能显示状态 0：低电平有效 1：高电平有效
0	VDES	RO	0	垂直数据使能显示状态 0：低电平有效 1：高电平有效

40.7.15. 第 1 层控制寄存器(LTDC_L1CR: 84h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	CLUTEN	RW	0	查色表使能 0：禁止查色表 1：使能查色表
3:2	RSV	-	-	保留
1	COLKEN	RW	0	色键使能 0：禁止色键 1：使能色键
0	LEN	RW	0	层使能 0：禁止层 1：使能层

40.7.16. 第 1 层水平位置配置寄存器(LTDC_L1WHPCR: 88h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	WHSPPOS	RW	0	窗口水平结束位置 定义窗口的一行的最后一个可见像素
15:12	RSV	-	-	保留
11:0	WHSTPOS	RW	0	窗口水平起始位置 定义窗口的一行的第一个可见像素

40.7.17. 第 1 层垂直位置配置寄存器(LTDC_L1WVPCR: 8Ch)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:16	WVSPPOS	RW	0	窗口垂直结束位置 定义窗口最后一个可见行
15:11	RSV	-	-	保留
10:0	WVSTPOS	RW	0	窗口垂直起始位置 定义窗口的第一个可见行

40.7.18. 第 1 层色键配置寄存器(LTDC_L1CKCR: 90h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:16	CKRED	RW	0	色键红色值
15:8	CKGREEN	RW	0	色键绿色值
7:0	CKBLUE	RW	0	色键蓝色值

40.7.19. 第 1 层像素格式配置寄存器(LTDC_L1PFCR: 94h)

位域	名称	属性	复位值	描述

31:3	RSV	-	-	保留
2:0	PF	RW	0	像素格式 000：ARGB8888 001：RGB888 010：RGB565 011：ARGB1555 100：ARGB4444 101：L8 110：AL44 111：AL88

40.7.20. 第 1 层常数 Alpha 配置寄存器(LTDC_L1CACR: 98h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CONSTA	RW	0xFF	常数 Alpha 这些位配置混合时使用的常数 Alpha。常数 Alpha 由硬件实现 255 等分。 示例：如果编程的常数 Alpha 为 0xFF，则常数 Alpha 值为 255/255=1。

40.7.21. 第 1 层默认颜色配置寄存器(LTDC_L1DCCR: 9Ch)

位域	名称	属性	复位值	描述
31:24	DCALPHA	RW	0	默认 Alpha 值
23:16	DCRED	RW	0	默认颜色红色
15:8	DCGREEN	RW	0	默认颜色绿色
7:0	CONSTA	RW	0	默认颜色蓝色

40.7.22. 第 1 层混合系数配置寄存器(LTDC_L1BFCR: A0h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留

10:8	BF1	RW	0x6	混合系数 1 100：常数 Alpha 110：像素 Alpha x 常数 Alpha 其他：保留 混合公式：BC = BF1xC + BF2xCs BC=混合后的颜色 BF1=混合系数 1 C = 当前层颜色 BF2 = 混合系数 2 Cs = 底层混合后的颜色 示例：仅使能第一层，BF1 配置为常数 Alpha BF2 配置为 1-常数 Alpha 常数 Alpha：再 LxCACR 寄存器中编程的常数 Alpha 为 240，因此 Alpha 的值等于 240/255=0.94 C：当前层颜色为 128 Cs：背景色为 48 第一层和背景色混合。 BC=0.94x128 +(1-0.94)x48 = 123。
7:3	RSV	-	-	保留
2:0	BF2	RW	0x7	混合系数 2 101：1 - 常数 Alpha 111：1-（像素 Alpha x 常数 Alpha）

40.7.23. 第 1 层颜色帧缓冲区地址寄存器(LTDC_L1CFBAR: ACh)

位域	名称	属性	复位值	描述
31:0	CFBADD	RW	0	颜色帧缓冲区起始地址

40.7.24. 第 1 层颜色帧缓冲区长度寄存器(LTDC_L1CFBLR: B0h)

位域	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:16	CFBP	RW	0	颜色帧缓冲区间隔 定义从像素某行的起始处到下一行的起始处的增量。
15:13	RSV	-	-	保留

12:0	CFBLL	RW	0	颜色帧缓冲区行长 这些位定义一行像素的长度+3

40.7.25. 第 1 层颜色帧缓冲区行数寄存器(LTDC_L1CFBLNR: B4h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:0	CFBLNBR	RW	0	帧缓冲区行数 定义缓冲区中的行数，行数跟有效高度对应。

40.7.26. 第 1 层 CLUT 写寄存器(LTDC_L1CLUTWR : C4h)

位域	名称	属性	复位值	描述
31:24	CLUTADD	WO	0	CLUT 地址 配置每个 RGB 值得 CLUT 地址
23:16	RED	WO	0	红色值
15:8	GREEN	WO	0	绿色值
7:0	BLUE	WO	0	蓝色值

40.7.27. 第 2 层控制寄存器(LTDC_L2CR: 104h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	CLUTEN	RW	0	查色表使能 0：禁止查色表 1：使能查色表
3:2	RSV	-	-	保留
1	COLKEN	RW	0	色键使能 0：禁止色键 1：使能色键
0	LEN	RW	0	层使能 0：禁止层 1：使能层

40.7.28. 第 2 层水平位置配置寄存器(LTDC_L2WHPCR: 108h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	WHSPPOS	RW	0	窗口水平结束位置 定义窗口的一行的最后一个可见像素
15:12	RSV	-	-	保留
11:0	WHSTPOS	RW	0	窗口水平起始位置 定义窗口的一行的第一个可见像素

40.7.29. 第 2 层垂直位置配置寄存器(LTDC_L2WVPCR: 10Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	WVSPPOS	RW	0	窗口垂直结束位置 定义窗口最后一个可见行
15:12	RSV	-	-	保留
11:0	WVSTPOS	RW	0	窗口垂直起始位置 定义窗口的第一个可见行

40.7.30. 第 2 层色键配置寄存器(LTDC_L2CKCR: 110h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:16	CKRED	RW	0	色键红色值
15:8	CKGREEN	RW	0	色键绿色值
7:0	CKBLUE	RW	0	色键蓝色值

40.7.31. 第 2 层像素格式配置寄存器(LTDC_L2PFCR: 114h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	PF	RW	0	像素格式 000：ARGB8888 001：RGB888 010：RGB565 011：ARGB1555 100：ARGB4444 101：L8 110：AL44 111：AL88

40.7.32. 第 2 层常数 Alpha 配置寄存器(LTDC_L2CACR: 118h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CONSTA	RW	0	常数 Alpha 这些位配置混合时使用的常数 Alpha。常数 Alpha 由硬件实现 255 等分。 示例：如果编程的常数 Alpha 为 0xFF，则常数 Alpha 值为 255/255=1。

40.7.33. 第 2 层默认颜色配置寄存器(LTDC_L2DCCR: 11Ch)

位域	名称	属性	复位值	描述
31:24	DCALPHA	RW	0	默认 Alpha 值
23:16	DCRED	RW	0	默认颜色红色
15:8	DCGREEN	RW	0	默认颜色绿色
7:0	CONSTA	RW	0	默认颜色蓝色

40.7.34. 第 2 层混合系数配置寄存器(LTDC_L2BFCR: 120h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留

10:8	BF1	RW	0	混合系数 1 100：常数 Alpha 110：像素 Alpha x 常数 Alpha 其他：保留 混合公式：BC = BF1xC + BF2xCs BC=混合后的颜色 BF1=混合系数 1 C = 当前层颜色 BF2 = 混合系数 2 Cs = 底层混合后的颜色 示例：仅使能第一层，BF1 配置为常数 Alpha BF2 配置为 1-常数 Alpha 常数 Alpha：再 LxCACR 寄存器中编程的常数 Alpha 为 240，因此 Alpha 的值等于 240/255=0.94 C：当前层颜色为 128 Cs：背景色为 48 第一层和背景色混合。 BC=0.94x128 +(1-0.94)x48 = 123。
7:3	RSV	-	-	保留
2:0	BF2	RW	0	混合系数 2 101：1 - 常数 Alpha 111：1-（像素 Alpha x 常数 Alpha）

40.7.35. 第 2 层颜色帧缓冲区地址寄存器(LTDC_L2CFBAR: 12Ch)

位域	名称	属性	复位值	描述
31:0	CFBADD	RW	0	颜色帧缓冲区起始地址

40.7.36. 第 2 层颜色帧缓冲区长度寄存器(LTDC_L2CFBLR: 130h)

位域	名称	属性	复位值	描述
31:29	RSV	-	-	保留
31:0	CFBADD	RW	0	颜色帧缓冲区间隔 定义从像素某行的起始处到下一行的起始处的增量。
15:13	RSV	-	-	保留

12:0	CFBLL	RW	0	颜色帧缓冲区行长 这些位定义一行像素的长度+3

40.7.37. 第 2 层颜色帧缓冲区行数寄存器(LTDC_L2CFBLNR: 134h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:0	CFBLNBR	RW	0	帧缓冲区行数 定义缓冲区中的行数，行数跟有效高度对应。

40.7.38. 第 2 层 CLUT 写寄存器(LTDC_L2CLUTWR : 144h)

位域	名称	属性	复位值	描述
31:24	CLUTADD	WO	0	CLUT 地址 配置每个 RGB 值得 CLUT 地址
23:16	RED	WO	0	红色值
15:8	GREEN	WO	0	绿色值
7:0	BLUE	WO	0	蓝色值

41. 图像加速器（DMA2D）

41.1. 概述

DMA2D 是专用于图像处理的专业 DMA。它可以执行下列操作：

用特定颜色填充目标图像的一部分或全部。

将源图像的一部分（或全部） 复制到目标图像的一部分（或全部）中

通过像素格式转换将源图像的一部分或全部复制到目标图像的一部分或者全部

将像素格式不同的两个源图像部分和/或全部混合，再将结果复制到颜色不同的部分或整个目标图像中。

在索引颜色模式或直接颜色模型下，支持所有经典颜色编码方案，并支持每像素 4 位等最高 32 位。DMA2D 自生具有专门的 CLUT 存储器。

41.2. 主要特性

- 采用单 AHB 主设备总线架构
- AHB 从设备编程接口支持 8/16/32 位访问（32 位的 CLUT 访问除外）
- 用户可编程工作区大小
- 用户可编程源区域和目标区域的偏移
- 用户可编程整个存储空间的源地址和目标地址
- 最多支持 2 个源的混合操作
- Alpha 值可修改（源值、固定值或调制的值）
- 用户可编程源颜色格式和目标的颜色格式
- 采用间接或直接颜色编码时，支持多达 11 种颜色格式，且支持每像素 4 位到最高 32 位
- 间接颜色模式下使用 2 个内部存储器存储 CLUT
- 通过 CPU 自动加载 CLUT 或对 CLUT 进行编程
- 用户可编程 CLUT 大小
- 使用内部定时器控制 AHB 带宽
- 支持 4 种工作模式：存储器到存储器、寄存器到存储器、存储器到存储器且支持像素格式转换、存储器到存储器且支持像素格式转换和混合。
- 可使用固定颜色进行区域填充
- 可从一个区域复制到另一个区域
- 在源图像和目标图像之间进行复制时进行像素格式转换
- 支持从颜色格式不同的两幅源图像复制并混合
- 可中止并挂起 DMA2D 操作
- 支持在传输用户可编程的目标行时生成水印中断
- 支持发生总线错误或访问冲突时生成中断
- 支持处理完成时生成中断

41.3. 功能描述

41.3.1. 功能概述

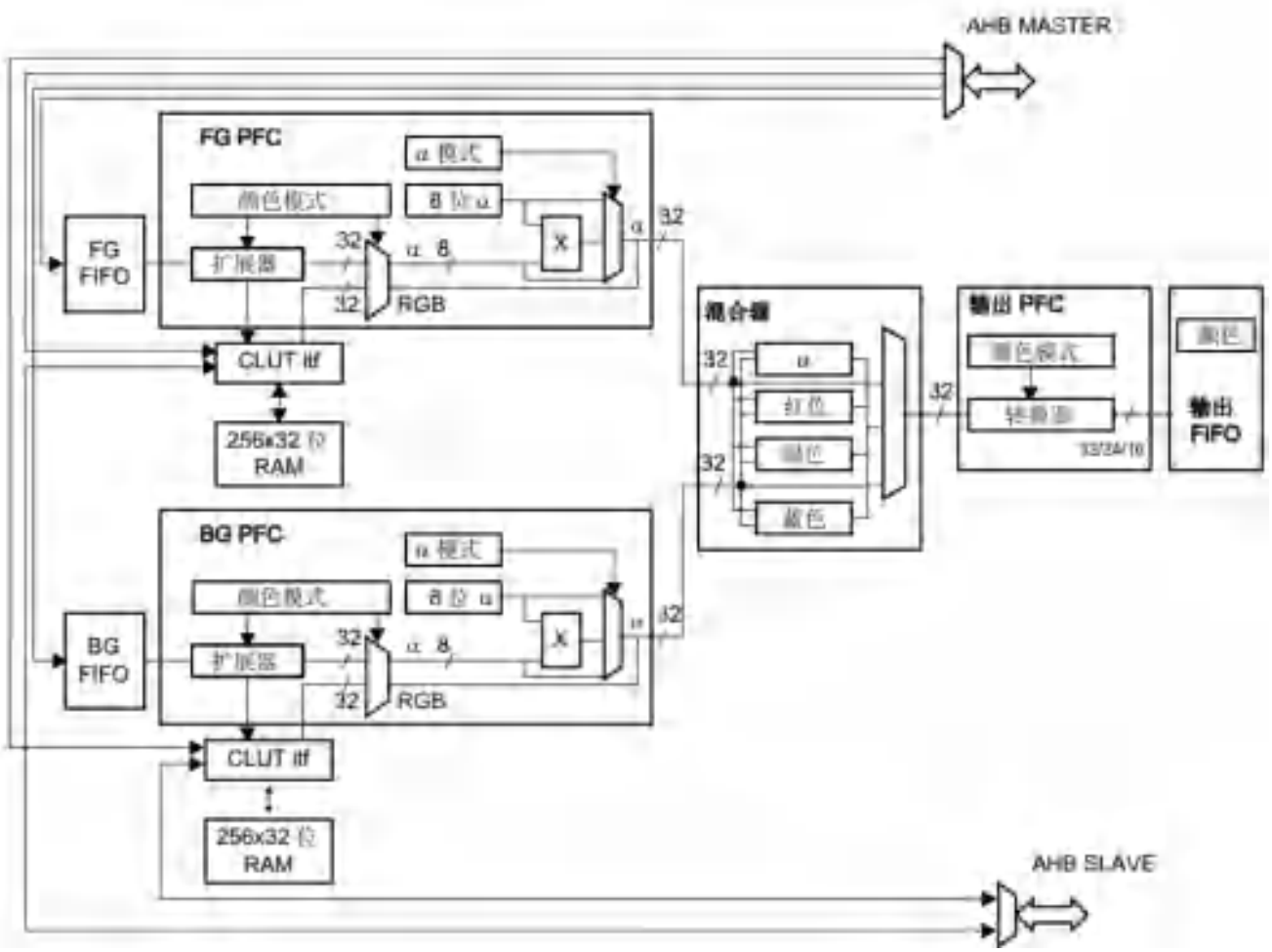
DMA2D 控制器执行直接存储器传输。作为一个 AHB 主设备，它可以控制 AHB 总线矩阵来启动 AHB 事务。

DMA2D 可以在以下四种模式下工作：

- 寄存器到存储器
- 存储器到存储器
- 存储器到存储器并执行像素格式转换
- 存储器到存储器并执行像素格式转换和混合。

AHB 从设备端口用于编程 DMA2D 控制器。

41.3.2. 结构框图



41.3.3. DMA2D 控制

通过 DMA2D 控制寄存器（DMA2D_CR）配置 DMA2D 控制器，允许选择：

用户应用可以执行下列操作：

- 选择工作模式

- 使能/禁止 DMA2D 中断
- 启动/挂起/中止进行中的数据传输

41.3.4. DMA2D 前景层 FIFO 和背景层 FIFO

DMA2D 前景层（FG）FIFO 和背景层（BG）FIFO 获取要复制和/或处理的输入数据。

这些 FIFO 根据相应像素格式转换器 PFC 中定义的颜色格式获取像素。

通过如下一组寄存器对它们进行编程：

- DMA2D 前景层存储器地址寄存器（DMA2D_FGMAR）
- DMA2D 前景层偏移寄存器（DMA2D_FGOR）
- DMA2D 背景层存储地址寄存器（DMA2D_BGMAR）
- DMA2D 背景层偏移寄存器（DMA2D_BGBOR）
- DMA2D 行数寄存器(行数和每行像素数) (DMA2D_NLR)

DMA2D 在寄存器到存储器模式下工作时，不激活任何 FIFO。

DMA2D 在存储器到存储器模式下工作时（无像素格式转换和混合操作），仅激活 FG FIFO，并将其用作缓冲区。

DMA2D 在存储器到存储器模式下工作时并支持像素格式转换时（无混合操作），不会激活 BG FIFO。

41.3.5. DMA2D 前景层和背景层像素格式转换器（PFC）

DMA2D 前景层和背景层像素格式转换器（PFC）执行像素格式转换。以生成每像素 32 位的值。PFC 还能够修改 alpha 通道。

转换器在第一阶段转换颜色格式。前景层像素和背景层像素的原始颜色格式分别通过 DMA2D_FGPFCCR 和 DMA2D_BGFPCCR 的 CM[3:0]位来配置。

表 41-1 输入时支持的颜色模式

CM[3:0]	颜色模式
0000	ARGB8888
0001	RGB888
0010	RGB565
0011	ARGB1555
0100	ARGB4444
0101	L8
0110	AL44
0111	AL88
1000	L4

1001	A8
1010	A4

颜色格式的编码方式如下：

- Alpha 值字段：透明
  - 0xFF 值对应不透明像素，0x00 对应透明像素
- R 字段代表红色
- G 字段代表绿色
- B 字段代表蓝色
- L 字段：亮度
  - 该字段是 CLUT 的索引，用于检索三个/四个 RGB/ARGB 分量。

如果原始格式为直接颜色模式，则通过将 MSB 复制到 LSB 扩展为每通道 8 位。这可以确保转换具有良好的线性。

如果原始格式不包括 Alpha 通道，则将自动将 Alpha 值设为 0xFF。

如果原始格式为间接颜色模式，则需要使用 CLUT，并且每个像素格式转换器与一个 256 个 32 位条目的 CLUT 相关联。

对于特定的 Alpha 模式 A4 和 A8，既不存储颜色信息，也不编制索引。用于生成图像的颜色是秃顶的，并且在 DMA2D_FGCOLR 寄存器中定义前景层像素的颜色，在 DMA2D_BGCOLR 寄存器中定义背景层像素的颜色。

表 41-2 存储器中的数据顺序

颜色模式	@ + 3	@ + 2	@ + 1	@ + 0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]
RGB565	R_1[4:0]G_1[5:3]	G_1[2:0]B_1[4:0]	R_0[4:0]G_0[5:3]	G_0[2:0]B_0[4:0]
ARGB1555	A_1[0]R_1[4:0]G_1[4:3]	G_1[2:0]B_1[4:0]	A_0[0]R_0[4:0]G_0[4:3]	G_0[2:0]B_0[4:0]
ARGB4444	A_1[3:0]R_1[3:0]	G_1[3:0]B_1[3:0]	A_0[3:0]R_0[3:0]	G_0[3:0]B_0[3:0]
L8	L_3[7:0]	L_2[7:0]	L_1[7:0]	L_0[7:0]
AL44	A_3[3:0]L_3[3:0]	A_2[3:0]L_2[3:0]	A_1[3:0]L_1[3:0]	A_0[3:0]L_0[3:0]
AL88	A_1[7:0]	L_1[7:0]	A_0[7:0]	L_0[7:0]
L4	L_7[3:0]L_6[3:0]	L_5[3:0]L_4[3:0]	L_3[3:0]L_2[3:0]	L_1[3:0]L_0[3:0]
A8	A_3[7:0]	A_2[7:0]	A_1[7:0]	A_0[7:0]
A4	A_7[3:0]A_6[3:0]	A_5[3:0]A_4[3:0]	A_3[3:0]A_2[3:0]	A_1[3:0]A_0[3:0]

通过 ARGB8888 模式支持按 32 位对齐 24 位 RGB888。

生成 32 位值后，即可根据 DMA2D_FGPFCCR/DMA2D_BGFPCCR 寄存器的 AM[1:0]字段修改 Alpha 通道。

表 41-3 Alpha 模式配置

AM[1:0]	Alpha 模式
00	不做修改
01	替换为 DMA2D_xxPFCCR 中的值
10	替换为原始值与 DMA2D_xxPFCCR 中的值/255 所得商的乘积
11	保留

41.3.6. DMA2D 前景层 FIFO 和背景层 CLUT 接口

CLUT 接口可管理对 CLUT 存储器的访问以及 CLUT 的自动加载。

支持如下三种访问：

- PFC 在像素格式转换期间读取 CLUT。
- CPU 对 CLUT 进行数据的读取或写入时，通过 AHB 从设备端口访问 CLUT。
- 执行自动加载 CLUT 时，通过 AHB 主设备端口进行 CLUT 写入。

可以通过两种不同方法执行 CLUT 存储器加载：

- 自动加载
  - 将 CLUT 地址编程到 DMA2D_FGCMAR 寄存器或 DMA2D_BGCMAR 寄存器。
  - 将 CLUT 大小编程到 DMA2D_FGPFCCR 寄存器或 DMA2D_BGPFCCR 寄存器。
  - 将 DMA2D_FGPFCCR 寄存器或 DMA2D_BGPFCCR 寄存器的 START 位置 1 以启动传输。自动加载过程期间，不可通过 CPU 访问 CLUT。如果出现冲突，弱 DMA2D_CR 中的 CAEIE 被置 1，则发生 CLUT 访问错误中断。
- 手动加载

应用程序必须通过 DMA2D AHB 从设备端口手动编程本地 CLUT 存储器映射到 CLUT。前景色 CLUT 从偏移地址 0x400 开始，背景层 CLUT 从偏移地址 0x800 开始。

CLUT 格式可以是 24 位或 32 位。通过 DMA2D_FGPFCCR 寄存器或 DMA2D_BGPFCCR 寄存器的 CCM 位配置格式。

表 41-4 支持的 CLUT 颜色模式

CCM	CLUT 颜色模式
0	32 位 ARGB8888
1	24 位 RGB888

表 41-5 存储器中的 CLUT 数据顺序

CLUT 颜色模式	@+3	@+2	@+1	@+0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]

41.3.7. DMA2D 混合器

DMA2D 混合器成对混合源像素以计算结果像素。

混合将按以下公式执行：

$$其中\ \alpha_{Mult} = \frac{\alpha_{FG} \cdot \alpha_{BG}}{255}$$

$$\alpha_{OUT} = \alpha_{FG} + \alpha_{BG} - \alpha_{Mult}$$

$$C_{OUT} = \frac{C_{FG} \cdot \alpha_{FG} + C_{BG} \cdot \alpha_{BG} - C_{BG} \cdot \alpha_{Mult}}{\alpha_{OUT}} \quad 其中\ C = R\ 或\ G\ 或\ B$$

商将向下取整

混合器不需要任何配置寄存器。是否使用混合器取决于 DMA2D_CR 寄存器的 MODE[1:0]字段中定义的 DMA2D 工作模式。

41.3.8. DMA2D 输出 PFC

输出 PFC 将像素格式从 32 位转换为指定的输出格式，输出格式在 DMA2D 输出像素格式转换器配置寄存器（DMA2D_OPFCCR）的 CM[2:0]字段中定义。

表 41-6 输出时支持的颜色模式

CM[2:0]	颜色模式
000	ARGB8888
001	RGB888
010	RGB565
011	ARGB1555
100	ARGB4444

41.3.9. DMA2D 输出 FIFO

输出 FIFO 根据输出 PFC 中定义的颜色格式对像素进行编程。

通过如下一组寄存器定义目标区域：

- DMA2D 输出存储器地址寄存器（DMA2D_OMAR）
- DMA2D 输出偏移寄存器（DMA2D_OOR）
- DMA2D 行数寄存器（行数和每行像素数）（DMA2D_NLR）

如果 DMA2D 在寄存器到存储器模式下工作，则配置的输出矩阵将以 DMA2D 输出颜色寄存器（DMA2D_OCOLR）中指定的颜色填充，该寄存器中包含固定的 32 位、24 位或 16 位值。

通过 DMA2D_OPFCCR 寄存器的 CM[2:0]字段选择格式。

表 41-7 存储器中的数据顺序

颜色模式	@+3	@+2	@+1	@+0
ARGB8888	A_0[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
RGB888	B_1[7:0]	R_0[7:0]	G_0[7:0]	B_0[7:0]
	G_2[7:0]	B_2[7:0]	R_1[7:0]	G_1[7:0]
	R_3[7:0]	G_3[7:0]	B_3[7:0]	R_2[7:0]
RGB565	R_1[4:0]G_1[5:3]	G_1[2:0]B_1[4:0]	R_0[4:0]G_0[5:3]	G_0[2:0]B_0[4:0]
ARGB1555	A_1[0]R_1[4:0]G_1[4:3]	G_1[2:0]B_1[4:0]	A_0[0]R_0[4:0]G_0[4:3]	G_0[2:0]B_0[4:0]
ARGB4444	A_1[3:0]R_1[3:0]	G_1[3:0]B_1[3:0]	A_0[3:0]R_0[3:0]	G_0[3:0]B_0[3:0]

通过 ARGB8888 模式支持 32 位对齐 RGB888。

41.3.10. DMA2D AHB 主设备端口定时器

AHB 主设备端口内嵌一个 8 位定时器，以便可选择限制交叉开关矩阵的带宽。此定时器由 AHB 时钟驱动，对两个连续访问之间的死区进行计数。这样可以限制带宽的使用。

通过 AHB 主设备端口定时器配置寄存器（DMA2D_AMPTCR）配置定时器使能和死区值。

41.3.11. DMA2D 事务

DMA2D 事务由给定数目的数据传输序列组成。可通过软件对数据数目和宽度进行编程。

每个 DMA2D 数据传输最多需要 4 个步骤：

1）从 DMA2D_FGMAR 寄存器寻址的存储器单元加载数据并按照 DMA2D_FGCR 中的定义进行像素格式转换。

2）从 DMA2D_BGMAR 寄存器寻址的存储单元加载数据并按照 DMA2D_BGCR 中的定义进行像素格式转换。

3）根据对 alpha 值进行 PFC 操作所得到的 alpha 通道，将所有检索到的像素混合。

4）根据 DMA2D_OCR 寄存器对合成像素进行像素格式转换，然后将数据编程到通过 DMA2D_OMAR 寄存器寻址的存储单元。

41.3.12. DMA2D 配置

源和目标数据传输在整个 4GB 区域（地址范围在 0x0000_0000 和 0xFFFF_FFFF 之间） 都可以寻址外设和存储器。

DMA2D 可在以下四种模式下工作，通过 DMA2D_CR 寄存器的 MODE[1:0]位选择工作模式：

- 寄存器到存储器
- 存储器到存储器
- 存储器到存储器并执行 PFC
- 存储器到存储器并执行 PFC 和混合

41.3.12.1. 寄存器到存储器

寄存器到存储器模式用于以预定义颜色填充用户自定义区域。

颜色格式在 DMA2D_OPFCCR 中设置。

DMA2D 不从任何源获取数据。它只将 DMA2D_OCOLR 寄存器中定义的颜色写入通过 DMA2D_OMAR 寻址以及 DMA2D_NLR 和 DMA2D_OOR 定义的区域。

41.3.12.2. 存储器到存储器

在存储器到存储器模式下，DMA2D 不执行任何图形数据转换。前景层输入 FIFO 充当缓冲区，数据从 DMA2D_FGMAR 中定义的源存储单元传输到 DMA2D_OMAR 寻址的目标存储器单元。

DMA2D_FGPFCCR 寄存器的 CM[3:0]位中编程的颜色模式决定输入和输出的每像素位数。

对于要传输的区域大小，源区域大小由 DMA2D_NLR 和 DMA2D_FGOR 寄存器定义，目标区域大小则由 DMA2D_NLR 和 DMA2D_OOR 寄存器定义。

41.3.12.3. 存储器到存储器并执行 PFC

在模式下，DMA2D 对源数据执行像素格式转换并将结果存储在目标存储单元。

对于要传输的区域大小，源区域大小由 DMA2D_NLR 和 DMA2D_FGOR 寄存器定义，目标区域大小则由 DMA2D_NLR 和 DMA2D_OOR 寄存器定义。

从 DMA2D_FGMAR 寄存器定义的位置获取数据，并由前景层 PFC 进行处理。原始像素格式通过 DMA2D_FGPFCCR 寄存器配置。

如果原始像素格式是直接颜色模式，则所有颜色通道都扩展到 8 位。

如果像素格式是间接颜色模式，则必须将 CLUT 加载到 CLUT 存储器中。

CLUT 加载可按如下顺序自动完成：

1）在 DMA2D_FGCMAR 中设置 CLUT 地址。

2）在 DMA2D_FGPFCCR 寄存器的 CS[7:0]位设置 CLUT 大小。

3）在 DMA2D_FGPFCCR 寄存器的 CCM 位设置 CLUT 格式（24 或 32 位）

4）将 DMA2D_FGPFCCR 寄存器的 START 位置 1 启动 CLUT 加载。

CLUT 加载完成时，DMA2D_ISR 寄存器将置位 CTCIF 标志；如果 DMA2D_CR 中的 CTCIE 位置 1，还将产生中断。CLUT 自动加载过程无法与传统的 DMA2D 传输同时进行。

CLUT 还可由 CPU 填充，或由任意其他主设备通过 AHB 从设备填充。在 DMA2D 传输进行期间和使用 CLUT 时无法访问 CLUT。

在颜色转换执行期间，可根据 DMA2D_FGPFCCR 寄存器中编程的值添加或更改 Alpha 值。

如果原始图像没有 alpha 通道，则会自动添加一个默认的 alpha 值 0xFF 以获得完全不透明的像素。可根据 DMA2D_FGPFCCR 寄存器的 AM[1:0]位修改 alpha 值。

- 保持不变
- 替换为 DMA2D_FGPFCCR 寄存器的 ALPHA[7:0]值中定义的值。

- 替换为原始值与 DMA2D_FGPFCCR 寄存器的 ALPHA[7:0]值除以 255 所得商的乘积。

结果得到的 32 位数据由 OUTPFC 编码成 DMA2D_OPFCCR 存储器的 CM[2:0]字段所指定的格式。输出像素格式不可是间接模式，原因是输出时不支持 CLUT 生成过程。

数据经处理后，将写入 DMA2D__OMAR 寻址的目标存储单元。

41.3.12.4. 存储器到存储器并执行 PFC 和混合

在此模式下，将在前景层 FIFO 和背景层 IFFO 获取两个源图像。

必须按存储器到存储器模式中所述配置两个像素格式转换器。由于这两个像素格式转换器各自独立且自身具有 CLUT 存储器，因此其配置可以不同。

在两个像素都通过相应的 PFC 转换为 32 位后，将根据以下公式进行混合：

$$\text{其中 } \alpha_{Mult} = \frac{\alpha_{FG} \cdot \alpha_{BG}}{255}$$

$$\alpha_{OUT} = \alpha_{FG} + \alpha_{BG} - \alpha_{Mult}$$

$$C_{OUT} = \frac{C_{FG} \cdot \alpha_{FG} + C_{BG} \cdot \alpha_{BG} - C_{BG} \cdot \alpha_{Mult}}{\alpha_{OUT}} \quad \text{其中 C = R 或 G 或 B}$$

商将向下取整

输出 PFC 将根据指定的输出格式对得到的 32 位像素值进行编码，并且编码数据将写入 DMA2D_OMAR 寻址的目标存储单元。

41.3.12.5. 配置错误检测

DMA2D 将在每次执行传输前检查配置是否正确。开始新的传输/自动加载时，如果检测搭配配置错误，硬件将设置配置错误中断标志。如果 DMA2D_CR 寄存器的 CEIE 位置 1，还将产生中断。

可检测到的错误配置如下：

- 前景层 CLUT 自动加载：DMA2D_FGCMAR 的 MA 位与 DMA2D_FGPFCCR 的 CCM 位不匹配。
- 背景层 CLUT 自动加载：DMA2D_BGCMAR 的 MA 位与 DMA2D_BGPFCCR 的 CCM 位不匹配。
- 存储器传输（寄存器到存储器模式除外）：DMA2D_FGMAR 的 MA 位于 MDA2D_FGPFCCR 的 CM 位不匹配。
- 存储器传输（寄存器到存储器模式除外）：DMA2D_FGPFCCR 中的 CM 位无效。
- 存储器传输（寄存器到存储器模式除外）：DMA2D_NLR 的 PL 位为奇，而 DMA2D_FGPFCCR 的 CM 位为 A4 或 L4。
- 存储器传输（寄存器到存储器模式除外）：DMA2D_NLR 的 LO 位为奇，而 DMA2D_FGPFCCR 的 CM 位为 A4 或 L4。
- 存储器传输（仅限混合模式）：DMA2D_BGMAR 中的 MA 位与 DMA2D_BGPFCCR 的 CM 位不一致。
- 存储器传输：DMA2D_BGPFCCR 的 CM 无效（仅限混合模式）。

- 存储器传输（仅限混合模式）：DMA2D_NLR 的 PL 位为奇，而 DMA2D_BGPFCCR 的 CM 位为 A4 或 L4
- 存储器传输（寄存器到存储器模式除外）：DMA2D_OMAR 的 MA 位于 MDA2D_OPFCCR 的 CM 位不匹配。
- 存储器传输（寄存器到存储器模式除外）：DMA2D_OPFCCR 中的 CM 位无效。
- 存储器传输：DMA2D_NLR 中的 NL 位为 0。
- 存储器传输：DMA2D_NLR 中的 PL 位为 0。

41.3.12.6. DMA2D 传输控制（启动、挂起、中止和完成）

配置 DMA2D 后，通过将 DMA2D_CR 寄存器中的 START 位置 1 可启动传输。传输完成时，START 位自动复位且 DMA2D_ISR 寄存器置位 TCIF 标志。如果 DMA2D_CR 寄存器中的 TCIE 位置 1，还将产生中断。

用户应用程序随时可以通过 DMA2D_CR 寄存器的 SUSP 位置 1 来挂起 DMA2D。随即可通过将 DMA2D_CR 寄存器的 ABORT 位置 1 中止事务，或通过 DMA2D 寄存器的 SUSP 位复位重新启动事务。

用户应用程序随时都可通过将 DMA2D_CR 寄存器的 ABORT 位置 1 来中止处理中的事务。

这种情况下，不会置位 TCIF 标志。

还可以通过 DMA2D_FGPFCCR 和 DMA2D_BGPFCCR 寄存器自身的 START 位中止或挂起 CLUT 自动传输过程。

41.3.12.7. 水印

可对水印编程，以在指定行的最后一个像素写入目标存储区域时产生中断。

行号在 DMA2D_LWR 寄存器的 LW[15:0]字段中定义。

在该行的最后一个像素传输完成时，DMA2D_ISR 将置位 TWIF 标志；在 DMA2D_CR 的 TWIE 位置 1 的情况下，还将产生中断。

41.3.12.8. 错误管理

可触发两种错误：

- AHB 主设备端口错误，通过 DMA2D_ISR 寄存器的 TEIF 标志指示。
- CLUT 访问引发的冲突（CPU 在 CLUT 加载或 DMA2D 传输执行期间尝试访问 CLUT），通过 DMA2D_ISR 寄存器的 CAEIF 标志指示。

这两个标志都与其在 DMA2D_CR 寄存器中的中断使能标志相关。

41.3.12.9. AHB 死区

要限制 AHB 带宽的使用，可在两个连续的 AHB 访问之间编程一个死区。

可通过将 DMA2D_AMTCR 寄存器中的 EN 位置 1 使能此特性。

死区值存储在 DMA2D_AMTCR 寄存器中的 DT[7:0]字段中。该值表示 AHB 总线商两个连续事务之间允许占用的最少周期数。

在 DMA2D 运行过程中的对死区值所做的更新将在下一次 AHB 传输时生效。

41.4. DMA2D 中断

发生如下事件时可以生成中断：

- 配置错误
- CLUT 传输完成
- CLUT 访问错误
- 到达传输水印
- 传输完成
- 传输错误

表 41-8 DMA2D 中断请求

中断事件	事件标志	使能控制位
配置错误	CEIF	CEIE
CLUT 传输完成	CTCIF	CTCIE
CLUT 访问错误	CAEIF	CAEIE
传输水印	TWF	TWIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

41.5. DMA2D 寄存器描述

41.5.1. 寄存器列表

DMA2D 寄存器基地址：0x4002B000

偏移	名称	复位值	描述
0x00	DMA2D_CR	0x00000000	DMA2D 控制寄存器
0x04	DMA2D_ISR	0x00000000	DMA2D 中断状态寄存器
0x08	DMA2D_IFCR	0x00000000	DMA2D 中断标志清零寄存器
0x0C	DMA2D_FGMAR	0x00000000	DMA2D 前景层存储器地址寄存器
0x10	DMA2D_FGOR	0x00000000	DMA2D 前景层偏移寄存器
0x14	DMA2D_BGMAR	0x00000000	DMA2D 背景层存储器地址寄存器
0x18	DMA2D_BGOR	0x00000000	DMA2D 背景层偏移寄存器
0x1C	DMA2D_FGPFCCR	0x00000000	DMA2D 前景层 PFC 控制寄存器
0x20	DMA2D_FGCOLR	0x00000000	DMA2D 前景层颜色寄存器
0x24	DMA2D_BGPFCCR	0x00000000	DMA2D 背景层 PFC 控制寄存器

0x28	DMA2D_BGCOLR	0x00000000	DMA2D 背景层颜色寄存器
0x2C	DMA2D_FGCMAR	0x00000000	DMA2D 前景层 CLUT 存储器地址寄存器
0x30	DMA2D_BGCMAR	0x00000000	DMA2D 背景层 CLUT 存储器地址寄存器
0x34	DMA2D_OPFCCR	0x00000000	DMA2D 输出 PFC 控制寄存器
0x38	DMA2D_OCOLR	0x00000000	DMA2D 输出颜色寄存器
0x3C	DMA2D_OMAR	0x00000000	DMA2D 输出存储器地址寄存器
0x40	DMA2D_OOR	0x00000000	DMA2D 输出偏移寄存器
0x44	DMA2D_NLR	0x00000000	DMA2D 行数寄存器
0x48	DMA2D_LWR	0x00000000	DMA2D 行水印寄存器
0x4C	DMA2D_AMTCR	0x00000000	DMA2D AHB 主设备定时器配置寄存器

41.5.2. 控制寄存器(DMA2D_CR: 00h)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18	AHB_LOCK	R/W	0	AHB 锁定
17:16	MODE	R/W	0	DMA2D 模式。 此位由软件置 1 和清零。无法在传输进行时修改此位。 00：存储器到存储器（仅限 FG 获取） 01：存储器到存储器并执行 PFC（仅限 FG PFC 激活时的 FG 获取） 10：存储器到存储器并执行混合（执行 PFC 和混合时的 FG 和 BG 获取） 11：寄存器到存储器（无 FG 和 BG，仅输出阶段激活）
15:14	RSV	-	-	保留
13	CEIE	R/W	0	配置错误中断使能。
12	CTCIE	R/W	0	CLUT 传输完成中断使能
11	CAEIE	R/W	0	CLUT 访问错误中断使能
10	TWIE	R/W	0	传输水印中断使能
9	TCIE	R/W	0	传输完成中断使能
8	TEIE	R/W	0	传输错误中断使能
7:3	RSV	-	-	保留
2	ABORT	R/W	0	中止 此位可用于中止当前传输。此位可通过软件置 1 并在 START 位复位时由硬件自动复位。

1	SUSP	R/W	0	挂起 此位可用于挂起当前传输。此位由软件置 1 和复位。此位在 START 位复位时由硬件自动复位。
0	START	R/W	0	启动 此位可用于根据各种配置寄存器中加载的参数启动 DMA2D。在下列情况下将自动复位此位。 -传输结束时 -通过用户应用程序将 DMA2D_CR 中的 ABORT 位置 1 中止数据传输时。 -数据传输出错时 -因配置错误或已经进行其他传输操作（CLUT 自动加载）导致数据传输未启动时。

41.5.3. 中断状态寄存器(DMA2D_ISR: 04h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	CEIF	RO	0	配置错误中断标志。 DMA2D_CR、DMA2D_FGPFCCR 或 DMA2D_BGPFCCR 的 START 位置 1 以及编程了错误的配置时，此位置 1。
4	CTCIF	RO	0	CLUT 传输完成中断标志。 完成将 CLUT 从系统存储区复制到 DMA2D 内部存储器时，此位置 1。
3	CAEIF	RO	0	CLUT 访问错误中断标志。 在从系统存储器自动将 CLUT 复制到 DMA2D 内部存储器期间，CPU 若访问 CLUT，此位将置 1。
2	TWIF	RO	0	传输水印中断标志。 带水印的最后一个像素完成传输时，此位置 1。
1	TCIF	RO	0	传输完成中断标志。 DMA2D 传输操作完成（仅限数据传输）时此位置 1。
0	TEIF	RO	0	传输错误中断标志。 DMA 传输期间（数据传输或 CLUT 自动加载）出错时此位置 1。

41.5.4. 中断标志清零寄存器(DMA2D_IFCR: 08h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	CCEIF	RO	0	清除配置错误中断标志。

4	CCTCIF	RO	0	清除 CLUT 传输完成中断标志。
3	CCAEIF	RO	0	清除 CLUT 访问错误中断标志。
2	CTWIF	RO	0	清除传输水印中断标志。
1	CTCIF	RO	0	清除传输完成中断标志。
0	CTEIF	RO	0	清除传输错误中断标志。

41.5.5. 前景层存储器地址寄存器(DMA2D_FGMAR: 0Ch)

位域	名称	属性	复位值	描述
31:0	MA	R/W	0	存储器地址 前景层图像所用数据的地址。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 地址对齐必须与所选图像格式相匹配。

41.5.6. 前景层偏移寄存器(DMA2D_FGOR: 10h)

位域	名称	属性	复位值	描述
31:14	RSV	-	0	保留
13:0	LO	R/W	0	行偏移。 用于前景层图像的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确认下一行的起始地址。 只有在禁止数据传输的情况下才能写下这些位。数据传输一旦启动，这些将变成为只读。 如果图像格式为每像素 4 位，则行偏移值必须为偶数。

41.5.7. 背景层存储器地址寄存器(DMA2D_BGMAR: 14h)

位域	名称	属性	复位值	描述
31:0	MA	R/W	0	存储器地址 背景层图像所用数据的地址。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 地址对齐必须与所选图像格式相匹配。

41.5.8. 背景层偏移寄存器(DMA2D_BGOR: 18h)

位域	名称	属性	复位值	描述
31:14	RSV	-	0	保留
13:0	LO	R/W	0	行偏移。 用于背景层图像的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确认下一行的起始地址。 只有在禁止数据传输的情况下才能写下这些位。数据传输一旦启动，这些将变成为只读。 如果图像格式为每像素 4 位，则行偏移值必须为偶数。

41.5.9. 前景层 PFC 控制寄存器(DMA2D_FGPFCCR: 1Ch)

位域	名称	属性	复位值	描述
31:24	ALPHA	R/W	0	Alpha 值。 这些位定义固定的 Alpha 通道值，该位可替代原始的 Alpha 值或与原始的 Alpha 值相乘，具体取决于通过 AM 位选择的 Alpha 模式。
23:18	RSV	-	0	保留
17:16	AM	R/W	0	Alpha 模式。用于选择将用于前景层图像的 alpha 通道值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。 00：不修改前景层图像的 Alpha 通道值。 01：原始前景层图像的 Alpha 替换为 ALPHA。 10：原始前景层图像的 Alpha 替换位 ALPHA 与原始 Alpha 通道值的乘积。
15:8	CS	R/W	0	CLUT 大小。 这些位定义前景层图像所用的 CLUT 的大小。CLUT 传输一旦启动，此字段将变为只读。CLUT 条目数等于 CS[7:0]+1。
7:6	RSV	-	-	保留
5	START	WO	0	启动。 可将此位置 1 以启动 CLUT 的自动加载过程。该位在以下情况下自动复位。 -传输结束时 -通过用户应用程序将 DMA2D_CR 的 ABORT 位置 1 中止传输时。 -传输出错时 -因配置错误或已经在进行其他传输操作（数据传输或自动背景层 CLUT 传输）导致传输未启动时。
4	CCM	R/W	0	CLUT 颜色模式 此位定义 CLUT 的颜色格式。只有在禁止数据传输的情况下才能写入该位。CLUT 传输一旦启动，此位未将变成只读。 0：ARGB8888 1：RGB888

3:0	CM	R/W	0	这些位定义前景层图像的颜色格式。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。 0000：ARGB8888 0001：RGB888 0010：RGB565 0010：ARGB1555 0100：ARGB4444 0101：L8 0110：AL44 0111：AL88 1000：L4 1001：A8 1010：A4

41.5.10. 前景层偏移寄存器(DMA2D_FGCOLR: 20h)

位域	名称	属性	复位值	描述
31:24	RSV	-	0	保留
23:16	RED	R/W	0	红色值。这些位定义层图像的 A4 或 A8 的红色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。
15:8	GREEN	R/W	0	绿色值。这些位定义层图像的 A4 或 A8 的绿色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。
7:0	BLUE	R/W	0	蓝色值。这些位定义层图像的 A4 或 A8 的蓝色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

41.5.11. 背景层 PFC 控制寄存器(DMA2D_BGPFCCR: 24h)

位域	名称	属性	复位值	描述
31:24	ALPHA	R/W	0	Alpha 值。 这些位定义固定的 Alpha 通道值，该位可替代原始的 Alpha 值或与原始的 Alpha 值相乘，具体取决于通过 AM 位选择的 Alpha 模式。
23:18	RSV	-	0	保留
17:16	AM	R/W	0	Alpha 模式。用于选择将用于背景层图像的 alpha 通道值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。 00：不修改背景层图像的 Alpha 通道值。 01：原始背景层图像的 Alpha 替换为 ALPHA。 10：原始背景层图像的 Alpha 替换位 ALPHA 与原始 Alpha 通道值的乘积。

15:8	CS	R/W	0	CLUT 大小。 这些位定义背景层图像所用的 CLUT 的大小。CLUT 传输一旦启动，此字段将变为只读。CLUT 条目数等于 CS[7:0]+1。
7:6	RSV	-	-	保留
5	START	WO	0	启动。 可将此位置 1 以启动 CLUT 的自动加载过程。该位在以下情况下自动复位。 -传输结束时 -通过用户应用程序将 DMA2D_CR 的 ABORT 位置 1 中止传输时。 -传输出错时 -因配置错误或已经在进行其他传输操作（数据传输或自动前景层 CLUT 传输）导致传输未启动时。
4	CCM	R/W	0	CLUT 颜色模式 此位定义 CLUT 的颜色格式。只有在禁止数据传输的情况下才能写入该位。CLUT 传输一旦启动，此位未将变成只读。 0：ARGB8888 1：RGB888
3:0	CM	R/W	0	这些位定义背景层图像的颜色格式。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。 0000：ARGB8888 0001：RGB888 0010：RGB565 0010：ARGB1555 0100：ARGB4444 0101：L8 0110：AL44 0111：AL88 1000：L4 1001：A8 1010：A4

41.5.12. 前景层偏移寄存器(DMA2D_BGCOLR: 28h)

位域	名称	属性	复位值	描述
31:24	RSV	-	0	保留
23:16	RED	R/W	0	红色值。这些位定义背景层图像的 A4 或 A8 的红色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。
15:8	GREEN	R/W	0	绿色值。这些位定义背景层图像的 A4 或 A8 的绿色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

7:0	BLUE	R/W	0	蓝色值。这些位定义背景层图像的 A4 或 A8 的蓝色值。只有在禁止数据传输对的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

41.5.13. 前景层 CLUT 存储器地址寄存器(DMA2D_FGCMAR: 2Ch)

位域	名称	属性	复位值	描述
31:0	MA	R/W	0	存储器地址 专用于前景层图像的 CLUT 所用数据的地址。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 地址对齐必须与所选图像格式相匹配。

41.5.14. 背景层 CLUT 存储器地址寄存器(DMA2D_BGCMAR: 30h)

位域	名称	属性	复位值	描述
31:0	MA	R/W	0	存储器地址 专用于背景层图像的 CLUT 所用数据的地址。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 地址对齐必须与所选图像格式相匹配。

41.5.15. 输出 PFC 控制寄存器(DMA2D_OPFCCR: 34h)

位域	名称	属性	复位值	描述
31:3	RSV	-	0	保留
2:0	CM	R/W	0	颜色模式。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 000：ARGB8888 001：RGB888 010：RGB565 011：ARGB1555 100：ARGB4444

41.5.16. 输出颜色寄存器(DMA2D_OCOLR: 38h)

位域	名称	属性	复位值	描述
31:24	ALPHA	R/W	0	Alpha 通道值。这些位定义输出颜色的 alpha 通道。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

23:16	RED	R/W	0	红色值。这些位定义输出图像的红色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。
15:8	GREEN	R/W	0	绿色值。这些位定义输出图像的绿色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。
7:0	BLUE	R/W	0	蓝色值。这些位定义输出图像的蓝色值。只有在禁止数据传输的情况下才能写入这些位。传输一旦启动，这些位将变为只读。

41.5.17. 输出存储器地址寄存器(DMA2D_OMAR: 3Ch)

位域	名称	属性	复位值	描述
31:0	MA	R/W	0	存储器地址 输出 FIFO 所用数据的地址。只有禁止数据传输的情况下才能写入该寄存器。数据传输一旦启动，此寄存器即变成只读。 地址对齐必须与所选图像格式相匹配。

41.5.18. 输出偏移寄存器(DMA2D_OOR: 40h)

位域	名称	属性	复位值	描述
31:14	RSV	-	0	保留
13:0	LO	R/W	0	行偏移。 用于输出的行偏移（以像素表示）。此值用于生成地址。行偏移将添加到各行末尾，用于确认下一行的起始地址。 只有在禁止数据传输的情况下才能写下这些位。数据传输一旦启动，这些将变成为只读。 如果图像格式为每像素 4 位，则行偏移值必须为偶数。

41.5.19. 行数寄存器(DMA2D_NLR: 44h)

位域	名称	属性	复位值	描述
31:30	RSV	-	0	保留
29:16	PL	R/W	0	每行像素数。 待传输区域的每行像素数。只有在禁止数据传输的情况下才能写入这些位。数据传输一旦启动，这些将变成为只读。
15:0	NL	R/W	0	行数。 待传输区域的行数。 只有在禁止数据传输的情况下才能写下这些位。数据传输一旦启动，这些将变成为只读。

41.5.20. 行水印寄存器(DMA2D_LWR: 48h)

位域	名称	属性	复位值	描述
31:16	RSV	-	0	保留
15:0	LW	R/W	0	行水印。 这些位可用以配置可产生中断的行水印。 在待水印行的最后一个像素传输完成时产生中断。只有在禁止数据传输的情况下才能写入这些位。数据传输一旦启动，这些将变成为只读。

41.5.21. AHB 主设备定时器配置寄存器(DMA2D_AMTCR: 4Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	0	保留
15:8	DT	R/W	0	死区 在 AHB 主设备端口上两个连续访问之间所插入的死区值，以 AHB 时钟周期数表示。这些位表示两个连续 AHB 访问之间允许占用的最小周期数。
7:1	RSV	-	-	保留
0	EN	R/W	0	使能 使能死区功能。

42. 模数转换器（ADC）

42.1. 概述

内置 3 个 12 位 5Msps 采样率的逐次比较型 ADC：

- ADC1 与 ADC2 紧密耦合，可在双重模式下运行(ADC1 为主器件，ADC2 为从器件)
- ADC3 为单独模块

每个 ADC 由 12 位逐次逼近型模数转换器组成。

每个 ADC 的复用通道多达 19 个。每次 A/D 转换可以按照单次、连续或间断模式执行。ADC 的结果存储在一个左对齐或右对齐的 16 位数据寄存器中。

每个 ADC 支持最大 16 次可设通道的规则转换，以及 4 次通道可设的注入转换。

ADC 具有模拟看门狗特性，允许应用检测输入电压是否超过了用户自定义的阈值上限或下限。

ADC 映射到 AHB 总线，从而可实现快速数据处理。

内置硬件过采样器，可提高模拟性能，同时还能减轻 CPU 进行相关计算的负担。

转换逻辑 ADC_CLK 由 ADC 专用时钟内部分频产生，ADC_CLK 不得超过 75MHz。

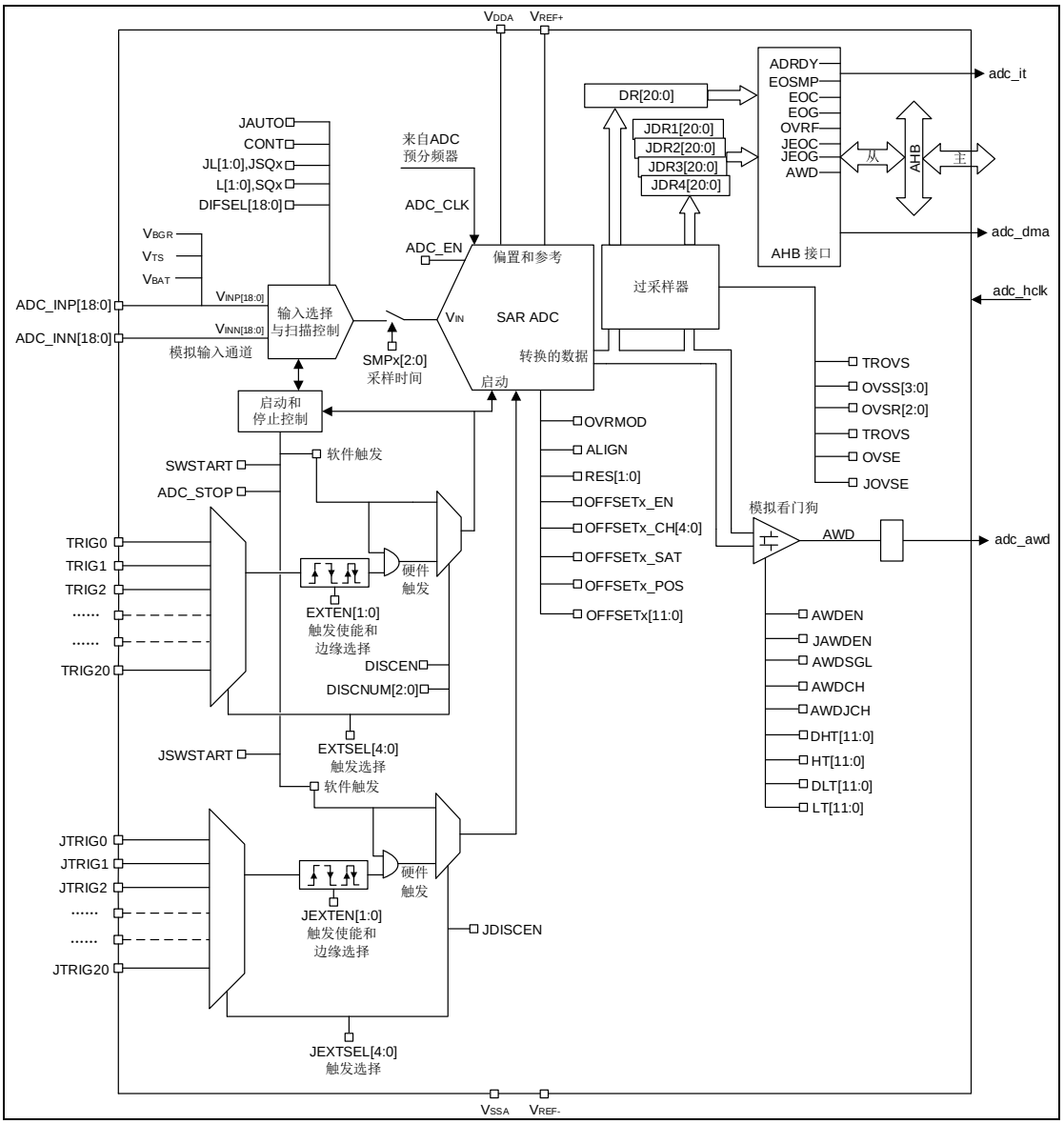
42.2. 主要特性

- 多达 3 个 ADC，其中 ADC1 和 ADC2 可以在双重模式下运行
  - ADC1 连接 16 个外部通道+2 个内部通道
  - ADC2 连接 14 个外部通道+4 个内部通道
  - ADC3 连接 16 个外部通道+2 个内部通道
- 12 位分辨率，也可配置成 10 位、8 位或 6 位分辨率
- 可通过降低分辨率来缩短转换时间
- 转换速率最高可达 5Msps（12 位分辨率）
- 支持自校准
- ADC 转换时间可以与 AHB 总线时钟频率无关
- 每个 ADC 包含 19 个通道，其中通道 0 为校准通道
- 芯片共有 4 个快速通道（ADC12_INP1,ADC12_INP3,ADC3_INP2,ADC3_INP3）与引脚直连，需软件配置模拟开关断开（寄存器 GPIOx_AIEN 对应 bit 清 0），其它通道通过模拟开关与引脚连接，因此需软件配置模拟开关闭合（寄存器 GPIOx_AIEN 对应 bit 置 1）
- 包含 7 条内部通道
  - DAC2 的通道 1 和通道 2 连接到 ADC1
  - 内建 BGR 连接到 ADC2
  - DAC1 的通道 1 和通道 2 连接到 ADC2
  - 温度传感器连接到 ADC3
  - VBAT 连接到 ADC3
- 支持单端输入或差分输入（可按通道进行编程）
- 采样结束、转换结束、组转换结束、模拟看门狗事件或溢出事件时产生中断

- 支持单次、连续转换模式
- 支持间断模式
- 最多支持 16 个规则通道组和 4 个注入通道组
- 采样时间可以按通道分别编程
- 规则转换和注入转换均有外部触发选项
- 规则通道转换期间有 DMA 请求产生
- AHB 总线便于系统集成，同时实现高速的读写操作
- 数据对齐以保持内置数据一致性
- 支持双重模式（两个 ADC 设备）
- 过采样器
  - 16 位数据寄存器
  - 过采样率可以在 2 到 256 之间调整
  - 可编程数据移位高达 8 位
- 支持模拟看门狗
- ADC 供电要求：1.8V~3.6V
- ADC 输入范围：VREF- ≤ VIN ≤ VREF+

42.3. 结构框图

图 42-1 ADC 框图



42.4. 引脚和内部信号

表 42-1 ADC 输入/输出引脚

引脚名称	信号类型	说明
V_{REF+}	输入，模拟参考正极	ADC 使用的正极参考电压 $1.8V \le V_{REF+} \le V_{DDA}$
V_{DDA}	输入，模拟电源	模拟电源等于 V_{DDA} $1.8V \le V_{DDA} \le 3.6V$
V_{REF-}	输入，模拟参考负极	ADC 使用的负极参考电压 $V_{REF-}=V_{SSA}$
V_{SSA}	输入，模拟电源地	等效于 VSS 模拟电源地
V_{INP}[18:0]	ADC 的正输入模拟通道	连接到外部通道 ADC_INP[i]或内部通道

V_{INN}[18:0]	ADC 的负输入模拟通道	连接到外部通道 ADC_INN[i]或 V_{REF-}
ADC_INP[18:0]	输入，正外部模拟输入	多达 19 个模拟输入通道
ADC_INN[18:0]	输入，负外部模拟输入	多达 19 个模拟输入通道

表 42-2 ADC 内部输入/输出信号

内部信号名称	信号类型	说明
TRIG[20:0]	输入	多达 21 个外部触发输入用于规则通道转换。
JTRIG[20:0]	输入	多达 21 个外部触发输入用于注入通道转换。
adc_awd	输出	内部模拟看门狗输出信号，连接至片上定时器。
adc_it	输出	ADC 中断信号。
adc_hclk	输入	AHB 时钟。
adc_dma	输出	ADC DMA 请求。

表 42-3 ADC 内部互连

信号名称	说明
ADC1_INP[12]	DAC2_OUT1，来源于数模转换器 DAC2 的输出信号 1
ADC1_INP[13]	DAC2_OUT2，来源于数模转换器 DAC2 的输出信号 2
ADC2_INP[13]	VBGR，内部 1.2V 基准电压
ADC2_INP[16]	DAC1_OUT1，来源于数模转换器 DAC1 的输出信号 1
ADC2_INP[17]	DAC1_OUT2，来源于数模转换器 DAC1 的输出信号 2
ADC3_INP[17]	VBAT，电池电压的 1/4 分压
ADC3_INP[18]	VTS，内部温度传感器的输出电压

42.5. 功能描述

42.5.1. 通道选择

一共有 19 个 ADC 通道(通道 0-通道 18)，其中通道 0（ADC_INP/INN0）为校准通道。使用时可以把转换组织成两组：规则组和注入组。在任意多个通道上以任意顺序进行的一系列转换构成成组转换。例如，可以如下顺序完成转换：ADC_INP/INN3、ADC_INP/INN8、ADC_INP/INN2、ADC_INP/INN2、ADC_INP/INN7、ADC_INP/INN2、ADC_INP/INN2、ADC_INP/INN15。

- 规则组由多达 16 个转换组成。规则通道和它们的转换顺序在 ADC_SQRx 寄存器中选择。规则组中转换的总数应写入 ADC_SQR1 寄存器的 L[3:0]位中。
- 注入组最多支持 4 个转换组成。注入通道在 ADC_JSQR 寄存器中选择，注入组的转换总数写入 ADC_JSQR 的 JL[1:0]位。

如果 ADC_SQRx 或 ADC_JSQR 寄存器在转换期间被更改，当前的转换继续完成，随后序列以新组继续进行转换。

表 42-4 ADC 模拟通道选择

	ADC1	ADC2	ADC3
INP0	校准通道	校准通道	校准通道
INN0	校准通道	校准通道	校准通道
INP1	PA5	PA5	PC3
INN1	NC	NC	NC
INP2	PF11	PF13	PH4
INN2	PF12	PF14	PF10
INP3	PA4	PA4	PH5
INN3	PA5	PA5	PH4
INP4	PC4	PC4	PF5
INN4	PC5	PC5	PH5
INP5	PB1	PB1	PF3
INN5	PB0	PB0	PF4
INP6	PF12	PF14	PF10
INN6	NC	NC	NC
INP7	PA7	PA7	PF8
INN7	NC	NC	NC
INP8	PC5	PC5	PF6
INN8	NC	NC	NC
INP9	PA6	PA6	PF4
INN9	PA7	PA7	NC
INP10	PC0	PC0	PC0
INN10	PC1	PC1	PC1
INP11	PC1	PC1	PC1
INN11	NC	NC	NC
INP12	DAC2_OUT1	PB2	PC2
INN12	NC	NC	NC
INP13	DAC2_OUT2	VBGR	PH2
INN13	NC	NC	PH3
INP14	PA2	PA2	PH3
INN14	NC	NC	PF8
INP15	PA3	PA3	PF9
INN15	NC	NC	PF6
INP16	PA0	DAC1_OUT1	PF7
INN16	PA1	NC	NC

INP17	PA1	DAC1_OUT2	VBAT
INN17	NC	NC	NC
INP18	PB0	PB0	VTS
INN18	NC	NC	NC

42.5.2. 单次转换模式

在单次转换模式下，ADC 模块只执行一组转换。CONT 位为 0 时，可通过以下方式启动此模式：

- 将 ADC_CR1 寄存器中的 SWSTART 位置 1（仅适用于规则通道）
- 将 ADC_CR1 寄存器中的 JSWSTART 位置 1（仅适用于注入通道）
- 外部触发（适用于规则通道和注入通道）

每次转换结束：

- 每次规则通道转换结束：
  - 转换数据被储存在 16 位 ADC_DR 寄存器中
  - EOC(转换结束)标志被设置
  - 如果设置了 EOCIE，则产生 EOC 中断。
  - 如果所有规则通道（由 ADC_SQR1.L 决定的转换长度）转换结束，EOG(规则组转换结束)标志被置位
  - 如果设置了 EOGIE，则产生 EOG 中断，ADC 停止
- 每次注入通道转换结束：
  - 转换数据被存储在 16 位 ADC_JDRx 寄存器
  - JOEC(转换结束)标志被设置
  - 如果设置了 JEOCIE，则产生 JEOC 中断
  - 如果所有注入通道（由 ADC_JQR.JL 决定的转换长度）转换结束，JEOG(注入组转换结束)标志被置位
  - 如果设置了 JEOGIE，则产生 JEOG 中断

如果要只转换一个通道，规则组或注入组的长度设置为 1。

42.5.3. 连续转换模式

连续转换只对规则组有效，连续转换模式中，当前面 ADC 转换一结束马上就启动另一次转换。CONT 位为 1 时，此模式可通过外部触发启动或将 ADC_CR1 寄存器中的 SWSTART 位置 1 来启动（仅适用于规则通道）。

每次转换结束：

- 转换数据被储存在 16 位的 ADC_DR 寄存器中
- EOC(转换结束)标志被设置
- 如果设置了 EOCIE，则产生中断
- 如果所有规则通道（由 ADC_SQR1.L 决定的转换长度）转换结束，EOG(规则组转换结束)标志被置位
- 如果设置了 EOGIE，则产生 EOG 中断

注入通道组不支持连续转换模式。只有在设置自动注入模式（JAUTO=1），在规则组完成后，注入通道自动开始。

42.5.4. 间断模式

对于规则组，此模式通过设置 ADC_CR1 的 DISCEN 位激活。它可以用来在规则组的转换中执行一个短序列的 n(n<=8)次转换，此转换是 ADC_SQRx 寄存器所选择的转换序列的一部分。数值 n 由 ADC_CR1 寄存器的 DISCNUM[2:0]位给出。

一个触发信号可以启动 ADC_SQRx 寄存器中描述的下轮 n 次转换，直到此序列所有的转换完成为止。总的序列长度由 ADC_SQR1 寄存器的 L[3:0]定义。

举例：

n=3，被转换的通道=1、2、3、4、6、7、9、10

第 1 次触发：转换的序列为 1,2,3

第 2 次触发：转换的序列为 4,6,7

第 3 次触发：转换的序列为 9,10

第 4 次触发：转换的序列为 1,2,3

注意：当以间断模式转换一个规则组时，转换序列结束后不自动从头开始。当所有子组被转换完成，下一次触发启动第一个子组的转换。在上述示例中，第 4 次触发重新转换了第 1 个子组中的 1,2,3，而不是在第 3 次触发中转换通道 1。

每次转换结束均产生 EOC 事件，一个规则组转换结束后产生 EOG 事件。当 ADC_CR1 寄存器的 DISCEN 位和 CONT 位同时使能时，即间断模式和连续转换模式同时使能，规则通道以连续模式进行转换。

对于注入组，可将 ADC_CR1 寄存器中的 JDISCEN 位置 1 来使能此模式。在出现触发事件之后，可使用该模式逐通道转换在 ADC_JSQR 寄存器中选择的序列，每次触发只进行一次转换。相当于规则组间断模式中的 n=1。

出现触发事件时，将启动在 ADC_JSQR 寄存器中选择的下一个通道转换，直到序列中的所有转换均完成为止。通过 ADC_JSQR 寄存器中的 JL[1:0] 位定义总序列长度。

示例：

n = 1，要转换的通道 = 1、2、3

第 1 次触发：转换通道 1

第 2 次触发：转换通道 2

第 3 次触发：转换通道 3

第 4 次触发：转换通道 1

注意：

1.转换完所有注入通道后，下一个触发信号将启动第一个注入通道的转换。在上述示例中，第 4 次触发重新转换了第 1 个注入通道。在每次转换结束均产生 JEOC 事件，一个规则组转换结束后产生 JEOG 事件。

2.不能同时使用自动注入和间断采样模式。

42.5.5. 注入通道管理

触发注入

要使用触发注入，必须将 ADC_CR1 寄存器中的 JAUTO 位清零。如果在规则通道组转换期间出现外部注入触发或者 JSWSTART 位置 1，则当前的转换会复位，并且注入通道序列会切换为单次扫描模式。然后，规则通道组的规则转换会从上次中断的规则转换处恢复。如果在注入转换期间出现规则事件，注入转换不会中断，但在注入序列结束时会执行规则序列。

自动注入

如果将 JAUTO 位置 1，则注入组中的通道会在规则组通道之后自动转换。这可用于转换最多由 20 个转换构成的序列，这些转换在 ADC_SQRx 和 ADC_JSQR 寄存器中编程。在此模式下，必须禁止注入通道上的外部触发。

如果 CONT 位和 JAUTO 位均已置 1，则在转换规则通道之后会继续转换注入通道。

在 JAUTO 为 1 时，需要禁止外部注入触发，并保证 JSWSTART 为 0。

注意：不能同时使用自动注入和间断采样模式。

42.5.6. 停止控制

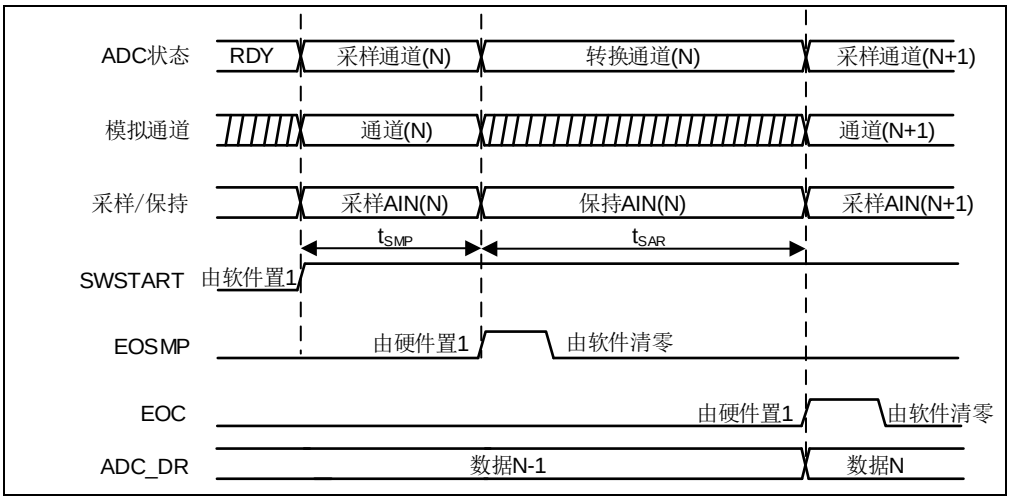
当 ADC_CR2 寄存器的 ADC_STP 置位后，ADC 控制器将在当前通道转换结束后停止 ADC 转换，并且停止后硬件会自动清除 ADC_STP 位，此时可以等待新的触发事件。当新的触发事件发生时，规则通道的转换序列将从 SQ1 开始启动，如果使能了过采样功能，过采样次数也将从 0 开始计数。

42.5.7. 时序图

如下图所示，从转换开始到转换结束所经过的总转换时间（t_{CONV}）是配置的采样时间（t_{SMP}）与逐次逼近时间（t_{SAR} 取决于数据分辨率）的总和。

$$t_{CONV} = t_{SMP} + t_{SAR}$$

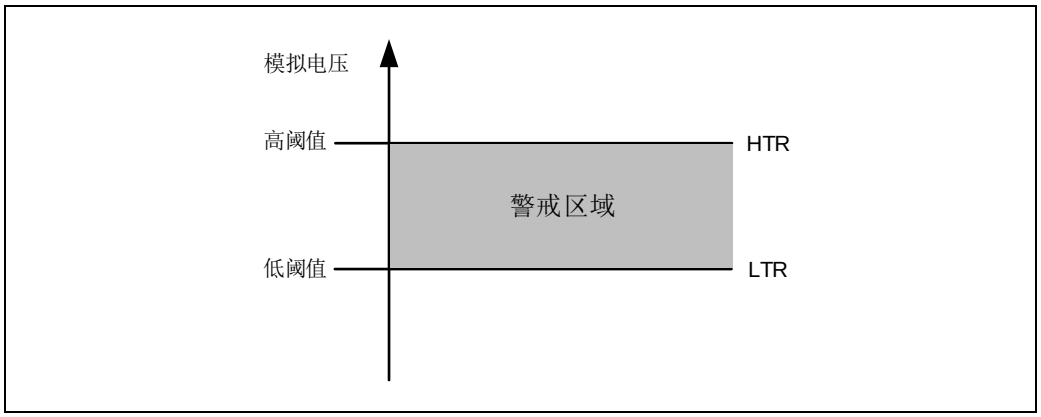
图 42-2 ADC 转换时序图



42.5.8. 模拟看门狗

如果被 ADC 转换的模拟电压低于低阈值或高于高阈值，AWD 模拟看门狗状态位被设置。单端输入通道的高低阈值分别由 ADC_HTR 寄存器的 HT 位和 ADC_LTR 寄存器的 LT 位决定，用无符号数表示；差分输入通道的高低阈值分别由 ADC_HTR 寄存器的 DHT 位和 ADC_LTR 寄存器的 DLT 位决定，也用无符号数表示。通过设置 ADC_IE 寄存器的 AWDIE 位以允许产生相应中断。

图 42-3 AWD 阈值图



通过配置 ADC_CR1 寄存器的 AWDEN 位或 JAWDEN 位，使能模拟看门狗功能，AWDSGL 位控制模拟看门狗作用于匹配 AWDCH/AWDJCH 的规则/注入通道或所有通道，如表所示。

表 42-5 AWD 控制位表

模拟看门狗警戒的通道	ADC_CR1 寄存器控制位		
	AWDSGL 位	AWDEN 位	JAWDEN 位
无	x	0	0
所有规则通道	0	1	0
所有注入通道	0	0	1
所有注入和规则通道	0	1	1
匹配 AWDCH 的规则通道	1	1	0
匹配 AWDJCH 的注入通道	1	0	1
匹配 AWDJCH 的注入或 匹配 AWDCH 的规则通道	1	1	1

对于模拟看门狗使用原始转换结果进行，不使用偏移计算后的数据。

如果转换分辨率小于 12 位（RES 位设置），由于始终对完整的 12 位原始数据比较（左对齐），因此编程阈值的 LSB 必须保持清零。

42.5.9. 数据对齐

ADC_CR2 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。可选择左对齐和右对齐两种方式，如图所示。采用左对齐时，数据基于半字对齐，分辨率设置为 6 位时除外。当分辨率设置为 6 位时，数据基于字节对齐。

注： 过采样模式下不支持左对齐。当 ROVSE 和/或 JOVSE 位置 1 时，忽略 ALIGN 位的值，并且 ADC 仅提供右对齐数据。

如果通道使能 offset 功能，转换数据将减去 ADC_OFRx 寄存器中写入的用户自定义偏移量 OFFSETx[11:0]，因此结果可以是一个负值，使用有符号数。SIGN 位表示扩展的符号值。如果不使能 offset，则转换结果为无符号数。

过采样模式下不支持偏移校准。如果 ROVSE 和/或 JOVSE 位置 1， ADCx_OFRx 寄存器中 OFFSETx_EN 位的值会被忽略（视为复位）。

图 42-4 数据右对齐

Offset禁止，无符号数

12位数据

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

10位数据

0	0	0	0	0	0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

8位数据

0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0

6位数据

0	0	0	0	0	0	0	0	0	0	D5	D4	D3	D2	D1	D0

Offset使能，有符号数

12位数据

SIGN	SIGN	SIGN	SIGN	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

10位数据

SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

8位数据

SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	D7	D6	D5	D4	D3	D2	D1	D0

6位数据

SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	SIGN	D5	D4	D3	D2	D1	D0

图 42-5 数据左对齐

42.5.10. 偏移补偿

支持可选的通道数据偏移补偿功能，通过 ADC_OFRx(x 为 1~4)寄存器设置。共有四组偏移补偿可以选择，每组有独立的使能位、补偿值、补偿方式、通道号和补偿结果格式。当对应寄存器 OFFSETx_EN 为 1 时，通过 OFFSETx_CH[4:0]选择需要进行偏移补偿的通道。使能后，此通道的转换结果减去或者加上 OFFSETx[11:0]定义的偏移量（OFFSETx_POS 选择）。结果有可能是一个负数，因此使用有符号数表示，或者强制使用无符号表示（OFFSETx_SAT 选择）。

对于有符号结果（OFFSETx_SAT=0），数据为 16 位的补码形式；如果 OFFSETx_POS 为 1，以 12 位数据右对齐为例，则 BIT12 位代表进位，非符号位。

对于无符号结果（OFFSETx_SAT=1），如果 OFFSETx_POS 为 1，加上偏移量计算，最大值为 0xFFF；减去偏移计算，最小值为 0x000。

如果多个偏移（OFFSETx）指向同一通道，相减时只会考虑使用 x 值最小的偏移。

在硬件过采样模式下，偏移补偿功能忽略。

对于模拟看门狗使用原始转换结果进行，不使用偏移计算后的数据。

42.5.11. 可编程的通道采样时间

开始转换之前，ADC 必须在待测量电压源于 ADC 内置采样电容之前建立直接连接。该采样时间必须足以使输入电压源为嵌入式电容充电至输入电压水平。

ADC 使用若干个 ADC_CLK 周期对输入电压采样，每个通道的采样周期数目 TS 可以通过 ADC_SMPRx(x=1, 2, 3)的 SMPy[3:0]位进行设置。

ADC 总转换时间 TCONV 的计算如下：其中 bit 为 12/10/8/6，由 RES[1:0]设置

TCONV = TS + (bit)TADC_CLK

例如：ADC_CLK = 40MHz，RES=0，bit=12， 采样时间为 3 周期

TCONV= (3+12) TADC_CLK = 15TADC_CLK =375ns。

ADC 通过将状态位 EOSMP 置 1 来指示采样阶段结束（仅限规则通道）。

42.5.12. 外部触发转换

转换可以由外部事件触发（例如定时器捕获，EXTI 线）。如果设置了 EXTEN 控制位，则外部事件就能够触发转换。EXTSEL[4:0]和 JEXTSEL[4:0]控制位允许应用程序选择 21 个可能的事件中的某一个，可以触发规则和注入组的采样。

注意：当外部触发信号被选为 ADC 规则或注入转换时，外部触发的极性可以通过 EXTEN[1:0]和 JEXTEN[1:0]设置。

通道的触发选择，具体选择信号见下表：

表 42-6 ADC 规则/注入转换触发源选择表

TSEL[4:0]	规则转换触发源	注入转换触发源
00000	TIM1_CC1	TIM1_TRGO
00001	TIM1_CC2	TIM1_CC4
00010	TIM1_CC3	TIM2_TRGO
00011	TIM2_CC2	TIM2_CC1
00100	TIM3_TRGO	TIM3_CC4
00101	TIM4_CC4	TIM4_TRGO
00110	EXTI11	EXTI15
00111	TIM8_TRGO	TIM8_CC4
01000	TIM8_TRGO2	TIM1_TRGO2
01001	TIM1_TRGO	TIM8_TRGO
01010	TIM1_TRGO2	TIM8_TRGO2
01011	TIM2_TRGO	TIM3_CC3
01100	TIM4_TRGO	TIM3_TRGO
01001	TIM6_TRGO	TIM3_CC1
01110	TIM15_TRGO	TIM6_TRGO
01111	TIM3_CC4	TIM15_TRGO
10000	TKEY_OUT（ADC1 和 ADC2）	TKEY_OUT（ADC1 和 ADC2）

10001	无	无
10010	LPTIM1_OUT	LPTIM1_OUT
10011	LPTIM2_OUT	LPTIM2_OUT
10010	LPTIM3_OUT	LPTIM3_OUT

可通过将 ADC_CR1 寄存器中的 SWSTART（对于规则转换）或 JSWSTART（对于注入转换）位置 1 来产生软件触发。

42.5.13. DMA 请求

因为规则通道转换的值储存在一个仅有的数据寄存器中，所以当转换多个规则通道时需要使用 DMA，这可以避免丢失已经存储在 ADC_DR 寄存器中的数据。

只有在规则通道的转换结束时才产生 DMA 请求，并将转换的数据从 ADC_DR 寄存器传输到用户指定的目的地址。

对于双 ADC 模式，主 ADC 的 DMA 请求可以在主从两个 ADC 转换都完成才产生，并且结果从 ADC_CDR 读取（DMADUAL=1X）；也可以主从 ADC 的 DMA 独立工作。对于规则交叉模式，如果同时需要使用注入功能，只能在 ADC_CDR 读取结果，DMA 也只能使用主从合并模式。

42.5.14. 双 ADC 模式

本产品支持双 ADC 模式（ADC1 为主，ADC2 为从）。 在双 ADC 模式里，根据 ADC1_CCR 寄存器中 DUALMOD[2:0]位所选的模式，转换的启动可以是 ADC1 主和 ADC2 从的交替触发或同步触发。

在双 ADC 模式下，配置外部事件触发转换时，必须设置为仅主 ADC 触发而禁止从 ADC 触发，以防止出现意外触发而启动不需要的从转换。

可实现以下四种模式：

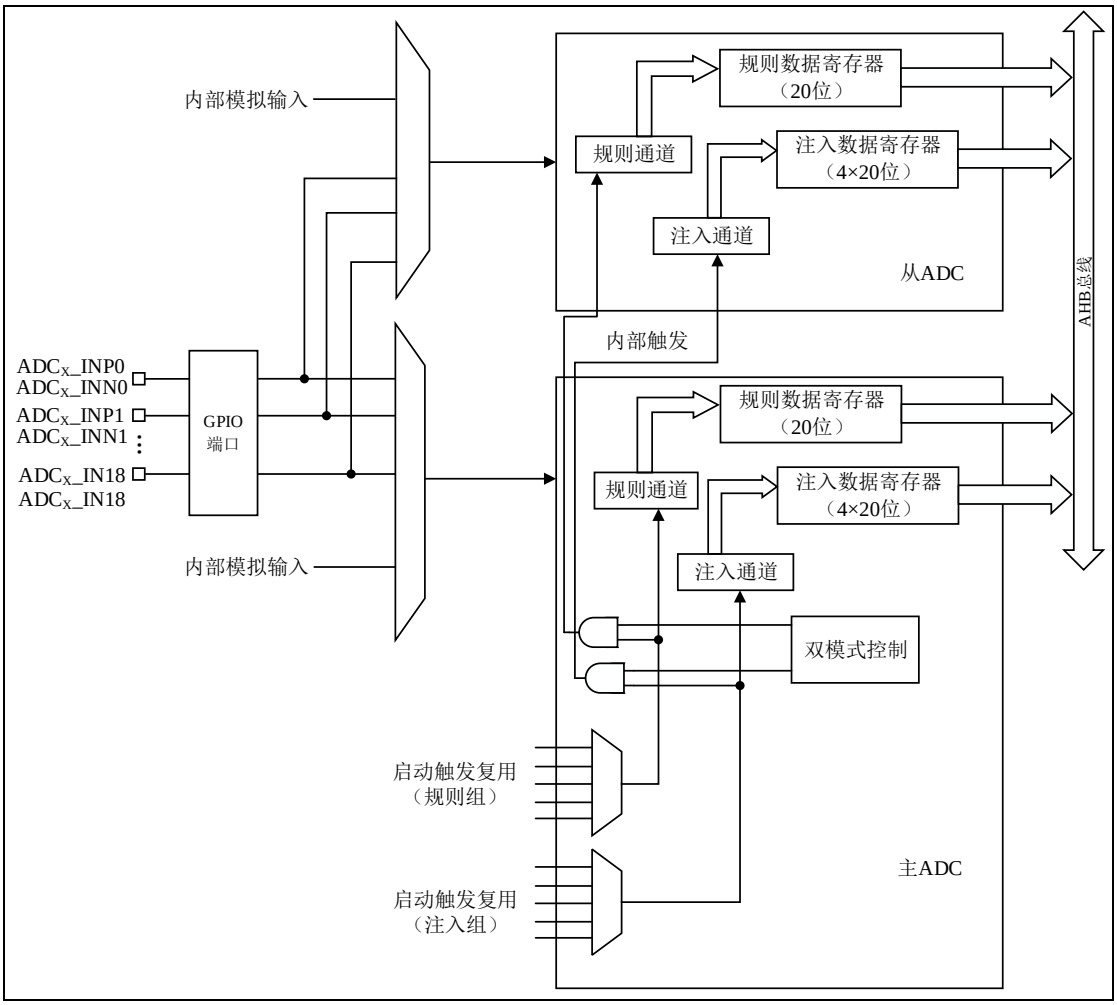
- 注入同步模式
- 规则同步模式
- 规则交叉模式
- 交替触发模式

也可按以下方式组合使用上述模式：

- 注入同步模式 + 规则同步模式
- 规则同步模式 + 交替触发模式
- 规则交叉模式 + 注入同步模式

注意：在双重 ADC 模式下，可在主从 ADC 共用规则数据寄存器（ADC_CDR）中读取转换的数据。

图 42-6 双 ADC 模式



42.5.14.1. 注入同步模式

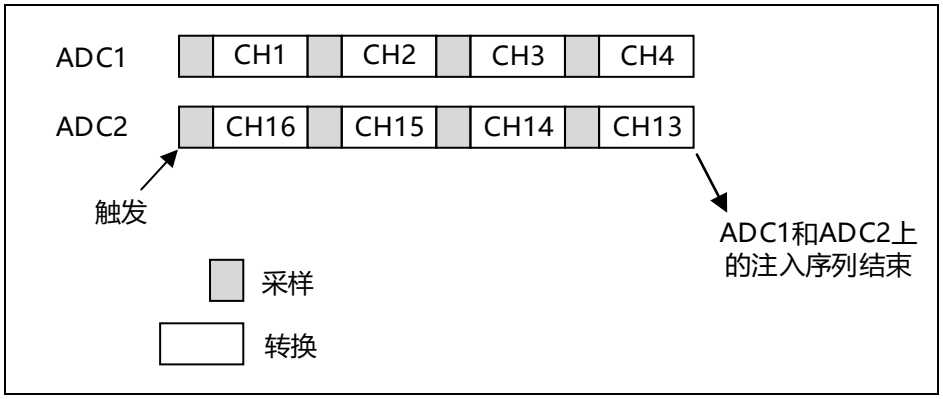
此模式转换一个注入通道组。外部触发来自 ADC1 的注入组多路选择(由 ADC1_JSQR 寄存器的 JEXTSEL[4:0]选择)，它同时给 ADC2 提供同步触发。

注意：不要在 2 个 ADC 上转换相同的通道(两个 ADC 在同一个通道上的采样时间不能重叠)。

在 ADC1 或 ADC2 的转换结束时：

- 转换的数据存储在每个 ADC 接口的 ADC_JDRx 寄存器中。
- 当 ADC1/ADC2 的注入通道全部完成转换后，会生成一个 JEOG 中断（如果已在两个 ADC 接口中的一个接口上使能)。

图 42-7 4 个通道的注入同步模式



在间断模式，每次触发只进行一次转换。

42.5.14.2. 规则同步模式

此模式转换一个规则通道组。外部触发来自 ADC1 的规则组多路选择(由 ADC1_CR1 寄存器的 EXTSEL[4:0]选择)，它同时给 ADC2 提供同步触发。

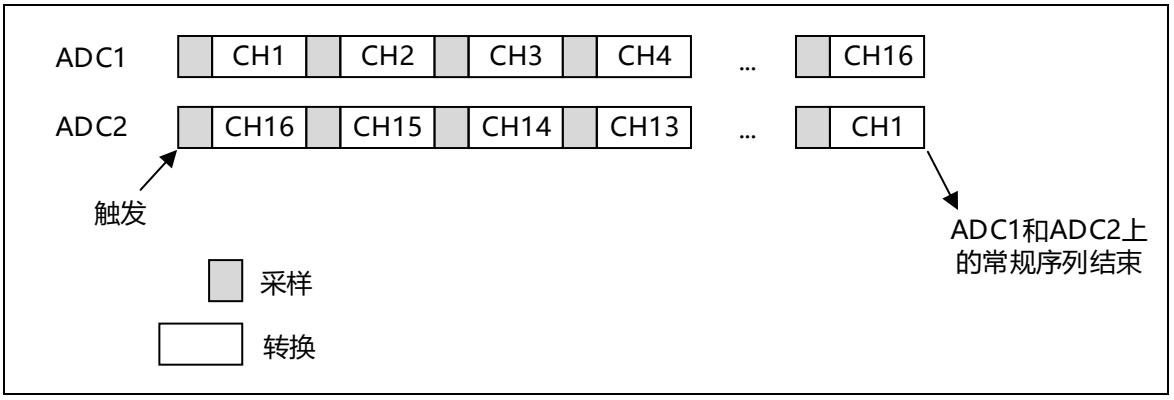
注意：不要在 2 个 ADC 上转换相同的通道(两个 ADC 在同一个通道上的采样时间不能重叠)。

在 ADC1 或 ADC2 的转换结束时：

- 产生一个 32 位 DMA 传输请求(如果设置了 DMA 位)，32 位的 ADC_CDR 寄存器内容传输到 SRAM 中，它高半个字包含 ADC2 的转换数据，低半个字包含 ADC1 的转换数据。
- 当所有 ADC1/ADC2 规则通道都被转换完时，产生 EOG 中断(若任一 ADC 接口开放了中断)。

对于规则同步模式，注入通道处理和单 ADC 模式相同，立刻结束当前规则转换，插入注入通道转换，等待注入通道完成后重新开始被打断的规则通道。

图 42-8 16 个通道的规则同步模式



42.5.14.3. 规则交叉模式

此模式只能用于规则组（通常为一个通道）。外部触发源来自 ADC1 的规则组多路选择(由 ADC1_CR1 寄存器的 EXTSEL[4:0]选择)。

出现外部触发之后：

- ADC1 立即启动
- 经过几个 ADC 时钟周期延迟后 ADC2 启动

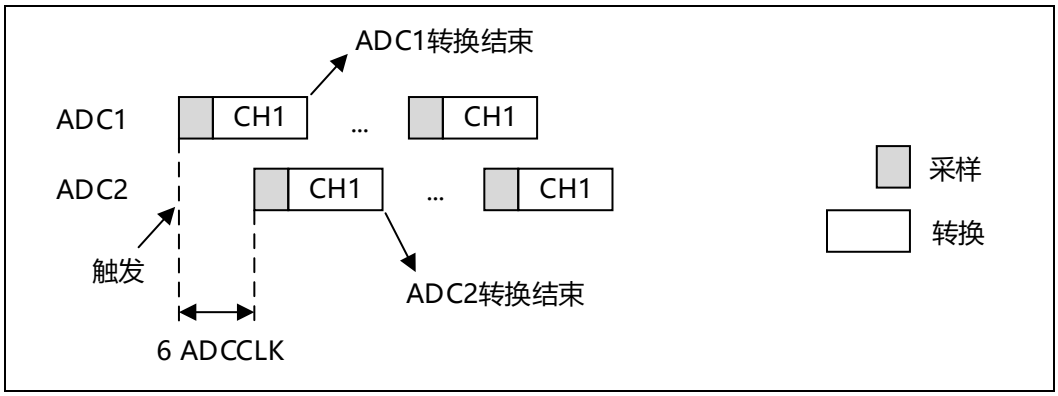
规则交叉模式下 2 个转换之间的最小延迟通过 ADC_CCR 寄存器中的 DELAY 位进行配置。但是，如果某个 ADC 的互补 ADC 仍在对其输入进行采样，则该 ADC 无法启动转换（在给定时间内，只有一个 ADC 能够对输入信号采样）。在这种情况下，延迟时间为采样时间+ 2 个 ADC 时钟周期。例如，如果两个 ADC 的 DELAY = 5 个时钟周期，且采样时间为 15 个时钟周期，则 ADC1 和 ADC2 之间的转换延迟为 17 个时钟周期。

如果 ADC1 和 ADC2 上的 CONT 位均置 1，则这两个 ADC 所选规则通道会连续进行转换。

ADC2 产生一个 EOC 中断后(需使能 EOC 中，ADC_IE.EOCIE=1)，产生一个 32 位的 DMA 传输请求(如果设置了 DMA 位)，ADC_CDR 寄存器的 32 位数据被传输到 SRAM，高半个字包含 ADC2 的转换数据，低半个字包含 ADC1 的转换数据。

对于规则交叉模式，注入通道处理和单 ADC 模式相同，立刻结束当前规则转换，插入注入通道转换，等待注入通道完成后重新开始被打断的规则通道。在注入功能使能时，规则通道只支持一个通道的连续模式，规则转换从 ADC_CDR 寄存器读取转换结果。

图 42-9 连续转换模式下 1 通道的规则交叉模式



42.5.14.4. 交替触发模式

此模式用于注入通道组。外部触发来自 ADC1 的注入组多路选择(由 ADC1_JSQR 寄存器的 JEXTSEL[4:0]选择)。

- 发生第一次触发时，将转换 ADC1 中注入组的所有通道
- 发生第二次触发时，将转换 ADC2 中注入组的所有通道
- 如此循环

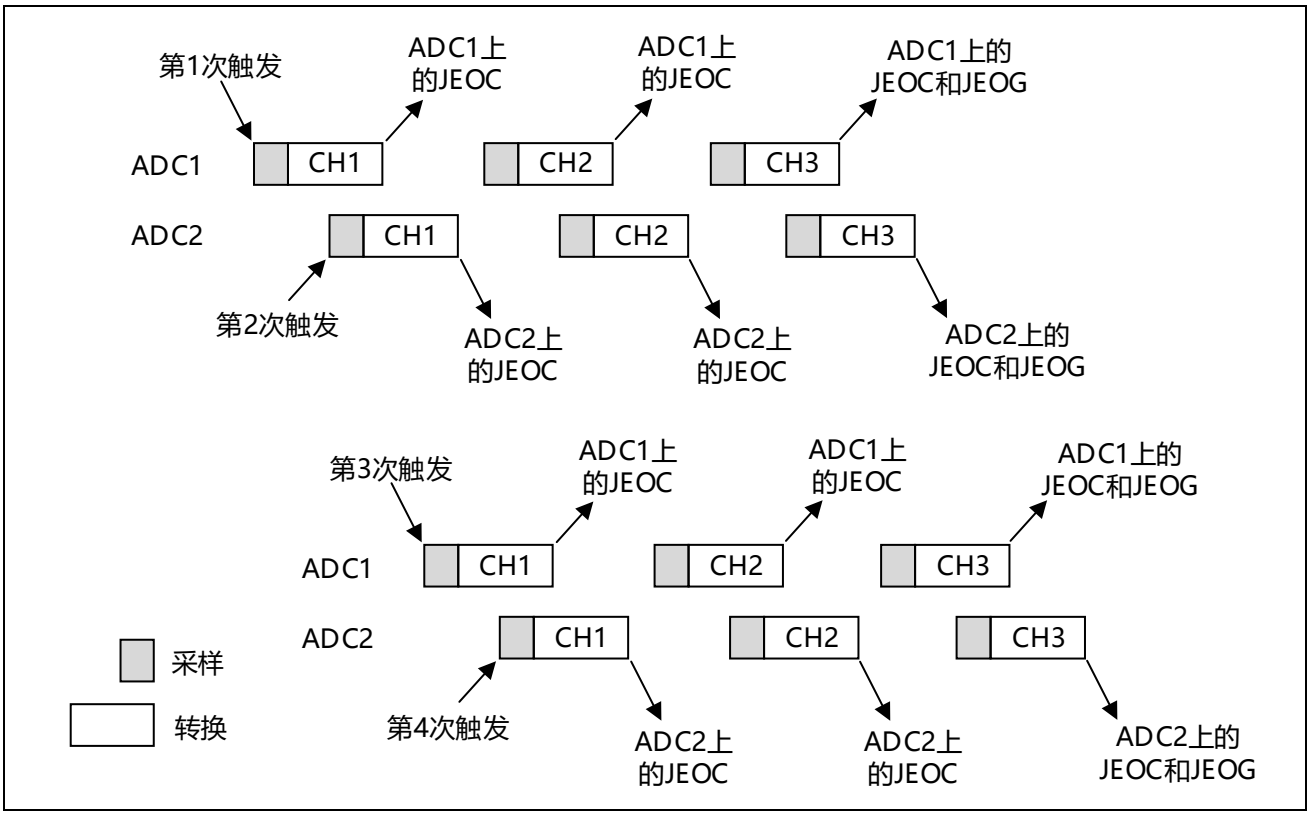
当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

当组中的所有注入 ADC2 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

如果使能了 JEOC 中断，每次转换完成都产生一个 JEOC 中断。

如果在组中的所有注入通道都完成转换后出现另一个外部触发，则可通过转换组中的注入 ADC1 通道来重新启动交替触发过程。

图 42-10 交替触发：各个 ADC 的注入组



如果使能 ADC1 和 ADC2 的注入不连续采样模式：

发生第一次触发时，将转换第一个注入 ADC1 通道

发生第二次触发时，将转换第一个注入 ADC2 通道

如此循环

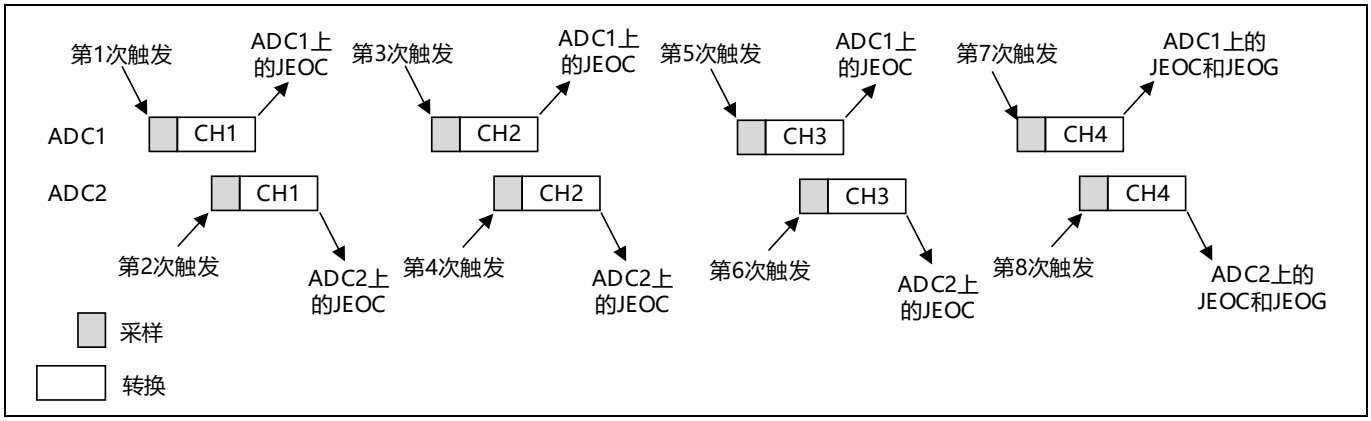
当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

当组中的所有注入 ADC2 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

如果使能了 JEOC 中断，每次转换完成都产生一个 JEOC 中断。

如果注入组中的所有通道都完成转换后出现另一个外部触发，则会重新启动交替触发过程。

图 42-11 交替触发：间断模式下的各个 ADC 的注入组



42.5.14.5. 独立模式

此模式里，双 ADC 同步不工作，每个 ADC 接口独立工作。

42.5.14.6. 混合的规则同步+注入同步模式

规则组同步转换可以被中断，以启动注入组的同步转换。

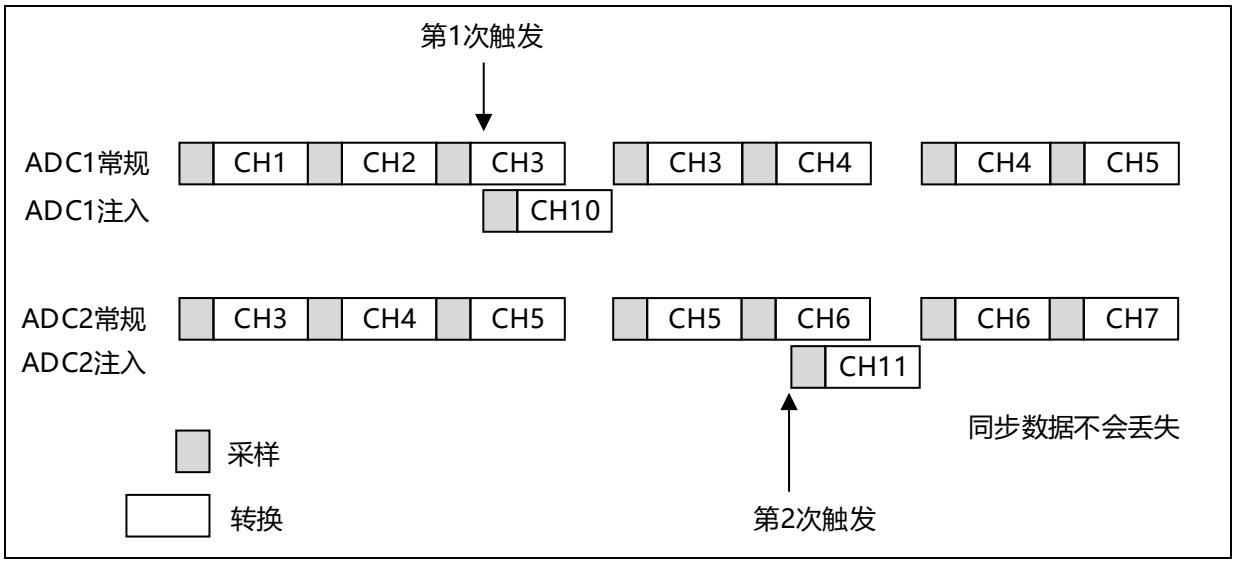
注：在混合的规则同步/注入同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

42.5.14.7. 混合的规则同步+交替触发模式

规则组同步转换可以被中断，以启动注入组交替触发转换。下图显示了一个规则同步转换被交替触发所中断。注入交替转换在注入事件到达后立即启动。如果规则转换已经在运行，为了在注入转换后确保同步，所有的 ADC(主和从)的规则转换被停止，并在注入转换结束时同步恢复。

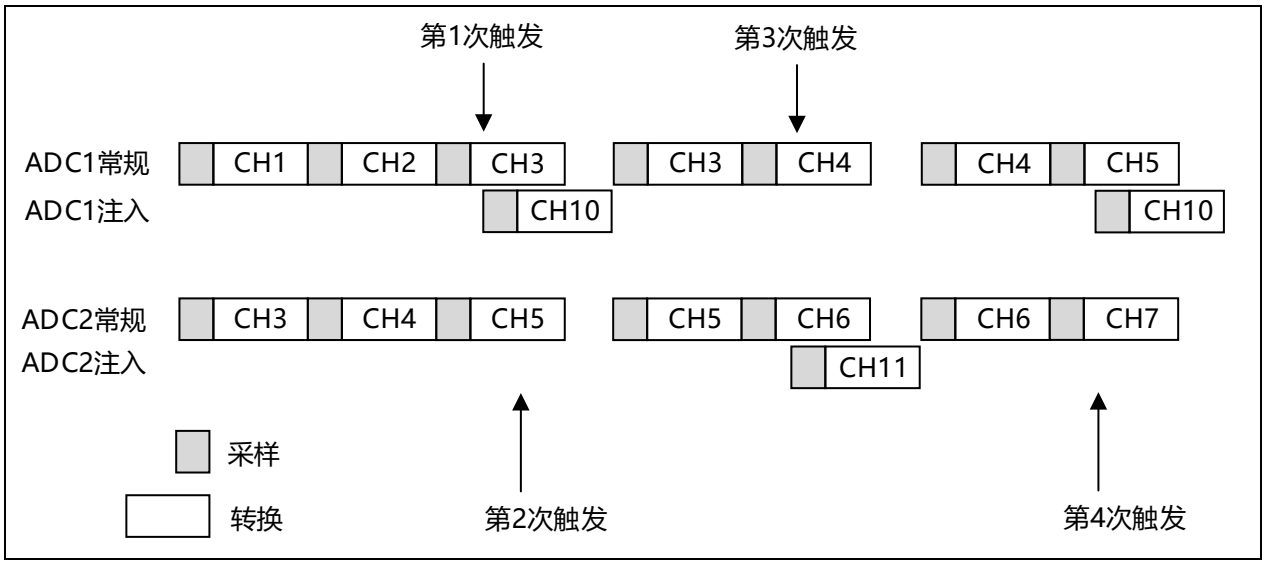
注：在混合的规则同步+交替触发模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

图 42-12 规则同步+交替触发



在已导致规则转换中断的注入转换期间出现的触发将被忽略。下图说明了这种情况下的行为（第 2 次触发被忽略）。

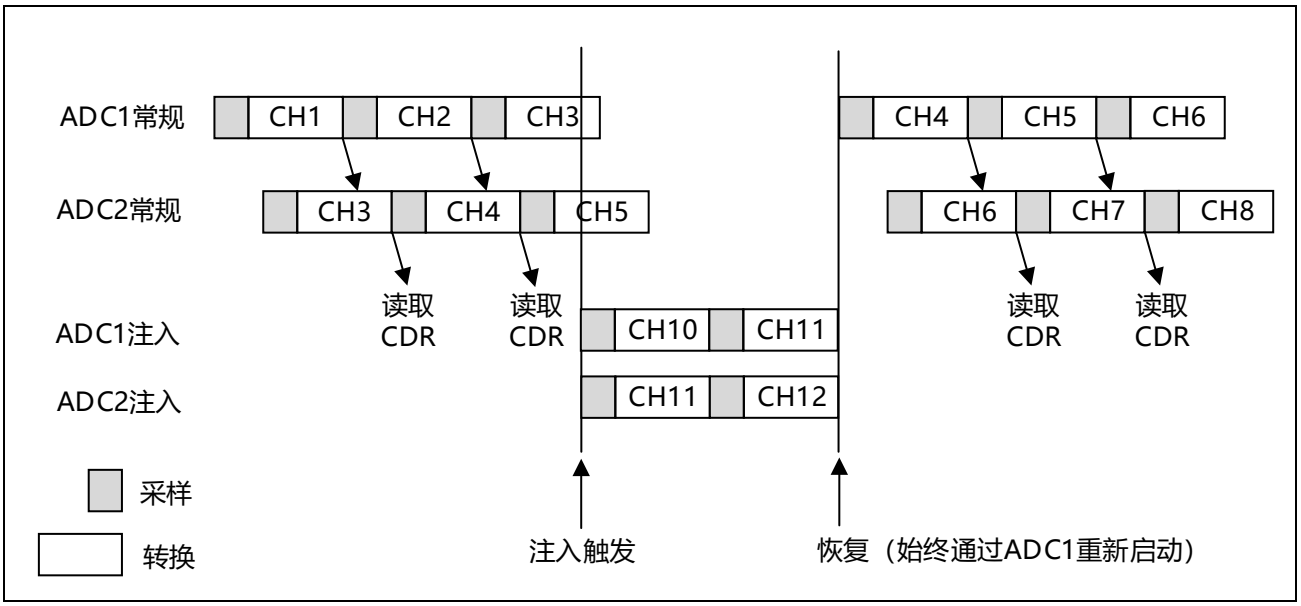
图 42-13 规则同步+交替触发：在注入转换期间出现新的注入触发事件



42.5.14.8. 混合的注入同步+规则交叉模式

一个注入事件可以中断一个交叉转换。这种情况下，交叉转换被中断，注入转换被启动，在注入序列转换结束时，交叉转换被恢复，并且从主 ADC（ADC1）开始新的转换。下图是这种情况的一个例子。

图 42-14 混合的注入同步+规则交叉模式



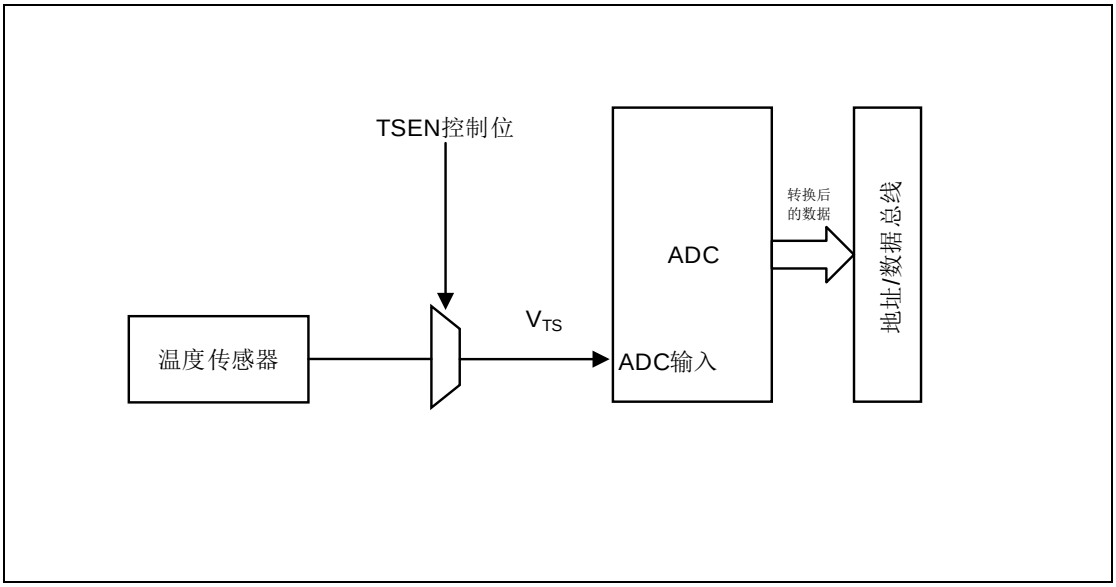
42.5.15. 温度传感器

温度传感器可以用来测量器件的环境温度(TA)。温度传感器内部连接到 ADC3_INP[18]输入通道，该通道把传感器输出的电压转换成数字值。温度传感器模拟引脚的采样时间必须大于器件数据手册中指定的稳定时间。

不使用时可将传感器置于掉电模式。

温度传感器输出电压随温度线性变化，由于生产过程的变化，温度变化曲线的偏移在不同芯片内部温度传感器更适合于检测温度的变化，而不是测量绝对的温度。如果需要测量精确的温度，应该使用一个外置的温度传感器。

图 42-15 温度传感器通道框图



使用温度传感器读温度：

1）选择 ADC3_INP[18]输入通道

2）选择合适的采样时间

3）设置 ADC_CCR 的 TSEN 位，以唤醒关电模式下的温度传感器，需要等待 1.5us，确保温度传感器稳定。

4）通过设置软件触发或外部触发启动 ADC 转换。

5）转换完成后，读 ADC 数据寄存器上的 VTS 数据结果。

6）使用下列公式得出温度

温度(°C) = {(VTS - VOS) / KT_Slope}

这里：VOS = VTEMP 在 0°C 时的数值

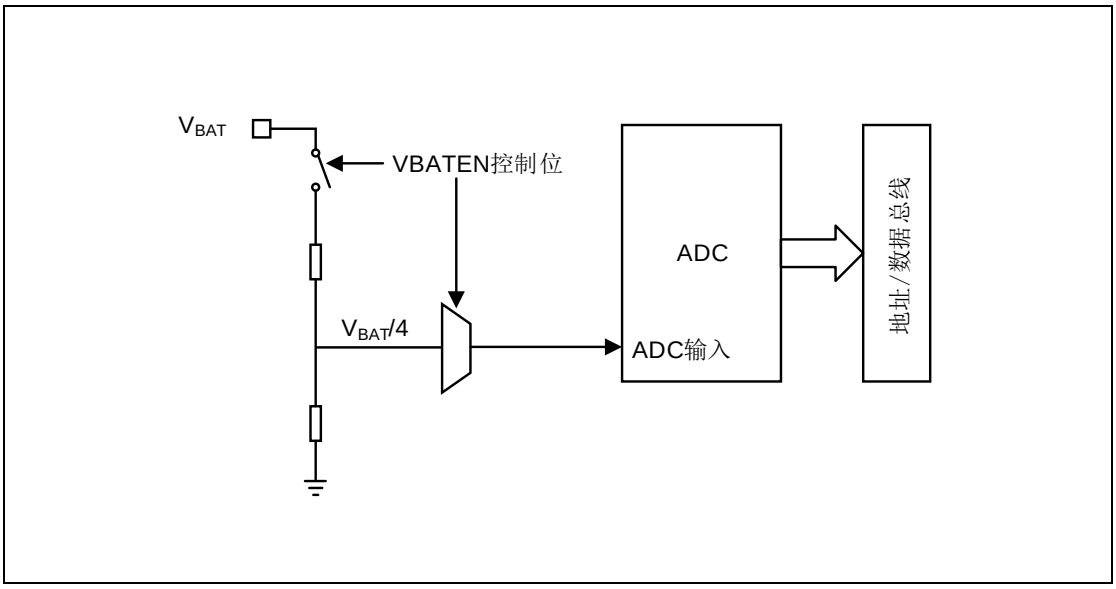
KT_Slope = 温度与 VTS 曲线的平均斜率(单位为 mV/ °C)

42.5.16. VBAT 电池监测

ADC_CCR 寄存器中的 VBATEN 位用于切换到 VBAT 电池监测，由于 VBAT 电压可能高于 VDDA，因此 VBAT 引脚需要从内部连接到桥接分配器（除以 4），以确保 ADC 正确运行。VBATEN 位置 1 时，会自动使能此桥，以将 VBAT/4 连接到 ADC3_INP[17]输入通道。因此，转换出的数字值为 VBAT 电压的四分之一。为防止电池出现意外的电能消耗，建议仅在必须要时为 ADC 转换使能桥接分配器。

转换 VBAT/4 电压时要应用采样时间值，请参见器件数据手册的电气特性部分。

图 42-16 VBAT 通道框图



VBAT 电压可以用一下公式计算：

$$V_{BAT} = V_{REFP} \frac{4 \times N}{4096} (V)$$

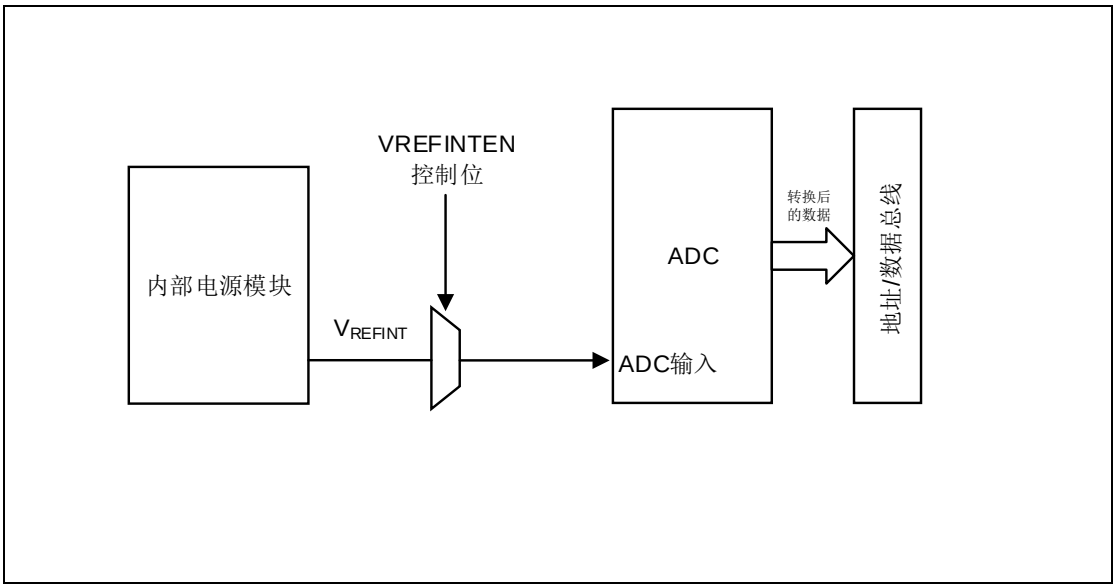
42.5.17. 监测内部参考电压

可以通过监测内部参考电压来获得用于评估 ADC VREF+电压的参考值。

内部参考电压 V_{REFINT} 在内部连接到输入通道 ADC2_INP[13]。

该通道的采样时间必须大于器件数据手册中指定的稳定时间。

图 42-17 VEFINT 通道框图



42.5.18. 单端和差分输入通道

可通过写入 ADC_DIFSEL 寄存器中的 DIFSEL[i]位将通道配置为单端输入或差分输入。必须在禁止 ADC（ADCEN=0）时写入此配置。

在单端输入模式下，通道“i”要转换的模拟电压是等于外部电压 VINP[i]（正输入）与 VREF-（负输入）之差。

在差分输入模式下，通道“i”要转换的模拟电压是等于外部电压 VINP[i]（正输入）与 VINN[i]（负输入）之差。

差分模式的输出数据是无符号数。

差分模式下的输入电压范围从 VREF-到 VREF+，可实现全量程范围为 2×VREF+。当 VINP[i]等于 VREF-，VINN[i]等于 VREF+时，最大的负输入差分电压对应的 ADC 的输出为 0x000。当 VINP[i]等于 VREF+，VINN[i]等于 VREF-时。最大的正输入差分电压对应的 ADC 的输出为 0xFFF。当 VINP[i]和 VINN[i]连接在一起时，零输入差分电压对应的 ADC 输出为 0x800。

$$转换后的值 = \frac{ADC\_Full\_Scale}{2} \times \left[1 + \frac{VINP - VINN}{VREF+}\right]$$

当 ADC 配置为差分模式时，两路输入的偏置电压均应为 VREF+/2。

输入信号应为差分信号（共模电压固定）。

42.5.19. 溢出控制

当 ADC_DR 寄存器的数据没有及时被软件或 DMA 读取时，将产生溢出事件，ADC_SR 寄存器的 OVERF 被置位。当发生溢出事件时，最新采样数据的处理方式有两种，由 ADC_CR2 寄存器的 OVRMOD 为控制，当 OVRMOD=0 时，最新采样数据被丢弃，ADC_DR 寄存器保留上次的转换数据；当 OVRMOD=1 时，最新采样数据保存到 ADC_DR 寄存器。当 OVERF 状态为高时，停止发送 EOC 事件的 DMA 请求，以保证通过 DMA 方式存放到 RAM 中的转换数据是有效的。

42.5.20. 过采样

如果使能了 ADC 的过采样功能，那么每个通道将进行 N 次转换后产生一次 EOC 事件，并对每次的转换结果进行累加，累加器的长度为 20-bit（256 x 12-bit），然后，右移实现除数为 M 的平均，并对右移舍弃的数据做四舍五入处理，截去数据位的高 4 比特，最终保留 16-bit 的有效数据；将结果保存到 ADC_DR 寄存器中。累加平均的表达式如下所示。

$$result = \frac{1}{M} \times \sum_{n=0}^{n=N-1} Conversion(t_n)$$

过采样率 N 由 ADC_CR2 寄存器的 OVSR[2:0]位决定，表示范围从 2 倍到 256 倍。平均系数 M 由 ADC_CR2 寄存器的 OVSS[3:0]决定，最大右移范围 8-bit。

规则组通道的过采样通过 OVSE 位使能，注入组通过 JOVSE 使能。规则组过采样的触发模式由 TROVS 控制。当 TROVS=1 时，一次触发进行一次 ADC 转换，此时忽略 DISCEN 位的内容，并将该位视为 1。当 TROVS=0 时，一次触发进行 N 次 ADC 转换。处理过采样数据时，不可使用对齐模式。ALIGN 位会被忽略，且数据始终采用右对齐格式。过采样模式下不支持偏移校准。如果 ROVSE 和/或 JOVSE 位置 1，ADCx_OFRx 寄存器中 OFFSETx_EN 位的值会被忽略（视为复位）。

对于分辨率不为 12 位的情况，原始数据格式为 12 位左对齐，LSB 为 0。例如分辨率为 6，则原始数据 DATA[11:6]为 000000。过采样时，累加运算低位也始终为 0。

42.5.21. ADC 中断

对于每个 ADC，可在下列情况下产生中断：

- 规则通道转换结束（EOC 标志）
- 规则组转换结束（EOG 标志）
- 注入通道转换结束（JEOC 标志）
- 注入组转换结束（JEOG 标志）
- 发生模拟看门狗检测时（AWD 标志）
- 规则转换数据溢出（OVERF 标志）
- 规则通道采样完成（EOSMP 标志）

可以使用单独的中断使能位控制中断。

注意：ADC1 和 ADC2 的中断映射在同一个中断向量上，ADC_SR 寄存器中有 1 个其他标志，但是它们没有相关联的中断：

- ADRDY（ADC 就绪状态）

表 42-7 中断使能和标志

中断事件	事件标志	使能控制位
规则通道转换结束	EOC	EOCIE
规则组转换结束	EOG	EOGIE
注入通道转换结束	JEOC	JEOCIE
注入组转换结束	JEOG	JEOGIE
模拟看门狗状态位置 1	AWD	AWDIE
规则转换数据溢出	OVERF	OVERFIE
规则通道采样完成	EOSMP	EOSMPIE

42.5.22. ADC 采样率计算

$$f_{SAMPLE} = \frac{f_{ADC}}{T_S + BIT}$$

- fSAMPLE 为 ADC 采样率
- fADC 为 ADC 时钟，ADC_CCR 寄存器中定义了 fADC 相对于 HCLK 的分频数，fADC 最大不能超过 75MHZ
- Ts 为 ADC_SMPR 寄存器中定义的采样周期数
- BIT 为数据分辨率位数，值可以取 12/10/8/6，由 RES[1:0]决定

42.5.23. ADC 外部输入阻抗计算

$$R_{AIN} = \frac{T_S}{f_{ADC} * C_{ADC} * \ln(2^{N+Y})} - R_{ADC}$$

- RAIN 为外部输入阻抗
- Ts 为 ADC_SMPR 寄存器中定义的采样周期数
- fADC 为 ADC 时钟，ADC_CCR 寄存器中定义了 fADC 相对于 HCLK 的分频数，fADC 最大不能超过 75MHZ
- CADC 为内部采样和保持电容，CADC=5pF
- RADC 为内部开关电阻，RADC=120Ω
- N 为分辨率，N=12（12bits 分辨率）
- Y 为精度（采集误差），Y 为正数时，精度为 1/2Y LSB；Y 为负数时，精度为 1*2Y LSB。（$1\ LSB = \frac{V_{refp}}{2^N}$）
  - Y=2，表示 1/4 LSB
  - Y=1，表示 1/2 LSB
  - Y=0，表示 1 LSB
  - Y=-1，表示 2 LSB
  - Y=-2，表示 4 LSB

42.6. 配置流程

42.6.1. 单 ADC 操作流程

1）设置 ADC 输入源：根据需要设置 ADC 输入源，输入 GPIO 设为 GPIO_ANALOG，并打开 GPIO 模拟开关

2）设置 ADC_CCR 寄存器，配置时钟模式 CKMODE，ADC_CLK 的分频（ADCDIV）和独立模式（DUALMOD）

3）设置 ADC_CR1/2 寄存器，配置工作模式，包括：

- 连续模式（CONT）
- 规则/注入转换触发源（EXTSEL/JEXTSEL）
- 选择模拟看门模拟功能(AWDEN/J AWDEN/ AWDSGL/ AWDCH)
- 是否支持 DMA 功能（DMA）
- 中断使能（EOCIE/ JEOCIE/AWDIE）
- 分辨率(RES)
- 数据对齐(ALIGN)
- 过采样(OVSR/ OVSS/ TROVS/ JOVSE)

4）设置 ADC_SQR1/2/3 寄存器，选择规则序列的长度和通道号

5）设置 ADC_DIFSEL 寄存器，选择每个通道差分/单端模式

6）设置 ADC_JSQR 寄存器，选择注入转换的通道号（可选）

7）设置 ADC_SMPR1/2/3 寄存器，配置每个通道的采样时间

8）设置 ADC_HTR/LTR 寄存器，选择模拟看门狗的高低阈值（可选）

9）设置 DMA 通道相关设置（可选）

10）使能 ADC 转换电路(ADC_CR2 寄存器中的 ADC_EN）

11）等待外部 TRIG 或者触发软件 TRIG（ADC_CR1 寄存器中的 SWSTART/ JSWSTART），使能转换序列

12）等待相应的中断信号，处理 ADC 转换的数据（ADC_DR/ADC_JDRx）

42.6.2. 双 ADC 操作流程

1）设置主 ADC1、从 ADC2 输入源：根据需要设置 ADC 输入源，输入 GPIO 设为 GPIO_ANALOG，并打开 GPIO 模拟开关

2）设置 ADC_CCR 寄存器，配置时钟模式 CKMODE，ADC_CLK 的分频（ADCDIV），并配置双 ADC 工作模式，包括：

- 双重 ADC 模式(DUALMOD)
- 双 ADC 模式下 DMA 功能选择(DMADUAL)
- 2 个采样阶段之间的延迟(DELAY)

3）按照“单 ADC 操作流程”中的第 3~9 步操作设置主 ADC1

4）按照“单 ADC 操作流程”中的第 3~9 步操作设置从 ADC2

5）使能 ADC 转换电路（ADC_CR2 寄存器中的 ADC_EN）

6）等待外部 TRIG 或者触发软件 TRIG（ADC_CR1 寄存器中的 SWSTART/ JSWSTART），使能转换序列

7）等待相应的中断信号，处理 ADC 转换的数据（ADC_DR/ADC_JDRx/ADC_CDR）

42.7. ADC 寄存器描述

42.7.1. 寄存器列表

ADC12 寄存器基地址：0x50000000

ADC3 寄存器基地址：0x50000400

各个 ADC 模块地址范围：

偏移	名称
0x000~0x0FC	主 ADC1 或 ADC3
0x100~0x1FC	从 ADC2
0x300~0x3FC	主 ADC 和从 ADC 共用寄存器（ADC12 或 ADC3）

ADC 模块内部地址偏移：

偏移	名称	复位值	描述
0x00	ADC_SR	0x00000000	ADC 状态寄存器
0x04	ADC_IE	0x00000000	ADC 中断使能寄存器
0x08	ADC_CR1	0x00000000	ADC 控制寄存器 1
0x0C	ADC_CR2	0x000001c0	ADC 控制寄存器 2
0x10	ADC_SMPR1	0x00000000	ADC 采样时间寄存器 1
0x14	ADC_SMPR2	0x00000000	ADC 采样时间寄存器 2
0x18	ADC_SMPR3	0x00000000	ADC 采样时间寄存器 3
0x1C	ADC_HTR	0x00000000	ADC 看门狗高阈值寄存器
0x20	ADC_LTR	0x00000000	ADC 看门狗低阈值寄存器
0x24	ADC_SQR1	0x00000000	ADC 规则序列寄存器 1
0x28	ADC_SQR2	0x00000000	ADC 规则序列寄存器 2
0x2C	ADC_SQR3	0x00000000	ADC 规则序列寄存器 3
0x30	ADC_JSQR	0x00000000	ADC 注入序列寄存器
0x34+(x-1)*4	ADC_JDRx	0x00000000	ADC 注入数据寄存器 x（x=1~4）
0x48	ADC_DR	0x00000000	ADC 规则数据寄存器
0x4C	ADC_ DIFSEL	0x00000000	ADC 单端/差分选择寄存器
0x60+(x-1)*4	ADC_OFRx	0x00000000	ADC 偏移寄存器 x（x=1~4）
0x84	ADC_ CALFACT	0x00400040	ADC 校准系数寄存器
0x88+(x-1)*4	ADC_CHDRx	0x00000000	ADC 通道数据寄存器 x（x=1~9）
0x300	ADC_CSR	0x00000000	ADC 共用状态寄存器（ADC12 使用）
0x304	ADC_CCR	0x00010000	ADC 共用控制寄存器（ADC12 和 ADC3 共用） BIT23，BIT21-BIT13，BIT11-BIT8，BIT4-BIT0 可控制 ADC12 BIT24，BIT22-BIT16，BIT13 可控制 ADC3
0x308	ADC_CDR	0x00000000	ADC 共用规则数据寄存器（ADC12 使用）

0x30C	ADC_CVRB	0x00000102	ADC 共用电压参考缓冲器寄存器（ADC3 使用）

42.7.2. ADC 状态寄存器(ADC_SR: 00h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	AFE_EOC	RO	0	ADC 模拟前端转换结束信号（仅供调试用，用户无须关注）
7	AWD	RCW1	0	模拟看门狗标志位 该位由硬件在转换的电压值超出了 ADC_LTR 和 ADC_HTR 寄存器定义的范围时设置，由软件写 1 清除 0：没有发生模拟看门狗事件 1：发生模拟看门狗事件
6	JEOG	RCW1	0	注入通道组转换结束位 该位由硬件在一个注入组序列转换结束时设置，由软件写 1 清除 0：一组转换未完成 1：一组转换完成
5	JEOC	RCW1	0	注入通道转换结束位 该位由硬件在注入通道转换结束时设置，由软件写 1 清除 0：转换未完成 1：转换完成
4	OVERF	RCW1	0	规则通道数据溢出位 该位在规则转换数据溢出时置位，由软件写 1 清除 0：规则转换数据未溢出 1：规则转换数据溢出
3	EOG	RCW1	0	规则通道组转换结束位 该位由硬件在一个规则组序列转换结束时设置，由软件写 1 清除 0：一组转换未完成 1：一组转换完成
2	EOC	RCW1	0	规则通道转换结束位 该位由硬件在规则通道（或者一个规则组序列）转换结束时设置，由软件写 1 清除或由读取 ADC_DR 时清除 0：转换未完成 1：转换完成
1	EOSMP	RCW1	0	规则通道采样完成标志 该位由硬件在规则通道采样结束时设置，由软件写 1 清除 0：采样未完成 1：采样完成
0	ADRDY	RO	0	ADC 就绪状态 该位由硬件置位 0：复位状态 1：就绪状态

42.7.3. ADC 中断使能寄存器(ADC_IE: 04h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	AWDIE	RW	0	允许产生模拟看门狗中断 该位由软件设置和清除，用于禁止或允许模拟看门狗产生中断。在扫描模式下，如果看门狗检测到超范围的数值时，设置了该位时扫描也不会中止 0：禁止模拟看门狗中断 1：允许模拟看门狗中断
6	JEOGIE	RW	0	允许产生 JEOG 中断 该位由软件设置和清除，用于禁止或允许注入通道组转换结束后产生中断。 0：禁止 JEOG 中断 1：允许 JEOG 中断。当硬件设置 JEOG 位时产生中断
5	JEOCIE	RW	0	允许产生注入通道转换结束中断 该位由软件设置和清除，用于禁止或允许注入通道转换结束后产生中断 0：禁止 JEOC 中断 1：允许 JEOC 中断。当硬件设置 JEOC 位时产生中断
4	OVERFIE	RW	0	允许产生规则通道数据溢出中断 该位由软件设置和清除，用于禁止或允许规则通道数据溢出产生中断 0：溢出中断禁止 1：允许产生溢出中断。规则序列上次结果未取走，又有新的转换结果时，产生中断
3	EOGIE	RW	0	允许产生 EOG 中断 该位由软件设置和清除，用于禁止或允许规则组转换结束后产生中断 0：禁止 EOG 中断 1：允许 EOG 中断。当硬件设置 EOG 位时产生中断
2	EOCIE	RW	0	允许产生 EOC 中断 该位由软件设置和清除，用于禁止或允许规则通道转换结束后产生中断 0：禁止 EOC 中断 1：允许 EOC 中断。当硬件设置 EOC 位时产生中断
1	EOSMPIE	RW	0	允许产生 EOSMP 中断 该位由软件设置和清除，用于禁止或允许 EOSMP 中断 0：禁止 EOSMP 中断 1：允许 EOSMP 中断。当硬件设置 EOSMP 位时产生中断
0	RSV	-	-	保留

42.7.4. ADC 控制寄存器 1 (ADC_CR1: 08h)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留

30:28	DISCNUM[2:0]	RW	000	规则通道间断模式通道计数 软件通过这些位定义在间断模式下，收到外部触发后转换规则通道的数目 000：1 个通道 001：2 个通道 … 111：8 个通道 注：需要小于或者等于 ADC_SQR1 中的 L 长度
27	JDISCEN	RW	0	注入通道上的间断模式使能控制 0：注入通道组禁用间断模式 1：注入通道组使用间断模式
26	DISCEN	RW	0	规则通道上的间断模式使能控制 0：规则通道组禁用间断模式 1：规则通道组使用间断模式
25	JAUTO	RW	0	注入组自动转换 该位由软件设置和清除。如果设置了此位，在规则组转换后使能注入组自动转换。 0：禁止注入组自动转换 1：使能注入组自动转换
24	CONT	RW	0	连续转换 该位由软件设置和清除。如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式 1：连续转换模式。
23	SWSTART	RW	0	开始转换规则通道 由软件设置该位用于启动一组规则通道的转换，在单次转换模式下，当 EOG 标志置位时，或者间断模式下，当 EOC 标志置位时，由硬件清除该位。连续转换模式下，由 ADC_CR2 的 ADC_STP 位，清除该位 0：复位状态 1：开始转换规则通道
22	JSWSTART	RW	0	开始转换注入通道 由软件设置该位用于启动一组注入通道的转换，在单次转换模式下，当 JEOG 标志置位时，由硬件清除该位。在间断模式下，当 JEOC 标志置位时，由硬件清除该位。在所有情况下，由 ADC_CR2 的 ADC_STP 位，清除该位。 0：复位状态 1：开始转换注入通道
21	RSV	-	-	保留

20:16	EXTSEL[4:0]	RW	00000	规则通道组转换的外部触发事件选择位 选择用于启动规则通道组转换的外部触发事件 00000： TRIG0 00001： TRIG1 00010： TRIG2 … 10010： TRIG18 10011： TRIG19 10100： TRIG20 其他： TRIG20
15:14	EXTEN[1:0]	RW	00	规则通道的外部触发使能 选择外部触发极性和使能规则组的触发 00：禁止触发检测 01：上升沿上的触发检测 10：下降沿上的触发检测 11：上升沿和下降沿上的触发检测
13	DMA	RW	0	直接存储器访问模式 该位由软件设置和清除。详见 DMA 控制器章节 0：不使用 DMA 模式； 1：使用 DMA 模式。
12	AWDEN	RW	0	在规则通道上开启模拟看门狗 该位由软件设置和清除。 0：在规则通道上禁用模拟看门狗； 1：在规则通道上使用模拟看门狗。
11	JAWDEN	RW	0	在注入通道上开启模拟看门狗 该位由软件设置和清除。 0：在注入通道上禁用模拟看门狗； 1：在注入通道上使用模拟看门狗。
10	AWDSGL	RW	0	在一个单一的通道上使用看门狗 该位由软件设置和清除，用于开启或关闭由 AWDCH/AWDJCH 位指定通道上的模拟看门狗功能 0：在所有通道上使用模拟看门狗 1：在单一通道上使用模拟看门狗
9:5	AWDJCH[4:0]	RW	00000	注入通道模拟看门狗通道选择位 这些位由软件设置和清除，用于选择模拟看门狗保护的注入输入通道 定义参见 AWDCH

4:0	AWDCH[4:0]	RW	00000	模拟看门狗规则通道选择位 这些位由软件设置和清除，用于选择模拟看门狗保护的规则输入通道 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 ... 01111：ADC 模拟输入通道 15 10000：ADC 模拟输入通道 16 10001：ADC 模拟输入通道 17 10010：ADC 模拟输入通道 18

42.7.5. ADC 控制寄存器 2 (ADC_CR2: 0Ch)

位域	名称	属性	复位值	描述
31	ADCCAL	RW	0	ADC 校准 0：校准已完成 1：写入 1 可校准 ADC。读取值为 1 表示正在进行校准。
30	ADCCALDIF	RW	0	校准的输入模式 此位由软件置 1 和清零，用于为校准配置单端输入模式或差分输入模式。 0：将在单端输入模式下启动校准 1：将在差分输入模式下启动校准
29	ADCRSTN	RW	0	ADCAFE 复位信号 清 0 复位，复位 AFE 内部数字部分。在 ADC 稳压器使能后，需要等待 20us 后再拉高。 在强制退出某次转换，需要立刻启动一次新的转换时，可以拉低 ADCRSTN 复位 AFE 数字部分，然后启动新的转换。 注：ADC 模块由模拟模块和数字控制模块构成，其中模拟模块称作模拟前端 Analog Front End（AFE），AFE 内部也包含寄存器、状态机等数字逻辑。
28	ADCVREGEN	RW	0	ADC 稳压器使能 执行使能 ADC 操作之前，必须先使能 ADC 稳压器，并且软件必须等待稳压器启动时间（20us）。 0：禁止 ADC 稳压器 1：使能 ADC 稳压器
27:26	RSV	-	-	保留
25	JOVSE	RW	0	注入组过采样功能使能 0：禁止 1：使能
24	TROVS	RW	0	规则组过采样触发模式 0：一次触发进行 N 次 ADC 转换，N 是由 OVSR 决定的过采样率 1：一次触发进行 1 次 ADC 转换

23:20	OVSS[3:0]	RW	0000	过采样移位系数 0000：不移位 0001：右移 1 位 0010：右移 2 位 0011：右移 3 位 0100：右移 4 位 0101：右移 5 位 0110：右移 6 位 0111：右移 7 位 1000：右移 8 位 其他：保留
19:17	OVSR[2:0]	RW	000	过采样率 000：2x 001：4x 010：8x 011：16x 100：32x 101：64x 110：128x 111：256x
16	OVSE	RW	0	规则组过采样功能使能 0：禁止 1：使能
15:10	RSV	-	-	保留
9:6	TRIMLDO12[3:0]	RW	0111	稳压器 LDO12 电压 TRIM 信号 软件设置，用于 TRIM 稳压器 LDO12。
5:4	RES[1:0]	RW	00	数据分辨率（Data resolution） 通过软件写入这些位可选择转换的分辨率 00：12 位 01：10 位 10：8 位 11：6 位
3	ALIGN	RW	0	数据对齐 软件设置，选择数据对齐模式 0：右对齐 1：左对齐
2	ADC_STP	RW	0	ADC 停止控制 该位由软件设置为 1 时，由硬件等待当前 ADC 转换结束后，由硬件清除该位 0：不执行 ADC 停止转换 1：写 1 用来停止 ADC 转换，读为 1 表明 ADC 转换仍在进行
1	OVRMOD	RW	0	溢出模式 0：发生溢出时 ADC_DR 保留上次采样数据 1：发生溢出时 ADC_DR 保存最新采样数据

0	ADC_EN	RW	0	开/关 A/D 转换器 该位由软件设置和清除。 0：关闭 ADC 1：开启 ADC

42.7.6. ADC 采样时间寄存器 1 (ADC_SMPR1: 10h)

位域	名称	属性	复位值	描述
31:28	SMP7[3:0]	RW	0000	通道 7 采样时间 具体定义见 SMP0
27:24	SMP6[3:0]	RW	0000	通道 6 采样时间 具体定义见 SMP0
23:20	SMP5[3:0]	RW	0000	通道 5 采样时间 具体定义见 SMP0
19:16	SMP4[3:0]	RW	0000	通道 4 采样时间 具体定义见 SMP0
15:12	SMP3[3:0]	RW	0000	通道 3 采样时间 具体定义见 SMP0
11:8	SMP2[3:0]	RW	0000	通道 2 采样时间 具体定义见 SMP0
7:4	SMP1[3:0]	RW	0000	通道 1 采样时间 具体定义见 SMP0
3:0	SMP0[3:0]	RW	0000	通道 0 采样时间 0000：3 周期 0001：5 周期 0010：7 周期 0011：10 周期 0100：13 周期 0101：16 周期 0110：20 周期 0111：30 周期 1000：60 周期 1001：80 周期 1010：100 周期 1011：120 周期 1100：160 周期 1101：260 周期 1110：320 周期 1111：640 周期

42.7.7. ADC 采样时间寄存器 2 (ADC_SMPR2: 14h)

位域	名称	属性	复位值	描述
31:28	SMP15[3:0]	RW	0000	通道 15 采样时间 具体定义见 SMP0
27:24	SMP14[3:0]	RW	0000	通道 14 采样时间 具体定义见 SMP0
23:20	SMP13[3:0]	RW	0000	通道 13 采样时间 具体定义见 SMP0
19:16	SMP12[3:0]	RW	0000	通道 12 采样时间 具体定义见 SMP0
15:12	SMP11[3:0]	RW	0000	通道 11 采样时间 具体定义见 SMP0
11:8	SMP10[3:0]	RW	0000	通道 10 采样时间 具体定义见 SMP0
7:4	SMP9[3:0]	RW	0000	通道 9 采样时间 具体定义见 SMP0
3:0	SMP8[3:0]	RW	0000	通道 8 采样时间 具体定义见 SMP0

42.7.8. ADC 采样时间寄存器 3 (ADC_SMPR3: 18h)

位域	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21:20	SMP_PLUS[1:0]	RW	00	采样时间增加值（不对外开放，调试使用） 00：不增加 01：1 周期 10：2 周期 11：3 周期
19:12	RSV	-	-	保留
11:8	SMP18[3:0]	RW	0000	通道 18 采样时间 具体定义见 SMP0
7:4	SMP17[3:0]	RW	0000	通道 17 采样时间 具体定义见 SMP0
3:0	SMP16[3:0]	RW	0000	通道 16 采样时间 具体定义见 SMP0

42.7.9. ADC 看门狗高阈值寄存器(ADC_HTR: 1Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DHT[11:0]	RW	0x0	差分通道的模拟看门狗高阈值 这些位定义了模拟看门狗的阈值高限，用无符号数表示
15:12	RSV	-	-	保留
11:0	HT[11:0]	RW	0x0	单端通道的模拟看门狗高阈值 这些位定义了模拟看门狗的阈值高限，用无符号数表示

42.7.10. ADC 看门狗低阈值寄存器(ADC_LTR: 20h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DLT[11:0]	RW	0x0	差分通道的模拟看门狗低阈值 这些位定义了模拟看门狗的阈值低限，用无符号数表示
15:12	RSV	-	-	保留
11:0	LT[11:0]	RW	0x0	单端通道的模拟看门狗低阈值 这些位定义了模拟看门狗的阈值低限，用无符号数表示

42.7.11. ADC 规则序列寄存器 1 (ADC_SQR1: 24h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:25	SQ5[4:0]	RW	00000	规则序列中的第 5 个转换
24:20	SQ4[4:0]	RW	00000	规则序列中的第 4 个转换
19:15	SQ3[4:0]	RW	00000	规则序列中的第 3 个转换
14:10	SQ2[4:0]	RW	00000	规则序列中的第 2 个转换
9:5	SQ1[4:0]	RW	00000	规则序列中的第 1 个转换 这些位由软件定义转换序列中的第 1 个转换通道的编号(0~18) 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 ... 01111：ADC 模拟输入通道 15 10000：ADC 模拟输入通道 16 10001：ADC 模拟输入通道 17 10010：ADC 模拟输入通道 18
4	RSV	-	-	保留

3:0	L[3:0]	RW	0000	规则通道序列长度 这些位由软件定义在规则通道转换序列中的通道数目 0000：1 个转换 0001：2 个转换 … 1111：16 个转换

42.7.12. ADC 规则序列寄存器 2 (ADC_SQR2: 28h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:25	SQ11[4:0]	RW	00000	规则序列中的第 11 个转换
24:20	SQ10[4:0]	RW	00000	规则序列中的第 10 个转换
19:15	SQ9[4:0]	RW	00000	规则序列中的第 9 个转换
14:10	SQ8[4:0]	RW	00000	规则序列中的第 8 个转换
9:5	SQ7[4:0]	RW	00000	规则序列中的第 7 个转换
4:0	SQ6[4:0]	RW	00000	规则序列中的第 6 个转换

42.7.13. ADC 规则序列寄存器 3 (ADC_SQR3: 2Ch)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24:20	SQ16[4:0]	RW	00000	规则序列中的第 16 个转换
19:15	SQ15[4:0]	RW	00000	规则序列中的第 15 个转换
14:10	SQ14[4:0]	RW	00000	规则序列中的第 14 个转换
9:5	SQ13[4:0]	RW	00000	规则序列中的第 13 个转换
4:0	SQ12[4:0]	RW	00000	规则序列中的第 12 个转换

42.7.14. ADC 注入通道寄存器(ADC_JSQR: 30h)

位域	名称	属性	复位值	描述

31:27	JEXTSEL[4:0]	RW	00000	注入通道组转换的外部触发事件选择位 选择用于启动注入通道组转换的外部触发事件 00000： JTRIG0 00001： JTRIG1 00010： JTRIG2 … 10010： JTRIG18 10011： JTRIG19 10100： JTRIG20 其他： JTRIG20
26:25	JEXTEN[1:0]	RW	00	注入通道的外部触发使能 选择外部触发极性和使能注入组的触发。 00：禁止触发检测 01：上升沿上的触发检测 10：下降沿上的触发检测 11：上升沿和下降沿上的触发检测
24:20	JSQ4[4:0]	RW	00000	注入序列中的第 4 个转换
19:15	JSQ3[4:0]	RW	00000	注入序列中的第 3 个转换
14:10	JSQ2[4:0]	RW	00000	注入序列中的第 2 个转换
9:5	JSQ1[4:0]	RW	00000	注入序列中的第 1 个转换 这些位由软件定义注入转换通道的编号(0~18)
4:2	RSV	-	-	保留
1:0	JL[1:0]	RW	00	注入序列长度 注入通道转换序列中的转换总数,从 JSQ1 开始 00：1 次转换（JSQ1） 01：2 次转换（JSQ1-JSQ2） 10：3 次转换（JSQ1-JSQ2-JSQ3） 11：4 次转换（JSQ1-JSQ2-JSQ3-JSQ4）

42.7.15. ADC 注入数据寄存器 x (ADC_JDRx: 34h+(x-1)*4, x=1~4)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:16	JCHX[4:0]	RO	00000	注入转换结果通道号 这些位为只读，包含了注入通道的转换结果对应通道号
15:0	JDATAX[15:0]	RO	0x0	注入转换的结果数据 这些位为只读，包含了注入通道的转换结果。数据可以为有符号或者无符号

42.7.16. ADC 规则数据寄存器(ADC_DR: 48h)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留

20:16	CH[4:0]	RO	00000	规则转换结果通道号 这些位为只读，包含了规则通道的转换结果对应通道号
15:0	DATA[15:0]	RO	0x0	规则转换的结果数据 这些位为只读，包含了规则通道的转换结果。数据可以为有符号或者无符号

42.7.17. ADC 差分模式选择寄存器(ADC_DIFSEL: 4Ch)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18:0	DIFSEL[18:0]	RW	0	通道 18 到 0 的差分模式 这些位将由软件置 1 和清零。它们用于选择将通道配置为单端模式或差分模式。 DIFSEL[i] = 0:将 ADC 模拟输入通道配置为单端模式 DIFSEL[i] = 1:将 ADC 模拟输入通道配置为差分模式 注：通道 0 为校准通道。DIFSEL[0]为只读信号。该位反映了 ADCCALDIF 的值。

注：只对外部输入通道有效，内部通道恒定为单端模式。

42.7.18. ADC 偏移寄存器 x (ADC_OFRx: 60h+(x-1)*4, x=1~4)

位域	名称	属性	复位值	描述
31	OFFSETX_EN	RW	0	偏移 x 使能 软件配置是否使能偏移 x 功能 0：偏移 x 功能禁止 1：偏移 x 功能使能
30:26	OFFSETX_CH[4:0]	RW	0	偏移 x 通道选择 软件定义偏移 x 功能对应的通道号
25	OFFSETX_SAT	RW	0	偏移 x 结果格式选择 0：计算结果为有符号数 1：计算结果为无符号数，最小为 0x000,最大为 0xFFF
24	OFFSETX_POS	RW	0	偏移 x 的计算方式 0：转换结果减去 OFFSETx[11:0] 1：转换结果加上 OFFSETx[11:0]
23:12	RSV	-	-	保留
11:0	OFFSETX[11:0]	RW	0	偏移量 x OFFSETx_CH[4:0]选择的通道偏移补偿量，对于规则通道和注入通道都有效。转换结果与 OFFSETx 计算后，放入 ADC_DR 或者 ADC_JDRx 中

42.7.19. ADC 校准系数寄存器(ADC_CALFACT: 84h)

位域	名称	属性	复位值	描述

31:23	RSV	-	-	保留
22:16	CALFACT_D[6:0]	RW	0x40	差分模式下的校准系数 这些位可由硬件或软件写入。差分输入校准完成后，会由硬件立即更新为校准系数。 软件可向这些位写入新的校准系数。如果新的校准系数不同于当前存储于模拟 ADC 中的校准系数，启动新的差分转换后，会立即应用新校准系数。
15:7	RSV	-	-	保留
6:0	CALFACT_S[6:0]	RW	0x40	单端模式下的校准系数 这些位可由硬件或软件写入。单端输入校准完成后，会由硬件立即更新为校准系数。 软件可向这些位写入新的校准系数。如果新的校准系数不同于当前存储于模拟 ADC 中的校准系数，启动新的单端转换后，会立即应用新校准系数。

42.7.20. ADC 通道数据寄存器 x(ADC_CHDRx: 88h+(x-1)*4, x=1~9)

位域	名称	属性	复位值	描述
31:16	CH2X_DR[15:0]	RO	0x0	通道 2x 的转换结果数据
15:0	CH2X-1_DR[15:0]	RO	0x0	通道 2x-1 的转换结果数据

42.7.21. ADC 共用状态寄存器(ADC_CSR: 300h, ADC12 使用)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23	AWD_ SLV	RO	0	从 ADC 模拟看门狗标志位 该位是从 ADC_SR 寄存器中 AWD 的副本
22	JEOG_ SLV	RO	0	从 ADC 注入通道组转换结束位 该位是从 ADC_SR 寄存器中 JEOG 的副本
21	JEOC_ SLV	RO	0	从 ADC 注入通道转换结束位 该位是从 ADC_SR 寄存器中 JEOC 的副本
20	OVERF_ SLV	RO	0	从 ADC 规则通道数据溢出位 该位是从 ADC_SR 寄存器中 OVERF 的副本
19	EOG_ SLV	RO	0	从 ADC 规则通道组转换结束位 该位是从 ADC_SR 寄存器中 EOG 的副本
18	EOC_ SLV	RO	0	从 ADC 规则通道转换结束位 该位是从 ADC_SR 寄存器中 EOC 的副本
17	EOSMP_ SLV	RO	0	从 ADC 规则通道采样完成标志 该位是从 ADC_SR 寄存器中 EOSMP 的副本
16	ADRDY_SLV	RO	0	从 ADC 就绪状态 该位是从 ADC_SR 寄存器中 ADRDY 的副本
15:8	RSV	-	-	保留

7	AWD_MST	RO	0	主 ADC 模拟看门狗标志位 该位是主 ADC_SR 寄存器中 AWD 的副本
6	JEOG_MST	RO	0	主 ADC 注入通道组转换结束位 该位是主 ADC_SR 寄存器中 JEOG 的副本
5	JEOC_MST	RO	0	主 ADC 注入通道转换结束位 该位是主 ADC_SR 寄存器中 JEOC 的副本
4	OVERF_MST	RO	0	主 ADC 规则通道数据溢出位 该位是主 ADC_SR 寄存器中 OVERF 的副本
3	EOG_MST	RO	0	主 ADC 规则通道组转换结束位 该位是主 ADC_SR 寄存器中 EOG 的副本
2	EOC_MST	RO	0	主 ADC 规则通道转换结束位 该位是主 ADC_SR 寄存器中 EOC 的副本
1	EOSMP_MST	RO	0	主 ADC 规则通道采样完成标志 该位是主 ADC_SR 寄存器中 EOSMP 的副本
0	ADRDY_MST	RO	0	主 ADC 就绪状态 该位是主 ADC_SR 寄存器中 ADRDY 的副本

42.7.22. ADC 共用控制寄存器(ADC_CCR: 304h, ADC12 和 ADC3 共用)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	VBATEN	RW	0	VBAT 通道使能信号（ADC3） 0：VBAT 关闭 1：VBAT 使能
23	VREFINTEN	RW	0	VREFINT 使能信号（ADC12） 0：VREFINT 输出三态 1：VREFINT 输出 1.2V 电压
22	TSEN	RW	0	温度传感器使能信号（ADC3） 0：温度传感器断电 1：温度传感器正常工作
21:16	ADCDIV [5:0]	RW	0x1	ADC_CLK 分频选择（ADC12 和 ADC3） 该位由软件设置和清除，决定 ADC_CLK 相对于 PLL3_P_CLK 或者 HCLK 的分频数 注：ADC_CLK 最大不超过 75MHZ 000000：不分频 000001：2 分频 000010：3 分频 … 111110：63 分频 111111：64 分频

15:14	DMADUAL[1:0]	RW	0	双 ADC 下 DMA 功能选择（ADC12） 00：主从 ADC 的各自 DMA 通道独立 01：Reserved 10：主 ADC1 产生 DMA 请求，ADC_CDR 寄存读取数据，分辨率为支持 12 和 10 比特，ADC_CDR 上半个字包含 ADC2 的转换数据，低半个字包含 ADC1 的转换数据 11：主 ADC1 产生 DMA 请求，ADC_CDR 寄存读取数据，分辨率为支持 8 和 6 比特，ADC_CDR 的 15:8 包含 ADC2 的转换数据(SLV_DR[7:0])，7:0 包含 ADC1 的转换数据(MST_DR[7:0])
13	CKMODE	RW	0	ADC 时钟模式（ADC12 和 ADC3） 该位由软件置 1 和清零，用于定义 ADC 时钟方案（主 ADC 和从 ADC 共用）。该位决定 ADC_CLK 的时钟来源： 0：HCLK 1：PLL3_P_CLK
12	RSV	-	-	保留
11:8	DELAY[3:0]	RW	0	2 个采样阶段之间的延迟（ADC12） 这些位在双重交错模式下使用。 0000：5 * TADCCLK 0001：6 * TADCCLK 0010：7 * TADCCLK ... 1111：20 * TADCCLK
7:5	RSV	-	-	保留
4:0	DUALMOD[4:0]	RW	0	双 ADC 模式选择（ADC12） — 所有 ADC 均独立： 00000：独立模式 — 00001 到 01001：双重模式，ADC1 和 ADC2 一起工作 00001：规则同步 + 注入同步混合模式 00010：规则同步 + 交替触发混合模式 00011：规则交叉 + 注入同步混合模式 00101：仅注入同步模式 00110：仅规则同步模式 00111：仅规则交叉模式 01001：仅交替触发模式

42.7.23. ADC 共用规则数据寄存器(ADC_CDR: 308h, ADC12 使用)

位域	名称	属性	复位值	描述
31:16	DATA2[15:0]	RO	0	多重 ADC 规则转换的第二组结果数据。在双重模式下，这些位包含 ADC2 的规则数据 这些位为只读，包含了规则通道的转换结果。数据可以为有符号或者无符号
15:0	DATA1[15:0]	RO	0	多重 ADC 规则转换的第一组结果数据。在双重模式下，这些位包含 ADC1 的规则数据 这些位为只读，包含了规则通道的转换结果。数据可以为有符号或者无符号

42.7.24. ADC 共用电压参考缓冲器寄存器(ADC_CVRB: 30Ch, ADC3 使用)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8:4	VRBTRIM[4:0]	RW	10000	电压参考缓冲器（VREFBUF）电压输出 Trim 值 写入这些位可调整内部参考缓冲器电压
3:2	VRS[1:0]	RW	00	参考电压调节 这些位选择由电压参考缓冲器生成的电压值 00：1.5V 01：1.8V 10：2.0V 11：2.5V
1	HIZ	RW	1	高阻态模式 此位控制模拟开关是否连接到 VREF+引脚 0：VREF+从内部连接到电压参考缓冲器输出 1：VREF+引脚为高阻态
0	ENVR	RW	0	电压参考缓冲器模式使能 0：禁止内部参考电压模式（外部参考电压模式） 1：使能内部参考电压模式（参考缓冲器使能或保持模式） ENVR 与 HIZ 组合构成如下功能(ENVR 为 bit1，HIZ 为 bit0) 00：缓冲器关闭，VREF+引脚下拉到 VSSA 01：缓冲器关闭，可以从外部输入 VREF+ 10：缓冲器开启，VREF+输出参考电压 11：保持模式，缓冲器关闭，通过外部电容保持电压

43. 通用数模转换器（DAC）

43.1. 概述

DAC 可以将 12 位的数字数据转换为外部引脚上的电压输出。数据可以采用 8 位或 12 位模式，在 12 位模式下，数据可以采用左对齐或右对齐模式。当使能了外部触发，DMA 可被用于更新输入端数字数据。DAC 模块有 2 个输出通道，每个通道都有一个单独的转换器。在 DAC 双通道模式下，2 个通道可以独立地进行转换，也可以同时进行转换并同步地更新 2 个通道的输出。

DAC 输出到 PAD 可以断开，内部连接到芯片内其它模拟外设。在输出电压时，可以利用 DAC 输出 BUFFER 来获得更高的驱动能力。DAC 输出支持在低功耗模式下的采样保持模式。

43.2. 主要特性

- 8 位或者 12 位分辨率
- 12 位模式下数据左对齐或右对齐
- 两个 DAC 转换器：各对应 1 个输出通道
- 支持有符号数输入
- 噪声波形生成
- 三角波形生成
- 锯齿波形生成
- DAC 双通道独立或同时转换
- DMA 双数据模式降低总线开销
- 每个通道都有 DMA 功能
- 外部触发转换
- 输出 BUFFER 可选，BUFFER 偏差可校准
- 每个通道可以与 PAD 断开且可以输出到内部互联模块
- STOP 模式支持采样保存功能
- 输入参考电压 VREFP

43.3. 结构框图

下图为 DAC 控制模块的框图

图 43-1 DAC 结构框图

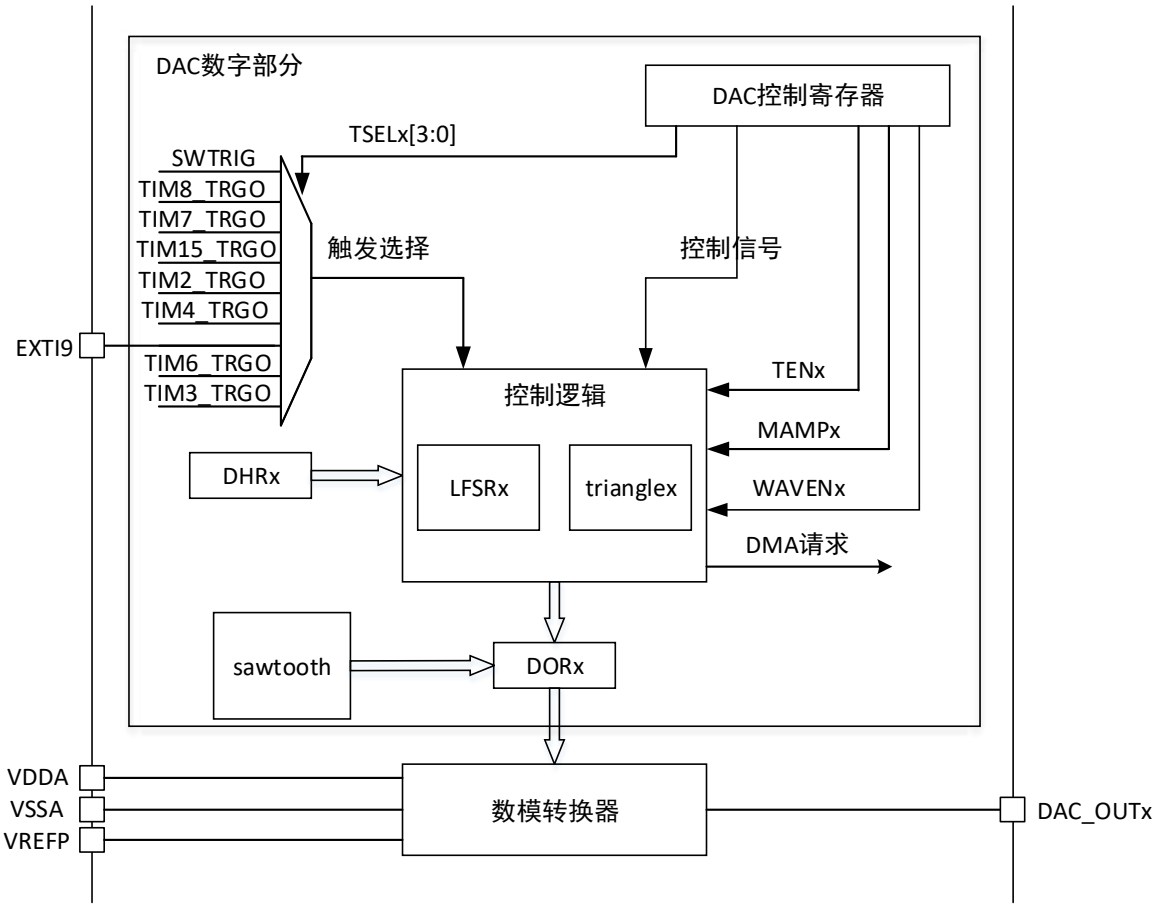


表 43-1 DAC 引脚

名称	信号类型	备注
VREFP	正模拟参考电压输入	DAC 高/正参考电压
VDDA	模拟电源输入	模拟电源
VSSA	模拟电源接地输入	模拟电源接地
DAC_OUTx	模拟输出信号	DAC 通道 x 模拟输出

注意： 使能 DAC 通道 x 后，DAC_OUTx 对应的 GPIO 引脚将自动连接到模拟转换器输出(DAC_OUTx) 。为了避免寄生电流消耗，应首先将 DAC_OUTx 对应的 GPIO 引脚配置为模拟模式。

43.4. 功能描述

43.4.1. DAC 通道使能

将 DAC_CR 寄存器的 ENx 位置 1 即可打开对 DAC 模拟通道 x 的供电。经过 tWAKEUP 启动时间后，DAC 通道 x 即被使能。注意： ENx 位只会使能 DAC 通道 x 的模拟部分，即便该位被置 0，DAC 通道 x 的数字部分仍然工作。

为了降低输出阻抗，并在没有外部运算放大器的情况下驱动外部负载，每个 DAC 模拟通道内部各自集成了一

个输出 BUFFER。可以通过设置 DAC 的寄存器 DAC_MCR 中的相应 MODEx 位开启或者关闭输出 BUFFER，默认为开启状态。

43.4.2. DAC 数据结构

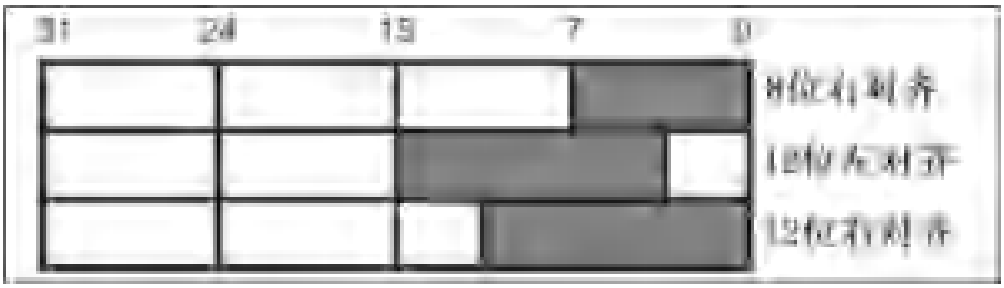
根据选择的配置模式，数据按照下文所述写入指定的寄存器：

■ DAC 单通道 x，有 3 种情况：

- 8 位数据右对齐：用户须将数据写入寄存器 DAC_DHR8Rx[7:0]位(实际是存入寄存器 DAC_DHRx[11:4]位，低位强制为 0)
- 12 位数据左对齐：用户须将数据写入寄存器 DAC_DHR12Lx[15:4]位(实际是存入寄存器 DAC_DHRx[11:0]位)
- 12 位数据右对齐：用户须将数据写入寄存器 DAC_DHR12Rx[11:0]位(实际是存入寄存器 DAC_DHRx[11:0]位)

根据对 DAC_DHRyyyx（其中 yyy 为 8R、12L 或者 12R，x 为 1 或者 2）寄存器的操作，经过相应的移位后，写入的数据被转存到 DAC_DHRx（x 为 1 或者 2）寄存器中(DAC_DHRx 是内部的数据保存寄存器，DAC_DHRyyyx 为访问接口寄存器)。随后 DAC_DHRx 寄存器的内容或被自动地传送到 DAC_DORx 寄存器，或通过软件触发或外部事件触发被传送到 DAC_DORx 寄存器。

图 43-2 DAC 单通道模式的数据寄存器

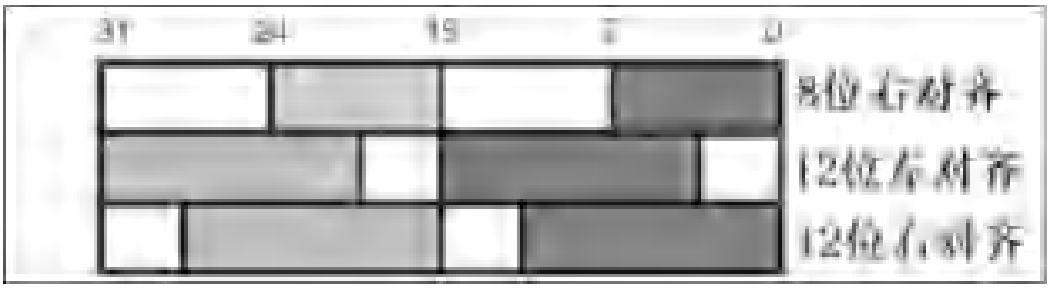


■ DAC 双通道，有 3 种情况：

- 8 位数据右对齐：用户须将 DAC 通道 1 数据写入寄存器 DAC_DHR8RD[7:0]位(实际是存入寄存器 DAC_DHR1[11:4]位，低位强制为 0)，将 DAC 通道 2 数据写入寄存器 DAC_DHR8RD[23:16]位(实际是存入寄存器 DAC_DHR2[11:4]位，低位强制为 0)
- 12 位数据左对齐：用户须将 DAC 通道 1 数据写入寄存 DAC_DHR12LD[15:4]位(实际是存入寄存器 DAC_DHR1[11:0]位)，将 DAC 通道 2 数据写入寄存器 DAC_DHR12LD[31:20]位(实际是存入寄存器 DAC_DHR2[11:0]位)
- 12 位数据右对齐：用户须将 DAC 通道 1 数据写入寄存 DAC_DHR12RD[11:0]位(实际是存入寄存器 DAC_DHR1[11:0]位)，将 DAC 通道 2 数据写入寄存器 DAC_DHR12RD[27:16]位(实际是存入寄存器 DAC_DHR2[11:0]位)

根据对 DAC_DHRyyyD 寄存器的操作，经过相应的移位后，写入的数据被转存到 DAC_DHR1 和 DAC_DHR2 寄存器中(DAC_DHR1 和 DAC_DHR2 是内部的数据保存寄存器 x)。随后，DAC_DHR1 和 DAC_DHR2 的内容或被自动地传送到 DAC_DORx 寄存器，或通过软件触发或外部事件触发被传送到 DAC_DORx 寄存器。

图 43-3 DAC 双通道模式的数据寄存器



■ 有符号/无符号数

DAC 输入支持有符号数或无符号数格式。当输入为无符号数 12bit 模式时，0x000 表示电压最小值， 0xFFF 代表电压最大值。

DAC 输入支持 2s 补码形式的有符号数，通过设置 DAC_MCR 寄存器 SINFORMATx 位选择有符号数。

在有符号数模式，写入 DAC_DHRx 寄存器的数据传送到 DAC_DORx 寄存器时，最高位（MSB）会取反。

DAC_DHR12Lx 寄存器可以用来存放 16 位有符号数。16 位中的最高 12 位用于 DAC 输出数据（DAC_DORx）,其中最高位会被取反。16 位中的最低 4 位会被忽略。

表 43-2 Data format (12 位模式)

SINFORMATx bit	写入 DHRx 寄存器数据	传送到 DORx 寄存器数据
0	0x000	0x000
0	0xFFF	0xFFF
1	0x7FF	0xFFF
1	0x000	0x800
1	0xFFF	0x7FF
1	0x800	0x000

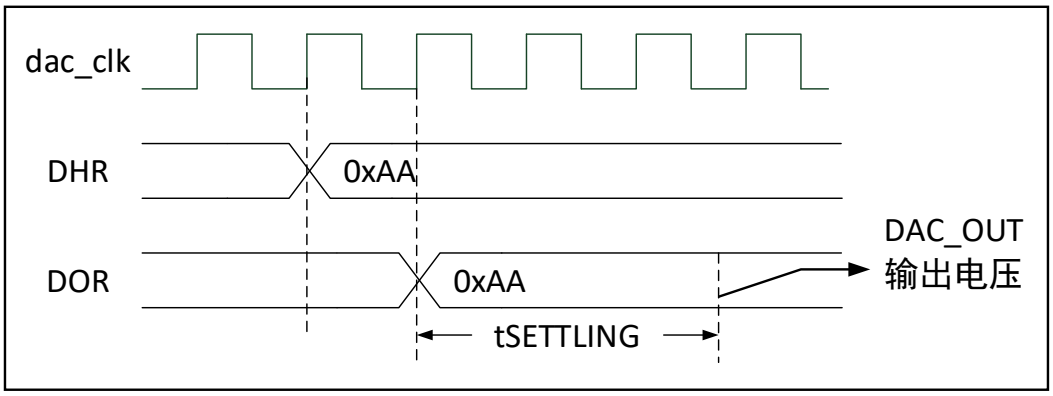
43.4.3. DAC 转换和输出电压

不能直接对寄存器 DAC_DORx 写入数据，任何输出到 DAC 通道 x 的数据都必须写入 DAC_DHRx 寄存器(数据实际写入 DAC_DHR8Rx、 DAC_DHR12Lx、 DAC_DHR12Rx、 DAC_DHR8RD、DAC_DHR12LD、或者 DAC_DHR12RD 寄存器)。

如果没有选中硬件触发(寄存器 DAC_CR1 的 TENx 位置 0)，写入寄存器 DAC_DHRx 的数据会自动加载到寄存器 DAC_DORx。如果选中硬件触发(寄存器 DAC_CR1 的 TENx 位置 1)，写入寄存器 DAC_DHRx 的数据将在触发事件到来时加载到 DAC 数据输出寄存器（DAC_DORx)。

一旦数据从 DAC_DHRx 寄存器装入 DAC_DORx 寄存器，在经过 tSETTLING 时间之后，输出即有效，tSETTLING 时间的长短依电源电压和模拟输出负载的不同会有所变化。

图 43-4 关闭触发（TEN=0）时的转换时序图



数字输入经过 DAC 被线性地转换为模拟电压输出，其范围为 0 到 VREFP。任一 DAC 通道引脚上的输出电压满足下面的关系：

DAC 输出 = VREF * (DOR / 4095)。

43.4.4. DAC 触发源选择

如果 DAC_CR 寄存器的 TENx 位(或 DAC_CR.TENx)被置 1，DAC 的转换将由触发事件进行触发(如定时器计数器、外部管脚信号等，详见表“DAC 触发源选择表”)。配置控制位 TSELx[3:0]可以选择 DAC 转换的触发源。注意 SWTRIG 为软件触发。

表 43-3 DAC 触发源选择表

触发源	类型	TSELx[3:0]
SWTRIG	软件触发位	0000
TIM8_TRGO	内部定时器的触发信号	0001
TIM7_TRGO		0010
TIM15_TRGO		0011
TIM2_TRGO		0100
TIM4_TRGO		0101
EXTI9	外部管脚	0110
TIM6_TRGO	内部定时器的触发信号	0111
TIM3_TRGO		1000

当 DAC 通道检测到 TIMx_TRGO 或者 EXTI9 触发信号时，存放在寄存器 DAC_DHRx 中的数据会被加载到寄存器 DAC_DORx 中。

如果选择软件触发，一旦 SWTRIG 位置 1，转换即开始。在数据从 DAC_DHRx 寄存器传送到 DAC_DORx 寄存器后， SWTRIG 位由硬件自动清 0。

锯齿波生成的复位触发选择和递增触发选择分别通过 STRSTTRIGSELx 和 STINCTRICSELx 实现。其中复位触发选择 STRSTTRIGSELx 与 TSELx 一致，见“DAC 触发源选择表”。递增触发选择 STRINCTRIGSELx 见下表。

表 43-4 递增触发选择

触发源	类型	STINCTRIGSELx[3:0]
SWTRIGB	软件触发位	0000

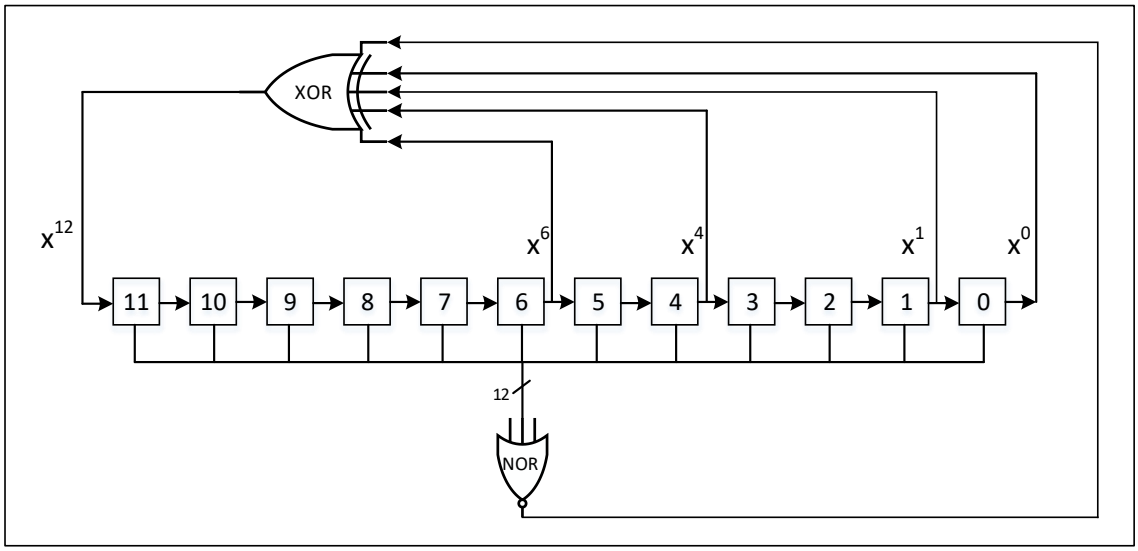
TIM8_TRGO	内部定时器的触发信号	0001
TIM7_TRGO		0010
TIM15_TRGO		0011
TIM2_TRGO		0100
TIM4_TRGO		0101
EXTI10	外部管脚	0110
TIM6_TRGO	内部定时器的触发信号	0111
TIM3_TRGO		1000

43.4.5. DAC 噪声及噪声叠加

DAC 通道有两种方式可以将噪声波加载到 DAC 输出数据： LFSR 噪声波和三角波。噪声波模式可以通过 DAC_CR 寄存器的 WAVEx 位来进行选择。噪声的幅值可以通过配置 DAC_CR 寄存器的 DAC 噪声波位宽（MAMPx）位来进行设置。

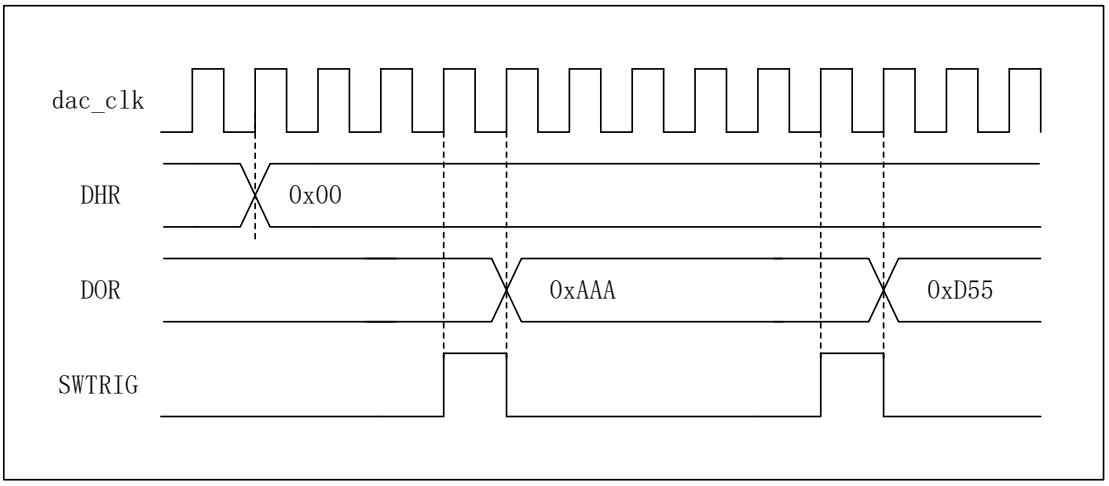
- LFSR 噪声模式：设置 WAVE[1:0]位为 01 选择 LFSR 噪声生成功能。在 DAC 控制逻辑中有一个线性反馈移位寄存器（LFSR）。寄存器 LFSR 的预装入值为 0xAAA。按照特定算法，在每次触发事件后更新该寄存器的值。

图 43-5 LFSR 噪声功能框图



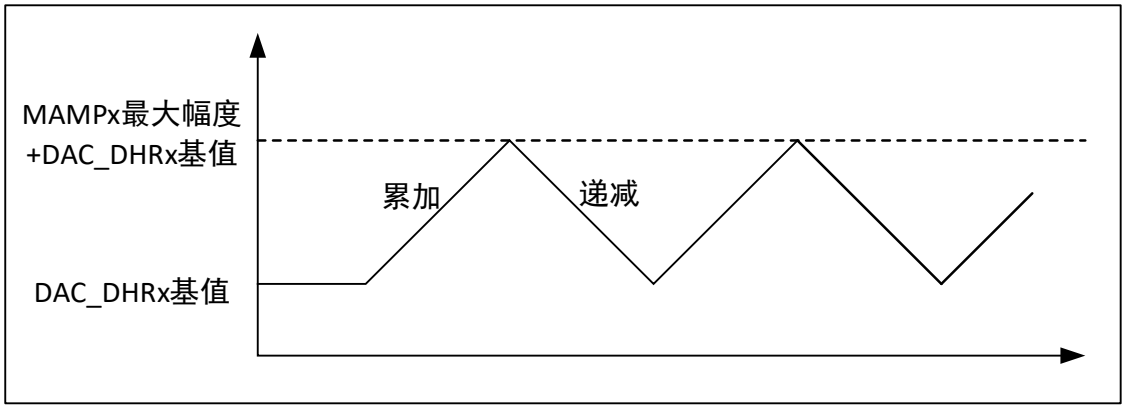
设置 DAC_CR 寄存器的 MAMPx[3:0]位可以屏蔽部分或者全部 LFSR 的数据，这样的得到的 LSFR 值与 DAC_DHRx 的数值相加，去掉溢出位之后即被写入 DAC_DORx 寄存器。MAMPx 设为 0000，只有 LFSR[0]有效,其余 bit 固定为 0；当 MAMPx 大于等于 1011 时，LFSR[11:0]全部有效。将 WAVEx[1:0]位置 0 可以复位 LFSR 波形的生成算法。为了产生噪声，必须使能 DAC 触发，即设 DAC_CR 寄存器的 TENx 位为 1。

图 43-6 带 LFSR 噪声模的 DAC 转换波形



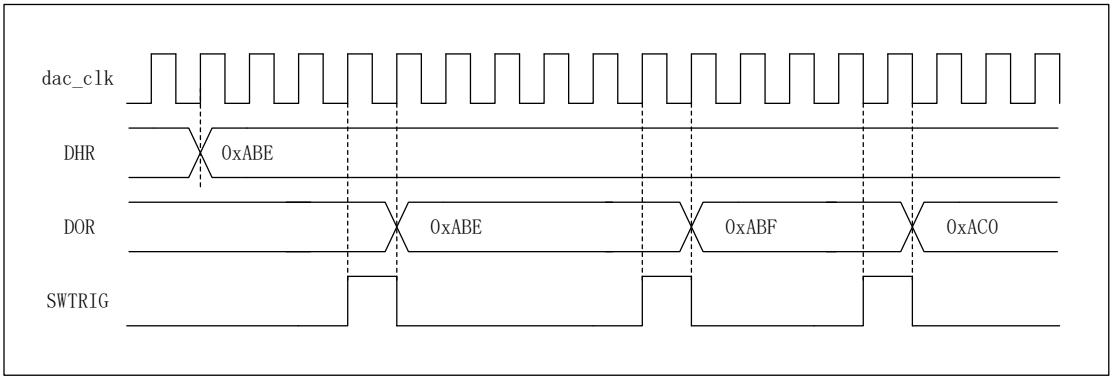
- 三角噪声模式：可以在 DC 或者缓慢变化的信号上加上一个小幅度的三角波噪声。设置 WAVEx[1:0]位为 10 选择 DAC 的三角波生成功能。设置 DAC_CR 寄存器的 MAMPx[3:0]位来选择三角波的幅度。内部的三角波计数器每次触发事件之后累加 1。计数器的值与 DAC_DHRx 寄存器的数值相加并丢弃溢出位后写入 DAC_DORx 寄存器。在传入 DAC_DORx 寄存器的数值小于 MAMP[3:0]位定义的最大幅度时，三角波计数器逐步累加。一旦达到设置的最大幅度，则计数器开始递减，达到 0 后再开始累加，依次循环。

图 43-7 三角波噪声功能框图



将 WAVEx[1:0]位置 0 可以复位三角波的生成。为了产生噪声，必须使能 DAC 触发，即设 DAC_CR 寄存器的 TENx 位为 1。

图 43-8 带三角波噪声的 DAC 转换波形



43.4.6. DAC 锯齿波

DAC 可以产生锯齿波，通过设置锯齿波的初始值、递增值和方向可以通过相关寄存器设置：

1）设置 WAVE[1:0]位为 11 使能锯齿波功能

2）锯齿波计数器（16 位）的初始值（复位值）通过 DAC_STRx 寄存器的 STRSTDATAx[11:0]设置

3）锯齿波计数器的递增值通过 DAC_STRx 寄存器的 STINCDATAx[15:0]设置

4）锯齿波计数器的方向通过 DAC_STRx 寄存器的 STDIRx 位设置

5）锯齿波计数器从 STRSTDATAx[11:0]开始计数（初始值计数器的高 12 位），当递增触发信号有效，计数器按照 STINCDATAx[15:0]的值递增或递减（12.4 位，低 4 位为小数部分）。DAC 转换使用锯齿波计数器的高 12 位。当计数器计数到 0x0000 或者 0xFFFF，计数值不再变化。当复位触发信号有效，计数器回到初始值 STRSTDATAx[11:0]（低 4 位为 0）。

图 43-9 递减锯齿波（STDIR=0）

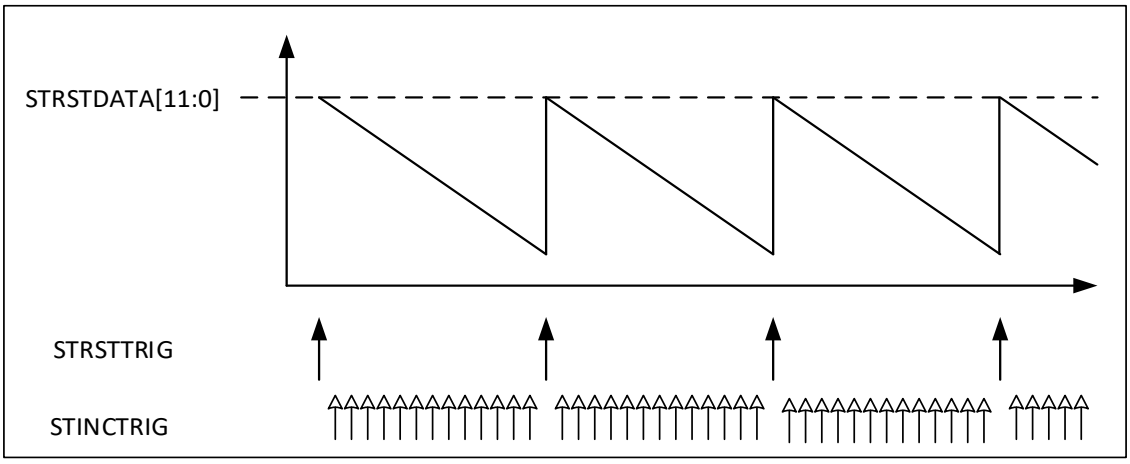
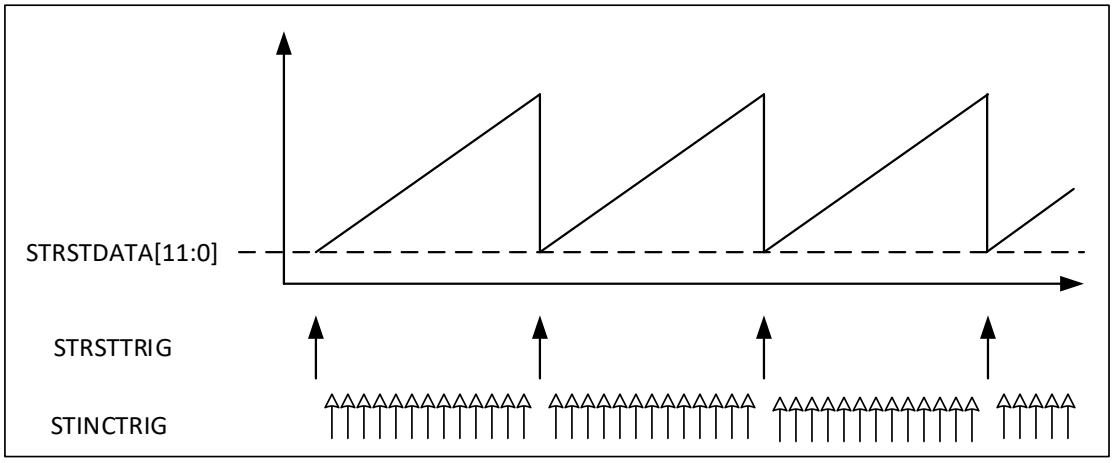


图 43-10 递增锯齿波（STDIR=1）



递增触发信号和复位触发信号通过 STINCTRIGSELx[3:0]和 STRSTTRIGSELx[3:0]选择。参见 DAC 触发选择章节。STRSTTRIG 比 STINCTRIG 有更高的优先级。锯齿波模式触发不受 TENx 控制

43.4.7. 采样保持模式

在采样保持模式下，DAC 完成一次转换后，会将 DAC 的模拟和 BUFFER 关闭，以降低整体功耗，同时将转换电压保持在输出通道的电容上。

在每次转换过程，有一个采样稳定时间（sample）需要等待。在保持阶段电容会慢慢漏电，电压会下降。因此经过一定的保持时间（hold），需要进行刷新充电操作（Refresh）。

在此模式下，sample、hold 和 Refresh 的计数都由 RC32K 完成。

采样保持模式总共分为三个阶段：

1）采样阶段。在转换数据变化后，需要把 DAC 输出充电到目标电压，具体时间取决于电容大小。采样（sample）时间可以通过 DAC_SHSRx 寄存器的 TSAMPLEx[9:0]设置。

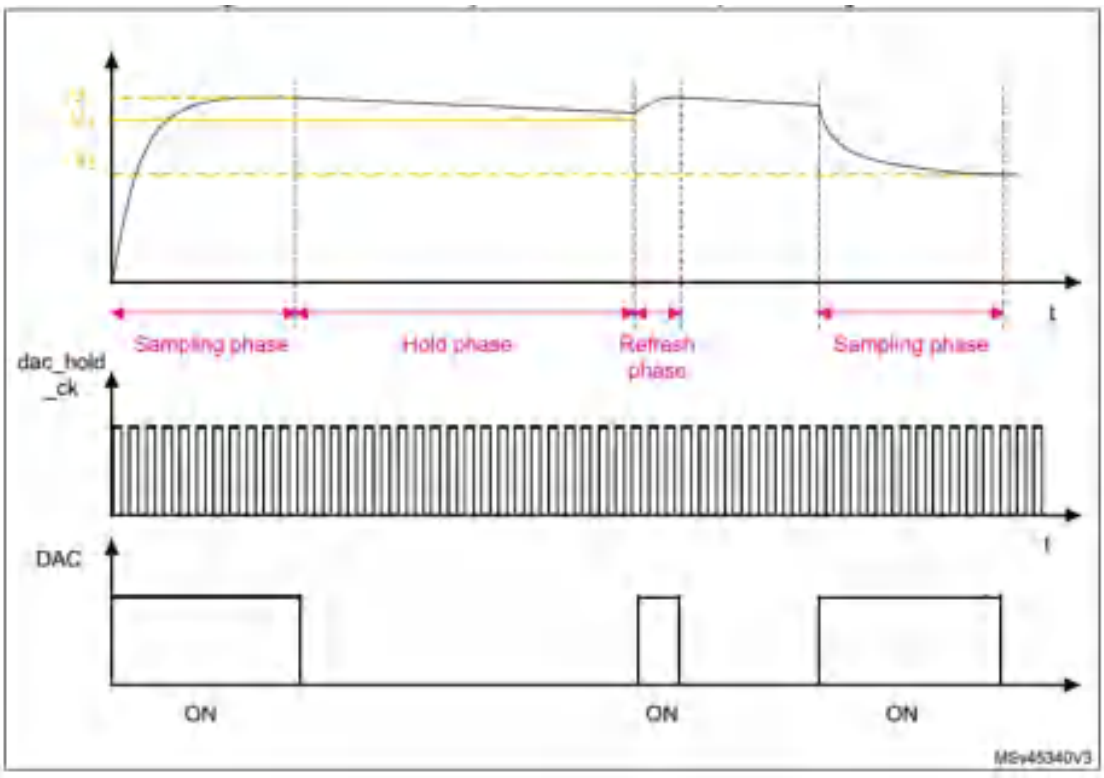
2）保持阶段。DAC 输出处于高阻状态，DAC 模拟部分和 BUFFER 处于关闭状态，用于降低功耗。保持（hold）时间通过 DAC_SHHR 寄存器的 THOLDx[9:0]位设置。

3）刷新（refresh）阶段。对 DAC 输出通道再次充电。刷新（refresh）时间通过 DAC_SHRR 寄存器的 TREFRESHx[7:0]设置。

上述三个阶段时间采用 RC32K 计数。例如 sample 时间为 350us，hold 时间为 2ms，refresh 时间为 100us。则 sample 需要 12 个周期，TSAMPLEx[9:0] = 11；hold 需要 62 个周期，THOLDx[9:0] = 62；refresh 需要 4 个周期，TREFRESHx[7:0] = 4。

这个过程如下图：

图 43-11 采样保持模式时序图



43.4.8. DMA 请求

每个 DAC 通道都具有 DMA 功能。2 个 DMA 通道可分别用于 2 个 DAC 通道的 DMA 请求。如果 DMAENx 位置 1，一旦有外部触发(而不是软件触发)发生，DAC_DHRx 寄存器的数据被传送到 DAC_DORx 寄存器，随后产生一个 DMA 请求。

■ 双通道的 DMA 模式

在双通道模式下，如果 2 个通道的 DMAENx 位都为 1，则会产生 2 个 DMA 请求。如果实际只需要一个 DMA 传输，则应只选择其中一个 DMAENx 位置 1。这样，程序可以在只使用一个 DMA 请求，一个 DMA 通道的情况下，处理工作在双通道模式的 2 个 DAC 通道（双 DAC 通道接口）。

因为数据从 DAC_DHRx 转移到 DAC_DORx 之后才产生 DMA 请求，因此第一笔数据必须在第一次触发信号有效前写入 DAC_DHRx 寄存器。

■ DMA 下溢

DAC 的 DMA 请求没有缓冲队列，因此如果第 2 个外部触发发生在响应第 1 个外部触发的 DMA 请求之前，则不能产生第 2 个 DMA 请求，但会产生 DMA underrun 标志 DMAUDRx（DAC_SR 寄存器）。DAC 通道仍将继续转换旧数据。

软件应通过写入 1 来将 DMAUDRx 标志清零，将所用 DMA 数据流的 DMAEN 位清零，并重新初始化 DMA 和 DAC 通道，以便正确地重新开始 DMA 传输。软件应修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢。最后，可通过使能 DMA 数据传输和转换触发来继续完成 DAC 转换。

对于各 DAC 通道，如果使能 DAC_CR 寄存器中相应的 DMAUDRIEx 位，还将产生中断。

■ 单通道的 DMA 双数据模式

在普通的模式下，一个 DMA 请求只传输 12 位（或 8 位）有效数据。AHB 总线位宽是 32 位的，理论上一次传输可以同时写入两个 12 位数据。设置 DAC_MCR 寄存器的 DMADOUBLEx 位可以使能一次 DMA 请求传输两个 12 位数据模式，即 DMA 双数据模式。

当使能 DMA 双数据模式，每两次外部触发产生一次 DMA 请求。

当检测到第一次外部触发，DAC_DHRx 和 DAC_DHRBx 的数据会传送到 DAC_DORx 和 DAC_DORBx 寄存器。当前有效的 DAC 数据存放在 DAC_DORx 寄存器中。随后，产生一次 DMA 请求。DMA 写入新的数据到 DAC_DHRx 和 DAC_DHRBx 中。

当检查到第二次外部触发，当前有效的 DAC 数据存放到在 DAC_DORBx 中（通过第一次触发从 DAC_DHRBx 传送），不会产生新的 DMA 请求。状态寄存器中的 DORSTATx 位表明当前 DOR 中哪个数据作用于模拟 DAC 转换。第二次触发会把 DORBx 数据搬运到 DORx 寄存器输出；如果设置了噪声叠加，数据从 DORBx 搬运到 DORx 时，会叠加上噪音。

DMA underrun 功能在 DMA 双数据模式同样有效。

DMA 双数据模式，DMA 请求只能用于单 DAC 通道。如果两个通道都需要使能双数据模式，每个 DMA 通道需要单独设置。

在 DMA 单数据模式和双数据模式切换，必须关闭 DAC 模块和 DMA 功能（ENx=0 和 DMAENx=0）。

43.4.9. DAC 双通道转换

当两个 DAC 通道同时工作时，为了在特定应用中最大限度利用总线带宽，两个 DAC 通道的 DAC_DORx 的值将同时被更新。

有 3 个寄存器可被用于加载 DAC_DHRx 的值，分别是：DAC_DHR8RD、DAC_DHR12RD 和 DAC_DHR12LD，只需要访问一个寄存器即可完成同时驱动 2 个 DAC 通道的操作。

当使能了外部触发时，两个 DAC 通道的 TENx 位都应被置位。当需要使能 DMA 功能时，某一个 DAC 通道的 DMAENx 位被置位即可。噪声模式和噪声位宽可以根据使用情况配置为相同或不同。

通过两个 DAC 通道和这三个双寄存器可以实现多种转换模式。但如果需要，所有这些转换模式也都可以通过单独的 DAC_DHRx 寄存器来实现。

下面几段内容将介绍所有这些模式。

43.4.9.1. 独立触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源

3）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD)。

DAC 通道 1 触发信号到达时，DAC_DHR1 寄存器的内容转移到 DAC_DOR1（三个总线时钟周期之后)。

DAC 通道 2 触发信号到达时，DAC_DHR2 寄存器的内容转移到 DAC_DOR2（三个总线周期之后)。

43.4.9.2. 独立触发（生成相同 LFSR)

要将 DAC 配置为此转换模式，需要遵循以下顺序:

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“01”，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD)

DAC 通道 1 触发信号到达时， LFSR1 计数器内容（使用相同的掩码）与 DAC_DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后)。 LFSR1 计数器随即更新。

DAC 通道 2 触发信号到达时， LFSR2 计数器内容（使用相同的掩码）与 DAC_DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后)。 LFSR2 计数器随即更新。

43.4.9.3. 独立触发（生成不同 LFSR)

要将 DAC 配置为此转换模式，需要遵循以下顺序:

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“ 01”，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的 LFSR 掩码值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD)

DAC 通道 1 触发信号到达时， LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后)。 LFSR1 计数器随即更新。

DAC 通道 2 触发信号到达时， LFSR2 计数器内容（使用 MAMP2[3:0] 配置的掩码）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后)。 LFSR2 计数器随即更新。

43.4.9.4. 独立触发（生成相同三角波)

要将 DAC 配置为此转换模式，需要遵循以下顺序:

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“10”，并在 MAMPx[3:0] 位中配置相同的最大振幅值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD)

DAC 通道 1 触发信号到达时， DAC 通道 1 三角波计数器内容（使用相同的三角波振幅）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后)。DAC 通道 1 三角波计数器随即更新。

DAC 通道 2 触发信号到达时， DAC 通道 2 三角波计数器内容（使用相同的三角波振幅）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

43.4.9.5. 独立触发（生成不同三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为不同的值，以配置不同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“10”，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的最大振幅值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

DAC 通道 1 触发信号到达时， DAC 通道 1 三角波计数器内容（使用 MAMP1[3:0] 配置的三角波振幅）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后）。 DAC 通道 1 三角波计数器随即更新。

DAC 通道 2 触发信号到达时， DAC 通道 2 三角波计数器内容（使用 MAMP2[3:0] 配置的三角波振幅）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。 DAC 通道 2 三角波计数器随即更新。

43.4.9.6. 独立触发（生成相同锯齿波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道的 STRSTTRIGSEL1[3:0]、STRSTTRIGSEL2[3:0]、 STINCTRIGSEL2[3:0]和 STINCTRIGSEL1[3:0]设置为不同的值，以配置不同的触发源

2）将两个 DAC 通道的 WAVEx[1:0] 设置为“11”，并在 STRSTDATAx[11:0]、 STINCDATAx[15:0]和 STDIRx 位中配置相同的锯齿波递增值、复位值和方向

DAC 通道 1 触发信号到达时，DAC 通道 1 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR1 中（四个总线时钟周期之后）。

DAC 通道 2 触发信号到达时，DAC 通道 2 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR2 中（四个总线时钟周期之后）。

43.4.9.7. 独立触发（生成不同锯齿波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道的 STRSTTRIGSEL1[3:0]、STRSTTRIGSEL2[3:0]、 STINCTRIGSEL2[3:0]和 STINCTRIGSEL1[3:0]设置为不同的值，以配置不同的触发源

2）将两个 DAC 通道的 WAVEx[1:0] 设置为“11”，并在 STRSTDATAx[11:0]、 STINCDATAx[15:0]和 STDIRx 位中配置不相的锯齿波递增值、复位值和方向

DAC 通道 1 触发信号到达时，DAC 通道 1 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR1 中（四个总线时钟周期之后）。

DAC 通道 2 触发信号到达时，DAC 通道 2 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR2 中（四个总线时钟周期之后）。

43.4.9.8. 同步软件启动

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或

DAC_DHR8RD）

在此配置中， DAC_DHR1 和 DAC_DHR2 寄存器内容会在一个总线时钟周期后分别转移到 DAC_DOR1 和 DAC_DOR2 中。

43.4.9.9. 同步触发（不产生波形）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

当触发信号到达时，DAC_DHR1 和 DAC_DHR2 寄存器内容将分别转移到 DAC_DOR1 和 DAC_DOR2 中（三个总线时钟周期之后）。

43.4.9.10. 同步触发（生成相同 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“01”，并在 MAMPx[3:0] 位中配置相同的 LFSR 掩码值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

触发信号到达时， LFSR1 计数器内容（使用相同的掩码）与 DAC_DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后）。LFSR1 计数器随即更新。同时，LFSR2 计数器内容（使用相同的掩码）与 DAC_DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。 LFSR2 计数器随即更新。

43.4.9.11. 同步触发（生成不同 LFSR）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“ 01”，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的 LFSR 掩码值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

触发信号到达时， LFSR1 计数器内容（使用 MAMP1[3:0] 配置的掩码）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后）。 LFSR1 计数器随即更新。同时， LFSR2 计数器内容（使用 MAMP2[3:0] 配置的掩码）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。 LFSR2 计数器随即更新。

43.4.9.12. 同步触发（生成相同三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“1x”，并在 MAMPx[3:0] 位中配置相同的最大振幅值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

触发信号到达时， DAC 通道 1 三角波计数器内容（使用相同的三角波振幅）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后）。 DAC 通道 1 三角波计数器随即更新。同时，DAC 通道 2 三角波计数器内容（使用相同的三角波振幅）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。DAC 通道 2 三角波计数器随即更新。

43.4.9.13. 同步触发（生成不同三角波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道触发使能位 TEN1 和 TEN2 置 1

2）将 TSEL1[2:0] 和 TSEL2[2:0] 设置为相同的值，以便为两个 DAC 通道配置相同的触发源

3）将两个 DAC 通道的 WAVEx[1:0] 设置为“1x”，并在 MAMP1[3:0] 和 MAMP2[3:0] 位中设置不同的最大振幅值

4）将 DAC 双通道数据加载到所需 DHR 寄存器（ DAC_DHR12RD、 DAC_DHR12LD 或 DAC_DHR8RD）

触发信号到达时，DAC 通道 1 三角波计数器内容（使用 MAMP1[3:0] 配置的三角波振幅）与 DHR1 寄存器内容相加，所得总和转移到 DAC_DOR1 中（三个总线时钟周期之后）。DAC 通道 1 三角波计数器随即更新。

同时，DAC 通道 2 三角波计数器内容（使用 MAMP2[3:0] 配置的三角波振幅）与 DHR2 寄存器内容相加，所得总和转移到 DAC_DOR2 中（三个总线时钟周期之后）。 DAC 通道 2 三角波计数器随即更新。

43.4.9.14. 同步触发（生成相同锯齿波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道的 STRSTTRIGSEL1[3:0]、STRSTTRIGSEL2[3:0]、 STINCTRIGSEL2[3:0]和 STINCTRIGSEL1[3:0]设置为相同的值，以便为两个 DAC 通道配置相同的触发源

2）将两个 DAC 通道的 WAVEx[1:0] 设置为“11”，并在 STRSTDATAx[11:0]、 STINCDATAx[15:0]和 STDIRx 位中配置相同的锯齿波递增值、复位值和方向

DAC 通道 1 触发信号到达时，DAC 通道 1 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR1 中（四个总线时钟周期之后）。

DAC 通道 2 触发信号到达时，DAC 通道 2 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR2 中（四个总线时钟周期之后）。

43.4.9.15. 同步触发（生成不同锯齿波）

要将 DAC 配置为此转换模式，需要遵循以下顺序：

1）将两个 DAC 通道的 STRSTTRIGSEL1[3:0]、STRSTTRIGSEL2[3:0]、 STINCTRIGSEL2[3:0]和 STINCTRIGSEL1[3:0]设置为相同的值，以便为两个 DAC 通道配置相同的触发源

2）将两个 DAC 通道的 WAVEx[1:0] 设置为“11”，并在 STRSTDATAx[11:0]、 STINCDATAx[15:0]和 STDIRx 位中配置不相的锯齿波递增值、复位值和方向

DAC 通道 1 触发信号到达时，DAC 通道 1 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR1 中（四个总线时钟周期之后）。

DAC 通道 2 触发信号到达时，DAC 通道 2 锯齿波计数器会更新，并把计数值加载到到 DAC_DOR2 中（四个总线时钟周期之后）。

43.5. 配置流程

43.5.1. 单个 DAC 操作流程（两个 DAC 独立工作）

设置 DAC 通道的触发使能位 TENx 为 1；

1）通过设置 TSELx[3:0]和 STMODRx，配置 DAC 通道的触发源；

2）设置 SINFORMAT 选择有无符号数

3）根据需要设置 DAC 通道的 WAVEx[1:0]位选择无噪、LFSR 噪声或者三角波噪声或者锯齿波模式，并设 MAMPx[3:0]为不同的 LFSR 屏蔽位或三角波幅值。

4）设置 DAC_STRx 选择锯齿波形（锯齿波模式）

5）设置 DMADOUBLEx 选择是否采用双数据模式

6）通过设置 DMAENx 选择是否使能 DMA 模式

7）设置 ENx 使能对应 DAC 通道

8）通过 DMA 或者软件将 DAC 通道转换数据装入所需的 DHR 寄存器(DHR12Rx、 DHR12Lx 或 DHR8Rx)。

43.5.2. DAC 双通道操作流程

参见 DAC 双通道转换章节

43.6. DAC 寄存器描述

43.6.1. 寄存器列表

DAC1（2 个通道）寄存器基地址：0x50000800

DAC2（2 个通道）寄存器基地址：0x50000C00

偏移	名称	复位值	描述
0x00	DAC_CR	0x00000000	DAC 控制寄存器
0x04	DAC_SWTRIGR	0x00000000	DAC 软件触发寄存器
0x08	DAC_DHR12R1	0x00000000	DAC 通道 1 12 位右对齐数据保持寄存器
0x0C	DAC_DHR12L1	0x00000000	DAC 通道 1 12 位左对齐数据保持寄存器
0x10	DAC_DHR8R1	0x00000000	DAC 通道 1 8 位右对齐数据保持寄存器
0x14	DAC_DHR12R2	0x00000000	DAC 通道 2 12 位右对齐数据保持寄存器
0x18	DAC_DHR12L2	0x00000000	DAC 通道 2 12 位左对齐数据保持寄存器
0x1C	DAC_DHR8R2	0x00000000	DAC 通道 2 8 位右对齐数据保持寄存器
0x20	DAC_DHR12RD	0x00000000	双 DAC 12 位右对齐数据保持寄存器
0x24	DAC_DHR12LD	0x00000000	双 DAC 12 位左对齐数据保持寄存器
0x28	DAC_DHR8RD	0x00000000	双 DAC 8 位右对齐数据保持寄存器
0x2C	DAC_DOR1	0x00000000	DAC 通道 1 数据输出寄存器
0x30	DAC_DOR2	0x00000000	DAC 通道 2 数据输出寄存器
0x34	DAC_SR	0x00000000	DAC 状态寄存器

0x38	DAC_CCR	0x00000000	DAC 校准控制寄存器
0x3C	DAC_MCR	0x00000000	DAC 模式控制寄存器
0x40	DAC_SHSR1	0x00000000	DAC 通道 1 采样时间寄存器
0x44	DAC_SHSR2	0x00000000	DAC 通道 2 采样时间寄存器
0x48	DAC_SHHR	0x00000000	DAC 保持时间寄存器
0x4C	DAC_SHRR	0x00000000	DAC 刷新时间寄存器
0x58	DAC_STR1	0x00000000	DAC 通道 1 锯齿波寄存器
0x5C	DAC_STR2	0x00000000	DAC 通道 2 锯齿波寄存器
0x60	DAC_STMODR	0x00000000	DAC 锯齿波模式寄存器

43.6.2. 控制寄存器(DAC_CR: 00h)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留
30	CEN2	RW	0	DAC 通道 2 calibration 使能
29	DMAUDIE2	RW	0	DAC 通道 2 Underrun 错误中断使能 0: 关闭 DAC 通道 2 Underrun 错误中断 1: 使能 DAC 通道 2 Underrun 错误中断
28	DMAEN2	RW	0	DAC 通道 2 DMA 使能 0：关闭 DAC 通道 2 DMA 模式； 1：使能 DAC 通道 2 DMA 模式
27:24	MAMP2	RW	0	DAC 通道 2 屏蔽/幅值选择器，用来在噪声生成模式下选择屏蔽位，在三角波生成模式下选择波形的幅值。 0000：不屏蔽 LSFR 位[0] / 三角波幅值等于 1； 0001：不屏蔽 LSFR 位[1:0] / 三角波幅值等于 3； 0010：不屏蔽 LSFR 位[2:0] / 三角波幅值等于 7； 0011：不屏蔽 LSFR 位[3:0] / 三角波幅值等于 15； 0100：不屏蔽 LSFR 位[4:0] / 三角波幅值等于 31； 0101：不屏蔽 LSFR 位[5:0] / 三角波幅值等于 63； 0110：不屏蔽 LSFR 位[6:0] / 三角波幅值等于 127； 0111：不屏蔽 LSFR 位[7:0] / 三角波幅值等于 255； 1000：不屏蔽 LSFR 位[8:0] / 三角波幅值等于 511； 1001：不屏蔽 LSFR 位[9:0] / 三角波幅值等于 1023； 1010：不屏蔽 LSFR 位[10:0] / 三角波幅值等于 2047； ≥1011：不屏蔽 LSFR 位[11:0] / 三角波幅值等于 4095
23:22	WAVE2	RW	0	DAC 通道 2 LFSR 噪声/三角波生成使能 00：关闭波形生成； 01：使能 LFSR 噪声波形发生器； 10：使能三角波噪声发生器 11：使能锯齿波发生器

21:18	TSEL2	RW	0	DAC 通道 2 触发选择，该位用于选择 DAC 通道 2 的外部触发事件。 0000： SWTRIG2； 0001： TIM8 TRGO 事件 0010： TIM7 TRGO 事件 0011： TIM15 TRGO 事件 0100： TIM2 TRGO 事件 0101： TIM4 TRGO 事件 0110： EXTI9 0111： TIM6 TRGO 事件 1000： TIM3 TRGO 事件 其它： 备用 注意：该位只能在 TEN2= 1(DAC 通道 2 触发使能)时有效
17	TEN2	RW	0	DAC 通道 2 触发使能 0：关闭 DAC 通道 2 触发，写入寄存器 DAC_DHRx 的数据在 1 个时钟周期后传入寄存器 DAC_DOR2； 1：使能 DAC 通道 2 触发，写入寄存器 DAC_DHRx 的数据在 3 个时钟周期后传入寄存器 DAC_DOR2。 注意：如果选择软件触发，写入寄存器 DAC_DHRx 的数据只需要 1 个时钟周期就可以传入寄存器 DAC_DOR2。 注：锯齿波模式触发不受 TENx 控制
16	EN2	RW	0	DAC 模拟通道 2 使能 0：关闭 DAC 模拟通道 2； 1：使能 DAC 模拟通道 2。 注：需要设置 MODEx 后使能
15	RSV	-	-	保留
14	CEN1	RW	0	DAC 通道 1 calibration 使能
13	DMAUDIE1	RW	0	DAC 通道 1 Underrun 错误中断使能 0: 关闭 DAC 通道 1 Underrun 错误中断 1: 使能 DAC 通道 1 Underrun 错误中断
12	DMAEN1	RW	0	DAC 通道 1 DMA 使能 0：关闭 DAC 通道 1 DMA 模式； 1：使能 DAC 通道 1 DMA 模式

11:8	MAMP1	RW	0	DAC 通道 1 屏蔽/幅值选择器，用来在噪声生成模式下选择屏蔽位，在三角波生成模式下选择波形的幅值。 0000：不屏蔽 LSFR 位[0] / 三角波幅值等于 1； 0001：不屏蔽 LSFR 位[1:0] / 三角波幅值等于 3； 0010：不屏蔽 LSFR 位[2:0] / 三角波幅值等于 7； 0011：不屏蔽 LSFR 位[3:0] / 三角波幅值等于 15； 0100：不屏蔽 LSFR 位[4:0] / 三角波幅值等于 31； 0101：不屏蔽 LSFR 位[5:0] / 三角波幅值等于 63； 0110：不屏蔽 LSFR 位[6:0] / 三角波幅值等于 127； 0111：不屏蔽 LSFR 位[7:0] / 三角波幅值等于 255； 1000：不屏蔽 LSFR 位[8:0] / 三角波幅值等于 511； 1001：不屏蔽 LSFR 位[9:0] / 三角波幅值等于 1023； 1010：不屏蔽 LSFR 位[10:0] / 三角波幅值等于 2047； ≥1011：不屏蔽 LSFR 位[11:0] / 三角波幅值等于 4095
7:6	WAVE1	RW	0	DAC 通道 1 LFSR 噪声/三角波生成使能 00：关闭波形生成； 01：使能 LFSR 噪声波形发生器； 10：使能三角波发生器。 11：使能锯齿波发生器
5:2	TSEL1	RW	0	DAC 通道 1 触发选择，该位用于选择 DAC 通道 1 的外部触发事件。 0000： SWTRIG1； 0001： TIM8 TRGO 事件 0010： TIM7 TRGO 事件 0011： TIM15 TRGO 事件 0100： TIM2 TRGO 事件 0101： TIM4 TRGO 事件 0110： EXTI9 0111： TIM6 TRGO 事件 1000： TIM3 TRGO 事件 其它： 备用 注意：该位只能在 TEN1= 1(DAC 通道 1 触发使能)时有效
1	TEN1	RW	0	DAC 通道 1 触发使能 0：关闭 DAC 通道 1 触发，写入寄存器 DAC_DHRx 的数据在 1 个时钟周期后传入寄存器 DAC_DOR1； 1：使能 DAC 通道 1 触发，写入寄存器 DAC_DHRx 的数据在 3 个时钟周期后传入寄存器 DAC_DOR1。 注意：如果选择软件触发，写入寄存器 DAC_DHRx 的数据只需要 1 个时钟周期就可以传入寄存器 DAC_DOR1。 注：锯齿波模式触发不受 TENx 控制
0	EN1	RW	0	DAC 模拟通道 1 使能 0：关闭 DAC 模拟通道 1； 1：使能 DAC 模拟通道 1。 注：需要设置 MODEx 后使能

43.6.3. 软件触发寄存器(DAC_SWTRIGR: 04h)

位域	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	SWTRIGB2	RW	0	DAC 通道 2 软件触发 B 0：无触发； 1：DAC 通道 2 锯齿波递增软件触发。 注意：软件写 1 触发通道 2 锯齿波递增操作，该位由硬件置 0。
16	SWTRIGB1	RW	0	DAC 通道 1 软件触发 B 0：无触发； 1：DAC 通道 1 锯齿波递增软件触发。 注意：软件写 1 触发通道 1 锯齿波递增操作，该位由硬件置 0。
15:2	RSV	-	-	保留
1	SWTRIG2	RW	0	DAC 通道 2 软件触发 0：关闭 DAC 通道 2 软件触发； 1：使能 DAC 通道 2 软件触发。 注意：一旦寄存器 DAC_DHR2 的数据传入寄存器 DAC_DOR2， (1 个时钟周期后)该位由硬件置 0。
0	SWTRIG1	RW	0	DAC 通道 1 软件触发 0：关闭 DAC 通道 1 软件触发； 1：使能 DAC 通道 1 软件触发。 注意：一旦寄存器 DAC_DHR1 的数据传入寄存器 DAC_DOR1， (1 个时钟周期后)该位由硬件置 0。

43.6.4. 通道 1 12 位右对齐数据保持寄存器(DAC_DHR12R1: 08h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DACC1DHRB[11:0]	RW	0	DAC 通道 1 的 12 位右对齐数据 B，表示 DAC 通道 1 DMA 双数据模式的 12 位数据 B。
15:12	RSV	-	-	保留
11:0	DACC1DHR[11:0]	RW	0	DAC 通道 1 的 12 位右对齐数据，表示 DAC 通道 1 的 12 位数据。

43.6.5. 通道 1 12 位左对齐数据保持寄存器(DAC_DHR12L1: 0Ch)

位域	名称	属性	复位值	描述
31:20	DACC1DHRB[11:0]	RW	0	DAC 通道 1 的 12 位左对齐数据 B，表示 DAC 通道 1 DMA 双数据模式的 12 位数据 B.
19:16	RSV	-	-	保留
15:4	DACC1DHR[11:0]	RW	0	DAC 通道 1 的 12 位左对齐数据，表示 DAC 通道 1 的的 12 位数据.
3:0	RSV	-	-	保留

43.6.6. 通道 1 8 位右对齐数据保持寄存器(DAC_DHR8R1: 10h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DACC1DHRB[11:4]	RW	0	DAC 通道 1 的 8 位右对齐数据 B，表示 DAC 通道 1 DMA 双数据模式的高 8 位数据 B。
7:0	DACC1DHR[11:4]	RW	0	DAC 通道 1 的 8 位右对齐数据，表示 DAC 通道 1 的高 8 位数据。

43.6.7. 通道 2 12 位右对齐数据保持寄存器(DAC_DHR12R2: 14h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DACC2DHRB[11:0]	RW	0	DAC 通道 2 的 12 位右对齐数据 B，表示 DAC 通道 2 DMA 双数据模式的 12 位数据 B。
15:12	RSV	-	-	保留
11:0	DACC2DHR[11:0]	RW	0	DAC 通道 2 的 12 位右对齐数据，表示 DAC 通道 2 的 12 位数据。

43.6.8. 通道 2 12 位左对齐数据保持寄存器(DAC_DHR12L2: 18h)

位域	名称	属性	复位值	描述
31:20	DACC2DHRB[11:0]	RW	0	DAC 通道 2 的 12 位左对齐数据 B，表示 DAC 通道 2 DMA 双数据模式的 12 位数据 B。
19:16	RSV	-	-	保留
15:4	DACC2DHR[11:0]	RW	0	DAC 通道 2 的 12 位左对齐数据，表示 DAC 通道 2 的 12 位数据.
3:0	RSV	-	-	保留

43.6.9. 通道 2 8 位右对齐数据保持寄存器(DAC_DHR8R2: 1Ch)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DACC2DHRB[11:4]	RW	0	DAC 通道 2 的 8 位右对齐数据 B，表示 DAC 通道 2 DMA 双数据模式的高 8 位数据 B。
7:0	DACC2DHR[11:4]	RW	0	DAC 通道 2 的 8 位右对齐数据，表示 DAC 通道 2 的高 8 位数据。

43.6.10. 双 DAC 12 位右对齐数据保持寄存器(DAC_DHR12RD: 20h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DACC2DHR[11:0]	RW	0	DAC 通道 2 的 12 位右对齐数据，表示 DAC 通道 2 的 12 位数据。
15:12	RSV	-	-	保留
11:0	DACC1DHR[11:0]	RW	0	DAC 通道 1 的 12 位右对齐数据，表示 DAC 通道 1 的 12 位数据。

43.6.11. 双 DAC 12 位左对齐数据保持寄存器(DAC_DHR12LD: 24h)

位域	名称	属性	复位值	描述
31:20	DACC2DHR[11:0]	RW	0	DAC 通道 2 的 12 位左对齐数据，表示 DAC 通道 2 的 12 位数据.
19:16	RSV	-	-	保留
15:4	DACC1DHR[11:0]	RW	0	DAC 通道 1 的 12 位左对齐数据，表示 DAC 通道 1 的 12 位数据.
3:0	RSV	-	-	保留

43.6.12. 双 DAC 8 位右对齐数据保持寄存器(DAC_DHR8RD: 28h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:16	DACC2DHR[11:4]	RW	0	DAC 通道 2 的 8 位右对齐数据，表示 DAC 通道 2 的高 8 位数据。
15:8	RSV	-	-	保留
7:0	DACC1DHR[11:4]	RW	0	DAC 通道 1 的 8 位右对齐数据，表示 DAC 通道 1 的高 8 位数据。

43.6.13. 通道 1 数据输出寄存器(DAC_DOR1: 2Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DACC1DORB[11:0]	RO	0	DAC 通道 1 输出数据 B,这些位为只读类型，存储由 DAC 通道 1 转换的数据 B。
15:12	RSV	-	-	保留
11:0	DACC1DOR[11:0]	RO	0	DAC 通道 1 输出数据,这些位为只读类型，存储由 DAC 通道 1 转换的数据。

43.6.14. 通道 2 数据输出寄存器(DAC_DOR2: 30h)

位域	名称	属性	复位值	描述

31:28	RSV	-	-	保留
27:16	DACC2DORB[11:0]	RO	0	DAC 通道 2 输出数据 B,这些位为只读类型，存储由 DAC 通道 2 转换的数据 B。
15:12	RSV	-	-	保留
11:0	DACC2DOR[11:0]	RO	0	DAC 通道 2 输出数据,这些位为只读类型，存储由 DAC 通道 2 转换的数据。

43.6.15. 状态寄存器(DAC_SR: 34h)

位域	名称	属性	复位值	描述
31	RSV	-	-	保留
30	CAL_FLAG2	RO	0	DAC 通道 2 校准标志 0：校准值低于 offset value，校准未完成 1：校准值等于或高于 offset value，校准完成
29	DMAUDR2	RO		DAC 通道 2 DMA underrun 标志 硬件置位，软件写 1 清零 0：DAC 通道 2 未发生 DMA underrun 错误； 1：DAC 通道 2 发生 DMA underrun 错误。
28	DORSTAT2	RO	0	DAC 通道 2 输出数据转换使用标志，在 DMA 双数据模式有效。 0：DOR[11:0]用于当前 DAC 输出转换 1：DORB[11:0]用于当前 DAC 输出转换
27:25	RSV	-	-	保留
24	SAMOV2	RO	0	DAC 通道 2 采样事件同步完成，可以关闭 PCLK（进入 STOP；一次 DAC 转换信号是否同步到 RC32K 标志。 0：同步完成 1：正在同步中，不能关闭 PCLK。
23:15	RSV	-	-	保留
14	CAL_FLAG1	RO	0	DAC 通道 1 校准标志 0：校准值低于 offset value，校准未完成 1：校准值等于或高于 offset value，校准完成
13	DMAUDR1	RO	0	DAC 通道 1 DMA underrun 标志 硬件置位，软件写 1 清零 0：DAC 通道 1 未发生 DMA underrun 错误； 1：DAC 通道 1 发生 DMA underrun 错误。
12	DORSTAT1	RO	0	DAC 通道 1 输出数据转换使用标志，在 DMA 双数据模式有效。 0：DOR[11:0]用于当前 DAC 输出转换 1：DORB[11:0]用于当前 DAC 输出转换
11:9	RSV	-	-	保留
8	SAMOV1	RO	0	DAC 通道 1 采样事件同步完成，可以关闭 PCLK（进入 STOP；一次 DAC 转换信号是否同步到 RC32K 标志。 0：同步完成 1：正在同步中，不能关闭 PCLK。

7:0	RSV	-	-	保留

43.6.16. 校准控制寄存器(DAC_CCR: 38h)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:16	OTRIM2	RW	0	DAC 通道 2 TRIM 值
15:5	RSV	-	-	保留
4:0	OTRIM1	RW	0	DAC 通道 1 TRIM 值

43.6.17. 模式控制寄存器(DAC_MCR: 3Ch)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	SINFORMAT2	RW	0	DAC 通道 2 有符号数选择: 0：输入数据为无符号数模式 1：输入数据为有符号数模式
24	DMADOUBLE2	RW	0	DAC 通道 2 DMA 双数据模式 0：DMA 普通模式 1：DMA 双数据模式
23:19	RSV	-	-	保留
18:16	MODE2	RW	0	DAC 通道 2 MODE 设置，在 DAC_ENx 使能前设置 具体定义参见 MODE1
15:10	RSV	-	-	保留
9	SINFORMAT1	RW	0	DAC 通道 1 有符号数选择: 0：输入数据为无符号数模式 1：输入数据为有符号数模式
8	DMADOUBLE1	RW	0	DAC 通道 1 DMA 双数据模式 0：DMA 普通模式 1：DMA 双数据模式
7:3	RSV	-	-	保留

2:0	MODE1	RW	0	DAC 通道 1 MODE 设置，在 DAC_ENx 使能前设置 正常模式： 000：DAC Buffer 使能，输出到 PAD(GPIO) 001：DAC Buffer 使能，输出到 PAD(GPIO)和内部 OPA 010：DAC Buffer 禁止，输出到 PAD(GPIO) 011：DAC Buffer 禁止，输出到内部 OPA 采样保持模式： 100：DAC buffer 使能，输出到 PAD(GPIO) 101：DAC Buffer 使能，输出到 PAD(GPIO)和内部 OPA 110：DAC Buffer 禁止，输出到 PAD(GPIO)和内部 OPA 111：DAC Buffer 禁止，输出到内部 OPA

43.6.18. 通道 1 采样时间寄存器(DAC_SHSR1: 40h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:0	TSAMPLE1	RW	0	DAC 通道 1 采样时间周期设置 周期数为 TSAMPLE1+1

43.6.19. 通道 2 采样时间寄存器(DAC_SHSR2: 44h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:0	TSAMPLE2	RW	0	DAC 通道 2 采样时间周期设置 周期数为 TSAMPLE2+1

43.6.20. 保持时间寄存器(DAC_SHHR: 48h)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25:16	THOLD2	RW	0	DAC 通道 2 保持时间周期设置 周期数为 THOLD2
15:10	RSV	-	-	保留
9:0	THOLD1	RW	0	DAC 通道 1 保持时间周期设置 周期数为 THOLD1

43.6.21. 刷新时间寄存器寄存器(DAC_SHRR: 4Ch)

位域	名称	属性	复位值	描述

31:24	RSV	-	-	保留
23:16	TREFRESH2	RW	0	DAC 通道 2 刷新时间周期设置 周期数为 TREFRESH2
15:8	RSV	-	-	保留
7:0	TREFRESH1	RW	0	DAC 通道 1 刷新时间周期设置 周期数为 TREFRESH1

43.6.22. 通道 1 锯齿波寄存器(DAC_STR1: 58h)

位域	名称	属性	复位值	描述
31:16	STINCDATA1	RW	0	DAC 通道 1 锯齿波递增值
15:13	RSV	-	-	保留
12	STDIR1	RW	0	DAC 通道 1 锯齿波方向设置 0：递减 1：递增
11:0	STRSTDATA1	RW	0	DAC 通道 1 锯齿波复位值

43.6.23. 通道 2 锯齿波寄存器(DAC_STR2: 5Ch)

位域	名称	属性	复位值	描述
31:16	STINCDATA2	RW	0	DAC 通道 2 锯齿波递增值
15:13	RSV	-	-	保留
12	STDIR2	RW	0	DAC 通道 2 锯齿波方向设置 0：递减 1：递增
11:0	STRSTDATA2	RW	0	DAC 通道 2 锯齿波复位值

43.6.24. 锯齿波模式寄存器(DAC_STMODR: 60h)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:24	STINCTRIGSEL2	RW	0	DAC 通道 2 锯齿波递增触发选择 0000： SWTRIGB2； 0001： TIM8 TRGO 事件 0010： TIM7 TRGO 事件 0011： TIM15 TRGO 事件 0100： TIM2 TRGO 事件 0101： TIM4 TRGO 事件 0110： EXTI10； 0111： TIM6 TRGO 事件 1000： TIM3 TRGO 事件 其它： 备用

23:20	RSV	-	-	保留
19:16	STRSTTRIGSEL2	RW	0	DAC 通道 2 锯齿波复位触发选择 触发选择与 DAC_CR.TSEL2 一致
15:12	RSV	-	-	保留
11:8	STINCTRIGSEL1	RW	0	DAC 通道 1 锯齿波递增触发选择 0000： SWTRIGB1； 0001： TIM8 TRGO 事件 0010： TIM7 TRGO 事件 0011： TIM15 TRGO 事件 0100： TIM2 TRGO 事件 0101： TIM4 TRGO 事件 0110： EXTI-10； 0111： TIM6 TRGO 事件 1000： TIM3 TRGO 事件 其它： 备用
7:4	RSV	-	-	保留
3:0	STRSTTRIGSEL1	RW	0	DAC 通道 1 锯齿波复位触发选择 触发选择与 DAC_CR.TSEL1 一致

44. 多通道数模转换器（MDAC）

44.1. 概述

多通道数字/模拟转换器（DAC）支持 12 个电压 DAC 和 4 个电流 DAC。电压 DAC 可以将 12 位的数字数据转换为外部引脚上的电压输出。电流 DAC 可以将 10 位数字数据转换为外部引脚的电流信号。其中通道 0~11 为电压型 DAC，通道 12~15 为电流型 DAC。

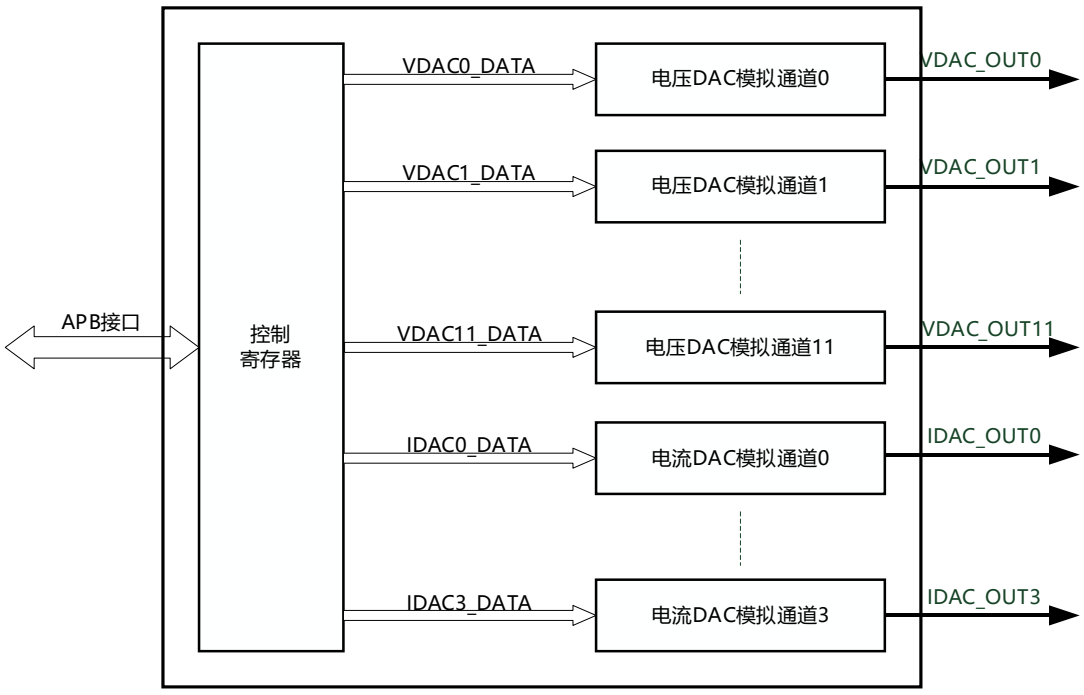
44.2. 主要特性

- 支持 16 个通道 DAC，其中 12 个通道为电压型，4 个通道为电流型
- 每个转换器对应 1 个输出通道，
- 电压型 DAC 支持 12 位分辨率
- 电流型 DAC 支持 10 位分辨率
- 电压型 DAC 支持校准功能
- 电流型 DAC 出厂 Trim
- 电压型输入参考电压 VREF+

44.3. 功能描述

44.3.1. 多通道 DAC 模块框图

图 12 多通道 DAC 模块框图



44.3.2. DAC 通道使能

将 DAC_CRx 寄存器的 EN 位置’ 1’ 即可打开对 DAC 模拟通道 x 的供电。经过一段启动时间 tWAKEUP，DAC 通道 x 即被使能。注意： EN 位只会使能 DAC 通道的模拟部分，即便该位被置’ 0’ ，DAC 通道的数字部分仍然工作。

44.3.3. DAC 转换

直接对寄存器 DACx_DOR 写入数据，数字数据就会送到电压或电流 DAC 模拟部分，转换为电压或者电流模拟信号。多通道 DAC 控制模块没有 DMA 功能，如果需要 DMA 搬运需要配合 Timer 的 DMA 触发功能。

44.4. MDAC 寄存器描述

44.4.1. 寄存器列表

MDAC 寄存器基地址：0x53001C00

偏移	名称	复位值	描述
0x00+(x*4)	VDACx_CR		电压 VDACx 控制寄存器(x=0~11)
0x30+(x*4)	IDACx_CR		电流 IDACx 控制寄存器(x=0~3)
0x40+(x*4)	DACx_DOR		DACx 数据输出寄存器(x=0~15)，其中 0~11 为电压 VDAC，12~15 为电流 IDAC

44.4.2. VDACx 控制寄存器(VDACx_CR: 00h+(x*4), x=0~11)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	CAL_FLAG	RO	0	VDAC 通道 x 校准标志 0：校准值低于 offset value，校准未完成 1：校准值等于或高于 offset value，校准完成
15:11	RSV	-	-	保留
10	SAMPLE_EN	RW	0	VDAC 通道 x 采样保持选择，保持默认值为 0 0：保持模式 1：采样模式
9:5	OTRIM	RW	0	VDAC 通道 x TRIM 值
4	CEN	RW	0	VDAC 通道 x calibration 使能

3:1	MODE	RW	0	VDAC 通道 x MODE 设置，在 EN 使能前设置 正常模式： 000：DAC Buffer 使能，输出到 PAD 010：DAC Buffer 禁止，输出到 PAD 其它模式不支持
0	EN	RW	0	VDAC 模拟通道 x 使能，x 为 0~11 0：关闭 VDAC 模拟通道 x； 1：使能 VDAC 模拟通道 x。 注：需要设置 MODE 后使能

44.4.3. IDACx 控制寄存器(IDACx_CR: 30h+(x*4), x=0~3)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5:1	ITURE	RW	0	IDAC 通道 x 电流修调档位，step 为 2%
0	EN	RW	0	IDAC 模拟通道 x 使能，x 为 0~3 0：关闭 IDAC 模拟通道 x； 1：使能 IDAC 模拟通道 x。

44.4.4. DACx 数据输出寄存器(DACx_DOR: 40h+(x*4), x=0~15)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DACDOR	RW	0	DAC 通道 x 输出数据，存储 DAC 通道 x 转换的数据。 其中 0~11 为电压型 DAC 转换数据，11:0 有效 12~15 为电流型 DAC 转换数据，9:0 有效。

45. 模拟比较器(COMP)

45.1. 概述

芯片内置一个超低功耗模拟比较器（COMP1）。

模拟电压比较器用于比较两个输入模拟电压的大小，并根据比较结果输出高/低电平。当比较器正端输入电压高于负端输入电压时，电压比较器输出高电平；当比较器正端输入电压低于负端输入电压时，电压比较器输出低电平。

比较器集成数字滤波功能，比较结果可输出至 GPIO 端口、定时器或产生中断，或触发芯片从低功耗模式（STOP）唤醒。与定时器的 PWM 输出结合使用时，构成逐周期电流控制环路。

模拟比较器可用于多种功能，包括：

- 在模拟信号的触发下从低功耗模式唤醒
- 模拟信号调理
- 与定时器的 PWM 输出结合使用时，构成逐周期电流控制环路

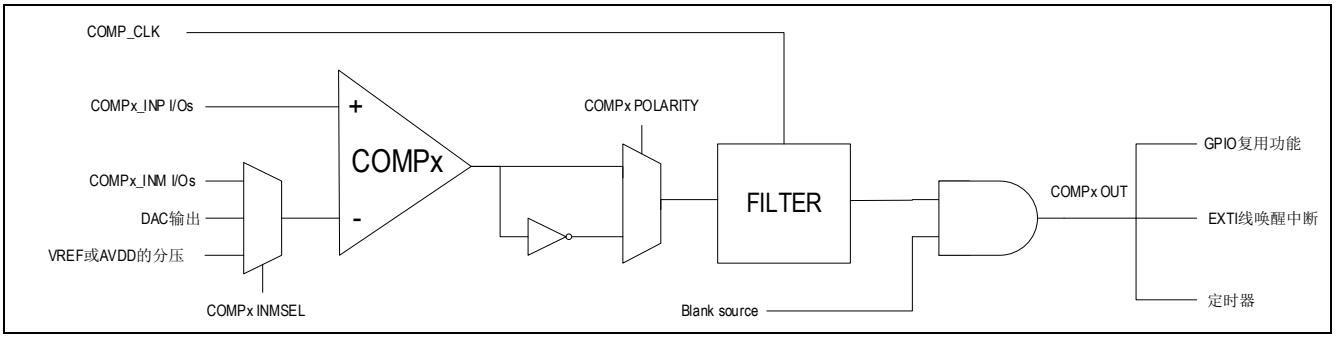
45.2. 主要特性

- 支持电压比较功能
- 输出极性可配置
- 比较器正端输入可配置：
  - 复用 I/O 引脚
- 比较器负端输入可配置：
  - 复用 I/O 引脚
  - DAC 的输出
  - VREF（比较器内部参考电压）或 VDDA 的分压
- 可编程的迟滞窗口
- 支持输出重定向到用于触发以下事件：
  - 刹车事件（用于快速 PWM 关断）
  - 捕获事件
  - Stop 唤醒事件
  - 外部中断事件
- 支持比较器输出到不同 I/O 引脚
- 比较器输出作为定时器的刹车输入或捕获输入
- 支持比较器输出通过定时器消隐
- 每个比较器输出都可作为 EXTI 控制器输入，作为 MCU 唤醒源，支持 Sleep 和 Stop 模式下的唤醒功能
- 支持输出滤波功能，滤波周期可配置
- 比较器的配置可由 COMP_CRx.LOCK 位锁定

45.3. 结构框图

下图为单个比较器结构框图：

图 45-1 单个 COMP 结构框图



45.4. 功能描述

45.4.1. 时钟和复位

COMP 的控制器时钟与 APB2 CLK 同步。在使用比较器之前，要先通过设置 RCC_APB2_IPCKENR.CMPxCKEN 比较器时钟使能位来使能比较器时钟。配置 RCC_APB2_IPRSTR.CMPxRST 比较器复位控制位可进行比较器的软件复位操作。

COMP 的滤波时钟可通过设置 RCC_CCR2.FLTCLK_SEL 选择 PCLK 的 32 分频时钟或 RC32K 时钟。

45.4.2. 正端输入

比较器的正端输入来自 GPIO 引脚。可通过 COMP_CRx.INPSEL 选择不同的 GPIO，且必须在 GPIOx_MD 中配置为模拟功能。

表 45-1 比较器正端输入

比较器	INPSEL[1:0]	GPIO 端口（模拟功能）
COMP1	00	PB0
	01	PB2
	10	保留
	11	保留

45.4.3. 负端输入

比较器负端输入可以选择来自芯片管脚、DAC 输出、或者来自 VREF（内部参考电压）或 VDDA 的分压。通过配置 COMP_CRx.INMSEL 位域选择不同的输入源：

1）负端输入源为 DAC 时，须配置对应的 DAC 工作起来。

2）负端输入源为 GPIO 时，须在 GPIOx_MD 中配置为模拟功能。

3）负端输入源为 VREF 或 VDDA 的分压时，须通过配置 COMP_CRx.CRV_SEL 位域选择分压源为 VDDA 或

VREF，须通过配置 COMP_CRx.CRV_EN 位域选择使能分压，须通过配置 COMP_CRx.CRV_CFG 位域选择分压系数来实现内部输入不同负端电压。

表 45-2 比较器负端输入

比较器	INMSEL[1:0]	GPIO 端口（模拟功能）
COMP1	00	DAC1（须使能 DAC）
	01	PB1（模拟功能）
	10	PC4（模拟功能）
	11	VREF 或 VDDA 分压（须使能分压，并配置分压系数）

45.4.4. 输出极性

通过配置 COMP_CRx.POLARITY 位域，选择将比较器的输出信号直接连接或者取反后连接到 GPIO（COMPx_OUT）、EXTI 总线 和 TIMx 定时器，见单个比较器结构框图。

45.4.5. 输出到 GPIO

比较器输出可映射到不同的 GPIO，须通过 GPIOx.MD 配置为复用功能模式，并在 GPIOx.AF0 或 GPIOx.AF1 中配置复用功能为比较器输出。

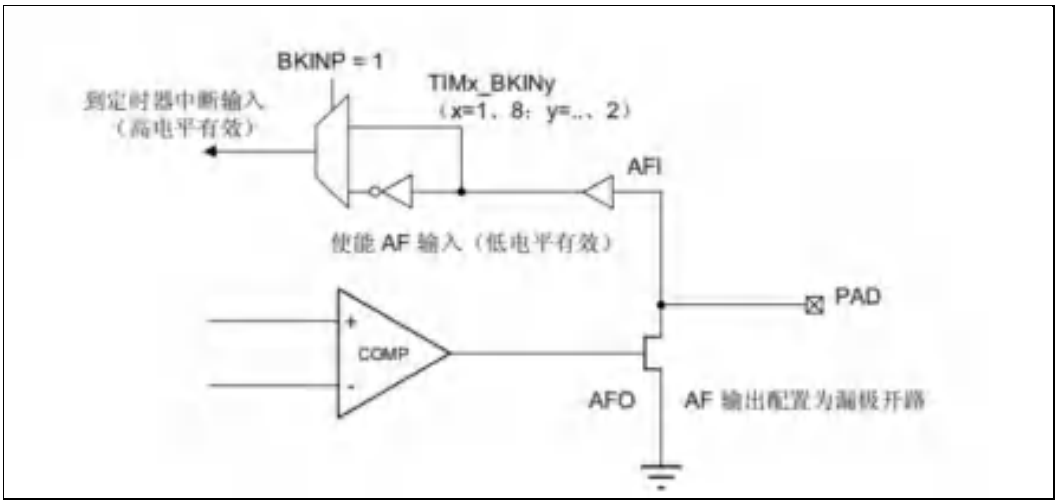
表 45-3 比较器输出

比较器	GPIO 端口	复用功能
COMP1	PA11	AF5
	PB8	AF1
	PC5	AF14
	PE12	AF15
	PF4	AF5

45.4.6. 输出重定向

COMP 通道的输出均可重定向到定时器刹车输入（TIMx_BKIN 或 TIMx_BKIN2），须在定时器中配置寄存器 TIMx_AF1 刹车输入。重定向的定时器请参考“定时器互联关系”章节。

图 45-2 比较器输出重定向



45.4.7. 锁定

对于具有特定功能安全要求的应用，可将 LOCK 置位进行写保护，使 COMP_CRx 变为只读，不能重新配置。

注意：COMP_CRx.LOCK 位域置 1 后无法清除，只有重新复位 MCU 或配置 RCC 控制器中 RCC_APB2_IPRSTR.CMPxRST 复位 COMP 模块才可清除。

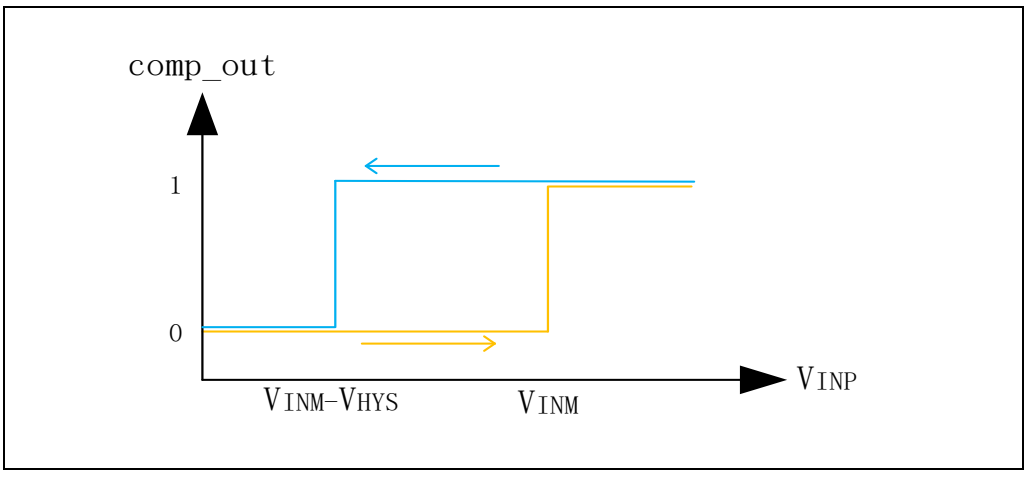
45.4.8. 迟滞比较

迟滞比较功能可以防止在比较电压附近时，输出产生振荡。通过配置 COMP_CRx.HYS 位域开启或更改迟滞窗口的电压范围值，当 COMP_CRx.HYS 位域最高位为 0 时禁止迟滞功能。

迟滞比较器具有很强的抗干扰能力，可以避免在两个输入电压值极为接近的情况下，由于两个输入电压的毛刺导致输出连续翻转的问题。

如图当 INP 高于 INM 时，比较器输出高电平，当 INP 低于 INM 时，比较器输出低电平，在两者之间是保持之前的比较器输出状态。

图 45-3 迟滞比较



45.4.9. 输出滤波

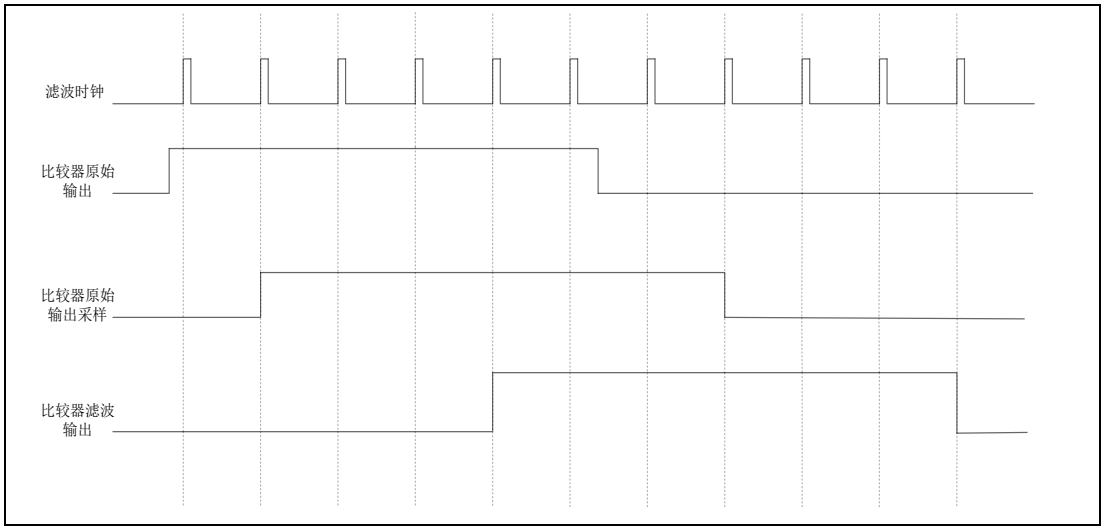
滤波功能可以滤除来自比较器输出端的尖峰毛刺，防止应用电路的误触发。可以通过配置 COMP_CRx.FLTEN 使能滤波功能，配置 COMP_CRx.FLTTIME 可以更改滤除毛刺的最大宽度。

可通过设置 RCC 模块中的 RCC_CCR2.FLTCLKSEL 位来选择比较器滤波时钟源，设置 0 时滤波时钟为 PCLK 的 32 分频，设置 1 时为 RC32K。在比较器作为 EXTI 中断的触发源，用于芯片从 Stop 模式唤醒，且需要使用比

较器滤波功能时，滤波时钟只能选择 RC32K。

图例为 FLTTIME 设置为 0b001 时的比较器滤波输出。

图 45-4 比较器输出滤波

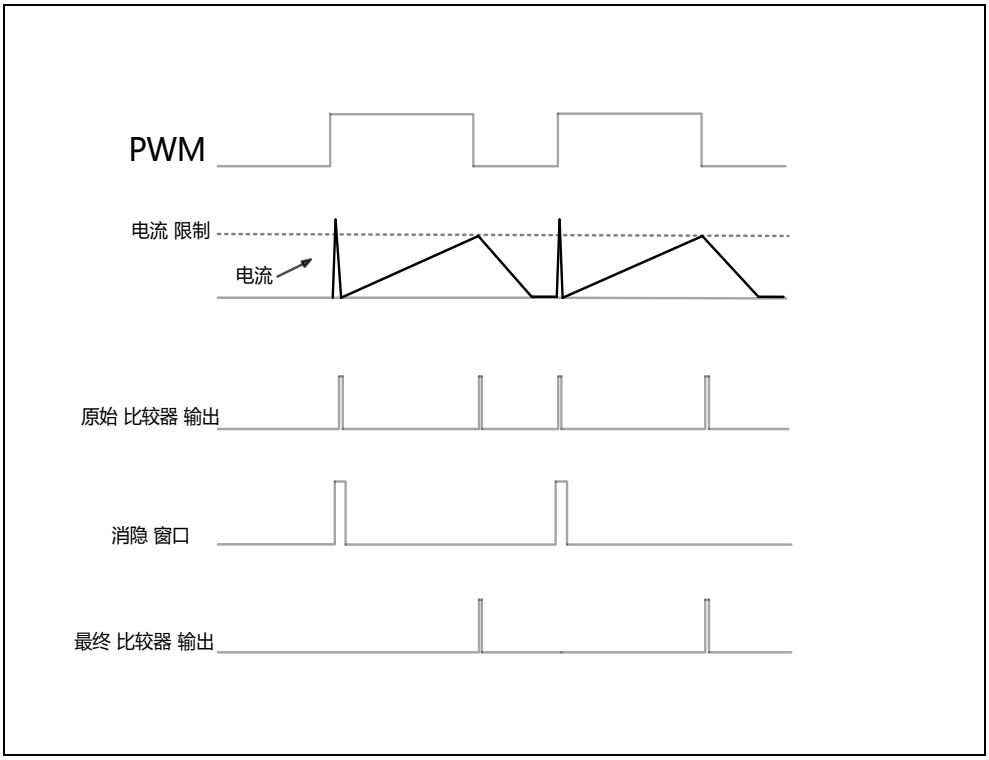


45.4.10. 输出消隐

消隐功能可以通过其他输入来切断比较器输出。通过配置控制寄存器的 COMP_CRx.BLANKSEL 位域开启或更改切断源，通过修改 TIMx 配置更改切断窗口时间宽度。

消隐源由定时器产生，用于消除尖峰脉冲，消隐功能效果如下图所示。

图 45-5 比较器输出消隐



比较器表 45-4 消隐源

比较器	BLANKSEL[2:0]	输入源
COMP1	000	不消隐

	001	TIM1_OC5
	010	TIM1_OC3
	011	TIM3_OC3
	100	TIM3_OC4
	101	TIM8_OC5
	110	TIM15_OC1
	111	全消隐

45.4.11. 中断与唤醒

比较器输出可作为 EXTI 中断的触发源，用于芯片从 Sleep 或 Stop 模式唤醒。

通过 EXTI 模块实现 COMPx 中断的程序：

1）将 EXTI 线（用于接收 comp_wkup 信号）配置为中断模式，选择上升沿、下降沿或任一边沿有效，然后使能 EXTI 线

2）配置并使能映射到相应 EXTI 线的 NVIC IRQ 通道

3）使能 COMPx

45.5. 配置流程

45.5.1. 通用配置流程

1）将比较器对应的 GPIO 口配置成模拟端口

2）配置比较器的正端信号选择（INPSEL）和负端信号选择（INMSEL）

3）设置输出极性 POLARITY

4）设置 CRV_SEL 选择基准分压源（内部基准源可选）

5）设置 CRV_EN 使能基准分压（基准分压可选）

6）设置分压系数 CRV_CFG（基准分压可选）

7）设置 FLTEN 使能滤波功能（滤波可选）

8）设置滤波时间配置（FLTTIME）（滤波可选）

9）设置 HYS 配置迟滞比较窗口值（迟滞比较可选）

10）设置 BLANKSEL 选择消隐源（消隐可选）

11）设置和切断源对应的系统定时器实现触发功能（切断可选）

12）设置 EN 使能比较器

13）设置控制寄存器 COMP_CR 的 LOCK 位锁定寄存器（可选）

14）在 VOUT 端测量或者在 COMP_SR 状态寄存器中读取输出信号

15）如需将输出用作定时器刹车输入功能，请在定时器刹车输入相关寄存器中配置（可选）

16）如需将输出用作 EXTI 功能，请配置 EXTI 的触发源为对应比较器（可选）

45.5.2. DAC 作为负端输入源的配置流程

1）RCC 中配置 RCC_AHB_IPCKENR 使能 DAC 时钟

2）配置 DAC 控制寄存器选择 DAC 功能

3）按照比较器通用配置流程进行配置，注意须将比较器的负端信号选择到 DAC，即 COMP_CRx.INMSEL=0b10

4）使能 DAC 通道

45.5.3. VREF 或 VDDA 分压作为负端输入源的配置流程

1）按照比较器通用配置流程进行配置，注意将比较器的负端信号选择到内部分压，即 COMP_CRx.INMSEL=0b11，再配置 COMP_CRx.CRV_SEL 选择分压源，COMP_CRx.CRV_EN 使能分压

45.5.4. 输出重定向的配置流程

1）RCC 中配置 RCC_APB1_IPCKENR、RCC_APB2_IPCKENR 使能相应 TIMx 时钟

2）配置 TIMx_AF1.BKINSEL 位域选择刹车输入源

3）配置 TIMx_AF1.BKINP 位域选择刹车输入极性控制

4）配置 TIMx_CR1 设置刹车输入的滤波功能

5）配置 TIMx_BDTR.BKP 位域选择刹车输入极性

6）配置 TIMx_AF1.BKINE 位域开启刹车输入使能

7）其他定时器功能，参考 TIMER 模块手册

8）按照比较器通用配置流程进行配置

45.5.5. 输出消隐的配置流程

1）消隐源来自 TIMx，参考《TIMER 用户手册》进行配置

2）按照比较器通用配置流程进行配置，通过配置 COMP_CRx.BLANKSEL 位域选择消隐源

45.5.6. 输出中断的配置流程

1）配置 RCC_APB2_IPCKENR.EXTICKEN 使能 EXTI 模块时钟

2）配置 EXTI 上升沿触发使能寄存器或下降沿触发使能寄存器，选择中断触发方式

3）配置 EXTI 模块寄存器 EXTI_IENR 中 COMP 对应的中断使能位

4）按照比较器通用配置流程进行配置

45.6. COMP 寄存器描述

45.6.1. 寄存器列表

COMP 寄存器基地址：0x40010200

偏移	名称	复位值	描述

0x00	COMP_CR1	0x00000000	COMP1 控制寄存器
0x04	COMP_SR1	0x00000000	COMP1 状态寄存器

45.6.2. COMP1 控制寄存器(COMP_CR1: 00h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	COMP_CR1 寄存器写保护控制 该位由软件设置，通过复位和时钟单元 RCC 复位清除 0：允许软件写入 COMP_CR1 寄存器 1：禁止软件写入 COMP_CR1 寄存器
30:29	RSV	RO	0	保留
28:25	CRV_CFG[3:0]	RW	0	基准电阻分压配置 分压为：(CRV_CFG+1)/20
24	CRV_SEL	RW	0	基准分压来源选择。 0：选择 AVDD 1：选择 VREF
23	CRV_EN	RW	0	基准分压使能信号 0：禁止 1：使能
21:22	RSV	RW	0	保留
20	POLARITY	RW	0	比较器 1 输出极性选择 0：直接输出 1：取反后输出
19	FLTEN	RW	0	比较器 1 滤波使能 0：禁止 1：使能
18:16	FLTTIME[2:0]	RW	0	比较器滤波时间配置，滤波时间按 FILT_CLK 计算，FILT_CLK 由复位和时钟单元 RCC 进行配置 000：1 个周期 001：2 个周期 010：4 个周期 011：16 个周期 100：64 个周期 101：256 个周期 110：1024 个周期 111：4095 个周期
15	RSV	RO	0	保留

14:12	BLANKSEL[2:0]	RW	0	比较器 1 消隐源选择 000：不消隐 001：TIM1_OC5 010：TIM1_OC3 011：TIM3_OC3 100：TIM3_OC4 101：TIM8_OC5 110：TIM15_OC1 111：全消隐
11:8	INPSEL[3:0]	RW	0	比较器 1 正端信号选择 0000：PB0 0001：PB2 其它：保留
7:4	INMSEL[3:0]	RW	0	比较器 1 负端信号选择 0000：DAC1 输出 0001：PB1 0010：PC4 0011：VREF 或 AVDD 的分压 其它：保留
3:1	HYS[2:0]	RO	0	比较器 1 迟滞窗口选择 0xx：禁止迟滞功能 100：10mV 101：20mV 110：30mV 111：40mV
0	EN	RW	0	比较器 1 使能位 0：禁止 1：使能

45.6.3. COMP1 状态寄存器(COMP_SR1: 04h)

位域	名称	属性	复位值	描述
31:5	RSV	RO	0	保留
4	VCOUT1_ORG	RO	0	比较器 1 原始输出状态
3:1	RSV	RO	0	保留
0	VCOUT1	RO	0	比较器 1 滤波输出状态

46. 电容触摸感应（TKEY）

46.1. 概述

TKEY 通过检测电容的变化来检测手指或触及触摸表面。

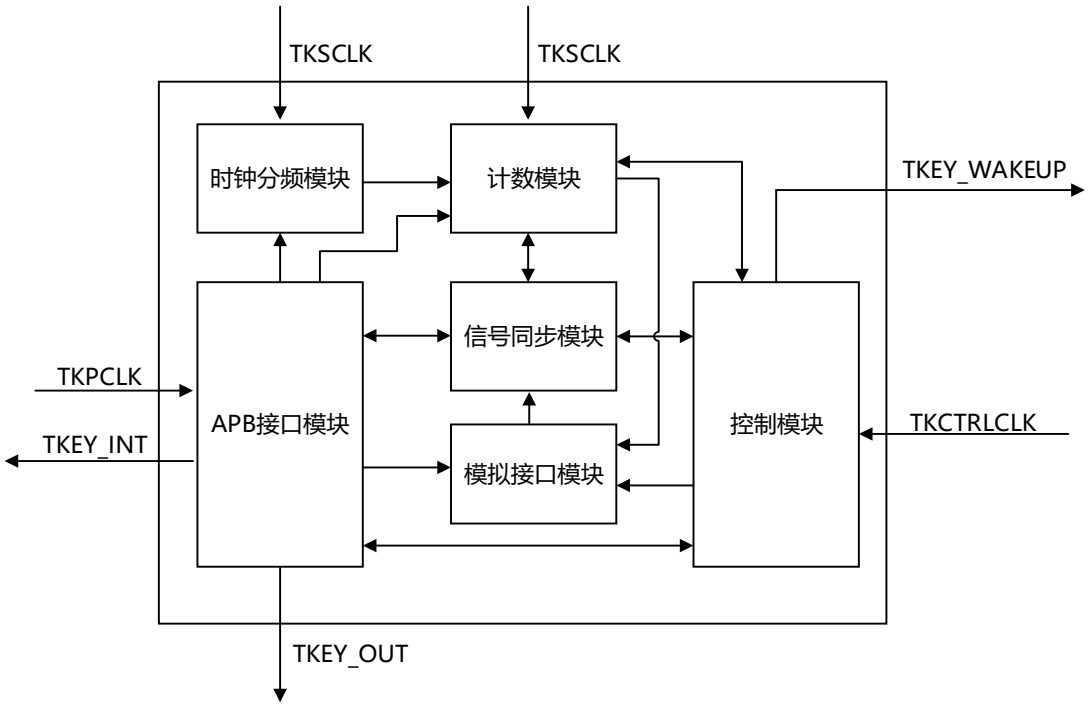
TKEY 模块提供了两种电容式触摸感应方法，分别为 CSD(Capsense Sigma-Delta)和 CSA(Capsense Successive-Aproximation)。在 CSA 模式下，可以选择 ADC 测量电容电量的方式来检测按键触摸。

46.2. 主要特性

- 支持 APB 总线
- 16 个电容检测通道
- 支持电阻和电容补偿
- 支持两种触摸感应方法
- CSD 模式无需外部电容
- CSD 模式支持电阻和电流源放电
- CSD 模式支持内部电容预充
- 触摸精度可调
- 支持扫描时钟扩频
- 支持屏蔽通道
- 支持自动模式，硬件判断按键触摸
- CSA 模式下支持比较器滤波
- CSA 模式下充电次数可设置
- CSA 模式下电容放电前的等待时间可设置

46.3. 结构框图

图 46-1 TKEY 结构框图



TKPCLK 为 TKEY APB 接口模块的时钟，时钟源为 PCLK2。

TKSCLK 为时钟分频模块和计数模块的模块时钟，时钟源是 PLL3QCLK。

TKCTRLCLK 为控制模块的模块时钟，时钟源是 RCL。

TKEY_INT 为 TKEY 中断信号

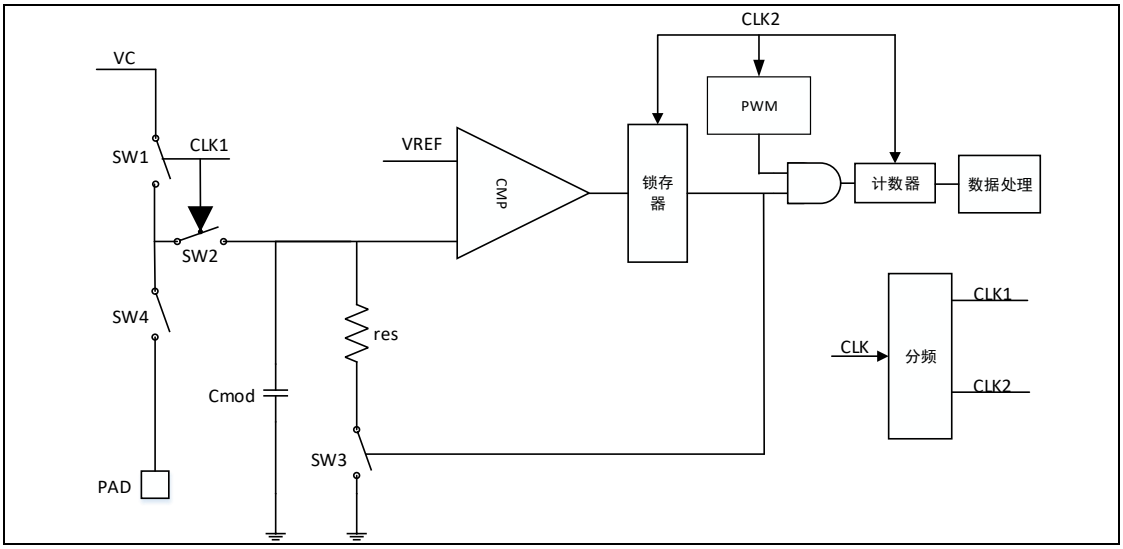
TKEY_OUT 为 ADC 的触发信号

TKEY_WAKEUP 为 TKEY 的唤醒信号

46.4. 电路结构

46.4.1. CSD 实现方案

图 46-2 CSD 实现方案



1）断开 SW2，闭合 SW1 和 SW4，VC 对 PAD 充电。

2）断开 SW1，闭合 SW2 和 SW4，PAD 再对内部 Cmod 进行电荷转移。

3）重复 1-2。

4）比较器检测 Cmod 上的电压，当 Cmod 上的电压升到超过比较器 VREF 时，比较器发生翻转，闭合 SW3，通过电阻来放电。

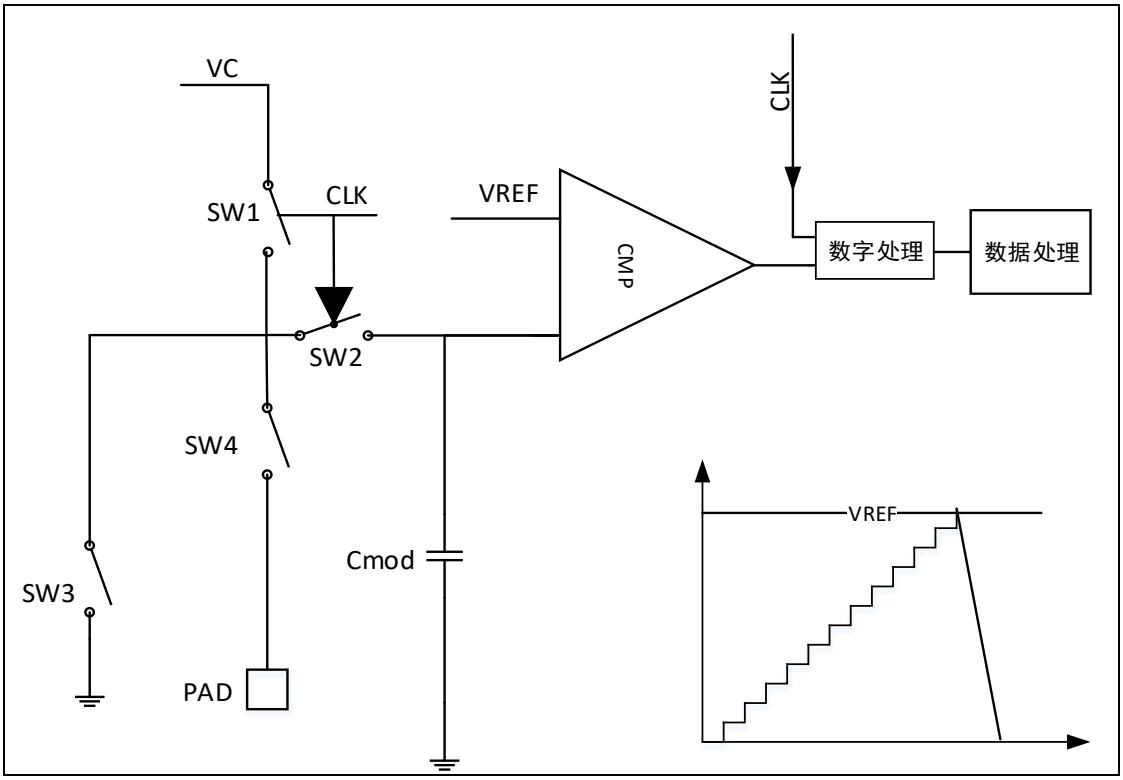
5）当 Cmod 上的电压低于 VREF 时，断开 SW3，停止放电。

6）通过 PWM 得到对应的测量窗口，测量相应窗口放电信号的占空比。

7）数据处理判断是否发生了触摸事件。如果 PAD 上电容增加（有手指触摸），则充电更快，Cmod 充电更快，此时充电时间短，放电时间不变，那么占空比就增大。

46.4.2. CSA 实现方案

图 46-3 CSA 实现方案



1）初始化：SW1 断开，SW2、SW3、SW4 闭合，PAD 和 Cmod 对地放电。初始化完成后，SW3 断开。

2）SW2 断开，SW1、SW4 闭合，VC 对 PAD 充电。

3）SW1 断开，SW2、SW4 闭合，PAD 对 Cmod 充电。

4）重复 2-3。

5）当 Cmod 上电压超过 VREF 时，比较器发生翻转。

6）计量初始化到比较器翻转时参考时钟 CLK 发生翻转的次数，传递给数据处理。

7）数据处理判断是否发生了触摸事件。

46.5. 功能描述

46.5.1. 模块时钟

TKEY 模块包含三个时钟域。

TKEY APB 时钟：TKPCLK

TKEY 扫描计数时钟：TKSCLK

TKEY 控制时钟：TKCTRLCLK

APB 接口模块工作在 TKPCLK 时钟域下；时钟分频模块和计数模块工作在 TKSCLK 时钟域下；控制模块工作在 TKCTRLCLK 时钟域下；信号同步模块和模拟接口模块工作在 TKPCLK，TKSCLK 和 TKCTRLCLK 时钟域下。

TKPCLK 来源于 PCLK2，可通过配置 PCLK2 来设置 TKPCLK。

TKSCLK 来源于 PLL3QCLK。可通过配置 PLL 来设置 TKSCLK。

TKCTRLCLK 来源于 RCL。可通过配置 RCL 来设置 TKCTRLCLK。

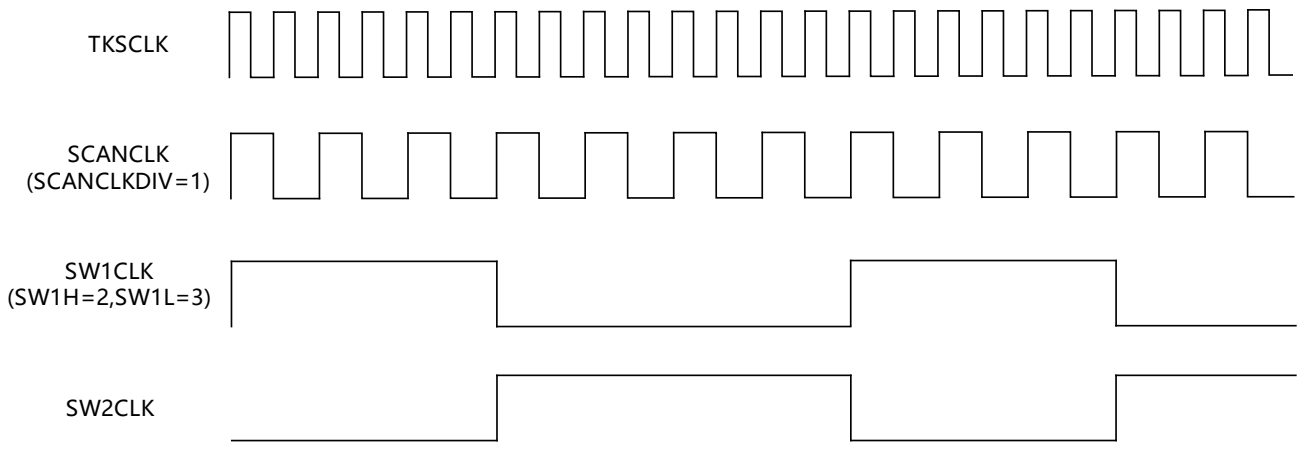
TKEY STOP 模式下需 PLL 常开，功耗较大。

46.5.2. 扫描时钟

将控制 SW1 和 SW2 闭合和断开的时钟 SW1CLK 和 SW2CLK 称为扫描时钟，即充放电时钟。SW1CLK 和 SW2CLK 互不交叠。扫描时钟的时钟源为 TKSCLK。扫描时钟支持两级分频。

第 1 级分频由 TKEY_DIVR 的 SCANCLKDIV 控制，FSCANCLK=FTKSCLK/(SCANCLKDIV+1)。第 2 级分频由 TKEY_SCCR 的 SW1H 和 SW1L 控制，FSW1CLK=FSW2CLK=FSCANCLK/(SW1H+SW1L+2)。

举例：SCANCLKDIV=1，SW1H=2，SW1L=3 时，扫描时钟 SW1CLK，SW2CLK 和 TKSCLK 的关系如下图：



46.5.3. 扫描时钟扩频

扫描时钟支持扩频功能。扫描时钟扩频功能由 TKEY_CR 的 SPREAD 位控制。当 SPREAD 置 1 时，使能扫描时钟扩频。扫描时钟扩频的抖动范围受 TKEY_CR 的 RANDM 位控制。

当 RANDM[1:0]的值为

00：扫描时钟随机抖动±1

01：扫描时钟随机抖动±1，±2

10：扫描时钟随机抖动±1，±2，±3

11：扫描时钟随机抖动±1，±2，±3，±4

使能扫描时钟后，扫描时钟的中心频率
FSW1CLK=FSW2CLK=FSCANCLK/((SW1H+SW1L+2)+(RANDM+1))。

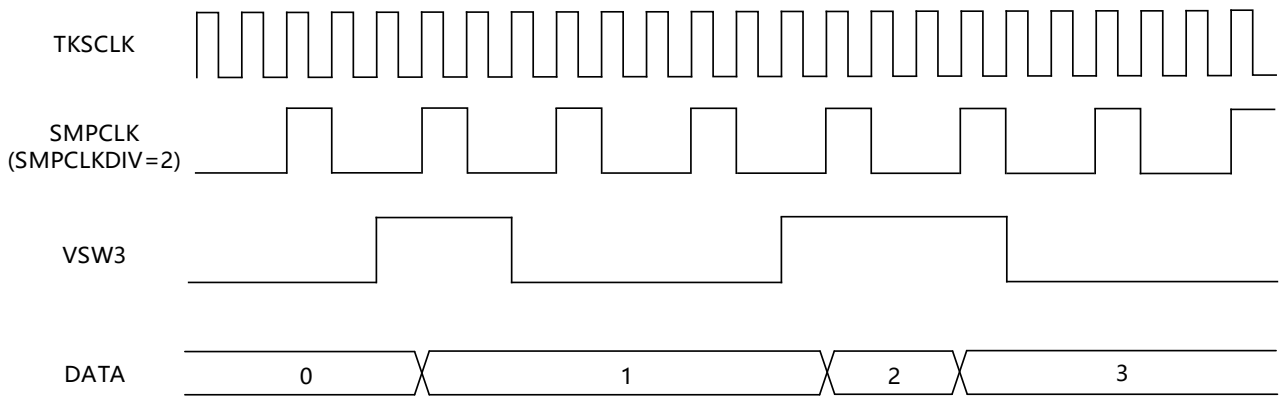
扫描时钟的随机抖动值会加在 SW1CLK 的高电平上，用于改变扫描时钟的频率。扫描时钟的频率会在使能扫描时钟扩频后降低。

随机抖动应在一段时间内保证数学累计为 0。2^10-1 个扫描时钟后抖动累计为 0。

46.5.4. 采样时钟

CSD 模式存在采样时钟。将计数放电信号 VSW3 占空比的时钟称为采样时钟。采样时钟的时钟源为 TKSCLK。采样时钟支持分频。分频由 TKEY_DIVR 的 SMPCLKDIV 控制，FSMPCLK=FTKSCLK/(SMPCLKDIV+1)。

举例：SMPCLKDIV=2 时，采样时钟 SMPCLK 和 TKSCLK 的关系如下图。采样数据 DATA 只会在 SMPCLK 的上升沿，并且 VSW3 为高时才会加 1。



46.5.5. 通道选择

TKEY 包含 16 个 CX 通道。可以通过通道使能寄存器 TKEY_CXENR 使能对应的 CX 通道。

CX 通道的正常使用还需要将控制寄存器 TKEY_CR 的通道使能总控制位 CHEN 位置 1。

46.5.6. 单次扫描模式

在单次扫描模式下，TKEY 模块执行一组按键扫描，CONT 位为 0 时，可以通过将 TKEY_CR 寄存器中的 START 位置 1 来启动单次扫描。

单次扫描模式：

- 每一个使能的 CX 通道扫描结束，扫描的通道号保存在状态寄存器 TKEY_SR 的 CHUNUM 中，扫描的数据保存在数据寄存器 TKEY_DR 中。
- 扫描的数据保存在相应的的通道 x 数据寄存器 TKEY_CHx 中。
- 所有使能的 CX 通道扫描结束，EOC 标志位置 1。
- 如果使能了 EOCIE，则产生 EOC 中断。
- 如果使能了 MEOCIE，则产生 MEOC 中断。

46.5.7. 连续扫描模式

在连续扫描模式下，TKEY 模块连续执行按键扫描，当前面的一组按键扫描结束时，等待时间间隔寄存器所设

定的时间，继续执行下一组的按键扫描。CONT 位为 1 时，可以通过将 TKEY_CR 寄存器中的 START 为 1 来启动连续扫描。

连续扫描模式启动后停下来需要禁止 TKEN。禁止 TKEN 的时间至少维持一个 TKCTRLCLK。

连续扫描时，状态寄存器 TKEY_SR 的 BUSY 标志一直为 1，除非发生 TIMEOUT 和 YESTOUCH 事件时，BUSY 标志会清 0，连续扫描模式停止。

连续扫描模式：

- 每一个使能的 CX 通道扫描结束，扫描的通道号保存在状态寄存器 TKEY_SR 的 CHUNUM 中，扫描的数据保存在数据寄存器 TKEY_DR 中。
- 扫描的数据保存在相应的的通道 x 数据寄存器 TKEY_CHx 中。
- 所有使能的 CX 通道扫描结束，EOC 标志位置 1。
- 如果使能了 EOCIE，则产生 EOC 中断。
- 如果使能了 MEOCIE，则产生 MEOC 中断。

46.5.8. 常规模式

控制寄存器 TKEY_CR 的 AUTO=0，常规模式下，开启按键扫描过程以后，可以根据通道的扫描数据软件处理扫描结果。

常规模式下，每一个通道的扫描数据是可以进行多次采样的。当 TKEY_CFGR1 中的 SMPSEL 为

0：按键采样 1 次输出数据，数据为 1 次采样的平均值

1：按键采样 3 次输出数据，数据为 1 次采样的平均值（去除最大值及最小值）

2：按键采样 6 次输出数据，数据为 4 次采样的平均值（去除最大值及最小值）

3：按键采样 10 次输出数据，数据为 8 次采样的平均值（去除最大值及最小值）

常规模式下的多次采样可以增加扫描扫描数据的稳定性，但是会增加扫描的时间。

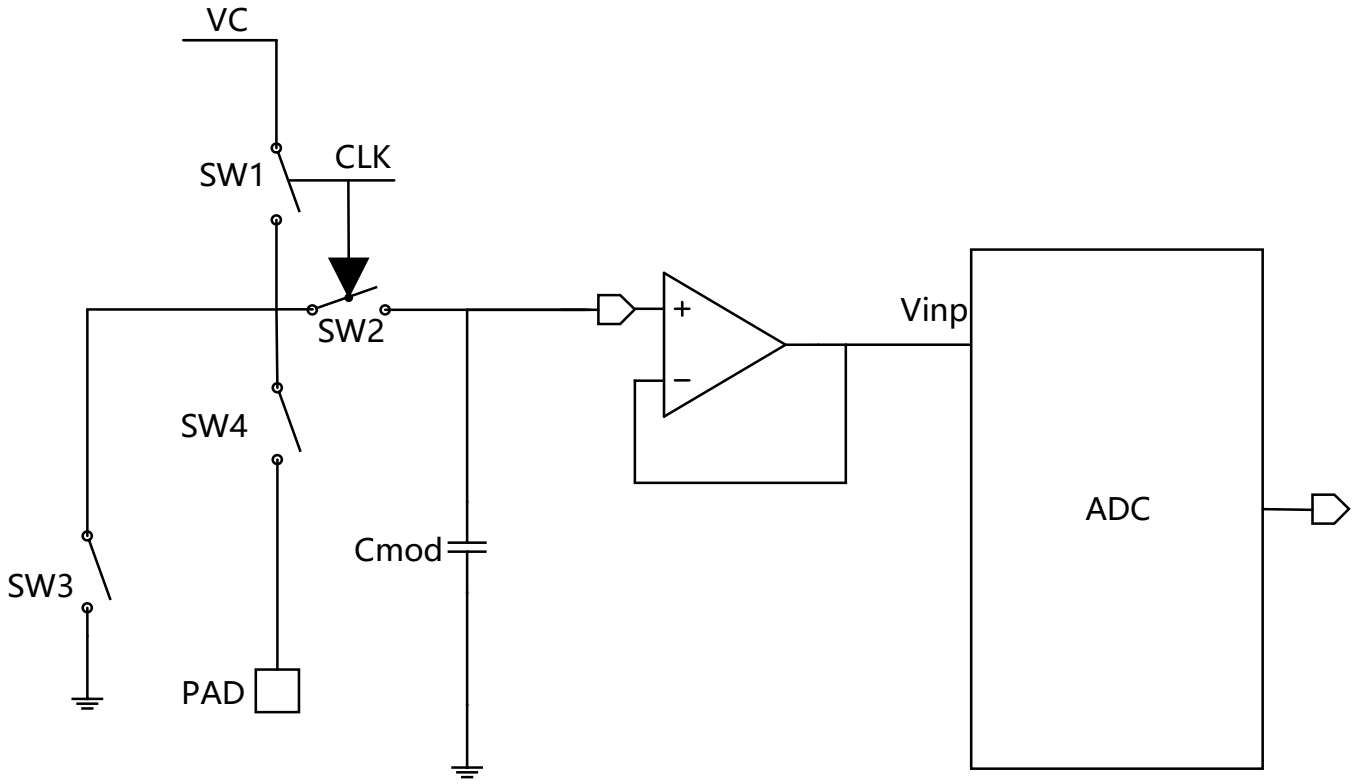
46.5.9. 自动模式

控制寄存器 TKEY_CR 的 AUTO=1，自动模式下，软件需要配置通道门限值和通道基准值，开启扫描之后，在 CSA 模式下，当扫描数据和通道门限值的和小于基准值时，判定为有按键按下；在 CSD 模式下，当扫描数据和通道门限值的和大于基准值时，判定为有按键按下。

自动模式下，可以通过 TKEY_CFGR1 中的 FLTSEL 来选择按键触摸条件硬件滤波次数。当 FLTSEL 不为 0 时，则使能了滤波功能。

例如当设置 FLTSEL=5，则滤波次数为 5。需要检测到连续 6 次的同一个按键被触摸才会判定为按键触摸。

46.5.10. ADC 模式



CSA 模式下，Cmod 的电压输出可以经过 BUFFER 连接到 ADC 的正端输入。在 PAD 的上的电荷转移到 Cmod 时，可以使用 ADC 检测 Cmod 的电压变化。有按键触摸和无按键触摸时，Cmod 端的电压变化有差别。根据 ADC 采样输出的数据可以识别到按键是否触摸。

为了使 CSA 模式下，充电次数可设，可以使能 TKEY_CR 的 CHGDONEEN 位，在 TKEY_NSETR 寄存器中写 CHGNUM，设置充电次数。在 TKEY_TWAITR 寄存器中写 WAITTIME，设置 CSA 模式下，充电次数达到设置次数后的等待时间，等待时间结束后，再经过 4 个 TKCTRLCLK，Cmod 开始放电。

可以使能 TKEY_CR 的 CHG，使得 CSA 的每一次充电都可以产生 CHG 标志位。如果使能了 CHGIE，会产生 CHG 中断。

46.5.11. 屏蔽通道

屏蔽通道一般用于屏蔽水滴。

屏蔽通道可以通过将 TKEY_CR 的 SHIELDEN 位置 1 来使能。使能后，屏蔽通道 TK_SHIELD 会出现和扫描时钟振幅、频率和相位相同的信号。

通道的寄生电容为 C_P，手指的电容为 C_F。通道的 PAD 周围一般会铺上填充网格连接到地。当水滴滴在 PAD 表面时，由于水的导电性，它会为电场线提供强烈的耦合路径以使它返回接地，这样就引入了一个与 C_P 并联的 C_{LD} 电容。电容变大，Cmod 充电时间变短，放电时间不变，占空比变大，扫描数据变大。由于水滴的造成数据增量可能等于手指触摸造成的增量，因此会发生误触。

为了抵消由于水滴造成的电容，需要将围绕 PAD 的填充网格连接到屏蔽通道，屏蔽通道的信号和扫描时钟振幅、频率和相位相同，这样水滴两侧的电压保持相同电位，所以消除了水滴引起的 C_{LD} 电容。因此扫描数据因水滴引起的增量很小。

46.5.12. 电阻补偿

电阻补偿功能是在通道串上一个补偿电阻，补偿电阻的阻值可以通过 TKEY_CFGR1 的 RCPSEL 位来选择。当各个通道之间的扫描数据差异较大或者 ESD 效果不好时，可以通过设置补偿电阻的方式来改善。补偿电阻也可以提高抗噪声能力。

46.5.13. 电容补偿

电容补偿功能是在通道并上一个补偿电容，补偿电容的容值可以通过 TKEY_CFGR1 的 CCPSEL 位来选择。当各个通道之间的扫描数据差异较大时，可以通过设置补偿电容的方式来改善。

46.5.14. CSD 预充功能

CSD 模式支持 Cmod 电容预充。将 TKEY_CR 的 PCEN 置 1 可以使能预充功能。

由于 CSD 模式的通道计数要在比较器第一次翻转后才开始，所以 Cmod 电压达到比较器翻转电压的时间会影响到 CSD 模式下的整体工作时间。

预充功能使能后，会在每个通道开启扫描前使用一个大的电压给 Cmod 充电，加速 Cmod 电压的建立，使得 Cmod 上的电压快速达到比较器的翻转电压。

46.5.15. CSD 放电模式

CSD 模式下，可以选择 Cmod 电容的放电模式。当比较器翻转时，可以通过电流源或者电阻来释放 Cmod 上的电荷（在图 1-3 中简单表示为电阻 res 放电）。放电模式通过 TKEY_CR 的 DISMS 位来选择。

当 DISMIS[2:0]为

000：电阻只在比较器输出为高时放电

001：电流源只在比较器输出为高时放电

010：电阻只在比较器输出为高时放电，电流源始终放电

011：电流源只在比较器输出为高时放电，电阻始终放电

100：电阻和电流源在比较器输出为高时放电

其他值：无作用

可以根据 Cmod 电容的大小以及具体的应用场景来选择不同的放电模式。

46.5.16. 时间设置说明

CSA 超时时间由 TKEY_TSETR 的 TSET 设置。CSA 超时时间 Ttout = (TSET+1)*TTKCTRLCLK。CSA 超时时间在扫描每一个通道时开始计时，当计数到超时时间，比较器输出低，则 TKEY_SR 的 TIMEOUT 被置位，表示扫描超时，相应的扫描通道数据置为 0xFFFF。

CSD PWM 周期由 TKEY_TSETR 的 TSET 设置。CSD PWM 周期 Tpwm = (TSET+1)*TTKCTRLCLK。CSD PWM 周期在扫描每一个通道时比较器第一次翻转后进行计数。采样时钟会在这 PWM 周期内对放电信号 VSW3 进行采样计数。

Cs 电容放电时间由 TKEY_TESTR 的 CST 设置。Cs 电容放电时间 TCs = (CST+1)*TTKCTRLCLK。CSA 模式下，每个通道扫描之前都会先执行放电。将 Cmod 上的电荷释放到地。

两组扫描之间的间隔时间由 TKEY_INTVLR 的 INTERVAL 设置。连续扫描模式下，当所有使能的通道扫描完成后开始计时，计时完成后开始下一次扫描。

46.5.17. TKEY OUT

TKEY OUT 信号是 TKEY 模块触发 ADC 的信号。当 CSA 模式充电次数达到设定次数或者 CSA 模式下每一次充电完成，TKEY_OUT 信号置 1，持续一个 TKPCLK 后清 0。

46.5.18. TKEY 唤醒

对于 TKEY，可以在下列情况产生唤醒信号

通道扫描超时（TIMEOUT）

按键触摸（YESTOUCH）

通道扫描结束（EOC）

可以使用单独的中断使能位控制唤醒信号

通道扫描超时和按键触摸的唤醒信号需要将 TKEY_CR 中的 START 位置 1 来清 0。通道扫描结束信号会自动清 0。

唤醒事件	使能控制位
通道扫描超时（TIMEOUT）	TIMEOUTIE
按键触摸（YESTOUCH）	YESTOUCHIE
通道扫描结束（EOC）	EOCIE

46.5.19. TKEY 中断

对于 TKEY，可在下列情况产生中断：

- CSA 模式充电次数达到设定次数（CHGDONE 标志）
- CSA 模式下每一次充电完成（CHG 标志）
- 比较器翻转完成标志（DONE 标志）
- 通道扫描超时（TIMEOUT 标志）
- 按键触摸（YESTOUCH 标志）
- 通道扫描结束（EOC 标志）

可以使用单独的中断使能位控制中断

中断事件	事件标志	使能控制位
CSA 模式充电次数达到设定次数	CHGDONE	CHGDONEIE
CSA 模式下每一次充电完成	CHG	CHGIE
比较器翻转完成标志	DONE	DONEIE
通道扫描超时	TIMEOUT	TIMEOUTIE
按键触摸	YESTOUCH	YESTOUCHIE
通道扫描结束	EOC	EOCIE

46.6. 使用说明

46.6.1. 时钟和复位

TKEY 模快包含三个时钟，分别是 TKPCLK，TKSCLK，TKCTRLCLK。三个时钟的产生需要先进行模块时钟使能。可以通过将复位和时钟控制寄存器（RCC_APB2CKENR）的 TKEYCKEN 置 1 来进行模块时钟的使能

TKEY 模块的复位可以通过将复位和时钟控制寄存器（RCC_APB2RSTR）的 TKEYRST 清 0 来进行模块的复位。

46.6.2. 通道引脚

根据需要配置 TKEY 的通道引脚，通道对应的 GPIO 需要配置为 GPIO_ANALOG，并打开 GPIO 的模拟开关。

46.6.3. 常规模式

TKEY 的常规模式主要用于扫描测量通道 Cx 的容值。常规模式的软件操作流程如下：

46.6.3.1. CSA 模式

1）配置 TKEY_CXENR 寄存器，使能将要扫描的通道（CXnEN）

2）配置 TKEY_DIVR 寄存器，设置扫描时钟分频（SCANCLKDIV）

3）配置 TKEY_SCCR 寄存器，设置扫描时钟 SW1 的高电平（SW1H）和 SW1 低电平宽度（SW1L）

4）配置 TKEY_INTVLR 寄存器，设置两组扫描之间的间隔时间（INTERVAL）

5）配置 TKEY_TSETR 寄存器，设置 CSA 模式超时时间（TSET）和 Cs 电容放电时间（CST）

6）配置 TKEY_CFLTR 寄存器，设置比较器滤波使能（CFLTEN）和比较器滤波值（CFLTNUM）

7）配置 TKEY_CFGR1 寄存器，包括：

- 常规模式下采样系数选择（SMPSEL）
- 通道电容补偿选择（CCPSEL）
- 通道电阻补偿选择（RCPSEL）
- 充电电压 VLDO 选择（VCHRSEL）
- 比较器电压选择（VCMPSEL）

8）配置 TKEY_IER 寄存器，设置通道扫描结束中断使能（EOCIE=1）

9）配置 TKEY_CR 寄存器，包括：

- 扫描时钟抖动控制（RANDM）
- TKEY 工作模式（AUTO=0）
- 扫描时钟扩频使能（SPREAD）
- 扫描模式控制（CONT）
- 屏蔽通道使能（SHIELDEN）
- TKEY 模式选择（TKCSA=1）

10）配置 TKEY_CR 寄存器，设置 TKEY 模块使能控制位（TKEN=1），软件等待 10us 后，设置通道使能总控制位（CHEN=1）

11）配置 TKEY_CR 寄存器，设置启动扫描（START=1）

12）等待 EOC 中断

13）从 TKEY_CHx 寄存器中读取扫描通道的数据

46.6.3.2. CSD 模式

1）配置 TKEY_CXENR 寄存器，使能将要扫描的通道（CXnEN）

2）配置 TKEY_DIVR 寄存器，设置扫描时钟分频（SCANCLKDIV）和采样比特流的时钟分频（SMPCLKDIV）

3）配置 TKEY_SCCR 寄存器，设置扫描时钟 SW1 的高电平（SW1H）和 SW1 低电平宽度（SW1L）

4）配置 TKEY_INTVLR 寄存器，设置两组扫描之间的间隔时间（INTERVAL）

5）配置 TKEY_TSETR 寄存器，设置 CSD 模式 PWM 周期的长度（TSET）

6）配置 TKEY_CFGR1 寄存器，包括：

- 常规模式下采样系数选择（SMPSEL）
- 通道电容补偿选择（CCPSEL）
- 通道电阻补偿选择（RCPSEL）
- 充电电压 VLDO 选择（VCHRSEL）
- 比较器电压选择（VCMPSEL）

7）配置 TKEY_CFGR2 寄存器，包括：

- 恒流源放电电流选择（IDISSEL）
- 电阻放电电阻选择（RDISSEL）

8）配置 TKEY_IER 寄存器，设置通道扫描结束中断使能（EOCIE=1）

9）配置 TKEY_CR 寄存器，包括：

- 放电模式选择（DISMS）
- 扫描时钟抖动控制（RANDM）
- TKEY 工作模式（AUTO=0）
- 扫描时钟扩频使能（SPREAD）
- 扫描模式控制（CONT）
- 屏蔽通道使能（SHIELDEN）
- 预充使能（PCEN）
- TKEY 模式选择（TKCSA=0）

10）配置 TKEY_CR 寄存器，设置 TKEY 模块使能控制位（TKEN=1），软件等待 10us 后，设置通道使能总控制位（CHEN=1）

11）配置 TKEY_CR 寄存器，设置启动扫描（START=1）

12）等待 EOC 中断

13）从 TKEY_CHx 寄存器中读取扫描通道的数据

46.6.3.3. ADC 模式

1）配置 TKEY_CXENR 寄存器，使能将要扫描的通道（CXnEN）

2）配置 TKEY_DIVR 寄存器，设置扫描时钟分频（SCANCLKDIV）

3）配置 TKEY_SCCR 寄存器，设置扫描时钟 SW1 的高电平（SW1H）和 SW1 低电平宽度（SW1L）

4）配置 TKEY_INTVLR 寄存器，设置两组扫描之间的间隔时间（INTERVAL）

5）配置 TKEY_TSETR 寄存器，设置 CSA 模式超时时间（TSET）和 Cs 电容放电时间（CST）

6）配置 TKEY_CFLTR 寄存器，设置比较器滤波使能（CFLTEN）和比较器滤波值（CFLTNUM）

7）配置 TKEY_NSETR 寄存器，设置 CSA 模式下的充电次数（CHGNUM）

8）配置 TKEY_TWAITR 寄存器，设置 CSA 模式下，充电次数达到设置次数后的等待时间（WAITTIME）

9）配置 TKEY_CFGR1 寄存器，包括：

- 通道电容补偿选择（CCPSEL）
- 通道电阻补偿选择（RCPSEL）
- 充电电压 VLDO 选择（VCHRSEL）
- 比较器电压选择（VCMPSEL）

10）配置 TKEY_IER 寄存器，设置 CSA 模式充电次数达到设定次数中断使能（CHGDONEIE=1）

11）配置 TKEY_CR 寄存器，包括：

- CSA 模式下，充电次数达到设定次数功能使能（CHGDONEEN）
- CSA 模式下，每一次充电完成功能使能位（CHGEN）
- 扫描时钟抖动控制（RANDM）
- TKEY 工作模式（AUTO=0）
- 扫描时钟扩频使能（SPREAD）
- 扫描模式控制（CONT）
- 屏蔽通道使能（SHIELDEN）
- TKEY 模式选择（TKCSA=1）

12）配置 TKEY_CR 寄存器，设置 TKEY 模块使能控制位（TKEN=1），软件等待 10us 后，设置通道使能总控制位（CHEN=1）

13）配置 TKEY_CR 寄存器，设置启动扫描（START=1）

14）等待 CHGDONE 中断

15）TKEY_OUT 可以触发 ADC 采样 Cmod 上的电压值，判断是否有按键触摸

46.6.4. 自动模式

TKEY 的自动模式主要用于检测通道 Cx 的容值的变化，当扫描通道的数据的变化大于设置的按键触摸阈值时，产生按键触摸事件。自动模式的软件操作流程如下：

46.6.4.1. CSA 模式

1）配置 TKEY_CXENR 寄存器，使能将要扫描的通道（CXnEN）

2）配置 TKEY_DIVR 寄存器，设置扫描时钟分频（SCANCLKDIV）

3）配置 TKEY_SCCR 寄存器，设置扫描时钟 SW1 的高电平（SW1H）和 SW1 低电平宽度（SW1L）

4）配置 TKEY_INTVLR 寄存器，设置两组扫描之间的间隔时间（INTERVAL）

5）配置 TKEY_TSETR 寄存器，设置 CSA 模式超时时间（TSET）和 Cs 电容放电时间（CST）

6）配置 TKEY_CFLTR 寄存器，设置比较器滤波使能（CFLTEN）和比较器滤波值（CFLTNUM）

7）配置 TKEY_CFGR1 寄存器，包括：

- 自动模式下，按键触摸条件硬件滤波次数选择（FLTSEL）
- 通道电容补偿选择（CCPSEL）
- 通道电阻补偿选择（RCPSEL）
- 充电电压 VLDO 选择（VCHRSEL）
- 比较器电压选择（VCMPSEL）

8）配置 TKEY_CHx 寄存器，设置通道唤醒阈值（CNTx）

9）配置 TKEY_IER 寄存器，设置自动模式下按键触摸中断使能（YESTOUCHIE=1）

10）配置 TKEY_CR 寄存器，包括：

- 扫描时钟抖动控制（RANDM）
- TKEY 工作模式（AUTO=1）
- 扫描时钟扩频使能（SPREAD）
- 扫描模式控制（CONT）
- 屏蔽通道使能（SHIELDEN）
- TKEY 模式选择（TKCSA=1）

11）配置 TKEY_CR 寄存器，设置 TKEY 模块使能控制位（TKEN=1），软件等待 10us 后，设置通道使能总控制位（CHEN=1）

12）配置 TKEY_CR 寄存器，设置启动扫描（START=1）

13）等待 YESTOUCH 中断

14）切换到常规模式进行按键扫描

46.6.4.2. CSD 模式

1）配置 TKEY_CXENR 寄存器，使能将要扫描的通道（CXnEN）

2）配置 TKEY_DIVR 寄存器，设置扫描时钟分频（SCANCLKDIV）和采样比特流的时钟分频（SMPCLKDIV）

3）配置 TKEY_SCCR 寄存器，设置扫描时钟 SW1 的高电平（SW1H）和 SW1 低电平宽度（SW1L）

4）配置 TKEY_INTVLR 寄存器，设置两组扫描之间的间隔时间（INTERVAL）

5）配置 TKEY_TSETR 寄存器，设置 CSD 模式 PWM 周期的长度（TSET）

6）配置 TKEY_CFGR1 寄存器，包括：

- 自动模式下，按键触摸条件硬件滤波次数选择（FLTSEL）
- 通道电容补偿选择（CCPSEL）
- 通道电阻补偿选择（RCPSEL）
- 充电电压 VLDO 选择（VCHRSEL）
- 比较器电压选择（VCMPSEL）

7）配置 TKEY_CFGR2 寄存器，包括：

- 恒流源放电电流选择（IDISSEL）
- 电阻放电电阻选择（RDISSEL）

8）配置 TKEY_CHx 寄存器，设置通道唤醒阈值（CNTx）

9）配置 TKEY_IER 寄存器，设置自动模式下按键触摸中断使能使能（YESTOUCHIE=1）

10）配置 TKEY_CR 寄存器，包括：

- 放电模式选择（DISMS）
- 扫描时钟抖动控制（RANDM）
- TKEY 工作模式（AUTO=1）
- 扫描时钟扩频使能（SPREAD）
- 扫描模式控制（CONT）
- 屏蔽通道使能（SHIELDEN）
- 预充使能（PCEN）
- TKEY 模式选择（TKCSA=0）

11）配置 TKEY_CR 寄存器，设置 TKEY 模块使能控制位（TKEN=1），软件等待 10us 后，设置通道使能总控制位（CHEN=1）

12）配置 TKEY_CR 寄存器，设置启动扫描（START=1）

13）等待 YESTOUCH 中断

14）切换到常规模式进行按键扫描

46.7. TKEY 寄存器描述

46.7.1. 寄存器列表

TKEY 寄存器基地址：0x40016400

偏移	名称	复位值	描述
0x00	TKEY_SR	0x00000000	状态寄存器
0x04	TKEY_IER	0x00000000	中断使能寄存器
0x08	TKEY_CR	0x00000000	控制寄存器
0x0C	TKEY_CFGR1	0x0000000f	配置寄存器 1
0x10	TKEY_CFGR2	0x00000000	配置寄存器 2
0x14	TKEY_INTVLR	0x00000000	时间间隔寄存器
0x18	TKEY_DIVR	0x00000000	时钟分频寄存器
0x1C	TKEY_SCCR	0x00000000	扫描时钟控制寄存器
0x20	TKEY_TSETR	0x00000000	时间设置寄存器
0x24	TKEY_CXENR	0x00000000	通道使能寄存器
0x28	TKEY_DR	0x00000000	数据寄存器
0x2C+(x*4)	TKEY_THx	0x00000000	通道 x 门限寄存器（x=0~15）
0x6C+(x*4)	TKEY_CHx	0x00000000	通道 x 数据寄存器（x=0~15）
0xAC	TKEY_CFLTR	0x00000003	比较器滤波寄存器
0xB0	TKEY_NSETR	0x00000000	充电次数设置寄存器
0xB4	TKEY_TWAITR	0x00000000	等待时间寄存器

46.7.2. 状态寄存器(TKEY_SR: 00h)

位域	名称	属性	复位值	描述
15:11	RSV	-	-	保留
10	CHGDONE	RO	0x0	CSA 模式充电次数达到设定次数标志 CSA 模式下，充电次数达到充电次数设置寄存器的值时置位，写 1 清 0。 0：充电次数未达到设定次数 1：充电次数达到设定次数
9	CHG	RO	0x0	CSA 模式每一次充电完成标志 CSA 模式下，PAD 对 Cmod 每一次充电完成时置位，写 1 清 0。 0：每次一次充电未完成 1：每一次充电完成
8	DONE	RO	0x0	比较器翻转完成标志 CSA 模式下，比较器翻转完成（比较器输出逻辑电平由低到高）时置位，软件写 1 清除。 0：比较器翻转未完成 1：比较器翻转完成 注：在 CHGDONEEN 位置 1 时，该标志位会在充电次数达到设定次数且等待时间达到设定次数才会置 1，此时忽略比较器的翻转行为。
7:4	CHNUM[3:0]	RO	0x0	常规模式下：扫描通道号或者扫描超时的通道号 自动模式下：按键触摸的通道号
3	BUSY	RO	0x0	通道扫描 BUSY 标志 注：单次扫描时停止扫描时需要等待 BUSY 标志为 0，连续扫描时 BUSY 标志一直为 1，除了发生 TIMEOUT 和 YESTOUCH 时，BUSY 标志会清 0。
2	TIMEOUT	RO	0x0	通道扫描超时标志 该位由硬件在扫描超时时置位，软件写 1 清除。 0：未发生超时事件 1：已发生超时事件 注：CSA 模式下，在超时时间计数到 0 时，比较器输出低，则通道扫描超时标志置位。CSD 模式下，当采样数据为 0xFFFF 时，通道扫描超时标志置位。
1	YESTOUCH	RO	0x0	按键触摸标志 该位由硬件在自动模式下，当扫描的通道发生按键触摸事件时置位，软件写 1 清除。 0：未发生按键触摸事件 1：已发生按键触摸事件
0	EOC	RO	0x0	通道扫描结束标志 该位由硬件在所有使能的通道扫描结束后置位，软件写 1 清除。 0：扫描未完成 1：扫描已完成

46.7.3. 中断使能寄存器(TKEY_IER: 04h)

位域	名称	属性	复位值	描述
15:11	RSV	-	-	保留
10	CHGDONEIE	RW	0x0	CSA 模式充电次数达到设定次数中断使能 0：禁止 1：使能
9	CHIE	RW	0x0	CSA 模式每一次充电完成中断使能 0：禁止 1：使能
8	DONEIE	RW	0x0	比较器翻转完成中断使能 0：禁止 1：使能
7:3	RSV	-	-	保留
2	TIMEOUTIE	RW	0x0	通道扫描超时中断使能 0：禁止 1：使能
1	YESTOUCHIE	RW	0x0	自动模式下按键触摸中断使能 0：禁止 1：使能
0	EOCIE	RW	0x0	通道扫描结束中断使能 0：禁止 1：使能

46.7.4. 控制寄存器(TKEY_CR: 08h)

位域	名称	属性	复位值	描述
15	CHGDONEEN	RW	0x0	CSA 模式下，充电次数达到设定次数功能使能位 0：禁止 1：使能
14	CHGEN	RW	0x0	CSA 模式每一次充电完成功能使能位 0：禁止 1：使能 注：该功能需要 CHGDONEEN 为 1 时使能才有效。

13:11	DISMS[2:0]	RW	0x0	CSD 模式下，放电模式选择 000：电阻只在比较器输出为高时放电 001：电流源只在比较器输出为高时放电 010：电阻只在比较器输出为高时放电，电流源始终放电 011：电流源只在比较器输出为高时放电，电阻始终放电 100：电阻和电流源在比较器输出为高时放电 其他：无作用 注：放电都只在 PWM 周期内进行。
10:9	RANDM[1:0]	RW	0x0	扫描时钟抖动控制 00：扫描时钟随机抖动±1 01：扫描时钟随机抖动±1，±2 10：扫描时钟随机抖动±1，±2，±3 11：扫描时钟随机抖动±1，±2，±3，±4 注：随机抖动应在一段时间内保证数学累计为 0。2^10-1 个扫描时钟后抖动累计为 0。
8	AUTO	RW	0x0	TKEY 工作模式 0：常规模式，软件处理扫描结果 1：自动模式，硬件自动判断按键触摸
7	SPREAD	RW	0x0	扫描时钟扩频使能 0：禁止 1：使能
6	CONT	RW	0x0	扫描模式控制 0：单次扫描 1：连续扫描 注：连续扫描模式启动后停下来需要禁止 TKEN。禁止 TKEN 的时间至少维持一个 TKCTRLCLK。
5	SHIELDEN	RW	0x0	屏蔽通道使能位 0：禁止 1：使能
4	PCEN	RW	0x0	CSD 模式下预充使能位 0：禁止 1：使能
3	CHEN	RW	0x0	通道使能总控制位 0：禁止 1：使能

2	START	RW	0x0	启动扫描 软件向该位写 1 启动扫描，扫描开始后，硬件自动清除该位 0：未启动扫描 1：启动扫描
1	TKCSA	RW	0x0	TKEY 模式选择 0：CSD 模式 1：CSA 模式
0	TKEN	RW	0x0	TKEY 模块使能控制位 0：禁止 1：使能 注：TKEY 内部模拟模块稳定需要 10us 时间，TKEN 置 1 后，需 10us 后才能设置其他位。

46.7.5. 配置寄存器 1(TKEY_CFGR1: 0Ch)

位域	名称	属性	复位值	描述
15:14	RSV	-	-	保留
13:11	FLTSEL[2:0]	RW	0x0	自动模式下，按键触摸条件硬件滤波次数选择 000：不滤波 001：2 次 010：3 次 011：4 次 100：5 次 101：6 次 110：7 次 111：8 次
10:9	SMPSEL[1:0]	RW	0x0	常规模式下，采样次数选择 00：按键采样 1 次输出数据，数据为 1 次采样的平均值 01：按键采样 3 次输出数据，数据为 1 次采样的平均值（去除最大值及最小值） 10：按键采样 6 次输出数据，数据为 4 次采样的平均值（去除最大值及最小值） 11：按键采样 10 次输出数据，数据为 8 次采样的平均值（去除最大值及最小值）

8:6	CCPSEL[2:0]	RW	0x0	通道电容补偿选择 000：0 pF 001：2 pF 010：4 pF 011：6 pF 100：8 pF 101：10 pF 110：12pF 111：14pF
5:4	RCPSEL[1:0]	RW	0x0	通道电阻补偿选择 00：0.1KΩ 10：2KΩ 10：4KΩ 11：8KΩ
3:2	VCHRSEL[1:0]	RW	0x3	充电电压 VLDO 选择 00：1.6V 01：2.0V 10：2.4V 11：VDD33
1:0	VCMPSEL[1:0]	RW	0x3	比较器电压选择 00：0.25*VLDO 01：0.5*VLDO 10：0.75*VLDO 11：VBG（1.0V 基准电压）

46.7.6. 配置寄存器 2(TKEY_CFGR2: 10h)

位域	名称	属性	复位值	描述
15:8	IDISSEL[7:0]	RW	0x0	恒流源放电电流选择 电流值 = 0.4uA* IDISSEL[7:0]
7:0	RDISSEL[7:0]	RW	0x0	电阻放电电阻选择 电流值=（比较器电压/512K）* RDISSEL[7:0]

46.7.7. 时间间隔寄存器(TKEY_INTVLR: 14h)

位域	名称	属性	复位值	描述

15:0	INTERVAL[15:0]	RW	0x0	两组扫描之间的间隔时间 TInterval = (INTERVAL+1)*TTKCTRLCLK 连续扫描模式下，当所有使能通道扫描完成后，开始下一次扫描时所等待的时间。

46.7.8. 时钟分频寄存器(TKEY_DIVR: 18h)

位域	名称	属性	复位值	描述
15:8	SCANCLKDIV[7:0]	RW	0x0	扫描时钟分频 分频值 = SCANCLKDIV+1 时钟源为 TKSCLK
7:0	SMPCLKDIV[7:0]	RW	0x0	CSD 模式下，采样比特流的时钟分频 分频值 = SMPCLKDIV+1 时钟源为 TKSCLK

46.7.9. 扫描时钟控制寄存器(TKEY_SCCR: 1Ch)

位域	名称	属性	复位值	描述
15:8	SW1H[7:0]	RW	0x0	扫描时钟 SW1 高电平（SW2 低电平） 00000000：1*TSCANCLK 00000001：2*TSCANCLK … 11111111：256*TSCANCLK
7:0	SW1L[7:0]	RW	0x0	扫描时钟 SW1 低电平（SW2 高电平） 00000000：1*TSCANCLK 00000001：2*TSCANCLK … 11111111：256*TSCANCLK

扫描中心频率：

当 SPREAD = 0 时，Fsw = FSCANCLK /(SW1H+SW1L+2)

当 SPREAD = 1 时，Fsw = FSCANCLK /((SW1H+SW1L+2)+(RANDM+1))

46.7.10. 时间设置寄存器(TKEY_TSETR: 20h)

位域	名称	属性	复位值	描述

15:4	TSET[11:0]	RW	0x0	CSA 模式超时时间设置（TKCSA=1） Ttout = (TSET+1)*TTKCTRLCLK 在扫描每一个通道时开始计时，当计时到超时时间，比较器输出低，则 TIMEOUT 标志被置位，表示扫描超时，相应的扫描通道数据置为 0xFFFF CSD 模式下 PWM 的周期（TKCSA=0） Tpwm = (TSET+1)*TTKCTRLCLK CSD 模式下 PWM 的周期的长度
3:0	CST[3:0]	RW	0x0	Cs 电容放电时间（CSA 模式） TCs = (CST+1)*TTKCTRLCLK

46.7.11. 通道使能寄存器(TKEY_CXENR: 24h)

位域	名称	属性	复位值	描述
15	CX15EN	RW	0x0	通道 Cx15 使能控制，参考 CX0EN 描述
14	CX14EN	RW	0x0	通道 Cx14 使能控制，参考 CX0EN 描述
13	CX13EN	RW	0x0	通道 Cx13 使能控制，参考 CX0EN 描述
12	CX12EN	RW	0x0	通道 Cx12 使能控制，参考 CX0EN 描述
11	CX11EN	RW	0x0	通道 Cx11 使能控制，参考 CX0EN 描述
10	CX10EN	RW	0x0	通道 Cx10 使能控制，参考 CX0EN 描述
9	CX9EN	RW	0x0	通道 Cx9 使能控制，参考 CX0EN 描述
8	CX8EN	RW	0x0	通道 Cx8 使能控制，参考 CX0EN 描述
7	CX7EN	RW	0x0	通道 Cx7 使能控制，参考 CX0EN 描述
6	CX6EN	RW	0x0	通道 Cx6 使能控制，参考 CX0EN 描述
5	CX5EN	RW	0x0	通道 Cx5 使能控制，参考 CX0EN 描述
4	CX4EN	RW	0x0	通道 Cx4 使能控制，参考 CX0EN 描述
3	CX3EN	RW	0x0	通道 Cx3 使能控制，参考 CX0EN 描述
2	CX2EN	RW	0x0	通道 Cx2 使能控制，参考 CX0EN 描述
1	CX1EN	RW	0x0	通道 Cx1 使能控制，参考 CX0EN 描述
0	CX0EN	RW	0x0	通道 Cx0 使能控制 0：禁止 1：使能

46.7.12. 数据寄存器(TKEY_DR: 28h)

位域	名称	属性	复位值	描述
15:0	DATA[15:0]	RO	0x0	通道扫描数据 常规模式下，该寄存器保存最后一个通道的扫描计数结果 自动模式下，该寄存器保存发生按键触摸事件通道的扫描计数结果。当 FLTSEL 不为 0 时，该寄存器保存发生按键触摸事件通道的最新的扫描计数结果。

46.7.13. 通道 x 门限寄存器(TKEY_THx: 2Ch+(x*4), x=0~15)

位域	名称	属性	复位值	描述
15:8	RSV	-	-	保留
7:0	DNX[7:0]	RW	0x0	通道 Cx 的变化量，即门限值

46.7.14. 通道 x 数据寄存器(TKEY_CHx: 6Ch+(x*4), x=0~15)

位域	名称	属性	复位值	描述
15:0	CNTX[15:0]	RW	0x0	常规模式下，表示通道 Cx 扫描计数值； 自动模式下，表示通道基线值，由软件在进入自动模式前写入。

46.7.15. 比较器滤波寄存器(TKEY_CFLTR: ACh)

位域	名称	属性	复位值	描述
15:5	RSV	-	-	保留
4:1	CFLTNUM[3:0]	RW	0x1	比较器滤波值（CSA 模式） 滤波值：CFLTNUM+1 滤波值单位：SW1 周期
0	CFLTEN	RW	0x1	比较器滤波使能位（CSA 模式） 0：禁止 1：使能

46.7.16. 充电次数设置寄存器(TKEY_NSETR: B0h)

位域	名称	属性	复位值	描述

15:0	CHGNUM[15:0]	RW	0x0	CSA 模式下，充电次数设置 配置为 0，则该功能不生效。必须要大于 0 并且 CHGDONEEN 置 1 此功能才生效。

46.7.17. 等待时间寄存器(TKEY_TWAITR: B4h)

位域	名称	属性	复位值	描述
15:0	WAITTIME[15:0]	RW	0x0	CSA 模式下，充电次数到达设置次数后的等待时间，计时单位：8*TTKSCLK。 等待时间结束后，再过 4 个 TTKCTRLCLK，Cmod 开始放电。

47. CRC 计算单元

47.1. 概述

循环冗余校验(Cyclic Redundancy Check CRC)单元是一种根据指定多项式从 8 位/16 位/32 位的数据中产生简短固定位数 CRC 校验码的模块。CRC 主要利用除法及余数的原理来检测或校验数据传输或者保存后可能出现的错误，它的特点是检错能力强，开销小，易于用编码器及检测电路实现。

在应用中，CRC 技术主要应用于核实数据传输或者数据存储的正确性和完整性。CRC 计算单元可以在运行期间计算软件的签名，并将该签名与链接时生成并存储在指定存储单元的参考签名加以比较。

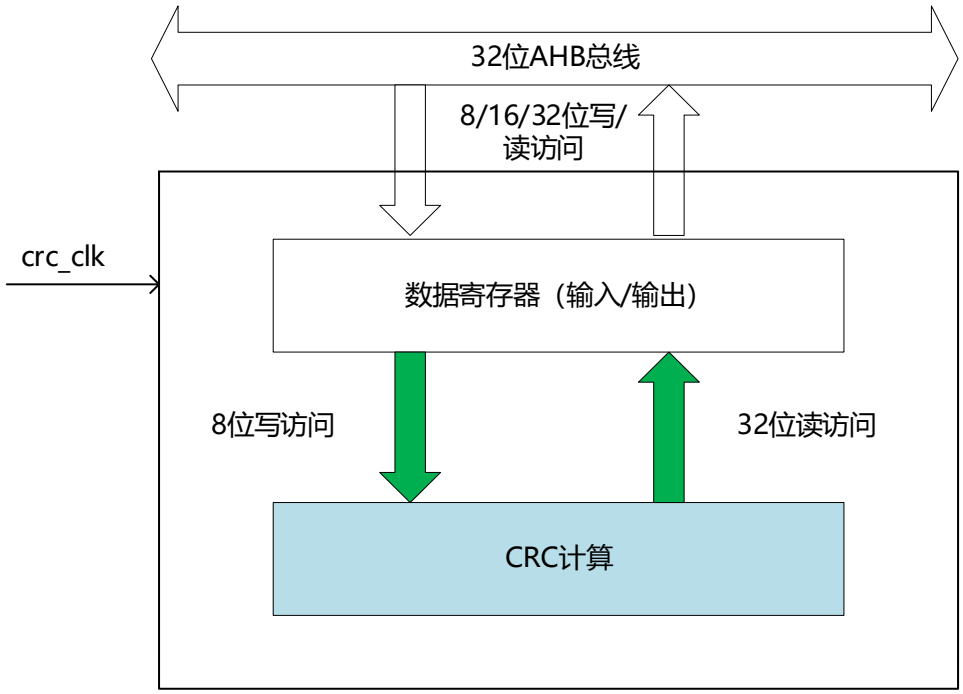
47.2. 主要特性

- 可编程 CRC 初始值
- 可编程结果异或值
- 支持位数可编程的(32 位/ 16 位/8 位/7 位)的完全可编程多项式
- 支持 8 位/16 位/32 位数据输入
- 支持初始值高低位倒序
- 支持计算结果高低位倒序
- 支持结果异或值高低位倒序
- 支持多项式高低位倒序
- 支持输入数据按字节/半字/字倒序
- 默认使用 CRC-32(以太网)多项式：0x04C11DB7
- 单输入/输出 32 位数据寄存器
- 32 位独立数据寄存器(可用于临时存储)
- 对于 32 位/16 位/8(7)位数据大小，CRC 计算分别在 4/2/1 个 AHB 时钟周期(HCLK)内完成

47.3. 功能说明

47.3.1. 功能框图

CRC 功能框图如下图所示。



47.3.2. CRC 操作说明

CRC 的本质是模 2 除法的余数，采用的除数不同，CRC 的类型也就不同。通常 CRC 的除数用生成多项式来表示，本模块支持 32/16/8/7 位的完全可编程多项式。在 K 位信息码（目标发送数据）后再拼接 R 位校验码，使整个编码长度为 N 位，因此这种编码也叫（N,K）码。通俗的说，就是在需要发送的信息后面附加一个数（即校验码），生成一个新的发送数据发送给接收端。这个附加数据要求能够使生成的新数据被生成多项式数整除。

47.3.2.1. 可编程初始化值

- 可使用 CRC_INIT 寄存器对 CRC 初始值进行编程。对 CRC_INIT 寄存器进行写访问时会自动初始化 CRC_DATA 寄存器。
- 向 CRC_CTRL 寄存器中的 RST 控制位写 1 也可将 CRC_DATA 初始化为 CRC_INIT 寄存器中的值。

47.3.2.2. 可编程结果异或值

- 可使用 CRC_OUTXOR 寄存器对 CRC 结果异或值进行编程。
- 如果结果异或值为 0(与结果异或值寄存器默认值相同)，则不需要设置也可以使用。

47.3.2.3. 数据寄存器操作

CRC 计算单元含有 1 个 32 位读/写数据寄存器（CRC_DATA)。它用于输入新数据（写访问）和返回之前 CRC 计算的结果（读访问)。

- 对该寄存器进行写操作时，作为输入寄存器，可以输入要进行 CRC 计算的数据。如需校验的数据是多个字节/半字/字只需按顺序逐次写入。如果不设置 CRC_CTRL 寄存器的 RST 位来清除 CRC_DATA 寄存器，对数据寄存器的每个写操作都会对之前的 CRC 值（存储在 CRC_DATA 中）和新值再做一次 CRC 计算；
- CRC 计算针对字节完成，具体运算数据取决于有效数据字节长度，从有效数据最低字节开始运算；
- 可连续写入数据，无需因之前的 CRC 计算而等待任何状态；
- 对该寄存器进行读操作时，返回上一次 CRC 计算的结果；
- 如果 CRC 数据小于 32 位，从 CRC_DATA 寄存器的最低有效位读取 CRC 结果；

- 可以通过设置控制寄存器的 DATA_LEN 位来选择数据寄存器有效数据长度；
- 可动态调整 CRC 输入数据大小，从而能最大程度地减少给定字节数的写访问次数。例如，对 5 个字节进行 CRC 计算时，可先写入字，然后写入字节

47.3.2.4. 倒序操作

倒序功能可以用于交换输入数据、输出数据、多项式值、初始值、结果异或值的位序。

■ 输入数据倒序功能

通过设置 CRC_CTRL 寄存器中 DATA_REV 位，输入数据可选择三种倒序形式。

以原始数据 0x1A2B3C4D 为例：

1）按字节倒序：

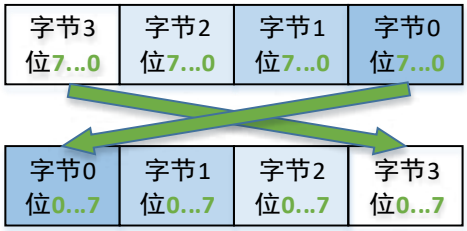
32 位数据被分成四组，组内完成位序颠倒。倒序后的数据为：0x58D43CB2

2）按半字倒序：

32 位数据被分成两组，组内完成位序颠倒。倒序后的数据为：0xD458B23C

3）按字倒序：

32 位数据被分成一组，组内完成位序颠倒，倒序后的数据为：0xB23CD458

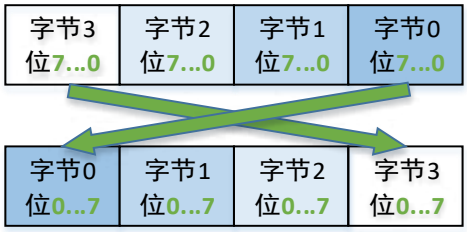


■ 输出数据倒序功能

通过将 CRC_CTRL 寄存器中 RSLT_REV 位置 1 可以将输出数据倒序。对输出数据来说，倒序形式为根据多项式长度按位进行倒序。

举例说明如下：

CRC32 计算结果 0x11223344 将被倒序成 0x22CC4488



CRC16 计算结果 0x4488 将被倒序成 0x1122

CRC8 计算结果 0x88 将被倒序成 0x11

CRC7 计算结果 0x44 将被倒序成 0x11。

■ 多项式/初始值/结果异或值倒序功能

通过将 CRC_CTRL 寄存器中相应控制位可以将多项式/初始值/结果异或值倒序。

对于多项式，初始值，结果异或值来说，倒序形式为根据多项式长度按位倒序，可参考输出数据倒序说明。与输出数据的区别就是 CRC7 和 CRC8 的倒序形式一样，都是按字节倒序。

注意：

由于 CRC 模块内部是按照倒序计算，所以用户在正常使用时需要将倒序参数都配置为 1 才能进行与常规 CRC 工具一致的的 CRC 运算，具体使用方式请参见配置流程说明。

47.3.2.5. 完全可编程多项式

CRC 多项式可以通过以下三处进行配置：

- 通过设置控制寄存器 CRC_CTRL 的 POLY_LEN 来选择多项式的长度(32/16/8/7)

- 通过设置控制寄存器 CRC_CTRL 的 POLY_REV 来选择是否进行多项式高低位倒序
- 通过写入 CRC_POLY 寄存器来配置多项式值

注意：

1）由于 CRC 模块内部是按照倒序计算，只有当 CRC_CTRL 的 POLY_REV 为 1 时，多项式寄存器中的值与参与运算的多项式值是一致的,即如果希望参与运算的多项式值为 0x4C11DB7，那么需要向多项式寄存器 CRC_POLY 中写入值 0x04C11DB7，并且设置 CRC_CTRL 的 POLY_REV 为 1，才能正确运算。

2）多项式的最高位默认为 1，不需要设置。比如多项式寄存器中的值为 0x04C11DB7，则对应的多项式为 poly = x32+x26+x23+x22+x16+x12+x11+x10+x8+x7+x5+x4+x2+x+1。

3）为实现可靠的 CRC 计算，CRC 计算期间不能实时更改多项式的值或大小。因此，如果正在执行 CRC 计算，则在更改多项式前，应用程序必须先复位 CRC，或者先执行 CRC_DATA 读操作。

4）默认的多项式值为 CRC-32（以太网）多项式：0x04C11DB7

47.3.3. 配置流程

1）设置控制寄存器 CRC_CTRL，选择多项式长度、有效数据长度以及输入数据、输出数据、初始值、结果异或值、CRC 结果是否倒序；

注意：由于 CRC 模块内部是按照倒序进行运算，所以用户需要将倒序参数都配置为 1 才能进行常规的 CRC 运算，即 POLY_REV, OUTXOR_REV, INIT_REV，DATA_REV，RSLT_REV 都配置为 1。此处配置与常规的 CRC 工具配置是相反的。

以多项式 0x04C11DB7 为例，参考配置如下：

参数配置	常规 CRC 软件配置	本模块配置
多项式(CRC32/MPEG-2)	0x04C11DB7	0x04C11DB7
宽度	32	32
初始值	0xFFFFFFFF	0xFFFFFFFF
输出异或值	0x00000000	0x00000000
输入数据倒序	不倒序	倒序
输出异或值倒序	不倒序	倒序
多项式倒序	不倒序	倒序
初始值倒序	不倒序	倒序
输出异或值倒序	不倒序	倒序

2）往多项式寄存器 CRC_POLY 写入多项式；

3）往初始值寄存器 CRC_INIT 写入初始值，如果初始值为 0，此步可省略；

4）往结果异或寄存器 CRC_OUTXOR 写入结果异或值，如果结果异或值为 0，此步可省略；

5）向 CRC_DATA 中依次写入 8/16/32 位 CRC 计算数据；

6）写完之后即可读 CRC_DATA，将一次返回 CRC 计算结果。

7）如果需要开始新的 CRC 运算，需通过设置寄存器 CRC_CTRL 的 RST 位来将 CRC_DATA 重置为 CRC_INIT 寄存器中的值。

47.4. 寄存器描述

47.4.1. 寄存器列表

CRC 寄存器基地址：0x40010C00

偏移	名称	复位值	描述
0x00	CRC_DATA		数据寄存器
0x04	CRC_CTRL		控制寄存器
0x08	CRC_INIT		初始值寄存器
0x0C	-		保留
0x10	CRC_OUTXOR		结果异或值寄存器
0x14	CRC_POLY		多项式寄存器
0x18	CRC_FDATA		独立数据寄存器

47.4.2. 数据寄存器(CRC_DATA: 00h)

位域	名称	属性	复位值	描述
31:0	DATA	RW	0	写：写入需要进行 CRC 校验计算的数据，如需要校验的数据是多个字节数据只需按顺序逐次写入 读：读出 CRC 计算结果，写入的数据无法再次读出，读操作返回的是上一次 CRC 计算的结果

47.4.3. 控制寄存器(CRC_CTRL: 04h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	PLOY _REV	RW	0	多项式是否进行高低位倒序 0：不倒序 1：倒序
9	OUTXOR_REV	RW	0	结果异或值是否进行高低位倒序 0：不倒序 1：倒序
8	INITIAL_REV	RW	0	CRC 初始值是否进行高低位倒序 0：不倒序 1：倒序
7	RSLT_REV	RW	0	CRC 计算结果是否进行高低位倒序 0：不倒序 1：倒序

6:5	DATA_REV	RW	0	CRC 计算数据是否进行高低位倒序 0：输入数据不倒序 1：输入数据按字节倒序 2：输入数据按半字倒序 3：输入数据按字倒序
4:3	PLOY_LEN	RW	0	多项式长度 0：32 位 1：16 位 2：8 位 3：7 位
2:1	DATA_LEN	RW	0	数据寄存器有效数据字节长度 0：1 个字节 1：2 个字节 2：3 个字节 3：4 个字节
0	RST	RW	0	写 1 复位 CRC_DATA 寄存器 CRC_DATA 寄存器将重新初始化为 CRC_INIT 寄存器中的值

47.4.4. 初始值寄存器(CRC_INIT: 08h)

位域	名称	属性	复位值	描述
31:0	INIT	RW	0	写入 CRC 初始值

47.4.5. 结果异或值寄存器(CRC_OUTXOR: 10h)

位域	名称	属性	复位值	描述
31:0	OUTXOR	RW	0	写入结果异或值

47.4.6. 多项式寄存器(CRC_POLY: 14h)

位域	名称	属性	复位值	描述
31:0	POLY	RW	0x04C11DB7	写入多项式值 同时需要配置控制寄存器中多项式长度位 如果多项式长度小于 32 位，则必须使用最低有效位编程多项式值

47.4.7. 独立数据寄存器(CRC_FDATA: 18h)

位域	名称	属性	复位值	描述
31:0	DATA	RW	0	独立数据寄存器位，这些位与 CRC 计算无关，可以给任何其他外设使用或用于其他目的。 此寄存器不受 CRC_CTRL 寄存器中 RST 位产生的 CRC 复位影响。

48. 高级加密算法（AES）

AES(Advanced Encryption Standard)，是美国国家标准与技术研究所用于加密电子数据的规范，该标准用来替代原先的 DES（Data Encryption Standard)，已经被多方分析且广为全世界所使用。AES 算法汇聚了设计简单、需要内存空间少、在所有的平台上运行良好、支持并行处理并且可以抵抗所有已知攻击等优点。经过五年的甄选流程，高级加密标准由美国国家标准与技术研究院 （NIST）于 2001 年 11 月 26 日发布于 FIPS PUB 197，并在 2002 年 5 月 26 日成为有效的标准。

48.1. 主要特性

- 符合联邦信息处理标准出版物（FIPS PUB 197，2001 年 11 月 26 日）规定的高级加密标准(AES)
- 支持 AES 加密和解密运算；
- 支持 ECB/CBC/CTR 模式；
- 支持 128/ 192/ 256 bit 密钥长度；
- 支持数据输入和输出 SWAP 模式，即大小端可配置；
- 支持在 CBC、CTR 模式下使用的 4 × 32 位初始化向量（IV）

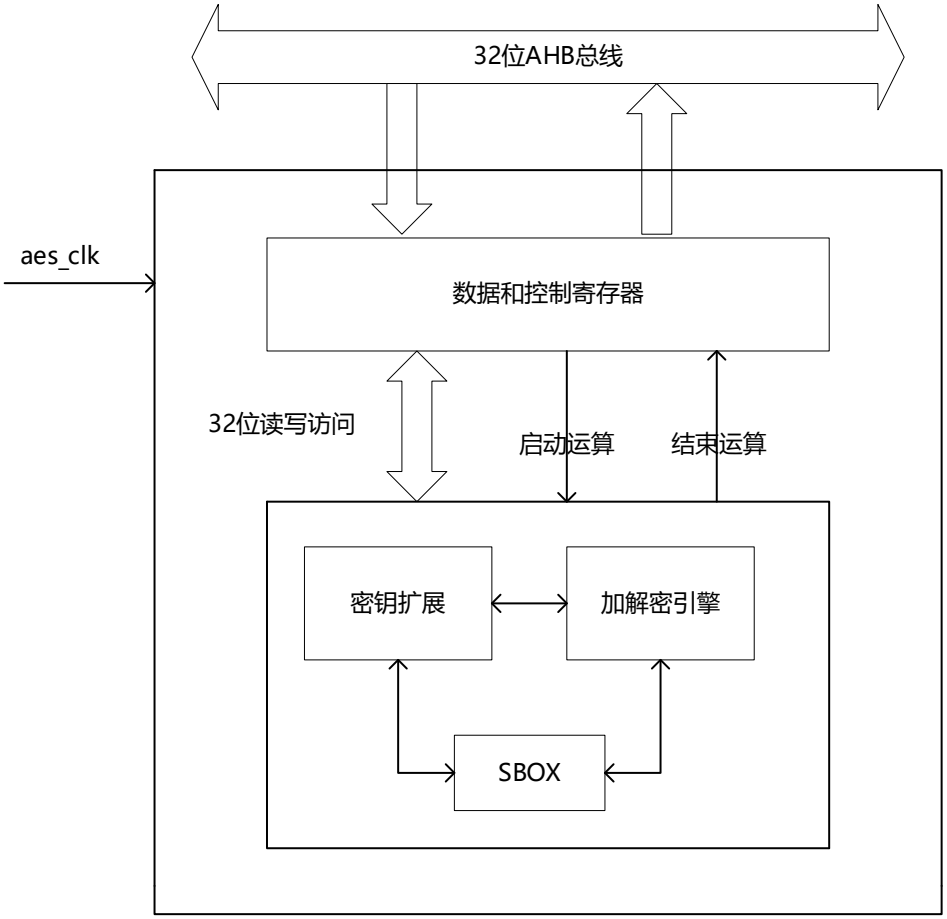
48.2. 功能说明及框图

AES 为分组密码算法，即把明文分成一组一组的，每组长度相等，每次加密一组数据，直到加密完整个明文。在 AES 标准规范中，分组长度只能是 128 位，也就是说，每个分组为 16 个字节（每个字节 8 位)。密钥的长度可以使用 128 位、192 位或 256 位。

由于 AES 算法使用块密码，因此，加密前需对不完整的输入数据块进行填充（将额外的位附加到数据串的尾端)，解密后需要丢弃填充位。AES 硬件不处理填充操作，需要通过软件进行处理。

AES 加解密处理器框图如下：

图 48-1 AES 处理流程图



AES 模块处理 128 位块所需的周期如下：

密钥长度	ECB	CBC	CTR
128b	44	44	44
192b	52	52	52
256b	60	60	60

48.3. 加解密模式

48.3.1. AES-ECB 模式

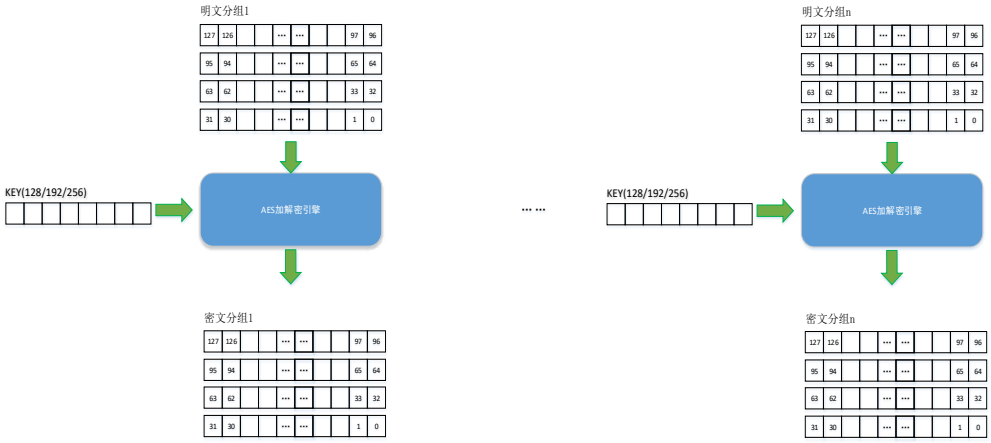
AES-ECB 模式加密流程如下：

1）将明文按照 128bit 长度进行分组，不足 128bit 长度的分组需按照一定规则进行填充。

2）将明文依次输入 AES 加密模块使用 128 位、192 位或 256 位密钥进行加密运算。

3）将每个分组对应得到的密文分组进行拼接并去掉填充，得到密文。

下图为 AES-ECB 模式加密示意图



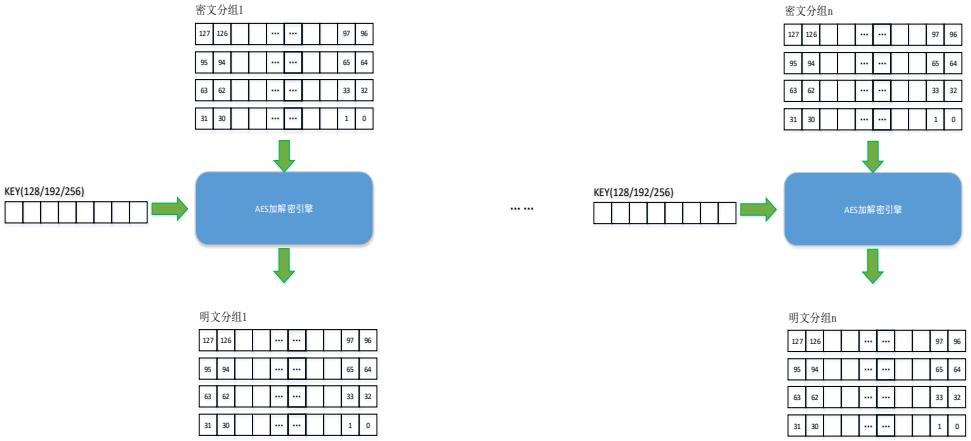
AES-ECB 模式解密流程如下：

1）将密文按照 128bit 长度进行分组，不足 128bit 长度的分组需按照一定规则进行填充。

2）将密文依次输入 AES 加密模块使用 128 位、192 位或 256 位密钥进行解密运算。

3）将每个分组对应得到的明文分组进行拼接并去掉填充，得到明文。

下图为 AES-ECB 模式解密示意图



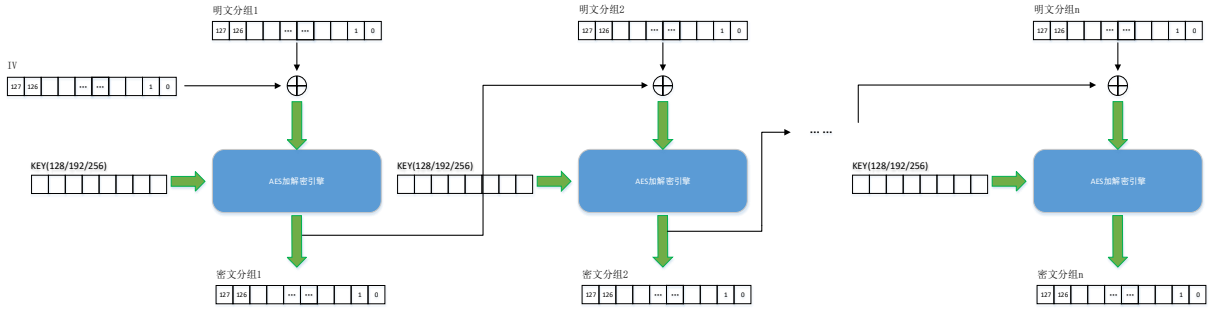
48.3.2. AES-CBC 模式

AES-CBC 模式加密流程如下：

1）将明文按照 128bit 长度进行分组，不足 128bit 长度的分组需按照一定规则进行填充。

2）初始向量 IV 和第一组明文异或后依次输入 AES 加密模块使用 128 位、192 位或 256 位密钥进行加密运算得到密文，上一组的密文用作下一组的 IV 输入。CBC 加密会不断将后续密文块和明文块链接到一起，直到完成最后一个明文块加密。

3）将每个分组对应得到的密文分组进行拼接并去掉填充，得到密文。

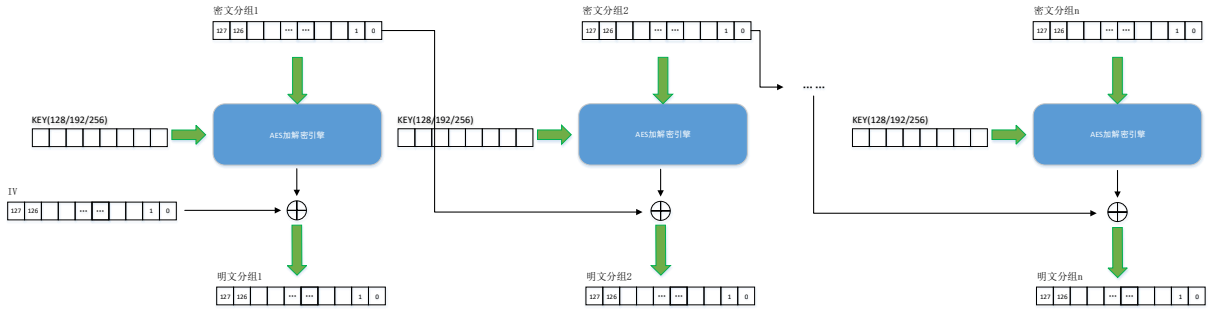


AES-CBC 模式解密流程如下：

1）将密文按照 128bit 长度进行分组，不足 128bit 长度的分组需按照一定规则进行填充。

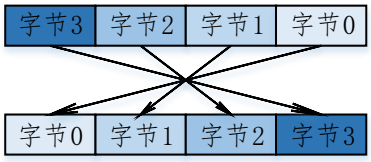
2）将第一组密文输入 AES 加密模块使用 128 位、192 位或 256 位密钥进行解密运算，运算结果与初始向量 IV（必须与加密相同）异或得到明文，上一组的密文用作下一组的 IV 输入。CBC 解密将以此方式继续进行，直到完成最后一个密文块解密。

3）将每个分组对应得到的明文分组进行拼接并去掉填充，得到明文。



48.4. SWAP 模式

AES 运算中，SWAP 模式作用是以字节为单位，将一个字内的字节高低位置进行交换，字节内相对比特位置保持不变，如下图所示。



SWAP 模式对密钥、初始向量、输入数据、输出数据同时有效。

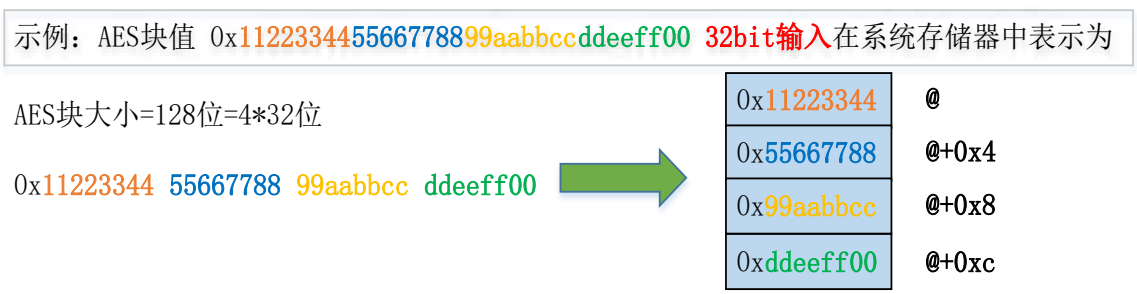
48.5. 数据输入方式

AES 模块数据的输入方式为：高位字数据存储在数组的低位元素中，每个字以大端方式存放，举例如下：

设待加密数据为：0x112233445566778899aabbccddeeff00

如果输入 AES 模块进行加解密运算的数据为 32bit 数组，则输入 AES 模块进行运算的数组为：

uint32_t plain_text[4] = {0x11223344, 0x55667788, 0x99aabbcc, 0xddeeff00};

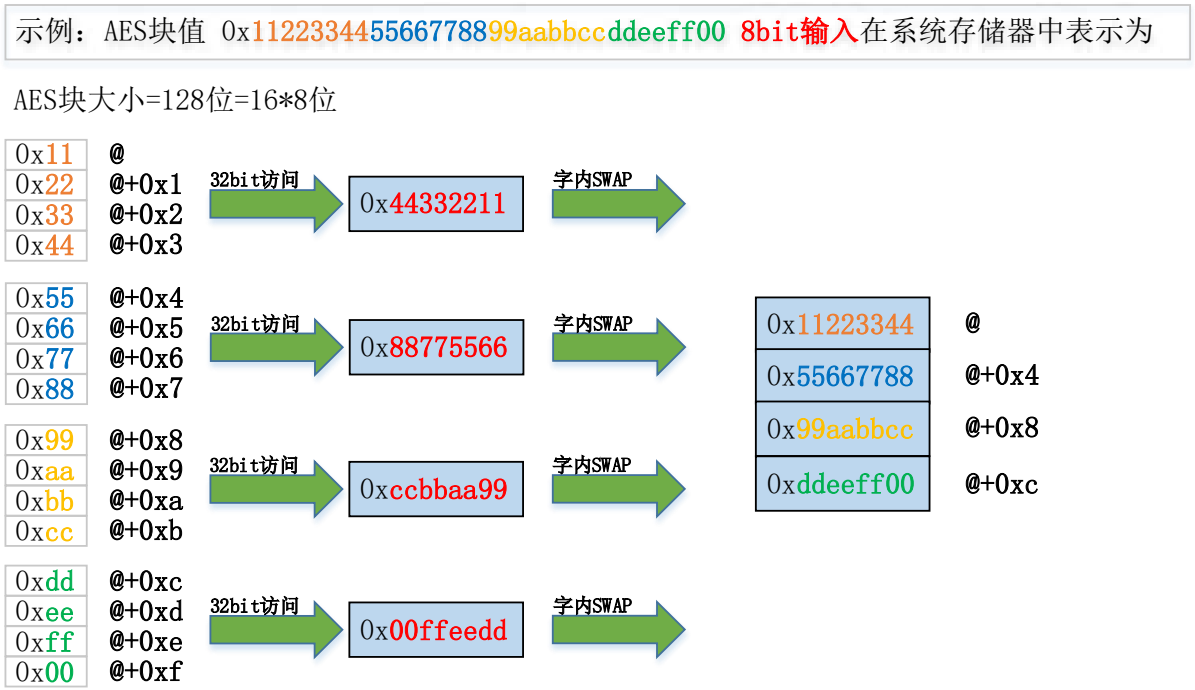


如果输入 AES 模块进行加解密运算的数据为 8bit 数组，则输入 AES 模块进行运算的数组为：

```
uint8_t plain_text[16] =
{0x11,0x22,0x33,0x44,0x55,0x66,0x77,0x88,0x99,0xaa,0xbb,0xcc,0xdd,0xee,0xff,0x00};
```

为了获得与 32bit 输入相同的运算结果，应该开启 SWAP 模式。

在 8bit 输入方式中，使能 SWAP 模式，进入 AES 模块运算数据变为如下图所示，与 32bit 输入方式相同。



48.6. 数据输出方式

输出数据格式与输入相同。

48.7. 库操作说明

调用 AES 算法库进行加解密操作步骤如下：

1）调用 hal_aes.lib 库中的 HAL_AES_SetKey 函数，配置密钥起始地址，密钥长度(128/192/256)以及大小端

对调(SWAP)是否使能，对 AES 模块进行初始化；

2）调用 hal_aes.lib 库中的 HAL_AES_Crypt 函数，输入待加密(解密)数据的起始地址，加(解)密块(128 bit)个数，选择加解密模式，CBC 模式初始向量的起始地址以及安全模式等参数，进行加(解)密运算，从配置的存放输出结果的起始地址处读取加(解)密结果。

49. OTFDEC 加解密模块（AES_SPI）

49.1. 概述

OTFDEC（On The Fly DECryption），即在线解密。代码和数据加密存储在片外 FLASH，运行时实时解密片外 FLASH 的密文代码和数据。

通过 OTFDEC 的正确配置，从外部 FLASH 执行密文代码，对 CPU 来说，就像是执行片内 FLASH 里存储的明文代码一样。

OTFDEC 模块支持 SPI 和 OSPI 外部 FLASH。外部 FLASH 要工作在内存映射模式下。

加密操作不是实时的，需要先加密好，再写入 FLASH。

OTFDEC 模块的加解密，采用 AES_CTR 算法。OTFDEC 模块也可以作为通用的 AES 算法引擎使用。

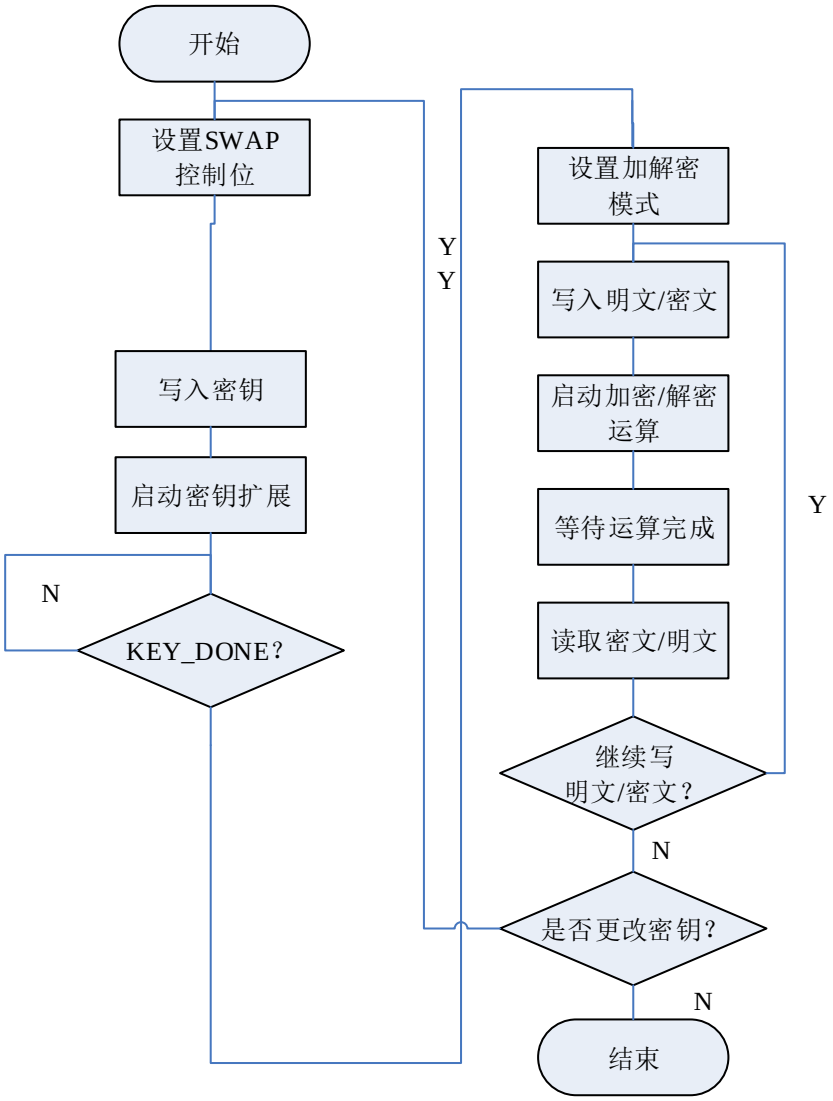
49.2. 主要特性

- 支持 AES 加密和解密运算
- 数据输入和输出支持 SWAP 模式，即大小端可配置
- 支持加解密过程中硬件虚拟 AES 运算，且虚拟运算轮数可配置
- 支持 AES128/192/256
- 支持 ECB/CTR 模式
- 支持 OTFDEC 功能(SPI7/SPI8/OSPI2），并支持三段分别进行 OTFDEC
- 加密运算不需要秘钥扩展，解密运算需要秘钥扩展

49.3. 使用流程

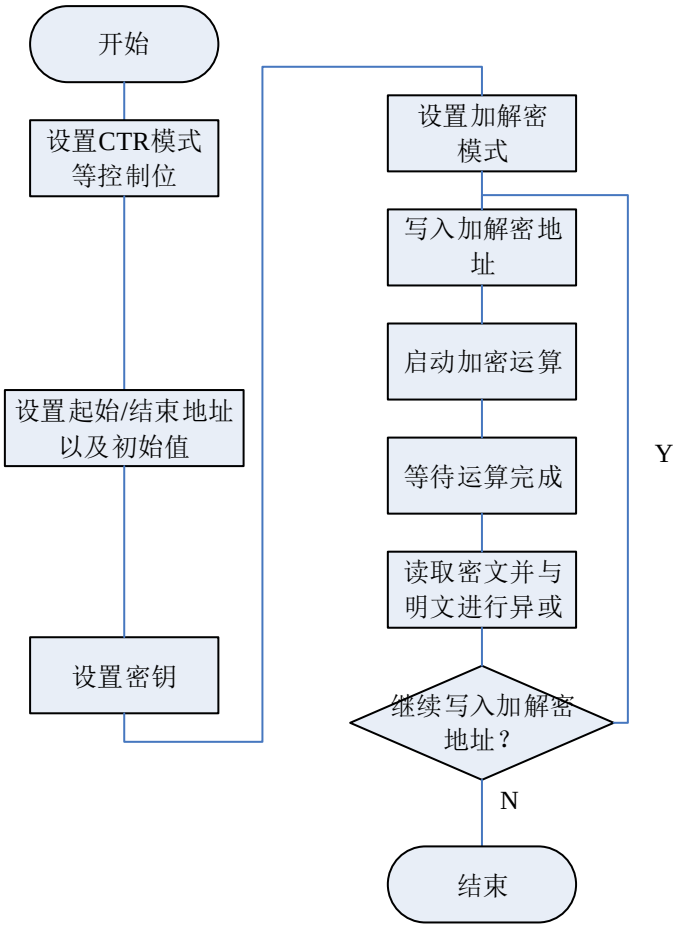
49.3.1. AES 模块加解密流程

图 49-1 AES 加解密流程图



49.3.2. AES 模块 CTR 模式加解密流程

图 49-2 AES CTR 模式加密流程图



49.3.3. OTFDEC 区域 1 使能流程

- 往 UID 寄存器写入芯片 UID
- 往 RANDOM0 和 RANDOM1 寄存器输入随机数
- 往 CTR 加解密结束地址寄存器输入区域 1 的结束地址
- 往 OTFDEC 控制寄存器写 1 使能区域 1 的 OTFDEC 功能
- 设置 OTFDEC_SPI 控制寄存器选择哪个 SPI 进行 OTFDEC

49.3.4. OTFDEC 区域 2 使能流程

- 往区域 2 密钥输入寄存器写入密钥
- 往 RANDOM2 和 RANDOM3 寄存器输入随机数
- 往 CTR 加解密结束地址 2 寄存器输入区域 2 的结束地址
- 设置 OTFDEC_SPI 控制寄存器选择哪个 SPI 进行 OTFDEC 以及区域 2 的 OTFDEC 使能

49.3.5. OTFDEC 区域 3 使能流程

- 往区域 3 密钥输入寄存器写入密钥
- 往 CTR 加解密结束地址 3 寄存器输入区域 3 的结束地址，也可以使用默认值
- 设置 OTFDEC_SPI 控制寄存器选择哪个 SPI 进行 OTFDEC 以及区域 3 的 OTFDEC 使能

49.4. AES_SPI 寄存器描述

49.4.1. 寄存器列表

AES_SPI 模块基地址为 0x50061000

偏移	名称	复位值	描述
0x00	AES_SPI_DATAIN	0x00000000	数据输入寄存器
0x04	AES_SPI_KEYIN	0x00000000	密钥输入寄存器
0x08	AES_SPI_OTFDEC_CTRL	0x00000000	OTFDEC 控制寄存器
0x0C	AES_SPI_CTRL[31:0]	0x00000000	控制寄存器
0x10	AES_SPI_STATE[31:0]	0x00000000	状态寄存器
0x14	AES_SPI_DATAOUT	0x00000000	数据输出寄存器
0x18	AES_SPI_START_ADDR	0x00000000	CTR 加解密起始地址寄存器
0x1C	AES_SPI_END_ADDR	0xFFFFFFFF	CTR 加解密结束地址寄存器
0x20	AES_SPI_INI_DATA	0x5A5A5A5A	CTR 加解密初始值寄存器
0x24	AES_SPI_ADDR	0x00000000	CTR 加解密地址寄存器
0x28	AES_SPI_UID	0x00000000	芯片唯一序列号 UID
0x2C	AES_SPI_RANDOM0	0x6689E4A1	随机数 RANDOM0
0x30	AES_SPI_RANDOM1	0xd5F37896	随机数 RANDOM1
0x38	AES_SPI_OTFDEC_SPI_CTRL	0x00000000	OTFDEC_SPI 控制寄存器
0x3C	AES_SPI_END_ADDR2	0xFFFFFFFF	CTR 加解密结束地址 2 寄存器
0x40	AES_SPI_END_ADDR3	0xFFFFFFFF	CTR 加解密结束地址 3 寄存器
0x44	AES_SPI_RANDOM2	0x4BE0EF4F	随机数 RANDOM2
0x48	AES_SPI_RANDOM3	0x90D6CDD4	随机数 RANDOM3
0x4c	AES_SPI_KEYIN1	0x00000000	区域 1 密钥输入寄存器
0x50	AES_SPI_KEYIN2	0x00000000	区域 2 密钥输入寄存器
0x54	AES_SPI_KEYIN3	0x00000000	区域 3 密钥输入寄存器

49.4.2. 数据输入寄存器(AES_SPI_DATAIN: 00h)

位域	名称	属性	复位值	描述

31:0	DATAIN	WO	0x00000000	32 位寄存器，用来存放明文或密文的数据。需要连续写入 4 次，组成 128 位域数据流。

49.4.3. 密钥输入寄存器（AES_SPI_KEYIN: 04h)

位域	名称	属性	复位值	描述
31:0	KEYIN	WO	0x00000000	32 位寄存器，用来存放密钥数据。需要连续写入 4 次，组成 128 位域密钥。

49.4.4. OTFDEC 控制寄存器(AES_SPI_OTFDEC_CTRL: 08h)

位域	名称	属性	复位值	描述
31:3	-	-	-	保留
2	EFUSE_KEY_SEL	RW	0	EFUSE 密钥选择位 0 = 选择使用 EFUSE 密钥 1 = 选择不使用 EFUSE 密钥
1	OTFDEC_KEY_SEL	RW	0	OTFDEC 密钥选择位 0 = 选择 MAINKEY+UID_RANDOM 1 = 选择 USERKEY+UID+RANDOM
0	OTFDEC_EN	RW	0	OTFDEC 使能位 0 = 禁止 1 = 使能

注意：该寄存器复位后只能写一次

49.4.5. 控制寄存器(AES_SPI_CTRL: 0Ch)

位域	名称	属性	复位值	描述
31:18	-	-	-	保留
17	ADDR_MUX	RW	0	OTFDEC 加密地址高 5 位选择 0：不选择 1：选择
16	RANDOM12_UP	WO	0	随机数 12 更新启动位 1：写 1 启动更新。读该位域总是返回 0。 0：写 0 不起作用。读该位域总是返回 0。
15	-	-	-	保留
14	ADDR_AUTO	RW	0	CTR 模式加密时 IV 地址选择，仅当 IV 来源为 CTR 加解密地址寄存器时有效 0 = 来自 CTR 加解密地址寄存器。 1 = 第一次来自 CTR 加解密地址寄存器，之后则地址自动加 0x10。
13	AES_MODE	RW	0	AES 模式选择 0：禁止。 1：使能。

12	CTR_EN	RW	0	CTR 模式使能 0：禁止 1：使能
11:9	VROUND	RW	0	虚拟运算四位数设置 000：2 001：4 010：8 011：16 100：32 101：64 其它：保留
8	VAES_EN	RW	0	虚拟运算使能 0：禁止 1：使能
7:6	KEY_LEN	RW	0	密钥长度 00：AES_128 01：AES_192 10：AES_256 其它：保留
5				保留
4	SWAP	RW	0	数据输入输出 SWAP 模式 1 = SWAP 模式使能。 0 = SWAP 模式禁止。
3	INT_EN	RW	0	中断使能位。 1 = 中断使能。 0 = 中断禁止。
2	CRYPT	RW	0	加密/解密指示位。 1 = 进行解密运算。 0 = 进行加密运算。
1	KEY_START	WO	0	密钥扩展运算启动位 1：写 1 启动密钥扩展运算。读该位域总是返回 0。 0：写 0 不起作用。读该位域总是返回 0。
0	CRYPT_START	WO	0	加解/解密运算启动位 1：写 1 启动加解/解密运算。读该位域总是返回 0。 0：写 0 不起作用。读该位域总是返回 0。

49.4.6. 状态寄存器(AES_SPI_STATE: 10h)

位域	名称	属性	复位值	描述
31:3	-	-	-	保留

2	KEY1_FLAG	RO	0	用户输入密钥是否与 efuse 中一样 1 = 一样。 0 = 不一样。
1	KEY_DONE	RO	0	AES 密钥扩展完成位 1 = AES 密钥扩展完成，对该位写 1，则清除标志位。 0 = AES 密钥扩展没有完成。
0	CRYPT_DONE	RO	0	AES 运算完成位 1 = AES 运算完成，对该位写 1，则清除标志位。如果中断使能，该位置位时，同时产生中断；向该位写 1，则清除中断。 0 = AES 运算没有完成。

49.4.7. 数据输出寄存器(AES_SPI_DATAOUT: 14h)

位域	名称	属性	复位值	描述
31:0	DATAOUT	RO	0x00000000	结果寄存器用于存放加解密的结果数据，只读。

49.4.8. CTR 加解密起始地址寄存器(AES_SPI_START_ADDR: 18h)

位域	名称	属性	复位值	描述
31:0	START_ADDR	WO	0x00000000	32 位寄存器，用来存放 CTR 加解密时的起始地址。

49.4.9. CTR 加解密结束地址寄存器(AES_SPI_END_ADDR: 1Ch)

位域	名称	属性	复位值	描述
31:0	END_ADDR	WO	0xFFFFFFFF	32 位寄存器，用来存放 CTR 加解密时的结束地址。

49.4.10. CTR 加解密初始值寄存器(AES_SPI_INI_DATA: 20h)

位域	名称	属性	复位值	描述
31:0	INI_DATA	WO	0x5A5A5A5A	32 位寄存器，用来存放 CTR 加解密时的初始值。

49.4.11. CTR 加解密地址寄存器(AES_SPI_ADDR: 24h)

位域	名称	属性	复位值	描述
31:0	ADDR	WO	0x00000000	32 位寄存器，用来存放 CTR 加解密时的地址。

49.4.12. 芯片唯一序列号输入寄存器(AES_SPI_UID: 28h)

位域	名称	属性	复位值	描述
31:0	UID	RW	0x00000000	32 位寄存器，用来存放 UID。需要连续写入 4 次，组成 128 位域数据流。上电写入 4 次之后就不能再写了。连续读 4 次，返回 128 位域数据流

49.4.13. 随机数 0 寄存器(AES_SPI_RANDOM0: 2Ch)

位域	名称	属性	复位值	描述
31:0	RANDOM0	RW	0x6689e4a1	32 位寄存器，用来存放 RANDOM0。上电后只能写一次。 当控制寄存器中的随机数 12 更新位写 1 则会更新该寄存器。

49.4.14. 随机数 1 寄存器(AES_SPI_RANDOM1: 30h)

位域	名称	属性	复位值	描述
31:0	RANDOM1	RW	0xd5f37896	32 位寄存器，用来存放 RANDOM1。上电后只能写一次。 当控制寄存器中的随机数 12 更新位写 1 则会更新该寄存器。

49.4.15. OTFDEC_SPI 控制寄存器(AES_SPI_OTFDEC_SPI_CTRL: 38h)

位域	名称	属性	复位值	描述
31:6	-	-	-	保留
5	RANDOM34_UP	WO	0	随机数更新启动位 1：写 1 启动更新。读该位域总是返回 0。 0：写 0 不起作用。读该位域总是返回 0。
4	OTFDEC3_EN	RW	0	OTFDEC3 第 3 个区域使能位 0 = 禁止 1 = 使能
3	OTFDEC2_EN	RW	0	OTFDEC2 第 2 个区域使能位 0 = 禁止 1 = 使能
2	OTFDEC_OSPI2_SEL	RW	0	OTFDEC_OSPI2 选择使能位 0 = 禁止 1 = 选择 OSPI2 进行 OTFDEC
1	OTFDEC_SPI8_SEL	RW	0	OTFDEC_SPI8 选择使能位 0 = 禁止 1 = 选择 SPI8 进行 OTFDEC
0	OTFDEC_SPI7_SEL	RW	0	OTFDEC_SPI7 选择使能位 0 = 禁止 1 = 选择 SPI7 进行 OTFDEC

49.4.16. CTR 加解密结束地址 2 寄存器(AES_SPI_END_ADDR2: 3Ch)

位域	名称	属性	复位值	描述
31:0	END_ADDR2	WO	0xFFFFFFFF	32 位寄存器，用来存放 CTR 第 2 块区域加解密时的结束地址。

49.4.17. CTR 加解密结束地址 3 寄存器(AES_SPI_END_ADDR3: 40h)

位域	名称	属性	复位值	描述
31:0	END_ADDR3	WO	0xFFFFFFFF	32 位寄存器，用来存放 CTR 第 3 块区域加解密时的结束地址。

49.4.18. 随机数 2 寄存器(AES_SPI_RANDOM2: 44h)

位域	名称	属性	复位值	描述
31:0	RANDOM3	RW	0x4be0ef4f	32 位寄存器，用来存放 RANDOM3。 当 OTFDEC_SPI 控制寄存器中的随机数 34 更新位写 1 则会更新该寄存器。

49.4.19. 随机数 3 寄存器(AES_SPI_RANDOM3: 48h)

位域	名称	属性	复位值	描述
31:0	RANDOM4	RW	0x90d6cdd4	32 位寄存器，用来存放 RANDOM4。当 OTFDEC_SPI 控制寄存器中的随机数 34 更新位写 1 则会更新该寄存器。

49.4.20. 区域 1 加密密钥输入寄存器(AES_SPI_KEYIN1: 4Ch)

位域	名称	属性	复位值	描述
31:0	KEYIN1	WO	0x00000000	32 位寄存器，用来存放区域 1 加密密钥数据。需要连续写入 4 次，组成 128 位位域密钥。

49.4.21. 区域 2 密钥输入寄存器(AES_SPI_KEYIN2: 50h)

位域	名称	属性	复位值	描述
31:0	KEYIN2	WO	0x00000000	32 位寄存器，用来存放区域 2 密钥数据。需要连续写入 4 次，组成 128 位位域密钥。

49.4.22. 区域 3 密钥输入寄存器(AES_SPI_KEYIN3: 54h)

位域	名称	属性	复位值	描述
31:0	KEYIN3	WO	0x00000000	32 位寄存器，用来存放区域 3 密钥数据。需要连续写入 4 次，组成 128 位位域密钥。

```
__IO uint32_t ADDR1;            //0x58
__IO uint32_t DATAOUT1;         //0x5C
```

50. 安全散列算法（SHA）

50.1. 概述

安全散列算法（英语：Secure Hash Algorithm，缩写为 SHA）是一个密码散列函数家族，是 FIPS 所认证的安全散列算法。它把任意长度数字的输入通过散列算法变换成和输入消息对应的，长度固定的输出，该输出就是散列值（又称消息摘要）。这种转换是一种压缩映射，也就是，散列值的空间通常远小于输入的空间，不同的输入可能会散列成相同的输出。由于不是一对一的映射，不可能从散列值来确定唯一的输入值，难以找到逆向规律。

简单的说 HASH 就是一种将任意长度的消息压缩到某一固定长度的消息摘要的函数。SHA-1 可以生成一个被称为消息摘要的 160 位（20 字节）散列值，散列值通常的呈现形式为 40 个十六进制数。SHA256 是 SHA2 的一种，可产生 256 位的哈希值，较 SHA1 更加的安全。

50.2. 主要特性

- 支持 SHA1/SHA256 算法
- 适合于数据验证应用，符合以下标准：
  - FIPS PUB 180-1（联邦信息处理标准出版物 180-1）安全散列标准规范（SHA-1）
  - FIPS PUB 180-2（联邦信息处理标准出版物 180-2）安全散列标准规范（SHA-224 和 SHA-256）
- 可自动填充来补足输入位串，从而适应 512 位（16×32 位）的摘要最小块大小
- 使用 SHA-1（或 SHA-256）算法处理一个 512 位数据块需要 3868（或 4185）个时钟周期
- 32 位结果输出寄存器，SHA1 需读 5 次，SHA256 读 8 次

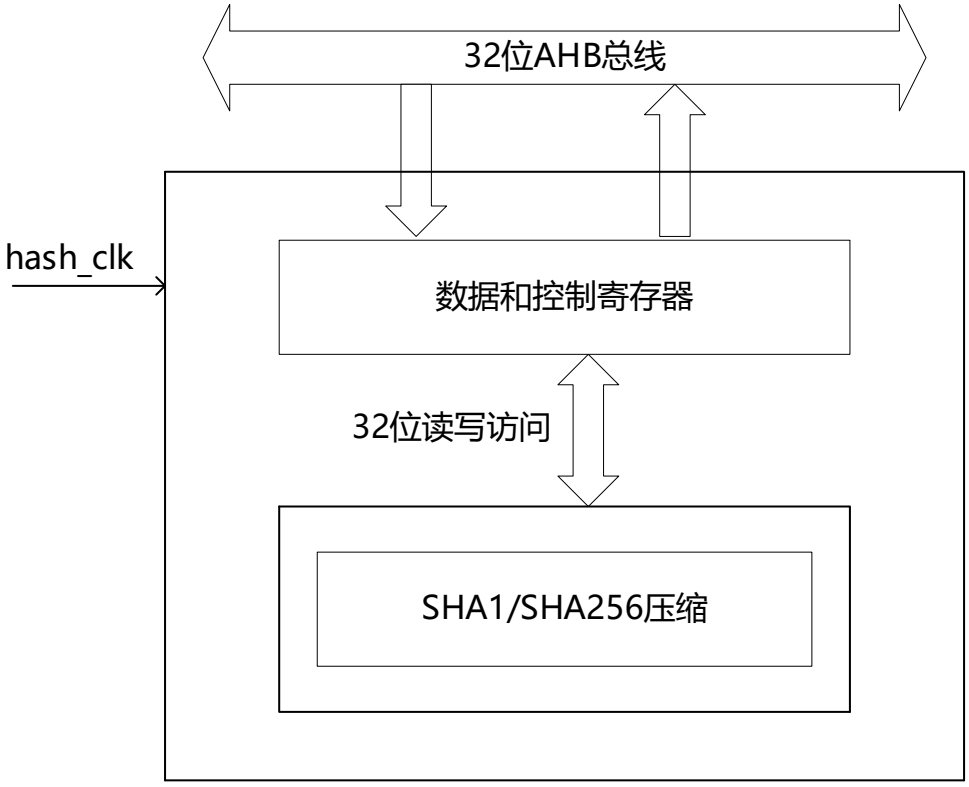
50.2.1. 功能说明及框图

Hash 算法也是现代密码体系中的一个重要组成部分。由于非对称算法的运算速度较慢，所以在数字签名协议中，单向散列函数扮演了一个重要的角色。

HASH 模块完全兼容由 FIPS PUB 180-1 标准（SHA1）、FIPS PUB 180-2 标准（SHA-224、SHA-256）定义的安全散列算法。对于每个算法，HASH 计算消息或数据文件的压缩表示。具体来讲，对于任意长度的输入位串，SHA-1 和 SHA-256 算法将分别生成一个 160 位和 256 位的输出位串，称为消息摘要。然后可以通过数字签名算法来处理此消息摘要，以便生成或验证消息的签名。

对消息摘要而不是对消息签名通常可提高处理的效率，因为消息摘要通常比消息要小得多。数字签名的验证程序和数字签名的创建程序必须使用相同散列算法。要找出某个与给定消息摘要对应的消息，或找出两个生成相同消息摘要的不同消息，在计算层面无法实现，所以 SHA-1 和 SHA-256 算法安全可靠。对传输中的消息进行任何更改都极有可能产生不同的消息摘要，从而导致签名验证失败。

图 50-1 SHA 结构框图



50.2.2. 数据输入

要由 HASH 处理的消息（或数据文件）应视为位串。根据 FIPS PUB 180-1 和 180-2 标准，该消息位串从左到右增长（十六进制字采用“大端”约定来表示），从而使得在每个字内，最高有效位存储在最左侧位的位置上。以位串表示为“01100001 01100010 01100011”的消息串“abc”为例，其 32 位字表示为 0x00636261，8 位字表示为 0x61626300。

本芯片提供的 HASH 库使用 8bit 数据流方式输入，涉及到 HASH 运算的数据格式都是大端模式，举例如下：

若消息：A = 0x1122334455667889900;

则输入 HASH 模块运算数据为：

A[0]=0x11	0x11	@
A[1]=0x22	0x22	@+0x1
A[2]=0x33	0x33	@+0x2
A[3]=0x44	0x44	@+0x3
A[4]=0x55	0x55	@+0x4
A[5]=0x66	0x66	@+0x5
A[6]=0x77	0x77	@+0x6
A[7]=0x88	0x88	@+0x7
A[8]=0x99	0x99	@+0x8
A[9]=0x00	0x00	@+0x9

50.2.3. 消息填充

因为消息的长度（位数）可以是任何整数值，而 HASH 算法每次处理 512-bit（16 个字）长度的报文分组序列，因此在运算之前需要检查消息长度附加填充比特，一般执行以下两个步骤：

1）首先对消息进行填充使其长度与 448 模 512 同余（长度=448mod512），填充的比特数范围是 1 到 512，填充比特串的最高位为 1，其余位为 0。
2）附加消息长度值。将用 64-bit 表示的原始消息（填充前）的位长度附加在步骤 1 的结果后（低位字节优先）。

经过这上述填充操作后生成的消息长度为=L(原消息长度) + 1 + k(填充 0 的个数) + 64(原消息位长度)。填充后消息长度是 512 位的整数倍。

FIPS PUB180-2 提供的示例如下：

假定原始消息是采用 ASCII 二进制编码格式的“abc”，长度 L=24：

字节 0 字节 1 字节 2 字节 3

01100001 01100010 01100011 UUUUUUUU

对上述位串进行填充，最高位为 1，得到：

01100001 01100010 01100011 1UUUUUUU

位串原始长度 L = 24，填充后以上位串中的位数是 25，继续追加 423 个“0”位，使得长度增加至 448 位，得到如下结果(十六进制，大端模式)：

61626380 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000

最后追加 64bit 的消息长度值，即 00000000 00000018。因此，填充后最终得到的消息为(十六进制，大端模式)：

61626380 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000000

00000000 00000000 00000000 00000018

50.3. 库函数操作说明

本芯片提供的 HASH 库包括 SHA1/SHA256 两种散列算法，这两种散列算法的调用都提供整体运算和迭代运算两种模式。

■ 整体运算流程如下：

1）调用 hal_sha1lib/hal_sha256.lib 驱动库中的 HAL_SHA1_Hash/HAL_SHA256_Hash 函数，输入待压缩数据的起始地址及字节长度，驱动库自动完成数据填充并生成散列值

2）运算结束后从摘要值的起始地址处读取摘要值(sha1 摘要长度 160bit，sha256 摘要长度 256bit)

■ 迭代运算流程如下：

1）调用 hal_sha1lib/hal_sha256.lib 驱动库中的 HAL_SHA1_Init/HAL_SHA256_Init 初始化 Hash 运算环境变量 SHA1_CTX，SHA256_CTX

2）调用 hal_sha1lib/hal_sha256.lib 驱动库中的 HAL_SHA1_Update/HAL_SHA256_Update 函数，输入运算环境变量、待压缩数据的起始地址及字节长度，进行迭代运算

3）调用 hal_sha1lib/hal_sha256.lib 驱动库中的 HAL_SHA1_Final/HAL_SHA256_Final ,输入运算环境变量，驱动库自动完成数据填充并生成散列值

4）运算结束后从摘要值的起始地址处读取摘要值(sha1 摘要长度 160bit，sha256 摘要长度 256bit)

51. CORDIC 加速算法（CORDIC）

51.1. CORDIC 概述

CORDIC（Coordinate Rotation Digital Computer）坐标旋转数字计算模块，提供 sin/cos/atan2/sinh/cosh/atanh/ln/sqrt 等数学运算的硬件加速。CORDIC 可以用于加速指纹算法也可以用于电机控制、测量、信号处理等（主要是三角函数）应用。

CORDIC 算法是 J.D.Volder1 于 1959 年首次提出，主要用于三角函数、双曲线、指数、对数的计算。该算法通过基本的加和移位运算代替乘法运算，使得矢量的旋转和定向的计算不再需要三角函数、乘法、开方、反三角、指数等函数。与软件实现相比，它加快了这些函数的计算速度，从而允许处理器用较低的频率工作，或者减少 CPU 占用时间，以便处理器执行其他任务。

51.2. 主要特性

- 24 位 CORDIC 旋转引擎；
- 支持圆形和双曲线模式；
- 支持 sin/cos/atan2/sinh/cosh/atanh/ln/sqrt 等函数；
- 支持 1~8 轮可编程精度

51.3. 功能说明

CORDIC 是一种高效的连续逐次逼近算法，其核心是利用加法和移位的迭代操作去替代复杂的乘法运算，从而非常有利于硬件实现。

CORDIC 算法一般用于计算三角和双曲线函数。在三角（圆）模式中，不断旋转单位矢量[1,0]，并减小旋转角度直到旋转角度的累积和等于输入角度θ，从而得到输入角度θ的正弦和余弦值(旋转矢量的 x 和 y 笛卡尔分量分别对应于θ的余弦和正弦)。反过来，可以通过不断地减小角度并旋转矢量[x,y]，直到得到单位矢量[1,0]，旋转角度的累积和等于向量[x,y]的角度值（对应于 y/x 的反正切值）。CORDIC 算法也可以用沿着双曲线的步长代替连续的圆形旋转用于计算双曲函数（sinh/cosh/atanh）。其他函数可以从上述基本函数导出。

下表列出了 CORDIC 模块支持的函数。

函数	参数 1	参数 2	结果 1	结果 2
CosineSin	θ	--	$\cos\theta$	$\sin\theta$
AtanSqrt	x	y	$\text{atan}2(y,x)$	$\sqrt{x^2+y^2}$
CoshSinh	θ	--	$\cosh\theta$	$\sinh\theta$
Atanh	θ	--	$\text{atanh}\theta$	--
Ln	x	--	$Ln(x)$	--
Sqrt	x	--	$\sqrt{x}$	--

有些函数同时产生两个输出。这是因为这两个结果是在同一个计算过程中同步生成的。

指数函数 $\exp(x)$ 可以由 $\sinh(x)$ 和 $\cosh(x)$ 的和来获得。以 N 为基数的对数 $\log_N(x)$ 可以通过将 $\ln(x)$ 乘以常数 K 来导出，其中 K = $\frac{1}{\ln(N)}$ 。

对于某些函数（Ln、Sqrt），可应用比例因子（scale）将函数的输入范围扩展到 Q31 格式支持的最大值[-1，1]之外。对于所有三角（圆）函数，缩放因子必须设置为 0，对于双曲函数缩放因子必须设置为为 1，具体参数设置请参考每个函数的参数说明。

51.4. 数据格式

■ Q31 数据格式(定点表示法)

CORDIC 模块统一采用 Q31 格式输入输出。Q31 数据格式是一种定点表示法。Q31 格式中，数字由 1 个符号位和 31 个二进制小数位表示，因此数字范围是-1(0x80000000)到 1-231(0x7FFFFFFF)。

如果应用中是使用 float 类型数据，那么在调用 Cordic 算法时需要将 float 类型转化成 Q31 格式数据进行运算，并将得到的 Q31 结果进行处理后，再转成 float 类型给应用程序使用，具体参数设置请参考每个函数的参数说明。

下面例子展示了浮点数和 Q31 格式相互转换的过程：

浮点数 0.5 转换成 Q31 格式 ：0.5* 231=1073741824=0x40000000，

Q31 格式 0x40000000 转换成浮点数 ：0x40000000/231 =0.5

51.4.1. 缩放因子

CORDIC 模块中的部分算法需要引入缩放因子(Scale)。缩放因子能够扩展输入参数范围以覆盖算法支持的范围，避免了输入、输出及内部寄存器饱和。

如果需要使用缩放因子，需要外部软件计算出相应缩放因子，并将输入参数完成相应缩放操作后再输入到算法模块中进行运算。相应地读出的计算结果也要根据缩放因子调整后才可以使用。具体设置请参考每个函数的参数说明。

由于数值缩放过程中数据截断会引起部分数据信息丢失，因此缩放因子会引入精度损失。

51.4.2. 运算精度

CORDIC 模块计算结果的精度取决于 CORDIC 算法的迭代次数。CORDIC 模块支持 1~8 次的精度配置，每一次迭代运算需要 3 个周期。

51.4.3. 库操作说明

51.4.3.1. CosineSine

函数功能	CosineSine 函数计算一个在π到π范围内的角度的正余弦 还可以用于极坐标到直角坐标的转换		
	参数	参数范围	参数描述
输入参数	θ	[-1,1]	以弧度为单位的角度值，需要除以π以便将输入参数范围转换到[-1,1]，Q31 格式
输出参数 1	$\cosh\theta$	[-1,1]	输入角度的余弦值，Q31 格式
输出参数 2	$\sin\theta$	[-1,1]	输入角度的正弦值，Q31 格式

CORDIC 驱动库中 Cosine/Sine 函数调用流程如下(以 float 型数据作为输入输出为例说明):

1）将输入的 float 格式数据乘以 231，转化成 Q31 格式数据；

2）调用 hal_cordicu.lib 库的 HAL_CORDIC_CosSin_x 函数(x 为计算精度，范围[1~8])，对 Q31 格式输入数据进行运算，得到 Q31 格式的 Cosine/Sine 运算结果；

3）将 Q31 格式的运算结果除以 231，得到对应 float 格式的 Cosine/Sine 运算结果；

51.4.3.2. AtanSqrt

函数功能	AtanSqrt 函数计算矢量 v=[x,y]在-π到π范围内的相位角,可参考函数 atan2(y,x)，还可以用于直角坐标到极坐标的转换		
	参数	参数范围	参数描述
输入参数 1	x	[-1,1]	x 坐标，如果|x|>1,必须软件缩放以满足 Q31 格式的输入范围
输入参数 2	y	[-1,1]	y 坐标，如果|y|>1,必须软件缩放以满足 Q31 格式的输入范围
输出参数 1	$\sqrt{x^2+y^2}$	[0,1]	输入矢量的模 如果|v|>1 计算结果将饱和输出为 1
输出参数 2	$\text{atan}2(y,x)$	[-1,1]	输入矢量的相位角 需要乘以π以得到弧度为单位的角度

CORDIC 驱动库中 AtanSqrt 函数调用流程如下(以用 float 型数据作为输入输出为例说明):

1）检查输入参数 x,y 是否满足[-1,1]，如不满足需要软件缩放至[-1,1]范围内；

2）将调整后的的 float 格式数据乘以 231，转化成 Q31 格式数据；

3）调用 hal_cordic 库的 HAL_CORDIC_AtanSqrt_x 函数(x 为计算精度，范围[1~8]),对 Q31 格式输入数据进行运算，得到 Q31 格式的 AtanSqrt 运算结果；

4）将 Q31 格式的 Sqrt 运算结果除以(CORDIC_F_31>>4, CORDIC_F_31 = 0xD2C90A46)，得到对应的 float 格式的 Sqrt 运算结果，如果第 1 步中有缩放调整，此处需反向缩放获得最终结果(缩放因子与输入相同)；

5）将 Q31 格式的 Atan 运算结果除以 231，得到对应 float 格式的 Atan 运算结果。如果想要得到以弧度为单位的角度值，需要将运算结果乘以π；

51.4.3.3. CoshSinh

函数功能	CoshSinh 函数计算双曲角度 x 的双曲正余弦 它也可用于计算指数函数 ex=coshx+sinhx 和 e-x=coshx-sinhx		
	参数	参数范围	参数描述
输入参数	θ	[-0.559,0.559]	输入双曲线角度值，算法支持参数范围为[-1.118~1.118]。由于这个范围超过 Q31 格式的表示范围，故输入算法运算时需先将输入参数除以 2，这样实际进入模块运算的参数范围就变为[-0.559,0.559]。
输出参数 1	$\cosh\theta$	[0.5,0.846]	输入角度的双曲余弦值，Q31 格式，需要乘以 2 才能得到最终结果
输出参数 2	$\sinh\theta$	[-0.683, 0.683]	输入角度的双曲正弦值，Q31 格式，需要乘以 2 才能得到最终结果

1）CORDIC 驱动库中 CoshSinh 函数调用流程如下(以用 float 型数据作为输入输出为例说明):

2）输入的 float 格式数据除以 2 乘以 231，将输入数据转化成[-0.559,0.559]范围内的 Q31 格式数据；

3）调用 hal_cordic 库的 HAL_CORDIC_CoshSinh_x 函数(x 为计算精度，范围[1~8]),对 Q31 格式输入数据进行运算，得到 Q31 格式的 CoshSinh 运算结果；

4）将 Q31 格式的运算结果除以 231 并乘以 2，得到 float 格式的 CoshSinh 运算结果；

51.4.3.4. Atanh

函数功能	Atanh 函数计算输入参数 x 的双曲反正切		
	参数	参数范围	参数描述
输入参数	θ	[-0.403 0.403]	算法支持参数范围为[-0.806~0.806]。输入算法模块运算时必须先将输入参数除以 2，这样实际进入模块运算的参数范围就变为[-0.403 0.403]
输出参数	$\text{atanh}\theta$	[-0.559 0.559]	输入参数的双曲反正切值，Q31 格式，需要乘以 2 才能得到最终结果

CORDIC 驱动库中 Atanh 函数调用流程如下(以用 float 型数据作为输入输出为例说明):

1）输入的 float 格式数据除以 2 乘以 231，将输入数据转化成[-0.403 0.403]范围的 Q31 格式数据；

2）调用 hal_cordic 库的 HAL_CORDIC_Atanh_x 函数(x 为计算精度，范围[1~8]),对 Q31 格式输入数据进行运算，得到 Q31 格式的 Atanh 运算结果；

3）将 Q31 格式的运算结果除以 231 并乘以 2，得到 float 格式的 Atanh 运算结果；

51.4.3.5. Ln

函数功能	Ln 函数计算输入参数 x 的自然对数		
	参数	参数范围	参数描述
输入参数	x	[0.054~0.875]	算法支持参数范围为[0.107~9.35]。输入算法模块运算时必须先将输入参数按照输入参数缩放表进行相应缩放再输入算法模块进行运算。这样实际进入模块运算的参数范围就变为[0.054~0.875]
输出参数	$Ln(x)$	[-0.279~0.137]	输入参数的自然对数值，Q31 格式，需要乘以 4 才能得到最终结果

CORDIC 驱动库中 Ln 函数调用流程如下(以用 float 型数据作为输入输出为例说明):

1）判断输入数据 x 范围，参照下表进行缩放并转成 Q31 格式；

输入数据范围	Scale	转 Q31 格式	输入参数范围(float 格式)
0.107≤x< 1	1	乘以 231 并除以 2	0.0535≤ARG1< 0.5
1≤ x< 3	2	乘以 231 并除以 4	0.25≤ARG1< 0.75
3≤ x< 7	3	乘以 231 并除以 8	0.375≤ARG1< 0.875
7≤ x≤ 9.35	4	乘以 231 并除以 16	0.4375≤ARG1< 0.584

2）调用 hal_cordic 库的 HAL_CORDIC_Ln_x 函数(x 为计算精度，范围[1~8])，对 Q31 格式输入数据进行运算，得到 Q31 格式的 Ln 运算结果；

3）将 Q31 格式的运算结果除以 231 并乘以 4，得到对应 float 格式的 Ln 运算结果；

51.4.3.6. Sqrt

函数功能	Sqrt 函数计算输入参数 x 的平方根		
	参数	参数范围	参数描述
输入参数	x	[0.027~0.875]	算法支持参数范围为[0.027~2.341]。输入算法模块运算时必须先将输入参数按照输入参数缩放表进行相应缩放再输入算法模块进行运算。这样实际进入模块运算的参数范围就变为[0.027~0.875]
输出参数	$\sqrt{x}$	[0.04~1]	输入参数的平方根，Q31 格式，需要按照输出参数转换表处理后才能得到最终结果

CORDIC 驱动库中 Sqrt 函数调用流程如下(以用 float 型数据作为输入输出为例说明):

1）判断输入数据 x 范围，参照下表进行缩放并转成 Q31 格式；

输入数据范围	Scale	转 Q31 格式	输入参数范围(float 格式)
0.027 ≤x< 0.75	0	乘以 231	0.027≤ARG1< 0.75
0.75 ≤ x < 1.75	1	乘以 231 并除以 2	0.375≤ARG1< 0.875
1.75 ≤ x ≤ 2.341	2	乘以 231 并除以 4	0.4375≤ARG1≤0.585

2）调用 hal_cordic 库的 HAL_CORDIC_Sqrt_x 函数(x 为计算精度，范围[1~8]),对 Q31 格式输入数据进行运算，得到 Q31 格式的 Sqrt 运算结果；

3）参照下面输出参数转换表，得到对应 float 格式的 Sqrt 运算结果(W_INV_Q31 = 0x6A012206)

Scale	对应操作
0	除以 W_INV_Q31
1	乘以 2 并除以 W_INV_Q31
2	乘以 4 并除以 W_INV_Q31

52. 随机数发生器（HRNG）

随机数是密码学算法的基础，是现代加密体系中最重要的部分之一。几乎所有的密码学算法都需要使用随机数。

HRNG 模块是一个以连续模拟噪声为基础的随机数发生器，模拟电路产生馈入线性反馈移位寄存器（LFSR）的种子，在主机读取时提供一个 32 位的随机数，可向应用程序提供作为 32 位采样的全熵输出。

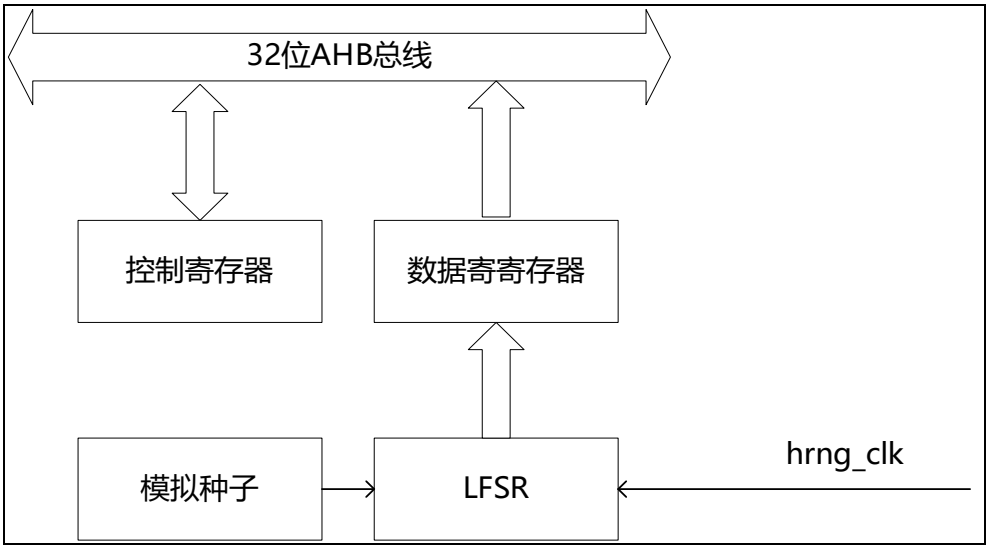
HRNG 可作为一个实时熵源用来构建符合 NIST 要求的确定性随机位发生器(DRBG)。

52.1. 主要特性

- 内含可靠噪声振荡器，提供由模拟熵源生成的 32 位真随机数；
- 符合国际 FIPS-140-2 和 NIST SP800-22 测试标准；
- 符合国密局《随机数检测规范》标准；
- RNG 每 32 个 HRNG_CLK 时钟周期会生成一个 32 位随机采样。
- 可被禁止以降低功耗

52.2. 功能说明及框图

图 52-1 HRNG 结构框图



真随机数(HRNG)发生器主要振荡环噪声源，后处理数字单元线性反馈移位寄存器 LFSR，控制寄存器，数据寄存器组成。程序员可以通过配置控制寄存器来使能和关闭振荡器，通过每隔一段时间读取数据寄存器中的数据获取随机数。

52.3. 随机数库操作说明

要获取随机数，请使用提供的随机数库并按以下步骤操作：

1）调用 hal_hrng.lib 库中的 HAL_HRNG_Init 对随机数模块进行初始化。

2）调用 hal_hrng.lib 库中的 HAL_HRNG_GetHrng 函数，输入存放采集随机数的起始地址，及需要采集的随

机数长度，完成随机数采集。

3）采集完成后，调用 hal_hrng.lib 库中的 HAL_HRNG_DeInit 函数关闭随机数模块。

53. 自定义指令（CDE）

为了支持 Arm 自定义指令，Armv8-M 体系结构需要一个新的体系结构扩展，这被命名为自定义数据路径扩展（CDE）。

自定义指令支持字节拼接、按位倒序、CRC 计算、汉明距离计算、bit 统计、SM3/SM4 辅助算法、乘累加等运算，执行周期数在 1~2 个系统时钟。

53.1. 自定义指令总表

CDE 支持以下指令：

函数名称	Arm 自定义指令	周期数
__HAL_CDE_BYTE_AMP(a,b)	__arm_cx2a(0, a, b, 4)	1
__HAL_CDE_BYTE_PACKN(a, b,n)	__arm_cx3a(0,(a),(b), n,11)	2
__HAL_CDE_3BYTE_PACKN(a,n)	__arm_cx2a(0,(a),n,5)	1
__HAL_CDE_7BYTE_PACKN(a,n)	__arm_cx2da(0,(a),n,6)	2
__HAL_CDE_BYTE_REVERSE(a)	__arm_cx1a(0, (a), 2)	1
__HAL_CDE_CRC32_32BIT(crc, d)	__arm_cx2a(0, (crc), (d), 1)	1
__HAL_CDE_CRC32_REVERSE(crc, d)	__arm_cx2a(0, (crc), (d), 2)	1
__HAL_CDE_CTZ(a)	__arm_cx2a(0, a, 0, 3)	1
__HAL_CDE_FFS(a)	__arm_cx2a(0, a, 1, 3)	1
__HAL_CDE_HAMMING_DIST(b, c)	__arm_cx3(0,b,c, 9)	1
__HAL_CDE_PACK8(a, b)	__arm_cx3(0,(a),(b), 4)	1
__HAL_CDE_PACK16(a, b)	__arm_cx3(0,(a),(b), 5)	1
__HAL_CDE_PACK24(a, b)	__arm_cx3(0,(a),(b), 6)	1
__HAL_CDE_PARITY(a)	__arm_cx1a(0, (a), 7)	1
__HAL_CDE_POPCNT(a)	__arm_cx1a(0, (a), 1)	1
__HAL_CDE_SM3_FF2(x, y, z)	__arm_cx3a(0,x, y, z,1)	2
__HAL_CDE_SM3_GG2(x, y, z)	__arm_cx3a(0,x, y, z,10)	2
__HAL_CDE_SM3_P0(a)	__arm_cx1a(0, (a), 3)	1
__HAL_CDE_SM3_P1(a)	__arm_cx1a(0, (a), 4)	1
__HAL_CDE_SM4_T1(a)	__arm_cx1a(0, (a), 5)	1
__HAL_CDE_SM4_T2(a)	__arm_cx1a(0, (a), 6)	1
__HAL_CDE_SUMLAQ(a, b, c)	__arm_cx3a(0,(a),(b),(c), 3)	2
__HAL_CDE_SUMUAQ(b, c)	__arm_cx3(0,(b),(c), 8)	1
__HAL_CDE_U16_REVERSE(a)	__arm_cx1a(0, (a), 8)	1
__HAL_CDE_UMLAQ(a, b, c)	__arm_cx3a(0,(a),(b),(c), 2)	2
__HAL_CDE_UMUAQ(b, c)	__arm_cx3(0,(b),(c), 7)	1

53.2. 自定义指令说明

53.2.1. BYTE_AMP

指令	BYTE_AMP
描述	字节放大 例如: bit[7:0] a = 01101011 b=1 : bit[15:0] 00 11 11 00 11 00 11 11 b=2 : bit[31:0] 00000000 000 111 111 000 111 000 111 111 b=4 : bit[31:0] 0000 1111 1111 0000 1111 0000 1111 1111

参考 c 代码	uint32_t byte_amp2(uint8_t a) {
	uint32_t a1,a2;
	uint32_t b=0;
	a1 = a&0x1;
	a1 |= a1<<1;
	b |= a1;
	
	a1 = a&0x2;
	a1 |= a1<<1;
	b |= a1<<1;
	
	a1 = a&0x4;
	a1 |= a1<<1;
	b |= a1<<2;
	
	a1 = a&0x8;
	a1 |= a1<<1;
	b |= a1<<3;
	
	a1 = a&0x10;
	a1 |= a1<<1;
	b |= a1<<4;
	
	a1 = a&0x20;
	a1 |= a1<<1;
	b |= a1<<5;
	
	a1 = a&0x40;
	a1 |= a1<<1;
	b |= a1<<6;
	
	a1 = a&0x80;
	a1 |= a1<<1;
	b |= a1<<7;
	
	return b;
	}
	uint32_t byte_amp3(uint8_t a) {
	uint32_t a1,a2;
	uint32_t b=0;
	a1 = a&0x1;
	a1 |= (a1<<1) | (a1<<2);
	b |= a1;
	
	a1 = a&0x2;

```
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<2;

    a1 = a&0x4;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<4;

    a1 = a&0x8;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<6;

    a1 = a&0x10;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<8;

    a1 = a&0x20;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<10;

    a1 = a&0x40;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<12;

    a1 = a&0x80;
    a1 |= (a1<<1) | (a1<<2);
    b |= a1<<14;

  return b;
}
uint32_t byte_amp4(uint8_t a){
    uint32_t a1,a2;
    uint32_t b=0;
    a1 = a&0x1;
    a1 |= (a1<<1) | (a1<<2)   | (a1<<3);
    b |= a1;

    a1 = a&0x2;
    a1 |= (a1<<1) | (a1<<2)| (a1<<3);
    b |= a1<<3;

    a1 = a&0x4;
    a1 |= (a1<<1) | (a1<<2)| (a1<<3);
    b |= a1<<6;

    a1 = a&0x8;
    a1 |= (a1<<1) | (a1<<2)| (a1<<3);
```

	b |= a1<<9;
	
	a1 = a&0x10;
	a1 |= (a1<<1) | (a1<<2)| (a1<<3);
	b |= a1<<12;
	
	a1 = a&0x20;
	a1 |= (a1<<1) | (a1<<2)| (a1<<3);
	b |= a1<<15;
	
	a1 = a&0x40;
	a1 |= (a1<<1) | (a1<<2)| (a1<<3);
	b |= a1<<18;
	
	a1 = a&0x80;
	a1 |= (a1<<1) | (a1<<2)| (a1<<3);
	b |= a1<<21;
	
	return b;
	}

函数声明	#define __HAL_CDE_BYTE_AMP(a,b) __arm_cx2a(0, a, b, 4)

53.2.2. BYTE_PACKN

指令	BYTE_PACKN
描述	此指令由两个 8 位数据生成 8 位数据

参考 c 代码	uint8_t BYTE_PACKN(uint8_t a, uint8_t b, uint8_t n) {
	return (a >> n) | (b << (8-n));
	}

函数声明	#define __HAL_CDE_BYTE_PACKN(a, b,n) __arm_cx3a(0,(a),(b), n,11)

53.2.3. 3BYTE_PACKN

指令	3BYTE_PACKN
描述	该指令由 32 位数据中的 4 个字节生成 3 字节数据

参考 c 代码	uint32_t BYTE3_PACKN(uint32_t a, uint8_t n) {
	uint8_t c,c1,c2,c3;
	uint8_t a1,a2,a3,a4;
	a1 = (a&0xff);
	a2 = (a&0xff00)>>8;
	a3 = (a&0xff0000)>>16;
	a4 = (a&0xff000000)>>24;
	c1 = (a1 >> n) | (a2 << (8-n));
	c2 = (a2 >> n) | (a3 << (8-n));
	c3 = (a3 >> n) | (a4 << (8-n));
	c = (c1&0xff) | ((c2<<8)&0xff00) | ((c3<<16)&0xff0000);
	return c;
	}

函数声明	#define __HAL_CDE_3BYTE_PACKN(a,n) __arm_cx2a(0,(a),n,5)

53.2.4. 7BYTE_PACKN

指令	7BYTE_PACKN
描述	该指令由 64 位数据中的 8 个字节数据生成 7 字节数据

参考 c 代码	uint64_t BYTE7_PACKN(uint64_t a, uint8_t n) {
	uint64_t c,c1,c2,c3,c4,c5,c6,c7;
	uint8_t a1,a2,a3,a4,a5,a6,a7,a8;
	a1 = (a&0xff);
	a2 = (a&0xff00)>>8;
	a3 = (a&0xff0000)>>16;
	a4 = (a&0xff000000)>>24;
	a5 = (a&0xff00000000)>>32;
	a6 = (a&0xff0000000000)>>40;
	a7 = (a&0xff000000000000)>>48;
	a8 = (a&0xff00000000000000)>>56;
	c1 = (a1 >> n) | (a2 << (8-n));
	c2 = (a2 >> n) | (a3 << (8-n));
	c3 = (a3 >> n) | (a4 << (8-n));
	c4 = (a4 >> n) | (a5 << (8-n));
	c5 = (a5 >> n) | (a6 << (8-n));
	c6 = (a6 >> n) | (a7 << (8-n));
	c7 = (a7 >> n) | (a8 << (8-n));
	c = (c1&0xff) | ((c2<<8)&0xff00) | ((c3<<16)&0xff0000) | ((c4<<24)&0xff000000) | ((c5<<32)&0xff00000000) | ((c6<<40)&0xff000000000) | ((c7<<48)&0xff000000000000);
	return c;
	}

函数声明	#define __HAL_CDE_7BYTE_PACKN(a,n) __arm_cx2da(0,(a),n,6)

53.2.5. BYTE_REVERSE

指令	BYTE_REVERSE
描述	该指令将 32 位数据以字节为单位，每个字节内按位倒序 例如，0x112233344 变为 0x8844cc22

<table>
<tr><td>参考 c 代码</td><td><pre>
UINT8 reverse_byte(UINT8 in_data){
	UINT8 temp;
	UINT8 i;
	temp = 0;
	for(i = 0; i < 8; i++)
	{
		if(in_data & (1 << i))
		{
			temp |= 0x80 >> i;
		}
	}
	return temp;
}
</pre></td></tr>
<tr><td>函数声明</td><td>#define __HAL_CDE_BYTE_REVERSE(a)	__arm_cx1a(0, (a), 2)</td></tr>
</table>

53.2.6. CRC32_32BIT

<table>
<tr><th>指令</th><th>CRC32_32BIT</th></tr>
<tr><td>描述</td><td>多项式：0x04C11DB7
crc32[31:0]=1+x^1+x^2+x^4+x^5+x^7+x^8+x^10+x^11+x^12+x^16
+x^22+x^23+x^26+x^32;
支持一次计算一个字
内部会对 d 进行字内倒序，
即输入 d[31:0] 会转换为 {d[7:0],d[15:8],d[23:16],d[31:24]}</td></tr>
<tr><td>参考 c 代码</td><td><pre>
#define POLYNOMIAL 0xedb88320	//(0x04C11DB7 按 32bit 倒序值)
uint32_t CRC32(uint32_t crc, uint32_t d) {
  crc = crc ^ (d&0xff);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  if (crc&1) crc = (crc>>1) ^ POLYNOMIAL;   else crc = (crc>>1);
  return crc;
}
</pre>具体使用方法可参考例程</td></tr>
<tr><td>函数声明</td><td>#define __HAL_CDE_CRC32_32BIT(crc, d)	__arm_cx2a(0, (crc), (d), 1)</td></tr>
</table>

53.2.7. CRC32_REVERSE

指令	CRC32_REVERSE
描述	多项式：0x04C11DB7 crc32[31:0]=1+x^1+x^2+x^4+x^5+x^7+x^8+x^10+x^11+x^12+x^16 +x^22+x^23+x^26+x^32; 支持一次计算一个字节 内部对初始值和数据以及结果进行反序

参考 c 代码	#define POLYNOMIAL 0xedb88320 //(0x04C11DB7 按 32bit 倒序值)
	uint32_t CRC32(uint32_t crc, uint32_t d) {
	crc = crc ^ (d&0xff);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	if (crc&1) crc = (crc>>1) ^ POLYNOMIAL; else crc = (crc>>1);
	return crc;
	}
	具体使用方法可参考例程

函数声明	#define __HAL_CDE_CRC32_REVERSE(crc, d) __arm_cx2a(0, (crc), (d), 2)

53.2.8. CTZ

指令	CTZ
描述	该指令返回数据值二进制表示形式中末尾 0 的个数 例如： 0x00 : 32 0x11 : 0 0x1122 : 1 0x80000000 : 31
参考 c 代码	__builtin_ctz
函数声明	#define __HAL_CDE_CTZ(a) __arm_cx2a(0, a, 0, 3)

53.2.9. FFS

指令	FFS
描述	该指令返回数据值二进制表示形式中从低位到高位，数据中 1 出现的位置 例如： 0x00 : 32 0x11 : 0 0x1122 : 1 0x80000000 : 31
参考 c 代码	__builtin_ffs
函数声明	#define __HAL_CDE_FFS(a) __arm_cx2a(0, a, 1, 3)

53.2.10. HAMMING_DIST

指令	HAMMING_DIST
描述	该指令计算两个输入数据之间的汉明距离 输入寄存器可以是任意值，有效输出值[0:31]

参考 c 代码	uint32_t HAMMING_DIST(uint32_t b, uint32_t c) {
	uint32_t i, a, val;
	val = b ^ c;
	a = 0;
	for (i = 0; i < 32; i++) {
	a = a + (val & 1);
	val = val >> 1;
	}
	return a;
	}

函数声明	#define __HAL_CDE_HAMMING_DIST(b, c) __arm_cx3(0,b,c, 9)

53.2.11. PACK8

指令	PACK8
描述	该指令从两个 32 位寄存器中提取 32 位结果 res[0:23] = a[8:31] res[24:31] = b[0:7]

参考 c 代码	uint32_t PACK8(uint32_t a, uint32_t b) {
	return (a >> 8) | (b << 24);
	}

函数声明	#define __HAL_CDE_PACK8(a, b) __arm_cx3(0,(a),(b), 4)

53.2.12. PACK16

指令	PACK16
描述	该指令从两个 32 位寄存器中提取 32 位结果 res[0:15] = a[16:31] res[16:31] = b[0:15]

参考 c 代码	uint32_t PACK16(uint32_t a, uint32_t b) {
	return (a >> 16) | (b << 16);
	}

函数声明	#define __HAL_CDE_PACK16(a, b) __arm_cx3(0,(a),(b), 5)

53.2.13. PACK24

指令	PACK24
描述	该指令从两个 32 位寄存器中提取 32 位结果 res[0:7] = a[24:31] res[8:31] = b[0:23]

参考 c 代码	uint32_t PACK24(uint32_t a, uint32_t b) {
	return (a >> 24) | (b << 8);
	}

函数声明	#define __HAL_CDE_PACK24(a, b) __arm_cx3(0,(a),(b), 6)

53.2.14. PARITY

指令	PARITY
描述	该指令计算 32 数据的二进制表示形式中 1 个数的奇偶 例如： 0x11 : 0 0x23 : 1 0xffff: 0
参考 c 代码	__builtin_parity
函数声明	#define __HAL_CDE_PARITY(a) __arm_cx1a(0, (a), 7)

53.2.15. POPCNT

指令	POPCNT
描述	对 32 位寄存器中的 1 进行计数 该指令的输出是一个 32 位数据，值为 1 的个数

参考 c 代码	uint32_t POPCNT(uint32_t a) {
	a = (a&0x55555555) + ((a&0xaaaaaaaa)>> 1);
	a = (a&0x33333333) + ((a&0xcccccccc)>> 2);
	a = (a&0x0f0f0f0f) + ((a&0xf0f0f0f0)>> 4);
	a = (a&0x00ff00ff) + ((a&0xff00ff00)>> 8);
	a = (a&0x0000ffff) + ((a&0xffff0000)>>16);
	return a;
	}

函数声明	#define __HAL_CDE_POPCNT(a) __arm_cx1a(0, (a), 1)

53.2.16. SM3_FF2

指令	SM3_FF2
描述	x 与 y 逻辑与的结果或上 x 与 z 逻辑与的结果或上 y 与 z 逻辑与的结果
参考 c 代码	#define SM3_FF2(x,y,z) (x&y)|(x&z)|(y&z)
函数声明	#define __HAL_CDE_SM3_FF2(x, y, z) __arm_cx3a(0,x, y, z,1)

53.2.17. SM3_GG2

指令	SM3_GG2
描述	x 与 y 逻辑与的结果或上 x 取反值与 z 逻辑与的结果
参考 c 代码	#define SM3_GG2(x,y,z) (x&y)|(~x&z)
函数声明	#define __HAL_CDE_SM3_GG2(x, y, z) __arm_cx3a(0,x, y, z,10)

53.2.18. SM3_P0

指令	SM3_P0
描述	将输入数据与输入数据左移 9 位及左移 17 位异或
参考 c 代码	#define ROTATE_LEFT(n,x) __ROR(x,32-n) #define SM3_P0(x) x^ ROTATE_LEFT(9,x) ^ ROTATE_LEFT(17,x)
函数声明	#define __HAL_CDE_SM3_P0(a) __arm_cx1a(0, (a), 3)

53.2.19. SM3_P1

指令	SM3_P1
描述	将输入数据与输入数据左移 15 位及左移 23 位异或
参考 c 代码	#define ROTATE_LEFT(n,x) __ROR(x,32-n) #define SM3_P1(x) x^ ROTATE_LEFT(15,x) ^ ROTATE_LEFT(23,x)
函数声明	#define __HAL_CDE_SM3_P1(a) __arm_cx1a(0, (a), 4)

53.2.20. SM4_T1

指令	SM4_T1
描述	将输入数据与输入数据左移 2 位及左移 10 位异或
参考 c 代码	#define ROTATE_LEFT(n,x) __ROR(x,32-n) #define SM4_T1(x) x^ ROTATE_LEFT(2,x) ^ ROTATE_LEFT(10,x) ^ ROTATE_LEFT(18,x) ^ ROTATE_LEFT(24,x)
函数声明	#define __HAL_CDE_SM4_T1(a) __arm_cx1a(0, (a), 5)

53.2.21. SM4_T2

指令	SM4_T2
描述	将输入数据与输入数据左移 13 位及左移 23 位异或
参考 c 代码	#define ROTATE_LEFT(n,x) __ROR(x,32-n) #define SM4_T2(x) x^ ROTATE_LEFT(13,x) ^ ROTATE_LEFT(23,x)
函数声明	#define __HAL_CDE_SM4_T2(a) __arm_cx1a(0, (a), 6)

53.2.22. SUMLAQ

指令	SUMLAQ
描述	该指令执行四个 8 位有符号乘 8 位无符号乘法运算，形成四个 16 位乘积 四个乘积和累加器“a”相加，返回 32 位结果

参考 c 代码	int32_t SUMLAQ(int32_t a, uint32_t b, uint32_t c) {
	int16_t a0, a1, a2, a3;
	a0 = (int16_t)(int8_t)((b >> 0) & 0xff) * (int16_t)(uint8_t)((c >> 0) & 0xff);
	a1 = (int16_t)(int8_t)((b >> 8) & 0xff) * (int16_t)(uint8_t)((c >> 8) & 0xff);
	a2 = (int16_t)(int8_t)((b >> 16) & 0xff) * (int16_t)(uint8_t)((c >> 16) & 0xff);
	a3 = (int16_t)(int8_t)((b >> 24) & 0xff) * (int16_t)(uint8_t)((c >> 24) & 0xff);
	return a + (int32_t)a0 + (int32_t)a1 + (int32_t)a2 + (int32_t)a3;
	}

函数声明	#define __HAL_CDE_SUMLAQ(a, b, c) __arm_cx3a(0,(a),(b),(c), 3)

53.2.23. SUMUAQ

指令	SUMUAQ
描述	该指令执行四个 8 位有符号乘 8 位无符号乘法运算，形成四个 16 位乘积 四个乘积相加，返回 32 位结果

参考 c 代码	int32_t SUMUAQ(uint32_t b, uint32_t c) {
	int16_t a0, a1, a2, a3;
	a0 = (int16_t)(int8_t)((b >> 0) & 0xff) * (int16_t)(uint8_t)((c >> 0) & 0xff);
	a1 = (int16_t)(int8_t)((b >> 8) & 0xff) * (int16_t)(uint8_t)((c >> 8) & 0xff);
	a2 = (int16_t)(int8_t)((b >> 16) & 0xff) * (int16_t)(uint8_t)((c >> 16) & 0xff);
	a3 = (int16_t)(int8_t)((b >> 24) & 0xff) * (int16_t)(uint8_t)((c >> 24) & 0xff);
	return (int32_t)a0 + (int32_t)a1 + (int32_t)a2 + (int32_t)a3;
	}

函数声明	#define __HAL_CDE_SUMUAQ(b, c) __arm_cx3(0,(b),(c), 8)

53.2.24. U16_REVERSE

指令	U16_REVERSE
描述	该指令在一个 32 位寄存器中，以 16bit 为单位，将 16bit 数据按位倒序 例如： 0x11223344 ： 0x448822cc

参考 c 代码	uint16_t reverse_u16(uint16_t in_data){
	uint16_t temp;
	uint16_t i;
	temp = 0;
	for(i = 0; i < 16; i++)
	{
	if(in_data & (1 << i))
	{
	temp |= 0x8000 >> i;
	}
	}
	return temp;
	}
	uint32_t u16_reverse(uint32_t a) {
	uint32_t a1,a2;
	a1 = reverse_u16(a&0xffff);
	a2 = reverse_u16((a>>16)&0xffff);
	a = a1 | (a2<<16);
	return a;
	}

函数声明	#define __HAL_CDE_U16_REVERSE(a) __arm_cx1a(0, (a), 8)

53.2.25. UMLAQ

指令	UMLAQ
描述	该指令执行四个 8x8 位无符号乘法运算，形成四个 16 位乘积 四个乘积和累加器“a”相加，返回 32 位结果

参考 c 代码	uint32_t UMLAQ(uint32_t a, uint32_t b, uint32_t c) {
	uint16_t a0, a1, a2, a3;
	a0 = (uint16_t)((b >> 0) & 0xff) * (uint16_t)((c >> 0) & 0xff);
	a1 = (uint16_t)((b >> 8) & 0xff) * (uint16_t)((c >> 8) & 0xff);
	a2 = (uint16_t)((b >> 16) & 0xff) * (uint16_t)((c >> 16) & 0xff);
	a3 = (uint16_t)((b >> 24) & 0xff) * (uint16_t)((c >> 24) & 0xff);
	return a + (uint32_t)a0 + (uint32_t)a1 + (uint32_t)a2 + (uint32_t)a3;
	}

函数声明	#define __HAL_CDE_UMLAQ(a, b, c) __arm_cx3a(0,(a),(b),(c), 2)

53.2.26. UMUAQ

指令	UMUAQ
描述	该指令执行四个 8x8 位无符号乘法运算，形成四个 16 位乘积 四个乘积相加，返回 32 位结果

参考 c 代码	uint32_t UMUAQ(uint32_t b, uint32_t c) {
	uint16_t a0, a1, a2, a3;
	a0 = (uint16_t)((b >> 0) & 0xff) * (uint16_t)((c >> 0) & 0xff);
	a1 = (uint16_t)((b >> 8) & 0xff) * (uint16_t)((c >> 8) & 0xff);
	a2 = (uint16_t)((b >> 16) & 0xff) * (uint16_t)((c >> 16) & 0xff);
	a3 = (uint16_t)((b >> 24) & 0xff) * (uint16_t)((c >> 24) & 0xff);
	return (uint32_t)a0 + (uint32_t)a1 + (uint32_t)a2 + (uint32_t)a3;
	}

函数声明	#define __HAL_CDE_UMUAQ(b, c) __arm_cx3(0,(b),(c), 7)

54. 版本历史

版本	日期	作者	描述
V1.0	2023-06-15	hangxin	初始版
V1.1	2024-08-15	hangxin	内容补充，善格式
V1.2	2024-10-12	hangxin	WKUP0-4 改为 WKUP1-5；补充 OTFDEC 章节；删除互感模式
V1.3	2024-1-9	hangxin	修改定时器章节的互连关系，及部分模块描述
V1.4	2024-1-17	hangxin	修改 LPTIM 框图
V1.5	2025-3-31	hangxin	修改多个模块内容

55. 版权声明

联系我们

公司：上海航芯电子科技股份有限公司

地址：上海市闵行区合川路 2570 号科技绿洲三期 2 号楼 702 室

邮编：200241

电话：+86-21-6125 9080

传真：+86-21-6125 9080-830

Email： service@hangchip.com

Website： www.hangchip.com