PY32F092 系列参考手册

32 位 ARM® Cortex®-M0+ 微控制器

Puya Semiconductor (Shanghai) Co., Ltd.

目录

1. 文档约定

1.1. 寄存器相关缩写词列表

缩写	描述
读/写(RW)	软件可以读写此位
只读(R)	软件只能读取此位
只写(W)	软件只能写入此位，读此位将返回复位值
读取/写入 0 清零(RC_W0)	软件可以读取此位，也可以通过写 0 清除此位，写 1 对此位无影响
读取/写入 1 清零(RC_W1)	软件可以读取此位，也可以通过写 1 清除此位，写 0 对此位无影响
读取/写入清零(RC_W)	软件可以读取此位，也可以通过写入寄存器来清除该位，写入该位的值并不重要
读取/读取清零(RC_R)	软件可以读取此位。读取此位会将其自动清零，写入此位对其值无影响
读取/读取置位(RS_R)	软件可以读取此位。读取此位会自动将其置 1，写入此位对其值无影响
读取/置位(RS)	软件可以读取此位，也可以将其置 1，写 0 对此位无影响
切换(T)	软件可以通过写入 1 来切换此位，写入 0 无影响
保留(Res.)	保留位，必须保持在复位值

1.2. 外设可用性

有关各型号产品的外设可用性以及数量信息，请参见相关器件数据手册。

2. 系统框图

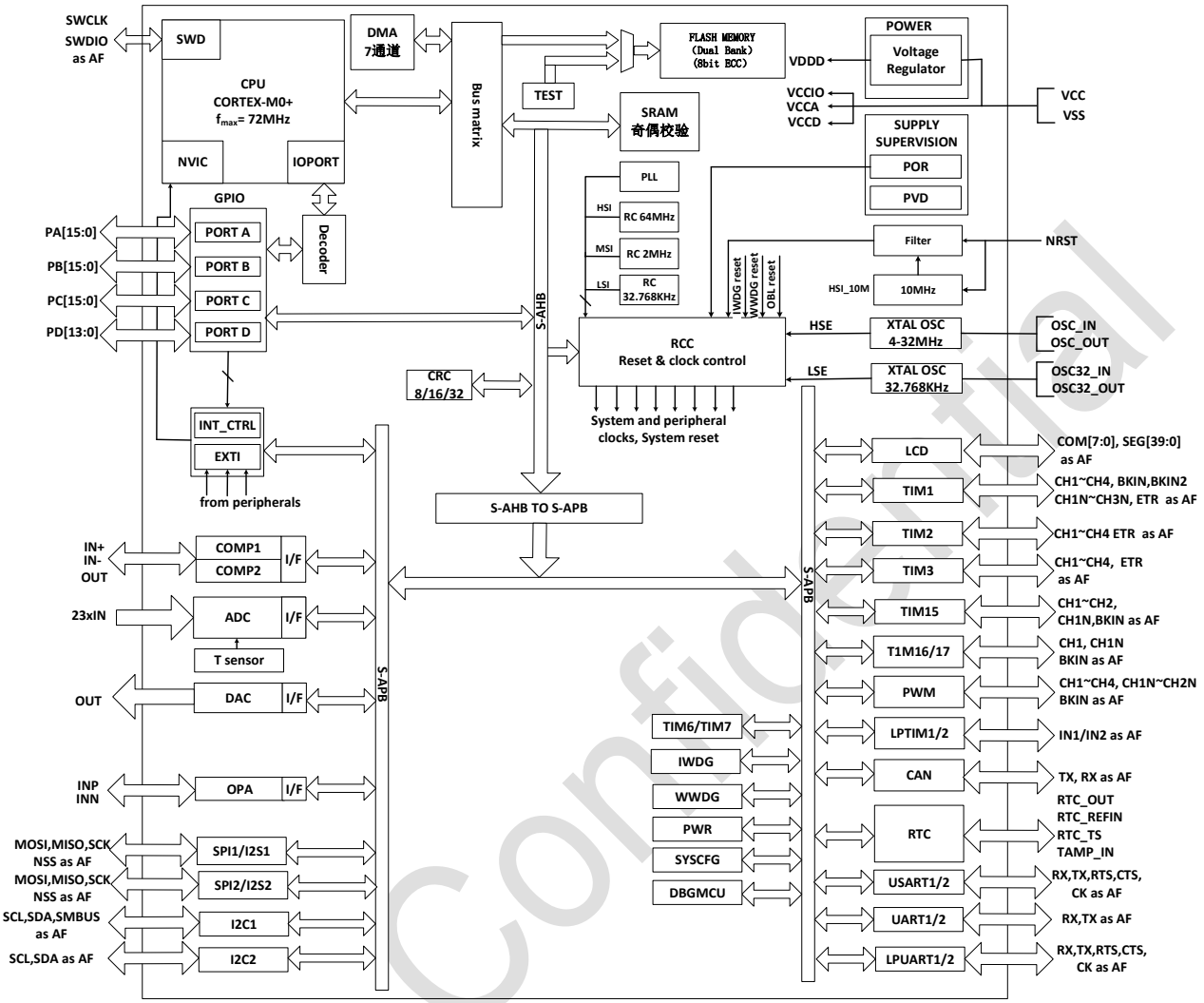


图 2-1 系统框图

3. 存储器和总线架构

3.1. 系统架构

系统由以下部分组成:

- 两个 Master
  - Cortex®-M0+核
  - 通用 DMA
- 三个 Slave
  - 内部 SRAM
  - 内部 Flash
  - 带 AHB-APB 桥的 AHB，用于连接所有 APB 外设

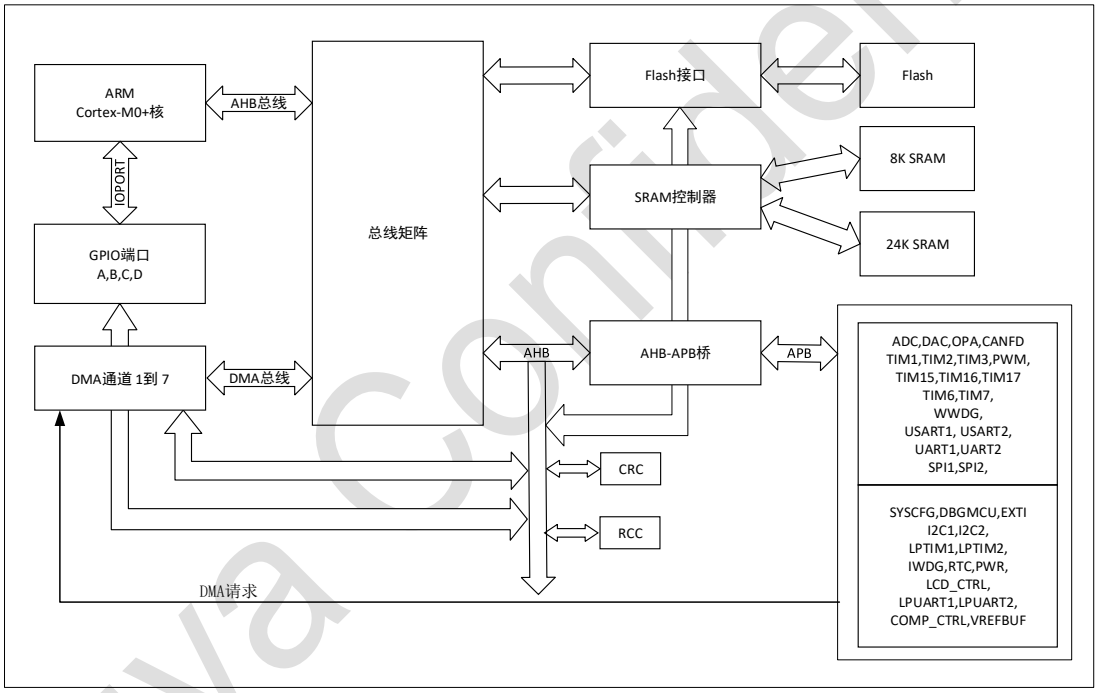


图 3-1 系统总线架构

- AHB 总线

该总线把 Cortex®-M0+内核的系统总线连接到总线矩阵，总线矩阵用来管理 CPU 内核和 DMA 之间的仲裁。

- DMA 总线

该总线把 DMA 的 AHB 主接口连接到总线矩阵，由总线矩阵管理 CPU 和 DMA 对 SRAM、Flash 存储器以及 AHB/APB 外设的访问。

- 总线矩阵

总线矩阵管理 CPU 内核系统总线和 DMA 主控总线之间的仲裁。该仲裁使用循环调度 算法。总线矩阵由主控总线（CPU、DMA）和被控总线（Flash 接口、SRAM 控制器和 AHB- APB 桥）。

- AHB- APB 桥（APB）

AHB-APB 桥可在 AHB 与 APB 总线之间实现完全同步的连接 。

每次芯片复位后，所有外设时钟都被关闭（SYSCFG，SRAM 和 Flash 接口除外）。使用外设前，必须在 RCC_AHBENR 或 RCC_APBENRx（x=1，2） 寄存器中使能其时钟。

有关 AHB 和 APB 外设地址映射信息，参考“寄存器映射”章节。

3.2. 存储器构成

3.2.1. 简介

程序存储器、数据存储器、寄存器和 IO 端口被统一编址在一个线性 4 GB 空间。该地址以小端编码形式存在（一个 word 中，最低字节分配在最低地址）。

整个寻址空间被划分成 8 个 512 MB 的块区域。

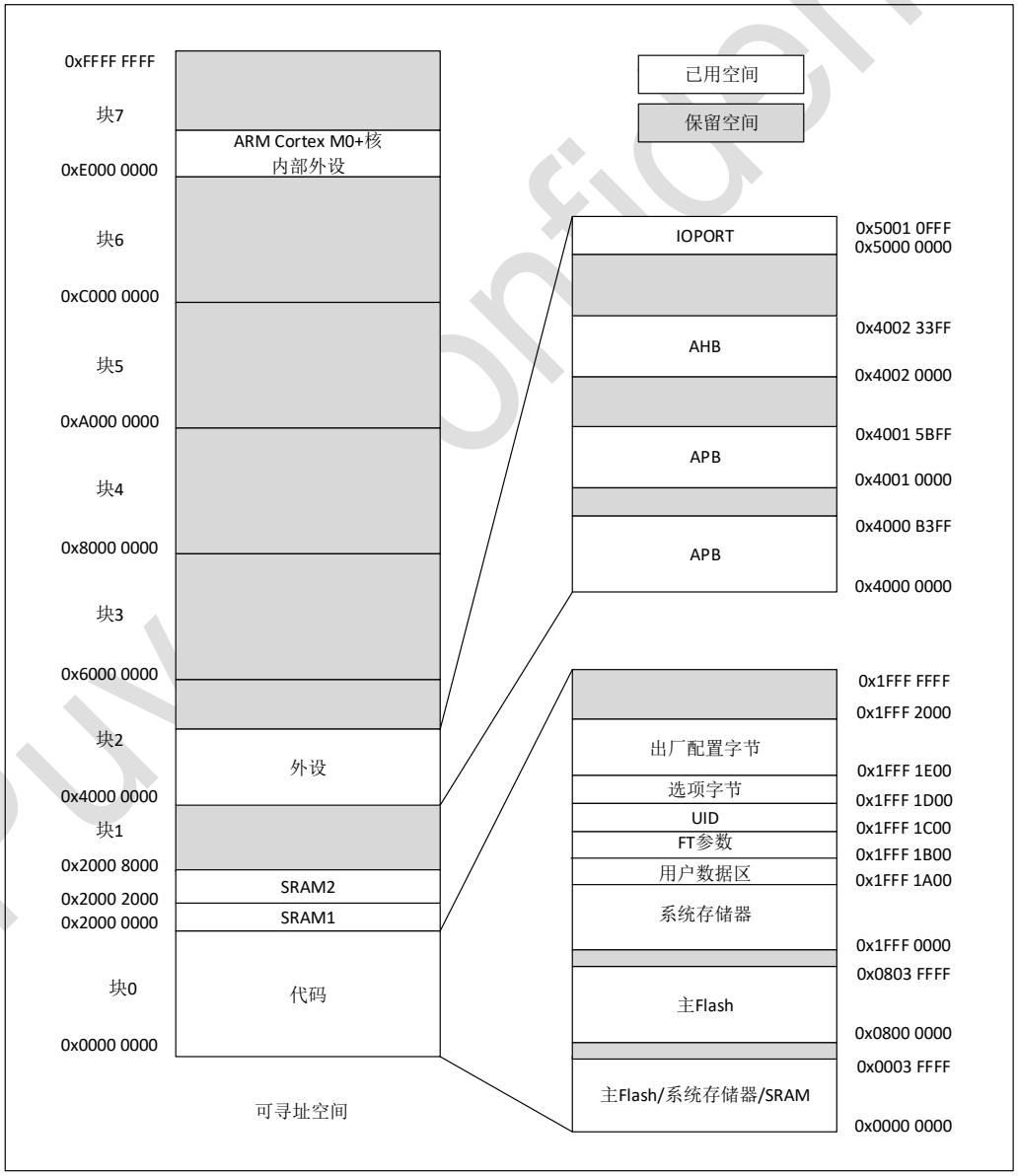


图 3-2 存储器映射

3.2.2. 存储器映射

表 3-1 存储器地址

类型	边界地址	大小	存储区	描述
SRAM	0x2000 8000-0x3FFF FFFF	~511 MB	保留	-
	0x2000 2000-0x2000 7FFF	24 KB	SRAM2	参见 3.3“嵌入式 SRAM”
	0x2000 0000-0x2000 1FFF	8 KB	SRAM1	
代码	0x1FFF 2000-0x1FFF FFFF	56 KB	保留	-
	0x1FFF 1E00-0x1FFF 1FFF	512 bytes	出厂配置字节	-
	0x1FFF 1D00-0x1FFF 1DFF	256 bytes	选项字节	参见 4.4“Flash 选项字节”
	0x1FFF 1C00-0x1FFF 1CFF	256 bytes	UID	部分内容参见 4.3“产品唯一身份标识码（UID）”
	0x1FFF 1B00-0x1FFF 1BFF	256 bytes	FT 参数	-
	0x1FFF 1A00-0x1FFF 1AFF	256 bytes	用户数据字节	参见 4.7“Flash 用户数据字节”
	0x1FFF 0000-0x1FFF 19FF	6.5 KB	系统存储器	-
	0x0804 0000-0x1FFE FFFF	~393 MB	保留	-
	0x0800 0000-0x0803 FFFF	256 KB	主 Flash	-
	0x0004 0000-0x07FF FFFF	~8 MB	保留	-
	0x0000 0000-0x0003 FFFF	256 KB	根据 Boot 配置选择，是 1）主 Flash 2）系统存储器 3）SRAM	-

3.2.3. 寄存器映射

表 3-2 外设寄存器地址

总线	边界地址	大小	外设
	0xE000 0000-0xE00F FFFF	1 MB	Cortex®-M0+核内部外设
IOPORT	0x5000 1800-0x5FFF FFFF	~256 MB	保留
	0x5000 1400-0x5000 17FF	1 KB	保留
	0x5000 1000-0x5000 13FF	1 KB	保留
	0x5000 0C00-0x5000 0FFF	1 KB	GPIOD
	0x5000 0800-0x5000 0BFF	1 KB	GPIOC
	0x5000 0400-0x5000 07FF	1 KB	GPIOB
	0x5000 0000-0x5000 03FF	1 KB	GPIOA
AHB	0x4002 3400-0x4FFF FFFF	~	保留
	0x4002 3000-0x4002 33FF	1 KB	CRC
	0x4002 2400-0x4002 2FFF	~	保留
	0x4002 2000-0x4002 23FF	1 KB	Flash
	0x4002 1800-0x4002 1FFF	2 KB	保留
	0x4002 1400-0x4002 17FF	1 KB	保留
	0x4002 1000-0x4002 13FF	1 KB	RCC

总线	边界地址	大小	外设
	0x4002 0400-0x4002 0FFF	1 KB	保留
	0x4002 0000-0x4002 03FF	1 KB	DMA
APB	0x4001 5C00-0x4001 FFFF	32 KB	保留
	0x4001 5800-0x4001 5BFF	1 KB	MCUDBG
	0x4001 5000-0x4001 57FF	2 KB	保留
	0x4001 4C00-0x4001 4FFF	1 KB	PWM
	0x4001 4800-0x4001 4BFF	1 KB	TIM17
	0x4001 4400-0x4001 47FF	1 KB	TIM16
	0x4001 4000-0x4001 43FF	1 KB	TIM15
	0x4001 3C00-0x4001 3FFF	1 KB	保留
	0x4001 3800-0x4001 3BFF	1 KB	USART1
	0x4001 3400-0x4001 37FF	1 KB	保留
	0x4001 3000-0x4001 33FF	1 KB	SPI1
	0x4001 2C00-0x4001 2FFF	1 KB	TIM1
	0x4001 2800-0x4001 2BFF	1 KB	保留
	0x4001 2400-0x4001 27FF	1 KB	ADC
	0x4001 0C00-0x4001 23FF	6 KB	保留
	0x4001 0800-0x4001 0BFF	1KB	V_{REFBUF}
	0x4001 0400-0x4001 07FF	1 KB	EXTI
	0x4001 0300-0x4001 03FF	1 KB	OPA
	0x4001 0200-0x4001 02FF		COMP1/COMP2
	0x4001 0000-0x4001 01FF		SYSCFG
	0x4000 B400-0x4000 FFFF	19 KB	保留
	0x4000 B000-0x4000 B3FF	1 KB	BKP(TAMP)
	0x4000 9C00-0x4000 AFFF	5 KB	保留
	0x4000 9800-0x4000 9BFF	1 KB	LPUART2
	0x4000 9400-0x4000 97FF	1 KB	LPTIM2
	0x4000 8400-0x4000 93FF	4 KB	保留
	0x4000 8000-0x4000 83FF	1KB	LPUART1
	0x4000 7C00-0x4000 7FFF	1 KB	LPTIM1
	0x4000 7800-0x4000 7BFF	1 KB	保留
	0x4000 7400-0x4000 77FF	1 KB	DAC
	0x4000 7000-0x4000 73FF	1 KB	PWR
	0x4000 6C00-0x4000 6FFF	1 KB	保留
	0x4000 6800-0x4000 6BFF	1 KB	CANMEM
	0x4000 6400-0x4000 67FF	1 KB	CAN
	0x4000 5C00-0x4000 63FF	2 KB	保留
	0x4000 5800-0x4000 5BFF	1 KB	I^2C2
	0x4000 5400-0x4000 57FF	1 KB	I^2C1
	0x4000 5000-0x4000 53FF	1 KB	保留

总线	边界地址	大小	外设
	0x4000 4C00-0x4000 4FFF	1KB	UART2
	0x4000 4800-0x4000 4BFF	1KB	UART1
	0x4000 4400-0x4000 47FF	1 KB	USART2
	0x4000 3C00-0x4000 43FF	2 KB	保留
	0x4000 3800-0x4000 3BFF	1 KB	SPI2
	0x4000 3400-0x4000 37FF	1 KB	保留
	0x4000 3000-0x4000 33FF	1 KB	IWDG
	0x4000 2C00-0x4000 2FFF	1 KB	WWDG
	0x4000 2800-0x4000 2BFF	1 KB	RTC
	0x4000 2400-0x4000 27FF	1 KB	LCD
	0x4000 1800-0x4000 23FF	3 KB	保留
	0x4000 1400 - 0x4000 17FF	1 KB	TIM7
	0x4000 1000 - 0x4000 13FF	1 KB	TIM6
	0x4000 0800-0x4000 0FFF	2 KB	保留
	0x4000 0400 - 0x4000 07FF	1 KB	TIM3
	0x4000 0000 - 0x4000 03FF	1 KB	TIM2

3.3. 嵌入式 SRAM

片内集成 32 KB SRAM，分 8 KB SRAM1 和 24 KB SRAM2，每个字节包含 1 位奇偶校验。通过字节（9 bits）、半字（18 bits）或者全字（36 bits）的方式可访问 SRAM。

注意：上电后，硬件不会对 SRAM 做复位操作。

3.3.1. SRAM 奇偶校验

IEC60730 标准要求提高 SRAM 存储器可靠性。因此，本产品的 SRAM 每 8 位附加 1 位奇偶校验位，数据宽度为 36 位。

用户可以使用 Flash 选项字节页中的选项位 SRAM_PE 启用奇偶校验。

SRAM 奇偶校验错误检测功能在写数据时附加奇偶校验位，而在读数据时检查奇偶校验位，并且能在发生奇偶校验错误时根据 SYSCFG 中 SRAM_SCSR.PERR_RSTEN 的配置产生复位或者 NMI 中断。同时该错误也可以连接到定时器的系统刹车输入。

3.3.2. SRAM 写保护

SRAM 可以采用 1K 字节的页面粒度进行写保护。

写保护可以在 SYSCFG 的 SRAM_WPR 写保护寄存器中启用。这是一个具有一次性写入“1”机制的寄存器，这意味着通过一次性写入“1”为 SRAM 的该页设置了“写入保护”，并且只能通过系统复位来清除本设置。

表 3-3 SRAM 写保护地址

Page number	Feild	Start address	End address
Page 0	SRAM1	0x0000 0000 / 0x2000 0000	0x0000 03FF / 0x2000 03FF

Page number	Feild	Start address	End address
Page 1		0x0000 0400 / 0x2000 0400	0x0000 07FF / 0x2000 07FF
Page 2		0x0000 0800 / 0x2000 0800	0x0000 0BFF / 0x2000 0BFF
Page 3		0x0000 0C00 / 0x2000 0C00	0x0000 0FFF / 0x2000 0FFF
Page 4		0x0000 1000 / 0x2000 1000	0x0000 13FF / 0x2000 13FF
Page 5		0x0000 1400 / 0x2000 1400	0x0000 17FF / 0x2000 17FF
Page 6		0x0000 1800 / 0x2000 1800	0x0000 1BFF / 0x2000 1BFF
Page 7		0x0000 1C00 / 0x2000 1C00	0x0000 1FFF / 0x2000 1FFF
Page 8	SRAM2	0x0000 2000 / 0x2000 2000	0x0000 23FF / 0x2000 23FF
Page 9		0x0000 2400 / 0x2000 2400	0x0000 27FF / 0x2000 27FF
Page 10		0x0000 2800 / 0x2000 2800	0x0000 2BFF / 0x2000 2BFF
Page 11		0x0000 2C00 / 0x2000 2C00	0x0000 2FFF / 0x2000 2FFF
Page 12		0x0000 3000 / 0x2000 3000	0x0000 33FF / 0x2000 33FF
Page 13		0x0000 3400 / 0x2000 3400	0x0000 37FF / 0x2000 37FF
Page 14		0x0000 3800 / 0x2000 3800	0x0000 3BFF / 0x2000 3BFF
Page 15		0x0000 3C00 / 0x2000 3C00	0x0000 3FFF / 0x2000 3FFF
Page 16		0x0000 4000 / 0x2000 4000	0x0000 43FF / 0x2000 43FF
Page 17		0x0000 4400 / 0x2000 4400	0x0000 47FF / 0x2000 47FF
Page 18		0x0000 4800 / 0x2000 4800	0x0000 4BFF / 0x2000 4BFF
Page 19		0x0000 4C00 / 0x2000 4C00	0x0000 4FFF / 0x2000 4FFF
Page 20		0x0000 5000 / 0x2000 5000	0x0000 53FF / 0x2000 53FF
Page 21		0x0000 5400 / 0x2000 5400	0x0000 57FF / 0x2000 57FF
Page 22		0x0000 5800 / 0x2000 5800	0x0000 5BFF / 0x2000 5BFF
Page 23		0x0000 5C00 / 0x2000 5C00	0x0000 5FFF / 0x2000 5FFF
Page 24		0x0000 6000 / 0x2000 6000	0x0000 63FF / 0x2000 63FF
Page 25		0x0000 6400 / 0x2000 6400	0x0000 67FF / 0x2000 67FF
Page 26		0x0000 6800 / 0x2000 6800	0x0000 6BFF / 0x2000 6BFF
Page 27		0x0000 6C00 / 0x2000 6C00	0x0000 6FFF / 0x2000 6FFF
Page 28		0x0000 7000 / 0x2000 7000	0x0000 73FF / 0x2000 73FF
Page 29		0x0000 7400 / 0x2000 7400	0x0000 77FF / 0x2000 77FF
Page 30		0x0000 7800 / 0x2000 7800	0x0000 7BFF / 0x2000 7BFF
Page 31		0x0000 7C00 / 0x2000 7C00	0x0000 7FFF / 0x2000 7FFF

3.4. Flash 概述

Flash 存储器有两个不同的物理区域组成：

- 主存储区，256 KB，分为 2 个 Bank。可根据不同产品配置为 256 KB/192 KB/128 KB/64 KB，用于存储用户程序和用户数据。当设定为小于 256 KB 容量时，软件对设定范围外空间的访问会产生 HardFault。
- 信息区，8 KB，它包括以下部分：
  - 出厂配置（Factory config）：512 bytes，用于存放芯片配置信息。
  - UID：256 bytes，用于存放芯片的 UID
  - 选项字节（Option bytes）：256 bytes，用于存放芯片硬件和存储保护的配置值
  - FT 信息区：256 bytes，用于存放 FT 参数
  - 用户数据字节（User data bytes）：256 bytes，用户数据区
  - 系统存储器（System memory）：6.5 KB，用于存放系统自举程序
  - Flash 接口实现基于 AHB 协议的指令读取和数据访问，它也通过寄存器实现了 Flash 的基本擦写/编程等操作。

3.5. 启动模式

通过 BOOT0 引脚、用户选项字节中的自举配置位 nBOOT1 和强制从 Flash 自举配置位 BOOT_LOCK，可选择如下几种不同的启动模式，如下表所示：

表 3-4 Boot 配置

BOOT_LOCK	自举模式配置		自举模式
	nBOOT1	BOOT0 引脚	
1	X	X	强制从 Flash 主存储区启动
0	X	0	选择 Flash 主存储区作为启动区
0	1	1	选择 Flash 系统存储区作为启动区
0	0	1	选择 SRAM 作为启动区

用户按照上表决定选择哪种启动模式。

复位后，CPU 从地址 0x0000 0000 取堆栈顶的值，然后从启动存储器的 0x0000 0004 地址开始执行指令。取决于被选择的启动模式，Flash 主存储区、系统存储器或者 SRAM 按照如下进行访问：

- 从 Flash 主存储区自举：Flash 主存储区跟启动存储器空间的 0x0000 0000 对齐，但是仍然可以按其本来的存储器空间（0x0800 0000）进行访问。也就是说，Flash 空间可以从地址 0x0000 0000 或者 0x0800 0000 访问到。
- 从 Flash 系统存储区自举：Flash 系统存储区对齐在启动存储器空间 0x0000 0000，但是仍然可以从它本来的地址空间 0x1FFF 0000 访问到。
- 从 SRAM 存储区自举：SRAM 对齐在启动存储器空间的 0x0000 0000，但是仍然可以通过 0x2000 0000 地址访问到。
- 强制从用户 Flash 自举：无论其他自举模式配置位如何，都能够使用 BOOT_LOCK 位强制从主 Flash 中的唯一入口点进行自举。

3.5.1. 物理映射

选择了自举模式后，应用软件仍然可以修改在代码区域使用的存储器。这个修改是通过对 SYSCFG_CFGR1 寄存器的 MEM_MODE 位进行编程来实现的。

3.5.2. 嵌入式自举程序

嵌入式自举程序在芯片生产阶段被写入，并存放在 Flash 系统存储区中。它用来使用下面串行接口进行对 Flash 存储器的再次写入：

- USART1，对应 PB11/PB12 或者 PC8/PC9 或者 PD5/PD6

4. 嵌入式 Flash

4.1. Flash 简介

Flash 接口可管理 CPU 对 Flash 进行的访问。该接口可针对 Flash 执行擦除和编程操作、读写保护和安全机制。

Flash 接口通过指令预取和缓存机制加速代码执行。

4.2. Flash 主要特性

- 主存储区：256 KB（双 bank，每个大小 16K x 64bit），用于存储用户程序和用户数据。
- 信息区：8 KB（1K x 64bits）
- 页（page）大小：256 bytes（32 x 64bits)
- 扇区（sector）大小: 8 KB（1K x 64bits)
- Flash 仅支持写 0 不支持写 1，擦除后的值为 1
- Flash 编程/擦除操作的最小单位是页

Flash 存储区控制接口电路的主要特性如下：

- Flash 读操作，读宽度为 72 位（64 位 + 8 个 ECC 位）
- Flash 编程/擦除操作，写宽度为 72 位（64 位 + 8 个 ECC 位）
- 由选项字节（RDP）激活的读保护
- 由选项字节（WRP)选择的写保护区域
- 由选项字节（PCROP)选择的专有代码读保护区域
- 误码校正（ECC)：每 64 位对应 8 位 ECC
- 支持：RWW（read while write）
  - 支持写 Bank1 时读 Bank0，以及写 Bank0 时读 Bank1
  - 不支持同时读 2 个 Bank
  - 不支持同时写 2 个 Bank
- 预取缓冲器
- 选项字节加载器

4.3. Flash 功能描述

4.3.1. Flash 构成

Flash 存储器由 64+8bit 宽的 cell 组成，具有支持读写功能（RWW）的双 Bank 架构，可以用作程序和数据的存储，页大小为 256 bytes，扇区大小为 8 KB。

从功能上，Flash 存储器分为主存储区和信息区，前者容量是 256 KB，后者容量为 8 KB。

页擦操作可以应用于主存储区和信息区，扇区擦不能应用于信息区。

未开启写保护时，Flash 主存储区可分别对 Bank0 和 Bank1 做片擦；Flash 信息区无论是否开启写保护，都不支持片擦。

表 4-1 Flash 存储区构成

区域	空间			地址	大小
主存储区	Bank0	扇区 0	页 0-31	0x0800 0000-0x0800 1FFF	8 KB
		扇区 1	页 32-63	0x0800 2000-0x0800 3FFF	8 KB
		扇区 2	页 64-95	0x0800 4000-0x0800 5FFF	8 KB
		…	…	…	…
		扇区 14	页 448-479	0x0801 C000-0x0801 DFFF	8 KB
		扇区 15	页 480-511	0x0801 E000-0x0801 FFFF	8 KB
	Bank1	扇区 0	页 0-31	0x0802 0000-0x0802 1FFF	8 KB
		扇区 1	页 32-63	0x0802 2000-0x0802 3FFF	8 KB
		扇区 2	页 64-95	0x0802 4000-0x0802 5FFF	8 KB
		…	…	…	…
		扇区 14	页 448-479	0x0803 C000-0x0803 DFFF	8 KB
		扇区 15	页 480-511	0x0803 E000-0x0803 FFFF	8 KB
系统存储区	Bank0	扇区 0	页 0-25	0x1FFF 0000-0x1FFF 19FF	6.5 KB
用户数据区（OTP）			页 26	0x1FFF 1A00-0x1FFF 1AFF	256 bytes
FT			页 27	0x1FFF 1B00-0x1FFF 1BFF	256 bytes
UID			页 28	0x1FFF 1C00-0x1FFF 1CFF	256 bytes
选项字节			页 29	0x1FFF 1D00-0x1FFF 1DFF	256 bytes
出厂配置 0			页 30	0x1FFF 1E00-0x1FFF 1EFF	256 bytes
出厂配置 1			页 31	0x1FFF 1F00-0x1FFF 1FFF	256 bytes

4.3.2. Flash ECC

Flash 中的数据宽度为 72 位：每 64bit 增加 8 位 ECC 位。

ECC 机制支持：

- 一位错误检测和纠正
- 两位错误检测

当检测和纠正一个错误时，在 FLASH_ECCR 寄存器中设置标志 ECCC（ECC 纠正）。如果设置了 ECCCIE，则会生成一个中断。

当检测到两个错误时，在 FLASH_ECCR 寄存器中设置标志 ECCD（ECC 检测）。在这种情况下，会生成一个 NMI 中断。

当检测到 ECC 错误时，故障地址及其相关的状态被保存在 FLASH_ECCR 寄存器中的 ADDR_ECC，SYSF_ECC 和 BK_ECC 中。

当设置了 ECCC 或 ECCD 时，如果出现新的 ECC 错误，则不更新 ADDR_ECC，SYSF_ECC 和 BK_ECC。仅当 ECC 标志清零后， FLASH_ECCR 寄存器才会更新。

4.3.3. Flash 读访问延迟

Flash 可以作为一个通用的存储器空间，被直接寻址访问。通过专门的读控制时序，可以对 Flash 存储器的内容进行读取。

取址和数据访问都是通过 AHB 总线进行的。读操作可以被 FLASH_ACR 寄存器的 LATENCY 位控制，即读取 Flash 增加一个或者不增加等待状态。

FLASH_ACR.LATENCY 位，当为 0，则不增加 flash 读操作的等待状态；当为 1，Flash 读操作增加 1 个等待状态；当为 2，Flash 读操作增加 2 个等待状态。该机制是为了匹配高速的系统时钟和相对低速的 Flash 读取速度，而进行的专门设计。

4.3.4. Flash 加速

4.3.4.1. 指令预取

每个 Flash 读操作可读取 64 位，可以是 2 行 32 位指令，也可以是 4 行 16 位指令，具体取决于烧写在 Flash 中的程序。这一 64 位当前指令行保存在当前缓冲区中。因此，对于顺序执行的代码，至少需要 2 个 CPU 周期来执行前一次读取的指令行。在 CPU 请求当前指令行时，可使用 CPU 系统总线的预取操作读取 Flash 中的下一个连续存放的指令行。

可将 Flash 访问控制寄存器(FLASH_ACR)中的 PRFTEN 位置 1，来使能预取功能。当访问 Flash 至少需要一个等待状态时，此功能非常有用。处理非顺序执行的代码（有分支）时，指令可能并不存在于当前使用的或预取的指令行中。这种情况下，CPU 等待时间至少等于等待状态数。

如果当前缓冲区中存在循环，则不会执行新的访问。

4.3.4.2. 缓存存储器

为了减少因指令跳转而损耗的时间，可将 2 行 64 位的指令（16 字节）保存到指令缓存存储器中。可将 FLASH 访问控制寄存器（FLASH_ACR)中的指令缓存使能（ICEN)位置 1，来使能这一特性。每当出现指令缺失（即请求的指令未存在于当前使用的指令行、预取指令行或指令缓存存储器中）时，系统会将新读取的行复制到指令缓存存储器中。如果 CPU 请求的指令已存在于指令缓存存储器中，则无需任何延时即可立即获取。指令缓存存储器存满后，可采用 LRU（最近最少使用）策略确定指令缓存存储器中待替换的指令行。此特性非常适用于包含循环的代码。

系统复位后指令缓存存储器使能。在 CPU 流水线执行阶段，将通过系统总线访问 Flash 中的 CPU 数据缓冲池。每条系统总线的读访问都将访问保存在当前缓冲区中的 64 位数据。因此，直到提供了请求的数据后，CPU 流水线才会继续执行。

4.3.5. Flash 编程和擦除操作

通过在线编程 ICP（In-circuit programming）或者在应用中编程 IAP（In-application programming）可以对 Flash 进行编程操作。

ICP：用来更新整个 Flash 存储器的内容，可以使用 SWD 协议或者系统自举程序支持的接口，把用户应用程序加载到 MCU 中。ICP 提供了快速和高效的设计迭代，并消除了不必要的器件封装处理或者插接。

IAP：可以使用芯片支持的通讯接口，下载要编程的数据到 Flash 中。IAP 允许用户在运行应用程序时重新编程 Flash 存储器。但是，部分应用程序必须事先通过 ICP 方式编程到 Flash。

如果在进行 Flash 编程或者擦写操作时，发生了复位，则 Flash 存储器的内容是不被保证的。

在对一个 BANK 区域进行编程或者擦写操作期间，DMA 或者 CPU 可以对另一个 BANK 进行读访问。当正在进行编程或者擦写操作时，不能对该 BANK 进行代码和数据的读取。

注意：对 Flash 编程和擦除操作，必须打开 HSI（硬件会根据 HSI 频率自动调整擦写参数）。

4.3.5.1. 解锁 Flash 操作

在复位后，Flash 存储器会被保护，防止不想要的（比如电干扰引起的）编程和擦除操作进行。复位后，写 FLASH_CR 寄存器是不被允许的（除了用作加载选项字节的 OBL_LAUNCH 位）。每次对 Flash 的擦除和编程操作，都必须通过写 FLASH_KEYR 寄存器，产生解锁时序，启用 FLASH_CR 寄存器的访问。

具体步骤如下：

步骤 1：向 FLASH_KEYR 寄存器写入 KEY1=0x4567 0123

步骤 2：向 FLASH_KEYR 寄存器写入 KEY2=0xCDEF 89AB

任何错误的时序都会锁定 FLASH_CR 寄存器，直到下一次系统复位。如果密钥操作顺序不正确，总线错误产生，并产生 HardFault 中断。这样的错误包括第一个写周期的 KEY1 不匹配，或者 KEY1 匹配，但第二个写周期的 KEY2 不匹配。

FLASH_CR 寄存器可以通过软件写 FLASH_CR 寄存器的 LOCK 位被再次锁住。

另外，当 FLASH_SR 寄存器的 BSY 位被置 1 时，FLASH_CR 寄存器不能写入。此时，任何尝试进行写该寄存器（FLASH_CR）的操作会引起 AHB 总线阻塞，直到 BSY 位被清零。

4.3.5.2. Flash 擦除操作

Flash 存储器可以按照页进行擦除操作，或者进行扇区和 bank 的片擦（扇区和 bank0 片擦对 Flash 信息区不起作用）。

- 页擦除

当某个页被 WRP 保护，它是不会被擦除的，此时 WRPERR 位被置位。当要进行页擦除操作时，要进行以下步骤：

1. 检查 FLASH_SR 寄存器 BSY 位，确认没有正在进行的 Flash 操作
2. 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护
3. 置位 FLASH_CR 寄存器的 PER 位和 EOPIE（如果需要产生 EOP 中断）位
4. 向该页写任意数据（必须 32bit 数据）
5. 等待 BSY 位被清零
6. 检查 EOP 标志位被置位
7. 清零 EOP 标志

- 扇区擦除

扇区擦除用来对 8 KB 的 Flash 主存储区进行擦除操作，但对信息区不起作用。另外，当某个扇区被 WRP 保护，它是不会被擦除的，此时 WRPERR 位被置位。

进行扇区擦除的步骤如下：

1. 检查 BSY 位，确认是否没有正在进行的 Flash 操作
2. 向 FLASH_KEYR 寄存器依次写 KEY1、KEY2，解除 FLASH_CR 寄存器保护
3. 置位 FLASH_CR 寄存器的 SER 位和 EOPIE（如果需要产生 EOP 中断）位
4. 向该扇区写任意数据（必须 32bit 数据）
5. 等待 BSY 位被清零
6. 检查 EOP 标志位被置位
7. 清零 EOP 标志

- Bank 片擦

Bank 片擦除用来对 Flash 主存储区 128 KB bank 区域进行擦除操作，但对信息区不起作用。另外，当 WRP 被使能，片擦除功能无效，不会产生片擦除操作，并且 WEPERR 位被置位。

进行片擦除的步骤如下：

1. 检查 BSY 位，确认是否没有正在进行的 Flash 操作
2. 向 FLASH_KEYR 寄存器依次写 KEY1，KEY2，解除 FLASH_CR 寄存器保护
3. 置位 FLASH_CR 寄存器的 MER0 或 MER1 位和 EOPIE（如果需要产生 EOP 中断）位
4. 向 Flash 主存储区对应 Bank 的任意空间写任意数据
5. 等待 BSY 位被清零
6. 检查 EOP 标志位被置位
7. 清零 EOP 标志

4.3.5.3. Flash 主存储区编程操作

Flash 编程操作之前需要先做擦除操作。擦除操作步骤参见“Flash 擦除操作”小节描述。

Flash 存储器每次以 64bit 双字为单位进行整个页的编程操作。

当 FLASH_CR 寄存器的 PG 位被置位，CPU 向 FLASH 存储器地址空间写两次 32bit 数据，硬件整合为 64bit 时，编程操作开始启动。编程操作结束，FLASH_CR 寄存器的 EOP 位会被置位。

注意：CPU 必须以字（word）为单位操作，进行半字（half-word）或者字节（byte）操作会产生 HardFault！

如果要编程的 Flash 地址空间，是被 FLASH_WRPR 寄存器设置为保护的区域，则编程操作会被忽略掉，同时 FLASH_SR 寄存器 WRPERR 位会被置位。

具体 Flash 编程的操作步骤如下所示：

1. 检查 FLASH_SR 寄存器的 BSY 位，判断是否当前没有正在继续的 Flash 操作
2. 如果没有正在进行的 Flash 擦除或者编程操作，则软件读出该 Page 的 64 个 word（如果该页已有数据存放且需要更新部分数据时，则进行该步骤，否则跳过该步骤）
3. 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护
4. 置位 FLASH_CR 寄存器的 PG 位和 EOPIE（如果需要产生 EOP 中断）位；（如有必要，先清零 FLASH_CR 寄存器的 PGSTRT 位）

5. 向目标地址进行第 1 到第 63 个 word 的编程操作（只接受 32bit 的编程）
6. 置位 FLASH_CR 寄存器的 PGSTRT
7. 写第 64 个 word
8. 等待 FLASH_SR 寄存器的 BSY 位被清零
9. 检查 FLASH_SR 寄存器的 EOP 标志位（当编程操作已经成功，该位被置位），然后软件清零该位
10. 如果不再有编程操作，则软件清除 PG 位

当上述步骤 7 成功执行，则编程操作自动启动，同时 BSY 位被硬件置位。

4.3.5.4. Flash 擦写时间配置

Flash 的编程和擦除的时间需要进行严谨的控制，否则会造成操作失败。上电默认情况，硬件设计将编程和擦除操作的时间参数，设置成 HSI 为 2 MHz 的参数。当 HSI 频率变化时，硬件会通过分频将擦写时间逻辑使用的时钟始终保持为 2 MHz。

4.4. 产品唯一身份标识码（UID）

唯一身份标识码典型应用场景：

- 用作序列号
- 对内部闪存编程时，将其用作密钥或加密原语以提高代码的安全性
- 激活安全自举过程等

产品唯一身份标识提供了一个对于任何设备都唯一的参考号码。

用户永远不能改变这些位。唯一身份标识符也可以以单字节/半字/字等不同方式进行读取，然后使用自定义的算法连接起来。

本产品 UID 内容如下表所示：

基址：0x1FFF 1C00

表 4-2 UID 标识码

偏移地址	描述	UID Bits							
		7	6	5	4	3	2	1	0
0	Lot Numer	Lot Number ASCII 码							
1	Lot Numer	Lot Number ASCII 码							
2	Lot Numer	Lot Number ASCII 码							
3	Lot Numer	Lot Number ASCII 码							
4	Wafer Number	Wafer Number							
5	Lot Numer	Lot Number ASCII 码							
6	Lot Numer	Lot Number ASCII 码							
7	Lot Numer	Lot Number ASCII 码							
8	内部编码	内部编码							
9	Y 坐标低位	Y 坐标低位							
10	X 坐标低位	X 坐标低位							
11	X,Y 坐标高地址	Y 坐标高位				X 坐标高位			

12	CP pass ID	CP pass ID
13	CRC8	0x1FFF 1C00~0x1FFF 1C0C 字节地址的数据的 CRC8 校验值
14	内部编码	内部编码
15	内部编码	内部编码

4.5. Flash 选项字节

4.5.1. Flash 选项字节描述

芯片内的 Flash 的信息区的部分区间作为选项字节使用，用来存放芯片或者用户针对应用需要对硬件进行的配置。比如，watchdog 可以选择为硬件或者软件模式。

为了数据的安全性，选项字节以正文及反码形式分别存储。

表 4-3 选项字节格式

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
选项字节 1 反码								选项字节 0 反码							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
选项字节 1								选项字节 0							

选项字节的内容可以从表“选项字节构成”所述的存储器地址读到，也可以从以下选项字节的相关寄存器读到：

- Flash 选项字节寄存器(FLASH_OPTR)
- Flash 写保护寄存器（FLASH_WRPR)
- Flash Bank0 专有代码保护地址寄存器（FLASH_PCROP0）
- Flash Bank1 专有代码保护地址寄存器（FLASH_PCROP1）

表 4-4 选项字节构成

Word 地址	描述
0x1FFF 1D00	用户选项字节 0 及反码
0x1FFF 1D04	用户选项字节 1 及反码
0x1FFF 1D08	保留
0x1FFF 1D0C	保留
0x1FFF 1D10	BANK0 写保护（WRP）地址选项字节及反码
0x1FFF 1D14	BANK1 写保护（WRP）地址选项字节及反码
0x1FFF 1D18	BANK0 PCROP 起始地址字节及反码
0x1FFF 1D1C	BANK0 PCROP 结束地址字节及反码
0x1FFF 1D20	BANK1 PCROP 起始地址字节及反码
0x1FFF 1D24	BANK1 PCROP 结束地址字节及反码
…	保留
0x1FFF 1DFC	保留

4.5.2. 用户选项字节 0

Flash 地址: 0x1FFF 1D00

出厂值：0x4055 BFAA

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~nBOOT1	~NRST_MODE	~WWDG_SW	~IWDG_SW	~IWDG_STOP	~IWDG_STDBY	~NRST_STOP	~NRST_STDBY	~RDP[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	IWDG_STOP	IWDG_STDBY	NRST_STOP	NRST_STDBY	RDP[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~nBOOT1	R	nBOOT1 的反码
30	~NRST_MODE	R	NRST_MODE 的反码
29	~WWDG_SW	R	WWDG_SW 的反码
28	~IWDG_SW	R	IWDG_SW 的反码
27	~IWDG_STOP	R	IWDG_STOP 的反码
26	~IWDG_STDBY	R	IWDG_STDBY 的反码
25	~NRST_STOP	R	NRST_STOP 的反码
24	~NRST_STDBY	R	NRST_STDBY 的反码
23:16	~RDP	R	RDP 的反码
15	nBOOT1	R	与 BOOT PIN 一起，选择芯片启动模式 具体描述参见 3.5“启动模式”章节
14	NRST_MODE	R	0：仅复位输入 1：GPIO 功能
13	WWDG_SW	R	0：硬件窗口看门狗 1：软件窗口看门狗
12	IWDG_SW	R	0：硬件独立看门狗 1：软件独立看门狗
11	IWDG_STOP	R	停止模式 IWDG 计数控制 0：停止模式下 IWDG 计数停止 1：停止模式下 IWDG 计数正常
10	IWDG_STDBY	R	待机模式 IWDG 计数控制 0：待机模式下 IWDG 计数停止 1：待机模式下 IWDG 计数正常
9	NRST_STOP	R	0：进入停止模式时产生复位 1：进入停止模式时不产生复位
8	NRST_STDBY	R	0：进入待机模式时产生复位 1：进入待机模式时不产生复位
7:0	RDP	R	读保护级别。 0xAA：级别 0，未激活读保护 非 0xAA：级别 1，激活芯片读保护

4.5.3. 用户选项字节 1

Flash 地址：0x1FFF 1D04

出厂值：0xFDED 0212

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	ECC_EN	~BOOT_LOCK	.~BFB	~SWD_MODE		~SRAM_PE	Res.	Res.	Res.	Res.
						R	R	R	R	R	R				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	ECC_EN	BOOT_LOCK	.BFB	SWD_MODE		SRAM_PE	BOR_LEV[2:0]			BOR_EN
						R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:26	Reserved	R	Bit[15:10]的反码
25	~ECC_EN	R	ECC_EN 的反码
24	~BOOT_LOCK	R	BOOT_LOCK 的反码
23	~BFB	R	BFB 的反码
22:21	~SWD_MODE	R	SWD_MODE 的反码
20	~SRAM_PE	R	SRAM_PE 的反码
19:10	Reserved	R	
9	ECC_EN	R	ECC 使能 0：禁止 ECC 1：使能 ECC
8	BOOT_LOCK	R	用于强制从用户 Flash 区域启动 0：基于 IO 引脚/选项位配置启动 1：从 Flash 主存储区强制启动
7	BFB	R	0：从 Bank0 启动(Flash Bank0 映射到 0x0800_0000/0x0000_0000，Flash Bank1 映射到 0x0802_0000/0x0002_0000) 1：从 Bank1 启动(Flash Bank1 映射到 0x0800_0000/0x0000_0000，Flash Bank0 映射到 0x0802_0000/0x0002_0000)
6:5	SWD_MODE[1:0]	R	SWD PIN 模式选择。 SWD_MODE[1:0] / PA4 / PA5 / PB6 / PB5 00(default) / SWDIO / SWCLK / GPIO / GPIO 01 / GPIO / GPIO / SWDIO / SWCLK 10 / GPIO / SWCLK / SWDIO / GPIO 11 / SWDIO / GPIO / GPIO / SWCLK PA5 和 PB5 通过 SWD_MODE 配置为 SWCLK 时，会置于下拉模式 PA4 和 PB6 通过 SWD_MODE 配置为 SWDIO 时，会置于上拉模式
4	SRAM_PE	R	SRAM 奇偶校验使能 0：禁止 SRAM 奇偶校验 1：使能 SRAM 奇偶校验
3:1	BOR_LEV[2:0]	R	000：BOR 上升阈值为 1.8V，下降阈值位 1.7V 001：BOR 上升阈值为 2.0V，下降阈值位 1.9V 010：BOR 上升阈值为 2.2V，下降阈值位 2.1V 011：BOR 上升阈值为 2.4V，下降阈值位 2.3V 100：BOR 上升阈值为 2.8V，下降阈值位 2.7V 101：BOR 上升阈值为 3.1V，下降阈值位 3.0V 110：BOR 上升阈值为 3.7V，下降阈值位 3.6V 111：BOR 上升阈值为 4.2V，下降阈值位 4.1V
0	BOR_EN	R	BOR 使能 0：BOR 不使能

			1：BOR 使能，BOR_LEV 起作用

4.5.4. Flash 保护配置 2（BANK0_WRP）

Flash 地址: 0x1FFF 1D10

出厂值：0x0000 FFFF

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BANK0_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BANK0_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~BANK0_WRP	R	BANK0_WRP 的反码
15:0	BANK0_WRP	R	0：扇区[y]被保护 1：扇区[y]无保护 y=0~15

4.5.5. Flash 保护配置 3（BANK1_WRP）

Flash 地址: 0x1FFF 1D14

出厂值：0x0000 FFFF

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BANK1_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BANK1_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~BANK1_WRP	R	BANK1_WRP 的反码
15:0	BANK1_WRP	R	0：扇区[y]被保护 1：扇区[y]无保护 y=0~15

4.5.6. Flash 保护配置 4（PCROP0SR）

Flash 地址: 0x1FFF 1D18

出厂值：0xFE00 01FF

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP0SR[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
7'h0							PCROP0SR[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	R	7'h7F
24:16	~PCROP0SR	R	PCROP0SR 的反码
15:9	Reserved	R	7'h0
8:0	PCROP0SR	R	BANK0 专有代码保护起始地址(以页为单位)

4.5.7. Flash 保护配置 5（PCROP0ER）

Flash 地址: 0x1FFF 1D1C

出厂值：0xFFFF 0000

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP0ER[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
7'h0							PCROP0ER[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	R	7'h7F
24:16	~PCROP0ER	R	PCROP0ER 的反码
15:9	Reserved	R	7'h0
8:0	PCROP0ER	R	BANK0 专有代码保护结束地址（以 Page 为单位)

4.5.8. Flash 保护配置 6（PCROP1SR）

Flash 地址: 0x1FFF 1D20

出厂值：0xFE00 01FF

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP1SR[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
7'h0							PCROP1SR[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	Reserved	R	7'h7F
24:16	~PCROP1SR	R	PCROP1SR 的反码
15:8	Reserved	R	7'h0
8:0	PCROP1SR	R	BANK1 专有代码保护起始地址

4.5.9. Flash 保护配置 7（PCROP1ER）

Flash 地址: 0x1FFF 1D24

出厂值：0xFFFF 0000

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
7'h7F							~ PCROP1ER[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7'h0							PCROP1ER[8:0]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	Reserved	R	7'h7F
24:16	~PCROP1ER	R	PCROP1ER 的反码
15:8	Reserved	R	7'h0
8:0	PCROP1ER	R	BANK1 专有代码保护结束地址(以 Page 为单位)

4.5.10. Flash 选项字节编程

复位后，FLASH_CR 寄存器中与选项字节相关的位是被写保护的。当对选项字节进行相关操作前，FLASH_CR 寄存器中的 OPTLOCK 位必须被清零（注意：每次擦除或编程都要如此操作）。

以下步骤用来解锁该寄存器：

1. 通过解锁时序，解锁 FLASH_CR 寄存器的写保护（参见 4.3.5.1“解锁 Flash 操作”）
2. 向 FLASH_OPTKEYR 寄存器，写 OPTKEY1=0x0819 2A3B
3. 向 FLASH_OPTKEYR 寄存器，写 OPTKEY2=0x4C5D 6E7F

任何错误的时序都会锁定 FLASH_CR 寄存器，直到下一次复位。如果密钥操作顺序不正确，会产生总线错误状态，并产生 HardFault 中断。

Flash 信息区的选项字节可以通过软件写 FLASH_CR 寄存器的 OPTLOCK 位为 1 锁定，以防止不想要的擦除/编程操作。

如果软件置位 FLASH_CR.LOCK 位，则 OPTLOCK 位也被自动置位。

4.5.10.1. 修改用户选项字节步骤

选项字节的编程操作，跟对 Flash 主存储区的操作不一样。为修改选项字节，需要进行如下步骤：

1. 用之前描述的步骤，清零 OPTLOCK 位
2. 检查 BSY 位，确认没有正在进行的 Flash 操作
3. 向选项字节寄存器 FLASH_OPTR/FLASH_WRPR/FLASH_PCROPR0/FLASH_PCROPR1 写期望的值（1~4 个 word）（若非字对齐写入，则忽略该操作）
4. 置位 OPTSTRT 位
5. 向 0x1FFF_1D00 写任意 32bit 数据（触发正式的写操作）（若非字对齐写入，则会产生 HardFault; 写入非 0x1FFF_1D00 地址，则出现错误，不能正常执行操作）
6. 等待 BSY 位被清零
7. 等待 EOP 拉高，软件清零

任何对选项字节的改动，硬件都会先把选项字节对应的整个页擦除掉，然后用 FLASH_OPTR、FLASH_WRPR 或者 FLASH_PCROPRx（x=0，1）寄存器的值，写到选项字节中。并且，硬件自动计算相应的补码，并把计算值写到选项字节的相应区域。

4.5.10.2. 选项字节加载

在 BSY 位被清零后，所有新的选项字节被写入了 Flash 信息存储区中，但是未应用于芯片系统。对选项字节寄存器进行读操作，仍然返回上一次被加载的选项字节里的值。仅当新值被加载后，才对芯片系统起作用。

选项字节的加载，在以下两种情况下进行：

- 当 FLASH_CR 寄存器中的 OBL_LAUNCH 位被置位
- 在上电复位后（POR/BOR）
- 退出待机模式，且 PWR_CR3.LD_EN 配置为 1

“加载选项字节”进行的操作是：对 Flash 信息存储区的选项字节页进行读操作，再把读出的数据存储在内部选项字节寄存器中（FLASH_OPTR、FLASH_WRPR 和 FLASH_PCROPRx）。这些内部寄存器配置系统，并可以被软件读。置位 OBL_LAUNCH 位，产生一个复位，在系统的复位下执行选项字节的装载。

每个选项字节位在它相同的双字地址（下一个半字）有相应的补码。在选项字节加载期间，会对选项位和其补码进行验证，这能确保加载被正确的进行了。

如果正补码匹配，则选项字节被复制到选项字节寄存器中。

如果正补码不匹配，则 FLASH_SR 寄存器的 OPTVERR 状态位被置位。选项字节寄存器按如下定义写入值：

- 对于用户选项
  - BOR_EN 位写成 0（BOR 不使能）
  - NRST_MODE 位写成 0（仅复位输入）
  - RDP 位写成 0xFF（即级别 1 读保护）
  - BOOT_LOCK 位写为 0
  - BFB 写为 0
  - SWD_MODE 写为 0
  - 其余不匹配的值都写成 1
- 对于 PCROP 选项，PCROPxSR= 0x000，PCROPxER =0x1FF，即所有 Flash 空间设定为保护
- 对于 WRP 选项，不匹配的值是缺省值“无写保护”

在系统复位后，选项字节的内容被复制到下面的选项寄存器（软件可读可写）：

- FLASH_OPTR
- FLASH_PCROPRx
- FLASH_WRPR

这些寄存器也被用来修改选项字节。如果这些寄存器未由用户修改，则反映了系统的选项状态。

4.6. Flash 出厂配置字节

芯片内的 Flash 的信息区的部分区间（共 2 个页）作为出厂配置字节使用。

出厂配置字节 0 存放供软件读取信息（仅有正码，无反码存放）：

- HSI 频率选择控制值，及对应的 Trimming 值
- 对应 HSI 2MHz 的擦写时间配置参数值

4.6.1. HSI_TRIMMING_FOR_USER

Flash 地址: 0x1FFF 1E04~0x1FFF 1E18

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSI_FS[2:0]			HSI_TRIM[12:0]												
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件需要从该地址读出数据，再写入 RCC_ICSCR 寄存器对应的 HSI_FS[2:0]和 HSI_TRIM[12:0]，以实现 HSI 频率的更改。

4.6.2. NORMAL_TS_CALIBRATION_PARA

Flash 地址: 0x1FFF 1E20

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	NTS_{CAL}[11:0]											
				R	R	R	R	R	R	R	R	R	R	R	R

软件需要从该地址读出数据，获取常温校准参数。

4.6.3. HIGH_TS_CALIBRATION_PARA

Flash 地址: 0x1FFF 1E24

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	HTS_{CAL}[11:0]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件需要从该地址读出数据，获取高温校准参数。

4.6.4. HSI2M_EPPARA0

Flash 地址: 0x1FFF 1E28(2 MHz)

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	TS1[10:0]										
					R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.							TS0[8:0]								
							R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_TS0、FLASH_TS1 寄存器，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.5. HSI2M_EPPARA1

Flash 地址: 0x1FFF 1E2C(2 MHz)

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS3[8:0]								
							R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_TS3 寄存器，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.6. HSI2M_EPPARA2

Flash 地址: 0x1FFF 1E30(2 MHz)

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	TPS3[12:0]												
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P[8:0]								
							R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_TPS3 和 FLASH_TS2P 寄存器，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.7. HSI2M_EPPARA3

Flash 地址: 0x1FFF 1E34(2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERTPE[17:16]	
														R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_PERTPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.8. HSI2M_EPPARA4

Flash 地址: 0x1FFF 1E38(2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMERTPE[17:16]	
														R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMERTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_SMERTPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.9. HSI2M_EPPARA5

Flash 地址: 0x1FFF 1E3C(2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRGTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_PRGTPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.6.10. HSI2M_EPPARA6

Flash 地址: 0x1FFF 1E40(2 MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PRETPE[13:0]													
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件从该地址读出对应 HSI2MHz 数据，再写入 FLASH_PRETPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.7. Flash 用户数据字节

4.7.1. Flash 用户数据描述

Flash 信息区存在 1 个页作为用户数据区。对本页的编程和擦除是按照 Flash 主存储区的方法来处理。另外，Flash 主存储区的片擦对本区域无效。

设定 USER OTP MEMORY LOCK 内容不会立刻更新，直到上电复位（POR/BOR）加载，从而会起到保护功能。

表 4-5 Flash User Data 用于存在用户数据

<table>
<tr><th>Page</th><th>Address</th><th>Contents</th></tr>
<tr><td rowspan="7">26</td><td>0x1FFF 1A00</td><td>Bit[31:16]:存放用户数据
Bit[15:0]: USER OTP MEMORY LOCK
<table>
<tr><th>USER OTP MEMORY LOCK</th><th>User Data protection</th></tr>
<tr><td>0xAA55</td><td>读：可以
编程和擦操作：禁止</td></tr>
<tr><td>其他值</td><td>读、编程和擦操作：可以</td></tr>
</table></td></tr>
<tr><td>0x1FFF 1A04</td><td>存放用户数据</td></tr>
<tr><td>0x1FFF 1A08</td><td>存放用户数据</td></tr>
<tr><td>…</td><td>存放用户数据</td></tr>
<tr><td>…</td><td>存放用户数据</td></tr>
<tr><td>…</td><td>存放用户数据</td></tr>
<tr><td>0x1FFF 1AFC</td><td>存放用户数据</td></tr>
</table>

4.7.2. Flash 用户数据编程

上电复位后，根据加载的 USER OTP MEMORY LOCK 的值，决定用户数据区是否可以编程。

如果可以编程，编程步骤如下：

修改用户数据

用户数据区的编程操作，跟对 Flash 主存储区的操作一样。需要进行如下步骤：

1) 检查 BSY 位，确认没有正在进行的 Flash 操作
2) 如果没有正在进行的 Flash 擦除或者编程操作，则软件读出该 Page 的 64 个 word
3) 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护
4) 置位 FLASH_CR 寄存器的 UPG 位和 EOPIE（如果需要产生 EOP 中断）位
5) 向 User data 区地址进行第 1 到第 63 个 word 的编程操作（只接受 32bit 的编程）
6) 置位 FLASH_CR 寄存器的 UPGSTRT
7) 写第 64 个 word
8) 等待 BSY 位被清零
9) 等待 EOP 拉高，软件清零

当步骤 7）成功执行，则编程操作自动启动，同时 BSY 位被硬件置位。

4.8. Flash 存储区保护

对 Flash 主存储区的保护包括以下几种机制：

- 读保护(RDP)，防止来自外部对 Flash 主存储区访问。
- 写保护(WRP)控制，以防止不想要的写操作（由于程序存储器指针的混乱）。写保护的粒度设计为 8 KB。
- 选项字节写保护，专门的解锁设计。
- 专有代码读出保护(PCROP)。PCROP 保护的粒度设计为 256 bytes。

4.8.1. 专有代码读出保护（PCROP）

通过配置，Flash 主存储区部分页可以防止来自第三方非授权的读写。

保护区为仅执行区域：它只能由 CPU 作为指令代码访问，而严格禁止所有其他访问（DMA、调试和 CPU 数据读取、写入和擦除）。当 RDP 保护从级别 1 更改为级别 0 时，擦除 PCROP 区域。

每个 PCROP 区域由与物理闪存地址相关的起始页和结束页地址定义。这些地址定义在 PCROP 地址寄存器 FLASH PCROP 起始地址寄存器（FLASH_PCROP0SR，FLASH_PCROP1SR）、FLASH PCROP 结束地址寄存器（FLASH_PCROP0ER，FLASH_PCROP1ER）。FLASH_PCROP0SR 和 FLASH_PCROP0ER 针对 Bank0，FLASH_PCROP1SR 和 FLASH_PCROP1ER 针对 Bank1.

PCROP 区域的范围定义为地址：

Flash 基址 + [PCROPxSR x 0x100]（包含）

到地址：

Flash 基址 + [(PCROPxER + 1) x 0x100]（不包含）。

当 PCROPxER = PCROPxSR 时，PCROPxSR（或 PCROPxER）配置的页受 PCROP 保护。

例如：

要对地址区域 0x0800 0800 到 0x0800 13FF 进行 PCROP 保护，当 BFB=0 时，寄存器配置如下所示:

– PCROP0SR = 0x08（PCROP 区域起始地址 0x0800 0800）

– PCROP0ER = 0x13（PCROP 区域结束地址 0x0800 13FF）

要对地址区域 0x0800 0800 到 0x0800 13FF 进行 PCROP 保护，当 BFB=1 时，寄存器配置如下所示:

– PCROP1SR = 0x08（PCROP 区域起始地址 0x0800 0800）

– PCROP1ER = 0x13（PCROP 区域结束地址 0x0800 13FF）

对 PCROP 保护区执行的任何读取数据访问都会触发 RDERR 标志错误。

任何受 PCROP 保护的地址也是写保护的，对这些地址之一的任何写访问都会触发 WRPERR。

任何 PCROP 区域也受到擦除保护。因此，对该区域中的页面的任何擦除都是不可能的（包括包含该区域的开始地址和结束地址的页面）。此外，如果一个区域受到 PCROP 保护，则不能执行软件 Bank 片擦除。

用户可以通过信息区选项字节来修改 PCROP 区域。当 RDP 从级别 1 更改为级别 0 时会禁用 PCROP；而 RDP 其他的变化(保持不变或者从级别 0 更改为级别 1)时，只允许扩大 PCROP 区域，不允许缩小 PCROP 区域，缩小 PCROP 区域的动作的结果是 PCROP 区域保持不变。

表 4-6 PCROP 保护

PCROPx 寄存器值(x = 0,1)	PCROP 保护区
PCROPxSR >PCROPxER	无 PCROP 区域。
PCROPxSR <PCROPxER	PCROPxSR 和 PCROPxER 之间的区域受到保护。
PCROPxSR =PCROPxER	PCROPxSR（或 PCROPxER）配置的页受到保护。

4.8.2. Flash 读保护

通过设置 RDP 选项字节，并进行 POR/BOR 或者 OBL 复位装载新的 RDP 选项字节，可以激活读保护功能。RDP 保护 Flash 主存储区和备份域寄存器。

如果调试器通过 SWD 仍连接时设置读保护，需要进行上电复位而不是其他系统复位。

当 RDP 选项字节和补码成对正确存在于选项字节时，Flash 主存储区会被保护。

表 4-7 Flash 读保护状态

RDP 字节值	RDP 补码字节值	读保护级别
0xAA	0x55	级别 0
除(0xAA 和 0x55)组合的任何值		级别 1

级别 0:无保护

可对 Flash 主存储区执行读、编程和擦除操作，也可对选项字节和备份寄存器进行任何操作。

级别 1:读保护

当选项字节里的 RDP 及其补码包含任何 [0xAA,0x55] 之外的组合，则级别 1 读保护生效。

- 用户模式：在用户模式下执行的程序（从 Flash 主存储区自举），可以对 Flash 主存储区、选项字节和备份寄存器进行所有操作。
- 调试、从 SRAM 自举以及从系统存储器自举模式 :在调试模式，或者当从 SRAM 或者系统存储区（系统自举程序）启动，Flash 主存储区和备份寄存器是不能被读数据访问的。在这些模式下，对 Flash 主存储区读或者写访问产生一个总线错误，以及产生一个 HardFault 中断。

当已处于级别 1（0xAA 之外任何数），如果要修改为级别 0（写 0xAA），硬件会对 Flash 主存储区进行片擦除操作（分别 bank0 和 bank1 片擦除）。

4.8.3. Flash 写保护 (WRP)

Flash 可以被设置成写保护，以应对不想要的写操作。定义 WRP 寄存器每 bit 的控制粒度为 8KB 的写保护（WRP）区域，即 1 个扇区大小。具体参见 WRP 寄存器的描述。

当被 WRP 的区域被激活，则不允许进行擦除或者编程操作。相应的，即使只有一个区域被设定为写保护，则片擦除功能不起作用。

此外，如果尝试对设为写保护的区域进行擦除或者编程操作，则 FLASH_SR 寄存器的写保护错误标识（WRPERR）会被置位。

注：写保护仅对 Flash 主存储区起作用，对系统存储区不起作用。

4.8.4. 选项字节写保护

缺省情况下，选项字节是可读，并进行写保护的。为获得对选项字节的擦除或者编程访问，需要向 OPTKEYR 寄存器写入正确的序列。

4.8.5. Flash 存储区保护权限汇总

表 4-8 Flash 访问权限

No.	访问对象	保护类型	保护等级	是否在保护区	错误类型	从 main flash 自举									非 main 区自举									
						CPU				DMA		SWD			CPU				DMA		SWD			
						读数据	读指令	写	擦	读数据	写/擦	读数据	写	擦	读数据	读指令	写	擦	读数据	写/擦	读数据	写	擦
1	主存储区/备份寄存器	RDP(2)	0	-	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
2			1	-	总线错误	Y(1)	Y(1)	Y(1)	Y(1)	Y(1)	N	N	N	N	N	N	N	N	N	N	N	N	N
3	主存储区	PCORP(2)	-	不在	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
4			-	在	读/写错误标志	N	Y	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
5		WRPR(2)	-	不在	-	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
6			-	在	写错误标志	Y	Y	N	N	Y	N	Y	N	N	Y	Y	N	N	Y	N	Y	N	N
11	系统存储区			-	写总线错误	Y	Y	N	N	Y	N	Y	N	N	Y	Y	N	N	Y	N	Y	N	N
12	OTP	OTP_LOCK=0		-	-	Y	-	Y	Y	Y	N	Y	Y	Y	Y	-	Y	N	Y	N	Y	Y	Y
13		OTP_LOCK=1		-	写错误标志	Y	-	N	N	Y	N	Y	N	N	Y	-	N	N	Y	N	Y	N	N
15	FT/UID/出厂配置			-	写总线错误	Y	-	N	N	Y	N	Y	N	N	Y	-	N	N	Y	N	Y	N	N
17	选项字节	-			-	Y	-	Y	-	Y	N	Y	Y	-	Y	-	Y	-	Y	N	Y	Y	-

注：

(1) 只要 SWD 访问过，CPU/DMA 访问都是不允许的。可以访问的情况是指，没有 SWD 连接过的情况；

(2) 不会同时出现总线错误和读写错误标志的情况，比如读写的地址，RDP=1 并且在 PCROP 的保护区，则只会出现总线错误。

4.9. Flash 中断

表 4-9 Flash 中断请求

中断事件	事件标志	事件标志/中断清零	中断使能
操作结束	EOP	写 EOP=1	EOPIE
写保护错误	WRPERR	写 WRPERR=1	ERRIE
Bank0 PCROP 保护错误	RDERR0	写 RDERR0=1	RDERR0IE
Bank1 PCROP 保护错误	RDERR1	写 RDERR1=1	RDERR1IE
ECC 错误纠正	ECCC	写 ECCC=1	ECCCIE

注：以下事件没有单独的中断标识，但会产生 HardFault：

- 解锁 FLASH_CR 寄存器的序列错误
- 解锁 Flash 选项字节的写操作序列错误
- Flash 编程操作未进行 32 位数据的对齐
- Flash 擦除（含页 擦除、扇区擦除和片擦除）操作未进行 32 位数据对齐

4.10. Flash 寄存器

4.10.1. Flash 访问控制寄存器 (FLASH_ACR)

偏移地址: 0x00

复位值: 0x0000 0700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.				Res.	Res.	Res.	Res.	Res.	Res.	LATENCY[1:0]	
					RW	RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	DCEN	RW	1	数据缓存使能 0：数据缓存禁止 1：数据缓存使能
9	ICEN	RW	1	指令缓存使能 0：指令缓存禁止 1：指令缓存使能
8	PRFTEN	RW	1	预取使能位 0：预取禁止 1：预取使能
7:2	Reserved	-	-	保留
1:0	LATENCY[1:0]	RW	0	Flash 读操作对应的等待状态： 0：Flash 读操作没有等待状态（系统时钟在 24 MHz 及以下） 1：Flash 读操作有 1 个等待状态，即每次读 Flash 需要 2 个系统时钟周期（系统时钟在 > 24 MHz，≤48MHz）

				2：Flash 读操作有 2 个等待状态，即每次读 Flash 需要 3 个系统时钟周期（系统时钟在 > 48 MHz，≤72 MHz）

4.10.2. Flash 密钥寄存器 (FLASH_KEYR)

偏移地址: 0x08

复位值: 0x0000 0000

所有寄存器位是只写，读出返回 0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[31:16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
KEY[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:0	KEY[31:0]	W	0x0000	下面的值必须被连续的写入，才能解锁 FLASH_CR 寄存器，并使能 Flash 的编程/擦除操作 KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.10.3. Flash 选项字节密钥寄存器 (FLASH_OPTKEYR)

偏移地址: 0x0C

复位值: 0x0000 0000

所有寄存器位是只写，读出返回 0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPTKEY[31:16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPTKEY[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:0	OPTKEY[31:0]	W	0x0000 0000	下面的值必须被连续的写入，才能解锁 Flash 的选项寄存器，并使能选项字节的编程/擦除操作 KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F 注：在解锁该寄存器前需要先解锁 FLASH_KEYR。

4.10.4. Flash 状态寄存器(FLASH_SR)

偏移地址: 0x10

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY1	BSY0
														R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTVERR	Res.	USRLOCK	RDERR1	RDERR0	Res.	Res.	Res.	Res.	Res.	Res.	WRPERR	Res.	Res.	Res.	EOP
RC_W1		R	RC_W1	RC_W1							RC_W1				RC_W1

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留

17	BSY1	R	0	Bank1 Busy 位 该位表示 Flash bank1 的操作正在进行。该位在 Flash 操作的开始被硬件置位，当操作完成或者错误产生时由硬件清零。
16	BSY0	R	0	Bank0 Busy 位 该位表示 Flash bank0 的操作正在进行。该位在 Flash 操作的开始被硬件置位，当操作完成或者错误产生时由硬件清零。
15	OPTVERR	RC_W1	0	选项字节校验错误标志。 当选项字节及其反码不匹配时，硬件置位该位。加载失败的选项字节，被强制成安全值。 软件写 1，清零该位。
14	Reserved	-	-	保留
13	USRLOCK	R	0	根据上电读用户数据区第一个 word 的低 16 位的值来指示用户数据区是否可写。 0：读到的值不为 0xAA55，用户数据区可写 1：读到的值为 0xAA55，用户数据区不可写 该位仅能被上电复位复位。
12	RDERR1	RC_W1	0	Bank1 PCROP 读取错误 当要读取的 Bank1 数据的地址属于 Flash 的读取保护区（PCROP 保护）时由硬件置位。如果 Flash_CR 中的 RDERR1IE 置 1，将生成中断。 软件写 1，清零该位。
11	RDERR0	RC_W1	0	Bank0 PCROP 读取错误 当要读取的 Bank0 数据的地址属于 Flash 的读取保护区（PCROP 保护）时由硬件置位。如果 FLASH_CR 中的 RDERR0IE 置 1，将生成中断。 软件写 1，清零该位。
10:5	Reserved	-	-	保留
4	WRPERR	RC_W1	0	写保护错误 当要被编程/擦除的地址处于写保护的 Flash 区域时（受 WRP 保护），硬件置位该位。 当用户数据区在 USRLOCK 为 1 时被编程/擦除，硬件也会置位该位。 当要被编程/擦除/读取数据的地址处于被专有代码保护的 Flash 区域时（PCROP），硬件置位该位。 软件写 1，清零该位。
3:1	Reserved	-	-	保留
0	EOP	RC_W1	0	当 Flash 的编程/擦除操作成功完成时，硬件置位。如果 FLASH_CR 寄存器的 EOPIE 位使能，产生中断。 软件写 1，清零该位。

4.10.5. Flash 控制寄存器 (FLASH_CR)

偏移地址: 0x14

复位值: 0xC000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPTLOCK	Res.	Res.	OBL_LAUNCH	Res.	ERRIE	EOPIE	RDERR1IE	RDERR0IE	Res.	Res.	PGSTRT	UPGSTRT	OPTSTRT	Res.
RS	RS			RS		RW	RW	RW	RW			RW	RW	RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	SER	Res.	Res.	Res.	Res.	Res.	UPER.	UPG	MER1	MER0	PER	PG
				RW						RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RS	1	FLASH_CR 寄存器锁定位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器被锁定。当检测到解锁时序后，该位被硬件清零，解锁 FLASH_CR 寄存器。 如果解锁操作失败，该位仍然保持置位状态，直到下一次系统复位。
30	OPTLOCK	RS	1	选项字节锁定位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器中与选项字节有关的位以及选项字节寄存器 FLASH_OPTR、FLASH_WRPR、FLASH_PCROPRx(x=0,1)都被锁定。当检测到解锁时序后，该位被硬件清零，解锁 FLASH_CR 寄存器以及选项字节相关寄存器。 对 OPTLOCK 位执行解锁序列之前， LOCK 位必须为清零状态。 如果解锁操作失败，该位仍然保持置位状态，直到下一次系统复位。
29:28	Reserved	-	-	保留
27	OBL_LAUNCH	RS	0	强制选项字节加载。 当置位时，该位强制系统进行选项字节的重载。该位仅当选项字节加载完成后被硬件清零。如果 OPTLOCK 位被置位，该位不能被写. 0：选项字节加载完成 1：产生选项字节加载请求，系统产生复位，进行选项字节的重载。
26	Reserved	-	-	保留
25	ERRIE	RW	0	错误中断使能位。 当 FLASH_SR 寄存器的 WRPERR 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
24	EOPIE	RW	0	操作结束中断使能位。 当 FLASH_SR 寄存器的 EOP 位被置位，该位使能中断的产生。 0：EOP 中断关闭 1：EOP 中断使能
23	RDERR1IE	RW	0	Bank1 PCROP 读取错误中断使能 当 FLASH_SR 中的 RDERR1 位被置位，如果该位使能，则产生中断请求。 0：无中断产生

				1：有中断产生
22	RDERR0IE	RW	0	Bank0 PCROP 读取错误中断使能 当 FLASH_SR 中的 RDERR0 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
21:20	Reserved	-	-	保留
19	PGSTRT	RW	0	Flash 主存储区的编程操作的启动位。 该位启动了 Flash 主存储区的编程操作，软件置位，在 FLASH_SR 寄存器的 BSY0 或 BSY1 位被清零后，硬件清零该位。
18	UPGSTRT	RW	0	Flash 信息区的用户数据区编程操作的启动位。 该位启动了 Flash 信息区的用户数据区编程操作，软件置位，在 FLASH_SR 寄存器的 BSY0 位被清零后，硬件清零该位。
17	OPTSTRT	RW	0	Flash 选项字节修改的启动位 该位启动了对选项字节的修改。软件置位，在 FLASH_SR 寄存器的 BSY0 位被清零后，硬件清零该位。 注意：当对 Flash 选项字节进行修改时，硬件自动把整个 256 bytes 的页进行擦除操作，再进行编程操作，其中也包括自动进行补码的写入。
16:12	Reserved	-	-	保留
11	SER	RW	0	8 KB 的扇区擦除操作 该寄存器由软件置位和清零。 0：禁止 Flash 的扇区擦除操作 1：使能 Flash 的扇区擦除操作 注： 1） 扇区擦除不会对 Flash 信息区起作用。 2） 扇区擦除对设定为 WRP 的区域不起作用。
10:6	Reserved	-	-	保留
5	UPER	RW	0	用户数据页擦除操作 该寄存器由软件置位和清零。 0：禁止 Flash 用户数据页的页擦除操作 1：使能 Flash 用户数据页的页擦除操作 注：如果用户数据区不能擦写，则该配置无效。
4	UPG	RW	0	用户数据区编程操作 该寄存器由软件置位和清零。 0：禁止用户数据区的编程操作 1：使能用户数据区的编程操作 注：如果用户数据区不能擦写，则该配置无效。
3	MER1	RW	0	Bank 1 片擦除操作 该寄存器由软件置位和清零。 0：禁止 Flash 的 Bank1 片擦除操作 1：使能 Flash 的 Bank1 片擦除操作
2	MER0	RW	0	Bank 0 片擦除操作 该寄存器由软件置位和清零。

				0：禁止 Flash 的 Bank0 片擦除操作 1：使能 Flash 的 Bank0 片擦除操作
1	PER	RW	0	页擦除操作 该寄存器由软件置位和清零。 0：禁止 Flash 的页擦除操作 1：使能 Flash 的页擦除操作
0	PG	RW	0	编程操作 该寄存器由软件置位和清零。 0：禁止 Flash 的编程操作 1：使能 Flash 的编程操作

4.10.6. Flash ECC 寄存器 (FLASH_ECCR)

偏移地址:0x18

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	ECCC	Res.	Res.	Res	Res.	Res.	ECCIE	Res.	SYSF_ECC	BK_ECC	Res.	Res.	Res.	Res.	ADDR_ECC[16]
RC_W1	RC_W1						RW		R	R					R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDR_ECC[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	复位值	Function
31	ECCD	RC_W1	0	ECC 检测位 在检测到超过 1bit ECC 错误时由硬件置位（仅当先前清除了 ECCC/ECCD 时）。当该位被设置时，生成 NMI 中断。 软件通过写入 1 清除。
30	ECCC	RC_W1	0	ECC 校正 当检测到并纠正了 1 位 ECC 错误时由硬件置位（仅当先前清除了 ECCC/ECCD 时）。当 ECCCIE 设置时，产生中断。 软件通过写入 1 清除。
29:25	Reserved	-	-	保留
24	ECCCIE	RW	0	ECC 校正中断使能 0：禁用 ECCC 中断 1：启用 ECCC 中断。 当 FLASH_ECCR 寄存器中的 ECCC 位被设置时，该位使能中断生成。
23	Reserved	-	-	保留
22	SYSF_ECC	R	0	Flash 信息区 ECC 失败标志位。 0：未使能 ECC 或者 ECC 未错 1：检测到 ECC 错误或有 ECC 纠错
21	BK_ECC	R	0	Flash ECC 故障发生的 Bank 表示哪个 Bank 与 ECC 纠错或 ECC 错误检测有关。 0: Bank 0 1: Bank 1 该标志仅针对 Flash 主存储区。
20:17	Reserved	-	-	保留
16:0	ADDR_ECC[16:0]	R	17’b0	ECC 故障地址

				该位指示哪个区域中的哪个地址与 ECC 纠错或 ECC 错误检测有关。 当产生了 ECCC 或者 ECCD 后，该寄存器表明的错误逻辑地址如下表所示： SYSF_ECC / BK_ECC / 错误逻辑地址 0 / 0 / {12'h080,3'b0, ADDR_ECC[16:0]} 0 / 1 / {12'h080,3'b0, ADDR_ECC[16:0]+Bank0 最大地址} 1 / 0 / {16'h1FFF,3'b0,ADDR_ECC[12:0]}; 注：出现 ECC 错误或者纠错后的标志位（SYSF_ECC 或者 BK_ECC）置位后如果软件未处理该标志，则出现新的错误不会再置位（不管是 SYSF_ECC 还是 BK_ECC）。

4.10.7. Flash 选项字节寄存器 (FLASH_OPTR)

偏移地址: 0x20

复位值: 0b0000 00xx xxxx xxxx xxxx xxxx xxxx xxxx。在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	ECC_EN	BOOT_LOCK	BFB	SWD_MODE		SRAM_PE	BOR_LEV[2:0]			BOR_EN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	IWDG_STOP	IWDG_STDBY	NRST_STOP	NRST_STDBY	RDP[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	ECC_EN	RW	-	ECC 使能 0：禁止 ECC 1：使能 ECC
24	BOOT_LOCK	RW	-	用于强制从用户 Flash 区域启动 0：基于 IO 引脚/选项位配置启动 1：从 Flash 主存储区强制启动
23	BFB	RW	-	0：从 Bank0 启动(Flash Bank0 映射到 0x0800_0000/0x0000_0000，Flash Bank1 映射到 0x0802_0000/0x0002_0000) 1：从 Bank1 启动(Flash Bank1 映射到 0x0800_0000/0x0000_0000，Flash Bank0 映射到 0x0802_0000/0x0002_0000)
22:21	SWD_MODE[1:0]	RW	-	SWD PIN 模式选择。 SWD_MODE[1:0] / PA4 / PA5 / PB6 / PB5 00(default) / SWDIO / SWCLK / GPIO / GPIO 01 / GPIO / GPIO / SWDIO / SWCLK 10 / GPIO / SWCLK / SWDIO / GPIO 11 / SWDIO / GPIO / GPIO / SWCLK PA5 和 PB5 通过 SWD_MODE 配置为 SWCLK 时，会置于下拉模式； PA4 和 PB6 通过 SWD_MODE 配置为 SWDIO 时，会置于上拉模式；

20	SRAM_PE	RW	-	SRAM 奇偶校验使能 0：禁止 SRAM 奇偶校验 1：使能 SRAM 奇偶校验
19:17	BOR_LEV[2:0]	RW	-	000：BOR 上升阈值为 1.8V，下降阈值位 1.7V 001：BOR 上升阈值为 2.0V，下降阈值位 1.9V 010：BOR 上升阈值为 2.2V，下降阈值位 2.1V 011：BOR 上升阈值为 2.4V，下降阈值位 2.3V 100：BOR 上升阈值为 2.8V，下降阈值位 2.7V 101：BOR 上升阈值为 3.1V，下降阈值位 3.0V 110：BOR 上升阈值为 3.7V，下降阈值位 3.6V 111：BOR 上升阈值为 4.2V，下降阈值位 4.1V
16	BOR_EN	RW	-	BOR 使能 0：BOR 未使能 1：BOR 使能
15	nBOOT1	RW	-	与 BOOT PIN 一起，选择芯片启动模式
14	NRST_MODE	RW	-	0：仅复位输入 1：GPIO 功能
13	WWDG_SW	RW	-	0：硬件窗口看门狗 1：软件窗口看门狗
12	IWDG_SW	RW	-	0：硬件独立看门狗 1：软件独立看门狗
11	IWDG_STOP	RW	-	停止模式 IWDG 计数控制 0：停止模式下 IWDG 计数停止 1：停止模式下 IWDG 计数正常
10	IWDG_STDBY	RW	-	待机模式 IWDG 计数控制 0：待机模式下 IWDG 计数停止 1：待机模式下 IWDG 计数正常
9	NRST_STOP	RW	-	0：进入停止模式时产生复位 1：进入停止模式时不产生复位
8	NRST_STDBY	RW	-	0：进入待机模式时产生复位 1：进入待机模式时不产生复位
7:0	RDP	RW	-	读保护级别。 0xAA：级别 0，未激活读保护 非 0xAA：级别 1，激活芯片读保护

4.10.8. Flash 写保护寄存器 (FLASH_WRPR)

偏移地址: 0x28

复位值: 0xXXXX XXXX

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BK1_WRP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BK0_WRP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	BK1_WRPx	RW	1	Bank1 扇区写保护 0：扇区 x(x=0~15)，有写保护，不允许进行编程和擦除 1：扇区 x(x=0~15)，无写保护
15:0	BK0_WRPx	RW	1	Bank0 扇区写保护 0：扇区 x(x=0~15)，有写保护，不允许进行编程和擦除 1：扇区 x(x=0~15)，无写保护

4.10.9. Flash Bank0 专有代码保护地址寄存器 (FLASH_PCROPR0)

偏移地址: 0x30

复位值: 0b0000 000X XXXX XXXX 0000 000X XXXX XXXX。在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0ER[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0SR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24:16	PCROP0ER[8:0]	RW		Bank0 PCROP 区域结束地址(以页为单位)
15:9	Reserved	-	-	保留
8:0	PCROP0SR[8:0]	RW		Bank0 PCROP 区域起始地址(以页为单位)

4.10.10. Flash Bank1 专有代码保护地址寄存器 (FLASH_PCROPR1)

偏移地址: 0x34

复位值: 0b0000 000X XXXX XXXX 0000 000X XXXX XXXX 。在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的选项字节位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1ER[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1SR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24:16	PCROP1ER[8:0]	RW		Bank1 PCROP 区域结束地址(以页为单位)
15:9	Reserved	-	-	保留
8:0	PCROP1SR[8:0]	RW		Bank1 PCROP 区域起始地址(以页为单位)

4.10.11. Flash 低功耗配置寄存器 (FLASH_LPCR)

偏移地址: 0x90

复位值: 0x0000 2800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPRUN_EXITT[2:0]			Res.	LPRUN_ENTERT[2:0]		
									RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLEEP_TIME[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_EN
RW	RW	RW	RW	RW	RW	RW	RW								RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22:20	LPRUN_EXITT[2:0]	RW	0x6	Flash 接口信号 FM_LPMODE 信号下降沿与 FM_ACLKBKx 上升沿的时间控制。该寄存器定义 MSISYS 时钟的计数个数。
19	Reserved	-	-	保留
18:16	LPRUN_ENTERT[2:0]	RW	0x6	Flash 接口信号 FM_LPMODE 信号上升沿与 FM_ACLKBKx 上升沿的时间控制。该寄存器定义 MSISYS 时钟的计数个数。
15:8	SLEEP_TIME[7:0]	RW	0x28	Flash 睡眠时间配置(基于 HSI_10M 计数) 当系统时钟选择 LSI 或者 LSE 时，为获得更优化的运行模式功耗，可选择使用该寄存器的功能（仅推荐在 LSI 或者 LSE 为系统时钟时，使用该功能）。 当使能该功能时，每半个系统时钟低电平周期内 Flash 处于 Sleep 状态的时间宽度为： t_{HSI_10M} * SLEEP_TIME 注意: t_{HSI_10M} 为 HSI_10M 的周期; 为确保 Flash 功能的正确，本寄存器最大设定值推荐设定为 0x28。
7:1	Reserved	-	-	保留
0	SLEEP_EN	RW	0	Flash 睡眠（Sleep）使能 0：禁止 1：使能

4.10.12. Flash TS0 寄存器 (FLASH_TS0)

FLASH_TS0~FLASH_TACLK2PW 寄存器的默认值为 HSI 频率为 2 MHz 时对应的时间配置。HSI 2 MHz 是基于 HSI 的频率分频产生的。

偏移地址: 0x100

复位值: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS0[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	TS0[8:0]	RW	9'h0F	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作。

4.10.13. Flash TS1 寄存器 (Flash_TS1)

偏移地址: 0x104

复位值: 0x0000 0024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	TS1[10:0]										
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:0	TS1[10:0]	RW	11'h24	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作。

4.10.14. Flash TS2P 寄存器 (FLASH_TS2P)

偏移地址: 0x108

复位值: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	TS2P[8:0]	RW	9'h0F	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.15. Flash TPS3 寄存器 (FLASH_TPS3)

偏移地址: 0x10C

复位值: 0x0000 0090

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	TPS3[12:0]												
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:0	TPS3[12:0]	RW	13'h90	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.16. Flash TS3 寄存器 (FLASH_TS3)

偏移地址: 0x110

复位值: 0x0000 000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS3[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	TS3[8:0]	RW	9'h0F	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.17. Flash 页擦 TPE 寄存器 (FLASH_PERTPE)

偏移地址: 0x114

复位值: 0x0000 1B58

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERTPE[17:16]	
														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERTPE[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:0	PERTPE[17:0]	RW	18'h1B58	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.18. Flash 片擦时间 TPE 寄存器 (FLASH_SMERTPE)

偏移地址: 0x118

复位值: 0x0000 1B58

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMERTPE[17:16]	
														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMERTPE[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:0	SMERTPE[17:0]	RW	18'h1B58	该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.19. Flash 编程时间 TPE 寄存器 (FLASH_PRGTPE)

偏移地址: 0x11C

复位值: 0x0000 07D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRGTPE[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PRGTPE[15:0]	RW	16'h7D0	该寄存器为以 HSI 2MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.20. Flash 预编程时间 TPE 寄存器 (FLASH_PRETPE)

偏移地址: 0x120

复位值: 0x0000 0190

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PRETPE[13:0]													
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	PRETPE[13:0]	RW	14'h0190	该寄存器为以 HSI 2MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

4.10.21. Flash 页写等待时间 TACLK2PW 寄存器 (FLASH_TACLK2PW)

偏移地址: 0x124

复位值: 0x0000 0006

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TACLK2PW[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	TACLK2PW	RW	8'h6	Flash ACLK 上升沿到 PW（页写）的等待时间。软件需要在配置 FLASH_CR 寄存器后按照 HSI 的频率配置该寄存器。 该寄存器为以 HSI 2 MHz 计数的计数值。 注：对该寄存器进行写操作需要解锁 CR 寄存器中的 LOCK 位，否则将忽略写操作

5.功耗管理单元（PMU）

5.1. 电源

5.1.1. 电源结构

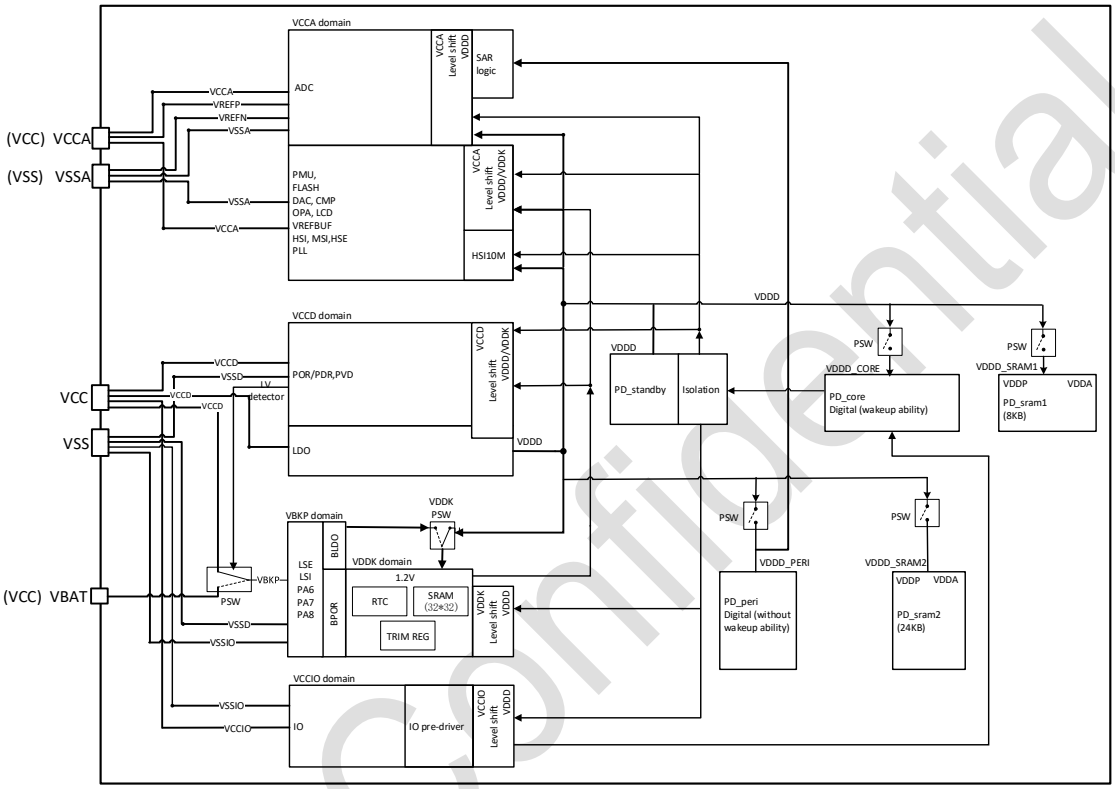


图 5-1 电源结构框图

表 5-2 电源框图

编号	电源	电源值	描述
1	V_{CC}	1.8 ~ 5.5 V	通过电源管脚为芯片提供电源，其供电模块为：部分模拟电路。
2	V_{CCA}	1.8 ~ 5.5 V	给大部分模拟模块供电，来自于 V_{CC} PAD（也可设计单独电源 PAD）。
3	V_{REFP}	1.8 ~ 5.5 V	给 ADC、DAC 提供参考电压。
4	V_{BAT}	1.8 ~ 5.5 V	电池电源。供电范围 1.7 ~ 5.5 V，当 V_{CC} 掉电时，V_{BKP} 域由 V_{BAT} 供电。
5	V_{DDD}	1.2 V	来自于 VR 的输出，为芯片待机模式唤醒电路、IWDG 模块供电。当 MR 供电时，输出 1.2 V。 当进入停止或者待机模式时，根据软件配置，可以为 MR、LPR 或者 DLPR 模式。
6	V_{DDD_CORE}	1.2 V	V_{DDD} 经过数字开关控制的输出，为内部主要逻辑电路（CPU、总线、RCC、以及具有停止模式唤醒能力的数字电路）、以及 SRAM1（8 KB）供电。
7	V_{DDD_SRAM1}	1.2 V	V_{DDD} 经过数字开关控制的输出，为 SRAM1（8 KB）供电。
8	V_{DDD_PEIR}	1.2 V	V_{DDD} 经过数字开关控制的输出，为不具有停止模式唤醒能力的数字电路供电。
9	V_{DDD_SRAM2}	1.2 V	V_{DDD} 经过数字开关控制的输出，为 SRAM2 供电。
10	V_{DDK}	1.2/1.0 V	备份电源。为备份域模块（RTC，备份寄存器，Trim 寄存器）提供电源。

5.1.2. 系统主要电源

5.1.2.1. ADC 和 DAC 参考电压

为确保低电压输入和输出上实现更高精度，用户可将 V_{REFP} 连接至一个低于 V_{CCA} 的独立参考电压。对于模拟输入（ADC）或输出（DAC）信号，V_{REFP} 为最高电压，以满量程值表示。

V_{REFP} 可由外部参考或内部缓冲的电压参考（V_{REFBUF}）来提供。

通过将 V_{REFBUF} 控制寄存器（VREFBUF_CR)中的 VREFBUF_EN 位置 1 来使能内部缓冲的电压参考。V_{REFBUF} 的电压值通过 VREFBUF_CR.VREFBUF_OUT_SEL 寄存器来选择。

5.1.2.2. RTC 电池备份域

为了在 V_{CC} 掉电时，还能保留备份寄存器的内容，并使 RTC 和 TAMP 继续工作，可将 V_{BAT} 引脚连接到由电池或其他备用电源提供的可选备用电源上。

V_{BAT} 引脚为 RTC 和 TAMP 单元、低速 RC 振荡器 LSI、LSE 振荡器和 PA6 到 PA8 I/O 供电，允许 RTC 和 TAMP 在主电源关闭时也可工作。V_{BAT} 电源的开关由 PMU 模块中内置的掉电复位电路进行控制。

注意：

1. 在 $t_{RSTTEMPO}$（V_{CC} 启动后的一段延迟）期间或检测到 PDR 后，V_{BAT} 与 V_{CC} 之间的电源开关仍连接到 V_{BAT}。
2. 在启动阶段，如果 V_{CC} 的建立时间小于 $t_{RSTTEMPO}$（有关 $t_{RSTTEMPO}$ 的值，请参见数据手册）且 $V_{CC} > V_{BAT} + 0.6$ V，会有电流经由 V_{CC} 和电源开关（V_{BAT}）之间连接的内部二极管注入 V_{BAT} 引脚。
3. 如果连接到 V_{BAT} 引脚的电源/电池无法承受此注入电流，则建议在该电源与 V_{BAT} 引脚之间连接一个外部低压降二极管。
4. 如果没有经过 V_{CC} 上电，仅 V_{BAT} 上电，则 PA6、PA7、PA8 的状态为不定态。

5.1.3. 动态电压调节

动态电压调节是一种电源管理技术，包括根据应用性能和功耗要求，增大或减小用于数字外设的电压。

动态电压调节增大数字外设的电压称作过压。它可提高器件性能，芯片运行在更高频率。

动态电压调节减小数字外设的电压称作欠压。它用来在保证定义性能的情况下节省功耗。

LDO 工作在 MR 模式时提供两种电压范围：

5.1.3.1. MR 模式 1：标准电压性能范围

主调压器提供的输出电压，可支持最大系统时钟频率为 72 MHz 的访问。读访问的 Flash 访问时间 30ns，可进行 Flash 编程和擦除操作。

5.1.3.2. MR 模式 3：低电压性能范围

主调压器提供的输出电压，用于在较低系统时钟频率时（如系统时钟为 8 MHz）通过降低电压来减小功耗。读访问的 Flash 访问时间为 30 ns，可进行编程和擦除操作。

当 PWR_CR1 寄存器中的 VR_MODE 配置为 2’b00 时，通过 PWR_CR1.MR_VSEL[1:0]位选择 MR 模式下的电压值。

5.1.3.3. MR 模式从低电压切换到高电压

MR 模式下，系统时钟频率从低频到高频，电压从低压到高压的切换步骤如下：

1. 电源控制寄存器 1 (PWR_CR1) 中的 VR_MODE 位配置为 2’b00
2. 根据支持的频率，参考 MR_VSEL[1:0]中支持的频率配置合适的 MR 电压模式
3. 等待 3 μs
4. 根据支持的频率配置 FLASH_ACR.LATENCY 的值
5. 切换系统时钟频率

5.1.3.4. MR 模式从高电压切换到低电压

MR 模式下，系统时钟频率从高频到低频，电压从高压到低压的切换步骤如下：

1. 电源控制寄存器 1 (PWR_CR1) 中的 VR_MODE 位配置为 2’b00
2. 切换系统频率
3. 根据支持的频率配置 FLASH_ACR.LATENCY 的值
4. 根据支持的频率，参考 MR_VSEL[1:0]中支持的频率配置合适的 MR 电压模式

5.1.3.5. 频率切换与 VR 工作模式切换

当频率切换时，需要同时切换 VR 工作模式时满足：

- MR->LPR 或者 MR->DLPR 时，需要先降低系统时钟频率再切换 VR 模式
- LPR->MR 或者 DLPR->MR 时，需要先切换 VR 模式，再升频（如果需要）

5.2. 电源监测

5.2.1. 上电复位(POR)/下电复位(PDR)/欠压复位(BOR)

芯片内设计 POR/PDR 模块，为芯片提供上电和下电复位。该模块在各种模式之下都保持工作。

除了 POR/PDR 外，还实现了欠压复位 BOR（Brown out reset）。BOR 仅可以通过选项字节，进行使能和关闭操作。

当 BOR 被打开时，BOR 的阈值可以通过选项字节进行选择，且上升和下降检测点都可以被单独配置。

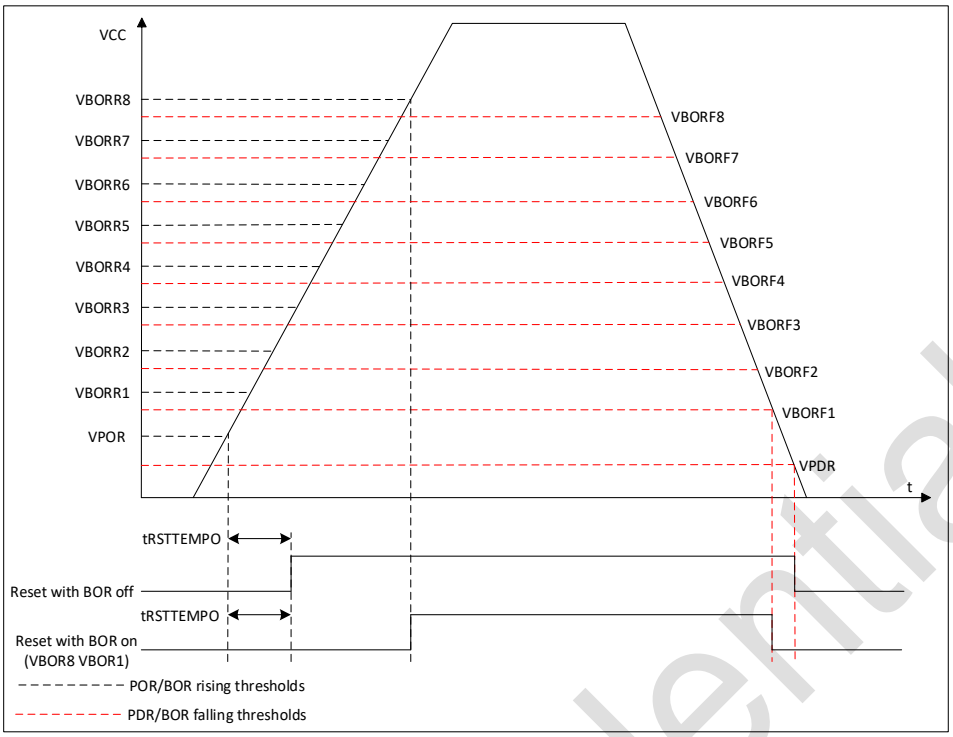


图 5-1 POR, PDR 和 BOR 阈值

5.2.2. 可编程电压检测器（PVD)

该模块可以用来检测 V_{CC} 电源，检测点可通过寄存器进行配置。当检测 V_{CC} 电源时，V_{CC} 高于或者低于 PVD 的检测阈值点（阈值由 PWR_CR2.PLS 配置）时，产生相应的标志。

该事件内部连接到 EXTI 的线 16，取决于 EXTI 线 16 上升/下降沿配置，当 V_{CC} 上升超过 PVD 的检测点，或者 V_{CC} 降低到 PVD 的检测点以下，产生中断，在中断服务程序中用户可以进行紧急关闭系统的任务。

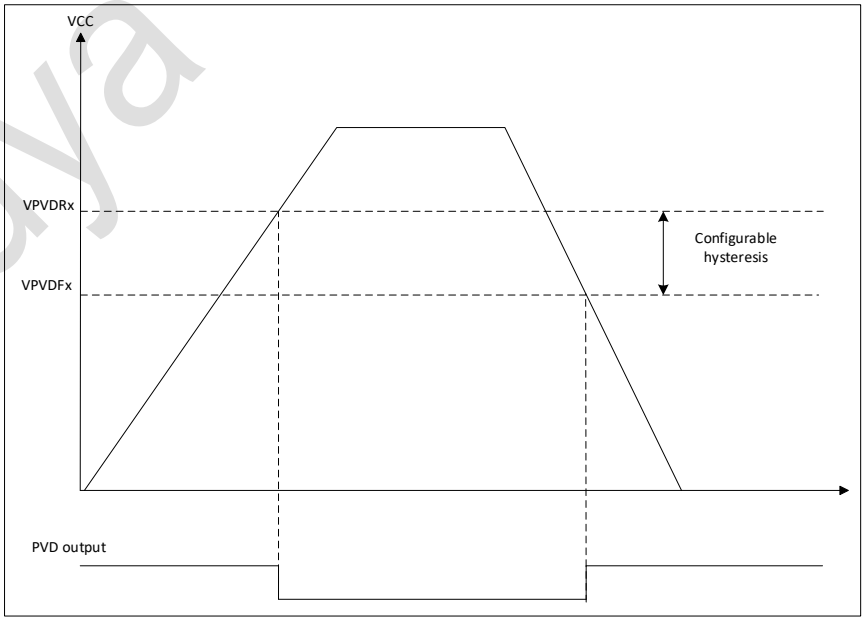


图 5-2 PVD 阈值

5.3. 低功耗控制

缺省状态下，芯片在系统或者电源复位之后，进入运行模式。当 CPU 不需要持续工作时，芯片可进入几种低功耗模式，例如，当等待外部事件时。软件可以在功耗、唤醒时间、唤醒源之间折衷选择。

5.3.1. 功耗模式

5.3.1.1. 分类

芯片有 7 种功耗模式：

- 运行模式（Run）：正常全功能模式，上电后的默认模式。
- 低功耗运行模式（Lower power run）：系统时钟选择 MSISYS，系统时钟频率最大为 2 MHz，数字电源 V_{DDD} 建议工作在 LPR 模式，以节省功耗。在此模式不支持 Flash 擦写。
- 睡眠模式（Sleep）：CPU HCLK 时钟关闭（NVIC，SysTick 等工作），外设可以配置为保持工作。（建议只使能必须工作的模块，在模块工作结束后关闭该模块），数字电源 V_{DDD} 建议工作在 MR 模式。
- 低功耗睡眠模式（Lower power sleep）：从低功耗进入睡眠模式；数字电源 V_{DDD} 建议工作在 LPR 模式
- 停止模式（Stop0/1/2/3）：高速时钟 PLL、HSI、HSE 和 MSI 关闭，LSI 和 LSE 可以根据唤醒源选择是否开启。数字电源 V_{DDD} 工作模式可配置为 MR、LPR 或者 DLPR 模式。这三种模式功耗依次降低，唤醒时间依次变长。
- 待机模式（Standby）：V_{DDD_CORE}、V_{DDD_PERI} 和 V_{DDD_SRAM2} 掉电，V_{DDD_SRAM1} 配置为掉电时，SRAM 数据丢失。V_{DDD} 和 V_{DDK} 工作，待机模式唤醒源，低功耗控制模块工作。

注意：从 Run 模式进入低功耗模式（Sleep/Stopx/Standby），需要配置 HSI 时钟为 8 MHz。

5.3.1.2. 功耗状态转换

下图为各状态转换关系：

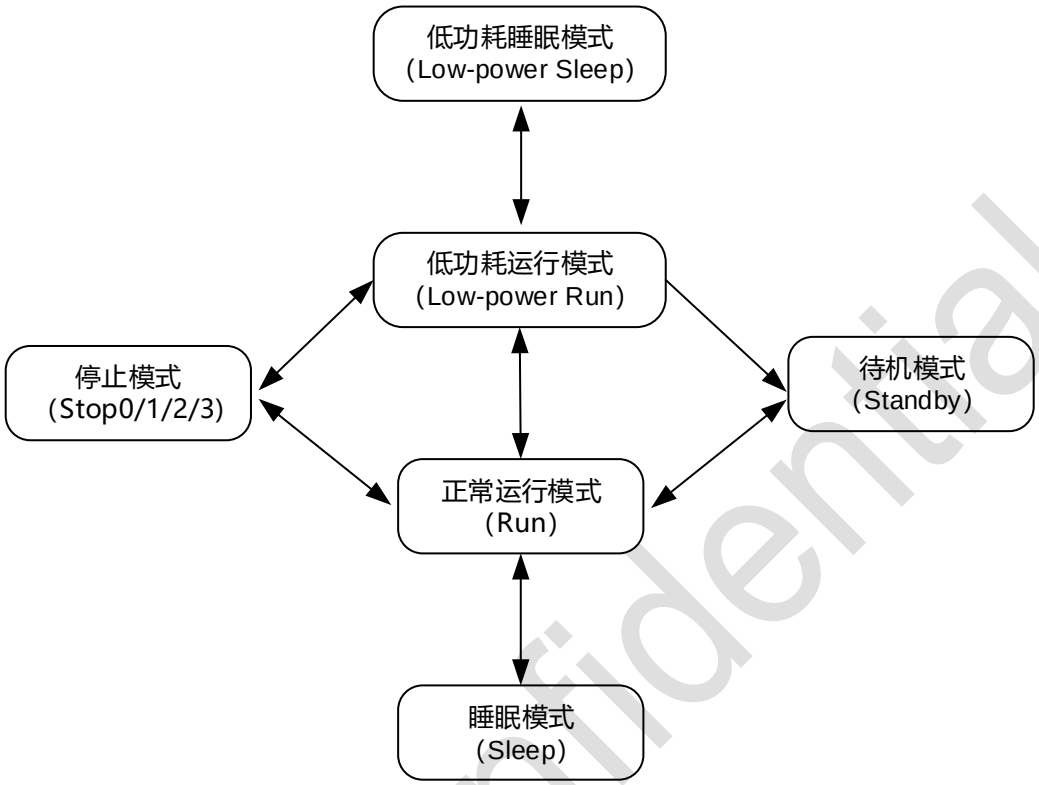


图 5-3 功耗状态转换

在各种模式下模块工作情况见下表，表中 VR 的工作模式以平衡性能和功耗的配置为例。

表 5-3 各种模式下模块工作情况 (1)

外设	RUN	LP RUN V_{DDD} VR@LPR	SLEEP	LP SLEEP V_{DDD} VR@LPR	STOP0		STOP1		STOP2		STOP3		STANDBY		VBAT
					V_{DDD} VR@LPR	唤醒能力	V_{DDD} VR@LPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@PD
CPU Core	Y	Y	-	-	-	-	-	-	-	-	-	-	-	-	-
Flash memory	Y	Y (9)	O (3)	O (3)	- (2)	-	- (2)	-	- (2)	-	- (2)	-	-	-	-
SRAM1	Y	Y	O (3)	O (3)	- (4)	-	- (4)	-	- (4)	-	- (4)	-	-	-	-
SRAM2	Y	Y	O (3)	O (3)	- (6)	-	- (5)	-	- (6)	-	- (5)	-		-	-
Brown-out reset (BOR)	O	O	O	O	O	O	O	O	O	O	O	O	O	O	-
PVD	O	O	O	O	O	O	O	O	O	O	O	O	-	-	-
DMA	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
HSI	O	O (10)	O	O (10)	-	-	-	-	-	-	-	-	-	-	-
MSI	O	O	O	O	-	-	-	-	-	-	-	-	-	-	-
HSE	O	O (10)	O	O (10)	-	-	-	-	-	-	-	-	-	-	-
LSI	O	O	O	O	O	-	O	-	O	-	O	-	O	-	O
LSE	O	O	O	O	O	-	O	-	O	-	O	-	O	-	O
PLL	O	O (10)	O	O (10)	-		-		-		-		-	-	-
HSE CSS	O	O	O	O	-	-	-	-	-	-	-	-	-	-	-
LSE CSS	O	O	O	O	O (8)	O	O (8)	O	O (8)	O	O (8)	O	-	-	-
RTC	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
BKP	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)	O (13)
USART1 USART2	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
UART	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
LPUART1 LPUART2	O	O	O	O	O	O	O	O	O	O	O	O	-	-	-
I2C1 I2C2	O	O	O	O	O	O	O	O	O	O	O	O	-	-	-
SPI1 SPI2	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-

外设	RUN	LP RUN V_{DDD} VR@LPR	SLEEP	LP SLEEP V_{DDD} VR@LPR	STOP0		STOP1		STOP2		STOP3		STANDBY		VBAT
					V_{DDD} VR@LPR	唤醒能力	V_{DDD} VR@LPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@DLPR	唤醒能力	V_{DDD} VR@PD
ADC	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
DAC	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
COMP1/2	O	O	O	O	O	O	O	O	O	O	O	O	-	-	-
OPA	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
Temperature sensor	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
Timers	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
PWM	O	O	O	O	-	-	-	-	- (7)	-	- (7)	-	-	-	-
LPTIM1 LPTIM2	O	O	O	O	O	O	O	O	O	O	O	O	-	-	-
IWDG	O	O	O	O	O	O	O	O	O	O	O	O	O	O	-
WWDG	O	O	O	O	-	-	-	-	- (7)		- (7)	-	-	-	-
SysTick timer	O	O	O	O	-	-	-	-	-	-	-	-	-	-	-
CRC	O	O	O	O	-	-	-	-	- (7)		- (7)	-	-	-	-
LCD	O	O	O	O	O (14)	-	O (14)	-	O (14)	-	O (14)	-	-	-	-
GPIOs	O	O	O	O	O	O	O	O	O	O	O	O	O (11)	O (11)	O (12)
CANFD	O	O	O	O	-	-	-	-	- (7)		- (7)	-	-	-	-

注1. Y = Yes (使能)；O = Optional (默认关闭，可以软件使能)；- = Not available

注2. Flash 不下电，但无时钟提供，进入最低功耗状态。此时，FLASH 需要 3us 唤醒时间。

注3. Flash 和 SRAM1/SRAM2 的时钟在 SLEEPING 模式可以配置开或者关。

注4. SRAM1 不下电，内容保持，但无时钟提供，进入最低功耗状态。

注5. SRAM2 配置为下电。

注6. SRAM2 不下电，内容保持，但无时钟提供，进入最低功耗状态。

注7. 该模块配置为下电。

注8. 进入低功耗模式之前，如果使能了 LSE CSS，则当 LSE CSS 出现问题时，会生成 CPU NMI 接口信号，会唤醒睡眠模式，但不会唤醒停止和待机模式。

注9. Flash 不下电，有时钟提供，进入低功耗运行状态。转换为 RUN 状态时需要等待切换完成。

注10. 为达到最低功耗，该模块建议配置为关闭。

注11. 待机模式唤醒的 IO 正常工作，V_{BKP} 域 IO(PA6/PA7/PA8)正常工作。

注12. 仅 PA6，PA7，PA8 正常工作。

注13. BKP PCLK 控制与 RTC PCLK 一致。

注14. LCD 在此低功耗模式下能维持显示

此外，正常 Run 模式下可以通过下述方法降低功耗：

- 降低系统时钟频率
- 对于不使用的外设，关闭其时钟（系统时钟和模块时钟）

注：在 DLPR 模式，通过 DLPR_VSEL 选择较低电压时，可能因为电压低不能正常工作。不同配置下的功耗值参考数据手册。

5.3.1.3. 进出低功耗模式

表 5-4 低功耗模式进入/退出汇总

模式	进入	唤醒源	唤醒时钟	该模式对时钟的需求/进入该模式步骤	进入该模式对时钟的影响	低功耗期间 VR 模式(VR_MODE 控制)		该模式对时钟的需求/退出该模式步骤	唤醒后 VR 模式（硬件根据进入前 VR_MODE 的配置自动切换或者保持）	
						V_{DDD} VR	V_{DDK} VR		V_{DDD} VR	V_{DDK} VR
睡眠	WFI 或从中断返回	任何中断	与进入之前一样	无要求	CPU HCLK 时钟停止，对其他时钟和时钟源没影响。	MR 注 1	关	无要求	MR	关
	WFE	唤醒事件								
低功耗运行	系统时钟选择 MSI， 且 LPMS=110	模式切换注 2	切换时钟为 HSI，后续时钟取决于软件配置注 3	时钟需求： 1.软件需要使能 MSI 2.系统时钟选择 MSI 且支持 MSIDIV 分频 步骤参见“软件流程”章节。	系统时钟降低到 2MHz 及以下	MR	关	时钟需求： 1.软件需要使能 HSI 2.系统时钟选择 HSI 且支持 HSIDIV 分频 步骤参见“软件流程”章节。	NA 注 3	关
						LPR	关		NA 注 3	关

模式	进入	唤醒源	唤醒时钟	该模式对时钟的需求/进入该模式步骤	进入该模式对时钟的影响	低功耗期间 VR 模式(VR_MODE 控制)		该模式对时钟的需求/退出该模式步骤	唤醒后 VR 模式(硬件根据进入前 VR_MODE 的配置自动切换或者保持)	
						V_{DDD} VR	V_{DDK} VR		V_{DDD} VR	V_{DDK} VR
低功耗睡眠	当前模式为“低功耗运行”，且 SLEEPING=1，且 SLEEPDEEP=0，且 WFI	任何中断	与进入之前一样（MSI）	时钟需求： 1.软件需要使能 MSI 2.系统时钟选择 MSI 且支持 MSIDIV 分频	CPU HCLK 时钟停止，对其他时钟和时钟源没影响。	MR	关	时钟需求： 1.满足进入时需求 2.唤醒时钟为 MSI	MR	关
	当前模式为“低功耗运行”，且 SLEEPING=1，且 SLEEPDEEP=0，且 WFE	唤醒事件	与进入之前一样（MSI）			LPR	关		LPR	关
停止	当前模式为低功耗运行，且 LPMS=000/001/010/011；且 SLEEPDEEP=1；且 WFI 或者从中断返回	任何配置为中断的 EXTI Line(EXTI 寄存器配置)，IWDG 复位，NRST；	与进入之前一样（MSI）	时钟需求： 1.软件需要使能 MSI 2.系统时钟选择 MSI 且支持 MSIDIV 分频	硬件关闭 MSI、HSI、HSE、PLL； LSI 和 LSE 由软件配置选择开或者关；	MR	关	时钟需求： 1.满足进入时需求 2.唤醒时钟为 MSI	MR	关
						LPR	关		LPR	关
						关	DLPR		LPR	关
	当前模式为低功耗运行，且 LPMS=000/001/010/011；且 SLEEPDEEP=1；且 WFE	任何配置为事件的 EXTI Line(EXTI 寄存器配置)，IWDG 复位，NRST；	与进入之前一样（MSI）	时钟需求： 1.软件需要使能 MSI 2.系统时钟选择 MSI 且支持 MSIDIV 分频	硬件关闭 MSI、HSI、HSE、PLL； LSI 和 LSE 由软件配置选择开或者关；	MR	关	时钟需求： 1.满足进入时需求 2.唤醒时钟为 MSI	MR	关
						LPR	关		LPR	关
						关	DLPR		LPR	关
	当前模式为正常运行，且 LPMS=000/001/010/011；且 SLEEPDEEP=1；且 WFI 或者从中断返回	任何配置为中断的 EXTI Line(EXTI 寄存器配置)，IWDG 复位，NRST；	HSI	时钟需求： 1.软件需要使能 HSI，且 HSI 时钟频率配置为 8MHz	硬件关闭 MSI、HSI、HSE、PLL； LSI 和 LSE 由软件配置选择开或者关；	MR	关	时钟需求： 1.满足进入时需求 2.唤醒时钟为 HSI	MR	关
						LPR	关		MR	关
						关	DLPR		MR	关
	当前模式为正常运行，且 LPMS=000/001/010/011；	任何配置为事件的 EXTI Line(EXTI 寄	HSI	时钟需求： 1.软件需要使能 HSI，且 HSI 时钟频率配置为	硬件关闭 MSI、HSI、HSE、PLL；	MR	关		MR	关
						LPR	关		MR	关

模式	进入	唤醒源	唤醒时钟	该模式对时钟的需求/进入该模式步骤	进入该模式对时钟的影响	低功耗期间 VR 模式(VR_MODE 控制)		该模式对时钟的需求/退出该模式步骤	唤醒后 VR 模式(硬件根据进入前 VR_MODE 的配置自动切换或者保持)	
						V_{DDD} VR	V_{DDK} VR		V_{DDD} VR	V_{DDK} VR
	且 SLEEPDEEP=1；且 WFE	存器配置), IWDG 复位, NRST;		8MHz	LSI 和 LSE 由软件配置选择开或者关;	关	DLPR	时钟需求: 1.满足进入时需求 2.唤醒时钟为 HSI	MR	关
待机	当前模式为低功耗运行，LPMS=101； 且 SLEEPDEEP=1, 且 WFI 或者从中断返回 ，或者 WFE	WKUP IO, IWDG 复位, RTC 唤醒,TAMP 事件, NRST;	HSI	需求: 1.软件需要使能 MSI	LSE 和 LSI 可选择开或者关；其他时钟关闭。	MR	关	时钟需求: 1.满足进入时需求 2.唤醒时钟为 HSI	MR	关
						LPR	关		MR	关
						关	DLPR		MR	关
	当前模式为正常运行，LPMS=101； 且 SLEEPDEEP=1, 且 WFI 或者从中断返回 ，或者 WFE	WKUP IO, IWDG 复位, RTC 唤醒、TAMP 事件, NRST;	HSI	时钟需求: 1.软件需要使能 HSI，且 HSI 时钟频率配置为 8MHz	LSE 和 LSI 可选择开或者关；其他时钟关闭。	MR	关	时钟需求: 1.满足进入时需求 2.唤醒时钟为 HSI	MR	关
						LPR	关		MR	关
						关	DLPR		MR	关
V_{BAT}	V_{CC} 掉电	V_{CC} 上电	NA	NA	LSE 和 LSI 维持掉电前的配置；其他时钟关闭。	关	DLPR	NA	MR	关

注 1：为保证模块运行正常，软件要配置 VR 的状态为 MR 模式，才能进入睡眠模式。

注 2：从低功耗运行模式切换到哪种模式，取决于 CPU 的配置。但如果 LPRUN 模式时 VR 工作模式为 LPR，状态切换后 VR 的工作模式为 MR，则需要等待 MR_RDY 后再使能更多模块工作。

注 3：从低功耗运行模式切换模式后，VR 工作模式取决于切换后的 VR 模式配置

注意：VR_MODE 配置为 DLPR 模式时，V_{DDD} LDO 关闭，V_{DDK} LDO 工作，产生数字电路需要的电源。

5.3.2. 降低系统时钟频率

在运行模式下，系统时钟的频率（SYSCLK，HCLK，PCLK）可以通过预分频寄存器配置分频去降低。这些预分频器也可以被用来在进入睡眠或者停止模式前，降低外设的频率。

5.3.3. 软件流程

5.3.3.1. 从“运行”模式切换为“低功耗运行”模式

在从 RUN 模式切换为 LPRUN 模式，以 MSI 时钟运行时需要按照如下步骤：

1）HSI 时钟是否使能？如果未使能必须使能 HSI

2）软件配置 RCC_CR.MSION=1 使能 MSI，等待 RCC_CR.MSIRDY=1 后切换系统时钟为 MSI(配置 RCC_CFGR.SW=3'b101，支持 MSIDIV 分频)；

3）软件配置 PWR_CR1.LPMS=3‘b110；

4）软件配置 PWR_CR1.VR_MODE 为 LPR 模式或者 MR 模式（LPR 模式功耗更低）；

5）如果需要配置 FLASH_ACR.LATENCY=0，或者关闭 HSI 时钟，需要等待 PWR_SR.LPRUN_RUN_SWF=0；

5.3.3.2. 从“低功耗运行”模式切换为“运行”模式

在从 LPRUN 模式切换为 RUN 模式，以更高频率运行时需要按照如下步骤：

1. 软件配置 RCC_CR.HSION=1 使能 HSI；
2. 软件配置 PWR_CR1.VR_MODE 为 MR 模式；
3. 软件配置 PWR_CR1.LPMS=3’b000；
4. 等待（此时 FLASH_ACR.LATENCY=0）：
   1) RCC_CR.HSIRDY=1
   2) PWR_SR.MR_RDY=1 且 PWR_SR.LPRUN_RUN_SWF =1
5. 根据 HSI 频率配置 FLASH_ACR.LATENCY 的值，然后切换系统时钟为 HSI(配置 RCC_CFGR.SW=3'b000，支持 HSIDIV 分频，支持每种 HSI 频率)；如果需要，可以关闭 MSI
6. 如果需要切换其他系统时钟源，按照系统时钟切换步骤执行；

5.3.3.3. 从“运行”模式进入“停止”低功耗模式

系统从 Run 模式进入 Stopx 模式的软件配置流程如下：

1）软件配置 RCC_CR.HSION=1 使能 HSI，且配置 HSI 频率为 8 MHz，等待 RCC_CR.HSIRDY=1 后切换系统时钟为 8 MHz

2）软件配置 PWR_CR1.LPMS=3‘b001/3’b010/3’b011/3’b100

3）软件配置 PWR_CR1.VR_MODE 为 MR 模式/LPR 模式/DLPR 模式（功耗依次降低）

4）软件配置 PWR_CR1.HSION_CTRL=1（默认值），配置 PWR_CR1.FLS_SLPTIME=2’b01（5μs）

5）进入低功耗模式

6）唤醒后 VR_MODE 为 MR 模式，MR 模式根据 PWR_CR1.MR_VSEL 配置；唤醒后系统时钟为 HSI8 MHz

5.3.3.4. 从“低功耗运行”模式进入“停止”低功耗模式

系统从 LPRUN 模式进入 STOPx 模式的软件配置流程如下：

1）当前为低功耗运行模式，系统时钟为 MSI；

2）软件配置 PWR_CR1.LPMS=3‘b001/3’b010/3’b011/3’b100；

3）软件配置 PWR_CR1.VR_MODE 为 MR 模式/LPR 模式/DLPR 模式（功耗依次降低）；

4）软件配置 PWR_CR1.HSION_CTRL=1（默认值），配置 PWR_CR1.FLS_SLPTIME=2’b01（5μs）；

5）进入低功耗模式；

6）唤醒后 VR_MODE 为 LPR 模式，LPR 模式根据 PWR_CR1.LPR_VSEL 配置；唤醒后系统时钟为 MSI；

5.3.3.5. 从“运行”模式进入“待机”低功耗模式

系统从 RUN 模式进入 STANDBY 模式的软件配置流程如下：

1）软件配置 RCC_CR.HSION=1 使能 HSI，且配置 HSI 频率为 8MHz，等待 RCC_CR.HSIRDY=1 后切换系统时钟为 8MHz；

2）软件配置 PWR_CR1.LPMS=3‘b101；

3）软件配置 PWR_CR1.VR_MODE 为 DLPR 模式；

4）进入低功耗模式；

5）唤醒后 VR_MODE 为 MR 模式，MR 模式根据 PWR_CR1.MR_VSEL 配置；唤醒后系统时钟为 HSI8MHz；

5.3.3.6. 从“低功耗运行”模式进入“待机”低功耗模式

系统从 LPRUN 模式进入 STOPx 模式的软件配置流程如下：

1）当前为低功耗运行模式，系统时钟为 MSI

2）软件配置 PWR_CR1.LPMS=3‘b101

3）软件配置 PWR_CR1.VR_MODE 为 DLPR 模式

4）进入低功耗模式

5）唤醒后 VR_MODE 为 MR 模式，MR 模式根据 PWR_CR1.MR_VSEL 配置；唤醒后系统时钟为 HSI8MHz

5.4. PWR 寄存器

5.4.1. Power 控制寄存器 1 (PWR_CR1)

偏移地址: 0x00

复位值: 0x0200 0000

该寄存器在退出待机模式后会被复位。

该寄存器由电源复位（POR/BOR）、系统复位及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	STDBY_MRRDY_WAIT[1:0]		HSION_CTRL	Res.	Res.	Res.	DLPR_VSEL[1:0]		LPR_VSEL[1:0]		MR_VSEL[1:0]	
				RW	RW	RW				RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	FLS_SLPTIME[1:0]		Res.	VR_MODE[1:0]		DBP	Res.	Res.	Res.	Res.	Res.	LPMS[2:0]		
		RW	RW		RW	RW	RW						RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:26	STDBY_MRRDY_WAIT[1:0]	RW	2’b0	待机模式唤醒，MR ready 且 HSI 时钟稳定后，需要等待的时间。 00：0μs 01：5μs 10：10μs 11：20μs 该寄存器会被系统复位复位为 00。
25	HSION_CTRL	RW	1	从所有停止模式（Stop0/1/2/3）唤醒时，HSI 使能的时间控制。 0：与 VR 同时打开，即唤醒时立刻使能 HSI 1：等待 MR 稳定后，使能 HSI 该寄存器会被系统复位复位为 1。
24:22	Reserved	-	-	保留
21:20	DLPR_VSEL[1:0]	RW	2’b0	DLPR 模式电压选择。 00：DLPR 模式 1，高电压模式 01：DLPR 模式 2，低电压模式 10：保留 11：保留
19:18	LPR_VSEL[1:0]	RW	2’b0	LPR 模式电压选择。 00：LPR 模式 1，高电压模式 01：LPR 模式 2，中电压模式 10：LPR 模式 3，中低电压模式 11：LPR 模式 4，低电压模式 注：电压值从模式 1->模式 4,从高到低
17:16	MR_VSEL[1:0]	RW	2’b0	MR 模式电压选择。 00：MR 模式 1，标准电压模式（支持最大频率 72 MHz）

				01：保留 10：MR 模式 3，低电压模式（支持最大频率 8MHz） 11：保留
15:14	Reserved	-	-	保留
13:12	FLS_SLPTIME	RW	2’b0	停止（Stop0/1/2/3）模式唤醒时序中，在 HSI 稳定后，在 FLASH 操作前需要等待时间。 2’b00：3μs 2’b01：5μs（建议配置） 2’b10：1μs 2’b11：0μs
11	Reserved	-	-	保留
10:9	VR_MODE[1:0]	RW	2’b0	调压器工作模式 00：MR 模式 01：LPR 模式 10：DLPR 模式 其它：保留（MR 模式） 注： 1. 待机模式唤醒后该寄存器复位为 00. 2. 所有停止模式，可以配置该寄存器为 00~10； 3. 待机模式，可以配置该寄存器为 00~10; 4. 该寄存器会被系统复位复位为 00（MR 模式）
8	DBP	RW	0	RTC 和备份域寄存器写保护 在复位后，RTC 和备份域寄存器处于写保护状态以防意外写入。要访问 RTC 和备份域寄存器该位必须设置为 1。 0：禁止访问 RTC 和备份寄存器 1：可以访问 RTC 和备份寄存器
7:3	Reserved	-	-	保留
2:0	LPMS[2:0]	RW	0	低功耗模式选择 000：运行模式 001：停止模式 0（所有模块不掉电） 010：停止模式 1（SRAM2 掉电） 011：停止模式 2（不具唤醒能力的模块掉电） 100：停止模式 3（SRAM2 和不具唤醒能力的模块掉电） 101：待机模式 110：低功耗运行模式 该寄存器会被系统复位复位为 3’b000（RUN 模式）。

5.4.2. Power 控制寄存器 2 (PWR_CR2)

偏移地址: 0x04

复位值: 0x0000 0000

注：该寄存器是与 PVD 功能相关寄存器。

该寄存器在退出待机模式后会被复位。

该寄存器由电源复位（POR/BOR）、系统复位及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	FLT_TIME[2:0]			FLTEN	PVD_OUT_SEL	PLS[2:0]			Res.	Res.	Res.	PVDE
				RW	RW	RW	RW	RW	RW	RW	RW				RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:9	FLT_TIME[2:0]	RW	3’b0	PVD 输出数字滤波时间配置 110：滤波时间大约为 30.7ms（1024 个 LSI 或者 LSE 时钟） 101：滤波时间大约为 3.8ms（128 个 LSI 或者 LSE 时钟） 100：滤波时间大约为 1.92ms（64 个 LSI 或者 LSE 时钟） 011：滤波时间大约为 480μs（16 个 LSI 或者 LSE 时钟） 010：滤波时间大约为 120μs（4 个 LSI 或者 LSE 时钟） 001：滤波时间大约为 60μs（2 个 LSI 或者 LSE 时钟） 000：滤波时间大约为 30μs（1 个 LSI 或者 LSE 时钟）
8	FLTEN	RW	0	PVD 输出数字滤波功能使能 0：禁止 1：使能
7	PVD_OUT_SEL	RW	0	PVD 数字输出控制。 0：比较器数字选择模拟 PVD 输出，不经过滤波和同步 1：比较器数字选择经过同步和滤波后的模拟 PVD 输出
6:4	PLS[2:0]	RW	3’b0	电压上升沿检测阈值（下降沿检测阈值相应减小 0.1V）。 000：V_{PVD0} (around 1.8V) 001：V_{PVD1} (around 2.0V) 010：V_{PVD2} (around 2.2V) 011：V_{PVD3} (around 2.4V) 100：V_{PVD4} (around 2.8V) 101：V_{PVD5} (around 3.1V) 110：V_{PVD6} (around 3.7V) 111：V_{PVD7} (around 4.2V) 注：在 PVD_EN 为 0 时配置该寄存器。 如果 SYSCFG_CFGR2.PVDL=1,则该寄存器写保护。只有当系统复位后，写保护才被复位。
3:1	Reserved	-	-	保留
0	PVDE	RW	0	电压检测使能位 0：电压检测禁止 1：电压检测使能 如果 SYSCFG_CFGR2.PVDL=1,则 PVDE 写保护。只有当系统复位后，写保护才被复位。

5.4.3. Power 控制寄存器 3 (PWR_CR3)

偏移地址: 0x08

复位值: 0x0000 0000

该寄存器退出待机模式后不会被复位。

该寄存器仅由 POR 复位，BOR 和系统复位不会复位该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	SRAM1PDSTD	APC	LD_EN	EWUP7	EWUP6	EWUP5	EWUP4	EWUP3	EWUP2	EWUP1	EWUP0
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	SRAM1PDSTD	RW	0	SRAM1 在待机模式掉电控制。 0：SRAM1 在待机模式内容保持 1：SRAM1 在待机模式掉电
9	APC	RW	0	应用上拉和下拉配置 该位确定是否应用 PWR_PUCRx 和 PWR_PDCRx 寄存器中定义的 I/O 上拉和下拉配置。 0：不应用 1：应用
8	LD_EN	RW	0	待机模式唤醒后执行上电加载使能。 0：待机模式唤醒后不做上电加载 1：待机模式唤醒后做上电加载 注：当配置该寄存器为 1 后，待机模式的唤醒时间会加长。
7	EWUP7	RW	0	使能 WKUP7 引脚（PB4）。 此位由软件置位和清零。 0：WKUP7 引脚用于通用 I/O。WKUP7 引脚上的事件不会从待机模式唤醒器件。 1：WKUP7 引脚用于从待机模式唤醒，唤醒沿由 WP7[1:0] 决定。
6	EWUP6	RW	0	使能 WKUP6 引脚（PA13）。 此位由软件置位和清零。 0：WKUP6 引脚用于通用 I/O。WKUP6 引脚上的事件不会从待机模式唤醒器件。 1：WKUP6 引脚用于从待机模式唤醒，唤醒沿由 WP6[1:0] 决定。
5	EWUP5	RW	0	使能 WKUP5 引脚（PA5）。 此位由软件置位和清零。 0：WKUP5 引脚用于通用 I/O。 WKUP5 引脚上的事件不会从待机模式唤醒器件。 1：WKUP5 引脚用于从待机模式唤醒，唤醒沿由 WP5[1:0] 决定。
4	EWUP4	RW	0	使能 WKUP4 引脚（PA1）。 此位由软件置位和清零。 0：WKUP4 引脚用于通用 I/O。WKUP4 引脚上的事件不会从待机模式唤醒器件。

				1：WKUP4 引脚用于从待机模式唤醒，唤醒沿由 WP4[1:0]决定。
3	EWUP3	RW	0	使能 WKUP3 引脚（PD4）。 此位由软件置位和清零。 0： WKUP3 引脚用于通用 I/O。WKUP3 引脚上的事件不会从待机模式唤醒器件。 1： WKUP3 引脚用于从待机模式唤醒，唤醒沿由 WP3[1:0]决定。
2	EWUP2	RW	0	使能 WKUP2 引脚（PC10）。 此位由软件置位和清零。 0：WKUP2 引脚用于通用 I/O。WKUP2 引脚上的事件不会从待机模式唤醒器件。 1：WKUP2 引脚用于从待机模式唤醒，唤醒沿由 WP2[1:0]决定。
1	EWUP1	RW	0	使能 WKUP1 引脚（PC5）。 此位由软件置位和清零。 0：WKUP1 引脚用于通用 I/O。WKUP1 引脚上的事件不会从待机模式唤醒器件。 1：WKUP1 引脚用于从待机模式唤醒，唤醒沿由 WP1[1:0]决定。
0	EWUP0	RW	0	使能 WKUP0 引脚（PB10）。 此位由软件置位和清零。 0：WKUP0 引脚用于通用 I/O。WKUP 引脚上的事件不会从待机模式唤醒器件。 1：WKUP0 引脚用于从待机模式唤醒，唤醒沿由 WP0[1:0]决定。

5.4.4. Power 控制寄存器 4 (PWR_CR4)

偏移地址: 0x0C

复位值: 0x0000 0000(reset by POR)

该寄存器退出待机模式后不会被复位。

该寄存器仅由 POR 复位，BOR 和系统复位不会复位该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WP7[1:0]		WP6[1:0]		WP5[1:0]		WP4[1:0]		WP3[1:0]		WP2[1:0]		WP1[1:0]		WP0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:14	WP7[1:0]	RW	2’b0	WKUP7 引脚（PB4）极性选择 该寄存器由软件置位和清零。 00：WKUP7 引脚上升沿唤醒 01：WKUP7 引脚下降沿唤醒 10，11：保留
13:12	WP6[1:0]	RW	2’b0	WKUP6 引脚（PA13）极性选择

				该寄存器由软件置位和清零。 00：WKUP6 引脚上升沿唤醒 01：WKUP6 引脚下降沿唤醒 10，11：保留
11:10	WP5[1:0]	RW	2’b0	WKUP5 引脚（PA5）极性选择 该寄存器由软件置位和清零。 00：WKUP5 引脚上升沿唤醒 01：WKUP5 引脚下降沿唤醒 10，11：保留
9:8	WP4[1:0]	RW	2’b0	WKUP4 引脚（PA1）极性选择 该寄存器由软件置位和清零。 00：WKUP4 引脚上升沿唤醒 01：WKUP4 引脚下降沿唤醒 10，11：保留
7:6	WP3[1:0]	RW	2’b0	WKUP3 引脚（PD4）极性选择 该寄存器由软件置位和清零。 00：WKUP3 引脚上升沿唤醒 01：WKUP3 引脚下降沿唤醒 10，11：保留
5:4	WP2[1:0]	RW	2’b0	WKUP2 引脚（PC10）极性选择 该寄存器由软件置位和清零。 00：WKUP2 引脚上升沿唤醒 01：WKUP2 引脚下降沿唤醒 10，11：保留
3:2	WP1[1:0]	RW	2’b0	WKUP1 引脚（PC5）极性选择 该寄存器由软件置位和清零。 00：WKUP1 引脚上升沿唤醒 01：WKUP1 引脚下降沿唤醒 10，11：保留
1:0	WP0[1:0]	RW	2’b0	WKUP0 引脚（PB10）极性选择 该寄存器由软件置位和清零。 00：WKUP0 引脚上升沿唤醒 01：WKUP0 引脚下降沿唤醒 10，11：保留

5.4.5. Power 状态寄存器 (PWR_SR)

偏移地址: 0x10

复位值: 0x0000 4200（reset by POR)

该寄存器退出待机模式后不会被复位。

该寄存器除 PVDO 外，其他位仅由 POR 复位，BOR 和系统复位不会复位。

该寄存器 PVDO 位由电源复位（POR/BOR）、系统复位及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDO.
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	VCC_PORF	PINRST_WUF	IWDGRST_WUF	LPRUN_RUN_SWF	Res.	MR_RDY	SBF	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0

	R	R	R	R		R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	PVDO	R	0	PVD 检测结果输出。 0：被检测的 V_{CC} 超出 PVD 选择的比较阈值 1：被检测的 V_{CC} 低于 PVD 选择的比较阈值
15	Reserved	-	-	保留
14	VCC_PORF	R	1	V_{CC} 上电复位标志。 0：未发生 V_{CC} 上电复位 1：发生 V_{CC} 上电复位
13	PINRST_WUF	R	0	PIN 复位唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CPINRSTWUF 位置 1 清零。 0：未发生唤醒事件 1：发生唤醒事件
12	IWDGRST_WUF	R	0	IWDG 复位唤醒待机模式标志。 该位通过硬件置 1,清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CIWDGRSTWUF 位置 1 清零。 0：未发生唤醒事件 1：发生唤醒事件
11	LPRUN_RUN_SWF	R	0	LPRUN 和 RUN 之间模式切换标志。置为 1 表明为 RUN 模式。在切换过程中，值的变化如下所述： --LPRUN 切换为 RUN 时： 0：系统从 LPRUN 模式切换为 RUN 模式进行中 1：系统从 LPRUN 模式切换为 RUN 模式完成 --RUN 切换为 LPRUN 时： 0：系统从 RUN 模式切换为 LPRUN 模式完成 1：系统从 RUN 模式切换为 LPRUN 模式进行中
10	Reserved	-	-	保留
9	MR_RDY	R	1	用于指示 MR 工作状态， 1：MR 模式稳定工作； 0：MR 模式关闭； 该寄存器当 VR_MODE 为 MR 模式时，该寄存器会保持置位状态。切换为其他模式（LPR 或者 DLPR）时该寄存器为清零状态。
8	SBF	R	0	待机模式标志。 此位由硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CSBF 位置 1 来实现。 0：系统未进入待机模式 1：系统已进入待机模式
7	WUF7	R	0	WKUP7 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF7 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
6	WUF6	R	0	WKUP6 唤醒待机模式标志。

				该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF6 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
5	WUF5	R	0	WKUP5 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF5 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
4	WUF4	R	0	WKUP4 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF4 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
3	WUF3	R	0	WKUP3 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF3 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
2	WUF2	R	0	WKUP2 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF2 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
1	WUF1	R	0	WKUP1 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF1 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件
0	WUF0	R	0	WKUP0 唤醒待机模式标志。 该位通过硬件置 1，清零则只能通过 POR 或将 PWR_SCR 寄存器中的 CWUF0 位置 1 清零。 0：未发生唤醒事件 1：收到唤醒事件

5.4.6. Power 状态清零寄存器 (PWR_SCR)

偏移地址: 0x14

复位值: 0x0000 0000

该寄存器仅由 POR 复位，BOR 和系统复位不会复位该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	CVCCPORF	CPINRSTWUF	CIWDGRSTWUF	Res.	Res.	Res.	CSBF	CWUF7	CWUF6	CWUF5	CWUF4	CWUF3	CWUF2	CWUF1	CWUF0
	W	W	W				W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	CVCCPORF	W	0	VCC 上电复位标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 VCC_PORF 信号
13	CPINRSTWUF	W	0	PINRST_WUF 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 PINRST_WUF 信号
12	CIWDGRSTWUF	W	0	IWDGRST_WUF 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 IWDGRST_WUF 信号
11:9	Reserved	-	-	保留
8	CSBF	W	0	SBF 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 SBF 信号
7	CWUF7	W	0	WKUP7 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF7 信号
6	CWUF6	W	0	WKUP6 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF6 信号
5	CWUF5	W	0	WKUP5 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF5 信号
4	CWUF4	W	0	WKUP4 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF4 信号
3	CWUF3	W	0	WKUP3 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF3 信号
2	CWUF2	W	0	WKUP2 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF2 信号
1	CWUF1	W	0	WKUP1 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF1 信号
0	CWUF0	W	0	WKUP0 唤醒标志清除信号。 通过设置该位为 1，清除 PWR_SR 中的 WUF0 信号

PWR_PUCRA 寄存器到 PWR_PDCRD 寄存器，是控制在待机模式时各端口的上下拉，要使能该部分寄存器，需要首先配置 PWR_CR3.APC=1。

配置该部分寄存器，需要注意：

1. 如果 PD13 作为复位引脚使用，进入待机模式前，需要配置 PWR_PUCRD.PU13=1；

5.4.7. Power PortA 上拉控制寄存器 (PWR_PUCRA)

偏移地址: 0x20

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	PU13	PU12	PU11	PU10	PU9	Res.	Res.	Res.	PU5	Res	PU3	PU2	PU1	PU0
		RW	RW	RW	RW	RW				RW		RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	PUy	RW	0	端口 A 上拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PA[y] 的上拉。如果相应的 PDy 位也置 1，则不激活上拉。 注： 1. PA4 作为 SWDIO，由 GPIO 配置为上拉，不受该寄存器配置 2. PA6~8 为 V_{BKP} 域 IO，不受该寄存器控制

5.4.8. Power PortA 下拉控制寄存器 (PWR_PDCRA)

偏移地址: 0x24

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	PD13	PD12	PD11	PD10	PD9	Res.	Res.	Res.	Res.	PD4	PD3	PD2	PD1	PD0
		RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	PDy	RW	0	端口 A 下拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PA[y] 的下拉。 注： 1. PA5 作为 SWCLK，由 GPIO 配置为下拉，不受该寄存器配置 2. PA6~8 为 V_{BKP} 域 IO，不受该寄存器控制

5.4.9. Power PortB 上拉控制寄存器 (PWR_PUCRB)

偏移地址: 0x28

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PUy	RW	0	端口 B 上拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PB[y] 的上拉。如果相应的 PDy 位也置 1，则不激活上拉。

5.4.10. Power PortB 下拉控制寄存器 (PWR_PDCRB)

偏移地址: 0x2C

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PDy	RW	0	端口 B 下拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PB[y] 的下拉。

5.4.11. Power PortC 上拉控制寄存器 (PWR_PUCRC)

偏移地址: 0x30

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PUy	RW	0	端口 C 上拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PC[y] 的上拉。如果相应的 PDy 位也置 1，则不激活上拉。

5.4.12. Power PortC 下拉控制寄存器 (PWR_PDCRC)

偏移地址: 0x34

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留

15:0	PDy	RW	0	端口 C 下拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PC[y] 的下拉。

5.4.13. Power PortD 上拉控制寄存器 (PWR_PUCRD)

偏移地址: 0x38

复位值: 0x0000 2000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PU15	PU14	PU13	PU12	PU11	PU10	PU9	PU8	PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PUy	RW	0x2000	端口 D 上拉控制位（y=0~13） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PD[y] 的上拉。如果相应的 PDy 位也置 1，则不激活上拉。 **注：**在进入待机模式前，需要配置复位 PIN（PD13）上拉。

5.4.14. Power PortD 下拉控制寄存器 (PWR_PDCRD)

偏移地址: 0x3C

复位值: 0x0000 0000

该寄存器在退出待机模式后不会被复位。

该寄存器由电源复位（POR/BOR）及 PWR 软复位（RCC_APBRSTR1.PWRRST）复位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD15	PD14	PD13	PD12	PD11	PD10	PD9	PD8	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PDy	RW	0	端口 D 下拉控制位（y=0~15） 在 PWR_CR3 寄存器中的 APC 位置 1 时将此位置 1，会激活 PD[y] 的下拉。

6.复位和时钟系统(RCC)

6.1. 复位

芯片内设计两种复位，分别是：电源复位和系统复位。

6.1.1. 电源复位

电源复位在以下几种情况下产生：

- 上下电复位（POR/PDR）
- 欠压复位（BOR）

除 RTC 域的寄存器，以及 PWR 控制器的部分寄存器外，上电复位和欠压复位会将其它所有寄存器设为其复位值。

退出待机模式时，V_{DDD_CORE}、V_{DDD_PERI} 和 V_{DDD_SRAM2}，V_{DDD_SRAM1} 域的所有寄存器都设置为其复位值。 其他域的寄存器（RTC、TAMP、WKUP、IWDG 以及待机模式控制）不受影响。

6.1.2. 系统复位

系统复位把大部分寄存器置成复位值，一些特殊寄存器，如复位标识位寄存器，不会被系统复位。

当产生以下事件时，产生系统复位：

- NRST 引脚复位
- 窗口看门狗复位(WWDG)
- 独立看门狗复位(IWDG)
- SYSRESETREQ 软件复位
- 选项字节加载复位（OBL）
- 低功耗模式安全复位

通过检查 RCC_CSR 寄存器的复位标识位，可以识别复位源。

注：在后续模块描述中出现的“系统复位”字样，包含了上述“电源复位”和“系统复位”。

6.1.2.1. PIN 引脚复位 (外部复位)

通过选项字节(NRST_MODE 位)加载，NRST 引脚可以被配置成下述模式（具体配置参见选项字节描述）：

- 复位输入

在该模式下，在 NRST 引脚上任何有效的复位信号被传递到内部逻辑，但是芯片内部产生的复位在 NRST 引脚上不输出。

在该配置模式下，GPIO 的 PD13 功能无效。

对 NRST 引脚有滤毛刺处理，设计保证 NRST 最小要满足 40us 宽度，少于该宽度的信号将被滤除。对于该滤毛刺电路采用 HSI_10M 进行计数处理，默认该时钟源关闭，NRST 的低电平打开该时钟，高电平则关闭该时钟。

- GPIO

在该模式下，该引脚可以用作标准的 GPIO，即 PD13。引脚上的复位功能无效。只能从器件内部复位源实现复位，并且不会将复位传送到此引脚。

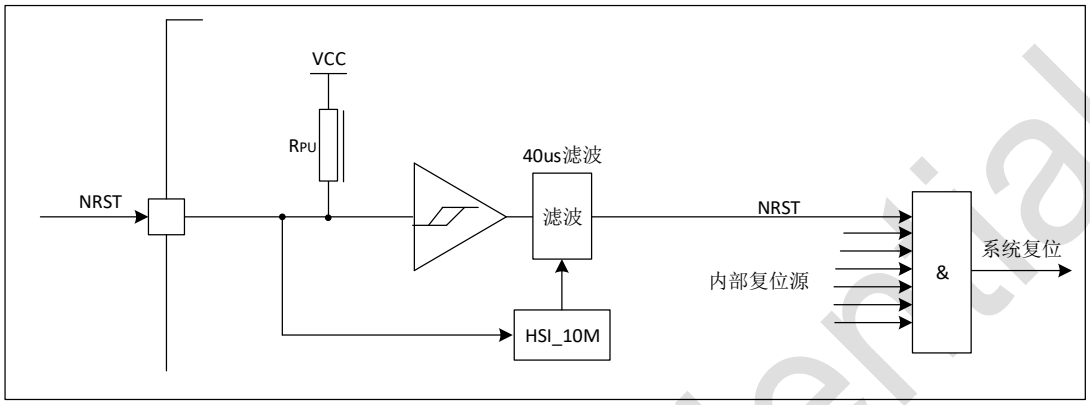


图 6-1 复位电路

6.1.2.2. 看门狗复位

参见“独立看门狗（IWDG）”和“系统窗口看门狗（WWDG）”章节。

6.1.2.3. 软件复位

通过置位 ARM Cortex®-M0+的中断和复位控制寄存器的 SYSRESETREQ 位，可实现软件复位。

6.1.2.4. 选项字节加载复位

软件通过配置 FLASH_CR.OBL_LAUNCH=1，产生选项字节加载复位，从而启动选项字节加载。

6.1.2.5. 低功耗模式安全复位

为了防止关键应用错误地进入低功耗模式，提供了三种低功耗模式安全复位。如果在选项字节中使能，则在下列情况下会产生这种复位：

- 进入待机模式

可通过清零用户选项字节中的 nRST_STDBY 位来使能此类复位。使能后，只要成功执行进入待机模式序列，器件就将复位，而非进入待机模式。

- 进入停止模式

可通过清零用户选项字节中的 nRST_STOP 位来使能此类复位。使能后，只要成功执行进入停止模式序列，器件就将复位，而非进入停止模式。

6.1.3. RTC 域复位

RTC 域具有两个特殊复位。只要发生以下事件之一，就会产生 RTC 域复位：

- 软件复位，通过将 RTC 域控制寄存器（RCC_BDCR）中的 BDRST 位置 1 触发。
- 在电源 V_{CC} 和 V_{BAT} 都已掉电后，其中任何一个又再上电。

RTC 域复位仅影响 LSE 振荡器、RTC、备份寄存器和 RCC RTC 域控制寄存器。

6.2. 时钟

6.2.1. 时钟源

本产品系统时钟源包括 HSI，MSI，HSE，PLL，LSI，LSE。还包括用做引脚复位滤波，以及 Flash 低功耗运行（系统时钟为 LSI/LSE）时控制 FM_SLEEP 信号的 HSI10M。

6.2.1.1. HSI 时钟

HSI 时钟信号是从内部 RC 振荡器生成的。频率包括 8MHz，16MHz，24MHz，48MHz 以及 64MHz。

HSI RC 振荡器的优点是成本较低（无需使用外部组件）。它还比 HSE 晶振具有更短的启动时间。然而，即使在校准之后，其精度也不如使用参考频率的振荡器（如石英晶振或陶瓷谐振器）精确。

从停止模式唤醒后，可选择源自 HSI 的 HSISYS 时钟作为系统时钟。它还可作为备份时钟源（辅助时钟）使用，以防 HSE 晶振发生故障。请参见“时钟安全系统（CSS）”。

时钟控制寄存器（RCC_CR）中的 HSIRDY 标志指示 HSI RC 是否稳定。在启动时，硬件将该位置 1 后，HSI 才可以使用。

HSI RC 可通过时钟控制寄存器（RCC_CR）中的 HSION 位打开或关闭。上电复位后默认打开。

HSI 校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同。为补偿这种差异，每个器件在出厂时都经过校准，各条件下的精度参见数据手册。

POR/BOR 复位后，HSI 频率及对应校准值将加载到内部时钟源校准寄存器（RCC_ICSCR）的 {HSI_FS_OPCR[2:0],HSI_TRIMCR[12:0] 位中。

应用中的电压或温度变化可能会影响 RC 振荡器的 HSI 频率。软件可通过配置内部时钟源校准寄存器（RCC_ICSCR）中的 HSI_TRIMCR[8:0]位进行调整。

有关如何测量 HSI 频率变化的更多详细信息，请参见“TIM16/TIM17 测量内外部时钟”。

6.2.1.2. MSI 时钟

MSI 2 MHz 时钟信号是从内部 RC 振荡器生成的。MSI RC 振荡器的优势在于可提供一个低成本（无外部元件）低功耗的时钟源。此振荡器用作 LPRUN 低功耗模式下的时钟，以降低功耗。

时钟控制寄存器（RCC_CR）中的 MSIRDY 标志指示 MSI RC 是否稳定。在启动时，硬件将该位置 1 后，MSI 才可以使用。如在时钟中断使能寄存器（RCC_CIER）中使能 MSIRDYIE，则可产生中断。

MSI RC 可通过时钟控制寄存器 (RCC_CR) 中的 MSION 位打开或关闭。

MSI 校准

因为生产工艺不同，不同芯片的 RC 振荡器频率也不同。为补偿这种差异，每个器件在出厂时都经过校准，各条件下的精度参见数据手册。

POR/BOR 复位后，MSI 校准值将加载给 MSI 模拟电路。

6.2.1.3. HSE 时钟

高速外部时钟信号 (HSE) 有 2 个时钟源：

- HSE 外部晶振/陶瓷谐振器
- HSE 用户外部时钟

谐振器和负载电容必须尽可能地靠近振荡器的引脚，以尽量减小输出失真和起振稳定时间。

负载电容值必须根据所选振荡器的不同做适当调整。

外部晶振/陶瓷谐振器（HSE 晶振）

4 ~ 32 MHz 外部振荡器的优点是精度非常高。

有关详细信息，请参见数据手册的电气特性部分。

时钟控制寄存器 (RCC_CR) 中的 HSERDY 标志指示 HSE 振荡器是否稳定。在启动时，硬件将此位置 1 后，此时钟才可以使用。如在时钟中断使能寄存器 (RCC_CIER) 中使能 HSERDYIE，则可产生中断。

使用外部时钟源配置寄存器 (RCC_ECSCR) 中的 HSE_DRIVER[1:0] 位，可在运行运行前配置晶振驱动强度，以实现稳健性、短启动时间和低功耗之间的最佳平衡。

HSE 晶振可通过时钟控制寄存器 (RCC_CR) 中的 HSEON 位打开或关闭。

外部源（ HSE 旁路）

在此模式下，必须提供外部时钟源。最高频率不超过 32 MHz。通过将时钟控制寄存器(RCC_CR) 的 HSEBYP 和 HSEON 位置 1 来选择此模式。在提供有 OSC_IN 和 OSC_OUT 引脚的器件上，必须使用占空比为 ~40-60% 的外部时钟信号（方波、正弦波或三角波）来驱动 OSC_IN 引脚，具体取决于频率（请参见数据手册）。 OSC_OUT 引脚可用作 GPIO。

有关引脚可用性的详细信息，请参见数据手册中的引脚排列部分。

6.2.1.4. PLL 时钟

内部 PLL 将在其输入上获取的基于 HSI 或 HSE 的时钟频率进行倍频，以产生独立的时钟输出。参考时钟允许的输入频率范围参见数据手册。

要修改 PLL 配置，请按照以下步骤操作：

1. 通过将时钟控制寄存器 (RCC_CR) 中的 PLLON 位设置为 0 禁止 PLL。
2. 等待至 PLLRDY 清零。 PLL 现已完全停止。
3. 根据需要更改参考时钟，以及倍频系数。
4. 通过将 PLLON 位置 1 再次使能 PLL。

如果已在时钟中断使能寄存器（RCC_CIER）中使能中断，则 PLL 就绪时便可产生中断。

6.2.1.5. LSI 时钟

内部低速 RC 时钟 LSI 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）和 RTC 等模块使用。时钟频率为 32.768 kHz。有关详细信息，请参见数据手册的电气特性部分。

LSI RC 可通过 RTC 域控制寄存器（RCC_BDCR）中的 LSION 位打开或关闭。

RTC 域控制寄存器（RCC_BDCR)中的 LSIRDY 标志指示 LSI 振荡器是否稳定。在启动时，硬件将此位置 1 后，此时钟才可以使用。如在时钟中断使能寄存器（RCC_CIER）中使能中断，则可产生中断。

6.2.1.6. LSE 时钟

LSE 晶振是 32.768 kHz 晶振或陶瓷谐振器，可作为实时时钟（RTC）的时钟源来提供时钟/日历或其他定时功能，具有功耗低且精度高的优点。

LSE 晶振通过 RTC 域控制寄存器（RCC_BDCR）中的 LSEEN 位打开和关闭。使用 RTC 域控制寄存器（RCC_BDCR）中的 LSE_DRIVER [1:0] 位，可在运行前配置晶振驱动强度，以实现稳健性、短启动时间和低功耗之间的最佳平衡。

RTC 域控制寄存器（RCC_BDCR）的 LSERDY 标志指示 LSE 晶振是否稳定。在启动时，硬件将此位置 1 后， LSE 晶振输出时钟信号才可以使用。如在时钟中断使能寄存器（RCC_CIER)中使能中断，则可产生中断。

外部源（ LSE 旁路）

在此模式下，必须提供外部时钟源。最高频率不超过 1 MHz。通过将 RTC 域控制寄存器(RCC_BDCR)的 LSEBYP 位置 1，以及将 RTC 域控制寄存器（RCC_BDCR)中 LSEEN 配置为使能来选择此模式。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 OSC32_IN 引脚。

6.2.1.7. HSI10M 时钟

该时钟作为低精度低功耗时钟，用作 NRST 引脚的滤波计数，以及 Flash 低功耗运行（系统时钟为 LSI/LSE）时控制 FM_SLEEP 信号。该时钟由硬件使能和关闭。

6.2.2. 时钟结构

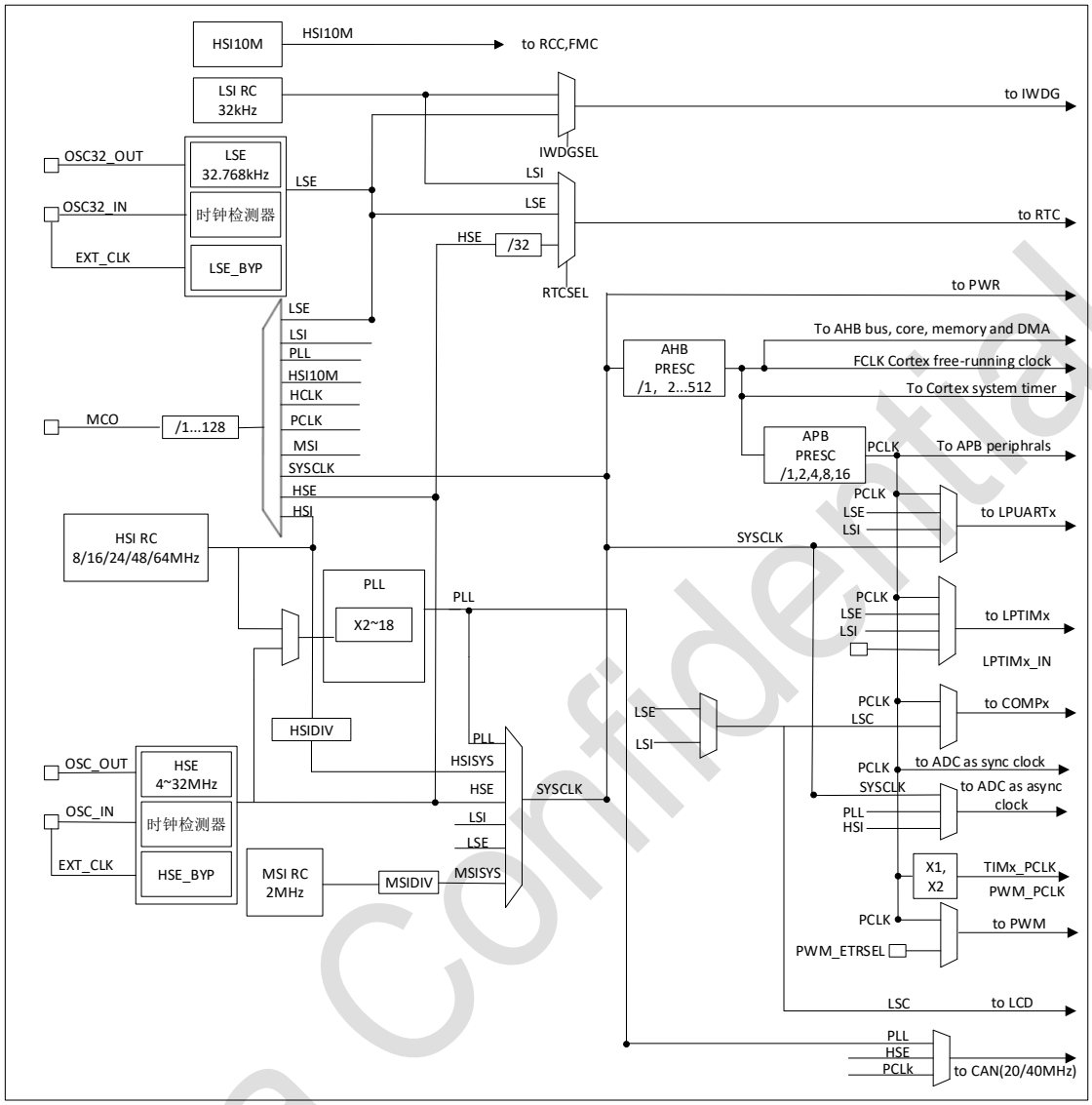


图 6-2 时钟结构

6.2.3. 系统时钟(SYSCLK)选择

可选择以下时钟之一作为系统时钟（SYSCLK):

- LSI
- LSE
- HSISYS
- HSE
- PLL

系统时钟最大频率为 72 MHz。系统复位后，选择从 HSI 振荡器得到的 HSISYS 时钟作为系统时钟。在直接使用 HSI 或者通过 PLL 使用时钟源来作为系统时钟时，以及在切换低功耗模式时，该时钟源无法停止。

只有在目标时钟源已就绪（时钟在启动延迟或 PLL 锁相后稳定时），才可从一个时钟源切换到另一个。在切换成功后（时钟配置寄存器 RCC_CFGR 中 SWS 已经更新为目标时钟源），才能关闭原先时钟源（如果需要）。

6.2.4. 时钟安全系统(CSS)

6.2.4.1. HSE CSS

通过配置 RCC_CR.HSECSSON，HSE 时钟安全系统可以被软件激活。在这种情况下，HSE 启动后，时钟检测功能被打开。当 HSE 被关闭后，时钟检测功能被关闭。

如果在 HSE 上发现时钟故障，HSE 会被自动关闭，时钟故障事件被送给 TIM1（高级定时器）、TIM15/TIM16/TIM17（通用定时器）和 PWM 的刹车输入端，并产生中断通知软件该故障（时钟安全系统中断 CSSI），进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex®-M0+的 NMI（不可屏蔽中断）异常向量。

注意：一旦 CSS 被使能，并且如果 HSE 时钟故障，就会产生 CSS 中断，并自动产生一个 NMI。该 NMI 将不断执行，直到 CSS 中断挂起位被清除。因此，在 NMI 的处理程序中必须通过设置时钟中断寄存器（RCC_CICR）里的 CSSC 位来清除 CSS 中断。

如果 HSE 被直接或者间接的用作系统时钟（间接的意思是：被作为 PLL 的输入端，并且 PLL 被用作系统时钟），时钟故障将导致系统时钟自动切换到 HSI，同时关闭 HSE。如果时钟故障时，HSE 是 PLL 的输入时钟，PLL 也将被关闭。

6.2.4.2. LSE CSS

通过配置 RCC_BDCR.LSECSSON，LSE 时钟安全系统可以被软件激活。在这种情况下，LSE 启动后，时钟检测功能被打开。当 LSE 被关闭后，时钟检测功能被关闭。

如果在 LSE 上发现时钟故障，LSE 会被自动关闭，时钟故障事件被送给 TIM1（高级 timer）和 TIM16/TIM17（通用 timer）的刹车输入端，并产生中断通知软件该故障（Clock Security System Interrupt CSSI），进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex®-M0+的 NMI（Non-maskable interrupt）异常向量。

注意: 一旦 LSECSS 被使能，并且如果 LSE 时钟故障，就会产生 CSS 中断，并自动产生一个 NMI。该 NMI 将不断执行，直到 LSECSS 中断挂起位被清除。因此，在 NMI 的处理程序中必须通过设置时钟中断寄存器（RCC_CICR）里的 LSECSSC 位来清除 LSECSS 中断。

如果 LSE 被用作系统时钟，时钟故障将导致系统时钟自动切换到 LSI，同时关闭 LSE。

6.2.5. TIM16/TIM17 测量内外部时钟

由于温度、电压、工艺及生产等因素，导致内部时钟源（如 HSI、LSI 等）的频率出现漂移现象。因此，需要根据系统外部工作环境的变化采取一些必要的手段来对频率的漂移进行校准。

对时钟漂移处理的基本思路是：在系统外部环境发生变化时，通过动态实时度量芯片的内部时钟，检测发现问题。然后，通过软件微调内部时钟 trimming 参数，从而实现动态校准的目的。

所有板上时钟源的频率都可通过对 TIM16 和 TIM17 通道 1 的输入捕获进行间接测量。

TIM16

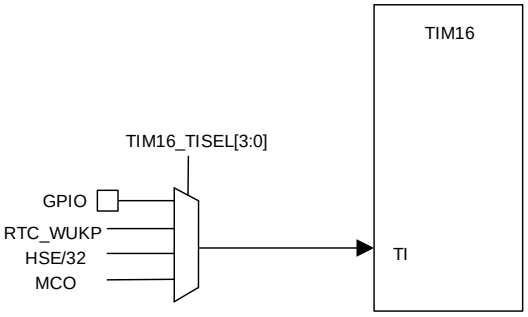


图 6-3 通过 TIM16 采样测量时钟

通过配置 TIM16_TISEL 寄存器的 TI1SEL[3:0] 位域，为 TIM16 的输入捕获通道 1 选择的时钟可以是以下时钟之一：

- GPIO（请参见器件数据手册中的复用功能映射）
- LSI 时钟
- MCO 输出时钟
- RTC 唤醒中断信号

最后一个选项需要使能 RTC 中断。MCO 输出时钟由时钟配置寄存器 (RCC_CFGR) 的 MCOSEL[3:0] 位域控制。可以为 MCO 引脚选择所有时钟源。

TIM17

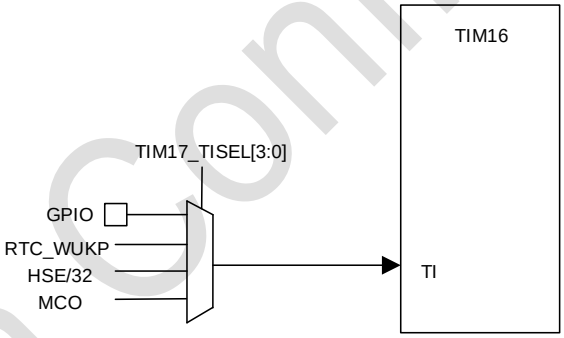


图 6-4 通过 TIM17 采样测量时钟

通过配置 TIM16_TISEL 寄存器的 TI1SEL[3:0] 位域，为 TIM16 的输入捕获通道 1 选择的时钟可以是以下时钟之一：

- GPIO（请参见器件数据手册中的复用功能映射）
- LSI 时钟
- MCO 输出时钟
- RTC 唤醒中断信号

最后一个选项需要使能 RTC 中断。MCO 输出时钟由时钟配置寄存器 (RCC_CFGR) 的 MCOSEL[3:0] 位域控制。可以为 MCO 引脚选择所有时钟源。

6.2.5.1. HSI 校准

HSI 时钟校准分为两个部分：时钟检测和时钟校准。

时钟检测

对于 TIM15、TIM16 和 TIM17，将 LSE 连接到通道 1 输入捕获的主要目的是为了能够精确测量 HSISYS 时钟（源自 HSI，被选作系统时钟）。对 LSE 时钟（时间参考）的连续边沿之间的 HSISYS 时钟脉冲进行计数即可测量 HSISYS（和 HSI）时钟周期。这种测量可以确定 HSI 振荡器频率，其精度与 LSE 振荡器使用的 32.768 kHz 石英晶振的精度几乎相同（通常为几十 ppm）。然后可对 HSI 振荡器进行调整以补偿与目标频率之间由于制造、工艺、温度和/或电压变化所导致的偏差。

HSI 振荡器设有针对此目的的专用校准位，且支持用户访问。其基本原理是基于相对的测量（例如，HSISYS/LSE 比）：因此，测量精度与两个时钟源之比紧密相关。增大比值可提高测量精度。

HSE 时钟（ 32 分频）由 HSE 振荡器生成，用作时间参考，是另一种达到出色 HSI 频率测量精度的最佳方法。建议在没有 LSE 时钟的情况下使用。

为进一步提高 HSI 振荡器校准精度，建议单独使用或组合使用以下措施来提高频率测量精度：

- 将 HSISYS 分频器值设置为 1，使 HSISYS 频率等于 HSI 频率
- 对多次连续测量的结果求平均值
- 使用定时器的输入捕获预分频器值（最多每八个周期捕获一次）
- 使用 RTC 和 RTC 唤醒中断信号的 LSE 时钟作为时间参考

最后一点显著增加了 HSI 时钟脉冲计数的参考周期，从而提高了单次测量的精度。为此，必须使能 RTC 唤醒中断。

时钟校准

一旦检测到 HSI 时钟异常，通过中断，通知软件处理。软件通过微调内部时钟 trimming 参数（通过更新 RCC_ICSCR.HSI_TRIMCR[8:0]实现），从而实现动态校准的目的。

注意：在微调 HSI 时钟校准值时，要与 RCC_ICSCR.HSI_FS_OPCR 的频率值匹配。

6.2.5.2. LSI 校准

与 HSI 一样，LSI 的时钟频率也会受到电压、温度、工艺及生产的影响而产生漂移。LSI 的校准采用与其频率相差较大的 HSE 或 HSI 来进行校准，校准方法与 HSI 类似。

LSI 的校准是 TIM16/TIM17 的输入捕获选择 LSI 时钟，HSE 作为系统时钟源。LSI 信号连续边沿之间的 HSE 时钟脉冲数由 TIM16/ TIM17 计数，表示 LSI 时钟周期。

原理上，仍然是相对频率的关系，即 HSE/LSI 的频率比：校准精度与该密切相关，比率值越大，度量的效果越好。

6.2.6. 时钟输出

为了方便板级应用，节省 BOM 成本，以及调试等的需求，需要芯片提供时钟输出功能。即把下表的 MCO 信号（并分频）通过 GPIO 的复用功能实现时钟输出功能。

表 6-1 输出时钟选择

时钟源/内部时钟	MCO 可输出的时钟源
HSI	√
HSE	√
PLL	√
LSE	√

LSI	√
HSI_10M	√
SYSCLK	√
HCLK	√
PCLK	√

注意：当对 MCO 时钟源进行切换，以及选择 GPIO AF 功能为 MCO 的起始阶段，MCO 可能会产生毛刺，需要避开该段时间。

6.2.7. 模块时钟类型

时钟类型：

- SYSCLK：RCC_CFGR.SW 配置时钟源的时钟，最高 72 MHz
- HCLK：SYSCLK 经过 HPRE 分频的时钟，最高 72 MHz
- PCLK：HCLK 经过 PPRE 分频的时钟，最高 72 MHz
- RTCCLK：RTC 模块内核时钟，时钟源由 RCC_BDCR.RTCSEL 寄存器配置，最高 1 MHz，常用 32kHz
- PWM_CLK：PWM 模块内核时钟，时钟源由 PWM 模块内部选择
- LPTIM_CLK：LPTIM 模块内核时钟，时钟源由 RCC_CCIPR.LPTIM*SEL 寄存器配置，最高 72MHz，常用 32 kHz
- IWDG_CLK：IWDG 模块内核时钟，时钟源由 RCC_CCIPR.IWDGSEL 寄存器配置
- LPUART_CLK：LPUART 模块内核时钟，时钟源由 RCC_CCIPR.LPUART*SEL 寄存器配置，最高 72 MHz
- CMP_CLK：COMP1/COMP2 模块内核时钟，时钟源由 RCC_CCIPR.COMP*SEL 寄存器配置，最高 72 MHz，常用 32 kHz
- PVD_CLK：PVD 模块内核时钟，时钟源由 RCC_CCIPR.PVDSEL 寄存器配置，最高 72 MHz，常用 32 kHz
- ADC_CLK：SARADC 模块内核时钟，时钟源及分频系数由 ADC_CCR.CKMODE 和 ADC_CCR.PRESC 寄存器配置
- LCD_CLK：LCD 模块内核时钟，时钟源由 RCC_BDCR.LSCSEL 寄存器配置选择 LSI 或者 LSE，频率为 32 kHz

6.3. RCC 寄存器

该模块的寄存器可以用字(32bit)、半字（16bit）和字节（8bit）访问。

6.3.1. RCC 时钟控制寄存器（RCC_CR）

偏移地址:0x00

复位值:0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	PLLRDY	PLLON	Res.	Res.	Res.	Res.	HSECSSON	HSEBYP	HSERDY	HSEON
						R	RW					RS	RW	R	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	HSIDIV[2:0]			HSIRDY	MSIRDY	HSION	MSION	HSIKERON	Res.	Res.	Res.	MSIDIV[2:0]		
		RW	RW	RW	R	R	RW	RW	RW				RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	PLLRDY	R	0	PLL 时钟就绪标志。 硬件置位，表明 PLL 时钟已锁定。 0：PLL 未锁定 1：PLL 已锁定
24	PLLON	RW	0	PLL 使能。 由软件置位和清零。 当系统进入任意一种停止模式或待机模式时，硬件会清零该位。当 PLL 时钟已被用作（SWS 为 PLL）或将要被用作系统时钟（SW 选择 PLL）时，该位不能被清零。 0：PLL 关闭 1：PLL 使能
23:20	Reserved	-	-	保留
19	HSECSSON	RS	0	时钟安全系统使能。 软件置位使能时钟安全系统。当该位被置位，如果 HSE 就绪时，硬件会进行时钟检测，当发现时钟失效后，硬件禁止时钟检测。 该位只能被置位，清零只能靠复位。 0：时钟安全系统关闭（时钟检测关闭） 1：时钟安全系统开启（如果 HSE 就绪，时钟检测开启，否则关闭）
18	HSEBYP	RW	0	旁路 HSE 外接晶振，选择管脚输入时钟。 软件置位和清零，用于用外部时钟旁路振荡器。外部时钟必须用 HSEON 使能。HSEBYP 位仅当 HSE 外接晶振禁止时才被置位。 0：不旁路 HSE 外接晶振 1：旁路 HSE 外接晶振，外接管脚输入时钟
17	HSERDY	R	0	HSE 时钟就绪标志位 硬件置位，表明 HSE 就绪。 0：HSE 未就绪 1：HSE 已就绪 注：当 HSEON 清零后，HSERDY 立即清零
16	HSEON	RW	0	HSE 时钟使能 软件可置位和清零。进入停止模式，硬件清零该位。如果 HSE 被直接或者间接用作系统时钟，则该位不能被复位。 0：HSE 关闭 1：HSE 开启
15:14	Reserved	-	-	保留
13:11	HSIDIV[2:0]	RW	0	HSI 时钟分频系数。 软件控制这些位设定 HSI 的分频系数，产生 HSISYS 时钟 000：1 001：2

				010：4 011：8 100：16 101：32 110：64 111：128
10	HSIRDY	R	0	HSI 时钟就绪标志。 硬件置位表明 HSI 振荡器稳定。该位只有当 HSION=1 时才有效。 0：HSI 振荡器未就绪； 1：HSI 振荡器就绪； 当 HSION 清零后，HSIRDY 将在 3 个 HSI 之后拉低。
9	MSIRDY	R	0	MSI 时钟就绪标志。 硬件置位表明 MSI 振荡器稳定。该位只有当 MSION=1 时才有效。 0：MSI 振荡器未就绪； 1：MSI 振荡器就绪； 当 MSION 清零后，MSIRDY 将在 3 个 MSI 之后拉低。
8	HSION	RW	1	HSI 时钟使能位。软件可以置位和清零该位。 当进入任意一种停止模式或者待机模式时，硬件清零该位，停止 HSI。 当 HSI 被直接或者间接用作系统时钟（也当退出停止或者待机模式，或者 HSE 作为系统时钟，并产生失效时），硬件会置位该寄存器。 0：HSI 关闭 1：HSI 开启 注：当 HSION 清零后，HSI 还会额外产生 3 个，HSIRDY 将在 3 个 HSI 后清零。
7	MSION	RW	0	MSI 时钟使能位。软件可以置位和清零该位。 当进入 LPRUN 模式，且软件选择 MSI 作为系统时钟时，需要使能该寄存器。 当从 LPRUN 模式进入某种停止模式，当从该停止模式唤醒后需要进入 LPRUN 模式，根据软件的配置，如果选择的是 MSI 时钟，则硬件在唤醒后会使能 MSI。 0：MSI 关闭 1：MSI 开启 注：当 MSION 清零后，MSI 还会额外产生 3 个，MSIRDY 将在 3 个 HSI 后清零。
6	HSIKERON	RW	0	HSI 为了外设内核总是启用使能。 当 I2C 作为停止模式的唤醒源时，为了保证通信速度，该寄存器可以置位，保持 HSI 在停止模式下打开，可避免由于 HSI 启动时间而降低通信速度。此位对 HSION 值没有影响。 0：对 HSI 振荡器没有影响。 1：HSI 振荡器即使在停止模式下也被强制使能。
5:3	Reserved	-	-	保留
2:0	MSIDIV[2:0]	RW	0	MSI 时钟分频系数。

				软件控制这些位设定 MSI 的分频系数，产生 MSISYS 时钟作为 LPRUN 模式的系统时钟 000：1 001：2 010：4 011：8 100：16 101：32 110：64 111：128

6.3.2. RCC 内部时钟源校准寄存器（RCC_ICSCR）

偏移地址:0x04

复位值:0x0000 1100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSI_FS_OPCR[2:0]			HSI_TRIMCR[12:0]												
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved		-	保留
15:13	HSI_FS_OPCR	RW	3’b000	HSI 频率选择： 000: 8 MHz 001: 8 MHz 010: 16 MHz 011: 保留 100: 24 MHz 101: 48 MHz 110: 64 MHz 111: 8 MHz 上电后，默认选择 8 MHz，在选项字节加载完成后，硬件通过加载 Flash trimming 页数据来更新 8 MHz 对应的 Trim 值。在系统复位后，硬件同样切换成 8 MHz。
12:0	HSI_TRIMCR[12:0]	RW	13’h1100	HSI 时钟频率校准值。 上电后硬件用 HSI 8 MHz 的默认校准值，在上电加载 Trimming 时会把出厂信息（存放在 0x1FFF 1F2C）写入该寄存器中。 软件通过读出存放在信息区相应地址的数据，写入该寄存器，实现 HSI 特定输出频率下的校准。 保存在 Flash 的如下地址内： 64 MHz 校准值存放地址：0x1FFF 1E18 48 MHz 校准值存放地址：0x1FFF 1E14 24 MHz 校准值存放地址：0x1FFF 1E10 16 MHz 校准值存放地址：0x1FFF 1E08 8 MHz 校准值存放地址：0x1FFF 1E04 通过向该寄存器写入校准值后，也可以该值为中心值，修改该寄存器数值，每增（减）1，则 HSI 的输出频率增（减）约 0.1%。 HSI_TRIM[12:9]:粗调位，HSI_TRIM[8:0]细调位。在 trim 时需要分两段控制，如粗调位不变改变细调位。

6.3.3. RCC 时钟配置寄存器（RCC_CFGR）

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCOPRE[3:0]				MCOSEL[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	PPRE[2:0]			HPRE[3:0]				Res.	Res.	SWS[2:0]			SW[2:0]		
	RW	RW	RW	RW	RW	RW	RW			R	R	R	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	MCOPRE[2:0]	RW	0	MCO（Microcontroller clock output）分频系数。软件控制这些位，设置 MCO 输出的分频系数: 0000: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 32 1101: 64 1110: 128 1111: 256 推荐在 MCO 输出使能前，设置这些位。
27:24	MCOSEL[3:0]	RW	0	MCO 选择 0000:无时钟, MCO 输出禁止 0001: SYSCLK 0010: HSI10M 0011: HSI 0100: HSE 0101: PLL CLK 0110: LSI 0111: LSE 1000: HCLK 1001: PCLK 1010: MSI 1011: LCD_HSI10M 其它: 无时钟 注：在时钟启动或者切换阶段可能会出现输出时钟不完整的情况。
23:15	Reserved	-	-	保留
14:12	PPRE[2:0]	RW	0	该位由软件控制。为了产生 PCLK 时钟，它设置 HCLK 的分频系数如下: 0xx：1 100：2 101：4 110：8 111：16
11:8	HPRE[3:0]	RW	0	AHB 时钟分频系数。 软件控制该位。为了产生 HCLK 时钟，它设置 SYSCLK 的分频系数如下: 0000: 1 1000: 2 1001: 4 1010: 8

				1011: 16 1100: 64 1101: 128 1110: 256 1111: 512 其它: 保留 为了保证系统正常工作，需要根据 VR 电源情况配置合适频率。 注：建议逐级切换分频系数。
7:6	保留	-	-	保留
5:3	SWS[2:0]	R	0	系统时钟切换状态位 这些位由硬件控制，表明当前哪个时钟源被用作系统时钟： 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE 101: MSI 其它: 保留
2:0	SW[2:0]	RW	0	系统时钟源选择位。 这些位由软件和硬件控制，用来选择系统时钟： 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE 101: MSI 其它: 保留 硬件配置为 HSISYS 的情况包括： 1）系统从停止模式退出 2）软件配置 001(HSE)，出现 HSE 故障（HSE 为系统时钟源，或者 HSE 作为 PLL 输入，PLL 作为系统时钟源）

6.3.4. RCC PLL 配置寄存器（RCC_PLLCFGR)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLLMUL[4:0]					Res.	PLLSRC
									RW	RW	RW	RW	RW		RW

注意：该寄存器需要在 PLL 使能(PLLON=1)前配置，如果需要改变配置，必须先关闭 PLL，重新配置后再使能 PLL。

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6:2	PLLMUL[4:0]	RW	5'b0	PLL 倍频系数 00000：x2 00001：x3 00010：x4 00011：x5 00100：x6 00101：x7 00110：x8 00111：x9 01000：x10

				01001：x11 01010：x12 01011：x13 01100：x14 01101：x15 01110：x16 01111：x17 10000：x18 其他：保留（可写）
1	Reserved	-	-	保留
0	PLLSRC	RW	0	PLL 时钟源选择： 0：HSI 1：HSE

6.3.5. RCC 外部时钟源配置寄存器（RCC_ECSCR)

偏移地址:0x10

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSE_FILT_ENB	HSE_RDYSEL[1:0]		HSE_DRIVER[1:0]	
											RW	RW	RW	RW	RW

注意：该寄存器需要在 HSE 使能(HSEON=1)前配置，如果需要改变配置，必须先关闭 HSE，重新配置后再使能 HSE。

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	HSE_FILT_ENB	RW	0	模拟 HSE 滤波控制。 0：使能滤波 1：禁止滤波
3:2	HSE_RDYSEL[1:0]	RW	2’b0	HSE 稳定时间选择。 HSEBYP=0: 00：4096 个 HSE 时钟； 01：2048 个 HSE 时钟； 10：8192 个 HSE 时钟； 11：不计稳定时间，直接输出； HSEBYP=1: 00：2048 个 HSE 时钟； 01：1024 个 HSE 时钟； 10：4096 个 HSE 时钟； 11：不计稳定时间，直接输出；
1:0	HSE_DRV[1:0]	RW	2’b0	HSE 驱动能力配置。 00: gm 3.5mA/V 01: gm 5.0mA/V 10: gm 7.5mA/V 11: gm 10mA/V

6.3.6. RCC 时钟中断使能寄存器（RCC_CIER)

偏移地址:0x18

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLL RDYIE	HSE RDYIE	HSI RDYIE	MSIRDYIE	LSE RDYIE	LSI RDYIE
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	PLLRDYIE	RW	0	PLL 就绪中断使能。 0：禁止 1：使能
4	HSERDYIE	RW	0	HSE 时钟就绪中断使能。 0：禁止 1：使能
3	HSIRDYIE	RW	0	HSI 时钟就绪中断使能。 0：禁止 1：使能
2	MSIRDYIE	RW	0	MSI 时钟就绪中断使能。 0：禁止 1：使能
1	LSERDYIE	RW	0	LSE 时钟就绪中断使能。 0：禁止 1：使能
0	LSIRDYIE	RW	0	LSI 时钟就绪中断使能。 0：禁止 1：使能

6.3.7. RCC 时钟中断标志寄存器（RCC_CIFR）

偏移地址:0x1C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	LSE CSSF	CSSF	Res.	Res.	PLL RDYF	HSE RDYF	HSI RDYF	MSI RDYF	LSE RDYF	LSI RDYF
						R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	LSECSSF	R	0	LSE 时钟安全系统（CSS）中断标志。 当硬件检测 LSE 振荡器检出故障时置位该寄存器。软件通过写 LSECSSC 寄存器 1 清零该位。 0：LSE 时钟故障中断未产生 1：LSE 时钟故障中断产生
8	CSSF	R	0	HSE 时钟安全系统中断标识位。

				当硬件检测 HSE 振荡器检出故障时置位该寄存器。软件通过写 CSSC 寄存器 1 清零该位。 0：HSE 时钟故障中断未产生 1：HSE 时钟故障中断产生
7:6	Reserved	-	-	保留
5	PLLRDYF	R	0	PLL 就绪中断标识位 当 PLL 锁定并且 PLLRDYIE 位置 1 时，硬件置位。软件通过置位 PLLRDYC 位，清零该位。 0：PLL 锁定引起的时钟锁定就绪中断未产生 1：产生 PLL 锁定引起的时钟就绪中断
4	HSERDYF	R	0	HSE 就绪中断标识位 当 HSE 稳定并且 HSERDYIE 使能，该位由硬件置位。软件通过置位 HSERDYC 位，清零该位。 0：无由 HSE 引起的时钟就绪中断 1：有由 HSE 引起的时钟就绪中断
3	HSIRDYF	R	0	HSI 就绪中断标识位 当 HSI 稳定并且 HSIRDYIE 使能，该位由硬件置位。软件通过置位 HSIRDYC 位，清零该位。 0：无由 HSI 引起的时钟就绪中断 1：有由 HSI 引起的时钟就绪中断
2	MSIRDYF	R	0	MSI 就绪中断标识位 当 MSI 稳定并且 MSIRDYIE 使能，该位由硬件置位。软件通过置位 MSIRDYC 位，清零该位。 0：无由 MSI 引起的时钟就绪中断 1：有由 MSI 引起的时钟就绪中断
1	LSERDYF	R	0	LSE 就绪中断标识位 当 LSE 稳定并且 LSERDYIE 使能，该位由硬件置位。软件通过置位 LSERDYC 位，清零该位。 0：无由 LSE 引起的时钟就绪中断 1：有由 LSE 引起的时钟就绪中断
0	LSIRDYF	R	0	LSI 就绪中断标识位 当 LSI 稳定并且 LSIRDYIE 使能，该位由硬件置位。软件通过置位 LSIRDYC 位，清零该位。 0：无由 LSI 引起的时钟就绪中断 1：有由 LSI 引起的时钟就绪中断

6.3.8. RCC 时钟中断清零寄存器（RCC_CICR）

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	LSECSSC	CSSC	Res.	Res.	PLL RDYC	HSE RDYC	HSI RDYC	MSI RDYC	LSE RDYC	LSI RDYC
						W	W			W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function

31:10	Reserved	-	-	保留
9	LSECSSC	W	0	LSE 时钟安全系统（CSS）中断标志清零。 0：无影响; 1：清零 LSECSSF 标志
8	CSSC	W	0	HSE 时钟安全中断清零位。 0：无影响 1：清除 CSSF 标志位
7:6	Reserved	-	-	保留
5	PLLRDYC	W	0	PLLS 就绪标志清零。 0：无影响 1：清除 PLLRDYF 位
4	HSERDYC	W	0	HSE 就绪标志清零。 0：无影响 1：清除 HSERDYF 位
3	HSIRDYC	W	0	HSI 就绪标志清零。 0：无影响 1：清除 HSIRDYF 位
2	MSIRDYC	W	0	MSI 就绪标志清零。 0：无影响 1：清除 MSIRDYF 位
1	LSERDYC	W	0	LSE 就绪标志清零。 0：无影响 1：清除 LSERDYF 位
0	LSIRDYC	W	0	LSI 就绪标志清零。 0：无影响 1：清除 LSIRDYF 位

6.3.9. RCC I/O 端口复位寄存器（RCC_IOPRSTR）

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	GPIODRST	RW	0	I/O PortD 复位。 0：无影响 1：PortD I/O 复位 由软件置位和清零。
2	GPIOCRST	RW	0	I/O PortC 复位。 0：无影响 1：PortC I/O 复位 由软件置位和清零。

1	GPIOBRST	RW	0	I/O PortB 复位。 0：无影响 1：PortB I/O 复位 由软件置位和清零。
0	GPIOARST	RW	0	I/O PortA 复位。 0：无影响 1：PortA I/O 复位 由软件置位和清零。

6.3.10. RCC AHB 外设复位寄存器（RCC_AHBRSTR）

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRC RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA RST
			RW												RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	CRCRST	RW	0	CRC 模块复位。 0：无影响 1：CRC 模块复位 由软件置位和清零。
11:1	Reserved	-	-	保留
0	DMARST	RW	0	DMA 复位。 0：无影响 1：DMA 模块复位 由软件置位和清零。

注：各个模块中描述的系统复位或者模块复位，无特殊说明都为系统复位+模块软复位。

6.3.11. RCC APB 外设复位寄存器 1（RCC_APBRSTR1）

偏移地址:0x2C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM1 RST	LPTIM2 RST	DACRST	PWRRST	DBGRST	Res.	Res.	Res.	OPARST	I2C2RST	I2CRST	LPUART1 RST	UART2 RST	UART1 RST	USART2 RST	LPUART2 RST
RW	RW	RW	RW	RW					RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2 RST	Res.	Res.	WWDG RST	Res	Res.	Res.	Res.	PWMRST	TIM7RST	TIM6 RST	Res.	Res.	TIM3 RST	TIM2 RST
	RW			RW					RW	RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIM1RST	RW	0	LP Timer1 模块复位。 0：无影响

				1：该模块复位 由软件置位和清零。
30	LPTIM2RST	RW	0	LP Timer2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
29	DACRST	RW	0	DAC 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
28	PWRRST	RW	0	Power 接口模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
27	DBGRST	RW	0	MCU Debug 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
26:24	Reserved	-	-	保留
23	OPARST	RW	0	OPA 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
22	I2C2RST	RW	0	I^2C2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
21	I2C1RST	RW	0	I^2C1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
20	LPUART1RST	RW	0	LPUART1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
19	UART2RST	RW	0	USART2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
18	UART1RST	RW	0	UART1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
17	USART2RST	RW	0	USART2 模块复位。 0：无影响 1：该模块复位

				由软件置位和清零。
16	LPUART2RST	RW	0	LPUART2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
15	Reserved	-	-	保留
14	SPI2RST	RW	0	SPI2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
13:12	Reserved	-	-	保留
11	WWDGRST	RW	0	WWDG 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
10:7	Reserved	-	-	保留
6	PWMRST	RW	0	PWM 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
5	TIM7RST	RW	0	TIM7 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
4	TIM6RST	RW	0	TIM6 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
3:2	Reserved	-	-	保留
1	TIM3RST	RW	0	TIM3 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
0	TIM2RST	RW	0	TIM2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。

6.3.12. RCC APB 外设复位寄存器 2（RCC_APBRSTR2）

偏移地址:0x30

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	VREFBUFRST	Res.	LCDRST	Res.	COMP2RST	COMP1RST	ADCRST	Res.	TIM17RST	TIM16RST	TIM15RST
					RW		RW		RW	RW	RW		RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Res.	USART1 RST	Res.	SPI1 RST	TIM1 RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFG RST
	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	保留
26	VREFBUFRST	RW	0	VREFBUF 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
25	Reserved	-	-	保留
24	LCDRST	RW	0	LCD 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
23	Reserved	-	-	保留
22	COMP2RST	RW	0	COMP2 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
21	COMP1RST	RW	0	COMP1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
20	ADCRST	RW	0	ADC 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
19	Reserved	-	-	保留
18	TIM17RST	RW	0	TIM17 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
17	TIM16RST	RW	0	TIM16 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
16	TIM15RST	RW	0	TIM15 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
15	Reserved	-	-	保留
14	USART1RST	RW	0	USART1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
13	Reserved	-	-	保留

12	SPI1RST	RW	0	SPI1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
11	TIM1RST	RW	0	TIM1 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。
10:1	Reserved	-	-	保留
0	SYSCFGRST	RWs	0	SYSCFG 模块复位。 0：无影响 1：该模块复位 由软件置位和清零。

注：各个模块中描述的系统复位或者模块复位，无特殊说明都为系统复位+模块软复位。

6.3.13. RCC I/O 端口时钟使能寄存器（RCC_IOPENR）

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	GPIODEN	RW	0	I/O PortD 时钟使能。 0：时钟禁止 1：时钟使能 由软件置位和清零。
2	GPIOCEN	RW	0	I/O PortC 时钟使能。 0：时钟禁止 1：时钟使能 由软件置位和清零。
1	GPIOBEN	RW	0	I/O PortB 时钟使能。 0：时钟禁止 1：时钟使能 由软件置位和清零。
0	GPIOAEN	RW	0	I/O PortA 时钟使能。 0：时钟禁止 1：时钟使能 由软件置位和清零。

6.3.14. RCC AHB 外设时钟使能寄存器（RCC_AHBENR）

偏移地址:0x38

复位值:0x0000 0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	CRC EN	Res.	Res.	SRAMEN	FLASH EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA EN
			RW			RW	RW								RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	CRCEN	RW	0	CRC 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
11:10	Reserved	-	-	保留
9	SRAMEN	RW	1	在睡眠模式下，SRAM 的时钟使能控制 0：在睡眠模式该模块时钟关闭 1：在睡眠模式该模块时钟使能 注：该位仅影响睡眠模式该模块的时钟使能，在 RUN 模式，该模块时钟不会关闭。
8	FLASHEN	RW	1	在睡眠模式下，FLASH 的时钟使能控制 0：在睡眠模式该模块时钟关闭 1：在睡眠模式该模块时钟使能 注：该位仅影响睡眠模式该模块的时钟使能，在 RUN 模式，该模块时钟不会关闭。
7:1	Reserved	-	-	保留
0	DMAEN	RW	0	DMA 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。

6.3.15. RCC APB 外设时钟使能寄存器 1（RCC_APBENR1）

偏移地址:0x3C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIM1 EN	LPTIM2 EN	DACEN.	PWR EN	DBG EN	Res.	Res.	Res.	OPA EN	I2C2 EN	I2C1 EN	LPUART1 EN	UART2 EN	UART1 EN	USART2 EN	LPUART2 EN
RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	SPI2 EN	Res.	Res.	WWDG EN	Res.	Res.	Res.	Res.	PWM. EN	TIM7 EN	TIM6 EN	Res.	Res.	TIM3 EN	TIM2 EN
	RW			RW					RW	RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIM1EN	RW	0	LPTIM 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。

30	LPTIM2EN	RW	0	LP Timer2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
29	DACEN	RW	0	DAC 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
28	PWREN	RW	0	Power 接口模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
27	DBGEN	RW	0	Debug 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
26:24	Reserved	-	-	保留
23	OPAEN	RW	0	OPA 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
22	I2C2EN	RW	0	I^2C2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
21	I2C1EN	RW	0	I^2C1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
20	LPUART1EN	RW	0	LP UART1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
19	UART2EN	RW	0	UART2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
18	UART1EN	RW	0	UART1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
17	USART2EN	RW	0	USART2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
16	LPUART2EN	RW	0	LP UART2 模块时钟使能。

				0：禁止 1：使能 由软件置位和清零。
15	Reserved	-	-	保留
14	SPI2EN	RW	0	SPI2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
13:12	Reserved	-	-	保留
11	WWDGEN	RW	0	WWDG 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
10:7	Reserved	-	-	保留
6	PWMEN	RW	0	PWM 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
5	TIM7EN	RW	0	TIM7 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
4	TIM6EN	RW	0	TIM6 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
3:2	Reserved	-	-	保留
1	TIM3EN	RW	0	TIM3 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
0	TIM2EN	RW	0	TIM2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。

6.3.16. RCC APB 外设时钟使能寄存器 2（RCC_APBENR2）

偏移地址:0x40

复位值:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	VERFBUFEN	Res.	LCDEN	Res.	COMP2EN	COMP1EN	ADCEN	Res.	TIM17EN	TIM16EN	TIM15EN
					RW		RW		RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	USART1EN	Res.	SPI1EN	TIM1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGEN
	RW		RW	RW											RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	保留
26	VREFBUFEN	RW	0	V_{REFBUF} 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
25	Reserved	-	-	保留
24	LCDEN	RW	0	LCD 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
23	Reserved	-	-	保留
22	COMP2EN	RW	0	COMP2 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
21	COMP1EN	RW	0	COMP1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
20	ADCEN	RW	0	ADC 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
19	Reserved	-	-	保留
18	TIM17EN	RW	0	TIM17 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
17	TIM16EN	RW	0	TIM16 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
16	TIM15EN	RW	0	TIM15 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
15	Reserved	-	-	保留
14	USART1EN	RW	0	USART1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
13	Reserved	-	-	保留
12	SPI1EN	RW	0	SPI1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。

11	TIM1EN	RW	0	TIM1 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。
10:1	Reserved	-	-	保留
0	SYSCFGEN	RW	1	SYSCFG 模块时钟使能。 0：禁止 1：使能 由软件置位和清零。

6.3.17. RCC 外设时钟配置寄存器（RCC_CCIPR）

偏移地址:0x54

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPTIM2SEL[1:0]		LPTIM1SEL[1:0]	
RW												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LPUART2SEL		LPUART1SEL		Res.	COMP2SEL	COMP1SEL	PVDSEL	Res.	Res.	Res.	Res.	Res.	Res.	TIMCLKCTRL
	RW	RW	RW	RW		RWs	RW	RW							RW

Bit	Name	R/W	Reset Value	Function
31:30	ADCSEL[1:0]	RW	2’b00	ADC 时钟源选择。 00:SYSCLK 01: PLL 10: HSI 11:保留 **注意**：切换时钟源前，需要先关闭 ADC 模块时钟使能，切换完成后再开启时钟使能。
29:20	Reserved	-	-	保留
19:18	LPTIM2SEL[1:0]	RW	2’b00	LPTIM2 内部时钟源选择。 00: PCLK 01: LSI 10: 保留 11: LSE **注意**：切换时钟源前，需要先关闭 LPTIM2 模块时钟使能，切换完成后再开启时钟使能。
17:16	LPTIM1SEL[1:0]	RW	2’b00	LPTIM1 内部时钟源选择。 00: PCLK 01: LSI 10: 保留 11: LSE **注意**：切换时钟源前，需要先关闭 LPTIM1 模块时钟使能，切换完成后再开启时钟使能。
15	Reserved	-	-	保留
14:13	LPUART2SEL[1:0]	RW	0	LPUART2 时钟源选择。 00: PCLK 01: SYSCLK 10: LSI 11: LSE **注意**：切换时钟源前，需要先关闭 LPUART2 模块时钟使能，切换完成后再开启时钟使能。

12:11	LPUART1SEL[1:0]	RW	0	LPUART1 时钟源选择。 00: PCLK 01: SYSCLK 10: LSI 11: LSE **注意**：切换时钟源前，需要先关闭 LPUART1 模块时钟使能，切换完成后再开启时钟使能。
10	Reserved	-	-	保留
9	COMP2SEL	RW	0	COMP2 模块时钟时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） **注意**: 1. 在使能 COMP2_FR.FLTEN 之前先配置选择 LSC 时钟。 2. 切换时钟源前，需要先关闭 COMP2 模块时钟使能，切换完成后再开启时钟使能。
8	COMP1SEL	RW	0	COMP1 模块时钟时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） **注意**: 1. 在使能 COMP2_FR2.FLTEN 之前先配置该寄存器选择时钟。 2. 切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
7	PVDSEL	RW	0	PVD 检测时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） **注意**：切换时钟源前，需要先关闭 COMP1 模块时钟使能，切换完成后再开启时钟使能。
6:1	Reserved	-	-	保留
0	TIMCLKCTRL	RW	0	TIMER PCLK 频率控制。 0：TIMER PCLK 为系统 PCLK*2，但频率不会超过 HCLK 1：TIMER PCLK 为系统 PCLK*1

6.3.18. RCC RTC 域控制寄存器（RCC_BDCR）

偏移地址:0x5C

复位值:0x0200 2800

1. 当 PWR_CR1.DBP 为 1 时，才允许写该寄存器。

2. BDRST，LSE_STARTUP，LSE_DRIVER，LSI_TRIMCR 仅能通过 BPOR 复位。其他寄存器位由 BPOR 和 RTC 软复位复位。

3. 时钟配置需要在时钟关闭时操作，即 LSE_DRIVER [1:0]，LSE_STARTUP[1:0]需要在 LSE 关闭时操作。LSI_TRIMCR[8:0]需要在 LSI 关闭时操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	IWDGSEL	LSE_STARTUP [1:0]		LSIRDY	LSION	LSI_TRIMCR[8:0]									BDRST
	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTCEN	LSCSEL	LSEEN[3:0]				RTCSEL[1:0]		Res.	LSECSSD	LSECSSON	LSE_DRIVER [1:0]		LSEBYP	LSERDY	Res.
RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	R	

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	IWDGSEL	RW	0	IWDG 内部时钟源选择。 0: LSI 1: LSE
29:28	LSE_STARTUP [1:0]	RW	0	LSE 晶振稳定时间选择。 LSEBYP=0： 00：4096 个 LSE 时钟周期 01：2048 个 LSE 时钟周期 10：8192 个 LSE 时钟周期 11：不计稳定时间，直接输出 LSEBYP=1： 00：2048 个 LSE 时钟周期 01：1024 个 LSE 时钟周期 10：4096 个 LSE 时钟周期 11：不计稳定时间，直接输出
27	LSIRDY	R	0	LSI 时钟就绪标志。 0：LSI 未就绪 1：LSI 已就绪
26	LSION	RW	0	LSI 时钟使能。 0：禁止 1：使能 软件置位和清零。在硬件使能 IWDG（通过选项字节）和软件使能 LSECSSON 时，硬件会对该位进行置位。
25:17	LSI_TRIMCR[8:0]	RW	9’h100	LSI 时钟频率校准。 上电后芯片硬件会把出厂信息（存放在 0x1FFF 1F24）写入该寄存器中，使 LSI 可以输出精准的 32.768 kHz 频率。 软件通过对该寄存器值进行改写，每增（减）1，使 LSI 的输出频率增（减）约 0.2%。
16	BDRST	RW	0	RTC 域软件复位。 0：无影响 1：复位
15	RTCEN	RW	0	RTC 时钟使能。 0：禁止 1：使能
14	LSCSEL	RW	0	低速时钟选择。 0: LSI 1: LSE
13:10	LSEEN[3:0]	RW	4’hA	使能寄存器，上电默认 LSE 使能 1010：关闭 LSE 0101：使能 LSE 软件必须写入 1010 才能将 LSE 关闭，软件必须写入 0101 才能启动 LSE.
9:8	RTCSEL[1:0]	RW	0	RTC 时钟源选择。 00: 无时钟 01: LSE 10: LSI 11: HSE 时钟 32 分频

				一旦 RTC 时钟源选择后不能再改变，除非以下情况: ■ RTC 域被复位为 00 ■ 选择为 LSE（LSECSSD=1）但没有 LSE ■ BDRST 软复位将该寄存器复位为 00
7	Reserved	-	-	保留
6	LSECSSD	R	0	LSE CSS（Clock security system)故障检测。 该位由硬件置位，表明 CSS 检测 32.768 kHz 晶振（LSE）检测到故障。 0：未检测到 LSE 故障 1：检测到 LSE 故障
5	LSECSSON	RW	0	LSE CSS 使能 0：禁止 1：使能 必须 LSEON=1 并且 LSERDY=1 后才能使能 LSECSSON。 一旦使能该位，不能再把该位禁止，除非 LSECSSD=1。
4:3	LSE_DRIVER[1:0]	RW	0	LSE 驱动能力设置 00: Icc 200nA, gm 2.5μA/V 01: Icc 300nA, gm 3.5μA/V 10: Icc 600nA, gm 7.5μA/V 11: Icc 1000nA, gm 10μA/V
2	LSEBYP	RW	0	LSE 振荡器旁路 0：未旁路，低速外部时钟选择晶振 1：旁路，低速外部时钟选择外部接口输入时钟 **注意：**只有当外部 32.768 kHz 振荡器禁止（LSEON=0 并且 LSERDY=0）时才能写该位。
1	LSERDY	R	0	LSE 振荡器就绪标志位。 由硬件置位和清零，用于指示外部低速振荡器已就绪（稳定）。 0：未就绪 1：已就绪
0	Reserved	-	-	保留

6.3.19. RCC 控制/状态寄存器 (RCC_CSR)

偏移地址:0x60

复位值:0x0000 0000

Reset by POR(flag bit), reset by system reset excluding NRST(NRST_FLTIDS)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PWRRSTF	PINRSTF	OBLRSTF	Res.	RMVF	SPERSTF	Res.	Res.	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R		RW	R						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	NRST_FLTDIS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							RW								

Bit	Name	R/W	Reset Value	Function
31	LPWRRSTF	R	0	停止和待机低功耗模式复位标志。 RMVF 置 1 会清零该位。

				0：未发生低功耗复位 1：发生低功耗复位
30	WWDGRSTF	R	0	窗口看门狗复位标志。 RMVF 置 1 会清零该位。 0：未发生该复位 1：发生该复位
29	IWDGRSTF	R	0	独立看门狗复位标志。 RMVF 置 1 会清零该位。 0：未发生复位 1：发生复位
28	SFTRSTF	R	0	CPU 软复位标志。 RMVF 置 1 会清零该位。 0：未发生复位 1：发生复位
27	PWRRSTF	R	0	POR/PDR 或 BOR 复位标志。 RMVF 置 1 会清零该位。 0：未发生复位 1：发生复位
26	PINRSTF	R	0	外部 NRST 管脚复位标志。 RMVF 置 1 会清零该位。 0：未发生复位 1：发生复位
25	OBLRSTF	R	0	选项字节加载复位标志。 RMVF 置 1 会清零该位。 0：未发生复位 1：发生复位
24	Reserved	-	-	保留
23	RMVF	RW	0	软件置位后，会清零复位标志。
22	SPERSTF	R	0	SRAM1/SRAM2 奇偶校验错误复位标志。 RMVF 置 1 会清零该位。 **注意：**该寄存器仅当 SYSCFG_SCSR. PERR_RSTEN 配置为 1 时才有效。
21:9	Reserved	-	-	保留
8	NRST_FLTDIS	RW	0	NRST 滤波禁止 0：使能 HSI_10M，且滤波 40us 宽度功能使能 1：滤波功能禁止
7:0	Reserved	-	-	保留

7. 通用 I/O（GPIO）

7.1. GPIO 简介

每个 GPIO 端口有：

4 个 32 位配置寄存器(GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR 和 GPIOx_PUPDR)

2 个 32 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR)

1 个 32 位置位/复位寄存器(GPIOx_BSRR)

1 个 32 位锁定寄存器(GPIOx_LCKR)

2 个复用功能选择寄存器(GPIOx_AFRH 和 GPIOx_AFRL)。

7.2. GPIO 主要特性

- 寄存器支持 Fast IO Port，以及 AHB 总线读写
- 输出状态：推挽或者开漏输出+上拉/下拉
- 数据输出来自数据寄存器(GPIOx_ODR)或者外设（复用功能输出）
- 每个 I/O 可进行速度选择
- 输入状态：浮空，上拉/下拉，模拟
- 数据输入送给输入数据寄存器(GPIOx_IDR)或者外设（复用功能输入）
- 置位/复位寄存器（GPIOx_BSRR），允许对 GPIOx_ODR 按位写
- 锁定机制（GPIOx_LCKR)会冻结 I/O 口配置功能
- 模拟功能
- 复用功能选择寄存器（每个 IO 口最多 16 种复用功能）
- 快速翻转
- 高度灵活的 I/O 多路选择功能，使得 I/O 口作为 GPIO，或者作为各种外设接口功能

7.3. GPIO 功能描述

每个 GPIO 的每个位，可以通过软件编程，进行几种模式的配置：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟
- 开漏输出，带上拉或者下拉
- 推挽输出，带上拉或者下拉
- 带上拉或者下拉的复用功能推挽
- 带上拉或者下拉的复用功能开漏

每个 I/O 口可以自由编程，然而 I/O 端口寄存器必须按 32 位字访问。GPIOx_BSRR 和 GPIOx_BRR 寄存器允许对任何 GPIOx_ODR 寄存器的读/更改的独立访问。这样，在读和更改访问之间产生中断请求时也不会发生危险。

下图为标准 I/O 端口的基本结构。

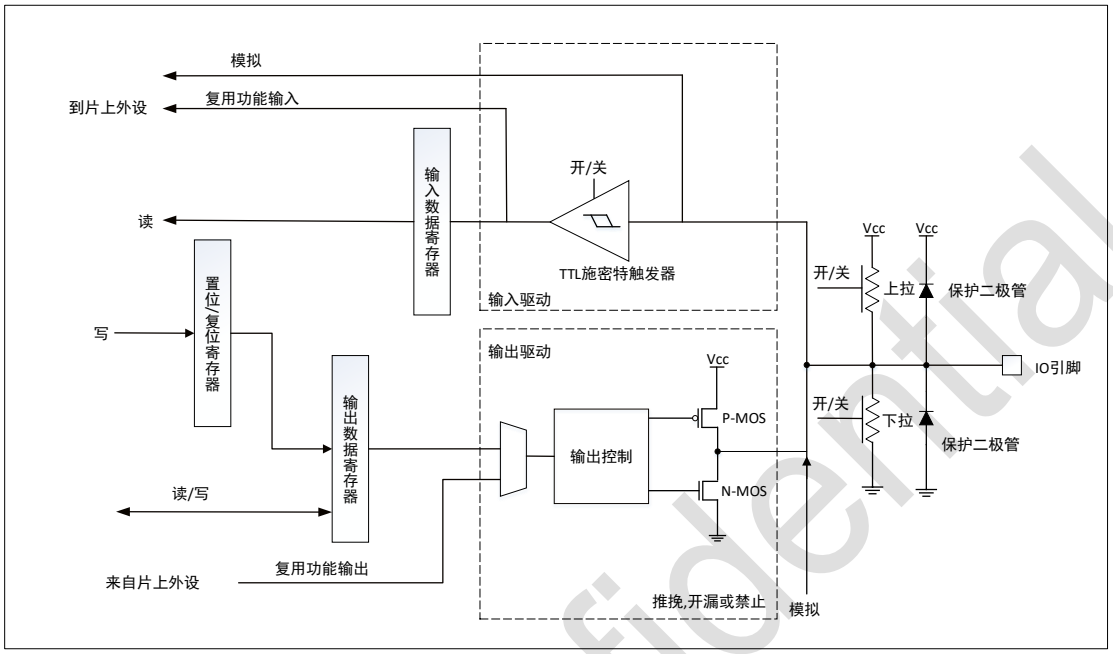


图 7-1 I/O 端口位基本结构

表 7-1 端口位配置表

MODE(i)[1:0]	OTYPE(i)	OSPEED(i)[1:0]		PUPD(i)[1:0]		IO 配置	
01	0	SPEED[1:0]		0	0	GP 输出	PP
	0			0	1	GP 输出	PP+PU
	0			1	0	GP 输出	PP+PD
	0			1	1	保留	
	1			0	0	GP 输出	OD
	1			0	1	GP 输出	OD+PU
	1			1	0	GP 输出	OD+PD
	1			1	1	保留(GP 输出 OD)	
10	0	SPEED[1:0]		0	0	AF	PP
	0			0	1	AF	PP+PU
	0			1	0	AF	PP+PD
	0			1	1	保留	
	1			0	0	AF	OD
	1			0	1	AF	OD+PU
	1			1	0	AF	OD+PD
	1			1	1	保留	
00	x	x	x	0	0	输入	浮空
	x	x	x	0	1	输入	PU
	x	x	x	1	0	输入	PD
	x	x	x	1	1	保留(输入浮空)	

11	x	x	x	0	0	输入/输出	模拟
	x	x	x	0	1	保留（不能配置 PUPD 为上下拉）	
	x	x	x	1	0		
	x	x	x	1	1		

GP =通用, PP =推挽, PU =上拉, PD =下拉, OD =开漏, AF =复用功能.

7.3.1. 通用 I/O(GPIO)

复位期间和复位后，复用功能未被激活，大多数 IO 被配置为模拟模式。

调试引脚默认被置于复用功能上拉或下拉模式，芯片包含两组调试接口的映射，选项配置如下表：

SWD_MODE[1:0]	PA4	PA5	PB6	PB5
00(default)	SWDIO	SWCLK	GPIO	GPIO
01	GPIO	GPIO	SWDIO	SWCLK
10	GPIO	SWCLK	SWDIO	GPIO
11	SWDIO	GPIO	GPIO	SWCLK

PA5 和 PB5 通过 SWD_OPTION 配置为 SWCLK 时，会置于下拉模式；

PA4 和 PB6 通过 SWD_OPTION 配置为 SWDIO 时，会置于上拉模式；

Boot 引脚（PD8）默认置于输入模式，下拉模式；

当管脚被配置为输出时，写入到输出数据寄存器（GPIOx_ODR）的值会输出到 I/O 上。有可能会使用推挽或者开漏模式的输出(低电平是输出的，高电平是高阻态)。

输入数据寄存器（GPIOx_IDR）在每个 AHB 时钟会获取 I/O 脚上的电平。

所有的 GPIO 引脚都有内部的弱上拉和弱下拉电阻，可以通过 GPIOx_PUPDR 寄存器使能或者不使能该功能。

7.3.2. I/O 引脚复用功能

设备 I/O 口通过多路选择器连着板级的外设/模块，一个外设每次可以通过复用功能连着一个 IO 口。这样可以避免同一个 IO 口上的可用外设出现冲突。

每个 I/O 口上的多路选择器多达 16 种复用功能输入（AF0 to AF15），可通过寄存器 GPIOx_AFRL (复用 0 到 7) 和 GPIOx_AFRH (复用 8 到 15)来配置。

- 复位后，多路选择器默认为 AF0。I/O 口的复用功能模式通过寄存器 GPIOx_MODER 配置
- 每个脚的复用功能分布在对应的数据手册上有说明

除了这种灵活的多路选择器架构，每个外设还有复用功能可以分布在不同的 I/O 口上，以便在更小的封装上使用到的外设数量最优化。

用户按照如下说明去配置 IO：

- 调试功能：每次复位后，这些调试功能脚就是默认为调试器立即可用的复用功能脚
- GPIO：在 GPIOx_MODER 将对应 I/O 口配置为输出、输入或者模拟模式
- 外设复用功能：
  - 寄存器 GPIOx_AFRL 或者 GPIOx_AFRH 配置对应的 I/O 为复用功能 x(x=0…15)
  - 寄存器 GPIOx_OTYPER, GPIOx_PUPDR 和 GPIOX_OSPEEDER 分别配置类型，上拉/下拉以及输出速度
  - 寄存器 GPIOx_MODER 是配置对应 I/O 为复用功能

- 额外功能
  - 无论 IO 口配置成任何模式，ADC，LCD 功能均在其模块的寄存器中使能。当 IO 口用做 ADC，LCD 使用时，推荐通过寄存器 GPIOx_MODER 将该口配置为模拟模式；
  - OPA 和 COMP 功能，除了在 OPA 和 COMP 模块的寄存器中使能外，还需要配置 SYSCFG 模块中 ANA2EN 寄存器。当 IO 口用做 OPA 和 COMP 功能使用时，推荐通过寄存器 GPIOx_MODER 将该口配置为模拟模式；
  - 对于 RTC，TAMP, WKUPx 和晶振等额外功能，在相应的 RTC、TAMP、PWR 和 RCC 模块寄存器里配置各自功能。这些配置比标准的 GPIO 配置具有更高优先级。

7.3.3. I/O 端口控制寄存器

每个 GPIO 口有四个 32 位内存映射控制寄存器(GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR 和 GPIOx_PUPDR)，可以配置多达 16 个 I/O 口。寄存器 GPIOx_MODER 用来选择 I/O 模式（输入、输出、复用、模拟）。寄存器 GPIOx_OTYPER 和 GPIOx_OSPEEDR 用来选择输出类型（推挽或开漏）和速度。寄存器 GPIOx_PUPDR 用来选择上拉/下拉。

7.3.4. I/O 数据寄存器

每个 GPIO 有 2 个 16 位内存映射的数据寄存器：输入和输出数据寄存器（GPIOx_IDR 和 GPIOx_ODR）。寄存器 GPIOx_ODR 保存了要输出的数据，可读可写。输入数据寄存器（GPIOx_IDR）用来保存 I/O 口上的电平状态，只读的。

7.3.5. I/O 数据位操作

置位/复位寄存器(GPIOx_BSRR)是一个 32 位寄存器，可以将输出数据寄存器(GPIOx_ODR)的单独位置位和复位。置位/复位寄存器位数是输出寄存器（GPIOx_ODR）的两倍。

GPIOx_ODR 的每一位对应 GPIOx_BSRR 的两个控制位：BS(i) and BR(i)。位 BS(i)置 1 可将 GPIOx_ODR 对应位置 1，位 BR(i)置 1 可将 GPIOx_ODR 对应位清 0。

寄存器 GPIOx_BSRR 任意位写 0 并不影响寄存器 GPIOx_ODR 对应的位。如果 GPIOx_BSRR 对某一位同时清 0 和置 1 操作，置 1 操作具有优先权。

使用寄存器 GPIOx_BSRR 改变寄存器 GPIOx_ODR 的对应位只有一次性的作用，并不会锁定寄存器 GPIOx_ODR 的位。寄存器 GPIOx_ODR 也可以直接访问。寄存器 GPIOx_BSRR 只是提供一种原子位操作处理方式。

当软件编程操作 GPIOx_ODR 的位时没必要关闭中断：在一次 AHB 写访问过程中有可能修改了一个或者多个位。

7.3.6. GPIO 锁定机制

寄存器 GPIOx_LCKR 通过一系列特殊写时序可以冻结 IO 的控制寄存器，包括 GPIOx_MODER,GPIOx_OTYPER,GPIOx_OSPEEDR,GPIOx_PUPDR,GPIOx_AFRL 和 GPIOx_AFRH。

一个特殊写/读时序可以操作寄存器 GPIOx_LCKR。当该寄存器的 Bit16 写入正确的时序，LCKR[15:0]写入值就可以锁定 I/O（在写入时序过程中，LCKR[15:0]写入值保持不变）。当在一个端口位上执行了锁定(LOCK)程序，在下一次 MCU 或者外设复位之前，将不能再更改端口位的配置。GPIOx_LCKR 的每个位冻结控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFRL 和 GPIOx_AFRH）对应的位。

LOCK 时序只能用字（32 位长）访问 GPIOx_LCKR 寄存器，因为 GPIOx_LCKR 位 16 设置的同时也会设置[15:0]位。

7.3.7. I/O 复用功能输入/输出

每个 I/O 有两个寄存器可以用来配置复用功能输入/输出模式。用户根据应用需求将复用功能复用到 IO 口上。

使用寄存器 GPIOx_AFRL 和 GPIOx_AFRH 可以在每一个 GPIO 口多路选择许多可能的外设功能，因此应用让每个 I/O 选择其中一种功能。AF 选择信号对于复用功能输入和复用功能输出都是相同的，对于给定 I/O 的复用功能输入/输出可以选择单独的通道。

7.3.8. 外部中断线/唤醒线

所有端口都有外部中断能力。为了使用外部中断线，端口必须禁止配置成模拟模式或者晶振管脚，并且需要触发输入使能。

7.3.9. 输入配置

当 I/O 口配置为输入：

- 输出缓冲器不使能
- 施密特触发器输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 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得到 I/O 状态

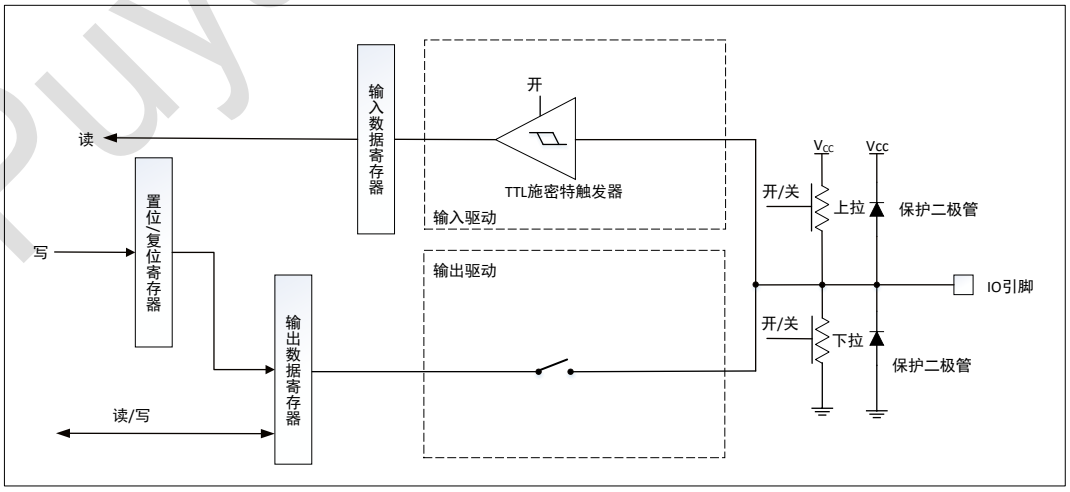


图 7-2 输入浮空/上拉/下拉配置

7.3.10. 输出配置

当 I/O 端口被配置为输出时：

- 输出缓冲器被激活
  - 开漏模式：输出寄存器上的’0’激活 N-MOS，而输出寄存器上的’1’将端口置于高阻状态(PMOS 从不被激活)。
  - 推挽模式：输出寄存器上的’0’激活 N-MOS，而输出寄存器上的’1’将激活 P-MOS。
- 施密特触发输入被激活
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 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得到 I/O 状态
- 对输出数据寄存器的读访问得到最后一次写的值

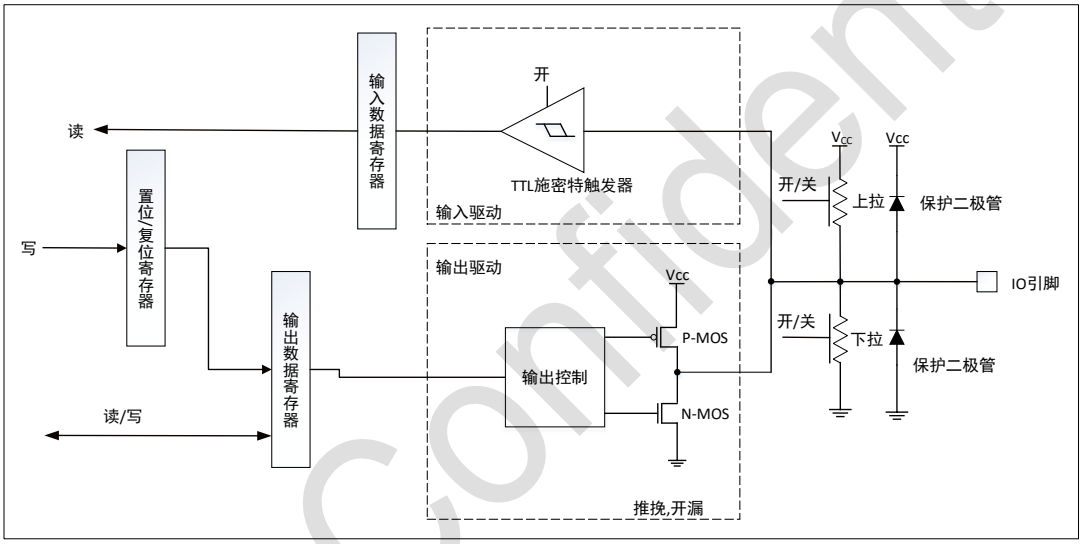


图 7-3 输出配置

7.3.11. 复用功能配置

当 I/O 端口被配置为复用功能时：

- 在开漏或推挽式配置中，输出缓冲器被打开
- 内置外设的信号驱动输出缓冲器(复用功能输出)
- 施密特触发输入被激活
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 在每个 AHB 时钟周期，出现在 I/O 脚上的数据被采样到输入数据寄存器
- 读输入数据寄存器时可得到 I/O 口状态

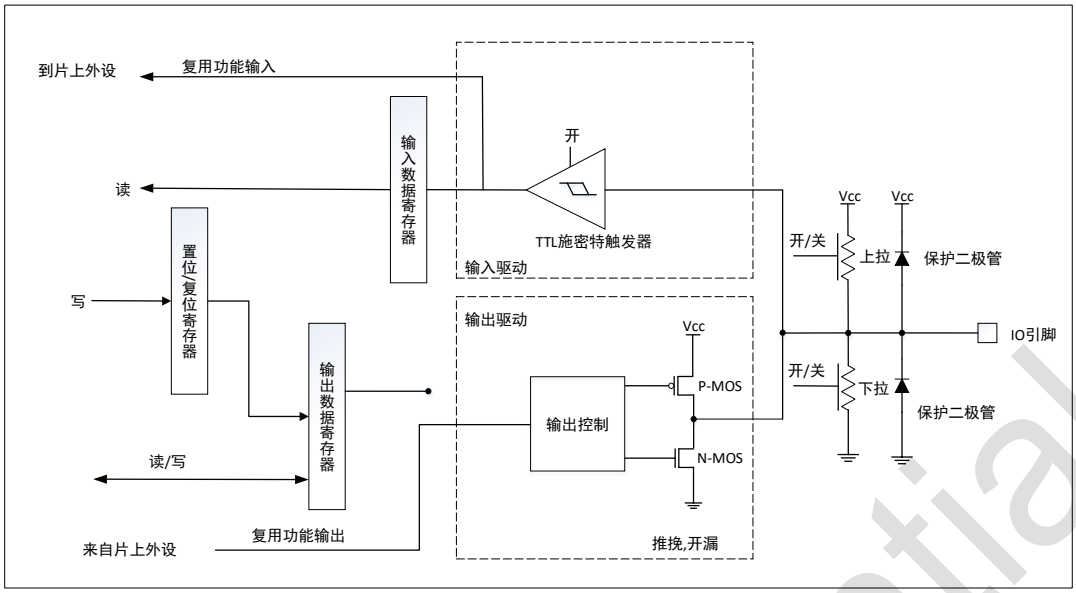


图 7-4 复用功能配置

7.3.12. 模拟配置

当 I/O 端口被配置为模拟配置时:

- 输出缓冲器被禁止;
- 禁止施密特触发输入，I/O 引脚的每个模拟输入的功耗变为零。施密特触发输出值被强置 为'0';
- 弱上拉和下拉电阻被禁止（需要软件设定）;
- 读取输入数据寄存器时数值为'0'。

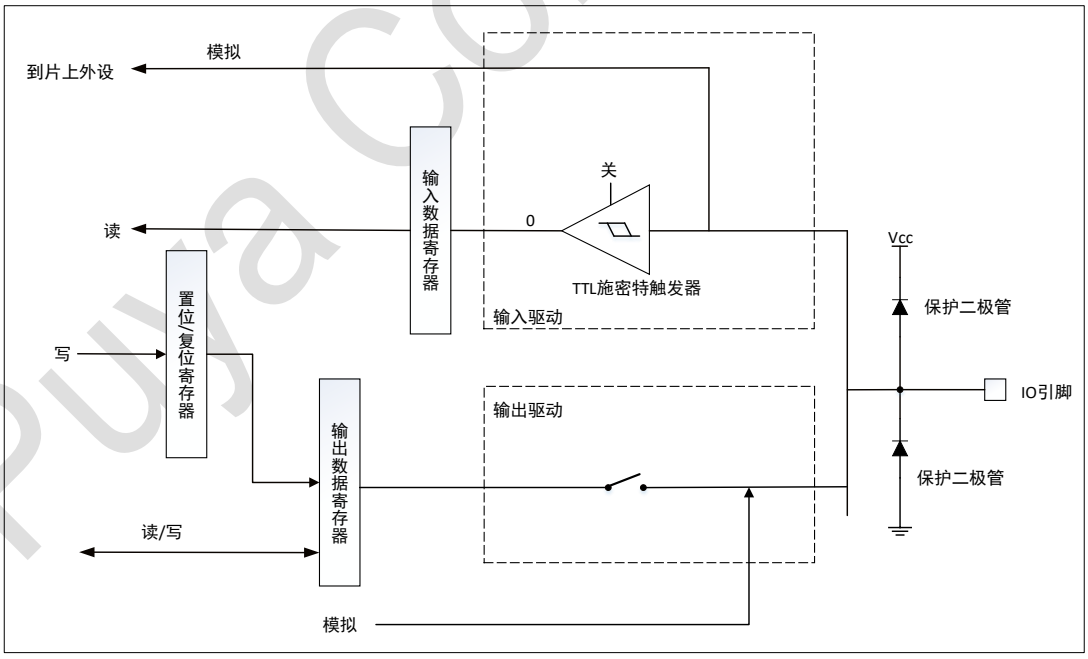


图 7-5 高阻抗模拟配置

7.3.13. 将 HSE/LSE 引脚用作 GPIO

当 HSE 或 LSE 功能被关闭（复位后的默认），相应的管脚可以当作正常的 GPIO 用。

当 HSE 或 LSE 功能打开（RCC_CR 寄存器器中设置 HSEON，RCC_BDCR 寄存器中配置 LSEEN），需要软件配置对应的端口为模拟端口。

当晶振配置为用户外部时钟模式，只有 OSC_IN 或者 OSC32_IN 保留给时钟输入，而 OSC_OUT 或 OSC32_OUT 脚仍然可以用作正常 GPIO。

7.3.14. 备份电源域 GPIO 引脚

当内核电源域断电（器件进入待机模式时），PA6/PA7/PA8 GPIO 功能会丢失。在这种情况下，如果 GPIO 没有被配置 RTC 相关复用功能，那么这些引脚是在模拟输入模式。

7.4. GPIO 寄存器

7.4.1. GPIO 端口模式寄存器 (GPIOx_MODER) (x=A,B,C,D)

偏移地址: 0x00

复位值:

- GPIOA:
  --SWD_MODE=00: 0xFFFF FAFF
  --SWD_MODE=01: 0xFFFF FFFF
  -- SWD_MODE=10: 0xFFFF FBFF
  -- SWD_MODE=11: 0xFFFF FEFF
- GPIOB：
  --SWD_MODE=00: 0xFFFF FFFF
  --SWD_MODE=01: 0xFFFF EBFF
  --SWD_MODE=10: 0xFFFF EFFF
  --SWD_MODE=11: 0xFFFF FBFF
- 0xFFFF FFFF for GPIOC
- 0x0FFC FFFF For GPIOD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	MODEy[1:0]	RW		y = 15..0 软件通过这些位配置相应的 I/O 模式 00: 输入模式 01: 通用输出模式 10: 复用功能模式 11: 模拟模式(大部分 IO 的复位状态)

7.4.2. GPIO 端口输出类型寄存器 (GPIOx_OTYPER) (x = A,B,C,D)

偏移地址: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	OTy[1:0]	RW	0	软件配置 I/O 的输出类型 0: 推挽输出 (复位状态) 1: 开漏输出

7.4.3. GPIO 端口输出驱动寄存器 (GPIOx_OSPEEDR) (x = A,B,C, D)

偏移地址: 0x08

复位值:

- GPIOA:
  --SWD_MODE=00: 0x0000 0100
  --SWD_MODE=01: 0x0000 0000
  -- SWD_MODE=10: 0x0000 0000
  -- SWD_MODE=11: 0x0000 0100
- GPIOB:
  --SWD_MODE=00: 0x0000 1000
  --SWD_MODE=01: 0x0000 0000
  -- SWD_MODE=10: 0x0000 1000
  -- SWD_MODE=11: 0x0000 0000
- 其他端口: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	OSPEEDy[1:0]	RW		Y = 15..0 软件配置 IO 口的输出速度 00：非常低 01：低速 10：高速 11：非常高

7.4.4. GPIO 端口上下拉寄存器 (GPIOx_PUPDR) (x = A,B,C, D)

偏移地址: 0x0C

复位值:

- GPIOA：

  --SWD_MODE=00: 0x0000 0900

  --SWD_MODE=01: 0x0000 0000

  --SWD_MODE=10: 0x0000 0800

  --SWD_MODE=11: 0x0000 0100

- GPIOB：

  --SWD_MODE=00: 0x0000 0000

  --SWD_MODE=01: 0x0000 1800

  --SWD_MODE=10: 0x0000 1000

  --SWD_MODE=11: 0x0000 0800

- 0x0000 0000(端口 C)
- 0x0002 0000(端口 D)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPD7[1:0]		PUPD6[1:0]		PUPD5[1:0]		PUPD4[1:0]		PUPD3[1:0]		PUPD2[1:0]		PUPD1[1:0]		PUPD0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	PUPDy [1:0]	RW		Y = 15..0 软件配置 I/O 口上拉或者下拉 00: 无上下拉 01: 上拉 10: 下拉 11: 保留（无上下拉）

7.4.5. GPIO 端口输入数据寄存器 (GPIOx_IDR) (x = A,B,C, D)

偏移地址: 0x10

复位值: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	IDy	R		y = 15..0 这是只读的，读出值位对应 I/O 口的状态。

7.4.6. GPIO 端口输出数据寄存器 (GPIOx_ODR) (x = A,B,C, D)

偏移地址: 0x14

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ODy[1:0]	RW	0	y = 15..0 软件可读可写。 说明：通过写入 GPIOx_BSRR 或 GPIOx_BRR 寄存器. (x=A,B,C,D)，可以分别对各个 OD 位进行独立的置位/复位。

7.4.7. GPIO 端口置位/复位寄存器 (GPIOx_BSRR) (x = A,B,C, D)

偏移地址: 0x18

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	BRy	W	0	y = 15..0 软件可写，读取这些位的返回值为 0。 0: 不会对相应的 ODy 位执行任何操作 1: 清除对应的 ODy 位 注意：如果同时设置 BSy 和 BRy 的对应位，BSy 位起作用。
15:0	BSy	W	0	y = 15..0 软件可写，读取这些位的返回值为 0。 0: 不会对相应的 ODy 位执行任何操作 1: 置位对应的 ODy 位

7.4.8. GPIO 端口配置锁定寄存器 (GPIOx_LCKR) (x = A,B,C, D)

当执行正确的写序列设置了 bit16（LCKK）时，该寄存器用来锁定端口位的配置。bit[15:0]用于锁定 GPIO 端口的配置。在写序列期间，不能改变 LCKR[15:0]。当对相应的端口执行了 LOCK 序列后，在下次系统复位前将不能再更改端口位的配置。

注：特殊写时序用来写 GPIOx_LCKR 寄存器。在锁定时序中仅仅只有字访问可以被执行。

每个锁定位冻结一个特定的配置寄存器（控制寄存器和复用功能寄存器）

偏移地址: 0x1C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	LCKK	RW	0	该位可随时读出，它只能通过锁键写入序列修改 0: 端口配置锁键位未激活

				1: 端口配置锁键位被激活，下次系统复位前 GPIOx_LCKR 寄存器被锁定。 锁键的写入时序: 写{LCKR[16]=1,LCKR[15:0]} 写{LCKR[16]=0,LCKR[15:0]} 写{LCKR[16]=1,LCKR[15:0]} 读 LCKR 读 LCKR[16]为 1. 最后一个读可省略，但该读操作可以用来确认锁键已被激活。 **注意:** 1. 在操作锁键的写入时序时，不能改变 LCK[15:0]的值。 2. 锁键时序的任何错误都会终止锁键被激活。 3. 在任一端口位上的第一个锁定序列之后，对 LCKK 位的任何读访问都将返回“1”，直到下一次系统复位或 GPIO 外设软复位。
15:0	LCKy	RW	0	y = 15..0 这些位可读可写，但只能在 LCKK 位为 0 时写入。 0: 端口配置未锁定 1: 端口配置已锁定

7.4.9. GPIO 复用功能低位寄存器 (GPIOx_AFRL) (x = A,B,C,D)

偏移地址: 0x20

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	AFSELy[3:0](y= 7 to 0)	RW	0	软件可写这些位配置复用功能 I/O AFSELy 选择: 0000:AF0 1000: AF8 0001:AF1 1001: AF9 0010:AF2 1010: AF10 0011:AF3 1011: AF11 0100:AF4 1100: AF12 0101:AF5 1101: AF13 0110:AF6 1110: AF14 0111:AF7 1111: AF15

7.4.10. GPIO 复用功能高位寄存器 (GPIOx_AFRH) (x = A,B,C,D)

偏移地址: 0x24

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	AFSELy[3:0](y= 8 to 15)	RW	0	软件可写这些位配置复用功能 I/O AFSELy 选择: 0000:AF0 1000: AF8 0001:AF1 1001: AF9 0010:AF2 1010: AF10 0011:AF3 1011: AF11 0100:AF4 1100: AF12 0101:AF5 1101: AF13 0110:AF6 1110: AF14 0111:AF7 1111: AF15

7.4.11. GPIO 端口位复位寄存器 (GPIOx_BRR) (x = A,B,C,D)

偏移地址: 0x28

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**2**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BRy	W	0	y = 15..0 这些位软件可写，读出来返回值是 0 0: 不会对相应的 ODy 位执行任何操作 1: 清除对应的 ODy 位

8. 外设互连

8.1. 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/或同步，节省了 CPU 资源，进而降低功耗。

此外，这些硬件连接消除了软件延迟，允许设计可预测系统。

这些互连可以在运行、睡眠、低功耗运行、低功耗睡眠、停止模式下工作，具体取决于外设。

8.2. 互连详情

8.2.1. 定时器输入触发(ITR)

目的

一些 TIMx 定时器在内部连接在一起，用于定时器的同步或者连接。当一个定时器为主模式时，它可以重置、启动、停止从模式定时器。

触发信号

在可配置定时器事件发生后，输出（来自主设备）出现在 TIMx_TRGO 信号上。

对于没有触发输出的 TIM16 和 TIM17 定时器，将使用输出比较 1 代替。

输入（到从设备）位于 TIMx_ITR0/ITR1/ITR2/ITR3/ITR4/ITR5 信号上。

表 8-1 定时器输入触发

TIMx 内部触发	定时器输入触发源（ITR）			
	TIM1	TIM2	TIM3	TIM15
ITR0(TIMx_SMCR.TS=0x0)	-	TIM1_TRGO	TIM1_TRGO	TIM1_TRGO
ITR1(TIMx_SMCR.TS=0x1)	TIM2_TRGO	-	TIM2_TRGO	TIM2_TRGO
ITR2(TIMx_SMCR.TS=0x2)	TIM3_TRGO	TIM3_TRGO	-	TIM3_TRGO
ITR3(TIMx_SMCR.TS=0x3)	TIM15_TRGO	TIM15_TRGO	TIM15_TRGO	-
ITR4(TIMx_SMCR.TS=0x8)	TIM16_OC1	TIM16_OC1	TIM16_OC1	TIM16_OC1
ITR5(TIMx_SMCR.TS=0x9)	TIM17_OC1	TIM17_OC1	TIM17_OC1	TIM17_OC1

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.2. 定时器外部触发(ETR)

目的

ADC 模拟看门狗信号输出、比较器的输出，以及定时器 ETR 引脚可以连接到定时器的外部触发 ETR。

触发信号

ADC 模拟看门狗输出 ADC_AWD、比较器的输出 COMP1_OUT,COMP2_OUT.

表 8-2 定时器外部触发

TIMx 外部触发 (TIMx_AF1.ETRSEL)	定时器外部触发源（ETR）				
	TIM1	TIM2	TIM3	TIM15	PWM
ETR0	TIM1_ETR	TIM2_ETR	TIM3_ETR	TIM15_ETR	PWM_ETR
ETR1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT
ETR2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT
ETR3	-	-	-	-	-
ETR4	-	-	-	-	SYSCLK
ETR5	ADC_AWD	TIM3_ETR	TIM2_ETR	ADC_AWD	PWM_ETR_HS

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.3. 清除定时器 OCxREF 信号

目的

比较器的输出，可以用于清除高级定时器 TIM1、通用定时器 TIM2、TIM3 的 OCxREF 信号。

触发信号

比较器的输出 COMP1_OUT,COMP2_OUT.

表 8-3 清除定时器 OCxREF 信号

OCxREF_CLR	清除定时器 OCxREF（OCxREF_CLR）源		
	TIM1	TIM2	TIM3
OCREF_CLR0	COMP1_OUT	COMP1_OUT	COMP1_OUT
OCREF_CLR1	COMP2_OUT	COMP2_OUT	COMP2_OUT

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.4. 定时器输入捕获

目的

比较器的输出，可以用作高级定时器 TIM1，通用定时器 TIM2/TIM3/TIM15 的输入捕获通道。

通用定时器 TIM15/TIM16 可以使用输入捕获通道 1 实现时钟测量。

表 8-4 输入捕获通道 1（TI1）源

TI1 (TIMx_TISEL.TI1SEL)	输入捕获通道 1(TI1)源					
	TIM1	TIM2	TIM3	TIM15	TIM16	TIM17
TI1_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM15_CH1	TIM16_CH1	TIM17_CH1
TI1_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT	LSECSS	-	-
TI1_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP1_OUT	MCO	MCO
TI1_IN3	-	-	-	COMP2_OUT	HSE_DIV32	HSE_DIV32
TI1_IN4	-	-	-	-	RTC_WKUP	RTC_WKUP
TI1_IN5	-	-	-	-	LSECSS	LSECSS
TI1_IN6	-	-	-	-	LSI	LSI

表 8-5 输入捕获通道 2（TI2）源

TI2 (TIMx_TISEL.TI2SEL)	输入捕获通道 2(TI2)源			
	TIM1	TIM2	TIM3	TIM15
TI2_IN0	TIM1_CH2	TIM2_CH2	TIM3_CH2	TIM15_CH2
TI2_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP2_OUT
TI2_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP1_OUT

表 8-6 输入捕获通道 3（TI3）源

TI3 (TIMx_TISEL.TI3SEL)	输入捕获通道 3(TI3)源		
	TIM1	TIM2	TIM3
TI3_IN0	TIM1_CH3	TIM2_CH3	TIM3_CH3
TI3_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT
TI3_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT

表 8-7 输入捕获通道 4（TI4）源

TI4 (TIMx_TISEL.TI4SEL)	输入捕获通道 4(TI4)源		
	TIM1	TIM2	TIM3
TI3_IN0	TIM1_CH4	TIM2_CH4	TIM3_CH4
TI3_IN1	COMP1_OUT	COMP1_OUT	COMP1_OUT
TI3_IN2	COMP2_OUT	COMP2_OUT	COMP2_OUT

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.5. 定时器刹车

目的

比较器的输出，可以用作高级定时器 TIM1，通用定时器 TIM15/TIM16/TIM17，以及 PWM 的刹车。

触发信号

比较器的输出 COMP1_OUT、COMP2_OUT，以及刹车引脚输入 TIMx_BRK。

表 8-8 定时器刹车输入

TIMx 刹车	定时器刹车(BRK)源				
	TIM1	TIM15	TIM16	TIM17	PWM
BRK0	TIM1_BRK	TIM2_BRK	TIM3_BRK	TIM15_BRK	PWM_BRK
BRK1	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT	COMP1_OUT
BRK2	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT	COMP2_OUT

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.6. 定时器系统刹车

目的

HSE 和 LSE CSS、CPU LOCKUP、SRAM 奇偶校验错误、FLASH ECC 检测错误，PVD 报警可面向 TIM1、TIM15、TIM16、TIM17 和 PWM 模块以定时器刹车形式生成系统错误。

刹车功能的目的是保护由这些定时器生成的 PWM 信号所驱动的功率器件。

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.7. 低功耗定时器 LPTIM 触发

目的

RTC 闹钟 A/B、 TAMP 输入检测、 COMP1/2_OUT 和 LPTIM1/2_ETR 引脚可用作触发信号以启动 LPTIM 计数器 LPTIM1/2。

表 8-9 LPTIM 外部触发

LPTIMx_CFGR.TRIGSEL[2:0]	低功耗定时器外部触发	
	LPTIM1	LPTIM2
EXT_TRIG0	LPTIM1_ETR	LPTIM2_ETR
EXT_TRIG1	RTC_ALARMA	RTC_ALARMA
EXT_TRIG2	RTC_ALARMB	RTC_ALARMB
EXT_TRIG3	TAMP_TRG	TAMP_TRG
EXT_TRIG4	COMP1_OUT	COMP1_OUT
EXT_TRIG5	COMP2_OUT	COMP2_OUT

相关功耗模式

这些互连在运行、睡眠、低功耗运行、低功耗睡眠和所有停止功耗模式下工作。

8.2.8. 红外输出波形控制(IRTIM)

目的

TIM16 或 TIM17 定时器的 TIMx_OC1 输出通道可生成红外输出波形。

表 8-10 IRTIM 控制信号分配

IRTIM 波形控制	IRTIM 波形控制信号源
调制包络信号	TIM16_OC1
载波信号	TIM17_OC1

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.9. 比较器消隐

目的

高级控制定时器 TIM1 和通用定时器 TIM2、TIM3、TIM15 和 PWM 可用作 COMP1 和 COMP2 的消隐窗口输入。

触发信号

定时器输出信号 TIMx_OCx/PWM_OC3 是 COMP1/COMP2 的消隐源输入。

表 8-11 比较器消隐输入

COMPx_CSR.BLANKSEL[2:0]	比较器消隐输入	
	COMP1	COMP2
BLANK0	-	-
BLANK1	TIM1_OC4	TIM1_OC4
BLANK2	TIM2_OC3	TIM2_OC3

BLANK3	TIM3_OC3	TIM3_OC3
BLANK4	TIM15_OC2	TIM15_OC2
BLANK5	PWM_OC3	PWM_OC3

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.10. ADC 硬件触发输入

目的

通用定时器 TIM2、TIM3 和 TIM15、高级定时器 TIM1 和 EXTI 可用于生成 ADC 触发事件。

触发信号

输出（来自定时器）位于 TIMx_TRGO、或 TIMx_OCx 信号事件上，以及 EXTI Line11、EXTI Line15 事件。

输入（到 ADC）位于 EXT[15:0] 和 JEXT[15:0] 信号上，选择分别由寄存器 ADC_CFGR.EXTSEL[3:0]和 ADC_JSQR.JEXTSEL[3:0]选择。

表 8-12 ADC 硬件触发源

EXTSEL[3:0]/JEXTSEL[3:0]	ADC 硬件触发源	
	规则触发	注入触发
0	TIM1_TRGO	TIM1_TRGO
1	TIM1_OC1	TIM1_OC1
2	TIM1_OC2	TIM1_OC2
3	TIM1_OC3	TIM1_OC3
4	TIM1_OC4	TIM1_OC4
5	TIM2_TRGO	TIM2_TRGO
6	TIM2_OC1	TIM2_OC1
7	TIM3_OC1	TIM3_OC1
8	TIM3_TRGO	TIM3_TRGO
9	TIM15_TRGO	TIM15_TRGO
10	-	-
11	-	-
12	EXTI_EVT11	EXTI_EVT11
13	EXTI_EVT15	EXTI_EVT15
14	-	-
15	-	-

8.2.11. ADC 内部通道源

目的

内部温度传感器输出电压 V_{TS}、内部参考电压 V_{REFINT} 、$V_{BAT}/3$、DAC 和 OPA 输出，连接到 ADC 输入通道。

表 8-13 ADC 内部通道源

ADC_CFGR.AWDCH[5:0]	ADC 内部通道源
6’b11011	VTS
6’b11100	V_{REFINT}
6’b11101	$V_{BAT}/3$
6’b11110	DAC_OUT
6’b11111	OPA_OUT

8.2.12. DAC 硬件触发输入

目的

通用定时器 TIM2、TIM3 和 TIM15，基本定时器 TIM6 和 TIM7、以及 EXTI 可用于生成 DAC 外部触发事件。

触发信号

输出（来自定时器）位于 TIMx_TRGO 事件上，以及 EXTI Line9 事件。

输入（到 DAC）位于 TSEL [7:0]信号上，选择分别由寄存器 DAC_CR1.TSEL[2:0]选择。

表 8-14 DAC 触发源

DAC_CR1.TSEL[2:0]	DAC 触发源
0	TIM6_TRGO
1	TIM3_TRGO
2	TIM7_TRGO
3	TIM15_TRGO
4	TIM2_TRGO
5	-
6	EXTI_EVT9
7	SWTRIG(软件触发)

9. 系统配置控制器(SYSCFG)

9.1. SYSCFG 主要特性

芯片内有一套配置寄存器，系统配置控制器的主要用途如下：

- I²C 类型 IO 滤波使能和关闭
- I²C Fm+模式的使能与关闭
- 根据不同启动模式,映射初始程序区
- DMA 外设通道选择控制
- 模拟 PAD2 使能
- 所有 GPIO 的噪声滤波器的使能与关闭
- PVD 锁定的使能与关闭
- Cortex®-M0+ LOCKUP 的使能与关闭
- Flash ECC 锁定的使能与关闭
- SRAM 奇偶校验的使能与关闭
- LED IO 控制

9.2. SYSCFG 寄存器

9.2.1. SYSCFG 配置寄存器 1(SYSCFG_CFGR1)

该寄存器的 MEM_MODE 位用作配置存储器地址 0x0000 0000 访问的种类。这两位用来选择软件的物理重映射，并旁路掉硬件启动选择。在上电或者 NRST 引脚复位后，这两位的值由硬件采样到的实际启动模式配置决定。

偏移地址：0x00

复位值：0x0000 000x(x 是被实际启动模式配置选择的存储器模式)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIO_AHB_SEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
							RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM_MODE[1:0]	
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	GPIO_AHB_SEL	RW	0	CPU FASTIO 或 CPU AHB 总线访问 GPIO 寄存器控制。 0：CPU FASTIO 总线访问 GPIO 寄存器 1：CPU AHB 总线访问 GPIO 寄存器 注：DMA 默认可以通过 AHB 总线访问 GPIO 寄存器，不受该位控制
23:2	Reserved	-	-	保留
1:0	MEM_MODE [1:0]	RW		存储器映射选择位

				软件置位，软件清零。该寄存器控制存储器的 0x0000 0000 地址的映射。在复位（仅包含上电复位、NRST 引脚复位和 OBL 复位）后，硬件重新采样 Boot 引脚和选项字节中定义，采用实际启动模式配置值。（**注意：**复位后采样 Boot 方式的时间点因复位不同而异，具体请咨询应用工程师） X0：逻辑地址 0x0000 0000 映射在 Flash 主存储区 01：逻辑地址 0x0000 0000 映射在 Flash 系统存储区 11：逻辑地址 0x0000 0000 映射在 SRAM

9.2.2. SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCL	PVDL	SPL	CLL
												RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	ECCL	RS	0	ECC 锁定. 此位由软件置 1，且只能被系统复位清零。此位能用于使能和锁定 FLASH ECC 错误连接到 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入。 软件写 0 到该位不会改变该位的值. 0：ECC 错误不与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入相连 1：ECC 错误与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入相连
2	PVDL	RS	0	PVD 锁定使能位 软件置位，系统复位清零。它可以被用作使能和锁定 PVD 连接给 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入，也锁定 PWR_CR2 寄存器的 PVDE 和 PLS[2:0]。 软件写 0 到该位不会改变该位的值。 0：PVD 中断不与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入连接。PVDE 位和 PLS[2:0]可以被软件写入。 1：PVD 中断与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入连接。PVDE 位和 PLS[2:0]只读。
1	SPL	RS	0	SRAM 奇偶校验锁定位. 此位由软件置 1，且只能被系统复位清零。此位能用于使能和锁定 SRAM1 的奇偶校验错误信号连接到 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入。 软件写 0 到该位不会改变该位的值。 0: SRAM 的奇偶校验错误信号不与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入相连 1: SRAM 的奇偶校验错误信号与 TIM1/TIM15/TIM16/TIM17/PWM 的刹车输入相连

0	CLL	RS	0	Cortex®-M0+ LOCKUP 位的使能位 软件置位，系统复位清零。它可以使能和锁定 Cortex®-M0+的 LOCKUP(HardFault)输出给 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入。 软件写 0 到该位不会改变该位的值。 0：Cortex®-M0+的 LOCKUP 输出不与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入连接 1：Cortex®-M0+的 LOCKUP 输出与 TIM1/TIM15/TIM16/TIM17/PWM 的系统刹车输入连接

9.2.3. SYSCFG 配置寄存器 3 (SYSCFG_CFGR3)

偏移地址：0x08

复位值：0x3F3F_3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	DMA4_MAP[5:0]						Res.	Res.	DMA3_MAP[5:0]					
		RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	DMA2_MAP[5:0]						Res.	Res.	DMA1_MAP[5:0]					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:24	DMA4_MAP[5:0]	RW	6’b111111	通道 4 外设请求映射。 定义参见通道 1 描述.
23:22	Reserved	-	-	保留
21:16	DMA3_MAP[5:0]	RW	6’b111111	通道 3 外设请求映射。 定义参见通道 1 描述.
15:14	Reserved	-	-	保留
13:8	DMA2_MAP[5:0]	RW	6’b111111	通道 2 外设请求映射。 定义参见通道 1 描述.
7:6	Reserved	-	-	保留
5:0	DMA1_MAP[5:0]	RW	6’b111111	通道 1 外设请求映射。 000000: ADC 000001: DAC 000010: 保留 000011: SPI1_RX 000100: SPI1_TX 000101: SPI2_RX 000110: SPI2_TX 000111: USART1_RX 001000: USART1_TX 001001: USART2_RX 001010: USART2_TX 001011: UART1_RX 001100: UART1_TX 001101: LPUART1_RX 001110: LPUART1_TX 001111: I2C1_RX 010000: I2C1_TX 010001: I2C2_RX 010010: I2C2_TX 010011: TIM1_CH1 010100: TIM1_CH2 010101: TIM1_CH3 010110: TIM1_CH4

				010111: TIM1_COM 011000: TIM1_TRG 011001: TIM1_UP 011010: TIM2_CH1 011011: TIM2_CH2 011100: TIM2_CH3 011101: TIM2_CH4 011110: TIM2_UP 011111: TIM2_TRG 100000: TIM3_CH1 100001: TIM3_CH2 100010: TIM3_CH3 100011: TIM3_CH4 100100: TIM3_UP 100101: TIM3_TRG 100110: TIM6_UP 100111: TIM7_UP 101000: TIM15_CH1 101001: TIM15_CH2 101010: TIM15_UP 101011: TIM15_TRG 101100: TIM15_COM 101101: TIM16_CH1 101110: TIM16_UP 101111: TIM17_CH1 110000: TIM17_UP 110001：保留 110010: LPUART2_RX 110011: LPUART2_TX 110100: PWM_CH1 110101: PWM_CH2 110110: PWM_CH3 110111: PWM_CH4 111000: PWM_UP 111001: UART2_RX 111010: UART2_TX

9.2.4. SYSCFG 配置寄存器 4 (SYSCFG_CFGR4)

偏移地址：0x0C

复位值：0x003F_3F3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA7_MAP[5:0]					
										RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	DMA6_MAP[5:0]						Res.	Res.	DMA5_MAP[5:0]					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:16	DMA7_MAP[5:0]	RW	6’b111111	通道 7 外设请求映射。 定义参见通道 1 描述.
15:14	Reserved	-	-	保留
13:8	DMA6_MAP[5:0]	RW	6’b111111	通道 6 外设请求映射。 定义参见通道 1 描述.
7:6	Reserved	-	-	保留
5:0	DMA5_MAP[5:0]	RW	6’b111111	通道 5 外设请求映射。 定义参见通道 1 描述.

9.2.5. GPIOA 噪声滤波使能寄存器（PA_ENS）

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PA_ENS[x]	RW	0x0000	GPIOA0~15 的噪声滤波使能 0: 关闭噪声滤波功能 1: 开启噪声滤波功能

9.2.6. GPIOB 噪声滤波使能寄存器（PB_ENS）

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PB_ENS[x]	RW	0x0000	GPIOB0~15 的噪声滤波使能 0: 关闭噪声滤波功能 1: 开启噪声滤波功能

9.2.7. GPIOC 噪声滤波使能寄存器（PC_ENS）

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留

15:0	PC_ENS[x]	RW	0x0000	GPIOC0~15 的噪声滤波使能 0: 关闭噪声滤波功能 1: 开启噪声滤波功能

9.2.8. GPIOD 噪声率使能寄存器（PD_ENS）

偏移地址:0x2C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PD_ENS[x]	RW	0x0000	GPIOD0~15 的噪声滤波使能 0: 关闭噪声滤波功能 1: 开启噪声滤波功能

9.2.9. I2C 类型 IO 配置寄存器（SYSCFG_IOCFG）

偏移地址：0x38

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	PD_EIIC[3:0]				Res.	PC_EIIC[8:0]								
		RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD_EIIC_11	PD_EIIC_10	PB_EIIC[5:0]						Res.	Res.	Res.	Res.	PA_EIIC[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:26	PD_EIIC[3:0]	RW	0	PD EIIC 信号控制。用作_I2C 类型 IO。 Bit0：控制 PD6 Bit1：控制 PD7 Bit2：保留 Bit3：保留
25	Reserved	-	-	保留
24:16	PC_EIIC[8:0]	RW	0	PC EIIC 信号控制。用作_I2C 类型 IO。 Bit0：控制 PC5 Bit1：控制 PC6 Bit2：控制 PC7 Bit3：控制 PC9 Bit4：控制 PC10 Bit5：控制 PC12 Bit6：控制 PC13 Bit7：控制 PC14

				Bit8：控制 PC15
15	PD_EIIC_11	RW	0	PD11 EIIC 信号控制。用作_I2C 类型 IO。
14	PD_EIIC_10	RW	0	PD10 EIIC 信号控制。用作_I2C 类型 IO。
13:8	PB_EIIC[5:0]	RW	0	PB IIC 信号控制。用作_I2C 类型 IO。 Bit0：控制 PB4 Bit1：控制 PB5 Bit2：控制 PB8 Bit3：控制 PB9 Bit4：控制 PB12 Bit5：控制 PB13
7:4	Reserved	-	-	保留
3:0	PA_EIIC[3:0]	RW	0	PA EIIC 信号控制。用作_I2C 类型 IO。 Bit0：控制 PA11 Bit1：控制 PA12 Bit2：控制 PA13 Bit3：控制 PA14

9.2.10. LED IO 配置寄存器（SYSCFG_LEDCFG）

偏移地址：0x3C

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC_ENSEG[1:0]		PB_ENSEG[13:0]													
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PB_EHS[7:0]								Res.	Res.	Res.	Res.	SEGIS_EN[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	PC_ENSEG[1:0]	RW	0	LED SEG 端口恒流使能控制。 Bit0：PC0（LED_SEG14） Bit1：PC1（LED_SEG15）
29:16	PB_ENSEG[13:0]	RW	0	LED SEG 端口恒流使能控制。 Bit0：PB2（LED_SEG0） Bit1：PB3（LED_SEG1） Bit2：PB4（LED_SEG2） Bit3：PB5（LED_SEG3） Bit4：PB6（LED_SEG4） Bit5：PB7（LED_SEG5） Bit6：PB8（LED_SEG6） Bit7：PB9（LED_SEG7） Bit8：PB10（LED_SEG8） Bit9：PB11（LED_SEG9） Bit10：PB12（LED_SEG10） Bit11：PB13（LED_SEG11） Bit12：PB14（LED_SEG12） Bit13：PB15（LED_SEG13）

15:8	PB_EHS[7:0]	RW	0	LED COM 120mA 灌电流控制。 Bit0：控制 PB2（LED_COM0） Bit1：控制 PB3（LED_COM1） Bit2：控制 PB4（LED_COM2） Bit3：控制 PB5（LED_COM3） Bit4：控制 PB6（LED_COM4） Bit5：控制 PB7（LED_COM5） Bit6：控制 PB8（LED_COM6） Bit7：控制 PB9（LED_COM7）
7:4	Reserved	-	-	保留
3:0	SEGIS_EN	RW	0	LED SEG 端口恒流源电路使能。 Bit0：恒流源 0 电路开关（for LED SEG0~2） Bit1：恒流源 1 电路开关（for LED SEG3~5） Bit2：恒流源 2 电路开关（for LED SEG6~10） Bit3：恒流源 3 电路开关（for LED SEG11~15）

9.2.11. GPIOA 模拟 PAD2 使能寄存器（PA_ANA2EN）

偏移地址:0x40

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA_ANA2EN[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PA_ANA2EN[x]	RW	0x0000	IO 端口 A PAD_ANA2 使能 0: PAD_ANA2 禁止 1: PAD_ANA2 使能 注 1：该寄存器需要在 IO 配置为模拟模式时使能。如果 IO 配置为非模拟模式，该寄存器需要配置为 0. 注 2：是否需要配置参见 PD_ANA2EN 寄存器下面描述。

9.2.12. GPIOB 模拟 PAD2 使能寄存器（PB_ANA2EN）

偏移地址:0x44

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PB_ANA2EN[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PB_ANA2EN[x]	RW	0x0000	IO 端口 B PAD_ANA2 使能 0: PAD_ANA2 禁止 1: PAD_ANA2 使能 注 1：该寄存器需要在 IO 配置为模拟模式时使能。如果 IO 配置为非模拟模式，该寄存器需要配置为 0. 注 2：是否需要配置参见 PD_ANA2EN 寄存器下面描述。

9.2.13. GPIOC 模拟 PAD2 使能寄存器（PC_ANA2EN）

偏移地址:0x48

复位值:0x0000 0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC_ANA2EN[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PC_ANA2EN[x]	RW	0x0000	IO 端口 C PAD_ANA2 使能 0: PAD_ANA2 禁止 1: PAD_ANA2 使能 注 1：该寄存器需要在 IO 配置为模拟模式时使能。如果 IO 配置为非模拟模式，该寄存器需要配置为 0. 注 2：是否需要配置参见 PD_ANA2EN 寄存器下面描述。

9.2.14. GPIOD 模拟 PAD2 使能寄存器（PD_ANA2EN）

偏移地址:0x4C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	PD_ANA2EN[13:0]													
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	PD_ANA2EN[x]	RW	0x000	IO 端口 D PAD_ANA2 使能 0: PAD_ANA2 禁止 1: PAD_ANA2 使能

				注 1：该寄存器需要在 IO 配置为模拟模式时使能。如果 IO 配置为非模拟模式，该寄存器需要配置为 0. 注 2：是否需要配置参见 PD_ANA2EN 寄存器下面描述。

关于模拟 PAD_ANA2，满足如下要求：

1) 当用做模拟功能时，仅比较器 COMP 的输入、运算放大器 OPA 输入/输出需要使能 Px_ANA2EN=1，其他作为模拟功能的引脚需要配置 Px_ANA2EN=0:

--COMP1 INP: PC15，PD0，PD1

--COMP1 INM: PA1, PA13，PC5

--COMP2 INP: PB14，PC3，PC4

--COMP2 INM: PA13，PB15，PC0

--OPA INP: PC2，PD3

--OPA INM: PC1，PD2

--OPA OUT：PC8，PC9，PD1，PD4

9.2.15. SRAM 控制状态寄存器（SYSCFG_SCSR）

偏移地址：0x70

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERR_RSTEN
							RC_W1								RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	SPF	RC_W1	0	SRAM 奇偶校验错误标志位. 当 SRAM 奇偶校验出现错误时，此位由硬件置 1。软件往此位写 1 即可清零 0: 没有检测到奇偶校验错误 1: 检测到奇偶校验错误
7:1	Reserved	-	-	保留
0	PERR_RSTEN	RW	0	SRAM 奇偶校验出错时响应控制。 0：SRAM 奇偶校验出错时产生 NMI 中断 1：SRAM 奇偶校验出错时产生系统复位

9.2.16. SRAM 写保护寄存器（SYSCFG_SWPR）

偏移地址：0x74

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P31WP	P30WP	P29WP	P28WP	P27WP	P26WP	P25WP	P24WP	P23WP	P22WP	P21WP	P20WP	P19WP	P18WP	P17WP	P16WP
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P15WP	P14WP	P13WP	P12WP	P11WP	P10WP	P9WP	P8WP	P7WP	P6WP	P5WP	P4WP	P3WP	P2WP	P1WP	P0WP
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	SRAM_PxWP	RW	0	SRAM 第 x 页(page)写保护 这些位由软件置 1，且只能被系统复位清零。 0：SRAM 第 x 页写保护关闭 1：SRAM 第 x 页写保护使能

10. 直接存储区访问（DMA）

10.1. DMA 简介

直接存储器存取(DMA)用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。无须CPU 干预，数据可以通过 DMA 快速地移动，节省了 CPU 的资源,进行其他操作。

DMA 控制器有 7 条 DMA 通道，每条通道负责管理来自 1 个或者多个外设对存储器访问的请求。DMA 控制器包括处理 DMA 请求的仲裁器，用于处理各个 DMA 请求的优先级。

10.2. DMA 主要特性

- 7 个独立可配置的通道
- 每个通道通过配置可以连接任一外设的硬件 DMA 请求，每个通道同样支持软件触发。
- 在同一个 DMA 模块上，多个请求间的优先权可以通过软件编程设置，优先权设置相等时由硬件决定（通道号越低优先级越高）
- 独立数据源和目标数据区的传输宽度(字节、半字、全字)，模拟打包和拆包的过程。源和目标地址必须按数据传输宽度对齐
- 可编程的源和目标地址，地址可选递增，递减或不变
- 每个通道都有 4 个事件标志(传输完成、块传输完成、半块传输完成，传输错误)，这 4 个事件标志进行“逻辑或”，成为一个单独的中断请求
- 支持存储器和存储器间、外设和存储器、存储器和外设、外设和外设的数据传输
- SRAM、APB 和 AHB 外设均可作为访问的源和目标，FLASH 只能作为源不能作为目标
- 支持单次触发模式和四种循环模式
  - 外设地址保持，存储器地址保持
  - 外设地址重新加载，存储器地址保持
  - 外设地址保持，存储器地址重新加载
  - 外设和存储器地址都重新加载
- 单次模式可编程传输数量 0~65535
- 循环模式支持无限循环和有限循环(1~255)
- 支持单一传输和批量传输
  - 单一传输：搬运 1 次数据回复 1 次 ACK；
  - 批量传输：搬运配置的数据量后回复 1 次 ACK(所有数据搬运结束后释放总线)；
- 支持存储器到储存器模式的两种传输方式
  - 快速模式：获得仲裁后始终占据总线，直到传输完所有数据后释放总线；
  - 轮换模式：传输 1 次数据后释放总线重新仲裁；
- 在循环模式下,支持进入块传输完成中断后暂停传输

10.3. DMA 功能描述

10.3.1. DMA 框图

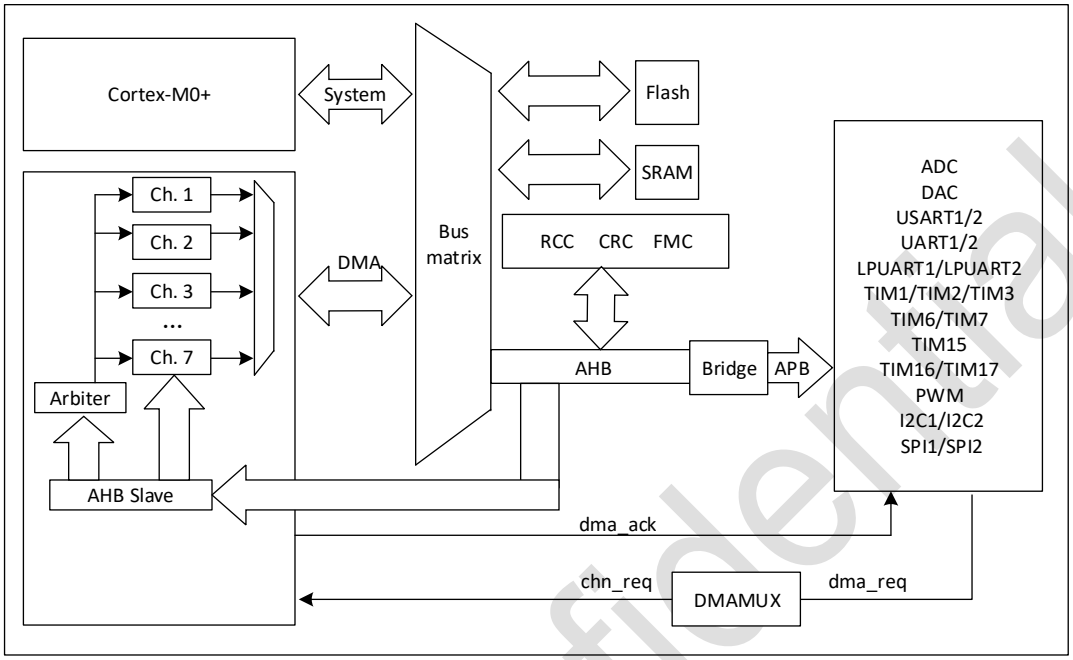


图 10-1 DMA 框图

10.3.2. CPU 和 DMA 总线共享

DMA 控制器和 CPU 共享 AMBA 数据总线，执行直接存储器数据传输。当 CPU 和 DMA 同时访问相同的目标(RAM 或外设)时，DMA 请求会暂停 CPU 访问系统总线达若干个周期，总线仲裁器执行循环调度，以保证 CPU 至少可以得到一半的系统总线带宽。

10.3.3. DMA 传输

在完成一个事件后，外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道的优先权处理请求。当 DMA 控制器开始访问发出请求的外设时，DMA 控制器在传输结束后发送给它一个应答信号。当从 DMA 控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，DMA 控制器撤销应答信号。如果有更多的请求时，外设可以启动下一个传送。

总之，每次 DMA 传输由三个操作组成：

- 从外设数据寄存器或者从当前外设/存储器地址寄存器取数，地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设地址或存储器单元。
- 存数据到外设寄存器或者当前外设/存储器地址寄存器指示的存储器地址，地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设地址或存储器单元。
- 执行一次 DMA_CNDTRx 寄存器的递减操作，该寄存器表明未完成的操作数目。

10.3.4. DMA 仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。

优先权管理分 2 个阶段：

- 软件：每个通道的优先权可以在 DMA_CCRx 寄存器中设置，有 7 个等级（优先级种类等于通道数量）：
  - 0
  - 1
  - 2
  - 3
  - 4
  - 5
  - 6

优先级 6>5>4>3>2>1>0。

- 硬件：如果 2 个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有更高的优先权。比如，通道 2 优先于通道 3。

10.3.5. DMA 通道

每个通道都可以在有固定地址的外设寄存器和存储器地址之间执行 DMA 传输。DMA 的传输数据量是可编程的，最大为 65535。包含要传输的数据数量的寄存器，在每次传输后递减。

10.3.5.1. 可编程数据大小

外设和存储器的传输数据宽度可以通过 DMA_CCRx 寄存器中的 MSIZE 和 PSIZE 位编程。

10.3.5.2. 可编程地址增量方式

通过设置 DMA_CCRx 寄存器中的 PINC 和 MINC 标志位，外设和存储器的指针在每次传输后可以有选择的完成自动增量。当设置位增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决于所选的数据宽度为 1，2，4。

第一个传输的地址是存放在 DMA_CPARx/DMA_CMARx 寄存器中的地址。在传输过程中，这些寄存器保持他们的初始值，软件不能改变和读出当前正在传输的地址（它在内部的当前外设/存储器地址寄存器中）。

当通道配置为单块传输时，传输结束后(即传输计数变为 0)将不再产生 DMA 操作。要开始新的 DMA 传输，需要在关闭 DMA 通道的情况下，在 DMA_CNDTRx 寄存器中重新写入传输数目。

在循环模式下，最后一次传输结束时，DMA_CNDTRx 寄存器的内容会自动被重新加载为其初始数值，内部的当前外设/存储器地址寄存器根据 MNORLD 和 PNORLD 的配置保持地址或者重新加载地址。

10.3.5.3. 循环模式

循环模式用于处理循环缓冲区和连续的数据传输。在 DMA_CCRx 寄存器中的 CIRC 位用于开启这一功能。当启动了循环模式，数据传输的数目变为 0 时，将会自动地被恢复配置通道时设置的初值，DMA 操作将会继续进行。

循环模式地址重产生方式有如下 4 种情况：

- 外设地址保持，存储器地址保持
- 外设地址重新加载，存储器地址保持
- 外设地址保持，存储器地址重新加载
- 外设和存储器地址都重新加载

循环模式分为以下两种循环次数：

- 无限循环：通道使能，DMA_CCRx.CIRC 为 1，且 DMA_CCCFGRx.NBT 为 0，单块传输数据量为 DMA CNDTRx.NDT；
- 有限循环：通道使能，DMA_CCRx.CIRC 为 1，且 DMA_CCCFGRx.NBT 不为 0。
  —DMA_CCCFGRx.NBT 大于 0 时，单块传输数据量为 DMA_CNDTRx.NDT；
  —DMA_CCCFGRx.NBT 等于 0 时，若 DMACCCEGRX.NDTL 等于 0 则结束传输；若 DMACCCFGRx.NDTL 大于 0 则继续传输一次 DMA CCCFGRx.NDTL 个数的数据后结束传输；

10.3.5.4. 存储器到存储器模式

DMA 通道的操作可以在没有外设请求的情况下进行，这种操作就是存储器到存储器模式。

当设置了 DMA_CCRx 寄存器中的 MEM2MEM 位之后，在软件设置了 DMA_CCRx 寄存器中的 EN 位启动 DMA 通道后，DMA 传输将马上开始。当 DMA_CNDTRx 寄存器变为 0 时，DMA 传输结束。

通过配置 DMC_CCRx.MEMARB 寄存器，存储器到存储器模式支持以下两种模式

- 快速模式：获得仲裁后始终占据总线，直到传输完所有数据后释放总线，该模式不能与循环模式同时使用；
- 轮换模式：传输 1 次数据后释放总线重新仲裁，该模式下其他通道也有获得仲裁的机会；

10.3.5.5. 传输类型

根据外设的传输需求，支持以下两种数据传输方式

- 单一传输：传输 1 次数据回复 1 次 dma_ack；
- 批量传输：传输 NDT 次数据回复 1 次 dma_ack，使用该方式传输数据的通道会在传输完所有数据后再释放总线；

10.3.5.6. 块传输完成中断后暂停

在循环模式下，可以设置 DMA_CCRx.BTCSUSP 为 1 使通道在进入到块传输完成中断后暂停传输数据，在清除块传输完成中断后恢复数据传输。

10.3.5.7. 通道配置步骤

按如下步骤配置 DMA 通道：

- 在 DMA_CPARx 寄存器中设置外设寄存器的地址。发生外设数据传输请求时，这个地址将是数据传输的源或目标。
- 在 DMA_CMARx 寄存器中设置数据存储器的地址。发生外设数据传输请求时，传输的数据将从这个地址读出或写入这个地址。
- 在 DMA_CNDTRx 寄存器中设置要传输的数据量。在每个数据传输后，这个数值递减。
- 在 DMA_CCRx 寄存器的 PL 位中设置通道的优先级。
- 在 DMA_CCRx 寄存器中设置数据传输的方向、循环模式、外设和存储器的增量模式、外设和存储器的数据宽度、传输一半产生中断或传输完成产生中断。
- 在 DMA_CCCFGRx 寄存器中设置循环传输次数。在有限循环时，每个块传输后，这个数值递减。
- 设置 DMA_CCRx 寄存器的 EN 位，启动该通道。

一旦启动了 DMA 通道，即可响应连到该通道上的外设的 DMA 请求。

当传输一半的数据后，半块传输标志(HBTIF)被置 1，当设置了允许半块传输中断位(HBTIE)时，将产生一个中断请求。单块数据传输结束后，块传输完成标志(BTCIF)被置 1，当设置了允许块传输完成中断位(BTCIE)时，将产生一个中断请求。多块数据全部传输结束后，传输完成标志(TCIF)被置 1，当设置了允许传输完成中断位(TCIE)时，将产生一个中断请求。

10.3.6. 可编程数据宽度，数据对齐和大小端

当存储器数据宽度 MSIZE 和外设数据宽度 PSIZE 不同时，DMA 按照下表进行数据对齐：

表 10-2 数据宽度和大小端（PINC=MINC=1）

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
8	8	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1:在 0x0 读 B0[7:0],在 0x0 写 B0[7:0] 2:在 0x1 读 B1[7:0],在 0x1 写 B1[7:0] 3:在 0x2 读 B2[7:0],在 0x2 写 B2[7:0] 4:在 0x3 读 B3[7:0],在 0x3 写 B3[7:0]	0x0/B0 0x1/B1 0x2/B2 0x3/B3
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1:在 0x0 读 B0[7:0],在 0x0 写 00B0[7:0] 2:在 0x1 读 B1[7:0],在 0x2 写 00B1[7:0] 3:在 0x2 读 B2[7:0],在 0x4 写 00B2[7:0] 4:在 0x3 读 B3[7:0],在 0x6 写 00B3[7:0]	0x0/00B0 0x2/00B1 0x4/00B2 0x6/00B3
8	32	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1:在 0x0 读 B0[7:0],在 0x0 写 000000B0[31:0] 2:在 0x1 读 B1[7:0],在 0x4 写 000000B1[31:0] 3:在 0x2 读 B2[7:0],在 0x8 写 000000B2[31:0] 4:在 0x3 读 B3[7:0],在 0xC 写 000000B3[31:0]	0x0/000000B0 0x4/000000B1 0x8/000000B2 0xC/000000B3

16	8	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1:在 0x0 读 B1B0[15:0],在 0x0 写 B0[7:0] 2:在 0x2 读 B3B2[15:0],在 0x1 写 B2[7:0] 3:在 0x4 读 B5B4[15:0],在 0x2 写 B4[7:0] 4:在 0x6 读 B7B6[15:0],在 0x3 写 B6[7:0]	0x0/B0 0x1/B2 0x2/B4 0x3/B6
16	16	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1:在 0x0 读 B1B0[15:0],在 0x0 写 B1B0[15:0] 2:在 0x2 读 B3B2[15:0],在 0x2 写 B3B2[15:0] 3:在 0x4 读 B5B4[15:0],在 0x4 写 B5B4[15:0] 4:在 0x6 读 B7B6[15:0],在 0x6 写 B7B6[15:0]	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6
16	32	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1:在 0x0 读 B1B0[7:0],在 0x0 写 0000B1B0[31:0] 2:在 0x2 读 B3B2[7:0],在 0x4 写 0000B3B2[31:0] 3:在 0x4 读 B5B4[7:0],在 0x8 写 0000B5B4[31:0] 4:在 0x6 读 B7B6[7:0],在 0xC 写 0000B7B6[31:0]	0x0/0000B1B0 0x4/000B3B2 0x8/0000B5B4 0xC/0000B7B6
32	8	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B0[7:0] 2:在 0x4 读 B7B6B5B4 [31:0],在 0x1 写 B4[7:0] 3:在 0x8 读 BBBAB9B8 [31:0],在 0x2 写 B8[7:0] 4:在 0xc 读 BFBEBDBC [31:0],在 0x3 写 BC[7:0]	0x0/B0 0x1/B4 0x2/B8 0x3/BC
32	16	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B1B0[7:0] 2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B5B4[7:0] 3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 B9B8[7:0] 4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BDBC[7:0]	0x0/B1B0 0x2/B5B4 0x4/B9B8 0x6/BDBC
32	32	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B3B2B1B0[7:0] 2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B7B6B5B4[7:0] 3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 BBBAB9B8[7:0] 4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BFBEBDBC[7:0]	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC

10.3.6.1. 不支持字节/半字访问

如果 DMA 以字节或半字写入不支持字节或半字写操作的 AHB 设备时(即 HSIZE 不适于该模块)，不会发生错误， DMA 将按照下面两个例子写入 32 位 HWDATA 数据：

- 当 HSIZE=半字时，写入半字'0xABCD'，DMA 将设置 HWDATA 总线为'0xABCDABCD'。
- 当 HSIZE=字节时，写入字节'0xAB'，DMA 将设置 HWDATA 总线为'0xABABABAB'。

假定 AHB/APB 桥是一个 AHB 的 32 位从机，它不处理 HSIZE 参数，它将按照下述方式把任何 AHB 上的字节或半字按 32 位传送到 APB 上：

- 一个 AHB 上对地址 0x0(或 0x1、0x2 或 0x3)的写字节数据'0xB0'操作，将转换到 APB 上对地址 0x0 的写字数据'0xB0B0B0B0'操作。

- 一个 AHB 上对地址 0x0(或 0x2)的写半字数据'0xB1B0'操作，将转换到 APB 上对地址 0x0 的写字数据'0xB1B0B1B0'操作。

10.3.7. 错误管理

读写一个保留的地址区域，将会产生 DMA 传输错误。在 DMA 读写操作时，发生 DMA 传输错误时，硬件会自动清除发生错误的通道所对应的通道配置寄存器（DMA_CCRx）的 EN 位，该通道操作被停止。此时，在 DMA_IFR 寄存器中对应该通道的传输错误中断标志位（TEIF）将被置位，如果在 DMA_CCRx 寄存器中设置了传输错误中断允许位，则将产生中断。

10.4. DMA 中断

每个 DMA 通道都可以在 DMA 块传输过半、块传输完成、传输完成和传输错误时产生中断。为应用的灵活性考虑，通过设置寄存器的不同位来打开这些中断。

表 10-3 DMA 中断请求

中断事件	事件标志位	使能控制位
块传输过半	HBTIF	HBTIE
块传输完成	BTCIF	BTCIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

注：当 DMA_CNDTRx 寄存器为 1 时，不会置 HBTIFx 位

10.5. DMA 外设请求映射

表 10-4 DMA 外设通道请求

Peripherals	Channel 1~7(每个通道都可以配置为如下请求)
ADC	ADC
DAC	DAC
SPI	SPI1_RX SPI1_TX SPI2_RX SPI2_TX
USART UART LPUART	USART1_RX USART1_TX USART2_RX USART2_TX UART1_RX UART1_TX UART2_RX UART2_TX LPUART1_RX LPUART1_TX LPUART2_RX LPUART2_TX
I2C	I2C1_RX I2C1_TX I2C2_RX I2C2_TX
TIM1	TIM1_CH1 TIM1_CH2 TIM1_CH3 TIM1_CH4 TIM1_UP

	TIM1_TRG TIM1_COM
TIM2	TIM2_CH1 TIM2_CH2 TIM2_CH3 TIM2_CH4 TIM2_UP TIM2_TRG
TIM3	TIM3_CH1 TIM3_CH2 TIM3_CH3 TIM3_CH4 TIM3_UP TIM3_TRG
TIM6	TIM6_UP
TIM7	TIM7_UP
TIM15	TIM15_CH1 TIM15_CH2 TIM15_COM TIM15_UP TIM15_TRG
TIM16	TIM16_CH1 TIM16_UP
TIM17	TIM17_CH1 TIM17_UP
PWM	PWM_CH1 PWM_CH2 PWM_CH3 PWM_CH4 PWM_UP

10.6. DMA 寄存器

10.6.1. DMA 中断状态寄存器（DMA_ISR）

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TEIF7	HBTIF7	BTCIF7	TCIF7	TEIF6	HBTIF6	BTCIF6	TCIF6	TEIF5	HBTIF5	BTCIF5	TCIF5
				R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEIF4	HBTIF4	BTCIF4	TCIF4	TEIF3	HBTIF3	BTCIF3	TCIF3	TEIF2	HBTIF2	BTCIF2	TCIF2	TEIF1	HBTIF1	BTCIF1	TCIF1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27	TEIF7	R	0	通道 7 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF7=1 清零。 0：无 TE 事件 1：发生 TE 事件
26	HBTIF7	R	0	通道 7 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF7=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
25	BTCIF7	R	0	通道 7 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF7=1 清零。

				0：无 BTC 事件 1：发生 BTC 事件
24	TCIF7	R	0	通道 7 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF7=1 清零。 0：无 TC 事件 1：发生 TC 事件
23	TEIF6	R	0	通道 6 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF6=1 清零。 0：无 TE 事件 1：发生 TE 事件
22	HBTIF6	R	0	通道 6 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF6=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
21	BTCIF6	R	0	通道 6 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF6=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
20	TCIF6	R	0	通道 6 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF6=1 清零。 0：无 TC 事件 1：发生 TC 事件
19	TEIF5	R	0	通道 5 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF5=1 清零。 0：无 TE 事件 1：发生 TE 事件
18	HBTIF5	R	0	通道 5 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF5=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
17	BTCIF5	R	0	通道 5 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF5=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
16	TCIF5	R	0	通道 5 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF5=1 清零。 0：无 TC 事件 1：发生 TC 事件
15	TEIF4	R	0	通道 4 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF4=1 清零。 0：无 TE 事件 1：发生 TE 事件
14	HBTIF4	R	0	通道 4 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF4=1 清零。 0：无 HBT 事件 1：发生 HBT 事件

13	BTCIF4	R	0	通道 4 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF4=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
12	TCIF4	R	0	通道 4 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF4=1 清零。 0：无 TC 事件 1：发生 TC 事件
11	TEIF3	R	0	通道 3 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF3=1 清零。 0：无 TE 事件 1：发生 TE 事件
10	HBTIF3	R	0	通道 3 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF3=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
9	BTCIF3	R	0	通道 3 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF3=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
8	TCIF3	R	0	通道 3 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF3=1 清零。 0：无 TC 事件 1：发生 TC 事件
7	TEIF2	R	0	通道 2 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF2=1 清零。 0：无 TE 事件 1：发生 TE 事件
6	HBTIF2	R	0	通道 2 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF2=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
5	BTCIF2	R	0	通道 2 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF2=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
4	TCIF2	R	0	通道 2 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF2=1 清零。 0：无 TC 事件 1：发生 TC 事件
3	TEIF1	R	0	通道 1 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF1=1 清零。 0：无 TE 事件 1：发生 TE 事件
2	HBTIF1	R	0	通道 1 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF1=1 清零。

				0：无 HBT 事件 1：发生 HBT 事件
1	BTCIF1	R	0	通道 1 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF1=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
0	TCIF1	R	0	通道 1 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF1=1 清零。 0：无 TC 事件 1：发生 TC 事件

10.6.2. DMA 中断标志清零寄存器（DMA_IFCR）

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	CTEIF7	CHBTIF7	CBTCIF7	CTCIF7	CTEIF6	CHBTIF6	CBTCIF6	CTCIF6	CTEIF5	CHBTIF5	CBTCIF5	CTCIF5
				W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTEIF4	CHBTIF4	CBTCIF4	CTCIF4	CTEIF3	CHBTIF3	CBTCIF3	CTCIF3	CTEIF2	CHBTIF2	CBTCIF2	CTCIF2	CTEIF1	CHBTIF1	CBTCIF1	CTCIF1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27	CTEIF7	W	0	通道 7 传输错误标志清零。 0：无影响 1：清零 TEIF7
26	CHBTIF7	W	0	通道 7 块半传输标志清零。 0：无影响 1：清零 HBTIF7
25	CBTCIF7	W	0	通道 7 块传输完成标志清零。 0：无影响 1：清零 BTCIF7
24	CTCIF7	W	0	通道 7 所有块传输完成标志清零。 0：无影响 1：清零 TCIF7
23	CTEIF6	W	0	通道 6 传输错误标志清零。 0：无影响 1：清零 TEIF6
22	CHBTIF6	W	0	通道 6 块半传输标志清零。 0：无影响 1：清零 HBTIF6
21	CBTCIF6	W	0	通道 6 块传输完成标志清零。 0：无影响 1：清零 BTCIF6
20	CTCIF6	W	0	通道 6 所有块传输完成标志清零。 0：无影响 1：清零 TCIF6

19	CTEIF5	W	0	通道 5 传输错误标志清零。 0：无影响 1：清零 TEIF5
18	CHBTIF5	W	0	通道 5 块半传输标志清零。 0：无影响 1：清零 HBTIF5
17	CBTCIF5	W	0	通道 5 块传输完成标志清零。 0：无影响 1：清零 BTCIF5
16	CTCIF5	W	0	通道 5 所有块传输完成标志清零。 0：无影响 1：清零 TCIF5
15	CTEIF4	W	0	通道 4 传输错误标志清零。 0：无影响 1：清零 TEIF4
14	CHBTIF4	W	0	通道 4 块半传输标志清零。 0：无影响 1：清零 HBTIF4
13	CBTCIF4	W	0	通道 4 块传输完成标志清零。 0：无影响 1：清零 BTCIF4
12	CTCIF4	W	0	通道 4 所有块传输完成标志清零。 0：无影响 1：清零 TCIF4
11	CTEIF3	W	0	通道 3 传输错误标志清零。 0：无影响 1：清零 TEIF3
10	CHBTIF3	W	0	通道 3 块半传输标志清零。 0：无影响 1：清零 HBTIF3
9	CBTCIF3	W	0	通道 3 块传输完成标志清零。 0：无影响 1：清零 BTCIF3
8	CTCIF3	W	0	通道 3 所有块传输完成标志清零。 0：无影响 1：清零 TCIF3
7	CTEIF2	W	0	通道 2 传输错误标志清零。 0：无影响 1：清零 TEIF2
6	CHBTIF2	W	0	通道 2 块半传输标志清零。 0：无影响 1：清零 HBTIF2
5	CBTCIF2	W	0	通道 2 块传输完成标志清零。 0：无影响 1：清零 BTCIF2
4	CTCIF2	W	0	通道 2 所有块传输完成标志清零。

				0：无影响 1：清零 TCIF2
3	CTEIF1	W	0	通道 1 传输错误标志清零。 0：无影响 1：清零 TEIF1
2	CHBTIF1	W	0	通道 1 块半传输标志清零。 0：无影响 1：清零 HBTIF1
1	CBTCIF1	W	0	通道 1 块传输完成标志清零。 0：无影响 1：清零 BTCIF1
0	CTCIF1	W	0	通道 1 所有块传输完成标志清零。 0：无影响 1：清零 TCIF1

10.6.3. DMA 通道 x 配置寄存器（DMA_CCRx）(x=1~7)

偏移地址:0x08+(x-1)*0x14（x=1~7）

复位值:0x000x 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	M2MARB	BTCSUSP	MNORLD	PNORLD	MEM2MEM	PL[2:0]		
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSIZE[1:0]		PSIZE[1:0]		MINC[1:0]		PINC[1:0]		TRANST	CIRC	DIR	TEIE	HBTIE	BTCIE	TCIE	EN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	M2MARB	RW	0	通道 x 存储器到存储器模式下仲裁方式（mem2mem=1 时有效） 0：传输完 1 次数据后不释放总线；（不支持循环模式） 1：传输完 1 次数据后释放总线; EN=0 时软件可读可写; EN=1 时软件只可读;
22	BTCSUSP	RW	0	通道 x 循环模式下进入 BTC 中断后处理方式 0：不暂停传输; 1：暂停传输; 该位配置为 1 时，进入 BTC 中断后软件清除 BTC 中断后恢复传输; EN=0 时软件可读可写; EN=1 时软件只可读;
21	MNORLD	RW	0	通道 x 的存储器地址循环模式（CIRC=1 时有效） 0：存储器地址重新加载使能; 1：存储器地址重新加载禁止; EN=0 时软件可读可写; EN=1 时软件只可读;
20	PNORLD	RW	0	通道 x 的外设地址循环模式（CIRC=1 时有效）

				0：外设地址重新加载使能； 1：外设地址重新加载禁止； EN=0 时软件可读可写； EN=1 时软件只可读；
19	MEM2MEM	RW	0	通道 x 存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能； EN=0 时软件可读可写； EN=1 时软件只可读；
18:16	PL[2:0]	RW	x	通道 x 优先级配置。 000：通道优先级为 0； 001：通道优先级为 1； 010：通道优先级为 2； 011：通道优先级为 3； 100：通道优先级为 4； 101：通道优先级为 5； 110：通道优先级为 6； 其它：保留 EN=0 时软件可读可写； EN=1 时软件只可读；
15:14	MSIZE[1:0]	RW	0	通道 x 存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留； EN=0 时软件可读可写； EN=1 时软件只可读；
13:12	PSIZE[1:0]	RW	0	通道 x 外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留； EN=0 时软件可读可写； EN=1 时软件只可读；
11:10	MINC[1:0]	RW	0	通道 x 存储器地址增量模式。 00：不变； 01：递增； 10：递减； 11：保留； EN=0 时软件可读可写； EN=1 时软件只可读；
9:8	PINC[1:0]	RW	0	通道 x 外设地址增量模式。 00：不变； 01：递增； 10：递减； 11：保留； EN=0 时软件可读可写；

				EN=1 时软件只可读；
7	TRANST	RW	0	通道 x 的数据传输方式（mem2mem=0 时有效） 0：单一传输； 1：批量传输； EN=0 时软件可读可写； EN=1 时软件只可读；
6	CIRC	RW	0	循环模式 0：禁止； 1：使能； EN=0 时软件可读可写； EN=1 时软件只可读；
5	DIR	RW	0	通道 x 数据传输方向。 0：从外设读； 1：从存储器读； EN=0 时软件可读可写； EN=1 时软件只可读；
4	TEIE	RW	0	通道 x 传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
3	HBTIE	RW	0	通道 x 块半传输中断（HBT）使能。 0：禁止； 1：HBT 中断使能；
2	BTCIE	RW	0	通道 x 块传输完成中断（BTC）使能。 0：禁止； 1：BTC 中断使能；
1	TCIE	RW	0	通道 x 所有块传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道 x 使能. 0：禁止； 1：通道 x 使能； 当所有块传输完成或者出现传输错误时硬件清零； 软件可读可写； 注 1：在无效配置，如配置 NDT=0，但配置 EN=1 时，硬件会清零 EN，以保证后续传输正常； 注 2：配置 DMA 寄存器遵循先配置后控制的原则，即在最后配置 EN=1；

10.6.4. DMA 通道 x 数据传输个数寄存器（DMA_CNDTRx）(x=1~7)

偏移地址:0x0C+(x-1)*0x14（x=1~7）

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NDT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	NDT[15:0]	RW	0	通道 x 数据传输数量。 数据传输数量为 0~65535。该寄存器只在通道不工作（DMA_CCRx.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输数量。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。 EN=0 时软件可读可写； EN=1 时软件只可读；

10.6.5. DMA 通道 x 外设地址寄存器（DMA_CPARx）(x=1~7)

偏移地址:0x10+(x-1)*0x14（x=1~7）

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PA[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	PA[31:0]	RW	0	通道 x 外设地址。 通道 x 外设数据寄存器的基址，作为数据传输的源或目标。 当 PSIZE=2’b01，不使用 PA[0]位。操作自动与半字地址对齐。 当 PSIZE=2’b10，不使用 PA[1:0]位。操作自动与字地址对齐。 EN=0 时软件可读可写； EN=1 时软件只可读；

10.6.6. DMA 通道 x 存储器地址寄存器（DMA_CMARx）(x=1~7)

偏移地址:0x14+(x-1)*0x14（x=1~7）

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MA[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MA[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	MA[31:0]	RW	0	通道 x 存储器地址。 通道 x 存储器地址，作为数据传输的源或目标。

				当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2’b10，不使用 MA[1:0]位。操作自动与字地址对齐。 EN=0 时软件可读可写； EN=1 时软件只可读；

10.6.7. DMA 通道 x 循环传输配置寄存器（DMA_CCCFGRx）(x=1~7)

偏移地址:0x18+(x-1)*0x14（x=1~7）

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDTL[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NBT[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	NDTL[15:0]	RW	0	有限循环模式下通道 x 最后一次块数据传输数量。 数据传输数量为 0~65535。 该寄存器只在通道不工作(DMA CCRx.EN=0)时写入。通道使能后该寄存器为只读，表明循环模式最后一次剩余传输数量。 该寄存器值在循环模式最后一个块传输时随每笔传输递减。 EN=0 时软件可读可写; EN=1 时软件只可读;
15:8	Reserved	-	-	保留
7:0	NBT[7:0]	RW	0	通道 x 块循环传输数量。 在循环模式下，该寄存器初始值为 0 时块循环传输数量为无穷，该寄存器初始值大于 0 时块循环传输数量为该寄存器值(1~255)。该寄存器只在通道不工作（DMA_CCRx.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输块数。该寄存器值在每个块传输完成后递减。 EN=0 时软件可读可写； EN=1 时软件只可读； 注:有限循环传输数据个数=NDT*NBT+NDTL

11. 中断和事件

11.1. 嵌套向量中断控制器 (NVIC)

11.1.1. NVIC 主要特性

- 31 个可屏蔽的中断通道（不包括 16 个 CPU 的中断）
- 4 个可编程的优先级（2 位中断优先级）
- 低延迟的异常和中断处理
- 功耗管理控制
- 系统控制寄存器的实现

NVIC 和 CPU 接口是紧耦合的，这使得低延迟中断处理和后到达中断的高效处理成为可能。包括 CPU 的异常，所有中断都被 NVIC 管理。

11.1.2. SysTick 校准值寄存器

SysTick 校准值被设为 9000，通过 SysTick 时钟置为 9 MHz（max f_{HCLK}/8），会产生 1ms 的时间基准。

11.1.3. 中断和异常向量

表 11-1 终端和异常向量表

向量编号	优先级	优先级类型	缩略语	描述	地址
-	-	-	-	保留	0x0000_0000
-	-3	固定	Reset	复位	0x0000_0004
-	-2	固定	NMI_Handler	NMI. 1） RCC 时钟安全检测故障(CSS) 2） SRAM 奇偶校验错误 3） Flash ECC 错误超过 1bit	0x0000_0008
-	-1	固定	HardFualt_Handler	所有类型的错误	0x0000_000C
-	-	-	-	保留	0x0000_0010~ 0x0000_0028
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000_002C
-	-	-	-	保留	0x0000_0030~ 0x0000_0034
-	5	可设置	PendSV	可挂起的系统服务请求	0x0000_0038
-	6	可设置	SysTick	系统节拍定时器	0x0000_003C
0	7	可设置	WWDG	窗口看门狗中断	0x0000_0040
1	8	可设置	PVD	通过 EXTI Line16 的 PVD 中断	0x0000_0044
2	9	可设置	RTC_TAMP	RTC 和 TAMP 中断	0x0000_0048

3	10	可设置	Flash	Flash 全局中断	0x0000_004C
4	11	可设置	RCC	RCC 全局中断	0x0000_0050
5	12	可设置	EXTI0_1	EXTI Line 0 和 1 中断	0x0000_0054
6	13	可设置	EXTI2_3	EXTI Line 0 和 1 中断	0x0000_0058
7	14	可设置	EXTI4_15	EXTI Line 4 到 15 中断	0x0000_005C
8	15	可设置	LCD_CANINT1	LCD 和 CAN 全局中断 1	0x0000_0060
9	16	可设置	DMA_Channel1	DMA 通道 1 中断	0x0000_0064
10	17	可设置	DMA_Channel2_3	DMA 通道 2 和 3 中断	0x0000_0068
11	18	可设置	DMA_Channel4_7	DMA 通道 4 到 7 中断	0x0000_006C
12	19	可设置	ADC_COMP	ADC 和 COMP 中断(COMP 与 EXTI Line 17 和 18 结合使用)	0x0000_0070
13	20	可设置	TIM1_BRK_UP_TRG_COM	TIM1 刹车、更新、触发和换相中断	0x0000_0074
14	21	可设置	TIM1_CC	TIM1 捕获比较中断	0x0000_0078
15	22	可设置	TIM2	TIM2 全局中断	0x0000_007C
16	23	可设置	TIM3	TIM3 全局中断	0x0000_0080
17	24	可设置	TIM6_DAC_LPTIM1	LPTIM1、TIM6 和 DAC 中断	0x0000_0084
18	25	可设置	TIM7_LPTIM2	LPTIM2 和 TIM7 中断	0x0000_0088
19	26	可设置	PWM_CANINT	PWM 和 CAN 全局中断 2	0x0000_008C
20	27	可设置	TIM15	TIM15 全局中断	0x0000_0090
21	28	可设置	TIM16	TIM16 全局中断	0x0000_0094
22	29	可设置	TIM17	TIM17 全局中断	0x0000_0098
23	30	可设置	I2C1	I2C1 全局中断	0x0000_009C
24	31	可设置	I2C2	I2C2 全局中断	0x0000_00A0
25	32	可设置	SPI1	SPI1 全局中断	0x0000_00A4
26	33	可设置	SPI2	SPI2 全局中断	0x0000_00A8
27	34	可设置	USART1	USART1 全局中断	0x0000_00AC
28	35	可设置	USART2	USART2 全局中断	0x0000_00B0
29	36	可设置	UART1_LPUART1	UART1 & LPUART1 全局中断	0x0000_00B4
30	37	可设置	UART2_LPUART2	UART2 &LPUART2 全局中断	0x0000_00B8
31	38	保留	-	-	0x0000_00BC

1. 灰色单元格对应于 Cortex®-M0+中断。

11.2. 扩展中断和事件控制器 (EXTI)

扩展中断和事件控制器，通过 configurable（可配置）和 direct（直接事件）输入(Lines)，管理着 CPU 和系统唤醒功能，并输出下述请求信号：

- 中断请求，产生 CPU 的中断输入（IRQ）
- 事件请求，产生 CPU 的事件输入（RXEV）
- 唤醒请求，送给功耗管理控制模块

EXTI 唤醒请求允许系统从停止模式唤醒，中断请求和事件请求也可以在运行/低功耗运行模式使用。

EXTI 允许管理多达 28 个可配置/直接事件 Line（19 个可配置事件 Line 和 8 个直接事件 Line）。

11.2.1. EXTI 主要特性

- 系统可以通过 GPIO 和指定模块（PVD/COMP/RTC/TAMP/LPTIM/LPUART/I2C）输入事件唤醒
- 可配置型事件（来自 I/O，或无中断挂起状态位的外设，产生脉冲的外设）
  - 可选有效触发沿（上升沿/下降沿）
  - 中断挂起标志位
  - 独立中断和事件产生屏蔽位
  - 可软件触发
- 直接型事件（具有相关标志和有中断挂起状态位的外设）
  - 固定的上升沿触发
  - 在 EXTI 模块里没有中断挂起位
  - 独立中断和事件产生屏蔽位
  - 无软件触发
- IO 端口选择

11.2.2. EXTI 框图

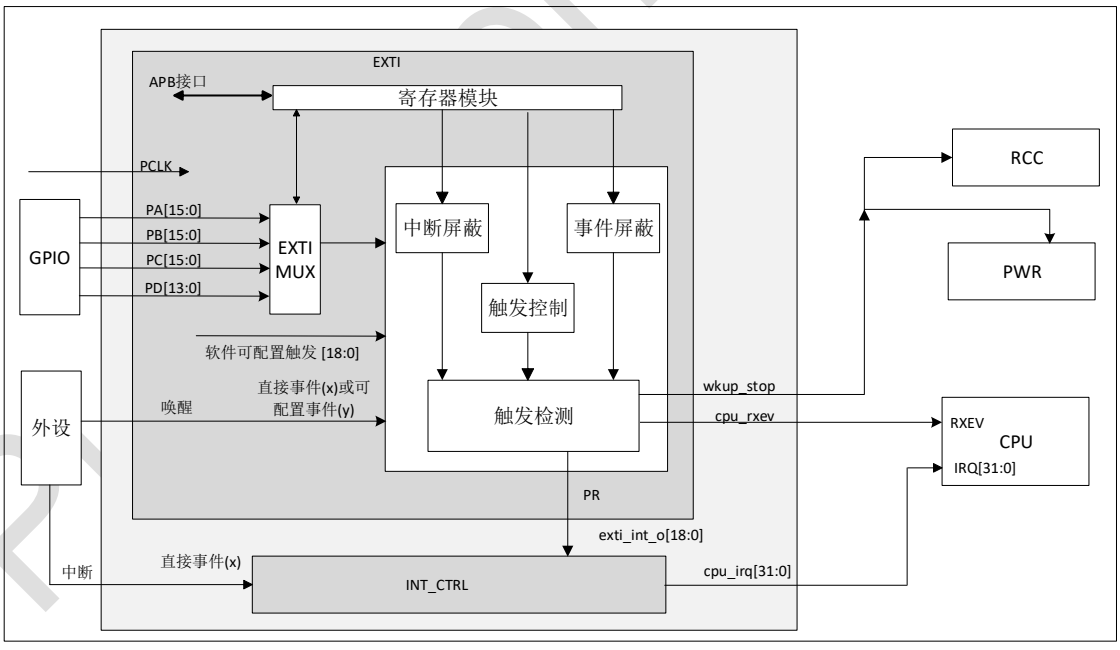


图 11-1 EXTI 框图

表 11-2 信号概述

信号名	I/O	说明
APB 接口	I/O	EXTI 寄存器总线接口。
PCLK	I	EXTI 系统时钟
可配置事件 (y)	I	来自外设的异步唤醒事件，在外设中没有相关中断和标志

直接事件 (x)	I	来自外设的同步和异步唤醒事件，在外设中具有相关中断和标志
PA/PB/PC[15:0] PD[13:0]	I	GPIO 端口
wkup_stop	O	输出到 PWR 和 RCC 模块用作唤醒处理
cpu_rxev	O	输出到 CPU 的 RXEV 接口信号
exti_int_o[18:0]	O	系统时钟域信号，可配置类型 Line 的中断
cpu_irq[31:0]	O	输出到 CPU 的 IRQ 接口信号

11.2.3. EXTI 功能说明

在停止模式下能产生唤醒或者中断事件信号的外设，连接至 EXTI 模块。

- 产生一个脉冲的，或者外设内部没有中断状态位的唤醒信号，被连接到 EXTI 模块的可配置类型 Line。此时 EXTI 模块产生一个中断挂起位（该位需要被清零），该 EXTI 中断会作为 CPU 的中断信号。
- 有关联的状态位（该位需要在外设被清零）的外设的中断和唤醒信号，连接到 EXTI 模块的直接类型 Line。
- 所有 GPIO 端口输入到 EXTI MUX 模块，通过选择配置，允许选中后作为系统唤醒信号。

11.2.4. EXTI 可配置事件输入唤醒

支持软件触发，通过配置 EXTI_SWIER 寄存器，软件可以触发唤醒功能。

有对应寄存器配置上升沿或者下降沿触发，或者双沿触发，硬件根据配置检测可配置类型事件输入信号，产生对应唤醒事件或者中断信号。

有中断屏蔽寄存器(EXTI_IMR)和事件屏蔽寄存器(EXTI_EMR)。事件使能后产生给 CPU 的事件。所有给 CPU 的事件'或'运算后输出到 CPU 的唯一事件输入信号 RXEV。

可配置类型事件有唯一的中断挂起请求寄存器，与 CPU 共享。挂起寄存器只有当 CPU 中断屏蔽寄存器（EXTI_IMR）配置为未屏蔽时才会置位。每一个可配置类型事件都会对应 CPU 外部中断信号（有些会复用到同一个 CPU 外部中断信号）。可配置类型事件中断需要 CPU 通过 EXTI_PR 寄存器确认（写 1 清零）。

注：当中断挂起寄存器（EXTI_PR）保持有效时（未清零），系统不能进入低功耗模式。

11.2.5. EXTI 直接事件输入唤醒

直接类型事件不会在 EXTI 模块产生中断，但会产生唤醒系统和 CPU 子系统的事件信号。CPU 在处理该种类型触发事件产生的中断时，要清零外设模块的中断状态位。

11.2.6. EXTI 复用

GPIO 被用以下方式连接到 16 个外部中断/事件 Line 上：

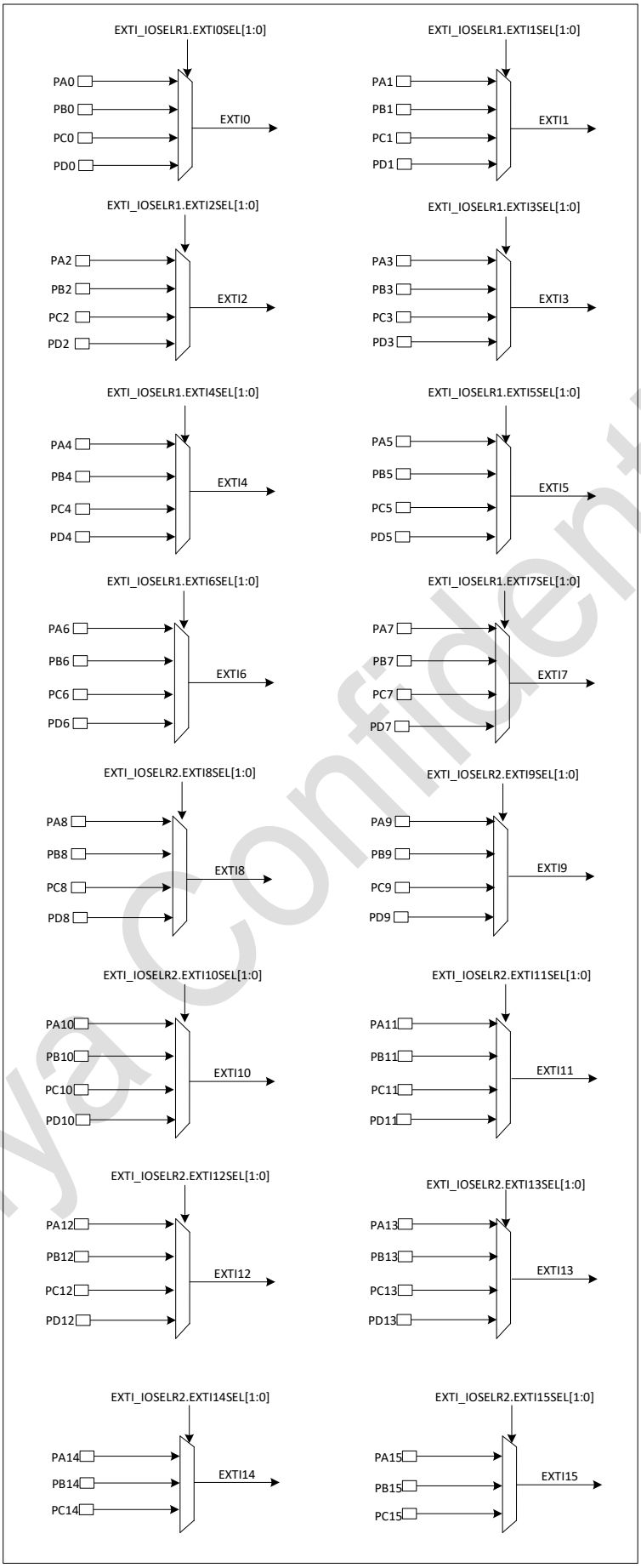


图 11-2 EXTI GPIO 复用

11.2.7. EXTI Line

所有 Line 连接内容如下表所示：

表 11-3 EXTI 线连接

EXTI Line	源	类型
0-15	GPIO	配置
16	PVD output	配置
17	COMP 1 output	配置
18	COMP 2 output	配置
19	-	-
20	RTC	直接
21	TAMP	直接
22	I2C1	直接
23	I2C2	直接
24	LPUART1	直接
25	LPUART2	直接
26	-	-
27	LPTIM1	直接
28	LPTIM2	直接
29	-	-

11.3. EXTI 寄存器

11.3.1. EXTI 上升沿触发选择寄存器 (EXTI_RTSR)

偏移地址：0x00

复位值：0x0000 0000

仅包含对可配置事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT18	RT17	RT16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18	RT18	RW	0	配置类型 EXTI line18 上升沿触发配置。 0：禁止 1：使能
17	RT17	RW	0	配置类型 EXTI line17 上升沿触发配置。 0：禁止 1：使能
16	RT16	RW	0	配置类型 EXTI line16 上升沿触发配置。 0：禁止 1：使能
15	RT15	RW	0	配置类型 EXTI line15 上升沿触发配置。 0：禁止 1：使能

14	RT14	RW	0	配置类型 EXTI line14 上升沿触发配置。 0：禁止 1：使能
13	RT13	RW	0	配置类型 EXTI line13 上升沿触发配置。 0：禁止 1：使能
12	RT12	RW	0	配置类型 EXTI line12 上升沿触发配置。 0：禁止 1：使能
11	RT11	RW	0	配置类型 EXTI line11 上升沿触发配置。 0：禁止 1：使能
10	RT10	RW	0	配置类型 EXTI line10 上升沿触发配置。 0：禁止 1：使能
9	RT9	RW	0	配置类型 EXTI line9 上升沿触发配置。 0：禁止 1：使能
8	RT8	RW	0	配置类型 EXTI line8 上升沿触发配置。 0：禁止 1：使能
7	RT7	RW	0	配置类型 EXTI line7 上升沿触发配置。 0：禁止 1：使能
6	RT6	RW	0	配置类型 EXTI line6 上升沿触发配置。 0：禁止 1：使能
5	RT5	RW	0	配置类型 EXTI line5 上升沿触发配置。 0：禁止 1：使能
4	RT4	RW	0	配置类型 EXTI line4 上升沿触发配置。 0：禁止 1：使能
3	RT3	RW	0	配置类型 EXTI line3 上升沿触发配置。 0：禁止 1：使能
2	RT2	RW	0	配置类型 EXTI line2 上升沿触发配置。 0：禁止 1：使能
1	RT1	RW	0	配置类型 EXTI line1 上升沿触发配置。 0：禁止 1：使能
0	RT0	RW	0	配置类型 EXTI line0 上升沿触发配置。 0：禁止 1：使能

可配置类型 EXTI Line 是边沿触发的，在这些 Line 上不能产生毛刺。如果在写 EXTI_RTSR 寄存器期间，配置中断线出现了上升沿，相关的挂起位不被置位。

在同一个 Line 上可以同时设置上升和下降沿，在该情况下，两种边沿都会产生触发条件。

11.3.2. EXTI 下降沿触发选择寄存器 (EXTI_FTSR)

偏移地址：0x04

复位值：0x0000 0000

仅包含对可配置事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT18	FT17	FT16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18	FT18	RW	0	配置类型 EXTI line18 下降沿触发配置。 0：禁止 1：使能
17	FT17	RW	0	配置类型 EXTI line17 下降沿触发配置。 0：禁止 1：使能
16	FT16	RW	0	配置类型 EXTI line16 下降沿触发配置。 0：禁止 1：使能
15	FT15	RW	0	配置类型 EXTI line15 下降沿触发配置。 0：禁止 1：使能
14	FT14	RW	0	配置类型 EXTI line14 下降沿触发配置。 0：禁止 1：使能
13	FT13	RW	0	配置类型 EXTI line13 下降沿触发配置。 0：禁止 1：使能
12	FT12	RW	0	配置类型 EXTI line12 下降沿触发配置。 0：禁止 1：使能
11	FT11	RW	0	配置类型 EXTI line11 下降沿触发配置。 0：禁止 1：使能
10	FT10	RW	0	配置类型 EXTI line10 下降沿触发配置。 0：禁止 1：使能
9	FT9	RW	0	配置类型 EXTI line9 下降沿触发配置。 0：禁止 1：使能

8	FT8	RW	0	配置类型 EXTI line8 下降沿触发配置。 0：禁止 1：使能
7	FT7	RW	0	配置类型 EXTI line7 下降沿触发配置。 0：禁止 1：使能
6	FT6	RW	0	配置类型 EXTI line6 下降沿触发配置。 0：禁止 1：使能
5	FT5	RW	0	配置类型 EXTI line5 下降沿触发配置。 0：禁止 1：使能
4	FT4	RW	0	配置类型 EXTI line4 下降沿触发配置。 0：禁止 1：使能
3	FT3	RW	0	配置类型 EXTI line3 下降沿触发配置。 0：禁止 1：使能
2	FT2	RW	0	配置类型 EXTI line2 下降沿触发配置。 0：禁止 1：使能
1	FT1	RW	0	配置类型 EXTI line1 下降沿触发配置。 0：禁止 1：使能
0	FT0	RW	0	配置类型 EXTI line0 下降沿触发配置。 0：禁止 1：使能

可配置类型 EXTI Line 是边沿触发的，在这些 Line 上不能产生毛刺。如果在写 EXTI_FTSR 寄存器期间，配置中断线出现了下降沿，相关的挂起位不被置位。

在同一个 Line 上可以同时设置上升和下降沿，在该情况下，两种边沿都会产生触发条件。

11.3.3. EXTI 软件中断事件寄存器 (EXTI_SWIER)

偏移地址：0x08

复位值：0x0000 0000

仅包含对可配置事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWIE18	SWIE17	SWIE16
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWIE15	SWIE14	SWIE13	SWIE12	SWIE11	SWIE10	SWIE9	SWIE8	SWIE7	SWIE6	SWIE5	SWIE4	SWIE3	SWIE2	SWIE1	SWIE0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18	SWIE18	RW	0	可配置类型 EXTI line18 软件上升沿触发配置。 0：无影响

				1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
17	SWIE17	RW	0	可配置类型 EXTI line17 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
16	SWIE16	RW	0	可配置类型 EXTI line16 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
15	SWIE15	RW	0	可配置类型 EXTI line15 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
14	SWIE14	RW	0	可配置类型 EXTI line14 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
13	SWIE13	RW	0	可配置类型 EXTI line13 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
12	SWIE12	RW	0	可配置类型 EXTI line12 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
11	SWIE11	RW	0	可配置类型 EXTI line11 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
10	SWIE10	RW	0	可配置类型 EXTI line10 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
9	SWIE9	RW	0	可配置类型 EXTI line9 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断

				该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
8	SWIE8	RW	0	可配置类型 EXTI line8 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
7	SWIE7	RW	0	可配置类型 EXTI line7 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
6	SWIE6	RW	0	可配置类型 EXTI line6 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
5	SWIE5	RW	0	可配置类型 EXTI line5 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
4	SWIE4	RW	0	可配置类型 EXTI line4 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
3	SWIE3	RW	0	可配置类型 EXTI line3 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
2	SWIE2	RW	0	可配置类型 EXTI line2 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
1	SWIE1	RW	0	可配置类型 EXTI line1 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。
0	SWIE0	RW	0	可配置类型 EXTI line0 软件上升沿触发配置。 0：无影响 1：产生上升沿触发事件，进而产生中断 该位当软件写 EXTI_PR 寄存器的对应位为 1 时清零。读返回 0（清零后）或者配置值（清零前）。

11.3.4. EXTI 挂起寄存器（EXTI_PR)

偏移地址：0x0C

复位值：0x0000 0000

仅包含对可配置事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR18	PR17	PR16
													RC_W1	RC_W1	RC_W1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18	PR18	RC_W1	0	配置类型 EXTI line18 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
17	PR17	RC_W1	0	配置类型 EXTI line17 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
16	PR16	RC_W1	0	配置类型 EXTI line16 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
15	PR15	RC_W1	0	配置类型 EXTI line15 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
14	PR14	RC_W1	0	配置类型 EXTI line14 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
13	PR13	RC_W1	0	配置类型 EXTI line13 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
12	PR12	RC_W1	0	配置类型 EXTI line12 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
11	PR11	RC_W1	0	配置类型 EXTI line11 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；

10	PR10	RC_W1	0	配置类型 EXTI line10 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
9	PR9	RC_W1	0	配置类型 EXTI line9 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
8	PR8	RC_W1	0	配置类型 EXTI line8 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
7	PR7	RC_W1	0	配置类型 EXTI line7 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
6	PR6	RC_W1	0	配置类型 EXTI line6 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
5	PR5	RC_W1	0	配置类型 EXTI line5 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
4	PR4	RC_W1	0	配置类型 EXTI line4 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
3	PR3	RC_W1	0	配置类型 EXTI line3 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
2	PR2	RC_W1	0	配置类型 EXTI line2 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
1	PR1	RC_W1	0	配置类型 EXTI line1 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
0	PR0	RC_W1	0	配置类型 EXTI line0 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；

11.3.5. EXTI 外部中断选择寄存器 1(EXTI_IOSELR1)

偏移地址：0x10

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EXTI7SEL[1:0]		Res.	Res.	EXTI6SEL[1:0]		Res	Res.	EXTI5SEL[1:0]		Res.	Res.	EXTI4SEL[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	EXTI3SEL[1:0]		Res.	Res.	EXTI2SEL[1:0]		Res	Res.	EXTI1SEL[1:0]		Res.	Res.	EXTI0SEL[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:28	EXTI7SEL[1:0]	RW	0	EXTI7 对应 GPIO 端口选择。 2'b00: PA[7] 引脚 2'b01: PB[7] 引脚 2'b10: PC[7] 引脚 2'b11: PD[7] 引脚
27:26	Reserved	-	-	保留
25:24	EXTI6SEL[1:0]	RW	0	EXTI6 对应 GPIO 端口选择。 2'b00: PA[6] 引脚 2'b01: PB[6] 引脚 2'b10: PC[6] 引脚 2'b11: PD[6] 引脚
23:22	Reserved	-	-	保留
21:20	EXTI5SEL[1:0]	RW	0	EXTI5 对应 GPIO 端口选择。 2'b00: PA[5] 引脚 2'b01: PB[5] 引脚 2'b10: PC[5] 引脚 2'b11: PD[5] 引脚
19:18	Reserved	-	-	保留
17:16	EXTI4SEL[1:0]	RW	0	EXTI4 对应 GPIO 端口选择。 2'b00: PA[4] 引脚 2'b01: PB[4] 引脚 2'b10: PC[4] 引脚 2'b11: PD[4] 引脚
15:14	Reserved	-	-	保留
13:12	EXTI3SEL[1:0]	RW	0	EXTI3 对应 GPIO 端口选择。 2'b00: PA[3] 引脚 2'b01: PB[3] 引脚 2'b10: PC[3] 引脚 2'b11: PD[3] 引脚
11:10	Reserved	-	-	保留
9:8	EXTI2SEL[1:0]	RW	0	EXTI2 对应 GPIO 端口选择。 2'b00: PA[2] 引脚 2'b01: PB[2] 引脚 2'b10: PC[2] 引脚

				2’b11: PD[2] 引脚
7:6	Reserved	-	-	保留
5:4	EXTI1SEL[1:0]	RW	0	EXTI1 对应 GPIO 端口选择。 2’b00: PA[1] 引脚 2’b01: PB[1] 引脚 2’b10: PC[1] 引脚 2’b11: PD[1] 引脚
3:2	Reserved	-	-	保留
1:0	EXTI0SEL1:0]	RW	0	EXTI0 对应 GPIO 端口选择。 2’b00: PA[0] 引脚 2’b01: PB[0] 引脚 2’b10: PC[0] 引脚 2’b11: PD[0] 引脚

11.3.6. EXTI 外部中断选择寄存器 2(EXTI_IOSELR2)

偏移地址：0x14

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EXTI15SEL[1:0]		Res.	Res.	EXTI14SEL[1:0]		Res	Res.	EXTI13SEL[1:0]		Res.	Res.	EXTI12SEL[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	EXTI11SEL[1:0]		Res.	Res.	EXTI10SEL[1:0]		Res	Res.	EXTI9SEL[1:0]		Res.	Res.	EXTI8SEL[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:28	EXTI15SEL[1:0]	RW	0	EXTI15 对应 GPIO 端口选择。 2’b00: PA[15] 引脚 2’b01: PB[15] 引脚 2’b10: PC[15] 引脚 2’b11: 保留
27:26	Reserved	-	-	保留
25:24	EXTI14SEL[1:0]	RW	0	EXTI14 对应 GPIO 端口选择。 2’b00: PA[14] 引脚 2’b01: PB[14] 引脚 2’b10: PC[14] 引脚 2’b11: 保留
23:22	Reserved	-	-	保留
21:20	EXTI13SEL[1:0]	RW	0	EXTI13 对应 GPIO 端口选择。 2’b00: PA[13] 引脚 2’b01: PB[13] 引脚 2’b10: PC[13] 引脚 2’b11: PD[13] 引脚
19:18	Reserved	-	-	保留
17:16	EXTI12SEL[1:0]	RW	0	EXTI12 对应 GPIO 端口选择。 2’b00: PA[12] 引脚

				2’b01: PB[12] 引脚 2’b10: PC[12] 引脚 2’b11: PD[12] 引脚
15:14	Reserved	-	-	保留
13:12	EXTI11SEL[1:0]	RW	0	EXTI11 对应 GPIO 端口选择。 2’b00: PA[11] 引脚 2’b01: PB[11] 引脚 2’b10: PC[11] 引脚 2’b11: PD[11] 引脚
11:10	Reserved	-	-	保留
9:8	EXTI10SEL[1:0]	RW	0	EXTI10 对应 GPIO 端口选择。 2’b00: PA[10] 引脚 2’b01: PB[10] 引脚 2’b10: PC[10] 引脚 2’b11: PD[10] 引脚
7:6	Reserved	-	-	保留
5:4	EXTI9SEL[1:0]	RW	0	EXTI9 对应 GPIO 端口选择。 2’b00: PA[9] 引脚 2’b01: PB[9] 引脚 2’b10: PC[9] 引脚 2’b11: PD[9] 引脚
3:2	Reserved	-	-	保留
1:0	EXTI8SEL[1:0]	RW	0	EXTI8 对应 GPIO 端口选择。 2’b00: PA[8] 引脚 2’b01: PB[8] 引脚 2’b10: PC[8] 引脚 2’b11: PD[8] 引脚

11.3.7. EXTI 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x18

复位值：0x1FF0 0000

注意：直接类型 line 的中断屏蔽位默认为 1，即允许该 line；配置类型 line 的屏蔽位，默认为 0，即屏蔽该 line。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	IM28	IM27	Res.	IM25	IM24	IM23	IM22	IM21	IM20	Res.	IM18	IM17	IM16
			RW	RW		RW	RW	RW	RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	IM28	RW	1	EXTI line28 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
27	IM27	RW	1	EXTI line27 作为中断唤醒 CPU 屏蔽控制。

				0：中断唤醒屏蔽 1：中断唤醒未屏蔽
26	Reserved	-	-	保留
25	IM25	RW	1	EXTI line25 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
24	IM24	RW	1	EXTI line24 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
23	IM23	RW	1	EXTI line23 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
22	IM22	RW	1	EXTI line22 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
21	IM21	RW	1	EXTI line21 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
20	IM20	RW	1	EXTI line20 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
19	Reserved	-	-	保留
18	IM18	RW	0	EXTI line18 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
17	IM17	RW	0	EXTI line17 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
16	IM16	RW	0	EXTI line16 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
15	IM15	RW	0	EXTI line15 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
14	IM14	RW	0	EXTI line14 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
13	IM13	RW	0	EXTI line13 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
12	IM12	RW	0	EXTI line12 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
11	IM11	RW	0	EXTI line11 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽

10	IM10	RW	0	EXTI line10 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
9	IM9	RW	0	EXTI line9 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
8	IM8	RW	0	EXTI line8 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
7	IM7	RW	0	EXTI line7 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
6	IM6	RW	0	EXTI line6 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
5	IM5	RW	0	EXTI line5 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
4	IM4	RW	0	EXTI line4 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
3	IM3	RW	0	EXTI line3 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
2	IM2	RW	0	EXTI line2 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
1	IM1	RW	0	EXTI line1 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
0	IM0	RW	0	EXTI line0 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽

11.3.8. EXTI 事件屏蔽寄存器 (EXTI_EMR)

偏移地址：0x1C

复位值：0x1FF0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	EM28	EM27	Res.	EM25	EM24	EM23	EM22	EM21	EM20	Res.	EM18	EM17	EM16
			RW	RW		RW	RW	RW	RW	RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留

28	EM28	RW	1	EXTI line28 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
27	EM27	RW	1	EXTI line27 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
26	Reserved	-	-	保留
25	EM25	RW	1	EXTI line25 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
24	EM24	RW	1	EXTI line24 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
23	EM23	RW	1	EXTI line23 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
22	EM22	RW	1	EXTI line22 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
21	EM21	RW	1	EXTI line21 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
20	EM20	RW	1	EXTI line20 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
19	Reserved	-	-	保留
18	EM18	RW	0	EXTI line18 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
17	EM17	RW	0	EXTI line17 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
16	EM16	RW	0	EXTI line16 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
15	EM15	RW	0	EXTI line15 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
14	EM14	RW	0	EXTI line14 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
13	EM13	RW	0	EXTI line13 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
12	EM12	RW	0	EXTI line12 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽

				1：事件唤醒未屏蔽
11	EM11	RW	0	EXTI line11 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
10	EM10	RW	0	EXTI line10 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
9	EM9	RW	0	EXTI line9 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
8	EM8	RW	0	EXTI line8 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
7	EM7	RW	0	EXTI line7 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
6	EM6	RW	0	EXTI line6 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
5	EM5	RW	0	EXTI line5 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
4	EM4	RW	0	EXTI line4 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
3	EM3	RW	0	EXTI line3 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
2	EM2	RW	0	EXTI line2 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
1	EM1	RW	0	EXTI line1 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
0	EM0	RW	0	EXTI line0 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽

12. 循环冗余校验计算(CRC)

12.1. CRC 简介

循环冗余校验(CRC)单元可以利用 7 位、8 位、16 位、32 位的自定义生成多项式，从 8 位、16 位、32 位的数据中计算出 CRC 码。

基于 CRC 的技术用于验证数据传输或存储的完整性。在功能安全标准的领域内，基于 CRC 的技术提供了一种验证闪存完整性的方法。

12.2. CRC 主要特性

- 默认使用 CRC-32（以太网）多项式：0x4C11DB7

  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

- 支持可编程的多项式，多项式位宽也是可编程的(支持 7，8，16，32 比特的多项式)
- 支持 8、16、32 位数据输入
- 可编程 CRC 初值
- 单个输入/输出 32 位数据和结果输出共用一个寄存器
- 通用 32 位寄存器（可被用作临时存储）
- 计算时间：32bits 数据 4 个 AHB 时钟；对于 16 位数据，需要 2 个 AHB 时钟周期；对于 8 位数据，需要 1 个 AHB 时钟周期(HCLK)
- 支持输入数据反转，输出数据反转
- CRC_DR 寄存器能被 AHB 总线以右对齐 8 位、右对齐 16 位、32 位的宽度访问，其他的寄存器只能以 32 位的宽度被 AHB 总线访问

12.3. CRC 功能描述

12.3.1. CRC 框图

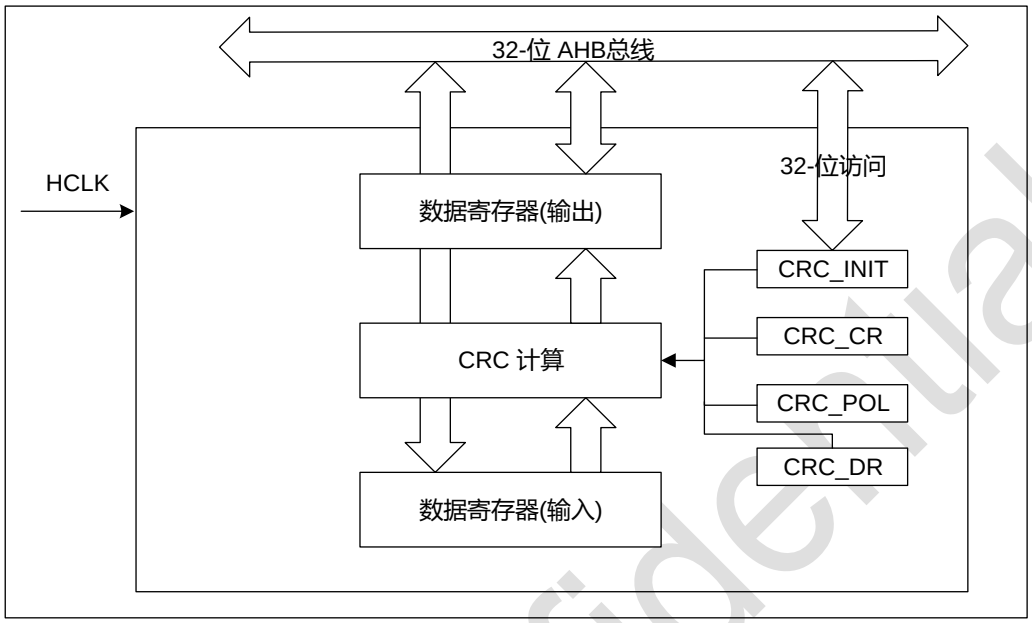


图 12-1 CRC 框图

12.3.2. CRC 功能描述

CRC 计算单元含有 1 个 32 位数据寄存器（CRC_DR）：

- 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 计算的新数据。
- 对该寄存器进行读操作时，返回上一次 CRC 计算的结果。

每一次写入数据寄存器，其计算结果是前一次 CRC 计算结果和新计算结果的组合(对整个 32 位字进行 CRC 计算，或逐字节地计算) ，具体取决于所写数据的格式。CRC_DR 寄存器能被字(32 位)、右对齐半字(16 位)、右对齐字节(8 位)访问，而其他的 CRC 寄存器只允许 32 位访问。

对于不同位宽(由 HSIZE 决定)的数据，CRC 计算模块需要不同的时间，具体如下：

- 计算 32 位数据需要 4 个 AHB 时钟周期
- 计算 16 位数据需要 2 个 AHB 时钟周期
- 计算 8 位数据需要 1 个 AHB 时钟周期

输入缓存允许用户在写入第一个数据之后立刻写入第二个数据，无需等待之前的数据计算完毕。

用户可以动态地调整数据大小以减少计算特定字节数的数据时，访问本模块的次数。例如，计算一个 5 字节大小的数据，可以往本模块写入 1 个字的数据，再写入 1 个字节的数据以得到计算结果。

本模块支持输入数据反转功能，以适应不同的端序方案。根据 CRC_DR 寄存器的 REV_IN[1:0]，输入数据反转能以 8 位，16 位，32 位为单位进行反转。

例如，对于需要 CRC 运算的输入数据 0x1A2B3C4D：

- 以字节(8 位)为单位进行反转之后的数据是 0x58D43CB2
- 以半字(16 位)为单位进行反转之后的数据是 0xD458B23C

- 以字(32 位)为单位进行反转之后的数据是 0xB23CD458

CRC_CR 寄存器中 REV_OUT 位能将输出数据反转。这项操作是按位完成的，例如：输出数据是 0x11223344,被反转后成为 0x22CC4488

将 CRC_CR 寄存器的 RESET 位置 1，能使 CRC 计算器的初始值装载一个可编程的值(默认值为 0xFFFFFFFF)。

CRC 初始值能被编程成 CRC_INIT 寄存器所包含的值。当 CRC_INIT 寄存器被写时，CRC_DR 寄存器会被自动地初始化。

CRC_IDR 寄存器可以用来保存与 CRC 计算相关的临时值。它不受 CRC_CR 寄存器中 RESET 位的影响。

多项式可编程性

多项式的系数可以通过 CRC_POL 寄存器完全地进行自定义，多项式的位宽可以根据 CRC_CR 寄存器的 POLYSIZE[1:0]配置为 7 位，8 位，16 位，32 位。不支持偶数多项式，例如：偶数多项式的形式是 $x+x^2+\ldots+x^n$，而奇多项式的形式是 $1+x+x^2+\ldots+x^n$。

如果 CRC 数据的位宽小于 32 位，CRC 的值将会从 CRC_DR 的最低有效位开始读起。

为了得到可信的 CRC 计算结果，不能在 CRC 的计算期间，更改多项式的值或大小。从结果而言，如果 CRC 计算正在进行，在更改 CRC 多项式之前，应用程序必须要么复位 CRC，要么读取 CRC_DR 的值。

默认的多项式是 CRC-32（以太网）多项式：0x4C11DB7。

12.4. CRC 寄存器

12.4.1. CRC 数据寄存器 (CRC_DR)

偏移地址：0x00

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DR	RW	32'hFFFFFFFF	数据寄存器。 当写新数据时，作为输入寄存器。当被读时，保持之前 CRC 计算结果。 如果 CRC 数据小于 32 位，该寄存器的低位用于读或写正确的数据。

12.4.2. CRC 独立数据寄存器 (CRC_IDR)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDR[31:16]															

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	IDR[31:0]	RW	0	通用 32bit 数据寄存器 这些位用作一个字节的临时存储。该寄存器不会被 CRC_CR 寄存器的 RESET 位复位。 注：此寄存器不参与 CRC 计算，可以存放任何数据。

12.4.3. CRC 控制寄存器 (CRC_CR)

偏移地址：0x08

复位值：0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REV_OUT	REV_IN[1:0]		POLYSIZE[1:0]		Res.	Res.	RESET
								RW	RW	RW	RW	RW			W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	REV_OUT	RW	0	反转输出数据 这个位决定输出数据的位顺序是否反转 0：输出数据的位顺序不反转 1：输出数据的位顺序反转
6:5	REV_IN[1:0]	RW	0	反转输入数据 这 2 个位决定了输入数据的位顺序是否反转 00：位顺序不反转 01：输入数据每个字节都反转 10：输入数据每个半字都反转 11：数据数据每个字都反转
4:3	POLYSIZE[1:0]	RW	0	CRC 多项式位宽 这 2 个位决定了 CRC 多项式的位宽 00：32 位 CRC 多项式 01：16 位 CRC 多项式 10：8 位 CRC 多项式 11：7 位 CRC 多项式
2:1	Reserved	-	-	保留
0	RESET	W	0	该位被软件置位，用来复位 CRC 计算单元，并把 CRC_INIT 寄存器的内容复制给数据寄存器。该位只能被置位，由硬件自动清零。

12.4.4. CRC 初始值寄存器 (CRC_INIT)

偏移地址：0x10

复位值：0xFFFF FFFF

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
INIT[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INIT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	INIT	RW	32'hFFFFFFFF	可编程 CRC 初值 这个寄存器用于存储 CRC 初值.

12.4.5. CRC 多项式寄存器 (CRC_POL)

偏移地址：0x14

复位值：0x04C1_1DB7

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
POL[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POL[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	POL	RW	32'h04C1_1DB7	可编程 CRC 多项式 这个寄存器用于存储 CRC 多项式的系数 如果 CRC 多项式的宽度小于 32 位,将以右对齐的方式存放在本寄存器内。

13. 模数转换器(ADC)

13.1. ADC 简介

芯片具有 1 个 12 位的 SARADC（逐次逼近型模数转换器）。该模块共有 28 个要被测量的通道，包括 23 个外部通道和 5 个内部通道。外部通道包含 3 对差分通道。

各通道的转换模式可以设定为单次、连续、非连续模式。转换结果存储在左对齐或者右对齐的 16 位数据寄存器中。

模拟看门狗允许应用程序检测输入电压是否超出了用户定义的高或者低阈值。

ADC 实现了在低频率下运行，可获得很低的功耗。

13.2. ADC 主要特性

- 高性能
  - 12bit、10bit、8bit 和 6bit 分辨率可配置
  - ADC 转换时间：0.3 μs @ 12bit（3 MSps, 2.5 T SMP，f_{ADC_CLK} = 48 MHz）
  - 支持规则转换和注入转换
  - 自校准（软件启动）
  - 可编程的采样时间
  - 可编程的数据对齐模式（左对齐或者右对齐）
  - 规则通道转换支持 DMA
  - 配置支持 16 个规则序列转换
  - 配置支持 4 个注入序列转换
  - 配置支持 4 个注入通道转换数据寄存器
- 低功耗
  - 为低功耗操作，降低工作频率，而仍然维持合适的 ADC 性能
  - 自动延迟转换模式：防止以低频 PCLK 运行产生溢出
- 模拟输入通道
  - 23 个外部模拟输入通道
  - 1 个内部温度传感器（V_{TS}）通道
  - 1 个内部参考电压（V_{REFINT}）通道
  - 1 个内部检测电源（V_{CC}/3）通道
  - 1 个内部 OPA 通道
  - 1 个内部 DAC 通道
  - 3 对差分通道
- 转换操作启动方式
  - 软件启动
  - 可配置极性的硬件启动
- 转换模式

- 单次（序列）模式：1 次触发对所有选中的通道执行一次转换
- 连续模式：1 次触发连续转换被选择的通道，直到软件终止
- 非连续模式：1 次触发转换子序列配置通道数，多次触发后转换所有通道

■ 中断产生

- 单个通道采样结束
- 单个通道转换结束
- 序列转换结束
- 模拟看门狗事件
- 溢出事件
- ADC 就绪事件

■ 模拟看门狗

- 配置所有通道支持看门狗
- 配置选择的某一通道支持看门狗

■ 过采样

- 16 位数据寄存器
- 过采样率可在 2x 到 256x 之间进行调整
- 可编程数据最多可移位 8 位

■ 数据预处理

- 增益补偿
- 偏移补偿

13.3. ADC 功能描述

13.3.1. ADC 框图

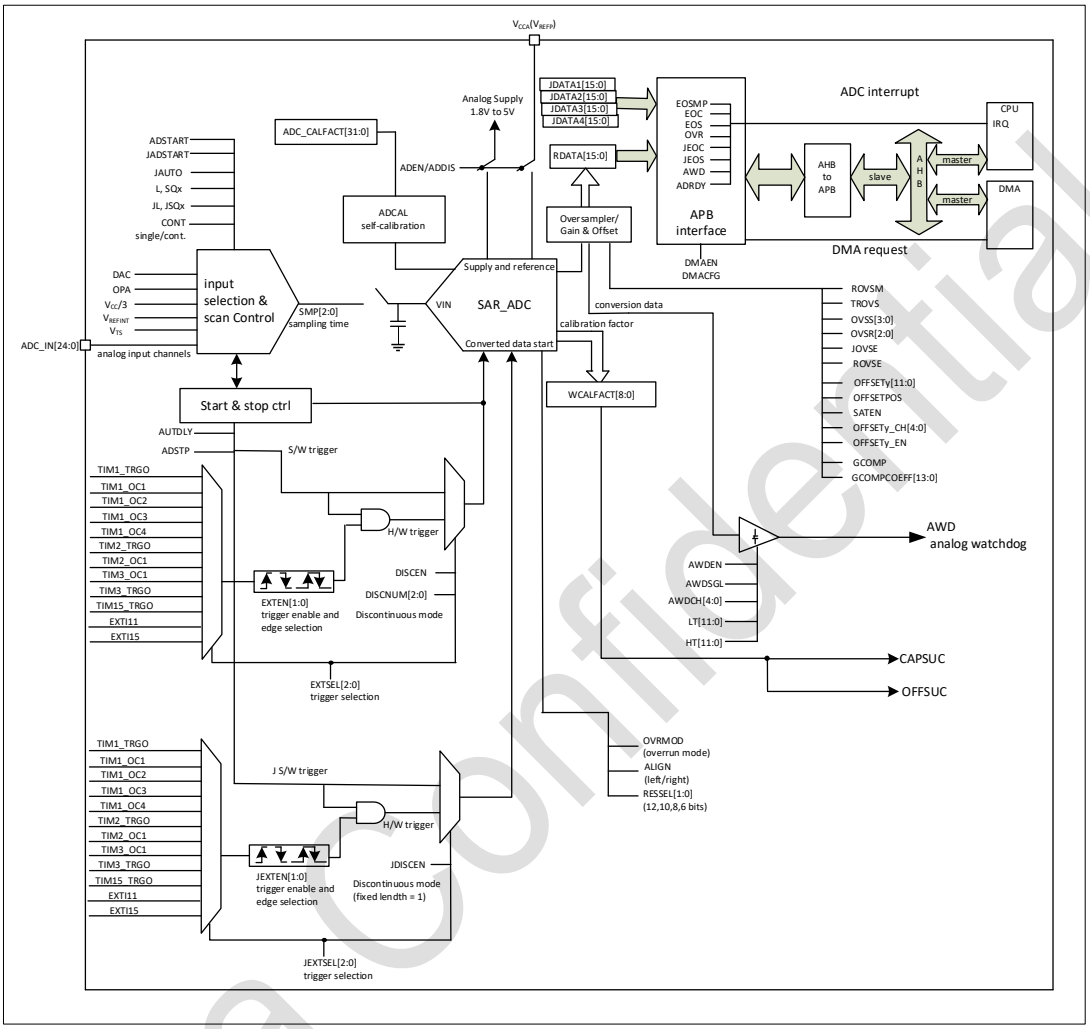


图 13-1 ADC 框图

13.3.2. ADC 单端和差分通道

ADC 通道可配置为单端输入或差分输入，通过 ADC_CCR.DIFF_EN 寄存器配置。当 DIFF_EN 为 1 时，部分通道为差分输入模式，否则为单端输入模式。

必须当 ADC 被禁用时（ADEN=0）配置。

在单端模式下，通道 i 的模拟电压为 VINP[i]和 V_{REF-}之差。

在差分模式下，通道 i 的模拟电压为 VINP[i]和 VINN[i]电压差。

差分模式下，输入电压范围为- V_{REF+}到 V_{REF+}，达到 2 x V_{REF+}的全刻度。当 VINP[i]等于 V_{REF-},VINN[i]等于 V_{REF+}则最大负端输入差分电压(-V_{REF+})对应 0x000 输出。当 VINP[i]等于 V_{REF+}, VINN[i]等于 V_{REF-}则最大正端输入差分电压(V_{REF+})对应 0xFFF 输出。当 VINP[i] 和 VINN[i] 连在一起，零输入差分电压对应 0x800 输出。

差分模式下的 ADC 灵敏度比单端模式下小两倍。

内部通道只采用单端模式。

下图为单端/差分通道的示意图：

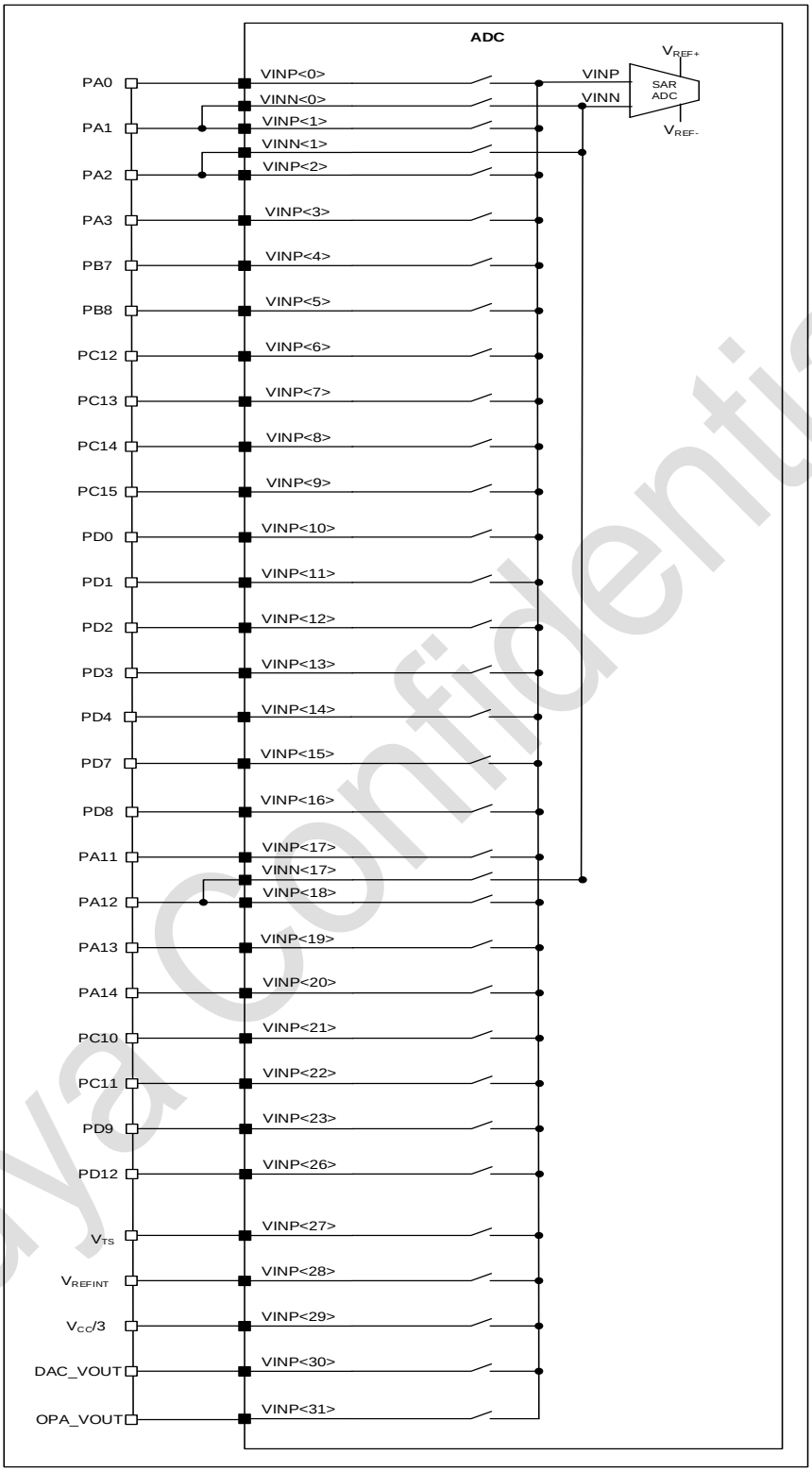


图 13-2 ADC 通道连接关系

支持 3 对差分对：

- PA0<->PA1，ADC_CCR.DIFF_EN=1 且通道选择 PA0 时，PA1 自动作为 INN 端
- PA1<->PA2，ADC_CCR.DIFF_EN=1 且通道选择 PA1 时，PA2 自动作为 INN 端

- PA11<->PA12，ADC_CCR.DIFF_EN=1 且通道选择 PA11 时，PA12 自动作为 INN 端

13.3.3. ADC 校准(ADCAL)

该 ADC 具有校准功能（软件启动）。ADC 通过校准过程计算校准系数，在下一次掉电之前，ADC 会应用此校准系数。校准过程中，应用不得使用 ADC，必须等待校准完成（ADEN=0）。

在使用 ADC 转换前，要进行校准操作。校准用于消除芯片和芯片之间的，由于工艺变化引起的误差。

ADC 软件校准

软件设置 ADCAL=1 可启动校准，校准只能在 ADC 未使能时 (ADEN=0)启动，且校准时 ADC_CLK 不能超过正常工作时最大 ADC_CLK（48 MHz）的 1/4，也即 12 MHz。

当校准完成后，ADCAL 被硬件清 0。校准完成后，可从 ADC_CALFACT 寄存器读出校准因子。校准因子会一直保持，直到产生系统复位。

当 ADC 的工作条件发生改变时（V_{CC} 改变是 ADC 偏移变化的主要因素，温度改变次之），推荐进行再次校准操作。

通过设置 ADC_CFGR2 寄存器中 CALNUM[2:0]位域来配置校准过程的重复次数，并对结果进行平均以得到更精确的校准结果。

内部校准寄存器可通过设置 ADC_CR 的 RSTCAL 复位，该位由硬件清除置 0。在校准寄存器被初始化后(即 RSTCAL 置 1 后)，该位即被清除。

校准的软件操作过程:

- 配置 ADC_CCR 寄存器，满足 ADC_CLK 时钟满足要求（不能超过正常工作时最大 ADC_CLK（48MHz）的 1/4）
- 确认 ADC_CR.ADEN=0
- 设置 ADC_CR.RSTCAL（该步骤是可选的）；
- 设置校准次数 ADC_CFGR2.CALNUM（该步骤是可选的）；
- 设置 ADC_CR.ADCAL=1
- 等待 ADC_SR.EOCAL=1
- 读取 ADC_CALFACT 寄存器的 CAPSUC 和 OFFSUC 状态位，确认校准是否成功。

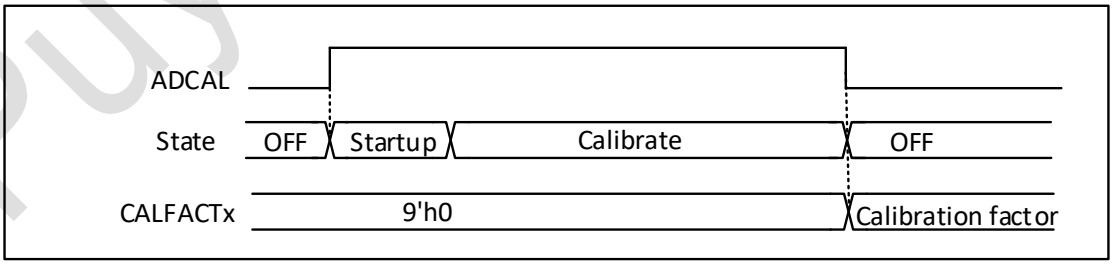


图 13-3 ADC 校准

13.3.4. 软件读写校准因子

软件可按照如下步骤读写校准因子:

写校准因子步骤

- 配置 ADC_CCR 寄存器，ADC_CLK 时钟源选择 PCLK
- 确保 ADEN=1，且没有转换正在进行，同时 ADCAL=0。
- 使能 WRVLD，FACTSEL[4:0]选择写入哪个电容，设置 WCALFACT[8:0]写入校准因子。
- 启动 AD 转换时，校准因子自动注入模拟 ADC。

读校准因子步骤

- 配置 ADC_CCR 寄存器，ADC_CLK 时钟源选择 PCLK
- 确保 ADEN=1，且没有转换正在进行，同时 ADCAL=0。
- 使能 RDVLD，FACTSEL[4:0]选择读哪个电容
- 读 RCALFACT[8:0]校准因子。

13.3.5. ADC 开关控制(ADEN, ADDIS, ADRDY)

芯片上电复位后，ADC 模块不使能，处于不工作状态(ADEN=0)。

ADEN 位用于控制位开启或关闭 ADC。ADC 启用后，在开始准确转换之前，ADC 需要一个稳定时间 t_{STAB}，如下图。有两个控制位用于启用或禁用 ADC：

- ADEN=1 时，启用 ADC。当 ADC 准备好进行操作时，标志 ADRDY 将被设置。
- ADDIS=1 时，禁用 ADC。一旦模拟 ADC 完全被禁用（不工作状态），硬件会自动清除 ADEN 和 ADDIS。

ADC 规则转换由设置 ADSTRART 来启动，根据 EXTEN 的配置，可以立即开始转换（软件启动）或者等待硬件触发后开始转换。

ADC 注入转换由设置 JADSTRART 来启动，根据 JEXTEN 的配置，可以立即开始转换（软件启动）或者等待硬件触发后开始转换。

13.3.5.1. 软件启用 ADC 的步骤

1. 通过写入'1'清除 ADC_ISR 寄存器中的 ADRDY 位。
2. 设置 ADEN=1 以启用 ADC。
3. 等待 ADRDY=1（在 ADC 启动时间后，ADRDY 将被设置）。如果已通过将 ADC_IER 寄存器中的 ADRDYIE 位置 1 来使能中断，可通过中断进行处理。

13.3.5.2. 软件禁用 ADC 的步骤

1. 检查 ADSTART 和 JADSTART 是否均为 0，确保没有正在进行的转换。如有需要，通过设置 ADC_CR 寄存器中 ADSTP=1 或 JADSTP=1 停止任何正在进行的常规或注入式转换，然后等待 ADSTP=0 和 JADSTP=0。
2. 设置 ADDIS=1 以禁用 ADC。
3. 如果应用程序有此需求，请等待直至 ADEN=0，即模拟 ADC 实际上被禁用（一旦 ADEN=0，ADDIS 将自动复位）。
4. 通过写入'1'清除 ADC_ISR 寄存器中的 ADRDY 位（可选）。

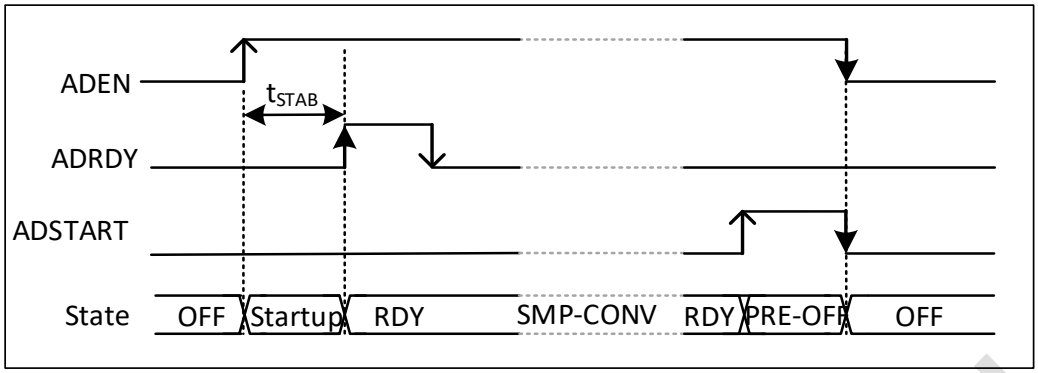


图 13-4 使能/禁止 ADC

13.3.6. ADC 控制位寄存器操作

ADC 寄存器操作，符合以下原则：

- ADC 已启用且没有待处理的禁用 ADC 请求（即 ADEN 为 1 且 ADDIS 为 0），软件才被允许写入 ADC_CR 寄存器中的 ADSTART、JADSTART 和 ADDIS 控制位。
- 当 ADC 已启用、可能正在转换，并且没有待处理的禁用 ADC 请求（即 ADSTART 或 JADSTART 等于 1 且 ADDIS 为 0）时，软件才被允许写入 ADC_CR 寄存器中的 ADSTP 或 JADSTP 控制位。
- 在 ADEN=1 且 ADSTART 和 JADSTART 为 0 时，软件才能配置 ADDIS=1；
- ADC_CFGR，ADC_CFGR2，ADC_SMPRx，ADC_SQRy，ADC_JSQR，ADC_OFRy，ADC_GCOMP，ADC_TR，ADC_IER 寄存器操作满足：
  - 规则序列相关寄存器，在 ADEN=1 且 ADSTART=0 时配置
  - 注入序列相关寄存器，在 ADEN=1 且 JADSTART=0 时配置

注意：硬件并未提供防止这些禁止写访问的保护措施，若发生此类写操作，ADC 的行为可能会进入未知状态。为了从这种状况中恢复，需要将 ADC 禁用（将 ADEN 清零至 0，并同时清除 ADC_CR 寄存器中的所有位）。

13.3.7. ADC 时钟

ADC 具有双时钟域架构，ADC 时钟(ADC_CLK) 独立 APB 时钟(PCLK)。ADC_CLK 可由两种可能的时钟源产生，如下图所示：

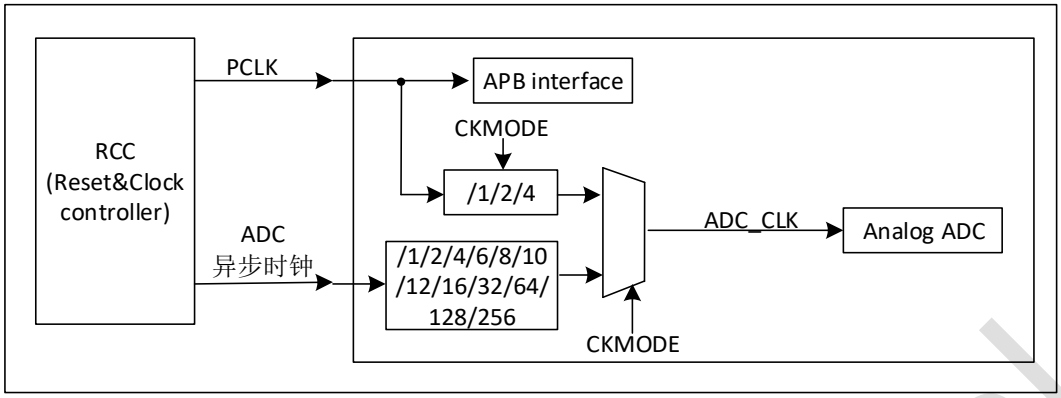


图 13-5 ADC 时钟结构

模拟 ADC 的输入时钟可在两个不同的时钟源之间进行选择：

a) ADC 时钟可以是名为“ADC 异步时钟”的特定时钟源，该时钟源独立于 APB 时钟，并与 APB 时钟异步。要选择此时钟方案，必须将 ADC_CCR 寄存器的 CKMODE[1:0] 位复位。

b) ADC 时钟可由 ADC 总线接口的 APB 时钟除以一个可编程因子（1、2 或 4，具体根据 CKMODE[1:0] 位而定）来提供。要选择此时钟方案， ADC_CCR.CKMODE[1:0] 寄存器位不能配置为“00”。

在选项 a) 中，对 ADC_CCR 寄存器中的 PRESC[3:0] 位进行编程时，生成的 ADC 时钟最后会除以预分频系数（1、2、4、6、8、10、12、16、32、64、128、256）。

选项 a) 的优点在于无论选择哪种 APB 时钟方案，都可以达到最大 ADC 时钟频率。

注意： 选择 CKMODE[1:0]=11（PCLK 1 分频）时，用户必须确保 PCLK 的占空比为 50%。可通过选择占空比为 50% 的系统时钟并在 RCC 中不配置 APB 预分频器来实现（RCC_CFGR 的 HPRE 和 PPRE 都配置为不分频）。

注：仅支持 ADC_CLK 频率小于或等于 PCLK 频率。

13.3.8. ADC 通道选择(SQRx, JSQR)

共有 28 路复用通道：

- 23 个从 GPIO 引脚引入的模拟输入 (ADC_IN0...ADC_IN24 可选)
- 5 个内部模拟输入(温度传感、内部参考电压、V_{CC}/3、OPA 输出和 DAC 输出)

ADC 可以转换一个单一通道或自动扫描一个序列通道。ADC 通道选择可通过 ADC_SQRx 或 ADC_JSQR 配置。

ADC 将转换分为两组：规则转换和注入转换。每个组包含一个转换序列，该序列可按任意顺序在任意通道上完成。例如，可按以下顺序对序列进行转换：ADC_IN3、ADC_IN8、ADC_IN2、ADC_IN2、ADC_IN0、ADC_IN2、ADC_IN2、ADC_IN15。所有通道都可以按注入或规则通道组进行转换。

- 一个**规则转换**组最多由 16 个转换构成。转换序列的规则通道及其转换顺序在 ADC_SQRx 寄存器中选择。规则转换组中的序列长度必须写入 L[3:0] 。
- 一个**注入转换**组最多由 4 个转换构成。注入通道及其转换顺序在 ADC_JSQR 寄存器中选择。注入转换组中的序列长度必须写入 JL[1:0] 位。

在规则转换可能发生期间，不得修改 ADC_SQRy 寄存器。为此，首先必须通过写入 ADSTP=1 来停止 ADC 的规则转换。

13.3.9. ADC 可编程采样时间(SMP)

在启动 ADC 转换之前，ADC 需要在被测电压源和内嵌采样电容间建立一个直接连接。采样时间必须足够长以便输入电压源对内嵌电容充电到输入电压的水平。

可编程采样时间根据输入电压的输入阻抗来调整转换速度。

ADC 采样输入电压所用的 ADC 时钟个数可用 ADC_SMPRx 寄存器中的 SMP[2:0]位来进行修改，每个通道可独立配置不同的采样时间。如有应用需求，则可用软件改变和适应不同通道间的采样时间。

总转换时间计算如下：

t_{CONV} = （采样时间 + (转换分辨率+0.5) ） x ADC 时钟周期

例如：

当 ADC_CLK = 16 MHz，分辨率为 12 位，且采样时间为 2.5 个 ADC 时钟周期：

t_{CONV} = （2.5 + 12.5） x ADC 时钟周期= 15 x ADC 时钟周期 = 0.9375 µs

EOSMP 标志位用来表明采样阶段的结束。

对采样时间的限制：

- 对于每个通道，SMP[2:0]位必须按照数据手册中 ADC 特性部分所指定的最小采样时间进行编程。

13.3.10. ADC 单次转换模式 (CONT=0,DISCEN=0)

单次转换模式下，ADC 执行一次序列转换，转换所有被选的通道。当 ADC_CFGR 寄存器中的 CONT=0，DISCEN=0 时，ADC 为单次转换模式。

该模式可由下述方法启动（硬件触发）：

- 配置 ADC_CFGR.EXTEN 和 ADC_JSQR.JEXTEN 寄存器为非 2’b00
- 在 ADC_CR 寄存器中设置 ADSTART 位（规则转换）
- 在 ADC_CR 寄存器中设置 JADSTART 位（注入转换）
- ADSTART=1 且硬件触发事件（规则转换）
- JADSTART=1 且硬件触发事件（注入转换）

规则转换，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_DR 中。
- EOC(转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断。

规则转换，所有通道序列转换完成后：

- EOS(序列结束) 标志置位(EOC 同时也会置位)
- 若 EOSIE 位置位则产生一个中断

注入转换，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_JDRx 中。
- JEOC(转换结束标志)标志置位
- 若 JEOCIE 位置位则产生一个中断。

注入转换，所有通道序列转换完成后：

- JEOS(序列结束) 标志置位(JEOC 同时也会置位)
- 若 JEOSIE 位置位则产生一个中断

转换结束后，ADC 转换停止直到新的触发事件或 ADSTART/JADSTART 重新置位。

若转换单一通道，则可配置序列长度为 1。

13.3.11. ADC 连续转换模式 (CONT=1,DISCEN=0)

该模式仅针对规则转换。

在连续转换模式中，当软件或硬件触发事件产生，ADC 执行一个序列转换。转换所有的通道且自动重新开始执行相同的序列转换。

当寄存器 ADC_CFGR 中的 CONT=1 时，ADC 选择为连续转换模式。

ADC 转换可由下述两种方法启动：

- 在 ADC_CR 寄存器中设置 ADSTART 位
- ADSTART=1 且硬件触发事件触发

在序列通道的转换期间，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_DR 中
- EOC (转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断。

通道序列转换完成后：

- EOS (序列结束)标志置位(EOC 同时也会置位)
- 若 EOSIE 位置位则产生一个中断

一次序列转换结束后，ADC 立即重新转换相同的序列通道。

注：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

ADC 不能同时处于非连续（discontinuous）转换模式和连续（continuous）转换模式 (即禁止同时配置 DISCEN=1，CONT=1)

注入通道不支持连续转换模式。唯一的例外是当注入通道被配置为在连续模式下的规则通道之后自动转换时（使用 JAUTO 位）。

13.3.12. ADC 非连续转换模式 (DISCEN=1 or JDISCEN=1)

13.3.12.1. 规则组模式

该模式由设置 ADC_CFGR 寄存器中的 DISCEN 位来开启。ADC 将选择的一个序列分成子序列(序列长度 1-8)，通过多次外部触发子序列，完成整个序列转换。DISCNUM 规定了规则通道子序列长度，ADC_SQRx 规定了一个规则序列所有转换通道，其中 L[3:0]位规定了该序列长度。

例如：

DISCEN=1, DISNUM=2，需要转换的通道为：0, 3, 7, 10，且 ADC_SQR0 中 SQ1=0，SQ2=3，SQ3=7，SQ4=10：

- 第一次触发：通道 0，3 被转换，每次转换完成产生一个 EOC 事件
- 第二次触发：通道 7，10 被转换，每次转换完成产生一个 EOC 事件，并会在通道 10 转换完成后产生 EOS 事件
- 第三次触发：通道 0，3 被转换，每次转换完成产生一个 EOC 事件
- ...

DISCEN=0，需要转换的通道为：0, 3, 7, 10，且 ADC_SQR0 中 SQ1=0，SQ2=3，SQ3=7，SQ4=10：

- 第一次触发：转换整个序列，依次为通道 0, 3, 7 和 10。

每次转换完成，产生一个 EOC 事件，转换到最后一个通道，除产生 EOC 外，还产生一个 EOS 事件。

- 任何触发事件都会重新开始完整的序列转换。

注：让 ADC 同时处于连续模式和非连续转换模式是不可能的事情，即禁止同时配置 DISCEN=1，CONT=1。

13.3.12.2. 注入组模式

该模式由设置 ADC_CFGR 寄存器中的 JDISCEN 位来开启。一个外部触发信号启动 ADC_JSQR 描述通道的 1 次转换，直到序列中所有转换完成为止。总序列长度由 JL[1:0]位定义。

例如：

JDISCEN=1,需要转换的通道为：1、2、3，且 ADC_JSQR 中 JSQ1=1，JSQ2=3，JSQ3=7：

- 第一次触发：转换通道 1，每次转换完成产生一个 JEOC 事件
- 第二次触发：转换通道 2，每次转换完成产生一个 JEOC 事件
- 第三次触发：转换通道 3，每次转换完成产生一个 JEOC 事件，序列结束产生 JEOS 事件
- 第四次触发：通道 1 被转换，每次转换完成产生一个 JEOC 事件
- ...

注：

- 当转换完成所有注入通道，下次触发启动第 1 个注入通道的转换
- 不能同时使用自动注入和非连续模式：当 JAUTO 置 1 时， DISCEN 和 JDISCEN 位必须通过软件保持清零状态。
- 必须避免同时为规则和注入组设置非连续模式

13.3.13. ADC 注入通道管理

13.3.13.1. 触发注入模式

触发注入模式，ADC_CFGR.JAUTO 位必须为 0。

- 通过外部触发器或通过设置 ADC_CR 寄存器中的 ADSTART 位启动一组规则通道的转换。
- 如果发生外部注入触发，或者在转换一组规则通道期间设置了 ADC_CR 寄存器中的 JADSTART 位，则当前转换会复位，并启动注入通道序列 (所有注入通道都转换一次)。
- 然后，从上次中断的规则转换中恢复规则组通道的转换。

- 如果在注入转换期间发生规则触发事件（多个触发被处理为 1 个的情况），则不会中断注入转换，而是在注入序列的末尾执行规则序列。（针对注入没有打断规则的情况，如果注入前打断过规则，则先恢复并完成被打断的规则）。特殊地，如果注入转换+等待（AUTDLY=1）+软件触发规则，那么软件需要在 JEOS 清零后再触发 ADSTART 启动规则转换。

注意：当使用触发注入时，必须确保触发事件之间的间隔长于注入序列。例如，如果序列长度为 30 个 ADC 时钟周期（采样时间为 2.5ADC_CLK 的两次转换），则触发器之间的最小间隔必须为 31 个 ADC 时钟周期。

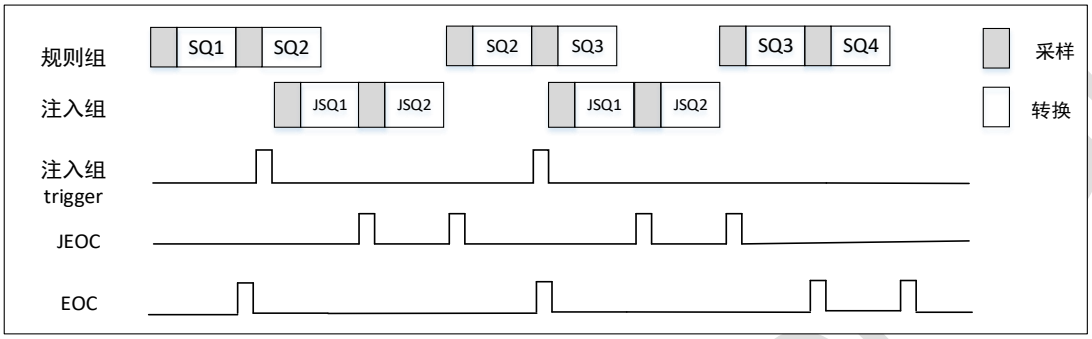


图 13-6 ADC 触发注入时序

13.3.13.2. 自动注入模式

如果设置了 ADC_CFGR 寄存器中的 JAUTO 位，则注入组中的通道在规则通道组之后自动转换。这可以用来转换在 ADC_SQRy 和 ADC_JSQR 中编程的最多 20 个转换序列寄存器（规则通道最多 16 个注入最多 4 个）。

在这种模式下，将 ADC_CR 寄存器中的 ADSTART 位设置为 1 启动规则转换，然后是注入转换(JADSTART 必须保持清零)。设置 ADSTP 位会终止规则转换和注入转换(不能使用 JADSTP 位)。

在此模式下，必须禁用注入通道上的外部触发。

如果除了 JAUTO 位之外还设置了 CONT 位，则连续转换“规则通道+注入通道”。

注意:不可能同时使用自动注入和非连续模式。

当 DMA 在 JAUTO 模式下用于导出规则转换序列的数据时，需要将其编程为循环模式(在 DMA_CCRx 寄存器中设置 CIRC 位)。如果 CIRC 位复位(单次模式)，在 DMA 传输完成事件发生时，JAUTO 序列将停止。

13.3.14. ADC 转换启动(ADSTART，JADSTART)

软件通过设置 ADSTART=1 启动 ADC 规则转换。

当 ADSTART 设置，则规则转换：

- 当 EXTEN=2’b00(软件触发) 时，立即开始
- 当 EXTEN 不等于 2’b00 时，检测到选中的规则硬件触发源的有效边沿开始

软件通过设置 JADSTART=1 启动 ADC 注入转换。

设置 JADSTART 后，则注入转换:

- 当 JEXTEN[1:0] = 2’b00(软件触发)时，立即开始
- 如果 JEXTEN[1:0]不等于 2’b00 时，检测选中的注入硬件触发源的有效边沿开始

注意:在自动注入模式下(JAUTO=1)，使用 ADSTART 位启动规则转换，然后是自动注入转换(JADSTART 必须保持清除)。

ADSTART 和 JADSTART 位也用于说明目前 ADC 转换操作是否正在进行。当 ADC 处于空闲时，可在 ADSTART=0 和 JADSTART=0 重新配置 ADC。

ADSTART 位可由硬件清除：

■ 单次或连续转换模式由软件触发 (CONT=0，EXTSEL=2'b00)

在任何规则转换序列结束时（EOS=1）或当 DISCEN = 1 时的子序列转换结束时

■ 非连续转换模式由软件触发 (CONT=0，DISCEN=1, EXTSEL=2'b00)

- 在 DISNUM 配置的子序列转换结束后(子序列最后一个通道的 EOC=1)

■ 在所有的情况下(CONT=X，EXTSEL=X)

- 在软件配置 ADSTP 后

注意：

1. 在连续模式 (CONT=1) 下，ADSTART 位不能由 EOS 引发的硬件清除，因为序列会自动重新开始转换。

2. 当单次转换模式选择硬件触发(CONT=0 且 EXTSEL≠2'b00)，则当 EOS 标志置位后，ADSTART 不会被硬件清 0。这就避免了需要软件重新设置 ADSTART 位且确保不会错过硬件触发事件。

JADSTART 由硬件清除：

■ 软件触发单次转换模式 (JEXTSEL = 2'b00)

- 如果 JDISCEN=1，在注入转换序列结束(JEOS 产生)或当 JDISCEN = 1 时的子序列转换结束

■ 在所有情况下(JEXTSEL=x)

- 在软件配置 JADSTP 后。

注意：当选择软件触发时，如果 EOC 标志仍然高，则不应将 ADSTART 置位。

13.3.15. ADC 时序

转换所用的时间由启动转换时间和与转换分辩率有关的逐次逼近时间组成。

$t_{ADC} = t_{SMPL} + t_{SAR} = [\ 2.5\ |min + 12.5\ |12bit\ ] \times t_{ADC\_CLK}$

$t_{ADC} = t_{SMPL} + t_{SAR} = 52.083ns\ |min + 260.375ns\ |12bit = 312.458ns\ |min\ (for\ f_{ADC\_CLK} = 48\ MHz)$

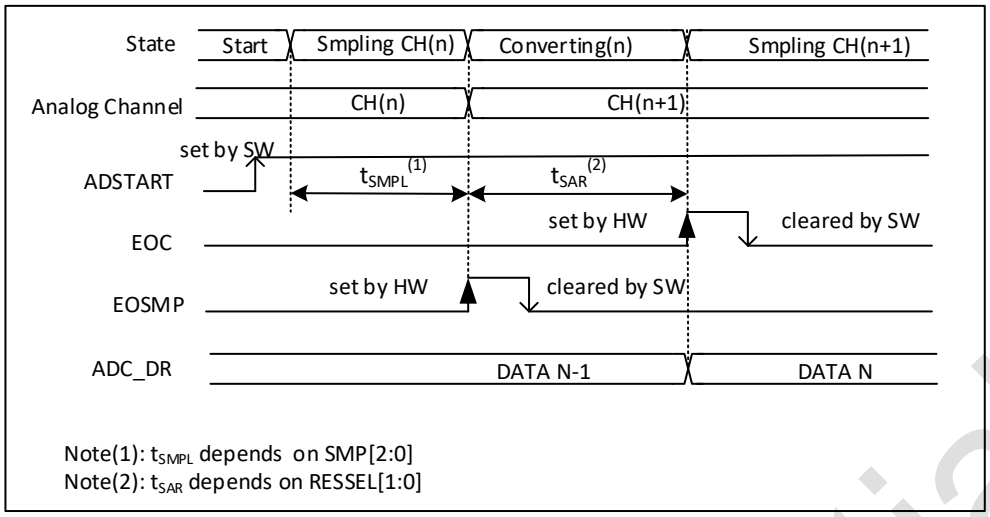


图 13-7 模数转换时序

13.3.16. ADC 停止正在进行的转换 (ADSTP, JADSTP)

通过设置 ADC_CR 寄存器中的 ADSTP=1 可以停上当前正在进行的规则转换，通过设置 ADC_CR 寄存器中的 JADSTP=1 可以停上当前正在进行的注入转换。

停止转换将复位正在进行的 ADC 操作。然后可以重新配置 ADC(例如:改变通道选择或触发器)，为新的操作做好准备。

请注意，在规则转换仍在运行时可以停止注入转换，反之亦然。例如，这允许在规则转换仍在运行时重新配置注入转换序列和触发(反之亦然)。

当 ADSTP 位由软件置位时，任何正在进行的规则转换被中止，部分结果被丢弃(ADC_DR 寄存器不更新当前转换)。

当 JADSTP 位由软件设置时，任何正在进行的注入转换将被中止，部分结果被丢弃(ADC_JDRy 寄存器不会更新为当前转换结果)。

扫描序列也被中止和复位(这意味着重新启动 ADC 将重新启动一个新序列)。

一旦这个过程完成，位 ADSTP/ADSTART(在规则转换的情况下)，或 JADSTP/JADSTART(在注入转换的情况下)被硬件清除，软件必须等待 ADSTART/JADSTART=0，然后才能开始进行新的转换。

注意:在自动注入模式下(JAUTO=1)，设置 ADSTP 位会同时终止规则转换和注入转换(不能使用 JADSTP)。

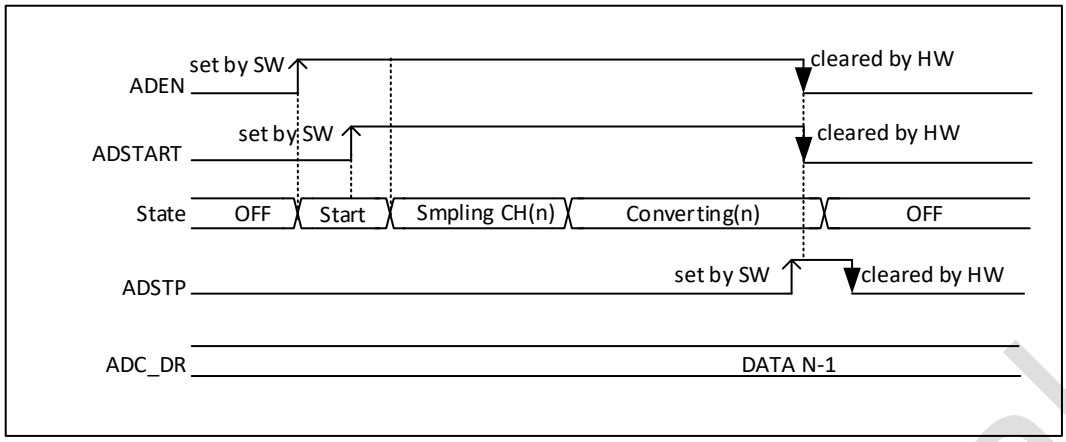


图 13-8 ADSTP 停止规则转换

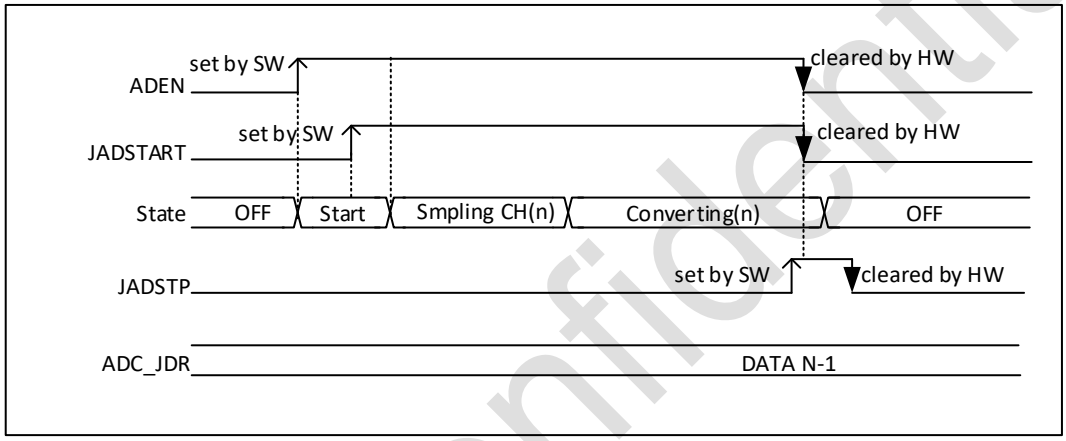


图 13-9 JADSTP 停止注入转换

13.3.17. ADC 硬件触发转换和触发极性 (EXTSEL, EXTEN, JEXTSEL, JEXTEN)

一次转换或一个序列的转换可由软件或外部事件(例如：定时器、输入引脚) 触发。如果 EXTEN[1:0]控制位(用于规则转换)或 JEXTEN[1:0]位(用于注入转换) 不等于 2'b00，则外部事件能够以所选极性触发转换。

一旦软件设置 ADSTART=1，规则触发选择就有效，一旦软件设置位 JADSTART=1，注入触发选择就有效。

在转换进行时发生的硬件触发会被忽略。ADSTP 为高电平期间内的任何规则触发将被忽略，JADSTP 为高电平期间内的任何注入触发将被忽略。

- 如果位 ADSTART=0，任何规则硬件触发都会被忽略。
- 如果位 JADSTART=0，任何注入硬件触发都将被忽略。

下表提供了 EXTEN[1:0]和 JEXTEN[1:0]值与触发源极性之间的对应关系。

表 13-1 EXTEN [1:0]与触发源极性对应关系

EXTEN[1:0]	源
00	规则硬件触发检测禁止，使能软件触发
01	规则硬件上升沿触发
10	规则硬件下降沿触发

11	规则硬件上升和下降沿触发

注：在规则转换时不能改变外部触发极性。

表 13-2 JEXTEN [1:0]与触发源极性对应关系

JEXTEN[1:0]	源
00	注入硬件触发检测禁止，使能软件触发
01	注入硬件上升沿触发
10	注入硬件下降沿触发
11	注入硬件上升和下降沿触发

注：在注入转换时不能改变外部触发极性。

EXTSEL 和 JEXTSEL 控制位从最多 16 个可能的事件中选择哪些事件可以触发规则组和注入组的转换。

规则组转换可以通过注入触发中断。

注意:不能动态更改规则和注入转换的触发源选择。

13.3.18. ADC 可编程分辨率 (RES) – 快速转换

用降低转换分辨率来获取更快的转换时间 (t_{SAR}) 是可行的。转换分辨率可通过设置 ADC_CFGR 寄存器中的 RES[1:0] 来配置为 12/10/8/6 位模式。当应用不需要高精度数据时，可用低的转换分辨率来加快转换时间。

分辨率模式减少逐次逼近的转换时间，如下表所示：

RES[1:0]	t_{SAR} (ADC 时钟周期)	t_{SAR}(ns) @ f_{ADC} = 24MHz	t_{SMP} (ADC 时钟周期)	t_{ADC}(t_{SMP} = 2.5) (ADC 时钟周期)	t_{SAR}(ns) @ f_{ADC} = 24MHz
12	13.5	562.5ns	2.5	16	666.7ns
10	11.5	479ns	2.5	14	583ns
8	9.5	395.8ns	2.5	12	500ns
6	7.5	312.5ns	2.5	10	416.7ns

13.3.19. ADC 转换/采样结束 (EOC, JEOC, EOSMP)

ADC 将每个规则转换结束(EOC)事件和每个注入转换(JEOC)事件通知应用程序。

一旦 ADC_DR 寄存器中有新的规则转换数据可用，ADC 就设置 EOC 标志。如果设置了位 EOCIE，则可以产生中断。EOC 标志由软件通过向其写入 1 或通过读取 ADC_DR 来清除。

一旦新的注入转换数据在 ADC_JDRy 寄存器中可用，ADC 就会设置 JEOC 标志。如果位 JEOCIE 被设置，可以产生中断。JEOC 标志由软件通过向其写入 1 或通过读取相应的 ADC_JDRy 寄存器来清除。

ADC 还通过设置状态位 EOSMP(仅用于规则转换)来通知采样阶段的结束。软件对 EOSMP 标志写入 1 即可清除。如果 EOSMPIE 位设置，可以产生中断。

13.3.20. ADC 转换序列结束 (EOS，JEOS)

当规则序列完成(EOS)和注入序列完成(JEOS)，ADC 通知应用程序。

一旦 ADC_DR 寄存器中有了规则转换序列的最后一个数据，ADC 就会设置 EOS 标志。如果设置了位 EOSIE，则可以产生中断。EOS 标志由软件通过向其写入 1 来清除。

一旦注入转换序列的最后一个数据完成，ADC 就设置 JEOS 标志。如果位 JEOSIE 被设置，可以产生中断。JEOS 标志由软件通过向它写入 1 来清除。

13.3.21. ADC 时序举例(单次/连续模式，硬件/软件触发)

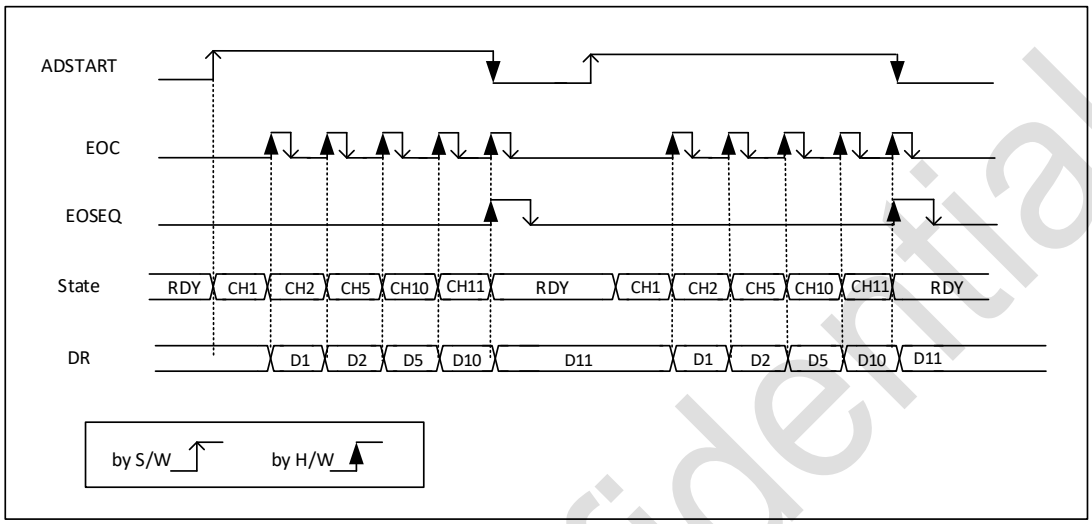


图 13-10 规则单次模式，软件触发

1. EXTEN=0x0, CONT=0
2. SQ1~5 分别配置为通道 1、2、5、10、11，AUTDLY=0

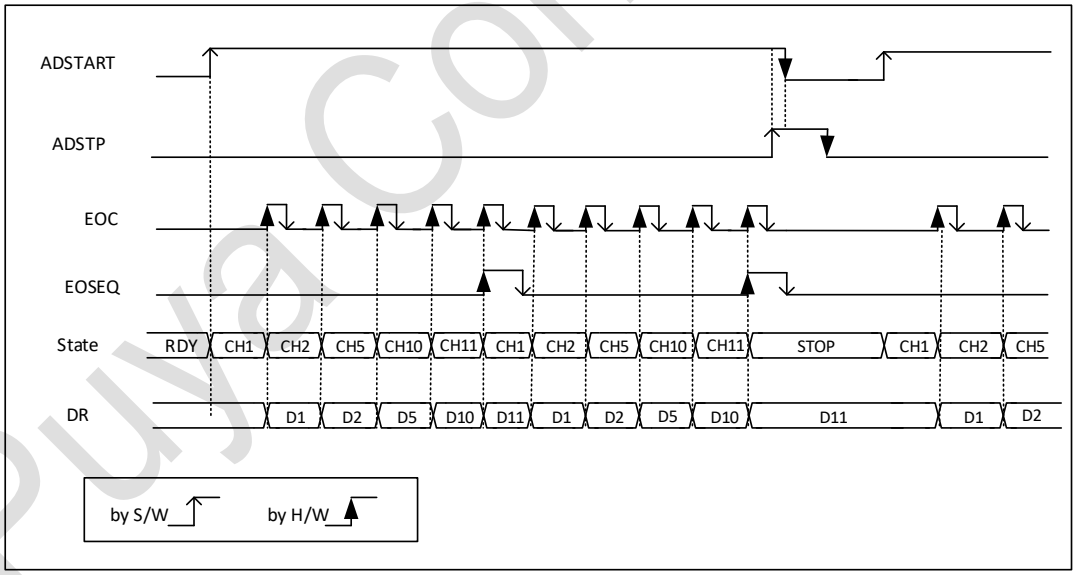


图 13-11 规则连续模式，软件触发

1. EXTEN=0x0, CONT=1,
2. SQ1~5 分别配置为通道 1、2、5、10、11，AUTDLY=0

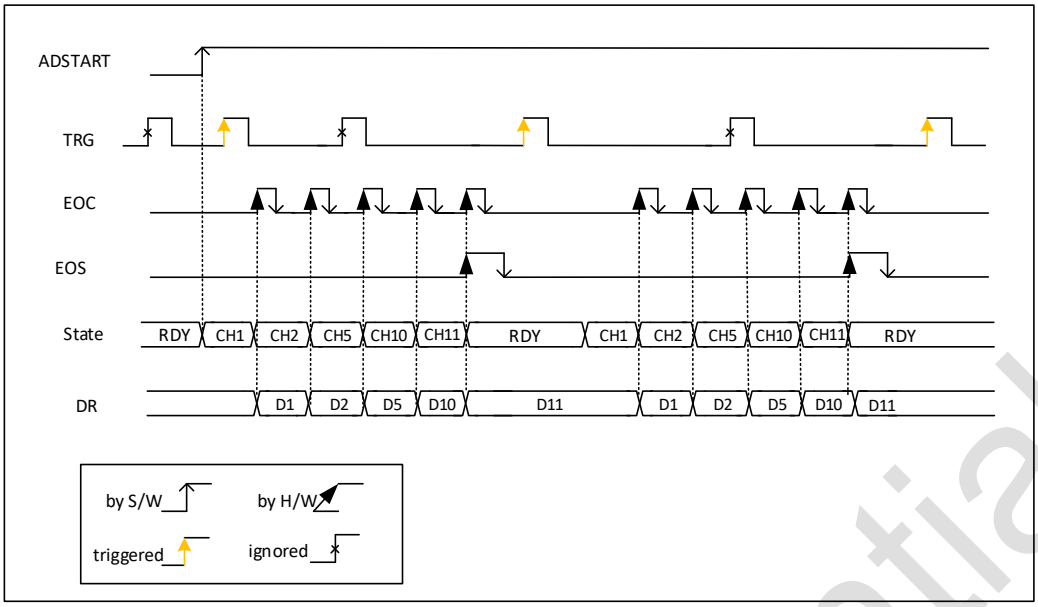


图 13-12 规则单次模式，硬件触发

1. EXTSEL=TRGx, EXTEN=0x1（ 上升沿), CONT=0
2. SQ1~5 分别配置为通道 1、2、5、10、11，AUTDLY=0

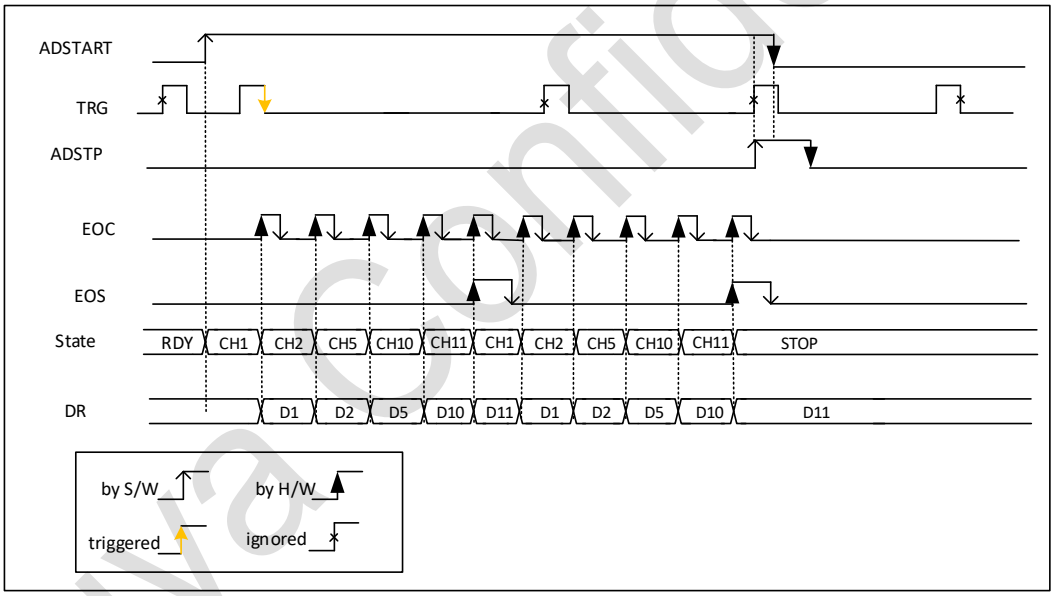


图 13-13 规则连续转换，硬件触发

1. EXTSEL=TRGx, EXTEN=0x2（ 下降沿), CONT=1
2. SQ1~5 分别配置为通道 1、2、5、10、11，AUTDLY =0

13.3.22. ADC 数据管理

13.3.22.1. 数据寄存器和数据对齐（ADC_DR, ADC_JDRy，ALIGN)

在每次规则转换结束(当 EOC 事件产生时)，转换的结果数据被存放到 16 位宽 ADC_DR 数据寄存器中。

在每次注入转换结束(当 JEOC 事件产生时)，转换的结果数据被存放到 16 位宽 ADC_JDRy 数据寄存器中。

ADC_DR 和 ADC_JDRy 数据格式与所配置的数据对齐和转换分辨率有关。ADC_CFGR 寄存器中的 ALIGN 位用于选择数据存储的对齐方式，数据可选为右对齐（ALIGN=0）或左对齐(ALIGN=1)。

下表为数据对齐方式和分辨率关系：（过采样无效 ROVSE=0，JOVSE=0）

表 13-3 数据对齐和分辨率

ALIGN	RES	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0	0x0				DATA[11:0]											
	0x1	0x0						DATA[9:0]									
	0x2	0x0								DATA[7:0]							
	0x3	0x0										DATA[5:0]					
1	0x0	DATA[11:0]												0x0			
	0x1	DATA[9:0]										0x0					
	0x2	DATA[7:0]								0x0							
	0x3	0x0								DATA[5:0]						0x0	

特殊情况：当左对齐时，数据以半字为单位对齐，除非分辨率设置为 6 位。在这种情况下，数据以字节为单位对齐。

注意：过采样模式下不支持左对齐。当设置了 ROVSE 和/或 JOVSE 位时，ALIGN 位值被忽略，ADC 只提供右对齐的数据。

13.3.22.2. 偏移(OFFSETy，OFFSETy_CH，OFFSETy_EN)

通过设置 ADC_OFRy 寄存器中 OFFSETy_EN = 1，可以将偏移 y（y = 1、2、3、4）施加到通道。应用可通过 ADC_OFRy 寄存器的位 OFFSETy_CH 选择偏移的通道。在这种情况下，转换结果减去 OFFSETy 中的用户定义的偏移量，结果可能是负值，因此读取的数据是有符号的，用 SEXT 位表示扩展的符号值。

注意：过采样模式下不支持偏移校正。设置 ROVSE 和/或 JOVSE 位时，ADC_OFRy 寄存器中 OFFSETy_EN 位的值被忽略（视为复位值）。

下表为 offset 计算值与数据分辨率的关系：

表 13-4 Offset 计算值 VS 分辨率

分辨率(RES[1:0])	原始转换结果和 offset 减法		结果	描述
	原始转换结果,左对齐	偏移		
00:12bit	DATA[11:0]	OFFSET[11:0]	12bit 有符号数	-
01:10bit	{DATA[11:2],00}	OFFSET[11:0]	10bit 有符号数	用户必须配置 OFFSET[1:0]=00
10:8bit	{DATA[11:4],0000}	OFFSET[11:0]	8bit 有符号数	用户必须配置 OFFSET[3:0]=0000
11:6bit	{DATA[11:6],000000}	OFFSET[11:0]	6bit 有符号数	用户必须配置 OFFSET[5:0]=000000

从与通道“i”对应的 ADC_DR（规则通道）或 ADC_JDRy（注入通道，y=1,2,3,4）读数据时：

--如果对应通道的偏移之一被使能（位 OFFSETy_EN = 1），则读数据是有符号的。

--如果此通道的四个偏移均未启用，则读数据无符号。

下图分别表示有符号数和无符号数数据对齐的方式。

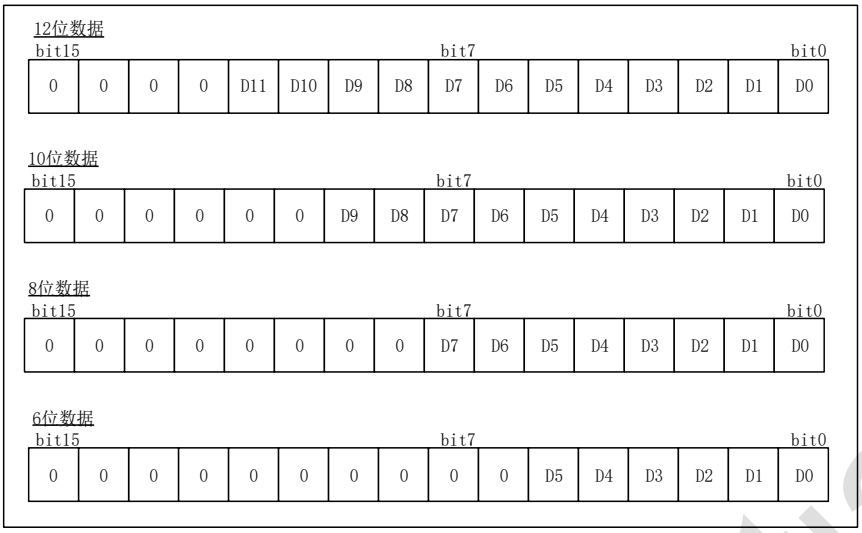


图 13-14 右对齐（偏移未使能，无符号数）

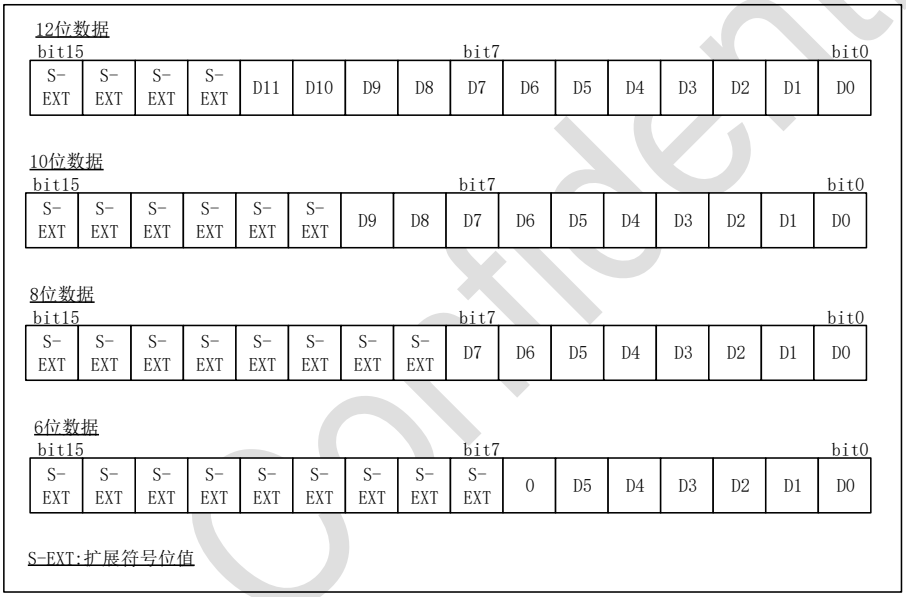


图 13-15 右对齐（偏移使能，有符号数）

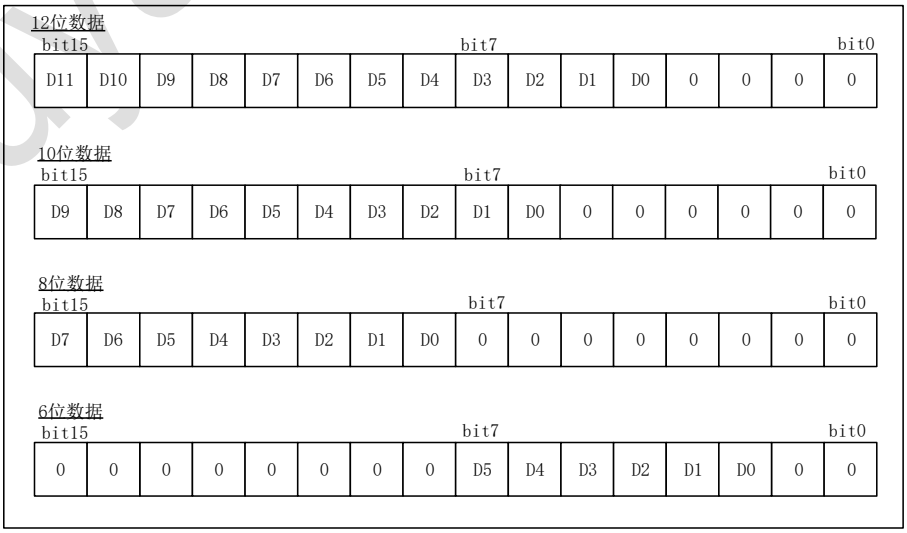


图 13-16 左对齐（偏移未使能，无符号数）

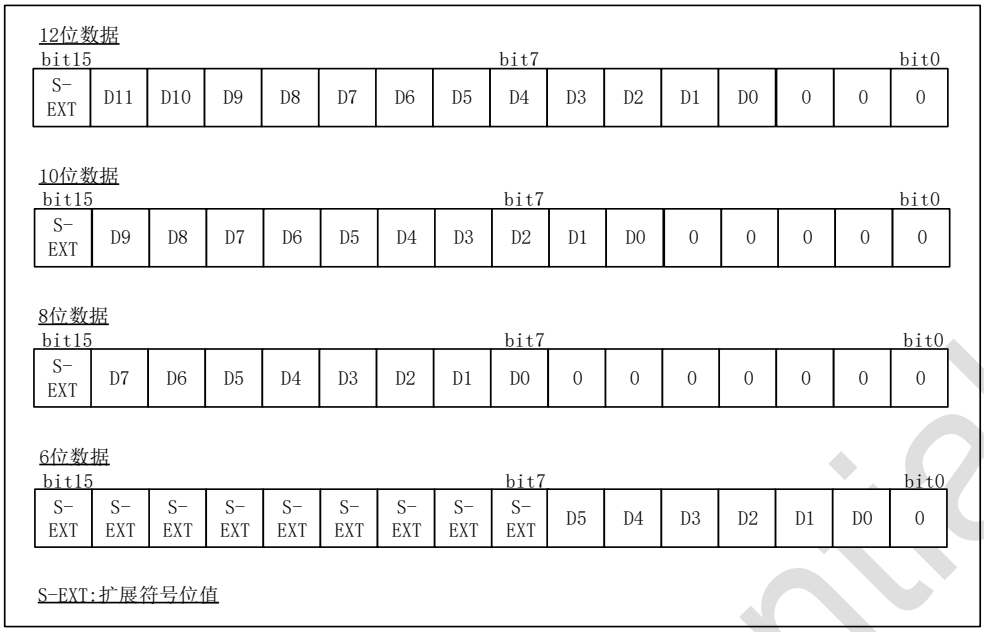


图 13-17 左对齐（偏移使能，有符号数）

13.3.22.3. 增益补偿(GCOMP，ADC_GCOMP)

当设置 ADC_CFGR2 寄存器 GCOMP 位时，对所有转换后的数据进行增益补偿。每次转换后，数据将使用以下公式进行计算。

DATA = DATA(adc result) x(GCOMPCOEFF)/4096

由于 GCOMPCOEFF 范围为 0-16383，因此实际增益补偿因子范围 0-3.999756。

在将结果数据存储在 ADC_DATA 或 ADC_JDRy 寄存器之前，将计算 LSB−1 值四舍五入，并将误差降至最低。

增益补偿对于过采样也是有效的。当增益补偿用于过采样模式时，在累加和右移操作之后执行增益计算，以最小化功耗（增益计算仅进行一次，而不是每次转换）。

13.3.22.4. 偏移补偿(SATEN，OFFSETPOS)

当在偏移操作期间在 ADC_OFRy 寄存器中设置 SATEN 位时，数据是无符号的。所有偏移数据在 0x000 处饱和（在 12 位分辨率）。当 OFFSETPOS 位被设置时，偏移方向是正的，并且数据在 0xFFF 饱和（在 12 位分辨率）。在 8 位分辨率中，数据分别在 0x00 和 0xFF 处饱和。

在偏移和增益补偿之后，对无符号数执行模拟看门狗比较。为了正确的看门狗操作，偏移补偿后的数据必须为无符号格式（ADC_OFRy 寄存器中的 SATEN 位设置为 1）。

13.3.22.5. ADC 过载(OVR，OVRMOD)

ADC 过载标志(OVR) 是指一个缓冲区过冲事件，当转换好的数据未被 CPU 或 DMA 及时读取时，另一个转换数据已经有效时，就发生了 ADC 过载。

若 EOC 还为‘1’的情况下，这时一个新的转换已经完成，那么 CPU 就会在 ADC_ISR 寄存器中的 OVR 标志被置位，表明 ADC 过载。当 ADC_IER 寄存器中的 OVRIE 置位时，产生一个 ADC 过载中断。

当过载事件发生时，ADC 会继续操作并且继续转换除非软件决定停止并复位这个序列转换，可用软件设置 ADC_CR 寄存器中的 ADSTP 为 1 来停止 ADC 转换 OVR 标志可用软件写 1 清除。

当发生过载事件时，可通过对 ADC_CFGR 寄存器中的 OVRMOD 位来设置 ADC 数据寄存器中的数据是被保持还是被覆盖：

- OVRMOD=0

- 过载事件发生时，数据寄存器的值不会被覆盖：之前的数据被保持，新的转换数据丢弃。若 OVR 保持为 1，则后续的转换会被执行但结果都被丢弃。

- OVRMOD=1

- 过载事件发生时，数据寄存器用最后的转换结果覆盖，先前未读的数据丢失，若 OVR 保持为 1，则后续的转换被执行且 ADC_DR 寄存器被新的转换结果覆盖，始终存放着最后转换的结果值。

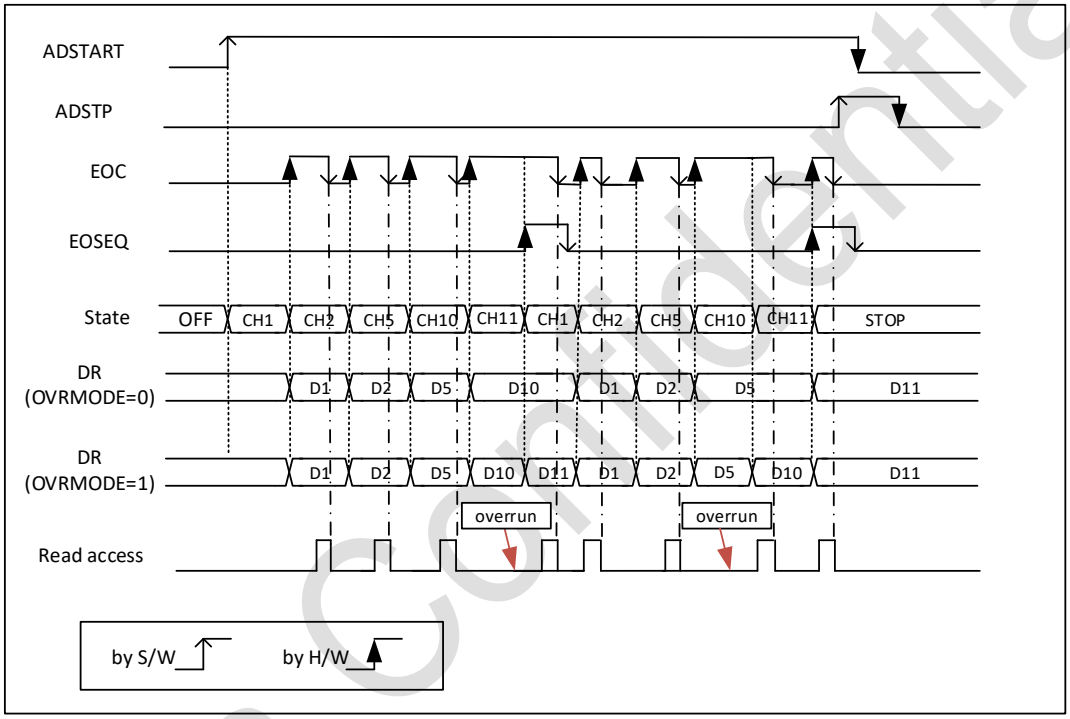


图 13-18 过载

13.3.22.6. 非 DMA 模式

若 ADC 的转换足够慢，转换序列可由软件来控制。这种情况下，软件应用 EOC 标志及其关联的中断去处理每个转换数据。当每次转换结束时，在 ADC_ISR 寄存器中的 EOC 位置位，此时可读 ADC_DR 寄存器的转换值。ADC_CFGR 寄存器中的 OVRMOD 位可配为 0 来管理过载事件。

13.3.22.7. 非过载非 DMA 模式

存在着转换一个或多个通道且不用每次转换结果都要读取的应用。这种情况下，OVRMOD 位必须置为 1 且软件应忽略 OVR 标志。当 OVRMOD=1 时，过载事件不能阻止 ADC 继续转换且 ADC_DR 寄存器中的数据一直为最后转换的数据。

13.3.22.8. DMA 模式

因为所有通道的规则转换结果数据存放到一个单一的数据寄存器中，故当转换通道超过 1 个时用 DMA 方式会更有效。这样可以避免丢失存在 ADC_DR 寄存器中的转换结果。当 DMA 模式开启时(ADC_CFGR 寄存器中的 DMAEN =1)，每次转换结束时都会产生一个 DMA 请求。这样就允许把在 ADC_DR 寄存器中的转换数据传送到软件指定的目标地址中。

尽管如此，因 DMA 不能够及时为 DMA 请求服务而产生的过载(OVR=1)时，ADC 就会停止产生 DMA 请求且相应结果是新的转换数据也不会再由 DMA 进行传输(当 OVR=0 时，会继续传输)。 这也可以认为所有传输到 RAM 中的数据都是有效的(因无效的数据再也不传输了)。

根据 OVRMOD 位的配置，ADC_DR 寄存器中的数据可选择为：保持或覆盖。

DMA 传输请求会被阻止直到软件清除 OVR 位。

有两种不同的 DMA 模式，其取决于 ADC_CFGR 寄存器中的 DMACFG 位的配置：

- DMA 一次模式 (DMACFG=0)

当 DMA 编程用于传输固定长度的数据时，可选用该模式。

- DMA 循环模式 (DMACFG=1)

当 DMA 编程为循环模式时，可选用该模式。

DMA 单次模式(DMACFG=0)

在这种模式下，ADC 在每次转换的数据有效时产生一次 DMA 请求。一旦 DMA 已达到最后一个 DMA 传输时，即使 ADC 转换已再次启动，ADC 停止产生 DMA 请求。(产生 DMA_EOT 中断时，下一次的 ADC 转换有可能已开始)

当 DMA 传输完成 (配置在 DMA 控制器中的所有传输已经完成):

- ADC 数据寄存器的内容冻结
- 任何进行中的转换终止。且结果值丢弃
- 不给 DMA 控制器发出新的 DMA 请求。假如仍有 ADC 转换启动，这种方式可避免产生一个 ADC 过载错误
- ADC 扫描序列停止并复位
- DMA 停止

DMA 连续模式(DMACFG=1)

在这种模式下，即使 DMA 达到最后一个 DMA 的传输，ADC 也会在每次转换的数据有效时产生一次 DMA 请求。这允许 DMA 配置为循环模式来处理连续模拟输入数据流。

13.3.23. ADC 模拟看门狗 (AWDEN, JAWDEN，AWDSGL, AWDCH, AWD_TR)

模拟看门狗的功能由在 ADC_CFGR 寄存器中置位 AWDEN 来开启。它可用于监控所选的单一通道或所有使能通道所配置电压范围(窗口)。

13.3.23.1. AWD 标志和中断

如果模拟电压转换由 ADC 低于低阈值 LT 或高于高阈值 HT 时，AWD 模拟看门狗的状态位被置位。阈值被编程到最多具有 12 位有效数据的 ADC_TR.HT 和 ADC_TR.LT 寄存器中。模拟看门狗中断

可用设置 ADC_IER 寄存器中的 AWDIE 位来使能。AWD 标志位可用软件写 1 来清除。当转换的数据分辨率小于 12 位 (由 DRES[1:0] 位来决定)，被编程阈值的低位必须保持清零，因为内部转换数据的比较都是按左对齐全 12 位的方式进行比较。

下表描述了针对模拟看门狗的所有可能的分辨率执行比较。

表 13-5 模拟看门狗比较

分辨率位 (RES)	模拟看门狗比较对象		注释
	原始转换数据，左对齐	阈值	
00: 12-bit	DATA[11:0]	LT[11:0] 和 HT[11:0]	
01: 10-bit	DATA[11:2],00	LT[11:0] 和 HT[11:0]	用户必须配置 LT[1:0]和 HT[1:0] 为 00
10: 8-bit	DATA[11:4],0000	LT[11:0] 和 HT[11:0]	用户必须配置 LT[3:0]和 HT[3:0] 为 0000
11: 6-bit	DATA[11:6],000000	LT[11:0] 和 HT[11:0]	用户必须配置 LT[5:0]和 HT[5:0] 为 000000

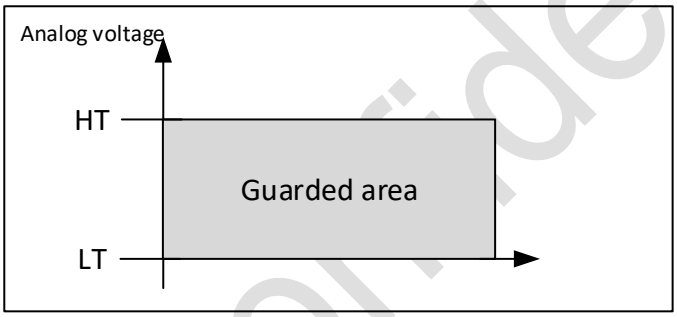


图 13-19 模拟看门狗保护区域

表 13-6 模拟看门狗通道选择

看门狗监测通道	AWDSGL 位	AWDEN 位	JAWDEN 位
无	x	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有注入和规则通道	0	1	1
单个注入通道	1	0	1
单个规则通道	1	1	0
单个注入或规则通道	1	1	1

13.3.23.2. ADC_AWD_OUT 输出信号产生

模拟看门狗与一个内部硬件信号 ADC_AWD_OUT 关联。该信号直接连接到片上某些定时器的 ETR（外部触发）输入端口。

当关联的模拟看门狗被启用时，ADC_AWD_OUT 会激活：

- 当受保护的转换超出编程设定的阈值时，ADC_AWD_OUT 会被置位。
- 当接下来的受保护转换结束且值在编程设定的阈值范围内时，ADC_AWD_OUT 会复位 (但若后续受保护转换仍超出设定阈值，则该信号保持为 1）。

- 当禁用 ADC（设置 ADDIS=1）时，ADC_AWD_OUT 也会被复位。请注意，停止规则或注入转换（设置 ADSTP=1 或 JADSTP=1）不会影响 ADC_AWD_OUT 的生成。

注意：AWD 标志由硬件设置并由软件复位；AWD 标志不影响 ADC_AWD_OUT 的生成（例如，如果软件没有清除标志，即使 AWD 标志保持为 1，ADC_AWD_OUT 也可能在 AWD 标志为 1 的情况下切换）。

ADC_AWD_OUT 信号由 PCLK 域生成。

每次 ADC 转换结束时都会执行 AWD 比较。规则、注入 ADC_AWD_OUT 同理。

下图为选择所有通道的时序：

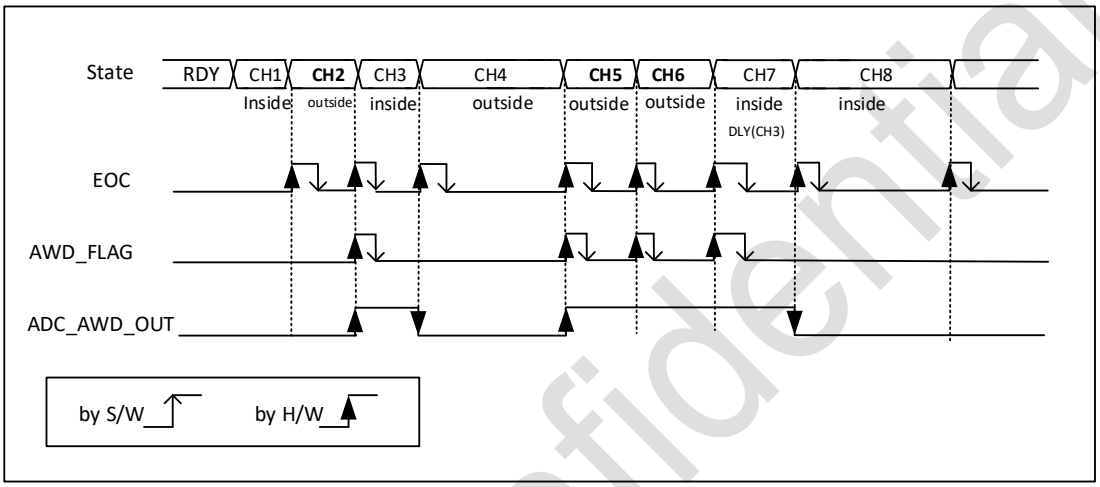


图 13-20 选择所有通道模拟看门狗

下图为软件不清除 AWD_FLAG 的情况：

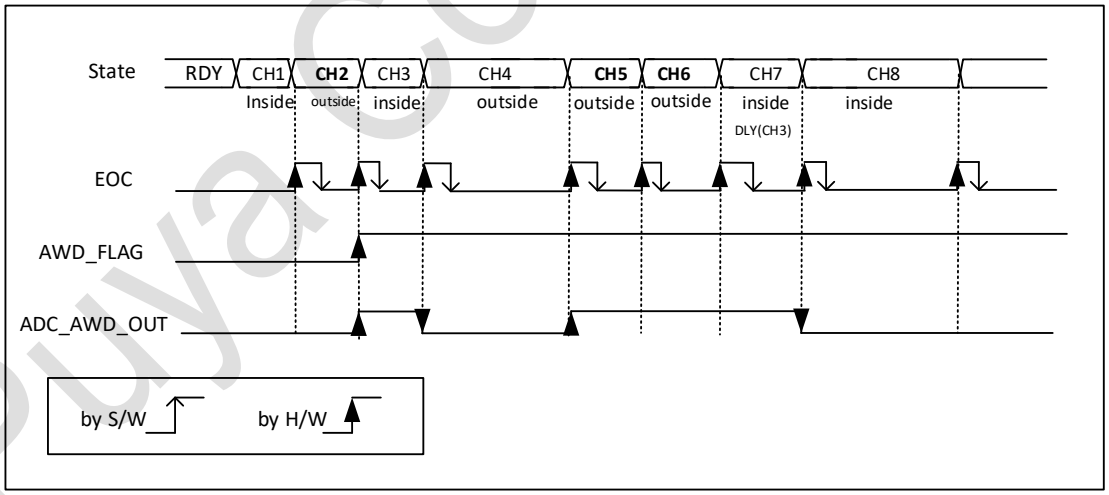


图 13-21 软件不清除 AWDx 的情况

13.3.23.3. 带增益和偏移补偿的模拟看门狗

当增益和偏移补偿使能时，模拟看门狗在补偿数据后比较阈值。

注意：当偏移补偿使能时(ADC_OFRy 寄存器中的 OFFSETy_EN 设置为 1)，数据溢出或下溢可能导致看门狗的结果错误。当饱和使能时(在 ADC_OFRy 中 SATEN 设置为 1)，看门狗提供正确的结果。然而，这将无法使用带符号数据格式。

13.3.24. ADC 动态低功耗特性

13.3.24.1. 自动延迟转换

自动延迟转换对于简化软件以及优化应用程序的性能非常有用，因为程序在低频运行时可能会遇到 ADC 溢出的风险。

当在 ADC_CFGR 寄存器中设置 AUTDLY 为 1 时，仅当所有之前的数据已处理完毕时，才能开始新的转换：

- 对于规则转换：一旦 ADC_DR 寄存器被读取或 EOC 位被清除；
- 对于注入转换：当 JEOS 位被清除时；

注意：注入转换结束后，如果 JEOS 不被清除，如果发生规则触发，则忽略该规则触发。

通过这种方式，可自动调整 ADC 的速度，使其适应系统读取数据的速度。

延迟在每个规则转换(无论 DISCEN=0 或 1)和每个注入转换序列(无论 JDISCEN=0 或 1)之后插入。

注意：不会在注入转换序列之间插入延时，但会在最后一个序列之后插入延时。

在转换期间，将忽略在此延迟期间发生的硬件触发事件(针对同一组转换)。

注意：对于软件触发器来说，在此延迟期间，仍然可以通过置位 ADSTART 或 JADSTART 来重新启动转换（满足前述原则，即规则转换 EOC 清除或者数据读走，注入转换 JEOS 清除；若不满足条件，则硬件保证不能置位 ADSTART 和 JADSTART）：在启动新的转换之前，由软件读取数据以避免数据覆盖丢失。

在不同组的转换之间不插入延迟(规则转换后即进行注入转换，反之亦然):

- 如果在规则转换的自动延迟期间发生注入触发，则注入转换立即开始。
- 一旦注入序列完成，ADC 会等待上一规则转换的延时（如果未结束），然后才会启动新的规则转换。

在自动注入模式(JAUTO=1)中，行为略有不同，只有在之前注入的转换序列的自动延迟结束时(当 JEOS 被清除时)，新的规则转换才能开始。这是为了确保软件可以在开始一个新序列之前读取给定序列的所有数据。

要在连续自动注入模式和自动延迟模式(JAUTO=1, CONT=1, AUTDLY =1)下停止转换，请执行以下步骤:

1. 等待，直到 JEOS=1(不再重新启动转换)
2. 清零 JEOS
3. 设置 ADSTP = 1
4. 读取规则数据。

在 AUTDLY 模式中，如果硬件规则触发事件发生在已经在进行的规则序列期间，或者发生在序列最后一次规则转换之后的延迟期间，则忽略该事件。但是，如果它发生在此延迟之后，则认为它是挂起的，即使它发生在紧随其后的延迟的注入序列中。然后转换在注入序列的延迟结束后开始。

在 AUTDLY 模式中，如果硬件注入的触发事件发生在已经在进行的注入序列期间，或者发生在序列的最后一次注入转换之后的延迟期间，则忽略该事件。

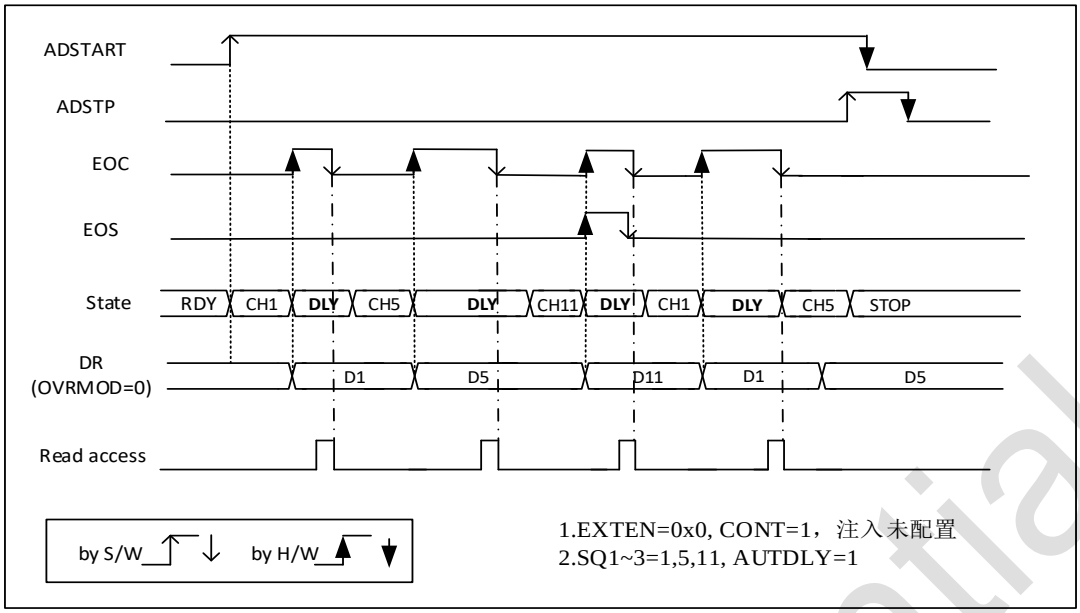


图 13-22 AUTODLY 模式，连续模式下的规则转换，软件触发

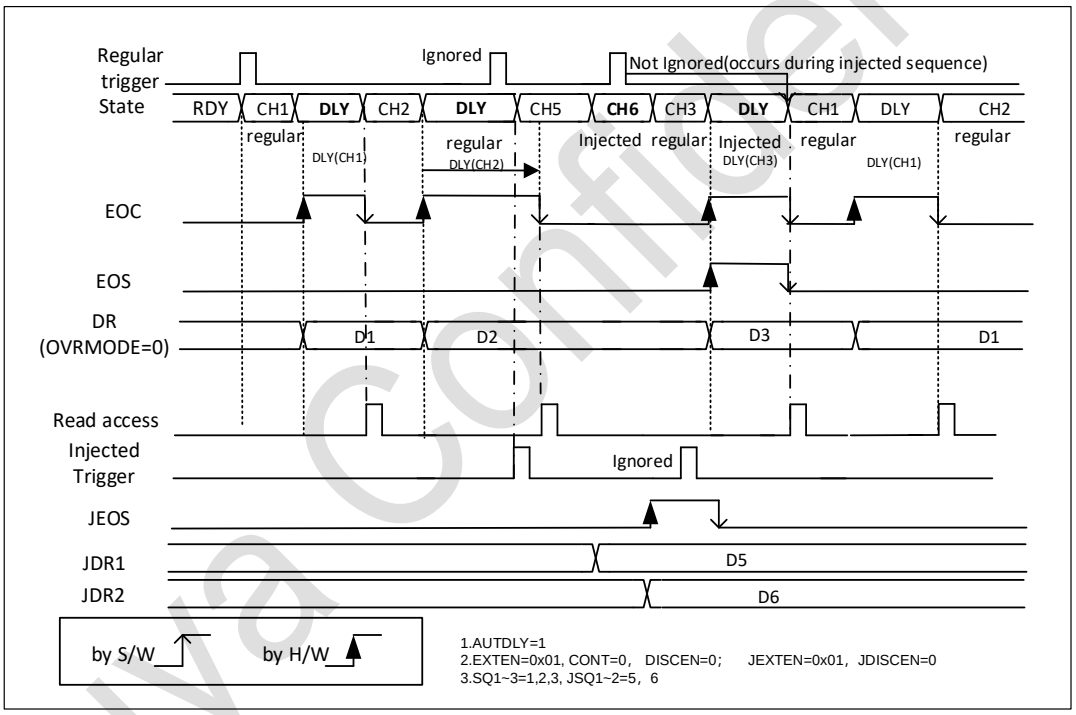


图 13-23 AUTODLY 模式，规则转换被注入中断，硬件触发

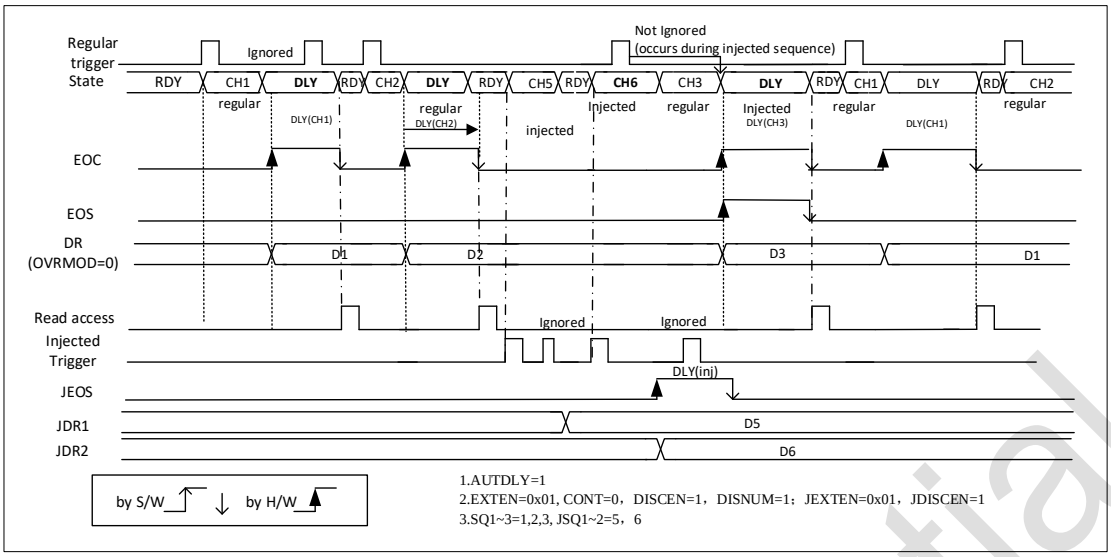


图 13-24 AUTODLY 模式，规则转换被注入中断，硬件触发，非连续转换

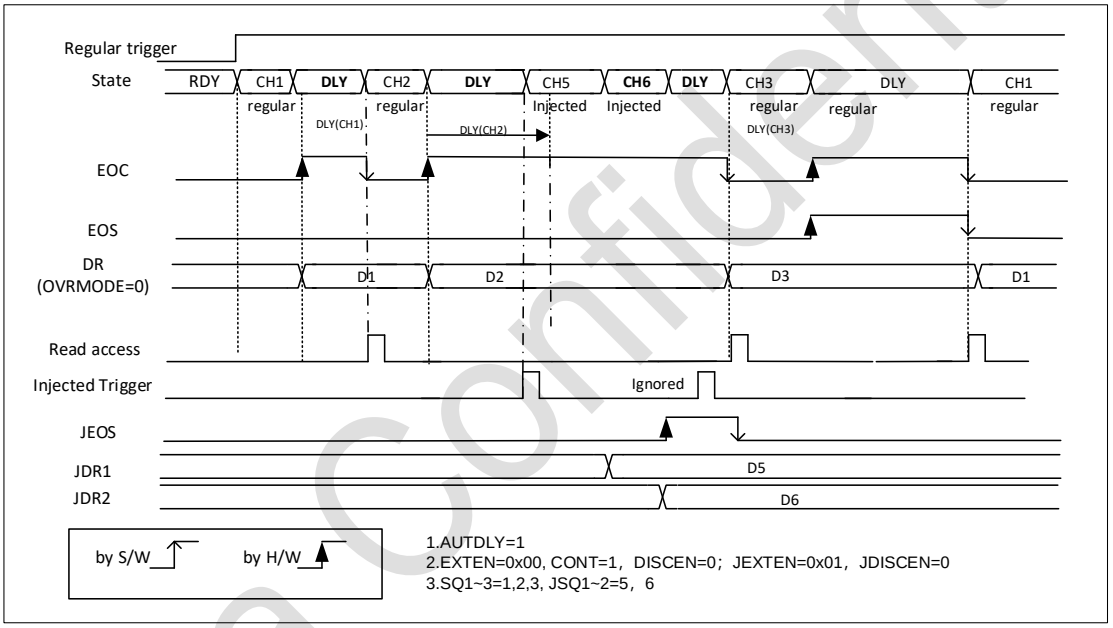


图 13-25 AUTODLY 模式，规则转换被注入中断，规则软件触发，注入硬件触发，连续转换

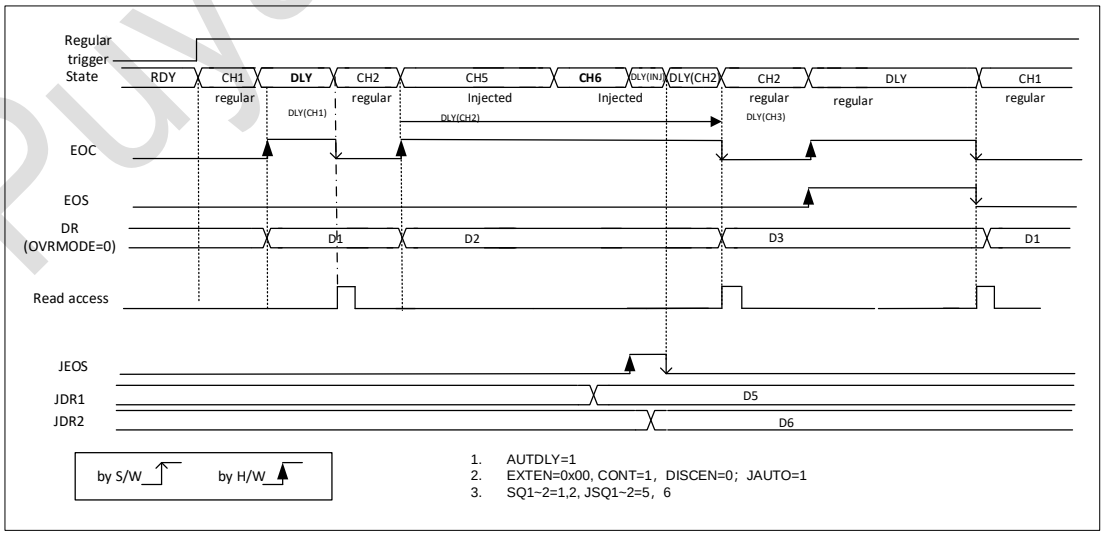


图 13-26 AUTODLY 模式，自动注入模式

13.3.25. ADC 过采样

过采样单元执行数据预处理以降低 CPU 负载。它能够处理多个转换，并将它们平均为单个数据(增大数据宽度，最高可达 16 位)。

过采样器提供了以下公式，其中 N 和 M 可以调整：

$$Result = \frac{1}{M} \times \sum_{n=0}^{n=N-1} Conversion(t_n)$$

过采样器允许通过硬件执行以下功能：平均、降低数据速率、提高信噪比、基本滤波。

过采样率 N 是通过 ADC_CFGR2 寄存器中的 OVFS[2:0]位定义的，可在 2x 到 256x 的范围内调节。除法系数 M 由高达 8 位的右位移位组成，并且使用 ADC_CFGR2 寄存器中的 OVSS[3:0]位来定义。

求和单元可以产生高达 20 位的结果 (256x 12 位结果) , 该结果首先右移。然后，在最终传输到 ADC_DR 数据寄存器之前被截断为 16 个最低有效位，将移位留下的最低有效位四舍五入到最接近的值。

注意：如果移位后的中间结果超过 16 位，则该结果被原样截断而不饱和。

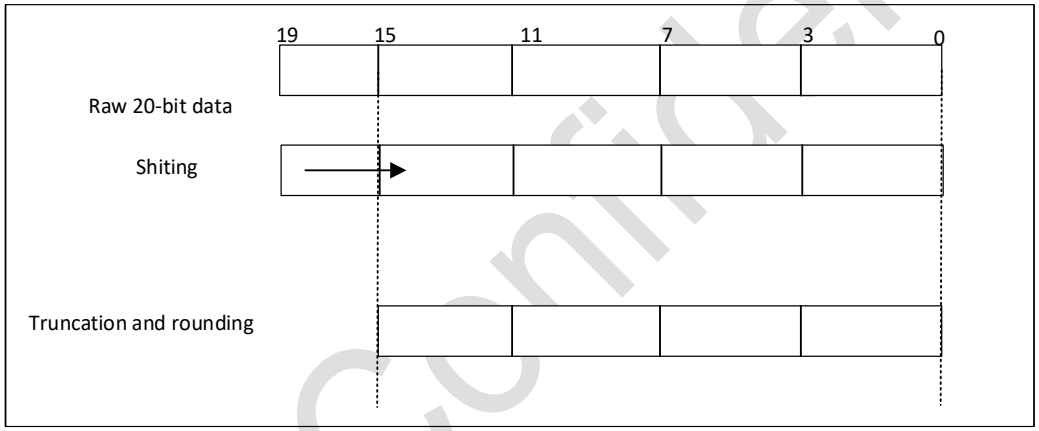


图 13-27 20bit 截断为 16bit

下图给出了处理的数字示例，从原始的 20 位累加数据到最终的 16 位结果。

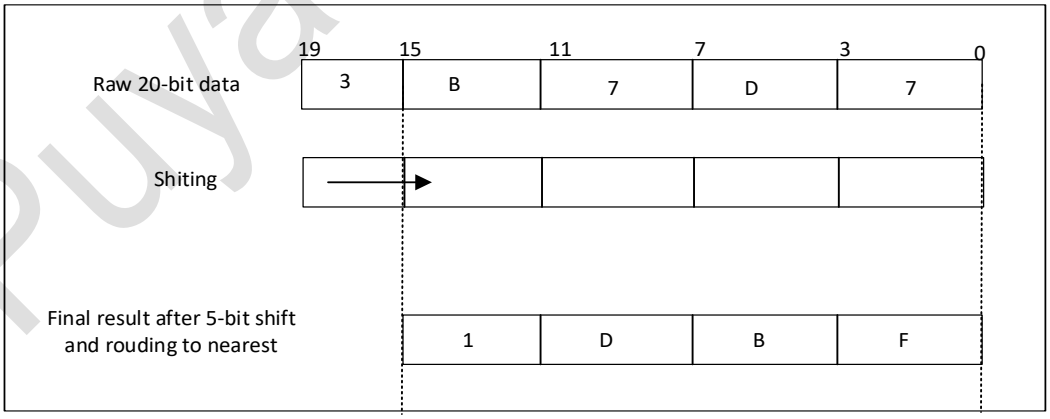


图 13-28 具有 5 位移位和舍入的数值示例

下表给出了原始转换数据为 0xFFF 的各种 N 和 M 组合的数据格式。

表 13-7 最大输出结果与 N 和 M 的关系（灰色单元格表示截断）

过采样	最大原始数据	无移位 OVSS=0000	1bit 移位 OVSS=0001	2bit 移位 OVSS=0010	3bit 移位 OVSS=0011	4bit 移位 OVSS=0100	5bit 移位 OVSS=0101	6bit 移位 OVSS=0110	7bit 移位 OVSS=0111	8bit 移位 OVSS=1000

率										
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020
4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFF0	0x FFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

过采样模式下的转换时序没有变化：在整个过采样序列中，采样时间保持相等。每 N 个转换提供一个新数据，通过 N x T_{CONV} = N x（t_{SMPL}+t_{SAR}）等效延迟。标志设置如下：

- 在每个采样阶段之后设置采样阶段结束标志（EOSMP）
- 当过采样结果可用时，每 N 次转换发生一次转换结束（EOC）
- 序列结束（EOS）发生在过采样数据序列完成之后（即在 N x 总序列长度转换之后）

13.3.25.1. 过采样时支持的 ADC 操作模式（单个 ADC 模式）

在过采样模式下，大多数 ADC 操作模式都保持不变：

- 单次模式/连续模式转换
- 通过软件或外部触发启动 ADC 转换
- 转换期间停止 ADC(中止)
- 通过 CPU/DMA 读数据
- 低功耗模式(AUTDLY)
- 可编程分辨率：在这种情况下，减少的转换值（根据 ADC_CFGR 寄存器中的 RES [1:0]位）以与 12 位转换相同的方式进行累加、截断、四舍五入和移位。

注意：使用过采样数据时，数据对齐不可用。ADC_CFGR 中的 ALIGN 位被忽略，并且数据总是以右对齐的方式提供。过采样模式下不支持偏移校正。设置 ROVSE 和/或 JOVSE 位时，ADC_OFRy 寄存器中 OFFSETy_EN 位的值被忽略（视为复位）。

13.3.25.2. 模拟看门狗

保持模拟看门狗功能（AWDSGL，AWDEN 和 JAWDEN 位），但有以下区别：

忽略 RES[1:0]位，始终使用完整的 12 位值 HT[11:0]和 LT[11:0]进行比较

对 16 位过采样结果 ADC_DR[15:4]中的最高有效 12 位执行比较

注意：使用高移位值时必须小心，这会减小比较范围。例如，如果过采样结果被移位 4 位，从而产生 12 位的数据右对齐，则有效的模拟看门狗比较只能在 8 位上执行。在 ADC_DR[11:4]和 HT[0:7]/LT[0:7]之间进行比较，并且 HT[11:8]/LT[11:8]必须复位。

13.3.25.3. 触发模式

平均器也可以用于基本的过滤。虽然不是一个非常强大的滤波器（缓慢的滚降和有限的阻带衰减），但它可以用作陷波滤波器来抑制恒定的寄生频率（通常来自电源或开关模式电源）。为此，可以使用

ADC_CFGR2 中的 TROVS 位启用特定的非连续模式，以便能够具有由用户定义的过采样频率，并且独立于转换时间本身。

下图显示了在非连续模式下如何响应触发启动转换。如果设置了 TROVS 位，则忽略 DISCEN 位的内容并将其视为 1。

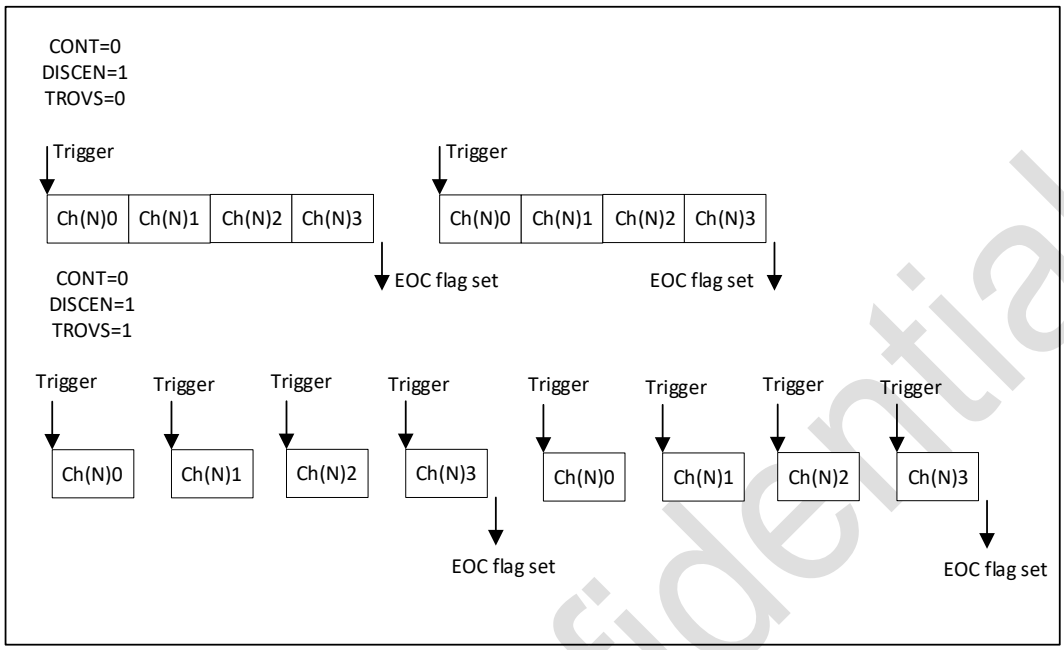


图 13-29 已触发的规则过采样模式(TROVS=1)

13.3.25.4. 过采样时注入和规则序列管理

在过采样模式下，注入序列和规则序列行为可能有所区别。如果两个序列必须同时使用，可以对它们启用过采样，但有一些限制（这与唯一的累加单元有关）。

13.3.25.5. 仅对规则通道进行过采样

规则过采样模式位 ROVSM 定义了如果规则过采样序列被注入的转换中断，则如何恢复该序列：

- 在连续模式中，会从上一有效数据开始重新累加（在由于注入触发而发出的转换中止请求之前）。这确保了无论注入频率如何都将完成过采样（假设触发之间至少可完成一次常规转换）；
- 在恢复模式下，从 0 重新开始累加（忽略之前的转换结果）。该模式可确保所有用于过采样的数据在单个时隙内进行了背靠背转换。需要注意的是，注入触发周期必须超过过采样时长。如果该条件未得到满足，将无法完成过采样，规则序列将被禁用。

下图为 4x 过采样率的例子。

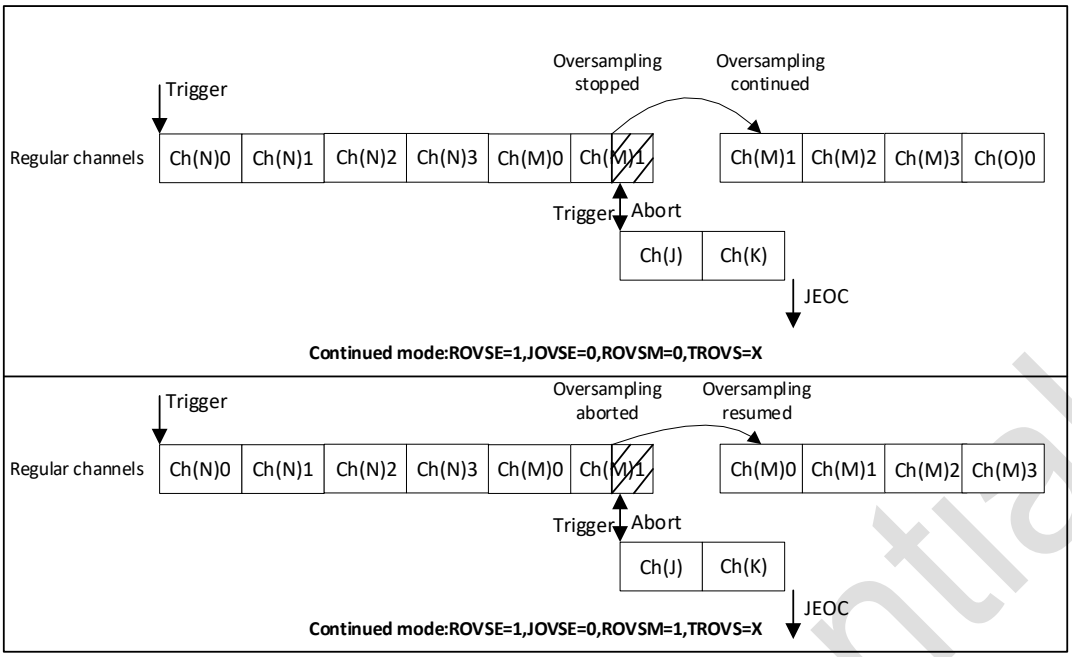


图 13-30 规则过采样模式（4 倍过采样率）

13.3.25.6. 仅对注入通道进行过采样

注入过采样模式位 JOVSE 专为注入序列中的转换使能过采样。

13.3.25.7. 规则通道和注入通道过采样

可以同时设置 ROVSE 和 JOVSE 位。在这种情况下，规则过采样模式被强制为恢复模式（忽略 ROVSM 位），如下图所示

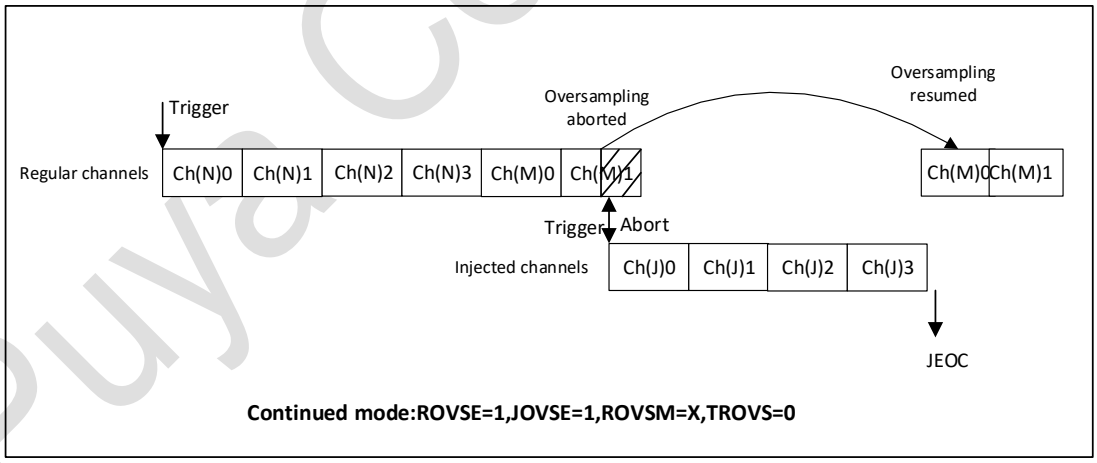


图 13-31 同时使用规则和注入转换的过采样模式

13.3.25.8. 规则通道过采样被注入触发中断

注入转换可以在触发规则模式下执行。在这种情况下，必须禁用注入模式过采样模式，并忽略 ROVSM 位（强制恢复模式）。JOVSE 位必须复位（软件需要配置为 0。如果配置为 1 则硬件在这种情况下会将 JOVSE 清零）。该行为如下图所示。

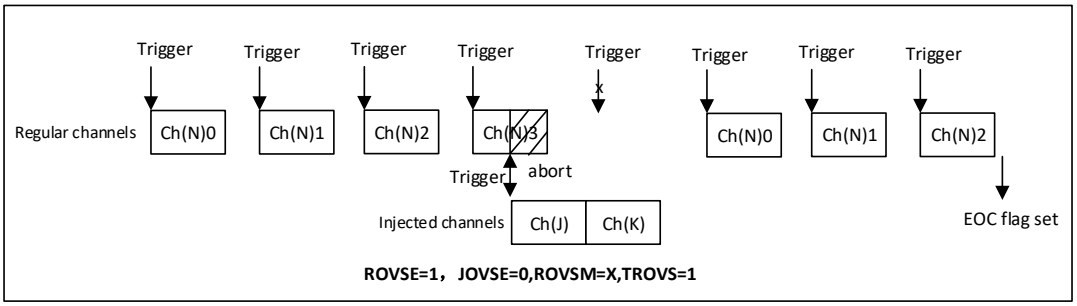


图 13-32 已触发规则过采样被注入中断

13.3.25.9. 自动注入模式

可以对自动注入的序列进行过采样，并将所有转换结果存储在寄存器中，以节省 DMA 资源。此模式仅在规则和注入过采样均使能时可用：JAUTO=1、ROVSE=1 和 JOVSE=1，不支持其他组合。ROVSM 位在自动注入模式下被忽略。下图显示了转换的顺序。

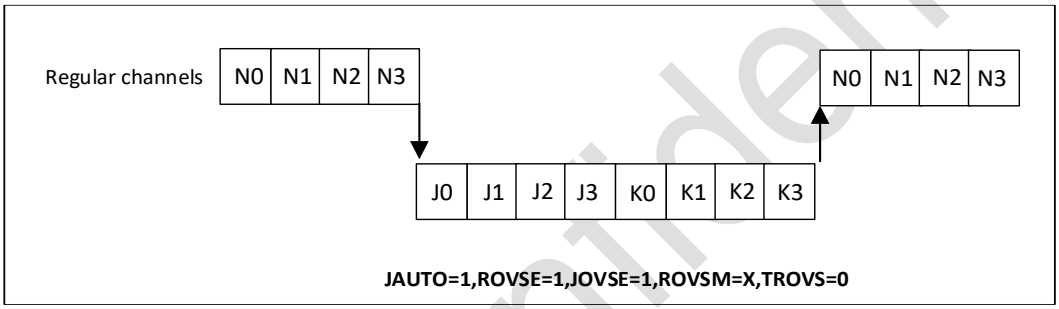


图 13-33 自动注入模式的过采样

也可以使用 TROVS 位启用触发模式。在这种情况下，ADC 必须配置如下：JAUTO=1，DISCEN=0，JDISCEN=0，ROVSE=1，JOVSE=1 和 TROVSE=1。（该配置禁止与 CONT 组合(使能 CONT 会导致功能异常)）。

13.3.25.10. 模式组合总结

表 13-8 过采样器模式小节

规则过采样 ROVSE	注入过采样 JOVSE	过采样模式 ROVSM; 0=连续 1=恢复	触发规则模式 TROVS	描述
1	0	0	0	规则连续模式
1	0	0	1	不支持
1	0	1	0	规则恢复模式
1	0	1	1	触发规则恢复模式
1	1	0	X	不支持
1	1	1	0	注入和规则恢复模式
1	1	1	1	不支持
0	1	X	X	注入过采样

13.3.26. 温度传感器

温度传感器可以用来测量器件的结点温度（T_J）。

温度传感器内部连接到 ADC 输入通道，该通道用于将传感器输出电压转换为数字值。温度传感器的采样时间必须大于数据手册给出的稳定时间的最小值。当温度传感器没被使用时，传感器可以置于断电模式。

温度传感器输出电压跟温度成线性变化关系，由于生产过程的变化，温度变化曲线的偏移在不同芯片上会有不同。为了提高这个准确度，每一颗的校准值会被产品测试单独给出并且保存在系统存储区域。

内部电压参考（V_{REFINT}）提供一个稳定电压输出给 ADC 和比较器。

注：必须设置 ADC_CCR.TSEN=1 位来激活内部温度传感器通道，设置 ADC_CCR.VREFINT_EN=1 位来激活内部参考电压。

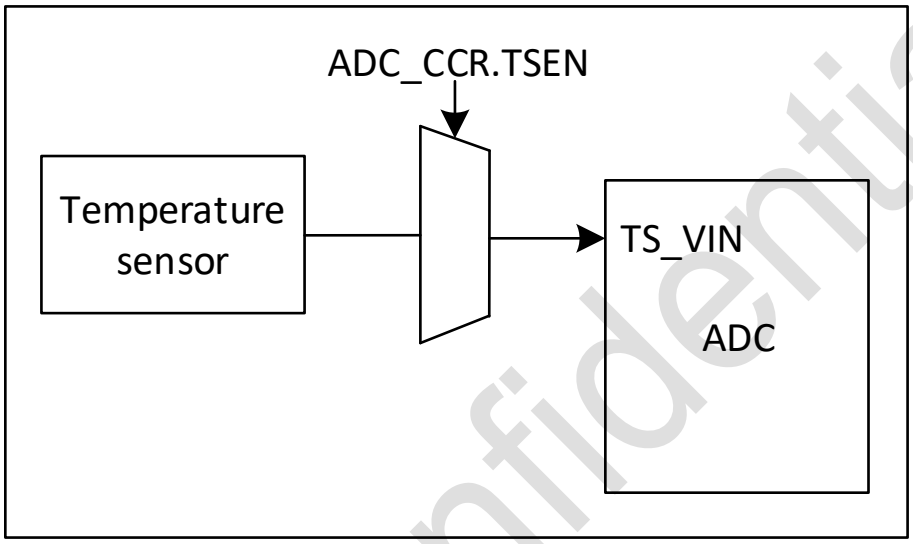


图 13-34 温度传感器框图

13.3.26.1. 读温度

要使用温度传感器，请执行如下操作：

- 选择 ADC_VINP[27]输入通道
- 选择一个采样时间，该采样时间要大于数据手册中所指定的最低采样时间
- 在 ADC_CCR 寄存器中置位 TSEN，用来唤醒从断电模式下的温度传感器
- 置位 ADEN 使能 ADC，然后置位 ADSTART(也可用外部触发)来启动 ADC 转换
- 从 ADC_DR 寄存器中读取 V_{TS} 转换数据

用下列公式计数温度：

$$Temperature\left(in\ \ ℃\right)=\frac{105℃-30℃}{TS_{CAL2}-TS_{CAL1}}\times(TS_{DATA}-TS_{CAL1})+30℃$$

TS_{CAL2} 代表 105℃温度传感器的校准值（从 Flash 地址 0x1FFF 1E24 读取）

TS_{CAL1} 代表 30℃温度传感器的校准值（从 Flash 地址 0x1FFF 1E20 读取）

TS_{DATA} 是 ADC 转换的实际输出值

注： 传感器从断电模式下唤醒时到能正确输出 V_{TS} 要有一个启动时间，ADC 从上电后启动也有一个启动时间，若要减少这个延时，则需要在同一时间设置 ADEN 和 TSEN 位。

13.3.27. 监测内部参考电压

可以通过监测内部参考电压来获得用于评估 ADC VREF+ 电压的参考值。

内部参考电压在内部连接到输入通道 ADC_VINP[28]。

该通道的采样时间必须大于产品数据手册中指定的稳定时间。

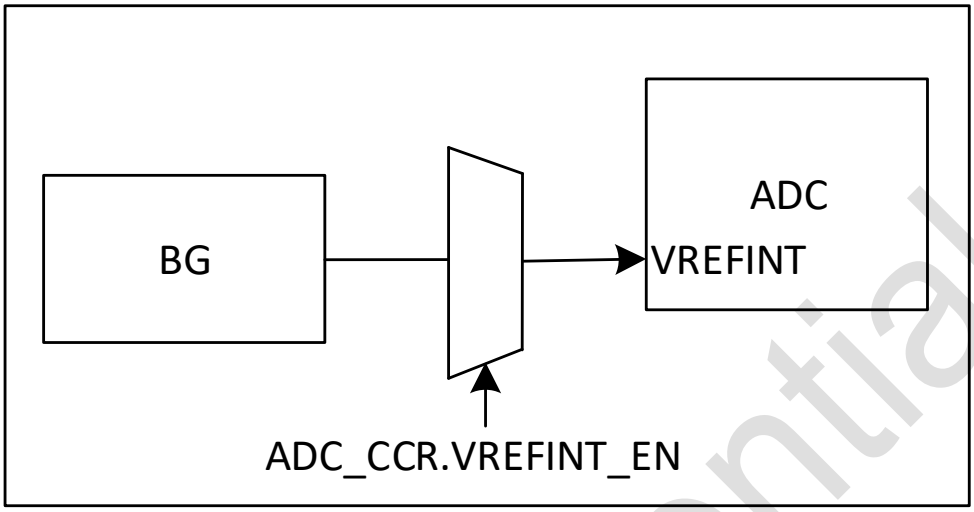


图 13-35 温度传感器框图

注：必须设置 ADC_CCR.VREFINT_EN=1 位来激活内部参考电压。

下面公式可以给出真实 VCC 的电压值：

$$VREFINT = 1.2V = \frac{ADC\_DATAx}{4095} \times VCC$$

V_{REFINT} 固定值为 1.2V；

ADC_DATA 是 V_{REFINT} 通道的转换数据。

把一个 ADC 测量到的通道电压相对值转化为绝对电压值

ADC 是根据模拟电源输入和转换通道上的电压比例给出一个数字值。大部分应用是需要把这个比例转换成一个电压值。对于 V_{CC} 可知的情况下并且 ADC 转换值是右对齐的，可用下面公式得到这个绝对电压值：

$$VCHANNEL = \frac{ADC\_DATAx}{4095} \times VCC$$

$V_{CHANNEL}$ 是通道电压；

ADC_DATA 是 ADC_DR 里面的转换数据；

4095 表示为 12 位。

13.3.28. 监测电源

由于 V_{CC} 电压可能不准，因此 V_{CC} 引脚需要内部连接到桥接分配器，以确保 ADC 正确运行。桥接器自动使能将 V_{CC}/3 连接到 ADC_VINP[29] 输入通道。

13.4. ADC 中断

ADC 中断可由以下任一事件产生：

- ADC 就绪（ADRDY 标志）

- 任何一次的规则转换结束 (EOC 标志)
- 规则序列转换结束 (EOS 标志)
- 任何一次的注入转换结束 (JEOC 标志)
- 注入序列转换结束 (JEOS 标志)
- 模拟看门狗检测发生 (AWD 标志)
- 采样阶段结束 (EOSMP 标志)
- 发生数据过载(OVR 标志)
- 校准完成（EOCAL 标志）

独自的中断使能位用于灵活设置 ADC 中断。

表 13-9 ADC 中断

中断事件	事件标志	使能控制
ADC 就绪	ADRDY	ADRDYIE
规则转换结束	EOC	EOCIE
规则序列转换结束	EOS	EOSIE
注入转换结束	JEOC	JEOCIE
注入序列转换结束	JEOS	JEOSIE
模拟看门狗状态置位	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
过载	OVR	OVRIE
校准完成	EOCAL	EOCALIE

13.5. ADC 寄存器

13.5.1. ADC 中断状态寄存器 (ADC_ISR)

偏移地址: 0x00

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	EOCAL	Res	Res	Res	AWD	JEOS	JEOC	OVR	EOS	EOC	EOSMP	ADRDY
				RC_W1				RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	EOCAL	RC_W1	0	校准完成标志 当校准完成时硬件置位，软件写 1 清零。 0：校准未启动或者校准未完成（或标志事件已通过软件确认并清零） 1：校准完成
10:8	Reserved	-	-	保留
7	AWD	RC_W1	0	模拟看门狗标志

				当转换电压值低于 ADC_TR.LT 寄存器或者高于 ADC_TR.HT 寄存器编程的值时硬件置位。软件写 1 清零。 0：无模拟看门狗事件发生（或标志事件已通过软件确认并清零） 1：模拟看门狗事件发生
6	JEOS	RC_W1	0	注入通道序列结束标志 当注入序列的所有通道转换结束时，该位由硬件置 1。软件写 1 清 0。 0：注入转换序列未完成（或标志事件已通过软件确认并清零） 1：注入转换序列完成
5	JEOC	RC_W1	0	注入通道转换结束标志 当每个通道每次转换结果后新的数据结果可以从 ADC_JDRy 寄存器读到时，硬件置位该位。软件写 1 清 0 或读取相应的 ADC_JDRy 寄存器清 0 0：注入通道转换未完成（或标志事件已通过软件确认并清零） 1：注入通道转换已完成
4	OVR	RC_W1	0	规则通道过载 当过载发生时，硬件置位该位。当 EOC 标志已置位表明一次新的转换已完成。该位写 1 清 0 0：无过载发生（或标志事件已通过软件确认并清零） 1：过载已发生
3	EOS	RC_W1	0	规则通道序列结束标志 选择的规则序列转换结束时硬件置位该位。软件写 1 清 0 0：规则转换序列没有完成（或标志事件已通过软件确认并清零） 1：规则转换序列完成
2	EOC	RC_W1	0	规则通道转换结束标志 当每个规则通道每次转换结束后，新的数据结果可以从 ADC_DR 寄存器读到时，硬件置位该位。软件写 1 清 0 或读 ADC_DR 寄存器清 0 0：规则通道转换未完成（或标志事件已通过软件确认并清零） 1：规则通道转换已完成
1	EOSMP	RC_W1	0	通道采样结束标志 在每次规则转换的采样阶段结束时，硬件置位该位，软件写 1 清 0 0：采样阶段未结束（或标志事件已通过软件确认并清零） 1：采样阶段结束
0	ADRDY	RC_W1	0	ADC 就绪 ADC 使能后（位 ADEN=1 或 ADCAL=1）以及 ADC 达到准备好接收转换请求的状态时，会通过硬件将该位置 1。 通过软件写入 1 可将该位清零。 0： ADC 未准备好开始转换（或标志事件已通过软件确认并清零） 1： ADC 已准备好开始转换

13.5.2. ADC 中断使能寄存器（ADC_IER)

偏移地址: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	EOCALIE	Res.	Res.	Res.	AWDIE	JEOSIE	JEOCIE	OVRIE	EOSIE	EOCIE	EOSMPIE	ADRDYIE
				RW				RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	EOCALIE	RW	0	校准完成中断使能 0：校准完成中断禁止 1：校准完成中断使能
10:8	Reserved	-	-	保留
7	AWDIE	RW	0	模拟看门狗中断使能位 0：禁止模拟看门狗中断 1：使能模拟看门狗中断 当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
6	JEOSIE	RW	0	注入序列转换结束中断使能 0：禁止 JEOS 中断 1：使能 JEOS 中断 当 JADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
5	JEOCIE	RW	0	注入通道转换结束中断使能 0：禁止 JEOC 中断 1：使能 JEOC 中断 当 JADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
4	OVRIE	RW	0	规则通道过载中断使能位 0：禁止 ADC 过载中断 1：使能 ADC 过载中断 当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
3	EOSIE	RW	0	规则通道序列结束中断使能位 0：禁止序列结束中断 1：使能序列结束中断 当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
2	EOCIE	RW	0	规则通道转换结束中断使能位 0：禁止转换结束中断 1：使能转换结束中断 当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
1	EOSMPIE	RW	0	规则通道采样标志结束中断使能位 0：采样标志结束中断禁止 1：采样标志结束中断使能 当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。
0	ADRDYIE	RW	0	ADC 就绪中断使能位 0：禁止 ADC 就绪中断 1：使能 ADC 就绪中断 当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)软件可以写该位。

13.5.3. ADC 控制寄存器（ADC_CR)

偏移地址: 0x08

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAL	Res.	Res.	Res.	RSTCAL	Res.	Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.
RS				RS											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res.	Res	JADSTP	ADSTP	JADSTART	ADSTART	ADDIS	ADEN
										RS	RS	RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RS	0	ADC 校准启动。 软件设置启动 ADC 校准，校准完成后硬件自动清零（成功或者失败）。 0: 校准完成 1: 写 1 校正 ADC，读为 1 表明校准正在进行 注: 1. 软件写 1 不能写 0，硬件清零。 2. ADEN=0 时才能写 ADCAL=1. 3. 仅当 ADEN=1、 ADSTART=0 且 JADSTART=0（ADC 已使能，当前未进行任何转换）时，才允许通过软件对 ADC_CALFACT 执行写操作来更新校准系数。
30:28	Reserved	-	-	保留
27	RSTCAL	RS	1’b0	校准复位使能位 该位由软件写 1 置位，由硬件写 0 清除。在校准寄存器被初始化后(即 RSTCAL 置 1 后)，该位即被清除。 0：校准寄存器已初始化 1：初始化校准寄存器 注：当正在进行转换时，如果设置 RSTCAL，清除校准寄存器需要额外的周期。
26:6	Reserved	-	-	保留
5	JADSTP	RS	0	ADC 停止注入转换命令。 软件置位停止和丢弃正在进行的注入转换（JADSTP 命令）。 当转换被丢弃并且准备接受新的转换命令时硬件会清除该位。 0: 没有正在进行的 ADC 停止注入转换命令 1: 写 1 停止 ADC 注入转换，读为 1 表明一个 JADSTP 命令正在进行中。 软件写该位为 0 为无效操作。 注：当 JADSTART=1 且 ADDIS=0 时才能写 JADSTP 为 1。
4	ADSTP	RS	0	ADC 停止规则转换命令。 软件置位停止和丢弃正在进行的转换（ADSTP 命令）。 当转换被丢弃并且准备接受新的转换命令时硬件会清除该位。 0: 没有正在进行的 ADC 停止转换命令 1: 写 1 停止 ADC 规则转换，读为 1 表明一个 ADSTP 命令正在进行中。 软件写该位为 0 为无效操作。

				注： 1. 当 ADSTART=1 且 ADDIS=0 时才能写 ADSTP 为 1。 2. 自动注入模式（JAUTO=1），ADSTP 能同时终止规则转换和自动注入转换。
3	JADSTART	RS	1’b0	ADC 启动注入转换 该位由软件设置，用于启动注入通道的 ADC 转换。根据 JEXTEN[1:0]的配置来决定转换是软件立即启动，还是由硬件触发事件来启动。 0：当前未进行 ADC 注入转换 1：写 1 启动 ADC 注入转换，读为 1 表明 ADC 正在操作，可能正在转换 该位由硬件清除的情况： – 在单次转换模式(CONT=0, DISCEN=0)，选择软件启动时(JEXTEN=00)：序列转换完成时（JEOS 标志置位） – 其他情况下：执行 JADSTP 命令之后，同时 JADSTP 标志又被硬件清 0 之时 注： 1. 软件只有当 ADEN=1 且 ADDIS=0 时才能配置 JADSTART=1. 软件写该位为 0 为无效操作。 2. 自动注入模式（JAUTO=1），置位 ADSTART 能启动规则和自动注入转换。
2	ADSTART	RS	0	ADC 启动规则转换 该位由软件设置，用于启动 ADC 规则转换。根据 EXTEN[1:0]的配置来决定转换是软件立即启动，还是由硬件触发事件来启动。 0: 当前未进行 ADC 规则转换 1: 写 1 启动 ADC 规则转换，读为 1 表明 ADC 正在操作，可能正在转换。 该位由硬件清除的情况： – 在单次转换模式 (CONT=0, DISCEN=0)，选择软件驱动时(EXTEN=00)：序列转换完成时（EOS 标志置位） – 在非连续转换模式 (CONT=0, DISCEN=1),当软件驱动时(EXTEN=00)：转换完成时（EOC 标志置位） – 其他情况下：执行 ADSTP 命令之后，同时 ADSTP 标志又被硬件清 0 之时 注： 1. 软件只有当 ADEN=1 且 ADDIS=0 时才能配置 ADSTART=1. 软件写该位为 0 为无效操作。 2. 自动注入模式（JAUTO=1），置位 ADSTART 能启动规则和自动注入转换。
1	ADDIS	RS	0	ADEN 禁止使能。 软件置位禁止 ADC。当 ADC 被禁止（ADEN 被硬件清零），硬件清除该位。软件写该位为 0 为无效操作。 0：当前未执行 ADDIS 命令 1: 写 1 禁止 ADC，读 1 表示 ADDIS 指令正在执行 注：当 ADEN=1，ADSTART=0 且 JADSTART=0（表明没有正在进行的转换），才能设置 ADDIS 为 1。

0	ADEN	RS	0	ADC 使能命令。 软件置位该位使能 ADC，ADRDY 标志置 1 后， ADC 准备好转换操作。软件写该位为 0 为无效操作。 0: 禁止 ADC (OFF 状态) 1: 使能 ADC 注：只有当 ADC_CR 寄存器的所有位为 0 时才能配置 ADEN 为 1.

注：软件将 ADEN 清零后再重新使能 ADEN 和 ADSTART 的间隔为 4 个 ADC_CLK 周期。

13.5.4. ADC 配置寄存器 (ADC_CFGR)

偏移地址: 0x0C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AWDCH[5:0]						JAUTO	JAWDEN	AWDEN	AWDSGL	Res.	JDISCEN	DISNUM[2:0]			DISCEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALIGN	AUTDLY	CONT	OVRMOD	EXTEN[1:0]		Res.	EXTSEL[3:0]				RES[1:0]		Res.	DMAFG	DMAEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	AWDCH[5:0]	RW	0000	模拟看门狗通道选择，软件可清除和设置该位。 模拟看门狗监测选择的输入通道 000000：外部通道 0 000001：外部通道 1 …. 010111：外部通道 23 011000：保留 011001：保留 011010：外部通道 26 011011：内部 V_{TS} 011100：内部 V_{REFINT} 011101：内部 $V_{CC}/3$ 011110：内部 DAC 输出 011111：内部 OPA 输出 其它：保留 注：只有当 ADSTART=0 且 JADSTART=0 时才能配置该位。
25	JAUTO	RW	1’b0	自动注入使能 该位由软件置位和清零，用于使能/禁止规则通道组转换结束后自动进行注入通道组转换 0：禁止注入通道组自动转换 1：使能注入通道组自动转换 注：只有当 ADSTART=0 且 JADSTART=0 时才能配置该位。
24	JAWDEN	RW	1’b0	注入通道模拟看门狗使能 在注入通道上使能模拟看门狗。该位由软件置位和清零。

				0：在注入通道上禁用模拟看门狗 1：在注入通道上使能模拟看门狗 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
23	AWDEN	RW	0	规则通道模拟看门狗使能 在规则通道上使能模拟看门狗。该位由软件置位和清零。 0：禁止模拟看门狗 1：使能模拟看门狗 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
22	AWDSGL	RW	0	在一个通道或者所有通道使能模拟看门狗 软件通过置位和清零该位，可以在 AWDCH[5:0]位设置的通道上或者所有通道上使能或者禁止模拟看门狗。 0：在所有通道上使能模拟看门狗 1：在一个通道上使能模拟看门狗（AWDCH[5:0]配置哪个通道） 注：仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
21	Reserved	-	-	保留
20	JDISCEN	RW	1’b0	注入通道非连续模式使能。 软件通过置位和清零该位，可以使能或者禁止注入通道组上的非连续模式。 0：注入通道组上禁用非连续模式 1：注入通道组上使能非连续模式 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
19:17	DISCNUM[2:0]	RW	3’b0	非连续模式转换通道数 这些位是由软件写入，用于定义在接收到外部触发后，在非连续模式下要转换的规则通道的数量。 000:1 通道 001:2 通道 … 111:8 通道 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
16	DISCEN	RW	0	非连续模式使能。 软件通过置位和清零该位，使能/禁止非连续模式。 0：禁止非连续模式 1：使能非连续模式 不可能既使能非连续模式又使能连续模式，即禁止设置 DISCEN=1 和 CONT=1. 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
15	ALIGN	RW	0	数据对齐。 软件通过置位和清零该位，选择右对齐或左对齐。 0：右对齐 1：左对齐

				注：仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写该位。
14	AUTDLY	RW	0	延迟转换模式。 软件通过置位和清零该位，使能/禁止等待转换模式。 0：禁止等待转换模式 1：使能等待转换模式 注：仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
13	CONT	RW	0	单次/连续转换模式。 软件可设置和清除该位。如果置为 1，直到该位被清除，否则在有触发发生时会一直发生转换。 不可能既使能非连续模式又使能连续模式，即禁止设置 DISCEN=1 和 CONT=1. 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位
12	OVRMOD	RW	0	过载管理模式 软件通过置位和清零该位，配置数据过载管理的方式。 0：当过载发生时，ADC_DR 寄存器保留原有数据 1：当过载发生时，ADC_DR 寄存器会被上一次转换结果覆盖 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
11:10	EXTEN[1:0]	RW	00	外部触发使能和极性选择 软件通过置位和清零该位，选择触发极性和使能触发。 00：硬件触发检测不使能（软件启动转换） 01：上升沿硬件触发检测 10：下降沿硬件触发检测 11：上升沿和下降沿硬件触发检测 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
9	Reserved	-	-	保留
8:5	EXTSEL[3:0]	RW	000	外部触发选择。 该位选择触发规则转换启动的外部事件。 0000：TRG0(TIM1_TRGO) 0001：TRG1(TIM1_OC1) 0010：TRG2(TIM1_OC2) 0011：TRG3(TIM1_OC3) 0100：TRG4(TIM1_OC4) 0101：TRG5(TIM2_TRGO) 0110：TRG6(TIM2_OC1) 0111：TRG7(TIM3_OC1) 1000：TRG8(TIM3_TRGO) 1001：TRG9(TIM15_TRGO) 1010：保留 1011：保留 1100：TRG12(EXTI11) 1101：TRG13(EXTI15)

				1110：保留 1111：保留 **注意：**当选择 EXTI11 或者 EXTI15 作为触发时，触发极性按照 EXTI 模块的极性定义，即只要满足 EXTI 的极性定义，EXTEN 配置为上升沿或者下降沿都可以触发转换。
4:3	RES[1:0]	RW	00	数据分辨率 软件设置该位选择转换分辨率。 00：12 位 01：10 位 10：8 位 11：6 位 注： 1. 仅当 ADEN=0 时运行软件写这些位 2. 仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
2	Reserved	-	-	保留
1	DMACFG	RW		DMA 访问模式配置 软件通过置位和清零该位，在两种 DMA 模式操作中选择并在 DMAEN = 1 时有效。 0：DMA 单次模式选择 1：DMA 循环模式选择 注：仅当 ADSTART=0 时且 JADSTART=0（确保没有正在进行的转换）允许软件写这些位。
0	DMAEN	RW	0	DMA 访问使能。 软件通过置位和清零该位，使能/禁止 DMA 请求。利用 DMA 控制器管理自动转换数据。 0：禁止 DMA 1：使能 DMA 注：仅当 ADSTART=0 时且 JADSTART=0（确保没有正在进行的转换）允许软件写这些位。

13.5.5. ADC 配置寄存器 2 (ADC_CFGR2)

偏移地址: 0x10

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CALNUM[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GCOMP
	RW	RW	RW												RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	ROVSM	TROVS	OVSS[3:0]				OVSR[2:0]			JOVSE	ROVSE
					RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:28	CALNUM[2:0]	RW	3’b0	校准平均次数 000: 1 次校准和平均 001: 2 次校准和平均 010: 4 次校准和平均

				011: 8 次校准和平均 100: 16 次校准和平均 101: 32 次校准和平均 其他: 保留
27:17	Reserved	-	-	保留
16	GCOMP	RW	0	增益补偿模式 软件通过置位和清零该位，以使能/禁止增益补偿模式。 0：禁止增益补偿 1：使能增益补偿，并应用于所有通道 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）
15:11	Reserved	-	-	保留
10	ROVSM	RW	1’b0	规则过采样模式 软件通过置位和清零该位，以选择规则过采样模式。 0：连续模式：触发注入转换时，过采样暂时停止，并在注入序列后继续（注入序列期间保留过采样缓冲区） 1：恢复模式：触发注射转换时，当前过采样被中止，并在注入序列后从头开始进行过采样（过采样缓冲区在注入序列开始转换时清零） 注意：只有当 ADSTART=0 时，软件才允许写入此位（这确保了没有转换正在进行）。
9	TROVS	RW	1’b0	触发规则过采样 软件通过置位和清零该位，以使能/禁止已触发过采样 0：在触发后连续完成通道的所有过采样转换 1：某一通道的每个过采样转换都需要重新触发 注意：只有当 ADSTART=0 时，才允许软件写入此位（这确保没有转换正在进行）
8:5	OVSS[3:0]	RW	4’b0	过采样移位控制 该位字段由软件配置，以定义应用于原始过采样结果的右移位数。 0000：无移位 0001：移位 1 位 0010：移位 2 位 0011：移位 3 位 0100：移位 4 位 0101：移位 5 位 0110：移位 6 位 0111：移位 7 位 1000: 移位 8 位 其他：预留 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入这些位（这确保了没有转换正在进行）。
4:2	OVSR[2:0]	RW	3’b0	过采样率 此位字段由软件配置，以定义过采样率。 000：2x 001：4x 010：8x

				011：16x 100：32x 101：64x 110：128x 111：256x 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入这些位（这可以确保没有转换正在进行）。
1	JOVSE	RW	1’b0	注入过采样使能 软件通过置位和清零该位，以使能/禁止注入过采样。 0：禁用注入过采样 1：使能注入过采样 注意：只有当 ADSTART=0 且 JADSTART=0 时，软件才允许写入此位（这确保没有正在进行的转换）
0	ROVSE	RW	1’b0	规则过采样使能 软件通过置位和清零该位，以使能/禁止规则过采样。 0：禁用规则过采样 1：使能规则过采样 注意：只有当 ADSTART=0 且 JADSTART=0 时，软件才允许写入此位（这确保没有正在进行的转换）

13.5.6. ADC 采样时间寄存器 1 (ADC_SMPR1)

偏移地址: 0x14

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP5[0]	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:0	SMPx[2:0]	RW	3’b000	采样时钟选择。 软件可配置该位选择通道 x 的采样时间。 000：2.5 ADC_CLK 001：6.5 ADC_CLK 010：12.5 ADC_CLK 011：24.5 ADC_CLK 100：47.5 ADC_CLK 101：92.5 ADC_CLK 110：247.5 ADC_CLK 111：640.5 ADC_CLK 仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。 注：当频率 ADC_CLK=PCLK 时，采样时间不能配置为 000（2.5ADC_CLK）

13.5.7. ADC 采样时间寄存器 2 (ADC_SMPR2)

偏移地址: 0x18

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP15[0]	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:0	SMPx[2:0]	RW	3’b000	参见 ADC_SMPR1 的描述

13.5.8. ADC 采样时间寄存器 3 (ADC_SMPR3)

偏移地址: 0x1C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP29[2:0]			SMP28[2:0]			SMP27[2:0]			SMP26[2:0]			SMP25[2:1]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP25[0]	SMP24[2:0]			SMP23[2:0]			SMP22[2:0]			SMP21[2:0]			SMP20[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:0	SMPx[2:0]	RW	3’b000	参见 ADC_SMPR1 的描述

13.5.9. ADC 采样时间寄存器 4 (ADC_SMPR4)

偏移地址: 0x20

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMP31[2:0]			SMP30[2:0]		
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5:0	SMPx[2:0]	RW	3’b000	参见 ADC_SMPR1 的描述

13.5.10. ADC 看门狗阈值寄存器 (ADC_TR)

偏移地址: 0x24

复位值: 0x0FFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	HT[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	LT[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	复位值	Function
31:28	Reserved	-	-	保留
27:16	HT[11:0]	RW	12'hFFF	模拟看门狗高阈值。 软件可配，定义模拟看门狗高阈值。 仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
15:12	Reserved	-	-	保留
11:0	LT[11:0]	RW	12'h0	模拟看门狗低阈值。 软件可配，定义模拟看门狗低阈值。 仅当 ADSTART=0 且 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。

13.5.11. ADC 通道选择寄存器 1 (ADC_SQR1)

偏移地址: 0x30

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ4[5:0]						SQ3[5:0]						SQ2[5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ2[3:0]				SQ1[5:0]						Res.	Res.	L[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:24	SQ4[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 4 个转换，这些位定义了转换序列中的第 4 个转换通道的编号。 对应通道参见 SQ1 定义。
23:18	SQ3[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 3 个转换，这些位定义了转换序列中的第 3 个转换通道的编号。 对应通道参见 SQ1 定义。
17:12	SQ2[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 2 个转换，这些位定义了转换序列中的第 2 个转换通道的编号。 对应通道参见 SQ1 定义。
11:6	SQ1[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 1 个转换，这些位定义了转换序列中的第 1 个转换通道的编号。 对应通道如下： 000000：外部通道 0（PA0） 000001：外部通道 1（PA1）

				000010：外部通道 2（PA2） 000011：外部通道 3（PA3） 000100：外部通道 4（PB7） 000101：外部通道 5（PB8） 000110：外部通道 6（PC12） 000111：外部通道 7（PC13） 001000：外部通道 8（PC14） 001001：外部通道 9（PC15） 001010：外部通道 10（PD0） 001011：外部通道 11（PD1） 001100：外部通道 12（PD2） 001101：外部通道 13（PD3） 001110：外部通道 14（PD4） 001111：外部通道 15（PD7） 010000：外部通道 16（PD8） 010001：外部通道 17（PA11） 010010：外部通道 18（PA12） 010011：外部通道 19（PA13） 010100：外部通道 20（PA14） 010101：外部通道 21（PC10） 010110：外部通道 22（PC11） 010111：外部通道 23（PD9） 011000：保留 011001：保留 011010：外部通道 26（PD12） 011011：内部 T_S 011100：内部 V_{REFINT} 011101：内部 $V_{CC}/3$ 011110：内部 DAC 输出 011111：内部 OPA 输出 其他：保留 仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
5:4	Reserved	-	-	保留
3:0	L[3:0]	RW	4’b0	规则通道序列长度 软件配置这些位的值。这些位定义了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 仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。

13.5.12. ADC 通道选择寄存器 2 (ADC_SQR2)

偏移地址: 0x34

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ9[5:0]						SQ8[5:0]						SQ7[5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ7[3:0]				SQ6[5:0]						SQ5[5:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:24	SQ9[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 9 个转换，这些位定义了转换序列中的第 9 个转换通道的编号。 对应通道参见 SQ1 定义。
23:18	SQ8[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 8 个转换，这些位定义了转换序列中的第 8 个转换通道的编号。 对应通道参见 SQ1 定义。
17:12	SQ7[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 7 个转换，这些位定义了转换序列中的第 7 个转换通道的编号。 对应通道参见 SQ1 定义。
11:6	SQ6[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 6 个转换，这些位定义了转换序列中的第 6 个转换通道的编号。 对应通道参见 SQ1 定义。
5:0	SQ5[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 5 个转换，这些位定义了转换序列中的第 5 个转换通道的编号。 对应通道参见 SQ1 定义。

13.5.13. ADC 通道选择寄存器 3 (ADC_SQR3)

偏移地址: 0x38

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SQ14[5:0]						SQ13[5:0]						SQ12[5:4]	
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ12[3:0]				SQ11[5:0]						SQ10[5:0]					
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:24	SQ14[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 14 个转换，这些位定义了转换序列中的第 14 个转换通道的编号。

				对应通道参见 SQ1 定义。
23:18	SQ13[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 13 个转换，这些位定义了转换序列中的第 13 个转换通道的编号。 对应通道参见 SQ1 定义。
17:12	SQ12[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 12 个转换，这些位定义了转换序列中的第 12 个转换通道的编号。 对应通道参见 SQ1 定义。
11:6	SQ11[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 11 个转换，这些位定义了转换序列中的第 11 个转换通道的编号。 对应通道参见 SQ1 定义。
5:0	SQ10[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 10 个转换，这些位定义了转换序列中的第 10 个转换通道的编号。 对应通道参见 SQ1 定义。

13.5.14. ADC 通道选择寄存器 4 (ADC_SQR4)

偏移地址: 0x3C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	SQ16[5:0]						SQ15[5:0]					
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:6	SQ16[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 16 个转换，这些位定义了转换序列中的第 16 个转换通道的编号。 对应通道参见 SQ1 定义。
5:0	SQ15[5:0]	RW	6'b0	通道选择。 软件配置这些位的值。规则序列中的第 15 个转换，这些位定义了转换序列中的第 15 个转换通道的编号。 对应通道参见 SQ1 定义。

13.5.15. ADC 数据寄存器（ADC_DR)

偏移地址: 0x40

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

RDATA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	RDATA[15:0]	R	16'h0	规则通道转换数据。 这些位是只读的。上次转换通道的转换结果放于此寄存器。数据是左对齐或者右对齐的。

13.5.16. ADC 注入通道选择寄存器（ADC_JSQR)

偏移地址:0x4C

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JSQ4[5:0]						JSQ3[5:0]						JSQ2[5:2]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JSQ2[1:0]		JSQ1[5:0]						JEXTEN[1:0]		JEXTSEL[3:0]				JL[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	JSQ4[5:0]	RW	6'b0	注入序列第 4 次转换 软件配置这些位的值。注入序列中的第 4 个转换，这些位定义了注入转换序列中的第 4 个转换通道的编号。 对应通道参见 ADC_SQR1.SQ1 定义。 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
25:20	JSQ3[5:0]	RW	6'b0	注入序列第 3 次转换 软件配置这些位的值。注入序列中的第 3 个转换，这些位定义了注入转换序列中的第 3 个转换通道的编号。 对应通道参见 ADC_SQR1.SQ1 定义。 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
19:14	JSQ2[5:0]	RW	6'b0	注入序列第 2 次转换 软件配置这些位的值。注入序列中的第 2 个转换，这些位定义了注入转换序列中的第 2 个转换通道的编号。 对应通道参见 ADC_SQR1.SQ1 定义。 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
13:8	JSQ1[5:0]	RW	6'b0	注入序列第 1 次转换 软件配置这些位的值。注入序列中的第 1 个转换，这些位定义了注入转换序列中的第 1 个转换通道的编号。 对应通道参见 ADC_SQR1.SQ1 定义。 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
7:6	JEXTEN[1:0]	RW	2'b0	注入通道触发使能和极性选择 这些位由软件设置和清除，以选择外部触发极性并使能注入组的触发。 00：禁用硬件触发检测（可以通过软件启动转换） 01：上升沿的硬件触发检测

				10：下降沿的软件触发检测 11：上升沿和下降沿都有硬件触发检测 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
5:2	JEXTSEL[3:0]	RW	5’b0	注入通道组外部触发事件选择。 该寄存器选择触发注入转换启动的外部事件。 0000：TRG0(TIM1_TRGO) 0001：TRG1(TIM1_OC1) 0010：TRG2(TIM1_OC2) 0011：TRG3(TIM1_OC3) 0100：TRG4(TIM1_OC4) 0101：TRG5(TIM2_TRGO) 0110：TRG6(TIM2_OC1) 0111：TRG7(TIM3_OC1) 1000：TRG8(TIM3_TRGO) 1001：TRG9(TIM15_TRGO) 1010：保留 1011：保留 1100：TRG12(EXTI11) 1101：TRG13(EXTI15) 1110：保留 1111：保留 **注意：**当选择 EXTI11 或者 EXTI15 作为触发时，触发极性按照 EXTI 模块的极性定义，即只要满足 EXTI 的极性定义，EXTEN 配置为上升沿或者下降沿都可以触发转换。 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
1:0	JL[1:0]	RW	2’b0	注入通道序列长度 这些位由软件编写，用于定义注入通道转换序列中的转换总数。 00：1 个转换 01：2 个转换 10：3 个转换 11：4 个转换 注：仅当 JADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。

13.5.17. ADC 偏移寄存器（ADC_OFRy）

偏移地址: 0x60+0x04(y-1)，（y=1 to 4）

复位值: 0x0000 0000

注：SATEN、OFFSETPOS 只有 ADC_OFR1 有，并且 ADC_OFR1~4 共用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSETy_EN	OFFSETy_CH[5:0]						SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	OFFSETy[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	OFFSETy_EN	RW	1’b0	OFFSETy 使能 该位由软件写入，以使能或禁用偏移值 OFFSETy。 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。
30:25	OFFSETy_CH[5:0]	RW	6’b0	数据偏移 y 的通道选择 这些位由软件写入，以定义 OFFSETy 的偏移将应用的通道。 某些通道没有物理连接，因此不能为数据偏移量 y 选择。 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。
24	SATEN	RW	1’b0	饱和使能 此位由软件设置和清除，以使能偏移在 0x000 和 0xFFF 饱和。 0：无饱和控制，偏移结果可以有符号 1：启用饱和，偏移结果无符号且在 0x000 和 0xFFF 处饱和 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。
23	OFFSETPOS	RW	1’b0	正偏移 此位由软件设置和清除，以使能正偏移。 0：负偏移 1：正偏移 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。
22:12	Reserved	-	-	保留
11:0	OFFSETy[11:0]	RW	12’b0	OFFSETy_CH[4:0]通道的数据偏移 y。 这些位由软件写入，以定义在转换通道（可以是规则的或注入的）时从原始转换数据中减去的偏移 y。应用数据偏移 y 的通道必须在位 OFFSETy_CH 中配置。转换结果可以从 ADC_DR（规则转换）或 ADC_JDRy 寄存器（注入转换）中读取。 如果多个偏移（OFFSETy）指向同一通道，则仅考虑具有最低 x 值的偏移进行减法。 例如：如果 OFFSET1_CH=4 且 OFFSET2_CH=4，则这是在转换通道 4 时减去 OFFSET1[11:0]。 注意：只有当 ADSTART=0、JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。

13.5.18. ADC 注入通道数据寄存器（ADC_JDRy)

偏移地址: 0x80+0x04(y-1)，（y=1 to 4）

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	JDATA[15:0]	R	16’h0	注入转换数据

				这些位只读。注入通道 y 的转换结果放于此寄存器。数据是左对齐或者右对齐的。

13.5.19. ADC 校准因子寄存器（ADC_CALFACT)

偏移地址: 0xB4

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCALFACT[8:0]									Res.	CAPSUC	OFFSUC	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R	R	R		RC_W1	RC_W1				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDVLD	WRVLD	FACTSEL[4:0]					WCALFACT [8:0]								
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	RCALFACT[8:0]	R	9’h00	读校准结果寄存器（补码表示）。 当 RDVLD 有效时，读取 ADC 校准结果。只有 ADCAL 为 0 时，软件才能读取。 当 RDVLD 有效时，读取溢出标志位。只有 ADCAL 为 0 时，软件才能读取。
22	Reserved	-	-	保留
21	CAPSUC	RC_W1	1’h0	电容校准状态位。 表示 ADC 电容校准是否成功。硬件置位，软件写 1 清零。 0：电容校准失败 1：电容校准成功
20	OFFSUC	RC_W1	1’h0	偏移校准状态位。 表示 ADC 偏移校准是否成功。硬件置位，软件写 1 清零。 0：偏移校准失败 1：偏移校准成功
19:16	Reserved	-	-	保留
15	RDVLD	RW	1’h0	溢出标志位读使能。 0：禁止读溢出标志 1：使能读溢出标志 也是校准因子读使能。 0：禁止读校准因子 1：使能读校准因子
14	WRVLD	RW	1’h0	校准因子写使能。 0：禁止写校准因子 1：使能写校准因子
13:9	FACTSEL[4:0]	RW	5’h00	读/写校准值选择位 当 RDVLD/WRVLD 有效时，选择需要读的校准结果/写入的校准因子加载到哪个电容。 {cov_error,cal_error}由硬件置位，软件清零。 读{cov_error,cal_error}时需使能 RDVLD。 写{cov_error,cal_error}时需使能 WDVLD。 通过写 WCALFACT[1]清零 cov_error，通过写 WCALFACT[0]清零 cal_error。 5’d1：dos

				5’d2：dcpm5 5’d3：dcpm4 5’d4：dcpm3 5’d5：dcpm2 5’d6：dcpm1 5’d7：dcpn4 5’d8：dcpn3 5’d9：dcpn2 5’d10：dcpn1 5’d11：dcpk3 5’d12：dcm5 5’d13：dcm4 5’d14：dcm3 5’d15：dcm2 5’d16：dcm1 5’d17：dcnm0 5’d18：{cov_error,cal_error}: cov_error: 当 AD 转换时，将校准结果补偿到 AD 转换时溢出，硬件置位该位，软件写 1 清零。 cal_error: 当校准时，校准过程中产生溢出，硬件置位该位，软件写 1 清零。
8:0	WCALFACT [8:0]	RW	9’h00	写校准因子寄存器（补码表示）。 当 WRVLD 有效时，WCALFACT 值加载到 ADC 中。 当 ADCAL 为 0 时，软件才能写入。

13.5.20. ADC 增益补偿寄存器（ADC_GCOMP）

偏移地址: 0xC0

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	GCOMPCOEFF[13:0]													
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	GCOMPCOEFF[13:0]	RW	14’h0	增益补偿系数 这些位由软件配置，以编程增益补偿系数。 00 1000 0000 0000：增益因子为 0.5 … 01 0000 0000 0000：增益因子为 1 10 0000 0000 0000：增益因子为 2 11 0000 0000 0000：增益因子为 3 … 该系数除以 4096，得到范围从 0 到 3.999756 的增益因子。 注意：此增益补偿仅在 ADC_CFGR2 寄存器的 GCOMP 位为 1 时有效。

13.5.21. ADC 通用配置寄存器（ADC_CCR)

偏移地址: 0x308

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	DIFF_EN	VREFSEL	PWR_MODE[2:0]			Res.	TSEN	VREFINT_EN	PRESC[3:0]				CKMODE[1:0]	
		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	DIFF_EN	RW	0	差分输入使能 0：单端输入 1：差分输入
28	VREFSEL	RW	0	ADC 参考电压选择。 0：选择 V_{REFP}（V_{REFP} 不存在时连接 V_{CC}） 1：选择 V_{REFBUF}（需要使能 VREFBUF_CR.VREFBUF_EN）
27:25	PWR_MODE[2:0]	RW	3’b0	ADC 内比较器的功耗模式配置 000: typical 6μA 001: typical 7μA 010: typical 8μA 011: typical 9μA 100: typical 2μA 101: typical 3μA 110: typical 4μA 111: typical 5μA 注： 仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、JADSTART=0， ADSTP=0、JADSTP=0，ADDIS=0 且 ADEN=0），才允许通过软件对这些位执行写操作。
24	Reserved	-	-	保留
23	TSEN	RW	0	温度传感器使能位，软件可设置和清除该位，使能/禁止温度传感器。 0：温度传感器通道禁止 1：温度传感器通道使能 注：仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、JADSTART=0， ADSTP=0、JADSTP=0， ADDIS=0 且 ADEN=0），才允许通过软件对该位执行写操作。
22	VREFINT_EN	RW	0	基准 V_{REFINT} 使能位，软件可设置和清除该位，使能/不使能基准 V_{REFINT} 0：不使能 1：使能 注：仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、JADSTART=0， ADSTP=0、JADSTP=0， ADDIS=0 且 ADEN=0），才允许通过软件对该位执行写操作。
21:18	PRESC[3:0]	RW	0	ADC 预分频系数，针对异步时钟模式

				由软件配置，以选择 ADC 的时钟频率。 0000：输入 ADC 时钟未分频 0001：输入 ADC 时钟 2 分频 0010：输入 ADC 时钟 4 分频 0011：输入 ADC 时钟 6 分频 0100：输入 ADC 时钟 8 分频 0101：输入 ADC 时钟 10 分频 0110：输入 ADC 时钟 12 分频 0111：输入 ADC 时钟 16 分频 1000：输入 ADC 时钟 32 分频 1001：输入 ADC 时钟 64 分频 1010：输入 ADC 时钟 128 分频 1011：输入 ADC 时钟 256 分频 其他值：保留 注：仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、JADSTART=0，ADSTP=0、JADSTP=0，ADDIS=0 且 ADEN=0），才允许通过软件对这些位执行写操作。
17:16	CKMODE[1:0]	RW	0	ADC 时钟模式 由软件配置，用于定义为模拟 ADC 提供时钟的方式： 00：ADC_CLK（异步时钟模式），由 RCC 配置源 01：PCLK/2（同步时钟模式） 10：PCLK/4（同步时钟模式） 11：PCLK（同步时钟模式）。使能此配置时， PCLK 的时钟占空比必须为 50%。 在所有同步时钟模式下，从定时器触发到转换开始的延迟过程中，不存在抖动。 注： 仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、JADSTART=0， ADSTP=0、JADSTP=0， ADDIS=0 且 ADEN=0），才允许通过软件对这些位执行写操作。
15:0	Reserved	-	-	保留

14. 数模转换器(DAC)

14.1. DAC 简介

数字/模拟转换模块(DAC)是 12 位数字输入电压输出的数字/模拟转换器。DAC 可以配置为 8 位或 12 位模式，也可以与 DMA 控制器配合使用。 DAC 工作在 12 位模式时，数据可以设置成左对齐或右对齐。DAC 可以通过引脚输入参考电压 V_{REFP}（V_{REFP} 是否存在参见数据手册）以获得更精确的转换结果。

14.2. DAC 主要特性

- 1 个 DAC 转换器：每个转换器对应 1 个输出通道
- 12 位模式下数据左对齐或者右对齐
- 同步更新功能
- 噪声波形生成
- 三角波形生成
- 通道支持 DMA 功能
- 支持 DMA 下溢错误检测
- 外部触发转换
- 输入参考电压 V_{REFP}（V_{CCA}）、和 V_{REFBUF}

14.3. DAC 功能描述

14.3.1. DAC 框图

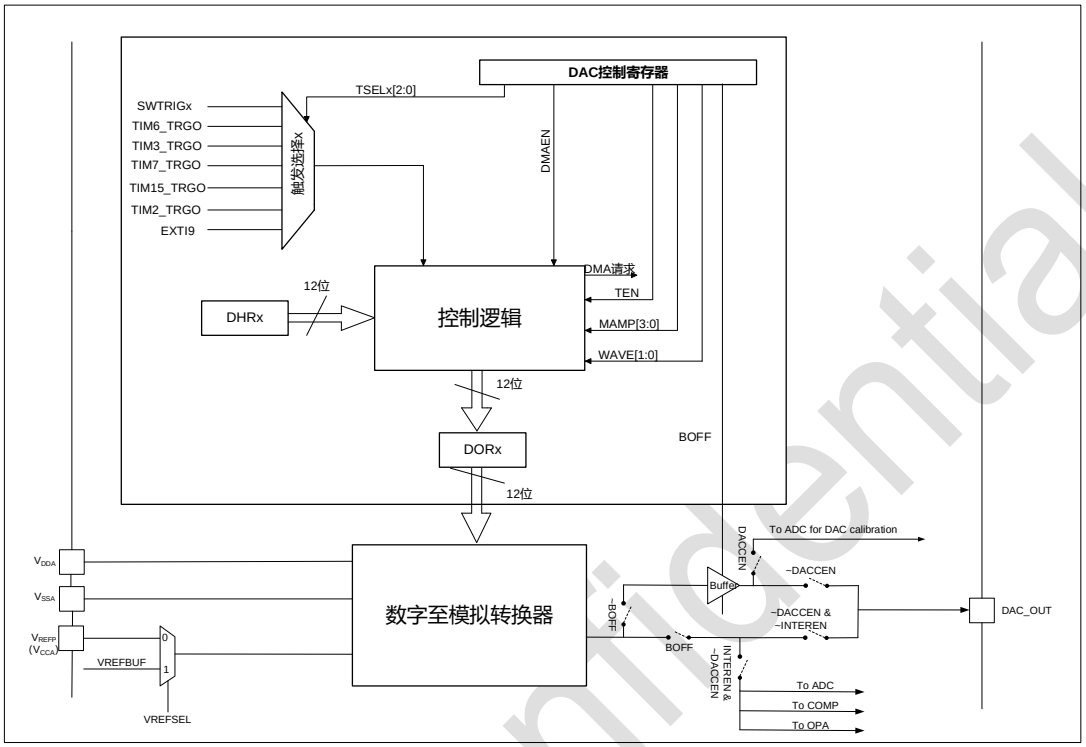


图 14-1 DAC 框图

14.3.2. DAC 通道使能

将 DAC_CR1 寄存器的 EN 位置‘1’即可打开对 DAC 的供电。经过一段启动时间 t_{WAKEUP}（数值参见数据手册），DAC 即被使能。

注意：EN 位只会使能 DAC 的模拟部分，即便该位被置‘0’， DAC 的数字部分（如寄存器）仍然工作。

14.3.3. DAC 输出缓存使能

DAC 集成了 1 个输出缓存，可以用来减少输出阻抗，无需外部运放即可直接驱动外部负载。输出缓存可以通过设置 DAC_CR1 寄存器的 BOFF 位来使能或者关闭。

14.3.4. DAC 校准

14.3.4.1. DAC 外部校准（通过 ADC）

DAC 可以通过 ADC 完成校准，外部参考电压与内部参考电压可单独保存校准值，以选择内部参考电压 V_{REFBUF}（VREFSEL=1）为例，步骤如下：

1. 将 ADC 使能，ADC 选择内部通道模式，内部通道选择 DAC 输入（ADC 需要先校准好）
2. 使能 V_{REFBUF}，并根据需求选择 VREFBUF 电压值

3. 配置 DAC_CR1 寄存器的 EN 和 DACCEN 为 1，BOFF=0；DAC_CALR 寄存器的 DAC_OS_TRIMCR_VREFBUF = 6’h11。

4. 固定 DAC 的 DAC_DOR.DACDOR[11:0]=0x7FF（2047），调节 DAC_OS_TRIMCR_VREFBUF 的值，直至 ADC 的输出值为 2047±1，此时校准完成。

注：Trim 范围为±15mV，DAC_OS_TRIMCR_VREFBUF 增加两位，ADC 的输出会增加 1LSB @3.3V 25C

14.3.4.2. DAC 内部校准

DAC 可以内部校准，外部参考电压与内部参考电压可单独保存校准值。以选择内部参考电压 V_{REFBUF}（VREFSEL=1）为例，步骤如下：

1. 配置 DAC_CR1 寄存器的的 EN=1, BUF_CAL =1,BOFF=0；DAC_CALR 寄存器的 DAC_OS_TRIMCR_VREFBUF[5:0]=6’h11。

2. 固定 DAC 的 DAC_DOR.DACDOR[11:0]=12’h7FF（2047），调节 DAC_OS_TRIMCR_VREFBUF[5:0]的值，观测 DAC_DOUT 寄存器的值，当该寄存器的值从 0 变到 1 时，记录此时的 DAC_OS_TRIMCR_VREFBUF [5:0]的值。

14.3.5. DAC 数据格式

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 数据对齐方式，有 3 种情况：
  - 8 位数据右对齐：用户须将数据写入寄存器 DAC_DHR8R[7:0]位(实际是存入寄存器 DACDHR [11:4]位)
  - 12 位数据左对齐：用户须将数据写入寄存器 DAC_DHR12L[15:4]位(实际是存入寄存器 DACDHR [11:0]位)
  - 12 位数据右对齐：用户须将数据写入寄存器 DAC_DHR12R[11:0]位(实际是存入寄存器 DACDHR[11:0]位)

根据对 DAC_DHRyyyx 寄存器的操作，经过相应的移位后，写入的数据被转存到 DACDHR 寄存器中(DACDHR 是内部的数据保存寄存器)。随后，DACDHR 寄存器的内容或被自动地传送到 DACDOR 寄存器，或通过软件触发或外部事件触发被传送到 DACDOR 寄存器。

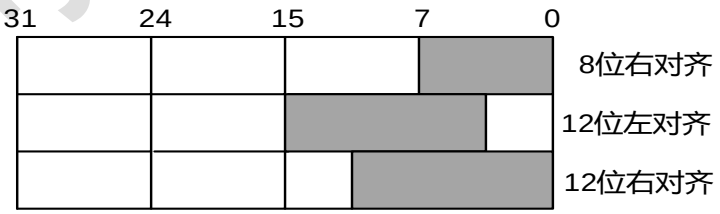


图 14-2 单 DAC 通道模式的数据寄存器

14.3.6. DAC 转换

不能直接对寄存器 DAC_DOR 写入数据，任何输出到 DAC 的数据都必须写入 DACDHR 寄存器(数据实际写入 DAC_DHR8R、DAC_DHR12L、DAC_DHR12R 寄存器)。

如果没有选中硬件触发(寄存器 DAC_CR1 的 TEN 位置‘0’)，存入寄存器 DACDHR 的数据会在 1 个 APB 时钟周期后自动传至寄存器 DACDOR。如果选中硬件触发(寄存器 DAC_CR1 的 TEN 位置‘1’)，数据传输在触发发生以后 3 个 APB 时钟周期后完成。

一旦数据从 DACDHR 寄存器装入 DACDOR 寄存器，在经过时间 $t_{SETTLING}$（数值参见数据手册）之后，输出即有效，这段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。

注：当 DAC 未使能时，DACDOR 的值不会输出到模拟。

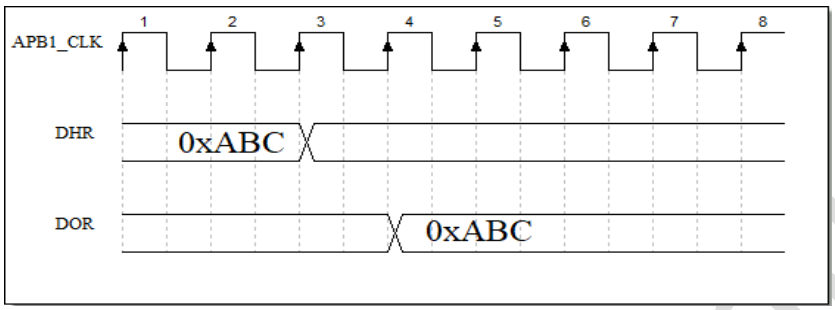


图 14-3 TEN=0 时转换时序图

14.3.7. DAC 输出电压

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 V_{REFP}。

任一 DAC 通道引脚上的输出电压满足下面的关系：

DAC 输出 = V_{REFP} x (DACDOR / 4096)。

14.3.8. DAC 触发选择

如果 TEN 位被置 1， DAC 转换可以由某外部事件触发(定时器计数器、外部中断线)。配置控制位 TSEL[2:0]可以选择 8 个触发事件之一触发 DAC 转换，如下表所示（本产品支持 7 个有效触发）。

表 14-1 DAC 触发选择

触发源	类型	TSEL[2:0]
定时器 6 TRGO 事件	来自片上定时器的内部信号	000
定时器 3 TRGO 事件		001
定时器 7 TRGO 事件		010
定时器 15 TRGO 事件		011
定时器 2 TRGO 事件		100
保留		101
EXTI Line9	外部引脚	110
SWTRIG(软件触发)	软件控制位	111

每次 DAC 接口侦测到来自选中的定时器 TRGO 或者外部中断线 9 的上升沿输出时，最近存放在寄存器 DACDHR 中的数据会被传送到寄存器 DACDOR 中。在 3 个 APB 时钟周期后，寄存器 DACDOR 更新为新值。

如果选择软件触发，一旦 DAC_CR2.SWTRIG 位置‘1’，转换即开始。在数据从 DACDHR 寄存器传送到 DACDOR 寄存器后，SWTRIG 位由硬件自动清‘0’。

注意：

1、不能在 EN 为‘1’时改变 TSEL[2:0]位。

2、如果选择软件触发，数据从寄存器 DACDHR 传送到寄存器 DACDOR 只需要 1 个 APB 时钟周期。

3、触发频率不能超过 1M。

14.3.9. DMA 功能

14.3.9.1. DMA 请求

DAC 具有 DMA 功能。

如果 DMAEN 位置‘1’，一旦有外部触发(而不是软件触发)发生，则会在 3 个时钟周期后产生一个 DMA 请求，然后 DACDHR 寄存器的数据被传送到 DACDOR 寄存器。

14.3.9.2. DMA 下溢检测

DAC 的 DMA 请求没有缓冲队列，因此如果第二个外部触发在接收到对第一个外部触发的应答(第一个请求)之前到达，则不会发出新的 DMA 请求，并将 DAC_SR 寄存器中的 DMA 下溢标志 DMAUDR 将置“1”，报告错误情况。然后禁用 DMA 数据传输，不再处理任何 DMA 请求。DAC 通道继续转换旧数据。

软件应通过写入“1”来将 DMAUDR 标志清零，将所用 DMA 通道的使能位清零，并重新初始化 DMA 和 DAC 通道，以便正确地重新开始 DMA 传输。同时软件应修改 DAC 触发转换频率或减轻 DMA 工作负载，以避免再次发生 DMA 下溢情况。最后，可通过使能 DMA 通道以及配置外部触发来继续完成 DAC 转换。

对于各 DAC 通道，如果使能 DAC_CR1 寄存器中相应的 DMAUDRIE 位，还将产生相应的中断。

14.3.10. DAC 噪声生成

可以利用线性反馈移位寄存器(LFSR)产生幅度变化的伪噪声。设置 WAVE[1:0]位为‘01’选择 DAC 噪声生成功能。LFSR 的预装入值为 0xAAA。按照特定算法，在每次触发事件后 3 个 APB 时钟周期之后更新该寄存器的值。

设置 DAC_CR1 寄存器的 MAMP[3:0]位可以屏蔽部分或者全部 LFSR 的数据，这样得到的 LFSR 的值与 DACDHR 的数值相加，如果得到的数值有溢出，则将寄存器所能保留的最大值写入 DACDOR 中，如果得到的数值没有溢出，则将该值写入 DACDOR 中。

如果寄存器 LFSR 值为 0x000，则会注入‘1’(防锁定机制)。

将 WAVE[1:0]位置‘0’可以复位 LFSR 波形的生成算法。

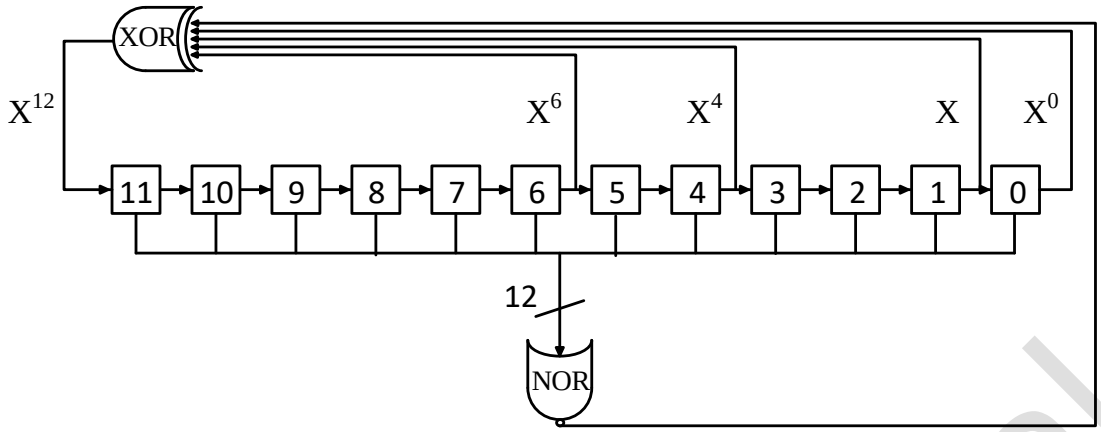


图 14-4 DAC LFSR 算法

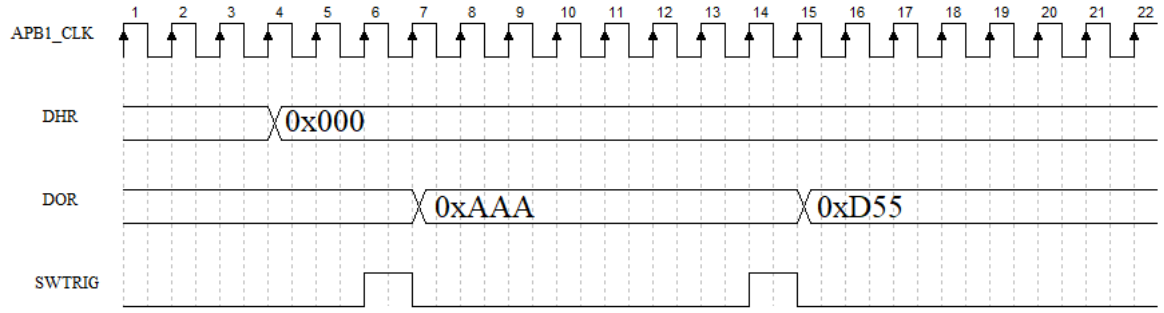


图 14-5 带 LFSR 波形生成的 DAC 转换(使能软件触发)

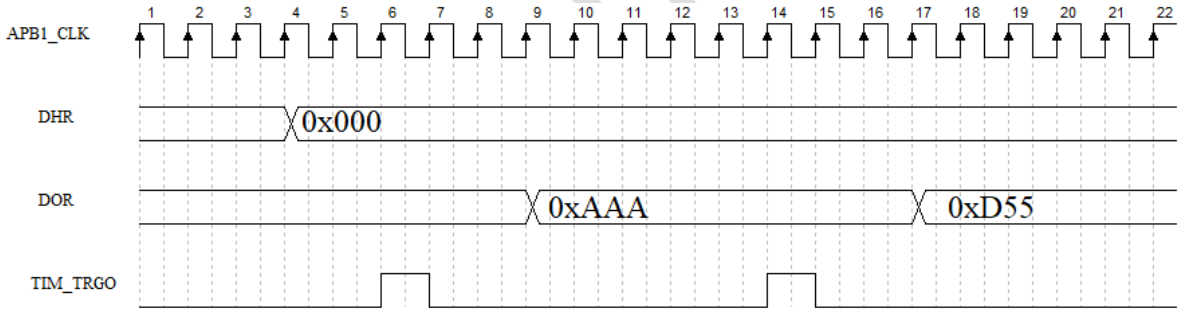


图 14-6 带 LFSR 波形生成的 DAC 转换(使能硬件触发)

注意：1. 为了产生噪声，必须使能 DAC 触发，即设 DAC_CR1 寄存器的 TEN 位为‘1’；

2. 若 DHR 的值发生了改变，则 LFSR 寄存器的值会复位为 0xAAA；

3. 若 TEN 从“1”降为“0”，则 LFSR 寄存器的值也会复位为 0xAAA。

14.3.11. DAC 三角波生成

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。设置 WAVE[1:0]位为“1x”选择 DAC 的三角波生成功能。设置 DAC_CR1 寄存器的 MAMP[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 3 个 APB 时钟周期后累加 1。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，周而复始。当设置的基值加上 MAMP 的最大幅度值大于 DACDOR 寄存器的最大值时，计数器则在 DACDOR 寄存器达到最大值时开始递减，递减到 DACDOR 达到基值后再开始累加，周而复始。

将 WAVE[1:0]位置‘0’可以复位三角波的生成。

注意：当基值为寄存器 DACDOR 的最大值时，不论 MAMP 为何值，DACDOR 的输出均为其最大值。

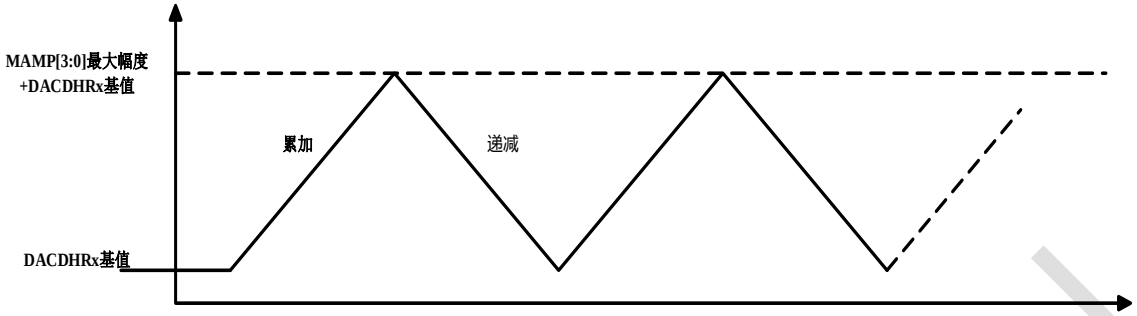


图 14-7 DAC 三角波生成

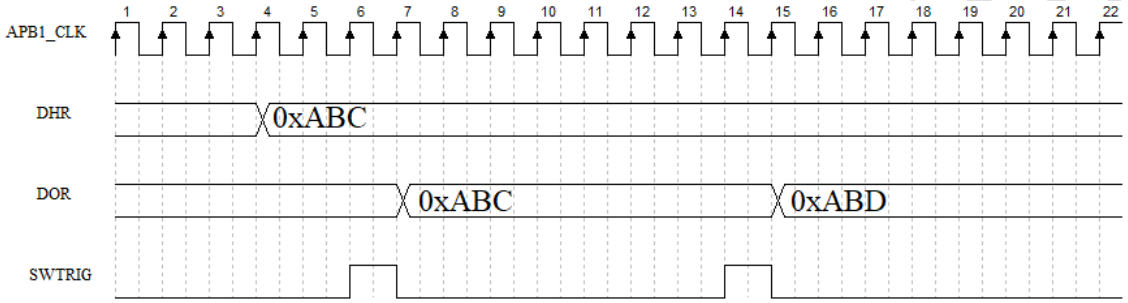


图 14-8 带三角生成的 DAC 转换(使能软件触发)

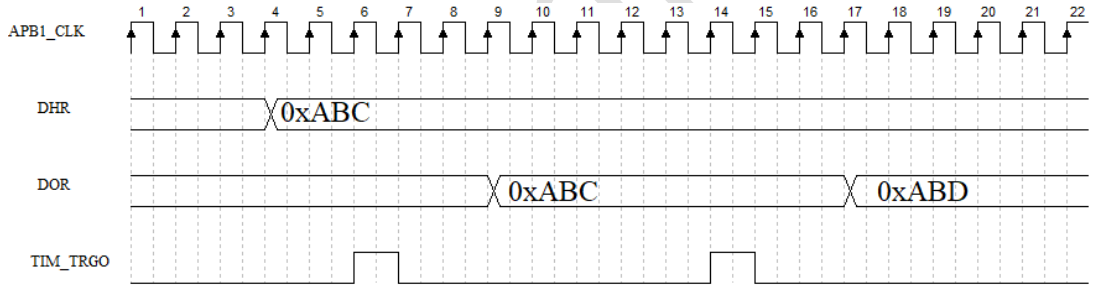


图 14-9 带三角生成的 DAC 转换(使能硬件触发)

注意：

1. 为了产生三角波，必须使能 DAC 触发，即设 DAC_CR1 寄存器的 TEN 位为‘1’；
2. MAMP[3:0]位必须在使能 DAC 之前设置，否则其值不能修改；
3. 若 DACDHR 的值发生了改变，三角波计数器复位为 0；
4. 若 TEN 从“1”降为“0”，三角波计数器也复位为 0。

14.3.12. DAC 输出时钟

DAC 输出时钟(DAC_CLK)为模拟端采样 DACDOR 数据时钟，该时钟只有在 EN 为“1”时有效。

在选择硬件触发时，检测到硬件触发上升沿后的 5 个 APB 周期置“1”，在保持 3 个 APB 周期后清 0；

在选择软件触发时，检测到软件触发上升沿后的 3 个 APB 周期置“1”，在保持 3 个 APB 周期后清 0；

在选择不触发时，检测到数据变化的 2 个 APB 周期置“1”，在保持 3 个 APB 周期后清 0。

注意：当 TEN 从“1”降为“0”后 1 个 APB 周期，DACDOR 的数据会更新为 DACDHR 寄存器中的数据。如果 DACDHR 寄存器中的数据有过更新，那么当 DACDOR 更新为 DACDHR 的值时，会产生一个 DAC_CLK；反之，则不会产生 DAC_CLK。

14.3.13. DAC 输出

DAC 结果可以外部输出至 IO 或内部比较器、运放或者 ADC。输出控制如下表所示：

表 14-2 DAC 输出

EN	BOFF	INTEREN	DACCEN	外部输出	内部输出	工作模式
1	1	1	0	高阻	BOFF 输出	内部 BOFF 输出
1	0	X	1	非 BOFF 输出	非 BOFF 输出	DAC 校准模式
1	0	X	0	非 BOFF 输出	高阻	外部非 BOFF 输出
1	1	0	0	BOFF 输出	高阻	外部 BOFF 输出
0	X	X	X	高阻	高阻	禁止

14.4. DAC 寄存器

14.4.1. DAC 控制寄存器 1 (DAC_CR1)

地址偏移：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUF_CAL	DACCEN	DMAUDRIE	DMAEN	MAMP[3:0]				WAVE[1:0]		TSEL[2:0]			TEN	BOFF	EN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	BUF_CAL	RW	0	DAC 内部校准使能 0：内部校准禁止 1：内部校准使能
14	DACCEN	RW	0	DAC 外部校准使能（通过 ADC）。 该位由软件设置和清除。 0：外部校准（通过 ADC）禁止； 1：外部校准（通过 ADC）使能；
13	DMAUDRIE	RW	0	DAC DMA 下溢中断使能。 该位由软件设置和清除。 0：DAC DMA 下溢中断禁止； 1：DAC DMA 下溢中断使能。
12	DMAEN	RW	0	DAC DMA 使能。 该位由软件设置和清除。 0：禁止 DAC DMA 模式； 1：使能 DAC DMA 模式。

11:8	MAMP[3:0]	RW	0	DAC 屏蔽/增幅选择器。 由软件设置这些位，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增幅值。 0000：不屏蔽 LSFR 位 0/三角波增幅值等于 1 0001：不屏蔽 LSFR 位[1:0]/三角波增幅值等于 3 0010：不屏蔽 LSFR 位[2:0]/三角波增幅值等于 7 0011：不屏蔽 LSFR 位[3:0]/三角波增幅值等于 15 0100：不屏蔽 LSFR 位[4:0]/三角波增幅值等于 31 0101：不屏蔽 LSFR 位[5:0]/三角波增幅值等于 63 0110：不屏蔽 LSFR 位[6:0]/三角波增幅值等于 127 0111：不屏蔽 LSFR 位[7:0]/三角波增幅值等于 255 1000：不屏蔽 LSFR 位[8:0]/三角波增幅值等于 511 1001：不屏蔽 LSFR 位[9:0]/三角波增幅值等于 1023 1010：不屏蔽 LSFR 位[10:0]/三角波增幅值等于 2047 ≥1011：不屏蔽 LSFR 位[11:0]/三角波增幅值等于 4095
7:6	WAVE[1:0]	RW	0	DAC 噪声/三角波生成使能。 该 2 位由软件设置和清除。 00:关闭波形发生器 01:使能噪声波形发生器 1x:使能三角波发生器
5:3	TSEL[2:0]	RW	0	DAC 触发选择。 该 3 位用于选择 DAC 的外部触发事件。 000:TIM6 TRGO 事件 001:TIM3 TRGO 事件 010: TIM7 TRGO 事件 011: TIM15 TRGO 事件 100: TIM2 TRGO 事件 101: 保留 110:外部中断线 9 111:软件触发 注意:该 3 位只能在 TEN = 1 (DAC 触发使能)时使用。
2	TEN	RW	0	DAC 触发使能。 该位由软件设置和清除,用来使能/关闭 DAC 的触发。 0：禁止 DAC 触发，写入 DACDHR 寄存器的数据在 1 个 APB 时钟周期后转入 DACDOR 寄存器; 1：使能 DAC 通道 1 触发，写入 DACDHR 寄存器的数据在 3 个 APB 时钟周期后转入 DACDOR 寄存器。 注意:如果选择软件触发，写入寄存器 DAC_DHR 的数据只需要一个 APB 时钟周期就可以传入寄存器 DACDOR。
1	BOFF	RW	0	关闭 DAC 输出缓存。 该位由软件设置和清除，用来使能/禁止 DAC 的输出缓存。 0:使能 DAC 输出缓存 1:禁止 DAC 输出缓存
0	EN	RW	0	DAC 使能。 该位由软件设置和清除，用来使能/禁止 DAC。 0:禁止 DAC 1:使能 DAC

14.4.2. DAC 控制寄存器 2 (DAC_CR2)

地址偏移：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTEREN	VREFSEL	SW TRIG
													RW	RW	W

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	INTEREN	RW	0	DAC 内部输出使能。 0：DAC 输出到外部 IO 1：DAC 输出到内部模拟模块
1	VREFSEL	RW	0	DAC 参考电压选择。 0：参考电压选择 V_{REFP}(无 V_{REFP} 时连接 V_{CCA}) 1：参考电压选择 V_{REFBUF}
0	SWTRIG	W	0	DAC 软件触发 该位由软件设置和清除，用来使能/禁止软件触发。 0：禁止 DAC 软件触发; 1：使能 DAC 软件触发。 注意:一旦寄存器 DACDHR 的数据传入寄存器 DACDOR，(1 个 APB 时钟周期后)该位由硬件置 0.

14.4.3. DAC 12 位右对齐数据保持寄存器(DAC_DHR12R)

地址偏移：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res.	Res.	Res.	Res.	DACDHR[11:0]										
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	DACDHR[11:0]	RW	0	DAC 的 12 位右对齐数据 该位由软件写入，表示 DAC 的 12 位数据。

14.4.4. DAC 12 位左对齐数据保持寄存器(DAC_DHR12L)

地址偏移：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACDHR[11:0]												Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:4	DACDHR[11:0]	RW	0	DAC 的 12 位左对齐数据。 该位由软件写入，表示 DAC 的 12 位数据。
3:0	Reserved			保留

14.4.5. DAC 8 位右对齐数据保持寄存器(DAC_DHR8R)

地址偏移：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACDHR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	DACDHR[7:0]	RW	0	DAC 的 8 位右对齐数据 该位由软件写入，表示 DAC 的 8 位数据。

14.4.6. DAC 数据输出寄存器(DAC_DOR)

地址偏移：0x2C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Res.	Res.	Res.	Res.	DACDOR[11:0]										
				R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	DACDOR[11:0]	R	0	DAC 输出数据. 该位只读，表示 DAC 的输出数据。

14.4.7. DAC 状态寄存器(DAC_SR)

地址偏移：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	DMAUDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
		RC_W1													

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	DMAUDR	RC_W1	0	DAC DMA 下溢标志。

				该位由硬件置位，软件写 1 清零。 0：未发生 DAC DMA 下溢 1：发生 DAC DMA 下溢 注意：只有在 DMAUDR 置“1”后，软件才能写“1”清“0”
12:0	Reserved	-	-	保留

14.4.8. DAC 校准寄存器(DAC_CALR)

地址偏移：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DAC_CR[4:0]				
											RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	DAC_DOUT	DAC_OS_TRIMCR_VREFBUF[5:0]						DAC_OS_TRIMCR_VCC[5:0]					
			R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20:16	DAC_CR[4:0]	RW	5’h0	DAC 预留功能寄存器
15:13	Reserved	-	-	保留
12	DAC_DOUT	R	1’b0	DAC 选择内部 Buffer 校准时的数据。
11:6	DAC_OS_TRIMCR_VREFBUF [5:0]	RW	6’h0	DAC 参考电压选择 V_{REFBUF} 时 offset 校准值。
5:0	DAC_OS_TRIMCR_VCC[5:0]	RW	6’h0	DAC 参考电压选择 V_{REFP}（V_{CCA}）时 offset 校准值。

15. 比较器(COMP)

15.1. COMP 简介

芯片内集成 2 个通用比较器（General purpose comparators，COMP），分别是 COMP1 和 COMP2。这两个模块可以作为单独的模块，也可以与定时器组合在一起使用。

用作比较器模拟功能的 INP/INM，必须在 GPIO 寄存器中被配置成模拟模式，且需要使能 SYSCFG.P*_ANA2EN 寄存器中对应的位。

比较器可以使用在：

- 在模拟信号的触发下从低功耗模式唤醒
- 模拟信号调节
- 与定时器的 PWM 输出连接时，构成逐周期电流控制回路

15.2. COMP 主要特性

- 每个比较器有可配置的正或者负输入，以实现灵活的电压选择。输入引脚来源有：
  - 复用 I/O 引脚
  - 温度传感器的输出
  - 支持内部参考电压 V_{REFBUF} 和 V_{CCA} 的 64 阶分压
  - V_{REFINT}
  - DAC 输出作为 INP 或者 INM 输入
  - OPA 输出作为 INP 输入
- 可编程迟滞
- 可编程速度和功耗
- 输出可以被连接到 I/O 或者定时器/低功耗定时器的输入作为触发：
  - OCREF_CLR 事件（逐周期电流控制）
  - 用于快速 PWM 关断的刹车
  - 定时器输入捕获
  - 低功耗定时器输入触发
- COMP1 和 COMP2 可以组合成窗口比较器
- 每个比较器具有中断产生能力，用于芯片从低功耗模式（睡眠和所有停止模式）的唤醒（通过 EXTI）
- 可配置数字滤波
- 消隐源比较器输出
- 配置寄存器写保护（LOCK 功能）

15.3. COMP 功能描述

15.3.1. COMP 框图

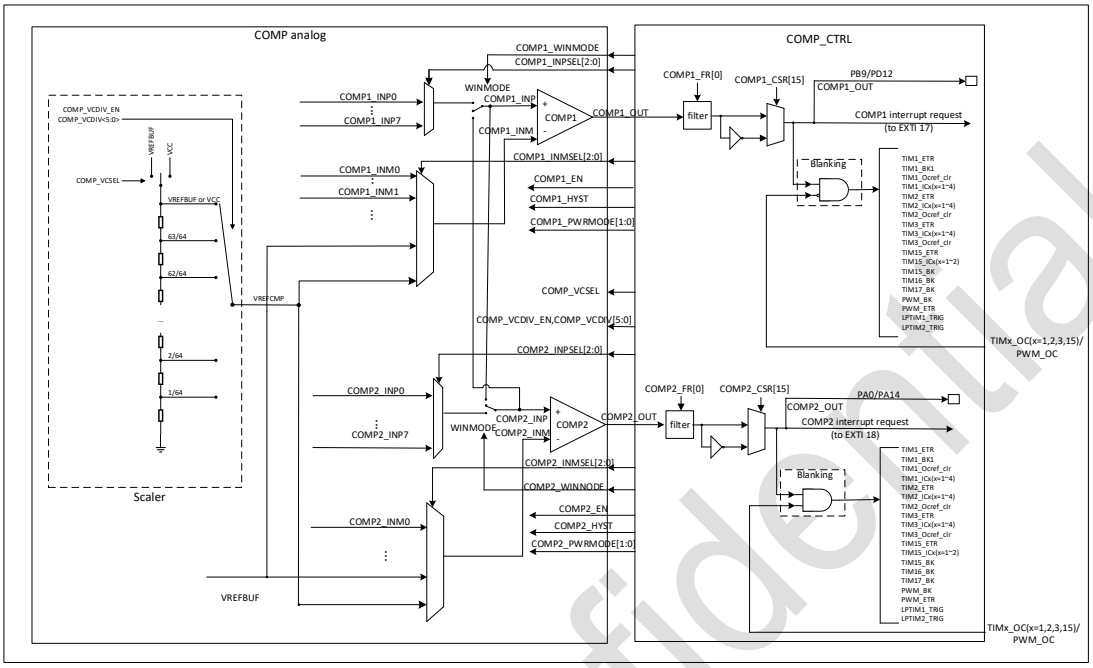


图 15-1 COMP 框图

15.3.2. COMP 接口和内部信号

用作比较器输入的 I/O，必须在 GPIO 寄存器中被配置成模拟模式，且需要使能 SYSCFG.P*_ANA2EN 寄存器中对应 PAD。

比较器输出可以通过在 GPIO 的可选功能（AF）连接到 I/O 引脚。

输出也可以在内部连接到各种定时器的输入，达到以下目的：

- 连接刹车输入，紧急关断 PWM 信号
- 使用 OCREF_CLR 输入的逐周期电流控制
- 定时器的输入捕获
- 低功耗定时器的输入触发

比较器 1 的非反相输入分配参见 COMP1_CSR 寄存器的 INPSEL[2:0]位域描述，反相输入分配参见 COMP1_CSR 寄存器的 INMSEL[2:0]位域描述。比较器 2 的非反相输入分配参见 COMP2_CSR 寄存器的 INPSEL[2:0]位域描述，反相输入分配参见 COMP2_CSR 寄存器的 INMSEL[2:0]位域描述。

15.3.3. COMP 复位和时钟

COMP 模块有如下时钟源：

PCLK（APB 时钟），系统总线时钟

COMP 时钟，用于模拟比较器输出后的电路（模拟输出的锁存电路、滤毛刺电路等）的时钟，可选择为 PCLK、LSE 或者 LSI。当需要在停止模式下工作时，需要选择 LSE 或者 LSI。

COMP 模块的复位信号源有：

1）APB 复位，系统复位

2）COMP 模块软复位（RCC_APBRSTR2.COMPxRST）

15.3.4. COMP 的滤波功能

如果芯片的工作环境恶劣，比较器的输出会出现噪声信号。通过置位 COMPx_FR.FLTENx 使能数字滤波模块，则比较器的输出波形中脉宽小于 COMPx_FR.FLTCNT[15:0]设定时间的噪声信号都可以被滤除。禁止数字滤波模块，则数字滤波模块的输出信号与输入信号相同。

注意：设置 COMP 滤波时间，开启滤波使能应在 COMPx_CSR.COMPx_EN 使能前完成。

滤波示意如下所示：

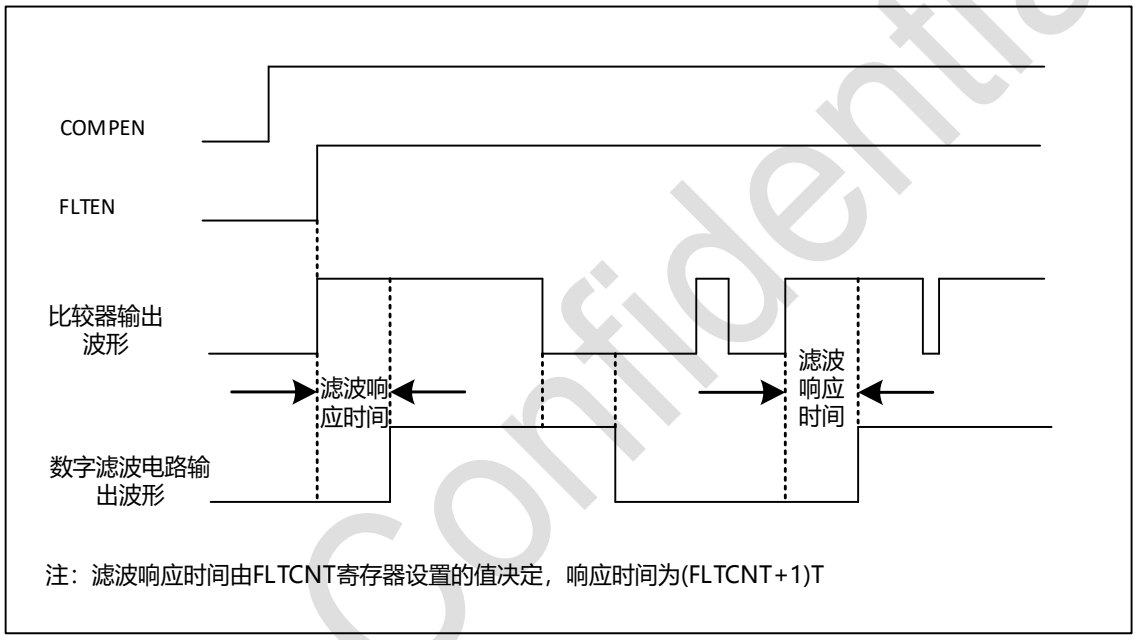


图 15-2 COMP 滤波示意图

15.3.5. COMP 输出消隐

消隐功能的目的是防止电流调节在 PWM 周期开始时因短电流尖峰而跳闸（通常是电源开关反并联二极管中的恢复电流）。这是通过设置一个由定时器输出比较信号定义的消隐窗口实现。每个比较器的通道消隐源由软件通过配置 COMPx_CSR 寄存器的 BLANKSEL[2:0]位单独选择。消隐信号的补码与比较器输出进行逻辑“与”运算，以产生比较器通道 x 输出。

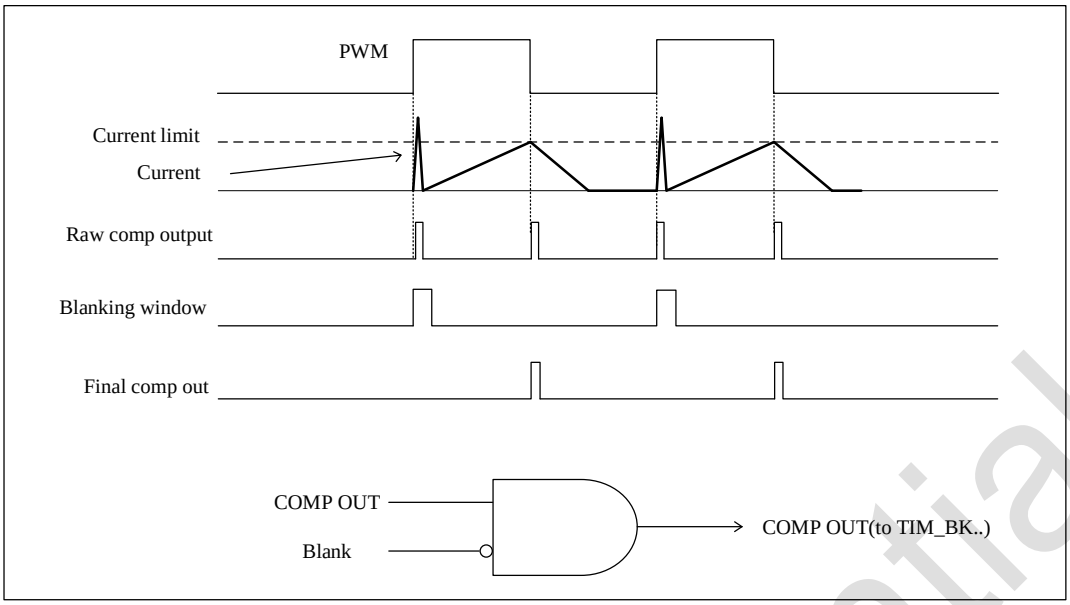


图 15-3 COMP 消隐

15.3.6. COMP 锁定机制

比较器可以用在安全的用途，例如过流和热保护。对于有特定功能性安全需求的应用，可对比较器配置进行保护，以防发生意外修改（例如，当程序计数器损坏时）。

为此，可以对比较器控制和状态寄存器进行写保护（只读）。

一旦比较器通道配置完成，其 LOCK 位设置为 1，这使得整个寄存器变成只读，包括 LOCK 位。

写保护仅能够被芯片的复位信号移除。

15.3.7. 窗口比较器

窗口比较器的作用是监测模拟电压是否在阈值上下限所定义的范围内。

可以使用两个比较器创建窗口比较器。被监测的模拟电压同时连接到两个比较器的非反相（+端）输入，阈值上下限电压分别连接到两个比较器的反相输入端（-端）。

通过使能 WINMODE 位，可以将两个比较器的非反相（+端）连接到一起，起到节省一个 I/O 引脚的作用。

例如当比较器 2 的 WINMODE 被置 1 时，比较器的正向输入端将接入 COMP1INP，当 WINMODE 被清 0 时，比较器的正向输入端将接入 COMP2INP。这样做可以省掉一个 IO 的 pin 脚。

注意：两个 COMP 的 WINMODE 模式不能同时使能。

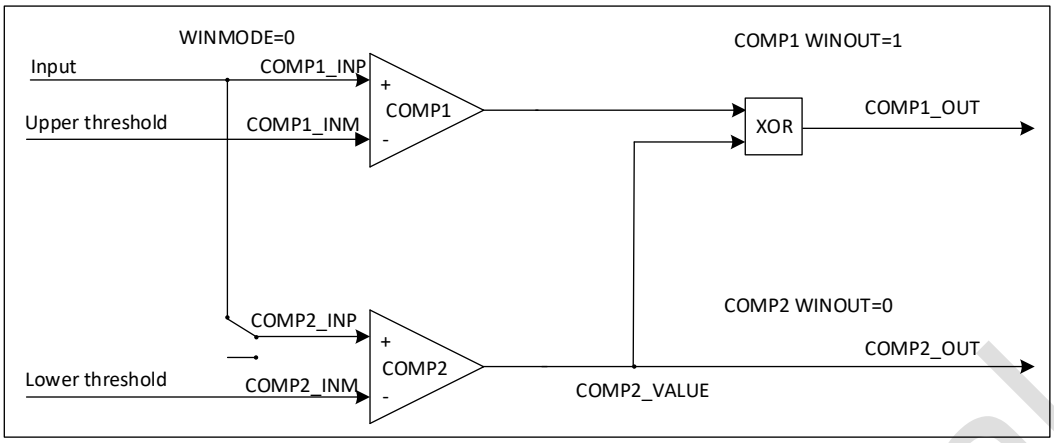


图 15-4 窗口比较器

15.3.8. 迟滞

为避免在噪声信号情况产生意外输出转换，比较器可以使能带有迟滞的功能（通过使能 COMP1_CSR 和 COMP2_CSR 的 HYST 位，可分别打开 COMP1 和 COMP2 的迟滞功能）。

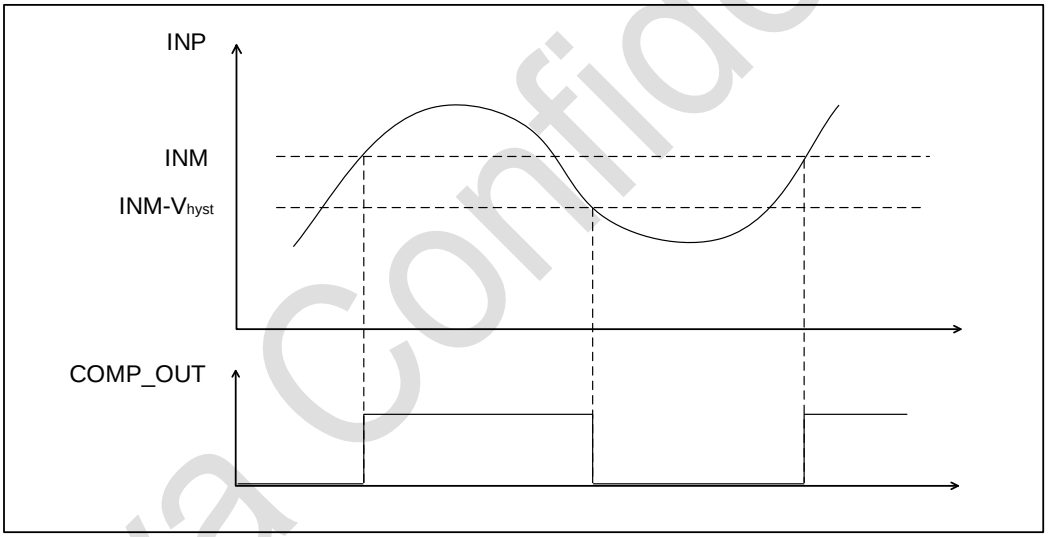


图 15-5 比较器迟滞

15.3.9. 低功耗模式

下表为各种低功耗模式下比较器的工作情况。在低功耗模式下，为了降低功耗，COMPx_CSR 寄存器的 PWRMODE 位需要配置为 1.

注意：INP 或者 INM 支持选择 V_{REFBUF}、V_{REFINT}、TS、DAC 或者 OPA 的输出，参考电源支持选择 V_{REFBUF} 时，会有较大功耗。在低功耗模式下，需要合理配置。

表 15-1 低功耗模式下的比较器特性

模式	说明
睡眠	对比较器无影响。 比较器中断可使器件退出睡眠模式。
低功耗运行	对比较器无影响。

低功耗睡眠	对比较器无影响。 比较器中断可使器件退出低功耗睡眠模式。
停止	对比较器无影响。 比较器中断可使器件退出所有停止模式。
待机	比较器控制器掉电，退出待机模式后必须重新初始化。

15.4. COMP 中断

比较器输出在芯片内部连接到 EXTI 控制器。每个比较器有单独的 EXTI Line（Line17，Line 18），能够产生中断或者事件。该机制还可用作从低功耗的唤醒。

要使能 COMPx 中断，需要遵循以下顺序:

1. 将对应于 COMPx 输出事件的 EXTI Line 配置为中断模式并将其使能，然后选择上升沿有效、下降沿有效或二者均有效
2. 配置并使能映射到相应 EXTI Line 的 NVIC IRQ 通道
3. 使能 COMPx

15.5. COMP 寄存器

配置 COMP1_CSR, COMP1_FR 和 COMP_CCSR 寄存器需要使能 RCC_APBENR1.COMP1EN 来开启 COMP1 的 APB 时钟。

配置 COMP2_CSR 和 COMP2_FR 寄存器需要使能 RCC_APBENR1.COMP2EN 来开启 COMP2 的 APB 时钟。

15.5.1. COMP1 控制和状态寄存器(COMP1_CSR)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP_OUT	INT_OUT_SEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRMODE	Res.	HYST
RW	R	RW											RW		RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL[2:0]			WINMODE	Res.	Res.	Res.	INPSEL[2:0]			INMSEL[2:0]			Res.	COMP1_EN
RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP1_CSR 寄存器锁定 软件置位，系统复位清零。当被置位，则会锁定 COMP1_CSR 寄存器的所有位。 0：未锁定，可读写整个寄存器 1：锁定，整个寄存器只读
30	COMP_OUT	R	0	COMP1 输出状态 该位只读，它反映了 COMP1 在经过极性选择和消隐后的输出电平。
29	INT_OUT_SEL	RW	0	比较器 1 内部数字输出控制。

				0：比较器内部数字输出选择模拟比较器直接输出（可以消隐），不经过滤波或数字同步 1：比较器内部数字输出选择与外部输出一致（可以滤波，可以消隐）。
28:19	Reserved	-	-	保留
18	PWRMODE	RW	0	COMP1 功耗模式选择 软件可读可写，选择了功耗和由此而来的 COMP1 的速度 0：高速 1：中速
17	Reserved	-	-	保留
16	HYST	RW	0	COMP1 迟滞功能使能控制 0：迟滞功能关闭 1：迟滞功能使能
15	POLARITY	RW	0	COMP1 极性选择 软件可读可写（如果没有被锁定） 0：不反相 1：反相
14:12	BLANKSEL[2:0]	RW	0	比较器 1 消隐信号选择 000: 消隐不使能 001: TIM1_OC4 010: TIM2_OC3 011: TIM3_OC3 100: TIM15_OC2 101: PWM_OC3 110: 保留 111: 保留
11	WINMODE	RW	0	窗口模式的 COMP1 非反相的输入选择 软件可读可写（如果没有被锁定） 0：通过 COMP1 的 INPSEL[2:0]选择 1：选择 COMP2 的 COMP2_INP 信号 注意两个 COMP 的 WINMODE 模式不能同时使能。
10:8	Reserved	-	-	保留
7:5	INPSEL[2:0]	RW	00	COMP1 非反相输入的信号选择 000：COMP1_INP0(OPA_VOUT) 001：COMP1_INP1(DAC_VOUT) 010：COMP1_INP2(PC15) 011：COMP1_INP3(PD0) 100：COMP1_INP4(PD1) 101：COMP1_INP5(保留) 110：COMP1_INP6(保留) 111：COMP1_INP7(保留)
4:2	INMSEL[2:0]	RW	00	COMP1 反相输入的信号选择 000：COMP1_INM0(PC5) 001：COMP1_INM1(PA1) 010：COMP1_INM2(DAC_VOUT) 011：COMP1_INM3(电阻分压输出 V_{REFCMP}) 100：COMP1_INM4(TS_VIN)(需要使能 ADC 模块的 ADC_CCR.TS_EN 寄存器)

				101：COMP1_INM5(VREFINT)（需要使能 ADC 模块的 ADC_CCR.VREFINT_EN 寄存器） 110：COMP1_INM6(V_{REFBUF})（需要使能 V_{REFBUF} 模块的 VREFBUF_EN 寄存器） 111：COMP1_INM7(PA13)
1	Reserved	-	-	保留
0	COMP1_EN	RW	0	COMP1 使能位 软件可读可写（如果没有被锁定） 0：禁止 1：使能

15.5.2. COMP1 滤波寄存器（COMP1_FR)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1[15:0]	RW	0x0	比较器 1 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved	-	-	保留
0	FLTEN1	RW	0x0	比较器 1 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注意: 该位必须在 COMP1_EN 为 0 时置位

15.5.3. COMP 共用控制和状态寄存器(COMP_CCSR)

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SSEL	VCDIV[5:0]						VREFCMP_EN
								RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	SSEL	RW	0	VREFCMP 电压源选择 0：V_{REFBUF} 1：V_{CCA}
6:1	VCDIV[5:0]	RW	0	COMP1,COMP2 分压选择 00_0000: 1/64 V_{REF}

				00_0001: 2/64 V_{REF} 00_0010: 3/64 V_{REF} … 11_1110: 63/64 V_{REF} 11_1111: V_{REF}
0	VREFCMP_EN	RW	0	V_{REFCMP} 使能 0：禁止 1：使能

15.5.4. COMP2 控制和状态寄存器(COMP2_CSR)

偏移地址:0x10

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	COMP_OUT	INT_OUT_SEL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRMODE	Res.	HYST
RW	R	RW											RW		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	BLANKSEL[2:0]			WINMODE	Res.	Res.	Res.	INPSEL[2:0]			INMSEL[2:0]			Res.	COMP2_EN
RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP2_CSR 寄存器锁定 软件置位，系统复位清零。当被置位，则会锁定 COMP2_CSR 寄存器的所有位。 0：未锁定，可读写整个寄存器 1：锁定，整个寄存器只读
30	COMP_OUT	R		COMP2 输出状态 该位只读，它反映了 COMP2 在经过极性选择的输出电平。
29	INT_OUT_SEL	RW	0	比较器 2 内部数字输出控制。 0：比较器内部数字输出选择模拟毕竟出去直接输出（可以消隐），不经过滤波或数字同步 1：比较器内部数字输出选择与外部输出一致（可以滤波，可以消隐）。
28:19	Reserved	-	-	保留
18	PWRMODE	RW	0	COMP2 功耗模式选择 软件可读可写，选择了功耗和由此而来的 COMP2 的速度 0：高速 1：中速
17	Reserved	-	-	保留
16	HYST	RW	0	COMP2 迟滞功能使能控制 0：迟滞功能关闭 1：迟滞功能使能
15	POLARITY	RW	0	COMP2 极性选择 软件可读可写（如果没有被锁定） 0：不反相 1：反相
14:12	BLANKSEL[2:0]	RW	0	比较器 2 消隐信号选择 000: 消隐不使能 001: TIM1_OC4

				010: TIM2_OC3 011: TIM3_OC3 100: TIM15_OC2 101: PWM_OC3 110: 保留 111: 保留
11	WINMODE	RW	0	窗口模式的 COMP2 非反相的输入选择 软件可读可写（如果没有被锁定） 0：通过 COMP2 的 INPSEL[2:0]选择 1：选择 COMP1 的 COMP1_INP 信号 注意两个 COMP 的 WINMODE 模式不能同时使能。
10:8	Reserved	-	-	保留
7:5	INPSEL[2:0]	RW	000	COMP2 非反相输入的信号选择，软件可读可写 000：COMP2_INP0(DAC_VOUT) 001：COMP2_INP1(OPA_VOUT) 010：COMP2_INP2(PB14) 011：COMP2_INP3(PC3) 100：COMP2_INP4(PC4) 101：COMP2_INP5(保留) 110：COMP2_INP6(保留) 111：COMP2_INP7(保留)
4:2	INMSEL[2:0]	RW	000	COMP2 反相输入的信号选择 000：COMP2_INM0(PA13) 001：COMP2_INM1(PB15) 010：COMP2_INM2(DAC_VOUT) 011：COMP2_INM3(电阻分压输出 V_{REFCMP}) 100：COMP2_INM4(TS_VIN)(需要使能 ADC 模块的 ADC_CCR.TS_EN 寄存器) 101：COMP2_INM5(V_{REFINT})(需要使能 ADC 模块的 ADC_CCR.VREFINT_EN 寄存器) 110：COMP2_INM6(V_{REFBUF})(需要使能 V_{REFBUF} 模块的 VREFBUF_EN 寄存器) 111：COMP2_INM7(PC0)
1	Reserved	-	-	保留
0	COMP2_EN	RW	0	COMP2 使能位 软件可读可写（如果没有被锁定） 0：禁止 1：使能

15.5.5. COMP2 滤波寄存器（COMP2_FR)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT2[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2[15:0]	RW	0x0	比较器 2 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。 采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved	-	-	保留
0	FLTEN2	RW	0x0	比较器 2 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注意: 该位必须在 COMP2_EN 为 0 时置位

16. 运算放大器(OPA)

16.1. OPA 简介

OPA 模块适用于简易放大器应用。内部的 1 个运放可以使用外部电阻进行级联。OPA 的输入范围是 0V 到 V_{CCA}，输出范围是 0.2 V 到 V_{CCA}-0.2 V。

用作 OPA 模拟功能的 I/O，必须在 GPIO 寄存器中被配置成模拟模式，且需要使能 SYSCFG.P*_ANA2EN 寄存器中 OPA 映射的 PAD。

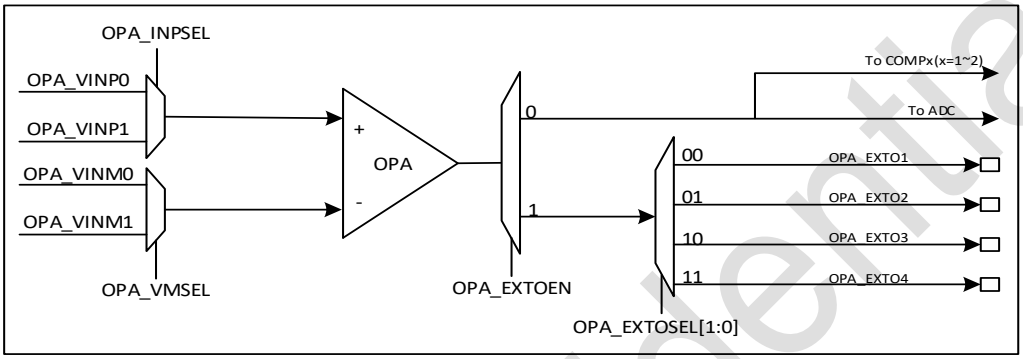


图 16-1 OPA 框图

16.2. OPA 主要特性

- 1 个独立配置运放
- V_{INP} 和 V_{INM} 可分别配置选择 2 路
- OPA 的输入范围是 到 V_{CCA}，输出范围是 0.2V 到 V_{CCA}-0.2V
- 可内部输出到 COMP1~2 以及 ADC，外部输出可配置选择 4 路 IO
- 仅可配置为通用运放模式模式

16.2.1. OPA 功能描述

OPA 可以通过使用外部元件组成放大器来放大小信号模拟输入信号，输出为放大后的信号。

16.3. OPA 寄存器

16.3.1. OPA 输出控制寄存器（OPA_OCR)

偏移地址:0x30

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPA_EXTOSEL[1:0]		OPA_EXTOEN
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:1	OPA_EXTOSEL[1:0]	RW	0	OPA PAD 外部输出选择 00：OPA 外部输出到 PD4 01：OPA 外部输出到 PD1 10：OPA 外部输出到 PC8 11：OPA 外部输出到 PC9
0	OPA_EXTOEN	RW	0	OPA 输出使能 0：内外部都不输出； 1：OPA 外部输出至 PAD，且内部输出至 COMP1~2 和 ADC； 注：当该寄存器配置为 1，且 OPA 输出至内部 COMP 或者 ADC 时，需要配置 SYSCFG.Px_ANA2EN 寄存器的对应位使能 ANA_PAD2（PC8/PC9/PD1/PD4）.

16.3.2. OPA 控制寄存器（OPA_CR）

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPAEN	Res.	Res.	OPA_INMSEL	Res.	OPA_INPSEL
										RW			RW		RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	OPAEN	RW	0	OPA 模块使能，高电平有效
4:3	Reserved	-	-	保留
2	OPA_VINMSEL	RW	0	OPA VINM 选择控制。 0：OPA VINM 选择 PD2 1：OPA VINM 选择 PC1
1	Reserved	-	-	保留
0	OPA_VINPSEL	RW	0	OPA VINP 选择控制。 0：OPA VINP 选择 PD3 1：OPA VINP 选择 PC2

17. 液晶显示控制器(LCD)

17.1. LCD 简介

LCD 控制器是一款适用于单色无源液晶显示器(LCD)的数字控制器/驱动器，最多具有 8 个公用端子（COM）和 40 个区段端子（SEG），用以驱动 160 (4x40)或 288 (8x36)个 LCD 图像元素。端子的确切数量取决于数据手册中所述的器件引脚。LCD 由若干区段（像素或完整符号）组成，这些区段均可点亮或熄灭。每个区段都包含一层在两根电极之间对齐的液晶分子。当向液晶施加高于阈值的电压时，相应的区段可见。区段电压必须为交流，以避免液晶中出现电涌效应（这将影响显示效果）。之后，必须在区段两端生成波形以避免出现直流。

17.2. LCD 主要特性

LCD 主要特性如下：

- 高度灵活的帧速率控制
- 支持静态、1/ 2、1/ 3、1/ 4、1/ 6 和 1/ 8 占空比
- 支持 1/ 2、1/ 3 和 1/4 偏置电压
- 可支持显示方式：Type A 或 Type B
- 3 种驱动波形生成方式：内部电阻分压、外部电阻分压，外部电容分压
  - 内部电阻分压可通过软件配置内部高驱动分压电阻导通时间的方式降低功耗，匹配 LCD 面板所需的电容电荷；也可通过软件配置 LCD 的对比度，调节 LCD 面板的亮度
  - 外部电阻分压可通过调节外部电阻阻值，匹配 LCD 面板所需的电容电荷
  - 外部电容分压可通过软件配置电容驱动次数，匹配 LCD 面板所需的电容电荷
- 支持 LCD 闪烁功能且可配置多种闪烁频率
- 通过可编程的帧间死区时间，调节显示亮度
- 未使用的 LCD 区段和公共引脚可配置为数字或模拟功能
- 双缓冲存储器，允许通过应用固件随时更新 LCD_RAM 寄存器中的数据，而不影响所显示数据的完整性
  - 多达 16x32 位寄存器的 LCD 数据 RAM
- 支持低功耗模式：LCD 控制器可在睡眠、低功耗运行、低功耗睡眠和停止模式下进行显示
- 可配置帧中断

17.3. LCD 功能描述

用作 LCD 的 I/O，必须在 GPIO 寄存器中被配置成模拟模式。

LCD 模块具体实现的功能如下：

- 可以选择外部低速晶振或内部低速 RC 时钟作为工作时钟(RCC 模块产生)
- 根据 LCD 时钟产生 COM 口开关波形；

- 根据当前的寄存器设置产生 SEG 开关的原始波形
- LCD RAM 用来存储 LCD 的显示数据，也可以用作普通数据寄存器使用
- LCD RAM 中的数据改写后，等待 UDD 置位后 LCD 显示的数据发生变化
- 产生支持模拟内部升压模式的控制时序

17.3.1. LCD 框图

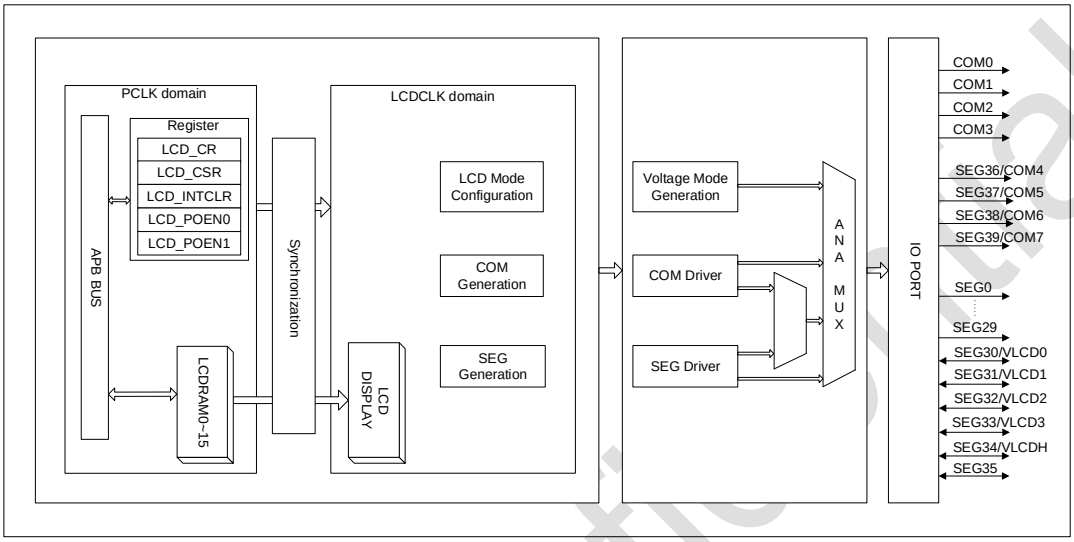


图 17-1 LCD 功能框图

17.3.2. LCD 时钟(刷新频率)

LCD_CR.LCDCLK 寄存器可以选择 LCD 刷新频率，即每帧传输的时间，有如下四个频率可选：

00: 64 Hz

01: 128 Hz

10: 256 Hz

11: 512 Hz

当时钟（外部低速晶振/内部低速 RC，32.768 kHz）输入后根据选择的频率进行分频.

17.3.3. LCD 扫描波形

以 1/8 Duty，1/3 Bias 为例，TypeA 和 TypeB 扫描波形如下所示。

8 个 COM 端会依次有效，在某个公共端有效的时间内，SEG 输出合适的电平，与 COM 电平共同施加在 LCD 面板上，压差大的段码将被显示，压差小的段码不显示。

1/3 Bias 表示 LCD 驱动电路能够输出 4 种驱动电平，对偏置电压平均分配得到。

17.3.3.1. LCD TypeA 扫描波形

下图仅画出了 3 个 COM 端。

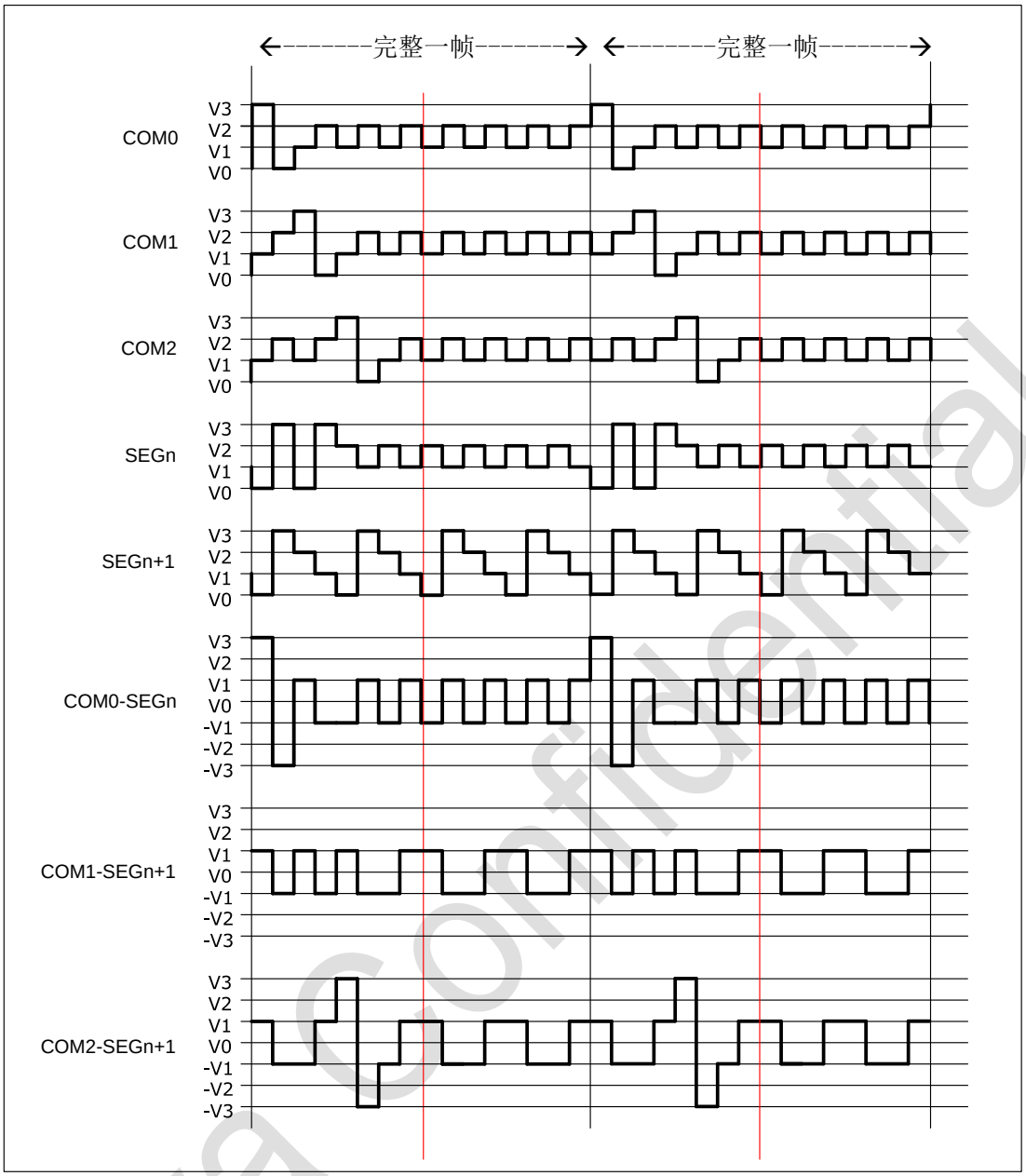


图 17-2 LCD 1/8 Duty, 1/3 Bias，TypeA 扫描波形

17.3.3.2. LCD TypeB 扫描波形

下图仅画出了 3 个 COM 端。

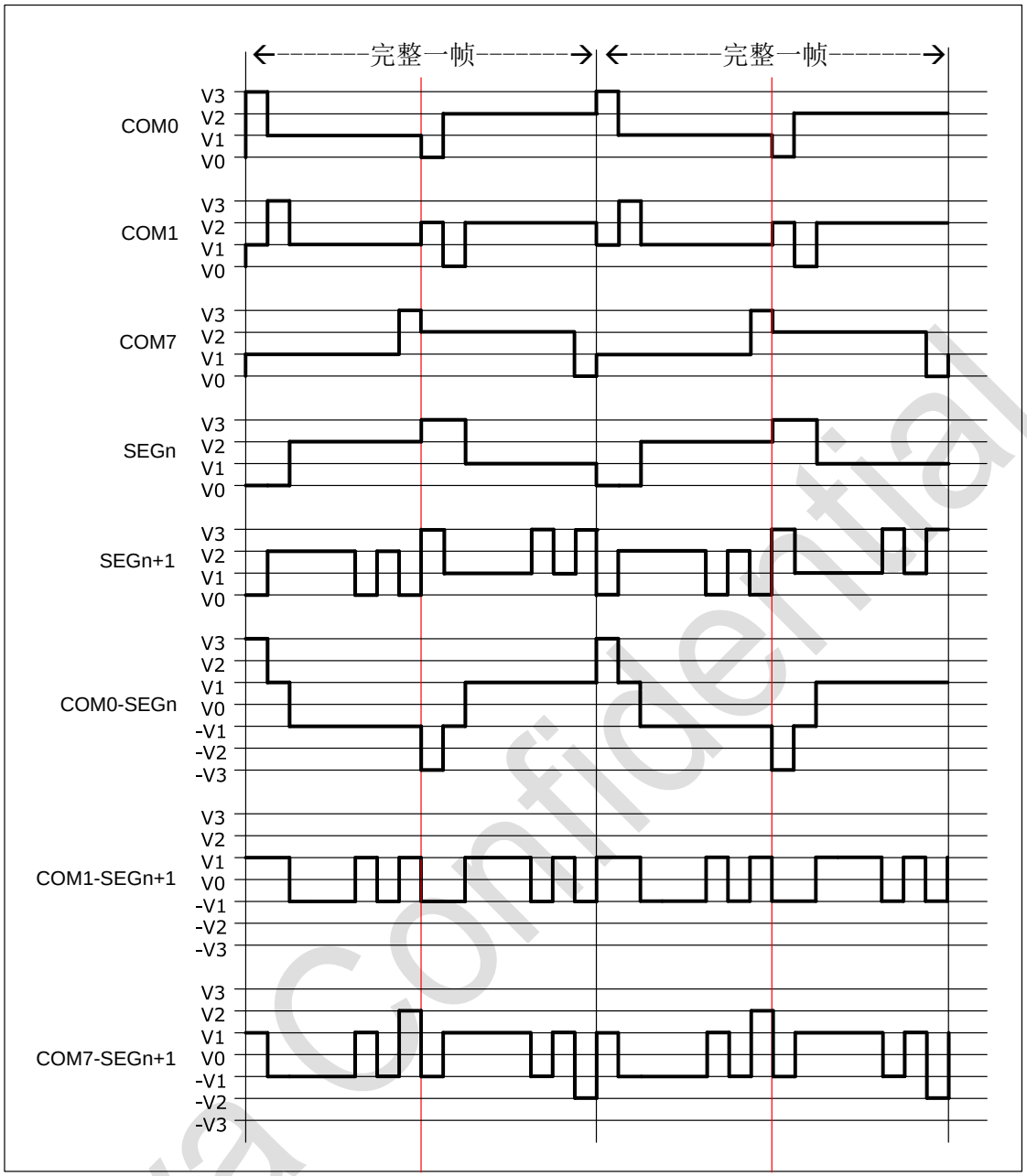


图 17-3 LCD 1/8 Duty, 1/3 Bias，TypeB 扫描波形

17.3.4. LCD 对比度控制

17.3.4.1. 波形幅度

可以通过 LCD_CR.CONTRAST 寄存器调整波形幅度的方式调整对比度。仅当偏置电压来源选择内部电阻分压时有效。

17.3.4.2. 死区

可以通过编程帧间死区来控制对比度。在死区内， COM 和 SEG 值置于 VSS。 LCD_CR 寄存器中的 DEAD[2:0] 位可用于编程一段最长为七个相位周期的时间。此死区可降低对比度，而无需修改帧速率。下图为 1/4 分压，DEAD[2:0]=3 时的 typeA 和 typeB 下的 COM0 的波形。

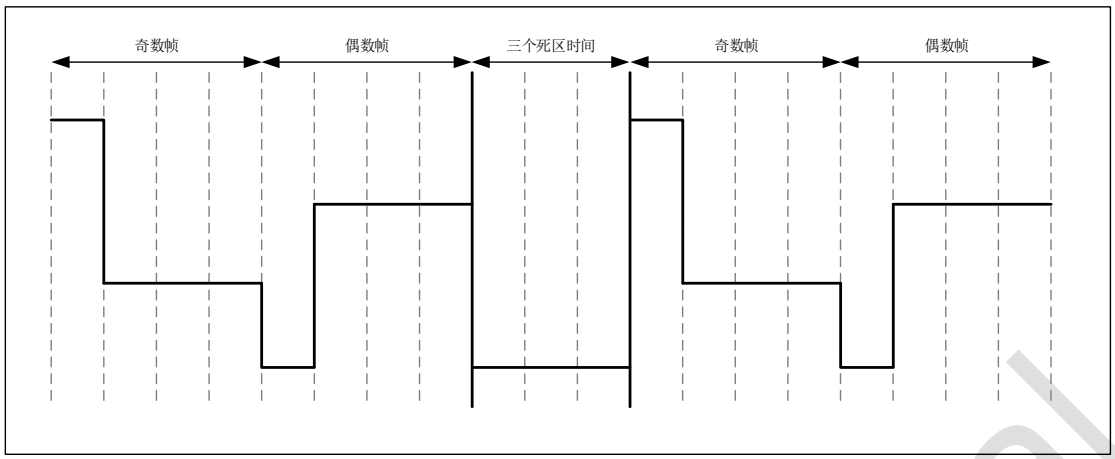


图 17-4 typeA 下的 COM0 的波形

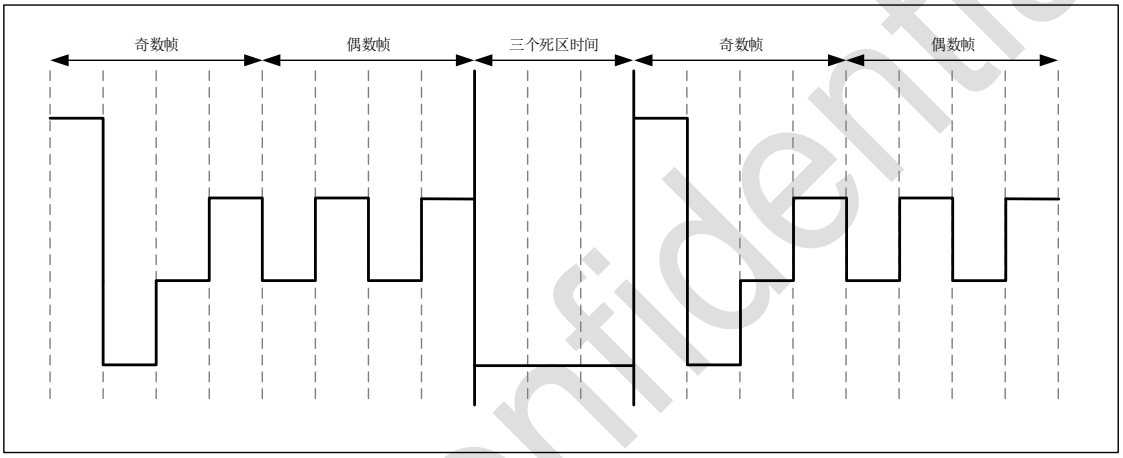


图 17-5 typeB 下的 COM0 的波形

17.3.5. LCD 显示数据存储模式

不同显示模式下， LCD 显示数据存储寄存器（LCDRAM）的可用空间如下：

17.3.5.1. 模式 1

- 1/8 占空比模式下，存储空间最大为 36×8 bits

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

- 1/6 占空比模式下，存储空间最大为 38×6 bits

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

■ 1/4 占空比模式下，存储空间最大为 40×4 bits

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

■ 1/3 占空比模式下，存储空间最大为 40×3 bits

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

■ 1/2 占空比模式下，存储空间最大为 40×2 bits

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

■ 静态显示模式下，存储空间最大为 40×1 bit

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7

	SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
COM0									LCDRAM8
COM1									LCDRAM9
COM2									LCDRAMA
COM3									LCDRAMB
COM4									LCDRAMC
COM5									LCDRAMD
COM6									LCDRAME
COM7									LCDRAMF

17.3.5.2. 模式 0

- 1/8 占空比

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

- 1/6 占空比

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

■ 1/4 占空比

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

■ 1/3 占空比

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

■ 1/2 占空比

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

■ **静态**

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	LCDRAM0
SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	LCDRAM1
SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	LCDRAM2
SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	LCDRAM3
SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	LCDRAM4
SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	LCDRAM5
SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	LCDRAM6
SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	LCDRAM7
																								SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	LCDRAM8
																								SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	LCDRAM9
																								SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	LCDRAMA
																								SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	LCDRAMB
																								SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	LCDRAMC
																								SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	LCDRAMD
																								SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	LCDRAME
																								SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	LCDRAMF

17.3.6. LCD 偏置电路

LCD 的偏置电压具有 3 种来源：内部电阻分压、外部电阻分压和外部电容分压。当选择内部电阻分压时，芯片会自动切换内部的电路以产生符合偏置和占空比的电压。当选择外部电阻分压或外部电容分压时，需要用户在芯片的外围引脚搭建相关电路。

17.3.6.1. 内部电阻模式

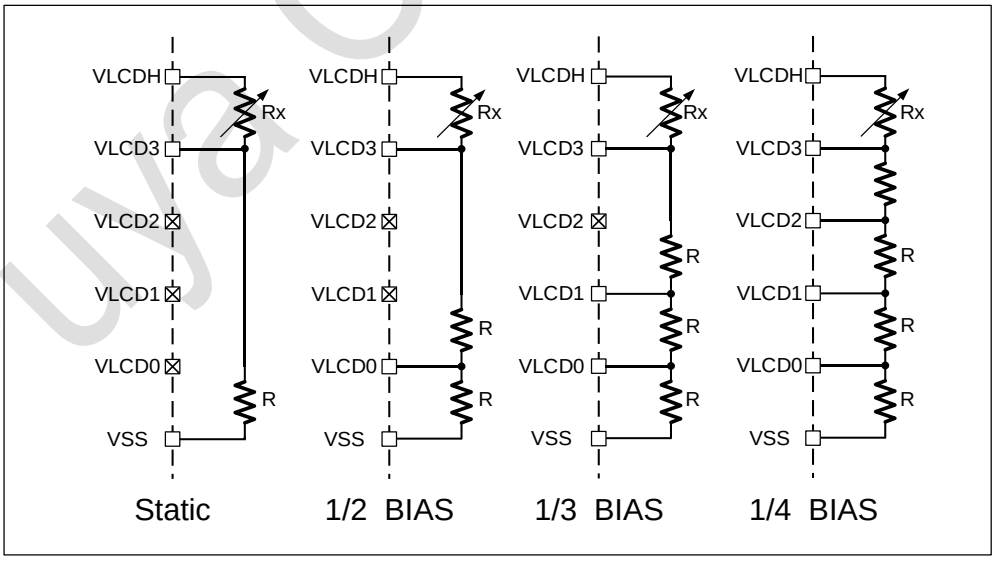
内部电阻模式下， VLCDH,VLCD3~VLCD0 可以作为 LCD SEG 输出或者 IO 端口使用。

内部电阻模式，LCD 的驱动电压由 LCD_CR.CONTRAST 控制， 如下表所示：

表 17-1 内部电阻模式

LCD_CR.CONTRAST	VLCD(ALL bias)
0	1.00*V_{CC}
1	0.92* V_{CC}
2	0.86* V_{CC}
3	0.8* V_{CC}
4	0.75* V_{CC}
5	0.71* V_{CC}
6	0.67* V_{CC}
7	0.63* V_{CC}
8	0.6* V_{CC}
9	0.57* V_{CC}
10	0.55* V_{CC}
11	0.52* V_{CC}
12	0.5* V_{CC}
13	0.48* V_{CC}
14	0.46* V_{CC}
15	0.44* V_{CC}

17.3.6.2. 外部电阻模式



当使用外部电阻驱动方式，需要外部电路支持，要额外占用 MCU 上的 VLCDH、VLCD3、VLCD2、VLCD1、VLCD0 五个 IO 管脚，用于外部电阻分压。

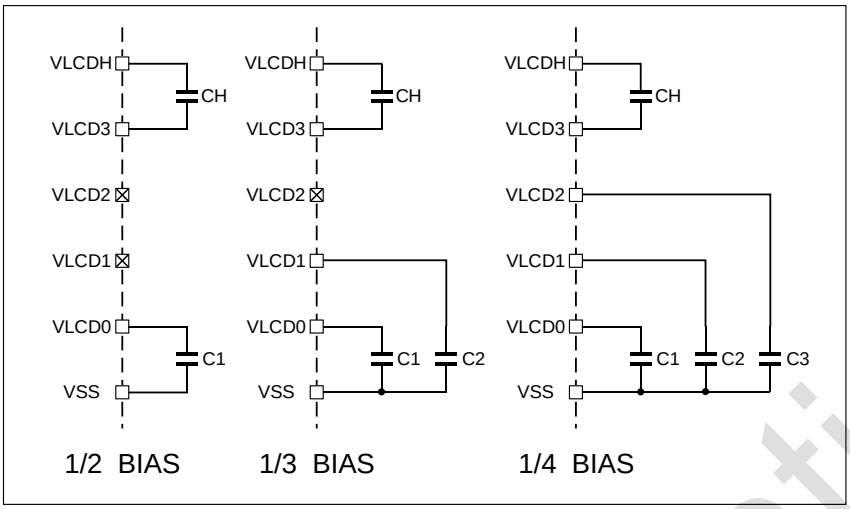
注意：

1. Rx 为可调电阻，用于调节 LCD 显示对比度。根据 LCD 屏幕选择合适电阻值。

2. 根据使用 LCD 屏幕选择合适的电阻 R，建议≥30KΩ

3. X 代表对应 VLCDX 通道不使用，可用做 LCD SEG 输出或 IO 端口使用

17.3.6.3. 外部电容模式



当使用外部电容驱动方式，需要外部电路支持，要额外占用 MCU 上的 VLCDH、VLCD3、VLCD2、VLCD1、VLCD0 五个 IO 管脚，用于外部电容分压。

注意：

1. 外部电容推荐 C1=C2=C3=CH=0.1μF
2. X 代表对应 VLCDX 通道不使用，可用做 LCD SEG 输出或 IO 端口使用

17.3.7. 双缓冲存储器

使用此双缓冲存储器，LCD 控制器可保证所显示信息连贯，而无需使用中断来控制 LCD_RAMx 修改。

应用软件可以通过 APB 接口访问第一缓冲级 (LCD_RAMx)。一旦修改了 LCD_RAMx，便会将 LCD_CSR 寄存器中的 UDR 标志置 1。该 UDR 标志（更新显示请求）会请求将更新的信息移到第二缓冲级 (LCD_DISPLAYx)。

在更新到 LCD_DISPLAYx（在当前帧结束后下一帧开始前）之前，LCD_RAMx 处于写保护状态并且 UDR 标志一直保持高电平状态。一旦更新完成，另一个标志 UDD（更新显示完成）便会置 1， 并产生一个中断（如果 LCD_CSR 寄存器中的 UDDIE 位置 1）。

17.4. LCD 中断

表 17-2 LCD 中断

中断事件	事件标志	使能控制
更新显示数据完成中断	UDD	UDDIE
帧起始中断	SOF	SOFIE
每次闪烁的“灭”起始中断	BKF	BKFIE

17.5. 软件配置流程

17.5.1. LCD 初始化流程

1. 首先设置时钟源 LCD 模块时钟，时钟源由 RCC_BDCR.LSCSEL 寄存器配置选择 LSI 或者 LSE。然后配置 RCC_APBENR2.LCDEN=1 使能 LCD 模块时钟。
2. 通过寄存器 LCD_CR.LCDCLK 设置 LCDCLK 扫描时钟频率
3. 通过寄存器 LCD_CSR.MODE 设置 LCD RAM 显示模式选择(RAM 与 COM/SEG 对应关系)，

4. 通过寄存器 LCD_CR.DEAD 与 LCD_CSR.BLINKEN/BLINKCNT 设置死区和闪烁频率
5. 通过寄存器 LCD_CR.COMSEG_DRV_TYPE 设置 LCD 输出波形选择 TypeA 或者 TypeB
6. 选择 VCCA 电源模式（默认）
7. 通过寄存器 LCD_CR.DUTY 选择 LCD 波形占空比
8. 如果是非静态模式，通过寄存器 LCD_CR.BIAS 设置偏压模式，之后再通过寄存器 LCD_CR.BSEL 选择偏置电压模式
9. 如果是非静态模式，仅当偏置电压选择内部电阻分压时，可以设置对比度(LCD_CR.CONTRAST)和内部电阻的快读充电模式的充电时间(LCD_CR.FCCTL)
10. 如果是非静态模式，仅当偏置电压选择外部电容时，可以通过寄存器 LCD_CSR.SWCAP_DRV_NUM 设置一个 LCDCLK 周期内的外部电容驱动次数
11. 清除 LCD_CR.EN=0 并且 LCD_CSR.SOF=0
12. 清除 LCD_RAM0~15 数据，将其全写为 0
13. 设置 LCD_CSR.UDR 为 1
14. 使能 LCD_CR.EN=1
15. 等待 LCD_CSR.ENS 为 1
16. 等待硬件配置 LCD_CSR.SOF 为 1(代表第一帧数据写入)
17. 清除 LCD_CSR.SOF 为 0
18. 初始化配置结束。

17.5.2. LCD 数据更新

LCD RAM 的数据更新频率必须小于帧频率。此外只要写入 LCD_RAM 后将 LCD_CSR.UDR 置 1，LCD 显示信息就可自动在当前帧结束后更新，下一帧便可显示新数据。流程如下：

1. 等待 LCD_CSR.UDR 为 0
2. 显示新数据写入 LCD_RAM0~15
3. 清除 LCD_CSR.UDD 为 0
4. 设置 LCD_CSR.UDR 为 1
5. 等待 LCD_CSR.UDD 为 1
6. 再次写入数据则返回第 1 步

17.5.3. 关闭 LCD 流程

1. 设置 LCD_CR.EN 为 0
2. 等待 LCD_CSR.ENS 为 0
3. 清除 LCD_CSR.SOF 为 0

17.6. LCD 寄存器

17.6.1. LCD 控制寄存器 (LCD_CR)

偏移地址: 0x00

复位值: 0x0086_00C2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	DEAD[2:0]			FCCTL[2:0]			COMSEG_DRV_TYPE	Res.	Res.	Res.	Res.
					RW	RW	RW	RW	RW	RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONTRAST[3:0]				Res.	BSEL[1:0]		DUTY[3:0]			BIAS[1:0]		Res.	LCD_CLK[1:0]		EN
RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	保留
26:24	DEAD[2:0]	RW	3’b000	死区持续时间 这些位通过软件写入，用于配置帧间死区的长度。在死区内， COM 和 SEG 电压电平保持为 0 V 以降低对比度，而无需修改帧速率。 000：无死区 001： 1 个相位周期死区 010： 2 个相位周期死区 ..... 111： 7 个相位周期死区 此寄存器在 EN=0 的情况下更新时，更新频率应小于等于最慢时钟（LCDCLK/PCLK）的 5 倍，即为两次更新间隔必须大于等于 5 个慢时钟周期。 此寄存器在 EN=1 的情况下也可以更新，但是更新频率应小于帧的频率，并且此寄存器总是在下一帧开始才应用新值。
23:21	FCCTL[2:0]	RW	3’b100	快速充电模式，充电时间控制位。（1 LCD COM 周期=1/2 LCDCLK） 000：1/2 LCD COM 周期 001：1/4 LCD COM 周期 010：1/8 LCD COM 周期 011：1/16 LCD COM 周期 100：1/32 LCD COM 周期 101：1/64 LCD COM 周期 110：1/128 LCD COM 周期 111：快速充电模式无效 注：该寄存器针对内部电阻模式，其他模式该寄存器配置无效。 此寄存器只可在 EN=0 时更新。
20	COMSEG_DRV_TYPE	RW	1’b0	LCD 波形类型选择 0：TypeA 1：TypeB 此寄存器只可在 EN=0 时更新。
19:16	Reserved	-	-	保留
15:12	CONTRAST[3:0]	RW	4’b0	LCD 对比度调整 CONSTRAST 值越大，LCD 波形的幅度越小。 0x0：LCD 波形幅度最大，对比度最大； …… 0xF：LCD 波形幅度最小，对比度最小； 注：仅当偏置电压来源选择内部电阻分压时有效。
11	Reserved	-	-	保留
10:9	BSEL	RW	2’b0	偏置电压模式选择 00：内部电阻驱动模式 01：外部电阻驱动模式 10：外部电容驱动模式 11：保留（功能按照内部电阻驱动模式）

8:6	DUTY	RW	3'b011	LCD 占空比选择 000: 静态 001: 1/2 占空比 010: 1/3 占空比 011: 1/4 占空比 100: 保留 101: 1/6 占空比 110: 保留 111: 1/8 占空比
5:4	BIAS	RW	2'b0	00：1/4 偏压 01：1/2 偏压 10：1/3 偏压 11：保留（功能按照 1/4 偏压）
3	Reserved	-	-	保留
2:1	LCDCLK	RW	2'b01	LCD 扫描频率选择 00: 64Hz 01: 128Hz 10: 256Hz 11: 512Hz 注：LCD 帧频率 = LCD 扫描频率×Duty 此寄存器只可在 EN=0 时更新。
0	EN	RW	1'b0	LCD 使能控制 1：使能 0：禁止 使能 EN 时，COMSEG_DRV_TYPE、BIAS、BSEL、DUTY 等寄存器处于写保护状态

17.6.2. LCD 控制状态寄存器 (LCD_CSR)

偏移地址: 0x04

复位值: 0x0000_0080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	ENS	UDD	SOF	BKF
												R	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDR	UDDIE	SOFIE	BKFIE	Res.	MODE	SWCAP_DRV_NUM[2:0]			BLINKE	BLINKCNT[5:0]					
RS	RW	RW	R		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19	ENS	R	0	LCD 使能状态。 此位通过硬件置 1 和清零。它指示 LCD 控制器状态。 0：禁止 LCD 控制器 1：使能 LCD 控制器 注：当 LCD_CR 寄存器中的 EN 位从 0 变为 1 时，ENS 位立即置 1。禁用时，此位反映 LCD 的实际状态，因此在最后显示的帧结束时变为 0。
18	UDD	R	0	更新显示完成 此位由硬件置 1，通过向 LCD_INTCLR 寄存器中的 UDDC 写入 1 进行清零。位置 1 的优先级高于位清零。 0：无事件

				1：更新显示请求完成。 如果 LCD_CSR 寄存器中的 UDDIE 位置 1，将产生 UDD 中断。 注： 如果器件处于停止模式（未提供 PCLK），即使 UDDIE = 1， UDD 也不会产生中断。 如果显示未使能，将永远不会发生 UDD 中断。
17	SOF	R	0	此位在新的帧开始时由硬件置 1，同时更新显示数据。此位通过向 LCD_INTCLR 寄存器中的 SOFC 位写入 1 进行清零。位清零的优先级高于位置 1。 0：无事件 1：发生帧起始事件。如果 SOFIE 位置 1，则产生 LCD 帧起始中断。
16	BKF	R	0	LCD 闪烁数据完成中断标志 0：无 LCD 闪烁数据完成中断 1：LCD 闪烁数据完成中断产生
15	UDR	RS	0	更新显示请求 每次软件修改 LCD_RAMx 时，必须将 UDR 位置 1 以将更新的数据传送到第二级缓冲器（LCD_DISPLAYx）。UDR 位在更新结束之前保持置 1，在此期间 LCD_RAMx 处于写保护状态。 0：无影响 1：更新显示请求 当使能显示时，仅针对激活（取决于占空比配置）的存储单元执行更新。 注：当此位已为 1 时，写入 0 或写入 1 都无影响。此位只能由硬件清零。仅当 LCD_CR.EN=1 时，此位才能清零。 EN 禁止时，经过 3~4 个慢时钟完成 LCD_DISPLAY 的更新； EN 使能时，在帧开始时完成更新 LCD_DISPLAY，并在更新完成时对 UDR 清零。
14	UDDIE	RW	0	更新显示完成中断使能 此位由软件置 1 和清零。 0：禁止 LCD 更新显示完成中断 1：使能 LCD 更新显示完成中断
13	SOFIE	RW	0	帧起始中断使能 此位由软件置位和清零。 0：禁止帧起始中断 1：使能帧起始中断
12	BKFIE	RW	0	显示屏完成一次中断使能 此位由软件置位和清零。 0：禁止闪烁数据完成中断 1：使能闪烁数据完成中断
11	Reserved	-	-	保留
10	MODE	RW	0	LCD RAM 显示模式选择 0: 模式 0 1: 模式 1
9:7	SWCAP_DRV_NUM	RW	3’b001	外部电容模式时，在一个 LCDCLK 内驱动外部电容的次数，驱动次数越多负载越强。 000: 4

				001: 8 010: 16 011: 32 100: 64 其它:保留 注：该寄存器针对外部电容模式，其他模式该寄存器配置无效。 此寄存器只可在 EN=0 时更新。
6	BLINKEN	RW	0	LCD 闪屏使能 0：禁止 1：使能
5:0	BLINKCNT	RW	0	闪屏频率与 LCD 中断间隔设置 LCD 闪烁频率 = LCD 帧频率 / (BlinkCnt+1)； LCD 中断间隔= (BlinkCnt+1)*(1/LCD 帧频率)。 不同占空比下所需要显示的 COM 口也不同，此 BLINKCNT 是占空比下所有的 COM 口显示的次数，例如 1/8 占空比，BLINKCNT=6'd31,则 8 个 COM 口的数据一共要刷新 32 次后才会起 BKF 中断。 此寄存器在 EN=0 的情况下更新时，更新此寄存器的的频率应小于等于最慢时钟（LCD_CLK/PCLK）的 5 倍，即为两次更新间隔必须大于等于 5 个慢时钟周期。 此寄存器在 EN=1 的情况下也可以更新，但是更新频率应小于帧的频率，并且此寄存器总是在下一帧才开始应用新值，并且闪烁的帧计数会从零开始计数。 注 1：LCD 未使能时，第一次更新该寄存器后，在 5 个 LSI/LSE 时钟周期内会屏蔽第 2 次更新； 注 2：LCD 使能后，第一次更新该寄存器后，在当前帧期间会屏蔽第 2 次更新；且更新值在新帧才生效；

17.6.3. LCD 中断清零寄存器（LCD_INTCLR）

偏移地址: 0x08

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDDC	SOFC	BKFC
													W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18	UDDC	W	0	UDDF 中断标志清除，写 1 清除，写 0 无效
17	SOFC	W	0	SOF 中断标志清除，写 1 清除，写 0 无效
16	BKFC	W	0	BKF 中断标志清除，写 1 清除，写 0 无效
15:0	Reserved	-	-	保留

17.6.4. LCD 输出配置寄存器 0（LCD_POEN0）

偏移地址: 0x0C

复位值: 0xFFFF_FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Sx	RW	0xFFFFFFFF	SEG31~0 输出控制位 0: SEG31~0 输出使能 1: SEG31~0 输出关闭，可以使用其他功能

17.6.5. LCD 输出配置寄存器 1（LCD_POEN1）

偏移地址: 0x10

复位值: 0x0000_0FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	C3	C2	C1	C0	S39C7	S38C6	S37C5	S36C4	S35	S34	S33	S32
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:8	Cx	RW	0xF	COMx 输出控制位（COM3~COM0） 0: COM 输出使能 1: COM 输出关闭
7:4	SxCy	RW	0xF	SEGx/COMy 输出控制位 0: SEG39~36/COM7~4 输出使能 1: SEG39~36/COM7~4 输出关闭 SEG/COM 引脚功能选择由 LCD_CR.DUTY 决定
3:0	Sx	RW	0xF	SEG35~32 输出控制位 0: SEG 输出使能 1: SEG 输出关闭

17.6.6. LCD_RAM0~7

偏移地址: 0x14-0x30

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Dx	RW	0	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG/COM 交叉点不亮; 1 对应的 SEG/COM 交叉点亮;

17.6.7. LCD_RAM8~15

偏移地址: 0x34~0x50

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	D7	D6	D5	D4	D3	D2	D1	D0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	Dx	RW	0	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG/COM 交叉点不亮; 1 对应的 SEG/COM 交叉点亮;

18. 高级定时器(TIM1)

18.1. TIM1 简介

高级控制定时器（TIM1）包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

此类定时器可用于多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者生成输出波形（输出比较、PWM 和带死区插入的互补 PWM）。

使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度和波形周期从几微秒调制到几毫秒。

高级控制定时器（TIM1）和通用定时器（TIMy）彼此完全独立，不共享任何资源。它们可以一起同步操作。

18.2. TIM1 主要特性

- 16 位递增、递减、递增/递减自动重载计数器
- 16 位可编程预分频器，计数器时钟频率的分频系数为 1 ~ 65535 之间的任意数值
- 多达 4 个独立的通道
  - 输入捕获
  - 输出比较
  - PWM 产生（边沿或者中心对齐模式）
  - 单脉冲模式输出
  - 可重触发的单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路
- 重复计数器，在计数指定周期数后，才更新定时器的寄存器
- 刹车输入可以将定时器的输出信号置为用户可选的安全配置中
- 发生如下事件时生成中断/DMA 请求：
  - 更新：计数器向上、向下溢出，计数器初始化（通过软件或者内外部触发）
  - 触发事件
  - 输入捕获
  - 输出比较
  - 刹车输入
- 支持用于定位的增量（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逐周期的电流管理

18.3. TIM1 功能描述

18.3.1. 功能框图

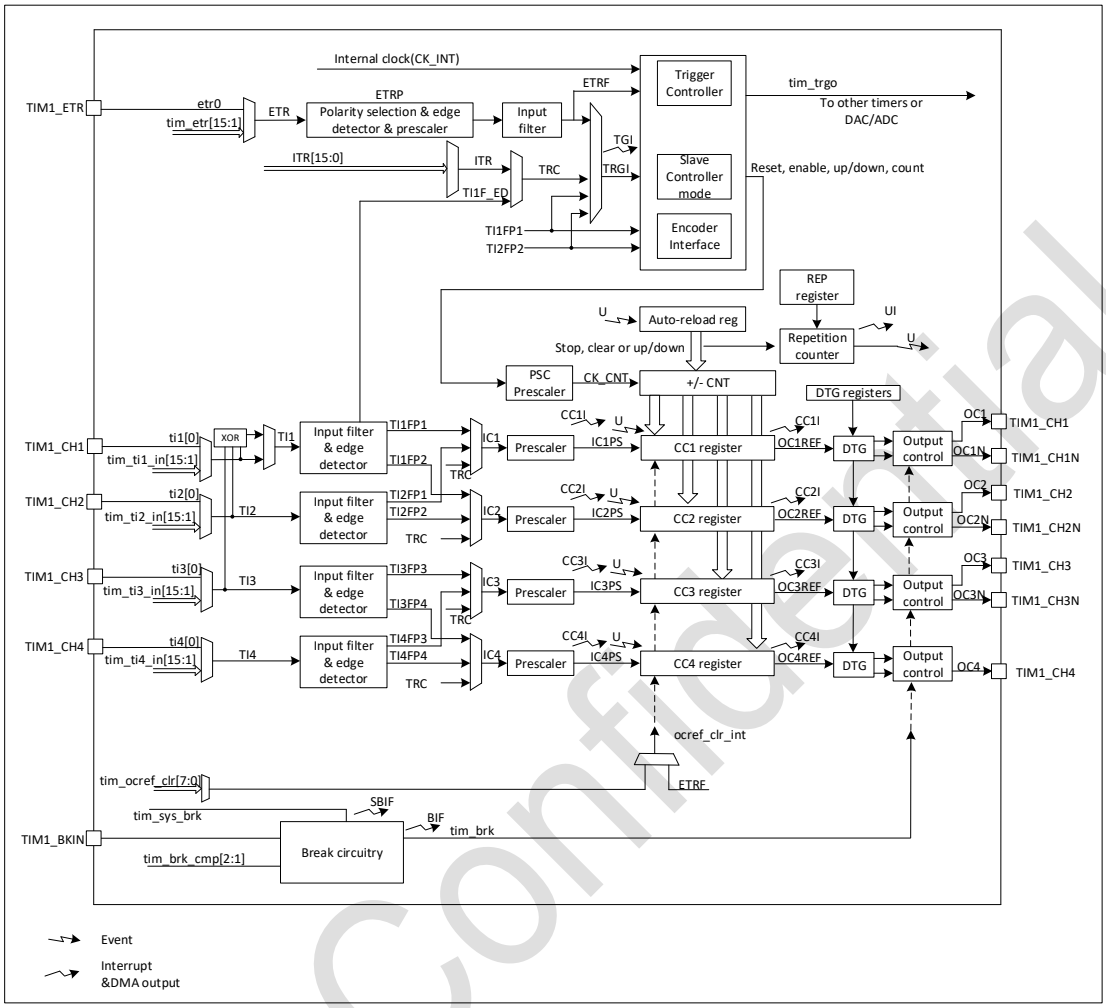


图 18-1 高级控制定时器框图

18.3.2. 时基单元

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。计数器可递增计数、递减计数或交替进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

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。

时基单元包括：

- 计数器寄存器（TIM1_CNT）
- 预分频寄存器（TIM1_PSC）
- 自动装载寄存器（TIM1_ARR）
- 重复计数寄存器（TIM1_RCR）

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存器，这取决于 TIM1_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIM1_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIM1_CR1 寄存器中的计数器启动位(CEN)置 1 时，才会启动计数器。

注意，计数器将在 TIM1_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。

18.3.2.1. 预分频描述

预分频器可以将计数器的时钟按 1 到 65536 之间的任意值分频。它是基于一个（在 TIM1_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。

下面两个图给出了在预分频器运行时，更改计数器参数的例子。

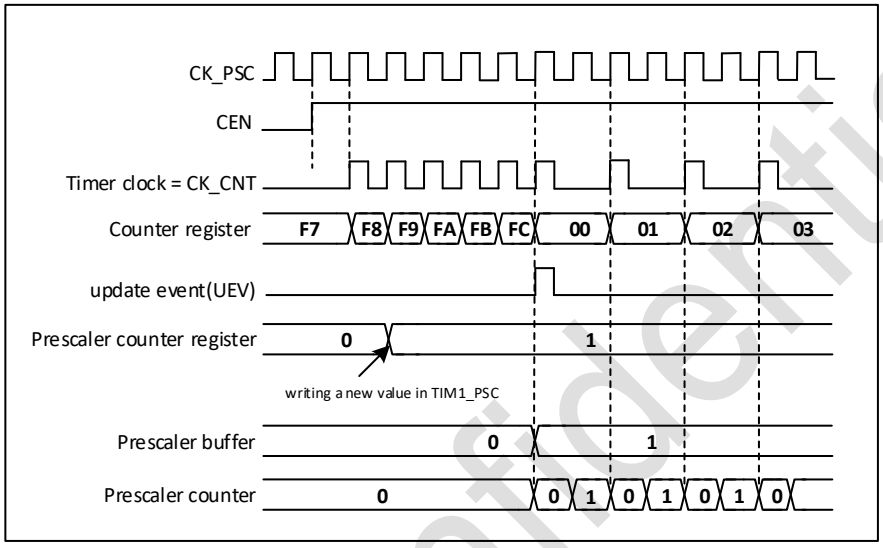


图 18-2 预分频器分频由 1 变为 2 时的计数器时序图

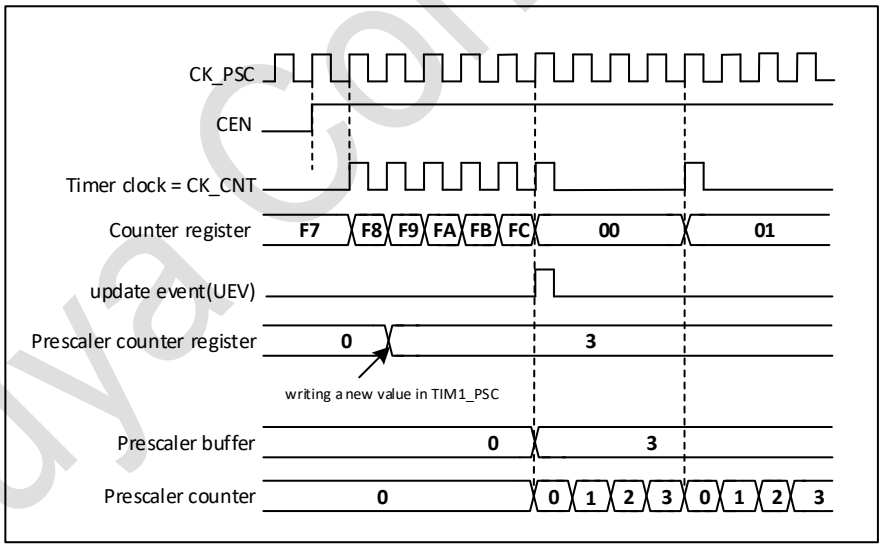


图 18-3 预分频器分频由 1 变为 4 时的计数器时序图

18.3.3. 计数模式

18.3.3.1. 递增计数模式

递增计数模式，是从 0 计数到自动装载值（TIM1_ARR 寄存器），然后又从 0 重新开始计数，并产生一个计数的溢出事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中设定的次数加一次（(TIM1_RCR) +1）后，将生成更新事件(UEV)。否则，将在每次计数器上溢时产生更新事件。

在 TIM1_EGR 寄存器中设置 UG 位(通过软件方式或者使用从模式控制器)也同样可以产生一个更新事件。

设置 TIM1_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIM1_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIM1_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）。

- 重复计数器被重新加载为 TIM1_RCR 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值(TIM1_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIM1_PSC 寄存器的内容)。

下图给出一些例子，当 TIM1_ARR=0x36 时计数器在不同时钟频率下的动作。

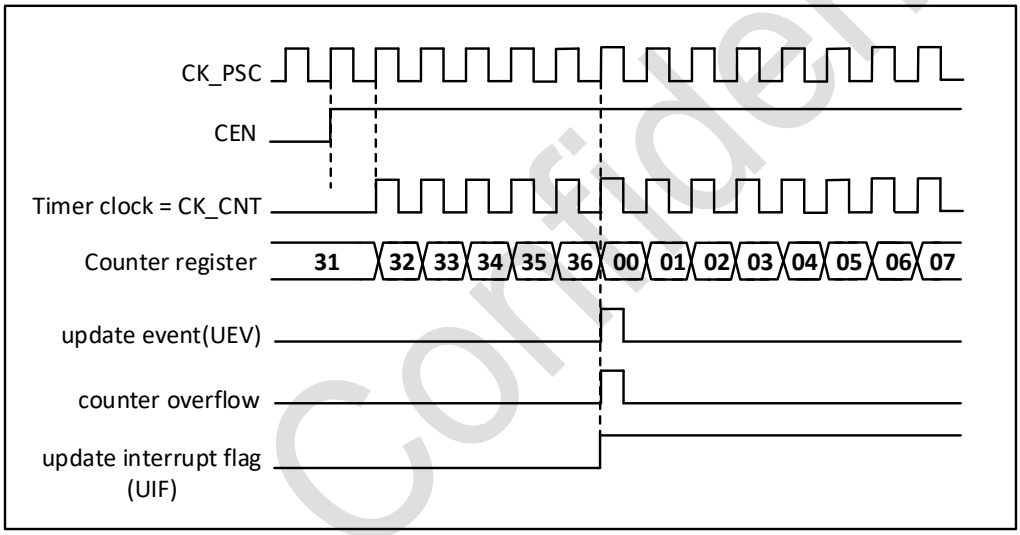


图 18-4 计数器时序图，内部时钟 1 分频

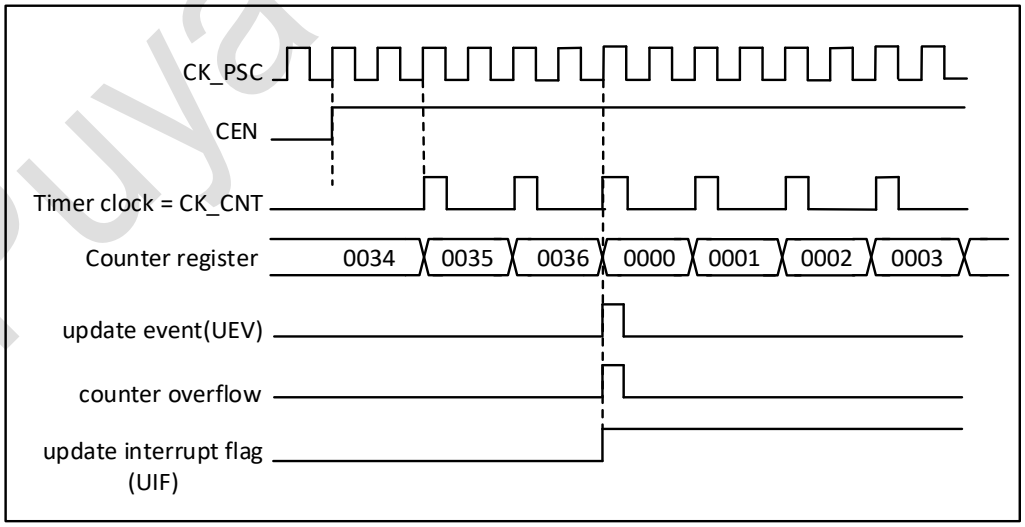


图 18-5 计数器时序图，内部时钟 2 分频

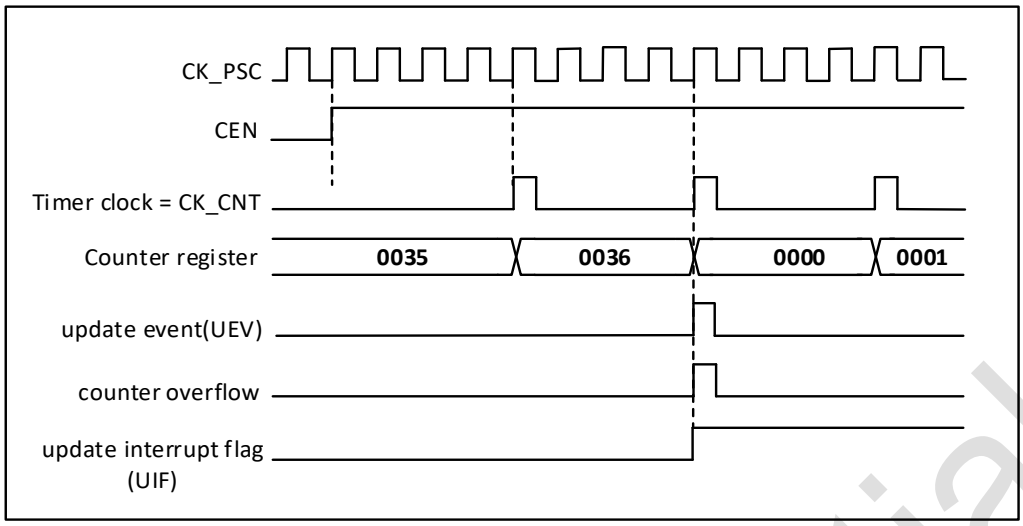


图 18-6 计数器时序图，内部时钟 4 分频

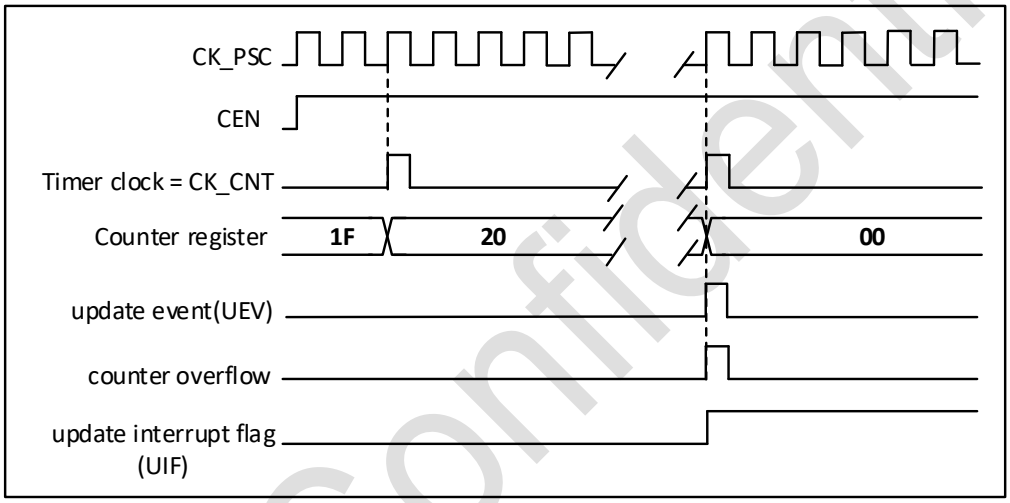


图 18-7 计数器时序图，内部时钟 N 分频

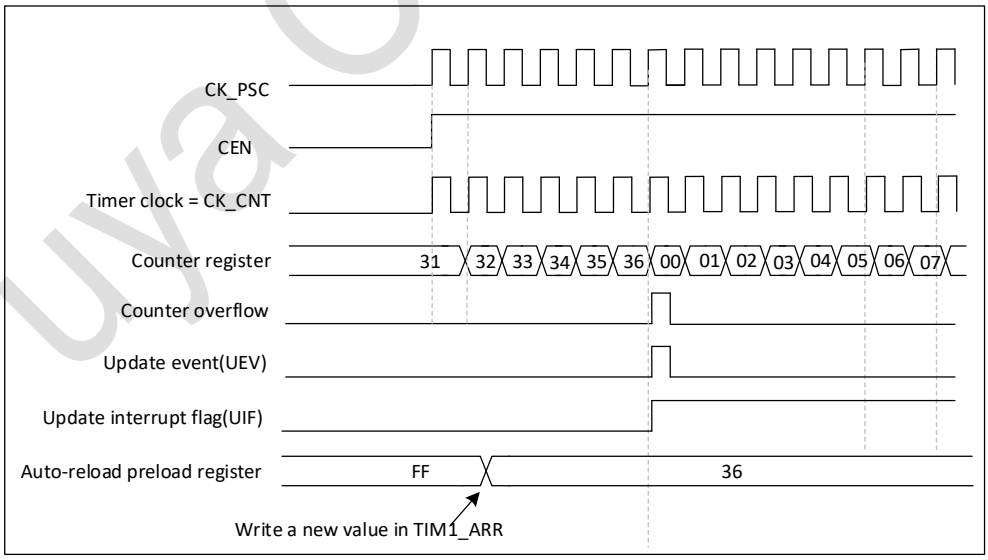


图 18-8 计数器时序图，ARPE=0 时更新事件（TIM1_ARR 未预装载）

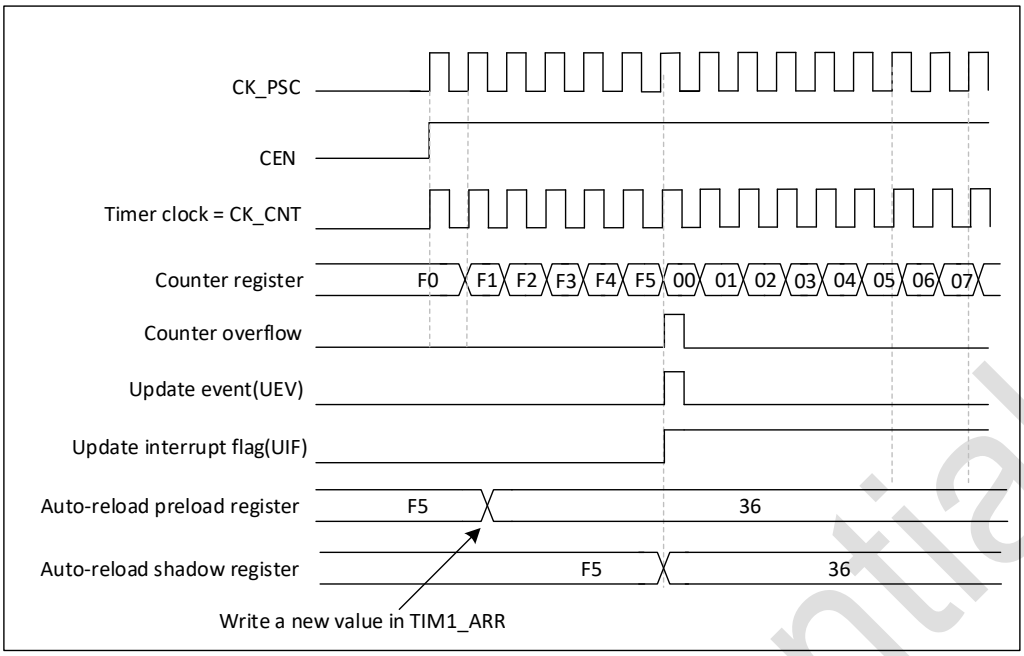


图 18-9 计数器时序图，ARPE=1 时更新事件（TIM1_ARR 预装载）

18.3.3.2. 递减计数模式

递减计数模式，从自动装载的值开始递减计数到 0，然后重新开始从自动装载的值递减计数，并产生一个向下溢出事件。

如果使用了重复计数器，当递减计数的重复次数达到重复计数器寄存器中设定的次数加一次(TIM1_RCR) +1）后，将生成更新事件(UEV)。否则每次计数器下溢时才产生更新事件。

在 TIM1_EGR 寄存器中设置 UG 位(通过软件方式或者使用从模式控制器)也同样可以产生一个更新事件。

设置 TIM1_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。

此外，如果设置了 TIM1_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIM1_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）。

- 重复计数器被重置为 TIM1_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载的值(TIM1_PSC 寄存器的内容)。
- 当前的自动加载寄存器被更新为预装载值(TIM1_ARR 寄存器中的内容)。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

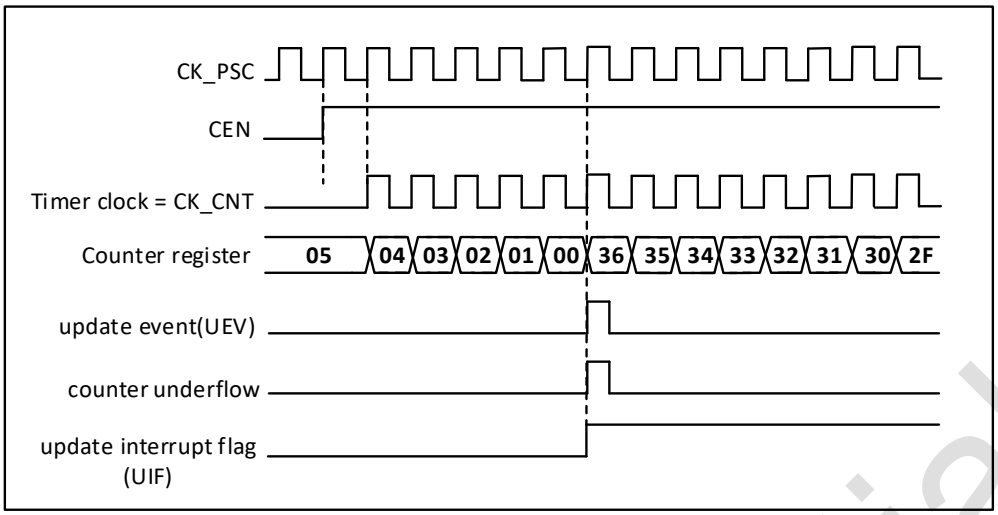


图 18-10 计数器时序图，内部时钟 1 分频

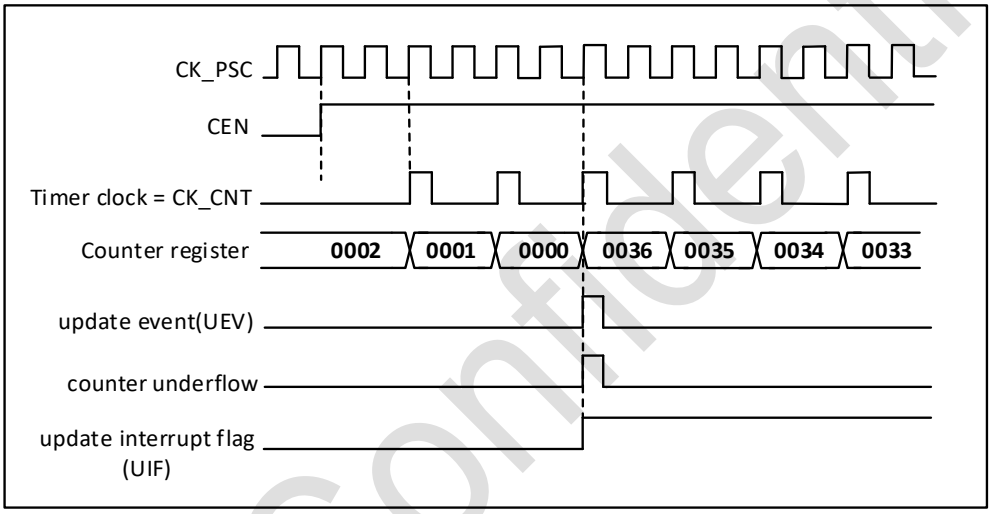


图 18-11 计数器时序图，内部时钟 2 分频

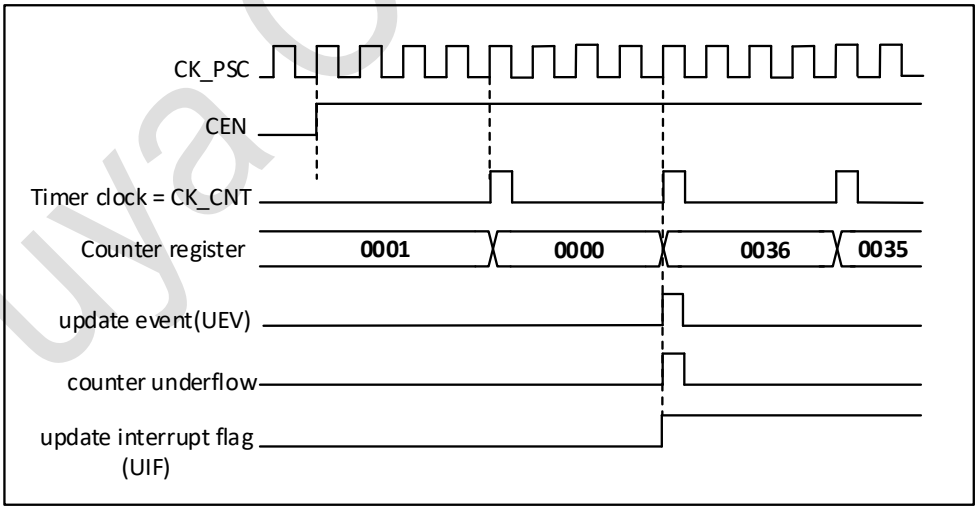


图 18-12 计数器时序图，内部时钟 4 分频

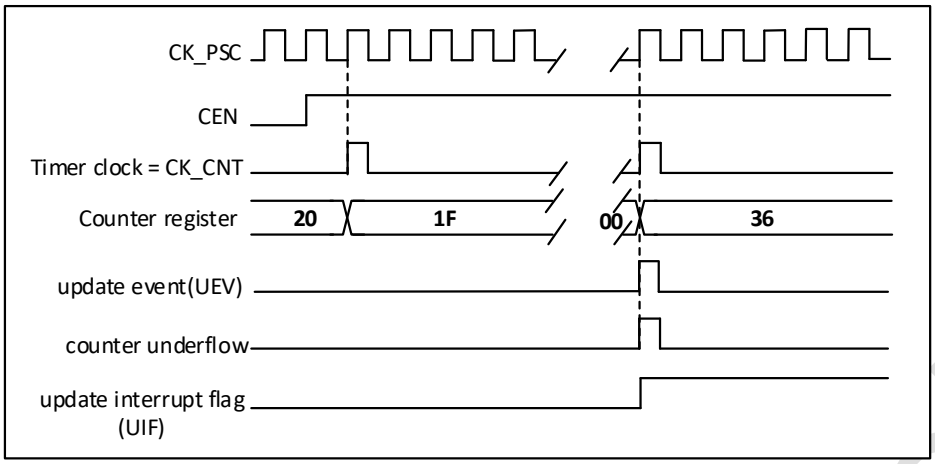


图 18-13 计数器时序图，内部时钟 N 分频

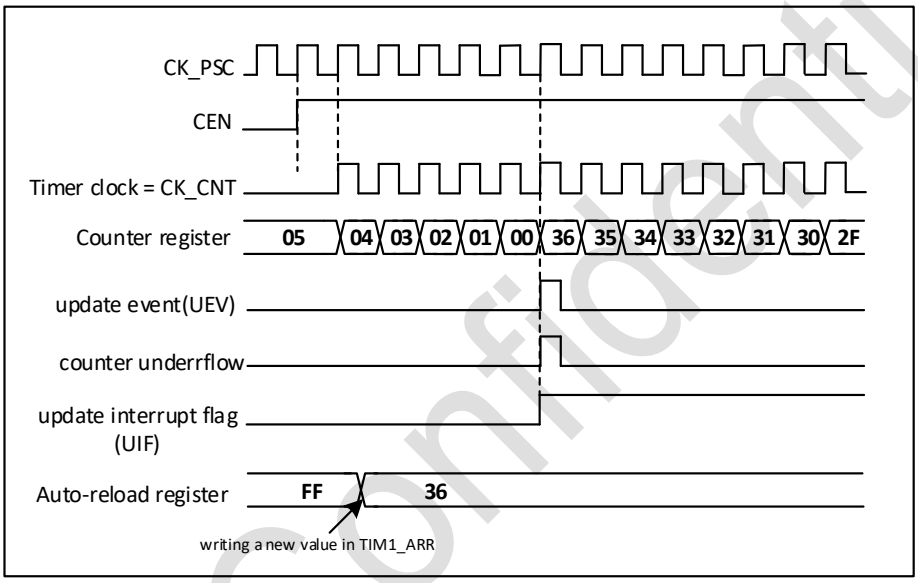


图 18-14 计数器时序图，未使用重复计数器时更新事件

18.3.3.3. 中心对齐模式

在中心对齐模式，计数器从 0 开始计数到自动重载值(TIM1_ARR 寄存器)−1，产生一个计数器上溢事件；然后从自动重载值递减计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

当 TIM1_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS =“01”）、计数器递增计数（中心对齐模式 2，CMS =“10”）以及计数器递增/递减计数（中心对齐模式 3，CMS =“11”）。

在此模式下，不能写入 TIM1_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 TIM1_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIM1_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或递减计数。

此外，如果设置了 TIM1_CR1 寄存器中的 URS 位(选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIM1_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）。

- 重复计数器被重置为 TIM1_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载值(TIM1_PSC 寄存器的内容)
- 当前的自动装载寄存器被更新为预装载值(TIM1_ARR 寄存器中的内容)。

注意：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前更新，因此下一个周期才是预期的值(计数器被装载为新的值)。

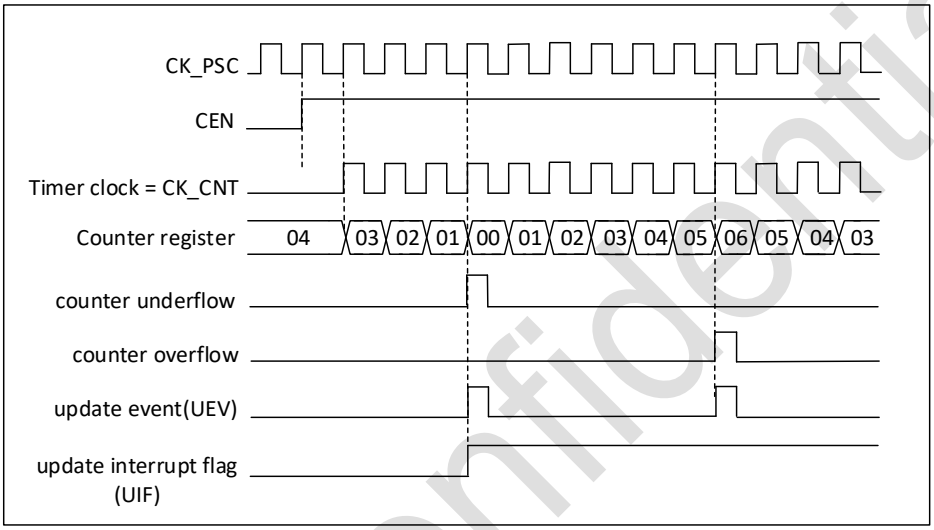


图 18-15 计数器时序图，内部时钟 1 分频, TIM1_ARR = 0x6

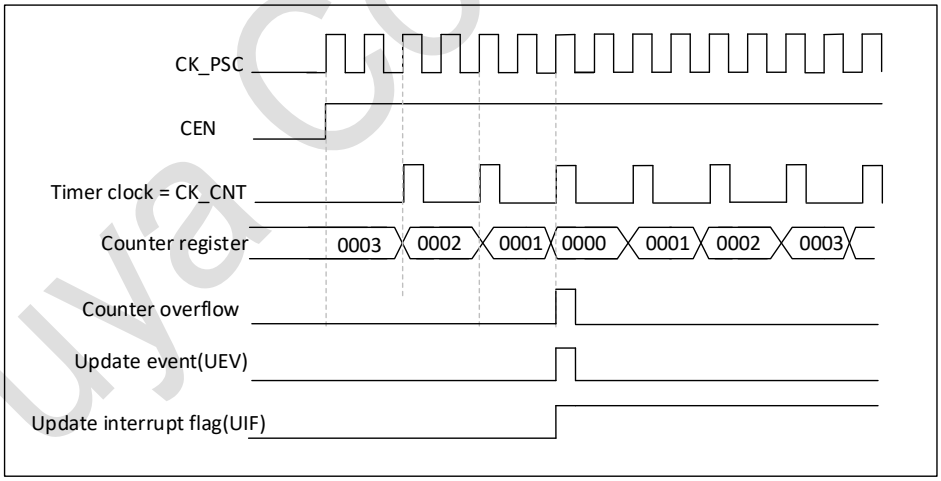


图 18-16 计数器时序图，内部时钟 2 分频

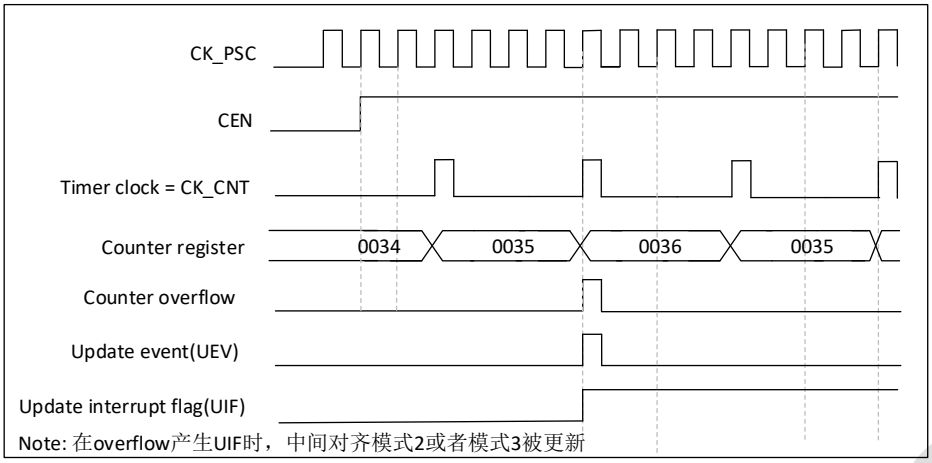


图 18-17 计数器时序图，内部时钟 4 分频, TIM1_ARR=0x36

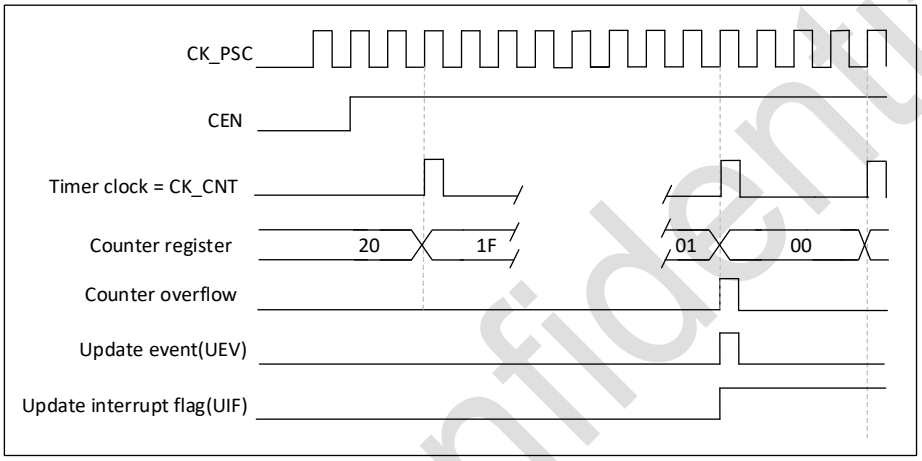


图 18-18 计数器时序图，内部时钟 N 分频

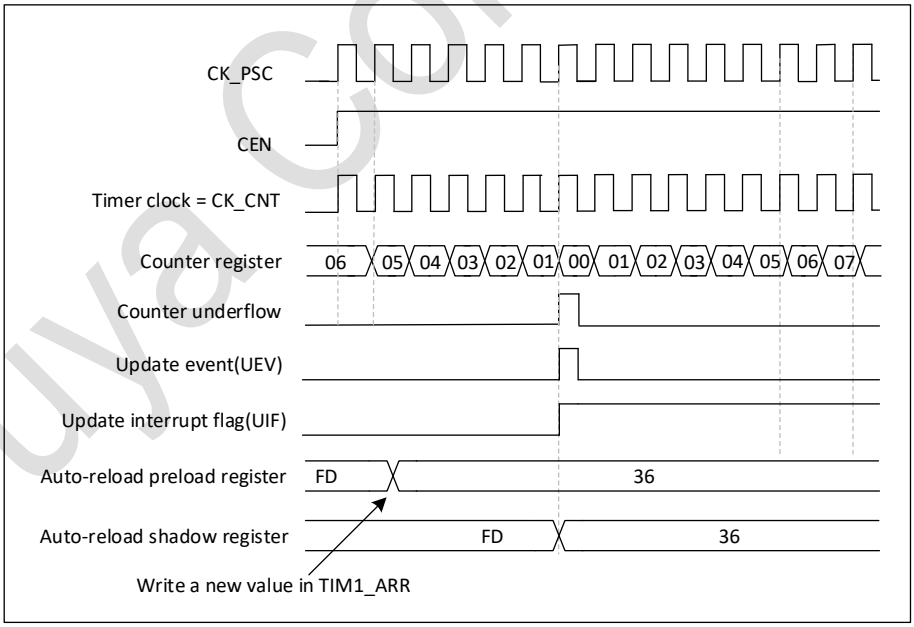


图 18-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

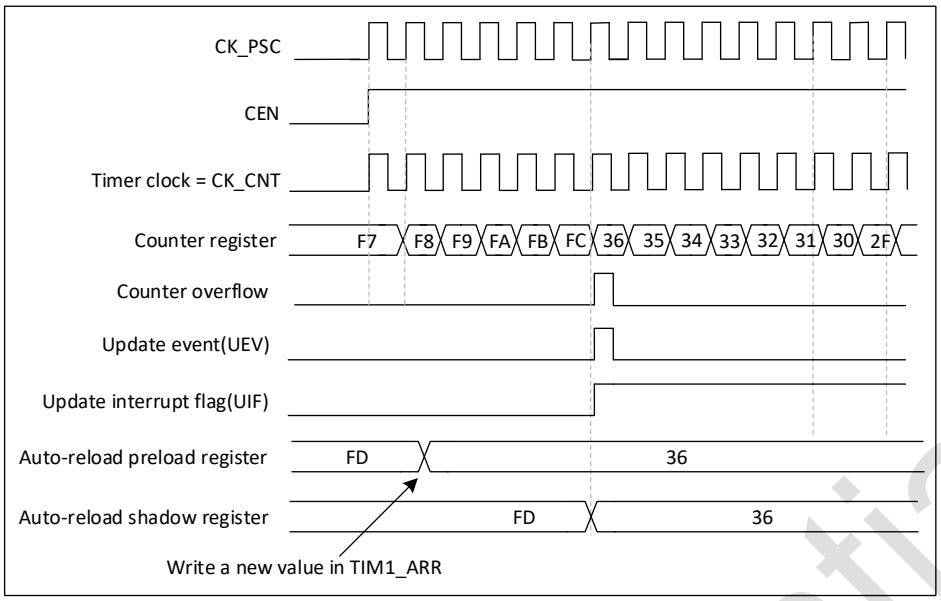


图 18-20 计数器时序图，ARPE=1 时的更新事件(计数器上溢)

18.3.4. 重复计数器

时基单元描述了计数器向上、向下溢出的更新事件（UEV）是如何产生的。它实际上仅当重复计数器计数到零才产生。这在产生 PWM 信号时很有用。

这意味着在每 N+1 次计数上溢或下溢时（N 是 TIM1_RCR 重复计数寄存器中的值），数据被从预装载寄存器传送到影子寄存器（TIM1_ARR 自动重载入寄存器，TIM1_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIM1_CCRx）。

重复计数器在下述任何一条件成立时递减：

- 递增计数模式下每次计数器上溢
- 递减计数模式下每次计数下溢
- 中心对齐模式下每次上溢和每次下溢时。虽然这样限制了 PWM 的最大循环周期为 128，但它能够在每个 PWM 周期两次更新占空比。在中心对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 2xTck。

重复计数器是自动加载的，重复速率是由 TIM1_RCR 寄存器的值定义。当更新事件由软件产生（通过设置 TIM1_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 TIM1_RCR 寄存器中的内容被重载如到重复计数器。

在中心对齐模式下，如果 RCR 值为奇数，更新事件将在上溢或下溢时发生，这取决于何时写入 RCR 寄存器以及何时启动计数器：如果在启动计数器前写入 RCR，则 UEV 在上溢时发生。如果在启动计数器后写入 RCR，则 UEV 在下溢时发生。

例如，对于 RCR = 3 时，更新事件被产生在第 4 个上溢或者下溢事件（取决于何时写入 RCR 的值）。

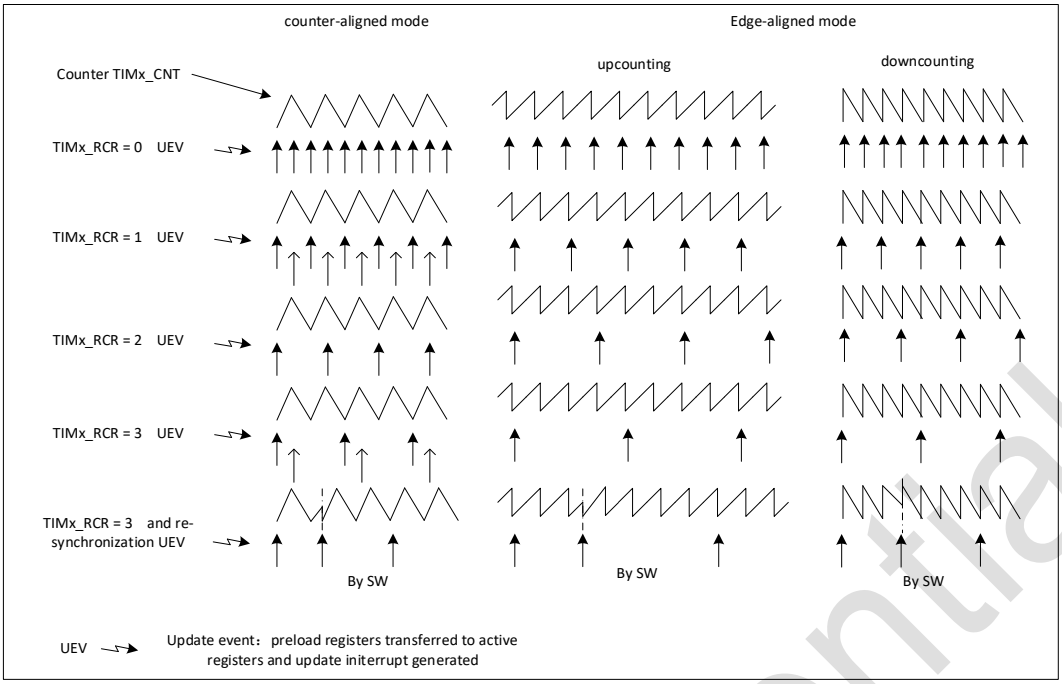


图 18-21 不同模式和 TIM1_RCR 寄存器设置下的更新频率示例

18.3.5. 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
- 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 TIM1_SMCR 寄存器中的 ETP 位定义。触发信号可通过 ETPS[1:0] 位域编程的分频比进行预分频，然后通过 ETF[3:0] 位域进行数字滤波。

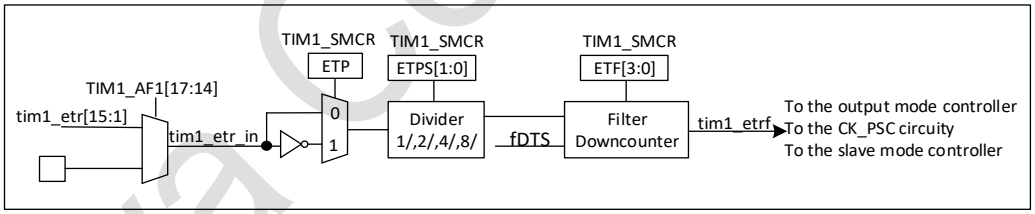


图 18-22 外部触发输入

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。使用 ETRSEL[3:0]位域进行选择。具体选择参见 TIM1_AF1.ETRSEL 寄存器的定义。

18.3.6. 时钟源

计数器的时钟可以由以下时钟源提供：

- 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。例如，可以配置一个定时器 1 作为另一个定时器 3 的预分频器。

18.3.6.1. 内部时钟源（CK_INT）

如果从模式控制器被禁止（SMS=000），则 CEN、DIR（TIM1_CR1 寄存器）和 UG 位（TIM1_EGR 寄存器）为实际控制位，并且只能被软件修改（UG 除外，仍保持自动清零）。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。

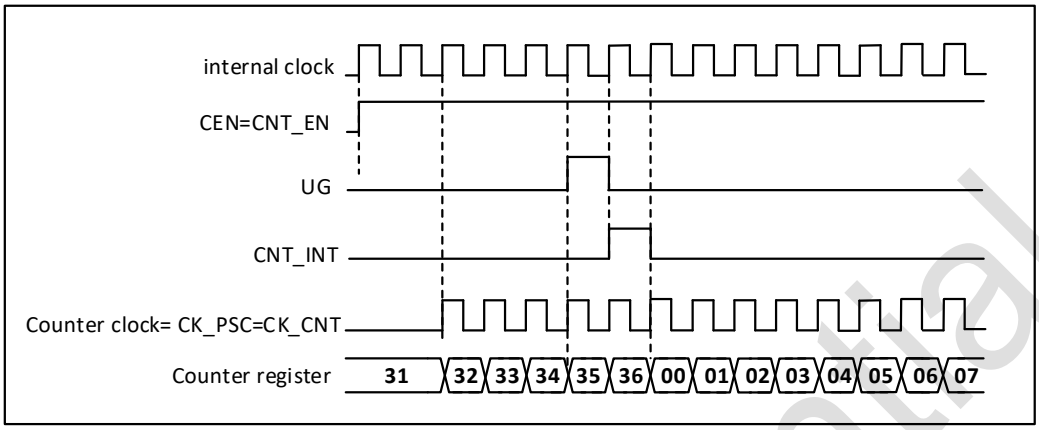


图 18-23 正常模式下的控制电路，内部时钟 1 分频

18.3.6.2. 外部时钟源模式 1

当 TIM1_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

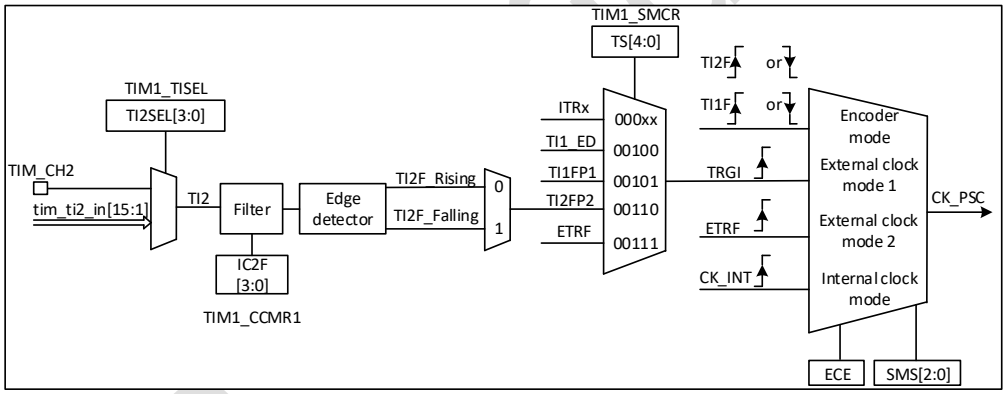


图 18-24 TI2 外部时钟连接示例

例如，要使递增计数器在 TI2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIM1_TISEL 寄存器中配置 TI2SEL[3:0]位选择正确的 TI2x 源（内部或外部）。
2. 通过在 TIM1_CCMR1 寄存器中写入 CC2S =“01”来配置通道 2，使其能够检测 TI2 输入的上升沿。
3. 通过在 TIM1_CCMR1 寄存器中写入 IC2F[3:0]位来配置输入滤波带宽（如果不需要任何滤波器，请保持 IC2F=0000）。
4. 通过在 TIM1_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
5. 通过在 TIM1_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
6. 通过在 TIM1_SMCR 寄存器中写入 TS=00110 来选择 TI2 作为触发输入源。
7. 通过在 TIM1_CR1 寄存器中写入 CEN=1 来使能计数器。

注：由于捕获预分频器不用于触发操作，因此用户无需对其进行配置。

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

TI2 的上升沿与实际计数器时钟之间的延迟是由于 TI2 输入的重新同步电路引起的。

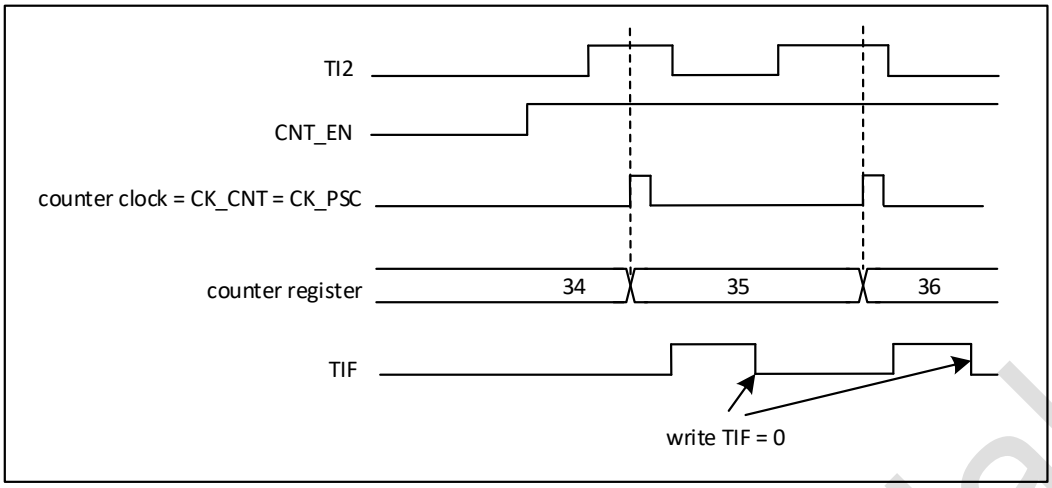


图 18-25 外部时钟模式 1 控制电路时序

18.3.6.3. 外部时钟源模式 2

通过写 TIM1_SMCR 寄存器的 ECE 为 1，选定此模式。计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

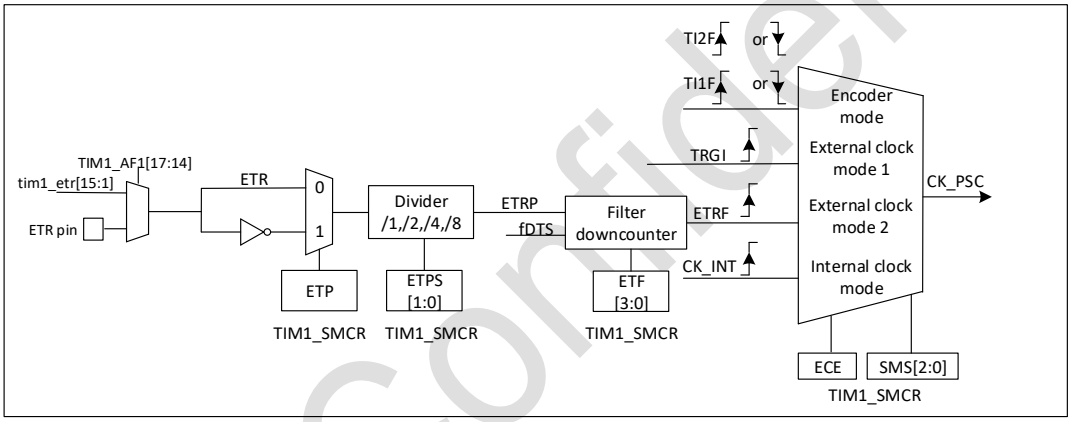


图 18-26 外部触发输入模块

例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIM1_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIM1_SMCR 寄存器中写入 ETPS[1:0]=01 来设置预分频器。
3. 通过在 TIM1_SMCR 寄存器中写入 ETP=0 来选择 ETR 引脚的上升沿检测。
4. 通过在 TIM1_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
5. 通过在 TIM1_CR1 寄存器中写入 CEN=1 来使能计数器。

ETR 每出现 2 个上升沿，计数器计数一次。

ETR 的上升沿与实际计数器时钟之间的延迟是由于 ETRP 信号的重新同步电路引起的。

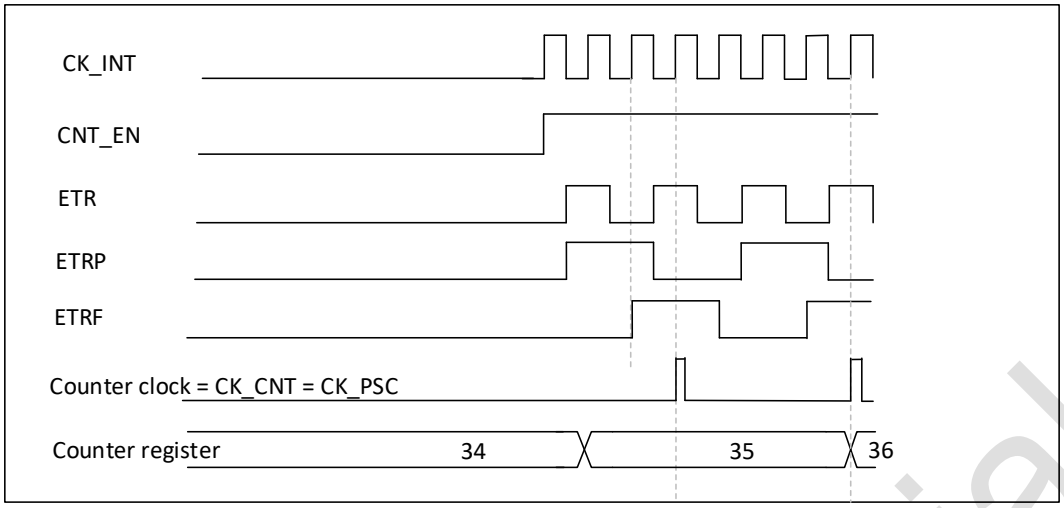


图 18-27 外部时钟模式 2 控制电路时序

18.3.7. 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入部分（数字滤波、多路复用和预分频器）和一个输出部分（比较器和输出控制）构建而成。

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

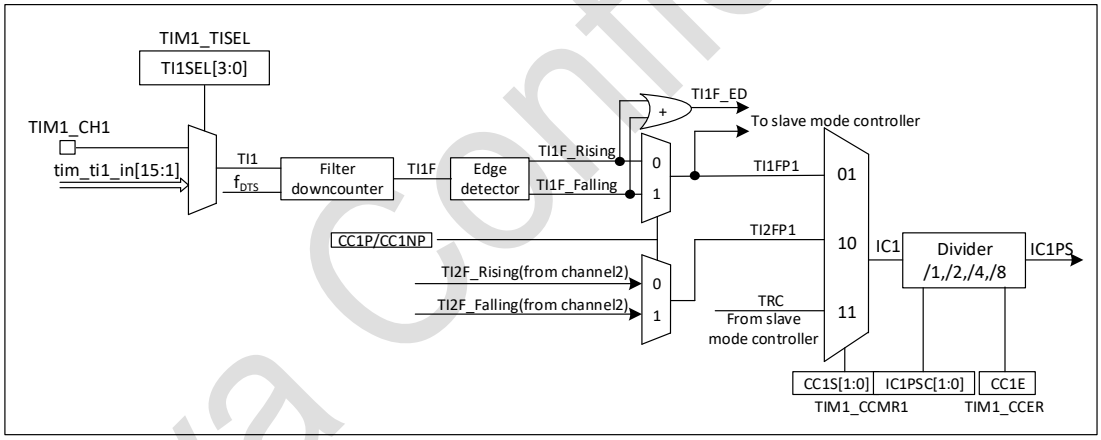


图 18-28 捕获/比较通道(示例：通道 1 输入阶段)

输出部分产生一个中间波形 OCxREF(高有效)作为基准，链的末端决定最终输出信号的极性。

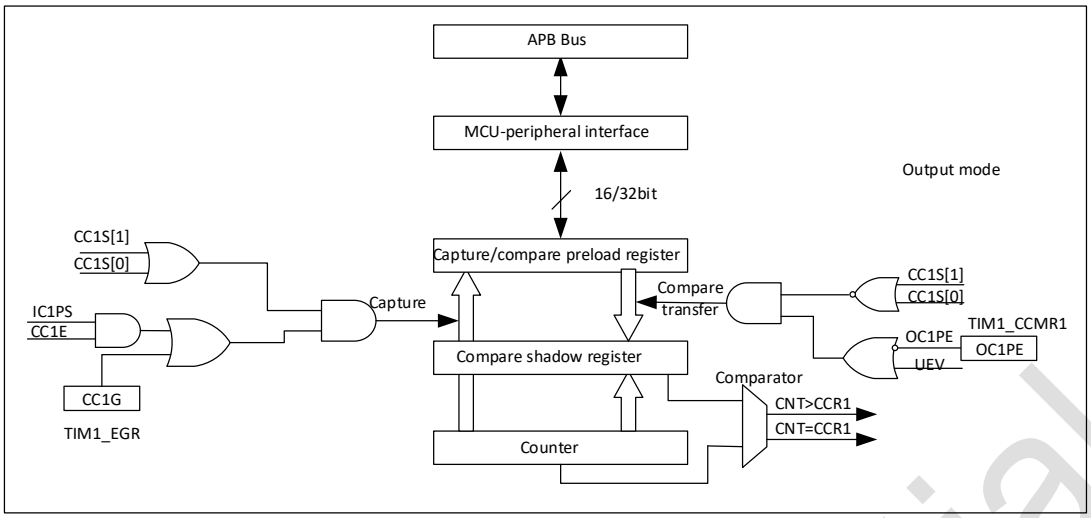


图 18-29 捕获/比较通道 1 主电路

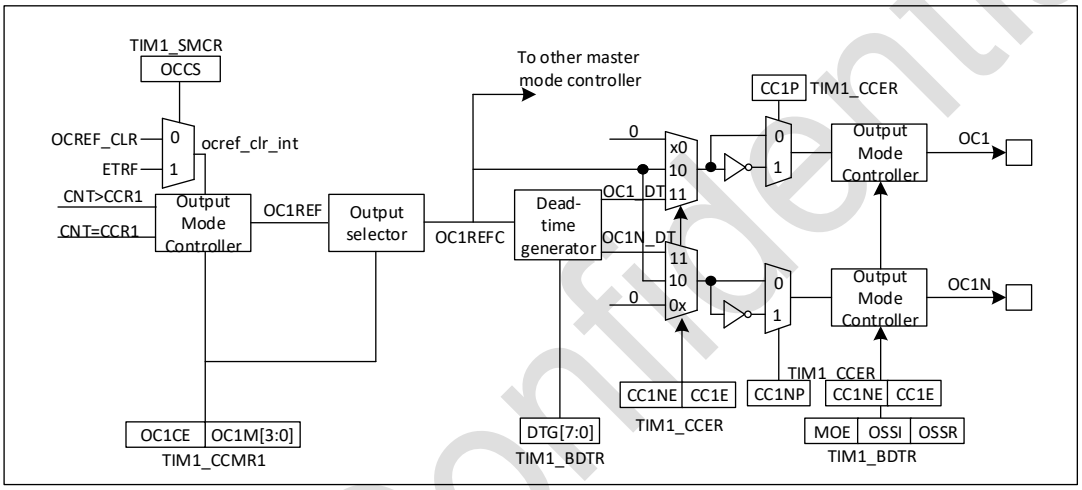


图 18-30 捕获/比较通道的输出阶段(通道 1 到 3)

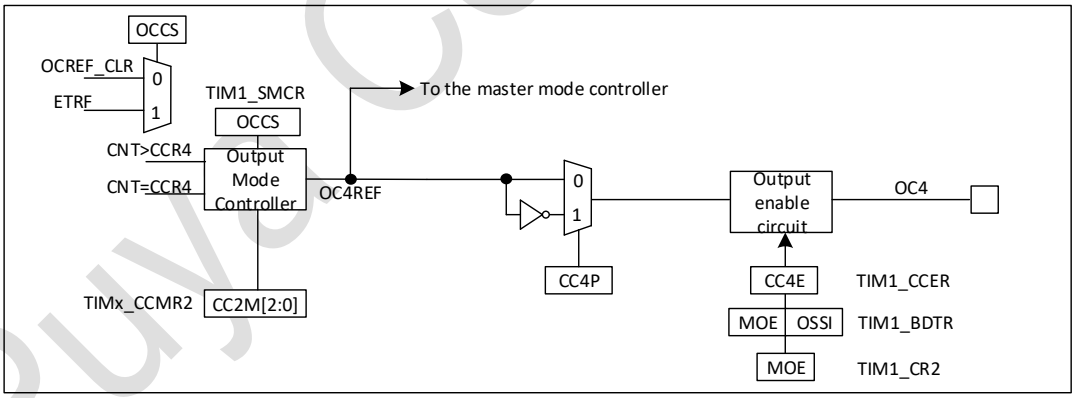


图 18-31 捕获/比较通道的输出阶段(通道 4)

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，然后将影子寄存器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。

18.3.8. 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/比较寄存器中。当发生捕获事件时，相应的 CCxIF 标志（TIM1_SR 寄存器）被置 1，如果中断和 DMA 操作被打开，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，则重复捕获标志

CCxOF（TIM1_SR 寄存器）被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIM1_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 TIM1_CCR1 寄存器中，步骤如下：

- 使用 TIM1_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）
- 选择有效输入端 TIM1_CCR1 必须连接到 TI1 输入，所以写入 TIM1_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为'00'，通道被配置为输入，并且 TIM1_CCR1 寄存器变为只读。
- 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 TIM1_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；在检测到连续 8 个具有新电平的采样(以 f_{DTS} 频率)后，可以确认在 TI1 上的一次跳边沿，然后向 TIM1_CCMR1 寄存器中写入 IC1F=0011。
- 通过在 TIM1_CCER 寄存器中将 CC1P 位和 CC1NP 位写入 0，选择 TI1 上的有效转换边沿(本例中为上升沿)
- 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，因此需要禁止预分频器(写 TIM1_CCMR1 寄存器的 IC1PSC=00)。
- 设置 TIM1_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 TIM1_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIM1_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM1_CCR1 寄存器。
- CC1IF 标志被设置(中断标志)。当发生至少连续两次捕获时，但 CC1IF 未被清除，则 CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则会产生一个 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与读取数据之前可能产生的捕获溢出信息。

注：设置 TIM1_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

18.3.9. 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：

- 两个 ICx 信号被映射到同一个 TIx 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 选择其中一个 TIxFP 信号作为触发输入信号，而从模式控制器被配置成复位模式。
- 例如，当需要测量输入到 TI1 上的 PWM 信号的长度(TIM1_CCR1 寄存器)和占空比(TIM1_CCR2 寄存器)时，具体步骤如下(取决于 CK_INT 的频率和预分频器的值）。
- 使用 TIM1_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）。
- 选择 TIM1_CCR1 的有效输入：置 TIM1_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIM1_CCR1 中和清除计数器)，置 CC1P 和 CC1NP 为 0(上升沿有效)。
- 选择 TIM1_CCR2 的有效输入：置 TIM1_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIM1_CCR2)：置 CC2P=1，CC2NP=0(下降沿有效)。
- 选择有效的触发输入信号：置 TIM1_SMCR 寄存器中的 TS=00101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIM1_SMCR 中的 SMS=0100。

- 使能捕获：置 TIM1_CCER 寄存器中 CC1E=1 且 CC2E=1。

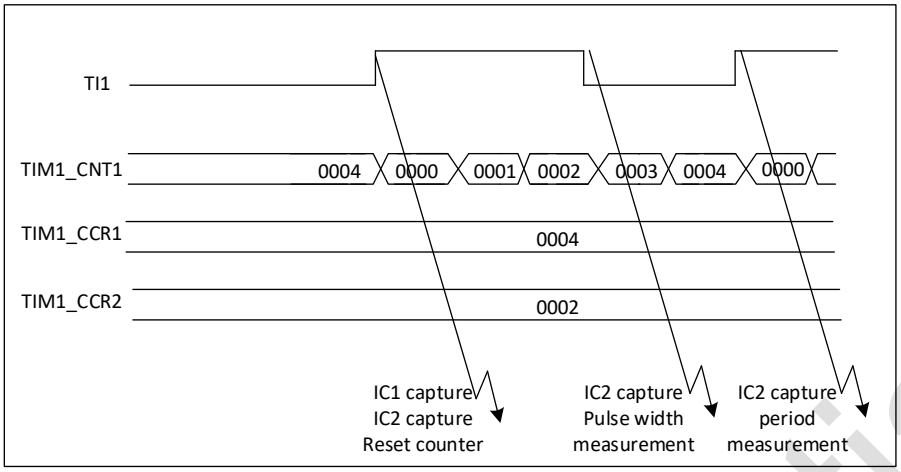


图 18-32 PWM 输入模式时序

18.3.10. 强制输出模式

在输出模式(TIM1_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 TIM1_CCMRx 寄存器中相应的 OCxM=0101，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。置 TIM1_CCMRx 寄存器中的 OCxM=0100，可强置 OCxREF 信号为低。

该模式下，在 TIM1_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。这将会在下面的输出比较模式一节中介绍。

18.3.11. 输出比较模式

此项功能用于控制一个输出波形，或者指示一段给定的时间已经到时。

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：

- 将输出比较模式(TIM1_CCMRx 寄存器中的 OCxM 位)和输出极性(TIM1_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=0000)、也可以被设置成有效电平(OCxM=0001)、被设置成无效电平(OCxM=0010)或进行翻转(OCxM=0011)。
- 设置中断状态寄存器中的标志位(TIM1_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断使能(TIM1_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的 DMA 使能位(TIM1_DIER 寄存器中的 CCxDE 位，TIM1_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIM1_CCMRx 中的 OCxPE 位选择 TIM1_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。时间的精度可以达到计数器的一个计数周期。输出比较模式也能用来输出一个单脉冲(在单脉冲模式下)。

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 TIM1_ARR 和 TIM1_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：

- 设置 OCxM=0011，则计数器与 CCRx 匹配时翻转 OCx 的输出引脚，
- 置 OCxPE = 0 时禁用预装载寄存器
- 置 CCxP = 0 选择极性为高电平有效
- 置 CCxE = 1 使能输出

5. 设置 TIM1_CR1 寄存器的 CEN 位启动计数器

TIM1_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，条件是未使能预装载寄存器(OCxPE='0'，否则 TIM1_CCRx 的影子寄存器只能在发生下一次更新事件时进行更新)。下图给出了一个例子。

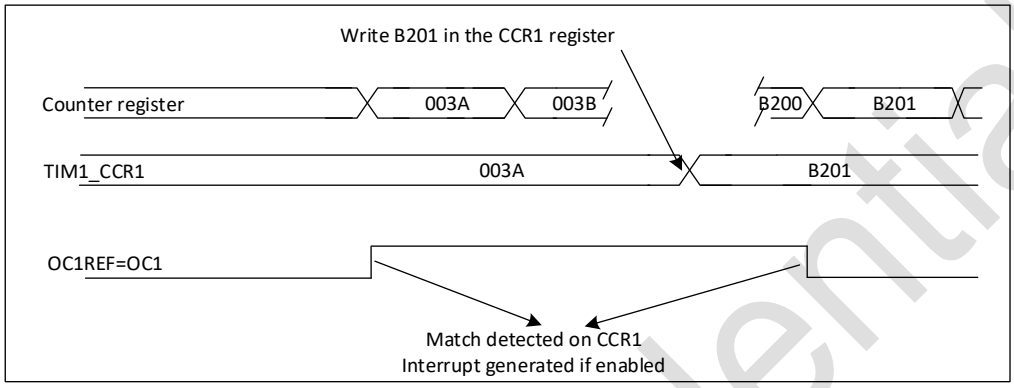


图 18-33 输出比较模式，翻转 OC1

18.3.12. PWM 模式

脉冲宽度调制模式（PWM）可以允许产生一个由 TIM1_ARR 寄存器确定频率、由 TIM1_CCRx 寄存器确定占空比的信号。

在 TIM1_CCMRx 寄存器中的 OCxM 位写入“0110”（PWM 模式 1）或“0111”（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 TIM1_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIM1_CR1 寄存器的 ARPE 位为 1 使能自动重装载的预装载寄存器，(在递增计数或中心对称模式中)。

由于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 TIM1_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 TIM1_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。通过 CCxE、CCxNE、MOE、OSSI 和 OSSR 位(TIM1_CCER 和 TIM1_BDTR 寄存器中)的组合控制 OCx 的输出。详见 TIM1_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIM1_CNT 和 TIM1_CCRx 始终在进行比较，以确定是符合 TIM1_CCRx≤TIM1_CNT 或者符合 TIM1_CNT≤TIM1_CCRx(依据计数器的计数方向)。

根据 TIM1_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中心对齐的 PWM 信号。

18.3.12.1. PWM 边沿对齐模式

递增计数配置

当 TIM1_CR1 寄存器中的 DIR 位为低的时候执行递增计数。下面是一个 PWM 模式 1 的例子。当 TIM1_CNT<TIM1_CCRx 时，PWM 参考信号 OCxREF 为高电平，否则为低电平。如果 TIM1_CCRx 中的比较值大于自动重装载值(TIM1_ARR)，则 OCxREF 保持为'1'。如果比较值为 0，则 OCxREF 保持为'0'。 下图为 TIM1_ARR=8 时边沿对齐的 PWM 波形实例。

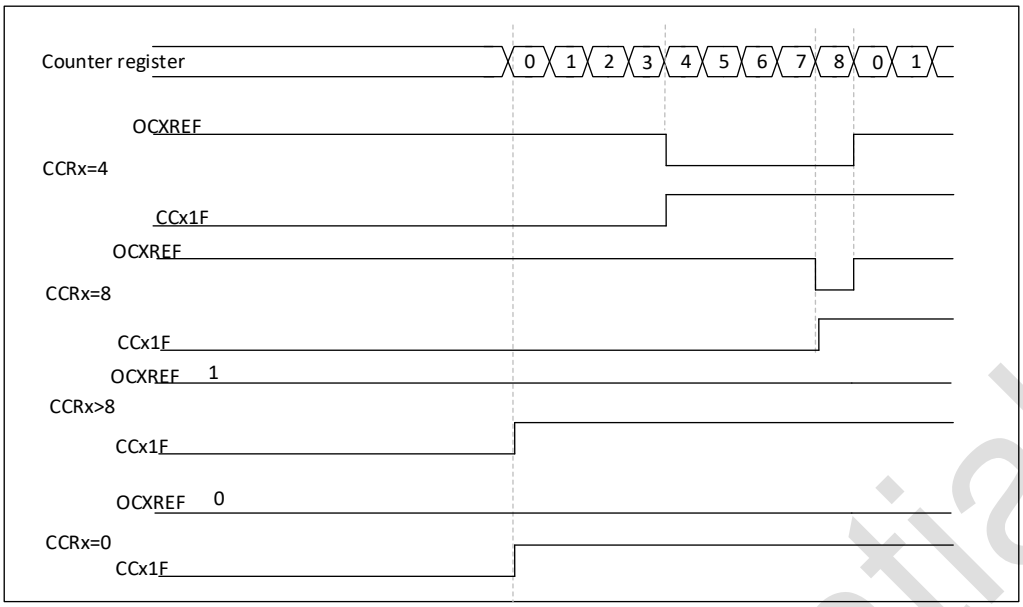


图 18-34 边沿对齐方式 PWM 输出波形（ARR=8）

递减计数配置

当 TIM1_CR1 寄存器的 DIR 位为高时执行递减计数。

在 PWM 模式 1，当 TIM1_CNT>TIM1_CCRx 时参考信号 OCxREF 为低电平，否则为高电平。如果 TIM1_CCRx 中的比较值大于 TIM1_ARR 中的自动重装载值，则 OCxREF 保持为'1'。该模式下不能产生 0%占空比的 PWM 波形。

18.3.12.2. PWM 中心对齐模式

当 TIM1_CR1 寄存器中的 CMS 位不为'00'时为中心对齐模式(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递增计数时被置 1、在计数器递减计数时被置 1、或在计数器递增和递减计数时置 1。TIM1_CR1 寄存器中的计数方向位(DIR)由硬件更新，不能用软件修改。

下图给出一些中心对齐的 PWM 波形的例子

- TIM1_ARR = 8
- PWM 模式 1
- TIM1_CR1 寄存器的 CMS=01，在中心对齐模式下，当计数器递减计数时，比较标志置 1.

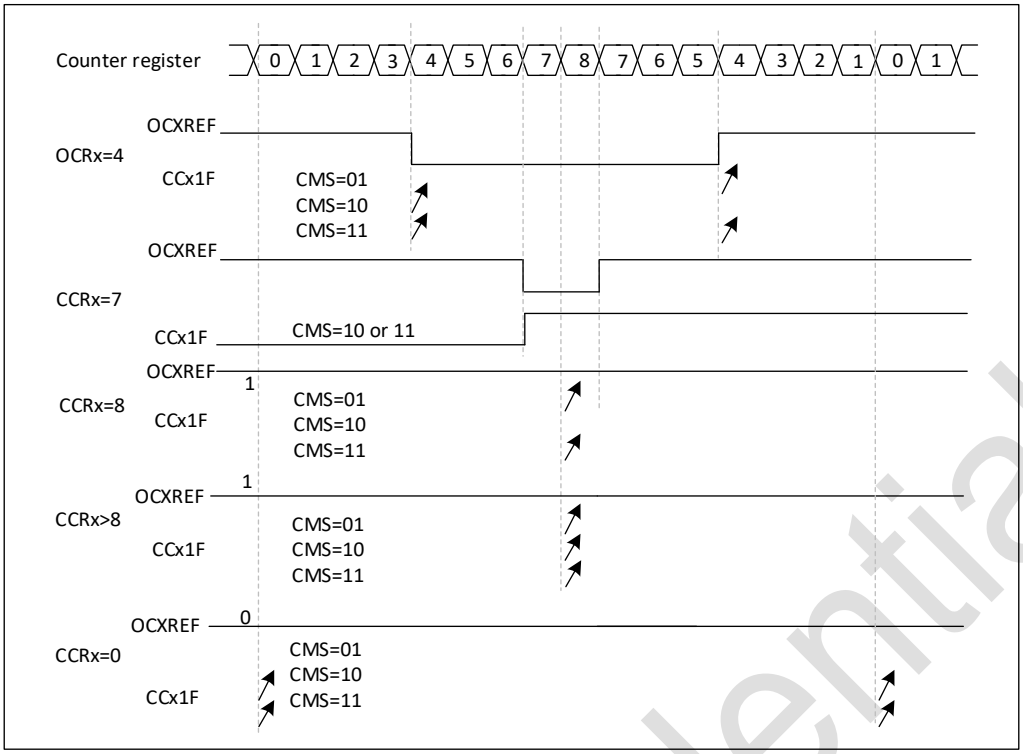


图 18-35 中心对齐模式波形(ARR=8)

中心对齐模式使用建议：

- 进入中心对齐模式时，使用当前的递增/递减计数配置；这就意味着计数器递增还是递减计数取决于 TIM1_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。
- 不推荐当运行在中心对齐模式时改写计数器，否则会产生不可预知的结果。尤其是：
  - 如果写入计数器的值大于自动重加载的值(TIM1_CNT>TIM1_ARR)，则方向不会被更新。例如，如果计数器正在递增计数，则会继续递增计数。
  - 如果将 0 或者 TIM1_ARR 的值写入计数器，计数方向会更新，但不产生更新事件 UEV。
- 使用中心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 TIM1_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。

18.3.13. 互补输出和死区插入

高级控制定时器(TIM1)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，用户必须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。

通过配置 TIM1_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：TIM1_CCER 寄存器的 CCxE 和 CCxNE 位，TIM1_BDTR 和 TIM1_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见 TIM1_CCER 寄存器后面表格中描述的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)，死区仍然有效。

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 8 位的死区发生器 DTG[7:0]。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高电平有效：

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

如果延迟大于当前有效的输出宽度(OCx 或者 OCxN)，则不会产生相应的脉冲。

下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。(假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1)

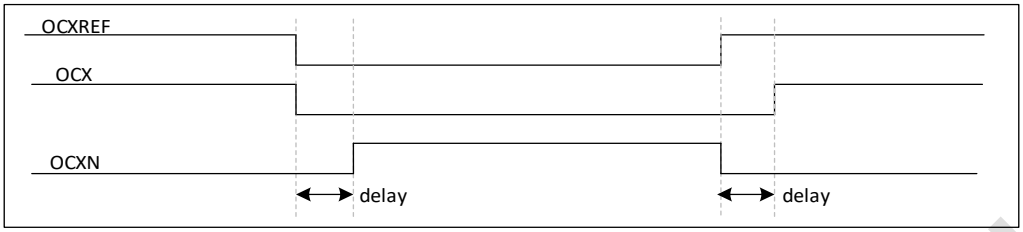


图 18-36 带死区插入的互补输出

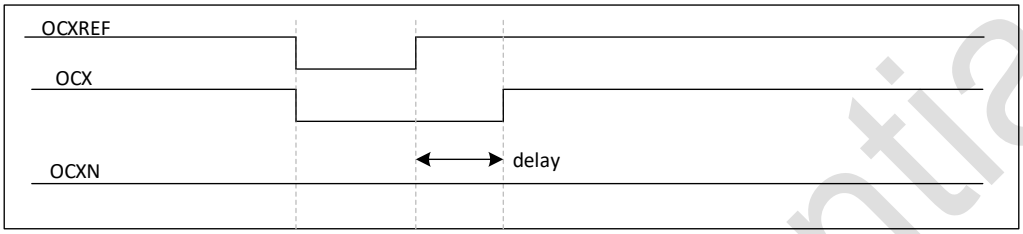


图 18-37 延迟时间大于负脉冲宽度的死区波形

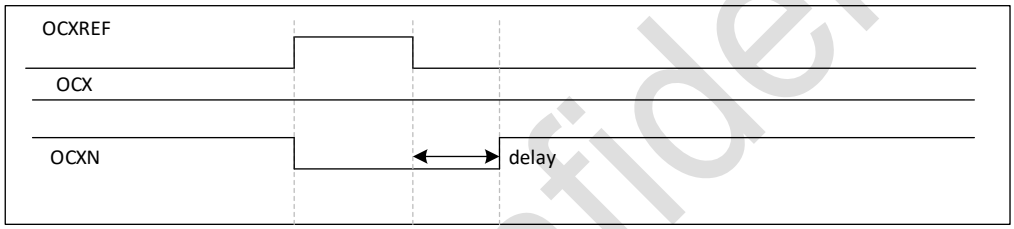


图 18-38 延迟时间大于正脉冲宽度的死区波形

每一个通道的死区延时都是相同的，是由 TIM1_BDTR 寄存器中的 DTG 位编程配置。

18.3.13.1. 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置模式、输出比较模式或 PWM 模式)，通过配置 TIM1_CCER 寄存器的 CCxE 和 CCxNE 位，可以将 OCxREF 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)，而同时使互补输出保持处于无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时保持有效电平和带死区的互补输出。

注： 当只使能 OCxN(CCxE=0, CCxNE=1)时，两者不互补，当 OCxREF 有效时 OCxN 立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当同时使能 OCx 和 OCxN 时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 则相反，当 OCxREF 低时 OCxN 变为有效。

18.3.14. 使用刹车功能

刹车功能的目的是保护由 TIM1 定时器生成的 PWM 信号所驱动的功率开关。两个刹车输入通常连接到功率级和三相逆变器的故障输出。激活时，刹车电路会关闭 PWM 输出，并将其强制为预定义的安全状态。也可选择一些内部 MCU 事件来触发输出关断。

刹车通道源包括系统级故障（时钟失效和奇偶校验错误等）和应用故障（来自输入引脚和内置比较器），可以在死区持续时间后将输出强制为预定义的电平（有效或无效）。

刹车期间的输出使能信号和输出电平取决于多个控制位：

- TIM1_BDTR 寄存器中的 MOE 位，允许通过软件使能/禁止输出，在发生刹车事件时复位。

- TIM1_BDTR 寄存器中的 OSSI 位，定义定时器将输出控制在无效状态下，还是释放对 GPIO 控制器的控制（通常使其处于高阻态模式）
- TIM1_CR2 寄存器中的 OISx 和 OISxN 位，将输出设置为关断电平（有效或无效）。无论 OISx 和 OISxN 的值为何，均无法在给定时间将 OCx 和 OCxN 输出同时设置为有效电平。

退出复位状态后，刹车功能处于禁止状态，MOE 位处于低电平。通过设置 TIMx_BDTR 寄存器中的 BKE 位来使能刹车功能。可通过配置同一寄存器中的 BKP 位来选择刹车输入的极性。BKE 和 BKP 位可同时修改。

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 TIM1_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。

刹车源既可以是刹车输入引脚，或者以下内部源（详细内容参见“外设互联”章节）：

- CPU LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟错误事件
- 来自比较器的输出
- SRAM 字节校验错误
- FLASH ECC 错误

也可由软件通过 TIM1_EGR 寄存器中的 BG 位产生刹车事件。无论 BKE 使能位的值如何，都可以使用 BG 通过软件产生刹车。

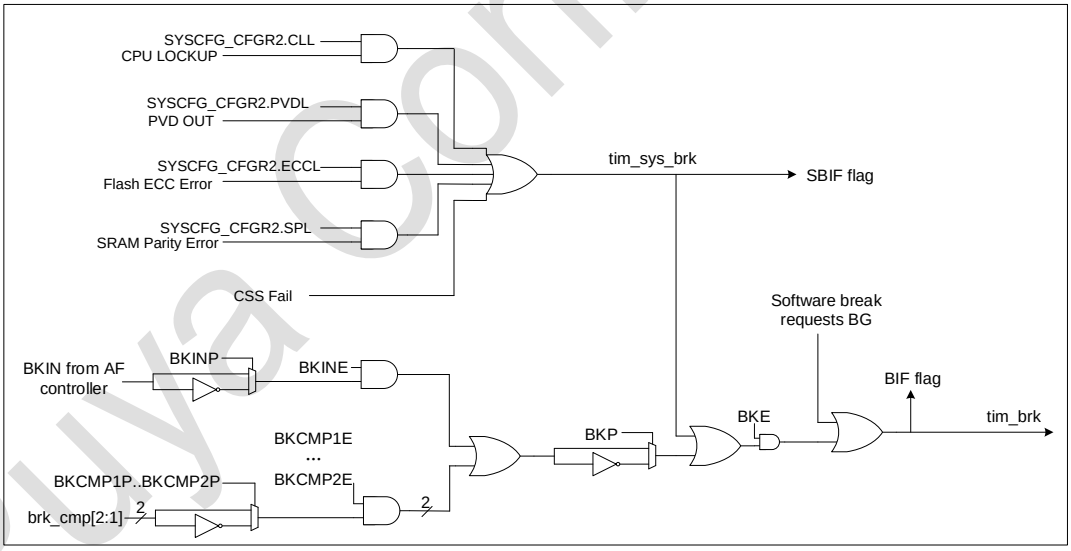


图 18-39 刹车电路图

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、空闲状态或者释放对 GPIO 的控制（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，将以 TIM1_CR2 寄存器中的 OISx 位设定的电平驱动每一个输出通道。如果 OSSI=0，则定时器释放输出控制（由 GPIO 控制器接管），否则使能输出始终为高电平。
- 当使用互补输出时：
  - 输出首先被置于无效的状态(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 个定时器的时钟周期)。

— 如果 OSSI=0，定时器释放使能输出（由强制高阻态的 GPIO 控制器接管），否则使能输出将保持高电平或在 CCxE 或 CCxNE 位之一为高电平时立即变为高电平。

- 如果设置了 TIM1_DIER 寄存器中的 BIE 位，当刹车状态标志(TIM1_SR 寄存器中的 SBIF 和 BIF 位)为'1'时，则产生一个中断。
- 如果设置了 TIM1_BDTR 寄存器中的 AOE 位，则 MOE 位会在下一个更新事件（UEV）时自动再次置位；例如，这可以用来进行整形。否则，MOE 始终保持低电平直到应用将其再次置'1'；这种情况下，这个特性可以确保安全，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，不能设置 MOE 位为 1 (自动地或者通过软件)。同时，状态标志 BIF 不能被清除。

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 TIM1_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 TIM1_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

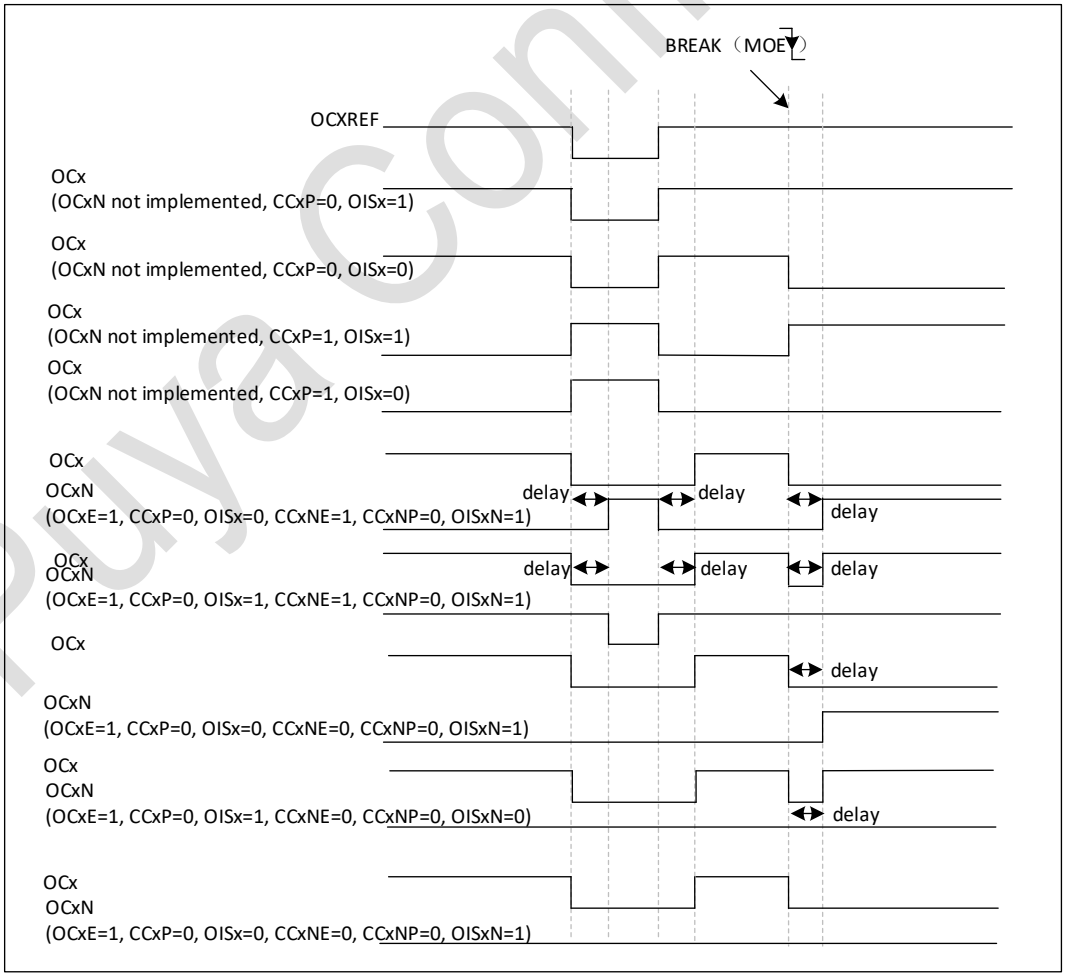


图 18-40 响应刹车事件的不同输出行为（OSSI=1）

18.3.15. 发生外部事件时清除 OCxREF 信号

对于给定通道，在 ocref_clr_int 输入上施加高电平（相应 TIM1_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 OCxREF 信号清零。 OCxREF 信号将保持低电平，直到发生下一更新事件(UEV)。该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。通过配置 TIM1_SMCR 寄存器中的 OCCS 位，可在 OCREF_CLR 输入和 ETRF（滤波器后的 ETR）之间选择 ocref_clr_int 输入。

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：TIM1_SMCR 寄存器中的 ETPS[1:0]=00。
2. 必须禁止外部时钟模式 2：TIM1_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 TIM1 被置于 PWM 模式。

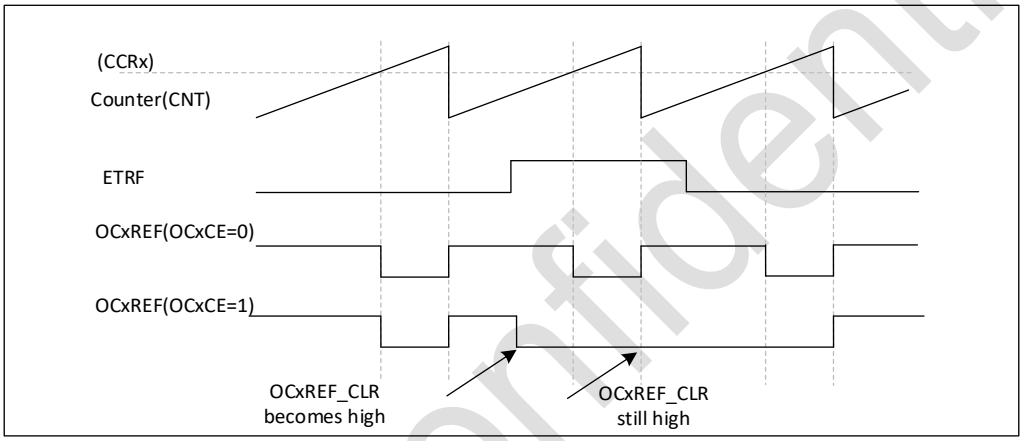


图 18-41 清除 TIM1 的 OCxREF

注：如果 PWM 的占空比为 100%（CCRx>ARR），则下次计数器溢出时会再次使能 OCxREF。

18.3.16. 6 步 PWM

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 事件时，这些预装载位将传送到影子寄存器位。因此，用户可以预先设置好下一步骤的配置，并在同一个时刻同时修改所有通道的配置。COM 可以通过设置 TIM1_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(TIM1_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIM1_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIM1_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，OCx 和 OCxN 输出的行为。

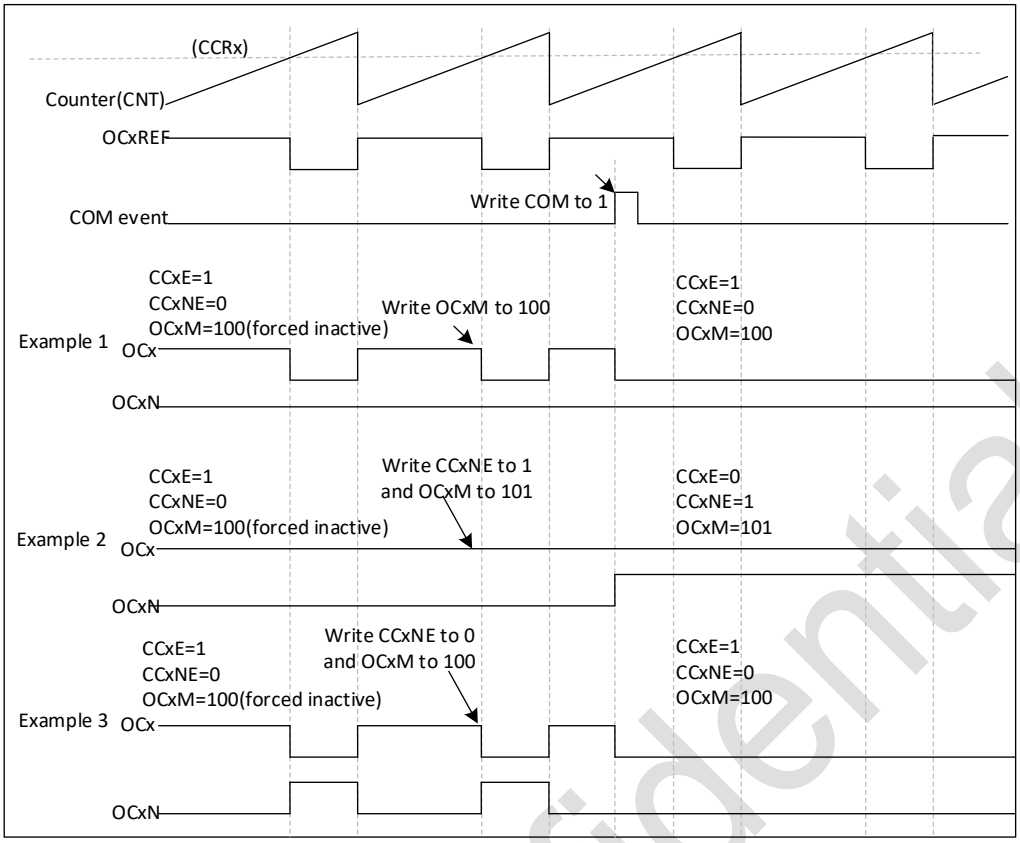


图 18-42 COM 事件生成 6 步 PWM 的示例 (OSSR=1)

18.3.17. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。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，产生一个脉宽可被程序控制的脉冲。

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 TIM1_CR1 寄存器的 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：

- 递增计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)
- 递减计数方式：计数器 CNT > CCRx

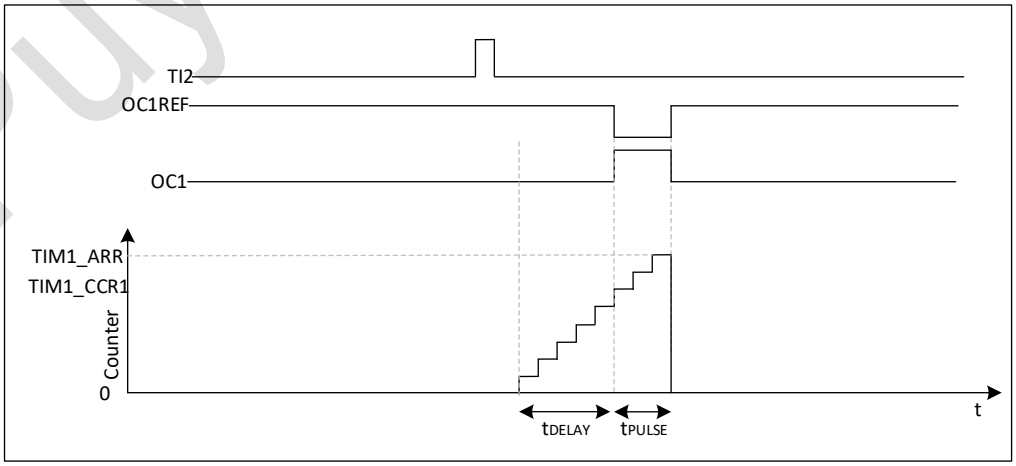


图 18-43 单脉冲模式示例

例如，当需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发源：

- 使用 TIM1_TISEL 寄存器中的 TI2SEL[3:0]位选择正确的 TI2x 源
- 置 TIM1_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIM1_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 置 TIM1_SMCR 寄存器中的 TS=00110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIM1_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)

- t_{DELAY} 由 TIM1_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIM1_ARR – TIM1_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 TIM1_CCMR1 寄存器的 OC1M=0111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIM1_CCMR1 中的 OC1PE=1 和 TIM1_CR1 寄存器中的 ARPE；然后在 TIM1_CCR1 寄存器中填写比较值，在 TIM1_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，TIM1_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 TIM1_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。TIM1_CR1 寄存器中的 OPM=0 时，即选择重复模式。

18.3.17.1. 特殊情况：OCx 快速使能

在单脉冲模式下，TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时使输出翻转。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果需要以最小延时输出波形，可以将 TIM1_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 OCxREF（和 OCx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用。

18.3.18. 可再触发单脉冲模式(OPM)

该模式允许计数器在一个激励信号的触发下启动，并产生长度可编程的脉冲，与上节所述的不可再触发单脉冲模式有以下区别：

- 触发一发生，脉冲立即开始（无可编程延迟）。
- 如果在前一个触发所产生的脉冲完成之前发生新的触发，则延长脉冲。

要使用可重触发的单脉冲模式，计时器必须处于从模式，TIM1_SMCR 寄存器中的位 SMS[3:0]=“1000”（组合模式+触发模式），OCxM[3:0]位设置为“1000”或“1001”（可重新触发 OPM 模式 1 或 2）。

如果此时计时器配置为递增计数模式，则相应的 CCRx 必须设置为 0（ARR 寄存器设置脉冲长度）。如果计时器配置为递减计数模式，CCRx 必须大于或等于 ARR。

注：

1. 出于兼容性原因，OCxM[3:0]和 SMS[3:0]位字段被分成两部分，最高有效位与 3 个最低有效位位置不连续。

2. 该模式不得与中心对齐 PWM 模式一起使用。TIM1_CR1 中必须有 CMS[1:0]=00。

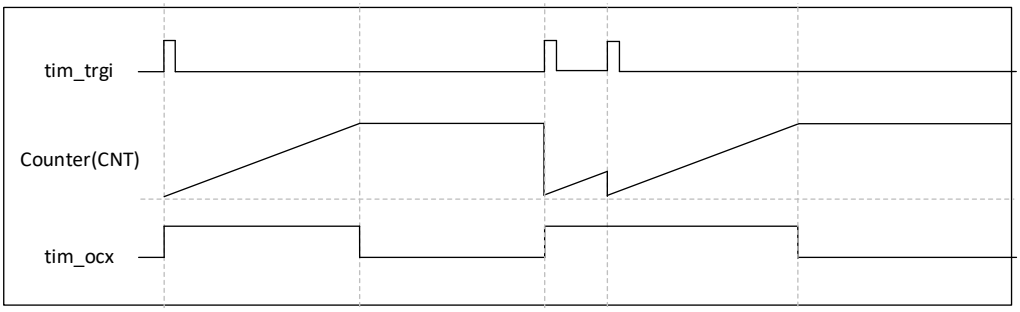


图 18-44 可再触发单脉冲模式

18.3.19. 编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI1 的边沿计数，则置 TIM1_SMCR 寄存器中的 SMS=0001；如果只在 TI2 边沿计数，则置 SMS=0010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=0011。

通过设置 TIM1_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。CC1NP 和 CC2NP 必须保持低电平。

两个输入 TI1 和 TI2 被用来作为正交编码器的接口。参看下表，假定计数器已经启动(TIM1_CR1 寄存器中的 CEN=1)，则计数器的时钟由 TI1FP1 或 TI2FP2 上的每次有效信号转换驱动。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转换序列，产生了计数脉冲和方向信号。依据两个输入信号的转换序列，计数器递增或递减计数，同时硬件对 TIM1_CR1 寄存器的 DIR 位进行相应的设置。任何输入（ TI1 或 TI2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 TI1 或 TI2 边沿处计数，还是同时在 TI1 和 TI2 处计数。

编码器接口模式相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIM1_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIM1_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

注：使能编码器模式时，预分频器必须设置为零。

在这个模式下，计数器根据正交编码器的速度和方向自动进行修改，因此计数器的内容始终指示编码器的位置。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 18-1 计数方向与编码器信号的关系

有效边沿	相反信号的电平 (TI1FP1 对应 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

一个正交编码器可以直接与 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分输出转换位数字信号，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。在这个例子中，我们假定配置如下：

- CC1S='01'(TIM1_CCMR1 寄存器，TI1FP1 映射到 TI1)
- CC2S='01'(TIM1_CCMR1 寄存器，TI1FP2 映射到 TI2)
- CC1P='0'，CC1NP='0'(TIM1_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1)
- CC2P='0'，CC2NP='0'(TIM1_CCER 寄存器，TI1FP2 不反相，TI1FP2=TI2)
- SMS='0011'(TIM1_SMCR 寄存器，所有的输入均在上升沿和下降沿有效)
- CEN='1'(TIM1_CR1 寄存器，计数器使能)

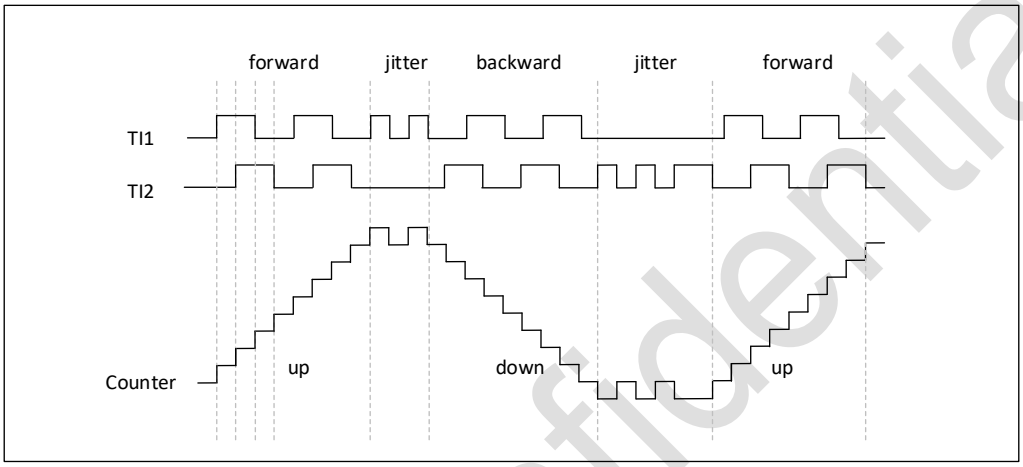


图 18-45 编码器接口模式下的计数器工作示例

下图举例说明 TI1FP1 极性反相时计数器的行为（除 CC1P=“ 1”外，其它配置与上例相同）。

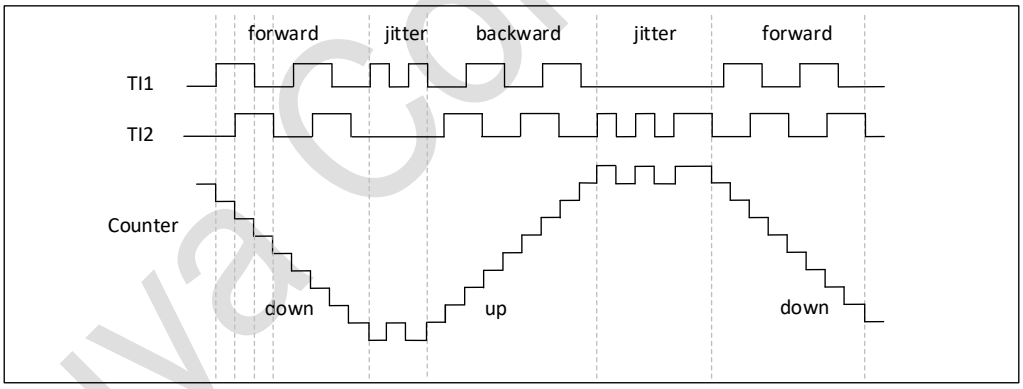


图 18-46 TI1FP1 极性反相时的编码器接口模式

当定时器配置成编码器接口模式时，提供传感器当前的位置信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度、加速度、减速度）。指示机械零点的编码器输出可被用作此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的，并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

18.3.20. 定时器输入异或模式

TIM_CR2 寄存器的 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 TIM1_CH1、TIM1_CH2 和 TIM1_CH3。

异或输出可与触发或输入捕获等所有定时器输入功能配合使用。这样便于测量两个输入信号上边沿之间的间隔。

18.3.21. 连接霍尔传感器

使用高级定时器（TIM1）产生 PWM 信号驱动马达时，可以使用另一个通用定时器（TIM2）作为“接口定时器”来连接霍尔传感器。3 个定时器输入脚（TIM2_CH1、TIM2_CH2 和 TIM2_CH3）通过一个异或门连接到 TI1 输入通道（通过设置 TIM2_CR2 寄存器中的 TI1S 位来选择），并由“接口定时器”捕获这个信号。

从模式控制器被配置到复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器重新从 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。捕获值对应于输入上两次变化的间隔时间，可提供与电机转速相关的信息。

“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 TIM1 各个通道的属性，而高级定时器 TIM1 产生 PWM 信号驱动电机。因此接口定时器通道必须编程为一个指定的延迟（输出比较或 PWM 模式）之后产生一个正脉冲，该脉冲通过 TRGO 输出发送到高级定时器 TIM1。

举例：霍尔输入连接到 TIM2 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 TIM1 的 PWM 配置。

- 置 TIM2_CR2 寄存器的 TI1S 位为’1’，三个定时器输入经过异或运算后进入 TI1 输入通道。
- 时基编程：置 TIM2_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，该周期长于传感器上的两次变化的时间间隔。
- 设置通道 1 为捕获模式(选中 TRC)：置 TIM2_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM2 模式，并具有所需延时：置 TIM2_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIM2_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，必须选择正确的 ITR 输入作为触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载(TIM1_CR2 寄存器中 CCPC=1)，同时 COM 事件由触发输入控制(TIM1_CR2 寄存器中 CCUS=1)。在一次 COM 事件后， PWM 控制位(CCxE、OCxM) 写入下一步的配置，这可以在处理 OC2REF 上升沿产生的中断子程序里实现。

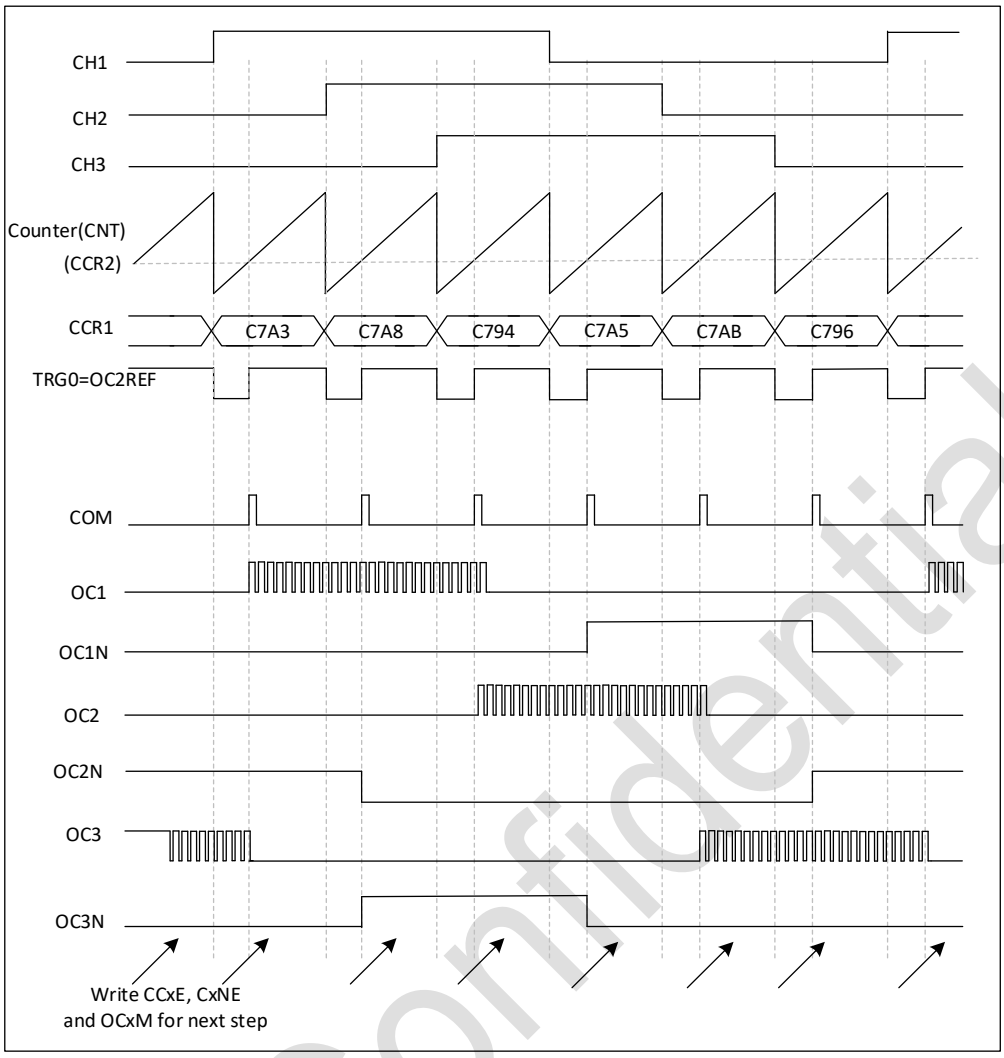


图 18-47 霍尔传感器示例

18.3.22. 定时器和外部触发同步

TIM1 定时器能够在多种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。

18.3.22.1. 从模式：复位模式

在发生一个触发输入事件时，计数器及其预分频器能够重新被初始化；同时，如果 TIM1_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(TIM1_ARR，TIM1_CCRx)都将更新。

在以下的例子中，TI1 输入端的上升沿导致递增计数器被清零：

- 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)。由于捕获预分频器不用于触发操作，所以不需要配置。CC1S 位只选择输入捕获源，即 TIM1_CCMR1 寄存器中 CC1S=01。置 TIM1_CCER 寄存器中 CC1P=0，CC1NP=0 以确定极性(只检测上升沿)。
- 置 TIM1_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM1_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIM1_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(TIM1_SR 寄存器中的 TIF 位)被设置，根据 TIM1_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 TIM1_ARR=0x36 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 输入端的重新同步电路。

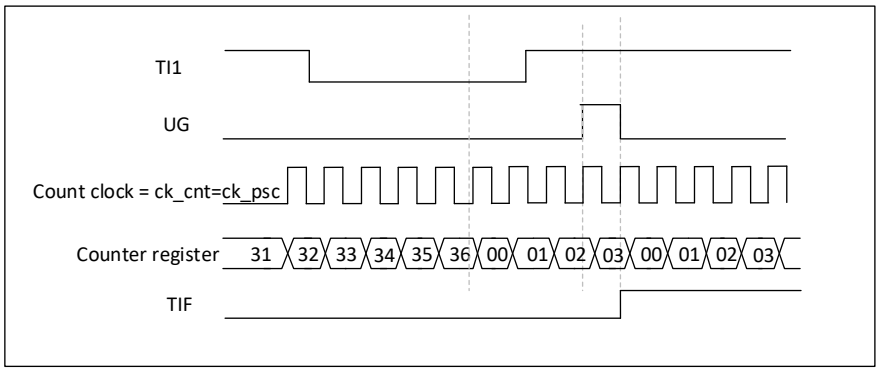


图 18-48 复位模式下的控制电路

18.3.22.2. 从模式：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 为低时递增计数：

- 通道 1 配置为检测 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 TIM1_CCMR1 寄存器中 CC1S=01。置 TIM1_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIM1_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM1_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIM1_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则不论触发输入电平如何，计数器都不能启动。

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 TIM1_SR 中的 TIF 标志。

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。

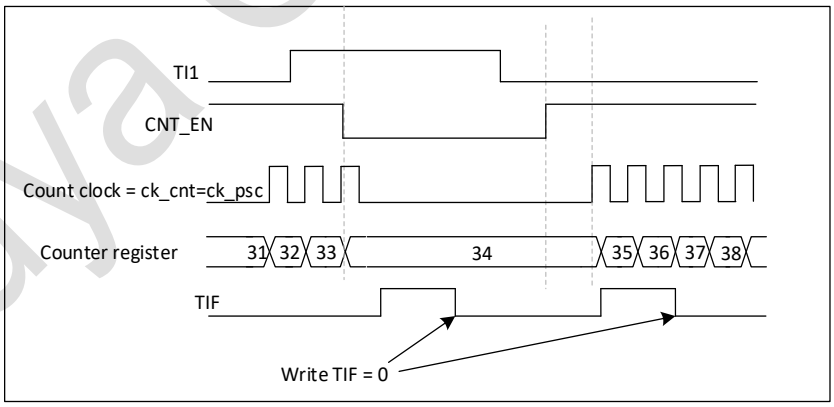


图 18-49 门控模式下的控制电路

18.3.22.3. 从模式：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 输入的上升沿开始递增计数：

- 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)。捕获预分频器不用于触发操作，所以不需要配置。CC2S 位只用于选择输入捕获源，置

TIM1_CCMR1 寄存器中 CC2S=01。置 TIM1_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

- 置 TIM1_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM1_SMCR 寄存器中 TS=00110，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 输入端的重新同步电路。

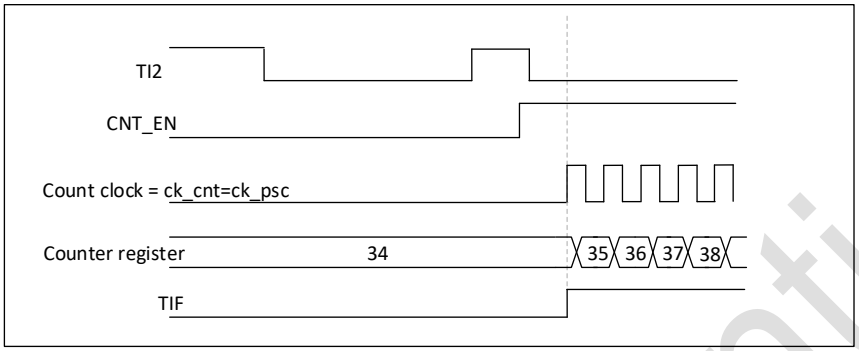


图 18-50 触发模式下的控制电路

18.3.22.4. 从模式：组合复位+触发模式

在这种情况下，在出现所选触发输入(TRGI)上升沿时，重新初始化计数器，生成一个寄存器更新事件，并启动计数器。

该模式用于单脉冲模式。

18.3.22.5. 从模式：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 TIM1_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 的每一个上升沿递增计数一次：

1. 通过 TIM1_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，配置 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
   - 捕获预分频器不用于触发操作中，不需要配置
   - 置 TIM1_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 置 TIM1_CCER 寄存器中 CC1P=0、CC1NP=0 以确定极性(只检测上升沿)
3. 置 TIM1_SMCR 寄存器中 SMS=0110，配置定时器为触发模式。置 TIM1_SMCR 寄存器中 TS=00101，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 输入端的重新同步电路。

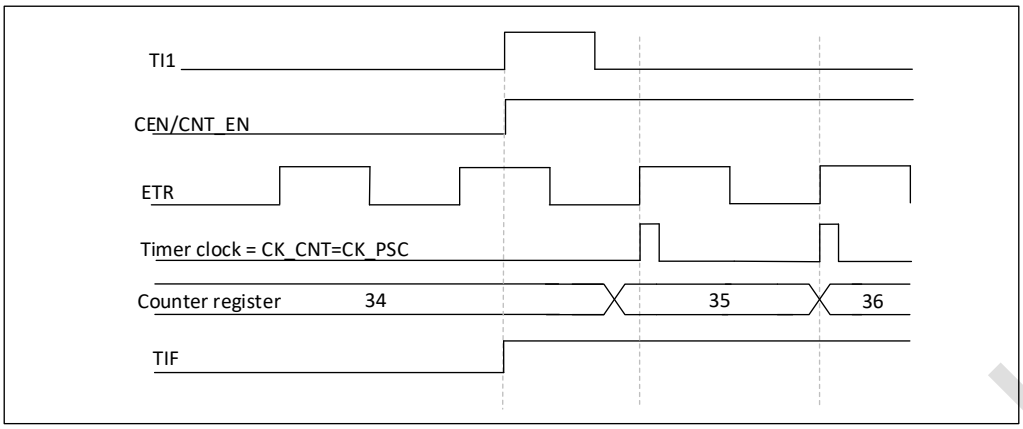


图 18-51 外部时钟模式 2+触发模式下的控制电路

注：必须先使能接收 TRGO 信号的从外设（定时器、 ADC 等）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，不得实时更改时钟频率（预分频器）。

18.3.23. 定时器同步

TIM 定时器在内部相连，用于定时器的同步或者链接功能。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或时钟等操作。

参考 TIM2/TIM3 中描述。

18.3.24. DMA 连续传送模式

TIM1 定时器能够根据一个事件生成多个 DMA 请求。主要目的是能够对定时器的一部分多次重新编程而无需软件开销，但也可用于定期读取一行中的多个寄存器。

DMA 控制器目标唯一，必须指向虚拟寄存器 TIM1_DMAR。发生给定的定时器事件时，定时器会启动 DMA 请求序列（突发）。每次写入 TIM1_DMAR 寄存器都会重定向到其中一个定时器寄存器。

TIM1_DCR 寄存器中的 DBL[4:0]位设置 DMA 连续传送长度。当对 TIM1_DMAR 地址进行读或写访问时，定时器进行一次连续传送，即传送次数（按半字或字节）。

TIM1_DCR 寄存器中的 DBA[4:0]位定义 DMA 传送的 DMA 基址（通过 TIM1_DMAR 地址执行读/写访问时）。DBA 定义为从 TIM1_CR1 寄存器地址开始计算的偏移量：

示例：

00000：TIM1_CR1

00001：TIM1_CR2

00010：TIM1_SMCR

例如，定时器 DMA 连续传送功能用于在发生更新事件后将 CCRx 寄存器（x = 2、3、4）的内容更新为通过 DMA 传输到 CCRx 寄存器中的多个半字。

具体操作步骤如下：

1. 将相应的 DMA 通道配置如下：
   – DMA 通道外设地址为 DMAR 寄存器地址。
   – DMA 通道存储器地址为包含要通过 DMA 传输到 CCRx 寄存器的数据的 RAM 缓冲区地址。
   – 要传输的数据量= 3（参见下文注释）。
   – 禁止循环模式。
2. 通过将 DBA 和 DBL 位域配置如下来配置 DCR 寄存器：

DBL = 3 次传输， DBA = 0xE。

3. 使能 TIM1 更新 DMA 请求（DIER 寄存器中的 UDE 位置 1）。

4. 使能 TIM1

5. 使能 DMA 通道

本例适用于每个 CCRx 寄存器只更新一次的情况。如果每个 CCRx 寄存器要更新两次，则要传输的数据量应为 6。下面以包含 data1、data2、data3、data4、data5 和 data6 的 RAM 缓冲区为例。数据将按照如下方式传输到 CCRx 寄存器：在第一个更新 DMA 请求期间，data1 传输到 CCR2， data2 传输到 CCR3，data3 传输到 CCR4；在第二个更新 DMA 请求期间， data4 传输到 CCR2， data5 传输到 CCR3，data6 传输到 CCR4。

注：可以将空值写入保留的寄存器中。

18.4. TIM1 中断

表 18-2 中断说明

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
比较/捕获 3	CC3IF	CC3IE
比较/捕获 4	CC4IF	CC4IE
COM	COM	COMIE
触发	TIF	TIE
刹车	BIF	BIE
系统刹车	SBIF	BIE

18.5. 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIM1_STOP 的设置，TIM1 计数器可以继续正常工作或者停止工作。

18.6. TIM1 寄存器

18.6.1. TIM1 控制寄存器 1 (TIM1_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	00	时钟分频因子 这 2 位定义定时器时钟(CK_INT)频率与死区发生器以及数字滤波器(ETR,TIx)所用的死区及采样时钟（t_{DTS}）之间的分频比例。

				00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载使能 0：TIM1_ARR 寄存器没有缓冲 1：TIM1_ARR 寄存器进行缓冲
6:5	CMS[1:0]	RW	00	中心对齐模式选择 00：边沿对齐模式。计数器依据方向位(DIR)递增或递减计数。 01：中心对齐模式 1。计数器交替地递增和递减计数。仅当计数器递减计数时，配置为输出的通道 (TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志才置位。 10：中心对齐模式 2。计数器交替地递增和递减计数。仅当计数器递增计数时，配置为输出的通道(TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志才置位。 11：中心对齐模式 3。计数器交替地递增和递减计数。当计数器递增和递减计数时，配置为输出的通道(TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位均被置位。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中心对齐模式。
4	DIR	RW	0	计数方向 0：计数器递增计数 1：计数器递减计数 注：当计数器配置为中心对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0	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	0	更新请求源 软件通过该位选择 UEV 事件源. 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： – 计数器上溢/下溢 – 设置 UG 位 – 从模式控制器产生更新事件

				1：如果允许产生更新中断或 DMA 请求，则只有计数器上溢/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	更新禁止 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： - 计数器上溢/下溢 - 设置 UG 位 - 从模式控制器产生更新事件 影子寄存器装入预装载值。 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，则重新初始化计数器和预分频器。
0	CEN	RW	0	计数器使能 0：禁止计数器 1：使能计数器 注：在软件设置了 CEN 位后，才可以使用外部时钟、门控模式和编码器模式。触发模式可以通过硬件自动地设置 CEN 位为 1。

18.6.2. TIM1 控制寄存器 2 (TIM1_CR2)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
						RW									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	MMS[3]	RW	0	参考 MMS[2:0]描述
24:15	Reserved	-	-	保留
14	OIS4	RW		输出空闲状态 4(OC4 输出)。参见 OIS1 位。
13	OIS3N	RW	0	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位
12	OIS3	RW	0	输出空闲状态 3(OC3 输出)。参见 OIS1 位。
11	OIS2N	RW	0	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。参见 OIS1 位
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出) 0：当 MOE=0 时，(如果 OC1N 有效，则经过死区时间之后) OC1N=0

				1：当 MOE=0 时，(如果 OC1N 有效，则经过死区时间之后) OC1N=1 注：已经设置了 LOCK(TIM1_BDTR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出) 0：当 MOE=0 时，死区后 OC1=0 1：当 MOE=0 时，死区后 OC1=1 注：已经设置了 LOCK(TIM1_BDTR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7	TI1S	RW	0	TI1 选择 0：TIM1_CH1 管脚连到 TI1 输入。 1：TIM1_CH1、TIM1_CH2 和 TIM1_CH3 管脚经异或后连到 TI1 输入。
6:4	MMS[2:0]	RW	3’b0	主模式选择 这些位可选择主模式下将要发送到从定时器以实现同步的信息(TRGO)。可能的组合如下： 0000：复位 – TIM1_EGR 寄存器的 UG 位作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位会有一个延迟。 0001：使能 – 计数器使能信号 CNT_EN 作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或在一段时间内控制从定时器。当配置为门控模式时，计数器使能是通过 CEN 控制位和触发输入信号的逻辑**与**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 TIM1_SMCR 寄存器中 MSM 位的描述)。 0010：更新 – 更新事件作为触发输出(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。 0011：比较脉冲 – 一旦发生一次捕获或一次比较成功时，当 CC1IF 标志被置为 1 时，触发输出送出一个正脉冲(TRGO)。 0100：比较 – OC1REF 信号作为触发输出(TRGO)。 0101：比较 – OC2REF 信号作为触发输出(TRGO)。 0110：比较 – OC3REF 信号作为触发输出(TRGO)。 0111：比较 – OC4REF 信号作为作为触发输出(TRGO)。 1000：编码器时钟输出 -编码器时钟信号作为触发输出，仅当 SMS[3:0]的值为 0001、0010 和 0011 时可用.
3	CCDS	RW	0	捕获/比较的 DMA 选择 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	0	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COMG 位更新。 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COMG 位或 TRGI 上升沿更新。 注：该位只对具有互补输出的通道起作用。
1	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位 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，仅当发生换向事件（COM）（COMG 位置 1 或在 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。 注：该位只对具有互补输出的通道起作用。

18.6.3. TIM1 从模式控制寄存器（TIM1_SMCR)

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[4:3]		Res.	Res.	Res.	SMS[3]
										RW	RW				RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
21:20	TS[4:3]	RW	3’b0	详见 TS 描述
19:17	Reserved	-	-	保留
16	SMS[3]	RW	0	详见 SMS 描述
15	ETP	RW	0	外部触发极性 该位选择是 ETR 或者 ETR 的反相用作触发操作。 0：ETR 不进行反相，高电平或者上升沿有效 1：ETR 反相，低电平或者下降沿有效
14	ECE	RW	0	外部时钟使能。这位使能外部时钟模式 2。 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2，计数器时钟由 ETRF 信号的有效沿提供； 注：

				1：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=00111）具有相同效果。 2：外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 TRGI 不得连接 ETRF（TS 位不得为 00111）。 3：如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。
13:12	ETPS[1:0]	RW	2’b0	外部触发预分频器 外部触发信号 ETRP 频率不能超过 TIM1CLK 频率的 1/4。可以通过使能预分频器来降低 ETR 的频率。该方法在输入快速外部时钟时非常有用。 00：预分频器关闭 01：ETR 频率的 2 分频 10：ETR 频率的 4 分频 11：ETR 频率的 8 分频
11:8	ETF[3:0]	RW	4’b0	外部触发滤波 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 个事件后才视为一个有效输出边沿。 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}= f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}= f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}= f_{DTS}/16$, N=5 1011：采样频率 $f_{SAMPLING}= f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	0	主/从模式 0：不执行任何操作

				1：触发输入(TRGI)上的事件被延迟，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。 此设置适用于由单个外部事件对多个定时器进行同步的情况。
6:4	TS[2:0]	RW	3’b0	触发选择 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测（TI1F_ED） 00101：滤波后的定时器输入 1（TI1FP1） 00110：滤波后的定时器输入 2（TI1FP2） 00111：外部触发输入（ETRF） 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 其它：保留 注：这些位只能在未使用(如 SMS=0000)时更改，以避免在转换时产生错误的边沿检测。 注：ITRx 的映射参见“外设互联”章节。
3	OCCS	RW	0	OCREF 清零选择位。该位用于选择 OCREF 的清零源。 0：OCREF_CLR_INT 连接到 OCREF_CLR 输入(COMP1 或 COMP2 输出) 1：OCREF_CLR_INT 连接到 ETRF
2:0	SMS[2:0]	RW	3’b0	从模式选择。 当选择了外部信号，触发信号(TRGI)的有效边沿与外部输入上所选的极性相关(见输入控制寄存器和控制寄存器的说明)。 0000：禁止从模式---如果 CEN=1，则预分频器直接由内部时钟驱动。 0001: 编码器模式 1---根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿递增/递减计数。 0010: 编码器模式 2---根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿递增/递减计数。 0011：编码器模式 3---根据其他输入的电平，计数器在 TI1FP1 和 TI2FP2 的边沿向递增/递减计数。

				0100：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。 0101：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。 0110：触发模式---计数器在触发输入 TRGI 的上升沿启动(但不复位)，只控制计数器的启动。 0111：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=00100)时，则不得使用门控模式。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。 若 SMS[3]=1，SMS[2:0]必须配置为 0. 1000：“组合复位+触发”模式 ---所选触发输入(TRGI)的上升沿重新初始化计数器，生成寄存器更新并启动计数器。 注：在编码器模式下，不要使用 UEV 作为 TRGO 输出信号（即 MMS 不能配置为 010）。

18.6.4. TIM1 DMA/中断使能寄存器 (TIM1_DIER)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	TDE	RW	0	触发 DMA 请求使能 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	COM 的 DMA 请求使能 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
12	CC4DE	RW	0	捕获/比较 4 的 DMA 请求使能

				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	0	捕获/比较 3 的 DMA 请求使能 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	0	捕获/比较 2 的 DMA 请求使能 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	0	捕获/比较 1 的 DMA 请求使能 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	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	刹车中断使能 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	触发中断使能 0：禁止触发中断 1：允许触发中断
5	COMIE	RW	0	COM 中断使能 0：禁止 COM 中断 1：允许 COM 中断
4	CC4IE	RW	0	捕获/比较 4 中断使能 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	RW	0	捕获/比较 3 中断使能 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	捕获/比较 2 中断使能 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	捕获/比较 1 中断使能 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

18.6.5. TIM1 状态寄存器 (TIM1_SR)

偏移地址:0x010

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Res.	Res.	Res.	Res.	IC2IR	IC1IR	Res.	Res.
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0					RC_W0	RC_W0		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	SBIF	CC4OF	CC3OF	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1F	UIF
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
28	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
27	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
26	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获且由下降沿触发捕获事件时，该标志由硬件置 1。它由软件清零或通过读 TIM1_CCR1 清零。 0：无下降沿捕获产生； 1：发生下降沿捕获事件。
25	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
24	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
23:20	Reserved	-	-	保留
19	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
18	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件时，该标志由硬件置 1。它由软件清零或通过读 TIM1_CCR1 清零。 0：无上升沿捕获产生； 1：发生上升沿捕获事件。
17:14	Reserved	-	-	保留
13	SBIF	RC_W0	0	系统刹车中断标志 系统刹车输入一旦有效，该标志信号会被硬件置位。系统刹车输入无效后可通过软件清零。 该标志必须复位以使 PWM 重新开始工作。 0：未发生刹车事件。 1：系统刹车输入上检测到有效电平。如果 TIM1_DIER 寄存器的 BIE 位=1 则会产生中断。
12	CC4OF	RC_W0	0	捕获/比较 4 重复捕获标志

				参见 CC1OF 描述
11	CC3OF	RC_W0	0	捕获/比较 3 重复捕获标志 参见 CC1OF 描述
10	CC2OF	RC_W0	0	捕获/比较 2 重复捕获标志 参见 CC1OF 描述
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标志 仅当相应的通道被配置为输入捕获时，该标志可由硬件置 1。软件写 0 可清除该位。 0：未产生重复捕获； 1：TIM1_CCR1 寄存器已经捕获到计数器的值且 CC1IF 已经置位。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标志 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：未产生刹车事件； 1：刹车输入上检测到有效电平。如果 TIM1_DIER 寄存器中 BIE=1，则会生成中断。
6	TIF	RC_W0	0	触发中断标志 除门控模式外的其它所有模式下，当使能从模式控制器后，在 TRGI 输入端检测到有效边沿时，由硬件对该位置 1。选择门控模式时，该标志将在计数器启动或停止时置 1，但需要通过软件清零。 0：未产生触发事件； 1：触发中断等待响应
5	COMIF	RC_W0	0	COM 中断标志 一旦产生 COM 事件（当 CCxE、 CCxNE、 OCxM 已更新时）该位由硬件置 1。它由软件清零。 0：未产生 COM 事件； 1：COM 中断等待响应
4	CC4IF	RC_W0	0	捕获/比较 4 中断标志 参考 CC1IF 描述
3	CC3IF	RC_W0	0	捕获/比较 3 中断标志 参考 CC1IF 描述
2	CC2IF	RC_W0	0	捕获/比较 2 中断标志 参考 CC1IF 描述
1	CC1IF	RC_W0	0	捕获/比较 1 中断标志 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，但中心对齐模式除外(参考 TIM1_CR1 寄存器的 CMS 位)。它由软件清零。 0：不匹配；

				1：TIM1_CNT 的值与 TIM1_CCR1 的值匹配。或者当 TIM1_CCR1 的值大于 TIM1_ARR 时，CC1IF 将在计数器递增计数和中心对齐计数的上溢时，或递减计数的下溢时置 1。 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，它由软件清零或通过读 TIM1_CCR1 清零。 0：未产生输入捕获； 1：TIM1_CCR1 寄存器已捕获到计数器值(在 IC1 上已检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标志 当产生更新事件时该位由硬件置 1。它由软件清 0。 0：未产生更新事件； 1：更新事件等待响应。当寄存器被更新时该位由硬件置 1： − 若 TIM1_CR1 寄存器的 UDIS=0，当 REP_CNT=0 时产生更新事件(重复递减计数器上溢或下溢时)； − 若 TIM1_CR1 寄存器的 UDIS=0、URS=0，当 TIM1_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)； − 若 TIM1_CR1 寄存器的 UDIS=0、URS=0，当 CNT 被触发事件重新初始化时产生更新事件。（参考从模式控制寄存器(TIM1_SMCR)）

18.6.6. TIM1 事件产生寄存器（TIM1_EGR)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清零。 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	W	0	产生触发事件

				该位由软件置 1，用于产生一个触发事件，由硬件自动清零。 0：无动作； 1：TIM1_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	W	0	捕获/比较事件，产生控制更新 该位由软件置 1，由硬件自动清零。 0：无动作； 1: 当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对有互补输出的通道有效。
4	CC4G	W	0	产生捕获/比较 4 事件 参考 CC1G 描述
3	CC3G	W	0	产生捕获/比较 3 事件 参考 CC1G 描述
2	CC2G	W	0	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	W	0	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清零。 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**： 使能时， CC1IF 标志置 1 并发送相应的中断或 DMA 请求。 **若通道 CC1 配置为输入**： TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断或 DMA 请求。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 该位由软件置 1，硬件自动清零。 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零(但是预分频系数不变)。若在中心对齐模式下或 DIR=0(递增计数)则计数器被清零；若 DIR=1(递减计数)则计数器自动装载 TIM1_ARR 的值。

18.6.7. TIM1 捕获/比较模式寄存器 1 (TIM1_CCMR1)

偏移地址:0x18

复位值:0x0000 0000

同一寄存器可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获和输出比较模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[3]
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

18.6.7.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F[3:0]	RW	4’b0	输入捕获 2 滤波器。 参见 IC1F[3:0]的描述。
11:10	IC2PSC[1:0]	RW	2’b0	输入/捕获 2 预分频器。 参见 IC1PSC[1:0]的描述。
9:8	CC2S[1:0]	RW	2’b0	捕获/比较 2 选择。 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC2S 仅在通道关闭时(TIM1_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	4’b0	输入捕获 1 滤波器。 该位域定义了 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6

				1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC[1:0]	RW	2’b0	输入/捕获 1 预分频器。 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_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[1:0]	RW	2’b0	CC1S[1:0]：捕获/比较 1 选择。 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注： CC1S 仅在通道关闭时(TIM1_CCER 寄存器的 CC1E=0)才是可写的。

18.6.7.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC2M[3]	RW	0	见 OC2M 描述
23:17	Reserved	-	-	保留
16	OC1M[3]	RW	0	见 OC1M 描述
15	OC2CE	RW	0	输出比较 2 清 0 使能 参见 OC1CE 的描述。
14:12	OC2M[2:0]	RW	3’b0	输出比较 2 模式选择 参见 OC1M 的描述。
11	OC2PE	RW	0	输出比较 2 预装载使能 参见 OC1PE 的描述。
10	OC2FE	RW	0	输出比较 2 快速使能 参见 OC1FE 的描述。
9:8	CC2S[1:0]	RW	2’b0	捕获/比较 2 选择 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC2S 仅在通道关闭时(TIM1_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0	输出比较 1 清零使能 0：OC1REF 不受 ETRF 或者 OCREF_CLR 输入的影响; 1：一旦检测到 ETRF 或者 OCREF_CLR 输入高电平，则清除 OC1REF。
6:4	OC1M[2:0]	RW	3’b0	输出比较 1 模式 该位定义了 OC1、OC1N 的输出参考信号 OC1REF 的行为。OC1REF 高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 0000：冻结。输出比较寄存器 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用; 0001：设置通道 1 为匹配时输出有效电平。当计数器 TIM1_CNT 的值与捕获/比较寄存器 1(TIM1_CCR1)匹配时，强制 OC1REF 为高电平。 0010 ：设置通道 1 为匹配时输出无效电平。当计数器 TIM1_CNT 的值与捕获/比较寄存器 1(TIM1_CCR1)匹配时，强制 OC1REF 为低电平。 0011：翻转。当 TIM1_CCR1=TIM1_CNT 时，OC1REF 发生翻转。 0100：强制为无效电平。强制 OC1REF 为低电平。 0101：强制为有效电平。强制 OC1REF 为高电平。 0110：PWM 模式 1---在递增计数时，一旦 TIM1_CNT<TIM1_CCR1，通道 1 为有效状态，否则为无 效状态；在递减计数时，一旦 TIM1_CNT>TIM1_CCR1，通道 1 为无效状态(OC1REF=0)，否则为有效状态(OC1REF=1)。 0111：PWM 模式 2---在递增计数时，一旦 TIM1_CNT<TIM1_CCR1，通道 1 为无效状态，否则为有效状态；在递减计数时，一旦 TIM1_CNT>TIM1_CCR1，通道 1 为有效状态，否则为无效状态。 1000：可恢复 OPM 模式 1---在递增计数时，通道处于有效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且通道在下一次更新时再次变为有效状态；在递减计数模式下，通道处于无效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且信道在下一次更新时再次变为无效状态。

				1001：可恢复 OPM 模式 2---在递增计数时，通道处于无效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且通道在下一次更新时再次变为无效状态；在递减计数模式下，通道处于有效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且信道在下一次更新时再次变为有效状态。 其他值：保留 注 1：一旦 LOCK 级别设为 3(TIM1_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。 注 3：使用可恢复 OPM 模式时，计数模式不要配置为中心对齐计数模式。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIM1_CCR1 寄存器的预装载功能，可随时写入 TIM1_CCR1 寄存器，写入后将立即使用新值。 1：使能 TIM1_CCR1 寄存器的预装载功能，可读写访问预装载寄存器，TIM1_CCR1 的预装载值在每次生成更新事件时装载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIM1_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：只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（TIM1_CR1 寄存器中的 OPM 位置 1）。其他情况下则无法保证该行为。
2	OC1FE	RW	0	输出比较 1 快速使能 该位用于加快触发器输入事件对 CC 输出的影响。 0：即使触发器开启，CC1 也将根据计数器和 CCR1 的值正常操作。当触发器的输入出现边沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：触发输入上出现有效边沿相当于 CC1 输出上发生比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和 CC1 输出间的延时被缩短为 3 个时钟周期。 只在通道被配置成 PWM1 或 PWM2 模式时 OCxFE 才起作用。
1:0	CC1S[1:0]	RW	2’b0	捕获/比较 1 选择 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC1S 仅在通道关闭时(TIM1_CCER 寄存器的 CC1E=0)才是可写的。

18.6.8. TIM1 捕获/比较模式寄存器 2 (TIM1_CCMR2)

偏移地址:0x1C

复位值:0x0000 0000

参见 TIM_CCMR1 的说明。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC3M[3]
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

18.6.8.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC4F[3:0]	RW	4’b0	输入捕获 4 滤波器。 参见 IC1F 的描述。
11:10	IC4PSC[1:0]	RW	2’b0	输入/捕获 4 预分频器。 参见 IC1PSC 的描述。
9:8	CC4S[1:0]	RW	2’b0	捕获/比较 4 选择。 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC4S 仅在通道关闭时(TIM1_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F[3:0]	RW	4’b0	输入捕获 3 滤波器。 参见 IC1F 的描述。
3:2	IC3PSC[1:0]	RW	2’b0	输入/捕获 3 预分频器。 参见 IC1PSC 的描述。
1:0	CC3S[1:0]	RW	2’b0	捕获/比较 3 选择。 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，IC1 映射在 TRC 上。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC3S 仅在通道关闭时(TIM1_CCER 寄存器的 CC3E=0)才是可写的。

18.6.8.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC4M[3]	RW	0	参见 OC4M 的描述。
23:17	Reserved	-	-	保留
16	OC3M[3]	RW	0	参见 OC3M 的描述。
15	OC4CE	RW	0	输出比较 4 清 0 使能。 参见 OC1CE 的描述。
14:12	OC4M[2:0]	RW	3’b0	输出比较 4 模式。 参见 OC1M 的描述。
11	OC4PE	RW	0	输出比较 4 预装载使能。 参见 OC1PE 的描述。
10	OC4FE	RW	0	输出比较 4 快速使能。 参见 OC1FE 的描述。
9:8	CC4S[1:0]	RW	2’b0	捕获/比较 4 选择。 该位定义通道的方向（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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC4S 仅在通道关闭时(TIM1_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0	输出比较 3 清 0 使能。 参见 OC1CE 的描述。
6:4	OC3M[2:0]	RW	3’b0	输出比较 3 模式。 参见 OC1M 的描述。
3	OC3PE	RW	0	输出比较 3 预装载使能。 参见 OC1PE 的描述。
2	OC3FE	RW	0	输出比较 3 快速使能。 参见 OC1FE 的描述。
1:0	CC3S[1:0]	RW	2’b0	捕获/比较 3 选择。 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC3S 仅在通道关闭时(TIM1_CCER 寄存器的 CC3E=0)才是可写的。

18.6.9. TIM1 捕获/比较使能寄存器 (TIM1_CCER)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	CC4P	RW	0	输入/捕获 4 输出极性。参考 CC1P 的描述。
12	CC4E	RW	0	输入/捕获 4 输出使能。参考 CC1E 的描述。
11	CC3NP	RW	0	输入/捕获 3 互补输出极性。参考 CC1NP 的描述。
10	CC3NE	RW	0	输入/捕获 3 互补输出使能。参考 CC1NE 的描述。
9	CC3P	RW	0	输入/捕获 3 输出极性。参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能。参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	CC2NE	RW	0	输入/捕获 2 互补输出使能。参考 CC1NE 的描述。
5	CC2P	RW	0	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(TIM1_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	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，其输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，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，CC1E 位的值。

				此位将在具有互补输出的通道上进行预装载。如果 TIM1_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件（COM）时，CC1NE 有效位才会从预装载位获取新值。
1	CC1P	RW	0	输入/捕获 1 输出极性 **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]位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性： 00：非反相/上升沿。TIxFP1 上升沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式、编码器模式）。 01：反相/下降沿。TIxFP1 下降沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：非反相/双沿。TIxFP1 上升和下降沿都有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式）。这个配置不能应用于编码器模式下。 注： 1.对于互补输出通道，这一位是预载的。如果 TIM1_CR2 寄存器中的 CCPC 位置 1，那么 CC1P 的实际有效位只有在 COM 事件发生时才会加载预载值。 注：一旦 LOCK 级别(TIM1_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。
0	CC1E	RW	0	输入/捕获 1 输出使能 **CC1 通道配置为输出**： 0：关闭 - OC1 禁止输出，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N、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、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 TIM1_CCR1 寄存器。 0：禁止捕获 1：使能捕获

				注：对于互补输出通道，这一位是预载的。如果 TIM1_CR2 寄存器中的 CC1PC 位被设置，那么 CC1E 的实际有效位只有在 COM 事件发生时才会加载预载值。

表 18-3 具有刹车功能的互补通道 OCx 和 OCxN 的输出控制位

控制位					输出状态	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
1	x	x	0	0	输出禁止(不由定时器驱动)，OCx=0, OCx_EN=0	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
		0	0	1	禁止输出（不由定时器驱动)，OCx=0，OCx_EN=0	OCxREF+极性 OCxN=OCxREF 异或 CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性 OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
		x	1	1	OCREF+极性+死区，OCx_EN=1	OCREF 的互补（not OCREF）+ 极性 +死区，OCxN_EN=1
		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 异或 CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性 OCx=OCxREF 异或 CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平），OCxN=CCxNP, OCxN_EN=1
0	0	x	x	x	输出禁止(不由定时器驱动)	
	1		0	0		
	1		0	1	关闭状态（输出使能且为无效电平） 异步：OCx=CCxP, OCx_EN=1，OCxN=CCxNP, OCxN_EN=1（如果触发 BRK）。 随后（仅当触发 BRK 时才有效），若时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，则 OCx=OISx 且 OCxN=OISxN。	
	1		1	0		
	1	x	1	1		

18.6.10. TIM1 计数器寄存器（TIM1_CNT)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	0	计数器值

18.6.11. TIM1 预分频寄存器（TIM1_PSC)

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	0	预分频器值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时（包括计数器通过 TIM1_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）要装载到有效预分频器寄存器的值。

18.6.12. TIM1 自动装载寄存器（TIM1_ARR)

偏移地址:0x2C

复位值:0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16’hFFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

18.6.13. TIM1 重复计数寄存器（TIM1_RCR)

偏移地址:0x30

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	REP[7:0]	RW	8’b0	重复计数器的值 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）；使能更新中断时，则同时会影响产生更新中断的速率。

				每次递减计数器 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 只有在重复更新事件（UIF）发生时才重载 REP 值，因此对 TIM1_RCR 寄存器写入的新值只在下次重复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； - 在中心对齐模式下，PWM 半周期的数目；

18.6.14. TIM1 捕获/比较寄存器 1 (TIM1_CCR1)

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1[15:0]	RW	16'b0	捕获/比较 1 的值 **若 CC1 通道配置为输出**: CCR1 包含了要装载当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIM1_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIM1_CNT 比较的值，并且在 OC1 端口上输出信号。 **若 CC1 通道配置为输入**: CCR1 为上一次输入捕获 1 事件（IC1）发生时的计数器值。此时，只能读取 TIM1_CCR1 寄存器，无法对其进行编程。

18.6.15. TIM1 捕获/比较寄存器 2 (TIM1_CCR2)

偏移地址:0x38

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2[15:0]	RW	16'b0	捕获/比较 2 的值 **若 CC2 通道配置为输出**:

				CCR2 包含了要装载当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIM1_CCMR1 寄存器(OC2PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIM1_CNT 比较的值，并且在 OC2 端口上输出信号。 **若 CC2 通道配置为输入**: CCR2 为上一次输入捕获 2 事件（IC2）发生时的计数器值。此时，只能读取 TIM1_CCR2 寄存器，无法对其进行编程。

18.6.16. TIM1 捕获/比较寄存器 3 (TIM1_CCR3)

偏移地址:0x3C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR3[15:0]	RW	16’b0	捕获/比较 3 的值 **若 CC3 通道配置为输出**: CCR3 包含了要装载当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIM1_CCMR2 寄存器(OC3PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIM1_CNT 比较的值，并且在 OC3 端口上输出信号。 **若 CC3 通道配置为输入**: CCR3 为上一次输入捕获 3 事件（IC3）发生时的计数器值。此时，只能读取 TIM1_CCR3 寄存器，无法对其进行编程。

18.6.17. TIM1 捕获/比较寄存器 4 (TIM1_CCR4)

偏移地址:0x40

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4[15:0]	RW	16'b0	捕获/比较 4 的值 **若 CC4 通道配置为输出**: CCR4 包含了要装载当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIM1_CCMR2 寄存器(OC4PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIM1_CNT 比较的值，并且在 OC4 端口上输出信号。 **若 CC4 通道配置为输入**: CCR4 为上一次输入捕获 4 事件（IC4）发生时的计数器值。此时，只能读取 TIM1_CCR4 寄存器，无法对其进行编程。

18.6.18. TIM1 刹车/死区寄存器 (TIM1_BDTR)

偏移地址:0x44

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。此位由软件置 1，也可根据 AOE 位状态自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态，具体取决于 OSSI 位; 1：如果设置了相应的使能位（TIM1_CCER 寄存器的 CCxE、CCxNE 位），则使能 OC 和 OCN 输出。
14	AOE	RW	0	自动输出使能 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自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 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(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入； 1：使能刹车输入。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
11	OSSR	RW	0	运行模式下“关闭状态”选择 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（捕获/比较使能寄存器(TIM1_CCER)）。 0：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，使能 OC/OCN 输出并输出无效电平。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 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（捕获/比较使能寄存器(TIM1_CCER)）。 0：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其无效电平。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	2’b0	锁定设置 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 TIM1_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、TIM1_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，TIM1_CCER 寄存器的 CCxP/CCxNP 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，TIM1_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 TIM1_BDTR 寄存器，则其内容冻结直至系统复位。
7:0	DTG[7:0]	RW	8'b0	死区发生器设置 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] × T_{dtg}， T_{dtg} = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG[5:0]) ×T_{dtg}，T_{dtg} = 2 × T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 8 × T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 16 ×T_{DTS}； 例：若 T_{DTS} = 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(TIM1_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则这些位不能被修改。

18.6.19. TIM1 输入选择寄存器 (TIM1_TISEL)

偏移地址:0x5C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	TI4SEL[3:0]	RW	4'b0	TI4 输入选择 0000：TIM1_CH4 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
23:20	Reserved	-	-	保留
19:16	TI3SEL[3:0]	RW	4'b0	TI3 输入选择 0000：TIM1_CH3 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
15:12	Reserved	-	-	保留
11:8	TI2SEL[3:0]	RW	4'b0	TI2 输入选择 0000：TIM1_CH2

				0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL[3:0]	RW	4’b0	TI1 输入选择 0000：TIM1_CH1 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

18.6.20. TIM1 备用选项寄存器 1 (TIM1_AF1)

偏移地址:0x60

复位值:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]		Res.	Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW			RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:14	ETRSEL[3:0]	RW	4’b0	外部触发源选择 该位段用于选择 ETR 输入源 0000：TIM1_ETR 0001：COMP1_OUT 0010：COMP2_OUT 0011：保留 0100：保留 0101：ADC_AWD 其它：保留 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
13:12	Reserved	-	-	保留
11	BKCMP2P	RW	0	来源于比较器 2 刹车输入极性控制。 该位选择来源于比较器 2 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 2 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效) 1：比较器 2 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效) 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
10	BKCMP1P	RW	0	来源于比较器 1 刹车输入极性控制。

				该位选择来源于比较器 1 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 1 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 1 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	比较器 2 刹车输入使能。 比较器 2 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：禁止比较器 2 刹车输入 1：使能比较器 2 刹车输入 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
1	BKCMP1E	RW	0	比较器 1 刹车输入使能。 比较器 1 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：禁止比较器 1 刹车输入 1：使能比较器 1 刹车输入 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。
0	BKINE	RW	1	BKIN 刹车输入使能。 比较器 1 刹车 BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

18.6.21. TIM1 备用选项寄存器 2 (TIM1_AF2)

偏移地址:0x64

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OCRSEL[2:0]		
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18:16	OCRSEL[2:0]	RW	3'b0	OCREF 复位源选择 该位段用于选择 OCREF_CLR 输入源。 000：COMP1_OUT 001：COMP2_OUT 其它：保留 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
15:0	Reserved	-	-	保留

18.6.22. TIM1 DMA 控制寄存器（TIM1_DCR)

偏移地址:0x3DC

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL[4:0]	RW	5'b0	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIM1_DMAR 寄存器的地址进行读或写时，定时器进行一次连续传送）。 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 10001：18 次传输
7:5	Reserved	-	-	保留
4:0	DBA[4:0]	RW	0 0000	DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIM1_DMAR 寄存器的地址进行读或写时），DBA 定义为从 TIM1_CR1 寄存器所在地址开始的偏移量： 00000：TIM1_CR1， 00001：TIM1_CR2，

				00010：TIM1_SMCR， .....

18.6.23. TIM1 DMA 连续传输寄存器 (TIM1_DMAR)

偏移地址:0x3E0

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[15:0]	RW	0	DMA 连续传送寄存器 对 TIM1_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIM1_CR1 地址 + (DBA + DMA 指针) x4，其中： “TIM1_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIM1_DCR 寄存器中定义的基地址； “DMA 指针”是由 DMA 自动控制的偏移量，它取决于 TIM1_DCR 寄存器中定义的 DBL。

注：在使用 DMA 连续传输功能时，必须将 DMA 中对应通道的 CNDTR 寄存器的值与 TIM1_DCR 寄存器中 DBL 的值对应起来，否则将不能正常使用 DMA 连续传输功能。

19. 通用定时器(TIM2/TIM3)

19.1. TIM2/TIM3 简介

TIM2 通用定时器是由可编程分频器驱动的 32 位自动重装载计数器构成。

TIM3 通用定时器是由可编程分频器驱动的 16 位自动重装载计数器构成。

除计数宽度不同外，TIM2 和 TIM3 的其他功能完全一致，功能描述中以 TIM2 为例说明。

它适合多种场合，包括测量输出入信号的脉冲长度（输入捕获）或者产生输出波形（输出比较和 PWM）。

使用定时器预分频器和 RCC 时钟控制器预分频器，脉冲长度和波形周期可以在几个微秒到几个毫秒间调整。

19.2. TIM2/TIM3 主要特性

通用 TIM2/TIM3 定时器功能包括：

- 32 位（TIM2）/16 位（TIM3）递增、递减和递增/递减自动装载计数器
- 16 位可编程预分频器，计数器时钟频率的分频系数为 1~65536 之间的任意数值
- 4 个独立通道
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中心对齐模式）
  - 单脉冲模式输出
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路
- 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
- 触发输入作为外部时钟或者逐周期电流管理

19.3. TIM2/TIM3 功能描述

19.3.1. 功能框图

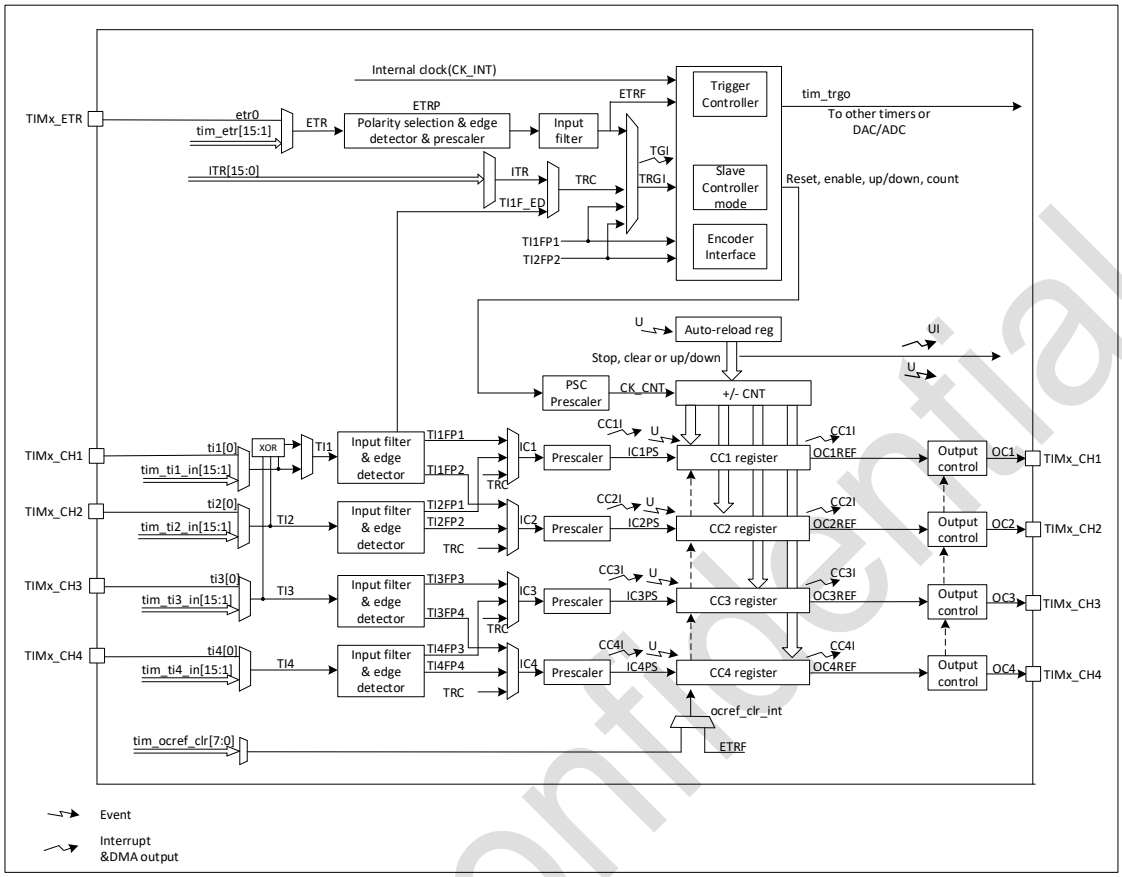


图 19-1 通用定时器框图 (TIM2/TIM3)

19.3.2. 时基单元

可编程通用定时器的主要模块由一个 16 位/32 位计数器及其相关的自动重载寄存器组成。计数器可递增计数、递减计数或交替进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

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。

时基单元包括：

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

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位(CEN)置 1 时，才会启动计数器。

注意，计数器将在 TIMx_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。

19.3.2.1. 预分频器描述

预分频器可以将计数器的时钟按 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。

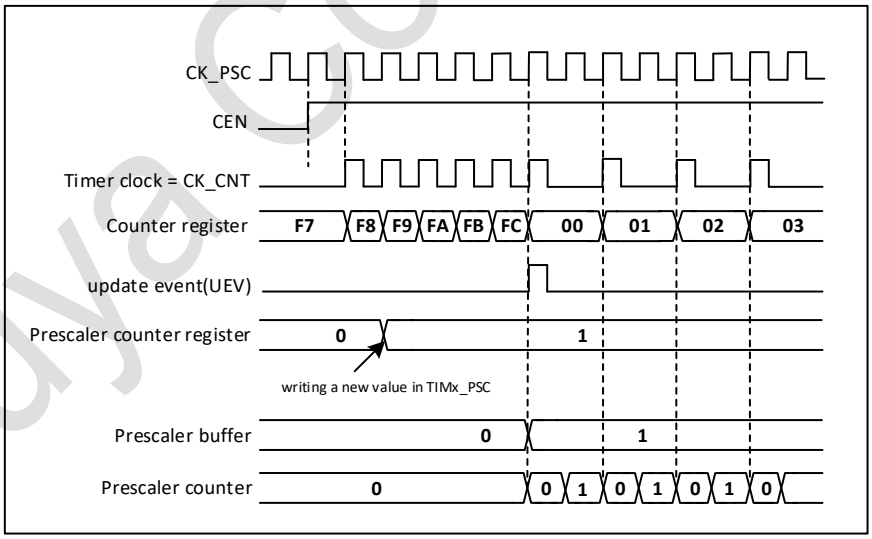


图 19-2 预分频器分频由 1 变为 2 时的计数器时序图

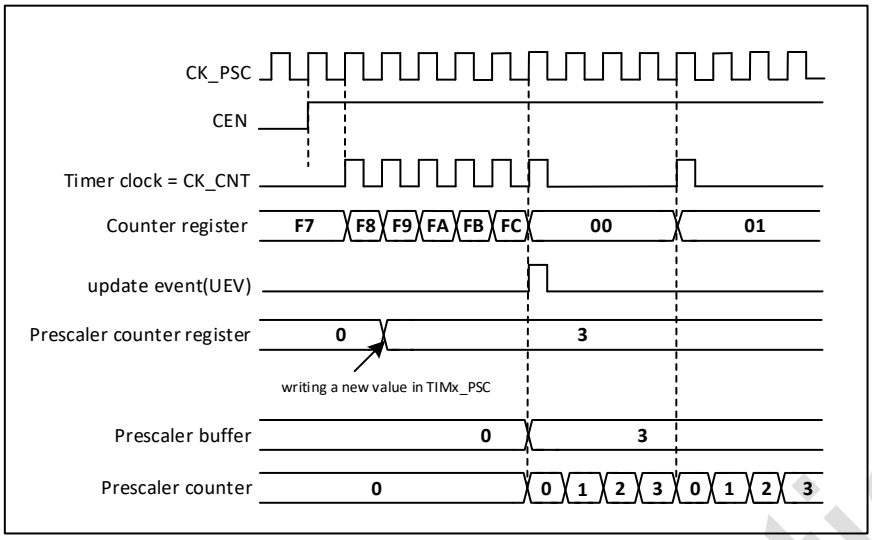


图 19-3 预分频器分频由 1 变为 4 时的计数器时序图

19.3.3. 计数模式

19.3.3.1. 递增计数模式

递增计数模式，是从 0 计数到自动装载值（TIMx_ARR 寄存器），然后又从 0 重新开始计数，并产生一个计数的溢出事件。

在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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）。

- 自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

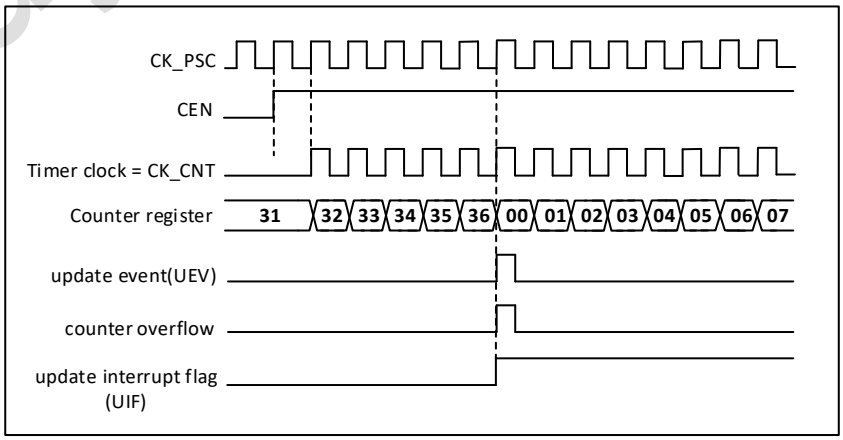


图 19-4 计数器时序图，内部时钟 1 分频

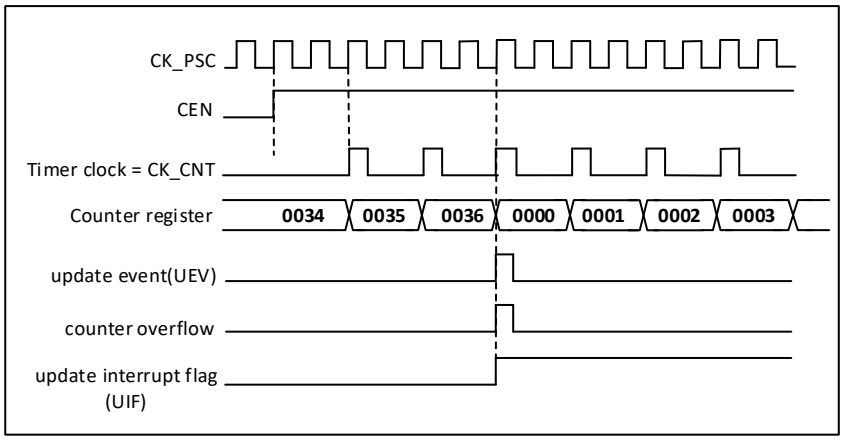


图 19-5 计数器时序图，内部时钟 2 分频

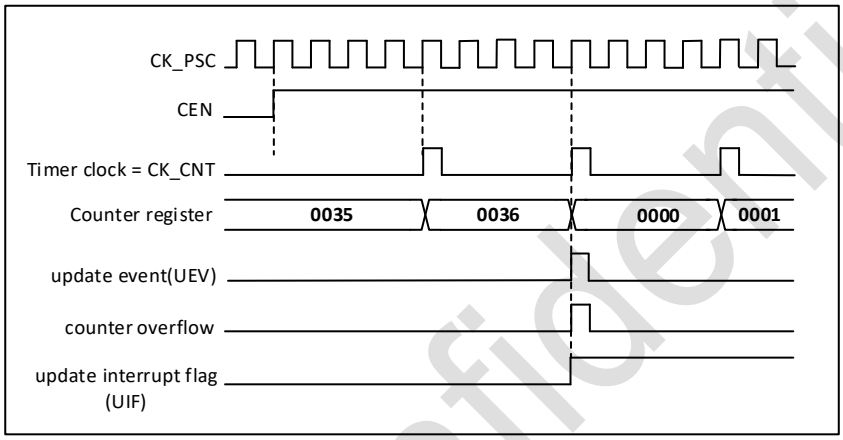


图 19-6 计数器时序图，内部时钟 4 分频

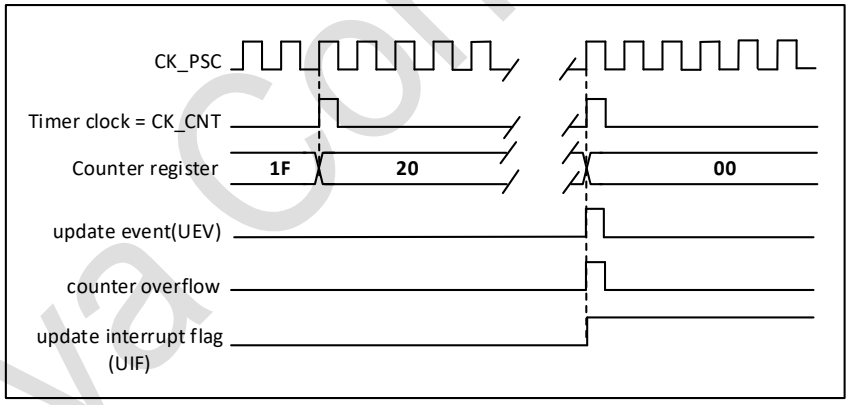


图 19-7 计数器时序图，内部时钟 N 分频

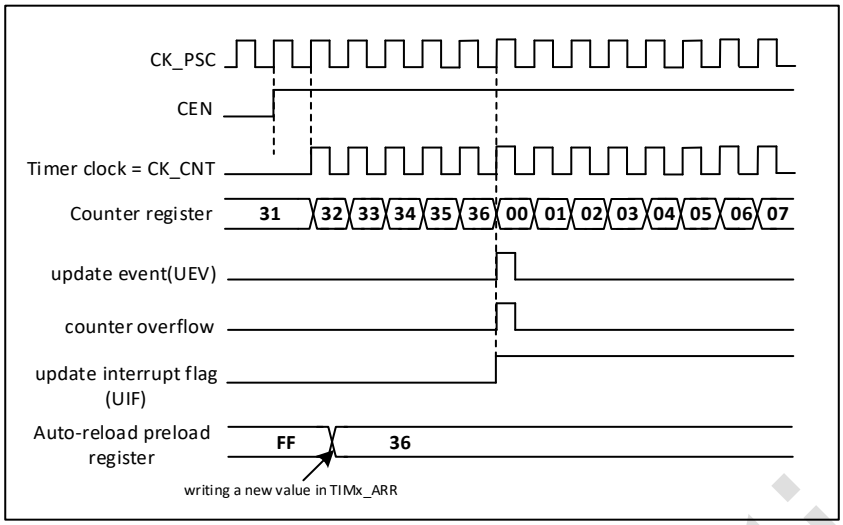


图 19-8 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

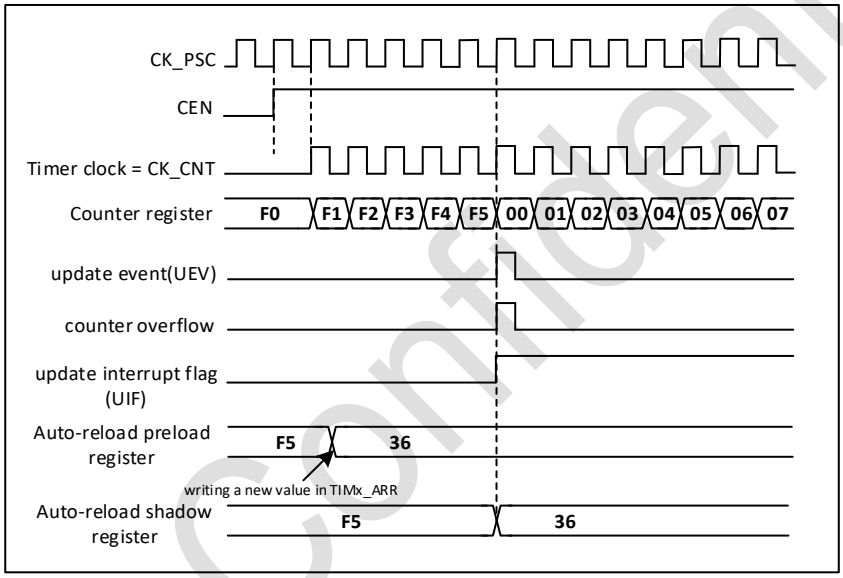


图 19-9 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载）

19.3.3.2. 递减计数模式

在递减计数模式下，计数器从自动重载值（ TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

每次发生计数器下溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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）。

- 预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

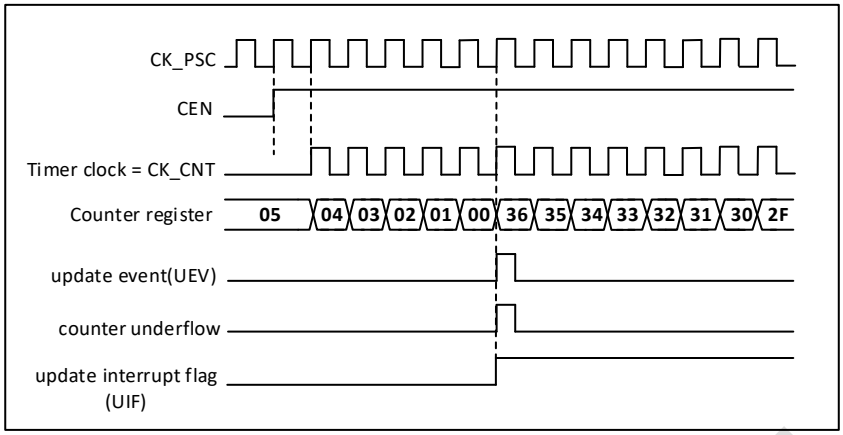


图 19-10 计数器时序图，内部时钟 1 分频

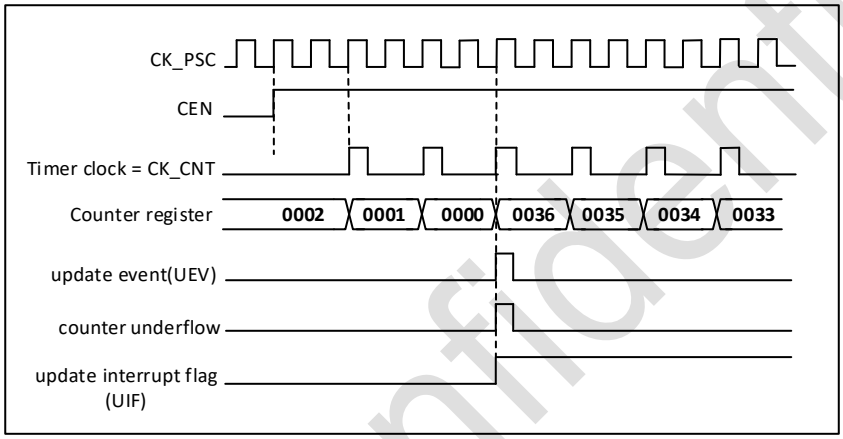


图 19-11 计数器时序图，内部时钟 2 分频

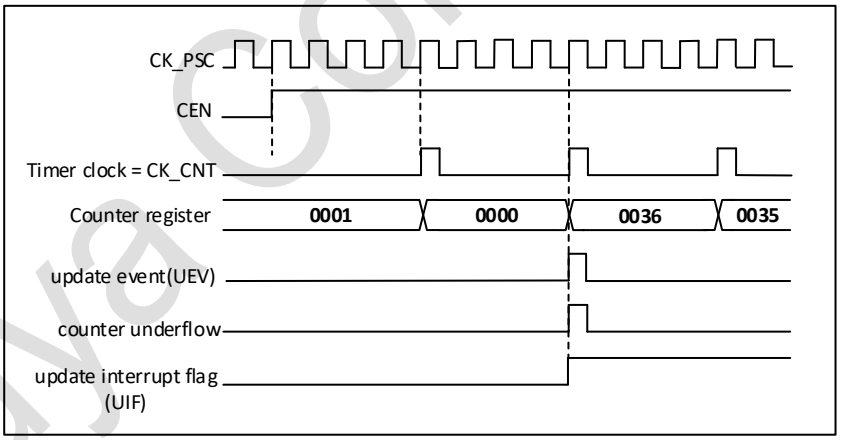


图 19-12 计数器时序图，内部时钟 4 分频

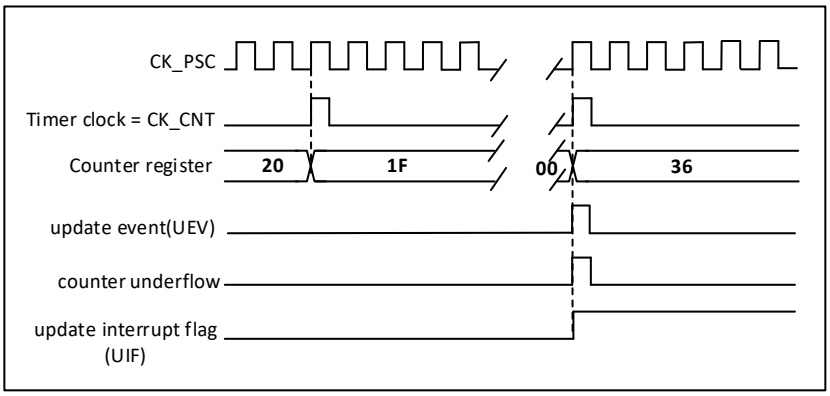


图 19-13 计数器时序图，内部时钟 N 分频

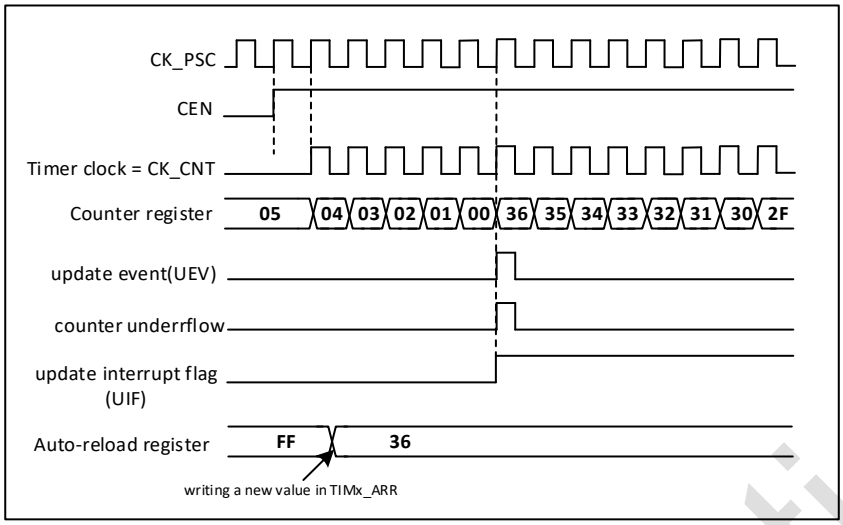


图 19-14 计数器时序图，未使用重复计数器时更新事件

19.3.3.3. 中心对齐模式

在中心对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器溢出事件，然后递减计数到 1，并产生一个计数器下溢事件，然后再从 0 开始重新计数。

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

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续递增或递减计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIM1_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）。

- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR)

注意：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前更新，因此下一个周期才是预期的值(计数器被装载为新的值)。

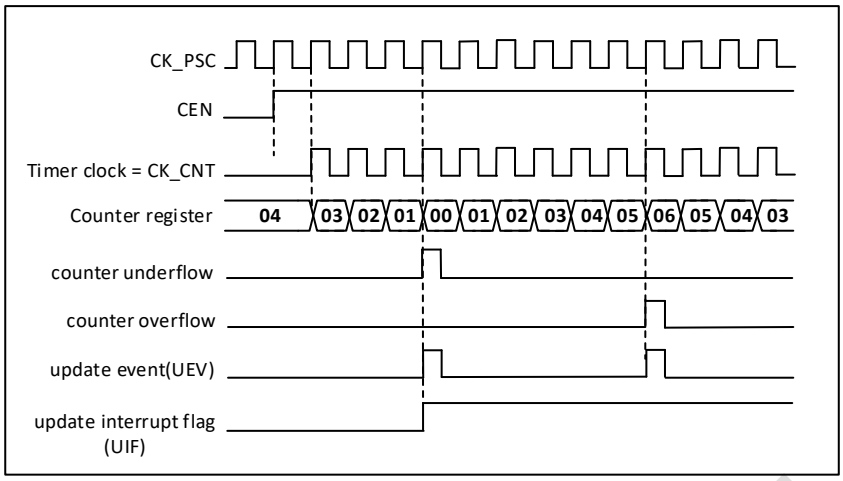


图 19-15 计数器时序图，内部时钟 1 分频, TIM1_ARR = 0x6

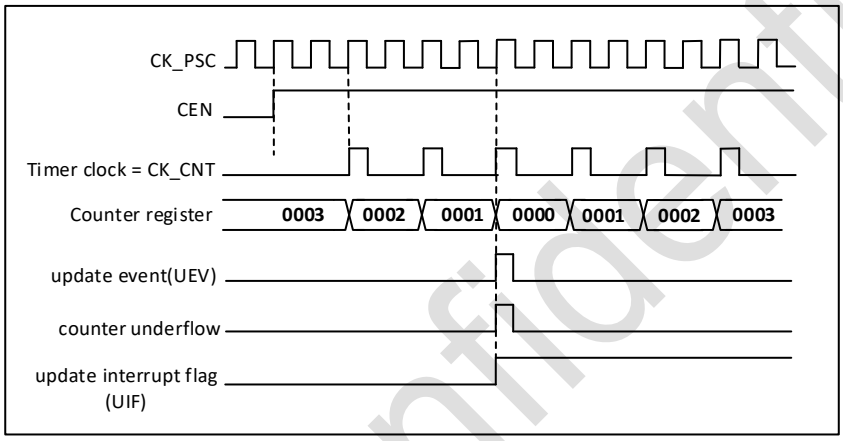


图 19-16 计数器时序图，内部时钟 2 分频

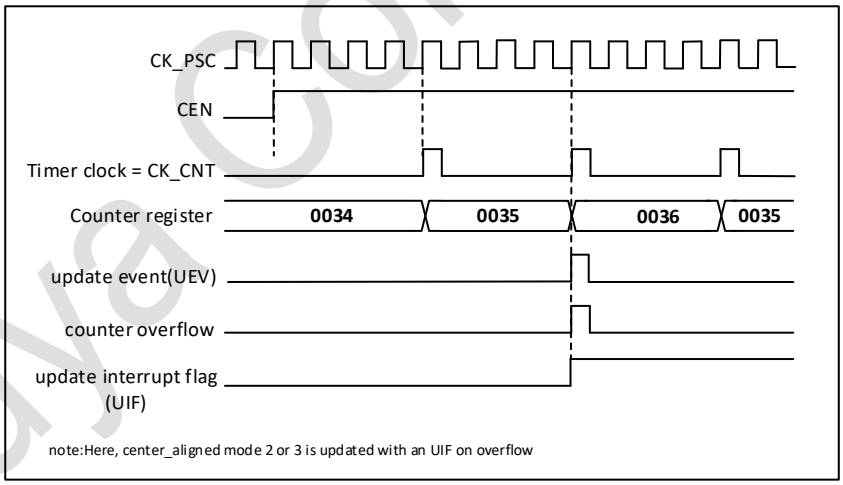


图 19-17 计数器时序图，内部时钟 4 分频, TIM1_ARR=0x36

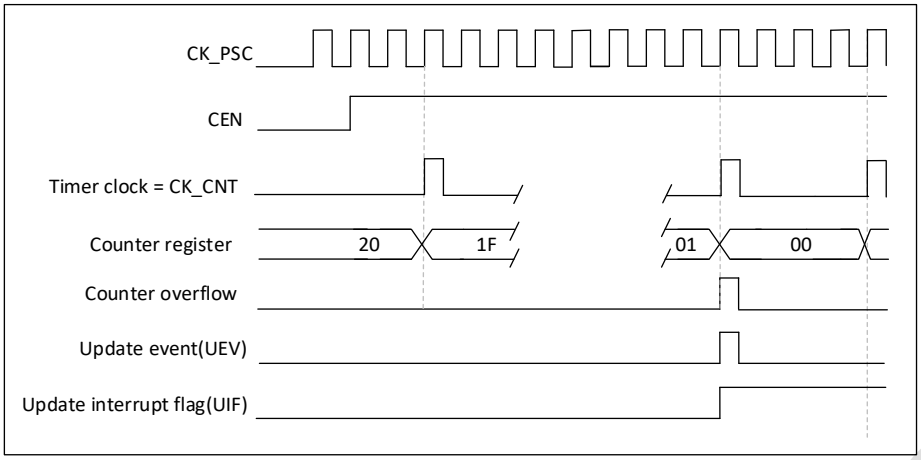


图 19-18 计数器时序图，内部时钟 N 分频

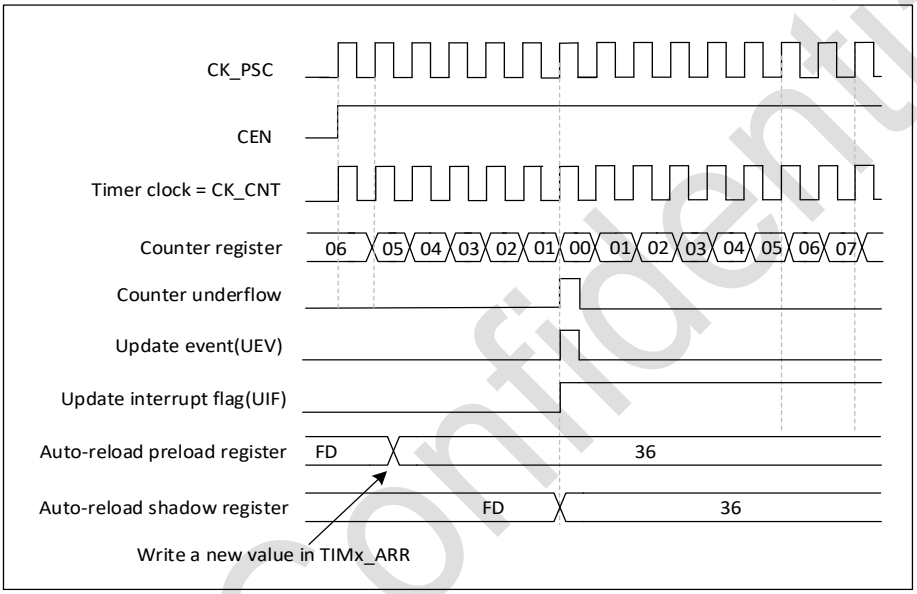


图 19-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

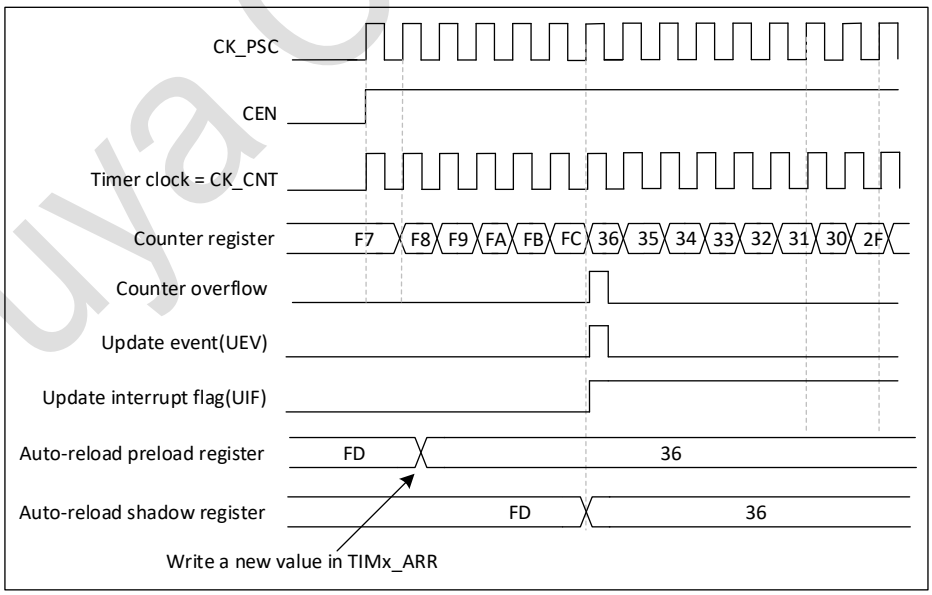


图 19-20 计数器时序图，ARPE=1 时的更新事件（计数器上溢)

19.3.4. 时钟源

计数器的时钟可以由以下时钟源提供：

- 内部时钟（CK_INT）

- 外部时钟模式 1：外部输入引脚（TIx）
- 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。例如，可以配置一个定时器 Timer1 作为另一个定时器 Timer2 的预分频器。

19.3.4.1. 内部时钟源（CK_INT）

如果从模式控制器被禁止（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）为实际控制位，并且只能被软件修改（UG 除外，仍保持自动清零）。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。

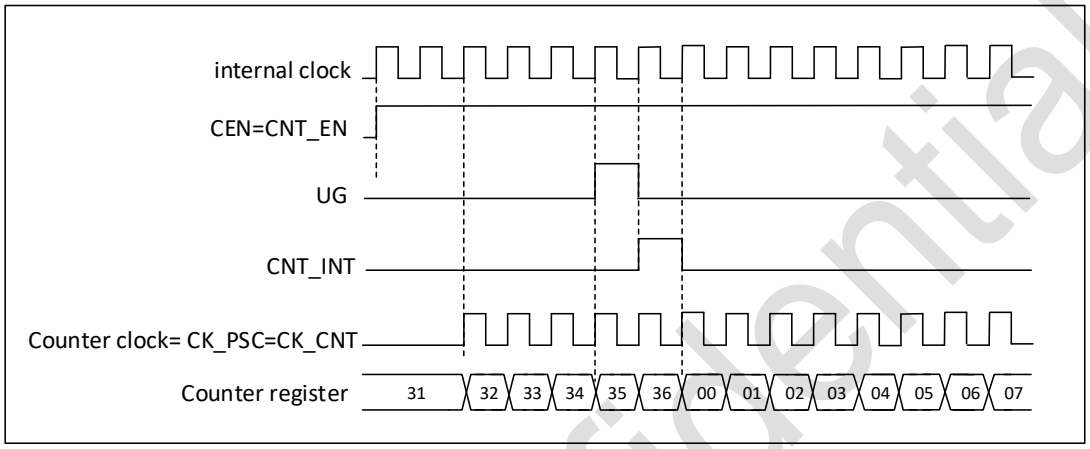


图 19-21 正常模式下的控制电路，内部时钟 1 分频

19.3.4.2. 外部时钟源模式 1

当 TIMx_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

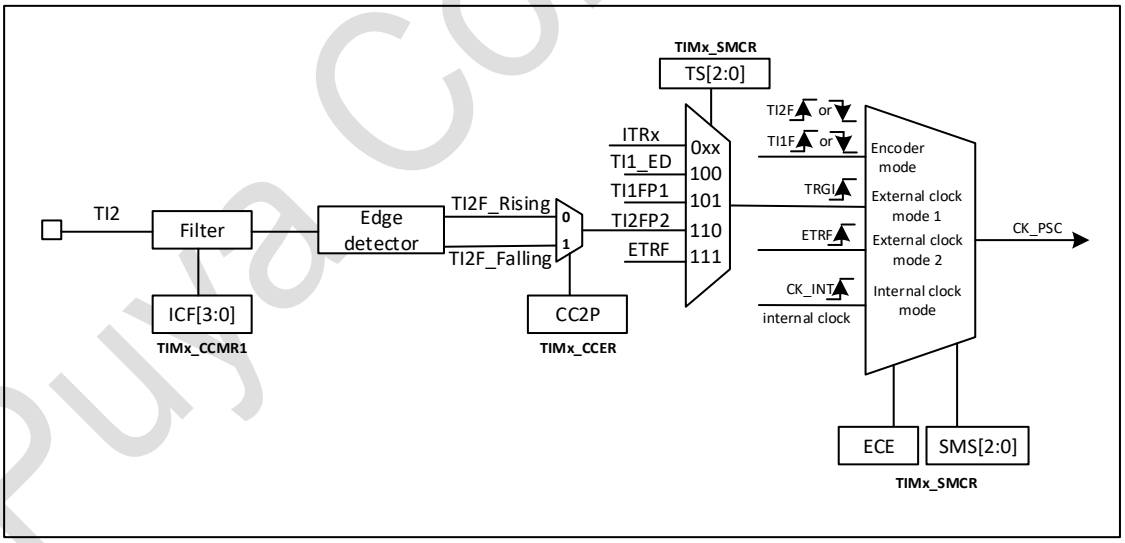


图 19-22 TI2 外部时钟连接示例

例如，要使递增计数器在 TI2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIMx_TISEL 寄存器中配置 TI2SEL[3:0]位选择正确的 TI2x 源（内部或外部）。
2. 配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿
3. 配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)
4. 配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性
5. 配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1

6. 配置 TIMx_SMCR 寄存器中的 TS=00110，选定 TI2 作为触发输入源
7. 设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注： 由于捕获预分频器不用于触发操作，因此用户无需对其进行配置。

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

TI2 的上升沿与实际计数器时钟之间的延迟是由于 TI2 输入的重新同步电路引起的。

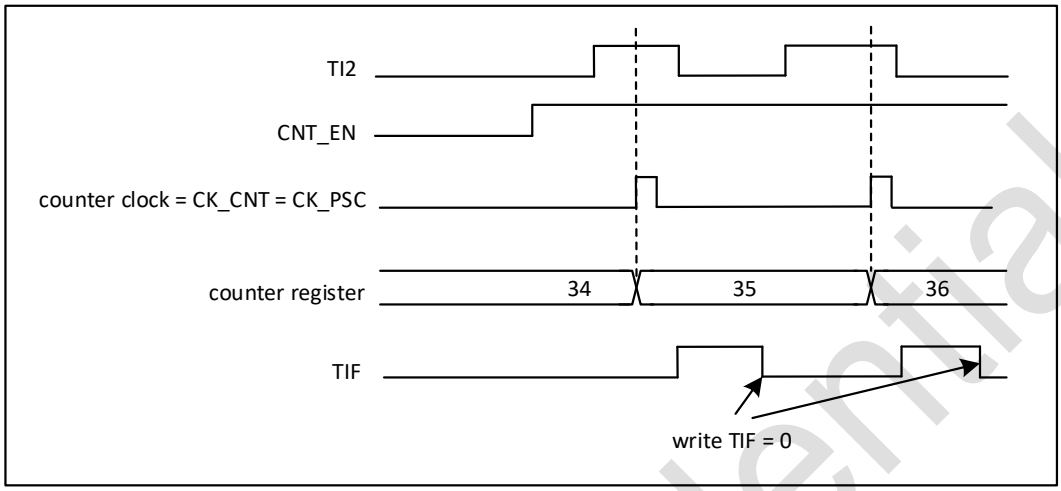


图 19-23 外部时钟模式 1 控制电路时序

19.3.4.3. 外部时钟源模式 2

通过写 TIM1_SMCR 寄存器的 ECE 为 1，选定此模式。计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图：

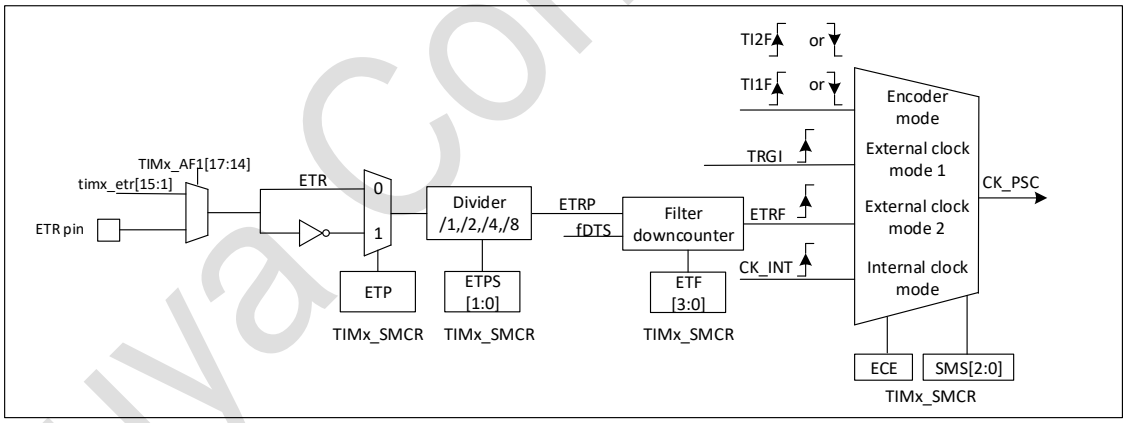


图 19-24 外部触发输入模块

例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIMx_SMCR 寄存器中写入 ETPS[1:0]=01 来设置预分频器。
3. 通过在 TIMx_SMCR 寄存器中写入 ETP=0 来选择 ETR 引脚的上升沿检测。
4. 通过在 TIMx_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
5. 通过在 TIM1x_CR1 寄存器中写入 CEN=1 来使能计数器。

ETR 每出现 2 个上升沿，计数器计数一次。

ETR 的上升沿与实际计数器时钟之间的延迟是由于 ETRP 信号的重新同步电路引起的。

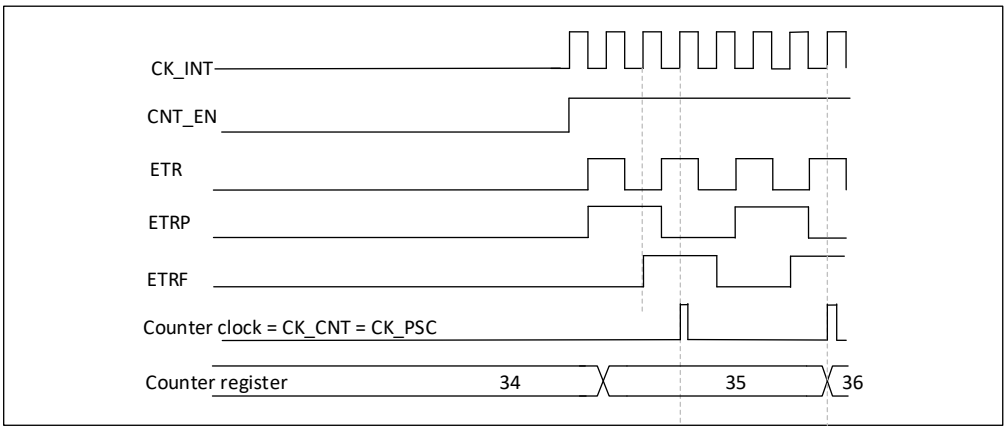


图 19-25 外部时钟模式 2 控制电路时序

19.3.5. 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入部分（数字滤波、多路复用和预分频器）和一个输出部分（比较器和输出控制）构建而成。

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）。

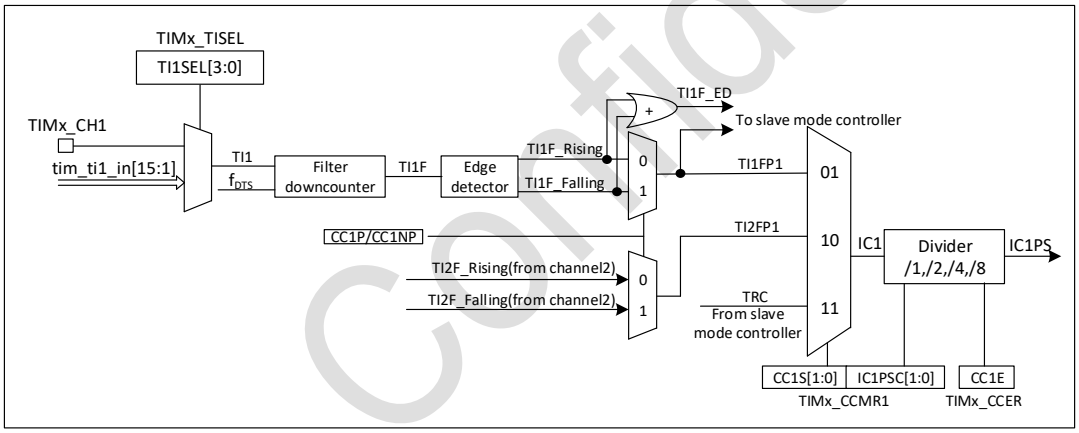


图 19-26 捕获/比较通道(示例：通道 1 输入阶段)

输出部分产生一个中间波形 OCxREF(高有效)作为基准，链的末端决定最终输出信号的极性。

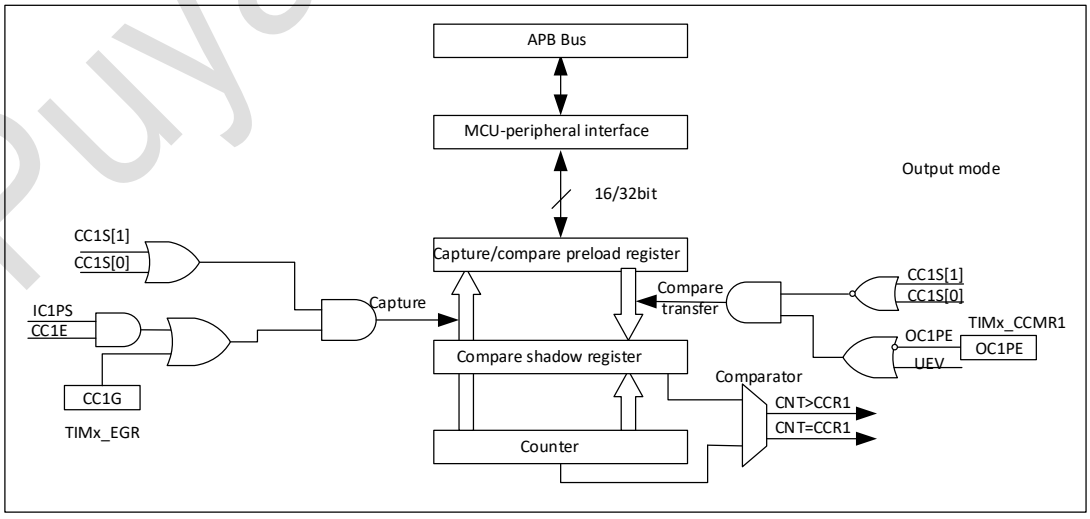


图 19-27 捕获/比较通道 1 主电路

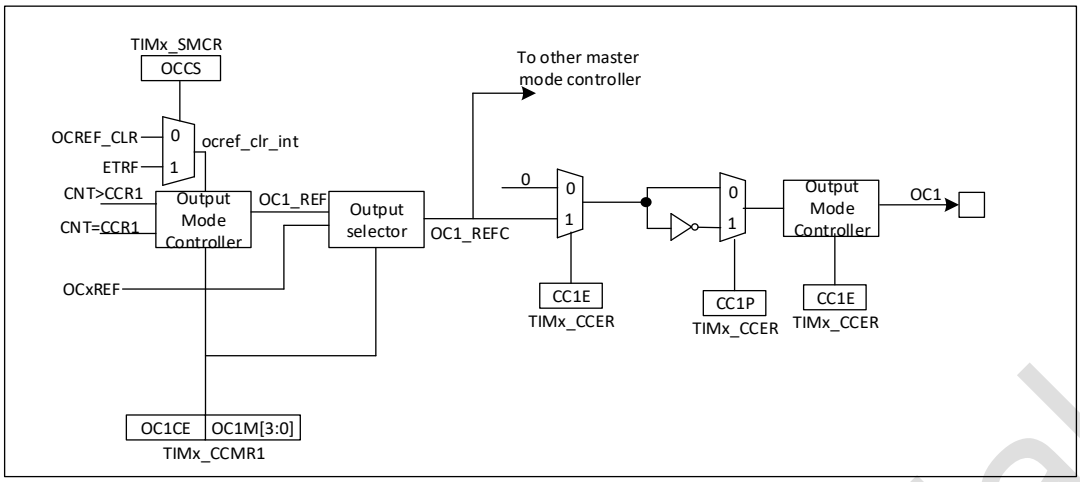


图 19-28 捕获/比较通道的输出阶段 (通道 1)

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。

19.3.6. 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，如果中断和 DMA 操作被打开，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，则重复捕获标志 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：

- 使用 TIMx_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）
- 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 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。
- 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)（和 CC1NP=0）
- 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 IC1PSC=00)。
- 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：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，但 CC1IF 未曾被清除，则 CC1OF 也被置 1。

- 如设置了 CC1IE 位，则会产生一个中断。
- 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。

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。

19.3.7. 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：

- 两个 ICx 信号被映射到同一个 TIx 输入。
- 这两个 ICx 信号为边沿有效，但是极性相反。
- 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，当需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)时，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

- 使用 TIMx_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）。
- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 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)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

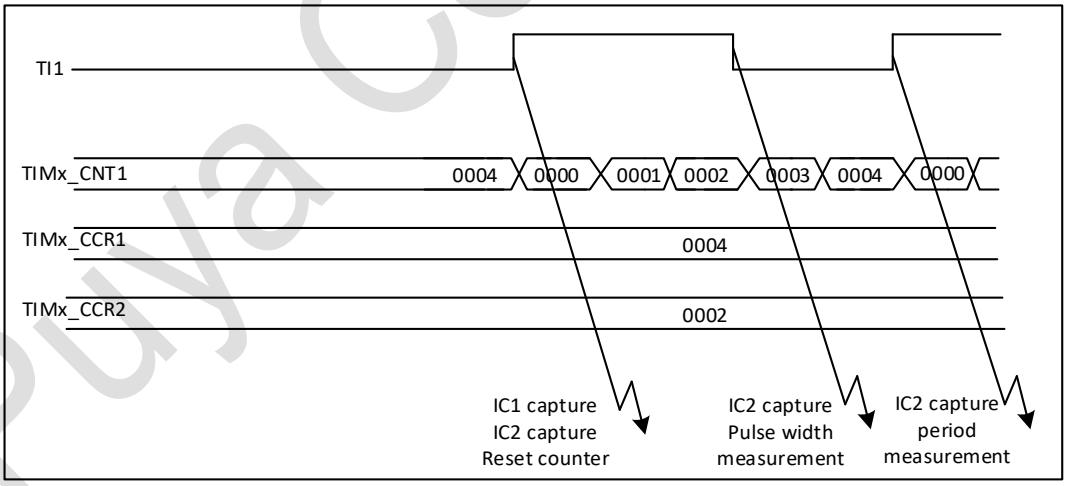


图 19-29 PWM 输入模式时序

19.3.8. 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)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。置 TIMx_CCMRx 寄存器中的 OCxM=0100，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。这将会在下面的输出比较模式一节中介绍。

19.3.9. 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：

- 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=0000)、被设置成有效电平(OCxM=0001)、被设置成无效电平(OCxM=0010)或进行翻转(OCxM=0011)。
- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
- 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。
- 若设置了相应的 DMA 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。时间的精度可以达到计数器的一个计数周期。输出比较模式也能用来输出一个单脉冲(在单脉冲模式下)。

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：
   - 设置 OCxM=0011，则计数器与 CCRx 匹配时翻转 OCx 的输出引脚，
   - 置 OCxPE = 0 禁用预装载寄存器
   - 置 CCxP = 0 选择极性为高电平有效
   - 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进行更新)。下图给出了一个例子。

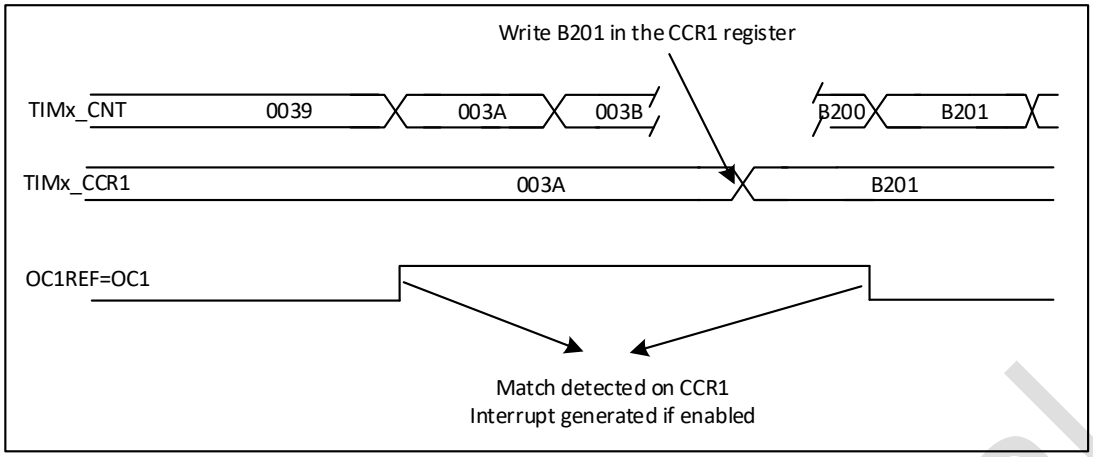


图 19-30 输出比较模式，翻转 OC1

19.3.10. PWM 模式

PWM 脉宽调制模式可以允许产生一个由 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，使能自动重装载的预装载寄存器(在递增计数或中心对称模式中)。

由于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 CCxE、CCxNE、MOE、OSSI 和 OSSR 位(TIMx_CCER 和 TIMx_BDTR 寄存器)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，以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx(依据计数器的计数方向)。

虽然这样，为了满足 OCREF_CLR 功能（OCREF 能被 ETR 信号产生的外部事件清理直到下一个 PWM 周期），OCREF_CLR 仅可以通过如下方式给出：

1．当比较结果改变。

2．通过设置 TIMx_CCMRx 寄存器的 OcxM，输出比较模式由“冻结”配置（OCxM=000）切换到任意 PWM 模式（OCxM=“110”或”111”）。

在定时器运行时，这是通过软件强制切换到 PWM 模式。

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 信号或中心对齐的 PWM 信号。

19.3.10.1. PWM 边沿对齐模式

递增计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行递增计数。参看下面是一个 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=8 时边沿对齐的 PWM 波形实例。

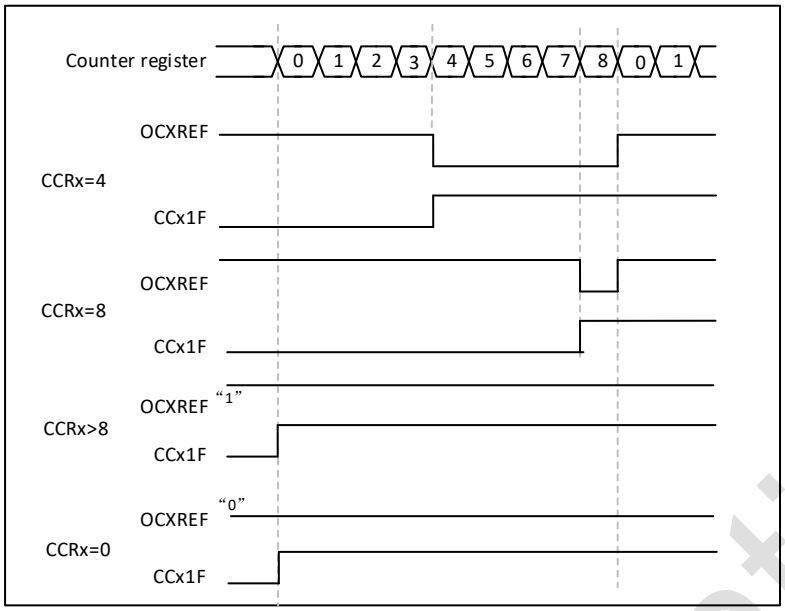


图 19-31 边沿对齐方式 PWM 输出波形（ARR=8）

递减计数配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行递减计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低电平，否则为高电平。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%占空比的 PWM 波形。

19.3.10.2. PWM 中心对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中心对齐模式(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递增计数时被置 1、在计数器递减计数时被置 1、或在计数器递增和递减计数时置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不能用软件修改。

下图给出一些中心对齐的 PWM 波形的例子

- TIMx_ARR = 8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中心对齐模式下，当计数器递减计数时设置比较标志

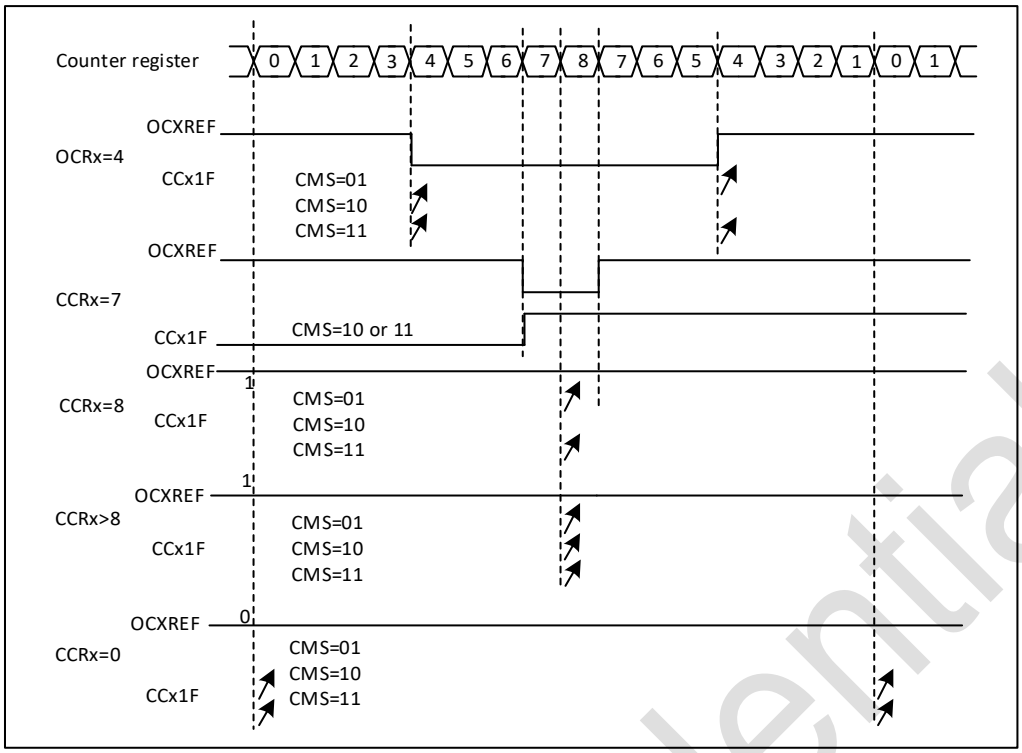


图 19-32 中心对齐模式波形(ARR=8)

中心对齐模式使用建议：

- 进入中心对齐模式时，使用当前的递增/递减计数配置；这就意味着计数器递增还是递减计数取决于 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。
- 不推荐当运行在中心对齐模式时改写计数器，否则会产生不可预知的结果。尤其是：
  - 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，如果计数器正在递增计数，则会继续递增计数。
  - 如果将 0 或者 TIMx_ARR 的值写入计数器，计数方向会更新，但不产生更新事件 UEV。
- 使用中心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

19.3.11. 在外部事件时清除 OCxREF 信号

对于给定通道，在 ocref_clr_int 输入上施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 OCxREF 信号清零。 OCxREF 信号将保持低电平，直到发生下一更新事件(UEV)。该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。

通过配置 TIMx_SMCR 寄存器中的 OCCS 位，可在 OCREF_CLR 输入和 ETRF（滤波器后的 ETR）之间选择 ocref_clr_int。

对于给定通道，在 ETRF 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置 1），可将 OCxREF 信号复位。 OCxREF 信号将保持低电平，直到发生下一更新事件(UEV)。该功能只能在输出比较模式和 PWM 模式下使用。在强制模式下不起作用。

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。

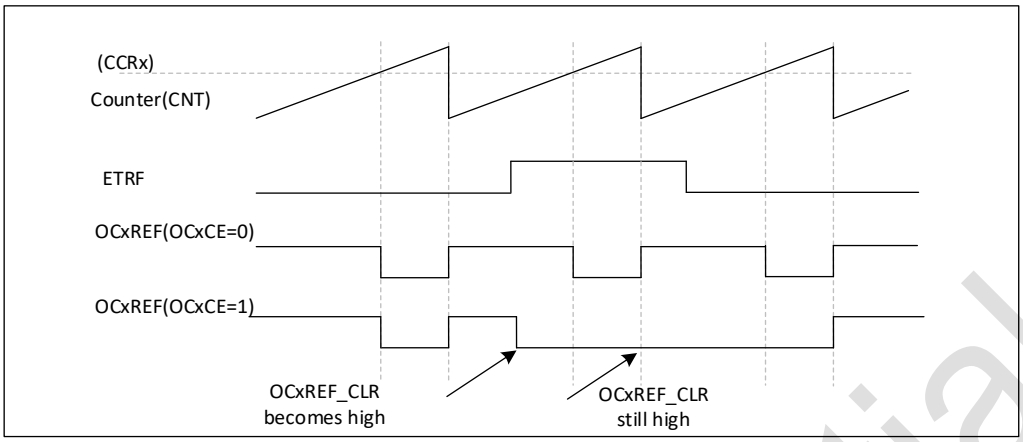


图 19-33 清除 TIMx 的 OCxREF

19.3.12. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。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，产生一个脉宽可被程序控制的脉冲。

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 寄存器的 OPM 位将选择单脉冲模式，这样可以让计数器自动的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：

- 递增计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)
- 递减计数方式：计数器 CNT > CCRx

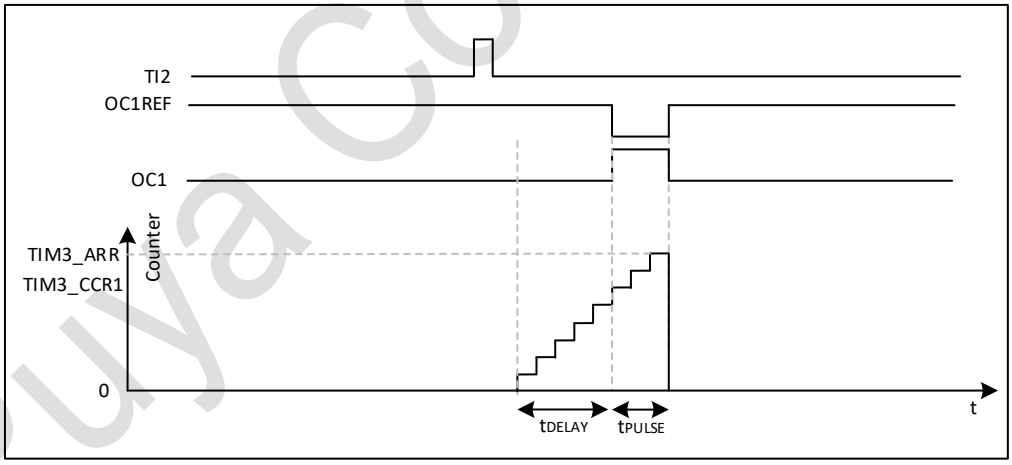


图 19-34 单脉冲模式示例

例如，当需要在从 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。

使用 TI2FP2 作为触发 1:

- 使用 TIMx_TISEL 寄存器中的 TI2SEL[3:0]位选择正确的 TI2x 源
- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=00110，TI2FP2 作为从模式控制器的触发(TRGI)。

- 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)

- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 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。

19.3.12.1. 特殊情况：OCx 快速使能

在单脉冲模式下， TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时使输出翻转。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果需要以最小延时输出波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 OCxRef（和 OCx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用。

19.3.13. 编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI1 的边沿计数，则置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI2 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。CC1NP 和 CC2NP 必须保持低电平。

两个输入 TI1 和 TI2 被用来作为正交编码器的接口。参看下表，假定计数器已经启动(TIM1_CR1 寄存器中的 CEN=1)，则计数器的时钟由 TI1FP1 或 TI2FP2 上的每次有效信号转换驱动。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转换序列，产生了计数脉冲和方向信号。依据两个输入信号的转换序列，计数器递增或递减计数，同时硬件对 TIM1_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)的发生信号转换时都会重新计算 DIR 位。

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。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

在这个模式下，计数器根据正交编码器的速度和方向自动进行修改，因此计数器的内容始终指示编码器的位置。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 19-1 计数方向与编码器信号的关系

有效边沿	相反信号的电平 (TI1FP1 对应 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

一个正交编码器可以直接与 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分输出转换位数字信号，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。在这个例子中，我们假定配置如下：

- CC1S=’01’(TIMx_CCMR1 寄存器，IC1FP1 映射到 TI1)
- CC2S=’01’(TIMx_CCMR1 寄存器，IC2FP2 映射到 TI2)
- CC1P=’0’，CC1NP=’0’(TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
- CC2P=’0’，CC2NP=’0’(TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
- SMS=’011’(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效).
- CEN=’1’(TIMx_CR1 寄存器，计数器使能)

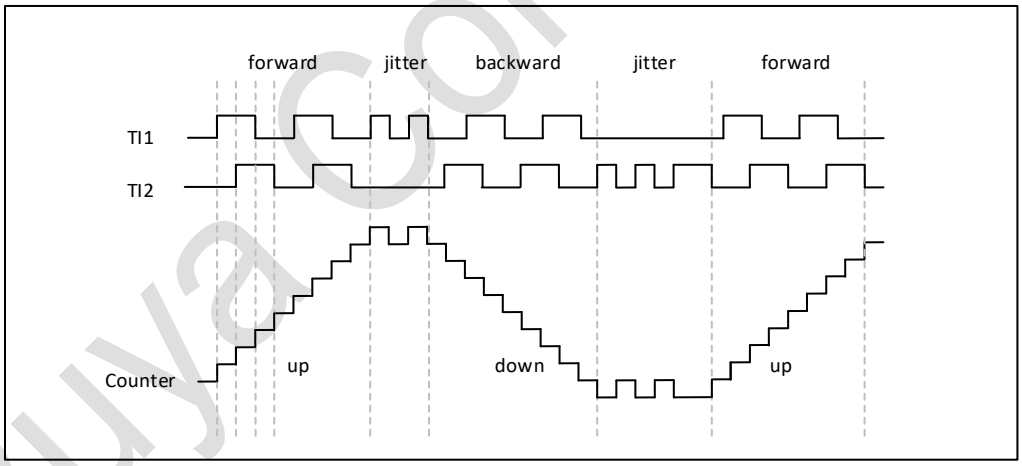


图 19-35 编码器接口模式下的计数器工作示例

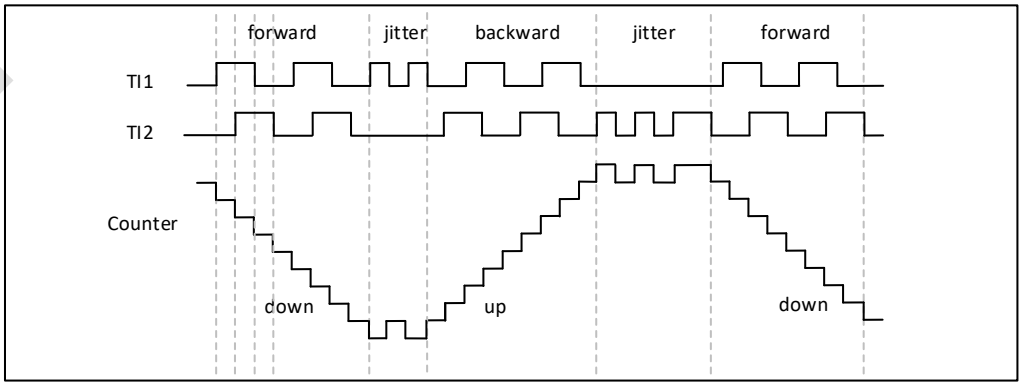


图 19-36 TI1FP1 极性反相时的编码器接口模式

当定时器配置成编码器接口模式时，提供传感器当前的位置信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度、加速度、减速度）。指示机械零点的编码器输出可被用作此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的，并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

19.3.14. 定时器输入异或模式

TIMx_CR2 寄存器的 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。

异或输出可与触发或输入捕获等所有定时器输入功能配合使用。这样便于测量两个输入信号上边沿之间的间隔。

19.3.15. 定时器和外部触发同步

TIMx 定时器可与外部触发以下列模式实现同步：复位模式、门控模式和触发模式。

19.3.15.1. 从模式：复位模式

在发生一个触发输入事件时，计数器及其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)都将更新。

在以下的例子中，TI1 输入端的上升沿导致递增计数器被清零：

- 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)。由于捕获预分频器不用于触发操作，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，CC1NP=0 以确定极性(只检测上升沿)。
- 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=00101，选择 TI1 作为输入源。
- 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=0x36 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。

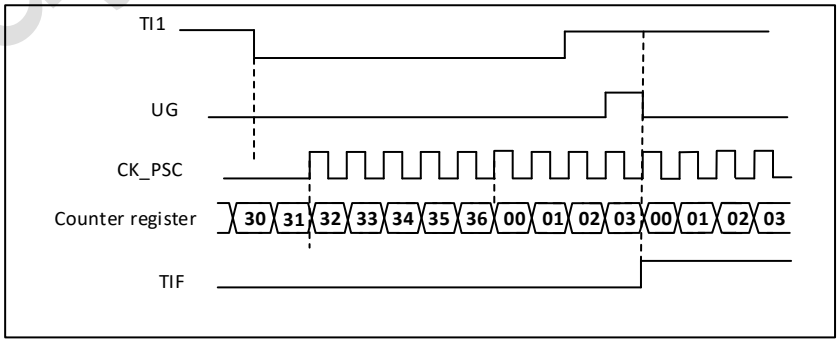


图 19-37 复位模式下的控制电路

19.3.15.2. 从模式：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 为低时递增计数：

- 通道 1 配置为检测 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(和 CC1NP=0)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=00101，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，则不论触发输入电平如何，计数器都不能启动。

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。

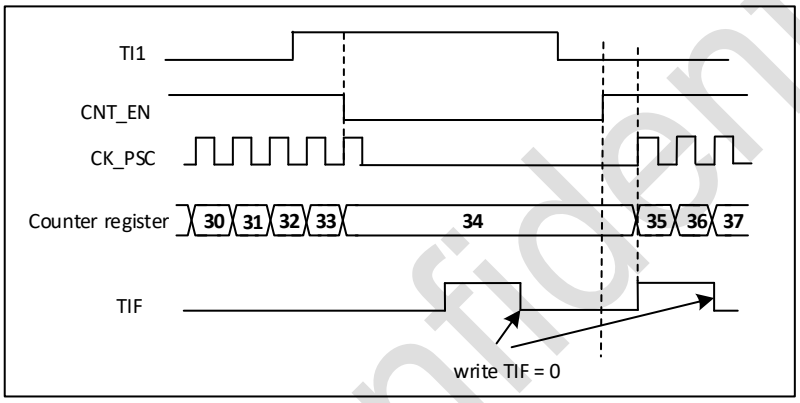


图 19-38 门控模式下的控制电路

19.3.15.3. 从模式：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 输入的上升沿开始递增计数：

- 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 TIM2x_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1(和 CC2NP=0)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=00110，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。

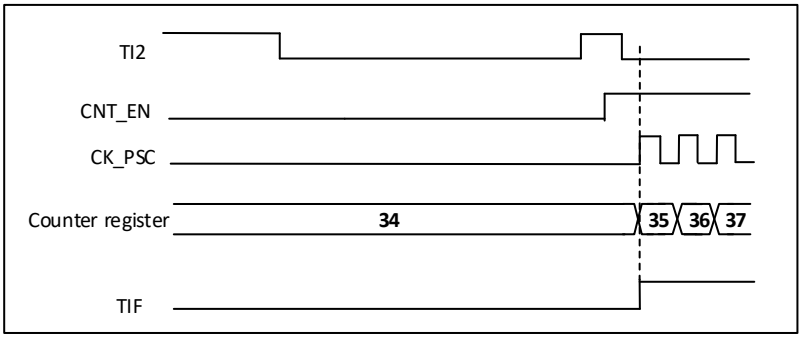


图 19-39 触发模式下的控制电路

19.3.15.4. 从模式：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 的每一个上升沿递增计数一次：

1. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：
   - ETF=0000：无滤波器
   - ETPS=00：禁止预分频器
   - 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：无滤波器
   - 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=00101，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 输入端的重新同步电路。

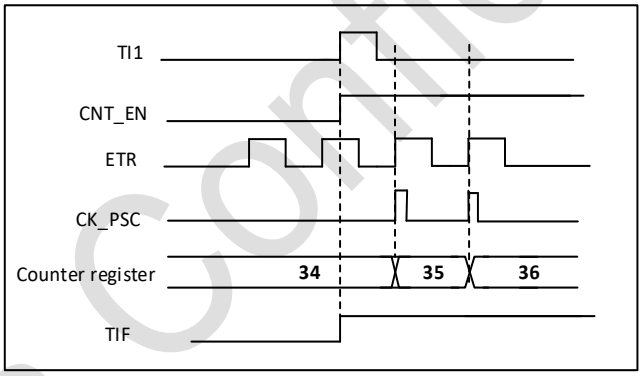


图 19-40 外部时钟模式 2 + 触发模式下的控制电路

19.3.16. 定时器同步

TIMx 定时器在内部相连，用于定时器的同步或者链接功能。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或时钟等操作。

下图显示了触发选择和主模式选择框图。

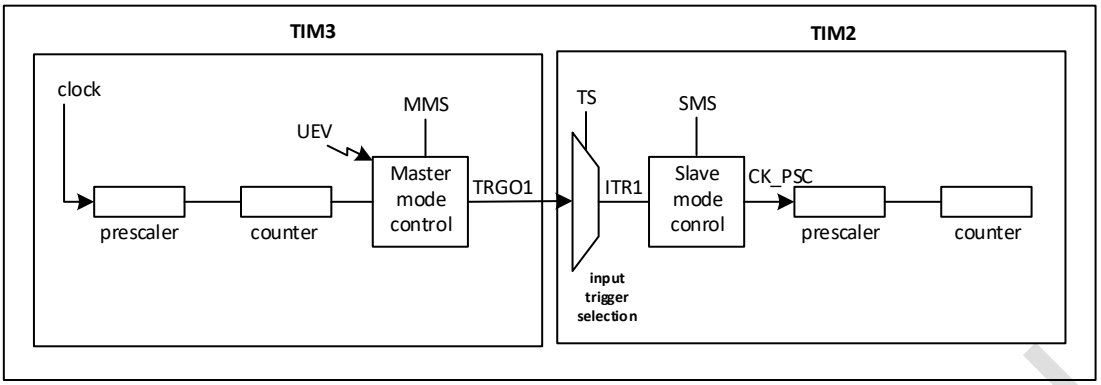


图 19-41 主/从定时器示例

19.3.16.1. 一个定时器作为另一个预分频器

如：可以配置定时器 3 作为定时器 2 的预分频器。进行下述操作：

- 配置定时器 3 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。如果配置 TIM3_CR2 寄存器的 MMS=’010’时，则每次产生一个更新事件时，TRGO1 上输出一个上升沿信号。
- 连接定时器 3 的 TRGO 输出至定时器 2，设置 TIM2_SMCR 寄存器的 TS=’00000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式 1(TIM2_SMCR 寄存器的 SMS=111)；这样定时器 2 即可由定时器 3 周期性的上升沿(即定时器 3 的计数器溢出)信号驱动。
- 最后，必须设置相应(TIM3_CR1/TIM2_CR1 寄存器)的 CEN 位分别启动两个定时器，确保先启动定时器 2，后启动定时器 3。

注：如果 OCx 已被选中为定时器 3 的触发输出(MMS=01xx)，它的上升沿用于驱动定时器 2 的计数器。

19.3.16.2. 一个定时器使能另一个

在这个例子中，定时器 2 的使能由定时器 3 的输出比较控制。参考上图的连接。只当定时器 3 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT}=f_{CK\_INT}/3$)得到。

- 配置定时器 3 为主模式，送出它的输出比较参考信号(OC1REF)为触发输出(TIM3_CR2 寄存器的 MMS=0100)
- 配置定时器 3 的 OC1REF 波形(TIM3_CCMR1 寄存器)
- 配置定时器 2 从定时器 3 获得输入触发(TIM3_SMCR 寄存器的 TS=00000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- 置 TIM3_CR1 寄存器的 CEN=1 以启动定时器 3

注：定时器 *2* 的时钟与定时器 3 的时钟不同步，这个模式只影响定时器 2 计数器的使能信号。

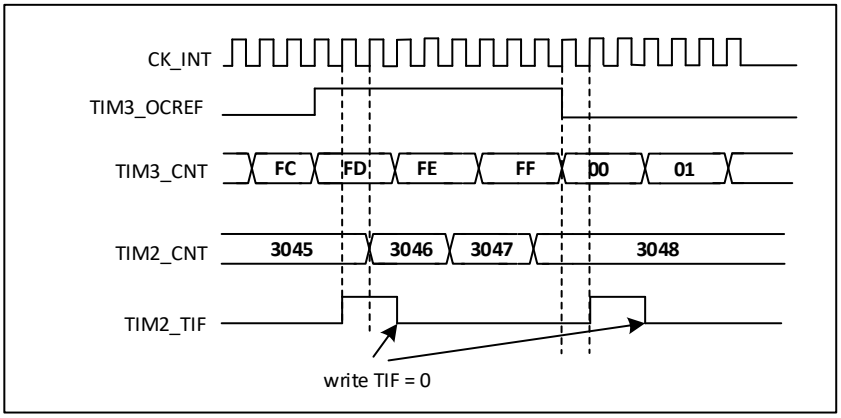


图 19-42 使用 TIM3 的 OC1REF 对 TIM2 实施门控控制

在上图的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 3 之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入需要的任意数值。写 TIMx_EGR 寄存器的 UG 位即可复位两定时器。

在下一个例子中，需要同步定时器 2 和定时器 3。定时器 3 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；两个定时器的预分频器系数相同。写'0'到 TIM3_CR1 的 CEN 位将禁止定时器 3，定时器 2 随即停止。

- 配置定时器 3 为主模式，送出定时器使能信号（CNT_EN）做为触发输出(TIM3_CR2 寄存器 MMS=0001 的)。
- 配置定时器 3 的 OC1REF 波形(TIM1_CCMR1 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=00000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM3_EGR 寄存器的 UG='1'，复位定时器 3。
- 置 TIM2_EGR 寄存器的 UG='1'，复位定时器 2。
- 写'0xE7'至定时器 2 的计数器(TIM2_CNT)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN='1'以使能定时器 2。
- 置 TIM3_CR1 寄存器的 CEN='1'以启动定时器 3。
- 置 TIM3_CR1 寄存器的 CEN='0'以停止定时器 3。

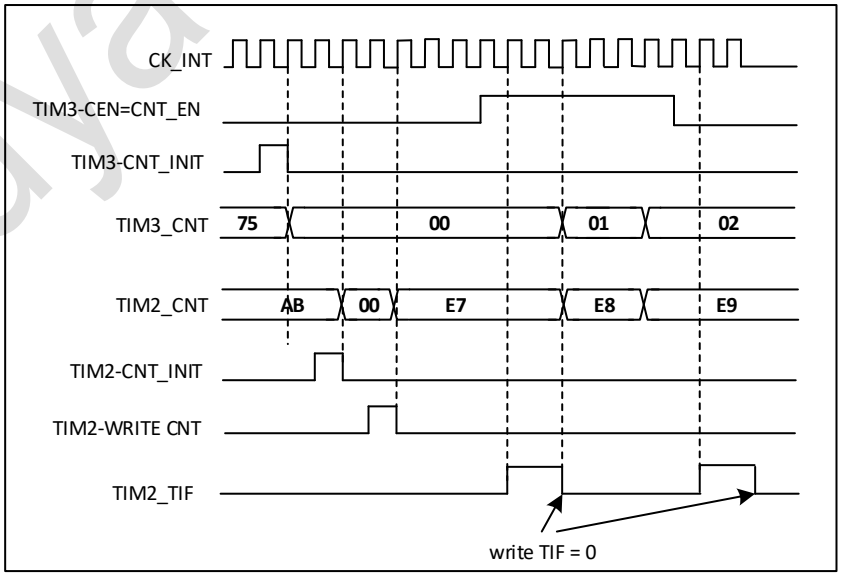


图 19-43 使用 TIM3 的使能信号对 TIM2 实施门控控制

19.3.16.3. 一个定时器启动另一个

在这个例子中，使用定时器 3 的更新事件使能定时器 2。一旦定时器 3 产生更新事件，定时器 2 即从它当前的数值(可以是非 0)按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CEN 位被自动地置'1'，同时计数器开始计数直到写'0'到 TIM2_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT}=f_{CK\_INT}/3$)。

- 配置定时器 3 为主模式，送出它的更新事件(UEV)做为触发输出(TIM3_CR2 寄存器的 MMS=0010)
- 配置定时器 3 的周期(TIM3_ARR 寄存器)
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=00000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=110)
- 置 TIM3_CR1 寄存器的 CEN=1 以启动定时器 3

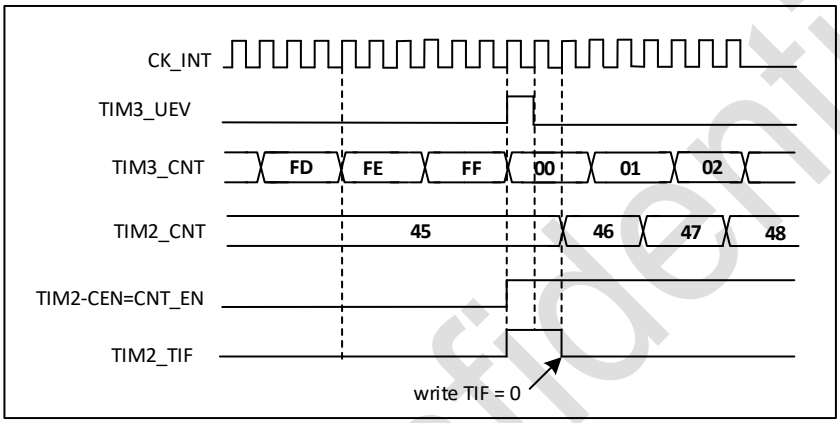


图 19-44 TIM3 的更新时间使能 TIM2

在上一个例子中，可以在启动计数之前初始化两个计数器。下图显示在与上图相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

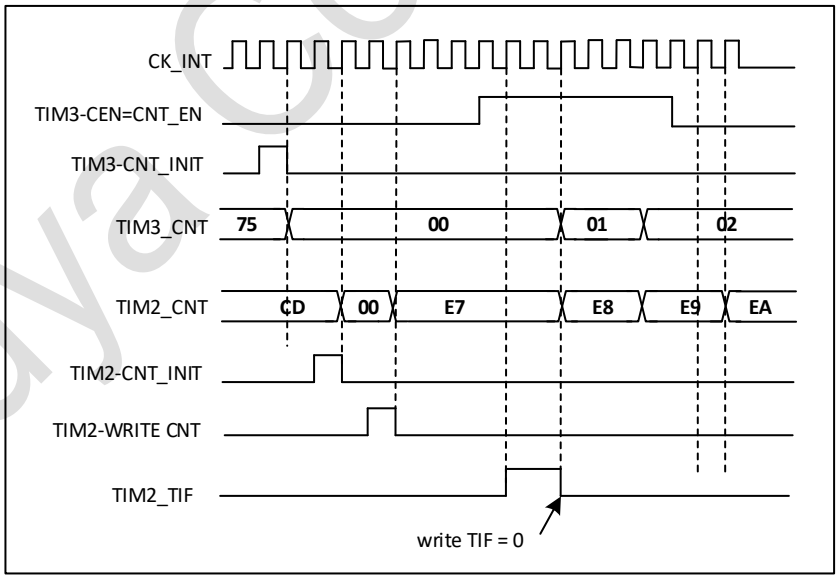


图 19-45 使能 TIM3 启动 TIM2

19.3.16.4. 外部触发同时启动两个定时器

这个例子中当定时器 3 的 TI1 输入上升时使能定时器 2，使能定时器 3 的同时使能定时器 2。为保证计数器的对齐，定时器 3 必须配置为主/从模式(对应 TI1 而言 TIM3 为从；对于 TIM2 而言，TIM3 为主)：

- 配置定时器 3 为主模式，送出它的使能作为触发输出(TIM3_CR2 寄存器的 MMS='0001')。

- 配置定时器 3 为从模式，从 TI1 获得输入触发(TIM3_SMCR 寄存器的 TS=’00100’)。
- 配置定时器 3 为触发模式(TIM3_SMCR 寄存器的 SMS=’110’)。
- 配置定时器 3 为主/从模式，TIM3_SMCR 寄存器的 MSM=’1’。
- 配置定时器 2 从定时器 3 获得输入触发(TIM2_SMCR 寄存器的 TS=00000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=’110’)。

当定时器 3 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被设置。

注：在这个例子中，在启动之前两个定时器都被初始化(设置相应的 UG 位)，两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器(TIMx_CNT)在定时器间插入一个偏移。下图中能看到主/从模式下在定时器 3 的 CNT_EN 和 CK_PSC 之间有个延迟。

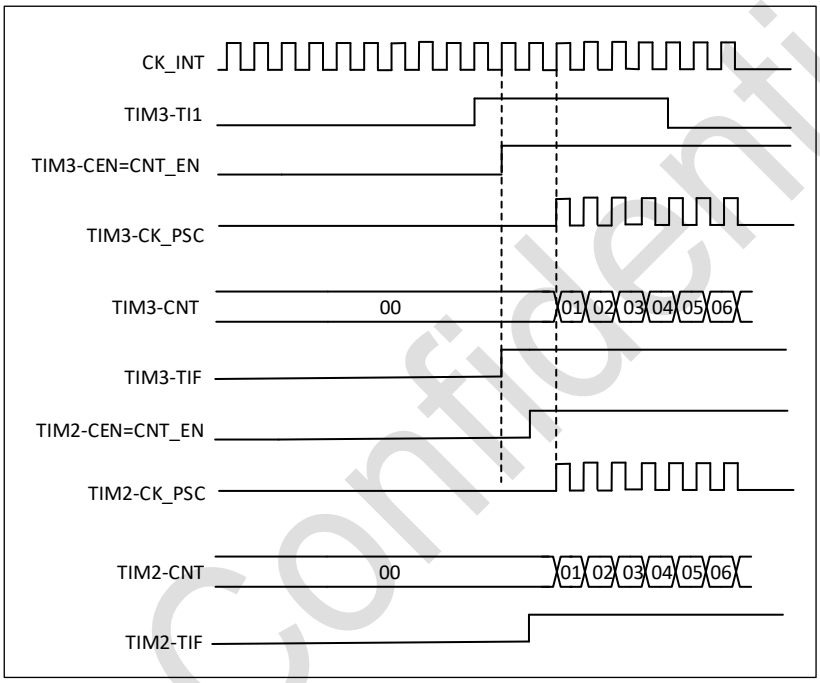


图 19-46 TIM3 的 TI1 触发 TIM3 和 TIM2

19.4. TIM2/TIM3 中断

表 19-2 TIM2/TIM3 中断

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
比较/捕获 3	CC3IF	CC3IE
比较/捕获 4	CC4IF	CC4IE
触发	TIF	TIE

19.5. TIM2/TIM3 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIM2/TIM3 计数器可以继续正常工作或者停止工作。

19.6. TIM2/TIM3 寄存器

19.6.1. TIMx 控制寄存器 1 (TIMx_CR1) (x=2,3)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	00	时钟分频因子。 这 2 位定义定时器时钟(CK_INT)频率与数字滤波器(ETR,TIx)所用的采样时钟（t_{DTS}）之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： TIMx_ARR 寄存器没有缓冲 1： TIMx_ARR 寄存器进行缓冲器
6:5	CMS[1:0]	RW	00	中心对齐模式选择。 00：边沿对齐模式。计数器依据方向位(DIR)递增或递减计数。 01：中心对齐模式 1。计数器交替地递增和递减计数。仅当计数器递减计数时，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才置位。 10：中心对齐模式 2。计数器交替地递增和递减计数。仅当计数器递增计数时，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才置位。 11：中心对齐模式 3。计数器交替地递增和递减计数。当计数器递增和递减计数时，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均被置位。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中心对齐模式。
4	DIR	RW	0	计数方向。 0：计数器递增计数 1：计数器递减计数

				注：当计数器配置为中心对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0	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	0	更新请求源。 软件通过该位选择 UEV 事件的源。 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： − 计数器上溢/下溢 − 设置 UG 位 − 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器上溢/下溢产生一个更新中断或 DMA 请求。
1	UDIS	RW	0	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： − 计数器上溢/下溢 − 设置 UG 位 − 从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值。 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	0	允许计数器。 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。

19.6.2. TIMx 控制寄存器 2 (TIMx_CR2) (x=2,3)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
								RW	RW	RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	MMS[3]	RW	0	参考 MMS[2:0]描述

24:8	Reserved	-	-	保留
7	TI1S	RW	0	TI1 选择。 0：TIMx_CH1 管脚连到 TI1 输入。 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 管脚经异或后连到 TI1 输入。
6:4	MMS[2:0]	RW	000	主模式选择。 这些位可选择主模式下将要发送到从定时器以实现同步的信息(TRGO)。可能的组合如下： 0000：复位 – TIM1_EGR 寄存器的 UG 位作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位会有一个延迟。 0001：使能 – 计数器使能信号 CNT_EN 作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或在一段时间内控制从定时器。当配置为门控模式时，计数器使能是通过 CEN 控制位和触发输入信号的逻辑**与**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 TIM1_SMCR 寄存器中 MSM 位的描述)。 0010：更新 – 更新事件作为触发输出(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。 0011：比较脉冲 – 一旦发生一次捕获或一次比较成功时，当 CC1IF 标志被置为 1 时，触发输出送出一个正脉冲(TRGO)。 0100：比较 – OC1REF 信号作为触发输出(TRGO)。 0101：比较 – OC2REF 信号作为触发输出(TRGO)。 0110：比较 – OC3REF 信号作为触发输出(TRGO)。 0111：比较 – OC4REF 信号作为作为触发输出(TRGO)。 1000：编码器时钟输出 -编码器时钟信号作为触发输出，仅当 SMS[3:0]的值为 0001、0010 和 0011 时可用。 其它：保留
3	CCDS	RW	0	捕获/比较的 DMA 选择。 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。
2:0	Reserved	-	-	保留

19.6.3. TIMx 从模式控制寄存器 (TIMx_SMCR) (x=2,3)

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[4:3]		Res.	Res.	Res.	Res.
										RW	RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[4:3]	RW	000	详见 TS 描述
19:16	Reserved	-	-	保留
15	ETP	RW	0	外部触发极性。 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 反相，低电平或下降沿有效。
14	ECE	RW	0	外部时钟使能位。 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：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=00111）具有相同效果。 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；不过此类情况下 TRGI 不得连接 ETRF（TS 位不得为 00111）。 注 3：如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。
13:12	ETPS[1:0]	RW	2’b0	外部触发预分频器。 外部触发信号 ETR 频率不能超过 TIM1CLK 频率的 1/4。可以通过使能预分频器来降低 ETR 的频率。该方法在输入快速外部时钟时非常有用。 00：预分频器关闭 01：ETR 频率的 2 分频 10：ETR 频率的 4 分频 11：ETR 频率的 8 分频
11:8	ETF[3:0]	RW	4’b0	外部触发滤波。 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 个事件后才视为一个有效输出边沿。 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6

				0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}= f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}= f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}= f_{DTS}/16$, N=5 1011：采样频率 $f_{SAMPLING}= f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	0	主/从模式。 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。此设置适用于由单个外部事件对多个定时器进行同步的情况。
6:4	TS[2:0]	RW	3’b0	触发选择 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测（TI1F_ED） 00101：滤波后的定时器输入 1（TI1FP1） 00110：滤波后的定时器输入 2（TI1FP2） 00111：外部触发输入（ETRF） 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 其它：保留 注：这些位只能在未使用(如 SMS=0000)时更改，以避免在转换时产生错误的边沿检测。 注：ITRx 的映射参见“外设互联”章节。
3	OCCS	RW	0	OCREF 清除选择位。该位用于选择 OCREF 的清除源。 0：OCREF_CLR_INT 连接到 OCREF_CLR 输入 1：OCREF_CLR_INT 连接到 ETRF
2:0	SMS[2:0]	RW	3’b0	从模式选择 当选择了外部信号，触发信号(TRGI)的有效边沿与外部输入上所选的极性相关(见输入控制寄存器和控制寄存器的说明)。

				000：禁止从模式---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 TI2FP2 的电平，计数器在 TI1FP1 的边沿递增/递减计数. 010：编码器模式 2---根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿递增/递减计数。 011：编码器模式 3---根据其他输入的电平，计数器在 TI1FP1 和 TI2FP2 的边沿向递增/递减计数。 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)，只控制计数器的启动。 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=00100)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。

19.6.4. TIMx DMA/中断使能寄存器 (TIMx_DIER) (x=2,3)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW		RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	TDE	RW	0	触发 DMA 请求使能 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	Reserved	-	-	保留
12	CC4DE	RW	0	捕获/比较 4 的 DMA 请求使能 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	0	捕获/比较 3 的 DMA 请求使能 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	0	捕获/比较 2 的 DMA 请求使能

				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	0	捕获/比较 1 的 DMA 请求使能 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	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	Reserved	-	-	保留
6	TIE	RW	0	触发中断使能 0：禁止触发中断 1：允许触发中断
5	Reserved	-	-	保留
4	CC4IE	RW	0	捕获/比较 4 中断使能 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	RW	0	捕获/比较 3 中断使能 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	捕获/比较 2 中断使能 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	捕获/比较 1 中断使能 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

19.6.5. TIMx 状态寄存器 (TIMx_SR) (x=2,3)

偏移地址:0x010

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Res.	Res.	Res.	Res.	IC2IR	IC1IR	Res.	Res.
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0					RC_W0	RC_W0		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1F	UIF
			RC_W0	RC_W0	RC_W0	RC_W0			RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。

28	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
27	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
26	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标志可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生下降沿捕获事件。
25	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
24	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
23:20	Reserved	-	-	保留
19	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
18	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标志可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生上升沿捕获事件。
17:13	Reserved	-	-	保留
12	CC4OF	RC_W0	0	捕获/比较 4 过捕获标志 参见 CC1OF 描述
11	CC3OF	RC_W0	0	捕获/比较 3 过捕获标志 参见 CC1OF 描述
10	CC2OF	RC_W0	0	捕获/比较 2 过捕获标志 参见 CC1OF 描述
9	CC1F	RC_W0	0	捕获/比较 1 过捕获标志 仅当相应的通道被配置为输入捕获时，该标志可由硬件置 1。写 0 可清除该位。 0：无过捕获产生； 1：CC1IF 置 1 时，计数器的值已经被捕获到 TIMx_CCR1 寄存器。
8:7	Reserved	-	-	保留
6	TIF	RC_W0	0	触发器中断标志 除门控模式外的其它所有模式下，当使能从模式控制器后，在 TRGI 输入端检测到有效边沿时，由硬件对该位置 1。选择门控模式时，

				该标志将在计数器启动或停止时置 1，但需要通过软件清零。 0：无触发器事件产生； 1：触发器中断等待响应
5	Reserved	-	-	保留
4	CC4IF	RC_W0	0	捕获/比较 4 中断标志 参考 CC1IF 描述
3	CC3IF	RC_W0	0	捕获/比较 3 中断标志 参考 CC1IF 描述
2	CC2IF	RC_W0	0	捕获/比较 2 中断标志 参考 CC1IF 描述
1	CC1IF	RC_W0	0	捕获/比较 1 中断标志 **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，但在中心对齐模式下除外(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。或者当 TIM1_CCR1 的值大于 TIM1_ARR（递增计数和中心对齐计数的上溢时、递减计数的下溢时 CC1IF 置 1）。 **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：输入捕获产生并且计数器值已装入 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标志 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：更新事件等待响应。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 REP_CNT=0 时产生更新事件(重复递减计数器上溢或下溢时)； − 若 TIMx_CR1 寄存器的 UDIS=0、URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

				– 若 TIMx_CR1 寄存器的 UDIS=0、URS=0，当 CNT 被触发事件重初始化时产生更新事件。（参考从模式控制寄存器(TIMx_SMCR)）

19.6.6. TIMx 事件产生寄存器（TIMx_EGR) (x=2,3)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	TG	W	0	产生触发事件 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，使能后可发生相关中断或 DMA 传输事件。
5	Reserved	-	0	保留，一直为 0
4	CC4G	W	0	产生捕获/比较 4 事件 参考 CC1G 描述
3	CC3G	W	0	产生捕获/比较 3 事件 参考 CC1G 描述
2	CC2G	W	0	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	W	0	产生捕获/比较 1 事件 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**： CC1IF 置位，若使能则产生相应的中断和 DMA。 **若通道 CC1 配置为输入**： 当前的计数器值捕获至 TIMx_CCR1 寄存器， CC1IF 置位，若使能则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 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

				中心对齐模式下或 DIR=0(递增计数)则计数器被清 0， 若 DIR=1(递减计数)则计数器用 TIMx_ARR 的值。

19.6.7. TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1) (x=2,3)

偏移地址:0x18

复位值:0x0000 0000

同一寄存器可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获和输出比较模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.6.7.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F	RW	4’b0	输入捕获 2 滤波器。 参见 IC1F[3:0]的描述。
11:10	IC2PSC[1:0]	RW	2’b0	输入/捕获 2 预分频器。 参见 IC1PSC[1:0]的描述。
9:8	CC2S[1:0]	RW	0	捕获/比较 2 选择。 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。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 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[3:0]	RW	4’b0	输入捕获 1 滤波器。 该位域定义了 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=6

				0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC[1:0]	RW	2’b0	输入/捕获 1 预分频器 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[1:0]	RW	00	捕获/比较 1 选择。 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。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

19.6.7.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	OC2CE	RW	0	输出比较 2 清 0 使能。 参见 OC1CE 的描述。
14:12	OC2M[2:0]	RW	3’b0	输出比较 2 模式选择。 参见 OC1M[2:0]的描述。
11	OC2PE	RW	0	输出比较 2 预装载使能。 参见 OC1PE 的描述。
10	OC2FE	RW	0	输出比较 2 快速使能。 参见 OC1FE 的描述。
9:8	CC2S[1:0]	RW	2’b0	捕获/比较 2 选择。 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。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 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	0	输出比较 1 清 0 使能。 0：OC1REF 不受 ETRF 或者 OCREF_CLR 输入的影响； 1：一旦检测到 ETRF 或者 OCREF_CLR 输入高电平，则清除 OC1REF。
6:4	OC1M[2:0]	RW	3’b0	输出比较 1 模式。 该位定义了 OC1、OC1N 的输出参考信号 OC1REF 的行为。OC1REF 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：设置通道 1 为匹配时输出有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)匹配时，强制 OC1REF 为高电平。 010 ：设置通道 1 为匹配时输出无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)匹配时，强制 OC1REF 为低电平。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，OC1REF 发生翻转。 100：强制为无效电平。强制 OC1REF 为低电平。 101：强制为有效电平。强制 OC1REF 为高电平。 110：PWM 模式 1---在递增计数时，一旦 TIMx_CNT<TIMx_CCR1，通道 1 为有效状态，否则为无效状态；在递减计数时，一旦 TIMx_CNT>TIMx_CCR1，通道 1 为无效状态(OC1REF=0)，否则为有效状态(OC1REF=1)。 111：PWM 模式 2---在递增计数时，一旦 TIMx_CNT<TIMx_CCR1，通道 1 为无效状态，否则为有效状态；在递减计数时，一旦 TIMx_CNT>TIMx_CCR1，通道 1 为有效状态，否则为无效状态。 注 1：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	0	输出比较 1 预装载使能。 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，写入后将立即使用新值。

				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载入当前寄存器中。 注 1：仅在单脉冲模式下，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	0	输出比较 1 快速使能。 该位用于加快触发器输入事件对 CC 输出的影响。 0：即使触发器开启，CC1 也将根据计数器和 CCR1 的值正常操作。当触发器的输入出现边沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：触发输入上出现有效边沿相当于 CC1 输出上发生比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和 CC1 输出间的延时被缩短为 3 个时钟周期。 只在通道被配置成 PWM1 或 PWM2 模式时 OCxFE 才起作用。
1:0	CC1S[1:0]	RW	2’b0	捕获/比较 1 选择。 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。此模式仅在通过 TS 位（TIM1_SMCR 寄存器）选择内部触发器输入时有效。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

19.6.8. TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2) (x=2,3)

偏移地址:0x1C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

19.6.8.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IF4F	RW	0000	输入捕获 4 滤波器。 参见 IC1F 的描述。
11:10	IC4PSC[1:0]	RW	2’b0	输入/捕获 4 预分频器。 参见 IC1PSC 的描述。

9:8	CC4S[1:0]	RW	2’b0	捕获/比较 4 选择。 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。 此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 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[3:0]	RW	4’b0	输入捕获 1 滤波器。 参见 IC1F 的描述。
3:2	IC3PSC[1:0]	RW	2’b0	输入/捕获 3 预分频器。 参见 IC1PSC 的描述。
1:0	CC3S[1:0]	RW	2’b0	CC3S[1:0]： 捕获/比较 1 选择。 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 TI3 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。 此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

19.6.8.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	OC4CE	RW	0	输出比较 4 清 0 使能。 参见 OC1CE 的描述。
14:12	OC4M[2:0]	RW	3’b0	输出比较 4 模式 参见 OC4M 的描述。
11	OC4PE	RW	0	输出比较 4 预装载使能。 参见 OC1PE 的描述。
10	OC4FE	RW	0	输出比较 4 快速使能。 参见 OC1FE 的描述。
9:8	CC4S[1:0]	RW	2’b0	捕获/比较 4 选择。 该位定义通道的方向（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。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 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	0	输出比较 3 清 0 使能。 参见 OC1CE 的描述。
6:4	OC3M[2:0]	RW	3’b0	输出比较 3 模式。 参见 OC1M[3:0]的描述。
3	OC3PE	RW	0	输出比较 3 预装载使能。 参见 OC1PE 的描述。
2	OC3FE	RW	0	输出比较 3 快速使能。 参见 OC1FE 的描述。
1:0	CC3S[1:0]	RW	2’b0	捕获/比较 3 选择。 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。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发器输入时有效。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

19.6.9. TIMx 捕获/比较使能寄存器（TIMx_CCER) (x=2,3)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CC4NP	Res.	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
RW		RW	RW	RW		RW	RW	RW		RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	CC4NP	RW	0	输入/捕获 4 互补输出极性。 参考 CC1NP 的描述。
14	Reserved	-	-	保留
13	CC4P	RW	0	输入/捕获 4 输出极性。 参考 CC1P 的描述。
12	CC4E	RW	0	输入/捕获 4 输出使能。 参考 CC1E 的描述。

11	CC3NP	RW	0	输入/捕获 3 互补输出极性。 参考 CC1NP 的描述。
10	Reserved	-	-	保留
9	CC3P	RW	0	输入/捕获 3 输出极性。 参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能。 参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。 参考 CC1NP 的描述。
6	Reserved	-	-	保留
5	CC2P	RW	0	输入/捕获 2 输出极性。 参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。 参考 CC1E 的描述。
3	CC1NP	RW	0	该位和 CC1P 联合使用定义 TI1FP1/TI2FP1 极性， 参考 CC1P 描述。
2	Reserved	-	-	保留
1	CC1P	RW	0	输入/捕获 1 输出极性。 **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]位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性： 00：非反相/上升沿。TIxFP1 上升沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式、编码器模式）。 01：反相/下降沿。TIxFP1 下降沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：非反相/双沿。TIxFP1 上升和下降沿都有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式）。这个配置不能应用于编码器模式下。
0	CC1E	RW	0	输入/捕获 1 输出使能 **CC1 通道配置为输出：** 0：关闭 - OC1 禁止输出 1：开启 - OC1 信号输出到对应的输出引脚 **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 19-3 标准 OCx 通道的输出控制位

CCxE 位	OCx output State
0	输出禁止（OCx=0,OCx_EN=0）
1	OCx=OCxREF+极性,OCx_EN=1

19.6.10. TIMx 计数器寄存器（TIMx_CNT) (x=2,3)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	RW	0	计数器的值 注：TIM3 有效位为[15:0]

19.6.11. TIMx 预分频寄存器（TIMx_PSC) (x=2,3)

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	16’b0	预分频器值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时（包括计数器通过 TIM1_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）要装载到有效预分频器寄存器的值。

19.6.12. TIMx 自动装载寄存器（TIMx_ARR) (x=2,3)

偏移地址:0x2C

复位值:0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32’hFFFFFFFF	自动重装载的值

				ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。 注：TIM3 有效位为[15:0]

19.6.13. TIMx 捕获/比较寄存器 1 (TIMx_CCR1) (x=2,3)

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR1[31:0]	RW	32’b0	捕获/比较 1 的值 **若 CC1 通道配置为输出**: CCR1 包含了要装载当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIMx_CNT 比较的值，并且在 OC1 端口上输出信号。 **若 CC1 通道配置为输入**: CCR1 为上一次输入捕获 1 事件（IC1）发生时的计数器值。此时，只能读取 TIMx_CCR1 寄存器，无法对其进行编程。 注：TIM3 有效位为[15:0]

19.6.14. TIMx 捕获/比较寄存器 2 (TIMx_CCR2) (x=2,3)

偏移地址:0x38

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR2[31:0]	RW	32’b0	捕获/比较 2 的值 **若 CC2 通道配置为输出**: CCR2 包含了要装载当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器(OC2PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。

				实际捕获/比较寄存器包含要与计数器 TIMx_CNT 比较的值，并且在 OC2 端口上输出信号。 **若 CC2 通道配置为输入**： CCR2 为上一次输入捕获 2 事件（IC2）发生时的计数器值。此时，只能读取 TIMx_CCR2 寄存器，无法对其进行编程。 注：TIM3 有效位为[15:0]

19.6.15. TIMx 捕获/比较寄存器 3 (TIMx_CCR3) (x=2,3)

偏移地址:0x3C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR3[31:0]	RW	32'b0	捕获/比较 3 的值 **若 CC3 通道配置为输出**： CCR3 包含了要装载当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器(OC3PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIMx_CNT 比较的值，并且在 OC3 端口上输出信号。 **若 CC3 通道配置为输入**： CCR3 为上一次输入捕获 3 事件（IC3）发生时的计数器值。此时，只能读取 TIMx_CCR3 寄存器，无法对其进行编程。 注：TIM3 有效位为[15:0]

19.6.16. TIMx 捕获/比较寄存器 4 (TIMx_CCR4) (x=2,3)

偏移地址:0x40

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR4[31:0]	RW	32'b0	捕获/比较 4 的值 **若 CC4 通道配置为输出**：

				CCR4 包含了要装载当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器(OC4PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIMx_CNT 比较的值，并且在 OC4 端口上输出信号。 **若 CC4 通道配置为输入**: CCR4 为上一次输入捕获 4 事件（IC4）发生时的计数器值。此时，只能读取 TIMx_CCR4 寄存器，无法对其进行编程。 注：TIM3 有效位为[15:0]

19.6.17. TIMx 输入选择寄存器（TIMx_TISEL) (x=2,3)

偏移地址:0x5C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	TI4SEL[3:0]	RW	4’b0	TI4 输入选择. 0000：TIMx_CH4 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
23:20	Reserved	-	-	保留
19:16	TI3SEL[3:0]	RW	4’b0	TI3 输入选择. 0000：TIMx_CH3 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
15:12	Reserved	-	-	保留
11:8	TI2SEL[3:0]	RW	4’b0	TI2 输入选择. 0000：TIMx_CH2 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL[3:0]	RW	4’b0	TI1 输入选择. 0000：TIMx_CH1 0001：COMP1_OUT

				0010：COMP2_OUT 其他：保留

19.6.18. TIMx 备用选项寄存器 1 (TIMx_AF1) (x=2,3)

偏移地址:0x60

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW														

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:14	ETRSEL[3:0]	RW	4’b0	外部触发源选择（etr_in source selection） 该位段用于选择 ETR 输入源 0000：TIM2_ETR for TIM2,TIM3_ETR for TIM3 0001：COMP1_OUT 0010：COMP2_OUT 0011：保留 0100：保留 0101：TIM3_ETR for TIM2,TIM2_ETR for TIM3 其它：保留
13:0	Reserved	-	-	保留

19.6.19. TIMx 备用选项寄存器 2 (TIMx_AF2) (x=2,3)

偏移地址:0x64

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OCRSEL[2:0]		
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18:16	OCRSEL[2:0]	RW	3’b0	OCREF 复位源选择。 该位段用于选择 ocref_clr 输入源 0000：COMP1_OUT 0001：COMP2_OUT 其它：保留
15:0	Reserved	-	-	保留

19.6.20. TIMx DMA 控制寄存器 (TIMx_DCR) (x=2,3)

偏移地址:0x3DC

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL[4:0]	RW	5'b0	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送）。 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输
7:5	Reserved	-	-	保留
4:0	DBA[4:0]	RW	5'b0	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIMx_DMAR 寄存器的地址进行读或写时），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， .....

19.6.21. TIMx DMA 连续传输地址寄存器 (TIMx_DMAR) (x=2,3)

偏移地址:0x3E0

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	32'b0	DMA 连续传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIMx_CR1 地址 +（ DBA + DMA 索引）*4，其中： “TIMx_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 指针”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。

20. 通用定时器(TIM15)

20.1. TIM15 主要特性

- 16 位自动装载递增计数器
- 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
- 两个通道作为：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿对齐模式）
  - 单脉冲模式输出
- 带可编程死区时间的互补输出（仅适用于通道 1）
- 使用外部信号控制定时器且可实现多个定时器互连的同步电路
- 重复计数器，在计数指定周期数后，才更新定时器的寄存器
- 刹车输入可以将定时器的输出信号置为用户可选的安全配置中
- 如下时间发生时产生中断/DMA：
  - 更新：计数器上溢，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
  - 刹车输入

20.2. TIM15 功能描述

20.2.1. 功能框图

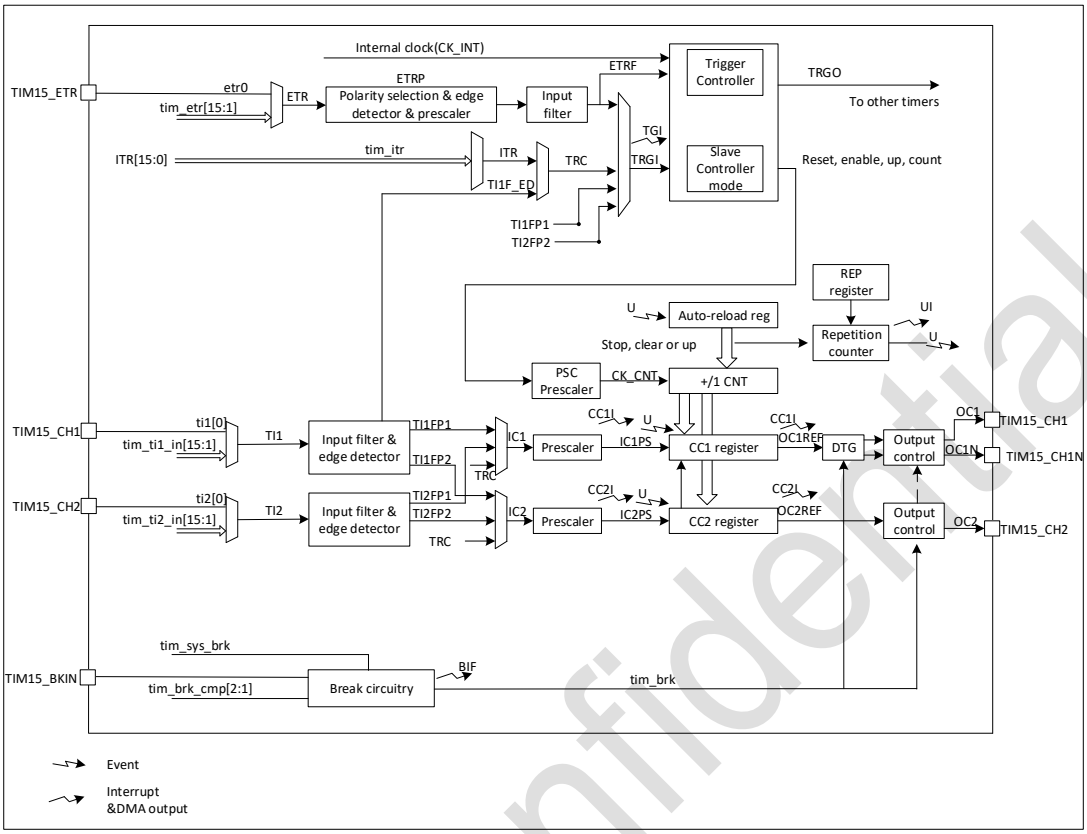


图 20-1 TIM15 框图

20.2.2. 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位的递增计数器，计数器的时钟可通过预分频器进行分频。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包括：

- 计数器寄存器（TIM15_CNT）
- 预分频寄存器（TIM15_PSC）
- 自动装载寄存器（TIM15_ARR）
- 重复计数寄存器（TIM15_RCR）

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存器，这取决于 TIM15_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值并且 TIM1_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。

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 TIM15_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 TIM15_CR 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

20.2.2.1. 预分频描述

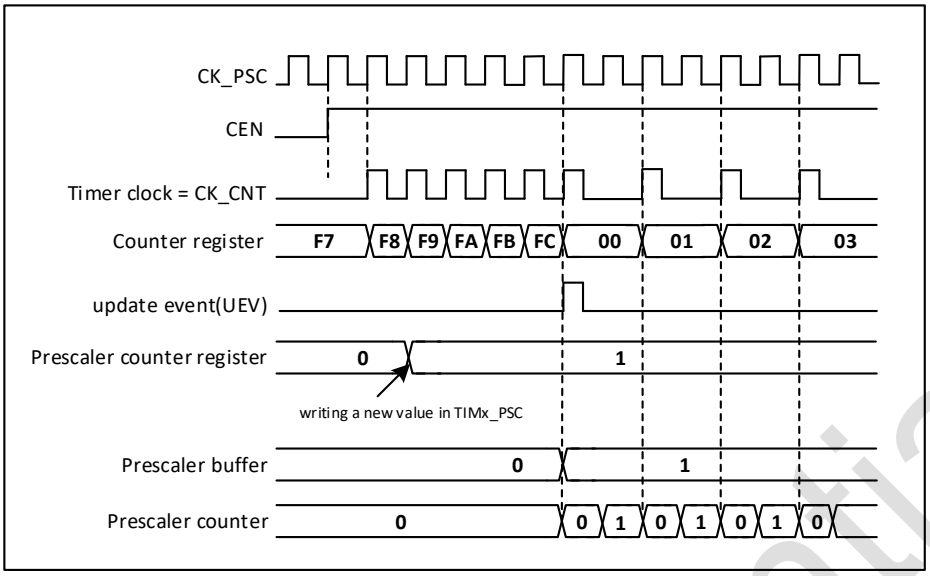


图 20-2 预分频器分频由 1 变为 2 时的计数器时序图

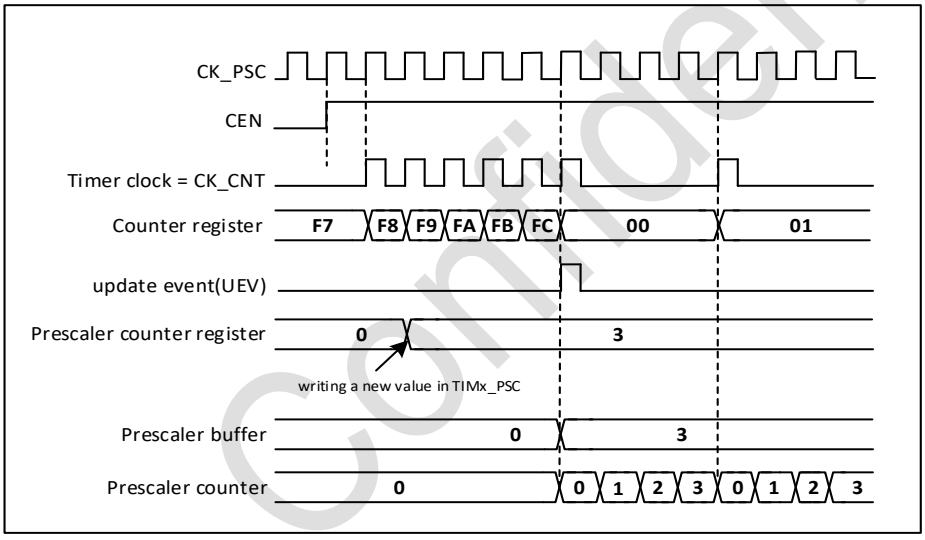


图 20-3 预分频器分频由 1 变为 4 时的计数器时序图

20.2.3. 计数模式（递增）

计数器从 0 计数到自动装载值（TIM15_ARR 寄存的值），然后又从 0 重新开始计数，并产生一个计数器溢出事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中设定的次数加一次（(TIM15_RCR) +1）后，将生成更新事件(UEV)。否则，将在每次计数器上溢时产生更新事件。

在 TIM15_EGR 寄存器中设置 UG 位(通过软件方式或者使用从模式控制器)也同样可以产生一个更新事件。

设置 TIM15_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIM15_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIM15_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）。

- 重复计数器被 TIM15_RCR 寄存器中的值重新装载。
- 自动装载影子寄存器被重新置入预装载寄存器的值(TIM15_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIM15_PSC 寄存器的内容)。

下图显示了几个在不同频率下的计数器行为，当 TIM15_ARR=0x36。

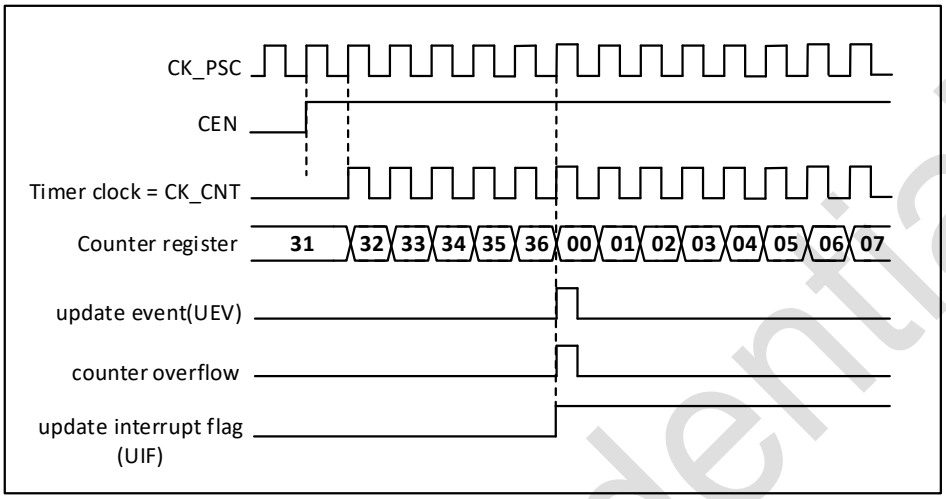


图 20-4 计数器时序图，内部时钟 1 分频

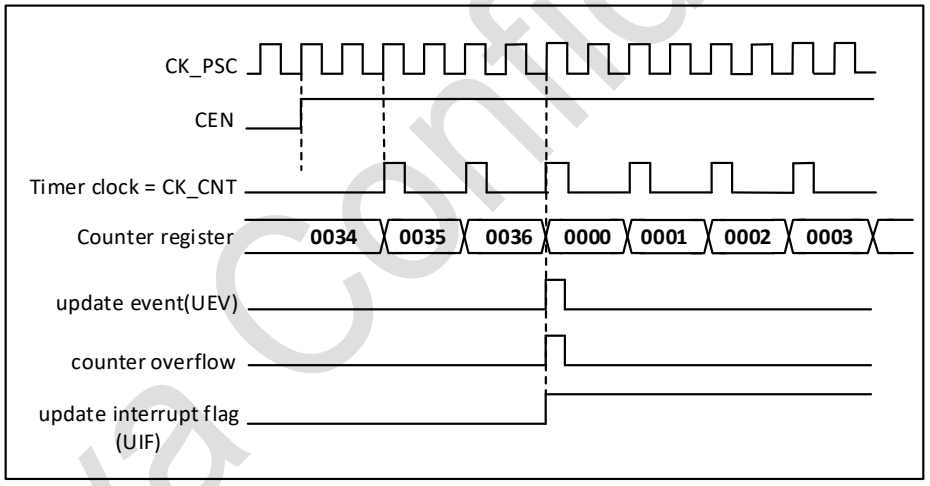


图 20-5 计数器时序图，内部时钟 2 分频

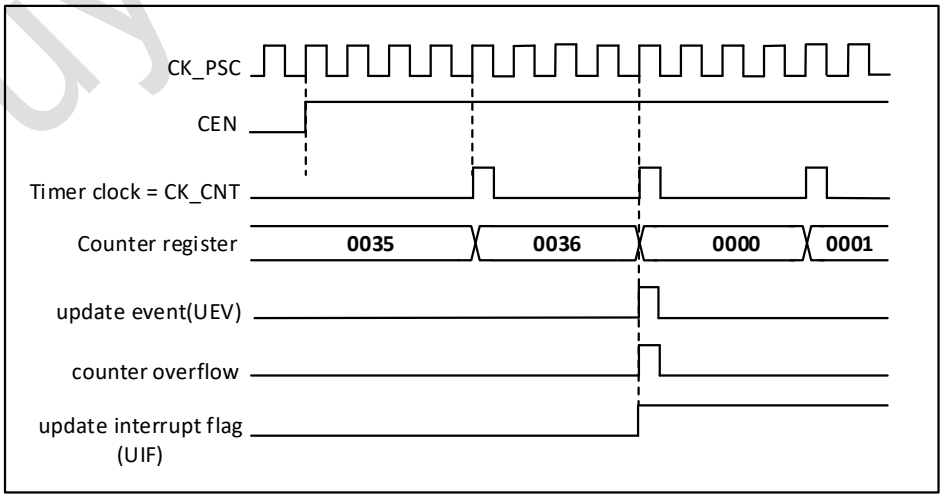


图 20-6 计数器时序图，内部时钟 4 分频

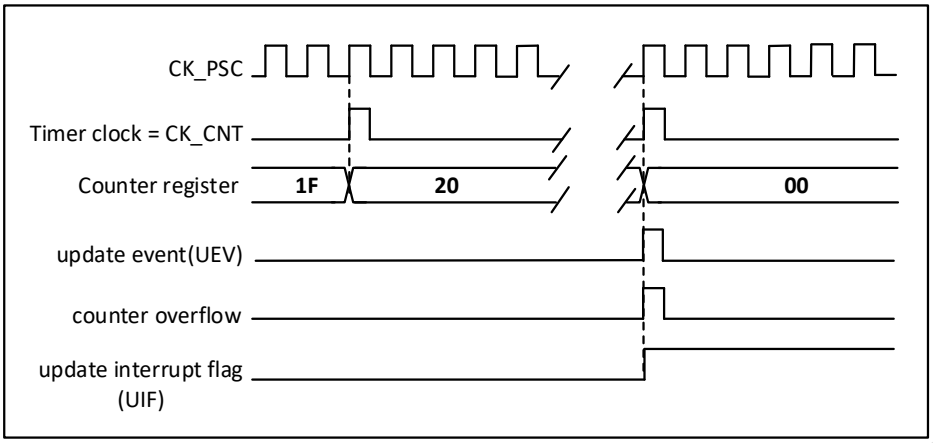


图 20-7 计数器时序图，内部时钟 N 分频

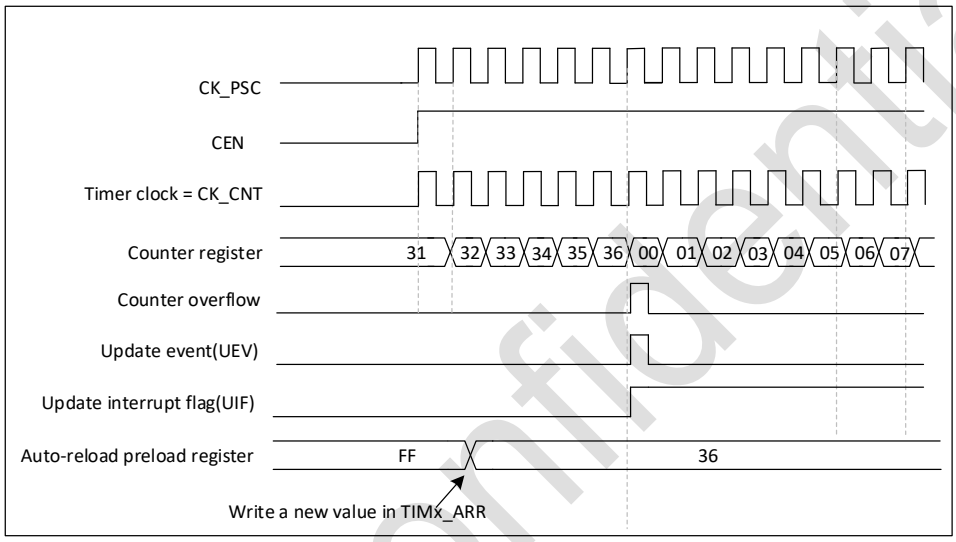


图 20-8 计数器时序图，ARPE=0 时更新事件（TIM15_ARR 未预装载）

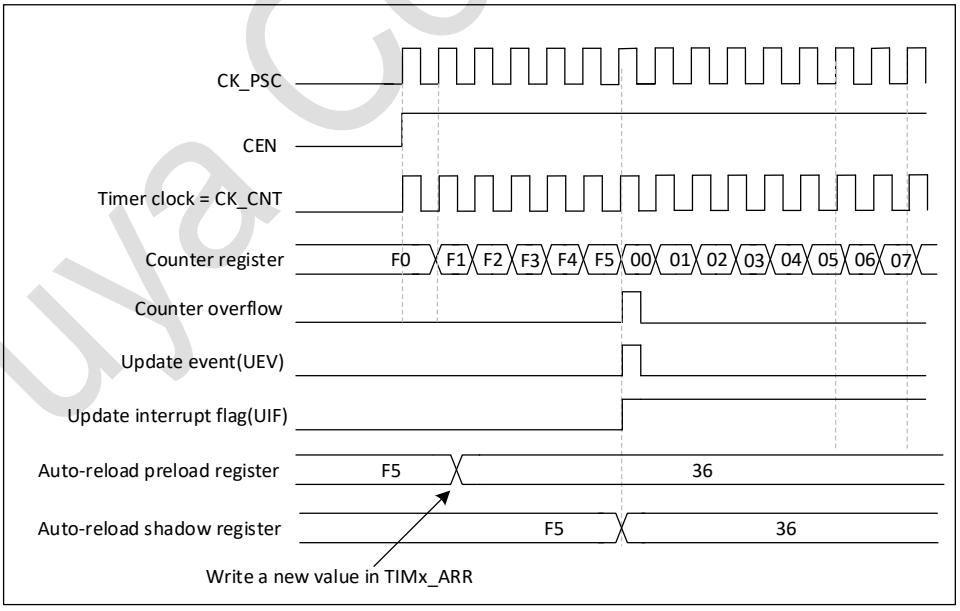


图 20-9 计数器时序图，ARPE=1 时更新事件（TIM1_ARR 预装载）

20.2.4. 重复计数器

时基单元描述了计数器上溢时更新事件（UEV）是如何产生的，然而实际上它只能在重复计数器达到 0 的时候产生。这个特性对产生 PWM 信号有用。

这意味着在每 N 次计数上溢时(N 是 TIM15_RCR 重复计数器寄存器中的值)，数据从预装载寄存器传输到影子寄存器（TIM15_ARR 自动重载寄存器，TIM15_PSC 预分频寄存器，还有在比较模式下的捕获/比较寄存器 TIM15_CCRx）。

重复计数器，在递增计数器模式的每个计数上溢时递减。

重复计数器是自动重载的，重复速率是由 TIM15_RCR 寄存器的值定义。当更新事件由软件（通过设置 TIM15_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 TIM15_RCR 寄存器中的内容被重载到重复计数器中。

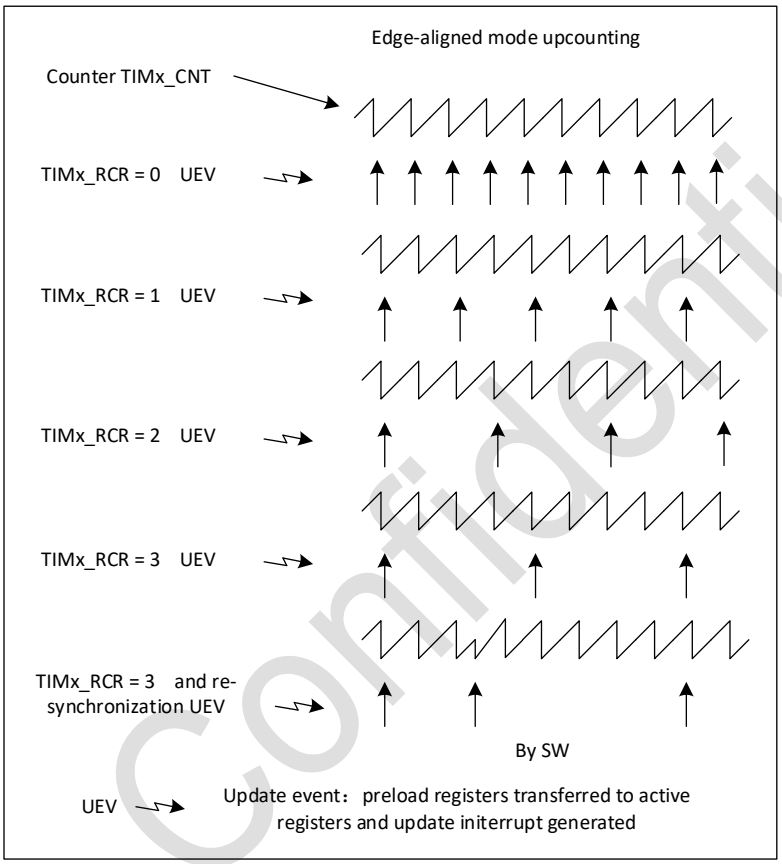


图 20-10 不同模式和 TIM15_RCR 寄存器设置下的更新频率示例

20.2.5. 时钟源

20.2.5.1. 内部时钟源（CK_INT)

计数器的时钟由内部时钟（CK_INT）提供。TIM15_CR1 寄存器的 CEN 位和 TIM15_EGR 寄存器的 UG 位是实际的控制位，只能通过软件修改（除了 UG 位被自动清除外）。一旦置 CEN 位为 1，预分频器的时钟就由内部时钟 CK_INT 提供。

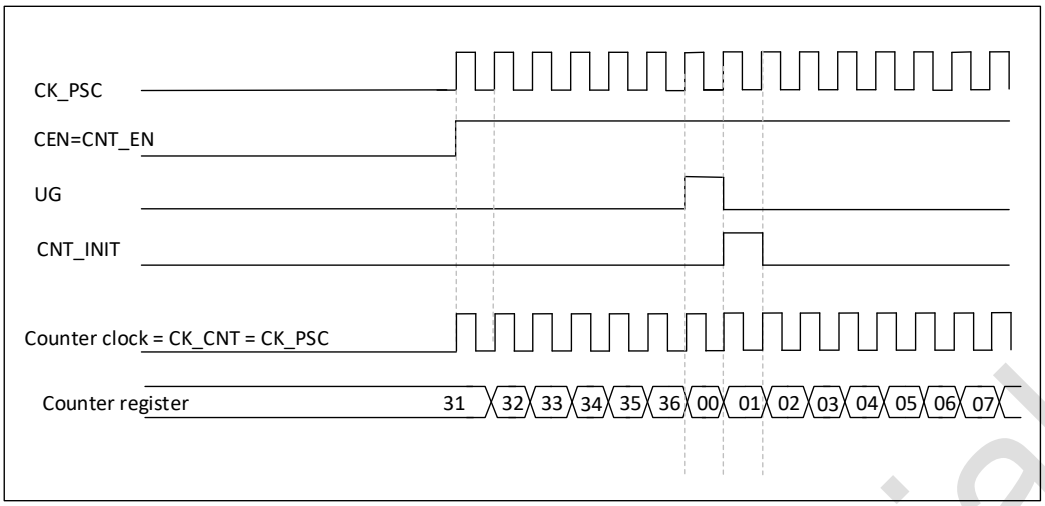


图 20-11 正常模式下的控制电路，内部时钟 1 分频

20.2.5.2. 外部时钟模式 1

当 TIM15_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

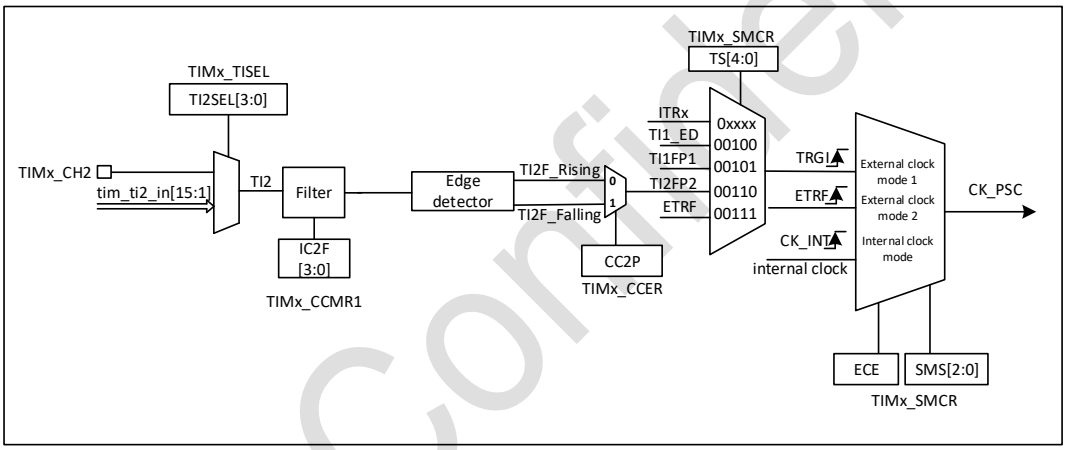


图 20-12 TI2 外部时钟连接示例

例如，要使递增计数器在 TI2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIM15_TISEL 寄存器中配置 TI2SEL[3:0]位选择正确的 TI2x 源（内部或外部）。
2. 配置 TIM15_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿
3. 配置 TIM15_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)
4. 配置 TIM15_CCER 寄存器的 CC2P=0 和 CC2NP=0，选定上升沿极性
5. 配置 TIM15_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1
6. 配置 TIM15_SMCR 寄存器中的 TS=00110，选择 TI2 作为触发输入源
7. 设置 TIM15_CR1 寄存器的 CEN=1，启动计数器

注：由于捕获预分频器不用作触发，所以不需要对它进行配置。

当 TI2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

TI2 的上升沿与实际计数器时钟之间的延迟是由于 TI2 输入的重新同步电路引起的。

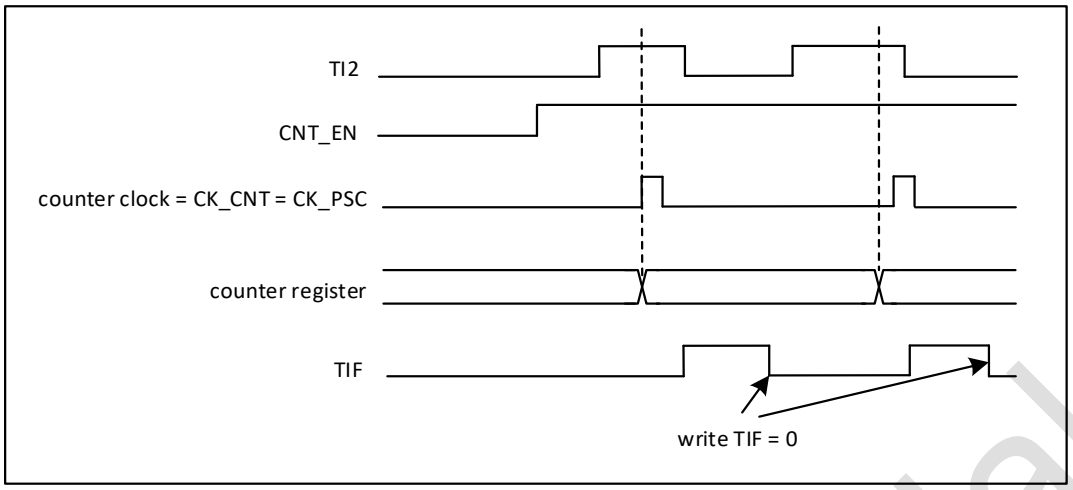


图 20-13 外部时钟模式 1 控制电路时序

20.2.5.3. 外部时钟模式 2

通过写 TIM15_SMCR 寄存器的 ECE 为 1，选定此模式。计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图：

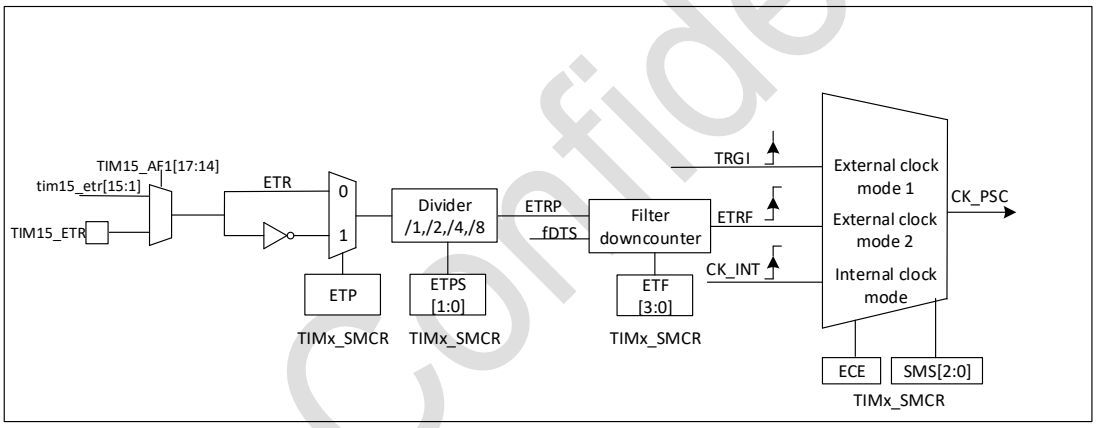


图 20-14 外部触发输入 ETR 连接

例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIM15_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIM15_SMCR 寄存器中写入 ETPS[1:0]=01 来设置预分频器。
3. 通过在 TIM15_SMCR 寄存器中写入 ETP=0 来选择 ETR 引脚的上升沿检测。
4. 通过在 TIM15_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
5. 通过在 TIM15_CR1 寄存器中写入 CEN=1 来使能计数器。

ETR 每出现 2 个上升沿，计数器计数一次。

ETR 的上升沿与实际计数器时钟之间的延迟是由于 ETRP 信号的重新同步电路引起的。

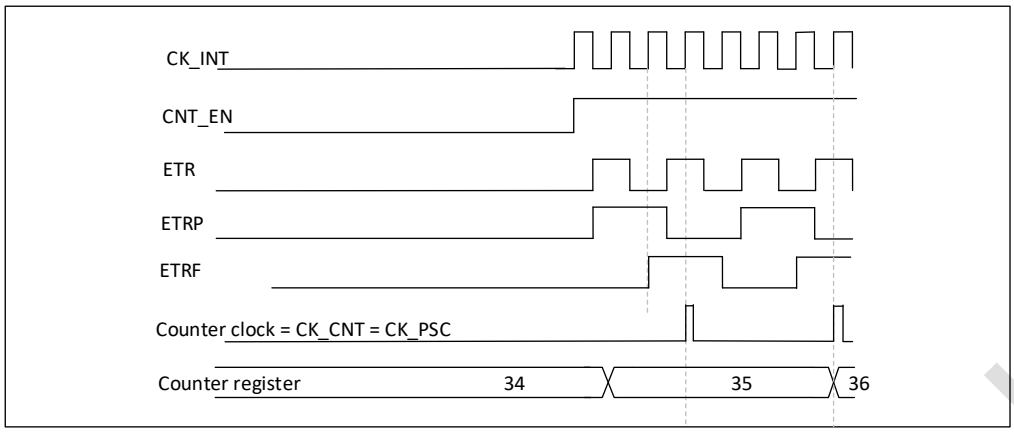


图 20-15 外部时钟模式 2 时序

20.2.6. 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入部分（数字滤波、多路复用和预分频器）和一个输出部分（比较器和输出控制）构建而成。

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）。

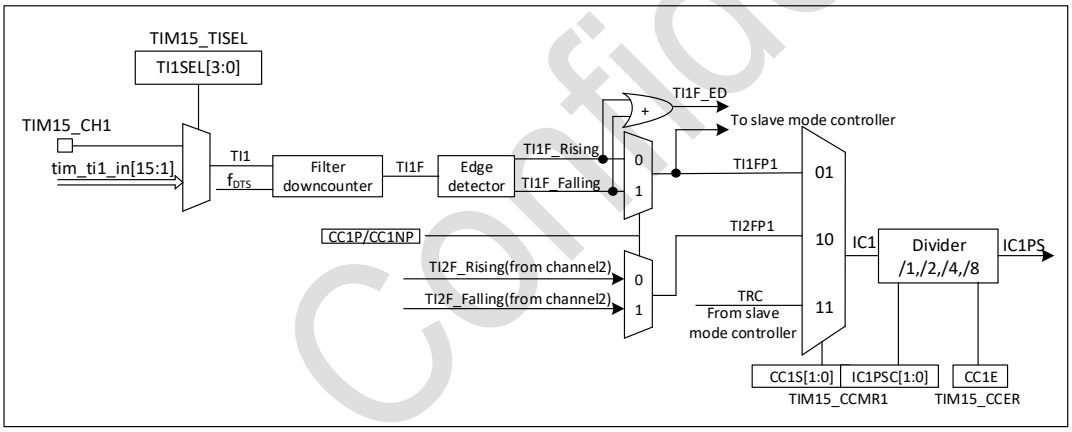


图 20-16 捕获/比较通道(示例：通道 1 输入阶段)

输出部分产生一个中间波形 OCxREF(高有效)作为基准，链的末端决定最终输出信号的极性。

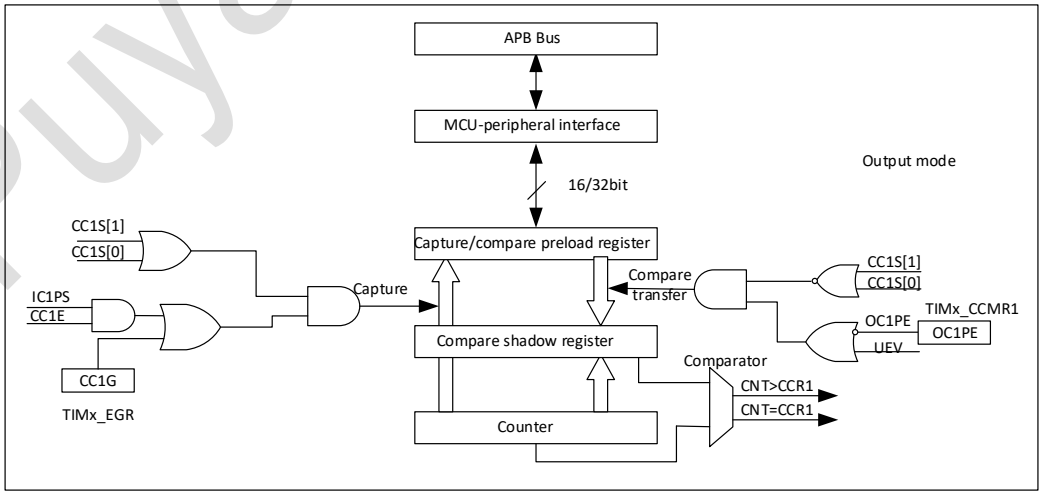


图 20-17 捕获/比较通道 1 主电路

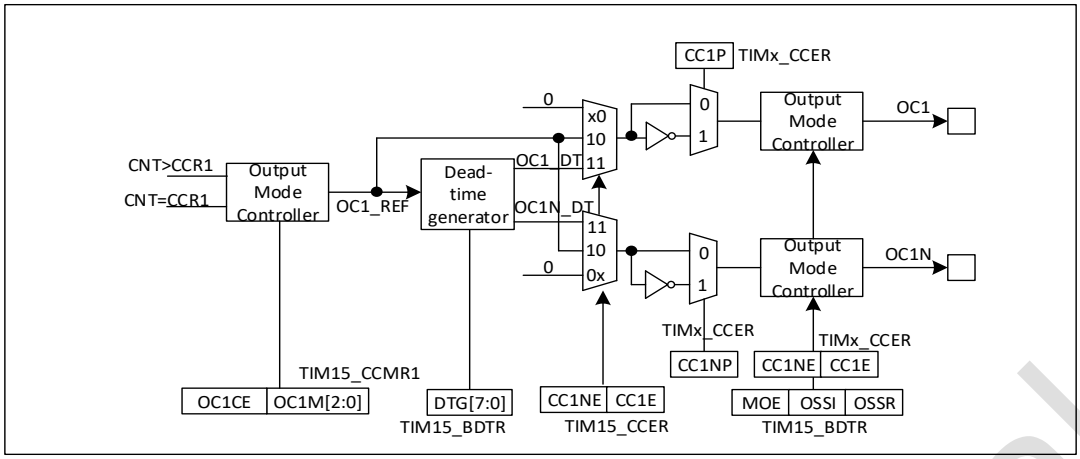


图 20-18 捕获/比较通道的输出阶段 (通道 1)

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。

20.2.7. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIM15_CCRx）中。当捕获事件发生时，相应的 CCxIF 标志（TIM15_SR 寄存器）被置 1，如果使能了中断或者 DMA 操作，则将产生中断或者 DMA 操作。如果捕获事件发生时，CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIM15_SR 寄存器）被置 1。写 CCxIF 可清除 CCxIF，或读取存储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 TIM15_CCR1 寄存器中，步骤如下：

- 使用 TIM1_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）
- 选择有效输入端：TIM15_CCR1 必须连接到 TI1 输入，所以写入 TIM15_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为 00 时，通道被配置为输入，并且 TM15_CCR1 寄存器变为只读。
- 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 TIM15_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；在检测到连续 8 个具有新电平的采样 (以 fDTS 频率) 后，可以确认在 TI1 上一次跳边沿，即在 TIM15_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIM15_CCER 寄存器中写入 CC1P=0 和 CC1NP=0（上升沿）
- 配置输入预分频器。在这个例子中，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 TIM15_CCMR1 寄存器的 IC1PSC=00）。
- 设置 TIM15_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 TIM15_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIM15_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM15_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1.
- 如设置了 CC1IE 位，则会产生一个中断请求。

- 如设置了 CC1DE 位，则会产生一个 DMA 请求

为了处理中断溢出，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中断溢出标志之后和读取数据之前可能产生的中断溢出信息。

注：设置 TIM15_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。

20.2.8. 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：

- 两个 ICx 信号被映射到同一个 Tix 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 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，当需要测量输入到 TI1 上的 PWM 信号的长度(TIM15_CCR1 寄存器)和占空比(TIM15_CCR2 寄存器)时，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

- 使用 TIM15_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）
- 选择 TIM15_CCR1 的有效输入：置 TIM15_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIM15_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIM15_CCR2 的有效输入：置 TIM15_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FPx 的有效极性(捕获数据到 TIM15_CCR2)：置 CC2P=1，CC2NP=0 (下降沿有效)。
- 选择有效的触发输入信号：置 TIM15_SMCR 寄存器中的 TS=00101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIM15_SMCR 中的 SMS=100。
- 使能捕获：置 TIM15_CCER 寄存器中 CC1E=1 且 CC2E=1。

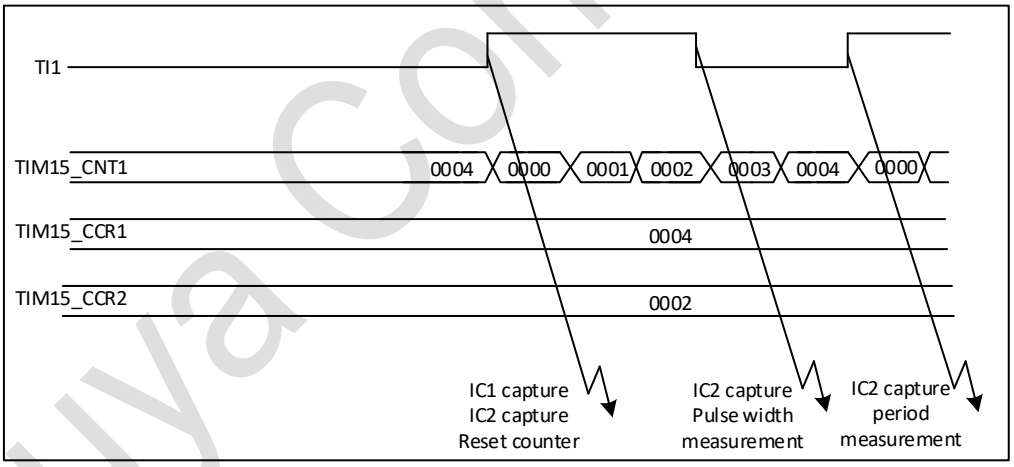


图 20-19 PWM 输入模式时序

20.2.9. 强制输出模式

在输出模式(TIM15_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 TIM15_CCMRx 寄存器中相应的 OCxM=0101，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。置 TIM15_CCMRx 寄存器中的 OCxM=0100，可强置 OCxREF 信号为低。

该模式下，在 TIM15_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。这将会在下面的输出比较模式一节中介绍。

20.2.10. 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，输出比较工作做如下操作：

- 将输出比较模式(TIM15_CCMRx 寄存器中的 OCxM 位)和输出极性(TIM15_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)。
- 设置中断状态寄存器中的标志位(TIM15_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断屏蔽(TIM15_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的 DMA 使能位(TIM15_DIER 寄存器中的 CCxDE 位，TIM1_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIM15_CCMRx 中的 OCxPE 位选择 TIM15_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。

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

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 TIM15_ARR 和 TIM15_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：
   - 设置 OCxM=0011，则计数器与 CCRx 匹配时翻转 OCx 的输出引脚
   - 置 OCxPE = 0 禁用预装载寄存器
   - 置 CCxP = 0 选择极性为高电平有效
   - 置 CCxE = 1 使能输出
5. 设置 TIM15_CR1 寄存器的 CEN 位启动计数器

TIM15_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 TIM15_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。

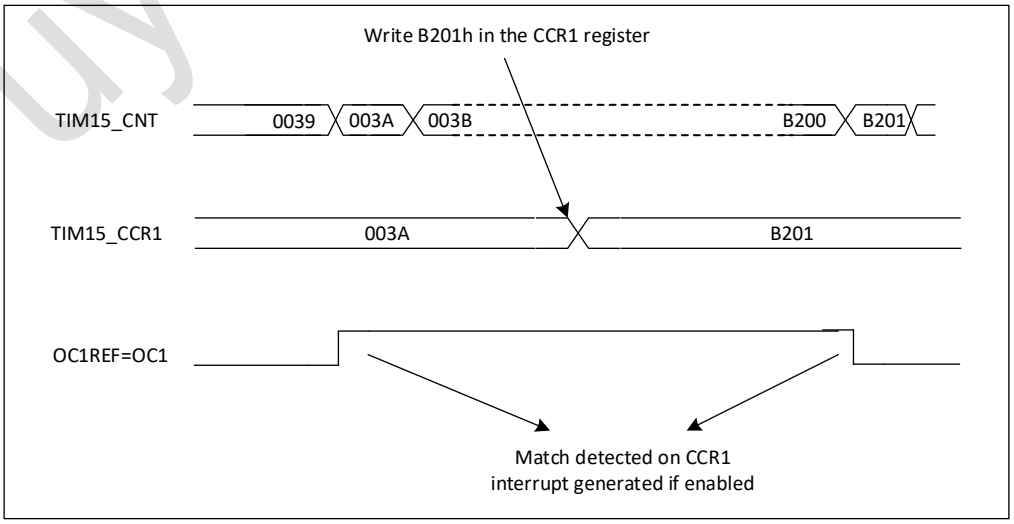


图 20-20 输出比较模式，翻转 OC1

20.2.11. PWM 模式

脉冲宽度调节模式可以产生一个由 TIM15_ARR 寄存器确定频率、由 TIM15_CCRx 寄存器确定占空比的信号。

在 TIM15_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 TIM15_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIM15_CR1 寄存器的 ARPE 位为 1 使能自动重装载的预装载寄存器，(在递增计数或中心对称模式中)。

由于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 TIM15_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 TIM15_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或者低电平有效。TIM15_CCER 寄存器中的 CCxE 位控制 OCx 输出使能。

在 PWM 模式（模式 1 或者模式 2），TIM15_CNT 和 TIM15_CCRx 始终在进行比较，以确定是否符合 TIM15_CNT ≤ TIM15_CCRx。

定时器当计数器是递增计数时能够产生边沿对齐模式的 PWM。

20.2.11.1. PWM 边沿对齐模式

下面是一个 PWM 模式 1 的例子。当 TIM15_CNT<TIM15_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM15_CCRx 中的比较值大于自动重装载值(TIM15_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。下图为 TIM15_ARR=8 时边沿对齐的 PWM 波形实例。

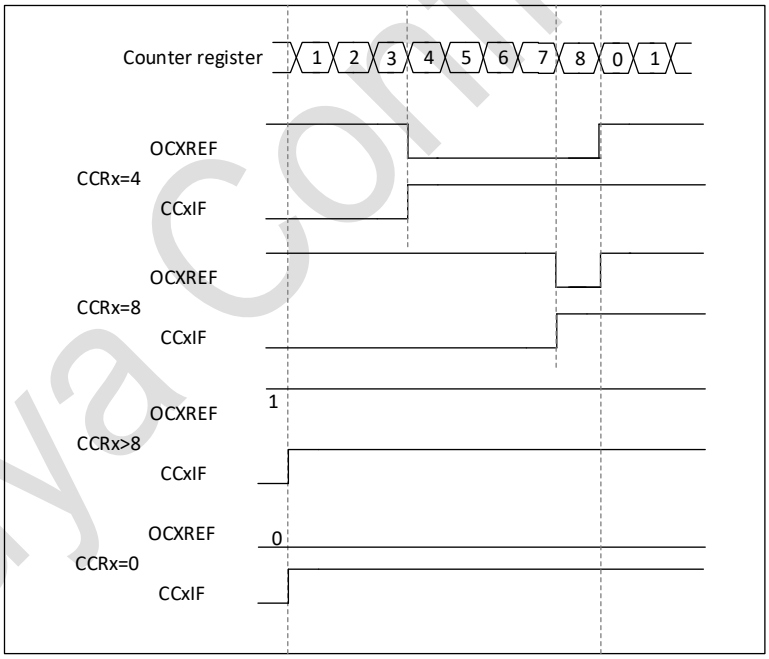


图 20-21 边沿对齐方式 PWM 输出波形（ARR=8）

20.2.12. 互补输出和死区插入

TIM15 能够输出一对互补信号，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，用户必须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 TIM15_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：TIM15_CCER 寄存器的 CCxE 和 CCxNE 位，TIM15_BDTR 和 TIM15_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 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 8 位的死区发生器 DTG[7:0]。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)

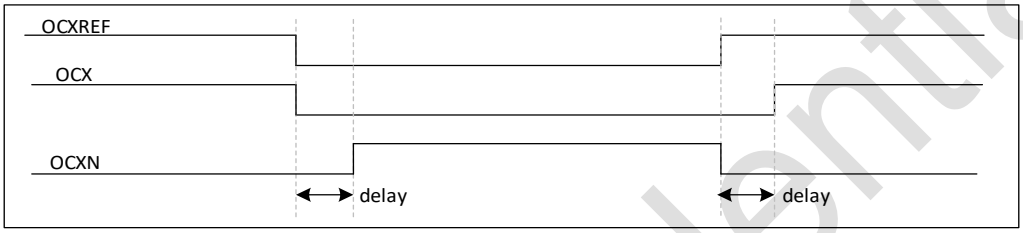


图 20-22 带死区插入的互补输出

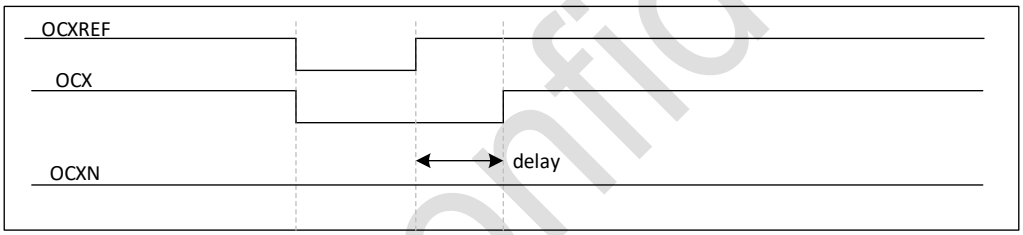


图 20-23 延迟时间大于负脉冲宽度的死区波形

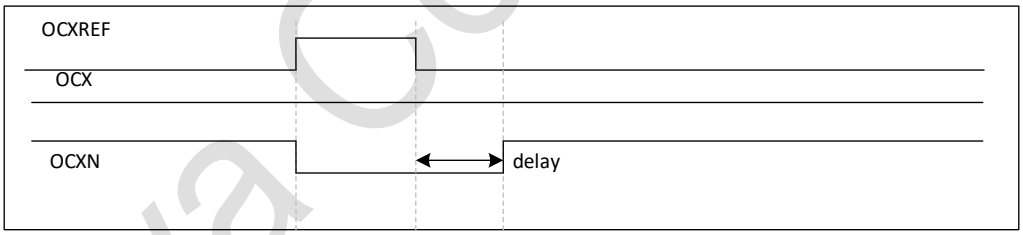


图 20-24 延迟时间大于正脉冲宽度的死区波形

每一个通道的死区延时都是相同的，是由 TIM15_BDTR 寄存器中的 DTG 位编程配置。

20.2.12.1. 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置模式、输出比较模式或 PWM 模式)，通过配置 TIM15_CCER 寄存器的 CCxE 和 CCxNE 位，可以将 OCxREF 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)，而同时使互补输出保持处于无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时保持有效电平和带死区的互补输出。

注：当只使能 OCxN(CCxE=0, CCxNE=1)时，两者不互补，当 OCxREF 有效时 OCxN 立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当同时使能 OCx 和 OCxN 时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 则相反，当 OCxREF 低时 OCxN 变为有效。

20.2.13. 使用刹车功能

刹车功能的目的是保护由 TIM15 定时器生成的 PWM 信号所驱动的功率开关。两个刹车输入通常连接到功率级和三相逆变器的故障输出。激活时，刹车电路会关闭 PWM 输出，并将其强制为预定义的安全状态。也可选择一些内部 MCU 事件来触发输出关断。

刹车通道源包括系统级故障（时钟失效和奇偶校验错误等）和应用故障（来自输入引脚和内置比较器），可以在死区持续时间后将输出强制为预定义的电平（有效或无效）。

刹车期间的输出使能信号和输出电平取决于多个控制位：

- TIM15_BDTR 寄存器中的 MOE 位，允许通过软件使能/禁止输出，在发生刹车事件时复位。
- TIM15_BDTR 寄存器中的 OSSI 位，定义定时器将输出控制在无效状态下，还是释放对 GPIO 控制器的控制（通常使其处于高阻态模式）
- TIM15_CR2 寄存器中的 OISx 和 OISxN 位，将输出设置为关断电平（有效或无效）。无论 OISx 和 OISxN 的值为何，均无法在给定时间将 OCx 和 OCxN 输出同时设置为有效电平。

退出复位状态后，刹车功能处于禁止状态，MOE 位处于低电平。通过设置 TIM15_BDTR 寄存器中的 BKE 位来使能刹车功能。可通过配置同一寄存器中的 BKP 位来选择刹车输入的极性。BKE 和 BKP 位可同时修改。

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 TIM1_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。

刹车源既可以是刹车输入引脚，或者以下内部源：

- CPU LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟错误事件
- 来自比较器的输出
- SRAM 字节校验错误
- FLASH ECC 错误

也可由软件通过 TIM15_EGR 寄存器中的 BG 位产生刹车事件。无论 BKE 使能位的值如何，都可以使用 BG 通过软件产生刹车。

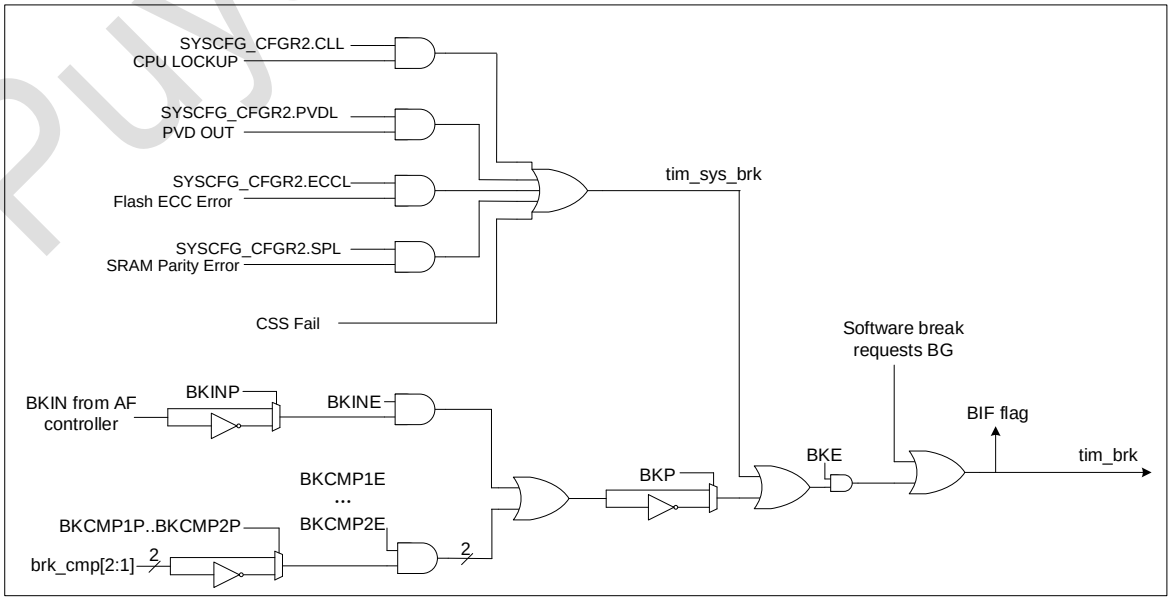


图 20-25 刹车电路图

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、空闲状态或者释放对 GPIO 的控制（由 OSSI 位选择)。这个特性在 MCU 的时钟振荡器关闭时依然有效。
- 一旦 MOE=0，将以 TIM15_CR2 寄存器中的 OISx 位设定的电平驱动每一个输出通道。如果 OSSI=0，则定时器释放输出控制（由 GPIO 控制器接管），否则使能输出始终为高电平。
- 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 个定时器的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出（由强制高阻态的 GPIO 控制器接管），否则使能输出将保持高电平或在 CCxE 或 CCxNE 位之一为高电平时立即变为高电平。
- 如果设置了 TIM15_DIER 寄存器中的 BIE 位，当刹车状态标志(TIM15_SR 寄存器中的 SBIF 和 BIF 位)为’1’时，则产生一个中断。
- 如果设置了 TIM15_BDTR 寄存器中的 AOE 位，则 MOE 位会在下一个更新事件（UEV）时自动再次置位；例如，这可以用来进行整形。否则，MOE 始终保持低电平直到应用将其再次置’1’；这种情况下，这个特性可以确保安全，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，不能设置 MOE 位为 1 (自动地或者通过软件)。同时，状态标志 BIF 不能被清除。

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 TIM15_BDTR 寄存器中的 BKE 位开启。

这里有两种方式产生刹车：

- 通过可编程极性的 BKR 输入，同时在 TIM15_BDTR 寄存器中使能 BKE
- 通过软件设置 TIM15_EGR 中的 BG 位。

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 TIM15_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种。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。

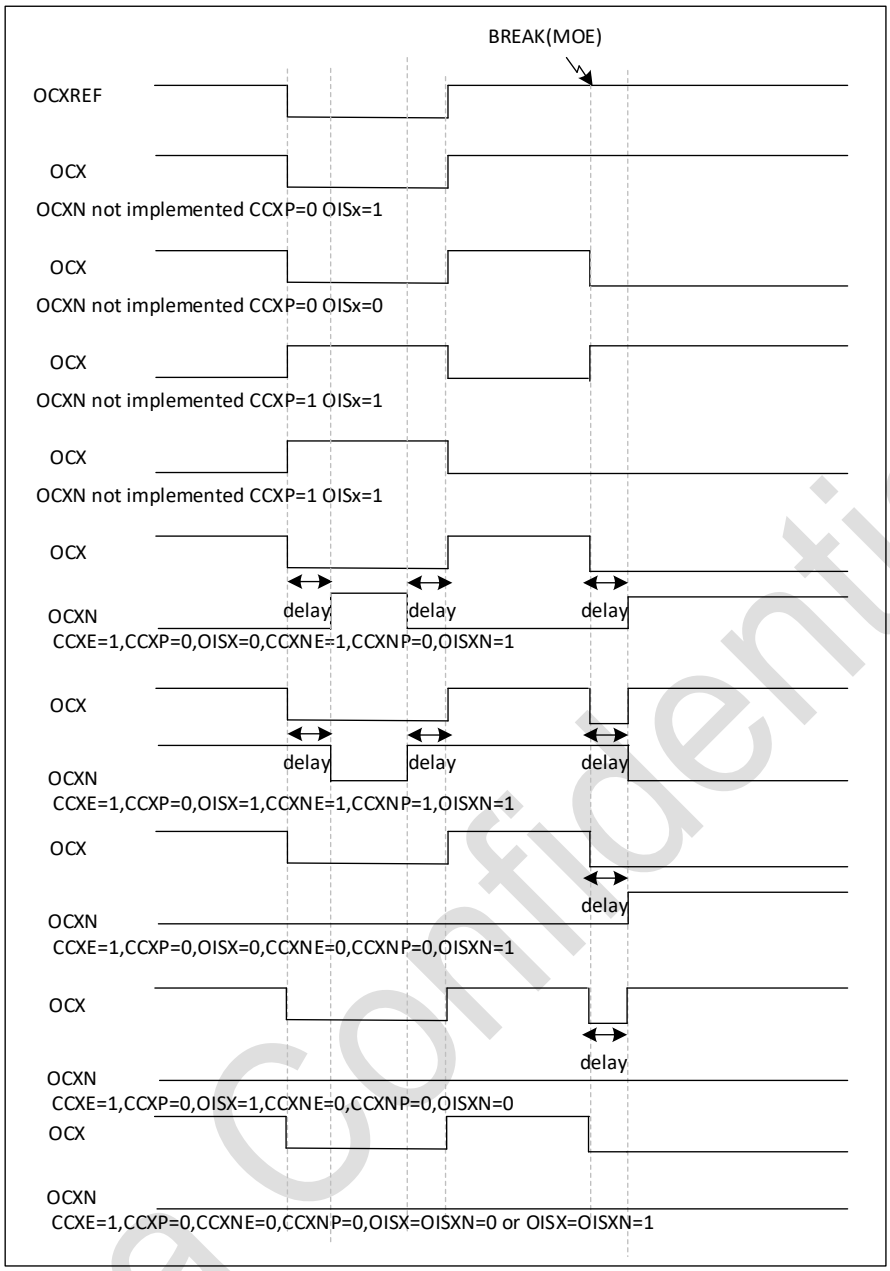


图 20-26 响应刹车事件的不同输出行为

20.2.14. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。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，产生一个脉宽可被程序控制的脉冲。

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 TIM15_CR1 寄存器的 OPM 位将选择单脉冲模式，这样可以让计数器自动的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：

- 递增计数模式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)

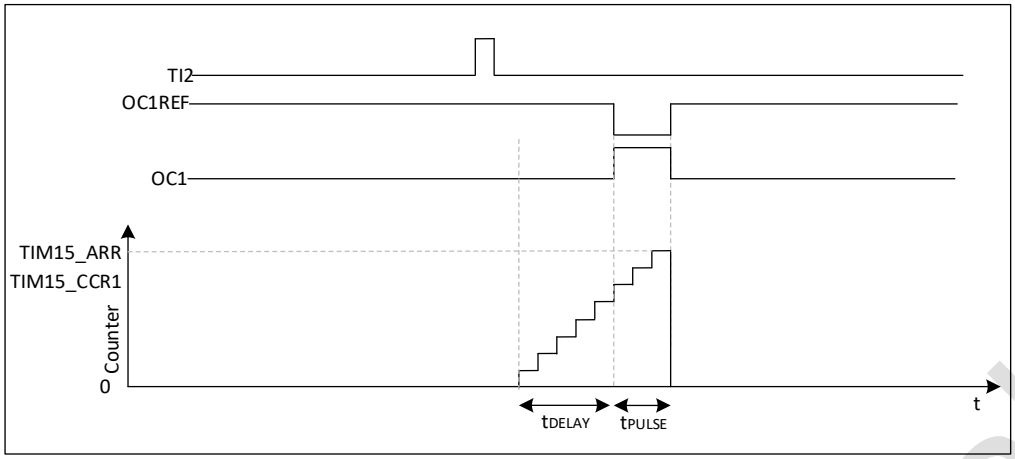


图 20-27 单脉冲模式示例

例如，当需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发源:

- 使用 TIM15_TISEL 寄存器中的 TI2SEL[3:0]位选择正确的 TI2x 源
- 置 TIM15_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIM15_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 置 TIM15_SMCR 寄存器中的 TS=00110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIM15_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)

- t_{DELAY} 由 TIM15_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIM15_ARR – TIM15_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 TIM15_CCMR1 寄存器的 OC1M=0111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIM15_CCMR1 中的 OC1PE=1 和 TIM15_CR1 寄存器中的 ARPE；然后在 TIM15_CCR1 寄存器中填写比较值，在 TIM15_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，TIM15_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 TIM15_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。TIM15_CR1 寄存器中的 OPM=0 时，即选择重复模式。

20.2.15. 定时器和外部触发同步

TIM15 定时器能够在多种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。

20.2.15.1. 从模式：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 TIM15_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(TIM15_ARR，TIM15_CCRx)都将更新。

在以下的例子中，TI1 输入端的上升沿导致递增计数器被清零:

- 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)。由于捕获预分频器不用于触发操作，所以不需要配置。CC1S 位只选择输入捕获源，即 TIM15_CCMR1 寄存器中 CC1S=01。置 TIM15_CCER 寄存器中 CC1P=0（和 CC1NP=0）以确定极性(只检测上升沿)。

- 置 TIM15_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM15_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIM15_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(TIM15_SR 寄存器中的 TIF 位)被设置，根据 TIM15_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 TIM15_ARR=0x36 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。

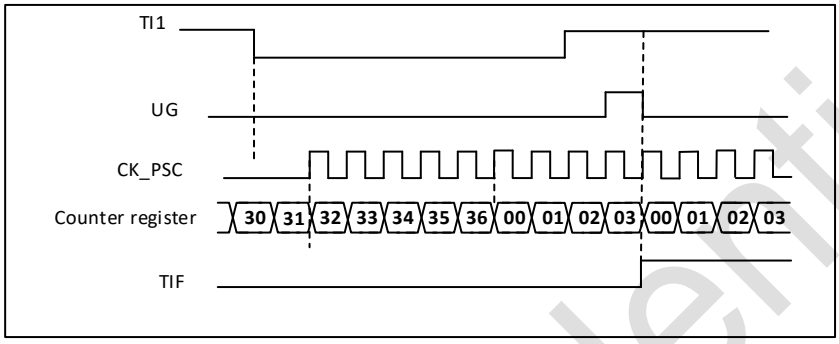


图 20-28 复位模式下的控制电路

20.2.15.2. 从模式：门控模式

按照选中的输入端的电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时递增计数：

- 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 TIM15_CCMR1 寄存器中 CC1S=01。置 TIM15_CCER 寄存器中 CC1P=1(和 CC1NP=0)以确定极性(只检测低电平)。
- 置 TIM15_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM15_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIM15_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则不论触发输入电平如何，计数器不能启动。

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 TIM15_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。

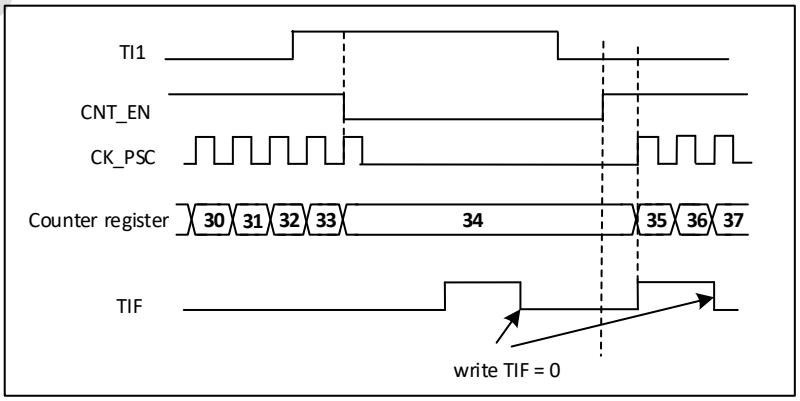


图 20-29 门控模式下的控制电路

20.2.15.3. 从模式：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 输入的上升沿开始递增计数：

- 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 TIM15_CCMR1 寄存器中 CC2S=01。置 TIM15_CCER 寄存器中 CC2P=1(和 CC2NP=0)以确定极性(只检测低电平)。
- 置 TIM15_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM15_SMCR 寄存器中 TS=00110，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。

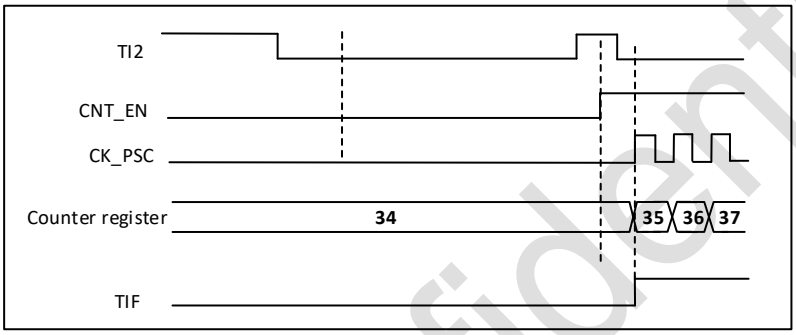


图 20-30 触发模式下的控制电路

20.2.15.4. 从模式：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 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 TIM15_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 的每一个上升沿递增计数一次：

1. 通过 TIM15_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 TIM15_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 置 TIM15_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)
3. 置 TIM15_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIM15_SMCR 寄存器中 TS=00101，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。

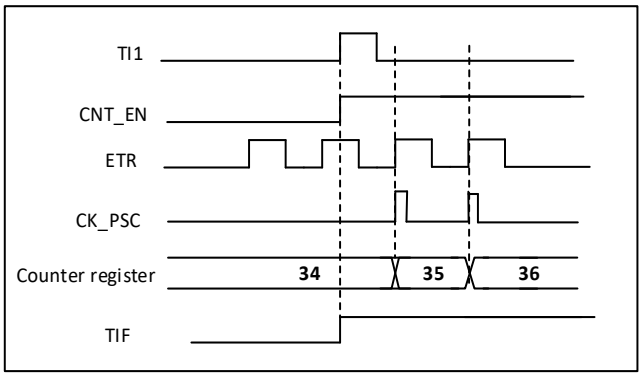


图 20-31 外部时钟模式 2+触发模式下的控制电路

20.2.16. 定时器同步

不同定时器在内部相连，用于定时器的同步或者链接功能。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或时钟等操作。

具体描述参考 TIM2/TIM3 中相关章节。

20.3. TIM15 中断

表 20-1 TIM15 中断

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
COM	COMIF	COMIE
触发	TIF	TIE
刹车	BIF	BIE

20.4. TIM15 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIM15_STOP 的设置，TIM15 计数器可以继续正常工作或者停止工作。

20.5. TIM15 寄存器

20.5.1. TIM15 控制寄存器 1 (TIM15_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
						RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	2'b0	时钟分频因子。 这 2 位定义定时器时钟(CK_INT)频率与死区发生器以及数字滤波器(ETR,TIx)所用的死区及采样时钟（t_{DTS}）之间的分频比例。

				00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载使能 0：TIM15_ARR 寄存器没有缓冲 1：TIM15_ARR 寄存器被装入缓冲器
6:4	Reserved	-	-	保留
3	OPM	RW	0	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	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： - 计数器上溢/下溢 - 设置 UG 位 - 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	更新禁止 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 影子寄存器装入预装载值。 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，则重新初始化计数器和预分频器。
0	CEN	RW	0	计数器使能。 0：禁止计数器 1：使能计数器 注：在软件设置了 CEN 位后，才可以使用外部时钟、门控模式和编码器模式。触发模式可以通过硬件自动地设置 CEN 位为 1。

20.5.2. TIM15 控制寄存器 2 (TIM15_CR2)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	OIS2	OIS1N	OIS1	Res.	MMS[2:0]			CCDS	CCUS	Res.	CCPC
					RW	RW	RW		RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留

10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。定义参考 OIS1.
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出)。 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(TIM15_BDTR 寄存器)级别 1、2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出)。 0：当 MOE=0 时，死区后 OC1=0 1：当 MOE=0 时，死区后 OC1=1 注：已经设置了 LOCK(TIM15_BDTR 寄存器)级别 1、2 或 3 后，该位不能被修改。
7	Reserved	-	-	保留
6:4	MMS	RW	0	主模式选择。 这些位可选择主模式下将要发送到从定时器以实现同步的信息(TRGO)。可能的组合如下： 0000：复位---TIM1_EGR 寄存器的 UG 位作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位会有一个延迟。 0001：使能---计数器使能信号 CNT_EN 作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或在一段时间内控制从定时器。当配置为门控模式时，计数器使能是通过 CEN 控制位和触发输入信号的逻辑**与**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 TIM1_SMCR 寄存器中 MSM 位的描述)。 0010：更新---更新事件作为触发输出(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。 0011：比较脉冲---一旦发生一次捕获或一次比较成功时，当 CC1IF 标志被置为 1 时，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)。 其他值：保留
3	CCDS	RW	0	捕获/比较的 DMA 选择 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	0	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COMG 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 COMG 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	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位 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，仅当发生换向事件（COM)（COMG 位置 1 或在 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。 注：该位只对具有互补输出的通道起作用。

20.5.3. TIM15 从模式控制寄存器（TIM15_SMCR）

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[5:4]		Res.	Res.	Res.	Res.
										RW	RW				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Res.	SMS[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[5:4]	RW	000	详见 TS 描述
19:16	Reserved	-	-	保留
15	ETP	RW	0	外部触发极性。该位选择是否 ETR 或者 ETR 的反向被用作触发操作。 0：ETR 不进行反相，高电平或者上升沿有效 1：ETR 反相，低电平或者下降沿有效
14	ECE	RW	0	外部时钟使能。这位使能外部时钟模式 2 0：外部时钟模式 2 不使能 1：外部时钟模式 2 使能，计数器工作在 ETRF 信号的有效沿 注： 1：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=00111）具有相同效果。 2：外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 TRGI 不得连接 ETRF（TS 位不得为 00111）。 3：如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。
13:12	ETPS[1:0]	RW	00	外部触发预分频器。 外部触发信号 ETR 频率不能超过 TIM1CLK 频率的 1/4。可以通过使能预分频器来降低 ETR 的频率。该方法在输入快速外部时钟时非常有用。 00：预分频器关闭 01：ETR 频率的 2 分频 10：ETR 频率的 4 分频 11：ETR 频率的 8 分频
11:8	ETF[3:0]	RW	0000	外部触发滤波。 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 个事件后才视为一个有效输出边沿。 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=8

				1000：采样频率 $f_{SAMPLING}$= f_{DTS}/8，N=6 1001：采样频率 $f_{SAMPLING}$= f_{DTS}/8， N=8 1010：采样频率 $f_{SAMPLING}$= f_{DTS}/16, N=5 1011：采样频率 $f_{SAMPLING}$= f_{DTS}/16，N=6 1100：采样频率 $f_{SAMPLING}$=f_{DTS}/16，N=8 1101：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=5 1110：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=6 1111：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=8
7	MSM	RW	0	主/从模式。 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。 此设置适用于由单个外部事件对多个定时器进行同步的情况。
6:4	TS[2:0]	RW	0	触发选择 这 5 位选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测（TI1F_ED） 00101：滤波后的定时器输入 1（TI1FP1） 00110：滤波后的定时器输入 2（TI1FP2） 00111：外部触发输入（ETRF） 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 其它：保留 注：这些位只能在未使用(如 SMS=0000)时更改，以避免在转换时产生错误的边沿检测。 注：ITRx 的映射参见“外设互联”章节。
3	Reserved	-	-	保留
2:0	SMS[2:0]	RW	0	从模式选择 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：禁止从模式---如果 CEN=1，则预分频器直接由内部时钟驱动。 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)，只控制计数器的启动。 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=00100)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。

20.5.4. TIM15 DMA/中断使能寄存器 (TIM15_DIER)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDE	COMDE	Res.	Res.	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	Res.	Res.	CC2IE	CC1IE	UIE
	RW	RW			RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
14	TDE	RW	0	触发 DMA 请求使能 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	COM 的 DMA 请求使能 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
12:11	Reserved	-	-	保留
10	CC2DE	RW	0	捕获/比较 2 的 DMA 请求使能 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	0	捕获/比较 1 的 DMA 请求使能 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	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	刹车中断使能 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	触发中断使能 0：禁止触发中断 1：允许触发中断
5	COMIE	RW	0	COM 中断使能 0：禁止 COM 中断 1：允许 COM 中断
4:3	Reserved	-	-	保留
2	CC2IE	RW	0	捕获/比较 2 中断使能 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	捕获/比较 1 中断使能 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

20.5.5. TIM15 状态寄存器 (TIM15_SR)

偏移地址:0x010

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	Res.	Res.	CC2IF	CC1IF	UIF
					RC_W0	RC_W0		RC_W0	RC_W0	RC_W0			RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	CC2OF	RC_W0	0	捕获/比较 2 过捕获标志。 参见 CC1OF 描述
9	CC1OF	RC_W0	0	捕获/比较 1 过捕获标志。 仅当相应的通道被配置为输入捕获时，该标志可由硬件置 1。写 0 可清除该位。 0：无过捕获产生； 1：CC1IF 置 1 时，TIM15_CCR1 寄存器已经捕获到计数器的值且 CC1IF 已经置位。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标志。 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。如果 TIM1_DIER 寄存器中 BIE=1，则会生成中断。
6	TIF	RC_W0	0	触发器中断标志。 除门控模式外的其它所有模式下，当使能从模式控制器后，在 TRGI 输入端检测到有效边沿时，由硬件对该位置 1。选择门控模式时，该标志将在计数器启动或停止时置 1，但需要通过软件清零。 0：无触发器事件产生； 1：触发器中断等待响应
5	COMIF	RC_W0	0	COM 中断标志 一旦产生 COM 事件（当 CCxE、 CCxNE、 OCxM 已更新时）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	Reserved	-	-	保留
2	CC2IF	RC_W0	0	捕获/比较 2 中断标志。 参考 CC1IF 描述。
1	CC1IF	RC_W0	0	捕获/比较 1 中断标志 **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。它由软件清 0。 0：无匹配发生； 1：TIM15_CNT 的值与 TIM15_CCR1 的值匹配。

				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 TIM15_CCR1 清 0。 0：无输入捕获产生； 1：输入捕获产生并且计数器值已装入 TIM15_CCR1 (在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标志。 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：更新事件等待响应。当寄存器被更新时该位由硬件置 1: − 若 TIM15_CR1 寄存器的 UDIS=0，产生更新事件(计数器上溢)； − 若 TIM15_CR1 寄存器的 UDIS=0、URS=0，当 TIM15_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

20.5.6. TIM15 事件产生寄存器(TIM15_EGR)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	Res.	Res.	CC2G	CC1G	UG
								W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件。 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	W	0	产生触发事件。 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：TIM15_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	W	0	捕获/比较事件，产生控制更新。 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：该位只对有互补输出的通道有效。
4:3	Reserved	-	-	保留
2	CC2G	W	0	产生捕获/比较 2 事件。 参考 CC1G 描述。
1	CC1G	W	0	产生捕获/比较 1 事件。 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： CC1IF 置位，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**： 当前的计数器值捕获至 TIM15_CCR1 寄存器，CC1IF 置位，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	W	0	产生更新事件。 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)。

20.5.7. TIM15 捕获/比较模式寄存器 (TIM15_CCMR1)

偏移地址:0x18

复位值:0x0000 0000

同一寄存器可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获和输出比较模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		Res.	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

20.5.7.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F[3:0]	RW	4’b0	输入捕获 2 滤波器 参见 IC1F[3:0]的描述。
11:10	IC2PSC[1:0]	RW	2’b0	输入/捕获 2 预分频器 参见 IC1PSC[1:0]的描述。
9:8	CC2S[1:0]	RW	2’b0	捕获/比较 2 选择 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。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发器输入时有效。 注：CC2S 仅在通道关闭时(TIM15_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	4’b0	输入捕获 1 滤波器 该位域定义了 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿： 0000：无滤波器，以 f_{DTS} 采样

				0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC[1:0]	RW	2’b0	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM15_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[1:0]	RW	2’b0	CC1S[1:0]：捕获/比较 1 选择。 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。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发器输入时有效。 注： CC1S 仅在通道关闭时(TIM15_CCER 寄存器的 CC1E=0)才是可写的。

20.5.7.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14:12	OC2M[2:0]	RW	3’b0	输出比较 2 模式选择 参见 OC1M 的描述。
11	OC2PE	RW	0	输出比较 2 预装载使能。 参见 OC1PE 的描述。
10	OC2FE	RW	0	输出比较 2 快速使能。 参见 OC1FE 的描述。
9:8	CC2S	RW	0	捕获/比较 2 选择。 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。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发器输入时有效。 注：CC2S 仅在通道关闭时(TIM15_CCER 寄存器的 CC2E=0)才是可写的。
7	Reserved	-	-	保留
6:4	OC1M[2:0]	RW	00	输出比较 1 模式 该位定义了 OC1、OC1N 的输出参考信号 OC1REF 的行为。OC1REF 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 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用； 001 ： 设置通道 1 为匹配时输出有效电平。当计数器 TIM15_CNT 的值与捕获/比较寄存器 1(TIM15_CCR1)匹配时，强制 OC1REF 为高电平。 010 ： 设置通道 1 为匹配时输出无效电平。当计数器 TIM15_CNT 的值与捕获/比较寄存器 1(TIM15_CCR1)匹配时，强制 OC1REF 为低电平。 011：翻转。当 TIM15_CCR1=TIM15_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---在递增计数时，一旦 TIM15_CNT<TIM15_CCR1 时通道 1 为有效电平，否则 为无效电平； 111： PWM 模式 2---在递增计数时，一旦 TIM15_CNT<TIM15_CCR1 时通道 1 为无效电平，否则为有效电平； 注 1：一旦 LOCK 级别设为 3(TIM15_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	0	输出比较 1 预装载使能 0：禁止 TIM15_CCR1 寄存器的预装载功能，可随时写入 TIM15_CCR1 寄存器，且新值马上起作用。 1：开启 TIM15_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM15_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIM15_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，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	0	输出比较 1 快速使能。 该位用于加快触发器输入事件对 CC 输出的影响。

				0：即使触发器开启，CC1 也将根据计数器和 CCR1 的值正常操作。当触发器的输入出现边沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：触发输入上出现有效边沿相当于 CC1 输出上发生比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和 CC1 输出间的延时被缩短为 3 个时钟周期。 只在通道被配置成 PWM1 或 PWM2 模式时 OCxFE 才起作用。
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 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 TIM15_SMCR 寄存器的 TS 位选择）。 注： CC1S 仅在通道关闭时(TIM15_CCER 寄存器的 CC1E=0)才是可写的。

20.5.8. TIM15 捕获/比较使能寄存器 (TIM15_CCER)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2NP	Res.	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	CC2NP	RW	0	[CC2NP,CC2P]位选择作为触发或捕获信号的 TI1FP2 和 TI2FP2 的极性。 参考 CC1P 的描述。
6	Reserved	-	-	保留
5	CC2P	RW	0	输入/捕获 2 输出极性 参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能 参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(TIM15_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	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，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，CC1E 位的值。

				此位将在具有互补输出的通道上进行预装载。如果 TIM1_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件（COM）时，CC1NE 有效位才会从预装载位获取新值。
1	CC1P	RW	0	输入/捕获 1 输出极性 **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]位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性： 00：非反相/上升沿。TIxFP1 上升沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式、编码器模式）。 01：反相/下降沿。TIxFP1 下降沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：非反相/双沿。TIxFP1 上升和下降沿都有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）；TIxFP1 非反相（门控模式）。这个配置不能应用于编码器模式下。 注 1：对于互补输出通道，这一位是预载的。如果 TIM1_CR2 寄存器中的 CCPC 位置 1，那么 CC1P 的实际有效位只有在 COM 事件发生时才会加载预载值。 注 2：一旦 LOCK 级别(TIM15_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。
0	CC1E	RW	0	输入/捕获 1 输出使能 **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、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、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 TIM15_CCR1 寄存器。 0：捕获禁止 1：捕获使能 注：对于互补输出通道，这一位是预载的。如果 TIM1_CR2 寄存器中的 CC1PC 位被设置，那么 CC1E 的实际有效位只有在 COM 事件发生时才会加载预载值。

表 20-2 具有刹车功能的互补通道 OCx 和 OCxN 的输出控制位

控制位					输出状态	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
1	x	x	0	0	输出禁止(不由定时器驱动)，OCx=0, OCx_EN=0	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0

		0	0	1	禁止输出（不由定时器驱动），OCx=0，OCx_EN=0	OCxREF+极性 OCxN=OCxREF 异或 CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性 OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
		x	1	1	OCREF+极性+死区，OCx_EN=1	OCREF 的互补（not OCREF）+ 极性 +死区，OCxN_EN=1
		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 异或 CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性 OCx=OCxREF 异或 CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平），OCxN=CCxNP, OCxN_EN=1
0	0	x	x	x	输出禁止(不由定时器驱动)	
	1		0	0		
	1		0	1	关闭状态（输出使能且为无效电平） 异步：OCx=CCxP, OCx_EN=1，OCxN=CCxNP, OCxN_EN=1（如果触发 BRK）。 随后（仅当触发 BRK 时才有效），若时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，则 OCx=OISx 且 OCxN=OISxN。	
	1		1	0		
	1	x	1	1		

20.5.9. TIM15 计数器寄存器(TIM15_CNT)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	0	计数器的值

20.5.10. TIM15 预分频寄存器(TIM15_PSC)

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	0	预分频器值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时（包括计数器通过 TIM15_EGR 寄存器中的 UG 位清零时，或在配置为“复位模

				式”时通过触发控制器清零时）要装载到有效预分频器寄存器的值。

20.5.11. TIM15 自动装载寄存器 (TIM15_ARR)

偏移地址:0x2c

复位值:0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	0xFFFF	自动重装载的值 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。

20.5.12. TIM15 重复计数器 寄存器(TIM15_RCR)

偏移地址:0x30

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	REP[7:0]	RW	8’b0	重复计数器的值 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）；使能更新中断时，则同时会影响产生更新中断的速率。 每次递减计数器 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 只有在重复更新事件（UIF）发生时才重载 REP 值，因此对 TIM1_RCR 寄存器写入的新值只在下次重复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;

20.5.13. TIM15 捕获/比较寄存器 1(TIM15_CCR1)

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留

15:0	CCR1[15:0]	RW	0	捕获/比较 1 的值 **若 CC1 通道配置为输出**: CCR1 包含要装载的当前捕获/比较 1 寄存器的值 (预装载值)。如果在 TIM15_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际捕获/比较寄存器包含要与计数器 TIM15_CNT 比较的值，并且在 OC1 端口上输出信号。 **若 CC1 通道配置为输入**: CCR1 为上一次输入捕获 1 事件（IC1）发生时的计数器值。此时，只能读取 TIM15_CCR1 寄存器，无法对其进行编程。

20.5.14. TIM15 捕获/比较寄存器 2(TIM15_CCR2)

偏移地址:0x38

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2[15:0]	RW	0	捕获/比较 2 的值 **若 CC2 通道配置为输出**: CCR2 包含要装载的当前捕获/比较 2 寄存器的值（预装载值）。如果在 TIM15_CCMR1 寄存器(OC2PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。实际捕获/比较寄存器包含要与计数器 TIM15_CNT 比较的值，并且在 OC2 端口上输出信号。 **若 CC2 通道配置为输入**: CCR2 为上一次输入捕获 2 事件（IC2）发生时的计数器值。此时，只能读取 TIM15_CCR2 寄存器，无法对其进行编程。

20.5.15. TIM15 刹车和死区寄存器(TIM15_BDTR)

偏移地址:0x44

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。此位由软件置 1，也可根据 AOE 位状态自动置 1。它仅对配置为输出通道有效。

				0：禁止 OC 和 OCN 输出或强制为空闲状态，具体取决于 OSSI 位； 1：如果设置了相应的使能位（TIM15_CCER 寄存器的 CCxE、CCxNE 位），则使能 OC 和 OCN 输出。
14	AOE	RW	0	自动输出使能 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自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 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(TIM15_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入； 1：开启刹车输入。 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
11	OSSR	RW	0	运行模式下“关闭状态”选择 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：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，开启 OC/OCN 输出并输出无效电平。 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 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：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其无效电平。 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	2’b0	锁定设置 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 TIM15_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、TIM15_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，TIM15_CCER 寄存器的 CCxP/CCxNP 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，TIM15_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 TIM15_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	8’b0	死区发生器设置。 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] × T_{dtg}， T_{dtg} = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG[5:0]) ×T_{dtg}，T_{dtg} = 2 ×T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 8 ×T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 16 ×T_{DTS}； 例：若 T_{DTS} = 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(TIM15_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则这些位不能被修改。

20.5.16. TIM15 输入选择寄存器（TIM15_TISEL）

偏移地址:0x5C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:8	TI2SEL[3:0]	RW	4’b0	TI2 输入选择. 0000：TIM15_CH2 0001：COMP2_OUT 0010：COMP1_OUT 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL[3:0]	RW	4’b0	TI1 输入选择. 0000：TIM15_CH1 0001：LSE_CSS 0010：COMP1_OUT 0011：COMP2_OUT 其他：保留

20.5.17. TIM15 备用选项寄存器 1（TIM15_AF1）

偏移地址:0x60

复位值:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[1:0]		Res.	Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW			RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:14	ETRSEL[3:0]	RW	4'b0	外部触发源选择 该位段用于选择 ETR 输入源 0000：TIM15_ETR 0001：COMP1_OUT 0010：COMP2_OUT 0011：保留 0100：保留 0101：ADC_AWD 其它：保留 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
13:12	Reserved	-	-	保留
11	BKCMP2P	RW	0	来源于比较器 2 的刹车输入极性控制。 该位选择来源于比较器 2 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 2 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 2 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	来源于比较器 1 的刹车输入极性控制。 该位选择来源于比较器 1 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 1 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 1 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效）

				注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	比较器 2 刹车输入使能。 比较器 2 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 2 刹车输入关闭 1：比较器 2 刹车输入开启 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	比较器 1 刹车输入使能。 比较器 1 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 1 刹车输入关闭 1：比较器 1 刹车输入开启 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能。 比较器 1 刹车 BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIM15_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

20.5.18. TIM15 DMA 控制寄存器 (TIM15_DCR)

偏移地址:0x3DC

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	复位值	Function
31:13	Reserved	-	-	保留
12:8	DBL[4:0]	RW	5’b0	DMA 连续传送长度 这些位定义了 DMA 的传送长度（当对 TIM15_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送），即：定义被传送的字节数目： 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输
7:5	Reserved	-	-	保留
4:0	DBA[4:0]	RW	5’b0	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIM15_DMAR 寄存器的地址进行读或写时），DBA 定义为从 TIM15_CR1 寄存器所在地址开始的偏移量： 00000：TIM15_CR1，

				00001：TIM15_CR2， 00010：TIM15_SMCR， .....

20.5.19. TIM15 DMA 连续传输地址寄存器(TIM15_DMAR)

偏移地址:0x3E0

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA 连续传送寄存器 对 TIM15_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIM15_CR1 地址 + (DBA + DMA 指针)x4，其中： “TIM15_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIM15_DCR 寄存器中定义的基地址； “DMA 指针”是由 DMA 自动控制的偏移量，它取决于 TIM15_DCR 寄存器中定义的 DBL。

21. 通用定时器(TIM16/17)

本项目的 TIM16 和 TIM17 功能完全一样。

21.1. TIM16 /TIM17 主要特性

- 16 位自动装载递增计数器
- 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
- 一个通道作为：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿对齐模式）
- 带可编程死区时间的互补输出
- 重复计数器，在计数指定周期数后，才更新定时器的寄存器
- 刹车输入可以将定时器的输出信号置为用户可选的安全配置中
- 如下时间发生时产生中断/DMA：
  - 更新：计数器上溢
  - 输入捕获
  - 输出比较
  - 刹车输入

21.2. TIM16/TIM17 功能描述

21.2.1. 功能框图

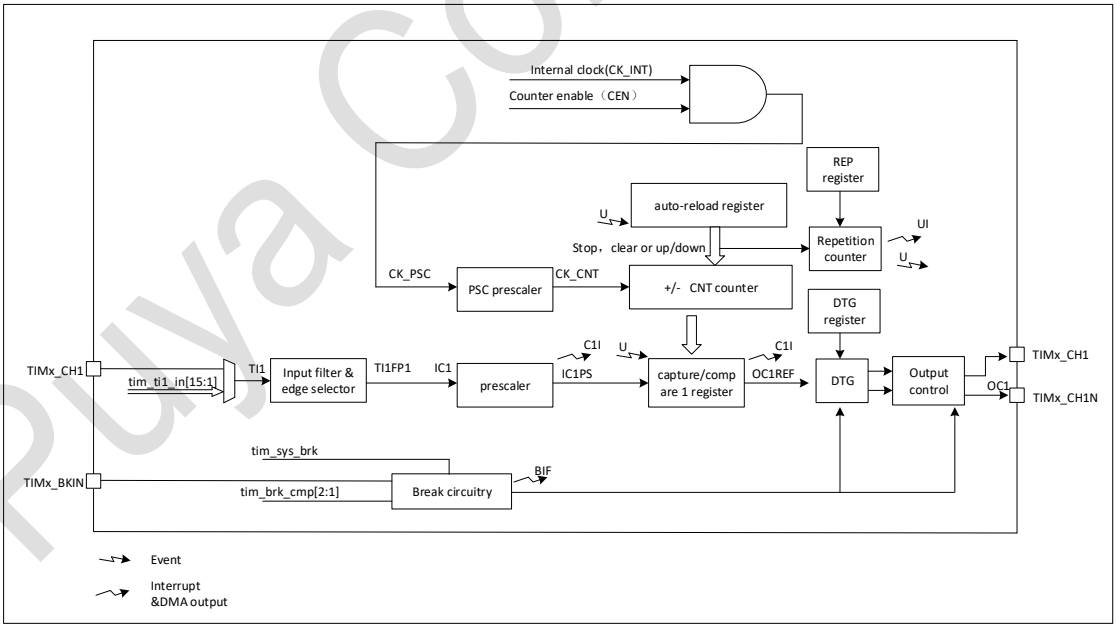


图 21-1 TIM16 和 TIM17 框图

21.2.2. 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位的递增计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包括：

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

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位(CEN)置 1 时，才会启动计数器。

注意，计数器将在 TIMx_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。

21.2.2.1. 预分频描述

预分频器可以将计数器的时钟按 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。

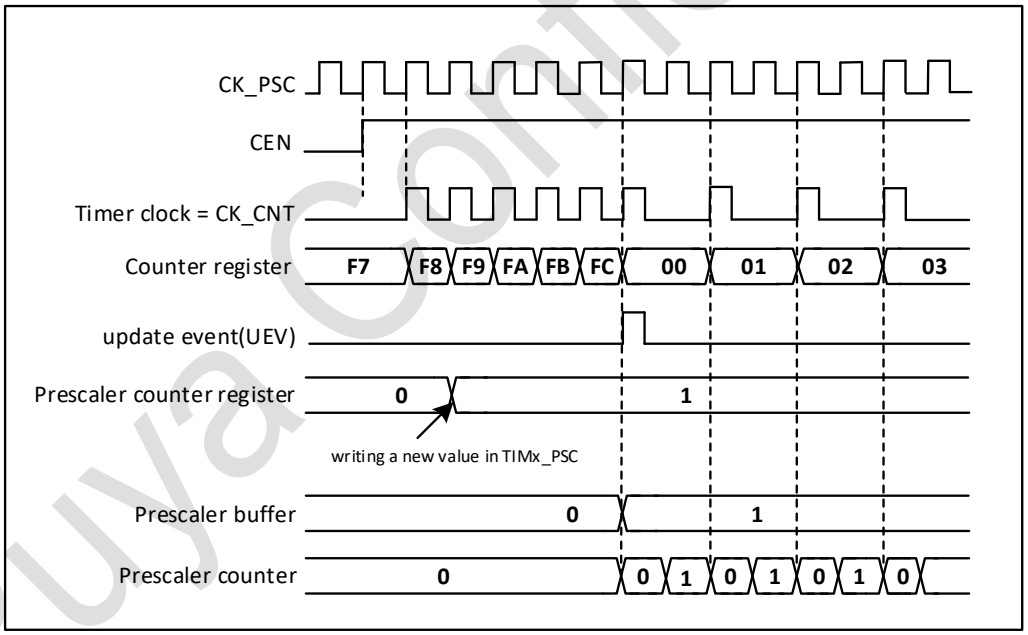


图 21-2 预分频器分频由 1 变为 2 时的计数器时序图

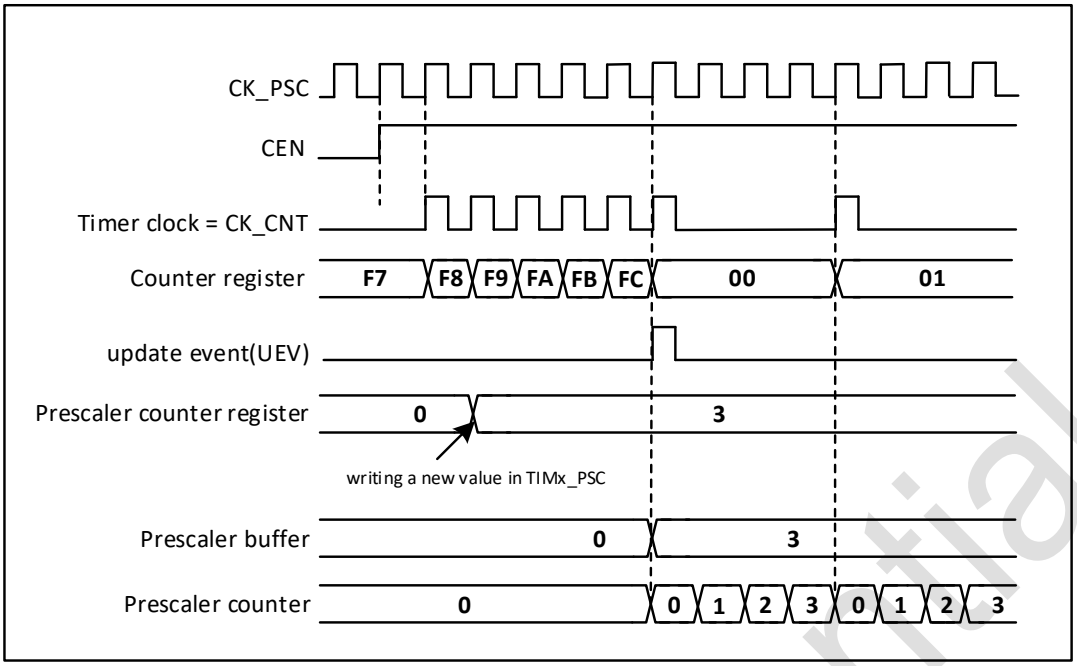


图 21-3 预分频器分频由 1 变为 4 时的计数器时序图

21.2.3. 计数模式（递增）

递增计数模式，是从 0 计数到自动装载值（TIMx_ARR 寄存器），然后又从 0 重新开始计数，并产生一个计数的溢出事件。

在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件（UEV）；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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）。

- 重复计数器被 TIMx_RCR 寄存器中的值重新装载。
- 自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图显示了几个在不同频率下的计数器行为，当 TIMx_ARR=0x36。

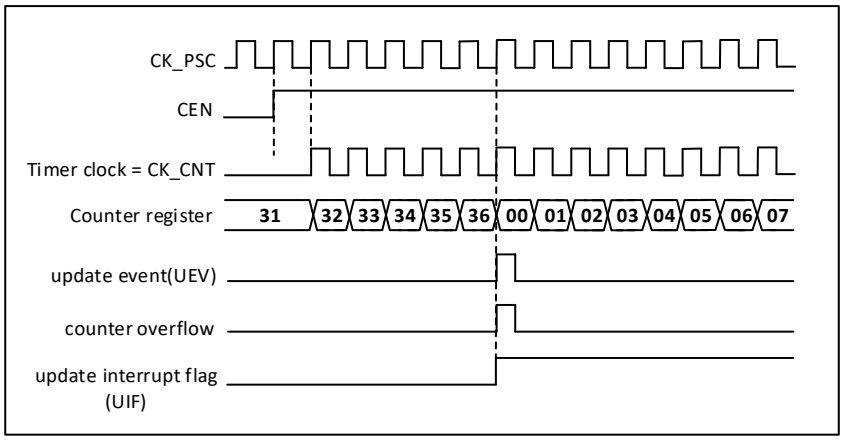


图 21-4 计数器时序图，内部时钟 1 分频

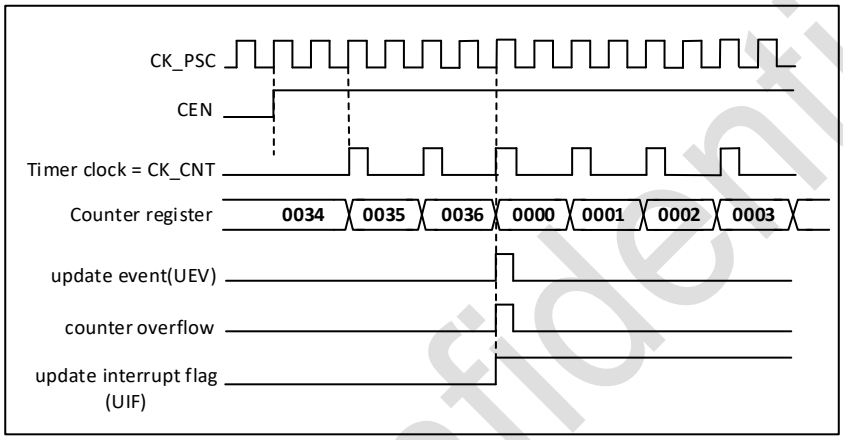


图 21-5 计数器时序图，内部时钟 2 分频

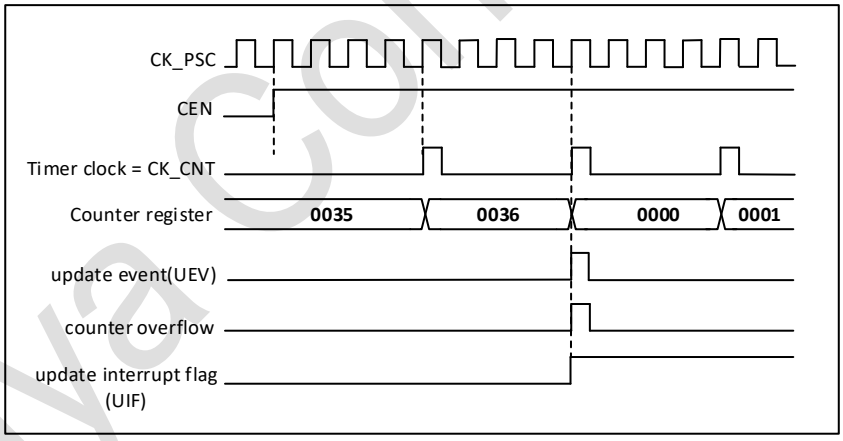


图 21-6 计数器时序图，内部时钟 4 分频

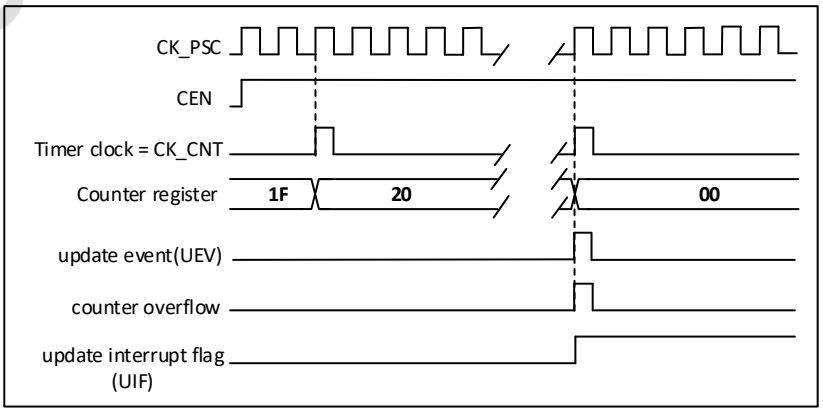


图 21-7 计数器时序图，内部时钟 N 分频

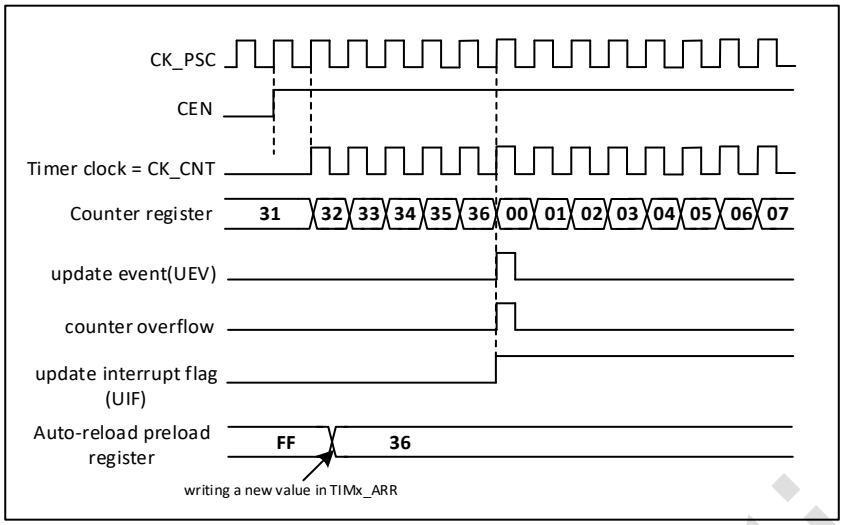


图 21-8 计数器时序图，ARPE=0 时更新事件（TIMx_ARR 未预装载）

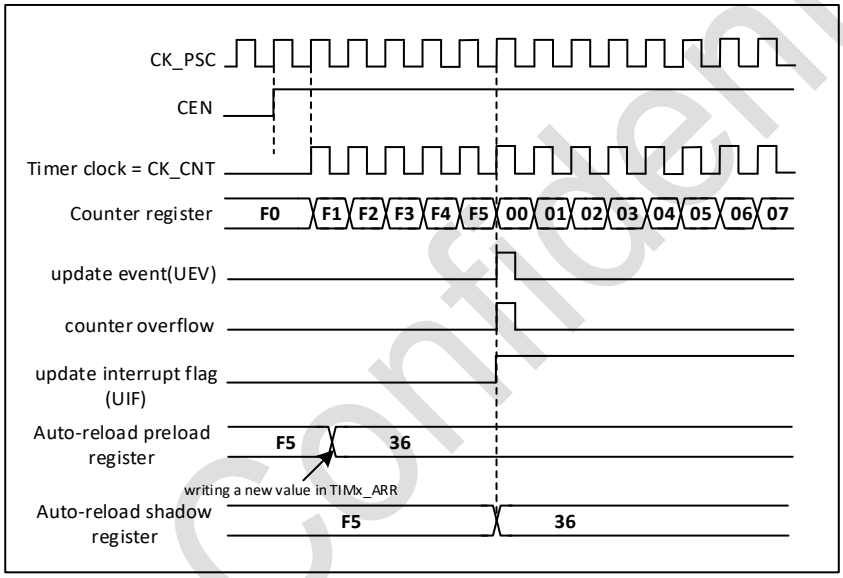


图 21-9 计数器时序图，ARPE=1 时更新事件（TIMx_ARR 预装载）

21.2.4. 重复计数器

时基单元描述了计数器上溢时更新事件（UEV）是如何产生的，它实际上仅当重复计数器计数到零才产生。这在产生 PWM 信号时很有用。

这意味着在每 N 次计数上溢时，数据从预装载寄存器传输到影子寄存器（TIMx_ARR 自动重载寄存器，TIMxPSC 预分频寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx），N 是 TIMx_RCR 重复计数器寄存器中的值。

重复计数器，在递增计数器模式的每个计数上溢时递减。

重复计数器是自动重载的，重复速率是由 TIMx_RCR 寄存器的值定义。当更新事件由软件（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到重复计数器中。

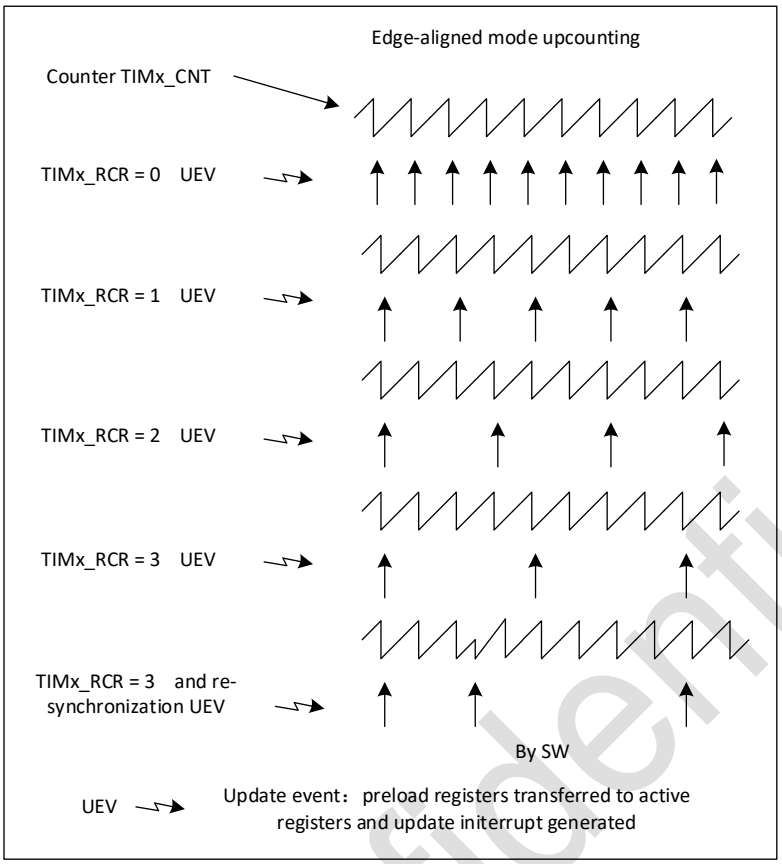


图 21-10 不同模式和 TIMx_RCR 寄存器设置下的更新频率示例

21.2.5. 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），只能通过软件改变他们。一旦置 CEN 位为 1，内部时钟即向分频器提供时钟。

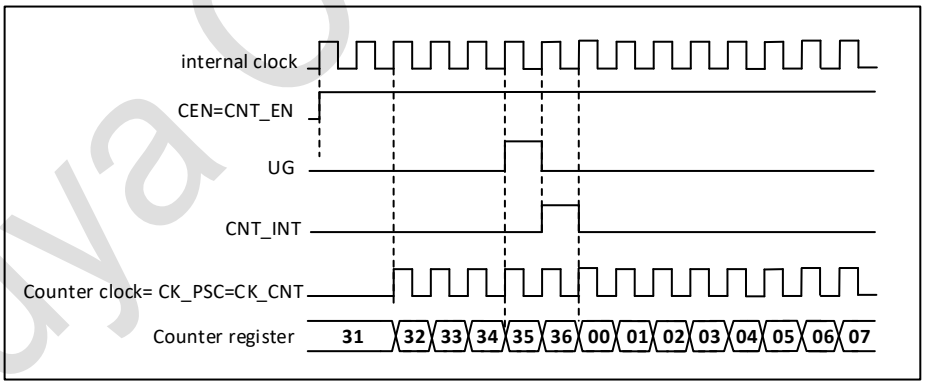


图 21-11 正常模式下的控制电路，内部时钟 1 分频

21.2.6. 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入部分（数字滤波、多路复用和预分频器）和一个输出部分（比较器和输出控制）构建而成。

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）。

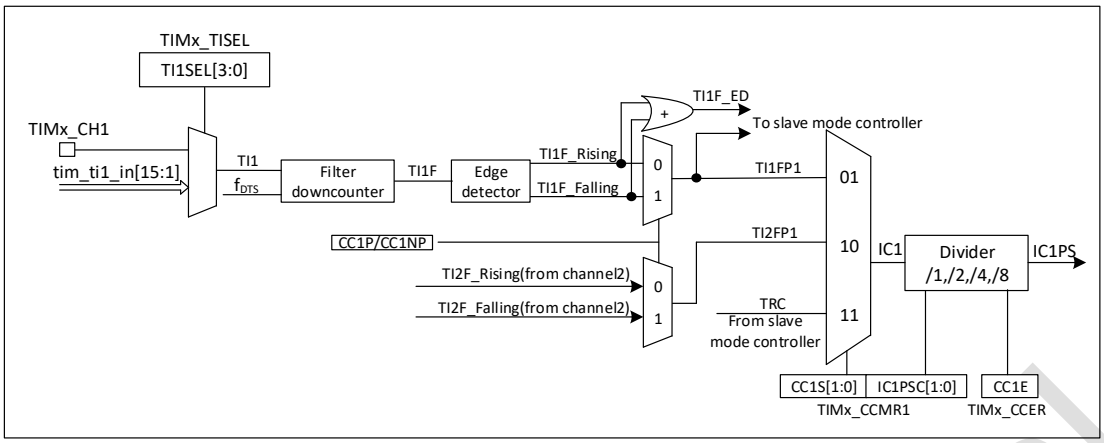


图 21-12 捕获/比较通道(示例：通道 1 输入阶段)

输出部分产生一个中间波形（高有效）作为基准，链的末端决定最终输出信号的极性。

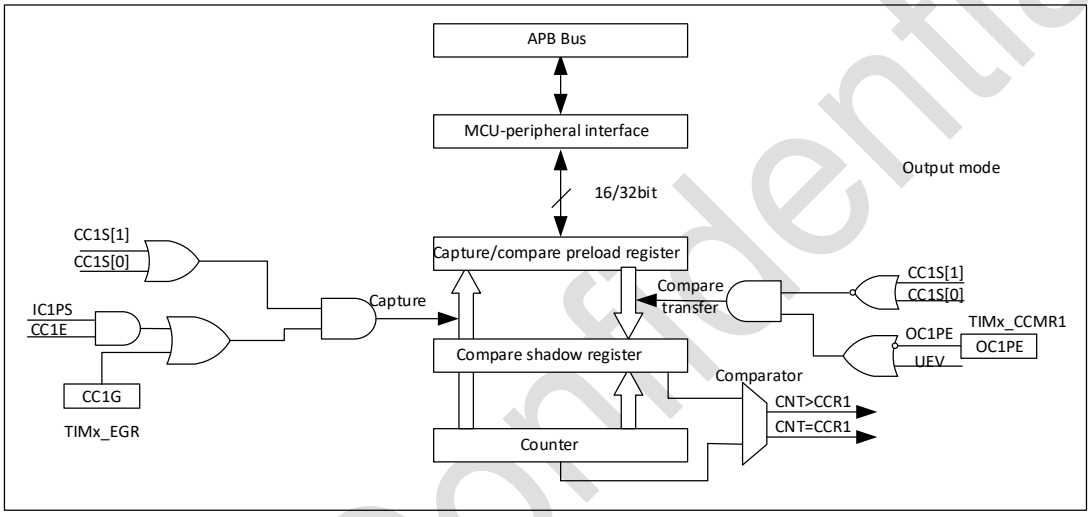


图 21-13 捕获/比较通道 1 主电路

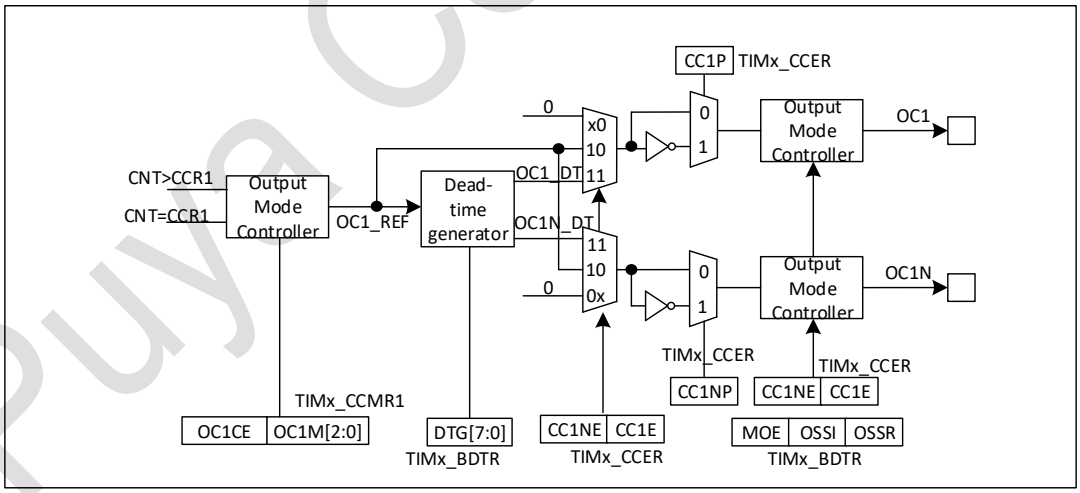


图 21-14 捕获/比较通道的输出阶段 (通道 1)

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。

21.2.7. 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，如果中断和 DMA 操作被打

开，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，则重复捕获标志 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：

- 使用 TIMx_TISEL 寄存器中的 TI1SEL[3:0]位选择正确的 TI1x 源（内部或外部）
- 选择有效输入端 TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为'00'，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 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 f_{DTS} 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。
- 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)（和 CC1NP=0）
- 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 IC1PSC=00)。
- 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：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1.
- 如设置了 CC1IE 位，则会产生一个中断请求。
- 如设置了 CC1DE 位，则会产生一个 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。

注：输入捕获中断请求能通过软件设置在 TIMx_EGR 中相应的 CCxG 位来产生。

21.2.8. 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)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。置 TIMx_CCMRx 寄存器中的 OCxM=0100，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。这将会在下面的输出比较模式一节中介绍。

21.2.9. 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：

- 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=0000)、被设置成有效电平(OCxM=0001)、被设置成无效电平(OCxM=0010)或进行翻转(OCxM=0011)。

- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
- 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。
- 若设置了相应的 DMA 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。

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

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：
   - 设置 OCxM=0011，则计数器与 CCRx 匹配时翻转 OCx 的输出引脚，
   - 置 OCxPE = 0 禁用预装载寄存器
   - 置 CCxP = 0 选择极性为高电平有效
   - 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进行更新)。下图给出了一个例子。

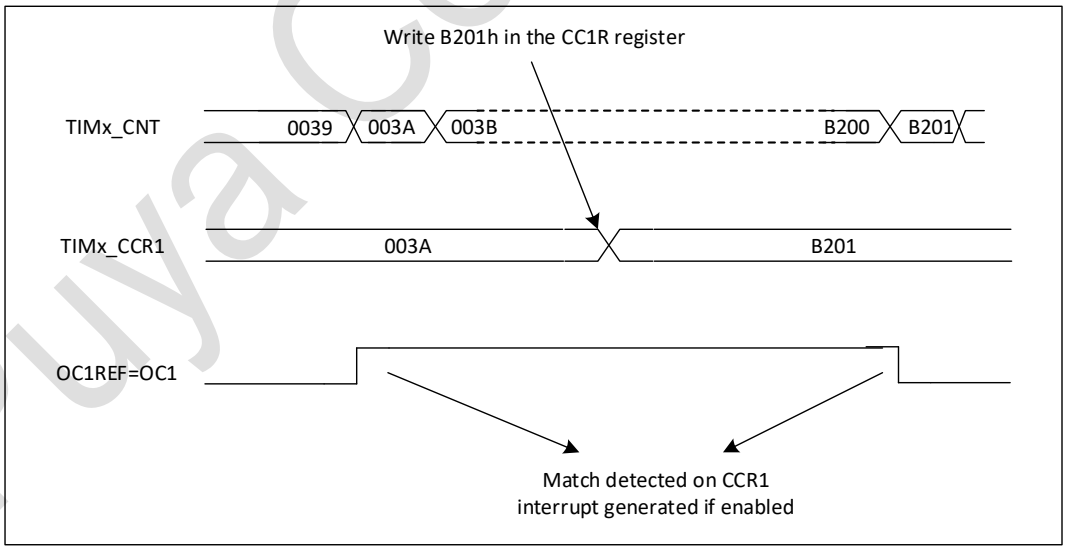


图 21-15 输出比较模式，翻转 OC1

21.2.10. PWM 模式

PWM 脉宽调制模式可以允许产生一个由 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，使能自动重装载的预装载寄存器(在递增计数或中心对称模式中)。

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者低电平有效。OCx 的输出使能通过 CCxE、CCxNE、MOE、OSSI 和 OSSR 位(TIMx_CCER 和 TIMx_BDTR 寄存器)的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式（模式 1 或者模式 2），TIMx_CNT 和 TIMx_CCRx 始终在进行比较，以确定是否符合 TIMx_CNT ≤ TIMx_CCRx。

定时器仅当计数器是递增计数时才能够产生边沿对齐模式的 PWM。

21.2.10.1. PWM 边沿对齐模式

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=8 时边沿对齐的 PWM 波形实例。

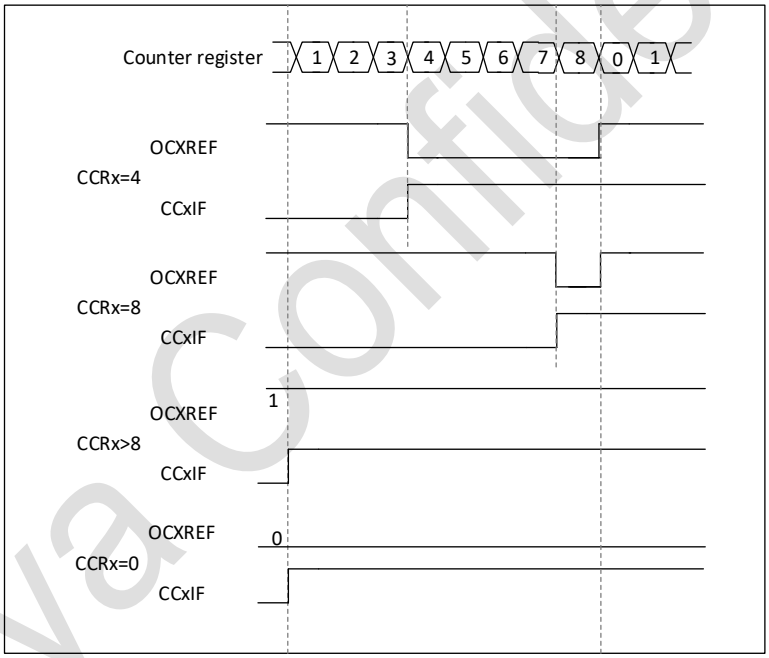


图 21-16 边沿对齐方式 PWM 输出波形（ARR=8）

21.2.11. 互补输出和死区插入

TIM16/17 能够输出一对互补信号，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（参考 TIMx_CCER 后面表格中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 8 位的死区发生器 DTG[7:0]。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)

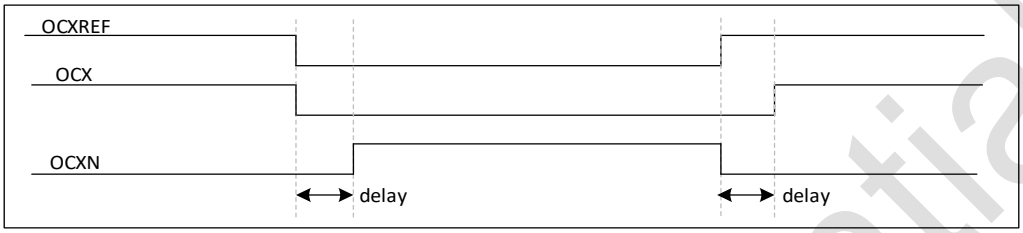


图 21-17 带死区插入的互补输出

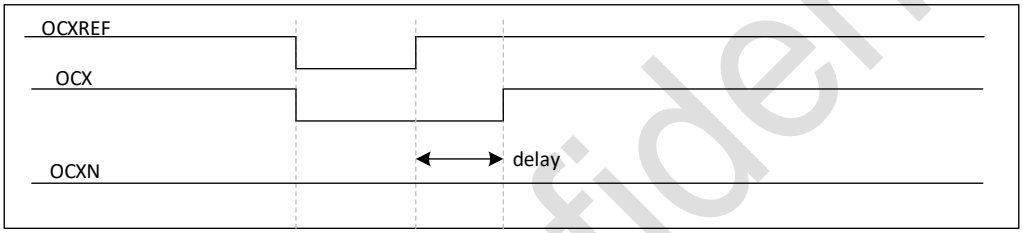


图 21-18 延迟时间大于负脉冲宽度的死区波形

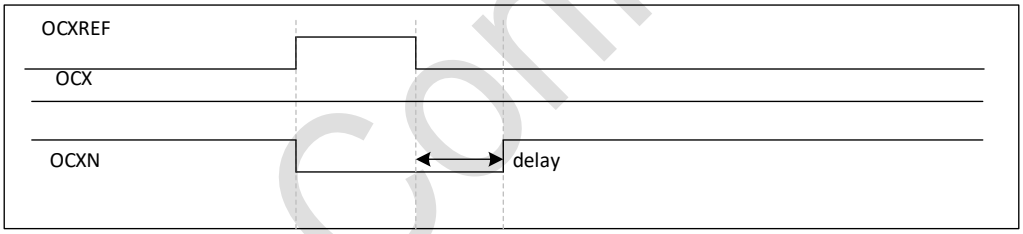


图 21-19 延迟时间大于正脉冲宽度的死区波形

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。

21.2.11.1. 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置模式、输出比较模式或 PWM 模式)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，可以将 OCxREF 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)，而同时使互补输出保持处于无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时保持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。

21.2.12. 使用刹车功能

当使用刹车功能时，依据额外的控制位（TIMx_BDTR 寄存器中的 MOE，OSSI 和 OSSR，TIMx_CR2 寄存器中的 OISx 和 OISxN），输出使能信号和无效电平信号都会被修改。无论什么情况下，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源（BRK）信号通道可以是外部输入信号（BKIN）或者如下内部信号：

- 内核 LOCKUP 输出
- PVD 输出
- CSS 检测产生的时钟失效事件信号
- 来自比较器的输出
- SRAM 字节校验错误
- FLASH ECC 错误

也可由软件通过 TIM15_EGR 寄存器中的 BG 位产生刹车事件。无论 BKE 使能位的值如何，都可以使用 BG 通过软件产生刹车。

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 个定时器的时钟周期)。
  - 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 SBIF 和 BIF 位)为’1’时，则产生一个中断。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

刹车可以由 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。

这里有两种方式产生刹车：

- 通过可编程极性的 BKR 输入，同时在 TIMx_BDTR 寄存器中使能 BKE
- 通过软件设置 TIMx_EGR 中的 BG 位。

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。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。

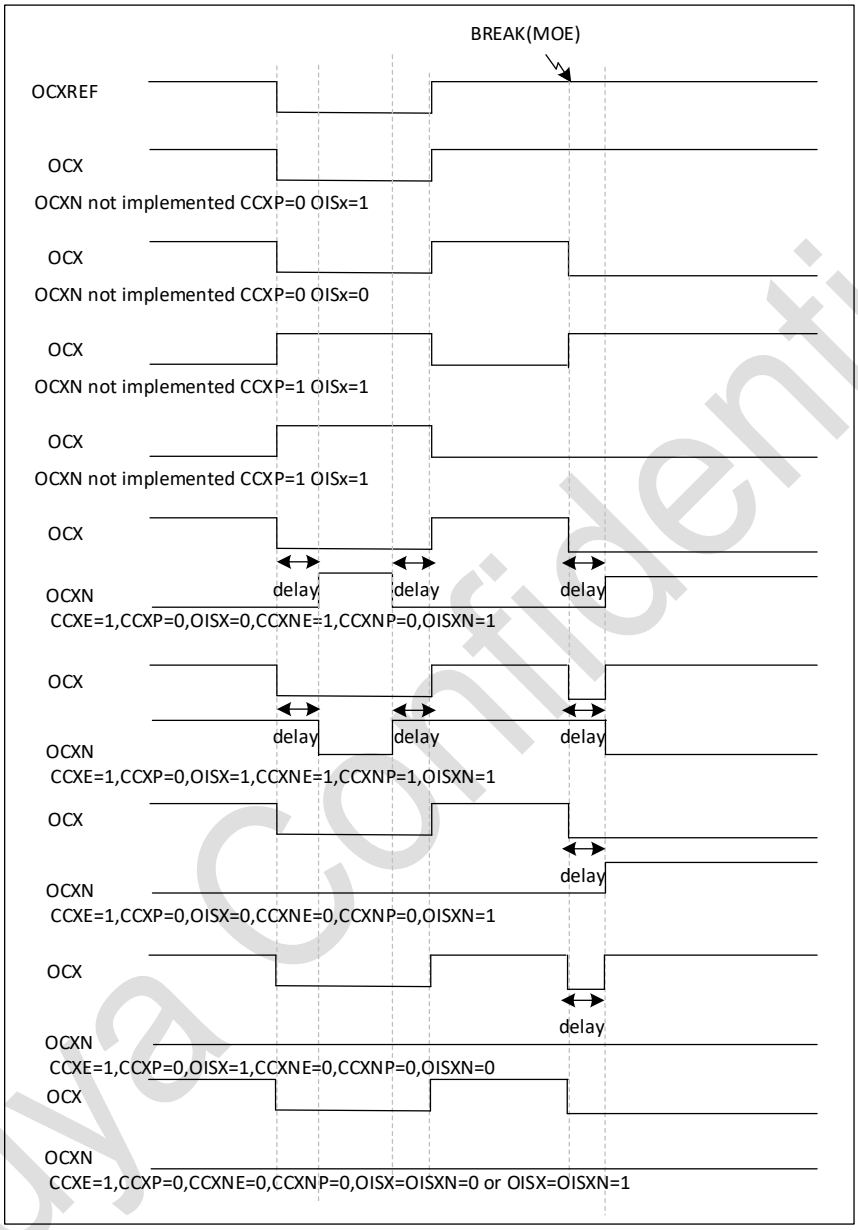


图 21-20 响应刹车事件的不同输出行为

21.3. TIM16/TIM17 中断

表 21-1 TIM6/TIM7 中断

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
COM	COMIF	COMIE
触发	TIF	TIE
刹车	BIF	BIE

21.4. TIM16/TIM17 调试模式

当芯片进入调试模式时，根据 MCUDBG 模块中 DBG_TIMx_STOP 的设置，TIM16/TIM17 计数器可以继续正常工作或者停止工作。

21.5. TIM16/TIM17 寄存器

21.5.1. TIM16/17 控制寄存器 1 (TIMx_CR1) (x=16,17)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	Res.	URS	UDIS	CEN
						RW	RW	RW					RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	2'b0	时钟分频因子。 这 2 位定义定时器时钟(CK_INT)频率与死区发生器以及数字滤波器(ETR,TIx)所用的死区及采样时钟（t_{DTS}）之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位。 0： TIM1_ARR 寄存器没有缓冲 1： TIM1_ARR 寄存器被装入缓冲器
6:3	Reserved	-	-	保留
2	URS	RW	0	更新请求源。 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： − 计数器上溢/下溢 − 设置 UG 位 − 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器上溢/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	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： − 计数器上溢/下溢 − 设置 UG 位 − 从模式控制器产生的更新 影子寄存器装入预装载值。

				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，则重新初始化计数器和预分频器。
0	CEN	RW	0	计数器使能。 0：禁止计数器 1：使能计数器 注: 在软件设置了 CEN 位后，才可以使用外部时钟、门控模式和编码器模式。触发模式可以通过硬件自动地设置 CEN 位为 1。

21.5.2. TIM16/17 控制寄存器 2 (TIMx_CR2) (x=16,17)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	OIS1N	OIS1	Res.	Res.	Res.	Res.	CCDS	Res.	Res.	CCPC
						RW	RW					RW			RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出)。 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_BDTR 寄存器)级别 1、2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出)。 0：当 MOE=0 时，死区后 OC1=0 1：当 MOE=0 时，死区后 OC1=1 注：已经设置了 LOCK(TIMx_BDTR 寄存器)级别 1、2 或 3 后，该位不能被修改。
7:4	Reserved	-	-	保留
3	CCDS	RW	0	捕获/比较的 DMA 选择 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。
2:1	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位 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，仅当发生换向事件（COM）（COMG 位置 1 或在 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。 注：该位只对具有互补输出的通道起作用。

21.5.3. TIM16/17 DMA/中断使能寄存器（TIMx_DIER) (x=16,17)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1DE	UDE	BIE	Res.	COMIE	Res.	Res.	Res.	CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	CC1DE	RW	0	捕获/比较 1 的 DMA 请求使能 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	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	刹车中断使能 0：禁止刹车中断 1：允许刹车中断
6	Reserved	-	-	保留
5	COMIE	RW	0	COM 中断使能 0：禁止 COM 中断 1：允许 COM 中断
4:2	Reserved	-	-	保留
1	CC1IE	RW	0	捕获/比较 1 中断使能 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

21.5.4. TIM16/17 状态寄存器 (TIMx_SR) (x=16,17)

偏移地址:0x010

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1OF	Res.	BIF	Res.	COMIF	Res.	Res.	Res.	CC1F	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	CC1OF	RC_W0	0	捕获/比较 1 过捕获标志 仅当相应的通道被配置为输入捕获时，该标志可由硬件置 1。写 0 可清除该位。 0：无过捕获产生； 1：CC1IF 置 1 时，TIMx_CCR1 寄存器已经被捕获到计数器的值。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标志 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	Reserved	-	-	保留
5	COMIF	RC_W0	0	COM 中断标志 一旦产生 COM 事件（当 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	Reserved	-	-	保留
1	CC1IF	RC_W0	0	捕获/比较 1 中断标志 **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。它由软件清 0。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。或者当 TIMx_CCR1 的值大于 TIMx_ARR 时，CC1IF 将在计数器递增计数上溢时置位。 **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：输入捕获产生并且计数器值已装入 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标志 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：更新事件等待响应。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 REP_CNT=0 时产生更新事件(计数器上溢)； − 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

21.5.5. TIM16/17 事件产生寄存器(TIMx_EGR) (x=16,17)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	Res.	COMG	Res.	Res.	Res.	CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件 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	Reserved	-	-	保留
5	COMG	W	0	捕获/比较事件，产生控制更新 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：该位只对有互补输出的通道有效。
4:2	Reserved	-	-	保留
1	CC1G	W	0	产生捕获/比较 1 事件 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**： CC1IF 置位，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**： 当前的计数器值捕获至 TIMx_CCR1 寄存器，CC1IF 置位，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	W	0	产生更新事件 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)。

21.5.6. TIM16/17 捕获/比较模式寄存器 1(TIMx_CCMR1) (x=16,17)

偏移地址:0x18

复位值:0x0000 0000

同一寄存器可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其他位在输入捕获和输出比较模式下的功能均不同。对于任一给定位，OCxx 用于说明通道配置为输出时该位对应的功能，ICxx 则用于说明通道配置为输入时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[2:0]			OC1PE	Res.	CC1S[1:0]	
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1PSC[1:0]			
								RW	RW	RW	RW	RW	RW	RW	

21.5.6.1. 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:4	IC1F[3:0]	RW	4’b0	输入捕获 1 滤波器 该位域定义了 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8

				1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC[1:0]	RW	2’b0	输入/捕获 1 预分频器 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[1:0]	RW	2’b0	CC1S[1:0]：捕获/比较 1 选择。 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：保留 11：保留 注： CC1S 仅在通道关闭时(TIM16/17_CCER 寄存器的 CC1E=0)才是可写的。

21.5.6.2. 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6:4	OC1M[2:0]	RW	00	输出比较 1 模式 该位定义了 OC1、OC1N 的输出参考信号 OC1REF 的行为。OC1REF 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：设置通道 1 为匹配时输出有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)匹配时，强制 OC1REF 为高电平。 010 ：设置通道 1 为匹配时输出无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)匹配时，强制 OC1REF 为低电平。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，OC1REF 发生翻转。 100：强制为无效电平。强制 OC1REF 为低电平。 101：强制为有效电平。强制 OC1REF 为高电平。 110 ： PWM 模式 1--- 在递增计数时，一旦 TIMx_CNT<TIMx_CCR1，通道 1 为有效状态，否则为无效状态; 111 ： PWM 模式 2--- 在递增计数时，一旦 TIMx_CNT<TIMx_CCR1，通道 1 为无效状态，否则为有效状态;

				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	0	输出比较 1 预装载使能 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIM1_CCR1 寄存器，且新值马上起作用。 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，TIM1_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 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。
2	Reserved	-	-	保留
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 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：保留； 11：保留； 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

21.5.7. TIM16/17 捕获/比较使能寄存器 (TIMx_CCER) (x=16,17)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 该位和 CC1P 联合使用定义 TI1FP1/TI2FP1 极性，参考 CC1P 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	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，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，CC1E 位的值。
1	CC1P	RW	0	输入/捕获 1 输出极性 **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]位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性： 00：非反相/上升沿。TIxFP1 上升沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）； TIxFP1 非反相（门控模式、编码器模式）。 01：反相/下降沿。TIxFP1 下降沿有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：非反相/双沿。TIxFP1 上升和下降沿都有效（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作）； TIxFP1 非反相（门控模式）。这个配置不能应用于编码器模式下。 注 1：一旦 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。 注 2：对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，那么 CC1P 的实际有效位只有在 COM 事件发生时才会加载预载值。
0	CC1E	RW	0	输入/捕获 1 输出使能 **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、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、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 TIMx_CCR1 寄存器。 0：捕获禁止 1：捕获使能 注：对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CC1PC 位被设置，那么 CC1E 的实际有效位只有在 COM 事件发生时才会加载预载值。

表 21-2 具有刹车功能的互补通道 OCx 和 OCxN 的输出控制位

控制位					输出状态	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
1	x	x	0	0	输出禁止(不由定时器驱动)，OCx=0, OCx_EN=0	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
		0	0	1	禁止输出 (不由定时器驱动)，OCx=0，OCx_EN=0	OCxREF+极性 OCxN=OCxREF 异或 CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性 OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
		x	1	1	OCREF+极性+死区，OCx_EN=1	OCREF 的互补（not OCREF）+ 极性 +死区，OCxN_EN=1

		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 异或 CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性 OCx=OCxREF 异或 CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平）, OCxN=CCxNP, OCxN_EN=1
0	0	x	x	x	输出禁止(不由定时器驱动)	
	1		0	0		
	1		0	1	关闭状态（输出使能且为无效电平） 异步：OCx=CCxP, OCx_EN=1，OCxN=CCxNP, OCxN_EN=1（如果触发 BRK）。 随后（仅当触发 BRK 时才有效），若时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，则 OCx=OISx 且 OCxN=OISxN。	
	1		1	0		
	1		1	1		

21.5.8. TIM16/17 计数器寄存器(TIMx_CNT) (x=16,17)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	0	计数器的值

21.5.9. TIM16/17 预分频寄存器 (TIMx_PSC) (x=16,17)

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	16'b0	预分频器值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为"复位模式"时通过触发控制器清零时）要装载到有效预分频器寄存器的值。

21.5.10. TIM16/17 自动装载寄存器 (TIMx_ARR) (x=16,17)

偏移地址:0x2C

复位值:0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	0xFFFF	自动重装载的值 ARR 包含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。

21.5.11. TIM16/17 重复计数器 寄存器(TIMx_RCR) (x=16,17)

偏移地址:0x30

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	REP[7:0]	RW	8’b0	重复计数器的值 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）；使能更新中断时，则同时会影响产生更新中断的速率。 每次递减计数器 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 只有在重复更新事件（UIF）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重复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;

21.5.12. TIM16/17 捕获/比较寄存器 1(TIMx_CCR1) (x=16,17)

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1[15:0]	RW	0	捕获/比较 1 的值 **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)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIMx_CNT 比较的值，并且在 OC1 端口上输出信号。 **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。此时，只能读取 TIMx_CCR1 寄存器，无法对其进行编程。

21.5.13. TIM16/17 刹车和死区寄存器寄存器(TIMx_BDTR) (x=16,17)

偏移地址:0x44

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。此位由软件置 1，也可根据 AOE 位状态自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态，具体取决于 OSSI 位; 1：如果设置了相应的使能位（TIMx_CCER 寄存器的 CCxE、CCxNE 位），则使能 OC 和 OCN 输出。
14	AOE	RW	0	自动输出使能 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	0	刹车输入极性 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。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入； 1：开启刹车输入。 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。
11	OSSR	RW	0	运行模式下“关闭状态”选择 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：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，开启 OC/OCN 输出并输出无效电平。 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	0	空闲模式下“关闭状态”选择 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：处于无效状态时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其无效电平。 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[1:0]	RW	2’b0	锁定设置 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_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，TIMx_CCER 寄存器的 CCxP/CCxNP 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，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[7:0]	RW	8'b0	死区发生器设置。 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] × T_{dtg}， T_{dtg} = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG[5:0]) ×T_{dtg}，T_{dtg} = 2 ×T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 8 ×T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 16 ×T_{DTS}； 例：若 T_{DTS} = 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(TIM15_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则这些位不能被修改。

21.5.14. TIM16/17 输入选择寄存器 (TIMx_TISEL) (x=16,17)

偏移地址:0x5C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1SEL[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	TI1SEL	RW	0	TI1 输入选择. 0000：TIMx_CH1 0001：保留 0010：MCO 0011：HSE_DIV32 0100：RTC 闹钟唤醒 0101：LSE_CSS 0110：LSI 其他：保留

21.5.15. TIM16/17 备用选项寄存器 1 (TIMx_AF1) (x=16,17)

偏移地址:0x60

复位值:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
-	-	-	-	RW	RW	RW	-	-	-	-	-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	BKCMP2P	RW	0	来源于比较器 2 的刹车输入极性控制。 该位选择来源于比较器 2 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 2 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 2 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	来源于比较器 1 的刹车输入极性控制。 该位选择来源于比较器 1 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 1 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 1 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	比较器 2 刹车输入使能。 比较器 2 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 2 刹车输入关闭 1：比较器 2 刹车输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	比较器 1 刹车输入使能。

				比较器 1 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 1 刹车输入关闭 1：比较器 1 刹车输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能。 比较器 1 刹车 BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

21.5.16. TIM16/17 DMA 控制寄存器（TIMx_DCR) (x=16,17)

偏移地址:0x3DC

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	DBL[4:0]					Res.	Res.	Res.	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL[4:0]	RW	0 0000	DMA 连续传送长度 这些位定义了 DMA 的传送长度（当对 TIMx_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送），即：定义被传送的字节数目： 00000： 1 次传输 00001： 2 次传输 00010： 3 次传输 ..... 10001： 18 次传输
7:5	Reserved	-	-	保留
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIMx_DMAR 寄存器的地址进行读或写时），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 ..... 00011：TIMx_DIER .....

21.5.17. TIM16/17 DMA 连续传输地址寄存器(TIMx_DMAR) (x=16,17)

偏移地址:0x3E0

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAB[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DMAB[31:0]	RW	0	DMA 连续传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作: TIMx_CR1 地址 + (DBA + DMA 指针)x4，其中: “TIMx_CR1 地址”是控制寄存器 1 的地址; “DBA”是 TIMx_DCR 寄存器中定义的基地址; “DMA 指针”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。

22. 基本定时器(TIM6/TIM7)

22.1. TIM6/TIM7 简介

基本定时器 TIM6 包含一个 32 位自动重载计数器， TIM7 包含一个 16 位自动重载计数器。该计数器由可编程预分频器驱动。

此类定时器不仅可用作通用定时器以生成时基，还可以专门用于驱动数模转换器 (DAC)。实际上，此类定时器内部连接到 DAC 并能够通过其触发输出驱动 DAC。

这些定时器彼此完全独立，不共享任何资源。

22.2. TIM6/TIM7 主要特性

TIM6 和 TIM7 定时器的主要功能包括：

- 32/16 位自动装载计数器(TIM6 配置为 32 位计数)
- 16 位可编程(可以实时修改)的预分频器，计数器时钟频率的分频系数为 1～65535 之间的任意数值
- 触发 DAC 的同步电路
- 在更新事件（计数器溢出）发生时产生中断/DMA

22.3. TIM6/TIM7 功能描述

22.3.1. 功能框图

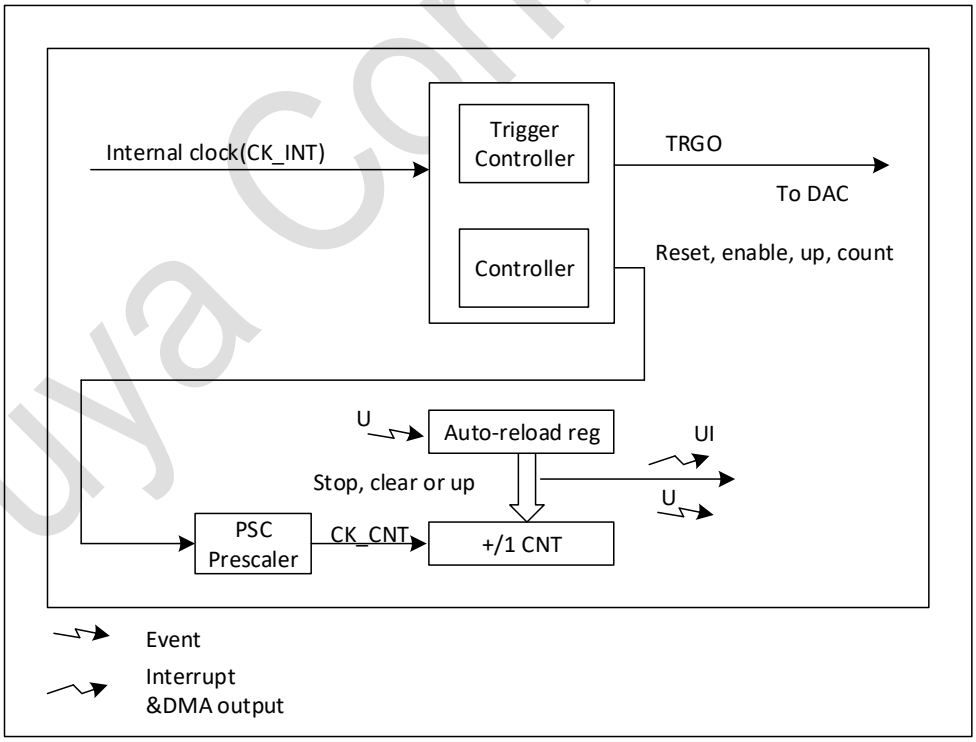


图 22-1 TIM6/TIM7 通用框图

22.3.2. 时基单元

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存

器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位(CEN)置 1 时，才会启动计数器。

注意，计数器将在 TIMx_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。

22.3.2.1. 预分频器描述

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。新的预分频参数将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。

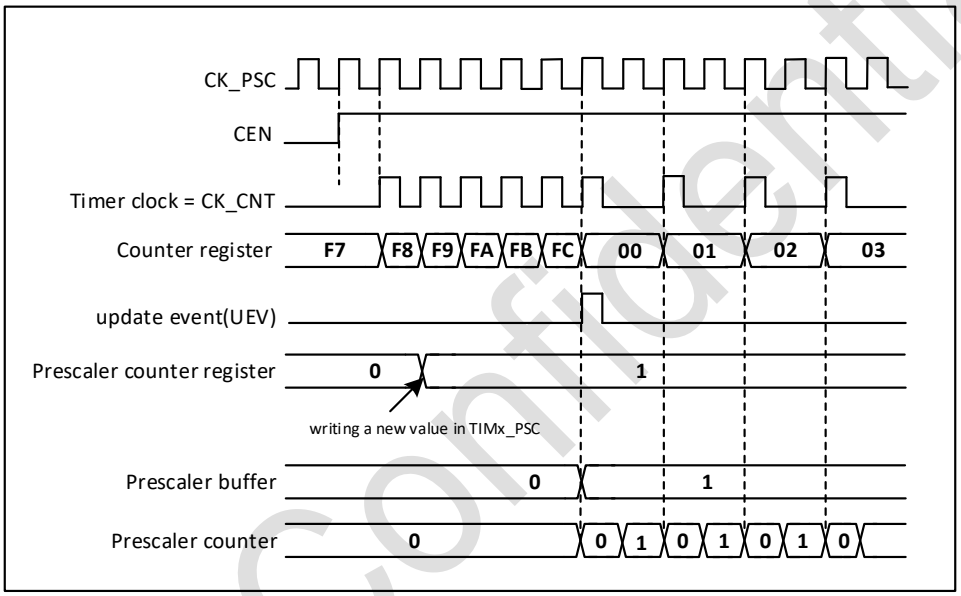


图 22-2 预分频器分频由 1 变为 2 时的计数器时序图

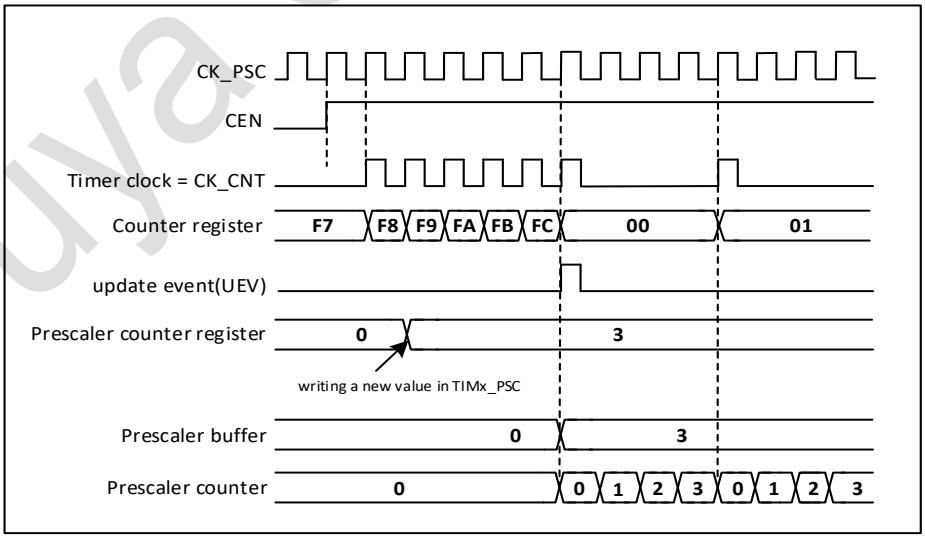


图 22-3 预分频器分频由 1 变为 4 时的计数器时序图

22.3.3. 计数模式（递增）

计数器从 0 计数到自动加载值(TIMx_ARR 的内容)，然后重新从 0 开始计数并且产生一个计数上溢事件。

每次计数上溢都会产生更新事件。而在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，如下寄存器都被更新，硬件同时设置更新标志位(TIMx_SR 寄存器中的 UIF 位)置位 (依据 URS 位)：

- 自动装载影子寄存器重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

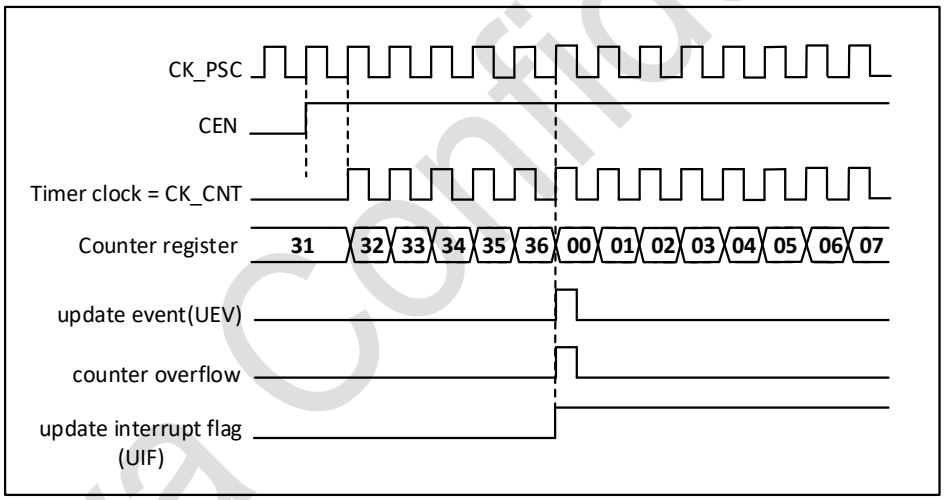


图 22-4 计数器时序图，内部时钟 1 分频

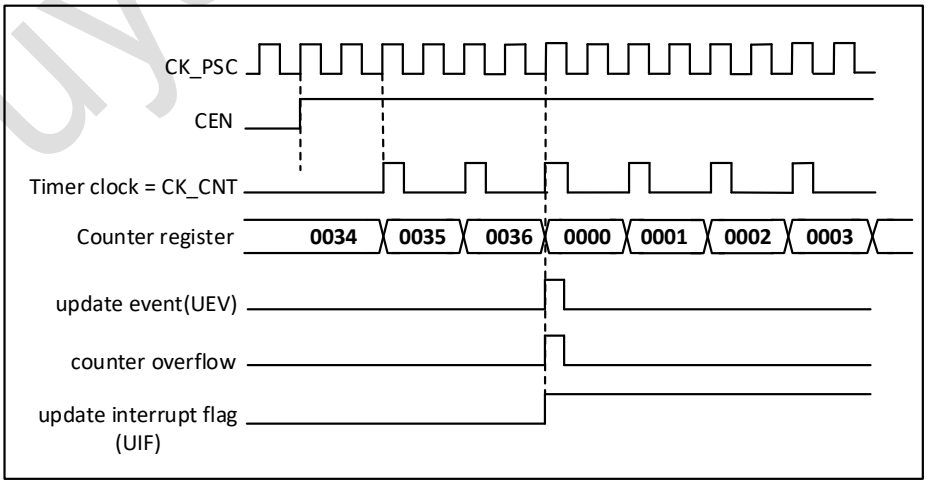


图 22-5 计数器时序图，内部时钟 2 分频

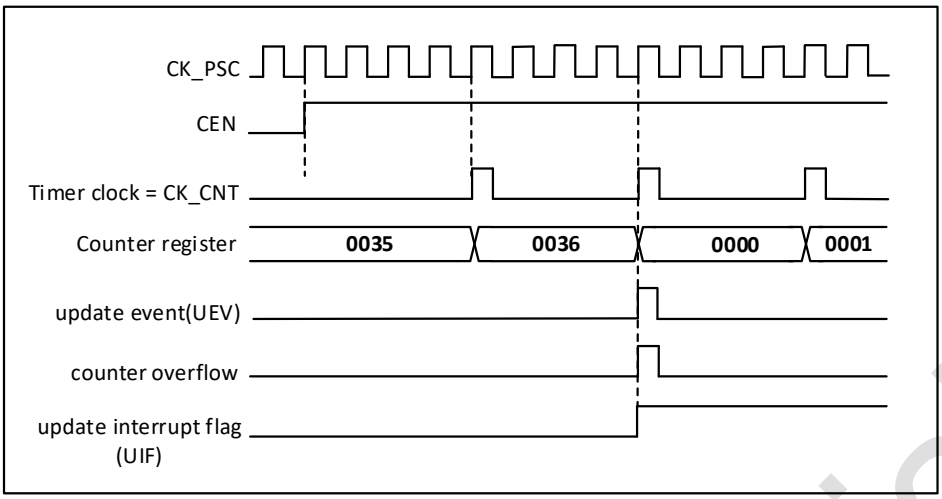


图 22-6 计数器时序图，内部时钟 4 分频

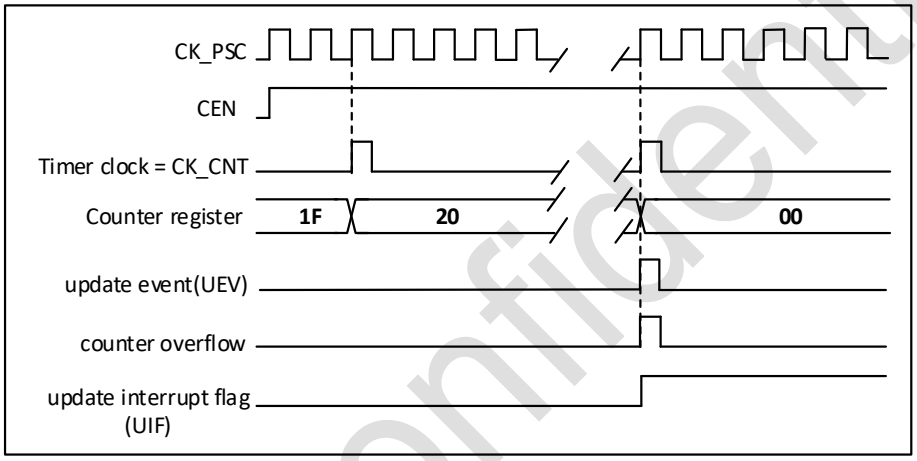


图 22-7 计数器时序图，内部时钟 N 分频

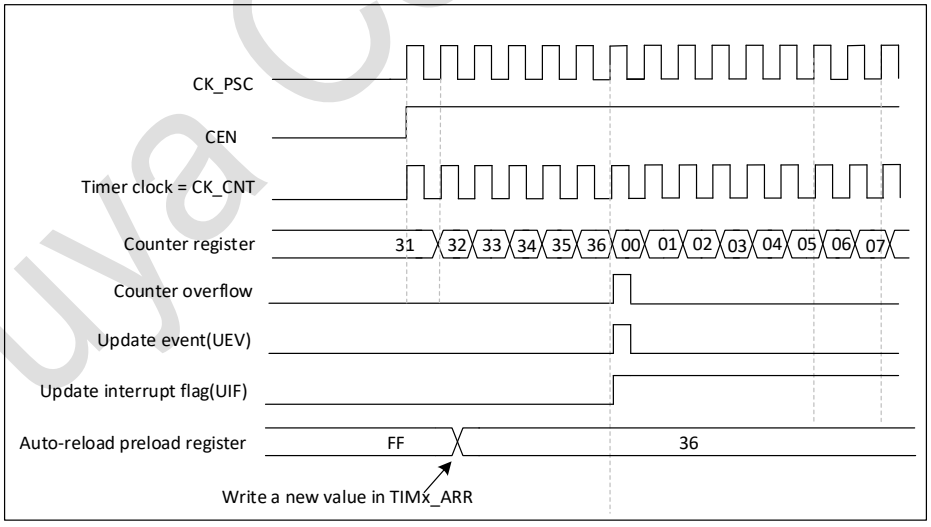


图 22-8 计数器时序图，ARPE=0 时的更新事件(TIMx_ARR 未预装载)

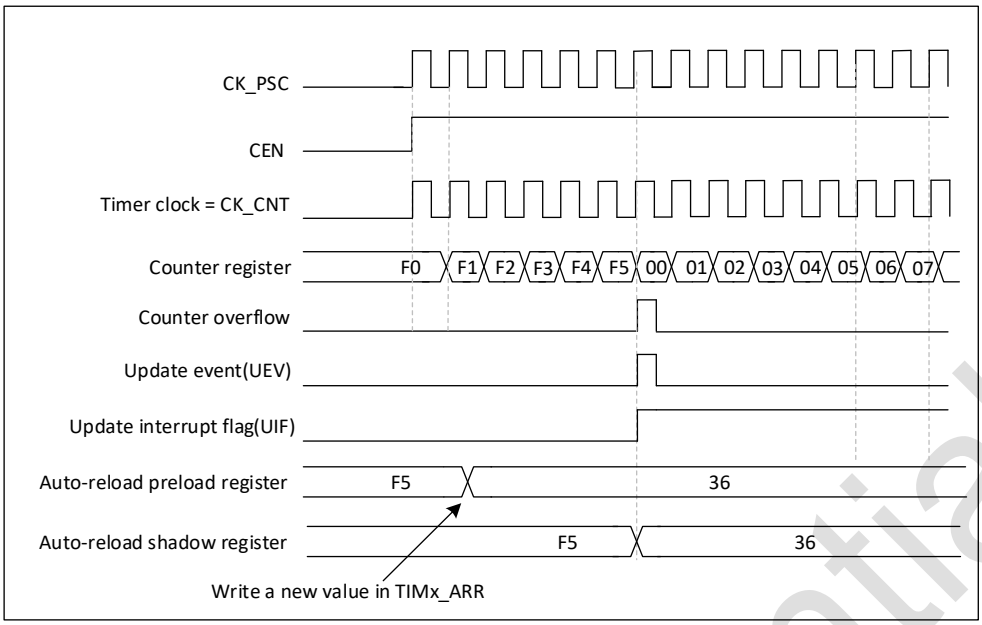


图 22-9 计数器时序图，ARPE=1 时的更新事件(TIMx_ARR 预装载)

22.4. TIM6/TIM7 调试模式

当微控制器进入调试模式时(Cortex®-M0+核停止)，根据 DBGMCU 模块中 DBG_TIMx_STOP 的设置，TIM6、TIM7 计数器可以或者继续正常操作，或者停止。

22.5. TIM6/TIM7 寄存器

22.5.1. TIM6/TIM7 控制寄存器 1（TIMx_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
								RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	ARPE	RW	0	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	Reserved	-	-	保留
3	OPM	RW	0	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	0	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： – 计数器上溢/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，则只有计数器上溢/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	更新禁止 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 影子寄存器装入预装载值。 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，则重新初始化计数器和预分频器。
0	CEN	RW	0	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，门控模式才能工作。触发模式可以自动地通过硬件设置 CEN 位为 1。 在单脉冲模式下，当产生更新事件时 CEN 被自动清除。

22.5.2. TIM6/TIM7 控制寄存器 2（TIMx_CR2）

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMS[2:0]			Res.	Res.	Res.	Res.
									RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6:4	MMS[2:0]	RW	3’b0	主模式选择 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)。该触发输出可用于在同一时间启动多个定时器或在一段时间内控制从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑或运算组合而成。 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。
3:0	Reserved	-	-	保留

22.5.3. TIM6/TIM7 DMA/中断使能寄存器（TIMx_DIER）

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
							RW								RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	UDE	RW	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7:1	Reserved	-	-	保留
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

22.5.4. TIM6/TIM7 状态寄存器（TIMx_SR）

偏移地址:0x10

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	UIF	RC_W0	0	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：更新事件发生。当寄存器被更新时该位由硬件置 1： − 若 TIMx_CR1 寄存器的 UDIS=0，当计数器上溢时产生更新事件； − 若 TIMx_CR1 寄存器的 UDIS=0、URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

22.5.5. TIM6/TIM7 事件产生寄存器（TIMx_EGR）

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
															W

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	UG	W	0	产生更新事件 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)。

22.5.6. TIM6/TIM7 计数寄存器（TIMx_CNT）

偏移地址:0x24

复位值:0x0000 0000

TIM6：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	RW	32’b0	计数器值

TIM7：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	16’b0	计数器值

22.5.7. TIM6/TIM7 预分频寄存器（TIMx_PSC）

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	16'b0	预分频器的值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时要装载到有效预分频器寄存器的值（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为"复位模式"时通过触发控制器清零时）。

22.5.8. TIM6/TIM7 自动重载寄存器（TIMx_ARR）

偏移地址:0x2C

复位值:0xFFFF FFFF

TIM6:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'hFFFF FFFF	自动重装载的值。 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。

TIM7:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'hFFFF	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。

23. 脉冲宽度调制(PWM)

23.1. PWM 主要特性

PWM 的主要功能包括:

- 16bit 递增、递减或者递增/递减的自动重装载计数器
- 可编程周期和占空比
- 支持通过外部输入时钟为系统时钟的倍频的情况下计数
- 可编程分频器，允许对计数器的时钟频率进行 1 到 65536 的分频
- 支持重要寄存器写保护
- 多达 4 个独立的通道
- 支持 2 个通道死区时间可编程的互补输出
- 输出极性可配置
- 支持边沿对齐和中心对齐
- 刹车输入可以将定时器的输出信号置为用户可选的安全配置
- 支持 1 路刹车输入
- 支持外部时钟计数
- 支持 DMA
- 中断事件
  - 更新：计数器向上、向下溢出
  - 输出比较
  - 刹车输入

23.2. PWM 功能描述

23.2.1. 功能框图

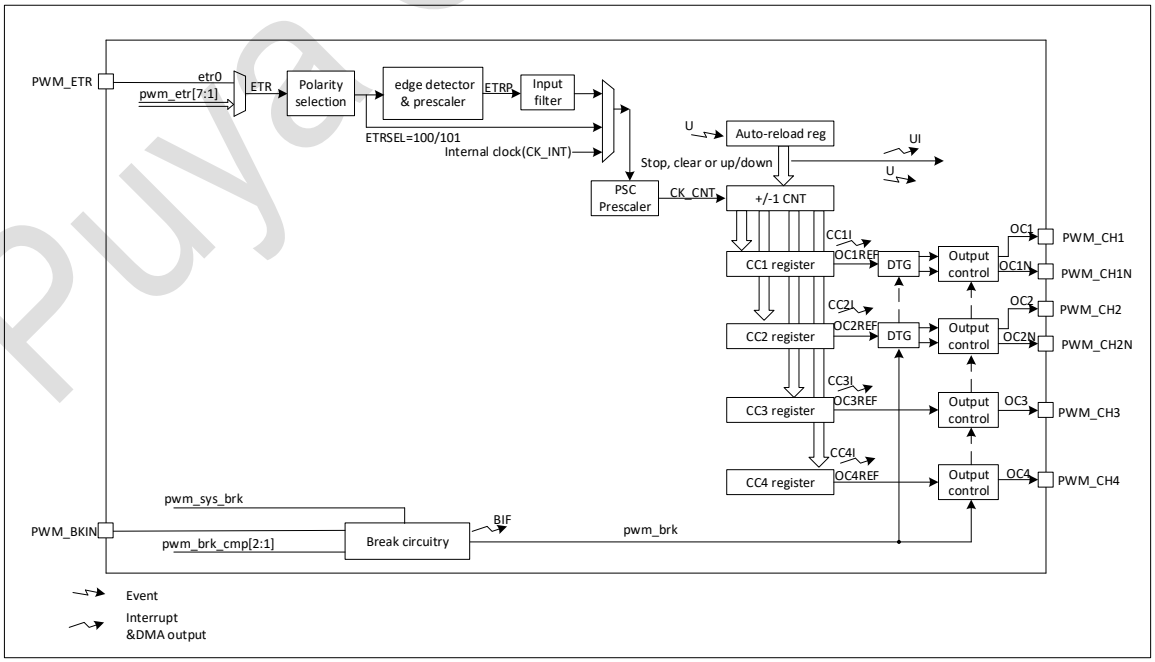


图 23-1 PWM 框图

23.2.2. 计数模式

23.2.2.1. 递增计数模式

在递增计数模式中，计数器从 0 计数到自动加载值(PWM_ARR 的内容)，然后重新从 0 开始计数并且产生一个计数上溢事件。

而在 PWM_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过设置 PWM_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 PWM_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

当发生一个更新事件时,所有以下的寄存器都被更新,硬件同时(依据 URS 位)设置更新标志位(PWM_SR 寄存器中的 UIF 位)：

- 自动装载影子寄存器被重新置入预装载寄存器的值(PWM_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(PWM_PSC 寄存器的内容)。

下图给出一些例子，当 PWM_ARR=0x36 时计数器在不同时钟频率下的动作。

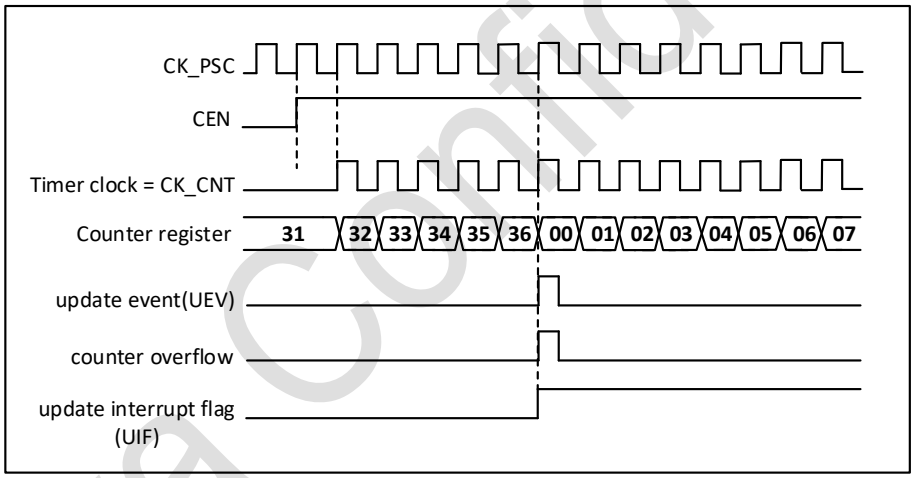


图 23-2 计数器时序图，内部时钟分频因子为 1

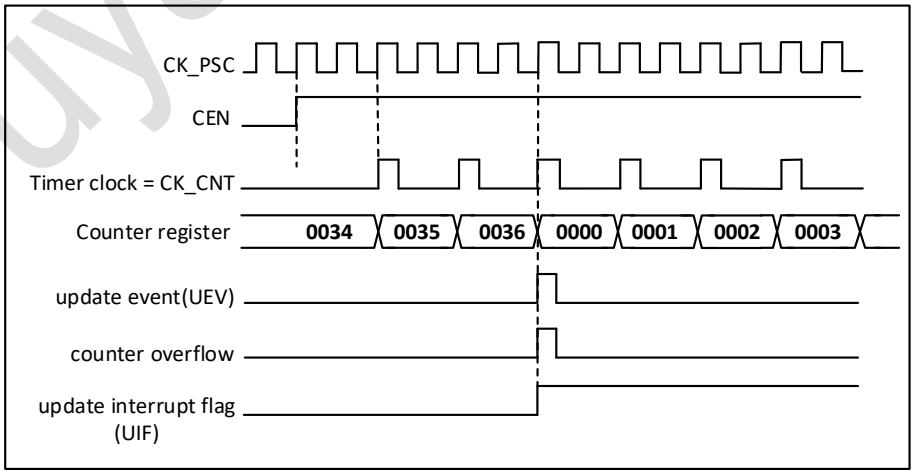


图 23-3 计数器时序图，内部时钟分频因子为 2

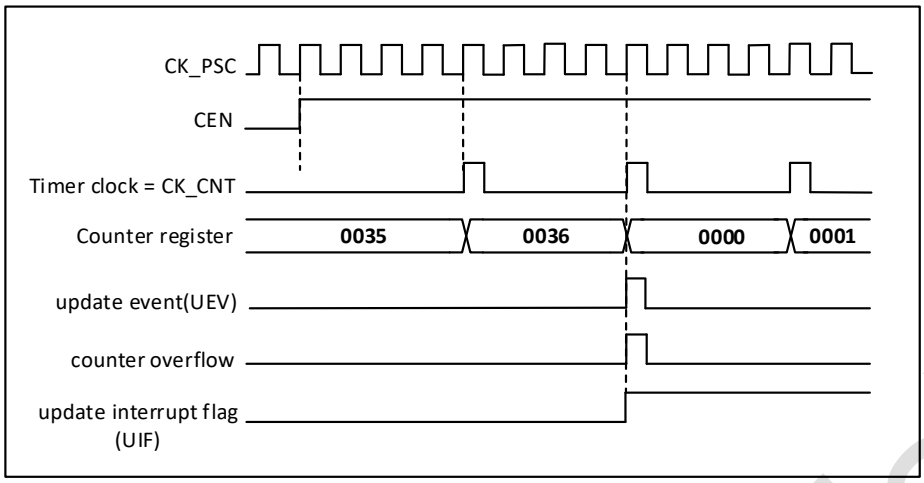


图 23-4 计数器时序图，内部时钟分频因子为 4

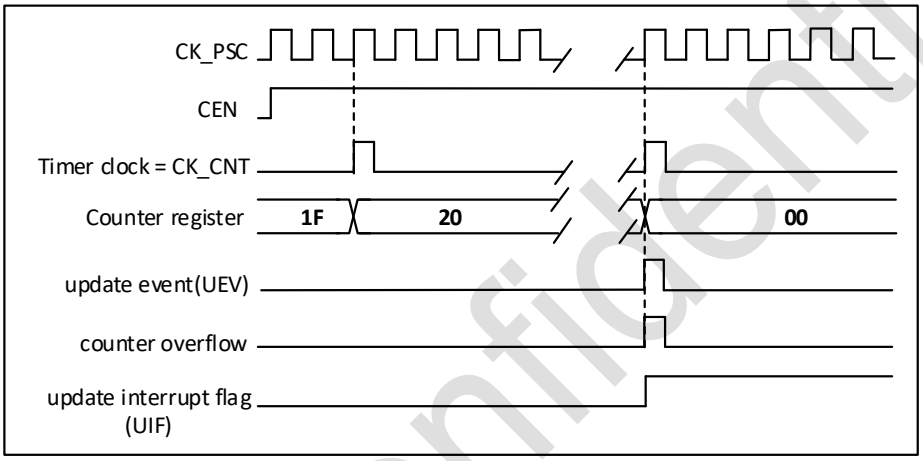


图 23-5 计数器时序图，内部时钟分频因子为 N

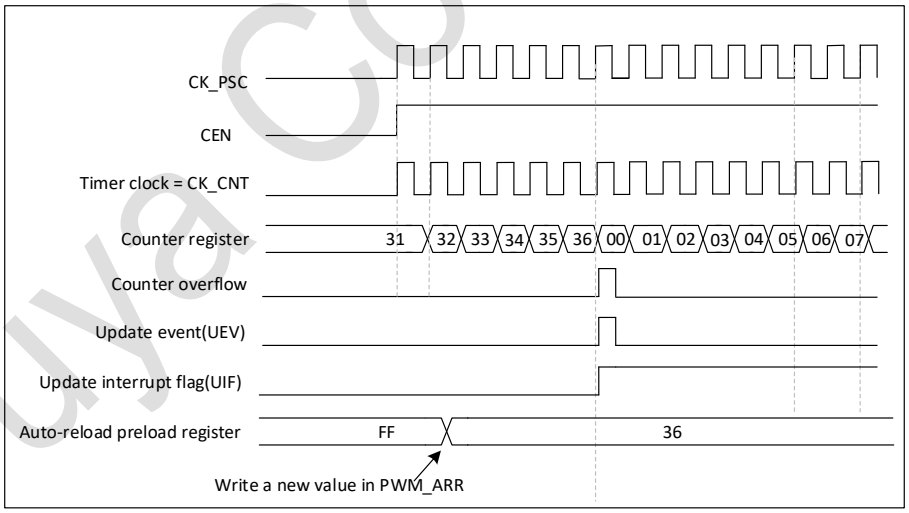


图 23-6 计数器时序图，当 ARPE=0 时的更新事件(未预装 PWM_ARR)

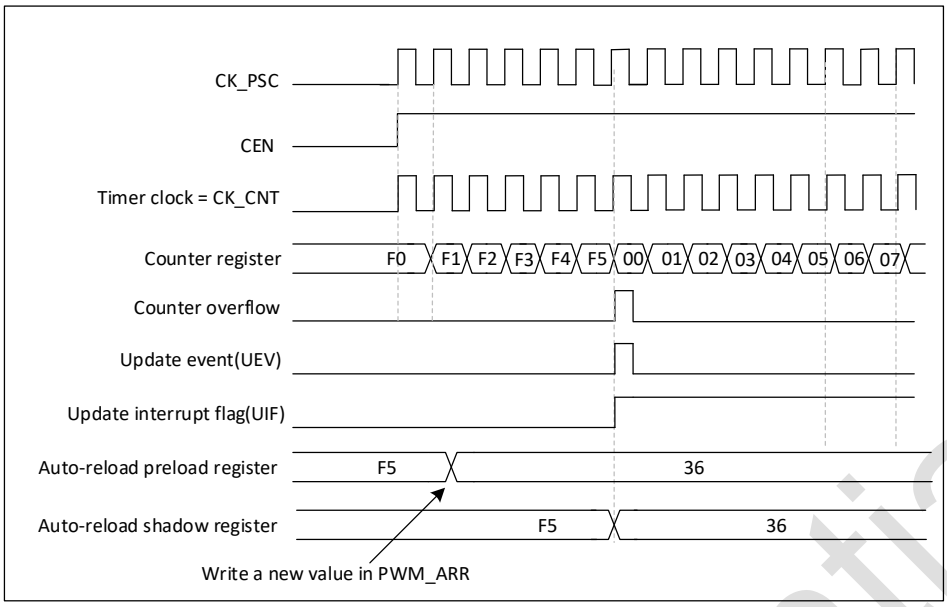


图 23-7 计数器时序图，当 ARPE=1 时的更新事件(预装 PWM_ARR)

23.2.2.2. 递减计数模式

在递减计数模式中，计数器从自动加载值(PWM_ARR 的内容)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下溢事件。每次计数下溢都会产生更新事件。

当发生一个更新事件时，所有以下的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

以下是一些当 PWM_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

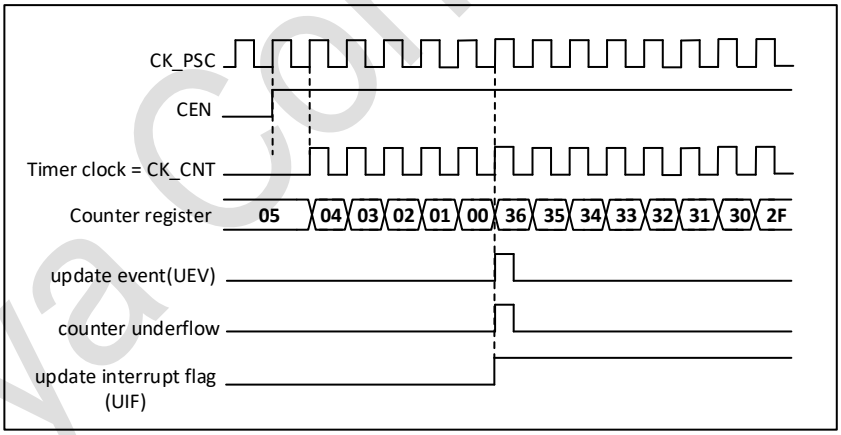


图 23-8 计数器时序图，内部时钟分频因子为 1

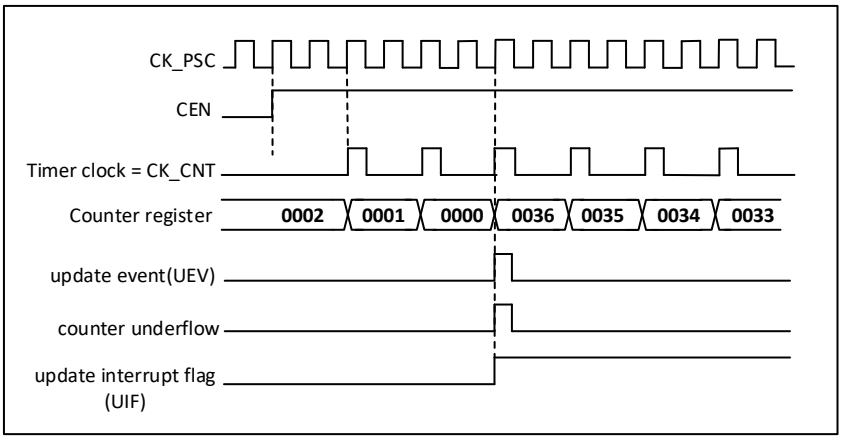


图 23-9 计数器时序图，内部时钟分频因子为 2

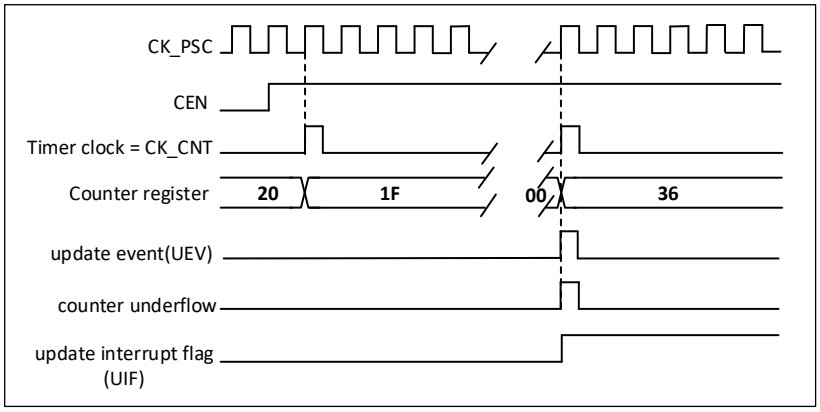


图 23-10 计数器时序图，内部时钟 N 分频

23.2.2.3. 中心对齐模式

在中心对齐模式，计数器从 0 开始计数到自动加载的值(PWM_ARR 寄存器)−1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

通过配置 PWM_CR1 寄存器中的 CMS 位不为 ‘00’ 可以得到中心对齐模式。

在此模式下，不能写入 PWM_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件。然后，计数器重新从 0 开始计数，预分频器内部计数器也重新从 0 开始计数。

当发生更新事件时，所有的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

以下是一些计数器在不同时钟频率下的操作的例子：

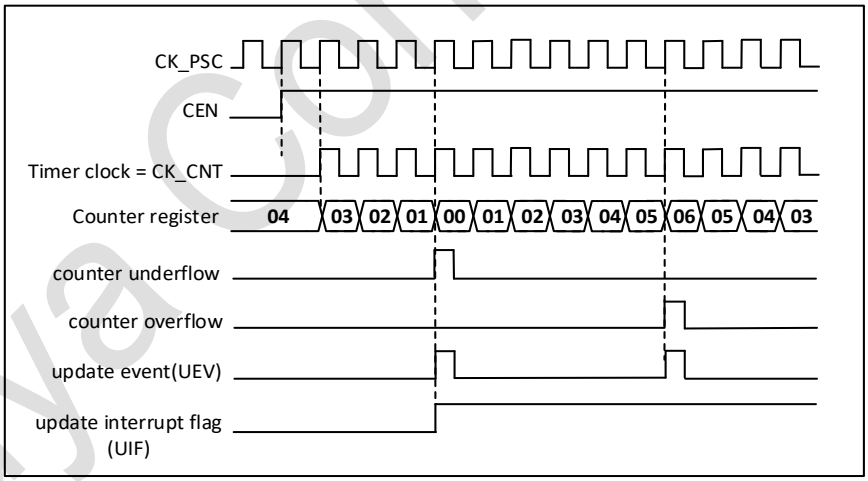


图 23-11 计数器时序图，内部时钟分频因子为 1， PWM_ARR=0x6

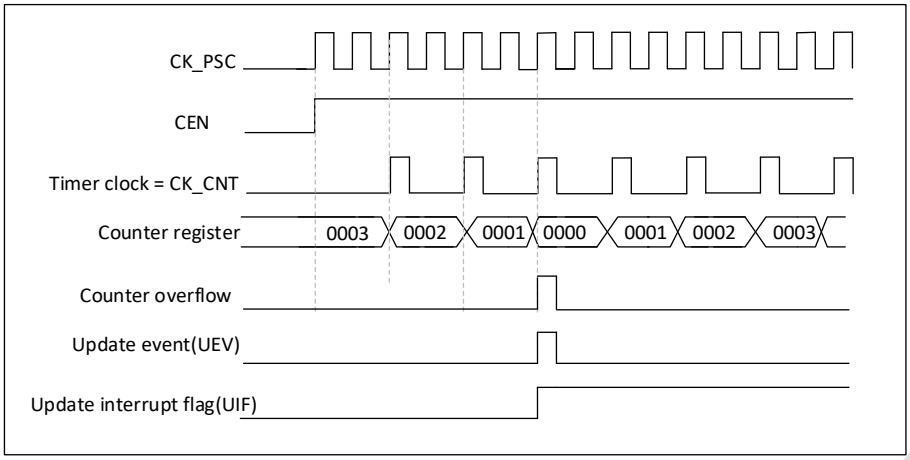


图 23-12 计数器时序图，内部时钟分频因子为 2

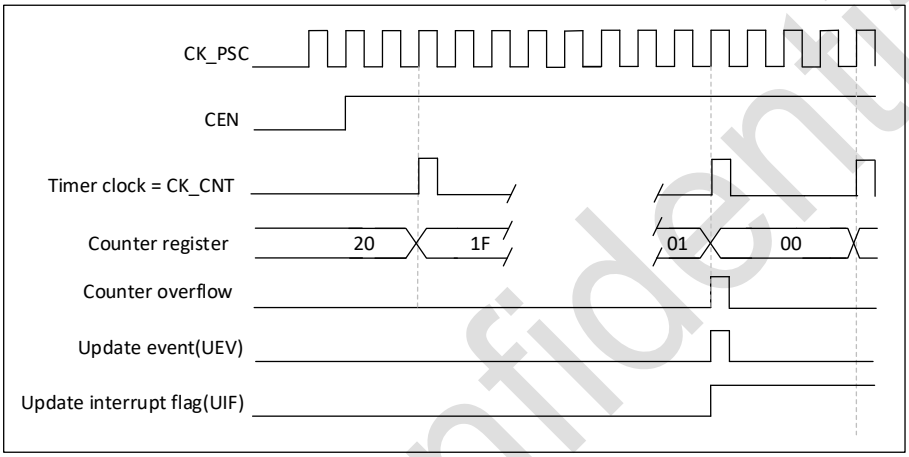


图 23-13 计数器时序图，内部时钟 N 分频

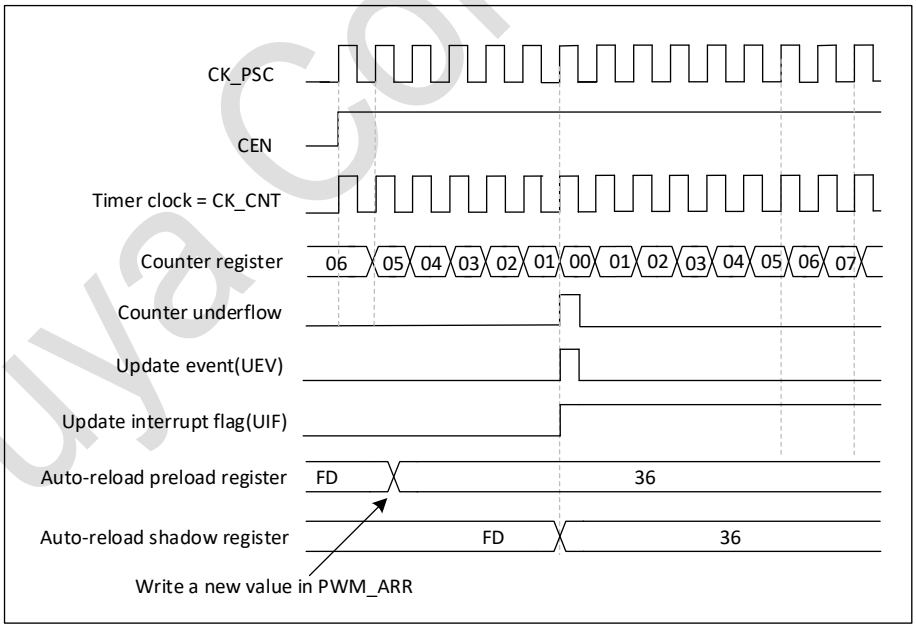


图 23-14 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

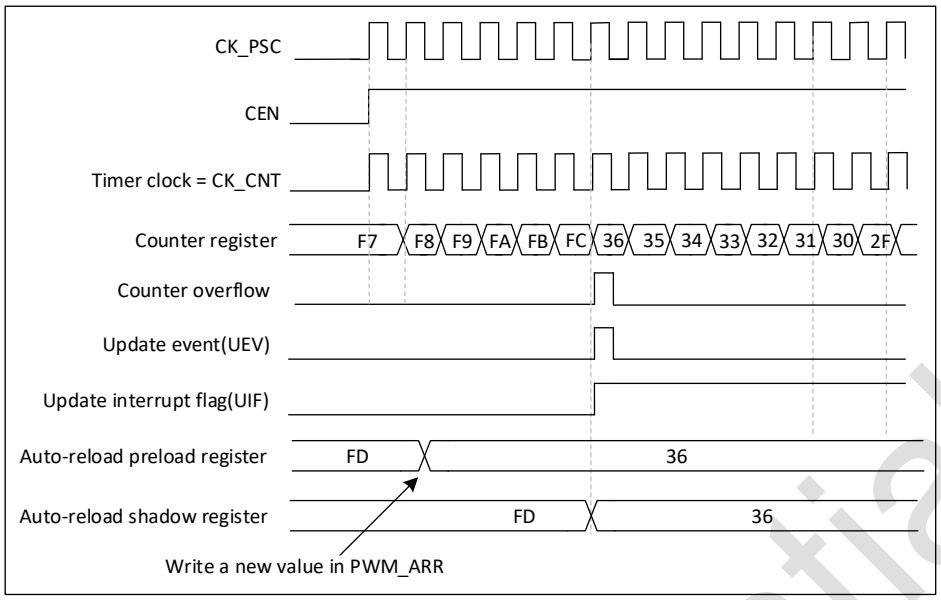


图 23-15 计数器时序图，ARPE=1 时的更新事件(计数器上溢)

23.2.3. 时钟选择

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：外部触发输入 ETR

23.2.3.1. 内部时钟源

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。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

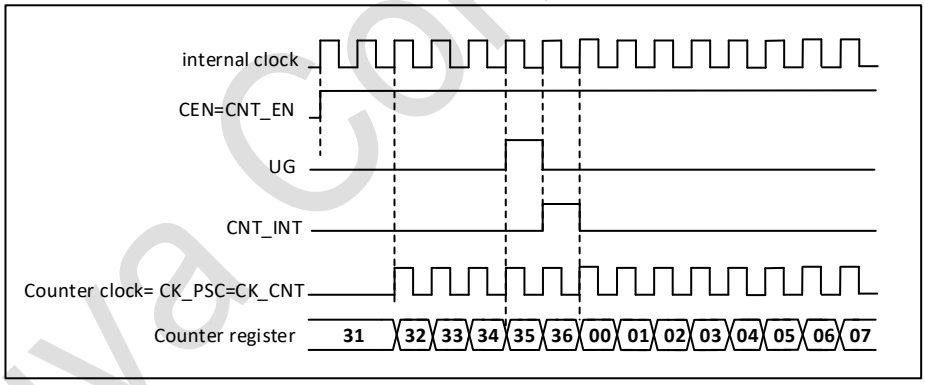


图 23-16 一般模式下的控制电路，内部时钟分频因子为 1

23.2.3.2. 外部时钟源模式

选定此模式的方法为：令 PWM_SMCR 寄存器中的 ECE=1，计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。当 ETRSEL 选择 100/101 时，外部时钟可高于 CK_INT，因此不能进行分频和滤波操作。

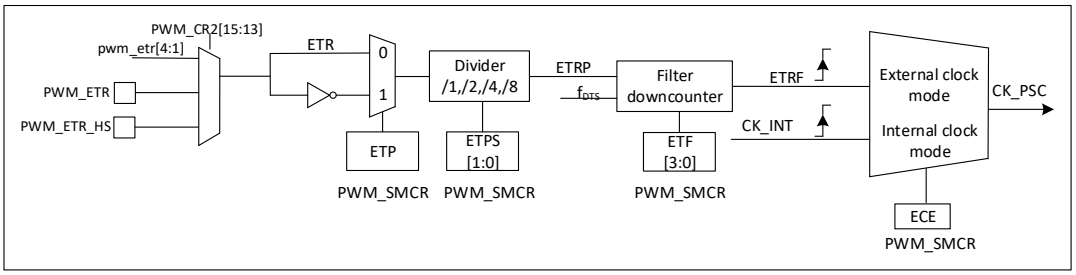


图 23-17 外部触发输入框图

例如，要配置在 PWM_ETR_HS 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：

1. 外部触发源选择 PWM_ETR_HS，配置 PWM_CR2 寄存器中的 ETRSEL[2:0]=101；
2. 本例中不能使能滤波器，配置 PWM_SMCR 寄存器中的 ETF[3:0]=0000；
3. 设置预分频器，配置 PWM_SMCR 寄存器中的 ETPS[1:0]=01；
4. 选择 ETR 输入端的上升沿，配置 PWM_SMCR 寄存器中的 ETP=0；
5. 开启外部时钟模式，写 PWM_SMCR 寄存器中的 ECE=1；
6.启动计数器，写 PWM_CR1 寄存器中的 CEN=1；

计数器在每 2 个 ETR 上升沿计数一次。

在 ETR 的上升沿和计数器实际时钟之间的延时取决于 ETRP 信号的重新同步电路。如果 ETRSEL=100/101 时，ETR 的上升沿和计数器实际时钟之间没有延时。

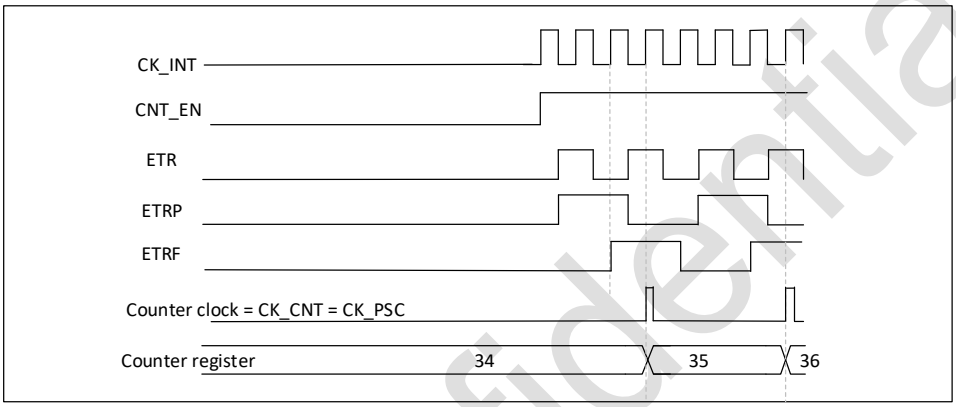


图 23-18 外部时钟模式下的控制电路

23.2.4. 输出比较模式

每一个比较通道都是围绕着一个比较寄存器(包含影子寄存器)，包括输出部分(比较器和输出控制)。

下面两张图是一个比较通道概览。

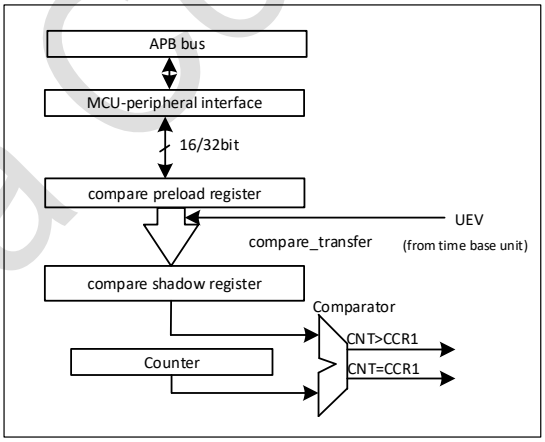


图 23-19 比较通道 1 的主电路

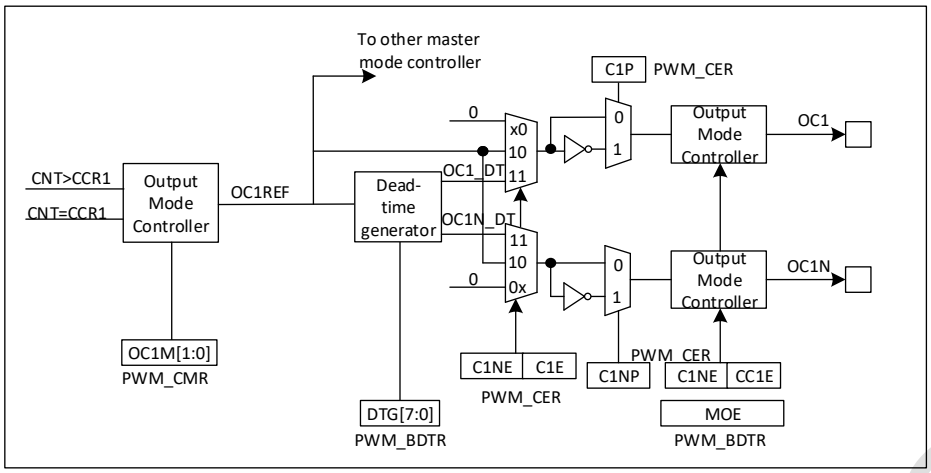


图 23-20 比较通道的输出部分(通道 1)

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。

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。

23.2.5. PWM 模式

该模块能产生一个由 PWM_ARR 寄存器确定频率、由 PWM_CCRx 寄存器确定占空比的信号。

在 PWM_CMR 寄存器中的 OCxM 位写入“10”（PWM 模式 1）或“11”（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。

如果要改变 PWM 的周期或者占空比，修改 PWM_ARR 和 PWM_CCRx 寄存器后，在下一个周期新值才会生效。

OCx 的极性可以通过软件在 PWM_CER 寄存器中的 CxP 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 PWM_CER 中 CxE 控制。

在 PWM 模式(模式 1 或模式 2)下，PWM_CNT 和 PWM_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 PWM_CCRx≤PWM_CNT 或者 PWM_CNT≤PWM_CCRx。

根据 PWR_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中心对齐的 PWM 信号。

23.2.5.1. PWM 边沿对齐模式

递增计数配置

当 PWM_CR1 寄存器中的 DIR 位为低的时候执行递增计数。参看下面是一个 PWM 模式 1 的例子。

当 PWM_CNT<PWM_CCRx 时，PWM 为 CxP 定义的有效电平，否则为无效电平。

如果 PWM_CCRx 中的比较值大于自动重装载值(PWM_ARR)，则 PWM 输出保持为’1’。如果比较值为 0，则输出保持为’0’。

下图为 PWM_ARR=8 时边沿对齐的 PWM 波形实例。

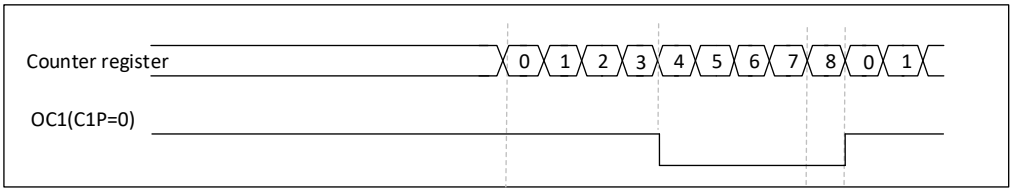


图 23-21 边沿对齐方式 PWM 输出，递增（ARR=8）

递减计数配置

当 PWM_CR1 寄存器的 DIR 位为高时执行递减计数。

在 PWM 模式 1，当 PWM_CNT>PWM_CCRx 时输出为无效电平，否则为有效电平。如果 PWM_CCRx 中的比较值大于 PWM_ARR 中的自动重装载值，则输出保持为'1'。该模式下不能产生 0%的 PWM 波形。

23.2.5.2. PWM 中心对齐模式

当 PWM_CR1 寄存器中的 CMS 位为 1 时为中心对齐模式。此时，PWM_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。

下图给出一些中心对齐的 PWM 波形的例子：

- PWM_ARR = 8
- PWM 模式 1
- PWM_CR1 寄存器的 CMS=01

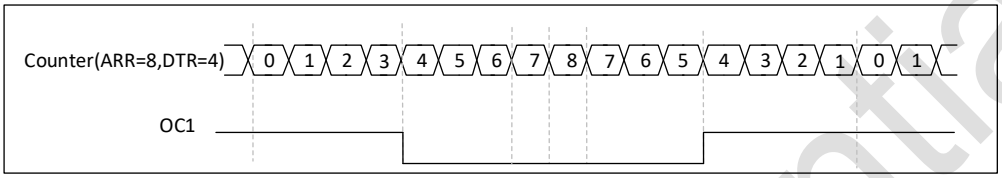


图 23-22 中心对齐模式 PWM 输出(ARR=8)

中心对齐模式使用建议：

- 进入中心对齐模式时，使用当前的递增/递减计数配置；这就意味着计数器递增还是递减计数取决于 PWM_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。
- 不推荐当运行在中心对齐模式时改写计数器，否则会产生不可预知的结果。尤其是：
  - 如果写入计数器的值大于自动重加载的值(PWM_CNT>PWM_ARR)，则方向不会被更新。例如，如果计数器正在递增计数，它就会继续递增计数。
  - 如果将 0 或者 PWM_ARR 的值写入计数器，计数方向会更新，但不产生更新事件 UEV。
- 使用中心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 PWM_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。

23.2.6. 互补输出和死区插入

PWM 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 PWM_CER 寄存器中的 CxP 和 CxNP 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)。

同时设置 CxE 和 CxNE 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 8 位的死区发生器 DTG[7:0]。

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。

下图显示了插入死区时 OCx 和 OCxN 的输出。(假设 CxP=0、CxNP=0、MOE=1、CxE=1 并且 CxNE=1)

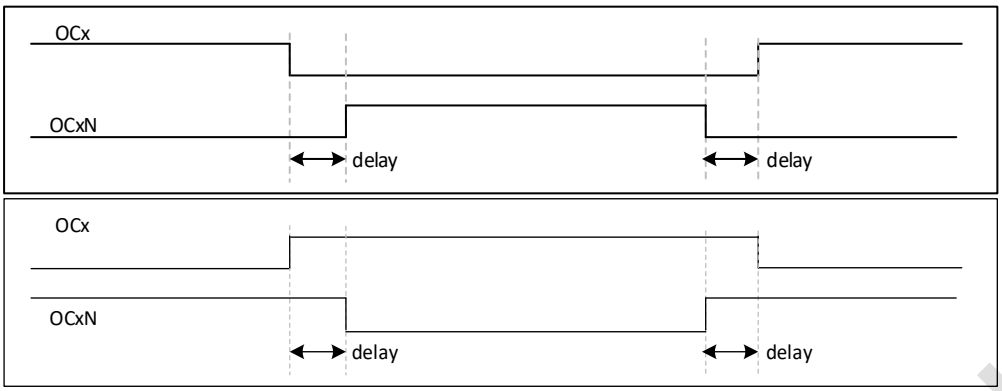


图 23-23 带死区插入的互补输出

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

23.2.7. 使用刹车功能

当使用刹车功能时，依据额外的控制位，输出使能信号和无效电平信号都会被修改。无论什么情况下，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源既可以是刹车输入引脚，或者以下内部源：

- CPU LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟错误事件
- 来自比较器的输出
- SRAM 字节校验错误
- FLASH ECC 错误

系统复位后，刹车电路被禁止，MOE 位为低。设置 PWM_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 PWM_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，将输出禁止。这个特性在 MCU 的时钟振荡器关闭时依然有效。
- 如果设置了 PWM_DIER 寄存器中的 BIE 位，当刹车状态标志(PWM_SR 寄存器中的 BIF 位)为’1’时，则产生一个中断。

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。

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 PWM_BDTR 寄存器中的 BKE 位开启。

23.3. PWM 中断

表 23-1 PWM 中断

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
输出比较 1	OC1IF	OC1IE
输出比较 2	CC2IF	OC2IE
输出比较 3	CC3IF	OC3IE
输出比较 4	CC4IF	OC4IE
刹车	BIF	BIE

23.4. PWM 调试模式

当微控制器进入调试模式时(Cortex®-M0+核停止)，根据 DBGMCU 模块中 DBG_PWM_STOP 的设置，PWM 可以或者继续正常操作，或者停止。

23.5. PWM 寄存器

23.5.1. PWM 控制寄存器 1 (PWM_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	Res.	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	2'b0	时钟分频因子。 这 2 位定义在定时器时钟(CK_INT)频率和死区时间和死区发生器与数字滤波器(ETR)所用的采样时钟（t_{DTS}）之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： PWM_ARR 寄存器没有缓冲 1： PWM_ARR 寄存器被装入缓冲器
6:5	CMS[1:0]	RW	00	计数模式。 00：边沿对齐模式。计数器依据方向位(DIR)递增或递减计数。 01：中心对齐模式 1。计数器交替地递增和递减计数。配置为输出通道的输出比较中断标志位，只在计数器递减计数时被设置。 10：中心对齐模式 2。计数器交替地递增和递减计数。配置为输出的通道的输出比较中断标志位，只在计数器递增计数时被设置。

				11：中心对齐模式 3。计数器交替地递增和递减计数。配置为输出的通道的输出比较中断标志位，在计数器递增和递减计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中心对齐模式。
4	DIR	RW	0	计数方向。 0：计数器递增计数 1：计数器递减计数 注：当计数器配置为中心对齐模式时，该位为只读。
3:1	Reserved	-	-	保留
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： − 计数器上溢/下溢 − 设置 UG 位 1：如果允许产生更新中断或 DMA 请求，则只有计数器上溢/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	更新禁止 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： − 计数器上溢/下溢 − 设置 UG 位 影子寄存器装入预装载值。 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，则重新初始化计数器和预分频器。
0	CEN	RW	0	计数器使能。 0： 禁止计数器 1： 使能计数器 注：在软件设置了 CEN 位后，才可以使用外部时钟、门控模式和编码器模式。触发模式可以通过硬件自动地设置 CEN 位为 1。

23.5.2. PWM 控制寄存器 2（PWM_CR2）

偏移地址:0x04

复位值:0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[2:0]			Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW	RW		RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:13	ETRSEL[2:0]	RW	3’b0	外部触发源选择 该位段用于选择 ETR 输入源

				000：PWM_ETR_LS（低速） 001：COMP1_OUT 010：COMP2_OUT 011：保留 (以上部分 ETR 源可以使能滤波) 100：SYS_CLK 101：PWM_ETR_HS（高速） （当配置为 100/101 时，触发频率可以超过 PWM PCLK，故无法使能分频和滤波功能，需要保证无毛刺输入） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 1，则这些位不能被修改。
12	Reserved	-	-	保留
11	BKCMP2P	RW	0	来源于比较器 2 刹车输入极性控制。 该位选择来源于比较器 2 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 2 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 2 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	来源于比较器 1 刹车输入极性控制。 该位选择来源于比较器 1 的刹车输入极性，必须与 BKP 极性位同时配置。 0：比较器 1 刹车输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：比较器 1 刹车输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	比较器 2 刹车输入使能。 比较器 2 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 2 刹车输入关闭 1：比较器 2 刹车输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	比较器 1 刹车输入使能。

				比较器 1 刹车输入与其它刹车源进行或逻辑作为刹车输入。 0：比较器 1 刹车输入关闭 1：比较器 1 刹车输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能。 该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

23.5.3. PWM 从模式控制寄存器（PWM_SMCR）

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE.	ETPS[1:0]		ETF[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW								

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ETP	RW	0	外部触发极性。该位选择是 ETR 或者 ETR 的反相被用作触发操作。 0：ETR 不进行反相，高电平或者上升沿有效 1：ETR 反相，低电平或者下降沿有效
14	ECE	RW	0	外部时钟使能位 该位启用外部时钟模式 0：禁止外部时钟模式； 1：使能外部时钟模式。 计数器由 ETRF 信号上的任意有效边沿驱动。
13:12	ETPS[1:0]	RW	2’b0	外部触发预分频器。 外部触发信号 ETRP 的频率必须最多是计数时钟频率的 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：预分频器关闭 01：ETR 频率的 2 分频 10：ETR 频率的 4 分频 11：ETR 频率的 8 分频 **注意**:当 PWR_CR2.ETRSEL 选择 100/101 时，不支持预分频。
11:8	ETF[3:0]	RW	4’b0	外部触发滤波 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 个事件后才视为一个有效输出边沿。 0000：无滤波器，异步 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2

				0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}= f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}= f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}= f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}= f_{DTS}/16$, N=5 1011：采样频率 $f_{SAMPLING}= f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8 注意:当 ETRSEL 选择 100/101 时，不支持滤波。
7:0	Reserved	-	-	保留

23.5.4. PWM DMA/中断使能寄存器 (PWM_DIER)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	OC4DE	OC3DE	OC2DE	OC1DE	UDE	BIE	Res.	Res.	OC4IE	OC3IE	OC2IE	OC1IE	UIE
			RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	OC4DE	RW	0	输出比较 4 的 DMA 请求使能 0：禁止输出比较 4 的 DMA 请求 1：允许输出比较 4 的 DMA 请求
11	OC3DE	RW	0	输出比较 3 的 DMA 请求使能 0：禁止输出比较 3 的 DMA 请求 1：允许输出比较 3 的 DMA 请求
10	OC2DE	RW	0	输出比较 2 的 DMA 请求使能 0：禁止输出比较 2 的 DMA 请求 1：允许输出比较 2 的 DMA 请求
9	OC1DE	RW	0	输出比较 1 的 DMA 请求使能 0：禁止输出比较 1 的 DMA 请求 1：允许输出比较 1 的 DMA 请求
8	UDE	RW	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	刹车中断 0：禁止刹车中断 1：允许刹车中断
6:5	Reserved	-	-	保留

4	OC4IE	RW	0	比较输出 4 中断使能 0：禁止比较输出 4 中断 1：允许比较输出 4 中断
3	OC3IE	RW	0	比较输出 3 中断使能 0：禁止比较输出 3 中断 1：允许比较输出 3 中断
2	OC2IE	RW	0	比较输出 2 中断使能 0：禁止比较输出 2 中断 1：允许比较输出 2 中断
1	OC1IE	RW	0	比较输出 1 中断使能 0：禁止比较输出 1 中断 1：允许比较输出 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：使能更新中断

23.5.5. PWM 状态寄存器 (PWM_SR)

偏移地址:0x010

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARROK	PSCOK	CNTOK	CCR4OK	CCR3OK	CCR2OK	CCR1OK	Res.	BIF	Res.	Res.	OC4IF	OC3IF	OC2IF	OC1F	UIF
RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0		RC_W0			RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ARROK	RC_W0	0	PWM_ARR 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 ARROK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_ARR 的写操作已成功完成。 该寄存器由软件清零。
14	PSCOK	RC_W0	0	PWM_PSC 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 PSCOK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_PSC 的写操作已成功完成。 该寄存器由软件清零。
13	CNTOK	RC_W0	0	PWM_CNT 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CNTOK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CNT 的写操作已成功完成。 该寄存器由软件清零。
12	CCR4OK	RC_W0	0	PWM_CCR4 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR4OK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR4 的写操作已成功完成。 该寄存器由软件清零。

11	CCR3OK	RC_W0	0	PWM_CCR3 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR3OK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR3 的写操作已成功完成。 该寄存器由软件清零。
10	CCR2OK	RC_W0	0	PWM_CCR2 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR2OK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR2 的写操作已成功完成。 该寄存器由软件清零。
9	CCR1OK	RC_W0	0	PWM_CCR1 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR1OK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR1 的写操作已成功完成。 该寄存器由软件清零。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标志 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。
6:5	Reserved	-	-	保留
4	OC4IF	RC_W0	0	输出比较 4 中断标志 参考 OC1IF 描述
3	OC3IF	RC_W0	0	输出比较 3 中断标志 参考 OC1IF 描述
2	OC2IF	RC_W0	0	输出比较 2 中断标志 参考 OC1IF 描述
1	OC1IF	RC_W0	0	输出比较 1 中断标志 当计数器值与比较值匹配时该位由硬件置 1。 0：无匹配发生； 1：PWM_CNT 的值与 PWM_CCR1 的值匹配。
0	UIF	RC_W0	0	更新中断标志 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：更新事件产生(计数器上溢或者下溢或者置位 UG)。 该位由硬件置 1。

23.5.6. PWM 事件产生寄存器（PWM_EGR）

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
															W

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	UG	W	0	产生更新事件。 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)。

23.5.7. PWM 输出比较模式寄存器 1(PWM_CMR)

偏移地址:0x18

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	OC4PE	OC3PE	OC2PE	OC1PE	OC4M[1:0]		OC3M[1:0]		OC2M[1:0]		OC1M[1:0]	
				RW	RW	RW	RW	RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	OC4PE	RW	0	输出比较 4 预装载使能
10	OC3PE	RW	0	输出比较 3 预装载使能
9	OC2PE	RW	0	输出比较 2 预装载使能
8	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 PWM_CCR1 寄存器的预装载功能，可随时写入 PWM_CCR1 寄存器，且新值马上起作用。 1：开启 PWM_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，PWM_CCR1 的预装载值在每次产生更新事件时装载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(PWM_BDTR 寄存器中的 LOCK 位)则该位不能被修改。
7:6	OC4M[1:0]	RW	0	OC4 PWM 模式。参考 OC1M 的描述。
5:4	OC3M[1:0]	RW	0	OC3 PWM 模式。参考 OC1M 的描述。
3:2	OC2M[1:0]	RW	0	OC2 PWM 模式。参考 OC1M 的描述。
1:0	OC1M[1:0]	RW	0	OC1 PWM 模式。 10：PWM 模式 1---在递增计数时，一旦 PWM_CNT<PWM_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 PWM_CNT>PWM_CCR1 时通道 1 为无效电平，否则为有效电平。 11：PWM 模式 2---在递增计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为无效电平，否则为有效电平；在递减计数时，一旦 PWM_CNT>PWM_ CCR1 时通道 1 为有效电平，否则为无效电平。 其他值:保留 注：有效电平由 PWM_CER 寄存器配置，无效电平为取反。

23.5.8. PWM 输出比较使能寄存器 (PWM_CER)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	C4P	C4E	Res.	Res.	C3P	C3E	C2NP	C2NE	C2P	C2E	C1NP	C1NE	C1P	C1E
		RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	C4P	RW	0	OC4 输出极性。参考 C1P 的描述。
12	C4E	RW	0	OC4 输出使能。参考 C1E 的描述。
11:10	Reserved	-	-	保留
9	C3P	RW	0	OC3 输出极性。参考 C1P 的描述。
8	C3E	RW	0	OC3 输出使能。参考 C1E 的描述。
7	C2NP	RW	0	OC2 互补输出极性。参考 C1NP 的描述。
6	C2NE	RW	0	OC2 互补输出使能。参考 C1NE 的描述。
5	C2P	RW	0	OC2 输出极性。参考 C1P 的描述。
4	C2E	RW	0	OC2 输出使能。参考 C1E 的描述。
3	C1NP	-	0	OC1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 则该位不能被修改。
2	C1NE	RW	0	OC1 互补输出使能 0：关闭 - OC1N 禁止输出，输出电平为高阻。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平为 OC1 的互补。
1	C1P	RW	0	OC1 输出极性 0：OC1 高电平有效 1：OC1 低电平有效 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 则该位不能被修改。
0	C1E	RW	0	OC1 输出使能 0：关闭 - OC1 禁止输出，输出电平为高阻。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 C1P 的值。

MOE	CxE	CxNE	OCx 输出状态	OCxN 输出状态
1	0	0	输出禁止(不由定时器驱动)，OCx=0, OCx_EN=0	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
	0	1	输出禁止(不由定时器驱动)，OCx=0，OCx_EN=0	OCxREF + 极性 OCxN=OCxREF 异或 CCxNP, OCxN_EN=1
	1	0	OCxREF +极性 OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(不由定时器驱动)，OCxN=0, OCxN_EN=0
	1	1	OCREF +极性+ 死区 OCx_EN=1	OCREF 的互补（OCREF 非）+极性+ 死区 OCxN_EN=1
0	X	X	输出禁止(与定时器断开)	

23.5.9. PWM 计数寄存器（PWM_CNT)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	0	计数器的值

23.5.10. PWM 预分频器（PWM_PSC)

偏移地址:0x28

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	0	预分频器的值 计数器的时钟频率(CK_CNT)等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时，要装载到有效预分频器寄存器的值。

23.5.11. PWM 自动装载寄存器（PWM_ARR)

偏移地址:0x2c

复位值:0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'hFFFF	自动重装载的值 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。

23.5.12. PWM 比较寄存器 1(PWM_CCR1)

偏移地址:0x34

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1[15:0]	RW	16'b0	比较 1 的值。 CCR1 包含了装入当前比较 1 寄存器的值（预装载值）。 如果在 PWM_CMR 寄存器(OC1PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC1 端口上输出 PWM 信号。

23.5.13. PWM 比较寄存器 2(PWM_CCR2)

偏移地址:0x38

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2[15:0]	RW	0	比较 2 的值。 CCR2 包含了装入当前比较 2 寄存器的值（预装载值）。 如果在 PWM_CMR 寄存器(OC2PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC2 端口上输出 PWM 信号。

23.5.14. PWM 比较寄存器 3(PWM_CCR3)

偏移地址:0x3C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR3[15:0]	RW	0	比较 3 的值。 CCR3 包含了装入当前比较 3 寄存器的值（预装载值）。 如果在 PWM_CMR 寄存器(OC3PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC3 端口上输出 PWM 信号。

23.5.15. PWM 比较寄存器 4(PWM_CCR4)

偏移地址:0x40

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4[15:0]	RW	0	比较 4 的值 CCR4 包含了装入当前比较 4 寄存器的值（预装载值）。 如果在 PWM_CMR 寄存器(OC4PE 位)中未选择预装载特性，则该值立刻生效。否则，只有当更新事件发生时该值才生效。 实际比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC4 端口上输出 PWM 信号。

23.5.16. PWM 刹车和死区寄存器(PWM_BDTR)

偏移地址:0x44

复位值:0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	Res.	Res.	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	刹车主输出使能 一旦刹车输入有效，该位被硬件异步清 0。 0：禁止 OCx 和 OCxN 输出； 1：如果设置了相应的使能位（PWM_CER 寄存器的 CxE、CxNE 位），则开启 OCx 和 OCxN 输出。
14	AOE	RW	0	自动输出使能 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自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入； 1：开启刹车输入。
11:10	Reserved	-	-	保留
9:8	LOCK[1:0]	RW	00	锁定设置 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 PWM_BDTR 寄存器的 DTG/BKE/BKP/AOE 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 CxP/CNxP 极性位

				11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CxM 控制位 注：在系统复位后，只能写一次 LOCK 位，一旦写入 PWM_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	0000 0000	死区时间设置。 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] × T_{dtg}，T_{dtg} g = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG[5:0]) ×T_{dtg}，T_{dtg} = 2 × T_{DTS}； DTG[7:5]=110 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 8 × T_{DTS}； DTG[7:5]=111 => DT=(32+DTG[4:0]) ×T_{dtg}，T_{dtg} = 16 × T_{DTS}； 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(PWM_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则这些位不能被修改。

23.5.17. PWM DMA 控制寄存器(PWM_DCR)

偏移地址:0x48

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res.	Res	DBL[4:0]					Res	Res	Res	DBA[4:0]				
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	0	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度(当对 PWM_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 以这样的传输为例：DBL=7 字节，DBA=PWM_CR1： --如果 DBL=7 字节，DBA=PWM_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (PWM_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引=DBL，其中(PWM_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(PWM_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	Reserved	-	-	保留
4:0	DBA	RW	0	DMA 基地址 这些位定义了 DMA 在连续模式下的基地址(当对 PWM_DMAR 寄存器进行读或写时)， DBA 定义为从 PWM_CR1 寄存器所在地址开始的偏移量： 00000：PWM_CR1， 00001：PWM_CR2， 00010：PWM_SMCR， 00011：PWM_DIER， 00100：PWM_SR， ……..

23.5.18. PWM 连续模式的 DMA 地址（PWM_DMAR）

偏移地址:0x4C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAR[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	0	DMA 连续传送寄存器。 对 PWM_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： PWM_CR1 地址 + (DBA + DMA 索引)x4 其中： “PWM_CR1 地址”是控制寄存器 1(PWM_CR1)所在的地址； “DBA”是 PWM_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 PWM_DCR 寄存器中定义的 DBL。

注：在使用 DMA 连续传输功能时，必须将 DMA 中对应通道的 CNDTR 寄存器的值与 PWM_DCR 寄存器中 DBL 的值对应起来，否则该将不能正常使用。

24. 红外接口 (IRTIM)

芯片内集成为遥控而用的红外接口（IRTIM）。它可以与一个红外 LED 一起实现遥控的功能。

为产生红外遥控信号，必须打开红外接口（Infrared interface），并且正确配置 TIM16 的通道 1(TIM16_OC1)和 TIM17 通道 1（TIM17_OC1），以产生正确的波形。

红外接收器可以很容易通过一个基本输入捕获模式实现。

所有标准的红外脉冲调制模式可以通过编程两个定时器的输出比较通道获得。

TIM17 被用来产生高频载波信号，而 TIM16 可以产生调制包络。

红外功能输出到 IR_OUT 引脚，这个功能的激活是通过使能 GPIO_AFRx 寄存器的相关复用功能位实现的。

LED 需要大的灌电流驱动能力（针对 PB2 ~ PB9），这可以通过 SYSCFG_LEDCFG 寄存器的 PB_EHS 寄存器被激活，这样就足够支撑直接控制红外 LED 大的灌电流而用。

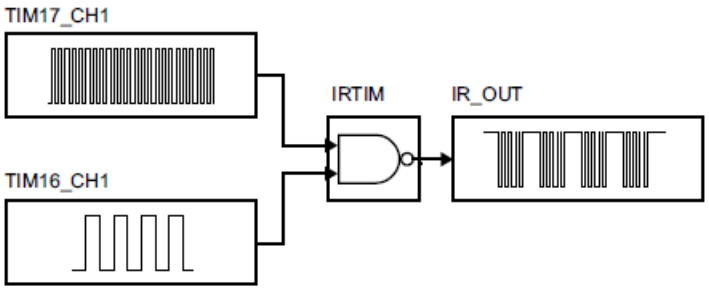


图 24-1 IRTIM 实现

25. 低功耗定时器 (LPTIM)

25.1. LPTIM 简介

本产品包含 2 个 LPTIM，其中 LPTIM1 支持 32bit 计数，LPTIM2 支持 16bit 计数。LPTIM 将系统从低功耗模式中唤醒的能力使得它适合于实现除待机模式之外的低功耗应用。LPTIM 支持没有内部时钟源也能运行，可以用作“脉冲计数器”。

LPTIM 引入了一种灵活的时钟方案，可提供所需的功能和性能，同时将功耗降至最低。

25.2. LPTIM 主要特性

- 16 位/32 位递增计数器
- 3 位预分频器，具有 8 个可能的分频因子（1、2、4、8、16、32、64、128）
- 可选时钟
  - 内部时钟源：LSE，LSI 或 APB 时钟
- 16/32 位自动重载寄存器
- 16/32 位比较寄存器
- 单次/连续模式
- 可选软件/硬件输入触发
- 触发信号可配置数字滤波器
- 可配置输出：
  - 脉冲
  - PWM
- 可配置 IO 触发极性
- 编码器模式

25.3. LPTIM 功能描述

25.3.1. LPTIM 框图

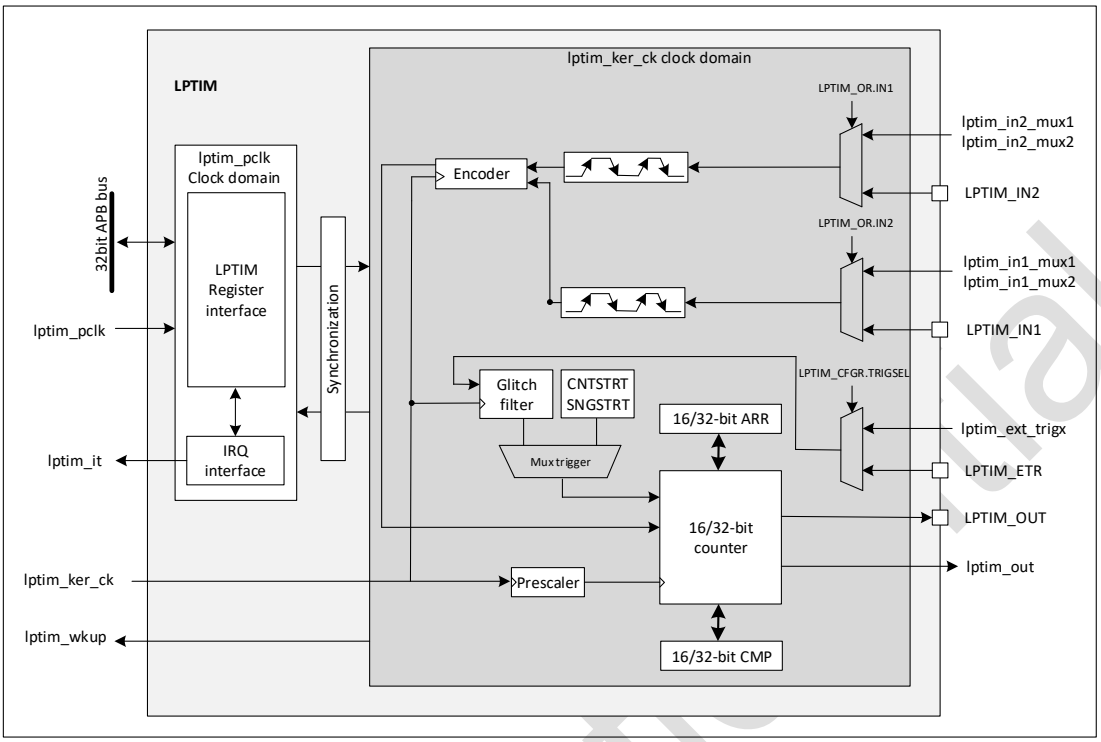


图 25-1 LPTIM 框图

25.3.2. LPTIM 引脚和接口描述

表 25-1 LPTIM 接口信号

名称	输入/输出	描述
lptim_pclk	输入	LPTIM APB 时钟
lptim_ker_ck	输入	LPTIM 内部时钟源
lptim_in1_mux*	输入	LPTIM 输入组 1 内部来源（2 个） lptim_in1_mux1：COMP1_OUT lptim_in1_mux2：COMP2_OUT
LPTIM_IN1	输入	LPTIM 输入组 1 IO 来源
lptim_in2_mux*	输入	LPTIM 输入组 2 内部来源（2 个） lptim_in2_mux1：COMP1_OUT lptim_in2_mux2：COMP2_OUT
LPTIM_IN2	输入	LPTIM 输入组 2 IO 来源
lptim_ext_trigx	输入	LPTIM 外部触发来源 x（来自其他模块），参见“外设互连”章节。
LPTIM_ETR	输入	LPTIM 外部触发 IO 来源
lptim_it	输出	LPTIM 全局中断
lptim_wkup	输出	LPTIM 唤醒事件
lptim_out	输出	LPTIM 输出（内部互连使用）

25.3.3. LPTIM 复位和时钟

通过 RCC 模块，可以使用内部时钟信号对 LPTIM 进行时钟控制（该时钟信号可以在 PCLK、LSI、LSE 中进行选择）。

25.3.4. 触发信号毛刺滤波器

LPTIM 输入，无论是外部（映射到 GPIO）还是内部（在芯片级映射到其他嵌入式外围设备，如比较器）触发，都受到数字滤波器的保护，可防止任何毛刺和噪声扰动在 LPTIM 内部传播。这是为了防止意外计数或触发。在激活数字滤波器之前，应首先向 LPTIM 提供一个内部时钟源。这是保证滤波器正常运行所必需的。LPTIM 包含如下数字滤波器：

- 数字滤波器保护 LPTIM 内部触发输入。数字滤波器灵敏度由 TRGFLT 位控制。

注意：数字滤波器灵敏度由组控制。不可能在同一组内单独配置每个数字滤波器灵敏度。

滤波器灵敏度作用于在 LPTIM 某一输入上检测到的连续相等样本的数量，以将信号电平变化视为有效转换。下图为在编程 2 个连续样本的情况下毛刺滤波器行为的示例。

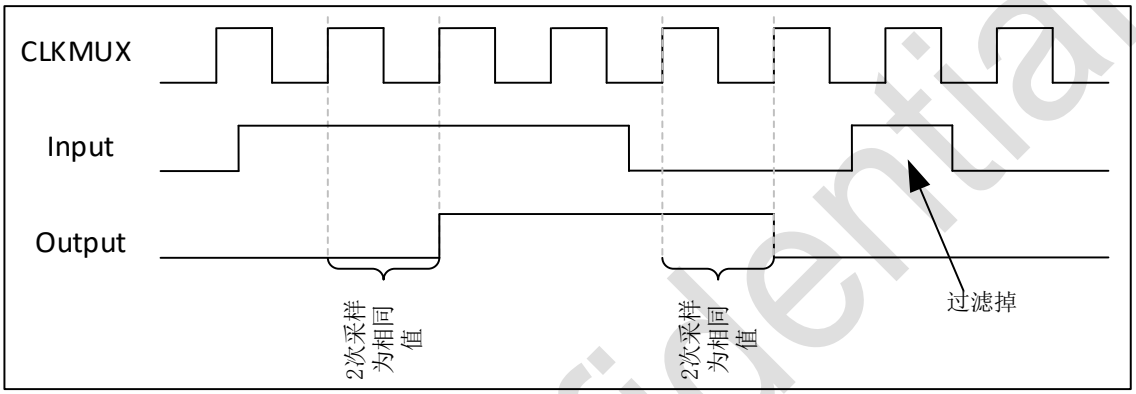


图 25-2 毛刺滤波器行为示例（2 个连续样本）

注意：如果没有提供内部时钟信号，则必须通过将 TRGFLT 位设置为“0”来停用数字滤波器。在这种情况下，可以使用外部模拟滤波器（IO 模块实现）来保护 LPTIM 外部输入免受毛刺干扰。

25.3.5. 预分频器

LPTIM 计数器，由一个可配置的 2 次幂预分频器控制驱动。预分频器分频比由 PRESC[2:0]控制。

下表列出了所有情况：

表 25-2 预分频器的分频比

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

25.3.6. 触发选择

LPTIM 计数器可以通过软件启动，也可以在检测到 8 个（可选最大输入个数）触发输入之一的激活边沿后启动。

TRIGEN[1:0]用于确定 LPTIM 触发源：

- 当 TRIGEN[1:0]等于“00”时，一旦 CNTSTRT 或 SNGSTRT 位之一由软件设置，LPTIM 计数器就会启动。TRIGEN[1:0]的其他值可用于配置触发输入使用的有效边沿，支持上升沿和下降沿。一旦检测到有效边沿，LPTIM 计数器就会启动。
- 当 TRIGEN[1:0]不是‘00’时，TRIGSEL[2:0]用于选择 8 个触发输入中的哪一个启动计数器。

外部触发器被视为 LPTIM 的异步信号。因此，在触发检测之后，由于同步，在定时器开始运行之前需要两个计数器时钟周期的延迟。

如果在定时器已经启动时发生新的触发事件，它将被忽略。

在设置 SNGSTRT/CNTSTRT 位之前必须使能定时器。当定时器被禁用时，对这些位的任何写入都将被硬件丢弃。

注意：当通过软件启动计数器时（TRIGEN[1:0] = 00），LPTIM_CR 寄存器更新（设置 SNGSTRT 或 CNTSTRT 位之一）和计数器有效启动之间有 3 个计数时钟周期的延迟。

25.3.7. 操作模式

LPTIM 具有两种如下工作模式：

- 连续计数模式：计时器自由运行，从触发事件开始运行，直到计时器被禁止时才停止。
- 单次计数模式：定时器从一个触发事件开始，当达到 ARR 值时停止。

25.3.7.1. 单次模式

要使能单次计数，LPTIM_CR.SNGSTRT 寄存器位必须置 1。

设置 SNGSTRT 将启动计数器进行单次计数。

一个新的触发事件将重新启动计时器。在计数器启动之后，到达 ARR 之前，任何触发事件都将被忽略。

如果选择外部触发，则在设置 SNGSTRT 位后以及计数器寄存器停止（包含零值）后，再产生外部触发事件，LPTIM 将启动计数器进行新的单次计数周期，如下图所示。

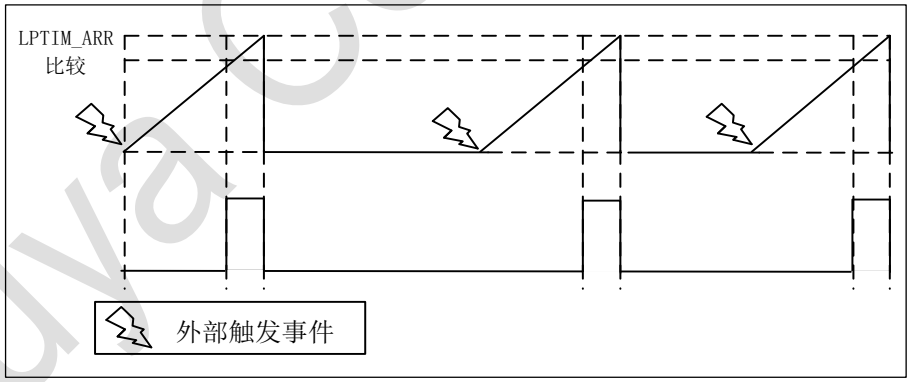


图 25-3 LPTIM 输出,单次计数模式

当 LPTIM_CFGR.WAVE 位置 1 时，单次模式被激活。这种情况下，计数器仅在第一次触发后启动一次，随后的任何触发事件均被忽略。如下图所示：

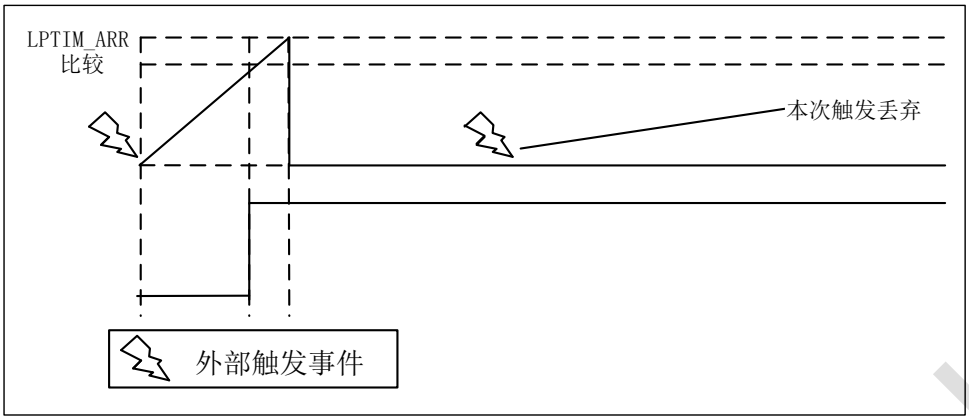


图 25-4 LPTIM 输出,单次计数模式（WAVE=1）

25.3.7.2. 连续模式

要使能连续计数，LPTIM_CR.CNTSTRT 位必须置 1。

如果选择外部触发，则在 CNTSTRT 置位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。

在软件启动(TRIGEN[1:0]='00')的情况下，设置 LPTIM_CR.CNTSTRT 将启动计数器进行连续计数。

连续计数波形如下：

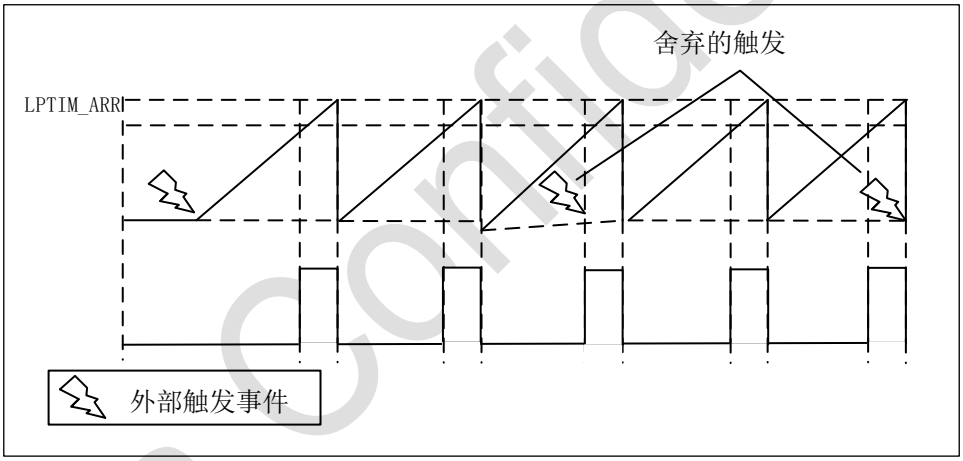


图 25-5 LPTIM 输出,连续计数模式

LPTIM_CR.SNGSTRT 和 LPTIM_CR.CNTSTRT 位只能在定时器使能时置位（LPTIM_CR.ENABLE 为“1”）。

可以“实时”地从单次模式转变为连续模式：

- 如果之前选择了连续模式，则置位 LPTIM_CR.SNGSTRT 会将 LPTIM 切换到单次模式。计数器（如果激活）将在到达 ARR 就会停止。
- 如果先前选择了单触发模式，则置位 LPTIM_CR.CNTSTRT 会将 LPTIM 切换到连续模式。计数器（如果激活）将在达 ARR 后就会重新启动。

25.3.8. 波形产生

两个寄存器，LPTIM_ARR（自动重载寄存器）和 LPTIM_CMP（比较寄存器），用于在 LPTIM 输出上生成几种不同的波形。

定时器可以生成以下波形：

- PWM 模式：LPTIM 输出置位条件为 LPTIM_CNT 中的计数器值超过了 LPTIM_CMP 中的比较值。一旦 LPTIM_ARR 和 LPTIM_CNT 寄存器之间发生匹配，LPTIM 输出就会复位。

- 单脉冲模式：输出波形类似于第一个脉冲的 PWM 模式，然后输出永久复位。
- 一次设置模式：输出波形类似于单脉冲模式，不同点在于输出会保持在最后一个信号电平（取决于输出配置的极性）。

上述模式要求 LPTIM_ARR 寄存器值严格大于 LPTIM_CMP 寄存器值。

LPTIM 输出波形可通过 WAVE 位进行如下配置：

- 将 WAVE 位重置为“0”会强制 LPTIM 生成 PWM 波形或单脉冲波形，具体取决于设置的位：CNTSTRT 或 SNGSTRT。
- 将 WAVE 位设置为“1”会强制 LPTIM 生成一次设置模式波形。

WAVPOL 位控制 LPTIM 输出极性。更改立即生效，因此输出默认值将在重新配置极性后立即更改，甚至在启用定时器之前。可以生成频率高达 1/2 LPTIM 时钟频率的信号。下图显示了可在 LPTIM 输出上生成的三种可能波形。此外，它还显示了使用 WAVPOL 位改变极性的效果。

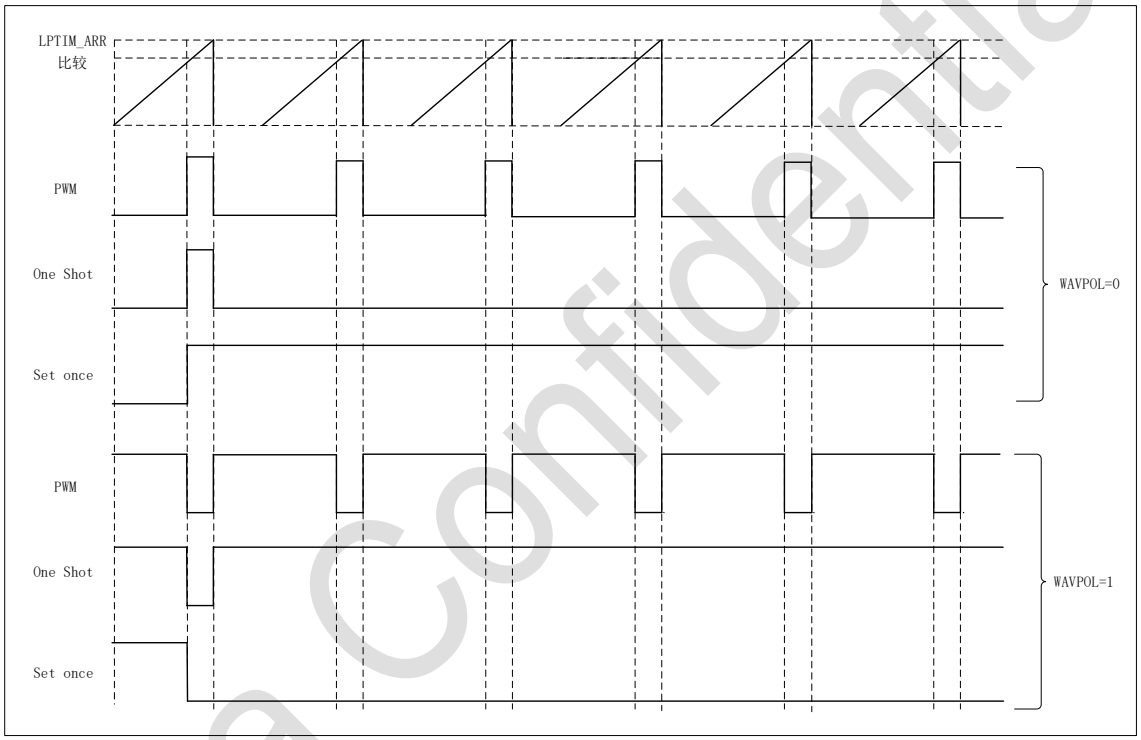


图 25-6 LPTIM 波形产生

25.3.9. 寄存器更新

LPTIM_ARR 寄存器在 APB 总线写操作后立即更新，如果定时器已经启动，则与下一个 LPTIM 更新事件同步进行更新。

PRELOAD 位控制 LPTIM_ARR 寄存器的更新方式：

- 当 PRELOAD 位被复位为“0”：LPTIM_ARR 寄存器在任何写访问后立即更新。
- 当 PRELOAD 位被设为“1”时：如果定时器已经启动，则 LPTIM_ARR 寄存器将在当前周期结束时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用不同的时钟，因此在 APB 写入,和写入的值被应用到计数器比较器时，存在一定的延迟。在此延迟周期内，必须避免对这些寄存器进行任何额外的写操作。

LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK，指示对 LPTIM_ARR 和 LPTIM_CMP 寄存器的写操作是否完成。

对 LPTIM_ARR 和 LPTIM_CMP 寄存器进行写操作后，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 标志位或者 CMPOK 标志位置位之前的任何连续写入都将导致不可预测的结果。

25.3.10. 计数器模式

可选择以下计数模式：

- LPTIM 由内部时钟源计时

LPTIM 时钟由内部时钟源提供，LPTIM 计数器配置为在每个内部时钟脉冲后更新。

25.3.11. 定时器使能

LPTIM_CR 寄存器中的 ENABLE 位用于使能/不使能 LPTIM 内核逻辑。置位 ENABLE 位后，需要延迟 3 个**计数器时钟**（内核时钟经过 PRESC 分频后的时钟）才能真正使能 LPTIM。

仅当 LPTIM 禁用时，才能修改 LPTIM_CFGR 和 LPTIM_IER 寄存器。

25.3.12. 定时器复位

为了将 LPTIM_CNT 寄存器的内容复位，提供如下所述两种复位机制。

注意：异步复位和同步复位应确保互斥使用。

25.3.12.1. 异步复位

异步复位由 LPTIM_CR 寄存器的 RSTARE 位控制。当该位被置为 1 时，任何读 LPTIM_CNT 寄存器的访问都将其内容复位为零。异步复位应在未提供 LPTIM 内核时钟的时间范围内触发。

应注意，为了可靠地读取 LPTIM_CNT 寄存器，必须进行 2 次读访问并比较其结果，结果一致，被则认为是读出值是可靠的。

需要注意的是：

- 使能异步复位时，读取两次 LPTIM_CNT 寄存器的结果是不同的（第一次读会复位 LPTIM_CNT）。
- 在 LPTIM 计数时钟选择 PCLK 时，连续访问 2 次也不能保证读出值可靠。

25.3.12.2. 同步复位

同步复位由 LPTIM_CR.COUNTRST 位控制。将 COUNTRST 位置 1 后，复位信号送给 LPTIM 的内核时钟域。所以需要注意在复位起作用前，LPTIM 内核时钟域的逻辑电路已过去几个时钟周期了。这将使从复位触发到复位生效，LPTIM 计数器多计了额外几个数。

由于 COUNTRST 是 APB 时钟域的，而 LPTIM 计数器位于 LPTIM 内核时钟域，所以当向 COUNTRST 位写入 1 时，需要内核时钟的 3 个时钟周期的延迟来同步来自 APB 时钟域的复位信号。

25.3.13. 编码器模式

该模式允许处理来自用于检测旋转元件角位置的正交编码器的信号。编码器接口模式简单地充当具有方向选择的外部时钟。这意味着计数器只是在 0 和 ARR 值之间连续计数（0 到 ARR 或 ARR 到 0，具体取决于方向）。因此必须在启动前配置 LPTIM_ARR。从两个外部输入信号 Input1 和 Input2 生成一个时钟信号来为 LPTIM 计数器提供时钟。这两个信号之间的相位决定了计数方向。

编码器模式仅在 LPTIM 由内部时钟源计数时可用。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟的 1/4。这是强制性要求以保证 LPTIM 的正常运行。

方向更改由 LPTIM_ISR 寄存器中的两个向下和向上标志指示。此外，如果启用 DOWNIE 位，则可以为两个方向更改事件生成中断。

要激活编码器模式，必须将 ENC 位设置为“1”。LPTIM 必须首先配置为连续模式。

当编码器模式处于活动状态时，LPTIM 计数器会根据增量编码器的速度和方向自动修改。因此，它的内容总是代表编码器的位置。由递增和递减标志指示的计数方向对应于编码器转子的旋转方向。

根据 CKPOL[1:0]位配置的边沿灵敏度，可能会出现不同的计数情况。下表总结了可能的组合，假设 Input1 和 Input2 不同时切换。

表 25-3 计数方式

有效沿	反向信号电平 (input1 for input2 input2 for input1)	input1 信号		input2 信号	
		上升沿	下降沿	上升沿	下降沿
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

下图显示了配置双沿灵敏度的编码器模式的计数序列。

在此模式下，预分频器分频比必须等于其复位值 1（PRESC[2:0]位必须为“000”）。

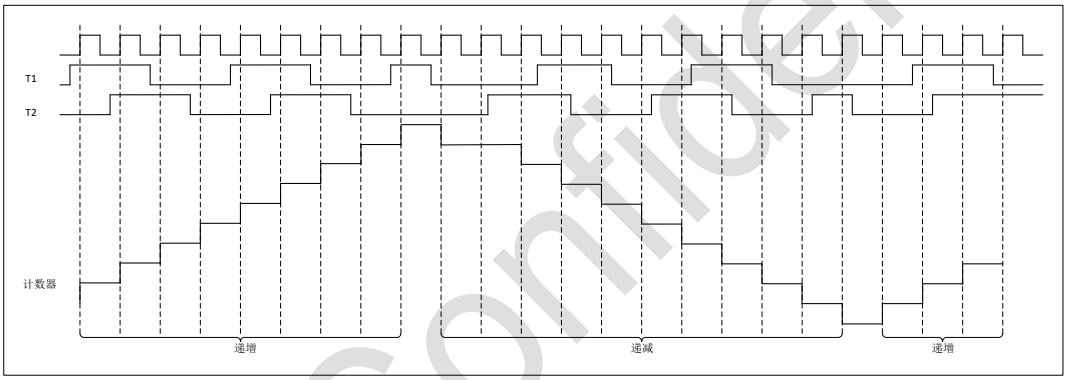


图 25-7 LPTIM 计数模式

25.3.14. 调试模式

当芯片进入调试模式，取决于 DBGMCU 模块的 DBG_LPTIMx_STOP 位的设定，LPTIM 或者继续正常工作，或者停止工作。

25.3.15. 编程指南

LPTIM 的初始化满足如下条件：

1.配置 RCC 模块的 LPTIM1EN/LPTIM2EN 后，紧接着编程 LPTIM_CFGR.TRIGEN，然后等待 5 个内核时钟后才能配置 LPTIM_CR 寄存器的 SNGSTRT 或者 CNTSTRT；

25.4. LPTIM 低功耗模式

表 25-4 低功耗模式对 LPTIM 的影响

低功耗模式	描述
睡眠	功能无影响. LPTIM 中断（使能后）会退出睡眠模式.
停止	LSE/LSI 时钟存在时功能无影响。 LPTIM 中断（使能后）会退出所有停止模式.
待机	LPTIM 外设断电，在退出待机模式后必须重新初始化。

注意：在使用 LPTIM 唤醒时，需要确保两次唤醒的间隔时间大于 6 个 PCLK 周期，否则将导致不可预知的结果。

25.5. LPTIM 中断

如果下列事件在 LPTIM_IER 寄存器内使能，则这些事件将生成中断/唤醒事件：

- 比较匹配（中断/唤醒）
- 自动重新加载匹配（如果是编码器模式，则无论方向如何）（中断/唤醒）
- 自动重载寄存器写入完成（中断）
- 比较寄存器写入完成（中断）
- 方向改变（编码器模式）（中断/唤醒）

注意:如果在 LPTIM_ISR 寄存器（状态寄存器）中的相应标志置 1 后，LPTIM_IER 寄存器（中断使能寄存器）中的相应位被置 1，则不产生中断。

表 25-5 LPTIM 中断事件

中断事件	描述
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，会产生中断标志。
自动重载匹配	当计数器寄存器的内容(LPTIM_CNT)与自动重新加载寄存器的内容匹配(LPTIM_ARR)，中断标志置位
自动重载寄存器写入完成	当对LPTIM_ARR寄存器的写操作完成，中断标志被置位
比较寄存器写入完成	当对 LPTIM_CMP 寄存器的写操作完成时，会产生中断标志。
方向改变	在编码器模式下使用。 嵌入了两个中断标志以指示方向改变： --UP 标志指示递增计数方向改变 --DOWN 标志表示减计数方向改变

25.6. LPTIM 寄存器

25.6.1. LPTIM 中断和状态寄存器（LPTIM_ISR)

地址偏移：0x000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWM	UP	ARROK	CMPOK	Res.	ARRM	CMPM
									R	R	R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWN	R	0	计数器由递增计数变为递减计数 在编码器模式下，DOWN 位由硬件设置用于通知应用程序计数器已从递增计数变为递减计数。可以通过向 LPTIM_ICR 寄存器中的 DOWNCF 位写入 1 来清除 DOWN 标志。

				注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UP	R	0	计数器由递减计数变为递增计数 在编码器模式下，UP 位由硬件设置用于通知应用程序计数器已从递减计数变为递增计数。可以通过向 LPTIM_ICR 寄存器中的 UPCF 位写入 1 来清除 UP 标志。注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROK	R	0	自动重载寄存器更新 OK。 ARROK 由硬件设置，以通知应用程序 APB 总线对 LPTIM_ARR 的写操作已成功完成。向 LPTIM_ICR.ARROKCF 写入 1 可清除 ARROK 标志。
3	CMPOK	R	0	比较寄存器更新 OK CMPOK 由硬件置位，用于通知应用程序 APB 总线对 LPTIM_CMP 寄存器的写操作已成功完成。向 LPTIM_ICR.CMPOKCF 写入 1 可清除 ARROK 标志。
2	Reserved	-	-	保留
1	ARRM	R	0	自动重载匹配 ARRM 由硬件设置，通知应用程序 LPTIM_CNT 寄存器值匹配 LPTIM_ARR 寄存器的值。向 LPTIM_ICR 寄存器的 ARRMCF 位写入 1 可清除 ARRM 标志。
0	CMPM	R	0	比较匹配 CMPM 位由硬件置位，用于通知应用程序 LPTIM_CNT 寄存器值达到 LPTIM_CMP 寄存器的值。向 LPTIM_ICR 寄存器的 CMPMCF 位写入 1 可清除 CMPM 标志。

25.6.2. LPTIM 中断清零寄存器（LPTIM_ICR)

地址偏移：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWMCF	UPCF	ARROKCF	CMPOKCF	Res.	ARRM CF	CMPMCF
									W	W	W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWNCF	W	0	计数方向改变为向下清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 DOWN 标志。 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UPCF	W	0	方向改变为向上清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 UP 标志。 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROKCF	W	0	自动重载寄存器更新 OK 清除标志。 向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARROK 标志。
3	CMPOKCF	W	0	比较寄存器更新 OK 清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 CMPOK 标志。
2	Reserved	-	-	保留
1	ARRMCF	W	0	自动重载匹配清除标志

				向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARRM 标志。
0	CMPMCF	W	0	比较匹配清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 CMPM 标志。

25.6.3. LPTIM 中断使能寄存器 (LPTIM_IER)

地址偏移：0x008

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWNIE	UPIE	ARROKIE	CMPOKIE	Res.	ARRMIE	CMPMIE
									RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWNIE	RW	0	计数器由递增计数变为递减计数中断使能 0：禁用 DOWN 中断 1：使能 DOWN 中断 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UPIE	RW	0	计数器由递减计数变为递增计数中断使能 0：禁用 UP 中断 1：使能 UP 中断 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROKIE	RW	0	自动重载寄存器更新 OK 中断使能。 0：禁用 ARROK 中断 1： 使能 ARROK 中断
3	CMPOKIE	RW	0	比较寄存器更新 OK 中断使能 0：禁用 CMPOK 中断 1：使能 CMPOK 中断
2	Reserved	-	-	保留
1	ARRMIE	RW	0	自动重载匹配中断使能 0：禁用 ARRM 中断 1： 使能 ARRM 中断
0	CMPMIE	RW	0	比较匹配中断使能 0：禁用 CMPM 中断 1：使能 CMPM 中断

25.6.4. LPTIM 配置寄存器 (LPTIM_CFGR)

地址偏移：0x00C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	Res.	PRELOAD	WAVPOL	WAVE	Res.	TRIGEN[1:0]		Res.
							RW		RW	RW	RW		RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIGSEL[2:0]			Res.	PRESC[2:0]			Res.	TRGFLT[1:0]		Res.	Res.	Res.	CKPOL[1:0]		Res.
RW	RW	RW		RW	RW	RW		RW	RW				RW	RW	

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	ENC	RW	0	编码器模式使能

				ENC 位控制编码器模式 0：禁用编码器模式 1：启用编码器模式 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
23	Reserved	-	-	保留
22	PRELOAD	RW	0	寄存器更新模式。 预加载位控制 LPTIM_ARR 寄存器更新模式。 0:每次 APB 总线写访问后更新寄存器； 1:寄存器在当前 LPTIM 计数周期结束时更新； 注：本寄存器仅针对 LPTIM_ARR 寄存器。
21	WAVPOL	RW	0	波形极性 WAVPOL 位控制输出极性 0：LPTIM 输出反映了 LPTIM_CNT 和 LPTIM_CMP 寄存器之间的比较结果 1：LPTIM 输出反映了 LPTIM_CNT 和 LPTIM_CMP 寄存器之间比较结果相反的值
20	WAVE	RW	0	波形形状. WAVE 位控制输出形状 0：停用一次设置模式，PWM 还是单脉冲波形具体取决于定时器的启动方式（用于 PWM 的 CNTSTRT 或用于单脉冲波形的 SNGSTRT）。 1：激活一次设置模式
19	Reserved	-	-	保留
18:17	TRIGEN[1:0]	RW	0	触发使能和极性 TRIGEN 位控制 LPTIM 计数器是否由外部触发启动。如果选择了外部触发选项，则触发有效边沿可以采用 2 种配置： 00：软件触发（计数启动由软件发起） 01：上升沿为有效沿 10：下降沿为有效沿 11：保留
16	Reserved	-	-	保留
15:13	TRIGSEL[2:0]	RW	0	触发选择器 TRIGSEL 位在以下 8 个可用源中选择将作为 LPTIM 触发事件的触发源： 000：lptim_ext_trig0（LPTIM1_ETR for LPTIM1，LPTIM2_ETR FOR LPTIM2） 001：lptim_ext_trig1（RTC 闹钟 A） 010：lptim_ext_trig2（RTC 闹钟 B） 011：lptim_ext_trig3（TAMP 事件） 100：lptim_ext_trig4（COMP1_OUT） 101：lptim_ext_trig5（COMP2_OUT） 110：lptim_ext_trig6（reserved） 111：lptim_ext_trig7（reserved）
12	Reserved	-	-	保留
11:9	PRESC[2:0]	RW	0	时钟预分频器。

				PRESC 位配置预分频器分频系数。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	Reserved	-	-	保留
7:6	TRGFLT[1:0]	RW		触发的可配置数字滤波器 TRGFLT 值设置当内部触发器上发生电平变化时应检测到的连续相等采样值的数量，然后才将其视为有效电平转换。必须存在内部时钟源才能使用此功能。 00：任何触发活动电平变化都被视为有效触发 01：触发有效电平变化必须稳定至少 2 个时钟周期才被认为是有效触发。 10：触发有效电平变化必须稳定至少 4 个时钟周期才被认为是有效触发。 11：触发有效电平变化必须稳定至少 8 个时钟周期才被认为是有效触发。
5:3	Reserved	-	-	保留
2:1	CKPOL[1:0]	RW	0	处于活动状态编码器模式选择。 00：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 1 处于活动状态。 01：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 2 处于活动状态。 10：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 3 处于活动状态。 11：保留 注意：如果 LPTIM 不支持编码器模式功能，则该位域保留。
0	Reserved	-	-	保留

注意：配置 CFGR 寄存器时建议一次写所有配置位，若分开配置时，两次配置之间需要间隔 3 个内核时钟。

25.6.5. LPTIM 控制寄存器 (LPTIM_CR)

地址偏移：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	COUNTRST	CNTSTRT	SNGSTRT	ENABLE
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	RSTARE	RW	0	读取后复位使能 此位由软件置 1 和清 0。当 RSTARE 设置为“1”时，对 LPTIM_CNT 的任何读取访问寄存器将异步重置 LPTIM_CNT 寄存器内容。

3	COUNTRST	RW	0	计数器复位。 该位由软件置 1，硬件清 0。设置为“1”时，此位将触发 LPTIM_CNT 计数寄存器同步复位。由于此复位的同步特性，它只需要在同步延迟 3 个 LPTIM 内核时钟周期之后释放（LPTIM 内核时钟可能是与 APB 时钟不同）。 注：在 COUNTRST 已被硬件清除为“0”之前，软件绝不能将其设置为“1”.因此，软件在尝试将其设置为“1”之前，应检查 COUNTRST 位是否已清零为“0”
2	CNTSTRT	RW	0	定时器启动连续模式。 该位由软件置位。 如果在进行单次计数模式计数时该位被置 1，则定时器不会在下一个 LPTIM_ARR 和 LPTIM_CNT 寄存器匹配的脉冲模式计数时停止。LPTIM 计数器保持在连续模式下计数。 注：仅当 LPTIM 使能时，此位才能置 1。该位在同步到内核时钟后由硬件清零。
1	SNGSTRT	RW	0	LPTIM 启动单次模式。 该位由软件置位，由硬件清零。该位置 1 将以单脉冲模式启动 LPTIM。 注：仅当 LPTIM 使能时，此位才能置 1。它将由硬件自动复位。
0	ENABLE	RW	0	LPTIM 使能位,由软件设置和清零 0:LPTIM 禁用 1:LPTIM 使能

25.6.6. LPTIM 比较寄存器 (LPTIM_CMP)

地址偏移：0x014

复位值：0x0000 0000

LPTIM1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CMP	RW	0x0000	比较值。 CMP 是 LPTIM 的比较值 当 LPTIM 使能后才能更新该寄存器。

LPTIM2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CMP	RW	0x0000	比较值。 CMP 是 LPTIM 的比较值 当 LPTIM 使能后才能更新该寄存器。

25.6.7. LPTIM 自动重载寄存器 (LPTIM_ARR)

地址偏移：0x018

复位值：0x0000 0001

LPTIM1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'h1	自动重新加载值 ARR 是 LPTIM 的自动重载值。 当 LPTIM 使能后才能更新该寄存器。

LPTIM2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'h1	自动重新加载值 ARR 是 LPTIM 的自动重载值。 当 LPTIM 使能后才能更新该寄存器。

25.6.8. LPTIM 计数器寄存器 (LPTIM_CNT)

地址偏移：0x01C

复位值： 0x0000 0000

LPTIM1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	R	0x0000	计数器值 当 LPTIM 以异步时钟运行时，读取 LPTIM_CNT 寄存器可能返回不可靠的值。因此在这种情况下，有必要执行两次连续的读访问并验证返回的两个值是否相同。需要注意的是，对于可靠的 LPTIM_CNT 寄存器读访问，需要两次连续的读操作并比较，相等时，可以认为读取访问是可靠的。

LPTIM2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function

31:16	Reserved	-	-	保留
15:0	CNT[15:0]	R	0	计数器值 当 LPTIM 以异步时钟运行时，读取 LPTIM_CNT 寄存器可能返回不可靠的值。因此在这种情况下，有必要执行两次连续的读访问并验证返回的两个值是否相同。需要注意的是，对于可靠的 LPTIM_CNT 寄存器读访问，需要两次连续的读操作并比较，相等时，可以认为读取访问是可靠的。

25.6.9. LPTIM 选择寄存器 (LPTIM_OR)

地址偏移：0x01C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN1[2:1]		IN1[2:1]		IN2[0]	IN1[0]
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	ResetValue	Function
31:16	Reserved	-	-	保留
5:4	IN2[2:1]	RW	0	IN2SEL[1:0]：LPTIM 输入 2 选择 IN2SEL 位控制 LPTIM 输入 2 多路复用器，它将 LPTIM 输入 2 连接到以下可用输入之一。 00：COMP1 01：COMP2 其它：reserved
3:2	IN1[2:1]	RW	0	IN2SEL[1:0]：LPTIM 输入 1 选择 IN2SEL 位控制 LPTIM 输入 1 多路复用器，它将 LPTIM 输入 1 连接到以下可用输入之一。 00：COMP1 01：COMP2 其它: reserved
1	IN2[0]	RW	0	LPTIM 输入 2 重映射 1：连接到 IN2[2:1]指示的 COMP 输出 0：连接到 GPIO
0	IN1[0]	RW	0	LPTIM 输入 1 重映射 1：连接到 IN1[2:1]指示的 COMP 输出 0：连接到 GPIO

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

26. 独立看门狗（IWDG）

26.1. IWDG 简介

芯片内集成了一个独立看门狗（简称 IWDG），该模块具有高安全级别、时序精确及灵活使用的特点。IWDG 发现并解决由于软件失效造成的功能混乱，并在计数器达到指定的超时值时触发系统复位。

IWDG 由 LSI/LSE 提供时钟，这样即使主时钟发生故障时，也能保持工作。

IWDG 最适合应用于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应用。

26.2. IWDG 主要特性

- 自由运行递减计数器
- 由 LSI 或者 LSE 提供时钟（在停止或者待机模式也可以工作）
- 支持硬件模式
- 可配置在停止模式下停止计数
- 当递减计数器值为 0x000 时产生复位

26.3. IWDG 功能描述

26.3.1. IWDG 框图

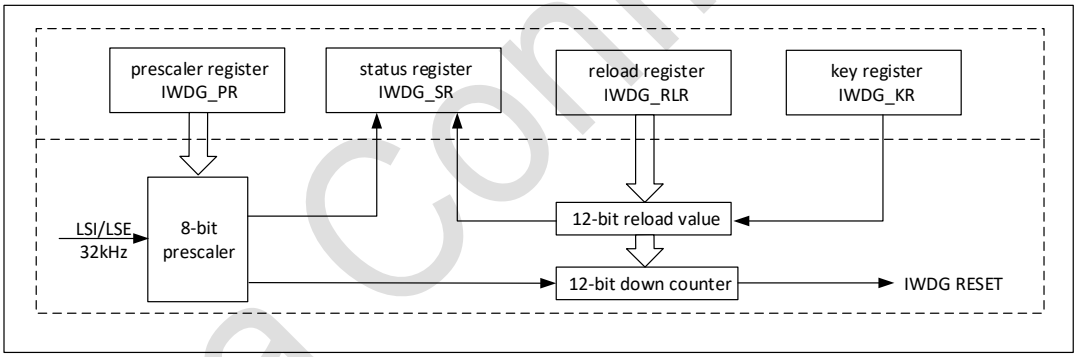


图 26-1 独立看门狗框图

当通过向 IWDG 密钥寄存器(IWDG_KR)写 0x0000 CCCC，计数器开始从 0xFFF 递减计数。当到达计数最终值时（0x000），产生一个复位信号（IWDG 复位）。

不管何时，0x0000 AAAA 被写入 IWDG 密钥寄存器时，IWDG_RLR（重载寄存器）的值被再次装载到计数器中，IWDG 不会产生复位。

硬件看门狗

如果上电加载的用户选项字节设置了使能硬件看门狗，则 IWDG 上电被自动使能（同时使能 LSI 时钟作为 IWDG 的计数时钟），并且如果在计数器计数到终值之前，IWDG 密钥寄存器没被软件改写，则产生复位信号。

26.3.2. 寄存器访问保护

对寄存器 IWDG_PR、IWDG_RLR 的写访问是被保护的。为了修改他们，用户必须先向 IWDG_KR 寄存器写 0x0000 5555。对这些寄存器的写其他数将破坏时序，如写 0x0000AAAA 加载，寄存器将被再次保护。

状态寄存器指示 IWDG_PR、IWDG_RLR 寄存器的值是否正在更新。

26.3.3. 调试模式

如果 CPU 进入调试模式，IWDG 继续计数还是停止，取决于 DBGMCU 模块中 DBG_IWDG_STOP 的配置。

26.3.4. 停止模式

在 Flash 信息区的选项字节中有 IWDG_STOP 位，可以控制在系统进入停止模式时 IWDG 继续正常计数还是停止计数。

26.4. IWDG 寄存器

26.4.1. IWDG 密钥寄存器 (IWDG_KR)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY[15:0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	KEY[15:0]	W	0x00	Key 值。 软件必须以一定的时间间隔向该寄存器写入 0xAAAA，否则，当计数器计数到 0 时，看门狗会产生复位 0x5555：表示允许访问 IWDG_PR、IWDG_RLR 寄存器 0xCCCC：表示启动 IWDG（如果选择了硬件看门狗则不受此命令字限制）。

26.4.2. IWDG 预分频寄存器 (IWDG_PR)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[2:0]		
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:0	PR[2:0]	RW	0	预分频值。 通过配置该寄存器选择计数器时钟的预分频值。 要改变该寄存器，IWDG_SR 寄存器的 PVU 必须为 0。 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

26.4.3. IWDG 重载寄存器 (IWDG_RLR)

偏移地址:0x08

复位值:0x0000 0FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	RL[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	RL[11:0]	RW	0xFFF	IWDG 计数器重装载值。 当向 IWDG_KR 寄存器写入 0xAAAA 时，RL 值会传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此 RL 值和时钟预分频值来计算。 只有当 IWDG_SR.RVU=0 时，才能对寄存器进行修改。

26.4.4. IWDG 状态寄存器 (IWDG_SR)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
														R	R

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	RVU	R	0	看门狗计数器重装值更新。 该位由硬件置 1，表明重装载值正在更新。当重装载值更新结束后，此位由硬件清零。
0	PVU	R	0	看门狗预分频值更新。 该位由硬件置 1，表明预分频值正在更新。当预分频值更新结束后，此位由硬件清零。

注：在更新 IWDG_PR 和 IWDG_SR.RLR 前，要分别等待 IWDG_PVU 和 IWDG_SR.RVU 为 0。但在更新 IWDG_PR、IWDG_RLR 后，不必再等待 IWDG_SR.PVU、IWDG_SR.RVU 为 0，可继续执行下面的代码。

27. 系统窗口看门狗（WWDG）

27.1. WWDG 简介

系统窗口看门狗(WWDG) 通常被用来监测由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。除非程序在 T6 位变成 0 前刷新递减计数器的值，否则看门狗电路在达到预置的时间周期时，会产生一个系统复位。如果在递减计数器达到窗口寄存器值之前刷新控制寄存器中的 7 位递减计数器值，也会产生系统复位。这意味着必须在限定的时间窗口内刷新计数器。

WWDG 时钟由 APB 时钟经预分频后提供，通过可配置的时间窗口来检测应用程序非正常的过迟或过早的操作。

WWDG 最适合那些要求看门狗在精确计时窗口起作用的应用程序。

27.2. WWDG 主要特性

- 支持硬件模式
- 可编程的自由运行递减计数器
- 复位条件
  - 当递减计数器低于 0x40 时（如果看门狗已激活）
  - 在窗口之外重载递减计数器时复位（如果看门狗已激活）
- 提前唤醒中断（EWI）：当递减计数器等于 0x40 时被触发（如果该功能使能，且看门狗被激活）

27.3. WWDG 功能描述

如果 WWDG 被激活（WWDG_CR.WDGA 被置位），并且当 7 位递减计数器（T[6:0]位）从 0x40 减小到 0x3F（T6 被清掉），则引起复位。如果软件再装载计数器时，计数器值大于存储器窗口寄存器的值，产生复位。

应用程序必须在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止复位。仅当计数器值小于窗口寄存器的值并且高于 0x3F 时，才能执行上述写操作。要存储到 WWDG_CR 寄存器的值必须介于 0xFF 和 0xC0 之间。

27.3.1. WWDG 框图

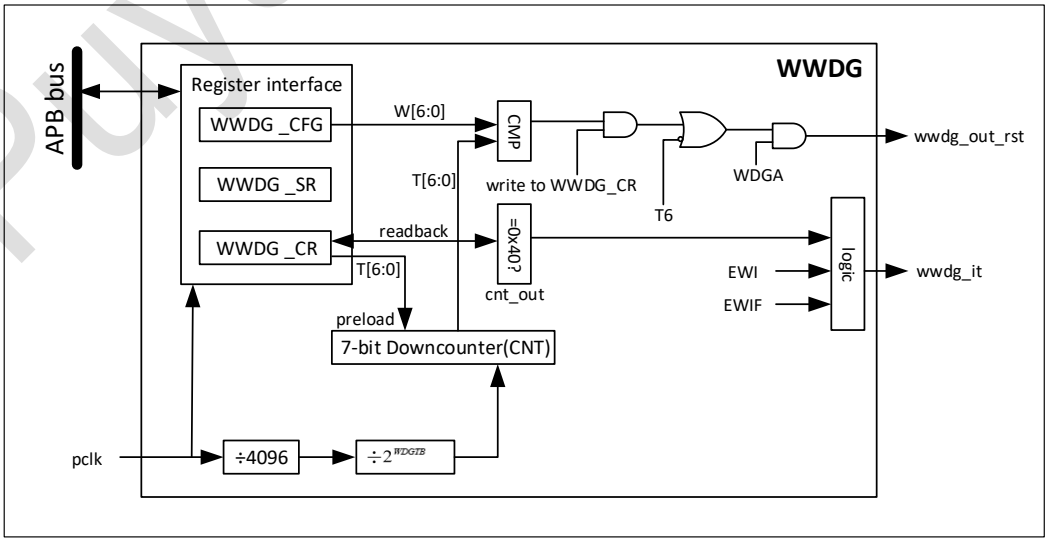


图 27-1 窗口看门狗框图

27.3.2. 使能看门狗

当用户选项字节中 WWDG_SW 位选择“软件窗口看门狗”，看门狗在复位后总处于关闭状态。然后通过设置 WWDG_CR 寄存器的 WDGA 位，WWDG 模块被使能，然后除非复位发生，否则不能再次关闭。

当用户选项字节中 WWDG_SW 选择“硬件 WWDG”，则看门狗在复位后总处于使能，无法禁止。

27.3.3. 控制递减计数

递减计数器处于自由运行状态，即使禁止看门狗，递减计数器仍继续递减计数。当使能看门狗时，必须将 T6 位置 1，以防止立即复位。

T[5:0]位包含 WWDG 产生复位之前增加的计数值，该数据表示了在看门狗产生复位之前的时间延迟。

配置寄存器(WWDG_CFR) 中包含窗口的上限值：要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载。

27.3.4. 看门狗中断高级特性

如果在产生实际复位之前必须执行特定的安全操作或数据记录，则可使用提前唤醒中断(EWI)。通过设置 WWDG_CFR 寄存器中的 EWI 位使能 EWI 中断。当递减计数器的值为 0x40 时，将生成 EWI 中断。在复位器件之前，可以使用相应的中断服务程序 (ISR) 来触发特定操作（例如通信或数据记录）。

在某些应用中，可以使用 EWI 中断来管理软件系统检查和/或系统恢复/功能退化，而不会生成 WWDG 复位。在这种情况下，相应的中断服务程序 (ISR) 可用来重载 WWDG 计数器以避免 WWDG 复位，然后再触发所需操作。

通过将 0 写入 WWDG_SR 寄存器中的 EWIF 位来清除 EWI 中断。

注意：当由于在更高优先级任务中有系统锁定而无法使用 EWI 中断时，最终会产生 WWDG 复位。

27.3.5. 如何设置看门狗超时

对 WWDG_CR 寄存器通常通过向 T6 位写 1，以避免产生复位。

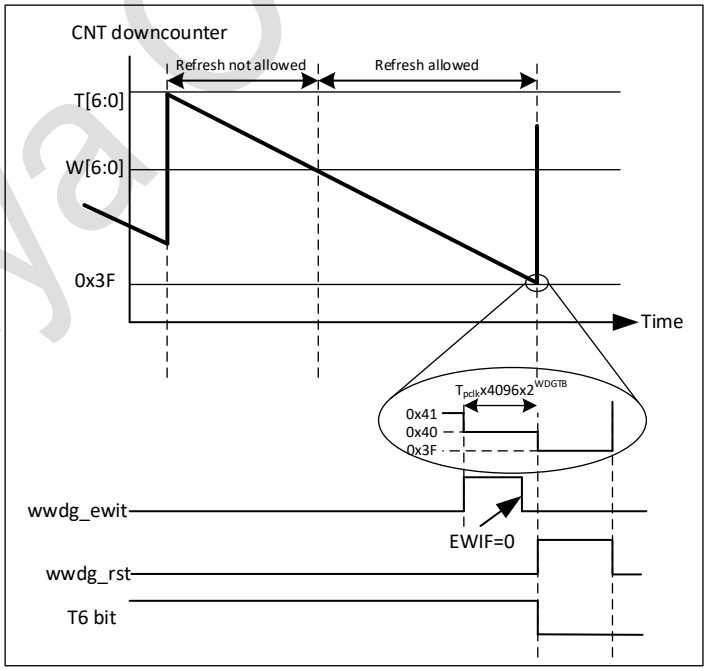


图 27-2 窗口看门狗时序图

计算 WWDG 超时值的公式如下：

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

其中：

t_{WWDG}: WWDG 超时

t_{PCLK}: APB 时钟周期，以 ms 为测量单位

4096：对应于内部分频器的值

例如，假设 APB 频率等于 72 MHz，将 WDGTB[1:0] 设置为 3 并将 T[5:0] 设置为 63：

$$t_{WWDG} = (1/72000) \times 4096 \times 2^3 \times (63 + 1) = 29.127ms.$$

27.3.6. 调试模式

如果 CPU 进入调试模式，WWDG 继续计数还是停止，取决于 DBGMCU 模块中 DBG_WWDG_STOP 的配置。

27.4. WWDG 寄存器

27.4.1. WWDG 控制寄存器 (WWDG_CR)

偏移地址:0x00

复位值:0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T[6:0]						
								RS	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	WDGA	RS	0	激活位 (Activation bit) 此位由软件置'1'，只有在复位后由硬件清'0'。当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗 1：使能看门狗
6:0	T[6:0]	RW	7'h7F	7 位计数器（MSB 至 LSB）。 该寄存器用来存储看门狗的计数值。每（4096x2^{WDGTB}）个 PCLK 周期减 1.当计数值从 0x40 变为 0x3F 时（T[6]变为 0），产生看门狗复位。

27.4.2. WWDG 配置寄存器 (WWDG_CFR)

偏移地址:0x04

复位值:0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB[1:0]		W[6:0]						
						RS	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	EWI	RS	0	提前唤醒中断。

				该位置 1，则当计数器值达到 0x40 时，即产生中断。 此中断由硬件在复位后清除。
8:7	WDGTB[1:0]	RW	2’b0	定时器时基 预分频器的时基设置如下： 00：CK 计数器时钟（PCLK 除以 4096）除以 1 01：CK 计数器时钟（PCLK 除以 4096）除以 2 10：CK 计数器时钟（PCLK 除以 4096）除以 4 11：CK 计数器时钟（PCLK 除以 4096）除以 8
6:0	W[6:0]	RW	7’h7F	7 位窗口值。 该寄存器包含了用来与递减计数器进行比较用的窗口值。

27.4.3. WWDG 状态寄存器 (WWDG_SR)

偏移地址:0x08

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	EWIF	RC_W0	0	提前唤醒中断标志。 当计数器值达到 0x40，此位由硬件置 1. 软件写 0 清零，写 1 无效。 当中断未被使能时，该位也会置 1.

28. 实时时钟 (RTC)

28.1. RTC 简介

实时时钟 (RTC) 提供用于管理所有低功耗模式的自动唤醒单元。

实时时钟 (RTC) 是一个独立的 BCD 定时器/计数器。RTC 提供具有可编程闹钟中断功能的日历时钟/日历。

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。

RTC 可在 V_{BAT} 模式下工作。

28.2. RTC 主要特性

- 日历具有亚秒、秒、分、小时（12 或 24 格式）、星期几、日、月、年，格式为 BCD（二进码十进数）
- 自动调整每月是 28、 29（闰年）、 30 还是 31 天
- 两个可编程闹钟。
- 可运行时纠正 1 到 32767 个 RTC 时钟脉冲。这可用于与主时钟同步。
- 参考时钟检测：可使用更加精确的第二时钟源（50 或 60 Hz）来提高日历的精确度。
- 数字校准电路具有 0.95 ppm 的分辨率，以补偿石英晶振的误差。
- 时间戳特性可用于保存日历内容。此功能可由时间戳引脚上的事件触发，或由入侵事件触发，也可由切换到 V_{BAT} 模式的事件触发。
- 用于周期性事件的 17 位自动重载唤醒定时器 (WUT)，具有可编程分辨率和周期。
- 可以选择以下三种 RTC 的时钟源：
  - HSE 时钟除以 32
  - LSE 时钟
  - LSI 时钟

  由 LSE 或者 LSI 提供时钟时， RTC 可在 V_{BAT} 模式下和所有低功耗模式下工作。
- 3 个专门的可屏蔽中断：
  - 闹钟中断
  - 时间戳中断
  - 唤醒定时器中断

28.3. RTC 功能描述

28.3.1. RTC 框图

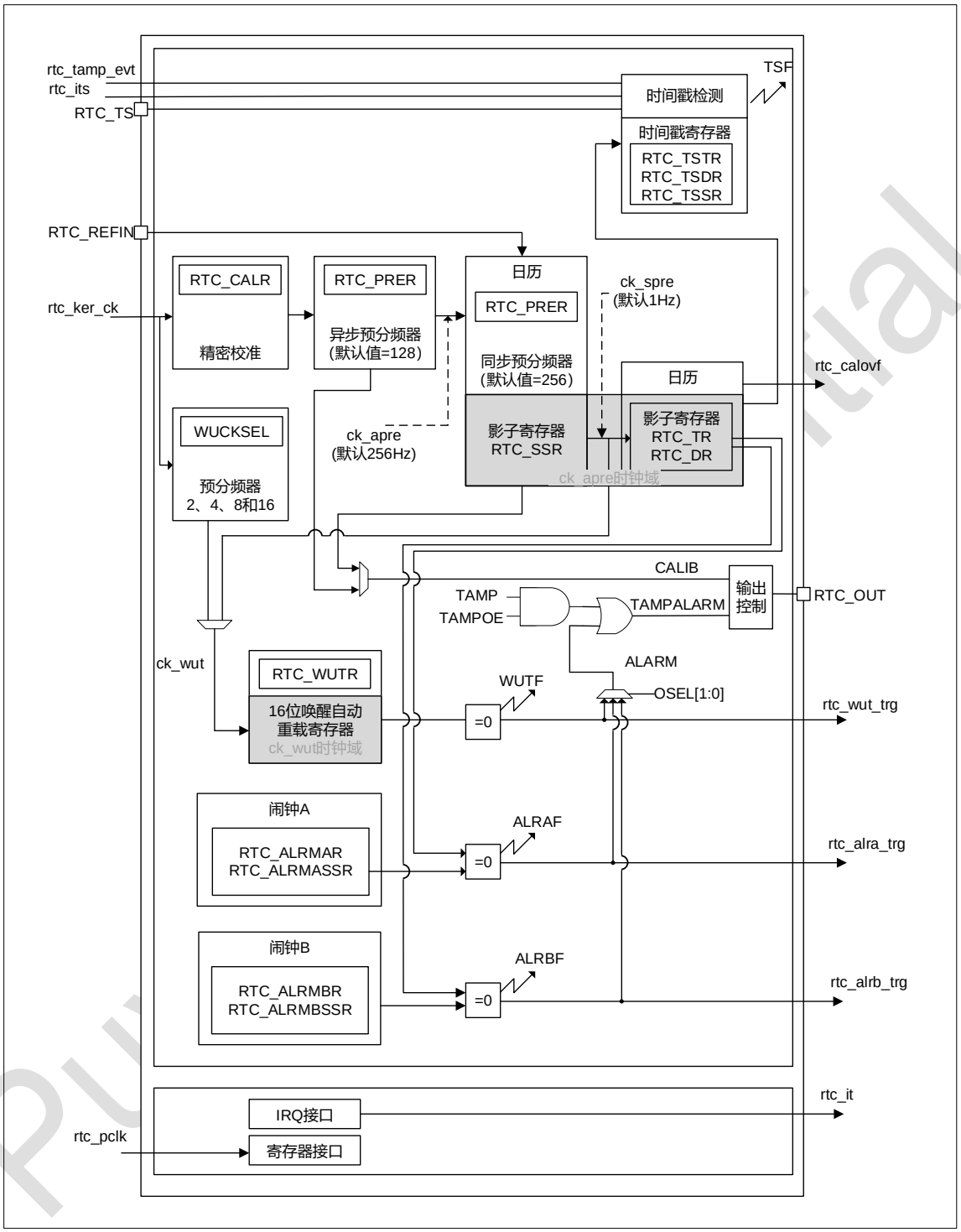


图 28-1 RTC 框图

28.3.2. RTC 引脚和内部信号

表 28-1 RTC 输入/输出引脚

名称	输入/输出	描述
RTC_TS	输入	RTC 时间戳输入
RTC_REFIN	输入	RTC 50 或 60 Hz 参考时钟输入
RTC_OUT	输出	RTC 输出，可选择如下输出之一： --CALIB：512 Hz 或 1 Hz 时钟输出（LSE 频率为 32.768 kHz）。可通过将 RTC_CR 寄存器中的 COE 位置 1 来使能此输出。 --TAMPALRM：此输出是 TAMP 和 ALARM 输出的逻辑或运算结果。可通过配置 RTC_CR 寄存器中的 OSEL[1:0] 位使能 ALARM，可选择闹钟 A、闹钟 B 或唤醒输出。可通过将 RTC_CR 寄存器中的 TAMPOE 位置 1 使能 TAMP，以选择入侵事件输出。

表 28-2 RTC 内部输入/输出信号

名称	输入/输出	描述
rtc_ker_ck	输入	RTC 内核时钟，在本文档中也称为 RTCCLK。
rtc_pclk	输入	RTC APB 时钟
rtc_its	输入	RTC 内部时间戳事件
rtc_tamp_evt	输入	在 TAMP 外设中检测到的入侵事件（内部或外部）
rtc_it	输出	RTC 中断
rtc_alra_trg	输出	RTC 闹钟 A 事件检测触发
rtc_alrb_trg	输出	RTC 闹钟 B 事件检测触发
rtc_wut_trg	输出	RTC 唤醒定时器事件检测触发
rtc_calovf	输出	RTC 日历溢出

28.3.3. RTC 和 TAMP 控制的 GPIO

VDDK 域 IO PA[8]包含 RTC 和 TAMP 功能，这部分功能由 RTC 和 TAMP 直接控制，而不管 GPIO AF 功能配置如何。

RTC_OUT、RTC_TS 和 TAMP_IN 映射到同一引脚 (PA8)。映射在 PA8 上的 RTC 和 TAMP 功能可用于所有低功耗模式和 V_{BAT} 模式。

输出机制遵循下表中所示的优先级顺序。

PA8 引脚功能		OSEL[1:0]（ALARM 输出使能）	TAMPOE（TAMPER 输出使能）	COE 位（CALIB 输出使能）	TAMPALRM_TYPE	TAMPALRM_PU	TAMPE（TAMP_IN 输入使能）	TSE（RTC_TS 输入使能）
TAMPALRM 推挽输出		01 或 10 或 11	0	无关	0	0	无关	无关
		00	1					
		01 或 10 或 11	1					
TAMPALRM 开漏输出	没有上下拉	01 或 10 或 11	0	无关	1	0	无关	无关
		00	1					
		01 或 10 或 11	1					

	内部上拉	01 或 10 或 11	0	无关	1	1	无关	无关
		00	1					
		01 或 10 或 11	1					
CALIB 输出推挽		00	0	1	无关	无关	无关	无关
TAMP_IN 输入浮空		00	0	0	无关	无关	1	0
RTC_TS 和 TAMP_IN 输入浮空		00	0	0	无关	无关	1	1
RTC_TS 输入浮空		00	0	0	无关	无关	0	1
唤醒引脚或标准 GPIO		00	0	0	无关	无关	0	0

28.3.4. RTC 时钟和预分频器

通过配置 RCC_BDCR.RTCSEL 寄存器，RTC 时钟源 (RTCCLK) 通过可以从 LSE 时钟、LSI 时钟以及 HSE 时钟三者中选择。

可编程的预分频器阶段可生成 1 Hz 的时钟，用于更新日历。为最大程度地降低功耗，预分频器分为 2 个可编程的预分频器：

- 一个通过 RTC_PRER 寄存器的 PREDIV_A 位配置的 7 位异步预分频器。
- 一个通过 RTC_PRER 寄存器的 PREDIV_S 位配置的 15 位同步预分频器。

注意： 使用两个预分频器时，推荐将异步预分频器配置为较高的值，以最大程度降低功耗。

要使用频率为 32.768 kHz 的 LSE 获得频率为 1 Hz 的内部时钟 (ck_spre)，需要将异步预分频系数设置为 128，并将同步预分频系数设置为 256。

分频系数的最小值为 1，最大值为 2^{22}。

这对应于约为 4 MHz 的最大输入频率。

f_{ck_apre} 可根据以下公式得出：

$$f_{CK\_APRE} = \frac{f_{RTCCLK}}{PREDIV\_A + 1}$$

ck_apre 时钟用于为二进制 RTC_SSR 亚秒递减计数器提供时钟。当该计数器计数到 0 时，会使用 PREDIV_S 的内容重载 RTC_SSR。

f_{ck_spre} 可根据以下公式得出：

$$f_{CK\_SPRE} = \frac{f_{RTCCLK}}{(PREDIV\_S + 1)x(PREDIV\_A + 1)}$$

ck_spre 时钟既可以用于更新日历，也可以用作 16 位唤醒自动重载定时器的时基。为获得较短的超时周期，还可以将 16 位唤醒自动重载定时器与经可编程的 4 位异步预分频器分频的 RTCCLK 一同运行。

28.3.5. 实时时钟和日历

RTC 日历时间和日期寄存器可通过与 PCLK（APB 时钟）同步的影子寄存器来访问。这些时间和日期寄存器也可以直接访问，这样可避免等待同步的持续时间。

- RTC_SSR 对应于亚秒
- RTC_TR 对应于时间

- RTC_DR 对应于日期

每隔一个 RTCCLK 周期，当前日历值将被复制到影子寄存器，同时 RTC_ICSR 寄存器的 RSF 位被置 1。在停机和待机模式下不会执行复制操作。退出这两种模式时，影子寄存器会在最长 4 个 RTCCLK 周期后进行更新。

当应用读取日历寄存器时，它会访问影子寄存器的内容。也可以通过将 RTC_CR 寄存器的 BYPSHAD 控制位置 1 来直接访问日历寄存器。默认情况下，该位被清零，用户访问影子寄存器。

在 BYPSHAD=0 模式下读取 RTC_SSR、 RTC_TR 或 RTC_DR 寄存器时， APB 时钟频率(f_{APB})必须至少为 RTC 时钟频率 (f_{RTCCLK}) 的 7 倍。

影子寄存器通过系统复位来复位。

28.3.6. 可编程闹钟

RTC 单元提供两个可编程闹钟，即闹钟 A 和闹钟 B。以下说明针对闹钟 A，但同样适用于闹钟 B。

可通过 RTC_CR 寄存器中的 ALRAE 位来使能可编程闹钟功能。

如果日历亚秒、秒、分钟、小时、日期或日与闹钟寄存器 RTC_ALRMASSR 和 RTC_ALRMAR 中编程的值相匹配，则 ALRAF 标志会被置为 1。可通过 RTC_ALRMAR 寄存器的 MSKx 位以及 RTC_ALRMASSR 寄存器的 MASKSSx 位单独选择各日历字段。

可通过 RTC_CR 寄存器中的 ALRAIE 位来使能闹钟中断。

注意： 如果选择秒字段（RTC_ALRMAR 中的 MSK1 位复位），则 RTC_PRER 寄存器中设置的同步预分频器分频系数必须至少为 3，才能确保闹钟正确地运行。

闹钟 A 和闹钟 B（如果已通过 RTC_CR 寄存器中的位 OSEL[0:1] 使能）可连接到 TAMPALRM 输出。可通过 RTC_CR 寄存器的 POL 位配置 TAMPALRM 输出极性。

28.3.7. 周期性自动唤醒

周期性唤醒标志由 16 位可编程自动重载递减计数器产生。唤醒定时器范围可扩展至 17 位。

可通过 RTC_CR 寄存器中的 WUTE 位来使能此唤醒功能。

唤醒定时器的时钟输入 ck_wut 可以是：

- 2、4、8 或 16 分频的 RTC 时钟 (RTCCLK)。

当 RTCCLK 为 LSE (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。

- ck_spre（通常为 1 Hz 内部时钟）

当 ck_spre 频率为 1 Hz 时，可得到的唤醒时间为 1s 到 36h 左右，分辨率为 1 秒。这一较大的可编程时间范围分为两部分：

– WUCKSEL [2:1] = 10 时为 1s 到 18h

– WUCKSEL [2:1] = 11 时约为 18h 到 36h。在后一种情况下，会将 216 添加到 16 位计数器的当前值。当初始化序列完成之后，定时器开始递减计数。在低功耗模式下使能唤醒功能时，递减计数保持有效。此外，当定时器达到 0 时， RTC_SR 寄存器的 WUTF 标志会置 1，并且唤醒计数器会使用其重载值（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_CR 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。

如果已通过 RTC_CR 寄存器的位 OSEL[1:0] 使能周期性唤醒标志，则该标志可连接到 TAMPALRM 输出。可通过 RTC_CR 寄存器的 POL 位配置 TAMPALRM 输出极性。

系统复位以及低功耗模式（睡眠、停机和待机）对唤醒定时器没有任何影响。

28.3.8. RTC 初始化和配置

28.3.8.1. RTC 寄存器访问

RTC 寄存器为 32 位寄存器。除了当 BYPSHAD=0 时对日历影子寄存器执行的读访问之外，APB 接口会在访问 RTC 寄存器时引入 2 个等待周期。

28.3.8.2. RTC 寄存器写保护

系统复位后，可通过电源控制外设中的 DBP 位来保护 RTC 寄存器以防止非正常的写访问。必须将 DBP 位置 1 才能使能 RTC 寄存器的写访问。

RTC 域复位后，不是所有的 RTC 寄存器均受到写保护。

通过向写保护寄存器 RTC_WPR 写入一个密钥来使能对受保护 RTC 寄存器的写操作。

要解锁受保护 RTC 寄存器的写保护，需要执行以下步骤：

- 将 0xCA 写入 RTC_WPR 寄存器。
- 将 0x53 写入 RTC_WPR 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。

28.3.8.3. 日历初始化和配置

要编程包括时间格式和预分频器配置在内的初始时间和日期日历值，需按照以下顺序操作：

1. 将 RTC_ICSR 寄存器中的 INIT 位置为 1， RTC 进入初始化模式。在此模式下，日历计数器将停止工作并且其值可更新。

2. 轮询 RTC_ICSR 寄存器中的 INITF 位。当 INITF 置 1 时进入初始化模式。大约需要 3 个 RTCCLK 时钟周期（由于时钟同步）。

3. 编程 RTC_PRER 寄存器的两个预分频系数，为日历计数器产生 1 Hz 的时钟。

4. 在影子寄存器（RTC_TR 和 RTC_DR）中加载初始时间和日期值，然后通过 RTC_CR 寄存器中的 FMT 位配置时间格式（12 或 24 小时制）。

5. 通过清零 INIT 位退出初始化模式。随后，自动加载实际日历计数器值，在 4 个 RTCCLK 时钟周期后重新开始计数。

当初始化序列完成之后，日历开始计数。

注意：系统复位后，应用可读取 RTC_ICSR 寄存器中的 INITS 标志，以检查日历是否已初始化。如果该标志为 0，表明自系统复位以来，日历还尚未初始化过，其年份字段一直还保持着 RTC 域复位默认值 (0x00)。

要在初始化之后读取日历，必须首先用软件检查 RTC_ICSR 寄存器的 RSF 标志是否置 1。

28.3.8.4. 夏令时

可通过 RTC_CR 寄存器的 SUB1H、 ADD1H 和 BKP 位管理夏令时。

利用 SUB1H 或 ADD1H，软件只需单次操作便可在日历中减去或增加一个小时，无需执行整个初始化步骤。

此外，软件还可以使用 BKP 位来记录是否曾经执行过此操作。

28.3.8.5. 编程闹钟

要对可编程的闹钟进行编程或更新，必须执行类似的步骤。以下步骤针对闹钟 A，但同样适用于闹钟 B。

1. 将 RTC_CR 中的 ALRAE 位清零以禁止闹钟 A。
2. 编程闹钟 A 寄存器 (RTC_ALRMASSR/RTC_ALRMAR)。
3. 将 RTC_CR 寄存器中的 ALRAE 位置 1 以再次使能闹钟 A。

注意：由于时钟同步缘故，RTC_CR 寄存器的每次更改需要在大约 3 个 RTCCLK 时钟周期后执行。

28.3.8.6. 编程唤醒定时器

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_ICSR 中的 WUTWF，直到此位置 1，以确保可以访问唤醒自动重载计数器和 WUCKSEL[2:0] 位。在日历初始化模式下，必须跳过此步骤。大约需要 3 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计数。由于时钟同步的缘故，在 WUTE 清零后，WUTWF 位也会清零，但需要花费多达 3 个 RTCCLK 时钟周期。

28.3.9. 读取日历

当 RTC_CR 寄存器中的 BYPSHAD 控制位清零时要正确读取 RTC 日历寄存器（RTC_SSR、RTC_TR 和 RTC_DR），APB 时钟频率 (f_{PCLK})必须等于或大于 RTC 时钟频率 (f_{RTCCLK}) 的七倍。这可以确保同步机制的安全性。

如果 APB 时钟频率低于 RTC 时钟频率的七倍，则软件必须读取日历时间寄存器和日期寄存器两次。这样，当两次读取的 RTC_TR 结果相同时，才能确保数据正确。否则必须执行第三次读访问。任何情况下，APB1 的时钟频率都不能低于 RTC 的时钟频率。

每次将日历寄存器中的值复制到 RTC_SSR、RTC_TR 和 RTC_DR 影子寄存器时，RTC_ICSR 寄存器中的 RSF 位都会被置 1。每隔一个 RTCCLK 周期执行一次复制。为确保这 3 个值来自同一时刻点，读取 RTC_SSR 或 RTC_TR 时会锁定高阶日历影子寄存器中的值，直到读取 RTC_DR。为避免软件对日历执行读访问的时间间隔小于 1 个 RTCCLK 周期：第一次读取日历之后必须通过软件将 RSF 清零，并且软件必须等待到 RSF 置 1 之后才可再次读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

从低功耗模式（停机模式或待机模式）唤醒之后，必须通过软件将 RSF 清零。之后，软件必须等待至 RSF 再次置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

RSF 位必须在唤醒之后而不是进入低功耗模式之前进行清零。

系统复位之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。实际上，系统复位会将影子寄存器复位为其默认值。

初始化之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。同步之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

当 RTC_CR 寄存器中的 BYPSHAD 控制位置 1 时（旁路影子寄存器）

读取日历寄存器时会直接从日历计数器获取值，这样便无需等待直至 RSF 位为 1。这对于从低功耗模式（停止模式或待机模式）退出后的情况特别有用，因为影子寄存器在这些模式下没有被更新。

当 BYPSHAD 位置 1 时，如果在对寄存器的两次读访问之间出现 RTCCLK 沿，则不同寄存器的结果彼此可能不一致。此外，如果在读操作期间出现 RTCCLK 沿，则可能导致其中一个寄存器的值不正确。软件必须分两次读取所有寄存器，然后将两次结果加以比较来确认数据是否一致和正确。此外，软件也可以只比较两次读取日历寄存器得到的结果的最低位。

注意：当 BYPSHAD=1 时，读取日历寄存器的指令需要一个额外的 APB 周期才能完成。

28.3.10. 复位 RTC

日历影子寄存器（RTC_SSR、 RTC_TR 和 RTC_DR）以及 RTC 状态寄存器 (RTC_ICSR)的某些位通过所有可用的系统复位源复位为各自的默认值。

相反，以下寄存器则通过 RTC 域复位来复位为各自的默认值并且不受系统复位的影响： RTC 当前日历寄存器、 RTC 控制寄存器 (RTC_CR)、预分频器寄存器 (RTC_PRER)、 RTC 校准寄存器 (RTC_CALR)、 RTC 移位寄存器 (RTC_SHIFTR)、 RTC 时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）、唤醒定时器寄存器 (RTC_WUTR) 以及闹钟 A 和闹钟 B 寄存器（RTC_ALRMASSR/RTC_ALRMAR 和 RTC_ALRMBSSR/RTC_ALRMBR）。

此外，当由 LSE 提供时钟时，如果复位源并非 RTC 域复位源（有关不受系统复位影响的 RTC 时钟源的详细信息，请参见 RCC），则 RTC 将在系统复位时保持运行状态。发生 RTC 域复位时，RTC 会停止工作，并且所有 RTC 寄存器都会设置为各自的复位值。

28.3.11. RTC 同步

RTC 可与高精度的远程时钟同步。在读取亚秒字段后（RTC_SSR 或 RTC_TSSSR），即可计算远程时钟的时间与 RTC 之间的精准偏差。之后，可使用 RTC_SHIFTR 对 RTC 的时钟进行零点几秒的“平移”，经过调整后可消除此偏差。

RTC_SSR 包含同步预分频器计数器的值。这样，便可计算分辨率低至 1/(PREDIV_S + 1)秒的 RTC 的准确时间。因此，可通过增大同步预分频器的值 (PREDIV_S[14:0]) 来提高分辨率。将 PREDIV_S 设置为 0x7FFF 时，可得到允许的最大分辨率（30.52 μs，时钟频率为 32768 Hz）。

但是，提高 PREDIV_S 意味着必须降低 PREDIV_A 才能将同步预分频器的输出维持在 1 Hz。

这样，异步预分频器的输出频率会增大， RTC 的动态功耗也会相应增加。

可以使用 RTC 平移控制寄存器(RTC_SHIFTR)对 RTC 进行微调。可以用大小为 1/(PREDIV_S+1)秒的分辨率对 RTC_SHIFTR 进行写操作，将时钟平移（延迟或提前）最长 1 秒。在这种平移操作中，会将 SUBFS[14:0] 值加到同步预分频器计数器 SS[15:0] 中：这将使时钟产生延迟。如果同时将 ADD1S 位置 1，则会增加一秒，与此同时减去的时间为零点几秒，因此将使时钟提前。

注意：初始化平移操作前，用户必须检查确认 SS[15] = 0，以确保不会发生上溢。

对 RTC_SHIFTR 寄存器执行写操作以启动平移操作时，硬件会将 SHPF 标志置 1 以指示平移操作挂起。完成平移操作时，硬件会将该位清零。

注意： 该同步功能与参考时钟检测功能不兼容：当 REFCKON = 1 时，固件不能对 RTC_SHIFTR 执行写操作。

28.3.12. RTC 参考时钟检测

RTC 日历更新可与参考时钟 RTC_REFIN（通常为市电频率， 50 Hz 或 60 Hz）同步。

RTC_REFIN 参考时钟的精度应高于 32.768 kHz LSE 时钟。使能 RTC_REFIN 检测时（将 RTC_CR 的 REFCKON 位置 1），日历仍由 LSE 提供时钟，而 RTC_REFIN 用于补偿不准确的日历更新频率 (1 Hz)。

每个 1 Hz 时钟边沿都与最近的 RTC_REFIN 时钟边沿进行比较（如果在给定的时间窗口内发现一个边沿）。在大多数情况下，两个时钟边沿恰好对齐。当 1 Hz 时钟由于 LSE 时钟不精确而发生偏离时， RTC 会稍微偏移 1 Hz 时钟，以便后续的 1 Hz 时钟边沿能够对齐。利用这种机制，可使日历像参考时钟一样精确。

RTC 使用 32.768 kHz 石英产生的 256 Hz 时钟 (ck_apre) 检测是否存在参考时钟源。大约在日历每次更新时（每 1 秒钟），便会在时间窗口期间执行一次检测。检测到第一个参考时钟边沿时，该窗口等于 7 个 ck_apre 周期。随后的日历更新使用长度为 3 个 ck_apre 周期的较小窗口。

每次在窗口中检测到参考时钟时，都会强制输出 ck_apre 时钟的异步预分频器进行重载。当参考时钟与 1 Hz 时钟对齐时，此操作不起作用，因为预分频器会在同一时刻重载。当时钟不对齐时，重载操作会微调后续的 1 Hz 时钟边沿，使其与参考时钟对齐。

如果参考时钟停止（在 3 个 ck_apre 窗口内未出现参考时钟边沿），日历将仅根据 LSE 时钟进行连续更新。 RTC 随后使用 ck_spre 边沿上居中的大检测窗口（7 个 ck_apre 周期）等待参考时钟。

使能 RTC_REFIN 检测后，必须将 PREDIV_A 和 PREDIV_S 设置为各自的默认值：

- PREDIV_A = 0x007F
- PREDIV_S = 0x00FF

注意：RTC_REFIN 时钟检测在待机模式下不可用。

28.3.13. RTC 精密数字校准

RTC 频率可采用约 0.954 ppm 的分辨率进行数字校准， 校准范围为 -487.1 ppm 到 +488.5 ppm。

使用一系列微调 （增加和/或减少单独的 RTCCLK 脉冲）进行频率校正。这些微调的分布非常均匀，因此 RTC 的校准效果相当好，即使在短时间内持续观察也是如此。

当输入频率为 32768 Hz 时，精密数字校准的周期约为 220 个 RTCCLK 脉冲，即 32 秒。此周期由一个通过 RTCCLK 提供时钟信号的 20 位计数器 cal_cnt[19:0] 维持。

精密数字校准寄存器 (RTC_CALR) 可指定 32 秒周期内要减少的 RTCCLK 时钟周期数：

- 将位 CALM[0] 置 1 时， 32 秒周期内将只减少 1 个脉冲。
- 将 CALM[1] 置 1 时，将减少 2 个周期。
- 将 CALM[2] 置 1 时，将减少 4 个周期。
- 依此类推，将 CALM[8] 置 1 时，将减少 256 个时钟。

注意： CALM[8:0] (RTC_CALR) 可指定 32 秒周期内要减少的 RTCCLK 脉冲数。将位 CALM[0] 置 1 时， 32 秒周期内将只减少 1 个脉冲（当 cal_cnt[19:0] = 0x80000 时）；将 CALM[1] 置 1 时，将减少 2 个周期（当 cal_cnt = 0x40000 和 0xC0000 时）；将 CALM[2] 置 1 时，将减少 4 个周期（当 cal_cnt = 0x20000/0x60000/0xA0000/0xE0000 时）；依此类推，将 CALM[8] 置 1 时，将减少 256 个时钟（当 cal_cnt = 0xXX800 时）。

使用适当分辨率时， CALM 可使 RTC 频率减少最多 487.1 ppm，而 CALP 可用于使频率增加 488.5 ppm。将 CALP 置 1，可每隔 211 个 RTCCLK 周期有效插入一个额外的 RTCCLK 脉冲，这意味着每 32 秒周期可增加 512 个时钟。

与 CALM 和 CALP 配合使用时，可在 32 秒周期内增加一个范围为 -511 到 +512 RTCCLK 周期的偏差，对应的校准范围为 -487.1 ppm 到 +488.5 ppm，分辨率约为 0.954 ppm。

若输入频率 (F_{RTCCLK}) 已知，可通过以下公式计算有效校准频率 (F_{CAL})：

$$F_{CAL} = F_{RTCCLK} \times [1 + (CALP \times 512 - CALM) / (2^{20} + CALM - CALP \times 512)]$$

28.3.13.1. PREDIV_A < 3 条件下的校准

当异步预分频器值（RTC_PRER 寄存器中的 PREDIV_A 位）小于 3 时，不能将 CALP 位置 1。如果 CALP 已置 1 并且 PREDIV_A 位的值小于 3，则会忽略 CALP，即假定 CALP 等于 0 而执行校准。

要在 PREDIV_A 小于 3 的条件下执行校准，应降低同步预分频器值 (PREDIV_S) 以便每秒内可加速 8 个 RTCCLK 时钟周期，这意味着每 32 秒可增加 256 个时钟周期。因此，仅使用 CALM 位，可在每 32 秒内有效增加 255 到 256 个时钟脉冲（对应的校准范围为 243.3 ppm 到 244.1 ppm）。

在标称 RTCCLK 频率 32768 Hz 下，当 PREDIV_A 等于 1 时（分频系数为 2），应将 PREDIV_S 设置为 16379 而不是 16383（少 4）。唯一相关的其他情况是，当 PREDIV_A 等于 0 时，应将 PREDIV_S 设置为 32759 而不是 32767 （少 8）。

如果以这种方式减少 PREDIV_S，则采用以下公式计算校准输入时钟的有效频率：

$$F_{CAL} = F_{RTCCLK} \times [1 + (256 - CALM) / (2^{20} + CALM - 256)]$$

在这种情况下，如果 RTCCLK 恰好为 32768.00 Hz，则当 CALM[7:0] 等于 0x100 时（CALM 范围的中值），说明设置正确。

28.3.13.2. 验证 RTC 校准

通过测量 RTCCLK 的精确频率，计算正确的 CALM 和 CALP 值以确保 RTC 精度。此外，还为应用提供了一个可选的 1 Hz 输出，用来测量和验证 RTC 精度。

如果在有限的间隔内测量 RTC 的精确频率，则会导致测量期间产生最多 2 个 RTCCLK 时钟周期的测量误差，具体取决于数字校准周期与测量周期的对齐方式。

但是，如果测量周期与校准周期的长度相同，则可以消除此测量误差。在这种情况下，观测到的唯一误差是由数字校准的分辨率导致的误差。

- 默认情况下，校准周期为 32 秒。

在此模式下，测量整个 32 秒内 1 Hz 输出的精度，可确保测量误差在 0.477 ppm 内（32 秒内为 0.5 个 RTCCLK 周期，受校准分辨率限制）。

- 可将 RTC_CALR 寄存器的 CALW16 位置 1，以强制 16 秒的校准周期。

此时，可在 16 秒内测量 RTC 精度，产生的最大误差为 0.954 ppm（16 秒内为 0.5 个 RTCCLK 周期）。但是，由于校准分辨率降低，长期的 RTC 精度也会降到 0.954 ppm：将 CALW16 置 1 时， CALM[0] 位将始终保持为 0。

- 可将 RTC_CALR 寄存器的 CALW8 位置 1，以强制 8 秒的校准周期。

此时，可在 8 秒内测量 RTC 精度，产生的最大误差为 1.907 ppm（8 秒内为 0.5 个 RTCCLK 周期）。长期的 RTC 精度也会降到 1.907 ppm：将 CALW8 置 1 时， CALM[1:0] 位将始终保持为 00。

28.3.13.3. 动态重校准

当 RTC_ICSR/INITF=0 时，可动态更新校准寄存器 (RTC_CALR)，具体步骤如下：

1. 轮询 RTC_ICSR/RECALPF（重新校准挂起标志）。
2. 若该标志为 0，则可以根据需要向 RTC_CALR 写入新值。随后 RECALPF 位会被自动置为 1。
3. 新校准设置将在对 RTC_CALR 执行写操作之后的三个 ck_apre 周期内生效。

28.3.14. 时间戳功能

将 RTC_CR 寄存器的 TSE 或 ITSE 位置 1 可使能时间戳。

将 TSE 置 1 时：

当在 RTC_TS 引脚上检测到时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。

将 TAMPTS 置 1 时：

当在 TAMP_INx 引脚上检测到入侵事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。

将 ITSE 置 1 时：

当检测到内部时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、 RTC_TSTR 和 RTC_TSDR）中。切换至 V_{BAT} 电源可生成内部时间戳事件。由于内部事件或外部事件而发生时间戳事件时， RTC_SR 寄存器中的时间戳标志位 (TSF) 将置 1。如果是内部事件， RTC_SR 寄存器中的 ITSF 标志也将置 1。

通过将 RTC_CR 寄存器中的 TSIE 位置 1，可在发生时间戳事件时生成中断。

如果在时间戳标志 (TSF) 已置 1 的条件下检测到新的时间戳事件，则时间戳上溢标志 (TSOVF)将置 1，而时间戳寄存器（RTC_TSTR 和 RTC_TSDR）将保持上一事件的结果。

注意：由于同步过程， TSF 将在时间戳事件后最多 4 个 ck_apre 周期内置 1。

TSOVF 的产生中不存在延迟。也就是说如果两个时间戳事件接连发生， TSOVF 可能为“1”而 TSF 为“0”。因此，建议只在检测到 TSF 为“1”后再轮询 TSOVF。

注意：如果在 TSF 位清零后紧接着发生时间戳事件，则 TSF 和 TSOVF 位都将置 1。为防止在时间戳事件发生的同时屏蔽该事件，除非已将 TSF 位读取为 1，否则应用程序不得将 0 写入 TSF 位。

此外，入侵事件可能导致时间戳被记录。请参见 RTC 控制寄存器 (RTC_CR) 中的 TAMPTS 控制位的说明。

28.3.15. 校准时钟输出

将 RTC_CR 寄存器中的 COE 位置 1 时，会在 CALIB 器件输出上提供一个参考时钟。

如果 RTC_CR 寄存器中的 COSEL 位置 0 且 PREDIV_A = 0x7F，则 CALIB 频率为 $f_{RTCCLK/64}$。

这相当于 RTCCLK 频率为 32.768 kHz 时， 512 Hz 的校准输出。 CALIB 占空比是不规则的：下降沿上存在轻微抖动。因此推荐使用上升沿。

如果 COSEL 置 1 且“PREDIV_S+1”为 256 的非零整数倍（比如：PREDIV_S[7:0] = 0xFF），则 CALIB 频率为 $f_{RTCCLK}/(256 * (PREDIV\_A+1))$。这相当于 RTCCLK 频率为 32.768 kHz 时， 1 Hz 的校准输出，其中预分频器为默认值（PREDIV_A = Ox7F、 PREDIV_S = 0xFF）。

注意：选择 CALIB 输出时，将自动配置 RTC_OUT 引脚。

COSEL 位清零时， CALIB 输出为异步预分频器的第 6 级输出。

COSEL 位置 1 时， CALIB 输出为同步预分频器的第 8 级输出。

28.3.16. 入侵和闹钟输出

RTC_CR 寄存器中的 OSEL[1:0] 控制位用于激活闹钟输出 TAMPALRM，以及选择输出的功能。这些功能可反映 RTC_SR 寄存器中相应标志的内容。

当 RTC_CR 中的 TAMPOE 控制位置 1 时，所有外部和内部入侵标志都将进行逻辑或运算，然后连接到 TAMPALRM 输出。如果 OSEL = 00，则 TAMPALRM 输出仅反映入侵标志。如果 OSEL ≠ 00，则 TAMPALRM 上的信号提供入侵标志和闹钟 A、 B 或唤醒标志。

TAMPALRM 输出的极性由 RTC_CR 中的 POL 控制位确定，这样当 POL 置 1 时会输出选定标志位的相反值。

28.3.16.1. TAMPALRM 输出

使用 RTC_CR 寄存器中的控制位 TAMPALRM_TYPE 可将 TAMPALRM 引脚配置为输出开漏或输出推挽。可凭借 RTC_CR 中的 TAMPALRM_PU 在输出模式下应用内部上拉。

注意： TAMPALRM 输出使能后，其优先级高于 RTC_OUT 上的 CALIB。

选择 TAMPALRM 输出时，将自动配置 RTC_OUT 引脚。如果在 RTC 中将 TAMPALRM 配置为开漏，则必须将 RTC_OUT GPIO 配置为输入。

28.4. RTC 低功耗模式

下表为 RTC 在低功耗模式下的工作情况：

表 28-3 RTC 低功耗模式

模式	说明
睡眠	无影响。 RTC 中断能唤醒睡眠模式。
停止	当 RTC 时钟源选择 LSI 或者 LSE 时，RTC 保持工作。 RTC 中断能唤醒停止模式。
待机	当 RTC 时钟源选择 LSI 或者 LSE 时，RTC 保持工作。 RTC 中断能唤醒待机模式。

下表为各模式下 RTC 引脚功能：

表 28-4 RTC 引脚功能

功能	非待机模式以外的所有低功耗模式下的功能性	待机模式下的功能性	V_{BAT} 模式下的的功能性
RTC_TS	有	有	有
RTC_REFIN	有	无	无
RTC_OUT	有	有	有

28.5. RTC 中断

下表为 RTC 中断请求：

表 28-5 RTC 中断请求

中断事件	事件标志(1)	使能控制位(2)	中断清除方法	退出睡眠 模式	退出停机和待机模式
闹钟 A	ALRAF	ALRAIE	在 CALRAF 中写 1	有	有(3)
闹钟 B	ALRBF	ALRBIE	在 CALRBF 中写 1	有	有(3)
时间戳	TSF	TSIE	在 CTSF 中写 1	有	有(3)
唤醒定时器中断	WUTF	WUTIE	在 CWUTF 中写 1	有	有(3)

1. 事件标志位于 RTC_SR 寄存器中。
2. 中断屏蔽标志（由事件标志位和使能控制位的逻辑与运算结果产生）位于 RTC_MISR 寄存器中。
3. 仅当 RTC 时钟源为 LSE 或 LSI 时，才能从停止和待机模式唤醒。

28.6. RTC 寄存器

28.6.1. RTC 时间寄存器 (RTC_TR)

RTC_TR 是日历时间影子寄存器。只能在初始化模式下对该寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x00

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:0x0000 0000（当 BYPSHAD=0 时，当 BYPSHAD=1 时不受影响）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22	PM	RW	0	AM/PM 符号。 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	RW	0	小时的十位（BCD 格式）
19:16	HU[3:0]	RW	0	小时的个位（BCD 格式）
15	Reserved	-	-	保留
14:12	MNT[2:0]	RW	0	分钟的十位（BCD 格式）
11:8	MNU[3:0]	RW	0	分钟的个位（BCD 格式）
7	Reserved	-	-	保留
6:4	ST[2:0]	RW	0	秒的十位（BCD 格式）
3:0	SU[3:0]	RW	0	秒的个位（BCD 格式）

28.6.2. RTC 日期寄存器 (RTC_DR)

RTC_DR 是日历日期影子寄存器。只能在初始化模式下对该寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x04

复位值（RTC 域复位）:0x0000 2101

复位值（系统复位）:0x00002101（当 BYPSHAD=0 时，当 BYPSHAD=1 时不受影响）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	YT[3:0]				YU[3:0]			
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:20	YT[3:0]	RW	0	年的十位（BCD 格式）
19:16	YU[3:0]	RW	0	年的个位（BCD 格式）
15:13	WDU[2:0]	RW	0x1	星期几 000：禁止 001：星期一 … 111：星期日

12	MT	RW	0	月的十位（BCD 格式）
11:8	MU[3:0]	RW	0x1	月的个位（BCD 格式）
7:6	Reserved	-	-	保留
5:4	DT[1:0]	RW	0	日期的十位（BCD 格式）
3:0	DU[3:0]	RW	0x1	日期的个位（BCD 格式）

28.6.3. RTC 亚秒寄存器 (RTC_SSR)

偏移地址:0x08

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:0x0000 0000（当 BYPSHAD=0 时，当 BYPSHAD=1 时不受影响）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SS[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	SS[15:0]	R	0	亚秒值。 该寄存器是同步预分频器计数器的值。 亚秒值= (PREDIV_S - SS) / (PREDIV_S + 1) 注：仅当执行平移操作之后， SS 才能大于 PREDIV_S。在这种情况下，正确的时间/日期比 RTC_TR/RTC_DR 所指示的时间/日期慢一秒钟

28.6.4. RTC 初始化控制和状态寄存器（RTC_ICSR）

此寄存器受写保护。

偏移地址：0x0C

复位值（RTC 域复位）:0x0000 0007

复位值（系统复位）:不受影响（INIT、 INITF 和 RSF 位除外，它们被清零）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RECALPF
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INIT	INITF	RSF	INITS	SHPF	WUTWF	ALRBWF	ALRAWF
								RW	R	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	RECALPF	R	0	重新校准挂起标志. 当软件对 RTC_CALR 寄存器执行写操作时，RECALPF 状态标志将自动置 1，指示 RTC_CALR 寄存器已屏蔽。当采用新的校准设置时，此位恢复为 0。
15:8	Reserved	-	-	保留
7	INIT	RW	0	初始化模式 0：自由运行模式

				1：初始化模式，用于编程时间（RTC_TR）和日期寄存器（RTC_DR）以及预分频器寄存器(RTC_PRER)。计数器停止计数，当 INIT 被复位后，计数器从新值开始计数。
6	INITF	R	0	初始化标志 当此位置 1 时， RTC 处于初始化状态，此时可更新事件、日期和预分频器寄存器。 0：不允许更新日历寄存器 1：允许更新日历寄存器
5	RSF	RC_W0	0	寄存器同步标志 每次将日历寄存器的值复制到影子寄存器（RTC_SSRx、RTC_TRx 和 RTC_DRx）时，都会由硬件将此位置 1。在初始化模式下、平移操作挂起时(SHPF = 1) 或在旁路影子寄存器模式(BYPSHAD = 1)下，此位由硬件清零。该位还可由软件清零。 在初始化模式下，该位可由软件或硬件清零。 0：日历影子寄存器尚未同步 1：日历影子寄存器已同步
4	INITS	R	0	初始化状态标志 当日历年份字段不为 0 时（RTC 域复位状态），由硬件将该位置 1。 0：日历尚未初始化 1：日历已经初始化
3	SHPF	R	0	平移操作挂起. 只要通过对 RTC_SHIFTR 寄存器执行写操作来启动平移操作，此标志便由硬件置 1。执行完相应的平移操作后，此标志由硬件清零。对 SHPF 位执行写入操作不起作用。 0：没有平移操作挂起 1：某个平移操作挂起
2	WUTWF	R	1	唤醒定时器写标志. 在 RTC_CR 寄存器中的 WUTE 位置 0 后，当 WUT 的值可更改时，由硬件将此位置 1。 在 RTC_CR 寄存器中的 WUTE 位置 1 后，硬件在 3 个 RTCCLK 后将该位清零。 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
1	ALRBWF	R	1	闹钟 B 写标志 在 RTC_CR 寄存器中的 ALRBE 位置 0 之后，当闹钟 B 的值可更改时，由硬件将此位置 1。 在 RTC_CR 寄存器中的 ALRBE 位置 1 之后，硬件在 3 个 RTCCLK 后将该位清零。 该位在初始化模式下由硬件清零。 0：不允许更新闹钟 B 1：允许更新闹钟 B
0	ALRAWF	R	1	闹钟 A 写标志.

				在 RTC_CR 寄存器中的 ALRAE 位置 0 后，当闹钟 A 的值可更改时，由硬件将此位置 1。 在 RTC_CR 寄存器中的 ALRAE 位置 1 之后，硬件在 3 个 RTCCLK 后将该位清零。 该位在初始化模式下由硬件清零。 0：不允许更新闹钟 A 1：允许更新闹钟 A

28.6.5. RTC 预分频器寄存器 (RTC_PRER)

只能在初始化模式下对该寄存器执行写操作。必须通过两次独立的写访问执行初始化。

此寄存器受写保护。

偏移地址:0x10

复位值（RTC 域复位）:0x007F 00FF

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PREDIV_A[6:0]						
									RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	PREDIV_S[14:0]														
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22:16	PREDIV_A [22:16]	RW	0x7F	异步预分频系数 下面是异步分频系数的公式： ck_apre 频率 = RTCCLK 频率/(PREDIV_A+1)
15	Reserved	-	-	保留
14:0	PREDIV_S[14:0]	RW	0x00FF	同步预分频系数 下面是同步分频系数的公式： ck_spre 频率 = ck_apre 频率/(PREDIV_S+1) 注：在平移功能时，该寄存器配置值必须大于 1.

28.6.6. RTC 唤醒定时器寄存器 (RTC_WUTR)

仅当 RTC_ICSR 中的 WUTWF 置 1 时才可对此寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x14

复位值（RTC 域复位）:0x0000 FFFF

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WUT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	WUT[15:0]	RW	0xFFFF	唤醒自动重载值位

				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。 当 WUCKSEL[2] = 1 时，唤醒定时器变为 17 位，WUCKSEL[1] 等效为 WUT[16]，即要重载到定时器的最高有效位。 WUTF 第一次置 1 发生在 WUTE 置 1 之后 WUT 到 (WUT+1) 个 ck_wut 周期之间。禁止在 WUCKSEL[2:0] = 011 (RTCCLK/2) 时将 WUT[15:0] 设置为 0x0000。

28.6.7. RTC 控制寄存器（RTC_CR）

此寄存器受写保护。

偏移地址： 0x18

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	TAMPALARM_TYPE	TAMPALARM_PU	Res.	Res.	TAMPOE	TAMPTS	ITSE	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	Res.	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	TAMPALRM_TYPE	RW	0	TAMPALRM 输出类型 0： TAMPALRM 为推挽输出 1： TAMPALRM 为开漏输出
29	TAMPALARM_PU	RW	0	TAMPALRM 上拉使能 0：未在 TAMPALRM 输出上应用上拉 1：在 TAMPALRM 输出上应用上拉
28:27	Reserved	-	-	保留
26	TAMPOE	RW	0	入侵检测输出到 TAMPALRM 使能 0：入侵标志不连接到 TAMPALRM 1：入侵标志连接到 TAMPALRM（结合 OSEL 提供的信号和 POL 提供的极性）。
25	TAMPTS	RW	0	发生入侵检测事件时激活时间戳 0：发生入侵检测事件时不保存 RTC 时间戳 1：发生入侵检测事件时保存 RTC 时间戳 即便 RTC_CR 寄存器中的 TSE = 0， TAMPTS 仍有效。时间戳标志在入侵标志之后置 1，因此，如果 TAMPTS 和 TSIE 置 1，建议禁止入侵中断以避免处理 2 个中断。
24	ITSE	RW	0	内部事件时间戳使能 0：禁止内部时间戳事件 1：使能内部时间戳事件

23	COE	RW	0	校准输出使能 此位使能 CALIB 输出 0：禁止校准输出 1：使能校准输出
22:21	OSEL[1:0]	RW	0	输出选择 这些位用于选择要连接到 TAMPALRM 输出的标志。 00：禁止输出 01：使能闹钟 A 输出 10：使能闹钟 B 输出 11：使能唤醒输出
20	POL	RW	0	输出极性 此位用于配置 TAMPALRM 输出的极性。 0：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]）或 TAMPxF/ITAMPxF 置 1 时（TAMPOE = 1 时），该引脚为高电平。 1：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]）或 TAMPxF/ITAMPxF 置 1 时（TAMPOE = 1 时），该引脚为低电平。
19	COSEL	RW	0	校准输出选择 当 COE=1 时，此位可选择 CALIB 上输出的信号。 0：校准输出为 512 Hz 1：校准输出为 1 Hz 在 RTCCLK 为 32.768 kHz 且预分频器为其默认值（PREDIV_A = 127 且 PREDIV_S = 255）的条件下，这些频率有效。
18	BKP	RW	0	备份 用户可对此位执行写操作以记录是否已对夏令时进行更改。
17	SUB1H	W	0	减少 1 小时（冬季时间更改） 当该位在初始化模式以外的模式下置 1 时，如果当前小时不是 0，则日历时间将减少 1 小时。此位始终读为 0。 当前小时为 0 时，将此位置 1 没有任何作用。 0：无影响 1：将当前时间减少 1 小时。这可用于冬季时间更改
16	ADD1H	W	0	增加 1 小时（夏季时间更改） 当该位在初始化模式以外的模式下置 1 时，日历时间将增加 1 小时。此位始终读为 0。 0：无影响 1：将当前时间增加 1 小时。这可用于夏季时间更改
15	TSIE	RW	0	时间戳中断使能 0：禁止时间戳中断 1：使能时间戳中断
14	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
13	ALRBIE	RW	0	闹钟 B 中断使能

				0：禁止闹钟 B 中断 1：使能闹钟 B 中断
12	ALRAIE	RW	0	闹钟 A 中断使能 0：禁止闹钟 A 中断 1：使能闹钟 A 中断
11	TSE	RW	0	时间戳使能 0：禁止时间戳 1：使能时间戳
10	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 注： 唤醒定时器禁止时，需要等到 WUTWF=1 后才能重新使能
9	ALRBE	RW	0	闹钟 B 使能 0：禁止闹钟 B 1：使能闹钟 B
8	ALRAE	RW	0	闹钟 A 使能 0：禁止闹钟 A 1：使能闹钟 A
7	Reserved	-	-	保留
6	FMT	RW	0	小时格式 0：24 小时/天格式 1：AM/PM 小时格式
5	BYPSHAD	RW	0	旁路影子寄存器 0：日历值（从 RTC_SSR、 RTC_TR 和 RTC_DR 读取时）取自影子寄存器，该影子寄存器每两个 RTCCLK 周期更新一次。 1：日历值（从 RTC_SSR、 RTC_TR 和 RTC_DR 读取时）直接取自日历计数器。 注： 如果 APB1 时钟的频率低于 7 倍的 RTCCLK 频率，则必须将 BYPSHAD 置 1。
4	REFCKON	RW	0	RTC_REFIN 参考时钟检测使能（50 Hz 或 60 Hz） 0：禁止 RTC_REFIN 检测 1：使能 RTC_REFIN 检测 注： PREDIV_S 必须为 0x00FF。
3	TSEDGE	RW	0	时间戳事件有效边沿 0： RTC_TS 输入上升沿生成时间戳事件 1： RTC_TS 输入下降沿生成时间戳事件 TSEDGE 发生更改时，必须复位 TSE 以避免将 TSF 意外置 1。
2:0	WUCKSEL[2:0]	RW	0	ck_wut 唤醒时钟选择 000：选择 RTC/16 时钟 001：选择 RTC/8 时钟 010：选择 RTC/4 时钟 011：选择 RTC/2 时钟

				10x：选择 ck_spre 时钟（通常为 1 Hz） 11x：选择 ck_spre 时钟（通常为 1 Hz）并将 WUT 计数器值增加 2^{16}

注： 只能在初始化模式下 (RTC_ICSR/INITF = 1) 对此寄存器的位 6 和 4 执行写操作。

WUT = 唤醒单元计数器值。当 WUCKSEL[2:1 = 11] 时， WUT = (0x0000 到 0xFFFF) + 0x10000（增加的值）。

只能在 RTC_CR WUTE 位 = 0 且 RTC_ICSR WUTWF 位 = 1 时对此寄存器的位 2 到 0 执行写操作。

注：不支持在日历小时递增时更改小时，如果这样做可能会出现屏蔽日历小时增量的情况。

ADD1H 和 SUB1H 的更改在下一秒生效。

28.6.8. RTC 写保护寄存器 (RTC_WPR)

偏移地址:0x24

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[7:0]							
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	KEY[7:0]	W	0	写保护密钥 可通过软件对该字节执行写操作。 读取该字节时，始终返回 0x00。

28.6.9. RTC 校准寄存器（RTC_CALR）

此寄存器受写保护。

偏移地址：0x28

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CALP	CALW8	CALW16	Res.	Res.	Res.	Res.	CALM[8:0]								
RW	RW	RW					RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	CALP	RW	0	将 RTC 的频率增加 488.5 ppm 0：不增加 RTCCLK 脉冲。 1：每 211 个脉冲有效插入一个 RTCCLK 脉冲（将频率增加 488.5 ppm）。 此功能应与 CALM 结合使用，后者在高分辨率下会降低日历的频率。如果输入频率为 32768 Hz，则在 32 秒窗口中增加的 RTCCLK 脉冲数按如下公式计算：(512 × CALP) - CALM。
14	CALW8	RW	0	使用 8 秒校准周期 当 CALW8 置 1 时，选择 8 秒校准周期。

				注： CALW8 = 1 时， CALM[1:0] 将始终保持为 00。
13	CALW16	RW	0	使用 16 秒校准周期 当 CALW16 置 1 时，选择 16 秒校准周期。如果 CALW8 = 1，则不得将此位置 1。 注： 当 CALW16 = 1 时， CALM[0] 将始终保持为 0。
12:9	Reserved	-	-	保留
8:0	CALM[8:0]	RW	0	负校准 在 2^{20} 个 RTCCLK 脉冲内屏蔽 CALM 个脉冲（如果输入频率为 32768 Hz，则为 32 秒）来降低日历的频率。其分辨率为 0.9537 ppm。 要提高日历的频率，则应将此功能与 CALP 结合使用。

28.6.10. RTC 平移控制寄存器（RTC_SHIFTR）

此寄存器受写保护。

偏移地址：0x2C

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD1S	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
W															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	SUBFS[14:0]														
	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31	ADD1S	W	0	增加一秒钟 0：无影响 1：对时钟/日历增加一秒钟 此位为只写位且始终读为 0。当平移操作挂起（RTC_ICSR 中的 SHPF = 1）时，对此位执行写操作无作用。 此函数应与 SUBFS 配合使用（请参见下文介绍），以便有效地向原子操作机制的时钟添加亚秒值。
30:15	Reserved	-	-	保留
14:0	SUBFS [14:0]	W	0	减少亚秒值 此位为只写位且始终读为 0。当平移操作挂起（RTC_ICSR 中的 SHPF = 1）时，对此位执行写操作无作用。 写入 SUBFS 的值将加到同步预分频器计数器中。由于该计数器递减计数，此操作可有效地从时钟减去（延迟）以下时间： 延迟（秒） = SUBFS / (PREDIV_S + 1) 当 ADD1S 函数与 SUBFS 结合使用时，可有效地将亚秒值增加到时钟（提前时钟），使时钟提前以下时间： 提前（秒） = (1 - (SUBFS / (PREDIV_S + 1)))。

				注： 对 SUBFS 执行写操作将使 RSF 清零。软件随后会等待至 RSF = 1 以确定影子寄存器已更新为平移后的时间。

28.6.11. RTC 时间戳时间寄存器 (RTC_TSTR)

仅当 RTC_SR 中的 TSF 置 1 时，此寄存器的内容才有效。当 TSF 位复位时，清零此寄存器。

偏移地址:0x30

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PM	HT[1:0]		HU[3:0]			
									R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MNT[2:0]			MNU[3:0]				Res.	ST[2:0]			SU[3:0]			
	R	R	R	R	R	R	R		R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22	PM	R	0	AM/PM 符号。 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	R	0	小时的十位（BCD 格式）
19:16	HU[3:0]	R	0	小时的个位（BCD 格式）
15	Reserved	-	-	保留
14:12	MNT[2:0]	R	0	分钟的十位（BCD 格式）
11:8	MNU[3:0]	R	0	分钟的个位（BCD 格式）
7	Reserved	-	-	保留
6:4	ST[2:0]	R	0	秒的十位（BCD 格式）
3:0	SU[3:0]	R	0	秒的个位（BCD 格式）

28.6.12. RTC 时间戳日期寄存器 (RTC_TSDR)

仅当 RTC_SR 中的 TSF 置 1 时，此寄存器的内容才有效。当 TSF 位复位时，清零此寄存器。

偏移地址:0x34

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res.	Res.	DT[1:0]		DU[3:0]			
R	R	R	R	R	R	R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:13	WDU[2:0]	R	0x0	星期几 000：禁止 001：星期一 … 111：星期日
12	MT	R	0	月的十位（BCD 格式）
11:8	MU[3:0]	R	0x0	月的个位（BCD 格式）

7:6	Reserved	-	-	保留
5:4	DT[1:0]	R	0	日期的十位（BCD 格式）
3:0	DU[3:0]	R	0x0	日期的个位（BCD 格式）

28.6.13. RTC 时间戳亚秒寄存器 (RTC_TSSSR)

仅当 RTC_SR 中的 TSF 置 1 时，此寄存器的内容才有效。当 TSF 位复位时，清零此寄存器。

偏移地址:0x38

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SS[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	SS[15:0]	R	0	亚秒值。 当发生时间戳事件时，该寄存器是同步预分频器计数器的值。

28.6.14. RTC 闹钟 A 寄存器 (RTC_ALRMAR)

仅当 RTC_ICSR 中的 ALRAWF 置 1 时或在初始化模式下，才可以对此寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x40

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	MSK4	RW	0	闹钟 A 日期屏蔽 0：如果日期/日匹配，则闹钟 A 置 1 1：在闹钟 A 比较中，日期/日无关
30	WDSEL	RW	0	星期几选择 0： DU[3:0] 代表日期的个位 1： DU[3:0] 代表星期几 DT[1:0] 为无关位
29:28	DT[1:0]	RW	0	日期的十位（BCD 格式）
27:24	DU[3:0]	RW	0	日期的个位（BCD 格式）
23	MSK3	RW	0	闹钟 A 小时屏蔽 0：如果小时匹配，则闹钟 A 置 1 1：在闹钟 A 比较中，小时无关
22	PM	RW	0	AM/PM 符号。 0：AM 或 24 小时制 1：PM

21:20	HT[1:0]	RW	0	小时的十位（BCD 格式）
19:16	HU[3:0]	RW	0	小时的个位（BCD 格式）
15	MSK2	RW	0	闹钟 A 分钟屏蔽 0：如果分钟匹配，则闹钟 A 置 1 1：在闹钟 A 比较中，分钟无关
14:12	MNT[2:0]	RW	0	分钟的十位（BCD 格式）
11:8	MNU[3:0]	RW	0	分钟的个位（BCD 格式）
7	MSK1	RW	0	闹钟 A 秒屏蔽 0：如果秒匹配，则闹钟 A 置 1 1：在闹钟 A 比较中，秒无关
6:4	ST[2:0]	RW	0	秒的十位（BCD 格式）
3:0	SU[3:0]	RW	0	秒的个位（BCD 格式）

28.6.15. RTC 闹钟 A 亚秒寄存器 (RTC_ALRMASSR)

仅当 RTC_ICSR 中的 ALRAWF 置 1 时或在初始化模式下，才可以对此寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x44

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	MASKSS[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				RW	RW	RW	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	SS[14:0]														
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	MASKSS[3:0]	RW	0	屏蔽从此位开始的最高有效位 0：不对闹钟 A 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余位域均匹配）。 1：在闹钟 A 比较中， SS[14:1] 为无关位。仅比较 SS[0]。 2：在闹钟 A 比较中， SS[14:2] 为无关位。仅比较 SS[1:0]。 3：在闹钟 A 比较中， SS[14:3] 为无关位。仅比较 SS[2:0]。 ... 12：在闹钟 A 比较中， SS[14:12] 为无关位。比较 SS[11:0]。 13：在闹钟 A 比较中， SS[14:13] 为无关位。比较 SS[12:0]。 14：在闹钟 A 比较中， SS[14] 为无关位。比较 SS[13:0]。 15：所有 15 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。 同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。

				注： 同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。
23:15	Reserved	-	-	保留
14:0	SS[14:0]	RW	0	亚秒值。 该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 A。仅比较位 0 到 MASKSS-1

28.6.16. RTC 闹钟 B 寄存器 (RTC_ALRMBR)

仅当 RTC_ICSR 中的 ALRBWF 置 1 时或在初始化模式下，才可以对此寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x48

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	MSK4	RW	0	闹钟 B 日期屏蔽 0：如果日期/日匹配，则闹钟 B 置 1 1：在闹钟 B 比较中，日期/日无关
30	WDSEL	RW	0	星期几选择 0： DU[3:0] 代表日期的个位 1： DU[3:0] 代表星期几 DT[1:0] 为无关位
29:28	DT[1:0]	RW	0	日期的十位（BCD 格式）
27:24	DU[3:0]	RW	0	日期的个位（BCD 格式）
23	MSK3	RW	0	闹钟 B 小时屏蔽 0：如果小时匹配，则闹钟 B 置 1 1：在闹钟 B 比较中，小时无关
22	PM	RW	0	AM/PM 符号。 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	RW	0	小时的十位（BCD 格式）
19:16	HU[3:0]	RW	0	小时的个位（BCD 格式）
15	MSK2	RW	0	闹钟 B 分钟屏蔽 0：如果分钟匹配，则闹钟 B 置 1 1：在闹钟 B 比较中，分钟无关
14:12	MNT[2:0]	RW	0	分钟的十位（BCD 格式）
11:8	MNU[3:0]	RW	0	分钟的个位（BCD 格式）
7	MSK1	RW	0	闹钟 B 秒屏蔽 0：如果秒匹配，则闹钟 B 置 1 1：在闹钟 B 比较中，秒无关
6:4	ST[2:0]	RW	0	秒的十位（BCD 格式）
3:0	SU[3:0]	RW	0	秒的个位（BCD 格式）

28.6.17. RTC 闹钟 B 亚秒寄存器（RTC_ALRMBSSR)

仅当 RTC_ICSR 中的 ALBAWF 置 1 时或在初始化模式下，才可以对此寄存器执行写操作。

此寄存器受写保护。

偏移地址:0x4C

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	MASKSS[3:0]				Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				RW	RW	RW	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	SS[14:0]														
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	MASKSS[3:0]	RW	0	屏蔽从此位开始的最高有效位 0：不对闹钟 B 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余位域均匹配）。 1：在闹钟 B 比较中， SS[14:1] 为无关位。仅比较 SS[0]。 2：在闹钟 B 比较中， SS[14:2] 为无关位。仅比较 SS[1:0]。 3：在闹钟 B 比较中， SS[14:3] 为无关位。仅比较 SS[2:0]。 ... 12：在闹钟 B 比较中， SS[14:12] 为无关位。比较 SS[11:0]。 13：在闹钟 B 比较中， SS[14:13] 为无关位。比较 SS[12:0]。 14：在闹钟 B 比较中， SS[14] 为无关位。比较 SS[13:0]。 15：所有 15 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。 同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。 注： 同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，此位才不为 0。
23:15	Reserved	-	-	保留
14:0	SS[14:0]	RW	0	亚秒值。 该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 B。仅比较位 0 到 MASKSS-1

28.6.18. RTC 状态寄存器（RTC_SR)

此寄存器受写保护。

偏移地址：0x50

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITSF	TSOVF	TSF	WUTF	ALRBF	ALRAF
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	ITSF	R	0	内部时间戳标志 发生内部时间戳事件时，由硬件将此标志置 1。
4	TSOVF	R	0	时间戳溢出标志。 当在 TSF 已置 1 的情况下发生时间戳中断时，由硬件将此标志置 1。 建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则溢出事件可能会被漏掉。
3	TSF	R	0	时间戳标志 发生时间戳事件时，由硬件将此标志置 1。 如果 ITSF 标志已置 1，必须将 TSF 与 ITSF 一起清零。
2	WUTF	R	0	唤醒定时器标志. 当唤醒自动重载计数器计数到 0 时，由硬件将此标志置 1。 软件必须在 WUTF 再次置 1 的 1.5 个 RTCCLK 周期之前将该标志清零。
1	ALRBF	R	0	闹钟 B 标志 当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 B 寄存器 (RTC_ALRMBR) 匹配时，由硬件将该标志置 1。
0	ALRAF	R	0	闹钟 A 标志. 当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 A 寄存器 (RTC_ALRMAR) 匹配时，由硬件将该标志置 1。

28.6.19. RTC 屏蔽中断状态寄存器（RTC_MISR）

此寄存器受写保护。

偏移地址：0x54

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITSMF	TSOVMF	TSMF	WUTMF	ALRBMF	ALRAMF
										R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留

5	ITSMF	R	0	内部时间戳屏蔽标志 发生内部时间戳事件时，且 ITSE 为 1 时由硬件置位，触发中断。
4	TSOVMF	R	0	时间戳溢出屏蔽标志。 当在 TSMF 已置 1 的情况下发生时间戳事件时，由硬件将此标志置 1。 建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则溢出事件可能会被漏掉。
3	TSMF	R	0	时间戳屏蔽标志 发生时间戳事件时，且 TSE 为 1 时，硬件将此标志置 1。 如果 ITSMF 标志已置 1，必须将 TSMF 与 ITSMF 一起清零。
2	WUTMF	R	0	唤醒定时器屏蔽标志. 当唤醒自动重载计数器计数到 0 时，且 WUTIE 为 1 时由硬件将此标志置 1。 软件必须在 WUTF 再次置 1 的 1.5 个 RTCCLK 周期之前将该标志清零。
1	ALRBMF	R	0	闹钟 B 屏蔽标志。 当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 B 寄存器 (RTC_ALRMBR) 匹配，且 ALRBIE 为 1 时，由硬件将该标志置 1。
0	ALRAMF	R	0	闹钟 A 屏蔽标志。 当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 A 寄存器 (RTC_ALRMAR) 匹配，且 ALRAIE 为 1 时，由硬件将该标志置 1。

28.6.20. RTC 状态清零寄存器（RTC_SCR）

偏移地址：0x5C

复位值（RTC 域复位）:0x0000 0000

复位值（系统复位）:不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CITSF	CTSOVF	CTSF	CWUTF	CALRBF	CALRAF
										W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	CITSF	W	0	清零内部时间戳标志 写入 1 时清零 ITSF 位
4	CTSOVF	W	0	清零时间戳溢出标志。 写入 1 时清零 TSOVF 位

				建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则溢出事件可能会被漏掉。
3	CTSF	W	0	清零时间戳标志 写入 1 时清零 TSF 位 如果 ITSF 标志已置 1，必须通过置 1 CRSF 和 CITSF 的方式将 TSF 与 ITSF 一起清零。
2	CWUTF	W	0	清零唤醒定时器标志. 写入 1 时清零 WUTF 位
1	CALRBF	W	0	清零闹钟 B 标志 写入 1 时清零 ALRBF 位
0	CALRAF	W	0	清零闹钟 A 标志. 写入 1 时清零 ALRAF 位

29. 入侵和备份寄存器（TAMP&BKP）

29.1. TAMP 简介

备份寄存器是 8 个 32 位的寄存器，可用来存储 32 个字节的用户应用程序数据。这些寄存器的内容受入侵检测电路的保护，因此可用于存储敏感数据。1 个入侵引脚和 1 个内部入侵可用于实现防入侵检测。外部入侵引脚既可配置为边沿检测，也可配置为带或不带过滤的电平检测。

该模块处在备份域里，当 V_{CC} 电源被切断，他们仍然由 V_{BAT} 维持供电。当系统在待机模式下被唤醒，或系统复位或电源复位(POR/BOR)时，它们也不会被擦除。该描述针对 V_{BAT} 独立封装的情况。

备份寄存器由备份域上电复位及 RTC 域软复位复位（BPOR+BDRST），且在检测到侵入事件时，擦除备份寄存器内容。

29.2. TAMP 主要特性

- 支持 32 字节数据备份寄存器
- 用来管理防侵入检测并具有中断功能的状态/控制寄存器
- 1 个外部入侵检测事件
- 1 个内部入侵事件
- 任何入侵检测均可生成一个 RTC 时间戳事件
- 任何入侵检测均可擦除备份寄存器
- 备份寄存器受 FLASH 的 RDP(读保护)保护，当 RDP 等级从 1 变为 0 时，擦除备份寄存器的所有数据

29.3. TAMP 功能描述

29.3.1. TAMP 框图

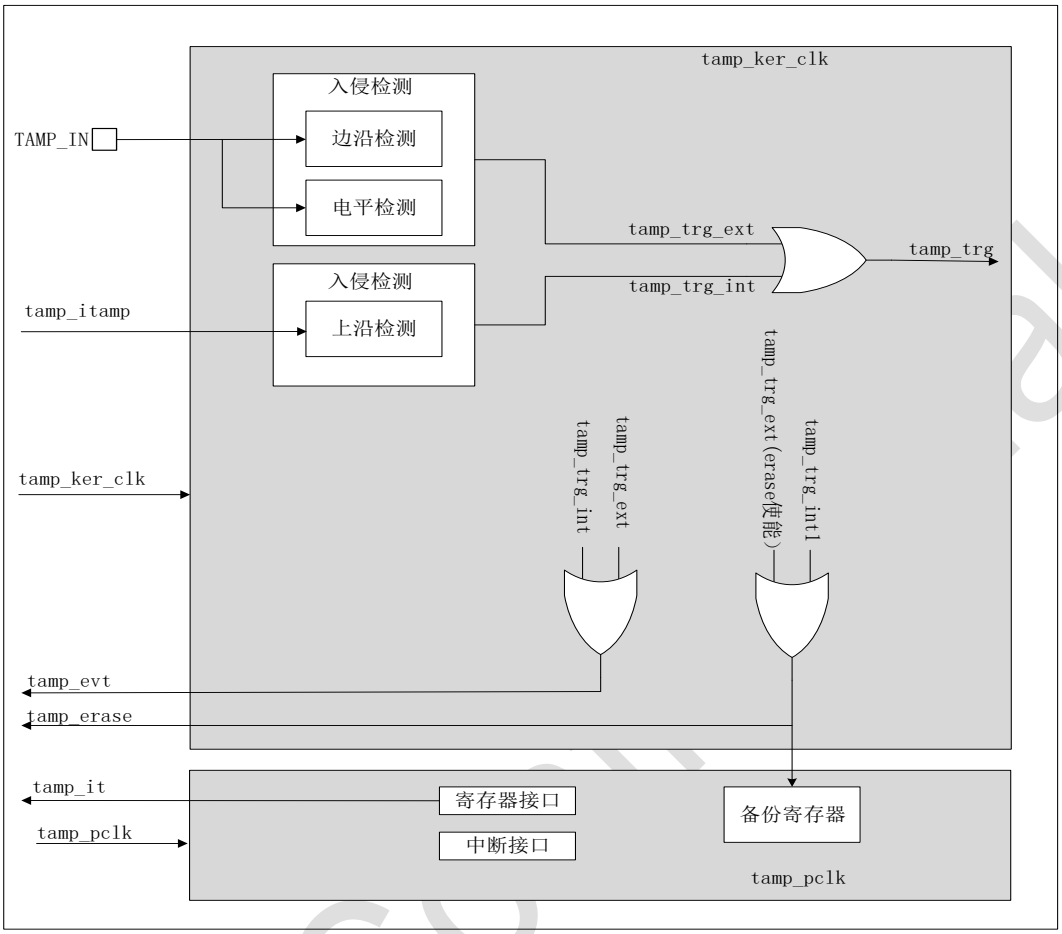


图 29-1 TAMP 框图

29.3.2. TAMP 引脚和接口描述

表 29-1 TAMP 信号描述

Names	Signal type	Description
TAMP_IN	输入	TAMP 外部触发引脚（PA8）
tamp_itamp	输入	TAMP 内部触发引脚
tamp_ker_clk	输入	TAMP 内核时钟，连接到 rtc_ker_ck，也即 RTCCLK
tamp_pclk	输入	TAMP APB 时钟
tamp_evt	输出	检测到入侵事件，用于生成 RTC 时间戳
tamp_erase	输出	检测到入侵事件，用于擦除机密信息：备份寄存器
tamp_it	输出	TAMP 中断
tamp_trg	输出	TAMP 触发输出，来自外部触发源“或”内部触发源

29.3.3. TAMP 寄存器写保护

系统复位后，对 TAMP 寄存器的访问被禁止，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 TAMP 寄存器的访问：

- 通过设置寄存器 RCC_APBENR1 的 PWREN 来打开电源管理模块的时钟
- 置位电源控制寄存器 PWR_CR1 的 DBP 位来使能对 TAMP 寄存器的访问。

29.3.4. 入侵检测

入侵检测可配置用于以下目的：

- 擦除备份寄存器（默认配置）
- 生成中断，能够从停止模式和待机模式唤醒
- 为 LPTIM 生成硬件触发

29.3.4.1. TAMP 备份寄存器

备份寄存器（TAMP_BKPxR）不会通过系统复位来复位，也不会在器件从待机模式唤醒时复位。

备份寄存器在发生入侵检测事件时复位， TAMPxNOER 位置 1 时或者 TAMP_CR2 寄存器中的 TAMPxMSK 置 1 时除外。

注意： 当 Flash 的读取保护从 1 级更改为 0 级时，备份寄存器也将被擦除。

29.3.4.2. 入侵检测初始化

可通过将 TAMP_CR1 寄存器中相应的 TAMPxE 位置 1 来使能各输入。

每个 TAMP_INx 入侵检测输入与 TAMP_SR 寄存器中的标志 TAMPxF 相关联。

将 TAMPxMSK 清零时：

TAMPxF 标志将在引脚上出现入侵事件后使能，并存在下述延迟：

- 当 TAMPFLT 不为 0x0 时（带过滤的电平检测），延迟为 TAMPFLT 配置的 RTCCLK 周期数
- 当 TAMPFLT = 0x0（边沿检测），无延迟

在此期间，只要 TAMPxF 置 1，在软件清零前就无法检测到同一引脚上出现的新入侵。

将 TAMPxMSK 置 1 时：

在上述延迟期间以及在 3 个 RTCCLK 附加周期内，无法检测到同一引脚上出现的新入侵。

通过将 TAMP_IER 寄存器中的 TAMPxIE 位置 1，可在发生入侵检测事件时（当 TAMPxF 置 1 时）生成中断。当相应的 TAMPxMSK 置 1 时，不允许将 TAMPxIE 置 1。

29.3.4.3. 出现入侵事件时生成触发输出

LPTIM 可将入侵事件检测用作触发输入。

将 TAMP_CR2 寄存器中的 TAMPxMSK 位清零时，必须通过软件将 TAMPxF 标志清零以便在同一引脚上检测新入侵。

将 TAMPxMSK 位置 1 时，TAMPxF 标志会被屏蔽，并在 TAMP_SR 寄存器中保持清零。此配置允许在停止模式下自动触发 LPTIM，无需通过系统唤醒来执行 TAMPxF 清零。在这种情况下，备份寄存器不清零。

29.3.4.4. 入侵事件的时间戳

当 RTC_CR 中的 TAMPTS 置 1 时，任何入侵事件都会导致时间戳事件发生。在这种情况下，如同发生正常时间戳事件一样， RTC_SR 中的 TSF 位或 TSOVF 位会置 1。在 RTC_SR 中的 TSF 或 TSOVF 置 1 的同时， TAMP_SR 中受影响的入侵标志寄存器 TAMPxF 也会随之置 1。

29.3.4.5. 对入侵输入的边沿检测（被动模式）

如果 TAMPFLT 位设置为 00，当相应的 TAMPxTRG 位上出现上升沿或下降沿时，TAMP_INx 引脚将生成入侵检测事件。

注意：使用边沿检测时，建议在使能入侵检测后（通过读取 GPIO 寄存器）以及向备份寄存器中写入敏感值前立即通过软件检查入侵引脚电平，以确保使能入侵事件检测后才出现有效边沿。

当 TAMPFLT = 00 且 TAMPxTRG = 0（上升沿检测）时，如果在使能入侵检测前入侵输入已处于高电平，则可通过硬件来检测入侵事件。

检测到入侵事件并清零后，应当在重新编程备份寄存器 (TAMP_BKPxR) 之前禁止 TAMP_INx，然后再重新使能（TAMPxEN 置 1）。这可防止应用程序在 TAMP_INx 输入值仍指示入侵检测时，对备份寄存器执行写操作。这相当于对 TAMP_INx 输入的电平检测。

注意： 当 V_{CC} 电源关闭时，入侵检测仍有效。要避免意外复位备份寄存器，应将 TAMPx 映射的引脚从外部连接到正确的电平。

29.3.4.6. 对入侵输入的带过滤电平检测（被动模式）

通过将 TAMPFLT 设置为非零值可执行带过滤的电平检测。在 TAMPxTRG 位指定的电平连续出现 2、 4 或 8 个（取决于 TAMPFLT 值）采样时生成入侵检测事件。

29.4. 低功耗模式

表 29-2 低功耗模式对 TAMP 的影响

Mode	Description
睡眠	功能无影响. TAMP 中断（使能后）会退出睡眠模式.
停止	对所有功能均无影响，但带过滤的电平检测模式除外，该模式仅在时钟源为 LSE 或 LSI 时保持激活状态。 入侵事件会退出停止模式.
待机	对所有功能均无影响，但带过滤的电平检测模式除外，该模式仅在时钟源为 LSE 或 LSI 时保持激活状态。 入侵事件会退出待机模式.

29.5. TAMP 中断

在中断状态寄存器中设置中断通道。中断输出也会被激活。

表 29-3 中断请求

中断事件	事件标志	使能控制位	中断清除	退出睡眠模式	退出停止和待机模式
外部入侵 x	TAMP_SR.TAMP1F	TAMP1IE	在 CTAMP1F 中写 1	支持	支持注 1
内部入侵	TAMP_SR.ITAMP1F	ITAMP1IE	在 CITAMP1F 中写 1	支持	支持注 1

注 1：在带过滤的电平检测被动入侵模式下，只有当 TAMP 时钟源为 LSE 或 LSI 时，才可以从停止和待机模式唤醒。

29.6. TAMP 寄存器

29.6.1. TAMP 控制寄存器 1 (TAMP_CR1)

地址偏移：0x00

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1EN
															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1EN
															RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	ITAMP1EN	RW	0	内部入侵 1 使能： RTC 日历溢出 0：禁止内部入侵 1 1：使能内部入侵 1 在 RTC 日历达到其最大值时生成入侵。日历随后会冻结，不会溢出。
15:1	Reserved	-	-	保留
0	TAMP1EN	RW	0	TAMP_IN1 的入侵检测使能 0：禁止 TAMP_IN 的入侵检测。 1：使能 TAMP_IN 的入侵检测。

29.6.2. TAMP 控制寄存器 2 (TAMP_CR2)

地址偏移：0x04

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1TR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1MSK
							RW								RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1NOER
															RW

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	TAMP1TRG	RW	0	外部入侵 1 输入的有效电平（禁止工作模式） 如果 TAMPFLT ≠ 00： 0：入侵 1 输入保持低电平会触发入侵检测事件。 1：入侵 1 输入保持高电平会触发入侵检测事件。 如果 TAMPFLT = 00： 0：入侵 1 输入上升沿和高电平会触发入侵检测事件。 1：入侵 1 输入下降沿和低电平会触发入侵检测事件。
23:17	Reserved	-	-	保留
16	TAMP1MSK	RW	0	入侵 1 屏蔽 0：入侵 1 事件生成触发事件，必须通过软件将 TAMP1F 清零来允许下一次入侵事件检测。 1：入侵 1 事件生成触发事件。屏蔽 TAMP1F 并由硬件在内部将其清零。不擦除备份寄存器。 TAMP1MSK 置 1 时，不得使能入侵 1 中断。
15:1	Reserved	-	-	保留
0	TAMP1NOER	RW	0	TAMP_IN1 入侵不擦除备份寄存器 0：入侵 1 事件擦除备份寄存器。 1：入侵 1 事件不擦除备份寄存器。

29.6.3. TAMP 滤波控制寄存器 (TAMP_FLTCR)

地址偏移：0x0C

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMPFLT[1:0]		TAMPFREQ[2:0]		
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4:3	TAMPFLT[1:0]	RW	0	TAMP_IN 滤波器计数 这些位决定了为激活入侵事件需要在指定电平 (TAMP1TRG) 上的连续采样次数。 TAMPFLT 对每个 TAMP_IN 输入都有效。 0x0：在 TAMP_IN 输入转变为有效电平的边沿激活入侵事件（TAMP_IN 输入上无内部上拉）。 0x1：在有效电平上连续执行 2 次采样后激活入侵事件。 0x2：在有效电平上连续执行 4 次采样后激活入侵事件。 0x3：在有效电平上连续执行 8 次采样后激活入侵事件。
2:0	TAMPFREQ[2:0]	RW	0	入侵采样频率 这些位决定了对每个 TAMP_INx 输入进行采样时的频率。 0x0： RTCCLK/32768（RTCCLK = 32768 Hz 时为 1 Hz） 0x1： RTCCLK/16384（RTCCLK = 32768 Hz 时为 2 Hz） 0x2： RTCCLK/8192（RTCCLK = 32768 Hz 时为 4 Hz） 0x3： RTCCLK/4096（RTCCLK = 32768 Hz 时为 8 Hz） 0x4： RTCCLK/2048（RTCCLK = 32768 Hz 时为 16 Hz） 0x5： RTCCLK/1024（RTCCLK = 32768 Hz 时为 32 Hz） 0x6： RTCCLK/512（RTCCLK = 32768 Hz 时为 64 Hz） 0x7： RTCCLK/256（RTCCLK = 32768 Hz 时为 128 Hz）

29.6.4. TAMP 中断使能寄存器 (TAMP_IER)

地址偏移：0x2C

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1IE
															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1IE
															RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	ITAMP1IE	RW	0	内部入侵 1 中断使能 0：禁止入侵 1 中断。 1：使能入侵 1 中断。
15:1	Reserved	-	-	保留

0	TAMP1IE	RW	0	外部入侵 1 中断使能 0：禁止入侵 1 中断。 1：使能入侵 1 中断。

29.6.5. TAMP 状态寄存器 (TAMP_SR)

地址偏移：0x30

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1F
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1F
															R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	ITAMP1F	R	0	RTC 日历溢出入侵检测标志 0：未检测到 RTC 日历溢出 1：检测到 RTC 日历溢出
15:1	Reserved	-	-	保留
0	TAMP1F	R	0	外部入侵 1 检测标志 0：未检测到外部入侵 1 1：检测到外部入侵 1

29.6.6. TAMP 屏蔽中断状态寄存器 (TAMP_MISR)

地址偏移：0x34

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ITAMP1MF
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAMP1MF
															R

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	ITAMP1MF	R	0	RTC 日历溢出入侵检测中断 0：未检测到 RTC 日历溢出中断 1：检测到 RTC 日历溢出中断 当 RTC 日历溢出入侵检测标志产生，且对应中断使能时，硬件置位该寄存器。
15:1	Reserved	-	-	保留
0	TAMP1MF	R	0	外部入侵 1 检测中断 0：未检测到外部入侵 1 中断 1：检测到外部入侵 1 中断 当外部入侵 1 检测标志产生，且对应中断使能时，硬件置位该寄存器。

29.6.7. TAMP 状态清零寄存器 (TAMP_SCR)

地址偏移：0x3C

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CITAMP1F
															W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CTAMP1F
															W

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	CITAMP1F	W	0	TAMP_SR 中 RTC 日历溢出入侵检测标志清零
15:1	Reserved	-	-	保留
0	CTAMP1F	W	0	TAMP_SR 中外部入侵 1 检测标志清零

29.6.8. TAMP 备份 x 寄存器 (TAMP_BKPxR)

地址偏移：0x100+0x04*x,(x=0~7)

RTC 域复位值： 0x0000_0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BKP[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BKP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	BKP[31:0]	RW	0	应用可向/从这些寄存器写入/读取数据。 当 V_{CC} 关闭时，这些寄存器由 V_{DDK} 供电，因而系统复位时，这些寄存器不会复位，并且当器件在低功耗模式下工作时，寄存器的内容仍然有效。 在默认配置中，此寄存器在发生入侵检测事件时复位。只要至少一个内部或外部入侵标志置 1，就会将此寄存器强制为复位值。 禁止读取保护 (RDP) 时，此寄存器也会复位。

30. 内部集成电路接口(I²C)

30.1. I²C 简介

I²C(内部集成电路)总线接口连接微控制器和串行 I²C 总线间的通信。它提供多主机功能，控制所有 I²C 总线特定的顺序、协议、仲裁和时序。支持标准（Sm）、快速（Fm）、快速模式增强（Fm+）。

它还与 SMBus（系统管理总线）和 PMBus（电源管理总线）兼容。

根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

I²C 接口可配置支持停止和睡眠低功耗模式唤醒（地址匹配）。

本产品包含 2 个 I²C，2 个 I²C 的功能完全相同。

30.2. I²C 主要特性

- 从机和主机模式
- 支持不同通讯速度
  - 标准模式（Sm）：高达 100 kHz
  - 快速模式（Fm）：高达 400 kHz
  - 快速模式增强（Fm+）：高达 1 MHz
- 作为主机
  - 产生时钟
  - 产生开始位（Start）和停止位（Stop）
- 作为从机
  - 可编程的 I²C 地址检测(1 个可掩码配置)
  - 可响应 2 个从地址的双地址能力
  - 停止位检测
- 7 位/10 位寻址模式
- 支持广播呼叫（General call）
- 状态标志位
  - 发送/接收模式标志位
  - 字节传输完成标志位
  - I²C 忙标志位
- 错误标志位
  - 主模式仲裁丢失
  - 地址/数据传输后的 ACK 错误
  - 起始/停止位错误
  - 过载/欠载 (时钟延长功能禁止)
- 可选的时钟延长功能
- 具备 DMA 能力的单字节缓冲区
- 软件复位
- 模拟噪声滤波功能
- 可配置的 PEC（数据包错误校验）产生和验证
  - PEC 值可以在 Tx 模式下的最后一个字节发送
  - 接收最后字节做 PEC 错误检查

- 支持 SMBus
  - SCL 低电平超时延时 25 ms
  - 主机累积 SCL 低电平扩展时间 10 ms
  - 从机累积 SCL 低电平扩展时间 25 ms
  - 带 ACK 控制的硬件 PEC 产生/校验
  - 支持地址解析协议（ARP）
  - 超时和空闲条件检测
- 支持 PMBus
- 支持地址匹配时从停止低功耗模式唤醒
  - 支持可配置的超时时间

30.3. I²C 功能描述

30.3.1. I²C 框图

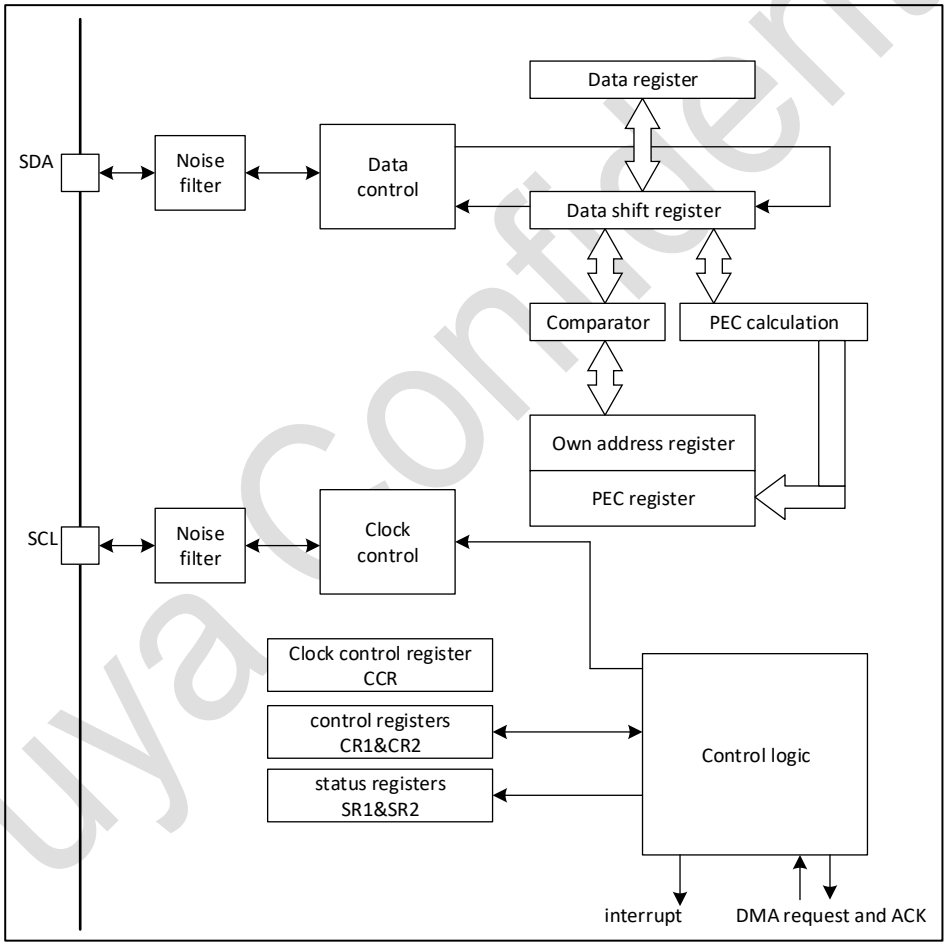


图 30-1 I²C 框图

30.3.2. 模式选择

I²C 支持以下四种模式：

- 从机发送
- 从机接收
- 主机发送
- 主机接收

默认工作在从机模式。接口在生成起始条件后，自动地从从机模式切换到主机模式。

30.3.2.1. 通讯流程

作为主机，I²C 接口启动数据传输，并产生时钟信号。串行数据的传输总是以起始位开始，并以停止位结束。起始位和停止位都是在主机模式下由软件控制产生。

作为从机，I²C 接口能识别自己的地址(7 位或 10 位)和广播呼叫地址。软件能够控制开启或禁止对广播呼叫地址的识别。

数据和地址按 8 位（字节）进行传输，MSB 在前。跟在起始位后的 1 或 2 个字节是地址(7 位地址模式为 1 个字节，10 位地址模式为 2 个字节)。地址只在主机模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收方必须回送一个应答位(ACK)给发送方。参见下图。

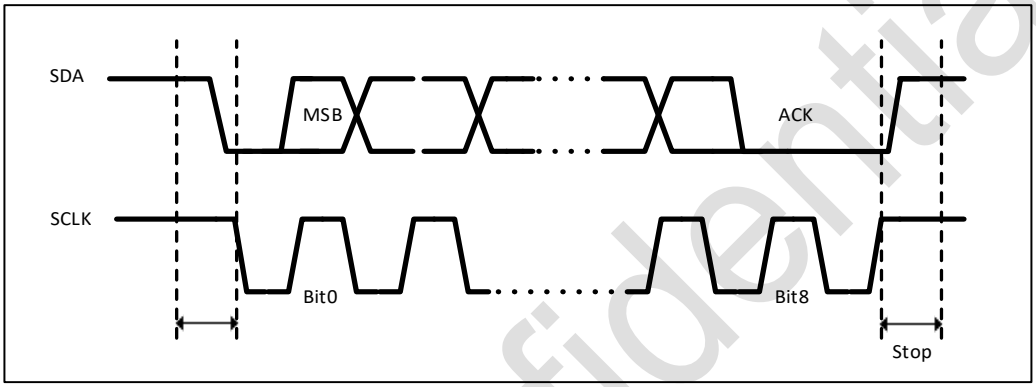


图 30-2 I2C 总线协议

软件可使能或禁止应答位（ACK）。软件也可以选择 I²C 接口地址（7 位/10 位双寻址与/或广播呼叫地址）。

30.3.3. I²C 初始化

30.3.3.1. 使能和禁止外设

I²C 的时钟模块先要通过 RCC_APBENR1 寄存器的 I2CxEN 位打开，然后通过设置 I2C_CR1 的 PE 位为 1 使能 I²C 模块。

30.3.3.2. 模拟噪声滤波器

通过将 I2C_CR1 寄存器中的 PE 位置 1 来使能 I²C 外设之前，如有必要，用户必须配置噪声滤波器。默认情况下，SDA 和 SCL 输入上集成了模拟噪声滤波器。该模拟滤波器符合 I²C 规范，此规范要求在快速模式和快速模式加强下滤除一定脉冲宽度的噪声（滤波宽度参见数据手册）。用户可通过将 SYSCFG_IOCFG.Px_EIIC 位置 1 来使能该模拟滤波器。

30.3.3.3. I²C 时序

为满足在主机和从机模式下，准确的数据保持和建立时间，需要进行 I2C 时序的设定。这是通过写 I2C_CCR 和 I2C_TRISE 寄存器实现的。

30.3.4. I²C 从模式

默认情况下，I²C 接口总是工作在从机模式。从从机模式切换到主机模式，需要产生一个起始条件。

为了产生正确的时序，必须在 I2C_CR2 寄存器中设定该模块的输入时钟。输入时钟的频率必须至少是:

标准模式下为：4 MHz

快速模式下为：8 MHz

快速模式增强下为：16 MHz

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，并与芯片的地址 OAR1 或者广播呼叫地址(如果 ENGC=1)相比较。

如果启用了额外地址（I2C_OAR2.ENDUAL=1），地址会和芯片的 OAR2 地址进行比较，并且 OAR2 寄存器中的地址可以通过 OA2MSK[2:0]进行屏蔽，最多可以屏蔽 7 个 OA2 地址。因此，当 OA2MSK 配置为 1 到 6 时，将分别只有 OA2[7:2]、OA2[7:3]、OA2[7:4]、OA2[7:5]、OA2[7:6]或 OA2[7]与接收到的地址作比较。只要 OA2MSK 不等于 0，OA2 的地址比较器便会排除 I2C 保留地址（0000 XXX 和 1111 XXX），这些地址将不会得到应答。如果 OA2MSK=7，接收到的所有 7 位地址（保留地址除外）均得到应答。

头段或地址不匹配：

I²C 接口将其忽略并等待另一个起始条件。

地址匹配：

I²C 接口产生以下时序和状态：

- 如果 I2C_CR1.ACK 被软件置'1'，则产生一个应答脉冲
- 硬件置位 I2C_SR1.ADDR 位，如果设置了 ITEVTEN 位，则产生中断
- 在从机模式下，I2C_SR2.TRA 位指示当前是处于接收器模式还是发送器模式。

30.3.4.1. 从模式发送

在接收到地址并清除 I2C_SR1.ADDR 位后，（如果地址字节的最低位是 1）从机将数据（字节）从 I2C_DR 寄存器，经由内部移位寄存器发送到 SDA 上。

从机拉低 SCL，直到 I2C_SR1.ADDR 位被清除，并且待发送数据已写入 I2C_DR 寄存器（参考 EV1、EV3）。

当收到应答脉冲时：TxE 位被硬件置位，如果设置了 ITEVTEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 位被置位，但在下一个数据发送结束之前，没有新数据写入到 I2C_DR 寄存器，则 BTF 位被置位。从机拉低 SCL，直到 BTF 位被软件清零（读 I2C_SR1 之后，再写入 I2C_DR 寄存器）。

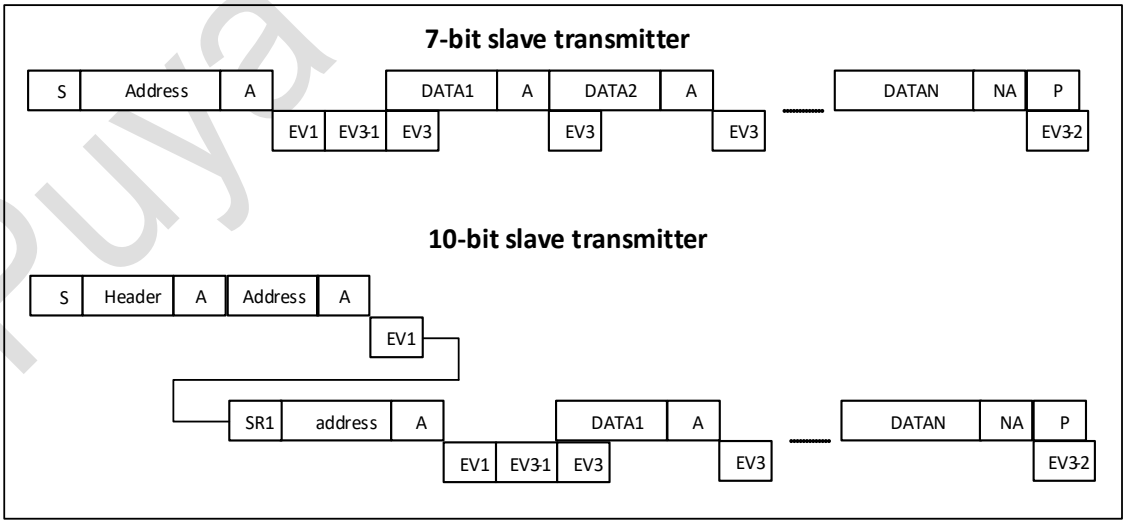


图 30-3 从机发送序列图

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, NA= Non-acknowledge, EVx=Event(with interrupt if ITEVFEN=1)

EV1：ADDR=1，通过先读 I2C_SR1 寄存器，再读 I2C_SR2 寄存器清零 ADDR 位

EV3-1：TxE=1，移位寄存器空，数据寄存器空，向 I2C_DR 寄存器写 DATA1

EV3：TxE=1，移位寄存器非空，数据寄存器空，向 I2C_DR 寄存器写（DATA2）清零 TxE

EV3-2：AF=1；软件向 AF 位写 0 清零该位

30.3.4.2. 从模式接收

在接收到地址并清除 ADDR 后，（如果地址字节的最低位是 0）从机将通过内部移位寄存器把从 SDA 线接收到的字节存进 DR 寄存器。I²C 接口在接收到每个字节后都执行下列操作：

- 如果设置了 ACK 位，则产生一个应答脉冲
- 硬件设置 RxNE=1。如果设置了 ITEVTEN 和 ITBUFEN 位，则产生一个中断。

如果 RxNE 被置位，并且在接收新的数据结束之前，DR 寄存器未被读出，则 BTF 位被置位，在清除 BTF（读出 I2C_SR1 之后再读 I2C_DR 寄存器）之前，从机一直拉低 SCL。(见下图)。

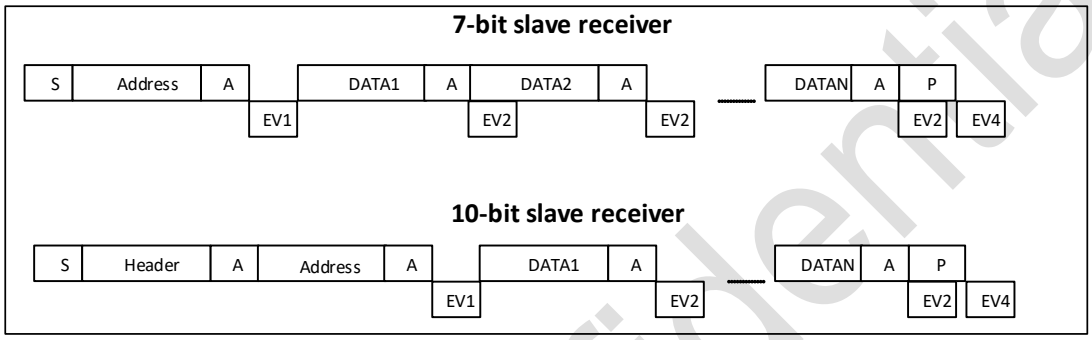


图 30-4 从机接收序列图

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledged, EVx= Event(with interrupt if ITEVFEN=1)

EV1：ADDR=1，通过先读 I2C_SR1，后读 I2C_SR2，实现 ADDR 的清零

EV2：RxNE=1，读 I2C_DR 寄存器清零该位

EV4：STOPF=1，通过先读 I2C_SR1 寄存器，后写 I2C_CR1 寄存器实现对该位的清零。

注意:

1) EV1 事件拉低 SCL，直到相应软件序列结束。

2) EV2 软件序列必须在当前字节传输完成前已完成。

当用户检查 I2C_SR1 寄存器内容后，应该对每个发现置位的标志位，进行完整的清除序列操作。比如 ADDR 和 STOPF 标志位，需要用以下操作序列：

--如果 ADDR=1，先读 I2C_SR1，再读 I2C_SR2；如果 STOPF=1，先读 I2C_SR1，再写 I2C_CR1。

这样做的目的是确保如果 ADDR 和 STOPF 两位都被发现置位，都能被清除掉。

30.3.4.3. 关闭从机通讯

在传输完最后一个数据字节后，主机产生一个停止位，从机检测到该停止位时：

- 硬件置位 STOPF，如果设置了 ITEVTEN 位，则产生一个中断。
- 通过先读 I2C_SR1，后写 I2C_CR1，实现对 STOPF 位的清零。（参见上图的 EV4）

30.3.5. I²C 主模式

在主机模式时，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位开始，并以停止位结束。

当通过 START 位在总线上产生了起始条件，设备就进入了主机模式。

以下是主机模式所要求的操作顺序：

- 在 I2C_CR2 寄存器中设定该模块的输入时钟以产生正确的时序
- 配置时钟控制寄存器
- 配置上升时间寄存器
- 编程 I2C_CR1 寄存器启动外设
- 置 I2C_CR1 寄存器中的 START 位为 1，产生起始条件

I^2C 模块的输入时钟频率必须至少是：

- 标准模式下为：4 MHz
- 快速模式下为：8 MHz
- 快速模式增强下为：16 MHz

30.3.5.1. SCL 主时钟产生

I2C_CCR 寄存器以上升沿或者下降沿，产生 SCL 的高电平和低电平。由于从机可能拉长 SCL 信号，在 SCL 上升沿产生后，主机在 TRISE 寄存器编程的时间到达时，检查来自总线的 SCL 信号。

- 如果 SCL 是低电平，意味着从机正在拉长 SCL 总线，高电平计数器停止计数，直到 SCL 被检测到高电平。这是为了确保 SCL 参数的最小高电平时间。
- 如果 SCL 是高电平，高电平计数器保持计数。

实际上，即使从机不拉长 SCK，从 SCL 上升沿产生，到 SCL 上升沿被检测到，这样的反馈环路也是要花费些时间的。这个回路的时间与 SCL 的上升时间（SCL 的 VIH 数据检测），SCL 输入路径的模拟噪声滤波，以及芯片内部由于用 APB 时钟进行的 SCL 同步有关系。通过配置 TRISE 寄存器，使得无论 SCL 上升时间如何，SCL 的频率保持稳定。

30.3.5.2. 起始位

当 I2C_SR2.BUSY=0 时，设置 I2C_CR1.START=1，I2C 接口将产生一个起始条件，并切换至主机模式(I2C_SR2.MSL 被置位)。

注意：在主机模式下设置 START 位，将在当前字节传输完后，由硬件产生一个重复起始位（ReStart）。

一旦发出起始位：

- I2C_SR1.SB 位被硬件置位，如果设置了 ITEVTEN 位，则会产生一个中断。
- 主机读 I2C_SR1 寄存器，再把从机地址写入 I2C_DR 寄存器。（EV5）

30.3.5.3. 从机地址传输

从机地址通过内部移位寄存器被送到 SDA 线上。

- 在 10 位地址模式时，发送一个头序列产生以下事件：
  - ADD10 位被硬件置位，如果设置了 ITEVTEN 位，则产生一个中断。
  - 随后软件读 I2C_SR1 寄存器，接着读 I2C_SR2 寄存器。
  - ADDR 位被硬件置位，如果设置了 ITEVTEN 位，则产生一个中断。
  - 随后软件读 I2C_SR1 寄存器，接着读 I2C_SR2 寄存器。
- 在 7 位地址模式时，送出一个地址字节。

该地址字节一旦被送出，

  - ADDR 位被硬件置位，如果设置了 ITEVTEN 位，则产生一个中断。
  - 随后软件读 I2C_SR1 寄存器，接着读 I2C_SR2 寄存器。
  - 根据送出从机地址的最低位，软件决定进入发送器模式，还是进入接收器模式。
- 在 7 位地址模式时，
  - 要进入发送器模式，主机发送从地址时置最低位为'0'。

- 要进入接收器模式，主机发送从地址时置最低位为'1'。

■ 在 10 位地址模式时，

- 要进入发送器模式，主机先发送头字节（11110xx0），然后发送最低位为 0 的从地址（头字节中的 xx 值 10 位地址中的最高 2 位）
- 要进入接收器模式，主机先发送头字节（11110xx0），然后发送最低位为 0 的从地址。然后再重新发送一个开始条件，后面跟着头字节（11110xx1）（头字节中的 xx 值 10 位地址中的最高 2 位）.

■ TRA 位指示主机是在接收器模式还是发送器模式。

30.3.5.4. 主机发送

在发送了地址和清除了 ADDR 位后，主机通过内部移位寄存器将字节从 I2C_DR 寄存器发送到 SDA 线上。

主机等待，直到第一个数据字节被写入 I2C_DR 寄存器（参见 EV8_1）。

当收到 ACK 脉冲时，TxE 位被硬件置位，如果设置了 INEVFEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 被置位，且在上一次数据发送结束之前，没有写新的数据字节到 I2C_DR 寄存器，则 BTF 被硬件置位。在清除 BTF（先读 I2C_SR1 之后，再写 I2C_DR 寄存器）之前，I2C 接口将保持 SCL 为低电平。

关闭通讯

在 I2C_DR 寄存器中写入最后一个字节后，通过设置 STOP 位产生一个停止条件(见图的 EV8_2)，然后 I2C 接口将自动回到从模式(MSL 位清除)。

注：当 TxE 或 BTF 位置位时，停止条件应安排在出现 EV8_2 事件时。

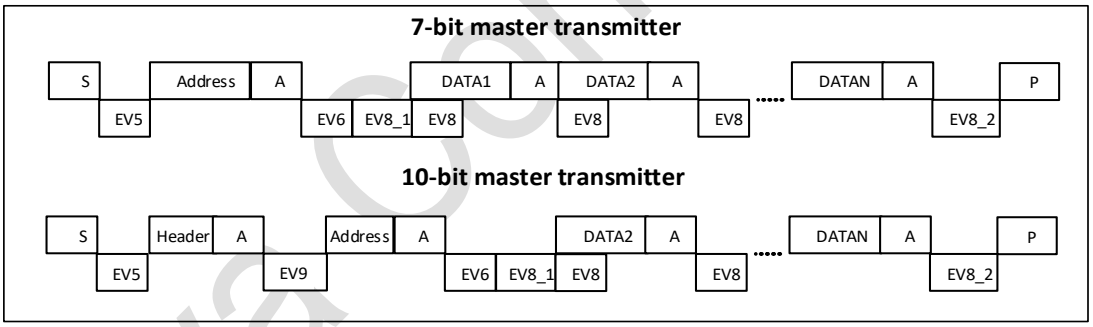


图 30-5 主机发送序列图

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，通过读 I2C_SR1，再向 I2C_DR 寄存器写数据，实现对该位的清零

EV6：ADDR=1，通过先读 I2C_SR1，再读 I2C_SR2，实现对该位的清零

EV8_1：TxE=1，移位寄存器空，数据寄存器空，向 I2C_DR 寄存器写 DATA1

EV8：TxE=1，移位寄存器非空，数据寄存器空，向 I2C_DR 寄存器写 DATA2，该位被清零

EV8_2：TxE=1, BTF=1，写停止位寄存器（STOP），当硬件发出停止位时，TxE 和 BTF 被清零

EV9：ADD10=1，读 I2C_SR1 然后写 I2C_DR 寄存器清除该事件。

注意:

1） EV5, EV6, EV8_1 和 EV8_2 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

2） EV8 软件序列必须在当前字节发送完成前执行完毕。在 EV8 的软件序列不能在当前传输的字节结束前被完成，推荐使用 BTF 代替 TxE，但这样会导致通讯减慢。

30.3.5.5. 主机接收

在发送地址和清除 ADDR 之后，I²C 接口进入主机接收模式。在此模式下，I²C 接口从 SDA 线接收数据字节，并通过内部移位寄存器送至 I2C_DR 寄存器。在每个字节后，I²C 接口依次执行以下操作：

- 如果 ACK 位被置位，发出一个应答脉冲。
- 硬件设置 RxNE=1，如果设置了 INEVFEN 和 ITBUFEN 位，则会产生一个中断。
- 如果 RxNE 位被置位，并且在接收新数据结束前，I2C_DR 寄存器中的数据没有被读走，硬件将设置 BTF=1，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再读出 I2C_DR 寄存器将清除 BTF 位。

30.3.5.6. 关闭通讯

方法 1:当 I²C 被设成应用程序中最高优先级的中断

主机在从从机接收到最后一个字节后，发送一个 NACK。接收到 NACK 后，从机释放对 SCL 和 SDA 线的控制。主机就可以发送一个停止/重复起始位。

- 为了在收到最后一个字节后产生一个 NACK 脉冲，在读倒数第二个数据字节之后(在倒数第二个 RxNE 事件之后)必须清除 ACK 位。
- 为了产生一个停止/重复起始位，软件必须在读倒数第二个数据字节之后(在倒数第二个 RxNE 事件之后)设置停止/起始位。
- 只接收一个字节时，刚好在 EV6 之后(EV6_1 时，清除 ADDR 之后)要关闭应答和停止条件的产生位。在产生了停止位后，I²C 接口自动回到从模式(MSL 位被清除)。

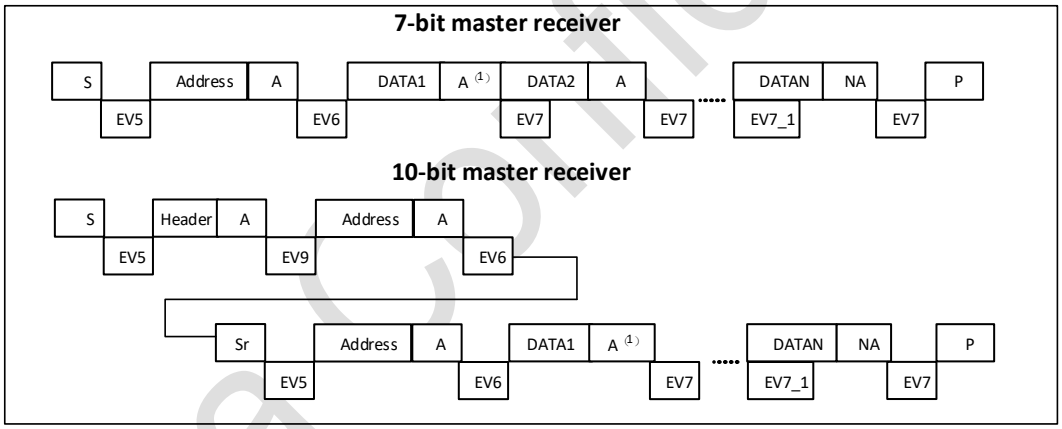


图 30-6 方法 1：主机接收时的传输序列

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，读 I2C_SR1，再写 I2C_DR 寄存器，该位被清零

EV6：ADDR=1，读 I2C_SR1，再读 I2C_SR2，该位被清零

EV6_1：无相关的标志事件，仅用作 1 个字节的接收。

EV7：RxNE=1，读 I2C_DR 寄存器，该位被清零

EV7_1：RxNE=1，读 I2C_DR 寄存器，写 ACK=0 并置位 STOP

EV9：ADD10=1，读 I2C_SR1 然后写 I2C_DR 寄存器清除该事件。

注意：

1） 如果是单个字节接收，则上述标注为（1）的地方会是 NA

2） EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

3） EV7 软件序列必须在当前字节发送完成前执行完毕。在 EV7，软件序列不能在当前传输的字节传输完成前被完成。推荐使用 BTF 代替 RxNE，但这样会导致通讯减慢。

4） EV6_1 或者 EV7_1 的软件序列必须在当前字节传输的 ACK 之前完成。

方法 2: I²C 的中断在应用中不是最高优先级，或者使用查询方式

用这个方法，DATA N-2 没有被读，因此在 DATA N-1 之后，通讯被拉长（RxNE 和 BTF 都被置位）。然后，在读 I2C_DR 寄存器的 DATA N-2 前，清 ACK 位，以确保在 DATA N 的 ACK 之前被清掉。在此之后，在读 DATA N-2 之后，置位停止/起始位，并读 DATA N-1。在 RxNE 置位后，读 DATA N。

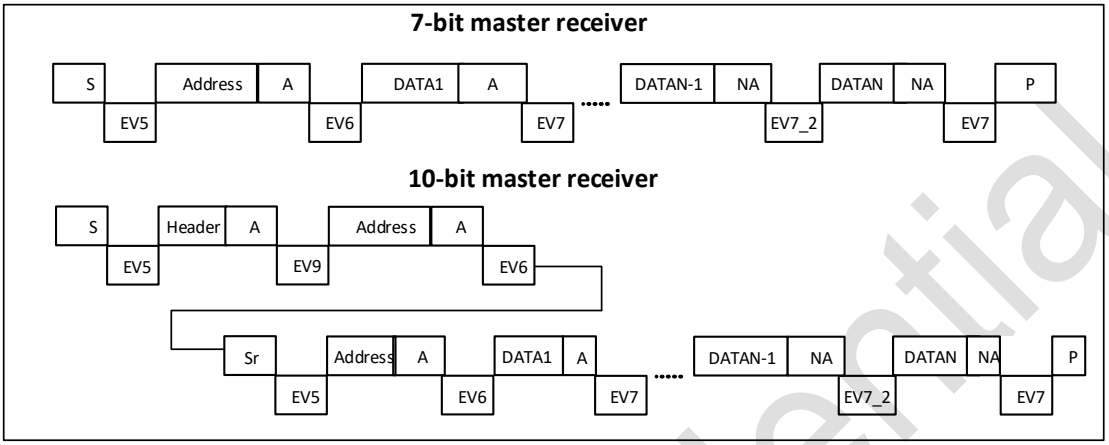


图 30-7 方法 2: 主机接收传输序列 N>2

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，先读 I2C_SR1 寄存器，再写 I2C_DR 寄存器，清零该位

EV6：ADDR，先读 I2C_SR1，再读 I2C_SR2，清零该位

EV7：RxNE=1，读 I2C_DR 寄存器清零该位

EV7_2：BTF=1, DATA N-2 存在 I2C_DR 寄存器中，DATA N-1 存在移位寄存器中，写 ACK=0，读 I2C_DR 寄存器中的 DATA N-2。置位 STOP，读 DATA N-1

EV9：ADD10=1，读 I2C_SR1 然后写 I2C_DR 寄存器清除该事件。

注意：

1） EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

2） EV7 软件序列必须在当前字节发送完成前执行完毕。在 EV7，软件序列不能在当前传输的字节传输完成前被完成。推荐使用 BTF 代替 RxNE，但这样会导致通讯减慢。

- 当 3 个字节要被读走：
  - RxNE=1 = > 无操作(DATA N-2 未读)。
  - 接收 DATA N-1
  - BTF=1，移位和数据寄存器满：I2C_DR 寄存器存放了 DATA N-2，移位寄存器存放了 DATA N-1 = > SCL 拉低：总线上没有其他要被接收的数据
  - 清零 ACK 位
  - 读 I2C_DR 寄存器中的 DATA N-2 = > 这将启动移位寄存器对 DATA N 的接收
  - DATA N 接收完成（回复 NACK）
  - 写 START 或者 STOP 位
  - 读 DATA N-1
  - RxNE=1
  - 读 DATA N

以上流程是针对 N > 2 的描述。1 个字节和 2 个字节的接收，要用不同的处理方式，参见以下描述：

- 2 个字节接收的情况
  - 软件置位 POS 和 ACK 位
  - 等待硬件置位 ADDR
  - 软件清零 ADDR 位
  - 软件清零 ACK 位
  - 等待硬件置位 BTF
  - 软件写 STOP 位
  - 软件读 I2C_DR 两次

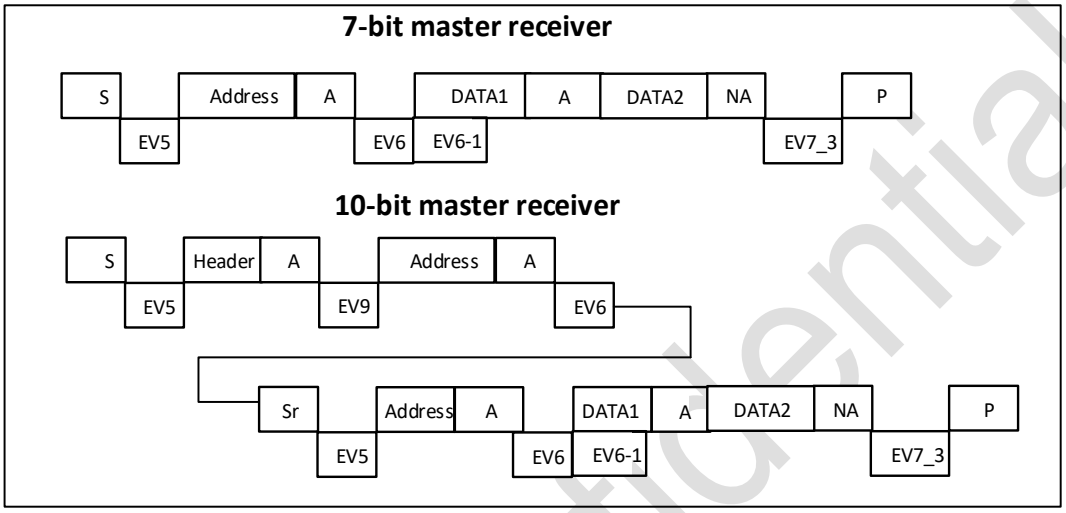


图 30-8 方法 2: 主机接收时的传输时序 N=2

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，先读 I2C_SR1 寄存器，再写 I2C_DR 寄存器，清零该位

EV6：ADDR=1，先读 I2C_SR1 寄存器，后读 I2C_SR2 寄存器，清零 ADDR 位

EV6_1：无相关的标志位事件。在 EV6 后，也就是地址被清零后，ACK 应该被清零

EV7_3：BTF=1，写 STOP=1，之后读两次 I2C_DR（DATA1 和 DATA2）

EV9：ADD10=1，读 I2C_SR1 然后写 I2C_DR 寄存器清除该事件。

注意:

1） EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

2） EV6_1 的软件序列必须在当前字节传输的 ACK 之前完成

- 单个字节接收的情况时，软件操作
  - 在 ADDR 事件里，清零 ACK 位
  - 清零 ADDR
  - 写 STOP 或者 START 位
  - 在 RxNE 标志置位后，读数据

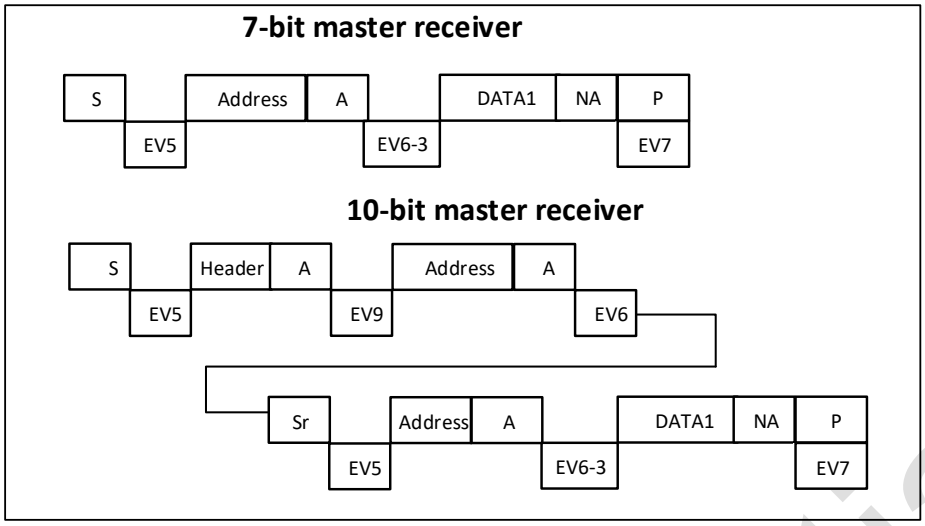


图 30-9 方法 2: 主机接收时的传输时序 N=1

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，先读 I2C_SR1 寄存器，再写 I2C_DR 寄存器，清零该位

EV6_3：ADDR=1，写 ACK=0。先读 I2C_SR1 寄存器，后读 I2C_SR2 寄存器，清零 ADDR 位。在 ADDR 被清零后，写 STOP=1

EV7：RxNE=1，读 I2C_DR 寄存器清零该位

EV9：ADD10=1，读 I2C_SR1 然后写 I2C_DR 寄存器清除该事件。

注意:

EV5, EV6, EV8_1 和 EV8_2 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束。

30.3.6. 错误条件

30.3.6.1. 总线错误

在一个地址或数据字节传输期间，当 I²C 接口检测到一个外部的停止或起始位则产生总线错误。此时：

- 置位 BERR 位，如果设置了 ITERREN 位，则产生一个中断；
- 在从机模式：数据被丢弃，硬件释放总线：
  - 如果是错误的起始位，从机认为是一个重复起始位，并等待地址或停止位
  - 如果是错误的停止位，从机按正常的停止位操作，同时硬件释放总线
- 在主机模式：硬件不释放总线，同时不影响当前的传输状态。此时由软件决定是否要中止当前的传输。

30.3.6.2. 应答错误 (AF)

当接口检测到一个无应答位（NACK）时，产生应答错误。此时：

- AF 位被置位，如果设置了 ITERREN 位，则产生一个中断
- 当发送器接收到一个 NACK 时，必须复位通讯：
  - 如果是处于从机模式，硬件释放总线。
  - 如果是处于主机模式，软件必须生成一个停止位或者重复起始位。

30.3.6.3. 仲裁丢失(ARLO)

当 I²C 接口检测到仲裁丢失时产生仲裁丢失错误，此时：

- ARLO 位被硬件置位，如果设置了 ITERREN 位，则产生一个中断

- I²C 接口自动回到从模式(MSL 位被清除)。当 I²C 接口丢失了仲裁，则它无法在同一个传输中响应它的从地址，但它可以在赢得总线的主机发送重复起始位之后响应
- 硬件释放总线

30.3.6.4. 上溢/下溢错误 (OVR)

在从机模式下，如果禁止时钟延长，I²C 接口正在接收数据时，当它已经接收到一个字节(RxNE=1)，但在 I2C_DR 寄存器中前一个字节数据还没有被读出，则发生上溢错误。

此时：

- 最后接收的数据被丢弃
- 在上溢错误时，软件应清除 RxNE 位，发送器应该重新发送最后一个字节

在从机模式下，如果禁止时钟延长，I²C 接口正在发送数据时，在下一个字节的时钟到达之前，新的数据还未写入 I2C_DR 寄存器(TxE=1)，则发生下溢错误。此时：

- 在 I2C_DR 寄存器中的前一个字节将被重复发出
- 用户应该确定在发生下溢时，接收端应丢弃重复接收到的数据。发送端应按 I²C 总线标准在规定的时间更新 I2C_DR 寄存器

在发送第一个字节时，必须在清除 ADDR 之后且在第一个 SCL 上升沿之前写入 I2C_DR 寄存器；如果不能做到这点，则接收方应该丢弃第一个数据。

30.3.7. SDA/SCL 线控制

- 如果允许时钟延长：
  - 发送模式：如果 TxE=1 且 BTF=1：I²C 接口在传输前保持时钟线为低，以等待软件读取 I2C_SR1，然后把数据写进数据寄存器 I2C_DR(I2C_DR 和移位寄存器都是空的)。
  - 接收模式：如果 RxNE=1 且 BTF=1：I²C 接口在接收到数据字节后保持时钟线为低，以等待软件读 I2C_SR1，然后读数据寄存器 I2C_DR(DR 和移位寄存器都是满的)。
- 如果在从机模式中禁止时钟延长：
  - 如果 RxNE=1，在接收到下个字节前 I2C_DR 还没有被读出，则发生上溢。接收到的最后一个字节丢失。
  - 如果 TxE=1，在必须发送下个字节之前却没有新数据写进 I2C_DR，则发生下溢。相同的字节将被重复发出。
  - 硬件未实现对写冲突的控制。

30.3.8. DMA 请求

DMA 请求(当被使能时)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变满，则产生 DMA 请求。DMA 必须在当前字节传输结束之前被初始化和使能。DMAEN 位（I2C_CR2 寄存器中）必须在 ADDR 事件发生前使能。

在主机或者从机模式，当时钟延展功能使能，DMAEN 位可以在清零 ADDR 之前的 ADDR 事件期间置位。DMA 请求必须在当前字节传输完成之前响应。当 DMA 传输数据长度达到 DMA 设定的值时，DMA 控制器向 I²C 发送 EOT（传输结束），并产生传输完成中断（如果中断使能位有效）：

- 主机发送：在 EOT 中断服务程序中，需禁止 DMA 请求，然后在等到 BTF 事件后，置位 STOP 位。
- 主机接收：当要接收的数据数目大于或等于 2 时，DMA 控制器发送一个硬件信号 EOT_1，它对应 DMA 传输(字节数 - 1)。如果在 I2C_CR2 寄存器中设置了 LAST 位，硬件在发送完 EOT_1 后

的下一个字节，将自动发送 NACK。在中断允许的情况下，用户可以在 DMA 传输完成的中断服务程序中产生一个停止条件。

30.3.8.1. DMA 发送

通过置位 I2C_CR2 寄存器中的 DMAEN 位，可以使能 DMA 模式。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） 配置 SYSCFG_CFGR3 或 SYSCFG_CFGR4 的 DMAx_MAP 寄存器选择 DMA 通道
2） 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器地址。数据将在每个 TxE 事件后，从存储器传送至这个地址。
3） 在 DMA_CMARx 寄存器中设置存储器地址。数据在每个 TxE 事件后从这个存储区传送至 I2C_DR。
4） 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 TxE 事件后，此值将被递减。
5） 利用 DMA_CCRx 寄存器中的 PL[0:1]位配置通道优先级。
6） 设置 DMA_CCRx 寄存器中的 DIR 位，并根据应用要求可以配置在整个传输完成一半或全部完成时发出中断请求。
7） 通过设置 DMA_CCRx 寄存器上的 EN 位激活通道。

当 DMA 控制器中设置的数据传输数量已经完成时，DMA 控制器给 I²C 接口发送一个代表传输结束的 EOT/EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：如果使用 DMA 进行发送时，不要设置 I2C_CR2 寄存器的 ITBUFEN 位。

30.3.8.2. DMA 接收

通过设置 I2C_CR2 寄存器中的 DMAEN 位可以激活 DMA 接收模式。每次接收到数据字节时，将由 DMA 把 I2C_DR 寄存器的数据传送到设置的存储区(参考 DMA 说明)。设置 DMA 通道进行 I²C 接收，须执行以下步骤(x 是通道号)：

1） 配置 SYSCFG_CFGR3 或 SYSCFG_CFGR4 的 DMAx_MAP 寄存器选择 DMA 通道
2） 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器的地址。数据将在每次 RxNE 事件后从此地址传送到存储区。
3） 在 DMA_CMARx 寄存器中设置存储区地址。数据将在每次 RxNE 事件后从 I2C_DR 寄存器传送到此存储区。
4） 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 RxNE 事件后，此值将被递减。
5） 用 DMA_CCRx 寄存器中的 PL[0:1]配置通道优先级。
6） 清除 DMA_CCRx 寄存器中的 DIR 位，根据应用要求可以设置在数据传输完成一半或全部完成时发出中断请求。
7） 设置 DMA_CCRx 寄存器中的 EN 位激活该通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I2C 接口发送一个代表传输结束的 EOT/EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：如果使用 DMA 进行接收时，不要设置 I2C_CR2 寄存器的 ITBUFEN 位。

30.3.9. 包错误校验

包错误校验(PEC)用于提高通信的可靠性，使用下述 CRC-8 多项式对每一位串行数据进行计算：
C(x) = x8 + x2 + x + 1

- PEC 计算由 I2C_CR1 寄存器的 ENPEC 位激活。PEC 使用 CRC-8 算法对所有信息字节进行计算，包括地址和读/写位在内。

- 在发送时：在最后一个 TxE 事件时设置 I2C_CR1 寄存器的 PEC 传输位，PEC 将在最后一个字节后被发送。
- 在接收时：在最后一个 RxNE 事件之后设置 I2C_CR1 寄存器的 PEC 位，如果下个接收到的字节不等于内部计算的 PEC，接收器发送一个 NACK。如果是主接收器，不管校对的结果如何，PEC 后都将发送 NACK。PEC 位必须在接收当前字节的 ACK 脉冲之前设置。

- 在 I2C_SR1 寄存器中可获得 PECERR 错误标志/中断。
- 如果 DMA 和 PEC 都被激活：
  - 在发送时：当 I²C 接口从 DMA 控制器处接收到 EOT 信号时，它在最后一个字节后自动发送 PEC。
  - 在接收时: 当 I²C 接口从 DMA 处接收到一个 EOT_1 信号时，它将自动把下一个字节作为 PEC，并且校验它。在接收到 PEC 后产生一个 DMA 请求。
- 为了允许中间 PEC 传输，在 I2C_CR2 寄存器中有一个控制位(LAST 位)用于判别是否真是最后一个 DMA 传输。如果确实是最后一个主接收器的 DMA 请求，在接收到最后一个字节后自动发送 NACK。
- 仲裁丢失时 PEC 计算失效。

30.3.10. 系统管理总线 SMBus

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。

该外设与 SMBus 规范兼容 (http://smbus.org)。

系统管理总线规范涉及三类器件：

- 从设备，用于接收或响应命令。
- 主设备，用于发出命令、生成时钟和中止传输。
- 主机，专用的主设备，可提供连接系统 CPU 的主接口。主机必须具有主-从设备功能，且必须支持 SMBus 主机通知协议。系统中只允许存在一个主机。

该外设可配置为主设备或从设备，也可配置为主机。

本项目的 I²C 模块支持 SMbus/PMbus 功能。

SMbus 与 I²C 比较的不同点：

表 30-1 SMbus 与 I2C 不同点

SMBus	I²C
最大传输速度 100 kHz	最大传输速度~1MHz
最小传输速度 10 kHz	无最小传输速度
25ms 时钟低超时	无时钟超时
固定的逻辑电平	逻辑电平由 V_{DDD} 决定
不同的地址类型(保留、动态等)	7 位、10 位和广播呼叫从地址类型
不同的总线协议（快速命令、处理呼叫等)	无总线协议

SMbus 与 I²C 比较的相似点：

- 2 线的总线协议(1 个时钟，1 个数据) + 可选的 SMBus 提醒线
- 主-从通信，主机提供时钟
- SMBus 数据格式类似于 I²C 的 7 位地址格式

30.3.10.1. 总线协议

任何给定器件都有 11 种可用命令协议。器件既可以在这 11 种协议中任选其一，也可以使用全部 11 种协议进行通信。这 11 种协议分别为快速命令、发送字节、接收字节、写入字节、写入字、读取字节、读取字、过程调用、块读取、块写入以及块写入-块读取过程调用。

这些协议应通过用户软件实施。

有关这些协议的详细信息，请参见 SMBus 规范 (http://smbus.org)。

30.3.10.2. 地址解析协议(ARP)

通过给每个从机动态地分配一个新的唯一地址，可以解决 SMBus 的从地址冲突。地址解析协议(ARP)具有以下的特性：

- 使用标准 SMBus 物理层仲裁机制分配地址；
- 当设备维持供电期间，分配的地址仍保持不变，也允许设备在断电后保留其地址。
- 在地址分配后，没有额外的 SMBus 的打包开销(也就是说访问分配地址的设备与访问固定地址的设备所用时间是一样的)；
- 任何一个 SMBus 主机可以遍历总线。

30.3.10.3. SMBus 提醒模式（ALERT）

SMBus 提醒是一个带中断线的可选信号，用于那些希望扩展它们的控制能力而牺牲一个引脚的设备。SMBALERT 和 SCL、SDA 信号一样，是一种线与信号。SMBALERT 通常和 SMBus 广播呼叫地址一起使用。与 SMBus 有关的消息为 2 字节。一个只具有从功能的设备，可以通过设置 I2C_CR1 寄存器上的 ALERT 位，使用 SMBALERT 给主机发信号表示它希望进行通信。主机处理该中断并通过提醒响应地址 ARA(地址值为 0001100x)访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应答 ARA。此状态是由 I2C_SR1 寄存器中的 SMBALERT 状态标志来标识的。主机执行一个修改过的接收字节操作。由从发送设备提供的 7 位设备地址被放在字节的 7 个最高位上，第八个位可以是’0’或’1’。

如果多个设备把 SMBALERT 拉低，最高优先级设备(最小的地址)将在地址传输期间通过标准仲裁赢得通信权。在确认从地址后，此设备不得再拉低它的 SMBALERT，如果当信息传输完成后，主机仍看到 SMBALERT 低，就知道需要再次读 ARA。

没有实现 SMBALERT 信号的主机可以定期访问 ARA。

有关 SMBus 提醒模式的更多详细资料，请参考 2.0 版的 SMBus 规范(http://smbus.org/specs/)。

30.3.10.4. 超时错误（TIMEOUT）

在定时规范上 I^2C 和 SMBus 之间有很多差别。SMBus 定义了一个时钟低超时，25ms 的超时。SMBus 规定 $t_{LOW:SEXT}$ 为从机的累积时钟低扩展时间。SMBus 规定 $T_{LOW:MEXT}$ 为主机的累积时钟低扩展时间。更多超时细节请参考 2.0 版的 SMBus 规范(http://smbus.org/specs/)。

I2C_SR1 中的状态标志 TIMEOUT 或 Tlow 错误表明了这个特性的状态。

将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须在 SMBus 规范规定的时间最大值之前检测出超时情况。

$t_{TIMEOUT}$ 检查

要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0]位编程为定时器重载值，以检查 tTIMEOUT 参数。必须将 TIDLE 位配置为“0”，以检测 SCL 低电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 的低电平持续时间超过（TIMEOUTA+1) x 2048 x t_{PCLK}，I2C_SR1 寄存器的 TIMEOUT 标志将置 1。

（最大 $t_{TIMEOUT}$ = 25 ms）。

注：TIMEOUTEN 位置 1 时，不允许更改 TIMEOUTA[11:0]位和 TIDLE 位的配置。

$t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 检查

必须根据外设配置为主设备还是从设备来配置 TIMEOUTB 定时器，以便为从设备校验 $t_{LOW:SEXT}$，为主器件校验 $t_{LOW:MEXT}$。由于标准只规定了最大值，用户可以为这两个参数选择相同的值。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。

如果 SMBus 外设延展 SCL 的累积时间超过(TIMEOUTB+1) x 2048 x t_{PCLK}，则 I2C_SR1 寄存器中的 TIMEOUT 标志将置 1。

总线空闲检测

要使能 t_{IDLE} 检查，必须将 12 位 TIMEOUTA[11:0]字段编程为定时器重载值，以获取 t_{IDLE} 参数。必须将 TIDLE 位配置为“1”，以检测 SCL 和 SDA 高电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过(TIMEOUTA+1) x 4 x t_{PCLK}，I2C_SR1 寄存器中的 TIMEOUT 标志将置 1。

注：TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

30.3.10.5. 如何使用 SMBus 模式的接口

为了从 I^2C 模式切换到 SMBus 模式，应该执行下列步骤：

- 设置 I2C_CR1 寄存器中的 SMBUS 位；
- 按应用要求配置 I2C_CR1 寄存器中的 SMBTYPE 和 ENARP 位。

如果要把设备配置成主机，产生起始条件的步骤见“I2C 主模式”章节。否则，参见“I2C 从模式”章节。

软件程序必须处理多种 SMBus 协议。

- 如果 ENARP=1 且 SMBTYPE=0，使用 SMBus 设备默认地址。
- 如果 ENARP=1 且 SMBTYPE=1，使用 SMBus 主机头字段。
- 如果 SMBALERT=1，使用 SMBus 提醒响应地址

30.3.10.6. PMbus

PMbus 是基于 SMbus 而来的，传输逻辑与 SMbus 完全一致，区别在于 PMbus 定义了一些与电源管理相关的功能（由软件完成）。

30.3.10.7. SMBus: I2C_TIMEOUTR 寄存器配置示例

仅当支持 SMBus 功能时，才涉及本节内容。

将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms

表 30-2 不同 PCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms）

f_{PCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	$t_{TIMEOUT}$
8 MHz	0x61	0	1	98 x 2048 x 125 ns = 25 ms
16 MHz	0xC3	0	1	196 x 2048 x 62.5 ns = 25 ms
32 MHz	0x186	0	1	391 x 2048 x 31.25 ns = 25 ms
48 MHz	0x249	0	1	586 x 2048 x 20.08 ns = 25 ms

将 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的最大持续时间配置为 8 ms

表 30-3 不同 PCLK 频率下的 TIMEOUTB 设置示例

f_{PCLK}	TIMEOUTB[11:0]位	TEXTEN 位	$t_{LOW:EXT}$
8 MHz	0x1F	1	32 x 2048 x 125 ns = 8 ms
16 MHz	0x3F	1	64 x 2048 x 62.5 ns = 8 ms
48 MHz	0xBB	1	188 x 2048 x 20.08 ns = 8 ms

将 t_{IDLE} 的最大持续时间配置为 50 µs

表 30-4 不同 PCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs）

f_{PCLK}	TIMEOUTB[11:0]位	TIDLE 位	TIMEOUTEN 位	T_{IDLE}
8 MHz	0x63	1	1	100 x 4 x 125 ns = 50 us
16 MHz	0xC7	1	1	64 x 4 x 62.5 ns = 50 us
48 MHz	0x257	1	1	600 x 4 x 20.08 ns = 50 us

SMBus 超时规范

表 30-5 超时规范

符号	参数	限值		单位
		最小值	最大值	
$t_{TIMEOUT}$	检测时钟低电平超时	25	35	ms
$t_{LOW:SEXT}$(1)	累积时钟低电平延长时间（从设备）	-	25	ms
$t_{LOW:MEXT}$(2)	累积时钟低电平延长时间（主设备）	-	10	ms
t_{IDLE}	SCL 和 SDA 高电平超时	-	50	us

1） $t_{LOW:SEXT}$ 是一段累积时间，即给定从设备在一条消息的最初起始到停止期间时钟信号可延展的时间。其他从设备或主设备也可能延长时钟，进而导致时钟低电平总延长时间超过 $t_{LOW:SEXT}$。因此，测量该参数时该设备应该是全速主设备寻址的唯一设备。

2） $t_{LOW:MEXT}$ 是一段累积时间，即主设备在消息的每个字节（定义为 START 到 ACK、ACK 到 ACK 或 ACK 到 STOP）内时钟信号可延展的时间。从设备或其他主设备也可能延长时钟，进而导致时钟低电平总时间超过 $t_{LOW:MEXT}$（针对给定字节）。因此，测量该参数时该全速主设备只寻址一个从设备。

30.4. I²C 中断

表 30-6 I²C 中断请求

中断事件	事件标志	开启控制位
起始位已发送(主机)	SB	ITEVTEN
地址已发送(主机) 或地址匹配(从机)	ADDR	
10 位地址头序列已发送(主)	ADD10	
已收到停止(从机)	STOPF	
数据字节传输完成	BTF	
接收缓冲区非空	RxNE	ITEVTEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失(主机)	ARLO	
响应失败	AF	
上溢/下溢	OVR	
PEC 错误	PECERR	

超时/T_{LOW} 错误	TIMEOUT	
SMBus 提醒	SMBALERT	

30.5. I²C 调试模式

当微控制器进入调试模式时，根据 DBGMCU 模块中的 DBG_I2Cx_SMBUS_TIMEOUT 配置位，SMBUS 超时控制或者继续正常工作或者停止。

30.6. I²C 低功耗

在 MCU 处于停止低功耗模式下，可以通过 I²C 接收地址，若地址匹配则可以唤醒 MCU，反之则维持之前低功耗状态（以仅 I²C 唤醒为例）。

1）实现以上功能需要在进入停止低功耗模式之前使能 I2C_CR1 的 PE 和 WUPEN，并且在 RCC 中使能 I²C 的时钟。

2）停止低功耗模式唤醒过程还支持可配置的超时时间，可通过设置 I2C_CR1 的 WKUP_CNT 和 WKUP_DIV 进行配置，若使能 ITERREN，则在产生超时之后唤醒 MCU，若不使能 ITERREN，则在产生超时之后维持原先状态。

30.7. I²C IO 配置

用作 I²C 的 IO，必须要在 GPIO 寄存器中被配置成复用，开漏输出，上拉模式，并且需要使能 SYSCFG_IOCFG.P*_EIIC 寄存器的对应位。

30.8. I²C 寄存器

30.8.1. I²C 控制寄存器 1 (I2C_CR1)

偏移地址:0x00

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WKUP_CNT[1:0]		WKUP_DIV[1:0]	
												RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	WUPEN	SMBUS	PE
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:18	WKUP_CNT[1:0]	RW	2'b10	低功耗唤醒超时计数 00：2 01：8 10：32 11：128 注：超时时间的计算公式为 (WKUP_DIV/f_{PCLK})*WKUP_CNT 例：WKUP_DIV 选择 2'b11 即 1024，f_{PCLK} 为 8M，WKUP_CNT 选择 2'b 00 即 2，则超时时间为 (1024/8)*2=256us 注：f_{PCLK} 为 72M 时，超时时间为 3.5us-1817.6us；f_{PCLK} 为 4M 时，超时时间为 64us-32768us

17:16	WKUP_DIV[1:0]	RW	2’b0	PCLK 的分频系数，分频后的时钟用于计数所有停止低功耗模式唤醒时的超时时间。 00：128 01：256 10：512 11：1024
15	SWRST	RW	0	软件复位 当被置位时，I²C 处于复位状态。在复位释放前，要确保 I²C 的引脚被释放，总线是空闲状态。 0：I²C 模块不处于复位状态 1：I²C 模块处于复位状态 注：该位可以用于错误或锁定状态时重新初始化 I²C。如 BUSY 位为 1，在总线上又没有检测到停止位时。
14	Reserved	-	-	保留
13	ALERT	RW	0	SMBus 提醒 0：将 SMBA 引脚释放为高电平并禁止提醒响应地址头：0001100x 后跟 NACK； 1：将 SMBA 引脚释放为高电平并使能提醒响应地址头：0001100x 后跟 ACK； PE=0 时，由硬件清除。 注：该寄存器如果不支持 SMBus 功能，则固定为 0.
12	PEC	RW	0	数据包出错校验 软件可置位和清零该位，硬件可以在传送 PEC 后、接收到起始位、或者接收到停止位、或者当 PE=0 时清零该位； 0：无 PEC 传输 1：PEC 传输（在发送或接收模式） 注：仲裁丢失时，PEC 的计算失效。
11	POS	RW	0	ACK/PEC 位置（用于数据接收） 软件可置位/清零该寄存器，或 PE=0 时由硬件清零。 0：ACK 位控制当前移位寄存器内正在接收的字节的(N)ACK。PEC 位表明当前移位寄存器内的字节是 PEC 1：ACK 位控制在移位寄存器里接收的下一个字节的(N)ACK。PEC 位表明在移位寄存器里接收的下一个字节是 PEC 注：POS 位只能用在 2 字节的接收配置中，必须在接收数据之前配置。 为了 NACK 第 2 个字节，必须在清除 ADDR 之后清除 ACK 位。 为了检测第 2 个字节的 PEC，必须在配置了 POS 位之后，ADDR 延展事件时设置 PEC 位。
10	ACK	RW	0	应答使能 软件可置位/清零该寄存器，或 PE=0 时由硬件清零。 0：无应答返回 1：在接收到一个字节后返回一个应答。（匹配的地址或数据）

9	STOP	RW	0	产生停止位 软件可以置位/清零该寄存器，或者当检测到停止位时，由硬件清除；当检测到超时错误时，硬件置位。 在主模式下： 0：无停止位产生 1：在当前字节传输完成后或在当前起始位发出后产生停止位 在从模式下： 0：无停止位产生 1：在当前字节传输完成后释放 SCL 和 SDA 线
8	START	RW	0	起始位产生 软件可置位/清零该寄存器，或当起始位发出后或 PE=0 时由硬件清零。 主模式： 0：无起始位产生 1：产生起始位/重复起始位 从模式： 0：无起始位产生 1：当总线空闲时，产生起始位（并由硬件自动切换到主机模式）
7	NOSTRETCH	RW	0	禁止时钟延长（从机） 当 ADDR 或 BTF 标志被置位时，该位用于从机禁止时钟延长，直到被软件复位。 0：允许时钟延长 1：禁止时钟延长
6	ENGC	RW	0	广播呼叫使能 0：禁止广播呼叫。以 NACK 响应地址 00h 1：允许广播呼叫。以 ACK 响应地址 00h
5	ENPEC	RW	0	PEC 使能 0：禁止 PEC 计算 1：开启 PEC 计算
4	ENARP	RW	0	ARP 使能 0：禁止 ARP 1：使能 ARP 如果 SMBTYPE=0，使用 SMBus 设备的默认地址； 如果 SMBTYPE=1，使用 SMBus 的主地址。 注：该寄存器如果不支持 SMBus 功能，则固定为 0.
3	SMBTYPE	RW	0	SMBus 类型 0：SMBus 设备 1：SMBus 主机 注：该寄存器如果不支持 SMBus 功能，则固定为 0.
2	WUPEN	RW	0	停止低功耗模式唤醒使能（从机） 0：禁止停止低功耗模式唤醒 1：使能停止低功耗模式唤醒 注：如果不支持停止低功耗模式唤醒功能，该位保留并由硬件强制为零。
1	SMBUS	RW	0	SMBus 模式

				0：I²C 模式 1：SMBus 模式 注：该寄存器如果不支持 SMBus 功能，则固定为 0.
0	PE	RW	0	I²C 模块使能 0：禁止 1：I²C 使能。相应的 I/O 口需配置为复用功能。 注：如果清除该位时通讯正在进行，在当前通讯结束后，I²C 模块被禁用并返回空闲状态。 由于在通讯结束后 PE=0，所有的位被清除。 在主机模式下，通讯结束之前，绝不能清除该位。

注：当设置了 STOP/START/PEC 位，在硬件清除这些位之前，软件不要执行任何对 I2C_CR1 的写操作；否则有可能会第 2 次设置 STOP/START/PEC 位。

30.8.2. I²C 控制寄存器 2 (I2C_CR2)

偏移地址:0x04

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Res.	FREQ[6:0]						
			RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	LAST	RW	0	DMA 最后一次传输。 0：下一次 DMA 的 EOT 不是最后的传输 1：下一次 DMA 的 EOT 是最后的传输 注：该位在主机接收模式使用，以便在最后一次接收数据时可以产生一个 NACK。
11	DMAEN	RW	0	DMA 请求使能。 0：禁止 DMA 请求 1：当 TxE=1 或 RxNE=1 时，允许 DMA 请求。
10	ITBUFEN	RW	0	缓冲器中断使能。 0：当 TxE=1 或 RxNE=1 时，不产生中断 1：当 TxE=1 或 RxNE=1 时，产生中断（不管 DMAEN 是何值）
9	ITEVTEN	RW	0	事件中断使能。 0：禁止 1：允许事件中断 在下列条件下，将产生该中断： ■ SB=1（主机模式） ■ ADDR=1（主/从模式） ■ ADD10=1（主机模式） ■ STOPF=1（从机模式） ■ BTF=1，但没有 TxE 或 RxNE 事件 ■ 如果 ITBUFFEN=1，TxE 事件为 1 ■ 如果 ITBUFEN=1，RxNE 事件为 1
8	ITERREN	RW	0	出错中断使能。

				0：禁止出错中断； 1：使能出错中断； 在下列条件下，将产生该中断： ■ BERR=1 ■ ARLO=1 ■ AF=1 ■ OVR=1 ■ PECERR=1 ■ TIMEOUT=1 ■ SMBALERT=1
7	Reserved	-	-	保留
6:0	FREQ	RW	0	I²C 模块时钟频率。 必须用支持的 APB 时钟频率的值配置该寄存器，以产生与 I²C 协议兼容的数据建立和保持时间。 允许可设定的最小频率是 4MHz（100k）、8MHz（400k），16MHz（1MHz）最大频率是芯片最高的 APB 时钟频率。 0000000：禁止 0000001：禁止 0000010：禁止 0000011：禁止 0000100：4MHz ….. 0100100：36MHz ….. 0110000：48MHz ….. 1001000：72MHz 大于 1001000：禁止。

30.8.3. I²C 自身地址寄存器 1 (I2C_OAR1)

偏移地址:0x08

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDMODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ADDMODE	RW	0	寻址模式（从模式）。 0：7 位从地址（不响应 10 位地址）； 1：10 位从地址（不响应 7 位地址）； 注：该寄存器如果不支持 10 位地址功能，则固定为 0.
14:10	Reserved	-	-	保留
9:8	ADD[9:8]	RW	0	接口地址。 7 位地址模式该寄存器无效。 10 位地址模式位地址的 9~8 位。 注：该寄存器如果不支持 10 位地址功能，则固定为 0.

7:1	ADD[7:1]	RW	0	接口地址的 7~1 位。
0	ADD0	RW	0	接口地址。 7 位地址模式该寄存器无效。 10 位地址模式位地址的 0 位。 注：该寄存器如果不支持 10 位地址功能，则固定为 0.

30.8.4. I²C 自身地址寄存器 2 (I2C_OAR2)

注意：该寄存器如果不支持双字节地址功能，则固定为 0.

偏移地址:0x0C

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	OA2MSK[2:0]			ADD2[7:1]							ENDUAL
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:8	OA2MSK[2:0]	RW	0	ADD2 屏蔽配置。 000：不屏蔽； 001：屏蔽 ADD2[1]，只比较 ADD2[7:2]； 010：屏蔽 ADD2[2:1]，只比较 ADD2[7:3]； 011：屏蔽 ADD2[3:1]，只比较 ADD2[7:4]； 100：屏蔽 ADD2[4:1]，只比较 ADD2[7:5]； 101：屏蔽 ADD2[5:1]，只比较 ADD2[7:6]； 110：屏蔽 ADD2[6:1]，只比较 ADD2[7]； 111：屏蔽 ADD2[7:1]，不进行比较，对接收到的全部 7 位地址（保留位除外）应答。 注：当 OA2MSK 不为 0 时，即使比较匹配，也不会对保留的 I²C 地址 （ 0b0000xxx 和 0b1111xxx）应答。
7:1	ADD2[7:1]	RW	0	接口地址的 7~1 位。 双地址模式下地址的 7~1 位。
0	ENDUAL	RW	0	双地址模式使能位。 0：在 7 位地址模式下，只有 OAR1 被识别； 1：在 7 位地址模式下，OAR1 和 OAR2 都被识别。

30.8.5. I²C 数据寄存器 (I2C_DR)

偏移地址:0x10

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留

7:0	DR[7:0]	RW	0	8 位数据寄存器，芯片内部实际是两个独立的缓冲区共用一个地址，分别用于存放接收到的数据（RX_DR）、放置要发送到总线的数据（TX_DR）。 **发送器模式：** 当写一个字节至 DR 寄存器时（实际写到 TX_DR），自动启动数据传输。一旦传输开始（TxE=1），如果能及时把下一个需传输的数据写入 DR 寄存器，I²C 模块将保持连续的数据流。 **接收器模式：** 接收到的字节被存储到 DR 寄存器（实际是 RX_DR）（RxNE=1）。在接收到下一个字节（RxNE=1）之前读出数据寄存器，即可实现连续的数据接收。 注： 1) 在从机模式下，地址不会被存储进数据寄存器 DR 2) 硬件不处理写冲突（如果 TxE=0，仍能写入数据寄存器） 3) 如果在处理 ACK 脉冲时发生 ARLO 事件，接收到的字节不会被存储到数据寄存器里，因此不能读到

30.8.6. I²C 状态寄存器（I2C_SR1)

偏移地址:0x14

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMBALERT	TIMEOUT	Res.	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	SMBALERT	RC_W0	0	SMBus 提醒状态。 SMBus 主机模式： 0：无 SMBus 提醒； 1：在 SMBA 引脚产生 SMBus 提醒事件； SMBus 从机模式： 0：没有 SMBA 响应地址头序列； 1：收到 SMBA 响应地址头序列直到 SMBA 引脚变低。 该位由软件写 0 或 PE=0 时由硬件清除。 注：该寄存器如果不支持 SMBUS 功能，则固定为 0.
14	TIMEOUT	RC_W0	0	超时或 Tlow 错误。 0：无超时错误； 1：SCL 为低的持续时间已达到 25ms（超时）；或者主机低电平累计时钟扩展时间超过 10ms（Tlow：mext）；或从机低电平累积时钟扩展时间超过 25ms（Tlow:sext）；或者 Tidle 的时间超时；或者停止低功耗模式唤醒时达到超时时间。 当在从机模式下设置该位：从机复位通讯，硬件释放总线； 当在主机模式下设置该位：硬件发出停止位。

				该位由软件写 0 清除，或当 PE=0 时由硬件清除。 注: 该功能仅在 SMBUS 模式和 I²C 模式唤醒停止低功耗模式时有作用。
13	Reserved	-	-	保留
12	PECERR	RC_W0	0	在接收时发生 PEC 错误。 0：无 PEC 错误，接收到 PEC 后返回 ACK（如果 ACK=1） 1：有 PEC 错误，接收到 PEC 后返回 NACK（不管 ACK 为何值） 该位由软件写 0 清除，或当 PE=0 时由硬件清除。
11	OVR	RC_W0	0	上溢/下溢标志。 0：无上溢/下溢； 1：产生上溢/下溢。 当 NOSTRETCH=1 时，在从模式下该位被硬件置位； 在接收模式中当收到一个新的字节时（包括 ACK 应答脉冲），数据寄存器里的内容还未被读出，则新接收的字节将丢失。 在发送模式中当要发送一个新的字节时，却没有新的数据写入数据寄存器，同样的字节将被发送两次。 该位由软件写 0 清除，或当 PE=0 时由硬件清除。 注：如果数据寄存器的写操作发生时间非常接近 SCL 的上升沿，发送的数据是不确定的，并发生保持时间错误。
10	AF	RC_W0	0	应答失败标志。 0：没有应答失败； 1：应答失败。 当没有返回应答时，硬件将置位该寄存器。 该位由软件写 0 清除，或当 PE=0 时由硬件清除。
9	ARLO	RC_W0	0	仲裁丢失（主机模式）。 0：没有检测到仲裁丢失； 1：检测到仲裁丢失。 当接口失去对总线上另一个主机的控制时，硬件将置位该寄存器。 该位由软件写 0 清除，或在 PE=0 时由硬件清除。 在 ARLO 事件之后，I²C 接口自动切换回从模式（M/SL=0）。 注：在 SMBUS 模式下，在从机模式下对数据的仲裁仅发生在数据解读，或应答传输区间（不包括地址的应答）。
8	BERR	RC_W0	0	总线出错标志。 0：无起始或者停止位出错； 1：起始或者停止位出错。 当接口检测到错误的起始或者停止位，硬件将该位置 1。 该位由软件写 0 清除，或者在 PE=0 时由硬件清除。
7	TxE	R	0	数据寄存器为空（发送时）标志。 0：数据寄存器非空； 1：数据寄存器为空。 在发送数据时，数据寄存器为空时该位被置 1，在发送地址阶段不设置该位。

				软件写数据到 DR 寄存器可清除该位，或在产生一个起始或停止位后，或当 PE=0 时由硬件自动清除。 如果收到一个 NACK，或下一个要发送的字节时 PEC（PEC=1），该位不被置位。 注：在写入第 1 个要发送的数据后，或设置了 BTF 时写入数据，都不能清除 TxE 位，因为此时数据寄存器为空。
6	RxNE	R	0	数据寄存器非空（接收时）标志。 0：数据寄存器为空； 1：数据寄存器非空。 在接收时，当数据寄存器不为空，置位该寄存器。在接收地址阶段，该寄存器不置位。 软件对数据寄存器的读写操作会清除该寄存器，或当 PE=0 时由硬件清除。 注：当设置了 BTF 时，读取数据不能清除 RxNE 位，因为此时数据寄存器仍为满。
5	Reserved	-	-	保留
4	STOPF	R	0	停止条件检测位（从模式）。 0：没有检测到停止位； 1：检测到停止位。 在一个应答之后（如果 ACK=1），当从机在总线上检测到停止位时，硬件将该位置 1。 软件读取 I2C_SR1 寄存器后，对 I2C_CR1 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。 注：在收到 NACK 后，STOPF 位不会置位。
3	ADD10	R	0	10 位地址头序列已发送（主机模式）。 0：没有 ADD10 事件发生。 1：主机已经将第一个地址字节发送出去。 在 10 位地址模式下，当主机已将第一个字节发送出去时，硬件将该位置 1. 软件读取 I2C_SR1 寄存器后，对 I2C_DR 寄存器的写操作将清除该位；或当 PE=0 时，硬件清除该位。 注 1：收到 NACK 后，该寄存器不被置位。 注 2：该寄存器如果不支持 10bit 地址功能，则固定为 0.
2	BTF	R	0	字节传输结束标志位。 0：字节传输未完成 1：字节传输完成 在下列情况下硬件将置位该寄存器（从机模式，NOSTRETCH=0 时；主机模式，与 NOSTRETCH 无关）： — 接收时，当收到一个新字节（包括 ACK 脉冲）且数据寄存器还未被读取（RxNE=1）。 — 发送时，当一个新数据应该被发送，且数据寄存器还未被写入新的数据（TxE=1）。 软件读取 I2C_SR1 寄存器后，对数据寄存器的读或写操作将清除该位；或发送一个起始或停止位后，或当 PE=0 时，由硬件清除。

				注：在收到一个 NACK 后，BTF 位不会被置位。 如果下一个要传输的字节是 PEC（I2C_SR2.TRA=1,I2C_CR1.PEC=1）,BTF 位不会被置位。
1	ADDR	R	0	地址已被发送（主机模式）/地址匹配（从机模式）。 软件读取 I2C_SR1 寄存器后，再读 I2C_SR2 寄存器将清除该位；或当 PE=0 时，由硬件清除。 **地址匹配（从机）：** 0：地址不匹配或没有收到地址； 1：收到的地址匹配。 当收到的从机地址与 OAR 寄存器匹配、或发生广播呼叫、或 SMBus 设备接收到默认地址、或 SMBus 主机识别出 SMBus 提醒时，硬件将置位该位。 **注:** 在从机模式下，推荐进行完整的清零序列，即在 ADDR 被置位后，先读 I2C_SR1 寄存器，再读 I2C_SR2 寄存器。 **地址已发送（主机）：** 0：地址发送没有结束； 1：地址发送结束。 —10 位地址模式时，当收到地址的第二个字节的 ACK 后置位； —7 位地址时，当收到 ACK 字节后置位。 **注:** 在收到 NACK 后，该寄存器不会被置位。
0	SB	R	0	起始位标志（主模式）。 0：未发送起始条件； 1：起始条件已发送； —当发送起始位时，置位该寄存器。 —软件读取 I2C_SR1 寄存器后，对 I2C_DR 寄存器的写操作将清除该位；或当 PE=0 时，由硬件清除。

30.8.7. I²C 状态寄存器 2 (I2C_SR2)

偏移地址:0x18

复位值:0x0000_0000

Note: 即使 ADDR 标志位在读 I2C_SR1 寄存器后被置位，在读 I2C_SR1 之后再读 I2C_SR2 寄存器，也会清零 ADDR 标志位。因此，仅在发现 I2C_SR1 寄存器的 ADDR 位被置位或者 STOPF 位被清零时，I2C_SR2 寄存器才必须被读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PEC[7:0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Res.	TRA	BUSY	MSL
R	R	R	R	R	R	R	R	R	R	R	R		R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:8	PEC[7:0]	R	0	数据包出错校验寄存器。 当 ENPEC=1 时，该寄存器存放内部的 PEC 的值。
7	DUALF	R	0	双地址标志（从机模式）.

				0：接收到的地址与 OAR1 匹配； 1：接收到的地址与 OAR2 匹配。 当产生一个停止位或一个重复的起始位时，或 PE=0 时，硬件清除该寄存器。 注：该寄存器如果不支持双地址功能，则固定为 0.
6	SMBHOST	R	0	接收到 SMBus 主机头序列标志（从机模式）。 0：未收到 SMBus 主机的地址； 1：当 SMBTYPE=1 且 ENARP=1 时，收到 SMBus 主机地址。 当产生一个停止位或一个重复的起始位时，或 PE=0 时，硬件清除该寄存器。 注：该寄存器如果不支持 SMBUS 功能，则固定为 0。
5	SMBDEFAULT	R	0	SMBus 从机默认地址（从机模式）。 0：未收到 SMBus 设备的默认地址； 1：当 ENARP=1 时，收到 SMBus 设备的默认地址。 当产生一个停止位或一个重复的起始位时，或 PE=0 时，硬件清除该寄存器。 注：该寄存器如果不支持 SMBUS 功能，则固定为 0。
4	GENCALL	R	0	广播呼叫地址（从机模式）。 0：未收到广播呼叫地址； 1：当 ENGC=1 时，收到广播呼叫的地址。 当产生一个停止位或一个重复的起始位时，或 PE=0 时，硬件清除该寄存器。
3	Reserved	-	-	保留
2	TRA	R	0	发送/接收标志。 0：接收到数据 1：数据已发送 在整个地址传输阶段的结尾，该寄存器根据地址字节的 R/W 位来设定。 当检测到停止位（STOPF=1），或者重复的起始位、或者总线仲裁丢失（ARLO=1），或当 PE=0 时，硬件清除该寄存器。
1	BUSY	R	0	总线忙标志。 0：在总线上无数据通讯 1：在总线上正在进行数据通讯 当检测到 SDA 或 SCL 为低电平时，硬件置位。 当检测到一个停止位时，硬件清零。 该寄存器指示当前正在进行的总线通讯，当接口被禁用（PE=0）时该寄存器仍然被更新。
0	MSL	R	0	主从模式。 0：从机模式 1：主机模式 —当接口处于主机模式（SB=1）时，硬件置位； —当总线上检测到一个停止位 (STOPF=1)、仲裁丢失（ARLO=1）、或当 PE=0 时，硬件清零。

30.8.8. I²C 时钟控制寄存器(I2C_CCR)

偏移地址:0x1C

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	F+	Res.	CCR[11:0]											
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	F/S	RW	0	I²C 主模式选择。 0：标准模式 1：快速模式
14	DUTY	RW	0	快速模式时的占空比。 0：快速模式下：$T_{low}/T_{high}=2$ 1：快速模式下：$T_{low}/T_{high}=16/9$
13	F+	RW	0	I²C Fm+主模式选择。 0：标准模式或者快速模式，具体选择哪种由 bit15 决定; 1：快速模式增强; 注：在配置该寄存器为 1 时，不关心 bit15 的值;
12	Reserved	-	-	保留
11:0	CCR[11:0]	RW	0	快速/标准模式下的时钟控制分频系数（主机模式）。 该分频系数用于设置主机模式下的 SCL 时钟。 ■ 标准模式或者 SMBus 模式: — T_{high}=CCR x Tpclk — T_{low} =CCR x Tpclk ■ 快速模式: — DUTY=0: – T_{high}=CCR x Tpclk – T_{low} =2 x CCR x Tpclk — DUTY=1(为达到 400 kHz): – Thigh=9 x CCR x Tpclk – Tlow =16 x CCR x Tpclk ■ 快速模式增强时: — DUTY=0: – T_{high}=3*CCR x Tpclk – T_{low} =5 x CCR x Tpclk — DUTY=1(为达到 1 MHz): – Thigh=2 x CCR x Tpclk – Tlow =3 x CCR x Tpclk 注: ■ 允许设定的最小值为 0x04，在快速 DUTY 模式下允许的最小值为 0x01 $T_{high}=t_{r(SCL)}+t_{w(SCLH)}$ $T_{low}=t_{r(SCL)}+t_{w(SCLL)}$ ■ 这些延时没有过滤器 ■ 只有当 PE=0 时才能配置该寄存器 ■ f_{CK} 应当是 10 MHz 的整数倍,这样可以正确产生 400 kHz 的波形

注：bit15和bit13结合起来扩展模式如下表所示:

F+	F/S	模式
0	0	标准模式
0	1	快速模式
1	0	快速模式增强
1	1	快速模式增强

30.8.9. I²C TRISE 寄存器 (I2C_TRISE)

偏移地址:0x20

复位值:0x0000_0082

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	THOLDDATA_SEL	THOLDDATA[4:0]					TRISE[6:0]						
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	THOLDDATA_SEL	RW	0	数据保持时间选择 0：默认硬件计算数据保持时间 1：通过 THLDDATA 配置数据保持时间
11:7	THLDDATA[4:0]	RW	1	在标准模式/快速模式/快速模式增强下的最大的数据保持时间(发送模式)。 这些位是用于数据发送模式下，保证数据保持时间的最小保持时间。 例如：标准模式允许的最大 SDA 下降时间位 300ns。如果在 I2C_CR2 寄存器种 FREQ[6:0]中的值等于 0x08，Tpclk =125 ns，则 TRISE 中配置为 0x03（300ns/125ns = 2.4 +1） 如果结果不为整数，则将整数部分写入 THLDDATA，以确保配置。
6:0	TRISE	RW	0x2	在标准/快速模式下的最大上升时间（主机模式）。 这些位应该提供在主机模式下，SCL 反馈回路的最大持续时间。这样做的目的是无论 SCL 上升沿持续时间多少，SCL 都能保持一个稳定的频率。 这些位必须设置为 I²C 总线规范里给出的最大的 SCL 上升时间，增长步幅为 1。 例如：标准模式中最大允许 SCL 上升时间为 1000ns。如果在 I2C_CR2 寄存器中 FREQ[6:0]中的值等于 0x08，T_{PCLK}=125ns，则 TRISE 中配置为 0x09（1000ns/125ns = 8 + 1 = 9）。 滤波器的值也可以加到 TRISE 内。 如果结果不为整数，则将整数部分写入 TRISE，以确保 t_{HIGH} 参数。 注：当 PE=0 时才能设置该寄存器。

30.8.10. I²C TIMEOUT 寄存器 (I2C_TIMEOUTR)

偏移地址:0x24

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEXTEN	Res.	Res.	Res.	TIMEOUTB[11:0]											
RW				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMEOUTEN	Res.	Res.	TIDLE	TIMEOUTA[11:0]											
RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	TEXTEN	RW	0	时钟信号延展超时使能 0：禁止时钟信号延展超时检测。 1：使能时钟信号延展超时检测。当 I^2C 接口执行 SCL 延展的累积时间超过 LOW:EXT 时，将检测到超时错误 (TIMEOUT=1)。
30:28	Reserved	-	-	保留
27:16	TIMEOUTB[11:0]	RW	0	总线超时 B 该字段用于配置累积时钟延展超时： 在主机模式下，将检测主设备的累积时钟低电平延展时间 ($t_{LOW:MEXT}$) 在从机模式下，将检测从设备的累积时钟低电平延展时间 ($t_{LOW:SEXT}$) $t_{LOW:EXT}$= (TIMEOUTB+1) x 2048 x t_{PCLK} 注：仅可在 TEXTEN=0 时写入这些位
15	TIMEOUTEN	RW	0	时钟超时使能 0：禁止 SCL 超时检测。 1: 使能 SCL 超时检测: 当 SCL 的低电平时间超过 $t_{TIMEOUT}$ (TIDLE=0)，或 SCL 的高电平时间超过 t_{IDLE} (TIDLE=1) 时，将检测到超时错误 (TIMEOUT=1)。
14:13	Reserved	-	-	保留
12	TIDLE	RW	0	空闲时钟超时检测 0：TIMEOUTA 用于检测 SCL 低电平超时 1：TIMEOUTA 用于检测 SCL 和 SDA 高电平超时（总线空闲条件） 注：仅可在 TIMOUTEN=0 时写入该位。
11: 0	TIMEOUTA[11:0]	RW	0	TIMEOUTA[11:0]：总线超时 A 该字段用于配置： – SCL 低电平超时条件 $t_{TIMEOUT}$（当 TIDLE=0 时），$t_{TIMEOUT}$= (TIMEOUTA+1) x 2048 x t_{PCLK} –总线空闲条件，即 SCL 和 SDA 高电平（当 TIDLE=1 时），t_{IDLE}= (TIMEOUTA+1) x 4 x t_{PCLK} 注：仅可在 TIMOUTEN=0 时写入这些位。

31. 通用同步/异步收发器(USART)

本项目设计实现了 2 个 USART 模块，功能完全一致。USART1 和 USART2 均支持 LIN、S7816 和 IrDA。

31.1. USART 简介

通用同步异步收发器(USART) 能够灵活地与外部设备进行全双工数据交换，满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 利用分数波特率发生器提供宽范围的波特率选择。

USART 不仅支持同步单向通信和半双工单线通信，以及 LIN（局域互连网络）、智能卡协议、 IrDA（红外线数据协会）、SIR ENDEC 规范和调制解调器操作（CTS/RTS)，还支持多处理器通信。

使用多缓冲器配置的 DMA 方式，可以实现高速数据通信。

31.2. USART 主要特性

- 全双工异步通信
- NRZ 标准格式
- 可配置 16 倍或者 8 倍过采样，从而在速度容差与时钟容差之间取得最佳平衡
- 发送和接收共用的可编程波特率，最高达 4.5Mbit/s （72MHz，16 倍过采样）
- 自动波特率检测
- 可编程的数据长度 8 位或者 9 位
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 可配置的停止位（0.5/1/1.5/2 位）
- 用于同步通信的同步主/从模式和时钟输出/输入
- 单线半双工通讯
- 通过 DMA 缓冲接收/发送字节
- 独立的发送和接收使能位
- Tx/Rx 引脚可配置交换
- RS232 硬件流控制
- 检测标志
  - 接收缓冲区满
  - 发送缓冲区空
  - 传输结束
- 奇偶校验控制
  - 发送校验位
  - 对接收数据进行校验
- LIN 主模式同步中断发送功能和 LIN 从模式中断检测功能
  - 当 USART 硬件配置成 LIN 时，生成 13 位断开符，检测 10/11 位停止符合
- 正常模式下支持 3/16 位持续时间的 IrDA SIR 编解码器
- 智能卡模式功能
  - 智能卡接口支持 ISO7816-3 标准里定义的智能卡协议，支持 T=0 和 T=1 异步协议
  - 智能卡工作模式下，支持 0.5 和 1.5 个停止位
  - NACK 信号宽度可配置为 1/1.5/2ETU
  - 发送数据支持传输错误重发机制

- 支持 EGT 可设 0~256，并支持超时中断

■ 带标志的中断源

- 发送寄存器空
- CTS 改变
- 发送完成
- 接收寄存器非空
- 溢出错误
- 检测到总线空闲
- 帧错误
- 噪音操作
- 校验错误
- 接收超时
- 块结束
- LIN 断开

■ 多处理器通信

- 如果地址不匹配，则进入静默模式
- 从静默模式唤醒：通过空闲检测和地址标记检测

31.3. USART 功能描述

31.3.1. USART 框图

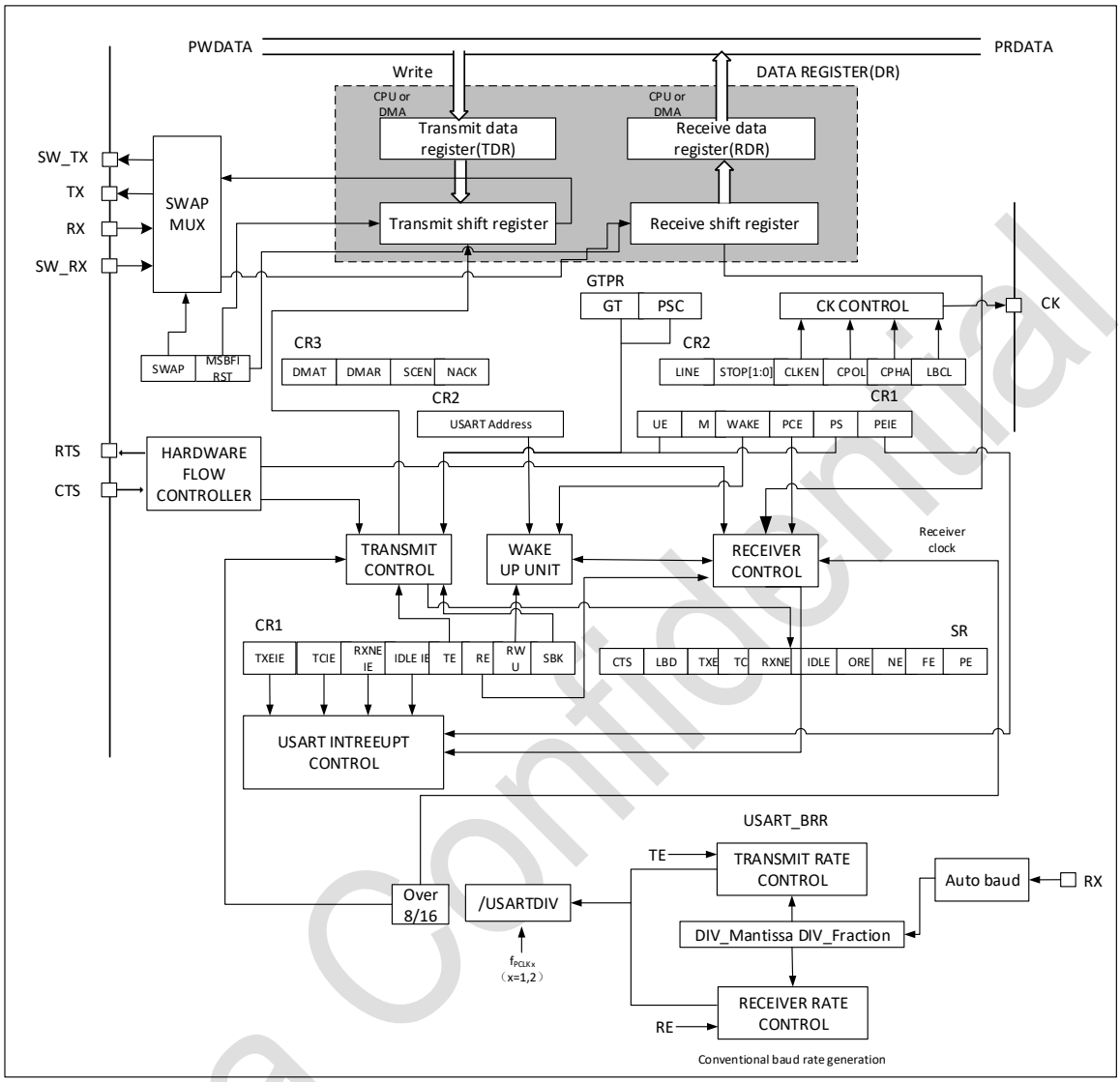


图 31-1 USART 框图

31.3.2. USART 信号

31.3.2.1. 双向通信

USART 双向通信至少需要两个脚：接收数据输入(RX)和发送数据输出(TX)

- RX：接收数据串行输入

采用过采样技术来区别数据和噪音，从而恢复数据。

- TX：发送数据串行输出

当发送器被禁止时，输出引脚恢复到它的 I/O 端口配置。当发送器被激活，并且不发送数据时，TX 引脚处于高电平。在单线和智能卡模式里，此 I/O 口被同时用于数据的发送和接收。

31.3.2.2. RS232 硬件流控制模式

在 RS232 硬件流控制模式下需要以下引脚：

- CTS（清除以发送）

如果驱动为高电平，则该信号用于在当前传输结束时阻止数据发送。

- RTS（请求以发送）

如果为低电平，则该信号用于指示 USART 已准备好接收数据。

31.3.2.3. 同步主模式和智能卡模式

同步主模式和智能卡模式下需要以下引脚：

- CK

该引脚在同步主模式和智能卡模式下用作时钟输出，在同步从模式下用作时钟输入。

在同步主模式下，该引脚用于输出发送器数据时钟，以便按照主机模式进行同步发送(在起始位和结束位上没有时钟脉冲，可通过软件配置向最后一个数据位发送时钟脉冲)。RX 引脚上可以同步接收数据。该机制可用来控制带移位寄存器的外设(例如 LCD 驱动器)。时钟相位和极性都是软件可编程的。

在智能卡模式下，CK 向智能卡输出时钟。

- NSS

该引脚在同步从模式下用作从器件选择输入。

注：NSS 和 CTS 共用同一个引脚。

31.3.3. USART 字符说明

字长可以通过编程 USART_CR1 寄存器中的 M 位，配置成 8 位或 9 位数据。在起始位期间，TX 脚处于低电平，在停止位期间处于高电平。

空闲字符被视为完全由’1’组成的一个完整的数据帧（停止位的电平也为‘1’)。

断开字符被视为在一个帧周期内全部收到’0’(包括停止位期间，也是’0’)。在断开帧结束时，发送器再插入 1 或 2 个停止位(‘1’)来应答起始位。

发送和接收由通用波特率发生器驱动，当发送器和接收器的使能位分别置位时，将分别产生发送时钟和接收时钟。

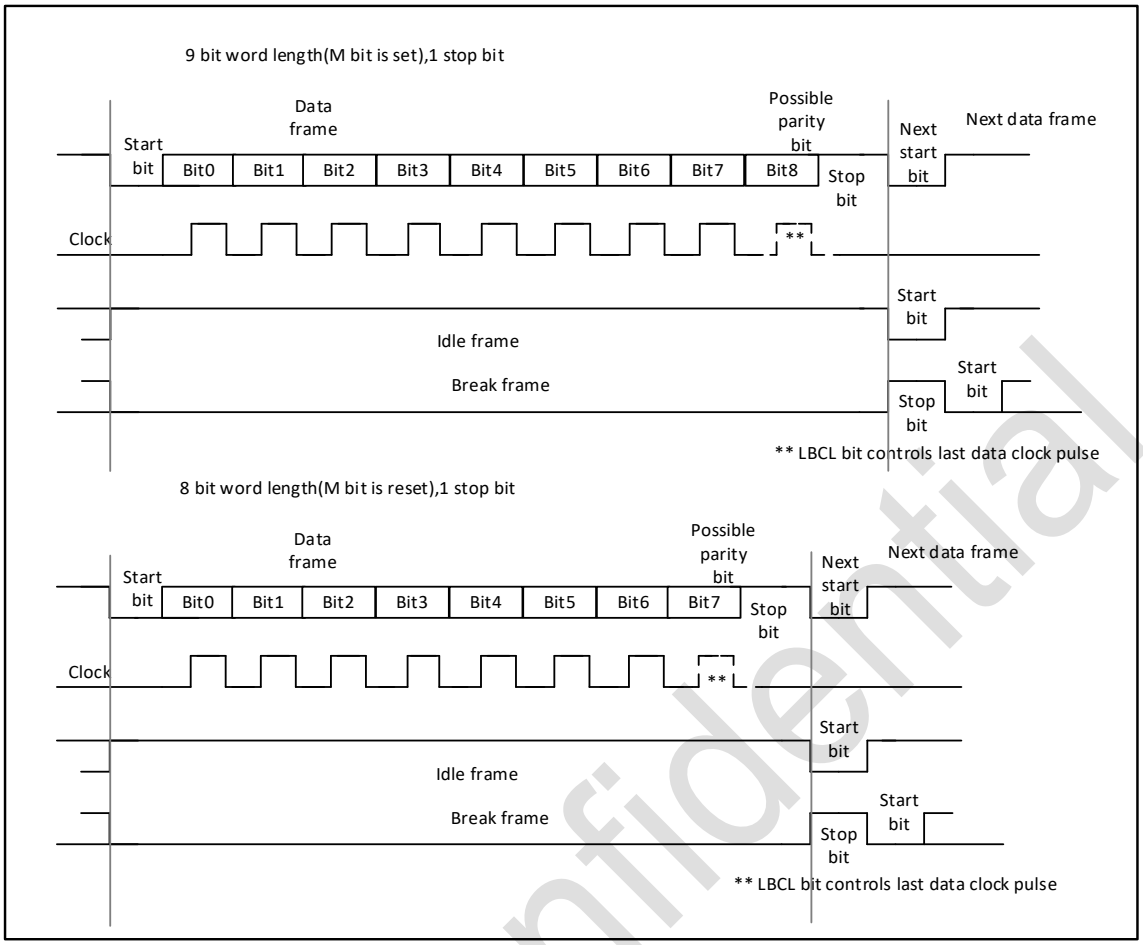


图 31-2 字长编程

31.3.4. USART 发送器

发送器根据 USART_CR1.M 寄存器位的状态发送 8 位或 9 位的数据字。当发送使能位(TE)被设置时，发送移位寄存器中的数据在 TX 脚上输出，相应的时钟脉冲在 CK 脚上输出。

31.3.4.1. 字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式 USART_DR 寄存器包含了一个内部总线和发送移位寄存器之间的缓冲器。

每个字符之前都有一个周期的低电平的起始位；字符由可配置数量的停止位终止。USART 支持多种停止位的配置：0.5，1，1.5 和 2 个停止位。

注意：

在数据传输期间不能复位 TE 位，否则将破坏 TX 脚上的数据，因为波特率计数器停止计数。正在传输的当前数据将丢失。

TE 位被激活后将发送一个空闲帧。

31.3.4.2. 配置停止位

随每个字符发送的停止位的位数可以通过控制寄存器 2 的位 13、12 进行编程。

1) 1 个停止位：停止位位数的默认值。

2) 2 个停止位：可用于常规 USART 模式、单线模式以及调制解调器模式。

3) 0.5 个停止位：用于智能卡模式接收数据时使用。

4）1.5 个停止位：用于智能卡模式发送和接收数据时使用。

空闲帧包括停止位。

断开帧是 10 位低电平，后跟停止位(当 USART_CR1.M=0 时)；或者 11 位低电平，后跟停止位(USART_CR1.M=1 时)。不可能传输更长的断开帧(长度大于 10 或者 11 位)。

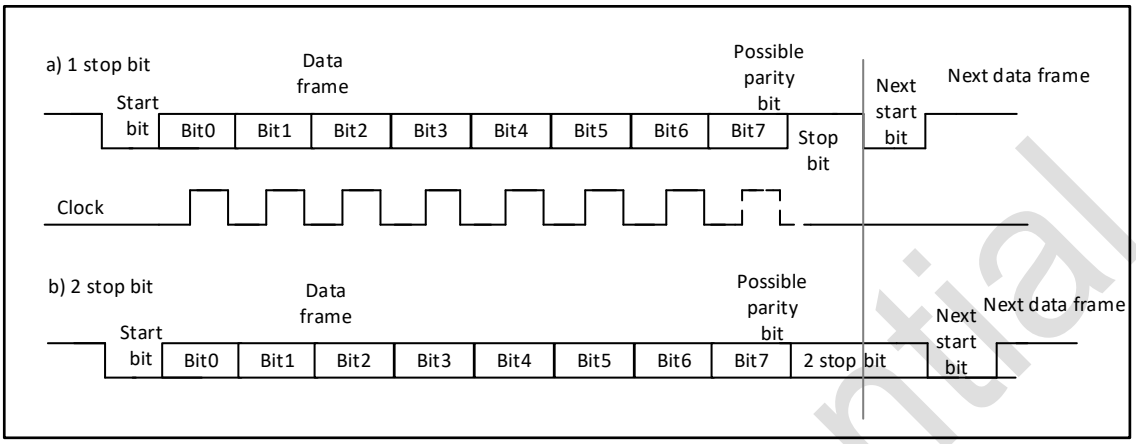


图 31-3 可配置的停止位

配置步骤：

1) 通过在 USART_CR1 寄存器上置位 UE 位来激活 USART

2) 编程 USART_CR1 的 M 位来定义字长。

3) 在 USART_CR2 的 STOP 寄存器编程停止位的位数。

4) 如果采用多缓冲器通信，配置 USART_CR3 中的 DMA 使能位(DMAT)。按“多缓冲器通信”中的描述配置 DMA 寄存器。

5) 利用 USART_BRR 寄存器选择期望的波特率。

6) 设置 USART_CR1 中的 TE 位，发送一个空闲帧作为第一次数据发送。

7) 把要发送的数据写进 USART_DR 寄存器(此动作清除 TXE 位)。在只有一个缓冲器的情况下，对每个待发送的数据重复步骤 7。

8) 在 USART_DR 寄存器中写入最后一个数据字后，要等待 TC=1，它表示最后一个数据帧的传输结束。当需要关闭 USART 或需要进入暂停模式（如进入低功耗模式）之前，需要确认传输结束，避免破坏最后一次传输。

31.3.4.3. 单字节通讯

对数据寄存器进行写操作清零 TXE 位。TXE 位由硬件来置位，它表明：

- 数据已经从 TDR 移送到移位寄存器，数据发送已经开始
- TDR 寄存器被清空
- 下一个数据可以被写进 USART_DR 寄存器而不会覆盖先前的数据

如果 TXEIE 位被设置，TXE 标志将产生一个中断。

如果此时 USART 正在发送数据，对 USART_DR 寄存器的写操作会将数据存进 TDR 缓存器，并在当前传输结束时把该数据复制进移位寄存器。

如果此时 USART 没有在发送数据，处于空闲状态，对 USART_DR 寄存器的写操作直接把数据放进移位寄存器，数据传输开始，TXE 位立即被置起。

当一帧发送完成时(停止位发送后)并且设置了 TXE 位，TC 位被置起，如果 USART_CR1 寄存器中的

TCIE 位被置起时，则会产生中断。

在 USART_DR 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式之前，必须先等待 TC=1(详见下图)。

软件使用下列流程清除 TC 位：

1．读一次 USART_SR 寄存器；

2．写一次 USART_DR 寄存器。

注：TC 位也可以通过软件对它写‘0’来清除。此清零方式只推荐在多缓冲器通信模式下使用。

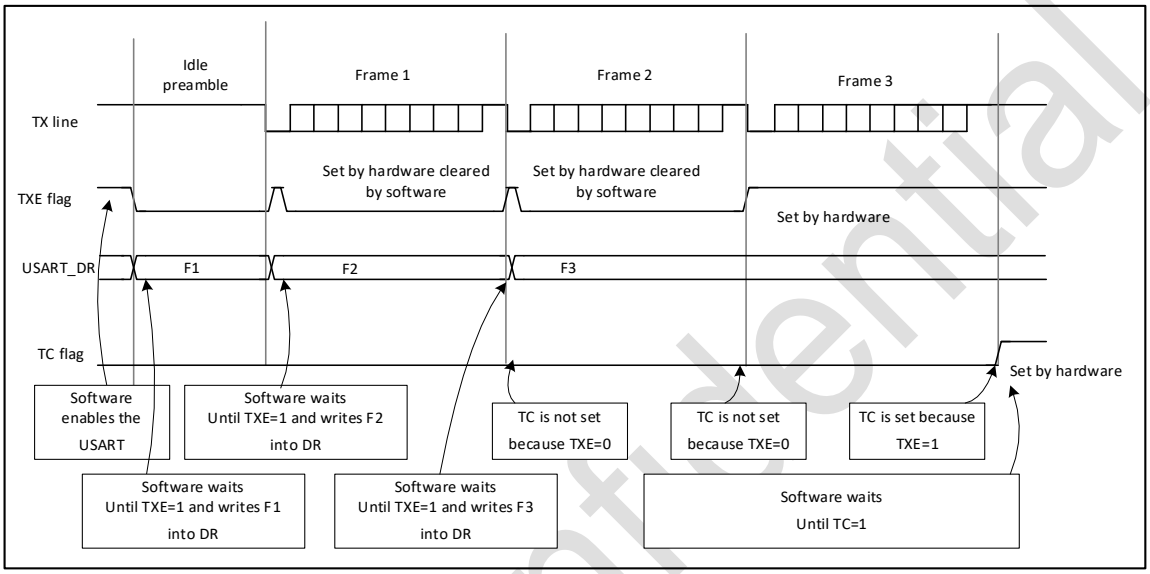


图 31-4 发送时的 TC/TXE 行为

31.3.4.4. 断开字符

设置 SBK 可发送一个断开字符帧。断开帧长度取决 M 位。如果设置 SBK=1，在完成当前数据发送后，将在 TX 线上发送一个断开字符。断开字符发送完成时(在断开符号的停止位时)SBK 被硬件复位。USART 在最后一个断开帧的结束处插入一逻辑‘1’，以保证能识别下一帧的起始位。

注意：如果在开始发送断开帧之前，软件又复位了 SBK 位，断开字符将不被发送。如果要发送两个连续的断开帧，SBK 位应该在前一个断开字符的停止位之后置起。

31.3.4.5. 空闲字符

置位 TE 将使得 USART 在第一个数据帧前发送一个空闲帧。

31.3.5. USART 接收器

USART 可以根据 USART_CR1 的 M 位接收 8 位或 9 位的数据字。

31.3.5.1. 起始位检测

在 USART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。该序列为：1 1 1 0 X 0 X 0 X 0 0 0 0

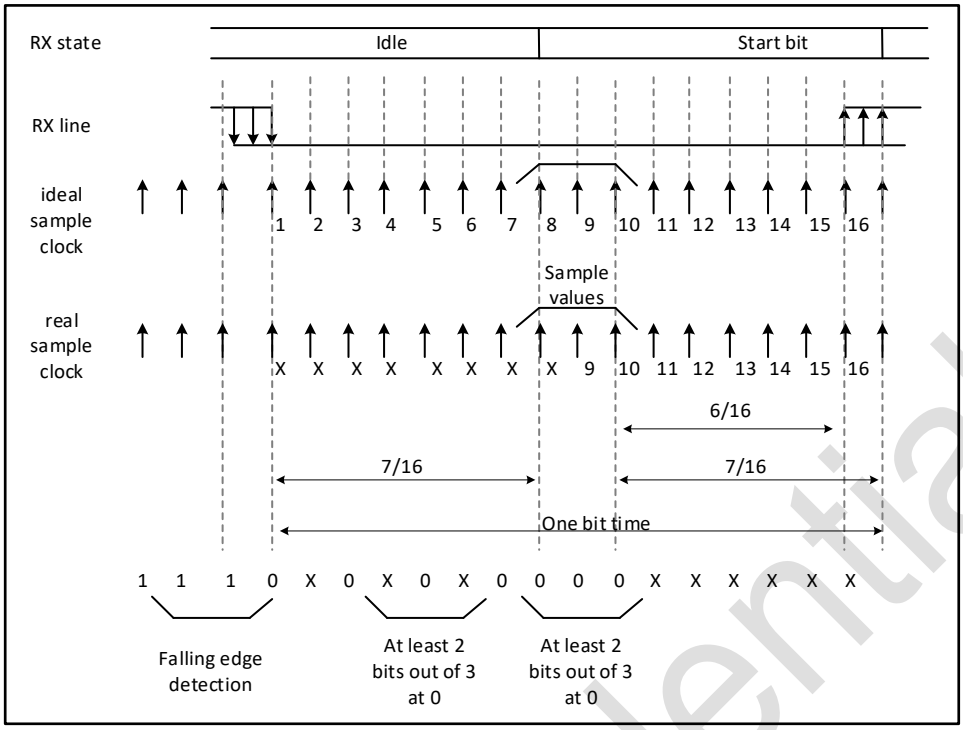


图 31-5 起始位检测

如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态(不设置标志位)等待下降沿。如果 3 个采样点都为‘0’(在第 3、5、7 位的第一次采样，和在第 8、9、10 的第二次采样都为’0’)，则确认收到起始位，这时设置 RXNE 标志位，如果 RXNEIE=1，则产生中断。

如果两次采样，3 个采样点上仅有 2 个是‘0’(第 3、5、7 位的采样点和第 8、9、10 位的采样点)，那么起始位仍然是有效的，但是会设置 NE 噪声标志位。如果不能满足这个条件，则中止起始位的检测过程，接收器会回到空闲状态(不设置标志位)。

如果一次采样，3 个采样点上仅有 2 个是‘0’(第 3、5、7 位的采样点或第 8、9、10 位的采样点)，那么起始位仍然是有效的，但是会设置 NE 噪声标志位。

31.3.5.2. 字符接收

在 USART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里，USART_DR 寄存器包含的缓冲器位于内部总线和接收移位寄存器之间。

配置步骤：

1. 将 USART_CR1 寄存器的 UE 置 1 来激活 USART。
2. 编程 USART_CR1 的 M 位定义字长
3. 在 USART_CR2 的 STOP 寄存器编程停止位的位数
4. 如果需多缓冲器通信，选择 USART_CR3 中的 DMA 使能位(DMAR)。按多缓冲器通信所要求的配置 DMA 寄存器。
5. 利用波特率寄存器 USART_BRR 选择期望的波特率。
6. 设置 USART_CR1 的 RE 位。激活接收器，开始检测起始位。

当接收到一字符时：

- 置位 RXNE 位，表明移位寄存器的内容被传送到 RDR。也就是说，数据已经被接收并且可以被读出(包括与之有关的错误标志)。

- 如果 RXNEIE 位被设置，产生中断。
- 在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起
- 在多缓冲区通信时，RXNE 在每个字节接收后置位。该标志在 DMA 读取接收数据寄存器时被清零。
- 在单缓冲区模式里，软件读 USART_DR 寄存器将 RXNE 位清零。RXNE 标志也可以通过对它写 0 来清零。RXNE 位必须在下一字符接收结束前被清零，以避免溢出错误。

注意：在接收数据时，RE 位不应该被复位。如果 RE 位在接收时被清零，当前字节的接收被丢失。

31.3.5.3. 断开字符

当接收到一个断开字符帧时，USART 将按照帧错误对其进行处理。

31.3.5.4. 空闲字符

当检测到空闲帧时，其处理步骤和接收到普通数据帧一样，但如果 IDLEIE 位被设置将产生一个中断。

31.3.5.5. 上溢错误

如果 RXNE 还没有被复位，又接收到一个字符，则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。RXNE 标志是接收到每个字节后置位的。

当 RXNE 标志置位时，如果在接收到下一个数据或尚未处理上一个 DMA 请求，则产生溢出错误。

当溢出错误产生时：

- ORE 位被置位。
- RDR 内容将不会丢失。读 USART_DR 寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。随后接收到的数据都将丢失。
- 如果 RXNEIE 或 EIE 被设置，则产生中断。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 ORE 位

注意：当 ORE 位置位时，表明至少有 1 个数据已经丢失。有两种可能性：

- 如果 RXNE=1，上一个有效数据还在接收寄存器 RDR 上，可以被读出。
- 如果 RXNE=0，这意味着上一个有效数据已经被读走，RDR 已经没有要读取的数据。当上一个有效数据在 RDR 中被读取的同时又接收到新的(也就是丢失的)数据时，此种情况可能发生。在读序列期间(在 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据，此种情况也可能发生。

31.3.5.6. 噪声错误

使用过采样技术(同步模式除外)，通过区别有效输入数据和噪音来进行数据恢复。

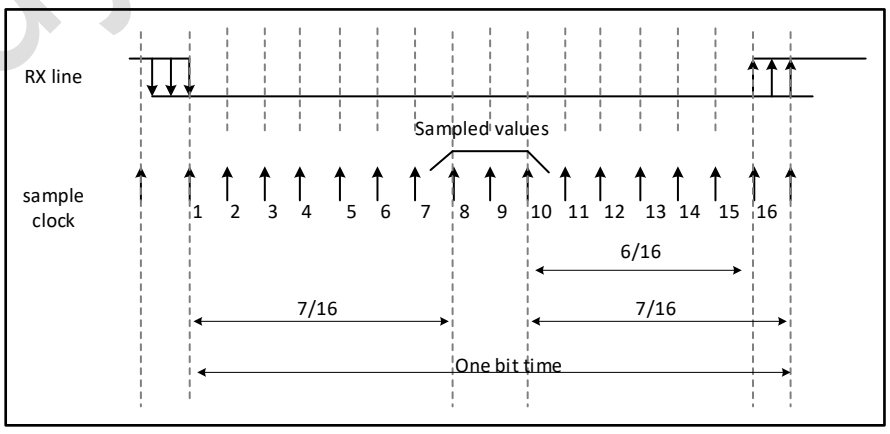


图 31-6 噪声检测时数据采样

表 31-1 检测噪声的数据采样

采样值	NE 状态	接收的位	数据有效性
000	0	0	有效
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效
111	0	1	有效

当在接收帧中检测到噪音时：

- 在 RXNE 位的上升沿设置 NE 标志。
- 无效数据从移位寄存器传送到 USART_DR 寄存器。
- 在单个字节通信情况下，没有中断产生。然而，因为 NE 标志位和 RXNE 标志位是同时被设置，RXNE 将产生中断。在多缓冲区通信情况下，如果已经设置了 USART_CR3 寄存器中 EIE 位，将产生一个中断。
- 先读出 USART_SR，再读出 USART_DR 寄存器，将清除 NE 标志位

31.3.5.7. 帧错误

当以下情况发生时检测到帧错误：

- 如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。
- 当帧错误被检测到时：
- FE 位被硬件置起
- 无效数据从移位寄存器传送到 USART_DR 寄存器。
- 在单字节通信时，没有中断产生。然而，这个位和 RXNE 位同时置起，后者将产生中断。在多缓冲器通信情况下，如果 USART_CR3 寄存器中 EIE 位被置位的话，将产生中断。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 FE 位。

31.3.5.8. 在接收时配置停止位

可通过 USART_CR 的控制位配置要接收的停止位的数量：可以是 1 或 2 个（正常模式下），也可以是 0.5 或 1.5 个（智能卡模式下）。

- 0.5 个停止位（在智能卡模式下接收时）：不会对 0.5 个停止位进行采样。结果，选择 0.5 个停止位时，无法检测到帧错误和断开帧。
- 1 个停止位：对 1 个停止位的采样在第 8，第 9 和第 10 采样点上进行。
- 1.5 个停止位(智能卡模式)：当以智能卡模式发送时，器件必须检查数据是否被正确的发送出去。所以接收器功能块必须被激活(USART_CR1 寄存器中的 RE =1)，并且在停止位的发送期间采样数据线上的信号。如果出现校验错误，智能卡会在发送方采样 NACK 信号时，即总线上停止位对应的时间内时，拉低数据线，以此表示出现了帧错误。FE 在 1.5 个停止位结束时和 RXNE 一起被置位。对 1.5 个停止位的采样是在第 16，第 17 和第 18 采样点进行的。1.5 个的停止位可以被分成 2 部分：一个是 0.5 个时钟周期，期间不做任何事情。随后是 1 个时钟周期的停止位，在这段时间的中点处采样。

- 2 个停止位：对 2 个停止位的采样是在第一停止位的第 8，第 9 和第 10 个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被设置。第二个停止位不再检查帧错误。在第一个停止位结束时 RXNE 标志将被设置。

31.3.6. USART 波特率生成

接收器和发送器（Rx 和 Tx）的波特率均设置为 USART_BRR 寄存器中编程的值。

Tx / Rx 波特率 $= f_{CK} /(16*USARTDIV)$

这里的 fCK 是给外设的时钟，USARTDIV 是一个存放在 USART_BRR 寄存器中的无符号的定点数。

注：在写入 USART_BRR 之后，波特率计数器会被波特率寄存器的新值替换。因此，不要在通信进行中改变波特率寄存器的数值。

如何从 USART_BRR 寄存器值得到 USARTDIV

例 1:

如果 DIV_Mantissa = 27，DIV_Fraction = 12 (USART_BRR=0x1BC),

则:

Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = 12/16 = 0.75

所以 USARTDIV = 27.75

例 2:

要求 USARTDIV = 25.62,

就有:

DIV_Fraction = 16*0.62 = 9.92

最接近的整数是：10 = 0x0A

DIV_Mantissa = mantissa (25.620) = 25 = 0x19

于是，USART_BRR = 0x19A

例 3:

要求 USARTDIV = 50.99

就有:

DIV_Fraction = 16*0.99 = 15.84

最接近的整数是：16 = 0x10 => DIV_frac[3:0]溢出 => 进位必须加到小数部分

DIV_Mantissa = mantissa (50.990 + 进位) = 51 = 0x33

于是：USART_BRR = 0x330，USARTDIV=51

表 31-2 设置波特率时的误差计算

波特率		fpclk = 36MHz			fpclk = 72MHz		
序号	kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.400	937.5	0.00%	2.4	1875	0.00%
2	9.6	9.600	234.375	0.00%	9.6	468.75	0.00%
3	19.2	19.200	117.1875	0.00%	19.2	234.375	0.0%
4	57.6	57.600	39.0625	0.00%	57.6	78.125	0.0%
5	115.2	115.385	19.5	0.15%	115.2	39.0625	0.0%
6	230.4	230.769	9.75	0.16%	230.769	19.5	0.16%
7	460.8	461.538	4.875	0.16%	461.538	9.75	0.16%
8	921.6	923.076	2.4375	0.16%	923.076	4.875	0.16%
9	2250	2250.000	1	0.00%	2250	2	0%
10	4500	不可能	不可能	不可能	4500	1	0%

注：CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

31.3.7. USART 接收容忍度

只有当整体的时钟系统偏差小于 USART 异步接收器能够容忍的范围，USART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的偏差(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足： DTRA + DQUANT + DREC + DTCL < USART 接收器的容忍度。

USART 接收器在下两表中指定的最大容许偏差下可正确接收数据，具体取决于以下设置：

- 由 USART_CR1 寄存器的 M 位定义的 10 或 11 位字符长度
- 由 USART_CR3 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样
- USART_BRR 寄存器的 BRR[3:0]位等于或不等于 0000

表 31-3 DIV_Fraction 为 0 时 USART 接收容忍度

M 位	OVR8=0		OVR8=1	
	认为 NF 是错误	不认为 NF 是错误	认为 NF 是错误	不认为 NF 是错误
0	1.2%	5.05%	2.43%	2.52%
1	1.2%	4.98%	2.43%	2.45%

表 31-4 DIV_Fraction 不为 0 时 USART 接收容忍度

M 位	OVR8=0		OVR8=1	
	认为 NF 是错误	不认为 NF 是错误	认为 NF 是错误	不认为 NF 是错误
0	1.2%	5.05%	2.45%	2.5%
1	1.2%	4.92%	2.45%	2.45%

31.3.8. USART 自动波特率检测

USART 能够基于一个字符的接收，检测并自动设定 USART_BRR 寄存器的值。自动波特率检测在以下场景是有用：

- 系统通讯速率提前未确认
- 系统正在使用相对低精度的时钟源，该机制允许在不测量时钟偏差的情况下，获得正确的波特率时钟源频率必须与预期的通讯速率兼容。必须选择 16 倍过采样，并从 fCK/65535 和 fCK/16 之间选择)
- 过采样 16 时，波特率范围为 $f_{CK}/65535$ ~ $f_{CK}/16$。
- 过采样为 8 时，波特率范围为 $f_{CK}/65535$ 和 $f_{CK}/8$。

在激活自动波特率检测之前，必须通过 USART_CR3 寄存器的 ABRMOD[1:0] 选择自动波特率检测模式。基于不同的字符模式，有各种模式。

在这些自动波特率模式下，波特率在同步数据接收期间会被测量几次，每次测量都会跟之前进行对比。

这些模式是：

- 模式 0: 任何以 1 开始的字符。在这种情况下，USART 测量起始位的宽度（下降沿到上升沿）

- 模式 1：任何以 10xx 位开始的字符。在这种情况下，USART 测量起始位和第一个数据位的宽度。该测量在下降沿到下降沿之间完成，以确保在慢速信号上升情况下确保更好的精度。
- 模式 2：0x7F 字符帧。在此情况下，首先在起始位结束处更新波特率，然后在位 6 结束时更新波特率。
- 模式 3：0x55 字符帧。在此情况下，首先在起始位结束处更新波特率，然后在位 0 的结束时更新波特率，最后在位 6 的结束时更新波特率。同时，对 RX 线路的每个中间转换执行其它检查。

另外，对每个 RX 的传输检查也会并行进行。如果 RX 上的传输没有与接收器充分的同步（接收器基于 bit 0 计算的波特率），则会产生错误。

通过置位 USART_CR3 寄存器的 ABREN 位，可以激活自动波特率检测功能。USART 将等待 RX 上的第一个字符。置位 USART_SR 寄存器的 ABRF 标志，显示了自动波特率检测的完成。如果通讯线上是有噪声的，则自动波特率检测的正确进行不能被保证。在这种情况下，BRR 的值可能被破坏，ABRE 错误标志位将被置位。如果通讯速度与自动波特率检测范围不兼容（16 倍过采样是，位宽不在 16 和 65535 个时钟周期之间），ABRE 错误也会发生。

RXNE 中断表明完成操作。在以后的任何时刻，自动波特率检测可通过复位 ABRF 标志（通过写 0）再次启动。

注意：如果在自动波特期间，清零 UE，BRR 值可能被破坏。

31.3.9. USART 多处理器通信

通过 USART 可以实现多处理器通信(将几个 USART 连在一个网络里)。例如某个 USART 设备可以是主，它的 TX 输出和其他 USART 从机的 RX 输入相连接；USART 从机各自的 TX 输出逻辑地与在一起，并且和主机的 RX 输入相连接。

在多处理器配置中，理想情况下通常只有预期的消息接收方主动接收完整的消息内容，从而减少由所有未被寻址的接收器造成的冗余 USART 服务开销。

可通过静默功能将未被寻址的设备置于静默模式。在静默模式里：

- 不得将接收状态位置 1
- 禁止任何接收中断
- USART_CR1 寄存器中的 RWU 位被置 1。RWU 可以被硬件自动控制或在某个条件下由软件写入

根据 USART_CR1 寄存器中的 WAKE 位状态，USART 可以用两种方法进入或退出静默模式。

- 如果 WAKE 位被复位：进行空闲总线检测。
- 如果 WAKE 位被设置：进行地址标记检测。

31.3.9.1. 空闲帧检测(WAKE=0)

当 RWU 位被写 1 时，USART 进入静默模式。当检测到一空闲帧时，它被唤醒。然后 RWU 被硬件清零，但是 USART_SR 寄存器中的 IDLE 位并不置起。RWU 还可以被软件写 0。下图给出利用空闲总线检测来唤醒和进入静默模式的一个例子。

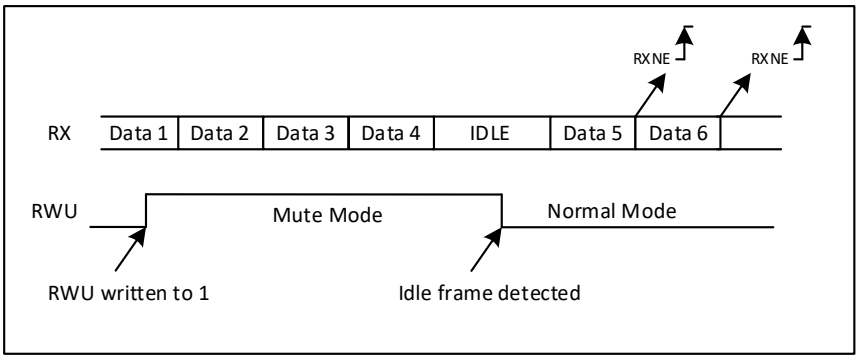


图 31-7 使用空闲帧检测的静默模式

31.3.9.2. 地址标记检测 (WAKE=1)

在这个模式里，如果字节的 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，目标接收器的地址被放在 4 个 LSB 中。这个 4 位地址被接收器同它自己地址做比较，接收器的地址被编程在 USART_CR2 寄存器的 ADD[3:0]。

如果接收到的字节与它的编程地址不匹配时，USART 进入静默模式。此时，硬件设置 RWU 位。

接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 USART 已经在静默模式。

当接收到的字节与接收器配置地址匹配时，USART 退出静默模式。然后清零 RWU 位，正常接收随后的字节。收到这个匹配的地址字节时将设置 RXNE 位，因为 RWU 位已被清零。

当接收缓冲器不包含数据时(USART_SR 的 RXNE=0)，RWU 位可以被写 0 或 1。否则，该次写操作被忽略。下图给出利用地址标志检测来唤醒和进入静默模式的例子。

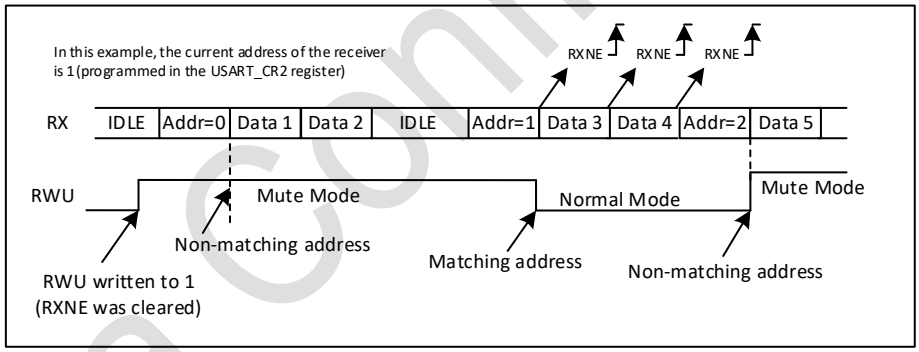


图 31-8 使用地址标记检测的静默模式

31.3.10. USART 奇偶校验控制

设置 USART_CR1 寄存器上的 PCE 位，可以使能奇偶控制(发送时生成一个奇偶位，接收时进行奇偶校验)。根据 M 位定义的帧长度，可能的 USART 帧格式列在下表中。

表 31-5 帧模式

M 位	PCE 位	USART 帧
0	0	起始位—8 位数据—停止位
0	1	起始位—7 位数据—奇偶校验位—停止位
1	0	起始位—9 位数据—停止位
1	1	起始位—8 位数据—奇偶校验位—停止位

在用地址标志唤醒设备时，地址的匹配只考虑到数据的 MSB 位，而不用关心校验位。(MSB 是数据位中最后发出的，后面紧跟校验位或者停止位)

31.3.10.1. 偶校验

校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中'1'的个数为偶数。

例如：数据=00110101，有 4 个’1’，如果选择偶校验(在 USART_CR1 中的 PS = 0)，校验位将是’0’。

31.3.10.2. 奇校验

此校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中’1’的个数为奇数。

例如：数据=00110101，有 4 个’1’，如果选择奇校验(在 USART_CR1 中的 PS = 1)，校验位将是’1’。

31.3.10.3. 接收时进行奇偶校验检查

如果奇偶校验检查失败，则 USART_SR 寄存器中的 PE 标志置 1；如果 USART_CR1 寄存器中 PEIE 位置 1，则会生成中断。

31.3.10.4. 发送时的奇偶校验生成

如果 USART_CR1 寄存器中的 PCE 位置 1，则在数据寄存器中所写入数据的 MSB 位会进行传送，但是会由奇偶校验位进行更改（如果选择偶校验 (PS=0)，则“1”的数量为偶数；如果选择奇校验 (PS=1)，则“1”的数量为奇数）。

31.3.11. USART 同步模式

通过写 USART_CR2 寄存器的 CLKEN 位为 1，选择同步模式。在同步模式里，下列位必须保持清零状态：

- USART_CR2 寄存器中的 LINEN 位
- USART_CR3 寄存器中的 SCEN,HDSEL 和 IREN 位

USART 允许用户以主机模式控制双向同步串行通信。CK 脚是 USART 发送器时钟的输出。在起始位和停止位期间，CK 脚上没有时钟脉冲。根据 USART_CR2 寄存器中 LBCL 位的状态，决定在最后一个有效数据位期间产生或不产生时钟脉冲。USART_CR2 寄存器的 CPOL 位允许用户选择时钟极性，USART_CR2 寄存器上的 CPHA 位允许用户选择外部时钟的相位。

在总线空闲期间，实际数据到来之前以及发送断开符号的时候，外部 CK 时钟不被激活。

同步模式时，USART 发送器和异步模式里工作一模一样。但是因为 CK 是与 TX 同步的(根据 CPOL 和 CPHA)，所以 TX 上的数据是随 CK 同步发出的。

同步模式的 USART 接收器工作方式与异步模式不同。如果 RE=1，数据在 CK 上采样(根据 CPOL 和 CPHA 决定在上升沿还是下降沿)，不需要任何过采样。但必须考虑建立时间和保持时间(取决于波特率：1/16 位时间)。

注意：

- CK 引脚可以与 TX 引脚一起联合工作。因而，只有在使能了发送器(TE = 1)，并且发送数据时(写入数据至 USART_DR 寄存器)才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。
- 应该在发送器和接收器都禁止时，正确配置 LBCL,CPOL 和 CPHA 位；当发送器或接收器使能后，这些位不能被改变。
- 建议在同一条指令中设置 TE 和 RE，以减少接收器的建立时间和保持时间。
- USART **只支持主同步模式**：它不能用来自其他设备的输入时钟接收或发送数据(CK 永远是输出)。

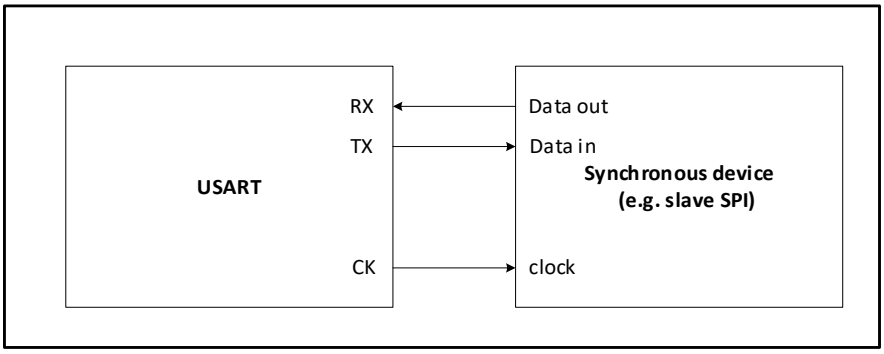


图 31-9 USART 同步传输例子

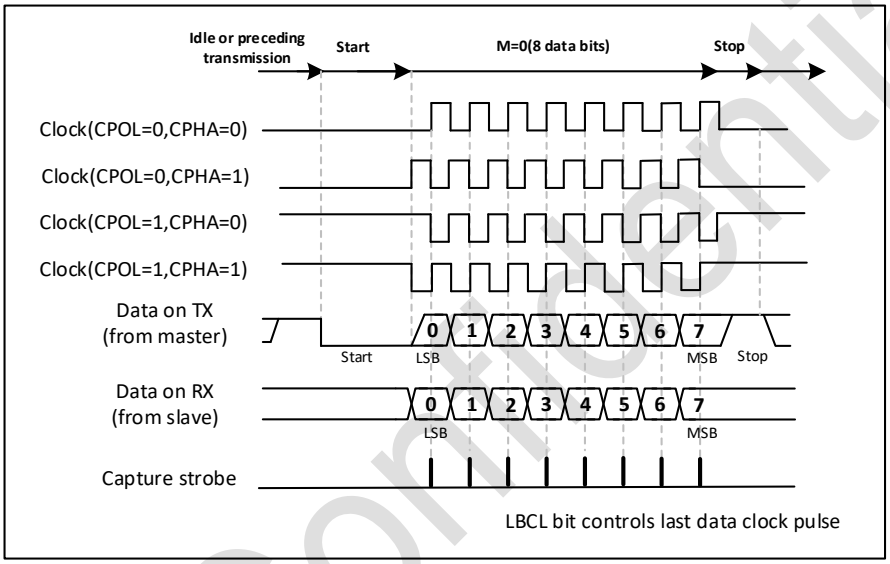


图 31-10 USART 数据时钟序时序图(M=0)

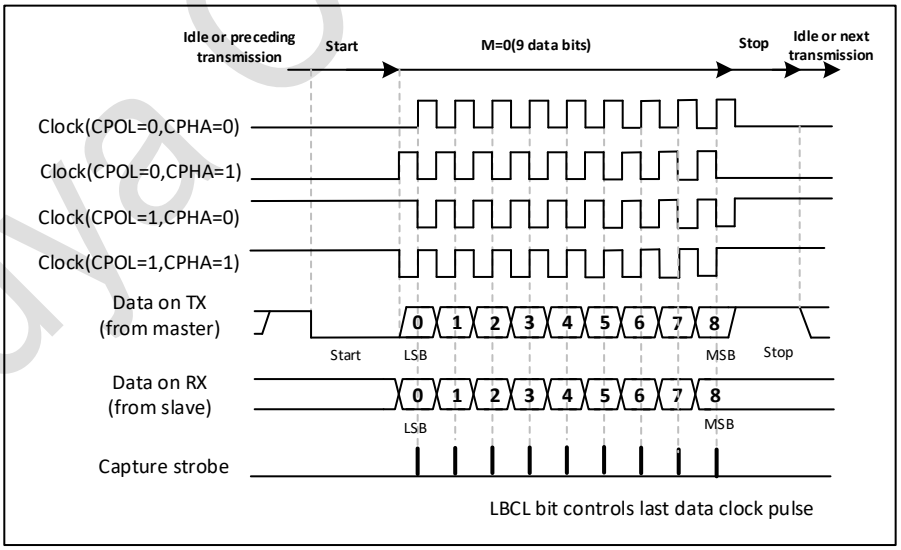


图 31-11 USART 数据时钟序时序图(M=1)

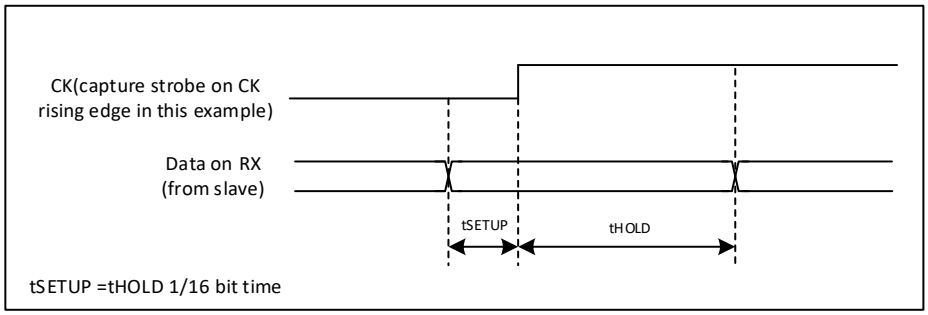


图 31-12RX 数据采样/保持时间

注：在智能卡模式下 CK 的功能不同，有关细节请参考智能卡模式部分。

31.3.12. USART 单线半双工通信

单线半双工模式通过设置 USART_CR3 寄存器的 HDSEL 位选择。在这个模式里，下面的位必须保持清零状态：

- USART_CR2 寄存器的 LINEN 和 CLKEN 位
- USART_CR3 寄存器中的 SCEN 和 IREN 位

USART 可以配置成遵循单线半双工协议。在单线半双工模式下，TX 和 RX 引脚在芯片内部互连。使用“半双工选择”(USART_CR3 中的 HDSEL 位)选择半双工和全双工通信。

当 HDSEL 为 ‘1’ 时

- 不再使用 RX 引脚
- 无数据传输时， TX 引脚始终处于释放状态。因此，它在空闲状态或接收状态时表现为一个标准 I/O 口。这就意味该 I/O 在不被 USART 驱动时，将 TX 配置为复用功能开漏并外接上拉电阻。

除此以外，通信与正常 USART 模式类似。由软件来管理线上的冲突(例如通过使用一个中央仲裁器)。特别的是，发送从不会被硬件所阻碍。当 TE 位被设置时，只要数据一写到数据寄存器，发送就会持续进行。

31.3.13. USART DMA 连续通讯

USART 可以利用 DMA 进行连续通信。接收缓冲区和发送缓冲区的 DMA 请求是独立产生的。

31.3.13.1. DMA 发送

使用 DMA 进行发送，可以通过设置 USART_CR3 寄存器上的 DMAT 位激活。当 TXE 位被置为 ‘1’ 时，DMA 就从指定的 SRAM 区传送数据到 USART_DR 寄存器。为 USART 的发送分配一个 DMA 通道的步骤如下(x 表示通道号)：

1) 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 USART 使用的 DMA 通道
2) 在 DMA 控制寄存器中写入 USART_DR 寄存器地址，将其配置为传输的目标地址。每次发生 TXE 事件后，数据都从存储器移动到此地址。
3) 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的源地址。每次发生 TXE 事件后，数据都从该存储区域加载到 USART_DR 寄存器中。
4) 在 DMA 控制寄存器中配置要传输的总的字节数。
5) 在 DMA 寄存器上配置通道优先级。
6) 根据应用程序的要求，配置在传输完成一半还是全部完成时产生 DMA 中断。
7) 通过写 0，清零 USART_SR 寄存器的 TC 位。
8) 在 DMA 寄存器上激活该通道。

当传输完成 DMA 控制器指定的数据量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

在发送模式下，当 DMA 传输完所有要发送的数据时，DMA 控制器设置 DMA_ISR 寄存器的 TCIF 标志；检测 USART_SR 寄存器的 TC 标志可以确认 USART 通信是否结束，这样可以在关闭 USART 或系统进入低功耗模式之前避免破坏最后一次传输的数据；软件需要先等待 TXE=1，再等待 TC=1。

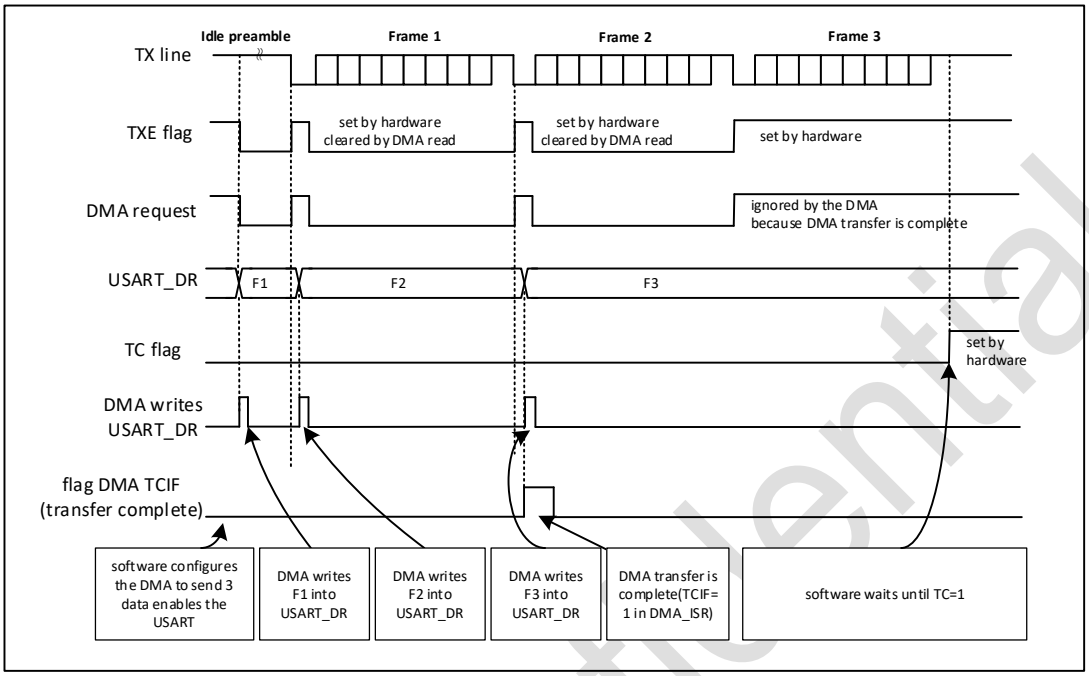


图 31-13 使用 DMA 发送

31.3.13.2. DMA 接收

可以通过设置 USART_CR3 寄存器的 DMAR 位激活使用 DMA 进行接收，每次接收到一个字节，DMA 控制器就就把数据从 USART_DR 寄存器传送到指定的 SRAM 区(参考 DMA 相关说明)。为 USART 的接收分配一个 DMA 通道的步骤如下(x 表示通道号)：

1) 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 USART 使用的 DMA 通道
2) 通过 DMA 控制寄存器把 USART_DR 寄存器地址配置成传输的源地址。在每个 RXNE 事件后，将从此地址读出数据并传输到存储器。
3) 通过 DMA 控制寄存器把存储器地址配置成传输的目的地址。在每个 RXNE 事件后，数据将从 USART_DR 传输到此存储器区。
4) 在 DMA 控制寄存器中配置要传输的总的字节数。
5) 在 DMA 寄存器上配置通道优先级。
6) 根据应用程序的要求配置在传输完成一半还是全部完成时产生 DMA 中断。
7) 在 DMA 控制寄存器上激活该通道。

当接收完成 DMA 控制器指定的传输量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

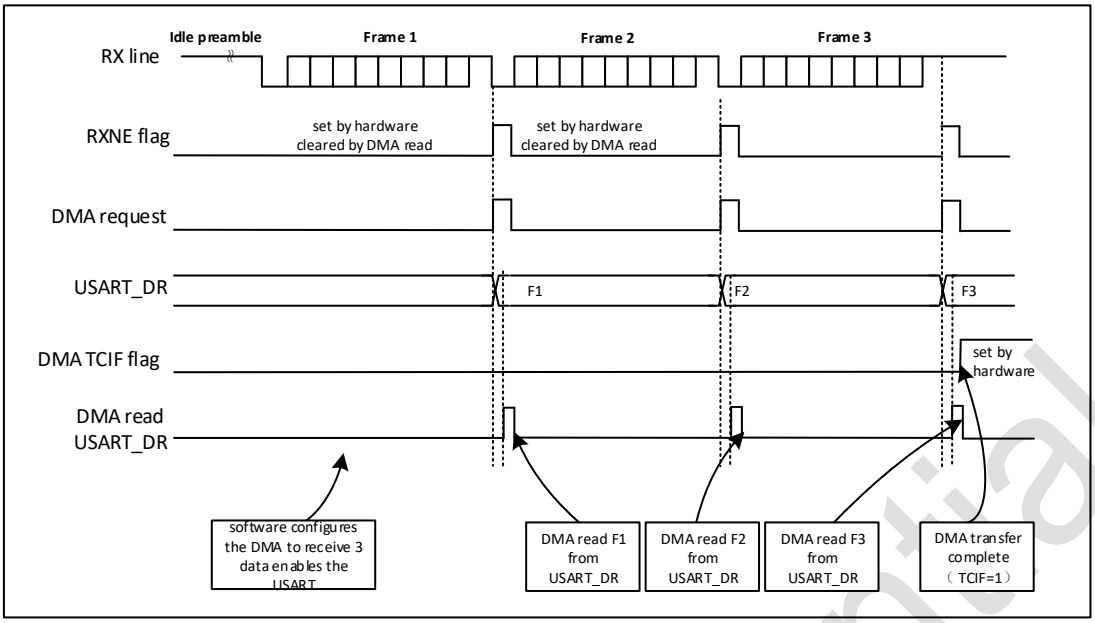


图 31-14 使用 DMA 接收

31.3.13.3. 多处理器通讯错误标志和中断产生

在多缓冲器通信的情况下，通信期间如果发生任何错误，在当前字节传输后将置起错误标志。如果中断使能位被设置，将产生中断。在单个字节接收的情况下，和 RXNE 一起被置起的帧错误、溢出错误和噪音标志，有单独的错误标志中断使能位；如果设置了，会在当前字节传输结束后，产生中断。

31.3.14. RS232 硬件流控制

利用 nCTS 输入和 nRTS 输出可以控制 2 个设备间的串行数据流。下图表明在这个模式里如何连接 2 个设备。

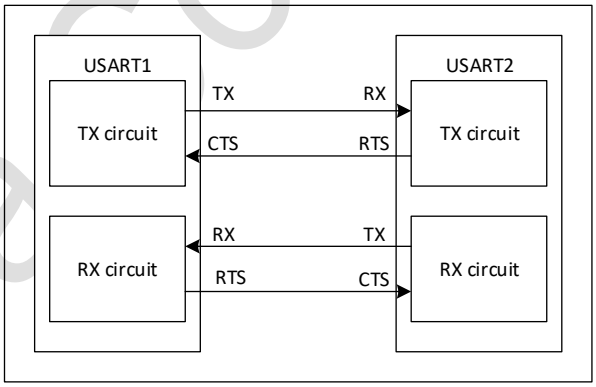


图 31-15 USART 间的硬件流控制

31.3.14.1. RS232 RTS 流控制

如果使能 RTS 流控制 (RTSE=1)，只要 USART 接收器准备好接收新的数据，nRTS 就变成有效(接低电平)。当接收寄存器内有数据到达时，nRTS 被释放，表明发送过程会在当前帧结束后停止。

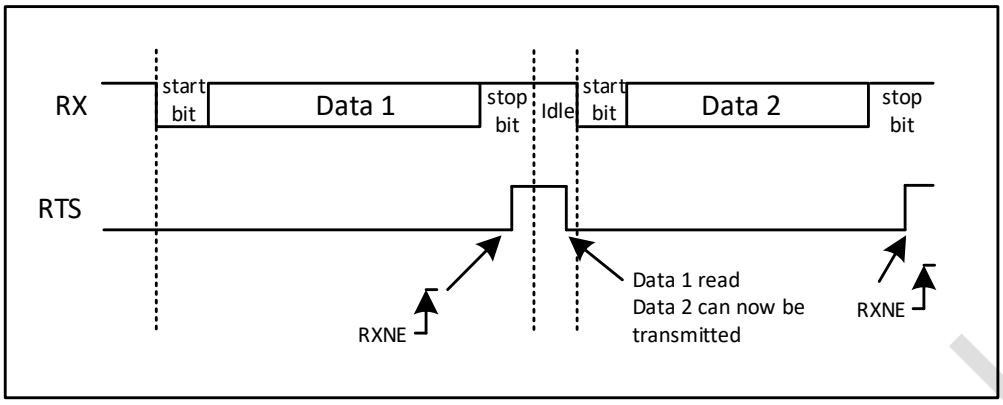


图 31-16 RTS 流控制

31.3.14.2. RS232 CTS 流控制

如果使能 CTS 流控制（CTSE=1），发送器在发送下一帧前检查 nCTS 输入。如果 nCTS 有效(被拉成低电平)，则会发送下一个数据（假设数据已准备好发送，也就是 TXE=0），否则不会发送下一帧数据。若 nCTS 在传输期间被变成无效，则当前的传输完成后，发送器停止。

当 CTSE=1 时，只要 nCTS 输入一变换状态，硬件就自动设置 CTS 状态位。它表明接收器是否准备好进行通信。如果设置了 USART_CT3 寄存器的 CTSIE 位，则产生中断。

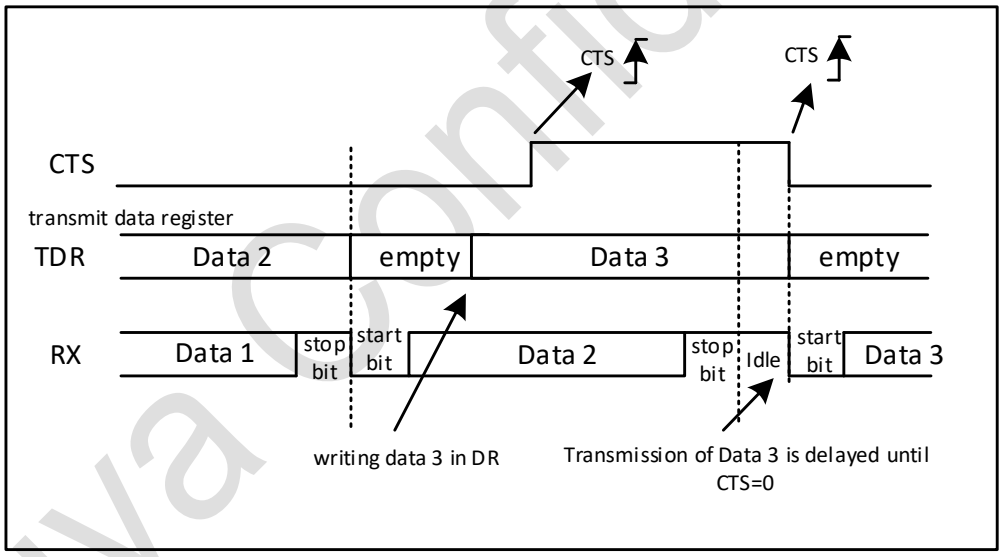


图 31-17 CTS 流控制

31.3.15. USART LIN(局域互联网)模式

配置 USART_CR2.LINEN=1 选择 LIN 模式。在 LIN 模式下，下列位必须保持为 0：

- USART_CR2 寄存器的 STOP/CLKEN;
- USART_CR3 寄存器的 SCEN/HDSEL/IREN.

31.3.15.1. LIN 发送

与一般 USART 发送相比，LIN 发送有如下区别：

- M=0，数据长度为 8bit；
- 需要配置 USART_CR2.LINEN=1。置位 SBK 后将发送 13bit 0 作为断开帧。然后发送一位“1”，以允许对下一个起始位检测。

31.3.15.2. LIN 接收

当 LIN 模式使能后，将激活断开符号检测电路。该检测完全独立于 USART 接收器。在空闲状态或某个帧期间，只要发生中断即可检测出来。

接收器使能后(USART_CR1 的 RE=1)，电路便开始监测 RX 上的起始信号。监测起始位的方法同检测断开符号或数据是一样的。当检测到起始位后，电路对每个接下来的位进行采样，方法与数据采样相同（第 8、第 9 和第 10 次采样）。如果 10 个(当 USART_CR2 的 LBDL= 0) 或 11 个(当 USART_CR2 的 LBDL= 1)连续位都是 0，且其后跟随分隔符，则 USART_SR 的 LBD 标志被设置。如果 LBDIE 位=1，则产生中断。在确认断开符号前，要检查分隔符，因为它意味 RX 线已经恢复到高电平。

如果在第 10 或 11 个采样点之前采样到了 1，检测电路取消当前检测并重新寻找起始位。

如果 LIN 模式被禁止，接收器会作为正常的 USART 继续工作，不再进行断开符的检测。

如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误(也就是停止位检测到‘0’，这种情况出现在断开帧)，接收器就停止，直到断开符号检测电路接收到一个‘1’(这种情况发生在断开符号不完整时)，或接收到分隔符(这种情况发生在已经检测到一个完整的断开符号)为止。

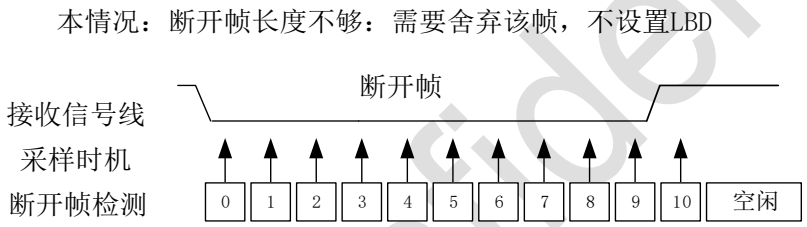


图 31-18 LIN 模式下的断开帧长度不够的情况

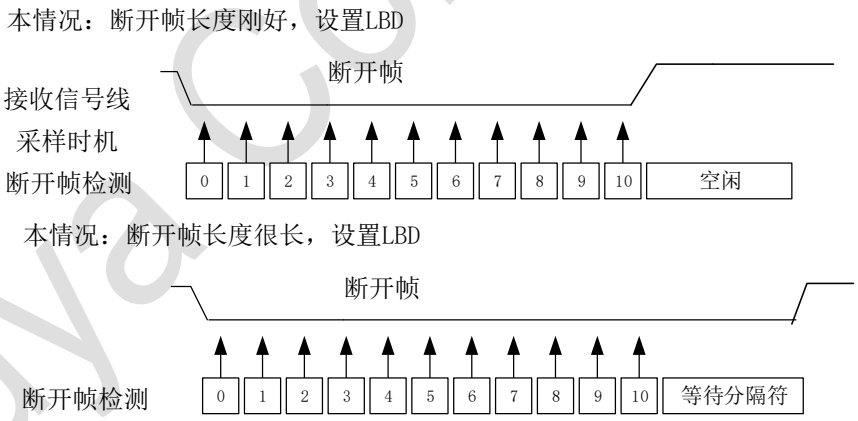
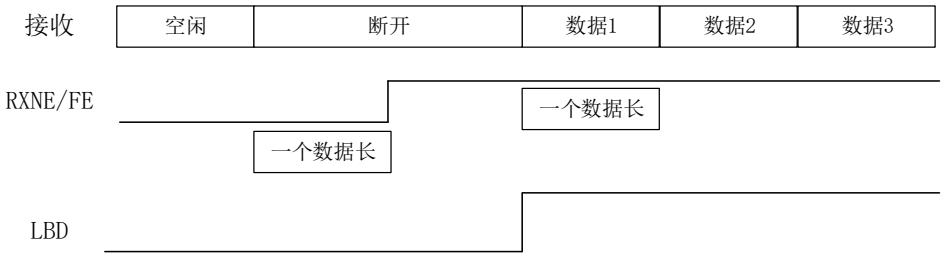


图 31-19 LIN模式下的断开帧长度够的情况

断开发生在空闲后



断开发生在正在接收数据时

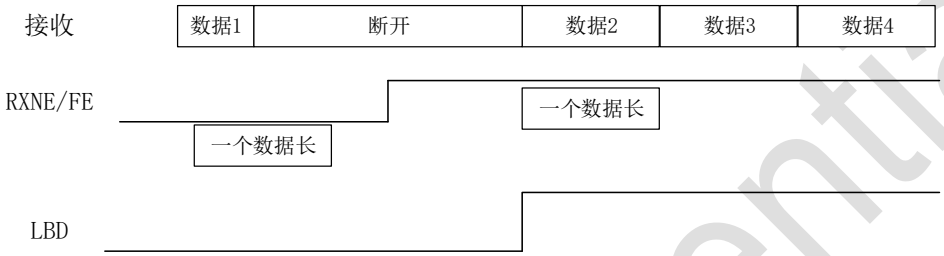


图 31-20 LIN 模式下的断开检测与帧错误的检测

31.3.16. USART 智能卡模式

设置 USART_CR3 寄存器的 SCEN 位选择智能卡模式。在智能卡模式下，下列位必须保持清零：

- USART_CR2 寄存器的 LINEN 位
- USART_CR3 寄存器的 HDSEL 位和 IREN 位

此外，CLKEN 位可以被设置，以提供时钟给智能卡。

注意：支持智能卡模式时，必须在使能 SCEN 后才能配置 TE 和 RE 寄存器位。

该接口符合 ISO7816-3 标准的异步智能卡协议。同时支持 T=0（字符模式）和 T=1（块模式）。

USART 应该被设置为：

- 8 位数据位加奇偶校验位：此时 USART_CR1 寄存器中 M=1、PCE=1
- 发送和接收时为 1.5 个停止位：即 USART_CR2 寄存器的 STOP=11。也可以在接收时选择 0.5 个停止位，但为了避免在 2 种配置间转换，建议在发送和接收时使用 1.5 个停止位。

下图给出的例子说明了数据线上，在有校验错误和没校验错误两种情况下的信号。

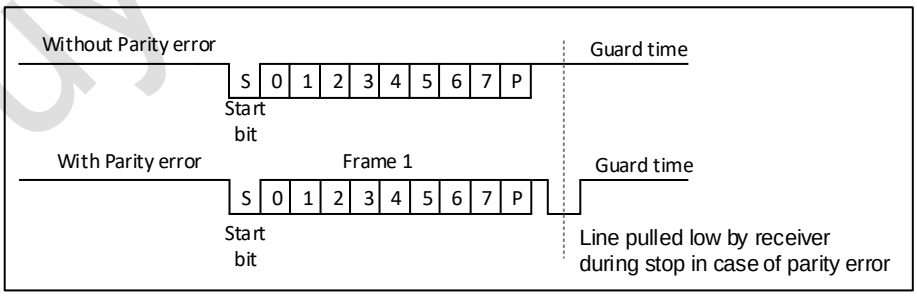


图 31-21 ISO7816-3 异步协议

当与智能卡相连接时，USART 的 TX 驱动一条双向线（智能卡也驱动）。为了做到这点，RX 必须和 TX 连接到相同的 I/O 口。在发送开始位和数据字节期间，发送器的输出使能位 TX_EN 被置起，在发送停止位期间被释放(弱上拉)，因此在发现校验错误的情况下接收器可以将数据线拉低。如果 TX_EN 不被使用，在停止位期间 TX 被拉到高电平：这样的话，只要 TX 配置成开漏，接收器也可以驱动这根线。

智能卡是一个单线半双工通信协议

- 从发送移位寄存器发送数据会经过至少 1/2 波特时钟。在正常操作时，一个满的发送移位寄存器将在下一个波特时钟沿开始移位。在智能卡模式里，此发送还会进一步经过 1/2 波特周期的延迟。
- 如果在接收一个设置为 0.5 或 1.5 个停止位的数据帧期间，检测到一奇偶校验错误，在完成接收该帧后(即停止位结束时)，发送线被拉低一个波特时钟周期。这是告诉智能卡发送到 USART 的数据没有被正确地接收到。此 NACK 信号(拉低发送线一个波特时钟周期)在发送端将产生一个帧错误(发送端被配置成 1.5 个停止位)。应用程序可以根据协议处理重新发送数据。如果设置了 NACK 控制位，发生校验错误时接收器会给出一个 NACK 信号；否则就不会发送 NACK。
- 发送时，USART 会在两个连续字符之间插入保护时间（按照保护时间寄存器中编程的值）。由于保护时间在前一个字符的停止位后测量，因此必须将 GT[7:0]寄存器编程为所需 CGT（字符保护时间，如 ISO7816-3 规范所定义）减去 12（一个字符的持续时间）。
- TC 标志的置起可以通过编程保护时间寄存器得以延时。在正常操作时，当发送移位寄存器变空并且没有新的发送请求出现时，TC 被置起。在智能卡模式里，空的发送移位寄存器将触发保护时间计数器开始递增计数，直到计数达到保护时间寄存器中的值。TC 在这段时间被强制拉低。当保护时间计数器达到保护时间寄存器中的值时，TC 被置高。
- TC 标志的撤销不受智能卡模式的影响。
- 如果发送器检测到一个帧错误(收到接收器的 NACK 信号)，发送器的接收功能模块不会把 NACK 当作起始位检测。根据 ISO7816-3 协议，接收到的 NACK 的持续时间可以是 1 或 2 波特时钟周期。
- 在接收器这边，如果一个校验错误被检测到，并且发送了 NACK，接收器不会把 NACK 检测成起始位。

注意：

1. 断开符号在智能卡模式里没有意义。一个带帧错误的 00h 数据将被当成数据而不是断开符号。
2. 当翻转 TE 位时，不会发送空闲帧。ISO 协议没有定义空闲帧。

下图详述了 USART 是如何采样 NACK 信号的。在这个例子里，USART 正在发送数据，并且被配置成 1.5 个停止位。为了检查数据的完整性和 NACK 信号，USART 的接收功能块被激活。

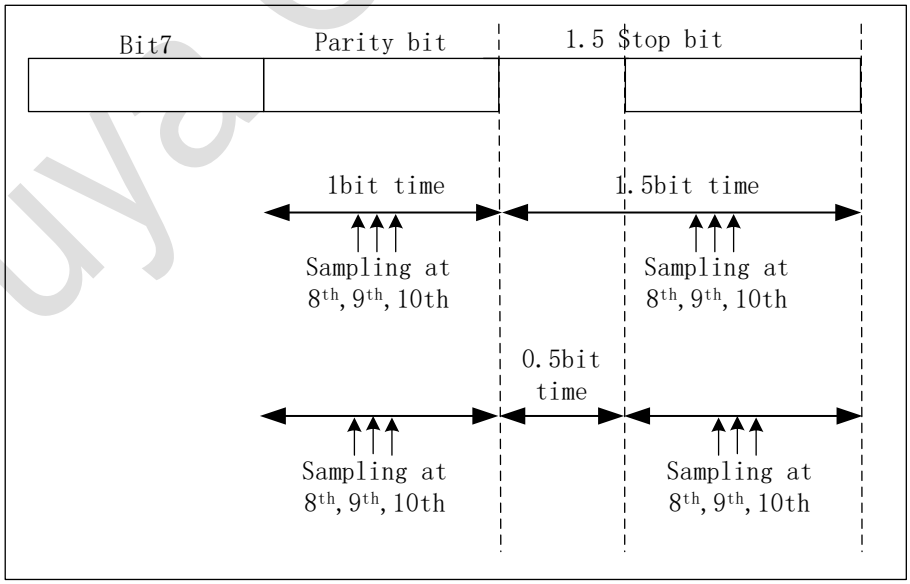


图 31-22 使用 1.5 停止位检测奇偶检验错

USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式里，CK 不和通信直接关联，而是先通过一个 5 位预分频器简单地用内部的外设输入时钟来驱动智能卡的时钟。分频率在预分频寄存器 USART_GTPR 中配置。CK 频率可以在 fCK/2 到 fCK/62 之间进行编程，这里的 fCK 是外设输入时钟。

31.3.16.1. 块模式（T=1）

在 T=1（块）模式下，通过将 USART_CR3 寄存器中的 NACK 位清零可停用奇偶校验错误发送。在块模式下请求从智能卡执行读操作时，软件必须将 RTOR 寄存器编程为 BWT（块等待时间- 值 11）。如果在经过此时间段后未从智能卡接收到应答，将生成超时中断。如果该时间段后接收到第一个字符，则通过 RXNE 中断发出信号指示。

注意：即使在使用 DMA 模式下的 USART 从块模式下的智能卡读取时，也必须使能 RXNE 中断。同时，只有在接收到第一个字节后才可使能 DMA。

接收到第一个字符（RXNE 中断）后，为允许自动检查两个连续字符间的最长等待时间，必须将 USART_RTOR.RTO 寄存器编程为 CWT（字符等待时间-值 11）。此时间以波特率时间单位表示。前一个字符结束后，如果智能卡未在小于 CWT 的时间段内发送新字符，USART 将通过 RTOF 标志和中断（当 RTOIE 位置 1 时）向软件指示此情况。

注意：按照智能卡协议定义，BWT/CWT 的值应从最后一个字符开始（起始位）时定义。必须将 RTO 寄存器分别编程为 BWT-11 或 CWT-11，并考虑最后一个字符本身的长度。

块长度计数器用于对 USART 接收到的所有字符进行计数。当 USART 进行发送时，此计数器复位。块 长度由智能卡在块的第三个字节（起始字段）中传达。必须将此值编程到 USART_RTOR 寄存器中的 BLEN 字段。使用 DMA 模式时，在块开始之前，必须将此寄存器字段编程为最小值(0x0)。对于该值，在接收到第四个字符后生成中断。软件必须读取 LEN 字段（第三个字节），其值必须从接收缓冲区中读取。

在中断驱动接收模式下，块长度可以由软件或者通过编程 BLEN 值来检查。但是在块开始前，可以编程 BLEN 的最大值（0xFF)。接收到第三个字符后，将编程实际值。

如果块使用 LRC（纵向冗余校验，1 个结尾字节），则 BLEN=LEN。如果块使用 CRC 机制（2 个结尾字节），则必须编程 BLEN=LEN+1。块总长度（包括起始字段、结尾字段和信息字段）等于 BLEN+4。块结束的信号通过 EOBF 标志和中断（EOBIE 位置 1 时）发送给软件。

如果块长度出现错误，则通过 RTO 中断（字符等待时间上溢）发送块结束信号。

注意：错误检查代码（LRC/CRC）必须通过软件计算/验证。

31.3.16.2. 正向约定和反向约定

智能卡协议定义了两种约定：正向约定和反向约定。

正向约定定义为：LSB 在前，逻辑位值 1 对应于线路的 H 状态，奇偶校验为偶校验。要使用此约定，必须编程以下控制位：MSBFIRST=0，DATAINV=0（默认值）。

反向约定定义为：MSB 在前，逻辑位值 1 对应于信号线路的 L 状态，奇偶校验为偶校验。要使用此约定，必须编程以下控制位：MSBFIRST=1，DATAINV=1。

注意：将逻辑数据值取反（0=H，1=L）时，奇偶校验位将同样取反。

为识别智能卡约定，智能卡会将初始字符 TS 作为 ATR（复位应答）的第一个字符发送。TS 支持两种格式：LHHL LLL LLH 和 LHHL HHH LLH。

- (H) LHHL LLL LLH 建立反向约定：状态 L 编码为值 1，时间分量 2 传送最高有效位（MSB 在前）。按反向约定解码时，传送的字节等于“3F”。
- (H) LHHL HHH LLH 建立正向约定：状态 H 编码为值 1，时间分量 2 传送最低有效位（LSB 在前）。按正向约定解码时，传送的字节等于“3B”。

在 2 到 10 的九个时间分量中，如果有偶数个位设置为 1，则字符的奇偶校验正确。

由于 USART 不了解智能卡使用哪种约定，因此 USART 需要能够识别任意一种模式并相应操作。模式

识别不在硬件中完成，而是通过软件序列完成。此外，假设以正向约定配置 USART（默认）而智能卡以反向约定 (TS = LHHL LLL LLH) 应答，则 USART 接收到的字节将为“03”，奇偶校验将为奇校验。

因此，有两种方法可用于识别 TS 模式：

方法 1

将 USART 编程为标准智能卡模式/正向约定。这种情况下，TS 模式接收会向智能卡生成奇偶校验错误中断和错误信号。

- 奇偶校验错误中断通知软件智能卡未以正向约定正确应答。之后，软件重新将 USART 编程为反向约定
- 为响应错误信号，智能卡会重试同一 TS 字符，此时重新编程后的 USART 将正确接收该字符或者，为应答奇偶校验错误中断，软件可决定重新编程 USART，同时向智能卡生成新的复位命令，然后再次等待 TS。

方法 2

将 USART 编程为 9 位/无奇偶校验模式，无位反向。在此模式下，按如下方式接收两种 TS 模式中的任一种：

(H) LHHL LLL LLH = 0x103 -> 选择反向约定

(H) LHHL HHH LLH = 0x13B -> 选择正向约定

软件根据这两种模式检查接收到的字符，如果其中任意一种匹配，则相应编程 USART 以接收下一个字符。

如果两种都未被识别，则可能复位智能卡以重新开始协商。

31.3.17. USART IrDA SIR ENDEC 功能模块

通过设置 USART_CR3 寄存器的 IREN 位选择 IrDA 模式。在 IrDA 模式里，下列位必须保持清零：

- USART_CR2 寄存器的 LINEN,STOP 和 CLKEN 位
- USART_CR3 寄存器的 SCEN 和 HDSEL 位。

31.3.17.1. IrDA 正常模式

IrDA SIR 物理层规定使用反相归零 (RZI) 调制方案，该方案用一个红外光脉冲代表逻辑 ‘0’。SIR 发送编码器对从 USART 输出的 NRZ(非归零)比特流进行调制。输出脉冲流被传送到一个外部输出驱动器和红外 LED。USART 为 SIR ENDEC 最高只支持到 115.2Kbps 速率。在正常模式里，脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 USART。在空闲状态里，解码器输入通常是高(标志状态)。发送编码器输出的极性和解码器的输入相反。当解码器输入为低时，检测到一个起始位。

- IrDA 是一个半双工通信协议。如果发送器忙(也就是 USART 正在送数据给 IrDA 编码器)，IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙(也就是 USART 正在接收从 IrDA 解码器来的解码数据)，从 USART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- SIR 发送逻辑把 ‘0’ 作为高脉冲发送，把 ‘1’ 作为低电平发送。脉冲的宽度规定为正常模式时位周期的 3/16。
- SIR 接收逻辑把高电平状态解释为’1’，把低脉冲解释为 ‘0’ 。
- 发送编码器输出与解码器输入有着相反的极性。当空闲时，SIR 输出处于低状态。
- SIR 解码器把 IrDA 兼容的接收信号转变成给 USART 的比特流。

- IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收器端的尖峰脉冲检测逻辑滤除宽度小于 2 个 PSC 周期的脉冲(PSC 是在 IrDA 低功耗波特率寄存器 USART_GTPR 中编程的预分频值)。宽度小于 1 个 PSC 周期的脉冲一定被滤除掉，但是那些宽度大于 1 个而小于 2 个 PSC 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。当 PSC=0 时，IrDA 编码器/解码器不工作。
- 接收器可以与低功耗发送器通信。
- 在 IrDA 模式里，USART_CR2 寄存器上的 STOP 位必须配置成 1 个停止位。

注意：

1. 宽度小于 2 个大于 1 个 PSC 周期的脉冲可能会也可能不会被滤除。

2. 接收器的建立时间应该由软件管理。IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时(IrDA 是一个半双工协议)。

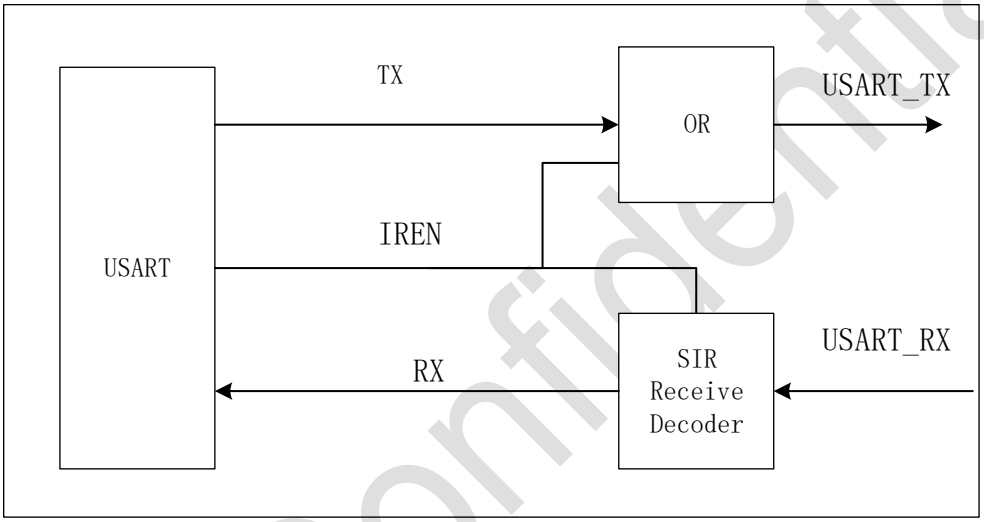


图 31-23 IrDA SIR ENDEC 框图

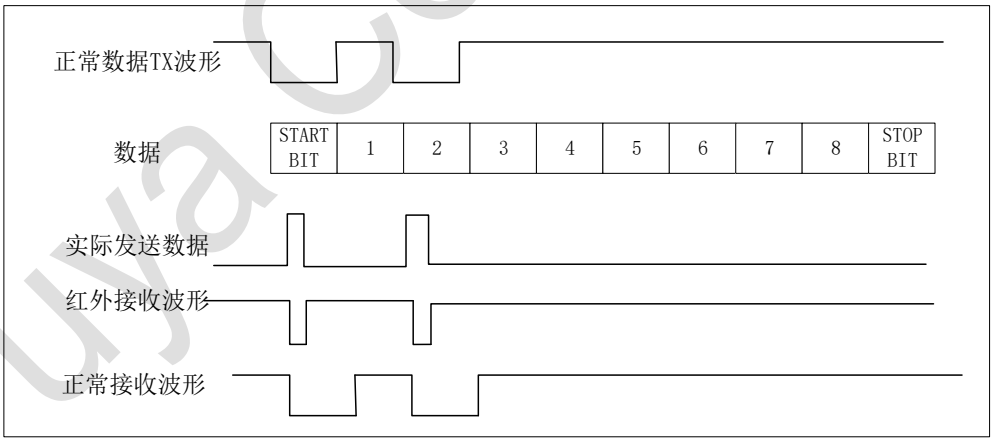


图 31-24 IrDA 数据调制（3/16）—普通模式

31.4. USART 中断

表 31-6 USART 中断

序号	中断事件	事件标志	使能位	中断清除方法
1	发送数据寄存器空	TXE	TXEIE	TXE 在 USART_DR 中被写入数据时清零
2	CTS 中断	CTS	CTSIE	写 USART_SR.CTS 为 0 清零
3	传送完成	TC	TCIE	1. 写 USART_SR.TC 为 0 清零 2. 读 USART_SR 寄存器后写 USART_DR 寄存器清零
4	接收寄存器非空（读数据准备好）	RXNE	RXNEIE	1. 写 USART_SR.RXNE 为 0 清零 2. 读 USART_DR 寄存器清零
5	溢出错误	ORE		读 USART_SR 寄存器后读 USART_DR 寄存器清零
6	空闲帧	IDLE	IDLEIE	读 USART_SR 寄存器后读 USART_DR 寄存器清零
7	奇偶校验错误	PE	PEIE	读 USART_SR 寄存器后读 USART_DR 寄存器清零
8	LIN 断开标志	LBD	LBDIE	写 USART_SR.LBD 为 0 清零
9	多处理器通讯时，噪声、溢出和帧错误	NE/ORE/FE	EIE	读 USART_SR 寄存器后读 USART_DR 寄存器清零
10	发送奇偶校验错误（智能卡）	TXPE	PEIE	写 USART_SR.TXPE 为 0 清零
11	接收超时	RTOF	RTOIE	写 USART_SR.RTOF 为 0 清零
12	块结束	EOBF	EOBIE	写 USART_SR.EOBF 为 0 清零

所有 USART 中断共用同一个中断向量。

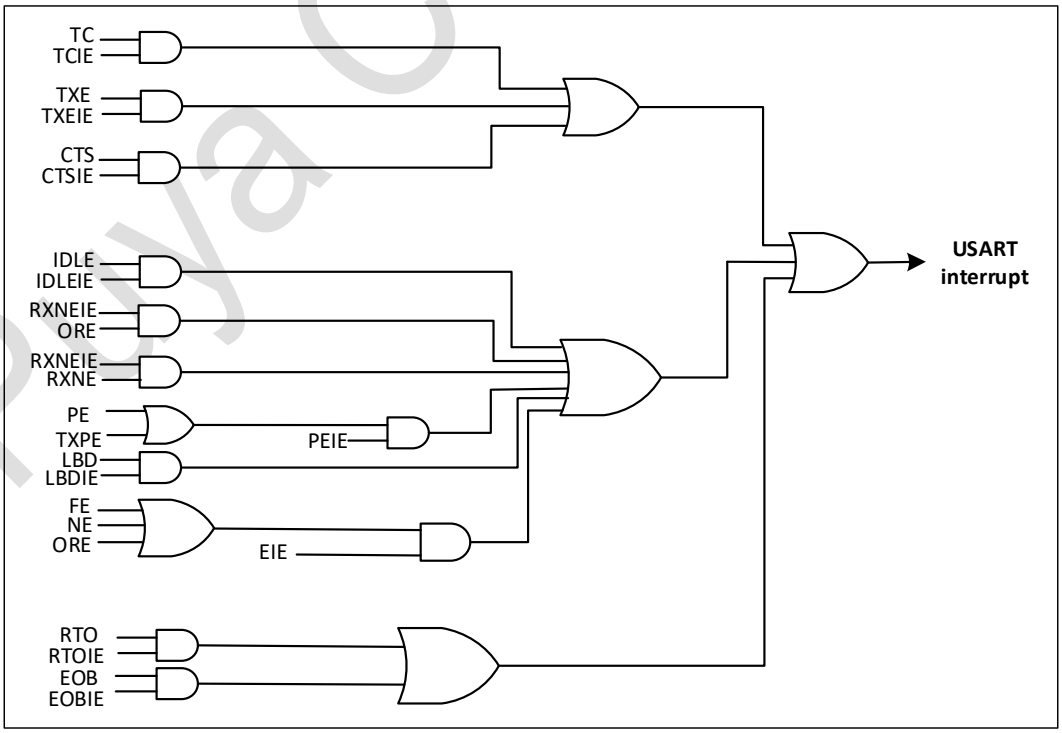


图 31-25 USART 中断映射图

31.5. 软件流程

31.5.1. 智能卡模式使用 BLEN、EOBF 寄存器的流程

智能卡模式块帧包含 3 个字段，分别是序言字段、信息字段和尾声字段，如下表所示：

表 31-7 智能卡模式块帧

序言字段			信息字段	尾声字段
节点地址 NAD	协议控制字节 PCB	长度 LEN	可选 INF	错误检测 LRC 或 CRC EDC
1 字节	1 字节	1 字节	0-254 字节	1/2 字节

31.5.1.1. DMA 模式

DMA 模式，按照如下流程配置：

1.在帧的起始或起始前，配置 BLEN=4，当接收到 INF 的第一字节后，硬件可产生一个 EOBF 中断。

2.软件在 EOBF 中断程序中，解析长度字段（第三个数据），配置 BLEN=LEN。

注：该步骤中 EOBF 清零的操作需要在一包数据之内完成，否则可能导致后续数据清除计数，导致实际的 EOBF 置位偏后。

3.之后硬件正常接收信息字段，会在块结束后（LRC/CRC 之后）重新置位 EOBF。表示块接收结束。

注：我们的 BLEN 配置后，接收到 BLEN 个字节数据即置起 EOBF。也就是说，BLEN 的值包含了信息字段和尾声字段。

31.5.1.2. 中断模式

方法 1：利用 RXNE 状态位

软件每收到 1 个字节数据后利用 RXNE 中断计数，判断块结束。（不使用 EOBF，不使能 EOBIE）。

方法 2：利用 EOBF 状态位

1.在帧的起始或起始前，配置 BLEN=0xff，

2.每接收到一个字节数据，进入一次 RXNE 中断，需要解析数据；

3.接收到第三字节后解析 LEN 字段后，在第四字节接收后将实际的 LEN 配置到 BLEN 寄存器；

4.之后硬件正常接收信息字段，会在块结束后（LRC/CRC 之后）置起 EBOF。表示块接收结束。

注：我们的 BLEN 配置后，接收到 BLEN 个字节数据后即置起 EOBF。也就是说，BLEN 的值包含了信息字段和尾声字段。

31.6. USART 寄存器

31.6.1. USART 状态寄存器 (USART_SR)

偏移地址:0x00

复位值:0x0000_00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WAIT_RPT	TXBUSY	RXBUSY	TXPE	RTOF	EOBF	Res.	Res.
								R	R	R	RC_W0	RC_W0	RC_W0		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	ABRRQ	ABRE	ABRF	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE

			RW	R	R	RC_W0	RC_W0	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	WAIT_RPT	R	0	该寄存器适用于智能卡模式。 硬件在发送了 NACK 信号，正在等待对方重发数据时置位；在收到数据起始位或者进入发送状态时硬件清零; 软件可以查询此位，但是不能改写。
22	TXBUSY	R	0	发送数据忙标志。（发送完成后自动清零） 0：数据发送空闲 1：处于数据发送状态，发送移位寄存器正在发送数据。
21	RXBUSY	R	0	接收数据忙标志。（接收完成后自动清零） 0：数据接收空闲； 1：处于数据接收状态，接收移位寄存器正在接收数据；
20	TXPE	RC_W0	0	发送数据奇偶校验错误的标志位。硬件置位，软件写 0 清零。置位条件参见 TREPEN 的描述。 该寄存器适用于智能卡模式。
19	RTOF	RC_W0	0	已经达到在 RTOR 寄存器中编程的超时值后，如果无任何通信，此位由硬件置 1。 如果 USART_CR2 寄存器中 RTOIE=1，则会生成中断。软件写 0 清零。 在智能卡模式下，该超时对应于 CWT 或 BWT 时间。 0：未达到超时值 1：已达到超时值，未接收到任何数据 注：如果 RTOR 寄存器中编程的时间值将 2 个字符隔开，则 RTOF 不置 1。如果此时间大于该值+2 个采样时间（2/16 或 2/8，具体取决于过采样模式），则 RTOF 标志置 1。 即使 RE = 0，计数器仍会计数，但 RTOF 仅在 RE = 1 时置 1。如果 RE 置 1 时已经超时，则 RTOF 将置 1。 如果 USART 不支持接收器超时功能，该位保留且保持复位值。
18	EOBF	RC_W0	0	块结束标志 接收到完整块后，此位由硬件置 1（例如 T=1 智能卡模式）。接收到的字节数（从块的起始处，包括起始字段）等于或大于 BLEN + 4 时执行检测。 如果 EOBIE=1，则产生中断。软件写 0 清零。0：未达到块结束 1：已达到块结束（字符数） 注：如果不支持智能卡模式，该位保留且保持复位值。
17:13	Reserved	-	-	保留
12	ABRRQ	RW	0	自动波特率请求。 该位写 1 会复位 ABRF 标志位，并且请求下一帧的自动波特率检测。

11	ABRE	R	0	自动波特率错误标志。 当自动波特率检测出错（波特率超出范围或者字符比较错误）时，硬件置位该寄存器。 软件通过写 1 到 ABRRQ 寄存器清零该位。
10	ABRF	R	0	自动波特率检测标志。 当自动波特率设置（同时设置 RXNE=1，当中断使能后产生中断），或者自动波特率检测操作出错（ABRE=1，RXNE=1，FE=1）时该位由硬件置 1。 软件通过写 1 到 ABRRQ 位清零该位。
9	CTS	RC_W0	0	CTS 标志。 如果配置 CTSE=1，则当 CTS 输入状态变化，该寄存器为 1。软件写 0 清零。当 CTSIE=1 时，产生 CTS 中断。 0：CTS 线上的值未改变 1：CTS 线上的值改变
8	LBD	RC_W0	0	LIN 断开帧检测标志。 当检测到 LIN 断开帧时，硬件配置该寄存器。软件写 0 清零。当 LBDIE=1 时产生中断。 0：未检测到 LIN 断开帧； 1：检测到 LIN 断开帧； 注：该寄存器如果不支持 LIN 功能，则固定为 0。
7	TXE	R	1	传输寄存器空标志。 发送时，当 USART_DR 寄存器数据传送到移位寄存器，硬件置位该寄存器。当 TXEIE=1 时，产生中断。 写 USART_DR 寄存器会清零该位。 0：数据未传送到移位寄存器 1：数据传送到移位寄存器
6	TC	RC_W0	1	传送完成标志。 传送数据帧完成后，且 TXE=1，则硬件置位该寄存器。TCIE=1 时产生中断。 软件先读 USART_SR 寄存器然后写 USART_DR 寄存器会清零该位（针对多处理器通讯）。软件同时可以写 0 清零。 0：传送未完成 1：传送完成
5	RXNE	RC_W0	0	接收数据寄存器不空标志。 接收时，当移位寄存器值传送到 USART_DR 寄存器，硬件置位该寄存器。 软件读 USART_DR 寄存器或向该位写 0 清零该位。 当 RXNEIE=1 时，产生中断。 0：未收到数据 1：接收到数据，可以读出
4	IDLE	R	0	空闲标志。 检测空闲帧，硬件置位该寄存器。当 IDLEIE=1 时产生中断。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。

				0：未检测到空闲帧 1：检测到空闲帧
3	ORE	R	0	溢出错误标志。 当 RXNE=1 时，在移位寄存器中接收到的数据正准备转移到 RDR 寄存器时，硬件设置该位。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 当 RXNEIE=1 时，产生中断。 0：未产生溢出错误 1：产生溢出错误 注：该寄存器置位时，接收 DR 寄存器内容不会丢失，但移位寄存器内容被覆盖。 当 EIE 或者 RXNEIE=1 时，产生 ORE 中断。
2	NE	R	0	噪声错误标志。 在数据帧接收到噪声时，硬件置位该寄存器。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 0：未检测到噪声错误 1：检测到噪声错误 注：当 RXNE 与 NE 同时产生时，NE=1 时不产生中断，而在设置 RXNE 标志时产生中断。在多缓冲器通讯模式下，当 EIE=1 时 NE=1 会产生中断。
1	FE	R	0	帧错误标志。 当检测到不同步、过多的噪声或断开字符时，由硬件设置此位。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 0：未检测到帧错误 1：检测到帧错误或断开字符 注：当 RXNE 与 FE 同时产生时，FE=1 时不产生中断，而在 RXNE=1 时产生中断。如果当前传输的数据既产生了帧错误，又产生了过载错误，硬件还是会继续该数据的传输，并且只设置 ORE 标志位。 在多缓冲器通讯模式下，当 FE=1 且 EIE=1 时会产生中断。
0	PE	R	0	校验值错误。 当接收时校验值错误时，硬件置位该寄存器。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。但软件在清该位前必须等待 RXNE=1。 当 PEIE 时，产生中断。 0：未产生奇偶校验错误 1：产生奇偶校验错误

31.6.2. USART 数据寄存器 (USART_DR)

偏移地址: 0x04

复位值: undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	DR[8:0]	RW	undefined	接收/发送数据寄存器。 取决于读还是写操作，前者是接收到的数据，后者是发送的数据。 DR 寄存器物理上由两个寄存器组成（一个是发送的 TDR，一个是接收的 RDR），所以 DR 寄存器实现了读和写的两个功能。 TDR 寄存器在内部总线和输出移位寄存器之间提供了并行的接口，RDR 寄存器在输入移位寄存器和内部总线之间提供了并行接口。 当奇偶校验使能打开进行发送操作时，写 MSB 位（bit7 或者 bit8）是无效的，因为已被校验位代替了。 当使能奇偶校验进行接收操作时，读出的 MSB 位是接收到的校验位。

31.6.3. USART 波特率寄存器 (USART_BRR)

偏移地址:0x08

复位值:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV_Mantissa[11:0]												DIV_Faction[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

注意：只有在禁止 USART (USART_CR1.UE=0) 时才能写入此寄存器。在自动波特率检测模式，硬件更新该寄存器。

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:4	DIV_Mantissa[15:4]	RW	0	12bit 整数
3:0	DIV_Fraction[3:0]	RW	0	4bit 小数

31.6.4. USART 控制寄存器 1 (USART_CR1)

偏移地址:0x0C

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	DATAINV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWAP	Res.	Res.
				RW									RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSBFIRST	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27	DATAINV	RW	1'b0	二进制数据反向。 0: 按正/正向逻辑发送/接收数据寄存器中的逻辑数据（1=H，0=L）

				1: 按负/反向逻辑发送/接收数据寄存器中的逻辑数据（1=L，0=H） 注：奇偶校验位同样按此定义翻转。 该位域只能在 USART 被禁用(UE = 0)时写入。 注 1：同步模式，该寄存器需要配置为 0. 注 2：自动波特率检测时，该寄存器配置为 1，还是按照 0x7F 或者 0x55 检测，即收到的数据需要反向后为 0x7F 或者 0x55。这种情况下 DR 寄存器收到的数据会做 DATAINV 的调整。
26:19	Reserved	-	-	保留
18	SWAP	RW	1’b0	交换 TX/RX 引脚。 0: 使用标准引脚中定义的 TX/RX 引脚 1: 交换 TX 和 RX 引脚功能。这使得可以在与另一个 USART 交叉连接的情况下工作。 该位域只能在 USART 被禁用(UE = 0)时写入。
17:16	Reserved	-	-	保留
15	MSBFIREST	RW	1’b0	最高有效位在前。 0: 在起始位之后，首先使用数据位 0 发送/接收数据。 1: 在起始位之后，首先使用 MSB(位 7/8)发送/接收，数据。 该位域只能在 USART 被禁用(UE = 0)时写入。 注 1：同步模式，该寄存器需要配置为 0. 注 2：自动波特率检测时，该寄存器配置为 1，还是按照 0x7F 或者 0x55 检测，即收到的数据需要为 0x7F 或者 0x55。这种情况下 DR 寄存器收到的数据会做 MSBFIRST 的调整。
14	Reserved	-	-	保留
13	UE	RW	0	USART 使能。 当该位清零后，USART 模块会立即停止当前操作。 该位由软件置位和清零。 0：禁止 USART 预分频器和输出，低功耗模式 1：使能 USART 注：在清零 UE 位之前 DMA 通道需要禁止，TE 和 RE 也要关闭。
12	M	RW	0	0：1 个起始位，8 个数据位，n 个停止位 1：1 个起始位，9 个数据位，n 个停止位 该位域只能在 USART 被禁用(UE = 0)时写入。
11	WAKE	RW	0	接收唤醒方式。 从静默模式唤醒方式。由软件置位或者清零。 0：空闲帧唤醒 1：地址唤醒 该位域只能在 USART 被禁用(UE = 0)时写入。
10	PCE	RW	0	奇偶校验控制。 0：奇偶校验禁止 1：奇偶校验使能

				奇偶校验位：9 位数据帧的第 9 位；8 位数据帧的第 8 位。 该位域只能在 USART 被禁用(UE = 0)时写入。
9	PS	RW	0	奇偶校验选择。由软件置位和清零。 0：偶校验 1：奇校验 该位域只能在 USART 被禁用(UE = 0)时写入。
8	PEIE	RW	0	PE 中断使能。由软件置位和清零。 0：禁止 1：PE 中断使能
7	TXEIE	RW	0	TXE 中断使能。由软件置位和清零。 0：禁止 1：TXE 中断使能
6	TCIE	RW	0	传送结束中断使能。由软件置位和清零。 0：禁止 1：TC 中断使能
5	RXNEIE	RW	0	RXNE 中断使能；由软件置位和清零。 0：禁止 1：ORE 或者 RXNE 中断使能
4	IDLEIE	RW	0	IDLE 中断使能。由软件置位和清零。 0：禁止 1：IDLE 中断使能
3	TE	RW	0	传送使能。 0：传送禁止 1：传送使能 注：在关闭 UE 时需要同时关闭 TE；在使能 TE 时需要先使能 UE 或者同时使能 UE。
2	RE	RW	0	接收使能。 0：接收禁止 1：接收使能，开始检测起始位
1	RWU	RW	0	接收唤醒。 该位表明 USART 是否为静默模式。 当接收到静默模式序列，该寄存器置位；如果接收到唤醒序列，该寄存器清零。具体哪种唤醒序列（地址或者空闲帧）由寄存器 USART_CR1.WAKE 位控制。 0：接收器为工作模式 1：接收器为静默模式 注 1：在设置该位进入静默模式前，USART 要已经先接收了一个数据字节，否则在静默模式下，不能被空闲总线检测唤醒。 注 2：当配置成地址标记检测唤醒（WAKE=1），在 RXNE 被置位时，不能用软件修改 RWU 位。
0	SBK	RW	0	发送断开帧。 软件置位该寄存器，发送断开字节。断开帧的停止位发送后，硬件清零该寄存器。

				0：不发送断开字节 1：发送断开字节

31.6.5. USART 控制寄存器 2 (USART_CR2)

偏移地址:0x10

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	RREPEN	TREPEN	REP_T	ERSW[1:0]		RTOEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	RW	RW	RW	RW	RW	RW									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	Res.	ADD[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW			RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	RREPEN	RW	1’b0	该寄存器适用于智能卡模式。 接收数据奇偶校验错的处理方式选择. 0：奇偶校验错，不自动发送 NACK，置 PE 标志，当 PEIE=1 时产生中断。 1：奇偶校验错，根据 T=0 协议自动回发 NACK。单一字节连续接收次数超过 REP_T 后，置 PE 标志，当 PEIE=1 时产生中断。 该位域只能在 SCEN= 1 时写入。 RREPEN 与 NACK 寄存器配置不同值时，发送 NACK 和置位 PE 的情况如下:
29	TREPEN	RW	1’b0	该寄存器适用于智能卡模式。 发送数据奇偶校验错的处理方式选择。 0：收到 NACK 时不进行自动回发，硬件置 TXPE 标志，若开启 PEIE，则产生中断; 1：收到奇偶校验出错标志（NACK），根据 T = 0 协议自动进行回发。在单一字节重复发送次数超过 REP_T 后，硬件置 TXPE 标志，若开启 PEIE，则产生中断; 该位域只能在 SCEN=1 时写入。
28	REP_T	RW	1’b0	该寄存器适用于智能卡模式。 控制接收数据奇偶校验出错时自动重发次数 0：1 次

RREPEN	NACK	描述
0	0	奇偶校验错误，不发 NACK，置位 PE。
0	1	奇偶校验错误，不发 NACK，置位 PE。
1	0	奇偶校验错误，不发 NACK，连续接收次数超过 REP_T 后置位 PE。
1	1	奇偶校验错误，发 NACK，连续接收次数超过 REP_T 后置位 PE。

				1：3 次 该位域只能在 SCEN= 1 时写入。
27:26	ERSW	RW	2’b0	NACK 宽度选择。 11：NACK 宽度为 1bit; 10：NACK 宽度为 1.5bit; 01：NACK 宽度为 2bit; 00：NACK 宽度为 2bit; 该位域只能在 SCEN= 1 时写入。 该寄存器适用于智能卡模式。
25	RTOEN	RW	1’b0	接收器超时使能。 此位由软件置 1 和清零。 0：禁止接收器超时功能。 1：使能接收器超时功能。 使能此功能后，如果 RX 线路在 RTOR（接收器超时寄存器）中编程的持续时间内处于空闲状态（无接收），则 USART_SR 寄存器中的 RTOF 标志置 1。
24:15	Reserved	-	-	保留
14	LINEN	RW	0	LIN 模式使能。 软件置位和清零。 0：LIN 模式禁止； 1：LIN 模式使能； LIN 模式下，通过使能 SBK 位，发送 LIN 同步断开帧（13 低位）。 注：该寄存器如果不支持 LIN 功能，则固定为 0.
13:12	STOP[1:0]	RW	2’b0	Stop 位配置。 00：1 stop bit； 01：0.5stop； 10：2 stop 位； 11：1.5stop；
11	CLKEN	RW	0	CK 引脚使能。 0：CK 引脚禁止； 1：CK 引脚使能； 不支持同步模式时，该位保留。
10	CPOL	RW	0	时钟极性。 同步模式，CK 引脚输出时钟极性选择。 0：空闲时，CK 引脚为稳定低值； 1：空闲时，CK 引脚为稳定高值； 只有当 UE=0 时才能写入此位。
9	CPHA	RW	0	该位在同步模式下用于选择 CK 引脚输出时钟的相位。它与 CPOL 位一起工作，以产生所需的时钟/数据关系。 0：第一个时钟传输是首个数据捕获沿； 1：第二个时钟传输是首个数据捕获沿； 只有当 UE=0 时才能写入此位。
8	LBCL	RW	0	最后一位数据的时钟脉冲是否在 CK 引脚输出。 0：最后一位数据的时钟脉冲不在 CK 引脚输出；

				1：最后一位数据的时钟脉冲在 CK 引脚输出； 只有当 UE=0 时才能写入此位。
7	Reserved	-	-	保留
6	LBDIE	RW	0	LIN 断开中断使能。 0：禁止； 1：中断产生； 注：该寄存器如果不支持 LIN 功能，则固定为 0.
5	LBDL	RW	0	LIN 断开帧检测长度。 0：10 位断开帧检测； 1：11 位断开帧检测； 注：该寄存器如果不支持 LIN 功能，则固定为 0.
4	Reserved	-	-	保留
3:0	ADD[3:0]	RW	4’b0	USART 地址。 该寄存器用于多处理器静默模式，用作 4 位地址唤醒时的地址。

31.6.6. USART 控制寄存器 3 (USART_CR3)

偏移地址:0x14

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	EOBIE	RTOIE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	RW	RW													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	ABRMOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	EOBIE	RW	1’b0	EOB 中断使能。 0：禁止； 1：中断使能；
29	RTOIE	RW	1’b0	RTO 中断使能。 0：禁止； 1：中断使能；
28:15	Reserved	-	-	保留
14:13	ABRMOD[1:0]	RW	2’b0	自动波特率检测模式。 00：从开始位开始测量波特率 01：下降沿到下降沿测量 10：0x7F 帧检测方式 11：0x55 帧检测方式 当 ABREN=0 或者 UE=0 时，才能配置该寄存器。
12	ABREN	RW	0	自动波特率使能。 0：自动波特率禁止 1：自动波特率使能
11	OVER8	RW	0	过采样模式。 0：16 倍过采样 1：8 倍过采样

				该位仅在 UE=0 时可被写。 注：使能同步模式或者 LIN 模式时，该寄存器需要配置为 0.
10	CTSIE	RW	0	CTS 中断使能。 0：CTS 中断禁止； 1：CTS 中断使能；
9	CTSE	RW	0	CTS 使能。 0：CTS 硬件流控制模式禁止； 1：CTS 硬件流控制模式使能。当有当 CTS 输入为 0 时，才会传输数据。此时，当数据写入数据寄存器后，要等待 CTS 有效后才会启动传输。
8	RTSE	RW	0	RTS 使能。 0：RTS 硬件流控制禁止； 1：RTS 硬件流控制使能，只有当接收缓冲区未满时才会请求下一个数据。当前数据发送完成后，发送操作暂停。如果可以接收数据了，将 RTS 置为有效（0）。
7	DMAT	RW	0	发送时使能 DMA。 0：禁止； 1：发送时使能 DMA；
6	DMAR	RW	0	接收时使能 DMA。 0：禁止； 1：接收时使能 DMA；
5	SCEN	RW	0	智能卡模式使能。 0：智能卡模式禁止； 1：智能卡模式使能； 当 UE=0 时，才能配置该寄存器。
4	NACK	RW	0	智能卡 NACK 使能。 0：奇偶校验错误时发送 NACK 禁止； 1：奇偶校验错误时发送 NACK 使能；
3	HDSEL	RW	0	半双工选择。 0：非半双工模式； 1：选择半双工模式； 当 UE=0 时，才能配置该寄存器。
2	IRLP	RW	0	IrDA 低功耗。 0：正常模式； 1：IrDA 低功耗模式； 当 UE=0 时，才能配置该寄存器。
1	IREN	RW	0	IrDA 模式使能。 软件使能和清零该寄存器。 0：IrDA 禁止； 1：IrDA 使能； 当 UE=0，TE=0，RE=0 时，才能使能该寄存器。
0	EIE	RW	0	错误中断使能。 0：禁止； 1：帧错误 FE、溢出错误 ORE、噪声 NF 中断使能。

31.6.7. USART 保护时间和预分频寄存器（USART_GTPR)

偏移地址:0x18

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GT[7:0]								PSC[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:8	GT[7:0]	RW	0	保护时间值。 该域定义了以波特时钟为单位的保护时间。在智能卡模式，需要此功能。当保护时间过去后，才会设置发送完成标志。 注：该寄存器如果不支持智能卡功能，则固定为 0.
7:0	PSC[7:0]	RW	0	预分频器值。 ■ 在红外(IrDA)低功耗模式下: PSC[7:0]=红外低功耗波特率。 对系统时钟分频以获得低功耗模式下的频率: 源时钟被寄存器中的值(仅有 8 位有效)分频 00000000：保留 – 不要写入该值; 00000001：对源时钟 1 分频; 00000010：对源时钟 2 分频; …… ■ 在红外(IrDA)的正常模式下：PS 只能设置为 00000001。 ■ 在智能卡模式下: PSC[4:0]：预分频值 对系统时钟进行分频，给智能卡提供时钟。 寄存器中给出的值(低 5 位有效)乘以 2 后，作为对源时钟的分频因子。 00000：保留–不要写入该值; 00001：对源时钟进行 2 分频; 00010：对源时钟进行 4 分频; 00011：对源时钟进行 6 分频; …… 11111：对源时钟进行 62 分频; 注意：位[7:5]在智能卡模式下没有意义。

31.6.8. USART 接收超时寄存器（USART_RTOR)

偏移地址:0x1C

复位值: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLEN[7:0]								RTO[23:16]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTO[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	BLEN	RW	0	块长度。 该寄存器定义智能卡模式下 T=1 接收的块长度。 它的值等于信息字符数+结尾字段长-1 例如： BLEN =0：0 信息字符+LEC BLEN =1：0 信息字符+CRC BLEN = 255:254 个信息字符+CRC 在智能卡模式，当 TXE=0 时块长度计算被复位。 这个位域也可以在其他模式中使用。在这种情况下，当 RE = 0(接收器禁用)和/或 EOBF 位被写入 0 时，块长度计数器被重置。
23:0	RTO[23:0]	RW	0	在 RX 线上没有活动时的接收超时时间。 在标准模式下，在设定的时间下，如果在接收到最后一个字符后，在 RTO 对应的时间内未检测到新的起始位，则 RTOF 标志位置起。 在智能卡模下，这个值用来执行 CWT 和 BWT。在这个情况下，CWT 和 BWT 是从最后接收到的字符起始位开始计算的。

32. 通用异步收发器 (UART)

32.1. UART 简介

本项目设计实现了 2 个 UART 模块，2 个功能完全一致。

UART 是一种可编程通用异步收发器。

32.2. UART 主要特性

- AMBA APB 接口
- 支持 5/6/7/8/9 位串行数据
- 支持 1/2 位 STOP 位（5 位数据时：1/1.5 位 STOP）
- 支持发送地址/数据
- 支持固定奇偶校验
- 支持断开帧
- 起始位错误检测
- 支持可编程分数波特率：可编程串行数据波特率，计算如下：波特率=（串行时钟频率）/（16*除数）
- 支持 4 位小数波特率
- 支持 Tx/Rx 引脚互换功能
- 支持大小端切换 MSB FIRST 功能
- 支持 DMA 传输

32.3. UART 功能描述

32.3.1. UART 框图

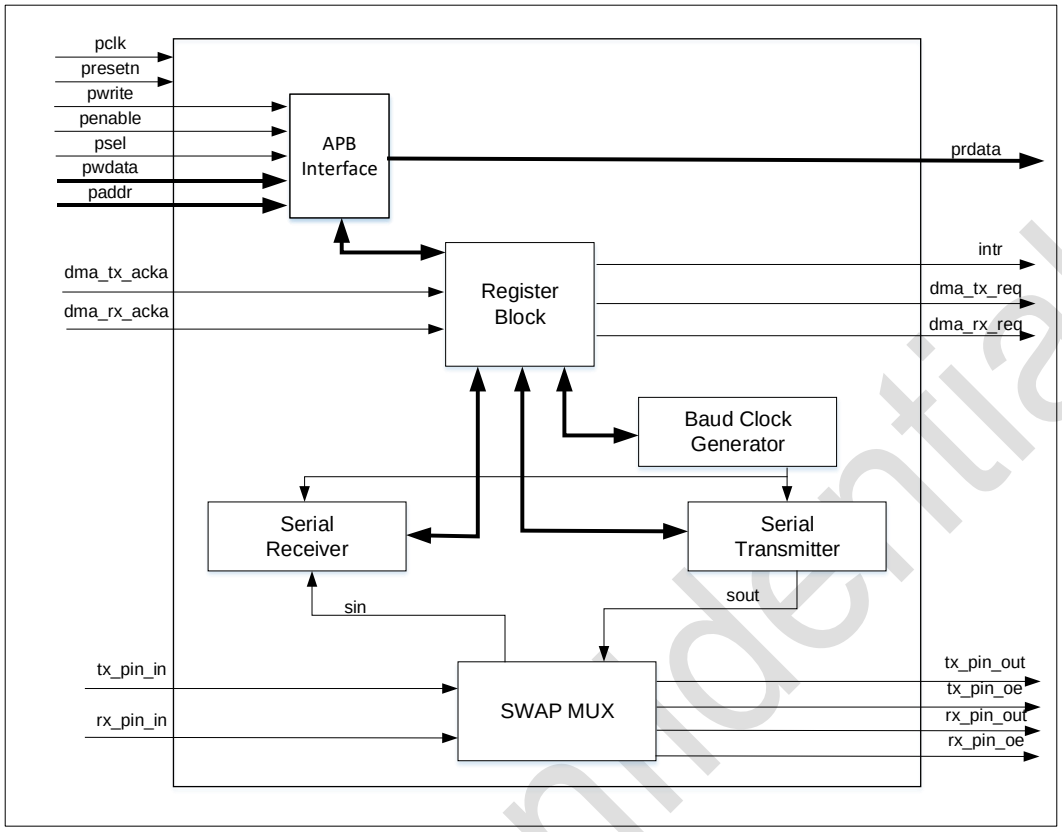


图 32-1 UART 框图

注：1.粗体线表示存在且为总线。

主要模块：

- APB 从接口（APB interface）---APB 总线接口。
- 寄存器块（Register Block）---负责 UART 的主要功能，包括控制、状态和中断生成。
- 波特时钟生成器（Baud Clock Generator）---生成发送器和接收器波特时钟以及输出参考时钟信号。
- 串行发送器（Serial Transmitter）---将写入 UART 的并行数据转换为串行形式，并添加控制寄存器指定的所有附加位，用于传输。
- 串行接收器（Serial Receiver）---将 UART 中接收的控制寄存器指定的串行数据字符转换为并行形式。此块控制：
  - 奇偶校验错误检测
  - 帧错误检测
  - 断开帧检测
- 引脚互换处理（SWAP MUX）---SWAP 使能时的引脚处理

32.3.2. UART(RS232)串行协议

因为 UART 和所选设备之间的串行通信是异步的，所以在串行数据中添加了额外的位（起始和停止）来指示开始和结束。利用这些比特可以使两个设备同步。这种由起始位和停止位组成的串行数据结构被称为一个字符，如下图所示。

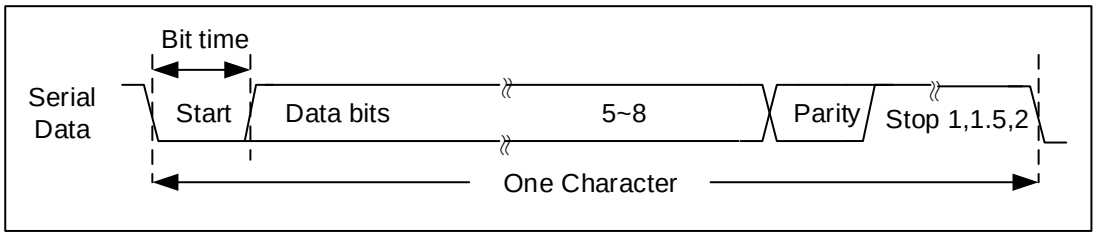


图 32-2 串行数据格式

一个额外的奇偶校验位可以添加到串行字符。该位出现在字符结构中的最后一个数据位之后和停止位之前，以便为 UART 提供对接收到的数据执行简单错误检查的能力。

UART 线路控制寄存器用于控制串行字符特性。数据字的各个位在起始位之后发送，从最低有效位（LSB）开始。它们后面是可选的奇偶校验位，后面是停止位，可以是 1、1.5 或 2。

注：UART 实现的停止位持续时间可能会更长，原因如下：

- 在某些配置的字符之间插入的空闲时间

传输中的所有比特都是在完全相同的持续时间内传输的；这方面的例外是当使用 1.5 个停止位时的半停止位。该持续时间被称为比特周期或比特时间；一位时间等于十六个波特时钟。

为了确保线路的稳定性，一旦检测到起始位，接收器就在位时间的大约中点对串行输入数据进行采样。因为每个比特传输的波特时钟的确切数量是已知的，所以计算采样的中点并不困难；即在起始位的中点采样之后每十六个波特时钟。

与串行输入去抖动一起，这种采样有助于避免错误起始位的检测。短故障通过去抖动被过滤掉，并且在线路上没有检测到转换。如果故障足够宽，可以通过去抖动来避免滤波，则会检测到下降沿。然而，只有当线路在半个采样周期后再次被采样为低时，才检测到起始位。

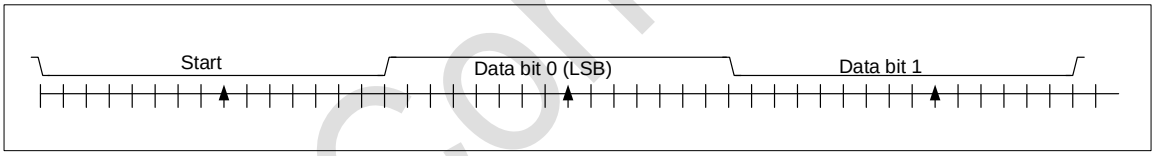


图 32-3 接收器串行数据采样点

作为 16550 标准的一部分，一个可选的波特时钟参考输出信号（baudout_n）为需要它的接收设备提供定时信息。UART 的波特率由单时钟实现中的串行时钟 pclk 以及分频锁存寄存器（BRR）控制。

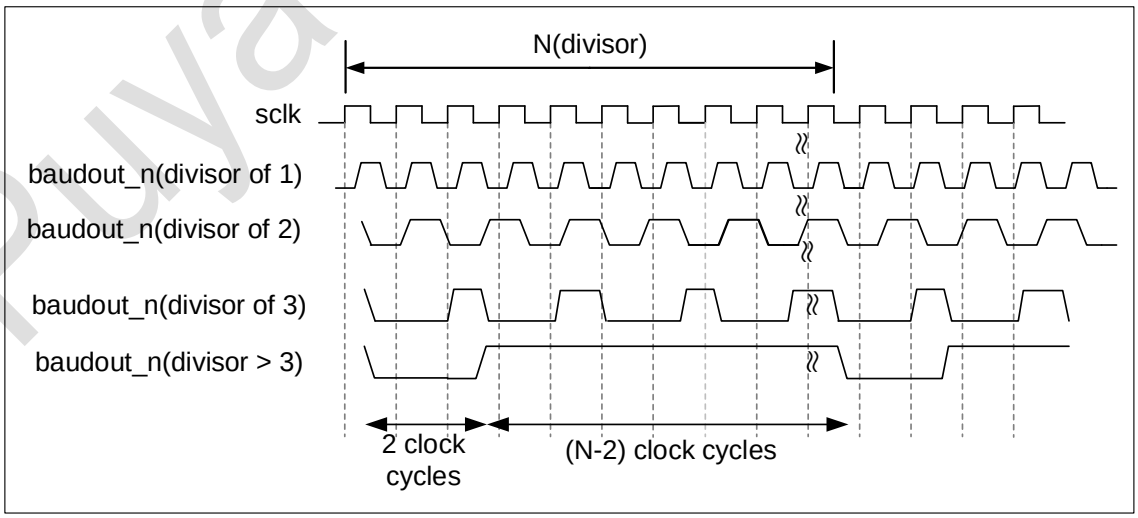


图 32-4 不同除数值的baudout_n输出的时序图

32.3.3. UART 9 位数据传输

UART 在发送和接收模式下都可以被配置为具有 9 位数据传输。字符中的第 9 位出现在字符的第 8 位

之后和奇偶校验位之前。下图显示了字符的串行传输，其中 D8 表示第 9 位，还显示了 9 位模式下的一般串行传输。（此为正常小端模式，如果是大端模式则调转 9 位数据位的顺序）

9 位数据

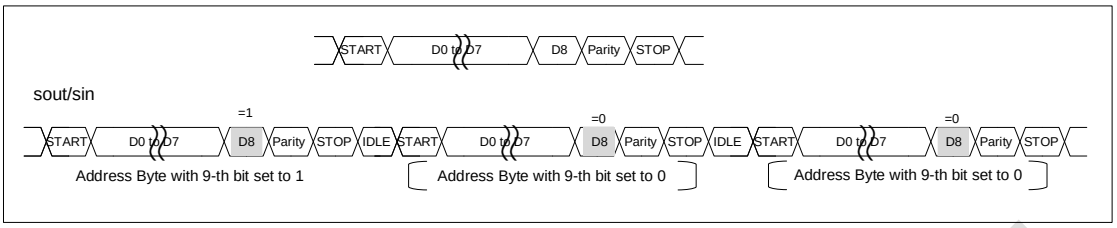


图 32-5 9位字符

通过启用 9 位数据传输模式，UART 可以用于多点系统，其中一个主机连接到系统中的多个从机。主机与其中一个从机交流。当主机想要将数据块传输到从机时，它首先发送一个地址字节来识别目标从机。

地址/数据字节之间的区分是基于输入字符中的第 9 位来完成的。如果第 9 位设置为 0，则该字符表示一个数据字节。如果第 9 位设置为 1，则该字符表示地址字节。所有从系统将地址字节与它们自己的地址进行比较，并且只有目标从机（其中地址匹配）能够从主机接收数据。主机开始向目标从机发送数据字节。未寻址的从机忽略传入的数据，直到接收到新的地址字节。

在上图中，请注意一个地址后面跟着 2 个数据字节。地址字节在第 9 位（D8）设置为 1 的情况下输出，而数据字节在第九位（D8）设置为 0 的情况下发出。奇偶校验位是一个可选字段。

用于 9 位数据传输的 UART 的配置执行以下操作：

- UART_CR3.M_E 位用于启用或禁用 9 位数据传输。
- 在接收的情况下，UART_CR3.ADDR_MATCH 用于在基于硬件和软件的地址匹配之间进行选择。
- UART_CR3.SEND_ADDR 位用于在发送的情况下使能发送地址。
- UART_CR3.TX_MODE 位用于在基于硬件和软件的地址传输之间进行选择。
- UART_TAR 和 UART_RAR 寄存器分别用于发送地址和匹配接收到的地址。
- UART_DR（TDR/RDR）寄存器为 9 位寄存器，用于在 9 位模式下进行数据传输。
- UART_SR.ADDR_RCVD 位用于指示地址接收中断。

32.3.3.1. 发送模式

UART 支持两种类型的传输模式：

- 传输模式 0（当 UART_CR3.TX_MODE 设置为 0 时）
- 传输模式 1（当 UART_CR3.TX_MODE 设置为 1 时）

模式 0

在传输模式 0 中，地址被编程在传输地址寄存器（TAR）中，数据被写入传输保持寄存器（TDR）。TDR 寄存器的第 9 位在此模式中不适用。

下图说明了基于 SEND_ADDR（CR3[2]）、TDR 空条件的地址和数据传输。

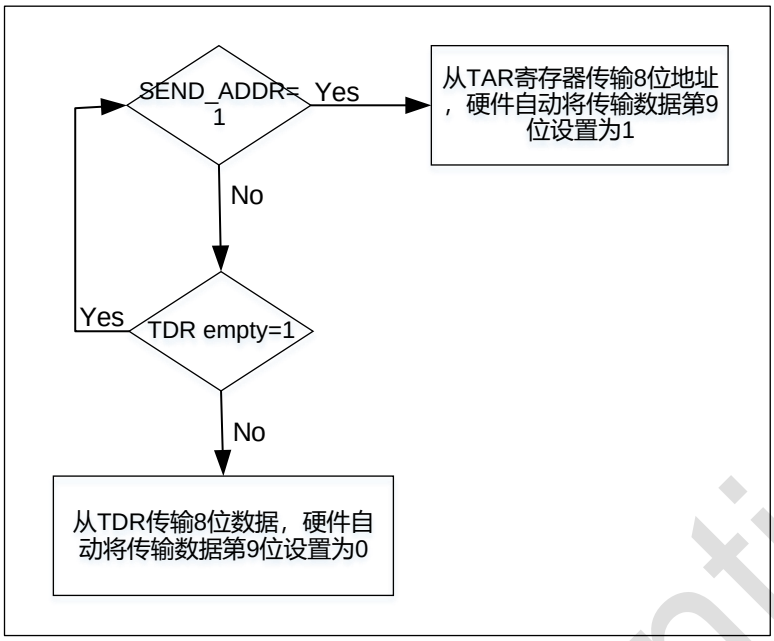


图 32-6 自动地址传输流程图

要向其传输数据的目标从机地址被编程在 TAR 寄存器中。必须启用 SEND_ADDR（CR3[2]）位来传输串行 UART 线上 TAR 寄存器中的目标从机地址，其中第 9 个数据位设置为 1，表示正在发送地址到从机。地址字符开始在线上传输后，UART 清除 SEND_ADDR 位。

传输到目标从机所需的数据通过传输保持寄存器（TDR）进行编程。数据在 UART 线上传输，第 9 个数据位设置为 0，表示正在发送数据到从机。

模式 1

在传输模式 1 中，TDR 寄存器为 9 位宽，地址和数据都通过 TDR 寄存器编程。UART 不区分地址和数据。SEND_ADDR（UART_CR3[2]）位和传输地址寄存器（UART_TAR）不适用于此模式。根据是否必须发送地址/数据，软件必须用 1/0 写入第 9 位。

M_E	TX_MODE	SEND_ADDR	使用场景
0	X	X	发 8 位 TDR 发数据
1	0	0	发“0+8 位 TDR 数据 “
1	0	1	发“1+8 位 TAR 地址”
1	1	X	发 “9 位 TDR 数据”

注：该表说明作为发送方可以使用的几种发送场景和对应的配置。

32.3.3.2. 接收模式

UART 支持两种接收模式：

- 硬件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 1 时）
- 软件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 0 时）

硬件地址匹配接收模式

在硬件地址匹配接收模式中，如果接收字符的第 9 位设置为 1，则 UART 将接收字符与在接收地址寄存器（UART_RAR）中编程的地址进行匹配：

如果接收到的地址与 UART_RAR 寄存器中的编程地址匹配成功，则随后接收数据字节。

如果地址匹配失败，则 UART 控制器丢弃数据字符，直到接收到匹配的地址为止。

下图显示了基于地址匹配功能的数据字节接收流程图。

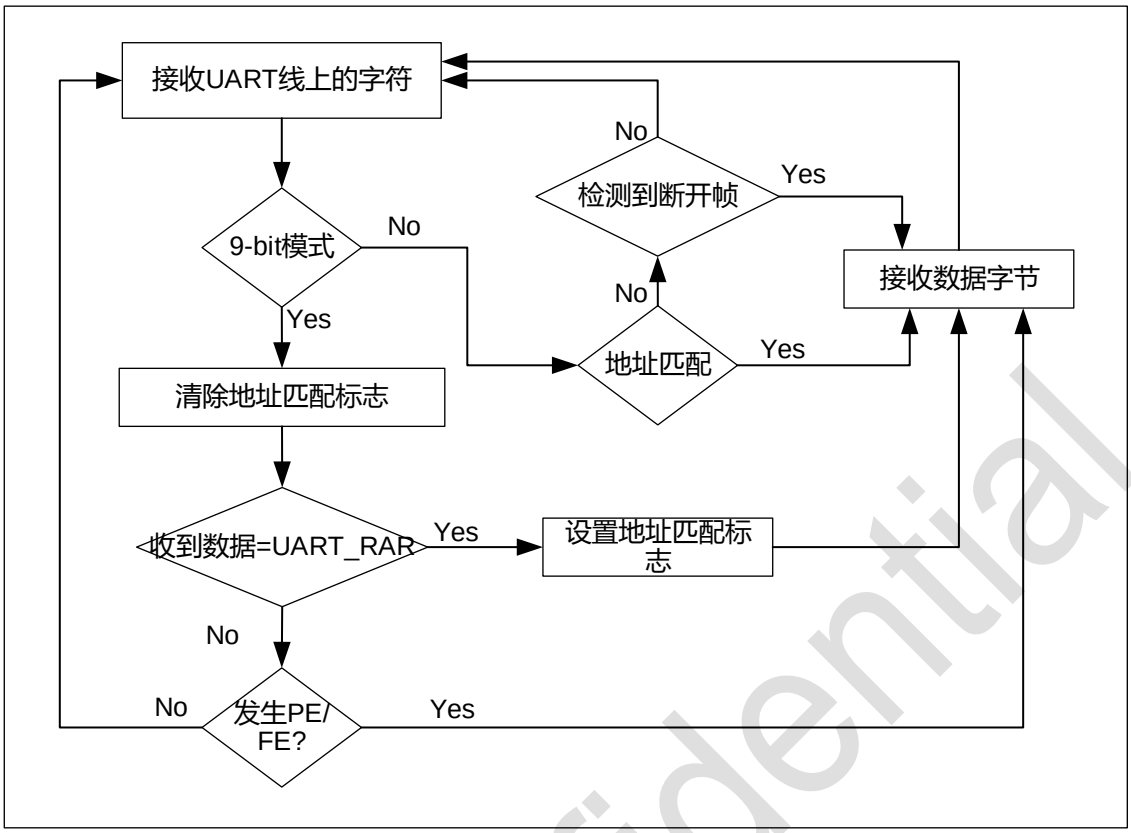


图 32-7 硬件地址匹配接收模式

UART 接收字符，而不管第 9 位数据是否设置为 1。如果接收到的字符的第 9 位被设置为 1，则它清除内部地址匹配标志，然后将接收到的 8 位字符信息与 UART_RAR 寄存器中编程的地址进行比较。

如果接收到的地址字符与 UART_RAR 寄存器中编程的地址匹配成功，则地址匹配标志被设置为 1，并且接收到的字符推送到 UART_RDR 寄存器，并且 UART_SR 寄存器中的 ADDR_RCVD 位被设置为指示地址已被接收，随后的数据字节（接收字符的第 9 位设置为 0）被推送到 UART_RDR。

如果地址与UART_RAR寄存器匹配失败并且(奇偶校验或者接收的地址字符中发现帧错误)的情况下，则接收的地址字符串仍然被推送到 UART_RDR 寄存器，ADDR_RCVD 和 PE/FE 错误位被设置为 1。

如果接收到任何中断字符，UART 将其视为一个特殊字符，并将其推送到 UART_RDR 寄存器，而不考虑地址匹配标志。

软件地址匹配接收模式

在这种操作模式中，UART 不执行与 UART_RAR 寄存器的接收地址字符（第 9 位数据设置为 1）的地址匹配。UART 始终接收 9 位数据，移位进 UART_RDR 寄存器。每当接收到地址字节并通过状态寄存器中的 ADDR_RCVD 位指示时，必须由用户自行比较地址。

32.3.4. UART 波特率

UART 的波特率由 PCLK 和分频锁存寄存器（UART_BRR）和小数波特率寄存器 UART_BRRF 控制。

波特率由以下因素决定：

- 串行时钟工作频率（PCLK）
- 所需波特率
- 波特率生成器除数值 divisor（由 UART_BRR 寄存器组成）
- 可接受的波特率误差%error

计算波特率的方程式如下：

波特率=串行时钟工作频率/（16* DIVISOR）——（1）

其中，DIVISOR–用于对 BRR 进行编程的数字（十六进制）。

串行时钟频率–UART 的 PCLK 频率。

根据等式（1），DIVISOR 可以计算为：DIVISOR=串行时钟频率/（16*波特率）

（等式算出的小数部分放在 UART_BRRF）

同样从等式（1）中，还可以示出：串行时钟频率= 波特率* 16* 除数

波特率和选定波特率定之间的误差如下所示：

百分比错误=（ |波特率-选定波特率| ）/波特率* 100%

表 32-1 设置波特率时的误差计算

波特率		f_{PCLK} = 4MHz			f_{PCLK} = 24MHz			f_{PCLK} = 48MHz		
序号	kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.404	104	0.16%	2.4	625	0.00%	2.4	1250	0.00%
2	9.6	9.615	26	0.16%	9.615	156	0.16%	9.615	312	0.16%
3	19.2	19.231	13	0.16%	19.231	78	0.16%	19.231	156	0.16%
4	57.6	62.5	4	8.51%	57.692	26	0.16%	57.692	52	0.16%
5	115.2	125	2	8.51%	115.385	13	0.16%	115.385	26	0.16%
6	230.4	250	1	8.51%	250	6	8.51%	230.769	13	0.16%
7	460.8	不可能	不可能	不可能	500	3	8.51%	500	6	8.51%
8	921.6	不可能	不可能	不可能	1500	1	62.76%	1000	3	8.51%
9	2250	不可能	不可能	不可能	不可能	不可能	不可能	3000	1	33.33%
10	4500	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能

注:

1. CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

2.在配置时钟的时候，由于波特率时钟是在系统时钟上进行分频的，但得到的时钟频率与实际时钟频率会有误差。配置的时钟误差需要在 2%以内，才可以正常工作。

32.3.5. UART 接收容忍度

只有当整体的时钟系统地变化小于 UART 异步接收器能够容忍的范围，UART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的变化(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足： DTRA + DQUANT + DREC + DTCL < UART 接收器的容忍度

对于正常接收数据， UART 接收器的容忍度等于最大能容忍的变化为 **96%~105%。**

32.3.6. UART DMA 通信

UART 可以利用 DMA 连续通讯。Rx 缓冲器和 Tx 缓冲器的 DMA 请求分别产生。

32.3.6.1. DMA 发送

通过配置 UART_CR3.DMAT 使能利用 DMA 发送。当发送数据寄存器为空，DMA 就从指定的存储区传送数据到 UART_DR 寄存器。

DMA 发送的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 USART 使用的 DMA 通道
- 配置 UART_TDR 寄存器地址作为 DMA 传输的目标地址。在每个发送数据寄存器为空事件后，数据将被传送到这个地址；
- 配置存储器（如 SRAM）地址作为 DMA 传输的源地址。在每次发送数据寄存器为空事件后，将从此存储器读数据加载到 UART_TDR 寄存器；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 使能 DMA 通道；
- 当传输达到配置的传输字节数，DMA 控制器产生中断请求。

32.3.6.2. DMA 接收

通过配置 UART_CR3.DMAR 使能利用 DMA 接收。每收到一个字节，DMA 就把数据从 UART_DR 寄存器传送到指定的 SRAM 区。

DMA 接收的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 USART 使用的 DMA 通道
- 配置 UART_RDR 寄存器地址作为 DMA 传输的源地址。在每个接收数据寄存器非空事件后，数据将从此 UART_RDR 寄存器地址读出数据传输到存储器；
- 配置 SRAM 存储器地址作为 DMA 传输的目标地址。在每次接收数据寄存器非空事件后，数据将从 UART_RDR 寄存器传送到此地址；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 使能 DMA 通道；
- 当传输达到配置的传输字节数，DMA 控制器产生中断请求。

32.4. UART 中断

UART 中断输出信号 (intr) 的断言——只要几个优先中断类型中的一个被启用并激活，就会发生中断。

以下中断类型可以通过 UART_CR2 寄存器启用：

- 接收器数据可用（RXNEIE）
- 发送寄存器空（TXEIE）
- 发送保持寄存器为空（在可编程 TDR 中断模式下）（TDREIE）
- 忙错误检测指示（BUSYERRIE）
- 接收线路状态（LSIE）

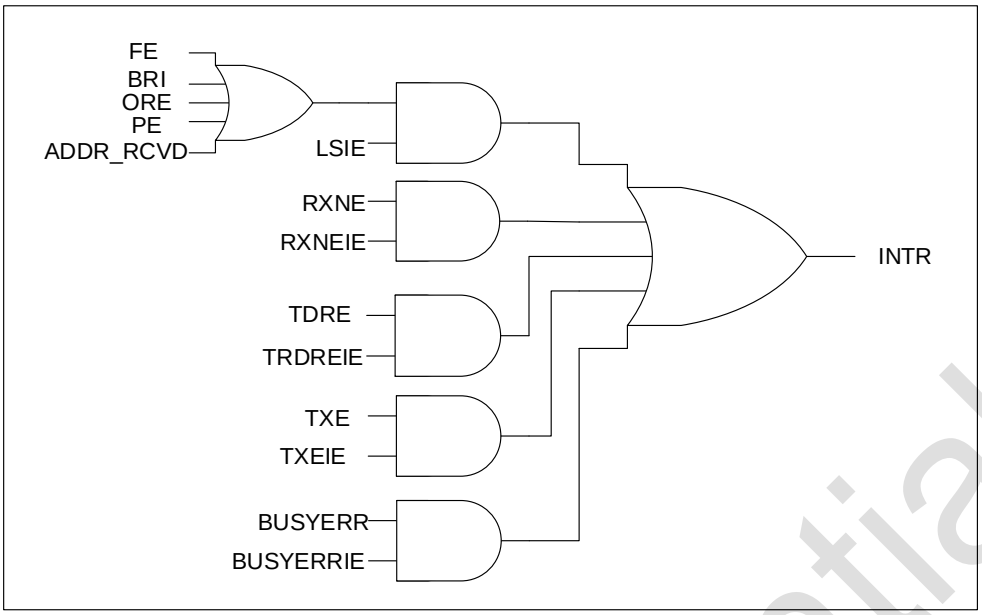


图 32-8 中断映射图

这些中断类型在下表中有详细说明。

表 32-2 中断控制功能

中断置位和清零功能		
中断类型	**中断源**	**中断清零控制**
接收线路状态	溢出/奇偶校验/帧错误中断或地址接收中断	对应位写 1 清 0
接收数据可用 RXNE	接收器数据可用	读取接收器缓冲寄存器
发送保持寄存器为空 TDRE	发送保持寄存器为空	写入 TDR
发送寄存器为空 TXE	发送寄存器为空	写入 TDR
忙错误检测 BUSYERR	UART 繁忙时（BUSY[0]设置为 1），主机尝试写入 CR1 寄存器。	对应位写 1 清 0

32.5. UART 寄存器

术语：设置=设置为 1；清除=清除为 0；

32.5.1. UART 数据寄存器 (UART_DR)

地址偏移：0x00

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	DR[8:0]	RW	9'b0	接收/发送数据寄存器。 一共是由两个寄存器组成（一个是发送的 TDR,一

				个是接收的 RDR），所以 DR 寄存器实现了读和写的两个功能。 RDR 寄存器是 UART 模式下串行输入端口上接收到的数据字节。只有状态寄存器（SR）中的接收非空（RXNE）位被设置时，该寄存器中的数据才有效。 必须在下一个数据到达之前读取 RDR 中的数据，否则它将被覆盖，从而导致溢出错误。 TDR 寄存器是在 UART 模式下串行输出端口上传输的数据。软件保证只有当 TDR 空（UART_SR.TDRE）位被设置时，数据才能再次写入 TDR。 如果 TDRE 被设置，则向 TDR 写入单个字符将清除 TDRE。在再次设置 TDRE 之前对 TDR 的任何额外写入都会导致 TDR 数据被重写。 仅当 UART_CR3.TX_MODE = 1 时，第 9 位才适用。

32.5.2. UART 波特率寄存器 (UART_BRR)

地址偏移：0x04

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BRR	RW	16'b0	该寄存器构成 16 位 divisor，divisor 包含 UART 的波特率除数。输出波特率等于串行时钟（PCLK）频率除以波特率除数值的十六倍，如下所示：波特率=（串行时钟频率）/（16*divisor）。 请注意，当除数锁存寄存器（BRR）设置为零时，波特时钟被禁用，不会发生串行通信。 此外，一旦设置了 BRR，在发送或接收数据之前，应等待至少 8 个时钟周期。

32.5.3. UART 状态寄存器 (UART_SR)

地址偏移：0x10

复位值：0x0000_0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	BUSY_ERR	BUSY	ADDR_RCVD	Res.	TXE	TDRE	BRI	FE	PE	ORE	RXNE
					RC_W1	R	RC_W1		R	O	RC_W1	RC_W1	RC_W1	RC_W1	R

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留

10	BUSY_ERR	RC_W1	1’b0	忙检测错误，检测到忙时的误操作。 0：无忙误操作错误。 1：UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 该位写 1 清零。
9	BUSY	R	1'b0	UART 忙 这表示串行传输正在进行中，清除时表示 UART 处于空闲或非活动状态。 在以下任何一种情况下，此位都将设置为 1（忙）： ---串行接口上正在进行发送 ---串行接口上正在进行接收 ---传输 TDR 中存在的数据，波特除数为非零（BRR 不等于 0） ---接收 RDR 中存在数据 注意：即使可能从另一个设备发送了新字符，UART 忙位也可能被清除。也就是说，如果 UART 在 TDR 和 RDR 中没有数据，并且没有正在进行的传输，并且新字符的起始位刚刚到达 UART。这是由于直到位周期的中间才看到有效的开始，并且该持续时间取决于已编程的波特除数。 0（IDLE）：UART 处于空闲或非活动状态 1（忙）：UART 忙（主动传输数据）
8	ADDR_RCVD	RC_W1	1'b0	地址接收。 如果启用 9 位数据模式（UART_CR3.M_E = 1），则该位用于指示接收数据的第 9 位被设置为 1。该位还可以用于指示输入字符是地址还是数据。 0：表示字符为数据。 1：表示字符为地址。 注意：用户需要确保在下一个地址字节到达之前清除中断（该位写 1 清 0）。 如果清除中断有延迟，则软件将无法区分多个地址相关的中断。
7	Reserved	-	-	保留
6	TXE	R	1'b1	发送为空。 每当当前没有发送数据且 TDR 为空的时候，就会设置此位。如果 TXEIE 使能，则会产生 TXE 中断 0：发送不为空 1：发送为空
5	TDRE	R	1'b1	发送保持寄存器空。 该位指示 TDR 为空。 每当数据从 TDR 传输到发送器移位寄存器且没有新数据写入 TDR 时，该位被设置。如果 TDRIE 使能，则会产生中断。 0：TDR 非空 1：TDR 为空
4	BRI	RC_W1	1'b0	断开中断位 这用于指示在串行输入数据上检测到中断序列。

				如果在 UART 模式下，每当串行输入保持在逻辑“0”状态的时间超过起始时间+数据位+奇偶校验位+停止位的总和时，就会设置它。对该位写 1 将清除 BRI 位。 0：未检测到断开序列 1：检测到断开序列
3	FE	RC_W1	1'b0	帧错误位。 这用于指示接收机中出现帧错误。当接收机在接收的数据中没有检测到有效的停止位时，就会发生帧错误。 应该注意的是，如果发生断开，也将设置帧错误（UART_SR.FE）位，如断开中断（UART_SR.BRI）位所示。之所以会发生这种情况，是因为断开字符通过将输入保持到逻辑 0 的时间长于字符的持续时间而隐式地生成帧错误。对该位写 1 将清除 FE 位。 0：无帧错误 1：帧错误
2	PE	RC_W1	1'b0	奇偶校验错误位。 如果设置了奇偶校验使能（UART_CR1.PEN）位，则该寄存器用于指示接收器发生奇偶校验错误。 应该注意的是，如果发生了断开，在这种情况下，如果奇偶校验生成和检测被启用（UART_CR1.PCE = 1）并且奇偶校验被设置为奇数（UART_CR1.PS = 0），则 PE 也会被设置。对该位写 1 将清除 PE 位。 0：无奇偶校验错误 1：奇偶校验错误
1	ORE	RC_W1	1'b0	溢出错误位。 这用于指示溢出错误的发生。 如果在读取以前的数据之前接收到新的数据字符，则会发生这种情况。 当新字符在从 RDR 读取前一个字符之前到达接收器时，设置 ORE 位。当这种情况发生时，RDR 中的数据将被覆盖。 对该位写 1 将清除 ORE 位。 0：无溢出错误 1：溢出错误
0	RXNE	R	1'b0	数据就绪位。 这用于指示接收器在 RDR 中至少包含一个字符。 读取 RDR 时，此位被清除。 0：数据未就绪 1：数据就绪

32.5.4. UART 控制寄存器 1 (UART_CR1)

地址偏移：0x14

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	MSBFIRST	SWAP	Res.	SBK	SP	PS	PCE	STOP	M[1:0]	
						RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	MSBFIRST	RW	1'b0	最高有效位在前。 0：起始位后，收发第 0 位数据； 1：起始位后，收发第 5/6/7/8/9 位数据； 在数据传输过程中，不能修改这个位。
8	SWAP	RW	1'b0	TX/RX 引脚互换。 0：TX/RX 引脚按照标准引脚映射定义； 1: TX/RX 引脚互换。此时用作交叉连接其他 UART 时。
7	Reserved	-	-	保留
6	SBK	RW	1'b0	断开控制位。 这用于将断开发送到接收设备。如果设置为 1，串行输出将强制进入间隔（逻辑 0）状态。输出线被强制为低电平，直到 SBK 位被清除。 0：释放串行输出以进行数据传输 1：串行输出被迫进入间隔状态
5	SP	RW	1'b0	固定奇偶校验。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。此位用于强制固定奇偶校验值。当 PCE、PS 和 SP 设置为 1 时，奇偶校验位被发送并检查为逻辑 0。如果 PCE 和 SP 被设置为 1，并且 PS 是逻辑 0，则奇偶校验位被发送并被检查为逻辑 1。如果此位设置为 0，则 SP（固定奇偶校验）被禁用。 0：固定奇偶校验已禁用 1：固定奇偶校验已启用
4	PS	RW	1'b0	偶数奇偶校验选择。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当奇偶校验被启用（PCE 设置为 1）时，这用于在偶数和奇数奇偶校验之间进行选择。如果设置为 1，则传输或检查偶数个逻辑“1”。如果设置为零，则传输或检查奇数个逻辑“1”。 0：奇校验 1：偶校验
3	PCE	RW	1'b0	奇偶校验启用。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。该位用于控制发送和接收的串行字符中的奇偶校验生成和检测。 0：禁用奇偶校验 1：启用奇偶校验
2	STOP	RW	1'b0	停止位的数量。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。这用于选择外围设备将发送和接收的每个字符的停止位数。如果设置为零，则在串行数据中传输一个停止位。如果设置为 1 并且数据位设置为 5（UART_CR1.M 设置为 0），则发送一个半停止位。否则，传输两个停止位。

				注意，无论所选择的停止位的数量如何，接收器将仅检查第一个停止位。 注：由于某些配置的字符之间插入的空闲时间和传输方向上的波特时钟除数值，UART 实现的停止位持续时间可能会更长； 0：1 个停止位 1：1.5/2 个停止位
1:0	M[1:0]	RW	2'b0	数据长度选择。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当 UART_CR3 中的 M_E 设置为 0 时，此寄存器用于选择外围设备将发送和接收的每个字符的数据位数。 0x0：每个字符 5 个数据位 0x1：每个字符 6 个数据位 0x2：每个字符 7 个数据位 0x3：每个字符 8 个数据位

32.5.5. UART 控制寄存器 2 (UART_CR2)

地址偏移：0x18

复位值：0x0000_0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	BUSYERRIE	LSIE	TDREIE	RXNEIE
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	TXEIE	RW	1'b0	启用发送空中断。这用于启用/禁用传输发送空中断的生成。 0：禁用发送空中断 1：使能发送空中断
3	BUSYERRIE	RW	1’b1	启用 BUSYERR 状态中断。这用于启用/禁用 BUSYERR 状态中断生成。UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 0：禁用 BUSYERR 状态中断 1：使能 BUSYERR 状态中断
2	LSIE	RW	1'b0	启用接收器线路状态中断。这用于启用/禁用接收器线路状态中断的生成。 该中断标志是 PE、FE、ORE、BRI、ADDR_RCVD 中断标志的组合。 0：禁用接收器线路状态中断 1：使能接收器线路状态中断
1	TDREIE	RW	1'b0	启用传输保持寄存器空中断。这用于启用/禁用传输保持寄存器空中断的生成。 0：禁用传输保持寄存器空中断 1：使能传输保持寄存器空中断
0	RXNEIE	RW	1'b0	启用接收数据可用中断。这用于启用/禁用接收数据

				可用中断。 0：禁用接收数据中断 1：使能接收数据中断

32.5.6. UART 控制寄存器 3 (UART_CR3)

地址偏移：0x1C

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA_SOFT_ACK	DMAT	DMAR	TX_MODE	SEND_ADDR	ADDR_MATCH	M_E
									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DMA_SOFT_ACK	RW	1’b0	软件清除 DMA 发送请求和 DMA 接收请求。 0：不影响 1：清除 DMA 发送请求和 DMA 接收请求
5	DMAT	RW	1'b0	传送时使能 DMA。 0：禁止 1：使能发送 DMA
4	DMAR	RW	1'b0	接收时使能 DMA。 0：禁止 1：使能接收 DMA
3	TX_MODE	RW	1'b0	传输模式控制位。该比特用于控制 9-bit 数据传输期间的传输模式的类型。 数值： 1：在这种操作模式下，传输保持寄存器（TDR）为 9 位宽。 用户需要确保 TDR 寄存器的地址/数据写入正确。 地址：第 9 位设置为 1 数据：第 9 位数设置为 0 注意：传输地址寄存器（UART_TAR）不适用于此操作模式。 0：在此操作模式中，传输保持寄存器（TDR）的宽度为 8 位。 用户需要将地址编程到传输地址寄存器（UART_TAR）中，并将数据编程到 TDR 寄存器中。
2	SEND_ADDR	RW	1'b0	发送地址控制位。此位用作控制旋钮，供用户在传输模式下确定何时发送地址。 0：9 位字符内容来自 TDR 寄存器 1：9 位字符内容：第 9 位设置为 1，其余 8 位将与“传输地址寄存器”中正在编程的内容相匹配。 注： 1.在发出地址字符后，该位由硬件自动清除。用户不

				应将此位编程为 0。 2.此字段仅在 M_E 位设置为 1 且 TX_MODE 设置为 0 时适用。
1	ADDR_MATCH	RW	1'b0	地址匹配模式。此位用于在接收期间启用地址匹配功能。 ■ 在地址匹配模式下，UART 将等待第 9 位设置为 1 的传入字符。并进一步检查地址是否与“接收地址匹配寄存器”中编程的地址匹配。如果匹配，则后续字符将被视为有效数据，UART 开始接收数据。 ■ 在正常模式下，UART 将开始接收数据，9 位字符将形成。用户负责读取数据并区分 b/n 地址和数据。 0：正常模式 1：地址匹配模式 注：仅当 M_E 设置为 1 时，此字段才适用。
0	M_E	RW	1'b0	9 数据使能 此位用于启用用于发送和接收传输的 9 位数据 0：禁止 9 位数据 1：使能 9 位数据

32.5.7. UART 接收地址寄存器 (UART_RAR)

地址偏移：0x20

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RAR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	RAR[7:0]	RW	8'b0	接收模式下的地址匹配寄存器。 如果在输入字符中第 9 位被设置为 1，则将对照该寄存器值检查剩余的 8 位。如果匹配成功，则第 9 位设置为 0 的后续字符将被视为数据字节，直到接收到下一个地址字节。 注： 1. 仅当“ADDR_MATCH”（UAR_CR3[1]）和“M_E”（UART_CR3[0]）位设置为 1 时，此寄存器才适用。 2. RAR 可以在任何时间点被编程。但是，当任何接收正在进行时，用户不得更改此寄存器值。

32.5.8. UART 发送地址寄存器 (UART_TAR)

地址偏移：0x24

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TAR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	TAR[7:0]	RW	8'b0	这是传输模式下的地址匹配寄存器。 如果 M_E 位被启用，并且“SEND_ADDR”（UART_CR3[2]）位被设置为 1，则 UART 将发送第 9 位设置为 1 的 9 位字符，剩余的 8 位地址将从该寄存器发送。 注： 1. 此寄存器仅用于发送地址。正常数据应通过编程 TDR 寄存器发送。 2. 在 UART 串行通道上开始发送地址后，硬件将自动清除“SEND_ADDR”位

32.5.9. UART 波特率小数寄存器 (UART_BRRF)

地址偏移：0x28

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BRRF[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	BRRF[3:0]	RW	4'b0	波特率小数部分。 小数值由（BRRF）/（2^4）确定。

33. 低功耗通用异步收发器（LPUART）

LPUART 是一种 UART，允许在有限功耗下双向 UART 通信。仅需 32.768 kHz LSE 时钟即可进行高达 9600 波特/秒的 UART 通信。当 LPUART 由与 LSE 时钟不同的时钟源驱动时，可以达到更高的波特率。

即使当微控制器处于低功耗模式，能耗极低时，LPUART 也会等待 UART 帧的到来。LPUART 包含所有必要的硬件支持，使在最小功耗下可以进行异步串行通信。

它支持半双工单线通信和调制解调器操作(CTS/RTS)，还支持多处理器通信。

DMA（直接存储器访问）可用于数据发送/接收。

33.1. LPUART 主要特性

- AMBA APB 接口
- 全双工异步通讯
- NRZ 标准模式（标记/空格）
- 波特率可编程
- 32.768 kHz 时钟，波特率范围 300~9600 波特/秒。更高波特率需要更高时钟频率支持
- 双时钟域：PCLK 及专用内核时钟
- 数据字长度可配置（7 位/8 位/9 位）
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 停止位数可配置（1/2 位）
- 单线半双工通讯
- 支持 DMA 连续传输
- 发送和接收独立使能
- 独立发送/接收信号极性控制
- Tx/Rx 引脚可以互换
- 调制解调器和 RS-485 收发器的硬件流控制
- 发送检测标志
  - 忙标志
  - 传输结束
- 奇偶校验控制：
  - 发送时产生奇偶校验位
  - 检查接收的数据字节的奇偶性
- 4 个错误标志
  - 上溢错误
  - 噪声检测
  - 帧结构错误
  - 奇偶校验错误
- 中断源标志
  - CTS 改变
  - 发送寄存器空
  - 发送完成
  - 接收数据寄存器非空

- 检测到总线空闲
- 溢出错误
- 帧错误
- 噪音操作
- 校验错误
- 地址字节匹配

- 多处理器通讯：当地址不匹配时 LPUART 进入静默模式
- 支持静默模式唤醒（空闲帧/地址标记检测）

33.2. LPUART 功能描述

33.2.1. LPUART 框图

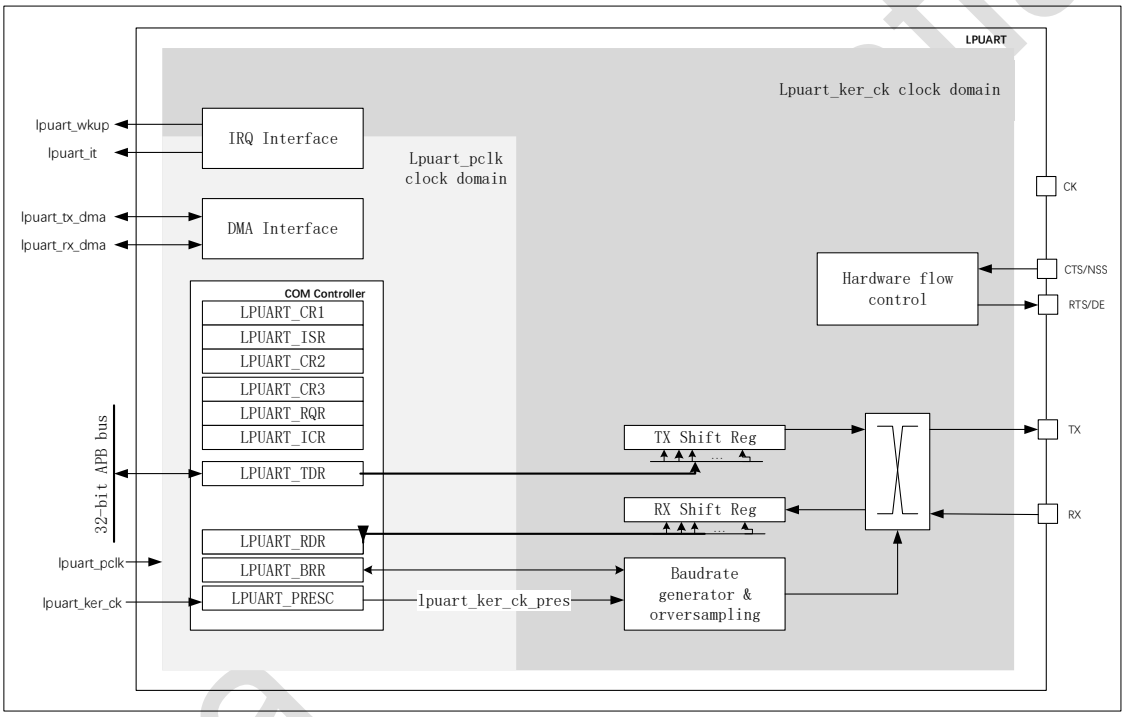


图 33-1 LPUART 框图

33.2.2. LPUART 信号

任何 LPUART 双向通信至少需要两个引脚：接收数据输入(RX)和发送数据输出(TX)。

- RX：接收数据串行输入。
- TX：发送数据输出。如果关闭发送器，该输出引脚模式由其 I/O 端口配置决定。如果使能了发送器但没有待发送的数据，则 TX 引脚处于高电平。在单线模式下，该 I/O 用于发送和接收数据。

33.2.2.1. 硬件流控制模式

- CTS：低电平开始发送，发送结束变为高电平
- RTS：低电平表明 LPUART 准备好接收数据

33.2.2.2. RS485 硬件流控制模式

- DE：驱动器使能，激活外部收发器的发送

注：DE 和 RTS 复用同一引脚

33.2.3. LPUART 字符描述

根据{M1，M0}配置，分为 7 位/8 位/9 位三种数据长度。

M[1:0]= 2’b10，7 位；

M[1:0]= 2’b00：8 位；

M[1:0]= 2’b01：9 位；

在默认情况下，信号（TX 或 RX）在起始位工作期间处于低电平状态。在停止位工作期间处于高电平状态。

通过极性配置控制，可以单独针对每个信号对这些值取反。

空闲字符定义：在 7 位/8 位/9 位数据长度+停止位时间内为全 1；

断开字符定义：在 7 位/8 位/9 位数据长度时间内为全 0；断开帧结尾，传送器插入 2 个停止位。即 8 位模式，断开帧长度为 10 位 0；9 位模式，断开帧长度为 11 位 0.

发送和接收操作由通用波特率发生器驱动。发送器和接收器的使能位置 1 时，将分别生成发送时钟和接收时钟。

下面是各个不同长度帧的波形。

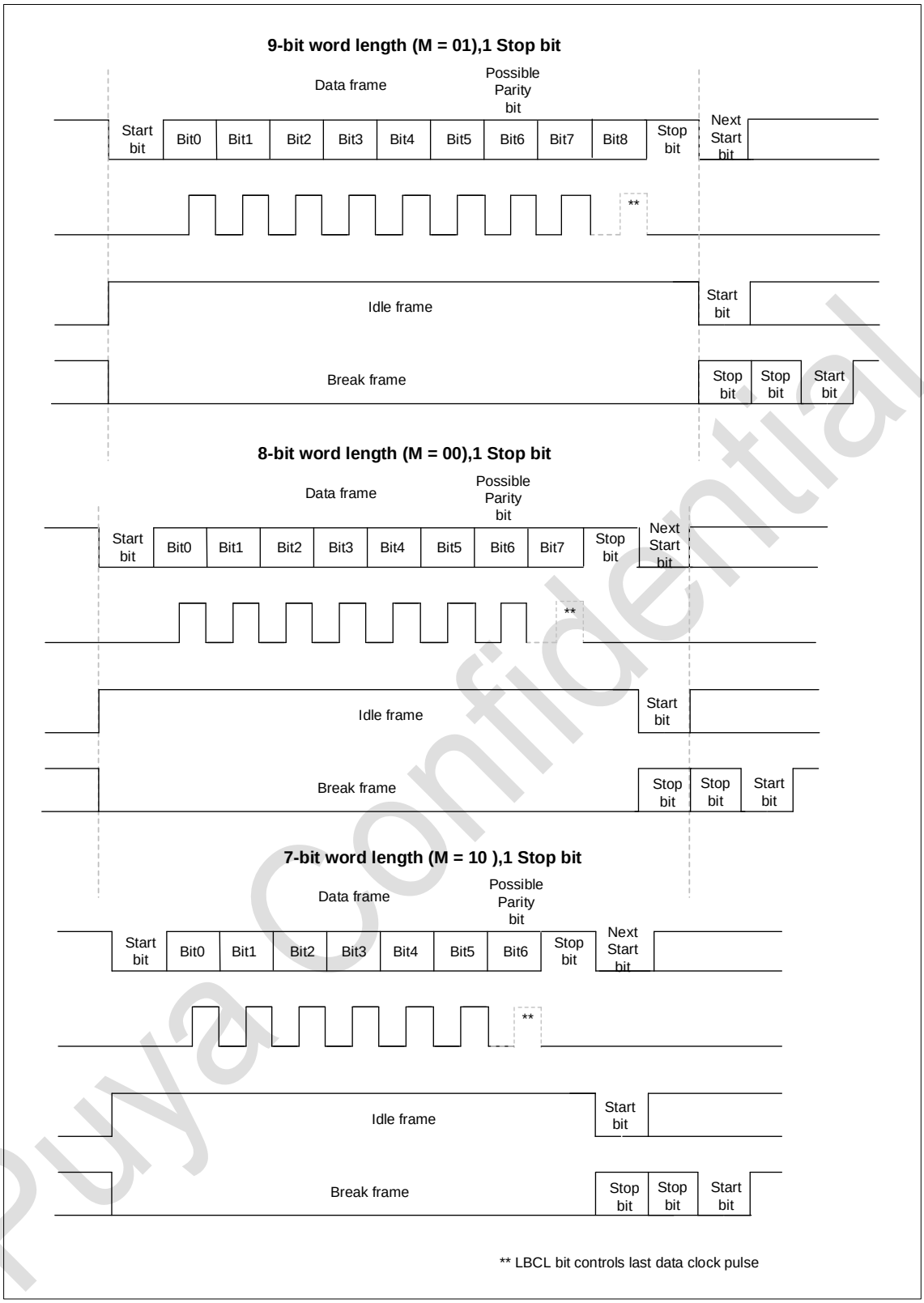


图 33-2 LPUART 字长编程

33.2.4. LPUART 发送

LPUART 发送步骤如下所述。

33.2.4.1. 发送步骤

- 配置 LPUART_CR1.{M1,M0}，定义传输数据宽度（7/8/9 位）；
- 配置 LPUART_BRR 寄存器，选择波特率；
- 配置 LPUART_CR2.STOP 寄存器，定义停止位个数(1/2 位);

- 配置 LPUART_CR1.UE=1,使能 LPUART;
- 多处理器传输时，配置 LPUART_CR3.DMAT=1，使能 DMA;
- 配置 TE=1，硬件插入空闲帧时序;
- 写传输数据到 LPUART_TDR 寄存器（该操作清零 TXE 位）。该项操作可能重复多次。
- 写最后一个数据到 LPUART_TDR 后，软件等待 TC=1，表明最后一帧发送完成。
- 硬件产生发送时序：
  - 产生起始位，开始传送
  - 传送移位寄存器输出数据（默认 LSB）到 TX 引脚
  - 停止位（根据配置发送个数）
- 传送完最后 1 帧数据后，产生 TC=1（TXE=1 时）。当 TCIE=1，产生 TC 中断。

33.2.4.2. TC/TXE 时序

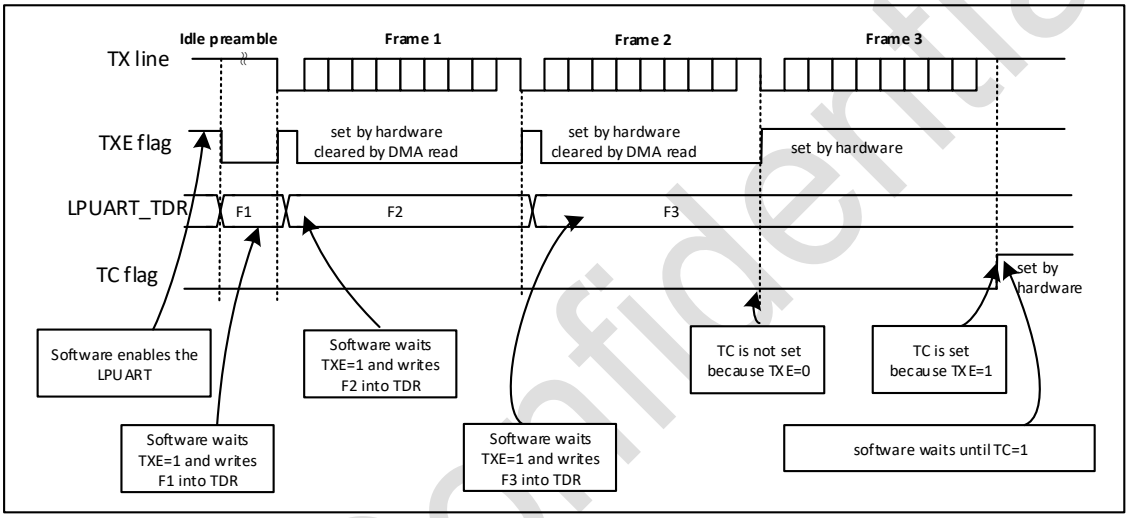


图 33-3 发送时的 TC/TXE 行为

33.2.4.3. 特殊帧

断开帧

设置 SBKRQ=1 将会在当前传输完成后传送一个断开帧。断开帧的长度取决于{M1,M0}寄存器的配置。断开帧发送完毕后，硬件会清零 SBKRQ。

空闲帧

设置 TE=1，在第一个数据帧之前先发送一个空闲帧。

33.2.5. LPUART 接收

LPUART 可接收 7 位、 8 位或 9 位的数据字，具体取决于 LPUART_CR1 寄存器中的{M1,M0}位。

33.2.5.1. 起始位检测

在 RX 上检测到下降沿后，硬件开始检测起始位。在该位的中间检测到 0，则判定该位为起始位。如果检测到的是 1，则置位噪声错误标志（NE），当前起始位无效，重新检测起始位。

33.2.5.2. 数据字符接收

接收步骤

字符接收配置步骤：

- 配置 LPUART_CR1.{M1，M0}，选择接收长度；

- 配置 LPUART_BRR 寄存器，选择波特率；
- 配置 USAR_CR2.STOP，选择 1 位或者 2 位停止位；
- 配置 LPUART_CR1.UE=1，使能 LPUART；
- 多处理器传输时，配置 LPUART_CR3.DMAR=1，使能 DMA 接收;
- 配置 LPUART_CR1.RE，使能接收，开始检测起始位；

当接收到字符后：

- 置位 RXNE。表明移位寄存器的内容已被转移到 RDR，即数据可以被读出
- 当 RXNEIE=1 时，产生中断；
- 在接收时，如果检测到帧错误、噪声错误或溢出错误，则设置相应错误标志位
- 多处理器通讯：
  - RXNE 在每个字节接收后被置位，并由 DMA 对接收数据寄存器的读操作而清零
- 在单缓冲器模式：
  - 软件读 LPUART_RDR 寄存器对 RXNE 位清零。RXNE 位必须在下一字符接收结束前被清零，以免溢出错误

33.2.5.3. 特殊帧及错误处理

-断开字符

当接收到一个断开帧时，LPUART 按错误帧处理。

-空闲字符

当检测到空闲帧时，按正常帧处理，如果 IDLEIE=1，则产生中断。

-上溢错误

如果 RXNE 没有被复位，又接收到一个字符，则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。每接收到 1 个字节后置位 RXNE 标志。如果下一个数据已收到或先前 DMA 请求还没执行时，RXNE 标志仍为置起状态，溢出错误产生。

溢出错误产生时，执行如下错误处理：

- 置位 ORE；
- RDR 内容不会丢失。读 LPUART_RDR 寄存器仍可正常读出先前数据；
- 移位寄存器内容被覆盖，随后接收到的数据丢失
- 如果 RXNEIE=1，或者 EIE=1，则产生中断
- 写 ORECF=1 清零 ORE 位

ORE 置位，表明至少 1 个数据已经丢失。有如下两种情况：

- 如果 RXNE = 1，则最后一个有效数据存储于接收寄存器（RDR）中并且可进行读取
- 如果 RXNE=0，则表示最后一个有效数据已被读取，因此 RDR 中没有要读取的数据。会发生这种情况，已读取 RDR 寄存器中的最后一个有效数据的同时接收到新（并且丢失）的数据。

33.2.5.4. 时钟源选择

LPUART 的时钟源由 RCC_CCIPR.LPUARTxSEL 寄存器配置。在配置 LPUART_CR1.UE=1 前，必须配置时钟源。

时钟源的选择决定了通讯速度范围。

LPUART 双时钟域，一个为内核时钟（lpuart_ker_ck），一个为系统时钟（lpuart_pclk）。当用作从低功耗唤醒时，内核时钟源由 RCC_CCIPR.LPUARTxSEL 寄存器配置。

当不支持双时钟域时，内核时钟与系统时钟相同。

当 MCU 进入低功耗模式，内核时钟源选择低频时，LPUART 可以接收数据并唤醒 MCU，然后软件或者 DMA 可以读取 LPUART_RDR 中数据。

33.2.5.5. 帧错误

如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。

检测到帧错误时做如下处理：

- 硬件设置 FE 位；
- 无效数据从移位寄存器传送到 LPUART_RDR 寄存器；
- 在单字节通讯情况下，不产生中断。但因为 NE 标志位和 RXNE 标志位时同时被设置，FE 位出现上升沿。在多处理器通讯情况下，如果 LPUART_CR3.EIE=1，将产生一个中断。

写 LPUART_ICR.FECF=1 可以清零 FE 标志。

33.2.5.6. 配置停止位个数

可配置位 1/2 个停止位：

- 1 停止位：在第 8/9/10 采样点采样停止位；
- 2 停止位：在第 1 个停止位采样后采样第 2 个停止位。在检测到第 2 个停止位结束后设置 RXNE 和 FE 标志。发生帧错误时不检测第 1 个停止位。

33.2.6. LPUART 波特率产生

发送和接收配置相同波特率。

$$\text{Tx/Rx 波特率} = \frac{256 \ X \ lpuart\_ker\_ck\_pres}{LPUARTDIV}$$

LPUARTDIV 在 LPUART_BRR 寄存器配置。在通讯进行时不要改变波特率的值。LPUART_BRR 寄存器值不能低于 0x300.

fCK 的频率范围为 3 ~ 4096 倍*波特率。

LPUART 时钟源为 LSE 时，最大波特率为 9600 波特率。当 LPUART 由不同于 LSE 时钟的时钟源进行时钟时，可以达到更高的波特率。例如，LPUART 时钟源频率为 100 MHz，可达到的最大波特率约为 33 M 波特。

表 33-1 lpuart_ker_ck_pres = 32,768 kHz 时编程波特率的误差计算

波特率	lpuart_ker_ck_pres = 32,768 kHz		
期望值	实际值	在波特率寄存器中编程的值	误差%
0.3 KBps	0.3 Kbps	0x6D3A	0
0.6 KBps	0.6 Kbps	0x369D	0
1200 Bps	1200.087 bps	0x1B4E	0.007
2400 Bps	2400.17 bps	0xDA7	0.007
4800 Bps	4801.72 bps	0x6D3	0.035
9600 Bps	9608.94 bps	0x369	0.035

表 33-2 lpuart_ker_ck_pres = 100MHz 时编程波特率的误差计算

波特率	lpuart_ker_ck_pres = 100 MHz		
期望值	实际值	在波特率寄存器中编程的值	误差%
38400 波特	38400.04 波特	A2C2A	0.0001
57600 波特	38400.04 波特	6C81C	0.0001
115200 波特	115200.12 波特	3640E	0.0001
230400 波特	230400.23 波特	1B207	0.0001
460800 波特	460804.61 波特	D903	0.001

921600 波特	921625.81 波特	6C81	0.0028
4000000 波特	4000000.00 波特	1900	0
10000000 波特	10000000.00 波特	A00	0
20000000 波特	20000000.00 波特	500	0
33000000 波特	33032258.06 波特	307	0.1

33.2.7. LPUART 接收容忍度

只有当整体的时钟系统地变化小于 LPUART 异步接收器能够容忍的范围，LPUART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的变化(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足： DTRA + DQUANT + DREC + DTCL+DWU < LPUART 接收器的容忍度

其中 DWU 是使用低功耗模式唤醒时由于采样点偏差引起的误差。

LPUART 接收器可以在指定的最大容忍偏差下正确接收数据（理论值）

表 33-3 LPUART 接收容忍度

{M1,M0}位	768 < BRR < 1024	1024 < BRR < 2048	2048 < BRR < 4096	4096 ≤ BRR
8 位({M1,M0} =00), 1 停止位	2%	2.8%	3.90%	4.5%
9 位({M1,M0} =01), 1 停止位	2%	2.3%	3.53%	4.4%
7 位({M1,M0} =10), 1 停止位	2.4%	2.9%	4.7%	4.4%
8 位({M1,M0} =00), 2 停止位	1.8%	2.9%	3.8%	4.3%
9 位({M1,M0} =01), 2 停止位	2%	2.4%	3.4%	3.8%
7 位({M1,M0} =10), 2 停止位	2.4%	2.9%	4%	4.4%

注：当接收到的帧中包含恰好是 10/11/9 位的空闲帧（对应 M= ‘b00/11/10’）时，表中指定的数据在特殊情况下可能会略有不同。

33.2.8. LPUART 多处理器通信

通过 LPUART 可以实现多处理器通讯（将几个 LPUART 连在一个网络里）。例如某个 LPUART 设备可以是主，其 TX 输出和其他 LPUART 从机的 RX 输入相连接；LPUART 从机各自的 TX 输出逻辑与在一起，并且和主机的 RX 输入相连接。

未被寻址的设备可以配置 MME 寄存器置于静默模式。在静默模式：

- 不会设置任何接收状态位；
- 禁止所有接收中断；
- LPUART_ISR.RWU=1.通过配置 LPUART_RQR.MMRQ，硬件和软件可以控制 LPUART 进入静默模式；

根据 LPUART_CR1.WAKE 的值，LPUART 可以用两种方式进入或退出静默模式：

- WAKE=0：空闲总线检测；
- WAKE=1：地址标志检测；

33.2.8.1. 空闲总线检测

当 MMRQ 写 1，RMU 自动置 1 后，LPUART 进入静默模式。

当检测到空闲帧时，LPUART 被唤醒。然后 RWU 被硬件清零，但是 LPUART_ISR.IDLE 并不置起。

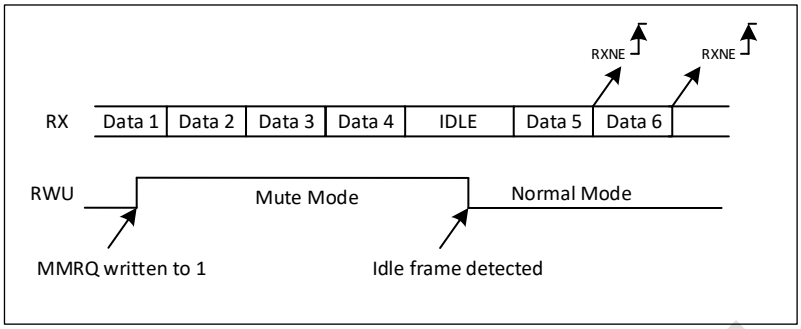


图 33-4 用总线帧检测的静默模式

注：

如果在 IDLE 字符已经过去时将 MMRQ 位置 1，将不会进入静默模式（RWU 未置 1）。

如果在线路处于空闲状态时激活 LPUART，在一个 IDLE 帧持续时间后（不只在接收一个字符帧后）会检测到空闲状态。

33.2.8.2. 4 位/7 位地址标志检测

在这个模式，如果 MSB=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，目标接收器的地址被放在 4 个或者 7 个 LSB 中（由 ADDM7 寄存器选择几个）。这个 4 位/7 位地址被接收器同 LPUART_CR2 寄存器中 ADD 中配置的地址比较。

如果接收到的字节与配置地址不匹配，LPUART 进入静默模式。此时，硬件设置 RWU。接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 LPUART 已经在静默模式。

当 MMRQ 配置为 1 时，LPUART 也会进入静默模式，并设置 RWU=1.

如果接收到的字节与配置地址匹配，LPUART 退出静默模式。此时，清零 RWU，随后正常接收数据。因为 RWU 已被清零，此种情况会设置 RXNE 位。

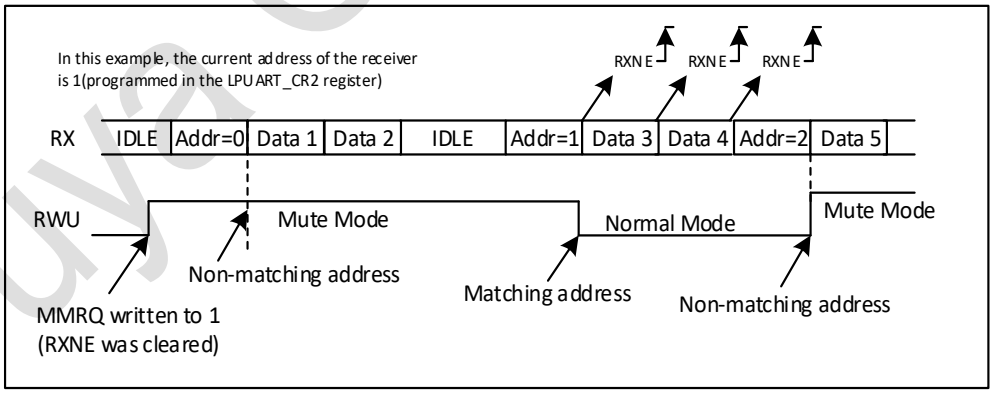


图 33-5 用地址标志检测的静默模式

33.2.9. LPUART 奇偶校验

设置 LPUART_CR1.PCE=1，可以使能奇偶控制（发送时生成一个奇偶位，接收时进行奇偶校验）。根据{M1,M0}位定义的帧长度，支持的 LPUART 帧格式如下表所示：

表 33-4 支持的 LPUART 帧格式

{M1,M0}位	PCE 位	LPUART 帧
00	0	| 开始位 | 8 位数据 | 停止位 |

00	1	|开始位| 7 位数据| 奇偶校验位 |停止位|
01	0	|开始位| 9 位数据|停止位|
01	1	|开始位| 8 位数据|奇偶校验位|停止位|
10	0	|开始位| 7 位数据|停止位|
10	1	|开始位| 6 位数据|奇偶校验位|停止位|

33.2.9.1. 偶校验

偶校验位使得一帧中的 6 或 7 或 8 个 LSB 数据以及校验位中“1”的个数为偶数。

33.2.9.2. 奇校验

奇校验位使得一帧中的 6 或 7 或 8 个 LSB 数据以及校验位中“1”的个数为奇数。

33.2.9.3. 接收校验

奇偶校验错误时设置 PE=1，当 PEIE=1 时产生中断。软件写 PECF=1 清零 PE。

33.2.9.4. 发送校验位

发送时 MSB 由奇偶校验位代替。

33.2.10. LPUART 单线半双工通信

当配置 LPUART_CR3.HDSEL=1 时，LPUART 进入单线半双工通讯模式。

在此模式下：

- TX 和 RX 在模块内部连接；
- RX 引脚不用；
- 当无数据传输时释放 TX 引脚作为标准 I/O

33.2.11. DMA 连续通信

LPUART 可以利用 DMA 连续通讯。接收缓冲器和发送缓冲器的 DMA 请求分别产生。

33.2.11.1. 利用 DMA 发送

通过配置 LPUART_CR3.DMAT 使能利用 DMA 发送。当 TXE=1，DMA 就从指定的 SRAM 区传送数据到 LPUART_DR 寄存器。

DMA 发送的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 LPUART 使用的 DMA 通道
- 配置 LPUART_TDR 寄存器地址作为 DMA 传输的目标地址。在每个 TXE 事件后，数据将被传送到这个地址；
- 配置存储器（如 SRAM）地址作为 DMA 传输的源地址。在每次 TXE 事件后，将从此存储器读数据加载到 LPUART_TDR 寄存器；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 根据应用，配置在传输完成一半还是全部完成时产生 DMA 中断；
- 写 TCCF=1 清零 LPUART_ISR.TC 标志；
- 使能 DMA 通道；

当传输达到配置的传输字节数，DMA 控制器产生中断请求。

当 DMA 写完所有要传输数据后，设置 TCIF=1.然后软件需要等待 TC=1，表明 LPUART 已经发送完最后一帧数据。

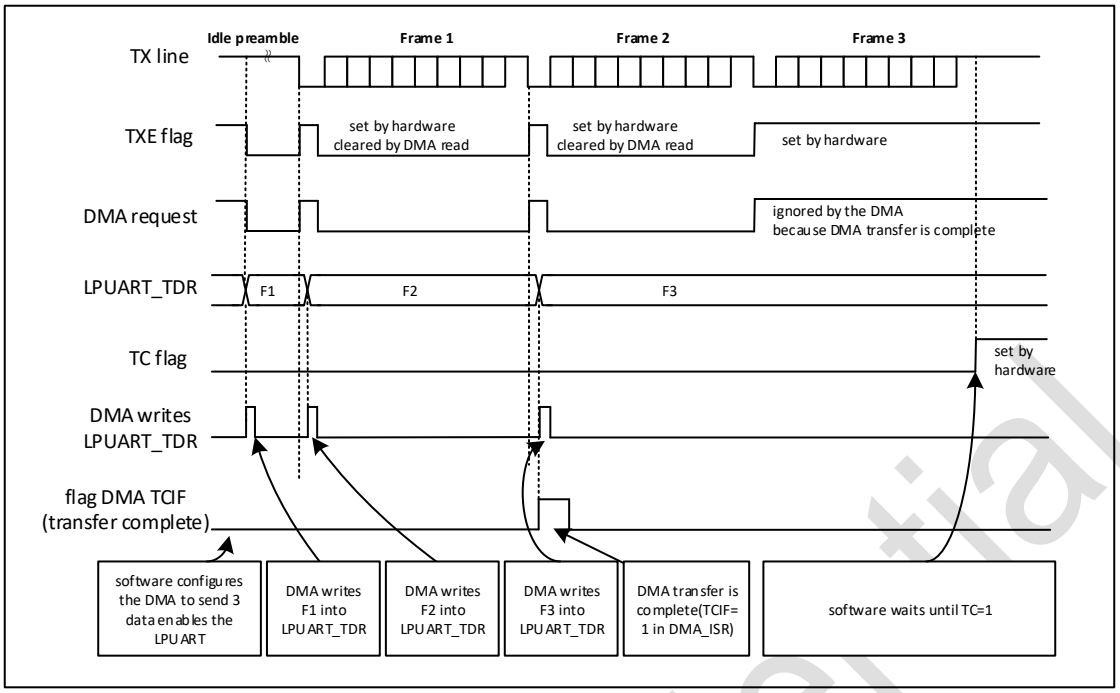


图 33-6 LPUART 用 DMA 发送

33.2.11.2. 利用 DMA 接收

通过配置 LPUART_CR3.DMAR 使能利用 DMA 接收。每收到一个字节，DMA 就把数据从 LPUART_RDR 寄存器传送到指定的 SRAM 区。

DMA 接收的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 LPUART 使用的 DMA 通道
- 配置 LPUART_RDR 寄存器地址作为 DMA 传输的源地址。在每个 RXNE 事件后，数据将从此 LPUART_RDR 寄存器地址读出数据传输到存储器；
- 配置 SRAM 存储器地址作为 DMA 传输的目标地址。在每次 RXNE 事件后，数据将从 LPUART_RDR 寄存器传送到此地址；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 根据应用，配置在传输完成一半还是全部完成时产生 DMA 中断；
- 使能 DMA 通道；

当传输达到配置的传输字节数，DMA 控制器产生中断请求。

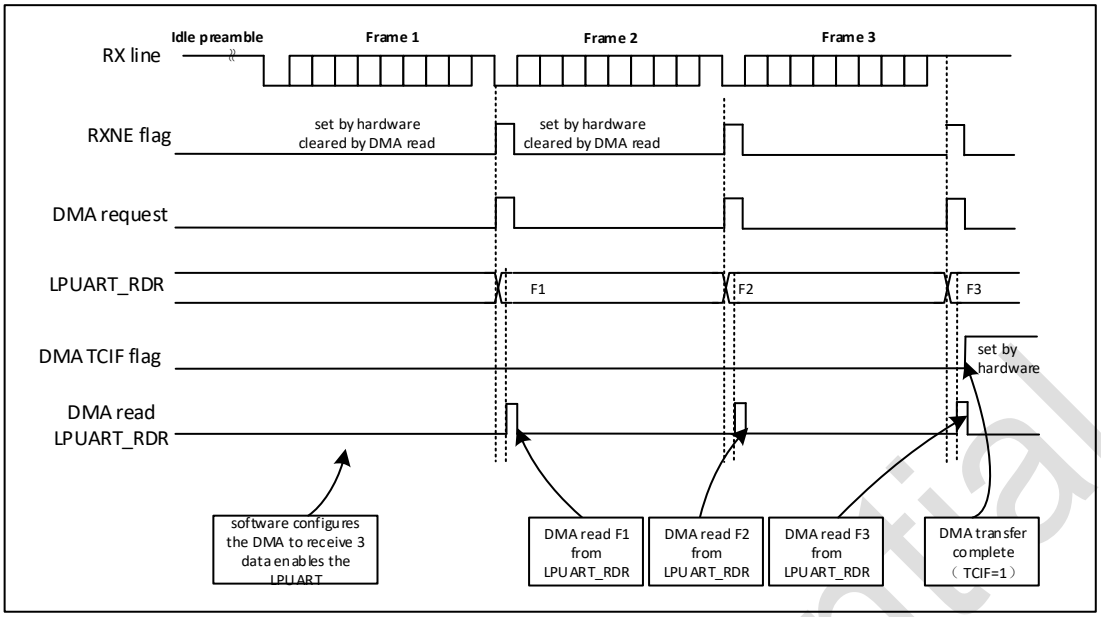


图 33-7 LPUART 用 DMA 接收

33.2.11.3. 错误标志和中断

在多处理器通讯模式，在传输时产生任何错误都会在当前字节传送完成后产生错误标志，如果对应中断使能，则产生中断。

33.2.12. RS232 硬件流控制和 RS485 驱动器使能

通过 CTS 和 RTS 可以控制两个 LPUART 模块的串行数据流。

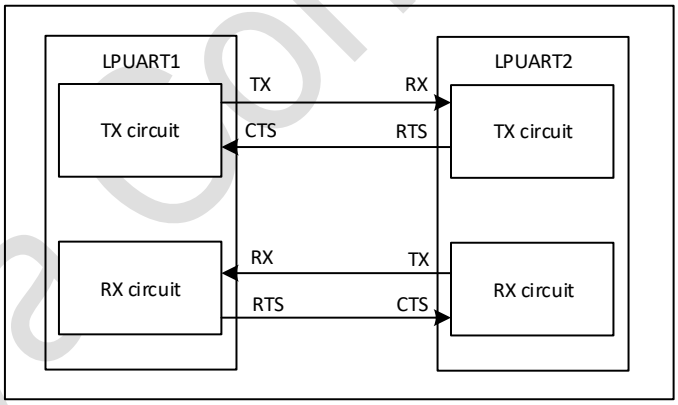


图 33-8 硬件流控制

33.2.12.1. RS232 RTS 流控制

通过配置 LPUART_CR3.RTSE=1，使能 RTS 流控制。当 LPUART 准备好接收数据时，将 RTS 拉低。当接收寄存器满时，RTS 拉高（无效），表明当前帧是最后一帧。

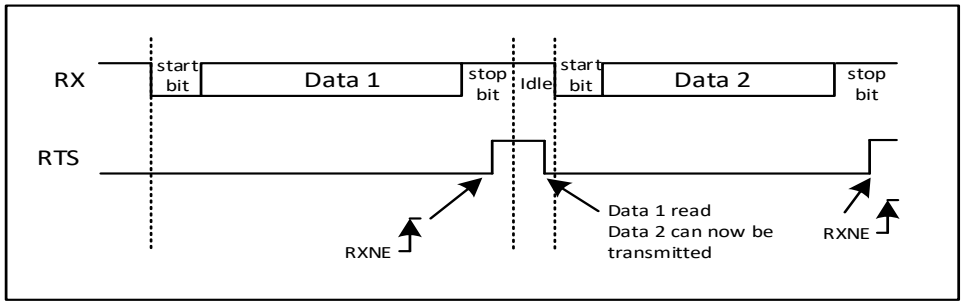


图 33-9 RTS 流控制

33.2.12.2. RS232 CTS 流控制

通过配置 LPUART_CR3.CTSE=1，使能 CTS 流控制。使能后，在传送下一帧前检测 CTS 信号。如果 CTS=0（有效），传送下一笔数据（假设有数据需要传输）。如果 CTS=1（无效），则当前传输完成（在 stop bit 位前完成）。

当 CTSE=1，只要 CTS 信号反转，硬件就置位 CTS。表明接收器准备好通讯或未准备好通讯。当 CTSIE=1 时，产生中断。

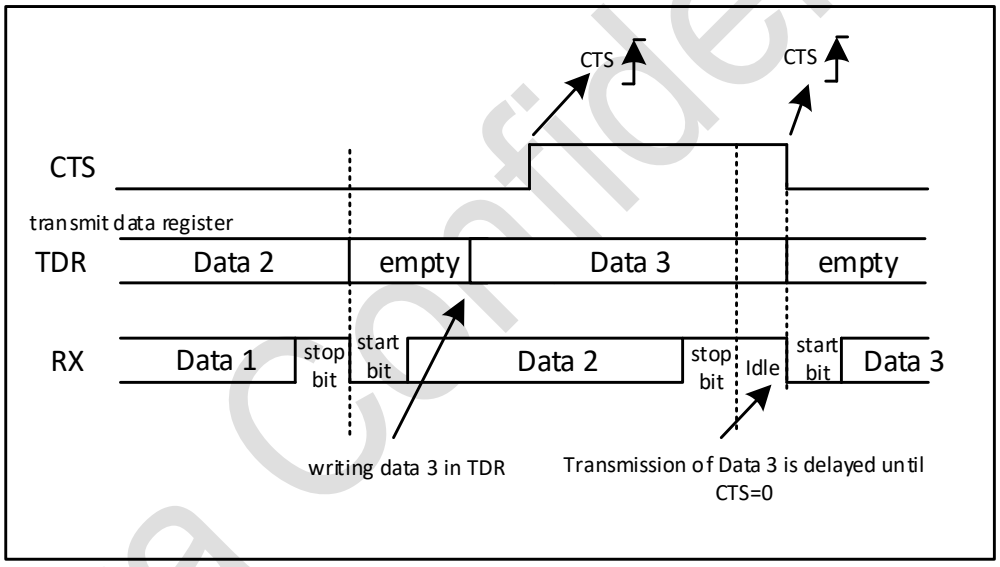


图 33-10 CTS 流控制

注：CTS 在当前帧结束前至少维持 3 个 LPUART 源时钟的高电平。CTSCF 也需要维持至少 2xPCLK 周期。

33.2.12.3. RS485 驱动使能

配置 LPUART_CR3.DEM=1，使能 DE 功能。通过 DE 信号，用户可以激活外部传输控制。DE 信号的有效极性通过 LPUART_CR3.DEP 寄存器配置。

有效时间由 LPUART_CR1.DEAT 配置，为 DE 信号有效沿与起始位开始之间的时间。

释放时间由 LPUART_CR1.DEDT 配置，为最后一个停止位与 DE 无效之间的时间。

具体时间计算如下所示，其中 P=BRR[20:11]:

- DE 插入时间：
  - （1+（DEAT x P））x f_{CK}，P≠0
  - （1+DEAT）x f_{CK}，P=0
- DE 释放时间：

— （1+（DEDT x P））x f_{CK}，P≠0

— （1+DEDT）x f_{CK}，P=0

33.2.13. LPUART 低功耗模式

LPUART 支持低功耗模式，使得在 PCLK 不存在时仍能传输数据。

当 UESM=1，LPUART 能在低功耗模式下唤醒 MCU。

33.2.13.1. 根据 WUS 产生中断唤醒请求

当检测到唤醒事件，硬件产生 WUF 标志。如果 WUFIE=1 时，产生 WUF 中断。在此情况下， WUF 标志置 1 便足以将 MCU 从低功耗模式唤醒。

注：

- 进入低功耗模式前，保证没有 LPUART 传输；
- 不管 MCU 是在低功耗模式还是工作模式，只要检测到唤醒事件，就会置位 WUF 标志；
- 当初始化和使能 LPUART 接收后，必须要通过 REACK 位确认 LPUART 是否真正使能，然后再配置进入低功耗模式；
- 当使能 DMA 接收，在进入低功耗模式前必须关闭 DMA 接收；然后从低功耗唤醒后再重新使能 DMA；

33.2.13.2. 从静默模式进入低功耗模式

如果 LPUART 在进入低功耗模式前处于静默模式：

- 不能用空闲帧唤醒静默模式，因为在低功耗模式下不支持空闲帧检测；
- 用地址匹配的方式唤醒静默模式，则从低功耗模式唤醒也要用地址匹配的方法。当进入低功耗时，如果 RXNE=1，则 LPUART 将不会进入低功耗模式，而维持在静默模式；

33.2.13.3. 低功耗模式时内核时钟关闭

如果低功耗模式下，内核时钟也关闭，则 LPUART 在 RX 线上检测到下降沿时，会产生信号去使能内核时钟。

如果唤醒事件 LPUART_CR3.WUS 配置为地址匹配方式：

- 内核时钟使能后，接收完一帧后如果地址匹配，则 MCU 唤醒，开始正常数据接收；

下图为唤醒事件后地址验证匹配的情况：

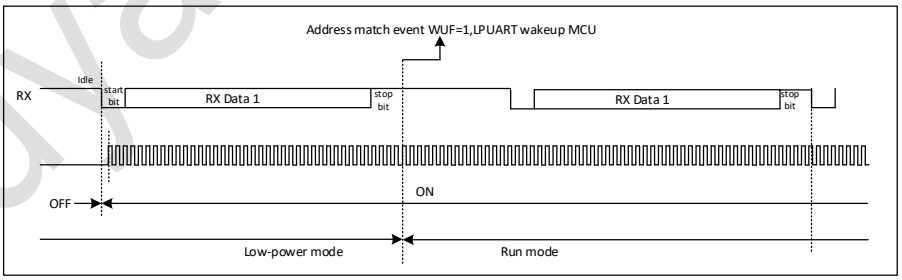


图 33-11 LPUART 地址匹配唤醒

- 如果唤醒事件后地址验证不匹配，则内核时钟再次关闭，MCU 不会被唤醒，系统仍在低功耗模式。

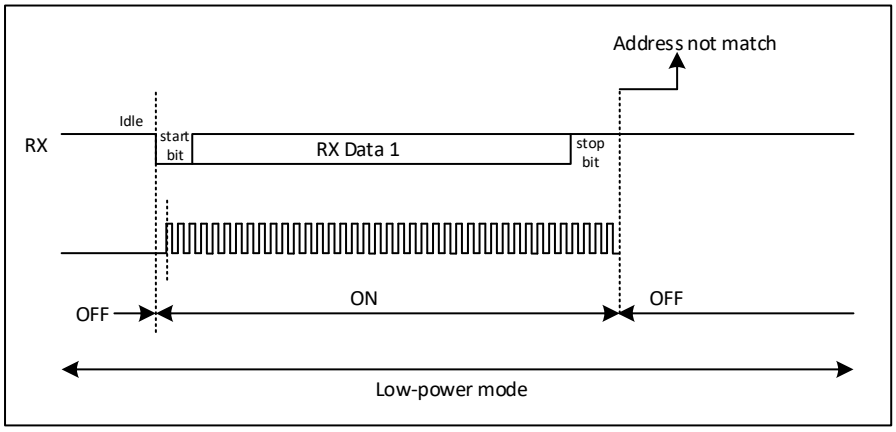


图 33-12 LPUART 地址不匹配无法唤醒

注： 将地址匹配或任一接收的帧用作唤醒事件时，如上两图适用。如果唤醒事件为起始位检测，则 LPUART 会在起始位结束时向 MCU 发送唤醒事件。

33.3. LPUART 中断

表 33-5 LPUART 中断

中断事件	事件标志	使能位	中断清零	中断信号是否有效	
				中断	唤醒
发送数据寄存器空	TXE	TXEIE	写数据到 TDR	YES	NO
CTS 中断	CTSIF	CTSIE	软件配置 CTSCF=1	YES	NO
传送完成	TC	TCIE	写数据到 TDR 或 TCCF=1	YES	NO
接收寄存器未空（读数据准备好）	RXNE	RXNEIE	读 RDR 寄存器	YES	YES
溢出错误	ORE	RXNEIE	ORECF=1	YES	NO
空闲帧	IDLE	IDLEIE	IDLECF=1	YES	NO
奇偶校验错误	PE	PEIE	PECF=1	YES	NO
多处理器通讯时，噪声、溢出和帧错误	NE/ORE/FE	EIE	NECF=1 清零 NE；ORECF=1 清零 ORE；FECF=1 清零 FE；	YES	NO
地址字节匹配	CMF	CMIE	CMCF=1	YES	NO
从低功耗模式唤醒	WUF	WUFIE	WUCF=1	YES	YES

33.4. LPUART 寄存器

33.4.1. LPUART 控制寄存器 1（LPUART_CR1）

偏移地址：0x00

复位值：0x0000_0000

如果要配置该寄存器的 DEAT 和 DEDT 位，为了提高配置效率，需要一次配置 LPUART_CR1 中除 UE 外的其他所有寄存器。如果要多次配置该寄存器，每两次配置之间需要间隔 5 个 LPUART 内核时钟周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]				
			RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	M1	RW	0	{M1,M0}的值配置长度，由软件置位或者清零。 2’b00: 1 起始位，8 数据位，n 停止位； 2’b01: 1 起始位，9 数据位，n 停止位； 2’b10: 1 起始位，7 数据位，n 停止位； 只能在 LPUART 未使能时（UE=0）才能写入该位。
27:26	Reserved	-	-	保留
25:21	DEAT[4:0]	RW	5’h0	该寄存器配置驱动使能（DE）信号有效和起始位之间的时间。用采样时间作为单位。 如果不支持 DE 功能，则该位保留。 只能在 LPUART 未使能时（UE=0）才能写入该位
20:16	DEDT[4:0]	RW	5’h0	该寄存器配置发送帧停止位与 DE 信号无效之间的时间。用采样时间作为单位（1/8 或者 1/16 位周期，取决于过采样率）。如果 DEDT 时间内有写 LPUART_TDR 寄存器请求发送，则数据只有当 DEDT 和 DEAT 都结束后才发送。 如果不支持 DE 功能，则该位保留。 只能在 LPUART 未使能时（UE=0）才能写入该位
15	Reserved	-	-	保留
14	CMIE	RW	0	字符匹配中断使能。该位由软件置位或者清零。 0：中断禁止； 1：CMF 中断使能；
13	MME	RW	0	静音模式使能。 0：接收器工作在工作模式； 1：接收器在工作模式和静默模式之间切换；
12	M0	RW	0	与 M1 结合配置字长度。 只能在 LPUART 未使能时（UE=0）才能写入该位
11	WAKE	RW	0	接收唤醒方式。 从静默模式唤醒方式。由软件置位或者清零。 0：空闲帧唤醒； 1：地址标志唤醒； 只能在 LPUART 未使能时（UE=0）才能写入该位
10	PCE	RW	0	奇偶校验控制。 0：奇偶校验禁止； 1：奇偶校验使能； 奇偶校验位：9bit 数据的第 9 位；8bit 数据的第 8 位；7bit 数据的第 7 位。 只能在 LPUART 未使能时（UE=0）才能写入该位

9	PS	RW	0	奇偶校验选择。由软件置位和清零。 0：偶校验； 1：奇校验； 只能在 LPUART 未使能时（UE=0）才能写入该位
8	PEIE	RW	0	PE 中断使能。由软件置位和清零。 0：禁止； 1：PE 中断使能；
7	TXEIE	RW	0	传输寄存器空中断使能。由软件置位和清零。 0：禁止； 1：TXE 中断使能；
6	TCIE	RW	0	传送结束中断使能。由软件置位和清零。 0：禁止； 1：TC 中断使能；
5	RXNEIE	RW	0	接收寄存器非空中断使能；由软件置位和清零。 0：禁止； 1：ORE 或者 RXNE 中断使能；
4	IDLEIE	RW	0	IDLE 中断使能。由软件置位和清零。 0：禁止； 1：IDLE 中断使能；
3	TE	RW	0	传送使能。 0：传送禁止； 1：传送使能；
2	RE	RW	0	接收使能。 0：接收禁止； 1：接收使能，开始检测 start 位；
1	UESM	RW	0	停止低功耗模式下 LPUART 唤醒使能。 该寄存器为 1，如果 LPUART 时钟选择 LSI 或 LSE，LPUART 能唤醒 MCU。 0：LPUART 不能唤醒 MCU； 1：LPUART 模式唤醒 MCU。使能后，LPUART 时钟源选择 LSI 或 LSE。 从低功耗模式唤醒后，该位清零。
0	UE	RW	0	LPUART 使能。 当该位清零后，LPUART 模块会立即停止当前操作。LPUART 的配置会保持，但所有状态标志位，LPUART_ISR 寄存器值会变为默认值。该位由软件置位和清零。 0：LPUART 预分频器和输出禁止，低功耗模式； 1：LPUART 使能； 软件需要等待 LPUART_ISR.TC 置位后，才能清零 UE 位，进入低功耗模式； 同时，在清零 UE 位之前 DMA 通道需要禁止。 该位清零时，LPUART 配置寄存器值维持，但 LPUART_ISR 状态寄存器全部复位。

33.4.2. LPUART 控制寄存器 2（LPUART_CR2）

偏移地址：0x04

复位值：0x0000_0000

如果要配置该寄存器的 ADD 位，为了提高配置效率，需要一次配置 LPUART_CR2 中所有寄存器。如果要多次配置该寄存器，每两次配置之间需要间隔 5 个 LPUART 内核时钟周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								Res.	Res.	Res.	TXOE_ALWAYS_ON	MSBFIRST	DATAINV	TXINV	RXINV
RW	RW	RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	Res.	STOP[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDM7	Res.	Res.	Res.	Res.
RW		RW	RW								RW				

Bit	Name	R/W	Reset Value	Function
31:24	ADD[7:0]	RW	8’b0	LPUART 地址。 该寄存器用于多处理器静默或者停止模式，用作地址唤醒时的地址。传送器发送的 MSB 位为 1。 在正常接收模式时，该寄存器用作字符检测。此时，比对接收内容与 ADD[7:0]相比较，如果匹配，则置位 CMF 标志位。 只有在 RE=0 或者 UE=0 时，该寄存器可以被写。
23:21	Reserved	-	-	保留
20	TXOE_ALWAYS_ON	RW	1	该寄存器用于控制 LPUART 输出使能。 0：硬件自主控制输出使能开/关 1：软件控制输出使能常开 只能在 LPUART 未使能时（UE=0）才能写入该位 注：在半双工模式，需要将此位关闭。
19	MSBFIRST	RW	0	最高有效位在前。 0：起始位后，收发第 0 位数据； 1：起始位后，收发第 7/8/9 位数据； 只能在 LPUART 未使能时（UE=0）才能写入该位
18	DATAINV	RW	0	二进制数据逆处理。 0：以正/正向逻辑收发数据寄存器数据。（1=H，0=L） 1：以反/反向逻辑收发数据寄存器数据。奇偶位也会逆处理。（1=L，0=H） 只能在 LPUART 未使能时（UE=0）才能写入该位
17	TXINV	RW	0	TX 引脚有效电平反相。 0：TX 引脚为标准逻辑电平（V_{CC} =1/空闲, Gnd=0/标记）； 1：TX 引脚信号值取反（V_{CC} =0/标记, Gnd=1/空闲)。 只能在 LPUART 未使能时（UE=0）才能写入该位
16	RXINV	RW	0	RX 引脚有效电平反相。 0：RX 引脚为标准逻辑电平（V_{CC} =1/空闲, Gnd=0/标记）； 1：RX 引脚信号值取反（V_{CC} =0/标记, Gnd=1/空闲)。 只能在 LPUART 未使能时（UE=0）才能写入该位
15	SWAP	RW	0	TX/RX 引脚互换。 0：TX/RX 引脚按照标准引脚排列定义； 1：TX/RX 引脚互换。此时用作交叉连接其他 LPUART 时。

				只能在 LPUART 未使能时（UE=0）才能写入该位
14	Reserved	-	-	保留
13:12	STOP[1:0]	RW	2'b0	停止位配置。 00：1 停止位； 01，11：保留； 10：2 停止位； 只能在 LPUART 未使能时（UE=0）才能写入该位
11:5	Reserved	-	-	保留
4	ADDM7	RW	0	7 位/4 位地址检测配置。 0：4 位地址检测； 1：7 位地址检测（8 位数据模式）； 只能在 LPUART 未使能时（UE=0）才能写入该位
3:0	Reserved	-	-	保留

33.4.3. LPUART 控制寄存器 3（LPUART_CR3）

偏移地址：0x08

复位值：0x0000_0000

如果要配置该寄存器的 WUS 位，为了提高配置效率，需要一次配置 LPUART_CR3 中所有寄存器。如果要多次配置该寄存器，每两次配置之间需要间隔 5 个 LPUART 内核时钟周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUFIE	WUS[1:0]		Res.	Res.	Res.	Res.
									RW	RW	RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
RW	RW	RW	RW		RW	RW	RW	RW	RW			RW			RW

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22	WUFIE	RW	0	低功耗唤醒中断使能。 0：中断禁止 1：WUF 中断使能
21:20	WUS[1:0]	RW	0	低功耗唤醒选择。 00：地址匹配产生 WUF 01：保留 10：起始位检测产生 WUF 11：RXNE 产生 WUF 只能在 LPUART 未使能时（UE=0）才能写入该寄存器。
19:16	Reserved	-	-	保留
15	DEP	RW	0	DE 极性选择。 0：DE 高电平有效 1：DE 低电平有效 只能在 LPUART 未使能时（UE=0）才能写入该位
14	DEM	RW	0	DE 模式使能。 0：禁止 1：DE 模式使能，DE 信号在 RTS 引脚输出。 用户通过使能该位来控制外部传输。 只能在 LPUART 未使能时（UE=0）才能写入该位
13	DDRE	RW	0	接收错误时是否禁止 DMA。

				0：接收错误时不禁止 DMA。相应出错标志置位，但 RXNE=0. 不会产生 DMA 请求，所以出错数据不会输出。 1：接收错误时禁止 DMA。RXNE 和出错标志位都会置位。在出错标志位清零前，屏蔽 DMA 请求。 只能在 LPUART 未使能时（UE=0）才能写入该位
12	OVRDIS	RW	0	溢出禁止。 0：当接收到新数据前已收数据没读时置位溢出出错标志（PRE） 1：溢出功能禁止。接收到新数据后，RXNE=1，ORE=0，新数据会覆盖 LPUART_RDR 寄存器中数据。 只能在 LPUART 未使能时（UE=0）才能写入该位
11	Reserved	-	-	保留
10	CTSIE	RW	0	CTS 中断使能。 0：禁止 1：CTSIF 中断使能
9	CTSE	RW	0	CTS 使能。 0：CTS 硬件流控制禁止 1：CTS 模式使能。当有当 CTS 输入为 0 时，才会传输数据。此时，当数据写入数据寄存器后，要等待 CTS 有效后才会启动传输。 只能在 LPUART 未使能时（UE=0）才能写入该位
8	RTSE	RW	0	RTS 使能。 0：RTS 硬件流控制禁止 1：RTS 输出使能，只有当接收缓冲区未满时才会请求下一个数据。当前数据发送完成后，发送操作暂停。如果可以接收数据了，将 RTS 置为有效（0）。 只能在 LPUART 未使能时（UE=0）才能写入该位
7	DMAT	RW	0	传送时使能 DMA。 0：禁止 1：传送时使能 DMA
6	DMAR	RW	0	接收时使能 DMA。 0：禁止 1：接收时使能 DMA
5:4	Reserved	-	-	保留
3	HDSEL	RW	0	半双工选择。 0：非半双工模式 1：半双工模式选择 只能在 LPUART 未使能时（UE=0）才能写入该位
2:1	Reserved	-	-	保留
0	EIE	RW	0	错误中断使能。 0：禁止 1：帧错误 FE、溢出错误 ORE、噪声 NF 中断产生

33.4.4. LPUART 波特率寄存器（LPUART_BRR）

偏移地址：0x0C

复位值：0x0000_0300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BBR[19:16]			
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BBR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	BBR[19:0]	RW	0x300	LPUART 波特率。 禁止在 LPUART_BRR 寄存器中写入小于 0x300 的值。 只能在 LPUART 未使能时（UE=0）才能写入该寄存器。

33.4.5. LPUART 请求寄存器（LPUART_RQR）

偏移地址：0x18

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMRQ	SBKRQ	Res.
													W	W	

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	MMRQ	W	0	静音模式请求。 该位写 1 使 LPUART 进入静默模式，并置位 RWU 标志。
1	SBKRQ	W	0	发送中止请求。 该位写 1 会置位 SBKF，并发送 BREAK 请求。
0	Reserved	-	-	保留

33.4.6. LPUART 中断和状态寄存器（LPUART_ISR）

偏移地址：0x1C

复位值：0x0080_00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
									R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
					R	R		R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22	REACK	R	0	接收使能确认标志。 硬件置位/复位该寄存器。表明硬件在进入低功耗模式前准备好接收。 如果 LPUART 不支持从停止低功耗模式唤醒，则该位保留。
21	TEACK	R	0	发送使能确认标志。 硬件置位/复位该寄存器。

				当在 LPUART_CR1 寄存器中写入 TE=0 生成空闲帧请求，然后写入 TE=1 以满足 TE=0 的最小周期时，可以使用该寄存器。
20	WUF	R	0	从低功耗模式唤醒标志。 当检测到唤醒事件时硬件写该位为 1. 事件通过 WUS 位域定义。通过向 LPUART_ICR 寄存器中的 WUCF 写入 1，此位由软件清零。如果 LPUART_CR3 寄存器中 WUFIE=1，则会生成中断。 注：当 UESM 清零时，WUF 标志也清零。
19	RWU	R	0	接收到静默模式唤醒。 当接收到静默模式序列，该寄存器置位；如果接收到唤醒序列，该寄存器清零。具体哪种唤醒序列（地址标记或者空闲帧）由寄存器 LPUART_CR1.WAKE 位控制。 0：接收器为工作模式； 1：接收器为静默模式；
18	SBKF	R	0	发送中止标志。 该寄存器表明请求发送断开字符。由软件写 1 到 LPUART_RQR.SBKRQ 寄存器置位该寄存器。硬件在发送完中止字符的停止位后清零该寄存器。 0：不发送断开字符； 1：发送断开字符；
17	CMF	R	0	地址匹配标志。 当接收到 ADD[7:0]值匹配的字符时置位该寄存器。软件写 1 到 CMCF 寄存器会清零该位。
16	BUSY	R	0	忙标志。 当 RX 线接收数据时（正确接收到起始位），硬件该寄存器置位。接收结束，硬件清零该寄存器。 0：LPUART 空闲； 1：正在进行接收；
15:11	Reserved	-	-	保留
10	CTS	R	0	CTS 标志。 0：CTS 线为 1； 1：CTS 线为 0；
9	CTSIF	R	0	CTS 中断标志。 0：CTS 状态线未改变； 1：CTS 状态线值改变；
8	Reserved	-	-	保留
7	TXE	R	1	传输寄存器空标志，可以写数据到 LPUART_TDR。 0：数据寄存器满； 1：数据寄存器空；
6	TC	R	1	传送完成标志。 传送数据帧完成后，且 TXE=1，则硬件置位该寄存器。TCIE=1 时产生中断。 软件写 1 到 TCCF 寄存器清零该位。 0：传送未完成； 1：传送完成；

5	RXNE	R	0	读数据寄存器不空标志。 当移位寄存器值传送到 LPUART_RDR 寄存器，硬件置位该寄存器。 软件读 LPUART_RDR 寄存器则清零该位。 当 RXNEIE=1 时，产生中断。 0：未收到数据； 1：接收数据准备好读出；
4	IDLE	R	0	空闲标志。 检测到空闲帧，硬件置位该寄存器。当 IDLEIE=1 时产生中断。 软件写 1 到 IDLECF 清零该位。 0：未检测到 IDLE 帧； 1：检测到 IDLE 帧；
3	ORE	R	0	溢出错误标志。 当 RXNE=1 时，在移位寄存器中接收到的数据正准备转移到 RDR 寄存器时，硬件设置该位。 软件写 1 到 ORECF 寄存器清零该位。 当 RXNEIE=1 或者 EIE=1 时，产生中断。 注：该寄存器置位时，RDR 寄存器内容不会丢失，但移位寄存器内容被覆盖。 当 EIE=1 时，产生 ORE 中断。
2	NE	R	0	START 位噪声标志。 在数据帧接收到噪声时，硬件置位该寄存器。 软件写 NFCF=1 会清零该位。
1	FE	R	0	帧错误标志。 当检测到不同步、过多的噪声或中断字符时，由硬件设置此位。 软件写 FECF=1 清零该位。 当 EIE=1 时，产生中断。
0	PE	R	0	校验值错误。 当接收时校验值错误时，硬件置位该寄存器。 软件写 PECF=1 清零该位。 当 PEIE=1 时，产生中断。

33.4.7. LPUART 中断标志清零寄存器（LPUART_ICR）

偏移地址：0x20

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	M1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
											W			W	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
						W			W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20	WUCF	W	0	低功耗唤醒唤醒标志清零。 软件写 1 会清零 WUF 寄存器。
19:18	Reserved	-	-	保留
17	CMCF	W	0	地址匹配标志清零。

				软件写 1 会清零 CMF 寄存器。
16:10	Reserved	-	-	保留
9	CTSCF	W	0	CTS 标志清零。 软件写 1 清零 CTSIF 寄存器。
8:7	Reserved	-	-	保留
6	TCCF	W	0	传送完成标志清零。 软件写 1 清零 TC 寄存器。
5	Reserved	-	-	保留
4	IDLECF	W	0	空闲标志清零。 软件写 1 清零 IDLE 寄存器。
3	ORECF	W	0	溢出错误标志清零。 软件写 1 清零 ORE 寄存器。
2	NECF	W	0	噪声标志清零。 软件写 1 会清零 NE 寄存器。
1	FECF	W	0	帧错误标志清零。 软件写 1 清零 FE 寄存器。
0	PECF	W	0	校验值错误标志清零。 软件写 1 清零 PE 寄存器。

33.4.8. LPUART 接收数据寄存器（LPUART_RDR）

偏移地址：0x24

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[8:0]								
							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	RDR[8:0]	R	0	接收数据寄存器。

33.4.9. LPUART 发送数据寄存器（LPUART_TDR）

偏移地址：0x28

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	TDR[8:0]	RW	0	发送数据寄存器。 当校验使能时，该寄存器的 bit7 或者 bit8（取决于数据长度）在发送时被校验位代替。 注意：建议先开启 TE 和 UE 之后，再写 TDR 数据。

33.4.10. LPUART 预分频器寄存器（LPUART_PRESC）

偏移地址：0x2C

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	PRESCALER	RW	0	输入时钟预分频寄存器。 0000: 不分频; 0001: 2 分频; 0010: 4 分频; 0011: 6 分频; 0100: 8 分频; 0101: 10 分频; 0110: 12 分频; 0111: 16 分频; 1000: 32 分频; 1001: 64 分频; 1010: 128 分频; 1011: 256 分频; 其它: 256 分频。 只能在 LPUART 未使能时（UE=0）才能写入该寄存器。

34. 串行外设接口/集成电路内置音频总线 (SPI/I²S)

本项目设计实现了 2 个 SPI/I²S 模块，两个模块的功能完全一样，均包含 I²S 及 SPI CRC 功能。

34.1. SPI/I²S 简介

SPI/I2S 接口可以配置为支持 SPI 协议或者支持 I²S 音频协议。SPI 接口默认工作在 SPI 方式，可以通过软件把功能从 SPI 模式切换到 I²S 模式。

串行外设接口(SPI)允许芯片与外部设备以半双工、全双工、单工同步的串行方式通信。此接口可以被配置成主模式，并为外部从机提供通信时钟(SCK)。接口还能以多主配置方式工作。它可用于多种用途，包括使用一条双向数据线的双线单工同步传输，还可使用 CRC 校验的可靠通信。

I2S 也是同步串行接口通讯协议。它支持四种音频标准，包括飞利浦 I²S 标准，MSB 和 LSB 对齐标准，以及 PCM 标准。它在半双工通讯中，可以工作在主和从两种模式下。当它作为主机时，通过接口向外部的从机提供时钟信号。

34.2. SPI/I²S 主要特性

34.2.1. SPI 主要特性

- 支持主机或者从机模式
- 3 线全双工同步传输
- 2 线半双工同步传输（有双向数据线）
- 2 线单工同步传输（无双向数据线）
- 8 位或者 16 位传输帧选择
- 支持多主模式
- 8 个主模式波特率预分频系数（最大为 $f_{PCLK}/2$）
- 从模式频率（最大为 $f_{PCLK}/2$）
- 主模式和从模式下均可以由软件或硬件进行 NSS 管理：主/从操作模式的动态改变
- 可编程的时钟极性和相位
- 可编程的数据顺序，MSB 在前或 LSB 在前
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 支持可靠通信的硬件 CRC
  - 在发送模式下，CRC 值可以被作为最后一个字节发送
  - 在全双工模式中对接收到的最后一个字节自动进行 CRC 校验
  - CRC 错误标志
- 支持 Motorola 和 TI 模式
- 可触发引起中断的主模式故障、过载标志
- 2 个具备 DMA 能力的深度为 4，宽度为 16bit（当数据帧设置为 8bit 时，宽度为 8bit）的嵌入式 Rx 和 Tx FIFOs

34.2.2. I²S 主要特性

- 半双工通信(仅发送或接收)
- 主或者从操作
- 8 位线性可编程预分频器，获得精确的音频采样频率(8 kHz 到 192 kHz)
- 数据格式可以是 16 位，24 位或者 32 位
- 音频信道固定数据包帧为 16 位(16 位数据帧)或 32 位(16、 24 或 32 位数据帧)
- 可编程的时钟极性(稳定态)
- 从发送模式下的下溢标志位、接收模式下的上溢标志位（主或从），以及接收和发送模式下的帧错误标志（仅适用于从机）
- 16 位数据寄存器用来发送和接收，在通道两端各有一个寄存器
- 支持的 I²S 协议：
  - I²S 飞利浦标准
  - MSB 对齐标准(左对齐)
  - LSB 对齐标准(右对齐)
  - PCM 标准(16 位通道帧上带长或短帧同步或者 16 位数据帧扩展为 32 位通道帧)
- 数据方向总是 MSB 在先
- 发送和接收都具有 DMA 能力
- 主时钟可以输出到外部音频设备，比率固定为 256xfs(fs 为音频采样频率)

34.3. SPI 功能描述

34.3.1. SPI 框图

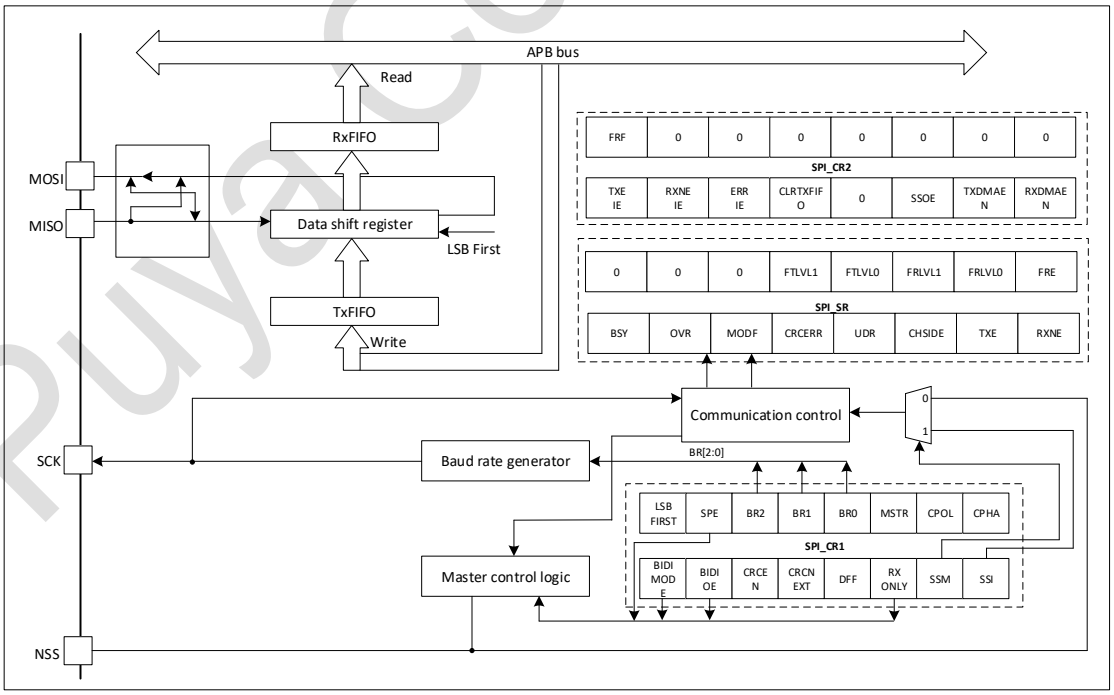


图 34-1 SPI 框图

SPI 通过 4 个引脚与外部器件相连：

MISO：主机输入/从机输出引脚。该引脚在从模式下发送数据，在主模式下接收数据。

MOSI：主机输出/从机输入引脚。该引脚在主模式下发送数据，在从模式下接收数据。

SCK：串口时钟，作为主机的输出，从机的输入。

NSS：从机选择。取决于 SPI 和 NSS 的设定，该引脚可以用作：

- 选择要通讯的从机
- 同步数据帧 或者
- 检测多主机间的冲突

SPI 总线允许在一个主机和一个或者多个从机之间的通讯。总线由至少两根线组成：一个是时钟，另一个是用于同步传输的数据。根据应用场景，可以选择增加另外一根数据线和从机选择信号。

34.3.2. 一个主机和一个从机的通信

针对不同的应用场景，SPI 可以使用几种不同的配置进行通讯。这些配置使用 2 线、3 线（软件 NSS），或者 3 线、4 线（硬件 NSS）。通讯始终由主机启动。

34.3.2.1. 全双工通信

默认情况下，SPI 被配置成全双工通讯。在这种配置下，主机和从机的移位寄存器，在 MOSI 和 MISO 引脚之间，使用两个单向的线连到一起。在 SPI 通讯期间，数据在主机提供的时钟沿同步的被移位。主机通过 MOSI 线发送数据，从 MISO 线接收来自从机的数据。当数据帧传输完成（所有位完成移位），在主机和从机之间即完成信息交换。

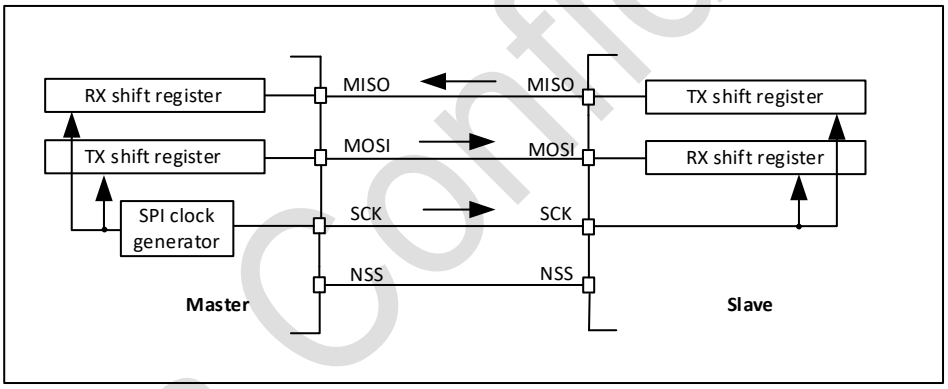


图 34-2 全双工单主/单从机应用

34.3.2.2. 半双工通信

通过设定 BIDIMODE 位（SPI_CR1 寄存器），SPI 可以工作在半双工模式。在这种配置下，用 1 根数据线完成主机和从机移位寄存器的连接。在通讯过程中，在 SCK 的时钟沿，数据在两个移位寄存器之间以 BIDIOE（SPI_CR1 寄存器）选择的方向，同步移位。在该配置下，主机的 MISO 引脚和从机的 MOSI 引脚被释放作为 GPIO 给其他应用使用。

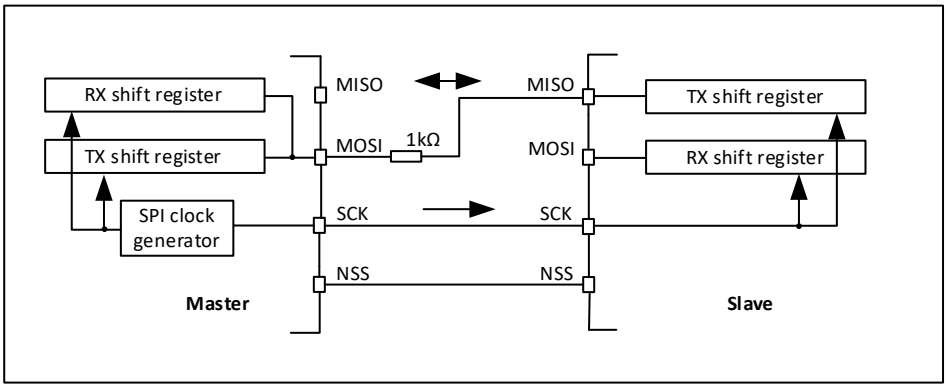


图 34-3 半双工单主/从机应用

（1）NSS 引脚可以被使用在主机和从机之间进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

（2）在该配置下，主机的 MISO 引脚和从机的 MOSI 引脚可以用作 GPIO。

（3）当以双向模式工作的两个节点间的通信方向不是同步变化时，会出现临界情况，新发送器访问共用数据线，而前一个发送器仍保持线路上的相反值（值取决于 SPI 配置和通信数据）。两个节点会出现冲突，在共用线上短暂提供相反的输出电平，直到下一个节点也相应地改变其方向设置。建议此模式下在 MISO 和 MOSI 引脚之间插入串行电阻以在这种情况下保护输出并限制电流在二者之间流过。

34.3.2.3. 单工通信

通过使用 RXONLY（SPI_CR1 寄存器）位将 SPI 设置为只发送模式或只接收模式，可使 SPI 以单工模式进行通信。在这个配置下，在主机和从机的移位寄存器之间只使用 1 根线。另一对 MISO 和 MOSI 引脚不被使用，可以被释放成 GPIO。

- 只发送模式（RXONLY=0）：配置与全双工模式相同。应用忽略在未使用的输入引脚上的信息。这个引脚可以被用作标准的 GPIO。
- 只接收模式（RXONLY=1）：通过置位 RXONLY，应用可以禁止 SPI 输出功能。
  - 在从机模式配置下，MISO 输出被禁止，该引脚被用作 GPIO。在从机选择信号有效时，从机继续从 MOSI 引脚接收数据。接收到的数据事件是否发生取决于数据缓冲区的配置。
  - 在主机模式配置下，MOSI 输出被禁止，引脚可以用作 GPIO。只要 SPI 处于使能状态，便不断生成时钟信号。停止时钟的唯一方式是将 RXONLY 位或 SPE 位清零，直至来自 MISO 引脚的传入模式结束，然后基于相应配置填充数据缓冲区结构。

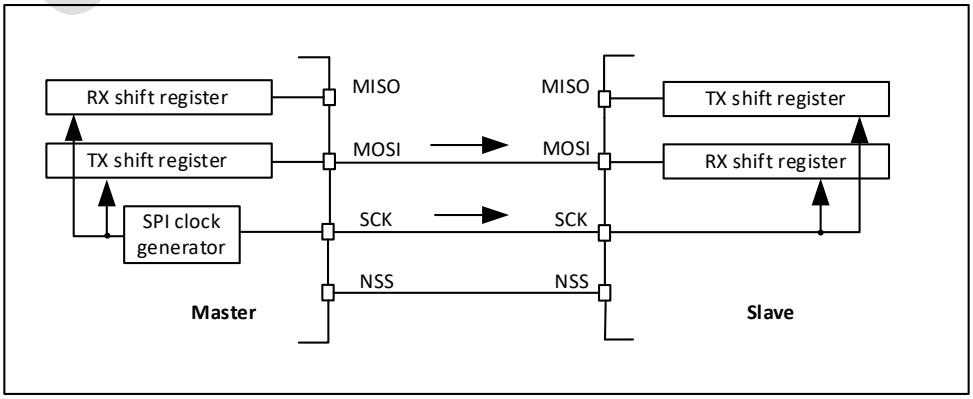


图 34-4 单工单主/从机应用(主机只发送/从机只接收模式)

（1）在主机和从机之间可以使用 NSS 引脚进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

（2）在 Rx 移位寄存器的输入上捕获意外的输入信息。在标准的只发送模式下，所有与传输接收相关的事件都被必须被忽略（如 OVF 标志）。

（3）在该配置下，两边的 MISO 引脚都被用作 GPIO。

注：任何单工通信都可以通过固定半双工模式中的方向设置来实现（使能双向模式，同时 BDIO 位保持不变）。

34.3.3. 标准多从机通信

在一个有两个或者更多个独立从机的配置里，主机使用 GPIO 引脚为每个从机，来管理片选线。主机必须通过拉低与从机 NSS 输入相连的 GPIO 电平来选择某个从机。执行该操作后，便建立了标准主机和专门的从机之间的通讯。

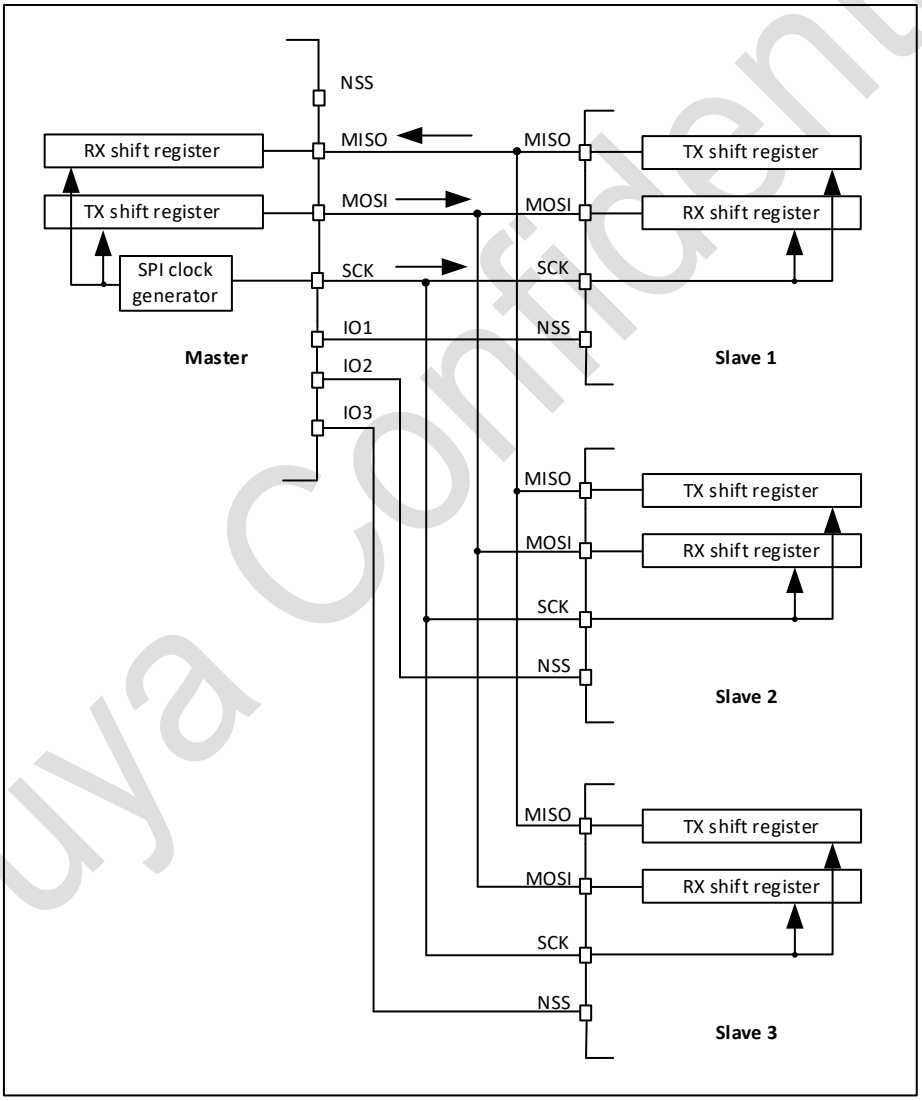


图 34-5 主机选择 3 个从机

在这种配置下不使用 NSS 引脚。必须通过 SSM=1, SSI=1 来防止任何 MODF 错误。

由于从机的 MISO 引脚连接到一起，所有从机 MISO 引脚的 GPIO 必须配置为开漏复用功能。

34.3.4. 多主机通信

如果 SPI 总线未用于多主机功能，用户可使用内置功能来检测试图同时控制总线的两个节点间是否存在潜在冲突。对于该检测，NSS 引脚配置为硬件输入模式。

由于此时只有一个结点可将其输出施加到公用数据线上，因此该模式连接的 SPI 节点不能超过两个。

当节点无效时，默认情况下均保持从机模式。一旦一个节点要接管对总线的控制，它会将自身切换到主机模式，然后通过专用 GPIO 引脚向其他节点的从机选择输入施加有效电平。会话完成后，有效的从机选择信号将被释放，控制总线的节点会短暂切换回被动从模式，等待下一个会话开始。

如果两个节点同时发出各自的控制请求，则会出现总线冲突（模式故障 MODF 事件）。随后，用户可应用某个简单的仲裁过程（例如，在两个节点上施加不同的预定义超时来推迟下一次尝试）。

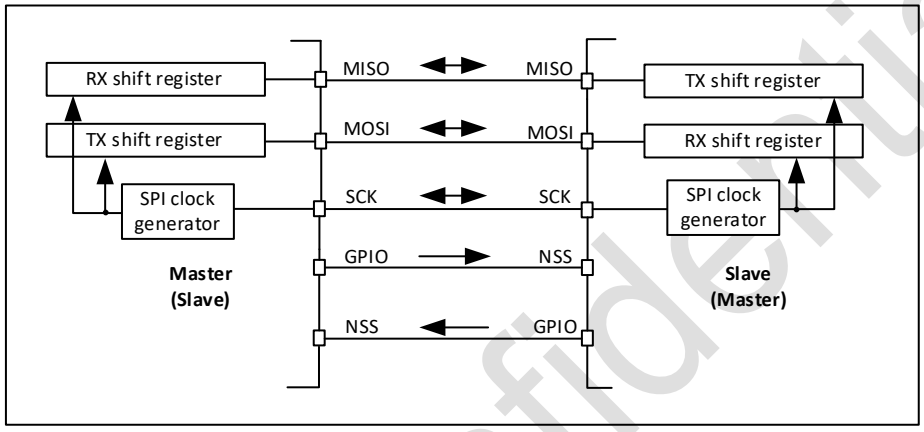


图 34-6 多主机应用

NSS 引脚在两个节点都被配置成硬件输入模式。当被动节点被配置成从机时，其有效电平将使能 MISO 输出控制。

34.3.5. 从机选择接口 (NSS)

在从机模式，NSS 作为标准的片选输入，使从机能与主机通讯。在主机模式，NSS 既可以作为输出又可以作为输入。当作为输入时，它可以防止多主机的总线冲突，当作为输出时，它可以驱动单个从机的从机选择信号。

可以使用 SPI_CR1 寄存器的 SSM 位设置硬件或者软件从机选择管理：

- 软件 NSS 管理（SSM=1）：在这个配置下，由 SPI_CR1 寄存器中的 SSI 位的值内部驱动从机选择信息。外部 NSS 引脚空闲，可供其他应用使用。
- 硬件 NSS 管理（SSM=0）：在这个情况下，有几个可能的配置：
  - NSS 输出使能（SSM=0, SSOE=1）：这个配置仅在作为主机时使用。硬件管理 NSS 引脚。只要在主机模式下使能 SPI (SPE=1)， NSS 信号便会被驱动为低电平，并且会一直保持低电平状态，直至关闭 SPI (SPE =0)。在多主机应用中，SPI 不能进行这种 NSS 配置。
  - NSS 输出禁止（SSM=0, SSOE=0）：如果 MCU 在总线上作为主机，此配置可实现多主模式功能。如果在该模式下将 NSS 引脚拉至低电平，SPI 将进入主模式故障状态，器件将在从模式下自动进行重新配置。在从机模式，NSS 引脚作为标准的片选输入，当 NSS 为低时选择从机。

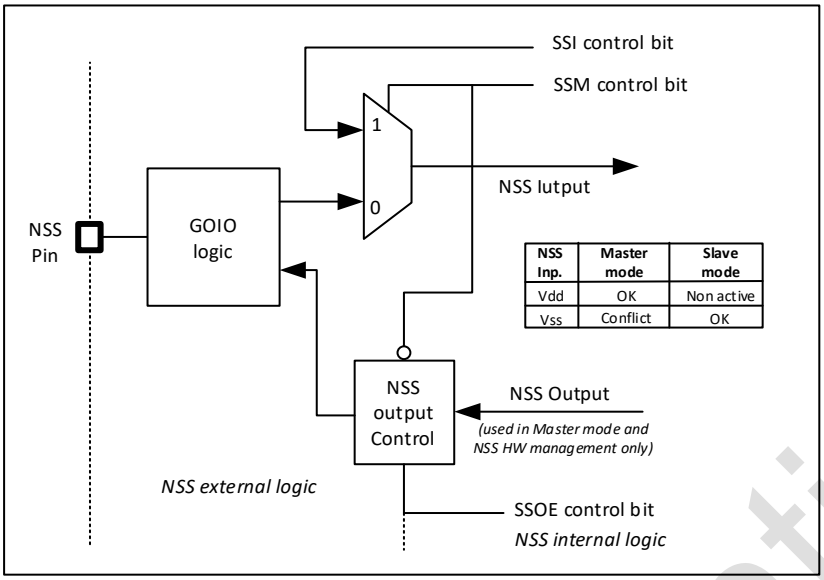


图 34-7 硬件/软件从模式选择管理

34.3.6. 通信格式

在 SPI 通讯期间，接收和发送操作同时进行。串行时钟（SCK）将数据线上的信息移位和采样操作同步。通讯格式取决于时钟相位、时钟极性和数据帧格式。为了能够进行通讯，主机和从机必须遵循相同的通讯格式。

34.3.6.1. 时钟相位和极性控制

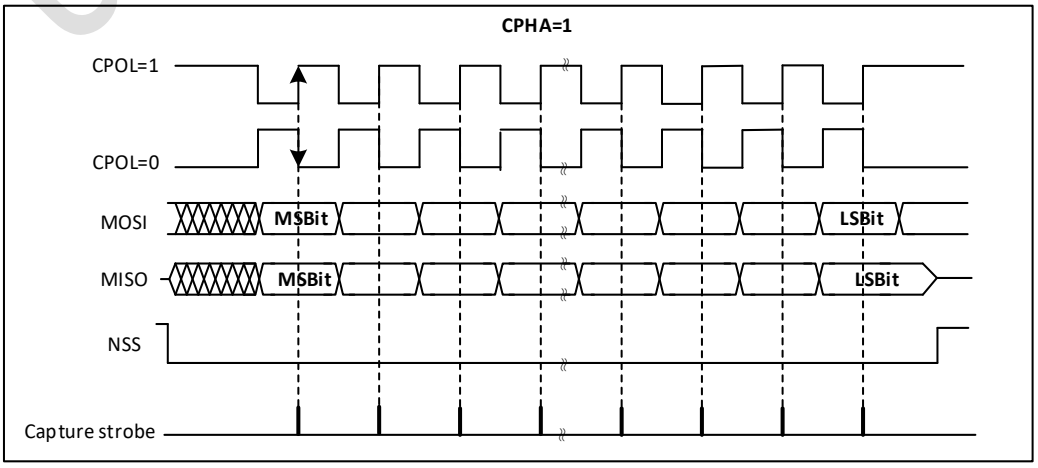
通过 SPI_CR1 寄存器中的 CPOL 和 CPHA 位，可以用软件选择四种可能的时序关系。CPOL (时钟极性) 位控制不传输任何数据时的时钟空闲状态值。此位对主器件和从器件都有作用。如果复位 CPOL，SCK 引脚在空闲状态处于低电平。如果将 CPOL 置 1，SCK 引脚在空闲状态处于高电平。

如果将 CPHA 位置 1,则会在 SCK 引脚的第二个边沿捕获传输的第一个数据位(如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。如果将 CPHA 位复位，则会在 SCK 引脚的第一个边沿捕获传输的第一个数据位（如果将 CPOL 位置 1，则为下降沿；如果将 CPOL 位复位，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。

CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

在 CPOL/CPHA 改变之前，SPI 必须被禁止（SPE=0）。

SCK 的空闲状态必须与 SPI_CR1 寄存器选择的极性对应。



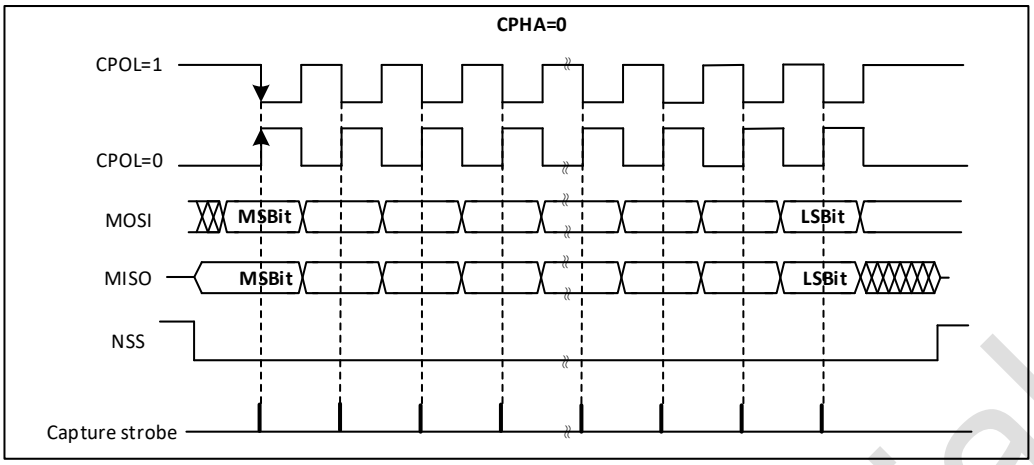


图 34-8 时钟/数据时序图

数据位的顺序取决于 LSBFIRST 位的设置。

34.3.6.2. 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 LSBFIRST 位的值。通过使用 DFF 位(SPI_CR1 寄存器)，选择数据帧的位数。可选择为 8 位或者 16 位长度，该设置对于发送和接收都适用。

34.3.7. 配置 SPI

对于主机和从机，SPI 的配置流程几乎一样。对于具体的模式建立，遵循专门的章节介绍。当进行标准的通讯，进行以下步骤：

1. 写相关的 GPIO 寄存器：配置 MOSI、MISO 和 SCK 引脚
2. 写 SPI_CR1 寄存器
   1) 通过 BR[2:0]配置时钟波特率（从机模式不需要）
   2) 配置 CPOL 和 CPHA，定义数据传输和串行时钟之间的关系
   3) 通过 RXONLY 或者 BIDIMODE 和 BIDIOE（RXONLY 和 BIDIMODE 不能同时有效），选择单工或者半双工模式
   4) 配置 LSBFIRST 定义帧格式
   5) 如果需要 CRC，则配置 CRCL 和 CRCEN 位（ SCK 时钟信号处于空闲状态时）
   6) 配置 SSM 和 SSI
   7) 配置 MSTR 位（在多主机 NSS 配置中，如果主机被配置防止 MODF 错误，要避免 NSS 的冲突状态)
   8) 配置 DFF 位，选择数据帧位数
3. 写 SPI_CR2 寄存器
   1) 配置 SSOE（从机模式不需要）
4. 写 SPI_CRCPR 寄存器：需要时配置 CRC 多项式
5. 写相应的 DMA 寄存器 (包括 SYSCFG_CFGR3 和 SYSCFG_CFGR4 配置 DMA 通道)：配置 DMA 的 SPI TX 和 Rx 通道

34.3.8. 使能 SPI 步骤

建议在主机发送时钟前使能 SPI 从机。否则，数据传输可能会不正常。从机的数据寄存器必须包含待发送的数据才能开始与主机通信（在通信时钟的第一个边沿；如果时钟信号连续，则是在正在进行的通信结束前）。使能 SPI 从机前，SCK 信号必须稳定为所选极性对应的空闲状态电平。

当 SPI 被使能并且 TXFIFO 不空，或者向 TXFIFO 进行下一个写操作，全双工模式（或者只发送）的主机开始通讯。

在任何主机只接收模式（RXONLY=1，或者 BIDIMODE=1 且 BIDIOE=0），SPI 使能后，主机开始通讯且主机立即开始提供时钟。

对于 DMA 处理，遵从具体的章节内容。

34.3.9. 数据发送和接收

34.3.9.1. RXFIFO 和 TXFIFO

SPI 所有数据通讯都通过深度为 4，宽度为 16bit（当数据帧设置为 8bit 时，宽度为 8bit）的 FIFO。该特性使 SPI 能够以连续数据流进行工作，并防止由于 CPU 来不及处理数据导致的通讯问题。发送和接收有独立的 FIFO，叫做 TXFIFO 和 RXFIFO。这些 FIFO 被用在所有的 SPI 模式。

FIFO 的处理取决于多种参数，包括：数据交换模式（全双工、半双工）、数据帧格式。

读 SPI_DR 寄存器会得到最早存放在 RXFIFO 中还未被读走的数据结果。写 SPI_DR 寄存器，会在 FIFO 发送队列的最后位置，存入被写的数据。读写访问必须与数据宽度对齐。FTLVL[1:0]和 FRLVL[1:0]位显示了两个 FIFO 当前的占用级别。

对 SPI_DR 寄存器的读访问必须通过 RXNE 事件管理。当 RXFIFO 数据非空，该事件被触发。当 RXNE 被清零，RXFIFO 就被认为是空的。

相似地，写要发送的数据帧，通过 TXE 事件管理。当 TXFIFO 的占用级别小于或者等于总容量的一半时，该事件就会被触发。否则，TXE 被清零，并且 TXFIFO 被认为是满的。

用这样的方式，RXFIFO 和 TXFIFO 都可以存 4 个数据帧。

TXE 和 RXNE 事件都都可以通过查询或者中断方式处理。

另一种管理数据交换的方式是使用 DMA。

如果在 RXFIFO 已满时收到下一个数据，将发生上溢事件（OVR 标志）。上溢事件可通过轮询或中断方式来处理。

BSY 位被置 1 表示当前正在处理数据帧。当时钟信号连续运行时，在主机中，BSY 标志在数据帧之间保持置 1 状态，但在从机中，BSY 标志在数据帧传输之间变为低电平并持续最短的一段时间（一个 SPI 时钟）。

某些应用场景下，当向 TXFIFO 中写入数据，可以通过设置 CLRTXFIFO 位，清空 TXFIFO 数据，从而往 TXFIFO 里写新的数据重新进行通信。

34.3.9.2. 序列处理

一些数据帧可以通过单序列传递来完成一条信息。使能发送后，当主机的 TXFIFO 里有任何数据，序列开始，只要主器件的 TXFIFO 中存在数据便一直继续。时钟信号由主机连续提供，直到 TXFIFO 为空，然后时钟信号停止，等待其他的数据。

在只接收模式，即半双工（BIDIMODE=1, BIDIOE=0）或者单工模式（BIDIMODE=0,RXONLY=1），当使能 SPI 并且只激活接收模式时，主机就立即开始接收。主机一直会提供时钟并连续地接收数据，直到主机停止 SPI 或者关闭只接收模式。

当主机能够以连续的模式（SCK 信号是连续的）提供所有通讯，主机必须要考虑从机处理数据流的能力。当有必要时，主机必须降低通讯速度，并提供更慢的时钟，或者或带有足够延时的单独帧或数据段。要注意的是，对于主机或者从机来说，没有下溢错误信号，来自从机的数据始终由主机处理，即使从机无法及时正确地准备数据也是如此。对于从机，更好的方法是使用 DMA，尤其当数据帧较短且

总线速率较高的情况。

在多从器件系统中，每个序列必须通过 NSS 脉冲进行控制，从而只选择其中一个从器件进行通信。在单个从器件系统中，无需通过 NSS 来控制从器件，但此时提供此脉冲通常会更好，以在每个数据序列开始时同步从器件。NSS 可以由软件和硬件两种方式管理。

当 BSY 位置 1 时，表示正在处理数据帧事务。当所进行的帧交互完成时，RXNE 标志将置 1。最后一位采样后，整个数据帧会存储到 RXFIFO 中。

34.3.9.3. 关闭 SPI 步骤

当关闭 SPI 时，必须按照本段中介绍的关闭步骤进行操作。当外设时钟停止时，在系统进入低功耗模式前做到这一点是十分重要的。否则会损坏正在进行的交互。在某些模式下，禁止步骤是停止所进行的连续通信的唯一方式。

全双工或者只发送模式下，主机可以在停止提供要发送的数据时完成交互。在这种情况下，在最后的数据交互后，时钟被停止。在这些模式下，SPI 被禁止之前，用户必须使用标准的禁止流程。当 SPI 在主机发送时，如果此时一个帧交互正在进行，或者下一个数据帧存在 TXFIFO 中，此时关闭 SPI，则 SPI 的的状态是不可预测的。

当主机处在只接收模式，停止连续时钟的唯一方法是关闭外设（SPE=0）。这必须在最后一个数据帧传输内的特定时间段，即第一位采样与最后一位传输开始之间完成（以便接收全部数量的预期数据帧并防止在最后一个有效数据帧后读取任何其他的“空”数据）。在该模式下关闭 SPI 时必须遵循特定步骤。

关闭 SPI 后，已接收但未读取的数据始终存储在 RXFIFO 中，这些数据必须在下次使能 SPI 后进行处理，然后才能启动新序列。为防止存在未读取的数据，需确保关闭 SPI 时 RXFIFO 为空，可通过正确的关闭步骤来关闭 SPI，也可以通过控制外设复位专用的特定寄存器以软件复位的方式来初始化所有 SPI 寄存器从而关闭 SPI（RCC_APBRSTx 寄存器）。

标准的禁止流程是基于 BSY 状态，并查看 FTLVL[1:0]，以确保传输彻底完成。还可以在必须识别正在处理的传输是否结束的特定情况下完成这种检查，例如：

- 当 NSS 信号被软件管理，主机要向从机提供正确的 NSS 脉冲。
- 最后的数据帧或者 CRC 帧仍在传输过程中，来自 DMA 或者 FIFO 的交互数据流完成。

正确的禁止流程是（只接收模式除外）：

1. 等待 FTLVL[1:0]=00（没有数据要发送）
2. 等待 BSY=0（最后的数据被处理完成）
3. 关闭 SPI（SPE=0）
4. 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

对于特定只接收模式，正确的禁止流程是：

1. 在最后一个数据帧传输过程中，通过禁止 SPI（SPE=0），打断接收流程
2. 等待 BSY=0（最后的数据帧已被处理）
3. 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

34.3.9.4. DMA 通信

为了以最大速度工作并且方便避免上溢所需的数据寄存器读/写过程， SPI 提供了 DMA 功能，该功能采用了简单的请求/应答协议。

当 TXE 或者 RXNE 置位，会产生 DMA 请求。Tx 和 Rx 缓冲区有独立的请求。

- 发送时，每次 TXE 置为 1，则产生 DMA 请求。然后 DMA 会向 SPI_DR 寄存器写入数据。
- 接收时，每次 RXNE 置为 1，则产生 DMA 请求。然后 DMA 读 SPI_DR 寄存器的数据。

当 SPI 被仅用作发送数据，可以只使能 SPI Tx DMA 通道。在这个情况下，因为被接收到的数据没有被读走，OVR 标志位置位。当 SPI 被仅用作接收数据，可以使能 SPI Rx DMA 通道。

在发送时，当 DMA 已经写入了所有要发送的数据（DMA_ISR 寄存器的 TCIF 标志位被置位），可以通过监测 BSY 标志来确保 SPI 通讯已完成。这是用来避免当禁止 SPI 或者进入停止模式时，最后的传输被破坏。软件必须先等待 FTLVL[1:0]=00，然后再等待 BSY=0。

通过 DMA 开始通讯时，为防止 DMA 通道管理引发错误事件发生，必须遵从以下步骤：

1. 使能 DMA Rx 缓冲区（SPI_CR2 的 RXDMAEN 位）（如果 Rx DMA 被使用）
2. 配置 SYSCFG_CFGR3.DMAx_MAP 寄存器，选择 SPI 使用的 DMA 通道；并配置 DMA 的寄存器（参考 DMA 模块的描述）
3. 使能 DMA Tx 缓冲区（在 SPI_CR2 寄存器的 TXDMAEN 位）（如果 Tx DMA 被使用）
4. 配置 SPE=1 使能 SPI

强制用以下步骤关闭通讯：

1. 如果使能了 DMA，通过操作 DMA 寄存器来关闭 DMA 模块（参考 DMA 模块的描述）
2. 通过 SPI 关闭流程关闭 SPI
3. 通过清除 TXDMAEN 和 RXDMAEN（SPI_CR2 寄存器），关闭 DMA Tx 和 Rx 缓冲区（如果 DMA Tx 和 Rx 被使用）

34.3.9.5. 时序

本节介绍一些典型的时序，这些时序对于查询、中断或者 DMA 都是有效的。为了简化，假定 LSBFIRST=0, CPOL=0, CPHA=1。也不提供完整的 DMA 操作配置。

1. 激活 NSS 并使能 SPI 后，从机开始控制 MISO；当 NSS 被释放或者 SPI 关闭时从机失去对 MISO 的控制。必须为从机提供充足的时间，以便在传输开始前准备好主机专用的数据。
   在主机端，仅在 SPI 使能后，SPI 外设才会控制 MOSI 和 SCK 信号（也包括 NSS 信号）。如果 SPI 被关闭，SPI 外设就从 GPIO 断开，在这些线上的电平值取决于 GPIO 的设定。
2. 在主机端，如果通讯是连续的，则 BSY 在帧之间保持有效。在从机端，BSY 信号在数据帧之间通常会保持至少一个时钟周期的低电平状态。
3. 只有当 TXFIFO 是满的，TXE 信号才被清零。
4. 在 TXDMAEN 位一被置位，DMA 仲裁过程即开始。在 TXEIE 被置位后，产生 TXE 中断。当 TXE 信号有效时，开始向 TxFIFO 传输数据，直到 TxFIFO 变满，或者 DMA 传输完成。
5. 如果所有要被发送的数据装进了 TxFIFO，在 SPI 总线传输前 DMA Tx TCIF 标志就会被拉高。这个标志在 SPI 交互完成之前，一直都为高。
6. 在数据封装模式，TxE 和 RxNE 事件是成对出现的，每个读/写 FIFO 的访问是 16bit 宽，直至数据帧数为偶数。如果 TxFIFO 是 3/4 满，FTLVL 状态停在 FIFO 全满级别。这就是为什么最后一个奇数帧不能在 TxFIFO 变成 1/2 满之前存储。该数据帧以 8-bit 的访问方式（由 DMA 内部信号在 LDMA_TX 置 1 时自动实现）存储在 TxFIFO 中。
7. 为了接收数据封装模式的最后一个奇数数据帧，当最后一个数据帧被处理时，Rx 阈值必须被改变成 8-bit（由 DMA 内部信号在 LDMA_RX 置 1 时自动实现）。

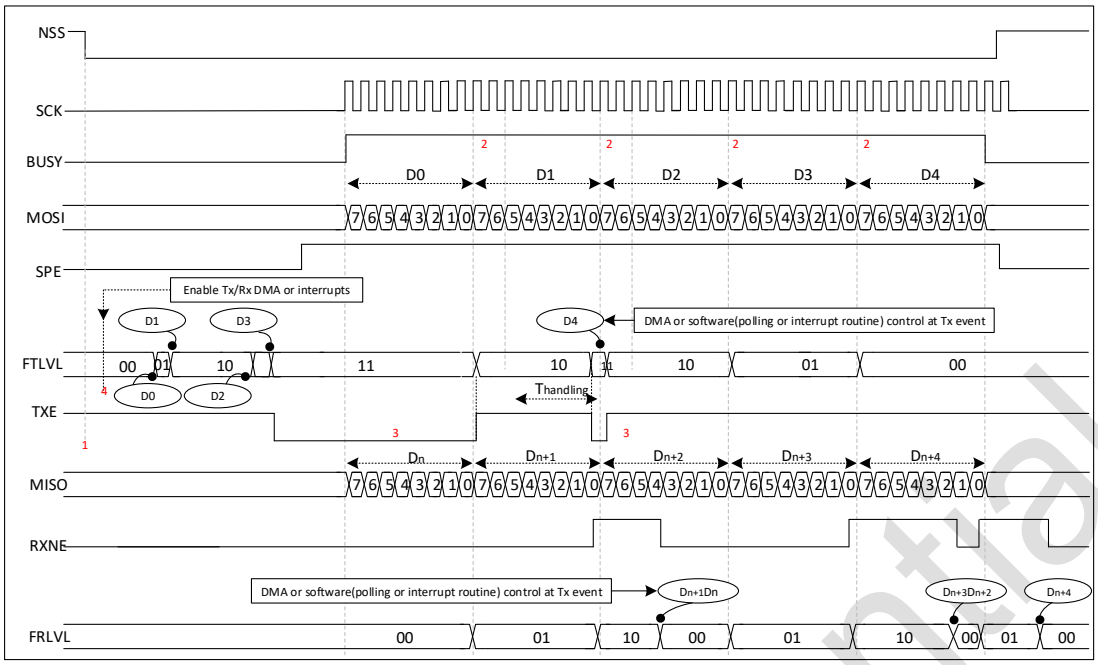


图 34-9 主模式全双工通讯(帧长=8)

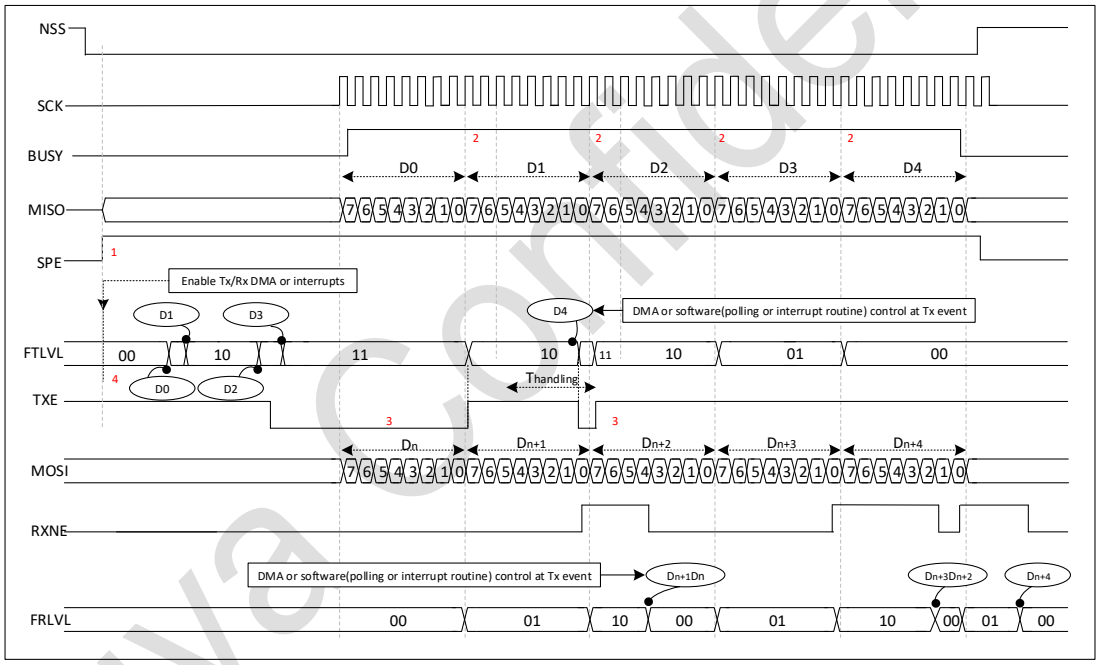


图 34-10 从模式全双工通讯(帧长=8)

34.3.10. 状态标志

应用程序通过 3 个状态标志可以完全监控 SPI 总线的状态。

34.3.10.1. 发送缓存空标志(TXE)

当 TXFIFO 有足够的空间存放要发送的数据时，TXE 标志位置位。TXE 标志位与 TXFIFO 占用级别有关。该标志位变高并保持高电平，直到 TXFIFO 占用级别小于等于 1/2 FIFO 深度。如果 SPI_CR2 寄存器的 TXEIE 置位，则会产生中断请求。当 TXFIFO 占用级别大于 1/2，该位被自动清零。

34.3.10.2. 接收缓存非空(RXNE)

当接收到数据，RXNE 置位。

如果 SPI_CR2 寄存器的 RXNEIE 位置位，则产生中断。

当上述条件不再成立，则 RXNE 被硬件自动清零。

34.3.10.3. 忙标志(BSY)

BSY 标志由硬件设置与清除(写入此位无作用)，此标志表明 SPI 正在进行数据传输（ SPI 总线繁忙）。

当它被设置为'1'时，表明 SPI 正忙于通信，但有一个例外：主模式的双向接收模式下(MSTR=1、IDIMODE=1 并且 BIDIOE=0)，在接收期间 BSY 标志保持为低（接收 CRC 期间除外）。

在软件要关闭 SPI 模块并进入停止低功耗模式(或关闭设备时钟)之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输。

BSY 标志还可以用于在多主机系统中避免写冲突。

除了主模式的双向接收模式(MSTR=1、IDIMODE=1 并且 BIDIOE=0)，当传输开始时， BSY 标志被置'1'。

以下情况该标志将被清除为'0'：

- 正确关闭 SPI
- 主机模式，当检测到故障时（MODF=1）
- 主机模式，完成了数据发送并且不准备发送任何新数据时
- 从机模式，在每个数据传输之间，BSY 标志置为 0，并保持至少一个 SPI 时钟周期

注: 不要使用 BSY 标志处理每个数据发送和接收。使用 TXE 和 RXNE 更合适。

34.3.11. 错误标志

34.3.11.1. 主模式故障（MODF)

主模式故障（MODF）仅发生在：当 NSS 作为输入信号（SSOE=0），NSS 引脚硬件管理模式下，主机的 NSS 脚被拉低；或者在 NSS 引脚软件管理模式下，SSI 位被置为'0'时，MODF 位被自动置位。主模式故障对 SPI 设备有以下影响：

- MODF 位置为'1'，如果设置了 ERRIE 位，则产生 SPI 中断；
- SPE 位被清为'0'。这将停止一切输出，并且关闭 SPI 接口；
- MSTR 位被清为'0'，因此强迫此设备进入从模式。

下面的步骤用于清除 MODF 位：

1. 当 MODF 位被置为'1'时，执行一次对 SPI_SR 寄存器的读或写操作；
2. 然后写 SPI_CR1 寄存器。

在有多个 MCU 的系统中，为了避免出现多个从机的冲突，必须先拉高该主机的 NSS 引脚，再对 MODF 位进行清零。在完成清零之后， SPE 和 MSTR 位可以恢复到它们的原始状态。

出于安全的考虑，当 MODF 位为'1'时，硬件不允许设置 SPE 和 MSTR 位。

通常配置下，从机的 MODF 位不能被置为'1'。然而，在多主机配置里，一个设备可以在设置了 MODF 位的情况下，处于从机模式；此时，MODF 位表示可能出现了多主冲突。中断程序可以执行一个复位或返回到默认状态来从错误状态中恢复。

34.3.11.2. 上溢（OVF)

当主机或者从机接收了数据，但 RXFIFO 没有足够的空间存储接收到的数据时，产生上溢情况。如果软件或者 DMA 没有足够的时间读走之前接收到的数据（RXFIFO 中存放），该情况就会发生。

当上溢情况发生，新收到的数据不会覆盖以前存放在 RXFIFO 的数据。接收到的新数据被忽略，并且丢弃所有接下来发送的数据。

依次读出 SPI_DR 寄存器和 SPI_SR 寄存器可将 OVR 清除。

34.3.11.3. CRC 错误（CRCERR）

当 SPI_CR1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。如果移位寄存器中接收的值与 SPI_RXCRC 的值不匹配， SPI_SR 寄存器中的 CRCERR 标志将置 1。该标志由软件清零。

34.3.11.4. TI 模式帧格式错误（FRE）

如果 SPI 在从模式下工作，并配置为符合 TI 模式协议，则在通信进行期间出现 NSS 脉冲时，将检测到 TI 模式帧格式错误。出现此错误时，SPI_SR 寄存器中的 FRE 标志将置 1。发生错误时不会关闭 SPI，但会忽略 NSS 脉冲，并且 SPI 会等待下一个 NSS 脉冲，然后再开始新的传输。由于错误检测可能导致丢失两个数据字节，因此数据可能会损坏。

读取 SPI_SR 寄存器时，将清零 FRE 标志。如果 ERRIE 位置 1，则检测到 NSS 错误时将生成中断。在这种情况下，由于无法保证数据的一致性，应关闭 SPI，并在重新使能从 SPI 后，由主器件重新发起通信。

34.3.12. TI 模式

34.3.12.1. 主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPI_CR2 寄存器的 FRF 位来配置 SPI，以兼容此协议。

时钟极性和相位都被强制为遵循 TI 协议，和 SPI_CR1 中的设置无关。NSS 管理也特定于 TI 协议，在这种情况下，无法通过 SPI_CR1 和 SPI_CR2 寄存器（SSM、SSI 和 SSOE）来对 NSS 管理进行配置。

在从机模式下，SPI 波特率预分频器用于控制在当前传输完成时 MISO 引脚切换为高阻态的时刻。可以使用任意波特率，因此可以非常灵活地确定此时刻。但是，波特率通常设置为外部主时钟波特率。MISO 信号变为高阻态的延时（$t_{release}$）取决于内部重新同步以及通过 SPI_CR1 寄存器的 BR[2:0] 位设置的波特率值。具体公式如下：

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

如果从机在数据帧传输期间检测到错位的 NSS 脉冲，FRE 标志将置 1。

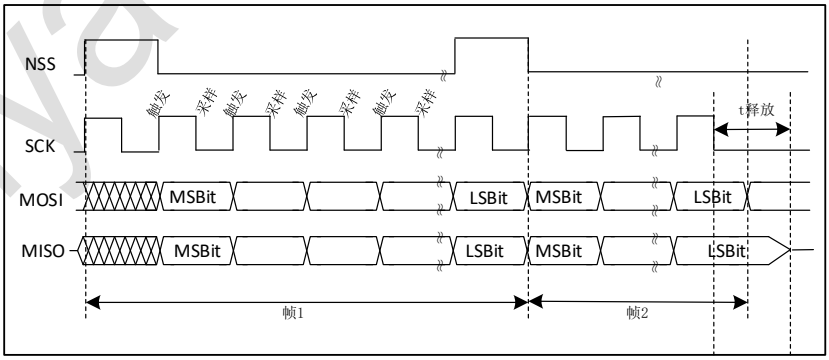


图 34-11 TI 模式传输

34.3.13. SPI CRC

为了检查发送和接收数据的可靠性，利用两个独立的 CRC 计算器。 SPI 提供 CRC8 或 CRC16 计算，且数据帧长度为 8 时选择 CRC8，数据帧长度为 16 时选择 CRC16.

34.3.13.1. CRC 原理

在 SPI 启用（SPE = 1）之前，通过设置 SPI_CR1 寄存器中的 CRCEN 位来启用 CRC 计算。CRC 值是使用每个位上的奇数可编程多项式计算的。该计算在由 SPI_CR1 寄存器中的 CPHA 和 CPOL

位定义的采样时钟边沿上进行处理。计算出的 CRC 值在数据块结束时自动校验，并且由 CPU 或 DMA 管理其传输。当检测到根据接收数据内部计算的 CRC 与发送器发送的 CRC 不匹配时，设置 CRCERR 标志以指示数据错误。处理 CRC 计算的正确程序取决于 SPI 配置和选择的传输管理方式。

34.3.13.2. CPU 管理 CRC 传输

通信开始后将一直持续到必须发送或接收 SPIx_DR 寄存器中的最后一个数据帧。之后，SPIx_CR1 寄存器中的 CRCNEXT 位必须置 1，以指示当前处理的数据帧传输后将处理 CRC 帧传输。CRCNEXT 位必须在最后一个数据帧传输结束前置 1。在 CRC 传输期间，CRC 计算将冻结。

所接收的 CRC 以数据字节或数据字的形式存储在 RXFIFO 中。因此仅在 CRC 模式下，接收缓冲区才必须被视为一个 16 位缓冲区且一次仅接收一个数据帧。

CRC 格式的传输通常在数据传输结束时需要多出一个数据帧。

接收最后一个 CRC 数据后，将执行自动校验，将接收的值与 SPI_RXCRC 寄存器中的值进行比较。软件必须校验 SPI_SR 寄存器中的 CRCERR 标志，以确定数据传输是否损坏。软件通过向 CRCERR 标志写入“0”来将其清零。

接收 CRC 后，CRC 值存储到 RXFIFO 中，且必须在 SPI_DR 寄存器中进行读取，以将 RXNE 标志清零。

34.3.13.3. DMA 管理 CRC 通信

当使能 SPI，并开启带 CRC 的 DMA 模式进行通信时，在通信结束时自动发送和接收 CRC（在仅接收模式下读取 CRC 数据除外），CRCNEXT 位并非一定要通过软件来处理。 SPI 发送 DMA 通道计数器必须设置为要发送的数据帧数，其中不包括 CRC 帧。在接收器侧，接收的 CRC 值在传输结束时通过 DMA 自动处理，但用户必须注意刷新 RXFIFO 中接收的 CRC 信息（因为该信息始终加载到其中）。在全双工模式下，接收 DMA 通道计数器可设置为要接收的数据帧数，其中包括 CRC。

在只接收模式下，DMA 接收通道计数器应仅包含已传输的数据量，而不包括 CRC 计算。之后，基于通过 DMA 实现的完整传输，必须通过软件从 FIFO 中读回所有 CRC 值，因为 FIFO 在该模式下用作单个缓冲区。

如果传输过程中出现故障，则在数据和 CRC 传输结束时，将 SPI_SR 寄存器中的 CRCERR 标志位置 1。

34.3.13.4. CRC 使用流程

使能 CRC 的流程如下：

- 配置 CPOL、CPHA、LSBFIRST、BR、SSM、SSI 和 MSTR 寄存器
- 在 SPI_CRCPR 寄存器输入多项式
- 通过设置 SPI_CR1 寄存器 CRCEN 位使能 CRC 计算，该操作也会清除寄存器 SPI_RXCRCR 和 SPI_TXCRC
- 设置 SPI_CR1 寄存器的 SPE 位启动 SPI 功能
- 启动通信并且维持通信，直到只剩最后一个字节或者半字
- 在把最后一个字节或半字写进发送缓冲器时，设置 SPI_CR1 的 CRCNEXT 位，指示硬件在发送完成最后一个数据之后，发送 CRC 的数值。在发送 CRC 数值期间，停止 CRC 计算

- 当最后一个字节或半字被发送后，SPI 发送 CRC 数值，CRCNEXT 位被清除。同样，接收到的 CRC 与 SPI_RXCRCR 值进行比较，如果比较不匹配，则设置 SPI_SR 寄存器的 CRCERR 标志位，当设置了 SPI_CR2 寄存器的 ERRIE 时，则产生中断。

注意：在使能 CRC 后，接收或者发送数据会一直做 CRC 计算。所以，如果数据不再需要 CRC 计算，要及时关闭 CRC。

如果在通信期间关闭了 SPI，再使能 SPI 及 CRC，则必须遵循以下顺序：

- 关闭 SPI
- 将 CRCEN 清零
- 使能 CECEN
- 使能 SPI

34.3.14. SPI 中断

表 34-1 SPI 中断请求

中断事件	事件标志	使能控制位
TXFIFO 等待被装载	TXE	TXEIE
数据接收到 RXFIFO 中	RXNE	RXNEIE
主模式故障事件	MODF	ERRIE
上溢错误	OVR	
TI 帧格式错误	FRE	
CRC 校验错误	CRCERR	

34.4. I²S 功能描述

34.4.1. I²S 框图

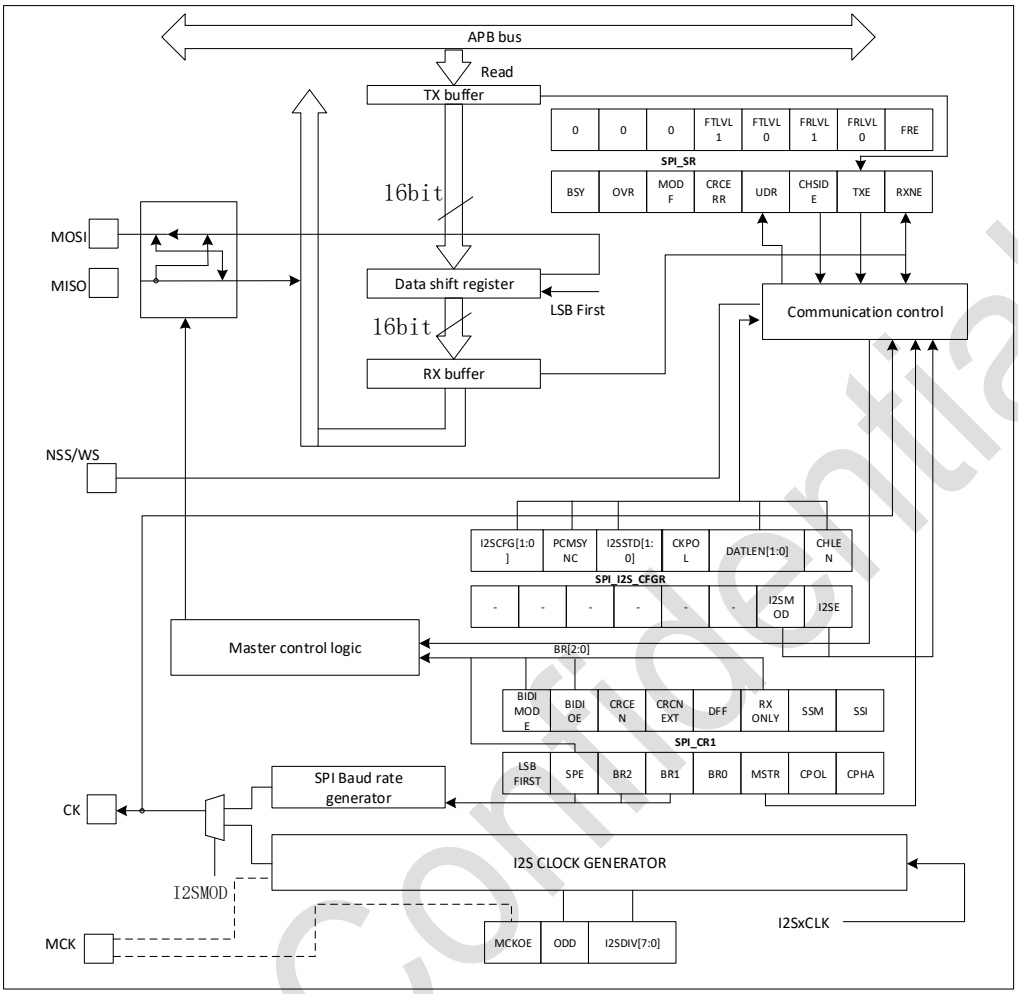


图 34-12 I²S 框图

通过将寄存器 SPI_I2SCFGR 的 I2SMOD 位置为 ‘1’，即可使能 I2S 功能。此时，可以把 SPI 模块用作 I2S 音频接口。I2S 接口与 SPI 接口使用大致相同的引脚、标志和中断。

I2S 与 SPI 共用 3 个引脚：

SD：串行数据(映射至 MOSI 引脚)，用来发送和接收 2 路时分复用通道的数据（仅半双工模式）；

WS：字选(映射至 NSS 引脚)，主模式下作为数据控制信号输出，从模式下作为数据控制信号输入；

CK：串行时钟(映射至 SCK 引脚)，主模式下作为时钟信号输出，从模式下作为时钟信号输入。

在某些外部音频设备需要主时钟时，可以另有一个附加引脚输出时钟：

MCK：主时钟(映射至 MISO 引脚)，在 I2S 配置为主模式，寄存器 SPI_I2SPR 的 MCKOE 位为 ‘1’ 时，作为输出额外的时钟信号引脚使用。输出时钟信号的频率预先设置为 256 × Fs，其中 Fs 是音频信号的采样频率。

设置成主模式时，I²S 使用自身的时钟发生器来产生通信用的时钟信号。这个时钟发生器也是主时钟输出的时钟源。I²S 模式下有 2 个额外的寄存器，一个是与时钟发生器配置相关的寄存器 SPI_I2SPR，另一个是 I²S 通用配置寄存器 SPI_I2SCFGR(可设置音频标准、从/主模式、数据格式、数据包帧、时钟极性等参数)。

在 I²S 模式下不使用寄存器 SPI_CR1 和所有的 CRC 寄存器。同样，I²S 模式下也不使用寄存器

SPI_CR2 的 SSOE 位，和寄存器 SPI_SR 的 MODF 位和 CRCERR 位。

I²S 使用与 SPI 相同的寄存器 SPI_DR 用作 16 位宽模式数据传输。

34.4.2. 支持音频协议

三线总线支持 2 个声道上音频数据的时分复用：左声道和右声道，但是只有一个 16 位寄存器用作发送或接收。因此，软件在对数据寄存器写入数据时，必须根据当前传输中的声道写入；同样，在读取寄存器数据时，必须通过检查寄存器 SPI_SR 的 CHSIDE 位来判明接收到的数据属于哪个声道。左声道总是先于右声道发送数据(CHSIDE 位在 PCM 协议下无意义)。

有四种可用的数据和包帧组合。可以通过以下四种数据格式发送数据：

- 16 位数据打包进 16 位帧
- 16 位数据打包进 32 位帧
- 24 位数据打包进 32 位帧
- 32 位数据打包进 32 位帧

在使用 16 位数据扩展到 32 位帧时，前 16 位(MSB)是有意义的数据，后 16 位(LSB)被强制为 0，该操作不需要软件干预，也不需要有 DMA 请求(仅需要一次读/写操作)。

24 位和 32 位数据帧需要 CPU 对寄存器 SPI_DR 进行 2 次读或写操作，在使用 DMA 时，需要 2 次 DMA 传输。对于 24 位数据，扩展到 32 位后，最低 8 位由硬件置 0。

对于所有的数据格式和通讯标准，总是先发送最高位(MSB)。

I²S 接口支持四种音频标准，可以通过设置寄存器 SPI_I2SCFGR 的 I2SSTD[1:0]位和 PCMSYNC 位来选择。

34.4.2.1. I²S 飞利浦标准

在此标准下，引脚 WS 用来指示正在发送的数据属于哪个声道。在发送第一位数据(MSB)前该引脚有效 1 个时钟周期。

16/32 位全精度时序图如下图所示，发送方在时钟信号(CK)的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

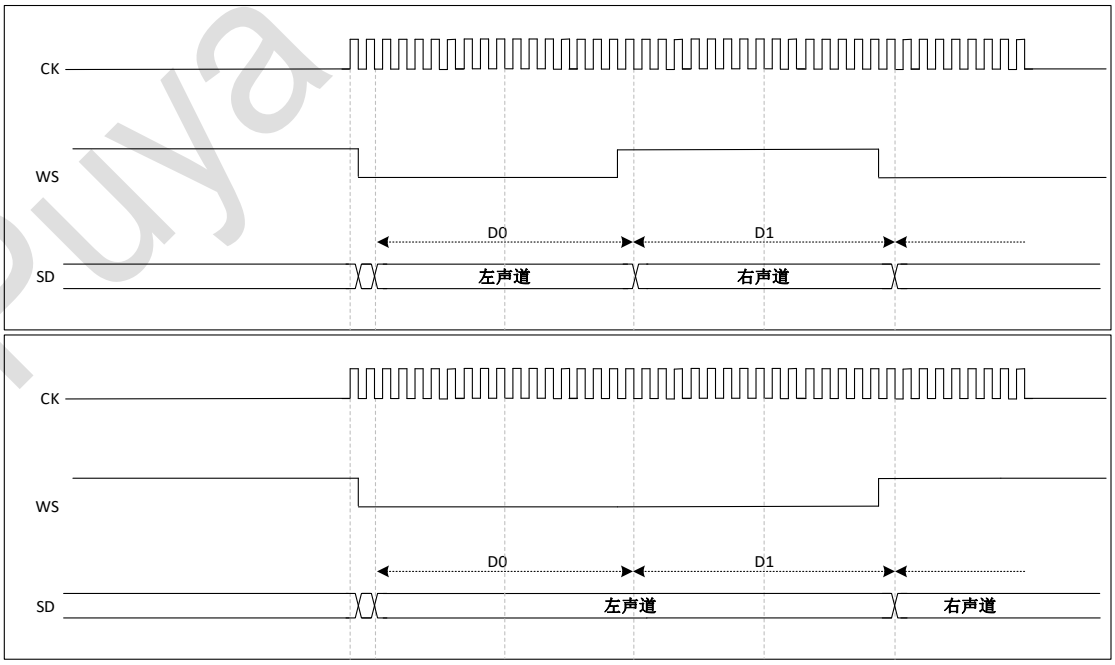


图 34-13 I²S 飞利浦协议波形 (16/32 位全精度, CKPOL=0)

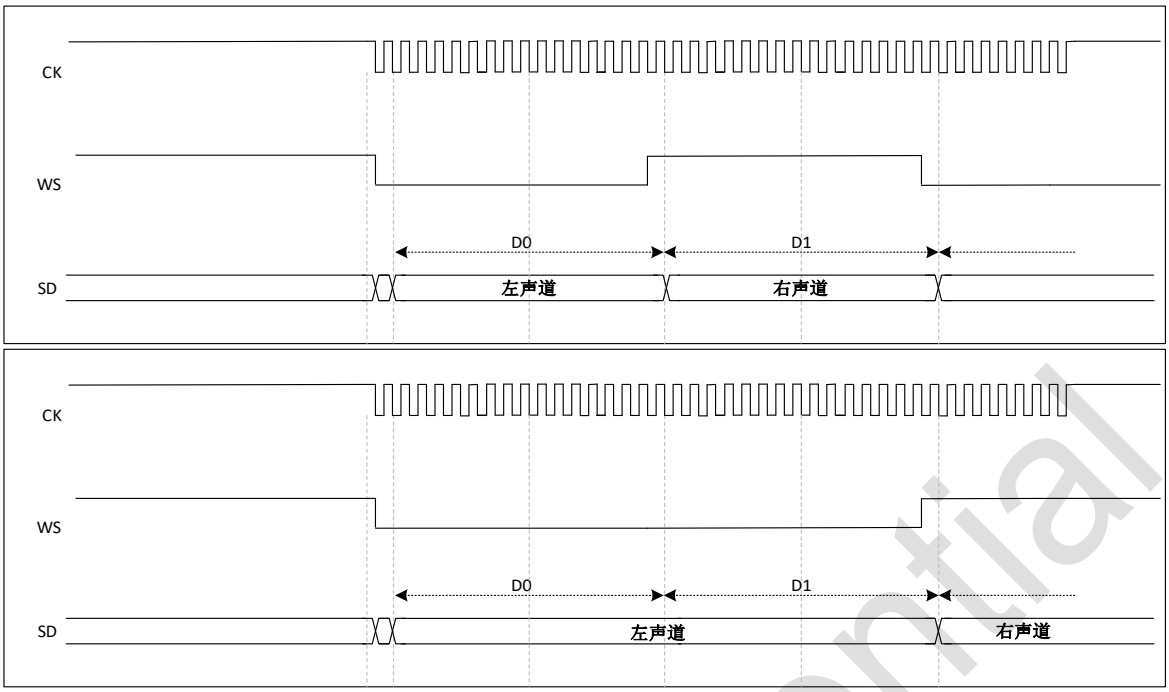


图 34-14 I²S 飞利浦协议波形 (16/32 位全精度, CKPOL=1)

24 位帧时序图如下:

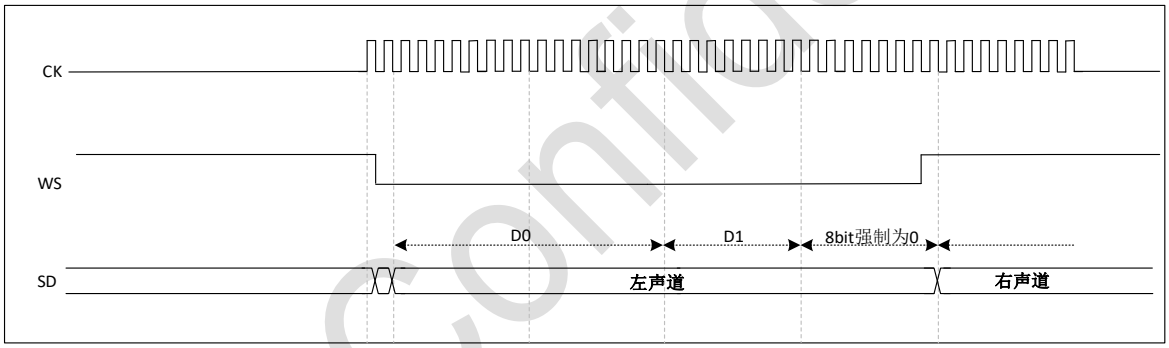


图 34-15 I²S 飞利浦协议波形(24 位帧, CKPOL=0)

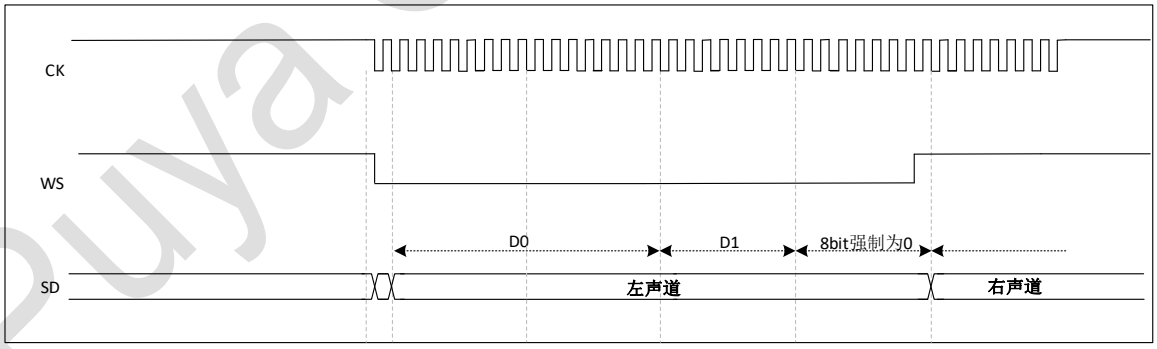


图 34-16 I²S 飞利浦协议波形(24 位帧, CKPOL=1)

此模式需要对寄存器 SPI_DR 进行 2 次读或写操作。

- 在发送模式下：如果需要发送 0x8EAA33(24 位)，第一次写入 0x8EAA，第二次写入 0x33xx，低 8 位无意义
- 在接收模式下：如果接收 0x8EAA33(24 位)，第一次接收 0x8EAA，第二次读出 0x3300，只高 8 位为有意义的数据，低 8 位始终为 0

下图为 16 位扩展到 32 位声道帧的时序:

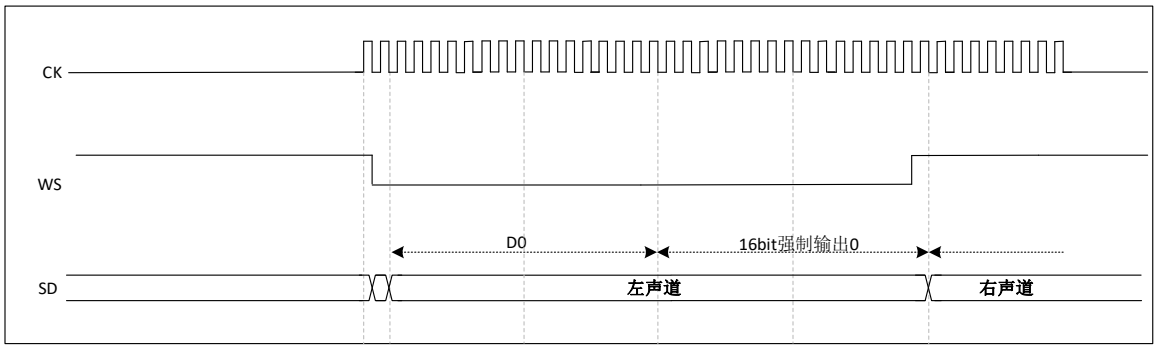


图 34-17 I²S 飞利浦协议波形(16 位扩展到 32 位)

在 I²S 配置阶段，如果选择将 16 位数据扩展到 32 位声道帧，只需要访问一次寄存器 SPI_DR。用来扩展到 32 位的低 16 位被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x76A30000)，只需要操作一次 SPI_DR，写入数据 0x76A3.

发送时，需要将 MSB 写入寄存器 SPI_DR：标志位 TXE 为 ‘1’ 表示可以写入新的数据，如果允许了相应的中断，则可以产生中断。发送是由硬件完成的，即使还未发送出后 16 位的 0x0000，也会设置 TXE 并产生相应的中断。

接收时，每次收到高 16 位半字(MSB)后，标志位 RXNE 置 ‘1’ ，如果允许了相应的中断，则可以产生中断。

这样，就延长了两个写入或读取操作之间的时间间隔，从而可防止出现下溢或上溢情况（具体取决于数据传输方向）。

34.4.2.2. MSB 对齐标准

在此标准下， WS 信号和第一个数据位，即最高位(MSB)同时产生。

16/32 位全精度时序图如下图所示,发送方在时钟信号的下降沿改变数据;接收方是在上升沿读取数据。

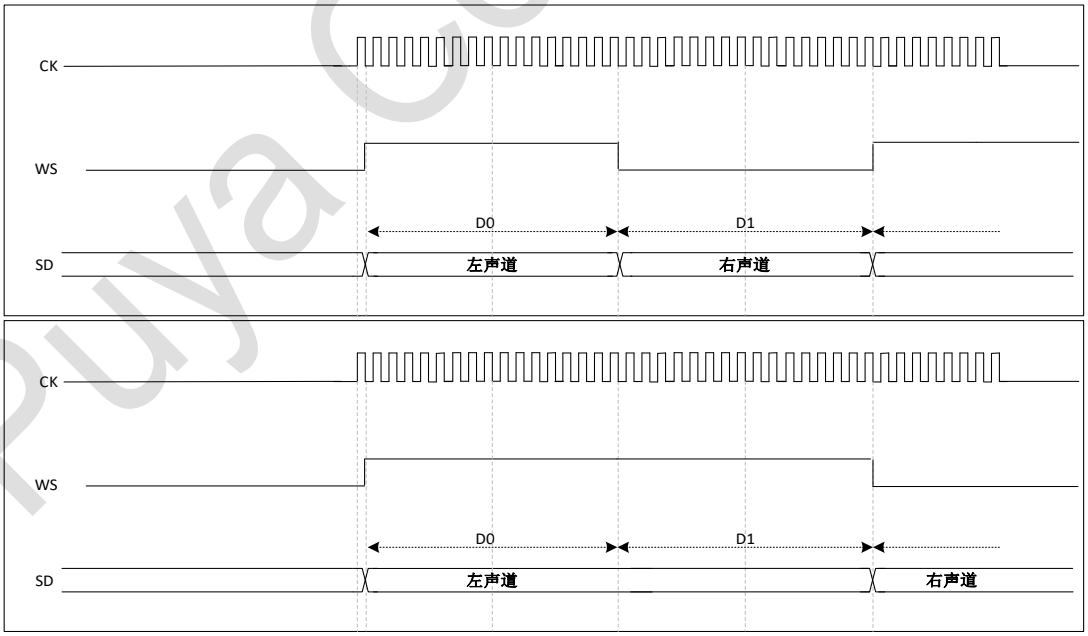


图 34-18 I²S MSB 对齐标准 (16/32 全精度, CKPOL=0)

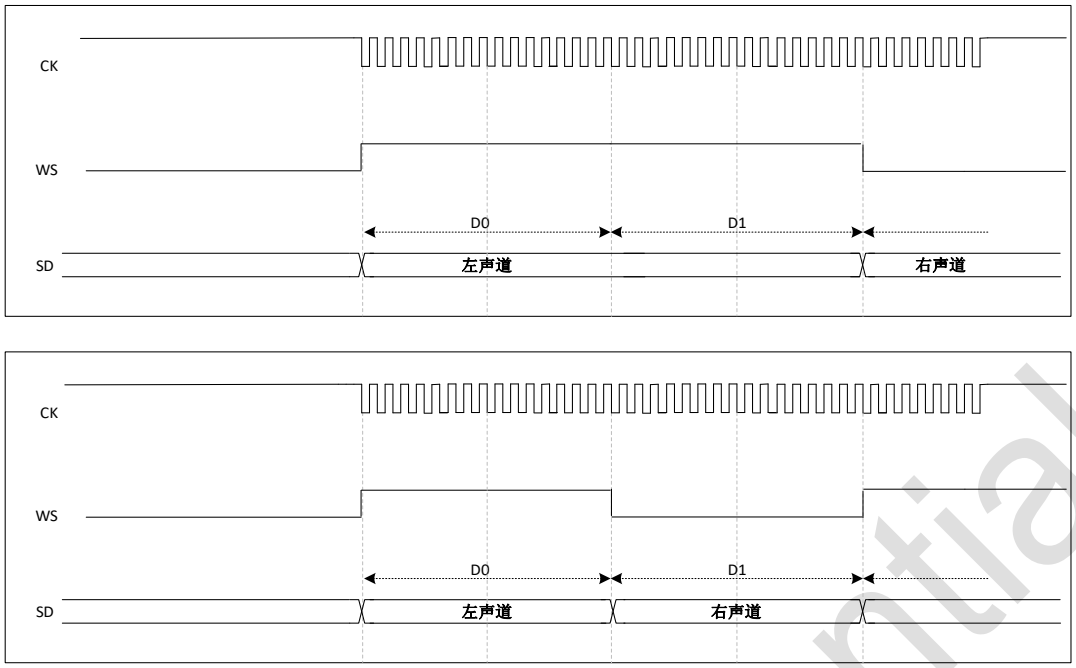


图 34-19 I²S MSB 对齐标准 (16/32 全精度, CKPOL=1)

24 位帧时序图如下：

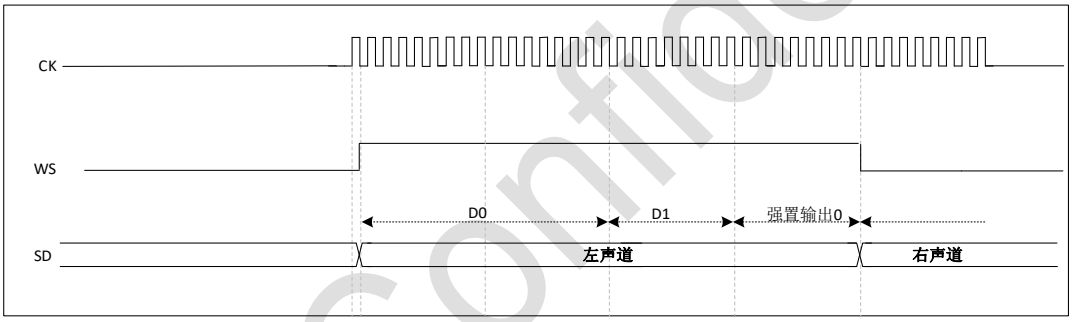


图 34-20 I²S MSB 对齐标准(24 位帧，CKPOL=0)

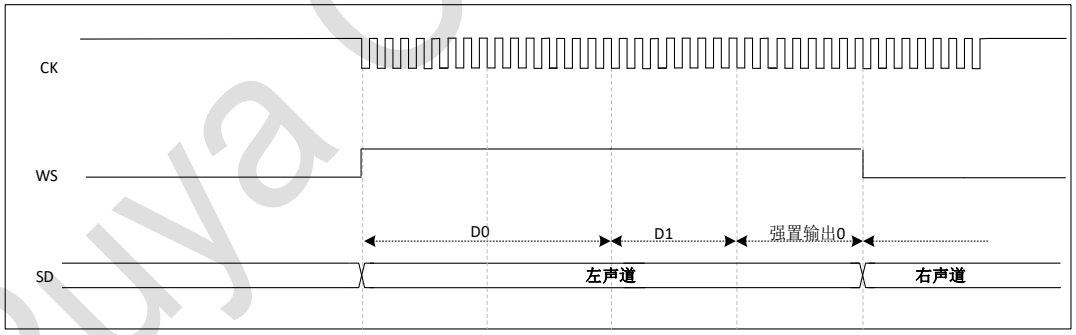


图 34-21 I²S MSB 对齐标准(24 位帧, CKPOL=1)

下图为 16 位扩展到 32 位声道帧的时序：

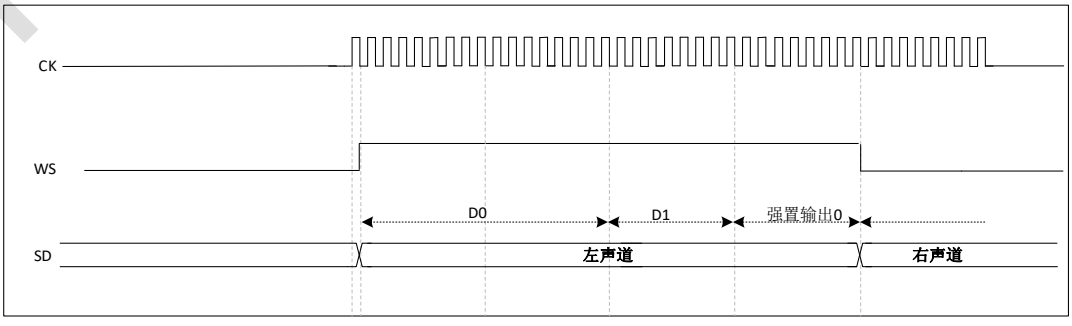


图 34-22 I²S MSB 对齐标准(16 位扩展到 32 位，CKPOL=0)

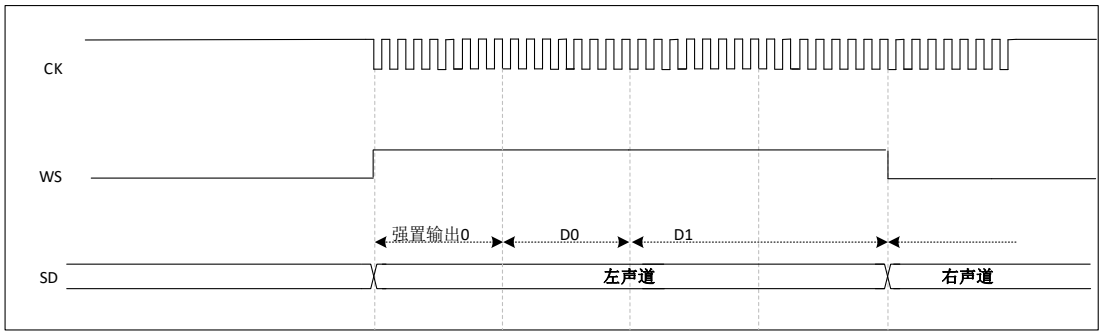


图 34-23 I²S MSB 对齐标准(16 位扩展到 32 位，CKPOL=1)

一旦有效数据开始从 SD 引脚送出，就发生下一次 TXE 事件。在接收时，一旦接收到有效数据(而不是 0x0000 部分)，就发生 RXNE 事件。

34.4.2.3. LSB 对齐标准

此标准与 MSB 对齐标准类似(在 16 位或 32 位全精度帧格式下无区别)。

16/32 位全精度时序图如下图所示，发送方在时钟信号的下降沿改变数据；接收方是在上升沿读取数据。

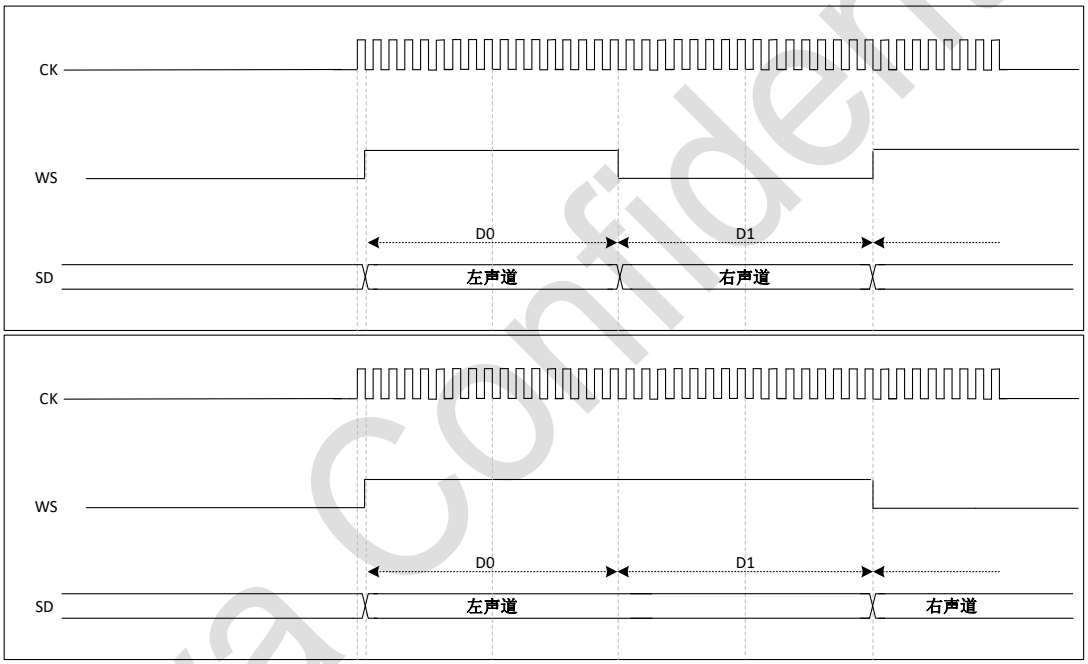


图 34-24 I²S LSB 对齐标准 (16/32 位全精度, CKPOL=0)

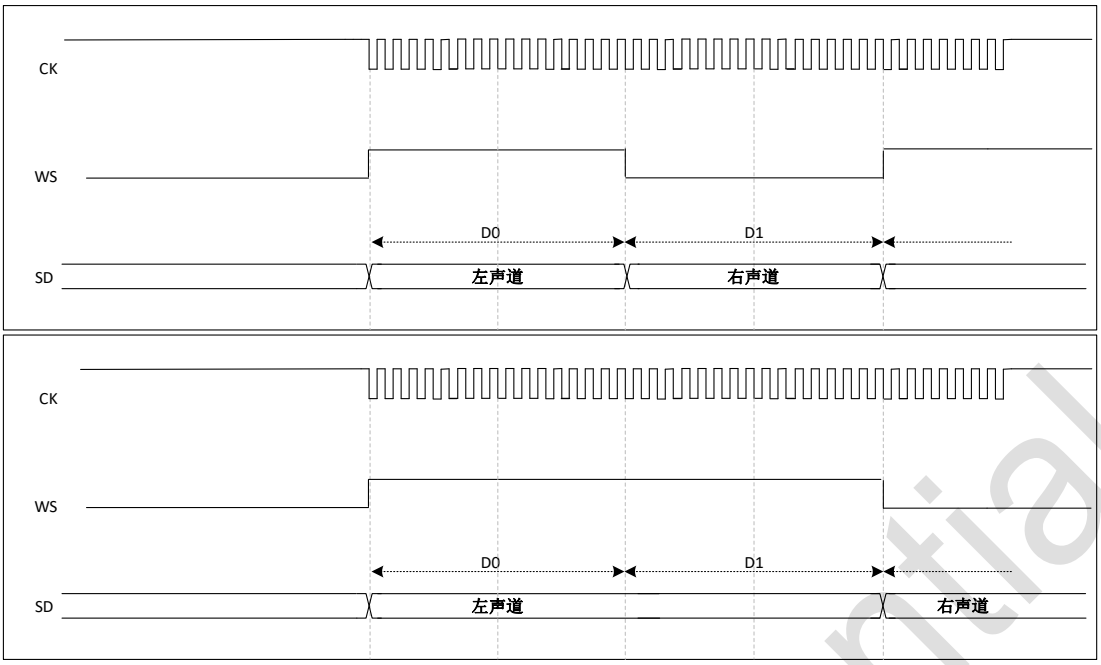


图 34-25 I²S LSB 对齐标准 (16/32 位全精度, CKPOL=1)

24 位帧时序图如下:

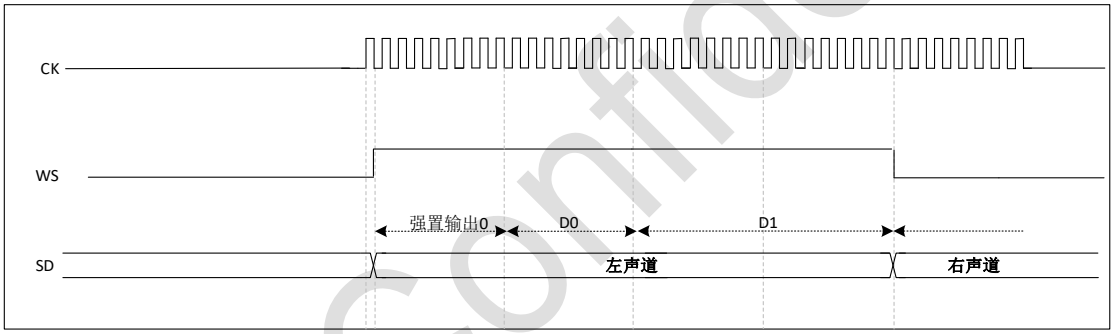


图 34-26 I²S LSB 对齐标准(24 位帧, CKPOL=0)

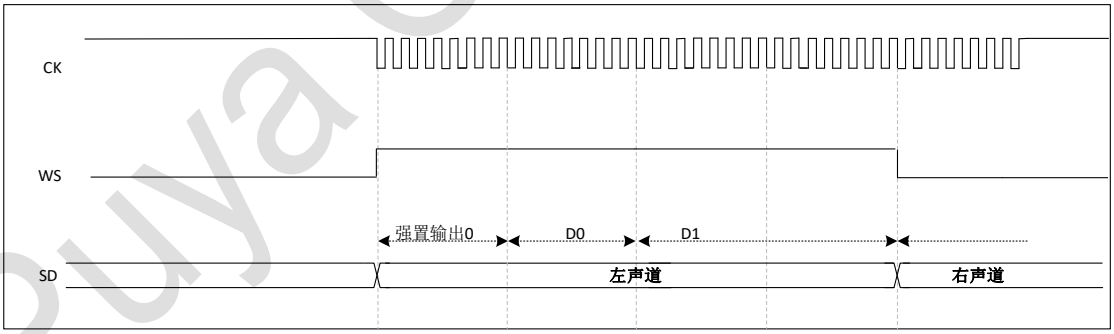


图 34-27 I²S LSB 对齐标准(24 位帧, CKPOL=1)

■ 在发送模式下

如果要发送数据 0x3478AE，需要通过软件或者 DMA 对寄存器 SPI_DR 进行 2 次写操作。第一次写入数据寄存器 0xXX34，第二次写入数据寄存器 0x78AE。

■ 在接收模式下

如果要接收数据 0x3478AE，需要在 2 个连续的 RXNE 事件发生时，分别对寄存器 SPI_DR 进行 1 次读操作。第一次读出 0x0034，只有低 8 位有意义；第二次读出 0x78AE。

下图为 16 位扩展到 32 位声道帧的时序:

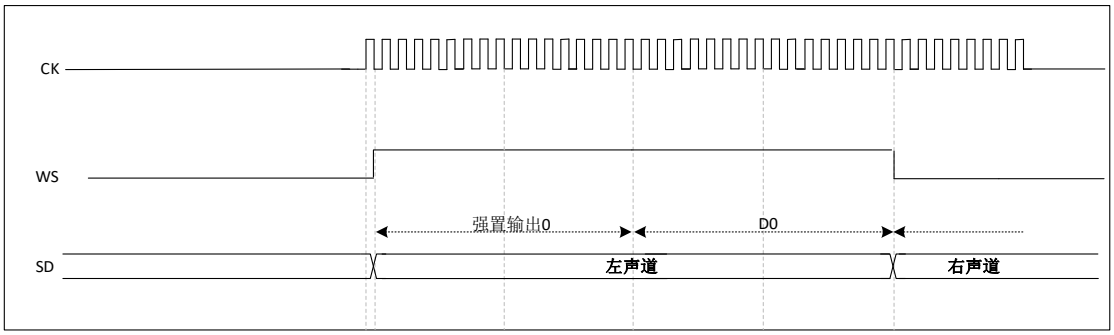


图 34-28 I²S LSB 对齐标准(16 位扩展到 32 位，CKPOL=0)

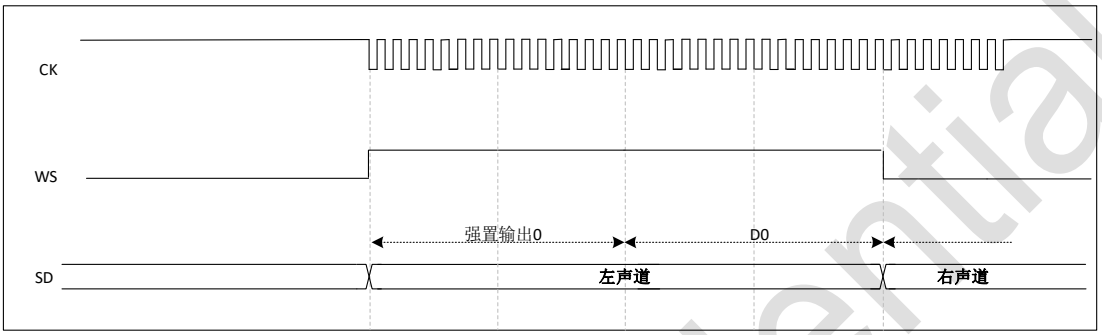


图 34-29 I²S LSB 对齐标准(16 位扩展到 32 位，CKPOL=1)

在 I²S 配置阶段，如果选择将 16 位数据扩展到 32 声道帧，只需要访问一次寄存器 SPI_DR。此时，扩展到 32 位后的高半字(16 位 MSB)被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x0000 76A3)，只需要操作一次 SPI_DR 寄存器，写入 0x76A3.

发送时，如果 TXE 为 ‘1’ ，用户需要写入待发送的数据(即 0x76A3)。用来扩展到 32 位的 0x0000，部分由硬件首先发送出去，一旦有效数据开始从 SD 引脚送出，就发生下一次 TXE 事件。

接收时，一旦接收到有效数据(而不是 0x0000 部分)，就发生 RXNE 事件。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

34.4.2.4. PCM 标准

在 PCM 标准下，不存在声道选择的信息。PCM 标准有 2 种可用的帧结构，短帧或者长帧，可以通过设置寄存器 SPI_I2SCFGR 的 PCMSYNC 位来选择。

在 PCM 模式下，输出信号（WS 和 SD）在 CK 信号的上升沿采样。输入信号（WS 和 SD）在 CK 的下降沿捕获。

请注意，CK 和 WS 在主模式下配置为输出。

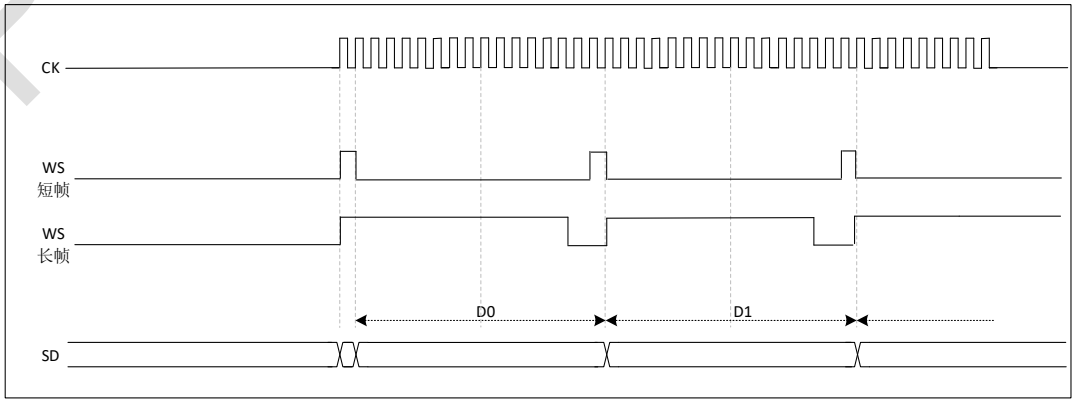


图 34-30 I²S PCM 标准(16 位帧，CKPOL=0)

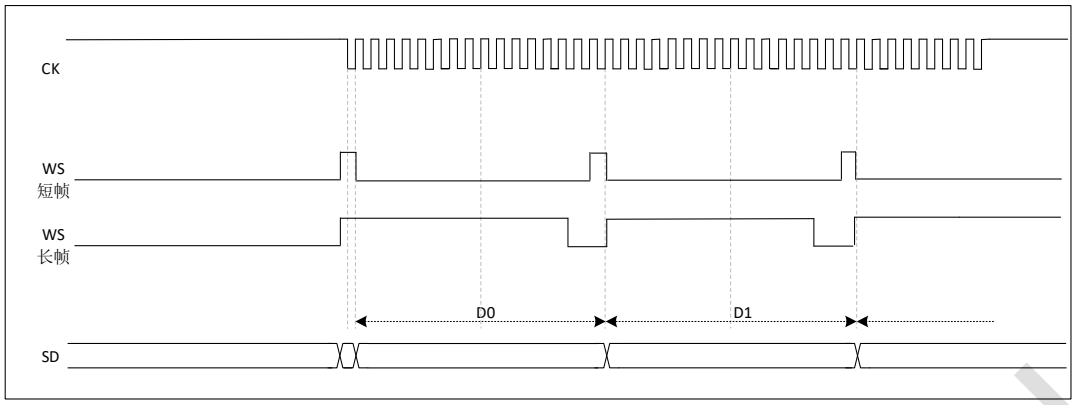


图 34-31 I²S PCM 标准 (16 位帧，CKPOL=1)

对于长帧，主模式下，用来同步的 WS 信号有效的时间固定为 13 位。

对于短帧，用来同步的 WS 信号长度只有 1 位。

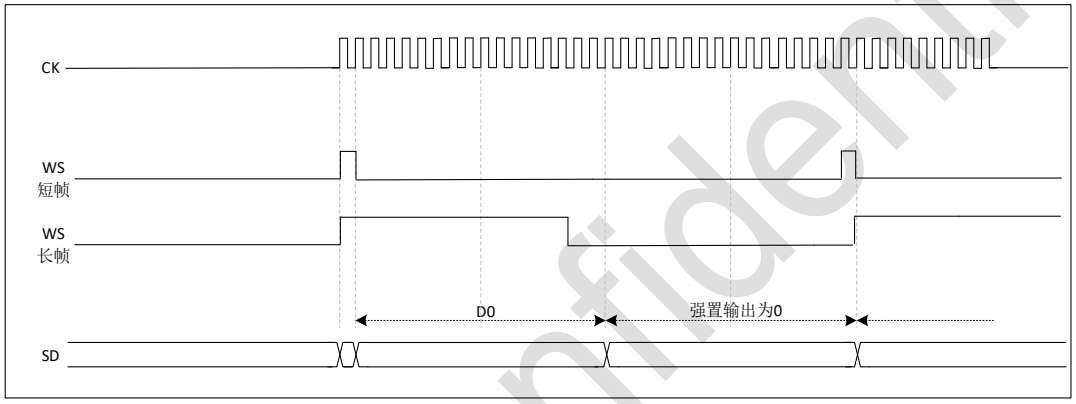


图 34-32I²S PCM 标准(16 位扩展到 32 位, CKPOL=0)

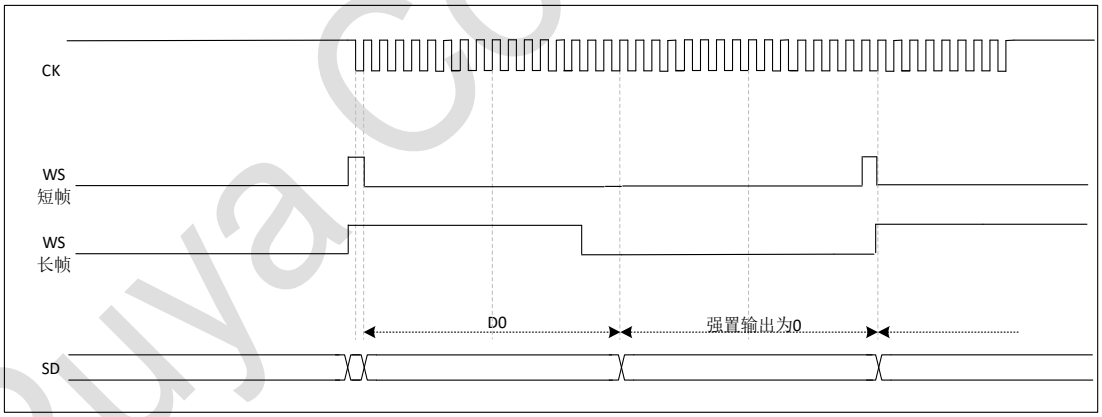


图 34-33 I²S PCM 标准 (16 位扩展到 32 位, CKPOL=1)

无论哪种模式(主或从)、哪种同步方式(短帧或长帧)，即使是从模式，需要通过设置 SPI_I2SCFGR 寄存器的 DATLEN 位和 CHLEN 位来指定连续的 2 帧数据以及 2 个同步信号之间的时间差。

34.4.3. 时钟产生

I²S 的比特率即确定了在 I²S 数据线上的数据流和 I2S 的时钟信号频率。即

I²S 比特率 = 每个声道的比特数 x 声道数目 x 音频采样频率.

对于一个具有左右声道和 16 位音频信号，I²S 比特率计算如下:

I²S 比特率 = 16×2×fs

如果包长为 32 位，则有：

I²S 比特率=32×2×fs

音频采样频率定义如下图所示：

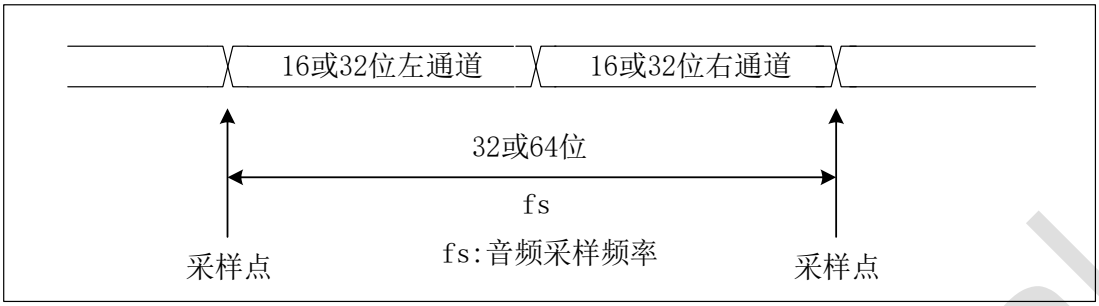


图 34-34 音频采样频率定义

在主模式下，为了获得需要的音频频率，需要正确地对线性分频器进行设置。

I²S 时钟发生器结构如下图所示：

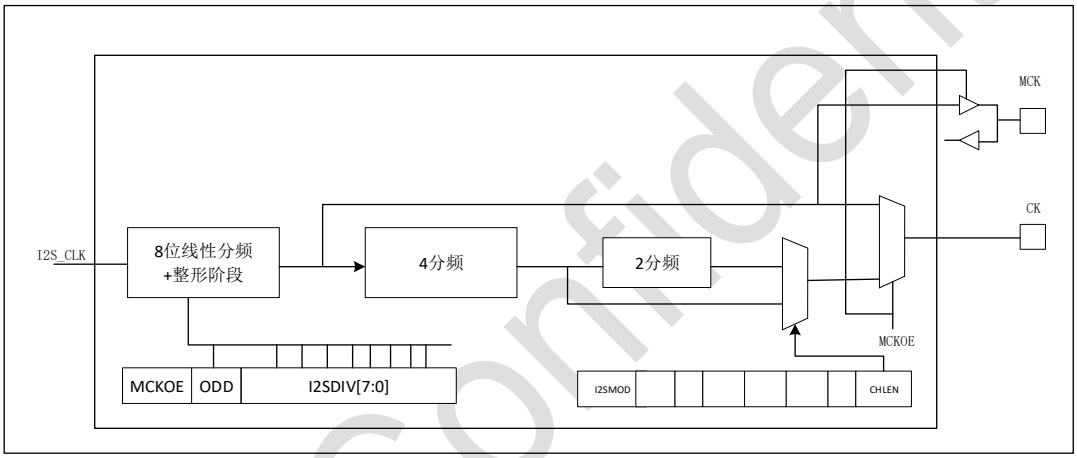


图 34-35 I²S 时钟发生器结构

I2S_CLK 由 RCC 模块产生，且与 PCLK 同步。

音频的采样频率可以是 192kHz、96 kHz、48 kHz、44.1 kHz、32 kHz、22.05 kHz、16 kHz、11.025 kHz 或者 8 kHz(或任何此范围内的数值)。为了获得需要的频率，需按照以下公式设置线性分频器而获得：

34.4.3.1. I²S 模式时钟产生

输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为 ‘1’)：

$fs=f_{I2S\_CLK}/[256x((2xI2SDIV)+ODD)]$

禁止输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为 ‘0’)：

声道的帧长为 16 位时，$fs=f_{I2S\_CLK}/[32x((2xI2SDIV)+ODD)]$

声道的帧长为 32 位时，$fs=f_{I2S\_CLK}/[64x((2xI2SDIV)+ODD)]$

34.4.3.2. PCM 模式时钟产生

输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为 ‘1’)：

$fs=f_{I2S\_CLK}/[128x((2xI2SDIV)+ODD)]$

禁止输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为 ‘0’)：

声道的帧长为 16 位时，$fs=f_{I2S\_CLK}/[16x((2xI2SDIV)+ODD)]$

声道的帧长为 32 位时，$fs=f_{I2S\_CLK}/[32x((2xI2SDIV)+ODD)]$

使用标准的 8MHz HSE 时钟得到精确的音频频率，见下表。

表 34-2 使用 8MHz HSE 的音频频率精度

SYSCLK (MHz)	I2SDIV		ODD		MCLK	期望 fs(Hz)	实际 fs(Hz)		误差	
	16 位	32 位	16 位	32 位			16 位	32 位	16 位	32 位
72	11	6	1	0	无	96000	97826.09	93750	1.90%	2.34%
72	23	11	1	1	无	48000	47872.34	48913.04	0.27%	1.90%
72	25	13	1	0	无	44100	44117.65	43269.23	0.04%	1.88%
72	35	17	0	1	无	32000	32142.86	32142.86	0.45%	0.45%
72	51	25	0	1	无	22050	22058.82	22058.82	0.04%	0.04%
72	70	35	1	0	无	16000	15957.45	16071.43	0.27%	0.45%
72	102	51	0	0	无	11025	11029.41	11029.41	0.04%	0.04%
72	140	70	1	1	无	8000	8007.117	7978.723	0.09%	0.27%
72	2	2	0	0	有	96000	70312.5	70312.5	26.76%	26.76%
72	3	3	0	0	有	48000	46875	46875	2.34%	2.34%
72	3	3	0	0	有	44100	46875	46875	6.29%	6.29%
72	4	4	1	1	有	32000	31250	31250	2.34%	2.34%
72	6	6	1	1	有	22050	21634.62	21634.62	1.88%	1.88%
72	9	9	0	0	有	16000	15625	15625	2.34%	2.34%
72	13	13	0	0	有	11025	10817.31	10817.31	1.88%	1.88%
72	17	17	1	1	有	8000	8035.714	8035.714	0.45%	0.45%

34.4.4. I²S 传输

34.4.4.1. 主模式

I²S 可配置为主模式。这意味着将在 CK 引脚输出串行时钟，在 WS 引脚生成字选信号。主时钟 (MCK) 可以输出，也可以不输出，具体由 SPI_I2SPR 寄存器中的 MCKOE 位控制。

配置流程如下：

1. 设置寄存器 SPI_I2SPR 的 I2SDIV[7:0]定义与音频采样频率相符的串行时钟波特率。同时也要定义寄存器 SPI_I2SPR 的 ODD 位。

2. 设置 CKPOL 位定义通信用时钟在空闲时的电平状态。如果需要向外部的 DAC/ADC 音频器件提供主时钟 MCK，则将寄存器 SPI_I2SPR 的 MCKOE 位置为 ‘1’ ，并按照不同的 MCK 输出状态，计算 I2SDIV 和 ODD 的值。

3. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位为 ‘1’ 激活 I²S 功能，设置 I2SSTD[1:0]和 PCMSYNC 位选择所用的 I²S 标准，设置 CHLEN 选择每个声道的数据位数。还要设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I²S 主模式和方向（发送器或者接收器）。

4. 如果需要，可以通过设置寄存器 SPI_CR2 来打开所需的中断功能和 DMA 功能。

5. 必须将寄存器 SPI_I2SCFGR 的 I2SE 位置为 ‘1’ 。

引脚 WS 和 CK 需要配置为输出模式。如果寄存器 SPI_I2SPR 的 MCKOE 位为 ‘1’ ，引脚 MCK 也要配置成输出模式。

发送流程

当将 1 个半字(16 位)的数据写入发送缓存区后，发送流程开始。

假设第一个写入发送缓存的数据对应的是左声道数据。当数据从发送缓存区移到移位寄存器时，标志位 TXE 置 ‘1’ ，这时，要把对应右声道的数据写入发送缓存区。标志位 CHSIDE 提示了目前待传输

的数据对应哪个声道。TXE 标志置 1 时，CHSIDE 标志有意义，因为该标志在 TXE 变为高电平时进行更新。

在先左声道后右声道的数据都传输完成后，才能被认为是一个完整的数据帧。不可以只传输部分数据帧，如仅有左声道的数据。

当发出第一位数据的同时，半字数据被并行地传送至 16 位移位寄存器，然后后面的位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓存移至移位寄存器时，标志位 TXE 置为‘1’，如果寄存器 SPI_CR2 的 TXEIE 位为‘1’，则产生中断。

写入数据的操作取决于所选择的 I2S 标准。为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。

要通过将 I2SE 清零来关闭 I2S，必须等待 TXE = 1 且 BSY = 0。

接收流程

接收流程与发送流程类似，区别在第 3 步中选择接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收。即每次填满接收缓存后，标志位 RXNE 置‘1’，如果寄存器 SPI_CR2 的 RXNEIE 位为‘1’，则产生中断。根据配置的数据和声道长度，收到左声道或右声道的数据会需要 1 次或者 2 次把数据传送到接收缓存的过程。

对寄存器 SPI_DR 进行读操作即可清除 RXNE 标志位。

每次接收以后即更新 CHSIDE。它的值取决于 I²S 单元产生的 WS 信号。读取数据的操作取决于所选择的 I²S 标准。

如果前一个接收到的数据还没有被读取，又接收到新数据，即发生上溢，标志位 OVR 被置为‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，表示发生了错误。

关闭 I²S

若要关闭 I²S 功能，需要执行特别的操作，以保证 I²S 模块可以正常地完成传输周期而不会开始新的数据传输。

具体操作流程：

若要关闭 I2S 功能，需要执行特别的操作，以保证 I2S 模块可以正常地完成传输周期而不会开始新的数据传输。操作过程与数据配置和通道长度、以及音频协议的模式相关：

- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 LSB(低位)对齐模式(I2SSTD=10)
  - 等待倒数第二个(n-1)RXNE=1；
  - 等待 17 个 I²S 时钟周期(使用软件延迟)；
  - 关闭 I²S(I2SE=0)。
- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 MSB(高位)对齐、 I²S 或 PCM 模式(分别为 I2SSTD=00，I2SSTD=01 或 I2SSTD=11)
  - 等待最后一个 RXNE=1；
  - 等待 1 个 I²S 时钟周期(使用软件延迟)；
  - 关闭 I²S(I2SE=0)。
- 所有其它 DATLEN 和 CHLEN 的组合，无论通过 I2SSTD 选择何种音频模式，使用下述方式关闭 I²S：
  - 等待倒数第二个(n-1)RXNE=1；
  - 等待一个 I²S 时钟周期(使用软件延迟)；

— 关闭 I²S(I2SE=0)。

34.4.4.2. 从模式

在从模式下，I²S 可以设置成发送和接收模式。从模式的配置方式基本遵循和配置主模式一样的流程。在从模式下，不需要 I²S 接口提供时钟。时钟信号和 WS 信号都由外部主 I²S 设备提供，连接到相应的引脚上。因此用户无需配置时钟。

配置步骤如下：

1. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位激活 I²S 功能；设置 I2SSTD[1:0]来选择所用的 I²S 标准；设置 DATLEN[1:0]选择数据的比特数；设置 CHLEN 选择每个声道的数据位数。设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I²S 从模式的数据方向。
2. 根据需要，设置寄存器 SPI_CR2 打开所需的中断功能和 DMA 功能。
3. 必须设置寄存器 SPI_I2SCFGR 的 I2SE 位为 ‘1’ 。

发送流程

当外部主机发送时钟信号，并且当 NSS_WS 信号请求传输数据时，发送流程开始。必须先使能从机，并且写入 I²S 数据寄存器之后，外部主机才能开始通信。

对于 I²S 的 MSB 对齐和 LSB 对齐模式，第一个写入数据寄存器的数据项对应左声道的数据。当通信开始时，数据从发送缓冲器传送到移位寄存器，然后标志位 TXE 置为 ‘1’ ；这时，要把对应右声道的数据项写入 I²S 数据寄存器。

标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。与主模式的发送流程相比，在从模式中，CHSIDE 取决于来自外部主 I²S 的 WS 信号。这意味着从机在接收到主机生成的时钟信号之前，就要准备好第一个要发送的数据。 WS 信号为 ‘1’ 表示先发送左声道。

注：在主器件发出的第一个时钟出现在 CK 线上时，必须至少提前 2 个 PCLK 周期置位 I2SE。

当发出第一位数据的时候，半字数据并行地通过 I²S 内部总线传输至 16 位移位寄存器，然后其它位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓冲器传送至移位寄存器时，标志位 TXE 置 ‘1’ ，如果寄存器 SPI_CR2 的 TXEIE 位为 ‘1’ ，则产生中断。注意，在对发送缓冲器写入数据前，要确认标志位 TXE 为 ‘1’ 。写入数据的操作取决于所选中的 I2S 标准，详见 “支持音频协议” 章节。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。如果在数据尚未写入 SPI_DR 寄存器时下一个数据通信的首个时钟边沿到来，下溢标志将置 ‘1’ 并可能产生中断。通过这种方式，软件可以获知所传输的数据不正确。如果 SPI_CR2 寄存器的 ERRIE 位置 ‘1’ ，则当 SPI_SR 寄存器中的 UDR 标志变为 ‘1’ 时，将产生中断。这种情况下，必须关闭 I²S 并从左通道开始重新启动数据传输。

要通过清除 I2SE 位关闭 I²S，必须先等待 TXE=1 并且 BSY=0。

接收流程

接收流程与发送流程类似，区别在 “配置流程” 的第 1 步中通过寄存器 I2SCFG[1:0]选择接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收，即每次填满接收缓存，标志位 RXNE 置 ‘1’ ，如果寄存器 SPI_CR2 的 RXNEIE 位为 ‘1’ ，则产生中断。按照不同的数据和声道长度设置，收到左声道或者右声道数据会需要 1 次或者 2 次传输数据至接收缓冲器的过程。

每次接收到数据(将要从 SPI_DR 读出)以后即更新 CHSIDE，它对应 I²S 单元产生的 WS 信号。读取 SPI_DR 寄存器，将清除 RXNE 位。读取数据的操作取决于所选中的 I²S 标准。

在还没有读出前一个接收到的数据，又接收到新数据时，即产生上溢，并设置标志位 OVR 为‘1’；如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，指示发生了错误。

要关闭 I²S 功能时，需要在接收到最后一次 RXNE=1 时将 I2SE 位清‘0’。

注：外部主机器件需要有通过音频声道发送/接收 16 位或 32 位数据包的功能。

34.4.5. I²S 标志

34.4.5.1. 状态标志

有 3 个状态标志位供用户监控 I²S 总线的状态。

忙标志位(BSY)

BSY 标志由硬件设置与清除(写入此位无效)，该标志位指示 I²S 通信层的状态。

该位为‘1’时表明 I²S 通讯正在进行中，但有一个例外：主接收模式(I2SCFG=11)下，在接收期间 BSY 标志始终为低。

在软件要关闭 I²S 模块之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输。

当传输开始时，BSY 标志被置为‘1’，除非 I²S 模块处于主接收模式。

下述情况时，该标志位被清除：

- 当传输结束时(除了主发送模式，这种模式下通信是连续的)；
- 当关闭 I2S 模块时。

当通信是连续的时候：

- 在主发送模式时，整个传输期间，BSY 标志始终为高；
- 在从模式时，每个数据项传输之间，BSY 标志在变低并持续 1 个 I2S 时钟周期内。

注：请勿使用 BSY 标志处理每次数据发送或接收，最好改用 TXE 标志和 RXNE 标志。

发送缓存空标志位(TXE)

该标志位为‘1’表示发送缓冲器为空，可以对发送缓冲器写入新的待发送数据。在发送缓冲器中已有数据时，标志位清‘0’。在 I²S 被关闭时(I2SE 位为‘0’)，该标志位也为‘0’。

接收缓存非空标志位(RXNE)

该标志位置‘1’表示在接收缓存里有接收到的有效数据。在读取寄存器 SPI_DR 时，该位清‘0’。

声道标志位(CHSIDE)

在发送模式下，该标志位在 TXE 为高时刷新，指示从 SD 引脚上发送的数据所在的声道。如果在从发送模式下发生了下溢错误，该标志位的值无效，在重新开始通讯前需要把 I²S 关闭再打开。

在接收模式下，该标志位在寄存器 SPI_DR 接收到数据时刷新，指示接收到的数据所在的声道。如果发生错误(如上溢 OVR)，该标志位无意义，需要将 I²S 关闭再打开(同时，如果必要修改 I2S 的配置)。

在 PCM 标准下，无论短帧格式还是长帧格式，这个标志位都没有意义。

如果寄存器 SPI_SR 的标志位 OVR 或 UDR 为‘1’，且寄存器 SPI_CR2 的 ERRIE 位为‘1’，则会产生中断，而后可以通过读寄存器 SPI_SR 来清除中断标志。

34.4.5.2. 错误标志

I²S 单元有 2 个错误标志位。

下溢标志位(UDR)

在从发送模式下，如果数据传输的第一个时钟边沿到达时，新的数据仍然没有写入 SPI_DR 寄存器，

该标志位会被置‘1’。在寄存器 SPI_I2SCFGR 的 I2SMOD 位置‘1’后，该标志位才有效。如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，就会产生中断。

通过对寄存器 SPI_SR 进行读操作来清除该标志位。

上溢标志位(OVR)

如果还没有读出前一个接收到的数据时，又接收到新的数据，即产生上溢，该标志位置‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断指示发生了错误。这时，接收缓存的内容，不会刷新为从发送设备送来的新数据。对寄存器 SPI_DR 的读操作返回最后一个正确接收到的数据。其他所有在上溢发生后由发送设备发出的 16 位数据都会丢失。

通过先读寄存器 SPI_SR 再读寄存器 SPI_DR，来清除该标志位。

34.4.6. I²S DMA

在 I²S 模式下，DMA 的工作方式与在 SPI 模式下完全相同。除了由于不存在数据传输保护机制，I²S 模式下没有 CRC 功能外，没有其他差别。

34.4.7. I²S 中断

I²S 中断请求如下：

表 34-3 I²S 中断

中断事件	事件标志位	使能标志位
发送缓冲器空标志位	TXE	TXEIE
接收缓冲器非空标志位	RXNE	RXNEIE
下溢标志位	OVR	ERRIE
上溢标志位	UDR	

34.5. SPI/I²S 寄存器

34.5.1. SPI 控制寄存器 1 (SPI_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	BIDIMODE	RW	0	双向数据模式使能。 0：“双线单向”模式 1：“单线双向”模式 注：该位不适用于 I²S 模式。
14	BIDIOE	RW	0	双向模式输出使能。 与 BIDIMODE 位一起配置“单线双向”模式下数据的输出方向。 0：输出禁止（只接收模式） 1：输出使能（只发送模式）

				在主模式下，使用 MOSI 引脚；在从模式下，使用 MISO 引脚。 注：该位不适用于 I²S 模式。
13	CRCEN	RW	0	硬件 CRC 校验使能。 0：禁止 CRC 计算； 1：启动 CRC 计算。 注：只有在禁止 SPI 时(SPE=0)，才能写该位，否则出错。 注：该位不适用于 I²S 模式。
12	CRCNEXT	RW	0	下一个发送 CRC。 0：下一个发送的值来自发送缓冲区。 1：下一个发送的值来自发送 CRC 寄存器。 注：在 SPI_DR 寄存器写入最后一个数据后应马上设置该位。 注：该位不适用于 I²S 模式。
11	DFF	RW	0	数据帧格式 0：使用 8 位数据帧格式进行发送/接收； 1：使用 16 位数据帧格式进行发送/接收。 注：只有当 SPI 禁止(SPE=0)时，才能写该位，否则出错。 注：该位不适用于 I²S 模式。
10	RXONLY	RW	0	仅接收控制。 该位和 BIDIMODE 位一起决定在“双线单向”模式下的传输方向。在多个从机的配置中，在未被访问的从机上该位置 1，使得只有被访问的从机才有输出，因而不会造成数据线上有数据冲突。 0：全双工（发送和接收） 1：禁止输出（只接收模式） 注：该位不适用于 I²S 模式。
9	SSM	RW	0	软件从机管理。 当 SSM 置位，NSS 引脚上的电平由 SSI 位的值决定。 0：禁止软件从机管理 1：使能软件从机管理 注：该位不适用于 I²S 模式和 SPI TI 模式。
8	SSI	RW	0	内部从机选择。 该寄存器只有当 SSM=1 时才有效。该寄存器决定了 NSS 上的电平，在 NSS 引脚上的 I/O 操作无效。 注：该位不适用于 I²S 模式和 SPI TI 模式。
7	LSBFIRST	RW	0	帧格式。 0：先发送 MSB 1：先发送 LSB 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I²S 模式和 SPI TI 模式。
6	SPE	RW	0	SPI 使能。 0：禁止 SPI 1：使能 SPI 注：该位不适用于 I²S 模式。
5:3	BR[2:0]	RW	0	波特率控制。 000：$f_{PCLK}/2$ 001：$f_{PCLK}/4$

				010：f_{PCLK}/8 011：f_{PCLK}/16 100：f_{PCLK}/32 101：f_{PCLK}/64 110：f_{PCLK}/128 111：f_{PCLK}/256 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I^2S 模式。
2	MSTR	RW	0	主机选择。 0：配置为从机 1：配置为主机 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I^2S 模式。
1	CPOL	RW	0	时钟极性。 0：空闲状态时，SCK 保持低电平 1：空闲状态时，SCK 保持高电平 通讯进行时不能改变该寄存器的值。 注：除了在 TI 模式下应用 CRC 的情况外，该位不适用于 I^2S 模式和 SPI TI 模式。
0	CPHA	RW	0	时钟相位。 0：数据采样从第一个时钟边沿开始 1：数据采样从第二个时钟边沿开始 通讯进行时不能改变该寄存器的值。 注：除了在 TI 模式下应用 CRC 的情况外，该位不适用于 I^2S 模式和 SPI TI 模式。

34.5.2. SPI 控制寄存器 2 (SPI_CR2)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res.	SSOE	TXDMAEN	RXDMAEN
RW								RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	FRF	RW	0	帧格式： 0：SPI Motorola 模式 1：SPI TI 模式 只有在禁止 SPI(SPE=0)后才能对此位进行操作
14:8	Reserved	-	-	保留
7	TXEIE	RW	0	发送缓冲区空中断使能 0：禁止 TXE 中断 1：使能 TXE 中断。TXE=1 时产生中断请求。
6	RXNEIE	RW	0	接收缓冲区非空中断使能 0：禁止 RXNE 中断

				1：使能 RXNE 中断。RXNE=1 时产生中断请求。
5	ERRIE	RW	0	错误中断使能。 0：禁止错误中断 1：使能错误中断。当 CRCERR、OVR 或 MODF 为 1 时，产生中断请求。
4	CLRTXFIFO	RW	0	清空 TXFIFO 软件置位，硬件复位 0：没作用 1：清空 TXFIFO 注：只有当 SPI 禁止(SPE=0)时，才能写该位，否则无效。
3	Reserved	-	-	保留
2	SSOE	RW	0	SS 输出使能。 0：禁止在主模式下 SS 输出，该设备可以工作在多主机模式 1：开启主模式下 SS 输出，该设备不能工作在多主机模式。 注：I^2S 模式下不使用。
1	TXDMAEN	RW	0	发送缓冲区 DMA 使能。 0：禁止发送缓冲区 DMA 1：使能发送缓冲区 DMA。当 TXE=1，则发出 DMA 请求。
0	RXDMAEN	RW	0	接收缓冲区 DMA 使能。 0：禁止接收缓冲区 DMA 1：使能接收缓冲区 DMA。当 TXE=1，则发出 DMA 请求。

34.5.3. SPI 状态寄存器 (SPI_SR)

偏移地址:0x08

复位值:0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	FTLVL [1:0]		FRLVL [1:0]		FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
			R	R	R	R	R	R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:11	FTLVL	R	0	FIFO 发送级别。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满(当 FIFO 阈值大于 1/2，即认为是满) 注：I^2S 模式下不使用。
10:9	FRLVL	R	0	FIFO 接收级别。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满 注：I^2S 模式和带 CRC 校验的 SPI 仅接收模式下不使用。
8	FRE	R	0	FRE：帧格式错误。 该标志用于 SPI TI 从模式。

				此标志由硬件置 1，在读取 SPI_SR 时由软件复位。 0：未发生帧格式错误 1：发生帧格式错误
7	BSY	R	0	忙标志。 0：SPI 不忙； 1：SPI 处于通讯，或者发送缓冲非空。
6	OVR	R	0	溢出标志。 0：无溢出错误 1：产生溢出错误 该寄存器由硬件置位，或者软件序列复位（上溢和下溢序列不同）。
5	MODF	R	0	模式故障。 0：无模式故障 1：发生模式故障 该寄存器由硬件置位，或者软件序列复位。 注：I^2S 模式下不适用。
4	CRCERR	R	0	CRC 错误标志。 0：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值匹配； 1：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值不匹配。 该位由硬件置位，由软件写 ‘0’ 而复位。 注：I^2S 模式下不适用。
3	UDR	R	0	下溢标志位。 0：未发生下溢； 1：发生下溢错误。 硬件置位，软件序列清零。 注： SPI 模式下不适用。
2	CHSIDE	R	0	声道控制。 0：发送或者接收左声道； 1：发送或者接收右声道。 注：在 SPI 和 I^2S PCM 模式下不适用。
1	TXE	R	1	发送缓冲空。 0：发送缓冲非空 1：发送缓冲为空
0	RXNE	R	0	接收缓冲非空。 0：接收缓冲非空 1：接收缓冲为空

34.5.4. SPI 数据寄存器（SPI_DR)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DR[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留

15:0	DR[15:0]	RW	0	数据寄存器。 要发送或者接收到的数据。 数据寄存器作为 RxFIFO 和 TxFIFO 的接口。当要读数据，实际访问 RxFIFO，而要写数据，实际访问 TxFIFO。 注：取决于 DFF 位（数据帧宽度选择），数据发送或者接收是 8 位或者 16 位。 对于 8 位数据帧，数据寄存器是基于右对齐的 8 位数据进行发送和接收的。当在接收模式，DR[15:8]硬件置为 0。 对于 16 位数据帧，数据寄存器是 16 位的，整个 DR[15:0]都用作发送和接收。

34.5.5. SPI CRC 多项式寄存器 (SPI_CRCPR)

偏移地址:0x10

复位值:0x0000 0007

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRCPOLY[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CRCPOLY[15:0]	RW	0x0007	CRC 多项式寄存器。 该寄存器包含了 CRC 计算时用到的多项式。 其复位值为 0x0007，根据应用可以设置其他数值。 注：在 I^2S 模式下不适用。 注：多项式值只能是奇数，不支持偶数值。

34.5.6. SPI 接收 CRC 寄存器 (SPI_RXCRC)

偏移地址:0x14

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXCRC [15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	RXCRC[15:0]	R	0	接收 CRC 寄存器 在启用 CRC 计算时，RXCRC[15:0]中包含了依据收到的字节计算的 CRC 数值。当在 SPI_CR1 的 CRCEN 位写入‘1’时，该寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的标准。

				注：当 BSY 标志为 ‘1’ 时读该寄存器，将可能读到不正确的数值。 注：在 I²S 模式下不适用。

34.5.7. SPI 发送 CRC 寄存器 (SPI_TXCRC)

偏移地址:0x18

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXCRC [15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	TXCRC[15:0]	R	0	接收 CRC 寄存器 在启用 CRC 计算时， RXCRC[15:0]中包含了依据收到的字节计算的 CRC 数值。当在 SPI_CR1 的 CRCEN 位写入 ‘1’ 时，该寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的标准。 注：当 BSY 标志为 ‘1’ 时读该寄存器，将可能读到不正确的数值。 注：在 I²S 模式下不适用。

34.5.8. SPI_I2S 配置寄存器 (SPI_I2SCFGR)

偏移地址:0x1C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG[1:0]		PCMSYNC	Res.	I2SSTD[1:0]		CKPOL	DTALEN[1:0]		CHLEN
				RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	I2SMOD	RW	0	I²S 模式选择。 0: 选择 SPI 模式； 1: 选择 I²S 模式。 注：该位只有在关闭了 SPI 或者 I²S 时才能设置。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
10	I2SE	RW	0	I²S 使能。 0：关闭 I²S； 1：I²S 使能。 注：在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
9:8	I2SCFG	RW	0	I²S 模式设置。 00: 从机发送；

				01: 从机接收； 10: 主机发送； 11: 主机接收。 注：该位只有在关闭了 I²S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
7	PCMSYNC	RW	0	PCM 帧同步。 0: 短帧同步； 1: 长帧同步。 注: 该位只在 I2SSTD = 11 (使用 PCM 标准)时有意义。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
6	Reserved	-	-	保留
5:4	I2SSTD	RW	0	I²S 标准选择。 00: I²S 飞利浦标准； 01: 高字节对齐标准 (左对齐)； 10: 低字节对齐标准(右对齐)； 11: PCM 标准。 注：为了正确操作，只有在关闭了 I²S 时才能设置该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
3	CKPOL	RW	0	无效状态时钟极性。 0: I²S 时钟无效状态为低电平； 1: I²S 时钟无效状态为高电平。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
2:1	DATLEN	RW	0	待传输数据长度。 00: 16 位数据长度； 01: 24 位数据长度； 10: 32 位数据长度； 11: 不允许。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。
0	CHLEN	RW	0	声道长度 (每个音频声道的数据位数)。 0: 16 位宽； 1: 32 位宽。 只有在 DATLEN = 00 时该位的写操作才有意义，否则声道长度都由硬件固定为 32 位。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0。

34.5.9. SPI_I2S 预分频器寄存器 (SPI_I2SPR)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV[7:0]							
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	MCKOE	RW	0	主机时钟输出使能 0: 关闭主机时钟输出; 1: 主机时钟输出使能。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.
8	ODD	RW	0	预分频器的奇数因子。 0: 实际分频系数 = I2SDIV *2; 1: 实际分频系数 = (I2SDIV * 2)+1。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.
7:0	I2SDIV	RW	0	I²S 线性预分频器。 禁止设置 I2SDIV [7:0] = 0 或者 I2SDIV [7:0] = 1。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.

35. 电压参考缓冲器(V_{REFBUF})

35.1. V_{REFBUF} 简介

本芯片内置有电压参考缓冲器 V_{REFBUF}，当使用内建 VREFBUF 时，需要通过 PA0 引脚外接 1μF 电容，此时 PA0 需要配置成模拟模式。若 PA0 复用为 GPIO，内部 V_{REFBUF} 无法使用，同时支持通过 PA0 外部输入参考电压给 ADC 或 DAC 使用，此时 PA0 也需要配置成模拟模式。

35.2. V_{REFBUF} 寄存器

35.2.1. V_{REFBUF} 控制寄存器 (VREFBUF_CR)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VREFBUF_EN	VREFBUF_OUT_SEL[2:0]		
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	VREFBUF_EN	RW	0	V_{REFBUF} 使能。 0：禁止 V_{REFBUF} 1：使能 V_{REFBUF}
2:0	VREFBUF_OUT_SEL [2；0]	RW	0	V_{REFBUF} 模块输出电压选择（不同产品该寄存器代表电压值会不同）。 000: 1.024V 001: 1.5V 010: 2.048V 011: 2.5V 100: 0.6V 其它：保留 注：各个电压对应的 Trim 值在 Flash 地址如下： 0.6V 校准值存放地址：0x1FFF 1C24 1.024V 校准值存放地址：0x1FFF 1C28 1.5V 校准值存放地址：0x1FFF 1C2C 2.048V 校准值存放地址：0x1FFF 1C30 2.5V 校准值存放地址：0x1FFF 1C34

36. FD 控制器局域网路(FDCAN)

36.1. FDCAN 简介

控制器局域网络（CAN）子系统由一个 CAN 模块、一个共享消息 RAM 存储器组成。

CAN 模块符合 ISO 11898-1: 2015（CAN 协议规范第 2.0 版 A、B 部分）和 FD CAN 协议规范第 1.0 版（CAN with Flexible Data-Rate Specification Version 1.0）。

0.8KB 的消息 RAM 存储器可实现接收过滤器（Rx Filter）、接收 FIFO（Rx FIFO）、接收缓冲区（Rx Buffer）、发送事件 FIFO（Tx Event FIFO）、发送缓冲区（Tx Buffer）功能。

36.2. FDCAN 主要特性

- 适用 ISO 11898-1:2015 标准
- FD CAN 最多支持收发 64 个数据字节
- 支持 CAN 错误日志
- 支持 AUTOSAR 和 SAE J1939
- 支持接收过滤功能
- 两个可配置的接收 FIFO
- 接收到高优先级消息时单独发出信号指示
- 最多 3 个专用接收缓冲区
- 最多 3 个专用发送缓冲区
- 可配置的发送 FIFO 或队列
- 可配置的发送事件 FIFO
- 主机 CPU 可直接访问消息 RAM
- 支持可编程回环测试模式
- 支持可屏蔽模块中断
- 两个时钟域：CAN 内核时钟和 APB 总线接口时钟
- 支持调试

36.3. FDCAN 功能描述

36.3.1. FDCAN 框图

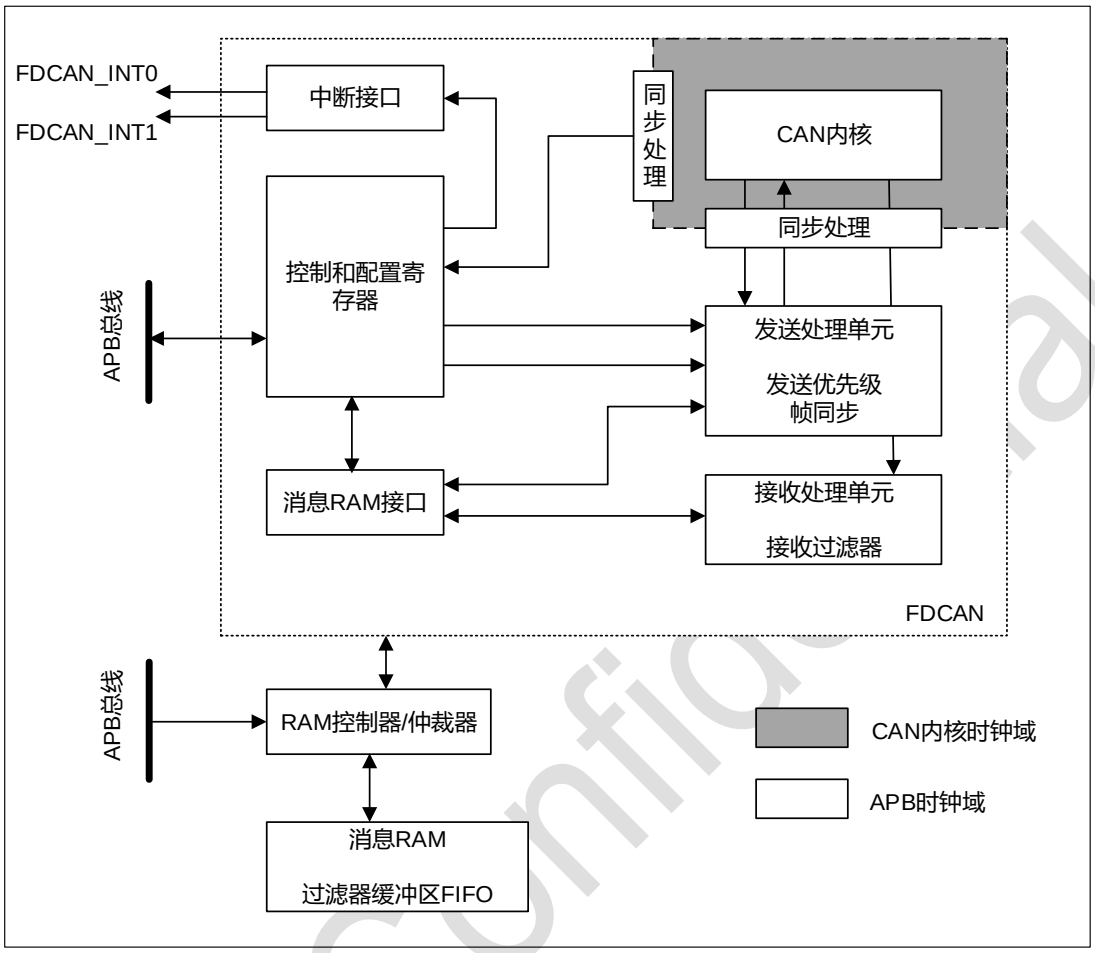


图 36-1 FDCAN 框图

CAN 内核

CAN 内核包含协议控制器、接收与发送移位寄存器。它可处理 ISO 11898-1:2015 的所有协议功能，并支持 11 位标准 ID 和 29 位扩展 ID。

同步处理

同步模块将来自 APB 时钟域的信号同步到 CAN 内核时钟域，以及将来自 CAN 内核时钟域的信号同步到 APB 时钟域。

时钟

APB 时钟域的时钟为 FDCAN 控制逻辑时钟，CAN 内核时钟域的时钟为 FDCAN 通信时钟。为了保证 FDCAN 通信的稳定性，控制逻辑时钟频率必须大于等于通信时钟频率。

发送处理单元

该处理单元控制消息从外部消息 RAM 到 CAN 内核的传输。最多可配置 32 个发送缓冲区进行发送。发送缓冲区可用作专用发送缓冲区、发送 FIFO（或队列），或二者的组合。发送事件 FIFO 将发送时间戳与对应的消息 ID 存储在一起。另外还支持取消发送。

接收处理单元

该处理单元控制接收到的消息从 CAN 内核到外部消息 RAM 的传输。接收处理单元支持两个接收 FIFO（每个 FIFO 的大小均可配置）和最多 64 个专用接收缓冲区，用于存储所有通过接收过滤的消

息。与接收 FIFO 有所不同，专用接收缓冲区仅用于存储特定 ID 的消息。接收时间戳与其消息存储在一起。对于 11 位标准 ID，最多可定义 128 个过滤器；对于 29 位扩展 ID，最多可定义 64 个过滤器。

APB 接口

将 FDCAN 连接至 APB 总线。

消息 RAM

接口通过 RAM 控制器和仲裁器将 FDCAN 访问连接到外部 0.8KB 消息 RAM。

双中断线

FDCAN 提供 FDCAN_INT0 和 FDCAN_INT1 两条中断线。对 FDCAN_ILE.EINT0 和 FDCAN_ILE.EINT1 位进行编程，可分别使能或禁止这两条中断线。

36.3.2. FDCAN 工作模式

36.3.2.1. 正常工作

FDCAN 完成初始化（初始化流程参见“37.4 FDCAN 软件初始化”章节）且 FDCAN_CCCR.INIT 清零后，FDCAN 会将其自身与 CAN 总线同步并准备好进行通信。接收消息时，通过接收过滤器接收到的消息（包含消息 ID 和 DLC）会存储到专用的接收缓冲区，或接收 FIF0，或接收 FIF1。发送消息时，先初始化或更新专用发送缓冲区和（或）发送 FIFO（队列）。不支持在接收到远程帧后自动发送。

36.3.2.2. FDCAN 操作

FDCAN 帧传输有两种类型，第一种是不带比特率切换的 FDCAN 帧，第二种是带比特率切换的 FDCAN 帧（控制段、数据段和 CRC 段的传输比特率比帧开始和结束时的高）。

经典 CAN（ClassicalCAN）标准帧中的 r0 位（IDE 位之后，标准帧格式中的保留位）、扩展帧中的 r1 位（RTR 位之后，扩展帧格式中的第一个保留位），在 FDCAN 帧格式中解码为 FDF 位。FDF 位为隐性（逻辑 1）表示 FDCAN 帧，为显性（逻辑 0）表示经典 CAN 帧。在 FDCAN 帧中，FDF 位之后的 res 和 BRS（BitRateSwitch，比特率切换）两个位，决定是否切换 FDCAN 帧中的比特率，并通过 res 显性和 BRS 隐性指示已切换比特率。res 位隐性的编码保留，供将来协议扩展使用。如果 FDCAN 接收到的帧中 FDF 位和 res 位都是隐性，则会通过将 FDCAN_PSR.PXE 置位的方式发出协议异常事件。如果使能协议异常处理（FDCAN_CCCR.PXHD=0），FDCAN 将在下一采样点将工作状态从接收（FDCAN_PSR.ACT=0b10）变为同步（FDCAN_PSR.ACT=0b00）。如果禁止协议异常处理（FDCAN_CCCR.PXHD=1），FDCAN 会将隐性 res 位当作格式错误处理，并将以错误帧进行响应。

FD CAN 模式通过编程 FDCAN_CCCR.FDOE 来使能。如果 FDCAN_CCCR.FDOE=1，则会使能 FD CAN 帧的发送和接收。经典 CAN 帧始终可进行发送和接收。是发送 FD CAN 帧还是经典 CAN 帧，可通过相应发送缓冲区元素中的 FDF 位配置。如果 FDCAN_CCCR.FDOE=0，接收到的帧会被当作经典 CAN 帧，从而会在接收到 FD CAN 帧时发送错误帧。如果 FD CAN 模式已禁止，即使发送缓冲区元素的 FDF 位置 1，也不会发送 FD CAN 帧。仅当 FDCAN_CCCR.INIT 和 FDCAN_CCCR.CCE 均置位时，才能更改 FDCAN_CCCR.FDOE 和 FDCAN_CCCR.BRSE。

如果 FDCAN_CCCR.FDOE=0，则会忽略 FDF 和 BRS 位的设置，帧以经典型 CAN 格式发送。如果 FDCAN_CCCR.FDOE=1 且 FDCAN_CCCR.BRSE=0，则仅会评估发送缓冲区元素的 FDF 位。如果 FDCAN_CCCR.FDOE=1 且 FDCAN_CCCR.BRSE=1，则会使能比特率切换发送 FD CAN 帧。所有 FDF 和 BRS 位置位的发送缓冲区元素均会以比特率切换的 FD CAN 格式发送。

建议仅当以下条件满足时，才在 CAN 操作期间切换模式：

- FD CAN 数据阶段的故障率显著高于 FD CAN 仲裁阶段的故障率。在这种情况下，应禁止发送的 FD CAN 比特率切换。
- 系统启动期间，所有节点在被验证能够以 FD CAN 格式进行通信之前，都将发送经典 CAN 消息。如果通过验证，所有节点会切换为 FD CAN 操作。
- CAN 局部网络（CAN Partial Networking）中的唤醒消息必须以经典 CAN 格式发送。
- EOL 编程（End-of-line programming）时，并非所有节点都支持 FD CAN，非 FD CAN 的节点会保持在静默模式直到编程结束。随后，所有节点切换回经典 CAN 通信。

在 FD CAN 格式中， DLC 代码 0 到 8 的编码（对应的数据字节长度）与经典 CAN 相同，代码 9 到 15 的编码与经典 CAN 不同，如下表所示

表 36-1 FDCAN 中的 DLC 编码

DLC 编码		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
数据字节数	经典 CAN	0	1	2	3	4	5	6	7	8	8	8	8	8	8	8	8
	FD CAN	0	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64

在 FD CAN 帧中，如果 BRS 位为隐性，则位时长将在 BRS（比特率切换）位后，在帧内切换。在 BRS 位之前，在 FD CAN 仲裁段，位时长按照寄存器 FDCAN_NBTP（Nominal Bit Timing & Prescaler Register，标称位时长和预分频寄存器）的定义执行。在接下来的 FD CAN 数据段，位时长按照寄存器 FDCAN_DBTP（Data Bit Timing & Prescaler Register，数据位时长和预分频寄存器）的定义执行。位时长在 CRC 分隔符之后，或检测到错误时从数据段位时长切换回来，以先发生的事件为准。

FD CAN 数据段的最大可配置比特率取决于 CAN 通信时钟频率。举例来说，如果 CAN 通信时钟频率为 20MHz，最短可配置的位时间为 4 个时间片（tq），则数据阶段的比特率为 5Mbps。

在带比特率切换和不带比特率切换的 FD CAN 两种数据帧格式中，位 ESI（Error Status Indicator，错误状态指示符）的值，由发送开始时发送节点的错误状态决定。如果发送节点为错误被动状态，则 ESI 以隐性位发送，否则以显性位发送。

36.3.2.3. 收发器延迟补偿

在 FD CAN 传输的数据段，只有一个节点进行发送，其他所有节点都是接收节点。总线的长度没有影响。当通过引脚 FDCAN_TX 发送时， FDCAN 通过引脚 FDCAN_RX 从其本地 CAN 收发器接收发送的数据。接收数据因发送器延迟而延迟，如果该延迟大于 TSEG1（采样点之前的时间段），则会检测到位错误。为了使数据段位时间可以比发送器延迟更短，引入了延迟补偿。如果没有发送器延迟补偿，FD CAN 帧数据段的比特率将受到发送器延迟的限制。

FDCAN 的协议单元实现了延迟补偿机制来补偿发送器延迟，从而实现在 FD CAN 数据段以更高的比特率进行传输，而不受特定 CAN 收发器延迟的影响。

为了检查发送节点的数据段是否存在位错误，会将延迟的发送数据与第二采样点 SSP 处的接收数据进行比较。如果检测到位错误，发送节点将在下一个常规采样点对该位错误作出反应。在仲裁段，延迟补偿始终被禁用。

发送器延迟补偿支持数据位时间短于发射器延迟的配置， ISO 11898-1:2015 对此进行了详细描述，它通过置位 FDCAN_DBTP.TDC 使能。

在第二采样点 SSP 将接收到的位与发送的位进行比较。 SSP 的位置定义为，从 FDCAN 发送输出引脚 FDCAN_TX 通过收发器到接收输入引脚 FDCAN_RX 测得的延迟，与由 FDCAN_TDCR.TDCO 配置的发送器延迟补偿偏移之和。发送器延迟补偿偏移，用于调整 SSP 在已接收到的位中的位置（例如数据段的位时间的一半）。第二采样点的位置向下取整到下一个整数 mtq（Minimum time quantum，即最小时间片，为一个 FDCAN 通信时钟周期）。

FDCAN_PSR.TDCV 显示实际的发送器延迟补偿值。当 FDCAN_CCCR.INIT 置位时，FDCAN_PSR.TDCV 被清零; 当 FDCAN_DBTP.TDC 置位时，FDCAN_PSR.TDCV 在每次发送 FD 帧时都会更新。

对于在 FDCAN 中实现的发送器延迟补偿，必须考虑以下边界条件:

- 测得的 FDCAN_TX 到 FDCAN_RX 的延迟与配置的发送器延迟补偿偏移 FDCAN_TDCR.TDCO 之和，必须小于数据段的 6 个位时间。
- 测得的 FDCAN_TX 到 FDCAN_RX 的延迟与配置的发送器延迟补偿偏移 FDCAN_TDCR.TDCO 之和，必须小于或等于 127mtq。如果此和大于 127mtq，则发送器延迟补偿使用最大值 127mtq。
- 数据段在 CRC 分隔符的采样点结束，将停止在 SSP 检查接收位。

如果通过编程 FDCAN_DBTP.TDC=1 使能发送器延迟补偿，则会在每个 FD CAN 帧的 FDF 位到 res 位的下降沿处开始测量。当在发送节点的 FDCAN_RX 引脚上检测到该边沿时，停止测量。该测量的分辨率为一个 mtq。

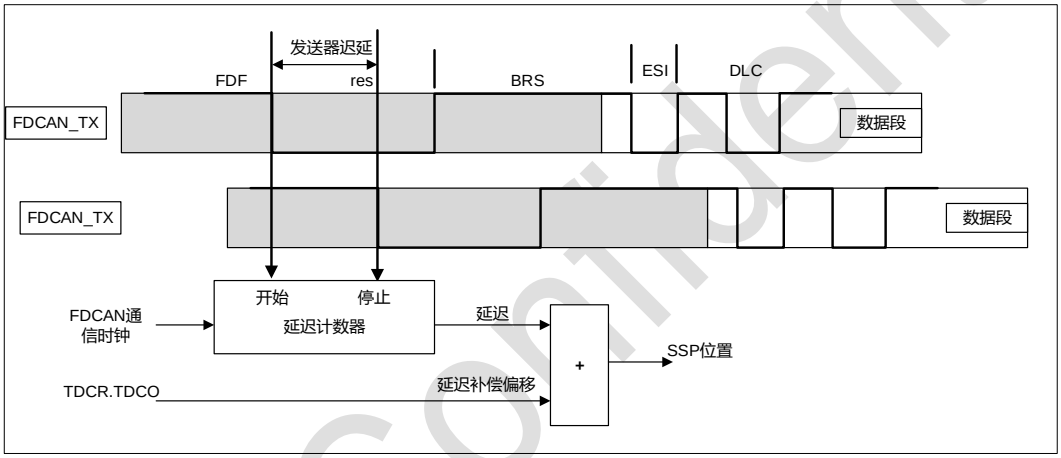


图 36-2 FDCAN 收发器延迟测量

为了避免接收到的 FDF 位中的显性毛刺，在接收到的 res 位的下降沿之前结束延迟补偿测量，从而导致 SSP 位置提前，可通过编程 FDCAN_TDCR.TDCF 来使能发送器延迟补偿滤波窗口，从而定义 SSP 位置的最小值。 FDCAN_RX 引脚上的显性边缘会导致 SSP 位置更加提前，但对于发射器延迟测量而言这将被忽略。当 SSP 位置至少为 FDCAN_TDCR.TDCF 且 FDCAN_RX 引脚为低电平时，测量停止。

36.3.2.4. 受限工作模式

在受限工作模式下，节点能够接收数据帧和远程帧，并能对有效帧进行确认，但不会发送数据帧、远程帧、主动错误帧或过载帧。在出现错误或过载情况时，不会发送显性位，而是等待出现总线空闲条件，以重新同步到 CAN 通信。错误记录(FDCAN_TDCR.CEL)动作时，错误计数器(FDCAN_TDCR.REC、FDCAN_TDCR.TEC)被冻结。主机可通过置位 FDCAN_CCCR.ASM 将 FDCAN 设置为受限工作模式，仅当 FDCAN_CCCR.CCE 和 FDCAN_CCCR.INIT 均设为 1 时，主机才能将 FDCAN_CCCR.ASM 置位。FDCAN_CCCR.ASM 可随时通过主机清零。

当发送处理单元无法及时从消息 RAM 中读取数据时，会自动进入受限工作模式。要退出受限工作模式，主机 CPU 必须清零 FDCAN_CCCR.ASM。

受限工作模式可用于适应不同 CAN 比特率的应用。在这种情况下，应用程序会测试不同的比特率，并在收到有效帧后退出受限操作模式。

注：受限工作模式不得与回环模式（内部或外部）结合使用。

36.3.2.5. 总线监听模式

配置 FDCAN_CCCR.MON 为 1，将 FDCAN 设置为总线监听模式。在总线监听模式下（见 ISO 11898-1:2015,10.14 Bus monitoring）， FDCAN 能够接收有效数据帧和有效远程帧，但不能启动发送。在该模式下，FDCAN 仅在 CAN 总线上发送隐性位，如果 FDCAN 需要发送一个显性位（ACK 位、过载标志、主动错误标志），该位将在内部被改道发送，以便 FDCAN 可以监听该显性位，但 CAN 总线可以保持隐性状态。在总线监听模式下，寄存器 FDCAN_TXBRP 保持复位状态。

总线监听模式可用于分析 CAN 总线上的流量，同时又不会因发送显性位而对总线造成影响。 下图显示了总线监听模式下 FDCAN_TX 和 FDCAN_RX 信号与 FDCAN 的连接。

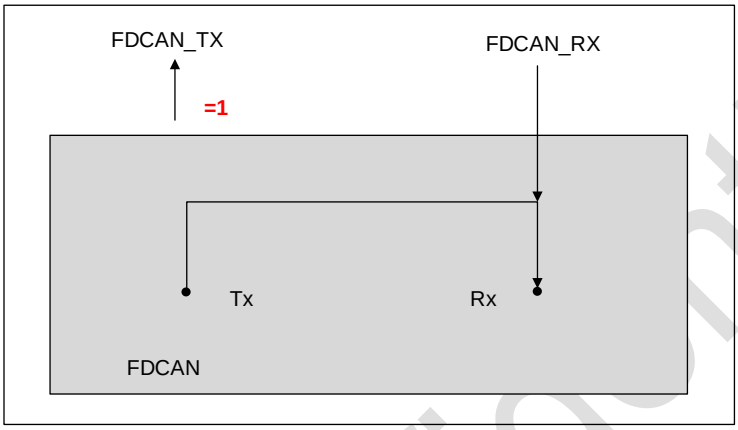


图 36-3 FDCAN 总线监听模式下的引脚控制

36.3.2.6. 禁止自动重发（DAR）模式

根据 CAN 规范（见 ISO 11898-1:2015, 8.3.4 Recovery Management）， FDCAN 可自动重发仲裁失败或在发送期间被错误干扰的帧。默认使能自动重发。

在 DAR 模式下，所有发送在 CAN 总线上启动后都会自动取消。发送缓冲区发送请求挂起位 FDCAN_TXBRP.TRPx 在成功发送后复位，如果取消时发送尚未开始、发送已因仲裁失败而中止、或帧发送期间发生错误，该位也会复位。

- 成功发送：
  - 对应的发送缓冲区发送已发生位 FDCAN_TXBTO.TOx 置位
  - 对应的发送缓冲区取消已完成位 FDCAN_TXBCF.CFx 不置位
- 即使已取消但也成功发送：
  - 对应的发送缓冲区发送已发生位 FDCAN_TXBTO.TOx 置位
  - 对应的发送缓冲区取消已完成位 FDCAN_TXBCF.CFx 置位
- 仲裁失败或帧发送受到干扰：
  - 对应的发送缓冲区发送已发生位 FDCAN_TXBTO.TOx 不置位
  - 对应的发送缓冲区取消已完成位 FDCAN_TXBCF.CFx 置位

如果帧发送成功，并且使能了发送事件的存储，则会写入发送事件 FIFO 元素，其中事件类型 ET=0b10（即使已取消也会进行发送）。

36.3.2.7. 掉电（休眠模式）

FDCAN 可通过 CC 控制寄存器 FDCAN_CCCR.CSR 设置为掉电模式。只要时钟停止请求有效，FDCAN_CCCR.CSR 位就读为 1。

当所有挂起的发送请求都完成后， FDCAN 会一直等待直到检测到总线空闲状态。检测到总线空闲状态后， FDCAN 将 FDCAN_CCCR.INIT 设置为 1 以防止任何进一步的 CAN 传输。然后， FDCAN

通过将 FDCAN_CCCR.CSA 置 1 确认其已准备好进入掉电状态。在模块时钟（控制逻辑时钟和通信时钟）关闭之前，可继续访问寄存器，但 FDCAN_CCCR.INIT 保持为 1。

注：在 CAN 总线受到严重干扰的情况下，可能永远不会达到空闲状态，因此 FDCAN 不会置位 FDCAN_CCCR.INIT。这种情况可通过轮询 FDCAN_PSR.ACT 来检测。如果 FDCAN 没有进入空闲状态，软件可以写 CCCR.INIT=1，这将立即停止 FDCAN 的 CAN 通信，无论是否正在进行发送或接收。

要退出掉电模式，应用程序必须在复位 CC 控制寄存器标志 FDCAN_CCCR.CSR 之前开启模块时钟。FDCAN 将通过复位 FDCAN_CCCR.CSA 确认时钟已开启。之后，应用程序可通过复位 FDCAN_CCCR.INIT 重新开始 CAN 通信。

36.3.2.8. 测试模式

要使能对 FDCAN 测试寄存器 FDCAN_TEST (请参见章节[FDCAN 测试寄存器 (FDCAN_TEST)]) 的写访问，必须将 FDCAN_CCCR.TEST 置 1，从而能够配置测试模式和测试功能。

通过编程 FDCAN_TEST.TX，发送引脚 FDCAN_TX 可实现四种输出功能。除了串行数据输出这一默认功能之外，还可驱动 CAN 采样点信号来监测 FDCAN 的位时长，并可输出恒定的显性或隐性电平。从 FDCAN_TEST.RX 可读取引脚 FDCAN_RX 的实际值。这两种功能（FDCAN_TEST.TX 和 FDCAN_TEST.RX）均可用于检查 CAN 总线的物理层。

由于 CAN 通信时钟与控制逻辑时钟之间的同步机制，从配置 FDCAN_TEST.TX 到引脚 FDCAN_TX 实现指定功能之间，可能有几个控制逻辑时钟周期的延迟。这一点还适用于通过 FDCAN_TEST.RX 读取引脚 FDCAN_RX 的情况。

注：

测试模式仅限用于生产测试和自检。对 FDCAN_TX 引脚进行软件控制会干扰所有 CAN 协议功能。不建议实际应用时使用测试模式。

36.3.2.9. 外部回环模式

配置 FDCAN_TEST.LBCK 为 1， 将 FDCAN 设置为外部回环模式。在回环模式下， FDCAN 将其自身发送的消息作为接收的消息来处理，并将消息存储（如果这些消息通过了接收过滤）到接收缓冲区或接收 FIFO 中。下图显示了外部回环模式下 FDCAN_TX 和 FDCAN_RX 引脚信号与 FDCAN 的连接。

该模式用于硬件自检。为了不受外部影响， FDCAN 在回环模式下忽略 ACK 错误（在数据帧或远程帧的 ACK 域采样到隐性位）。在此模式下，FDCAN_TX 输出的信号在内部直接反馈到接收处理单元，FDCAN_RX 输入引脚的实际值被忽略。从引脚 FDCAN_TX 可以监听到发送的消息。

36.3.2.10. 内部回环模式

配置 FDCAN_TEST.LBCK 和 FDCAN_CCCR.MON 为 1，将 FDCAN 设置为内部环回模式。该模式可用于“热自检（Hot Selftest） ”，也就是说， FDCAN 可以进行检测，同时又不会影响接入的正在运行的 CAN 系统。在此模式下，引脚 FDCAN_RX 与 FDCAN 断开连接，引脚 FDCAN_TX 保持隐性。 下图显示了内部环回模式下 FDCAN_TX 和 FDCAN_RX 引脚信号与 FDCAN 的连接。

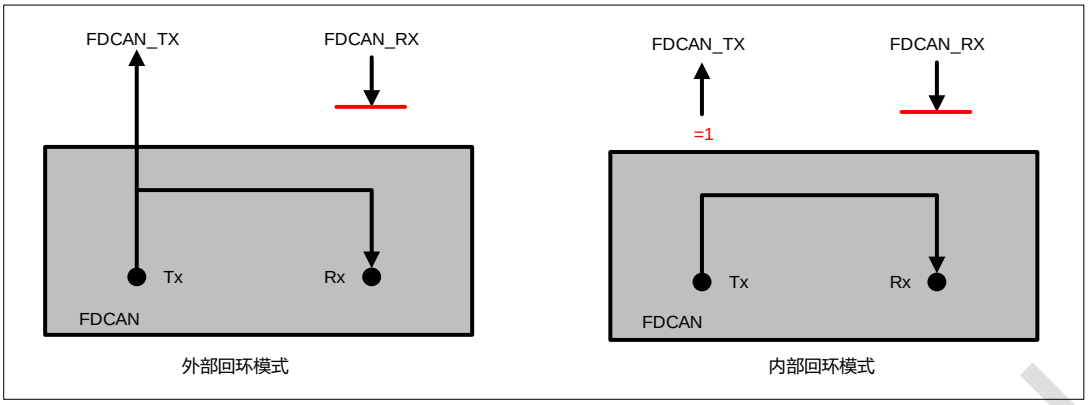


图 36-4 FDCAN 回环模式的引脚控制

36.3.3. FDCAN 时间戳生成

FDCAN 提供了一个 16 位循环计数器来生成内部时间戳。计数时间单位可通过预分频器 FDCAN_TSCC.TCP 配置为 CAN 位时间 1~16 倍。计数器值可通过 FDCAN_TSCV.TCV 读取。对寄存器 FDCAN_TSCV 进行写访问会将计数器复位为 0。时间戳计数器循环时，中断标志 FDCAN_IR.TSW 会置位。

在接收到或发送 SOF（Start Of Frame，帧起始）时，计数值被捕获并存储到接收缓冲区或接收 FIFO（RXTS[15:0]）或发送事件 FIFO（TXTS[15:0]）元素的时间戳部分。

36.3.4. FDCAN 超时计数器

FDCAN 提供了一个 16 位超时计数器来指示接收 FIF0、接收 FIF1 和发送事件 FIFO 的超时条件。它作为递减计数器工作，并且与时间戳计数器一样，使用 FDCAN_TSCC.TCP 控制的预分频器。超时计数器通过寄存器 FDCAN_TOCC 配置。实际计数器值可从 FDCAN_TOCV.TOC 读取。仅当 FDCAN_CCCR.INIT=0 时，才会启动超时计数器。当 FDCAN_CCCR.INIT=1 时，超时计数器停止计数，例如当 CAN 进入 Bus_Off 状态时。

工作模式通过 FDCAN_TOCC.TOS 选择。在连续模式下工作时，计数器在 FDCAN_CCCR.INIT 复位时启动。对 FDCAN_TOCV 进行写操作，会将计数器预设为 FDCAN_TOCC.TOP 配置的值，并继续递减计数。

当超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为 FDCAN_TOCC.TOP 配置的值。存储了第一个 FIFO 元素后，开始递减计数。对 FDCAN_TOCV 执行写入操作不起作用。

计数器达到 0 时，中断标志 FDCAN_IR.TOO 置位。在连续模式下，计数器在达到 FDCAN_TOCC.TOP 后立即重启。

注：超时计数器的时钟信号来自 CAN 内核采样点信号。因此，由于 CAN 内核同步（重同步）机制的原因，超时计数器递减的时间点可能有所不同。如果在 FD CAN 中使用了比特率切换功能，则超时计数器在仲裁字段和数据字段中使用不同的时钟计数。

36.3.5. FDCAN 接收处理

接收处理单元控制着接收过滤、已接收消息到接收缓冲区或其中一个接收 FIFO（共两个）的传输以及接收 FIFO 放入和获取索引。

36.3.5.1. 接收过滤

FDCAN 能够配置两组接收过滤器，一组供标准 ID 使用，另一组供扩展 ID 使用。这两组过滤器可分配给接收缓冲区、接收 FIFO0 或接收 FIFO1。对于接收过滤，每个过滤器列表从第 0 个元素开始执行比较，直到第一个匹配的元素。接收过滤在第一个匹配的元素处停止。不会为此消息评估剩下的过滤

器元素。

- 主要特性有：
  - 每个过滤器元素可配置为
  * ID 范围过滤器（起止范围）
  * 一个或两个专用 ID 的过滤器
  * ID 位屏蔽的经典过滤器
  - 每个过滤器元素均可配置为实现接收过滤或拒绝过滤
  - 每个过滤器元素均可单独使能/禁止
  - 按顺序比较过滤器元素，到第一个匹配的过滤器元素后停止执行
- 相关配置寄存器位：
  - 全局过滤器配置（FDCAN_GFC）
  - 标准 ID 过滤器配置（FDCAN_SIDFC）
  - 扩展 ID 过滤器配置（FDCAN_XIDFC）
  - 扩展 ID 与运算掩码（FDCAN_XIDAM）
- 根据过滤器元素的配置（SFEC/EFEC），发生匹配时会触发下列操作之一：
  - 将接收到的帧存储到接收 FIFO0 或 FIFO1 中
  - 将接收到的帧存储到接收缓冲区中
  - 拒绝接收到的帧
  - 将高优先级消息中断标志 FDCAN_IR.HPM 置 1

将高优先级消息中断标志 FDCAN_IR.HPM 置 1，并将接收到的帧存储到接收 FIF0 或 FIF1 中接收到完整的 ID 后，开始接收过滤。接收过滤完成后，如果找到匹配的接收缓冲区或接收 FIFO，消息处理单元便会开始将接收到的消息数据以 32 位的形式写入匹配的接收缓冲区或接收 FIFO 中。如果 CAN 协议控制器检测到错误条件（例如 CRC 错误），则会丢弃此消息，丢弃后的影响如下：

接收缓冲区

匹配接收缓冲区的新数据标志不会置 1，但接收缓冲区 (部分) 会被接收到的数据覆盖。有关错误类型，请参见 FDCAN_PSR.LEC 和 FDCAN_PSR.DLEC。

接收 FIFO

匹配接收 FIFO 的放入索引不会更新，但相关接收 FIFO 元素（部分）会被接收到的数据覆盖。有关错误类型，请参见 FDCAN_PSR.LEC 和 FDCAN_PSR.DLEC。如果匹配的接收 FIFO 在覆盖模式下工作，必须考虑接收 FIFO 覆盖模式中介绍的边界条件。

注: 当接收的消息写入两个接收 FIFO 之一或写入接收缓冲区中时，未匹配的已接收 ID 会独立于所用过滤器之外进行存储。接收过滤的过程很大程度上取决于所配置的过滤器元素的顺序。

36.3.5.2. ID 范围过滤器

该过滤器会将所有接收到的帧与 SF1ID/SF2ID 和 EF1ID/EF2ID 定义的范围内的消息 ID 进行匹配检查。范围过滤与扩展帧一起使用时，有两种可能的情况：

- EFT=0b00：应用范围过滤器之前，会将已接收帧的消息 ID 与扩展 ID 与运算掩码（FDCAN_XIDAM）进行与运算
- EFT=0b11：不会为范围过滤使用扩展 ID 与运算掩码（FDCAN_XIDAM）

36.3.5.3. 专用 ID 过滤器

可将过滤器元素配置为过滤一个或两个特定的消息 ID。要过滤一个特定的消息 ID，过滤器元素必须配

置为 SF1ID=SF2ID 和 EF1ID=EF2ID。

36.3.5.4. ID 位屏蔽过滤器

ID 位屏蔽过滤用于通过屏蔽已接收消息 ID 的某些位来过滤消息 ID 组。进行 ID 位屏蔽过滤时，SF1ID/EF1ID 用作消息 ID 过滤器， SF2ID/EF2ID 用作过滤器 ID 掩码（ID Mask）。

ID 掩码中为 0 的位将屏蔽掉已配置 ID 过滤器的对应位，即，已接收消息 ID 在该位位置的值不与接收过滤器对应位的值进行比较，只有对应掩码位为 1 的已接收消息 ID 的位才进行比较。

如果所有掩码位均为 1，则仅当接收到的消息 ID 和消息 ID 过滤器完全相同时，才会出现匹配。如果所有掩码位均为 0，则所有消息 ID 都匹配。

36.3.5.5. 标准消息 ID 过滤

下图介绍了标准消息 ID（11 位 ID）的过滤流程。章节【标准消息 ID 过滤器元素】介绍了标准消息 ID 过滤器元素。

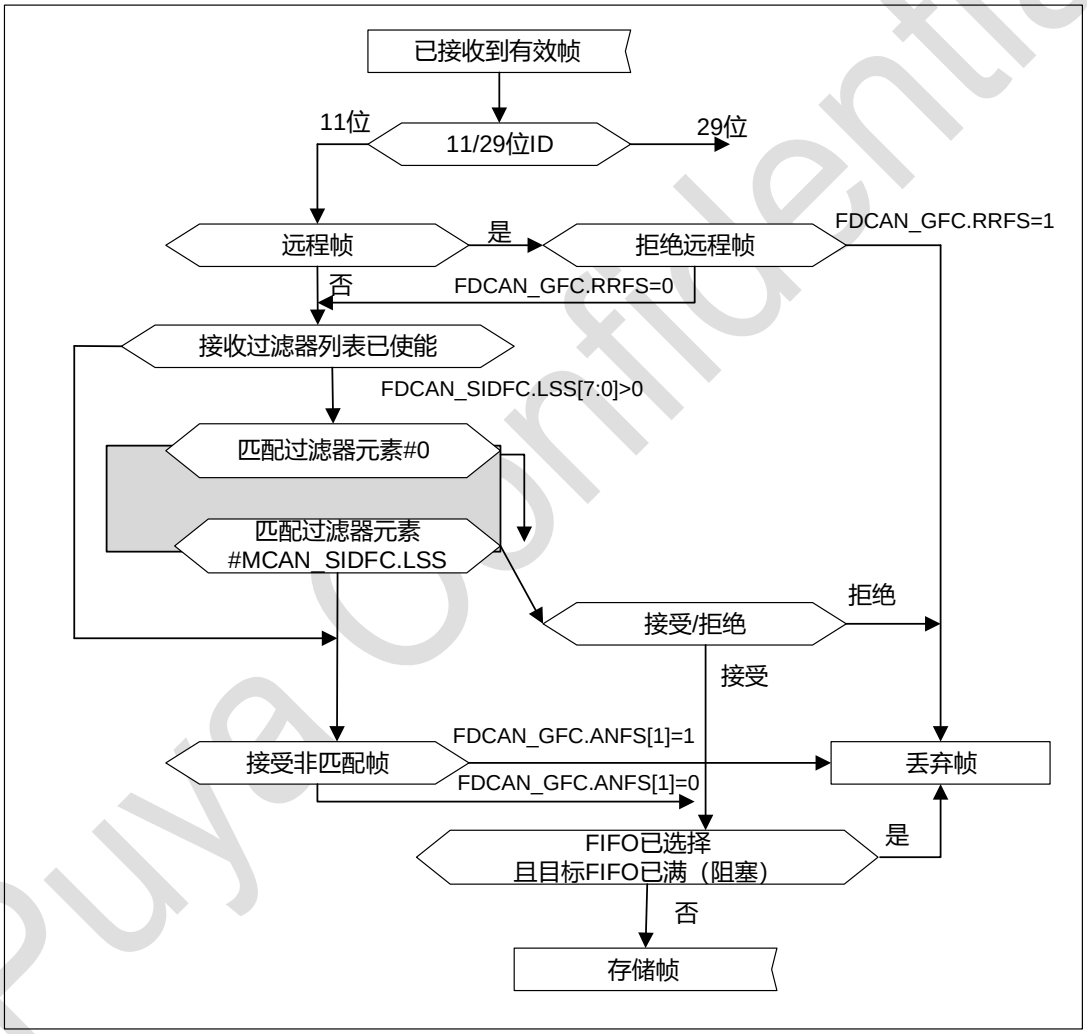


图 36-5 FDCAN 标准消息 ID 过滤器路径

在全局过滤器配置寄存器（FDCAN_GFC）和标准 ID 过滤器配置寄存器（FDCAN_SIDFC）控制下，接收消息的 ID、远程传输请求位（RTR）和 ID 扩展位（IDE）将与配置的过滤器元素列表进行比较。

36.3.5.6. 扩展消息 ID 过滤

下图介绍了扩展消息 ID（29 位 ID）的过滤流程。章节【扩展消息 ID 过滤器元素】介绍了扩展消息 ID 过滤器元素。

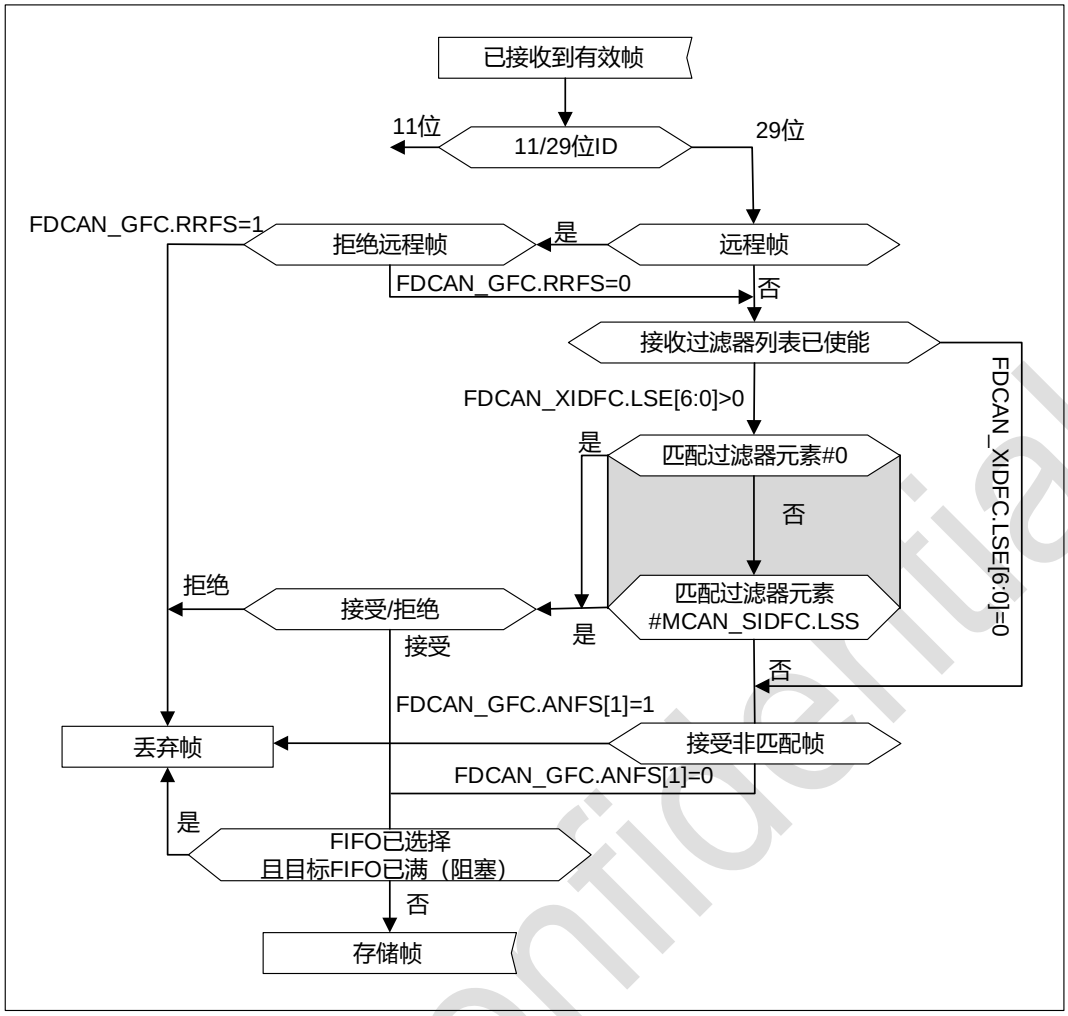


图 36-6 FDCAN 扩展消息 ID 过滤器路径

在全局过滤器配置寄存器（FDCAN_GFC）和扩展 ID 过滤器配置寄存器（FDCAN_XIDFC）控制下，接收消息的 ID、远程传输请求位（RTR）和 ID 扩展位（IDE）将与配置的过滤器元素列表进行比较。扩展 ID 与掩码（FDCAN_XIDAM）会在执行过滤器列表之前与接收到的 ID 进行与运算。

36.3.5.7. 接收 FIFO

接收 FIFO0 和接收 FIFO1 分别最多可保存 64 个元素（实际元素个数与消息 RAM 容量有关）。两个接收 FIFO 的配置通过寄存器 FDCAN_RXF0C 和 FDCAN_RXF1C 完成。

通过接收过滤的消息会传输到匹配过滤器元素配置的接收 FIFO 中。有关可用于接收 FIFO0 和接收 FIFO1 的过滤器机制的说明，请参见章节【接收过滤】 。章节【接收缓冲区和 FIFO 元素】介绍了接收缓冲区和 FIFO 元素。

为了避免接收 FIFO 溢出，可使用接收 FIFO 水位线（Watermark）。当接收 FIFO 填充级别达到 FDCAN_RXFnC.FnWM 配置的接收 FIFO 水位线时，中断标志 FDCAN_IR.RFnW 置位。当接收 FIFO 的放入索引和获取索引相遇时，接收 FIFO 满的条件产生，标志位 FDCAN_RXFnS.FnF 置位，中断标志位 FDCAN_IR.RFnF 置位。

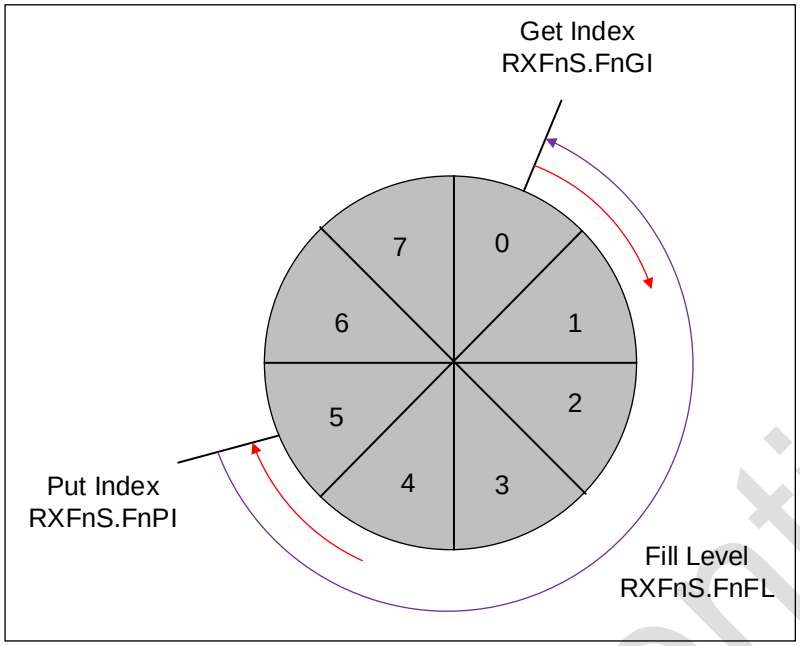


图 36-7 FDCAN 接收 FIFO 状态

当从接收 FIFO 中读取消息时，接收 FIFO 获取索引 FDCAN_RXFnS.FnGI 乘以 FIFO 元素大小后，必须再加上起始地址 FDCAN_RXFnC.FnSA。

表 36-2 FDCAN 接收缓冲器和 FIFO 元素大小

RXESC.RBDS[2:0] RXESC.FnDS[2:0]	数据段（字节）	元素大小（字）
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

36.3.5.8. 接收 FIFO 阻塞模式

接收 FIFO 阻塞模式是通过 FDCAN_RXFnC.FnOM=0 配置的，该模式是接收 FIFO 的默认工作模式。

达到接收 FIFO 已满条件时(FDCAN_RXFnS.FnPI=FDCAN_RXFnS.FnGI)，标志位 RXFnS.FnF=1，中断标志 FDCAN_IR.RFnF=1。此时， FDCAN 不会继续向相应的接收 FIFO 写入消息，除非从相应的接收 FIFO 读取消息且其获取索引已递增。

如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，并会通过 FDCAN_RXFnS.RFnL=1 指示消息丢失。此外，中断标志 FDCAN_IR.RFnL 也会置位。

36.3.5.9. 接收 FIFO 覆盖模式

接收 FIFO 覆盖模式是通过 FDCAN_RXFnC.FnOM=1 配置的。

达到接收 FIFO 已满条件时(FDCAN_RXFnS.FnPI=FDCAN_RXFnS.FnGI)，标志位 RXFnS.FnF=1。此时，FIFO 接收的下一条消息将覆盖最早收到的消息，放入索引和获取索引都会加 1。

如果接收 FIFO 在覆盖模式下工作，并且接收 FIFO 已满，则应至少从获取索引加 1 处开始读取接收

FIFO 元素。这是因为，当 CPU 正在从消息 RAM 读取消息时，恰好有接收到的消息正在被写入消息 RAM，这可能导致 CPU 读到的数据与实际不一致，从接收 FIFO 读取数据时向获取索引中添加偏移量可避免此问题。偏移量取决于 CPU 访问接收 FIFO 的速度。 下图显示了读取接收 FIFO 时获取索引的偏移量为 2。在这种情况下，存储在元素 1 和 2 中的两条消息会丢失。

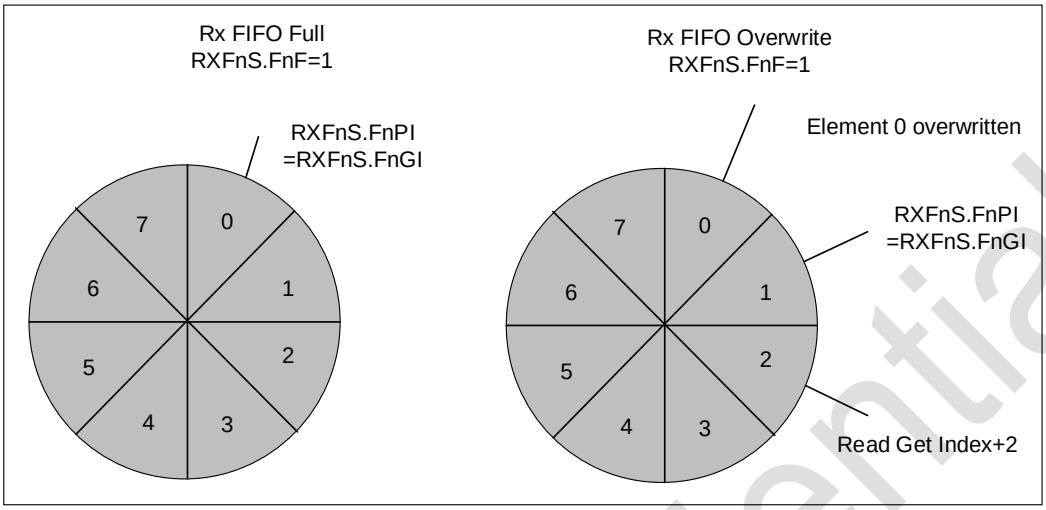


图 36-8 FDCAN 接收 FIFO 溢出处理

从接收 FIFO 读取数据后，必须将读取的最后一个元素的编号写入接收 FIFO 确认索引寄存器 FDCAN_RXFnA.FnA 中，使增加后的获取索引指向下一个元素。如果放入索引未增加到该接收 FIFO 元素编号，接收 FIFO 已满标志会复位（FDCAN_RXFnS.FnF=0）。

36.3.5.10. 专用接收缓冲区

FDCAN 支持多达 64 个专用接收缓冲区。专用接收缓冲区的起始地址是通过 RXBC.RBSA 配置的。

要使用专用接收缓冲区，对于标准 ID 消息，必须配置标准消息 ID 过滤器元素的 SFEC 位为 0b111、SFID2 位为 0b00（请参见章节【标准消息 ID 过滤器元素】）；对于扩展 ID 消息，必须配置扩展消息 ID 过滤器元素的 EFEC 位为 0b111、 EFID2 位为 0b00（请参见章节【扩展消息 ID 过滤器元素】 ）。 下表为专用接收缓冲区的过滤器配置示例。

接收到的消息被过滤器元素接受后，会存储在过滤器元素引用的消息 RAM 中的接收缓冲区中。存储格式与接收 FIFO 元素的格式相同。此外，中断寄存器中的标志 FDCAN_IR.DRX（专用接收缓冲区中存储的消息）也会置 1。

表 36-3 FDCAN 接收缓冲区过滤器配置示例

过滤器元素	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID 消息 1	00	00 0000
1	ID 消息 2	00	00 0001
2	ID 消息 3	00	00 0010

匹配的接收到的消息最后一个字写入消息 RAM 后，寄存器 FDCAN_NDAT1/FDCAN_NDAT2 中对应的新数据标志会置位。只要新数据标志保持置位的状态，相应的接收缓冲区就会被锁定，以免更新为新的匹配消息。新数据标志必须由主机向相应的位写 1 进行复位。

接收缓冲区新数据标志置位时，其对应的消息 ID 过滤器元素将不进行接收过滤，其它的过滤器元素会继续进行接收过滤。其它的消息 ID 过滤器元素可能导致接收到的消息存储到另一接收缓冲区或接收 FIFO 中，或者被拒绝，具体取决于过滤器配置。

36.3.5.11. 接收缓冲区处理

- 复位中断标志 FDCAN_IR.DRX
- 读取新数据寄存器
- 从消息 RAM 中读取消息
- 复位已处理消息的新数据标志

36.3.6. FDCAN 发送处理

发送处理单元处理专用发送缓冲区、发送 FIFO 和发送队列的发送请求。它控制着发送消息到 CAN 内核的传输、放入和获取索引以及发送事件 FIFO。最多可为消息发送设置 32 个发送缓冲区，每个发送缓冲区可以独立配置发送模式（经典 CAN 模式或 FD CAN 模式）。发送缓冲区的描述见章节【发送缓冲区元素】。 下表描述了发送帧的可能配置。

表 36-4 FDCAN 发送帧的可能配置

FDCAN_CCCR		发送缓冲区元素		发送帧
BRSE	FDOE	FDF	BRS	
忽略	0	忽略	忽略	经典 CAN
0	1	0	忽略	经典 CAN
0	1	1	忽略	不切换比特率的 FD
1	1	0	忽略	经典 CAN
1	1	1	0	不切换比特率的 FD
1	1	1	1	切换比特率的 FD

注：AUTOSAR 至少需要三个发送队列缓冲区，并支持发送取消。

当发送缓冲区请求挂起寄存器 FDCAN_TXBRP 更新时，发送处理单元会启动发送扫描来检查优先级最高的挂起的发送请求（消息 ID 最小的发送缓冲区）。 FDCAN_TXBRP 更新的条件是，一个发送缓冲区发送完成，或主机通过写 FDCAN_TXBAR 添加发送请求，或主机通过写 FDCAN_TXBCR 取消已挂起的发送。

36.3.6.1. 发送暂停

发送暂停功能用于 CAN 消息 ID（永久性地）被指定为特定值、并且无法轻易更改的 CAN 系统中。这些消息 ID 的 CAN 仲裁优先级可能高于其他已定义的消息，而在特定的应用中，其相对仲裁优先级应反转。这样可能会导致如下情况: 一个 ECU 突发多条较高仲裁优先级的 CAN 消息造成另一个仲裁优先级较低的 ECU CAN 消息被延迟。

举例来说，如果 CAN ECU-1 启用了此功能，并且其应用软件请求发送四条消息，成功发送第一条消息后，将等待两个总线空闲的 CAN 位时间，然后才允许开始发送下一条请求的消息。如果其他 ECU 有挂起的消息，则会在空闲时间开始发送这些消息，不需要与 ECU-1 的下一条消息进行仲裁。接收到消息后， ECU-1 可在接收到的消息释放 CAN 总线后立即开始进行下一次发送。

此功能是通过 FDCAN_CCCR.TXP 控制的。如果此位置位，则每次成功发送消息后， FDCAN 都将暂停两个 CAN 位时间，然后再开始进行下一次发送。这样，即使网络中其他节点的 ID 优先级较低，也能发送消息。默认禁止此功能（FDCAN_CCCR.TXP=0）。此功能可使来自单个节点突发的多条消息变得稀疏一些，避免发送较低优先级消息的 ECU 出现应用程序错误地认为自己请求过多传输。

36.3.6.2. 专用发送缓冲区

专用发送缓冲区可完全在主机 CPU 的控制下发送消息。每个专用发送缓冲区都配置了特定的消息 ID。如果多个发送缓冲区配置为相同消息 ID，则会先发送编号最小的发送缓冲区中的消息，这些发送缓冲区应按升序请求发送，即先请求发送缓冲区编号最小的，或者，通过单次写入 FDCAN_TXBAR 同时请

求发送。

如果缓冲区中的消息有更新，通过置位 FDCAN_TXBAR.ARn 请求发送。。已请求发送的消息，会根据消息 ID 先在内部与发送 FIFO 或发送队列中的消息进行仲裁，然后在外部与 CAN 总线上的消息进行仲裁，最后根据仲裁结果发送出去。

消息 RAM 中分配的专用发送缓冲区元素大小为若干个 32 位字（4 字节）。因此，消息 RAM 中专用发送缓冲区的起始地址，是通过将发送缓冲区索引（0~31）乘以元素大小（字），再乘以 4（转换为字节），然后与 FDCAN_TXBC.TBSA 相加计算得到。

表 36-5 FDCAN 发送缓冲器、 FIFO（队列）元素大小

TXESC.TBDS2[2:0]	数据段（字节）	元素大小（字）
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

36.3.6.3. 发送 FIFO

发送 FIFO 模式是通过将 FDCAN_TXBC.TFQM 配置为 0 实现的。存储在发送 FIFO 中的消息，是从获取索引 FDCAN_TXFQS.TFGI 指向的消息开始发送的。每次发送后，获取索引会循环递增，直至发送 FIFO 为空。发送 FIFO 会按消息写入的顺序，发送来自不同发送缓冲区但 ID 相同的消息。FDCAN 通过计算获取索引与放入索引之差，得到可用（空闲）的发送 FIFO 元素个数（TXFQS.TFFL）。

新消息必须写入以放入索引 FDCAN_TXFQS.TFQPI 指向的发送缓冲区开始的发送 FIFO 中。添加发送请求会使放入索引递增，指向下一空闲的发送 FIFO 元素。当放入索引与获取索引相遇时，会指示发送 FIFO 已满（FDCAN_TXFQS.TFQF=1）。这种情况下，在下一条消息已发送且获取索引递增之前，不应继续向发送 FIFO 写入消息。

如果有单条消息添加到发送 FIFO，通过对发送 FIFO 放入索引对应的 FDCAN_TXBAR 位写 1 来请求发送。

如果有多条（假设有 n 条）消息添加到发送 FIFO，则会写入以放入索引开始的 n 个连续发送缓冲区中。随后通过 FDCAN_TXBAR 请求发送，放入索引循环递增 n。请求发送的发送缓冲区数不应超过发送 FIFO 空闲级别（FDCAN_TXFQS.TFFL）指示的空闲发送缓冲区数。

如果获取索引指向的发送缓冲区的发送请求已取消，那么获取索引会递增到下一个有发送请求挂起的发送缓冲区，并重新计算发送 FIFO 空闲级别（FDCAN_TXFQS.TFFL）。如果取消其他任何发送缓冲区的发送，获取索引和 FIFO 空闲级别保持不变。

消息 RAM 中分配的发送 FIFO 元素大小为若干个 32 位字（4 字节）。因此，下一个可用（空闲）的发送 FIFO 的起始地址，是通过将放入索引 FDCAN_TXFQS.TFQPI（0~31）乘以元素大小（字），再乘以 4，然后与 FDCAN_TXBC.TBSA 相加计算得到。

36.3.6.4. 发送队列

发送队列模式是通过将 FDCAN_TXBC.TFQM 配置为 1 实现的。存储在发送队列中的消息，是先从消息 ID 最小（优先级最高）的开始发送的。如果多个发送队列配置为相同消息 ID，如果多个发送队列配置为相同消息 ID，由于发送顺序与存储消息的发送队列的编号有关，而这些发送队列的编号依赖于当

前放入索引的状态，因此无法预测发送顺序。

新消息必须写入放入索引 FDCAN_TXFQS.TFQPI 指向的发送缓冲区中。放入索引一直指向发送队列中编号最小的空闲缓冲区。如果发送队列已满（FDCAN_TXFQS.TFQF=1），则放入索引无效，并且在至少有一个请求的消息已发出或挂起的发送请求已取消之前，不应继续向发送队列写入消息。

应用程序可使用寄存器 FDCAN_TXBRP 来代替放入索引，并可将消息放入任何没有挂起发送请求的发送缓冲区中。

消息 RAM 中分配的发送队列缓冲区元素大小为若干个 32 位字（4 字节）。因此，下一个可用（空闲）的发送队列缓冲区的起始地址，是通过将放入索引 FDCAN_TXFQS.TFQPI（0~31）乘以元素大小（字），再乘以 4，然后与 FDCAN_TXBC.TBSA 相加计算得到。

36.3.6.5. 混合使用专用发送缓冲区和发送 FIFO

在这种情况下，消息 RAM 中的发送缓冲区会被划分为一组专用发送缓冲区和一个发送 FIFO。专用发送缓冲区的数量是通过 FDCAN_TXBC.NDTB 配置的。分配给发送 FIFO 的发送缓冲区数量是通过 FDCAN_TXBC.TFQS 配置的。如果 FDCAN_TXBC.TFQS 配置为 0，则仅会使用专用发送缓冲区。

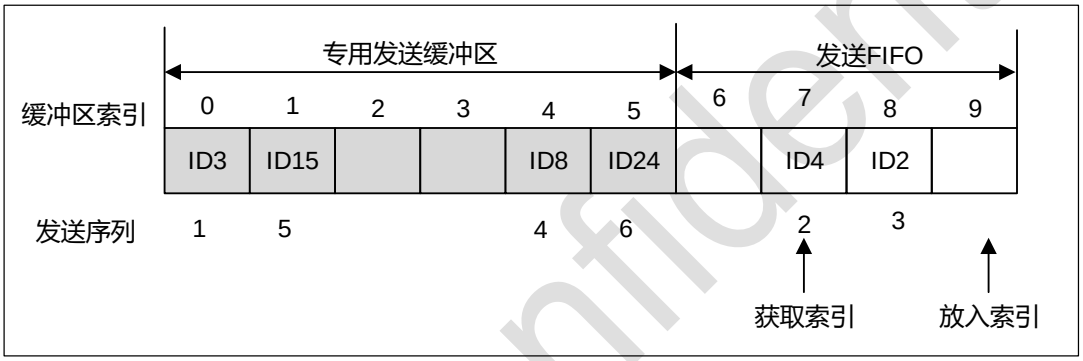


图 36-9 FDCAN 混合配置专用发送缓冲区和发送 FIFO 示例

发送优先级：

- 扫描所有激活了发送请求的专用发送缓冲区和最早挂起发送请求的发送 FIFO 缓冲区（FDCAN_TXEFS.TFGI 指向的缓冲区）
- 消息 ID 最小的缓冲区优先级最高，下次将发送该缓冲区的消息

36.3.6.6. 混合使用专用发送缓冲区和发送队列

在这种情况下，消息 RAM 中的发送缓冲区会被划分为一组专用发送缓冲区和一个发送队列。专用发送缓冲区的数量是通过 FDCAN_TXBC.NDTB 配置的。发送队列缓冲区的数量是通过 FDCAN_TXBC.TFQS 配置的。如果 FDCAN_TXBC.TFQS 配置为 0，则仅会使用专用发送缓冲区。

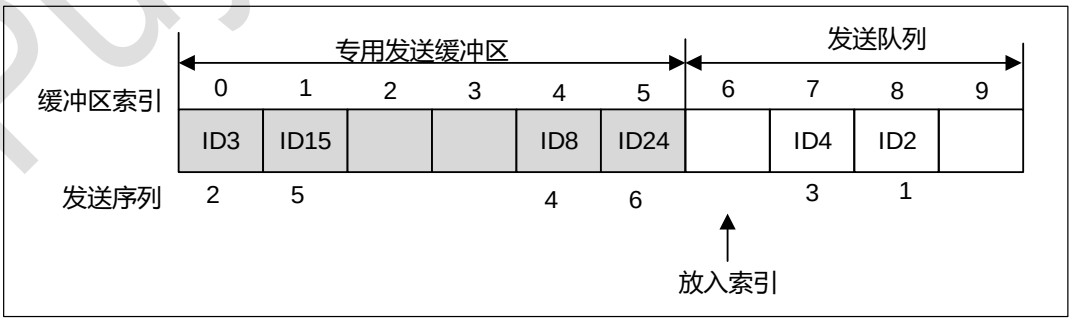


图 36-10 FDCAN 混合配置专用发送缓冲区和发送队列示例

发送优先级：

- 扫描所有激活了发送请求的专用发送缓冲区和发送队列缓冲区
- 消息 ID 最小的发送缓冲区优先级最高，下次将发送该缓冲区的消息

36.3.6.7. 发送取消

FDCAN 支持发送取消功能。这个功能特别适用于网关应用和基于 AUTOSAR 的应用。要取消专用发送缓冲区或发送队列缓冲区的发送请求，主机必须向寄存器 FDCAN_TXBCR 中相应的位（发送缓冲区索引对应的位）写 1。发送取消不适用于发送 FIFO 模式。

成功取消后，寄存器 FDCAN_TXBCF 的相应位会置 1。

如果在缓冲区发送过程中请求取消发送，则只要发送正在进行中， FDCAN_TXBRP 相应的位就会保持置 1 状态。如果发送成功， FDCAN_TXBTO 和 FDCAN_TXBCF 相应的位会置 1。如果发送不成功，则不会重复发送，只会将 FDCAN_TXBCF 相应的位置位。

注：如果一个挂起的发送在开始发送之前被立即取消，即使该节点中有另一消息挂起，取消后也会有一个短暂的窗口期，在此期间不会开始任何发送。这样，另一节点便可以发送优先级可能比该节点中第二条消息低的消息。

36.3.6.8. 发送事件处理

FDCAN 用发送事件 FIFO 来支持发送事件的处理。 FDCAN 在 CAN 总线上发送消息后，消息 ID 和时间戳会存储在发送事件 FIFO 元素中。为了将发送事件关联到发送事件 FIFO 元素，已发送的发送缓冲区中的消息标志会被复制到发送事件 FIFO 元素中。

发送事件 FIFO 最多可配置 32 个元素，章节【发送事件 FIFO 元素】描述了发送事件 FIFO 元素。

发送事件 FIFO 的用途是将处理发送状态信息与处理发送消息分开，也就是让发送缓冲区仅保存要发送的消息，而将发送状态单独存储在发送事件 FIFO 中。这样做是有优势的，尤其是在处理动态管理的发送队列时，发送缓冲区可在发送成功后立即用于新消息。重写发送缓冲区之前，不需要保存该发送缓冲区的发送状态信息。

当 FDCAN_IR.TEFF 指示发送事件 FIFO 已满时，在至少已读取一个元素且发送事件 FIFO 获取索引递增之前，不会继续向发送事件 FIFO 写入元素。如果在发送事件 FIFO 已满时发生发送事件，此事件会被丢弃，中断标志 FDCAN_IR.TEFL 置位。

为了避免发送事件 FIFO 溢出，可使用发送事件 FIFO 水位线。当发送事件 FIFO 填充数量达到由 TXEFC.EFWM 配置的发送事件 FIFO 水位线时，中断标志 FDCAN_IR.TEFW 置位。

从发送事件 FIFO 读取数据时，必须将发送事件 FIFO 获取索引 TXEFS.EFGI 乘以 2（发送事件 FIFO 元素大小为两个字），再乘以 4，然后与发送事件 FIFO 起始地址 TXEFC.EFSA 相加，以得到目标数据所在的地址。

36.3.7. FDCAN FIFO 确认处理

接收 FIF0、接收 FIF1 和发送事件 FIFO 的获取索引，是通过对相应 FIFO 确认索引寄存器进行写操作来控制的，请参见章节【FDCAN 接收 FIFO0 确认寄存器（FDCAN_RXF0A）】、章节【FDCAN 接收 FIFO1 确认寄存器(FDCAN_RXF1A)】和章节【FDCAN 发送事件 FIFO 确认寄存器(FDCAN_TXEFA)】。对 FIFO 确认索引进行写操作会将 FIFO 获取索引设置为 FIFO 确认索引加 1，进而会更新 FIFO 填充级别。有两种用例：

1.如果只从 FIFO 读取了一个元素（获取索引指向的元素），则应将获取索引值写入 FIFO 确认索引寄存器中。

2.如果已从 FIFO 中读取了一个序列的元素，则仅需在该序列读取结束时，将最后一个元素的索引值写入 FIFO 确认索引寄存器，即可更新 FIFO 获取索引。

由于 CPU 可自由访问 FDCAN 的消息 RAM，在以任意顺序（不考虑获取索引）读取 FIFO 元素时需要特别注意，特别是在从两个接收 FIFO 之一读取高优先级消息时。在这种情况下，不应对 FIFO 确认索

引寄存器进行写操作，否则会导致获取索引指向错误的位置，还会改变 FIFO 填充级别，一些较早的 FIFO 元素可能会丢失。

注：应用程序必须确保写入到 FIFO 确认索引寄存器中的值是正确的， FDCAN 不会检查数值是否正确。

36.3.8. FDCAN 消息 RAM

36.3.8.1. 消息 RAM 配置

消息 RAM 的宽度为 32 位。 FDCAN 模块最多可在消息 RAM 中分配 212 个字，如下图所示。实际应用中不要求对所有部分都进行配置，各部分之间的顺序也没有限制。

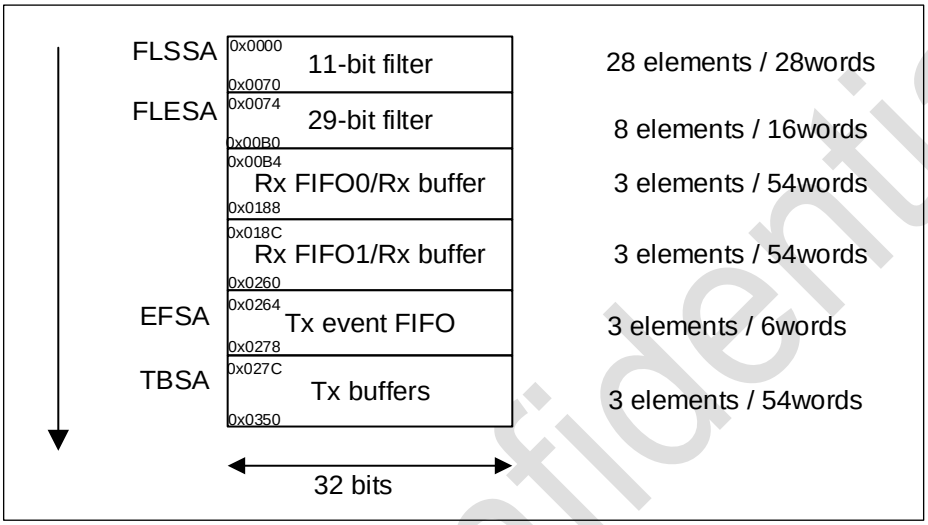


图 36-11 FDCAN 消息 RAM 配置

FDCAN 访问消息 RAM 时需要按 32 位对齐的地址进行寻址，而不是字节对齐的地址。可配置的起始地址是 32 位字地址，即仅评估地址的 15 到 2 位，忽略两个最低有效位。

注: FDCAN 不会检查消息 RAM 的配置是否存在错误，因此配置不同部分的起始地址以及各部分元素的数量时必须多加留意，以免造成数据入侵或丢失。

36.3.8.2. 接收缓冲区和 FIFO 元素

最多可在消息 RAM 中配置 2 个接收 FIFO/接收缓冲区。每个接收 FIFO/接收缓冲区均可配置为最多存储 3 个消息。可通过寄存器 FDCAN_RXESC 配置元素大小， FD CAN 消息的数据段最多可达 64 字节。

表 36-6 FDCAN 接收缓冲区和 FIFO 元素

位	31	24	23	16	15	8	7	0
R0	ESI	XTD RTR	ID[28:0]					
R1	ANMF	FIDX[6:0]	Res FDF BRS DLC[3:0]		RXTS[15:0]			
R2	DB3[7:0]		DB2[7:0]		DB1[7:0]		DB0[7:0]	
R3	DB7[7:0]		DB6[7:0]		DB5[7:0]		DB4[7:0]	
...	...		...		...		...	
Rn	DBm[7:0]		DBm-1[7:0]		DBm-2[7:0]		DBm-3[7:0]	

表 36-7 FDCAN 接收缓冲区和 FIFO 元素说明

位域	标记	说明
R0[31]	ESI	错误状态指示符（Error State Indicator） 0：发送节点为错误主动状态 1：发送节点为错误被动状态
R0[30]	XTD	扩展 ID 标志位（Extended Identifier） 向主机指示接收到的帧的 ID 是标准 ID 还是扩展 ID。

		0： 11 位标准 ID 1： 29 位扩展 ID
R0[29]	RTR	远程发送请求（Remote Transmission Request） 向主机指示接收到的是于数据帧还是远程帧。 0：接收到的帧是数据帧 1：接收到的帧是远程帧 注： FD CAN 没有远程帧。在 FD CAN 帧中（FDF=1），显性位 RRS（Remote Request Substitution）替代了 RTR 位。
R0[28:0]	ID[28:0]	ID（Identifier） 标准 ID 或扩展 ID 取决于 XTD 位，标准 ID 存储在 ID[28:18]中。
R1[31]	ANMF	接受非匹配帧（Accepted Non-matching Frame） 可通过 FDCAN_GFC.ANFS 和 FDCAN_GFC.ANFE 使能接受非匹配帧。 0：接收到的帧与过滤器索引 FIDX 指向的过滤器元素匹配 1：接收到的帧与任何接收过滤器元素都不匹配
R1[30:24]	FIDX[6:0]	过滤器索引（Filter Index） 0~127：匹配的接收过滤器元素的索引（ANMF =1 时无效）范围为 0 到 FDCAN_SIDFC.LSS]减 1 或 FDCAN_XIDFC.LSE 减 1。
R1[21]	FDF	FD 格式（FD Format） 0：经典 CAN 帧格式 1： FD CAN 帧格式（新的 DLC 编码和 CRC）
R1[20]	BRS	比特率切换（Bit Rate Switch） 0：接收帧没有比特率切换 1：接收帧有比特率切换
R1[19:16]	DLC[3:0]	数据长度代码 （Data Length Code） 0~8：经典 CAN 和 FD CAN：接收到的帧包含 0~8 个数据字节 9~15： CAN：接收到的帧包含 8 个数据字节 9~15： FD CAN：接收到的帧包含 12/16/20/24/32/48/64 个数据字节
R1[15:0]	RXTS[15:0]	接收时间戳（Rx Timestamp） 接收到 SOF（Start Of Frame）时捕获的时间戳计数器值。 分辨率取决于时间戳计数器预分频器 FDCAN_TSCC.TCP 的配置。
R2[31:24]	DB3[7:0]	数据字节 3（Data Byte 3）
R2[23:16]	DB2[7:0]	数据字节 2（Data Byte 2）
R2[15:8]	DB1[7:0]	数据字节 3（Data Byte 1）
R2[7:0]	DB0[7:0]	数据字节 2（Data Byte 0）
R3[31:24]	DB7[7:0]	数据字节 7（Data Byte 7）
R3[23:16]	DB6[7:0]	数据字节 6（Data Byte 6）
R3[15:8]	DB5[7:0]	数据字节 5（Data Byte 5）
R3[7:0]	DB4[7:0]	数据字节 4（Data Byte 4）
...	...	...
Rn[31:24]	DBm[7:0]	数据字节 m（Data Byte m）
Rn[23:16]	DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Rn[15:8]	DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Rn[7:0]	DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

36.3.8.3. 发送缓冲区元素

发送缓冲区可配置为专用发送缓冲区和发送 FIFO（或队列）。如果发送缓冲区由专用发送缓冲区和发送 FIFO（或队列）共享，则专用发送缓冲区在发送缓冲区的起始处，其后是发送 FIFO（或队列）。发送处理单元会评估发送缓冲区配置 FDCAN_TXBC.TFQS 和 FDCAN_TXBC.NDTB，以区分专用发送缓冲区与发送 FIFO（队列）。发送缓冲区元素的结构及相关说明如下表所示。 可通过寄存器 FDCAN_TXESC 配置元素大小， FD CAN 消息的数据段最多可达 64 字节。

表 36-8 FDCAN 发送缓冲区元素

位	31			24	23					16	15	8	7	0
T0	ESI	XTD	RTR	ID[28:0]										
T1	MM[7:0]				EFC	Res	FDF	BRS	DLC[3:0]		MM[15:8]		Res	
T2	DB3[7:0]				DB2[7:0]						DB1[7:0]		DB0[7:0]	
T3	DB7[7:0]				DB6[7:0]						DB5[7:0]		DB4[7:0]	
…	…				…						…		…	
Tn	DBm[7:0]				DBm-1[7:0]						DBm-2[7:0]		DBm-3[7:0]	

表 36-9 FDCAN 发送缓冲区元素说明

位域	标记	说明
T0[31]	ESI	错误状态指示符（Error State Indicator） 0：FD CAN 格式的 ESI 位仅取决于错误被动标志 1：FD CAN 格式的 ESI 位以隐性位发送 注： 发送缓冲区的 ESI 位会与错误被动标志进行或运算，以确定将发送的 FD 帧 ESI 位的值。 按照 FD CAN 协议规范的要求，错误主动节点可选择发送隐性 ESI 位，但错误被动节点将始终发送隐性 ESI 位。
T0[30]	XTD	扩展 ID 标志位（Extended Identifier） 0：11 位标准 ID 1：29 位扩展 ID
T0[29]	RTR	远程发送请求（Remote Transmission Request） 0：发送数据帧 1：发送远程帧 注： 当 RTR = 1 时，即使 FDCAN_CCCR.FDOE 使能以 FD CAN 格式进行发送，FDCAN 也会按照 ISO 11898-1:2015 的规定发送远程帧。
T0[28:0]	ID[28:0]	ID（Identifier） 标准 ID 或扩展 ID，取决于 XTD 位，标准 ID 必须写在 ID[28:18]中。
T1[31:24]	MM[7:0]	消息标记（Message Marker） 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中，用于标识发送消息状态。
T1[23]	EFC	事件 FIFO 控制（Event FIFO Control） 0：不存储发送事件 1：存储发送事件
T1[21]	FDF	FD 格式（FD Format） 0：以经典 CAN 格式发送帧 1：以 FD CAN 格式发送帧
T1[20]	BRS	比特率切换（Bit Rate Switch） 0： FD CAN 帧发送时不切换比特率

		1： FD CAN 帧发送时切换比特率 注： 仅当 FD CAN 使能（FDCAN_CCCR.FDOE=1）时，才会评估 ESI、 FDF 和 BRS 位。仅当额外设置 FDCAN_CCCR.BRSE =1 时，才会评估 BRS 位。
T1[19:16]	DLC[3:0]	数据长度代码（Data Length Code） 0~8：经典 CAN 和 FD CAN：发送帧包含 0~8 个数据字节 9~15： CAN：发送帧包含 8 个数据字节 9~15： FD CAN：发送帧包含 12/16/20/24/32/48/64 个数据字节
T1[15:8]	MM[15:8]	宽消息标记的高字节（High byte of Wide Message Marker） 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中，用于标识发送消息状态。仅当 FDCAN_CCCR.WMM=1 时有效。
T2[31:24]	DB3[7:0]	数据字节 3（Data Byte 3）
T2[23:16]	DB2[7:0]	数据字节 2（Data Byte 2）
T2[15:8]	DB1[7:0]	数据字节 3（Data Byte 1）
T2[7:0]	DB0[7:0]	数据字节 2（Data Byte 0）
T3[31:24]	DB7[7:0]	数据字节 7（Data Byte 7）
T3[23:16]	DB6[7:0]	数据字节 6（Data Byte 6）
T3[15:8]	DB5[7:0]	数据字节 5（Data Byte 5）
T3[7:0]	DB4[7:0]	数据字节 4（Data Byte 4）
...	...	...
Tn[31:24]	DBm[7:0]	数据字节 m（Data Byte m）
Tn[23:16]	DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Tn[15:8]	DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Tn[7:0]	DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

36.3.8.4. 发送事件 FIFO 元素

每个元素存储已发送消息相关的信息。通过读取发送事件 FIFO，主机 CPU 按消息的发送顺序获取这些信息。关于发送事件 FIFO 的状态信息，可从寄存器 FDCAN_TXEFS 中获取。发送事件 FIFO 元素的结构及相关说明如下表所示。

E1A: 当 FDCAN_CCCR.WMM=0 时，E1A.TXTS[15:0]保存由 FDCAN 的内部时间戳逻辑生成的 16 位时间戳。

E1B：当使能 16 位宽消息标志（FDCAN_CCCR.WMM=1）时， E1B.MM[15:8]为宽消息标志的高 8 位。

表 36-10 FDCAN 发送事件 FIFO 元素及说明

位	31			24	23				16	15	8	7	0
E0	ESI	XTD	RTR	ID[28:0]									
E1A	MM[7:0]				EF[1:0]	FDF	BRS	DLC[3:0]		TXTS[15:0]			
E1B	MM[7:0]				EF[1:0]	FDF	BRS	DLC[3:0]		MM[15:8]		Res	

位域	标记	说明
E0[31]	ESI	错误状态指示符（Error State Indicator） 0：发送节点为错误主动状态 1：发送节点为错误被动状态
E0[30]	XTD	扩展 ID 标志位（Extended Identifier） 0： 11 位标准 ID 1： 29 位扩展 ID

E0[29]	RTR	远程发送请求（Remote Transmission Request） 0：发送帧为数据帧 1：发送帧为远程帧
E0[28:0]	ID[28:0]	标识符（Identifier） 标准 ID 或扩展 ID，取决于 XTD 位。标准 ID 存储在 ID[28:18]中。
E1A/B[31:24]	MM[7:0]	消息标记（Message Marker） 从发送缓冲区复制到发送事件元素，用于标识发送消息状态。
E1A/B[23:22]	EF[1:0]	事件类型（Event Type） 00：保留 01：发送事件 10：取消后仍发送（在 DAR 模式下发送时始终为该值） 11：保留
E1A/B[21]	FDF	FD 格式（FD Format） 0：经典 CAN 帧格式 1：FD CAN 帧格式（新的 DLC 编码和 CRC）
E1A/B[20]	BRS	比特率切换（Bit Rate Switch） 0： FD CAN 帧发送时没有切换比特率 1： FD CAN 帧发送时有切换比特率
E1A/B[19:16]	DLC[3:0]	数据长度代码（Data Length Code） 0~8：经典 CAN 和 FD CAN：发送帧包含 0~8 个数据字节 9~15：经典 CAN：发送帧包含 8 个数据字节 9~15： FD CAN：发送帧包含 12/16/20/24/32/48/64 个数据字节
E1A[15:0]	TXTS[15:0]	发送时间戳（Tx Timestamp） 发送 SOF（Start Of Frame）时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 FDCAN_TSCC.TCP 的配置。
E1B[15:8]	MM[15:8]	宽消息标记的高字节（High byte of Wide Message Marker） 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中，用于标识发送消息状态。

36.3.8.5. 标准消息 ID 过滤器元素

最多可为 11 位标准 ID 配置 128 个过滤器元素 (每个元素大小为 1 个字) 。其地址为过滤器元素索引（0~127）乘以 4（转换成字节），加上起始地址 FDCAN_SIDFC.FLSSA。标准消息 ID 元素的结构及相关说明请见下表。

表 36-11 FDCAN 标准消息 ID 过滤器元素及说明

位	31	24	23	16	15	8	7	0
S0	SFT[1:0]	SFEC[2:0]	SFID1[10:0]		Res	SFID2[10:0]		

位域	标记	说明
S0[31:30]	SFT[1:0]	标准过滤器类型（Standard Filter Type） 00：从 SFID1 到 SFID2 的范围过滤器（SFID2 ≥ SFID1） 01： SFID1 或 SFID2 定义的双 ID 过滤器 10：经典过滤器： SFID1=ID， SFID2=掩码 11：禁止该过滤器元素 注：当 SFT=0b11 时，该过滤器元素被禁止，接收过滤继续进行（与 SFEC=0b000 时的行为相同） 。

S0[29:27]	SFEC[2:0]	标准过滤器元素配置（Standard Filter Element Configuration） 所有使能的过滤器元素都用于标准帧的接收过滤。当匹配到一个已使能的过滤器元素，或到达过滤器列表结尾处时，停止接收过滤。如果 SFEC=0b100、 0b101 或 0b110， 出现匹配时中断标志 FDCAN_IR.HPM 置位，并产生中断（如果使能）。在这种情况下，寄存器 FDCAN_HPMS 会更新为优先级匹配的状态。 000：禁止该过滤器元素 001：如果匹配，则存储到接收 FIFO0 中 010：如果匹配，则存储到接收 FIFO1 中 011：如果匹配，则拒绝 ID 100：如果匹配，则设置优先级，不存储 101：如果匹配，则设置优先级并存储到 FIFO0 中 110：如果匹配，则设置优先级并存储到 FIFO1 中 111：存储到接收缓冲区中或作为调试消息，忽略 SFT[1:0]的配置
S0[26:16]	SFID[10:0]	标准过滤器 ID1（Standard Filter ID 1） 标准 ID 过滤器元素的第一个 ID。当为接收缓冲区或调试消息过滤时，此位域定义要存储的标准消息的 ID。接收的 ID 必须完全匹配，不使用掩码机制。
S0[10:0]	SFID2[10:0]	标准过滤器 ID2（Standard Filter ID 2） 该位域具有不同含义，具体视 SFEC 的配置而定： – SFEC=0b001~0b110 时：标准 ID 过滤器元素的第二个 ID – SFEC=0b111 时： 用于接收缓冲区或调试消息的过滤器
	SFID2[10:9]	确定接收的消息存储到接收缓冲区中，还是作为调试消息序列中的消息 A、 B 或 C 进行处理。 00：将消息存储到接收缓冲区中 01：调试消息 A 10：调试消息 B 11：调试消息 C
	SFID2[5:0]	定义存储匹配消息的接收缓冲区到起始地址 RXBC.RBSA 的偏移（缓冲区索引） 。

36.3.8.6. 扩展消息 ID 过滤器元素

最多可为 29 位扩展 ID 配置 64 个过滤器元素（每个元素大小为 2 个字）。访问扩展消息 ID 过滤器元素时，其地址为过滤器元素索引（0~63）乘以 2，再乘以 4（转换成字节），然后加上起始地址 FDCAN_XIDFC.FLESA。扩展消息 ID 元素的结构及相关说明请见下表。

表 36-12 FDCAN 扩展消息 ID 过滤器元素

位	31 24		23 16	15 8	7 0
F0	EFEC[2:0]		EFID1[28:0]		
F1	EFT[1:0]	Res	EFID2[28:0]		

位域	标记	说明
F0[31:29]	EFEC[2:0]	扩展过滤器元素配置 （Extended Filter Element Configuration） 所有使能的过滤器元素都用于扩展帧的接收过滤。当匹配到一个已使能的过滤器元素，或到达过滤器列表结尾处时，停止接收过滤。如果 EFEC=0b100、 0b101 或 0b110，出现匹配时中断标志 FDCAN_IR.HPM 置位，并产生中断（若使能）。在这种情况下，寄存器 FDCAN_HPMS 会更新为优先级匹配的状态。 000：禁止该过滤器元素 001：如果匹配，则存储到接收 FIFO0 中

		010：如果匹配，则存储到接收 FIFO1 中 011：如果匹配，则拒绝 ID 100：如果匹配，则设置优先级 101：如果匹配，则设置优先级并存储到 FIFO0 中 110：如果匹配，则设置优先级并存储到 FIFO1 中 111：存储到接收缓冲区中或作为调试消息，忽略 EFT[1:0]的配置
F0[28:0]	EFID[28:0]	扩展过滤器 ID1（Extended Filter ID 1） 扩展 ID 过滤器元素的第一个 ID。当为接收缓冲区或调试消息过滤时，此位域定义要存储的扩展消息的 ID。接收的 ID 必须完全匹配，仅使用 FDCAN_XIDAM 掩码机制。
F1[31:30]	EFT[1:0]	扩展过滤器类型（Extended Filter Type） 00：从 EFID1 到 EFID2 的范围过滤器(EFID2 ≥ EFID1) 01： EFID1 或 EFID2 定义的双 ID 过滤器 10：经典过滤器： EFID1=ID， EFID2=掩码 11： 从 EFID1 到 EFID2 的范围过滤器（EFID2 ≥ EFID1）， 不使用 FDCAN_XIDAM 掩码
F1[28:0]	EFID2[28:0]	扩展过滤器 ID2（Extended Filter ID 2） 该位域具有不同含义，具体视 EFEC 的配置而定： – SFEC =0b001~0b110 时：扩展 ID 过滤器元素的第二个 ID – SFEC =0b111 时： 用于接收缓冲区或调试消息的过滤器
	SFID2[10:9]	确定接收的消息存储到接收缓冲区中，还是作为调试消息序列中的消息 A、 B 或 C 进行处理。 00：将消息存储到接收缓冲区中 01：调试消息 A 10：调试消息 B 11：调试消息 C
	SFID2[5:0]	定义存储匹配消息的接收缓冲区到起始地址 RXBC.RBSA 的偏移（缓冲区索引） 。

36.4. FDCAN 软件初始化

软件初始化是通过置位 FDCAN_CCCR.INIT 开始的。软件复位、硬件复位或进入 Bus_Off 会置位 FDCAN_CCCR.INIT。当 FDCAN_CCCR.INIT 被置位时， CAN 总线上传入和传出的消息都将停止，FDCAN_TX 引脚的状态变为隐性（高电平）。错误逻辑管理单元（EML）的计数器保持不变。将 FDCAN_CCCR.INIT 置位不会改变任何配置寄存器。将 FDCAN_CCCR.INIT 清零可以完成软件初始化，随后，位流处理单元（BSP）等待总线上出现 11 个连续隐性位的序列（Bus_Idle），以此将其自身与 CAN 总线上的数据传输进行同步，然后才能参与总线活动并开始消息传输。

仅当 FDCAN_CCCR.INIT 和 FDCAN_CCCR.CCE 均置位时，才能访问 FDCAN 配置寄存器。FDCAN 配置寄存器如下表所示。

表 36-13 FDCAN 配置寄存器列表

寄存器	说明
FDCAN_DBTP	FDCAN 数据位时间和预分频寄存器
FDCAN_TEST	FDCAN 测试寄存器
FDCAN_RWD	FDCAN RAM 看门狗寄存器
FDCAN_CCCR	FDCAN CC 控制寄存器
FDCAN_NBTP	FDCAN 标称位时间和预分频寄存器
FDCAN_TSCC	FDCAN 时间戳计数器配置寄存器

FDCAN_TOCC	FDCAN 超时计数器配置寄存器
FDCAN_TDCR	FDCAN 发送器延迟补偿寄存器
FDCAN_GFC	FDCAN 全局过滤器配置寄存器
FDCAN_SIDFC	FDCAN 标准 ID 过滤器配置寄存器
FDCAN_XIDFC	FDCAN 扩展 ID 过滤器配置寄存器
FDCAN_XIDAM	FDCAN 扩展 ID 与掩码寄存器
FDCAN_RXF0C	FDCAN 接收 FIFO0 配置寄存器
FDCAN_RXBC	FDCAN 接收缓冲区配置寄存器
FDCAN_RXF1C	FDCAN 接收 FIFO1 配置寄存器
FDCAN_RXESC	FDCAN 接收缓冲区和 FIFO 元素大小配置寄存器
FDCAN_TXBC	FDCAN 发送缓冲区配置寄存器
FDCAN_TXESC	FDCAN 发送缓冲区元素大小配置寄存器
FDCAN_TXEFC	FDCAN 发送事件 FIFO 配置寄存器

FDCAN_CCCR.CCE 仅在 FDCAN_CCCR.INIT 置位时才能被置位或清零，在 FDCAN_CCCR.INIT 清零时自动清零。

FDCAN_CCCR.CCE 置位后，以下寄存器会被复位：

FDCAN_HPMS——高优先级消息状态

FDCAN_RXF0S——接收 FIFO 0 状态

FDCAN_RXF1S——接收 FIFO 1 状态

FDCAN_TXFQS——发送 FIFO/队列状态

FDCAN_TXBRP——发送缓冲区请求挂起

FDCAN_TXBTO——发送缓冲区发送发生

FDCAN_TXBCF——发送缓冲区取消完成

FDCAN_TXEFS——发送事件 FIFO 状态

FDCAN_TXBAR——发送缓冲区添加请求

FDCAN_TXBCR——发送缓冲区取消请求

当 CCCR.CCE 置位时，超时计数器值 TOCV.TOC 预设为 TOCC.TOP 配置的值。

以下寄存器仅在 CCCR.CCE = ‘0’ 时可写：

TXBAR - Tx 缓冲区添加请求

TXBCR - Tx 缓冲区取消请求

仅当 FDCAN_CCCR.INIT 和 FDCAN_CCCR.CCE 均已置位时，主机才能将 FDCAN_CCCR.TEST 和 FDCAN_CCCR.MON 置位，主机可随时复位 FDCAN_CCCR.TEST 和 FDCAN_CCCR.MON。仅当 FDCAN_CCCR.INIT 和 FDCAN_CCCR.CCE 均已置位时，才能将 FDCAN_CCCR.DAR 置位或清零。

36.5. FDCAN 寄存器

36.5.1. FDCAN 字节序寄存器（FDCAN_ENDN）

地址偏移：0x004

复位值：0x87654321

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETV[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	ETV	R	0x87654321	字节序测试值

36.5.2. FDCAN 数据位时间和预分频寄存器（FDCAN_DBTP）

地址偏移：0x00C

复位值：0x0000 0A33

仅当 FDCAN_CCCR.CCE 和 FDCAN_CCCR.INIT 位均被置位时，才能对该寄存器进行写访问。 CAN 位时间的可编程范围为 4 到 49 个时间片（Time quanta， tq）。 CAN 时间片的可编程范围为 1 到 32 个 FDCAN 通信时钟周期（Minimum time quantum， mtq）。 tq = (DBRP + 1)mtq。

DTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。 DTSEG2 为 Phase_Seg2。

因此，位时间的长度为（编程值） [DTSEG1 + DTSEG2 + 3] tq 或（功能值） [Sync_Seg + Prop_Seg +Phase_Seg1 + Phase_Seg2] tq。

信息处理时间（IPT）为零，这意味着下一位数据在采样点之后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDC	Res.	Res.	DBRP[4:0]				
								RW			RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	DTSEG1[4:0]					DTSEG2[3:0]				DSJW[3:0]			
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	TDC	RW	0	发送器延迟补偿 0：禁止发送器延迟补偿 1：使能发送器延迟补偿
22:21	Reserved	-	-	保留
20:16	DBRP[4:0]	RW	0	数据比特率预分频 生成位时间片的 FDCAN 通信时钟预分频值，位时间为该时间片的倍数。预分 频有效值为 0 到 31，当 TDC=1 时，范围被限制在 0 到 1，硬件实际使用值为编 程值加 1。
15:13	Reserved	-	-	保留
12:8	DTSEG1[4:0]	RW	0	采样点之前的数据时间 有效值为 0 到 31，硬件实际使用值为编程值加 1。
7:4	DTSEG2[3:0]	RW	0	采样点之后的数据时间 有效值为 1 到 15，硬件实际使用值为编程值加 1。
3:0	DSJW[3:0]	RW	0	数据（重新）同步跳转宽度 有效值为 0 到 15，硬件实际使用值为编程值加 1。

注：当 FDCAN 通信时钟为 8MHz 时，复位值 0x0000 0A33 会将 FDCAN 数据段的比特率配置为 500Kbps。通过 DBTP 配置的 FD CAN 数据段比特率不能小于通过 NBTP 配置的仲裁段比特率。

36.5.3. FDCAN 测试寄存器（FDCAN_TEST）

地址偏移：0x010

复位值：0x0000 00X0（位 7 的复位值由 FDCAN_RX 引脚的实际电平决定）

仅当 FDCAN_CCCR.TEST 位被置 1 后，才能对该测试寄存器进行写访问。当 FDCAN_CCCR.TEST 位被清 0 时，所有测试寄存器功能都会设为其复位值。

回环模式和软件控制 FDCAN_TX 引脚属于硬件测试模式。 TX 配置为非 0b00 时可能会干扰 CAN 总线上的消息传输。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SVAL	TXBNS[4:0]				
										R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	PVAL	TXBNP[4:0]					RX	TX[1:0]		LBCK	Res.	Res.	Res.	Res.
		R	R	R	R	R	R	R	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21	SVAL	R	0	已开始有效 0： TXBNS 的值无效 1： TXBNS 的值有效
20:16	TXBNS[4:0]	R	5’b0	已开始发送的缓冲区编号 最后开始发送的发送缓冲区编号。 当 SVAL 位被置 1 时有效，有效范围为 0~31。
15:14	Reserved	-	-	保留
13	PVAL	R	0	已准备有效 0： TXBNP 的值无效 1： TXBNP 的值有效
12:8	TXBNP[4:0]	R	5’b0	已准备好发送的缓冲区编号 准备好发送的发送缓冲区编号。 当 PVAL 位被置 1 时有效，有效范围为 0~31。
7	RX	R	0	接收引脚 监测到的 FDCAN_RX 引脚的实际值 0： CAN 总线为显性（FDCAN_RX=0） 1： CAN 总线为隐性（FDCAN_RX=1）
6:5	TX[1:0]	RW	2’b0	发送引脚控制 00：复位值， FDCAN_TX 引脚由 CAN 内核控制，在 CAN 位时间结束时更新 01：可以在 FDCAN_TX 引脚监测采样点 10：FDCAN_TX 引脚输出显性电平（0） 11：FDCAN_TX 引脚输出隐性电平（1）
4	LBCK	RW	0	回环模式 0：复位值，禁止回环模式 1：使能回环模式（详见【外部回环模式】 和【内部回环模式】 章节）
3:0	Reserved	-	-	保留

36.5.4. FDCAN RAM 看门狗寄存器（FDCAN_RWD）

地址偏移：0x014

复位值：0x0000 0000

RAM 看门狗会监视消息 RAM 的 READY 信号输出。对消息 RAM 进行访问时，会启动消息 RAM 看门狗计数器，该计数器的起始值由 FDCAN_RWD.WDC 位进行配置。

当消息 RAM 通过激活其 READY 信号输出，指示访问成功完成时，该计数器会重新载入 FDCAN_RWD.WDC 的值。如果计数器递减为 0 之前消息 RAM 没有响应，计数器将停止计数，并将中断标志 FDCAN_IR.WDI 位置 1。 RAM 看门狗计数器由 FDCAN 控制逻辑时钟计数。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDV[7:0]								WDC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	ResetValue	Function
31:16	Reserved	-	-	保留
15:8	WDV[7:0]	R	8’b0	看门狗计数值 消息 RAM 看门狗计数器的实际值
7:0	WDC[7:0]	R	8’b0	看门狗配置 消息 RAM 看门狗计数器的起始值。如果使用初始值 0x00，则计数器被禁止。

36.5.5. FDCAN CC 控制寄存器（FDCAN_CCCR）

地址偏移：0x018

复位值：0x00000001

寄存器各个位的设置条件请参见【软件初始化】章节。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	WMM	Res.	BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	NISO	RW	0	非 ISO 模式选择 0：使用 ISO 11898-1:2015 规定的 FD CAN 帧格式 1：使用 Bosch FD CAN 规范 V1.0 规定的 FD CAN 帧格式
14	TXP	RW	0	传输暂停 若该位被置为 1，则 FDCAN 在成功传输完一帧数据后会暂停两个位时间，再进行 下一帧数据的传输。 0：禁止传输暂停 1：使能传输暂停
13	EFBI	RW	0	总线同步期间的边沿滤波 0：禁止边沿滤波 1：需要两个连续的显性 tq 才能检测硬同步边沿
12	PXHD	RW	0	协议异常处理禁止 0：使能协议异常处理 1：禁止协议异常处理 注：

				当协议异常处理被禁止时， FDCAN 将在检测到协议异常条件时传输错误帧。
11	WMM	RW	0	宽消息标志 当使用 16 位宽消息标志（WMM=1）时，发送事件 FIFO 中的 16 位内部时间戳是无效的。 0：使用 8 位消息标志 1：使用 16 位消息标志，替换发送事件 FIFO 中的 16 位时间戳
10	Reserved	-	-	保留
9	BRSE	RW	0	比特率切换 0：禁止发送时进行比特率切换 1：使能发送时进行比特率切换 注： 当 FD CAN 模式无效（FDOE=0）时， BRSE 的设置不起作用。
8	FDOE	RW	0	FD 模式使能 0：禁止 FD 模式 1：使能 FD 模式
7	TEST	RW	0	测试模式使能 0：正常模式，测试寄存器 FDCAN_TEST 保持复位值 1：测试模式，使能写访问测试寄存器 FDCAN_TEST
6	DAR	RW	0	禁止自动重发 0：使能允许自动重发未成功发送的消息 1：禁止自动重发
5	MON	RW	0	总线监听模式 仅当 CCE 和 INIT 位均置 1 时，主机才能将 MON 位置 1。此位可随时通过主机复位。 0：禁止总线监听模式 1：使能总线监听模式
4	CSR	RW	0	时钟停止请求 当已请求时钟停止时，在所有挂起的传输请求均已完成且 CAN 总线达到空闲状 态之后， INIT 位和 CSA 位将会依次被置 1。 0：未请求时钟停止 1：已请求时钟停止
3	CSA	RW	0	时钟停止请求 当已请求时钟停止时，在所有挂起的传输请求均已完成且 CAN 总线达到空闲状 态之后， INIT 位和 CSA 位将会依次被置 1。 0：未请求时钟停止 1：已请求时钟停止
2	ASM	RW	0	传输暂停 若该位被置为 1，则 FDCAN 在成功传输完一帧数据后会暂停两个位时间，再进行 下一帧数据的传输。 0：禁止传输暂停 1：使能传输暂停
1	CCE	RW	0	配置更改使能

				0：CPU 对受保护的配置寄存器无写访问权限 1：CPU 对受保护的配置寄存器有写访问权限（当 INIT=1 时）
0	INIT	RW	0	0：正常工作 1：启动初始化 注： 由于两个时钟域之间存在同步机制， INIT 的值在写入后可能会有一定延迟才能被正确读取。因此，在重新设定 INIT 值之前，请确认之前的设定值确实已被写入。

36.5.6. FDCAN 标称位时间和预分频寄存器（FDCAN_NBTP）

地址偏移：0x01C

复位值：0x0600 0A03

仅当 FDCAN_CCCR.CCE 和 FDCAN_CCR.INIT 位均被置 1 时，才能对该寄存器进行写访问。 CAN 标称位时间的可编程范围为 4 到 385 个时间片（Time quanta， tq）。 CAN 时间片的可编程范围为 1 到 512 个 FDCAN 通信时钟周期（Minimum time quantum， mtq）。 tq = (NBRP + 1)mtq。

NTSEG1 为 Prop_Seg 与 Phase_Seg1 之和。 NTSEG2 为 Phase_Seg2。

因此，位时间的长度为（编程值） [NTSEG1 + NTSEG2 + 3] tq 或（功能值） [Sync_Seg + Prop_Seg +Phase_Seg1 + Phase_Seg2] tq。

信息处理时间（IPT）为零，这意味着下一位数据在采样点之后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NSJW[6:0]							NBRP[8:0]								
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NTSEG1[7:0]									NTSEG2[6:0]						
RW	RW	RW	RW	RW	RW	RW	RW	Res.	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:25	NSJW[6:0]	RW	7’h6	标称（重）同步跳转宽度 有效值为 0 到 127，硬件实际使用值为编程值加 1。
24:16	NBRP[8:0]	RW	9’b0	标称比特率预分频 生成位时间片的 FDCAN 通信时钟预分频值，位时间为该时间片的倍数。预分 频有效值为 0 到 511，硬件实际使用值为编程值加 1。
15:8	NTSEG1[7:0]	RW	8’hA	采样点之前的标称时间段 有效值为 1 到 255，硬件实际使用值为编程值加 1。
7	Reserved	-	-	保留
6:0	NTSEG2[6:0]	RW	7’h3	采样点之后的标称时间段 有效值为 1 到 127，硬件实际使用值为编程值加 1。

36.5.7. FDCAN 时间戳计数器配置寄存器（FDCAN_TSCC）

地址偏移：0x020

复位值：0x0000 0000

该寄存器用于配置内部 16 位时间戳计数器。内部时间戳处理的介绍请参见章节【时间戳生成】。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCP[3:0]			
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TSS[1:0]	
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:16	TCP[3:0]	RW	4’b0	时间戳计数器预分频 将时间戳和超时计数器时间单位配置为 CAN 位时间的 1~16 倍。 有效设定值为 0~15，硬件实际使用值为编程值加 1。
15:2	Reserved	-	-	保留
1:0	TSS[1:0]	R	2’b0	时间戳选择 00：时间戳计数器的值一直为 0x0000 01：时间戳计数器的值根据 TCP 递增 10：禁止设定 11：时间戳计数器的值一直为 0x0000 （等同于设定值“00”）

36.5.8. FDCAN 时间戳计数器值寄存器（FDCAN_TSCV）

地址偏移：0x024

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TSC[15:0]															
RW	RW	RW	RW	RW	Res.	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	TSC[15:0]	RW	16’b0	时间戳计数器 在接收到或发送 SOF（Start Of Frame）时捕获的内部时间戳计数器值。 当 FDCAN_TSCC.TSS=0b01 时，时间戳计数器以 CAN 位时间的 1~16 倍进行递增，具体倍数由 FDCAN_TSCC.TCP 的设定值决定。 计数回卷时，中断标志 FDCAN_IR.TSW 会被置 1。写操作会将计数器清零。 注： 1. “回卷”是指时间戳计数器的值从非零变为零，并不是由于 FDCAN_TSCV 的写操作引起的清零。 2. 字节访问时，对该寄存器的任何一个字节进行写操作都会导致时间戳寄存器被清零。

36.5.9. FDCAN 超时计数器配置寄存器（FDCAN_TOCC）

地址偏移：0x028

复位值：0xFFFF 0000

超时计数器的介绍请参见【超时计数器】章节。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TOS[1:0]		ETOC
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	TOP[15:0]	RW	16'b0	超时周期 超时计数器（递减计数器）的起始值，配置超时周期。
15:3	Reserved	-	-	保留
2:1	TOS[1:0]	RW	2'b0	超时选择 在连续模式下操作时，对 FDCAN_TOCV 进行写访问会将计数器预设为 FDCAN_TOCC.TOP 配置的值，并继续向下递减。当超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为 FDCAN_TOCC.TOP 配置的值。当第一个 FIFO 元素被存储后，计数器开始递减计数。 00：连续模式 01：由发送事件 FIFO 控制超时 10：由接收 FIFO0 控制超时 11：由接收 FIFO1 控制超时
0	ETOC	RW	0	超时计数器使能 0：禁止超时计数器 1：使能超时计数器

36.5.10. FDCAN 超时计数器值寄存器（FDCAN_TOCV）

地址偏移：0x02C

复位值：0x0000 FFFF

超时计数器的介绍请参见【超时计数器】章节。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	TOC[15:0]	RW	16'hFFFF	超时计数器 超时计数器以 CAN 位时间的 1~16 倍进行递减，具体倍数由 FDCAN_TSCC.TCP 的设定值决定。当计数器递减到零时，中断标志位 FDCAN_IR.TOO 会被置 1，计数器停止计数。可以通过 FDCAN_TOCC.TOS 来设置启动和复位（或重启）条件。 注： 字节访问时，当 FDCAN_TOCC.TOS=0b00 时，对该寄存器的任何一个字节进行写操作都会将超时计数器预设为 FDCAN_TOCC.TOP 配置的值。

36.5.11. FDCAN 错误计数器寄存器（FDCAN_ECR）

地址偏移：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CEL[7:0]							
								RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R	RC_R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC[6:0]							TEC[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	0	
23:16	CEL[7:0]	RC_R	8'b0	CAN 错误记录 当 CAN 协议错误导致发送错误计数器 TEC 或接收错误计数器 REC 递增时，该错误记录计数器 CEL 都会递增。计数器值达到 0xFF 时，会停止计数。只有 RP 位置位但 REC 没有变化时，CEL 不会递增。 CEL 在 REC 和 TEC 增加后递增。 对 CEL 进行读访问会复位计数器。计数器值达到 0xFF 时，停止计数， TEC 或 REC 下一次增加时中断标志 FDCAN_IR.ELO 置位。 注：读寄存器 FDCAN_PSR 会将 CEL 清零。
15	RP	R	0	接收错误被动 0：接收错误计数器低于错误被动级别 128 1：接收错误计数器已达到错误被动级别 128
14:8	REC[6:0]	R	7'b0	接收错误计数器 接收错误计数器的实际状态，计数值范围为 0~127。
7:0	TEC[7:0]	R	8'b0	发送错误计数器 发送错误计数器的实际状态，计数值范围为 0~255。

注: 如果 FDCAN_CCCR.ASM 被置为 1,则检测到 CAN 协议错误时，CAN 协议控制器不会使 TEC 和 REC 递增，但 CEL 仍会递增。

36.5.12. FDCAN 协议状态寄存器（FDCAN_PSR)

地址偏移：0x044

复位值：0x0000 0707

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDCV[6:0]						
									R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	PXE	RFDF	RBRS	RESI	DLEC[2:0]			BO	EW	EP	ACT[1:0]		LEC[2:0]		
	RC_R	RC_R	RC_R	RC_R	RS_R	RS_R	RS_R	RW	RW	RW	RW	RW	RS_R	RS_R	RS_R

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22:16	TDCV[6:0]	R	7'b0	发送器延迟补偿值 第二采样点（Secondary Sample Point， SSP）的位置，由 FDCAN_TX 引脚到 FDCAN_RX 引脚之间测得的迟延值与 FDCAN_TDCR.TDCO 之和决定。 SSP 位置在数据段。 TDCV 为已发送位的起点与第二采样点之间的 mtq 数，有效值为 0~127 个 mtq。
15	Reserved	-	-	保留
14	PXE	RC_R	0	协议异常事件 0：自上次读访问后未发生协议异常事件 1：已发生协议异常事件 访问类型为 RX：进行读访问时复位。 注：读寄存器 FDCAN_PSR 会将 PXE 清零。
13	RFDF	RC_R	0	收到 FD CAN 消息 此位的置位与接收过滤无关。

				0：自从上次被 CPU 清零后未收到 FD CAN 格式的消息 1：收到 FDF 标志为 1 的 FD CAN 格式的消息 注：读寄存器 FDCAN_PSR 会将 RFDF 清零。
12	RBRS	RC_R	0	上次收到的 FD CAN 消息的 BRS 标志 该位与 RFDF 一起置位，与接收过滤无关。 0：上次收到的 FD CAN 消息的 BRS 标志未置位 1：上次收到的 FD CAN 消息的 BRS 标志已置位 注：读寄存器 FDCAN_PSR 会将 RBRS 清零。
11	RESI	RC_R	0	上次收到的 FD CAN 消息的 ESI 标志 该位与 RFDF 一起被设置，与验收过滤器无关。 0：上次收到的 FD CAN 消息的 ESI 标志未置位 1：上次收到的 FD CAN 消息的 ESI 标志已置位 注：读寄存器 FDCAN_PSR 会将 RESI 清零。
10:8	DLEC[2:0]	RS_R	0	数据段最后一次错误码 在设置了 BRS 标志的 FD CAN 帧的数据段中发生的最后一个错误的类型。错误码与 LEC 相同。当设置了 BRS 标志的 FD CAN 帧传输（接收或发送）没有错误时， DLEC 将被清零。 注：读寄存器 FDCAN_PSR 会将 DLEC[2:0]置位为 3’b111。
7	BO	R	0	Buss_Off 状态 0：FDCAN 未处于 Bus_Off 状态 1：FDCAN 处于 Bus_Off 状态
6	EW	R	0	警告状态 0：两个错误计数器均小于错误警告（Error_Warning）上限 96 1：至少有一个错误计数器已达到错误警告（Error_Warning）上限 96
5	EP	R	0	错误被动 0： FDCAN 处于错误主动（Error_Active）状态。FDCAN 正常参与总线通信，并在检测到错误时发送主动错误标志。 1： FDCAN 处于错误被动（Error_Passive）状态
4:3	ACT[1:0]	R	2’b0	通信状态 监测 FDCAN 的通信状态。 00：同步中，节点正在进行 CAN 通信同步 01：空闲，节点既不是接收器也不是发送器 10：接收器，节点作为接收器工作 11：发送器，节点作为发送器工作 注： 协议异常事件会将 ACT 设为 2’b00
2:0	LEC[2:0]	RS_R	3’b0	最后一次错误码 0：没有错误（No Error）。自成功接收或发送使 LEC 复位之后，没有错误发生。 1： 填充错误（Stuff Error）。在已接收到的消息中，某一部分出现了 5 个以上的相同 位，这种情况是不允许的。 2：格式错误（Form Error）。已接收帧的固定格式部分有格式错误。 3：ACK 错误（ACK Error）。 FDCAN 发送的消息没有被其他节点确认。

				4：位 1 错误（Bit1 Error）。在消息的发送过程中（仲裁段除外），设备想发送一个隐性电平（逻辑值 1），但是监测到的总线电平为显性（逻辑值 0）。 5：Bit0 错误（Bit0 Error）。在消息的发送过程中（包括 ACK 位、主动错误标志、过载标志），设备想发送一个显性电平（数据或 ID 的逻辑值 0），但是监测到的总线电平为隐性（逻辑值 1）。在总线 Bus_Off 恢复期间，每次监测到 11 位隐性位序列时，都会设置为该状态。这样， CPU 就能够监测总线恢复序列的进程（表明总线未卡在显性状态或持续受到干扰）。 6： CRC 错误（CRC Error）。已接收消息的 CRC 有错误。收到的 CRC 和用接收到的数据计算出的 CRC 不匹配。 7：无变化（No Change）。对协议状态寄存器的任何读访问会将 LEC 重新初始化为 7。 当 LEC 显示值为 7 时， 表示自上次 CPU 对协议状态寄存器进行读访问之后，没有检测到 CAN 总线事件。 注： 1.当设置了 BRS 标志的 FD CAN 帧已经到达数据段时，下一个 CAN 事件（错误或有效帧）将显示在 DLEC 中而不是 LEC 中。FD CAN CRC 序列中的固定填充位的错误将显示为格式错误（Form Error）而不是填充错误（Stuff Error） 。 2.Bus_Off 恢复序列（请参考 ISO11898-1:2015） 不能通过置位或清零 FDCAN_CCCR.INIT 来缩短。 如果设备进入 Bus_Off 状态， 它将自动置位 FDCAN_CCCR.INIT，停止所有总线活动。一旦 FDCAN_CCCR.INIT 被 CPU 清零， 设备在恢复正常操作之前，将等待 129 个总线空闲（129*11 个连续的隐性位）。在 Bus_Off 恢复序列结束时，错误管理计数器将被复位。在清零 FDCAN_CCCR.INIT 后的等待期间，每次监测到一个 11 位隐性序列时，就向 FDCAN_PSR.LEC 写入一个 Bit0 错误码，使 CPU 能够随时检查 CAN 总线是否处于显性状态或持续被干扰，并监测 Bus_Off 恢复序列。FDCAN_ECR.REC 用于对这些序列进行计数。 3. 读寄存器 FDCAN_PSR 会将 LEC[2:0]置位为 3’b111。

36.5.13. FDCAN 发送器延迟补偿寄存器（FDCAN_TDCR）

地址偏移：0x048

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDCO[6:0]								TDCF[6:0]						
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14:8	TDCO[6:0]	RW	7’b0	发送器延迟补偿 SSP 偏移 该偏移值是指，从 FDCAN_TX 引脚到 FDCAN_RX 引脚测得的延迟与第二采样点（SSP）之间的距离。有效设定值为 0~127mtq。

7	Reserved	-	-	保留
6:0	TDCF[6:0]	RW	7'b0	发送器延迟补偿滤波窗口宽度 定义 SSP 位置的最小值，对于发送器延迟的测量，将忽略 FDCAN_RX 引脚上会导致 SSP 位置提前的显性边沿。当 TDCF 设定值大于 TDCO 时，该功能被启用。有效设定值为 0~127mtq。

36.5.14. FDCAN 中断寄存器（FDCAN_IR）

地址偏移：0x050

复位值： 0x0000 0000

当如下列表中的一个条件被检测到时，对应标志位会置位。在主机将标志位清零之前，保持置位状态。所有标志位写 1 清零，写 0 无影响。硬复位会将寄存器清零。是否产生中断，由寄存器 FDCAN_IE 控制；哪条中断线产生中断，由寄存器 FDCAN_ILS 控制。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ARA	PED	PEA	WDI	BO	EW	EP	ELO	Res.	Res.	DRX	TOO	MRAF	TSW
		RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	ARA	RW	0	访问保留地址（Access to Reserved Address） 0：未对保留地址进行访问 1：已对保留地址进行访问
28	PED	RW	0	数据段协议错误（使用数据位时间）(Protocol Error in Data Phase (Data Bit Time isused)) 0：未检测到数据段协议错误 1： 检测到数据段协议错误（FDCAN_PSR.DLEC≠0/7）
27	PEA	RW	0	仲裁段协议错误（使用标称位时间） (Protocol Error in Arbitration Phase (NominalBit Time is used)) 0：未检测到仲裁段协议错误 1： 检测到仲裁段协议错误（FDCAN_PSR.LEC≠0/7）
26	WDI	RW	0	看门狗中断（Watchdog Interrupt） 0：未发生消息 RAM 看门狗事件 1：因 READY 信号缺失而发生消息 RAM 看门狗事件
25	BO	RW	0	Bus_Off 状态（Bus_Off Status） 0： Bus_Off 状态未改变 1： Bus_Off 状态已改变
24	EW	RW	0	警告状态（Warning Status） 0：错误警告（Error_Warning）状态未改变 1：错误警告（Error_Warning）状态已改变
23	EP	RW	0	错误被动（Error Passive） 0：错误被动（Error_Passive）状态未改变 1：错误被动（Error_Passive）状态已改变
22	ELO	RW	0	错误记录溢出（Error Logging Overflow） 0： CAN 错误记录计数器未溢出 1： CAN 错误记录计数器已溢出
21:20	Reserved	-	-	保留

19	DRX	RW	0	消息存储到专用接收缓冲区（Message stored to Dedicated Rx Buffer) 每当收到的消息已存储到专用接收缓冲区时，都会置位该标志。 0：接收缓存区未更新 1：至少有一条消息已存储到接收缓冲区
18	TOO	RW	0	发生超时（Timeout Occurred) 0：未发生超时 1：已发生超时
17	MRAF	RW	0	消息 RAM 访问失败（Message RAM Access Failure) 接收处理中置位条件： 1） 在接收到后续消息的仲裁段之前，尚未完成对已接收消息的接收过滤或存储。在这种情况下，接收过滤或消息存储被终止，接收处理单元将开始处理后续消息。 2） 无法向消息 RAM 写入消息，消息存储将终止。 在这两种情况下， FIFO 放入索引都不会更新，专用接收缓冲区的新数据标志不会置位，当下一条消息存储到该位置时，已部分存储的消息会被覆盖。 发送处理中置位条件： 1） 发送处理单元无法及时从消息 RAM 读取消息。在这种情况下，消息发送将终止。 在发送处理单元访问失败的情况下， FDCAN 被切换到受限工作模式（详见章节【受限工作模式】 ）。 要退出受限工作模式，主机 CPU 必须复位 FDCAN_CCCR.ASM。 0：未发生消息 RAM 访问失败 1：发生了消息 RAM 访问失败
16	TSW	RW	0	时间戳回卷（Timestamp Wraparound) 0：时间戳计数器未回卷 1：时间戳计数器已回卷
15	TEFL	RW	0	发送事件 FIFO 元素丢失（Tx Event FIFO Element Lost) 0：发送事件 FIFO 元素未丢失 1：发送事件 FIFO 元素有丢失；对空间大小为 0 的发送事件 FIFO 进行写操作时，该位也会置位
14	TEFF	RW	0	发送事件 FIFO 已满（Tx Event FIFO Full) 0：发送事件 FIFO 未满 1：发送事件 FIFO 已满
13	TEFW	RW	0	发送事件 FIFO 到达水位线（Tx Event FIFO Watermark Reached) 0：发送事件 FIFO 的填充级别低于水位线 1：发送事件 FIFO 的填充级别达到水位线
12	TEFN	RW	0	发送事件 FIFO 新条目（Tx Event FIFO New Entry) 0：发送事件 FIFO 无变化 1：发送处理单元写入了发送事件 FIFO 元素
11	TFE	RW	0	发送 FIFO 为空（Tx FIFO Empty) 0：发送 FIFO 非空 1：发送 FIFO 为空
10	TCF	RW	0	发送取消已完成（Transmission Cancellation Finished) 0：发送取消未完成 1：发送取消已完成

9	TC	RW	0	发送完成 (Transmission Completed) 0：发送未完成 1：发送已完成
8	HPM	RW	0	高优先级消息 (High Priority Message) 0：未收到高优先级消息 1：已收到高优先级消息
7	RF1L	RW	0	接收 FIFO1 消息丢失 (Rx FIFO 1 Message Lost) 0：接收 FIFO1 消息未丢失 1：接收 FIFO1 消息有丢失；对空间大小为 0 的接收 FIFO1 进行写操作时，该位也会置位
6	RF1F	RW	0	接收 FIFO1 已满 (Rx FIFO 1 Full) 0：接收 FIFO1 未满 1：接收 FIFO1 已满
5	RF1W	RW	0	接收 FIFO1 到达水位线 (Rx FIFO 1 Watermark Reached) 0：接收 FIFO1 的填充级别低于水位线 1：接收 FIFO1 的填充级别达到水位线
4	RF1N	RW	0	接收 FIFO1 新消息 (Rx FIFO 1 New Message) 0：没有新消息写入接收 FIFO1 1：有新消息写入接收 FIFO1
3	RF0L	RW	0	接收 FIFO0 消息丢失 (Rx FIFO 0 Message Lost) 0：接收 FIFO0 消息未丢失 1：接收 FIFO0 消息有丢失；对空间大小为 0 的接收 FIFO0 进行写操作时，该位也会置位
2	RF0F	RW	0	接收 FIFO0 已满 (Rx FIFO 0 Full) 0：接收 FIFO0 未满 1：接收 FIFO0 已满
1	RF0W	RW	0	接收 FIFO0 到达水位线 (Rx FIFO 0 Watermark Reached) 0：接收 FIFO0 的填充级别低于水位线 1：接收 FIFO0 的填充级别达到水位线
0	RF0N	RW	0	接收 FIFO0 新消息 (Rx FIFO 0 New Message) 0：没有新消息写入接收 FIFO0 1：有新消息写入接收 FIFO0

36.5.15. FDCAN 中断使能寄存器（FDCAN_IE）

地址偏移：0x054

复位值： 0x0000 0000

中断使能寄存器中的设置，决定中断寄存器（FDCAN_IR）中的哪些状态变化会指示在中断线上。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE	Res.	Res.	DRXE	TOOE	MRAFE	TSWE
		RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	ARAE	RW	0	访问保留地址使能 (Access to Reserved Address Enable)

				0：禁止中断 1：使能中断
28	PEDE	RW	0	数据阶段的协议错误使能 (Protocol Error in Data Phase Enable) 0：禁止中断 1：使能中断
27	PEAE	RW	0	仲裁阶段中的协议错误使能 (Protocol Error in Arbitration Phase Enable) 0：禁止中断 1：使能中断
26	WDIE	RW	0	看门狗中断使能 (Watchdog Interrupt Enable) 0：禁止中断 1：使能中断
25	BOE	RW	0	Bus_Off 状态中断 (Bus_Off Status Interrupt Enable) 0：禁止中断 1：使能中断
24	EWE	RW	0	警告状态中断使能 (Warning Status Interrupt Enable) 0：禁止中断 1：使能中断
23	EPE	RW	0	错误被动中断使能 (Error Passive Interrupt Enable) 0：禁止中断 1：使能中断
22	ELOE	RW	0	错误记录溢出中断使能 (Error Logging Overflow Interrupt Enable) 0：禁止中断 1：使能中断
21:20	Reserved	-	-	保留
19	DRXE	RW	0	消息存储到专用接收缓冲区中断使能 (Message stored to Dedicated RxBuffer Interrupt Enable) 0：禁止中断 1：使能中断
18	TOOE	RW	0	发生超时中断使能 (Timeout Occurred Interrupt Enable) 0：禁止中断 1：使能中断
17	MRAFE	RW	0	消息 RAM 访问失败中断使能 (Message RAM Access Failure Interrupt Enable) 0：禁止中断 1：使能中断
16	TSWE	RW	0	时间戳回卷中断使能 (Timestamp Wraparound Interrupt Enable) 0：禁止中断 1：使能中断
15	TEFLE	RW	0	发送事件 FIFO 元素丢失中断使能 (Tx Event FIFO Element Lost Interrupt Enable) 0：禁止中断 1：使能中断
14	TEFFE	RW	0	发送事件 FIFO 满中断使能 (Tx Event FIFO Full Interrupt Enable) 0：禁止中断 1：使能中断

13	TEFWE	RW	0	发送事件 FIFO 到达水位线中断使能(Tx Event FIFO Watermark Reached Interrupt Enable) 0：禁止中断 1：使能中断
12	TEFNE	RW	0	发送事件 FIFO 新条目中断使能 (Tx Event FIFO New Entry Interrupt Enable) 0：禁止中断 1：使能中断
11	TFEE	RW	0	发送 FIFO 空中断使能 (Tx FIFO Empty Interrupt Enable) 0：禁止中断 1：使能中断
10	TCFE	RW	0	发送取消已完成中断使能 (Transmission Cancellation Finished InterruptEnable) 0：禁止中断 1：使能中断
9	TCE	RW	0	发送完成中断使能 (Transmission Completed Interrupt Enable) 0：禁止中断 1：使能中断
8	HPME	RW	0	高优先级消息中断使能 (High Priority Message Interrupt Enable) 0：禁止中断 1：使能中断
7	RF1LE	RW	0	接收 FIFO1 消息丢失中断使能 (Rx FIFO 1 Message Lost Interrupt Enable) 0：禁止中断 1：使能中断
6	RF1FE	RW	0	接收 FIFO1 满中断使能 (Rx FIFO 1 Full Interrupt Enable) 0：禁止中断 1：使能中断
5	RF1WE	RW	0	接收 FIFO1 到达水位线中断使能 (Rx FIFO 1 Watermark Reached Interrupt Enable) 0：禁止中断 1：使能中断
4	RF1NE	RW	0	接收 FIFO1 新消息中断使能 (Rx FIFO 1 New Message Interrupt Enable) 0：禁止中断 1：使能中断
3	RF0LE	RW	0	接收 FIFO0 消息丢失中断使能 (Rx FIFO 0 Message Lost Interrupt Enable) 0：禁止中断 1：使能中断
2	RF0FE	RW	0	接收 FIFO0 满中断使能 (Rx FIFO 0 Full Interrupt Enable) 0：禁止中断 1：使能中断
1	RF0WE	RW	0	接收 FIFO0 到达水位线中断使能 (Rx FIFO 0 Watermark Reached Interrupt Enable) 0：禁止中断 1：使能中断
0	RF0NE	RW	0	接收 FIFO0 新消息中断使能 (Rx FIFO 0 New Message Interrupt Enable)

				0：禁止中断 1：使能中断

36.5.16. FDCAN 中断线选择寄存器（FDCAN_ILS）

地址偏移：0x058

复位值： 0x0000 0000

中断线选择寄存器，将中断寄存器（FDCAN_IR）中特定中断标志生成的中断分配到两条中断线之一。要产生中断，必须通过 FDCAN_ILE.EINT0 和 FDCAN_ILE.EINT1 使能相应的中断线。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	ARAL	PEDL	PEAL	WDIL	BOL	EWL	EPL	ELOL	Res.	Res.	DRXL	TOOL	MRAFL	TSWL
		RW	RW	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	ARAL	RW	0	访问保留地址中断线 (Access to Reserved Address Line) 0：选择中断线 0 1：选择中断线 1
28	PEDL	RW	0	数据段协议错误中断线 (Protocol Error in Data Phase Line) 0：选择中断线 0 1：选择中断线 1
27	PEAL	RW	0	仲裁段协议错误中断线 (Protocol Error in Arbitration Phase Line) 0：选择中断线 0 1：选择中断线 1
26	WDIL	RW	0	看门狗中断线 (Watchdog Interrupt Line) 0：选择中断线 0 1：选择中断线 1
25	BOL	RW	0	Bus_Off 状态中断线 (Bus_Off Status Interrupt Line) 0：选择中断线 0 1：选择中断线 1
24	EWL	RW	0	警告状态中断线 (Warning Status Interrupt Line) 0：选择中断线 0 1：选择中断线 1
23	EPL	RW	0	错误被动中断线 (Error Passive Interrupt Line) 0：选择中断线 0 1：选择中断线 1
22	ELOL	RW	0	错误记录溢出中断线 (Error Logging Overflow Interrupt Line) 0：选择中断线 0 1：选择中断线 1
21:20	Reserved	-	-	保留
19	DRXL	RW	0	消息存储到专用接收缓冲区中断线 (Message stored to Dedicated Rx Buffer Interrupt Line) 0：选择中断线 0 1：选择中断线 1
18	TOOL	RW	0	发生超时中断线 (Timeout Occurred Interrupt Line)

				0：选择中断线 0 1：选择中断线 1
17	MRAFL	RW	0	消息 RAM 访问失败中断线 (Message RAM Access Failure Interrupt Line) 0：选择中断线 0 1：选择中断线 1
16	TSWL	RW	0	时间戳回卷中断线 (Timestamp Wraparound Interrupt Line) 0：选择中断线 0 1：选择中断线 1
15	TEFLL	RW	0	发送事件 FIFO 元素丢失中断线 (Tx Event FIFO Element Lost Interrupt Line) 0：选择中断线 0 1：选择中断线 1
14	TEFFL	RW	0	发送事件 FIFO 已满中断线 (Tx Event FIFO Full Interrupt Line) 0：选择中断线 0 1：选择中断线 1
13	TEFWL	RW	0	发送事件 FIFO 到达水位线中断线 (Tx Event FIFO Watermark Reached Interrupt Line) 0：选择中断线 0 1：选择中断线 1
12	TEFNL	RW	0	发送事件 FIFO 新条目中断线 (Tx Event FIFO New Entry Interrupt Line) 0：选择中断线 0 1：选择中断线 1
11	TFEL	RW	0	发送 FIFO 为空中断线 (Tx FIFO Empty Interrupt Line) 0：选择中断线 0 1：选择中断线 1
10	TCFL	RW	0	发送取消已完成中断线 (Transmission Cancellation Finished Interrupt Line) 0：选择中断线 0 1：选择中断线 1
9	TCL	RW	0	发送完成中断线 (Transmission Completed Interrupt Line) 0：选择中断线 0 1：选择中断线 1
8	HPML	RW	0	高优先级消息中断线 (High Priority Message Interrupt Line) 0：选择中断线 0 1：选择中断线 1
7	RF1LL	RW	0	接收 FIFO1 消息丢失中断线 (Rx FIFO 1 Message Lost Interrupt Line) 0：选择中断线 0 1：选择中断线 1
6	RF1FL	RW	0	接收 FIFO1 已满中断线 (Rx FIFO 1 Full Interrupt Line) 0：选择中断线 0 1：选择中断线 1
5	RF1WL	RW	0	接收 FIFO1 到达水位线中断线 (Rx FIFO 1 Watermark Reached Interrupt Line) 0：选择中断线 0 1：选择中断线 1
4	RF1NL	RW	0	接收 FIFO1 新消息中断线 (Rx FIFO 1 New Message Interrupt Line)

				0：选择中断线 0 1：选择中断线 1
3	RF0LL	RW	0	接收 FIFO0 消息丢失中断线 (Rx FIFO 0 Message Lost Interrupt Line) 0：选择中断线 0 1：选择中断线 1
2	RF0FL	RW	0	接收 FIFO0 已已满中断线 (Rx FIFO 0 Full Interrupt Line) 0：选择中断线 0 1：选择中断线 1
1	RF0WL	RW	0	接收 FIFO0 到达水位线中断线 (Rx FIFO 0 Watermark Reached Interrupt Line) 0：选择中断线 0 1：选择中断线 1
0	RF0NL	RW	0	接收 FIFO0 新消息中断线 (Rx FIFO 0 New Message Interrupt Line) 0：选择中断线 0 1：选择中断线 1

36.5.17. FDCAN 中断线使能寄存器（FDCAN_ILE）

地址偏移：0x05C

复位值： 0x0000 0000

连接到 CPU 的两条中断线中的每一条，都可以通过设置 EINT0 和 EINT1 分别使能或禁用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EINT1	EINT0
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	EINT1	RW	0	中断线 1 使能 0：中断线 1 禁止 1：中断线 1 使能
0	EINT0	RW	0	中断线 0 使能 0：中断线 0 禁止 1：中断线 0 使能

36.5.18. FDCAN 全局过滤器配置寄存器（FDCAN_GFC）

地址偏移：0x080

复位值： 0x0000 0000

用于消息 ID 过滤的全局设置。全局过滤器配置，控制标准消息和扩展消息的过滤路径。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										ANFS[1:0]	ANFE[1:0]	RRFS	RRFE		
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5:4	ANFS[1:0]	RW	2'b0	接受不匹配的标准帧 定义如何处理，已接收的与过滤器列表任何元素都不匹配的 11 位标准 ID 消息。 00：在接收 FIFO0 中接受不匹配的标准帧 01：在接收 FIFO1 中接受不匹配的标准帧 10：拒绝不匹配的标准帧 11：拒绝不匹配的标准帧
3:2	ANFE[1:0]	RW	2'b0	接受不匹配的扩展帧 定义如何处理，已接收的与过滤器列表任何元素都不匹配的 29 位扩展 ID 消息。 00：在接收 FIFO0 中接受不匹配的扩展帧 01：在接收 FIFO1 中接受不匹配的扩展帧 10：拒绝不匹配的扩展帧 11：拒绝不匹配的扩展帧
1	RRFS	RW	0	拒绝远程标准帧 0：过滤 11 位标准 ID 的远程帧 1：拒绝所有 11 位标准 ID 的远程帧
0	RRFE	RW	0	拒绝远程扩展帧 0：过滤 29 位扩展 ID 的远程帧 1：拒绝所有 29 位扩展 ID 的远程帧

36.5.19. FDCAN 标准 ID 过滤器配置寄存器（FDCAN_SIDFC）

地址偏移：0x084

复位值： 0x0000 0000

用于 11 位标准 ID 过滤器的设置。标准 ID 过滤器配置，控制标准消息的过滤路径。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								LLS[7:0]							
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FLSSA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:16	LSS[7:0]	RW	8'b0	标准 ID 过滤器列表大小 0：无标准 ID 过滤器 1~128：标准 ID 过滤器元素数量 ＞128：大于 128 的设定值被解析为 128
15:2	FLSSA[15:2]	RW	14'b0	标准 ID 过滤器列表起始地址 设置标准 ID 过滤器列表的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

36.5.20. FDCAN 扩展 ID 过滤器配置寄存器（FDCAN_XIDFC）

地址偏移：0x088

复位值： 0x0000 0000

用于 29 位扩展 ID 过滤器的设置。扩展 ID 过滤器配置，控制扩展消息的过滤路径。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSE[6:0]						

									RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FLESA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22:16	LSE[7:0]	RW	8’b0	扩展 ID 过滤器列表大小 0：无扩展 ID 过滤器 1~64：扩展 ID 过滤器元素数量 ＞64：大于 64 的设定值被解析为 64
15:2	FLESA[15:2]	RW	14’b0	扩展 ID 过滤器列表起始地址 扩展 ID 过滤器列表的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

36.5.21. FDCAN 扩展 ID 与掩码寄存器（FDCAN_XIDAM）

地址偏移：0x090

复位值： 0x1FFF FFFF

用于 29 位扩展 ID 过滤器的设置。扩展 ID 过滤器配置，控制扩展消息的过滤路径。

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	EIDM[28:16]												
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EIDM[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28:0	EIDM[28:0]	RW	29’b0	扩展 ID 掩码 对于扩展帧的接收过滤，会将 EIDM 与已接收帧的 ID 进行与运算。用于屏蔽 SAE J1939 中的 29 位 ID。所有位的复位值都被设置为 1，掩码无效。

36.5.22. FDCAN 高优先级消息状态寄存器（FDCAN_HPMS）

地址偏移：0x094

复位值： 0x0000 0000

配置为生成优先级事件的消息 ID 过滤器元素，每次匹配时都会更新此寄存器。这可用于监测收到的高优先级消息的状态，并可对这些消息进行快速访问。

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FLST	FIDX[6:0]							MSI[1:0]		BIDX[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	FLST	R	0	过滤器列表 指示匹配的过滤器列表。 0：标准过滤器列表

				1：扩展过滤器列表
14:8	FIDX[6:0]	R	7’b0	过滤器索引 匹配的过滤器元素的索引。范围为 0 到 FDCAN_SIDFC.LSS 减 1 或 FDCAN_XIDFC.LSE 减 1。
7:6	MSI[1:0]	R	2’b0	消息存储器指示符 00：未选择 FIFO 01：FIFO 消息丢失 10：消息存储在 FIFO0 中 11：消息存储在 FIFO1 中
5:0	BIDX[5:0]	R	6’b0	缓存区索引 存储消息的接收 FIFO 元素的索引。仅当 MSI[1]=1 时有效。

36.5.23. FDCAN 新数据 1 寄存器（FDCAN_NDAT1）

地址偏移：0x098

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ND31-0	RW	32’b0	新数据 该寄存器保存接收缓冲区 0 到 31 的新数据标志。当接收帧更新接收缓冲区时，对应的标志位会置 1。在主机将标志位清零之前，标志位保持置 1 状态。通过向对应位写 1 将标志位清零，写入 0 不起任何作用。硬复位会将寄存器清零。 0：接收缓冲区未更新 1：接收缓冲区通过新消息更新

36.5.24. FDCAN 新数据 2 寄存器（FDCAN_NDAT2）

地址偏移：0x09C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	ND63-32	RW	32’b0	新数据 该寄存器保存接收缓冲区 32 到 63 的新数据标志。当接收帧更新接收缓冲区时，对应的标志位会置 1。在主机将标志位清零之前，标志位保持置 1 状态。通过向对应位写 1 将标志位清零，写入 0 不起任何作用。硬复位会将寄存器清零。 0：接收缓冲区未更新 1：接收缓冲区通过新消息更新

36.5.25. FDCAN 接收 FIFO0 配置寄存器（FDCAN_RXF0C）

地址偏移：0x0A0

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F0OM	F0WM[6:0]							Res.	F0S[6:0]						
RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F0SA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31	F0OM	RW	0	接收 FIFO0 工作模式 FIFO0 可工作在阻塞模式（Blocking mode）或覆盖模式（Overwrite mode） 0：接收 FIFO0 为阻塞模式 1：接收 FIFO0 为覆盖模式
30:24	F0WM[6:0]	RW	0	接收 FIFO0 水位线 0：禁止接收 FIFO0 水位线中断 1~64：接收 FIFO0 水位线中断（FDCAN_IR.RF0W）的级别 ＞64：禁止接收 FIFO0 水位线中断
23	Reserved	-	-	保留
22:16	F0S[6:0]	RW	0	接收 FIFO0 大小 0：无接收 FIFO0 1~64：接收 FIFO0 元素数量 ＞64：大于 64 的设定值被解析为 64 接收 FIFO0 元素的索引为 0 到 F0S 减 1
15:2	F0SA[15:2]	RW	0	接收 FIFO0 起始地址 消息 RAM 中接收 FIFO0 的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

36.5.26. FDCAN 接收 FIFO0 状态寄存器（FDCAN_RXF0S）

地址偏移：0x0A4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	RF0L	F0F	Res.	Res.	F0PI[5:0]					
						R	R			R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	F0GI[5:0]						Res.	F0FL[6:0]						
		R	R	R	R	R	R		R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	RF0L	R	0	接收 FIFO0 消息丢失 该位是中断标志 FDCAN_IR.RF0L 的副本。当 FDCAN_IR.RF0L 被复位时，该位也被复位 0：接收 FIFO0 无消息丢失 1：接收 FIFO0 有消息丢失；对空间大小为 0 的接收 FIFO0 进行写操作时，该位也会置位 注：

				当 FDCAN_RXF0C.F0OM=1 时，覆盖最早的消息，不会置位该标志位。
24	F0F	R	0	接收 FIFO0 满 0：接收 FIFO0 未满 1：接收 FIFO0 已满
23:22	Reserved	-	-	保留
21:16	F0PI[5:0]	R	6’b0	Rx FIFO 0 写入索引 Rx FIFO 0 写入索引指针，范围为 0 到 63。
15:14	Reserved	-	-	保留
13:8	F0GI[5:0]	R	6’b0	接收 FIFO0 获取索引 接收 FIFO0 读取索引指针，范围 0~63
7	Reserved	-	-	保留
6:0	F0FL[6:0]	R	7’b0	接收 FIFO0 填充级别 接收 FIFO0 中存储的元素数量，范围 0~64

36.5.27. FDCAN 接收 FIFO0 确认寄存器（FDCAN_RXF0A）

地址偏移：0x0A8

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F0AI[5:0]					
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5:0	F0AI[5:0]	RW	6’b0	接收 FIFO0 确认索引 主机从接收 FIFO0 读取消息或消息序列后，必须将从接收 FIFO0 中读取的最后一个元素的缓冲区索引写入 F0AI。此操作会将接收 FIFO0 获取索引 FDCAN_RXF0S.F0GI 设为 F0AI 加 1，并会更新 FIFO0 填充级别 FDCAN_RXF0S.F0FL

36.5.28. FDCAN 接收缓冲区配置寄存器（FDCAN_RXBC）

地址偏移：0x0AC

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RBSA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	0	保留
15:2	RBSA[15:2]	RW	14’b0	接收缓冲区的起始地址 配置消息 RAM 中接收缓冲区的起始地址（32 位地址），也用于引用调试消息 A、 B、 C
1:0	Reserved	-	0	保留

36.5.29. FDCAN 接收 FIFO1 配置寄存器（FDCAN_RXF1C）

地址偏移：0x0B0

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F1OM	F1WM[6:0]								F1S[6:0]						
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1SA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31	F1OM	RW	0	接收 FIFO1 操作模式 FIFO1 可工作在阻塞模式（Blocking mode）或覆盖模式（Overwrite mode） 0：接收 FIFO1 为阻塞模式 1：接收 FIFO1 为覆盖模式
30:24	F1WM[6:0]	RW	7’b0	接收 FIFO1 水印 0：禁止接收 FIFO1 水位线中断 1~64：接收 FIFO1 水位线中断（FDCAN_IR.RF1W）的级别 ＞64：禁止接收 FIFO1 水位线中断
23	Reserved	-	-	保留
22:16	F1S[6:0]	RW	7’b0	接收 FIFO1 大小 0：无接收 FIFO1 1~64：接收 FIFO1 元素数量 ＞64：大于 64 的设定值被解析为 64 接收 FIFO1 元素的索引为 0 到 F1S 减 1
15:2	F1SA[15:2]	RW	14’b0	接收 FIFO1 起始地址 消息 RAM 中接收 FIFO1 的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

36.5.30. FDCAN 接收 FIFO1 状态寄存器（FDCAN_RXF1S）

地址偏移：0x0B4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMS[1:0]		Res.	Res.	Res.	Res.	RF1L	F1F	Res.	Res.	F1PI[5:0]					
R	R					R	R			R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	F1GI[5:0]						Res.	F1FL[6:0]						
		R	R	R	R	R	R		R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:30	DMS[1:0]	R	2’b0	调试消息状态 00：空闲状态，等待接收调试消息 01：已接收调试消息 A 10：已接收调试消息 A、 B 11：已接收调试消息 A、 B、 C
29:26	Reserved	-	-	保留
25	RF1L	R	0	接收 FIFO1 消息丢失

				0：接收 FIFO1 无元素丢失 1：接收 FIFO1 有元素丢失；对空间大小为 0 的接收 FIFO1 进行写操作时，该位也会置位 注： 当 FDCAN_RXF1C.F1OM=1 时，覆盖最早的消息，不会置位该标志位。
24	F1F	R	0	接收 FIFO1 满 0：接收 FIFO1 未满 1：接收 FIFO1 已满
23:22	Reserved	-	-	保留
21:16	F1PI[5:0]	R	6’b0	接收 FIFO1 放入索引 接收 FIFO1 写入索引指针，范围 0~63
15:14	Reserved	-	-	保留
13:8	F1GI[5:0]	R	6’b0	接收 FIFO1 获取索引 接收 FIFO1 读取索引指针，范围 0~63
7	Reserved	-	-	保留
6:0	F1FL[6:0]	R	7’b0	接收 FIFO1 填充级别 接收 FIFO1 中存储的元素数量，范围 0~64

36.5.31. FDCAN 接收 FIFO1 确认寄存器（FDCAN_RXF1A）

地址偏移：0x0B8

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	F1AI[5:0]					
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5:0	F1AI[5:0]	RW	6’b0	接收 FIFO1 确认索引 主机从接收 FIFO1 读取消息或消息序列后，必须将从接收 FIFO1 中读取的最后一个元素的缓冲区索引写入 F1AI。此操作会将接收 FIFO1 获取索引 FDCAN_RXF1S.F1GI 设为 F1AI 加 1，并会更新 FIFO1 填充级别 FDCAN_RXF1S.F1FL

36.5.32. FDCAN 接收缓冲区和 FIFO 元素大小配置寄存器（FDCAN_RXESC）

地址偏移：0x0BC

复位值： 0x0000 0000

配置接收缓冲区和接收 FIFO 元素的数据字节数。大于 8 字节的数据段仅适用于 FD CAN 模式。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	RBDS[2:0]			Res.	F1DS[2:0]			Res.	F0DS[2:0]		
					RW	RW	RW		RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:8	RBDS[2:0]	RW	3'b0	接收缓冲区数据段大小 000： 8 字节 001： 12 字节 010： 16 字节 011： 20 字节 100： 24 字节 101： 32 字节 110： 48 字节 111： 64 字节
7	Reserved	-	-	保留
6:4	F1DS[2:0]	RW	3'b0	接收 FIFO1 数据段大小 000： 8 字节 001： 12 字节 010： 16 字节 011： 20 字节 100： 24 字节 101： 32 字节 110： 48 字节 111： 64 字节
3	Reserved	-	-	保留
2:0	F0DS[2:0]	RW	3'b0	接收 FIFO0 数据段大小 000： 8 字节 001： 12 字节 010： 16 字节 011： 20 字节 100： 24 字节 101： 32 字节 110： 48 字节 111： 64 字节

36.5.33. FDCAN 发送缓冲区配置寄存器（FDCAN_TXBC）

地址偏移：0x0C0

复位值： 0x0000 0000

仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位均被置 1 时，才能对该寄存器进行写访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	TFQM	TFQS[5:0]						Res.	Res.	NDTB[5:0]					
	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TBSA[13:0]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	TFQM	RW	0	发送 FIFO/队列模式

				0：发送 FIFO 模式 1：发送队列模式
29:24	TFQS[5:0]	RW	6’b0	发送 FIFO/队列大小 0：无发送 FIFO 或队列 1-32：发送 FIFO 队列的缓冲区大小 ＞32：大于 32 的设定值被解析为 32
23:22	Reserved	-	-	保留
21:16	NDTB[5:0]	RW	6’b0	专用发送缓冲区大小 0：无专用发送缓冲区 1-32：专用发送缓冲区大小 ＞32：大于 32 的设定值被解析为 32
15:2	TBSA[15:2]	RW	14’b0	发送缓冲区起始地址 消息 RAM 中发送缓冲区部分的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

注意：TFQS 和 NDTB 之和不得大于 32。 FDCAN 没有对该设置错误的监测。消息 RAM 中的发送缓冲区部分从专用发送缓冲区开始。

36.5.34. FDCAN 发送 FIFO/队列状态寄存器（FDCAN_TXFQS）

地址偏移：0x0C4

复位值： 0x0000 0000

发送 FIFO 或队列的状态与寄存器 FDCAN_TXBRP 中挂起的发送请求有关。因此，由于正在运行的发送扫描（TXBRP 尚未更新），添加或取消请求的效果可能会延迟。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TFQF	TFQPI[4:0]				
										R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	TFGI[4:0]					Res.	Res.	TFFL[5:0]					
			R	R	R	R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21	TFQF	R	0	发送 FIFO/队列满 0：发送 FIFO 或队列未满 1：发送 FIFO 或队列已满
20:16	TFQPI[4:0]	R	5’b0	发送 FIFO/队列放入索引 发送 FIFO 或队列的放入索引指针，范围 0-31
15:13	Reserved	-	-	保留
12:8	TFGI[4:0]	R	5’b0	发送 FIFO 获取索引 发送 FIFO 的读取索引指针，范围 0-31。若已配置为发送队列模式（FDCAN_TXBC.TFQM=1），则该位读取值为 0
7:6	Reserved	-	-	保留
5:0	TFFL[5:0]	R	6’b0	发送 FIFO 空闲级别 从 TFGI 开始的连续空闲的发送 FIFO 元素数量，范围 0-32。若已配置为发送队列模式（FDCAN_TXBC.TFQM=1），则该位读取值为 0

注：

在专用发送缓冲区与发送 FIFO 或发送队列组合配置的情况下，放入索引和获取索引是指，从第一个专用发送缓冲区开始的发送缓冲区编号。

例如：对于有 12 个专用发送缓冲区和 20 个发送 FIFO 缓冲区的组合配置，放入索引 15 指向发送 FIFO 的第 4 个缓冲区。

36.5.35. FDCAN 发送缓冲区元素大小配置寄存器（FDCAN_TXESC）

地址偏移：0x0C8

复位值： 0x0000 0000

配置发送缓冲区元素的数据字节数。大于 8 字节的数据段仅适用于 FD CAN 模式。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TBDS[2:0]		
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:0	TBDS[2:0]	RW	3’b0	发送缓冲区数据段大小 000： 8 字节 001： 12 字节 010： 16 字节 011： 20 字节 100： 24 字节 101： 32 字节 110： 48 字节 111： 64 字节

注：如果发送缓冲区元素的数据长度（由 DLC 配置），大于 TBDS 配置的数据长度，则发送缓冲区中未定义的数据字节将会以 0xCC（填充字节）发送。

36.5.36. FDCAN 发送缓冲区请求挂起寄存器（FDCAN_TXBRP）

地址偏移：0x0CC

复位值： 0x0000 0000

配置发送缓冲区元素的数据字节数。大于 8 字节的数据段仅适用于 FD CAN 模式。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	TRP31-0	R	0	发送请求挂起标志 每个发送缓冲区都有自己的发送请求挂起位。这些位通过寄存器 FDCAN_TXBAR 置位；在请求的发送已完成，或请求的发送已通过寄存器 FDCAN_TXBCR 取消后复位。 只有 FDCAN_TXBC 配置了的发送缓冲区对应的 TXBRP 位才会置位。 TXBRP 位置位后会启动发送扫描（见章节【发送处理】），以检查优先级最高的挂起的发送请求（即消息 ID 最小的发送缓冲区）。

				取消发送请求会复位相应的 TXBRP 位。如果请求取消时发送已经开始，则无论发送是否成功，在发送结束时 TXBRP 位都会被复位。TXBRP 位复位后，对应的取消请求位也会立即复位。 请求取消后，以下情况通过寄存器 FDCAN_TXBCF 指示取消已完成： • 发送成功及相应的 FDCAN_TXBTO 位置位后 • 取消时发送尚未开始 • 发送因仲裁失败而中止 • 帧发送期间发生错误 在 DAR 模式下，所有发送在失败后都会自动取消。所有失败的发送，对应的 FDCAN_TXBCF 位都会置位。 0：无发送请求挂起 1：发送请求挂起 注： 在发送扫描期间置位的 TXBRP 位，在本次发送扫描期间被忽略，在此期间，如果请求取消这些发送，则发送被立即取消，对应的 TXBRP 位复位。

36.5.37. FDCAN 发送缓冲区添加请求寄存器（FDCAN_TXBAR）

地址偏移：0x0D0

复位值： 0x0000 0000

配置发送缓冲区元素的数据字节数。大于 8 字节的数据段仅适用于 FD CAN 模式。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	AR31 - 0	RW	0	添加请求 每个发送缓冲区都有自己的添加请求位，写 1 置位相应的添加请求位，写 0 无影响。主机可一次设置多个发送缓冲区的发送请求。只有 FDCAN_TXBC 配置了的发送缓冲区对应的 TXBAR 位才会置位。没有运行发送扫描时，这些位立即复位，否则将保持置位到发送扫描完成。 0：未添加发送请求 1：已添加发送请求 注： 如果添加请求的发送缓冲区已有发送请求挂起（FDCAN_TXBRP 寄存器相应位已置位），该添加请求将被忽略。

36.5.38. FDCAN 发送缓冲区取消请求寄存器（FDCAN_TXBCR）

地址偏移：0x0D4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CR31-0	RW	0	取消请求 每个发送缓冲区都有自己的取消请求位，写 1 置位相应的取消请求位，写 0 无影响。主机可一次设置多个发送缓冲区的取消请求。只有 FDCAN_TXBC 配置了的发送缓冲区对应的 TXBCR 位才会置位。这些位保持置位直到对应的 FDCAN_TXBRP 位复位。 0：无取消请求挂起 1：有取消请求挂起

36.5.39. FDCAN 发送缓冲区发送已发生寄存器（FDCAN_TXBTO）

地址偏移：0x0D8

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	TO31-0	R	0	发送已发生 每个发送缓冲区都有自己的发送已发生标志位。当发送缓冲区发送完成， FDCAN_TXBRP 相应位清零时，对应的发送已发生标志位置位。向 FDCAN_TXBAR 相应位写 1 请求新的发送时，对应的发送已发生标志位复位。 0：未进行发送 1：已进行发送

36.5.40. FDCAN 发送缓冲区取消已完成寄存器（FDCAN_TXBCF）

地址偏移：0x0DC

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	CF31-0	R	0	取消已完成 每个发送缓冲区都有自己的取消已完成标志位。当通过 FDCAN_TXBCR 请求的取消已完成，FDCAN_TXBRP 相应位清零时，对应的取消已完成标志位置位。如果 FDCAN_TXBRP 相应位在取消时没有置位，取消已完成标志位会立即置位。向 FDCAN_TXBAR 相应位写 1 请求新的发送时，对应的取消已完成标志位复位。 0：无发送缓冲区取消 1：发送缓冲区已取消

36.5.41. FDCAN 发送缓冲区发送中断使能寄存器（FDCAN_TXBTIE）

地址偏移：0x0E0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	TIE31-0	RW	0	发送中断使能 每个发送缓冲区都有自己的发送中断使能位。 0：禁止发送中断 1：使能发送中断

36.5.42. FDCAN 发送缓冲区取消已完成中断使能寄存器（FDCAN_TXBCIE）

地址偏移：0x0E4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	CFIE31-0	RW	0	取消已完成中断使能 每个发送缓冲区都有自己的取消已完成中断使能位。 0：禁止取消已完成中断 1：使能取消已完成中断

36.5.43. FDCAN 发送事件 FIFO 配置寄存器（FDCAN_TXEFC）

地址偏移：0x0F0

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	EFWM[5:0]						Res.	Res.	EFS[5:0]					
		RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EFSA[15:2]														Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:24	EFWM[5:0]	RW	6’b0	事件 FIFO 水位线 0：禁止水位线中断 1~32：发送事件 FIFO 水位线中断（FDCAN_IR.TEFW）的级别 ＞32：禁止水位线中断

23:22	Reserved	-	-	保留
21:16	EFS[5:0]	RW	6’b0	事件 FIFO 大小 0：禁止发送事件 FIFO 1-32：发送事件 FIFO 的元素数量 ＞32：大于 32 的设定值被解析为 32 发送事件 FIFO 元素的索引为 0 到 EFS 减 1
15:2	EFSA[15:2]	RW	14’b0	事件 FIFO 起始地址 消息 RAM 中事件 FIFO 的起始地址（32 位字地址）
1:0	Reserved	-	-	保留

36.5.44. FDCAN 发送事件 FIFO 状态寄存器（FDCAN_TXEFS）

地址偏移：0x0F4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	TEFL	EFF	Res.	Res.	Res.	EFPI[5:0]				
						R	R				R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	EFGI[4:0]					Res.	Res.	EFFL[5:0]					
			R	R	R	R	R			R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	TEFL	R	0	发送事件 FIFO 元素丢失 该位是中断标志 FDCAN_IR. TEFL 的副本。当 FDCAN_IR. TEFL 被复位时，该位也被复位 0：发送事件 FIFO 无元素丢失 1：发送事件 FIFO 有元素丢失；对空间大小为 0 的发送事件 FIFO 进行写操作时，该位也会置位
24	EFF	R	0	事件 FIFO 已满 0：发送事件 FIFO 未满 1：发送事件 FIFO 已满
23:21	Reserved	-	-	保留
20:16	EFPI[4:0]	R	5’b0	事件 FIFO 放入索引 发送事件 FIFO 写入索引指针，范围 0~31
15:13	Reserved	-	-	保留
12:8	EFGI[4:0]	R	5’b0	事件 FIFO 获取索引 发送事件 FIFO 读取索引指针，范围 0~31
7:6	Reserved	-	-	保留
5:0	EFFL[5:0]	R	6’b0	事件 FIFO 填充级别 接收 FIFO0 中存储的元素个数，范围 0~32

36.5.45. FDCAN 发送事件 FIFO 确认寄存器（FDCAN_TXEFA）

地址偏移：0x0F8

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EFAI[4:0]				
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4:0	EFAI[4:0]	RW	5'b0	事件 FIFO 确认索引 主机从发送事件 FIFO 读取元素或元素序列后，必须将从事件 FIFO 中读取的最后一个元素的索引写入 EFAI。此操作会将发送事件 FIFO 获取索引 FDCAN_TXEFS.EFGI 设为 EFAI 加 1，并会更新事件 FIFO 填充级别 FDCAN_TXEFS.EFFL。

37. 调试支持(DBG)

37.1. Debug 简介

本芯片基于 Cortex®-M0+ CPU，该 CPU 核包含高级调试功能的硬件扩展。调试扩展允许内核可以在取址（指令断点）或取访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。

调试功能在由调试主机在连接和调试 MCU 时使用，调试的接口是 SWD。在 Cortex®-M0+ CPU 核中的调试功能是一套 ARM CoreSight 设计套件 。

Cortex®-M0+提供了集成的片上调试支持，由以下部分组成：

- SW-DP：串行线
- BPU：断点单元
- DWT：数据观察点触发

调试支持也包括了本芯片的调试集成功能：

- 灵活的调试引脚分配，SWIO@PA4/PB6、SWCLK@PA5/PB5
- MCU 调试盒（支持低功耗模式，控制外设时钟等）

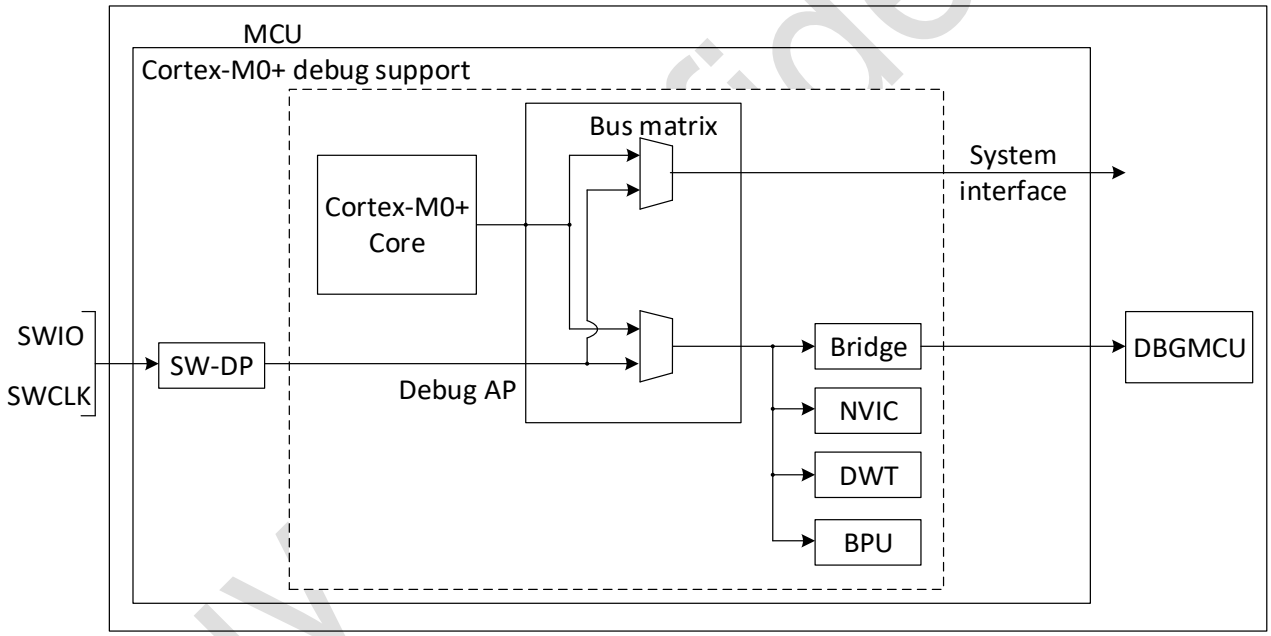


图 37-1 DBG 框图

37.2. 引脚排列和调试端口引脚

37.2.1. SWD 端口

调试功能相关的引脚有两个，可以作为 GPIO 的复用功能。这两个引脚，在封装形式是否可见请参考对应数据手册。

表 37-1 SWD 调试端口

SW-DP 引脚名称	SW 调试断开		引脚分配
	类型	调试分配	
SWDIO	I/O	串行数据输入/输出	PA4/PB6 注
SWCLK	I	串行时钟	PA5/PB5 注

注：支持两组 SWDIO/SWDCLK 的映射，具体描述参见 GPIO 章节。

37.2.2. SW-DP 引脚分频

在芯片复位后（系统复位或者上电复位），用作 SW-DP 的引脚被分配作为专门被调试主机立即使用的引脚。

然而，芯片提供了关闭 SWD 端口的可能性，并释放该引脚作为 GPIO 用。

37.2.3. SWD 引脚上下拉

一旦 SWD I/O 被软件释放，则 GPIO 控制器控制了这两个引脚。GPIO 控制寄存器的复位状态把 IO 置为同等的状态：

- SWDIO：输入上拉
- SWCLK：输入下拉

片内的上拉和下拉电阻为外围节省了增加电阻的需求。

37.3. 调试器 ID

芯片内存放 ID 代码。

芯片上电后，硬件读取 Flash 的出厂配置字节的 0x1FFF 1FF4 地址，装载到 DBG_IDCODE 寄存器中。该寄存器内容可被 SWD 和软件访问。

37.4. SWD 端口

37.4.1. SWD 协议介绍

这是个同步的串行通讯协议，使用以下两个引脚：

- SWCLK：来自主机给芯片的时钟信号
- SWDIO：双向数据信号

利用该协议，可以同时读取和写入两组寄存器组（DPACC 寄存器组和 APACC 寄存器组）。传输数据时， LSB 在前。

对于 SWDIO 双向管理，必须在电路板上对线路进行上拉（ Arm 建议采用 100 kΩ）。

每次在协议中更改 SWDIO 的方向时，都会插入转换时间，此时线路即不受主机驱动也不受目标驱动。默认情况下，此转换时间为一位时间，但可以通过配置 SWCLK 频率来调整。

37.4.2. SWD 协议序列

每个序列由以下阶段组成：

- 主机发送的包请求（8bits）
- 芯片发送的应答响应（3bits）
- 主机或者芯片的数据发送阶段（33bits）

表 37-2 数据包请求(8-bits)

位	名称	说明
0	Start	必须为“1”
1	ApnDP	0：DP 访问 1：AP 访问
2	RnW	0：写请求 1：读请求
4:3	A[3:2]	DP 或者 AP 寄存器的地址区域

5	Parity	以前位的校验位
6	Stop	0
7	Park	没有被主机驱动。由于上拉属性，会被芯片读出 1。

有关 DPACC 和 APACC 寄存器的详细说明，请参见 Cortex®-M0+ TRM。

数据包请求后面始终为转换时间（默认 1 位），此时主机和目标都不会驱动线路。

表 37-3 ACK 响应（3bits）

位	名称	说明
[2:0]	ACK	001：FAULT 010：WAIT 100：OK

如果一个读操作或者如果 1 个 WAIT 或者 FAULT 应答被接收到，则转向时间必须跟随 ACK 响应。

表 37-4 DATA 传输（33bits）

位	名称	说明
[31:0]	WDATA 或者 RDATA	写或者读数据
32	校验位	对[31:0]的奇偶校验位

如果是个读操作时，转向时间必须跟随着数据传输。

37.4.3. SW-DP 状态机 (复位，空闲状态，ID 代码)

SW-DP 的状态机有个定义了 SW-DP 的内部 ID 代码。它遵循 JEP-106 标准。这个 ID 代码是缺省的 ARM 代码，并被置位 0x0BC11477（对应 Cortex®-M0+）。

37.4.4. DP/AP 读写访问

- 对 DP 的读访问还没有发出：可以立即发送目标响应（如果 ACK=OK），也可以延迟发送目标响应（如果 ACK=WAIT）。
- 对 AP 的读访问已经发出。这意味着会在下次传输时返回访问结果。如果要执行的下次访问不是 AP 访问，则必须读取 DP-RDBUFF 寄存器来获取结果。
  每次进行 AP 读访问或 RDBUFF 读请求时都会更新 DP-CTRL/STAT 寄存器的 READOK 标志，以便了解 AP 读访问是否成功。
- SW-DP 有写缓冲区（用于 DP 或 AP 写入），这样即使在其他操作仍未完成时，也可以接受写入操作。如果写缓冲区已满，则目标确认响应为 WAIT。但 IDCODE 读取、CTRL/STAT 读取或 ABORT 写入除外，这几项操作在写缓冲区已满时也会被接受。
- 由于存在异步时钟域 SWCLK 和 HCLK，因此写操作后（奇偶校验位后）还需要两个额外的 SWCLK 周期，以使写入操作在内部生效。应在将线路驱动为低电平时 (空闲状态) 应用这些周期。在写 CTRL/STAT 寄存器以提出一个上电请求时，这一点特别重要。如果下一个操作（在内核上电后才有效的操作）立即执行，这将导致失败。SW-DP 寄存器当 ApnDP=0 时，可以访问这些寄存器。

37.4.5. SW-DP 寄存器

当 APnDP=0 时能够访问这些寄存器。

表 37-5 SW_DP 寄存器

A[3:2]	R/W	CTRLSEL 位或者 SELECT 寄存器	寄存器	注释
00	读	-	IDCODE	制造商代码设置为 Cortex®-M0+ 的默认 Arm 代码：0x0BC11477（标识 SW-DP）
00	写	-	ABORT	
01	读/写	0	DP-CTRL/STAT	目的： – 请求系统或调试上电 – 配置 AP 访问的传输操作 – 控制比较和验证操作 – 读取一些状态标志（上溢和上电确认）
01	读/写	1	WIRE CONTROL	用于配置物理串行端口协议（如转换时间的持续时间）
10	读	-	READ RESEND	使读取的数据能够从损坏的调试器传输中恢复,无需重复原始 AP 传输。
10	写	-	SELECT	用于选择当前访问端口和活动的 4 字寄存器窗口
11	读/写	-	READ BUFFER	由于已发出 AP 访问，因此该读缓冲区非常有用（在执行下个 AP 事务时提供读取 AP 请求的结果）。 此读取缓冲区捕获 AP 中的数据，显示为前一次读取的结果，无需启动新操作

37.4.6. SW-AP 寄存器

当 APnDP=1 时能够访问这些寄存器。

有多个 AP 寄存器，这些寄存器按以下组合进行寻址：

- 移位值 A[3:2]
- DP SELECT 寄存器的当前值

表 37-6 SW_AP 寄存器

地址	A[3:2] 值	描述
0x0	00	保留，保持为复位值
0x4	01	DP CTRL/STAT 寄存器，用作 ■ 请求一个系统或者调试的 power-up ■ 为 AP 访问配置传输操作 ■ 控制比较和验证操作 ■ 读一些状态标志（溢出、power-up 应答）
0x8	10	DP SELECTRION 寄存器：用作选择当前访问端口和 active 4 个 word 的寄存器在窗口。 ■ Bit 31:24：APSEL：选择当前 AP ■ Bit 23:8：保留 ■ Bit 7:4：APBANKSEL：在当前 AP,选择 active 4 个 word 寄存器窗口 ■ Bit 3:0：保留
0xC	11	DP RDBUFF 寄存器：用于提供调试者在一个操作序列后，得到最终的结果（不用请求新的 JTAG-DP 操作）

37.5. 内核调试寄存器

通过内核调试寄存器，可以调试内核。通过调试访问端口访问这些寄存器。由下面四个寄存器组成：

表 37-7 内核调试寄存器

寄存器	说明
DHCSR	32 位调试停止控制和状态寄存器： 此寄存器提供有关处理器状态的信息，能够使内核进入调试停止状态并提供处理器步进功能
DCRSR	17 位调试内核寄存器选择器寄存器： 此寄存器选择需要进行读写操作的处理器寄存器。
DCRDR	32 位调试内核寄存器数据寄存器： 此寄存器保存在寄存器与 DCRSR（选择器）寄存器选择的处理器之间读取和写入的数据。
DEMCR	32 位调试异常和监视控制寄存器： 此寄存器提供向量捕获和调试监视控制。

这些寄存器在系统复位时不复位。它们只能通过上电复位来复位。有关更多详细信息，请参见 Cortex®-M0+ TRM。

为了在复位后立即使内核进入调试停止状态，必须：

- 使能调试和异常监视控制寄存器的位 0 (VC_CORRESET)

使能调试停止控制和状态寄存器的位 0 (C_DEBUGEN)

37.6. BPU (断点单元)

Cortex®-M0+ BPU 实现提供了 4 个断点寄存器。

37.6.1. BPU 功能

处理器断点实现基于 PC 的断点功能。

参考 ARMv6-M ARM 和 ARM Coresight 组件技术参考手册，以获得更多关于 BPU Coresight 的身份寄存器和他们的地址和访问种类。

37.7. DWT (数据观察点)

Cortex®-M0+ DWT 实现提供了 2 个观察点寄存器。

37.7.1. DWT 功能

处理器的断点实现基于 PC 的断点功能。

37.7.2. DWT 程序计数采样寄存器

实现数据观察点单元的处理器，也实现了 ARMv6-M 可选的 DWT 程序计数器采样寄存器(DWT_PCSR)。该寄存器允许调试者周期性的采样 PC，而不用停止处理器。这个机制提供了粗粒度分析。

Cortex®-M0+ DWT_PCSR 记录了通过了条件代码的指令和未通过的指令。

37.8. MCU 调试组件 (DBGMCU)

MCU 调试组件帮助调试者提供以下支持：

- 低功耗模式
- 断点期间的定时器、看门狗的时钟控制
- 控制 I^2C SMBUS 超时功能是否正常

37.8.1. 低功耗模式调试支持

为进入低功耗模式，要执行 WFI 或者 WFE 指令。MCU 进入低功耗模式，或者是将 CPU 时钟停止，或者是减少 CPU 的功耗。

CPU 不允许在调试期间，关闭 FCLK 或者 HCLK。由于这些是调试者连接的需要，在一个调试期间，他们必须保持开启。MCU 集成了特殊的方法，允许用户在低功耗模式下调试软件。

因此，调试者主机必须先置某些调试配置寄存器的内容，以改变低功耗行为：

- 在睡眠模式：FCLK 和 HCLK 仍然有效。相应的，该模式不能引起任何对于标准调试功能的限制。
- 在停止模式：DBG_STOP 位必须被调试者提前置位。

这会使能内部 RC 振荡器时钟（HSI），为停止模式下提供 FCLK 和 HCLK。

37.8.2. 定时器/看门狗/I²C 调试

在一个断点期间，是有必要选择计数器、看门狗和 I²C SMBUS 超时计数器要怎样的行为：

- 可以继续在断点期间计数。例如，这是当一个 PWM 正在控制电机时通常被需要的。
- 可以继续在断点期间停止计数。这是看门狗的特性决定的。
- 可以停止 I²C SMBUS 超时功能。

37.9. DBGMCU 寄存器

37.9.1. DBG ID 寄存器 (DBG_IDCODE)

偏移地址: 0x00

仅支持 32-bit 地址访问，只读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_ID[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBG_ID[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:0	DBG_ID	R	x	仿真器 ID

37.9.2. 调试配置寄存器 (DBG_CR)

该寄存器配置在调试状态下的 MCU 低功耗模式。

该寄存器会被上电复位进行异步复位（不是系统复位）。它可以在系统复位下被调试者进行写操作。

如果调试者主机不支持该功能，对于软件使用者来说，写这些寄存器仍然是可能的。

偏移地址: 0x04

复位值: 0x0000 0000（不会被系统复位进行复位）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STOP	DBG_SLEEP
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	DBG_STOP	RW	0	调试停止模式。

				0：（FCLK=off，HCLK=off）。在停止模式，HCLK 和 FCLK 都会关闭。当从停止模式退出时，时钟配置与上电复位后相同（系统时钟为 HSI）。随后，软件需要重新配置时钟控制器。 1：（FCLK=on，HCLK=on）。当进入停止模式，HSI 不会关闭，FCLK 和 HCLK 由 HSI 产生。当退出停止模式，如果需要改变时钟控制，软件需要重新配置。
0	DBG_SLEEP	RW	0	调试睡眠模式。 0：（FCLK 开，HCLK 关）。在睡眠模式，FCLK 由原先配置好的系统时钟提供，HCLK 关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出后，软件不需要重新配置时钟。 1：（FCLK 开，HCLK 开）。在睡眠模式，FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。

37.9.3. DBG APB 外设冻结寄存器 1 (DBG_APB_FZ1)

该寄存器用来配置 TIMs、RTC、IWDG、WWDG、I2C SMBUS 外设在调试状态下的时钟。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

偏移地址: 0x08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_LPTIM1_STOP	DBG_LPTIM2_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res.	Res.	Res.	Res.	Res.
RW	RW								RW	RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	DBG_TIM6_STOP	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
			RW	RW	RW					RW	RW			RW	RW

复位值: 0x0000 0000

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM1_STOP	RW	0	当 CPU 停止时，LPTIM1 的计数器时钟控制位 0：使能 1：不使能
30	DBG_LPTIM2_STOP	RW	0	当 CPU 停止时，LPTIM2 的计数器时钟控制位 0：使能 1：不使能
29:23	Reserved	-	-	保留
22	DBG_I2C2_SMBUS_TIMEOUT	RW	0	当 CPU 停止时，I²C2 的 SMBUS 超时控制位 0：行为方式与正常模式下相同 1：冻结 SMBUS 超时
21	DBG_I2C1_ SMBUS_TIMEOUT	RW	0	当 CPU 停止时，I2C1 的 SMBUS 超时控制位 0：行为方式与正常模式下相同 1：冻结 SMBUS 超时
20:13	Reserved	-	-	保留
12	DBG_IWDG_STOP	RW	0	当 CPU 停止时，IWDG 计数器的时钟控制位 0：使能

				1：不使能
11	DBG_WWDG_STOP	RW	0	当 CPU 停止时，WWDG 计数器的时钟控制位 0：使能 1：不使能
10	DBG_RTC_STOP	RW	0	当 CPU 停止时，RTC 计数器的时钟控制位 0：使能 1：不使能
9:6	Reserved	-	-	保留
5	DBG_TIM7_STOP	RW	0	当 CPU 停止时，TIM7 计数器的时钟控制位 0：使能 1：不使能
4	DBG_TIM6_STOP	RW	0	当 CPU 停止时，TIM6 计数器的时钟控制位 0：使能 1：不使能
3:2	Reserved	-	-	保留
1	DBG_TIM3_STOP	RW	0	当 CPU 停止时，TIM3 计数器的时钟控制位 0：使能 1：不使能
0	DBG_TIM2_STOP	RW	0	当 CPU 停止时，TIM2 计数器的时钟控制位 0：使能 1：不使能

37.9.4. DBG APB 外设冻结寄存器 2(DBG_APB_FZ2)

该寄存器用来配置 timer 在 debug 下的时钟控制。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

偏移地址: 0x0C

Power on 复位值: 0x0000 0000

仅支持 32-bit 地址访问，只读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_PWM_STOP	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	DBG_TIM1_STOP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
				RW											

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19	DBG_PWM_STOP	RW	0	当 CPU 停止时，PWM 计数器的时钟控制位 0：使能 1：不使能
18	DBG_TIM17_STOP	RW	0	当 CPU 停止时，TIM17 计数器的时钟控制位 0：使能 1：不使能
17	DBG_TIM16_STOP	RW	0	当 CPU 停止时，TIM16 计数器的时钟控制位 0：使能 1：不使能

16	DBG_TIM15_STOP	RW	0	当 CPU 停止时，TIM15 计数器的时钟控制位 0：使能 1：不使能
15:12	Reserved	-	-	保留
11	DBG_TIM1_STOP	RW	0	当 CPU 停止时，TIM1 计数器的时钟控制位 0：使能 1：不使能
10:0	Reserved	-	-	保留

38. 更新历史

版本	日期	更新记录
V0.2	2025.11.12	初始版本

Puya Semiconductor Co., Ltd.