ACM32G103 用户手册

基于 ARM Star-MC1 内核的 32 位微控制器

版本：V2.0

日期：2024-1-20

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 write0 (RCW0):	只读，写 0 清 0，写 1 无效
read/clear write1 (RCW1):	只读，写 1 清 0，写 0 无效
read/clear by read (RCR):	只读，读该位后硬件自动清零该位，写无效
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）
eFlash:	嵌入式 FLASH（embedded Flash）

2. 存储器及系统架构（SYSCFG）

本芯片的内核采用 Star-MC1，支持 Cortex-M33 和 Cortex-M4F 指令集，系统最高频率达 120 MHz。内核支持一整套 DSP 指令用于数字信号处理，支持单精度 FPU 处理浮点数据，同时还支持 Memory Protection Unit（MPU）用于提升应用的安全性。

2.1. 内核处理器

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。

处理器内置单精度浮点处理单元，支持数字信号处理指令，提供实时处理和高效的中断处理能力。

处理器配置了 4KB 的指令缓存（ICACHE）和 4KB 的数据缓存（DCACHE），可提高代码执行效率。

处理器的具体规格可参见 Star-MC1 的 User Guide。

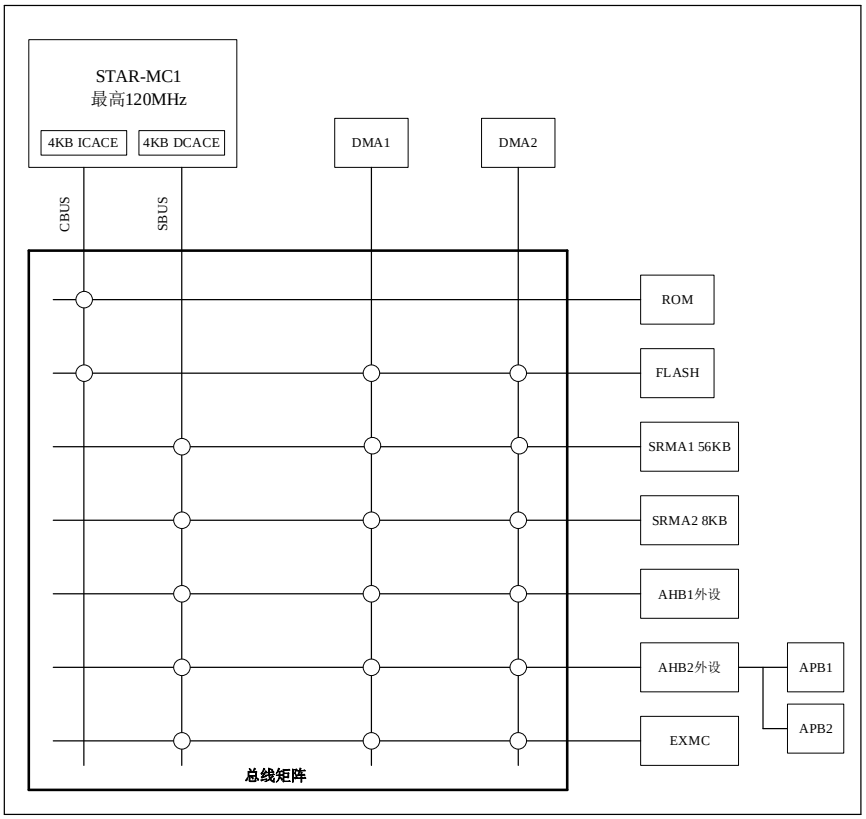
2.2. 系统架构

ACM32G103 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、两个 AHB 总线（AHB1 和 AHB2）和两个 APB 总线（APB1 和 APB2）。

Multi-AHB 总线矩阵可连接：

- 4 条主控总线
  - CBUS（内核 Code 总线）
  - SBUS（内核 System 总线）
  - DMA1
  - DMA2
- 7 条被控总线
  - 内部 ROM
  - 内部 Flash
  - SRAM1（56KB）
  - SRAM2（8KB）
  - AHB1 外设
  - AHB2 外设
  - EXMC

图 2-1 总线矩阵

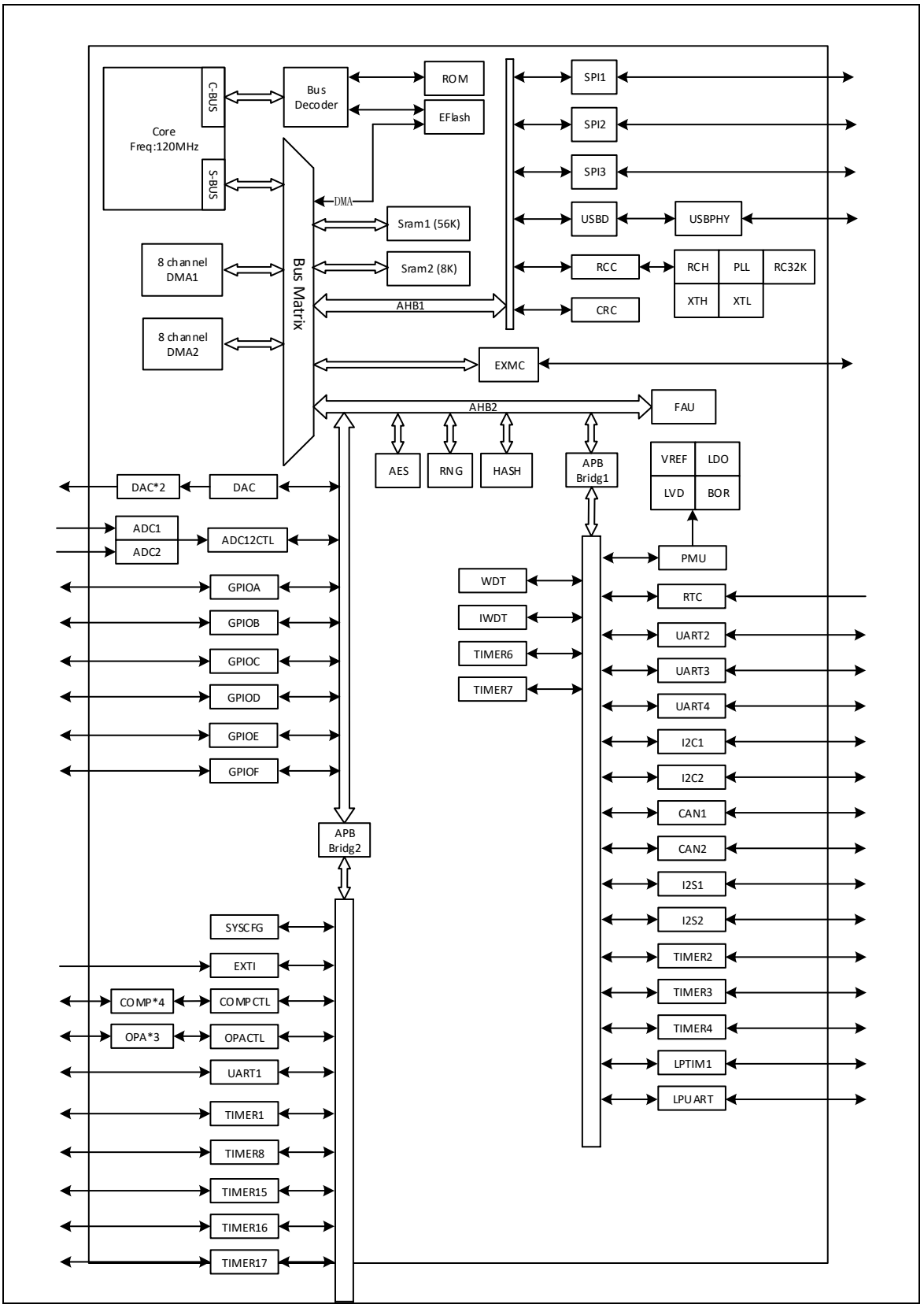


总线矩阵用于各主控总线间的访问仲裁管理。仲裁采用固定优先级算法。

- 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
- AHB2APB 总线桥：在 AHB 与 APB 总线间提供同步连接。在每次复位之后，所有的外设时钟处于关闭状态（除 FLASH 和 SRAM 外）。在用一个外设外前，软件必须打开相应的时钟使能位

系统架构图如下所示。AHB1 和 AHB2 的时钟频率一致且等于系统频率，APB1 和 APB2 的时钟频率由系统频率分频而来，分频比可独立配置。

图 2-2 系统架构

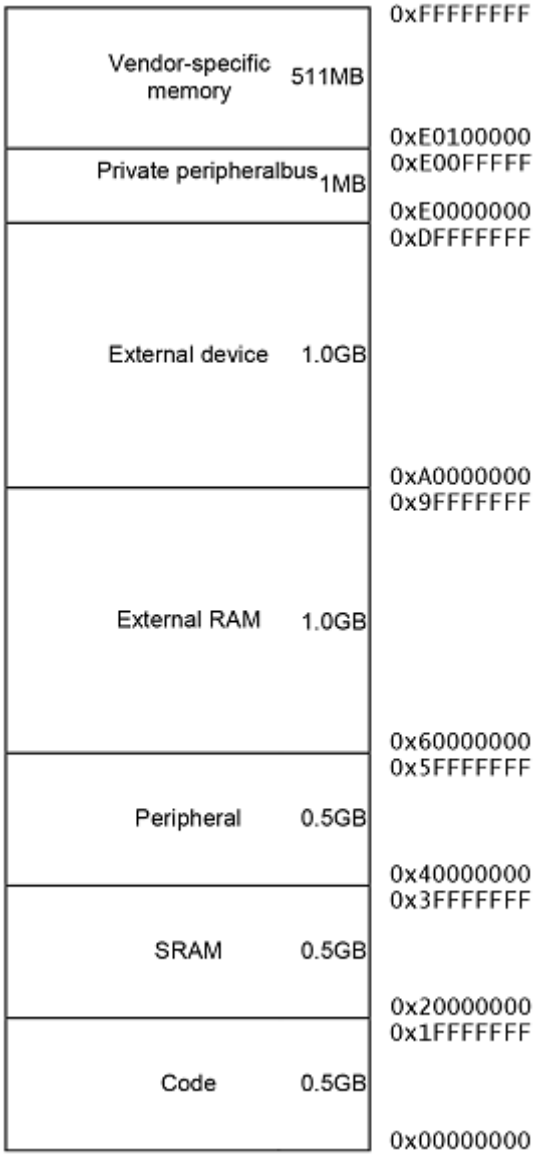


2.3. 存储器映射

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



本芯片的存储空间分为 ROM 启动和 EFLASH 启动两种模式，具体如下表所示。

ROM 上电时，ROM 的起始地址 0x0000_0000 和 0x1200_0000 同时有效（0x1200_0000 为 shadow 区）；eFlash 起始地址映射为 0x1000_0000。

系统支持存储地址空间重映射（remapping）机制，即将 256KB Flash 空间映射到 0x0 起始地址，以屏蔽 ROM 启动，直接从 eFlash 启动，便于用户控制所有系统资源，此时 EFLASH 空间起始地址为 0x0000_0000，ROM 空间起始地址为 0x1200_0000。

表 2-1 存储地址映射表

模块名	ROM 启动	EFlash 启动
ROM	0x0000_0000——0x0000_2FFF 0x1200_0000——0x1200_2FFF	0x1200_0000——0x1200_2FFF
eFlash	0x1000_0000——0x1003_FFFF	0x0000_0000——0x0003_FFFF
eFlash NVR	0x1008_0000——0x1008_0FFF	0x0008_0000——0x0008_0FFF
SRAM		

SRAM	0x2000_0000——0x2000_FFFF
APB1 Slave	
TIM2	0x4000_0000——0x4000_03FF
TIM3	0x4000_0400——0x4000_07FF
TIM4	0x4000_0800——0x4000_0BFF
保留	0x4000_0C00——0x4000_0FFF
TIM6	0x4000_1000——0x4000_13FF
TIM7	0x4000_1400——0x4000_17FF
保留	0x4000_1800——0x4000_1BFF
保留	0x4000_1C00——0x4000_1FFF
保留	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
保留	0x4000_3400——0x4000_37FF
I2S1	0x4000_3800——0x4000_3BFF
I2S2	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
保留	0x4000_5000——0x4000_53FF
I2C1	0x4000_5400——0x4000_57FF
I2C2	0x4000_5800——0x4000_5BFF
保留	0x4000_5C00——0x4000_5FFF
保留	0x4000_6000——0x4000_63FF
CAN1	0x4000_6400——0x4000_67FF
CAN2	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_FFFF

APB2 Slave	
SYSCFG	0x4001_0000——0x4001_002F
保留	0x4001_0030——0x4001_01FF
COMP	0x4001_0200——0x4001_02FF
OPA	0x4001_0300——0x4001_03FF
EXTI	0x4001_0400——0x4001_07FF
保留	0x4001_0800——0x4001_0BFF
TIM1	0x4001_2C00——0x4001_2FFF
保留	0x4001_3000——0x4001_33FF
TIM8	0x4001_3400——0x4001_37FF
UART1	0x4001_3800——0x4001_3BFF
保留	0x4001_3C00——0x4001_3FFF
TIM15	0x4001_4000——0x4001_43FF
TIM16	0x4001_4400——0x4001_47FF
TIM17	0x4001_4800——0x4001_4BFF
保留	0x4001_4C00——0x4001_FFFF
AHB1 Slave	
保留	0x4002_0000——0x4002_03FF
保留	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
EFC	0x4002_2000——0x4002_23FF
保留	0x4002_2400——0x4002_2FFF
CRC	0x4002_3000——0x4002_33FF
保留	0x4002_3400——0x4002_FFFF
SPI1	0x4003_0000——0x4003_03FF
SPI2	0x4003_0400——0x4003_07FF
SPI3	0x4003_0800——0x4003_0BFF
保留	0x4003_0C00——0x4003_0FFF
DMA1	0x4003_1000——0x4003_1FFF
DMA2	0x4003_2000——0x4003_2FFF
保留	0x4003_3000——0x4003_FFFF
保留	0x4004_0000——0x4004_03FF
保留	0x4004_0400——0x4004_07FF

保留	0x4004_0800——0x4004_0BFF
保留	0x4004_0C00——0x4004_0FFF
保留	0x4004_1000——0x4004_FFFF
USB2.0	0x4005_0000——0x4005_FFFF
保留	0x4006_0000——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
保留	0x4800_1800——0x4800_1BFF
保留	0x4800_1C00——0x4FFF_FFFF
ADC1/ADC2	0x5000_0000——0x5000_03FF
保留	0x5000_0400——0x5000_07FF
DAC1	0x5000_0800——0x5000_0BFF
保留	0x5000_0C00——0x5000_0FFF
保留	0x5000_1000——0x5000_13FF
保留	0x5000_1400——0x5000_17FF
保留	0x5000_1800——0x5005_FFFF
AES	0x5006_0000——0x5006_03FF
CORDIC	0x5006_0400——0x5006_07FF
HRNG	0x5006_0800——0x5006_0BFF
HASH	0x5006_0C00——0x5006_0FFF
保留	0x5006_1000——0x5FFF_FFFF
QSPI memory	
SPI3_MEM	0x9000_0000——0x9FFF_FFFF
External memory	
EXMC_MEM	0x6000_0000——0x6FFF_FFFF
EXMC_REG	0xA000_0000——0xA000_03FF

2.3.1. 片上 SRAM

芯片集成了多达 64KB 的 SRAM：

- 56KB SRAM1

- 8KB SRAM2，在 STOP2 低功耗模式下数据保持
- 支持字节、半字（16 位）以及字（32 位）访问
- 可在最高系统时钟频率下以 0 等待周期寻址
- 支持奇偶校验，校验出错时，能产生错误标志

2.3.2. 片上 FLASH

Flash 由主存储区和信息区（NVR 区）组成：

- 主存储区最大为 320KB
- NVR 区最大为 4KB，用于芯片配置

片上 Flash 的具体信息请参考 EFCFLASH 控制器章节。

2.4. BOOT 配置

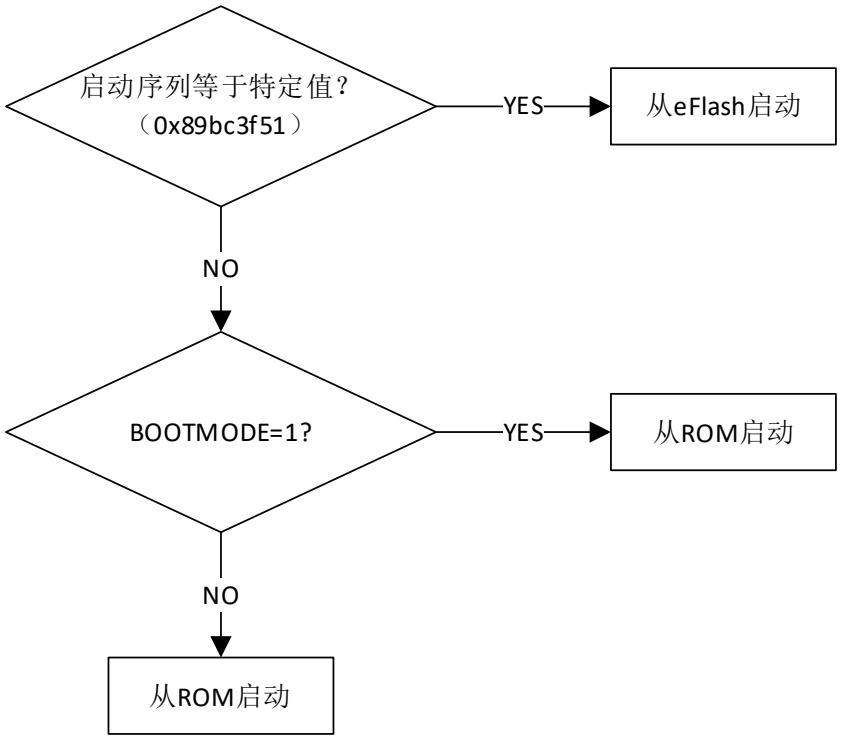
芯片有两种启动模式：ROM 启动和 eFlash 启动。

系统上电时，芯片会读取启动序列字段（位于 NVR 区域偏移地址：0x400）和系统寄存器 SYSCFG_WMR 的 BOOTMODE 标志位，决定是将 eFlash 还是将 ROM 映射到 0x0 地址。

BOOTMODE 标志位由上电时 BOOT 引脚的高低电平决定。复位后（上电复位、外部引脚复位、EFC 软复位）或从待机模式退出时，在 SYSCLK 的第四个上升沿锁存 BOOT 引脚的值到 SYSCFG_WMR 寄存器的 BOOTMODE 位，因此用户应确保复位或待机模式下 BOOT 引脚所需的电平状态。

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：0x0008_0208～0x0008_0214

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

地址	名称	描述
0x00	SYSCFG_SYSCR	系统控制寄存器
0x04	SYSCFG_WMR	工作模式寄存器
0x08	SYSCFG_VER	版本寄存器
0x0C	保留	
0x10	SYSCFG_PHYCR	USBPHY 模块控制寄存器
0x14	保留	

2.6.2. 系统控制寄存器(SYSCFG_SYSCR: 00h)

位域	名称	属性	默认值	功能描述
31:19	RSV	-	-	保留
18:17	IR_MODE	RW	00	IR_OUT 一端选择 TIM16_CH1 还是 UART1/2_TXD 00：IR_OUT=~(TIM17_CH1& TIM16_CH1) 01：IR_OUT=~(TIM17_CH1&UART1_TXD) 10：IR_OUT=~(TIM17_CH1&UART2_TXD) 11：保留
16	IR_POL	RW	0	IR_OUT 输出极性控制 0：原极性 1：取反
15:9	RSV	-	-	保留
8	SRAM_PEF	RCW1	0	SRAM 校验错误标志，写 1 清除 0：无 SRAM 校验错误 1：发生 SRAM 校验错误
7:4	RSV	-	-	保留
3	FLASH_PARITY_LOCK	RW	0	FLASH 奇偶校验错误锁定 0：FLASH 校验错误输出从 TIM1/8/15/16/17 的 break 输入端断开 1：FLASH 校验错误输出与 TIM1/8/15/16/17 的 break 输入端连接

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	SRAM_PARITY_LOCK	RW	0	SRAM 奇偶校验错误锁定 0：SRAM 校验错误输出从 TIM1/8/15/16/17 的 break 输入端断开 1：SRAM 校验错误输出与 TIM1/8/15/16/17 的 break 输入端连接
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 LOCKUP 说明：在 HardFault 中断函数中再次产生 HardFault，这个时候系统就会进入 LOCKUP 状态，对于这种异常状态只能通过 Reset 或者调试器介入来恢复

2.6.3. 工作模式寄存器(SYSCFG_WMR: 04h)

位域	名称	属性	默认值	功能描述
31:7	RSV	-	-	保留
6	RTC_READY	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	FT_DISABLE	RO	X	测试封口，用于表示测试封口状态。测试封口后，不能进入测试模式。 0：测试未封口，可以通过配置 TEST PIN 进入相应测试模式 1：测试封口，不能进入测试模式，始终工作在用户模式（正常工作）
4	REMAP_FLAG	RO	X	REMAP 标志，表示系统是工作在 ROM 模式还是 eFlash 模式： 0：系统工作在 ROM 模式（下载模式） 1：系统工作在 eFlash 模式（APP 运行模式） 该寄存器的值会反映在 REMAP 管脚上。
3	BOOTMODE	RO	X	BootMode 标志，用于选择芯片从 ROM 启动还是从 eFlash 启动，优先级低于启动序列字段。 0:指示系统从 eFlash 启动 1:指示系统从 ROM 启动（在启动序列不等于 0x89bc3f51 情况下）
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	0xFEDF0120	芯片版本 15~8：0x01 代表 MCU 产品 7~0：芯片版本，0x20 表示 V1 版本 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	默认值	功能描述
31:20	RSV	-	-	保留
19	CLKSEL_END	RO	X	USB PHY 外部时钟源查询标志（OSC_MODE[1]=0 有效） 0：查询中 1：查询完成 查到此位为 1 后表示时钟查询完成(查询时间约 20ms)，可以查询 CLKSEL_LRC_HXO 来确定时钟源选择结果
18	ADJ_END	RO	X	RC 校准完成（内部 RC 模式） 0：校准未完成 1：校准完成
17	PLL_LOCK	RO	X	PLL 锁定标志（外部晶振模式） 0：PLL 未锁定 1：PLL 锁定
16	CLKSEL_LRC_HXO	RO	X	时钟源选择标志 0：选择内部 RC 1：选择外部晶振
15:8	RSV	-	-	保留
7	DPPD	RW	0	USBPHY DP 下拉电阻使能，高有效
6	DMPD	RW	0	USBPHY DM 下拉电阻使能，高有效
5	RSV	-	-	保留
4:3	OSC_MODE[1:0]	RW	00	PHY 时钟选择 00：自动检测外部晶振（PHY 复位释放后 20ms 内检测外部晶振，超时选择内部 RC） 01：快速自动检测外部晶振（PHY 复位释放后 1us 内检测外部晶振，超时选择内部 RC） 10：强制选择内部 RC 11：强制选择外部晶振
2	PD_PLL	RW	1	控制 PLL 是否进入 Power Down 模式 0：PLL 正常工作 1：PLL 进入 PowerDown 模式
1	FPLL_H60M_L48M	RW	0	PHY 输出时钟 48M 和 60M 选择 0：48M 1：60M
0	PHY_RSTN	RW	1	USB_PHY 软复位，写 0 复位，写 1 撤销

3. 电源管理（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. 主要特性

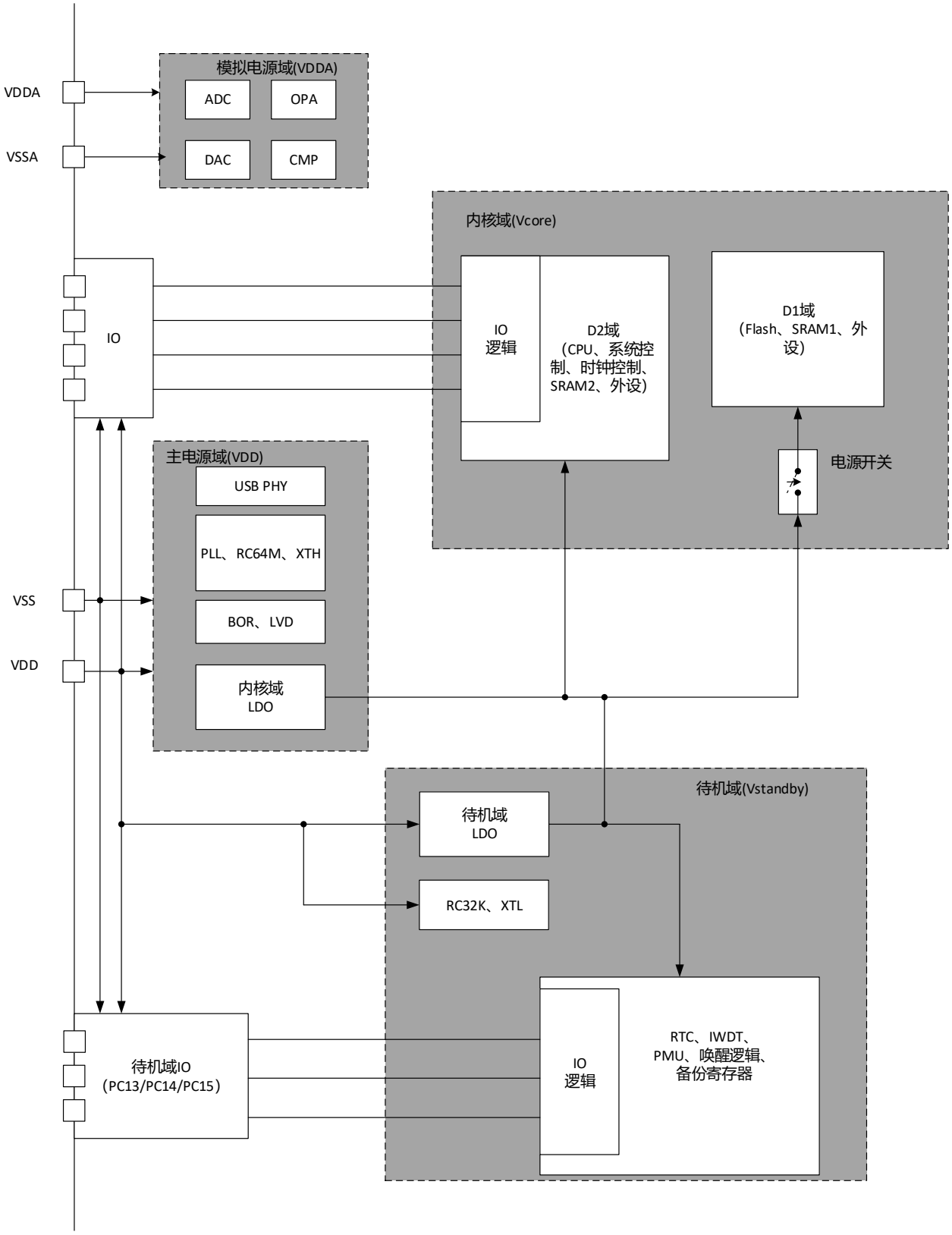
- 电源和电源域
  - 主电源域（VDD）
  - 模拟电源域（VDDA）
  - 内核域（Vcore）
  - 待机域（Vstandby）
- 系统电源稳压
  - 稳压器（内核域 LDO 和待机域 LDO）
- 电源监控
  - POR/PDR 监控器
  - BOR 监控器
  - LVD 监控器
- 电源管理
  - 工作模式
  - 电压调节控制
  - 低功耗模式

3.2. 供电电源

芯片的工作电压（VDD）范围 1.70~3.60V。如下图所示：

- VDD（1.70~3.60V）为 IO、稳压器、LVD、BOR、PLL、内部 RC 时钟等供电
- VDDA（1.70V~3.6V）为 ADC、DAC、OPA 和 CMP 等供电，当不使用这些外设时，最好将 VDDA 连接到 VDD 上
- VSSA，独立的模拟地
- VSS，芯片的公共地

图 3-1 电源管理结构框图



3.2.1. 主电源域（VDD）

VDD 电压输入范围为 1.7V~3.6V，主要给 IO、LDO（内核域 LDO 和待机域 LDO）、LVD、USB PHY、PLL、RC 时钟（RC64M 和 RC32K）、晶振时钟（XTH 和 XTL）等供电。

3.2.2. 模拟电源域（VDDA）

模拟电源域通过专用的 VDDA 和 VSSA 引脚供电，电压范围为 1.7V~3.6V ，给 ADC、DAC、OPA 和 CMP 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。

VDD 和 VDDA 压差的绝对值不要超过 0.4V。在许可范围内，压差越大，对模拟性能影响越大。如果 VDD 和 VDDA 来自同一个电源，可以用磁珠进行隔离。

3.2.3. 内核域（Vcore）

Vcore 内核域电源通过稳压器提供。

Vcore 域分为 2 个部分：

- D1 域中的 Flash、SRAM1 和外设
- D2 域中的 CPU、系统控制、时钟控制、IO 逻辑、SRAM2 和外设（EXTI、LPUART、LPTIM）

■ 稳压器

稳压器提供三种不同的工作模式：主模式（MR）、低功耗模式（LP）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 0

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。

- 停止模式 1

稳压器工作在低功耗模式（LP），为 Vcore 域供电以保存寄存器和存储器的内容，同时低功耗外设可以继续保持工作。稳压器输出电压可通过寄存器 PMU_CTRL0 的 LPLDO_LV 位来调节，可进一步降低系统在停止模式 1 下的功耗。稳压器工作在低功耗模式时，会增大系统从停止模式退出的时间。

- 停止模式 2

在停止模式 1 的基础上，D1 域掉电，进一步降低系统的功耗

- 待机模式

稳压器关闭且 Vcore 域掉电。

3.2.4. 待机域（Vstandby）

待机域包括以下模块：

- RC32K 时钟、XTL 低速晶振时钟
- PC13~PC15 I/O
- RTC、IWDT、PMU
- 唤醒逻辑
- 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。

3.3. 电源监控

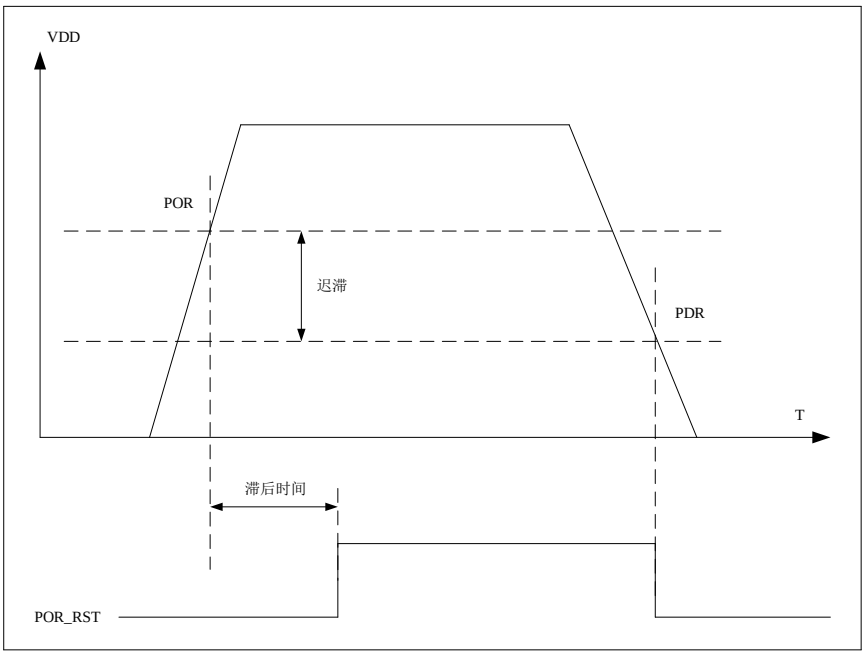
3.3.1. 上电复位(POR)/掉复位(PDR)

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. 欠电复位(BOR)

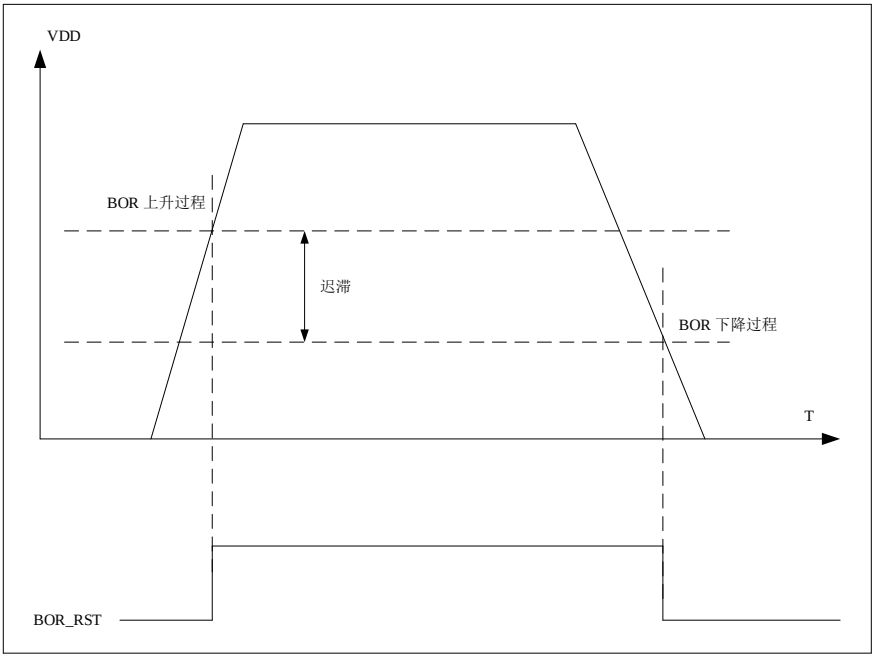
上电过程，欠电复位将芯片保持复位状态，直到 VDD 电源电压达到制定的 VBOR 阈值。

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



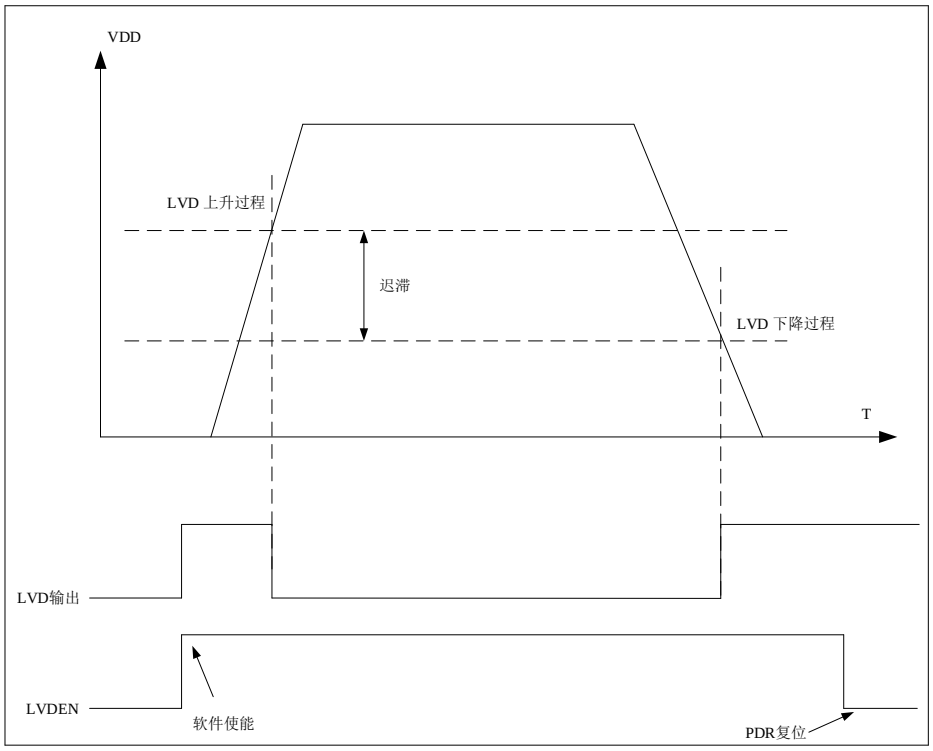
3.3.3. 电压检测模块(LVD)

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 0（STOP0）：RC64M 时钟可配置自动关闭；RC32K/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
- 停止模式 1（STOP1）：RC64M 时钟可配置自动关闭；RC32K/XTL 时钟下的外设可以工作；内核域 LDO 进入低功耗模式（LP），同时输出电压范围可通过寄存器来调节
- 停止模式 2（STOP2）：内核域中的 D1 域断电，D2 域中的寄存器保持掉电前的状态，后 8KB 的 SRAM 保持数据；内核域 LDO 进入低功耗模式（LP），同时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
- 断电模式（POWERDOWN）：内核域和待机域域断电，VDD 仍存在

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. STOP0

LDO 工作在主模式下，可通过寄存器 PMU_CTRL0 的 MLDO_LOWP 配置进一步降低功耗，以及 PMU_CTRL0 的 MLDO_LV 调节输出电压电平；通过配置寄存器 PMU_CTRL0 的 RC64MPDEN 控制进入 STOP0 模式后，RC64M 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 STOP0 模式 ➢ 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 STOP0 模式
退出模式	● 如果使用 WFI 指令或从 ISR 返回进入 STOP0 模式，在 NVIC 中使能的任何 EXTI 中断都能将系统从 STOP0 模式中唤醒 ● 如果使用 WFE 指令进入 STOP0 模式，系统将在有事件发生时立即退出 STOP0 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 STOP0 模式，唤醒后，程序重新从头开始执行

3.4.3. STOP1

LDO 工作在低功耗模式（LP）下，可通过寄存器 PMU_CTRL0 的 LPDO12_LV 调节输出电压电平可进一步降低功耗；通过配置寄存器 PMU_CTRL0 的 RC64MPDEN 控制进入 STOP1 模式后，RC64M 时钟是否自动关闭。

STOP1 模式	说明
进入模式	● 清除 NVIC 中断和事件，清除 Systick 中断 ● 设置系统时钟为 RC4M ● 配置寄存器 PMU_CTRL0 的 LPMS=1 ● 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 STOP1 模式 ➢ 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 STOP1 模式
退出模式	● 如果使用 WFI 指令或从 ISR 返回进入 STOP1 模式，在 NVIC 中使能的任何 EXTI 中断都能将系统从 STOP1 模式中唤醒 ● 如果使用 WFE 指令进入 STOP1 模式，系统将在有事件发生时立即退出 STOP1 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 STOP1 模式，唤醒后，程序重新从头开始执行

3.4.4. STOP2

LDO 工作在低功耗模式（LP）下，内核域中的 D1 域掉电，D2 域中的寄存器保持 STOP2 前的状态，后 8KB 的 SRAM 保持数据。

所有 I/O 不掉电并处于保持状态，同时 PC13/PC14/PC15 可通过 PMU_IOSEL 配置 I/O 的工作状态。

STOP2 模式	说明
进入模式	● 清除 NVIC 中断和事件，清除 Systick 中断 ● 设置系统时钟为 RC32K ● 配置寄存器 PMU_CTRL0 的 LPMS=2 ● 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 STOP2 模式 ➢ 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 STOP2 模式

退出模式	除 LVD、USB wakeup、GPIOA11、GPIOA12、COMP 的其他任何 EXTI 中断或事件都能将系统从 STOP2 模式中唤醒 ● RSTN 管脚复位/BOR 复位/IWDT 复位可将系统退出 STOP2 模式，唤醒后，程序将从停止的位置继续执行，并响应唤醒中断（见 NVIC）

3.4.5. 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=0b10x ● 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 唤醒引脚与 PAD 对应关系

唤醒引脚	PAD 引脚
WKUP1	PA0
WKUP2	PC13
WKUP3	PA2
WKUP4	PC5
WKUP5	PB5

3.4.6. POWER DOWN

内核域和待机域掉电，VDD 电源仍供电。

WAKEUP 引脚通过寄存器 PMU_CTRL2 和 PMU_CTRL3 进行使能和控制。

POWERDOWN 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=0b11x ● 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 POWERDOWN 模式 ➢ 如果 SLEEPONEXIT=1，系统将在退出优先级最低的 ISR 后进入 POWERDOWN 模式
退出模式	● WAKEUP 引脚（WKUP1 和 WKUP2，见**错误!未找到引用源。**）的有效电平/RSTN 管脚复位可将系统从 POWERDOWN 模式中唤醒； ● 唤醒后，系统将重新走上电过程

3.4.7. 模块状态

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	STOP0	STOP1	STOP2	STANDBY	POWN DOWN
CPU	Y	-	-	-	-	X	X
Flash	Y	Y	O	O	X	X	X
SRAM1	Y	Y	Y	Y	X	X	X
SRAM2	Y	Y	Y	Y	Y	X	X
备份寄存器	Y	Y	Y	Y	Y	Y	X
UART/I2C/SPI/I2S	O	O	-	-	X	X	X
LPUART	O	O	O	O	O	X	X
Timer/WDT	O	O	-	-	X	X	X
LPTIM	O	O	O	O	O	X	X
UAC（算法）	O	O	-	-	X	X	X
IWDT	O	O	O	O	O	O	X
USB	O	O	O	O	X	X	X
PLL	O	O	O	O	X	X	X
RC64M	O	O	O	O	X	X	X
RC32K	Y	Y	Y	Y	Y	Y	X
XTH	O	O	O	O	X	X	X
XTL	O	O	O	O	O	O	X
ADC	O	O	-	-	X	X	X
DAC	O	O	-	-	X	X	X
COMP	O	O	O	O	X	X	X
LVD	O	O	O	O	X	X	X

CORE LDO	Y	Y	Y	-	Y	X	X
STANDBY LDO	Y	Y	Y	Y	-	Y	X
RTC	O	O	O	O	O	O	X
GPIO	Y	Y	Y	Y	Y 注 2	Y 注 3	X 注 4

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：所有 GPIO 不掉电且处于保持状态，此外 PC13/PC14/PC15 还可通过 PMU_IOSEL 配置 I/O 的工作状态

注 3：除 PC13/PC14/PC15 外的所有 GPIO 均掉电（高阻态）；PC13/PC14/PC15 可通过 PMU_IOSEL 配置保持或运行状态；5 个 WAKEUP 引脚（WKUP1~WKUP5 详见**错误!未找到引用源。**）可作为唤醒源

注 4：所有 GPIO 都掉电（高阻态）；2 个 WAKEUP 引脚（WKUP1~WKUP2 详见**错误!未找到引用源。**）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		PMU 控制寄存器 0
0x04	PMU_CTRL1		PMU 控制寄存器 1
0x08	PMU_CTRL2		PMU 控制寄存器 2
0x0C	PMU_CTRL3		PMU 控制寄存器 3
0x10	PMU_SR		PMU 状态寄存器
0x14	PMU_STCLR		PMU 状态清除寄存器
0x18	PMU_IOSEL		STANDBY 域 IO 复用寄存器

3.5.2. 控制寄存器 0(PMU_CTRL0: 00h)

位域	名称	属性	默认值	功能描述
31:28	RSV	-	-	保留
27:16	STOP_WPT	RW	0x3C0	STOP0/STOP1/STOP2 唤醒时间设置（时钟周期数），以等待内核域 LDO 工作稳定（典型时间 15us） 1、STOP0/STOP1 模式唤醒以 RC64M 时钟计数或以 RC64M 的 16 分频时钟计数（当 RC64MDIVEN 使能时） 2、STOP2 模式唤醒以 RC32K 时钟计数
15	RSV	-	-	保留

14:12	LPLDO_LV	RW	100	在 STOP1/STOP2 模式下，内核域 LDO 进入低功耗模式（LP）；在 STANDBY 模式下，待机域的 LDO 工作且为低功耗模式； 此时 STOP1/STOP2/STANDBY 模式下的 LPLDO 输出电压可进一步配置调节 000：0.948V 001：0.985V 010：1.02V 011：1.06V 100：1.09V 101：1.13V 110：1.16V 111：1.20V
11	RSV	-	-	保留
10	MLDO_LOWP	RW	0	STOP0 模式下，内核域 LDO 进入主模式（MR），此时 LDO（MLDO）可配置是否进一步降低功耗 0：STOP0 下 MLDO 不降低自身功耗 1：STOP0 下 MLDO 降低自身功耗
9:8	MLDO_LV	RW	00	STOP0 模式下，LDO（MLDO）除了进一步降低功耗外（MLDO_LOWP=1），其输出电压可再配置调节 00：1.2V 01：1.0V 10：0.9V 11：0.8V
7	RTC_WE	RW	0	RTC 模块写保护位 0: RTC 模块写禁止 1: RTC 模块写使能
6	RSV	-	-	保留
5	RCHDIVEN	RW	0	STOP0/STOP1 模式下，RCH 是否自动切换为 16 分频输出 0：继续保持原分频选择（由 RCC_RCHCR.RCHDIV 确定） 1：切换 16 分频输出
4	RCHPDEN	RW	0	STOP0/STOP1 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	RSV	-	-	保留
2:0	LPMS	RW	000	低功耗模式选择位（内核进 DeepSleep 时有效）： 000：STOP0 模式 001：STOP1 模式 010：STOP2 模式 011：保留 10x：STANDBY 模式 11x：POWER DOWN 模式

3.5.3. 控制寄存器 1(PMU_CTRL1: 04h)

位域	名称	属性	默认值	功能描述
31:15	RSV	-	-	保留
14	LVDF	RO	0	LVD 低电压检测标志 0：VDD 电压等于或高于通过 LVD_SEL 位选择的 LVD 阈值 1：VDD 电压低于通过 LVD_SEL 位选择的 LVD 阈值
13:12	RSV	-	-	保留
11:9	FLT_TIME	RW	000	LVD 滤波时间配置，滤波时间按 FLTCLK 计算，FLTCLK 的时钟源可以为 PCLK 的 32 分频或 RCL，由 RCC 的 CCR2 的 bit18 来选择。 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 注意： 写入时，bit[11:9]表示 FLT_TIME 读取时，bit[12:10]表示 FLT_TIME
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 STOP0/STOP1 模式下用 LVD 唤醒，需要禁止数字滤波或者使用 RCL 时钟滤波 注意： 写入时，bit[8]表示 LVD_FLTEN 读取时，bit[9]表示 LVD_FLTEN
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：使能后，LVD 输出需要 2us 的稳定时间，在使能 2us 内可能误报警。

3.5.4. 控制寄存器 2(PMU_CTRL2: 08h)

位域	名称	属性	默认值	功能描述
31:28	RSV	-	-	保留
27:26	BOR_CFG	RW	00	BOR 复位阈值电压范围选择 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	RSV	-	-	保留
18:16	STDBY_WPT	RW	010	STANDBY 模式唤醒等待时间设置，以等待内核域 LDO 工作稳定（典型时间 15us） 000：等待 1 个 RC32K 001：等待 2 个 RC32K 010：等待 3 个 RC32K … 110：等待 7 个 RC32K 111：无效，不等待
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 个 RTCCLK 脉宽 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. 控制寄存器 3 (PMU_CTRL3: 0Ch)

位域	名称	属性	默认值	功能描述
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. 状态寄存器(PMU_SR: 10h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
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 如果 BORWUF 不清除，则进入 STANDBY 模式后会被立刻唤醒，且 SBF 位置 1
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 如果 IWDTWUF 不清除，则进入 STANDBY 模式后会被立刻唤醒，且 SBF 位置 1
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 如果 RSTWUF 不清除，则进入 STANDBY 模式后会被立刻唤醒，且 SBF 位置 1
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 如果 WUPFx 标志未清除，则进入 STANDBY 模式后会被立刻唤醒，且 SBF 位置 1

3.5.7. 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
31:11	RSV	-	-	保留
10	PC15_VALUE	RW	0	当 PC15_SEL=0b01 时，本位域控制 PC15 输出电平（以 PP 模式输出）： 0：PC15 输出低 1：PC15 输出高 注：读取的值为之前写入的值。

9	PC14_VALUE	RW	0	当 PC14_SEL=0b01 时，本位域控制 PC14 输出电平（以 PP 模式输出）： 0：PC14 输出低 1：PC14 输出高 注：读取的值为之前写入的值。
8	PC13_VALUE	RW	0	当 PC13_SEL=0b11 时，本位域控制 PC13 输出电平（以 PP 模式输出）： 0：PC13 输出低 1：PC13 输出高 当 PC13_SEL=0b01 时，本位域控制 PC13 输出类型： 0：PC13 为 OD 模式输出 1：PC13 为 PP 模式输出 注：读取的值为之前写入的值。
7	RSV	-	-	保留
6:5	PC15_SEL	RW	0	PC15 管脚功能选择 00：GPIO 功能，STANDBY 模式下为输入状态 01：输出 PC15_ VALUE，以 PP 模式输出 其它：输出关闭
4:3	PC14_SEL	RW	0	PC14 管脚功能选择 00：GPIO 功能，STANDBY 模式下为输入状态 01：输出 PC14_ VALUE，以 PP 模式输出 其它：输出关闭
2	RSV	-	-	保留
1:0	PC13_SEL	RW	0	PC13 管脚功能选择 00：GPIO 功能，STANDBY 模式下为输入状态 01：RTC 频率信号输出，根据 RTC_CR.FSEL 选择，输出对应的频率信号。此时 PC13_VALUE 控制 PC13 的输出类型（OD 或 PP 输出） 10：RTC 输入，作为时间戳功能（Tamper） 11：输出 PC13_VALUE，以 PP 模式输出

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 的配置将会忽略。

4. 复位和时钟单元（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

表 4-1 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，不影响待机域；
欠电复位（BOR）	当芯片供电电压 VDD 低于可编程的阈值电压时，产生欠电复位；复位内核域和 IWDT； 除待机域中的 IWDT 模块外，不影响待机域的其他模块； 芯片处于 STANDBY 低功耗模式时，BOR 复位唤醒后，复位情况同 POR12

注：电源域的划分详见电源管理 PMU 章节

4.1.2. 系统复位

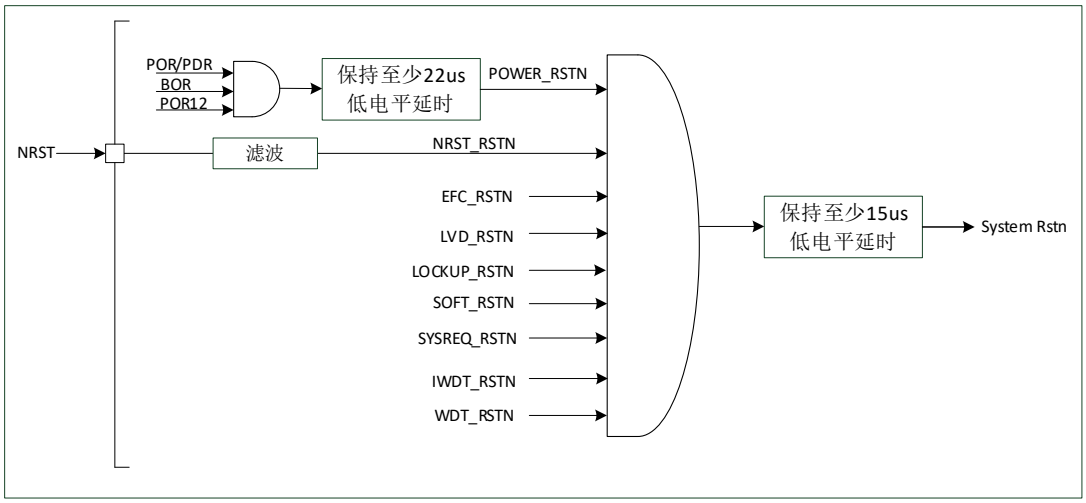
表 4-2 系统复位

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 EFC 控制器和 RCC 寄存器）； 不影响待机域； 芯片处于 STANDBY 低功耗模式时，NRST 复位唤醒后，复位情况同 POR12
EFC_RSTN	EFC 控制器软复位（配置寄存器 RCC_RCR 中的 EFCRST 位）；复位内核域（包括 EFC 控制器和 RCC 寄存器）； 不影响待机域；
LVD_RSTN	低压检测复位（使能寄存器 RCC_RCR 中的 LVDRSTEN 位）；复位内核域，不影响 EFC 控制器和 RCC 寄存器； 不影响待机域；

LOCKUP_RSTN	CPU LOCKUP 复位（使能寄存器 RCC_RCR 中的 LOCKRSTEN 位）；复位内核域，不影响 EFC 控制器和 RCC 寄存器； 不影响待机域；
SOFT_RSTN	软件复位；复位内核域，不影响 EFC 控制器和 RCC 寄存器； 不影响待机域；
SYSREQ_RSTN	CPU SYSREQ 复位；复位内核域，不影响 EFC 控制器和 RCC 寄存器； 不影响待机域；
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，不影响 EFC 控制器和 RCC 寄存器； 除待机域的 IWDT 模块外，不影响待机域的其他模块； 芯片处于 STANDBY 低功耗模式时，IWDT 复位唤醒后，除 IWDT 模块，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，不影响 EFC 控制器和 RCC 寄存器； 不影响待机域；

注：电源域的划分详见电源管理 PMU 章节。

图 4-1 系统复位电路



4.1.3. 待机域复位

表 4-3 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_STDBYCTRL）中的 STDBYRST 位产生复位待机域（除待机域的 IWDT 模块及 RCC_STDBYCTRL 寄存器）

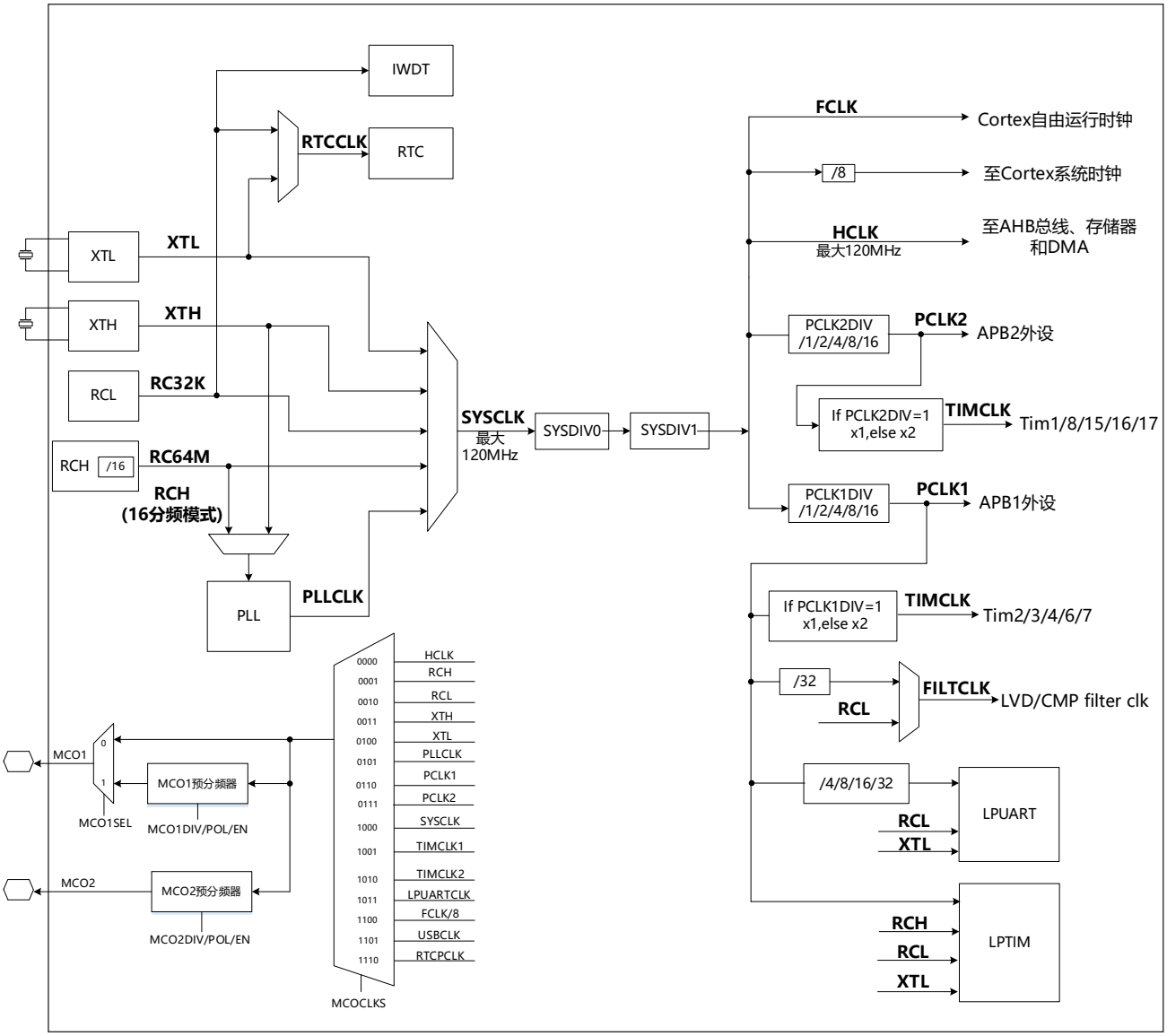
注：电源域的划分详见电源管理 PMU 章节

4.2. 时钟

RCC 有多种时钟源可供选择

- RCH（内部高速 RC 振荡器）时钟：64MHz
- RCL（内部低速 RC 振荡器）时钟：32KHz
- 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：最高 120MHz
- 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。

图 4-2 时钟结构图



注 1：SYSDIV0/SYSDIV1/PCLK1DIV/PCLK2DIV 等分频配置详见 RCC_CCR2 寄存器

注 2：AHB/APB1/APB2 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

AHB 的时钟频率与系统频率一致最高 120MHz，APB1 和 APB2 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 和 APB2 的时钟频率最高也可达 120MHz）。

RCC 通过 AHB 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 HCLK 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 时钟有内部 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 4 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。

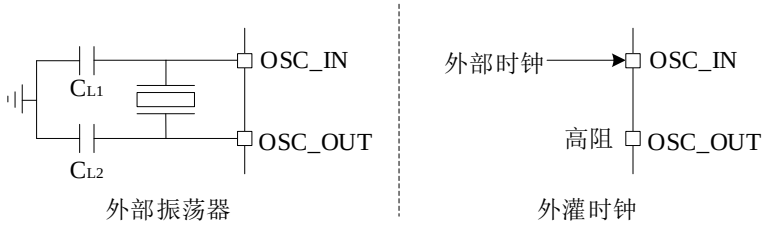
如果用户的应用基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。可以通过寄存器 RCC_STDBYCTRL 的 RCLTRIM 位来调整 RCL 的频率。

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

图 4-3 XTH 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 。

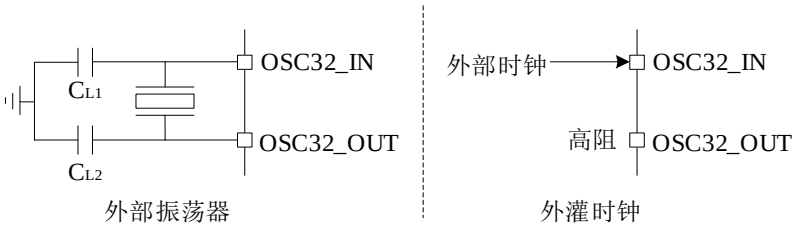
4.2.4. XTL 时钟

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

图 4-4 XTL 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。

■ XTH 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

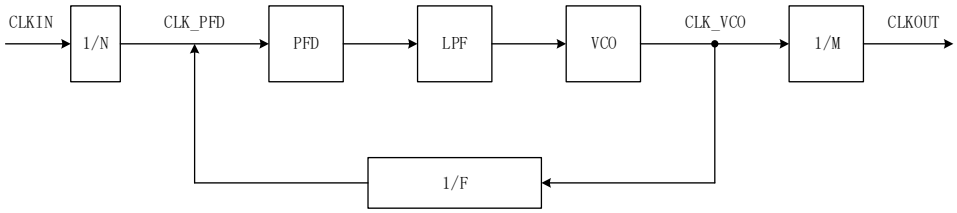
内部 PLL 可以用来倍频 RCH 或 XTH 的输出时钟。

PLL 具有以下特性：

- 时钟输入（CLKIN）频率范围：3MHz~48MHz
- PFD 输入（CLK_PFD）频率范围：3MHz~6MHz
- VCO 输出（CLK_VCO）频率范围：75MHz~150MHz
- 时钟输出（CLKOUT）频率范围：6MHz~150MHz

PLL 功能框图如下：

图 4-5 PLL 功能框图



PLL 输出公式为:

Fpll=(Fin* (PLLF+15)/(PLLN+1))/ 2PLLM

PLLN、PLLM、PLLF 由寄存器 RCC_PLLCR 配置。

■ PLL 配置流程

1）使能 PLL（将寄存器 RCC_PLLCR 的 PLLEN 位置 1）

2）退出 PLL 休眠模式（配置寄存器 RCC_PLLCR 的 PLLSLEEP 位为 0）

3）等待寄存器 RCC_PLLCR 的 PLLFREERUN 位置 1；

4）若需改变 PLL 输出频率，可重新配置寄存器 RCC_PLLCR 的 PLLN、PLLM 和 PLLF 位

5）将寄存器 RCC_PLLCR 的 PLLUPDATEEN 位置 1

6）等待寄存器 RCC_PLLCR 的 PLLLOCK 位或 PLLFREERUN 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 PLLLOCK 位不可靠，此时可查询 PLLFREERUN 位来判断 PLL 是否正常工作，PLLFREERUN 位跟 PLLRUNDLY 有关，PLL 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_PLLCR 的 PLLLOCKSEL 位选择 PLL 中断源是 PLLLOCK 还是 PLLFREERUN）。

4.2.6. 系统时钟

系统时钟（SYSCLK）有五种时钟源：RCH、RCL、XTH、XTL、PLL。在系统复位后，默认系统时钟为 RCH。在直接使用 RCH 或通过 PLL 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 4-4 系统时钟源

SYSCLKSEL	系统时钟源
000	RCH
001	RCL
010	XTH
011	XTL
100	PLLCLK

4.2.7. 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。

4.2.8. 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	描述
0x00	RCC_RCR	复位控制寄存器
0x04	RCC_RSR	复位源状态寄存器
0x08	RCC_APB1_IPRSTR	APB1 外设模块复位寄存器
0x0C	RCC_APB2_IPRSTR	APB2 外设模块复位寄存器
0x10	RCC_AHB_IPRSTR	AHB 外设模块复位寄存器
0x14	RCC_CCR1	时钟控制寄存器 1
0x18	RCC_CCR2	时钟控制寄存器 2

0x1C	RCC_CIR	时钟中断寄存器
0x20	RCC_APB1_IPCKENR	APB1 外设模块使能寄存器
0x24	RCC_APB2_IPCKENR	APB2 外设模块使能寄存器
0x28	RCC_AHB_IPCKENR	AHB 外设模块使能寄存器
0x2C	RCC_RCHCR	RCH 控制寄存器
0x30	RCC_XTHCR	XTH 控制寄存器
0x34	RCC_PLLCR	PLL 控制寄存器
0x38	RCC_CLKOCR	时钟输出控制寄存器
0x3C	RCC_STDBY_CTRL	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	SRSTMAP	W	1	SOFT RST Map： Soft reset 强制芯片从 Flash 中启动 复位除了模拟 IP、EFC、SENSOR 和系统寄存器模块的整个系统，下降沿触发。写 0 产生系统复位，下一个时钟自动回 1。
30	SRSTNOMAP	W	1	SOFT RST NO MAP: 复位能力同 SRSTMAP，但不更改 REMAP 状态。
29	EFCRST	W	1	eFlash 软复位，芯片重新从 ROM 启动。 0：复位 1：不复位
28:17	RSV	-	-	保留
16	REMAPIMP	RW	0	REMAP 立刻使能 0：无效 1：写 1 立刻使能 REMAP，下次 REMAP 先写 0 再写 1。 （Remap 机制请参见存储架构章节）
15:4	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	0	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 LVD 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 RSTFLAGCLR 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
15:9	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	SRSTNMF	RO	0	SOFT reset NO Map status： 0:上一次复位不是由 No Map 系统软复位设置引起 1:上一次复位由 No Map 系统软复位设置引起
6	EFCRSTF	RO	0	EFC Reset 标志： 0:上一次复位不是由 EFC 复位引起 1:上一次复位由 EFC 复位引起
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. APB1 外设模块复位控制寄存器(RCC_APB1RSTR : 08h)

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 写 0 复位，写 1 退出复位。
30	LPTIM1RST	RW	1	LPTIM1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
29:28	RSV	-	-	保留
27	PMURST	RW	1	PMU 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
26	RSV	-	-	保留
25	CAN2RST	RW	1	CAN2 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
24	CAN1RST	RW	1	CAN1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
23	RSV	-	-	保留
22	I2C2RST	RW	1	I2C2 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
21	I2C1RST	RW	1	I2C1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。

20	RSV	-	-	保留
19	UART4RST	RW	1	UART4 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
18	UART3RST	RW	1	UART3 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
17	UART2RST	RW	1	UART2 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
16	RSV	-	-	保留
15	I2S2RST	RW	1	I2S2 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
14	I2S1RST	RW	1	I2S1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
13:12	RSV	-	-	保留
11	WDTRST	RW	1	WDT 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
10:6	RSV	-	-	保留
5	TIM7RST	RW	1	TIM7 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
4	TIM6RST	RW	1	TIM6 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
3	RSV	-	-	保留
2	TIM4RST	RW	1	TIM4 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。

1	TIM3RST	RW	1	TIM3 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
0	TIM2RST	RW	1	TIM2 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。

4.3.5. APB2 外设模块复位控制寄存器(RCC_APB2RSTR: 0Ch)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	TIM17RST	RW	1	TIM17 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
12	TIM16RST	RW	1	TIM16 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
11	TIM15RST	RW	1	TIM15 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
10	RSV	-	-	保留
9	UART1RST	RW	1	UART1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
8	TIM8RST	RW	1	TIM8 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
7	RSV	-	-	保留
6	TIM1RST	RW	1	TIM1 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
5	RSV	-	-	保留

4	EXTIRST	RW	1	EXTI 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
3	OPARST	RW	1	OPA 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
2	COMPRST	RW	1	COMP 模块复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。
1	RSV	-	-	保留
0	SYSCFGRST	RW	1	SYSCFG 复位 0：复位 1：不复位 写 0 复位，写 1 退出复位。

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

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29	EXMCRST	RW	1	EXMC 模块复位 0：复位 1：不复位
28	RSV	-	-	保留
27	UACRST	RW	1	UAC 模块复位 0：复位 1：不复位
26:21	RSV	-	-	保留
20	DAC1RST	RW	1	DAC1 模块复位 0：复位 1：不复位
19	RSV	-	-	保留
18	ADC12RST	RW	1	ADC1/ADC2 模块复位 0：复位 1：不复位
17:16	RSV	-	-	保留

15	GPIOFRST	RW	1	GPIOF 模块复位 0：复位 1：不复位
14	GPIOERST	RW	1	GPIOE 模块复位 0：复位 1：不复位
13	GPIODRST	RW	1	GPIOD 模块复位 0：复位 1：不复位
12	GPIOCRST	RW	1	GPIOC 模块复位 0：复位 1：不复位
11	GPIOBRST	RW	1	GPIOB 模块复位 0：复位 1：不复位
10	GPIOARST	RW	1	GPIOA 模块复位 0：复位 1：不复位
9	RSV	-	-	保留
8	SPI3RST	RW	1	SPI3 模块复位 0：复位 1：不复位
7	SPI2RST	RW	1	SPI2 模块复位 0：复位 1：不复位
6	SPI1RST	RW	1	SPI1 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	USBRST	RW	1	USB 控制器复位 0：复位 1：不复位
3:1	RSV	-	-	保留
0	DMARST	RW	1	DMA1/DMA2 模块复位 0：复位 1：不复位

4.3.7. 时钟控制寄存器 1(RCC_CCR1: 14h)

位域	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	SYSCLKSEL	RW	000	系统时钟 SYSCLK 选择： 000：来自 RCH 001：来自 RCL 010：来自 XTH 011：来自 XTL 1xx：来自 PLLCLK 注：在进入 STOP2 模式前，需要将系统时钟切换到 RCL 时钟

4.3.8. 时钟控制寄存器 2(RCC_CCR2 : 18h)

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
30:22	RSV	-	-	保留
21:20	LPTIM1CLKSEL	RW	00	LPTIM1 时钟选择 00：PCLK 01：RCL 10：RCH 11：XTL
19	RSV	-	-	保留
18	FILTCLKSEL	RW	0	FLTCLK 选择，LVD 和 COMP 的滤波 时钟选择 0：PCLK 的 32 分频 1：RCL
17:16	LPUART1CLKSEL	RW	00	LPUART1 时钟选择 00: RCL 01: XTL 1x: PCLK 分频，由 LPUARTDIV 确定
15:14	LPUARTDIV	RW	11	LPUART1 使用 PCLK 分频选择 00:4 分频 01:8 分频 10:16 分频 11:32 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 SYSCLKDIV0 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。

4.3.9. 时钟中断寄存器(RCC_CIR: 1Ch)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留

20	PLLLOCKIC	WO	0	PLL LOCK 中断清除。 软件写 1 清除中断 PLLLOCKIF 0：无影响 1：清除 PLL 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
15:13	RSV	-	-	保留
12	PLLLOCKIE	RW	0	PLL LOCK 中断使能 0：禁止 PLL LOCK 中断 1：使能 PLL 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 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
7:5	RSV	-	-	保留
4	PLLLOCKIF	RO	0	PLL LOCK 中断标志 当 PLL 模块 LOCK，PLLLOCK(或 PLLFREERUN)标志有效后硬件置位。软件设置 PLLLOCKIC 清除 0：无 PLL LOCK 中断 1：产生 PLL 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

4.3.10. APB1 外设模块时钟使能寄存器(RCC_APB1ENR: 20h)

位域	名称	属性	复位值	描述
31	LPUART1CKEN	RW	0	LPUART1 模块时钟使能 0：禁止 1：使能
30	LPTIM1CKEN	RW	0	LPTIM1 模块时钟使能 0：禁止 1：使能
29:28	RSV	-	-	保留
27	PMUCKEN	RW	0	PMU 模块时钟使能 0：禁止 1：使能 注：ROM BOOT 中代码把 PMU 模块时钟使能了。
26	RSV	-	-	保留
25	CAN2CKEN	RW	0	CAN2 模块时钟使能 0：禁止 1：使能
24	CAN1CKEN	RW	0	CNA1 模块时钟使能 0：禁止 1：使能
23	RSV	-	-	保留

22	I2C2CKEN	RW	0	I2C2 模块时钟使能 0：禁止 1：使能
21	I2C1CKEN	RW	0	I2C1 模块时钟使能 0：禁止 1：使能
20	RSV	-	-	保留
19	UART4CKEN	RW	0	UART4 模块时钟使能 0：禁止 1：使能
18	UART3CKEN	RW	0	UART3 模块时钟使能 0：禁止 1：使能
17	UART2CKEN	RW	0	UART2 模块时钟使能 0：禁止 1：使能
16	RSV	-	-	保留
15	I2S2CKEN	RW	0	I2S2 模块时钟使能 0：禁止 1：使能
14	I2S1CKEN	RW	0	I2S1 模块时钟使能 0：禁止 1：使能
13:12	RSV	-	-	保留
11	WDTCKEN	RW	0	WDT 模块时钟使能 0：禁止 1：使能
10	RTCCKEN	RW	0	RTC 模块总线时钟使能 0：禁止 1：使能
9:6	RSV	-	-	保留
5	TIM7CKEN	RW	0	TIM7 模块时钟使能 0：禁止 1：使能
4	TIM6CKEN	RW	0	TIM6 模块时钟使能 0：禁止 1：使能
3	RSV	-	-	保留
2	TIM4CKEN	RW	0	TIM4 模块时钟使能 0：禁止 1：使能

1	TIM3CKEN	RW	0	TIM3 模块时钟使能 0：禁止 1：使能
0	TIM2CKEN	RW	0	TIM2 模块时钟使能 0：禁止 1：使能

4.3.11. APB2 外设模块时钟使能寄存器(RCC_APB2ENR: 24h)

位域	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	TIM17CKEN	RW	0	TIM17 模块时钟使能 0：禁止 1：使能
12	TIM16CKEN	RW	0	TIM16 模块时钟使能 0：禁止 1：使能
11	TIM15CKEN	RW	0	TIM15 模块时钟使能 0：禁止 1：使能
10	RSV	-	-	保留
9	UART1CKEN	RW	0	UART1 模块时钟使能 0：禁止 1：使能
8	TIM8CKEN	RW	0	TIM8 模块时钟使能 0：禁止 1：使能
7	RSV	-	-	保留
6	TIM1CKEN	RW	0	TIM1 模块时钟使能 0：禁止 1：使能
5	RSV	-	-	保留
4	EXTICKEN	RW	0	EXTI 模块时钟使能 0：禁止 1：使能
3	OPACKEN	RW	0	OPA 模块时钟使能 0：禁止 1：使能

2	COMPCKEN	RW	0	COMP 模块时钟使能 0：禁止 1：使能
1	RSV	-	-	保留
0	SYSCFGCKEN	RW	1	SYSCFG 时钟使能 0：禁止 1：使能

4.3.12. AHB 外设模块时钟使能寄存器(RCC_AHBENR: 28h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29	EXMCCKEN	RW	0	EXMC 模块时钟使能 0：禁止 1：使能
28	ROMCKEN	RW	1	ROM 模块时钟使能 0：禁止 1：使能
27	CORDICCKEN	RW	0	CORDIC 模块时钟使能 0：禁止 1：使能
26	HRNGCKEN	RW	0	HRENG 模块时钟使能 0：禁止 1：使能
25	AESCKEN	RW	0	AES 模块时钟使能 0：禁止 1：使能
24:21	RSV	-	-	保留
20	DAC1CKEN	RW	0	DAC1 模块时钟使能 0：禁止 1：使能
19	RSV	-	-	保留
18	ADC12CKEN	RW	0	ADC1/ADC2 模块时钟使能 0：禁止 1：使能
17:16	RSV	-	-	保留
15	GPIOFCKEN	RW	0	GPIOF 模块时钟使能 0：禁止 1：使能

14	GPIOECKEN	RW	0	GPIOE 模块时钟使能 0：禁止 1：使能
13	GPIODCKEN	RW	0	GPIOD 模块时钟使能 0：禁止 1：使能
12	GPIOCCKEN	RW	0	GPIOC 模块时钟使能 0：禁止 1：使能
11	GPIOBCKEN	RW	0	GPIOB 模块时钟使能 0：禁止 1：使能
10	GPIOACKEN	RW	0	GPIOA 模块时钟使能 0：禁止 1：使能
9	RSV	-	-	保留
8	SPI3CKEN	RW	0	SPI3 模块时钟使能 0：禁止 1：使能
7	SPI2CKEN	RW	0	SPI2 模块时钟使能 0：禁止 1：使能
6	SPI1CKEN	RW	0	SPI1 模块时钟使能 0：禁止 1：使能
5	CRCCKEN	RW	0	CRC 模块时钟使能 0：禁止 1：使能
4	USBCKEN	RW	0	USB 控制器时钟使能 0：禁止 1：使能
3	EFCCKEN	RW	1	eFlash 模块在 SLEEP 模式下时钟使能 0：禁止 1：使能
2	SRAMCKEN	RW	1	SRAM 在 SLEEP 模式下时钟使能 0：禁止 1：使能
1	DMA2CKEN	RW	0	DMA2 模块时钟使能 0：禁止 1：使能

0	DMA1CKEN	RW	0	DMA1 模块时钟使能 0：禁止 1：使能

4.3.13. RCH 模块控制寄存器(RCC_RCHCR: 2Ch)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16:8	RCHTRIM	RW	0x000	RCH 时钟 TRIM 值 TRIM 值越高频率越高
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

4.3.14. XTHCR 模块控制寄存器(RCC_XTHCR: 30h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	XTHRDY	RO	0	XTH 振荡器 ready。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 0：禁止 1：使能

4.3.15. PLL 模块控制寄存器(RCC_PLLCR: 34h)

位域	名称	属性	复位值	描述
31	PLLLOCKSEL	RW	0	PLL LOCK 信号选择，用于产生 PLL LOCK 中断 0：选择 PLLLOCK（模拟输出） 1：选择 PLLFREERUN（数字计数）
30	PLLFREERUN	RO	0	PLL 工作状态寄存器。 该位即时地反映 PLL 的工作状态，当 PLL 进入 SLEEP 模式或重新配置频率后，硬件自动清除该位；当 PLL 退出 SLEEP 模式且配置生效后，等待 PLLRUNDLY 设置时间后，硬件自动置位。 1：PLL 正常工作。 0：PLL 正常不正常。
29	PLLLOCK	RO	0	PLL 锁定状态位。 1：PLL 已经锁定。 0：PLL 未锁定。
28:23	PLLRUNDLY	RW	0x10	PLL 锁定等待时钟周期数。 PLL 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： PLLLOCKDLY * 512 * （1/PCLK） PCLK 为 32M 时，1 代表等待约 16us。
22	PLLUPDATEEN	RW	0	PLL 配置更新控制位，下一个周期自动回 0。重新改写 PLLM、PLLN 和 PLLF、PLLSRCSEL 后，需要该该比特置位新的配置才能生效。复位时间 60*（1/PCLK） 1：PLL 配置更新。 0：PLL 配置不更新。
21	PLLSLEEP	RW	1	PLL 休眠控制位（PLL 处于复位状态） 1：PLL 进入休眠。 0：PLL 不进入休眠。 注：先使能 PLL，再退出休眠。
20:18	RSV	-	-	保留
17:16	PLLM	RW	01	输出分频控制字段。PLLM 含有输出分频控制字段。对应分频值为 2 的 PLLM 次方。
15	RSV	-	-	保留
14:12	PLLN	RW	001	降频因子分频器字段。PLLN 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（PLLN+1）
11:8	RSV	-	-	保留

7:3	PLLF	RW	0x01	增频因子分频器字段。PLLF 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(PLLF+15)。 PLL 输出公式为: Fpll= (Fin* (PLLF+15)/(PLLN+1))/ 2PLLM PLL 输入时钟 Fin 频率范围 3~48MHz Fin/(PLLN+1)频率范围 3~6Mhz Fvco= Fin* (PLLF+15)/(PLLN+1)，频率范围在 75~150MHz。
2:1	PLLSRCSEL	RW	00	PLL 时钟源选择 0x：选择片内 RCH（需设置 16 分频模式） 1x：选择片外 XTH
0	PLLEN	RW	0	PLL 模块使能 0: 不使能，PLL 模块处于 power down 状态 1: 使能，PLL 模块使能

4.3.16. 时钟输出控制寄存器(RCC_CLKOCR: 38h)

位域	名称	属性	复位值	描述
31	MCO2EN	RW	1	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	RSV	-	-	保留
20:5	MCO1DIV	RW	0x004F	MCO1 预分频器分频系数： 分频数为寄存器值 + 1
4	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

3:0	MCOCLKS	RW	0000	MCO1/MCO2 时钟源选择 0000：HCLK 0001：RCH 0010：RCL 0011：XTH 0100：XTL 0101：PLLCLK 0110：PCLK1 0111：PCLK2 1000：SYSCLK 1001：TIMCLK1（APB1 总线分频时，为 PCLK1 的 2 倍；否则，与 PCLK1 同频） 1010：TIMCLK2（APB2 总线分频时，为 PCLK2 的 2 倍；否则，与 PCLK2 同频） 1011：LPUART1CLK 1100：FCLK/8 1101：USBPHY_PLL48M 1110：RTC_PCLK（与 PCLK1 同频） 其他：无时钟

4.3.17. STANDBY 电源域控制寄存器(RCC_STDBYCTRL: 3Ch)

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 注：BOR_EN/BOR_CFG/BORRST_EN 不会被复位
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	RSV	-	-	保留
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	Standby 模式下 RCL 时钟使能位 0：Standby 模式下 RCL 不使能 1：Standby 模式下 RCL 使能 注：芯片在非 Standby 模式下，RCL 时钟硬件固定使能，软件无法禁止
7:6	RSV	-	-	保留
5:3	XTLDRV	RW	111	XTL 振荡器驱动能力选择 XTLDRV[2]决定功耗模式： 0：正常功耗 1：低功耗 XTLDRV[1:0]决定功耗： 00：XTL 驱动 0（最低驱动） 01：XTL 驱动 1 10：XTL 驱动 2 11：XTL 驱动 3（最高驱动）
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：进行写操作后，需要等待至少 1 个 AHB 周期后才能读取

5. 嵌套矢量中断控制器（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 56 路可屏蔽向量中断（不包含内核的 16 个中断线）
- 16 个可编程中断优先级
- 低延迟的异常和中断处理
- 电源管理
- 系统控制寄存器的实现
- 包括内核异常在内的所有中断均通过 NVIC 进行管理

5.3. 中断源

表 5-1 NVIC 中断源表

NVIC 位置	优先级	优先级类型	名称	说明	地址
-	-	-	-	保留	0x0000
-	-3	固定	Reset	复位	0x0004
-	-2	固定	NMI	不可屏蔽中断	0x0008
-	-1	固定	HardFault	所有类型的错误	0x000C
-	0	可设置	MemManage	存储器管理	0x0010
-	1	可设置	BusFault	预取故障，存储器访问故障	0x0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0018
-	-	-	-	保留	0x001C-0x002B
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x002C
-	4	可设置	Debug Monitor	调试监控器	0x0030
-	-	-	-	保留	0x0034
-	5	可设置	PendSV	可挂起的系统服务	0x0038
-	6	可设置	SysTick	系统节拍定时器	0x003C
0	7	可设置	WDT	看门狗中断	0x0040
1	8	可设置	LVD	连到 EXTI 线 16 的电源电压检测（LVD）中断	0x0044

2	9	可设置	RTC	RTC 全局中断	0x0048
3	10	-	-		0x004C
4	11	可设置	EFC_INT	Flash 全局中断	0x0050
5	12	可设置	SRAM_PARITY	SRAM 读校验出错中断	0x0054
6	13	可设置	CLKRDY	时钟 Ready 中断，中断使能见系统寄存器 CIR	0x0058
7	14	可设置	EXTI0	EXTI 线 0 中断	0x005C
8	15	可设置	EXTI1	EXTI 线 1 中断	0x0060
9	16	可设置	EXTI2	EXTI 线 2 中断	0x0064
10	17	可设置	EXTI3	EXTI 线 3 中断	0x0068
11	18	可设置	EXTI4	EXTI 线 4 中断	0x006C
12	19	可设置	DMA1	DMA1 全局中断	0x0070
13	20	可设置	DMA2	DMA2 全局中断	0x0074
14	21	可设置	ADC1_2	ADC1 和 ADC2 全局中断	0x0078
15	22	-	-		0x007C
16	23	可设置	DAC	DAC 全局中断	0x0080
17	24	可设置	COMP1_2	连到 EXTI 线 20/21 的 COMP1/COMP2 中断	0x0084
18	25	可设置	USB	USB 全局中断	0x0088
19	26	可设置	CAN1	CAN1 全局中断	0x008C
20	27	可设置	CAN2	CAN2 全局中断	0x0090
21	28	可设置	EXTI9_5	EXTI 线 9~5 中断	0x0094
22	29	可设置	TIM1_BRK_UP_TRG_COM	TIM1 刹车/更新/触发和通信中断	0x0098
23	30	可设置	TIM1_CC	TIM1 捕获/比较中断	0x009C
24	31	可设置	TIM2	TIM2 全局中断	0x00A0
25	32	可设置	TIM3	TIM3 全局中断	0x00A4
26	33	可设置	TIM6	TIM6 全局中断	0x00A8
27	34	可设置	TIM7	TIM7 全局中断	0x00AC
28	35	可设置	TIM8_BRK_UP_TRG_COM	TIM8 刹车/更新/触发和通信中断	0x00B0
29	36	可设置	TIM8_CC	TIM8 捕获/比较中断	0x00B4
30	37	可设置	TIM15	TIM15 全局中断	0x00B8
31	38	可设置	TIM16	TIM16 全局中断	0x00BC
32	39	可设置	TIM17	TIM17 全局中断	0x00C0
33	40	可设置	I2C1	I2C1 全局中断	0x00C4
34	41	可设置	I2C2	I2C2 全局中断	0x00C8
35	42	可设置	SPI1	SPI1 全局中断	0x00CC
36	43	可设置	SPI2	SPI2 全局中断	0x00D0

37	44	可设置	SPI3	SPI3 全局中断	0x00D4
38	45	可设置	I2S1	I2S1 全局中断	0x00D8
39	46	可设置	I2S2	I2S2 全局中断	0x00DC
40	47	可设置	UART1	UART1 全局中断	0x00E0
41	48	可设置	UART2	UART2 全局中断	0x00E4
42	49	可设置	UART3	UART3 全局中断	0x00E8
43	50	可设置	UART4	UART4 全局中断	0x00EC
44	51	可设置	EXTI15_10	EXTI 线 15~10 中断	0x00F0
45	52	可设置	USB_WAKEUP	连到 EXTI 线 23 的 USB 唤醒中断	0x00F4
46	53	可设置	LPUART1	LPUART1 全局中断	0x00F8
47	54	可设置	LPTIM1	LPTIM1 全局中断	0x00FC
48	55	-	-		0x0100
49	56	可设置	AES	AES 全局中断	0x0104
50	57	可设置	FPU	FPU 全局中断	0x0108
51	58	-	-		0x010C
52	59	-	-		0x0110
53	60	可设置	TIM4	TIM4 全局中断	0x0114
54	61	可设置	COMP3_4	连到 EXTI 线 25/26 的 COMP3/COMP4 中断	0x0118
55	62	可设置	IWDT_WAKEUP	连到 EXTI 线 19 的 IWDT 唤醒中断	0x011C

6. 外部中断/事件控制器（EXTI）

6.1. 概述

EXTI 包含 26 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，下降沿触发和双边沿触发，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。挂起寄存器用于保持中断请求的状态线。

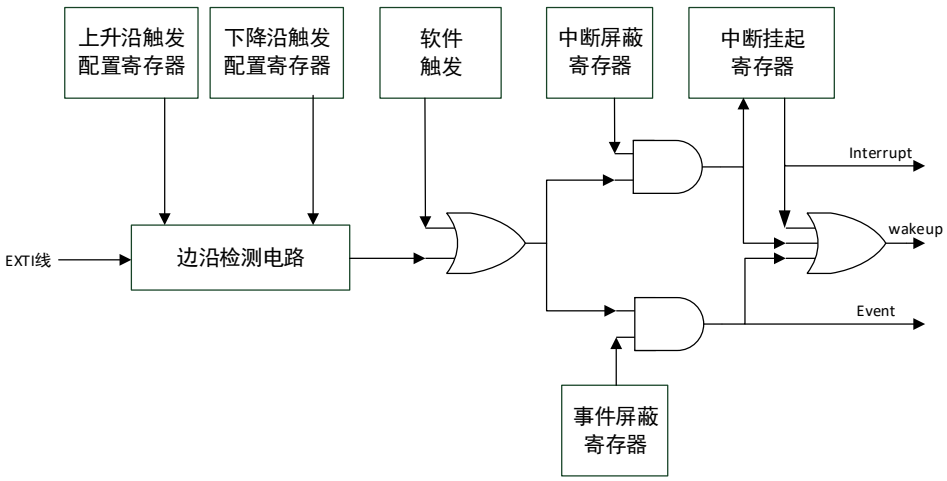
6.2. 主要特性

- 支持 26 个软件中断/事件请求
- 每个中断/事件都有独立的触发和屏蔽
- 每个中断线都有独立的状态位
- 支持脉冲或挂起输入类型
- 支持上升沿、下降沿或双边沿
- 可将系统从省电模式唤醒

6.3. 功能说明

6.3.1. 结构框图

图 6-1 外部中断及事件框图



6.3.2. 中断和事件触发源

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 10 根线，包括 LVD、 RTC、 LPUART1、IWDT（唤醒）、COMP1、COMP2、LPTIM1 和 USB_Wakeup、COMP3、COMP4。 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 源

EXTI 序号	对应源
0	PA0/PB0/PC0/PD0/PE0/PF0
1	PA1/PB1/PC1/PD1/PE1/PF1
2	PA2/PB2/PC2/PD2/PE2/PF2
3	PA3/PB3/PC3/PD3/PE3/PF3
4	PA4/PB4/PC4/PD4/PE4/PF4
5	PA5/PB5/PC5/PD5/PE5
6	PA6/PB6/PC6/PD6/PE6
7	PA7/PB7/PC7/PD7/PE7
8	PA8/PB8/PC8/PD8/PE8
9	PA9/PB9/PC9/PD9/PE9
10	PA10/PB10/PC10/PD10/PE10
11	PA11/PB11/PC11/PD11/PE11
12	PA12/PB12/PC12/PD12/PE12
13	PA13/PB13/PC13/PD13/PE13
14	PA14/PB14/PC14/PD14/PE14
15	PA15/PB15/PC15/PD15/PE15
16	LVD
17	RTC(中断)
18	LPUART1_Wakeup（中断）
19	IWDT
20	COMP1 输出信号
21	COMP2 输出信号
22	Reserved
23	USB_Wakeup（USB resume 信号）
24	LPTIM1_Wakeup（中断）
25	COMP3 输出信号
26	COMP4 输出信号
27~31	Reserved

6.3.3. 唤醒事件管理

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 ARM 内核系统控制寄存器（SCB_SCR）中使能 SEVONPEND（bit4）。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，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。

6.3.4. EXTI 功能描述

要产生中断，必须配置 NVIC 控制寄存器中相应的中断线使能位。通过沿触发使能寄存器 EXTI_RTRNR 和 EXTI_FTENR 选择边沿触发事件类型，并将中断使能寄存器 EXTI_IENR 的相应位写 1 使能中断请求，对应的挂起位也会被置 1（寄存器 EXTI_PDR）。在挂起寄存器（EXTI_PDR）的对应位写 1，将清除该中断请求。

要产生事件，必须先配置并使能对应的事件线。通过沿触发使能寄存器 EXTI_RTRNR 和 EXTI_FTENR 选择边沿触发事件类型，并将事件使能寄存器 EXTI_EENR 的相应位写 1 使能事件请求。当事件线上发生预设的边沿时，将产生一个事件脉冲，对应的挂起位不会置 1。

另外，通过在软件中对软件中断/事件寄存器（EXTI_SWIER）写 1，也可以产生中断/事件请求。

- 硬件中断配置
  - 要配置 26 根线作为中断源，请执行以下步骤：

1）配置 26 根中断线的中断使能位（EXTI_IENR)

2）配置中断线的触发配置位（EXTI_RTRNR 和 EXTI_FTENR）

3）配置对应到外部中断控制器（EXTI）的 NVIC 中断通道的使能和屏蔽位，使得 26 个中断线中的请求可以被正确地响应。

- 硬件事件配置

要配置 26 根线作为事件源，请执行以下步骤：

1）配置 26 根事件线的事件使能位（EXTI_EENR)

2）配置事件线的触发配置位（EXTI_RTRNR 和 EXTI_FTENR）

- 软件中断/ 事件配置

可将这 26 根线配置为软件中断/事件线。以下为产生软件中断的步骤。

1）配置 26 根中断/事件线的使能位（EXTI_IENR、EXTI_EENR）

2）在软件中断寄存器设置相应的请求位（EXTI_SWIER)

6.4. EXTI 寄存器描述

6.4.1. 寄存器列表

EXTI 寄存器基地址：0x40010400

偏移	名称	描述

0x0000	EXTI_IENR	中断使能寄存器
0x0004	EXTI_EENR	事件使能寄存器
0x0008	EXTI_RTENR	上升沿触发使能寄存器
0x000C	EXTI_FTENR	下降沿触发使能寄存器
0x0010	EXTI_SWIER	软件中断事件寄存器
0x0014	EXTI_PDR	挂起寄存器
0x0018	EXTI_CR1	外部 I/O 选择寄存器 1
0x001C	EXTI_CR2	外部 I/O 选择寄存器 2

6.4.2. 中断使能寄存器(EXTI_IENR: 00h)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	INTENX	RW	0x1060000	EXTIx 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。

6.4.3. 事件使能寄存器(EXTI_EENR: 04h)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	EVENX	RW	0x0000000	EXTIx 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。

6.4.4. 上升沿触发使能寄存器(EXTI_RTENR: 08h)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	RTENX	RW	0x1060000	EXTIx 上的上升沿触发使能 0：禁止来自线 x 上的上升沿触发（中断和事件） 1：使能来自线 x 上的上升沿触发（中断和事件） 注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号

6.4.5. 下降沿触发使能寄存器(EXTI_FTENR: 0Ch)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	FTENX	RW	0x000000	EXTIx 上的下降沿触发使能 0：禁止来自线 x 上的下降沿触发（中断和事件） 1：使能来自线 x 上的下降沿触发（中断和事件） 注：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

6.4.6. 软件中断事件寄存器(EXTI_SWIER: 10h)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	SWIEX	WO	0x0000000	中断/事件软件触发： 当该位为 0 时，写 1 将设置 PDR 中相应的挂起位。如果在 EXTI_IENR 和 EXTI_EENR 中允许产生该中断/事件，则此时将触发一个软件中断/事件。

6.4.7. 挂起寄存器(EXTI_PDR: 14h)

位域	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:0	PDRX	RCW1	0x0000000	EXTIx 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。

6.4.8. 外部中断配置寄存器(EXTI_CR1: 18h)

位域	名称	属性	复位值	描述
31:28	EXTI_7	RW	0000	EXTI_7 源选择。 0000: PA7 0001: PB7 0010: PC7 0011: PD7 0100: PE7

27:24	EXTI_6	RW	0000	EXTI_6 源选择。 0000: PA6 0001: PB6 0010: PC6 0011: PD6 0100: PE6
23:20	EXTI_5	RW	0000	EXTI_5 源选择。 0000: PA5 0001: PB5 0010: PC5 0011: PD5 0100: PE5
19:16	EXTI_4	RW	0000	EXTI_4 源选择。 0000: PA4 0001: PB4 0010: PC4 0011: PD4 0100: PE4 0101: PF4
15:12	EXTI_3	RW	0000	EXTI_3 源选择。 0000: PA3 0001: PB3 0010: PC3 0011: PD3 0100: PE3 0101: PF3
11:8	EXTI_2	RW	0000	EXTI_2 源选择。 0000: PA2 0001: PB2 0010: PC2 0011: PD2 0100: PE2 0101: PF2
7:4	EXTI_1	RW	0000	EXTI_1 源选择。 0000: PA1 0001: PB1 0010: PC1 0011: PD1 0100: PE1 0101: PF1

3:0	EXTI_0	RW	0000	EXTI_0 源选择。 0000: PA0 0001: PB0 0010: PC0 0011: PD0 0100: PE0 0101: PF0

6.4.9. 外部中断配置寄存器 2(EXTI_CR2: 1Ch)

位域	名称	属性	复位值	描述
31:28	EXTI_15	RW	0000	EXTI_15 源选择。 0000: PA15 0001: PB15 0010: PC15 0011: PD15 0100: PE15
27:24	EXTI_14	RW	0000	EXTI_14 源选择。 0000: PA14 0001: PB14 0010: PC14 0011: PD14 0100: PE14
23:20	EXTI_13	RW	0000	EXTI_13 源选择。 0000: PA13 0001: PB13 0010: PC13 0011: PD13 0100: PE13
19:16	EXTI_12	RW	0000	EXTI_12 源选择。 0000: PA12 0001: PB12 0010: PC12 0011: PD12 0100: PE12
15:12	EXTI_11	RW	0000	EXTI_11 源选择。 0000: PA11 0001: PB11 0010: PC11 0011: PD11 0100: PE11

11:8	EXTI_10	RW	0000	EXTI_10 源选择。 0000: PA10 0001: PB10 0010: PC10 0011: PD10 0100: PE10
7:4	EXTI_9	RW	0000	EXTI_9 源选择。 0000: PA9 0001: PB9 0010: PC9 0011: PD9 0100: PE9
3:0	EXTI_8	RW	0000	EXTI_8 源选择。 0000: PA8 0001: PB8 0010: PC8 0011: PD8 0100: PE8

7. 外设互连

7.1. 外设互连简介

多个外设间有直接连接。

这可以在外设间实现自动通信和同步，从而节省 CPU 资源和降低功耗。

此外，这些硬件连接可消除软件延时、设计可预测的系统并能够减少引脚和 GPIO 的数量。

7.2. 外设互连详细描述

7.2.1. 从定时器到定时器

表 7-1 Timer 输入触发源

Timer 输入触发源	Timer 输入触发源分配					
	TIM1	TIM2	TIM3	TIM4	TIM8	TIM15
ITR0	—	tim1_trgo	tim1_trgo	tim1_trgo	tim1_trgo	tim1_trgo
ITR1	tim2_trgo	—	tim2_trgo	tim2_trgo	tim2_trgo	tim2_trgo
ITR2	tim3_trgo	tim3_trgo	—	tim3_trgo	tim3_trgo	tim3_trgo
ITR3	tim4_trgo	tim4_trgo	tim4_trgo	—	tim4_trgo	tim4_trgo
ITR4	—	—	—	—	—	—
ITR5	tim8_trgo	tim8_trgo	tim8_trgo	tim8_trgo	—	tim8_trgo
ITR6	tim15_trgo	tim15_trgo	tim15_trgo	tim15_trgo	tim15_trgo	—
ITR7	tim16_oc1	tim16_oc1	tim16_oc1	tim16_oc1	tim16_oc1	tim16_oc1
ITR8	tim17_oc1	tim17_oc1	tim17_oc1	tim17_oc1	tim17_oc1	tim17_oc1

注：符号“—”表示无互连

一些 TIMx 定时器在内部连接在一起，用于计时器同步或连接。

当一个定时器配置为主模式时，可以重置、启动、停止或计时另一个配置为从模式的定时器。

主定时器通过 TRGO 或 OCx 信号（比较输出信号）输出，从定时器通过 ITRx 接收信号。如上表所示，TIM1 作为从定时器，其 ITR1 触发输入信号为 TIM2 的 TRGO 信号。

7.2.2. 从定时器和 EXTI 到 ADC

表 7-2 ADC 触发信号源

ADC 触发选择 (TSEL[3:0])	ADC 触发信号源分配	
	ADC1/2	
	规则通道(TRIG)	注入通道(JTRIG)
0	tim1_cc1	tim1_trgo
1	tim1_cc2	tim1_cc4

ADC 触发选择(TSEL[3:0])	ADC 触发信号源分配	
	ADC1/2	
	规则通道(TRIG)	注入通道(JTRIG)
2	tim1_cc3	tim2_trgo
3	tim2_cc2	tim2_cc1
4	tim3_trgo	tim3_cc4
5	tim4_cc4	tim4_trgo
6	exti11	exti15
7	tim8_trgo	tim8_cc4
8	tim8_trgo2	tim1_trgo2
9	tim1_trgo	tim8_trgo
10	tim1_trgo2	tim8_trgo2
11	tim2_trgo	tim3_cc3
12	tim4_trgo	tim3_trgo
13	tim6_trgo	tim3_cc1
14	tim15_trgo	tim6_trgo
15	tim3_cc4	tim15_trgo
16	tim1_cc4	tim16_cc1
17	lptim_out	lptim_out
18	tim7_trgo	tim7_trgo
19	tim8_cc4	tim17_cc1

ADC 的外部触发的详细情况，请参考 ADC 章节。

外部触发信号包括从 EXTI 模块输入的 IO 信号，TIMx 的 TRGO 或捕获比较事件，低功耗定时器的输出信号。

7.2.3. 从 ADC 到定时器

见从比较器到定时器章节。

7.2.4. 从定时器和 EXTI 到 DAC

表 7-3 DAC 触发信号

DAC 触发选择（TSEL，STRINCTRIGSEL）	DAC 触发信号分配	
	DAC	
	转换/复位触发选择	递增触发选择
0	SW	SW
1	TIM8_TRGO	TIM8_TRGO

DAC 触发选择（TSEL，STRINCTRIGSEL）	DAC 触发信号分配	
	DAC	
	转换/复位触发选择	递增触发选择
2	TIM7_TRGO	TIM7_TRGO
3	TIM15_TRGO	TIM15_TRGO
4	TIM2_TRGO	TIM2_TRGO
5	TIM4_TRGO	TIM4_TRGO
6	EXTI Line 9	EXTI Line 10
7	TIM6_TRGO	TIM6_TRGO
8	TIM3_TRGO	TIM3_TRGO
9	—	—

DAC 的外部触发的详细情况，请参考 DAC 章节。

外部触发信号包括从 EXTI 模块输入的 IO 信号，TIMx 的 TRGO 信号。

7.2.5. 从 RTC，比较器到低功耗定时器

表 7-4 LPTIM1 触发源

LPTIM 触发输入信号(TRIGSEL)	LPTIM1 触发源分配
lptim1_ext_trig0	LPTIM1_ETR（IO 引脚）
lptim1_ext_trig1	RTC_ALARM
lptim1_ext_trig2	RTC_TAMP1
lptim1_ext_trig3	RTC_TAMP2
lptim1_ext_trig4	comp1_out
lptim1_ext_trig5	comp2_out
lptim1_ext_trig6	comp3_out
lptim1_ext_trig7	comp4_out

LPTIM 的外部触发的详细情况，请参考 LPTIM 章节。

外部触发信号包括 IO 引脚输入，定时器报警信号，RTC 检测到的侵入信号（RTC 模块需使能侵入检测并配置了极性选择），比较器输出信号。

7.2.6. 从定时器到比较器

表 7-5 比较器消隐源

比较器消隐源信号选择（BLANKSEL[2:0]）	比较器消隐源分配			
	COMP1	COMP2	COMP3	COMP4
1	tim1_oc5	tim1_oc5	tim1_oc5	tim3_oc4
2	tim2_oc3	tim2_oc3	tim3_oc3	tim8_oc5
3	tim3_oc3	tim3_oc3	tim2_oc4	tim15_oc1
4	tim8_oc5	tim8_oc5	tim8_oc5	tim1_oc5
5	tim1_oc4	tim1_oc4	tim8_oc4	tim8_oc4
6	tim15_oc1	tim15_oc1	tim15_oc1	tim15_oc1
7	tim4_oc3	tim4_oc3	tim4_oc3	tim4_oc3

消隐（blanking）功能请见比较器章节。

Blanking 信号的来源为定时器的比较输出信号。

7.2.7. 内部模拟信号源到 ADC，比较器，运算放大器

表 7-6 ADC 通道源

ADC 通道号	ADC 通道源分配	
	ADC1	ADC2
IN0	PA0	PA0
IN1	PA1	PA1
IN2	PA2/OPA1_OUT（1）	PA2/OPA1_OUT（1）
IN3	PA3	PA3
IN4	PA4/DAC_OUT1（1）	PA4/DAC_OUT1（1）
IN5	PA5/DAC_OUT2（1）	PA5/DAC_OUT2（1）
IN6	PA6/OPA2_OUT（1）	PA6/OPA2_OUT（1）
IN7	PA7	PA7
IN8	PB0	PB0
IN9	PB1/OPA3_OUT（1）	PB1/OPA3_OUT（1）
IN10	PC0	PC0
IN11	PC1	PC1
IN12	PC2	PC2
IN13	PC3	PC3
IN14	PC4	PC4
IN15	PC5	PC5

ADC 通道号	ADC 通道源分配	
	ADC1	ADC2
IN16	TempSensor/VBAT(PB12)	pga2_out_int（2）
IN17	VBG1P2	pga3_out_int（2）
IN18	pga2_out_int（2）	PE7
IN19	pga1_out_int（2）	PE8

ADC 的通道输入选择功能详情请参考 ADC 章节。

ADC 通道映射到 GPIO 或内部连接到温度传感器，VBAT 或 OPAx_out_int，OPAx_OUT 等。

此信号代表 OPAx_OUT 信号可以输出到 GPIO 引脚，而 GPIO 引脚可以作为 ADC 的输入信号源。

该信号代表芯片内部信号，直接连接到 ADC 的输入。

表 7-7 比较器输入/输出信号

比较器输入/输出信号		比较器输入/输出信号分配				
		GPIO			DAC internal	VDDA or VREF1.2
COMP1	INP	PA1	PB1	PB10	COMP2_INPO	—
	INM	PA4	PA0	—	DAC1_CH1(PA4)	1/20VDDA~16/20VDDA
	OUT	PA0/PA6	PA11/PB0	PB8/PB10	—	—
COMP2	INP	PA7	PA3	PB11	COMP1_INPO	—
	INM	PA5	PA2	—	DAC1_CH2 (PA5)	1/20VDDA~16/20VDDA
	OUT	PA2/PA7	PA12/PB5	PB9/PB11	—	—
COMP3	INP	PA0	PC1	PC3	COMP4_INPO	—
	INM	PF1	PC0	—	DAC1_CH1(PA4)	1/20VDDA~16/20VDDA
	OUT	PC2	PB7	PB15	—	—
COMP4	INP	PB0	PE7	PE9	COMP3_INPO	—
	INM	PE8	PB2	—	DAC1_CH1(PA4)	1/20VDDA~16/20VDDA
	OUT	PB1	PB6	PB14	—	—

表 7-8 OPA 输入/输出信号

OPA 输入/输出信号		OPA 输入/输出信号分配					
		GPIO			DAC	ADC GPIO	ADC internal
OPAMP1	VINP	PA1	PA3	PA7	DAC1_CH1	—	—
	VINM	PA3	PC5	—	—	—	—
	VOUT	PA2	—	—	—	ADC_IN2	ADC1_IN19
OPAMP2	VINP	PA7	PB0	OPAMP1 输出	DAC1_CH2	—	—

OPA 输入/输出信号		OPA 输入/输出信号分配					
		GPIO			DAC	ADC GPIO	ADC internal
	VINM	PA5	PC5	—	—	—	—
	VOUT	PA6	—	—	—	ADC_IN6	ADC1_IN18 ADC2_IN16
OPAMP3	VINP	PB0	PA1	PB11	DAC1_CH1	—	—
	VINM	PB2	PB10	—	—	—	—
	VOUT	PB1	—	—	—	ADC_IN9	ADC1_IN17

7.2.8. 从比较器到定时器

比较器 COMPx 输出可以连接到定时器 TIMx 输入捕获或 TIMx_ETR 信号或 TIMx_OCREFCLR 信号。

比较器 COMPx 输出也可以产生定时器 TIMx 的刹车输入信号。

表 7-9 Timer 外部触发信号源

Timer 外部触发信号	Timer 外部触发信号源分配				
	TIM1	TIM2	TIM3	TIM4	TIM8
ETR0	tim1_etr(IO)	tim2_etr(IO)	tim3_etr(IO)	tim4_etr(IO)	tim8_etr(IO)
ETR1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out
ETR2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out
ETR3	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out
ETR4	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out
ETR5	adc1_awd	tim3_etr(IO)	tim2_etr(IO)	tim3_etr(IO)	adc2_awd(1)
ETR6	adc2_awd	tim4_etr(IO)	tim4_etr(IO)	tim5_etr(IO)	—
ETR7	—	tim5_etr(IO)	adc2_awd(1)	—	—

ADC 的模拟看门狗输出。

表 7-10 Timer OCREF 清除信号源

Timer OCREF 清除信号	Timer OCREF 清除信号源分配							
	TIM1	TIM2	TIM3	TIM4	TIM8	TIM15	TIM16	TIM17
ocref_clr0	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out
ocref_clr1	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out
ocref_clr2	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out
ocref_clr3	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out

表 7-11 Timer TI1 信号源

Timer TI1 输入信号	Timer TI1 信号源分配							
	TIM1	TIM2	TIM3	TIM4	TIM8	TIM15	TIM16	TIM17
ti1_in0	tim1_ch1	tim2_ch1	tim3_ch1	tim4_ch1	tim8_ch1	tim15_ch1	tim16_ch1	tim17_ch1
ti1_in1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	mco	mco
ti1_in2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	xth_div32(1)	usb_sof (2)
ti1_in3	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out	mco	rtc_wakeup	rtc_wakeup
ti1_in4	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out	uart1_rx	rcl	rcl

外部晶振频率的 1/32。

USB SOF 脉冲信号

表 7-12 Timer TI2 信号源

Timer TI2 输入信号	Timer TI2 信号源分配					
	TIM1	TIM2	TIM3	TIM4	TIM8	TIM15
ti2_in0	tim1_ch2	tim2_ch2	tim3_ch2	tim4_ch2	tim8_ch2	tim15_ch2
ti2_in1	—	comp1_out	comp1_out	comp1_out	—	comp2_out
ti2_in2	—	comp2_out	comp2_out	comp2_out	—	comp3_out
ti2_in3	—	comp3_out	comp3_out	comp3_out	—	mco
ti2_in4	—	comp4_out	comp4_out	comp4_out	—	uart2_rx

表 7-13 Timer TI3 信号源

Timer TI3 输入信号	Timer TI3 信号源分配				
	TIM1	TIM2	TIM3	TIM4	TIM8
ti3_in0	tim1_ch3	tim2_ch3	tim3_ch3	tim4_ch3	tim8_ch3
ti3_in1	—	comp4_out	comp3_out	—	—

表 7-14 Timer TI4 信号源

Timer TI4 输入信号	Timer TI4 信号源分配				
	TIM1	TIM2	TIM3	TIM4	TIM8
ti4_in0	tim1_ch4	tim2_ch4	tim3_ch4	tim4_ch4	tim8_ch4
ti4_in1	—	comp1_out	—	—	—
ti4_in2	—	comp2_out	—	—	—

TIMER 刹车通道来自输入引脚、比较器输出和系统级故障（见“从系统错误到定时器”章节）。

表 7-15 Timer 刹车信号源

Timer 刹车信号源	Timer 刹车信号源分配				
	TIM1	TIM8	TIM15	TIM16	TIM17
GPIO	tim1_bkin	tim8_bkin	tim15_bkin	tim16_bkin	tim17_bkin
COMP1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out
COMP2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out
COMP3	comp3_out	comp3_out	comp3_out	comp3_out	comp3_out
COMP4	comp4_out	comp4_out	comp4_out	comp4_out	comp4_out

7.2.9. 从系统错误到定时器

系统错误信号源来自：

- LOCKUP（Hardfault）输出
- SRAM 奇偶校验错误
- LVD 输出
- Flash 奇偶校验错误

刹车功能的目的是保护定时器产生的 PWM 信号驱动的功率开关。

表 7-16 系统错误信号到 Timer 信号

	系统错误信号到 Timer 信号分配				
	TIM1	TIM8	TIM15	TIM16	TIM17
系统错误信号	CPU LOCKUP	CPU LOCKUP	CPU LOCKUP	CPU LOCKUP	CPU LOCKUP
	SRAM Parity	SRAM Parity	SRAM Parity	SRAM Parity	SRAM Parity
	LVD	LVD	LVD	LVD	LVD
	Flash Parity	Flash Parity	Flash Parity	Flash Parity	Flash Parity

7.2.10. 从定时器到红外接口（IRTIM）

表 7-17 IRTIM 控制信号

IRTIM 控制信号	IRTIM 控制信号分配
调制包络信号	tim16_ch1
高频载波信号	tim17_ch1

TIM16/17 输出通道 TIMx_CH1 用于产生红外信号的波形输出。

8. 通用输入输出接口（GPIO）

8.1. 概述

芯片所有 GPIO 按组分类，每组 GPIO 最多包含 16 个通用数据输入输出接口。每个 GPIO 引脚可以独立配置为输入、输出（推挽或开漏）、复用的外设功能或模拟功能。

8.2. 主要特性

- 输入/输出方向可通过软件进行配置；
- 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
- GPIO 引脚可作为 EXTI 输入，且触发边沿可配置
- 支持复用功能输入/输出配置
- 支持锁定机制，可冻结 I/O 端口配置

8.3. 功能描述

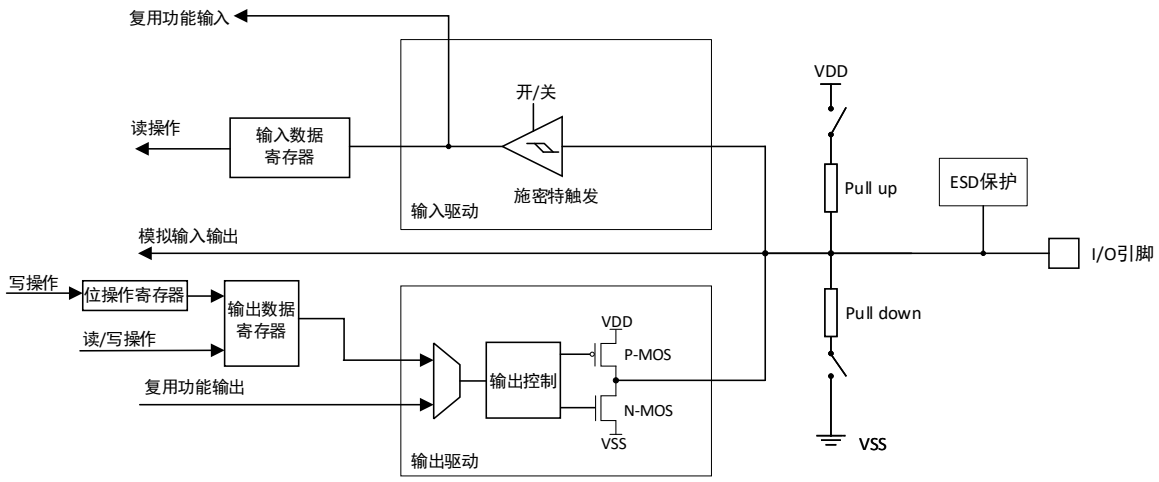
根据数据手册中列出的每个 I/O 端口的特性，可通过软件将 GPIO 端口的各个端口位分别配置为多种模式：

- 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。

图 8-1 GPIO 管脚结构框图



8.3.1. GPIO 初始状态

在复位期间或复位之后，绝大多数 GPIO 的复用功能并未激活，都被配置成模拟浮空模式。

但是芯片复位后：

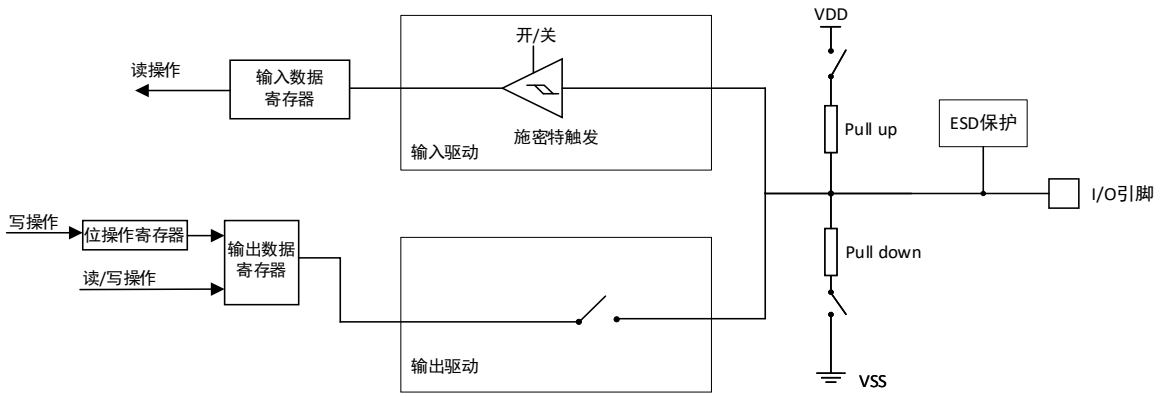
- 调试引脚处于复用功能上拉/下拉状态：
  - PA15：JTDI 为上拉模式；
  - PA14：JTCK / SWCLK 为下拉模式；
  - PA13：JTMS / SWDIO 为上拉模式；
  - PB4：NJTRST 为上拉模式；
  - PB3：JTDO 为下拉模式
- 系统引脚处于复用功能输出状态：
  - PA8：MCO1 输出低电平
  - PD4：RSTO 输出低电平
  - PD5：REMAP 输出低电平

8.3.2. 输入功能

当 GPIO 配置为输入功能时（寄存器 GPIOx_MD 配置输入模式）：

- 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）
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可获取 I/O 状态（寄存器 GPIOx_IDATA）

图 8-2 输入浮空/上拉/下拉配置

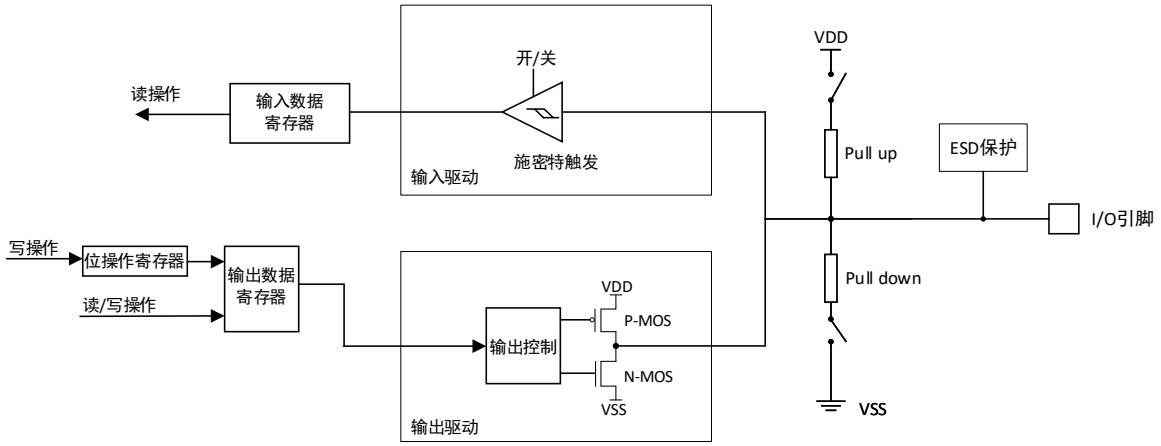


8.3.3. 输出功能

当 GPIO 配置为输出功能时（寄存器 GPIOx_MD 配置输出模式）：

- 输出缓冲器被打开，可选择推挽输出或开漏输出（寄存器 GPIOx_OTYP）
  - 开漏模式：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：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）
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可获取 I/O 状态（寄存器 GPIOx_IDATA）
- 对输出数据寄存器的读访问可获取最后的写入值

图 8-3 输出配置

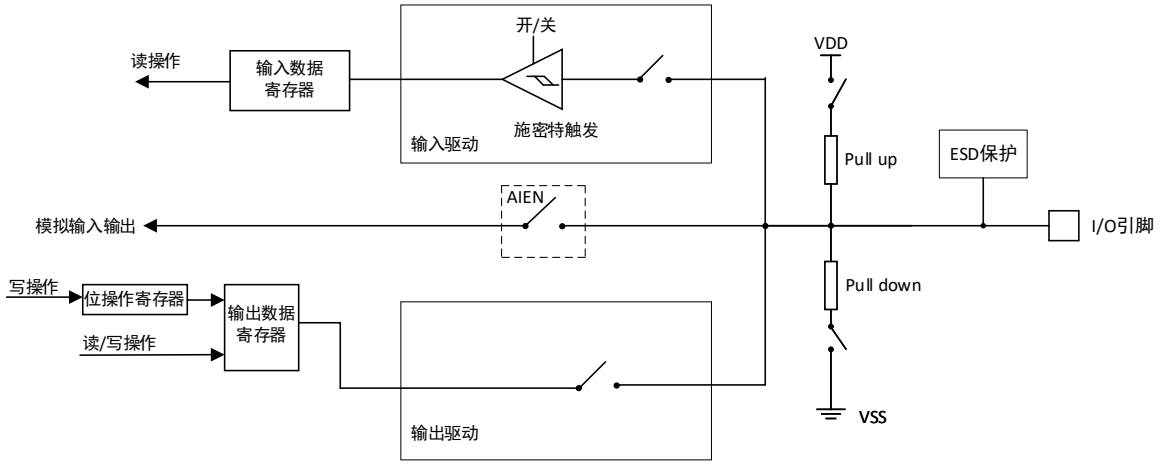


8.3.4. 模拟功能

当 GPIO 配置为模拟功能时（寄存器 GPIOx_MD 配置模拟模式）：

- 输出缓冲器被禁止
- 输入缓冲器被禁止
- 施密特触发器功能无效（无论配置是否使能）
- 上拉和下拉电阻需配置关闭（寄存器 GPIOx_PUPD）
- 用作 ADC/CMP/OPA 通道时，需配置模拟开关闭合（寄存器 GPIOx_AIEN）

图 8-4 模拟配置

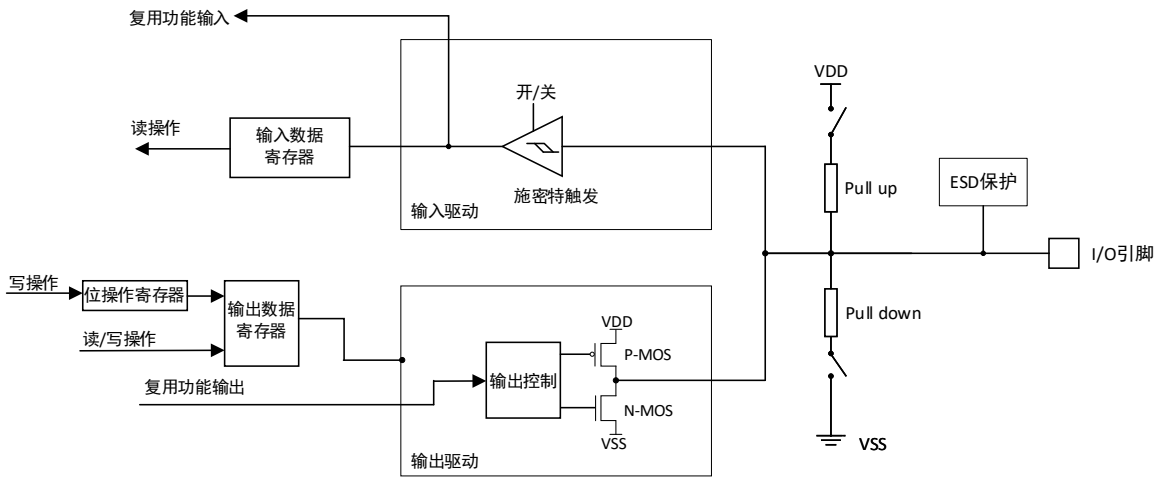


8.3.5. 复用功能

当 GPIO 配置为复用功能时（寄存器 GPIOx_MD 配置复用模式）：

- 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）
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可获取 I/O 状态（寄存器 GPIOx_IDATA）

图 8-5 复用功能配置



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

要将 I/O 配置成所需功能，用户必须按照以下步骤操作：

- 调试功能：芯片复位后，这些引脚即为复用的调试功能引脚。
- 系统功能：
  - 芯片复位后，MCO1（仅 PA8，详见芯片数据手册中的“复用功能映射”表）/RSTO/REMAP 引脚即为复用功能模式；
  - 需将 MCO2 引脚配置为复用功能模式。
- GPIO ：在 GPIOx_MD 寄存器中将所需 I/O 配置为输出、输入或模拟通道。
- 外设复用功能：
  - 在 GPIOx_AR0 或 GPIOx_AR1 寄存器中，将 I/O 连接到所需的 AFx
  - 通过 GPIOx_OTYP 和 GPIOx_PUPDR 寄存器，分别选择类型及上拉/下拉
  - 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能
- 其它功能：
  - 对于 ADC 和 DAC，在 GPIOx_MD 寄存器中将所需 I/O 配置为模拟模式，并在 ADC 和 DAC 寄存器中配置所需功能。
  - 对于 RTC_OUT、RTC_TS、RTC_TAMPx、WKUPx 和振荡器的其它功能，在相关的 RTC、PMU 和 RCC 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。有关 PMU 的 I/O 控制的详细信息，请参见 PMU_IOSEL 寄存器描述。

有关复用功能 I/O 引脚映射的详细信息，请参见芯片数据手册中的“复用功能映射”表。

8.3.6. 外部中断/事件线

所有端口都具有外部中断功能。要使用外部中断线，必须将端口配置为输入模式。请详见 EXTI 模块章节。

8.3.7. GPIO 锁定功能

通过将特定的序列应用到 GPIOx_LOCK 寄存器，可以锁定 GPIO 控制寄存器，在下一次复位之前，不能再更改端口的配置。锁定的寄存器包括 GPIOx_MD、GPIOx_OTYP、GPIOx_PUPD、GPIOx_AF0、GPIOx_AF1、GPIOx_DS0、GPIOx_DS1、GPIOx_SMIT、GPIOx_AIEN。

要对 GPIOx_LOCK 寄存器执行写操作，必须执行特定的写/读序列。当正确的 LOCK 序列应用到此寄存器的 bit16 位后，会使用 LOCKEN[15:0] 的值来锁定 I/O 的配置（在写序列期间，LOCKEN [15:0] 的值必须相同）。将 LOCK 序列应用到某个端口位后，在执行下一次 MCU 复位或外设复位之前，将无法对该端口位的值进行修改。每个 LOCKEN [15:0]位都会锁定控制寄存器（GPIOx_MD、GPIOx_OTYP、GPIOx_PUPD、GPIOx_AF0、GPIOx_AF1、GPIOx_DS0、GPIOx_DS1、GPIOx_SMIT、GPIOx_AIEN）中的对应位。

LOCK 序列只能通过对 GPIOx_LOCK 寄存器进行字访问的方式来执行，因为 GPIOx_LOCK 的第 16 位必须与 [15:0] 位同时操作。有关详细信息，请参见 GPIOx_LOCK 寄存器的说明。

8.3.8. GPIO 作为晶体振荡器引脚

当 XTH 或 XTL 晶体振荡器关闭（复位后的默认状态）时，可将相关的振荡器引脚用作常规 GPIO。

当 XTH 或 XTL 晶体振荡器打开（通过设置 RCC_XTHCR 寄存器中的 XTHEN 或 RCC_STDBY_CTRL 寄存器中的 XTLEN 位）时，振荡器会控制与其相关联的引脚，这些引脚的 GPIO 配置不起作用。

将振荡器配置为用户外部时钟模式时，仅为时钟输入保留 OSC_IN 或 OSC32_IN 引脚，OSC_OUT 或 OSC32_OUT 引脚仍可用作常规 GPIO。

8.3.9. GPIO 在待机域中的使用

当内核电源域掉电时（器件进入待机模式时），PC13/PC14/PC15 GPIO 功能会丢

失。在这种情况下，如果不通过 PMU_IOSEL 寄存器配置旁路其 GPIO 配置，这些引脚将被设置为数字输入模式。

8.4. 寄存器描述

8.4.1. 寄存器列表

GPIO 寄存器基地址：

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

偏移	名称	描述

0x00	GPIOx_MD	模式寄存器
0x04	GPIOx_OTYP	输出类型寄存器
0x08	GPIOx_PUPD	上下拉寄存器
0x0C	GPIOx_IDATA	输入引脚映射寄存器
0x10	GPIOx_ODATA	输出引脚映射寄存器
0x14	GPIOx_BSC	输出置位/清零寄存器
0x18	GPIOx_AF0	复用功能配置寄存器 0
0x1C	GPIOx_AF1	复用功能配置寄存器 1
0x20	GPIOx_DS0	驱动能力配置寄存器 0
0x24	GPIOx_DS1	驱动能力配置寄存器 1
0x28	GPIOx_SMIT	施密特使能寄存器
0x2C	GPIOx_LOCK	配置锁定寄存器
0x30	GPIOx_AIEN	模拟开关配置寄存器

8.4.2. 模式寄存器(GPIOx_MD: 00h)

位域	名称	属性	复位值	描述
31:0	MDY[1:0]	R/W	GPIOA: 0xABFEFFFF GPIOB: 0xFFFFFEBF GPIOC: 0xFFFFFFFF GPIOD: 0xFFFFFAFF GPIOE: 0xFFFFFFFF GPIOF: 0x000003BF	GPIO 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

8.4.3. 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	-	-	保留
15:0	OTYP	R/W	0x0000	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

8.4.4. GPIO 上下拉寄存器(GPIOx_PUPD: 08h)

位域	名称	属性	复位值	描述
31:0	PUPDY[1:0]	R/W	GPIOA: 0x64000000 GPIOB: 0x00000180 GPIOC~E: 0x00000000 GPIOF: 0x00000040	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

8.4.5. 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	RSV	-	-	保留
15:0	IDATA	RO	0x0000	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。

8.4.6. 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	RSV	-	-	保留
15:0	ODATA	R/W	0x0000	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。

8.4.7. 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	CLR	WO	0x0000	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	SET	WO	0x0000	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。

8.4.8. 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	AFSELY [3:0]	R/W	0x00000000	GPIOx 的 PINy(y=7 to 0)复用功能配置位 0000~1111：AF0~AF15

8.4.9. 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	AFSELY[3:0]	R/W	0x00000000	GPIOx 的 PINy(y=15 to 8)复用功能配置位 0000~1111：AF0~AF15

8.4.10. 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	DSY [3:0]	R/W	0x00000000	GPIOx 的 PINy(y=7 to 0) 驱动能力 5V 耐压 IO 的驱动能力配置： 0000：2mA 0001：4mA 0010：6mA 0011：8mA 0100：10mA 0101：12mA 0110：14mA 0111：16mA 1xxx：2mA 注：xxx 表示该位上赋任意值不影响结果 非 5V 耐压 IO 的驱动能力配置：0000：3mA 0001：6mA 0010：9mA 0011：12mA 0100：15A 0101：18mA 0110：21mA 0111：24mA 1xxx：3mA 以上为 VDD=3.3V 时驱动能力。其他供电电压的驱动能力请查阅相应芯片的 datasheet。

8.4.11. 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	DSY[3:0]	R/W	0x00000000	GPIOx 的 PINy(y=15 to 8) 驱动能力 5V 耐压 IO 的驱动能力配置： 0000：2mA 0001：4mA 0010：6mA 0011：8mA 0100：10mA 0101：12mA 0110：14mA 0111：16mA 1xxx：2mA 非 5V 耐压 IO 的驱动能力配置： 0000：3mA 0001：6mA 0010：9mA 0011：12mA 0100：15A 0101：18mA 0110：21mA 0111：24mA 1xxx：3mA 以上为 VDD=3.3V 时驱动能力。其他供电电压的驱动能力请查阅相应芯片的 datasheet。

8.4.12. 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	-	-	保留
15:0	SMTEN	R/W	0x0000	GPIOx 的 PINy(y=15 to 0)施密特输入使能位： 0：禁止 1：使能

8.4.13. 配置锁定寄存器(GPIOx_LOCK: 2Ch)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留

16	LOCKKEY	R/W	0	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 LOCKKEY）： 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 LOCKEN[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	LOCKEN	R/W	0x0000	GPIOx 的 PINy(y=15 to 0)配置锁定使能位： 0：禁止 1：使能 注：当 LOCKKEY=0 时该位可写入

8.4.14. 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	-	-	保留
15:0	AIEN	R/W	0x0000	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

9. DMA 控制器（DMA）

9.1. 概述

DMA 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。

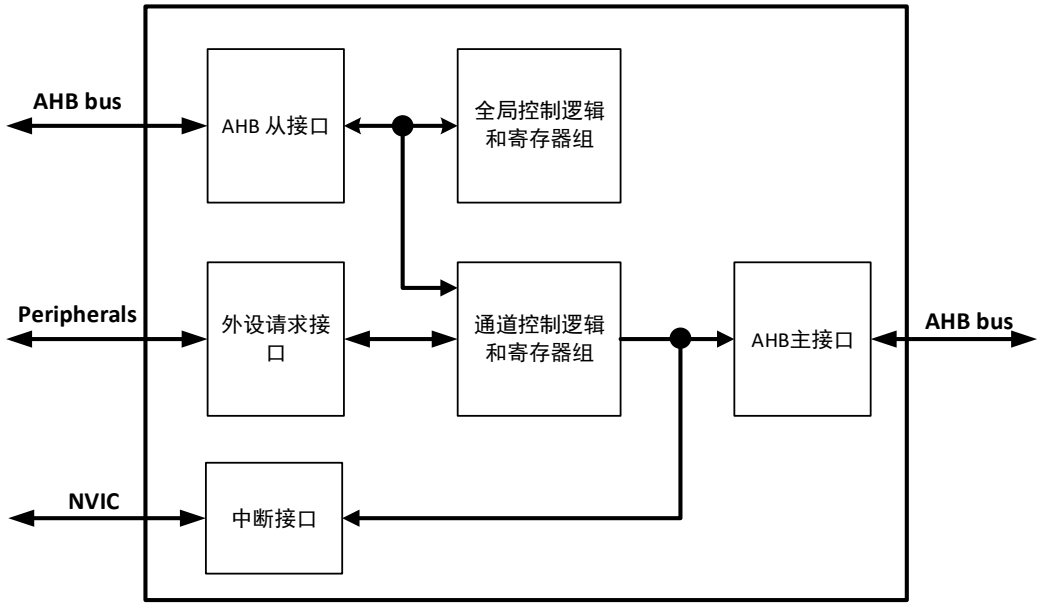
9.2. 主要特性

- 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。

9.3. 功能说明

9.3.1. 框图

图 9-1 DMA 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。
-

9.3.2. 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。

9.3.3. 外设请求

DMA1 和 DMA2 各自有 64 个外设请求，每个 DMA 内部的通道共享此 64 个外设请求。

表 9-1 目标外设和源外设请求号

请求号	DMA1 请求源	DMA2 请求源	请求号	DMA1 请求源	DMA2 请求源
REQ 0	ADC1	ADC1	REQ 32	TIM15_CH1	TIM16_CH1
REQ 1	SPI1_TX	SPI1_TX	REQ 33	TIM15_CH2	TIM16_UP

REQ 2	SPI1_RX	SPI1_RX	REQ 34	TIM15_UP	TIM17_CH1
REQ 3	SPI2_TX	SPI2_TX	REQ 35	TIM15_TRIG	TIM17_UP
REQ 4	SPI2_RX	SPI2_RX	REQ 36	TIM15_COM	TIM6_UP
REQ 5	UART1_TX	UART1_TX	REQ 37	I2S1_TX	I2S1_TX
REQ 6	UART1_RX	UART1_RX	REQ 38	I2S1_RX	I2S1_RX
REQ 7	UART2_TX	UART2_TX	REQ 39	DAC1_CH1	DAC1_CH1
REQ 8	UART2_RX	UART2_RX	REQ 40	DAC1_CH2	DAC1_CH2
REQ 9	I2C1_TX	I2C1_TX	REQ 41	I2S2_TX	I2S2_TX
REQ10	I2C1_RX	I2C1_RX	REQ 42	I2S2_RX	I2S2_RX
REQ11	I2C2_TX	I2C2_TX	REQ 43		
REQ12	I2C2_RX	I2C2_RX	REQ 44		
REQ13	TIM1_CH1	TIM8_CH1	REQ 45	UART4_TX	UART4_TX
REQ14	TIM1_CH2	TIM8_CH2	REQ 46	UART4_RX	UART4_RX
REQ15	TIM1_CH3	TIM8_CH3	REQ 47	SPI3_TX	SPI3_TX
REQ16	TIM1_CH4	TIM8_CH4	REQ 48	SPI3_RX	SPI3_RX
REQ17	TIM1_UP	TIM8_UP	REQ 49	TIM4_CH1	TIM7_UP
REQ18	TIM1_TRIG	TIM8_TRIG	REQ 50	TIM4_CH2	
REQ19	TIM1_COM	TIM8_COM	REQ 51	TIM4_CH3	
REQ20	TIM2_CH1	TIM3_CH1	REQ 52	TIM4_CH4	
REQ21	TIM2_CH2	TIM3_CH2	REQ 53	TIM4_UP	
REQ22	TIM2_CH3	TIM3_CH3	REQ 54	TIM4_TRIG	
REQ23	TIM2_CH4	TIM3_CH4	REQ 55		
REQ24	TIM2_UP	TIM3_UP	REQ 56		
REQ25	TIM2_TRIG	TIM3_TRIG	REQ 57		
REQ26	ADC2	ADC2	REQ 58		
REQ27			REQ 59		
REQ28	UART3_TX	UART3_TX	REQ 60		
REQ29	UART3_RX	UART3_RX	REQ 61		
REQ30	LPUART_TX	LPUART_TX	REQ 62		
REQ31	LPUART_RX	LPUART_RX	REQ 63		

9.3.4. 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。

9.3.5. 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。

9.3.6. 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。

9.3.7. 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。

9.3.8. 传输模式

9.3.8.1. 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。

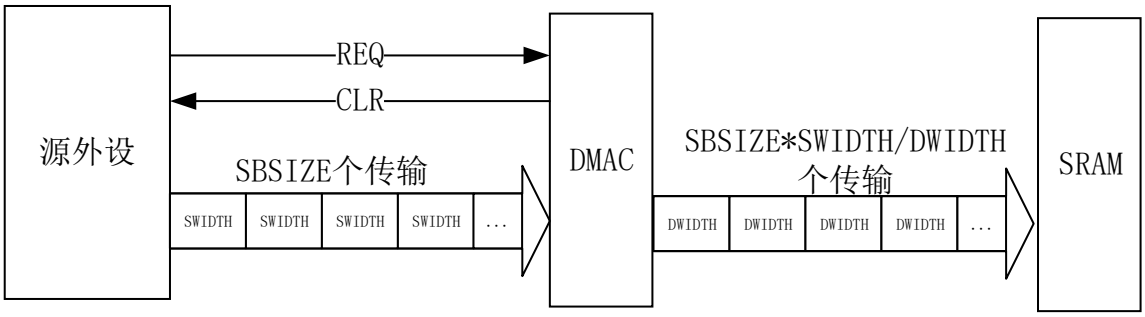
- 每次 AHB 总线传输的有效数据量将会被暂存到通道内置的 FIFO 中，当通道内置 FIFO 满时，将暂停从源地址读取数据。

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。



9.3.8.2. 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。

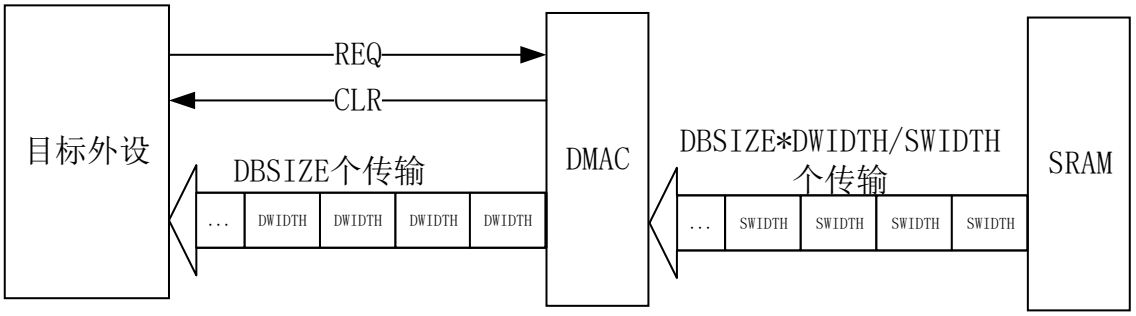
3）AHB 主接口将通道的 FIFO 中数据写入到目标地址。

- 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。



9.3.8.3. 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。

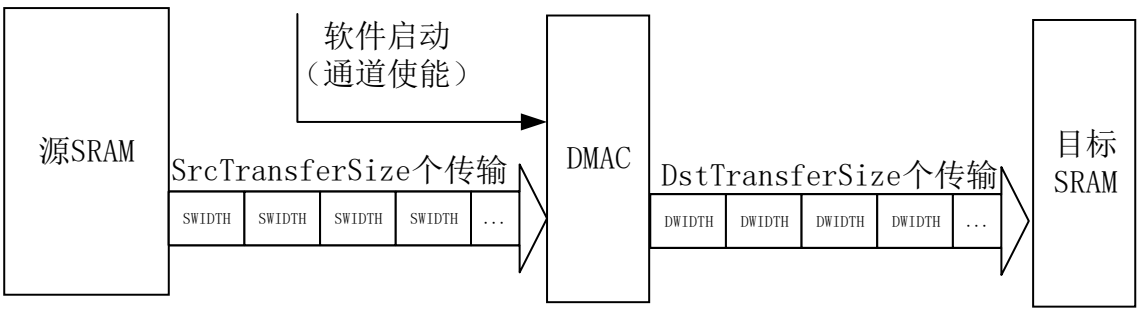
- AHB 主接口自动会发起 AHB 总线传输从源地址读取数据，每次 AHB 总线传输的有效数据量（字节/半字/字）由 DMAC_Cx_CTRL.SWIDTH 决定，AHB 总线传输读取数据的次数等于 DMAC_Cx_CTRL.SBSIZE。
- 每次 AHB 总线传输的有效数据量将会被暂存到通道内置的 FIFO 中，当通道内置 FIFO 满时，将暂停从源地址读取数据。

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。



9.3.9. 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）。

9.3.10. 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)。

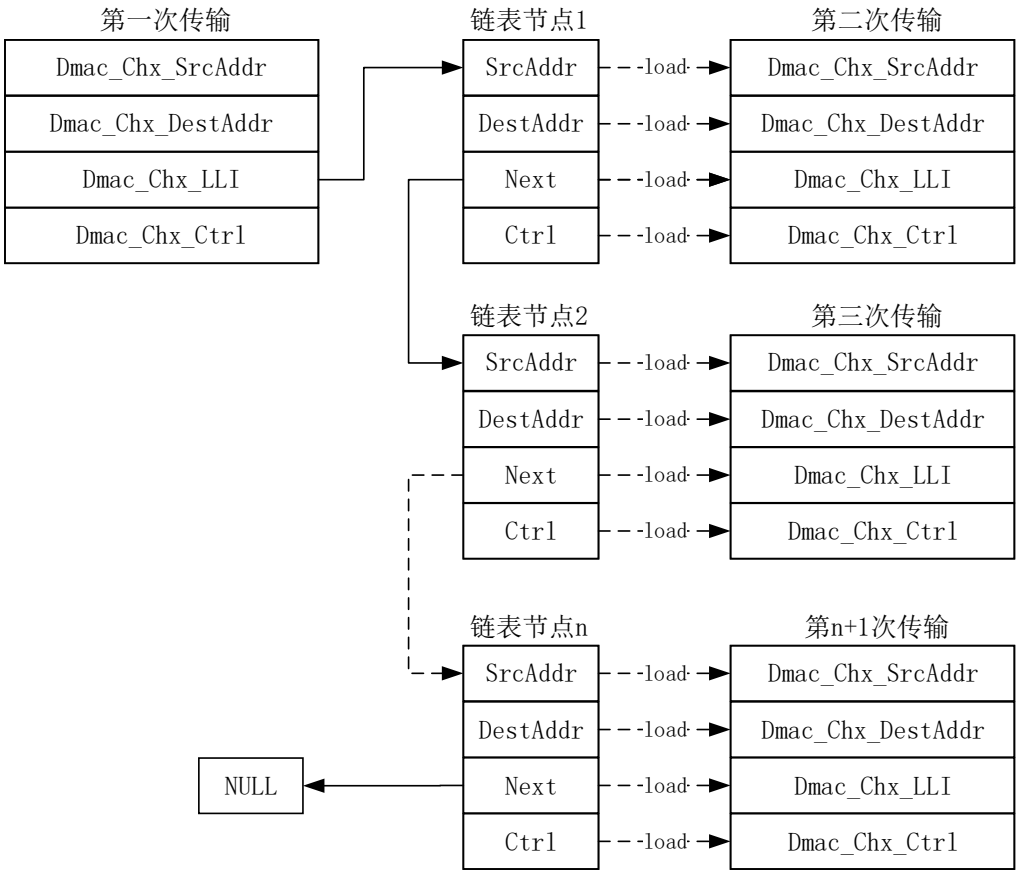
在传输完成后，如果通道 DMA_Cx_LLI.LLI 的值等于 0，则将通道关闭，硬件将 DMA_Cx_CONFIG.EN 清零。

表 9-1 链表结构体

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。

图 9-2 链表结构示意图



9.3.11. 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。

9.3.12. 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。

9.3.13. 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。

9.3.13.1. 源小端模式、目标小端模式

表 9-2 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

9.3.13.2. 源大端模式、目标大端模式

表 9-3 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

9.3.14. 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）。

9.3.15. 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。

9.3.16. 中断

9.3.16.1. 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。

9.3.16.2. 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。

9.3.16.3. 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。

9.3.16.4. 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。

9.4. 配置流程

9.4.1.1. 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。

9.4.1.2. 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）。

9.5. DMA 寄存器描述

9.5.1. 寄存器列表

DMA1 寄存器基地址：0x40031000

DMA2 寄存器基地址：0x40032000

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

0x10	DMA_INT_ERR_CLR		传输错误中断清除寄存器
0x14	DMA_RAW_INT_TC_STATUS		传输完成原始中断寄存器
0x18	DMA_RAW_INT_ERR_STATUS		传输错误原始中断寄存器
0x1C	DMA_EN_CH_STATUS		通道使能状态寄存器
0x30	DMA_CONFIG		DMA 配置寄存器
0x100, 0x120, 0x140,0x160, 0x180,0x1A0,0x1C0,0x1E0	DMA_Cx_SRC_ADDR		源通道 0/1/2/3/4/5/6/7 地址寄存器
0x104, 0x124, 0x144, 0x164, 0x184,0x1A4,0x1C4,0x1E4	DMA_Cx_DEST_ADDR		目标通道 0/1/2/3/4/5/6/7 地址寄存器
0x108, 0x128, 0x148, 0x168, 0x188,0x1A8,0x1C8,0x1E8	DMA_Cx_LLI		通道 0/1/2/3/4/5/6/7 链接表寄存器
0x10C, 0x12C, 0x14C, 0x16C, 0x18C,0x1AC,0x1CC,0x1EC	DMA_Cx_CTRL		通道 0/1/2/3/4/5/6/7 控制寄存器
0x110, 0x130h, 0x150, 0x170, 0x190 ,0x1B0,0x1D0,0x1F0	DMA_Cx_CONFIG		通道 0/1/2/3/4/5/6/7 配置寄存器

9.5.2. 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

9.5.3. 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。

9.5.4. 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

9.5.5. 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。

9.5.6. 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

9.5.7. 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

9.5.8. 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

9.5.9. 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

9.5.10. 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	ENDIAN	RW	0	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

9.5.11. 源通道地址寄存器(DMA_Cx_SRC_ADDR: 100h, 120h, 140h, 160h, 180h,1A0h,1C0h,1E0h)

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。

9.5.12. 目标通道地址寄存器(DMA_Cx_DEST_ADDR: 104h, 124h, 144h, 164h,184h,1A4h,1C4h,1E4h)

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。

9.5.13. 通道链接表寄存器(DMA_Cx_LLI: 108h, 128h, 148h, 168h,188h,1A8h,1C8h,1E8h)

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.
1:0	RSV	-	-	保留

9.5.14. 通道控制寄存器(DMA_Cx_CTRL: 10Ch, 12Ch, 14Ch, 16Ch,18Ch,1ACh,1CCh,1ECh)

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	DI	RW	0	目标地址递增使能位，置 1 使能目标地址递增，每个 AHB 总线传输后目标地址将加上一个递增量
28	SI	RW	0	源地址递增使能位，置 1 使能源地址递增，每个 AHB 总线传输后源地址将加上一个递增量
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。

9.5.15. 通道配置寄存器(DMA_Cx_CONFIG: 110h, 130h, 150h, 170h,190h,1B0h,1D0h,1F0h)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:24	SRCID	RW	0	源外设 ID。当源是存储器时，该值无效。
23:22	RSV	RW	0	保留

21:16	DESTID	RW	0	目标外设 ID。当目标是存储器时，该值无效。
15:9	RSV	-	-	保留
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。

10. EFLASH 控制器（EFC）

10.1. 概述

芯片上集成了 320KB 的 EFLASH 存储器，用于保存芯片所有的关键脱机信息和数据。EFC 为 EFLASH 控制器，在 CPU 的配置下，完成 eFlash Read、Program、Erase 等操作。

10.2. 主要特性

- 支持 EFLASH 的读（8/16/32bit）、写（32bit）、擦除等操作流程
- Read 等待时间由软件配置
- 主区多达 640 个 page，每个 page 512 字节
- NVR 区有 8 个 page，每个 page 512 字节
- chip 擦除时间 10ms，page 擦除时间为 2ms，word 写 30us（max），读时间 50ns（max）
- 读取保护/写保护功能
- NVR 配置预加载功能

10.3. 功能描述

10.3.1. 闪存结构

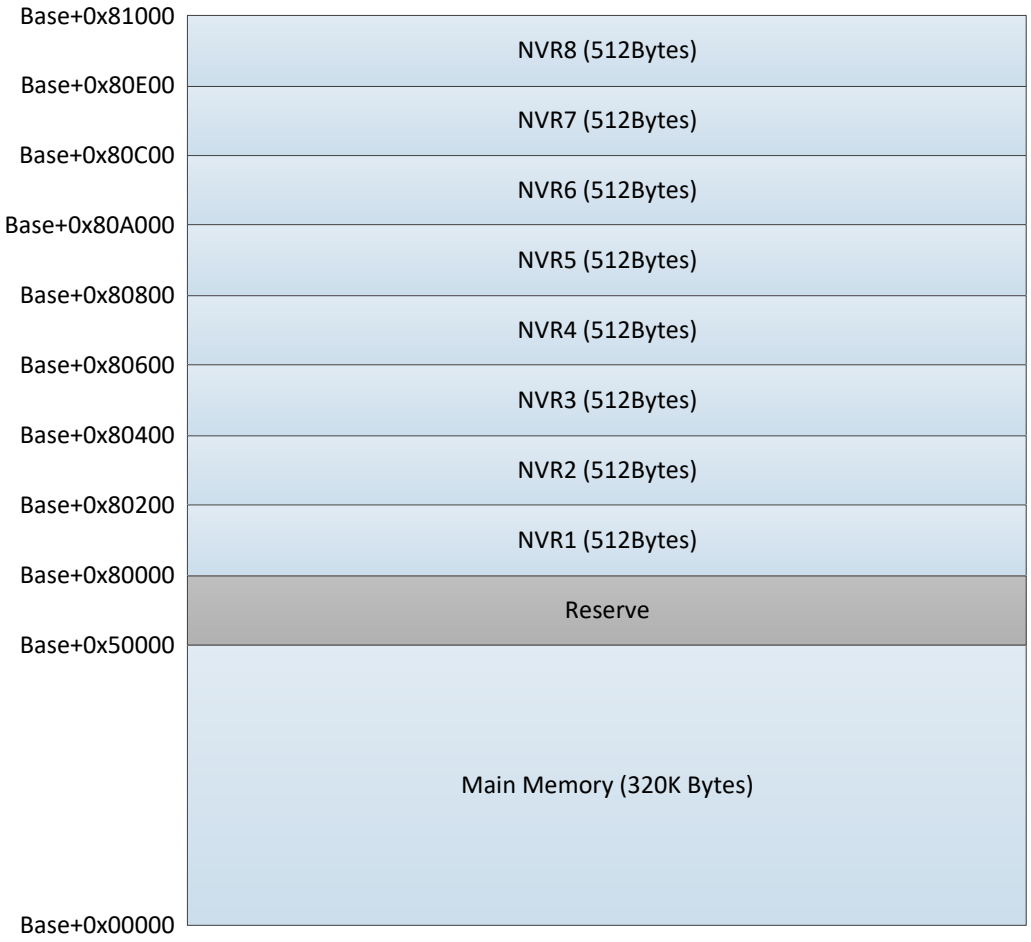
Flash 存储包括主区和 NVR 区两个部分。主存储区由 640 个 page 组成，每个 page 大小为 512 个字节，总共 320KFlash。NVR 区由 8 个 page 组成，每个 page 大小与主区相同为 512 个字节。

表格 10-1 闪存结构

闪存块	名称	地址范围	大小
主区闪存	Page 0	0x00000000~0x000001FF	512Byte
	Page 1	0x00000200~0x000003FF	512Byte
	Page 2	0x00000400~0x000005FF	512Byte
	…	…	…
	Page 639	0x0004FE00~0x0004FFFF	512Byte
NVR 闪存	Page 0	0x00080000~0x000801FF	512Byte
	Page 1	0x00080200~0x000803FF	512Byte
	Page 2	0x00080400~0x000805FF	512Byte
	…	…	…
	Page 7	0x00080E00~0x00080FFF	512Byte

10.3.2. 储存地址映射

图 10-1 EFLASH 地址映射



注：Base：ROM 启动为 0x1000_0000；EFLASH 启动为 0x0000_0000。

10.3.3. 存储保护

Flash 支持 RDP 读保护、WDP 写保护和 PCROP 专有代码读保护。下文对这些功能进行介绍，具体使用操作请参考《航芯 G103 系列芯片存储保护功能.docx》

10.3.3.1. RDP 读保护

RDP 读保护是一种全局 FLASH 读保护。可以防止代码拷贝、反向工程、JTAG 读取等攻击。用户应在固件下载完成后，再使能 RDP 保护。读保护分为 RDP L0 和 RDP L1 两个级别。

■ RDP L0

L0 级保护是默认的等级，FLASH 是完全开放的，BOOT 模式和 JTAG 模式下可对 FLASH 进行各种读写擦操作。

■ RDP L1

当 L1 保护等级激活，BOOT 模式和 JTAG 模式下不可访问 FLASH。但是从 FLASH 启动的用户代码是可以访问 FLASH 的。

当 L1 保护等级激活后，代码的更新不能通过 BOOTLOADER 或 JTAG 进行。但用户代码可以自行设计 IAP 功能来自更新代码。

■ RDP L1 激活

当芯片处于 L0 保护等级，可以通过下载工具使能 RDP L1，或者在用户代码中使能 RDP L1。

■ RDP L1 降级

当芯片处于 RDP L1 保护等级，如果用户希望回到 BOOT 模式更新代码，则需要进行 RDP 降级，让芯片回到 RDP L0 等级。

从 L1 降级到 L0 会导致 FLASH 数据全部擦除。

10.3.3.2. WRP 写保护

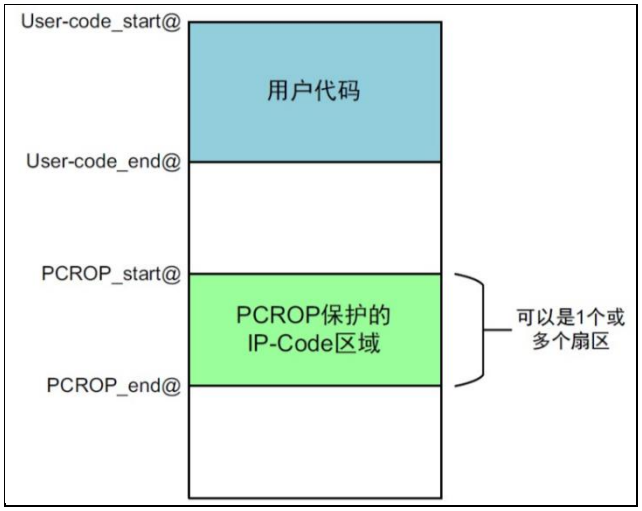
WRP 被用来保护特定扇区（以 2KB 为单位）的内容，防止代码被擦除或重写。

写保护可以通过下载工具或者在用户代码中使能。

10.3.3.3. PCROP 专有代码读保护

PCROP 是一个专有代码读出保护的功能。与 RDP 对整片 Flash 读保护不同的是，它只是针对 Flash 的某些特定区域进行代码的读写保护。它可以被用来保护一些 IP 代码，方便进行二次开发。

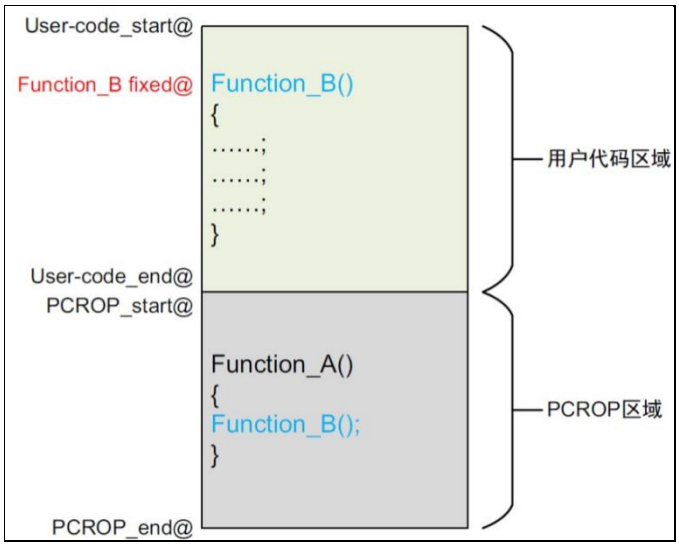
图 10-2 PCROP 保护



受 PCROP 保护的 IP 代码可以随意地被用户应用程序调用运行，同时又防止外界对 IP 代码的直接读写访问。

PCROP 区的代码也可以调用 PCROP 区外的处于固定地址的函数。

图 10-3 PCROP 保护区函数调用 PCROP 区外函数



受 PCROP 保护的区域是无法使用 D-Code 总线进行读访问的，所以在这片区域中只允许执行指令代码（通过 I-Code 总线取指令），数据读取是被禁止的。因此，受保护的 IP 代码不能访问存储于同一块区域内的关联数据，比如文字池（literal pools）、分支表（branch tables）以及在执行过程中需要通过 D-code 总线进行读取的常量数据。

换言之，受 PCROP 保护的代码只能是只执行的指令代码，而不包含任何数据。因此，我们在编译受 PCROP 保护的 IP 代码时，必须对其进行相应配置，以避免在 PRROP 区域生成文字池、常量数据等。

MCU 的中断向量表里都是些常量数据，所以包含中断向量表的扇区不可进行 PCROP。一般来讲向量表放在第一个扇区，所以该扇区不可进行 PCROP。

10.3.3.4. 用于存储保护的 NVR 寄存器

芯片 EFLASH 的 NVR 区域有一些专用于存储保护的寄存器（32bit 位宽）：

NVR 起始地址：Base + 0x00080000

表格 10-2 NVR 内存分配

名称	地址偏移	描述	默认值
REMAP	0x400	安全启动序列	not 0x89bc3f51：BOOT 启动 (default)； 0x89bc3f51：flash 启动。
RDP1_DEGRADE	0x488	RDP1 降级标志 是否需要 RDP 降级，高低 16 位取反有效	高低 16 位未取反：不需要 RDP 降级 (default)； 高低 16 位取反：申请 RDP 降级
PCROP_EN	0x600	Flash PCROP 保护使能。使能后，PCROP area A/B 只能执行，不能读取或者擦写。	not 0x89bc3f51：PCROP 功能不使能(default)； 0x89bc3f51：PCROP 功能使能
PCROP_AREA_A	0x604	PCROP area A 地址定义，以 page（512 字节）为单位。 [9:0] ： PCROP1A_STRT[9:0] [25:16] ： PCROP1A_END[9:0] 起始地址 PCROP1A_STRT*0x200（包括） 结束地址(PCROP1A_END+1)*0x200（不包括）	
PCROP_AREA_B	0x608	PCROP area B 地址定义，以 page（512 字节）为单位。 [9:0] ： PCROP1B_STRT[9:0] [25:16] ： PCROP1B_END[9:0] 起始地址 PCROP1B_STRT*0x200（包括） 结束地址(PCROP1B_END+1)*0x200（不包括）	
WRP_EN	0x620	Flash WRP 保护使能。使能后，WRP areaA/B 禁止擦写。	not 0x89bc3f51：WRP 功能不使能 (default)； 0x89bc3f51：WRP 功能使能
WRP_AREA_A	0x624	WRPP area A 地址定义，以 2K 字节为单位。 [7:0] ： WRP1A_STRT[7:0] [23:16] ： WRP1A_END[7:0] 起始地址 WRP1A_STRT*0x800（包括） 结束地址(WRP1A_END+1)*0x800（不包括）	

WRP_AREA_B	0x628	WRPP area B 地址定义，以 2K 字节为单位。 [7:0] ： WRP1B_STRT[7:0] [23:16] ： WRP1B_END[7:0] 起始地址 WRP1B_STRT*0x800（包括） 结束地址(WRP1B_END+1)*0x800（不包括）	
RDP1_EN	0x660	RDP1 使能标志。使能 RDP1 功能后，BOOT 模式和 JTAG 无法读取 Flash 主区内容	not 0x89bc3f51：RDP1 功能不使能(default)； 0x89bc3f51：RDP 功能使能

用户代码中对 NVR 区域寄存器的修改，需要需要读出整页，修改寄存器对应偏移地址数据，再擦除和编程该页。

10.4. 使用说明

10.4.1. 读（Read）操作

闪存的指令和数据访问是通过 AHB 总线完成的。Core 的 Flash 访问是通过 Cbus 进行。读访问可以有以下配置选项：

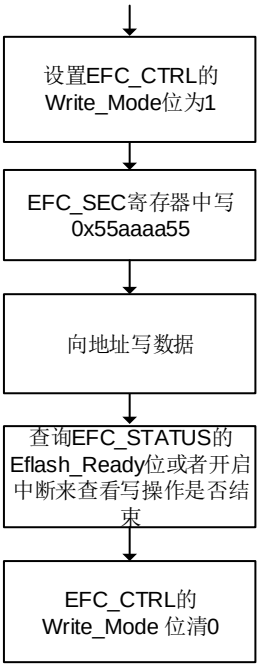
- 等待时间 RD_WAIT：可以随时更改的用于读取操作的等待状态的数量。读操作需要配置的 RD_WAIT 最小值需要保证读等待时间至少为 50ns，即(Rd_Wait+1.0)*系统频率周期≥50ns，配置太小会导致读取 eFlash 数据出错。

10.4.2. 编程（Program）操作

Program 操作，CPU 通过执行字（32bit）来对主闪存进行编程写入操作。软件必须设置 EFC_CTRL 寄存器中的 Write_Mode 位为 1；然后再对目标地址写数据。如果寻址的主闪速存储器区域被 WRP 写保护，则跳过编程操作，并设置 EFC_INTSTATUS 的 WRPERR 位发出警告。

Program 操作之前需要向 EFC_SEC 寄存器内写 0x55aaaa55，否则 EFC 忽略此次写操作。每次只能写一个 Word。等待 EFC_STATUS 的 Eflash_Ready 位变高或者开启 ErWr _done 中断来查看写操作是否结束。写完成后，清除 EFC_CTRL 的 Write_Mode 位。

图 10-4 Program 操作流程



10.4.3. 擦除（Erase）操作

Erase 操作分为 Page Erase 和 Chip Erase。Page Erase 时，每次只能擦除选中的一个 Page。Chip Erase 可以擦除整个主区，NVR 区不受影响。

Page Erase 应遵循以下步骤：

1）在 EFC_CTRL 寄存器中设置 Page_Erase_Mode 位为 1

2）向 EFC_SEC 寄存器内写 0x55aaaa55

3）对要擦除的 page 任意地址写 0，启动擦除

4）等待 EFC_STATUS 的 Eflash_Ready 变高或者开启 ErWr _done 中断

5）清除 EFC_CTRL 寄存器的 Page_Erase_Mode 位

6）读取擦除的页面并验证（可选）

Chip Erase 应遵循以下步骤：

1）在 EFC_CTRL 寄存器中设置 Chip_Erase_Mode 位为 1

2）向 EFC_SEC 寄存器内写 0x55aaaa55，

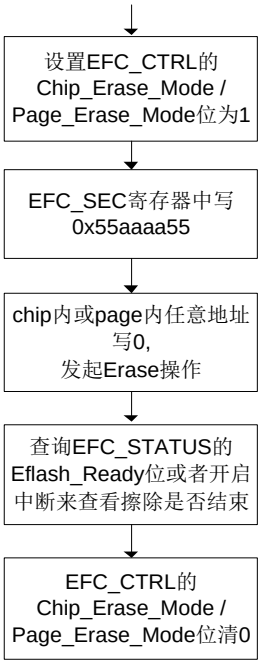
3）对主区任意地址写 0，启动擦除

4）等待 EFC_STATUS 的 Eflash_Ready 变高或者开启 ErWr _done 中断

5）清除 EFC_CTRL 寄存器的 Chip_Erase_Mode 位

6）读取主区并验证（可选）

图 10-5 Erase 操作流程



10.5. EFC 寄存器描述

寄存器基地址：0x40022000。

偏移	名称	描述
0x00	EFC_CTRL	控制寄存器
0x04	EFC_SEC	写擦除操作安全寄存器
0x08	EFC_ADCT	写擦除地址校验寄存器
0x0c	EFC_ERTO	擦除超时寄存器
0x10	EFC_WRTO	写超时寄存器
0x14	EFC_STATUS	状态寄存器
0x18	EFC_INTSTATUS	中断状态寄存器
0x1c	EFC_INEN	中断使能寄存器

10.5.1. 控制寄存器(EFC_CTRL: 00h)

位域	名称	属性	默认值	功能描述
31:15	RSV	-	-	保留

14	EWCNTEN	RW	0	Flash 擦/写完成标志选择位 1：Flash 在擦写操作时，将不再依靠 TBIT 信号（TBIT 为 1 表示 Flash 处于擦/写状态）判断擦写是否完成，而是由 EFC_ERTO 和 EFC_WRTO 来判断擦写是否完成。此位为 1 时，Fto_En 位也需要置 1，否则控制器将不会判断擦写是否完成，控制器将无法退出擦写操作 0：使用 TBIT 信号作为擦写完成的判断。当此位为 0 时，如果 Fto_En 位为 1，则当 EFC_ERTO 和 EFC_WRTO 定时超时或者 TBIT 信号有一个条件成立时，系统即退出擦写操作 注：TBIT 是 eFlash 存储器输出的一个信号，反映了 EFlash 的擦写是否完成
13	RESETB	RW	1	eFlash 复位控制位 0：eFlash 复位，任何操作发起无效 1：eFlash 复位释放，可以发起操作 任何操作需要在 Resetb 写 1 后延迟 10us 后才能发起
12	SLEEP	RW	0	eFlash Sleep 模式启动位 0：eFlash 退出 SLEEP 模式 1：eFlash 进入 SLEEP 模式 退出 SLEEP 模式时，要在 SLEEP 写 0 时延迟 10us，再去操作 eFlash。进入 SLEEP 模式的最短时间为 20us
11:7	RD_WAIT	RW	5	读等待时间设置位。读操作时 eFlash 端口等待的时钟周期数为 Rd_wait+1.0 eFlash 要求读等待时间至少为 50ns，即(Rd_Wait+1.0)*系统频率周期≥ 50ns （Recall read 模式下为 95ns） 0：等待 1.0 个系统时钟周期 1：等待 2.0 个系统时钟周期 .... MCU 取址时间为: Rd_wait + 3
6	FTO_EN	RW	0	写/擦除操作超时使能 1：写/擦除超时功能使能 0：写/擦除超时功能禁止
5:3	RSV	-	-	保留
2	CHIP_ERASE_MODE	RW	0	Chip Erase Mode 模式设置位 1：Chip Erase Mode 模式使能 0：Chip Erase Mode 模式禁止
1	PAGE_ERASE_MODE	RW	0	Page Erase Mode 模式设置位 1：Page Erase Mode 模式使能 0：Page Erase Mode 模式禁止
0	WRITE_MODE	RW	0	Write 模式设置位 1：写操作模式使能 0：写操作模式禁止

10.5.2. 写擦安全寄存器(EFC_SEC: 04h)

位域	名称	属性	默认值	功能描述
31:0	WRITE_LOCK_SER	W	0	向 eFlash 写数据/擦除前，须向此寄存器内写 0x55aaaa55 值，否则控制其忽略此次写操作

10.5.3. 擦除超时寄存器(EFC_ERTO: 0Ch)

位域	名称	属性	默认值	功能描述
31:15	RSV	-	-	保留
14:0	ERSD	RW	0x7fff	擦除 TimeOut 设置位。每一个单位代表 256 个系统时钟周期。当 eFlash 擦除操作在此时间断内未结束时，控制器将认为此操作有误，产生 TimeOut 状态信息，并复位内部状态机

10.5.4. 写超时寄存器(EFC_WRTO: 10h)

位域	名称	属性	默认值	功能描述
31:8	RSV	-	-	保留
7:0	WRCYCLE	RW	0xff	写操作 TimeOut 设置位。每一个单位代表 256 个系统时钟周期。当 eFlash 写操作在此时间断内未结束时，控制器将认为此操作有误，产生 TimeOut 状态信息，并复位内部状态机

10.5.5. 状态寄存器(EFC_STATUS: 14h)

位域	名称	属性	默认值	功能描述
31:18	RSV	-	-	保留
17	TBIT	RO	0	此位反映 eFlash TBIT 信号真实状态 0：Flash 未处于擦/写状态 1：Flash 处于擦/写状态
16:14	RSV	-	-	保留
13	RDP1_EN	RO	0	RDP Level1 使能 0：RDP level1 未使能 1：RDP level1 使能
12:9	RSV	-	-	保留
8	FLASH_CRYPT	RO	0	Flash 加解密状态 0：Flash 加密功能禁止 1：Flash 加密功能使能 芯片出厂是加密使能的，不能设置为加密禁止
7	JTAG_DISABLE	RO	0	JTAG 禁止状态 0：JTAG 接口未禁止 1：JTAG 接口禁止
6:5	RSV	-	-	保留
4	NVR4_LOCK	RO	0/1	Nvr4 区是否锁住 1：Nvr4 区已经锁住 0：Nvr4 区没有锁住

3	NVR3_LOCK	RO	0/1	Nvr3 区是否锁住。 1：Nvr3 区已经锁住 0：Nvr3 区没有锁住
2	NVR2_LOCK	RO	1	Nvr2 区是否锁住 1：Nvr2 区已经锁住 0：Nvr2 区没有锁住
1	NVR1_LOCK	RO	0/1	Nvr1 区是否锁住 1：Nvr1 区已经锁住 0：Nvr1 区没有锁住
0	EFLASH_READY	RO	1	eFlash 状态指示位。该反映 eFlash 工作的状态 1：eFlash 状态空闲 0：eFlash 状态忙

10.5.6. 中断状态寄存器(EFC_INTSTATUS: 18h)

位域	名称	属性	默认值	功能描述
31:11	RSV	-	-	保留
10	RD_PERR	RWC	0	读 Flash 产生校验错误 1：发生读 Flash 校验错误 0：未发生读 Flash 校验错误 写 1 清 0
9	ERWR_TO_ERR	RWC	0	擦除/写操作发生 TimeOut 中断状态位 1：发生擦除/写操作 TimeOut 0：未发生擦除/写操作 TimeOut 写 1 清 0
8	WRPERR	RWC	0	WPR 错误中断标志，对 PCROP 和 WRP 保护区域进行擦写操作产生错误标志 0：无 WRP 读错误 1：WRP 读错误 写 1 清 0
7	PCROPERR	RWC	0	PCROP 读错误中断标志，对 PCROP 保护区域进行读操作产生错误标志 0：无 PCROP 读错误 1：PCROP 读错误 写 1 清 0
6	RSV	-	-	保留
5	NVR_ERR	RW	0	1：Nvr 区发生操作错误。对 Lock 住的 Nvr 区域进行读之外的任何操作，或者对任一 Nvr 区域进行 Chip Erase 操作时（不论有没有 Lock 住），此位均会置 1 0：正常状态 写 1 清 0
4:1	RSV	-	-	保留

0	ERWR _DONE	RW	0	写/擦除完成中断状态位 1：写/擦除完成标志。写 1 清除该位。如果中断允许，则产生中断 0：写/擦除未完成

10.5.7. 中断使能寄存器(EFC_INEN: 1Ch)

位域	名称	属性	默认值	功能描述
31:11	RSV	-	-	保留
10	RD_ERR_IE	RW	0	读 Flash 发生校验错误中断使能 1：RD Error 中断使能 0：RD Error 中断不使能
9	ERWR_TO_IE	RW	0	擦除/写操作发生 TimeOut 中断使能 1：擦除/写操作 TimeOut 中断使能 0：擦除/写操作 TimeOut 中断不使能
8	WRPERR_IE	RW	0	WRPERR 中断使能 1：WRPERR 中断使能 0：WRPERR 中断不使能
7	RDERR_IE	RW	0	RDERR 中断使能 1：RDERR 中断使能 0：RDERR 中断不使能
6	RSV	-	-	保留
5	NVR_ERR_IE	RW	0	Nvr_Err 中断使能 1：Nvr_Err 中断使能 0：Nvr_Err 中断不使能
4:1	RSV	-	-	保留
0	ER_DONE_IE	RW	0	擦除完成中断使能 1：写/擦除中断使能 0：写/擦除中断不使能

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；

通用定时器根据不同配置又分为三个等级，如下表所示。

表 11-1 定时器的类型

定时器	高级 TIM1/TIM8	通用 L0 TIM2/TIM3/TIM4	通用 L2 TIM15	通用 L3 TIM16/TIM17	基本 TIM6/TIM7
预分频器	16 位	16 位	16 位	16 位	16 位
计数器	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
重复计数器	●	×	●	●	×
通道数	4	4	2	1	0
带死区时间插入的互补输出	●	×	●	●	×
刹车输入	●	×	●	●	×
单脉冲模式	●	●	●	●	●
正交编码	●	●	●	×	×
外部时钟模式 2	●	●	×	×	×
外部事件清除 OCxREF	●	●	×	×	×
6 步 PWM	●	×	×	×	×
强制输出模式	●	●	●	●	×
三路输入异或	●	●	×	×	×
DMA	●	●	●	●	●

●代表支持该功能。×代表不支持该功能。

12. 高级定时器（TIM1/TIM8）

12.1. 概述

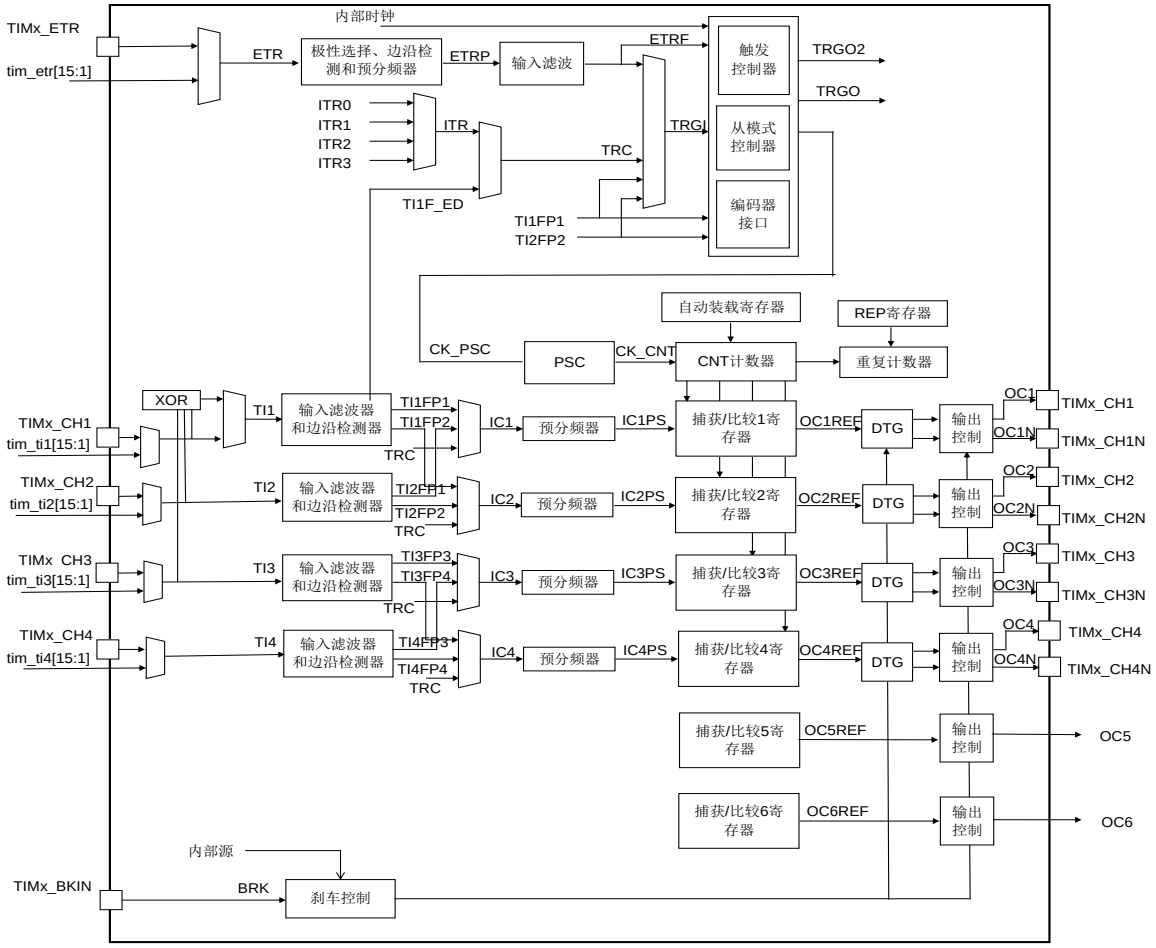
高级控制定时器（TIM1/TIM8）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

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	-	-

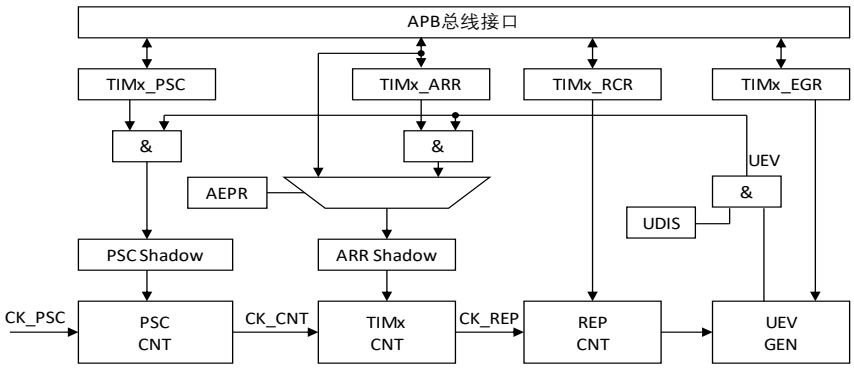
12.5. 功能描述

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

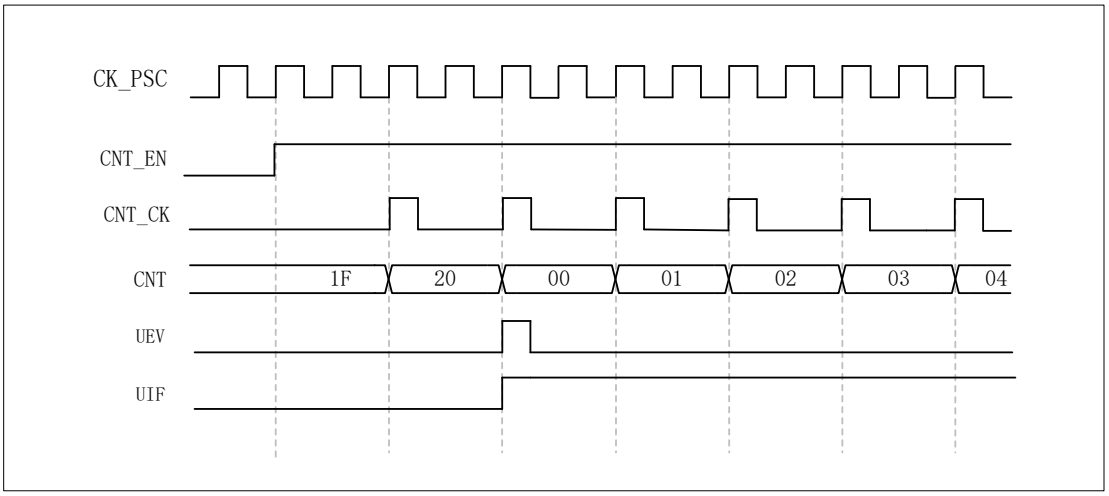
在向上计数模式中，计数器从 0 计数到自动重载载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件，如果 UDIS=0，就会产生一次更新事件。更新事件也可以由 UG 位置位产生（通过 EGR 寄存器软件置位或通过硬件从模式方式)。当更新事件产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件产生的更新事件才产生更新中断和更新 DMA 请求。

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) 。

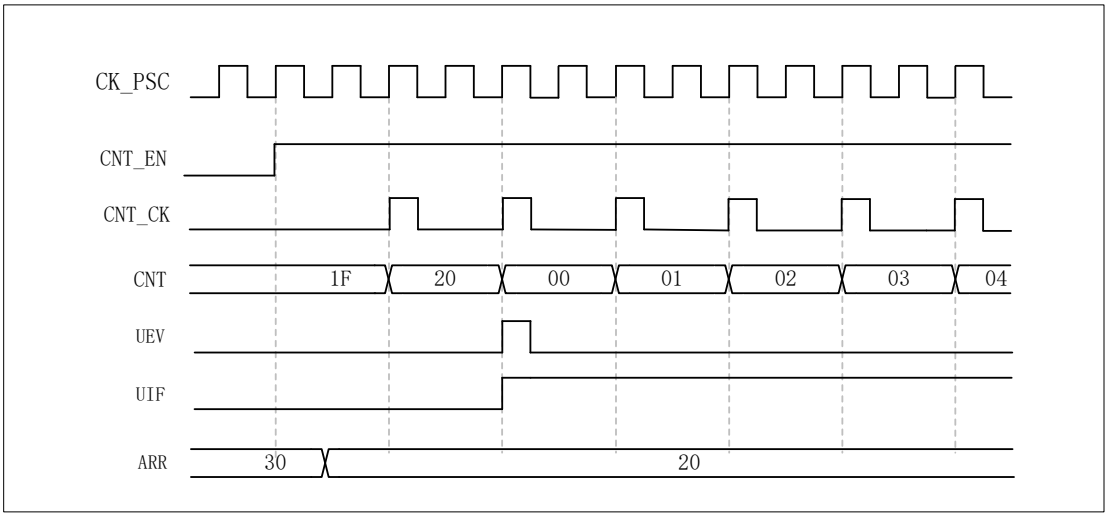
下图示例说明当 TIMx_ARR=0x20 时二分频下计数器的行为示意图。

图 12-3 向上计数模式更新事件



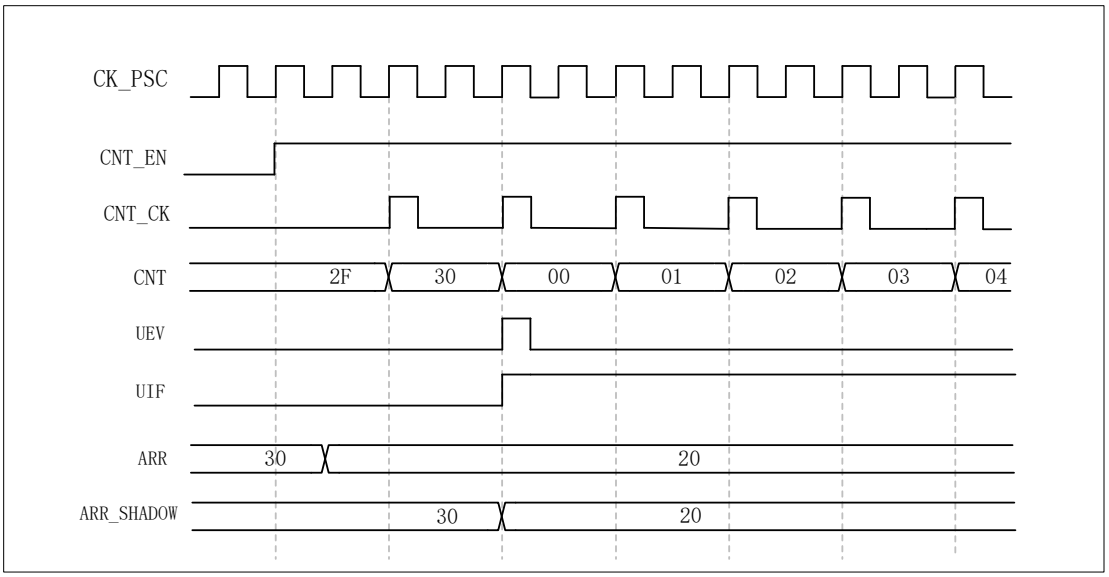
下图为 ARPE=0 时计数器的行为示意图：

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

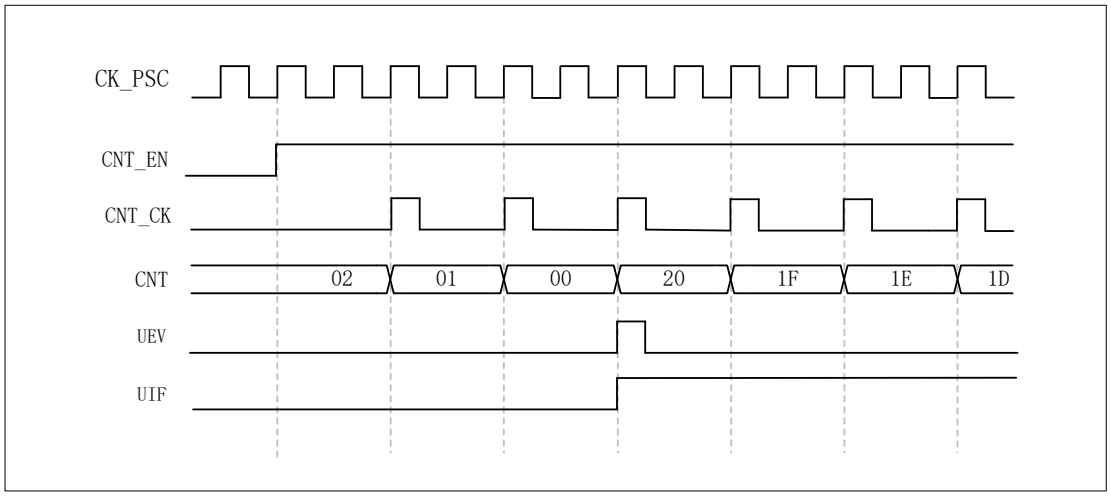
在向下模式中，计数器从自动重载值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动重载值重新开始并且产生一个计数器向下溢出事件，如果 UDIS=0，就会产生一次更新事件。更新事件也可以由 UG bit 置位产生（通过 EGR 寄存器软件置位或通过硬件从模式方式）。当更新事件产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件产生的更新事件才产生更新中断和更新 DMA 请求。

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) 。

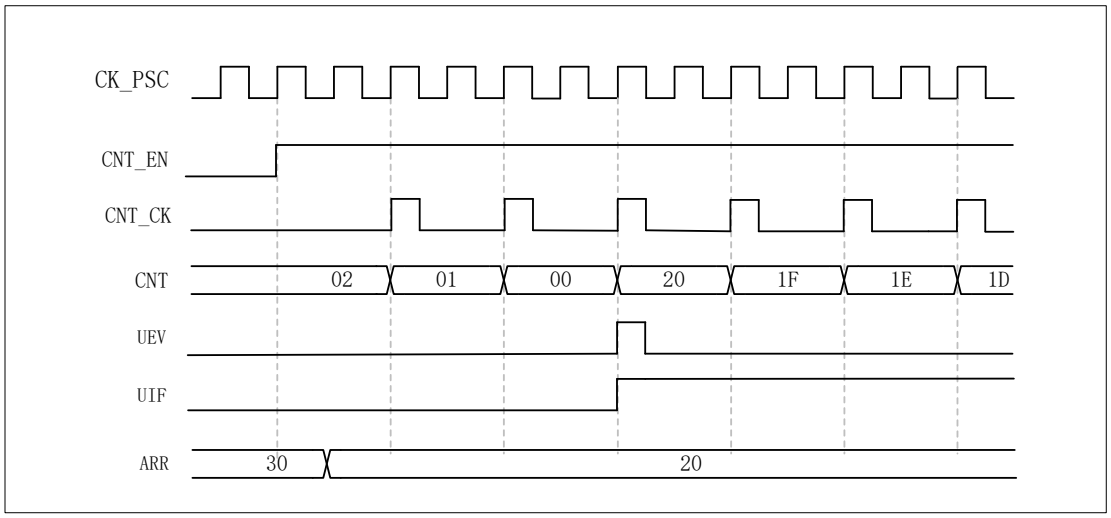
下图示例说明当 TIMx_ARR=0x20 时二分频下计数器的行为示意图。

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：

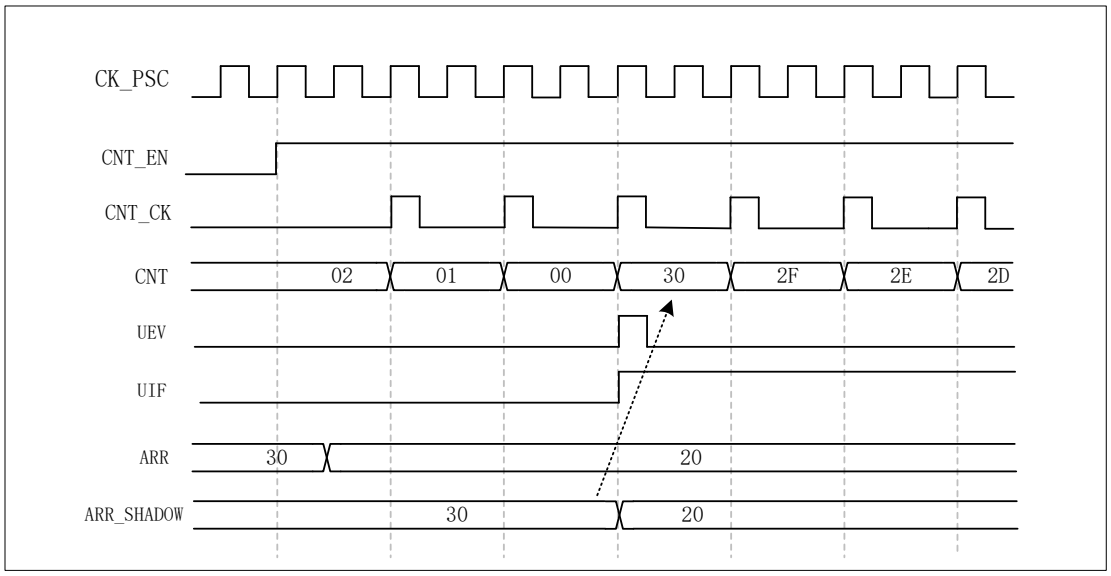
图 12-7 向上计数模式（禁止自动重装载预装载）



下图为 ARPE=1 时计数器的行为示意图：

当下溢事件脉冲发生时，ARR 值更新至影子寄存器，同时影子寄存器的值需要更新至 CNT 寄存器，因此会将上一次的 ARR 值更新至 CNT，等下一次下溢事件发生时，影子寄存器值生效。

图 12-8 向上计数模式（使能自动重装载预装载）



● 中央对齐模式

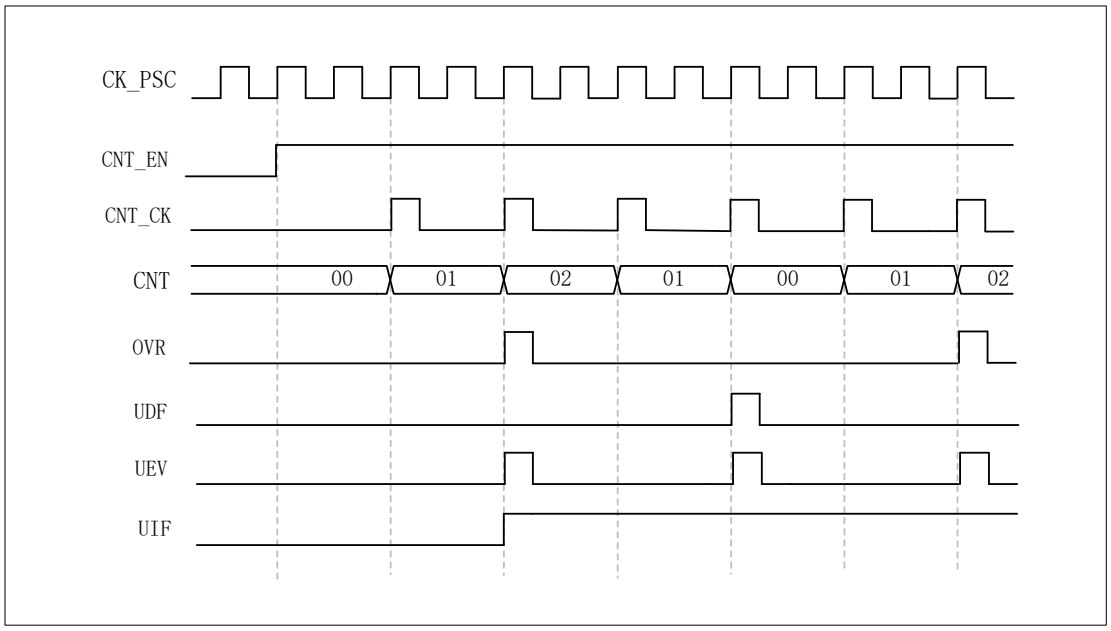
在中央对齐模式下，计数器交替的从 0 开始向上计数到 ARR-1，产生一次上溢出事件，然后再从 ARR 向下计数到 1，然后产生下溢出事件，然后又从 0 开始计数。向上计数时，定时器模块在计数器计数到自动加载值减一则产生一个上溢事件；向下计数时，定时器模块在计数器计数到 1 时产生一个下溢事件。无论上溢还是下溢，当 UDIS=0 时，都会产生一次更新事件。当更新事件（UEV/UG/硬件从模式方式）产生后，如果 TIMx_CR1 的 URS=0，并且更新中断或 DMA 已使能，就会产生更新中断或更新 DMA 请求；如果 URS=1，则只有溢出事件（UEV）产生的更新事件才产生更新中断和更新 DMA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

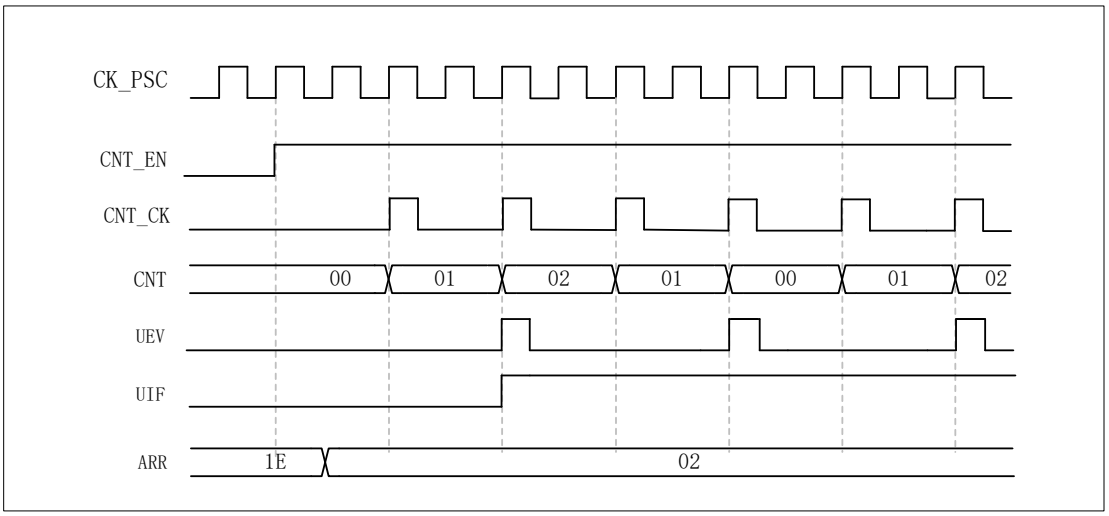
下图示例说明当 TIMx_ARR=0x02 时二分频下计数器的行为示意图。

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：

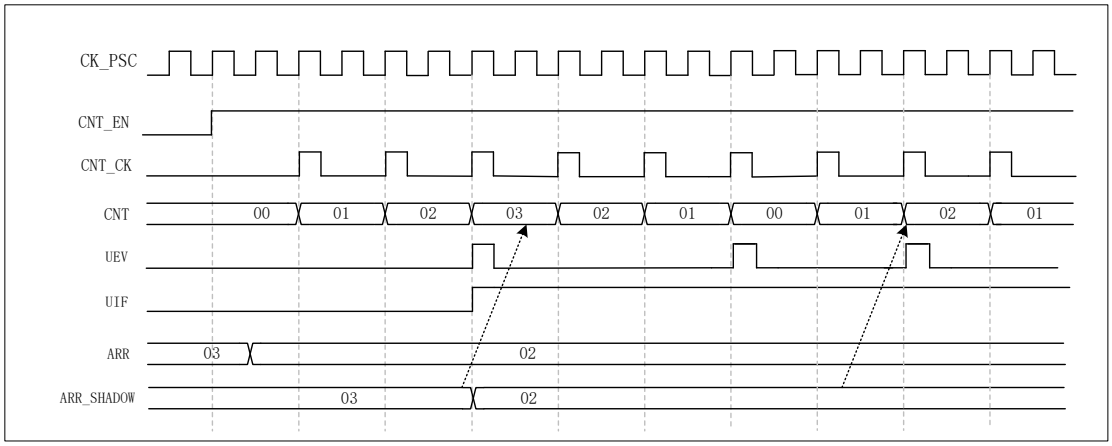
图 12-10 中央对齐模式（禁止自动重装载预装载）



下图为 ARPE=1 时计数器的行为示意图：

当上溢事件脉冲发生时，ARR 值更新至影子寄存器，同时影子寄存器的值需要更新至 CNT 寄存器，因此 CNT 会将上一次的影子寄存器的值更新至 CNT，等下一次上溢事件发生时，影子寄存器值生效。

图 12-11 中央对齐模式（使能自动重装载预装载）

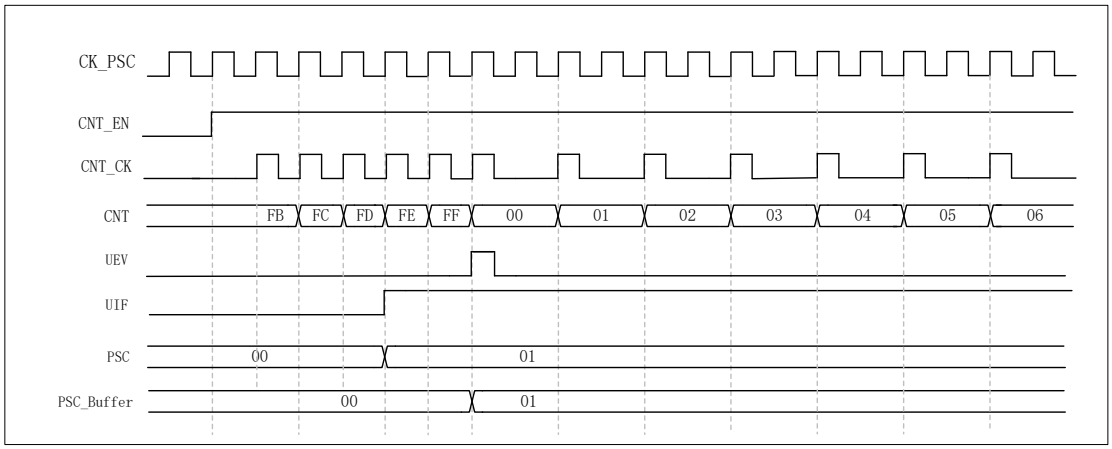


12.5.2. 预分频器

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



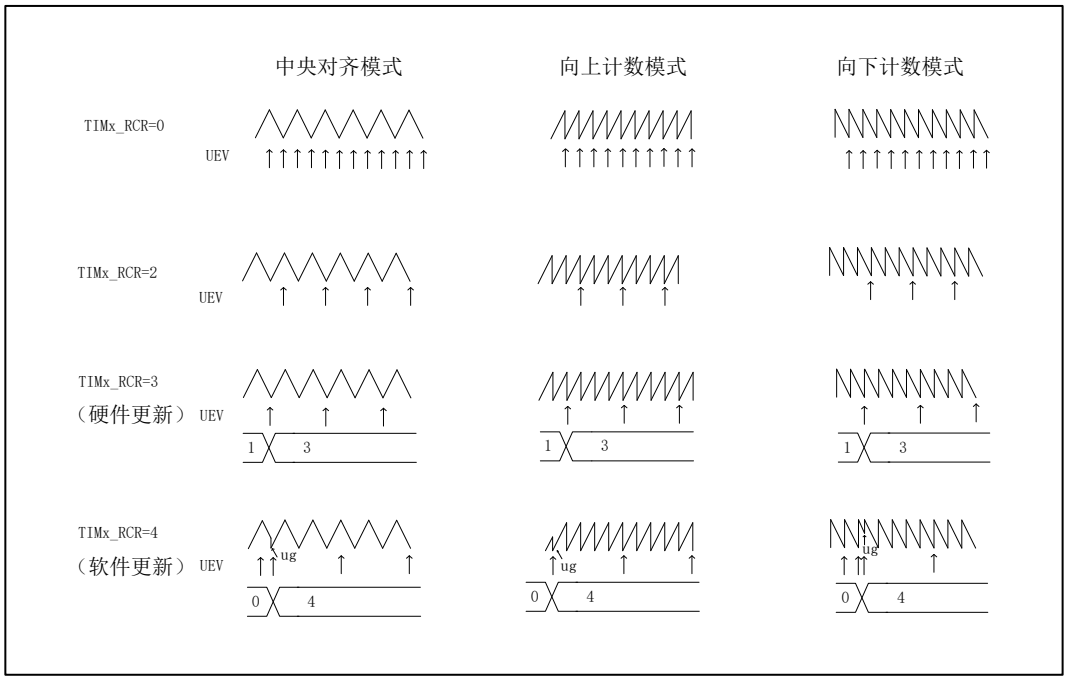
12.5.3. 重复计数器

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



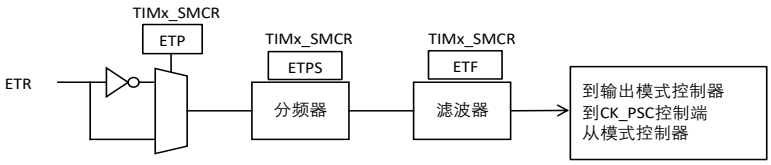
12.5.4. 外部触发输入

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

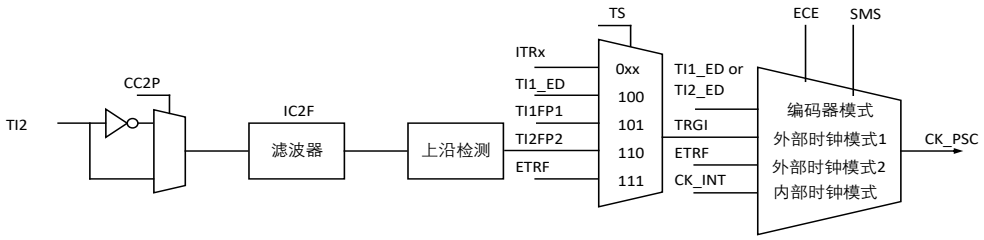
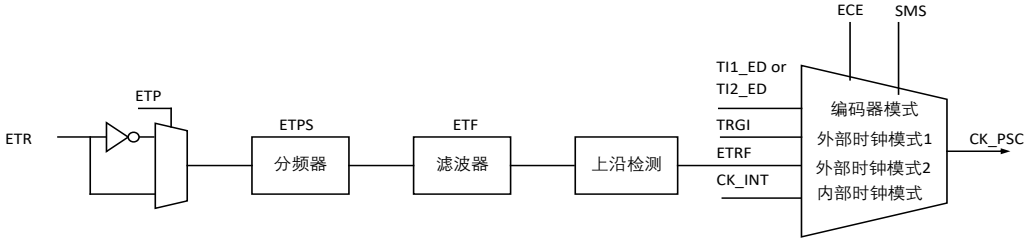


图 12-16 外部时钟模式 2



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。

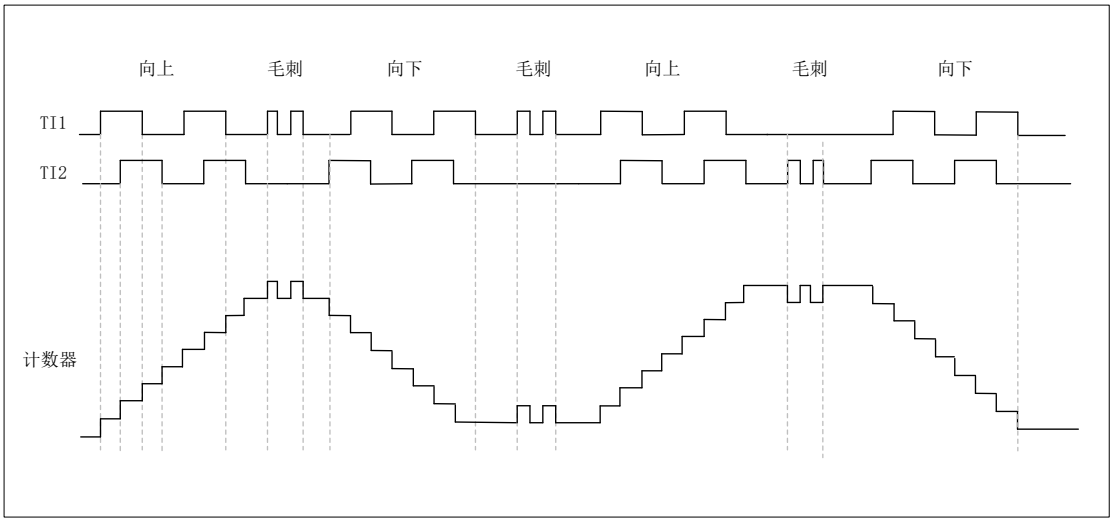
表 12-8 计数器方向和编码器信号的关系

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



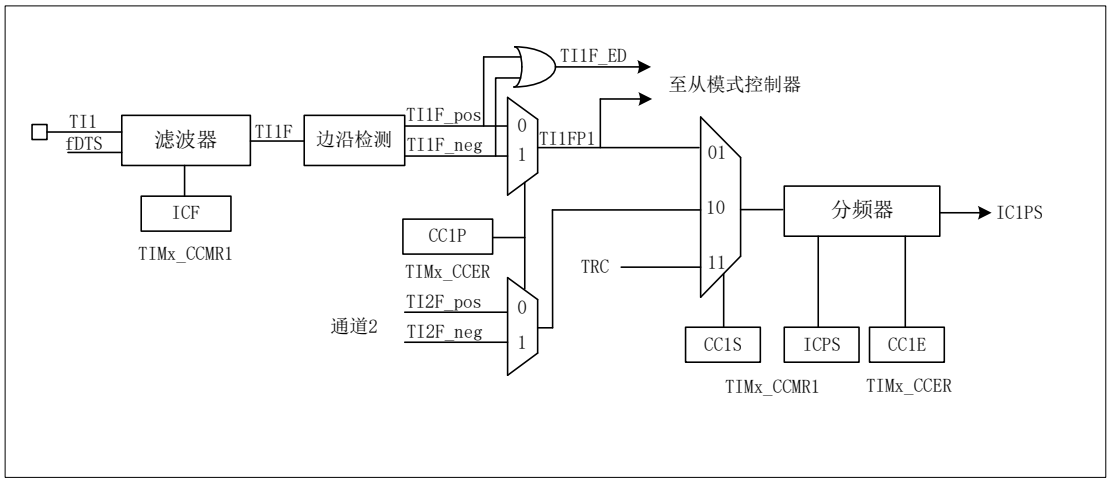
12.5.7. 捕获/比较通道

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

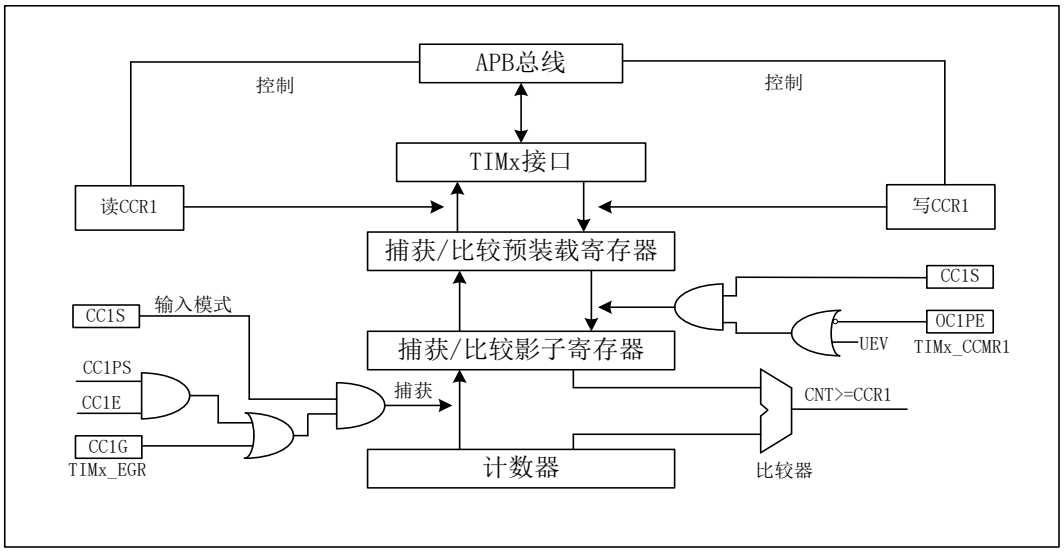


图 12-20 捕获/比较通道的输出部分(通道 1 至 3)

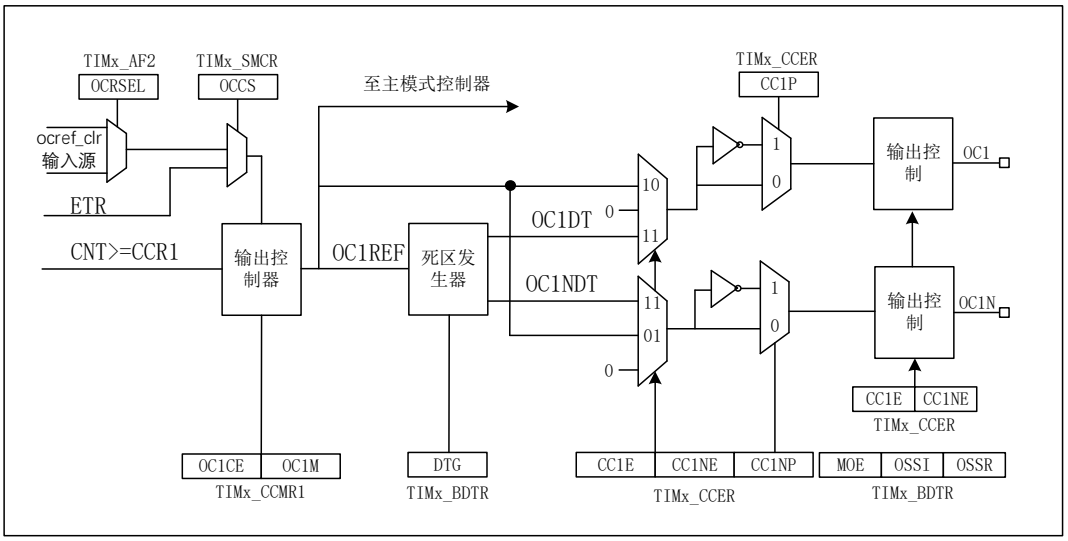
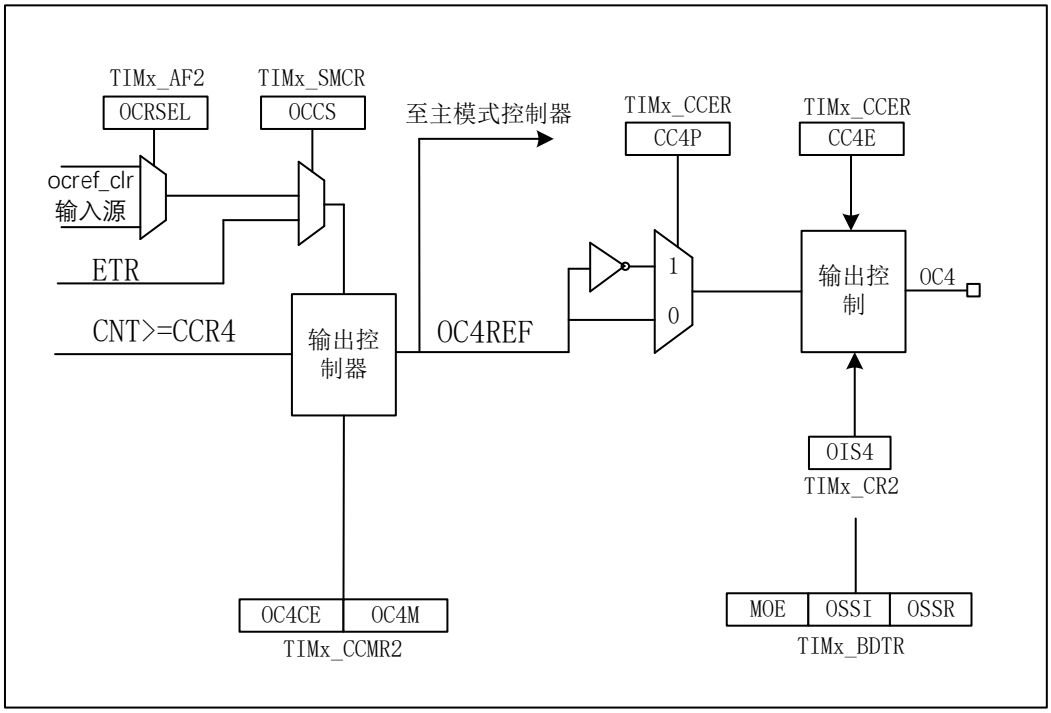


图 12-21 捕获/比较通道的输出部分(通道 4)



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_CCR1 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)。

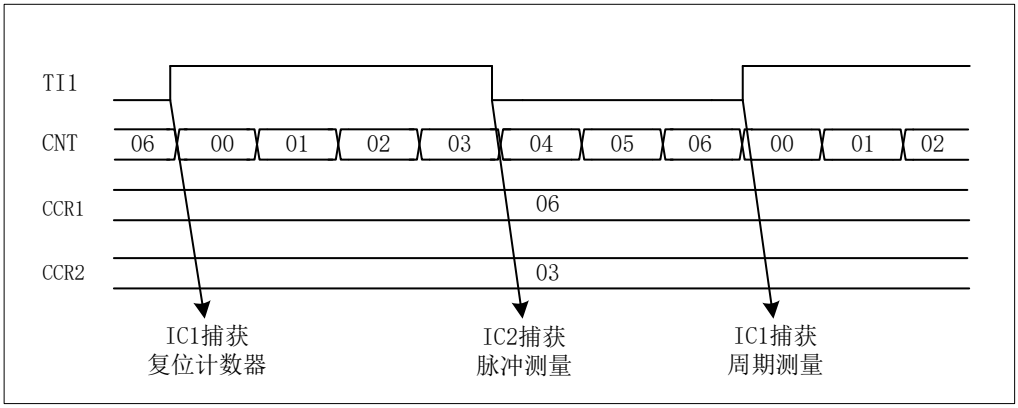
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

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

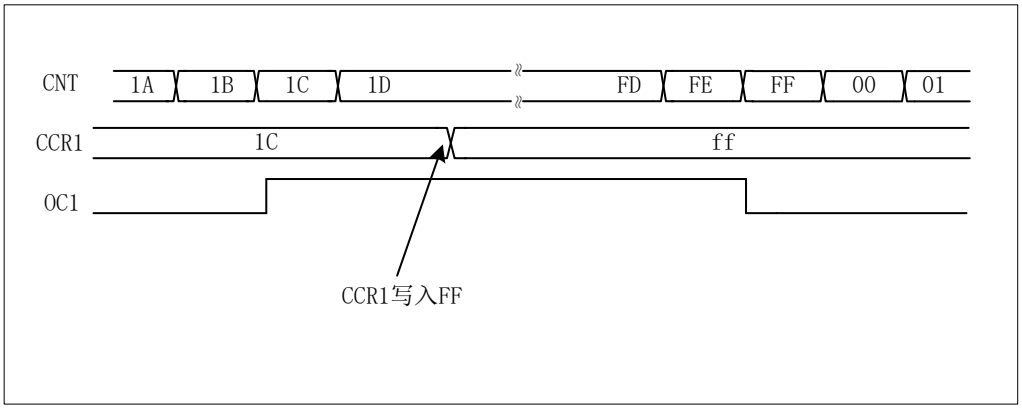
c）置 CCxP = 0 选择极性为高电平有效

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。

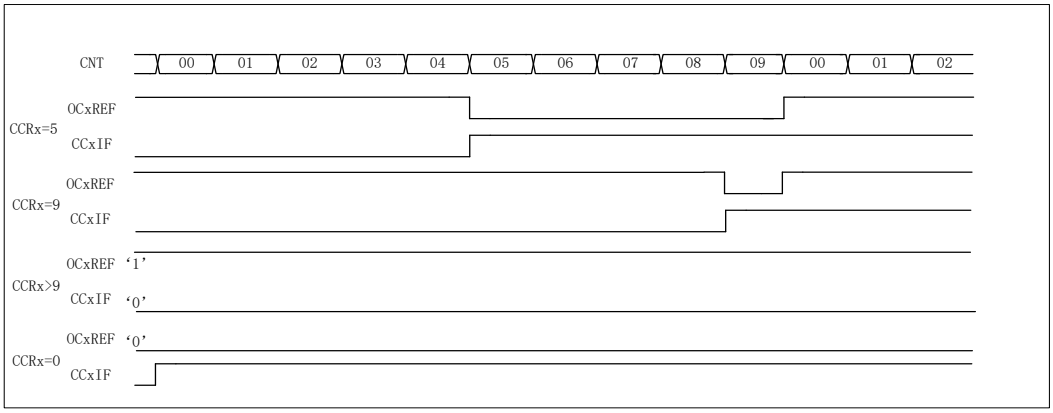
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ 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。

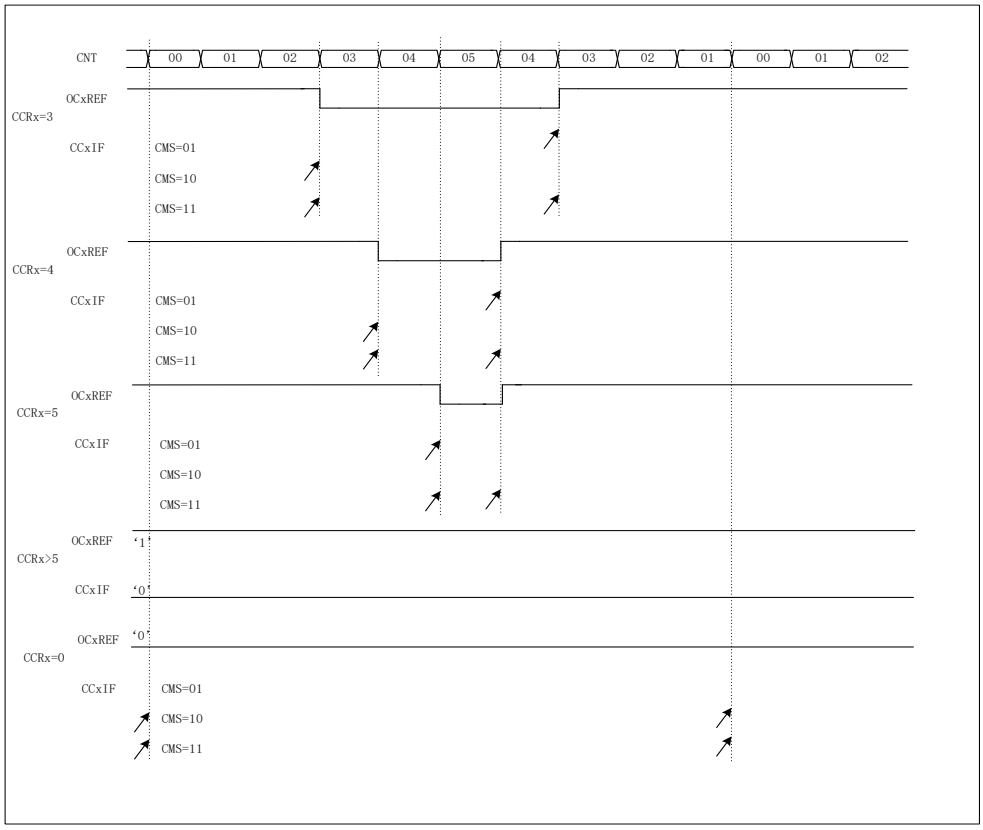
下图给出了一些中央对齐的 PWM 波形的例子。

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

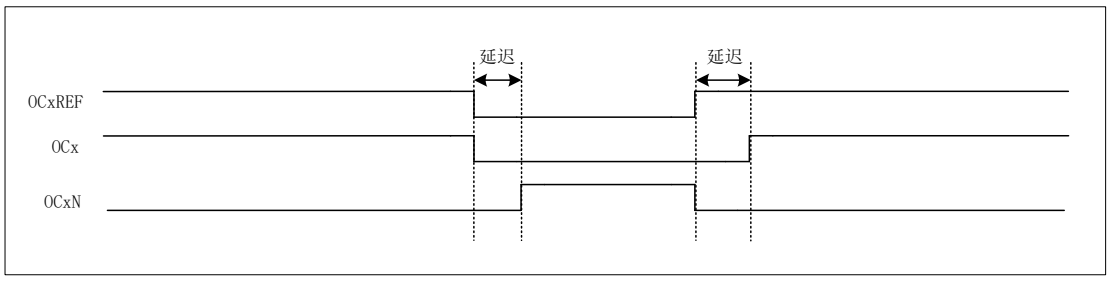


图 12-27 死区波形延迟大于负脉冲

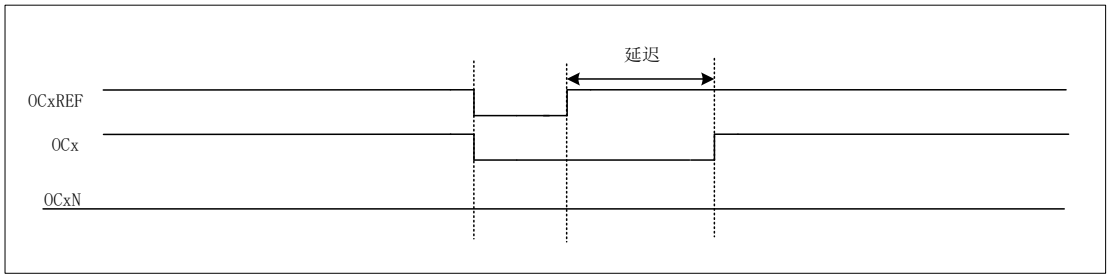
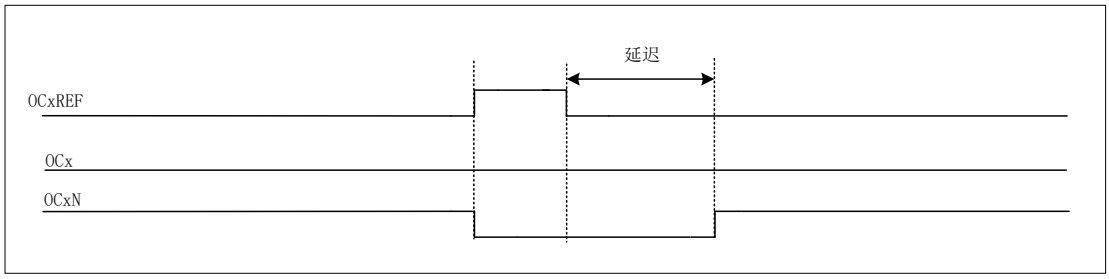


图 12-28 死区波形延迟大于正脉冲



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

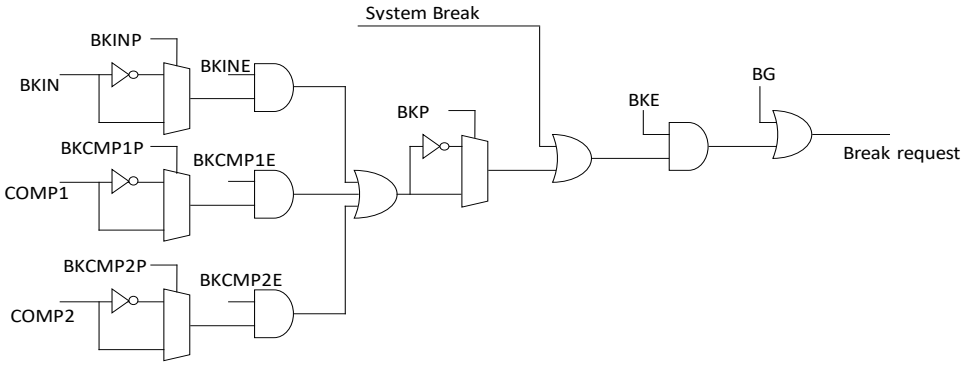
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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

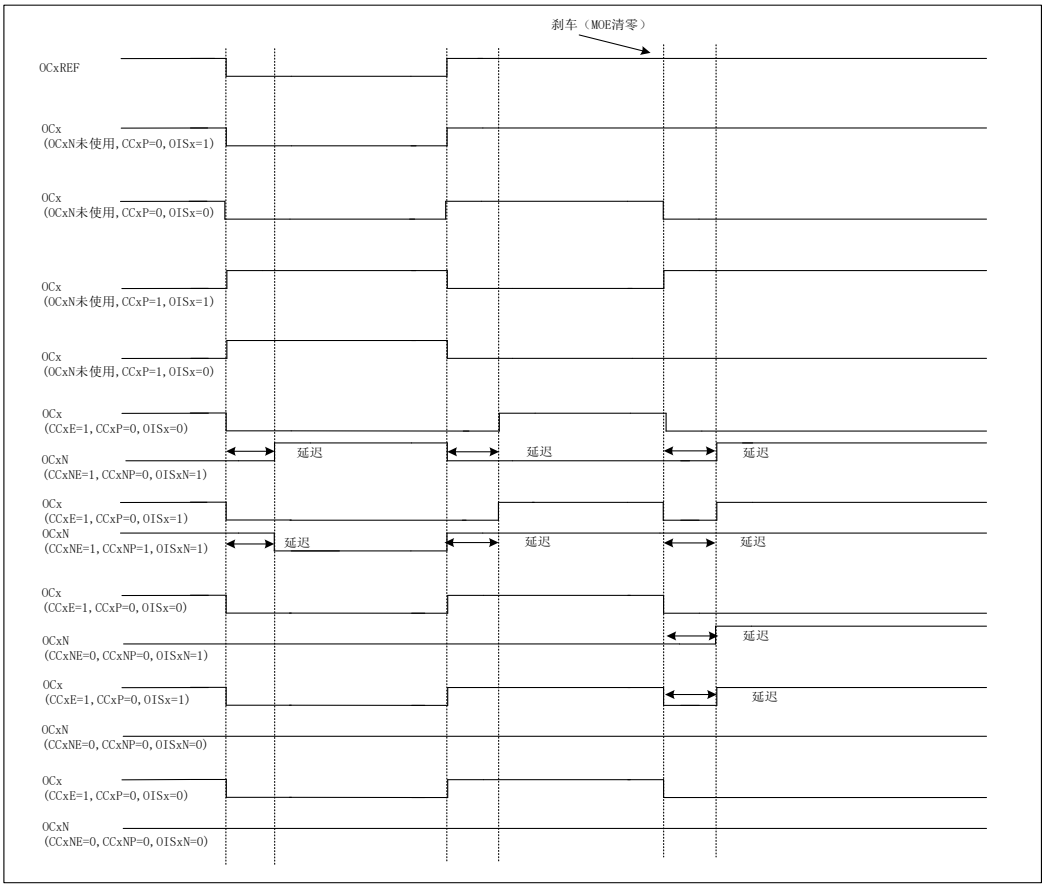
器释放使能输出，否则使能输出始终为高。

- 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_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 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：

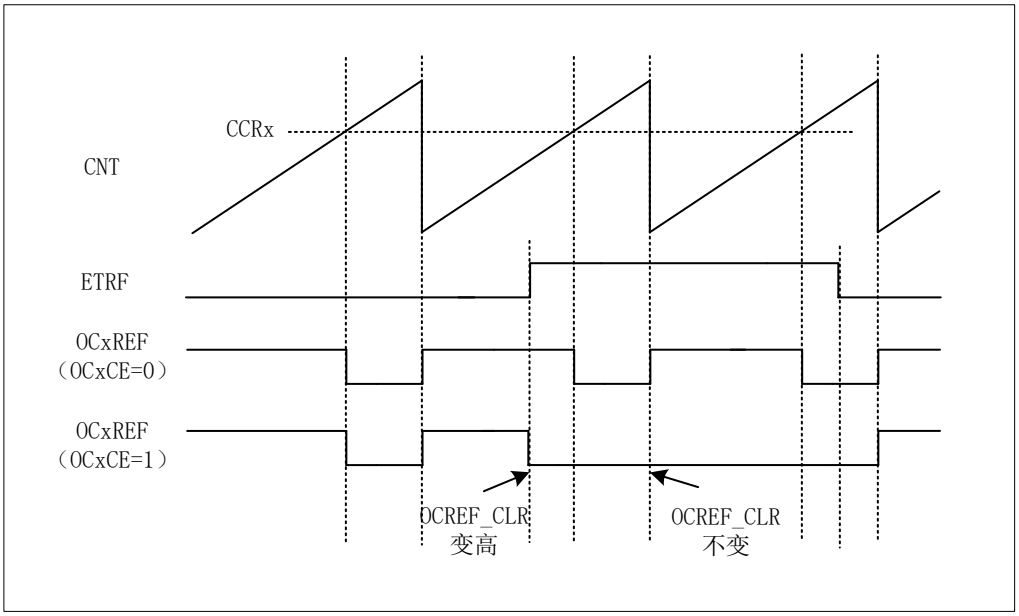
1）外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。

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



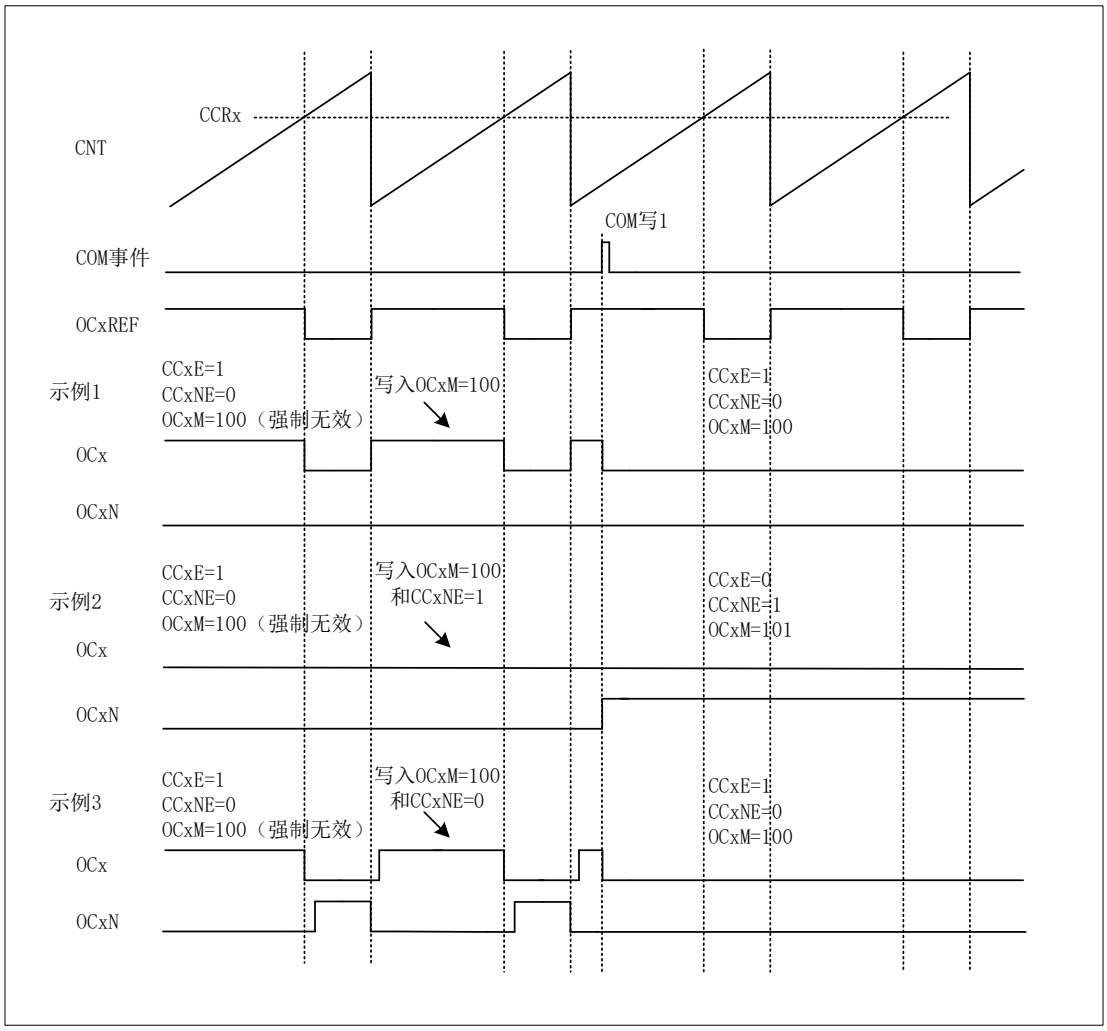
12.5.7.9. 产生六步 PWM 输出

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)



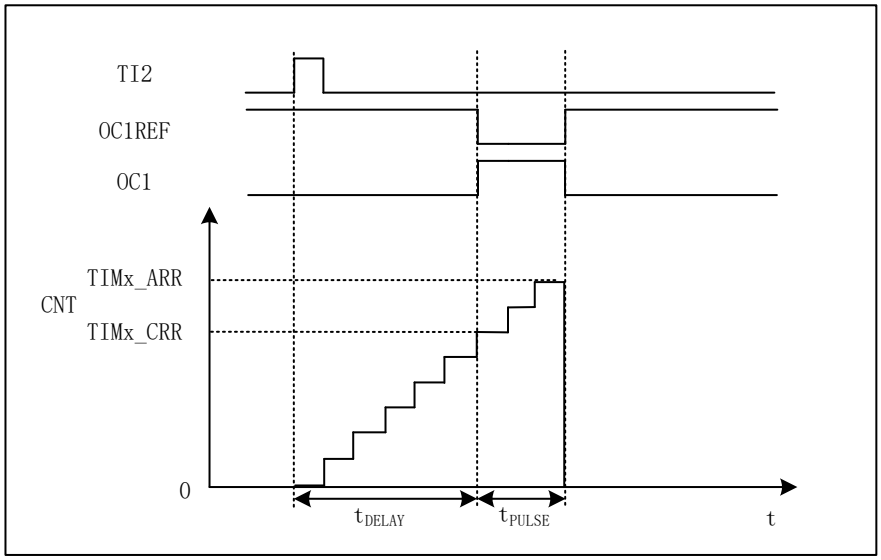
12.5.7.10. 单脉冲模式

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。

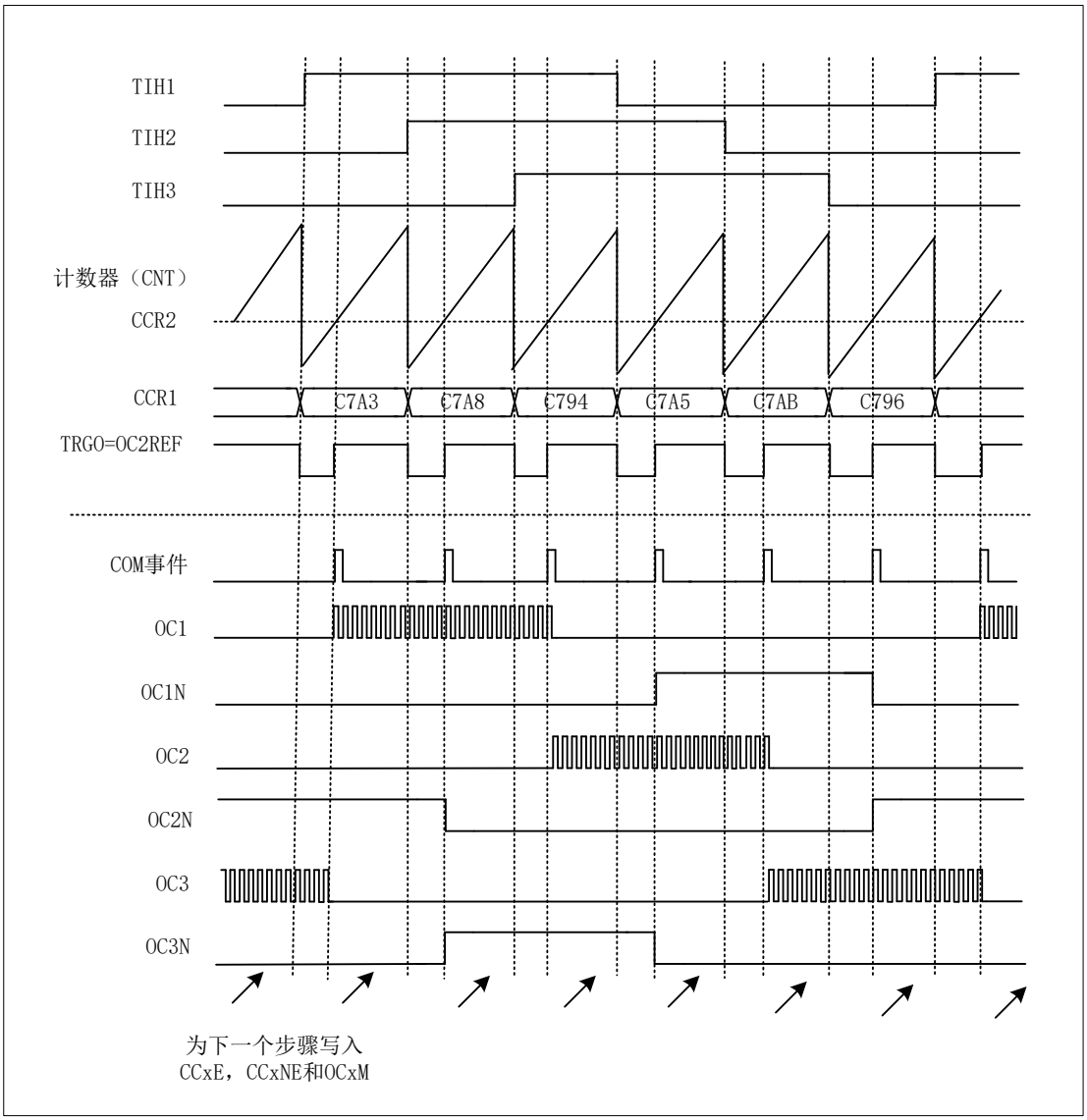
3）设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。

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：

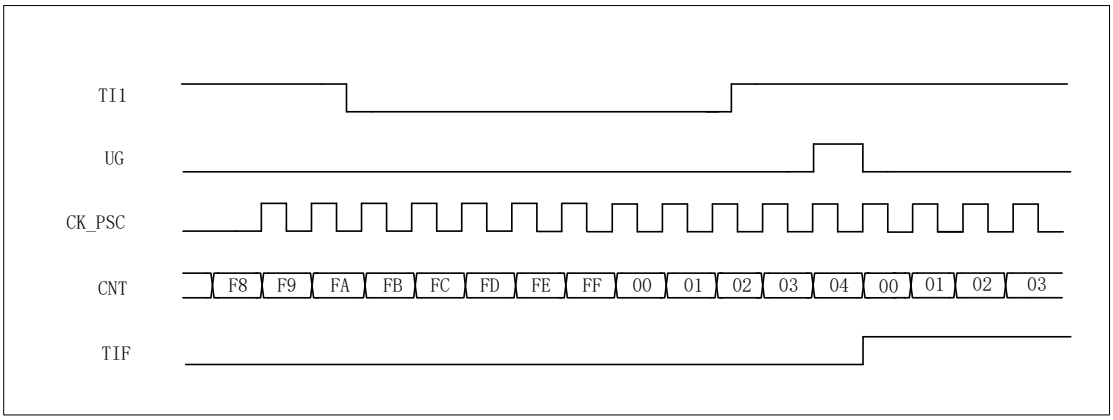
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

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。

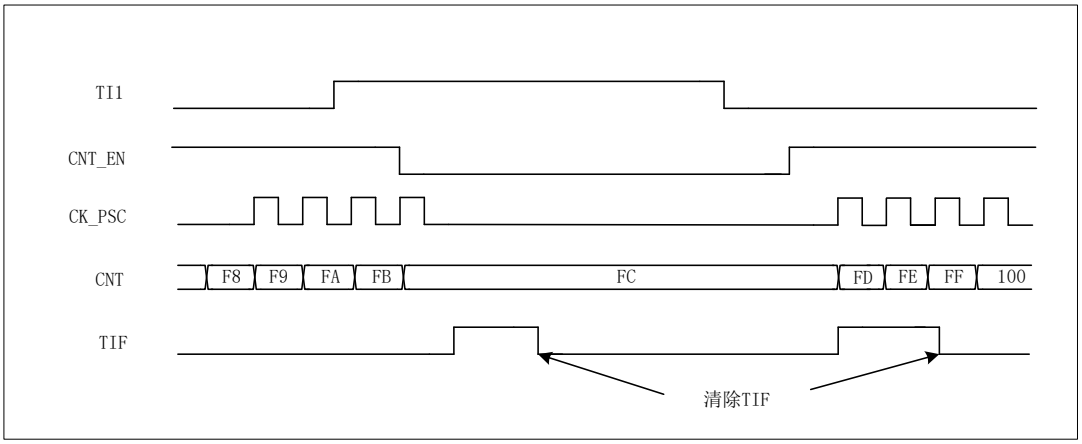
在如下的例子中，计数器只在 TI1 为低时向上计数：

- 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。

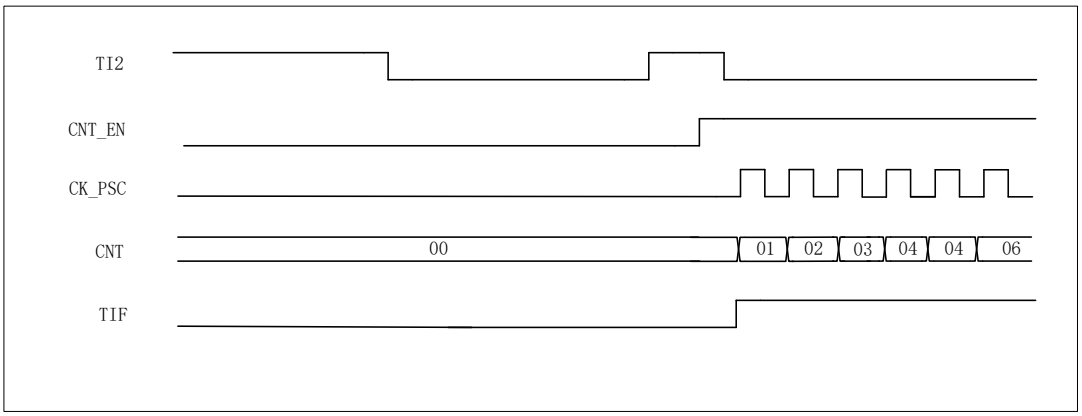
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 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： ⎯ 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

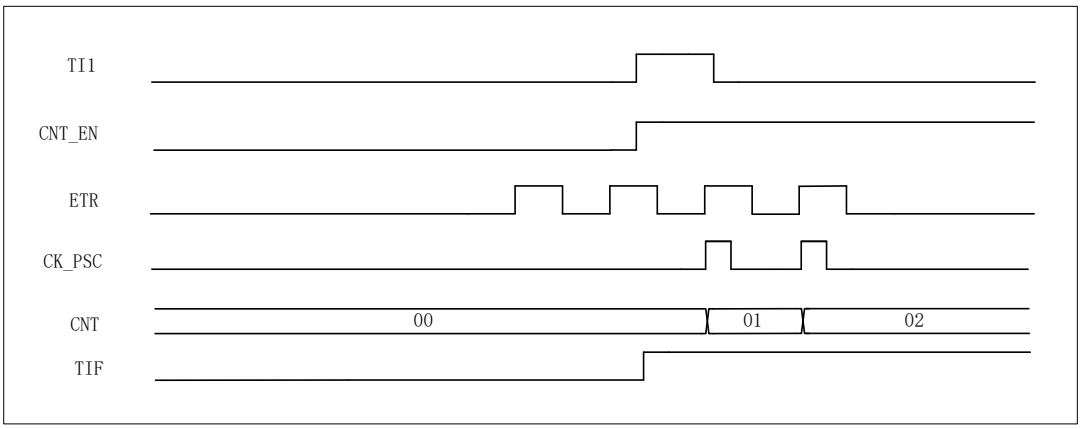
需要配置 — 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源 — 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

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 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/TIM8 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

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/TIM3/TIM4）

13.1. 概述

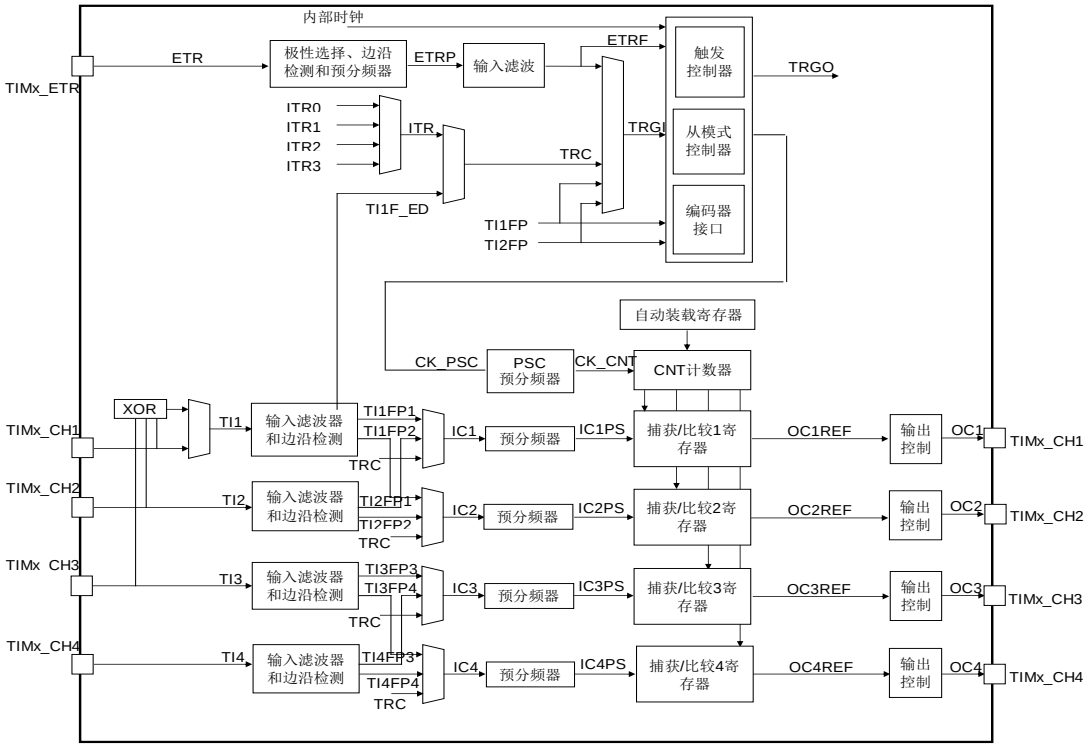
通用定时器（TIM2/TIM3/TIM4）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM 等)。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

13.2. 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	-	-	-

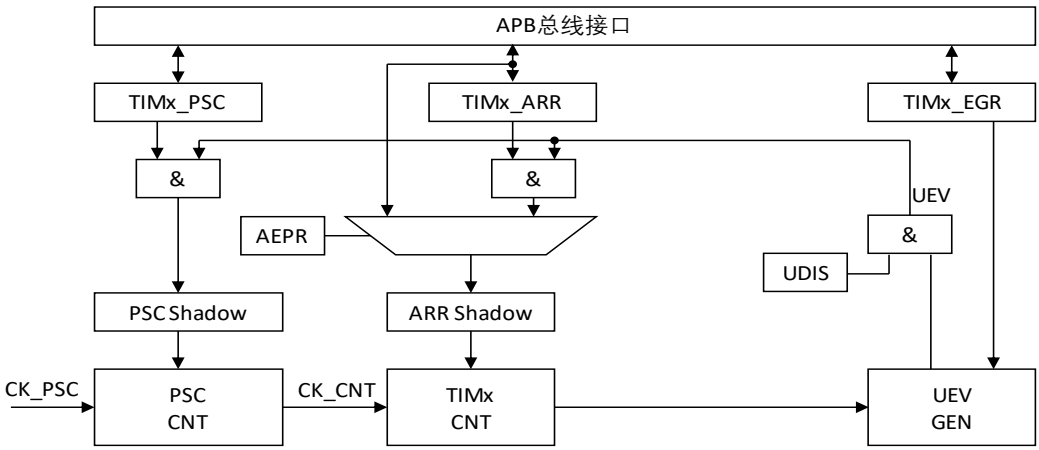
13.5. 功能描述

13.5.1. 计数单元

可编程通用定时器的主要部分是一个 16 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

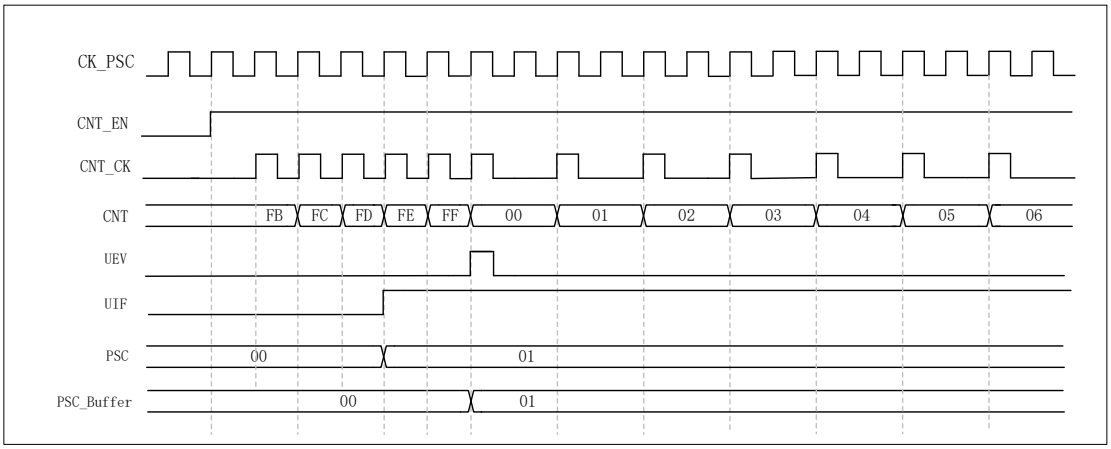
在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。向上计数模式中，定时器模块在计数器计数到自动加载值-1 产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMx_CR1 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。计数方向被硬件自动更新。

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

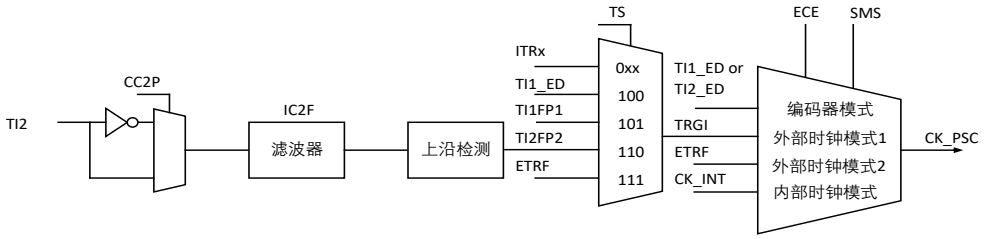
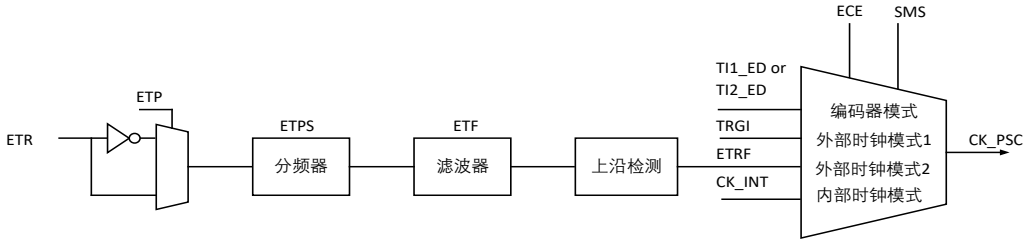


图 13-5 外部时钟模式 2



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。

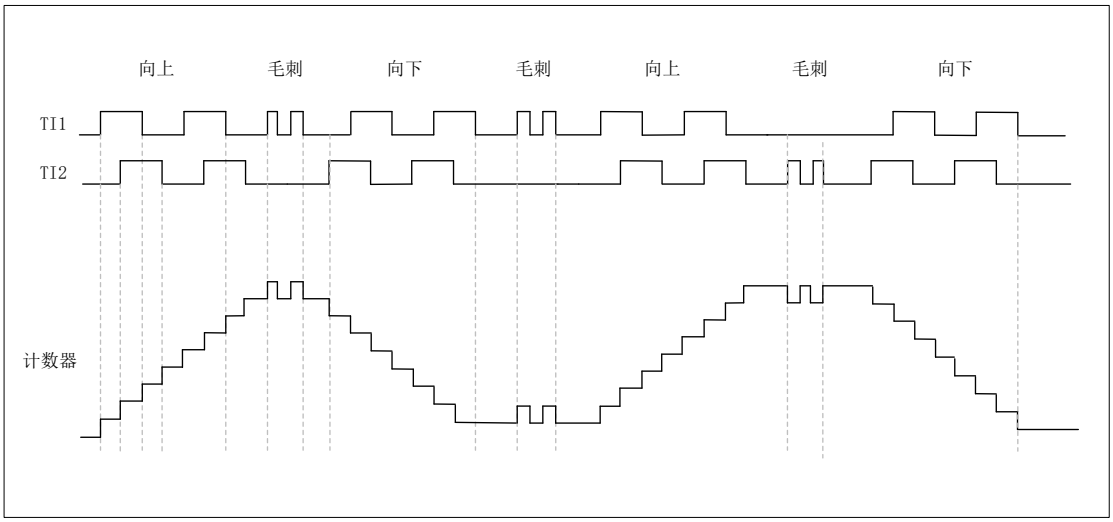
表 13-8 计数器方向和编码器信号的关系

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

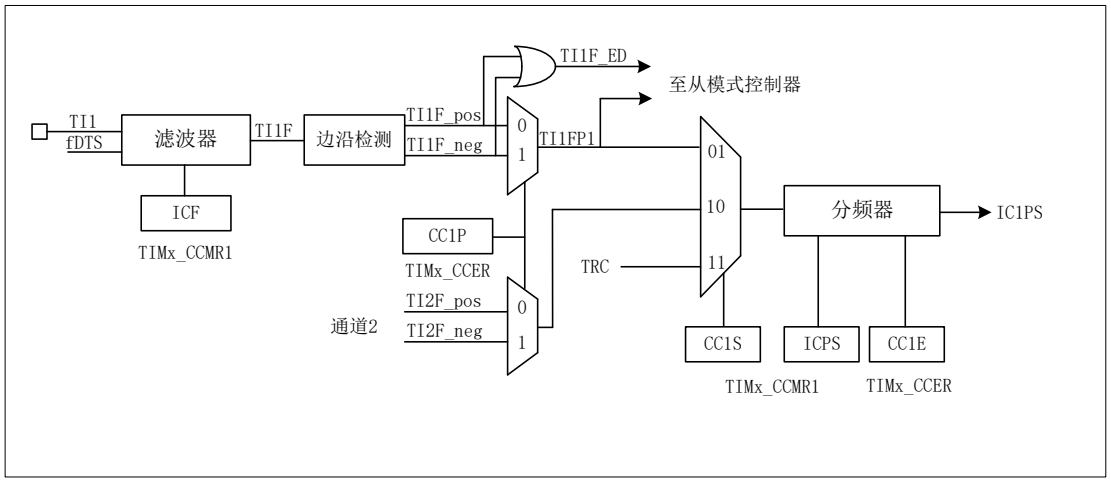


13.5.5. 捕获比较通道

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

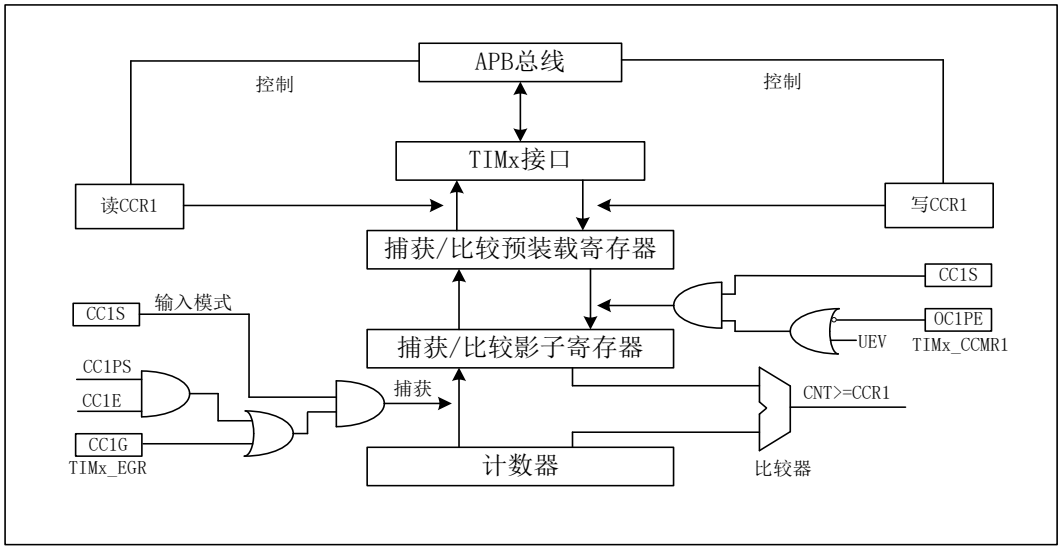
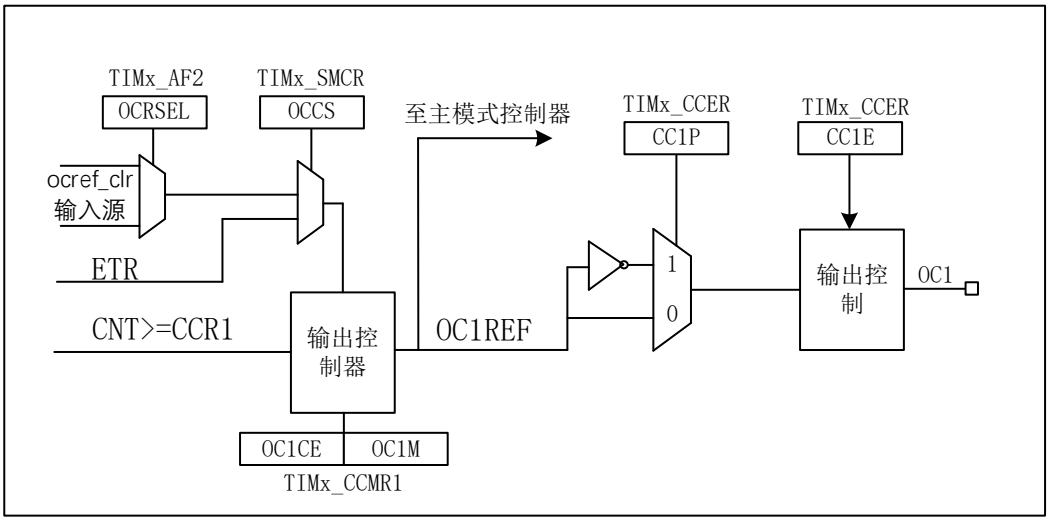


图 13-9 捕获/比较通道的输出部分(通道 1)



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_CCR1 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)。

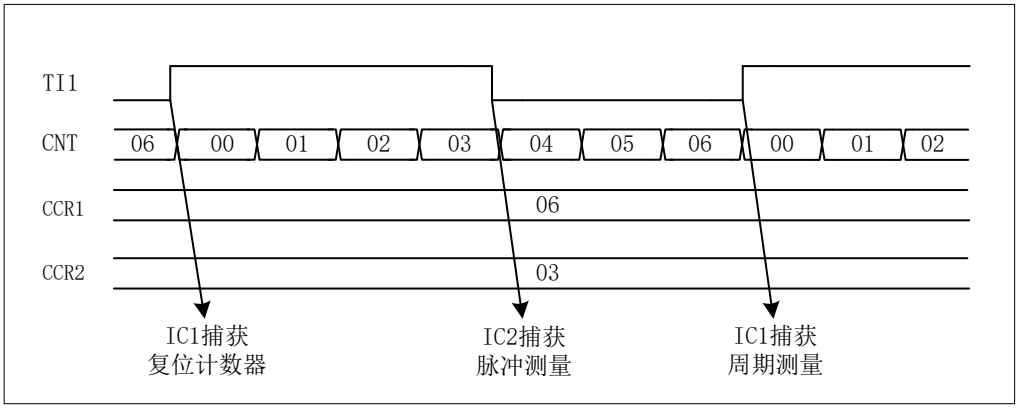
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

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

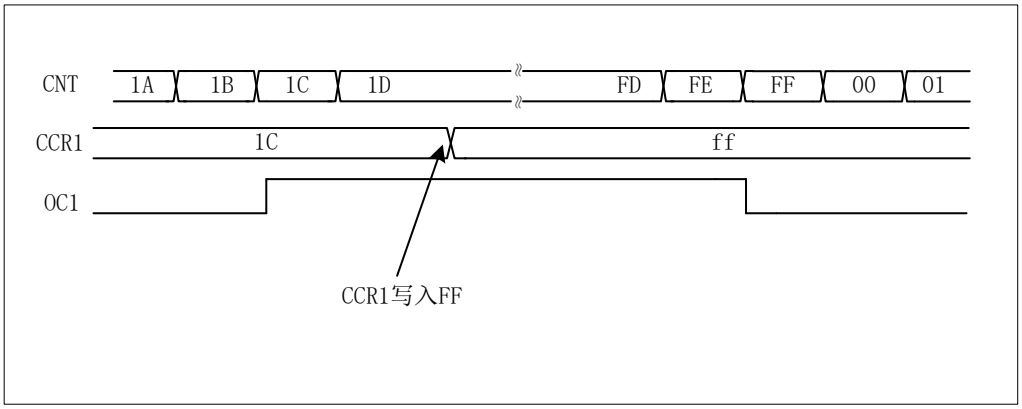
c）置 CCxP = 0 选择极性为高电平有效

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。

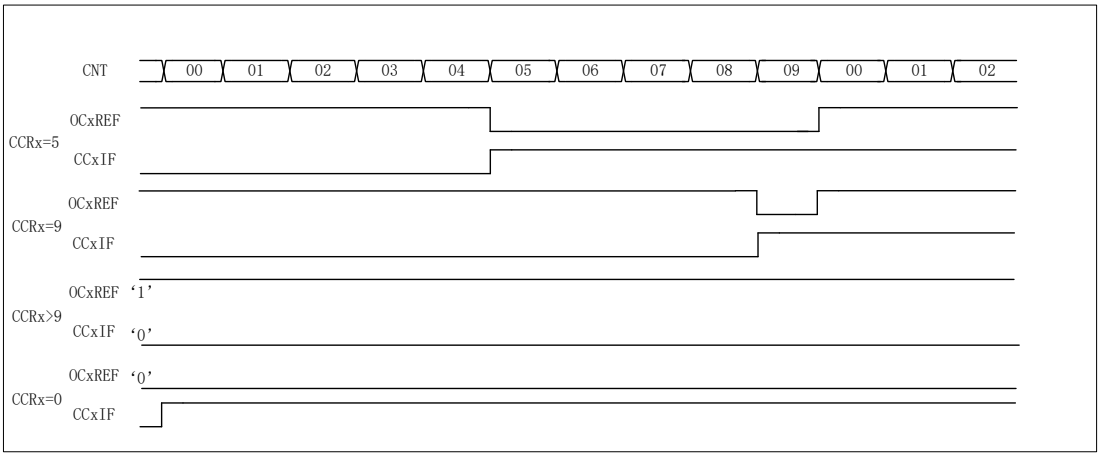
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。 根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

■ 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。

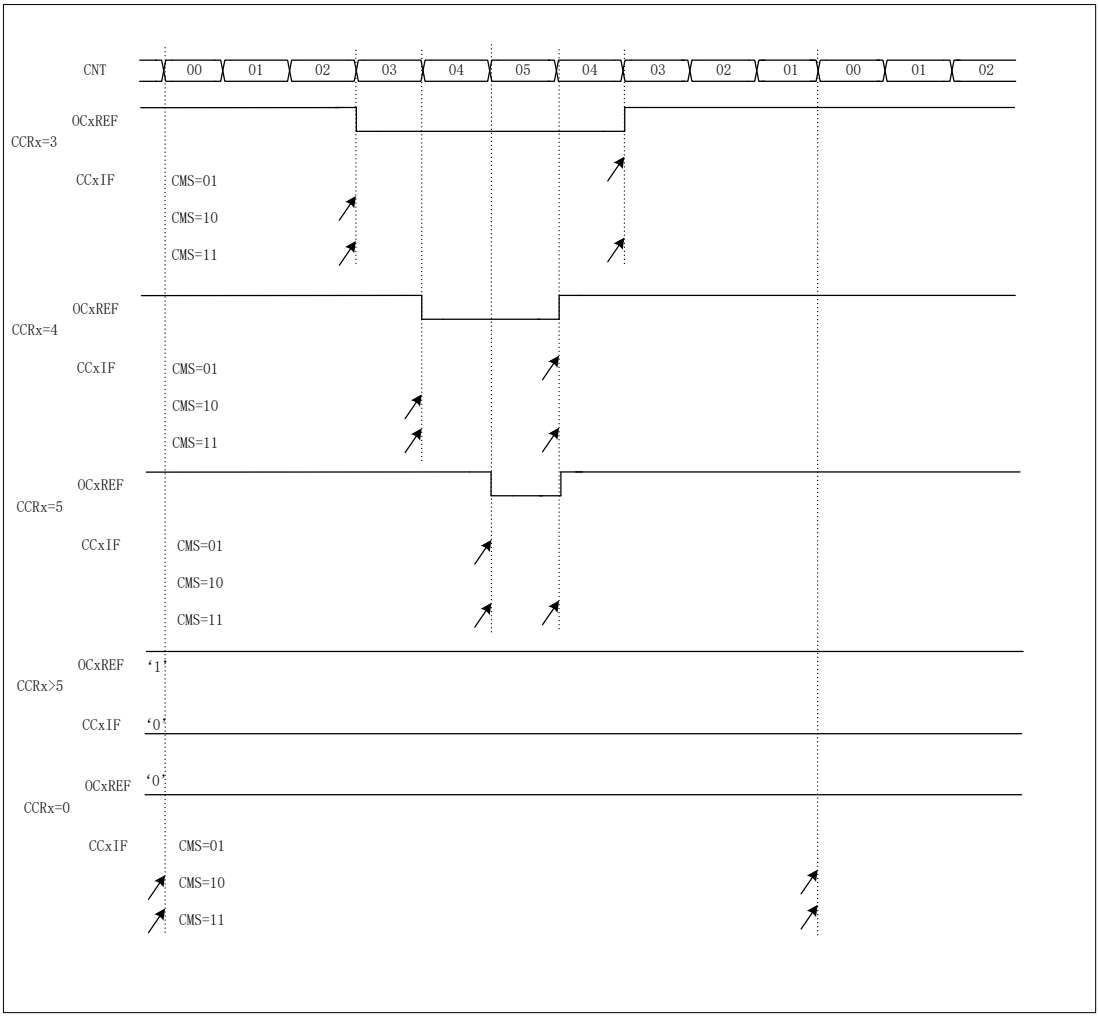
下图给出了一些中央对齐的 PWM 波形的例子

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。

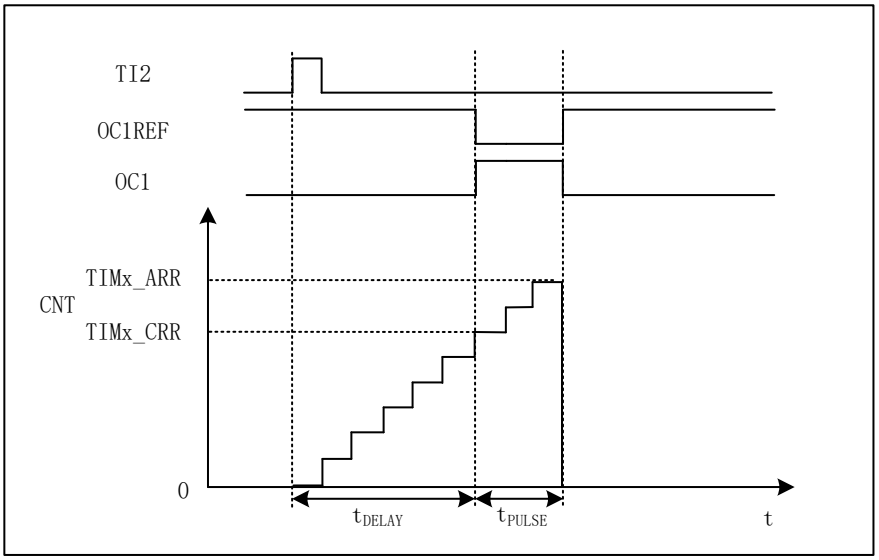
13.5.5.6. 单脉冲模式

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。

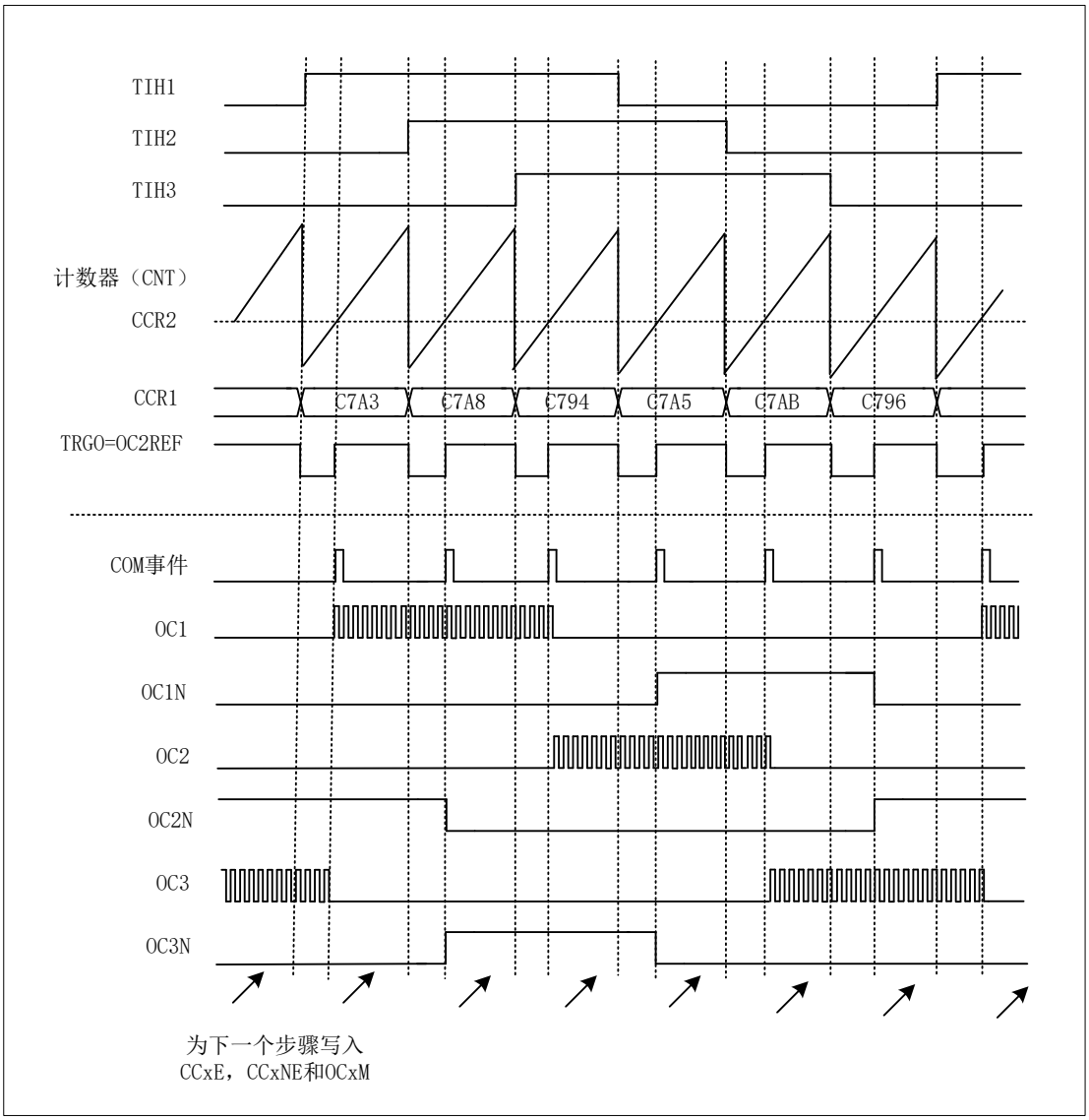
3）设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。

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：

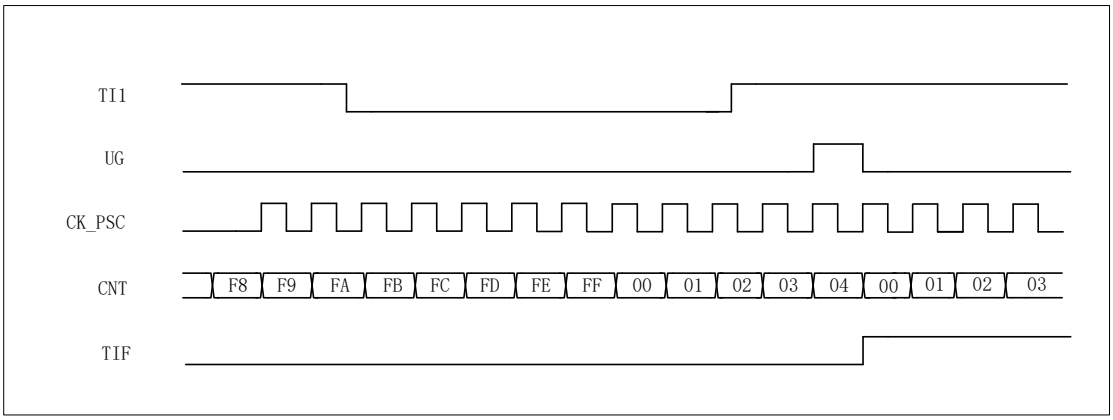
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

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)。

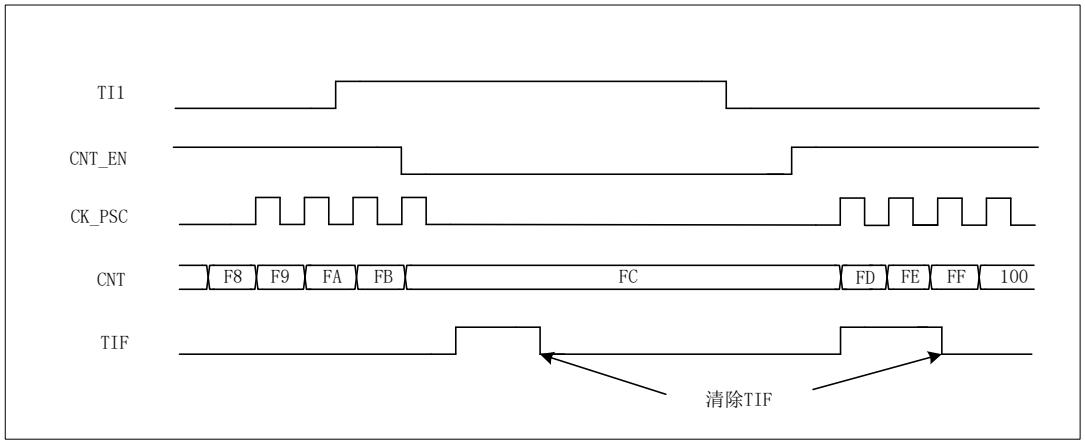
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

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：

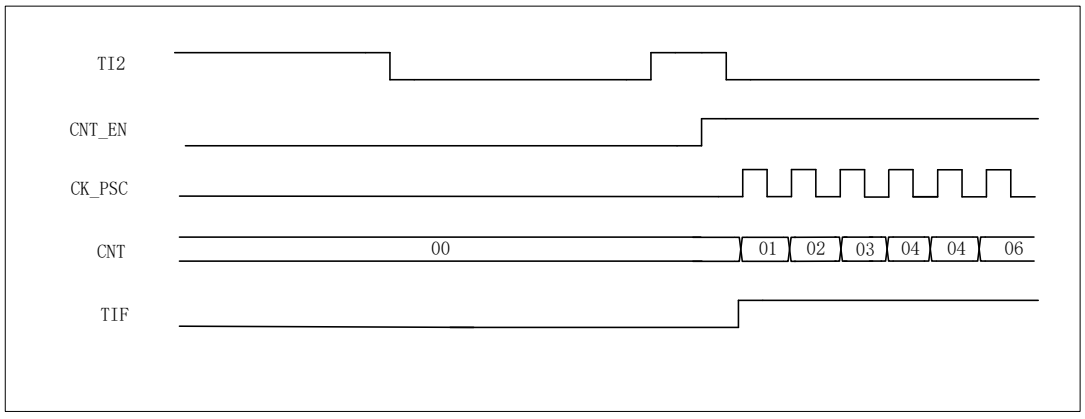
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

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

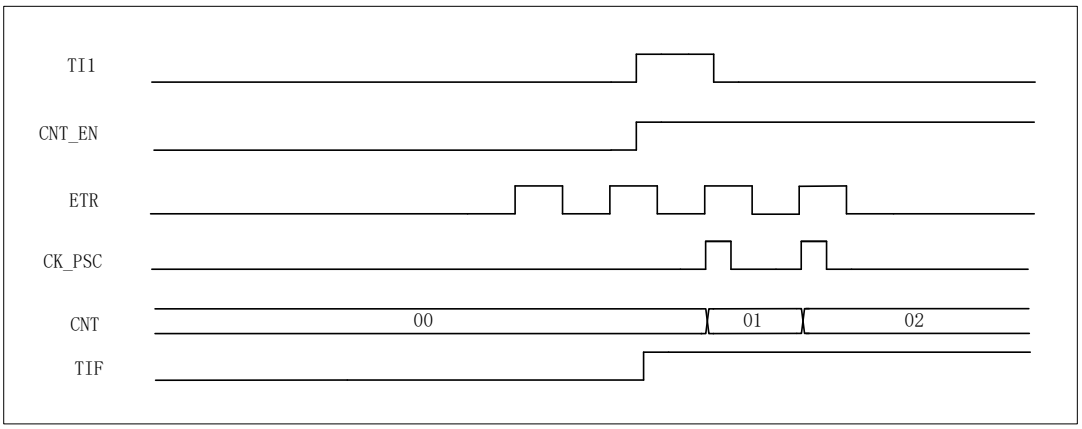
需要配置 — 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源 — 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

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/TIM3/TIM4 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

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)

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
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。

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
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。

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
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。

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
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。

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/TIM7）

14.1. 概述

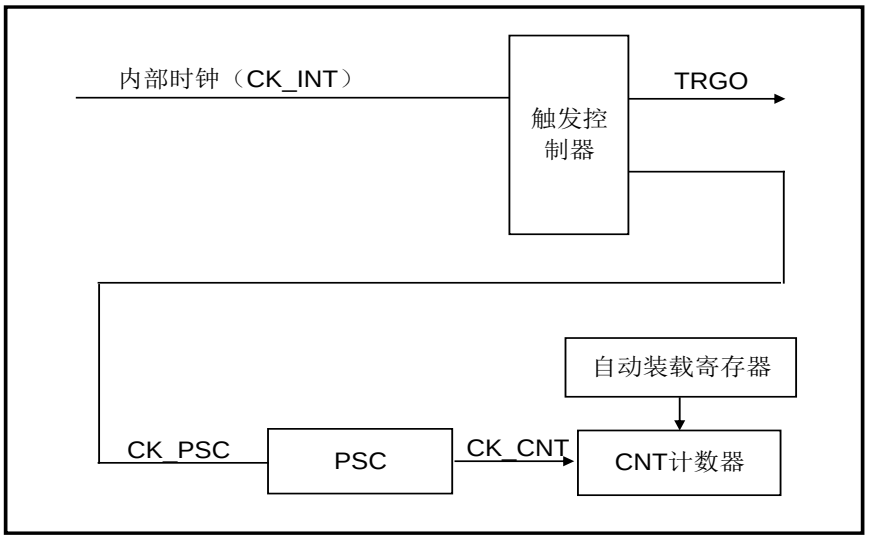
基本定时器（TIM6/TIM7）包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。它们可以作为通用定时器提供时间基准。

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



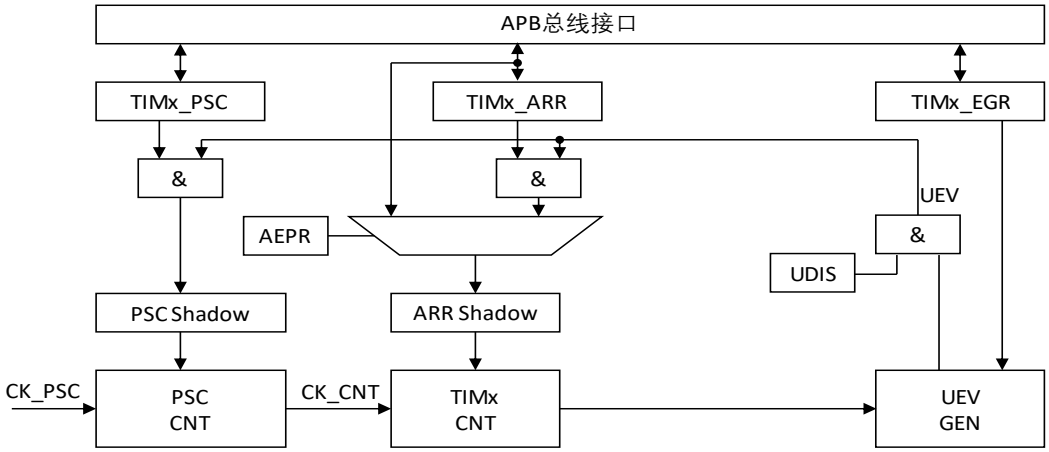
14.4. 功能描述

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

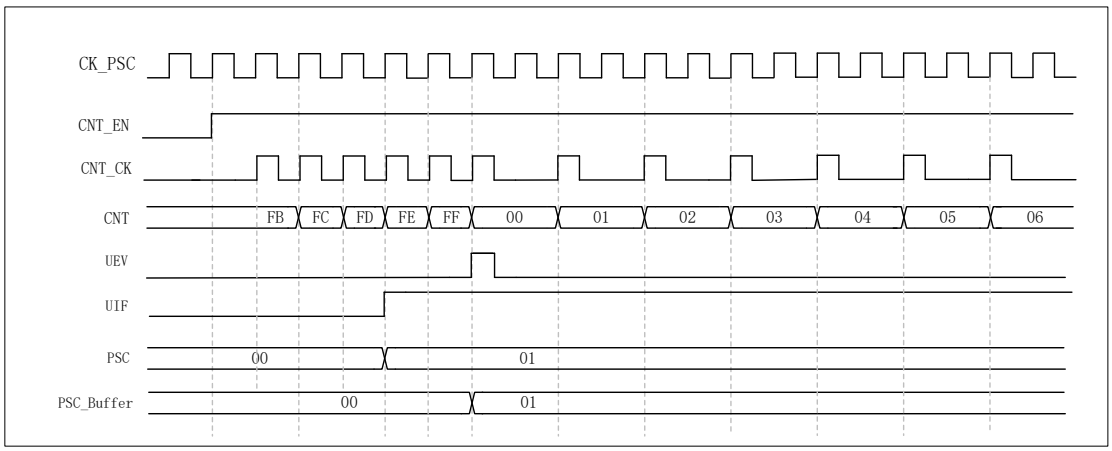
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

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/TIM7 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

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（TIM15）

15.1. 概述

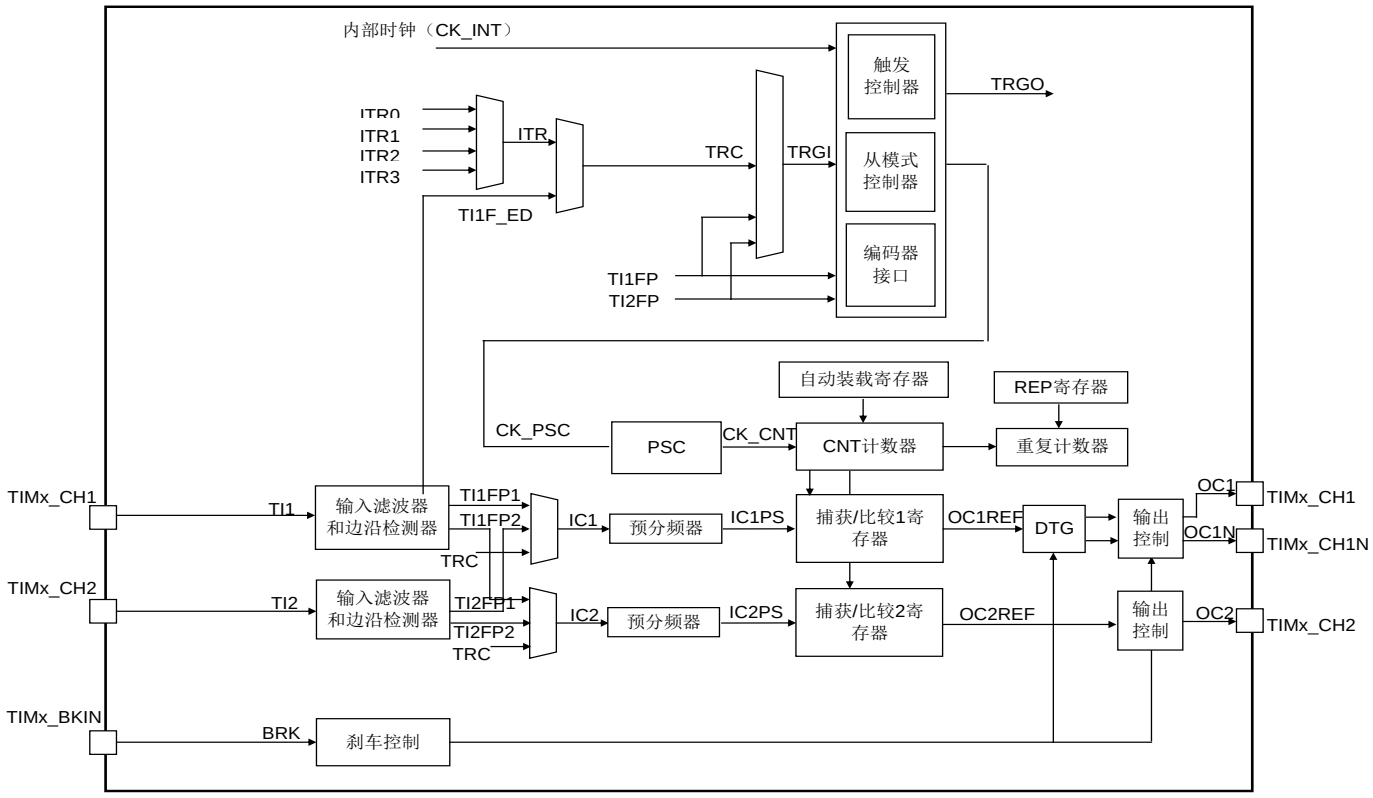
通用定时器由（TIM15）一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。 使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

15.2. 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

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）

	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 15-2 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 15-3 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 15-4 TIM1 OCREF_CLR 输入

OCRSEL[2:0]	TIM15 源
0000	comp1_out
0001	comp2_out
0010	comp3_out
0011	comp4_out
保留	-

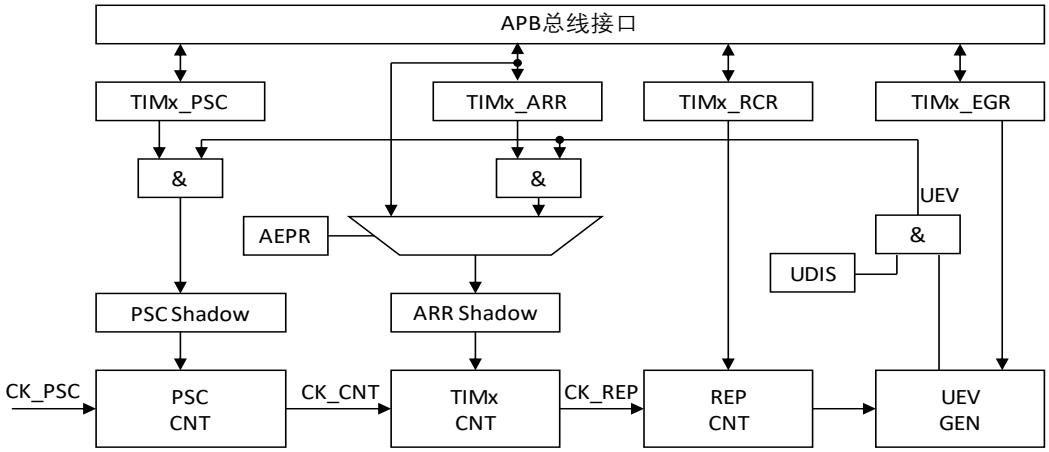
15.5. 功能描述

15.5.1. 计数单元

可编程定时器 TIM15 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 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 TIM15 支持一种计数模式：

- 向上计数模式

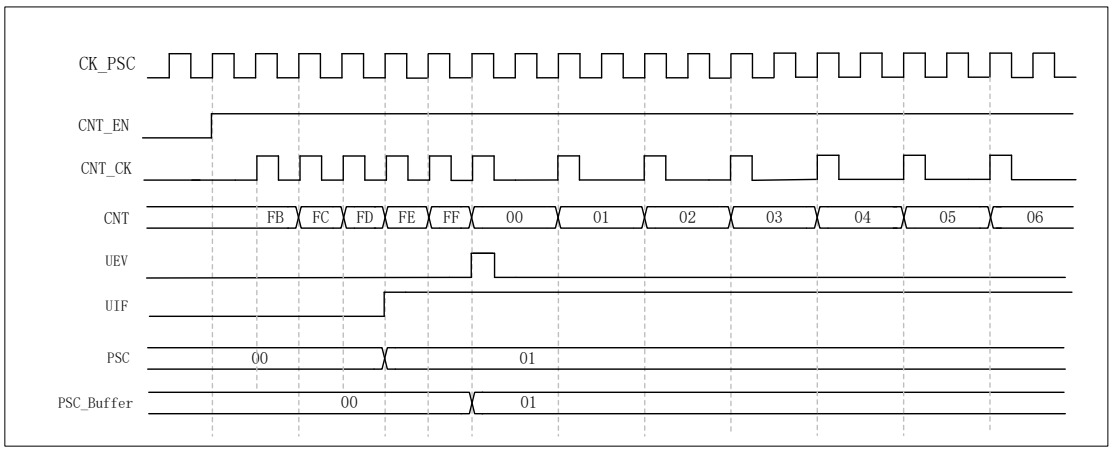
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

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



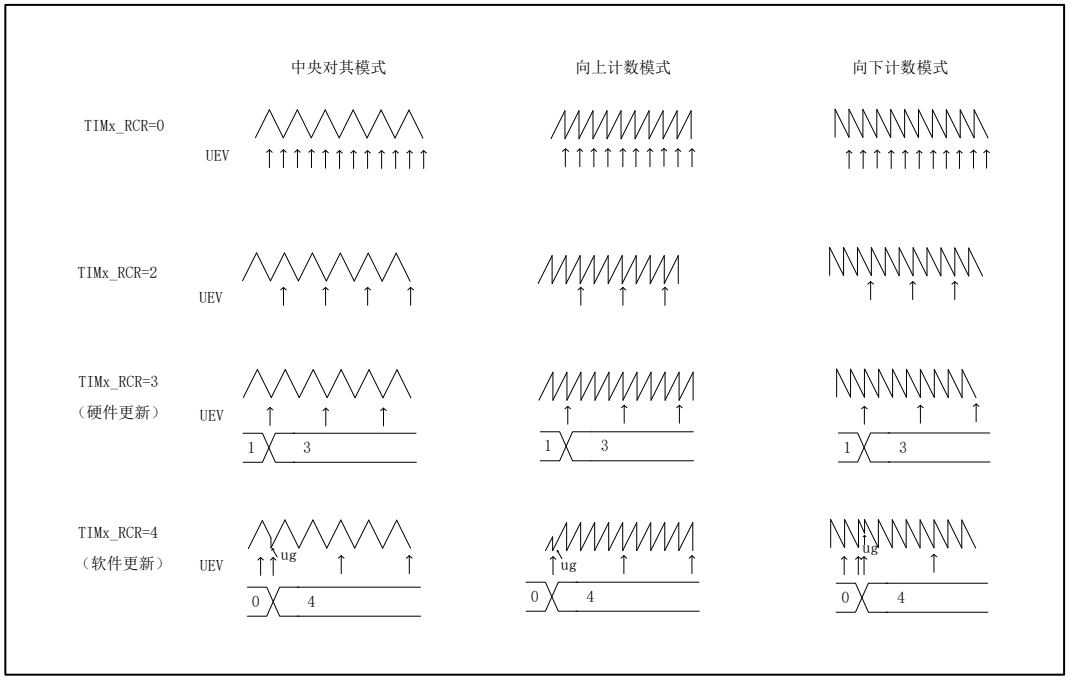
15.5.3. 重复计数器

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 15-4 重复计数器溢出事件



15.5.4. 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 15-5 计数器方向和编码器信号的关系

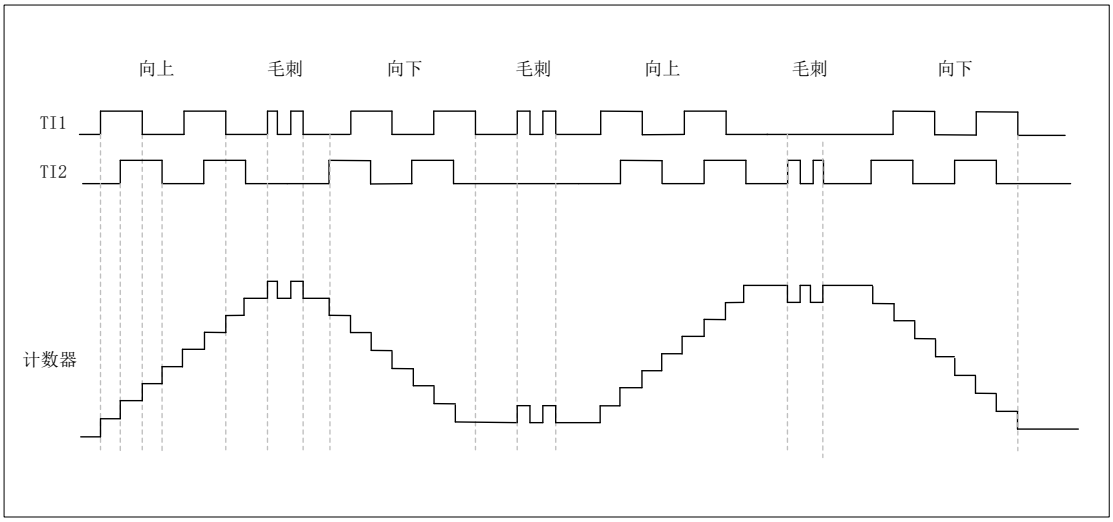
有效边沿	相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1）	TI1FP1		TI2FP2	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
TI1 和 TI2 都计数	高	向下计数	向上计数	向上计数	向下计数

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 15-5 编码器模式下计数器操作实例



15.5.5. 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 15-6 时钟源

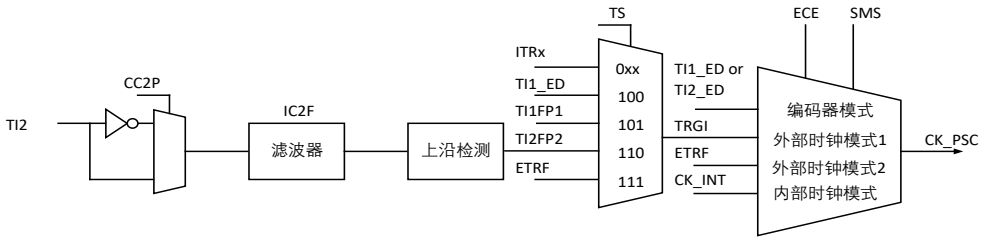


图 15-7 外部时钟模式 1

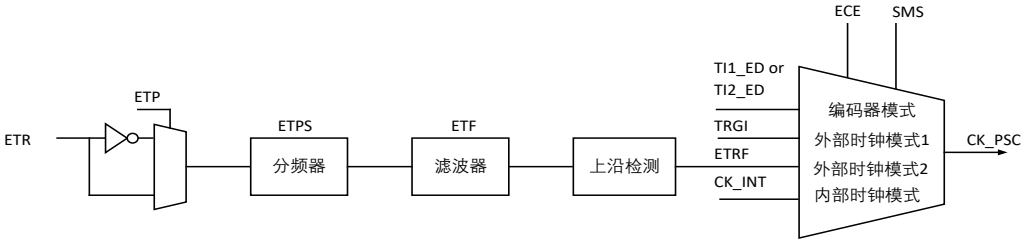


图 15-8 外部时钟模式 2

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：

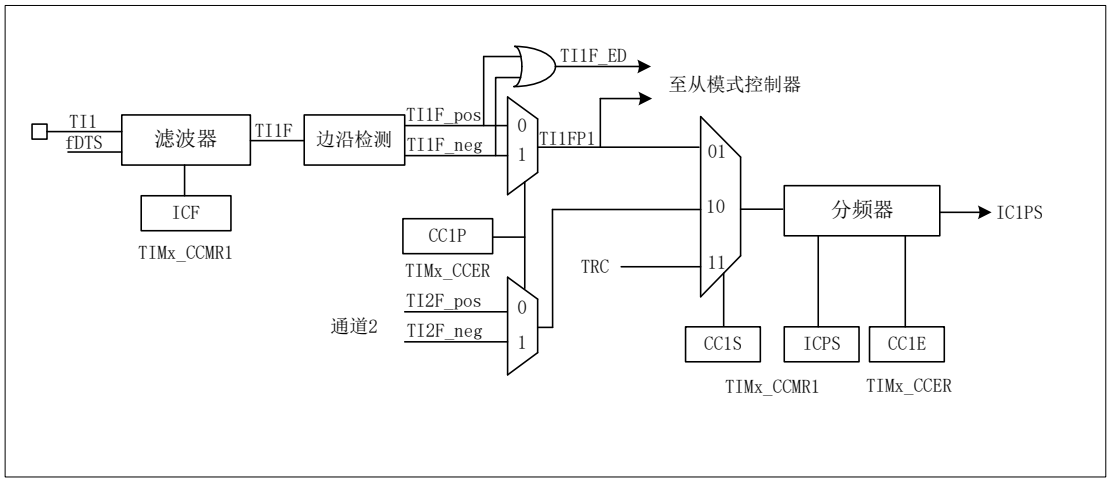
- 内部时钟(CK_INT) ：SMS=000，ECE=0，禁止从模式。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。
- 外部时钟模式 1：SMS=111，ECE=0，CK_PSC 由 TRGI 产生，TRGI 有八个信号源，并由 TIMx_SMCR.TS 寄存器选择。
  - TS=000，内部触发 0（ITR0）
  - TS=001，内部触发 0（ITR1）
  - TS=010，内部触发 0（ITR2）
  - TS=011，内部触发 0（ITR3）
  - TS=100，TI1 的边沿检测器（TI1F_ED）
  - TS=101，滤波后的定时器输入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

15.5.6. 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 15-9 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 15-10 捕获/比较通道 1 的主电路

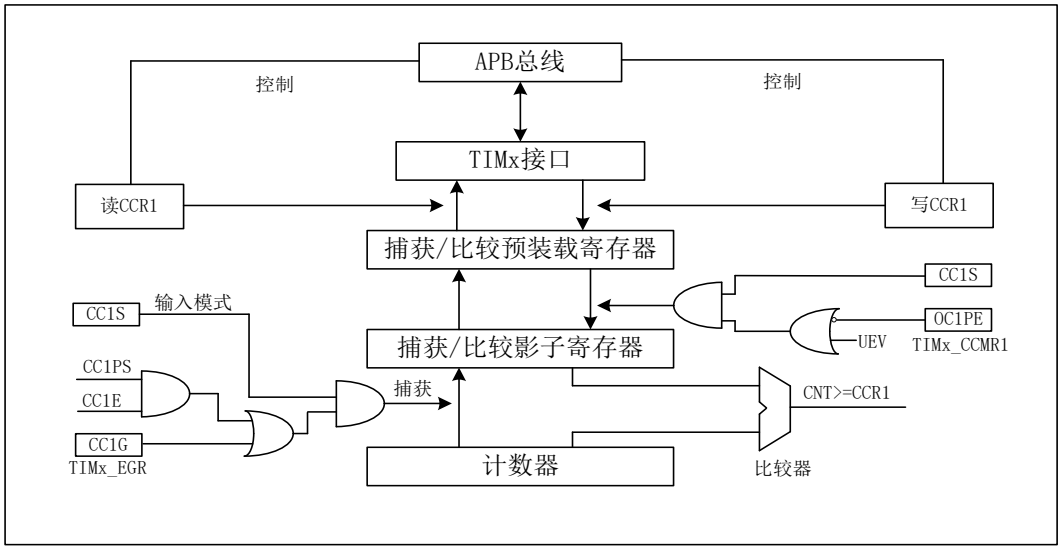


图 15-11 捕获/比较通道的输出部分(通道 1)

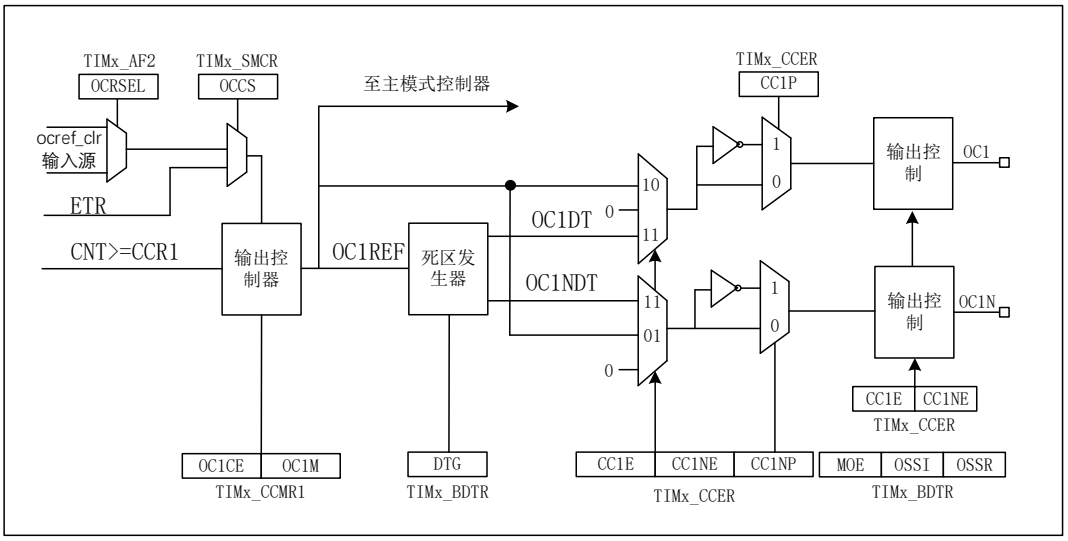
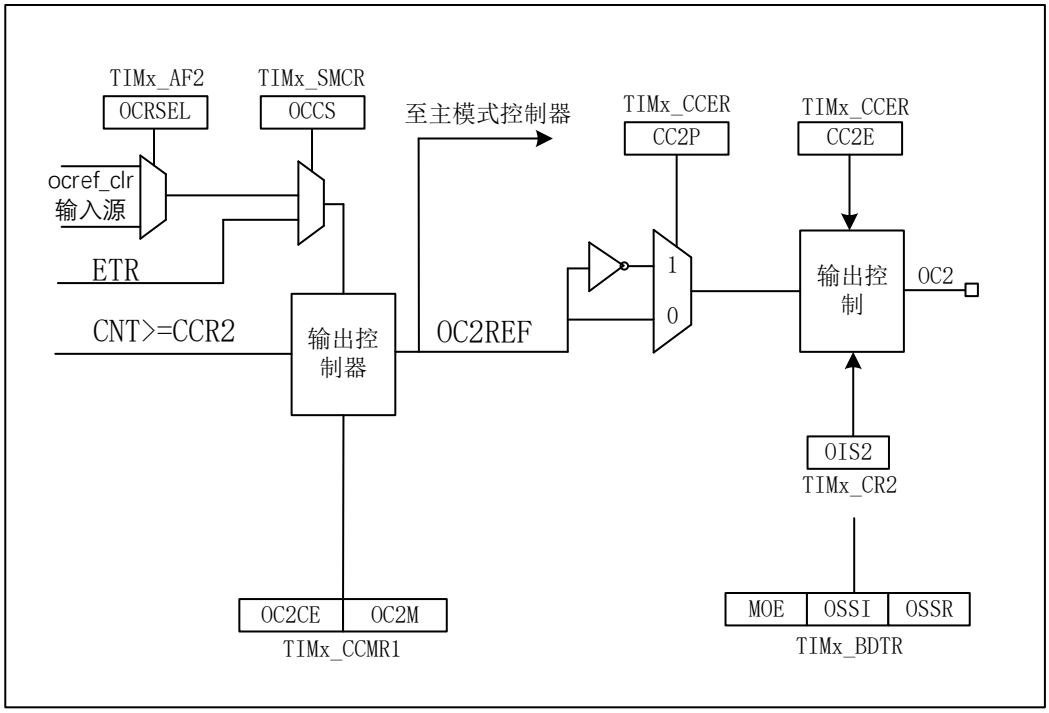


图 15-12 捕获/比较通道的输出部分(通道 2)



15.5.6.1. 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_CCR1 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。

15.5.6.2. 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)。

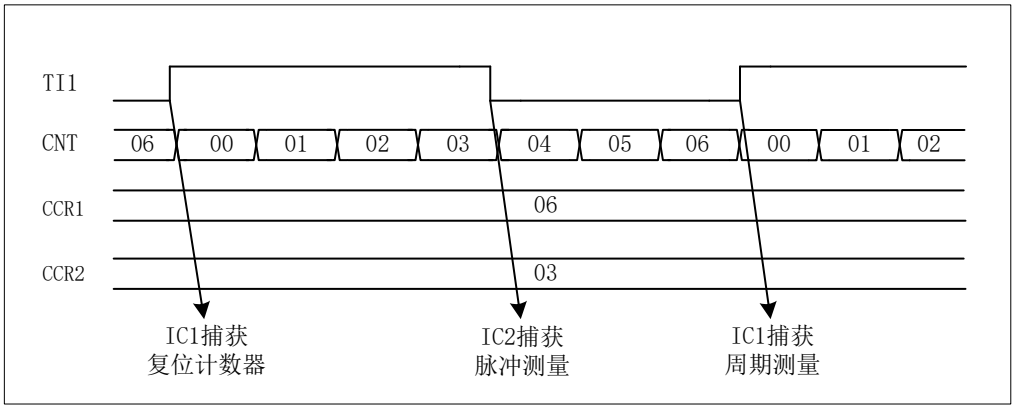
7）选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。

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 15-13 PWM 输入模式时序



15.5.6.3. 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。

15.5.6.4. 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

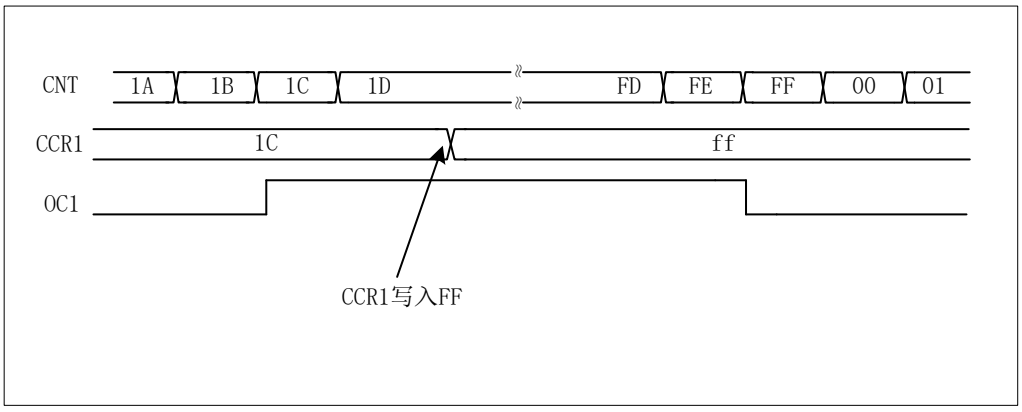
c）置 CCxP = 0 选择极性为高电平有效

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 15-14 输出比较模式，翻转 OC1



15.5.6.5. 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。

15.5.6.6. 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 15-15 带死区插入的互补输出

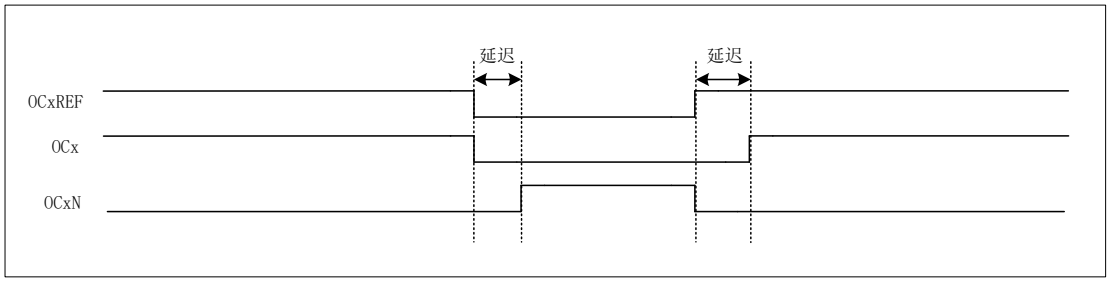


图 15-16 死区波形延迟大于负脉冲

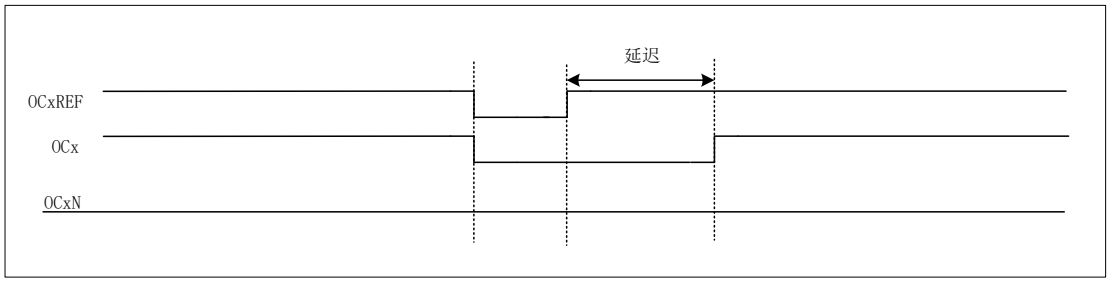
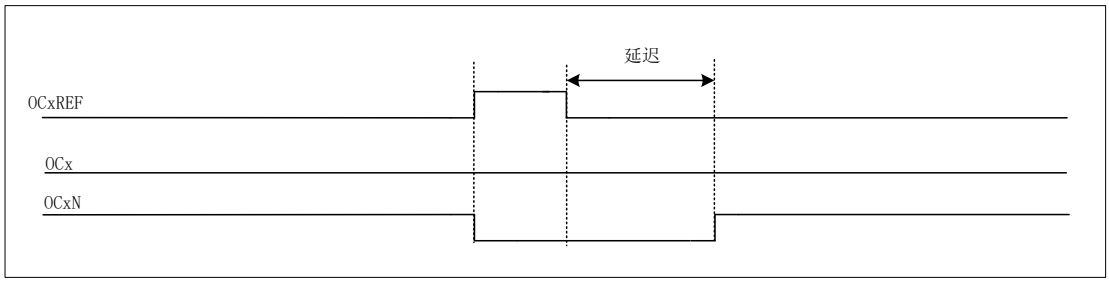


图 15-17 死区波形延迟大于正脉冲



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

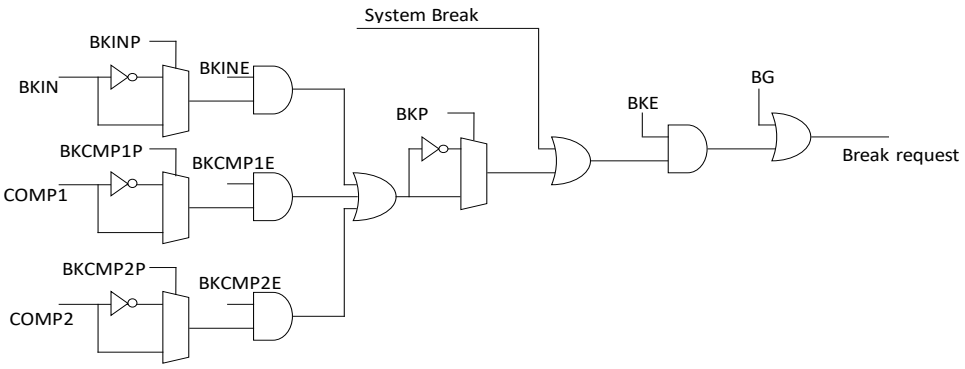
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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。

15.5.6.7. 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 15-18 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。

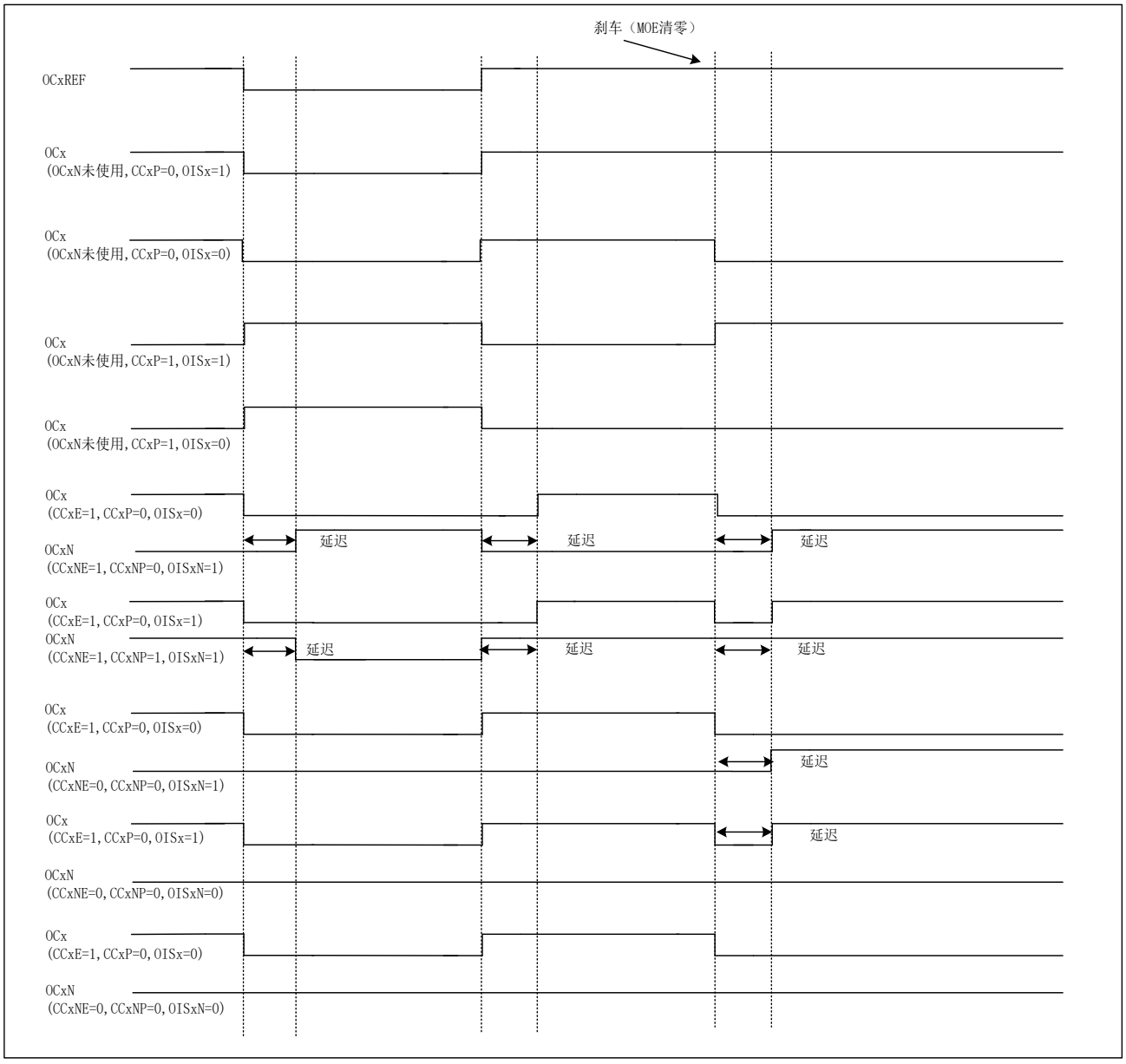
当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- 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_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 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 15-19 响应刹车的输出



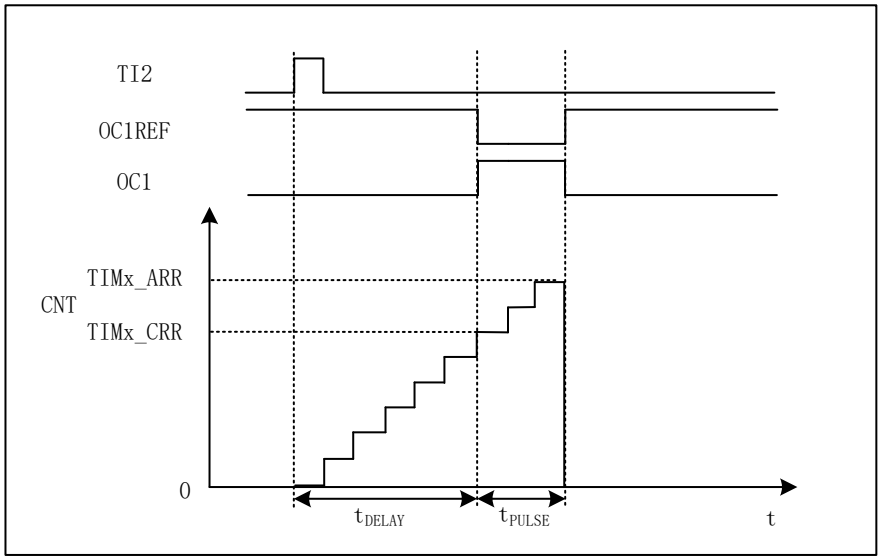
15.5.6.8. 单脉冲模式

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 15-20 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。

15.5.7. 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。

15.5.7.1. 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：

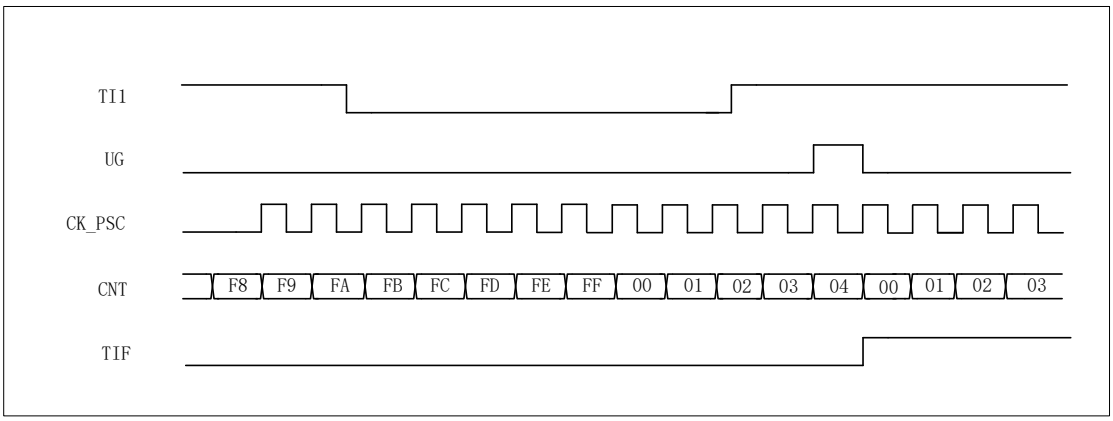
1）配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

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 15-21 复位模式下的控制电路



15.5.7.2. 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)。

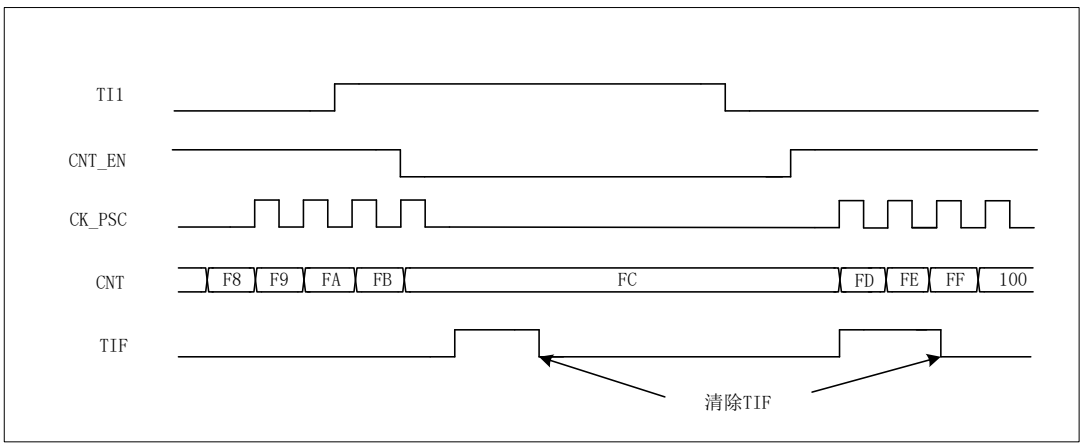
2）置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

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 15-22 门控模式下的控制电路



15.5.7.3. 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：

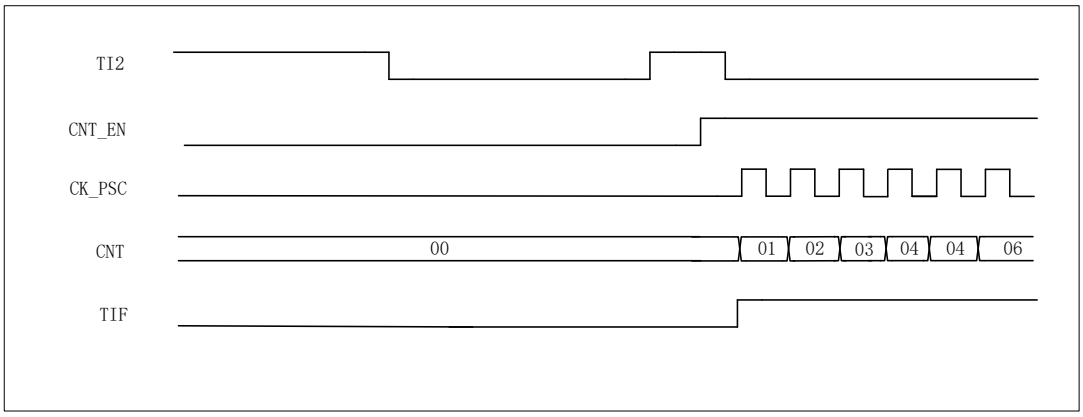
1）配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

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 15-23 触发模式下的控制电路



15.5.8. 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。

15.5.9. 定时器调试模式

定时器在调试时依然在运行。

15.6. TIM15 寄存器描述

15.6.1. 寄存器列表

TIM15 寄存器基地址：0x40014000

表 15-6 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

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，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。

15.6.3. 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。

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
31:8	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 15-7 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

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，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。

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	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。

15.6.7. 事件产生寄存器(TIMx_EGR: 14h)

位域	名称	属性	复位值	描述
31:8	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。

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。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)。

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	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 15-8 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		

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。

15.6.13. 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；

15.6.14. 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。

15.6.15. 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。

15.6.16. 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。

15.6.17. 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， ...

15.6.18. 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。

15.6.19. 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

15.6.20. 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。

15.6.21. 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

15.6.22. 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；

16. 通用定时器（TIM16/TIM17）

16.1. 概述

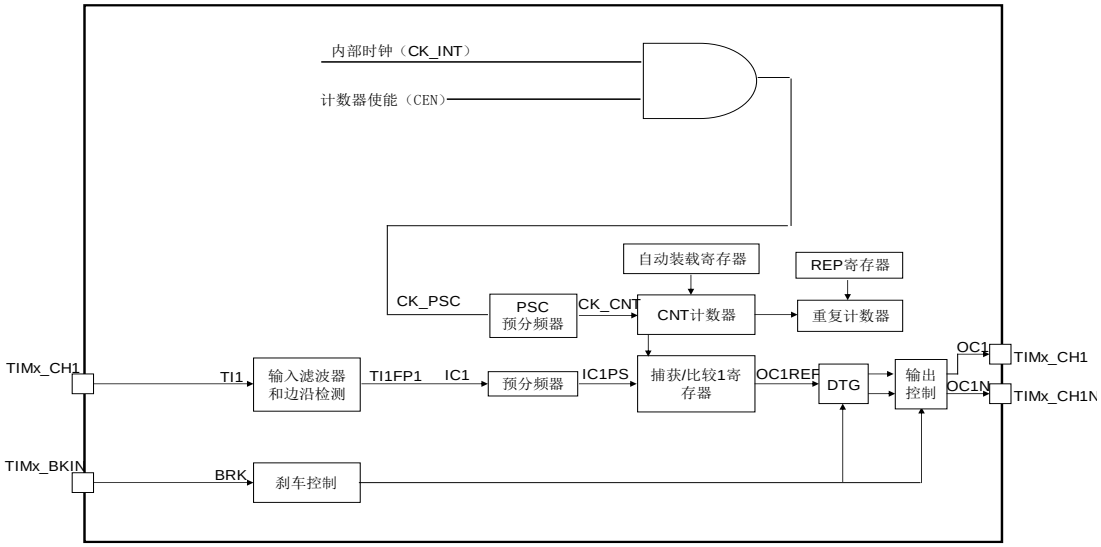
通用定时器（TIM16/TIM17）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。 它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。 使用定时器预分频器和系统时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。 高级控制定时器和通用定时器是完全独立的，它们不共享任何资源，但它们可以同步操作。

16.2. 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

16.3. 结构框图

图 16-1 通用定时器框图



16.4. TIMx 输入映射

表 16-1 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 16-2 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	-	-

16.5. 功能描述

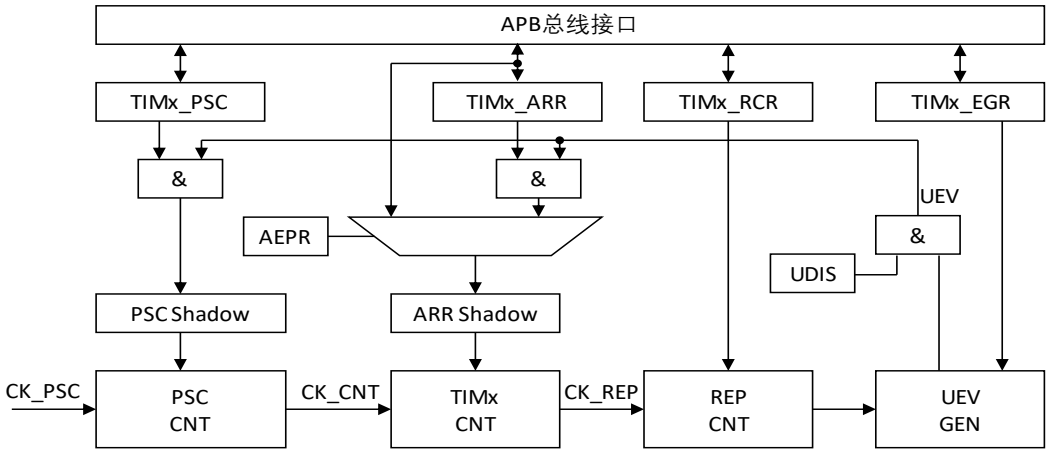
16.5.1. 计数单元

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 16-2 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

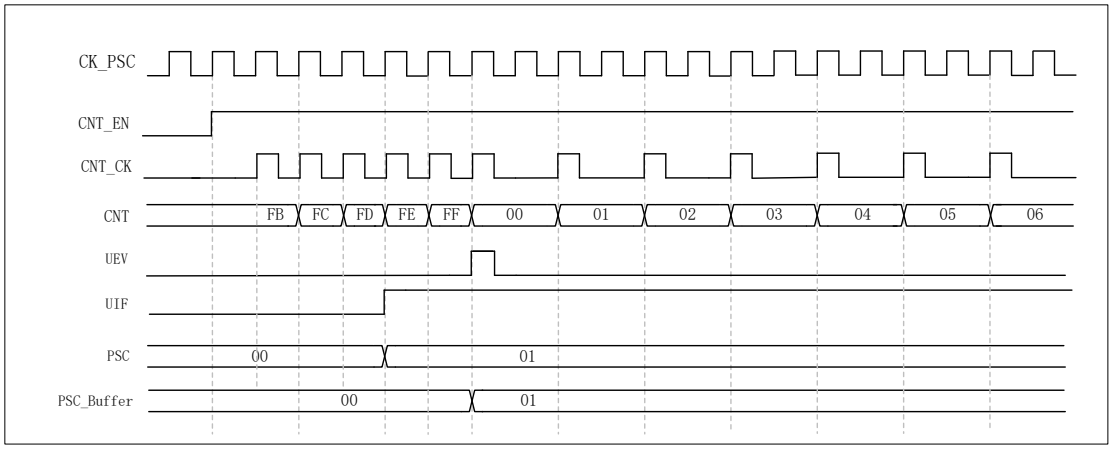
在向上计数模式中，计数器从 0 计数到自动加载值(TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。

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。

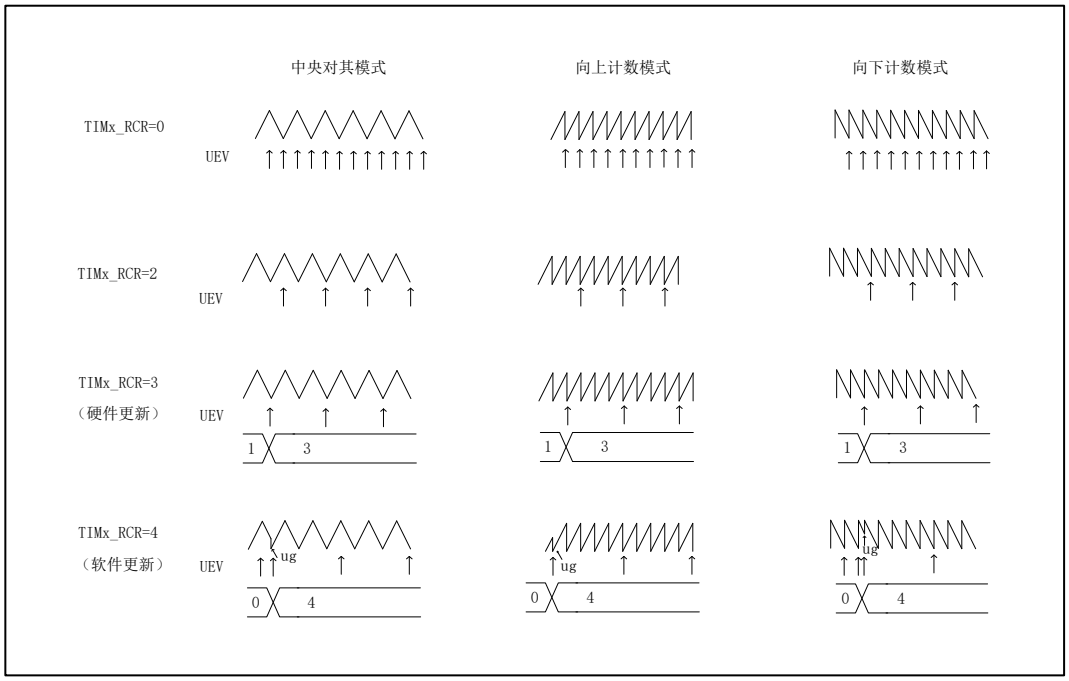
16.5.4. 重复计数器

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 16-4 重复计数器溢出事件

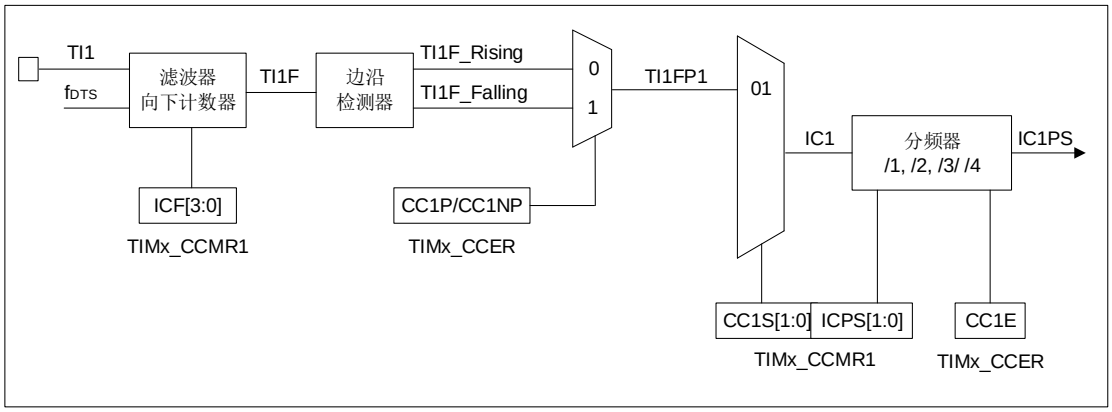


16.5.5. 捕获比较通道

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 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

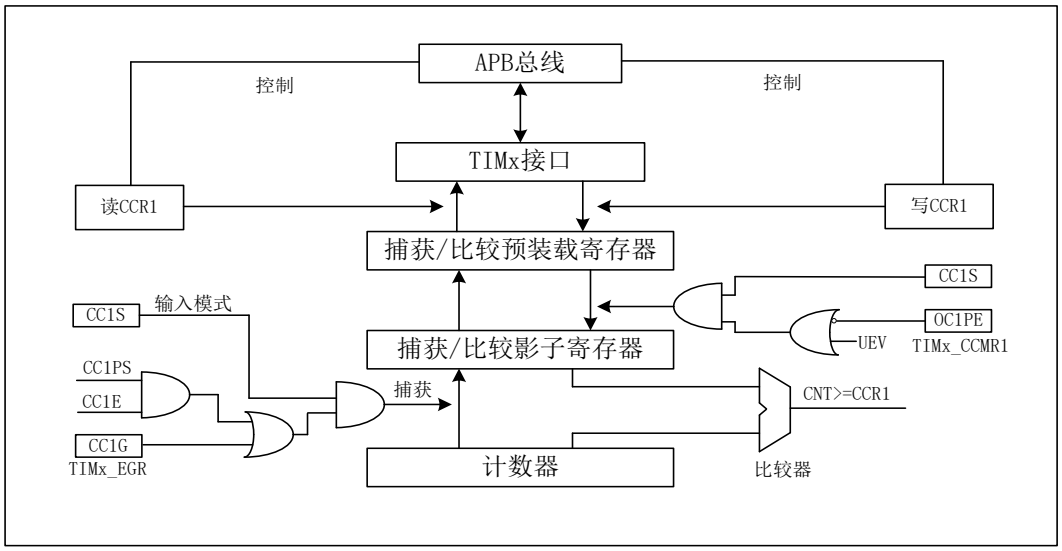
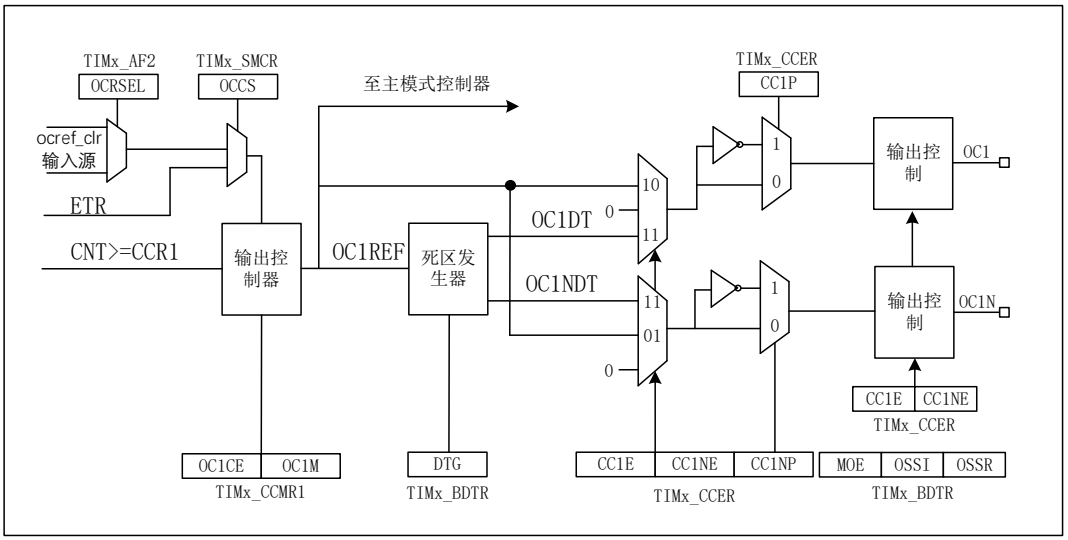


图 16-7 捕获/比较通道的输出部分(通道 1)



16.5.5.1. 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_CCR1 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。

16.5.5.2. 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。

16.5.5.3. 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

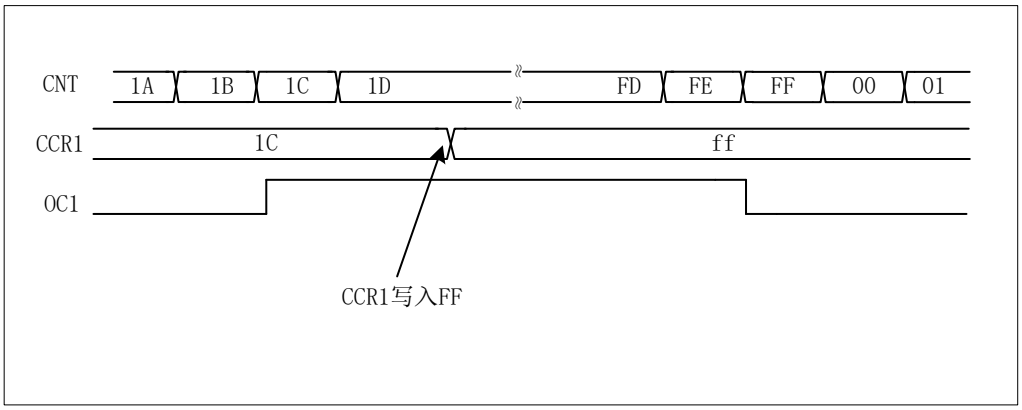
c）置 CCxP = 0 选择极性为高电平有效

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 16-8 输出比较模式，翻转 OC1



16.5.5.4. 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。

16.5.5.5. 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 16-4。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 16-9 带死区插入的互补输出

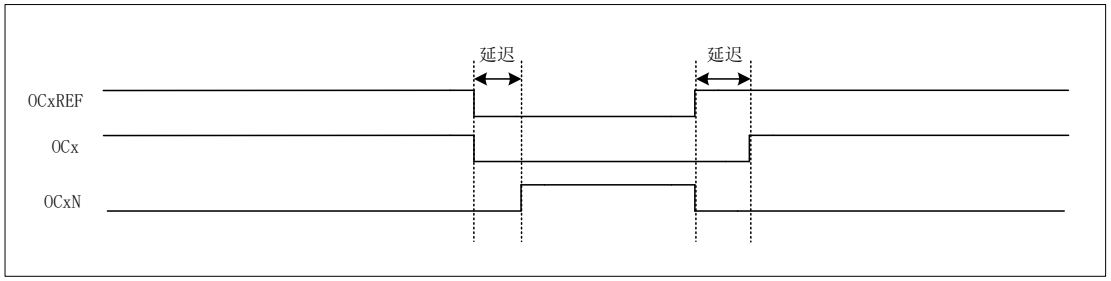


图 16-10 死区波形延迟大于负脉冲

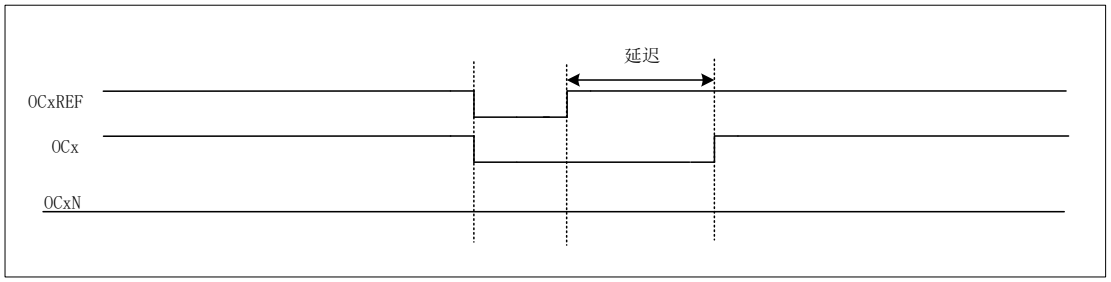
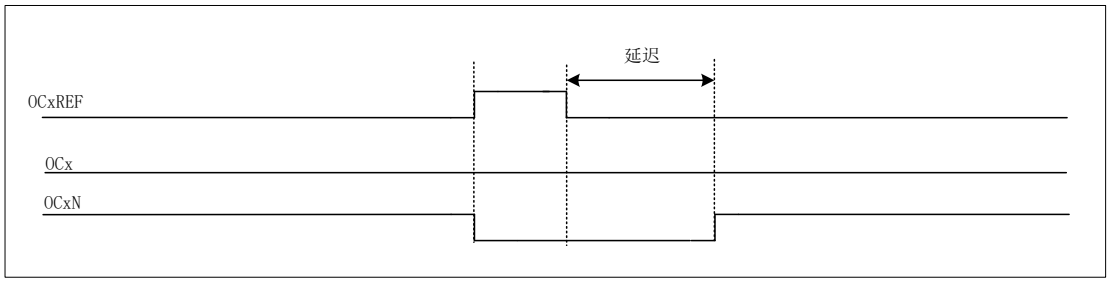


图 16-11 死区波形延迟大于正脉冲



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

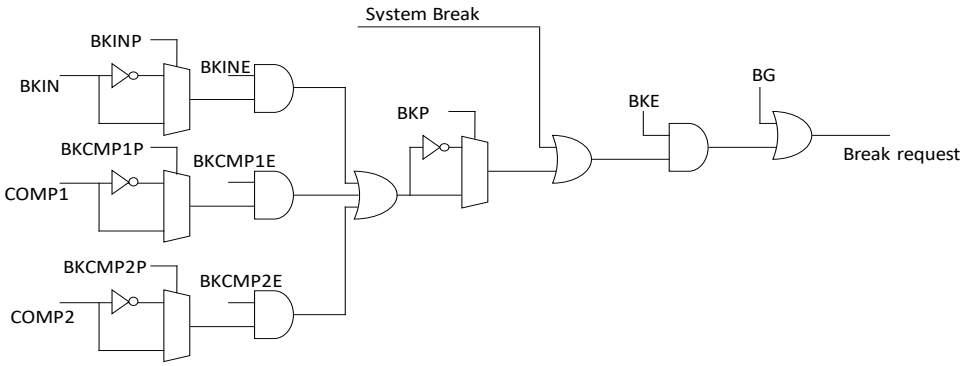
在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。 这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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。

16.5.5.6. 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 16-12 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。

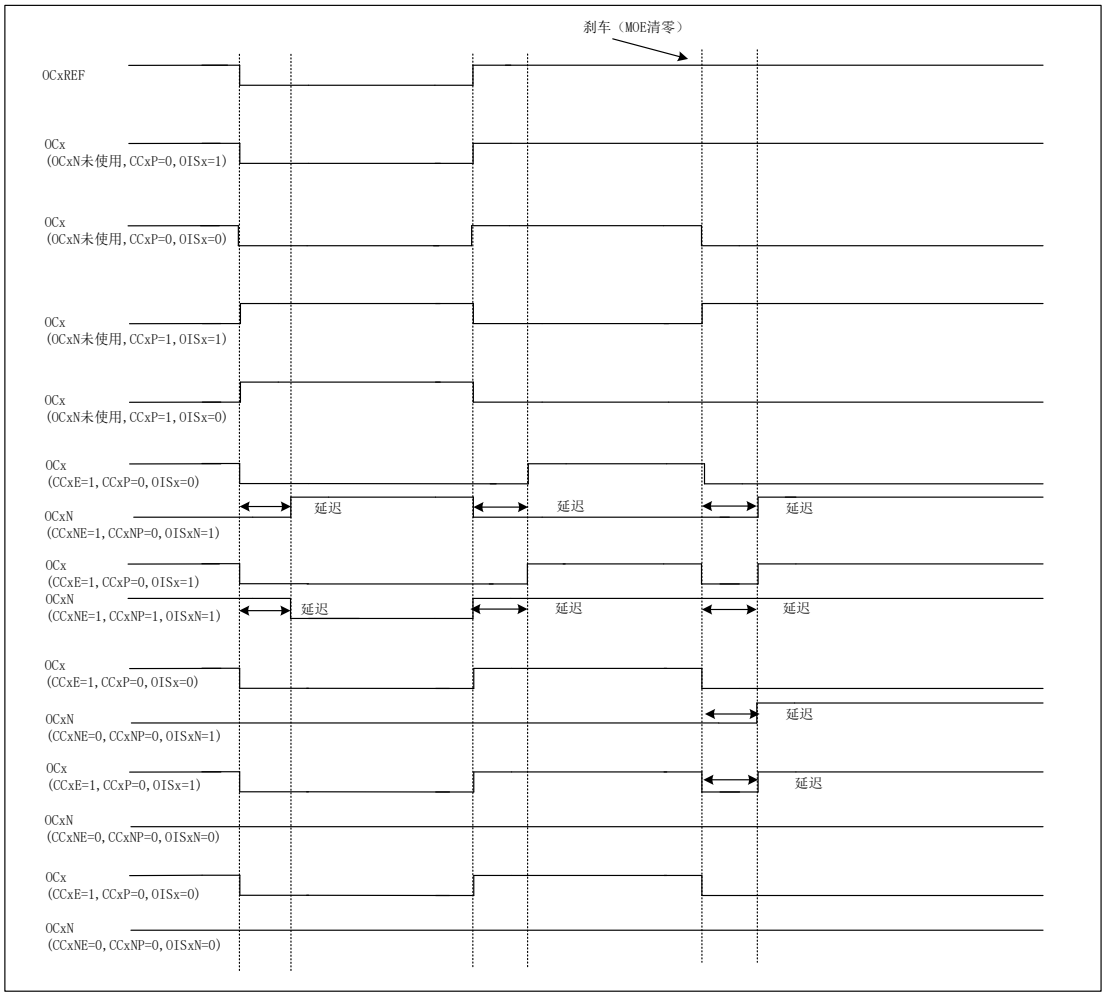
当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- 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_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 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 16-13 响应刹车的输出



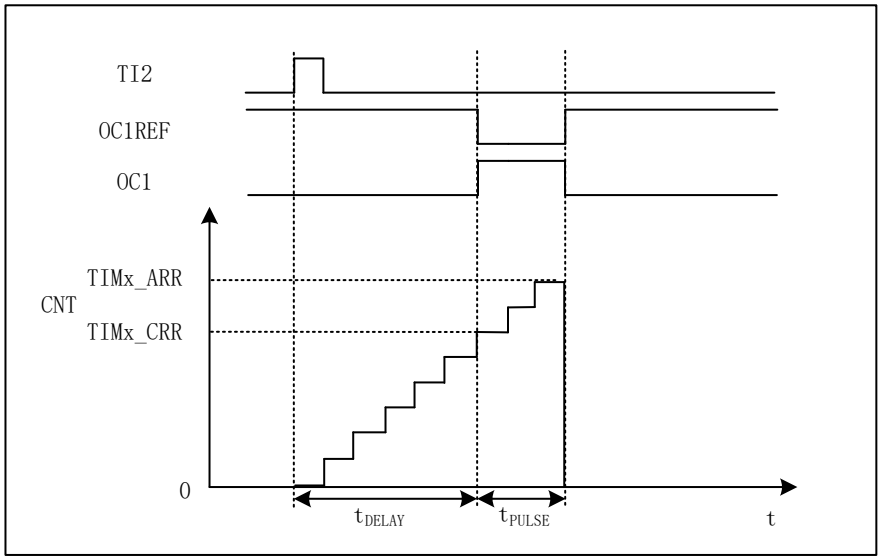
16.5.5.7. 单脉冲模式

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 16-14 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。

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. TIM16/TIM17 寄存器描述

16.6.1. 寄存器列表

TIM16 寄存器基地址：0x40014400

TIM17 寄存器基地址：0x40014800

表 16-3 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

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，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。

16.6.3. 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。

16.6.4. 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。

16.6.5. 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。

16.6.6. 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。

16.6.7. 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。

16.6.8. 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 16-4 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		

16.6.9. 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）

16.6.10. 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’

16.6.11. 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。

16.6.12. 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；

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. 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。

16.6.15. 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， ...

16.6.16. 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。

16.6.17. 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

16.6.18. 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。

16.6.19. 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

16.6.20. 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；

17. 低功耗定时器（LPTIM）

17.1. 概述

低功耗定时器（LPTIM）是一个 16 位定时器，可从降低功耗的最终发展中受益。由于 LPTIM 的时钟源具有多样性，因此 LPTIM 能够在所有电源模式（待机模式除外）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。

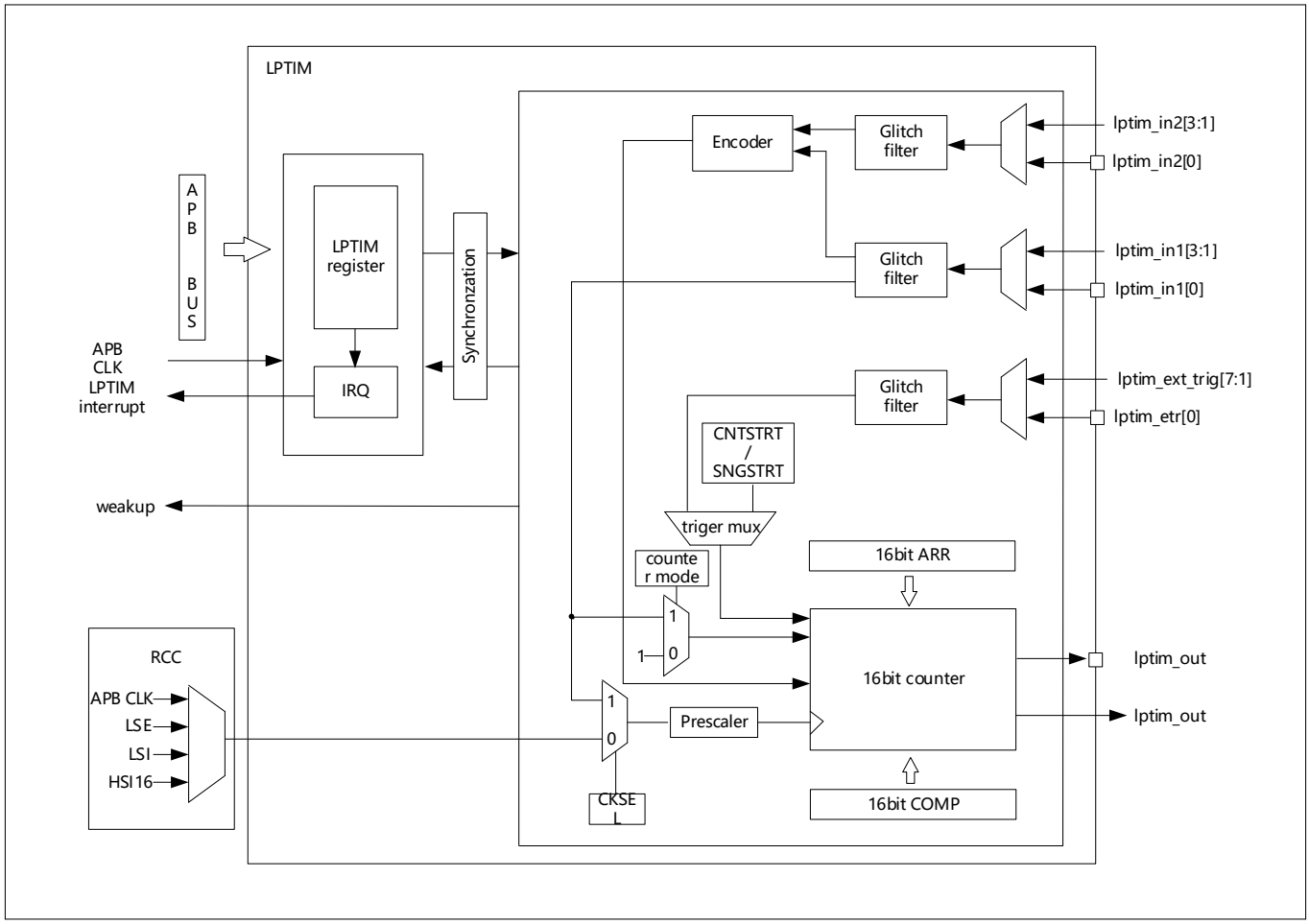
LPTIM 引入了一个灵活的时钟方案，该方案能够提供所需的功能和性能，同时还能最大程度地降低功耗。

17.2. 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：LSE、LSI、HSI16 或 APB 时钟
  - LPTIM 输入的外部时钟源（在没有 LP 振荡器运行的情况下工作，由脉冲计数器应用使用）
- 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

17.3. 框图

图 17-1 LPTIM 框图



lptim_out 是内部 LPTIM 输出信号，可以连接到内部外设。

17.4. 功能描述

17.4.1. LPTIM 触发映射

LPTIM_IN1_MUX、LPTIM_IN2_MUX、LPTIM_EXT_TRIG 映射互联请参看系统级模块互联手册中的 LPTIM 互联部分。

17.4.2. LPTIM 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、RCL、RCH 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 RCL、RCH、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。

17.4.3. 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。

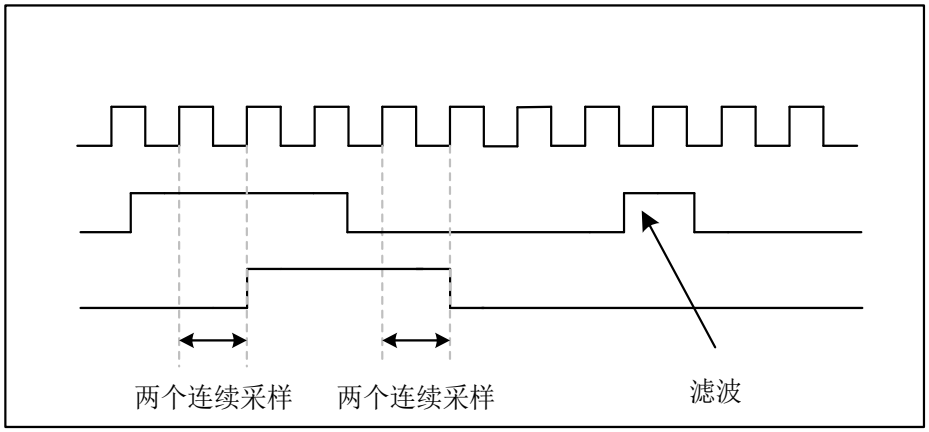
数字滤波器分为两组：

- 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 LPTIM_CFGR1.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 LPTIM_CFGR1.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 17-2 干扰滤波器滤波



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。

17.4.4. 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 17-1 预分频器分频

编程	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。

17.4.5. 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。

17.4.6. 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。

17.4.6.1. 单触发模式

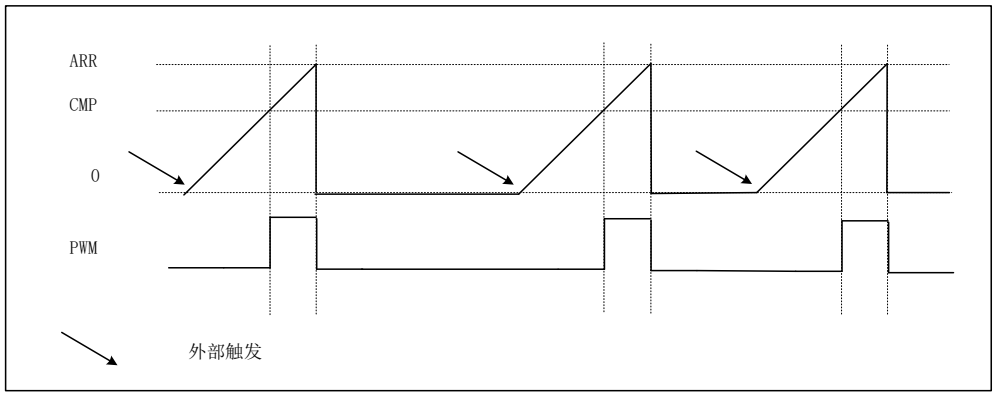
要启用单触发模式，必须将 LPTIM_CFGR1.SNGSTRT 位置 1。

- 当重复计数器功能不可用时：

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 17-3 单触发模式（重复计数器无效）

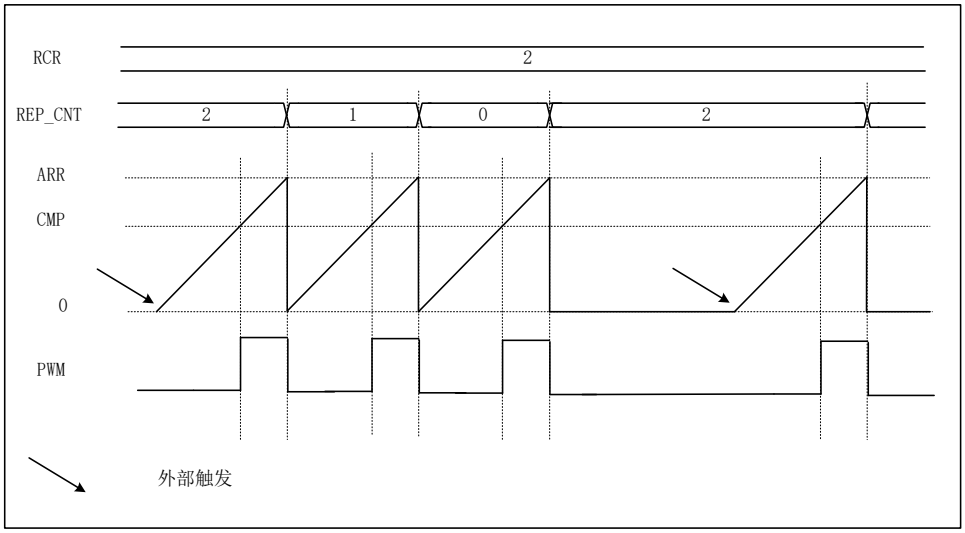


● 当重复计数器功能可用时：

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，如下图所示。

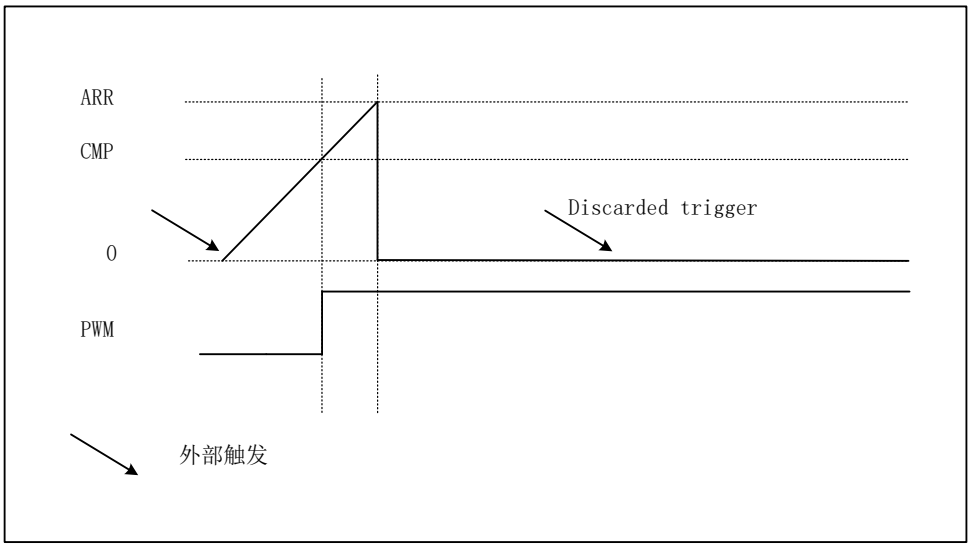
图 17-4 单触发模式（重复计数器有效）



● 单次模式：

LPTIMx_CFGR 寄存器中的 WAVE 位域置 1 时，将激活单次模式。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，如下图所示。

图 17-5 单次模式



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。

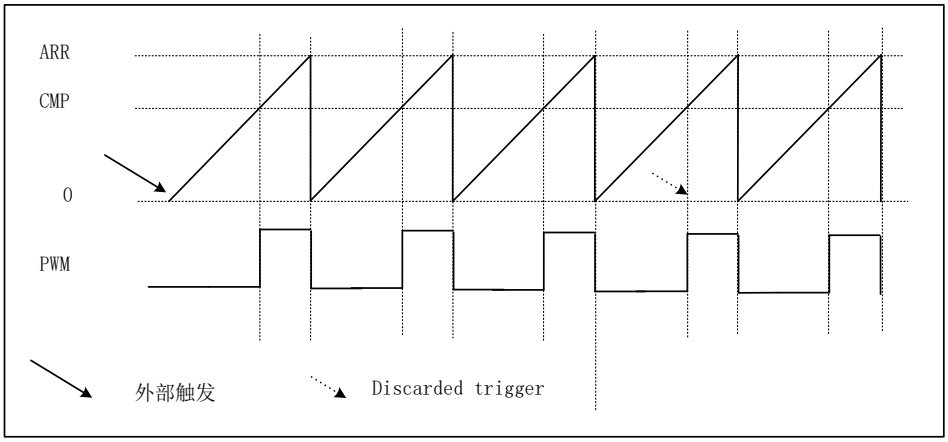
17.4.6.2. 连续模式

要启用连续计数，必须将 LPTIM_CFGR1.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，如下图所示。

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 17-6 连续模式



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。

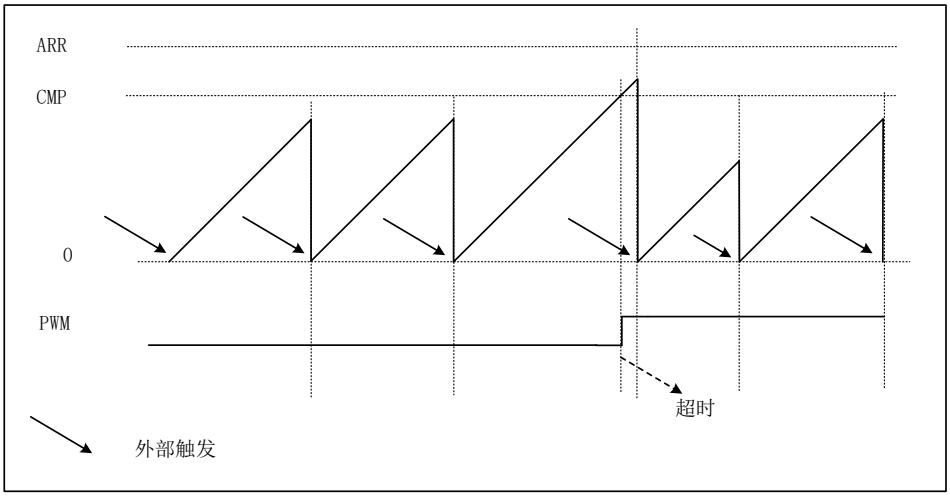
17.4.7. 超时功能

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

LPTIM_CFGR1.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。如下图所示，超时将会置起比较中断等，可用于唤醒等操作。

图 17-7 超时



17.4.8. 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，则会立即将 LPTIM 输出置 1。若 LPTIMx_ARR 寄存器与 LPTIMx_CNT 寄存器匹配，则会立即将 LPTIM 输出复位。
- 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。
- 单次模式：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。

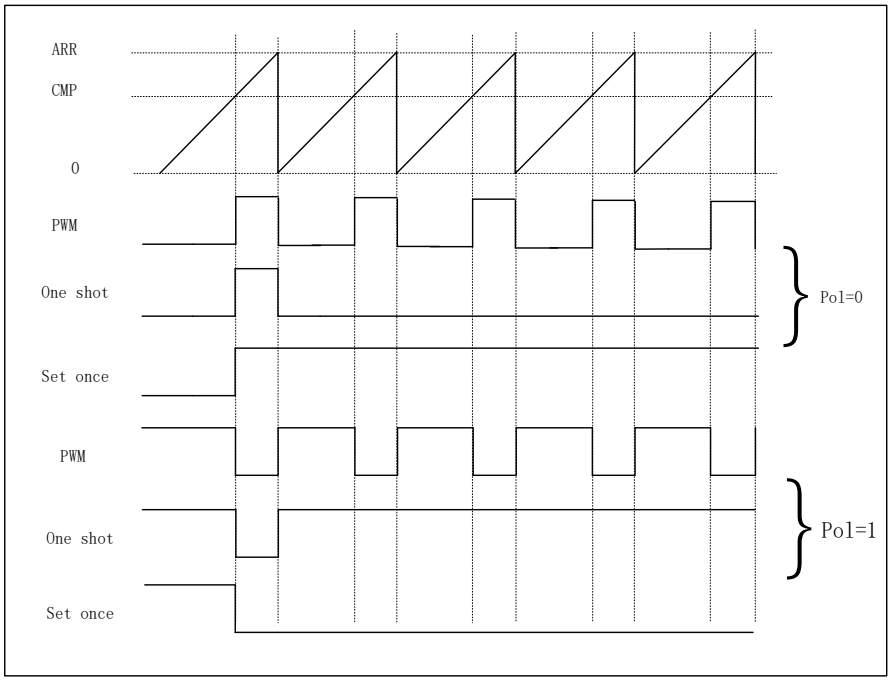
LPTIM 输出波形可通过 WAVE 位配置，具体如下：

- 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（LPTIM_CFGR1.CNTSTRT 或 LPTIM_CFGR1.SNGSTRT）置 1。
- 若将 LPTIM_CFGR1.WAVE 位置 1，则会强制 LPTIM 生成单次模式波形。

LPTIM_CFGR1.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。下图给出了可能在 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 17-8 生成波形



17.4.9. 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）。

LPTIM_CFGR1.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。

17.4.10. 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。LPTIM_CFGR1.CKSEL 位和 LPTIM_CFGR1.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 LPTIM_CFGR1.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）。

17.4.11. 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。

17.4.12. 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同时使用两种复位机制。因此，开发人员应确保专门使用这两种机制。因此，开发人员应该确保这两个机制的独占使用。

17.4.13. 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。同样，如果通过 DOWNIE 位使能，则两个方向改变事件都可能产生中断。

要激活编码器模式，必须将 LPTIM_CFGR1.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 LPTIM_CFGR1.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）。

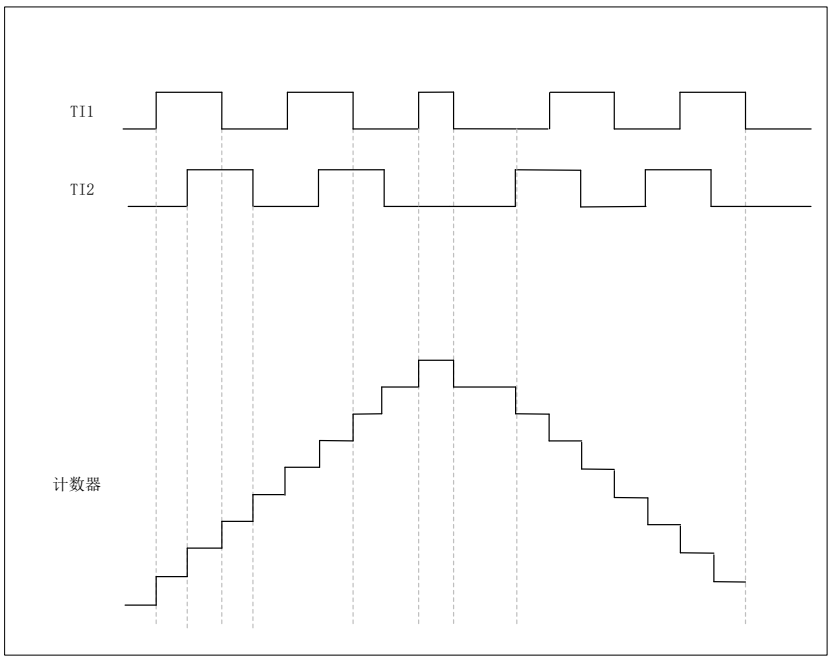
表 17-2 编码器模式计数方案

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 17-9 编码器模式计数



17.4.14. 重复计数器

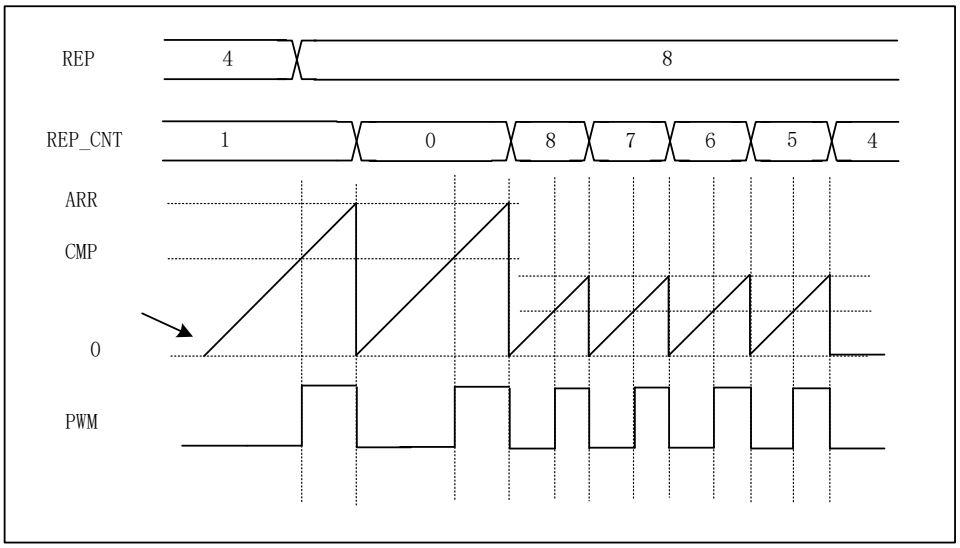
LPTIM 具有一个重复计数器，每次 LPTIM 计数器溢出事件发生时，该计数器就会递减 1。当重复计数器包含零并且 LPTIM 计数器溢出时，将生成重复计数器下溢事件。在每个重复计数器下溢事件之后，向重复计数器加载属于重复寄存器 LPTIM_RCR 的 REP[7:0]位字段的内容。

当重复计数器包含零并且 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 REP[7:0]寄存器设置为 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]位域对重复计数器的内容没有影响，直到发生下一个重复下溢事件为止。重复计数器继续减少每个 LPTIM 计数器的溢出事件，并且仅当产生重复下溢事件时，才将写入 REP[7:0]的新值加载到重复计数器中。此行为在下图中进行了描述。

图 17-10 重复计数器



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（UE）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（UEIE）控制位，则 UE 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 LSI 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。

17.4.15. 调试模式

当微控制器进入调试模式（内核暂停）时，LPTIM 计数器将继续正常工作或停止，具体取决于 DBG 模块中的 DBG_LPTIM_STOP 配置位。

17.5. 低功耗模式

表 17-3 低功耗下 LPTIM 行为

模式	描述
Sleep	无影响。LPTIM 中断可退出 Sleep 模式。
Low-power run	无影响。
Low-power sleep	无影响。LPTIM 中断可退出 Low-power sleep 模式。
Stop 0/1	无影响。LPTIM 时钟源选择 XTL、RCL 时，LPTIM 中断可退出 Stop0/1 模式。
Stop2	无影响。LPTIM 时钟源选择 XTL、RCL 时，LPTIM 中断可退出 Stop2 模式。
Standby	LPTIM 断电，退出 Standby、Powerdown 必须重新初始化 LPTIM。
Powerdown	

17.6. 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）。
- 更新事件
- 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 17-4 中断

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。

17.7. 配置流程

1）配置 RCC 中 LPTIM 的内部计数时钟选择（RCC_CCR2.RCC_CCR2 位）。

2）配置 LPTIM_CFGR1.CKSEL 时钟选择位，以及根据所需功能配置。LPTIM_CFGR1/LPTIM_CFGR12 对应功能位，如时钟极性、输出极性等。

3）配置 ARR/CMP/RCR 寄存器。

4）使能 LPTIM_CR.ENABLE 位。

5）按计数方式启动计数器（SNGSTRT、CNTSTRT）。

注：具体功能配置见对应功能描述章节。

17.8. LPTIM 寄存器

17.8.1. 寄存器列表

偏移	名称	描述
0x00	LPTIM_ISR	LPTIM 中断和状态寄存器
0x04	LPTIM_ICR	LPTIM 中断清零寄存器
0x08	LPTIM_IER	LPTIM 中断使能寄存器
0x0C	LPTIM_CFGR1	LPTIM 配置寄存器 1
0x10	LPTIM_CR	LPTIM 控制寄存器
0x14	LPTIM_CMP	LPTIM 比较寄存器
0x18	LPTIM_ARR	LPTIM 自动重载寄存器
0x1C	LPTIM_CNT	LPTIM 计数器寄存器
0x20	-	保留
0x24	LPTIM_CFGR2	LPTIM 配置寄存器 2
0x28	LPTIM_RCR	LPTIM 重复寄存器

17.8.2. 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	0x0	重复寄存器更新 OK REPOK 由硬件设置以通知应用程序对 LPTIM_RCR 寄存器的 APB 总线写操作已成功完成。 REPOK 标志可以通过向 REPOKCF 写入 1 来清除 LPTIM_ICR 寄存器中的位。 注：如果 LPTIM 不支持重复计数器功能，该位保留。
7	UE	R	0x0	LPTIM 更新事件发生 UE 由硬件设置以通知应用程序已发生更新事件。 UE 标志可以通过向 LPTIM_ICR 寄存器中的 UECF 位写入 1 来清除。 在自主模式下，一旦 LPTIM_ARR 寄存器被 DMA 写入，UE 标志就会由硬件自动清除。 注：如果 LPTIM 不支持重复计数器功能，该位保留。
6	DOWN	R	0x0	计数方向从递增变为递减 (Counter direction change up to down) 在编码器模式下，由硬件将 DOWN 位置 1 时，会通知应用计数方向由递增变为递减。
5	UP	R	0x0	计数方向从递减变为递增 (Counter direction change down to up) 在编码器模式下，由硬件将 UP 位置 1 时，会通知应用计数方向由递减变为递增。
4	ARROK	R	0x0	自动重载寄存器更新成功 (Autoreload register update OK) 由硬件将 ARROK 置 1 时，会通知应用 LPTIM_ARR 寄存器的 APB 总线写操作已成功完成。
3	CMPOK	R	0x0	比较寄存器更新成功 (Compare register update OK) 由硬件将 CMPOK 置 1 时，会通知应用 LPTIM_CMP 寄存器的 APB 总线写操作已成功完成。
2	EXTTRIG	R	0x0	外部触发边沿事件 (External trigger edge event) 由硬件将 EXTTRIG 置 1 时，会通知应用所选的外部触发输入上产生有效边沿。如果由于定时器已启 动而忽略触发事件，则不会将此标志置 1。
1	ARRM	R	0x0	自动重载匹配 (Autoreload match) 由硬件将 ARRM 置 1 时，会通知应用 LPTIM_CNT 寄存器的值已达到 LPTIM_ARR 寄存器的值。
0	CMPM	R	0x0	比较匹配 (Compare match) 由硬件将 CMPM 位置 1 时，会通知应用 LPTIM_CNT 寄存器的值已达到 LPTIM_CMP 寄存器的值。

17.8.3. 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	0x0	重复寄存器更新 OK 清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 REPOK 标志。 注：如果 LPTIM 不支持重复计数器功能，该位保留。
7	UECF	W	0x0	更新事件清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 UE 标志。 注：如果 LPTIM 不支持重复计数器功能，该位保留。

6	DOWNCF	W	0x0	方向变为递减清零标志（Direction change to down Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 DOWN 标志将清零。
5	UPCF	W	0x0	方向变为递增清零标志（Direction change to UP Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 UP 标志将清零。
4	ARROKCF	W	0x0	自动重载寄存器更新成功清零标志（Autoreload register update OK Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 ARROK 标志将清零。
3	CMPOKCF	W	0x0	比较寄存器更新成功清零标志（Compare register update OK Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 CMPOK 标志将清零。
2	EXTTRIGCF	W	0x0	外部触发有效边沿清零标志（External trigger valid edge Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 EXTTRIG 标志将清零。
1	ARRMCF	W	0x0	自动重载匹配清零标志（Autoreload match Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 ARRM 标志将清零。
0	CMPMCF	W	0x0	比较匹配清零标志（compare match Clear Flag） 将 1 写入此位时， LPT_ISR 寄存器中的 CMP 标志将清零。

17.8.4. 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	0x0	重复寄存器更新 OK 中断使能 0：禁止重复寄存器更新 OK 中断 1：重复寄存器更新 OK 中断使能 注：如果 LPTIM 不支持重复计数器功能，该位保留。
7	UEIE	RW	0x0	更新事件中断使能 0：更新事件中断禁止 1：更新事件中断使能 注：如果 LPTIM 不支持重复计数器功能，该位保留。 请参阅
6	DOWNIE	RW	0x0	方向变为递减中断使能（Direction change to down Interrupt Enable） 0：禁止 DOWN 中断 1：使能 DOWN 中断
5	UPIE	RW	0x0	方向变为递增中断使能（Direction change to UP Interrupt Enable） 0：禁止 UP 中断 1：使能 UP 中断
4	ARROKIE	RW	0x0	自动重载寄存器更新成功中断使能（Autoreload register update OK Interrupt Enable） 0：禁止 ARROK 中断 1：使能 ARROK 中断

3	CMPOKIE	RW	0x0	比较寄存器更新成功中断使能（Compare register update OK Interrupt Enable) 0：禁止 CMPOK 中断 1：使能 CMPOK 中断
2	EXTTRIGIE	RW	0x0	外部触发有效边沿中断使能（External trigger valid edge Interrupt Enable) 0：禁止 EXTTRIG 中断 1：使能 EXTTRIG 中断
1	ARRMIE	RW	0x0	自动重载匹配中断使能（Autoreload match Interrupt Enable) 0：禁止 ARRM 中断 1：使能 ARRM 中断
0	CMPMIE	RW	0x0	比较匹配中断使能（Compare match Interrupt Enable) 0：禁止 CMPM 中断 1：使能 CMPM 中断

注意： 必须在 LPTIM 已禁止时（LPTIM_CR.ENABLE 位为 0）才能修改 LPTIM_IER 寄存器

17.8.5. 配置寄存器 1 (LPTIM_CFGR1:0Ch)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	ENC	RW	0x0	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	0x0	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	0x0	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	0x0	波形极性（Waveform shape polarity) WAVEPOL 位控制输出极性 0： LPTIM_CNT 等于 LPTIM_CMP 时，切换为高电平；LPTIM_CNT 等于 LPTIM_ARR 时，切换为低电平；LPTIM 禁止时（LPTIM_CR.ENABLE=0）切换为低电平。其他时间保持电平。 1： LPTIM_CNT 等于 LPTIM_CMP 时，切换为低电平；LPTIM_CNT 等于 LPTIM_ARR 时，切换为高电平；LPTIM 禁止时（LPTIM_CR.ENABLE=0）切换为高电平。其他时间保持电平。
20	WAVE	RW	0x0	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（具体取决于 OPMODE 位） 1： 激活置 1 一次模式

19	TIMOUT	RW	0x0	超时使能（Timeout enable） TIMOUT 位控制超时功能 0：定时器已启动时到达的触发事件将被忽略 1：定时器已启动时到达的触发事件将复位并重新启动计数器
18:17	TRIGEN[1:0]	RW	0x0	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	RSV	-	-	保留
15:13	TRIGSEL[2:0]	RW	0x0	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： 000： lptim_ext_trig0 001： lptim_ext_trig1 010： lptim_ext_trig2 011： lptim_ext_trig3 100： lptim_ext_trig4 101： lptim_ext_trig5 110： lptim_ext_trig6 111： lptim_ext_trig7
12	RSV	-	-	保留
11:9	PRESC[2:0]	RW	0x0	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[1:0]	RW	0x0	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[1:0]	RW	0x0	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[1:0]	RW	0x0	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	0x0	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。

注意： 必须在 LPTIM 已禁止时（ENABLE 位复位为 0）才能修改 LPTIM_CFGR 寄存器。

17.8.6. 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	W	0x0	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	RS	0x0	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	RW	0x0	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	RW	0x0	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	0x0	LPTIM 使能（LPTIM enable) ENABLE 位由软件置 1 和清零。 0：禁止 LPTIM 1：使能 LPTIM

17.8.7. 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[15:0]	RW	0x0	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。

17.8.8. 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[15:0]	RW	0x1	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。

17.8.9. 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[15:0]	R	0x0	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。 应注意的是，为了实现可靠的 LPTIM_CNT 寄存器读访问，必须执行两次连续的读访问并进行比较。 当两次连续读访问的值相等时，可认为读访问可靠。

17.8.10. 配置寄存器 2 (LPTIM_CFGR2: 24h)

位域	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5:4	IN2SEL[1:0]	RW	0x0	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。 00: lptim_in2_mux0 01: lptim_in2_mux1 10: lptim_in2_mux2 11: lptim_in2_mux3

3:2	RSV	-	-	保留
1:0	IN1SEL[1:0]	RW	0x0	LPTIM 输入 1 选择（LPTIM input 1 selection） IN2SEL 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。 00: lptim_in1_mux0 01: lptim_in1_mux1 10: lptim_in1_mux2 11: lptim_in1_mux3

注意：LPTIM_RCR 寄存器只能在 LPTIM 使能（ENABLE 位设置为 1）时修改。

17.8.11. 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[7:0]	RW	0x0	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。

18. 独立看门狗（IWDT）

18.1. 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。

18.2. 主要特性

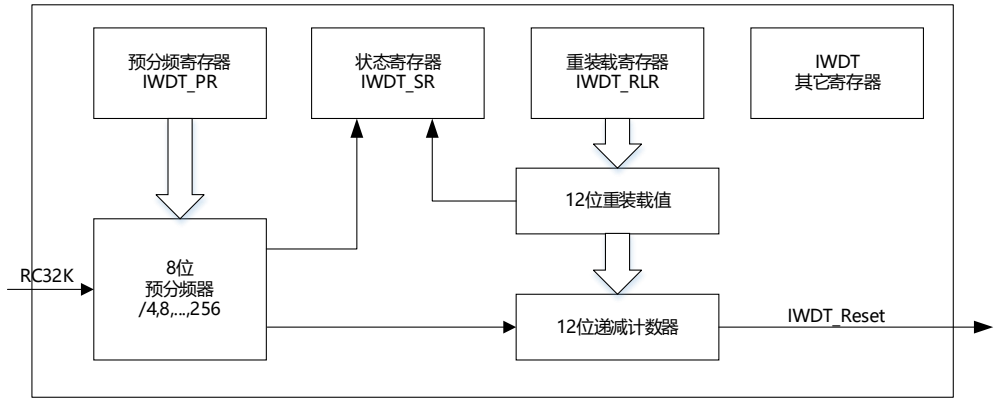
- 自由运行的 12 位向下计数器
- 看门狗被激活后，计数器计数至 0x000 时产生复位
- 当递减计数器在窗口外被重新装载，则产生复位
- 可编程预分频因子和可编程装载值
- 时钟由独立的 RCL 时钟提供(可在停止和待机模式下工作)
- 可作为 STOP 模式唤醒源

18.3. 功能描述

18.3.1. 结构框图

结构框图如下：

图 18-1 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。

18.3.2. 时钟

独立看门狗（IWDT）由其专用低速时钟（RCL）驱动，因此即便在主时钟发生故障时仍然保持工作状态。IWDT 和 RCL 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。RCL 通过设置 RCC_STDBY_CTRL 的 RCLEN 为高，使能 RCL 时钟。

18.3.3. 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)*tRCL*预分频系数

下表为根据不同预分频系数的超时时间。

表 18-1 独立看门狗超时时间（RCL 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。

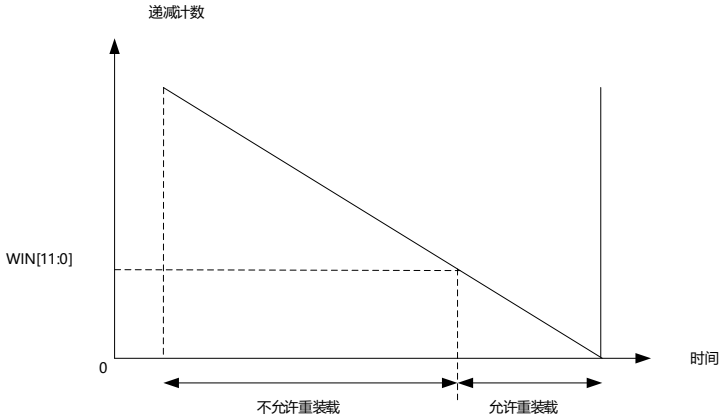
18.3.4. 寄存器访问保护

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。

18.3.5. 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 18-2 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。

18.3.6. 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)*tRCL*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。

18.3.7. 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。

18.3.8. 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 RCLEN 为高，使能 RCL 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。

18.4. 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。

18.5. IWDT 寄存器描述

18.5.1. 寄存器列表

IWDT 寄存器基地址：0x40003000

偏移	名称	描述
0x00	IWDT_CMDR	命令寄存器
0x04	IWDT_PR	预分频寄存器
0x08	IWDT_RLR	重装载寄存器
0x0C	IWDT_SR	状态寄存器
0x10	IWDT_WINR	窗口寄存器
0x14	IWDT_WUTR	唤醒值设置

18.5.2. 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。

18.5.3. 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）。

18.5.4. 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）。

18.5.5. 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

18.5.6. 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）。

18.5.7. 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）。

19. 系统看门狗（WDT）

19.1. 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。

19.2. 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。

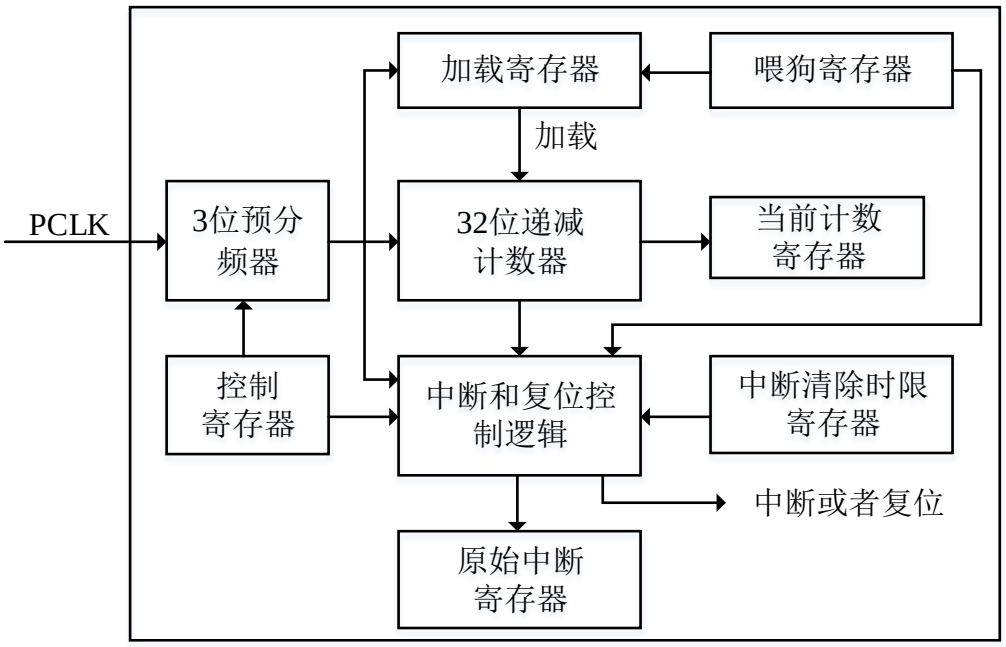
19.3. 功能描述

19.3.1. 概述

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。

19.3.2. 结构框图

图 19-1 WDT 结构图



19.3.3. 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

19.3.4. 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。

19.3.5. 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。

19.3.6. 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

19.3.7. 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。

19.4. 配置流程

19.4.1. 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。

19.4.2. 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。

19.5. WDT 描述

19.5.1. 寄存器列表

WDT 寄存器基地址：0x4000_2C00

偏移	名称	描述
0x00	WDT_LOAD	加载寄存器
0x04	WDT_COUNT	当前计数寄存器
0x08	WDT_CTRL	控制寄存器
0x0C	WDT_FEED	喂狗寄存器
0x10	WDT_INTCLRTIME	中断清除时限寄存器
0x14	WDT_RIS	原始中断状态寄存器

19.5.2. 加载寄存器(WDT_LOAD: 00h)

位域	名称	属性	复位值	描述
31:0	LOAD	R/W	0xFFFFFFFF	加载值

19.5.3. 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。

19.5.4. 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	R/W	0	看门狗使能 0：禁止看门狗 1：使能看门狗
6	MODE	R/W	0	看门狗模式 0：复位 1：中断
5	RSV	-	-	保留
4	INTEN	R/W	0	中断使能 0：禁止中断 1：使能中断
3	RSV	-	-	保留
2:0	DIVISOR	R/W	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

19.5.5. 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。

19.5.6. 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	R/W	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。

19.5.7. 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。

20. 实时时钟（RTC）

20.1. 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。

20.2. 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 RCL 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

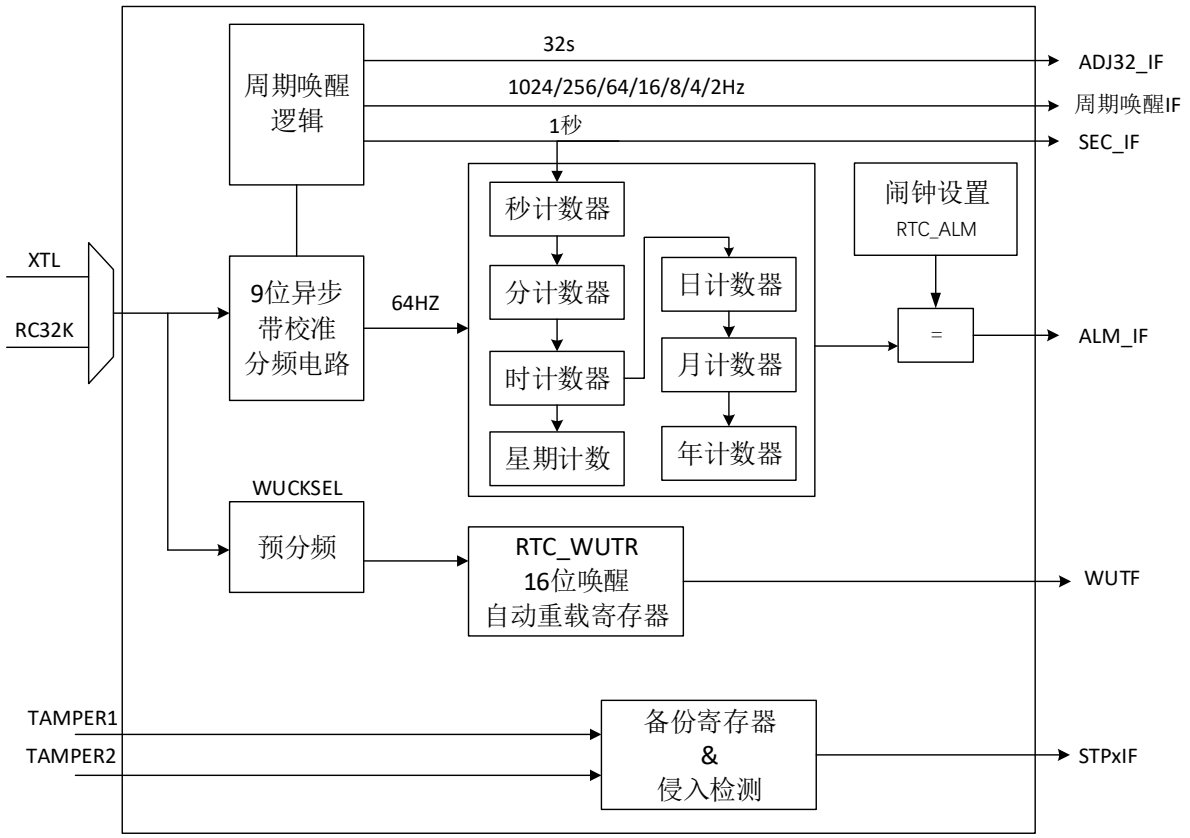
20.3. 功能描述

实时时钟的时钟源可配置为外部低速晶振 XTL 和内部低速 RCL；默认使用内部低速 RCL。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。

20.3.1. 结构框图

整体框图如下：

图 20-1 RTC 内部结构框图



20.3.2. 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。

20.3.3. 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。

20.3.4. 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。

20.3.5. 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。

20.3.6. 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。

20.3.7. 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。

20.3.8. 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。

20.3.9. 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。

20.3.10. RTC 低功耗

表格 20-1 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

20.3.11. 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

表 20-1 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

20.4. 配置流程

20.4.1. 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 RCL 时，需要使能 RCL 时钟 RCC. STDBYCTL.RCLEN=1；当选择外部低速晶振 XTL 时，需要使能 XTL 振荡器 RCC. STDBYCTL.XTLEN=1;

20.4.2. 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

20.4.3. 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。

20.4.4. 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。

20.4.5. 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。

20.4.6. 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。

20.4.7. 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

20.5. RTC 寄存器描述

20.5.1. 寄存器列表

RTC 寄存器基地址：0x40002800

偏移	名称	描述
0x00	RTC_WP	写保护寄存器
0x04	RTC_IE	中断使能寄存器
0x08	RTC_SR	中断标志寄存器
0x0C	RTC_SEC	秒计数寄存器
0x10	RTC_MIN	分计数寄存器
0x14	RTC_HOUR	时计数寄存器
0x18	RTC_DAY	日计数寄存器
0x1C	RTC_WEEK	周计数寄存器
0x20	RTC_MONTH	月计数寄存器
0x24	RTC_YEAR	年计数寄存器
0x28	RTC_ALARM	闹钟寄存器
0x2C	RTC_CR	控制寄存器
0x30	RTC_ADJUST	时钟误差补偿寄存器
0x44	RTC_CLKSTAMP1	时间戳寄存器 1
0x48	RTC_CALSTAMP1	日历戳寄存器 1
0x4C	RTC_CLKSTAMP2	时间戳寄存器 2
0x50	RTC_CALSTAMP2	日历戳寄存器 2
0x54	RTC_WUTR	唤醒定时器寄存器
0x70~0xAC	RTC_BAKUP0~15	备份寄存器 0~15

20.5.2. 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。

20.5.3. 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

20.5.4. 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

20.5.5. 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。

20.5.6. 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。

20.5.7. 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。

20.5.8. 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。

20.5.9. 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。

20.5.10. 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。

20.5.11. 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。

20.5.12. 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

20.5.13. 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

20.5.14. 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

20.5.15. 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

20.5.16. 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

20.5.17. 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

20.5.18. 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

20.5.19. 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

20.5.20. 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。

21. 通用异步收发器（UART）

21.1. 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。芯片上集成了四个 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。

21.2. 主要特性

21.2.1. 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
- 波特率支持整数和小数分频（有关最大 APB 频率时的波特率值，请参见数据手册）
- 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

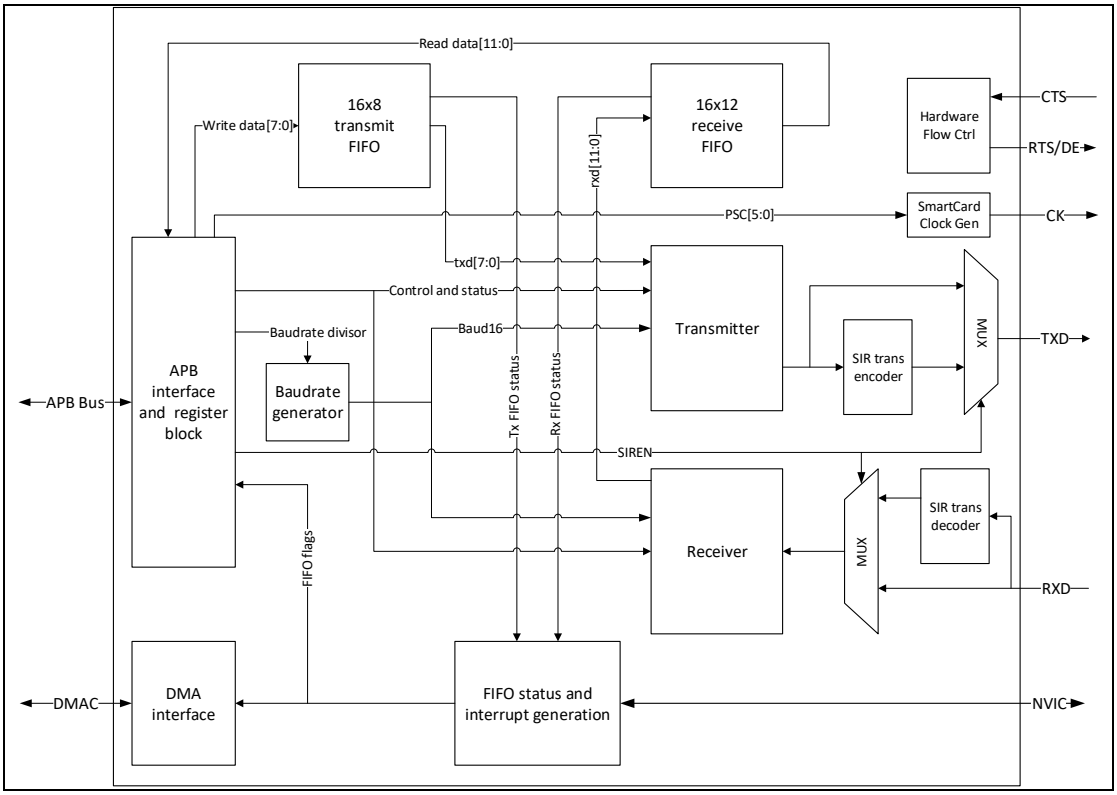
21.2.2. UART 的扩展功能特性

- 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

21.3. 功能描述

21.3.1. 结构框图

图 21-1 UART 结构框图



21.3.2. 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。

21.3.3. 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）。

21.3.4. 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。

21.3.5. 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。

21.3.6. 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

21.3.7. 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。

21.3.8. 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。

21.3.9. 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。

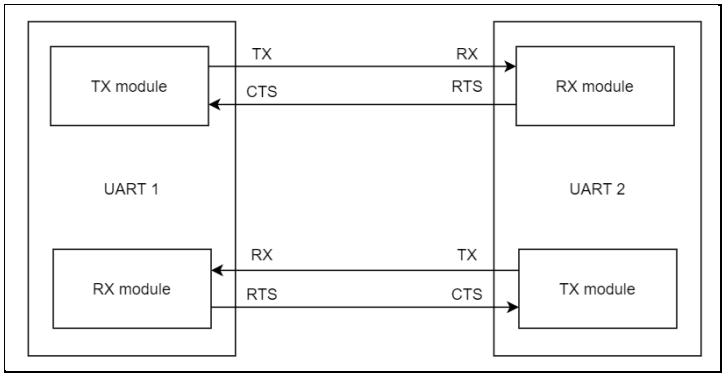
开启 FIFO 并配合 IDLEI 标识完成数据包的接收

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。

21.3.10. 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 21-2 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。

21.3.11. 通过 DMA 进行 UART 数据收发

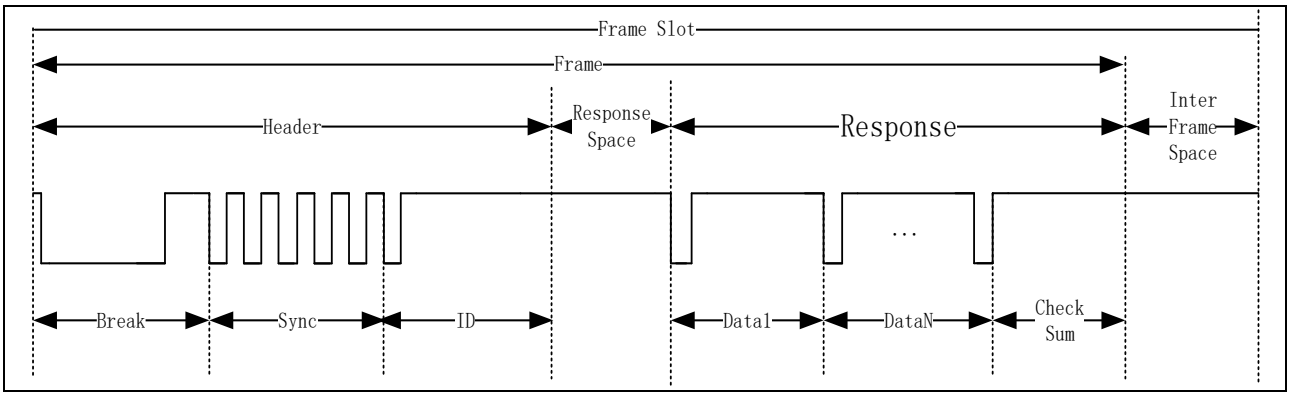
每个 UART 接口的 TX 和 RX 都支持 DMA 功能，都有其对应的 DMA 请求号。

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。

21.3.12. LIN 总线功能

图 21-3 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 21-4 LIN 间隔场格式

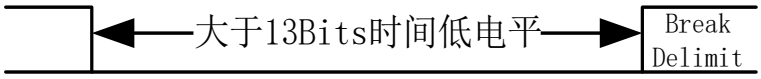
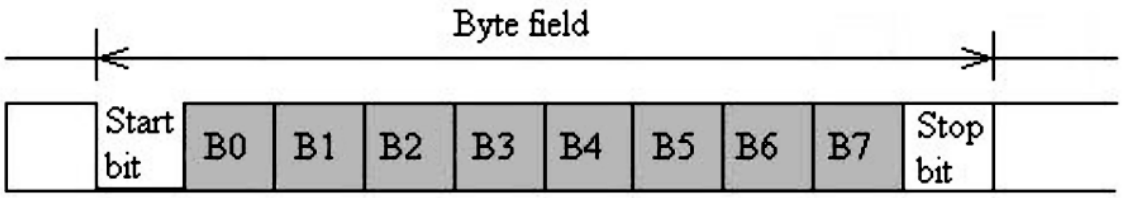


图 21-5 LIN 字符格式



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。

21.3.13. IrDA SIR 功能

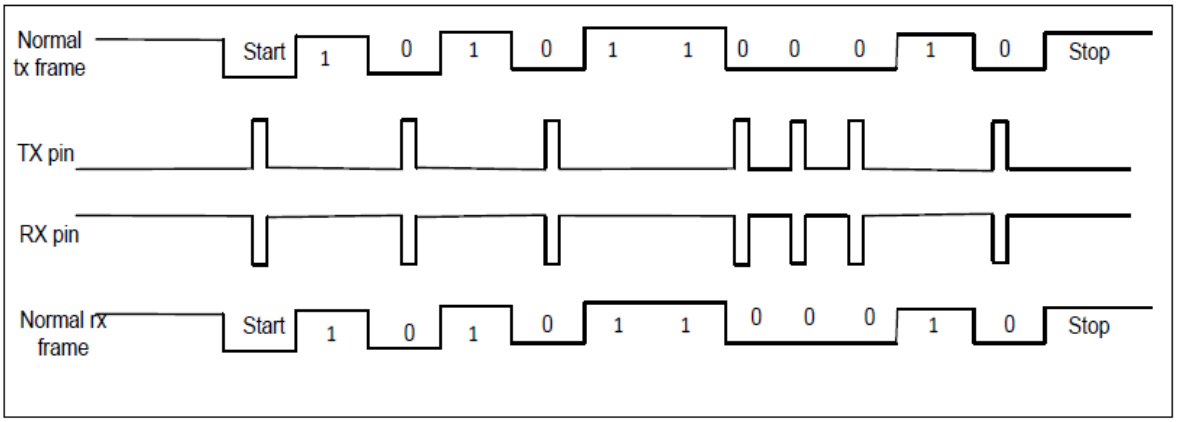
使能 UART_CR1.SIREN 位，即可打开 IrDA SIR 红外功能。

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。

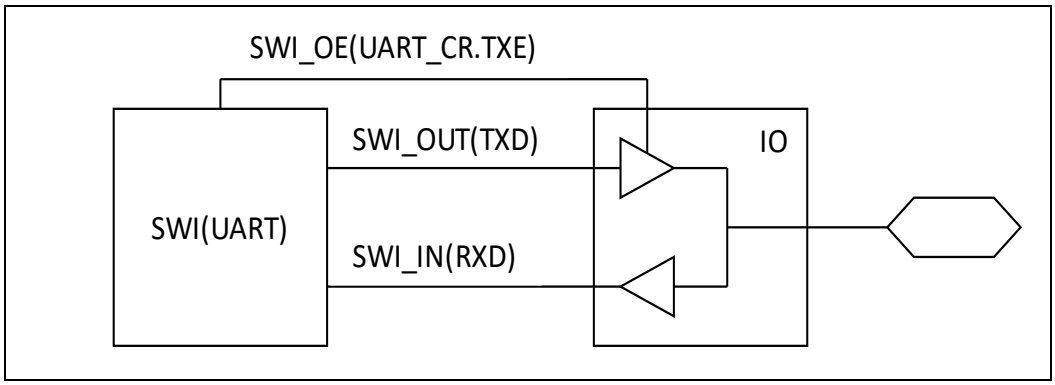
启用 Low Power 模式时，高电平脉冲宽度为为最高波特率 115200 时的一个位宽的 3/16。

图 21-6 IrDA SIR 数据调制解调



21.3.14. 单线半双工通信

图 21-7 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。

21.3.15. 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。

21.3.16. 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。

21.3.17. 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。

21.3.18. 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 Pin 来自动控制收发器的收发方向。ACM32G103 自带 RS485 收发器的方向控制 DE Pin，即 UARTx_RTS_DE Pin（详见 ACM32G103 数据手册）。

【注意，UART RTS 和 DE 共用一个管脚，DE 功能优先。】

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。

21.4. UART 中断

表格 21-1 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

21.5. 配置流程

21.5.1. 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

21.5.2. 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）

21.5.3. 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
  -

21.5.4. 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。

21.5.5. 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

21.5.6. 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

21.5.7. 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

21.5.8. 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

21.5.9. 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

21.6. 寄存器描述

21.6.1. 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

偏移	名称	描述
0x00	UART_DR	数据寄存器
0x04	UART_FR	标志寄存器
0x08	UART_BRR	波特率寄存器
0x0C	UART_IE	中断使能寄存器
0x10	UART_ISR	中断和状态寄存器
0x14	UART_CR1	控制寄存器 1
0x18	UART_CR2	控制寄存器 2
0x1C	UART_CR3	控制寄存器 3
0x20	UART_GTPR	保护时间和预分频寄存器
0x24	UART_BCNT	比特计时寄存器

21.6.2. 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	R/W	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

21.6.3. 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 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
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

21.6.4. 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	R/W	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	R/W	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

21.6.5. 中断使能寄存器(UART_IE: 0Ch)

位域	名称	属性	复位值	描述
31:15	RSV	RO	0	保留
14	ABRI	R/W	0	波特率自适应完成中断使能位 0：禁止 1：使能
13	IDLEI	R/W	0	IDLE 中断使能位 0：禁止 1：使能
12	BCNTI	R/W	0	Bit Count Timeout 中断使能位 0：禁止 1：使能
11	LBDI	R/W	0	LIN Break Detection 中断使能位 0：禁止 1：使能
10	OEI	R/W	0	overrun 中断使能位 0：禁止 1：使能
9	BEI	R/W	0	break error 中断使能位 0：禁止 1：使能

8	PEI	R/W	0	奇偶校验错误中断使能位 0：禁止 1：使能
7	FEI	R/W	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	R/W	0	发送中断使能位 0：禁止 1：使能
4	RXI	R/W	0	接收中断使能位 0：禁止 1：使能
3:0	RSV	RO	0	保留

21.6.6. 中断和状态寄存器(UART_ISR: 10h)

位域	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	ABRI	RO	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	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
12	BCNTI	RO	0	Bit Count Timeout 原始中断 0：无中断 1：有中断 写 1 清 0
11	LBDI	RO	0	LIN Break Detection 原始中断 0：无中断 1：有中断 写 1 清 0
10	OEI	RO	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	RO	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	RO	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	RO	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	RO	0	保留
5	TXI	RO	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： ➢ 将 UART_ICR[5]寄存器写 1 ➢ 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： ➢ 写 UART_ISR[5]寄存器 ➢ 发送未完成时，写新的数据到 UART_DR 寄存器 ● 写 1 清 0

4	RXI	RO	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： ➢ 写 UART_ICR[4]寄存器 ➢ 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： ➢ 写 UART_ISR[4]寄存器 ➢ 读 UART_DR 寄存器 ● 写 1 清 0
3:0	RSV	-	-	保留

21.6.7. 控制寄存器 1(UART_CR1: 14h)

位域	名称	属性	复位值	描述
15	CTSEN	R/W	0	CTS 流控制使能位 0：禁止 1：使能
14	RTSEN	R/W	0	RTS 流控制使能位 0：禁止 1：使能
13:12	RSV	RO	0	保留
11	RTS	R/W	0	RTS 输出管脚状态 0：RTS 输出高电平 1：RTS 输出低电平
10	RSV	R/W	0	保留
9	RXE	R/W	1	接收使能位 0：禁止 1：使能
8	TXE	R/W	1	发送使能位 0：禁止 1：使能
7:6	RSV	RO	0	保留

5	DMAONERR	R/W	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	R/W	0	发送 DMA 使能 0：不使能 1：使能
3	RXDMAE	R/W	0	接收 DMA 使能 0：不使能 1：使能
2	SIRLP	R/W	1’ b0	IrDA SIR 低功耗模式使能位： 0：禁止 1：使能
1	SIREN	R/W	1’ b0	IrDA SIR ENDEC 模块使能位： 0：禁止 1：使能
0	UARTEN	R/W	1b’ 0	UART 使能位 0：禁止 1：使能

21.6.8. 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	R/W	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	R/W	0	本设备的 UART 节点地址 该位域给出本设备 UART 节点的地址
10:8	RSV	RO	0	保留
7	CLKEN	R/W	0	CK 时钟使能 0：禁止 CK 引脚 1：使能 CK 引脚
6	NACK	R/W	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	R/W	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	R/W	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	R/W	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	R/W	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	R/W	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

21.6.9. 控制寄存器 3(UART_CR3: 1Ch)

位域	名称	属性	复位值	描述
15:13	RXIFLSEL	R/W	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	R/W	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	R/W	0	校验模式选择位 0：奇/偶校验 1：0/1 校验

8:6	WLEN	R/W	0	字宽选择位 000：5bits 001：6bits 010：7bits 011：8bits 1xx：9bits
5	FEN	R/W	0	FIFO 使能位 0：禁止 FIFO 1：使能 FIFO
4	RSV	RO	0	保留
3	STP2	R/W	0	停止位数选择位： 0：1 位停止位 1：2 位停止位
2	EPS	R/W	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	R/W	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	R/W	0	BREAK 发送使能位 0：禁止 1：使能

21.6.10. 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	R/W	0x00	智能卡模式下使用。保护时间，以 etu 为单位
7:0	PSC	R/W	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

21.6.11. 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	R/W	13	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。

22. 低功耗串口（LPUART）

22.1. 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。

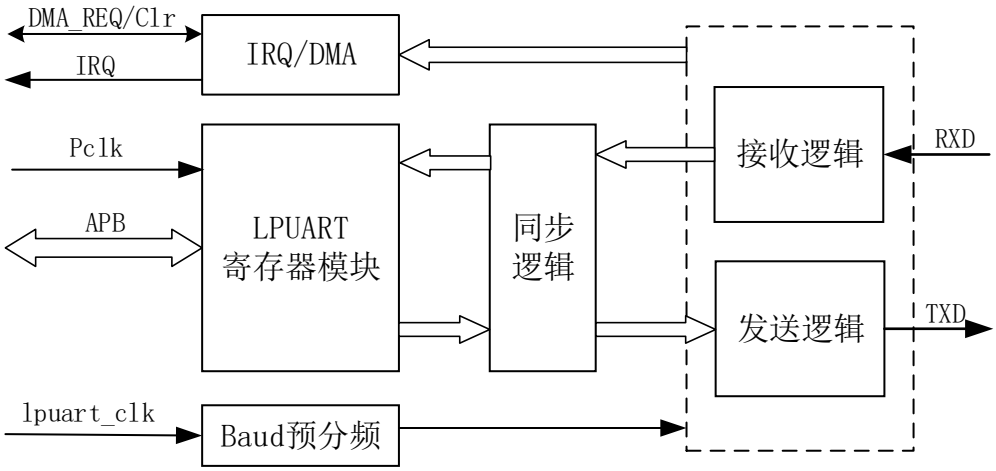
22.2. 主要特性

- 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
- 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
- STOP 模式下唤醒系统：收到字节必须与写入匹配地址相等时才能有效唤醒

22.3. 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。

22.3.1. 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

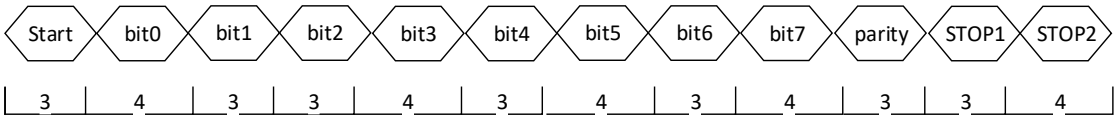
通过 RCC_CCR2 寄存器进行设置，当选择 RCL 或 XTL 为时钟源时，最大波特率支持 9600，LPUART 提供低功耗支持。当选择 PCLK 分频作为时钟源，支持更高波特率，LPUART 提供高性能支持。

22.3.2. 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。波特率整数部分配置为 3 分频（每个比特 3-4 个时钟，设置值为 2），接收采样设为第 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 22-1 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，波特率加权平均值为 3.416。

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。

22.3.3. 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。

22.3.4. 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。

22.3.5. 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。

22.3.6. 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。

22.3.7. 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。

22.3.8. 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。

22.3.9. 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

22.3.10. 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

22.4. 配置流程

22.4.1. 初始化

1）配置时钟源（RCC_CCR2. LPUART1CLKSEL 和 RCC_CCR2. LPUARTDIV）。

2）禁止发送（LPUART_CR.TXE）、禁止接收（LPUART_CR.RXE）。

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.TXE）、使能接收（LPUART_CR.RXE）。

22.4.2. 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。

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）。

22.4.3. 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。

22.5. LPUART 寄存器描述

22.5.1. 寄存器列表

LPUART 寄存器基地址：

偏移	名称	复位值	描述
0x00	LPUART_RXDR		接收数据寄存器
0x04	LPUART_TXDR		发送数据寄存器
0x08	LPUART_LCR		线控制寄存器
0x0C	LPUART_CR		控制寄存器
0x10	LPUART_IBAUD		波特率整数部分
0x14	LPUART_FBAUD		波特率整数部分
0x18	LPUART_IE		中断使能寄存器
0x1C	LPUART_SR		状态寄存器
0x20	LPUART_ADDR		地址寄存器

22.5.2. 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

22.5.3. 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

22.5.4. 线控寄存器(LPUART_LCR: 08h)

位域	名称	属性	复位值	描述
15:10	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

22.5.5. 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	TXE	RW	0	发送使能位 0：禁止 1：使能
0	RXE	RW	0	接收使能位： 0：禁止 1：使能

22.5.6. 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。

22.5.7. 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

22.5.8. 中断使能寄存器(LPUART_IE: 18h)

位域	名称	属性	复位值	描述
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

22.5.9. 状态寄存器(LPUART_SR: 1Ch)

位域	名称	属性	复位值	描述
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

22.5.10. 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

23. 串行外设接口（SPI）

23.1. 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），详见**错误!未找到引用源。**一节。

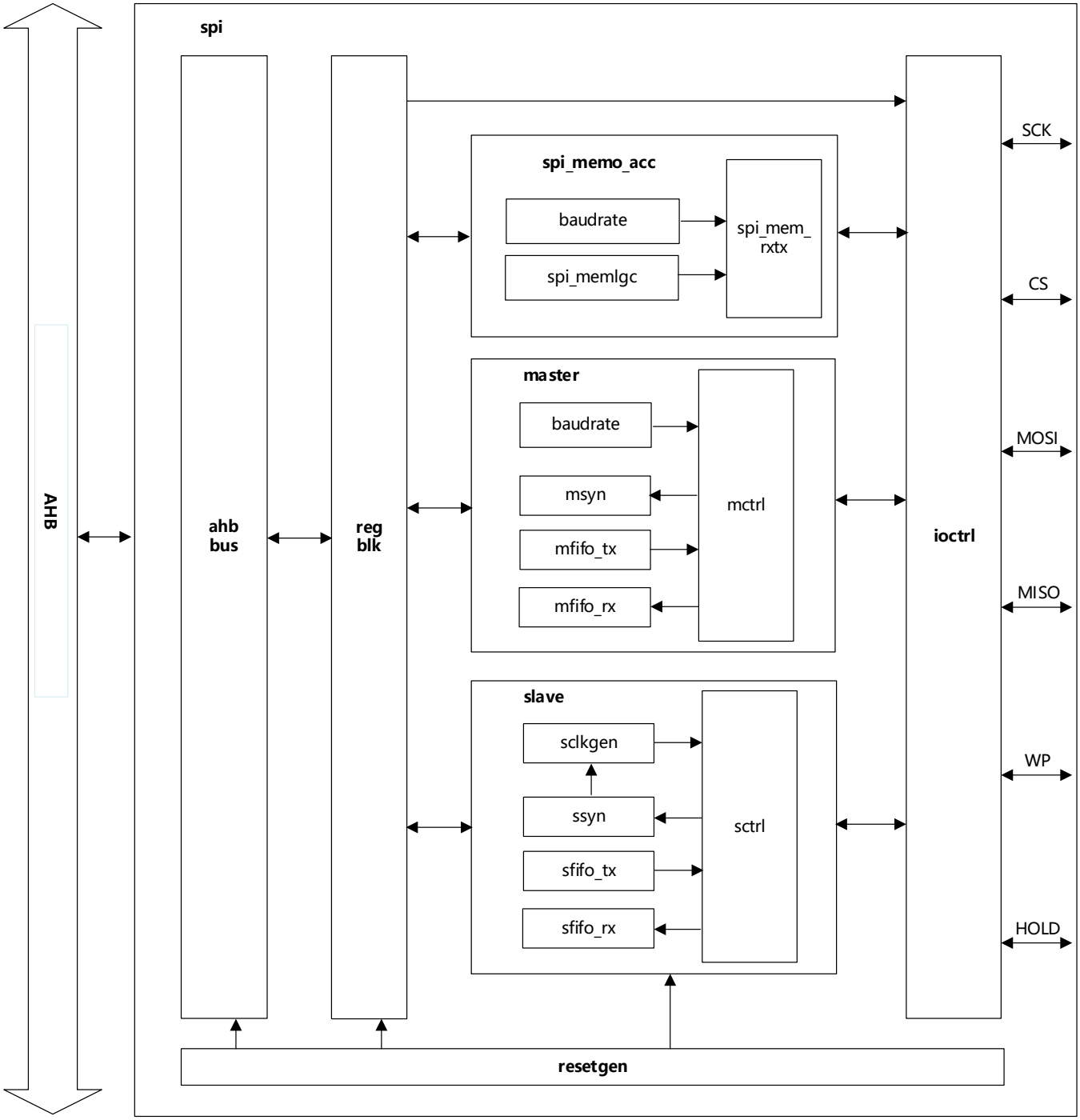
23.2. 主要特性

- 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
- 支持支持标准 SPI，支持 DSPI，支持 QSPI
- 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 XIP 内存映射模式（仅 SPI3 支持）
- 内存映射模式允许 8、16 和 32 位数据访问

23.3. 功能描述

23.3.1. 结构框图

图 23-1 SPI 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 模块为间接模式或 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。

23.3.2. 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（仅 SPI3 支持）。

23.3.3. 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。

表 23-1 三种模式下的引脚复用关系

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)

23.3.4. 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。

23.3.5. 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.

23.3.6. 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。

表 23-2 SPI 模式

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）

23.3.7. 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 23-2 SPI 一线模式时序图（MSB）

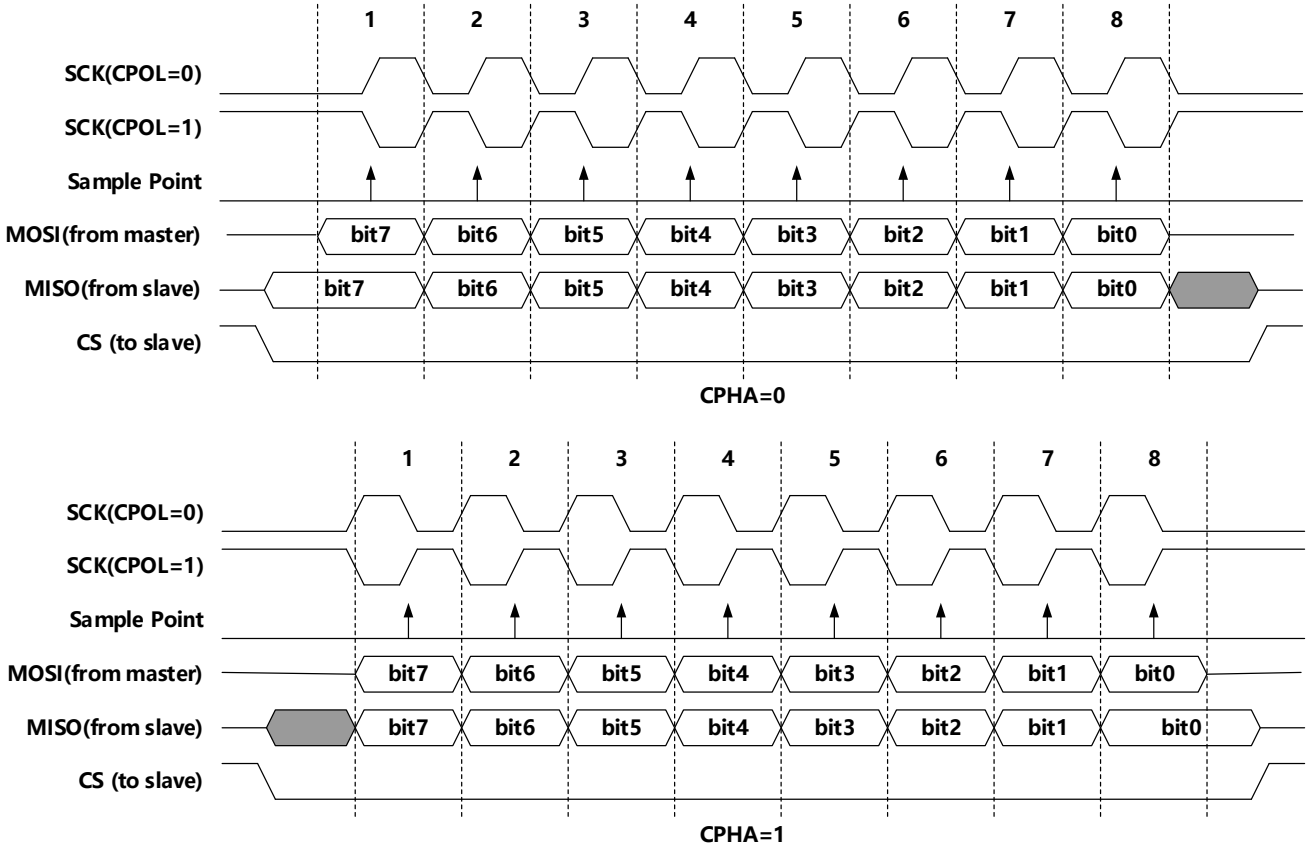
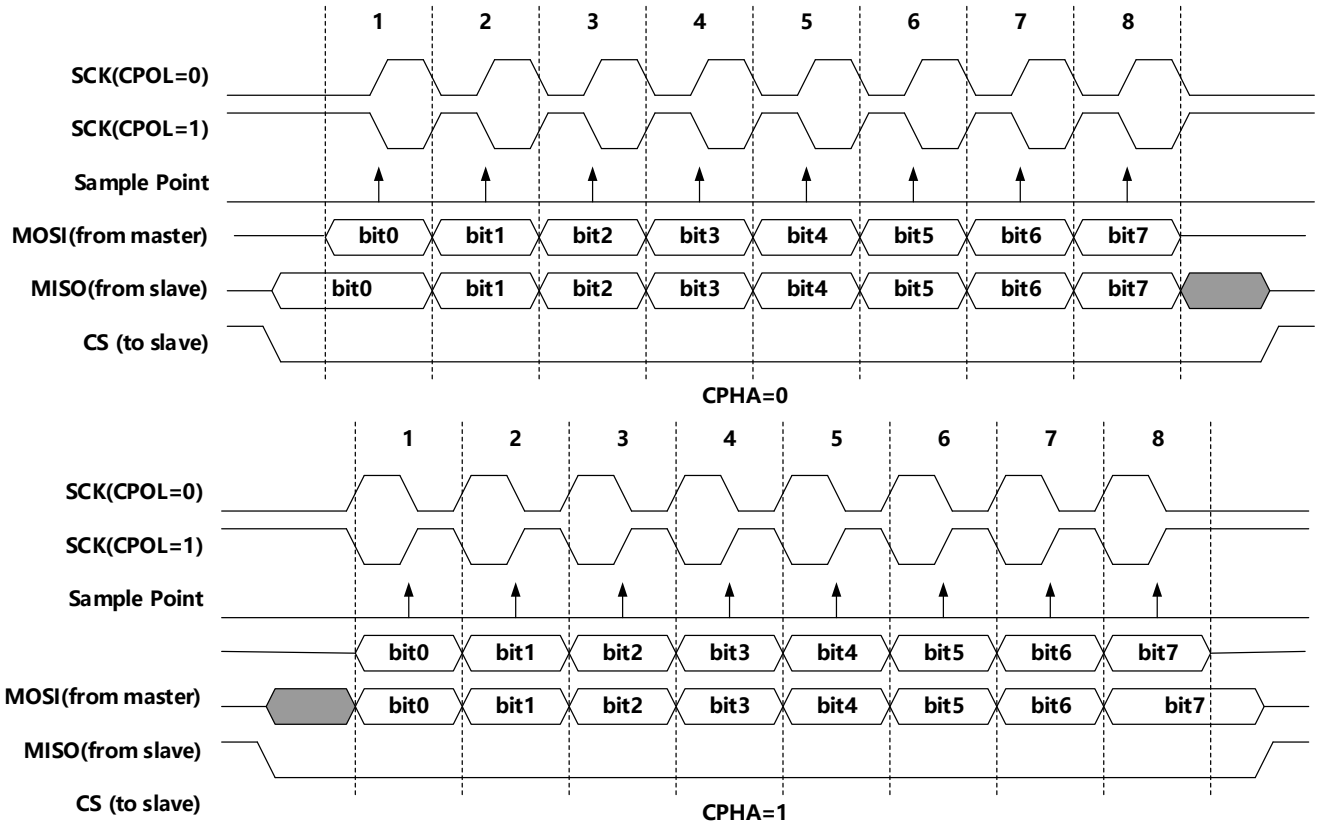


图 23-3 SPI 一线模式时序图（LSB）



23.3.8. 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：发送时自动切换到输出，接收时自动切换到输入。
- IO0：发送时自动切换到输出，接收时自动切换到输入。
- 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 23-4 SPI 二线模式时序图（MSB）

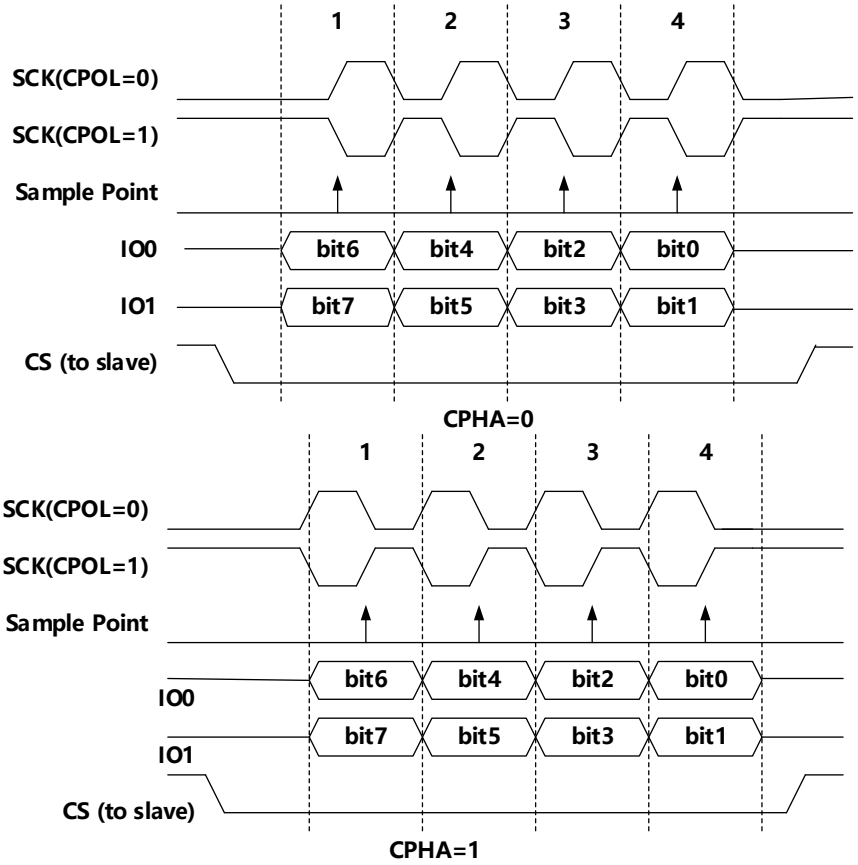
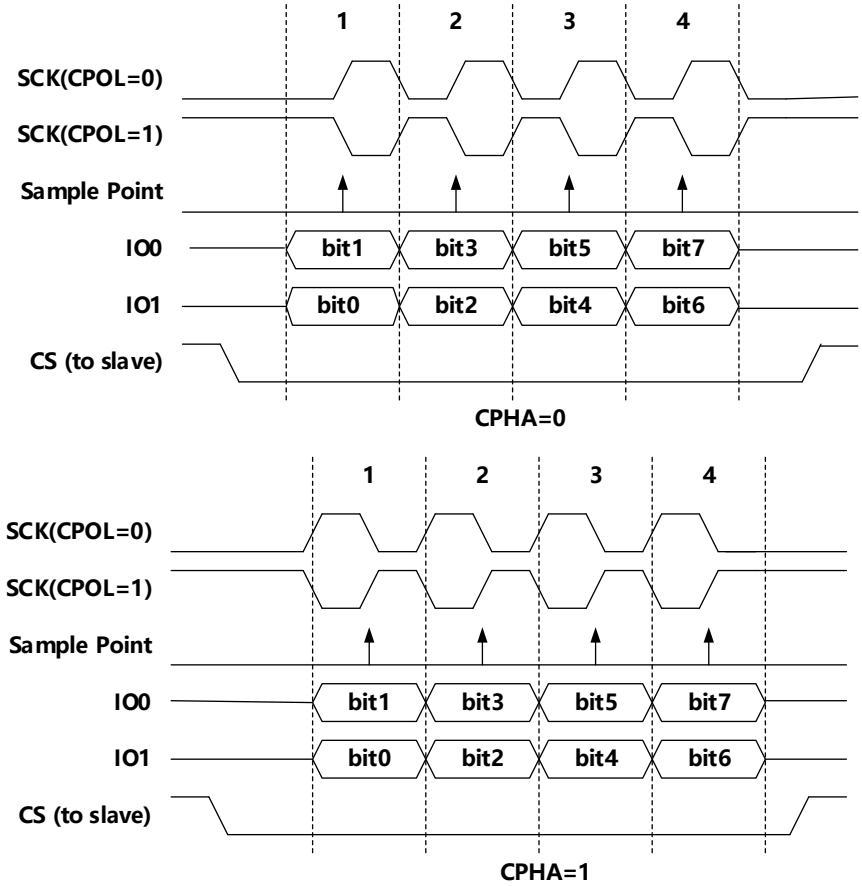


图 23-5 SPI 二线模式时序图（LSB）



23.3.9. 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 23-6 SPI 四线模式时序图（MSB）

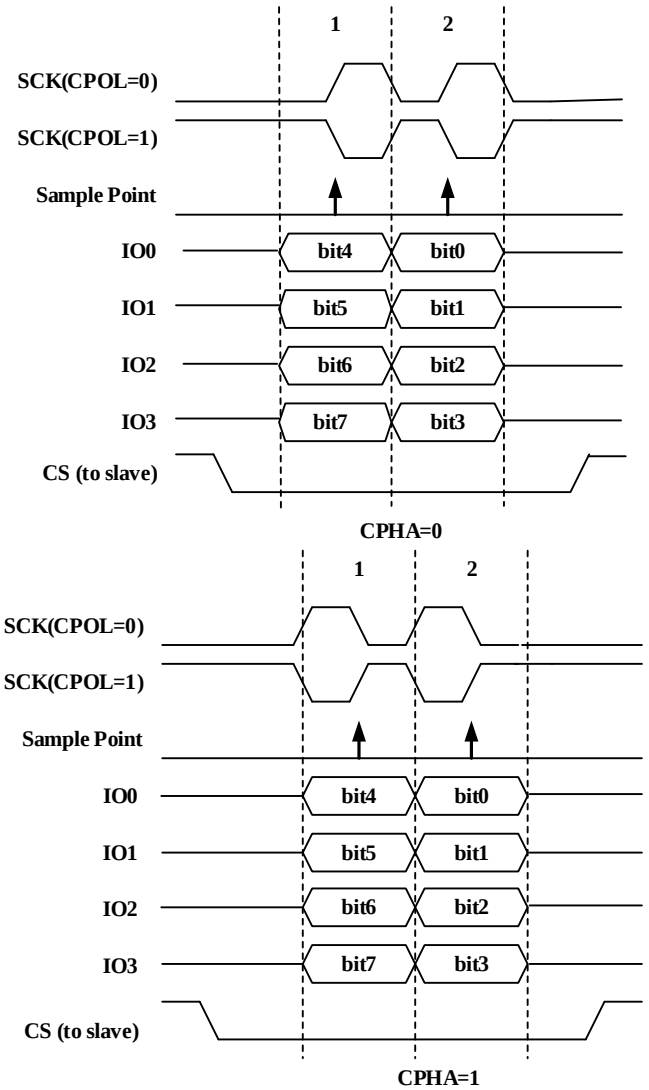
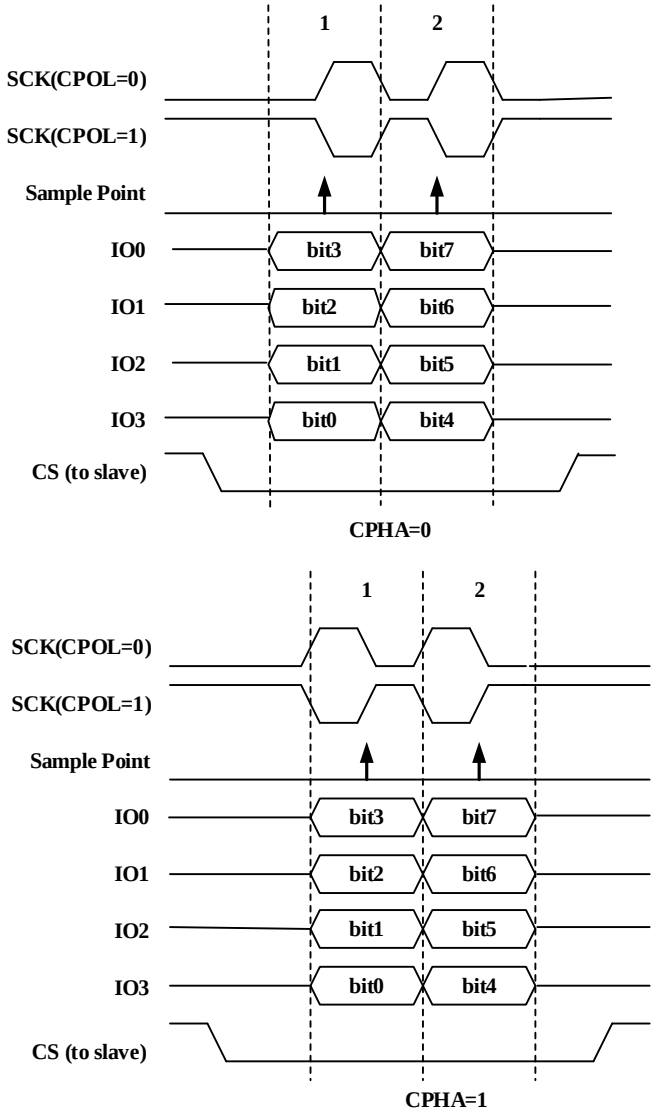


图 23-7 SPI 四线模式时序图（LSB）



23.3.10. 通信类型

■ 全双工

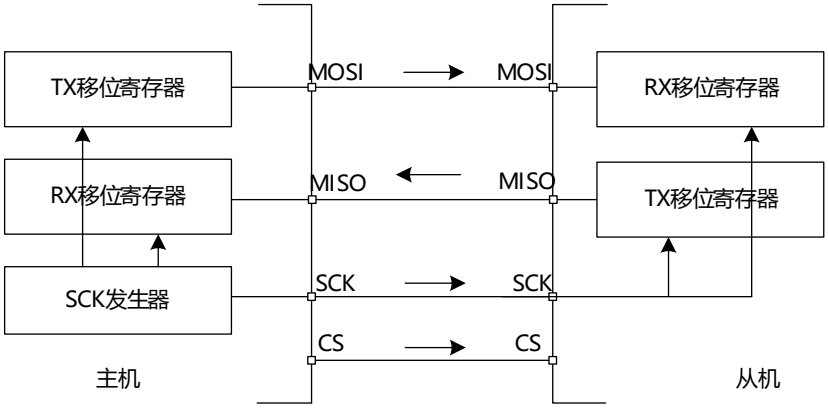
一线模式时，仅 FIFO 模式下 SPI 支持全双工通信，而双线模式、四线模式不支持全双工通信。

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 23-8 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 23-9 半双工主发从收

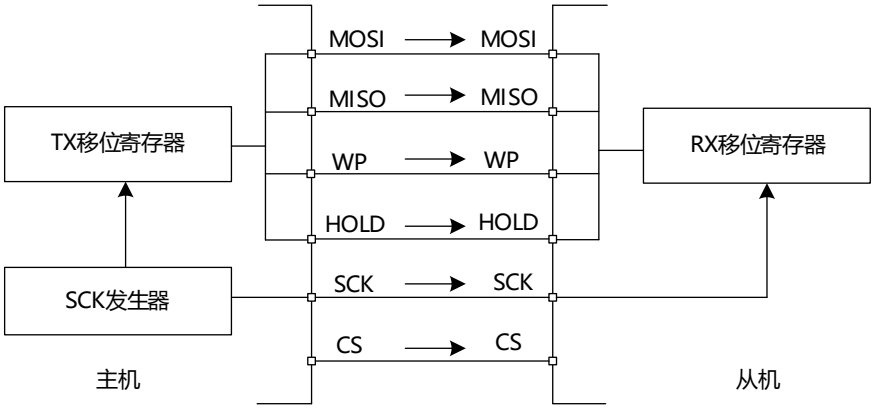
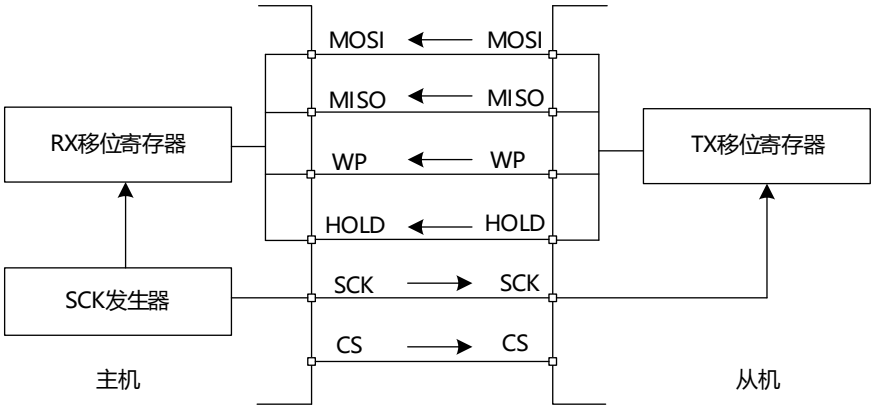


图 23-10 半双工主收从发



23.3.11. 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。

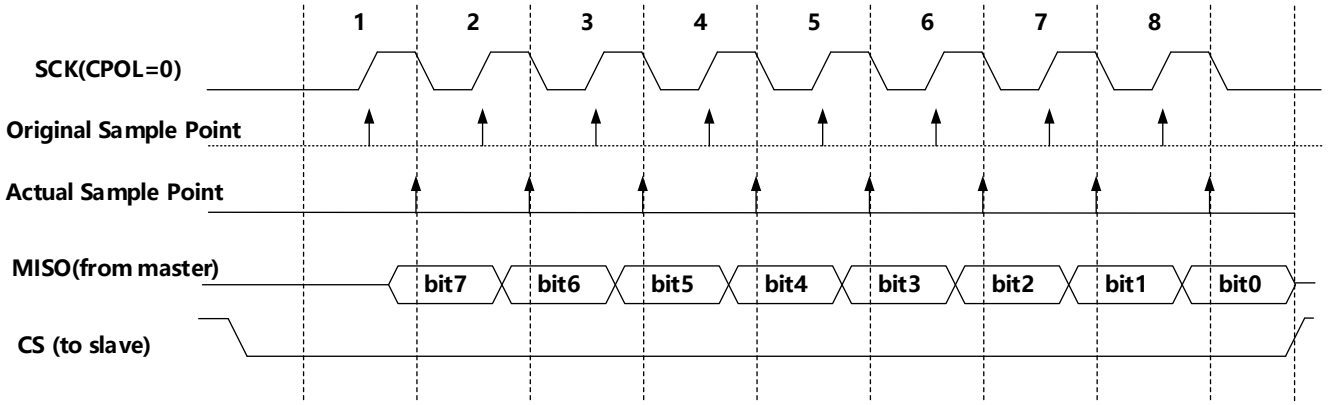
该功能也可实现一主多从控制，利用软件进行选择从机，从而进行数据传输。

23.3.12. 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。下图为二分频且 RX_CTL. SSHIFT 设置为 1 时的时序图。

图 23-11 移位采样示意图（CPOL=0，CPHA=0，MSB）



23.3.13. 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。

23.3.14. 传输等待功能

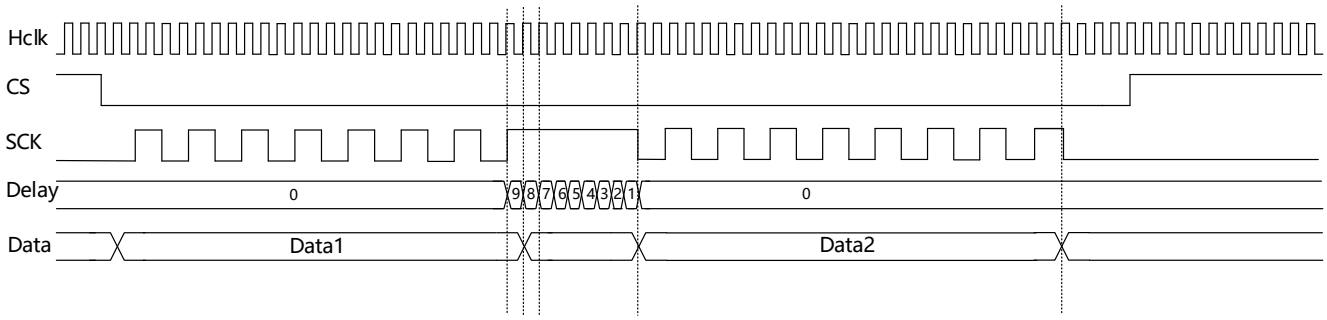
该功能仅在主模式下由 SPI_TXDelay 寄存器来决定，目的是为了在不降低 SCK 频率的条件下，降低 SPI 传输的速率。当通过 DMA 来传输数据或 CPU 与 SPI FIFO 之间数据交换速率较快时，SCK 处于连续不间断发送，速率较快，当设置 SPI_TXDelay 的值，将会有一个 32 位的计数器在每一字节传输结束后开始计数，当到达设

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 23-12 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；

23.3.15. 批量传输

该 SPI 模块仅支持批量传输，无法实现无限数据的收发，因此，在主或从收发开始前（CS 拉低前），需要先配置 SPI_BATCH 寄存器，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。

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。

23.3.16. SPI 中断和状态

表 23-3 SPI 中断

中断和状态	中断使能位	描述
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。
TX_BUSY	无	SPI 忙于发送标志位。

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。

- TX_BUSY：

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 SPI 主或从发送数据时的状态。

在从模式下，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不发送完成后，该位清零；

在主模式下，SPI_BATCH 设定后，通过 SPI_CS 寄存器启动 SPI 发送后，该位置一，当设定 SPI_BATCH 的数据量发送完成后该位清零。

如果软件要关闭 SPI 或 SPI 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。

- 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。

- 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。

23.3.17. SPI 的 DMA 传输

仅在间接模式下，SPI 的 TX 和 RX 支持 DMA 功能，有其对应的 DMA 请求号。

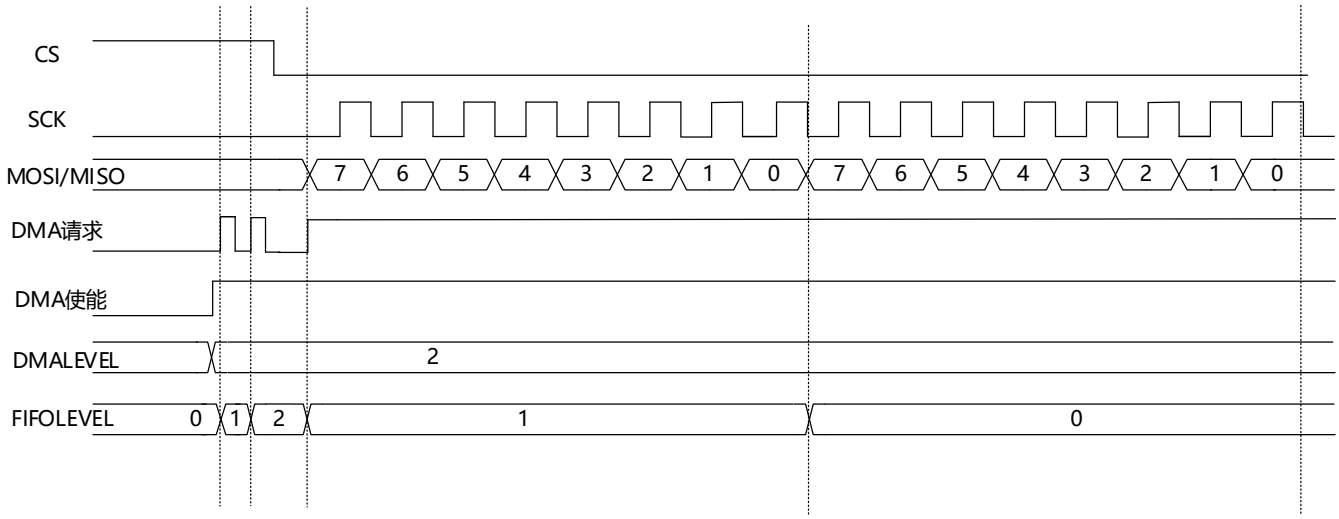
设置 SPI_TX_CTL.TX_DMA_REQ_EN 位使能 SPI 的 DMA 发送，设置 SPI_RX_CTL.RX_DMA_REQ_EN 位使能 SPI 的 DMA 接收。设置 SPI_TX_CTL.TX_DMA_Level 位域，值为 TX DMA 请求 level。当 TX FIFO 中的数据小于等于此值时，TX DMA 请求有效。设置 SPI_RX_CTL.RX_DMA_Level 位域，值为 RX DMA 请求 level。当 RX FIFO 中的数据大于等于此值时，RX DMA 请求有效。

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 23-13 SPI TX_DMA 请求传输示意图



23.3.18. 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。

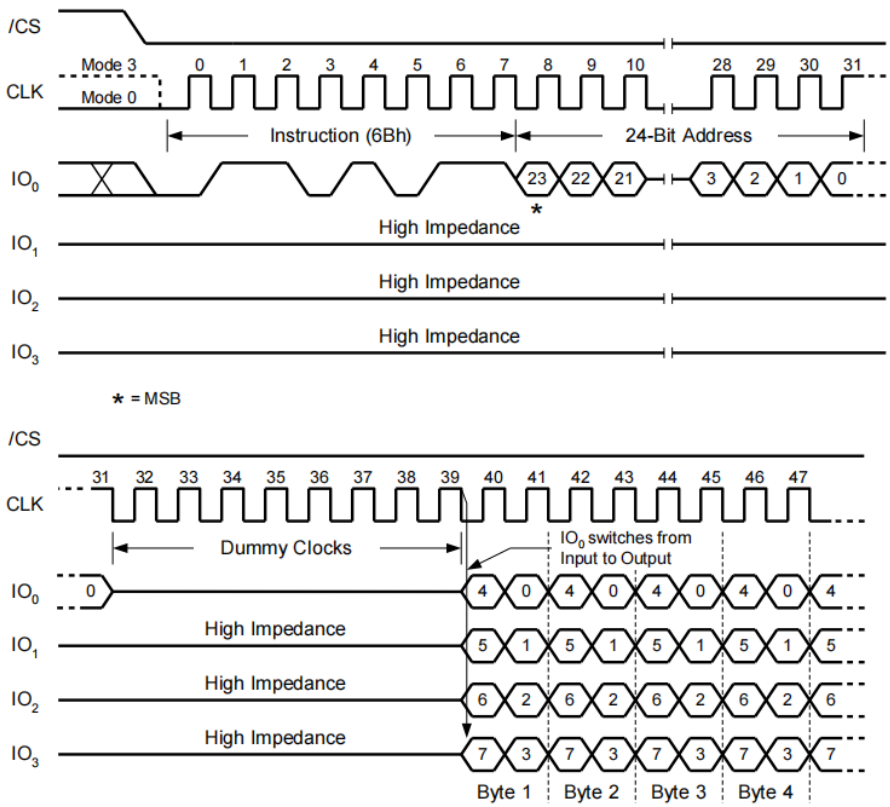
SPI 工作在从模式，当仅处于接收模式下，MISO 数据线的状态由 Dummy 控制位决定。SPI 引擎会将 Dummy 数据按比特移出。

23.3.19. 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、参数和数据这几个阶段。 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。

图 23-14 SPI 访问存储器命令序列



● 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，仅支持一线发送。指定执行操作的类型。

● 地址阶段

在地址阶段，将 1-3 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，仅支持一线发送。

相应关系见下表：

表 23-4 地址宽度

Add_width[4: 0]	地址宽度
0x08	8bit
0x10	16bit
0x18	24bit
others	24bit

● 参数阶段

在 FIFO 模式下待发送的参数字节数写入 FIFO 中进行发送。在内存映射模式下，由 SPI_MEMO_ACC（Para_Ord1、Para_Ord2、PARA_NO1、PARA_NO2）寄存器控制，可在地址前或地址后发送，发送内容由写入的 SPI_PARA 寄存器定义，仅支持一线发送。

● 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_CTL.X_Mode 字段进行配置。

23.3.20. 内存映射模式

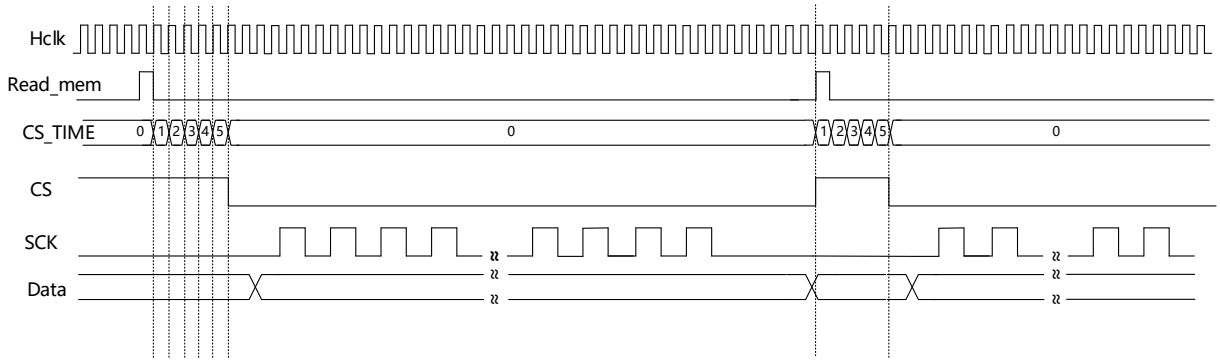
该 SPI 接口模块支持对满足 SPI 协议的存储器的快速读写功能，支持一线、二线、四线数据读写。当该功能开启时 MCU 可以如读取普通存储器一样通过 AHB 总线直接快速读取支持 SPI 协议的 sram 或者 nor_flash，其通过 SPI 接口当做 MCU 的附属存储器并映射到指定地址。该模式为内存映射模式。

内存映射模式的控制位由 SPI_MEMO_ACC 寄存器给出，SPI_CMD/SPI_PARA 为快速读写时所需的发送的参数。根据 SPI 访问存储器命令序列，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 模式有一部分共用控制寄存器，在 SPI_CTL 给出。

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。

图 23-15 内存映射模式 SPI 传输过程



SPI_CTL.X_Mode 位在内存映射模式下控制存储器读写序列数据阶段的传输模式，设置为一时，数据以一线方式传输；设置为二时，以二线模式传输数据；设置为三时，以四线模式传输数据。

SPI_CTL.LSB/SPI_CTL.CPHA/SPI_CTL.CPOL 这三位内存映射模式与间接模式共用，其用法与间接模式一致。

23.3.20.1. 参数配置

SPI_PARA 寄存器中的 Para1/Para2 存放的是可能使用的地址/Dummy 或命令参数。能够作为地址字段，能够作为 Dummy 字段，也能作为参数字段。可灵活的为存储器提供对应的时序。

发送时序由 SPI_MEMO_ACC 中的 Para_Ord1、Para_Ord2、PARA_NO1、PARA_NO2 寄存器设置地址、参数长度及发送次序，两组参数 Para1/Para2 均可配置在地址前或地址后进行发送，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。

如下图存储器时序为例，发送地址为 32bit，内存映射模式下自动译码 AHB 总线地址只有 24bit，因此需要利用参数发送 8bit 地址，作为 Address[31:24]发送序列，配置如下：

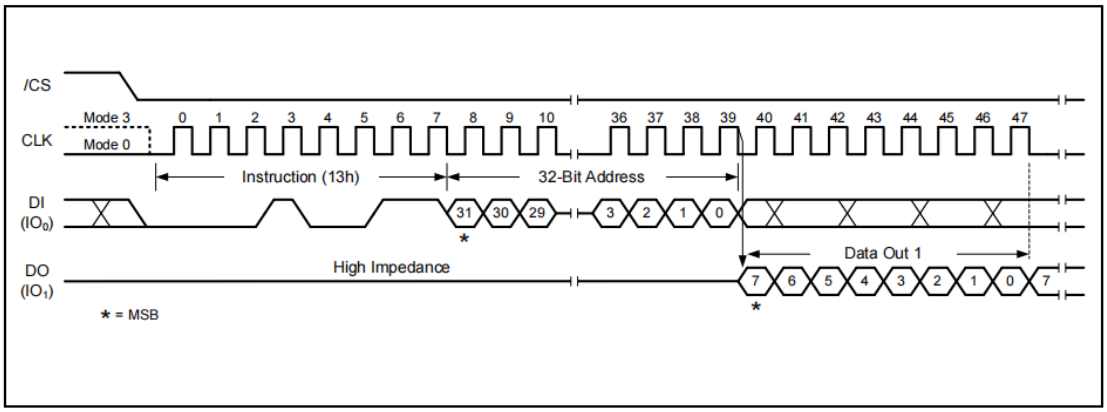
1）SPI_PARA.Para1 写入 0x0。

2）SPI_MEMO_ACC.Para_Ord1 配置为 0x0

3）SPI_MEMO_ACC.PARA_NO1 配置为 0x08

再按照内存映射模式操作流程进行通信即可。

图 23-16 内存映射模式命令序列



23.3.20.2. 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。

23.3.20.3. 读连续模式

启动连读使能位即 SPI_MEMO_ACC 寄存器的 CON_RD_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以及地址，直接读取数据，可提高数据读取速率。连读的条件为：

- 读取地址为连续值，即没有地址跳变。
- 没有进行过写操作。

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。

23.4. 配置流程

23.4.1. 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，直到 BATCH 个数据全部发完

5）等待 SPI 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 SPI 数据全部发送完成

7）清除 SPI_TX_CTRL 中的 TX_EN 位

8）清除 SPI 从设备选择寄存器，结束发送

23.4.2. 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，直到 BATCH 个数据全部收完

5）等待 SPI 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 SPI 数据全部接收完成

7）清除 SPI_RX_CTRL 中的 RX_EN 位

8）清除 SPI 从设备选择寄存器，结束接收

23.4.3. 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，直到 BATCH 个数据全部发完

6）清除 SPI_TX_CTRL 中的 TX_EN 位

7）结束发送

23.4.4. 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，直到 BATCH 个数据全部发完

5）清除 SPI_RX_CTRL 中的 RX_EN 位

6）结束发送

23.4.5. SPI 存储器内存映射模式读取

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 SPI 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 SPI 接口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）设置 SPI_CTL 的 X_mode 位

4）通过 SPI_BAUD 设置时钟波特率

5）在 SPI_CMD 寄存器的 Rd_Cmd 位中写入需要发送的读命令

6）SPI_PARA 寄存器的 Para1（2）位设置需要发送的参数 1（2）（如果需要）

7）通过 SPI_MOME_ACC 寄存器 Addr_width/ PARA_No1（2）/ Para_Ord1（2）设置地址、参数长度及发送次序

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

23.4.6. SPI SRAM 内存映射模式写入

SPI 接口模块支持对满足 SPI 协议的 sram 的写功能，AHB 可以直接对 sram 地址进行写操作，硬件会自动开启对 SPI sram 的通信，在通信完成后 SPI 接口会将值写入 sram。其使用方法如下：

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）设置 SPI_CTL 的 X_mode 位

4）通过 SPI_BAUD 设置时钟波特率

5）通过 SPI_CMD 寄存器的 Wr_Cmd 位设置需要发送的写命令

6）SPI_PARA 寄存器的 Para1（2）位设置需要发送的参数 1（2）（如果需要）

7）通过 SPI_MOME_ACC 寄存器 Addr_width/ PARA_No1（2）/ Para_Ord1（2）设置地址、参数长度及发送次序

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

23.5. SPI 寄存器描述

23.5.1. 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

SPI3 内存映射模式：0x90000000~0x9FFFFFFF

偏移	名称	描述
0x00	SPI_DAT	数据寄存器
0x04	SPI_BAUD	波特率设置寄存器
0x08	SPI_CTL	控制寄存器
0x0C	SPI_TX_CTL	发送控制寄存器
0x10	SPI_RX_CTL	接收控制寄存器
0x14	SPI_IE	中断控制寄存器
0x18	SPI_STATUS	状态寄存器
0x1C	SPI_TX_DELAY	发送等待寄存器
0x20	SPI_BATCH	批量数据个数寄存器
0x24	SPI_CS	从设备选择寄存器
0x28	SPI_OUT_EN	管脚输出使能
0x2C	SPI_MEMO_ACC	取值控制寄存器
0x30	SPI_CMD	取值命令寄存器
0x34	SPI_PARA	取值参数寄存器

23.5.2. 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

23.5.3. 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))。

23.5.4. 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	CRYPT_EN	RW	0x0	加解密使能位 1：使能 0：不使能
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

23.5.5. 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

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

23.5.6. 接收控制寄存器(SPI_RX_CTL: 10h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留
9:8	SSHIFT	RW	0x0	主机采样移位控制位 00：不发生移位 01：主机推迟 1 个 hclk 周期开始采集数据 10：主机推迟 2 个 hclk 周期开始采集数据 11：主机推迟 3 个 hclk 周期开始采集数据 注：spi 波特率设置为二分频时，在 FIFO 模式下，不能使用采样移位功能。取指模式下不支持采样移位功能。
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

23.5.7. 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

23.5.8. 状态寄存器(SPI_STATUS: 18h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
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	TX_BUSY	RO	0	SPI 忙于发送标志位 1：作从机时，SPI 正在发送数据或送状态下发送 FIFO 不为空；作主机时，SPI 主机正在发送数据 0：SPI 空闲 硬件清 0 和置 1

23.5.9. 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

23.5.10. 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
31:20	RSV	-	-	保留
19:0	BATCH_NUMBER	RW	0x0	该寄存器用来存储 SPI 总线上即将传输的数据字节个数

23.5.11. 从设备选择寄存器(SPI_CS: 24h)

位域	名称	属性	复位值	描述
31:1	RSV	-	-	保留

0	SPI_CS	WO	0x00	置 1 将使设备选择信号 SPI_CS 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，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。 读此位，反映 CS[0]管脚的状态。

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。

23.5.12. 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

23.5.13. SPIA 取值控制寄存器(SPI_MEMO_ACC: 2Ch)

位域	名称	属性	复位值	描述
31:19	RSV	-	-	保留位
18:14	ADDR_WIDTH	RW	0x10	发送地址的位数等于该寄存器值 地址从 AHB 地址总线 LSB 向上取值 最大值为 24，超过将会按 24 进行后续计算 0x08 : 8 bit 0x10 : 16bit 0x18 : 24bit 其他非 0：24bit

13:9	PARA_NO2	RW	0x0	发送参数 2 的位数等于该寄存器值 参数从 PARA2 的 LSB 向上取值 0 表示不使用参数 2，最大值为 16，超过将会按 16 进行后续计算 0x08 : 8 bit 0x10 : 16bit 0x00：不使能 其他非 0：16bit
8:5	PARA_NO1	RW	0x0	发送参数 1 的位数等于该寄存器值 参数从 PARA1 的 LSB 向上取值 0 表示不使用参数 1，最大值为 8，超过将会按 8 进行后续计算 0x08：8 bit 0x00：不使能 其他非 0：8bit
4	RVS	-	-	保留
3	CON_RD_EN	RW	0x0	连读使能位 1:使能连读 0:不使能连读
2	PARA_ORD2	RW	0x0	决定在发送地址前后加入参数 2 1:在地址后发送 0:在地址前发送
1	PARA_ORD1	RW	0x0	决定在发送地址前后加入参数 1 1:在地址后发送 0:在地址前发送
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

23.5.14. SPIA 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

23.5.15. SPIA 取值参数寄存器(SPI_PARA: 34h)

位域	名称	属性	复位值	描述
31:24	RSV	-	-	保留位
23:8	PARA2	RW	0x0	存放可能使用的地址 Dummy 或命令参数
7:0	PARA1	RW	0x0	存放可能使用的地址 Dummy 或命令参数

PS：内存映射模式多线模式下，命令、地址、参数，仍然以一线模式传输。
参数 1 和参数 2 如果需要同时发送，先发送参数 1，后发送参数 2。

24. 集成电路总线（I2C）接口

24.1. 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。

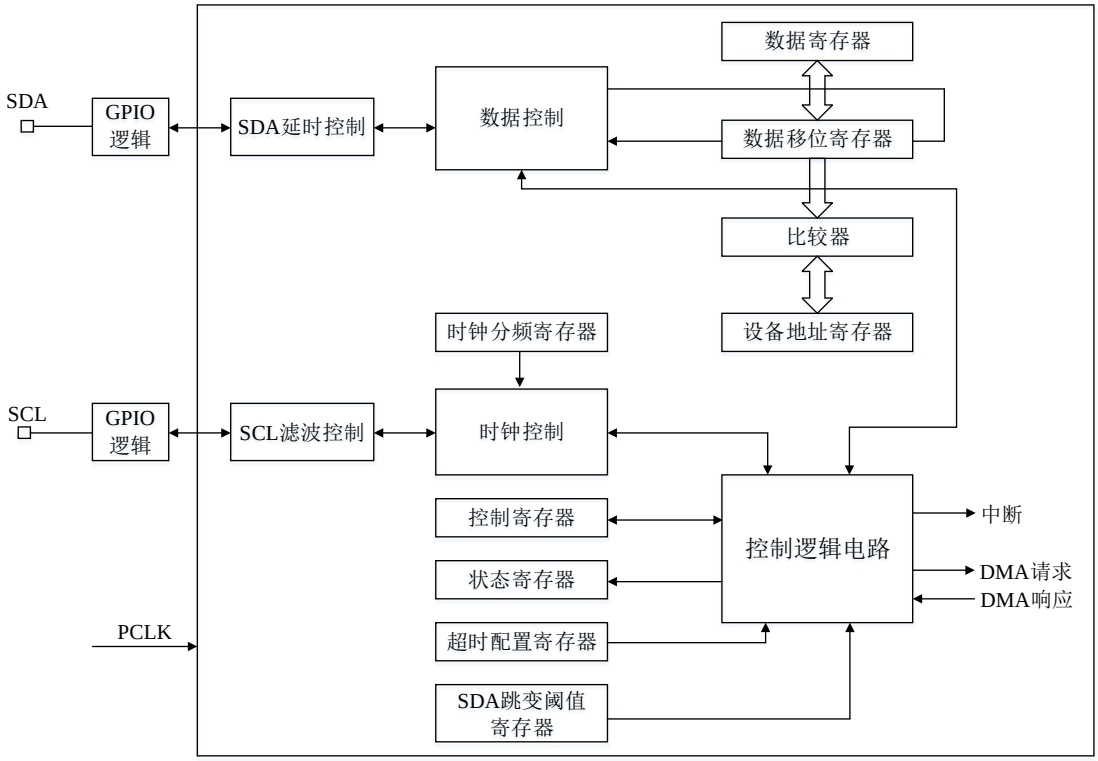
根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

24.2. 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 SMBus 2.0

24.3. 结构框图

图 24-1 I2C 结构框图



24.4. 功能描述

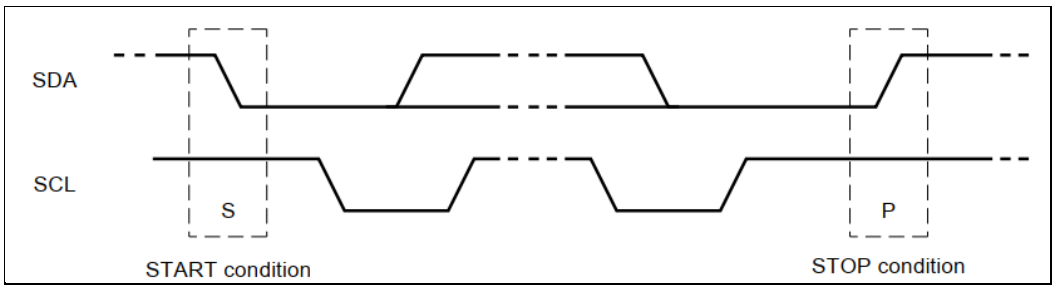
I2C 模块将数据从串行转换成并行（接收），或并行转换成串行（发送）。I2C 模块通过数据引脚 SDA 和时钟引脚 SCL 连接到 I2C 总线。它可以连接到标准（高至 100KHz）、快速（高至 400KHz）或快速增强（高至 1MHz）的 I2C 总线上。

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。

24.4.1. 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（参见图 22-2）。

图 24-2 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。

24.4.2. 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。

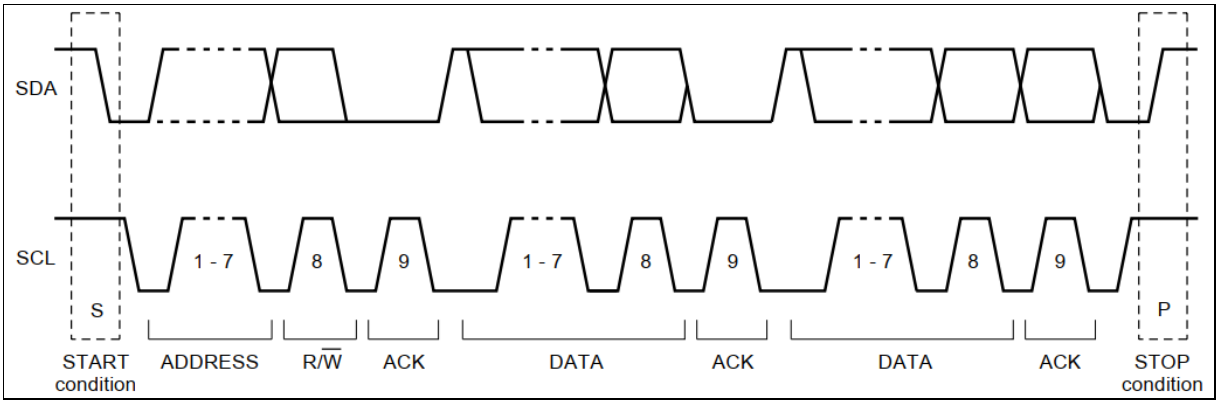
主模式时，I2C 接口启动数据传输并产生时钟信号，并可以发出停止条件信号停止传输。

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 24-3 I2C 总线传输

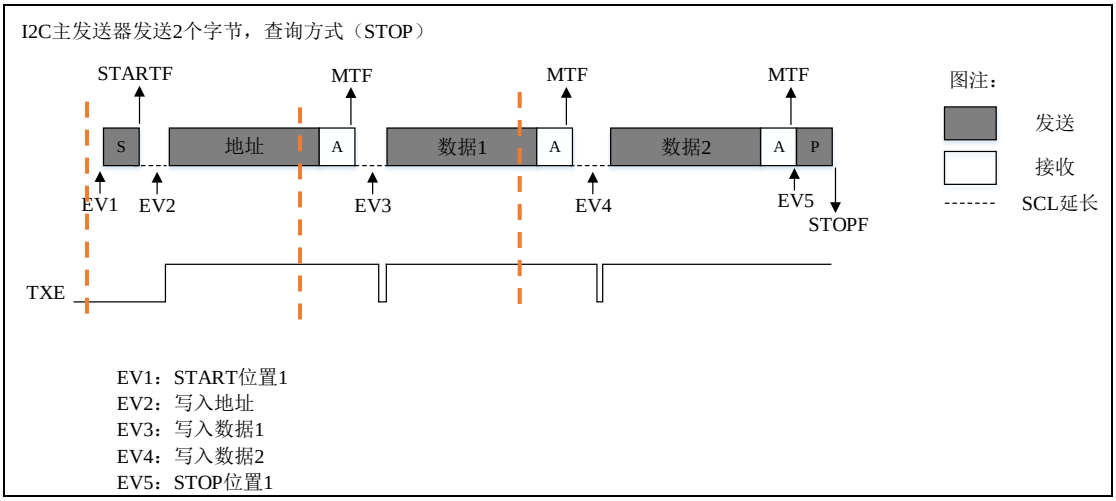


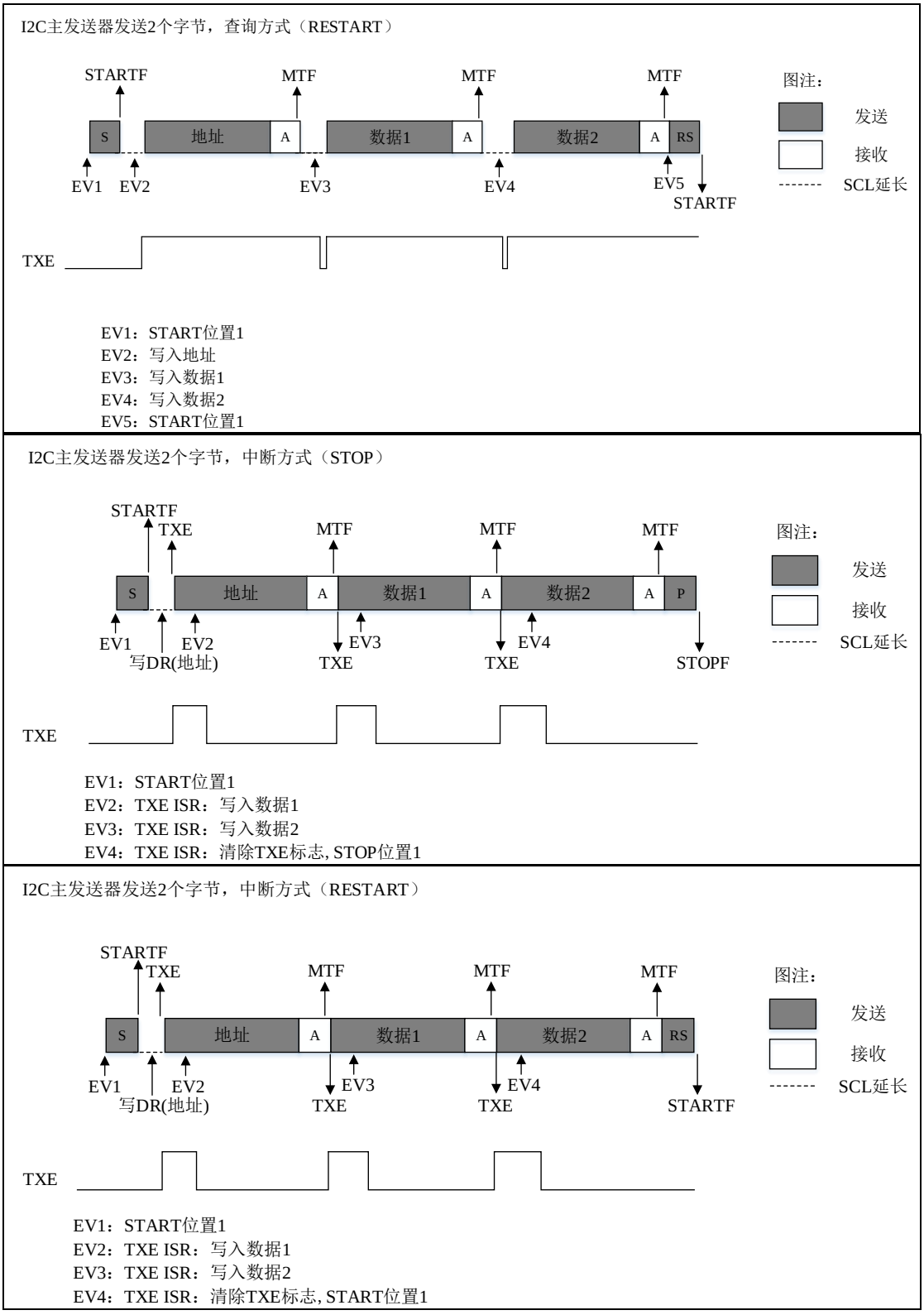
24.4.3. 从机地址 SLAVE_ADDR1/2/3

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。

24.4.4. 主模式发送

图 24-4 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。如过如果有 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 DR 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 24-5 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。

若当前字节传输完成，不去读取数据寄存器清除 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。

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。

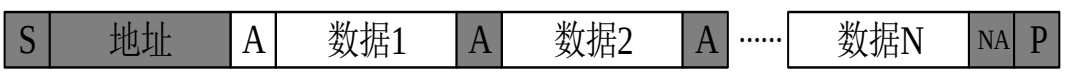
主机针对自从设备接收的最后一个字节发送 NACK。在接收到此 NACK 之后，从设备会释放对 SCL 和 SDA 线的控制。随后，主设备可发送一个停止位/重复起始位。

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 24-6 7 位主接收器的传送图

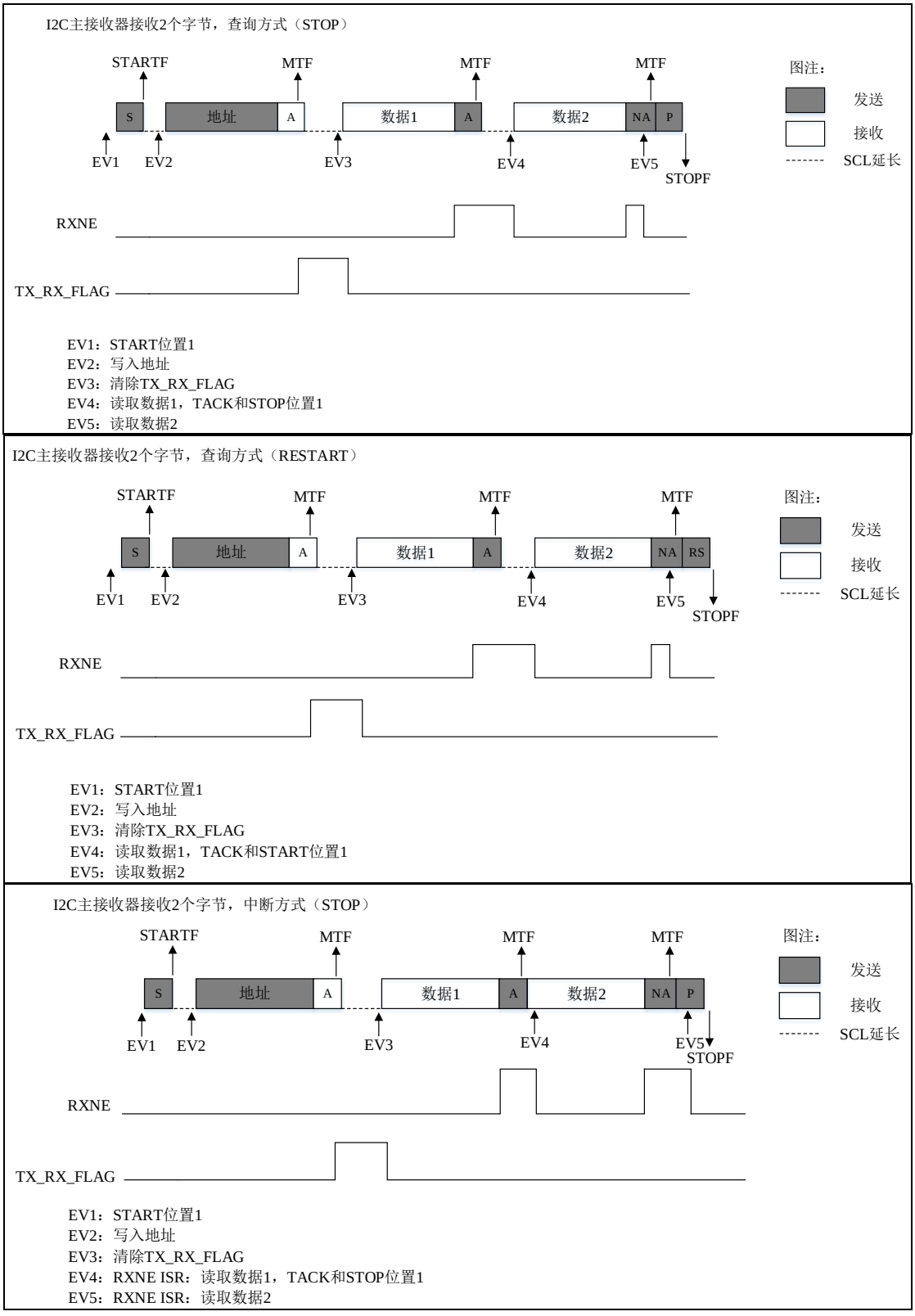


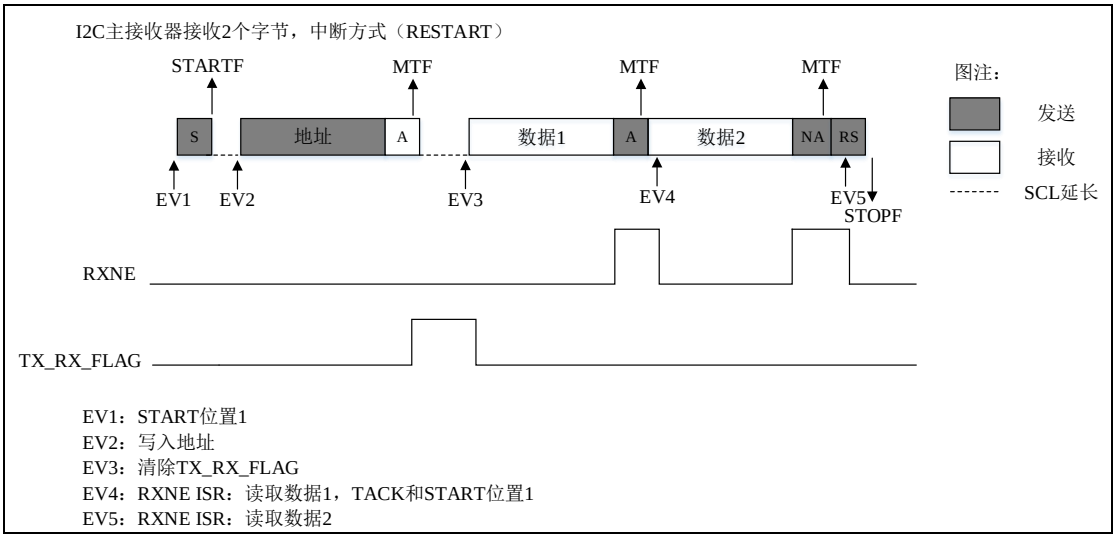
说明：S=Start(起始条件)， P=Stop(停止条件)，A=响应，NA=非响应

:主机到从机 :从机到主机

24.4.5. 主模式接收

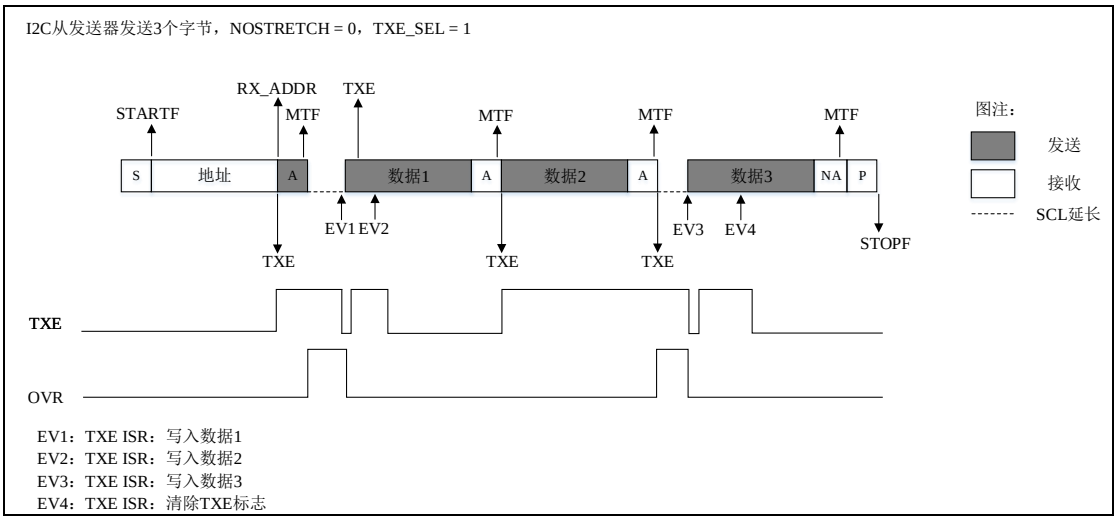
图 24-7 I2C 主模式接收序列图

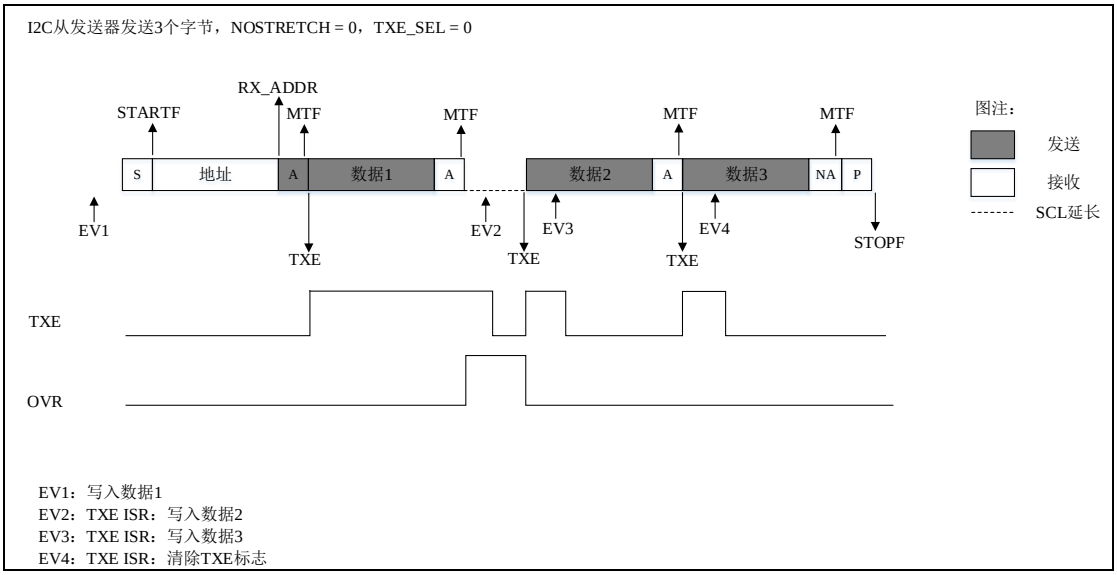


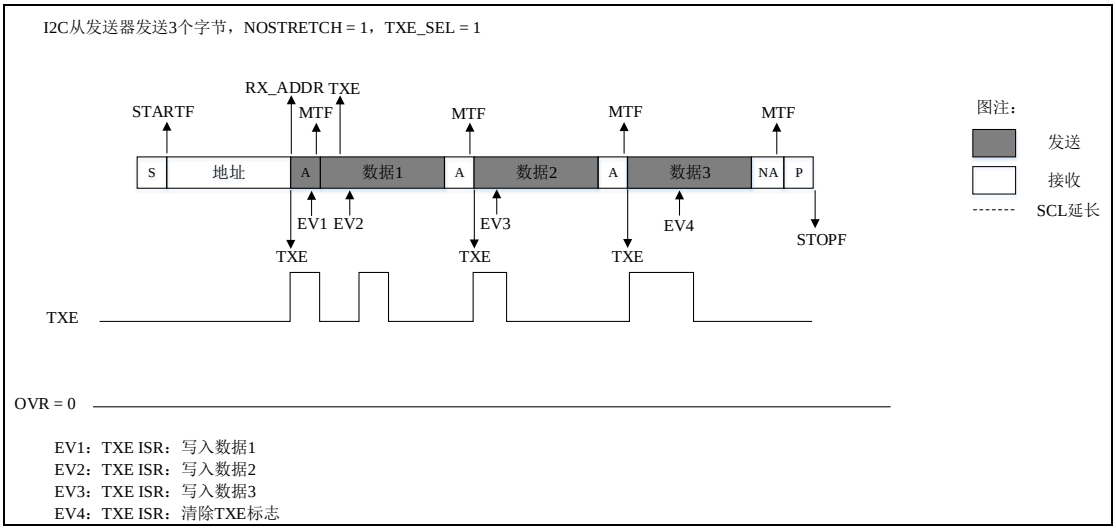


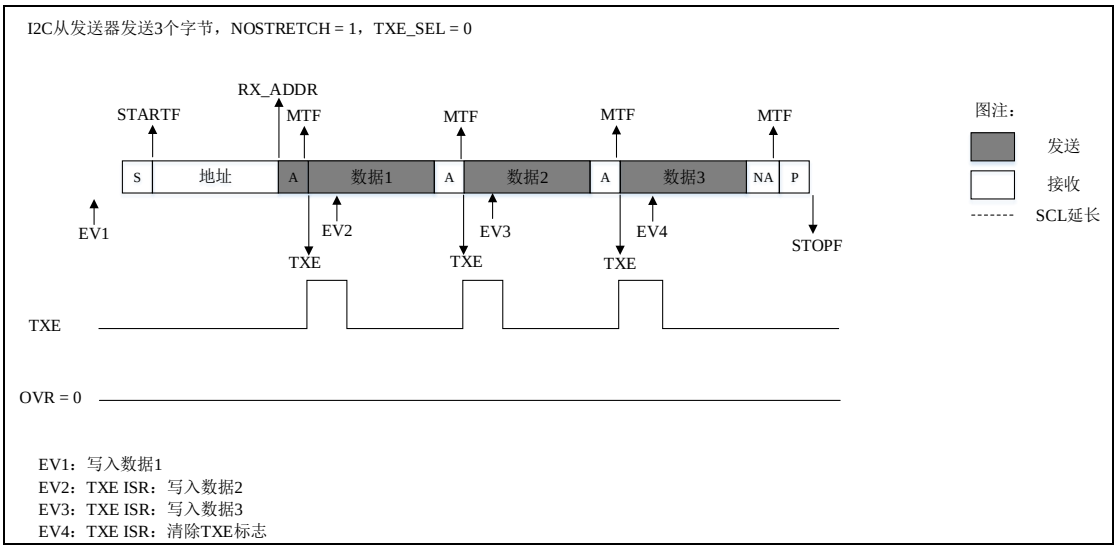
24.4.6. 从模式发送

图 24-8 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，如果在下一个数据发送结束之前新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。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 24-9 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。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 24-10 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

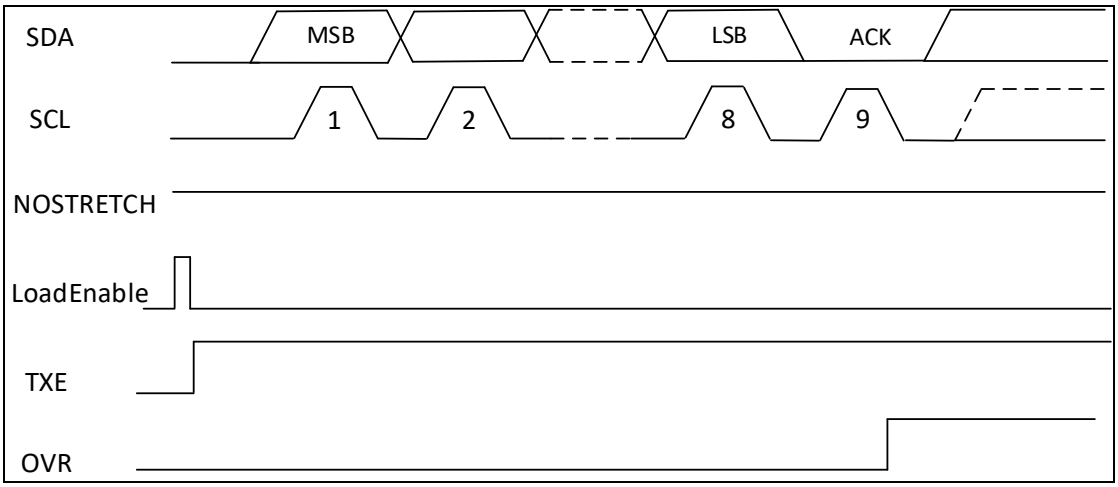
■ 关闭从通信

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。

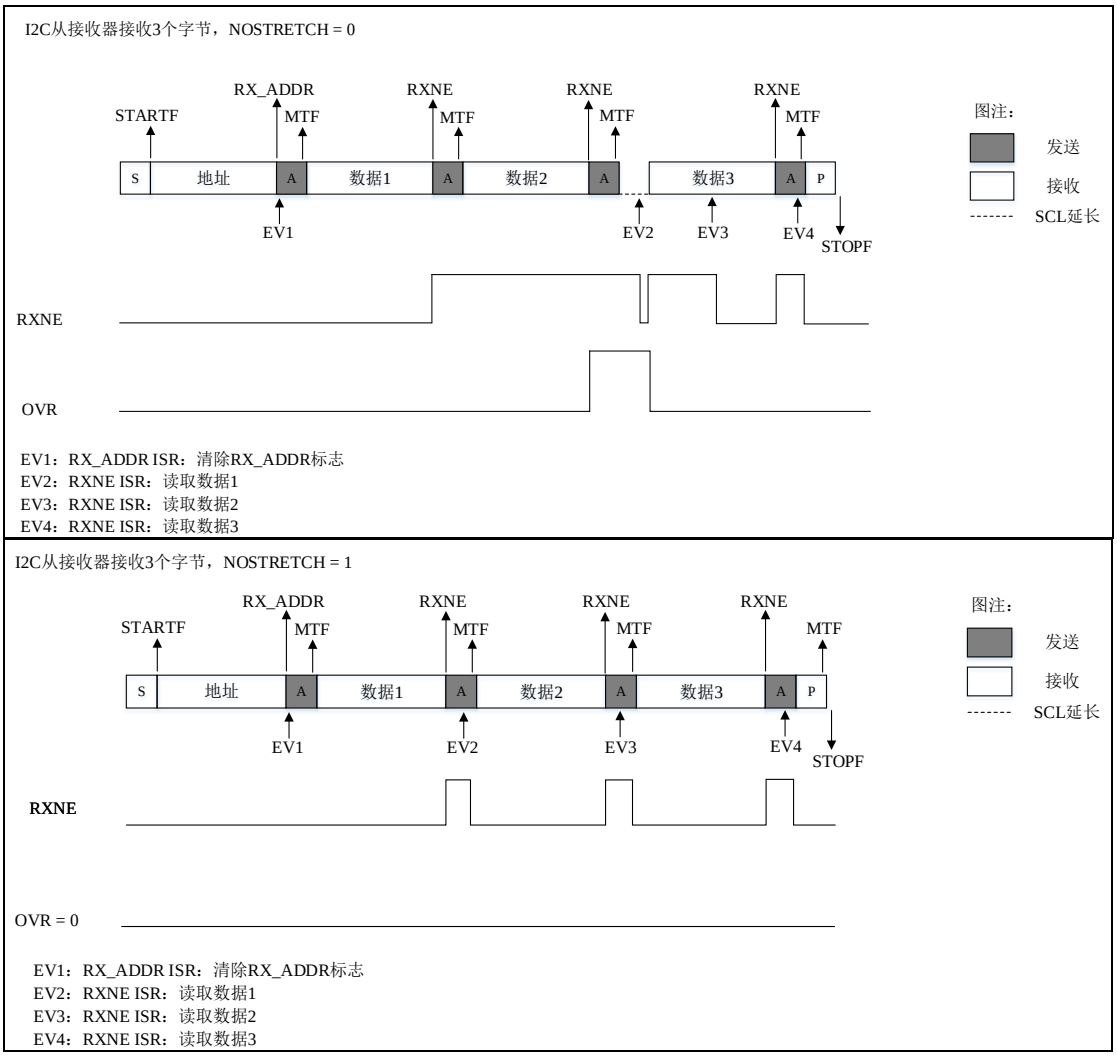
图 24-11 从机发送模式时钟延长时序



虚线部分 SCL 为低，主机无法发送 SCL。

24.4.7. 从模式接收

图 24-12 I2C 从模式接收时序

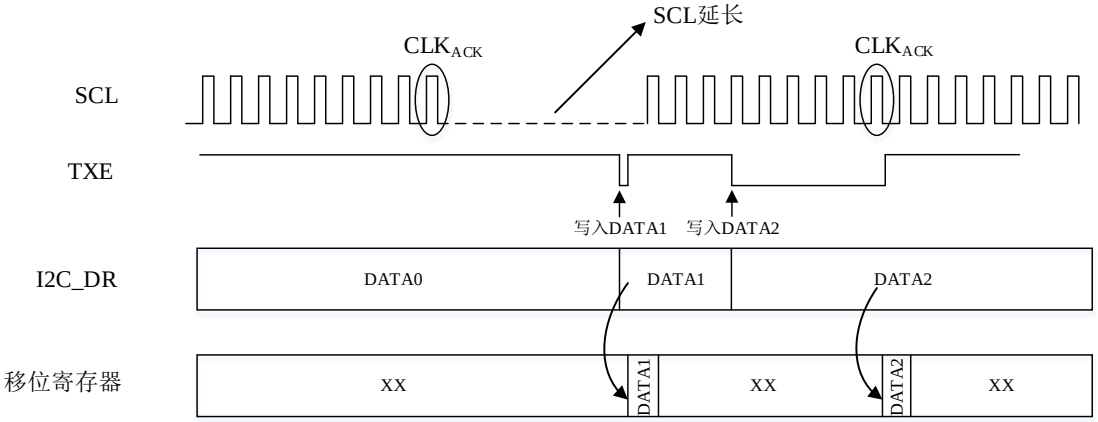


24.4.8. 时钟 SCL 延长

■ 主机主动延长 SCL

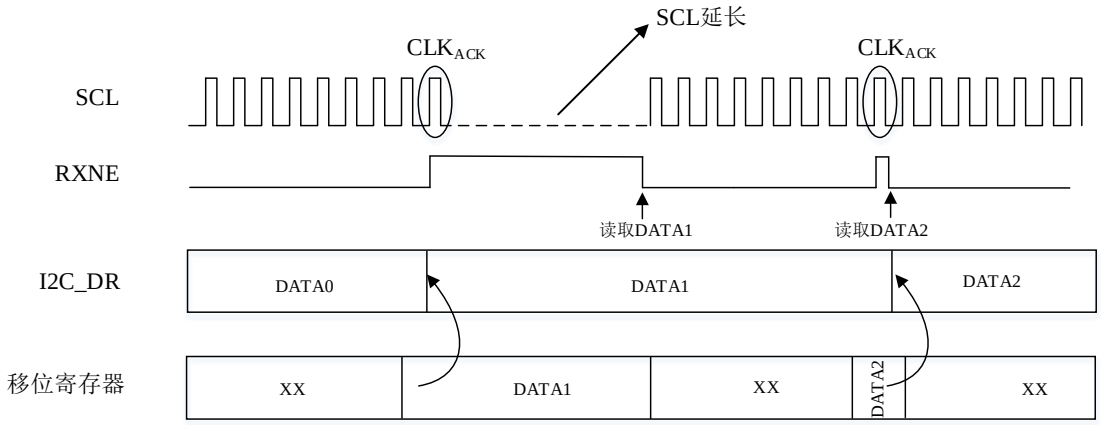
➢ 主发送模式下， 如果在上一次数据传输结束之前 TXE 位已置 1 但数据字节尚未写入数据寄存器，主机会延长 SCL 时钟，直到新的数据被写进数据寄存器。

图 24-13 主机延长 SCL（主机发送模式）



➢ 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。

图 24-14 主机延长 SCL（主机接收模式）

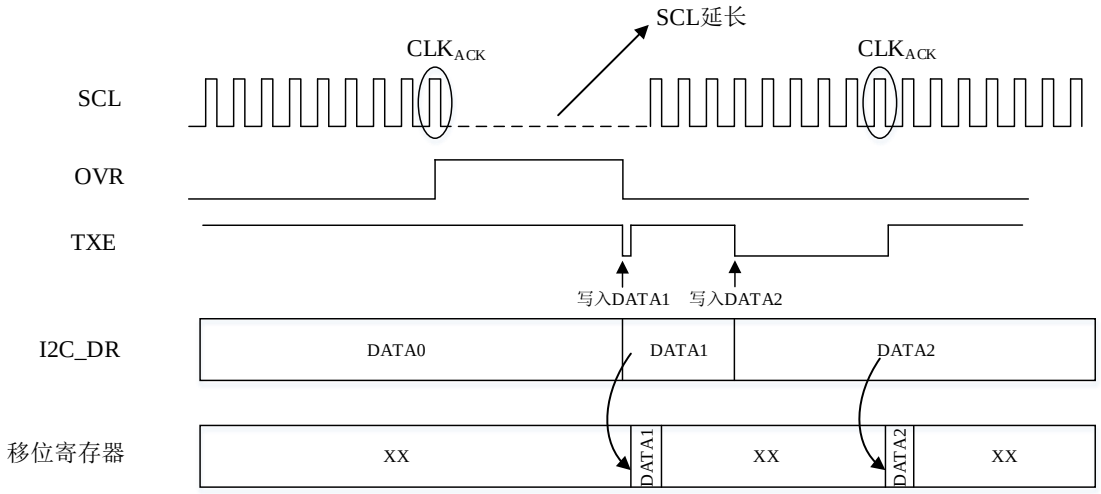


■ SCL 被从机延长(需满足 NOSTRTCH=0):

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。

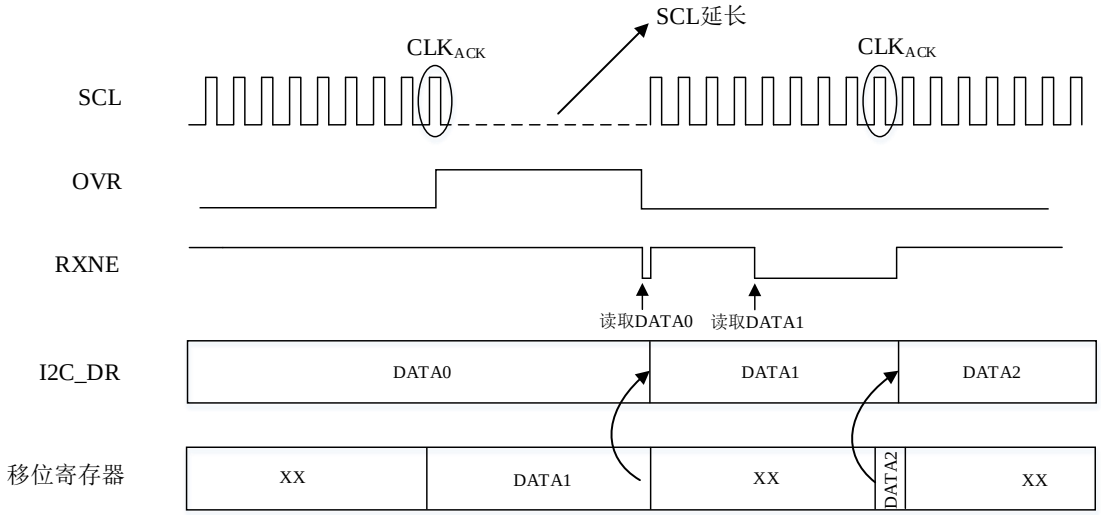
➢ 从机发送模式下，如果在下一次数据传输结束之前 TXE 位已置 1 但数据尚未写入数据寄存器，则 OVR 置 1。此时，如果收到的是 ACK 信号，则从机会拉低并延长 SCL 时钟，直到新的数据被写进数据寄存器。

图 24-15 从机延长 SCL（从机发送模式）



➢ 从机接收模式下，如果在下一次数据接收结束之前 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。

图 24-16 从机延长 SCL（从机接收模式）



24.4.9. 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。

■ 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_IN_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。

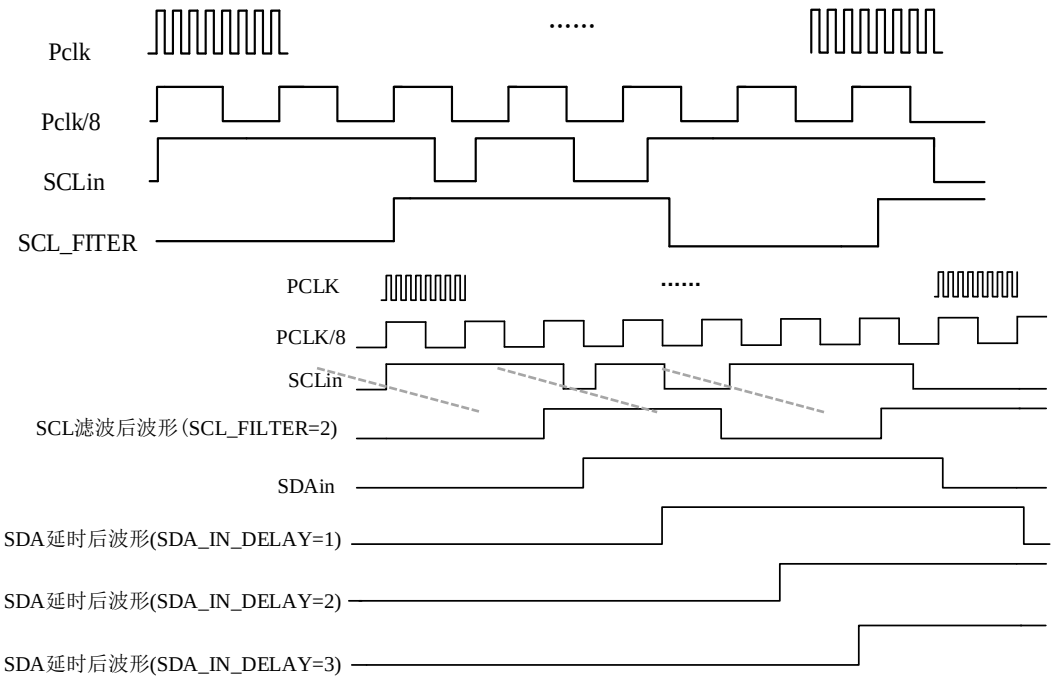
24.4.10. SCL 总线滤波算法

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 24-17 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。

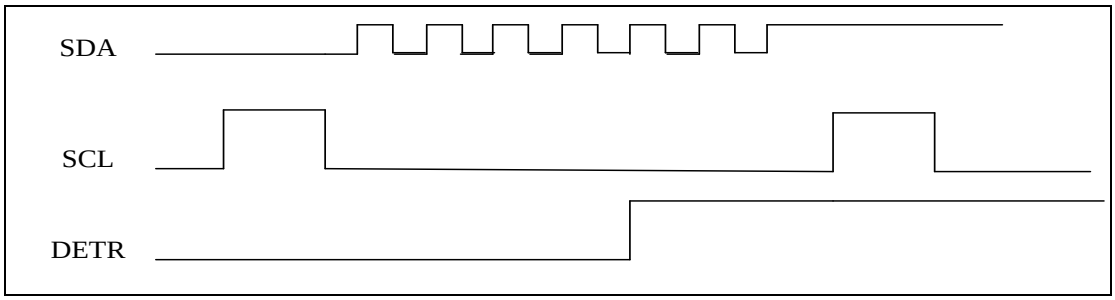
24.4.11. SCL 为低时检测 SDA 的跳变

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：

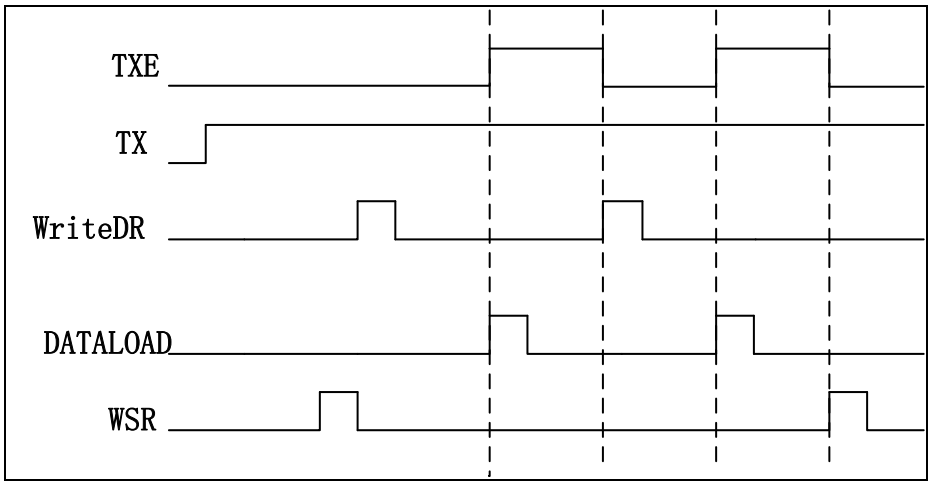
当 I2C_DET=5 时，在 SCL 低电平时，内部计数器开始工作，当检测到 SDA 上升沿跳变累积大于等于 5 时，I2C_SR.DETR=1；当 SCL 变成高电平时，内部计数器清零，等待 SCL 下个低电平到来时再开始计数。如下图所示：

图 24-18 SDA 跳变检测图



24.4.12. TXE 状态

图 24-19 TXE 功能说明图



I2C_SR.TXE 反应了 I2C_DR 数据是否被取走。

在接收模式下，TXE 为 0。

在发送模式下，写 I2C_DR 寄存器，TXE 清 0，发送器取走 I2C_DR 寄存器中的数据，TXE 置 1，TXE 位写 1 清 0；

备注：TX 1 为发送，0 为接收，DATALOAD 加载数据到数据移位寄存器，WriteDR 写 I2C_DR 寄存器，WSR 写状态寄存器。

24.4.13. TXE_SEL

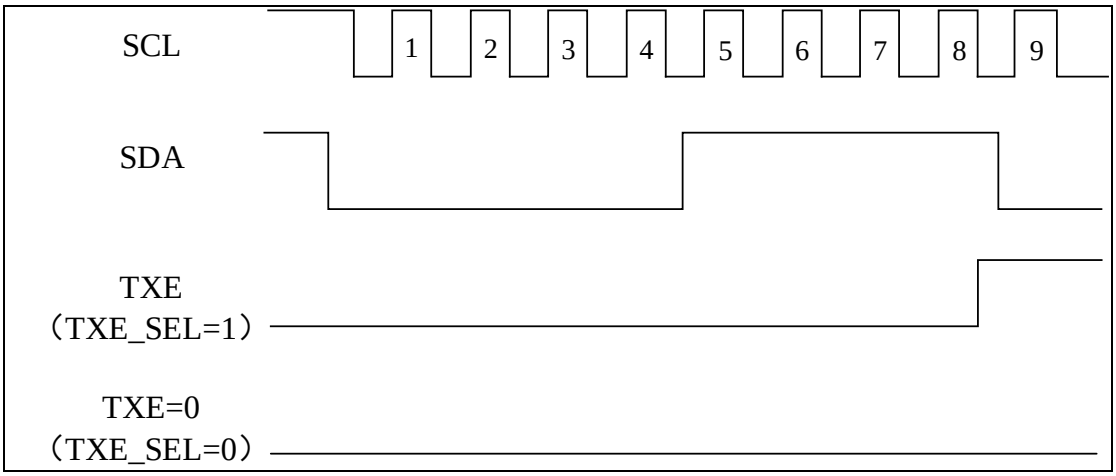
当主模式接收从模式发送时：

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。

图 24-20 TXE 状态变化



24.4.14. 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。

图 24-21 SMBus 超时规范

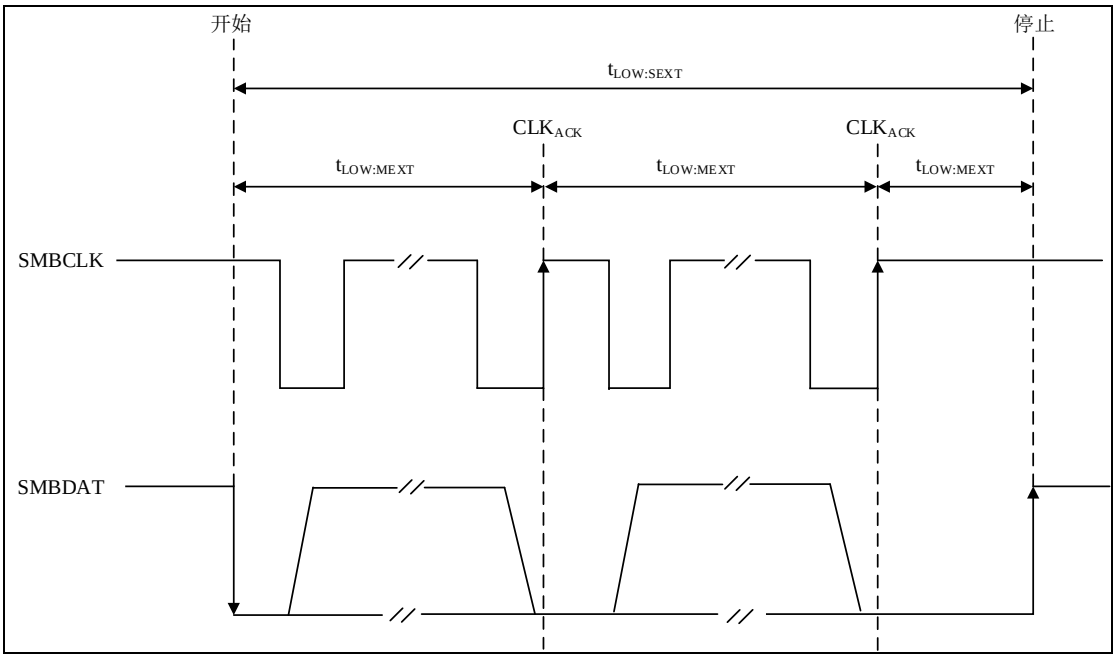
符号	参数	限值		单位
		最大值	最小值	
tTIMEOUT	检测时钟低电平超时	25	35	ms
tLOW:SEXT	累积时钟低电平延长时间（从器件）	-	25	ms
tLOW:MEXT	累积时钟低电平延长时间（主器件）	-	10	ms

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。

图 24-22 时钟延长时间



24.4.15. 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。

24.4.16. SMBus I2C_TIMEOUT 寄存器配置示例

将 tTIMEOUT 的最大持续时间配置为 25ms:

表 24-1 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 24-2 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

24.4.17. DMA 请求

DMA 请求(DMA_EN=1)仅用于数据传输。发送时数据寄存器变空或接收时数据寄存器非空，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。

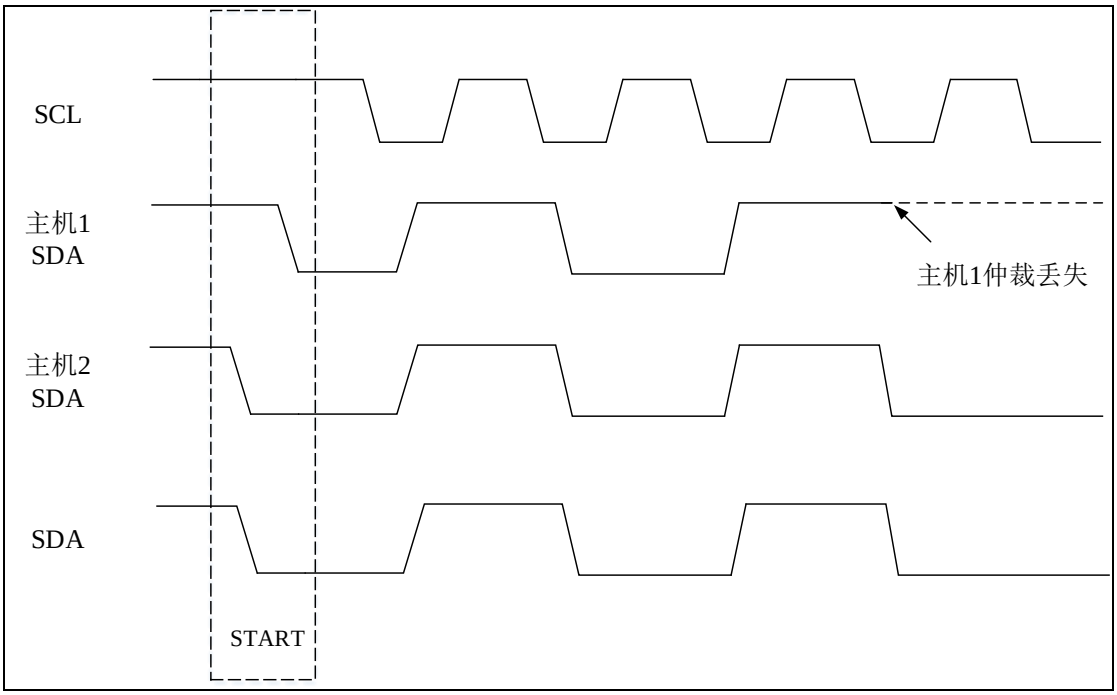
6）通过设置 DMAC_Cx_CONFIG 寄存器 EN 位激活通道。

24.4.18. 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 24-23 I2C 仲裁时序



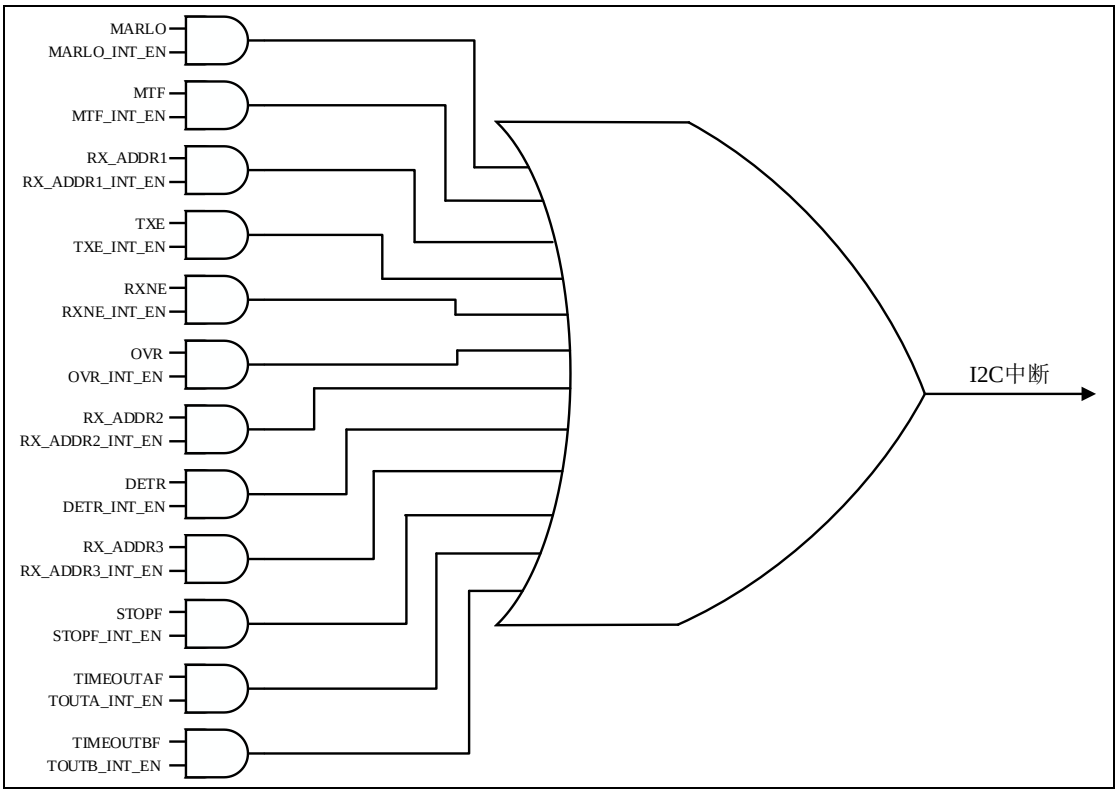
24.5. 中断及中断标志

下表列出了 I2C 中断请求列表。

表 24-3 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

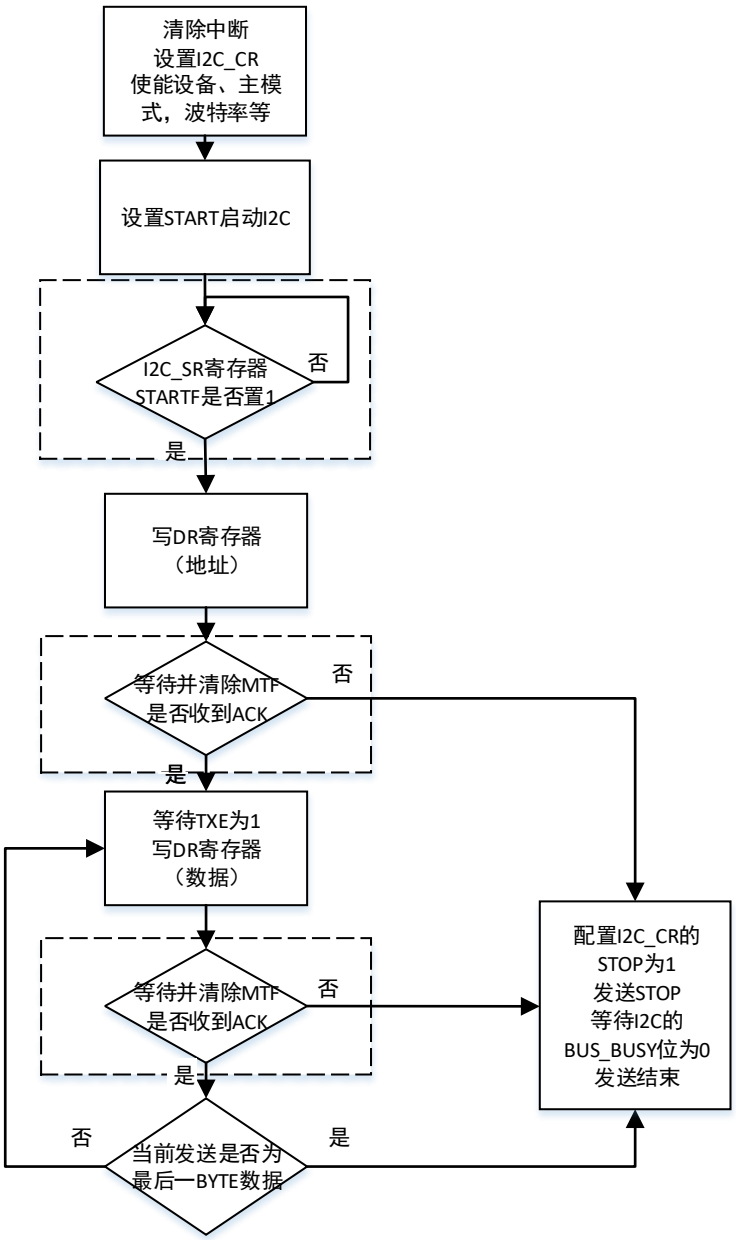
24-24 I2C 中断映射图图



24.6. 配置流程

24.6.1. 主发送器

图 24-25 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)

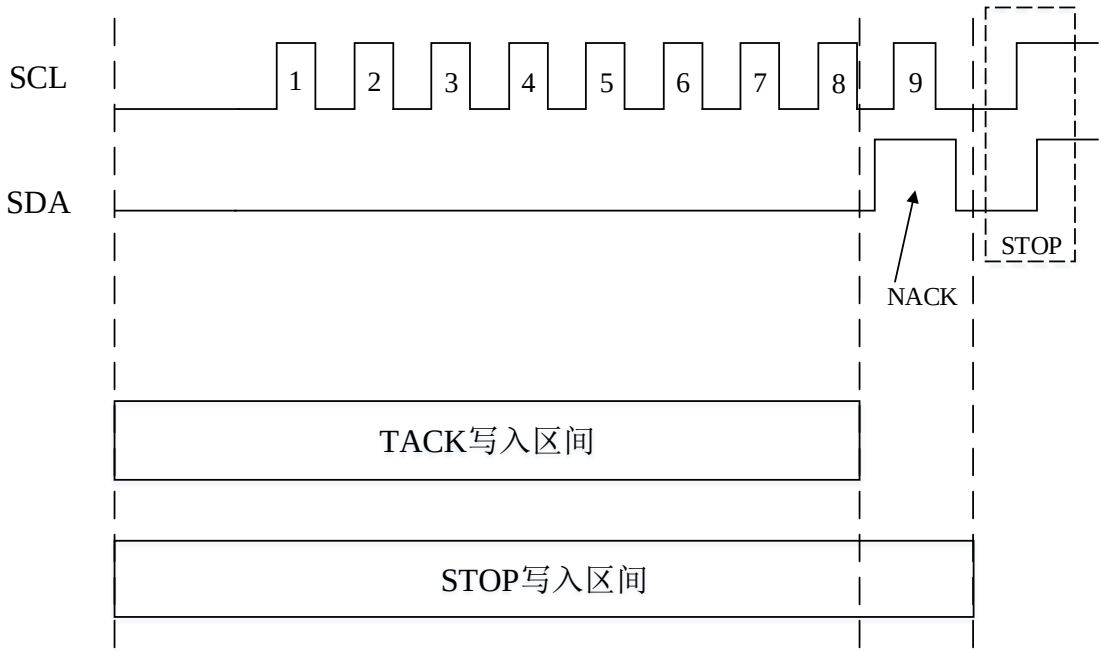
6）等待 I2C_SR.MTF 标志并清除 MTF，判断是否收到 ACK 后，表示从机正确。如果收到 NACK，软件写 I2C_CR.STOP 位为 1，结束发送并释放总线，软件等待 I2C_SR.BUS_BUSY 为 0 后退出

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 6 - 7 操作。在 7 操作中如确定回 ACK，可跳过

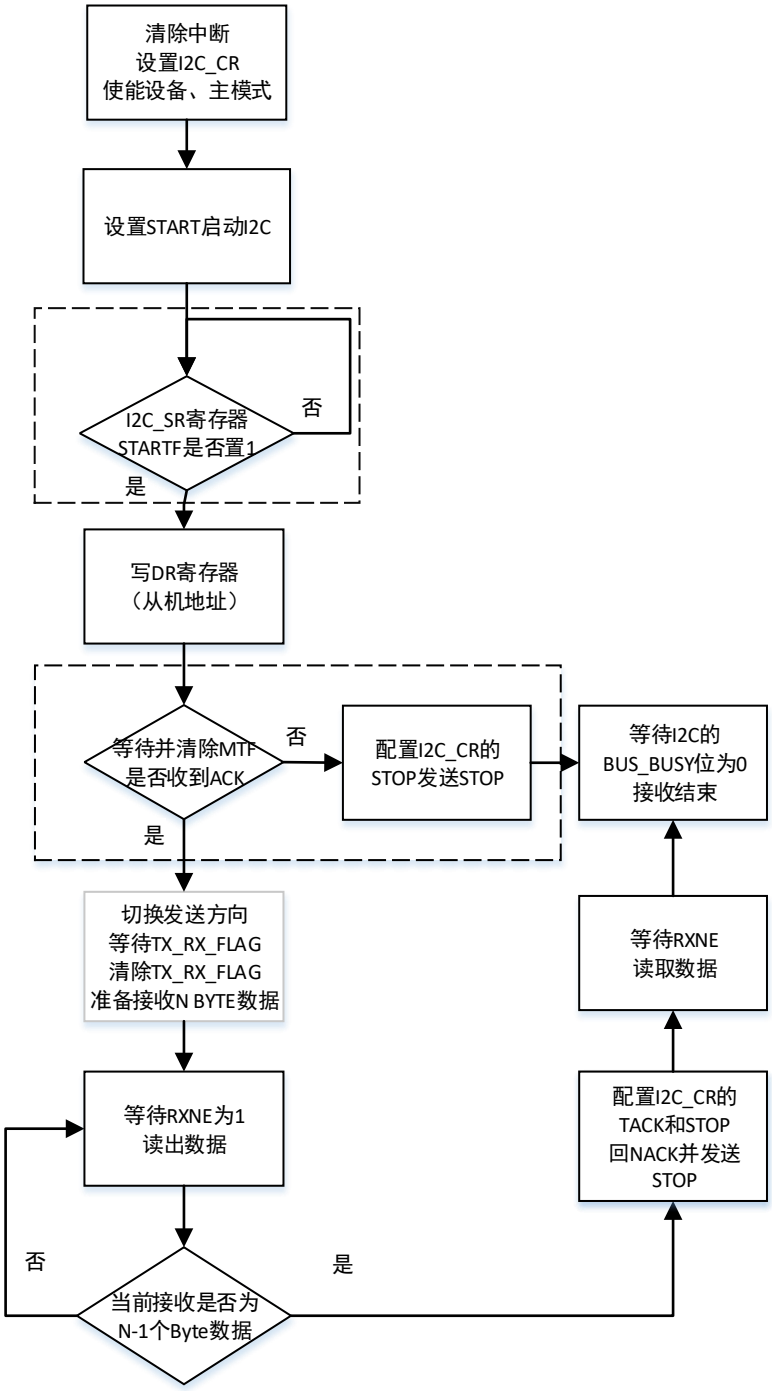
10）等到倒数第二个字节发送完成（MTF=1），向 I2C_DR 写完最后一个字节后。可以写 I2C_CR.STOP 为 1 结束发送。STOP 也可以在最后一个字节（MTF=1）发送完成后写入



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。

24.6.2. 主接收器

图 24-26 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 后，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。 (写 STOP 位后，需要重新写 DR 寄存器)

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。

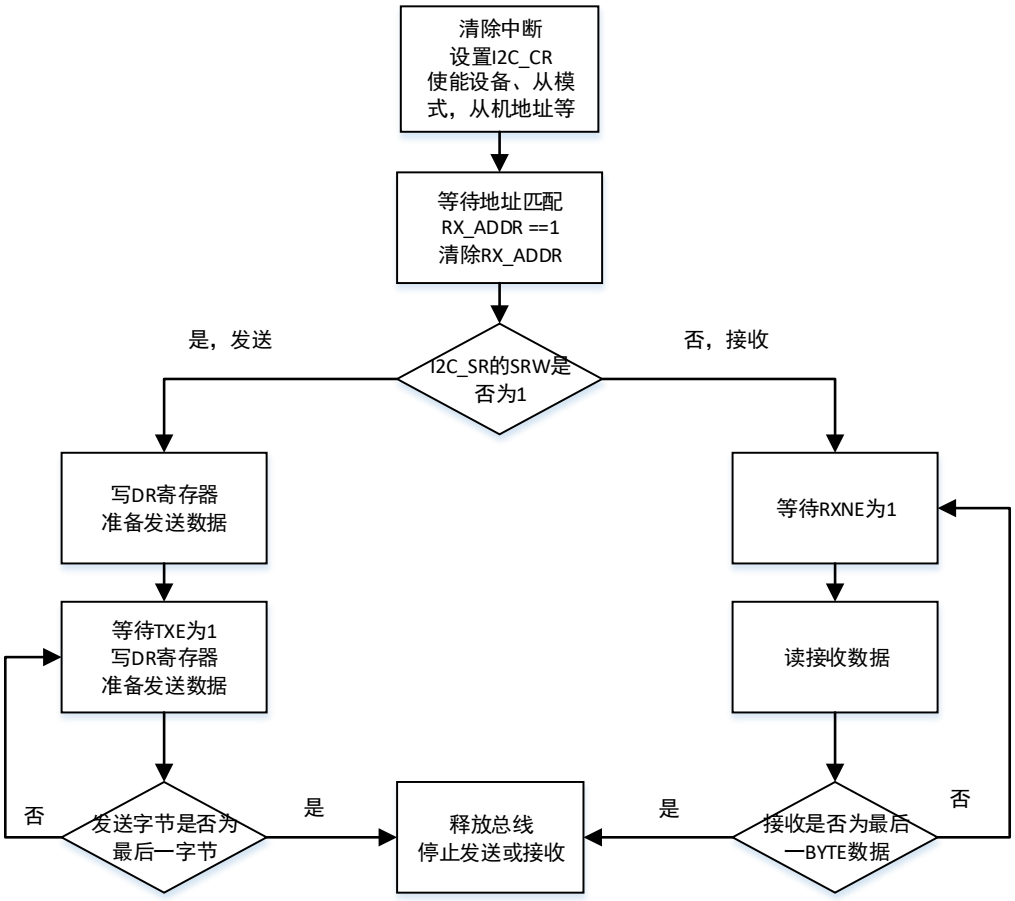
8）重复 7 操作

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

24.6.3. 从发送器

图 24-27 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

24.6.4. 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

24.7. 寄存器描述

24.7.1. 寄存器列表

I2C1 寄存器基地址：0x40005400

I2C2 寄存器基地址：0x40005800

偏移	名称	描述
0x00	I2C_SLAVE_ADDR1	设备地址寄存器 1
0x04	I2C_CLK_DIV	时钟分频寄存器
0x08	I2C_CR	控制寄存器
0x0C	I2C_SR	状态寄存器
0x10	I2C_DR	数据寄存器
0x14	I2C_SLAVE_ADDR2_3	设备地址寄存器 2_3
0x18	I2C_DET	SDA 跳变阈值寄存器
0x1C	I2C_FILTER	滤波寄存器
0x24	I2C_TIMEOUT	超时配置寄存器

24.7.2. 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[7:1]	RW	0x0	地址的 7~1 位
0	RSV	-	-	保留

24.7.3. 时钟分频寄存器(I2C_CLK_DIV: 04h)

位域	名称	属性	复位值	描述
31:12	RSV	-	-	保留
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

24.7.4. 控制寄存器(I2C_CR: 08h)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留
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。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 0：不使能自动切换功能 1：使能自动切换功能
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。

24.7.5. 状态寄存器(I2C_SR: 0Ch)

位域	名称	属性	复位值	描述
31:17	RSV	-	-	保留位
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	应答接收状态位 0：最近的发送应答周期接收到 ACK 1：最近的发送应答周期接收到 NACK 只有 START 条件将清除 RACK 位 注：主机发送模式下收到 NACK 停止发送，等待软件处理事件

24.7.6. 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

24.7.7. 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[7:1]	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[7:1]	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

24.7.8. 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

24.7.9. 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

24.7.10. 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

25. 集成电路内置音频接口（I2S）

25.1. 概述

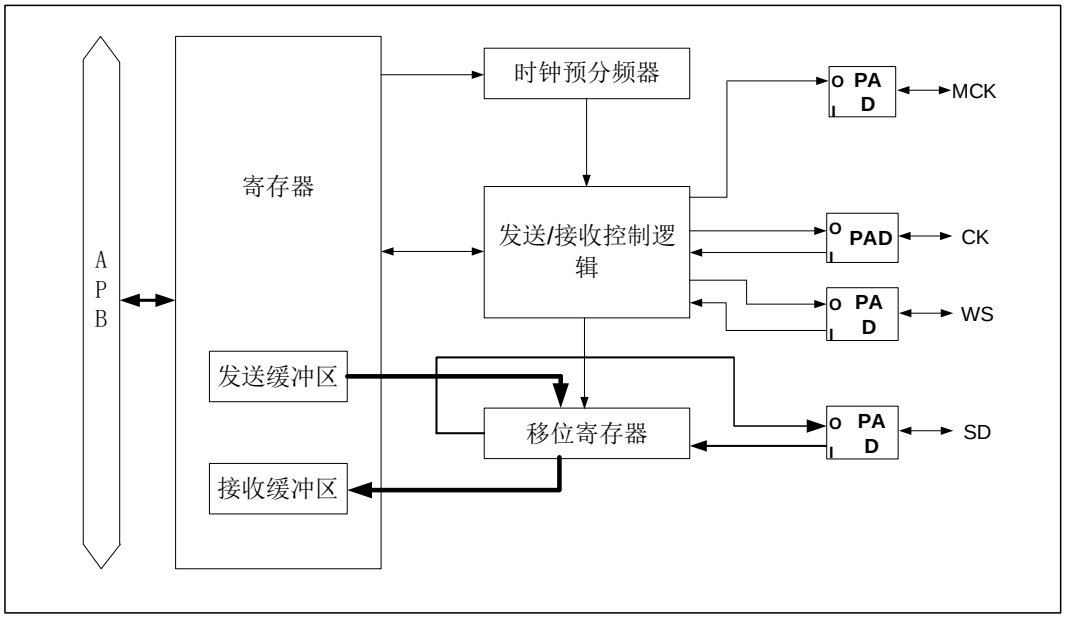
集成电路内置音频接口（Inter-IC Sound，缩写为 I2S）模块可以通过 I2S 音频协议与外部设备进行通信。I2S 接口支持四种音频标准，分别是 I2S 飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准。它可以在四种模式下运行，包括主机发送模式，主机接收模式，从机发送模式和从机接收模式。

25.2. 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；

25.3. 结构框图

图 25-1 I2S 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。

25.4. 功能描述

25.4.1. 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 25-1 I2S 比特率计算公式

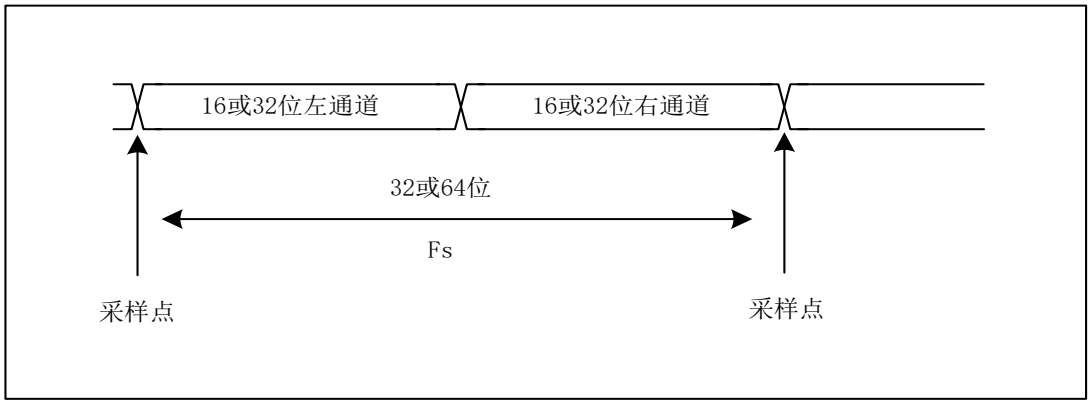
MCKOE	CHLEN	公式
0	0	PCLK / (DIV * 2 + OF)
0	1	PCLK / (DIV * 2 + OF)
1	0	PCLK / (8 * (DIV * 2 + OF))
1	1	PCLK / (4 * (DIV * 2 + OF))

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：

图 25-2 音频采样频率



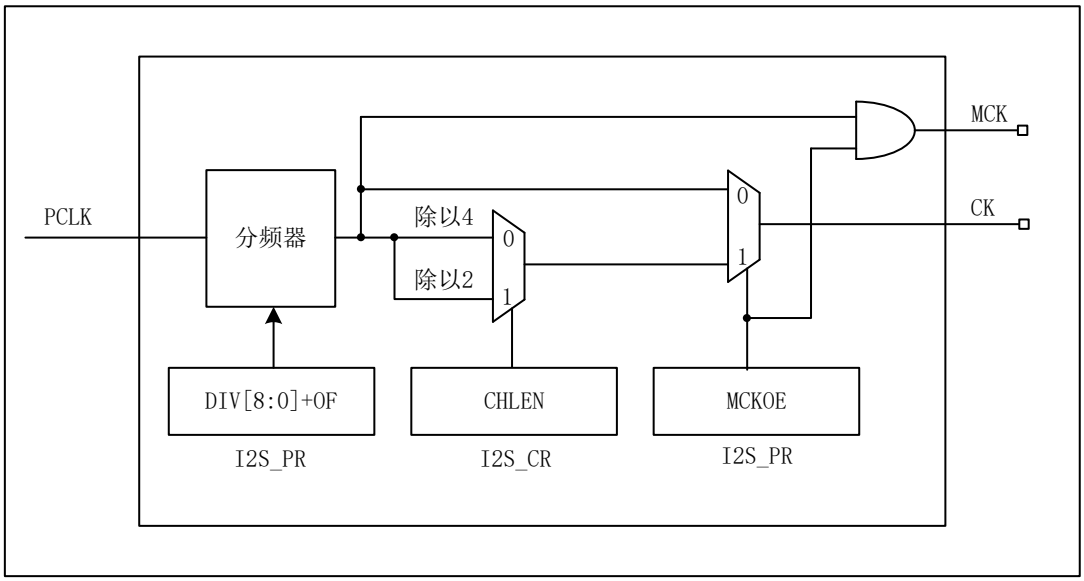
为了得到期望的音频采样率，时钟生成器需要按下表音频采样频率计算公式所列的公式进行配置。

表 25-2 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。

图 25-3 I2S 时钟发生器框图



25.4.2. 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。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。

25.4.3. 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 25-4 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=0)

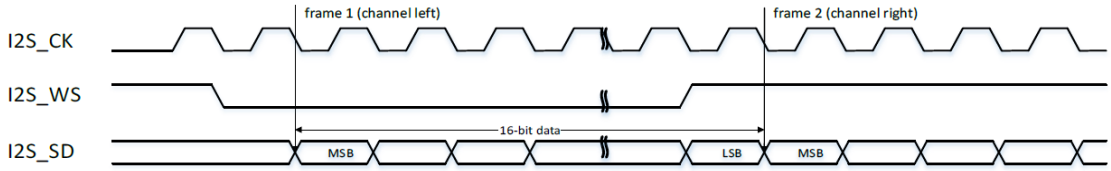


图 25-5 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=1)

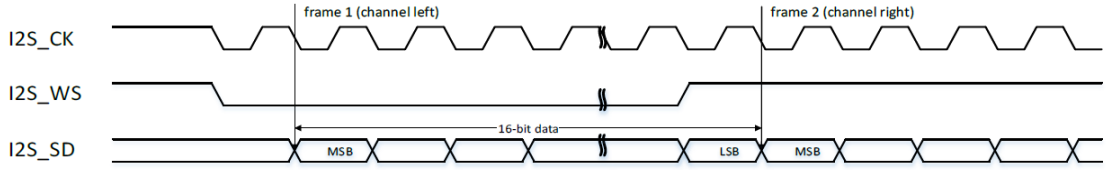


图 25-6 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=0)

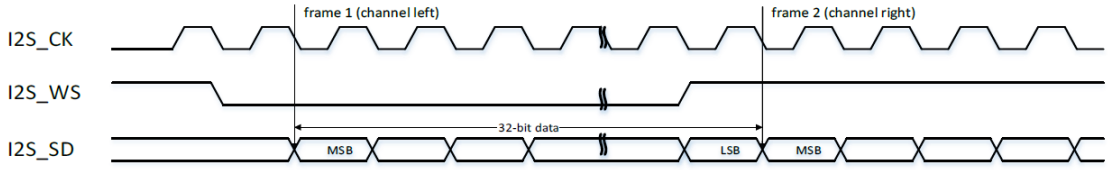
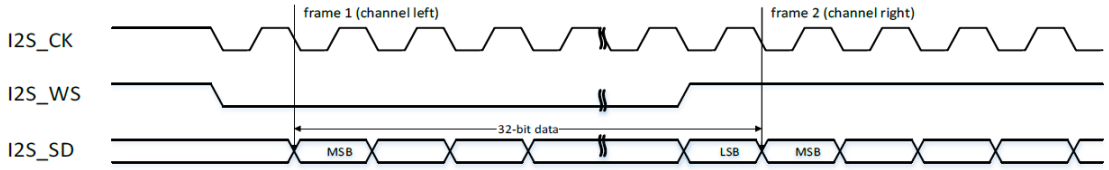


图 25-7 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=1)



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 25-8 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

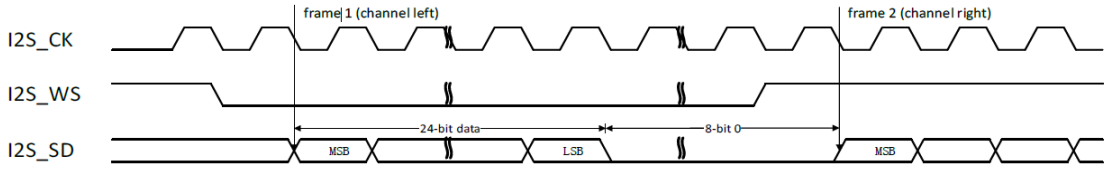
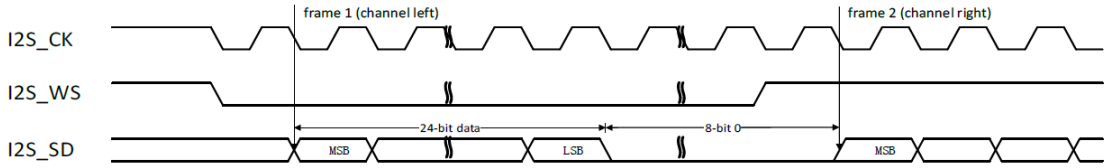


图 25-9 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=1)



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 25-10 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

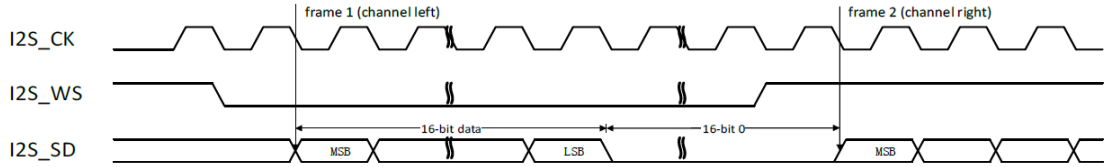
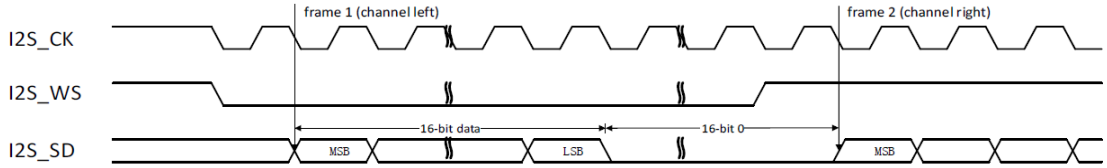


图 25-11 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



25.4.4. 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，有效数据从第一个时钟下降沿开始。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 25-12 MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=0)

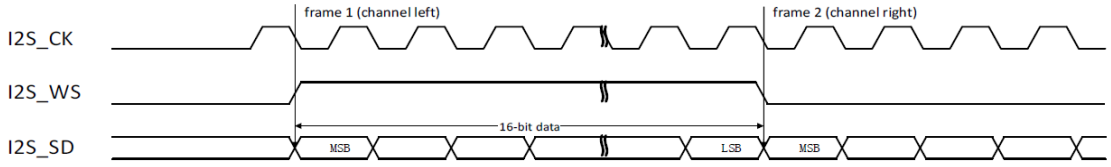


图 25-13 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=1)

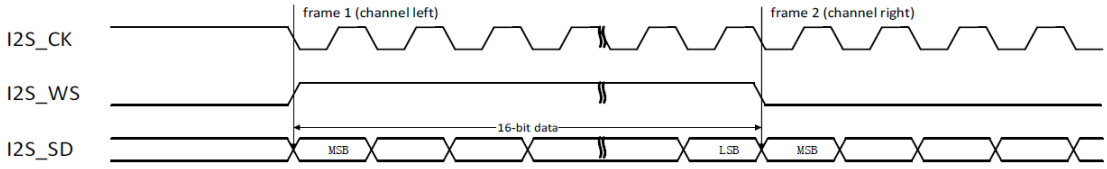


图 25-14 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=0)

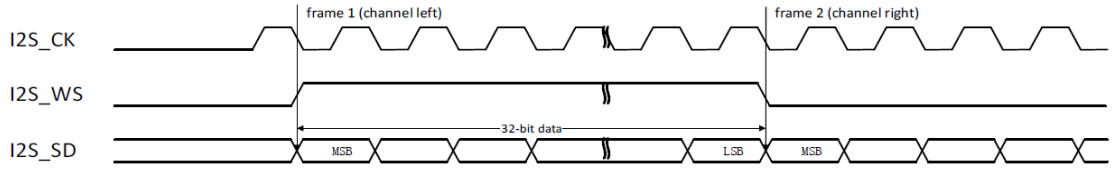


图 25-15 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=1)

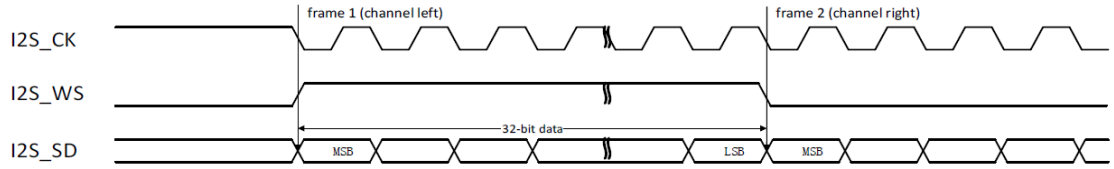


图 25-16 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

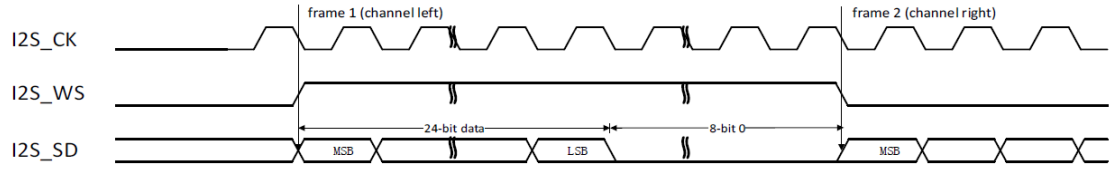


图 25-17 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)

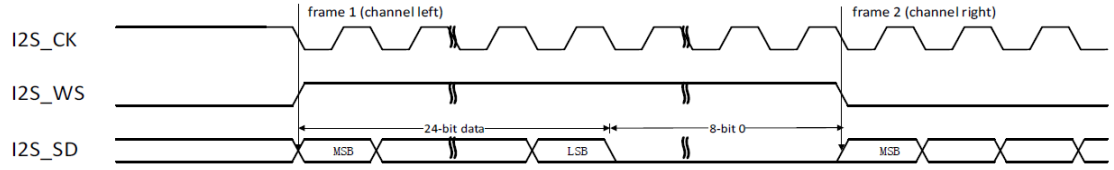


图 25-18 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

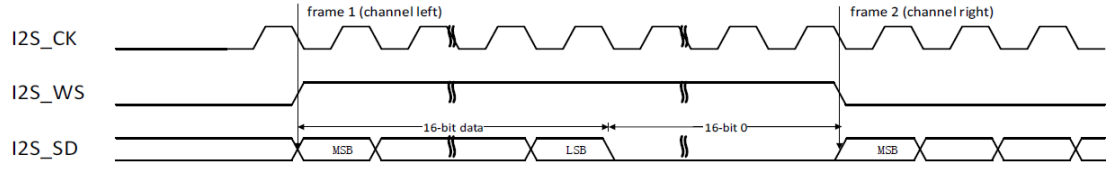
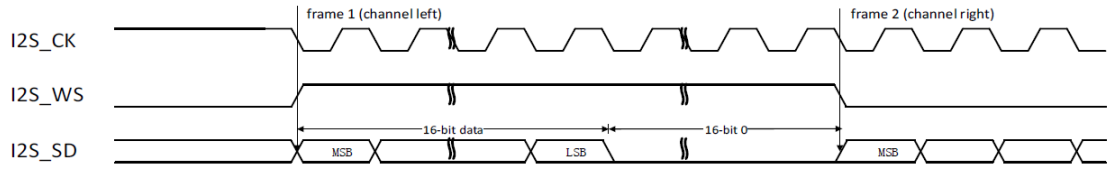


图 25-19 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



25.4.5. 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，有效数据从第一个时钟下降沿开始。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 25-20 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

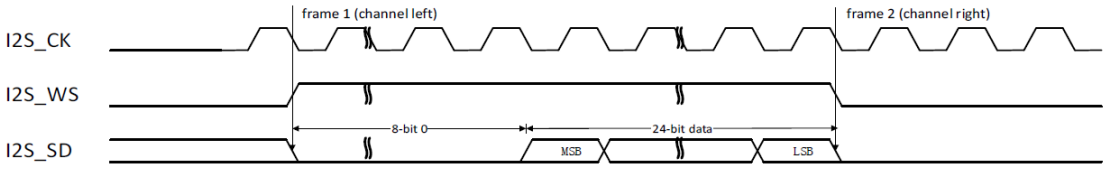
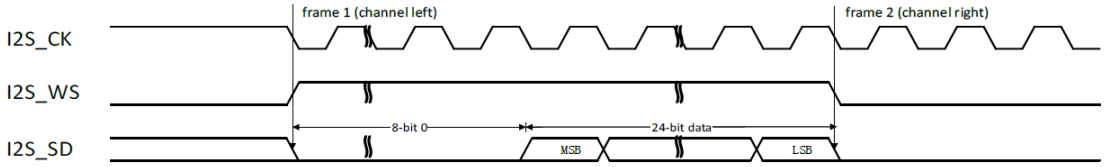


图 25-21 LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)



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 25-22 LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

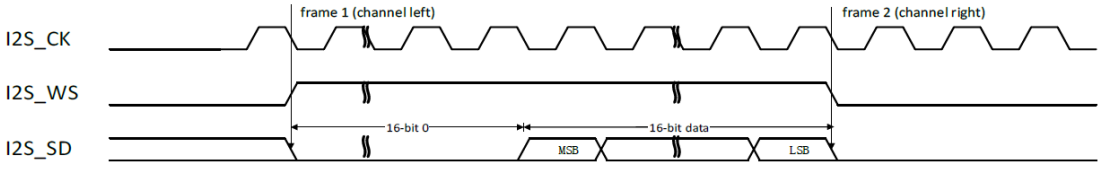
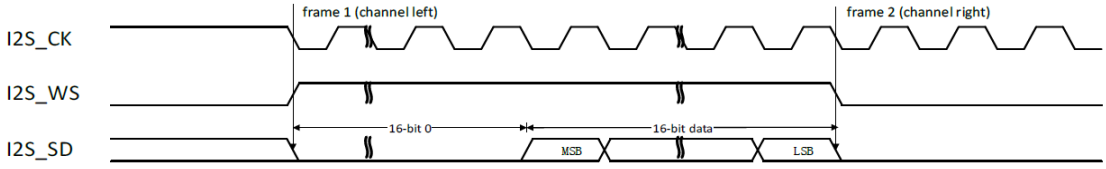


图 25-23 LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)



25.4.6. 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 25-24 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)

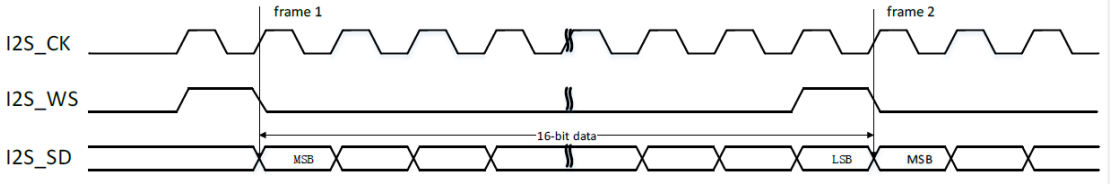


图 25-25 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)

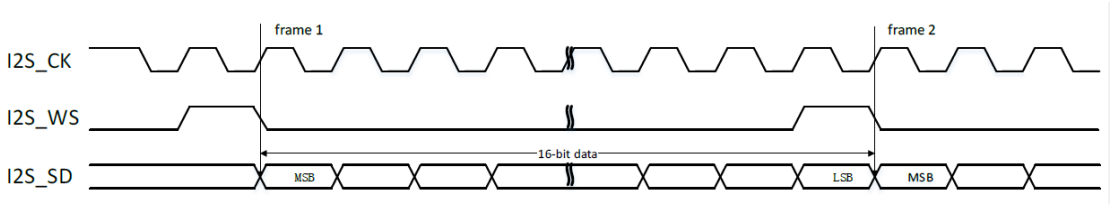


图 25-26 PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)

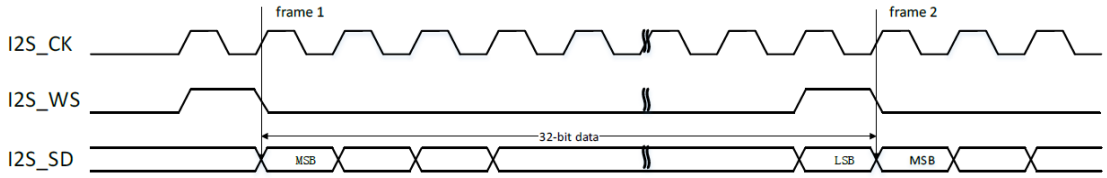


图 25-27 PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)

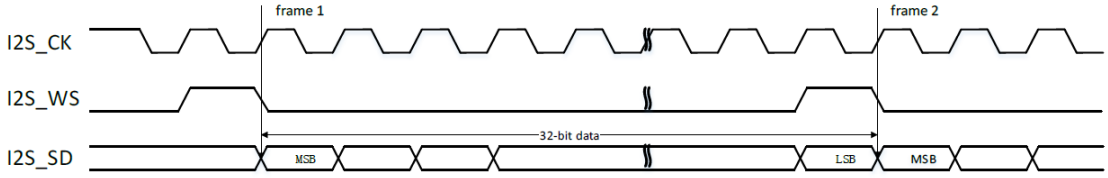


图 25-28 PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

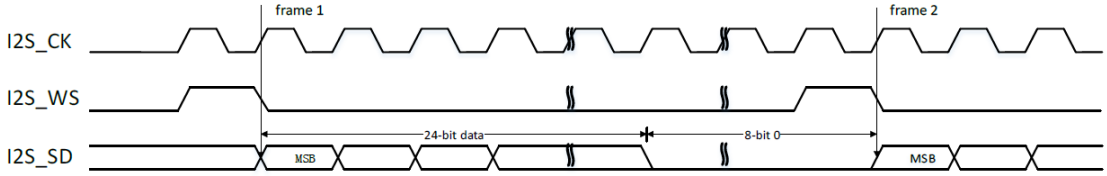


图 25-29 PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

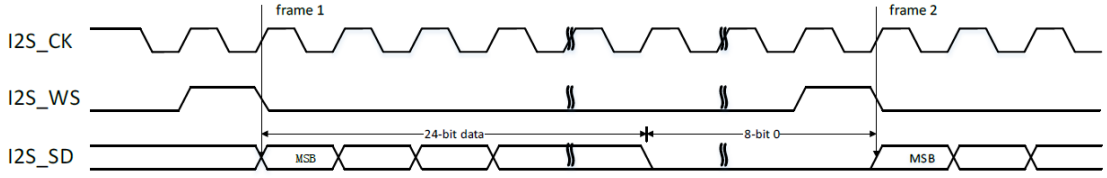


图 25-30 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)

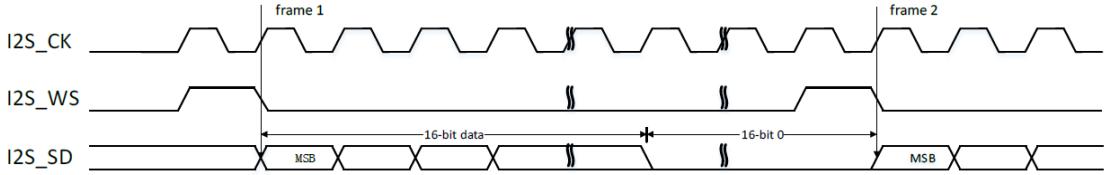
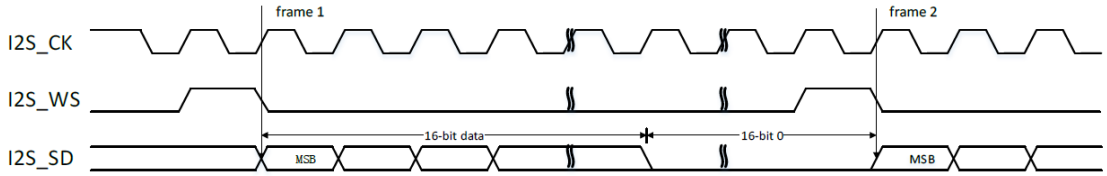


图 25-31 PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)



长帧同步模式的各种配置情况时序图如下所示。

图 25-32 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)

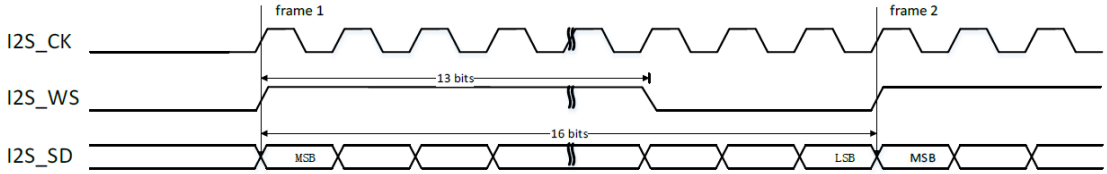


图 25-33 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)

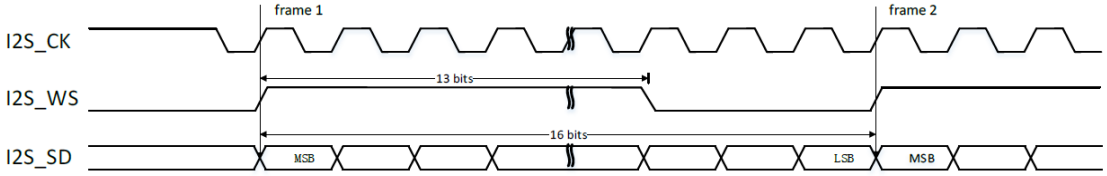


图 25-34 PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)

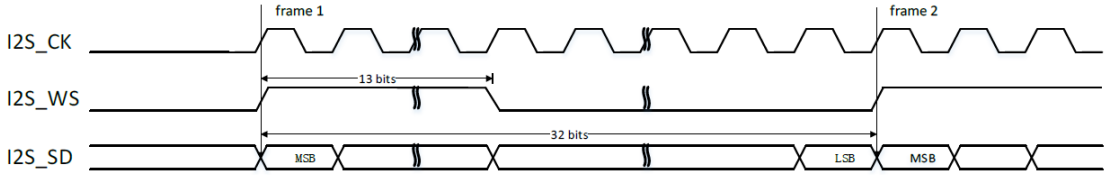


图 25-35 PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)

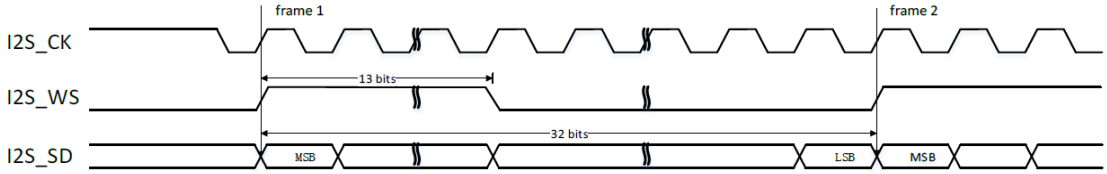


图 25-36 PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

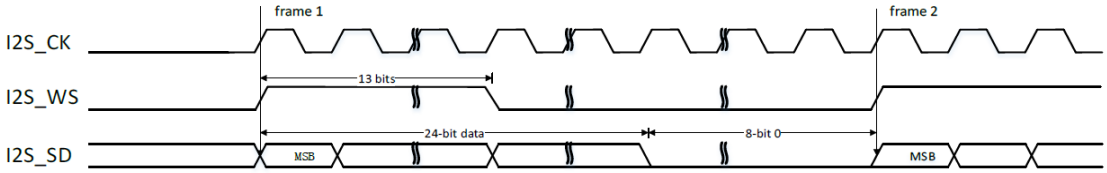


图 25-37 PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

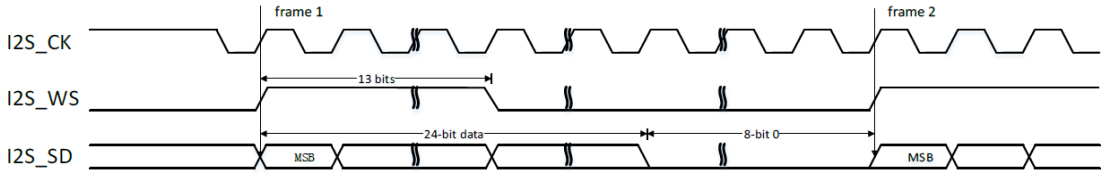


图 25-38 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)

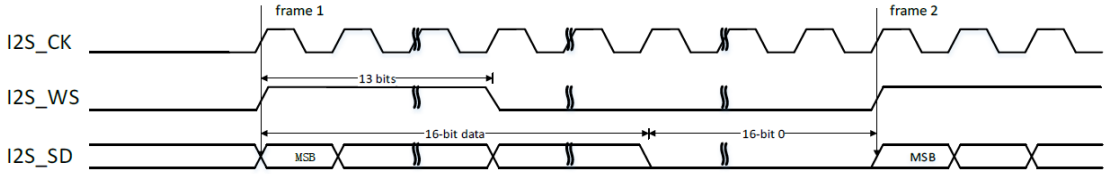
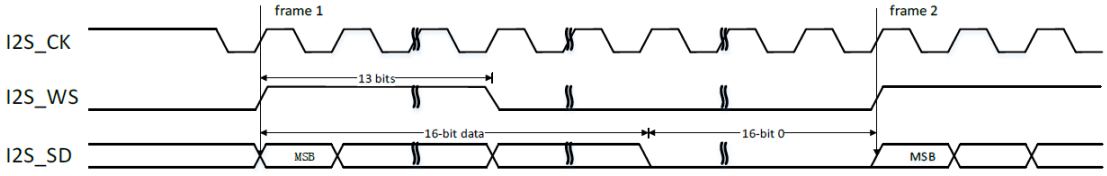


图 25-39 PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)



25.4.7. 运行模式

I2S 运行方式通过 I2S_CR.I2SOPMOD 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、主机接收模式四种运行模式，四种模式均为单工通信。

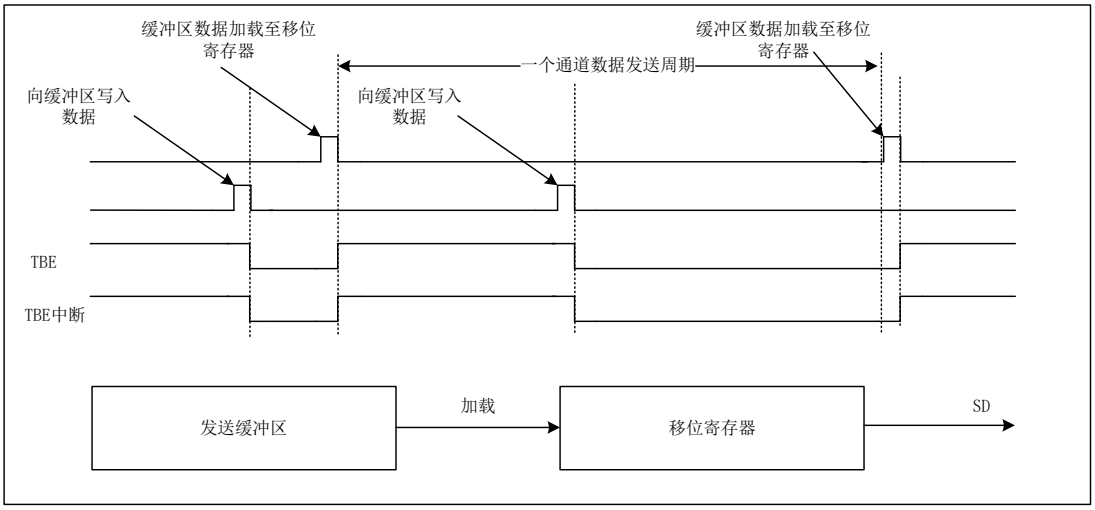
当配置为主机发送模式、主机接收模式时，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 I2S_PR.MCKOE 位来选择输出或者不输出主时钟（MCK）。

25.4.8. 主发模式

TBE 标志位被用来控制发送流程。如前文所述，TBE 标志位表示发送缓冲区空，此时，如果 I2S_IE 寄存器的 TBEIE 位为 1，将产生中断。首先，发送缓冲区为空(TBE 为 1)，且移位寄存器中没有发送序列。当 32 位数据被写入 I2S_DR 寄存器时(TBE 变为 0)，数据立即从发送缓冲区装载到移位寄存器中(TBE 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 TBE 为 1 时写入 I2S_DR 寄存器。数据写入 I2S_DR 寄存器之后，TBE 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 TBE 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。

图 25-40 主发模式



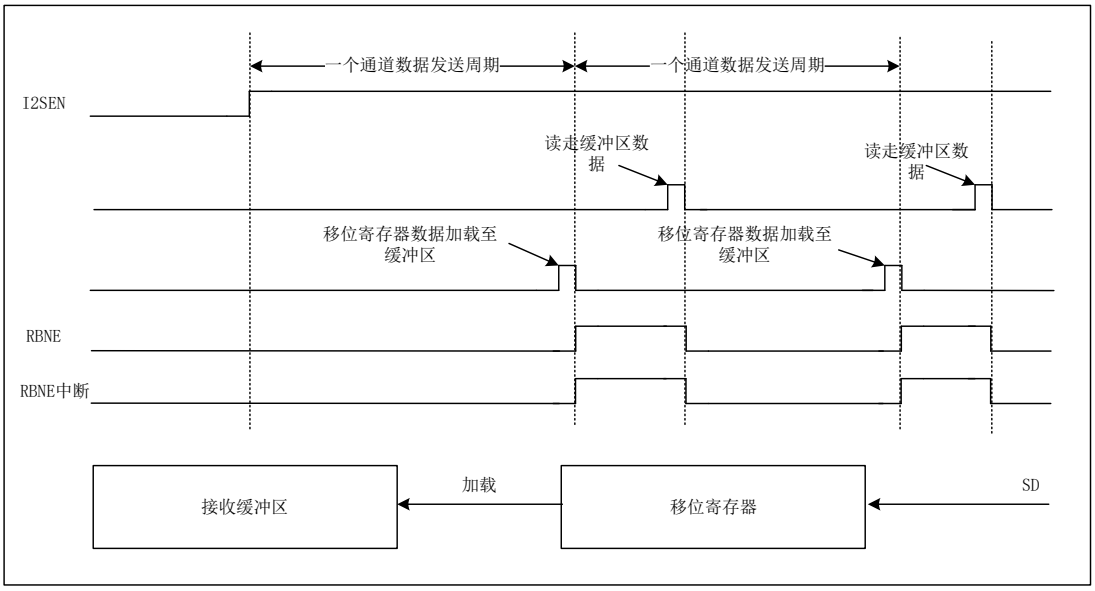
对于除 PCM 标准外的所有标准，I2SCH 标志用来区别当前传输数据所属的通道。I2SCH 标志在每次 TBE 标志由 0 变 1 的时候更新。刚开始 I2SCH 标志为 0，表示左通道的数据应该被写入 I2S_DR 寄存器。

为了关闭 I2S，I2SE 位必须在 TBE 标志为 1 且 TRANS 标志为 0 之后清零。

25.4.9. 主收模式

RBNE 标志被用来控制接收序列。如前文所述，RBNE 标志表示接收缓冲区非空，如果 I2S_IE 寄存器的 RBNEIE 位为 1，将产生中断。当 I2S_CR 寄存器的 I2SE 位被置 1 时，接收流程立即开始。首先，接收缓冲区为空(RBNE 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(RBNE 变为 1)。当 RBNE 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，RBNE 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。

图 25-41 主收模式



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 RBNE 标志由 0 变 1 时更新。

25.4.10. 从发模式

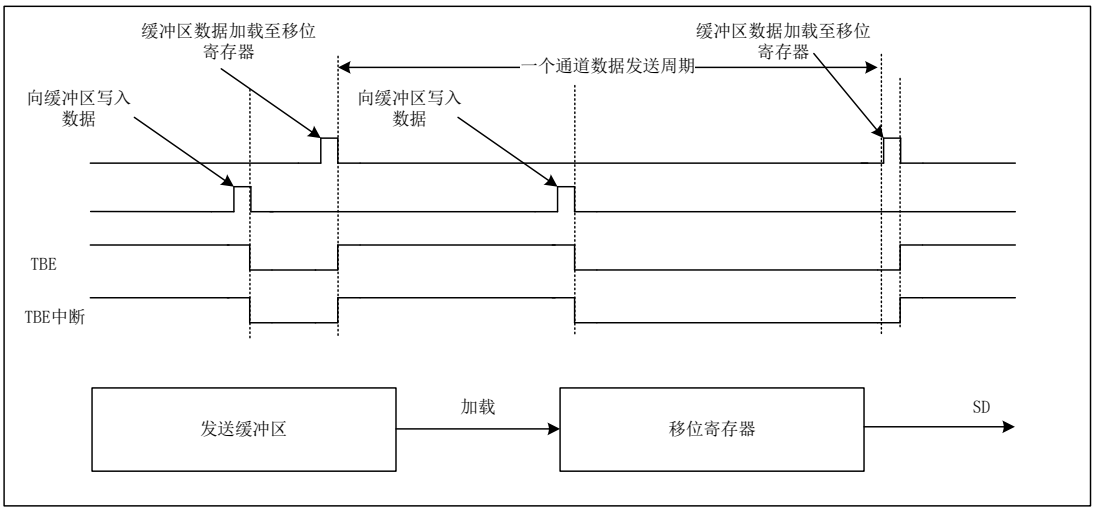
从机发送流程和主机发送流程相似，不同之处如下：

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_DR 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_DR 寄存器，否则会产生发送欠载错误。此时 TXURERR 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 TBE 标志变为 1 且 TRANS 标志变为 0 之后，才能清除 I2SE 位。

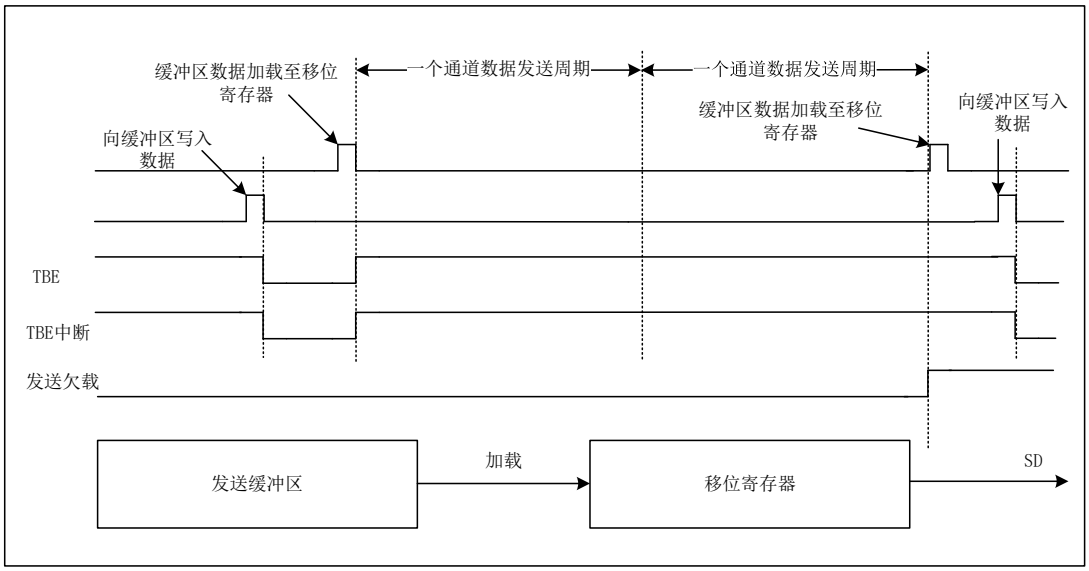
下图为在当前发送序列结束之前将下一个待发送的数据写入 I2S_DR 寄存器，未发生发送欠载错误的示意图，与主发流程相同。

图 25-42 从发模式（无发送欠载错误）



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。

图 25-43 从发模式（发送欠载错误）



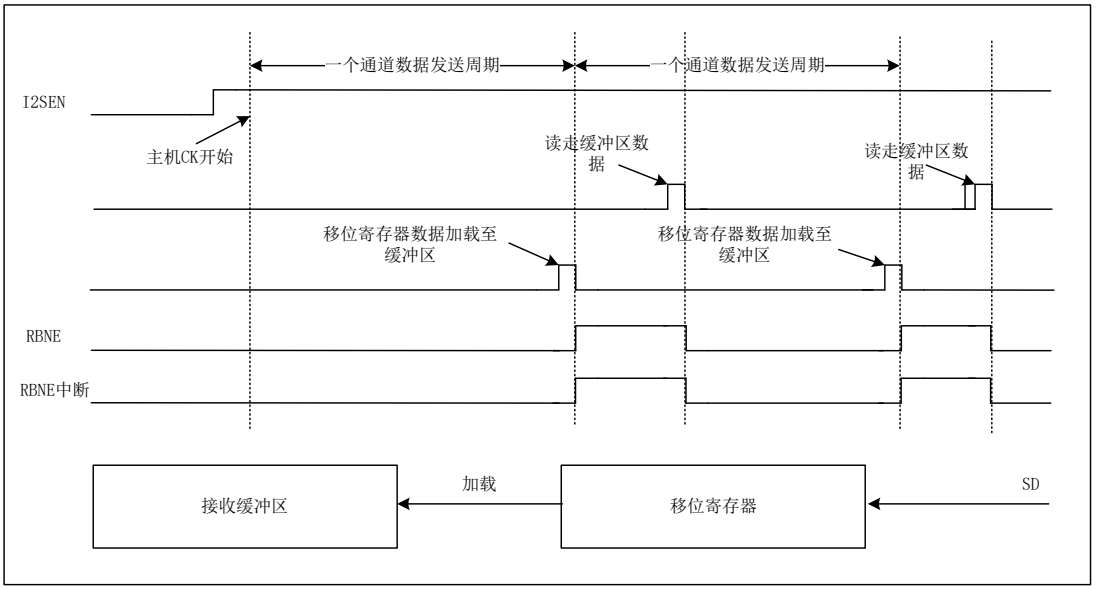
25.4.11. 从收模式

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 RBNE 之后立即清除 I2SE 位。

图 25-44 从收模式



25.4.12. 状态标志

- 接收缓冲区非空（RXBNE）

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（TXBE）

主或从模式的发送缓冲区写入数据后置一，数据载入移位寄存器将复位该标志位。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。

● 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（TXURERR）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 RXORERR），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（TXURERR）

仅在从发送模式下，如果在软件尚未将任何值加载到 I2S_DR 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（RXORERR）

主或从机尚未从 I2S_DR 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_DR 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_DR 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 RXORERR 位清零，应首先对 I2S_DR 寄存器执行读操作，然后再读 SPI_SR 寄存器即可清除该标志。

● 通信进行中标志(TRANS)

该标志表示 I2S 通信的状态，由硬件置 1 和清零（写入此标志没有任何作用）。

硬件将 TRANS 置 1 时，指示 I2S 正在忙于通信，如果软件需要关闭 I2S，可使用 TRANS 标志检测传输是否结束。以避免破坏最后一个数据的传输。为此，必须严格遵循下述步骤。

在传输开始时硬件将 TRANS 标志置 1。 出现以下情况时，TRANS 标志被硬件清零。

- 主发模式，缓冲区和移位寄存器中均没有数据时
- 主收模式，缓冲区数据未读走且在发生 RXORERR 后
- 作为从机，主机传输完成，不再发送 CK 后
- 关闭 I2S 时
- 当通信连续时：
- 在主发送模式下，BSY 标志在所有传输期间均保持高电平
- 在从模式下，BSY 标志在每次传输之间变为低电平并持续一个 I2S 时钟周期

● 帧错误(FERR)

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，只能通过模块复位的方式来修正传输。

25.4.13. 中断

表 25-3 中断

中断事件	中断标志	使能位
接收缓冲区非空中断	RXBNE	RBNEIE
发送缓冲区空中断	TXBE	TBEIE

错误中断	RXORERR	ERRIE
	TXURERR	

25.4.14. 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_DIER 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 TXBE、RXBNE 中断。

25.5. 配置流程

25.5.1. 主发模式

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.I2SSTD 位、PCMSMOD 位、I2SOPMO 位、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

4）配置 I2S_DIER.TBEIE 位、ERRIE 位和 TXDMATEN 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）等待 I2S_SR. TXBE 置位，向 I2S_DR 地址写入音频数据，I2S 立即开始传输。

7）通过 TRANS 或其他中断方式等待传输完成，当所有数据发送完成，等待 I2S_SR.TRANS 复位，I2S_CR.I2SE 位复位，关闭 I2S。

25.5.2. 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.I2SSTD 位、PCMSMOD 位、I2SOPMO 位、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

4）配置 I2S_DIER.RBNEIE 位、ERRIE 位和 RXRXDMAREN 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，传输立即开始。

6）根据 RXBNE 标志位进行数据读取。

7）通过 TRANS 或其他中断方式等待传输完成，I2S_CR.I2SE 位置 0，关闭 I2S。

25.5.3. 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.I2SSTD 位、PCMSMOD 位、I2SOPMO 位、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

3）配置 I2S_DIER.TBEIE 位、ERRIE 位和 TXDMATEN 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. TXBE 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 TRANS 或其他中断方式等待传输完成，当所有数据发送完成，等待 I2S_SR.TRANS 复位，I2S_CR.I2SE 位置 0，关闭 I2S。

25.5.4. 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.I2SSTD 位、PCMSMOD 位、I2SOPMO 位、DTLEN 位和 CHLEN 位，定义 I2S 的特性。

3）配置 I2S_DIER.RBNEIE 位、ERRIE 位和 RXRXDMAREN 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 RXBNE 标志位进行数据读取。

6）通过 TRANS 或其他中断方式等待传输完成，I2S_CR.I2SE 位置 0，关闭 I2S。

25.6. I2S 寄存器描述

25.6.1. 寄存器列表

I2S 寄存器基地址：0x40013000

偏移	名称	描述
0x00	I2S_DR	数据寄存器
0x04	I2S_CR	控制寄存器
0x08	I2S_PR	时钟预分频寄存器
0x0C	I2S_DIER	DMA/中断使能寄存器
0x10	I2S_SR	状态寄存器

25.6.2. 数据寄存器(I2S_DR: 00h)

位域	名称	属性	复位值	描述
31:0	I2S_DR[31:0]	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。向 I2S_DR 写数据将会把数据存入发送缓冲区，从 I2S_DR 读数据，将从接收缓冲区获得数据。

25.6.3. 控制寄存器(I2S_CR: 04h)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留

10	I2SE	RW	0x0	I2S 使能 0：I2S 禁止； 1：I2S 使能；
9:8	I2SOPMOD	RW	0x0	I2S 运行模式 00：从机发送模式 01：从机接收模式 10：主机发送模式 11：主机接收模式 当 I2S 模式关闭时配置该位。
7	PCMSMOD	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	I2SSTD[1:0]	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[1:0]	RW	0x0	数据长度（原始数据位数） 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（通道实际发送的数据位数） 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。

25.6.4. 时钟预分频寄存器(I2S_PR: 08h)

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	0x	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[8:0]	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。

25.6.5. I2S DMA/中断使能寄存器(I2S_DIER: 0Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TBEIE	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。
6	RBNEIE	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。
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 CONFERR 位，RXORERR 位或者 TXURERR 位置 1 时，产生中断。
4:2	RSV	-	-	保留
1	TXDMATEN	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。
0	RXDMAREN	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。

25.6.6. 状态寄存器(I2S_SR: 10h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留

8	FERR	RC W0	0x0	帧错误 I2S 模式： 0：没有 I2S 帧错误发生 1：I2S 帧错误发生 WS 的帧头与对应的 bit 位发生错位时，判定为帧错误，该位由硬件置位，可以通过写 0 清除或读状态寄存器清除。
7	TRANS	RO	0x0	通信进行中标志 0：I2S 空闲 1：I2S 当前正在发送或接收数据 该位由硬件置位和清除。
6	RXORERR	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 从机：接受数据时，接受缓冲区已满，主机仍提供时钟时，则发生接受过载。 主机：接受数据时，接受缓冲区已满，但从机还在发送数据，则发生接受过载。 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，然后读 I2S_SR 寄存器。
5:4	RSV	-	-	保留
3	TXURERR	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 从机：发送数据时，发生缓冲区已经空，但主机仍提供时钟，则发生发送欠载。 主机：无 该位由硬件置位，通过读 I2S_SR 寄存器清除。
2	I2SCH	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	TXBE	RO	0x1	发送缓冲区空 0：发送缓冲区非空 1：发送缓冲区空
0	RXBNE	RO	0x0	接收缓冲区非空 0：接收缓冲区空 1：接收缓冲区非空

26. 控制器区域网络（CAN）

26.1. 概述

CAN 是控制器局域网络(Controller Area Network)的简称，是一种异步的半双工通讯。ACM32G103 芯片中集成了 2 路 CAN。本芯片中的 CAN 模块已通过 BOSCH CAN2.0 测试。该外设支持 2.0A 和 B 版本的 CAN 协议。

26.2. 主要特性

- 支持 CAN2.0，包括 CAN2.0A 和 CAN2.0B
- 支持 11 比特和 29 比特的识别符
- 支持最低 125KB 波特率和 1MB 波特率
- 64 字节的接收 FIFO。可以同时容纳最多 5 个扩展帧
- 支持热拔插。
- 支持 7 组接收器滤波
- Single-Shot 传输选项
- 支持只监听模式
- 可以接收自己的信息
- 支持自测模式
- 支持 CAN 总线错误的中断
- 记录仲裁失败后的 bit 位置
- 读写错误计数器
- 可编程的错误上限警告
- 支持自动重发机制
- 支持单次发送

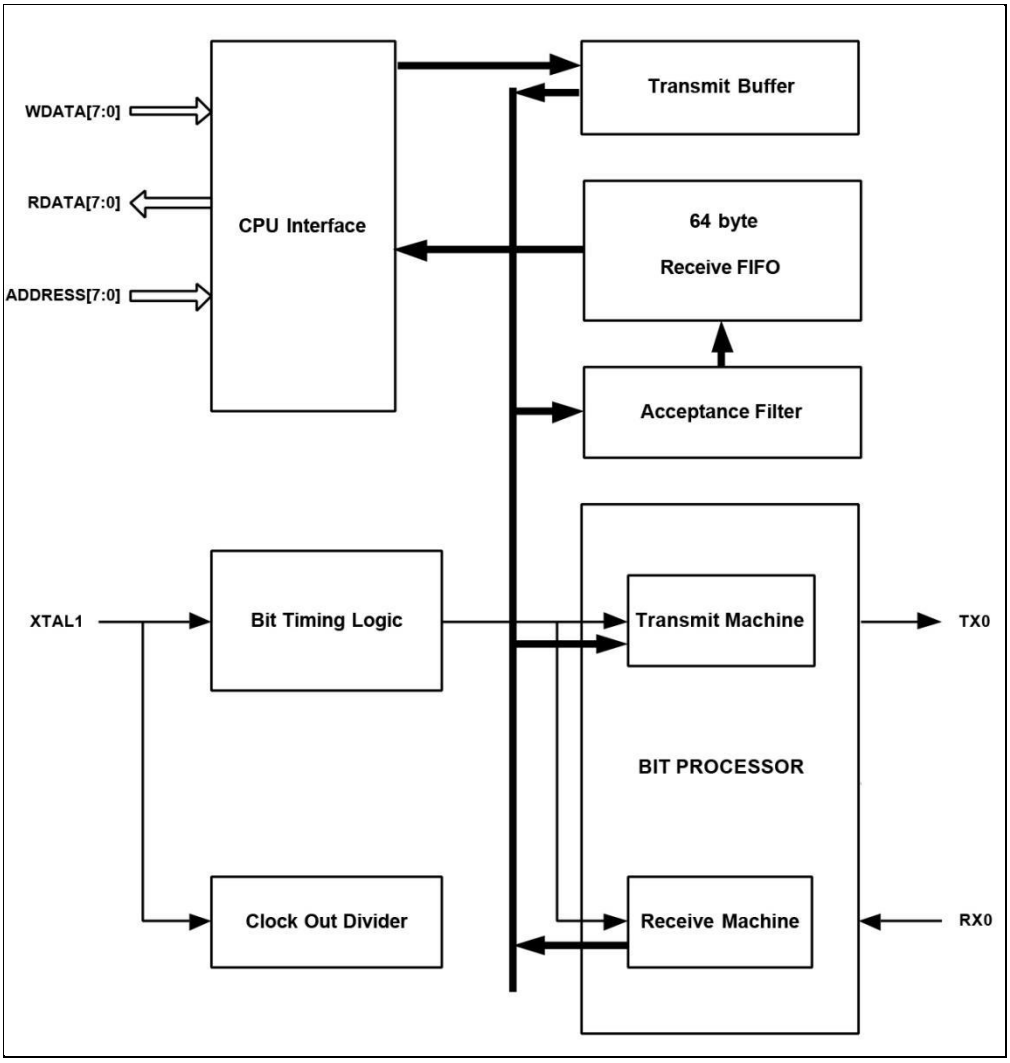
26.3. 功能描述

26.3.1. 简述

在如今的 CAN 应用中，网络节点数量日益增多，经常需要通过网关将数个网络连接在一起。系统中的消息数量（以及各个节点需要处理的消息）也有了显著增加。除应用程序消息外，还引入了网络管理和诊断消息。各种类型的消息需要一个增强的筛选机制进行处理。此外，应用程序任务需要更多的 CPU 时间，因此必须减少因消息接收而对实时处理造成的限制。接收 FIFO 方案使 CPU 能够长时间专门处理应用程序任务，而又不致丢失消息。

26.3.2. 功能框图

图 26-1 内部功能框图



注：CPU 通过独立的地址线、输入数据线、输出数据线来访问 CAN 模块。发送的数据被放到 CAN_TXBUFF 中去，并由发送器发送出去。接收的数据先通过接收滤波器过滤以后再放到 CAN_RXFIFO 中。CPU 通过一个 13bytes 宽度的窗口访问 CAN_RXFIFO。CAN_RXFIFO 总共有 64 个字节的缓存，可以一次性存储 5 组扩展帧。Bit-Timing-Logic 模块负责用来产生波特率。TX0 用于发送，RX0 用于接收。

26.3.3. 操作模式

CAN 有两种工作模式：

- 复位模式：可以修改时间参数和报文过滤参数。进入 Reset Mode 的方式有两种：第一种就是执行一次 CAN 模块复位，第二种方式是将 CAN_MOD.RM 置 1。另外，进入 bus off 时也会进入复位模式。
- 正常工作模式：可以正常发送接收 CAN 总线数据

初始阶段

1）初始化 CAN 引脚，使能 CAN 模块时钟，复位 CAN 模块

2）配置 CAN_MOD 寄存器，进入复位模式

3）配置 CAN 通讯速率，通过时序寄存器 CAN_BTR0 和 CAN_BTR1 设置

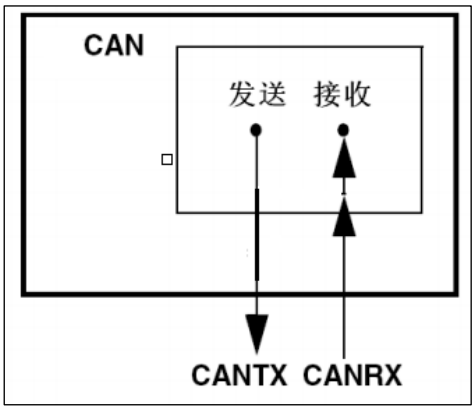
4）配置接收过滤模式和规则，通过 CAN_ACRx 和 CAN_AMRx 设置过滤规则，通过 CAN_MOD 设置过滤模式

5）配置使能/禁止相应中断

6）配置 CAN_MOD 寄存器，进入正常模式

进入正常工作的方式是将 CAN_MOD.RM 清零。

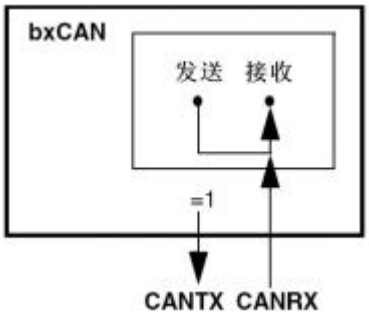
图 26-2 正常模式示意图



CAN 模块为了更加方便自测，增加了静默模式和环回模式，以及两者的组合模式。

静默模式

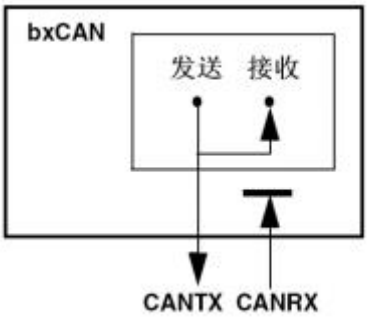
图 26-3 静默模式示意图



在该模式下，CAN 模块不能往总线上发送数据，但是可以接收自己发送的数据（经 CAN 模块内部），也可以接收总线上其他节点发送的数据。

环回模式

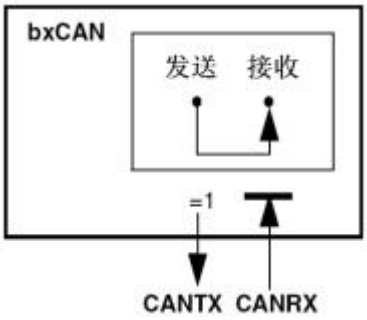
图 26-4 环回模式示意图



在该模式下，CAN 模块可以往总线上发送数据，只能接收自己发送的数据（经 CAN 模块内部），不能接收总线上其他节点发送的数据。

环回静默模式

图 26-5 环回静默模式示意图



在该模式下，CAN 模块既不能往总线上发送数据，也不能接收总线上其他节点发送的数据。只能接收自己发送的数据（经 CAN 模块内部）

如果 Bus 是空闲的，并且没有中断被挂起，那么可以将 CAN 模块进入 Sleep 模式。进入 Sleep 模式的方法是置位 CAN_MOD.SM。TX0 在 Sleep 模式时为高。

可以通过以下方式唤醒：

1）将 CAN_MOD.SM 置为 0

2）在 RX0 上检查的数据

3）NINT_IN 有一个低电平，NINT_IN 为 CAN 中断信号线，即有任意 CAN 中断触发，将会唤醒 Sleep 模式

唤醒后，CAN 会产生一个 Wake-Up 中断 WUI。

CAN 可以支持监听模式和自测模式，在 Reset Mode 和 Operating Mode 两种模式下都可使能。

开启监听模式，会使节点 强制进入“Error Passive”模式，并且只能用于接收数据，不能发送。CAN 在成功接收到一帧数据后也不会回复 ACK。在这种模式下，可以通过软件的方式来实现总线上的波特率检测，使 CAN 支持 hot plug-in 模式。

在自测试模式下，发送数据时不需要从远程 CAN 节点获取 ACK，从而可以用来接收自己发送的数据，实现自发自收。

CAN 也提供了一种时钟输出模式，但是只能在 Reset Mode 时选择，TX0 用来输出发送数据时的波特率时钟信号而不是数据（参考 CAN_OCR.TX0_SEL）。

26.3.4. 消息缓存

表 26-1 发送消息缓存结构

Standard Frame Format（SFF）		Extended Frame Format（EFF）	
CAN Address	Field	CAN Address	Field
60h	TX Frame Information	60h	TX Frame Information
64h	TX Identifier 1	64h	TX Identifier 1
68h	TX Identifier 2	68h	TX Identifier 2
6Ch	TX Data Byte 1	6Ch	TX Identifier 3
70h	TX Data Byte 2	70h	TX Identifier 4
74h	TX Data Byte 3	74h	TX Data Byte 1
78h	TX Data Byte 4	78h	TX Data Byte 2

7Ch	TX Data Byte 5	7Ch	TX Data Byte 3
80h	TX Data Byte 6	80h	TX Data Byte 4
84h	TX Data Byte 7	84h	TX Data Byte 5
88h	TX Data Byte 8	88h	TX Data Byte 6
8Ch	（Unused）	8Ch	TX Data Byte 7
90h	（Unused）	90h	TX Data Byte 8

表 26-2 接收消息缓存结构

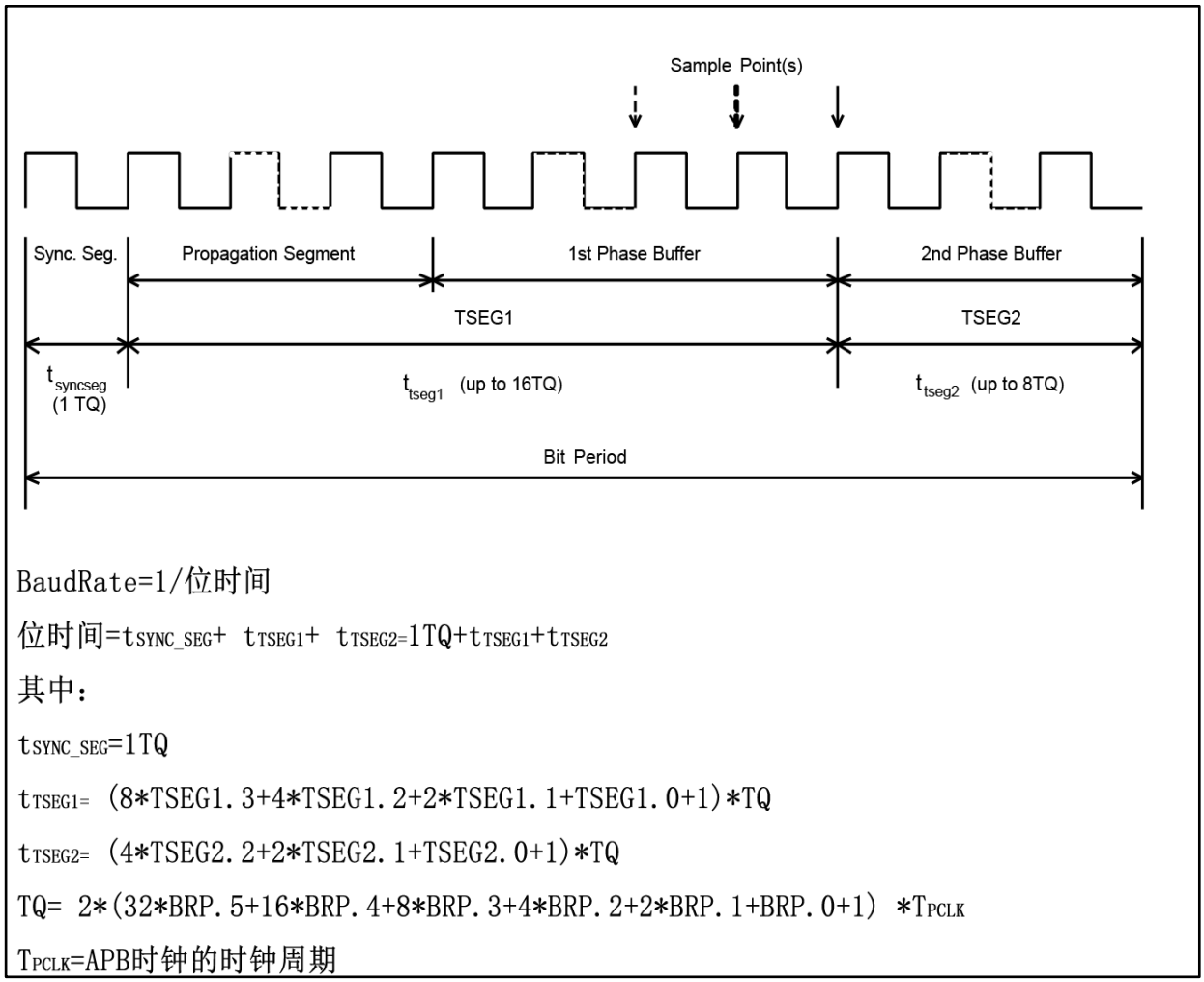
Standard Frame Format（SFF）		Extended Frame Format（EFF）	
CAN Address	**Field**	**CAN Address**	**Field**
60h	RX Frame Information	60h	RX Frame Information
64h	RX Identifier 1	64h	RX Identifier 1
68h	RX Identifier 2	68h	RX Identifier 2
6Ch	RX Data Byte 1	6Ch	RX Identifier 3
70h	RX Data Byte 2	70h	RX Identifier 4
74h	RX Data Byte 3	74h	RX Data Byte 1
78h	RX Data Byte 4	78h	RX Data Byte 2
7Ch	RX Data Byte 5	7Ch	RX Data Byte 3
80h	RX Data Byte 6	80h	RX Data Byte 4
84h	RX Data Byte 7	84h	RX Data Byte 5
88h	RX Data Byte 8	88h	RX Data Byte 6
8Ch	（Unused）	8Ch	RX Data Byte 7
90h	（Unused）	90h	RX Data Byte 8

26.3.5. Bit 时序和波特率配置

CAN 总线的波特率是通过每一个数据位的时序分解，分解的最小时间单位是 TQ，根据 BOSCH 标准，一个完整的位通常由 8-25 个 TQ 组成。

CAN 模块定义的位时序如图所示，共有三部分组成：

图 26-6 CAN 位时序图



BaudRate=1/位时间

位时间=t_{SYNC_SEG}+ t_{TSEG1}+ t_{TSEG2}=1TQ+t_{TSEG1}+t_{TSEG2}

其中：

t_{SYNC_SEG}=1TQ

t_{TSEG1}= (8*TSEG1.3+4*TSEG1.2+2*TSEG1.1+TSEG1.0+1)*TQ

t_{TSEG2}= (4*TSEG2.2+2*TSEG2.1+TSEG2.0+1)*TQ

TQ= 2*(32*BRP.5+16*BRP.4+8*BRP.3+4*BRP.2+2*BRP.1+BRP.0+1) *T_{PCLK}

T_{PCLK}=APB时钟的时钟周期

- 同步段(SYNC_SEG)：通常期望位的变化发生在该时间段内，其值固定为 1TQ
- 时间段 1(TSEG1)：定义采样点的位置。它包含 CAN 标准里的 PROP_SEG 和 PHASE_SEG1。 其值可以编程为 1 到 16 个 TQ。
- 时间段 2(TSEG2)：定义发送点的位置。它代表 CAN 标准里的 PHASE_SEG2。其值可以编程为 1 到 8 个 TQ

重新同步跳跃宽度(SJW)定义了在每位中可以延长或缩短多少个时间单元的上限，其值可以为 1 到 4 个 TQ。

如设置波特率为 500K，系统时钟 64M,APB 时钟为 64M，则 BRP=7，TQ=2*(7+1)*(1/64M)= 16*(1/64M)= 1/4M，设置 TSEG1=2，TSEG2=3，则数据位时间=1+(2+1)+(3+1)=8TQ，则 baud=1/8TQ=1/(8*1/4M)= 1/(2M)=500Kbps。

26.3.6. 仲裁机制

当 CAN 节点出现仲裁失败后，CAN_IR.ALI 将置 1，同时仲裁失败的位置将体现在 CAN_ECCR.ALC。如果开启了仲裁失败中断，CAN_IER.ALIE = 1，那么将产生仲裁失败中断。仲裁失败后，CAN 将在当前帧完成传输后进行自动重发。

26.3.7. 消息接收的过滤

在 CAN 总线中，所有的节点接收总线上的所有报文。为了让节点忽略与它无关的报文信息，CAN 允许对接收

的报文进行过滤，这是通过 7 组 32 位的接收过滤器来实现的。只有报文的标识符与过滤器匹配时，才会被写入到 CAN_RXFIFO。

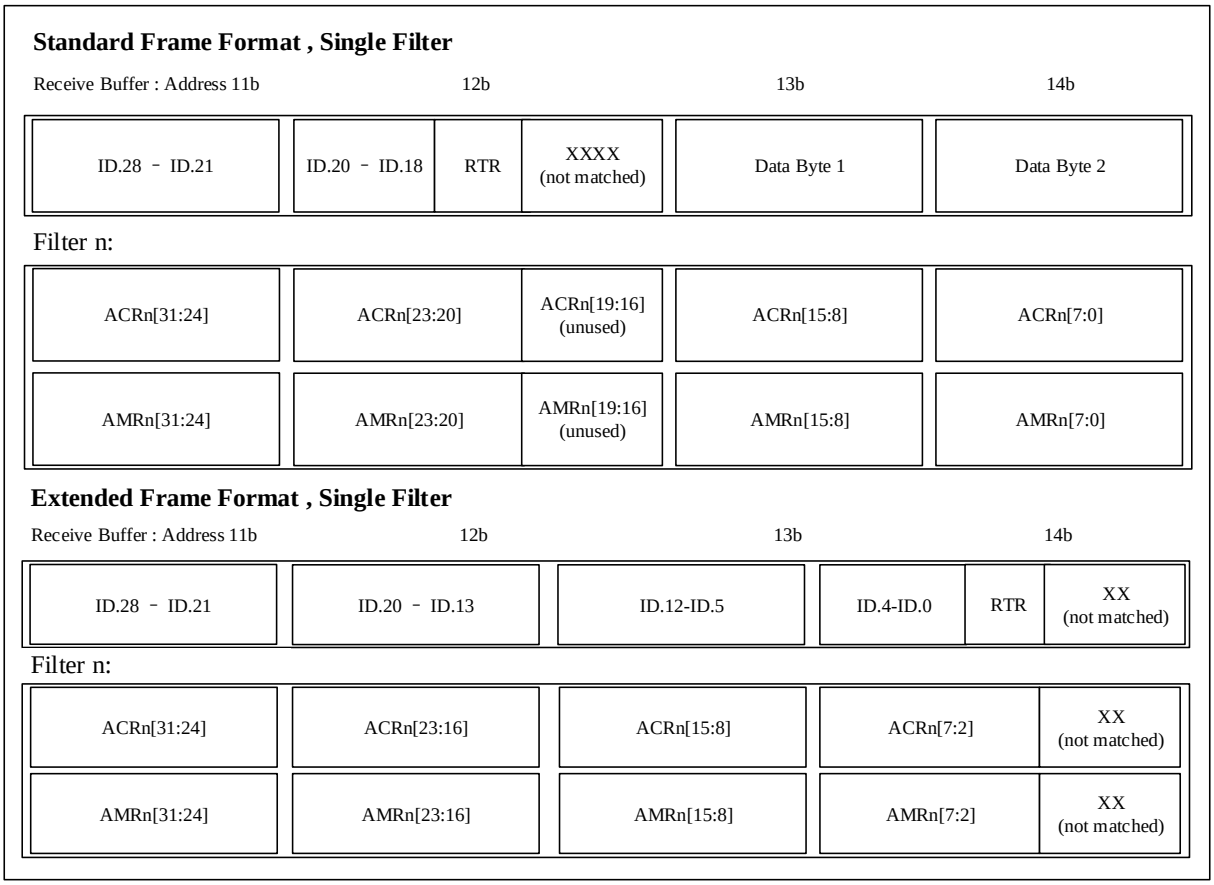
过滤器的实现是通过 CAN_ACRx 和 CAN_AMRx（x 为 0~6）共同完成的，CAN_ACR0 和 CAN_AMR0 是第一组接收过滤器，CAN_ACR6 和 CAN_AMR6 是第七组接收过滤器。

CAN_ACRx 用于存放需要被接收的总线上的帧 ID 号，而 CAN_AMRx 是一个屏蔽寄存器。可以通过设置 CAN_AMRx 里相应的位来决定是否过滤 CAN_ACRx 中对应的位，当 CAN_AMRx 中的某个位为 1 时，表示“Don’ t Care”，不对 CAN_ACRx 中相对应的位进行过滤；当 CAN_AMRx 中的位为 0 时，表示对 CAN_ACRx 中相对应的位进行过滤，只有当接收到的 CAN 报文 ID 对应的位与 CAN_ACRx 中对应的位一致，报文才能被接收进来写入到 CAN_RXFIFO。

CAN 一共有两种过滤模式：

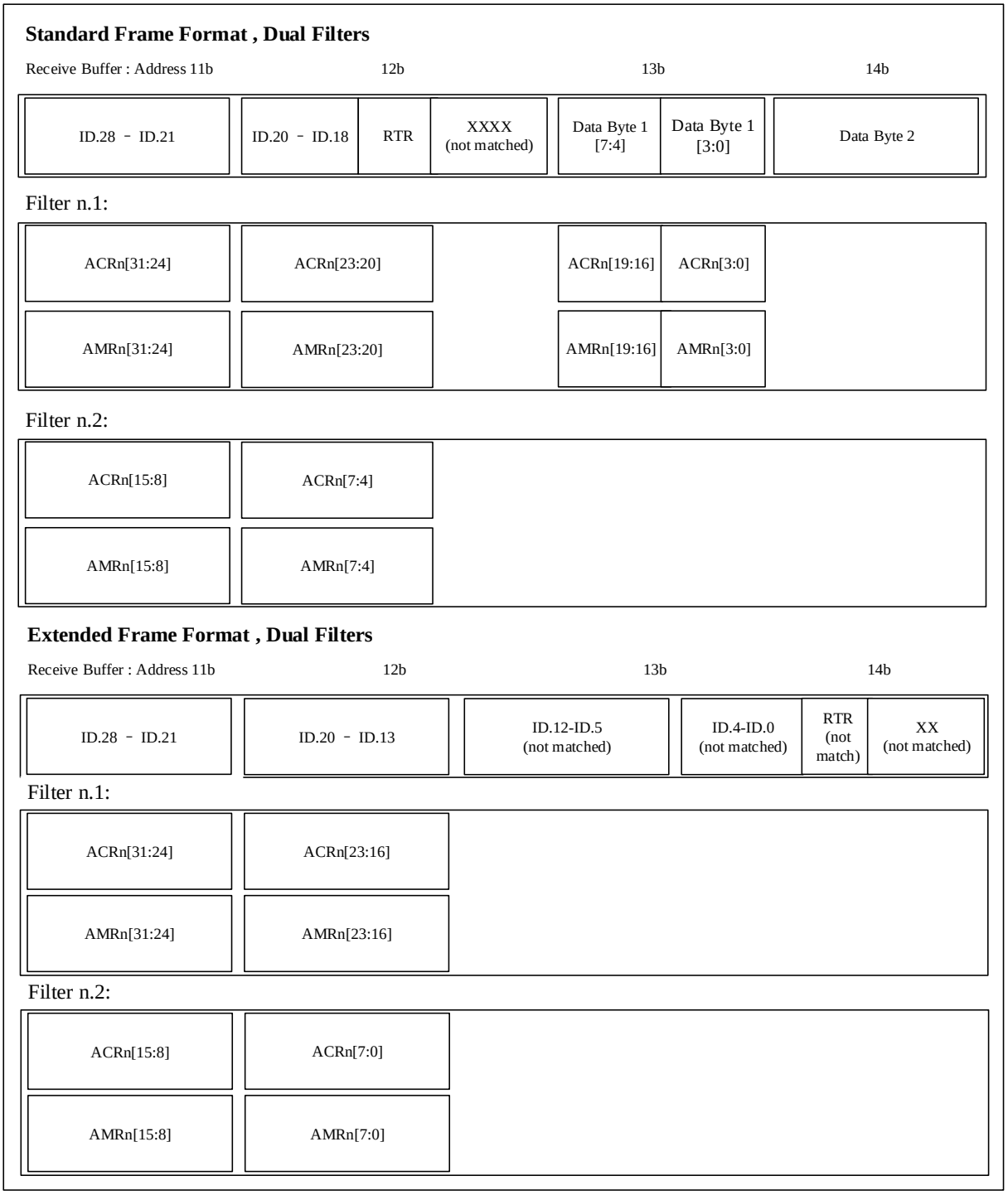
- 单过滤模式：CAN_ACRx 和 CAN_AMRx 配合作为一个 32 位的过滤器，过滤一组 ID 和数据，根据标准帧和扩展帧 ID 长度不同，单过滤模式过滤器结构也不同，结构如图所示。

图 26-7 单过滤模式过滤器示意图



- 双过滤模式：CAN_ACRx 和 CAN_AMRx 配合后分别作为两个 16bits 的过滤器，过滤两组不同格式的 ID 和数据。如果使能 2 个过滤器，那么接收的报文只要符合其中一个条件，就可以被接收进来，双过滤模式过滤器结构如图所示。

图 26-8 双过滤模式过滤器示意图



26.3.8. 消息的接收

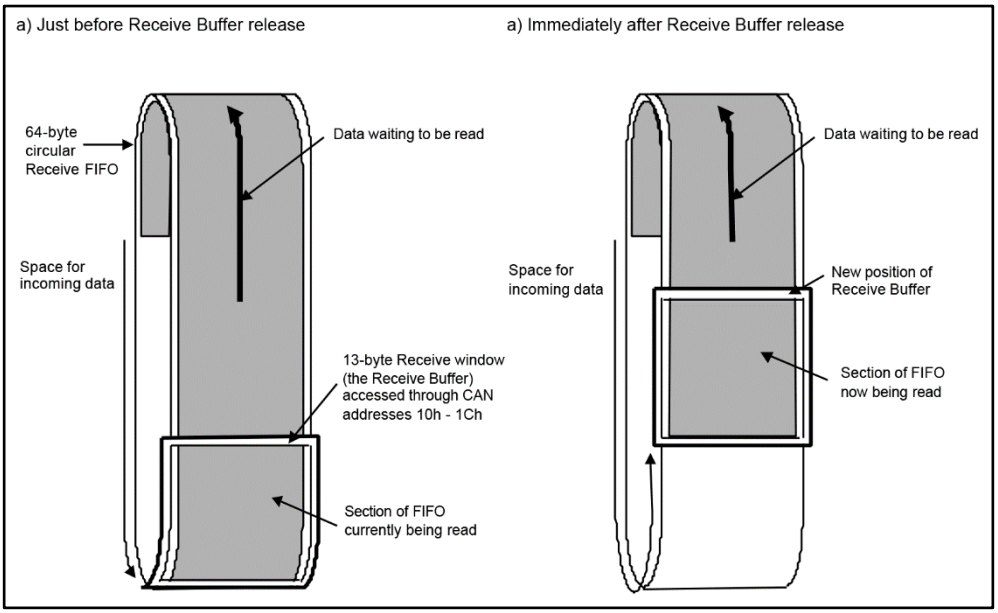
CAN 接收的数据首先通过接收滤波器才能写入到 CAN_RXFIFO。接收滤波器只会通过那些标识符符合条件的报文。

CAN 模块正在接收总线上数据时，CAN_SR.RS 位会被置位。一旦正确接收到一帧数据并且通过接收过滤器放入 CAN_RXFIFO 后，CAN_SR.RBS 位就会被置位，同时产生一个接收中断。

CAN_RXFIFO 是 64 字节深度，最多允许存放 5 组 EFF 数据。如果 CAN_RXFIFO 中没有空间接收新的数据，那么新的数据进来的话就会触发一次 Over Run，CAN_SR.DOS（bit1）位会被置位，接收的数据也会被丢弃，也产生一个中断。

放在 CAN_RXFIFO 中的数据通过一个 13 个字节的窗口 CAN_RXBUFF（地址空间是基于 CAN 模块的偏移 0x60~0x90）来读取。由于 CAN_RXFIFO 中可能存放多组报文，因此为了读取所有的报文，CPU 需要移动窗口，这是通过设置 CAN_CMR.RRB 位来实现的。如果还有数据需要被读取，那么这个数据将会被移动到窗口，如果没有数据被读取了，那么接收标志就会自动清零，接收过程如图所示。

图 26-9 CAN 接收示意图



CAN_RXBUFF 的数据结构如图所示，由 13 个字节组成，接收时只需按照 CAN_RXBUFF 结构读取相应的 RTR（远程帧 1/数据帧 0）、DLC（数据长度）、FF（标准帧 0/扩展帧 1）、ID 以及数据即可。

表 26-3 CAN_RXBUFFx 数据结构

Standard Frame Format（SFF）		Extended Frame Format（EFF）	
CAN Address	Field	CAN Address	Field
60h	RX Frame Information	60h	RX Frame Information
64h	RX Identifier 1	64h	RX Identifier 1
68h	RX Identifier 2	68h	RX Identifier 2
6Ch	RX Data Byte 1	6Ch	RX Identifier 3
70h	RX Data Byte 2	70h	RX Identifier 4
74h	RX Data Byte 3	74h	RX Data Byte 1
78h	RX Data Byte 4	78h	RX Data Byte 2
7Ch	RX Data Byte 5	7Ch	RX Data Byte 3
80h	RX Data Byte 6	80h	RX Data Byte 4
84h	RX Data Byte 7	84h	RX Data Byte 5
88h	RX Data Byte 8	88h	RX Data Byte 6
8Ch	（Unused）	8Ch	RX Data Byte 7
90h	（Unused）	90h	RX Data Byte 8

Transmit Frame（SFF）

CAN Address	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
60h	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
64h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
68h	ID.20	ID.19	ID.18	RTR	0	0	0	0

Transmit Frame（SFF）								
CAN Address	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
60h	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
64h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
68h	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
6Ch	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
70h	ID.4	ID.3	ID.2	ID.1	ID.0	RTR	0	0

CAN 可以接收自己发送给其他节点的数据。通过 CAN_CMR.SRR 位来使能此功能。CAN 自动产生发送和接收中断。

此功能的作用在于让 CAN 总线可以直接同时接收和发送，而不需要其他节点配合，方便测试。

26.3.9. 消息的发送

CAN 消息的发送有两种：一种是自动重发模式，一种是单次发送模式。

- 自动重发模式
  - 设置 CAN_CMR.TR 位，将以自动重发模式进行发送
- 单次发送模式
  - 同时设置 CAN_CMR.TR 位和 CAN_CMR.AT 位，将以单次模式来进行发送
  - 同时设置 CAN_CMR.SRR 位和 CAN_CMR.AT 位，将以单次自我接收模块进行发送

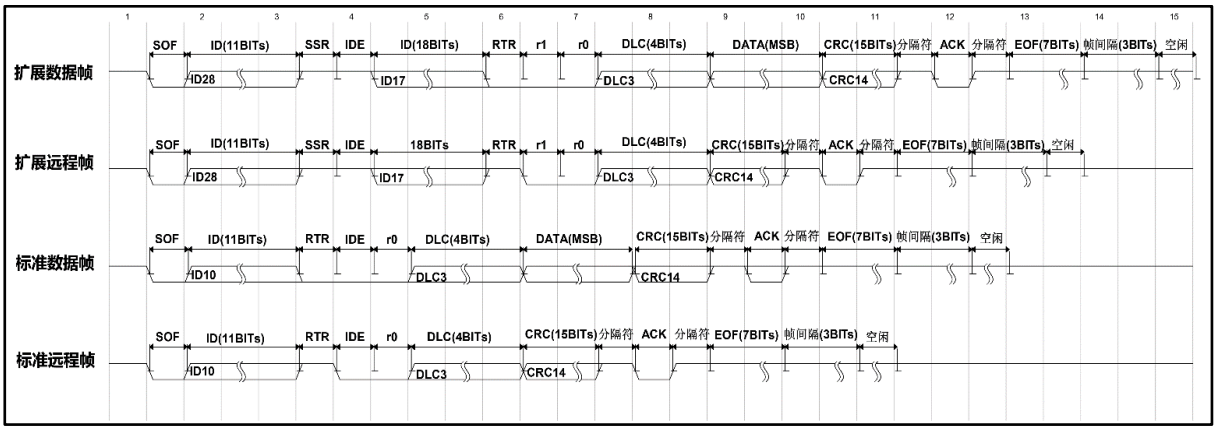
发送流程如下：

- 检查状态寄存器 CAN_SR 中发送状态
- 将要发送的数据（包括长度、ID、Data)写入到发送缓存写寄存器 CAN_TXBUFF 中
- 使能命令寄存器 CAN_CMR 中的发送请求(取决于是自动重发模式还是单次发送模式)，开始发送数据
- 等待状态寄存器 CAN_SR 中的发送完成置位，发送完成

根据 CAN 数据帧的结构，如下图所示。将需要发送的帧类型、帧长度、帧数据写入到 CAN_TXBUFF 中，不管数据帧格式是标准帧还是扩展帧，CAN_TXBUFF 由 13 个字节的数据组成，这样可以保证能写入一个数据长度为 8 字节的数据帧。需要注意的是，在向 CAN_TXBUFF 写数据之前需要先检查一下 CAN_SR.TBS 状态，确保 CAN_TXBUFF 被释放了，否则写入的数据就丢失。

写入到 CAN_TXBUFF（只写）中的帧，可以通过 CAN_TXFIFO（只读）来读取。

图 26-10 CAN 数据帧结构



为了将存放在 CAN_TXBUFF 里面的数据发送出去，可以通过设置 CAN_CMR.TR 发送请求位或者设置 CAN_CMR.SRR 自接收请求位。开始发送时，CAN_SR.TS 被置为 1 并且发送请求（TR 或 SRR）被清除。

发送的比特流是通过 TX0 发送出去的，如果遇到仲裁失败或者发送错误，CAN 能够自动重发。

在每个数据帧后都会自动发送一个 15bits 的 CRC 校验值。CRC 是根据 SOF、仲裁域、控制域和数据域产生。

如果发送还没开始的话，可以通过将 CAN_CMR.AT 置 1 来中断一帧数据的发送，但是一旦开始了，就不能中断发送过程了。

CAN_TXBUFF 的数据结构如图所示，由 13 个字节组成（地址空间是基于 CAN 模块的偏移 0x60~0x90），发送时只需按照 CAN_TXBUFF 结构填入相应的 RTR（远程帧 1/数据帧 0）、DLC（数据长度）、FF（Frame Format）（标准帧 0/扩展帧 1）、ID 以及数据即可。

表 26-4 CAN_TXBUFFx 数据结构

Standard Frame Format（SFF）		Extended Frame Format（EFF）	
CAN Address	**Field**	**CAN Address**	**Field**
60h	TX Frame Information	60h	TX Frame Information
64h	TX Identifier 1	64h	TX Identifier 1
68h	TX Identifier 2	68h	TX Identifier 2
6Ch	TX Data Byte 1	6Ch	TX Identifier 3
70h	TX Data Byte 2	70h	TX Identifier 4
74h	TX Data Byte 3	74h	TX Data Byte 1
78h	TX Data Byte 4	78h	TX Data Byte 2
7Ch	TX Data Byte 5	7Ch	TX Data Byte 3
80h	TX Data Byte 6	80h	TX Data Byte 4
84h	TX Data Byte 7	84h	TX Data Byte 5
88h	TX Data Byte 8	88h	TX Data Byte 6
8Ch	（Unused）	8Ch	TX Data Byte 7
90h	（Unused）	90h	TX Data Byte 8

Transmit Frame（SFF）								
CAN Address	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0

60h	FF	RTR	X（1）	X（1）	DLC.3	DLC.2	DLC.1	DLC.0
64h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
68h	ID.20	ID.19	ID.18	X（2）	X（1）	X（1）	X（1）	X（1）

Transmit Frame（SFF）								
CAN Address	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
60h	FF	RTR	X（1）	X（1）	DLC.3	DLC.2	DLC.1	DLC.0
64h	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
68h	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
6Ch	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
70h	ID.4	ID.3	ID.2	ID.1	ID.0	X（2）	X（1）	X（1）

注：X（1）表示不关心，但建议为 0

X（2）表示不关心，但建议和 CAN_RXBUFFx 中的 RTR 位匹配

26.3.10. 错误处理

CAN_ERRCNTR 中包括两个错误计数器，接收错误计数器 RXERR 和发送错误接收器 TXERR。并且错误的类型和错误的位置都可以在 CAN_ECCR（Error-Code-Capture-Register）中看到。CAN_ERRCNTR 中的 EWL 域，其值表示接收或者发送的错误值达到多少时产生一个警告。默认的 EWL 值为 96，不管接收错误还是发送错误达到这个值都会产生一个错误警告中断。

如果错误计数超过 127，那么 CAN 就会进入“Error Passive”状态。如果发送错误计数器超过 255，那么 CAN_SR.BS 就会被置 1（Bus Off）。CAN 就会进入 Reset Mode 并且会产生一个 EI 中断。在重现进入 Bus On 状态前，CAN 必须等待 128 个 Bus-Free-Sequence。

CAN 错误码表详见错误代码获取寄存器(CAN_ECCR: 1Ch)

26.3.11. 中断

表 26-5 中断

中断事件	使能控制位	事件标识	标识位清除
总线错误中断	BEIE	BEI	读清 0
仲裁失败	ALI	ALIE	读清 0
Error-Passive 中断	EPI	EPIE	读清 0
唤醒中断	WUI	WUIE	读清 0
数据过载中断	DOI	DOIE	读清 0
错误报警中断	EI	EIE	读清 0
发送中断	TI	TIE	读清 0
接收中断	RI	RIE	读清 0

26.4. 配置流程

26.4.1. CAN 初始化

1）配置 CAN 的工作模式（CAN_MOD）

2）配置 CAN 波特率（BAN_BTR）

3）配置 CAN 接收器滤波（CAN_ACRx，CAN_AMRx）以及 CAN_MOD 里的 FACTx（x：0~6）

4）配置 CAN 相关中断（CAN_IER）

26.4.2. 、CAN 消息帧发送

1）准备 CAN 消息帧，包括 CAN ID，IDE，RTR，DLC 及数据内容

2）将以上消息帧信息按照 TXBUFF 的结构写入寄存器 CAN_TXBUFFx

3）启动发送：

- 重发模式：设置 CAN_CMR.TR（自动重发）或 CAN_CMR.SRR（自接收式自动重发）；
- 单次模式：同时设置 CAN_CMR.TR 和 AT（单次发送）或 CAN_CMR.SRR 和 AT（自接收式单次发送）

4）查询是否发送完成（CAN_SR.TCS）。当开启 TIE 中断后，如果发送完成将产生 TI 中断

26.4.3. CAN 消息帧接收

当 CAN 接收到一帧经过过滤的消息后，该消息帧将会存贮在 RXFIFO 中，应用程序可以通过 RXBUFFx 寄存器接口来获取帧消息。

1）查询 CAN_SR.RBS 是否为 1，或等待 RI 中断（当开启了 RIE 中断后）

2）当 RBS=1 或 RI 中断产生后，读取 RXBUFFx。应用程序可以通过 RXBUFF 的数据结构形式来读取并解析帧的头部信息和数据内容

3）释放当前缓存（置位 CAN_CMR.RRB）

4）继续读取下一帧消息，直到 RBS=0

26.5. CAN 寄存器描述

26.5.1. 寄存器列表

CAN1 寄存器基地址：0x40006400

CAN2 寄存器基地址：0x40006800

偏移	名称	正常模式	复位模式	描述
0x00	CAN_MOD	R/W	R/W	模式寄存器
0x04	CAN_CMR	WO	WO	命令寄存器

0x08	CAN_SR	RO	RO	状态寄存器
0x0C	CAN_IR	RO	RO	中断寄存器
0x10	CAN_IER	R/W	R/W	中断使能寄存器
0x14	CAN_BTR	RO	R/W	总线时序控制寄存器
0x18	CAN_OCR	RO	R/W	输出控制寄存器
0x1C	CAN_ECCR	RO	RO	错误抓取寄存器
0x20	CAN_ERRCNTR	RO	R/W	错误计数寄存器
0x24	CAN_RSR	RO	R/W	接受报文状态寄存器
0x28	CAN_ACR0	RO	R/W	接收过滤寄存器 0
0x2C	CAN_ACR1	RO	R/W	接收过滤寄存器 1
0x30	CAN_ACR2	RO	R/W	接收过滤寄存器 2
0x34	CAN_ACR3	RO	R/W	接收过滤寄存器 3
0x38	CAN_ACR4	RO	R/W	接收过滤寄存器 4
0x3C	CAN_ACR5	RO	R/W	接收过滤寄存器 5
0x40	CAN_ACR6	RO	R/W	接收过滤寄存器 6
0x44	CAN_AMR0	RO	R/W	接收屏蔽寄存器 0
0x48	CAN_AMR1	RO	R/W	接收屏蔽寄存器 1
0x4C	CAN_AMR2	RO	R/W	接收屏蔽寄存器 2
0x50	CAN_AMR3	RO	R/W	接收屏蔽寄存器 3
0x54	CAN_AMR4	RO	R/W	接收屏蔽寄存器 4
0x58	CAN_AMR5	RO	R/W	接收屏蔽寄存器 5
0x5C	CAN_AMR6	RO	R/W	接收屏蔽寄存器 6
0x60~0x90	CAN_TXBUFF	WO	--	发送缓存寄存器（写操作）
0x60~0x90	CAN_RXBUFF	RO	RO	接收缓存寄存器（读操作）
0xA0~0x19C	CAN_RX_FIFO	RO	R/W	接收 FIFO
0x1A0~0x1D0	CAN_TX_FIFO	RO	RO	发送 FIFO

26.5.2. 模式寄存器(CAN_MOD: 00h)

位域	名称	属性	复位值	描述
31:23	RSV	-	-	保留
22	FACT6	R/W	0	过滤器 6 使能位 0：禁止过滤器 1：使能过滤器
21	FACT5	R/W	0	过滤器 5 使能位 0：禁止过滤器 1：使能过滤器

20	FACT4	R/W	0	过滤器 4 使能位 0：禁止过滤器 1：使能过滤器
19	FACT3	R/W	0	过滤器 3 使能位 0：禁止过滤器 1：使能过滤器
18	FACT2	R/W	0	过滤器 2 使能位 0：禁止过滤器 1：使能过滤器
17	FACT1	R/W	0	过滤器 1 使能位 0：禁止过滤器 1：使能过滤器
16	FACT0	R/W	0	过滤器 0 使能位 0：禁止过滤器 1：使能过滤器
15	RSV	-	-	保留
14	AFM6	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
13	AFM5	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
12	AFM4	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
11	AFM3	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
10	AFM2	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
9	AFM1	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
8	AFM0	R/W	0	接收过滤器模式 0: 双过滤模式，两个 16 位的过滤器 1：单过滤模式，单个 32 位的过滤器
7	RSV	-	-	保留
6	LBKM	R/W	0	环回模式
5	SILM	R/W	0	静默模式

4	SM	R/W	0	睡眠模式 0：正常状态 1：进入睡眠模式。（如果有中断挂起或者总线在传输数据，则立即会被唤醒） 注：仅能在 Operating 模式下进入
3	RSV	-	-	保留
2	STM	R/W	0	发送时不检查 ACK 0：检测 ACK。 1：无需检测 ACK
1	LOM	R/W	0	监听模式使能 0：正常模式，错误计数器停止工作 1：监听模式，会使节点强制进入 Error-passive 模式。CAN 接收到正确的数据帧也不会回发 ACK
0	RM	R/W	1	进入复位模式 0：正常模式，此位“1->0”可以进入正常模式 1：进入复位模式，并中断正在发送或接收的数据

26.5.3. 命令寄存器(CAN_CMR: 04h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	SRR	WO	0	自我接收请求 0：无效 1：同步接收自己即将发送的报文。（包含发送功能） 注意：不能和 TR 同时使能，如果 SRR 和 TR 同时使能，SRR 将被忽略
3	CDO	WO	0	清除过载状态 0：无效 1：清除过载状态 CAN_SR.DOS
2	RRB	WO	0	释放接收缓存 0：无效 1：释放接收缓存。
1	AT	WO	0	终止发送 0：无效 1：如果数据还未在总线上传输，则取消该发送请求。如果数据已在传输则无法再终止发送
0	TR	WO	0	发送请求 0：无效 1：发送报文

26.5.4. 状态寄存器(CAN_SR: 08h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	BS	RO	0	总线状态 0：总线处于激活状态 1：总线处于“Bus Off”状态
6	ES	RO	0	错误状态 0：接收和发送错误计数器低于报警值 1：接收或者发送错误计数器大于等于报警值
5	TS	RO	1(Reset) /0(Normal)	发送状态 0：没有报文正在被发送 1：正在发送报文
4	RS	RO	1(Reset) /0(Normal)	接收状态 0：没有报文正在被接收 1：正在接收报文
3	TCS	RO	1	发送完成状态 0：最近一次的发送请求还未完成 1：最近一次的发送请求已经完成
2	TBS	RO	1	发送缓存状态 0：发送缓存锁定状态，报文正在被发送或者等待被发送，CPU 不能访问发送缓存 1：发送缓存释放。CPU 可以访问发送缓存
1	DOS	RO	0	数据过载状态位 0：无数据过载 1：数据过载，因为接收缓存没有空间导致丢失报文
0	RBS	RO	0	接收缓存状态位 0：接收缓存为空 1：接收缓存不为空

26.5.5. 中断寄存器(CAN_IR: 0Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	BEI	RO	0	总线错误中断 0：无中断 1：有中断 注：读清 0

6	ALI	RO	0	仲裁失败中断 0：无中断 1：有中断 注：读清 0
5	EPI	RO	0	Error-Passive 中断 0：无中断 1：有中断 注：读清 0
4	WUI	RO	0	唤醒中断 0: 无中断 1：有中断 注：读清 0
3	DOI	RO	0	数据过载中断 0：无中断 1：有中断 注：读清 0
2	EI	RO	0	错误报警中断 0：无中断 1：有中断 注：读清 0
1	TI	RO	0	发送中断。 0：无中断 1：有中断 注：读清 0
0	RI	RO	0	接收中断 0：无中断 1：有中断 注：读清 0

26.5.6. 中断使能寄存器(CAN_IER: 10h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	BEIE	R/W	0	总线错误中断使能 0：禁止 1：使能
6	ALIE	R/W	0	仲裁失败中断使能 0：禁止 1：使能

5	EPIE	R/W	0	Error-Passive 中断使能 0：禁止 1：使能
4	WUIE	R/W	0	唤醒中断使能 0: 禁止 1：使能
3	DOIE	R/W	0	数据过载中断使能 0：禁止 1：使能
2	EIE	R/W	0	错误报警中断使能 0：禁止 1：使能
1	TIE	R/W	0	发送中断使能。 0：禁止 1：使能
0	RIE	R/W	0	接收中断使能 0：禁止 1：使能

26.5.7. 时序寄存器(CAN_BTR: 14h)

位域	名称	正常模式	复位模式	复位值	描述
31:16	RSV	-		-	保留
15:14	SJW	RO	R/W	0	同步偏移宽度
13:8	BRP	RO	R/W	0	波特率因子 TQ
7	SAM	RO	R/W	0	采样选择 0：采样一次 1：采样三次
6:4	TSEG2	RO	R/W	0	采样点尾部 = TSEG2+1
3:0	TSEG1	RO	R/W	0	采样点前部 = TSEG1+1

26.5.8. 输出寄存器(CAN_OCR: 18h)

位域	名称	正常模式	复位模式	复位值	描述
31:12	RSV	-		-	保留
11	CLOCK OFF	R/W	R/W	0	控制 TX0_SEL 选择 Clock out 时的开启和关闭 0：使能 Clock Out 1：禁止 Clock Out

10:8	CD	R/W	R/W	0	Clock Division。用于 TX0_SEL 选择 Clock out 3’ b000：fosc/2 3’ b001：fosc/4 3’ b010：fosc/6 3’ b011：fosc/8 3’ b100：fosc/10 3’ b101：fosc/12 3’ b110：fosc/14 3’ b111：保留
7:6	RSV	-	-	-	保留
5	CRC_ERR_COD	RO	RW	0	CRC 错误发生时产生的 ECC 值
4:2	TX0_SEL	RO	RW	3’ b000	3’ b000:TX0 3’ b001:~TX0 3’ b010:Simple Time 3’ b011:Tx_clock 3’ b100:Clock out 3’ b101:0 3’ b110:1 3’ b111:1
1:0	RSV	-	-	-	保留

26.5.9. 错误代码获取寄存器(CAN_ECCR: 1Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:13	RSV	-		-	保留
12:8	ALC	RO	RO	0	仲裁失败位置值=ALC + 1
7:6	ERRCODE	RO	RO	0	0b00：比特错误 0b01：形式错误 0b10：填充错误 0b11：其他错误
5	DIRECTION	RO	RO	0	出错方向： 0：发送时出错 1：接收时出错
4:0	SEGCODE	RO	RO	0	出错的段代码，见下表

表 26-6 SegCode 错误码表

SegCode[4:0]	描述	SegCode[4:0]	描述
00011	SOF	01010	数据域
00010	ID.28~ID.21	01000	CRC 域
00110	ID.20~ID.18	11000	CRC 分隔符

00100	SRTR 位	11001	ACK
00101	IDE 位	11011	ACK 分隔符
00111	ID.17~ID.13	11010	EOF
01111	ID.12~ID.5	10010	间隙
01110	ID.4~ID.0	10001	Active Error 标志
01100	RTR 位	10110	Passive Error 标志
01101	保留	10011	容忍显性位
01001	保留	10111	错误分隔符
01011	DLC	11100	过载标志

26.5.10. 错误计数寄存器(CAN_ERRCNTR: 20h)

位域	名称	正常模式	复位模式	复位值	描述
31:24	RSV	-		-	保留
23:16	EWL	RO	R/W	96	错误报警值
15:8	TXERR	RO	R/W	0	发送错误计数寄存器
7:0	RXERR	RO	R/W	0	接收错误计数寄存器

26.5.11. 接收报文状态寄存器(CAN_RSR: 24h)

位域	名称	正常模式	复位模式	复位值	描述
31:14	RSV	--	--	--	保留
13:8	RBSA	RO	R/W	0	RXBUFFx 当前对应于 RXFIFO 的起始地址
7:5	RSV	--	--	--	保留
4:0	RMC	RO	RO	0	接收 FIFO 中已接收的报文个数

26.5.12. 接收过滤寄存器(CAN_ACR0: 28h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR0	--	R/W	0	接收过滤

26.5.13. 接收过滤寄存器(CAN_ACR1: 2Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR1	--	R/W	0	接收过滤

26.5.14. 接收过滤寄存器(CAN_ACR2: 30h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR2	--	R/W	0	接收过滤

26.5.15. 接收过滤寄存器(CAN_ACR3: 34h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR3	--	R/W	0	接收过滤

26.5.16. 接收过滤寄存器(CAN_ACR4: 38h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR4	--	R/W	0	接收过滤

26.5.17. 接收过滤寄存器(CAN_ACR5: 3Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR5	--	R/W	0	接收过滤

26.5.18. 接收过滤寄存器(CAN_ACR6: 40h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	ACR6	--	R/W	0	接收过滤

26.5.19. 接收屏蔽寄存器(CAN_AMR0: 44h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR0	--	R/W	0	接收过滤屏蔽

26.5.20. 接收屏蔽寄存器(CAN_AMR1: 48h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR1	--	R/W	0	接收过滤屏蔽

26.5.21. 接收屏蔽寄存器(CAN_AMR2: 4Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR2	--	R/W	0	接收过滤屏蔽

26.5.22. 接收屏蔽寄存器(CAN_AMR3: 50h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR3	--	R/W	0	接收过滤屏蔽

26.5.23. 接收屏蔽寄存器(CAN_AMR4: 54h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR4	--	R/W	0	接收过滤屏蔽

26.5.24. 接收屏蔽寄存器(CAN_AMR5: 58h)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR5	--	R/W	0	接收过滤屏蔽

26.5.25. 接收屏蔽寄存器(CAN_AMR6: 5Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:0	AMR6	--	R/W	0	接收过滤屏蔽

26.5.26. 发送缓存写寄存器(CAN_TXBUFFx: 只写 60h~90h)

位域	名称	正常模式	复位模式	复位值	描述
31:8	RSV	-		-	保留
7:0	TXBUFFX	WO	--	0	发送缓存的字节 x

26.5.27. 接收缓存读寄存器(CAN_RXBUFFx: 只读 60h~90h)

位域	名称	正常模式	复位模式	复位值	描述

31:8	RSV	-		-	保留
7:0	RXBUFFX	RO	--	0	接收缓存的字节 x

26.5.28. 接收 FIFO 访问寄存器(CAN_RXFIFO: A0h~19Ch)

位域	名称	正常模式	复位模式	复位值	描述
31:8	RSV	--	--	--	保留
7:0	RXFIFOX	RO	R/W	0	接收 FIFO 字节 x (0~63)

26.5.29. 发送 FIFO 访问寄存器(CAN_TXFIFO: 1A0h~1D0h)

位域	名称	正常模式	复位模式	复位值	描述
31:8	RSV	--	--	-	保留
7:0	TXFIFOX	RO	RO	0	发送 FIFO 字节 x (0~12)

27. 外部存储器控制器（EXMC）

27.1. 概述

外部存储器控制器 EXMC，用来访问各种片外存储器。通过配置寄存器，EXMC 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、NOR Flash 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。

27.2. 主要特性

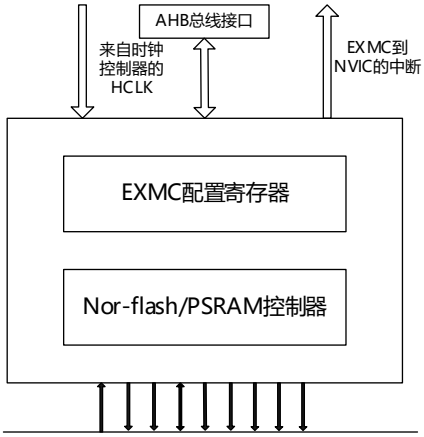
- 支持片外存储器类型：
  - SRAM
  - PSRAM
  - ROM
  - NOR Flash
  - 8080-LCD
- AMBA 协议与各种片外存储器协议转换；
- 时序参数可编程可以满足用户特定需求；
- 对于部分存储器类型支持独立的读写时序；
- 支持 8 位，或 16 位总线带宽；
- NOR Flash 和 PSRAM 支持地址总线和数据总线的复用；
- 提供写使能和字节选择信号；
- 当 AMBA 总线宽度与外部存储器数据宽度不同时，会自动分割操作。
- 支持同步模式、Burst 突发模式

27.3. 功能说明

27.3.1. 结构框图

EXMC 由 4 个模块组成：AHB 总线接口，EXMC 配置寄存器，NOR/PSRAM 控制器和外部设备接口。AHB 时钟（HCLK）是参考时钟。

图 27-1 EXMC 结构框图



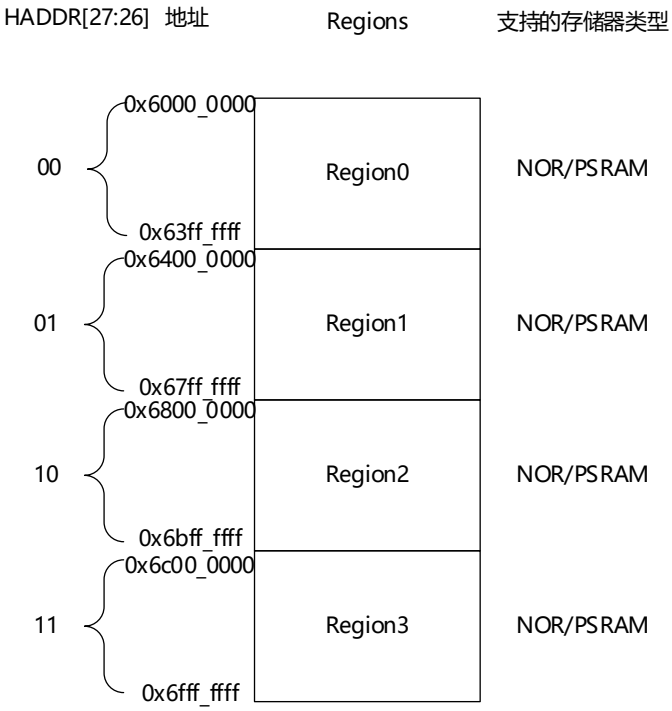
27.3.2. AHB 接口

EXMC 是 AHB 总线至外部设备协议的转换接口。32 位的 AHB 读写操作可以转化为几个连续的 8 位或 16 位读写操作。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，EXMC 读写访问需要遵从以下规范。

- 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，则可通过它的字节通道 EXMC_NBL[1:0]来访问对应的字节。否则禁止写操作，只允许读操作。

27.3.3. 外部设备地址映射

图 27-2 EXMC Bank 划分



EXMC 访问区域为 Bank0，占 64M*4 字节，用于访问 NOR、PSRAM 设备。Bank0 分为四个 Region 的地址映射。AHB 地址线 HADDR[27:26]作为四个 Region 的片选信号。

表 27-1 NOR/PSRAM 存储块选择

HADDR[27:26]	选择的存储块
00	存储块 1 NOR/PSRAM 1
01	存储块 1 NOR/PSRAM 2
10	存储块 1 NOR/PSRAM 3
11	存储块 1 NOR/PSRAM 4

由于 HADDR[25:0]是字节地址，而外部存储器访问有可能不是按字节访问的，所以会出现地址不一致的情况，但 EXMC 能实现对 HADDR 的调整以适应外部存储器的数据宽度。具体规则如下：

- 如果外部存储器的数据宽度是 8 位按字节对齐，HADDR[25:0]与 EXMC_A[25:0]相连，然后用 EXMC_A[25:0]去连接外部存储器的地址线；
- 如果外部存储器的数据宽度是 16 位按半字对齐，就需要将 HADDR 的字节地址转化为半字地址之后再连接外存储器，所以需要将 HADDR[25:1]与 EXMC_A[24:0]相连。然后用 EXMC_A[24:0]去连接外部存储器的地址线。

外部存储器地址:

表 27-2 外部存储器地址

数据宽度	连到存储器的地址线	最大访问存储器空间(位)
8 位	HADDR[25:0]与 EXMC_A[25:0]对应相连	64M 字节 x 8 = 512 M 位
16 位	HADDR[25:1]与 EXMC_A[24:0]对应相连， HADDR[0]未接	64M 字节/2 x 16 = 512 M 位

每个 NOR 闪存或 PSRAM 存储器块都可以配置成支持非对齐的数据访问。
在存储器一侧，依据访问的方式是异步或同步，需要考虑两种情况：

- 异步模式：这种情况下，只要每次访问都有准确的地址，完全支持非对齐的数据访问。
- 同步模式：这种情况下， EXMC 只发出一次地址信号，然后成组的数据传输通过时钟 CLK 顺序进行。

27.3.4. NOR Flash/PSRAM 控制器

控制器控制 Bank0，它可以支持 NOR Flash、PSRAM、SRAM、ROM 和蜂窝 RAM 外部存储器。

EXMC 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， EXMC 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 27-3 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

27.3.5. 外部存储器接口信号

表 27-4 NOR Flash 接口信号描述

EXMC 引脚	传输方向	模式	功能描述
EXMC_CLK	输出	同步	同步时钟信号
非复用 EXMC_A[25:0]	输出	异步/同步	地址总线信号
复用 EXMC_A[25:16]			
EXMC_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
EXMC_NE	输出	异步/同步	片选
EXMC_NOE	输出	异步/同步	输出使能（读使能）
EXMC_NWE	输出	异步/同步	写使能
EXMC_NWAIT	输入	异步/同步	等待输入信号
EXMC_NL（NADV）	输出	异步/同步	地址有效

表 27-5 PSRAM 非复用接口信号描述

EXMC 引脚	传输方向	模式	功能描述
EXMC_CLK	输出	同步	同步时钟信号
EXMC_A[25:0]	输出	异步/同步	地址总线
EXMC_D[15:0]	输入/输出	异步/同步	数据总线
EXMC_NE	输出	异步/同步	片选
EXMC_NOE	输出	异步/同步	输出使能（读使能）
EXMC_NWE	输出	异步/同步	写使能
EXMC_NWAIT	输入	异步/同步	等待输入信号
EXMC_NL（NADV）	输出	异步/同步	锁存（地址有效）使能
EXMC_NBL[1]	输出	异步/同步	高字节使能
EXMC_NBL[0]	输出	异步/同步	低字节使能

27.3.6. 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 27-6 EXMC 的 Bank0 支持的所有传输

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 EXMC 访问
	异步	W	32	16	分 2 次 EXMC 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 EXMC_NBL[1:0]
	异步	R	16	16	
	异步	W	16	16	
	异步	R	32	16	分 2 次 EXMC 访问
	异步	W	32	16	分 2 次 EXMC 访问
	同步	R	16	16	
	同步	R	32	16	
	同步	W	8	16	使用字节信号 EXMC_NBL[1:0]
	同步	W	16	16	
	同步	W	32	16	分 2 次 EXMC 访问
SRAM/ROM	异步	R	8	8	
	异步	R	8	16	
	异步	R	16	8	分 2 次 EXMC 访问
	异步	R	16	16	
	异步	R	32	8	分 4 次 EXMC 访问
	异步	R	32	16	分 2 次 EXMC 访问
	异步	W	8	8	
	异步	W	8	16	使用字节信号 EXMC_NBL[1:0]
	异步	W	16	8	
	异步	W	16	16	
	异步	W	32	8	

	异步	W	32	16	

27.3.7. 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（EXMC_CLK）的下降沿改变。

27.3.8. 异步事务

EXMC 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；
- EXMC 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 27-7 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	EXMC_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

EXMC 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 EXMC 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 EXMC_SNCTL 位 EXMODEN 置 1 使能扩展模式后，可以通过寄存器 EXMC_SNTCFG 和 EXMC_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 27-8 EXMC 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 EXMC_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 EXMC_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

模式 1-SRAM/CRAM

图 27-3 模式 1 读访问

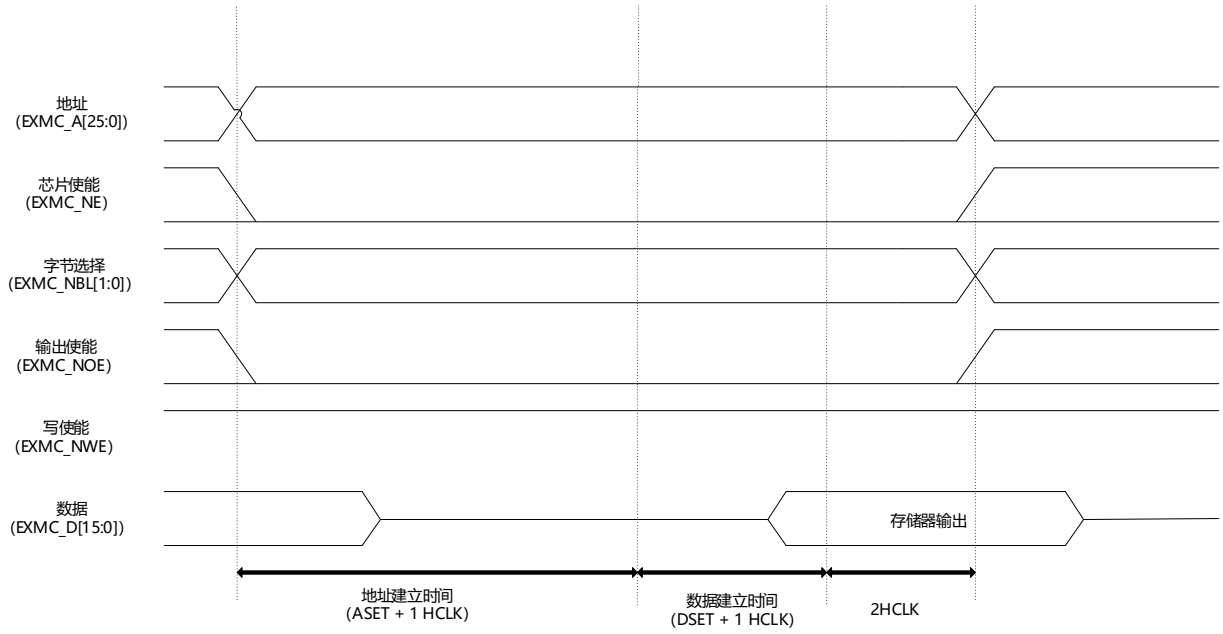


图 27-4 模式 1 写访问

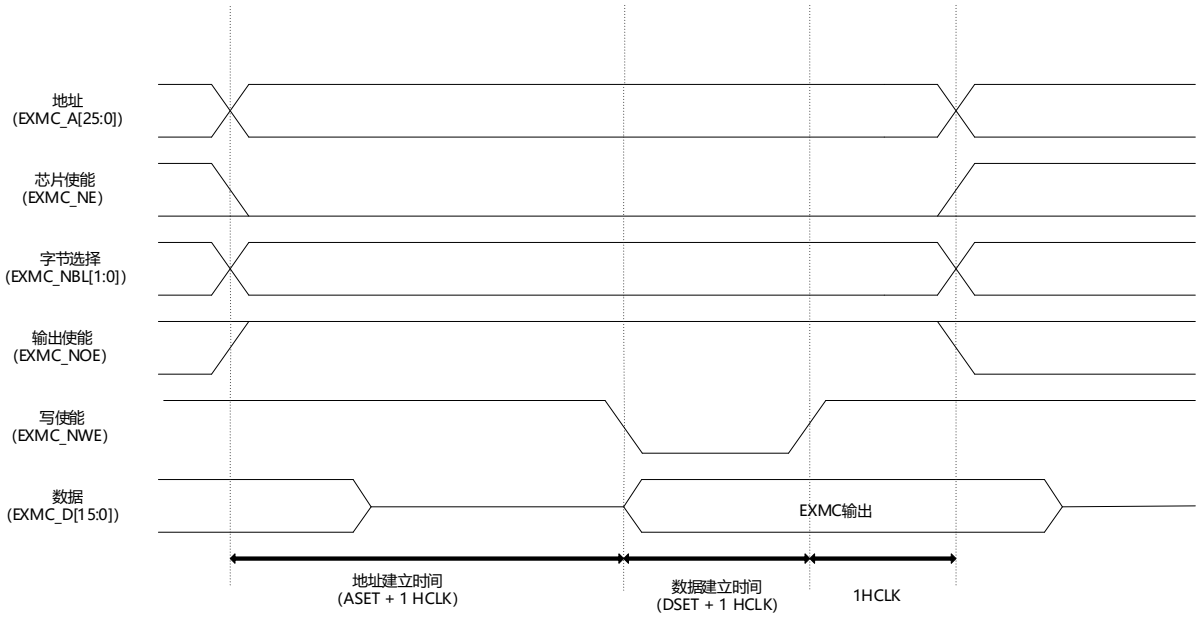


表 27-9 模式 1 相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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 EXMC_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

模式 A - SRAM/PSRAM（CRAM） OE 翻转：

图 27-5 模式 A 写访问

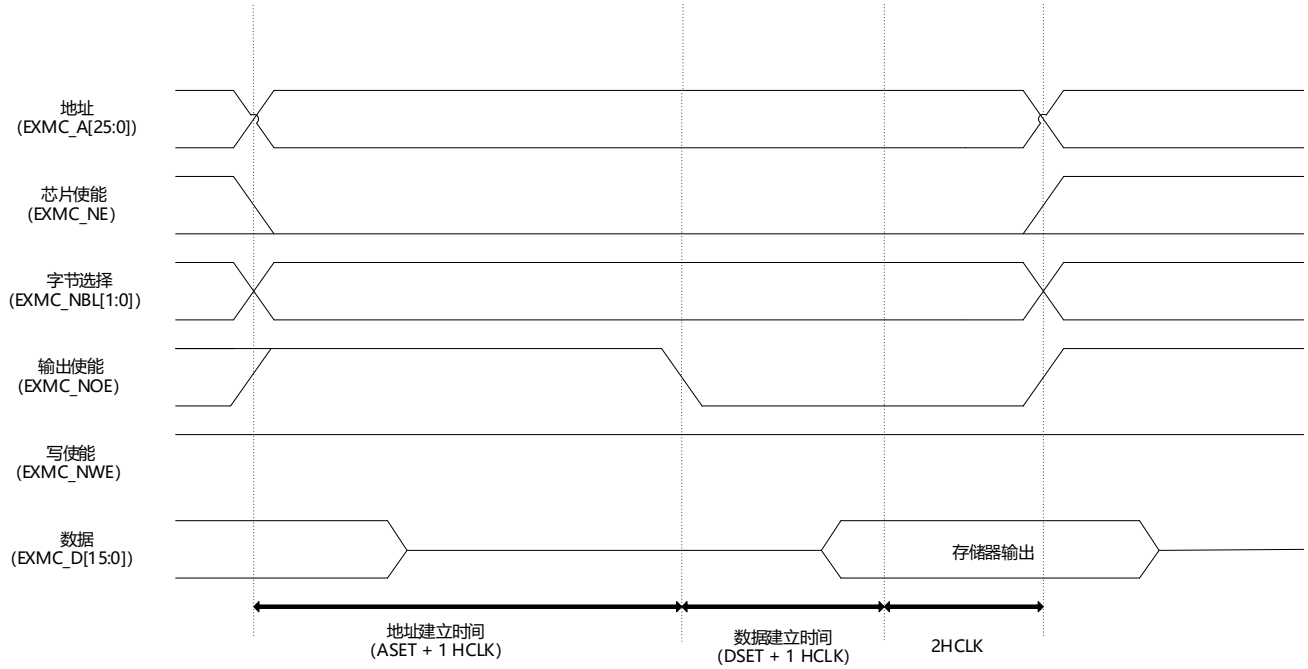
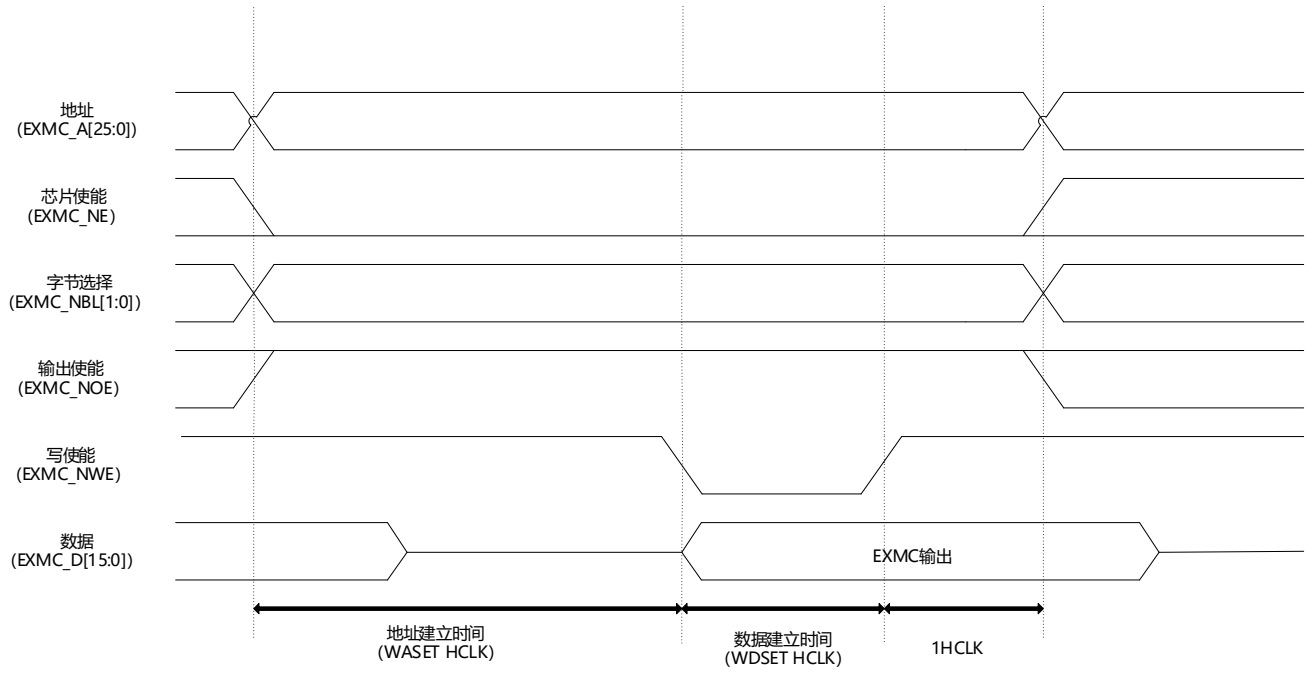


图 27-6 模式 A 读访问



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 27-10 模式 A 相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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 EXMC_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
EXMC_SNWTCFG（写）		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 B：0x1
27:20	RSV	0xFF
19:16	WBUSLAT	从 EXMC_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

模式 2/B-Nor Flash：

图 27-7 模式 2 读访问

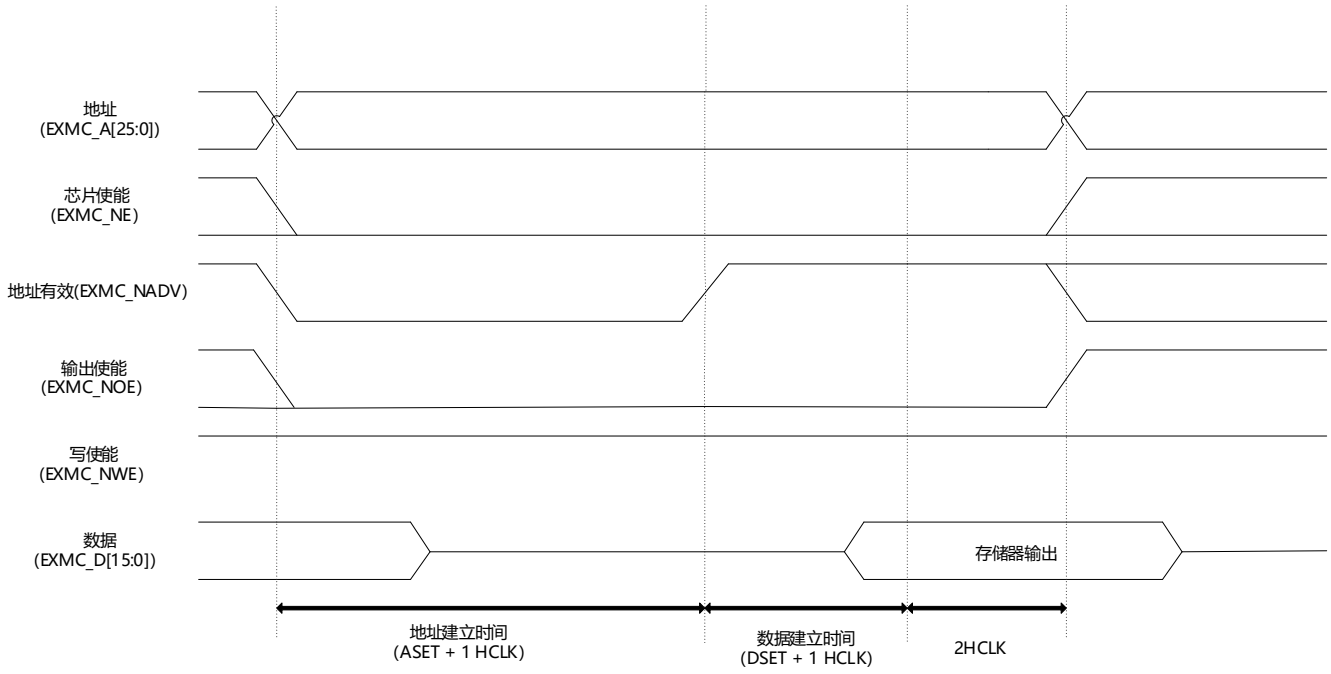


图 27-8 模式 2 写访问

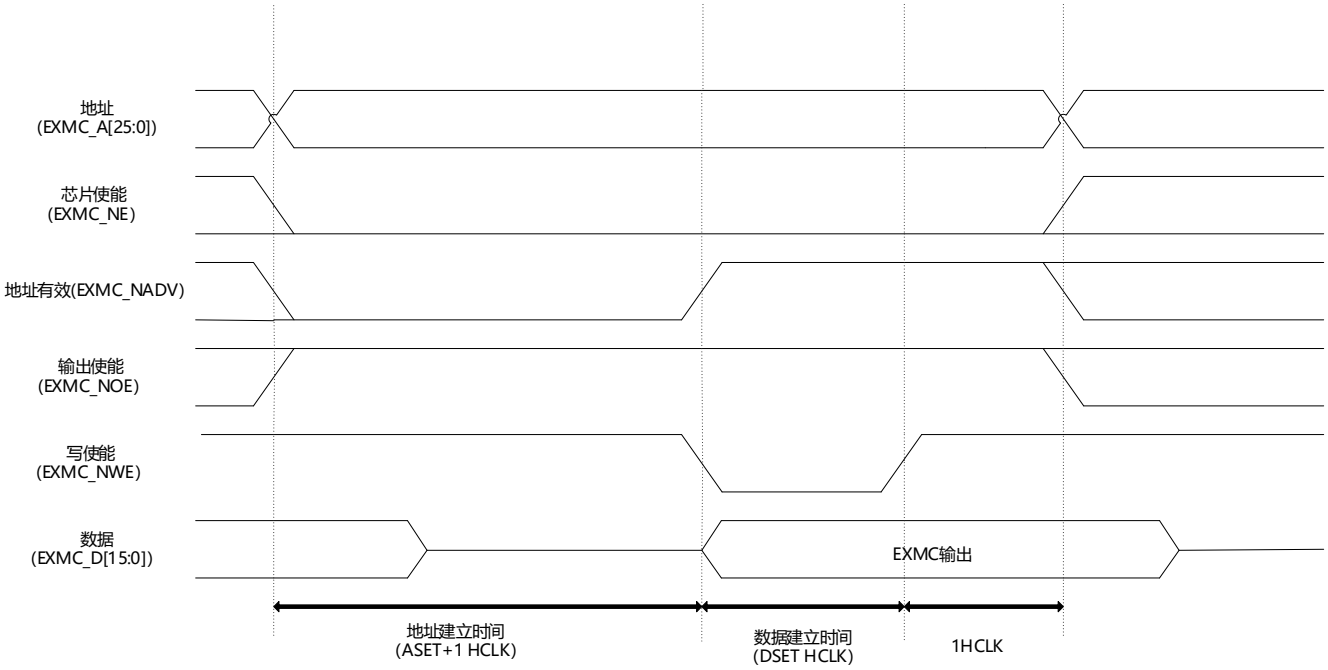
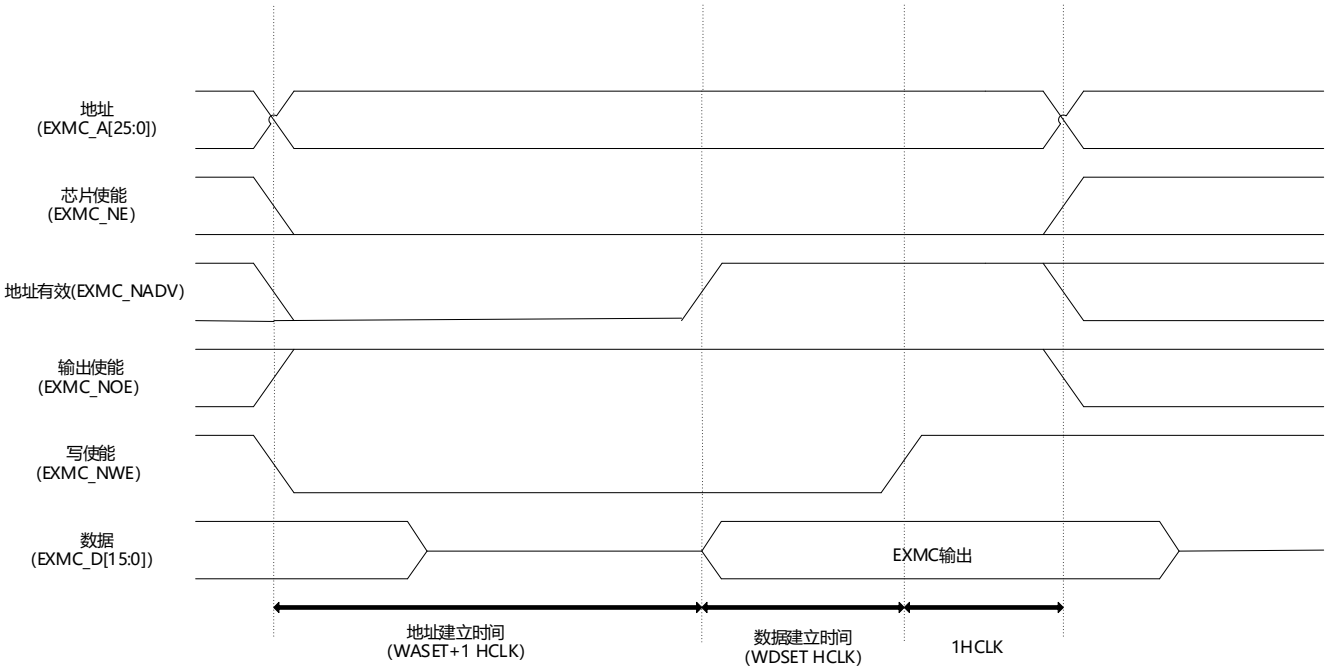


图 27-9 模式 B 写访问



模式 2/B 相关寄存器配置：

表 27-11 模式 2/B 相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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 EXMC_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
EXMC_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 EXMC_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

模式 C-NOR Flash OE 翻转：

图 27-10 模式 C 读访问

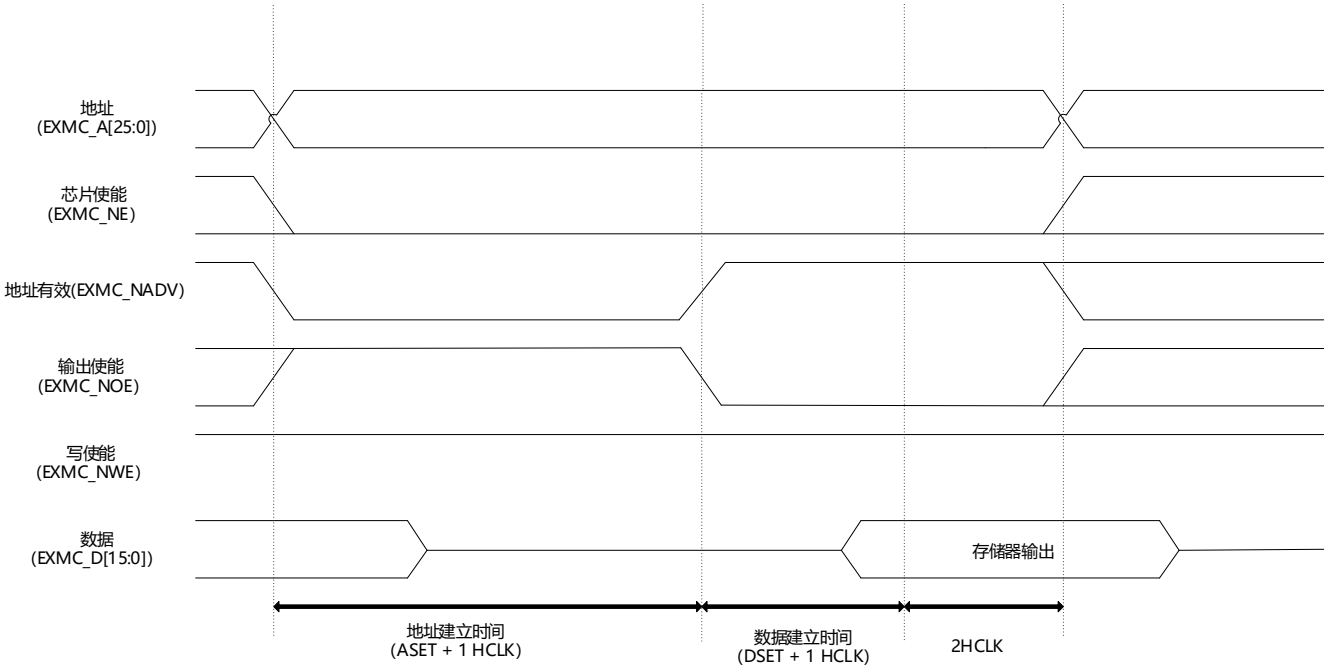
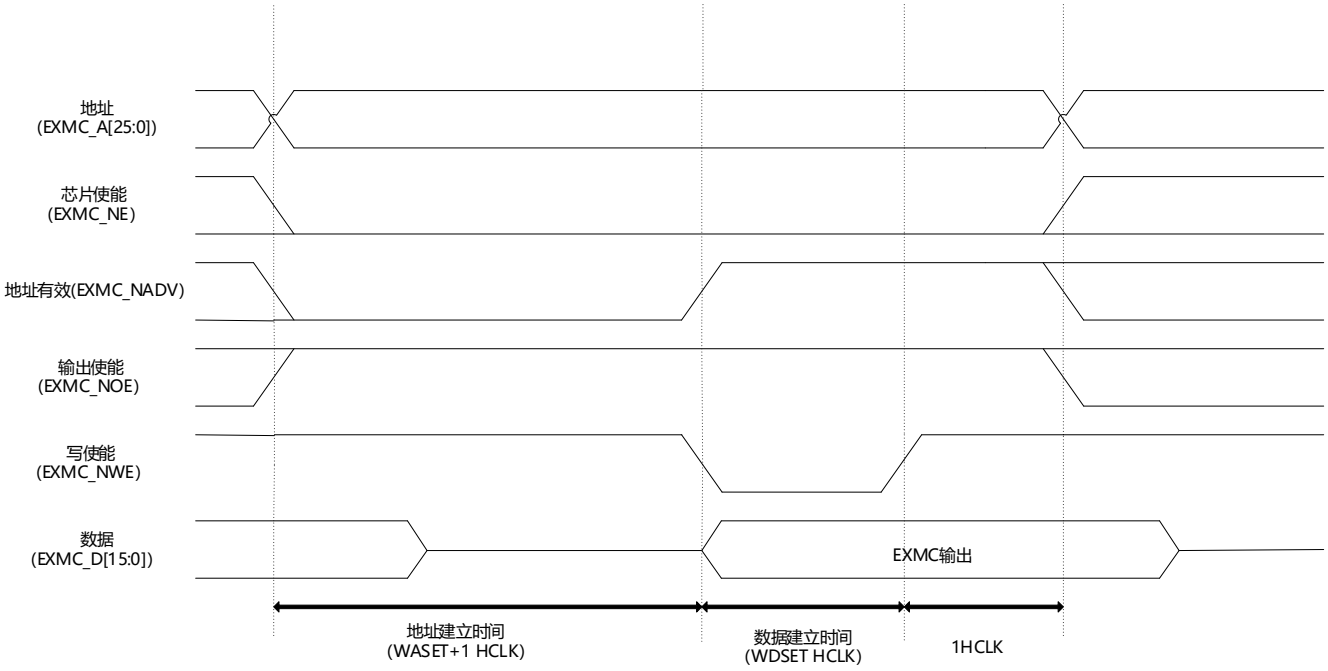


图 27-11 模式 C 写访问



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 27-12 模式 C 相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_SNTCFG		
31:30	RSV	0x0
29:28	ASYNCMOD	模式 C：0x2
27:24	DLAT	0x0
23:20	CKDIV	0x0
19:16	BUSLAT	从 EXMC_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
EXMC_SNWTCFG		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 C：0x2
27:20	RSV	0xFF
19:16	WBUSLAT	从 EXMC_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

模式 D -带地址扩展的异步操作：

图 27-12 模式 D 读访问

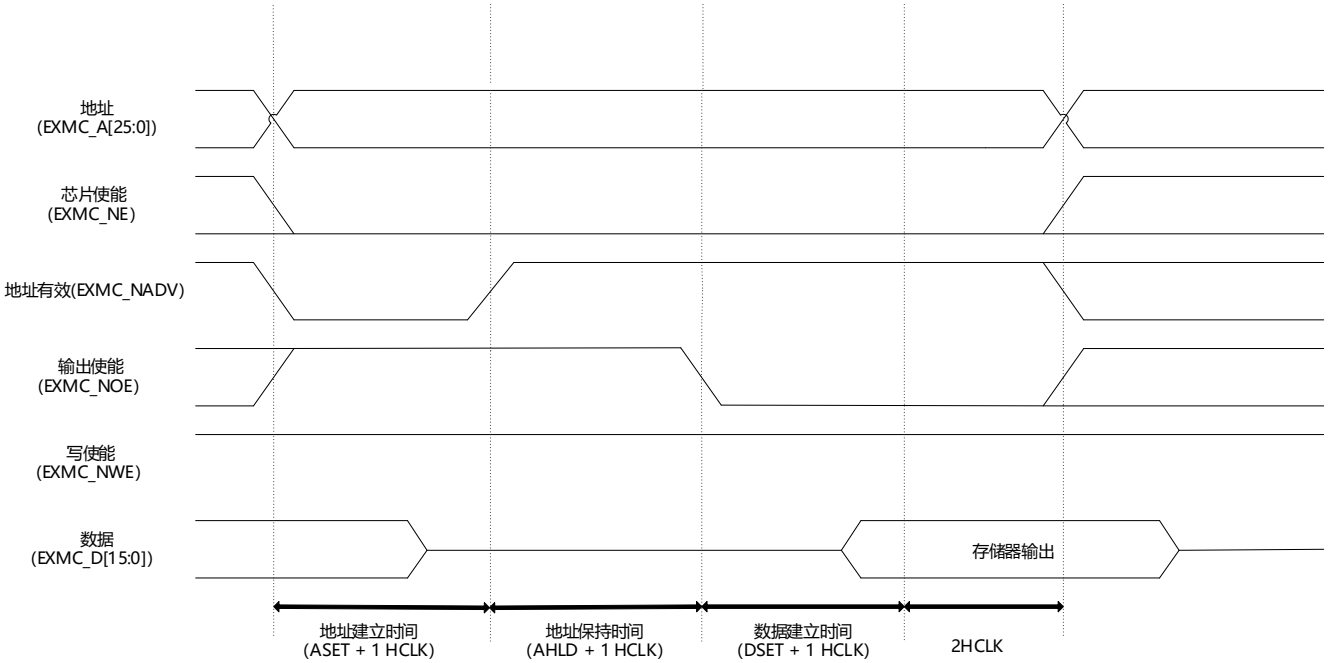


图 27-13 模式 D 写访问

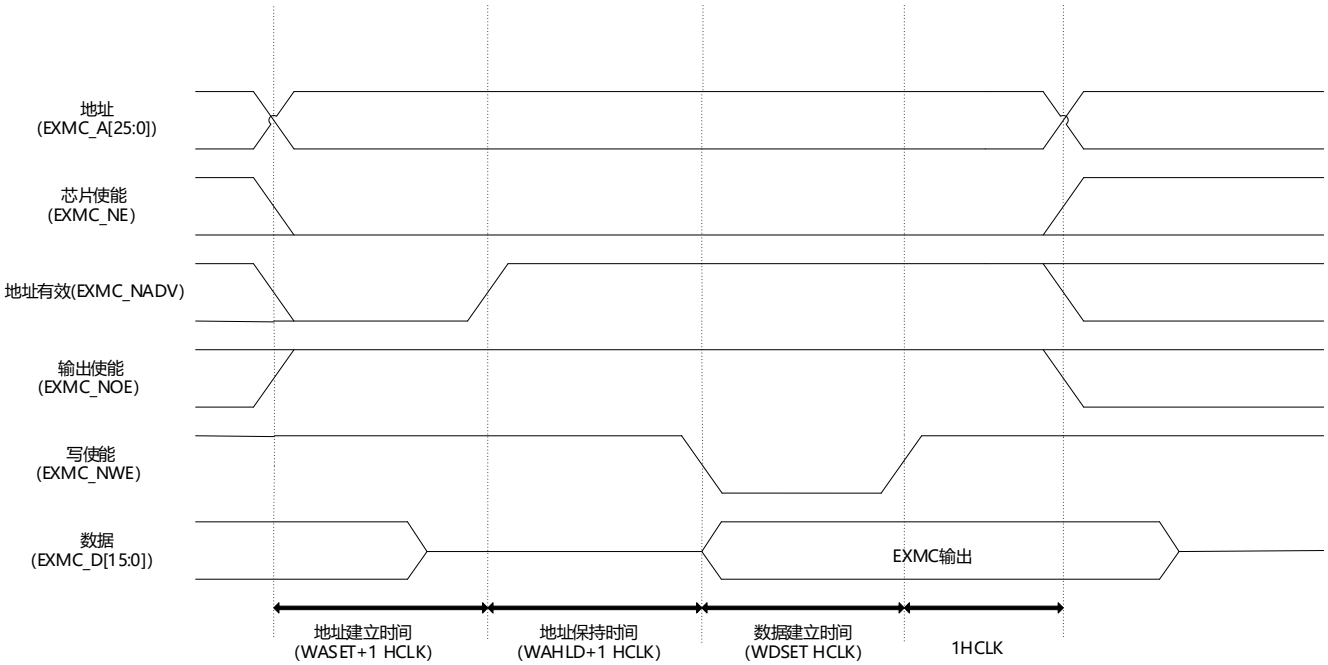


表 27-13 模式 D 相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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 EXMC_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
EXMC_SNWTCFG		
31:30	RSV	0x0
29:28	WASYNCMOD	模式 D：0x3
27:20	RSV	0xFF
19:16	WBUSLAT	从 EXMC_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

模式 AM - NOR Flash 地址/数据总线复用：

图 27-14 复用模式读访问

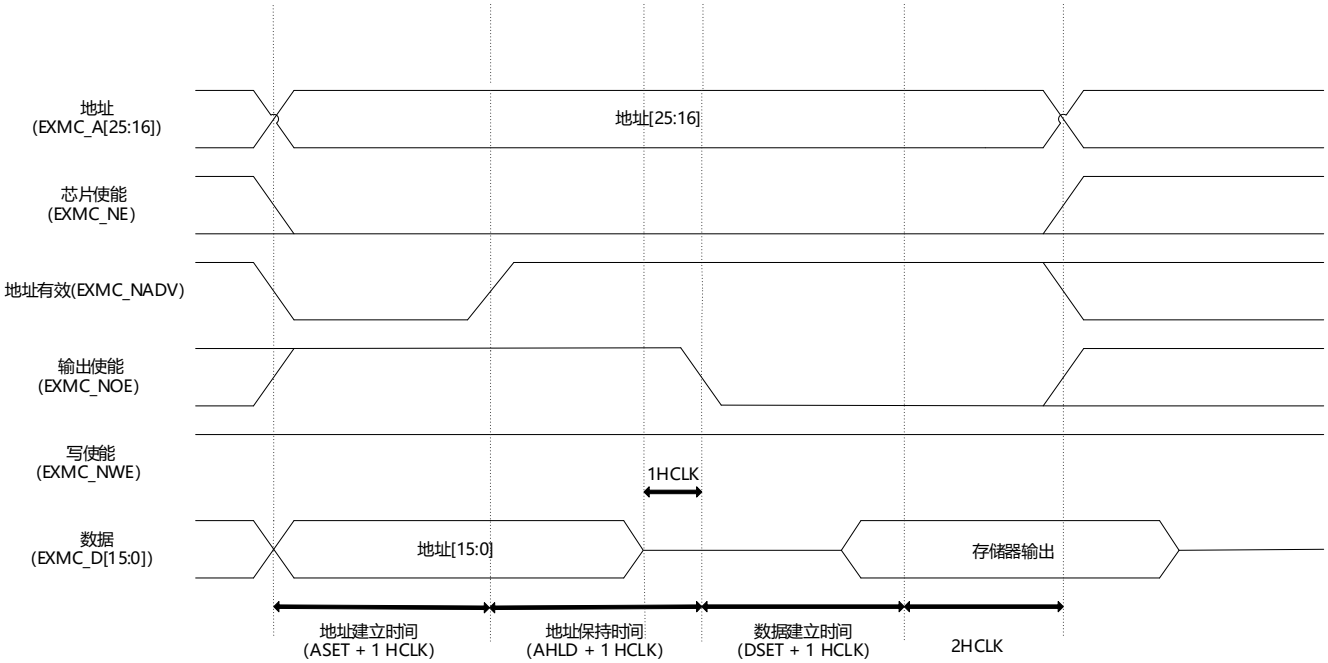


图 27-15 复用模式写访问

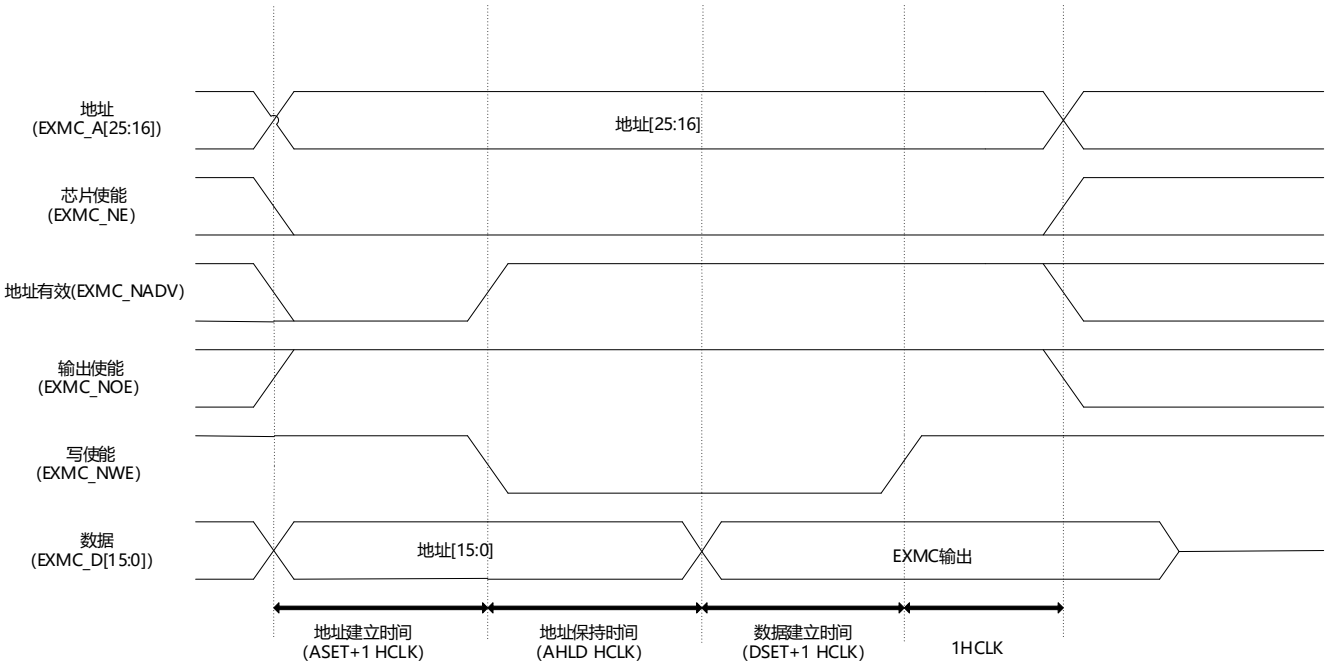


表 27-14 复用模式相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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 EXMC_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 EXMC_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 EXMC_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 EXMC_NOE/ EXMC_NWE 信号对齐：TDATA_SETUP≥ maxTWAIT_ASSERTION+4HCLK

2）若存储器等待信号与 EXMC_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 27-16 异步等待有效时的读时序

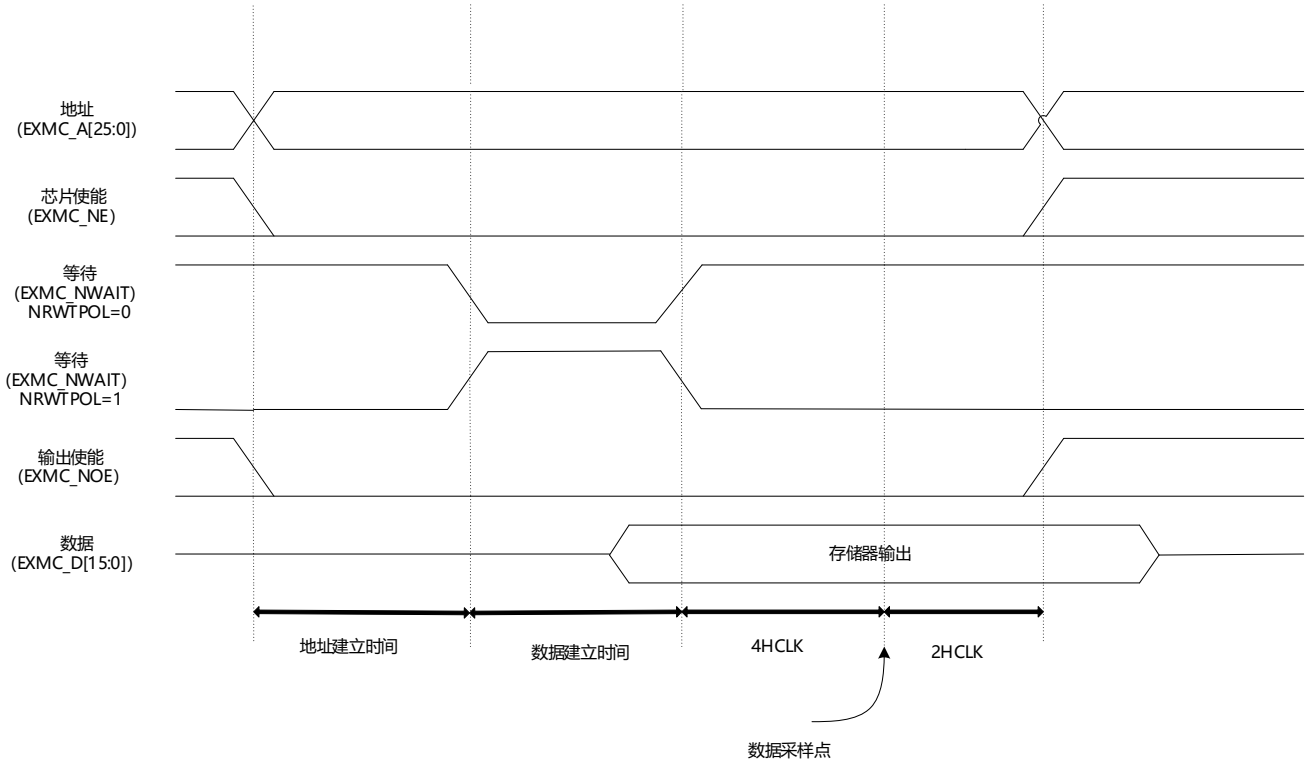
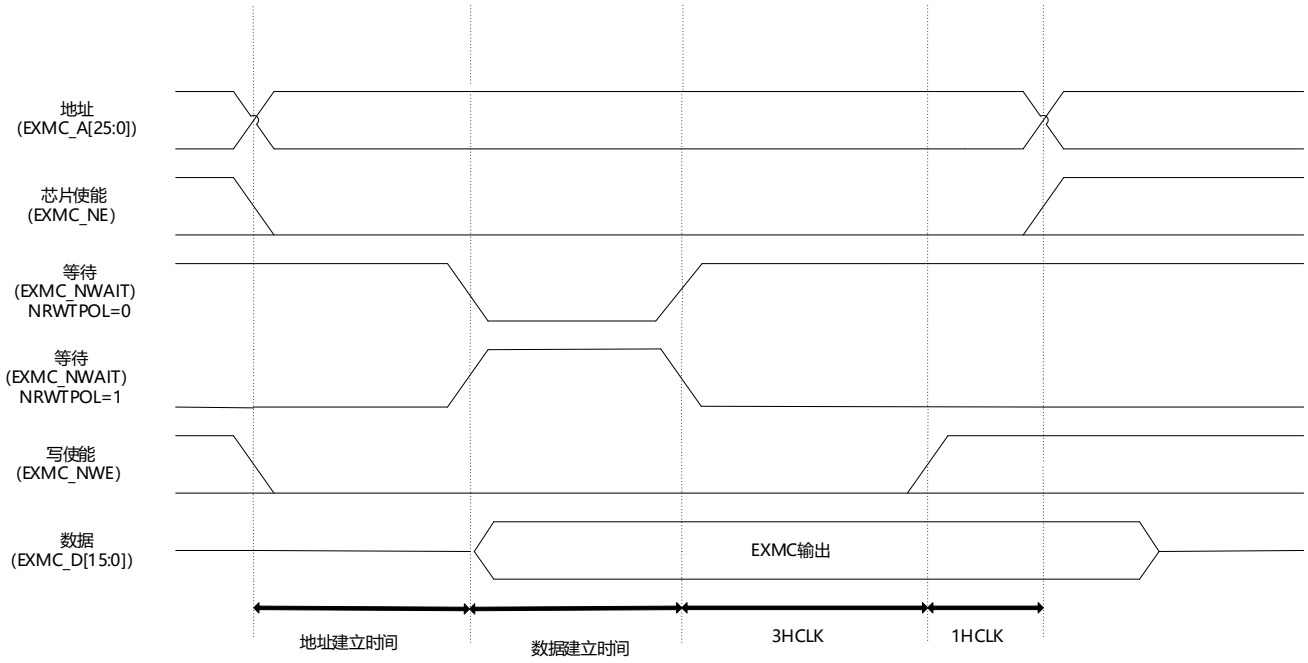


图 27-17 异步等待有效时的写时序



27.3.9. 同步事务

存储器时钟（EXMC_CLK）与系统时钟（HCLK）关系如下：

EXMC_CLK= HCLK / (CKDIV+1)

其中 CKDIV 是同步时钟分频比，通过配置寄存器 EXMC_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 EXMC_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 EXMC_NADV，二者之间的关系为：

NOR 闪存延迟=DLAT+2

- NOR 闪存延迟包含 EXMC_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 EXMC_NWAIT 信号与外部设备一致。该信号通过寄存器 EXMC_SNCTL 来设置，由位 NRWTEN 来使能，位 NRWTCFG 决定 EXMC_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 EXMC_NWAIT 信号极性。

在 NOR Flash 的同步突发模式中， 当寄存器 EXMC_SNCTL 的位 NRWTEN 置为 1， 则在数据延迟之后会检测 EXMC_NWAIT 信号。 如果检测到 EXMC_NWAIT 有效， 就会插入等待时钟， 直到 EXMC_NWAIT 变为无效。

- EXMC_NWAIT 有效极性：
  - NRWTPOL= 1， EXMC_NWAIT 高电平有效；
  - NRWTPOL= 0， EXMC_NWAIT 低电平有效。
- 在同步突发模式中， EXMC_NWAIT 信号有两种配置：
  - NRWTCFG = 1， EXMC_NWAIT 信号有效时， 当前时钟周期数据无效；
  - NRWTCFG = 0， EXMC_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 EXMC_NWAIT 信号有效的等待周期内， EXMC 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， EXMC 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 EXMC_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 EXMC 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 EXMC 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. EXMC 的 Bank0 支持的所有传输。

同步复用突发读时序 – NOR， PSRAM（CRAM）：

图 27-18 同步复用突发传输读时序

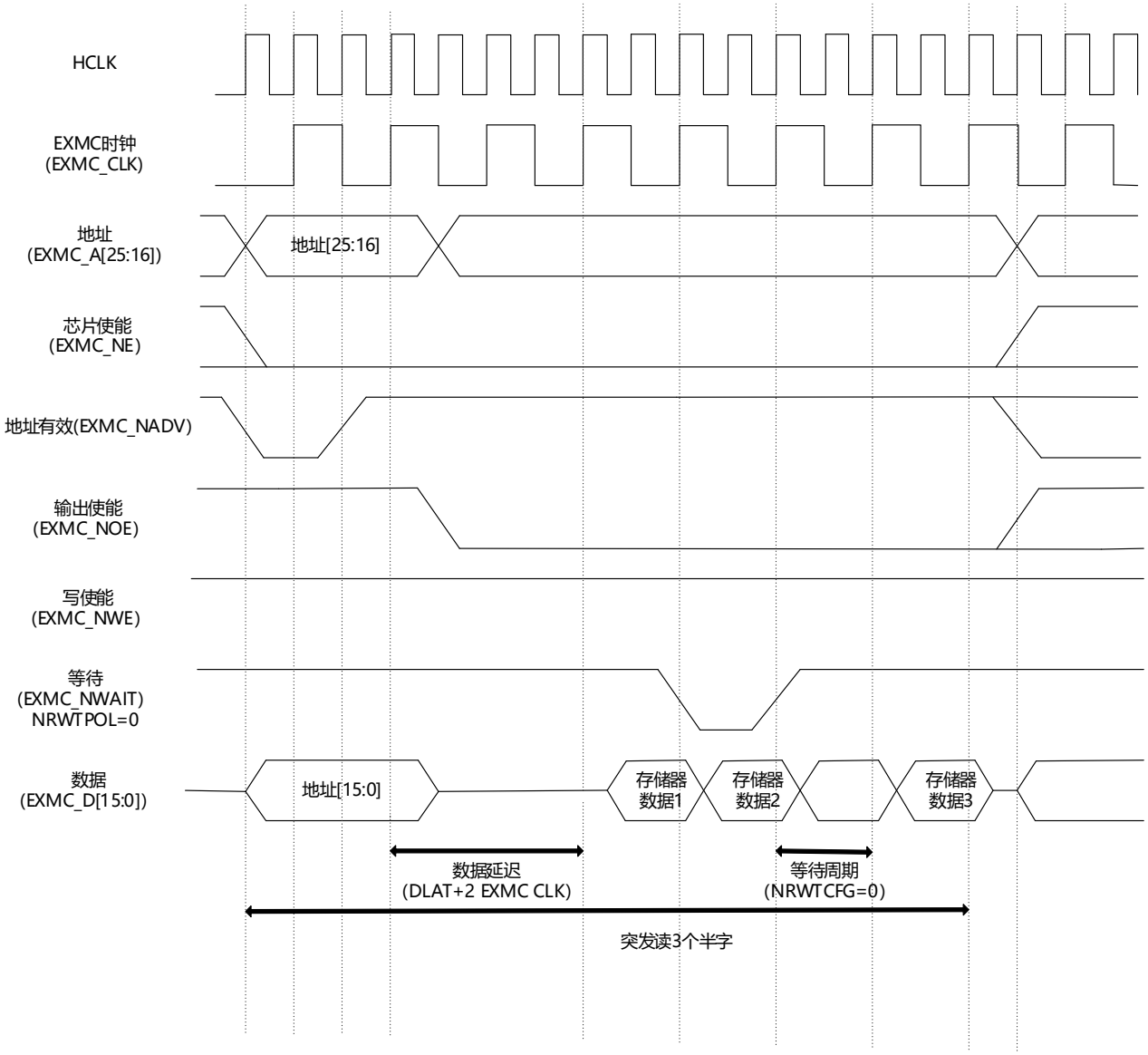


表 27-15 同步复用模式读时序相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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，EXMC_CLK=2HCLk
19:16	BUSLAT	从 EXMC_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 27-19 同步复用突发传输写时序

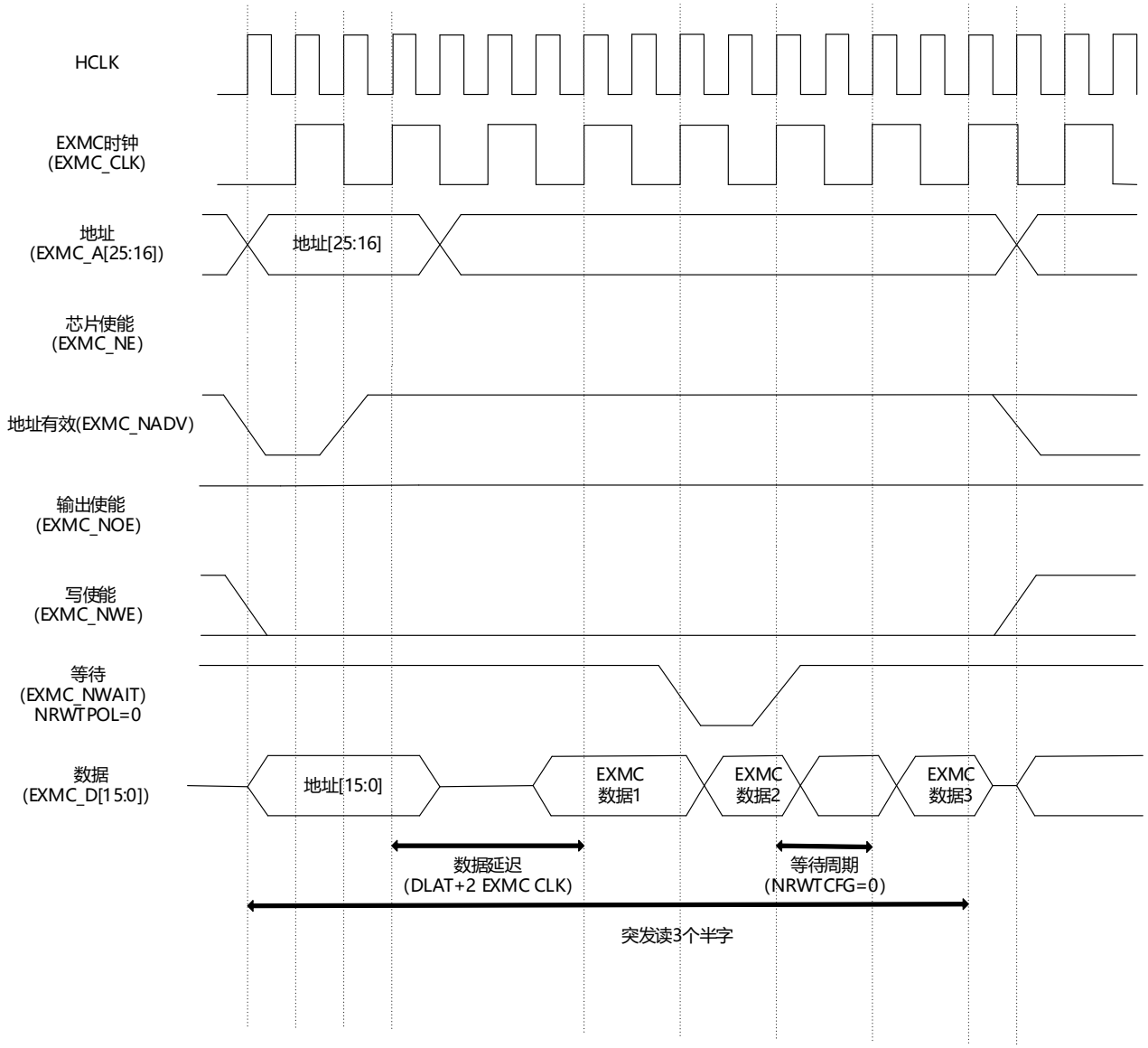


表 27-16 同步复用模式写时序相关寄存器配置

位/位域	位名	参考设定值
EXMC_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
EXMC_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，EXMC_CLK=2HCLk
19:16	BUSLAT	从 EXMC_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

27.4. 配置流程

27.4.1. 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 EXMC 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。

27.4.2. PSRAM 设置流程

1）将 PSRAM 对应 IO 口的外设复用功能配置到 EXMC 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。

27.4.3. SRAM 设置流程

1）将 SRAM 对应 IO 口的外设复用功能配置到 EXMC 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。

27.4.4. LCD8080 设置流程

1）将 LCD8080 对应 IO 口的外设复用功能配置到 EXMC 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。

27.5. EXMC 寄存器描述

27.5.1. 寄存器列表

EXMC 寄存器基地址：0xA0000000

EXMC 内存映射地址：0x60000000~0x6FFFFFFF

偏移	名称	描述
0x00	EXMC_SNCTL0	SRAM/NOR Flash 控制寄存器 0
0x08	EXMC_SNCTL1	SRAM/NOR Flash 控制寄存器 1
0x10	EXMC_SNCTL2	SRAM/NOR Flash 控制寄存器 2
0x18	EXMC_SNCTL3	SRAM/NOR Flash 控制寄存器 3
0x04	EXMC_SNTCFG0	SRAM/NOR Flash 时序配置寄存器 0
0x0C	EXMC_SNTCFG1	SRAM/NOR Flash 时序配置寄存器 1
0x14	EXMC_SNTCFG2	SRAM/NOR Flash 时序配置寄存器 2
0x1C	EXMC_SNTCFG3	SRAM/NOR Flash 时序配置寄存器 3
0x104	EXMC_SNWTCFG0	SRAM/NOR Flash 写时序寄存器 0
0x10C	EXMC_SNWTCFG1	SRAM/NOR Flash 写时序寄存器 1
0x114	EXMC_SNWTCFG2	SRAM/NOR Flash 写时序寄存器 2
0x11C	EXMC_SNWTCFG3	SRAM/NOR Flash 写时序寄存器 3

27.5.2. SRAM/NOR Flash 控制寄存器(EXMC_SNCTLx: 00h/08h/10h/18h)

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[2:0]	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 EXMC 对外部存储器的写操作，否则产生一个 AHB 错误 1： 允许 EXMC 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[1:0]	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[1:0]	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

27.5.3. SRAM/NOR Flash 时序配置寄存器(EXMC_SNTCFGx: 04h/0ch/14h/1ch)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:28	ASYNCMOD[1:0]	RW	00	异步访问模式 该位域仅在当 EXMC_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[3:0]	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 EXMC_CLK 时钟周期 0x1： 第一组突发传输时，数据延迟时间为 3 个 EXMC_CLK 时钟周期 …… 0xF： 第一组突发传输时，数据延迟时间为 17 个 EXMC_CLK 时钟周期
23:20	CKDIV[3:0]	RW	1111	同步模式时钟分频比，仅在同步模式有效 0x0：保留 0x1： EXMC_CLK 周期=2 个 HCLK 周期 …… 0xF： EXMC_CLK 周期=16 个 HCLK 周期
19:16	BUSLAT[3:0]	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[7:0]	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[3:0]	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[3:0]	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

27.5.4. SRAM/NOR Flash 写时序寄存器(EXMC_SNWTCFGx: 104h/10Ch/114h/11Ch)

位域	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:28	WASYNCMOD[1:0]	RW	00	异步访问模式 该位域仅在当 EXMC_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[3:0]	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 EXMC_CLK 时钟周期 0x1： 第一组突发传输时，数据延迟时间为 3 个 EXMC_CLK 时钟周期 …… 0xF： 第一组突发传输时，数据延迟时间为 17 个 EXMC_CLK 时钟周期
27:20	RSV	RW	1111	保留
19:16	WBUSLAT[3:0]	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[7:0]	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[3:0]	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[3:0]	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

28. 通用串行总线（USB）

28.1. 概述

USB 设备控制器是一个兼容 USB2.0 全速协议设备接口，其与 USB PHY 配合使用，提供芯片与 USB HOST 通讯的功能。仅支持从设备功能，不支持主机功能。

USB 外设实现了 USB2.0 全速总线和 AHB 总线间的接口。

USB 外设支持 USB 挂起/恢复操作，支持 Stop 模式下的 Resume 唤醒。

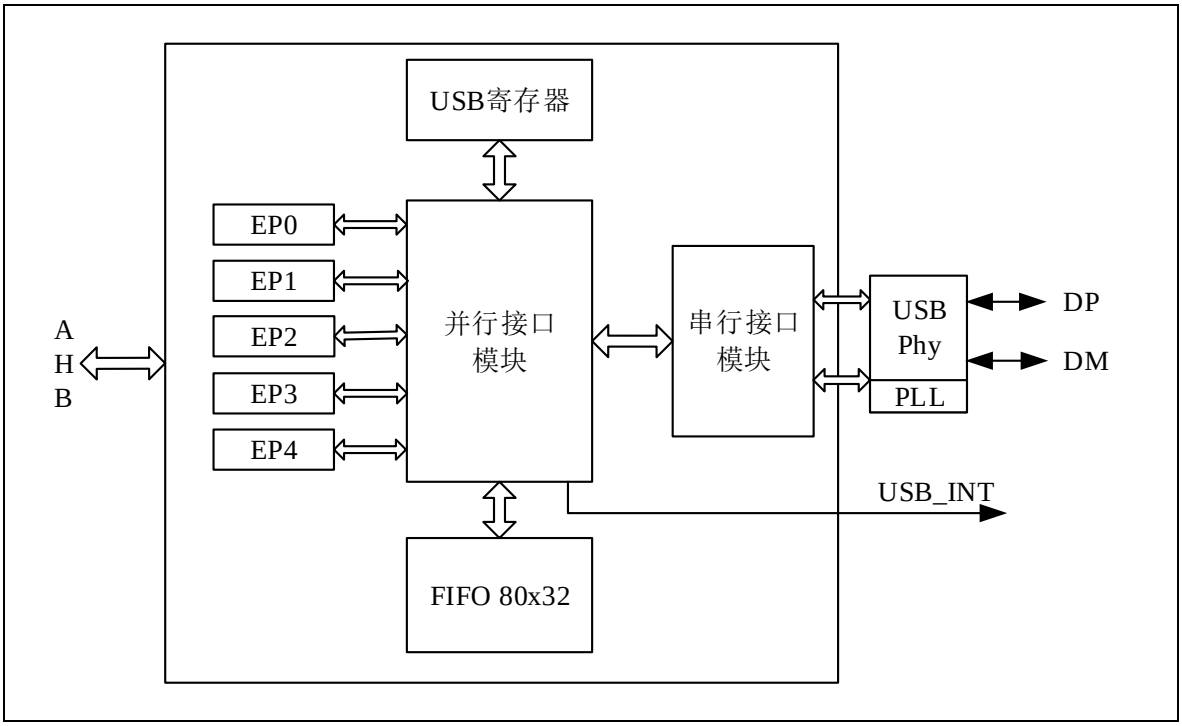
28.2. 主要特性

- 兼容 USB1.1 和 USB2.0 全速协议
- USBPHY 内的 48M 时钟既支持免晶振方式，也支持接外部晶振方式
- 含有 5 个通用双向传输 End Point（EP0、EP1、EP2、EP3、EP4）
- End Point 支持最大包长度 64Byte，支持 Memory 和 USB 两种访问功能
- FIFO 模式支持 32bit 方式访问
- Memory 访问支持 8、16、32bit 三种访问方式
- 支持控制传输、中断传输、批量传输，不支持同步传输
- 支持挂起、唤醒和远程唤醒功能
- 支持 Toggle 硬件比对与软件控制功能
- 支持每一个 End Point 数据传输产生中断功能
- 支持可选的 CRC 错误回复 NAK 功能
- 支持数据包超过最大包长度(64byte)自动回复 NAK 功能
- 支持 IN 操作主机未回 ACK，接下来 IN 操作 USB 设备回复 NAK 功能
- 支持令牌包与数据包 EOP 丢失检测，支持可选的丢失 EOP 自动回复 NAK 功能
- 支持 Stop 模式下 Resume 信号唤醒 MCU 功能

28.3. 功能描述

28.3.1. 结构框图

图 28-1 USB 结构框架图



28.3.2. 全速 PHY

芯片内嵌全速 USBPHY，支持 USB1.1 协议，支持 12Mpbs 全速工作模式。PHY 支持外部晶振和免晶振两种时钟模式，自动识别外部晶振。未识别到外部晶振自动进入免晶振时钟模式。

DP 管脚支持两种上拉电阻：2.1K 欧姆和 2.8K 欧姆，可通过软件独立使能两个上拉电阻。根据 USB 协议，在 USB 空闲期间把两个上拉电阻都使能；在 USB 通讯期间只使能 2.1K 电阻。

28.3.3. USB 状态

■ 初始态

复位之后，USB 处于初始状态，DP 和 DM 没有上拉电阻，通过主机端的 15KΩ 下拉电阻下拉为低电平。

■ 软连接

软件通过配置 WORKING_MODE 的 USB_DPPU_LO 和 USB_DPPU 位，使能 DP 的上拉电阻使 DP 处于高电平进入软连接状态。

■ 默认态

主机检查到 DP 拉高后会发出 Bus_Reset 命令，主机会拉低 DP 和 DM 持续 10ms。软件通过 USB_STATUS_DETECT_CNT（探测时间寄存器）设置检查时间。检查到后会设状态寄存器 INT_STAT_RAW 的 BUS_RESET 为高。此时，USB 控制器的设备地址为 0。

■ 配置态

主机通过 Set Configuration 命令选择 USB 配置。从机在收到此命令后要保证该配置中用到的端点已使能。

■ 挂起态（Suspend）

对于暂时不操作的从机，主机会发起 SUSPEND 操作，保持 DP 为高 DM 为低持续 3ms 以上。软件通过 USB_STATUS_DETECT_CNT（探测时间寄存器）设置检查时间。检查到 SUSPEND 命令后会设状态寄存器 INT_STAT_RAW 的 SUSPEND_RAW 为高。系统检查到 SUSPEND 后，如果没有需要操作的任务，芯片可以进入低功耗状态，具体参见低功耗模式章节。

28.3.4. EndPoint

USB 控制器包含 5 个通用双向传输 End Point（EP0、EP1、EP2、EP3、EP4），除了 EP0 之外的所有 End Point 支持中断传输、批量传输，只有 EP0 支持控制传输。每个 End Point 支持最大包长度 64Byte，支持 Memory 和 FIFO 两种访问功能。

28.3.5. FIFO 访问与 Memory 访问

此 USB 控制器支持 FIFO 访问与 Memory 访问两种方式。FIFO 只支持 32bit 访问，而 Memory 访问支持 8、16、32bit 访问。

■ FIFO 访问

FIFO 访问通过 EPxFIFO 访问入口寄存器进行，只能进行 32bit 的操作，需要字对齐。每个 End Point 只有一个 FIFO 访问入口地址。对于发送，先写入的数据先发送；对于接收，先接收的数据先读出。

■ Memory 访问

Memory 访问通过 EPxMEM Memory 访问入口进行，支持 8、16、32 位访问。每个 End Point 有 64Byte 的 memory 存储区域，对应的访问地址也为 64。对于发送，低位地址存放先发送数据；对于接收，低位地址存放先接收数据。

28.3.6. 地址设置 Set Address

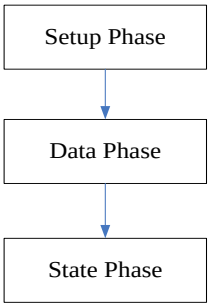
控制传输中 Set Address 命令可以全部由硬件完成，也可以由软件完成。软件如果需要知道 Set Address 命令已经发生，则可以使用中断位 SETADDR。

28.3.7. SETUP 数据和 EP0 控制传输数据：

控制传输中 set address 命令全部由硬件完成，软件如果需要知道 set address 命令已经发生，则可以通过对 SETADDR_EN 位置 1 来使能 SETADDR 中断。

USB 每次控制传输都要经过，SETUP phase，DATA phase(可选)，STATUS phase。

图 28-2 控制传输过程



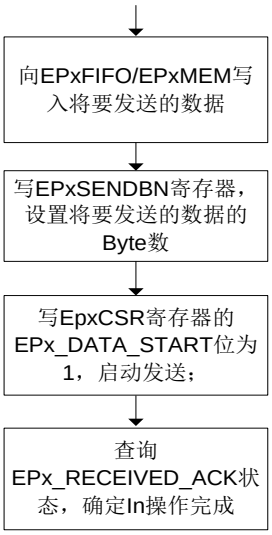
STATUS phase 由长度为 0 的数据包来完成。

- Setup phase：当控制传输 SETUP 数据接收完毕，USB 会产生 SUDAV_RAW 中断，这时 SETUP 数据会在寄存器 SETUP_0_3_DATA 和 SETUP_4_7_DATA；
- Data phase：当数据 Data Phase 方向为 OUT，那么当数据传输完成后 EP0_OUT_RAW 将会置 1，接收到的数据保存在 EP0 的 FIFO 中，而数据发生的状态将保存在 EP0CSR 中。当数据 Data Phase 方向为 IN，那么当 Host 发来 EP0 的 In Token，这时 EP0IN_RAW 中断将被触发（如果中断已使能），同时会将 EP0 FIFO 中的数据按照 EP0CSR 中的指示发送给 Host。如果数据没有准备好，那么 USB 控制器将自动回复 NAK packet；
- Status phase：Status Phase 的 Status 可能由 Host 发给 Device，也可能由 Device 发回 Host，取决于 Data Phase 的方向。这需要软件维护这一状态。如果需要由 Device 发给 Host，用户需要采用 Data Phase 的方法，将 0 长度数据准备好。USB 控制器会在收到 EP0 In Token 后将数据发给 Host。

28.3.8. Endpoint In 传输

当 Host 发来 EPx 的 IN token，这时 EPxIN_INT 中断将被触发（如果中断已使能），同时会将 EPx FIFO 中的数据按照 EPxCSR 中的指示发送会 Host，包括发送的数据长度等。Endpoint In 传输的流程图如下图所示：

图 28-3 Endpoint In 传输

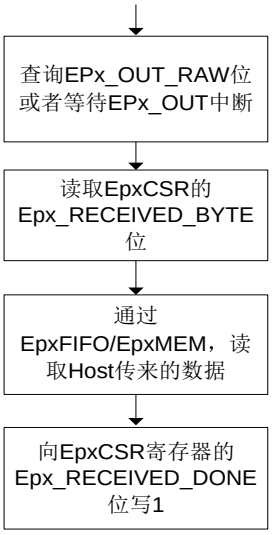


28.3.9. Endpoint Out 传输

当 EPxOUT_RAW 置 1 时，EPxCSR 寄存器的 EPx_RECEIVED_BYTE，字段记录了数据接收的长度，用户可以根据这一长度，从 EPxFIFO 或者 EPxMEM 中读取数据。当 EPxOUT 的数据读取完成之后，用户需要写一下 EPxCSR 寄存器的 EPx_RECEVIED_DONE 位，表示可以接收下一笔数据。Endpoint Out 传输的流程图如下图

所示：

图 28-4 Endpoint Out 传输



28.3.10. 中断处理

USB 设备对应每一个中断都有中断状态、中断使能，分别是：

- INT_ISR：全局中断状态寄存器，无论中断有无使能都会显示中断发生的状态，写 1 可以清除状态
- INT_EN：全局中断使能寄存器，使能某个全局中断的发生
- EPINT_ISR、EPINT_ISR2：端点中断状态寄存器，无论中断有无使能都会显示中断发生的状态，写 1 可以清除状态
- EPINT_EN、EPINT_EN2：端点中断使能寄存器，使能某个端点中断的发生

28.3.11. 远程唤醒

远程唤醒功能，由 WORKING_MODE 寄存器中的 USB_REMOTE_WAKEUP 位来实现。向 USB_REMOTE_WAKEUP 写 1，将会在 USB 端口上发出一个 K 状态，其持续的时间由软件控制。软件向 USB_REMOTE_WAKEUP 写 0，USB 端口停止发送 K 状态。

在发起远程唤醒前，需要确认当前 USB Phy 是否为使能状态，Phy 产生的 CLK48M 时钟是否开启。如果没有 CLK48M 时钟，控制器将无法发起远程唤醒操作。USB PHY 的相应配置由 SYSCFG 模块的 PHYCFG 决定。

28.3.12. 错误处理

对于通讯中发生的异常错误，USB 控制器支持以下几种错误和处理机制。

28.3.12.1. 令牌包与数据包 CRC 出错选择性回复 NAK

令牌包和数据包发生 CRC 错误，控制器可以选择性的回复 NAK，通过 WORKING_ MODE 寄存器中的 USB_CRCERR_NAK_En 来控制。当 USB_CRCERR_NAK_En 为 1 时，USB 设备在 CRC 出错时回复 NAK 握手包，否则将不回复握手包。

28.3.12.2. 数据包长度超过 64 Byte

当 USB 控制器的端点接收到的数据包长度超过 64 Byte 时，状态寄存器 INT_STAT_RAW 中的 DATA_BYTE_MORETHAN_64 位将会置 1。如果 INT_EN 寄存器中的 DATA_BYTE_ MORETHAN_64_EN 位为 1，将会将会产生中断。

28.3.12.3. 包丢失 EOP

当 WORKING_MODE 寄存器中的 USB_TOKEN_NOEOP_En 位与 USB_DATA_NOEOP _En 位为 1 时，令牌包或数据包在限定的包长度内未收到结束标志 EOP，状态寄存器 INT_STAT_RAW 中的 NOEOP_ERR_RAW 位将会置 1。如果 INT_EN 寄存器中的 NOEOP_ERR_RAW_EN 位为 1，将会将会产生中断。控制器对令牌包按长度 32 个 Full Speed 比特位来检测，当令牌包在超过 32 个 Full Speed 比特位时未收到 EOP，则会触发此错误，设备会复位内部状态机，从而不影响下一个包的接收。数据包按有效数据 64byte+8byte 检测，当数据包内有效数据长度超过此值时，也会产生此错误。

令牌包和数据包在发生此类错误时，可以选择是否向 USB 主机回复 NAK 握手包。分别通过设置 WORKING_MODE 寄存器中的 USB_TOKEN_NOEOP_NAK_En 位和 USB_DATA_NOEOP_NAK_En 位来设置。当 USB_TOKEN_NOEOP_NAK_En 位为 1 时，发生令牌包此类错误时，在令牌包的第 32+11 个 Full Speed 比特位后，USB 控制器将回复 NAK 握手包。当 USB_DATA_NOEOP_NAK_En 位为 1 时，发生数据包此类错误时，在数据包的第 64byte+8byte+11bit 个 Full Speed 比特位后，USB 控制器将回复 NAK 握手包。

28.3.12.4. IN 操作 ACK 超时下次 IN 操作回复 NAK

当主机发起 IN 操作时，如果在握手时主机需回复的 ACK 握手包超时，则根据不同的端点，状态寄存器 INT_STAT_RAW 中的 EPx_IN_HANDSHAKE_ERR_RAW 将会被置 1。当 EPx_IN_HANDSHAKE_ERR_RAW 为 1 且主机再次发起针对此端点的 IN 操作时，USB 控制器将会回复 NAK 握手包，直到软件清除 EPx_IN_HANDSHAKE_ERR_RAW 位。

28.3.13. 低功耗模式

USB 控制器属于主区电源域，在 Sleep 模式下能够正常工作，在其它低功耗模式不能工作。在 STOP0 和 STOP1 两种低功耗模式，主区未断电，DP 信号可以设置为唤醒源从这两种低功耗模式唤醒。在 STOP2、STANDBY 和 POWER DOWN 低功耗模式，主区断电，DP 不能作为唤醒源。低功耗对 USB 的影响如下表

表 28-1 USB 在不同休眠模式下的工作状态

模式	说明
SLEEP	无影响，USB 控制器正常工作 USB 中断可使芯片退出睡眠模式
STOP0/1	USB 控制器无法工作，USB 寄存器内容保持， DP 信号可使芯片退出低功耗模式
STOP2	USB 控制器掉电，无法工作或者 DP 唤醒。
STANDBY	USB 控制器掉电，无法工作或者 DP 唤醒。
POWER DOWN	USB 控制器掉电，无法工作或者 DP 唤醒。

在 USB 应用中，检测到主机 SUSPEND 命令后，需要使芯片进入低功耗模式，并使用 USB DP 下降沿唤醒。

■ 具体 STOP0 和 STOP1 进入低功耗模式流程如下：

1）配置 DP 的下降沿唤醒，将 EXTI 线 23 配置为中断模式使能，选择上升沿唤醒。

2）如果需要增加其它唤醒源，按照需求配置使能。

3）设置 PMU_CTRL0 的 LPMS 位选择 STOP0 或者 STOP1 模式；

4）设置 USB 的 WORKING_MODE 寄存器的 USB_SUSPEND 为 1，使 USBPHY 进入 SUSPEND

5）配置 SYSCFG_PHYCFG 寄存器的 PD_PLL 位为 1，关闭 USBPHY 的 PLL

6）设置 Core 的 SLEEPDEEP 位并执行 WFI 或者 WFE。

■ 低功耗退出后，需要恢复 USBPHY 的工作：

1）配置 SYSCFG_PHYCFG 寄存器的 PD_PLL 位为 0，打开 USBPHY 的 PLL

2）设置 USB 的 WORKING_MODE 寄存器的 USB_SUSPEND 为 0，使 USBPHY 进入工作状态

3）查询 EXTI 的 PDR[23]，如果为高代表主机通过 USB Resume 命令唤醒，清除 PDR[23],等待主机发新的 USB 命令。

4）查询 EXTI 的 PDR[23]，如果为低代表非 USB 唤醒，如果 USB 需要和主机通讯，从机需要发送远程唤醒命令唤醒主机。向 USB_REMOTE_WAKEUP 写 1，将会在 USB 端口上发出一个 K 状态，其持续的时间由软件控制。软件向 USB_REMOTE_WAKEUP 写 0，USB 端口停止发送 K 状态

28.4. USB 寄存器描述

28.4.1. 寄存器列表

USB 寄存器基地址：0x40050000

偏置	名称	描述
0x00	USB_WORKING_MODE	工作模式寄存器
0x04	USB_EP0CSR	EP0 传输控制寄存器
0x08	USB_EP1CSR	EP1 传输控制寄存器
0x0C	USB_EP2CSR	EP2 传输控制寄存器
0x10	USB_EP3CSR	EP3 传输控制寄存器
0x14	USB_EP4CSR	EP4 传输控制寄存器
0x18	USB_ADDR	USB 地址寄存器
0x1C	USB_SETUP_0_3_DATA	SETUP 数据包寄存器 0
0x20	USB_SETUP_4_7_DATA	SETUP 数据包寄存器 1
0x24	USB_EP_ADDR	End Point 地址配置寄存器
0x28	USB_CURRENT_PID	当前 USB 总线包 PID 寄存器
0x2C	USB_CURRENT_FRAME_NUMBER	Frame Number 寄存器
0x30	USB_CRC_ERROR_CNT	CRC 错误 Counter 寄存器
0x34	USB_STATUS_DETECT_CNT	Suspend/Resume/Reset 探测时间寄存器
0x40	USB_EP0SENDBN	EP0 发送数据数目寄存器
0x44	USB_EP1SENDBN	EP1 发送数据数目寄存器
0x48	USB_EP2SENDBN	EP2 发送数据数目寄存器

0x4C	USB_EP3SENDBN	EP3 发送数据数目寄存器
0x50	USB_EP4SENDBN	EP4 发送数据数目寄存器
0x100	USB_EP0FIFO	EP0 FIFO 访问入口
0x104	USB_EP1FIFO	EP1 FIFO 访问入口
0x108	USB_EP2FIFO	EP2 FIFO 访问入口
0x10C	USB_EP3FIFO	EP3 FIFO 访问入口
0x110	USB_EP4FIFO	EP4 FIFO 访问入口
0x200~0x23C	USB_EP0MEM	EP0 Memory 访问入口
0x240~0x27C	USB_EP1MEM	EP1 Memory 访问入口
0x280~0x2BC	USB_EP2MEM	EP2 Memory 访问入口
0x2c0~0x2FC	USB_EP3MEM	EP3 Memory 访问入口
0x300~0x33C	USB_EP4MEM	EP4 Memory 访问入口
0xFFE4	USB_INT_STAT_RAW	状态寄存器
0xFFE8	USB_INT_EN	中断使能寄存器
0xFFF0	USB_INT_CLR	中断清除寄存器

28.4.2. 工作模式寄存器(USB_WORKING_MODE: 00h)

位域	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	USB_EP0_ZOD_INTR_EN	RW	0x0	USB OUT 操作中的 0 长度数据包是否产生中断选择 1：使能此中断 0：不使能此中断
24	USB_DATA_NOEOP_EN	RW	0x0	USB 控制器在收到的数据包长度超过 64+8Byte 时，会认为此包数据丢失了 EOP，控制器内部会复位状态机，并产生错误状态 1：使能此功能 0：不使能此功能
23	USB_TOKEN_NOEOP_EN	RW	0x0	USB 控制器在收到的令牌包长度超过规定长度时，会认为此令牌包丢失了 EOP，控制器内部会复位状态机，并产生错误状态 1：使能此功能 0：不使能此功能
22	USB_DATA_NOEOP_NAK_EN	RW	0x0	USB 控制器在收到的数据包长度超过 64+8Byte 时，会认为此包数据丢失了 EOP。此位为 1 时，控制器会回复 NAK 握手包给主机 1：回复 NAK 0：不回复 NAK 此功能只有在 USB_DATA_ NOEOP_En 为 1 时有效

21	USB_TOKEN_NOEOP_NAK_EN	RW	0x0	USB 控制器在收到的令牌包长度超过规定长度时，会认为此令牌包丢失了 EOP。此位为 1 时，控制器会回复 NAK 握手包给主机 1：回复 NAK 0：不回复 NAK 此功能只有在 USB_TOKEN_ NOEOP_En 为 1 时有效
20	USB_REMOTE_WAKEUP	RW	0x0	USB 远程唤醒控制位 1：控制器发送 K 状态 0：控制器不发送 K 状态 唤醒结束后，此位需要软件写 0 清除
19:13	RSV	-	-	保留
12	USB_MORETHAN64_NAK_EN	RW	0x0	收到的数据包长度超过 64Byte，回复 NAK 握手包使能位 1：使能此功能 0：不使能此功能
11	USB_IN_TIMEOUT_NAK_EN	RW	0x0	控制器在 IN 操作数据包发送完毕后，未收到主机的 ACK 握手包，下次该端点的 IN 操作设备将回复 NAK，直到软件清除错误状态位 1：使能此功能 0：不使能此功能
10	USB_CRCERR_NAK_EN	RW	0x0	令牌包和数据包 CRC 错误回复 NAK 功能使能信号 1：使能此功能 0：不使能此功能
9:8	USB_LINE_STATE	R/-	0x0	USB Dp/Dm 信号状态位 8bit：DM 9bit：DP
7	RSV	-	-	保留
6	USB_DPPU_LO	RW	0x0	USB 控制器 Dp 信号 2.8k 上拉电阻控制位 1：开启上拉 0：关闭上拉
5	RSV	-	-	保留
4	USB_DPPU	RW	0x0	USB 控制器 Dp 信号 2.1k 上拉电阻控制位 1：开启上拉 0：关闭上拉
3	USB_BUS_AUTO_RST_EN	RW	0x1	USB 总线复位可以复位控制器地址、收发状态机、收发 FIFO 使能位 1：使能 0：不使能
2	USB_FORCE_RST	RW	0x0	复位控制器地址、收发状态机、收发 FIFO 1：复位 0：不复位 每次复位操作后，需软件写 0，清除该位
1	USB_SUSPEND	RW	0x0	1：设置 USBPhy 为挂起状态 0：清除 USBPhy 的挂起状态

0	SPEED_MODE	RW	0x1	USB 工作模式选择位 1：全速模式 0：低速模式 此芯片仅支持全速模式

28.4.3. EP0 传输控制寄存器(USB_EP0CSR: 04h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
27	EP0_SEND_HALT	WO	0x0	EP0 发送数据停止 1：写 1 停止发送。下一周期自动清零 0：写 0 无效 EP0_SEND_HALT 写 1 后，如果需要再次启动，需要写 EP0_DATA_START
26	EP0_RECEIVED_STALL	RO	0x0	EP0 接收到 Host 的 STALL 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP0_DATA_START 写 1 可以清除该位
25	EP0_RECEIVED_NAK	RO	0x0	EP0 接收到 Host 的 NAK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP0_DATA_START 写 1 可以清除该位
24	EP0_RECEIVED_ACK	RO	0x0	EP0 接收到 Host 的 ACK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP0_DATA_START 写 1 可以清除该位
23:21	RSV	-	-	保留
20	EP0_OUT_VALID	RO	0x0	Ep0 Out 有效标志，当有效数据进入 FIFO 时，此位会置 1。写 1 清零
19	EP0_OUT_TOGGLE_STATE	RO	0x0	EP0 Out/Setup 状态错误标志 1：Toggle 收到的与预期不符 0：Toggle 正常
18	EP0_OUT_TOGGLE_CTRL_EN	WO	0x0	EP0_OUT_TOGGLE_WANT 中的值生效使能 1：EP0_OUT_TOGGLE_WANT 值写入生效
17	EP0_OUT_TOGGLE_WANT	RW	0x0	EP0 Out/Setup 数据包 Toggle 的比对值 1：Data1 0：Data0

16	EP0_OUT_TOGGLE_VALUE	RO	0x0	EP0 收到的数据包的 Toggle 值 1：Data1 0：Data0
15	EP0_IN_TOGGLE_CTRL_EN	WO	0x0	EP0_IN_TOGGLE_VALUE 中值生效使能 1：EP0_IN_TOGGLE_VALUE 值写入生效
14	EP0_IN_TOGGLE_VALUE	RW	0x0	IN 操作 Toggle 控制位 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前 IN Toggle 的值
13	EP0_SEND_STALL_DONE	RW	0x0	EP0 发送 STALL 完成标志位 1：STALL 发送完成 0：STALL 未发送完成 写 1 清 0
12	EP0_SEND_STALL	WO	0x0	STALL 发送控制位 1：发送 STALL 0：不发送 STALL 一次写操作只发送一次 STALL
11	EP0_RECEIVED_DONE	WO	0x0	接收完成控制位 1：接收完成 0：接收未完成 向此位写 1，会将 FIFO 置为 Ready 状态，每次接收完数据并且读取完数据后都需要向此位写 1，否则下次 OUT/SETUP 操作，设备将会回 NAK
10	EP0_DATA_START	WO	0x0	发送 START 1：将 FIFO 中的数据发出 0：无操作
9	EP0_FIFOCLR	WO	0x0	EP0 FIFO 指针复位控制位 1：复位 FIFO 指针 0：不复位 FIFO 指针
8	EP0_EN	RW	0x1	EP0 端点使能位 1：使能 0：不使能
7:0	EP0_RECEIVED_BYTE	RO	0x0	EP0 接收到的数据 Byte 数目。此位需要小于最大包长度

28.4.4. EP1 传输控制寄存器(USB_EP1CSR: 08h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留

27	EP1_SEND_HALT	WO	0x0	EP1 发送数据停止 1：写 1 停止发送。下一周期自动清零 0：写 0 无效 EP1_SEND_HALT 写 1 后，如果需要再次启动，需要写 EP1_DATA_START
26	EP1_RECEIVED_STALL	RO	0x0	EP1 接收到 Host 的 STALL 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP1_DATA_START 写 1 可以清除该位
25	EP1_RECEIVED_NAK	RO	0x0	EP1 接收到 Host 的 NAK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP1_DATA_START 写 1 可以清除该位
24	EP1_RECEIVED_ACK	RO	0x0	EP1 接收到 Host 的 ACK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP1_DATA_START 写 1 可以清除该位
23:21	RSV	-	-	保留
20	EP1_OUT_VALID	RO	0x0	Ep1 Out 有效标志，当有效数据进入 FIFO 时，此位会置 1。 写 1 清零
19	EP1_OUT_TOGGLE_STATE	RO	0x0	EP1 Out/Setup 状态错误标志 1：Toggle 收到的与预期不符 0：Toggle 正常
18	EP1_OUT_TOGGLE_CTRL_EN	WO	0x0	EP1_OUT_TOGGLE_WANT 中的值生效使能 1：EP1_OUT_TOGGLE_WANT 值写入生效
17	EP1_OUT_TOGGLE_WANT	RW	0x0	EP1 Out/Setup 数据包 Toggle 的比对值 1：Data1 0：Data0
16	EP1_OUT_TOGGLE_VALUE	RO	0x1	EP1 收到的数据包的 Toggle 值 1：Data1 0：Data0
15	EP1_IN_TOGGLE_CTRL_EN	WO	0x0	EP1_IN_TOGGLE_VALUE 中值生效使能 1：EP1_IN_TOGGLE_VALUE 值写入生效
14	EP1_IN_TOGGLE_VALUE	RW	0x0	IN 操作 Toggle 控制位 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前 IN Toggle 的值

13	EP1_SEND_STALL_DONE	RO	0x0	EP1 发送 STALL 完成标志位 1：STALL 发送完成 0：STALL 未发送完成 写 1 清 0
12	EP1_SEND_STALL	RW	0x0	STALL 发送控制位 1：发送 STALL 0：不发送 STALL 一次写操作只发送多次 STALL，直到清除该位
11	EP1_RECEIVED_DONE	WO	0x0	接收完成控制位 1：接收完成 0：接收未完成 向此位写 1，会将 FIFO 置为 Ready 状态，每次接收完数据并且读取完数据后都需要向此位写 1，否则下次 OUT/SETUP 操作，设备将会回 NAK
10	EP1_DATA_START	WO	0x0	发送 START 1：将 FIFO 中的数据发出 0：无操作
9	EP1_FIFOCLR	WO	0x0	EP1 FIFO 指针复位控制位 1：复位 FIFO 指针 0：不复位 FIFO 指针
8	EP1_EN	RW	0x1	EP1 端点使能位 1：使能 0：不使能
7:0	EP1_RECEIVED_BYTE	RO	0x0	EP1 接收到的数据 Byte 数目。此位需要小于最大包长度

28.4.5. EP2 传输控制寄存器(USB_EP2CSR: 0Ch)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
27	EP2_SEND_HALT	WO	0x0	EP2 发送数据停止 1：写 1 停止发送。下一周期自动清零 0：写 0 无效 EP2_SEND_HALT 写 1 后，如果需要再次启动，需要写 EP2_DATA_START
26	EP2_RECEIVED_STALL	RO	0x0	EP2 接收到 Host 的 STALL 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP2_DATA_START 写 1 可以清除该位

25	EP2_RECEIVED_NAK	RO	0x0	EP2 接收到 Host 的 NAK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP2_DATA_START 写 1 可以清除该位
24	EP2_RECEIVED_ACK	RO	0x0	EP2 接收到 Host 的 ACK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP2_DATA_START 写 1 可以清除该位
23:21	-	-	-	-
20	EP1_OUT_VALID	RO	0x0	Ep1 Out 有效标志，当有效数据进入 FIFO 时，此位会置 1 写 1 清零
19	EP2_OUT_TOGGLE_STATE	RO	0x0	EP2 Out/Setup 状态错误标志 1：Toggle 收到的与预期不符 0：Toggle 正常
18	EP2_OUT_TOGGLE_CTRL_EN	WO	0x0	EP2_OUT_TOGGLE_WANT 中的值生效使能 1：EP2_OUT_TOGGLE_WANT 值写入生效
17	EP2_OUT_TOGGLE_WANT	RW	0x0	EP2 Out/Setup 数据包 Toggle 的比对值 1：Data1 0：Data0
16	EP2_OUT_TOGGLE_VALUE	RO	0x1	EP2 收到的数据包的 Toggle 值 1：Data1 0：Data0
15	EP2_IN_TOGGLE_CTRL_EN	WO	0x0	EP2_IN_TOGGLE_VALUE 中值生效使能 1：EP2_IN_TOGGLE_VALUE 值写入生效
14	EP2_IN_TOGGLE_VALUE	RW	0x0	IN 操作 Toggle 控制位 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前 IN Toggle 的值
13	EP2_SEND_STALL_DONE	RO	0x0	EP2 发送 STALL 完成标志位 1：STALL 发送完成 0：STALL 未发送完成 写 1 清 0
12	EP2_SEND_STALL	RW	0x0	STALL 发送控制位 1：发送 STALL 0：不发送 STALL 一次写操作只发送多次 STALL，直到清除该位

11	EP2_RECEIVED_DONE	WO	0x0	接收完成控制位 1：接收完成 0：接收未完成 向此位写 1，会将 FIFO 置为 Ready 状态，每次接收完数据并且读取完数据后都需要向此位写 1，否则下次 OUT/SETUP 操作，设备将会回 NAK
10	EP2_DATA_START	WO	0x0	发送 START 1：将 FIFO 中的数据发出 0：无操作
9	EP2_FIFOCLR	WO	0x0	EP2 FIFO 指针复位控制位 1：复位 FIFO 指针 0：不复位 FIFO 指针
8	EP2_EN	RW	0x1	EP2 端点使能位 1：使能 0：不使能
7:0	EP2_RECEIVED_BYTE	RO	0x0	EP2 接收到的数据 Byte 数目。此位需要小于最大包长度

28.4.6. EP3 传输控制寄存器(USB_EP3CSR: 10h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
27	EP3_SEND_HALT	WO	0x0	EP3 发送数据停止 1：写 1 停止发送。下一周期自动清零 0：写 0 无效 EP3_SEND_HALT 写 1 后，如果需要再次启动，需要写 EP3_DATA_START
26	EP3_RECEIVED_STALL	RO	0x0	EP3 接收到 Host 的 STALL 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP3_DATA_START 写 1 可以清除该位
25	EP3_RECEIVED_NAK	RO	0x0	EP3 接收到 Host 的 NAK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP3_DATA_START 写 1 可以清除该位
24	EP3_RECEIVED_ACK	RO	0x0	EP3 接收到 Host 的 ACK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP3_DATA_START 写 1 可以清除该位
23:21	-	-	-	-

20	EP3_OUT_VALID	RO	0x0	Ep3 Out 有效标志，当有效数据进入 FIFO 时，此位会置 1 写 1 清零
19	EP3_OUT_TOGGLE_STATE	RO	0x0	EP3 Out/Setup 状态错误标志 1：Toggle 收到的与预期不符 0：Toggle 正常
18	EP3_OUT_TOGGLE_CTRL_EN	WO	0x0	EP3_OUT_TOGGLE_WANT 中的值生效使能 1：EP3_OUT_TOGGLE_WANT 值写入生效
17	EP3_OUT_TOGGLE_WANT	RW	0x0	EP3 Out/Setup 数据包 Toggle 的比对值 1：Data1 0：Data0
16	EP3_OUT_TOGGLE_VALUE	RO	0x1	EP3 收到的数据包的 Toggle 值 1：Data1 0：Data0
15	EP3_IN_TOGGLE_CTRL_EN	WO	0x0	EP3_IN_TOGGLE_VALUE 中值生效使能 1：EP3_IN_TOGGLE_VALUE 值写入生效
14	EP3_IN_TOGGLE_VALUE	RW	0x0	IN 操作 Toggle 控制位 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前 IN Toggle 的值
13	EP3_SEND_STALL_DONE	RO	0x0	EP3 发送 STALL 完成标志位 1：STALL 发送完成 0：STALL 未发送完成 写 1 清 0
12	EP3_SEND_STALL	RW	0x0	STALL 发送控制位 1：发送 STALL 0：不发送 STALL 一次写操作只发送多次 STALL，直到清除该位
11	EP3_RECEIVED_DONE	WO	0x0	接收完成控制位。 1：接收完成 0：接收未完成 向此位写 1，会将 FIFO 置为 Ready 状态，每次接收完数据并且读取完数据后都需要向此位写 1，否则下次 OUT/SETUP 操作，设备将会回 NAK
10	EP3_DATA_START	WO	0x0	发送 START 1：将 FIFO 中的数据发出 0：无操作
9	EP3_FIFOCLR	WO	0x0	EP3 FIFO 指针复位控制位 1：复位 FIFO 指针 0：不复位 FIFO 指针
8	EP3_EN	RW	0x1	EP3 端点使能位 1：使能 0：不使能

7:0	EP3_RECEIVED_BYTE	RO	0x0	EP3 接收到的数据 Byte 数目。此位需要小于最大包长度

28.4.7. EP4 传输控制寄存器(USB_EP4CSR: 14h)

位域	名称	属性	复位值	描述
31:25	RSV	-	-	保留
27	EP4_SEND_HALT	WO	0x0	EP4 发送数据停止 1：写 1 停止发送。下一周期自动清零 0：写 0 无效 EP4_SEND_HALT 写 1 后，如果需要再次启动，需要写 EP4_DATA_START
26	EP4_RECEIVED_STALL	RO	0x0	EP4 接收到 Host 的 STALL 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP4_DATA_START 写 1 可以清除该位
25	EP4_RECEIVED_NAK	RO	0x0	EP4 接收到 Host 的 NAK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP4_DATA_START 写 1 可以清除该位
24	EP4_RECEIVED_ACK	RO	0x0	EP4 接收到 Host 的 ACK 握手包标志信号 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset 或者向 EP4_DATA_START 写 1 可以清除该位
23:21	-	-	-	-
20	EP4_OUT_VALID	RO	0x0	Ep4 Out 有效标志，当有效数据进入 FIFO 时，此位会置 1。写 1 清零
19	EP4_OUT_TOGGLE_STATE	RO	0x0	EP4 Out/Setup 状态错误标志 1：Toggle 收到的与预期不符 0：Toggle 正常
18	EP4_OUT_TOGGLE_CTRL_EN	WO	0x0	EP4_OUT_TOGGLE_WANT 中的值生效使能 1：EP4_OUT_TOGGLE_WANT 值写入生效
17	EP4_OUT_TOGGLE_WANT	RW	0x0	EP4 Out/Setup 数据包 Toggle 的比对值 1：Data1 0：Data0
16	EP4_OUT_TOGGLE_VALUE	RO	0x1	EP4 收到的数据包的 Toggle 值 1：Data1 0：Data0
15	EP4_IN_TOGGLE_CTRL_EN	WO	0x0	EP4_IN_TOGGLE_VALUE 中值生效使能 1：EP4_IN_TOGGLE_VALUE 值写入生效

14	EP4_IN_TOGGLE_VALUE	RW	0x0	IN 操作 Toggle 控制位 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前 IN Toggle 的值
13	EP4_SEND_STALL_DONE	RO	0x0	EP4 发送 STALL 完成标志位 1：STALL 发送完成 0：STALL 未发送完成 写 1 清 0
12	EP4_SEND_STALL	RW	0x0	STALL 发送控制位 1：发送 STALL 0：不发送 STALL 一次写操作只发送多次 STALL，直到清除该位
11	EP4_RECEIVED_DONE	WO	0x0	接收完成控制位 1：接收完成 0：接收未完成 向此位写 1，会将 FIFO 置为 Ready 状态，每次接收完数据并且读取完数据后都需要向此位写 1，否则下次 OUT/SETUP 操作，设备将会回 NAK
10	EP4_DATA_START	WO	0x0	发送 START 1：将 FIFO 中的数据发出 0：无操作
9	EP4_FIFOCLR	WO	0x0	EP4 FIFO 指针复位控制位 1：复位 FIFO 指针 0：不复位 FIFO 指针
8	EP4_EN	RW	0x1	EP4 端点使能位 1：使能 0：不使能
7:0	EP4_RECEIVED_BYTE	RO	0x0	EP4 接收到的数据 Byte 数目。此位需要小于最大包长度

28.4.8. USB 地址寄存器(USB_ADDR: 18h)

位域	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:0	USB_ADDR	RO	0x0	USB 地址寄存器

28.4.9. SETUP 数据包寄存器(USB_SETUP_0_3_DATA: 1Ch)

位域	名称	属性	复位值	描述
31:0	SETUP_0_3_DATA	RO	0x0	SETUP Data 包 Byte0~Byte3 寄存器

28.4.10. SETUP 数据包寄存器(USB_SETUP_4_7_DATA: 20h)

位域	名称	属性	复位值	描述
31:0	SETUP_4_7_DATA	RO	0x0	SETUP Data 包 Byte4~Byte7 寄存器

28.4.11. End Point 地址配置寄存器(USB_EP_ADDR: 24h)

位域	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	EP4_ADDR	RW	0x4	Ep4 地址配置
11:8	EP3_ADDR	RW	0x3	Ep3 地址配置
7:4	EP2_ADDR	RW	0x2	Ep2 地址配置
3:0	EP1_ADDR	RW	0x1	Ep1 地址配置

28.4.12. 总线包 PID 寄存器(USB_CURRENT_PID: 28h)

位域	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3:0	CURRENT_PID	RO	0x0	当前接收的 USB 包的 PID 值

28.4.13. Frame Number 寄存器(USB_CURRENT_FRAME_NUMBER: 2Ch)

位域	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:0	CURRENT_FRAME_NUMBER	RO	0x3f	当前帧序号

28.4.14. CRC 错误 Counter 寄存器(USB_CRC_ERROR_CNT: 30h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CRC_ERROR_CNT	RO	0x0	CRC 错误包的个数，在 USB Reset 时复位

28.4.15. 探测时间寄存器(USB_STATUS_DETECT_CNT: 34h)

位域	名称	属性	复位值	描述
31:9	RSV	-	-	保留

8:0	USB_STATUS_DETECT_CNT	RW	0xff	Reset/Resume/Suspend 检测阈值设定。设置时间为 USB_STATUS_DETECT_CNT*5.3us + 2.5us

28.4.16. EP0 发送数据数目寄存器(USB_EP0SENDBN: 40h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP0_SEND_BYTE	RW	0x0	EP0 发送数据 Byte 数量寄存器

28.4.17. EP1 发送数据数目寄存器(USB_EP1SENDBN: 44h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP1_SEND_BYTE	RW	0x0	EP1 发送数据 Byte 数量寄存器

28.4.18. EP2 发送数据数目寄存器(USB_EP2SENDBN: 48h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP2_SEND_BYTE	RW	0x0	EP2 发送数据 Byte 数量寄存器

28.4.19. EP3 发送数据数目寄存器(USB_EP3SENDBN: 4Ch)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP3_SEND_BYTE	RW	0x0	EP3 发送数据 Byte 数量寄存器

28.4.20. EP4 发送数据数目寄存器(USB_EP4SENDBN: 50h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP4_SEND_BYTE	RW	0x0	EP4 发送数据 Byte 数量寄存器

28.4.21. EP0 FIFO 访问入口(USB_EP0FIFO: 100h)

位域	名称	属性	复位值	描述
31:0	EP0FIFO	RW	0x0	EP0 FIFO 入口地址，只支持 32bit 访问

28.4.22. EP1 FIFO 访问入口(USB_EP1FIFO: 104h)

位域	名称	属性	复位值	描述
31:0	EP1FIFO	RW	0x0	EP1 FIFO 入口地址，只支持 32bit 访问

28.4.23. EP2 FIFO 访问入口(USB_EP2FIFO: 108h)

位域	名称	属性	复位值	描述
31:0	EP2FIFO	RW	0x0	EP2 FIFO 入口地址，只支持 32bit 访问

28.4.24. EP3 FIFO 访问入口(USB_EP3FIFO: 10Ch)

位域	名称	属性	复位值	描述
31:0	EP3FIFO	RW	0x0	EP3 FIFO 入口地址，只支持 32bit 访问

28.4.25. EP4 FIFO 访问入口(USB_EP4FIFO: 110h)

位域	名称	属性	复位值	描述
31:0	EP4FIFO	RW	0x0	EP4 FIFO 入口地址，只支持 32bit 访问

28.4.26. 状态寄存器(USB_INT_STAT_RAW: FFE4h)

位域	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_RAW	RO	0x0	IN/OUT/Setup 操作发生 Toggle 错误时，此位置 1
30	NOEOP_ERR_RAW	RO	0x0	设备接收到的令牌包如果长度超过协议规定值，或者数据包数据超过 64+8byte，此状态位会置 1
29	EP4_IN_HANDSHAKE_ERR_RAW	RO	0x0	EP4 IN 操作，主机未成功返回 ACK 信号时，此位会置 1
28	EP3_IN_HANDSHAKE_ERR_RAW	RO	0x0	EP3 IN 操作，主机未成功返回 ACK 信号时，此位会置 1
27	EP2_IN_HANDSHAKE_ERR_RAW	RO	0x0	EP2 IN 操作，主机未成功返回 ACK 信号时，此位会置 1
26	EP1_IN_HANDSHAKE_ERR_RAW	RO	0x0	EP1 IN 操作，主机未成功返回 ACK 信号时，此位会置 1

25	EP0_IN_HANDSHAKE_ERR_RAW	RO	0x0	EP0 IN 操作，主机未成功返回 ACK 信号时，此位会置 1
24	DATA_BYTE_MORETHAN_64_RAW	RO	0x0	收到的 DATA 数据包长度超过 64byte，此位会置 1
23	CRC_ERR_RAW	RO	0x0	收到令牌包或者数据包的 CRC Error，此位会置 1
22	SETADDR_RAW	RO	0x0	当 Host 设置 USB 设备地址完成，此位会置 1
21	TURNAROUND_ERROR_RAW	RO	0x0	Host 回复 Ack 包发生 TimeOut 中断
20	EP4_ACK_RAW	RO	0x0	Ep4 Ack 状态，发送或者接收 Ack 包，此位会置 1
19	EP4_OUT_RAW	RO	0x0	Ep4 Out 中断，当有效数据进入 FIFO 时，此位会置 1
18	EP4_IN_RAW	RO	0x0	Ep4 接收到 IN 令牌包时，此位会置 1
17	EP3_ACK_RAW	RO	0x0	Ep3 Ack 状态，发送或者接收 Ack 包，此位会置 1
16	EP3_OUT_RAW	RO	0x0	Ep3 Out 中断，当有效数据进入 FIFO 时，此位会置 1
15	EP3_IN_RAW	RO	0x0	Ep3 接收到 IN 令牌包时，此位会置 1
14	EP2_ACK_RAW	RO	0x0	Ep2 Ack 状态，发送或者接收 Ack 包，此位会置 1
13	EP2_OUT_RAW	RO	0x0	Ep2 Out 中断，当有效数据进入 FIFO 时，此位会置 1
12	EP2_IN_RAW	RO	0x0	Ep2 接收到 IN 令牌包时，此位会置 1
11	EP1_ACK_RAW	RO	0x0	Ep1 Ack 状态，发送或者接收 Ack 包，此位会置 1
10	EP1_OUT_RAW	RO	0x0	Ep1 Out 中断，当有效数据进入 FIFO 时，此位会置 1
9	EP1_IN_RAW	RO	0x0	Ep1 接收到 IN 令牌包时，此位会置 1
8	EP0_ACK_RAW	RO	0x0	Ep0 Ack 状态，发送或者接收 Ack 包，此位会置 1
7	EP0_OUT_RAW	RO	0x0	Ep0 Out 中断，当有效数据进入 FIFO 时，此位会置 1
6	EP0_IN_RAW	RO	0x0	Ep0 接收到 IN 令牌包时，此位会置 1
5	SUDAV_RAW	RO	0x0	接收到 Setup 数据包，此位会置 1
4	SETUPTOK_RAW	RO	0x0	接收到 Setup 令牌包，此位会置 1
3	SOF_RAW	RO	0x0	接收到 Sof 包，此位会置 1
2	RESUME_RAW	RO	0x0	Host Resume，此位会置 1
1	SUSPEND_RAW	RO	0x0	Host Suspend，此位会置 1
0	BUS_RESET_RAW	RO	0x0	Host Reset，此位会置 1

28.4.27. 中断使能寄存器(USB_INT_EN: FFE8h)

位域	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_EN	RW	0x0	TOGGLE_STATE_ERR 中断使能
30	NOEOP_ERR_EN	RW	0x0	NOEOP_ERR 中断使能
29	EP4_IN_HANDSHAKE _ERR_EN	RW	0x0	EP4_IN_HANDSHAKE _ERR 中断使能
28	EP3_IN_HANDSHAKE _ERR_EN	RW	0x0	EP3_IN_HANDSHAKE _ERR 中断使能
27	EP2_IN_HANDSHAKE _ERR_EN	RW	0x0	EP2_IN_HANDSHAKE _ERR 中断使能

26	EP1_IN_HANDSHAKE _ERR_EN	RW	0x0	EP1_IN_HANDSHAKE _ERR 中断使能
25	EP0_IN_HANDSHAKE_ERR_EN	RW	0x0	EP0_IN_HANDSHAKE _ERR 中断使能
24	DATA_BYTE_MORETHAN_64_EN	RW	0x0	收到的 DATA 数据包长度超过 64byte 中断使能
23	CRC_ERR_EN	RW	0x0	CRC 错误中断使能
22	SETADDR_EN	RW	0x0	当 Host 设置 USB 设备地址完成中断使能
21	TURNAROUND_ERROR_EN	RW	0x0	Host 回复 Ack 包发生 TimeOut 中断使能
20	EP4_ACK_EN	RW	0x0	Ep4 Ack 状态，发送或者接收 Ack 包中断使能
19	EP4_OUT_EN	RW	0x0	Ep4 Out 中断，当有效数据进入 FIFO 时产生中断使能
18	EP4_IN_EN	RW	0x0	Ep4 接收到 IN 令牌包中断使能
17	EP3_ACK_EN	RW	0x0	Ep3 Ack 状态，发送或者接收 Ack 包中断使能
16	EP3_OUT_EN	RW	0x0	Ep3 Out 中断，当有效数据进入 FIFO 时产生中断使能
15	EP3_IN_EN	RW	0x0	Ep3 接收到 IN 令牌包中断使能
14	EP2_ACK_EN	RW	0x0	Ep2 Ack 状态，发送或者接收 Ack 包中断使能
13	EP2_OUT_EN	RW	0x0	Ep2 Out 中断，当有效数据进入 FIFO 时产生中断使能
12	EP2_IN_EN	RW	0x0	Ep2 接收到 IN 令牌包中断使能
11	EP1_ACK_EN	RW	0x0	Ep1 Ack 状态，发送或者接收 Ack 包中断使能
10	EP1_OUT_EN	RW	0x0	Ep1 Out 中断，当有效数据进入 FIFO 时产生中断使能
9	EP1_IN_EN	RW	0x0	Ep1 接收到 IN 令牌包中断使能
8	EP0_ACK_EN	RW	0x0	Ep0 Ack 状态，发送或者接收 Ack 包中断使能
7	EP0_OUT_EN	RW	0x0	Ep0 Out 中断，当有效数据进入 FIFO 时产生中断使能
6	EP0_IN_EN	RW	0x0	Ep0 接收到 IN 令牌包中断使能
5	SUDAV_EN	RW	0x0	接收到 Setup 数据包中断使能
4	SETUPTOK_EN	RW	0x0	接收到 Setup 令牌包中断使能
3	SOF_EN	RW	0x0	接收到 Sof 包中断使能
2	RESUME_EN	RW	0x0	Host Resume 中断使能
1	SUSPEND_EN	RW	0x0	Host Suspend 中断使能
0	BUS_RESET_EN	RW	0x0	Host Reset 中断使能

28.4.28. 中断清除寄存器(USB_INT_CLR: FFF0h)

位域	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_CLR	RW	0x0	TOGGLE_STATE_ERR _RAW 中断寄存器清除位 1：TOGGLE_STATE_ERR_RAW 清 0 0：TOGGLE_STATE_ERR_RAW 保持不变

30	NOEOP_ERR_CLR	RW	0x0	NOEOP_ERR_RAW 中断寄存器清除位 1：NOEOP_ERR_RAW 清 0 0：NOEOP_ERR_RAW 保持不变
29	EP4_IN_HANDSHAKE _ERR_CLR	RW	0x0	EP4_IN_HANDSHAKE_ERR_RAW 中断寄存器清除位 1：EP4_IN_HANDSHAKE_ERR_RAW 清 0 0：EP4_IN_HANDSHAKE_ERR_RAW 保持不变
28	EP3_IN_HANDSHAKE _ERR_CLR	RW	0x0	EP3_IN_HANDSHAKE_ERR_RAW 中断寄存器清除位 1：EP3_IN_HANDSHAKE_ERR_RAW 清 0 0：EP3_IN_HANDSHAKE_ERR_RAW 保持不变
27	EP2_IN_HANDSHAKE _ERR_CLR	RW	0x0	EP2_IN_HANDSHAKE_ERR_RAW 中断寄存器清除位 1：EP2_IN_HANDSHAKE_ERR_RAW 清 0 0：EP2_IN_HANDSHAKE_ERR_RAW 保持不变
26	EP1_IN_HANDSHAKE _ERR_CLR	RW	0x0	EP1_IN_HANDSHAKE_ERR_RAW 中断寄存器清除位 1：EP1_IN_HANDSHAKE_ERR_RAW 清 0 0：EP1_IN_HANDSHAKE_ERR_RAW 保持不变
25	EP0_IN_HANDSHAKE_ERR_CLR	RW	0x0	EP0_IN_HANDSHAKE_ERR_RAW 中断寄存器清除位 1：EP0_IN_HANDSHAKE_ERR_RAW 清 0 0：EP0_IN_HANDSHAKE_ERR_RAW 保持不变
24	DATA_BYTE_MORETHAN_64_CLR	RW	0x0	DATA_BYTE_MORETHAN_64_RAW 中断寄存器清除位 1：DATA_BYTE_MORETHAN_64_RAW 清 0 0：DATA_BYTE_MORETHAN_64_RAW 保持不变
23	CRC_ERR_CLR	RW	0x0	CRC_ERR_RAW 中断寄存器清除位 1：CRC_ERR_RAW 清 0 0：CRC_ERR_RAW 保持不变
22	SETADDR_CLR	RW	0x0	SETADDR_RAW 中断寄存器清除位 1：SETADDR_RAW 清 0 0：SETADDR_RAW 保持不变
21	TURNAROUND_ERROR_CLR	RW	0x0	TURNAROUND_ERROR_RAW 中断寄存器清除位 1：TURNAROUND_ERROR_RAW 清 0 0：TURNAROUND_ERROR_RAW 保持不变
20	EP4_ACK_CLR	RW	0x0	EP4_ACK_RAW 中断寄存器清除位 1：EP4_ACK_RAW 清 0 0：EP4_ACK_RAW 保持不变
19	EP4_OUT_CLR	RW	0x0	EP4_OUT_RAW 中断寄存器清除位 1：EP4_OUT_RAW 清 0 0：EP4_OUT_RAW 保持不变
18	EP4_IN_CLR	RW	0x0	EP4_IN_RAW 中断寄存器清除位 1：EP4_IN_RAW 清 0 0：EP4_IN_RAW 保持不变
17	EP3_ACK_CLR	RW	0x0	EP3_ACK_RAW 中断寄存器清除位 1：EP3_ACK_RAW 清 0 0：EP3_ACK_RAW 保持不变

16	EP3_OUT_CLR	RW	0x0	EP3_OUT_RAW 中断寄存器清除位 1：EP3_OUT_RAW 清 0 0：EP3_OUT_RAW 保持不变
15	EP3_IN_CLR	RW	0x0	EP3_IN_RAW 中断寄存器清除位 1：EP3_IN_RAW 清 0 0：EP3_IN_RAW 保持不变
14	EP2_ACK_CLR	RW	0x0	EP2_ACK_RAW 中断寄存器清除位 1：EP2_ACK_RAW 清 0 0：EP2_ACK_RAW 保持不变
13	EP2_OUT_CLR	RW	0x0	EP2_OUT_RAW 中断寄存器清除位 1：EP2_OUT_RAW 清 0 0：EP2_OUT_RAW 保持不变
12	EP2_IN_CLR	RW	0x0	EP2_IN_RAW 中断寄存器清除位 1：EP2_IN_RAW 清 0 0：EP2_IN_RAW 保持不变
11	EP1_ACK_CLR	RW	0x0	EP1_ACK_RAW 中断寄存器清除位 1：EP1_ACK_RAW 清 0 0：EP1_ACK_RAW 保持不变
10	EP1_OUT_CLR	RW	0x0	EP1_OUT_RAW 中断寄存器清除位 1：EP1_OUT_RAW 清 0 0：EP1_OUT_RAW 保持不变
9	EP1_IN_CLR	RW	0x0	EP1_IN_RAW 中断寄存器清除位 1：EP1_IN_RAW 清 0 0：EP1_IN_RAW 保持不变
8	EP0_ACK_CLR	RW	0x0	EP0_ACK_RAW 中断寄存器清除位 1：EP0_ACK_RAW 清 0 0：EP0_ACK_RAW 保持不变
7	EP0_OUT_CLR	RW	0x0	EP0_OUT_RAW 中断寄存器清除位 1：EP0_OUT_RAW 清 0 0：EP0_OUT_RAW 保持不变
6	EP0_IN_CLR	RW	0x0	EP0_IN_RAW 中断寄存器清除位 1：EP0_IN_RAW 清 0 0：EP0_IN_RAW 保持不变
5	SUDAV_CLR	RW	0x0	接收到 Setup 数据包中断寄存器清除位 1：SUDAV_RAW 清 0 0：SUDAV_RAW 保持不变
4	SETUPTOK_CLR	RW	0x0	接收到 Setup 令牌包中断寄存器清除位 1：SETUPTOK_RAW 清 0 0：SETUPTOK_RAW 保持不变
3	SOF_CLR	RW	0x0	接收到 Sof 包中断寄存器清除位 1：SOF_RAW 清 0 0：SOF_RAW 保持不变

2	RESUME_CLR	RW	0x0	Host Resume 中断寄存器清除位 1：RESUME_RAW 清 0 0：RESUME_RAW 保持不变
1	SUSPEND_CLR	RW	0x0	Host Suspend 中断寄存器清除位 1：SUSPEND_RAW 清 0 0：SUSPEND_RAW 保持不变
0	BUS_RESET_CLR	RW	0x0	Host Reset 中断寄存器清除位 1：BUS_RESET_RAW 清 0 0：BUS_RESET_RAW 保持不变

29. 红外输出（IR）

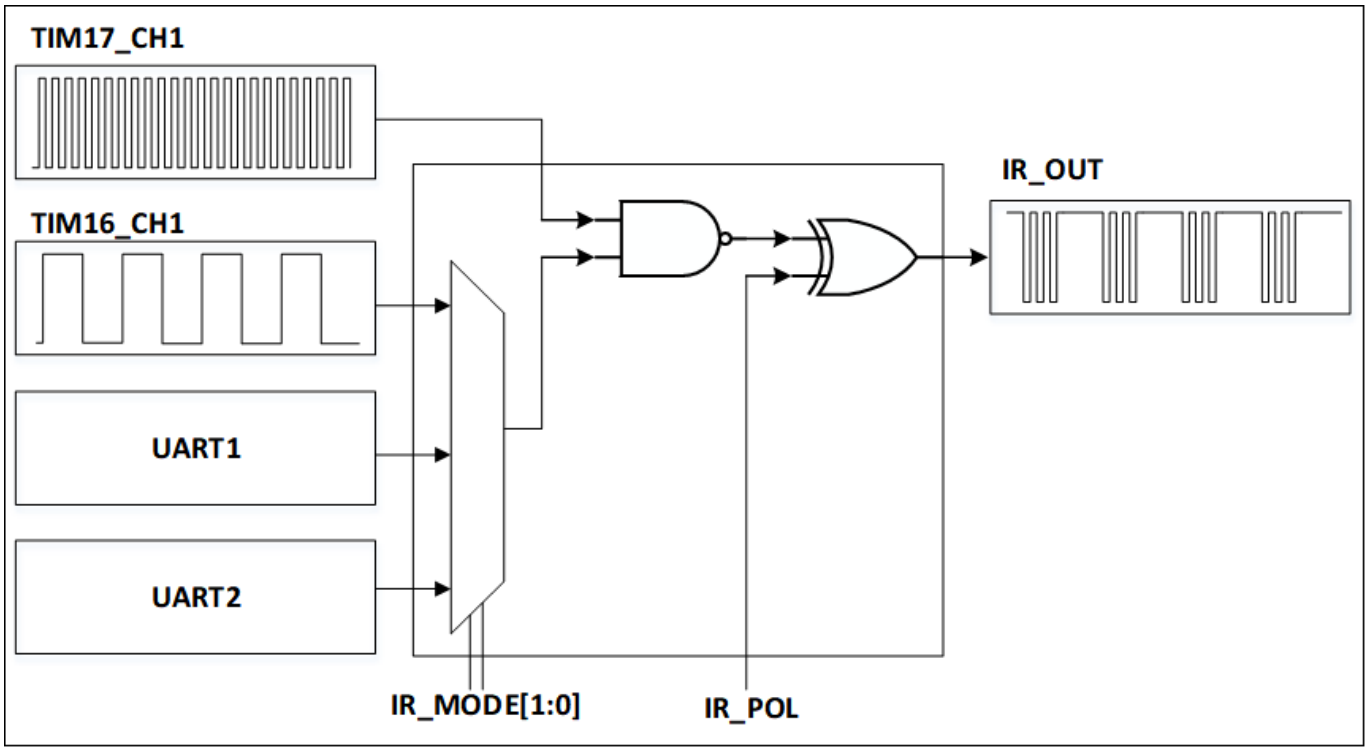
29.1. 概述

本模块实现红外信号的发送。可以实现 UART 信号与 PWM 信号组合输出。

29.2. 主要特性

- 支持 UART 与 PWM 信号组合输出
- 支持两个 PWM 信号组合输出
- 输出极性和配置
- 可在多个引脚同时输出。

29.3. 结构框图



-

29.4. 配置流程

1）设置信号源：系统控制寄存器 SYSCFG_SYSCR.IR_MODE 字段，选择参与组合的信号源

2）设置输出极性：系统控制寄存器 SYSCFG_SYSCR.IR_POL 字段，设置输出极性

3）设置 GPIO 复用模式为 IR_OUT

30. 模数转换器（ADC）

30.1. 概述

12 位 ADC 是一种逐次逼近型模拟数字转换器。本部分包含 2 个 12 位 ADC（ADC1 和 ADC2），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）。

ADC1 支持 20 个通道（IN0-IN19），其中 16 个为外部通道（IN0-IN15），IN16 通道功能可选，可以选择温度传感器（内部通道）或者 VBAT（外部通道），IN17-IN19 三个通道为内部通道。 ADC2 支持 20 个通道，其中 18 个为外部通道，IN16-IN17 两个通道为内部通道。

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。各通道的 A/D 转换可以单次、连续或间断模式执行。ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阈值。

ADC 映射到 AHB 总线，从而可实现快速数据处理。ADC 输入时钟（ADC_CLK）不得超过 48MHz，它是由 HCLK 经分频产生。

内置硬件过采样器，能减轻 CPU 进行相关计算的负担。

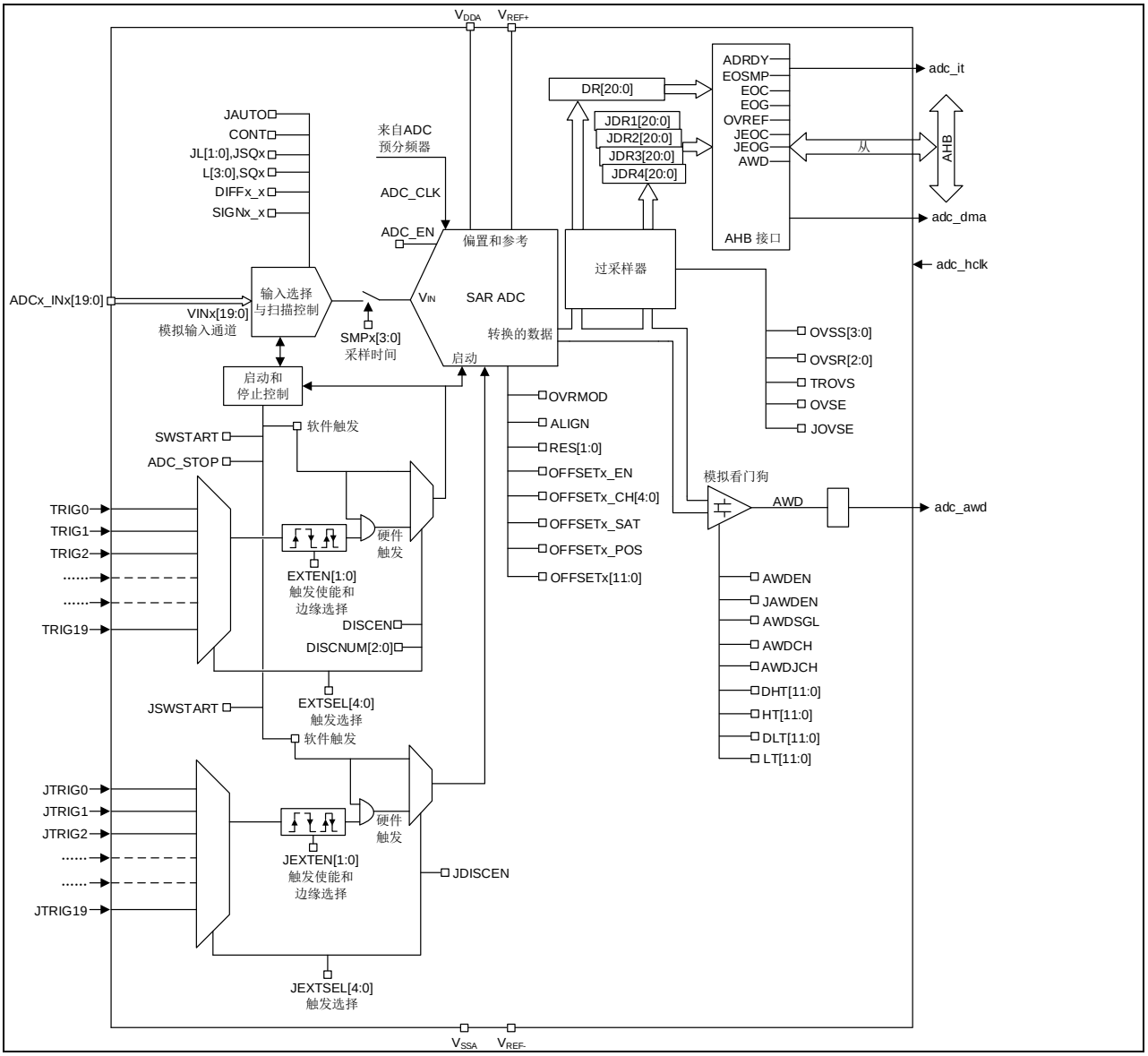
30.2. 主要特性

- 12 位分辨率，也可配置成 10 位、8 位或 6 位分辨率
- 规则转换结束、注入转换结束、发生模拟看门狗事件、规则转换发生溢出时和规则组采样结束时产生中断
- 支持单次、连续转换模式
- 转换速率最高可达 3Msps
- 20 个可用通道，包括外部信号源和内部信号源
  - 内建 BGR 连接到 ADC1
  - 温度传感器连接到 ADC1
  - VBAT 连接到 ADC1
  - The OPA1/2 内部输出连接到 ADC1
  - The OPA2/3 内部输出连接到 ADC2
- 支持单端信号转换和差分信号转换
- 规则转换和注入转换均有外部触发选项
- 支持间断转换模式
- 数据对齐以保持内置数据一致性
- 采样间隔可以按通道分别编程
- 支持双重模式（两个 ADC 设备）
- 双 ADC 支持交替触发模式
- 最多支持 16 个规则通道组和 4 个注入通道组
- AHB 总线便于系统集成，同时实现高速的读写操作
- 支持过采样：16 位数据寄存器，过采样率支持 2~256 倍
- ADC 供电要求：1.8V~3.6V

- ADC 输入范围：VREF- ≤ VIN ≤ VREF+
- 规则通道转换期间有 DMA 请求产生

30.3. 结构框图

图 30-1 ADC 框图



30.4. 引脚和内部信号

表 30-1 ADC 输入/输出引脚

引脚名称	信号类型	说明
VREF+	输入，模拟参考正极	ADC 使用的正极参考电压 1.8V≤VREF+≤VDDA
VDDA	输入，模拟电源	模拟电源等于 VDDA 1.8V≤VDDA≤3.6V

VREF-	输入，模拟参考负极	ADC 使用的负极参考电压 VREF-=VSSA
VSSA	输入，模拟电源地	等效于 VSS 模拟电源地
ADCX_IN[19:0]	输入，外部模拟输入	多达 20 个模拟输入通道（x=1 或 2）

表 30-2 ADC 内部输入/输出信号

内部信号名称	信号类型	说明
TRIG[19:0]	输入	共有多达 20 个外部触发输入用于规则通道转换。 这些输入由主 ADC1 和从 ADC2 共享。
JTRIG[19:0]	输入	共有多达 20 个外部触发输入用于注入通道转换。 这些输入由主 ADC1 和从 ADC2 共享。
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 30-3 ADC 内部互连

信号名称	说明
ADC1_IN[16]	信号来源可选，通过 BUF_ADDR 选择 0：内部温度传感器的输出电压 1：外部电池电压的 1/4 分压：VBAT/4
ADC1_IN[17]	VBG1P2，来源于 1.2V 基准电压
ADC1_IN[18]	OPA2_OUT，来源于运算放大器 OPA2 的输出信号
ADC1_IN[19]	OPA1_OUT，来源于运算放大器 OPA1 的输出信号
ADC2_IN[16]	OPA2_OUT，来源于运算放大器 OPA2 的输出信号
ADC2_IN[17]	OPA3_OUT，来源于运算放大器 OPA3 的输出信号

30.5. 功能描述

30.5.1. 通道选择

一共有 20 个 ADC 通道(通道 0-通道 19)，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：通道 3、通道 8、通道 2、通道 2、通道 0、通道 2、通道 2、通道 15。

注：通道 0 对应模拟输入 VIN0（ADC_IN0），通道 15 对应模拟输入 VIN15（ADC_IN15）。

- 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 30-4 ADC 模拟通道选择

	ADC1	ADC2
IN0	PA0	PA0
IN1	PA1	PA1
IN2	PA2/OPA1_OUT	PA2/OPA1_OUT
IN3	PA3	PA3
IN4	PA4/DAC_OUT1	PA4/DAC_OUT1
IN5	PA5/DAC_OUT2	PA5/DAC_OUT2
IN6	PA6/OPA2_OUT	PA6/OPA2_OUT
IN7	PA7	PA7
IN8	PB0	PB0
IN9	PB1/OPA3_OUT	PB1/OPA3_OUT
IN10	PC0	PC0
IN11	PC1	PC1
IN12	PC2	PC2
IN13	PC3	PC3
IN14	PC4	PC4
IN15	PC5	PC5
IN16	TempSensor/VBAT(PB12)	OPA2_OUT_INT
IN17	VBG1P2(ANA_TEST)	OPA3_OUT_INT
IN18	OPA2_OUT_INT	PE7
IN19	OPA1_OUT_INT	PE8

温度传感器、VBG1P2 内部通道和 VBAT 外部通道

温度传感器内部连接到与 VBAT 共用的通道 ADC1_IN16。一次只能选择一个转换（温度传感器或 VBAT）。选择 VBAT 时，电压通过 PB12 输入。内部参考电压 VBGP12 和 ADC1_IN17 相连接。可以按注入或规则通道对这三个通道进行转换。

注意：温度传感器、VBGP12 和 VBAT 只在主 ADC 外设上可用。

30.5.2. 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。

30.5.3. 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。

30.5.4. 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=0,1,2,3,6,7,9,10

第 1 次触发：转换的序列为 0,1,2

第 2 次触发：转换的序列为 3,6,7

第 3 次触发：转换的序列为 9,10

第 4 次触发：转换的序列为 0,1,2

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 0,1,2，而不是在第 3 次触发中转换通道 0。

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。在出现外部触发事件之后，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。

出现外部触发时，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不连续采样模式。

30.5.5. 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。

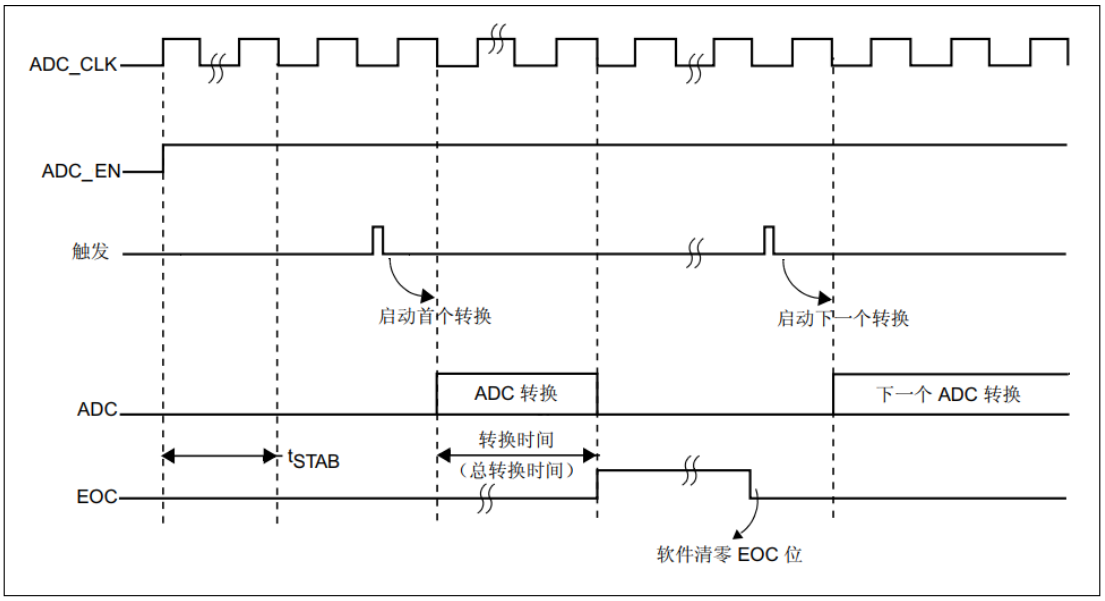
30.5.6. 停止控制

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。

30.5.7. 时序图

如下图所示，ADC 在开始精确转换需要一段稳定时间 tSTAB = 2us，在开始 ADC 转换并经过 16 个周期后，EOC 标志被设置，12 位 ADC 数据寄存器包含转换的结果。

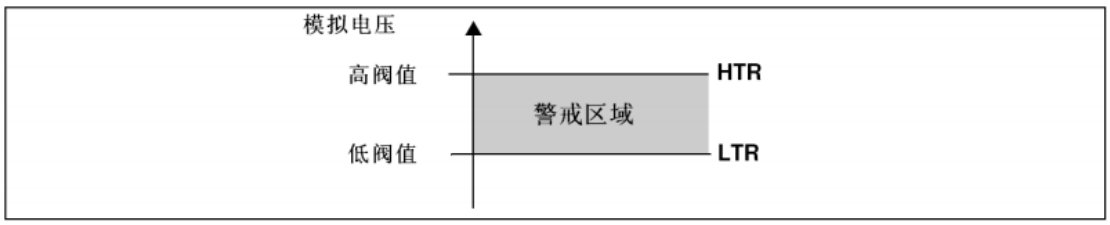
图 30-2 ADC 转换时序图



30.5.8. 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，比较时都当做无符号数。通过设置 ADC_IE 寄存器的 AWDIE 位以允许产生相应中断。

图 30-3 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 30-5 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。

30.5.9. 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 30-4 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 30-5 数据左对齐

30.5.10. 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。

如果 ADC 模拟通道使用有符号形式，则在进行 OFFSET 计算时，仍然认为原始转换结果为无符号数。

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。

30.5.11. 可编程的通道采样时间

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 + 3 - FASTMOD)TADC_CLK

例如：ADC_CLK = 40MHz，FASTMOD =1，RES=0， 采样时间为 3 周期

TCONV= (3+12+3-1) TADC_CLK = 17TADC_CLK =425ns。

注意：在 ADC 转换期间，新发生的触发事件将被忽略，直到转换结束，才启动新的转换序列，EOC 事件还是在转换结束后立即产生。

30.5.12. 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 20 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 30-6 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	TIM1_CC4	TIM16_CC1
10001	LPTIM_OUT	LPTIM_OUT
10010	TIM7_TRGO	TIM7_TRGO
10011	TIM8_CC4	TIM17_CC1

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。

30.5.13. 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，只能才 ADC_CDR 读取结果，DMA 也只能使用主从合并模式。

30.5.14. 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：

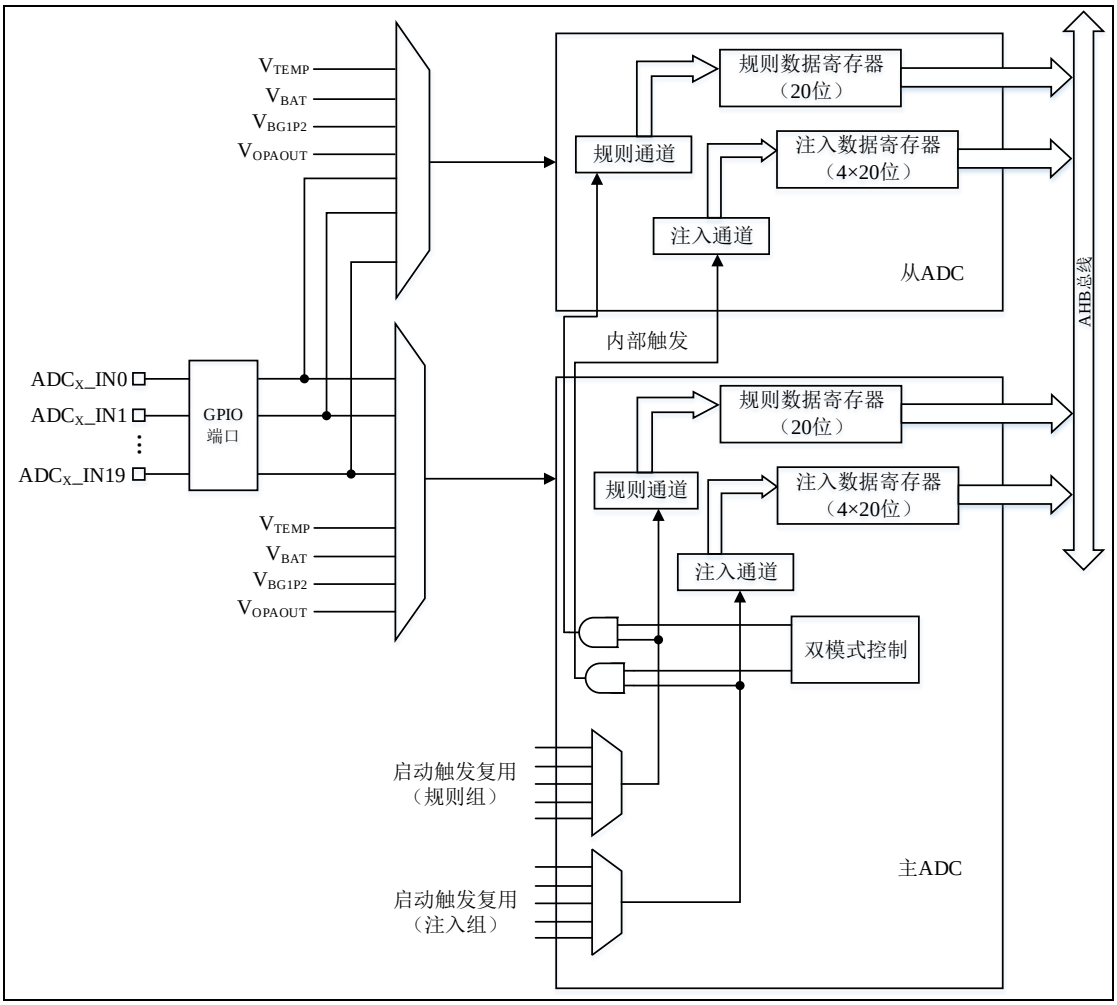
- 注入同步模式
- 规则同步模式
- 规则交叉模式
- 交替触发模式

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 30-6 双 ADC 模式



30.5.14.1. 注入同步模式

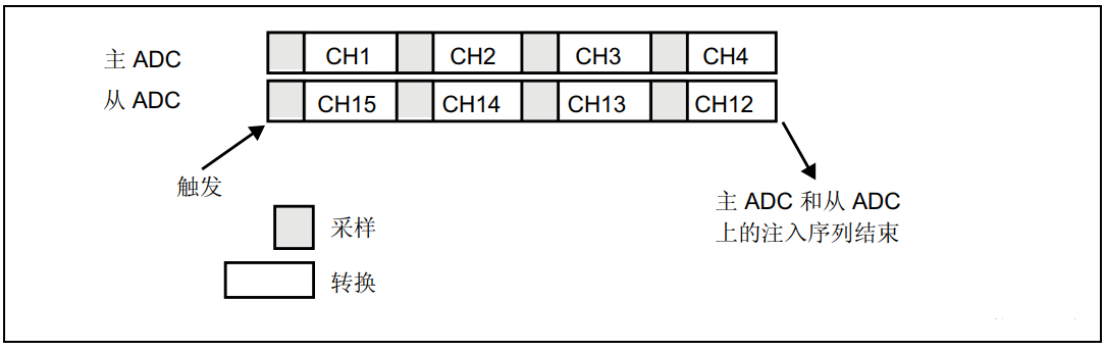
此模式转换一个注入通道组。外部触发来自 ADC1 的注入组多路选择(由 ADC1_JSQR 寄存器的 JEXTSEL[4:0]选择)，它同时给 ADC2 提供同步触发。

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 30-7 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。

30.5.14.2. 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。

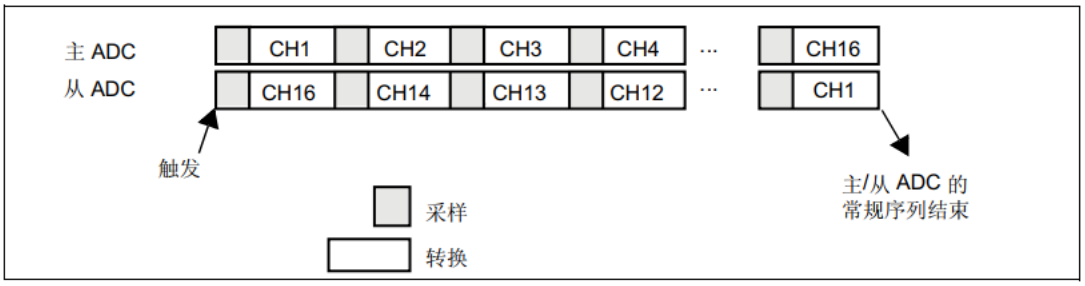
注意：不要在 2 个 ADC 上转换相同的通道(两个 ADC 在同一个通道上的采样时间不能重叠)。

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 30-8 16 个通道的规则同步模式



30.5.14.3. 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

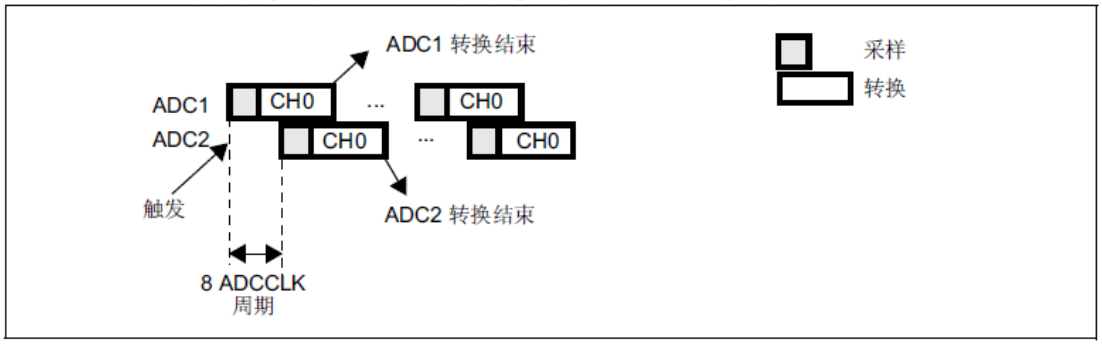
规则交叉模式下 2 个转换之间的最小延迟通过 ADC_CCR 寄存器中的 DELAY 位进行配置。但是，如果某个 ADC 的互补 ADC 仍在对其输入进行采样，则该 ADC 无法启动转换（在给定时间内，只有一个 ADC 能够对输入信号采样）。在这种情况下，延迟时间为采样时间+ 2 个 ADC 时钟周期。例如，如果两个 ADC 的 DELAY = 5 个时钟周期，且采样时间为 15 个时钟周期，则 ADC1 和 ADC2 之间的转换延迟为 17 个时钟周期。

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 30-9 连续转换模式下 1 通道的规则交叉模式



30.5.14.4. 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

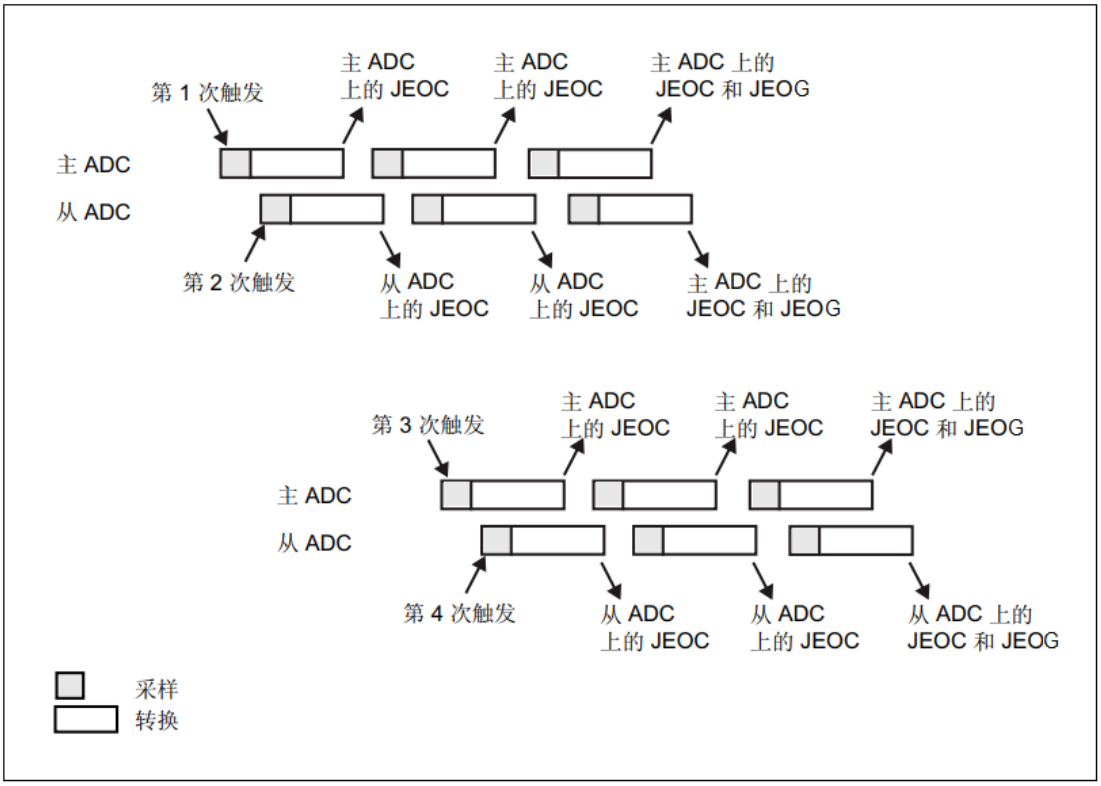
当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

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 30-10 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

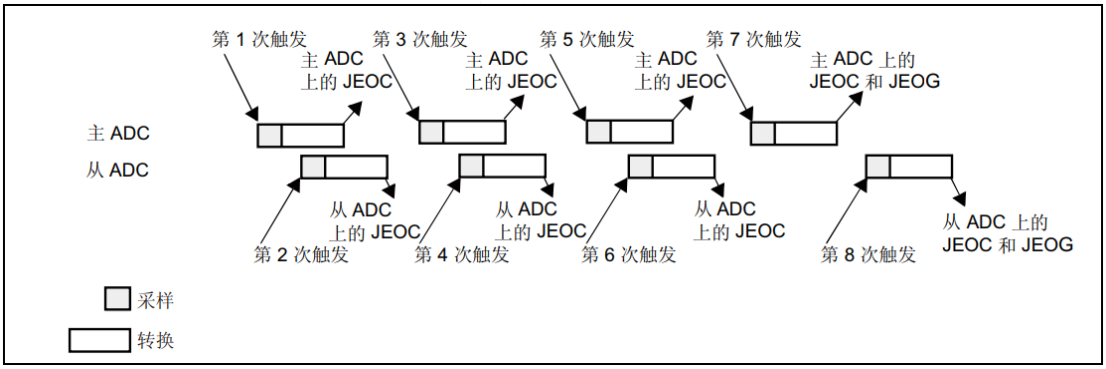
当组中的所有注入 ADC1 通道都转换完成后，会生成一个 JEOG 中断（如果已使能）。

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 30-11 交替触发：间断模式下的各个 ADC 的注入组



30.5.14.5. 独立模式

此模式里，双 ADC 同步不工作，每个 ADC 接口独立工作。

30.5.14.6. 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。

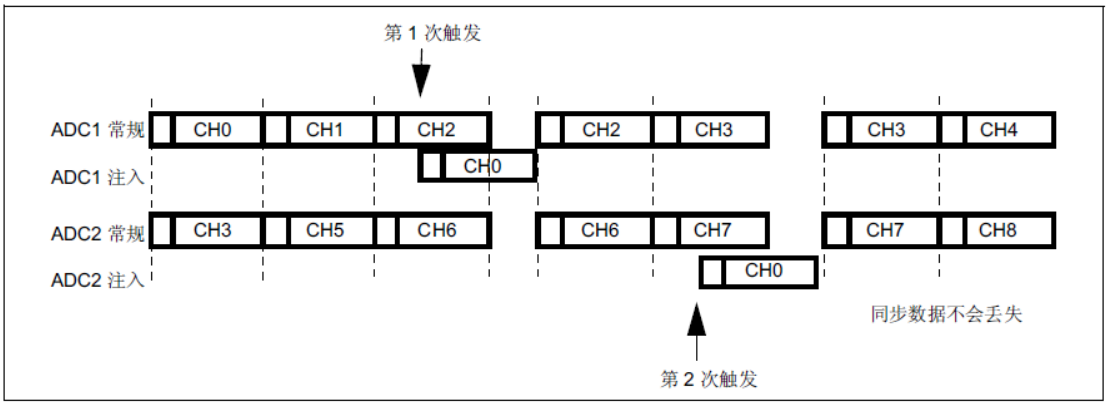
注：在混合的规则同步/注入同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

30.5.14.7. 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。

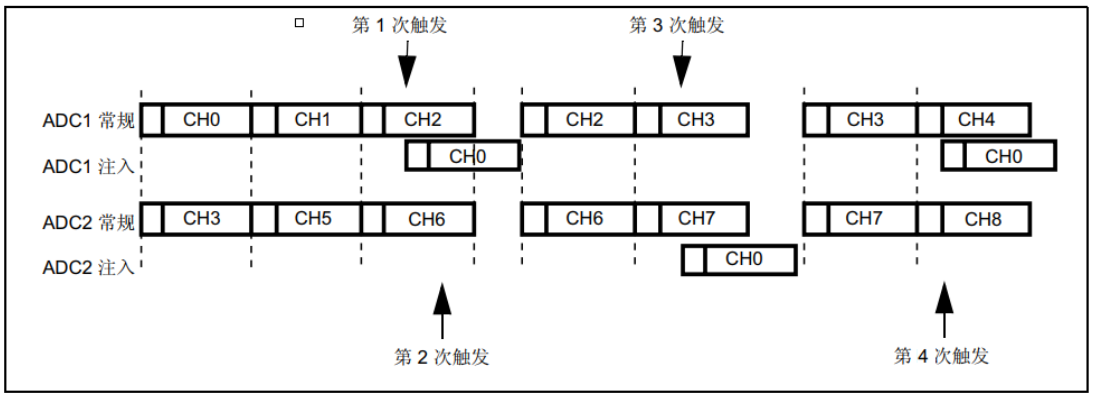
注：在混合的规则同步+交替触发模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

图 30-12 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）。

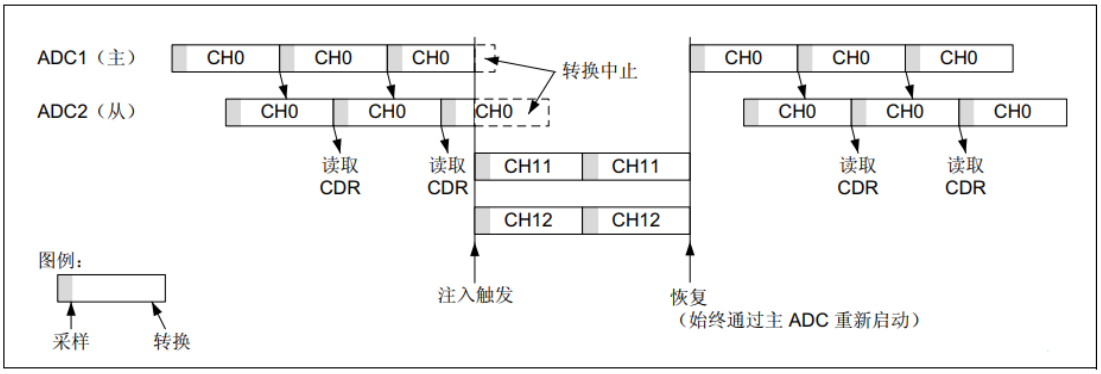
图 30-13 规则同步+交替触发：在注入转换期间出现新的注入触发事件



30.5.14.8. 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 30-14 混合的注入同步+规则交叉模式

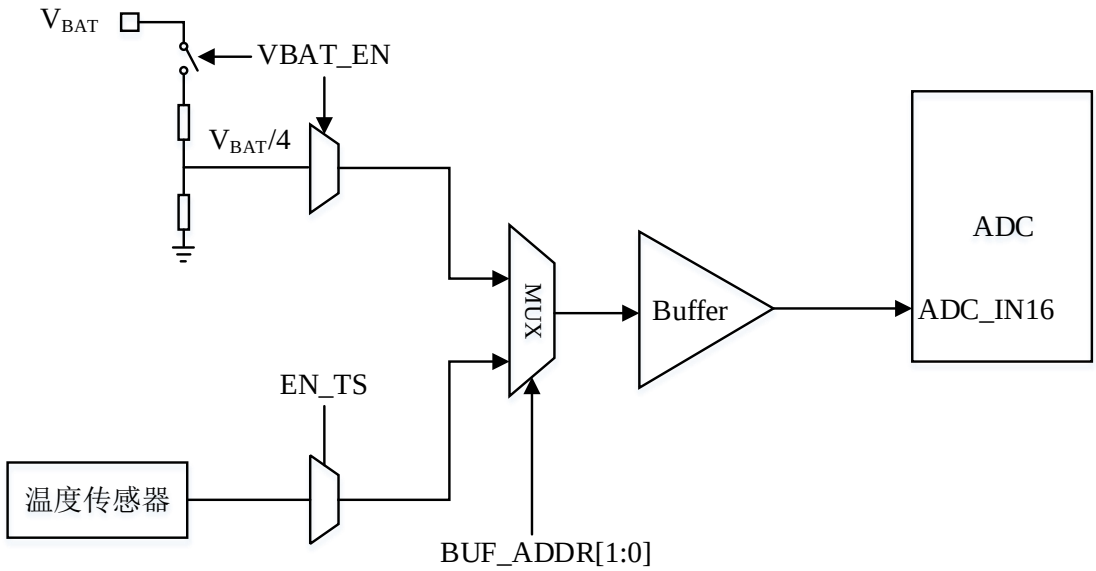


30.5.15. 温度传感器

温度传感器可以用来测量器件的环境温度(TA)。温度传感器在内部和 ADC 模拟输入通道 16 相连接，此通道把传感器输出的电压转换成数字值。

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 30-15 温度传感器



使用温度传感器读温度：

1）选择 ADC1_IN16 输入通道

2）选择采样时间

3）设置 ADC 共用温度传感器/REF 寄存器(ADC_CTSREF)的 EN_TS 位，以唤醒关电模式下的温度传感器，需要等待 10us，确保温度传感器稳定。

4）设置 ADC_CTSREF 的 BUF_ADDR 位，选择温度传感器，5us 后，温度传感器输出电压稳定。

5）通过设置软件触发或外部触发启动 ADC 转换。

6）转换完成后，读 ADC 数据寄存器上的 VTEMP 数据结果。

7）利用下列公式得出温度

温度(°C) = {(VTEMP - VOS) / KT_Slope}

这里：VOS = VTEMP 在 0°C 时的数值

KT_Slope = 温度与 VTEMP 曲线的平均斜率(单位为 mV/ °C)

30.5.16. VBAT 电池监测

ADC_CTSREF 寄存器中的 VBT_EN 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。VBT_EN 位置 1 时，会自动使能此桥。VBAT_EN 置 1 时，需等待 10us，等待 VBAT 电池检测模块稳定。配置 ADDR_BUF 为 1 时，选择了 VBAT 功能。会将 VBAT/4 连接到 ADC1_IN16 输入通道。因此，转换出的数字值为 VBAT 电压的四分之一。VBAT 电压可以通过 PB12 输入。

注：配置完 ADDR_BUF 后，5us 后输出电压稳定。

VBAT 电压可以用以下公式计算：

$$V_{BAT} = VREFP\frac{4 \times N}{4096}(V)$$

30.5.17. 差分信号转换

ADC 比较电路支持差分信号转换，选择差分模式（DIFFx_x=1）后，输入差分信号成对比较后再进行 AD 转换。差分信号分为 10 对：VIN0/VIN10、VIN1/VIN11、VIN2/VIN12、VIN3/VIN13、VIN4/VIN14、VIN5/VIN15、VIN6/VIN16、VIN7/VIN17、VIN8/VIN18、VIN9/VIN19。在 ADC_SQRx 或 ADC_JSQR 输入某一个通道号，按照通道号选择单端或差分模式。

对于差分模式，通道号只能选择正级（VIN0~VIN9）输入信号。差分模式支持把结果变为有符号数，通过设置 SIGNx_x=1 使能。

注：ADC1 的 VIN17-VIN19 和 ADC2 的 VIN16-VIN17 作为差分模式的负极通道已经接了固定的信号，不可选择，所以 ADC1 的 VIN17-19 和 ADC2 的 VIN16-VIN17 只能作为单端信号输入。ADC1 的 IN16 为 Buffer 通道，作为差分信号的输入端，需要输入端的信号是变化缓慢的信号，因此通常不用作差分信号转换。

30.5.18. 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。

30.5.19. 差分模式和有符号数

对应通道 VIN0~VIN19 支持单端和差分输入两种模式，其他通道只支持单端模式；同时与差分相对应，VIN0~VIN19 支持有符号和无符号两种输出。

单端模式下(DIFFx_x=0)，输入 VINx 为无符号数，无符号输出(SIGNx_x=0)为 DR[11:0]=VINx；有符号数最高位取反即为无符号数结果。例如 VIN0=0x004，配置 DIFF0_10=0/ SIGN0_10=0，则 DR[11:0]=0x004；配置 DIFF0_10=0/ SIGN0_10=1，则 DR[11:0]=0x804。

差分模式下(DIFFx_x=1)，输入 VINx_P 和 VINx_N 为无符号数，当 VINx_P>VINx_N，对应有符号输出(SIGNx_x=1)为 DR[11:0]=(VINx_P-VINx_N)/2，有符号数最高位取反即为无符号数结果。当 VINx_P<VINx_N，对应有符号输出(SIGNx_x=1)为 DR[11:0]= (~(VINx_N-VINx_P)/2)+1，有符号数最高位取反即为无符号数结果。一个差分对中序号小的为 VINx_P，序号大的为 VINx_N。例如当 VIN0=0x008, VIN10=0x000，配置 DIFF0_10=1/ SIGN0_10=0，则 DR[11:0]=(VIN0-VIN10)/2=0x004；在配置 DIFF0_10=1/ SIGN0_10=1 时，把最高位取反 DR=0x804。当 VIN0=0x000, VIN10=0x008，配置 DIFF0_10=1/ SIGN0_10=1，则 DR[11:0]=(~(VIN10-VIN0)/2)+1=0xffc；在配置 DIFF0_10=1/ SIGN0_10=0 时，把最高位取反 DR[11:0]=0x7fc。

30.5.20. 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。

30.5.21. 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 30-7 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

30.5.22. ADC 采样率计算

$$f_{SAMPLE} = \frac{f_{ADC}}{T_s + BIT + 3 - FASTMOD}$$

- fSAMPLE 为 ADC 采样率

- fADC 为 ADC 时钟，ADC_CR2 寄存器中定义了 fADC 相对于 HCLK 的分频数，fADC 最大不能超过 48MHZ
- 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
- FASTMOD 为快速转换模式

30.5.23. 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_CR2 寄存器中定义了 fADC 相对于 HCLK 的分频数，fADC 最大不能超过 48MHZ
- CADC 为内部采样和保持电容，CADC=10pF
- 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

30.6. 配置流程

30.6.1. 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 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）
- 结果是否为有符号位（ADC_SIGN）
- 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_DIFF 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）

30.6.2. 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 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）

30.6.3. 温度传感器操作

温度传感器的 VTEMP 与 ADC 转换电路的通道 16 相连，可将 BUF_ADDR 配置成 0 来选择温度传感器。为了使温度传感器工作正常，除了 ADC 通道转换的操作流程，还需要先进行额外的设置，流程如下：

1）设置 ADC_CTSREF 寄存器 EN_TS 为 1，使能温度传感器，温度传感器稳定需 10us 时间

2）设置 ADC_CTSREF 的 BUF_ADDR 位，选择温度传感器。配置完 BUF_ADDR 位 5us 后，VTEMP 输出电压稳定

30.7. ADC 寄存器描述

30.7.1. 寄存器列表

ADC 寄存器基地址：0x50000000

各个 ADC 模块地址范围：

偏移	名称
0x000~0x0FC	Master ADC1
0x100~0x1FC	Slave ADC2
0x300~0x3FC	Common Register

ADC 模块内部地址偏移：

偏移	名称	描述
0x00	ADC_SR	ADC 状态寄存器
0x04	ADC_IE	ADC 中断使能寄存器
0x08	ADC_CR1	ADC 控制寄存器 1
0x0C	ADC_CR2	ADC 控制寄存器 2
0x10	ADC_SMPR1	ADC 采样时间寄存器 1
0x14	ADC_SMPR2	ADC 采样时间寄存器 2
0x18	ADC_SMPR3	ADC 采样时间寄存器 3
0x1C	ADC_HTR	ADC 看门狗高阈值寄存器
0x20	ADC_LTR	ADC 看门狗低阈值寄存器
0x24	ADC_SQR1	ADC 规则序列寄存器 1
0x28	ADC_SQR2	ADC 规则序列寄存器 2
0x2C	ADC_SQR3	ADC 规则序列寄存器 3
0x30	ADC_JSQR	ADC 注入序列寄存器
0x34~0x40	ADC_JDRx	ADC 注入数据寄存器 x
0x48	ADC_DR	ADC 规则数据寄存器
0x4C	ADC_DIFF	ADC 单端/差分选择寄存器
0x50	ADC_SIGN	ADC 符号数选择寄存器
0x60~0x6C	ADC_OFRx	ADC 偏移寄存器 x
0x80	ADC_ANACFG	ADC 模拟配置寄存器
0x300	ADC_CSR	ADC 共用状态寄存器
0x304	ADC_CCR	ADC 共用控制寄存器
0x308	ADC_CDR	ADC 共用规则数据寄存器
0x30C	ASC_CTSREF	ADC 共用温度传感器/REF 寄存器

30.7.2. ADC 状态寄存器(ADC_SR: 00h)

位域	名称	属性	复位值	描述
31:8	RSV	-	-	保留
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

30.7.3. 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

30.7.4. 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，当 JEOC 标志置位时，由硬件清除该位 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 其他： TRIG19

15:14	EXTEN	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 10011：ADC 模拟输入通道 19

30.7.5. ADC 控制寄存器 2 (ADC_CR2: 0Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留

27	FASTMOD	RW	0	ADC 快速转换模式(转换时间最短) 注：正常转换模式需要 16 个 ADC_CLK，快速转换模式只需 15 个 ADC_CLK。 0：禁止 1：使能 注：模拟温度传感器不支持快速转换模式
26	AFE_RSTN	RW	0	ADCAFE 复位信号,复位 AFE 内部数字部分。在 ADC_EN 使能后，等待 5us 释放（设为高） 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 ADC_RSTN 复位 AFE 数字部分，然后启动新的转换 注：ADC 模块由模拟模块+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
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
15:6	RSV	-	-	保留

5:4	RES	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。当该位为 0 时，写入 1 将把 ADC 从断电模式下唤醒。启动时间需要 2us 0：关闭 ADC，并进入断电模式 1：开启 ADC

30.7.6. 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 0000：1 周期 0001：3 周期 0010：5 周期 0011：7 周期 0100：10 周期 0101：13 周期 0110：16 周期 0111：20 周期 1000：30 周期 1001：60 周期 1010：80 周期 1011：100 周期 1100：120 周期 1101：160 周期 1110：320 周期 1111：480 周期

30.7.7. 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

30.7.8. ADC 采样时间寄存器 3 (ADC_SMPR3: 18h)

位域	名称	属性	复位值	描述

31:16	RSV	-	-	保留
15:12	SMP19[3:0]	RW	0000	通道 19 采样时间 具体定义见 SMP0
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

30.7.9. 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，用有符号数表示
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

30.7.10. 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

30.7.11. 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(0~19) 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 10011：ADC 模拟输入通道 19
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

30.7.12. 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

30.7.13. 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

30.7.14. ADC 注入通道寄存器(ADC_JSQ: 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 其他： JTRIG19
26:25	JEXTEN	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(0~19)
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 10：3 次转换 11：4 次转换（JSQ1-JSQ2-JSQ3-JSQ4）

30.7.15. ADC 注入数据寄存器 x (ADC_JDRx: 34h~40h) (x 为 1~4)

位域	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:16	JCH[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	JDATA[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

30.7.16. 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

30.7.17. ADC 单端/差分选择寄存器(ADC_DIFF: 4Ch)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	DIFF9_19	RW	0	通道 9/19 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
8	DIFF8_18	RW	0	通道 8/18 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
7	DIFF7_17	RW	0	通道 7/17 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
6	DIFF6_16	RW	0	通道 6/16 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
5	DIFF5_15	RW	0	通道 5/15 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
4	DIFF4_12	RW	0	通道 4/14 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式

3	DIFF3_13	RW	0	通道 3/13 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
2	DIFF2_12	RW	0	通道 2/12 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
1	DIFF1_11	RW	0	通道 1/11 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式
0	DIFF0_10	RW	0	通道 0/10 单端/差分转换模式选择 该位由软件设置和清除 0：单端模式 1：差分模式

注：只对外部输入通道有效，内部通道恒定为单端模式。

30.7.18. ADC 符号数选择寄存器(ADC_SIGN: 50h)

位域	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	SIGN9_19	RW	0	通道 9/19 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
8	SIGN8_18	RW	0	通道 8/18 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
7	SIGN7_17	RW	0	通道 7/17 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
6	SIGN6_16	RW	0	通道 6/16 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
5	SIGN5_15	RW	0	通道 5/15 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数

4	SIGN4_14	RW	0	通道 4/14 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
3	SIGN3_13	RW	0	通道 3/13 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
2	SIGN2_12	RW	0	通道 2/12 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
1	SIGN1_11	RW	0	通道 1/11 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数
0	SIGN0_10	RW	0	通道 0/10 转换结果符号数选择 该位由软件设置和清除 0：结果为无符号数 1：结果为有符号数

注：只对外部输入通道有效，内部通道恒定为无符号数。

30.7.19. ADC 偏移寄存器 x (ADC_OFRx: 60h~6Ch) (x 为 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

30.7.20. ADC 模拟配置寄存器(ADC_ANACFG: 80h)

位域	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:3	CLK_DLY_SEL	RW	00	数据采样时钟延迟选择 00：默认延迟延迟时间 01：默认延迟延迟时间*1.2 10：默认延迟延迟时间*1.4 11：时钟下降沿采样
2:0	BIAS_C	RW	101	SAR 比较器 BIAS 功耗控制选择 000：25% bias 电流 001：50% bias 电流 010：75% bias 电流 011：100% bias 电流（1Msps default） 100：125% bias 电流 101：150% bias 电流（2Msps default） 110：175% bias 电流 111：200% bias 电流（3Msps default）

30.7.21. ADC 共用状态寄存器(ADC_CSR: 300h)

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

30.7.22. ADC 共用控制寄存器(ADC_CCR: 304h)

位域	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19:16	ADCDIV [3:0]	RW	0001	ADC_CLK 分频选择 该位由软件设置和清除，决定 ADC_CLK 相对于 HCLK 的分频数 注：ADC_CLK 最大不超过 48MHZ 0000：不分频 0001：2 分频 0010：3 分频 … 1110：15 分频 1111：16 分频
15:14	DMADUAL[1:0]	RW	0	双 ADC 下 DMA 功能选择 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])
13:12	RSV	-	-	保留

11:8	DELAY[3:0]	RW	0	2 个采样阶段之间的延迟 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 — 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

30.7.23. ADC 共用规则数据寄存器(ADC_CDR: 308h)

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

30.7.24. ADC 共用温度传感器和 REF 寄存器(ADC_CTSREF: 30Ch)

位域	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27	HIZ_EN	RW	1	当内嵌 VREF 关闭后，VREFBI 信号高阻使能位 高有效。VREFBI 信号高阻使能时，可以减少漏电，同时可以外部输入 VREFP 0：高阻禁止 1：高阻使能

26:25	VREFBI_SEL	RW	00	ADC 内嵌 VREF 电压选择 00：1.5V 01：2.0V 1x：2.5V 注：使用内嵌 VREF 时，VREFP 需断开不供电
24	VREFBI_EN	RW	0	ADC 内嵌 VREF 使能，不受 ADC_EN 控制。最大需要 200us 启动时间 0：禁止 1：使能 注：使用内嵌 VREF 时，VREFP 需断开不供电
23:19	VTRIM	RW	01110	内建 REF 电压输出 TRIM 值
18:15	TTRIM	RW	0111	内建 REF 电压温度系数控制（推荐使用默认值，客户可根据实测调整，具体方法请咨询厂家）
14:12	RSV	-	-	保留
11	VBAT_EN	RW	0	VBAT 通道使能 注：内嵌 1/4 分压电路，使能后可以实时测量电池电压，测量范围 1.0~5.5V，无需外部电阻分压。 0：VBAT 关闭 1：VBAT 使能
10:9	BUF_ADDR[1:0]	RW	00	BUF 通道功能选择 00：温度传感器 01：VBAT/4 xx：其它
8:5	ADJ_TD_OS[3:0]	RW	1000	温度传感器失调系数 TRIM 值（推荐使用默认值，客户可根据实测调整，具体方法请咨询厂家）
4:1	ADJ_TD_GA[3:0]	RW	1000	温度传感器增益系数 TRIM 值（推荐使用默认值，客户可根据实测调整，具体方法请咨询厂家）
0	EN_TS	RW	0	温度传感器使能信号 该位由软件设置和清除 0：温度传感器断电 1：温度传感器正常工作

31. 数模转换器（DAC）

31.1. 概述

数字/模拟转换器（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。

31.2. 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

31.3. 结构框图

下图为 DAC 控制模块的框图

图 31-1 DAC 结构框图

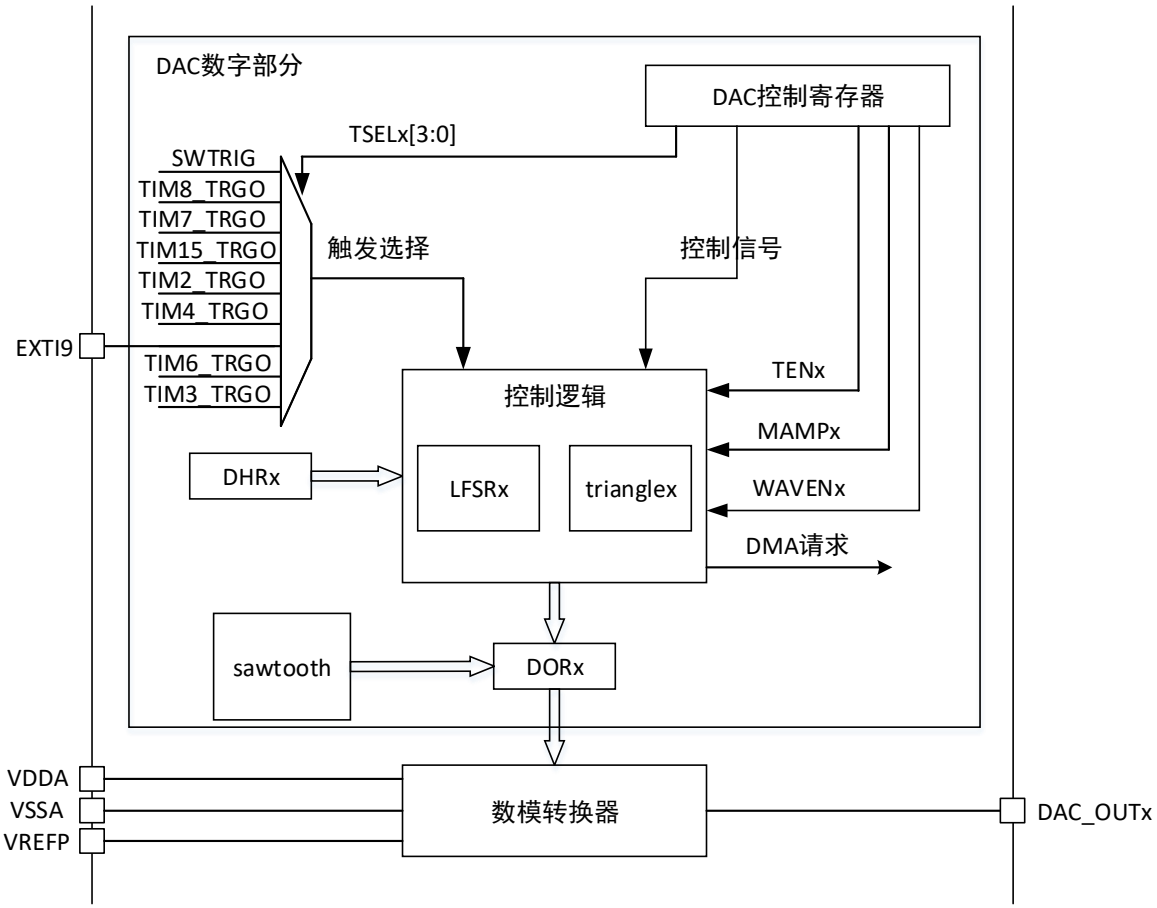


表 31-1 DAC 引脚

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，相应 GPIO 引脚（ PA4 或 PA5 ）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 PA4 或 PA5 引脚配置为模拟模式。

31.4. 功能描述

31.4.1. 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。

31.4.2. DAC 数据结构

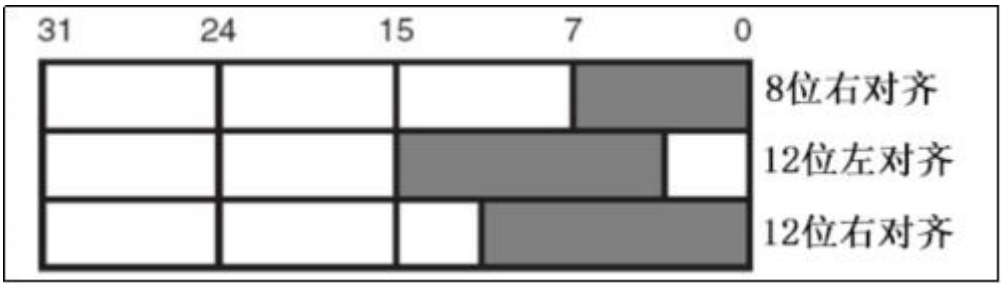
根据选择的配置模式，数据按照下文所述写入指定的寄存器：

■ 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 31-2 DAC 单通道模式的数据寄存器

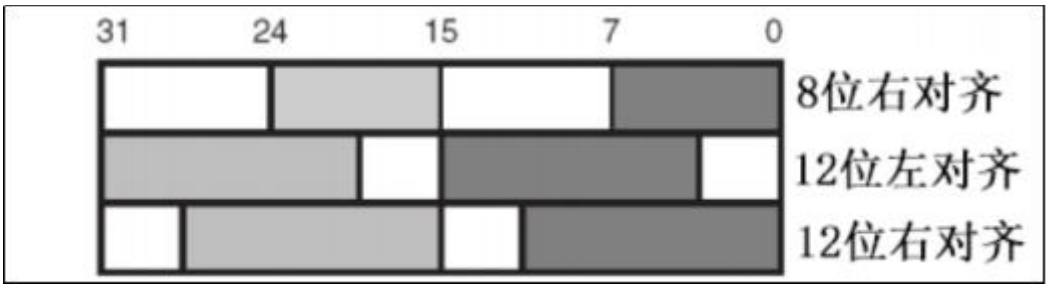


■ DAC 双通道，有 3 种情况：

- 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 31-3 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 31-2 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

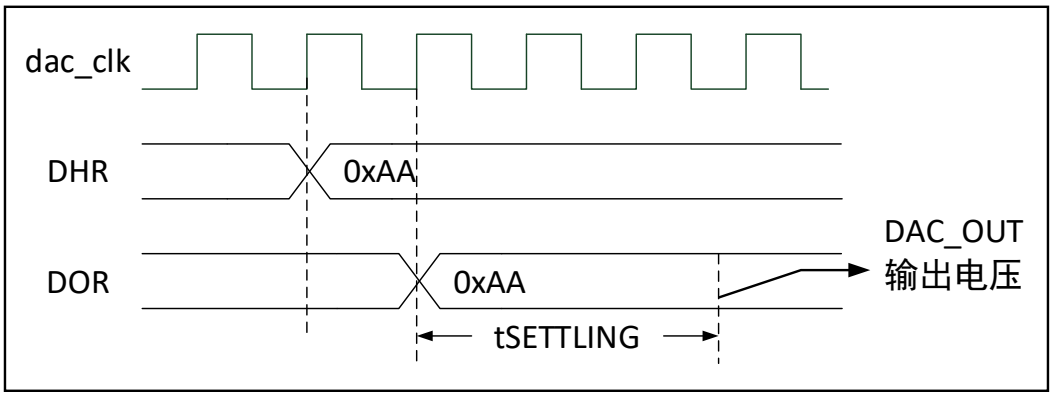
31.4.3. DAC 转换和输出电压

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 31-4 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)。

31.4.4. 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 31-3 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 31-4 递增触发选择

触发源	类型	STINCTRIGSELx[3:0]

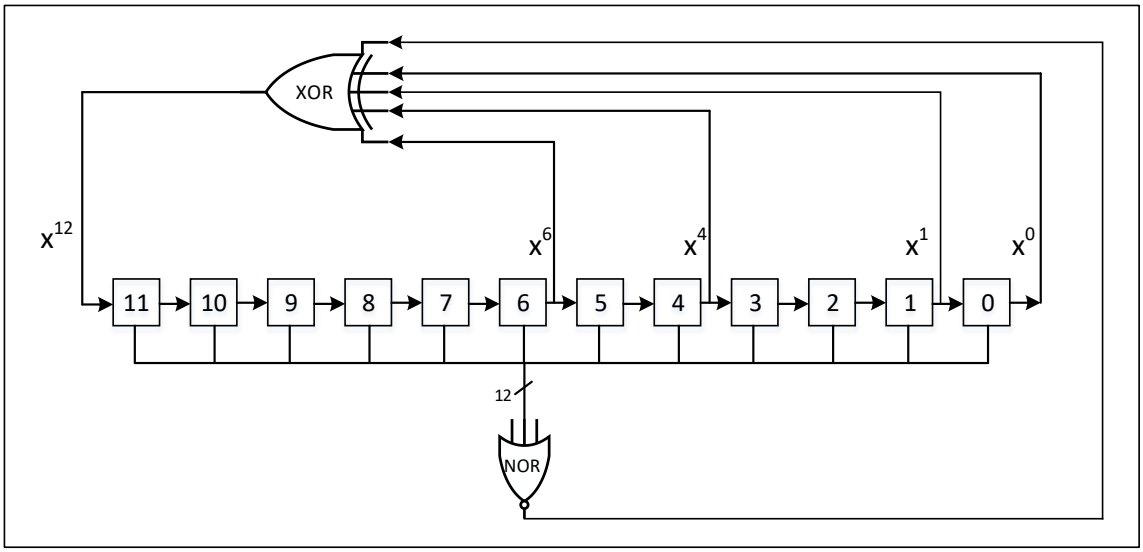
SWTRIGB	软件触发位	0000
TIM8_TRGO	内部定时器的触发信号	0001
TIM7_TRGO		0010
TIM15_TRGO		0011
TIM2_TRGO		0100
TIM4_TRGO		0101
EXTI10	外部管脚	0110
TIM6_TRGO	内部定时器的触发信号	0111
TIM3_TRGO		1000

31.4.5. 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。

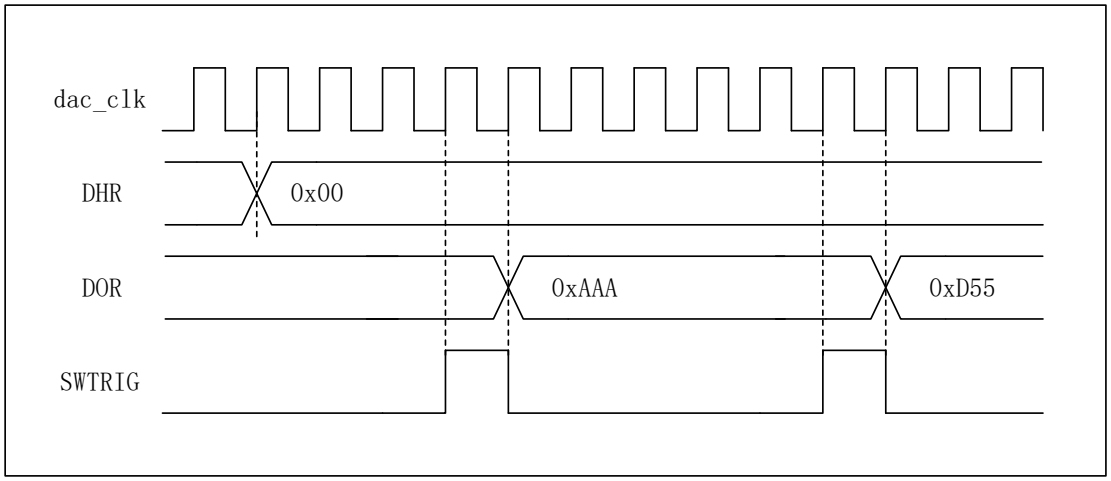
- LFSR 噪声模式：设置 WAVE[1:0]位为 01 选择 LFSR 噪声生成功能。在 DAC 控制逻辑中有一个线性反馈移位寄存器（LFSR）。寄存器 LFSR 的预装入值为 0xAAA。按照特定算法，在每次触发事件后更新该寄存器的值。

图 31-5 LFSR 噪声功能框图



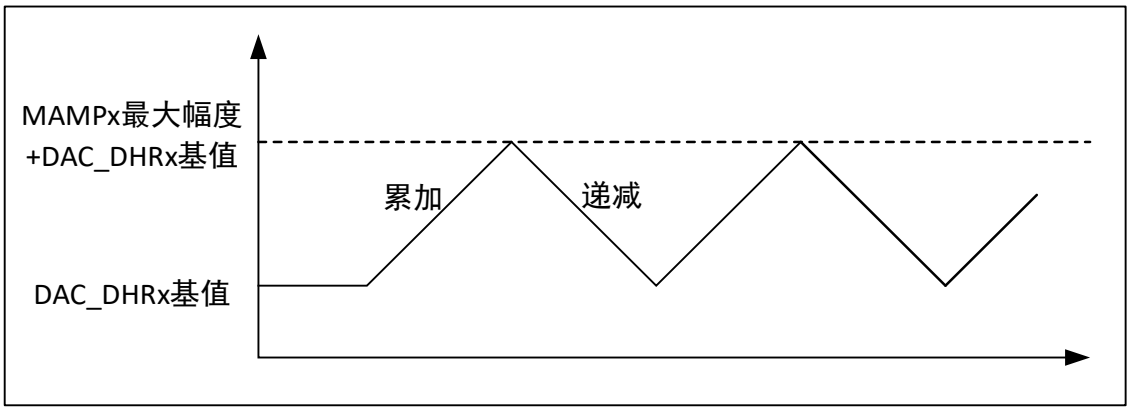
设置 DAC_CR 寄存器的 MAMPx[3:0]位可以屏蔽部分或者全部 LFSR 的数据，这样的得到的 LSFR 值与 DAC_DHRx 的数值相加，去掉溢出位之后即被写入 DAC_DORx 寄存器。MAMPx 设为 0000，只有 LFSR[0]有效,其余 bit 固定为 0；当 MAMPx 大于等于 1011 时，LFSR[11:0]全部有效。将 WAVEx[1:0]位置 0 可以复位 LFSR 波形的生成算法。为了产生噪声，必须使能 DAC 触发，即设 DAC_CR 寄存器的 TENx 位为 1。

图 31-6 带 LFSR 噪声模的 DAC 转换波形



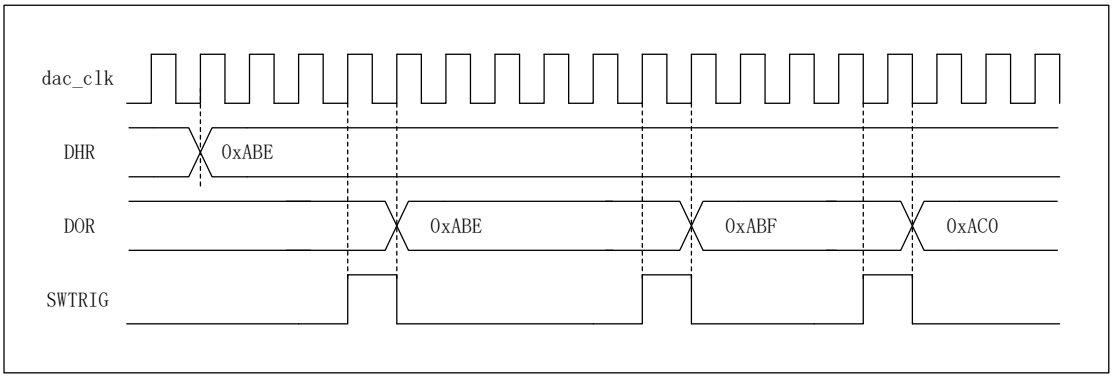
- 三角噪声模式：可以在 DC 或者缓慢变化的信号上加上一个小幅度的三角波噪声。设置 WAVEx[1:0]位为 10 选择 DAC 的三角波生成功能。设置 DAC_CR 寄存器的 MAMPx[3:0]位来选择三角波的幅度。内部的三角波计数器每次触发事件之后累加 1。计数器的值与 DAC_DHRx 寄存器的数值相加并丢弃溢出位后写入 DAC_DORx 寄存器。在传入 DAC_DORx 寄存器的数值小于 MAMP[3:0]位定义的最大幅度时，三角波计数器逐步累加。一旦达到设置的最大幅度，则计数器开始递减，达到 0 后再开始累加，依次循环。

图 31-7 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 31-8 带三角波噪声的 DAC 转换波形



31.4.6. 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 31-9 递减锯齿波（STDIR=0）

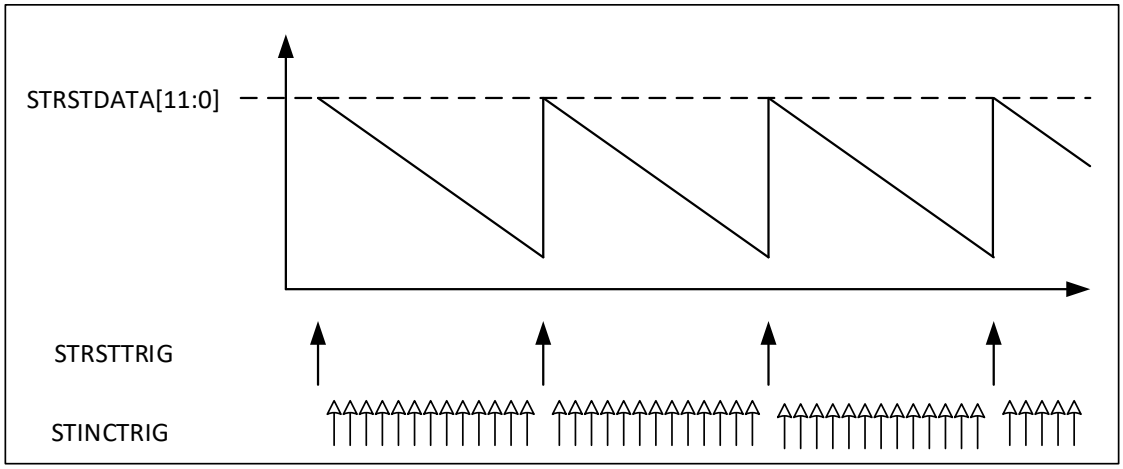
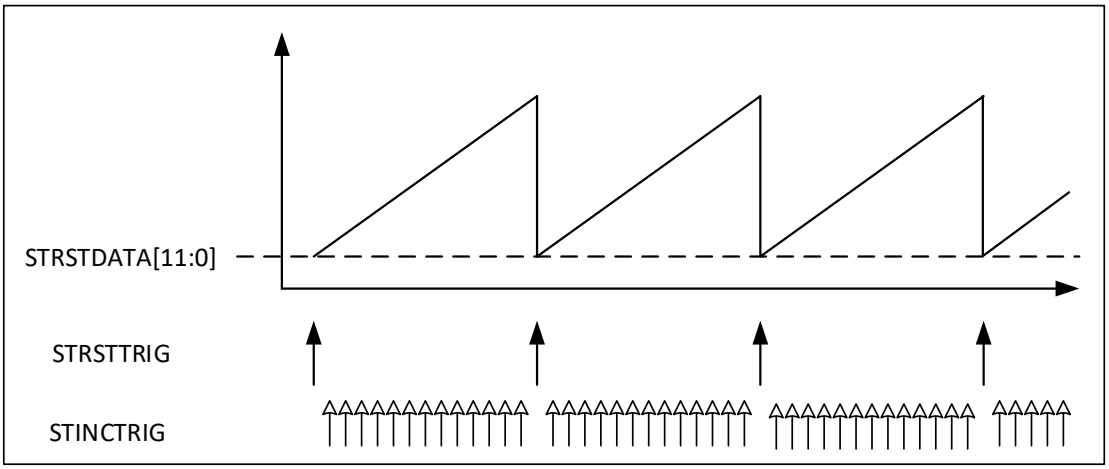


图 31-10 递增锯齿波（STDIR=1）



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

31.4.7. 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 RCL 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。

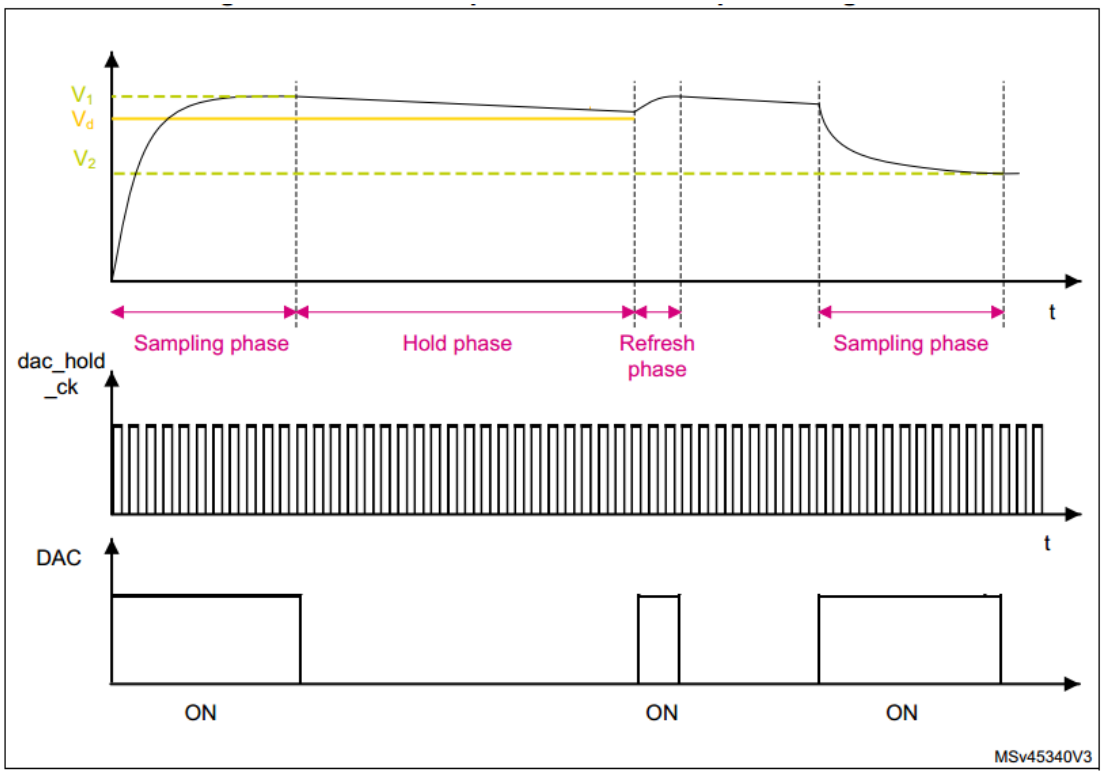
2）保持阶段。DAC 输出处于高阻状态，DAC 模拟部分和 BUFFER 处于关闭状态，用于降低功耗。保持（hold）时间通过 DAC_SHHR 寄存器的 THOLDx[9:0]位设置。

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 RCL 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 31-11 采样保持模式时序图



31.4.8. 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）。

31.4.9. 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。

31.4.9.1. 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）。

31.4.9.2. 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。

31.4.9.3. 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。

31.4.9.4. 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。

31.4.9.5. 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。

31.4.9.6. 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）。

31.4.9.7. 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）。

31.4.9.8. 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。

31.4.9.9. 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）。

31.4.9.10. 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。

31.4.9.11. 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。

31.4.9.12. 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。

31.4.9.13. 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。

31.4.9.14. 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）。

31.4.9.15. 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）。

31.5. 配置流程

31.5.1. 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)。

31.5.2. DAC 双通道操作流程

参见 DAC 双通道转换章节

31.6. DAC 寄存器描述

31.6.1. 寄存器列表

DAC 寄存器基地址：0x50000800

偏移	名称	描述
0x00	DAC_CR	DAC 控制寄存器
0x04	DAC_SWTRIGR	DAC 软件触发寄存器
0x08	DAC_DHR12R1	DAC 通道 1 12 位右对齐数据保持寄存器
0x0C	DAC_DHR12L1	DAC 通道 1 12 位左对齐数据保持寄存器
0x10	DAC_DHR8R1	DAC 通道 1 8 位右对齐数据保持寄存器
0x14	DAC_DHR12R2	DAC 通道 2 12 位右对齐数据保持寄存器
0x18	DAC_DHR12L2	DAC 通道 2 12 位左对齐数据保持寄存器

0x1C	DAC_DHR8R2	DAC 通道 2 8 位右对齐数据保持寄存器
0x20	DAC_DHR12RD	双 DAC 12 位右对齐数据保持寄存器
0x24	DAC_DHR12LD	双 DAC 12 位左对齐数据保持寄存器
0x28	DAC_DHR8RD	双 DAC 8 位右对齐数据保持寄存器
0x2C	DAC_DOR1	DAC 通道 1 数据输出寄存器
0x30	DAC_DOR2	DAC 通道 2 数据输出寄存器
0x34	DAC_SR	DAC 状态寄存器
0x38	DAC_CCR	DAC 校准控制寄存器
0x3C	DAC_MCR	DAC 模式控制寄存器
0x40	DAC_SHSR1	DAC 通道 1 采样时间寄存器
0x44	DAC_SHSR2	DAC 通道 2 采样时间寄存器
0x48	DAC_SHHR	DAC 保持时间寄存器
0x4C	DAC_SHRR	DAC 刷新时间寄存器
0x58	DAC_STR1	DAC 通道 1 锯齿波寄存器
0x5C	DAC_STR2	DAC 通道 2 锯齿波寄存器
0x60	DAC_STMODR	DAC 锯齿波模式寄存器

31.6.2. 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

31.6.3. 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。

31.6.4. 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。

31.6.5. 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

31.6.6. 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。

31.6.7. 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。

31.6.8. 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

31.6.9. 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。

31.6.10. 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。

31.6.11. 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

31.6.12. 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。

31.6.13. 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。

31.6.14. 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。

31.6.15. 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 RCL 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 RCL 标志。 0：同步完成 1：正在同步中，不能关闭 PCLK。
7:0	RSV	-	-	保留

31.6.16. 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

31.6.17. 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

31.6.18. 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

31.6.19. 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

31.6.20. 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

31.6.21. 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

31.6.22. 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

31.6.23. 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

31.6.24. 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

32. 模拟比较器(COMP)

32.1. 概述

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。

芯片内置四个超低功耗模拟比较器（COMP1、COMP2、COMP3、COMP4）。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。可用两个比较器组合成一个窗口比较器。

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。
-

32.2. 主要特性

- 支持电压比较功能
- 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
  - 内部基准电压 VREF 或 VDDA 的分压
- 输出极性可配置
- 可编程的迟滞窗口
- 支持输出重定向到用于触发一下事件：
  - 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
- 两个比较器可以组合构成窗口比较器
- 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
-

32.3. 结构框图

图 32-1 单个 COMP 结构框图

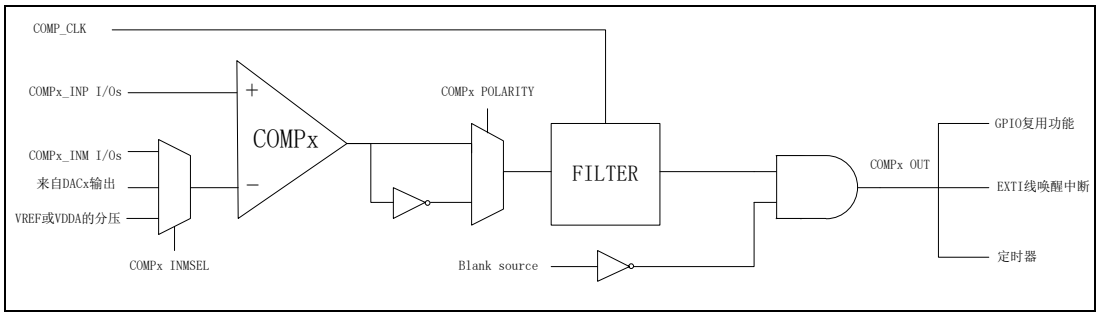
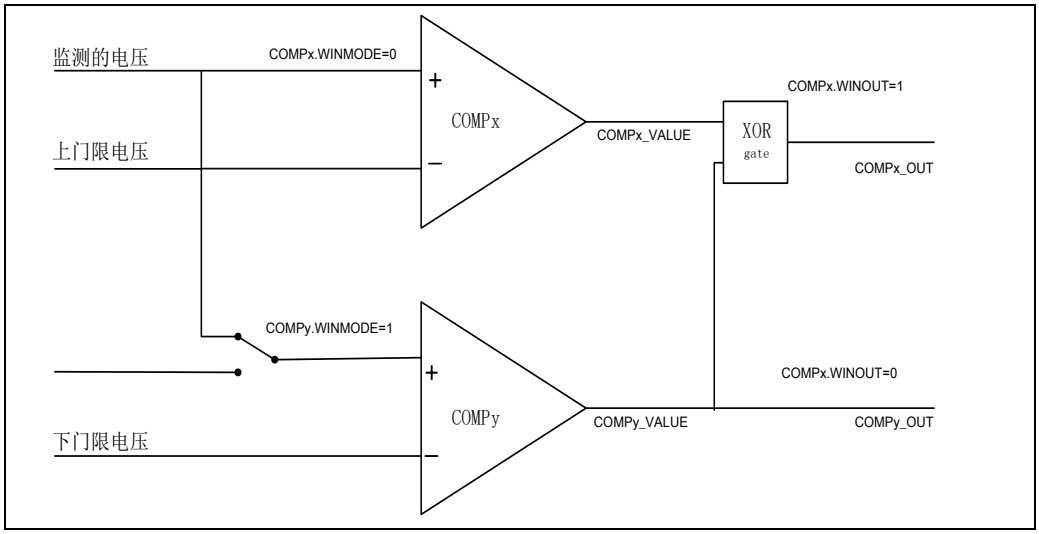


图 32-2 COMP 窗口模式示意图



32.4. 功能描述

32.4.1. 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 控制器中的 RCC_APB2_IPCKENR.CMPCKEN 比较器时钟使能位来使能比较器时钟。配置 RCC 控制器中 RCC_APB2_IPRSTR.CMPRST 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 控制器中的 RCC_CCR2.FLTCLK_SEL 选择 PCLK 的 32 分频时钟或 RCL 时钟。

32.4.2. 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 32-1 比较器正端输入

比较器	INPSEL[1:0]	GPIO 端口（模拟功能）
COMP1	00	PA1

	01	PB1
	10	PB10
	11	保留
COMP2	00	PA7
	01	PA3
	10	PB11
	11	保留
COMP3	00	PA0
	01	PC1
	10	PC3
	11	保留
COMP4	00	PB0
	01	PE7
	10	PE9
	11	保留

32.4.3. 负端输入

比较器负端输入可以选择来自芯片管脚、DAC 输出、或者来自内部基准电压（VREF 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 GPIO 时，须在 GPIOx_MD 中配置为模拟功能。

2）负端输入源为 DAC 时，须配置对应的 DAC 工作起来。

3）负端输入源为内部基准电压（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 32-2 比较器负端输入

比较器	INMSEL[1:0]	GPIO 端口（模拟功能）
COMP1	00	PA4（模拟功能）
	01	PA0（模拟功能）
	10	DAC1（须使能 DAC）
	11	VREF 或 VDDA（须使能分压，并配置分压系数）
COMP2	00	PA5（模拟功能）
	01	PA2（模拟功能）
	10	DAC2（须使能 DAC）
	11	VREF 或 VDDA（须使能分压，并配置分压系数）
COMP3	00	PF1（模拟功能）
	01	PC0（模拟功能）

	10	DAC1（须使能 DAC）
	11	VREF 或 VDDA（须使能分压，并配置分压系数）
COMP4	00	PE8（模拟功能）
	01	PB2（模拟功能）
	10	DAC1（须使能 DAC）
	11	VREF 或 VDDA（须使能分压，并配置分压系数）

32.4.4. 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。

32.4.5. 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。

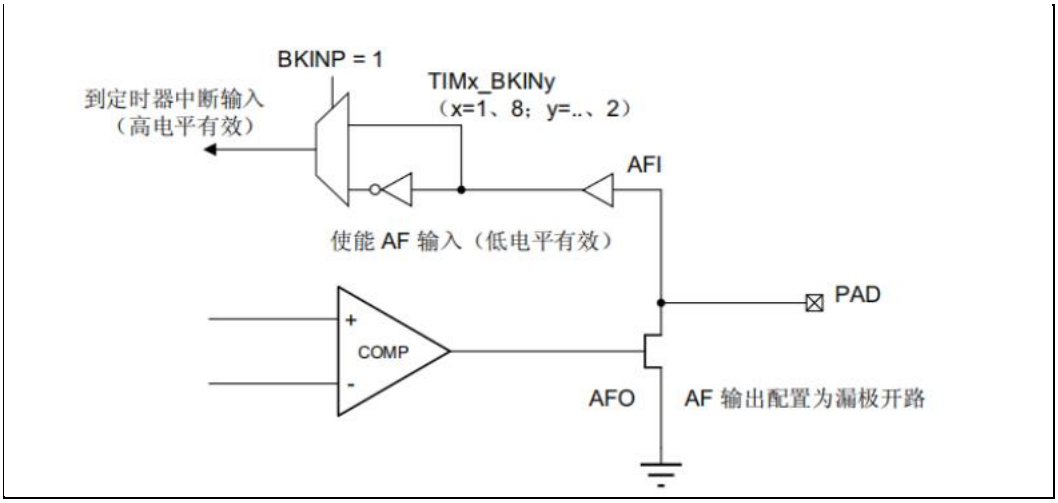
表 32-3 比较器输出

比较器	GPIO 端口	复用功能
COMP1	PA0	AF6
	PA6	AF6
	PA11	AF6
	PB0	AF6
	PB8	AF7
	PB10	AF6
COMP2	PA2	AF6
	PA7	AF6
	PA12	AF6
	PB5	AF6
	PB9	AF7
	PB11	AF6
COMP3	PB7	AF7
	PB15	AF6
	PC2	AF7
COMP4	PB1	AF7
	PB6	AF10
	PB14	AF7

32.4.6. 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 32-3 比较器输出重定向



32.4.7. 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.CMPRST 复位 COMP 模块才可清除。

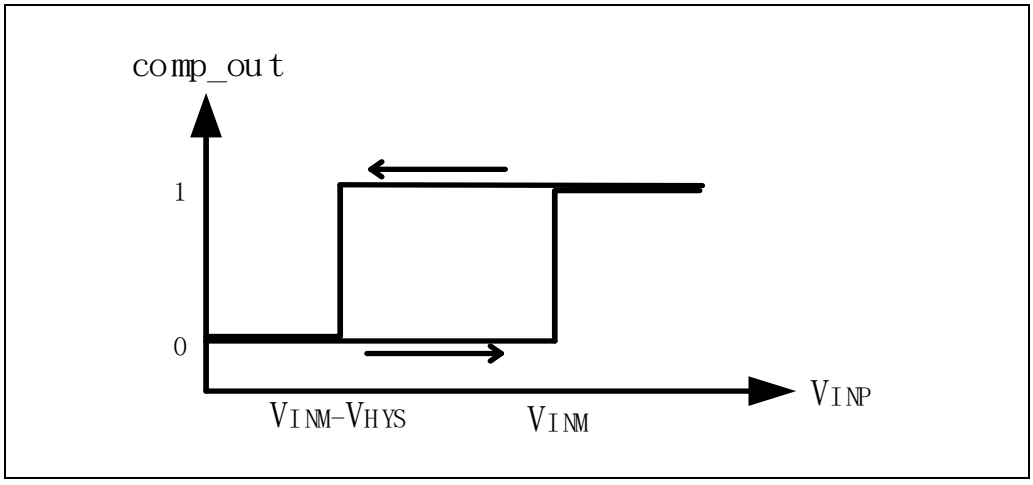
32.4.8. 迟滞比较

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 32-4 迟滞比较



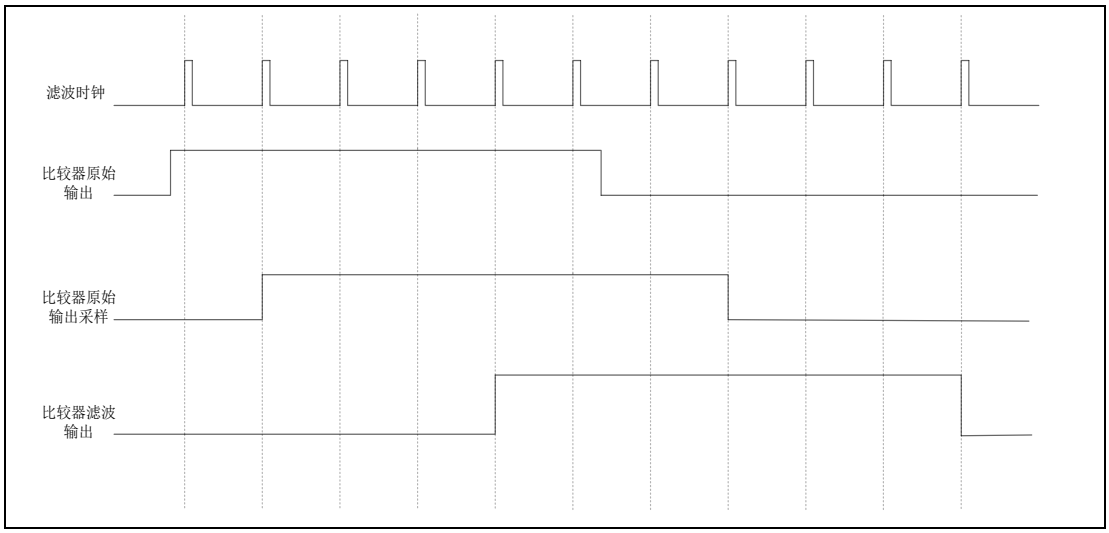
32.4.9. 输出滤波

滤波功能可以滤除来自比较器输入端的尖峰毛刺，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 RCL。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 RCL。

图例为 FLTTIME 设置为 0b001 时的比较器滤波输出。

图 32-5 比较器输出滤波



32.4.10. 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 32-6 比较器输出消隐

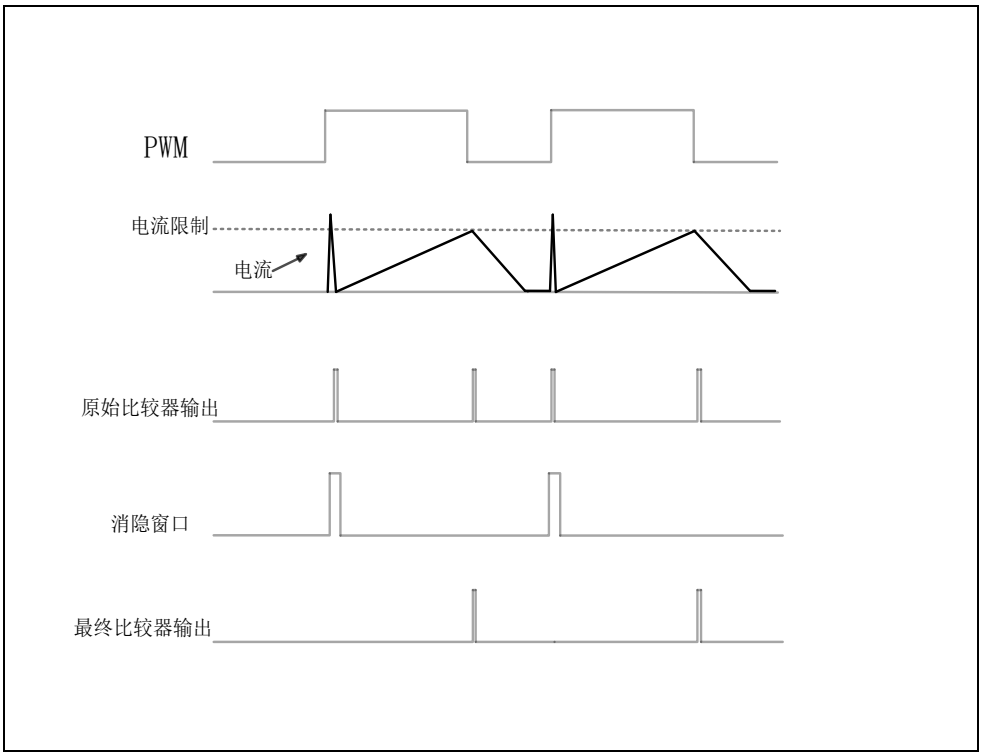


表 32-4 比较器消隐源

比较器	BLANKSEL[2:0]	输入源
COMP1	000	不消隐
	001	TIM1_OC5
	010	TIM2_OC3
	011	TIM3_OC3
	100	TIM8_OC5
	101	TIM1_OC4
	110	TIM15_OC1
	111	TIM4_OC3
COMP2	000	不消隐
	001	TIM1_OC5
	010	TIM2_OC3
	011	TIM3_OC3
	100	TIM8_OC5
	101	TIM1_OC4
	110	TIM15_OC1
	111	TIM4_OC3
COMP3	000	不消隐
	001	TIM1_OC5

	010	TIM3_OC3
	011	TIM2_OC4
	100	TIM8_OC5
	101	TIM8_OC4
	110	TIM15_OC1
	111	TIM4_OC3
COMP4	000	不消隐
	001	TIM3_OC4
	010	TIM8_OC5
	011	TIM15_OC1
	100	TIM1_OC5
	101	TIM8_OC4
	110	TIM15_OC1
	111	TIM4_OC3

32.4.11. 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

32.4.12. 窗口比较器

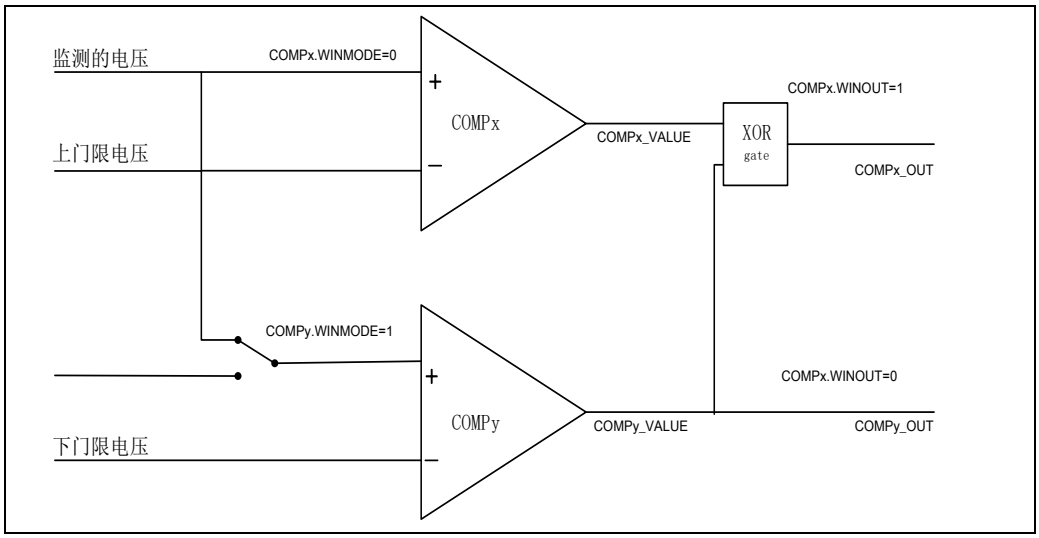
可以将两个比较器连接成窗口模式，设置两个负端为不同输入电压，其中一个为上门限电压，一个为下门限电压，监测的电压接到正向输入端，这样输入在门限电压之间时产生比较输出。

设置为窗口模式级联时，如下图所示：

1）需将 COMPx 的输出模式控制位域 COMP_CRx.WINOUT 设为 1，此时 COMPx 的输出为 COMPx_VALUE 异或 COMPy_VALUE 的结果。

2）需将 COMPy 的窗口模式正端输入选择位域 COMP_CRy.WINMODE 设为 1，此时 COMPy 的正端输入连接到 COMPx 的正端输入。

图 32-7 窗口比较器



32.5. 配置流程

32.5.1. 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）设置窗口模式正端输入选择 WINMODE（窗口模式可选）

4）设置输出模式 WINOUT（窗口模式可选）

5）设置输出极性 POLARITY

6）设置 CRV_SEL 选择基准分压源（内部基准源可选）

7）设置 CRV_EN 使能基准分压（基准分压可选）

8）设置分压系数 CRV_CFG（基准分压可选）

9）设置 FLTEN 使能滤波功能（滤波可选）

10）设置滤波时间配置（FLTTIME）（滤波可选）

11）设置 HYS 配置迟滞比较窗口值（迟滞比较可选）

12）设置 BLANKSEL 选择消隐源（消隐可选）

13）设置和切断源对应的系统定时器实现触发功能（切断可选）

14）设置 EN 使能比较器

15）设置控制寄存器 COMP_CR 的 LOCK 位锁定寄存器（可选）

16）在 VOUT 端测量或者在 COMP_SR 状态寄存器中读取输出信号

17）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）

18）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）

32.5.2. 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，参考《DAC 控制模块用户手册》

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

32.5.3. VREF 或 VDDA 分压作为负端输入源的配置流程

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内部基准电压，即 COMP_CRx.INMSEL=0b11

32.5.4. 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

32.5.5. 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

32.5.6. 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

32.5.7. 窗口比较器配置流程

按照比较器通用配置流程进行配置：通过配置 COMP_CRx.WINMODE 位域，将 COMPy 的正端输入连接到 COMPx 的正端输入；通过配置 COMP_CRx.WINOUT 位域控制比较器 1 输出模式。

32.6. COMP 寄存器描述

32.6.1. 寄存器列表

COMP 寄存器基地址：0x40010200

偏移	名称	描述
0x00	COMP_CR1	COMP1 控制寄存器
0x04	COMP_CR2	COMP2 控制寄存器
0x08	COMP_CR3	COMP3 控制寄存器
0x0c	COMP_CR4	COMP4 控制寄存器
0x10	COMP_SR	COMP 状态寄存器

32.6.2. 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，通过系统控制单元 SCU 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	BLANKTIME	RW	0	消隐时间
28:25	CRV_CFG	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
22	WINMODE	RW	0	比较器 1 窗口模式正端输入选择 0：比较器 1 的 INPSEL 决定 1：比较器 2 的正端
21	WINOUT	RW	0	比较器 1 输出模式控制 0：输出 VCOUT1 1：输出 VCOUT1 XOR VCOUT2

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	RW	0	比较器滤波时间配置，滤波时间按 FILT_CLK 计算，FILT_CLK 由系统控制单元 SCU 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	-	-	保留
14:12	BLANKSEL	RW	0	比较器 1 消隐源选择 000：不消隐 001：TIM1 OC5 010：TIM2 OC3 011：TIM3 OC3 100：TIM8_OC5 101：TIM1_OC4 110：TIM15_OC1 111：TIM4_OC3
11:8	INPSEL	RW	0	比较器 1 正端信号选择 0000：PA1 0001：PB1 0010：PB10 其它：保留
7:4	INMSEL	RW	0	比较器 1 负端信号选择 0000：PA4 0001：PA0 0010：DAC1 输出 0011：VREF 或 AVDD 的分压 其它：保留
3:1	HYS	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

32.6.3. COMP2 控制寄存器(COMP_CR2: 04h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	COMP_CR2 寄存器写保护控制 该位由软件设置，通过系统控制单元 SCU 复位清除 0：允许软件写入 COMP_CR2 寄存器 1：禁止软件写入 COMP_CR2 寄存器
30:29	BLANKTIME	RW	0	消隐时间
28:25	CRV_CFG	RW	0	基准电阻分压配置 分压为：CRV_CFG/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
22	WINMODE	RW	0	比较器 2 窗口模式正端输入选择 0：比较器 2 的 INPSEL 决定 1：比较器 1 的正端
21	WINOUT	RW	0	比较器 2 输出模式控制 0：输出 VCOUT2 1：输出 VCOUT1 XOR VCOUT2
20	POLARITY	RW	0	比较器 2 输出极性选择 0：直接输出 1：取反后输出
19	FLTEN	RW	0	比较器 2 滤波使能 0：禁止 1：使能
18:16	FLTTIME	RW	0	比较器滤波时间配置，滤波时间按 FILT_CLK 计算，FILT_CLK 由系统控制单元 SCU 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	-	-	保留
14:12	BLANKSEL	RW	0	比较器 2 消隐源选择 000：不消隐 001：TIM1 OC5 010：TIM2 OC3 011：TIM3 OC3 100：TIM8_OC5 101：TIM1_OC4 110：TIM15_OC1 111：TIM4_OC3
11:8	INPSEL	RW	0	比较器 2 正端信号选择 0000：PA7 0001：PA3 0010：PB11 其它：保留
7:4	INMSEL	RW	0	比较器 2 负端信号选择 0000：PA5 0001：PA2 0010：DAC2 输出 0011：VREF 或 AVDD 的分压 其它：保留
3:1	HYS	RO	0	比较器 2 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 2 使能位 0：禁止 1：使能

32.6.4. COMP3 控制寄存器(COMP_CR3: 08h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	COMP_CR3 寄存器写保护控制 该位由软件设置，通过系统控制单元 SCU 复位清除。 0：允许软件写入 COMP_CR3 寄存器 1：禁止软件写入 COMP_CR3 寄存器
30:29	BLANKTIME	RW	0	消隐时间
28:25	CRV_CFG	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
22	WINMODE	RW	0	比较器 3 窗口模式正端输入选择 0：比较器 3 的 INPSEL 决定 1：比较器 4 的正端
21	WINOUT	RW	0	比较器 3 输出模式控制 0：输出 VCOUT3 1：输出 VCOUT3 XOR VCOUT4
20	POLARITY	RW	0	比较器 3 输出极性选择 0：直接输出 1：取反后输出
19	FLTEN	RW	0	比较器 3 滤波使能 0：禁止 1：使能
18:16	FLTTIME	RW	0	比较器滤波时间配置，滤波时间按 FILT_CLK 计算，FILT_CLK 由系统控制单元进行配置 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	-	-	保留
14:12	BLANKSEL	RW	0	比较器 3 消隐源选择 000：不消隐 001：TIM1 OC5 010：TIM3 OC3 011：TIM2 OC4 100：TIM8_OC5 101：TIM8_OC4 110：TIM15_OC1 111：TIM4_OC3
11:8	INPSEL	RW	0	比较器 3 正端信号选择 0000：PA0 0001：PC1 0010：PE9 其它：保留

7:4	INMSEL	RW	0	比较器 3 负端信号选择 0000：PF1 0001：PC0 0010：DAC1 输出 0011：VREF 或 AVDD 的分压 其它：保留
3:1	HYS	RO	0	比较器 3 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 3 使能位 0：禁止 1：使能

32.6.5. COMP4 控制寄存器(COMP_CR4: 0ch)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	COMP_CR4 寄存器写保护控制 该位由软件设置，通过系统控制单元复位清除 0：允许软件写入 COMP_CR4 寄存器 1：禁止软件写入 COMP_CR4 寄存器
30:29	BLANKTIME	RW	0	消隐时间
28:25	CRV_CFG	RW	0	基准电阻分压配置 分压为：CRV_CFG/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
22	WINMODE	RW	0	比较器 4 口模式正端输入选择 0：比较器 4 的 INPSEL 决定 1：比较器 3 的正端
21	WINOUT	RW	0	比较器 4 输出模式控制 0：输出 VCOUT4 1：输出 VCOUT3 XOR VCOUT4
20	POLARITY	RW	0	比较器 4 输出极性选择 0：直接输出 1：取反后输出

19	FLTEN	RW	0	比较器 4 滤波使能 0：禁止 1：使能
18:16	FLTTIME	RW	0	比较器滤波时间配置，滤波时间按 FILT_CLK 计算，FILT_CLK 由系统控制单元进行配置。 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	-	-	保留
14:12	BLANKSEL	RW	0	比较器 4 消隐源选择 000：不消隐 001：TIM3 OC4 010：TIM8 OC5 011：TIM15 OC1 100：TIM1_OC5 101：TIM8_OC4 110：TIM15_OC1 111：TIM4_OC3
11:8	INPSEL	RW	0	比较器 4 正端信号选择 0000：PB0 0001：PE7 0010：PE10 其它：保留
7:4	INMSEL	RW	0	比较器 4 负端信号选择 0000：PE8 0001：PB2 0010：DAC1 输出 0011：VREF 或 AVDD 的分压 其它：保留
3:1	HYS	RO	0	比较器 4 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 4 使能位 0：禁止 1：使能

32.6.6. COMP 状态寄存器(COMP_SR: 10h)

位域	名称	属性	复位值	描述
31:4	REV	RO	0	保留位
7	VCOUT4_ORG	RO	0	比较器 4 原始输出状态
6	VCOUT3_ORG	RO	0	比较器 3 原始输出状态
5	VCOUT2_ORG	RO	0	比较器 2 原始输出状态
4	VCOUT1_ORG	RO	0	比较器 1 原始输出状态
3	VCOUT4	RO	0	比较器 4 滤波输出状态
2	VCOUT3	RO	0	比较器 3 滤波输出状态
1	VCOUT2	RO	0	比较器 2 滤波输出状态
0	VCOUT1	RO	0	比较器 1 滤波输出状态

33. 运算放大器(OPAMP)

33.1. 概述

OPAMP 模块包含三个独立配置的运算放大器。每个运算放大器有两个输入和一个输出，三个 I/O 可以连接到外部引脚。运算放大器可以在内部配置为一个跟随器，或者是一个可编程增益的放大器。放大器具有从 2 到 64 的非反相增益或从 -1 到 -63 的反相增益。正端输入可连接到内部 DAC。输出可连接到内部 ADC。

33.2. 主要特性

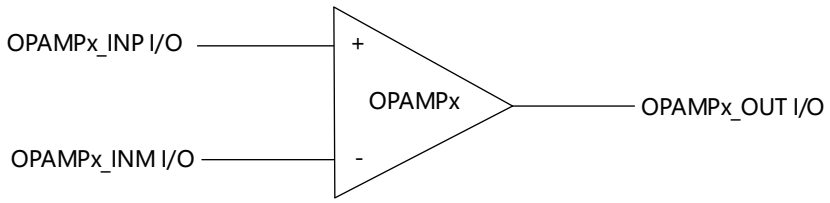
- 轨到轨输入输出
- 可配置成运放外置 SA 模式
- 可配置成可编程增益放大 PGA 模式，同相增益范围为 2 到 64， 反相增益范围为 -1 到 -63
- 可配置成跟随器 UG 模式，实现单位增益
- 运算放大器输出到 PAD
- 运算放大器输出可作为 ADC 的输入
- 运算放大器正端输入来自 PAD 或者是 DAC 的输出
- 运算放大器负端输入来自 PAD、UG 反馈网络输出或者是 PGA 的反馈电阻网络输出
-

33.3. 功能描述

运放可工作在正常模式和校准模式。校准模式下软件可对运放的失调电压进行校准。每个 OPAMP 均可单独使能，如果被禁止，则输出呈高阻态。

33.3.1. 结构框图

图 33-1 运放信号通路



33.3.2. 初始配置

运算放大器默认配置为正常模式，在该模式下，三个 IO 连接到外部引脚。在默认模式下，运算放大器使用出厂微调值。工作温度典型值为 25℃，工作电压 AVDD 典型值为 3.3V。配置 OPAMP1_CSR.HSM 比特来选择驱动模式为高功耗模式（最大驱动电流 500 微安）或低功耗模式（最大驱动电流 300 微安，默认）。

配置 OPAMPx_CSR.OPAEN 比特置 1，运算放大器即可工作。

33.3.3. 校准

运放可工作在正常模式和校准模式。

表 33-1 OPA 工作模式

模式	控制位			输出	
	EN	CAL_NEN	CAL_PEN	Vout	CALout flag
正常模式	1	0	0	analog	0
		1	1	analog	0
掉电模式	0	x	x	z	0
N 端校准模式	1	1	0	analog	x
P 端校准模式	1	0	1	analog	x

注：

1）当 Mode 为 Normal 模式，EN=1，CAL_NEN=1，CAL_PEN=1，TRIM_OSN/TRIM_OSP 中的校准值生效。当 Mode 为 Normal 模式，EN=1，CAL_NEN=0，CAL_PEN=0，TRIM_OSN/TRIM_OSP 中的校准值无效，此时采用默认值 0。

2）寄存器位 OPAMPx_CSR.CAL_OUT 作为校准完成的标志。

3）对于每个运算放大器和每种模式，可配置寄存器位 OPAMPx_CSR.TRIM_OSN 修改 N 差分对管失调电压的校准值，可配置寄存器位 OPAMPx_CSR.TRIM_OSP 修改 P 差分对管失调电压的校准值。

4）校准值可通过不断判断校准标志来计算，或从 NVR 中读取工厂校准值写入。OPA1 校准值 NVR 地址 0x0008024c, OPA2 校准值 NVR 地址 0x00080250, OPA3 校准值 NVR 地址 0x00080254。

校准配置流程：

1）配置 OPAMPx_CSR.EN 比特置 1，使能运算放大器。

2）配置 OPAMPx_CSR.TRIM_OSN 比特或 OPAMPx_CSR.TRIM_OSP 比特，修改 OPAMPx 的 N/P 差分对管失调电压的校准值。

3）配置 OPAMPx_CSR.CAL_NEN 或 OPAMPx_CSR.CAL_PEN 比特，使能 OPAMPx 的 N/P 差分对管失调电压校准。

4）校准完成标志判断，不断调整写入的校准值（0-31）直到校准完成标志置起。若写入的校准值是从 NVR 中读取的，则不用进行标志判断。

33.3.4. OPA 模式

OPA 在正常模式下可选择不同的工作模式。

表 33-2 OPA 工作模式

工作模式	MODE_SEL
SA(运放外置)	00
UG(单位增益)	01
PGA(倍数可调)	10
SA(运放外置)	11

33.3.4.1. 运放外置模式（SA 模式）

1）将运放配置成外置模式：MODE_SEL 配置成 0b00 或 0b11。

2）正端输入配置：通过 VINP_SEL 进行选择。具体参考信号走线章节。

3）负端输入配置：MODE_SEL 为 0b00 负端输入只可选择 VINM1，VINM0_EN 配置为 0；MODE_SEL 为 0b11 负端输入只可选择 VINM0，VINM0_EN 配置为 1。具体参考参考信号走线章节。

4）输出端配置：通过 OUT_SEL 进行选择。具体参考参考信号走线章节。

5）等效公式：若无外部电路，则 VOUT=VINP-VINM。若存在外部电路，则具体情况具体分析。

此时内部电阻被切断，需通过外部电路来调整运放功能。等效的电路如下图所示：

图 33-2 运放外置模式（MODE_SEL = 00）

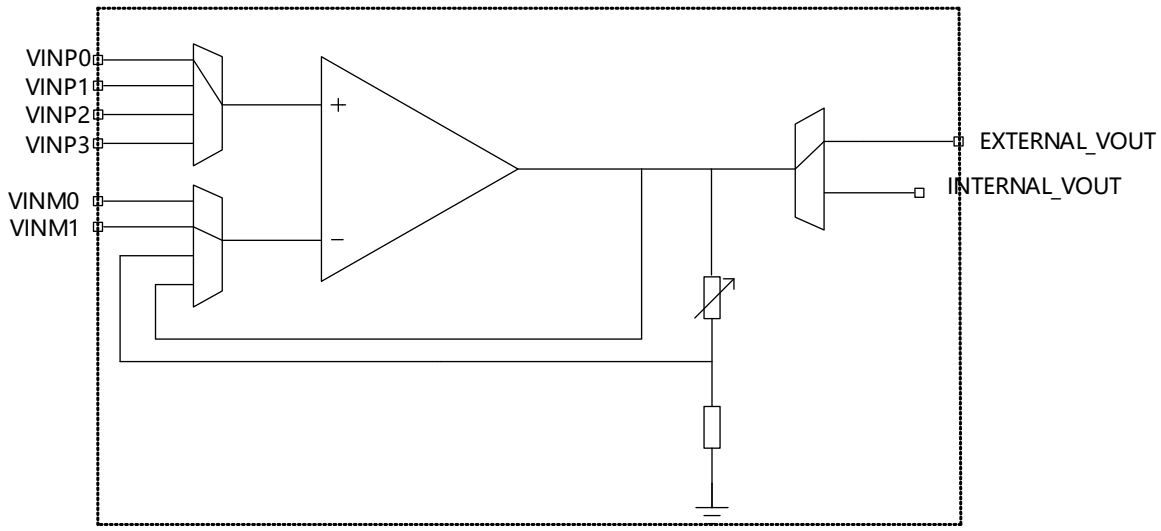
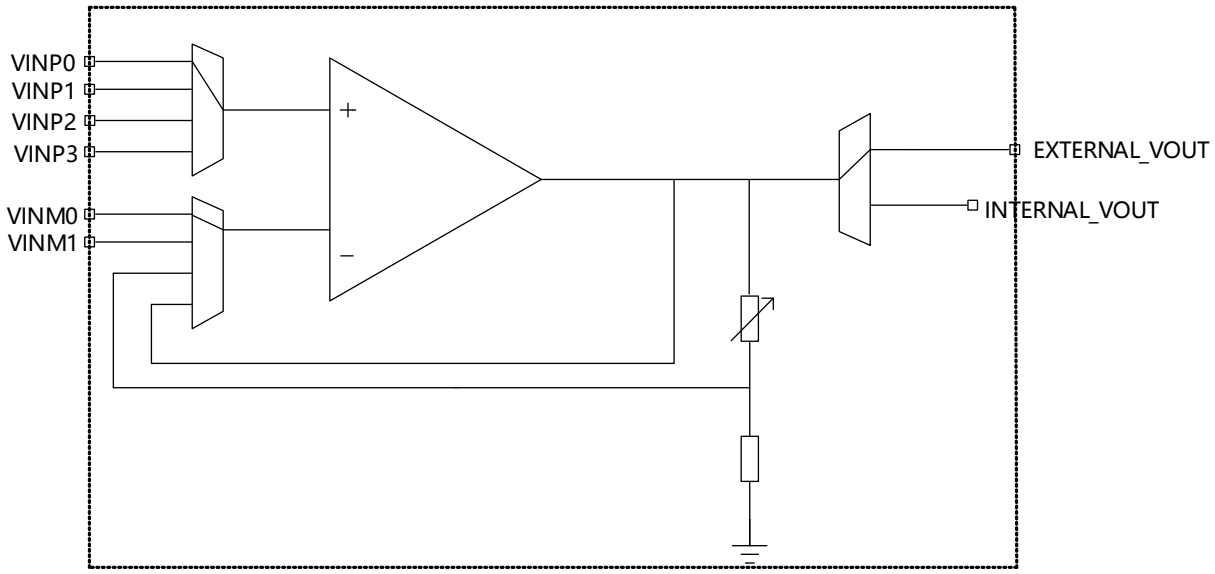


图 33-3 运放外置模式（MODE_SEL = 11）



33.3.4.2. 运放单位增益模式（UG 模式）

1）将运放配置成单位增益模式：MODE_SEL 配置成 0b01。

2）正端输入配置：通过 VINP_SEL 进行选择。具体参考参考信号走线章节。

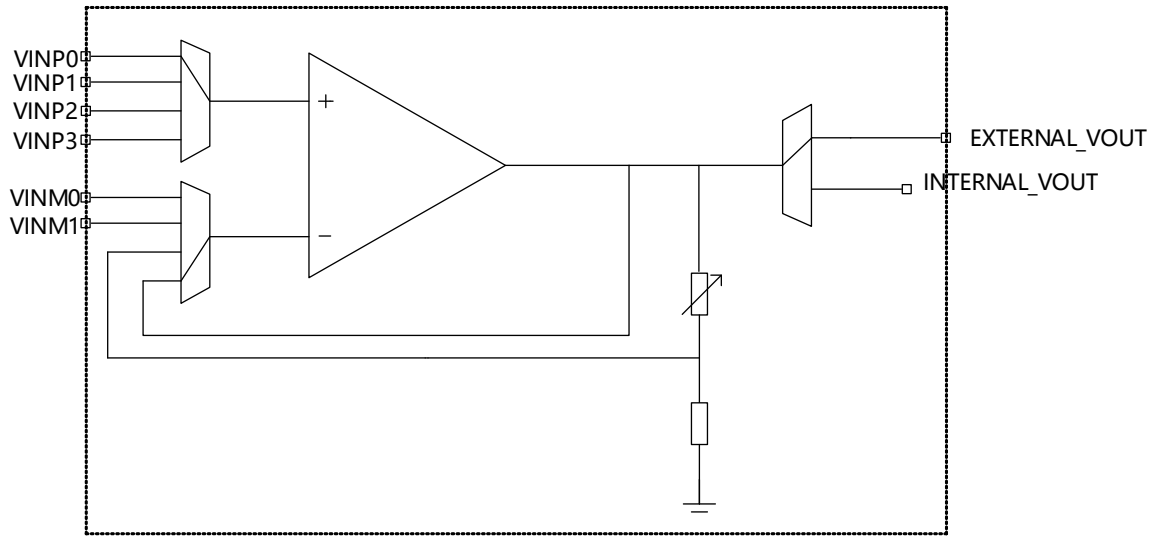
3）此模式下负端输入直接连接到输出端，所以外部负端输入无效。VINM0_EN 为 0。

4）输出端配置：通过 OUT_SEL 进行选择。具体参考参考信号走线章节。

5）等效公式：VOUT=VINP

此时内部电阻被切断，负端输入直接连接到输出端，此时运放增益为 1。等效的电路如下图所示：

图 33-4 运放单位增益模式



33.3.4.3. 可编程增益放大模式（PGA 模式）

1）将运放配置成可编程增益放大模式：MODE_SEL 配置成 0b10。

2）将运放配置成同相或反相模式：OPAx_CSR 中的 POL_SEL

3）设置增益倍数：GAIN_SEL，同相和反相的增益有区别。可以通过寄存器配置运放的放大倍数，从而可省略外部电阻。

4）正端输入配置：通过 VINP_SEL 进行选择。具体参考信号走线章节。

5）负端输入配置：配置为 PGA 同相模式时，负端输入端无效，VINM0_EN 为 0；配置为 PGA 反相模式时，负端输入有效，VINM0_EN 为 1。

6）输出端配置：通过 OUT_SEL 进行选择。具体参考参考信号走线章节。

7）等效公式：

- 同相：$VOUT=(1+R_X/R)*VINP$
- 反向：$VOUT=(1+R_X/R)*(VINP-VINM)+VINM$

图 33-5 可编程增益放大模式（同相）

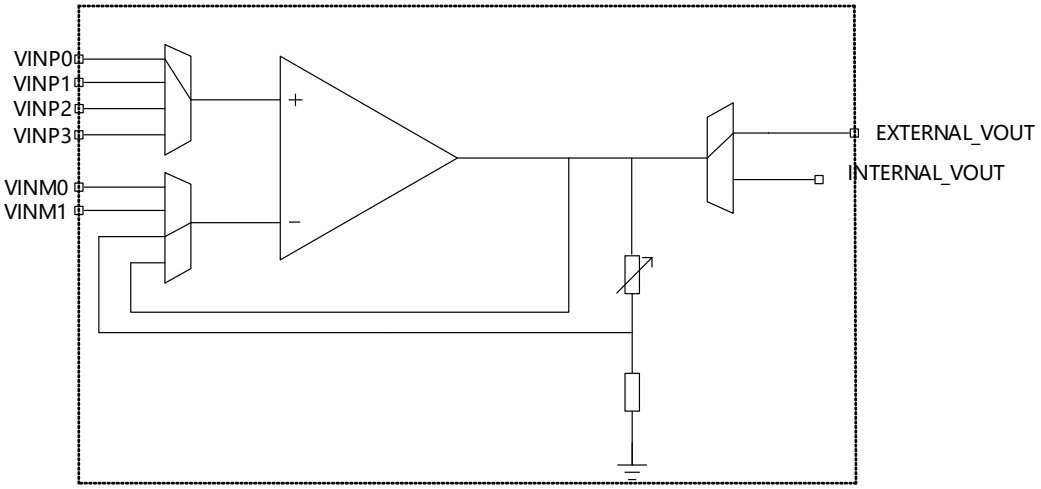
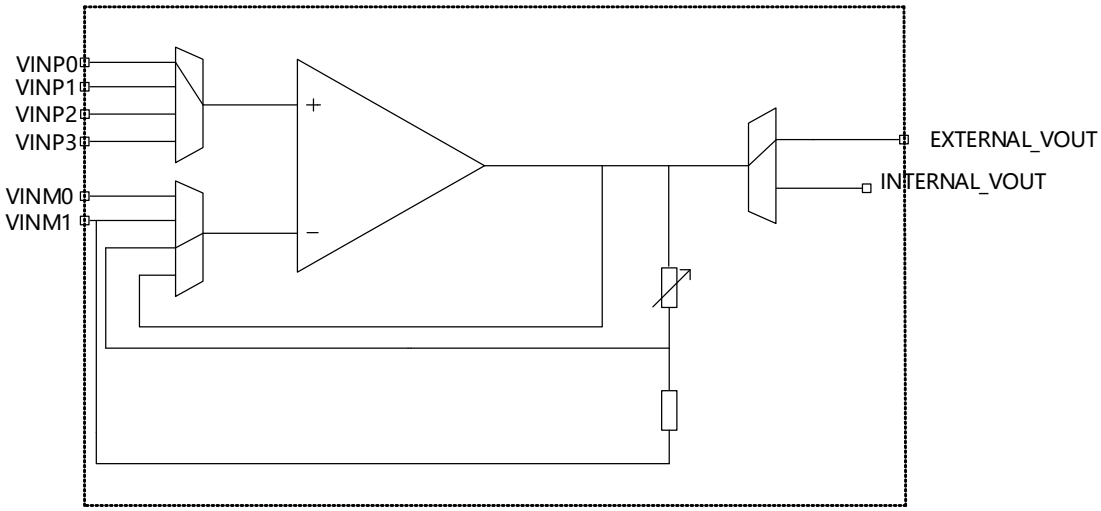


图 33-6 可编程增益放大模式（反相）



33.3.5. 信号走线

表 33-3 信号走线

工作模式	MODE_SEL	VINM0_EN	VINM1	VINM0
SA(运放外置)	00	0	输入	-
UG(单位增益)	01	0	-	-
PGA(倍数可调)	10	0	反相输入	-
SA(运放外置)	11	1	-	输入

注：若需要使用 PGA 反相模式，则只能选择 VINM1 作为负端输入。

表 33-4 OPA 引脚配置

OPA 编号	VINM1	VINM0	VINP	VOUT

	满足以下条件，VINM1 有效 1）MODE_SEL=0b00 或 MODE_SEL=0b10，POL_SEL=0b01 2）VINM0_EN=0	满足以下条件，VINM0 有效 1）MODE_SEL=0b11 2）VINM0_EN=1	通过 VINP_SEL 进行选择	通过 OUT_SEL 进行选择
OPA1	PC5	PA3	00：PA1 01：PA3 10：PA7 11：DAC1 输出	00：输出到 GPIO。 01：输出到内部（ADC 模块），并且断开外部输出。 其他：保留
OPA2	PC5	PA5	00：PA7 01：PB0 10：OPA1 输出 11：DAC2 输出	00：输出到 GPIO。 01：输出到内部（ADC 模块），并且断开外部输出。 其他：保留
OPA3	PB10	PB2	00：PB0 01：PA1 10：PB11 11：DAC1 输出	00：输出到 GPIO。 01：输出到内部（ADC 模块），并且断开外部输出。 其他：保留

33.4. 配置流程

33.4.1. 普通使用流程

1）将运放对应的 GPIO 口配置成模拟端口

2）参考 OPA 模式章节进行工作模式选择

3）参考信号走线章节进行正端输入、负端输入、输出端选择

4）驱动模式

5）校准：可选择默认值，或参考校准章节进行校准

- 使用默认值：CAL_NEN=0，CAL_PEN=0
- 参考校准章节进行校准

6）使能运放

33.4.2. 使用校准功能流程

参考校准章节进行校准

33.5. OPAMP 寄存器描述

33.5.1. 寄存器列表

OPAMP 寄存器基地址：0x40010300

偏移	名称	描述
0x00	OPAMP1_CSR	OPAMP1 控制/状态寄存器
0x04	OPAMP2_CSR	OPAMP2 控制/状态寄存器
0x08	OPAMP3_CSR	OPAMP3 控制/状态寄存器

33.5.2. 控制寄存器(OPAMP1_CSR: 00h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	OPAMP1_CSR 寄存器写保护控制 该位由软件设置，通过模块复位（RCC_APB2RSTR.OPARST 写 0）清除 0：允许软件写入 OPAMP1_CSR 寄存器 1：禁止软件写入 OPAMP1_CSR 寄存器
30:29	RSV	RO	0	保留位
28	HSM	RW	0	驱动模式选择 0：低驱动 1：高驱动
27:23	GAIN_SEL	RW	10000	PGA1 增益倍数选择 同相： 00000：64 00001：32 00010：16 00100：8 01000：4 10000：2 反相： 00000：63 00001：31 00010：15 00100：7 01000：3 10000：1
22:21	POL_SEL	RW	0	极性选择 00：同相 01：反相 其他：保留

20	VINM0_EN	RW	0	负端输入通道 0 使能 0：禁止 1：使能
19:18	MODE_SEL	RW	0	工作模式选择 00：SA（运放外置模式） 01：UG（单位增益） 10：PGA（倍数可调） 11：SA（运放外置模式）
17:16	VINP_SEL	RW	0	正端信号选择 00：PA1 01：PA3 10：PA7 11：DAC1 输出
15:14	OUT_SEL	RW	0	输出通道信号选择 00：输出到 GPIO 01：输出到内部，并且断开外部输出 其他：保留
13:9	TRIM_OSN	RW	0	OPAMP1 的 N 差分对管失调电压的修调值
8:4	TRIM_OSP	RW	0	OPAMP1 的 P 差分对管失调电压的修调值
3	CAL_OUT	RO	0	OPAMP1 的差分对管失调电压修调完成标志 0：未完成 1：完成
2	CAL_NEN	RW	0	OPAMP1 的 N 差分对管失调电压修调使能 0：禁止 1：使能
1	CAL_PEN	RW	0	OPAMP1 的 P 差分对管失调电压修调使能 0：禁止 1：使能
0	EN	RW	0	OPAMP1 使能位 0：禁止 1：使能

33.5.3. 控制寄存器(OPAMP2_CSR: 04h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	OPAMP2_CSR 寄存器写保护控制 该位由软件设置，通过模块复位（RCC_APB2RSTR.OPARST 写 0）清除 0：允许软件写入 OPAMP2_CSR 寄存器 1：禁止软件写入 OPAMP2_CSR 寄存器
30:29	RSV	RO	0	保留位

28	HSM	RW	0	驱动模式选择 0：低驱动 1：高驱动
27:23	GAIN_SEL	RW	10000	PGA2 增益倍数选择 同相： 00000：64 00001：32 00010：16 00100：8 01000：4 10000：2 反相： 00000：63 00001：31 00010：15 00100：7 01000：3 10000：1
22:21	POL_SEL	RW	0	极性选择 00：同相 01：反相 其他：保留
20	VINM0_EN	RW	0	负端输入通道 0 使能 0：禁止 1：使能
19:18	MODE_SEL	RW	0	工作模式选择 00：SA（运放外置模式） 01：UG（单位增益） 10：PGA（倍数可调） 11：SA（运放外置模式）
17:16	VINP_SEL	RW	0	正端信号选择 00：PA7 01：PB0 10：OPAMP1 输出 11：DAC2 输出
15:14	OUT_SEL	RW	0	输出通道信号选择 00：输出到 GPIO 01：输出到内部，并且断开外部输出 其他：保留
13:9	TRIM_OSN	RW	0	OPAMP2 的 N 差分对管失调电压的修调值
8:4	TRIM_OSP	RW	0	OPAMP2 的 P 差分对管失调电压的修调值

3	CAL_OUT	RO	0	OPAMP2 的差分对管失调电压修调完成标志 0：未完成 1：完成
2	CAL_NEN	RW	0	OPAMP2 的 N 差分对管失调电压修调使能 0：禁止 1：使能
1	CAL_PEN	RW	0	OPAMP2 的 P 差分对管失调电压修调使能 0：禁止 1：使能
0	EN	RW	0	OPAMP2 使能位 0：禁止 1：使能

33.5.4. 控制寄存器(OPAMP3_CSR: 08h)

位域	名称	属性	复位值	描述
31	LOCK	RW	0	OPAMP3_CSR 寄存器写保护控制 该位由软件设置，通过模块复位（RCC_APB2RSTR.OPARST 写 0）清除 0：允许软件写入 OPAMP3_CSR 寄存器 1：禁止软件写入 OPAMP3_CSR 寄存器
30:29	RSV	RO	0	保留位
28	HSM	RW	0	驱动模式选择 0：低驱动 1：高驱动
27:23	GAIN_SEL	RW	10000	PGA3 增益倍数选择 同相： 00000：64 00001：32 00010：16 00100：8 01000：4 10000：2 反相： 00000：63 00001：31 00010：15 00100：7 01000：3 10000：1

22:21	POL_SEL	RW	0	极性选择 00：同相 01：反相 其他：保留
20	VINM0_EN	RW	0	负端输入通道 0 使能 0：禁止 1：使能
19:18	MODE_SEL	RW	0	工作模式选择 00：SA（运放外置模式） 01：UG（单位增益） 10：PGA（倍数可调） 11：SA（运放外置模式）
17:16	VINP_SEL	RW	0	正端信号选择 00：PB0 01：PA1 10：PB11 11：DAC1 输出
15:14	OUT_SEL	RW	0	输出通道信号选择 00：输出到 GPIO。 01：输出到内部，并且断开外部输出。 其他：保留
13:9	TRIM_OSN	RW	0	OPAMP3 的 N 差分对管失调电压的修调值。
8:4	TRIM_OSP	RW	0	OPAMP3 的 P 差分对管失调电压的修调值。
3	CAL_OUT	RO	0	OPAMP3 的差分对管失调电压修调完成标志 0：未完成 1：完成
2	CAL_NEN	RW	0	OPAMP3 的 N 差分对管失调电压修调使能 0：禁止 1：使能
1	CAL_PEN	RW	0	OPAMP3 的 P 差分对管失调电压修调使能 0：禁止 1：使能
0	EN	RW	0	OPAMP3 使能位 0：禁止 1：使能

34. CRC 计算单元

34.1. 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。

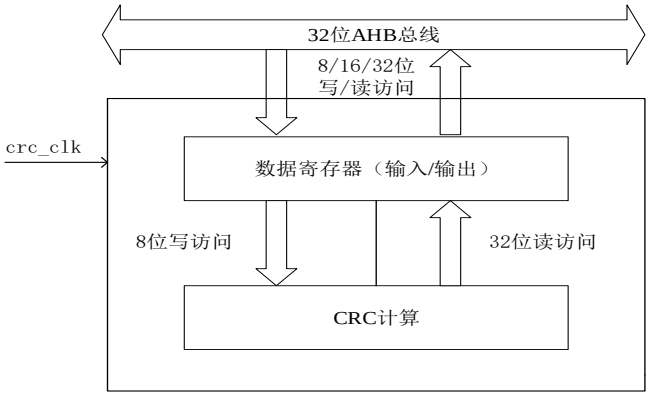
34.2. 主要特性

- 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；

34.3. 功能说明

34.3.1. 功能框图

图 34-1 CRC 功能框图



34.3.2. 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。

34.3.2.1. 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。

34.3.2.2. 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。

34.3.2.3. 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

34.3.2.4. 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。

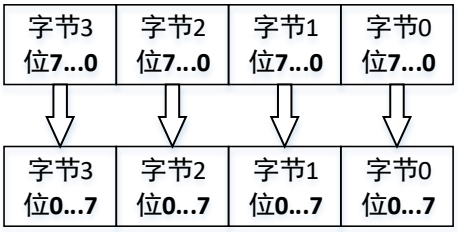
■ 输入数据倒序功能

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：

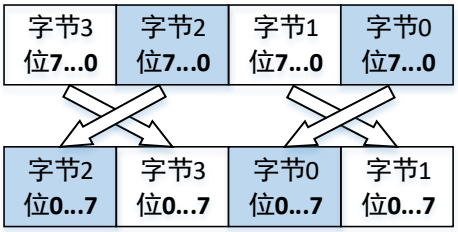
1）按字节倒序：

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



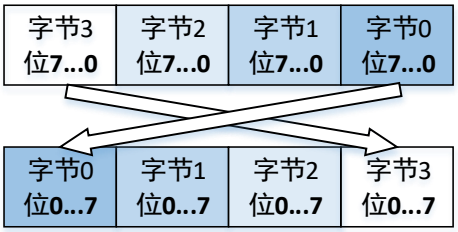
2）按半字倒序：

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

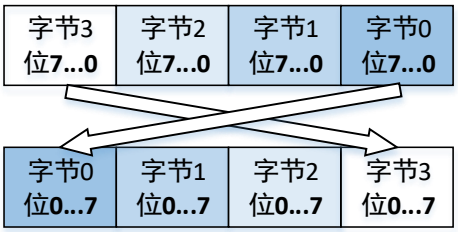


■ 输出数据倒序功能

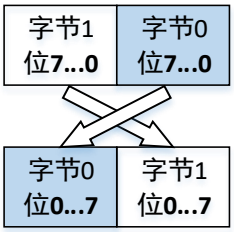
通过将 CRC_CTRL 寄存器中 RSLT_REV 位置 1 可以将输出数据倒序。对输出数据来说，倒序形式为根据多项式长度按位进行倒序。

举例说明如下：

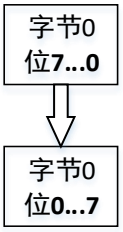
CRC32 计算结果 0x11223344 将被倒序成 0x22CC4488



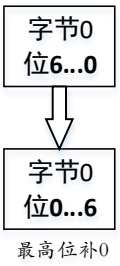
CRC16 计算结果 0x4488 将被倒序成 0x1122



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。

34.3.2.5. 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

34.3.3. 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。

34.4. 寄存器描述

34.4.1. 寄存器列表

CRC 寄存器基地址：0x40010C00

偏移	名称	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

34.4.2. 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

34.4.3. 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

34.4.4. 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

34.4.5. 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

34.4.6. 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

34.4.7. 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。

35. 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。

35.1. 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）

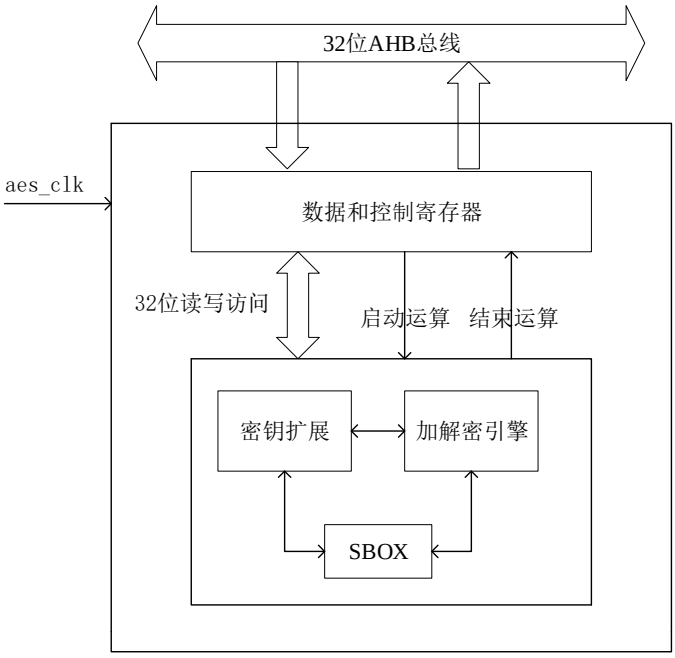
35.2. 功能说明及框图

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 35-1 AES 加解密处理器框图



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

35.3. 加解密模式

AES-ECB 模式

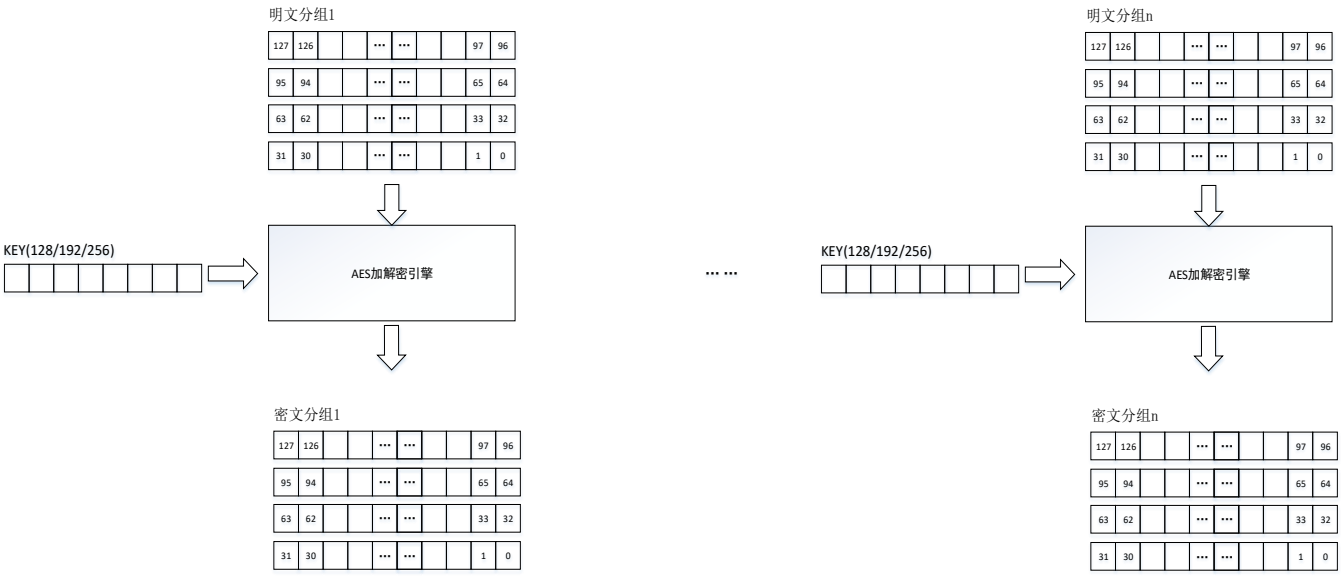
AES-ECB 模式加密流程如下：

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。

图 35-2 AES-ECB 模式加密示意图



AES-ECB 模式解密流程如下：

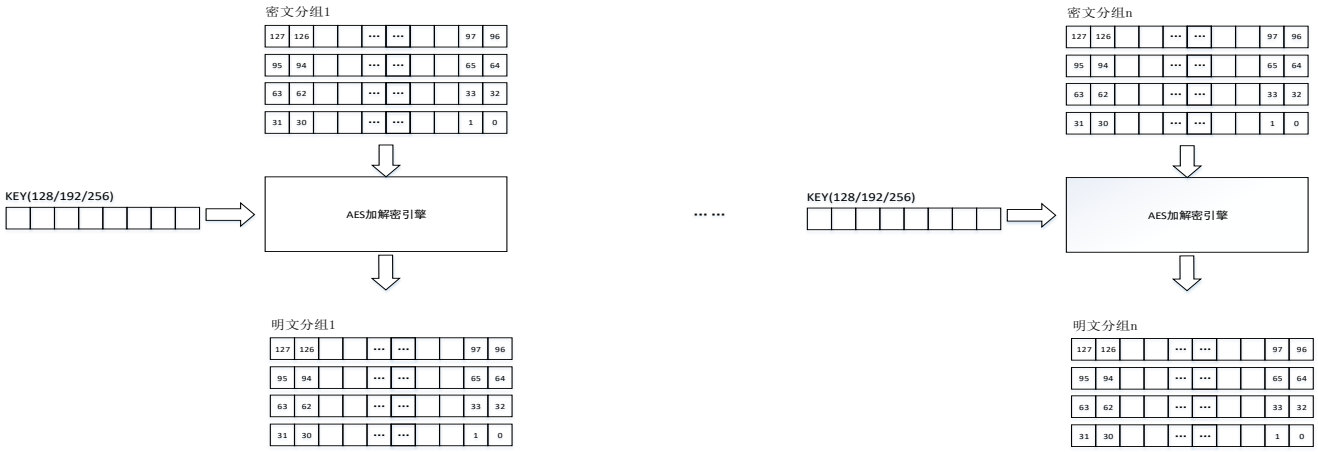
1）将密文按照 128bit 长度进行分组，不足 128bit 长度的分组需按照一定规则进行填充。

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

图 35-3 AES-ECB 模式解密示意图



AES-CBC 模式

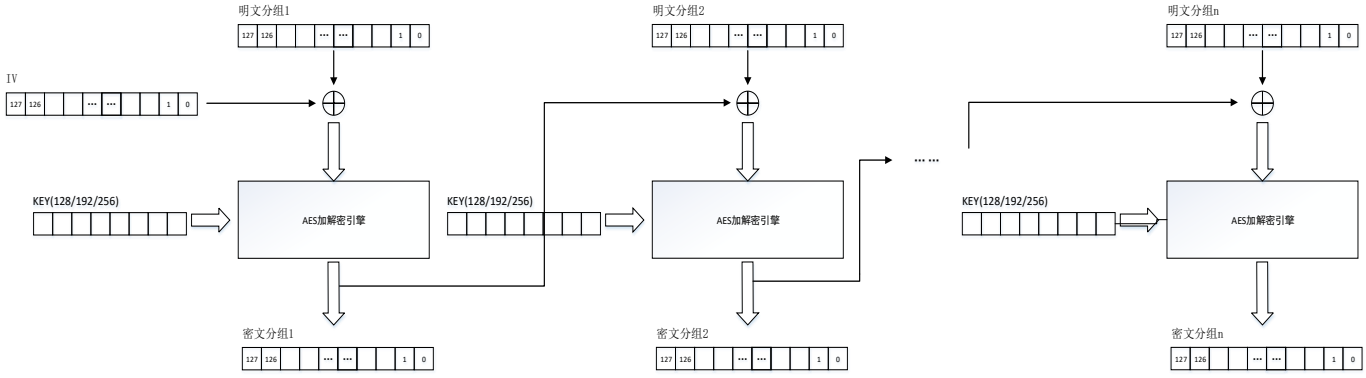
AES-CBC 模式加密流程如下：

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。

图 35-4 AES-CBC 模式加密流程图



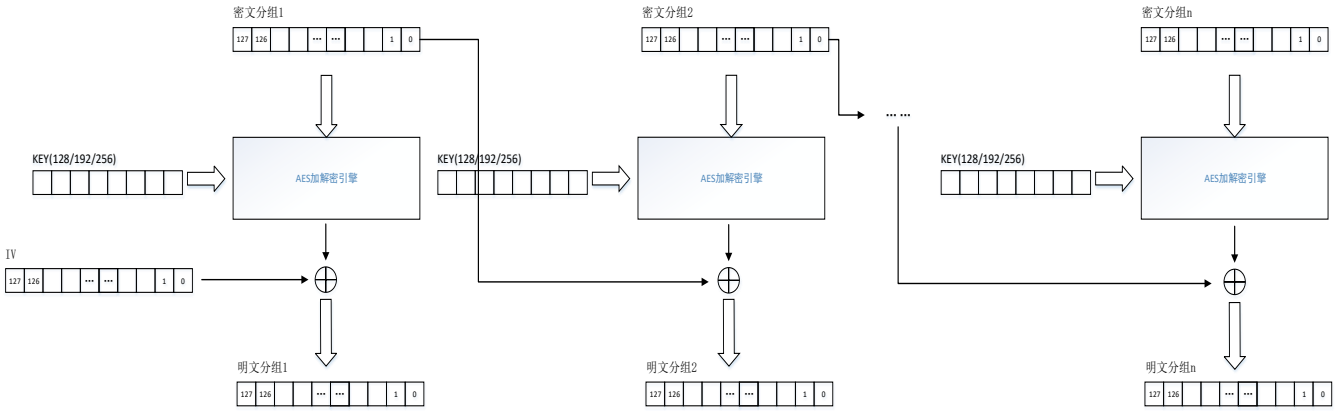
AES-CBC 模式解密流程如下：

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。

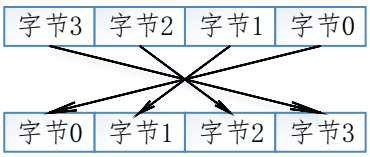
图 35-5 AES-CBC 模式解密流程



35.4. 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。

图 35-6 SWAP 模式



SWAP 模式对密钥、初始向量、输入数据、输出数据同时有效。

35.5. 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};

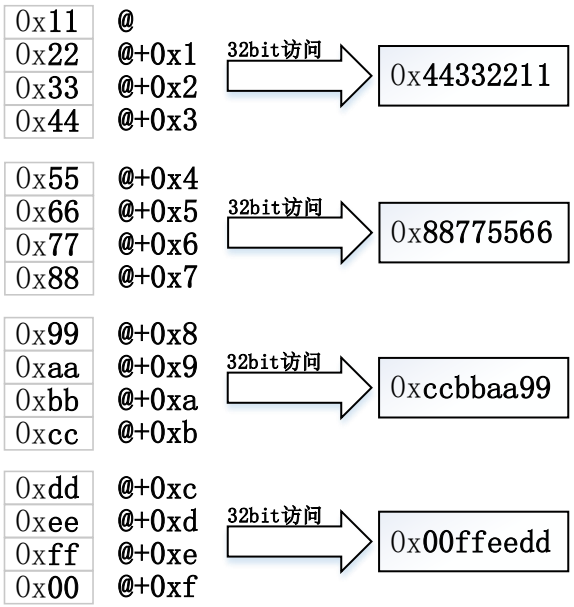
示例：AES块值 0x112233445566778899aabbccddeeff00 32bit输入在系统存储器中表示为		
AES块大小=128位=4*32位	0x11223344	@
0x11223344 55667788 99aabbcc ddeeff00	0x55667788	@+0x4
	0x99aabbcc	@+0x8
	0xddeeff00	@+0xc

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};

示例：AES块值 0x112233445566778899aabbccddeeff00 8bit输入在系统存储器中表示为

AES块大小=128位=16*8位

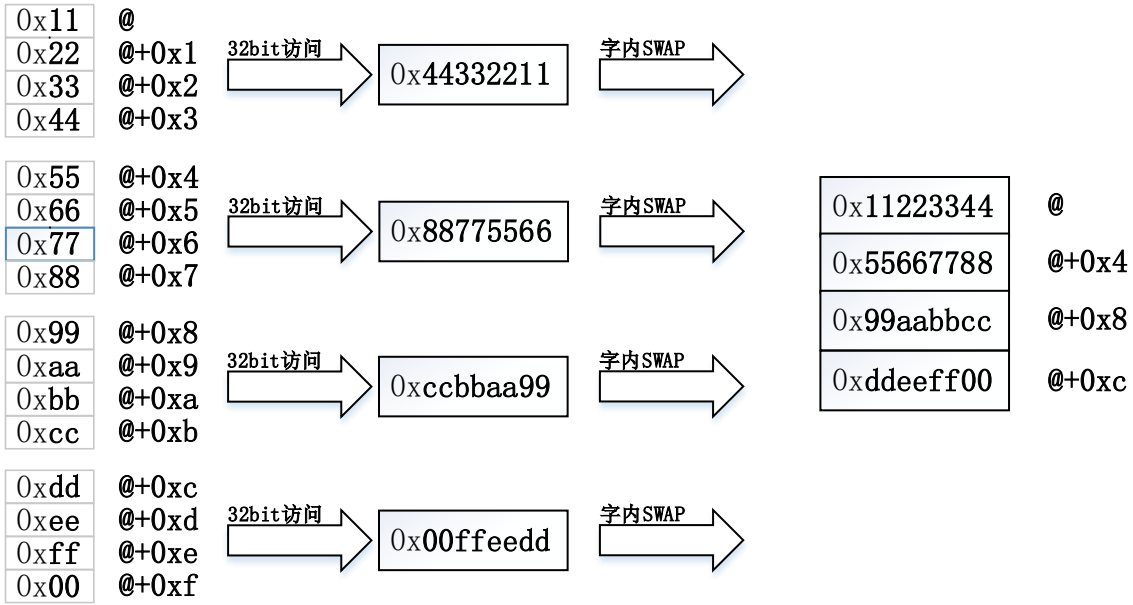


为了获得与 32bit 输入相同的运算结果，应该开启 SWAP 模式。

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。

示例：AES块值 0x112233445566778899aabbccddeeff00 8bit输入在系统存储器中表示为

AES块大小=128位=16*8位



35.6. 数据输出方式

输出数据格式与输入相同。

36. CORDIC 加速算法（CORDIC）

36.1. 概述

CORDIC 模块提供 sin/cos/atan2/sinh/cosh/atanh/ln/sqrt 等数学运算的硬件加速。该模块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，其主要通过 CORDIC（Coordinate Rotation Digital Computer）算法即坐标旋转数字计算方法实现。

CORDIC 算法是 J.D.Volder 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。

36.2. 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

36.3. 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	$\text{atan}\,2(y,x)$	$\sqrt{\text{x}^2+\text{y}^2}$
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 $\exp(\text{x})$ 可以由 $\sinh(x)$ 和 $\cosh(x)$ 的和来获得。以 N 为基数的对数 $\log_{\text{N}}(x)$ 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。

36.4. 数据格式

■ Q31 数据格式(定点表示法)

CORDIC 模块的 Cordic 算法统一采用 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

36.4.1. 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。

36.4.2. 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。

37. 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。

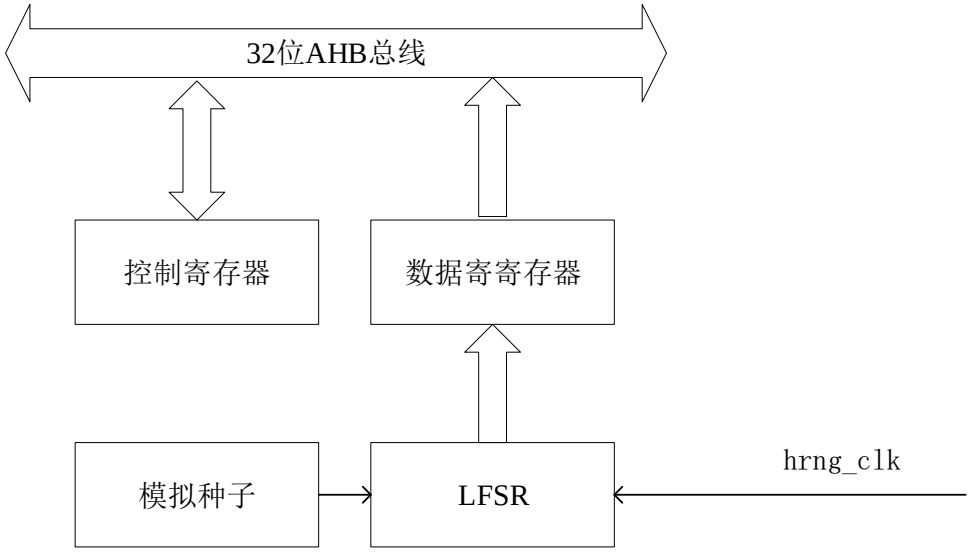
HRNG 可作为一个实时熵源用来构建符合 NIST 要求的确定性随机位发生(DRBG)。

37.1. 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

37.2. 功能说明及框图

图 37-1 HRNG 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。

38. 散列处理器（HASH）

38.1. 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。

38.2. 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

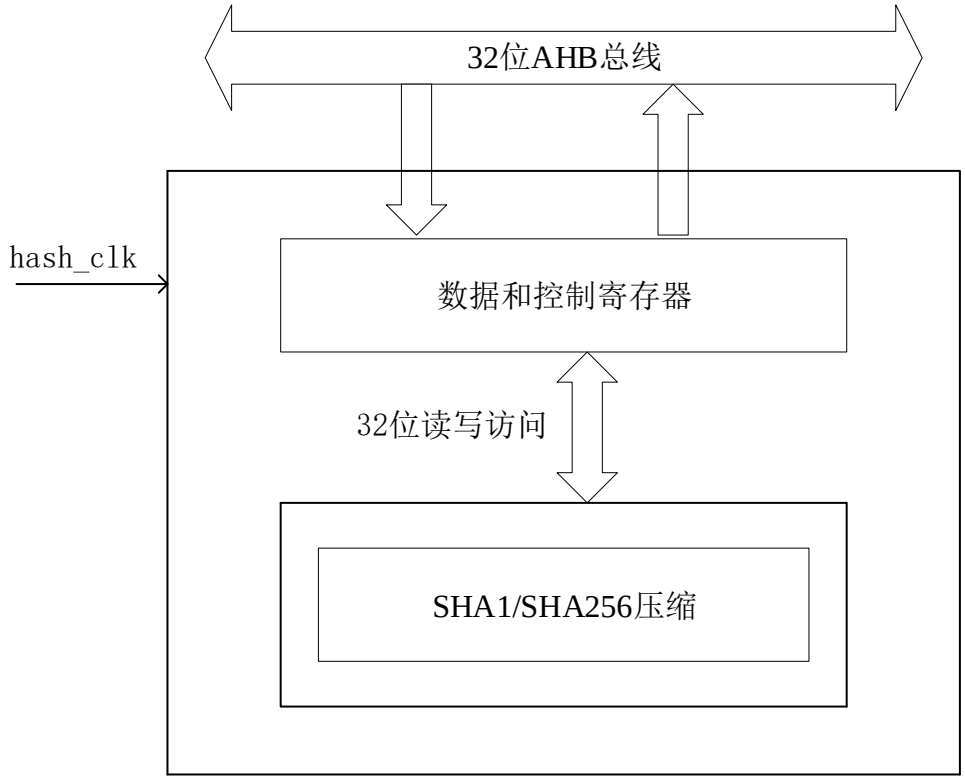
38.2.1. 功能说明及框图

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 38-1 SHA 结构框图



38.2.2. 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 = 0x11223344556677889900;

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

38.2.3. 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

39. 版本历史

版本	日期	作者	描述
V1.0	2022-11-07	Aisinochip	初始化版本
V1.1	2022-12-07	Aisinochip	修改文字描述错误，增加图表说明。
V1.2	2022-04-20	Aisinochip	修改模块寄存器错误。
V1.3	2022-08-07	Aisinochip	COMP3 INP2 改为 PC3。COMP4 INP2 改为 PE9。
V2.0	2024-01-20	Aisinochip	版式及内容较大优化

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@AisinoChip.com

Website： www.AisinoChip.com