目录

1. 文件约定

1.1. 寄存器缩写表

以下缩写用于寄存器的描述：

read/write(rw)	软件可读/写位。
read-only(r)	软件只读位。
write-only(w)	软件只写位，如果去读此比特位则返回复位值。
read/clear(rc_w1)	软件可读位，同时软件对此比特位写 1 清零，写 0 没有影响（无效）。
read/clear(rc_w0)	软件可读位，同时软件对此比特位写 0 清零，写 1 没有影响（无效）。
read/clear by read(rc_r)	软件可读位，软件读完之后自动清零此比特位，软件写此位没有影响（无效）。
read/set(rs)	软件可读位，同时软件可对此比特位置位，写 0 没有影响（无效）。
Reserved(Res)	保留位，必须保持复位值。

1.2. 有关术语

本章节简要定义了本文中使用的缩略语和缩写：

- Word：32 位数据
- Half-word：16 位数据
- Byte：8 位数据
- SWD-DP（SWD DEBUG PORT）: SWD-DP 提供了一个基于串行线调试（SWD）协议的 2 个引脚的接口（数据和时钟），详细内容参考 ARM Cortx-M0 的技术参考手册。
- IAP（in application programming）:IAP 是在用户程序运行的时候对微控制器的闪存进行编程的能力。
- ICP（in circuit programming）:ICP 是将设备安装在用户的应用板上时，使用 JTAG 协议、SWD 协议或引导加载器对微控制器的闪存进行编程的能力。
- Option bytes：存储在闪存中的产品配置位。
- OBL_LAUNCH：选项字节加载器。
- AHB：先进高性能总线。
- APB：高级外围总线。

1.3. 可用外设

有关所有销售类型的外设可用性和数量，请参考特定的数据。

2. 系统及存储概述

2.1. 系统结构

主系统包括：

- 两个主机：
- ARM Cotex-M0 内核
- 通用 DMA
- 三个从机：
- 内部 SRAM
- 内部闪存
- AHB 连接到 APB 网桥，连接到所有 APB 外设。

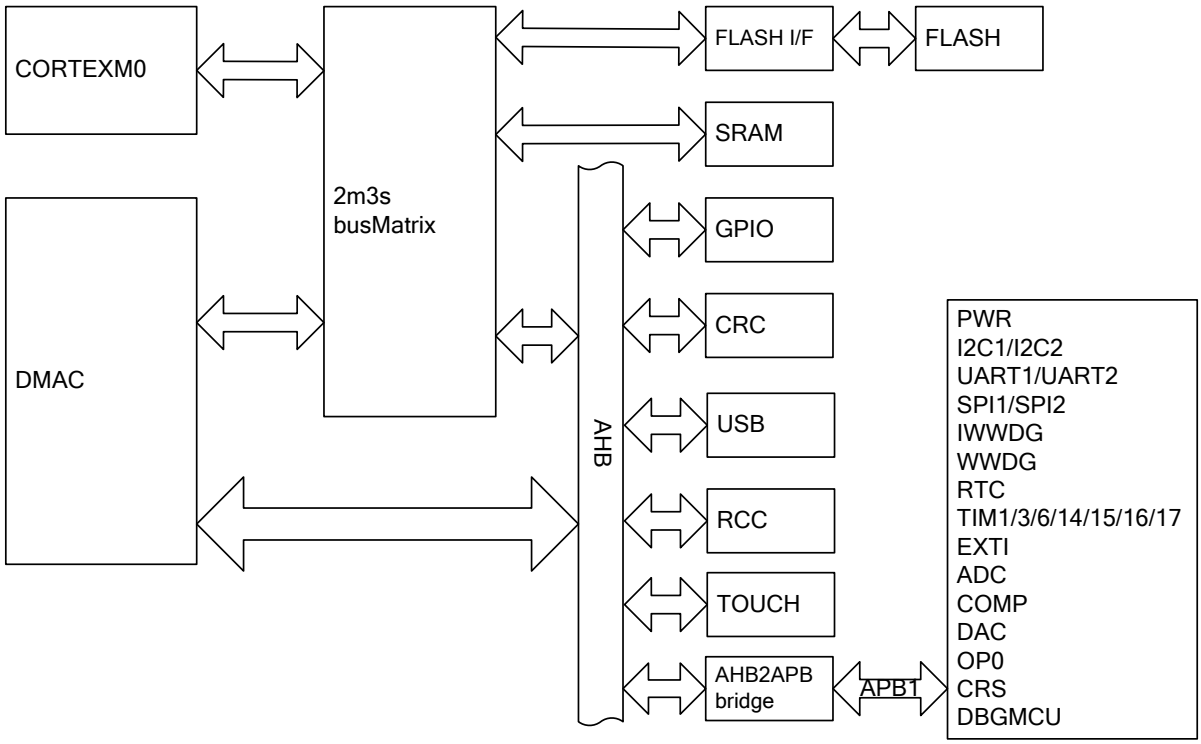


图 2.1 总线架构框图

✧ 系统总线

该总线将 ARM Cotex-M0 内核与外设总线系统连接到一个总线矩阵，该总线管理核心和 DMA 之间的仲裁。

✧ DMA 总线

该总线将 DMA 和 AHB 主接口连接到总线矩阵，该总线矩阵管理 CPU 和 DMA 对 SRAM、Flash 和外设的访问。

✧ 总线矩阵

总线矩阵管理核心系统总线和 DMA 主总线之间的访问仲裁。仲裁使用了轮询调度算法。

✧ AHB 到 APB 网桥

AHB 到 APB 桥提供了 AHB 和 APB 总线之间完全同步的连接。

请参阅 2.2.2 节，连接到此桥的外设的地址映射和内存映射寄存器的边界地址。

在每个设备复位之后，所有的外围时钟被禁用了（除 SRAM 和 Flash）。在使用外围设备之前，必须在 RCC_AHBENR、RCC_APB2ENR 或者 RCC_APB1ENR 寄存器中启用它的时钟。

注意：当在 APB 寄存器上执行 16 位或者 8 位访问时，该访问被转换为 32 位访问，桥接器重复 16 位或者 8 位数据以满足 32 位矢量的需求。

2.2. 存储器组织

2.2.1. 简介

程序存储器、数据存储器、寄存器和 I/O 端口被组织在相同的线性 4 位地址空间中。字节是在内存中以小端格式编码的。字中最低编号的字节被认为是字的低位有效字节，而最高编号的字节被认为是高位有效字节。可寻址内存空间分为 8 个主块，每个块有 512MByte。

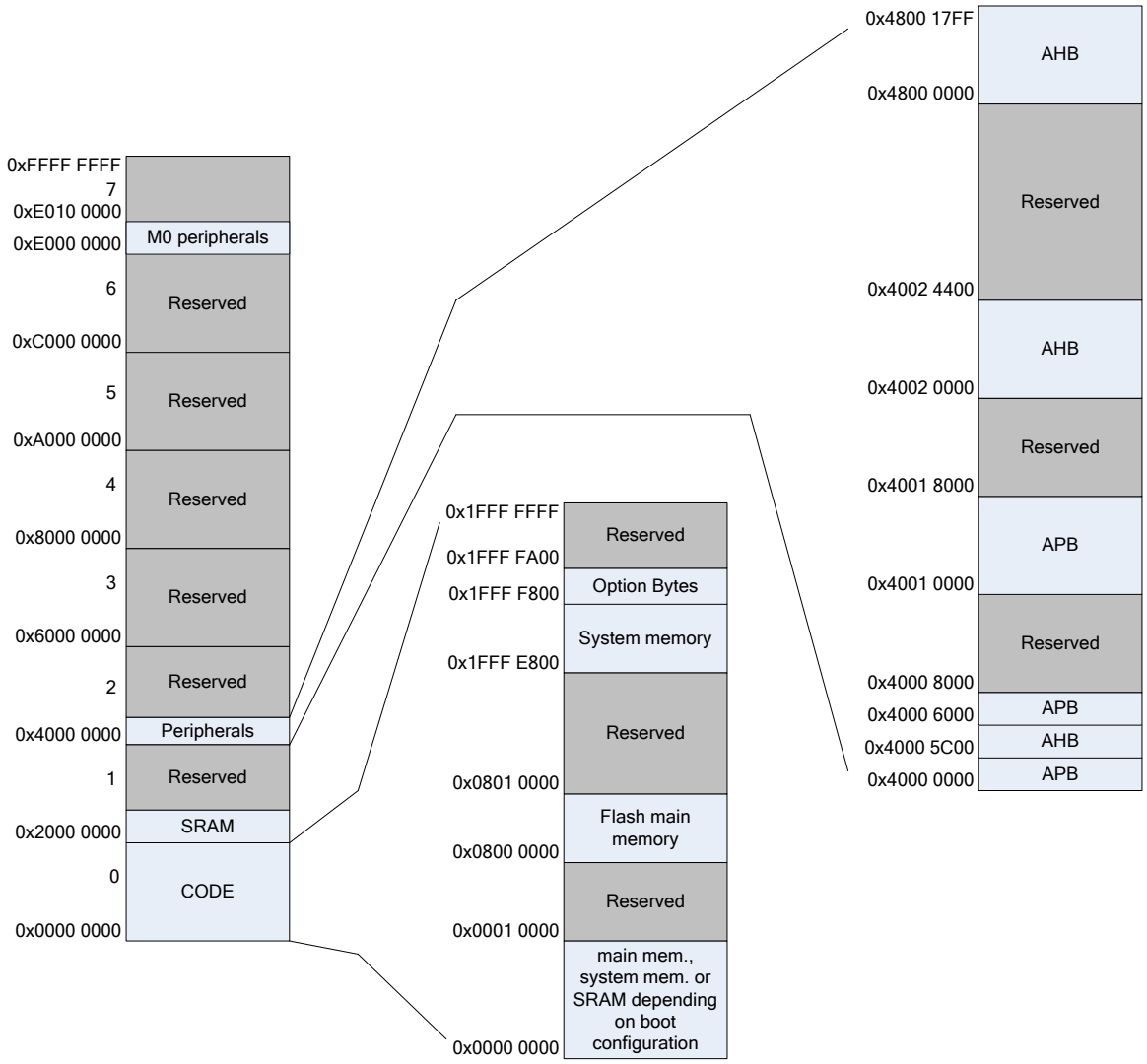


图 2.2 片内存储器映射

所有未分配给片上存储器和外设的内存区域都视为保留。有关可用内存和寄存器区域的详细映射 ,请参阅内存映射和寄存器边界地址和外围部分。

2.2.2. 内存映射和寄存器边界地址

有关内存映射的综合图表，请参见与您设备相对应的数据表

下表给出了设备中可用外设的边界地址。

表 2.1 存储器地址映射

总线	地址范围	大小	外设
AHB	0x4800 1800 - 0x5FFF FFFF	~384 MB	Reserved
	0x4800 1400 - 0x4800 17FF	1 kB	GPIOF
	0x4800 1000 - 0x4800 13FF	1 kB	Reserved
	0x4800 0C00 - 0x4800 0FFF	1 kB	GPIOD
	0x4800 0800 - 0x4800 0BFF	1 kB	GPIOC
	0x4800 0400 - 0x4800 07FF	1 kB	GPIOB
	0x4800 0000 - 0x4800 03FF	1 kB	GPIOA
	0x4002 4400 - 0x47FF FFFF	~128 MB	Reserved
	0x4002 4000 - 0x4002 43FF	1 kB	TSC
	0x4002 3400 - 0x4002 3FFF	3 kB	Reserved
	0x4002 3000 - 0x4002 33FF	1 kB	CRC
	0x4002 2400 - 0x4002 2FFF	3 kB	Reserved
	0x4002 2000 - 0x4002 23FF	1 kB	Flash Interface
	0x4002 1400 - 0x4002 1FFF	3 kB	Reserved
	0x4002 1000 - 0x4002 13FF	1 kB	RCC
	0x4002 0400 - 0x4002 0FFF	3 kB	Reserved
	0x4002 0000 - 0x4002 03FF	1 kB	DMA
APB	0x4001 8000 - 0x4001 FFFF	32 kB	Reserved
	0x4001 5C00 - 0x4001 7FFF	9 kB	Reserved
	0x4001 5800 - 0x4001 5BFF	1 kB	DBGMCU
	0x4001 4C00 - 0x4001 57FF	3 kB	Reserved
	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	Reserved
	0x4001 3800 - 0x4001 3BFF	1 kB	USART1
	0x4001 3400 - 0x4001 37FF	1 kB	Reserved
	0x4001 3000 - 0x4001 33FF	1 kB	SPI1
	0x4001 2C00 - 0x4001 2FFF	1 kB	TIM1
	0x4001 2800 - 0x4001 2BFF	1 kB	Reserved
	0x4001 2400 - 0x4001 27FF	1 kB	ADC
	0x4001 0800 - 0x4001 23FF	7 kB	Reserved
	0x4001 0400 - 0x4001 07FF	1 kB	EXTI
	0x4001 0000 - 0x4001 03FF	1 kB	SYSCFG+COMP+OP0+DAC

	0x4000 7400 - 0x4000 FFFF	35 kB	Reserved
	0x4000 7000 - 0x4000 73FF	1 kB	PWR
	0x4000 6C00 - 0x4000 6FFF	1 kB	CRS
	0x4000 6000 - 0x4000 6BFF	3 kB	Reserved
AHB	0x4000 5C00 - 0x4000 5FFF	1 kB	USB
APB	0x4000 5800 - 0x4000 5BFF	1 kB	I2C2
	0x4000 5400 - 0x4000 57FF	1 kB	I2C1
	0x4000 4800 - 0x4000 53FF	3 kB	Reserved
	0x4000 4400 - 0x4000 47FF	1 kB	USART2
	0x4000 3C00 - 0x4000 43FF	2 kB	Reserved
	0x4000 3800 - 0x4000 3BFF	1 kB	SPI2
	0x4000 3400 - 0x4000 37FF	1 kB	Reserved
	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	Reserved
	0x4000 2000 - 0x4000 23FF	1 kB	TIM14
	0x4000 1400 - 0x4000 1FFF	3 kB	Reserved
	0x4000 1000 - 0x4000 13FF	1 kB	TIM6
	0x4000 0800 - 0x4000 0FFF	2 kB	Reserved
	0x4000 0400 - 0x4000 07FF	1 kB	TIM3
	0x4000 0000 - 0x4000 03FF	1 kB	Reserved
SRAM	0x2000 2000 - 0x3FFF FFFF	~ 512 MB	Reserved
	0x2000 0000 - 0x2000 1FFF	8 kB	SRAM
Flash	0x1FFF FA00 - 0x1FFF FFFF	1.5 kB	Reserved
	0x1FFF F800 - 0x1FFF F9FF	0.5 kB	Option bytes
	0x1FFF E800 - 0x1FFF F7FF	4 kB	Sysem memory
	0x0801 0000 - 0x1FFF E7FF	~ 384 MB	Reserved
	0x0800 0000 - 0x0800 FFFF	64 kB	主程序区
	0x0001 0000 - 0x07FF FFFF	~ 128 MB	Reserved
	0x0000 0000 - 0x0000 FFFF	64 kB	主闪存存储器， 或系统存储器， 或 SRAM，取决于 BOOT 配置

2.3. 内置 SRAM

器件内置了 8k 字节的静态 SRAM.它可以以字节(8 位)、半字(16 位)或字(32 位)进行访问。对于该存储器 ,CPU 及 DMA 都可用最快的系统时钟且不插入任何等待进行访问。

2.4. 闪存存储器概述

闪存存储器有两个不同存储区域：

- 主闪存存储块，它包括应用程序和用户数据区（若需要时）
- 信息块，其包含两个部分：
  - 选项字节（Option bytes）- 内含硬件及存储保护用户配置选项。
  - 系统存储器（System memory）- 其包含 Boot loader 代码。参见第三章嵌入式闪存。

闪存接口基于 AHB 协议执行指令和数据存取。其预取缓冲的功能可加速 CPU 执行代码的速度。通过控制闪存寄存器来执行闪存的编程或者擦除操作。

2.5. 启动配置

在 FT32F0XX 中，可通过 BOOT0 脚及 nBOOT1 bit 的配置选择三种不同的启动模式，如下表所示。

表 2.2 启动模式

Boot模式配置		启动模式
nBOOT1	BOOT0管脚	
x	0	主程序区启动
1	1	系统区启动
0	1	内置SRAM启动

器件复位后，在SYSCLK 的第4 个上升沿锁存启动模式的配置，用户可通过设置启动模式配置来选择启动模式。

从待机模式唤醒时，CPU 会重新采样BOOT0 的引脚值及nBOOT1 bit，因此在有待机应用的场合需要保持启动模式的设置。在启动延迟之后，CPU 从地址0x0000 0000 获取堆栈顶的值，并从启动存储器的0x0000 0004 指示的地址开始执行代码。

根据选定的启动模式，主闪存存储器，系统存储器或SRAM 按照以下的说明访问：

- 从主闪存存储器启动：主闪存存储器被映射到启动存储空间（0x0000 0000），但仍然能从原有的地址空间（0x800 0000）访问。即闪存存储器的内容可从两个地址开始访问，0x0000 0000 或0x800 0000。
- 从系统存储器启动：系统存储器被映射到启动空间（0x0000 0000），但仍然能够在它原有的地址空间（0x1FFF E800）访问。
- 从内置的SRAM启动 :SRAM映射到启动空间(0x0000 0000),但其仍然能够在它原有的地址空间(0x2000 0000 ）访问

物理重新映射

一旦选择了加载模式，应用程序软件就可以修改代码区域中可访问的内存。这个修改是通过在SYSCFG配置寄存器1 (SYSCFG1)中对MEM_MODE位进行编程来执行的。与Cortex M3和M4不同，M0 CPU不支持向量表的重定位。对于位于与0x08000000不同地址的应用程序代码，必须添加一些额外的代码，以便能够服务于应用程序中断。一个解决方案是通过软件将向量表重新定位到内部SRAM:

- 将向量表从 Flash(在应用程序加载地址的基础上映射)复制到 SRAM 的基础地址 0x2000 0000。
- 使用 SYSCFG 配置寄存器 1，在地址 0x0000 0000 处重新映射 SRAM。
- 一旦中断发生，Cortex-M0 处理器将从 SRAM 中重新定位的向量表中获取中断处理程序的起始地址，然后它将跳转到 Flash 中执行中断处理程序。

这个操作应该在应用程序的初始化阶段完成。

嵌入式引导加载程序

嵌入式引导加载程序位于系统内存中，由厂家在生产时写入。该程序可以通过 UART 对闪存进行重新编程。

3. 嵌入式闪存

3.1. 主要特性

- 高达 68kB 闪存存储器
- 存储器结构：
  - 主闪存模块：最大16k字（16k×32位）
  - 信息模块：系统存储器,可被保护,高达4k字节（4kx8位）
  - 用户选项：高达0.5k字节 (0.5kx8位)
  - 工厂配置：高达0.5k字节 (0.5kx8位)
  - FMD访问： 高达3k字节 (3kx8位)

闪存接口的特性为：

- 带预取缓冲器的读接口（每字为4x32位）
- 选择字节加载器
- 闪存编程/擦除操作
- 访问/写保护
- 低功耗模式

3.2. 功能描述

3.2.1. 闪存结构

闪存空间由 64 位宽的存储单元组成，既可以存代码又可以存数据。主闪存块按 128 页（每页 0.5k 字节）或 32 扇区（每扇区 2k 字节）分块。

表 3.1 Flash 模块结构

Flash area	Address	Size(kB)	Name	Description
Main area	0x8000000~0x80001FF	0.5	Page0	Sector0
	0x8000200~0x80003FF	0.5	Page1	
	0x8000400~0x80005FF	0.5	Page2	
	0x8000600~0x80007FF	0.5	Page3	
	0x8000800~0x80009FF	0.5	Page4	Sector1
	0x8000A00~0x8000BFF	0.5	Page5	
	0x8000C00~0x8000DFF	0.5	Page6	
	0x8000E00~0x8000FFF	0.5	Page7	
	…	…	…	…
	0x800F000~0x800F1FF	0.5	Page120	Sector30
	0x800F200~0x800F3FF	0.5	Page121	
	0x800F400~0x800F5FF	0.5	Page122	
	0x800F600~0x800F7FF	0.5	Page123	

	0x800F800~0x800F9FF	0.5	Page124	Sector31
	0x800FA00~0x800FBFF	0.5	Page125	
	0x800FC00~0x800FDFF	0.5	Page126	
	0x800FE00~0x800FFFF	0.5	Page127	
Information area	0x1FFFE800~0x1FFFF7FF	4	-	System memory, MAIN
	0x1FFFF800~0x1FFFF9FF	0.5	-	UserOption

注意：

1.主存储区可以使用页擦除和扇区擦除，其它区只能使用页擦除。

2.全芯片擦除不影响系统存储区和工厂配置区。

3.访问 fuse 区的非物理地址将不产生总线错误。

IAP 代码（boot loader）存放在 System memory 区。

3.2.2. 读操作

嵌入式 Flash 模块可以像普遍存储空间一样直接寻址访问。任何对 Flash 模块内容的读操作都须经过专门的判断过程。取指令和取数据都是通过 AHB 总线读取访问，能够按照 Flash 访问控制寄存器（FLASH_ACR）中的选项所指定的方式执行：

- 取指：预取值缓冲区使能后可提高CPU运行速度
- 潜伏期：等待位的个数，保证正确的读取。

取指

CPU 通过 AHB 总线取指。预取指模块的功效在于提高取指效率。

预取缓冲区

预取缓冲区(4 个 32 位)：在每一次复位以后被自动打开，由于每个缓冲区的大小(32 位)小于闪存的带宽，一次读闪存的操作即可更新两个缓冲区的内容。由于预取缓冲区的存在，CPU 可以工作在更高的主频。CPU 每次取指最多为 32 位的字，取一条指令时，下一条指令已经在缓冲区中等待。

预取控制器

预取控制器会根据预取缓冲区的可用空间来把握访问 Flash 的时机。当预取缓冲区被读走 2 个空间时，预取控制器会发起一次读取请求。复位后，预取指缓冲区的默认状态是打开的。只有在 SYSCLK 低于 24MHz，并且 AHB 时钟没有经过任何分频的条件下（SYSCLK 必须等于 HCLK）才可以开/关预取指缓冲区。通常情况下，预取指缓冲区在初始化过程中就已经决定好开关状态了，而当时 MCU 运行在内部 8MHz 的振荡器下。

访问潜伏期

为了保护对 Flash 的正确读取，必须在 Flash 访问控制寄存器中的 LATENCY [2：0]中指定预取指控制器的速度比，这个数值等于每次访问 Flash 后到下次访问之间所需插入的等待周期的个数。复位后，这个值默认为零，也就是没有插入等待周期的状态

3.2.3. Flash 写和擦除操作

FT32F0XX 可以使用在线编程以及在应用编程。

ICP 是指使用 SWD 或 Boot loader 的方法在线改变 Flash 的内容，将用户代码烧录到单片机中。ICP 提供了一种简单高效的方法，免除了烧写芯片时的芯片装夹等问题。

与 ICP 方法不同的是，IAP（在应用编程）能够使用 MCU 支持的任何通信接口（I/Os，USB，UART，I2C，

SPI，等等）下载程序或者数据。IAP 允许用户在运行程序的过程中重写应用程序，前提是一部分应用程序必须预先用 ICP 的方法烧写进去。

烧写和擦除操作在整个产品工作电压范围内都可以完成。该操作有下列 7 个寄存器完成：

- 主区解锁寄存器（FLASH_KEYR）
- 选项字节关键字寄存器（FLASH_OPTKEYR）
- Flash控制寄存器（FLASH_CR）
- Flash状态寄存器（FLASH_SR）
- Flash地址寄存器（FLASH_AR）
- 选项字节寄存器（FLASH_OBR）
- 写保护寄存器（FLASH_WRPR）

只要 CPU 不去访问 Flash 空间，进行中的 Flash 写操作不会妨碍 CPU 的运行。也就是说，在对 Flash 进行写/擦除操作的同时，任何对 Flash 的访问都会令总线停顿，直到写/擦除操作完成后才会继续执行，这意味着在写/擦除 Flash 的同时不可以对它取指和访问数据。

需要注意的是当 HCLK 低于 125k 时，Flash 无法完成擦写。

对 Flash 空间的解锁

复位后，Flash 存储器默认是受保护状态的，这样可以防范意外的写和擦除动作。FLASH_CR 寄存器不允许被改写，除非执行一串针对 FLASH_KEYR 寄存器的解锁操作才能开启对 FLASH_CR 的访问权限。这串操作由下面 2 个写操作构成：

- 写关键字1 = 0x45670123
- 写关键字2 = 0xCDEF89AB

任何错误的顺序将会锁死 FLASH_CR 直至下次复位。

当发生关键字错误时，会由总线错误引发一次硬件错误中断。KEY1 出错会立即中断，KEY1 正确但 KEY2 错误时会在 KEY2 错的时候引发中断。FLASH_CR 寄存器能被再次锁上使用软件写 FLASH_CR 寄存器的 LOCK 位为 1。

主闪存编程

Flash 一次可以编程 32bit，在开锁且 FLASH_CR.PG 为 1 情况下，CPU 对 Flash 目标地址执行一次字写时，编程启动。编程过程中，Flash 的读操作禁止。需要注意的是，不能对 Flash 发起半字、或者字节的写操作，否则将引发总线错误并产生 hard fault 中断。

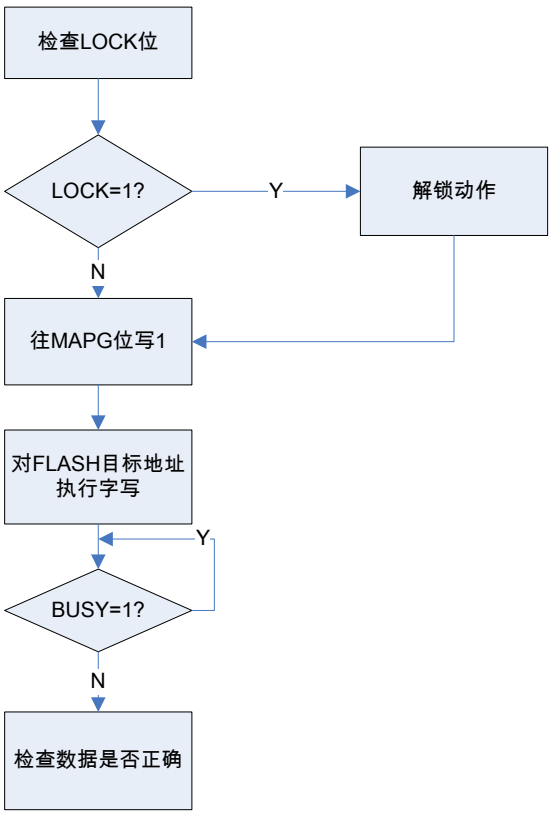


图 3.1 主闪存编程

Flash 存储器接口会预读一下待编程字节后是否为全 1，如果不是，那么编程操作会自动取消，并且在 FLASH_SR 寄存器的 PGERR 位上提示编程错误告警。

如果待编程地址所对应的 FLASH_WRPR 中的写保护位有效，同样也不会有编程动作，同样也会产生编程错误告警。编程动作结束后，FLASH_SR 寄存器中得 EOP 位会给出提示。

主Flash 存储器标准模式下的编程过程如下：

1. 检查FLASH_SR中的BSY 位，以确认上一操作已经结束
2. 置FLASH_CR 寄存器中的PG 位
3. 以半字为单位向目标地址写入数据
4. 等待FLASH_SR 寄存器中的BSY 归零
5. 读数据以校验

注：当FLASH_SR 中得BSY 位为1 的时候，这些寄存器不能写。

Flash存储器擦除

Flash 存储器可以按页为单位擦除，也可以整片擦除。

页擦除

擦除页的步骤如下：

1. 检查FLASH_SR 中的BSY 位，以确认上一操作已经结束
2. 置FLASH_CR寄存器中得PER 位为1
3. 写FLASH_AR寄存器以选择待擦除的页
4. 置FLASH_CR寄存器中的STRT位为1
5. 等待FLASH_SR中的BSY 归零
6. 读取已擦除页以校验

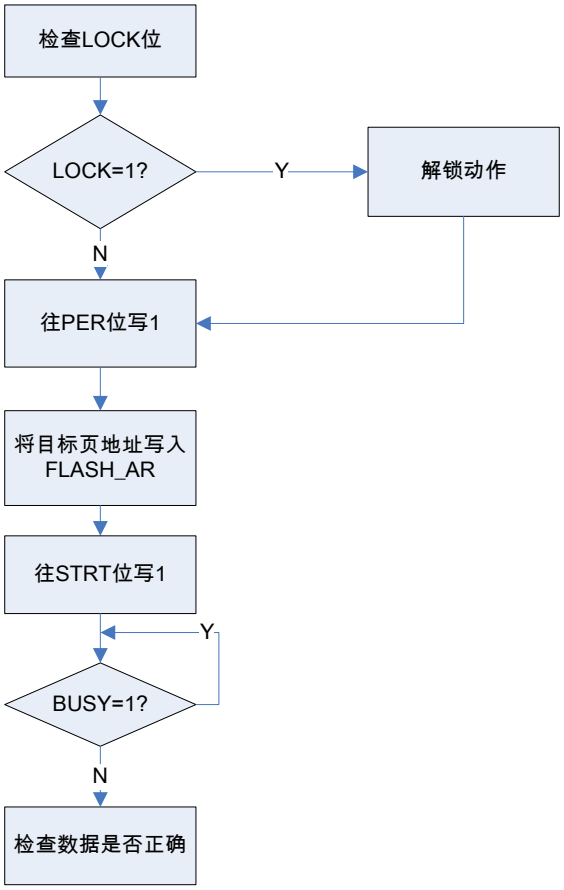


图3.2 页擦除

整片擦除

可以用整片擦除命令一次擦除整个 Flash 用户区，但信息块不会受这个命令影响，具体步骤如下：

1. 检查 FLASH_SR 中的 BSY 位，以确认上一操作已经结束
2. 置 FLASH_CR 寄存器中的 MER 位为 1
3. 置 FLASH_CR 寄存器中的 STRT 位为 1
4. 等待 BSY 位归零
5. 读取全部页并校验

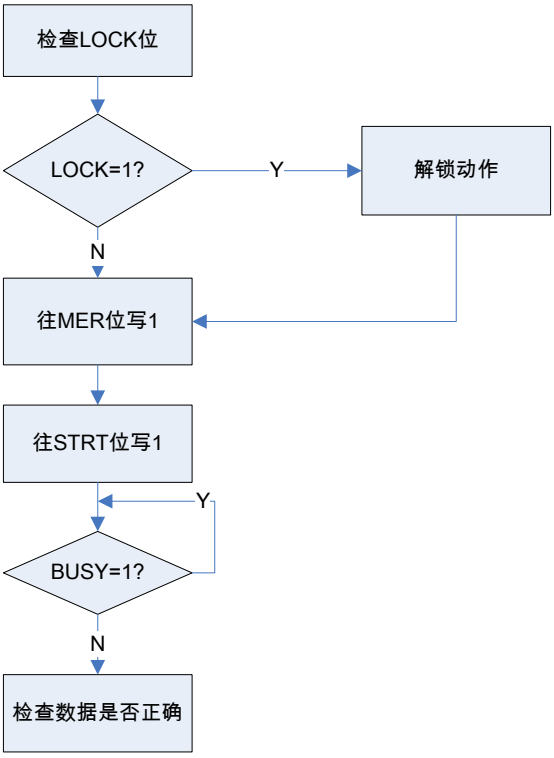


图3.3 片擦除

选项字节编程

选项字节的编程与常规用户地址不同，总共 4 个字节（2 个写保护，1 个读保护，1 个硬件配置）。解除 Flash 访问限制后，还需要对 FLASH_OPTKEYR 寄存器完成关键字写入操作。完成该操作后， FLASH_CR 寄存器中的 OPTWRE 位会被置'1'，然后就可以先置位 FLASH_CR 中的 OPTPG 位，再写目标地址。同样是会自动检查选项字节是否为 1，否则相关操作会被取消并且在 FLASH_SR 中的 WRPRTERR 位提示错误。编程操作结束后，会由 FLASH_SR 寄存器的 EOP 位给出提示。

选项字节为 16 位数据，有效数据为低 8 位，而高 8 位为低 8 位的反码。在编程过程中，硬件会自动将高 8 位设置为低 8 位的反码，保证选项字节的写入值总是对的。步骤如下：

1. 检查FLASH_SR 寄存器中的BSY 位，以确保上一操作结束
2. 解锁FLASH_CR 寄存器中的OPTWRE 位
3. 置FLASH_CR 寄存器中 OPTPG 位为1
4. 写数据到目标地址
5. 等待BSY 位归零
6. 读取并校验

当读保护选项字节由保护状态被改成非保护状态时，会自动引发一次整片擦除。如果用户只想改写其他的字节，则不会引发整片擦除，这个机制用于保护 Flash 的内容。

擦除过程

选项字节的擦除过程如下：

1. 检查FLASH_SR 寄存器中的BSY位，以确保上一操作结束
2. 解锁FLASH_CR 寄存器中的OPTWRE位
3. 置FLASH_CR 寄存器中的OPTER位为1
4. 置FLASH_CR 寄存器中的STRT位为1
5. 等待BSY位归零
6. 读取并校验

3.2.4. 存储保护

可以防范用户区 Flash 区的代码被不可信的代码读出，也可以防范在程序跑飞的时候对 Flash 的意外擦除，写保护的最小单位是一个扇区（4 页）。

1）读保护

这项保护是通过设置 RDP 半字，然后在系统重新上电复位，加载了新的 RDP 后起作用的。

注：如果在设置了读保护时，调试器仍然连接到 JTAG/SWD 接口，需要执行一次上电复位，而不是（没有调试器时的）系统复位。

通过对 RDP 选项设置来启动读保护功能，一共有 3 个保护级别。

表 3.3 读保护级别

RDP	nRDP	保护级别
0xAA	0x55	级别 0（FMD 出厂配置）
除 0xAA 或 0xCC 的其他任何值	除 0x55 或 0x33 外的其他任何值（不要求和 RDP 互补）	级别 1
0xCC	0x33	级别 2

无论处于哪个保护级别，系统存储区总是可读的。其写访问不对用户开放，其擦除、编程操作只能由 FMD 完成。

级别 0 ：无保护

保护功能被禁止，用户可以对主存储区及 UserOption 区进行读，编程和擦除操作。

级别 1：读保护

当 UserOption 区被擦除后，RDP 的值为 0xFF（nRDP 值可以不跟 RDP 互补），Flash 就处于这个级别。处于级别 1 保护时，Flash 的保护行为如下：

- 运行在用户模式下的代码可以不受限制的对主程序区和 UserOption 区进行读写、擦除访问；
- 在调试模式下，或者由 RAM 启动，或者引导区启动的代码均不能访问主程序区；在调试模式下，或者由 RAM 启动可以对 UserOption 区进行访问；由引导区启动的代码只能对 UserOption 区进行擦除操作不能进行读访问和编程操作。即使是任何不正确的读操作均会产生总线错误并引发 hardfault 中断；

任何对主程序区的编程、擦除操作均被禁止，PGERR 标志位置 1；

如果 RDP 字节编程为 0xAA（改变到级别 0），则控制器将自动执行一下片擦操作，待片擦操作完成后，再执行 RDP 的编程；

级别 2：无调试

级别 2 包含了级别 1 所具有的特性，调试功能被禁止。调试接口 SWD，从 RAM 启动或从系统存储区启动均无效。

处于主程序区的代码可以对主程序区执行任何读写、擦除操作，也可以对 UserOption 进行读写访问，但擦除操作是禁止的。

另外，RDP 字节不能被编程，也就是说，不能由级别 2 降级到其它级别。对于级别 2 的器件，FMD 也无法调试和分析。

表 3.4 保护级别下的工作状态

Flash 区	用户代码			调试/RAM 启动/系统区启动		
	Read	Write	Erase	Read	Write	Erase
Main area	Yes	Yes	Yes	No	No	No[4]
	Yes	Yes	Yes	N/A[1]	N/A[1]	N/A[1]

System area[2]	Yes	No	No	Yes	No	No
	Yes	No	No	N/A[1]	N/A[1]	N/A[1]
UserOption	Yes	Yes[3]	Yes	Yes[4]	Yes[3,4]	Yes
	Yes	Yes[5]	No	N/A[1]	N/A[1]	N/A[1]

Note:

1.当处于级别 2 时，调试接口，RAM 启动或者系统区启动均被禁止；

2.无论处于哪个级别和哪种启动模式，系统区都是只读访问（但 FMD 有能力访问）;

3.当由级别 1 改为级别 0 时，主程序区将自动被擦除；

4.处于级别 1 时，引导区不能读写 UserOption，除非把 RDP 写成 0xAA，即把级别 1 改为级别 0，系统将执行一次片擦；

5.除了 RDP 字节，所有 UserOption 均可编程；

2）写保护

写保护以一个扇区为单位（4 页）来控制，配置选项字节中的 WRP 位，随后的系统复位将加载新选项字节就可以使能这个保护。如果试图写入或擦除一个受保护的扇区，会引起 FLASH_SR 中的 WRPRTERR 标志位被置位。

解除保护

解除写保护有下述 2 种情形：

- 情形1：解除写保护，同时解除读保护：
  - 使用闪存控制寄存器（FLASH_CR）的OPTER 位擦除整个选项字节区域；
  - 写入正确的RDP 代码0xA5，允许读访问；这个操作将强制擦除主闪存存储器；
  - 进行系统复位，重装载选项字节（包含新的WRP 字节），写保护被解除。

使用这种方法，将解除整个主闪存模块的写保护。

- 情形2：解除写保护，同时保持读保护有效，这种情况常见于用户自己的实现在程序中编程的启动程序：
  - 使用闪存控制寄存器（FLASH_CR）的OPTER 位擦除整个选项字节区域；
  - 进行系统复位，重装载选项字节（包含新的WRP 字节）,写保护被解除。

选项字节的写保护

默认状态下 ,选项字节块始终是可以读且被写保护。要想对选项字节块进行写操作(编程/擦除)首先要在 OKEY 中写入正确的键序列(与上锁时一样),随后允许对选项字节块的写操作 ,FLASH_CR 寄存器的 OPTWRE 位标示允许写，清除这位将禁止写操作。

3.2.5. 闪存中断

表 3.5 闪存中断

中断事件	事件标志	使能控制位
操作结束	EOP	EOPIE
写保护错误	WRPRTERR	ERRIE
编程错误	PGERR	ERRIE

3.3. 寄存器映射

以下寄存器的基地址为 0x40022000。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	FLASH_ACR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PRFTBS	PRFTBE	LATENCY [3:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	0	0	0	1
0x04	FLASH_KEYR	FKEY[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	FLASH_ OPTKEYR	OPTKEY[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	FLASH_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EOP	WRPRTER	-	PGERR	-	BUSY
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	0	x	0
0x10	FLASH_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OBL_LAUNCH	EOPIE	-	ERRIE	OPTWRE	-	LOCK	STRT	OPTER	OPTPG	-	MER	PER	PG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	0	0	x	1	0	0	0	0	0	0
0x14	FLASH_AR	FAR[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	FLASH_OBR	DATA1[7:0]								DATA0[7:0]								-	-	VDDA_MONITO	nBOOT1	-	nRST_STDBY	nRST_STOP	WDG_SW	-	-	-	-	-	RDPRT[1:0]		OPTERR
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x20	FLASH_WRPR	WRP[31:0]																															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

3.3.1. FLASH_ACR

偏移地址：0x00

复位值：0x0000 0011

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—		PRFTBS	PRFTBE	LATENCY			
类型	RO-0	RO-0	RO	RW	RW	RW	RW	RW

Bit	Name	Function
31:6	NA	保留位，未定义
5	PRFTBS	预取缓存状态 0: 预取缓存已经停用 1: 预取缓存已经启用（当第一个缓存从 Flash 取指完成后自动置 1）
4	PRFTBE	预取缓存使能 0: 不使能预取控制器 1: 使能预取控制器 注意： 上电时配置成从系统区或 SRAM 启动时，PRFTBE 自动清 0； 主程序区切换到系统区或 SRAM 启动时，软件必须将 PRFTBE 清 0 且加入同步隔离指令 ISB； 只要 0 地址映射到系统区或 SRAM，硬件自动把 PRFTBE 位锁住为 0，不能被改写； PRFTBE 由 1 改写为 0 需要加入 ISB 指令；
3:0	LATENCY	Flash 读访问等待周期 ,上电 LATENCY=1 ,上电完后 LATENCY 由软件来配置。 n: Flash 读等待周期为 n 个 HCLK 时钟周期 0: HCLK ≤ 24MHz 1: 24MHz ≤ HCLK ≤ 48MHz 2: 48MHz ≤ HCLK ≤ 72MHz

3.3.2.FLASH_KEYR

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	Function
31:0	FKEY	主 Flash 区解锁寄存器，解锁写序列： 0x45670123 0xCDEF89AB 读操作下，Bit[1:0]返回的是内部状态

Bit[1:0]	Description
00	空闲
01	正确写入了 KEY1
10	正确写入了 KEY1 和 KEY2
11	错误状态

3.3.3.FLASH_OPTKEYR

偏移地址：0x08

复位值：0x0000 0000

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

Bit	Name	Function
31:0	OPTKEY	UserOption 区解锁寄存器，解锁写序列： 0x45670123 0xCDEF89AB 读操作下，Bit[1:0]返回的是内部状态

Bit[1:0]	Description
00	空闲
01	正确写入了 KEY1
10	正确写入了 KEY1 和 KEY2
11	错误状态

3.3.4.FLASH_SR

偏移地址：0x0C
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—		EOP	WRPRTERR	—	PGERR	—	BSY
类型	RO-0	RO-0	RW	RW	RO-0	RW	RO-0	RW

Bit	Name	Function
31:6,3,1	NA	保留位，未定义
5	EOP	编程、擦除操作结束标志 1：成功完成了一次编程或擦除操作，由软件写 1 清 0
4	WRPRTERR	写保护错误标志 1：试图对受写保护的地址编程时由硬件置 1，由软件写 1 清 0
2	PGERR	编程错误标志 1：试图对值为非 0xFFFF FFFF（即未擦除单元）启动编程时由硬件置 1，由软件写 1 清 0
0	BSY	Flash 编程、擦除忙标志 1：Flash 正在做编程或者擦除操作

3.3.5.FLASH_CR

偏移地址：0x10

复位值：0x0000 0040

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		OBL_LAUNCH	EOPIE	—	ERRIE	OPTWRE	—
类型	RO-0	RO-0	RW	RW	RO-0	RW	RW	RO-0
7:0	LOCK	STRT	OPTER	OPTPG	—	MER	PER	PG
类型	RW1	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:14,11,8	NA	保留位，未定义
13	OBL_LAUNCH	UserOption 重载使能 向此位置 1 时将产生一次系统复位及 UserOption 重载 读 1 表示重载过程进行，读 0 表示没有重载
12	EOPIE	EOP 中断使能，高有效
10	ERRIE	Flash 错误中断使能，包括 WRPRTERR 以及 PGERR 中断
9	OPTWRE	UserOption 写使能，由 UserOption 区解锁序列置 1
7	LOCK	Flash 锁定状态 1：Flash 处于锁定状态，开锁序列正确执行后清 0。但当开锁序列出错时，该位将一直置 1，发生系统复位后才能再次解锁 0：开锁状态
6	STRT	擦除启动位 置 1 时启动擦除操作，擦除完成后硬件自动清 0 读返回的是 ERASE 状态
5	OPTER	UserOption 擦除选择
4	OPTPG	UserOption 编程选择
2	MER	主程序区全芯片擦除选择
1	PER	主程序区页擦除选择
0	PG	主程序区页编程选择

3.3.6.FLASH_AR

偏移地址：0x14
复位值：0x0000 0000

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

Bit	Name	Function
31:0	FAR	Flash 地址 擦除操作时由该地址决定页或者扇区，编程操作时由硬件返回上一次编程的地址 注意：当 BSY 位为 1 时，该寄存器不能写访问

3.3.7.FLASH_OBR

偏移地址：0x1C
复位值：0xXXXX XXXX，所有位为只读位

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	DATA1							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	DATA0							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		VDDA_MONITOR	nBOOT1	—	nRST_STDBY	nRST_STOP	WDG_SW
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					RDPRT[1:0]		OPTERR
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
15:14,11,7:3	NA	保留位，未定义
31:24	DATA1	用户数据，其值由 UserOption 的 OPT1 决定： 当 OPT1 的[31:24]与[23:16]互补时 ,DATA1 为 OPT1[23:16] ,否则为 0xFFFF
23:16	DATA0	用户数据，其值由 UserOption 的 OPT1 决定： 当 OPT1 的[15:8]与[7:0]互补时，DATA1 为 OPT1[7:0]，否则为 0xFFFF
13	VDDA_MONITOR	VDDA 监控器使能 0：关闭 VDDA 监控器 1：使能 VDDA 监控器
12	nBOOT1	BOOT 模式配置位
10	nRST_STDBY	Standby 模式复位 0：进入 Standby 模式时自动产生复位 1：Standby 不会产生复位
9	nRST_STOP	STOP 模式复位 0：进入 STOP 时自动产生复位 1：STOP 不会产生复位
8	WDG_SW	独立看门狗 IWDG 使能设置 0：软件看门狗 1：硬件看门狗
2:1	RDPRT	读保护状态 00：保护级别 0，当 OPT0.RDP=0xAA 且 OPT0.nRDP=0x55 时 01 :保护级别 1 ,当 OPT0.RDP 不等于 0xAA 或者不等于 0xCC 时(OPT0.nRDP 的值不起作用) 10：未定义 11：保护级别 2，当 OPT0.RDP=0xCC 且 OPT0.nRDP=0x33 时
0	OPTERR	UserOption 中的 Option byte 错误 当 OPTBx 和 nOPTBx 非互补时，该位置 1，否则清 0 但当 OPTBx 和 nOPTBx 为全 1 时（擦除后的值），该位清 0

注意，bit[15:8]由 OPT0[31:16]决定：

当 OPT0.USER 和 OPT0.nUSER 互补时，FLASH_OBR[15:8]等于 OPT0.USER，否则为 0xFF；

Boot模式配置		引导区
nBOOT1	BOOT0管脚	
x	0	主程序区
1	1	系统区
0	1	内置SRAM

3.3.8.FLASH_WRPR

偏移地址：0x20

复位值：0xXXXX XXXX

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:0	WRP[31:0]							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:0	WRP	Flash 写保护设置，低有效 64kB 由 16 个扇区组成，每个扇区 4kB Bitx= 0：x 扇区的写保护有效 Bitx= 1：x 扇区的写保护无效 当 OPT2 的[15:8]与[7:0]互补时，WRP[7:0]等于 OPT2 [7:0]的值，否则为 0xFF； 当 OPT2 的[31:24]与[23:16]互补时，WRP[23:16]等于 OPT2[[23:16]的值，否则为 0xFF；

4. 选项字节

4.1. 选项字节说明

选项字节共有 8 个字节，由用户根据应用的需要配置；例如：可以选择使用硬件模式的看门狗或软件的看门狗。

在选项字节中每个 32 位的字被划分为下述格式：

位 31~24	位 23~16	位 15~8	位 7~0
选项字节 1 的反码	选项字节 1	选项字节 0 的反码	选项字节 0

选项字节块中选项字节的组织结构如表 4.1 所示。

选项字节可以从表 4.1 列出的存储器地址读出，或从选项字节寄存器（FLASH_OBR）读出。

注：新写入的选项字节（用户的或读/写保护的），在系统复位后才生效。

表 4.1 选项字节表

地址	[31:24]	[23:16]	[15:8]	[7:0]
0x1FFFF800	nUDMY1	UDMY1	nRDP	RDP
0x1FFFF804	nDATA1	DATA1	nDATA0	DATA0
0x1FFFF808	nWRP1	WRP1	nWRP0	WRP0
0x1FFFF80C	nWRP3	WRP3	nWRP2	WRP2
0x1FFFF810	nUSER	USER(1)	nUDMY0	UDMY0

注意：1. 对于 C 版之前芯片，USER 字节位于 0x1FFFF800 的[31:16]，C 版及之后位于 0x1FFFF810。

每次系统复位后，选项字节装载器（OBL_LAUNCH）读出信息块的数据，并保存在选项字节寄存器（FLASH_OBR）中；每个选择位都在信息块中有它的反码位，在装载选择位时反码位用于验证选择位是否正确，如果有任何的差别，将产生一个选项字节错误标志（OPTERR）。当发生选项字节错误时，对应的选项字节被强置为 0xFF。当选项字节和它的反码均为 0xFF 时（擦除后的状态），则关闭上述验证功能。

所有的选择位（不包括它们的反码位）用于配置该微控制器，CPU 可以读选项字节寄存器。

4.2. 寄存器映射

以下寄存器的基地址为 0x1FFFF800。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	User and read protection	nUSER								USER								nRDP								RDP							
										-	-	VDDA_MONITOR	nBOOT1	-	nRST_STOBY	nRST_STOP	WDG_SW																
	production value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
0x04	User data	nData1								Data1								nData0								Data0							
	production value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x08	write protection	nWRP1								WRP1								nWRP0								WPR0							
	production value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x0C	write protection	nWPR3								WRP3								nWRP2								WPR2							
	production value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

4.2.1. 用户和读保护选项

偏移地址：0x00

复位值：0x00FF 55AA

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	nUSER							
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	—	—	VDDA_ MONITOR	nBOOT1	—	nRST_ STDBY	nRST_ STOP	WDG_ SW
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	nRDP							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	RDP							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	nUSER	选项 [23:16] 互补位
23	NA	保留位
22	NA	保留位
21	VDDA_MONITOR	0：关闭 VDDA monitor 1：使能 VDDA monitor
20	nBOOT1	和 BOOT0 管脚一起，决定启动模式

		Boot模式配置 / 引导区 nBOOT1 / BOOT0管脚 x / 0 / 主程序区 1 / 1 / 系统区 0 / 1 / 内置SRAM
19	NA	保留位
18	nRST_STDBY	0：当进入 STANDBY 模式的时候复位产生 1：没有复位产生
17	nRST_STOP	0：当进入 STOP 模式的时候复位产生 1：没有复位产生
16	WDG_SW	0：软件看门狗 1：硬件看门狗
15:8	nRDP	读保护选项反码
7:0	RDP	读出保护选项字节，Flash 这个字节的值定义了 Flash 的保护级别。 0xAA：级别 0 0xXX（除去 0xAA 和 0xCC）：级别 1 0xCC：级别 2 注：读保护级别的状态存储在 FLASH_OBR 寄存器的 RDPRT 位中。

4.2.2. 用户数据选项

偏移地址：0x04

复位值：0x00FF 00FF

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	nData1							
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	Data1							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	nData0							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	Data0							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	nData1	用户数据 1 字节补码
23:16	Data1	用户数据 1 字节值
15:8	nData0	用户数据 0 字节补码
7:0	Data0	用户数据 0 字节值

4.2.3. 写保护选项 1

偏移地址：0x08

复位值：0x00FF 00FF

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	nWRP1							
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	WRP1							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	nWRP0							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	WRP0							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	nWRP1	Flash 存储器写保护选项字节 1 的补码
23:16	WRP1	Flash 存储器写保护选项字节 1 的值
15:8	nWRP0	Flash 存储器写保护选项字节 0 的补码
7:0	WRP0	Flash 存储器写保护选项字节 0 的值

4.2.4. 写保护选项 2

偏移地址：0x0C

复位值：0x00FF 00FF

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	nWRP3							
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	WRP3							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	nWRP2							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	WRP2							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	nWRP3	Flash 存储器写保护选项字节 3 的补码
23:16	WRP3	Flash 存储器写保护选项字节 3 的值
15:8	nWRP2	Flash 存储器写保护选项字节 2 的补码
7:0	WRP2	Flash 存储器写保护选项字节 2 的值

5. CRC 运算单元

5.1. 主要特性

- 采用 CRC32(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 8，16，32 位数据计算
- 可编程的初始值
- 单个 32 位数据输入/输出寄存器
- 输入数据缓存
- 计算 32 比特数据需要 4 个 AHB 时钟
- 通用的 8 位数据寄存器(可以用来存储临时数据)
- 支持输入数据和输出数据的反转处理

5.2. 功能描述

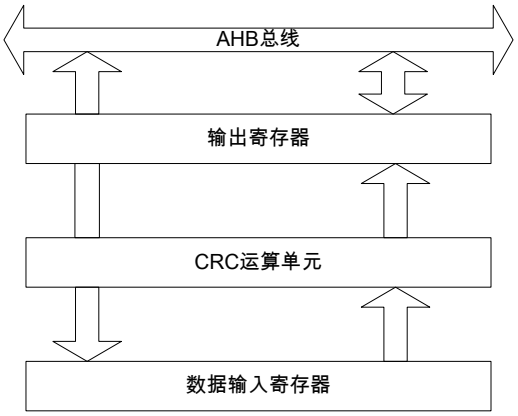


图 5.1CRC 运算单元框图

CRC 计算模块有一个可读写的数据寄存器 CRC_DR，当向 CRC_DR 写入数据时，新的数据就会立即进行运算；在读取 CRC_DR 的值时，返回的是上一次计算后的值；每次写入到这个寄存器中的值都会把当前的值与新写入的值进行运算得到新的值，运算过程是按字节进行运算的，运算完成所需的系统时钟周期数由写入的数据大小决定：运算完成一个字节需要一个 AHB 时钟周期，半字需要两个 AHB 时钟周期，字需要四个时钟周期。

AHB 总线只能按右对齐的方式对 CRC_DR 进行字节，半字，字的写操作；其他寄存器只能进行 32 位的读写操作；对 CRC_DR 寄存器的写操作可以动态的改变写入的数据大小，例如在计算 5 字节的 CRC 运算时，可以先写入 32 位的数据，然后再写入最后一个字节。

CRC 计算单元内部有一个数据缓存区，在当前数据计算未完成前，可以立即写入第二个数据，有效防止 AHB 总线的等待。

输入到数据寄存器的值可以进行 8，16 或者 32 位的反转操作，具体是那一种反转操作由 CRC_CR 寄存器的 REV_IN[1:0]配置决定，例如：要进行 CRC 运算的输入数据是 0x1A2B3C4D，则

- 经过字节反转后送入到 CRC 运算单元的值是：0x58D43C82
- 经过半字反转后送入到 CRC 运算单元的值是：0xD458B23C
- 经过整字反转后送入到 CRC 运算单元的值是：0xB23CD458

输出的数据也可以反转，只要置位 REV_OUT 位即可；例如 CRC 计算单元输出的值是 0x11223344，则经过输出反转后的值是 0x22CC4488。

CRC 运算单元的默认初始值 0xFFFFFFFF，在计算完一帧数据后，可以置位 CRC_CR 寄存器中 RESET 位来恢复 CRC 单元的初始值到 CRC_INIT；CRC 运算单元的初始值可以通过修改 CRC_INIT 的值来实现。

CRC_IDR 是一个独立的 8 位寄存器，其值不受 RESET 位的影响，可以用来存储临时数据。

5.3. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRC_DR	DR[31:0]																															
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x04	CRC_IDR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IDR[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x08	CRC_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	REV_OUT	REV_IN[1:0]		-	-	-	-	RESET
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0
0x10	CRC_INIT	CRC_INIT[31:0]																															
	Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

5.3.1. CRC_DR

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

Bit	Name	Function
31:0	DR	数据寄存器 写入到该寄存器的值会被送到 CRC 计算单元进行计算 读取该寄存器的值是读取到的上次计算的结果

5.3.2. CRC_IDR

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	IDR	通用的 8 位数据寄存器

5.3.3. CRC_CR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							

类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	REV_OUT	REV_IN		—				RESET
类型	RW	RW		RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	REV_OUT	数据寄存器取反输出
6:5	REV_IN	反转输入数据到 CRC 运算单元 00：输入数据不反转 01：输入数据按字节反转 10：输入数据按半字反转 11：输入数据按整字反转
4:1	NA	保留位，未定义
0	RESET	该位复位 CRC 运算单元的值到 CRC_INIT 的值，该位在置 1 后会自动清零 1：复位 CRC 运算单元的值 0：不对 CRC 运算单元的值进行复位

5.3.4. CRC_INIT

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

Bit	Name	Function
31:0	INIT	可编程的 CRC 运算单元的初始值

6. 电源控制

6.1. 电源

FT32F0XX 工作电压为 5V。内置电压调节器提供给内部 1.6V 和 1.5V 数字电路电压。

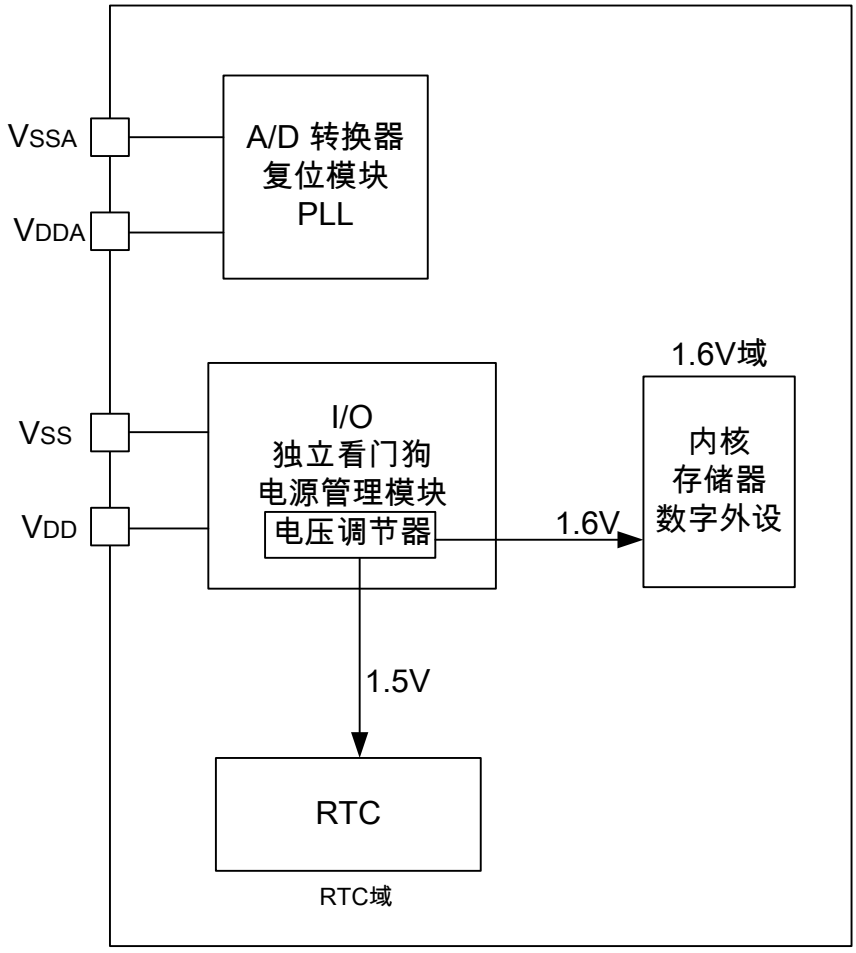


图 6.1 电源供电框图

6.1.1. 电压调节器

FT32F0XX 内部有两个电压调节器，一个 1.6V 电压调节器，给内核、存储器和数字外设供电，另一个 1.5V 电压调节器，给 RTC 域电路供电。

器件复位后 1.6V 电压调节器总是处于打开状态，其根据应用模式有三种工作模式：

- 运行模式：电压调节器以全功耗模式为内核、存储器和数字外设提供 1.6V 电源
- 停止模式（Stop mode）：电压调节器以低功耗模式为内核、存储器和数字外设提供 1.6V 电压
- 待机模式（Standby mode）：电压调节器关闭，内核、存储器和数字外设掉电

6.1.2. 上电复位和掉电复位

器件内置上电复位和掉电复位，当电压低于上电复位电压阈值时，会发生上电复位。如果监控电压一直低于上电复位电压阈值，芯片则会一直处于复位状态。

上电复位只监控 VDD 电源，在上电阶段，VDDA 必须大于或者等于 VDD。
掉电复位会监控 VDD 和 VDDA 电源 。VDDA 掉电检测功能可以被用户禁止，如果在应用中可以保证 VDDA 大于或者等于 VDD 电源，那么 VDDA 掉电检测可以禁止，这样可以降低功耗。

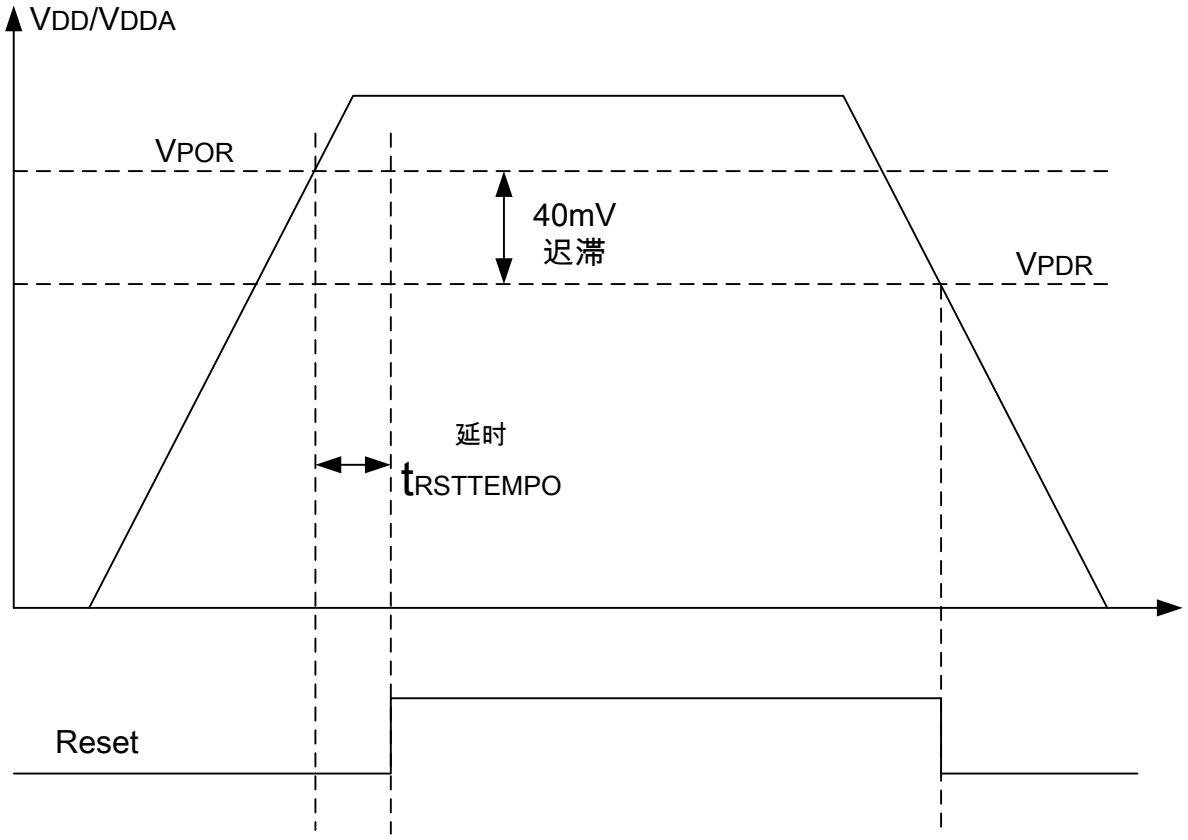


图 6.2 上电复位/掉电复位波形图

6.1.3. 可编程电压检测（PVD）

用户可用 PVD 检测 VDD 电源供电情况 ,通过 PWR_CR 寄存器的 PLS[3:0]位设置 PVD 阈值电压。置位 PVDE 使能 PVD 功能。

PWR_CR 寄存器的 PVDO 位指示 VDD 电压高于或低于 PVD 电压阈值，该事件被连接在 EXTI 16 线，在使能该中断情况下可以产生中断。

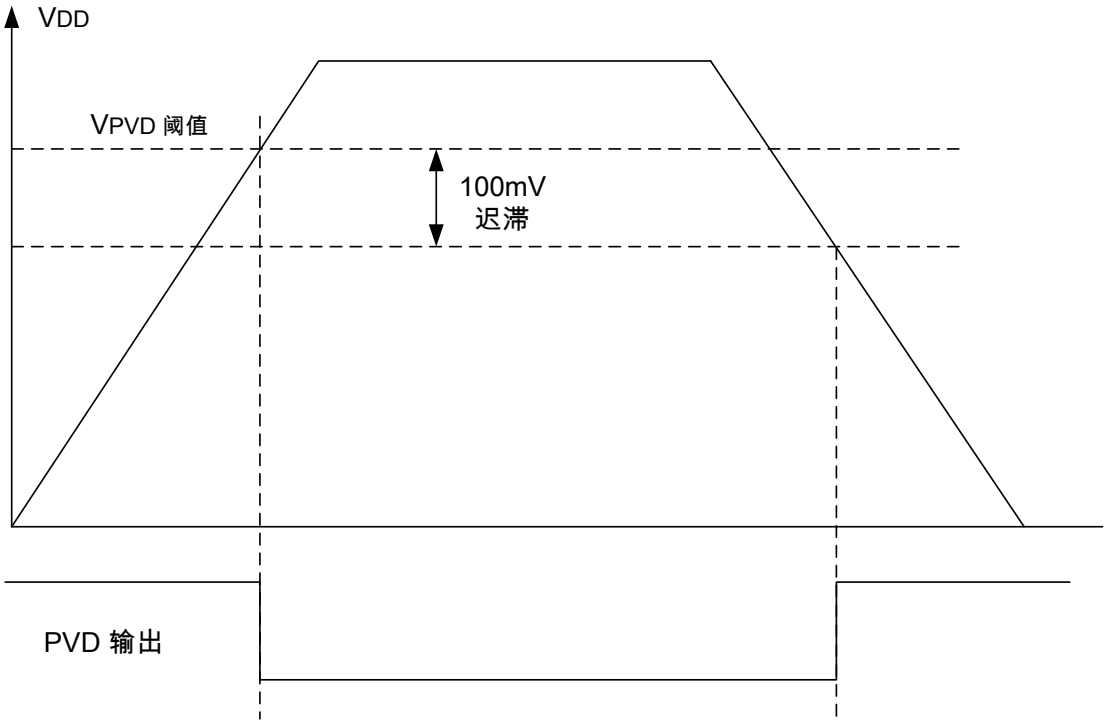


图 6.3 PVD 阈值

6.2. 低功耗模式

默认情况下，在系统复位或者电源复位后微控制器工作于运行模式。当 CPU 不需要继续运行时，可以利用多种低功耗模式来节省功耗。例如等待某个外部事件时，用户需要根据最低电源消耗、最短启动时间和可利用的唤醒源等条件，选定一个最佳的低功耗模式。

该芯片具有三种低功耗模式：

- 睡眠模式（Sleep mode）
- 停止模式（Stop mode）
- 待机模式（Standby mode）

此外，在运行模式下，可以用以下方法降低功耗：

- 降低系统时钟频率
- 关闭未用外设的时钟

表 6.1 低功耗模式

模式	进入	唤醒	对 1.6V 域影响	对 VDD 域影响	电压调节器
睡眠模式	WFI	任意中断	CPU 时钟关闭 其他时钟及模拟时钟无影响	无	打开
	WFE	唤醒事件			
停止模式	PDDS 和 LPDS 位 +SLEEPDEEP 位 +WFI 或 WFE	任意 EXTI 线 （在 EXTI 中使能）	所有 1.6V 域时钟关闭	HSI、HSE、HSI48 和 PLL 时钟关闭	打开或者处于低功耗模式（具体依据 PWR_CR 寄存器）
待机模式	PDDS 位	WKUP 管脚上升			关闭

模式	进入	唤醒	对 1.6V 域影响	对 VDD 域影响	电压调节器
	+SLEEPDEEP 位 +WFI 或 WFE	沿、RTC 闹钟、NRST 外部复位 ,独立看门狗复位			

注意：在使用 WFE 指令时，建议使用 SEV,WFE,WFE 连续的三条指令进入低功耗模式，因为单独一条 WFE 可能无法进入低功耗模式。使用 WFI 指令则只需一条指令进入低功耗模式。

6.2.1. 降低系统时钟频率

在运行模式下，系统时钟可通过编程预分频寄存器（RCC_CFGR）降低系统时钟频率。在进入睡眠模式前这些分频器也可用于降低外设时钟。

6.2.2. 外设时钟门控

在运行模式下，可随时通过停止外设时钟（HCLK 和 PCLK）来减少功耗。

为了在睡眠模式下减少更多功耗，可在执行 WFI 或者 WFE 指令前关闭外设时钟。

外设时钟使能控制位于 RCC_AHBENR、RCC_APB1ENR 和 RCC_APB2ENR 寄存器。

6.2.3. 睡眠模式

进入睡眠模式:
执行 WFI（等待中断）或 WFE（等待事件）指令可让 MCU 进入睡眠模式。依据 Cortex-M0 系统控制寄存器中 SLEEPONEXIT 位，有两种有效选项可用于选择进行睡眠模式进入机制：
Sleep-now：当 SLEEPONEXIT 位被清零，执行 WFI 或 WFE 后，MCU 立刻进入睡眠模式。
Sleep-on-exit：当 SLEEPONEXIT 位被置 1，MCU 从最低优先级的中断服务程序中断后，再进入睡眠模式。

在睡眠模式下，所有 IO 口状态与运行模式一致。

退出睡眠模式：
如果是执行 WFI 指令进入了睡眠模式，任意一个由 NVIC 识别的终端都可以将 MCU 从睡眠模式唤醒。
如果是执行 WFE 指令进入睡眠模式，当任意事件发生后，MCU 退出睡眠模式，唤醒事件可以通过以下方式产生：

- 在外设控制寄存器中使能了外设中断，但未在 NVIC 中使能中断，在 Cortex-M0 系统控制寄存器中置位 SEVONPEND 位。当 MCU 从睡眠中唤醒后 ,外设的挂起标志位和外设的 NVIC 中断标志位必须被清除。
- 配置一个外部或者内部 EXTI 线作为事件模式。当 CPU 从 WFE 唤醒后，因为与事件对应的挂起位未被置起，不必清除外设的中断挂起位或外设的 NVIC 中断通道挂起标志位。

因为没有在中断的进入或者退出上浪费时间 ,所以该模式唤醒所需的时间最短。有关退出睡眠模式的更多细节参考下表。

表 6.2 sleep-now 模式

sleep-now 模式	说明
进入	在以下条件下执行 WFI 或者 WFE： SLEEPDEEP=0 和 SLEEPONEXIT=0
退出	当执行 WFI 进入睡眠模式：中断，参考中断向量表 当执行 WFE 进入睡眠模式：唤醒事件，参考事件管理章节
唤醒延时	无

表 6.3 sleep-on-exit 模式

sleep-on-exit 模式	说明
进入	在以下条件下执行了 WFI 指令： SLEEPDEEP=0 和 SLEEPONEXIT=1
退出	当执行 WFI 进入睡眠模式：中断，参考中断向量表
唤醒延时	无

6.2.4. 停止模式

停止模式是在 Cortex-M0 的深度睡眠的基础上结合了外设的时钟控制机制，在停止模式下，1.6V 电压调节器可运行在正常或者低功耗模式，此时在 1.6V 供电域的所有时钟都被停止，PLL、HSE、HSI、HSI48 都被关闭，SRAM 和寄存器内容被保留。

在停止模式下，所有的 IO 引脚都保持它们在运行模式时的状态。

进入停止模式
为了在停止模式下降低更多功耗，可通过设置 PWR_CR 寄存器的 LPDS 位使 1.6V 的电压调节器处于低功耗模式。

如果正在进行闪存编程，必须等待对存储器访问完成后，系统才进入停止模式。

如果正在进行对 APB 的访问，必须等待对 APB 访问完成后，系统才能进入停止模式。

在停止模式中，可通过对独立的控制位编程来选择如下功能：

- 独立看门狗：独立看门狗可由写看门狗解锁寄存器或者硬件使能来启动。一旦启动无法停止除非进行系统复位。
- 实时时钟（RTC）：通过 RCC_BDCR 寄存器配置。
- 内部低速振荡器 LSI：通过 RCC_CSR 寄存器的 LSION 位配置。
- 外部 32.768kHz 振荡器 LSE：通过 RCC_BDCR 寄存器的 LSEON 位配置。

如果进入停止模式前，ADC 没有被关闭，那么 ADC 仍然可以在停止模式下工作。

退出停止模式
当由一个中断或者事件唤醒 MCU 退出停止模式时，HSI 作为系统时钟。

当电压调节器处于低功耗模式下，系统从停止模式退出时，将会有一段额外的启动延时。如果在停止模式期间保持内部调节器开启，则退出启动时间会缩短，但相应的功耗会增加。

表 6.4 停止模式进入和退出

停止模式	说明
进入	当在以下条件下执行 WFI 或 WFE 指令： SLEEPDEEP=1 PDDS=0 通过设置 LPDS 位选择电压调节器模式 注：为了进入停止模式，所有的 EXTI 挂起标志位，所有外设中断挂起位和 RTC 闹钟标志位必须清除。否则，停止模式进入过程会被忽略并继续执行程序。
退出	如果执行了 WFI 进入停止模式： EXTI 中任意中断线配置为中断模式，并且在 NVIC 中使能了该中断 如果执行了 WFE 进入停止模式： EXTI 中任意线被配置为事件模式
唤醒延时	HSI 唤醒事件+1.6V 电压调节器从低功耗模式唤醒的时间

6.2.5. 待机模式

待机模式可实现系统的最低功耗。该模式在 Cortex-M0 深度睡眠模式时关闭 1.6V 电压调节器。整个 1.6V 供电域被断电，PLL、HSI、HSE、HSI14 和 HSI48 振荡器都会关闭。SRAM 和寄存器内容丢失，除了部分在 VDD 域和 1.5V 供电域的寄存器内容不会丢失。

1.5V 电压调节器可以在进入待机模式前关闭，这样可以降低功耗，如果在进入待机模式前，1.5V 电压调节器未关闭，则进入待机模式后，1.5V 电压调节器正常工作，整个实时时钟（RTC）正常工作。

进入待机模式

在待机模式中，可通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗：独立看门狗可由写看门狗解锁寄存器或者硬件使能来启动。一旦启动无法停止除非进行系统复位。
- 实时时钟（RTC）：通过 RCC_BDCR 寄存器配置。
- 内部低速振荡器 LSI：通过 RCC_CSR 寄存器的 LSION 位配置。
- 外部 32.768kHz 振荡器 LSE：通过 RCC_BDCR 寄存器的 LSEON 位配置。

退出待机模式

当有外部复位、独立看门狗复位、WKUP 管脚的上升沿或者 RTC 事件产生时，MCU 会从待机模式唤醒，此时所有的寄存器都会被复位除了 PWR_CSR 寄存器。

从待机模式唤醒后，程序执行等同于复位后的执行。PWR_CSR 寄存器中 SBF 位用于指示 MCU 曾经进入了

待机模式。

表 6.5 待机模式进入和退出

待机模式	说明
进入	在以下条件下执行 WFI 或 WFE： SLEEPDEEP=1 PDDS=1 清除 WUF 位
退出	WKUP 管脚上升沿 RTC 闹钟事件 外部管脚复位 独立看门狗复位
唤醒延时	与复位相同

在待机模式下，除了以下 IO 外，其余 IO 都处于高阻状态：

- NRST 管脚
- 当被配置为 RTC 或者 LSE 相关功能的 PC13、PC14 和 PC15 管脚
- WKUPx 管脚

默认情况下 ,如果在进行调试 MCU 时 ,使 MCU 进入停止或待机模式 ,将会失去调试连接。这是因为 Cortex-M0 内核失去了时钟。然而，通过设置 DBGMCU_CR 寄存器相应位，可以在低功耗模式下进行调试。

6.2.6. 实时时钟 RTC 唤醒低功耗模式

RTC 可以通过闹钟模式将 MCU 从低功耗模式唤醒。基于此目的，在低功耗模式下，两种 RTC 时钟可以通过 RCC_BDCR 寄存器的 RTCSEL[1:0]位来选择：

- 低功耗 32.768kHz 时钟，该时钟提供一个低功耗且高精度的时间基准。
- 低功耗内部 RC 振荡器（LSI）：使用该时钟源，节省了一个 32.768kHz 晶振成本，但是内部 RC 振荡器会增加少量的电源消耗。

为了用 RTC 闹钟时间唤醒停止模式，必须进行如下操作：

- 配置外部中断线 17 位上升沿触发
- 配置 RTC 使其产生 RTC 闹钟事件

如果要使用 RTC 闹钟从待机模式中唤醒，无需配置外部中断线 17。

6.3. 电源控制模块寄存器映射

offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	PWR_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PLS[3]	DBP	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0
0x04	PWR_CSR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EWUP2	EWUP1	-	-	-	-	-	PVDO	SBF	WUF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	x	x	0	0	0

6.3.1. PWR_CR

偏移地址：0x00

复位值：0x0000 0200

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						PLS[3]	DBP
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
类型	RW	RW	RW	RW	RW-0	RW-0	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9	PLS[3]	PVD 阈值电压设置位，与 PLS[2:0]一起使用
8	DBP	禁止 RTC 域写保护位 0：禁止写入 RTC 域和 RCC_BDCR 寄存器 1：允许写入 RTC 域和 RCC_BDCR 寄存器
7:5	PLS[2:0]	与 PLS[3]组成 PLS[3:0]，PVD 阈值电压选择位 4’b0000：1.78V 4’b0001：1.88V 4’b0010：1.98V 4’b0011：2.08V 4’b0100：2.18V 4’b0101：2.28V 4’b0110：2.38V 4’b0111：2.48V 4’b1000：2.58V 4’b1001：2.68V

		4'b1010：2.78V 4'b1011：2.88V 4'b1100：3.64V 4'b1101：3.08V 4'b1110：3.97V 4'b1111：3.28V
4	PVDE	电源电压检测 PVD 使能位 0：PVD 禁止 1：PVD 打开
3	CSBF	清除待机模式标志位，该位始终读出 0 0：无效 1：清除 SBF 待机标志位
2	CWUF	清除唤醒标志位，该位始终读 0，清除 WUF 位需要两个 HCLK 时钟周期 0：无效 1：清除 WUF 唤醒标志位
1	PDDS	掉电控制位，该位由软件置位或清零 0：当 CPU 进入深度睡眠时，进入停止模式，电压调节器状态由 LPDS 位控制 1：当 CPU 进入深度睡眠时，进入待机模式
0	LPDS	深度睡眠下低功耗控制位，该位由软件置位或清零 0：在停止模式下电压调节器开启 1：在停止模式下电压调节器处于低功耗模式

6.3.2. PWR_CSR

偏移地址：0x04

复位值：0x0000 0000

注意：PWR_CSR 寄存器的写操作最高频率为 48MHz，超过该频率将不保证写操作的正确性。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						EWUP2	EWUP1
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	—					PVDO	SBF	WUF
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO	RO	RO

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	EWUPx	WKUPx 管脚控制位，x 代表 1、0，这些位由软件置位或清零 0：WKUPx 管脚用于通用 IO 功能，WKUP 引脚上的事件不能将 MCU 从待机

		模式唤醒 1：WKUPx 管脚用于将 MCU 从待机模式唤醒，WKUPx 引脚被强制置为输入下拉配置（WKUPx 管脚上的上升沿将 MCU 从待机模式唤醒）
7:3	NA	保留位
2	PVDO	PVD 输出，该位只能由硬件置位或清零。它仅在 PVDE 为 1 时有效 0：VDD 高于 PVD 设置的阈值电压 1：VDD 低于 PVD 设置的阈值电压
1	SBF	待机模式标志位，该位由硬件置位，只能由置 CSBF 位或者 POR/PDR 清零 0：系统不再待机模式 1：系统进入待机模式
0	WUF	唤醒标志位，该位由硬件置位，系统复位清零，退出待机模式不清零 0：无唤醒事件发生 1：有来自 RTC 闹钟或者 WKUPx 管脚的唤醒源 注：当 WKUPx 引脚已经是高电平时，在通过设置 EWUPx 位使能 WKUPx 管脚时，会检测到一个唤醒事件

7. 复位和时钟

7.1. 复位

FT32F0XX 内部由三个复位，分别是系统复位、电源复位以及 RTC 域复位。

7.1.1. 电源复位

电源复位包括两种情况：

1. 上电复位和掉电复位
2. 退出 Standby 模式复位
3. VDDA 掉电复位

7.1.2. 系统复位

系统复位包括以下几种复位源：

1. NRST 管脚上低电平
2. 窗口看门狗复位
3. 独立看门狗复位
4. 软件复位
5. 低功耗管理复位
6. 配置字加载复位
7. 电源复位

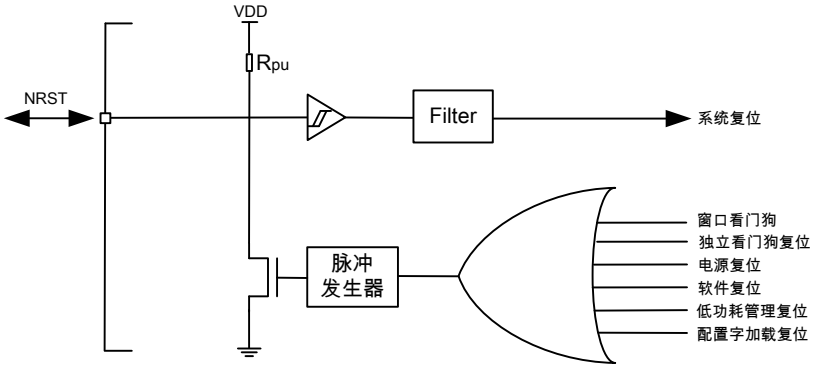


图 7.1 复位框图

复位标志位记录在 RCC_CSR 寄存器中，通过检查 RCC_CSR 寄存器可以判断哪个复位源引起了系统复位。低压域上电复位和掉电复位均不会清除复位标志位 ,高压域复位会清除除了其自身标志位外的其他复位标志位。

内部复位源引起系统复位时，NRST 管脚上会产生大于 20μs 的低脉冲信号。高压域上电复位、掉电复位和 VDDA 掉电复位都会在 NRST 管脚上产生大于 20us 低脉冲信号，如果上述复位信号一直有效，则 NRST 管脚一直为低电平。复位服务程序向量地址为 0x0000_0004。

软件复位：
将 ARM Cortex-M0 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1，则会发生系统复位。

低功耗管理复位可以通过两种方式产生：
1. 当进入 Standby 模式时产生复位：在用户配置字中配置 nRST_STDBY 位使能该复位功能，当使能了该复位时，进入 Standby 模式的序列被完整执行后，芯片不会进入 Standby 模式，而是发生系统复位。
2. 当进入 Stop 模式时产生复位 :在用户配置字中配置 nRST_STOP 位使能该复位功能 ,当使能了该复位时，进入 Stop 模式的序列被完整执行后，芯片不会进入 Stop 模式，而是发生系统复位。

配置字加载复位：
FLASH_CR 寄存器 OBL_LAUNCH 位置 1，则会发生配置字加载复位，该复位发生后，系统会重新加载配置字。

7.1.3. RTC 域复位

RTC 域复位只影响 RTC 和 RCC_BDCR 寄存器，RTC 域复位产生方式：
1. 软件置位 RCC_BDCR 寄存器中的 BDRST 位
2. 上电复位或者掉电复位

7.2. 时钟

FT32F0XX 中有以下几个时钟源可以作为系统时钟：
1. 内部 8MHz 振荡器（HSI）
2. 内部 14MHZ 振荡器（HSI14）
3. 内部 48MHz 振荡器（HSI48）
4. PLL 时钟
5. 外部高速晶振或者外部高速时钟源 HSE

芯片内置 40kHz 低速振荡器（LSI），用于独立看门狗和 RTC 模块。32.768kHz 低速外部晶振(LSE)用于 RTC 模块和 USART1 模块，14MHz 内部高速时钟可用于 ADC 转换时钟。

外设时钟使用情况：
1. Flash 模块编程时钟为 HCLK
2. ADC 模块时钟可选内部 14MHz 时钟和 APB 时钟的 2 分频或者 4 分频
3. USART1 模块时钟可选系统时钟、HSI、LSE 和 APB 时钟（PCLK）
4. I2C1 模块时钟可选系统时钟和 HSI
5. USB 模块时钟可选 HSI48 和 PLL
6. RTC 模块可选 LSE、LSI 或者 HSE 的 32 分频
7. 独立看门狗时钟为 LSI 时钟
8. Timerx 时钟频率自动被硬件确定，当 APB 分频比为 1 时，Timerx 时钟是 APB 时钟，否则 Timerx 时钟是 APB 时钟的 2 倍频

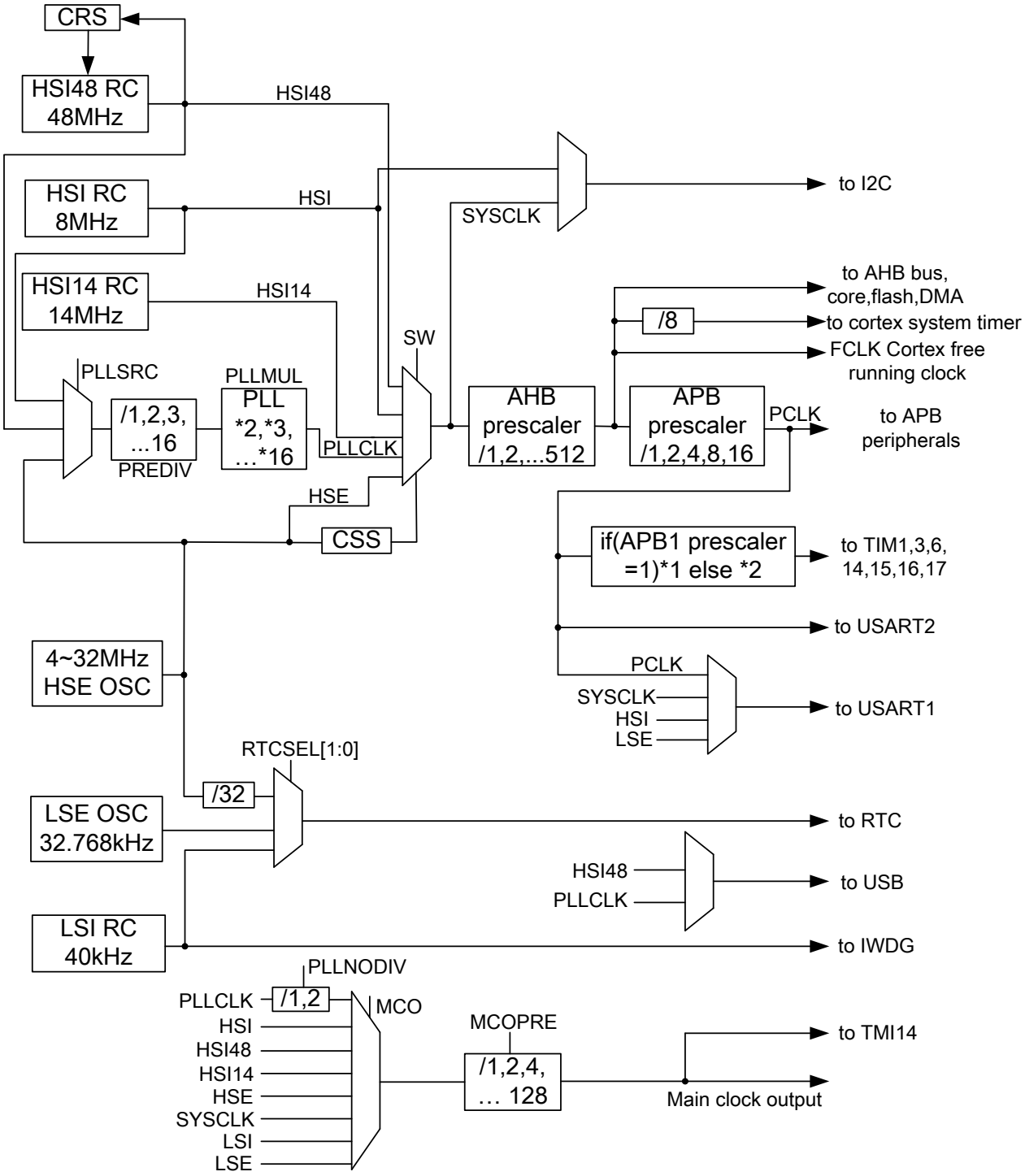


图 7.2 时钟框图

7.2.1. HSE 时钟

高速外部时钟产生方式有两种：

1. 外接高速晶振
2. 外部直接灌入时钟

当外部直接输入时钟时，OSC_IN 管脚作为时钟输入端，OSC_OUT 可以作为普通的 GPIO 使用；当外部接高速晶振时，OSC_IN 和 OSC_OUT 管脚都必须使用，同时还需要在外部接上电容。
外部接入晶振时，晶振频率范围可以为 4~32MHz。

用户通过配置 RCC_CR 寄存器中的 HSEON 打开或者关闭 HSE 时钟。

通过 RCC_CR 寄存器中的 HSERDY 位，可以判断外部晶振是否起振并且稳定，在启动阶段，HSE 不会被使用，等到 HSERDY 被置高后，才能使用 HSE 时钟。如果在 RCC_CIR 寄存器中使能了 HSE 时钟起振稳定中断，则可以产生中断。

注意：当 HSE 起振后，关闭 HSE 时钟需要等待 6 个 HSE 时钟周期后，才能关闭 HSE 时钟。如果某种原因导致 OSC_IN 管脚无时钟输入时，HSE 振荡器无法被关闭，OSC 管脚功能被锁定，不能用于其他功能，这样会引入不必要的电源消耗，为了避免这种情况，建议在打开 HSE 时钟时，使能时钟缺失检测功能。

使用 HSE 外部时钟输入模式时，需要配置 RCC_CR 寄存器中的 HSEBYP 和 HSEON 位，此模式时外部输入的最高频率为 32MHz。

7.2.2. HSI 时钟

HSI 时钟为内部 8MHz 振荡器产生的时钟，可以直接用于系统时钟和 PLL 的输入。

HSI 拥有功耗低和相当于 HSE 启动时间短的优点，但是 HSI 频率校验后精度相对于外部晶振较低。

由于制造工艺的原因，每个芯片内部振荡器频率有所差异，因此需要对内部振荡器频率进行校准。在芯片上电后，工厂校验值会被加载到 RCC_CR 寄存器的 HSICAL[7:0]，此时 HSI 已经被校准。如果应用中电压或者温度不同，可能导致频率有偏差，用户可以通过配置 RCC_CR 寄存器中 HSITRIM[4:0],对 HSI 进行校正。

通过 RCC_CR 寄存器中 HSIRDY 位，可以判断 HSI 有无稳定，在时钟启动阶段，HSI 不能输出，直到 HSI 稳定，HSI 才能被使用。

HSI 时钟通过 RCC_CR 寄存器中 HSION 打开或者关闭。

7.2.3. PLL 时钟

内部 PLL 输入时钟源为 HSI、HSI48 和 HSE 时钟。PLL 输入时钟，分频系数等设置必须在使能 PLL 之前完成，当 PLL 使能后，PLL 的配置无法改变。
PLL 输入频率最小为 0.8MHz，PLL 输出频率范围 16MHz~72MHz。

修改 PLL 配置。参考以下流程：
1. 清零 PLLON，禁止 PLL 工作
2. 检测 PLLRDY 位，确保 PLL 完全停止
3. 配置 PLL 参数

4. 置位 PLLON，使能 PLL
5. 检测 PLLRDY 位，等待 PLLRDY 位置 1

如果使能了 PLL 稳定中断，则当 PLL 稳定时会产生中断。

7.2.4. LSE 时钟

LSE 为外部低速时钟源。LSE 时钟晶振模式中晶振频率为 32.768kHz，该时钟具有功耗低且精度高等优点，LSE 专门用于 RTC 模块。

通过配置 RCC_BDCR 寄存器的 LSEON 位使能或者关闭 LSE 时钟，LSE 振荡器驱动能力可以通过 RCC_BDCR 寄存器中 LSEDRV[2:0]位进行设置。

当 RCC_BDCR 寄存器的 LSERDY 置高时，表明此时 LSE 时钟已经稳定。在未稳定时间内，LSE 时钟不输出。

如果使能了 LSE 时钟稳定中断，则当 LSE 时钟稳定时会产生中断。

打开 LSE 时钟后，芯片内部计时器计时 4096 个 LSE 时钟周期后，才会给出 LSE 稳定信号。当 LSE 稳定后，软件清零 LSEON，则需要等待 6 个 LSE 时钟周期后才会真正关闭 LSE 时钟。由于某种原因，OSC32_IN 管脚上没有时钟输入，LSE 振荡器无法被关闭，OSC32 管脚被锁定为时钟功能，这样会产生不必要的功耗，唯一解决该情况的办法是通过软件对 RTC 域进行复位。

配置 RCC_BDCR 寄存器 LSEBYP 和 LSEON 位，可以选择外部时钟输入模式。LSE 外部时钟输入模式中输入时钟最高频率为 1MHz，外部输入时钟占空比必须为 50%左右，在该模式下，OSC32_OUT 管脚可以作为普通 GPIO 使用。

7.2.5. LSI 时钟

LSI 时钟为内部低速时钟，其频率为 40kHz。LSI 时钟作为一个低功耗时钟用于 RTC 和独立看门狗模块，LSI 时钟可以工作在 Stop 模式和 Standby 模式。

软件配置 RCC_CSR 寄存器的 LSION 可以打开或者关闭 LSI 时钟。当 RCC_CSR 寄存器的 LSIRDY 置高时。表明此时 LSI 时钟已经稳定。在未稳定时间内，LSI 时钟不输出。

如果使能了 LSI 时钟稳定中断，则当 LSI 稳定后会产生中断。

注意：在程序开始运行的 4ms 内硬件会打开 LSI 时钟，此时 RCC_CSR 寄存器的 LSIRDY 是置高的。若此时软件配置 RCC_CSR 寄存器的 LSION 打开 LSI 时钟，RCC_CIR 寄存器的 LSIRDYF 将无法置高而无法产生中断。建议在上电时的 LSI 时钟稳定判断方式采用轮询 RCC_CSR 寄存器的 LSIRDY 位方式，而不是稳定中断方式。

7.2.6. HSI14 时钟

HSI14 时钟源为内部高速 14MHz 时钟源。其可以作为系统时钟和 ADC 时钟使用，也可以通过 MCO 管脚输出到片外。
软件配置 RCC_CR2 寄存器的 HSI14ON 使能 HSI14 时钟源，检查 RCC_CR2 寄存器的 HSI14RDY 位，可以判断当前 HSI14 时钟有无稳定，当使能了 HSI14 时钟稳定中断时，HSI14 稳定后会发生 HSI14 稳定中断。
HSI14 时钟可以在停止模式下正常运行。

7.2.7. HSI48 时钟

HSI48 时钟源为内部高速 48MHz 时钟，其可以作为系统时钟、USB 模块和 PLL 输入时钟使用，也可以通过 MCO 管脚输出到片外。

HSI48 内部时钟与时钟恢复系统（CRS）一起使用，可以给 USB 模块提供高精度的时钟。HSI48 时钟在 Stop 模式和 Standby 模式下处于关闭状态。软件配置 RCC_CR2 寄存器中 HSI48ON 位使能或者关闭 HSI48 时钟源，检查 RCC_CR2 寄存器的 HSI48RDY 位，可以判断当前 HSI48 时钟有无稳定，当使能了 HSI48 时钟稳定中断时，HSI48 稳定后会产生 HSI48 时钟稳定中断。

7.2.8. 时钟校准

HSI8 时钟校准公式如下：
校准值={~HSICAL[8],HSICAL[7:0]}+(HSITRIM[4:0]-5'h10),式中 HSICAL[8:0]为 HSITRIM[4:0]为默认值时读出的数据。

HSI8 校准值范围 9'h000~9'h1FF，其中 9'h000~9'h0FF 为向变小方向校准，9'h101~9'h1FF 为向变大方向校准，9'h100 不校准。

用户使用软件写 RCC_CR 寄存器 HSITRIM[4:0]位对 HSI8 时钟进行校时，配置的 HSITRIM[4:0]必须满足{~HSICAL[8],HSICAL[7:0]}+(HSITRIM[4:0]-5'h10) <=9'h1ff 或>=9'h000 (式中 HSICAL[8:0]为 HSITRIM[4:0]为默认值时读出的数据)，否则校准不正确。

HSI14 校准公式如下：
校准值={~HSI14CAL[7],HSI14CAL[6:0]}+(HSI14TRIM[4:0]-5'h10)，式中 HSI14CAL[7:0]为 HSI14TRIM[4:0]为默认值时读出的数据。

HSI14 时钟校准值范围 8'h00~8'hFF ,其中 8'h00~8'h7F 为向变小方向校准 ,8'h81~8'hFF 为向变大方向校准，8'h80 不校准。

用户使用软件写 RCC_CR2 寄存器 HSI14TRIM[4:0]位对 HSI14 时钟进行校时，配置的 HSI14TRIM[4:0]必须满足{~HSI14CAL[7],HSICAL[6:0]}+(HSI14TRIM[4:0]-5'h10) <=8'hFF 或>=8'h00(式中 HSI14CAL[7:0]为 HSI14TRIM[4:0] 为默认值时读出的数据）,否则校准不正确。

7.2.9. 系统时钟选择

以下时钟源可以作为系统时钟：

1. HSI
2. HSI14
3. HSE
4. HSI48
5. PLL

系统复位之后，系统时钟源默认为 HSI 时钟，当某个时钟源作为系统时钟时，这个时钟源不能被关闭。只有当目标时钟稳定后，系统时钟才会从当前时钟源切换到目标时钟源。RCC_CFGR 寄存器中 SWS[1:0]和 RCC_CFGR4 寄存器中 SWS[2]表明当前系统时钟所用时钟源。

7.2.10. 时钟安全系统（CSS）

时钟安全系统用于检测 HSE 时钟是否缺失，如果 HSE 时钟发生缺失，HSE 时钟会被硬件强制关闭。

时钟安全系统检测到 HSE 时钟缺失时，时钟缺失事件作为刹车事件输入 TIM1、TIM15、TIM16 和 TIM17 模块，同时自动产生时钟安全系统中断，该中断会一直执行直到 RCC_CIR 寄存器 CSSF 标志位被清除。该中断是不可屏蔽中断。在时钟安全系统中断服务程序中，通过置位 RCC_CIR 寄存器中 CSSC 位清除 CSSF 标志位。

HSE 直接作为系统时钟或者 HSE 作为 PLL 的时钟源且 PLL 作为系统时钟时 ,使能时钟安全系统功能 ,当 HSE 稳定后，时钟安全系统才开始检测 HSE 时钟，如果 HSE 时钟发生故障，时钟安全系统会将系统时钟直接切换到 HSI 时钟 ,关闭 HSE 时钟 ,如果 PLL 作为系统时钟源 ,则 PLL 时钟被关闭 ,同发生时钟安全系统中断。

7.2.11. RTC 时钟

配置 RCC_BDCR 寄存器中 RTCSEL[1:0]位可以选择 HSE 时钟的 32 分频、LSE 或者 LSI 时钟作为 RTC 时钟。选择 RTC 时钟后，不能重新配置 RTC 时钟，除非重新复位 RTC 域。系统配置必须保证 PCLK 频率大于或者等于 RTC 时钟频率。

7.2.12. 独立看门狗时钟

独立看门狗模块时钟源为 LSI 时钟，当独立看门狗被硬件或者软件使能后，LSI 时钟会自动打开，并且不能被关闭。

7.2.13. 时钟输出（MCO）

芯片可以将内部时钟输出到 MCO 管脚上。下面时钟可以作为 MCO 时钟源：

1. HSI

2. HSI14
3. 系统时钟
4. HSE
5. PLL 时钟的 2 分频或者不分频
6. LSE
7. LSI
8. HSI48

通过配置 RCC_CFGR 寄存器中 MCO[3:0]可以选择不同时钟源 ,配置 MCOPRE[2:0]可以改变输出时钟频率。

7.2.14. 内部和外部时钟测量

利用 TIM14 可以对芯片内部和外部时钟频率进行测量。配置 TIM14_OR 寄存器中 TI1_RMP[1:0]位 ,选择 MCO 输出作为 TIM14 的输入捕获通道的输入。利用时钟频率测量功能，可以清楚的确定芯片内部时钟源的频率，进而可以对时钟的频率进行校准。

7.2.15. 低功耗模式

Sleep 模式下，内核时钟内强制关闭，SRAM 时钟可以通过软件关闭，当所有连接到 APB 的外设时钟被关闭时，AHB 到 APB 桥时钟也被硬件强制关闭。

Stop 模式下 ,HSI、HSE、HSI48 和 PLL 时钟都会被硬件关闭 ,内核时钟也会被关闭。Standby 模式下 ,HSI、HSE、HSI14、HSI48、PLL 以及内核时钟都会被硬件关闭。在 debug 模式下，配置 DBGMCU_CR 寄存器的 DBG_STOP 和 DBG_STANDBY 位可以使系统不进入 Stop 和 Standby 模式。

Stop 和 Standby 模式唤醒后，HSI 时钟作为系统时钟使用。
如果 Flash 正在编程操作，那么此时不会立刻进入 Stop 模式，硬件会在 Flash 编程完成之后，再进行 Stop 模式。如果 APB 总线正在操作 ,此时不会立刻进入 Stop 模式 ,硬件会等待 APB 总线操作完成后 ,再进入 Stop 模式。

7.3.复位和时钟模块寄存器映射

offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RCC_CR	-	-	-	-	-	-	PLLRDY	PLLON	-	-	-	-	CSSON	HSEBYP	HSERDY	HSEON	HSICAL[7:0]								HSITRIM[4:0]					-	HSIRDY	HSION
	Reset	x	x	x	x	x	x	0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	x	1	1
0x04	RCC_CFGR	PLL NODIV	MCOPRE[2:0]			MCO[3:0]				-	-	PLLMUL[3:0]				PLLXTPRE	PLLSRC[1]	PLLSRC[0]	ADC PRE	-	-	-	PPRE[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
	Reset	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0
0x08	RCC_CIR	-	-	-	-	-	-	-	-	CSSC	HSI48RDYC	HSI4RDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC	-	HSI48RDYIE	HSI4RDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	HSI48RDYF	HSI14RDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	RCC_APB2RSTR	-	-	-	-	-	-	-	-	-	DBGMCURST	-	-	-	TIM17RST	TIM16RST	TIM15RST	-	USART1RST	-	SPI1RST	TIM1RST	-	ADCRST	-	-	-	-	-	-	-	-	SYSCFGRST
	Reset	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	x	0	x	0	0	x	0	x	x	x	x	x	x	x	x	0
0x10	RCC_APB1RSTR	-	-	-	PWRRST	CRSRST	-	-	-	USBRST	I2C2RST	I2C1RST	-	-	-	USART2RST	-	-	SPI2RST	-	-	WWDGRST	-	-	TIM14RST	-	-	-	TIM6RST	-	-	TIM3RST	-
	Reset	x	x	x	0	0	x	x	x	0	0	0	x	x	x	0	x	x	0	x	x	0	x	x	0	x	x	x	0	x	x	0	0
0x14	RCC_AHBENR	-	-	-	-	-	-	-	TSCEN	-	IOPFEN	-	IOPDEN	IOPCEN	IOPBEN	IOPAEN	-	-	-	-	-	-	-	-	-	-	CRCEN	-	-	-	SRAMEN	-	DMAEN
	Reset	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	x	x	x	x	x	x	x	x	x	x	0	x	x	x	1	x	0
0x18	RCC_APB2ENR	-	-	-	-	-	-	-	-	-	DBGMCUEN	-	-	-	TIM17EN	TIM16EN	TIM15EN	-	USART1EN	-	SPIEN	TIM1EN	-	ADCNE	-	-	-	-	-	-	-	-	SYSCFGCOMEN
	Reset	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	x	0	x	0	0	x	0	x	x	x	x	x	x	x	x	0
0x1C	RCC_APB1ENR	-	-	-	PWREN	CRSEN	-	-	-	USBEN	I2C2EN	I2C1EN	-	-	-	USART2E	-	-	SPI2EN	-	-	WWDGE	-	-	TIM14EN	-	-	-	TIM6EN	-	-	TIM3EN	-
	Reset	x	x	x	0	0	x	x	x	0	0	0	x	x	x	0	x	x	0	x	x	0	x	x	0	x	x	x	0	x	x	0	0
0x20	RCC_BDCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BDRST	RTCEN	RTCPD	RTCISO	-	-	-	RTCSEL [1:0]		-	-	LSEDRV [2:0]			LSEBYP	LSERDY	LSEON
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	x	x	0	0	x	x	0	1	0	0	0	0
0x24	RCC_CSR	LPWRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLRSTF	RMVF	V18PWRRSTF	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LSIRDY	LSION
	Reset	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0
0x28	RCC_AHBRSTR	-	-	-	-	-	-	-	TSCRST	-	IOPFRST	-	IOPDRST	IOPCREF	IOPBREF	IOPAREF	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Reset	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x2C	RCC_CFGR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PREDIV[3:0]			
	Reset																													0	0	0	0
0x30	RCC_CFGR3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ADCSW	USBSW	-	-	I2C1SW	-	-	USART1 SW[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	0	x	x	0	0
0x34	RCC_CR2	HSI48CAL[8:0]									-	-	-	-	-	HSI48RDY	HSI48ON	HSI14CAL[7:0]								HSI14TRIM[4:0]					HSI14DIS	HSI14RDY	HSI14ON
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0

0x38	RCC_HSECFG	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	HSEADD	HSEDRV[2:0]			HSEDRVEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0
0x3C	RCC_CFGR4	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	SWS[2]	SW[2]
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0
0x40	RCC_TRIM	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	'	HSICAL[8]
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

7.3.1. RCC_CR

偏移地址：0x00

复位值：0x0000 XX83

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—						PLLRDY	PLLON
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO	RW
23:16	—				CSSON	HSEBYP	HSERDY	HSEON
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RO	RW
15:8	HSICAL[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	HSITRIM[4:0]					—	HSIRDY	HSION
类型	RW	RW	RW	RW	RW	RO-0	RO	RW

Bit	Name	Function
31:26	NA	保留位，未定义
25	PLLRDY	PLL 时钟稳定标志位 1：PLL 时钟已稳定 0：PLL 时钟未稳定
24	PLLON	PLL 时钟使能位 软件通过置位和清零使能 PLL 当进入 Stop 和 Standby 模式时，该位会被硬件清零。当 PLL 时钟时钟作为系统时钟或者马上变为系统时钟时，该位不能被清零。 1：PLL 时钟打开 0：PLL 时钟关闭
23:20	NA	保留位，未定义
19	CSSON	时钟安全系统使能位，使能该功能后，当 HSE 时钟稳定，时钟检测功能才能工作 软件通过置位和清零控制时钟安全系统。当 CSSON 被置 1，HSE 时钟稳定后时钟检测功能打开，当检测到 HSE 时钟缺失后，时钟检测功能被禁止。 1：使能时钟安全系统功能 0：禁止时钟安全系统功能

18	HSEBYP	HSE 时钟的外部时钟输入模式选择位 通过软件置位或者清零该位选择外部时钟。使用外部时钟时 HSEON 位必须置 1。 1：使能 HSE 时钟的外部时钟输入模式 0：禁止 HSE 时钟的外部时钟输入模式 注：只能在 HSEON 位为零时写该位
17	HSERDY	HSE 时钟稳定标志位 1：HSE 时钟已稳定 0：HSE 时钟未稳定 注：HSEON 清零后，HSERDY 位在 5 个 HSE 时钟周期后清零
16	HSEON	HSE 时钟使能位 软件置位或者清零 在进入 Stop 或者 Standby 模式时该位被硬件清零。当 HSE 时钟直接或者间接作为系统时钟使用时，该位不能被清零。 1：HSE 时钟使能 0：HSE 时钟关闭
15:8	HSICAL[7:0]	HSI 时钟校准值，这些值在芯片读配置信息时被初始化，与 RCC_TRIM 中 HSICAL[8]一起使用
7:3	HSITRIM[4:0]	HSI 时钟修调位 这些位在 HSICAL[8:0]的基础上，让用户可以输入一个调整值，根据电压和温度的变化调整内部 HSI 振荡器的频率
2	NA	保留位，未定义
1	HSIRDY	HSI 时钟稳定标志位 1：HSI 时钟已稳定 0：HSI 时钟未稳定 注：HSION 清零后，HSIRDY 位在 3 个 HCLK 时钟周期后清零
0	HSION	HSI 时钟使能位 该位在退出 Stop 和 Standby 模式时或者 HSE 直接或者间接作为系统时钟时发生时钟缺失时 ,硬件会置 1。当 HSI 时钟直接或者间接作为系统时钟使用时，该位不能被清零。 1：HSI 时钟打开 0：HSI 时钟关闭

7.3.2. RCC_CFGR

偏移地址：0x04

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	PLLNODIV	MCOPRE[2:0]			MCO[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	—		PLLMUL[3:0]				PLLXTPRE	PLLSRC[1]

类型	RO-0	RO-0	RW	RW	RW	RW	RW	RW
15:8	PLLSRC[0]	ADCPRE	—			PPRE[2:0]		
类型	RW	RW	RO-0	RO-0	RO-0	RW	RW	RW
7:0	HPRE[3:0]				SWS[1:0]		SW[1:0]	
类型	RW	RW	RW	RW	RO	RO	RW	RW

Bit	Name	Function
31	PLLNODIV	PLL 时钟输出到 MCO 分频控制位 1： PLL 时钟不分频作为 MCO 时钟源 0： PLL 时钟的 2 分频作为 MCO 时钟源
30:28	MCOPRE[2:0]	MCO 分频控制位 这些位通过软件置位和清零。为了避免毛刺，建议在 MCO 输出关闭之后再改变 MCO 分频控制位。 111：MCO 输出 128 分频 110：MCO 输出 64 分频 101：MCO 输出 32 分频 100：MCO 输出 16 分频 011：MCO 输出 8 分频 010：MCO 输出 4 分频 001：MCO 输出 2 分频 000：MCO 输出不分频
27:24	MCO[3:0]	MCO 输出时钟选择位 1000：选择内部高速 48MHz 时钟（HSI48） 0111：选择 PLL 时钟 0110：选择外部高速时钟（HSE） 0101：选择内部 8MHz 时钟（HSI） 0100：选择系统时钟 0011：选择外部慢时钟（LSE） 0010：选择内部低速 40kHz 时钟（LSI） 0001：选择内部 14MHz 时钟（HSI14） 0000：时钟输出功能被关闭 注意：这些输出的时钟在 MCO 输出使能启动阶段或者 MCO 时钟源切换时可能存在时钟周期不完整
23:22	NA	保留位，未定义
21:18	PLLMUL[3:0]	PLL 输出时钟倍频系数控制位 这些位只有在 PLL 关闭后才能被修改。 1111：PLL 输入时钟×16 1110：PLL 输入时钟×16 1101：PLL 输入时钟×15 1100：PLL 输入时钟×14 1011：PLL 输入时钟×13 1010：PLL 输入时钟×12 1001：PLL 输入时钟×11

		1000：PLL 输入时钟×10 0111：PLL 输入时钟×9 0110：PLL 输入时钟×8 0101：PLL 输入时钟×7 0100：PLL 输入时钟×6 0011：PLL 输入时钟×5 0010：PLL 输入时钟×4 0001：PLL 输入时钟×3 0000：PLL 输入时钟×2
17	PLLXTPRE	PLL 输入时钟分频控制位 该位与 RCC_CFGR2 寄存器的 PREDIV[0]功能一样 读写该位等于读写 RCC_CFGR2 寄存器的 PREDIV[0]位
16:15	PLLSRC[1:0]	PLL 输入时钟源选择位，这些位只能在 PLL 关闭后设置 00：HSI 的 2 分频 01：HSI/PREDIV 10：HSE/PREDIV 11：HSI48/PREDIV
14	ADCPRE	保留位，可读写
13:11	NA	保留位
10:8	PPRE[2:0]	PCLK 预分频控制位 0xx：HCLK 不分频 100：HCLK 的 2 分频 101：HCLK 的 4 分频 110：HCLK 的 8 分频 111：HCLK 的 16 分频
7:4	HPRE[3:0]	HCLK 预分频控制位 0xxx：系统时钟不分频 1000：系统时钟的 2 分频 1001：系统时钟的 4 分频 1010：系统时钟的 8 分频 1011：系统时钟的 16 分频 1100：系统时钟的 64 分频 1101：系统时钟的 128 分频 1110：系统时钟的 256 分频 1111：系统时钟的 512 分频
3:2	SWS[1:0]	与 RCC_CFGR4 寄存器的 SWS[2]一起使用 ,组成 SWS[2:0],指示当前系统时钟的时钟源 1xx：HSI14 作为系统时钟 000：HSI 作为系统时钟 001：HSE 作为系统时钟 010：PLL 作为系统时钟 011：HSI48 作为系统时钟
1:0	SW[1:0]	与 RCC_CFGR4 寄存器的 SW[2]一起使用，组成 SW[2:0]，选择系统时钟源

		当退出 Stop 和 Standby 模式或者 HSE 直接或者间接作为系统时钟发生缺失时，SW[2:0]会被硬件自动清零 000：HSI 时钟作为系统时钟 001：HSE 时钟作为系统时钟 010：PLL 时钟作为系统时钟 011：HSI48 时钟作为系统时钟 1xx：HSI14 时钟作为系统时钟

7.3.3. RCC_CIR

偏移地址：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	CSSC	HSI48RDYC	HSI14RDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
类型	WO	RO-0	WO	WO	WO	WO	WO	WO
15:8	—	HSI48RDYIE	HSI4RDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
7:0	CSSF	—	HSI14RDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
类型	RO	RO-0	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:24	NA	保留位，未定义
23	CSSC	清除时钟安全系统中断标志位，软件置 1 清除安全系统中断标志位 CSSF 1：清除 CSSF 标志位 0：无影响
22	NA	保留位，未定义
21	HSI4RDYC	清除 HSI14 时钟稳定中断标志位，由软件置 1 清除 HSI14RDYF 标志 1：清除 HSI14RDYF 标志位 0：无影响
20	PLLRDYC	清除 PLL 时钟稳定中断标志位，由软件置 1 清除 PLLRDYF 标志 1：清除 PLLRDYF 标志位 0：无影响
19	HSERDYC	清除 HSE 时钟稳定中断标志位，由软件置 1 清除 HSERDYF 标志 1：清除 HSERDYF 标志位 0：无影响
18	HSIRDYC	清除 HSI 时钟稳定中断标志位，由软件置 1 清除 HSIRDYF 标志 1：清除 HSIRDYF 标志位

		0：无影响
17	LSERDYC	清除 LSE 时钟稳定中断标志位，由软件置 1 清除 LSERDYF 标志 1：清除 LSERDYF 标志位 0：无影响
16	LSIRDYC	清除 LSI 时钟稳定中断标志位，由软件置 1 清除 LSIRDYF 标志 1：清除 LSIRDYF 标志位 0：无影响
15:14	NA	保留位，未定义
13	HSI14RDYIE	HSI14 时钟稳定中断使能位 1：使能 HSI14 中断 0：禁止 HSI14 中断
12	PLLRDYIE	PLL 时钟稳定中断使能位 1：使能 PLL 中断 0：禁止 PLL 中断
11	HSERDYIE	HSE 时钟稳定中断使能位 1：使能 HSE 中断 0：禁止 HSE 中断
10	HSIRDYIE	HSI 时钟稳定中断使能位 1：使能 HSI 中断 0：禁止 HSI 中断
9	LSERDYIE	LSE 时钟稳定中断使能位 1：使能 LSE 中断 0：禁止 LSE 中断
8	LSIRDYIE	LSI 时钟稳定中断使能位 1：使能 LSI 中断 0：禁止 LSI 中断
7	CSSF	时钟安全系统中断标志位 当 jiance 到 HSE 时钟缺失时，该位由硬件置 1。写 CSSC 位清零该位。 1：有由 HSE 错误引起的时钟安全系统中断 0：无由 HSE 错误引起的时钟安全系统中断
6	NA	保留位，未定义
5	HSI14RDYF	HSI14 时钟稳定中断标志位 当 HSI14 时钟稳定、HSI14RDYIE=1 且 HSI14ON=1 时由硬件对该位置 1。 当 HSI14ON=0 时，无论 HSI14 是否稳定该位都一直为 0. 软件写 HSI14RDYC 位清零该位 1：有由 HSI14 时钟稳定引发的中断 0：无由 HSI14 时钟稳定引发的中断
4	PLLRDYF	PLL 时钟稳定中断标志位 当 PLL 时钟稳定且 PLLRDYIE=1 时由硬件对该位置 1。 软件写 PLLRDYC 位清零该位 1：有由 PLL 时钟稳定引发的中断 0：无由 PLL 时钟稳定引发的中断
3	HSERDYF	HSE 时钟稳定中断标志位

		当 HSE 时钟稳定且 HSERDYIE=1 时由硬件对该位置 1。 软件写 HSERDYC 位清零该位。 1：有由 HSE 时钟稳定引发的中断 0：无由 HSE 时钟稳定引发的中断
2	HSIRDYF	HSI 时钟稳定中断标志位 当 HSI 时钟稳定、HSIRDYIE=1 且 HSION=1 时由硬件对该位置 1。当 HSION=0 时，无论 HSI 是否稳定该位都一直为 0. 软件写 HSIRDYC 位清零该位。 1：有由 HSI 时钟稳定引发的中断 0：无由 HSI 时钟稳定引发的中断
1	LSERDYF	LSE 时钟稳定中断标志位 当 LSE 时钟稳定且 LSERDYIE=1 时由硬件对该位置 1。 软件写 LSERDYC 位清零该位。 1：有由 LSE 时钟稳定引发的中断 0：无由 LSE 时钟稳定引发的中断
0	LSIRDYF	LSI 时钟稳定中断标志位 当 LSI 时钟稳定且 LSIRDYIE=1 时由硬件对该位置 1。 软件写 LSIRDYC 位清零该位。 1：有由 LSI 时钟稳定引发的中断 0：无由 LSI 时钟稳定引发的中断

7.3.4. RCC_APB2RSTR

偏移地址：0x0C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—	DBGMCU RST	–			TIM17RST	TIM16RST	TIM15RST
类型	RO-0	RW	RO-0	RO-0	RO-0	RW	RW	RW
15:8	—	USART1RST	—	SPI1RST	TIM1RST	—	ADCRST	—
类型	RO-0	RW	RO-0	RW	RW	RO-0	RW	RO-0
7:0	—							SYSCFGRST
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
31:23	NA	保留位，未定义
22	DBGMCURST	DEBUG 模块复位，由软件置 1 或者清 0 1：复位 DEBUG 模块

		0：无作用
21:19	NA	保留位，未定义
18	TIM17RST	TIM17 模块复位，由软件置 1 或者清 0 1：复位 TIM17 模块 0：无作用
17	TIM16RST	TIM16 模块复位，由软件置 1 或者清 0 1：复位 TIM16 模块 0：无作用
16	TIM15RST	TIM15 模块复位，由软件置 1 或者清 0 1：复位 TIM15 模块 0：无作用
15	NA	保留位，未定义
14	USART1RST	USART1 模块复位，由软件置 1 或者清 0 1：复位 USART1 模块 0：无作用
13	NA	保留位，未定义
12	SPI1RST	SPI1 模块复位，由软件置 1 或者清 0 1：复位 SPI1 模块 0：无作用
11	TIM1RST	TIM1 模块复位，由软件置 1 或者清 0 1：复位 TIM1 模块 0：无作用
10	NA	保留位，未定义
9	ADCRST	ADC 模块复位，由软件置 1 或者清 0 1：复位 ADC 模块 0：无作用
8:1	NA	保留位，未定义
0	SYSCFGRST	系统配置模块复位，由软件置 1 或者清 0 1：复位系统配置模块 0：无作用

7.3.5. RCC_APB1RSTR

偏移地址：0x10

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—			PWRRST	CRSRST	—		
类型	RO-0	RO-0	RO-0	RW	RW	RO-0	RO-0	RO-0
23:16	USBRST	I2C2RST	I2C1RST	—			USART2RST	—
类型	RW	RW	RW	RO-0	RO-0	RO-0	RW	RO-0
15:8	—	SPI2RST	—		WWDGRST	—		TIM14RST

类型	RO-0	RW	RO-0	RO-0	RW	RO-0	RO-0	RW
7:0	—			TIM6RST	—		TIM3RST	—
类型	RO-0	RO-0	RO-0	RW	RO-0	RO-0	RW	RO-0

Bit	Name	Function
31:29	NA	保留位，未定义
28	PWRRST	电源接口模块复位，由软件置 1 或者清 0 1：复位电源接口模块 0：无作用
27	CRSRST	CRS 模块复位，由软件置 1 或者清 0 1：复位 CRS 模块 0：无作用
26:24	NA	保留位，未定义
23	USBRST	USB 模块复位，由软件置 1 或者清 0 1：复位 USB 模块 0：无作用
22	I2C2RST	I2C2 模块复位，由软件置 1 或者清 0 1：复位 I2C2 模块 0：无作用
21	I2C1RST	I2C1 模块复位，由软件置 1 或者清 0 1：复位 I2C1 模块 0：无作用
20:18	NA	保留位，未定义
17	USART2RST	USART2 模块复位，由软件置 1 或者清 0 1：复位 USART2 模块 0：无作用
16:15	NA	保留位，未定义
14	SPI2RST	SPI2 模块复位，由软件置 1 或者清 0 1：复位 SPI2 模块 0：无作用
13:12	NA	保留位，未定义
11	WWDGRST	窗口看门狗模块复位，由软件置 1 或者清 0 1：复位窗口看门狗模块 0：无作用
10:9	NA	保留位，未定义
8	TIM14RST	TIM14 模块复位，由软件置 1 或者清 0 1：复位 TIM14 模块 0：无作用
7:5	NA	保留位，未定义
4	TIM6RST	TIM6 模块复位，由软件置 1 或者清 0 1：复位 TIM6 模块 0：无作用

3:2	NA	保留位，未定义
1	TIM3RST	TIM3 模块复位，由软件置 1 或者清 0 1：复位 TIM3 模块 0：无作用
0	NA	保留位，未定义

7.3.6. RCC_AHBENR

偏移地址：0x14

复位值：0x0000 0004

注意：当外设时钟关闭时，外设模块寄存器不能被软件读到，同时读操作返回 0x0

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							TSCEN
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
23:16	—	IOPFEN	—	IOPDEN	IOPCEN	IOPBEN	IOPAEN	—
类型	RO-0	RW	RO-0	RW	RW	RW	RW	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	CRCEN	—			SRAMEN	—	DMAEN
类型	RO-0	RW	RO-0	RO-0	RO-0	RW	RO-0	RW

Bit	Name	Function
31:25	NA	保留位，未定义
24	TSCEN	Touch 模块时钟使能位 1：Touch 模块时钟打开 0：Touch 模块时钟关闭
23	NA	保留位，未定义
22	IOPFEN	GPIOF 时钟使能位 1：GPIOF 时钟打开 0：GPIOF 时钟关闭
21	NA	保留位，未定义
20	IOPDEN	GPIOD 时钟使能位 1：GPIOD 时钟打开 0：GPIOD 时钟关闭
19	IOPCEN	GPIOC 时钟使能位 1：GPIOC 时钟打开 0：GPIOC 时钟关闭
18	IOPBEN	GPIOB 时钟使能位 1：GPIOB 时钟打开 0：GPIOB 时钟关闭
17	IOPAEN	GPIOA 时钟使能位 1：GPIOA 时钟打开

		0：GPIOA 时钟关闭
16:7	NA	保留位，未定义
6	CRCEN	CRC 校验时钟使能 1：CRC 校验时钟打开 0：CRC 校验时钟关闭
5:3	NA	保留位，未定义
2	SRAMEN	SRAM 时钟使能位 软件可以打开或者关闭 SRAM 时钟 1：SRAM 时钟打开 0：SRAM 时钟关闭
1	NA	保留位，未定义
0	DMAEN	DMA 模块时钟使能位 1：DMA 模块时钟打开 0：DMA 模块时钟关闭

7.3.7. RCC_APB2ENR

偏移地址：0x18

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—	DBGMCUEN	—			TIM17EN	TIM16EN	TIM15EN
类型	RO-0	RW	RO-0	RO-0	RO-0	RW	RW	RW
15:8	—	USART1EN	—	SPI1EN	TIM1EN	—	ADCEN	—
类型	RO-0	RW	RO-0	RW	RW	RO-0	RW	RO-0
7:0	—							SYSCFGEN
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
31:23	NA	保留位，未定义
22	DBGMCUEN	MCU 调试模块时钟使能位 1：MCU 调试模块时钟打开 0：MCU 调试模块时钟关闭
21:19	NA	保留位，未定义
18	TIM17EN	TIM17 时钟使能位 1：TIM17 时钟打开 0：TIM17 时钟关闭
17	TIM16EN	TIM16 时钟使能位

		1：TIM16 时钟打开 0：TIM16 时钟关闭
16	TIM15EN	TIM15 时钟使能位 1：TIM15 时钟打开 0：TIM15 时钟关闭
15	NA	保留位，未定义
14	USART1EN	USART1 模块时钟使能位 1：USART1 模块时钟打开 0：USART1 模块时钟关闭
13	NA	保留位，未定义
12	SPI1EN	SPI1 模块时钟使能位 1：SPI1 模块时钟打开 0：SPI1 模块时钟关闭
11	TIM1EN	TIM1 时钟使能位 1：TIM1 时钟打开 0：TIM1 时钟关闭
10	NA	保留位，未定义
9	ADCEN	ADC 时钟使能位 1：ADC 时钟打开 0：ADC 时钟关闭
8:1	NA	保留位，未定义
0	SYSCFGEN	系统配置模块时钟使能位 1：系统配置模块时钟打开 0：系统配置模块时钟关闭

7.3.8. RCC_APB1ENR

偏移地址：0x1C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—			PWREN	CRSEN	—	—	—
类型	RO-0	RO-0	RO-0	RW	RW	RO-0	RO-0	RO-0
23:16	USBEN	I2C2EN	I2C1EN	—			USART2EN	—
类型	RW	RW	RW	RO-0	RO-0	RO-0	RW	RO-0
15:8	—	SPI2EN	—		WWDGEN	—		TIM14EN
类型	RO-0	RW	RO-0	RO-0	RW	RO-0	RO-0	RW
7:0	—			TIM6EN	—		TIM3EN	—
类型	RO-0	RO-0	RO-0	RW	RO-0	RO-0	RW	RO-0

Bit	Name	Function
31:29	NA	保留位，未定义

28	PWREN	电源接口时钟使能位 1：电源接口时钟打开 0：电源接口时钟关闭
27	CRSEN	CRS 模块时钟使能位 1：CRS 模块时钟打开 0：CRS 模块时钟关闭
26:24	NA	保留位，未定义
23	USBEN	USB 模块时钟使能位 1：USB 模块时钟打开 0：USB 模块时钟关闭
22	I2C2EN	I2C2 模块时钟使能位 1：I2C2 模块时钟打开 0：I2C2 模块时钟关闭
21	I2C1EN	I2C1 模块时钟使能位 1：I2C1 模块时钟打开 0：I2C1 模块时钟关闭
20:18	NA	保留位，未定义
17	USART2EN	USART2 模块时钟使能位 1：USART2 模块时钟打开 0：USART2 模块时钟关闭
16:15	NA	保留位，未定义
14	SPI2EN	SPI2 模块时钟使能位 1：SPI2 模块时钟打开 0：SPI2 模块时钟关闭
13:12	NA	保留位，未定义
11	WWDGEN	窗口看门狗模块时钟使能位 1：窗口看门狗模块时钟打开 0：窗口看门狗模块时钟关闭
10:9	NA	保留位，未定义
8	TIM14EN	TIM14 时钟使能位 1：TIM14 时钟打开 0：TIM14 时钟关闭
7:5	NA	保留位，未定义
4	TIM6EN	TIM6 时钟使能位 1：TIM6 时钟打开 0：TIM6 时钟关闭
3:2	NA	保留位，未定义
1	TIM3EN	TIM3 时钟使能位 1：TIM3 时钟打开 0：TIM3 时钟关闭
0	NA	保留位，未定义

7.3.9. RCC_BDCR

偏移地址：0x20

复位值：0x0000 0010

注意：RCC_BDCR 寄存器中 LSEON、LSEBYP、RTCSEL 和 RTCEN 在复位后处于写保护状态。只有在电源控制寄存器(PWR_CR)中 DBP 位置 1 后才能对这些位进行改动 ,任何内部或者外部复位不会影响这些位。

RCC_BDCR 寄存器的写操作最高频率为 48MHz，超过该频率将不保证写操作的正确性。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							BDRST
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
15:8	RTCEN	RTCPD	RTCISO	—			RTCSEL[1:0]	
类型	RW	RW	RO	RO-0	RO-0	RO-0	RW	RW
7:0	—		LSEDRV[2:0]			LSEBYP	LSERDY	LSEON
类型	RO-0	RO-0	RW	RW	RW	RW	RO	RW

Bit	Name	Function
31:17	NA	保留位，未定义
16	BDRST	RTC 域软件复位，该位由软件置 1 或清 0 1：RTC 域复位(包括 RCC_BDCR 寄存器) 0：RTC 域不复位
15	RTCEN	RTC 时钟使能位 1：RTC 时钟打开 0：RTC 时钟关闭
14	RTCPD	RTC 域电源控制位 1：RTC 域电源关闭 0：RTC 域电源打开
13	RTCISO	RTC 域隔离状态标志位 1：RTC 域处于隔离状态，不能对 RTC 域（1.5V 域）寄存器进行操作 0：RTC 域未处于隔离状态，可以对 RTC 域（1.5V 域）寄存器进行操作
12:10	NA	保留位，未定义
9:8	RTCSEL[1:0]	RTC 时钟源选择位，该位只能在 RTCSEL[1:0]=2'b00 时，进行写操作或者在利用 BDRST 复位之后可进行写操作 00：无时钟 01：LSE 作为 RTC 时钟 10：LSI 作为 RTC 时钟 11：HSE/32 作为 RTC 时钟
7:6	NA	保留位，未定义
5:3	LSEDRV[2:0]	LSE 振荡器驱动能力选择位 000：晶体模式弱驱动能力 001：晶体模式中低驱动能力

		010：晶体模式中较高驱动能力 011：晶体模式高驱动能力 1xx：晶体模式最高驱动能力
2	LSEBYP	LSE 外部时钟灌入模式选择 1：直接灌入时钟模式，无需晶振 0：未选择灌入时钟模式
1	LSERDY	LSE 时钟稳定标志位 1：LSE 时钟已稳定 0：LSE 时钟未稳定 注：LSEON 清零后，HSIRDY 位在 4 个 LSE 时钟周期后清零
0	LSEON	LSE 时钟使能位 1：LSE 时钟打开 0：LSE 时钟关闭

7.3.10. RCC_CSR

偏移地址：0x24

复位值：0x0880 0000

注意：RCC_CSR 寄存器的写操作最高频率为 48MHz，超过该频率将不保证写操作的正确性。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	LPWRRSTF	WWDGRSTF	IWWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLRSTF	RMVF
类型	RO	RO	RO	RO	RO	RO	RO	WO
23:16	VDDLRSTF	—						
类型	RO	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—						LSIRDY	LSION
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO	RW

注：系统复位不会清除复位标志位，上电复位或者掉电复位则会清除复位标志位。

Bit	Name	Function
31	LPWRRSTF	低功耗复位标志位，低功耗复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 1：发生了低功耗复位 0：未发生低功耗复位
30	WWDGRSTF	窗口看门狗复位标志位，窗口看门狗复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 1：发生了窗口看门狗复位 0：未发生窗口看门狗低功耗复位
29	IWWDGRSTF	独立看门狗复位标志位，独立看门狗复位发生时，由硬件置 1，由软件通过写

		RMVF 位清除该位。 1：发生了独立看门狗复位 0：未发生独立看门狗低功耗复位
28	SFTRSTF	软件复位标志位，软件复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 1：发生了软件复位 0：未发生软件复位
27	PORRSTF	芯片掉电或者上电复位标志位，芯片掉电或者上电复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 VDDA 发生欠压复位时，该位也会置 1 1：发生了芯片掉电或者上电复位 0：未发生芯片掉电或者上电复位
26	PINRSTF	NRST 管脚复位标志位 ,NRST 复位发生时 ,由硬件置 1 ,由软件通过写 RMVF 位清除该位。 1：发生了 NRST 管脚复位 0：未发生 NRST 管脚复位
25	OBLRSTF	配置字加载复位标志位，配置字加载复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 1：发生了配置字加载复位 0：未发生配置字加载复位
24	RMVF	清除复位标志 0：无作用 1：清除所有复位标志位
23	VDDLRSTF	VDDL 域 (1.6V 域) 上电或者掉电复位标志位，VDDL 域上电或者掉电复位发生时，由硬件置 1，由软件通过写 RMVF 位清除该位。 VDDA 发生欠压复位时，该位也会置 1 1：发生了 VDDL 与上电或者掉电复位 0：未发生 VDDL 与上电或者掉电复位
22:2	NA	保留位，未定义
1	LSIRDY	LSI 时钟稳定标志位 1：LSI 时钟已稳定 0：LSI 时钟未稳定 注：LSION 清零后，LSIRDY 位在 3 个 LSI 时钟周期后清零，但有以下例外: 程序开始运行的 4ms 内 LSIRDY 位会一直置高，即使 LSION 位清零。
0	LSION	LSI 时钟使能位 1：LSI 时钟打开 0：LSI 时钟关闭 退出待机模式时，该位被清零，发生系统复位时，该位也会被清零

7.3.11. RCC_AHBRSTR

偏移地址：0x28

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							TSCRST
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
23:16	—	IOPFRST	—	IOPDRST	IOPCRST	IOPBRST	IOPARST	—
类型	RO-0	RW	RO-0	RW	RW	RW	RW	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:25	NA	保留位，未定义
24	TSCRST	Touch 模块复位控制位，由软件置 1 或清 0 1：复位 Touch 模块 0：无作用
23	NA	保留位，未定义
22	IOPFRST	GPIOF 复位控制位，由软件置 1 或清 0 1：复位 GPIOF 0：无作用
21	NA	保留位，未定义
20	IOPDRST	GPIOD 复位控制位，由软件置 1 或清 0 1：复位 GPIOD 0：无作用
19	IOPCRST	GPIOC 复位控制位，由软件置 1 或清 0 1：复位 GPIOC 0：无作用
18	IOPBRST	GPIOB 复位控制位，由软件置 1 或清 0 1：复位 GPIOB 0：无作用
17	IOPARST	GPIOA 复位控制位，由软件置 1 或清 0 1：复位 GPIOA 0：无作用
16:0	NA	保留位，未定义

7.3.12. RCC_CFGR2

偏移地址：0x2C

复位值：0x0000 0000

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

Bit	Name	Function
31:4	NA	保留位，未定义
3:0	PREDIV[3:0]	PLL 输入分频系数，这些位仅能在 PLL 时钟关闭后改写 注:修改 RCC_CFGR 的 PLLXTPRE 位同时改变 PREDIV[0] 0000：PLL 输入不分频 0001：PLL 输入 2 分频 0010：PLL 输入 3 分频 0011：PLL 输入 4 分频 0100：PLL 输入 5 分频 0101：PLL 输入 6 分频 0110：PLL 输入 7 分频 0111：PLL 输入 8 分频 1000：PLL 输入 9 分频 1001：PLL 输入 10 分频 1010：PLL 输入 11 分频 1011：PLL 输入 12 分频 1100：PLL 输入 13 分频 1101：PLL 输入 14 分频 1110：PLL 输入 15 分频 1111：PLL 输入 16 分频

7.3.13. RCC_CFGR3

偏移地址：0x30

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							ADCSW
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
7:0	USBSW	—		I2C1SW	—		USART1SW[1:0]	
类型	RW	RO-0	RO-0	RW	RO-0	RO-0	RW	RW

Bit	Name	Function
31:9	NA	保留位，未定义
8	ADCSW	保留位，未定义
7	USBSW	USB 时钟源选择位 1：PLL 作为 USB 时钟源 0：HSI48 作为 USB 时钟源
6:5	NA	保留位，未定义
4	I2C1SW	I2C1 时钟源选择位 1：系统时钟作为 I2C1 时钟源 0：HSI 作为 I2C1 时钟源
3:2	NA	保留位，未定义
1:0	USART1SW[1:0]	USART1 时钟源选择位 11：HSI 时钟作为 USART1 时钟源 10：LSE 时钟作为 USART1 时钟源 01：系统时钟作为 USART1 时钟源 00：PCLK 作为 USART1 时钟源

7.3.14. RCC_CR2

偏移地址：0x34

复位值：0xXXX00 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	HSI48CAL[8:1]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
23:16	HSI48CAL[0]	—					HSI48RDY	HSI48ON
类型	RO	RO-0	RO-0	RO-0	RO-0	RO-0	RO	RW
15:8	HSI14CAL[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	HSI14TRIM[4:0]					HSI14DIS	HSI14RDY	HSI14ON
类型	RW	RW	RW	RW	RW	RW	RO	RW

Bit	Name	Function
31:23	HSI48CAL[8:0]	HSI48 时钟校准 在启动阶段这些位被自动初始化为出场校准参数
22:18	NA	保留位，未定义
17	HSI48RDY	HSI48 时钟稳定标志位 1：HSI48 时钟已稳定 0：HSI48 时钟未稳定 注：HSI48ON 清零后，HSI48RDY 位在 3 个 HCLK 时钟周期后清零
16	HSI48ON	HSI48 时钟使能位 1：HSI48 时钟打开 0：HSI48 时钟关闭
15:8	HSI14CAL[7:0]	HSI14 时钟校准 在启动阶段这些位被自动初始化为出场校准参数
7:3	HSI14TRIM[4:0]	HSI14 时钟微调位 这些位在 HSI14CAL[7:0]基础上，让用户可以输入一个调整值，根据电压和稳定的变化调整内部 HSI14 的频率。
2	HSI14DIS	ADC 请求打开 HSI14 时钟控制位 1：ADC 接口不能打开 HSI14 时钟 0：ADC 接口可以打开 HSI14 时钟
1	HSI14RDY	HSI14 时钟稳定标志位 1：HSI14 时钟已稳定 0：HSI14 时钟未稳定 注：HSI14ON 清零后，HSI14RDY 位在 3 个 HCLK 时钟周期后清零
0	HSI14ON	HSI14 时钟使能位 1：HSI14 时钟打开 0：HSI14 时钟关闭

7.3.15. RCC_HSECFG

偏移地址：0x38

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—			HSEADD	HSEDRV[2:0]			HSEDRVEN
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:5	NA	保留位，未定义
4	HSEADD	HSE 驱动电流增大控制位 1：HSE 驱动电流增大 0：HSE 驱动电流无变化
3:1	HSEDRV[2:0]	HSE 驱动大小调整位，数值大驱动能力强，数值小驱动能力弱 111：HSE 驱动能力强 …. 000：HSE 驱动能力弱
0	HSEDRVEN	HSE 驱动大小调整控制位 1：软件可以通过修改 HSEDRV[2:0]调整 HSE 驱动能力 0：不能通过修改 HSEDRV[2:0]调整 HSE 驱动能力

7.3.16. RCC_CFGR4

偏移地址：0x3C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—						SWS[2]	SW[2]
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO	RW

Bit	Name	Function
31:2	NA	保留位，未定义
1	SWS[2]	当前系统时钟的时钟源指示位，与 RCC_CFGR 中 SWS[1:0]一起使用，具体参见 RCC_CFGR 相关描述
0	SW[2]	系统时钟的时钟源选择位，与 RCC_CFGR 中 SW[1:0]一起使用，具体功能描述参见 RCC_CFGR 相关描述

7.3.17. RCC_TRIM

偏移地址：0x40

复位值：0x0000 000X

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

Bit	Name	Function
31:1	NA	保留位，未定义
0	HSICAL[8]	HSI 时钟校准位，与 RCC_CR 中 HSICAL[7:0]一起使用，具体参见 RCC_CR 寄存器中相关描述

8. 时钟恢复系统（CRS）

8.1. 介绍

时钟恢复系统（CRS）是一个先进的数字控制器，作用于内部可微调的 RC 振荡器 HSI48。通过与可选同步信号的比较，CRS 为振荡器输出频率计算提供了一个强有力的手段。它能够根据测量到的频率误差值自动调整振荡器微调值，同时保持了手动微调的可能性。

8.2. CRS 主要特性

- 带可编程的预分频和极性的可选同步源：
  - 外部引脚
  - LSE 振荡器输出
  - USB SOF 包接收信号
- 可由软件产生同步脉冲
- 自动振荡器微调功能，无需 CPU 操作
- 手动控制选项，更快的启动收敛
- 带自动误差值捕捉和重载的 16 位频率误差计数器
- 可编程限制值，自动频率误差值计算和状态报告。
- 可屏蔽中断/事件：
  - 预期同步（ESYNC）
  - 同步 OK（SYNCOK）
  - 同步警告（SYNCWARN）
  - 同步或微调错误（ERR）

8.3. CRS 功能描述

8.3.1. CRS 框图

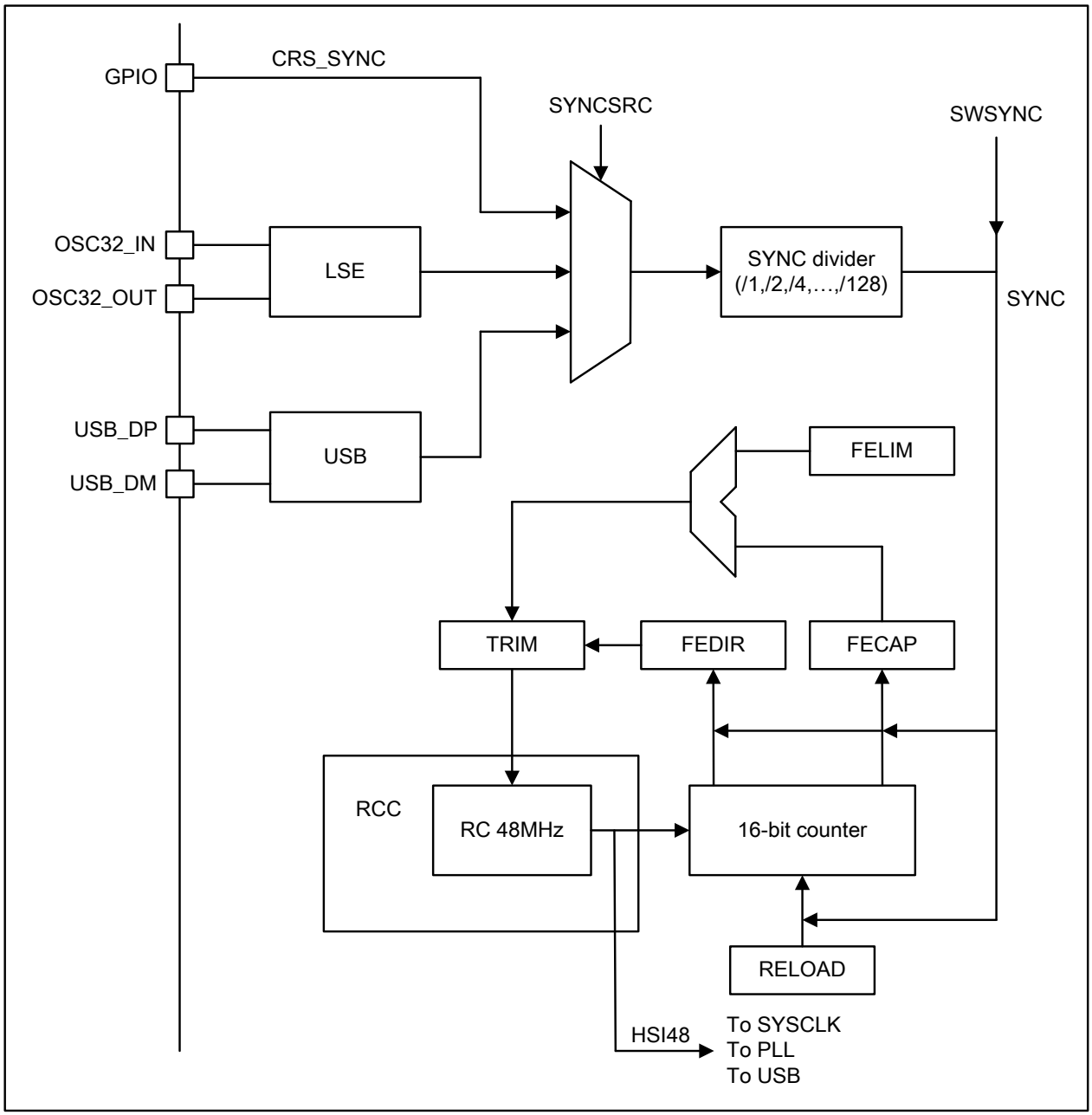


图 8.1. CRS 框图

8.3.2. 同步输入

CRS 同步（SYNC）源可以通过 CRS_CFGR 寄存器选择，可以是来自外部 CRS_SYNC 引脚、LSE 时钟或者 USB SOF 信号。为了更好的 SYNC 输入鲁棒性，一个简单的数字滤波器（由 HSI48 时钟采样，2/3 多数

有效)被实现来滤除任何毛刺。这个源信号也有一个可配置的极性,然后可以通过一个可编程二进制预分频器进行分频，以获得一个在合适频率范围内的同步信号（通常在 1 kHz 左右）。
有关 CRS 同步源配置的更多信息，请参阅第 7.6.2 节：CRS 配置寄存器（CRS_CFGR）。
还可以通过软件置位 CRS_CR 寄存器中的 SWSYNC 位来产生一个同步事件。

8.3.3. 频率误差测量

频率误差计数器是一个 16 位的向下/向上计数器，它在每个 SYNC 事件时重新加载 RELOAD 值。它一开始向下计数，直到零值，此时产生一个 ESYNC(预期同步)事件。然后它开始向上计数到最终停止的 OUTRANGE 限制值（如果没有收到同步时间），并产生一个 SYNCMISS 事件。OUTRANGE 限制值定义为频率误差限制值（CRS_CFGR 寄存器的 FELIM 字段）乘以 128 倍。
当检测到同步事件的时候,频率误差计数器的实际值及其计数方向将存储在 CRS_ISR 寄存器中的 FECAP(频率误差捕捉）字段和 FEDIR（频率误差方向）位。当检测到 SYNC 事件发生在向下计数阶段（在到达零值之前）时，这意味着实际频率低于目标频率（所以 TRIM 值应该递增）。当检测到 SYNC 事件发生在向上计数阶段时，这意味着实际频率高于目标频率（所以 TRIM 值应该递减）。

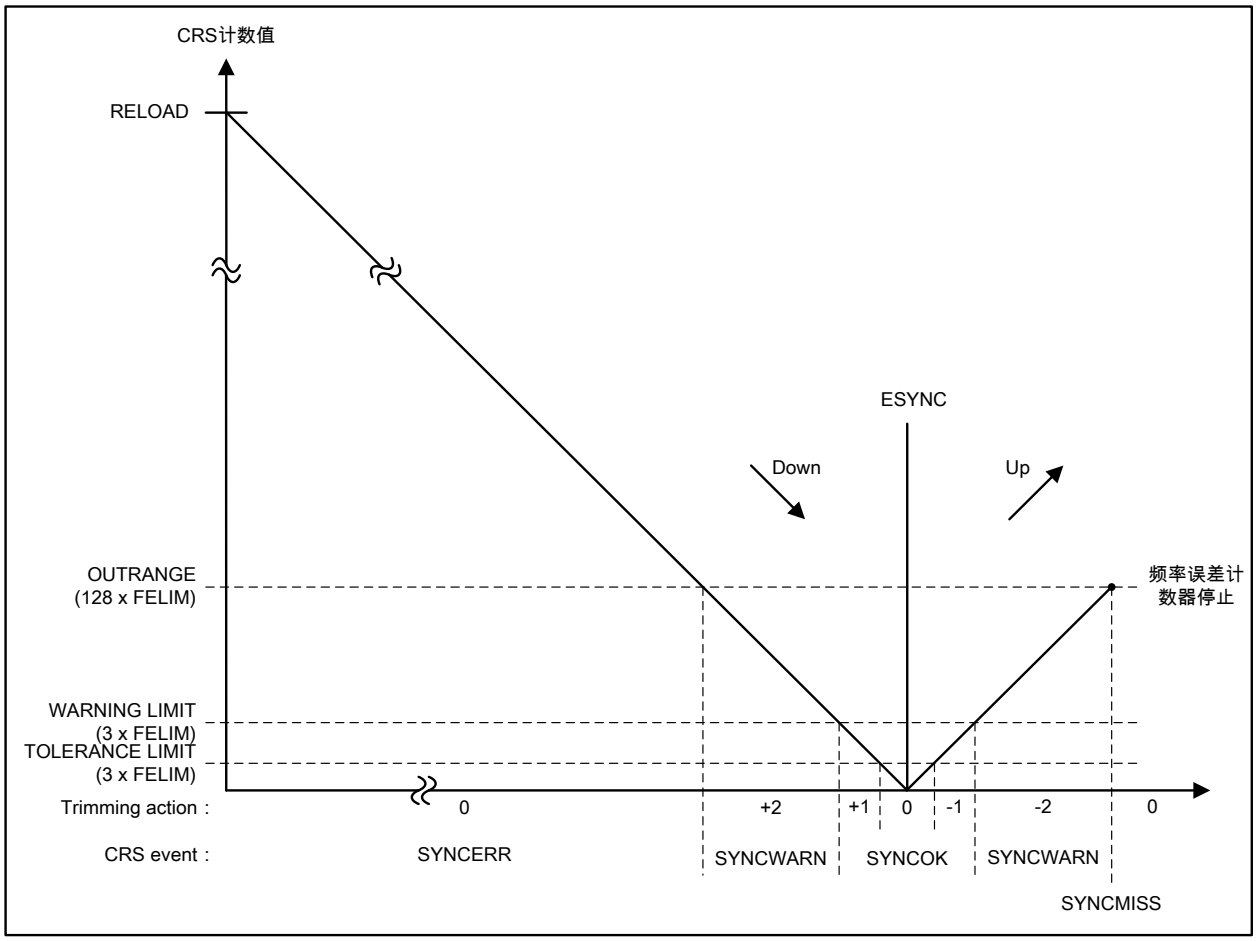


图 8.2.CRS 计数器行为

8.3.4. 频率误差计算和自动调整

测量的频率误差通过比较其值和一组限制值来计算：

- TOLERANCE LIMIT，由 CRS_CFGR 寄存器的 FELIM 字段直接给出
- WARNING LIMIT，定义为 FELIM 值的 3 倍
- OUTRANGE（错误限制值），定义为 FELIM 值的 128 倍

这个比较结果用来产生状态指示，也用于控制自动微调，该自动微调是通过设置 CRS_CR 寄存器的 AUTOTRIMEN 位来使能的：

- 当频率误差低于容许限制值时，意味着在 TRIM 字段里的实际微调值是最佳的，此时不需要再进行微调操作。
  - 指示 SYNCOK 状态
  - 在 AUTOTRIM 模式下的 TRIM 值不再变化
- 当频率误差低于警告限制值但高于或等于容许限制值时，意味着需要做一些微调动作，但一次微调步骤的调整就足以达到最佳的 TRIM 值。
  - 指示 SYNCOK 状态
  - 在 AUTOTRIM 模式下的 TRIM 值会进行一次微调步骤调整
- 当频率误差高于或低于警告限制值但低于错误限制值时，意味着需要做更有力的微调动作，并有可能在下一个周期还无法达到最佳的 TRIM 值。
  - 指示 SYNCWARN 状态
  - 在 AUTOTRIM 模式下的 TRIM 值会经过两次微调步骤调整
- 当频率误差高于或等于错误限制值时，意味着频率已经超出了微调范围。这种现象也可能发生在 SYNC 输入信号不干净，或者某些 SYNC 脉冲丢失的情况下（例如当一个 USB SOF 损坏的时候）。
  - 指示 SYNCERR 或 SYNCMISS 状态
  - 在 AUTOTRIM 模式下的 TRIM 值不再变化

注：如果 TRIM 字段的实际值非常接近它的限制值，自动微调将会迫使它上溢或者下溢，那么 TRIM 值会被设置为刚好的限制值，并指示 TRIMOVF 状态。

在 AUTOTRIM 模式（置位 CRS_CR 寄存器的 AUTOTRIMEN 位），CRS_CR 的 TRIM 字段是通过硬件调整，并且是只读的。

8.3.5. CRS 初始化和配置

RELOAD 值

根据目标频率和同步源分频后的频率之间的比值来选择 RELOAD 值。然后将其减一，以便在零值上达到预期同步。公式如下：

RELOAD = (f_{TARGET} / f_{SYNC}) - 1

RELOAD 字段的复位值对应于 48MHz 的目标频率和 1kHz 的同步信号频率（来自 USB 的 SOF 信号）。

FELIM 值

FELIM 值的选择与 HSI48 振荡器特性及其典型微调步长有密切关联。最佳值对应微调步长的一半，用 HSI48M 振荡器时钟的节拍数表示。可以使用以下公式：

FELIM = (f_{TARGET} / f_{SYNC}) * STEP[%] / 100% / 2

为了获得最好的微调响应，应该将结果四舍五入到最接近的整数值。如果不想在应用中有频繁的微调动作，则

可通过略微增加 FELIM 值来增加微调滞后。

FELIM 字段的复位值对应于(f_{TARGET} / f_{SYNC})=48000 和 0.14%的典型微调步长。

注意：没有硬件保护机制来防止错误配置 RELOAD 和 FELIM 字段，这可能会导致不稳定的微调响应，预期的操作模式需要正确设置 RELOAD 值(根据同步源频率),该值也应该大于 FELIM 值的 128 倍(OUTRANGE 限制值)。

8.4. CRS 低功耗模式

表 8.1. 低功耗模式对 CRS 的影响

模式	描述
Sleep	没影响。 CRS 中断可使得器件退出 Sleep 模式
Stop	CRS 寄存器被冻结。
Standby	CRS 停止工作，直到 Stop 或 Standby 模式退出和 HSI48 振荡器重启。

8.5. CRS 中断

表 8.2. 中断控制位

中断事件	事件标志	使能控制位	清零标志位
预期同步	ESYNCF	ESYNCIE	ESYNCC
同步 OK	SYNCOKF	SYNCOKIE	SYNCOKC
同步警告	SYNCWARNF	SYNCWARNIE	SYNCWARNC
同步或微调错误 (TRIMOVF,SYNCMISS,SYNCERR)	ERRF	ERRIE	ERRC

8.6. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRS_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TRIM[5:0]						SWSYNC	AUTOTRIMEN	CEN	-	ESYNCIE	ERRIE	SYNCWARNIE	SYNCOKIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	x	0	0	0	0
0x04	CRS_CFGR	SYNCPOL	-	SYNCSRC [1:0]		-	SYNC DIV [2:0]			FELIM[7:0]								RELOAD[15:0]															
	Reset	0	x	1	0	x	0	0	0	0	0	1	0	0	0	1	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1
0x08	CRS_ISR	FECAP[15:0]																FEDIR	-	-	-	-	TRIMOVF	SYNCMISS	SYNCERR	-	-	-	-	ESYNCF	ERRF	SYNCWARNF	SYNCOKF
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	0	0	0	x	x	x	x	0	0	0	0
0x0C	CRS_ICR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ESYNCC	ERRC	SYNCWARNC	SYNCOKC
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0

8.6.1. CRS_CR（CRS 控制寄存器）

地址偏移：0x00

复位值：0x0000 2000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		TRIM[5:0]					
类型	RO-0	RO-0	RW	RW	RW	RW	RW	RW
7:0	SWSYNC	AUTOTRIMEN	CEN	—	ESYNCIE	ERRIE	SYNCWARNIE	SYNCOKIE
类型	RT_W	RW	RW	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:14	NA	保留位，未定义
13:8	TRIM[5:0]	HSI48 振荡器平滑微调位 这些位提供一个用户可编程的微调数值给 HSI48 振荡器。它们可以通过编程来适配影响 HSI48 频率的电压和温度变化。 默认值为 32，对应微调间隔的中间。两个连续微调步骤之间的微调步长约为 67 kHz。更高的 TRIM 值对应更高的输出频率。

		当置位了 AUTOTRIMEN 位，该字段由硬件控制，并且是只读的。
7	SWSYNC	产生软件 SYNC 事件 该位由软件置位，用来产生一个软件 SYNC 事件。它会被硬件自动清零。 0：无动作 1：产生一个软件 SYNC 事件
6	AUTOTRIMEN	自动微调使能 该位使能 TRIM 位自动硬件调整功能，该功能根据两个 SYNC 事件之间的测量频率误差值进行调整。如果该位被置位，TRIM 位将是只读的。根据测量频率误差值，硬件会一次调整 TRIM 值一个步长或者两个步长。详见第 7.3.4 节：频率误差计算和自动微调。 0：禁用自动微调，TRIM 位可由用户调整。 1：使能自动微调，TRIM 位只读并由硬件控制。
5	CEN	频率误差计数器使能位 该位使能频率误差计数器的振荡器时钟。 0：禁用频率误差计数器 1：使能频率误差计数器 当该位置位时，CRS_CFGR 寄存器会是写保护的，并且不能被修改。
4	NA	保留位，未定义
3	ESYNCIE	预期同步中断使能位 0：禁用预期同步（ESYNCF）中断 1：使能预期同步（ESYNCF）中断
2	ERRIE	同步或微调错误中断使能位 0：禁用同步或微调错误（ERRF）中断 1：使能同步或微调错误（ERRF）中断
1	SYNCWARNIE	同步警告中断使能位 0：禁用同步警告（SYNCWARNF）中断 1：使能同步警告（SYNCWARNF）中断
0	SYNCOKIE	同步 OK 中断使能位 0：禁用同步 OK（SYNCOKF）中断 1：使能同步 OK（SYNCOKF）中断

8.6.2. CRS_CFGR（CRS 配置寄存器）

地址偏移：0x04

复位值：0x2022 BB7F

只有在禁用频率错误计数器（清零 CRS_CR 的 CEN 位）时才能写入此寄存器。使能计数器时，此寄存器是写保护的。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	SYNCPOL	—	SYNCSRC[1:0]		—	SYNCDIV[2:0]		
类型	RW	RO-0	RW	RW	RO-0	RW	RW	RW
23:16	FELIM[7:0]							

类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	RELOAD[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	RELOAD[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31	SYNCPOL	SYNC 极性选择 该位由软件置位和清零，用来选择 SYNC 信号源的输入极性。 0：SYNC 上升沿有效（默认） 1：SYNC 下降沿有效
30	NA	保留位，未定义
29:28	SYNCSRC[1:0]	SYNC 信号源选择 这些位由软件置位和清零，用来选择 SYNC 信号源。 00：选择 GPIO 作为 SYNC 信号源 01：选择 LSE 作为 SYNC 信号源 10：选择 USB SOF 作为 SYNC 信号源（默认） 11：保留 注：当使用 USB LPM（Link Power Management）和设备处于 Sleep 模式时，主机不会产生周期性的 USB SOF 信号。因此没有 SYNC 信号提供给 CRS 进行校准运行中的 HSI48。为了保证从 Sleep 模式唤醒后所需时钟的精度，应该使用 LSE 或者 GPIO 上的参考时钟作为 SYNC 信号。
27	NA	保留位，未定义
26:24	SYNCDIV[2:0]	SYNC 分频系数 这些位由软件置位和清零，用来控制 SYNC 信号分频系数。 000：不分频（默认） 001：2 分频 010：4 分频 011：8 分频 100：16 分频 101：32 分频 110：64 分频 111：128 分频
23:16	FELIM[7:0]	频率误差限制值 FELIM 包含的数值用于计算锁存到 CRS_ISR 寄存器 FECAP[15:0]位的捕捉频率误差值。更多关于 FECAP 计算的细节，请参阅第 7.3.4 节：频率误差计算和自动微调。
15:0	RELOAD[15:0]	计数器重载值 RELOAD 是在每个 SYNC 事件发生时加载到频率误差计数器里的数值。更多关于计数器行为的信息，请参阅第 7.3.3：频率误差测量。

8.6.3. CRS_ISR（CRS 中断状态寄存器）

地址偏移：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	FECAP[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
23:16	FECAP[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
15:8	FEDIR	—				TRIMOVF	SYNCMISS	SYNCERR
类型	RO	RO-0	RO-0	RO-0	RO-0	RO	RO	RO
7:0	—				ESYNCF	ERRF	SYNCWARNF	SYNCOKF
类型	RO-0	RO-0	RO-0	RO-0	RO	RO	RO	RO

Bit	Name	Function
31:16	FECAP[15:0]	频率误差捕捉值 FECAP 为最后一次 SYNC 事件时锁定的频率误差计数器数值。更多关于 FECAP 使用方法的信息，请参阅第 7.3.4 节：频率误差计算和自动微调。
15	FEDIR	频率误差方向 FEDIR 是频率误差计数器在最后一次 SYNC 事件时锁定的计数方向。它指示实际频率是低于还是高于目标。 0：向上计数方向，实际频率高于目标频率。 1：向下计数方向，实际频率低于目标频率。
14:11	NA	保留位，未定义
10	TRIMOVF	微调上溢或下溢 该标志由硬件在自动微调试图超过或低过 TRIM 值时置位。若置位了 CRS_CR 寄存器的 ERRIE 标志，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 ERRC 位来清零该位。 0：没有微调错误发生 1：发生微调错误
9	SYNCMISS	SYNC 丢失 当频率误差计数器到达 FELM 值的 128 倍，却没有检测到 SYNC 信号时，由硬件置位该标志。这意味着 SYNC 脉冲丢失了或者频率误差太大了（内部频率太高），不能通过调整 TRIM 值补偿回来，应该采取一些其他行动。此时，频率误差计数器停止（等待下一个 SYNC），若置位了 CRS_CR 寄存器的 ERRIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 ERRC 位来清零该位。 0：没有发生 SYNC 丢失错误 1：发生 SYNC 丢失错误

8	SYNCERR	SYNC 错误 当同步脉冲在 ESYNC 事件之前到达，并且测量频率误差大于或等于 FELIM 的 128 倍时，由硬件置位该标志。这意味着频率误差太大（内部频率太低），不能通过调整 TRIM 值补偿回来，应该采用一些其他行动。若置位了 CRS_CR 寄存器的 ERRIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 ERRC 位来清零该位。 0：没有发生 SYNC 错误 1：发生 SYNC 错误
7:4	NA	保留位，未定义
3	ESYNCF	预期同步标志位 当频率错误计数器到达零值时，由硬件置位该标志。若置位了 CRS_CR 寄存器的 ESYNCIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 ESYNCC 位来清零该位。 0：没有发生预期同步事件 1：发生预期同步事件
2	ERRF	错误标志位 若有任何同步或微调错误时，该标志由硬件置位。它是 TRIMOVF、SYNCMISS、SYNCERR 位的逻辑或输出。若置位了 CRS_CR 寄存器的 ERRIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 ERRC 位来清零该位，同时清零 TRIMOVF、SYNCMISS、SYNCERR 位。 0：没有发生同步或微调错误 1：发生同步或微调错误
1	SYNCWARNF	同步警告标志位 当测量频率误差大于或等于 FELIM 的 3 倍，但小于 FELIM 的 128 倍时，由硬件置位该标志。这意味着要补偿频率误差的话，TRIM 值至少要调整两个步长或更多。若置位了 CRS_CR 寄存器的 SYNCWARNIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 SYNCWARNC 位来清零该位。 0：没有发生同步警告 1：发生同步警告
0	SYNCOKF	同步 OK 标志位 当测量频率误差小于 FELIM 的 3 倍时，由硬件置位该标志。这意味着要么不需要调整 TRIM 值，要么通过一次微调步骤就足以补偿频率误差。若置位了 CRS_CR 寄存器的 SYNCOKIE 位，则会产生一个中断。由软件置位 CRS_ICR 寄存器的 SYNCOKC 位来清零该位。 0：没有发生同步 OK 1：发生同步 OK 事件。

8.6.4. CRS_ICR（CRS 中断标志清零寄存器）

地址偏移：0x0C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—				ESYNCC	ERRC	SYNCWARNC	SYNCOKC
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:4	NA	保留位，未定义
3	ESYNCC	预期同步清零标志 写 1 清零 CRS_ISR 寄存器的 ESYNCF 标志。
2	ERRC	错误清零标志 写 1 清零 TRIMOVF、SYNCMISS、SYNCERR 位，也因此清零了 CRS_ISR 寄存器的 ERRF 标志。
1	SYNCWARNC	同步警告清零标志 写 1 清零 CRS_ISR 寄存器的 SYNCWARNF 标志。
0	SYNCOKC	同步 OK 清零标志 写 1 清零 CRS_ISR 寄存器的 SYNCOKF 标志

9. 通用输入输出接口（GPIO）

9.1. 简介

GPIO 一共有 a b c d f 六组其中除了 f 组只有 6 个 IO 口和 d 组只有 1 个 IO 外，其他都是 16 个 IO 口。其中每个 IO 口都有四种模式：数字输入、数字输出、复用功能、以及一些添加功能。每个通用 IO 口都 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 GPIOx_ODR) 和 1 个 32 位置位/复位寄存器(GPIOx_BSRR)。A 和 B 还含有 1 个 32 位锁定寄存器(GPIOx_LCKR) 和 2 个 32 位替代功能寄存器(GPIOx_AFRH GPIOx_AFRL)。

9.2. 主要特征

GPIO 模块主要特性如下：

- 输出状态：推挽、开漏和上下拉。
- 输出数据：输出数据寄存器（GPIOx_ODR）或者外围设备即 AF 模式下的输出。
- 每个 IO 的输出速度都可以选择。
- 输入状态：悬空，上下拉，模拟，三种状态。
- 输入数据：输入数据寄存器（GPIOx_IDR）或者外围设备即 AF 模式下的输入。
- Bit 置位和复位寄存器（GPIOx_BSRR 只写不读），低 16 位写 1 表示对 GPIOx_ODRy 置位（可位操作 y=0…15），高 16 位写 1 表示对 GPIOx_ODRy 复位（可位操作 y=0…15）。
- 锁定寄存器（GPIOx_LCKR）可以锁定 PORTA 和 PORTB 的状态。
- 做模拟功能使用。
- 复用功能选择寄存器。
- IO 的快速翻转能力为每两个时钟周期翻转一次。
- 高度灵活的复用功能可以使用户作普通 GPIO 使用或者是 AF 复用功能使用。
- 总共有 55 个可以做数字使用的 IO。

9.3. 功能描述

根据每个 IO 端口的硬件性特征，每个 GPIO 都能由软件独立配置成以下模式

- 输入悬空、上拉或者下拉模式。
- 模拟模式。
- 上下拉可控的推挽或者开漏输出
- 上下拉可控的复用功能输出模式。

每个 IO 端口可以自由编程，而且 IO 的控制寄存器都是按 32 位字、半字或者字节访问的。置位/复位寄存器 GPIOx_BSRR 或者 GPIOx_BRR 可以实现对寄存器 GPIOx_ODR 的位读/改写，在读/更改访问之间产生 IRQ 中断也不会发生危险。

图 9.1 展示了 IO 的结构图，表 9.1 给出了端口的一些可能的配置。

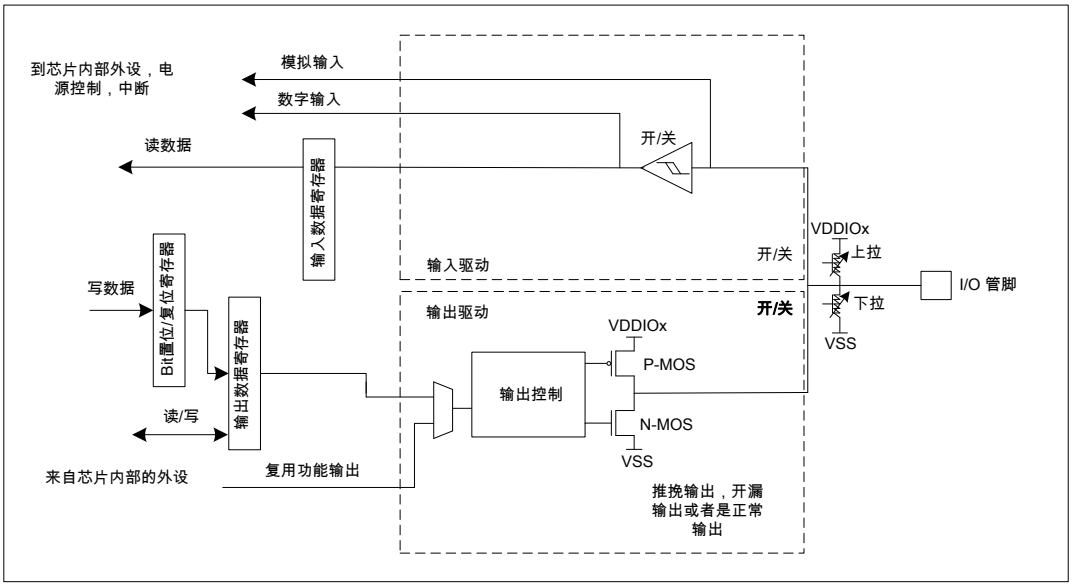


图 9.1 标准 IO 的结构

表 9.1 端位配置表口

MODER(i) [1:0]	OTYPER(i)	OSPEEDR(i) [1:0]		PUPDR(i) [1:0]		I/O 配置	
01	0	SPEED[1:0]		0	0	输出	推挽
	0			0	1	输出	推挽+上拉
	0			1	0	输出	推挽+下拉
	0			1	1	保留	
	1			0	0	输出	开漏
	1			0	1	输出	开漏+上拉
	1			1	0	输出	开漏+下拉
	1			1	1	保留（开漏）	
10	0	SPEED[1:0]		0	0	复用输出	推挽
	0			0	1	复用输出	推挽+上拉
	0			1	0	复用输出	推挽+下拉
	0			1	1	保留	
	1			0	0	复用输出	开漏
	1			0	1	复用输出	开漏+上拉
	1			1	0	复用输出	开漏+下拉
	1				1	保留	
00	x	x	x	0	0	输入	浮空
	x	x	x	0	1	输入	上拉
	x	x	x	1	0	输入	下拉
	x	x	x	1	1	上下拉同时打开	
11	x	x	x	0	0	输入输出	模拟模式
	x	x	x	0	1	保留	
	x	x	x	1	0		
	x	x	x	1	1		

9.3.1. 通用 GPIO

复位期间和复位后，复用功能未开启且大部分的 IO 端口被配置为输入悬空模式。
复位之后调试引脚被配置为复用功能的上拉/下拉模式：

- PA14：SWCLK 置于下拉模式
- PA13：SWDIO 置于上拉模式

当配置为输出模式时，写到输出数据寄存器（GPIOx_ODR）上的值输出到相应的引脚上去。而输入数据寄存器（GPIOx_IDR）在每个 AHB 的时钟下捕捉 IO 管脚上的的数据。同时所有的 GPIO 都受 GPIOx_PUPDR 寄存器的控制可以实现 IO 内部的弱上拉或是弱下拉。

9.3.2. IO 端口的复用功能和重映射

器件的 IO 口线是通过多路选择器连接到内部的各个外设模块，同一时刻仅允许外设的复用功能一个引脚连到 GPIO 的口线上，因此，同一口线上不能有冲突的引脚分配。
每个 IO 管脚有 7 种复功能的多路输入选择器，可以通过配置 GPIOx_AFRL（控制 7-0）和 GPIOx_AFRH（控制 15-8）来实现复用功能的选择。

- 复位后的 IO 口都是连接到复用功能 0。（IO 通过配置寄存器 GPIOx_MODER 来选择切换到复用功能。）
- 每个 IO 的复用功能在器件手册上都有详细的说明

除了这种灵活的 IO 复用结构，每个外设的复用功能可能映射到不同的 IO，这样可以实现小封装器件上优化更多的可用外设。

为了使用一个给定的 I/O 口配置，你必须按如下的原则执行：

- 调试功能：个器件复位后，这些引脚立即配置为复用功能用来支持调
- GPIO: 在 GPIOx_MODER 寄存器中配置所需的 I/O 口为输出，输入或模拟输入。
- 外设的复用功能：
  - 连接I/O到所需的AFx ，AFx定义在GPIOx_AFRL或GPIOx_AFRH 寄存器中
  - 通过对GPIOx_OTYPER, GPIOx_PUPDR 和GPIOx_OSPEEDER 寄存器来配置相应引脚的上拉/下拉和输出速度
  - 在GPIOx_MODER寄存器中配置所需的I/O口线为复用功能
- 附加功能：
  - 对于ADC和比较器，需在GPIOx_MODER寄存器中配置所需的I/O口线为模拟方式并在ADC或比较器寄存器中配置所需的功能。
  - 对于附加功能如RTC、WKUPx、Flash test/ANALOG_TEST、USB和振荡器，在关联的RTC、PWR、Flash、USB和RCC寄存器配置相应所需的功能。当这些功能配置生效时，GPIO本身的控制寄存器GPIOx_MODER和AFx寄存器无效。

有关详细的 IO 口线复用功能映射，请参考器件手册的“复用功能映射表”。

9.3.3. IO 端口控制寄存器

每组 GPIO 口都有 4 个 32 位的控制寄存器（GPIOx_MODER，GPIOx_OTYPER， GPIOx_OSPEEDR，GPIOx_PUPDR）用来配置多达 16 IO 口线。GPIOx_MODER 寄存器用来选择 IO 模式（如输入，输出、复用或模拟）。GPIOx_OTYPER 和 GPIOx_OSPEEDR 寄存器用来选择输出类型（如推挽或开漏）和速度（中、高、低）。GPIOx_PUPDR 寄存器用来选择上拉/下拉方式。

9.3.4. IO 端口数据寄存器

每组 GPIO 都有两个 16 位数据寄存器：输入数据寄存器（GPIOx_IDR）和输出数据寄存器（GPIOx_ODR）。GPIOx_ODR 是用于存储输出数据寄存器，其可进行读/写访问。从 IO 端口输入的数据存放在 GPIOx_IDR 中，所以该寄存器只可读不可写。

- GPIO 端口输出寄存器：GPIOx_ODR（x=A、B、C、D、F）
- GPIO 端口输入寄存器：GPIOx_IDR（x=A、B、C、D、F）

9.3.5. IO 数据位处理

Bit 置位和复位寄存器（GPIOx_BSRR）是一个 32 位寄存器，其允许应用对输出数据寄存器（GPIOx_ODR）的每个位进行置位和复位操作。Bit 置位和复位寄存器的有效数据宽度是 GPIOx_ODR 有效数据宽度的 2 倍。对于 GPIOx_ODR 中的每位，在 GPIOx_BSRR 中有两位与之对应： BS(i) 和 BR(i)。当对位 BS(i)写 1 时则设置相应的 ODR(i) 位。当对 BR(i) 写 1 时，则复位相应的 ODR(i) 位。对 GPIOx_BSRR 中的任意位写 0 都不会影响 GPIOx_ODR 寄存器的值。若对 GPIOx_BSRR 的 BS(i) 和 BR(i) 同时置 1 ，那么其置位用一操作具有优先权(即对相应位做置位操作)。用 GPIOx_BSRR 寄存器来改变 GPIOx_ODR 的相应位并不会锁定 GPIOx_ODR 的每 bit， GPIOx_ODR 位也可直接从这个寄存器本身进行访问。GPIOx_BSRR 寄存器提供对 GPIOx_ODR 寄存器原子位操作处理机制。

GPIOx_ODR 用 GPIOx_BSRR 置位或复位的访问机制不需要软件去关闭中断来访问 GPIOx_ODR，在一个 AHB 写访问周期可以改变 1 位或多位数据。

9.3.6. GPIO 锁定机制

个特定对 GPIOx_LCKR（x=A、B） 寄存器的写序列来冻结端口 A 和端口 B 的控制寄存器是可行的。被锁定的寄存器有：GPIOx_MODER，GPIOx_OTYPER，GPIOx_OSPEEDR，GPIOx_PUPDR， GPIOx_AFRL 和 GPIOx_AFRH。

要写 GPIOx_LCKR 寄存器需要一个特定的读/写序列。当正确的锁定序列作用于这个寄存器的第 16bit 时 LCKR[15:0] 的值用来锁定 IO 口的配置（在写序列期间要保持 LCKR[15:0]值不变）。当锁定序列已经作用于一个端口位，该端口位的值再也不能改变直到下一次复位。每个 GPIOx_LCKR 位冻结控制寄存器中(GPIOx_MODER ,GPIOx_OTYPER ,GPIOx_OSPEEDR，GPIOx_PUPDR ,GPIOx_AFRL 和 GPIOx_AFRH)的相应位。

9.3.7. IO 复用功能输入输出

每个 IO 端口都有一些复用的功能，作为多个内部外设模块的输入或者输出。通过配置寄存器 GPIOx_AFRL 和 GPIOx_AFRH 来选择相应的复用功能。

具体的复用功能被复用在哪些端口上详情参考数据手册

9.3.8. 外部中断/唤醒线

所有的 GPIO 都可以作为外部中断使用，在当做中断使用时必须配置为输入模式，具体详情可参考外部中断和事件控制器（EXTI）。

9.3.9. 输入配置

当 IO 口编程配置为输入时：

- 该输出缓冲区禁用
- 施密特触发器输入激活
- 由 GPIOx_PUPDR 寄存器的值来激活上拉和下拉电阻
- 在每个 AHB 时钟周期 IO 端口上的数据被采样进入输入数据寄存器
- 用对输入数据寄存器的读访问来获取 IO 口状态

如下图给出 IO 端口位的输入配置

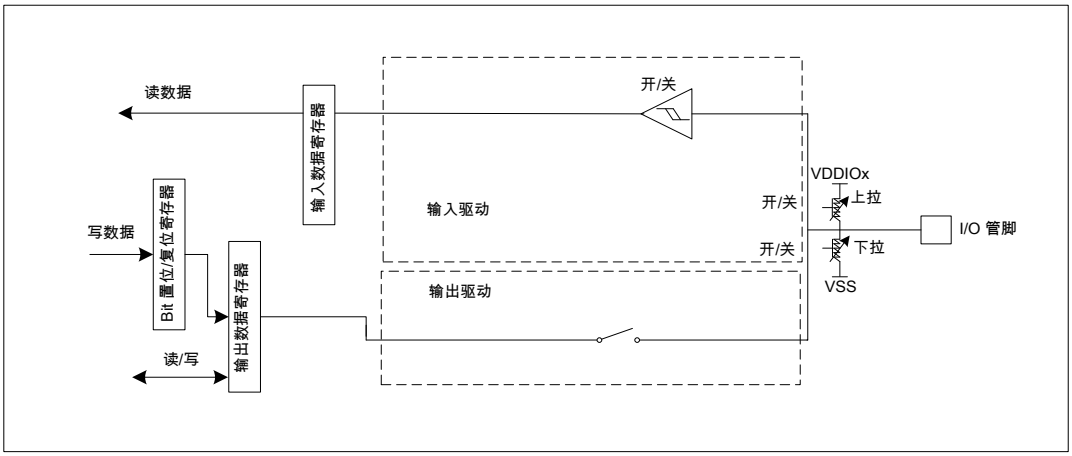


图 9.2 输入浮空/上拉/下拉配置

9.3.10. 输出配置

当 IO 口编程配置为输入时：

- 输出缓冲开启：
- 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(P-MOS 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
- 弱上拉和弱下拉电阻是否激活取决于 GPIOx_PUPDR 寄存器的值
- 在每个 AHB 时钟周期 I/O 引脚上的数据被采样进入输入数据寄存器
- 用对输入数据寄存器的读访问来获取 I/O 口状态
- 用对输出寄存器的读访问来获取最后写进该寄存器的值

如下图给出了 IO 端口位的输出配置。

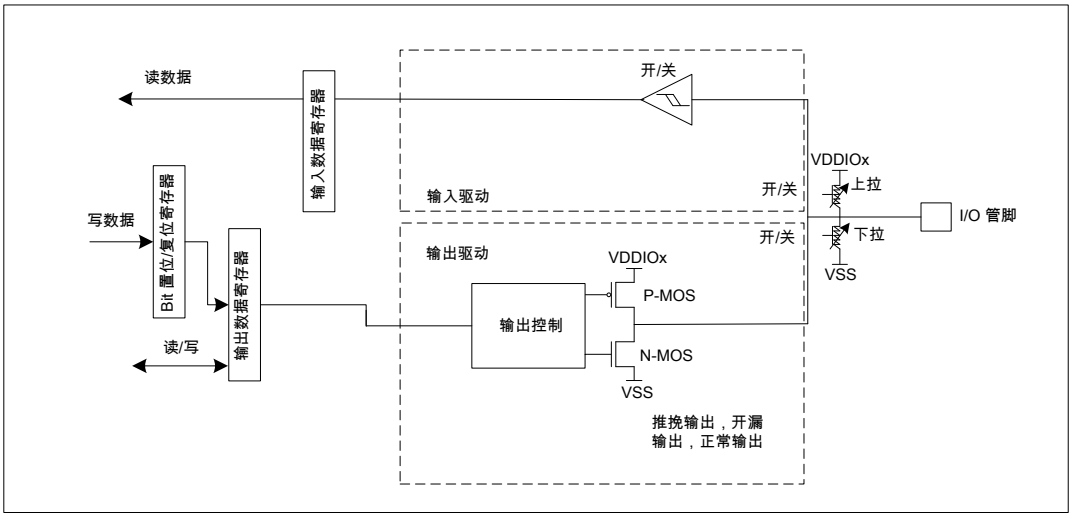


图 9.3 输出配置

9.3.11. 复用功能配置

当 IO 端口被配置为复用功能时：

- 在开漏或推挽模式下输出缓冲器可被配置
- 外设的信号驱动输出缓冲器
- 施密特触发输入被激活
- 弱上拉和弱下拉电阻是否激活取决于 GPIOx_PUPDR 寄存器的值
- 在每个AHB 时钟周期I/O引脚上的数据被采样进入输入数据寄存器
- 用对输入数据寄存器的读访问来获取 IO 口状态

  如下图给出了 I/O 端口位的复用功能配置。

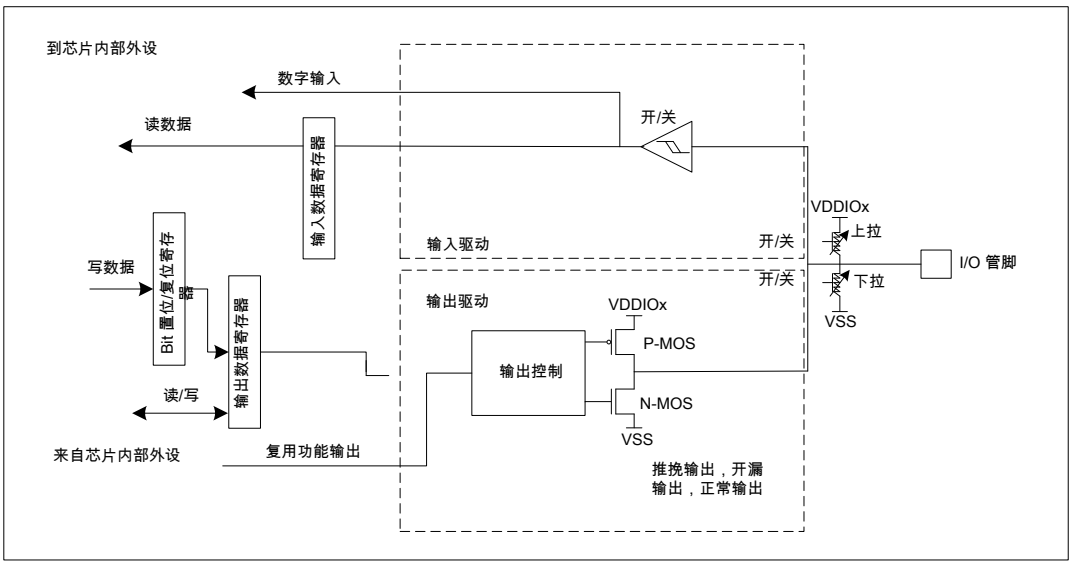


图 9.4 复用功能的配置

9.3.12. 模拟配置

当 IO 端口编程为模拟配置时：

- 输出缓冲器关闭
- 禁止施密特触发输入，实现了每个模拟 IO 引脚上的零消耗。施密特触发输出值被强置为'0'
- 弱上拉和下拉电阻被禁止
- 读取输入数据寄存器时数值为 0

如下图给出了 IO 端口位的高阻抗模拟输入配置。

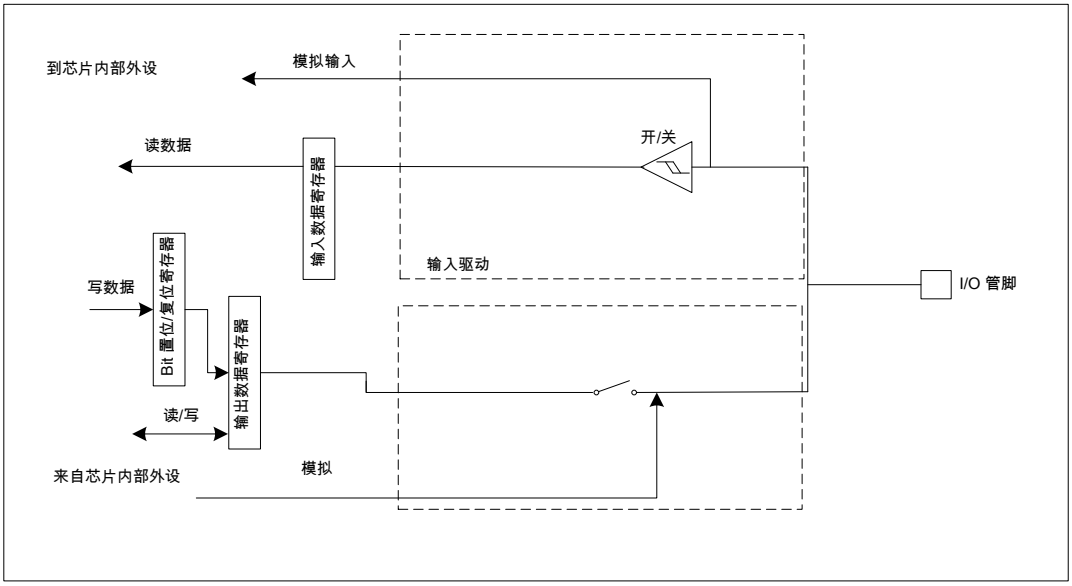


图 9.5 高阻抗模拟配置

9.3.13. LED 驱动模式设置

部分 GPIO 用来做 LED 驱动，但是 LED 驱动的时候需要较小的源电流和较大的灌电流，而普通的 GPIO 则源电流和灌电流差不多大。所以为了满足我们应用的需求，我们需要用寄存器（GPIOx_LEDM ）对 GPIO 进行控制来切换选择合适驱动模式。
拥有可以切换电流特性的 GPIO 一共有 13 个，分别为 PB0、PB1、PB3、PB4、PB5、PB6、PB7、PA8、PA9、PA10、PA13、PA14、PA15。

9.3.14. GPIO 做 HSE 或 LSE 引脚

当 HSE 或 LSE 振荡器关断时(复位后的缺省状态) ,相关振荡器引脚可以用做普通的 GPIO 口。当 HSE 或 LSE 振荡器开启(在 RCC_CSR 寄存器设置 HSEON 或 LSEON 位来开启)振荡器控制其相关引脚且相关引脚的 GPIO 配置无效。当振荡器配置为用户外部时钟方式，仅使用 OSC_IN 或 OSC32_IN 引脚做为时钟输入脚，OSC_OUT 或 OSC32_OUT 引脚可仍然配置为正常的 GPIO 引脚。

9.3.15. GPIO 做运放 OP0 的模拟输入脚

P0 运放也是一样，当配置完 P0 相关寄存器之后相应的（PA0、PA1、PA2）管脚也会被切成模拟模式。此时 GPIO 的配置寄存器对 GPIO 的配置无效。具体比较器的配置详情参考运放 P0 模块。

9.3.16. GPIO 做 I2C1 功能的快速模式

I2C1 快速模式：配置 I2C1 的寄存器改变在 I2C1 复用功能下的 pin 的驱动能力。（PA6、PA7、PA8、PA9）关于寄存器的配置请参考 I2C1 模块。

9.3.17. GPIO 做 TOUCH 功能的配置

当 GPIO 选择 AF 功能且选择是 TOUCH 功能的时候有个特点就是此时 GPIO 的施密特是关闭的状态。

9.3.18. 附加功能中的复用关系

一般附加功能都是不考虑 GPIO 本身控制寄存器的值而直接去控制 IO 端口的状态，但是一个 IO 端口如果有两个以上的附加功能就会存在有竞争关系，因此就会有优先级之分。

下表就表示带有多个附加功能的 IO 端口各附加功能之间的优先级。

表 9.2 端口附加功能的优先级

PIN	优先级从高到低从左往右依次排列			
PC13	RTC（输入>输出）	weak up	无	无
PC14	OSC	CLK_IN	RTC	无
PC15	OSC	RTC	无	无
PA0	ANALOG TEST	RTC	weak up	运放 P0
PA9	TM_PLL_DIV_CLK	TM_PMU_CPCLK	无	无

9.4. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	GPIOA_MODER	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]		MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
	Reset	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	GPIOx_MODER (where x = B..D , F)	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]		MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	GPIOx_OTYPER (where x = A.. D , F)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	GPIOA_OSPEEDR	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]		OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
	Reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	GPIOx_OSPEEDR (where x =B..D , F)	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]		OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOA_PUPDR	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]		PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
	Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOx_PUPDR (where x =B..D , F)	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]		PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	GPIOx_IDR (where x = A.. D , F)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x14	GPIOx_ODR (where x = A.. D , F)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	GPIOx_BSRR (where x = A.. D , F)	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	GPIOx_LCKR (where x = A B)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LCKK	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	GPIOx_AFRL (where x = A B)	AFSEL7 [3:0]				AFSEL6 [3:0]				AFSEL5 [3:0]				AFSEL4 [3:0]				AFSEL3 [3:0]				AFSEL2 [3:0]				AFSEL1 [3:0]				AFSEL0 [3:0]			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x24	GPIOx_AFRH (where x = A B)	AFSEL15				AFSEL14				AFSEL13				AFSEL12				AFSEL11				AFSEL10				AFSEL9				AFSEL8			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	GPIOx_BRR (where x = A.. D , F)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	GPIOx_LEDM (where x = A)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LEDM15	LEDM14	LEDM13	-	-	LEDM10	LEDM9	LEDM8	-	-	-	-	-	-	-	-
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	0	0	0	x	x	x	x	x	x	x	x
0x30	GPIOx_LEDM (where x = B)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LEDM7	LEDM6	LEDM5	LEDM4	LEDM3	-	LEDM1	LEDM0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0	0

9.4.1. GPIOx_MODER

端口模式寄存器（x=A、B、C、D、F）

偏移地址：0x00

复位值：

0x2800 0000 （端口 A）

0x0000 0000 （其他端口）

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:0	MODERy(y=15..0)	由软件写来配置 IO 口的模式 00：输入模式（复位状态） 01：通用输出模式 10：复用功能模式 11：模拟模式

9.4.2. GPIOx_OTYPER

端口输出类型寄存器（x=A、B、C、D、F）

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	Function
31:16	NA	保留位

15:0	OTy(y=15..0)	由软件写来配置 IO 口的输出类型 0：推挽输出（复位状态） 1：开漏输出

9.4.3. GPIOx_OSPEEDR

端口输出速递寄存器（x=A、B、C、D、F）

偏移地址：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:0	OSPEEDRy(y=15..0)	由软件写来配置 IO 口的速度 x0：低速 01：中速 11：高速

9.4.4. GPIOx_PUPDR

端口上下垃寄存器

偏移地址：0x0C

复位值：0x2400 0000（端口 A）

复位值：0x0000 0000（其他端口）

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:0	PUPDRy(y=15..0)	由软件写来配置 IO 口的上下拉 00：无上下拉 01：上拉 10：下拉 11：保留 注：当 MODER=2'b00 且 PUPDR=2'b11 时，I/O 上下拉同时打开，此时为 1/2VDD 模式。如果 MODER 不等于 2'b00，PUPDR=2'b11，则无上下拉。

9.4.5. GPIOx_IDR

端口输入数据寄存器（x=A、B、C、D、F）

偏移地址：0x10

复位值：0x0000 xxxx

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位
15:0	IDR[15:0]	端口输入数据，只可读。它们包含相应 IO 口的输入值。

9.4.6. GPIOx_ODR

端口输出数据寄存器（x=A、B、C、D、F）

偏移地址：0x14

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:0	ODR[15:0]	端口输出数据，可读可写。

注：端口置位和复位寄存器可以对 ODR 寄存器置位和复位。

9.4.7. GPIOx_BSRR

端口置位/复位寄存器（x=A、B、C、D、F）

偏移地址：0x18

复位值：0x0000 0000

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

Bit	Name	Function
31:16	BRx (x=15…0)	端口 x 复位，这些寄存器位都是只可写不可读，读的值都为 0. 0：对应的 ODRx 位无影响 1：复位对应的 ODRx 位
15:0	BSx (x=15…0)	端口 x 置位，这些寄存器位都是只可写不可读，读的值都为 0. 0：对应的 ODRx 位无影响 1：置位对应的 ODRx 位

9.4.8. GPIOx_LCKR

端口配置锁定寄存器（x=A、B）

如果给定一个正确的序列写到 LCKK 上，那么这个寄存器就是用来锁定端口位的配置。LCKy（y=15…0）的值用于锁定相应 GPIO 位的配置。在发出写序列期间 LCKR[15:0]的值不会改变。如果端口已经被锁定序列锁定，那么端口的一些配置值将不会被改变直到下次复位到来。

注：一个特定的写序列用于写 GPIOx_LCKR 寄存器。在这个锁定序列期间，仅能字（32 位）访问该寄存器。
每一个锁定位冻结一个指定的配置寄存器（控制和复用功能寄存器）

偏移地址：0x1C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							LCKK
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
15:8	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8

类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:17	NA	保留位
16	LCKK	锁定键 可随时读取。它仅能有锁定写序列来改写。 0：端口配置锁定键不激活。 1：端口配置锁定键激活。GPIOx_LCKR 寄存器锁定直到一个 MCU 复位产生 锁定键写序列： 写（LCKR[16] = 1'b1） + LCKR[15:0] 写（LCKR[16] = 1'b0） + LCKR[15:0] 写（LCKR[16] = 1'b0） + LCKR[15:0] 读 LCKR 读 LCKR[16]= 1'b1(这个操作可选，其作用是确认是否激活) 注：在锁定写序列期间，LCK[15:0]值必须不变 在锁定写序列中出现任何错误都会中断锁定操作。 在第一次锁定序列锁定任意端口之后，所有读 LCKK 位都只会返回 1 直到下次 CPU 复位的到来。
15:0	LCKy (y=15…0)	端口 y 锁写位 y (y= 0..15)可读写，但是只有在 LCKK 为 0 的时候写。 0：端口未配置锁定 1：端口配置锁定

9.4.9. GPIOx_AFRL

复用功能低位寄存器（x=A、B）

偏移地址：0x20

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—	AFRSEL7			—	AFRSEL6		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
23:16	—	AFRSEL5			—	AFRSEL4		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
15:8	—	AFRSEL3			—	AFRSEL2		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
7:0	—	AFRSEL1			—	AFRSEL0		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW

Bit	Name	Function
31:0	AFRSELy(y=7..0)	复用功能低位寄存器可以用来选择端口的复用功能。

		000：AF0 001：AF1 010：AF2 011：AF3 100：AF4 101：AF5 110：AF6 111：AF7

9.4.10. GPIOx_AFRH

复用功能高位寄存器（x=A、B）

偏移地址：0x24

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—	AFRSEL15			—	AFRSEL14		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
23:16	—	AFRSEL13			—	AFRSEL12		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
15:8	—	AFRSEL11			—	AFRSEL10		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW
7:0	—	AFRSEL9			—	AFRSEL8		
类型	RO-0	RW	RW	RW	RO-0	RW	RW	RW

Bit	Name	Function
31:0	AFRSELy(y=15..8)	复用功能低位寄存器可以用来选择端口的复用功能。 000：AF0 001：AF1 010：AF2 011：AF3 100：AF4 101：AF5 110：AF6 111：AF7

9.4.11. GPIOx_BRR

端口复位寄存器（x=A、B、C、D、F）

偏移地址：0x28

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8
类型	WO	WO	WO	WO	WO	WO	WO	WO
7:0	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
类型	WO	WO	WO	WO	WO	WO	WO	WO

Bit	Name	Function
31:16	NA	保留位
15:0	BRx (x=15…0)	端口 x 复位，这些寄存器位都是只可写不可读，读的值都为 0. 0：对应的 ODRx 位无影响 1：复位对应的 ODRx 位

9.4.12. GPIOA_LEDM

LED 模式驱动设置寄存器

偏移地址：0x30

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	LEDM15	LEDM14	LEDM13	—		LEDM10	LEDM9	LEDM8
类型	RW	RW	RW	RO-0	RO-0	RW	RW	RW
7:0	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:16	NA	保留位
15:0	LEDMx (x=15、14、13、10、9、8)	LED 模式驱动设置寄存器： 0：普通 IO 模式，源电流 22mA，灌电流 29mA 1：LED 驱动模式，源电流 3mA，灌电流 58mA

9.4.13. GPIOB_LEDM

LED 模式驱动设置寄存器

偏移地址：0x30

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31 : 24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	LEDM7	LEDM6	LEDM5	LEDM4	LEDM3	—	LEMD1	LEDM0
类型	RW	RW	RW	RW	RW	RO-0	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:0	LEDMx (x=7、6、5、4、3、1、0)	LED 模式驱动设置寄存器： 0：普通 IO 模式，源电流 22mA，灌电流 29mA 1：LED 驱动模式，源电流 3mA，灌电流 58mA

10. 系统配置控制器（SYSCFG）

主要功能

4. 在部分 IO 口上启用或禁用 I2C 超快模式（Fast Mode Plus）
5. 重映射部分 DMA 触发源到其他不同的 DMA 通道上
6. 重映射存储器到代码起始区域
7. 管理连接到 GPIO 口的外部中断
8. 管理系统的可靠性

10.1. 系统配置控制器寄存器

offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SYSCFG_CFGR1	-	-	-	-	-	-	-	-	-	-	-	I2C1_FMP	I2C_PB9_FMP	I2C_PB8_FMP	I2C_PB7_FMP	I2C_PB6_FMP	-	-	-	TIM17_DMA-RMP	TIM16_DMA-RMP	USART1_RX_DMA_RMP	USART1_TX_DMA_RMP	ADC_DMA_RMP	IRDA_ENV_SEL[1:0]		-	-	-	-	MEM_MODE	
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x
0x08	SYSCFG_EXTICR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EXTI3[2:0]				EXTI2[2:0]				EXTI1[2:0]				EXTI0[3:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	SYSCFG_EXTICR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EXTI7[2:0]				EXTI6[2:0]				EXTI5[2:0]				EXTI4[2:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SYSCFG_EXTICR3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EXTI11[2:0]				EXTI10[2:0]				EXTI9[2:0]				EXTI8[2:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	SYSCFG_EXTICR4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EXTI5[2:0]				EXTI4[2:0]				EXTI13[2:0]				EXTI12[2:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	SYSCFG_CFGR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PVD_LOCK	-	LOCKUP_LOCK
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0

10.1.1. SYSCFG_CFGR1

偏移地址：0x00

复位值：0x0000 000X

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	—	—	—	I2C1_FMP	I2C_PB9_FMP	I2C_PB8_FMP	I2C_PB7_FMP	I2C_PB6_FMP
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	—	—	—	TIM17_DMA_RMP	TIM16_DMA_RMP	USART1_RX_DMA_RMP	USART1_TX_DMA_RMP	ADC_DMA_RMP
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	IRDA_ENV_SEL[1:0]		—				MEM_MODE	
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:21	NA	通用寄存器位，无实际意义
20	I2C1_FMP	I2C1 超快模式（FM+）驱动能力配置选择位 0：超快模式仅仅由 I2C_Px_FMP 位控制 1：超快模式由 GPIOx_AFR 进行选择，没有配置 I2C1_Px_FMP 位时，只能通过 GPIOx_AFR 进行控制
19:16	I2C_PBx_FMP	I2C1 超快模式（FM+）驱动能力配置位，x 位 6~9 0：PBx 管脚设置为标准模式 1：PBx 管脚配置为超快模式
15:13	NA	通用寄存器位，无实际意义
12	TIM17_DMA_RMP	TIM17 DMA 请求重映射位，该位由软件置位和清零。 0：无重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射到 DMA 通道 1 上） 1：重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射到 DMA 通道 2 上）
11	TIM16_DMA_RMP	TIM16 DMA 请求重映射位，该位由软件置位和清零。 0：无重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射到 DMA 通道 3 上） 1：重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射到 DMA 通道 4 上）
10	USART1_RX_DMA_RMP	USART1_RX DMA 请求重映射控制位 0：无重映射（USART1_RX 请求映射在 DMA 通道 3 上） 1：重映射（USART1_RX 请求映射在 DMA 通道 5 上）
9	USART1_TX_DMA_RMP	USART1_TX DMA 请求重映射控制位 0：无重映射（USART1_TX 请求映射在 DMA 通道 2 上）

		1：重映射（USART1_TX 请求映射在 DMA 通道 4 上）
8	ADC_DMA_RMP	ADC DMA 请求重映射控制位 0：无重映射（ADC DMA 请求映射在 DMA 通道 1 上） 1：重映射（ADC DMA 请求映射在 DMA 通道 2 上）
7:6	IRDA_ENV_SEL[1:0]	红外调制包络信号源选择位 00: TIM16 01:USART1 10:USART2 11:保留位
5:2	NA	通用寄存器位，无实际意义
1:0	MEM_MODE[1:0]	存储映射选择位 x0：主闪存存储器映射到 0x00000000 01：系统闪存映射到 0x00000000 11：嵌入式 RAM 映射到 0x00000000

10.1.2. SYSCFG_EXTICR1

偏移地址：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	EXTI3[3:0]				EXTI2[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	EXTI1[3:0]				EXTI0[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:12	EXTI3[3:0]	EXTI 3 线配置位 0000：PA3 管脚 0001：PB3 管脚 0010：PC3 管脚 0011：保留 0100：保留 0101：保留 其他：保留
11:8	EXTI2[3:0]	EXTI 2 线配置位 0000：PA2 管脚

		0001：PB2 管脚 0010：PC2 管脚 0011：PD2 管脚 0100：保留 0101：保留 其他：保留
7:4	EXTI1[3:0]	EXTI 1 线配置位 0000：PA1 管脚 0001：PB1 管脚 0010：PC1 管脚 0011：保留 0100：保留 0101：PF1 管脚 其他：保留
3:0	EXTI0[3:0]	EXTI 0 线配置位 0000：PA0 管脚 0001：PB0 管脚 0010：PC0 管脚 0011：保留 0100：保留 0101：PF0 管脚 其他：保留

10.1.3. SYSCFG_EXTICR2

偏移地址：0x0C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	EXTI7[3:0]				EXTI6[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	EXTI5[3:0]				EXTI4[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:12	EXTI7[3:0]	EXTI 7 线配置位 0000：PA7 管脚

		0001：PB7 管脚 0010：PC7 管脚 0011：保留 0100：保留 0101：PF7 管脚 其他：保留
11:8	EXTI6[3:0]	EXTI 6 线配置位 0000：PA6 管脚 0001：PB6 管脚 0010：PC6 管脚 0011：保留 0100：保留 0101：PF6 管脚 其他：保留
7:4	EXTI5[3:0]	EXTI 5 线配置位 0000：PA5 管脚 0001：PB5 管脚 0010：PC5 管脚 0011：保留 0100：保留 0101：PF5 管脚 其他：保留
3:0	EXTI4[3:0]	EXTI 4 线配置位 0000：PA4 管脚 0001：PB4 管脚 0010：PC4 管脚 0011：保留 0100：保留 0101：PF4 管脚 其他：保留

10.1.4. SYSCFG_EXTICR3

偏移地址：0x10

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	EXTI11[3:0]				EXTI10[3:0]			

类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	EXTI9[3:0]				EXTI8[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:12	EXTI11[3:0]	EXTI 11 线配置位 0000：PA11 管脚 0001：PB11 管脚 0010：PC11 管脚 0011：保留 0100：保留 0101：保留 其他：保留
11:8	EXTI10[3:0]	EXTI 10 线配置位 0000：PA10 管脚 0001：PB10 管脚 0010：PC10 管脚 0011：保留 0100：保留 0101：保留 其他：保留
7:4	EXTI9[3:0]	EXTI 9 线配置位 0000：PA9 管脚 0001：PB9 管脚 0010：PC9 管脚 0011：保留 0100：保留 0101：保留 其他：保留
3:0	EXTI8[3:0]	EXTI 8 线配置位 0000：PA8 管脚 0001：PB8 管脚 0010：PC8 管脚 0011：保留 0100：保留 0101：保留 其他：保留

10.1.5. SYSCFG_EXTICR4

偏移地址：0x14

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	EXTI15[3:0]				EXTI14[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	EXTI3[3:0]				EXTI2[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位
15:12	EXTI15[3:0]	EXTI 15 线配置位 0000：PA15 管脚 0001：PB15 管脚 0010：PC15 管脚 0011：保留 0100：保留 0101：保留 其他：保留
11:8	EXTI14[3:0]	EXTI 14 线配置位 0000：PA14 管脚 0001：PB14 管脚 0010：PC14 管脚 0011：保留 0100：保留 0101：保留 其他：保留
7:4	EXTI13[3:0]	EXTI 13 线配置位 0000：PA13 管脚 0001：PB13 管脚 0010：PC13 管脚 0011：保留 0100：保留 0101：保留 其他：保留
3:0	EXTI12[3:0]	EXTI 12 线配置位 0000：PA12 管脚

		0001：PB12 管脚 0010：PC12 管脚 0011：保留 0100：保留 0101：保留 其他：保留

10.1.6. SYSCFG_CFGR2

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					PVD_LOCK	—	LOCKUP_LOCK
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RW

Bit	Name	Function
31:3	NA	保留位,禁止写 1
2	PVD_LOCK	可编程电压检测（PVD）锁定使能位，该位由软件置位系统复位清零 0：PVD 中断没有连接到 TIM1/15/16/17 刹车输入端，PVDE 和 PLS[2:0]位可以进行软件写操作 1：PVD 中断连接到 TIM1/15/16/17 刹车输入端，PVDE 和 PLS[2:0]位只能进行软件读操作
1	NA	保留位
0	LOCKUP_LOCK	Cortex-M0 LOCKUP 输出到 TIM 刹车输入的使能位 0：Cortex-M0 LOCKUP 输出没有连接到 TIM1/15/16/17 刹车输入端 1：Cortex-M0 LOCKUP 输出连接到 TIM1/15/16/17 刹车输入端

11. 直接存储器访问（DMA）

11.1. 介绍

直接存储器访问（DMA）用于在外设和存储器之间以及存储器到存储器之间提供高速数据传输。数据可以通过 DMA 快速移动，而不需要任何 CPU 操作。这使得 CPU 资源可以用于其他操作。
DMA 控制器有 5 个通道，每个通道用于管理来自一个或多个外围设备的存储器访问请求。有一个仲裁器来处理 DMA 请求之间的优先级。

11.2. DMA 主要特性

- 5 个独立可配置的通道（请求）
- 每一个通道都连接到专用的硬件 DMA 请求，每个通道还支持软件触发。这个配置是由软件完成的。
- DMA 通道的请求之间的优先级是软件可编程的（4 个级别，包括非常高、高、中、低），或在同等情况下的硬件顺序（请求 1 优先于请求 2，等等）
- 独立的源和目标传输大小（字节、半字、字），模拟打包和解包过程。源/目标地址必须与数据大小对齐。
- 支持循环缓冲区管理
- 对于每个通道有 3 个事件标志（DMA 半传输，DMA 传输完成和 DMA 传输错误），它们逻辑或在一起产生单个中断请求。
- 存储器到存储器传输
- 外设到存储器传输，存储器到外设传输，外设到外设传输
- 可作为源和目标访问外设
- 可编程的数据传输数量：可达 4095

11.3. DMA 功能描述

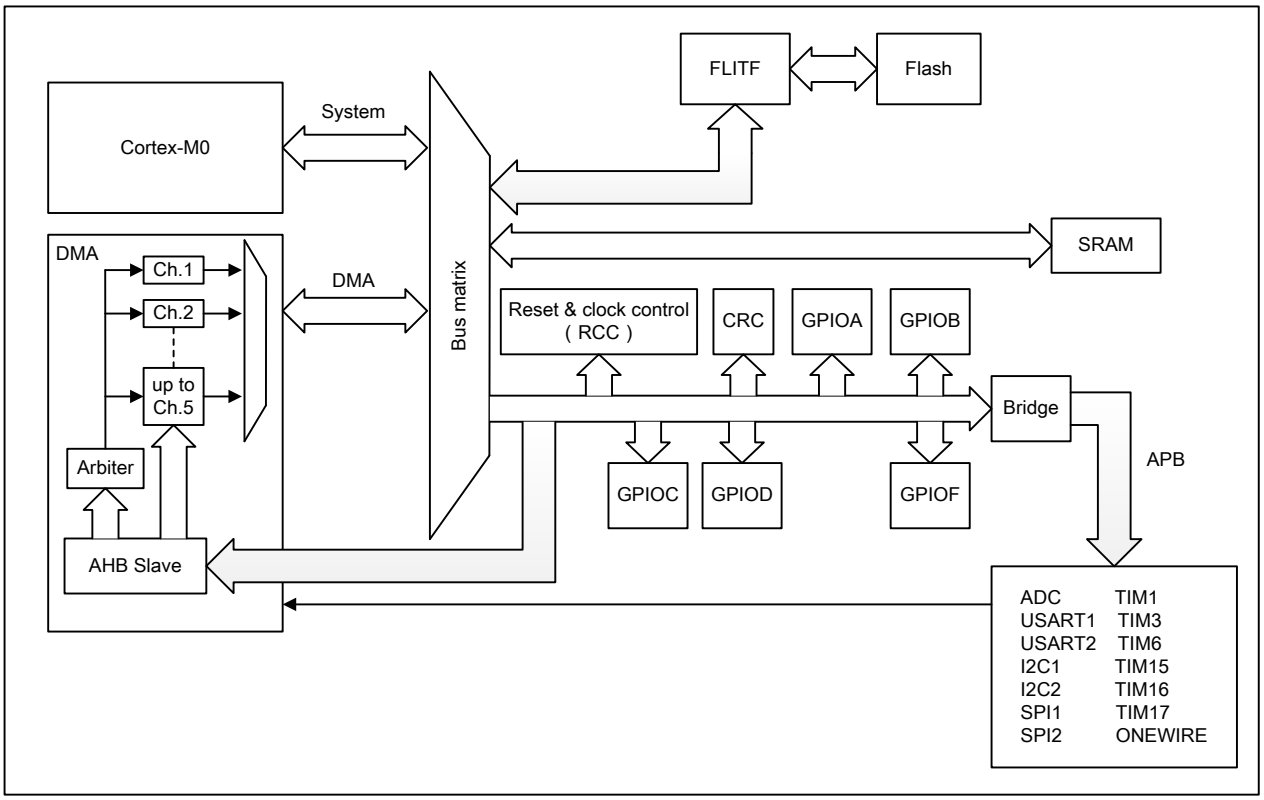


图 11.1. DMA 框图

DMA 控制器通过与 ARM® Cortex®-M0 内核共享系统总线来执行直接存储器访问。当 CPU 和 DMA 针对相同的目标（存储器或者外设）时，DMA 请求可能会停止 CPU 访问系统总线，这可能需要一些总线周期。总线矩阵实现轮询调度，从而保证了至少有一半的系统总线带宽（到存储器和到外设）给 CPU。

11.3.1. DMA 事务

事件发生后 ,外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道优先级为请求提供服务。只要 DMA 控制器访问外设，DMA 控制器就会向外设发送一个应答信号。外设一旦接收到 DMA 控制器的应答信号，就释放它的请求。一旦外设释放请求，DMA 控制就释放应答。如果有更多的请求，外设可以发起下一次事务请求。

综上所述，每个 DMA 传输包含三个操作：

- 从外设数据寄存器或者通过内部当前外设/存储器地址寄存器寻址的存储器位置加载数据。用于第一传输的起始地址是在 DMA_CPARx 或者 DMA_CMARx 寄存器中编程的基础外设/存储器地址。
- 将要存储的数据加载到外设数据寄存器或者通过内部当前外设/存储器地址寄存器寻址的存储器位置。用于第一传输的起始地址是在 DMA_CPARx 或者 DMA_CMARx 寄存器中编程的基础外设/存储器地址。
- DMA_CNDTRx 寄存器进行递减操作，其包含仍需执行的事务数。

11.3.2. 仲裁器

仲裁器根据通道请求的优先级管理它们，并启动外设/存储器的访问序列。
优先级分为两种阶段管理：

- 软件阶段：每个通道优先级都可以在 DMA_CCRx 寄存器中配置。有四个级别：
  - 非常高优先级
  - 高优先级
  - 中等优先级
  - 低优先级
- 硬件阶段：如有两个请求具有相同的软件优先级，则编号小的通道将优先于编号大的通道。例如，通道 2 优先于通道 4。

11.3.3. DMA 通道

每个通道都可以处理位于固定地址的外设寄存器和存储器地址之间的 DMA 传输。要传输的数据量(高达 4095) 是可编程的。包含要传输的数据项数量的寄存器在每次事务之后递减。

可编程数据大小
通过 DMA_CCRx 寄存器中的 PSIZE 和 MSIZE 位，可以完全变成外设和存储器的传输数据大小。

指针增量
根据 DMA_CCRx 寄存器中的 PINC 和 MINC 位，可以选择在每个事务之后自动对外设指针和存储器指针进行后置递增。如果启动了递增模式，则下一次传输的地址将会根据所选的数据大小来递增 1、2、4。第一次传输的地址是在 DMA_CPARx/DMA_CMARx 寄存器中编程的地址。在传输操作过程中，这些寄存器保留初始的编程值。当前的传输地址（在当前的内部外设/存储器地址寄存器中）不能被软件访问。
如果通道是在非循环模式下配置的，那么在最后一次传输之后就不会提供 DMA 请求服务（即当要传输的数据项数量到零之后）。为了重新加载新的数据项数量到 DMA_CNDTRx 寄存器，必须禁用 DMA 通道。
注 :如果一个 DMA 通道被禁用 ,其 DMA 寄存器不会被复位。DMA 通道寄存器(DMA_CCRx ,DMA_CPARx，DMA_CMARx）保留在通道配置阶段编写的初始值。
在循环模式中，在最后一次传输之后，DMA_CNDTRx 寄存器会自动重新加载初始编程值。当前的内部地址寄存器会重新加载 DMA_CPARx/DMA_CMARx 寄存器中的基础地址。

通道配置过程
应该按照以下顺序配置 DMA 通道 x（其中 x 时通道号）:

1. 在 DMA_CPARx 寄存器中设置外设寄存器地址。在外设事件发生后 ,数据将从这个地址移动到存储器中，或者从存储器移动到这个地址。
2. 在 DMA_CMARx 寄存器中设置存储器地址。在外设事件发生后，数据将写入这个存储地址，或者从这个存储地址读取。
3. 配置数据总量到 DMA_CNDTRx 寄存器中。在每次外设事件发生后，该值将会递减。
4. 使用 DMA_CCRx 寄存器中的 PL[1:0]位配置通道优先级。
5. 在 DMA_CCRx 寄存器中配置数据传输方向、循环模式、外设和存储器递增模式、外设和存储器数据大小、以及半传输和/或全传输的中断。

6. 通过设置 DMA_CCRx 寄存器中的 ENABLE 位来激活通道。

一旦使能了通道，它就可以服务任何连接到该通道的外设 DMA 请求。

一旦有一半的字节被传输，半传输标志（HTIF）会被置位，并且如果置位了半传输中断使能位（HTIE）则还会产生一个中断。在传输结束的时候，传输完成标志（TCIF）会被置位，并且如果置位了传输完成中断使能位（TCIE）则还会产生一个中断。

循环模式

循环模式可用于处理循环缓冲区和连续数据流(例如 ADC 扫描模式)。该功能通过 DMA_CCRx 寄存器的 CIRC 位来使能。当循环模式被激活时，要传输的数据数量会自动重载通道配置阶段编程的初始值，并且 DMA 请求会继续被服务。

存储器到存储器模式

DMA 通道也可以工作在没有外设请求触发的情况下。该模式叫做存储器到存储器模式。

如果置位了 DMA_CCRx 寄存器的 MEM2MEM 位，则一旦软件置位 DMA_CCRx 寄存器的使能位（EN），那么该通道就会马上发起传输。一旦 DMA_CNDTRx 寄存器到零，传输就会停止。存储器到存储器模式不能与循环模式同时使用。

11.3.4. 可编程的数据宽度、数据对齐方式和数据字节顺序

当 PSIZE 和 MSIZE 不相等时，DMA 会执行一些数据对齐操作，如表 11.1：可编程的数据宽度和字节顺序行为所示。

表 11.1. 可编程的数据宽度和字节顺序行为

源端口宽度	目标端口宽度	要传输的数据项数量	源内容：地址/数据	传输操作	目标内容：地址/数据
8	8	4	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3	1:READ B0[7:0] @0x0 then WRITE B0[7:0] 0x0 2:READ B1[7:0] @0x1 then WRITE B1[7:0] 0x1 3:READ B2[7:0] @0x2 then WRITE B2[7:0] 0x2 4:READ B3[7:0] @0x3 then WRITE B3[7:0] 0x3	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3
8	16	4	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3	1:READ B0[7:0] @0x0 then WRITE 00B0[15:0] 0x0 2:READ B1[7:0] @0x1 then WRITE 00B1[15:0] 0x2 3:READ B2[7:0] @0x2 then WRITE 00B2[15:0] 0x4 4:READ B3[7:0] @0x3 then WRITE 00B3[15:0] 0x6	@0x0 / 00B0 @0x2 / 00B1 @0x4 / 00B2 @0x6 / 00B3
8	32	4	@0x0 / B0 @0x1 / B1 @0x2 / B2 @0x3 / B3	1:READ B0[7:0] @0x0 then WRITE 000000B0[31:0] 0x0 2:READ B1[7:0] @0x1 then WRITE 000000B1[31:0] 0x4 3:READ B2[7:0] @0x2 then WRITE 000000B2[31:0] 0x8 4:READ B3[7:0] @0x3 then WRITE 000000B3[31:0] 0xC	@0x0 / 000000B0 @0x4 / 000000B1 @0x8 / 000000B2 @0xC / 000000B3
16	8	4	@0x0 / B1B0 @0x2 / B3B2 @0x4 / B5B4 @0x6 / B7B6	1:READ B1B0[15:0] @0x0 then WRITE B0[7:0] 0x0 2:READ B3B2[15:0] @0x2 then WRITE B2[7:0] 0x1 3:READ B5B4[15:0] @0x4 then WRITE B4[7:0] 0x2 4:READ B7B6[15:0] @0x6 then WRITE B6[7:0] 0x3	@0x0 / B0 @0x1 / B2 @0x2 / B4 @0x3 / B6

16	16	4	@0x0 / B1B0 @0x2 / B3B2 @0x4 / B5B4 @0x6 / B7B6	1:READ B1B0[15:0] @0x0 then WRITE B1B0[15:0] 0x0 2:READ B3B2[15:0] @0x2 then WRITE B3B2[15:0] 0x2 3:READ B5B4[15:0] @0x4 then WRITE B5B4[15:0] 0x4 4:READ B7B6[15:0] @0x6 then WRITE B7B6[15:0] 0x6	@0x0 / B1B0 @0x2 / B3B2 @0x4 / B5B4 @0x6 / B7B6
16	32	4	@0x0 / B1B0 @0x2 / B3B2 @0x4 / B5B4 @0x6 / B7B6	1:READ B1B0[15:0] @0x0 then WRITE 0000B1B0[31:0] 0x0 2:READ B3B2[15:0] @0x2 then WRITE 0000B3B2[31:0] 0x4 3:READ B5B4[15:0] @0x4 then WRITE 0000B5B4[31:0] 0x8 4:READ B7B6[15:0] @0x6 then WRITE 0000B7B6[31:0] 0xC	@0x0 / 0000B1B0 @0x4 / 0000B3B2 @0x8 / 0000B5B4 @0xC / 0000B7B6
32	8	4	@0x0 / B3B2B1B0 @0x4 / B7B6B5B4 @0x8 / BBBAB9B8 @0xC / BFBEBDBC	1:READ B3B2B1B0[31:0] @0x0 then WRITE B0[7:0] 0x0 2:READ B7B6B5B4[31:0] @0x4 then WRITE B4[7:0] 0x1 3:READ BBBAB9B8[31:0] @0x8 then WRITE B8[7:0] 0x2 4:READ BFBEBDBC[31:0] @0xC then WRITE BC[7:0] 0x3	@0x0 / B0 @0x1 / B4 @0x2 / B8 @0x3 / BC
32	16	4	@0x0 / B3B2B1B0 @0x4 / B7B6B5B4 @0x8 / BBBAB9B8 @0xC / BFBEBDBC	1:READ B3B2B1B0[31:0] @0x0 then WRITE B1B0[15:0] 0x0 2:READ B7B6B5B4[31:0] @0x4 then WRITE B5B4[15:0] 0x2 3:READ BBBAB9B8[31:0] @0x8 then WRITE B9B8[15:0] 0x4 4:READ BFBEBDBC[31:0] @0xC then WRITE BDBC[15:0] 0x6	@0x0 / B1B0 @0x2 / B5B4 @0x4 / B9B8 @0x6 / BDBC
32	32	4	@0x0 / B3B2B1B0 @0x4 / B7B6B5B4 @0x8 / BBBAB9B8 @0xC / BFBEBDBC	1:READ B3B2B1B0[31:0] @0x0 then WRITE B3B2B1B0[31:0] 0x0 2:READ B7B6B5B4[31:0] @0x4 then WRITE B7B6B5B4[31:0] 0x4 3:READ BBBAB9B8[31:0] @0x8 then WRITE BBBAB9B8[31:0] 0x8 4:READ BFBEBDBC[31:0] @0xC then WRITE BFBEBDBC[31:0] 0xC	@0x0 / B3B2B1B0 @0x4 / B7B6B5B4 @0x8 / BBBAB9B8 @0xC / BFBEBDBC

寻址不支持字节或半字写操作的 AHB 外设

当 DMA 启动一个 AHB 字节或半字写操作时，数据复制到 HWDATA[31:0]总线的未使用通道上。因此，当使用的 AHB 从外设不支持字节或半字操作（当外设不使用 HSIZE 时）且不产生任何错误时，DMA 将以下面的两个例子所示的方式写入 32 位 HWDATA 数据：

- 要写半字“0xABCD”，若 HSIZE 为半字则 DMA 设置 HWDATA 总线为“0xABCDABCD”。
- 要写字节“0xAB”，若 HSIZE 为字节则 DMA 设置 HWDATA 总线为“0xABABABAB”。

假设 AHB/APB 桥是一个不考虑 HSIZE 数据的 AHB 32 位从外设，它会将任何 AHB 字节或半字操作转为 32 位 APB 操作，方法如下：

- 将数据“0xB0”到 0x0（0x1、0x2、0x3）的 AHB 字节写操作会转换为将数据“0xB0B0B0B0”到 0x0 的 APB 字操作。
- 将数据“0xB1B0”到 0x0（0x2）的 AHB 半字写操作会转换为将数据“0xB1B0B1B0”到 0x0 的 APB 字操作。

例如，要写 APB 备份寄存器（与 32 位地址边界对齐的 16 位寄存器），软件必须将存储器源大小（MSIZE）配置为 16 位，将外设目标大小（PSIZE）配置为 32 位。

11.3.5. 错误管理

DMA 传输错误可以通过对保留地址空间的读写来生成。当 DMA 传输错误在 DMA 读写访问期间发生时，错误的通道将通过硬件清除对应通道配置寄存器（DMA_CCRx）中的 EN 位来禁用。DMA_IFR 寄存器的通道传输错误中断标志（TEIF）会被置位，并且若置位了 DMA_CCRx 寄存器的传输错误中断使能位（TEIE）则还会产生一个中断。

11.3.6. DMA 中断

对于每个 DMA 通道，可以在半传输、传输完成或者传输错误的时候产生一个中断。独立的中断使能位可灵活应用。

表 11.2. DMA 中断请求

中断事件	事件标志	使能控制位
半传输	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

DMA 控制器

在进入 DMA 之前，来自外设（TIMx、ADC、SPI、I2C、USART）的硬件请求只是简单的组合逻辑或在一起。这意味着在一个通道上，一次只能使能一个请求。

通过编程对应外设寄存器中的 DMA 控制位，可以独立地激活/撤销外设 DMA 请求。

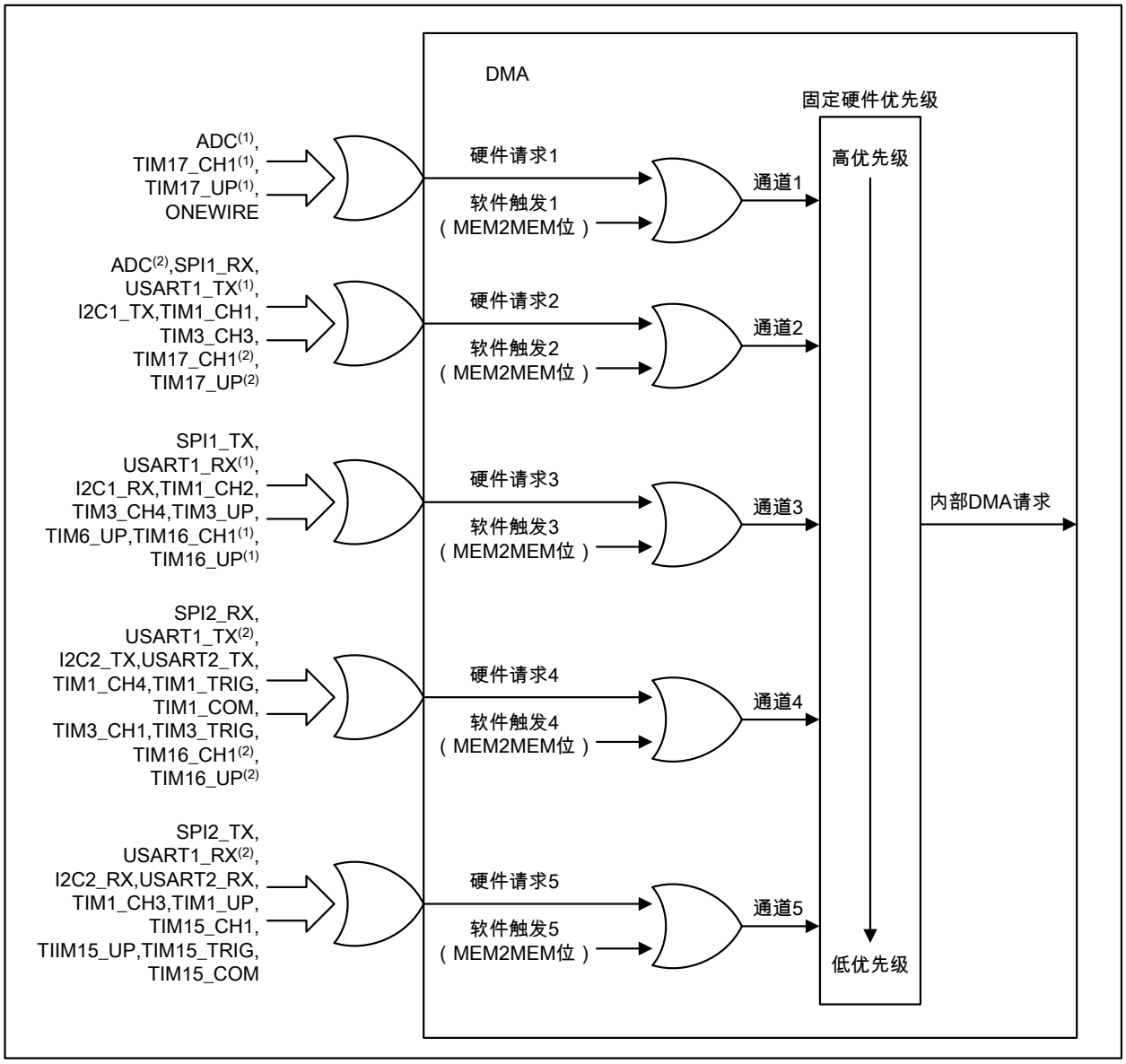


图 11.2. DMA 请求映射图

1. 只有当对应的 SYSCFG_CFGR1 寄存器中的映射位被清零的时候，DMA 请求才会映射到这个 DMA 通道。有关详细信息，请参阅章节 9.1.1：SYSCFG 配置寄存器 1（SYSCFG_CFGR1）。
2. 只有当对应的 SYSCFG_CFGR1 寄存器中的映射位被置位的时候，DMA 请求才会映射到这个 DMA 通道。有关详细信息，请参阅章节 9.1.1：SYSCFG 配置寄存器 1（SYSCFG_CFGR1）。

表 11.3 列出了每个通道的 DMA 请求。

表 11.3. 每个通道的 DMA 请求汇总

外设	通道 1	通道 2	通道 3	通道 4	通道 5
ADC	ADC(1)	ADC(2)	-	-	-
SPI	-	SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX
USART	-	USART1_TX(1)	USART1_RX(1)	USART1_TX(2) USART2_TX	USART1_RX(2) USART2_RX
I2C	-	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TIM1	-	TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_CH3 TIM1_UP
TIM3	-	TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	-
TIM6	-	-	TIM6_UP	-	-
TIM15	-	-	-	-	TIM15_CH1 TIM15_UP TIM15_TRIG TIM15_COM
TIM16	-	-	TIM16_CH1(1) TIM16_UP(1)	TIM16_CH1(2) TIM16_UP(2)	-
TIM17	TIM17_CH1(1) TIM17_UP(1)	TIM17_CH1(2) TIM17_UP(2)	-	-	-

1. 只有当对应的 SYSCFG_CFGR1 寄存器中的映射位被清零的时候，DMA 请求才会映射到这个 DMA 通道。有关详细信息，请参阅章节 9.1.1：SYSCFG 配置寄存器 1（SYSCFG_CFGR1）。
2. 只有当对应的 SYSCFG_CFGR1 寄存器中的映射位被置位的时候，DMA 请求才会映射到这个 DMA 通道。有关详细信息，请参阅章节 9.1.1：SYSCFG 配置寄存器 1（SYSCFG_CFGR1）。

11.4. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DMA_ISR	-	-	-	-	-	-	-	-	-	-	-	-	TEIF5	HTIF5	TCIF5	GIF5	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	DMA_IFCR	-	-	-	-	-	-	-	-	-	-	-	-	CTEIF5	CHTIF5	CTCIF5	CGIF5	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	DMA_CCR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MEM2MEM	PL [1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DMA_CNDTR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	NDT[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x10	DMA_CPAR1	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	DMA_CMAR1	MA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	DMA_CCR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MEM2MEM	PL [1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	DMA_CNDTR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	NDT[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x24	DMA_CPAR2	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	DMA_CMAR2	MA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	DMA_CCR3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MEM2MEM	PL [1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	DMA_CNDTR3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	NDT[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x38	DMA_CPAR3	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	DMA_CMAR3	MA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x44	DMA_CCR4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MEM2MEM	PL [1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	DMA_CNDTR4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	NDT[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	DMA_CPAR4	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	DMA_CMAR4	MA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x58	DMA_CCR5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MEM2MEM	PL [1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	DMA_CNDTR5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	NDT[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x60	DMA_CPAR5	PA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	DMA_CMAR5	MA[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1. DMA_ISR（DMA 中断状态寄存器）

地址偏移：0x00

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—				TEIF5	HTIF5	TCIF5	GIF5
类型	RO-0	RO-0	RO-0	RO-0	RO	RO	RO	RO
15:8	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:20	NA	保留位，未定义
19,15,11,7,3	TEIFx	通道 x 传输错误标志（x=1..5） 该位由硬件置位。由软件写 1 到 DMA_IFCR 寄存器对应位进行清零。 0：在通道 x 上没有传输错误（TE） 1：在通道 x 上发生传输错误（TE）
18,14,10,6,2	HTIFx	通道 x 半传输标志（x=1..5） 该位由硬件置位。由软件写 1 到 DMA_IFCR 寄存器对应位进行清零。 0：在通道 x 上没有半传输（HT）事件 1：在通道 x 上发生半传输（HT）事件
17,13,9,5,1	TCIFx	通道 x 传输完成标志（x=1..5） 该位由硬件置位。由软件写 1 到 DMA_IFCR 寄存器对应位进行清零。 0：在通道 x 上没有传输完成（TC）事件 1：在通道 x 上发生传输完成（TC）事件
16,12,8,4,0	GIFx	通道 x 全局中断标志（x=1..5） 该位由硬件置位。由软件写 1 到 DMA_IFCR 寄存器对应位进行清零。 0：在通道 x 上没有 TE、HT、TC 事件 1：在通道 x 上发生 TE、HT、TC 事件

11.4.2. DMA_IFCR（DMA 中断标志清零寄存器）

地址偏移：0x04

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—				CTEIF5	CHTIF5	CTCIF5	CGIF5
类型	RO-0	RO-0	RO-0	RO-0	W	W	W	W
15:8	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3
类型	W	W	W	W	W	W	W	W
7:0	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
类型	W	W	W	W	W	W	W	W

Bit	Name	Function
31:20	NA	保留位，未定义
19,15,11,7,3	CTEIFx	通道 x 传输错误清零位（x=1..5） 该位由软件置位。 0：无效 1：清零 DMA_ISR 寄存器中的对应 TEIF 标志
18,14,10,6,2	CHTIFx	通道 x 半传输清零位（x=1..5） 该位由软件置位。 0：无效 1：清零 DMA_ISR 寄存器中的对应 HTIF 标志
17,13,9,5,1	CTCIFx	通道 x 传输完成清零位（x=1..5） 该位由软件置位。 0：无效 1：清零 DMA_ISR 寄存器中的对应 TCIF 标志
16,12,8,4,0	CGIFx	通道 x 全局中断清零位（x=1..5） 该位由软件置位。 0：无效 1：清零 DMA_ISR 寄存器中的对应 GIF、TEIF、HTIF、TCIF 标志

11.4.3. DMA_CCRx（DMA 通道 x 配置寄存器）（x=1..5）

地址偏移：0x08 + 0d20 ×（通道编号 - 1）
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]	
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
7:0	MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	MEM2MEM	存储器到存储器模式 该位由软件置位和清零。 0：禁用存储器到存储器模式 1：使能存储器到存储器模式
13:12	PL[1:0]	通道优先级 这些位由软件置位和清零。 00：低 01：中等 10：高 11：非常高
11:10	MSIZE[1:0]	存储器数据大小 这些位由软件置位和清零。 00：8 位 01：16 位 10：32 位 11：保留
9:8	PSIZE[1:0]	外设数据大小 这些位由软件置位和清零。 00：8 位 01：16 位 10：32 位 11：保留
7	MINC	存储器递增模式 该位由软件置位和清零。 0：禁用存储器递增模式

		1：使能存储器递增模式
6	PINC	外设递增模式 该位由软件置位和清零。 0：禁用外设递增模式 1：使能外设递增模式
5	CIRC	循环模式 该位由软件置位和清零。 0：禁用循环模式 1：使能循环模式
4	DIR	数据传输方向 该位由软件置位和清零。 0：从外设读出 1：从存储器读出
3	TEIE	传输错误中断使能 该位由软件置位和清零。 0：禁用 TE 中断 1：使能 TE 中断
2	HTIE	半传输中断使能 该位由软件置位和清零。 0：禁用 HT 中断 1：使能 HT 中断
1	TCIE	传输完成中断使能 该位由软件置位和清零。 0：禁用 TC 中断 1：使能 TC 中断
0	EN	通道使能 该位由软件置位和清零。 0：禁用通道 1：使能通道

11.4.4. DMA_CNDTRx（DMA 通道 x 数据数量寄存器）（x=1..5）

地址偏移：0x0C + 0d20 × (通道编号 - 1)

复位值：0x0000 0000

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

7:0	NDT[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:12	NA	保留位，未定义
11:0	NDT[11:0]	要传输的数据数量 要传输的数据数量从 0 到 4095。此寄存器只能在通道被禁用的时候写入。一旦通道被使能，这个寄存器就是只读的，表示剩余的待传输字节数量。这个寄存器在每次 DMA 传输后递减。 一旦传输完成，该寄存器保持为零，若通道配置为循环模式则自动重载先前编程的值。 如果该寄存器位零，则无论通道是否使能，都不能服务任何事务。

11.4.5. DMA_CPARx（DMA 通道 x 外设地址寄存器）（x=1..5）

地址偏移：0x10 + 0d20 × (通道编号 - 1)
复位值：0x0000 0000
在通道使能的时候不能写该寄存器。

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

Bit	Name	Function
31:0	PA[31:0]	外设地址 外设寄存器的基址，数据将从该寄存器读写。 当 PSIZE 为 01（16 位）时，PA[0]位将被忽略。访问自动对齐到半字地址。 当 PSIZE 为 10（32 位）时，PA[1:0]位将被忽略。访问自动对齐到字地址。

11.4.6. DMA_CMARx（DMA 通道 x 存储器地址寄存器）（x=1..5）

地址偏移：0x14 + 0d20 × (通道编号 - 1)

复位值：0x0000 0000

在通道使能的时候不能写该寄存器。

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

Bit	Name	Function
31:0	MA[31:0]	存储器地址 存储器区域的基址，数据将从该寄存器读写。 当 MSIZE 为 01（16 位）时，PA[0]位将被忽略。访问自动对齐到半字地址。 当 MSIZE 为 10（32 位）时，PA[1:0]位将被忽略。访问自动对齐到字地址。

12. 中断与事件

12.1. 内嵌向量中断控制器（NVIC）

12.1.1. 主要特征

- 32 个可屏蔽中断通道（不包括 16 个 Cortex-M0 的中断线）
- 4 个可编程的优先级（使用了 2 位的中断优先级）
- 低延时的异常与中断处理
- 电源管理控制
- 系统控制寄存器的实现

12.1.2. 中断和异常向量

表 12.1 向量表

位置	优先级	优先级类型	名称	说明	地址
—	—	—	保留	—	0x00000000
	-3	固定	Reset	复位	0x00000004
	-2	固定	NMI	不可屏蔽中断。时钟安全系统(CSS)连接到 NMI 向量	0x00000008
	-1	固定	HardFault	硬件错误	0x0000000C
	3	可设置	SVCall	通用 SWI 指令调用的系统服务	0x0000002C
	5	可设置	PendSV	可挂起的系统服务	0x00000038
	6	可设置	SysTick	系统滴答定时器	0x0000003C
0	7	可设置	WWDG	窗口看门狗中断	0x00000040
1	8	可设置	PVD	连接到 EXTI 线的可编程电压检测中断	0x00000044
2	9	可设置	RTC	RTC 全局中断	0x00000048
3	10	可设置	Flash	Flash 全局中断	0x0000004C
4	11	可设置	RCC	RCC 全局中断	0x00000050
5	12	可设置	EXTI0_1	EXTI 线[1:0]中断	0x00000054
6	13	可设置	EXTI2_3	EXTI 线[3:2]中断	0x00000058
7	14	可设置	EXTI4_15	EXTI 线[15:4]中断	0x0000005C
8	—	—	保留	—	0x00000060
9	16	可设置	DMA_CH1	DMA 通道 1 中断	0x00000064
10	17	可设置	DMA_CH2_3	DMA 通道 2 和 3 中断	0x00000068
11	18	可设置	DMA_CH4_5	DMA 通道 4 和 5 中断	0x0000006C
12	19	可设置	ADC_COMP	ADC 和比较器中断	0x00000070

13	20	可设置	TIM1_BRK_UP_TRG_COM	TIM1 刹车、更新、触发和通信中断	0x00000074
14	21	可设置	TIM1_CC	TIM1 捕获比较中断	0x00000078
15	—	—	保留	—	0x0000007C
16	23	可设置	TIM3	TIM3 全局中断	0x00000080
17	24	可设置	TIM6	TIM6 全局中断	0x00000084
18	—	—	保留	—	0x00000088
19	27	可设置	TIM14	TIM14 全局中断	0x0000008C
20	28	可设置	TIM15	TIM15 全局中断	0x00000090
21	29	可设置	TIM16	TIM16 全局中断	0x00000094
22	30	可设置	TIM17	TIM17 全局中断	0x00000098
23	31	可设置	I2C1	I2C1 全局中断	0x0000009C
24	32	可设置	I2C2	I2C2 全局中断	0x000000A0
25	33	可设置	SPI1	SPI1 全局中断	0x000000A4
26	34	可设置	SPI2	SPI2 全局中断	0x000000A8
27	35	可设置	USART1	USART1 全局中断	0x000000AC
28	36	可设置	USART2	USART2 全局中断	0x000000B0
29	—	—	保留	—	0x000000B4
30	—	—	保留	—	0x000000B8
31	38	可设置	USB	USB 全局中断和 EXTI 18 线	0x000000BC

12.2. 外部中断和事件控制器（EXTI）

外部中断和事件控制器管理外部和内部异步事件和中断 ,并生成相应的事件请求到核内部中断控制器和电源管理的唤醒请求。

外部中断和事件控制器管理多达 28 个外部和内部事件线。

每个外部中断线可以独立选择触发沿，而内部线只能为上升沿触发。如果发生了中断触发，该终端可以被一直挂起，如果发生了事件，则该事件是一个脉冲，送往核内作为唤醒信号使用。内部中断标志位在其各自模块有标志位，在 EXTI 中无中断标志位。每条输入线可单独屏蔽其产生中断或者事件。另外，内部线只能在 Stop 模式下有效。该控制器允许写特定寄存器位触发相应的事件或者中断。

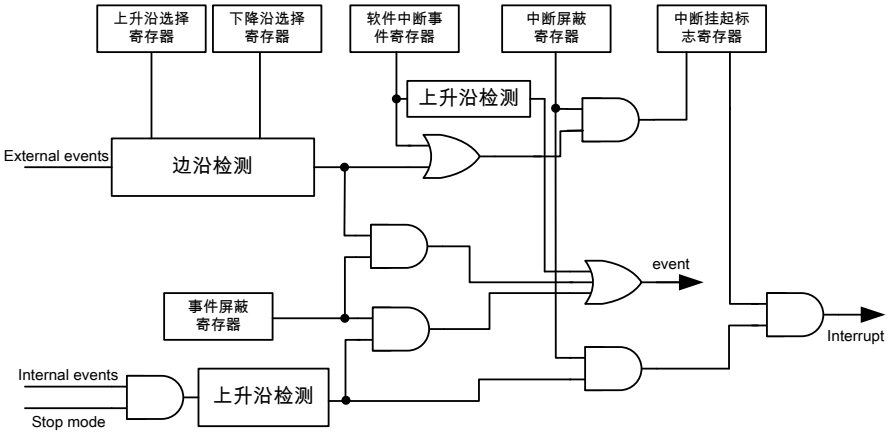


图 12.1 外部中断和事件控制器框图

12.2.1. 主要特征

外部中断和时间控制器主要特性如下：

- 支持多达 28 个中断或者事件请求
- 每个中断或者事件线都可单独屏蔽
- 在非 Stop 模式下，自动禁止内部线中断或者事件
- 独立的触发外部事件或者中断
- 每个外部中断线都有专用的状态位
- 软件触发外部中断或者事件

12.2.2. 事件管理

FT32F0XX 可以处理外部和内部事件来唤醒内核（WFE）。唤醒事件可由以下配置产生：

1 在外设控制器中使能一个外部中断，在 NVIC 中使能该中断或者 SEVONPEND 置 1。当 MCU 从 WFE 唤醒后，需要清除 EXTI 中相应的挂起标志位。

2 配置一个外部或者内部 EXTI 线为事件模式。当 MCU 从 WFE 唤醒后，因为对应事件线的挂起标志位不会被置起，因此不必清除相应中断挂起标志位或者 NVIC 中断挂起位。

12.2.3. 功能描述

对于外部中断线，要产生中断，必须先配置好触发方式并使能该中断线。根据具体应用的需要，可以通过配置 EXTI_RTSR 和 EXTI_FTSR 寄存器，选择上升沿触发、下降沿触发和双沿触发三种方式。置位 EXTI_IMR 寄存器位使能对应中断线产生中断功能。

对于内部中断线，触发方式只能为上升沿，不能进行配置，在 EXTI_IMR 中内部线中断默认使能，无对应挂起标志位。

对于外部事件线，要产生事件，必须先配置好触发方式并使能该事件线。根据具体应用的需要，可以通过配置 EXTI_RTSR 和 EXTI_FTSR 寄存器，选择上升沿触发、下降沿触发和双沿触发三种方式。置位 EXTI_EMR 寄

存器位使能对应事件线产生事件功能。

对于外部线，在使能了中断或者事件情况下，软件置位 EXTI_SWIER 寄存器位，可以产生中断或者事件。

硬件中断选择：
根据下列过程配置 EXTI 线为中断源：
1. 在 EXTI_IMR 寄存器中配置所选中断线的屏蔽位
2. 在 EXTI_RTSR 和 EXTI_FTSR 寄存器中配置所选中断线的触发方式
3. 配置对应的 NVIC 中断通道的使能位，使得中断线中的请求可以被正确地响应

硬件事件选择：
根据下列过程可配置 EXTI 线位事件源：
1. 在 EXTI_EMR 寄存器中配置所选事件的屏蔽位
2. 在 EXTI_RTSR 和 EXTI_FTSR 寄存器中配置所选事件线的触发方式

软件中断或者事件的选择：
通过以下过程可配置外部中断或者事件可以被配置为软件中断或事件线：
1. 在 EXTI_IMR 和 EXTI_EMR 配置相应的屏蔽位，选择事件或者中断模式
2. 在 EXTI_SWIER 寄存器中设置相应的请求位

12.2.4. 外部和内部中断/事件线映射

GPIO 口连接到 16 个外部中断或事件线如下图所示：

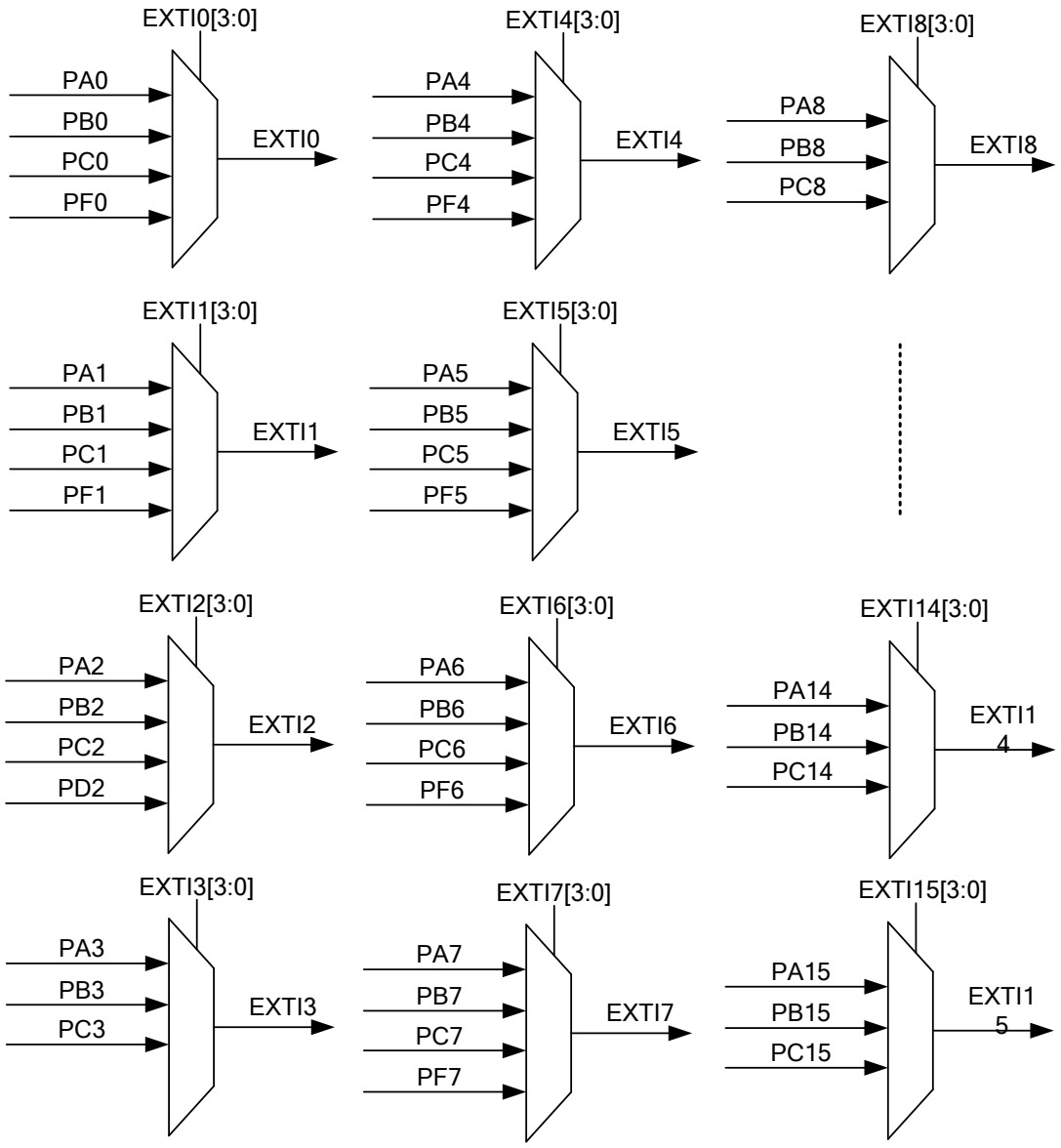


图 12.2 GPIO 到 EXTI 线映射图

上图中 EXTI1[3:0]~EXTI15[3:0]在 SYSCFG_EXTICRx 寄存器中，x 代表 1~4。

其余线连接如下：

- EXTI 16 线连接可编程电压检测输出（PVD）
- EXTI 17 线连接到 RTC 警铃事件
- EXTI 18 线连接到内部 USB 唤醒事件
- EXTI 19 线连接到 RTC 入侵和时间戳事件
- EXTI 20 线保留（内部保持为低电平）
- EXTI 21 线连接到比较器 1 的输出
- EXTI 22 线连接到比较器 2 的输出
- EXTI 23 线保留（内部保持为低电平）
- EXTI 24 线保留（内部保持为低电平）
- EXTI 25 线保留（内部保持为低电平）
- EXTI 26 线保留（内部保持为低电平）
- EXTI 27 线保留（内部保持为低电平）

- EXTI 28 线保留（内部保持为低电平）
- EXTI 29 线保留（内部保持为低电平）
- EXTI 30 线保留（内部保持为低电平）
- EXTI 31 线保留（内部保持为低电平）

12.3. 外部中断和事件控制器寄存器映射

offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	EXTI_IMR	IMR[31:0]																															
	Reset	0	1	1	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	EXTI_EMR	EMR[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	EXTI_RTSR	-	-	-	-	-	-	-	-	-	RTSR[22]	RTSR[21]	-	RTSR[19]	-	RTSR[17:0]																	
	Reset	x	x	x	x	x	x	x	x	x	0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	EXTI_FTSR	-	-	-	-	-	-	-	-	-	FTSR[22]	FTSR[21]	-	FTSR[19]	-	FTSR[17:0]																	
	Reset	x	x	x	x	x	x	x	x	x	0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	EXTI_SWIER	-	-	-	-	-	-	-	-	-	SWIER[22]	SWIER[21]	-	SWIER[19]	-	SWIER[17:0]																	
	Reset	x	x	x	x	x	x	x	x	x	0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	EXTI_PR	-	-	-	-	-	-	-	-	-	PR[22]	PR[21]	-	PR[19]	-	PR[17:0]																	
	Reset	x	x	x	x	x	x	x	x	x	0	0	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

12.3.1. EXTI_IMR

偏移地址：0x00

复位值：0x7F84 0000

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

Bit	Name	Function
31:0	IMR[31:0]	外部或内部线 x 中断屏蔽位，x 表示 31~0

		0：屏蔽来自线 x 上的中断请求 1：使能来自线 x 上的中断请求

12.3.2. EXTI_EMR

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	Function
31:0	EMR[31:0]	外部或内部线 x 事件屏蔽位，x 表示 31~0 0：屏蔽来自线 x 上的事件请求 1：使能来自线 x 上的事件请求

12.3.3. EXTI_RTSR

偏移地址：0x08

复位值：0x0000 0000

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

Bit	Name	Function
31:23	NA	保留位

22	RTSR[22]	外部线 22 上升沿触发使能位 0：屏蔽来自线 22 上的上升沿触发中断或事件 1：使能来自线 22 上的上升沿触发中断或事件
21	RTSR[21]	外部线 21 上升沿触发使能位 0：屏蔽来自线 21 上的上升沿触发中断或事件 1：使能来自线 21 上的上升沿触发中断或事件
20	NA	保留位
19	RTSR[19]	外部线 19 上升沿触发使能位 0：屏蔽来自线 19 上的上升沿触发中断或事件 1：使能来自线 19 上的上升沿触发中断或事件
18	NA	保留位
17:0	RTSR[17:0]	外部线 x 上升沿触发使能位，x 表示 17~0 0：屏蔽来自线 x 上的上升沿触发中断或事件 1：使能来自线 x 上的上升沿触发中断或事件

12.3.4. EXTI_FTSR

偏移地址：0x0C

复位值：0x0000 0000

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

Bit	Name	Function
31:23	NA	保留位
22	FTSR[22]	外部线 22 下降沿触发使能位 0：屏蔽来自线 22 上的下降沿触发中断或事件 1：使能来自线 22 上的下降沿触发中断或事件
21	FTSR[21]	外部线 21 下降沿触发使能位 0：屏蔽来自线 21 上的下降沿触发中断或事件 1：使能来自线 21 上的下降沿触发中断或事件
20	NA	保留位
19	FTSR[19]	外部线 19 下降沿触发使能位 0：屏蔽来自线 19 上的下降沿触发中断或事件 1：使能来自线 19 上的下降沿触发中断或事件
18	NA	保留位

17:0	FTSR[17:0]	外部线 x 下降沿触发使能位，x 表示 17~0 0：屏蔽来自线 x 上的下降沿触发中断或事件 1：使能来自线 x 上的下降沿触发中断或事件

12.3.5. EXTI_SWIER

偏移地址：0x10

复位值：0x0000 0000

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

Bit	Name	Function
31:23	NA	保留位
22	SWIER[22]	线 22 的软件中断或事件位 当该位为 0 时，该位写 1，如果对应的 EXTI_IMR 位置高，则 EXTI_PR 寄存器中对应位被置高，产生一个中断，如果对应的 EXTI_EMR 位置高，则产生一个事件，不置高 EXTI_PR 对应位。 通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为 0
21	SWIER[21]	线 21 的软件中断或事件位 当该位为 0 时，该位写 1，如果对应的 EXTI_IMR 位置高，则 EXTI_PR 寄存器中对应位被置高，产生一个中断，如果对应的 EXTI_EMR 位置高，则产生一个事件，不置高 EXTI_PR 对应位。 通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为 0
20	NA	保留位
19	SWIER[19]	线 19 的软件中断或事件位 当该位为 0 时，该位写 1，如果对应的 EXTI_IMR 位置高，则 EXTI_PR 寄存器中对应位被置高，产生一个中断，如果对应的 EXTI_EMR 位置高，则产生一个事件，不置高 EXTI_PR 对应位。 通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为 0
18	NA	保留位
17:0	SWIER[17:0]	线 x 的软件中断或事件位，x 代表 17~0 当该位为 0 时，该位写 1，如果对应的 EXTI_IMR 位置高，则 EXTI_PR 寄存器中对应位被置高，产生一个中断，如果对应的 EXTI_EMR 位置高，则产生

		一个事件，不置高 EXTI_PR 对应位。 通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为 0

12.3.6. EXTI_PR

偏移地址：0x14

复位值：0x0000 0000

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

Bit	Name	Function
31:23	NA	保留位
22	PR[22]	线 22 的挂起标志位 0：没有发生触发请求 1：发生了选择的触发请求 当外部中断线上发生了选择的边沿事件，该位被置 1。该位写 1 清零。
21	PR [21]	线 21 的挂起标志位 0：没有发生触发请求 1：发生了选择的触发请求 当外部中断线上发生了选择的边沿事件，该位被置 1。该位写 1 清零。
20	NA	保留位
19	PR [19]	线 19 的挂起标志位 0：没有发生触发请求 1：发生了选择的触发请求 当外部中断线上发生了选择的边沿事件，该位被置 1。该位写 1 清零。
18	NA	保留位
17:0	PR [17:0]	线 x 的挂起标志位（x=17 到 0） 0：没有发生触发请求 1：发生了选择的触发请求 当外部中断线上发生了选择的边沿事件，该位被置 1。该位写 1 清零。

13. 模数转换器（ADC）

13.1. 介绍

12 位 ADC 是一种逐次逼近型的模数转换器。多达 20 个多路复用通道，可测量 16 个外部源信号和 4 个内部源信号。各通道的 A/D 转换能以单次、连续、扫描或断续模式执行。ADC 结果存储在一个左对齐或右对齐的 16 位数据寄存器。另外还提供内部参考电压进行选择。

13.2. ADC 主要特性

- 高性能
  - 12 位、10 位、8 位或 6 位可配置分辨率
  - ADC 转换时间：1.0 μs @ 12 位分辨率（1 MHz），0.93 μs 转换时间@10 位分辨率，更快的转换时间能通过更低分辨率获得。
  - 自校准
  - 可编程采样时间
  - 带内嵌数据一致性的数据对齐
  - DMA 支持
- 低功耗
  - 应用为低功耗操作而降低 PCLK 频率，同时仍能保持最佳 ADC 性能。例如，无论 PCLK 频率怎样，都能保持 1.0 μs 转换时间。
  - 等待模式：在低频率 PCLK 的应用下，防止 ADC 溢出。
  - 自动关闭模式：除了在有效转换期间，ADC 会自动关闭。这显著地减少 ADC 的功耗。
- 模拟输入通道
  - 16 个外部模拟输入
  - 1 个内部温度传感器（V_{SENSE}）
  - 1 个内部基准电压（V_{REFINT}）
  - 1 个 IO 采样保持电路（V_{IOSH}）
  - 1 个运算放大电路（V_{OP}）
- 启动转换开始：
  - 由软件
  - 由带可配置极性的硬件触发（来自 TIM1、TIM3、和 TIM15 的内部定时器事件）
- 转换模式
  - 可转换单个通道，或扫描一序列的通道。
  - 单次模式在每次触发时转换选定输入
  - 连续模式连续地转换选定输入
  - 断续模式
- 中断生成在采样结束、转换结束、序列转换结束、模拟看门狗事件和溢出事件。
- 模拟看门狗
- 可配置的 ADC 参考电压

  - VDDA 作为 ADC 参考电压
  - ADC 专用的内部参考电压源，可选 2.5V
- ADC 供电需求：2.4 V 到 5.5 V
- ADC 输入范围：$V_{SSA} \le V_{IN} \le V_{REF+}$

13.3. ADC 引脚和内部信号

表 13.1. ADC 内部信号表

内部信号名	信号类型	描述
TRG_X	输入	ADC 转换触发
V_{SENSE}	输入	内部温度传感器输出电压
V_{REFINT}	输入	内部电压基准电路输出电压
V_{IOSH}	输入	IO 采样保持电路输出电压
V_{OP}	输入	运算放大电路输出电压

表 13.2. ADC 引脚表

名字	信号类型	注释
V_{DDA}	输入，模拟电源供电	模拟供电电源和 ADC 正极参考电压，$V_{DDA} \ge V_{DD}$
V_{SSA}	输入，模拟电源地	模拟供电电源的地。必须处于 VSS 电势
ADC_IN[15:0]	模拟输入信号	16 个模拟输入通道

13.4. ADC 功能描述

图 13.1 展示 ADC 框图，表 13.2 给出 ADC 引脚描述。

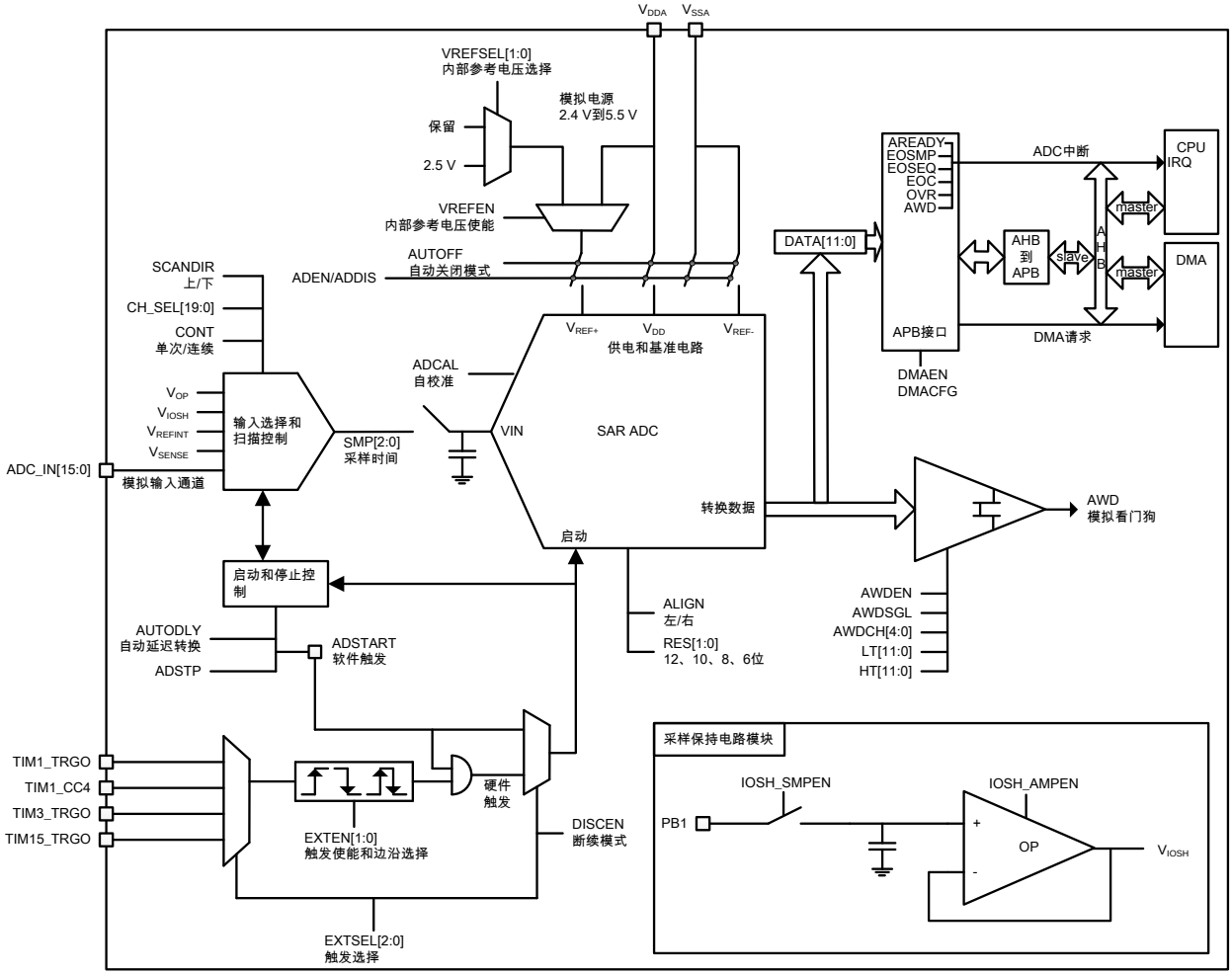


图 13.1. ADC 框图

13.4.1. 校准（ADCAL）

此 ADC 具备校准功能。在校准过程中，ADC 计算出一个校准系数，应用于 ADC 内部直到下次 ADC 关闭。应用不能在校准中使用 ADC，必须等到校准完成。

校准应该在 A/D 转换前执行。移除由于工艺变化导致芯片差异的偏移误差。

通过软件设置 ADCAL=1 来启动校准。校准只能在 ADC 禁用（当 ADEN=0）时启动。ADCAL 位在所有校准序列中保持为 1。当校准完成的同时，该位会被硬件清零。在这之后，校准系数可以从 ADC_DR 寄存器（从 6 到 0 位）中读出。

当 ADC 被禁用（ADEN=0）时，内部模拟校准的状态仍旧保持。当 ADC 操作环境改变时（对于 ADC 偏移变化来说，V_{DDA} 变化是主要贡献者，而温度变化只是很小程度的影响），推荐重跑校准周期。

校准系数在每次 ADC 关闭时丢失（例如进入 Standby 模式）。

校准软件过程：

1. 确保 ADEN=0 和 DMAEN=0
2. 设置 ADCAL=1
3. 等待 ADCAL=0
4. 校准系数可以从 ADC_DR 的 6:0 位读取到。

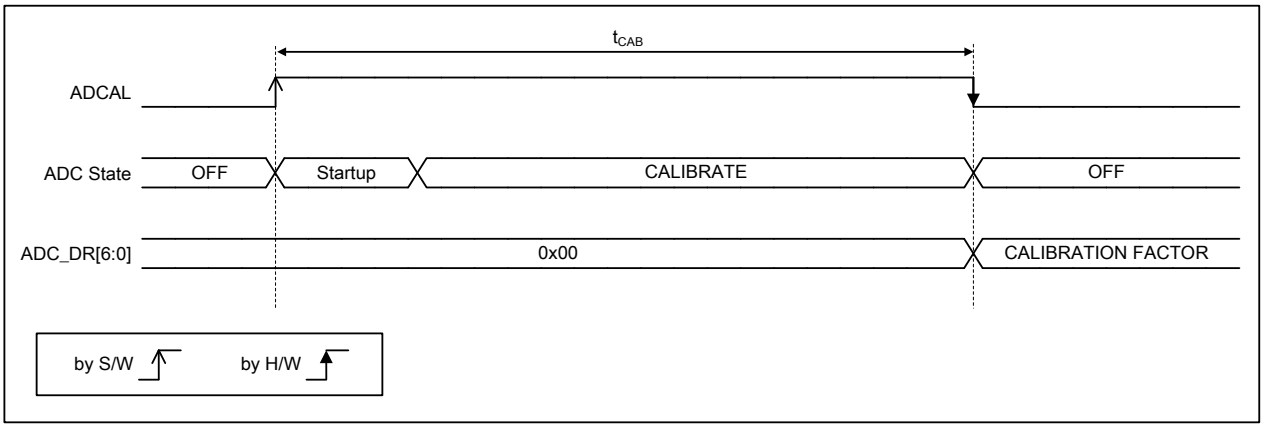


图 13.2. ADC 校准

13.4.2. ADC 开关控制（ADEN，ADDIS，ADRDY）

在 MCU 上电后，ADC 是被禁用的（ADEN=0），并且处于关闭模式。
如图 13.3 所示，在 ADC 开始精确转换之前需要一个稳定时间 t_{STAB}。
两个控制位被用于使能或禁用 ADC：

- 设置 ADEN=1 来使能 ADC。ADRDY 标志会在 ADC 准备好可操作时被置位。
- 设置 ADDIS=1 来禁用 ADC，并且将 ADC 置于关闭模式。ADEN 和 ADDIS 位会在 ADC 彻底被禁用时自动被硬件清零。

转换可以通过设置 ADSTART=1 来启动（参考章节 13.5 通过外部触发和触发极性的转换（EXTSEL，EXTEN）），若触发器被使能时，也可以通过一个外部触发事件的发生来启动。

遵循下述过程来使能 ADC：

1. 通过写 1 来清零 ADC_ISR 寄存器中的 ADRDY 位。
2. 设置 ADC_CR 寄存器中的 ADEN 位为 1。
3. 等待 ADC_ISR 寄存器中的 ADRDY 位为 1，并且继续写 ADEN 位为 1（ADRDY 位在 ADC 启动时间后被置位）。如果通过设置 ADC_IER 寄存器中的 ADRDYIE 位来使能了中断，那这个等待准备动作可以通过中断来处理。

遵循下述过程来禁用 ADC：

1. 检查 ADC_CR 寄存器中的 ADSTART 位是否为 0，确保没有转换正在进行。如果有需要，通过写 ADC_CR 寄存器中的 ADSTP 位为 1 来停止任何正在进行的转换，并且等待该位读为 0。
2. 设置 ADC_CR 寄存器中的 ADDIS 位为 1。
3. 如果应用有需要的话，等待 ADC_CR 寄存器中的 ADEN 位为 0，指示 ADC 已经完全被禁用了（ADDIS 位会在 ADEN=0 的同时自动复位）
4. 通过写 1 来清零 ADC_IST 寄存器中的 ADRDY 位（可选）。

注意：在 ADCAL=1 和 ADCAL 位被硬件清零后的四个 ADC 时钟周期内（校准结束期间），ADEN 位都不能被

置位。

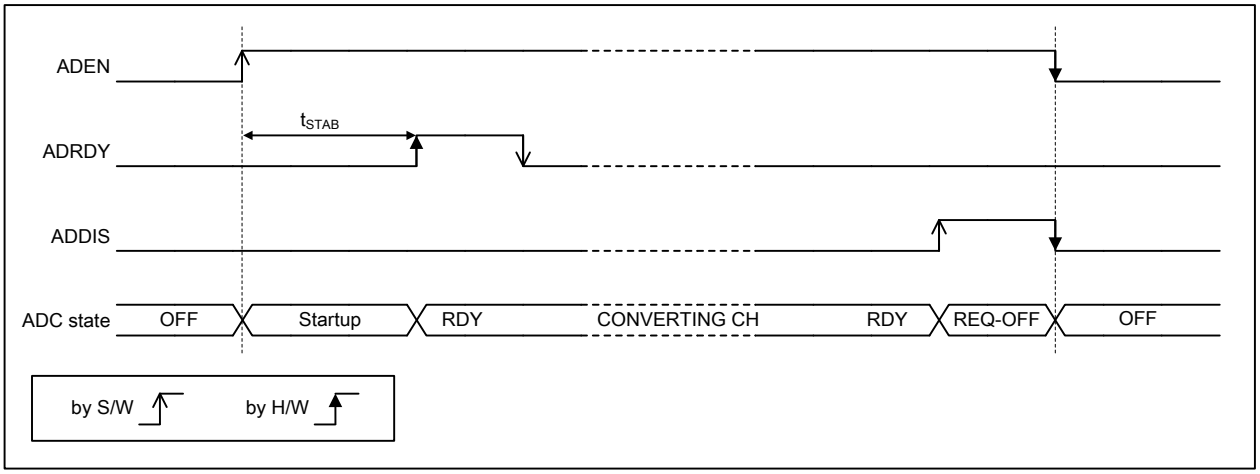


图 13.3. 使能或禁用 ADC

注：在自动关闭模式下（AUTOFF=1），电源开/关阶段是通过硬件自动执行的，并且不会置位 ADRDY 标志。

13.4.3. ADC 时钟（CKMODE）

ADC 具有双时钟域架构，能使得 ADC 通过独立于 APB 时钟（PCLK）的时钟（ADC 异步时钟）来驱动。

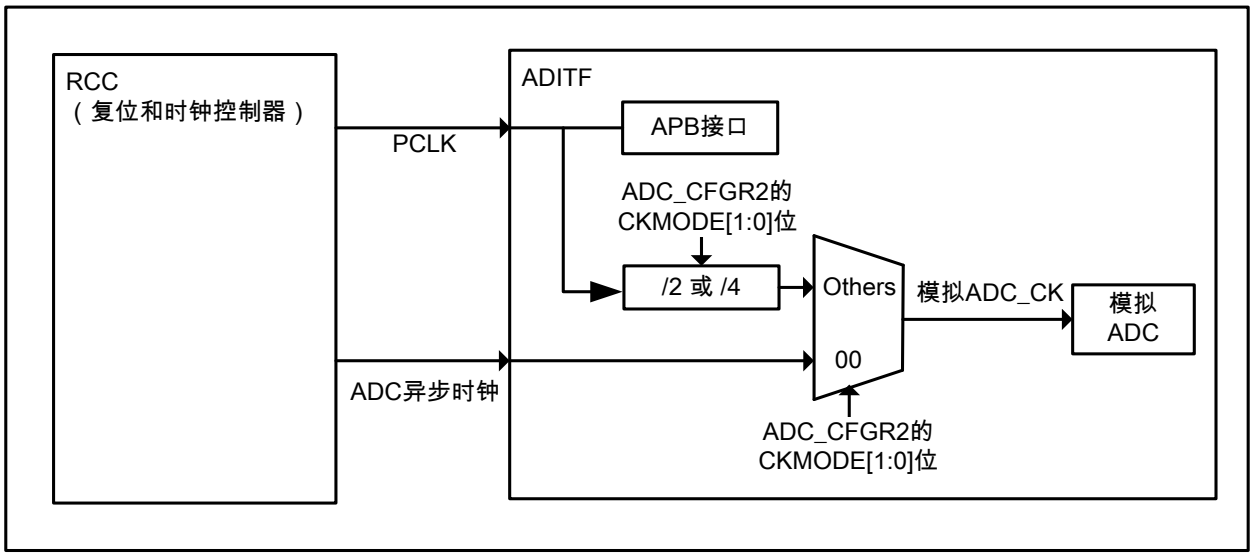


图 13.4. ADC 时钟方案

1. 参考章节 7：复位和时钟控制器（RCC），查看如何使能 PCLK 和 ADC 异步时钟。

模拟 ADC 的输入时钟能在两个不同时钟源中进行选择（看图 13.4：ADC 时钟方案，查看如何使能 PCLK 和 ADC 异步时钟）：

a) ADC 时钟可以是特定时钟源，命名为“ADC 异步时钟”，独立且异步于 APB 时钟。
参考 RCC 章节获取更多生成该时钟源的信息。
要选择该方案，ADC_CFGR2 寄存器的 CKMODE[1:0]位必须是复位的。
b) ADC 时钟能从 ADC 总线接口的 APB 时钟上衍生，依据 CKMODE[1:0]位通过可编程系数(2 或 4)分频。
要选择该方案，ADC_CFGR2 寄存器的 CKMODE[1:0]位必须是不同于“00”。
选项 a) 无论 APB 时钟方案如何选择，ADC 时钟频率都能达到最大。
选项 b) 仍需要时钟域再同步。两个时钟域之间的再同步会增加触发实现的不确定性。

表 13.3. 触发和转换开始之间的延迟

ADC 时钟源	CKMODE[1:0]	触发事件和转换开始之间的延迟
专用 14MHz 时钟	00	延迟不是确定的（抖动，2~3 * T_{ADC}）
PCLK 的 2 分频	01	
PCLK 的 4 分频	10	

13.4.4. 配置 ADC

在 ADC 禁用时（ADEN 必须为 0），软件才可写 ADC_CR 寄存器的 ADCAL 和 ADEN 位。
在 ADC 使能并且没有挂起的禁用 ADC 请求时（ADEN=1 和 ADDIS=0），软件才可写 ADC_CR 寄存器的 ADSTART 和 ADDIS 位。
对于在 ADC_IER、ADC_CFGR、ADC_SMPR、ADC_TR、ADC_CHSELR 和 ADC_CCR 寄存器中的所有其它控制位，软件仅当 ADC 使能（ADEN=1）并且没有转换正在进行（ADSTART=0）时才可写这些配置控制位。
在 ADC 使能（可能转换中）并且没有禁用 ADC 的挂起请求（ADSTART=1 和 ADDIS=0）时，软件才可写 ADC_CR 寄存器的 ADSTP 位。
注：没有硬件保护机制去防止软件执行上述规则禁止的写操作。如果这些禁止的写操作发生了，ADC 可能会进入一个未定义状态。要从这种情况下恢复正确操作，ADC 必须被禁用（清零 ADEN 和 ADC_CR 寄存器的所有位）。

13.4.5. 通道选择（CHSEL，SCANDIR）

有高达 20 个多路复用通道：
- 16 个来自 GPIO 引脚的模拟输入（ADC_IN0...ADC_IN15）
- 4 个内部模拟输入（温度传感器，内部基准电路，IO 采样保持电路，运算放大电路）
- 其中 IO 采样保持电路采样输入和 ADC_IN9 使用同一个 GPIO 引脚 PB1

可以转换单个通道或者自动扫描一序列通道。
被转换的通道序列必须编程在 ADC_CHSELR 通道选择寄存器：每一个模拟输入通道有其专用的选择位（CHSEL0...CHSEL19）。
被扫描的通道顺序能通过编程 ADC_CFGR1 寄存器的 SCANDIR 位来配置：
- SCANDIR=0：向前扫描，从通道 0 到通道 19
- SCANDIR=1：向后扫描，从通道 19 到通道 0

温度传感器、V_{REFINT}、V_{SH} 和 V_{OP} 内部通道

温度传感器连接到 ADC_IN16。内部基准电路 V_{REFINT} 连接到 ADC_IN17。IO 采样保持电路 V_{SH} 连接到 ADC_IN18。运算放大电路 V_{OP} 连接到 ADC_IN19。

13.4.6. 可编程采样时间（SMP）

在开始一个转换之前，ADC 需要在被测量的电压源和 ADC 内嵌采样电容之间建立一个直接连接。这个采样时间必须足够以使得输入电压源对采样和保持电容充电到输入电压电平。
依据输入电压源的输入电阻，有一个可编程的采样时间能够允许调整转换速度。
ADC 采样输入电压的 ADC 时钟周期数目能使用 ADC_SMPR 寄存器的 SMP[2:0]位进行修改。
可编程的采样时间对所有通道是通用的。如果应用有需要，软件能在每次转换之间更改和适应采样时间。
总的转换时间计算如下：

t_{CONV} = 采样时间 + 12.5 × ADC 时钟周期

例如：

ADC_CLK = 14MHz 和 1.5 个 ADC 时钟周期的采样时间：

t_{CONV} = 1.5 + 12.5 = 14 个 ADC 时钟周期 = 1 μs

ADC 通过置位 EOSMP 标志来指示采样阶段的结束。

13.4.7. 单次转换模式（CONT=0）

在单次转换模式中，ADC 执行一次转换序列，转换一次所有通道。当 ADC_CFGR1 寄存器的 CONT 位为 0 时，该模式被选中。转换通过下述方式开始：

- 置位 ADC_CR 寄存器的 ADSTART 位
- 硬件触发事件

在序列中，每次通道转换完成之后：

- 转换后的数据存储在 16 位的 ADC_DR 寄存器中
- EOC（end of conversion）标志被置位
- 若置位了 EOCIE 位则产生一个中断

在转换序列完成后：

- EOSEQ（end of sequence）标志被置位
- 若置位了 EOSEQIE 位则产生一个中断

直到新的外部触发事件发生或者 ADSTART 位被再次置位，ADC 都会停止。
注：要转换单个通道，编程一个长度为 1 的序列。

13.4.8. 连续转换模式（CONT=1）

在连续转换模式中，当一个软件或硬件触发事件发生时，ADC 执行一转换序列，转换一次所有通道，然后自动地重新开始和继续执行相同的转换序列。当 ADC_CFGR1 寄存器的 CONT 位为 1 时，该模式被选中。转换通过下述方式开始：

- 置位 ADC_CR 寄存器的 ADSTART 位
- 硬件触发事件

在序列中，每次通道转换完成之后：

- 转换后的数据存储在 16 位的 ADC_DR 寄存器中
- EOC（end of conversion）标志被置位
- 若置位了 EOCIE 位则产生一个中断

在转换序列完成后：

- EOSEQ（end of sequence）标志被置位
- 若置位了 EOSEQIE 位则产生一个中断

然后，一个新的序列马上重新开始，并且 ADC 继续重复转换序列。

注：要转换单个通道，编程一个长度为 1 的序列。
不能同时使能断续模式和连续模式：禁止同时设置 DISCEN=1 和 CONT=1。

13.4.9. 启动转换（ADSTART）

软件通过设置 ADSTART=1 来启动 ADC 转换。
当 ADSTART 被设置，则转换：

- 若 EXTEN=00 则马上启动（软件触发）
- 若 EXTEN≠00 则在选中的硬件触发的下次有效边沿启动。

ADSTART 位也被用于指示 ADC 操作目前是否正在进行。当 ADSTART=0，指示 ADC 处于空闲状态时，可以重新配置 ADC。
ADSTART 位由硬件清零:

- 由软件触发的单次模式（CONT=0，EXTEN=00）
  - 在任一个转换序列结束时（EOSEQ=1）
- 由软件触发的断续模式（CONT=0，DISCEN=1，EXTEN=00）
  - 在转换结束时（ECO=1）
- 在所有情况下（CONT=X，EXTEN=XX）
  - 在软件调用的 ADSTP 过程执行之后（查看章节 13.4.11：停止正在进行的转换（ADSTP））

注：在连续模式中（CONT=1），当 EOSEQ 标志被设置时，ADSTART 位不会被硬件清零，因为转换序列被自动地重新启动。
当在单次模式中选择硬件触发（CONT=0 和 EXTEN=01），ADSTART 位不会在 EOSEQ 标志置位时被硬件清零。这避免了软件需要再次设置 ADSTART，并且确保了下次触发事件不会丢失。

13.4.10. 时序

开始转换和转换结束之间的运行时间是可配置采样时间加上依赖数据分辨率的逐次逼近时间之和：

$t_{ADC} = t_{SMPL} + t_{SAR} = [1.5\ |_{min} + 12.5\ |_{12bit}] \times t_{ADC\_CLK}$

$t_{ADC} = t_{SMPL} + t_{SAR} = [107.1\ ns\ |_{min} + 892.8\ ns\ |_{12bit}] = 1\ \mu s\ |_{min}\ (f_{ADC\_CLK} = 14\ MHz)$

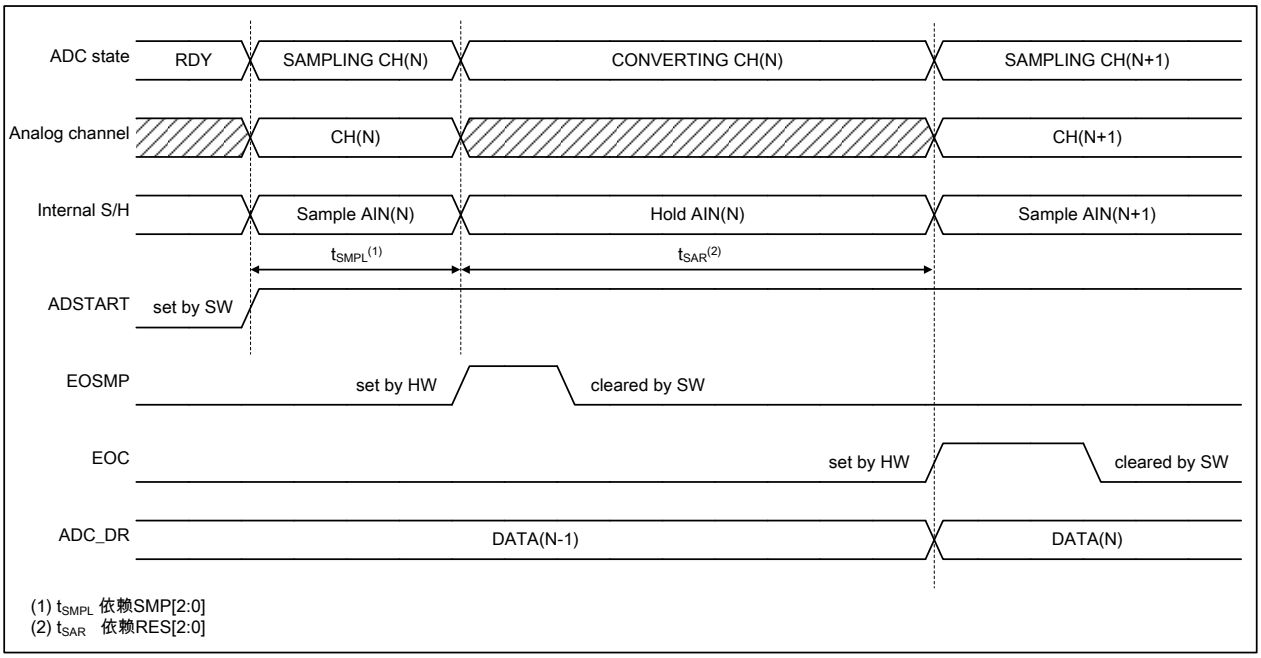


图 13.5. 模数转换时间

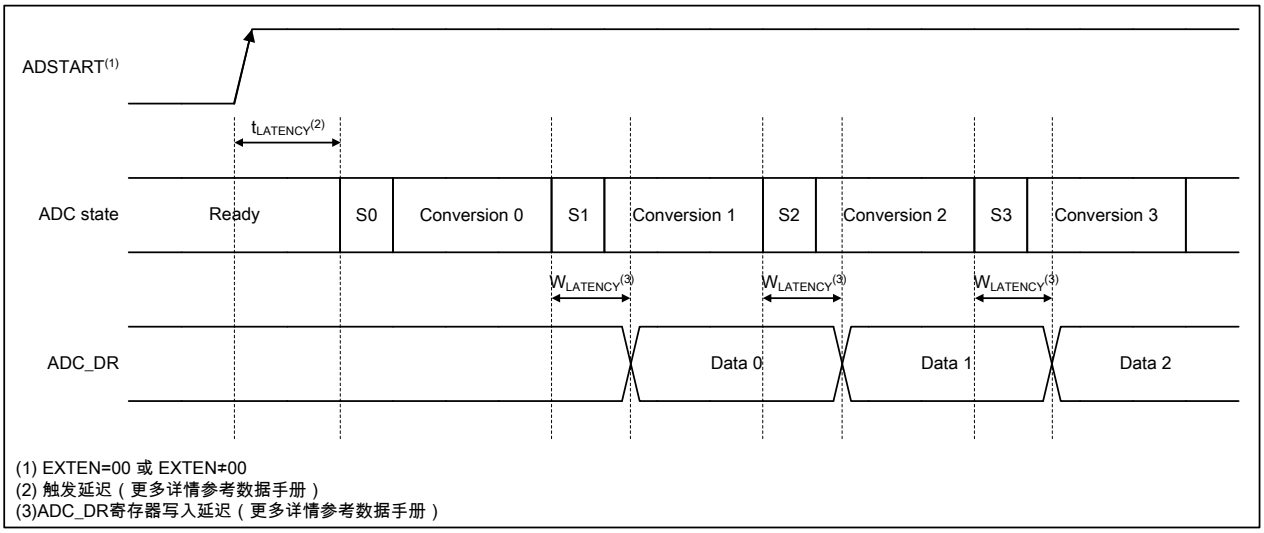


图 13.6. ADC 转换时序

13.4.11. 停止正在进行的转换（ADSTP）

软件可以通过设置 ADC_CR 寄存器的 ADSTP 位为 1 来决定停止正在进行的转换。

这将会复位 ADC 操作，并且 ADC 将会处于空闲状态，等待新的操作。

当 ADSTP 位被软件置位，任何正在进行的转换都会被中止，并且转换结果会被丢弃（ADC_DR 寄存器不会被当前转换更新）。

扫描序列也会被中止和复位（这意味着重新启动 ADC 将会重启一个新的序列）。

一旦这个过程完成，ADSTP 和 ADSTART 位将同时被硬件清零，软件需要等到 ADSTART=0 才可以启动新的转换。

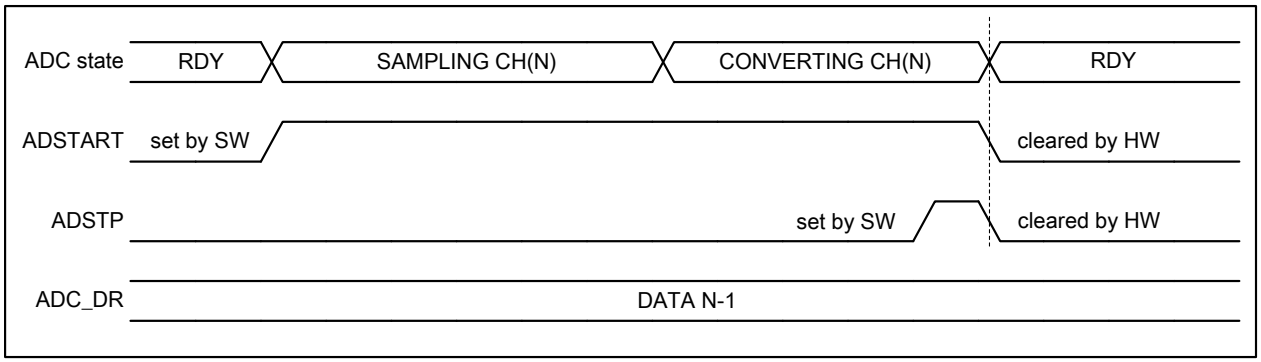


图 13.7. 停止正在进行的转换

13.4.12. IO 采样保持电路（IOSH_SMPEN，IOSH_AMPEN）

IO 采样保持电路主要是为了解决当前电压需要被测量但 ADC 无法立刻启动测量的问题 ,例如分压电路的分压系数测定。

在开始保持之前 ,保持电路需要在被测量的电压源和内嵌采样电容之间建立一个直接连接(IOSH_SMPEN=1，IOSH_AMPEN=1)。这个采样时间必须足够以使得输入电压源对采样和保持电容充电到输入电压电平。

依据输入电压源的输入电阻，软件需要通过设置 ADC_CR2 寄存器的 IOSH_SMPEN 来自行控制采样时间。

当采样完成后设置 IOSH_SMPEN=0 即可停止采样，该时刻的电压直到下次重新采样之前会被保持住一段时间，需要在该段时间内进行转换。

下面内容均为电阻分压电路系数测定示例。

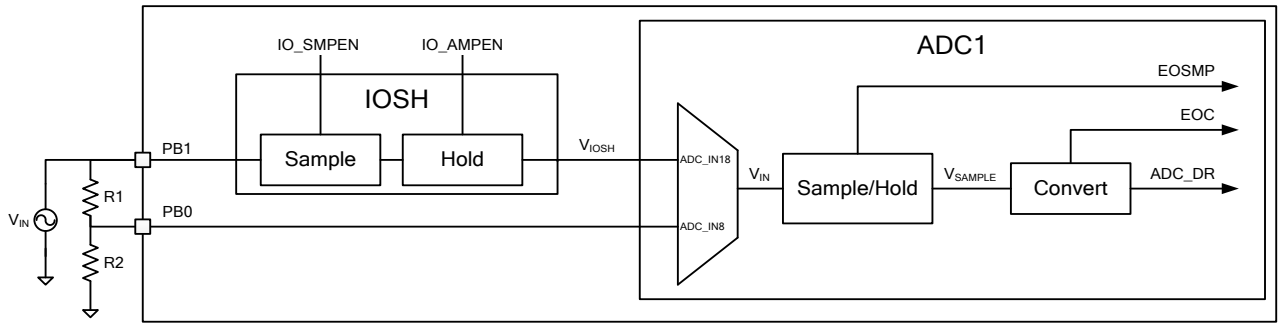
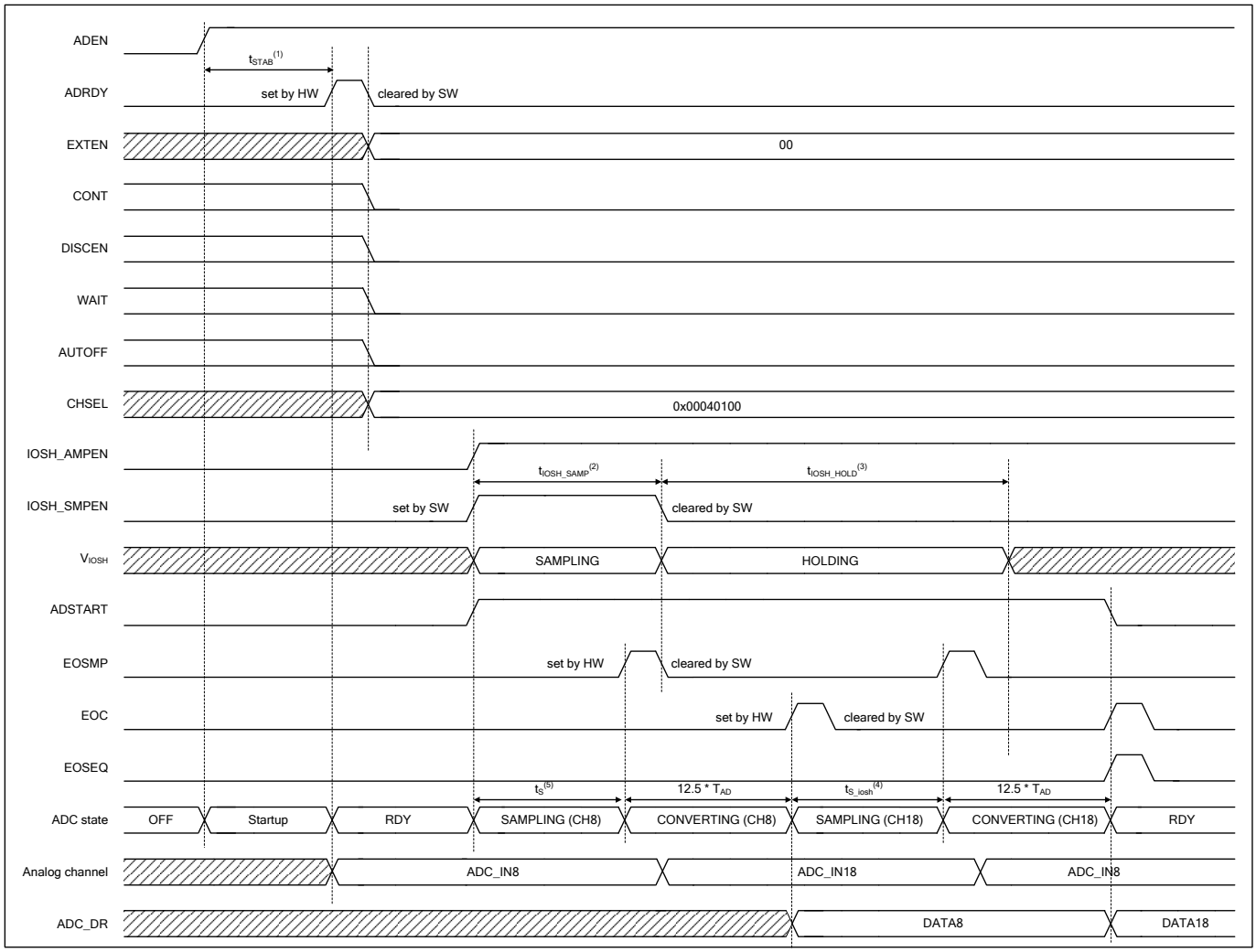


图 13.8. 电阻分压电路图



1. ADC
2. IO
3. IO
4. IO ADC
5. PB0 ADC_VIN8 ADC

图 13.9. IO 采样保持电路时序

IO 采样保持电路使用步骤：

1. 配置 PB1、PB0 口为模拟输入端口。
2. 设置 ADC_CR.ADEN=1 来启动 ADC 模块，并等待 ADC 启动稳定时间(t_{STAB})。
3. 配置 ADC 为软件触发、单次转换模式、无等待无自动关闭、从 CH8 到 CH18 的两个通道扫描顺序，设置 ADC_CFGR1 &= ~(0x0001EC04)、ADC_CHSELR = 0x00040100。
4. ADC 时钟和 ADC 采样时间需依据实际应用需求配置，需满足器件数据手册中定义的三个采样时间：PB0 口的 ADC 通道电压采样时间(t_S)、PB1 口的 IO 电压采样时间(t_{IOSH_SAMP})、读 IO 采样保持电路输出的 ADC 采样时间(t_{S_iosh})。
5. 设置 ADC_CR2 |= 0x00000300 来启动 IO 采样保持电路和 PB1 口的 IO 电压采样。
6. 设置 ADC_CR.ADSTART=1 来启动 ADC 单次序列转换(CH8->CH18)。
7. 等待 ADC_ISR.EOSMP=1，设置 ADC_CR2.IOSH_SMPEN=0 来保持 PB1 口的电压。
8. 等待 ADC_ISR.EOC=1，从 ADC_DR 中读取 PB0(CH8)的 ADC 结果数据。
9. 再次等待 ADC_ISR.EOC=1，从 ADC_DR 中读取 PB1(CH18)的 ADC 结果数据。

注意：需保证 IO 采样保持电路采样关闭(ADC_CR2.IOSH_SMPEN=0)到 ADC_VIN18 采样关闭(ADC_ISR.EOSMP=1)的时间小于 IO 采样保持电路电压保持时间(t_{IOSH_HOLD})。

10. 根据两次读取的 ADC 结果数据进行计算，可得出电阻分压电路的分压系数。
11. 若要更新分压系数，则清零 ADC_ISR.EOSMP 位(从 ADC_DR 中读取数据会自动清零 ADC_ISR.EOC 位)，重复步骤 5 到 10。

注：可由软件轮询 ADC_ISR.EOSMP 或 ADC_ISR.EOC，也可通过使用 ADC 中断实现。

13.5. 转换中的外部触发和触发极性（EXTSEL，EXTEN）

一个转换或者一个转换序列可以由软件也可以由外部事件（例如定时器捕捉）触发。如果 EXTEN[1:0]控制位不等于“2’b00”, 那外部事件能够根据选中极性来触发一次转换。一旦软件设置了 ADSTART 位为 1，触发选择有效。

在转换正在进行时发生的任何硬件触发都会被忽略掉。

如果 ADSTART 位为 0，发生的任何硬件触发都会被忽略掉。

表 13.4 提供 EXTEN[1:0]值和触发极性之间的对应。

表 13.4. 配置触发极性

源	EXTEN[1:0]
禁用触发检测	00
检测上升沿	01
检测下降沿	10
检测上升沿和下降沿	11

注：外部触发极性仅可在 ADC 没有在转换时被改变。

EXTSEL[2:0]控制位用于选择 8 个可触发转换的可能的事件。

表 13.5 给出正常转换的可能的外部触发。

软件源触发事件能够通过设置 ADC_CR 寄存器的 ADSTART 位来产生。

表 13.5. 外部触发

名字	源	EXTSEL[2:0]
TRG0	TIM1_TRGO	000
TRG1	TIM1_CC4	001
TRG2	保留	010
TRG3	TIM3_TRGO	011
TRG4	TIM15_TRGO	100
TRG5	保留	101
TRG6	保留	110
TRG7	保留	111

注：外部触发选择仅可在 ADC 没有在转换时被改变。

13.5.1. 断续模式（DISCEN）

该模式通过设置 ADC_CFGR1 寄存器的 DISCEN 位来使能。
在该模式中（DISCEN=1），需要一个硬件或软件触发事件来启动每一个定义在序列中的转换。反之，如果 DISCEN=0，一个硬件或软件触发事件就能依次启动所有定义在序列中的转换。
例如：

- DISCEN=1，被转换的通道= 0,3,7,10
  - 第 1 次触发：通道 0 被转换并且产生一个 EOC 事件
  - 第 2 次触发：通道 3 被转换并且产生一个 EOC 事件
  - 第 3 次触发：通道 7 被转换并且产生一个 EOC 事件
  - 第 4 次触发：通道 10 被转换并且同时产生一个 EOC 和 EOSEQ 事件
  - 第 5 次触发：通道 0 被转换并且产生一个 EOC 事件
  - 第 6 次触发：通道 3 被转换并且产生一个 EOC 事件
  - ...
- DISCEN=0，被转换的通道= 0,3,7,10
  - 第 1 次触发：完整的序列被转换：通道 0，然后通道 3、7 和 10。每一次转换产生一个 EOC 事件，并且最后一次也同时产生一个 EOSEQ 事件。
  - 后面的任何触发事件将会重启完整的序列。

注：不可能同时使能断续模式和连续模式：禁止同时设置 DISCEN=1 和 CONT=1。

13.5.2. 可编程的分辨率（RES）-快速转换模式

可以通过降低 ADC 分辨率来获取快速转换时间（t_{SAR}）。
分辨率可以通过编程 ADC_CFGR1 寄存器的 RES[1:0]位来配置成 12、10、8 或 6 位的其中一种。对于不需要高数据精度的应用来说，更低的分辨率允许有更快的转换时间。
注：RES[1:0]位仅可在 ADEN 位为复位状态时被改变。
转换结果始终是 12 位宽度，任何未使用的低位会被读成 0。
更低的分辨率减少逐次逼近步骤所需要的转换时间，如表 13.6 所示。

表 13.6. 对应分辨率的 t_{SAR} 时间

RES[1:0]位	t_{SAR} (时钟周期)	t_{SAR}(ns)@ f_{ADC}=14MHz	t_{SMPL}(min) (ADC 时钟周期)	t_{CONV} (ADC 时钟周期) (使用最小的 t_{SMPL})	t_{CONV}@ f_{ADC}=14MHz
12	12.5	893 ns	1.5	14	1000 ns
10	11.5	821 ns	1.5	13	928 ns
8	9.5	678 ns	1.5	11	785 ns
6	7.5	535 ns	1.5	9	643 ns

13.5.3. 转换结束，采样阶段结束（EOC，EOSMP 标志）

ADC 指示每一次转换结束（EOC）事件。

在新的转换数据结果有效存放于 ADC_DR 寄存器的同时，ADC 置位 ADC_ISR 寄存器的 EOC 标志。如果置位了 ADC_IER 寄存器的 EOCIE 位，则还会产生一个中断。EOC 标志可以通过软件写 1 或者读 ADC_DR 寄存器来清零。

ADC 也可以通过置位 ADC_ISR 寄存器的 EOSMP 标志来指示采样阶段的结束。EOSMP 标志通过软件写 1 清零。若置位了 ADC_IER 寄存器中的 EOSMPIE 位，则还会产生一个中断。

这些中断的目的是为了处理与转换保持同步。典型地，一个模拟多路复用器可以在转换阶段的隐性时间内进行使用，所以多路复用器会定位在下次采样开始的状态。

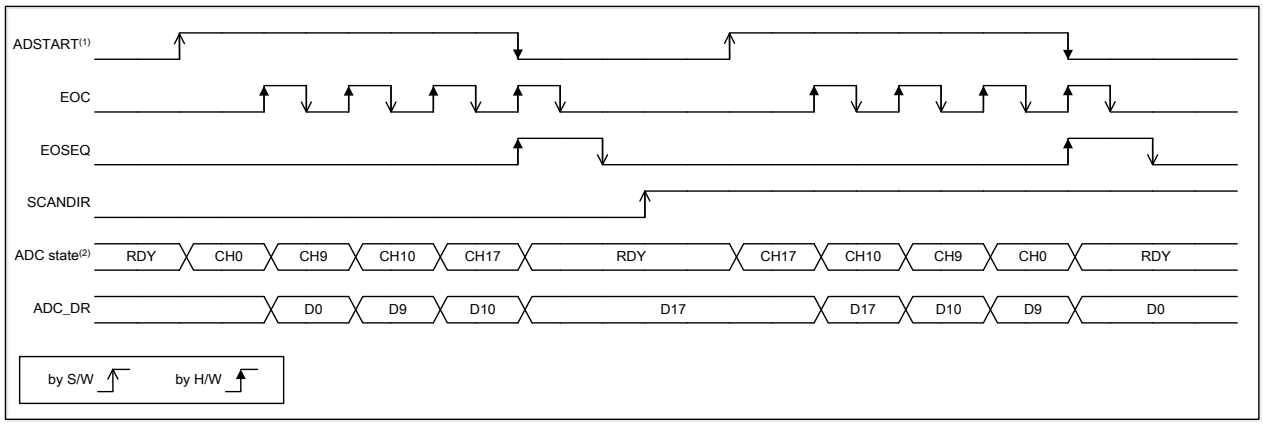
注：仅仅有非常短的时间在采样结束和转换结束之间，推荐使用轮询方式或者 WFE 指令而不是中断方式和 WFI 指令。

13.5.4. 转换序列结束（EOSEQ 标志）

ADC 通知应用每一次序列结束（EOSEQ）事件。

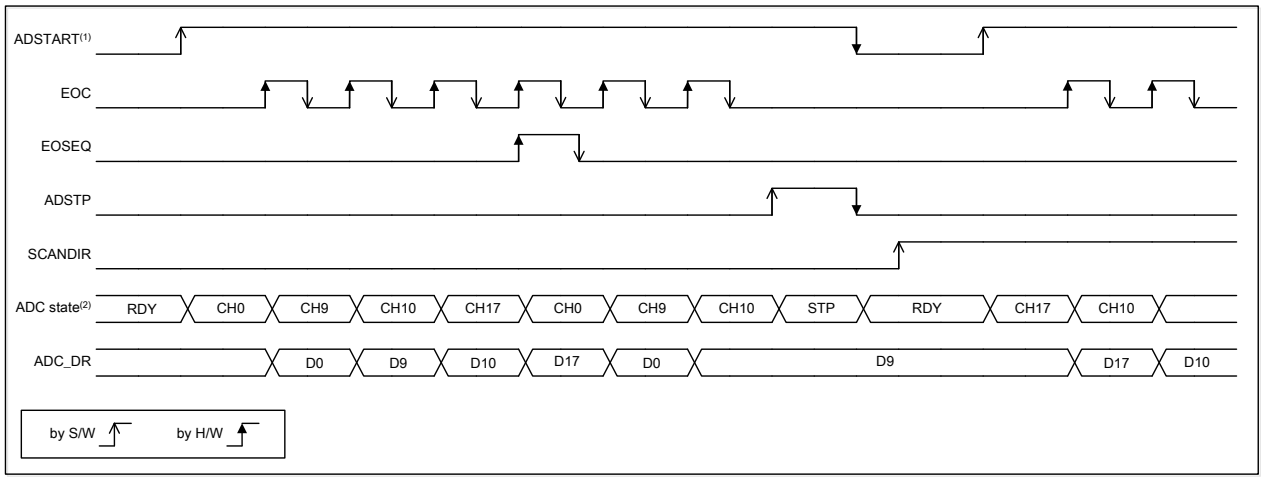
在最后一次转换序列的数据结果有效存放于 ADC_DR 寄存器的同时，ADC 置位 ADC_ISR 寄存器的 EOSEQ 标志。若置位了 ADC_IER 寄存器的 EOSEQIE 位，则还会产生一个中断。EOSEQ 标志通过软件写 1 清零。

13.5.5. 示例时序图（单次/连续模式，硬件/软件触发）



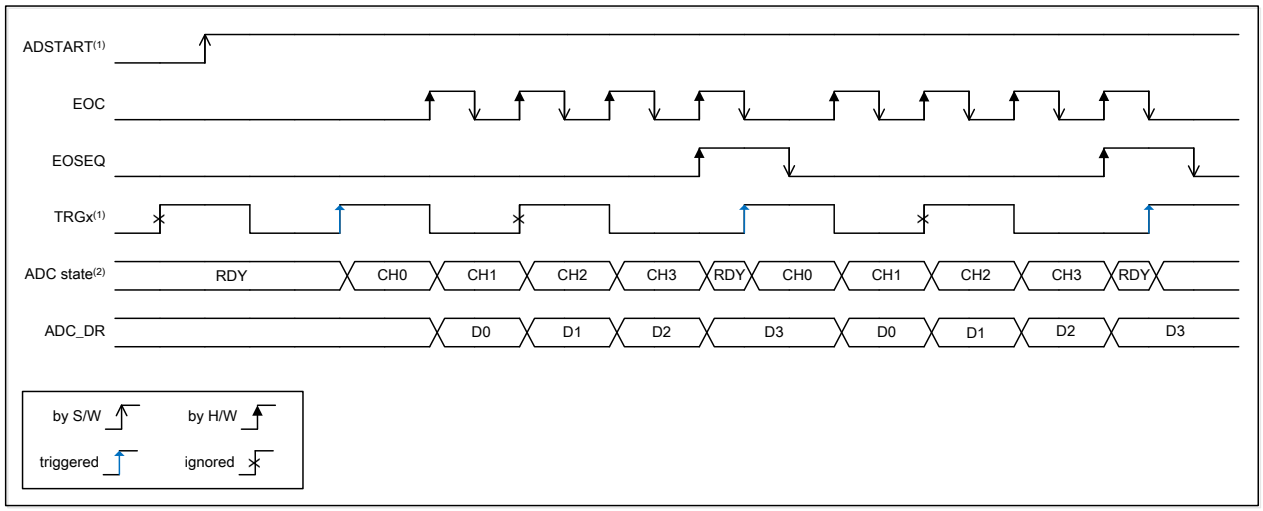
1. EXTEN=00, CONT=0
2. CHSEL=0x20601, WAIT=0, AUTOFF=0

图 13.10. 序列的单次转换，软件触发



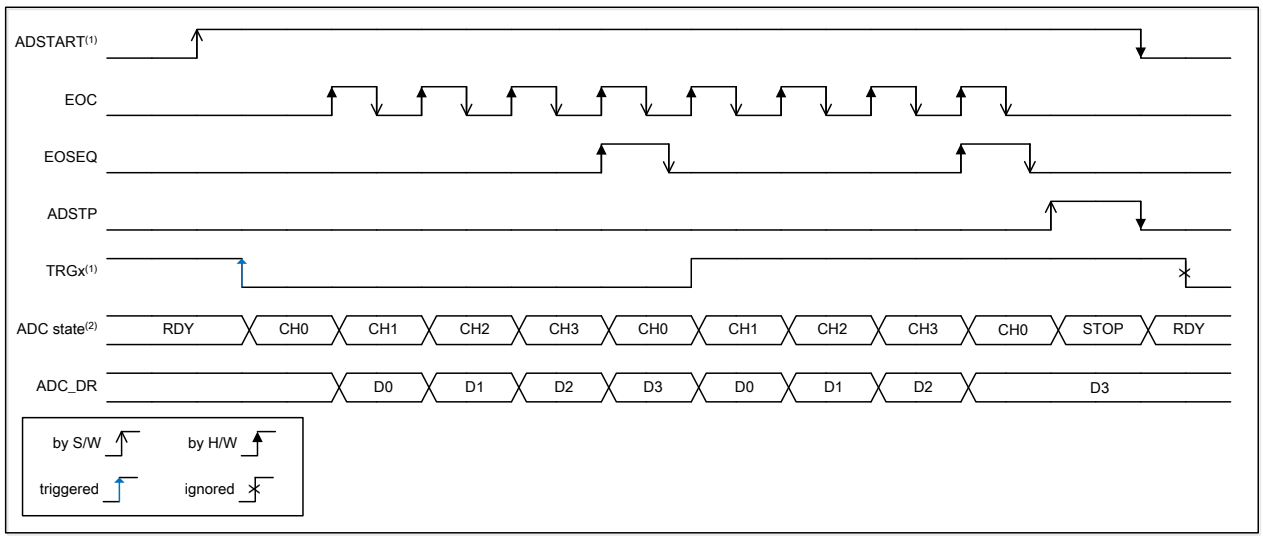
1. EXTEN=00, CONT=1
2. CHSEL=0x20601, WAIT=0, AUTOFF=0

图 13.11. 序列的连续转换，软件触发



1. EXTSEL=TRGx(超频), EXTEN=01(上升沿), CONT=0
2. CHSEL=0xF, SCANDIR=0, WAIT=0, AUTOFF=0

图 13.12. 序列的单次转换，硬件触发



1. EXTSEL=TRGx, EXTEN=10(下降沿), CONT=1
2. CHSEL=0xF, SCANDIR=0, WAIT=0, AUTOFF=0

图 13.13. 序列的连续转换，硬件触发

13.6. 数据管理

13.6.1. 数据寄存器和数据对齐（ADC_DR，ALIGN）

在每一次转换的结束（当一个 EOC 事件发生时），已转换的数据结果被存放于有 16 位宽度的 ADC_DR 数据寄存器。

ADC_DR 的格式依赖于已配置的数据对齐和分辨率。

ADC_CFGR1 寄存器中的 ALIGN 位选择转换后存储的数据对齐方式。数据可以是右对齐（ALIGN=0）或者左对齐（ALIGN=1），如图 13.14 所示。

ALIGN	RES	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0	0x0				DR[11:0]											
	0x1	0x00						DR[9:0]									
	0x2	0x00								DR[7:0]							
	0x3	0x000										DR[5:0]					
1	0x0	DR[11:0]												0x0			
	0x1	DR[9:0]										0x00					
	0x2	DR[7:0]								0x00							
	0x3	0x00								DR[5:0]						0x0	

图 13.14. 数据对齐和分辨率

13.6.2. ADC 溢出（OVR，OVRMOD）

溢出标志（OVR）指示数据溢出事件。在新的转换数据有效时，已转换的数据还没有被 CPU 或者 DMA 及时读取，就会发生溢出。

当新的转换已经完成的同时，如果 EOC 标志仍旧为 1，则会置位 ADC_ISR 寄存器中的 OVR 标志。若置位了

ADC_IER 寄存器中的 OVRIE 位，则还会产生一个中断。

当一个溢出条件发生，ADC 保持操作并且继续转换，除非软件决定通过设置 ADC_CR 寄存器的 ADSTP 位来停止和复位序列。

OVR 标志通过软件写 1 清零。

通过编程 ADC_CFGR1 寄存器中的 OVRMOD 位，可以配置数据在溢出事件发生时是保留还是覆盖。

- OVRMOD=0
  - 溢出事件保留即将被覆盖的数据寄存器：老的数据被保持，而新的转换被忽略。如果 OVR 标志保持为 1，后续的转换可以被执行，但结果数据会被忽略。
- OVRMOD=1
  - 数据寄存器会被最后一次转换结果覆盖 ,而上一次未读取的数据将会丢失。如果 OVR 标志保持为 1，后续的转换会被执行，并且 ADC_DR 寄存器始终包含最后一次转换的数据。

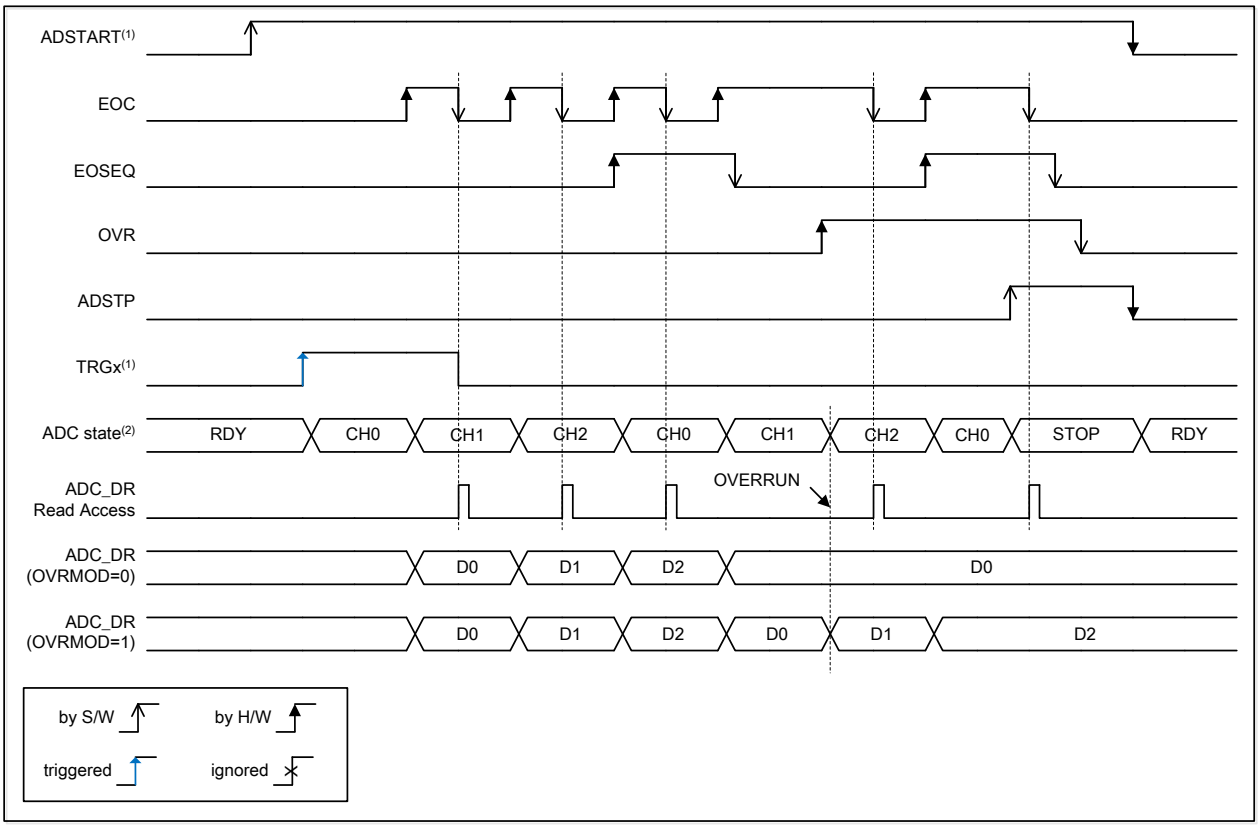


图 13.15. 溢出示例（OVR）

13.6.3. 不使用 DMA 管理一序列已转换的数据

如果转换足够慢，转换序列能够由软件处理。在这种情况下，软件必须使用 EOC 标志和相关的中断来处理每一个数据结果。每一次转换完成后，ADC_ISR 寄存器中的 EOC 位都会被置位，同时 ADC_DR 寄存器可被读取。此时 ADC_CFGR1 寄存器的 OVRMOD 位应该配置成 0 来管理溢出事件作为一个错误事件。

13.6.4. 在不使用 DMA 和无溢出下管理已转换的数据

该方式有利于使 ADC 转换单个或多个通道而不去读取每次转换之后数据的应用。在这种情况下，OVRMOD 位必须配置成 1，并且 OVR 标志应该被软件忽略。当 OVRMOD=1 时，一个溢出事件不会阻止 ADC 继续转换，并且 ADC_DR 寄存器始终包含最后一次转换数据。

13.6.5. 使用 DMA 管理已转换的数据

因为所有转换通道数值都会存放在单个数据寄存器 ,所以在转换超过一个通道的情况下使用 DMA 会更有效率。这可以避免丢失存储在 ADC_DR 寄存器中的转换数据结果。
当 DMA 模式使能的时候（设置 ADC_CFGR1 寄存器的 DMAEN 位为 1）, 在每一个通道转换结束后产生一个 DMA 请求。这样允许将转换数据从 ADC_DR 寄存器中传到软件指定的目标位置。
注：ADC_CFGR1 寄存器中的 DMAEN 位必须要在 ADC 校准阶段完成之后才可设置。
即使这样，如果因为 DMA 不能够及时服务 DMA 传输请求而导致溢出发生（OVR=1）的话，ADC 停止产生 DMA 请求 , 并且对应新转换的数据不会再通过 DMA 进行传输。这意味着所有已经传输到 RAM 的数据都可以认为是有效的。
根据 OVRMOD 位的配置，数据可以被保留或者覆盖。（参考章节 13.6.2：ADC 溢出（OVR，OVRMOD））。DMA 传输请求会被阻塞住，直到软件清零 OVR 位。
根据应用场合提出了两种不同的 DMA 模式，通过 ADC_CFGR1 寄存器的 DMACFG 位进行配置：

- DMA 单次模式（DMACFG=0）
  当 DMA 被编程用于传输一个固定数量的数据时，选用该模式。
- DMA 循环模式（DMACFG=1）
  当 DMA 被编程为循环模式或者双缓冲模式时，选用该模式。

DMA 单次模式（DMACFG=0）
在该模式中 ,ADC 在每一次新的转换数据有效的时候产生一个 DMA 请求。一旦 DMA 已经到达最后一次 DMA 传输，即使转换已经再次启动，ADC 仍会停止产生 DMA 请求（当 DMA_EOT 中断发生时，查看章节 10:直接存储器访问控制器（DMA））。

当 DMA 传输完成时（所有配置于 DMA 控制器的传输均已完成）:

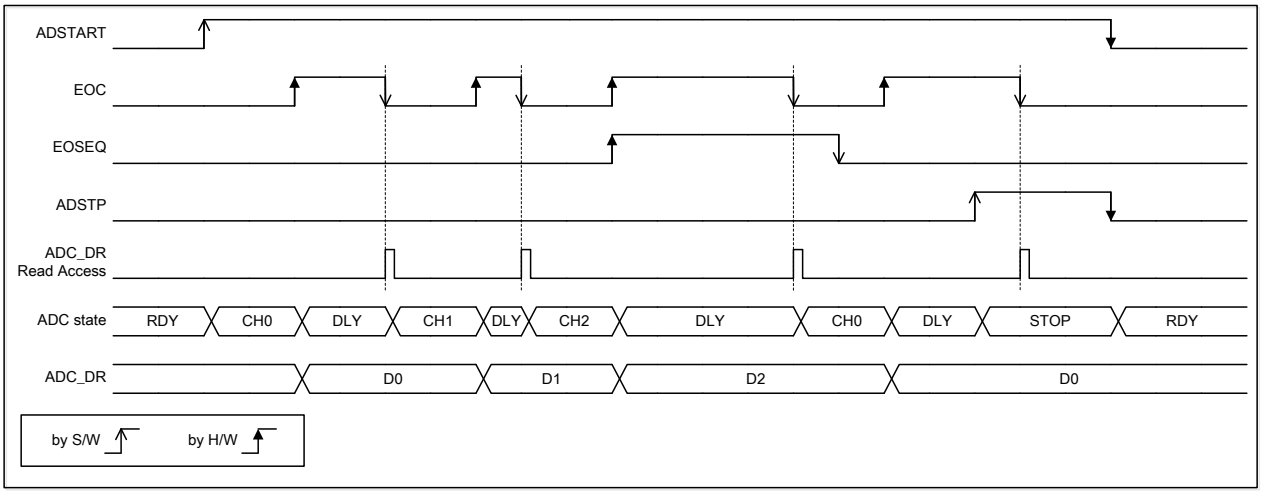
- ADC 数据寄存器的内容将被冻结。
- 任何正在进行的转换将被停止，并且其不完整的结果会被忽略。
- 没有新的 DMA 请求再发给 DMA 控制器。假如还有已经启动的转换，这可以避免产生一个溢出错误。
- 扫描序列将被停止和复位。
- DMA 会被停止

DMA 循环模式（DMACFG=1）
在该模式中，即使 DMA 已经到达最后一次 DMA 传输，ADC 总是在每一次新的转换数据有效的时候产生一个 DMA 请求。这允许 DMA 被配置在循环模式来处理一个连续的模拟输入数据流。

13.7. 低功耗特性

13.7.1. 等待模式转换

等待模式转换能被用于简化软件以及优化有 ADC 溢出发生风险的低频应用的性能。
当 ADC_CFGR1 寄存器的 WAIT 位被设置为 1，新的转换仅可在前一个数据已经被处理后才能启动，比如 ADC_DR 寄存器已经被读取或者 EOC 位已经被清零。
这是一种能自动适配 ADC 速度和读取数据的系统速度的方法。
注：任何发生在转换正在进行时或者读操作之前的等待时间中的硬件触发都会被忽略。

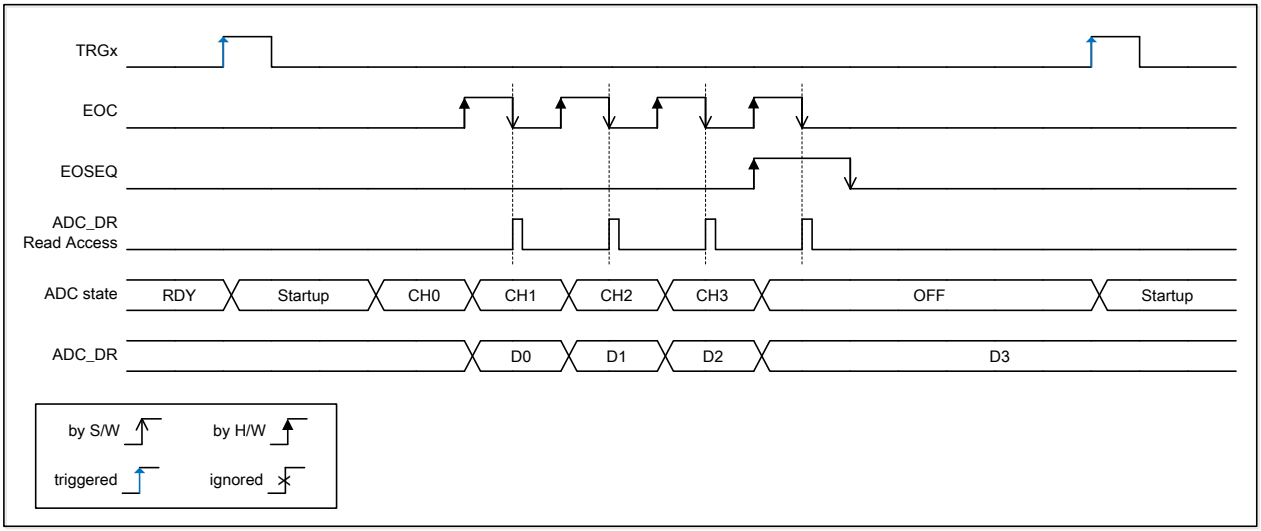


1. EXTEN=00, CONT=1
2. CHSEL=0x7, SCANDIR=0, WAIT=1, AUTOFF=0

图 13.16. 等待模式转换（连续模式，软件触发）

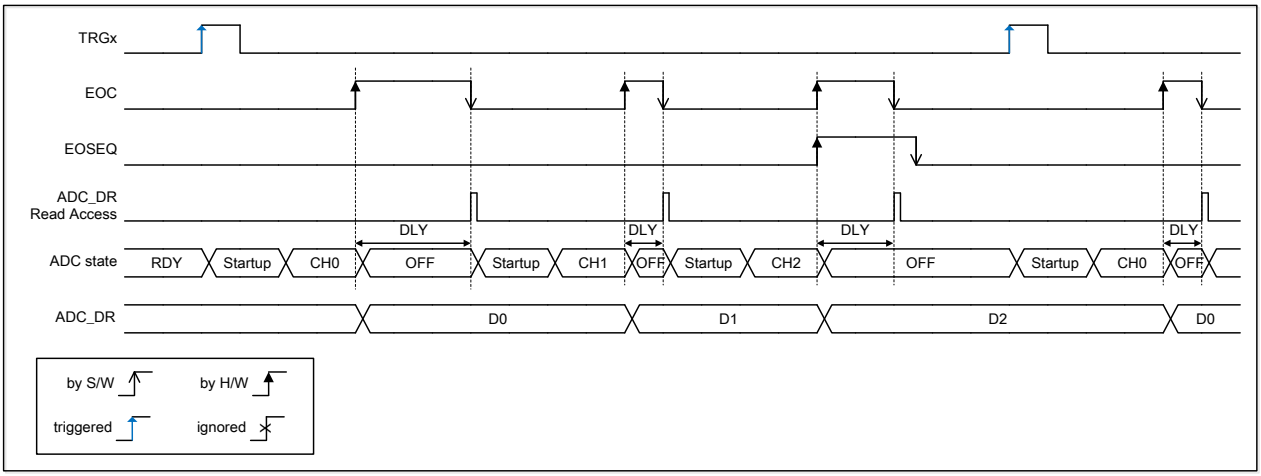
13.7.2. 自动关闭模式（AUTOFF）

ADC 有一个叫做自动关闭模式的自动功耗管理功能，通过设置 ADC_CFGR1 寄存器的 AUTOFF 位为 1 来使能。
当 AUTOFF=1 时，ADC 在没有转换的时候始终关闭，并且在转换启动的时候自动重启（通过软件或硬件触发）。一个自动启动时间自动插入在启动转换的触发事件和 ADC 采样时间之间。一旦转换序列完成，ADC 将会自动禁用。
对于那些只需要相对少量转换或者转换请求时间间隔足够长（例如低频率硬件触发）的应用，自动关闭模式能使得的功耗急剧减少，证明用于切换 ADC 开和关的额外功耗和额外时间是合理的。
自动关闭模式能跟等待模式转换（WAIT=1）组合在一起用于低频应用。如果 ADC 在等待阶段内自动关闭并且在 ADC_DR 寄存器被应用读取的同时重启，这种组合能提供很显著的省电。
注：请参考章节 7：复位和时钟控制（RCC），查看如何管理专用 14MHz 内部振荡器的说明。ADC 接口能自动切换开/关 14MHz 内部振荡器来节电。



1. EXTSEL=TRGx, EXTEN=01(上升沿), CONT=x, ADSTART=1, CHSEL=0xF, SCANDIR=0, WAIT=0, AUTOFF=1

图 13.17. 在 WAIT=0，AUTOFF=1 下的行为



1. EXTSEL=TRGx, EXTEN=01(上升沿), CONT=x, ADSTART=1, CHSEL=0x7, SCANDIR=0, WAIT=1, AUTOFF=1

图 13.17. 在 WAIT=1，AUTOFF=1 下的行为

13.8. 模拟窗口看门狗（AWDEN，AWDSGL，AWDCH，AWD_HTR/LTR，AWD）

AWD 模拟看门狗功能通过设置 ADC_CFGR1 寄存器的 AWDEN 位来使能。用来监控选中的单个通道或者所有已使能的通道（查看表 13.7：模拟看门狗通道选择），保持通道在配置的电压范围内，如图 13.18 所示。

如果 ADC 转换的模拟电压在低阈值以下或者在高阈值以上，AWD 模拟看门狗状态位会被置位。这些阈值被编程到 ADC_HTR 和 ADC_LTR 16 位寄存器的低 12 有效位。通过设置 ADC_IER 寄存器的 AWDIE 位可以使能中断。

AWD 标志通过软件写 1 来清零。

当转换小于 12 位分辨率的数据时（由 RES[1:0]位决定），已编程阈值的低有效位必须保持清零，因为内部比较器总是以全 12 位原始转换数据（左对齐）的方式执行。

表 13.7 描述了所有可能的分辨率下的比较。

表 13.7. 模拟看门狗比较

分辨率位	模拟看门狗比较		说明
RES[1:0]	原始转换数据，左对齐(1)	阈值	
00: 12 位	DATA[11:0]	LT[11:0]和 HT[11:0]	-
01: 10 位	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 位	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 位	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”

1. 看门狗比较的是任何对齐计算之前的原始转换数据。

表 13.8 展示如何配置 ADC_CFGR1 寄存器中的 AWDSGL 和 AWDEN 位来使能一个或多个通道的模拟看门狗。

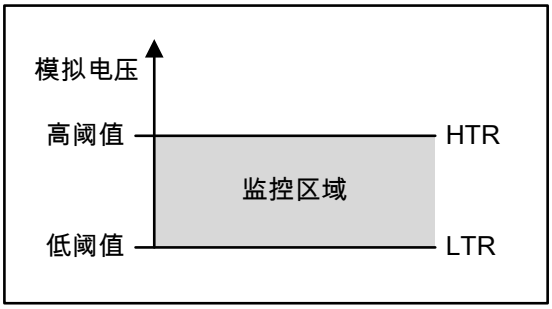


图 13.18. 模拟看门狗监控区域

表 13.8. 模拟看门狗通道选择

模拟看门狗监控的通道	AWDSGL 位	AWDEN 位
无	x	0
所有通道	0	1
单个通道(1)	1	1

1. 通过 AWDCH[4:0]位进行选择。

13.9. 温度传感器和内部基准电压

温度传感器用于测量器件的结点温度(T_J)。温度传感器在内部连接到 ADC_IN16 输入通道，用于转换传感器输出电压成一个数值。温度传感器模拟引脚的采样时间必须大于数据手册中定义的最小 T_{S_temp} 值。未使用的时候，传感器可以置于关闭模式。

温度传感器的输出电压与温度成线性关系，但是这个特性可能会因为工艺变化而导致芯片到芯片有很显著的变化。为了提高温度传感器的精度（特别是绝对温度测量），校准值会由 FMD 在产品测试中对每一部分单独测量，并存放到系统存储区域。额外信息参考对应的器件数据手册。

内部电压基准(V_{REFINT})为 ADC 和比较器提供一个稳定（带隙）电压输出。V_{REFINT} 在内部连接到 ADC_IN17 输入通道。

VREFINT 的精准电压值由 FMD 在产品测试中对每一部分单独测量，并存放到系统存储区域。只可只读模式访问。

图 13.19 展示温度传感器、内部电压基准和 ADC 之间的连接框图。

TSEN 位必须置位来使能 ADC_IN16 的转换（温度传感器），INTVREFEN 位必须置位来使能 ADC_IN17 的转

换（V_{REFINT}）。

主要特性

- 支持温度范围：-40 到 105℃
- 线性度：最大±2℃，精度取决于校准

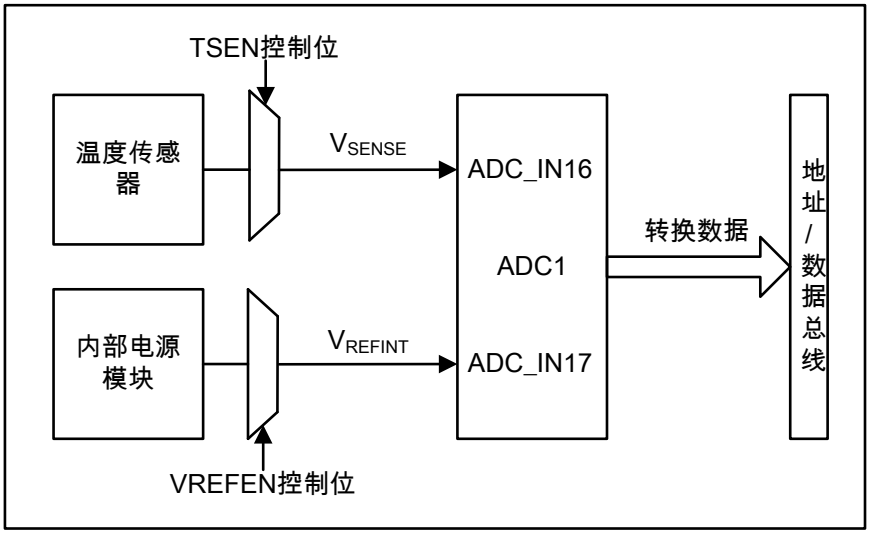


图 13.19. 温度传感器和 VREFINT 通道框图

读温度

1. 选择 ADC_IN16 输入通道
2. 选择一个器件数据手册中定义的合适的采样时间（T_{S_temp}）。
3. 置位 ADC_CCR 寄存器中的 TSEN 位来唤醒关闭模式下的温度传感器，并等待稳定时间（t_{START}）
4. 通过置位 ADC_CR 寄存器中的 ADSTART 位来启动 ADC 转换（或者通过外部触发）
5. 读取 AC_DR 寄存器中的结果数据
6. 通过下列公式计算实际温度：

$$温度(℃) = \frac{110℃ - 25℃}{TS\_CAL2 - TS\_CAL1} \times (TS\_DATA - TS\_CAL1) + 25℃$$

此处：

- TS_CAL2 是在 110℃下获得的温度传感器校准值
- TS_CAL1 是在 25℃下获得的温度传感器校准值
- TS_DATA 是 ADC 转换的实际温度传感器输出值

  有关 TS_CAL1 和 TS_CAL2 校准点的更多信息，请参考特定的设备数据手册。

注：温度传感器在关闭模式唤醒之后到输出 V_{SENSE} 正确电平之前需要有一个启动时间。ADC 在上电之后也有一个启动时间，所以为了最小化延时，ADEN 和 TSEN 位应该同时被置位。

通过内部基准电压计算实际 V_{DDA} 电压值

应用于微控制器的 V_{DDA} 电源供电电压可能会发生变化或者没有已知的那么精准。嵌入的内部电压基准（V_{REFINT}）及其由 ADC 在 V_{DDA}=3.3V 制造工艺中获取的校准数据，可以用于评估实际 V_{DDA} 电压电平。

下列公式给出器件供电的实际 VDDA 电压：

$$V_{DDA} = 3.3V \times VREFINT\_CAL / VREFINT\_DATA$$

说明：

- VREFINT_CAL 是 VREFINT 校准值
- VREFINT_DATA 是 ADC 转换的实际 VREFINT 输出值

转换与供电有关系的 ADC 测量成为一个绝对电压数值

ADC 是设计来提供一个数字值，该数字值对应模拟电源供电和转换通道上施加的电压之比。对于大多数应用程序用例，有必要将这个比率转换为与 V_{DDA} 无关的电压。对于已知 V_{DDA} 数值和 ADC 转换值是右对齐的应用程序，可以使用以下公式得到这个绝对值：

$$V_{CHANNELx} = \frac{V_{DDA}}{FULL\_SCALE} \times ADC\_DATAx$$

对于 V_{DDA} 数值未知的应用，必须使用内部电压基准，V_{DDA} 可以使用章节：使用内部基准电压来计算实际 V_{DDA} 电压中提供的表达式来替换，得到如下公式：

$$V_{CHANNELx} = \frac{3.3V \times VREFINT\_CAL \times ADC\_DATAx}{VREFINT\_DATA \times FULL\_SCALE}$$

说明：

- VREFINT_CAL 是 VREFINT 校准值
- ADC_DATAx 是 ADC 在通道 x 上测量到的数值（右对齐）
- VREFINT_DATA 是 ADC 转换的实际 VREFINT 输出值
- FULL_SCALE 是 ADC 输出的最大数字值。例如 12 位分辨率为 2^{12}-1=4095 或者 8 位分辨率时为 2^{8}-1=255。

注：如果 ADC 测量使用的是 12 位右对齐之外的输出格式，那么在进行计算之前，必须首先将所有参数转换为兼容的格式。

13.10. ADC 专用的内部参考电压源

默认情况下，ADC_CR2 寄存器的 VREFEN 为 0，内部参考电压源不使能，ADC 的参考电压选择 VDDA。
配置 VREFSEL 来选择内部参考电压源的档位，仅 1 档可选：2.5V，其余保留。
配置完 VREFSEL 后再置位 VREFEN 位来使能内部参考电压源，此时 ADC 的参考电压将会选择内部参考电压源。

注意：ADC 的参考电压配置要在 ADC 转换未进行时才可进行更改，并且有一段的启动稳定时间，具体参考 ADC 的电气特性。

13.11. ADC 中断

通过以下事件中的任意一个产生中断：

- ADC 上电，在 ADC 准备好时（ADRDY 标志）
- 任何一个转换结束（EOC 标志）
- 序列转换结束（EOSEQ 标志）
- 在模拟看门狗检测事件发生时（AWD 标志）
- 在采样阶段结束发生时（EOSMP 标志）

- 在数据溢出发生时（OVR 标志）

独立中断使能位可用于灵活设置 ADC 中断。

表 13.9. ADC 中断

中断事件	事件标志	使能控制位
ADC 准备好	ADRDY	ADRDYIE
转换结束	EOC	EOCIE
序列转换结束	EOSEQ	EOSEQIE
看门狗状态位被置位	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
溢出	OVR	OVRIE

13.12. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	ADC_ISR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	AWD	-	-	OVR	EOSEQ	EOC	EOSMP	ADRDY
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0	0	0	0	0
0x04	ADC_IER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	AWDIE	-	-	OVRIE	EOSEQIE	EOCIE	EOSMPIE	ADRDYIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	0	0	0	0	0
0x08	ADC_CR	ADCAL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ADSTP	-	ADSTART	ADDIS	ADEN
	Reset	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0
0x0C	ADC_CFGR1	-	AWDCH[4:0]					-	-	AWDEN	AWDSGL	-	-	-	-	-	DISCEN	AUTOFF	WAIT	CONT	OVRMOD	EXTEN [1:0]		-	EXTSEL [2:0]			ALIGN	RES [1:0]		SCANDIR	DMACFG	DMAEN
	Reset	x	0	0	0	0	0	x	x	0	0	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0
0x10	ADC_CFGR2	CKMODE [1:0]		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Reset	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x14	ADC_SMPR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SMP [2:0]		
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0
0x20	ADC_TR	-	-	-	-	HT[11:0]												-	-	-	-	LT[1:0]											
	Reset	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
0x28	ADC_CHSELR	-	-	-	-	-	-	-	-	-	-	-	-	CHSEL19	CHSEL18	CHSEL17	CHSEL16	CHSEL15	CHSEL14	CHSEL13	CHSEL12	CHSEL11	CHSEL10	CHSEL9	CHSEL8	CHSEL7	CHSEL6	CHSEL5	CHSEL4	CHSEL3	CHSEL2	CHSEL1	CHSEL0
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	ADC_DR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DATA[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x308	ADC_CCR	-	-	-	-	-	-	-	-	TSEN	INTVREFEN	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	Reset	x	x	x	x	x	x	x	x	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x30C	ADC_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IO_SMPEN	IO_AMPEN	-	-	-	VREF_DECIB	VREFSEL [1:0]		VREFEN	-
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	x	x	1	0	0	0	x

13.12.1. ADC_ISR（ADC 中断和状态寄存器）

地址偏移：0x00

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	AWD	—		OVR	EOSEQ	EOC	EOSMP	ADRDY
类型	RC_W1	RO-0	RO-0	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	Function
31:8	NA	保留位，未定义
7	AWD	模拟看门狗标志 当转换电压超出 ADC_LTR 和 ADC_HTR 寄存器编程的数值范围时，该位由硬件置位。软件写 1 清零。 0：没有模拟看门狗事件发生（或者标志事件已被软件确认并清除） 1：发生模拟看门狗事件
6:5	NA	保留位，未定义
4	OVR	ADC 溢出 当溢出发生时该位由硬件置位，若 EOC 标志已被置位则意味着新的转换已经完成。软件写 1 清零。 0：没有溢出发生（或者标志事件已被软件确认并清除） 1：溢出已经发生
3	EOSEQ	序列结束标志 CHSEL 位所选通道的序列转换结束时，该位由硬件置位。软件写 1 清零。 0：转换序列未完成（或者标志事件已被软件确认并清除） 1：转换序列已完成
2	EOC	转换结束标志 在每一次通道转换结束时，ADC_DR 寄存器中的新数据结果有效，该位由硬件置位。软件写 1 清零，或者读 ADC_DR 寄存器清零。 0：通道转换未完成（或者标志事件已被软件确认并清除） 1：通道转换已完成
1	EOSMP	采样结束标志 在转换期间的采样阶段结束时，该位由硬件置位。软件写 1 清零。 0：未到采样阶段结束时刻（或者标志事件已被软件确认并清除） 1：采样阶段已到达

0	ADRDY	ADC 准备好 ADC 已被使能(ADEN=1)后 ,当 ADC 已到达准备好接收转换请求的状态时，该位由硬件置位。软件写 1 清零。 0：ADC 还未准备好启动转换（或者标志事件已被软件确认并清除） 1：ADC 已准备好启动转换 注：在自动关闭模式中（AUTOFF=1），电源开/关阶段会被硬件自动执行，ADRDY 标志不会被置位。

13.12.2. ADC_IER（ADC 中断使能寄存器）

地址偏移：0x04

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	AWDIE	—		OVRIE	EOSEQIE	EOCIE	EOSMPIE	ADRDYIE
类型	RW	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	AWDIE	模拟看门狗中断使能 软件置位和清零，用来使能或禁用模拟看门狗中断。 0：模拟看门狗中断已禁用 1：模拟看门狗中断已使能 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。
6:5	NA	保留位，未定义
4	OVRIE	溢出中断使能 软件置位和清零，用来使能或禁用溢出中断。 0：溢出中断已禁用 1：溢出中断已使能。中断会在 OVR 位被置位时产生。 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。
3	EOSEQIE	序列转换结束中断使能 软件置位和清零，用来使能或禁用序列转换结束中断。 0：EOSEQ 中断已禁用 1：EOSEQ 中断已使能。中断会在 EOSEQ 位被置位时产生。 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。
2	EOCIE	转换结束中断使能

		软件置位和清零，用来使能或禁用转换结束中断。 0：EOC 中断已禁用 1：EOC 中断已使能。中断会在 EOC 位被置位时产生。 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。
1	EOSMPIE	采样结束中断使能 软件置位和清零，用来使能或禁用采样结束中断。 0：EOSMP 中断已禁用 1：EOSMP 中断已使能。中断会在 EOSMP 位被置位时产生。 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。
0	ADRDYIE	ADC 准备好中断使能 软件置位和清零，用来使能或禁用 ADC 准备好中断。 0：ADRDY 中断已禁用 1：ADRDY 中断已使能。中断会在 ADRDY 位被置位时产生。 注：仅在 ADSTART=0（确保没有转换正在进行时）可写该位。

13.12.3. ADC_CR（ADC 控制寄存器）

地址偏移：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	ADCAL	—						
类型	RS	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—			ADSTP	—	ADSTART	ADDIS	ADEN
类型	RO-0	RO-0	RO-0	RS	RO-0	RS	RS	RS

Bit	Name	Function
31	ADCAL	ADC 校准 由软件置位来启动 ADC 校准。 在校准完成后由硬件清零。 0：校准完成 1：写 1 校准 ADC。读 1 意味着校准进行中。 注：仅在 ADC 禁用（ADCAL=0，ADSTART=0，ADSTP=0，ADDIS=0，ADEN=0）时可置位 ADCAL。
30:5	NA	保留位，未定义
4	ADSTP	ADC 停止转换命令 由软件置位来停止和忽略正在进行的转换（ADSTP 命令）。 在转换实际上已经被废弃，并且 ADC 已准备好接收新的启动转换命令时，由

		硬件清零。 0：没有 ADC 停止转换命令正在进行 1：写 1 停止 ADC。读 1 意味着 ADSTP 命令正在进行。 注：仅在 ADSTART=1 和 ADDIS=0（ADC 已使能，并且可能正在转换，并且没有挂起的 ADC 禁用请求）时可置位 ADSTP。
3	NA	保留位，未定义
2	ADSTART	ADC 启动转换命令 由软件置位来启动 ADC 转换。根据 EXTEN[1:0]配置位，转换可以马上开始（软件触发配置），也可以等硬件触发事件发生时（硬件触发配置）。 由硬件清零： — 在单次转换模式中（CONT=0，DISCEN=0），若选择软件触发（EXTEN=00）则在序列转换结束（EOSEQ）标志断言时清零。 — 在断续转换模式中（CONT=0，DISCEN=1），若选择软件触发（EXTEN=00）则在转换结束（EOC）标志断言时清零。 — 在所有其它情况下：在执行完 ADSTP 命令之后清零，同时 ADSTP 位也由硬件清零。 0：没有 ADC 转换正在进行。 1：写 1 启动 ADC。读 1 意味着 ADC 正在操作并且可能正在转换。 注：仅在 ADEN=1 和 ADDIS=0（ADC 已使能并且没有挂起的禁止 ADC 请求）时可置位 ADSTART。
1	ADDIS	ADC 禁用命令 由软件置位来禁用 ADC（ADDIS 命令）并且将 ADC 置于关闭状态（OFF 状态）。 在 ADC 已经完全禁用时由硬件清零（ADEN 也同时由硬件清零）。 0：没有 ADDIS 命令正在进行 1：写 1 禁用 ADC。读 1 意味着 ADDIS 命令正在执行中。 注：仅在 ADEN=1 和 ADSTART=0（确保没有转换正在进行）时可置位 ADDIS。
0	ADEN	ADC 使能命令 由软件置位来使能 ADC。一旦 ADRDY 标志已被设置，ADC 将完全准备好操作。 当 ADC 在 ADDIS 命令执行后被禁用时，由硬件清零。 0：ADC 已被禁用（OFF 状态） 1：写 1 使能 ADC。 注：仅在 ADC_CR 寄存器中的所有位为 0（ADCAL=0，ADSTP=0，ADSTART=0，ADDIS=0，ADEN=0）时可置位 ADEN。

13.12.4. ADC_CFGR1（ADC 配置寄存器 1）

地址偏移：0x0C
复位值：0x0000 0000

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

31:24	—	AWDCH[4:0]					—	
类型	RO-0	RW	RW	RW	RW	RW	RO-0	RO-0
23:16	AWDEN	AWDSGL	—					DISCEN
类型	RW	RW	RO-0	RO-0	RO-0	RO-0	RO-0	RW
15:8	AUTOFF	WAIT	CONT	OVRMOD	EXTEN[1:0]		—	EXTSEL[2]
类型	RW	RW	RW	RW	RW	RW	RO-0	RW
7:0	EXTSEL[1:0]		ALIGN	RES[1:0]		SCANDIR	DMACFG	DMAEN
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31	NA	保留位，未定义
30:26	AWDCH[4:0]	模拟看门狗通道选择 这些位由软件置位和清零。他们选择模拟看门狗监视的输入通道。 00000：由模拟看门狗监视的 ADC 模拟输入通道 0 00001：由模拟看门狗监视的 ADC 模拟输入通道 1 ..... 10011：由模拟看门狗监视的 ADC 模拟输入通道 19 其他值：保留，必须不使用。 注：AWDCH[4:0]位选中的通道也必须在 CHSELR 寄存器中设置。 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。
25:24	NA	保留位，未定义
23	AWDEN	模拟看门狗使能 该位由软件置位和清零。 0：模拟看门狗已禁用 1：模拟看门狗已使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
22	AWDSGL	在单一通道或者所有通道上使能看门狗 该位由软件置位和清零，用来使能由 AWDCH[4:0]位指定的通道或者所有通道。 0：在所有通道上使能模拟看门狗 1：在单个通道上使能模拟看门狗 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
21:17	NA	保留位，未定义
16	DISCEN	断续模式 该位由软件置位和清零，用来使能/禁用断续模式。 0：断续模式禁用 1：断续模式使能 注：不可能同时使能断续模式和连续模式：禁止同时置位 DISCEN 和 CONT 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
15	AUTOFF	自动关闭模式 该位由软件置位和清零，用来使能/禁用自动关闭模式。 0：自动关闭模式禁用

		1：自动关闭模式使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
14	WAIT	等待转换模式 该位由软件置位和清零，用来使能/禁用等待转换模式。 0：等待转换模式禁用 1：等待转换模式使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
13	CONT	单次/连续转换模式 该位由软件置位和清零。若置位则转换将持续进行，直到清除为止。 0：单次转换模式 1：连续转换模式 注：不可能同时使能断续模式和连续模式：禁止同时置位 DISCEN 和 CONT 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
12	OVRMOD	溢出管理模式 该位由软件置位和清零，用来配置数据溢出的管理方式。 0：当检测到溢出时，ADC_DR 寄存器保持为老的数据 1：当检测到溢出时，ADC_DR 寄存器用最后一次转换结果覆盖 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
11:10	EXTEN[1:0]	外部触发使能和极性选择 这些位由软件置位和清零，用来选择外部触发极性和使能触发器。 00：硬件触发检测禁用（可由软件启动转换） 01：在上升沿进行硬件触发检测 10：在下降沿进行硬件触发检测 11：在上升和下降沿都进行硬件触发检测 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
9	NA	保留位，未定义
8:6	EXTSEL[2:0]	外部触发选择 这些位选择用于触发转换启动的外部事件（详见表 12.5：外部触发）: 000：TRG0 001：TRG1 010：TRG2 011：TRG3 100：TRG4 101：TRG5 110：TRG6 111：TRG7 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
5	ALIGN	数据对齐 该位由软件置位或清零，用来选择右或左对齐。参考图 12.12：数据对齐和分辨率。 0：右对齐 1：左对齐 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。

4:3	RES[1:0]	数据分辨率 这些位由软件写入，用来选择转换的分辨率。 00：12 位 01：10 位 10：8 位 11：6 位 注：仅在 ADEN=0 时可写这些位。
2	SCANDIR	扫描序列方向 该位由软件置位和清零，用来选择在序列中扫描通道的方向。 0：向前扫描（从 CHSEL0 到 CHSEL19） 1：向后扫描（从 CHSEL19 到 CHSEL0） 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
1	DMACFG	直接存储访问配置 该位由软件置位和清零，用来选择两种 DMA 操作模式，仅在 DMAEN=1 时有效。 0：选择 DMA 单次模式 1：选择 DMA 循环模式 详见章节 12.6.5：使用 DMA 管理已转换的数据。 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
0	DMAEN	直接存储访问使能 该位由软件置位和清零，用来使能 DMA 请求的产生。允许使用 DMA 控制器来自动管理转换数据。详见章节 12.6.5：使用 DMA 管理已转换的数据。 0：DMA 禁用 1：DMA 使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。

13.12.5. ADC_CFGR2（ADC 配置寄存器 2）

地址偏移：0x10
复位值：0x0000 0000

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

Bit	Name	Function

31:30	CKMODE[1:0]	ADC 时钟模式 这些位由软件置位和清零，用于定义模拟 ADC 怎样被时钟驱动： 00：ADCCLK（异步时钟模式），在产品级产生（参考 RCC 章节） 01：PCLK/2（同步时钟模式） 10：PCLK/4（同步时钟模式） 11：保留 在所有同步时钟模式中，从定时器触发到转换启动的延迟没有抖动。 注：仅在 ADC 禁用（ADCAL=0，ADSTART=0，ADSTP=0，ADDIS=0，ADEN=0）时可写这些位。
29:0	NA	保留位，未定义

13.12.6. ADC_SMPR（ADC 采样时间寄存器）

地址偏移：0x14

复位值：0x0000 0000

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

Bit	Name	Function
31:3	NA	保留位，未定义
2:0	SMP[2:0]	采样时间选择 这些位由软件写入，用来选择应用于所有通道的采样时间。 000：1.5 个 ADC 时钟周期 001：7.5 个 ADC 时钟周期 010：13.5 个 ADC 时钟周期 011：28.5 个 ADC 时钟周期 100：41.5 个 ADC 时钟周期 101：55.5 个 ADC 时钟周期 110：71.5 个 ADC 时钟周期 111：239.5 个 ADC 时钟周期 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。

13.12.7. ADC_TR（ADC 看门狗阈值寄存器）

地址偏移：0x20

复位值：0x0FFF 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—				HT[11:8]			
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW
23:16	HT[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	—				LT[11:8]			
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW
7:0	LT[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:28	NA	保留位，未定义
27:16	HT[11:0]	模拟看门狗高阈值 这些位由软件写入，用来定义模拟看门狗的高阈值。参考章节 12.8：模拟窗口看门狗（AWDEN，AWDSGL，AWDCH，AWD_HTR/AWD_LTR，AWD）。 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。
15:12	NA	保留位，未定义
11:0	LT[11:0]	模拟看门狗低阈值 这些位由软件写入，用来定义模拟看门狗的低阈值。参考章节 12.8：模拟窗口看门狗（AWDEN，AWDSGL，AWDCH，AWD_HTR/AWD_LTR，AWD）。 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。

13.12.8. ADC_CHSELR（ADC 通道选择寄存器）

地址偏移：0x28

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—				CHSEL19	CHSEL18	CHSEL17	CHSEL16
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW
15:8	CHSEL15	CHSEL14	CHSEL13	CHSEL12	CHSEL11	CHSEL10	CHSEL9	CHSEL8
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CHSEL7	CHSEL6	CHSEL5	CHSEL4	CHSEL3	CHSEL2	CHSEL1	CHSEL0
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:20	NA	保留位，未定义
19:0	CHSELx	通道 x 选择位 这些位由软件写入，用来定义所要转换的转换序列的通道。 0：输入通道 x 不被选为转换通道 1：输入通道 x 被选为转换通道 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。

13.12.9. ADC_DR（ADC 数据寄存器）

地址偏移：0x40
复位值：0x0000 0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DATA[15:0]	转换数据 这些位是只读的。其包含最后转换通道的转换结果。数据是左还是右对齐格式如图 12.12：数据对齐和分辨率所示。 仅在校准完成的时候，DATA[6:0]才包含校准系数。

13.12.10. ADC_CCR（ADC 通用配置寄存器）

地址偏移：0x308
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	TSEN	INTVREFEN	—					

类型	RW	RW	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:24	NA	保留位，未定义
23	TSEN	温度传感器使能 该位由软件置位和清零，用来使能/禁用温度传感器。 0：温度传感器禁用 1：温度传感器使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
22	INTVREFEN	V_{REFINT} 使能 该位由软件置位和清零，用来使能/禁用 V_{REFINT}。 0：V_{REFINT} 禁用 1：V_{REFINT} 使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
21:0	NA	保留位，未定义

13.12.11. ADC_CR2（ADC 控制寄存器 2）

地址偏移：0x30C

复位值：0x0000 0010

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						IOSH_SMPEN	IOSH_AMPEN
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	—			VREF_DECIB	VREFSEL[1:0]		VREFEN	—
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RO-0

Bit	Name	Function
31:10	NA	保留位，未定义
9	IOSH_SMPEN	IO 采样保持电路采样使能位 该位由软件置位和清零，用来使能/禁用 IO 采样保持电路采样功能。 0：IO 采样保持电路采样功能禁用

		1：IO 采样保持电路采样功能使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
8	IOSH_AMPEN	IO 采样保持电路保持使能位 该位由软件置位和清零，用来使能/禁用 IO 采样保持电路保持功能。 0：IO 采样保持电路保持功能禁用 1：IO 采样保持电路保持功能使能 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
7:5	NA	保留位，未定义
4	VREF_DECIB	ADC 参考电压电路 Bias 电流减半控制位 该位由软件置位和清零，用来选择是否减半 ADC 参考电压电路 Bias 电流。可使 ADC 参考电压电路的输出电压更精准。 0：标准 ADC 参考电压电路 Bias 电流 1：减半 ADC 参考电压电路 Bias 电流 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
3:2	VREFSEL[1:0]	ADC 参考电压电路输出电压选择位 这些位由软件置位和清零，用来选择 ADC 参考电压电路的输出电压电平。 00：保留 01：保留 10：2.5 V 11：保留 注：仅在 ADSTART=0（确保没有转换正在进行）时可写这些位。
1	VREFEN	ADC 参考电压电路使能位 该位由软件置位和清零，用来使能 ADC 参考电压电路输出。 注：仅在 ADSTART=0（确保没有转换正在进行）时可写该位。
0	NA	保留位，未定义

14. 比较器

仅 FT32F072 系列芯片拥有此模块。

14.1. 比较器说明

芯片内嵌两个通用比较器 COMP1 和 COMP2，可独立使用（适用所有终端上的 I/O 口），也可与定时器结合使用。它们可用于多种功能，包括：

- 由模拟信号触发的从低功耗模式唤醒
- 模拟信号调理
- 与 DAC 和定时器输出的 PWM 相结合，组成逐周期的电流控制回路

14.2. 比较器主要特性

- 每个比较器有可能选门限
  - 8 个 I/O 引脚
  - DAC 输出
- 输出端可以重定向到一个I/O端口或多个定时器输入端，可以触发以下事件：
  - 捕获事件
  - OCref_clr 事件
  - 输入刹车事件
- 两个比较器可以组合在一个窗口比较器中使用
- 每个比较器都可产生中断，并支持从 Sleep 和 Stop 模式唤醒 (通过 EXTI 控制器)。

14.3. 比较器的功能描述

14.3.1. 简介

比较器框图如下：

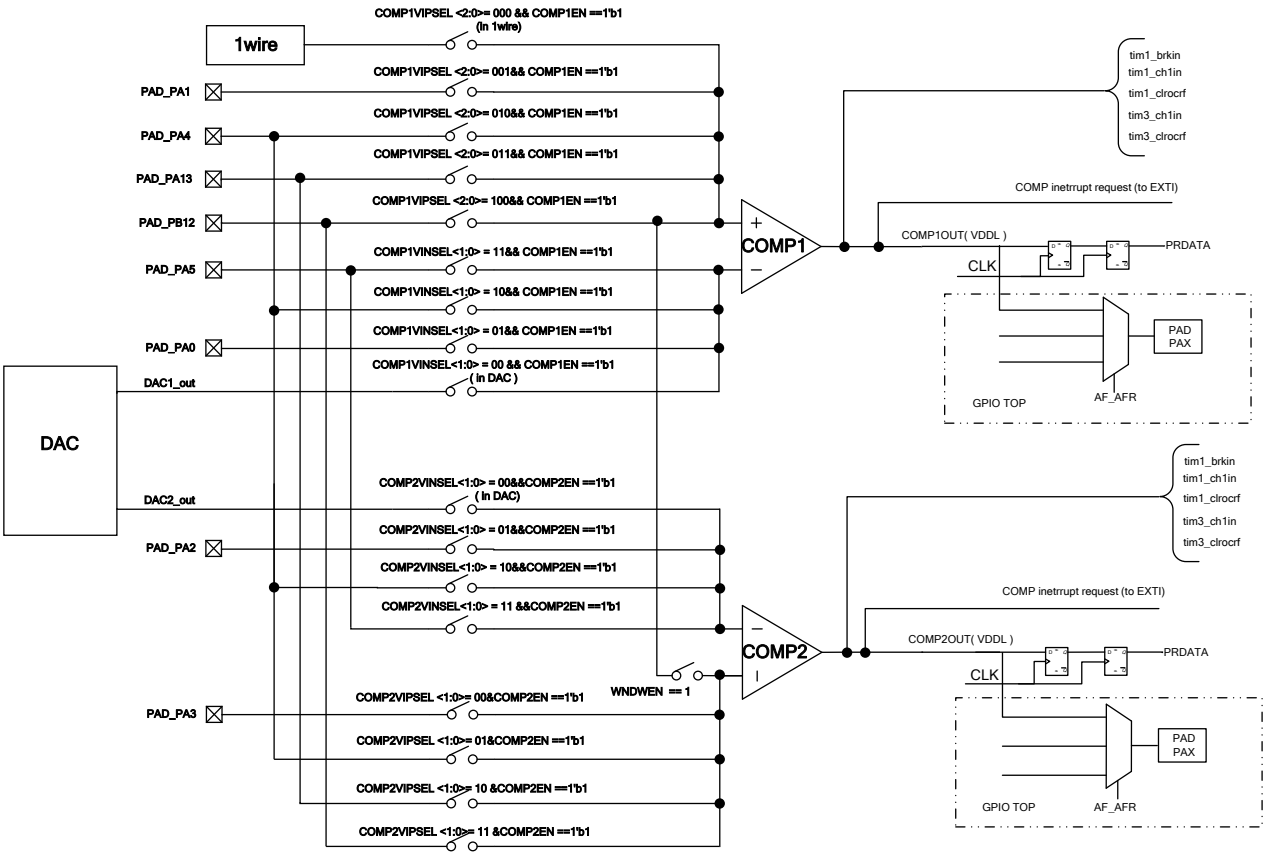


图 14.1 比较器框图

14.3.2. 比较器的输入口和内部信号

作为比较器的输入口必须为模拟口，通过 GPIO 的寄存器来配置所需要的 GPIO 为模拟口。
比较器的输出可以重定向到 IO 的端口，参考数据手册中的复用关系表。
同时比较器的输出也可以作为 timer 的输入用来实现以下功能：

- 捕获事件
- OCref_clr 事件
- 输入刹车事件

14.3.3. 比较器复位和时钟

COMP 时钟控制器提供的时钟与 PCLK 同步（APB 时钟）。
RCC 控制器中没有单独的比较器时钟使能控制位，比较器的时钟和复位使能控制与 SYSCFG 模块共用。
注：极性选择逻辑与输出端口的重定向工作独立于 PCLK 时钟，这使得比较器可以在停止模式下工作。

14.3.4. 比较器锁定方法

比较器能够用来做安全保护比如过流或者高温。当应用到这些情况的时候，必须要保证比较器的配置不会在过流或者高温导致程序跑飞的时候被误改写。所以为实现此目的，比较的寄存器被锁定之后就只可读不可写。锁定机制是当比较器的配置完成之后如果对锁定位置 1 那么比较器的寄存器就变为只可读不可写，直到下一次的 MCU 复位到来。

14.3.5. 比较器中断

比较器的输出会连接到 EXTI 模块，每个比较器有自己的中断线，会产生相应的中断或事件。也可以用作低功耗的唤醒。

14.3.6. DAC 输出电压

DAC 的输出是连接到比较器的模拟输入，作为比较器的输入电压。输入被线性的转换为模拟输出电压，转换公式如下：

DAC1_out=V_{DACREF}* (DAC1DATA<6:0>+1)/128　(0x20 ≤ DAC1DATA<6:0> ≤ 0x7F)

DAC2_out=V_{DACREF} * (DAC2DATA<6:0>)/128　(0x00 ≤ DAC2DATA<6:0> ≤ 0x5F)

14.4. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1C	COMPCSR	COMP2LOCK	COMP2OUT	–	–	COMP2POL	COMP2OUTSEL			WNDWEN	–	–	COMP2VINSEL		COMP2VIPSEL		COMP2EN	COMP1LOCK	COMP1OUT	–	–	COMP1POL	COMP1OUTSEL			–	–	COMP1VINSEL		COMP1VIPSEL			COMP1EN
	Reset	0	0	x	x	0	0	0	0	0	x	x	0	1	0	0	0	0	0	x	x	0	0	0	0	x	x	0	1	0	0	1	0
0x20	DACCTRL	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	DACREFSEL		DACEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0		0
0x24	DAC1DATA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	DAC1DATA						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0
0x28	DAC2DATA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	DAC2DATA						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0

14.4.1. COMPCSR

比较器控制寄存器：COMPCSR

地址：0x01C

Default = 0x00080012

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	COMP2LOCK	COMP2OUT	—		COMP2POL	COMP2OUTSEL		
类型	RW	RO	RO-0	RO-0	RW	RW	RW	RW
23:16	WNDWEN	—		COMP2VINSEL		COMP2VIPSEL		COMP2EN
类型	RW	RO-0	RO-0	RW	RW	RW	RW	RW
15:8	COMP1LOCK	COMP1OUT	—		COMP1POL	COMP1OUTSEL		—
类型	RW	RO	RO-0	RO-0	RO-0	RW	RW	RO-0
7:0	—		COMP1VINSEL		COMP1VIPSEL			COMP1EN
类型	RO-0	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31	COMP2LOCK	P 比较器配置寄存器锁定位（一旦写入只能被系统复位清除） 0 ：表示 COMPCSR[31:16]可读可写。 1 ：表示 COMPCSR[31:16]只可读。
30	COMP2OUT	P 比较器的结果输出 0 ：表示输出为低 1 ：表示输出为高
29:28	NA	保留位
27	COMP2POL	输出极性选择 0 ： 表示 P 比较器输出不取反。 1 ： 表示 P 比较器输出取反。
26:24	COMP2OUTSEL	P 比较器的输出选择，选择输出结果到哪里 000 ：no selection 001 ：timer1 break input 010 ：timer1 input capture 1 011 ：timer1 OCrefclear input 100 ：no selection 101 ：no selection 110 ：timer3 input capture 1 111 ：timer3 OCrefclear input
23	WNDWEN	比较器 window mode 选择，窗口比较模式受 NCMP 的使能控制 0 : disable 1 : enable

22:21	NA	保留位
20:19	COMP2VINSEL	P 比较器的负端选择器（复位值：01） 00 ：选择 DAC2_OUT 01 ：选择 PAD PA2 10 ：选择 PAD PA4 11 ：选择 PAD PA5
18:17	COMP2VIPSEL	P 比较器的正端选择器,当窗口比较器开启时优先选择窗口模式 00 ：选择 PAD PA3 01 ：选择 PAD PA4 10 ：选择 PAD PA13 11 ：选择 PAD PB12
16	COMP2EN	P 比较器的使能 0 : disable 1 : enable
15	COMP1LOCK	N 比较器配置寄存器锁定位（一旦写入只能被系统复位清除） 0 ：表示 COMPCSR[15:0]可读可写。 1 ：表示 COMPCSR[15:0]只可读。
14	COMP1OUT	N 比较器的结果输出 0 ：表示输出为低 1 ：表示输出为高
13:12	NA	保留位
11	COMP1POL	输出极性选择 0 ： 表示 N 比较器输出不取反。 1 ： 表示 N 比较器输出取反
10:8	COMP1OUTSEL	N 比较器的输出选择，选择输出结果到哪里 000 ：no selection 010 ：timer1 input capture 1 011 ：timer1 OCrefclear input 100 ：no selection 101 ：no selection 110 ：timer3 input capture 1 111 ：timer3 OCrefclear input
7:6	NA	保留位
5:4	COMP1VINSEL	N 比较器的负端选择器（复位值：01） 00 ：选择 DAC1_OUT 01 ：选择 PAD PA0 10 ：选择 PAD PA4 11 ：选择 PAD PA5
3:1	COMP1VIPSEL	N 比较器的正端选择器（复位值：001） 000 ：选择 1wire 001 ：选择 PAD PA1 010 ：选择 PAD PA4 011 ：选择 PAD PA13

		100 ：选择 PAD PB12
0	COMP1EN	N 比较器的使能 1：enable 0：disable

14.4.2. DACCTRL

DAC 的相关寄存器

地址：0x020

Default : 0x00000000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15：8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					DACREFSEL		DACEN
类型	RO-0	RO-0	RO-0	RO-0	RO-0	WR		WR

Bit	Name	Function
31:3	NA	保留位
2:1	DACREFSEL	DAC 的参考电压 V_{DACREF} 选择 11 ： VDDA 10 ： 4V 电压 01 ： 3V 电压 00 ： 2V 电压
0	DACEN	DAC 的工作使能 1：enable 0：disable

14.4.3. DAC1DATA

DAC1 的 7bit 数字输入数据

地址：0x24

Default ：0x00000000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15：8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	DAC1DATA						
类型	RO-0	WR						

Bit	Name	Function
31:7	NA	保留位
6:0	DAC1DATA	DAC1 的 7bit 数字输入数据

14.4.4. DAC2DATA

DAC2 的 7bit 数字输入数据

地址：0x28

Default ：0x00000000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15：8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	DAC2DATA						
类型	RO-0	WR						

Bit	Name	Function
31:7	NA	保留位
6:0	DAC2DATA	DAC2 的 7bit 数字输入数据

15. 运算放大器

芯片内部集成了一个运算放大器，用于对模拟信号进行处理。仅 FT32F072 系列芯片拥有此模块。

15.1. 运放 0 功能描述

运放 0 具有以下特点：

1. 输入失调电压可校准
2. 2.49MHz 的单位增益带宽
3. 外部直接或者串 4kΩ 电阻到反相端
4. 输出可选 40kΩ~320kΩ 电阻反馈到反相端
5. 输出可连接外部管脚

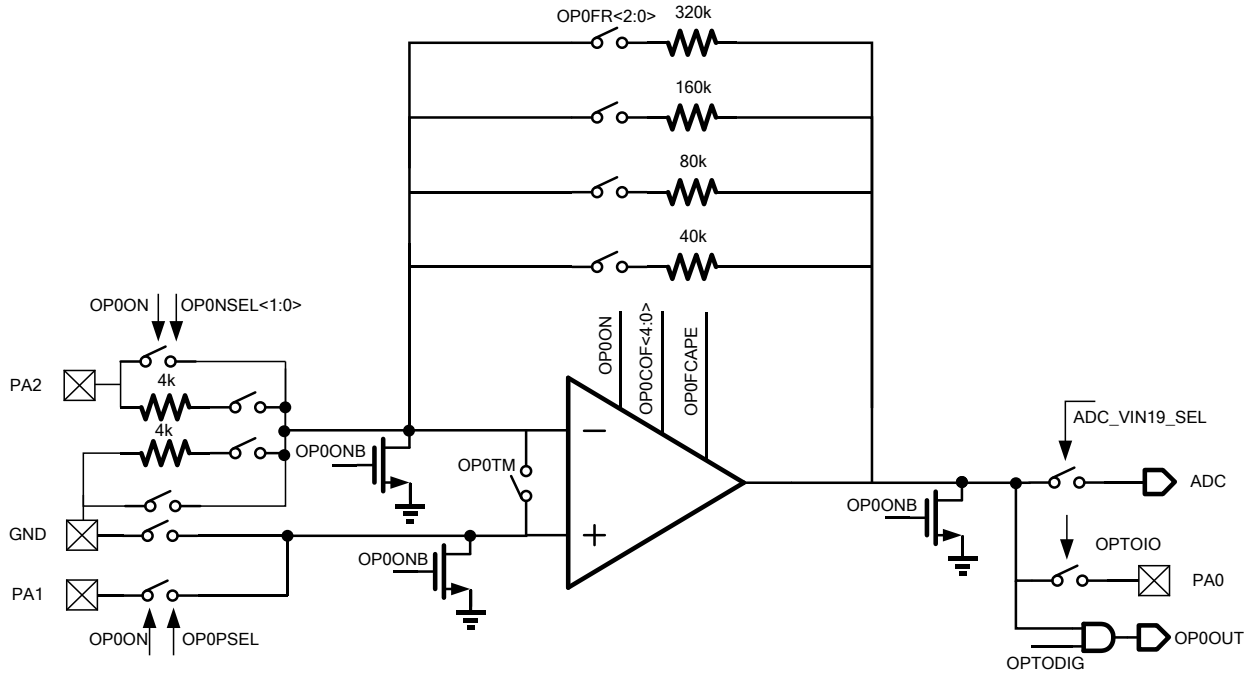


图 15.1 运放结构框图

15.1.1. 校准输入失调电压

运放的输入失调电压可以用软件校准，校准的步骤如下

1. 把 OP0ON 置 1，打开运放；
2. 把 OP0TM 置 1，运放进入输入失调校准模式；
3. 把 OP0NSEL<1:0>置 2’b00，反相端直接接地；
4. 把 OP0PSEL 置 1，正相端直接接地；

5. 把 OP0FCAPE 设置为 0；
6. 把 OPTODIG 置 1，使运放输出到寄存器；
7. 把 OP0FR<2:0>设置为 3'b000，运放没有反馈网络；
8. 把 PA2 悬空
9. 把 OP0COF<4:0>设置为 0x10，延时最少 300μs；
10. 设置 OP0COF 等于 OP0COF+1，延时最少 300μs；
11. 判断 OP0OUT 是否翻转，如果是，则保存当前的 OP0COF 值，记为 A，修改 OP0COF 值为 0x0F，延时最少 300μs，执行步骤 13；否则执行步骤 12；
12. 若 OP0COF 等于 0x0F，则退出校准流程，校准失败，否则执行步骤 10；
13. 设置 OP0COF 等于 OP0COF-1，延时最少 300μs；
14. 判断 OP0OUT 是否翻转，如果是，则保存当前的 OP0COF 值，记为 B，修改 OP0COF 值为(A+B)/2 的取整值，校准流程结束，校准成功；否则执行步骤 15；
15. 若 OP0COF 等于 0x10，则退出校准流程，校准失败；否则执行步骤 13。

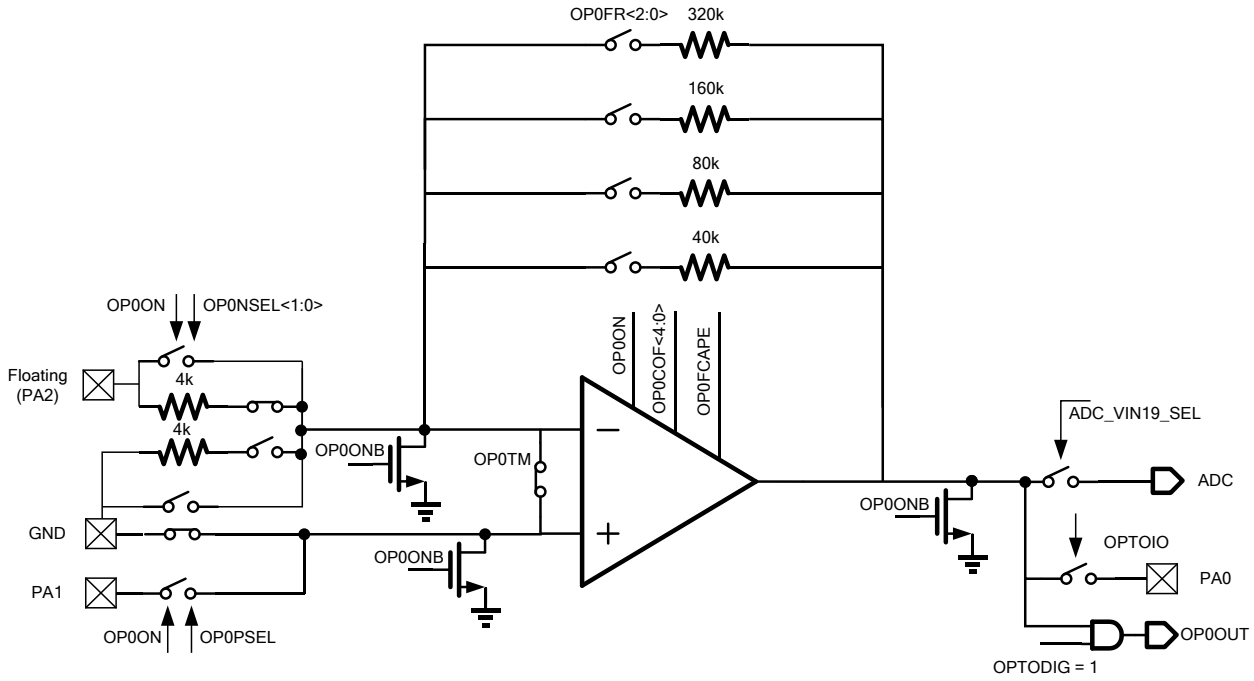


图 15.2 运放校准模式下的连接

15.2. 运算放大器寄存器映射

offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30	OP_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OP0OUT	OP0PSEL	OP0NSEL [1:0]		OP0FR[2:0]			OP0FCAPE	OP0TM	OPTODIG	OPTOIO	OP0COF[4:0]					OP0ON
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0

注：

1.运算放大器寄存器与系统配置模块共用一个基地址，同时要操作运算放大器寄存器，必须先打开系统配置模式时钟

2.软件复位系统配置模块也会复位运算放大器模块

15.2.1. OP_CR

偏移地址：0x00

复位值：0x0000 D800

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							OP0OUT
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO
15:8	OP0PSEL	OP0NSEL		OP0FR			OP0FCAPE	OP0TM
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OPTODIG	OPTOIO	OP0COF					OP0ON
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:17	NA	保留位，未定义
16	OP0OUT	运放 0 输出
15	OP0PSEL	运放 0 正相端输入选择： 0：直接连接到 PA1 1：直接连接到 GND
14:13	OP0NSEL	运放 0 反相端输入选择： 2’b00：直接连接到 GND; 2’b01：直接连接到 PA2; 2’b10：串联 4K 电阻连接到 PA2; 2’b11：串联 4K 电阻连接到 GND;
12:10	OP0FR	运放 0 反馈电阻选择： 3’b0xx：没有反馈网络 3’b100：反馈电阻为 40k; 3’b101：反馈电阻为 80k; 3’b110：反馈电阻为 160k;

		3'b111：反馈电阻为 320k;
9	OP0FCAPE	运放 0 补偿电容使能位 1：禁止（运放用做比较器） 0：使能
8	OP0TM	运放 0 输入失调校准模式 1：运放 0 进入失调校准模式 0：运放 0 进入正常模式
7	OPTODIG	运放 0 输出到寄存器控制位 1：使能 0：禁止
6	OPTOIO	运放 0 输出到 PA0 使能位 1：使能 0：禁止
5:1	OP0COF	运放 0 输入失调校准
0	OP0ON	运放 0 使能位 1：使能 0：禁止

16. 高级控制定时器（TIM1）

16.1. TIM1 简介

高级控制定时器（TIM1）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。
它适合多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM 等）。
使用定时器预分频器和 RCC 时钟控制预分频器 ,可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。
高级控制定时器（TIM1）和通用定时器（TIMx）是完全独立的，它们不共享任何资源。它们可以同步操作，具体描述参考 16.3.20 章节的内容。

16.2. TIM1 主要特性

TIM1 定时器的功能包括：

- 16 位向上、向下、向上/向下自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新事件：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 支持用于定位的增量编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

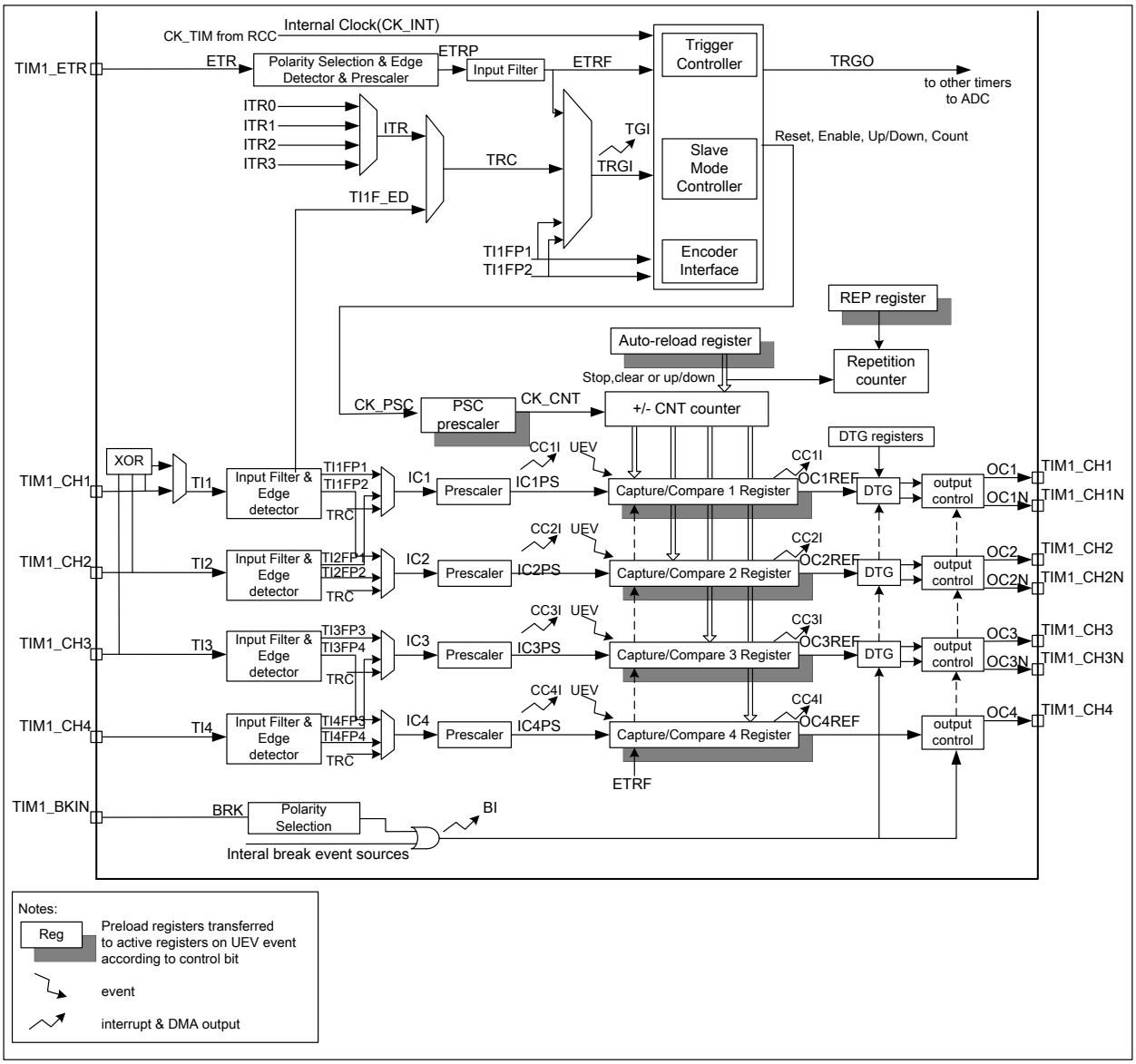


图 16.1 高级控制定时器框图

16.3. TIM1 功能描述

16.3.1. 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。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

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

- 重复计数寄存器（TIMx_RCR）

自动装载寄存器是预先装载的，写或读自动装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容会被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效。（更多有关使能计数器的细节，请参考控制器的从模式描述）

注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

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。新的预分频器的参数在下一次更新事件到来时被采用。

图 16.2 和图 16.3 给出了在运行时更改预分频器因子，计数器的相关动作的例子。

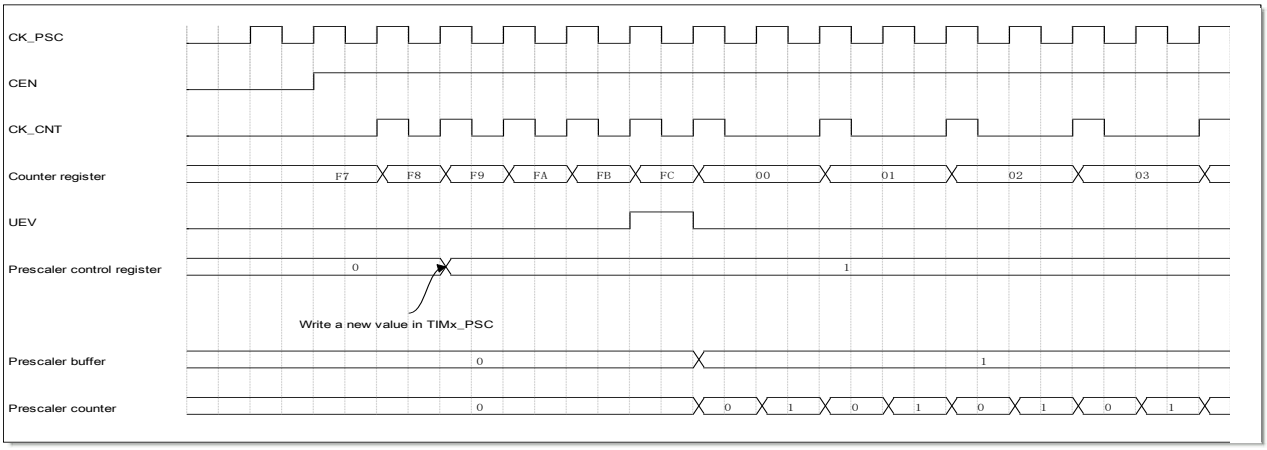


图 16.2 预分频系数从 1 变到 2 时，计数器的时序图

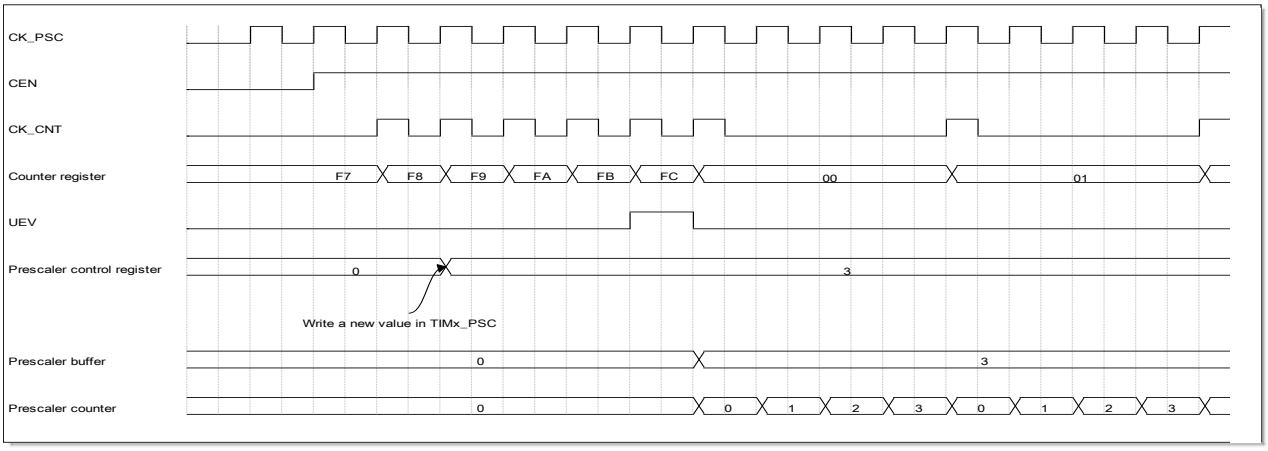


图 16.3 预分频系数从 1 变到 4 时，计数器的时序图

16.3.2. 计数器模式

向上计数模式

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

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数（TIMx_RCR）时，才产生更新事件（UEV）；否则每次计数器溢出都会产生更新事件。

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。通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 重复计数器被重新加载为 TIMx_RCR 寄存器中的内容。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

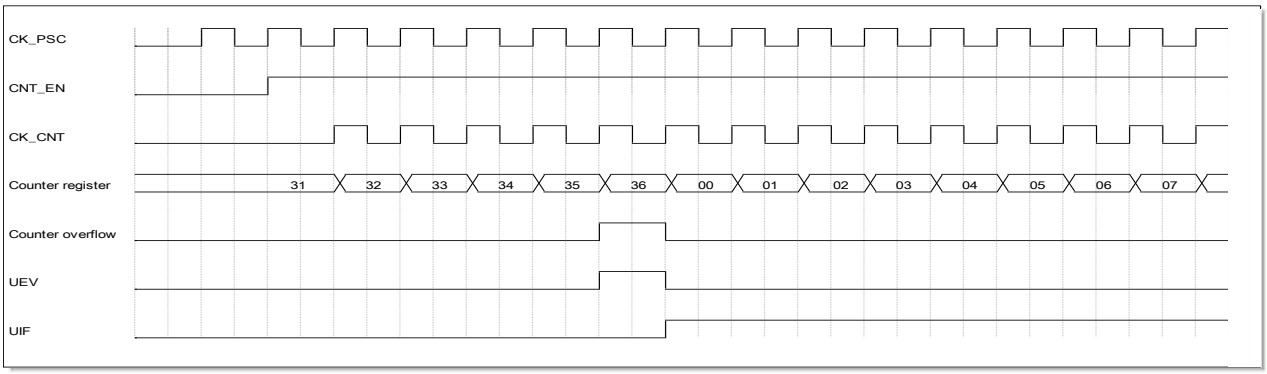


图 16.4 计数器时序图，内部时钟分频因子为 1

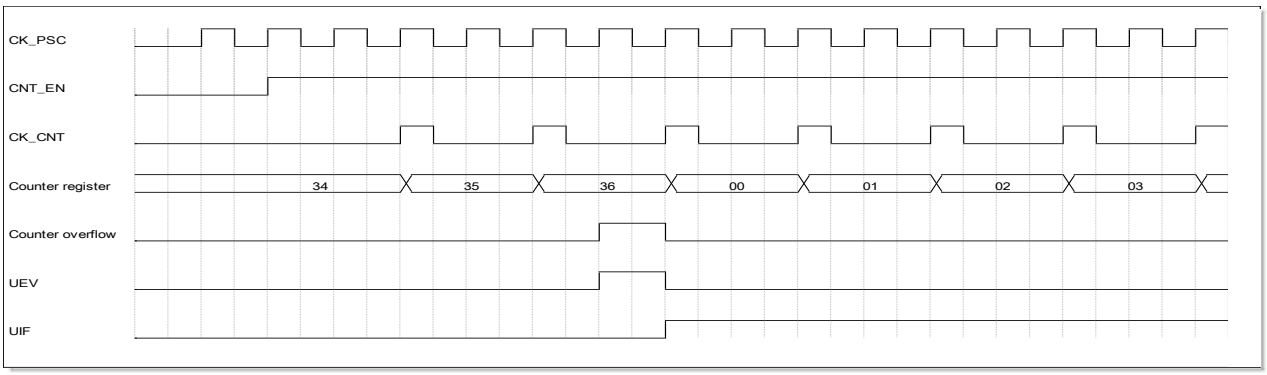


图 16.5 计数器时序图，内部时钟分频因子为 2

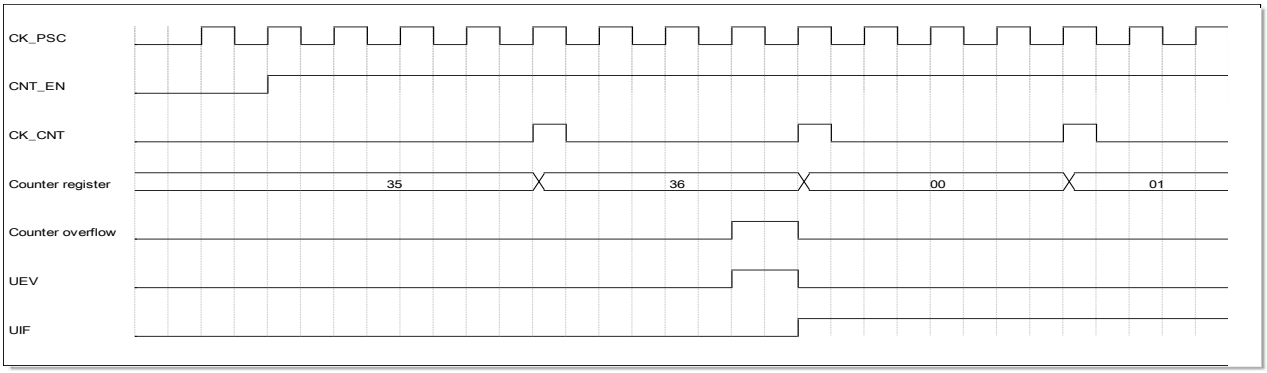


图 16.6 计时器时序图，内部时钟分频因子为 4

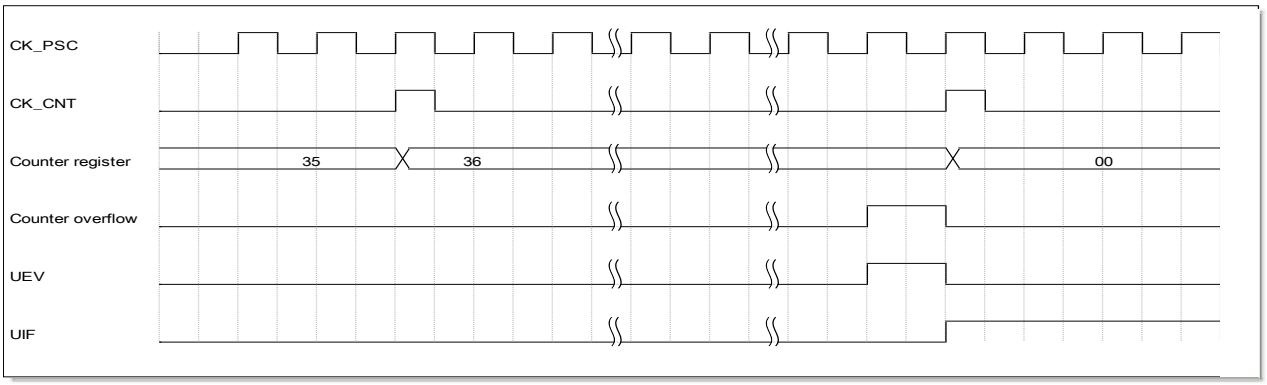


图 16.7 计数器时序图，内部时钟分频因子为 N

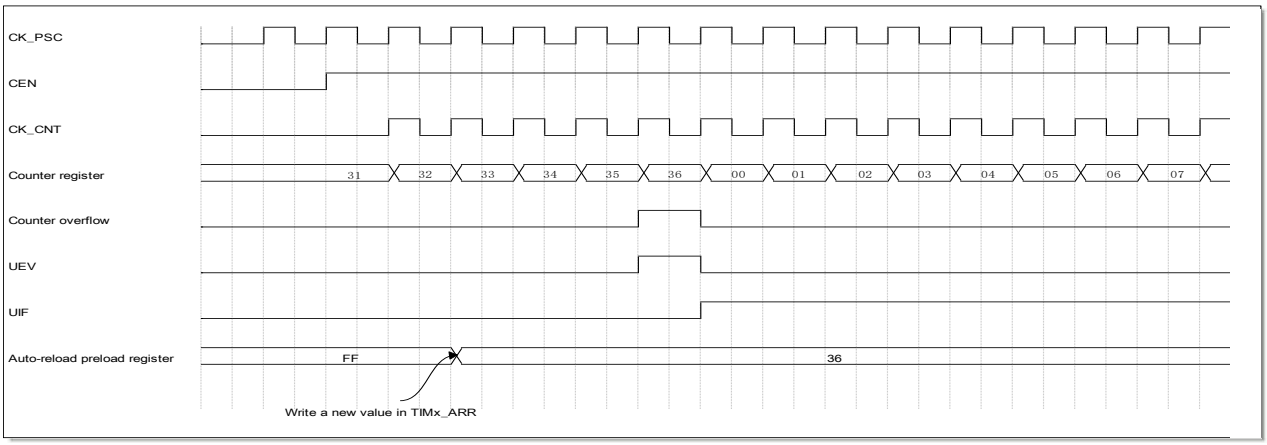


图 16.8 计数器时序图，当 ARPE=0 时的更新事件

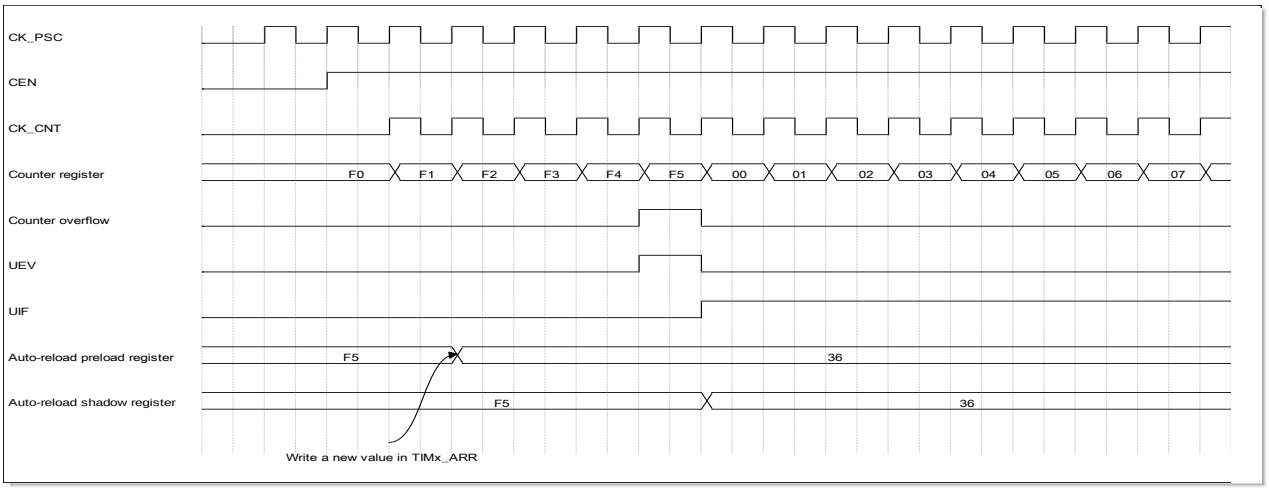


图 16.9 计数器时序图，当 ARPE=1 时的更新事件

向下计数模式

在向下计数模式中，计数器从自动装载值（TIMx_ARR 寄存器的值）开始向下计数到 0，然后又从自动装载值重新开始计数，并且产生一个计数器向下溢出事件。

如果使用了重复计数器，当向下计数器重复了重复计数寄存器（TIMx_RCR）中设定的次数后，才产生更新事件（UEV），否则每次计数器下溢都会产生更新事件。

设置 TIMx_EGR 寄存器的 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。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 重复计数器被重新加载为 TIMx_RCR 寄存器中的内容。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值；注意，如果在计数器重载之前自动装载寄存器被更新，此时在下一个周期才是预期的值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

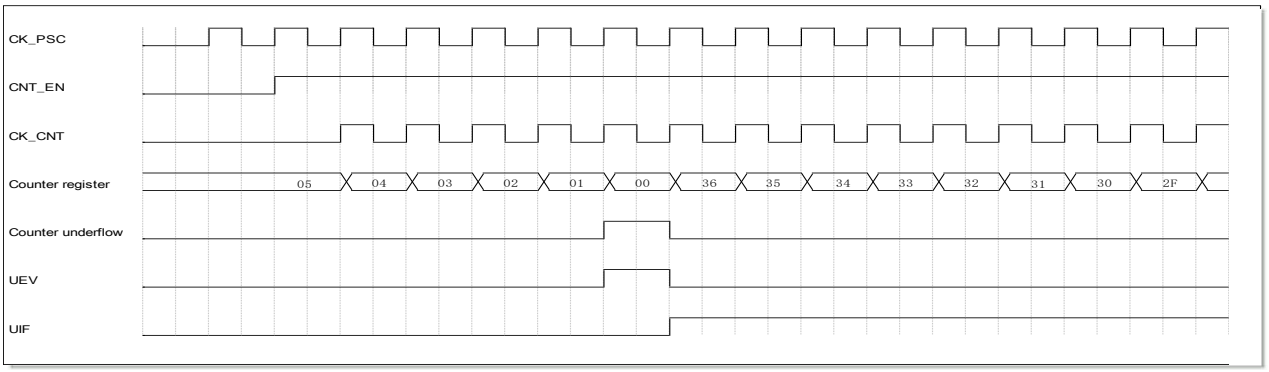


图 16.10 计数器时序图，内部时钟分频因子为 1

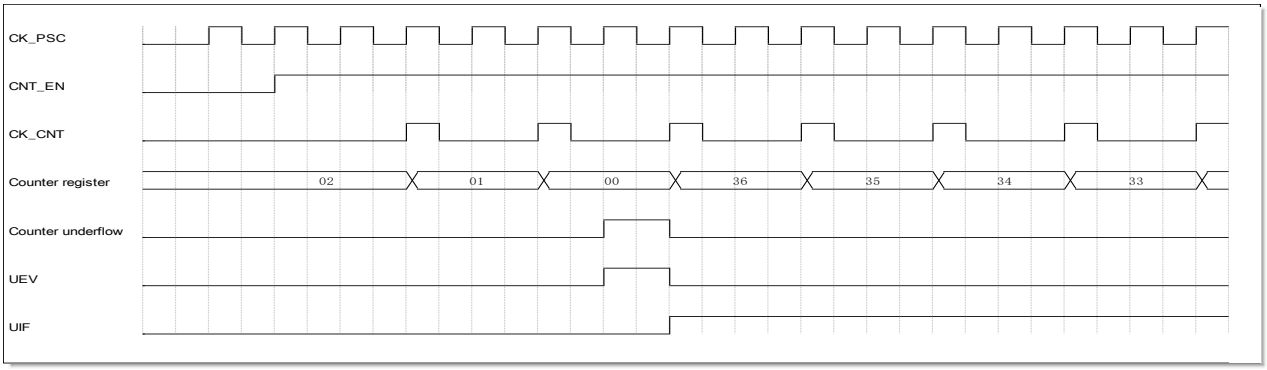


图 16.11 计数器时序图，内部时钟分频因子为 2

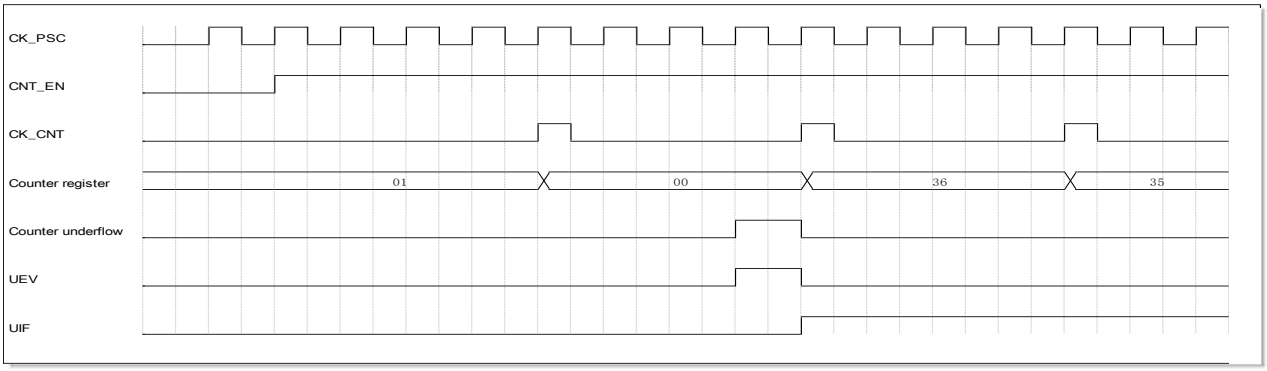


图 16.12 计数器时序图，内部时钟分频因子为 4

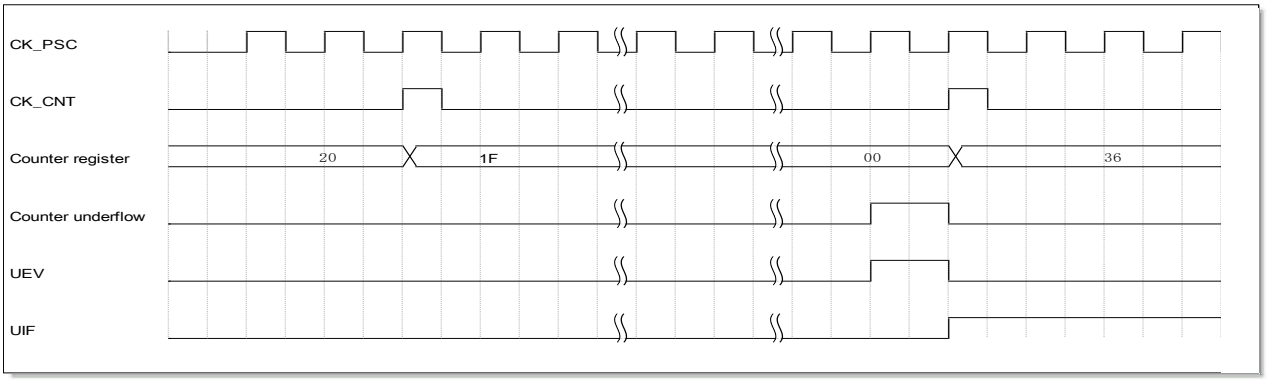


图 16.13 计数器时序图，内部时钟分频因子为 N

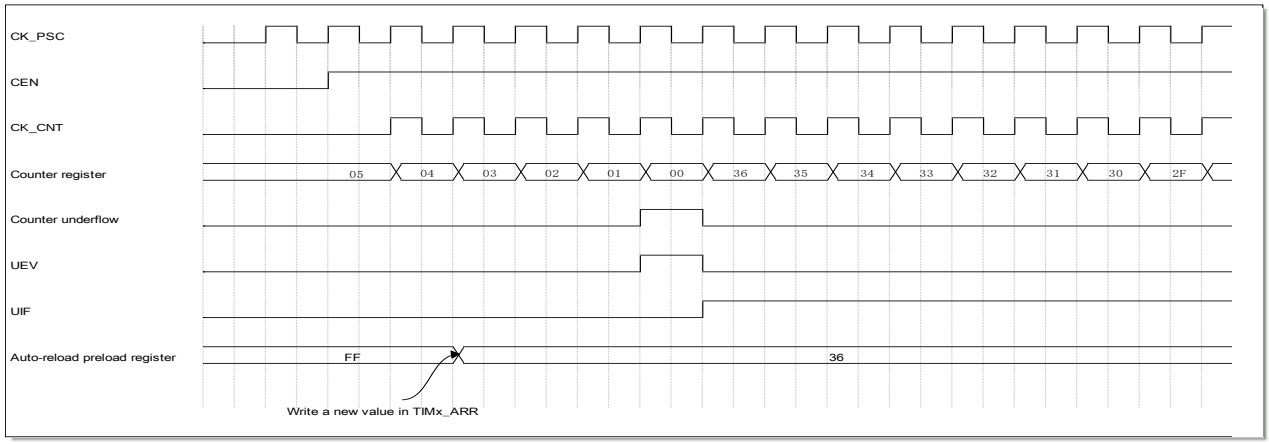


图 16.14 计数器时序图，没有使用重复计数器时的更新事件

中央对齐模式

在中央对齐模式下中，计数器从 0 开始计数到自动装载值（TIMx_ARR 寄存器）-1，产生一个计数器溢出事件，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。

当 TIMx_CR1 的 CMS 位不为 00 时，使能中央对齐模式。已配置为输出通道的输出比较中断标志在以下情况下被置位：计数器向下计数（中央对齐模式 1，CMS=01），计数器向上计数（中央对齐模式 2，CMS=10），计数器向下和向上计数（中央对齐模式 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。

在此模式下，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。此时，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。然而，计数器仍会根据当前自动装载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位（选择更新请求源）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 重复计数器被重新加载为 TIMx_RCR 寄存器中的内容。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值；注意，如果因为计数器上溢而产生更新，在计数器重载之前自动装载寄存器被更新，此时在下一个周期才是预期的值。

下面一些时序图展示了计数器在不同时钟频率下的计数情况。

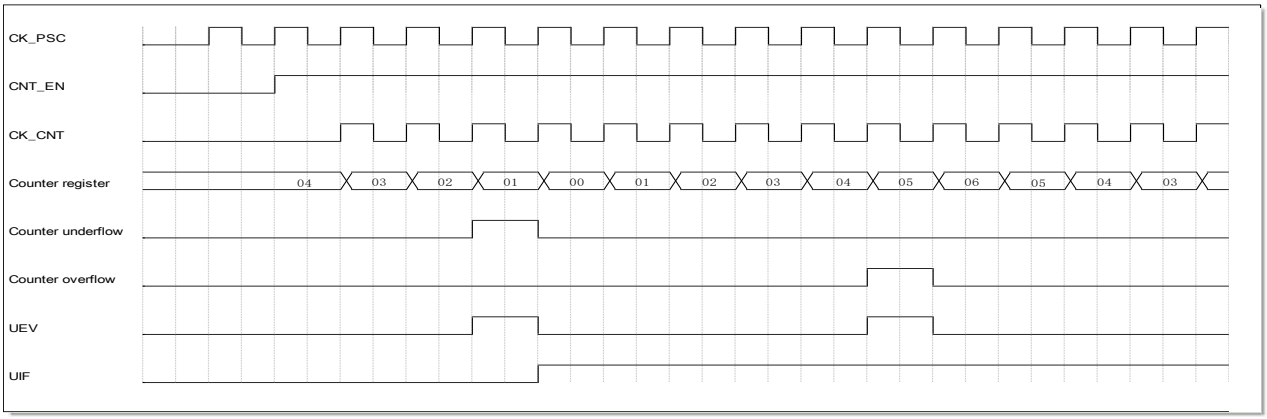


图 16.15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

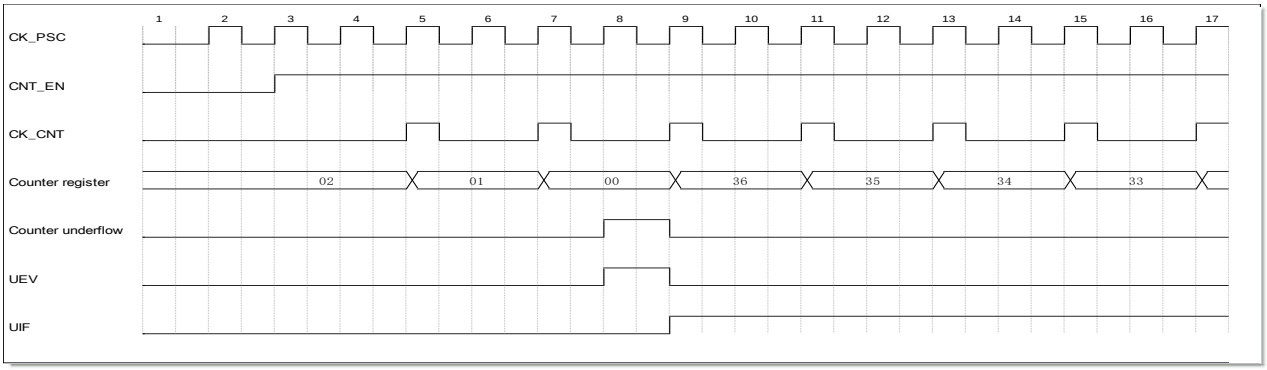


图 16.16 计数器时序图，内部时钟分频因子为 2

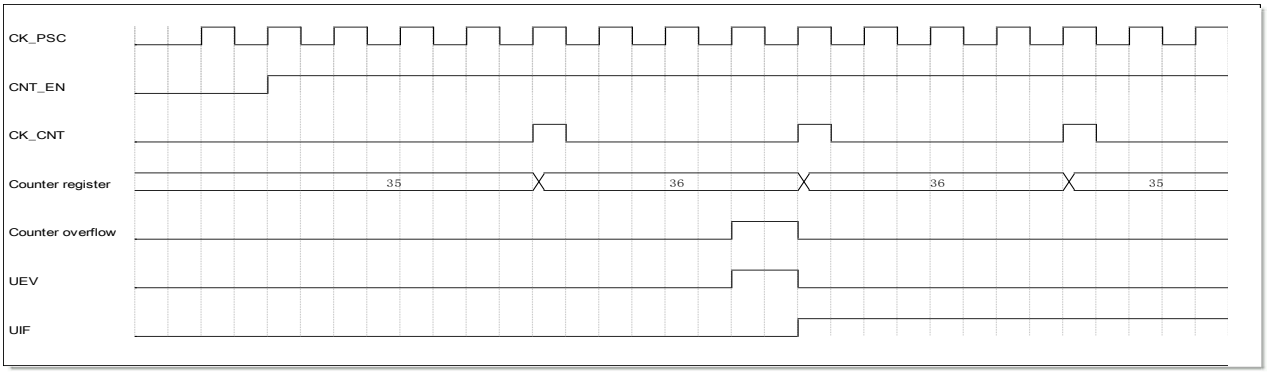


图 16.17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36

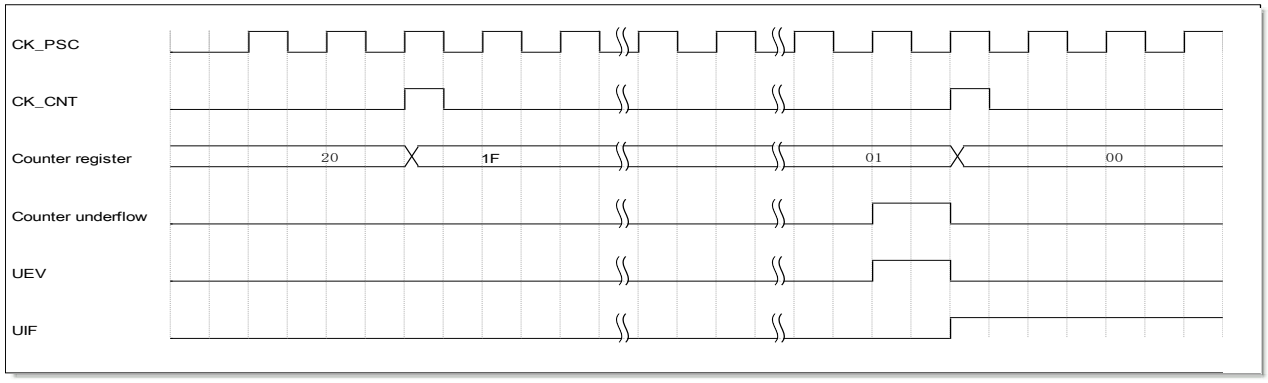


图 16.18 计数器时序图，内部时钟分频因子为 N

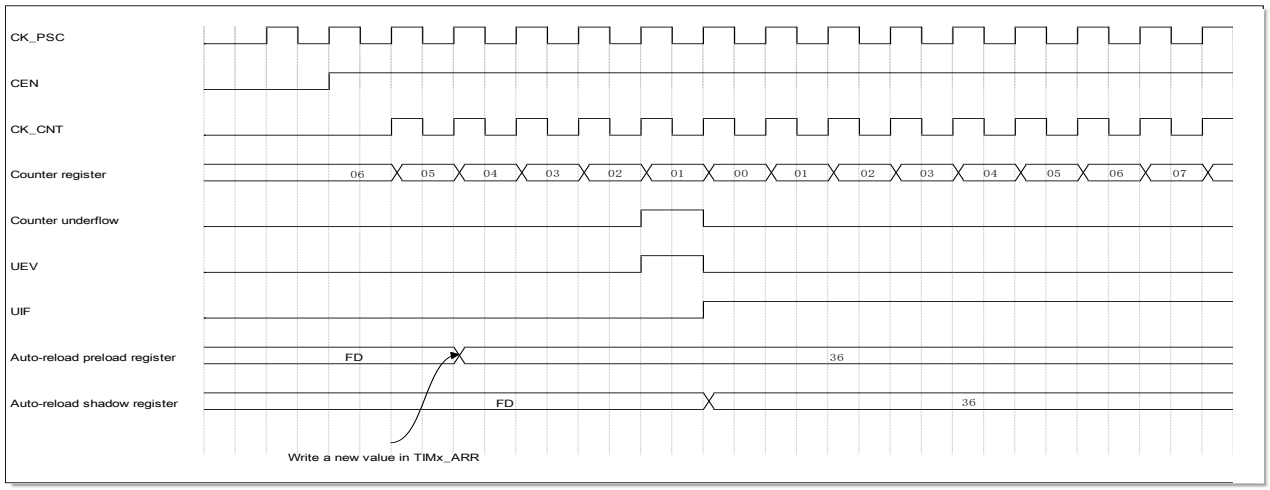


图 16.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

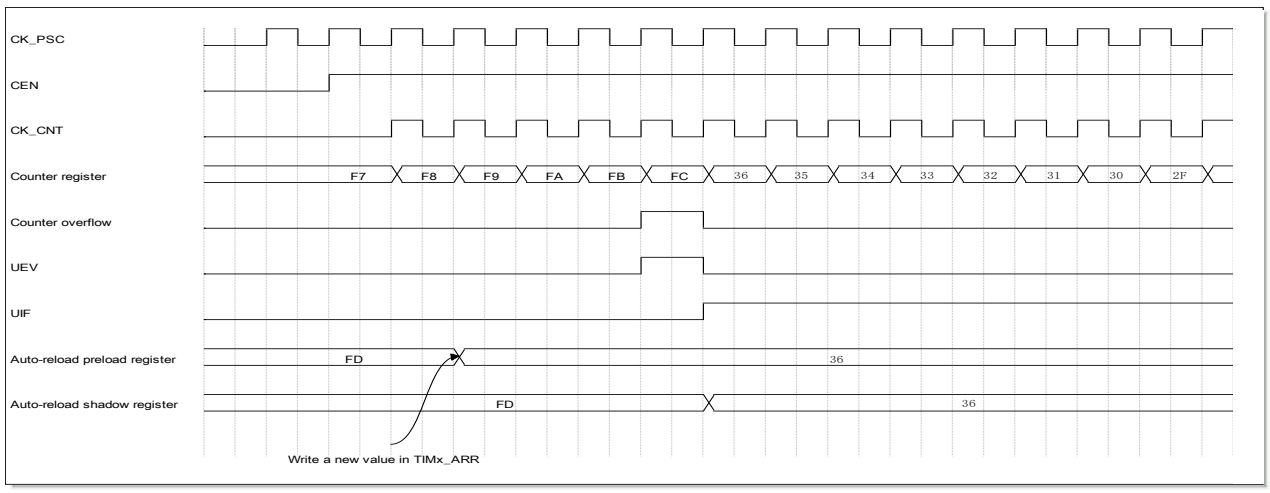


图 16.20 计数器时序图，ARPE=1 时的更新事件（计数器上溢）

16.3.3. 重复计数器

16.3.1 章节的时基单元解释了计数器上溢/下溢时是如何产生更新事件（UEV）的，然而事实上它只能在重复

计数器计数到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器（TIMx_ARR 自动重加载寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的 TIMx_CCRx 捕获/比较寄存器），N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时
- 向下计数模式下每次计数器下溢时
- 中央对齐模式下每次计数器的上溢和每次计数器的下溢。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 2 次更新占空比。在中央对齐模式下，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 2*Tck。

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

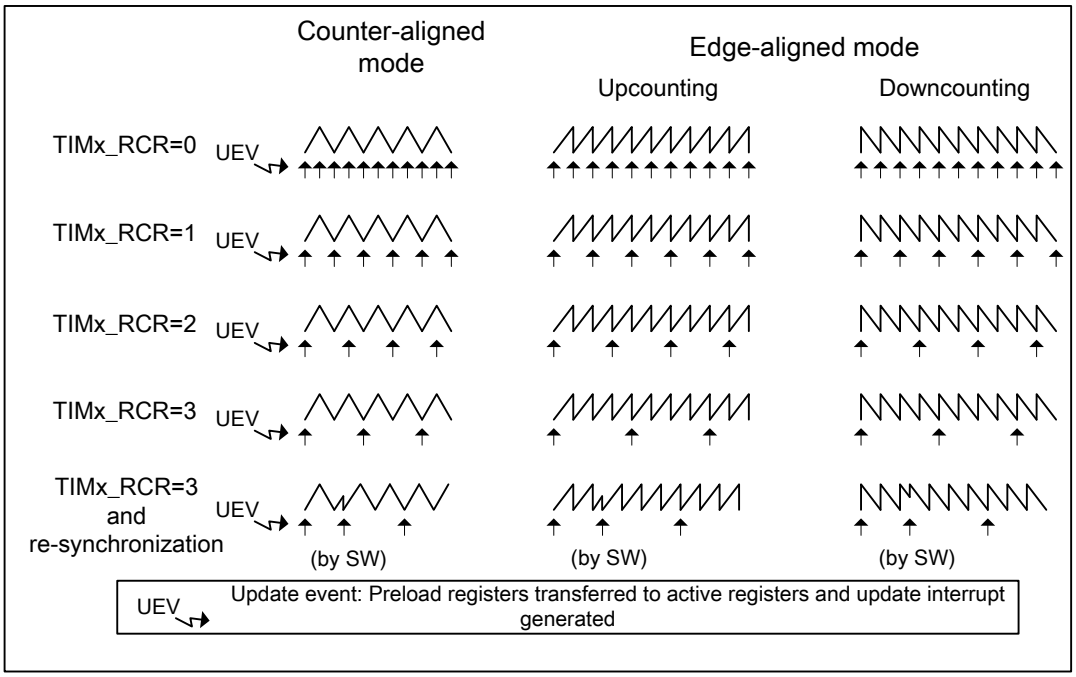


图 16.21 不同模式及不同 TIMx_RCR 寄存器设置下更新速率的例子

16.3.4. 时钟源

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置定时器 TIM1 作为定时器 TIM2 的预分频器。

内部时钟源（CK_INT）

如果禁止了从模式控制器（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）

是实际上的控制位，并且只能被软件修改（除非 UG 位自动被清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示了在不带预分频器时，控制电路和向上计数器在正常模式下的动作。

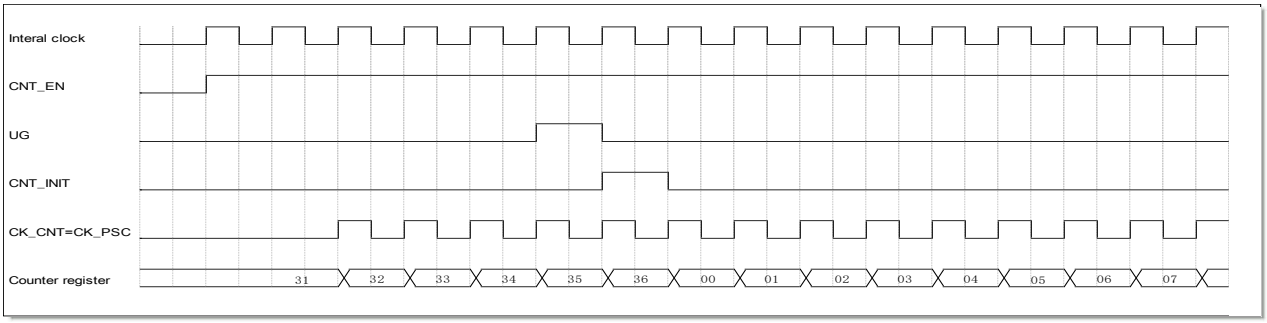


图 16.22 内部时钟分频是 1 时正常模式下的控制时序图

外部时钟源模式 1

当 TIMx_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器在选定输入端的每个上升沿或下降沿计数。

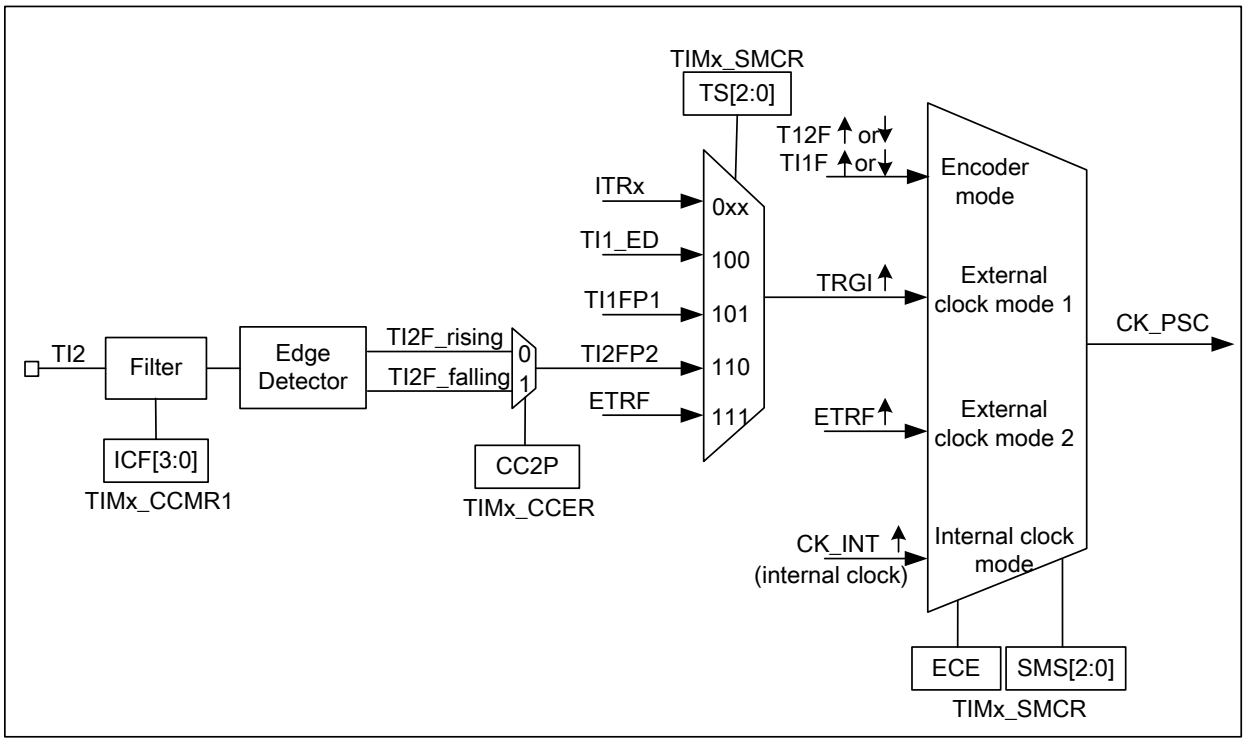


图 16.23 TI2 外部时钟连接示例

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 写 TIMx_CCMR1 寄存器的 CC2S=01，配置通道 2 检测 TI2 输入的上升沿。
2. 写 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）。
3. 写 TIMx_CCER 寄存器的 CC2P=0，选择上升沿极性。
4. 写 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1。
5. 写 TIMx_SMCR 寄存器的 TS=110，选择 TI2 作为触发输入源。
6. 写 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注意： 捕获预分频器不用作触发，所以不需要对它进行配置。

当 TI2 上升沿出现时，计数器计数一次且 TIF 标志被置位。

当 TI2 的上升沿和计数器实际时钟之间的延时取决于在 TI2 输入端的重新同步电路。

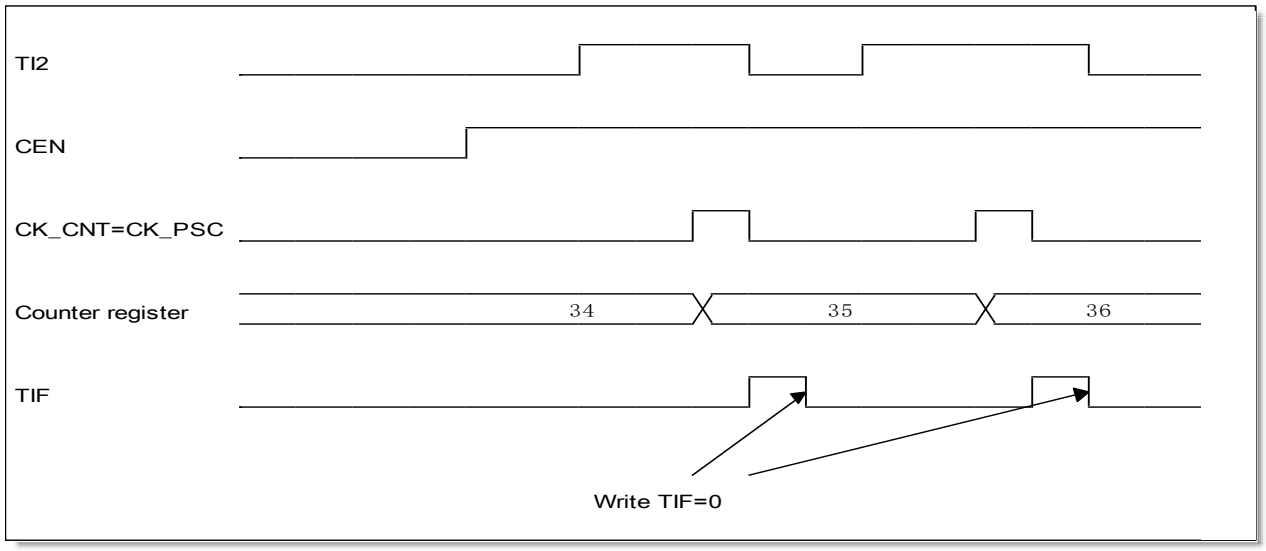


图 16.24 外部时钟模式 1 时的控制时序图

外部时钟源模式 2

当 TIMx_SMCR 寄存器的 ECE=1 时，此模式被选中。计数器在外部触发 ETR 的每个上升沿或下降沿计数。

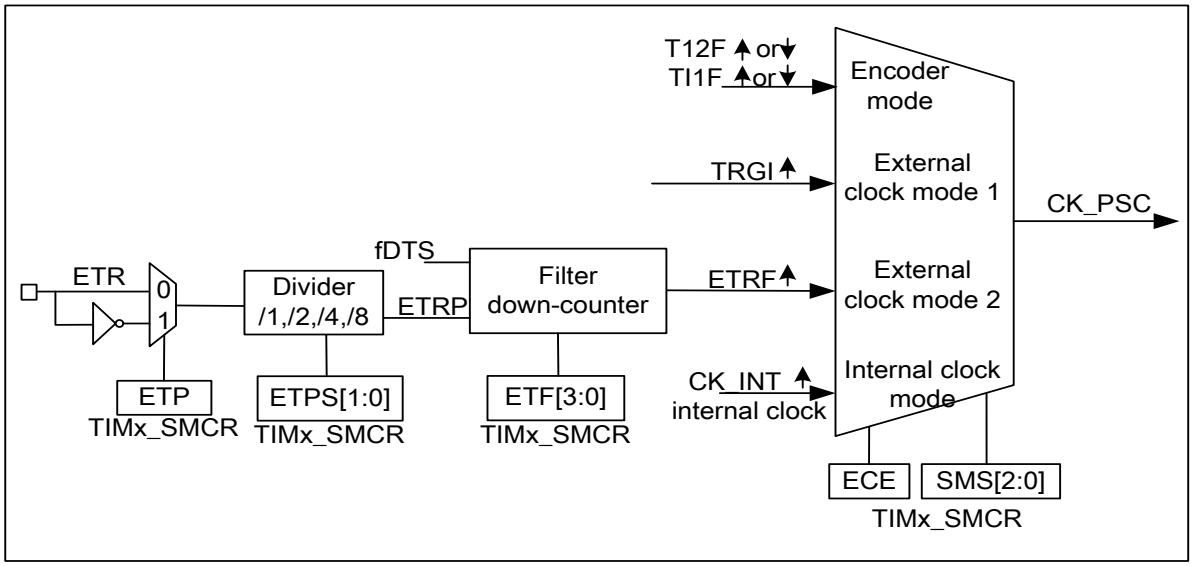


图 16.25 外部触发输入框图

例如，要配置向上计数器在 ETR 的每 2 个上升沿计数一次，使用下列步骤：

1. 本例中不需要滤波器，写 TIMx_SMCR 寄存器中的 ETF[3:0]=0000。
2. 写 TIMx_SMCR 寄存器中的 ETPS[1:0]=01，设置预分频器分频比为 2。
3. 写 TIMx_SMCR 寄存器中的 ETP=0，选择检测 ETR 的上升沿。

4. 写 TIMx_SMCR 寄存器中的 ECE=1，开启外部时钟模式 2。
5. 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。

计数器在每 2 个 ETR 上升沿计数一次。

ETR 的上升沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。

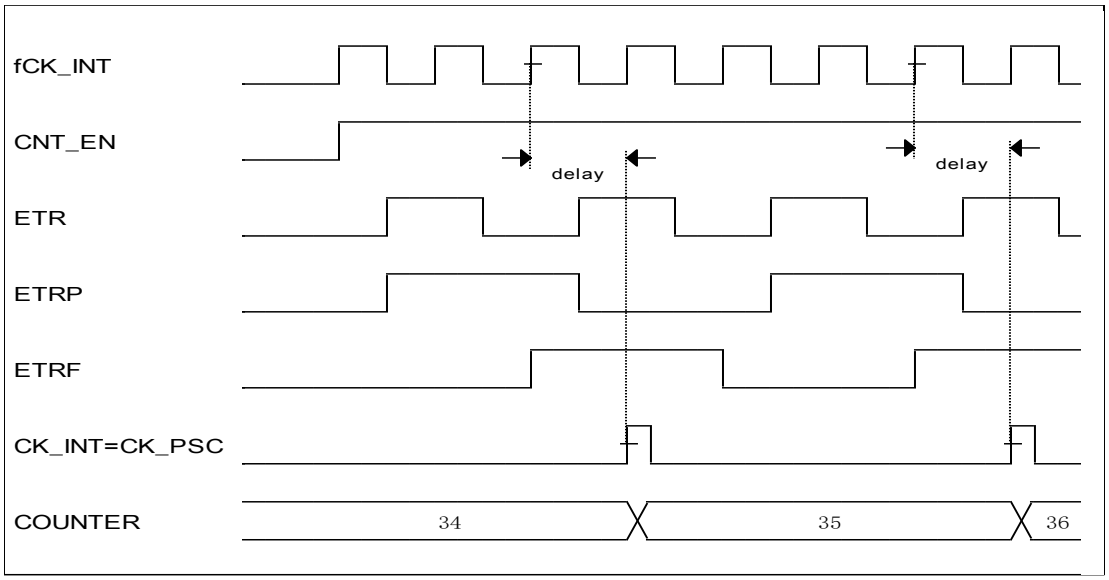


图 16.26 外部时钟模式 2 时的控制时序图

16.3.5. 捕获/比较通道

每一个捕获/比较通道都是包括一个捕获/比较寄存器（包含影子寄存器），一个捕获的输入部分（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）。

图 16.26 至图 16.29 是一个捕获/比较通道的概览。

输入部分采样相应的 TIx 输入信号，产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿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。该信号紧接着被预分频产生信号 IC1PS。

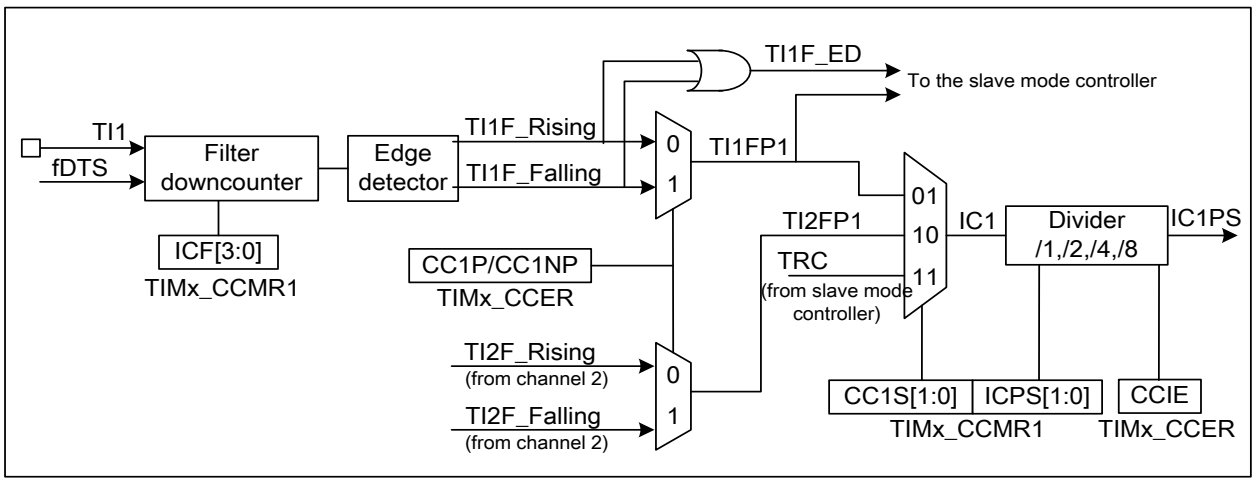


图 16.27 捕获/比较通道（如：通道 1 输入部分）

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。

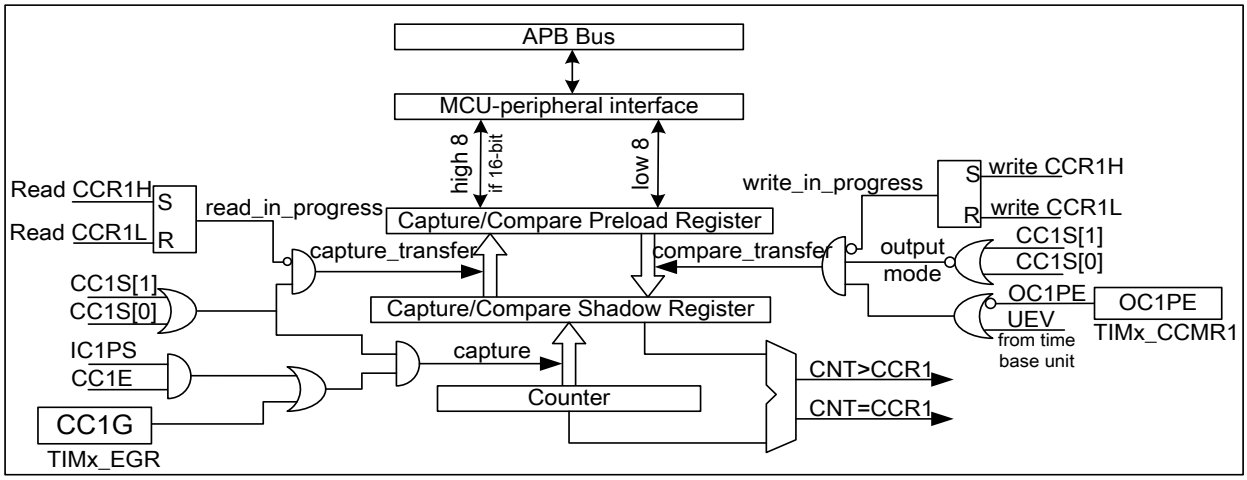


图 16.28 捕获/比较通道 1 主要框

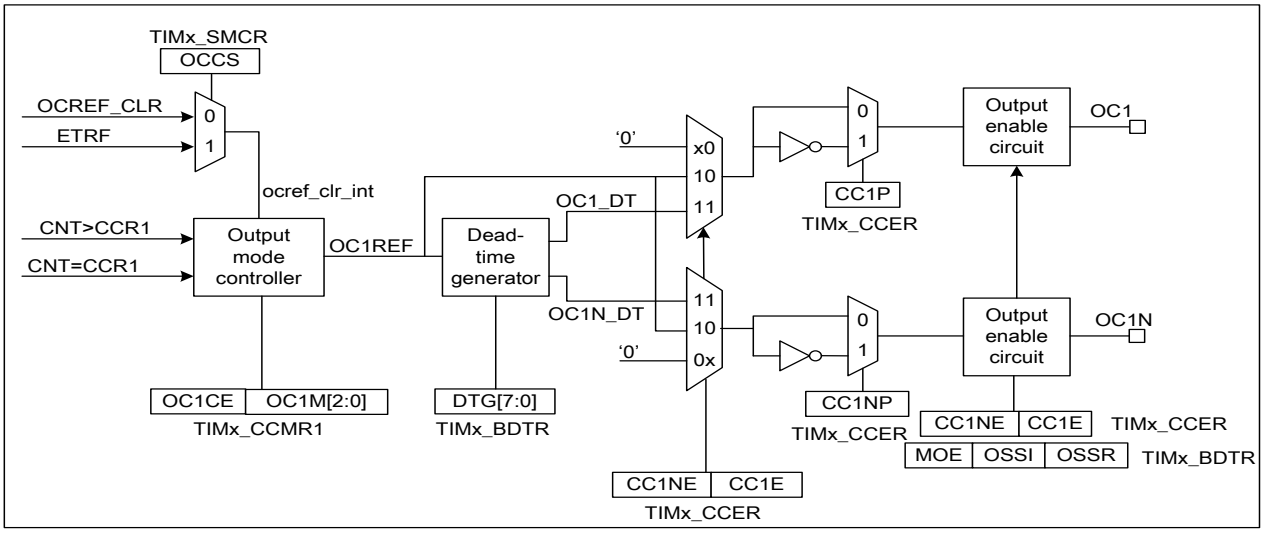


图 16.29 捕获/比较通道的输出部分（通道 1 至 3）

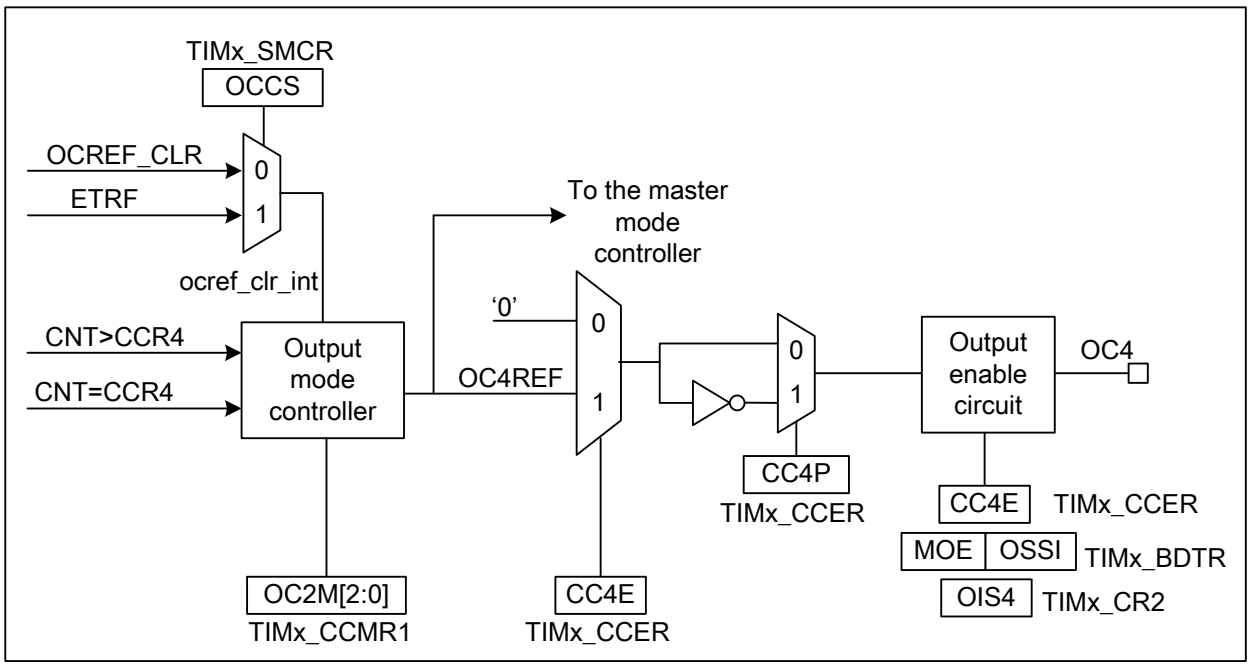


图 16.30 捕获/比较通道的输出 部分（通道 4）

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。

16.3.6. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿跳变时，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIMx_SR 寄存器）被置 1，如果使能了中断或者 DMA 则产生一个相应的中断或者一个相应的 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMx_SR 寄存器）被置 1。软件写写 CCxIF=0 可清楚 CCxIF，或者读取存储在 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：

- 选择有效输入端 :TIMx_CCR1 必须连接到 TI1 输入 ,所以需要写 TIMx_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为 0，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
- 根据连接到计数器的输入信号特点，配置输入滤波器为所需的带宽（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 频率采样）连续采样 8 次，以确认在 TI1 上一次新的边沿变换。然后在 TIMx_CCMR1 寄存器中写 IC1F=0011。
- 配置输入预分频器。在本例中，我们希望在每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
- 写 TIMx_CCER 寄存器的 CCIE=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 标志被置位。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如果设置了 CC1IE 位，则会产生一个中断。
- 如果设置了 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。

注意：通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断或 DMA 请求。

16.3.7. 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 信号被映射至同一个 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 信号被作为处罚输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的周期(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下（取决于 CK_INT 的频率和预分频器的值）：

- 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 寄存器中的 CC2S=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TIMx_CCR2 中）：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（选中 TI1FP1）。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 寄存器中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中的 CC1E=1 且 CC2E=1。

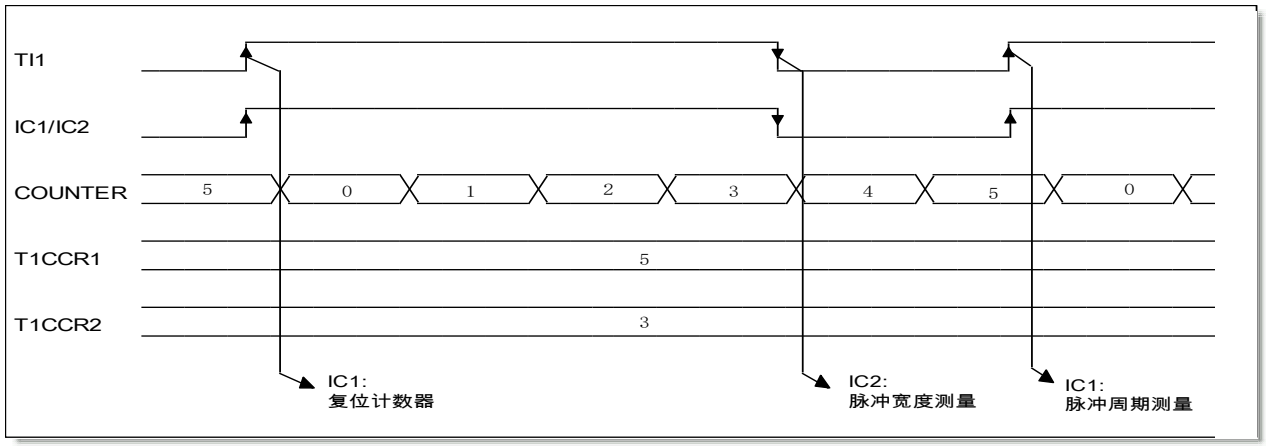


图 16.31 PWM 输入模式时序图

16.3.8. 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强制为有效或者无效状态，而不依赖于输出比较寄存器和计数器之间的比较结果。

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强制输出比较信号（OCxREF/OCx）为有效状态。这样

OCxREF 被强制为高电平（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 被强制为高电平。
置 TIMx_CCMRx 寄存器中的 OCxM=100，可强制 OCxREF 信号为低。
该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。

16.3.9. 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=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 ）。
- 置位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。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也可以用来输出一个单脉冲。
输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器 ）。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   - 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被更新 ）。下图给出了一个例子。

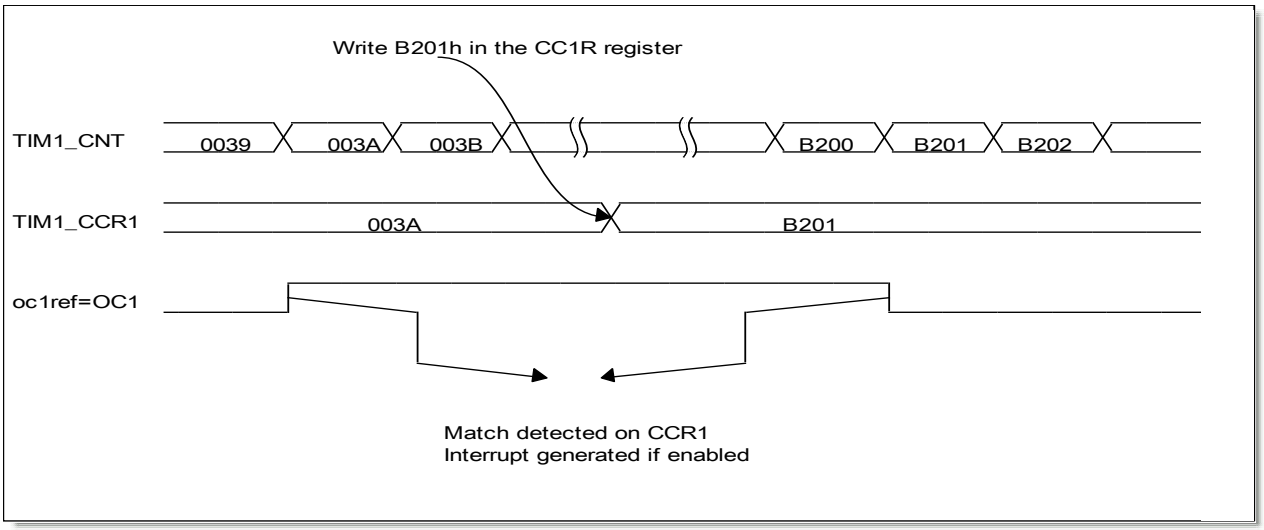


图 16.32 输出比较模式，翻转 OC1

16.3.10. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率，由 TIMx_CCRx 寄存器确定占空比的信号。在 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

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

PWM 边沿对齐模式

- 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值(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。

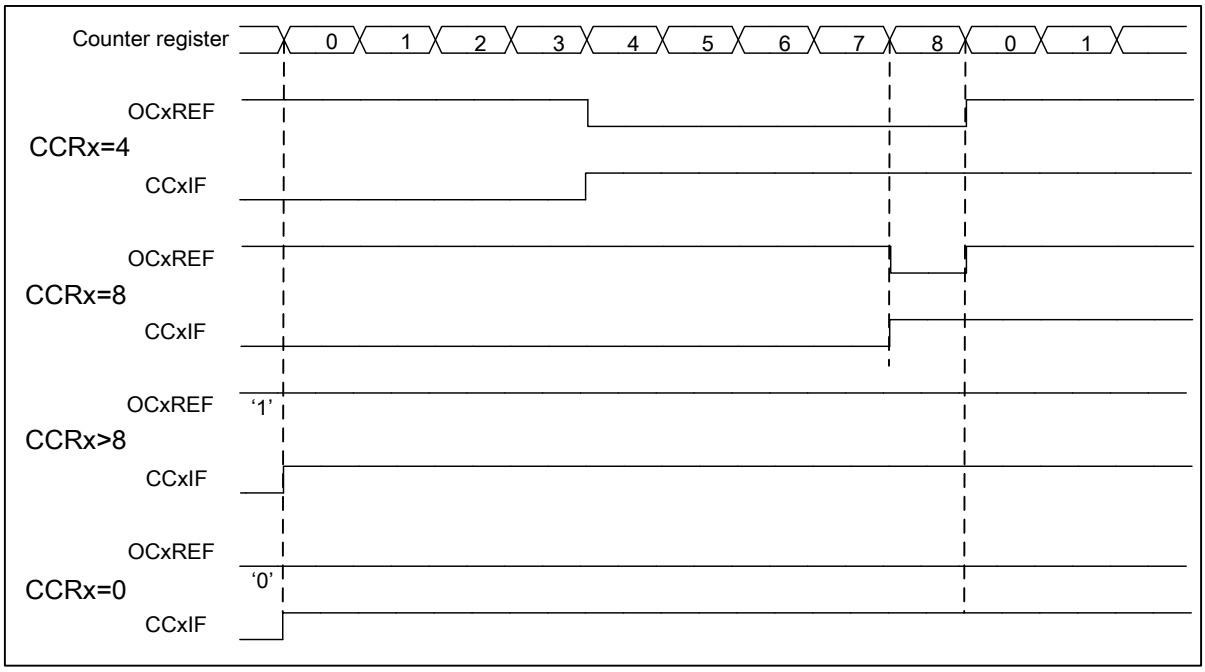


图 16.33 边沿对齐的 PWM 波形（ARR=8）

- 向下计数配置

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

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为 0 时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时，在计数器向下计数时，或在计数器向上和向下计数器时被置 1。TIMx_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。下图给出了一些中央对齐的 PWM 波形的例子

- TIMx_ARR=8
- PWM 模式 1

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

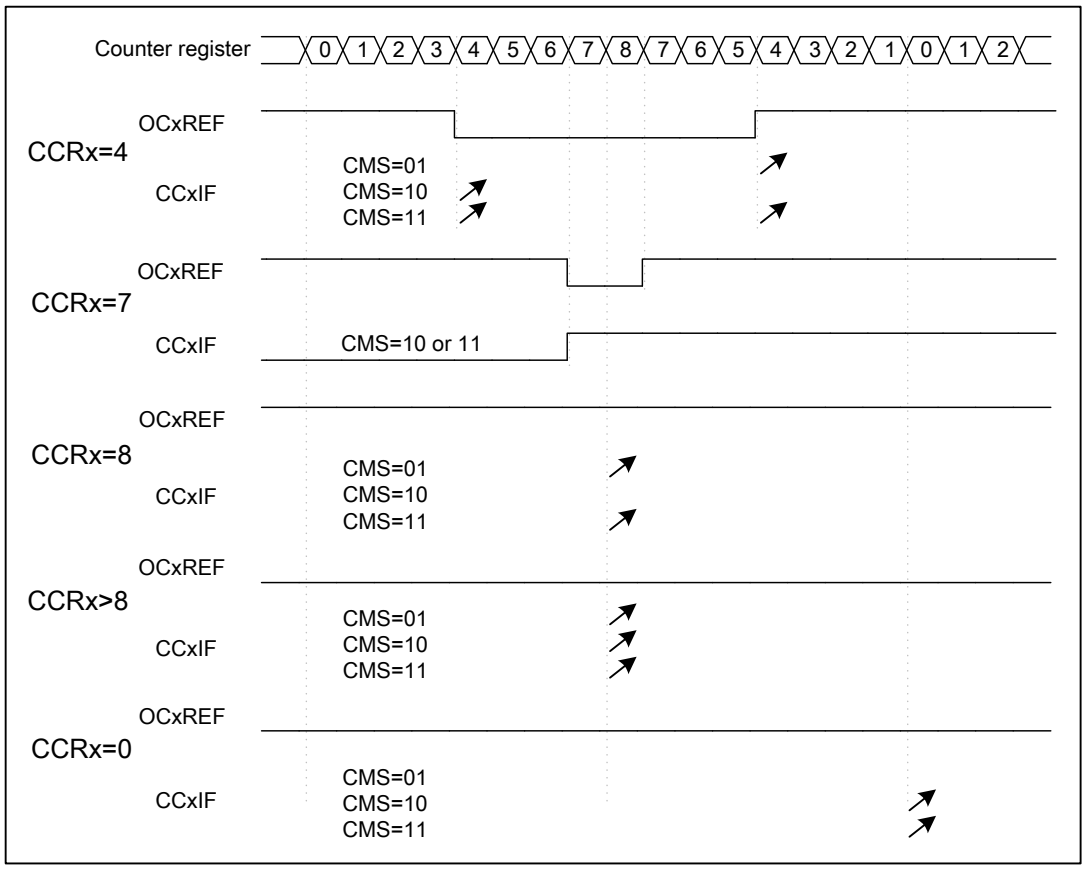


图 16.34 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

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

16.3.11. 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开关和接通。

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 通过下列控制的组合进行控制：TIMx_CCER 寄存器中的 CCxE 和 CCxNE，TIMx_BDTR 寄存器和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详细说明见表格：带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。特别的是，在转换到 IDLE 状态时（MOE 下降到 0）死

区被激活。

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

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

如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。

下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系（假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

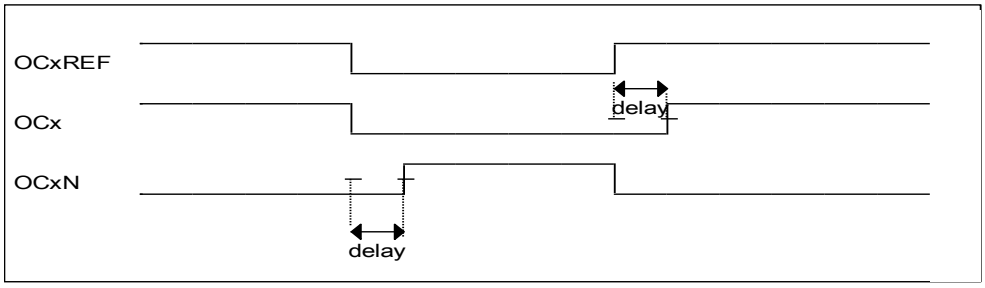


图 16.35 带死区插入的互补输出

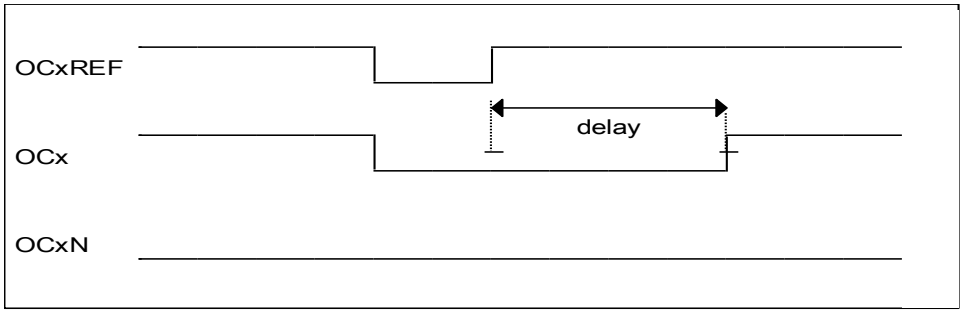


图 16.36 死区波形延迟大于负脉冲

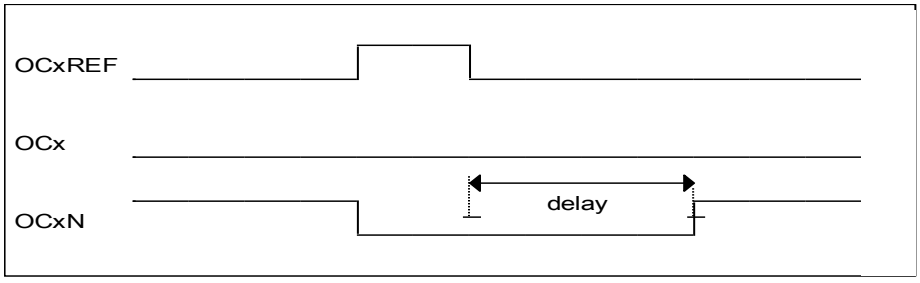


图 16.37 死区波形延迟大于正脉冲

每一个通道的死去延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。详细说明将 TIM1 刹车章节和死区寄存器（TIMx_BDTR）中的延时计算。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强制、输出比较或 PWM），通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注意：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 OCxNE=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时（CCxE=CCxNE=1），当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

16.3.12. 使用刹车功能

当使用刹车功能时，依据相应的控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位），输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详细说明见表格：带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。

刹车源既可以是外部刹车输入引脚又可以是下列任意一种内部刹车源：

- 内核的 LOCKUP 输出
- PVD 输出
- CSS 检测到的时钟缺失事件
- 比较器的输出

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，才能正确的读回写入的位。

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出的是同步信号。

当发生刹车时（在刹车输入端出现选定的电平），有下列动作：

- MOE 位被异步清除，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。
- 当使能互补输出时：
  - 输出首先被置于复位状态即无效状态（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电平。注意：因为 MOE 的重新同步，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或者一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置恶劣 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志（TIMx_SR 寄存器中的 BIF 位）为 1 时，则产生一个中断。

- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次写入 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接到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

刹车由 BRK 输入产生，它的有效极性是可配置的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性）。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参考章节 16.4.18：TIM1 刹车和死区寄存器（TIMx_BDTR）。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

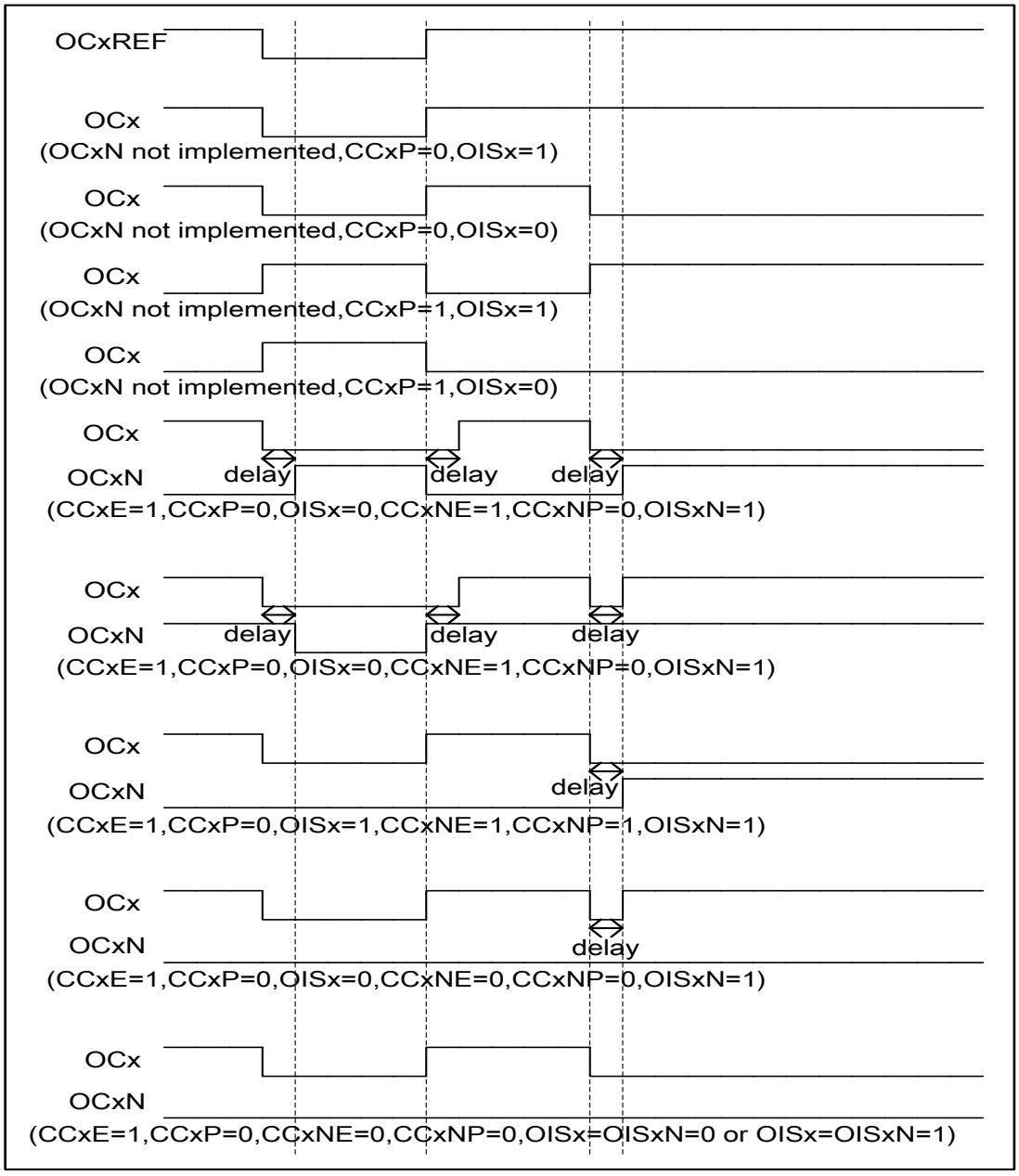


图 16.38 响应刹车的输出时序图

16.3.13. 外部事件清除 OCxREF 信号

当信号 OCREF_CLR_INPUT（TIMx_CCMRx 寄存器中的 OCxCE 位为 1）为高电平时，能把一个给定通道的 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强制模式。

OCREF_CLR_INPUT 的信号源有两个，分别为内部 OCREF_CLR 信号或 ETRF 信号（ETR 经过滤波后的信号），可通过 TIMx_SMCR 寄存器中的 OCCS 位进行选择。

例如，OCxREF 信号可以连接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. 外部触发预分频必须关闭：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）可以根据需要进行配置。

下图显示了当 ETRF 输入变为高时，对应不同的 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。

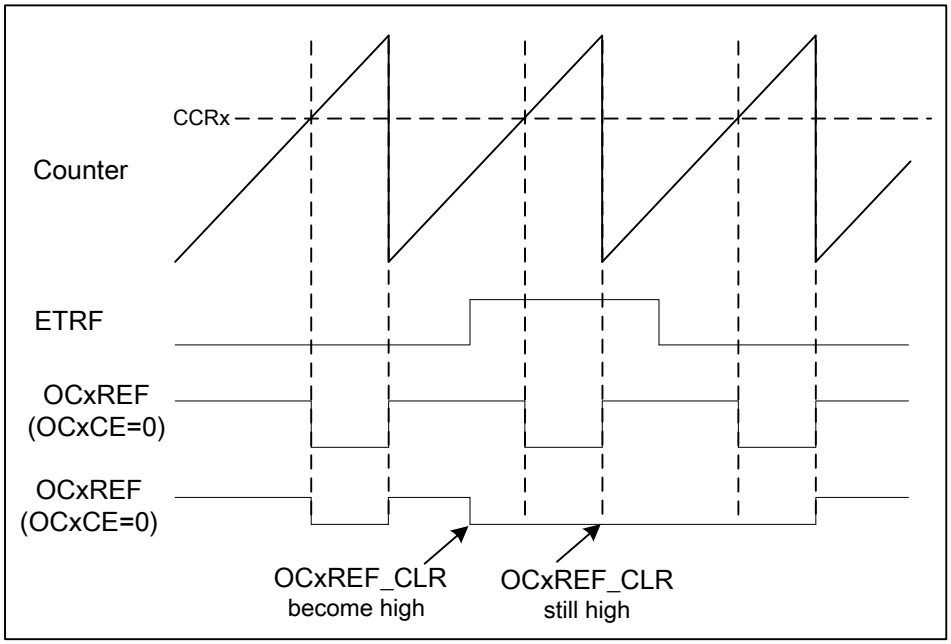


图 16.39 清除 OCxREF 的时序图

注意：在 100%的 PWM 时（如果 CCRx>ARR），OCxREF 在下一次计数溢出时被再次使能。

16.3.14. 六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器中。因此，可以预先设置好下一步骤所需配置，并且在同一时刻同时更改所有通道的配置。COM 换相事件可以通过设置 TIMx_EGR 寄存器中的 COM 位由软件产生，或者在 TRGI 上升沿由硬件产生。

当发生 COM 换相事件时，会置位一个标志位（TIMx_SR 寄存器中的 COMIF 位），这时如果已设置了 TIMx_DIER 寄存器中的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器中的 COMDE 位，则产生一个 DMA 请求。

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

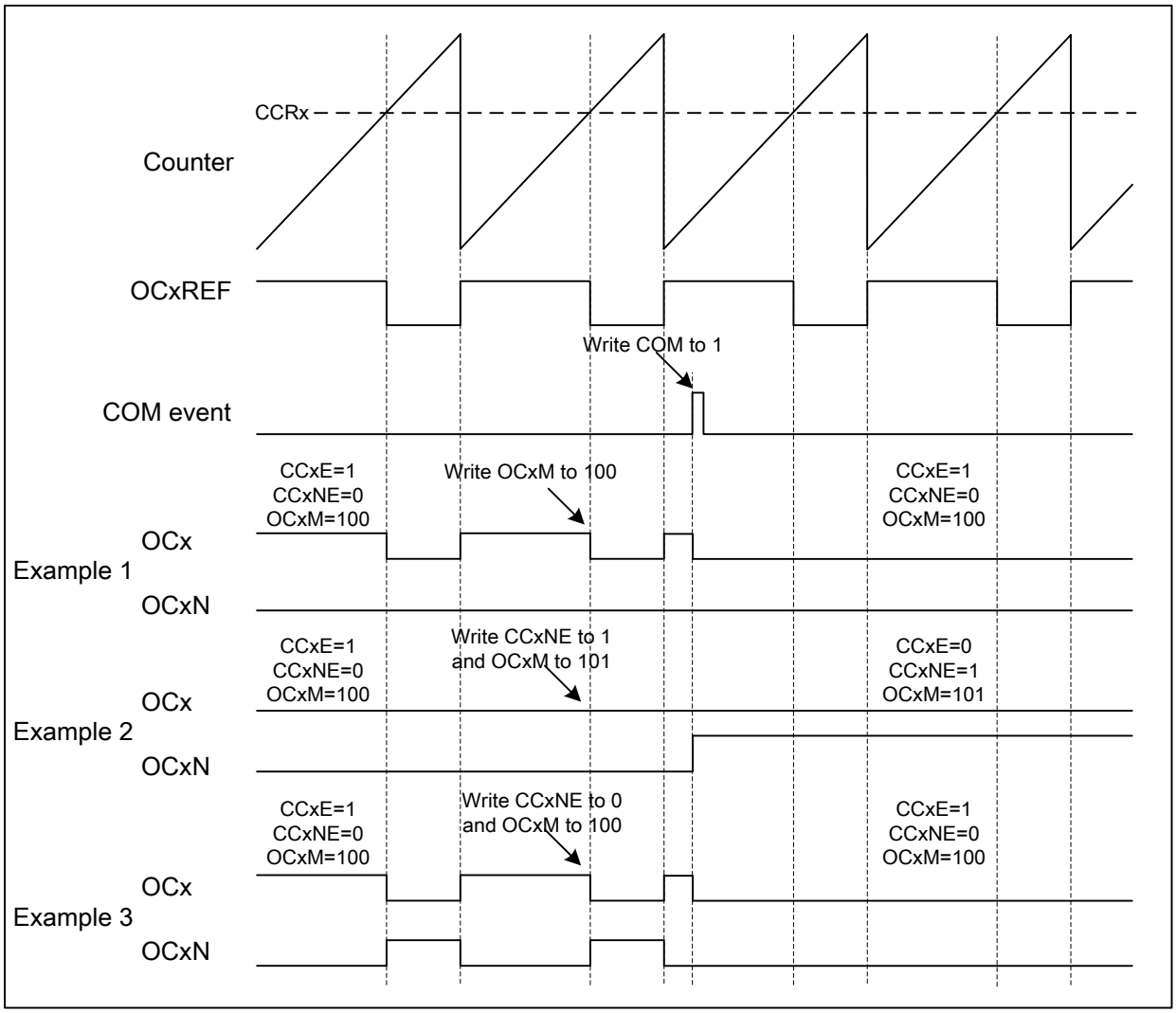


图 16.40 使用 COM 产生六步 PWM（OSSR=1）

16.3.15. 单脉冲模式

单脉冲模式（OPM）是前述众多模式中的一个特例。这种模式允许计数器通过响应一个激励来启动，并在一个程序可控的延时之后产生一个脉宽程序可控的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器在产生下一个更新事件 UEV 时自动停止。

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

- 在向上计数模式下，计数器 CNT<CCRx<=ARR（特别的，0<CCRx）。
- 在向下计数模式下，计数器 CNT>CCRx。

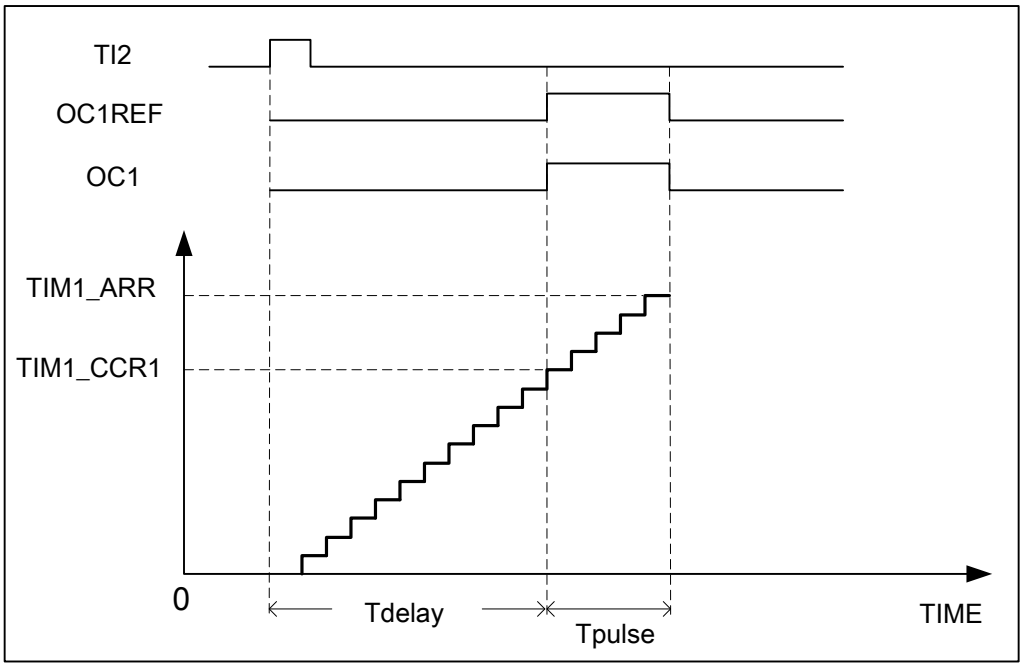


图 16.41 单脉冲模式示意图

例如 ,需要从 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。
假设 TI2FP2 作为触发 1：

- 写 TIMx_CCMR1 寄存器中的 CC2S=01，将 TI2FP2 映像到 TI2 上。
- 写 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 写 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
- 写 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 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换；首先要写 TIMx_CCMR1 寄存器的 OC1M=1111，选择 PWM 模式 2；根据需要有选择的使能预装载寄存器，写 TIMx_CCMR1 寄存器中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；此时必须向 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动装载值，设置 UG 位来产生一个更新事件，然后等待 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。当 TIMx_CR1 寄存器中的 OPM=0，进入重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但这些操作需要一定的时钟周期，因此限制了可得到的最小延时T_{delay}。
如果需要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF 直接响应激励而不再依赖比较的结果 ,输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

16.3.16. 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 TI2 的边沿计数，则写 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则写 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则写 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进行配置。CC1NP 和 CC2NP 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口，参看表格 16.1。假定计数器已经启动（TIMx_CR1 寄存器中的 CEN=1），则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 再通过输入滤波器和极性控制后产生的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位修改。不管计数器是依靠 TI1 还是 TI2 或者同时依靠 TI1 和 TI2 计数，在任一输入端的跳变都会重新计算 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、重复计数器、触发输出特性等仍能正常工作。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，计数器增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 16.1 计数方向与编码器信号的关系

有效计数沿	相对信号的电平值 (TI1FP1 for TI2) (TI2FP2 for TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
TI1	高	向下	向上	——	——
	低	向上	向下	——	——
TI2	高	——	——	向上	向下
	低	——	——	向下	向上
TI1 or TI2	高	向下	向上	向上	向下
	低	向上	向下	向下	向上

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰的能力。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，TI1FP1 映射到 TI1）。
- CC2S=01（TIMx_CCMR2 寄存器，TI2FP2 映射到 TI2）。
- CC1P=0（TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1）。
- CC2P=0（TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2=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）。

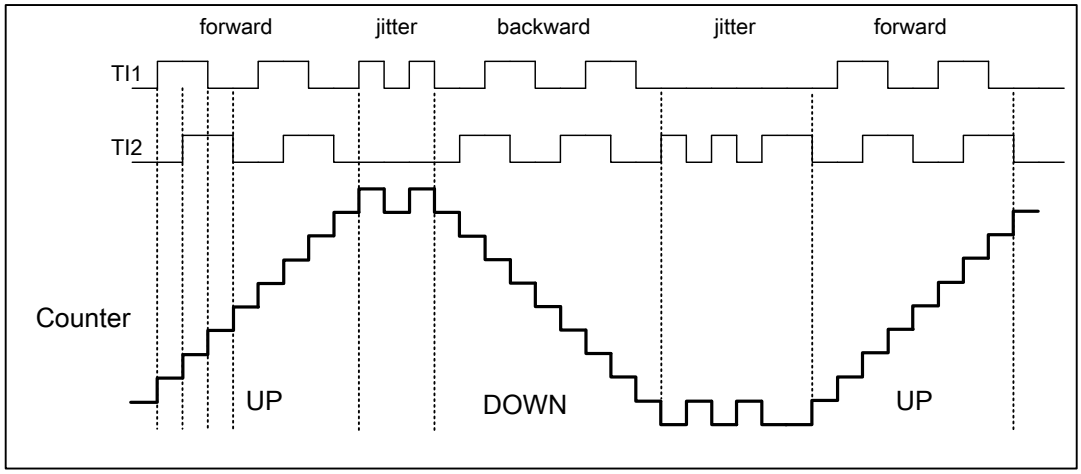


图 16.42 编码器模式下的计数器操作

下图为当 TI1FP1 极性反相时计数器的操作实例（CC1P=1，其他配置与上例相同）

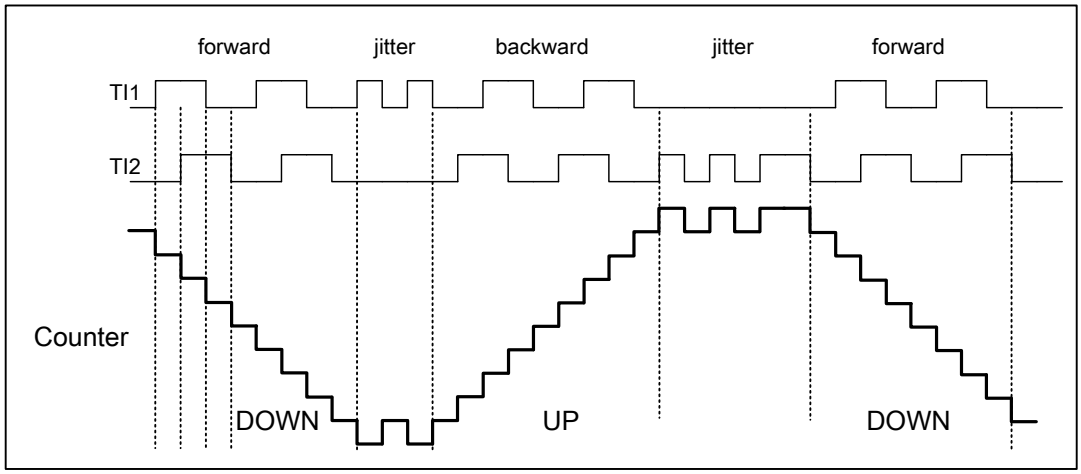


图 16.43 编码器模式下的计数器操作（TI1FP1 反相）

当定时器配置成编码器接口模式时，用于指示传感器当前位置。配合使用第二个定时器配置为捕获模式，通过测量两个编码器事件的间隔，从而获得动态信息（速度、加速度、负加速度）。编码器输出还可用来指示机械零点。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果有必要，可以把计数器的值锁存到第三个输入捕获寄存器（由另一个定时器产生的捕获信号必须是周期性）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

16.3.17. 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或端的 4 个输入端为 TIMx_CH1、TIMx_CH2、TIMx_CH3 和 TIMx_CH4。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

16.3.18. 霍尔传感器的接口

使用高级控制定时器（TIM1）产生 PWM 信号驱动马达，用另一个 TIMx（TIM3）定时器作为接口定时器来连接霍尔传感器。4 个定时器输入脚（CC1，CC2，CC3，CC4）通过一个异或门连接到 TI1 输入通道（通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择），“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。这样每当 4 个输入之一变化时，计数器从 0 重新开始计数。由此产生一个霍尔输入端的任何变化而触发的时间基准。

接口定时器模式下，捕获/比较通道 1 被配置为捕获模式，捕获信号为 TRC。捕获值反映了输入端两次变化之间的时间间隔，指示出了马达速度的信息。

接口定时器可以用来在输出模式产生一个脉冲，这个脉冲可以（通过触发一个 COM 换相事件）用于改变高级定时器（TIM1）各个通道的属性，TIM1 产生 PWM 信号驱动马达。因此，接口定时器通道必须编程为在一个指定的延时（输出比较或 PWM 模式）之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器（TIM1）。

举例：霍尔输入连接到 TIMx 定时器，要求在每次霍尔输入发生变化之后的一个指定的时刻，改变高级控制定时器的 PWM 配置。

- 写 TIMx_CR2 寄存器中的 TI1S 位为 1，配置 4 个定时器输入逻辑异或到 TI1 输入。
- 时基编程：写 TIMx_ARR 为其最大值（计数器必须通过 TI1 的变化清零）。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
- 设置通道 1 为捕获模式（TRC）：写 TIMx_CCMR1 寄存器中的 CC1S=01，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM 模式 2，带指定的延时：写 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：写 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被变成为 PWM 信号，捕获/比较控制信号为预装载的（TIMx_CR2 寄存器中等的 CCPC=1），由触发输入控制 OCM 换相事件（TIMx_CR2 寄存器中的 CCUS=1）。在一次 COM 换相事件后，下一步再写入 PWM 控制位（CCxE、OCxM），这可以在处理 OC2REF 上升沿的中断子程序里实现。

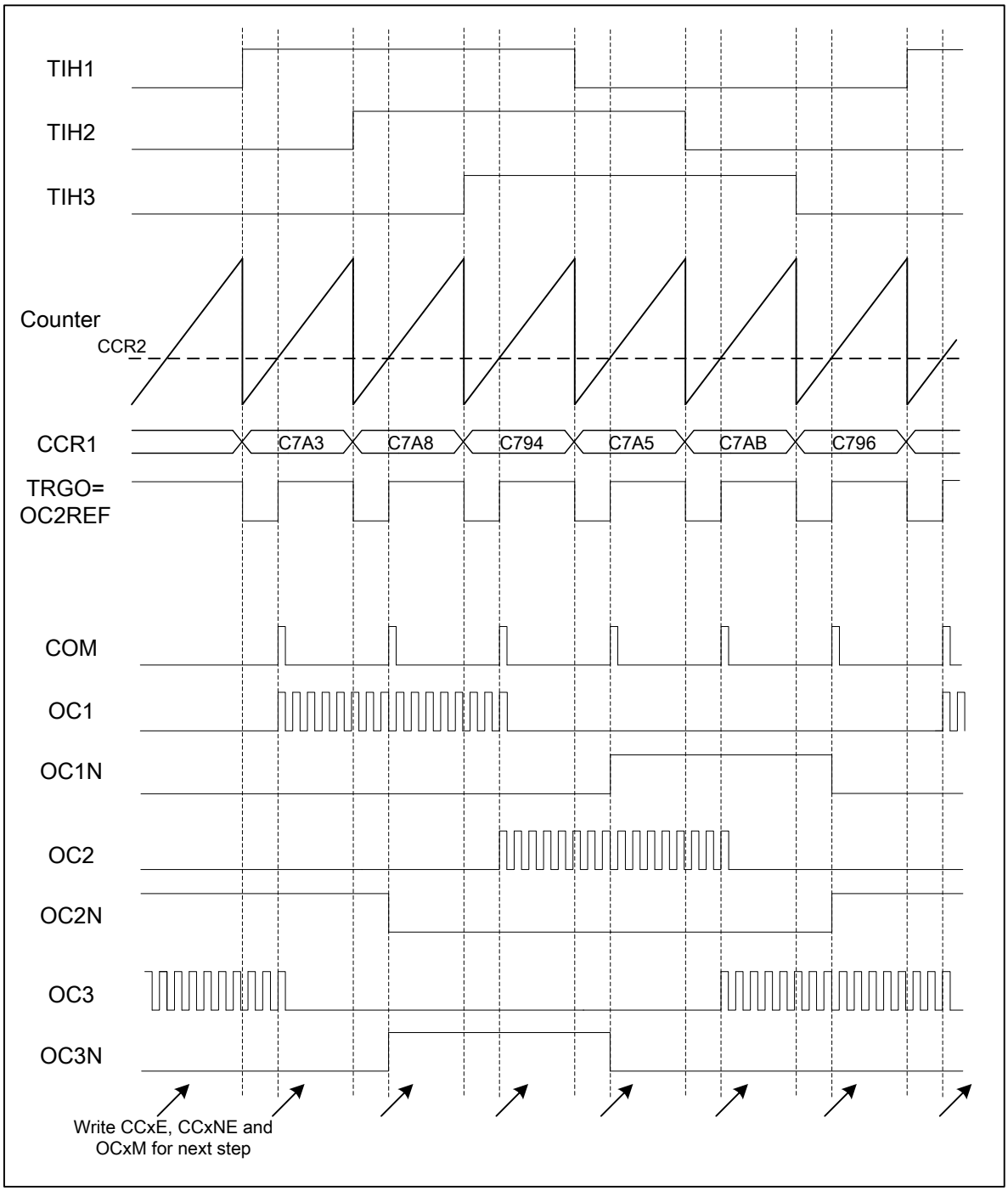


图 16.44 霍尔传感器接口实例

16.3.19. TIMx 定时器和外部触发的同步

TIMx 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。

从模式：复位模式

在一个触发输入事件发生时，计数器和它的预分频器被重新初始化；同时，如果 TIMx_CR1 寄存器的 URS 位

为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIMx_ARR、TIMx_CCRx）都被更新。
在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中的 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）。
- 写 TIMx_SMCR 寄存器中的 SMS=100，配置定时器为复位模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 写 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，并根据 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 上升沿河计数器的实际复位之间的延时取决于 TI1 输入端的再同步电路。

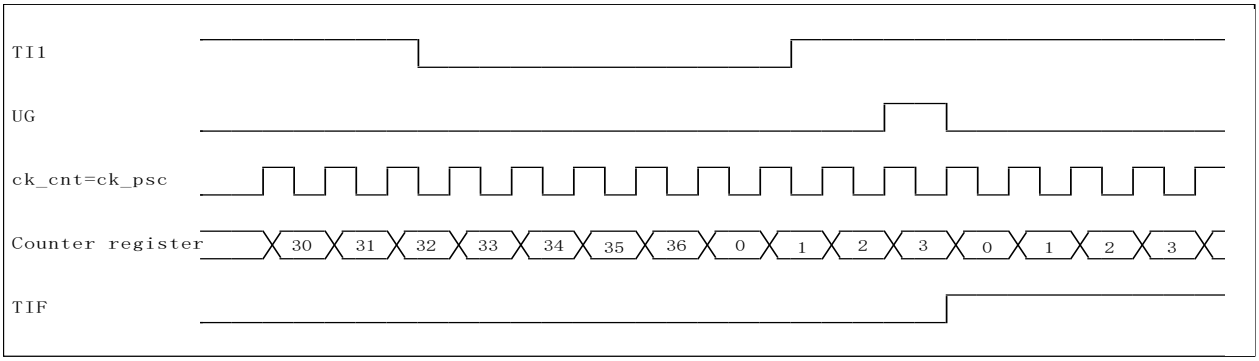


图 16.45 复位模式下的控制时序图

从模式：门控模式

可以按照选中的输入端电平使能计数器。
在如下的例子中，计数器只能在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波器，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，写 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 写 TIMx_SMCR 寄存器中的 SMS=101，配置定时器为门控模式；写 TIMx_SMCR 寄存器中的 TS=101，选择 TI1 作为输入源。
- 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都会将 TIMx_SR 寄存器中的 TIF 标志置位。
TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的再同步电路。

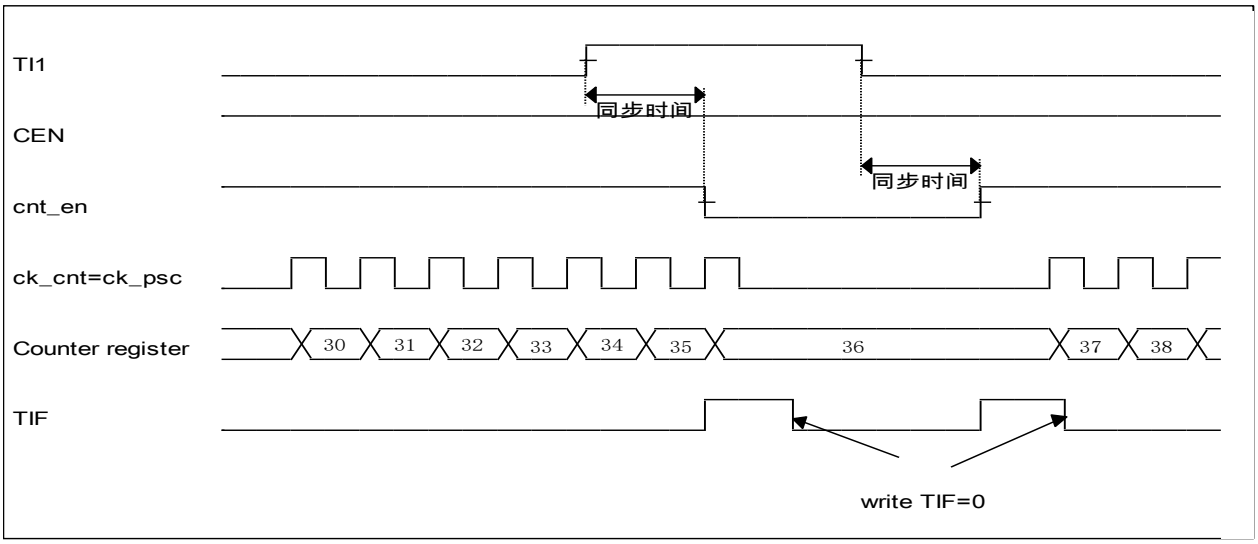


图 16.46 门控模式下的控制时序图

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 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，写 TIMx_CCMR1 寄存器中的 CC2S=01。写 TIMx_CCER 寄存器中的 CC2P=1 和 CC2NP=0 以确定极性（只检测低电平）。
- 写 TIMx_SMCR 寄存器中的 SMS=110，配置定时器为触发模式；写 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，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 输入端的再同步电路。

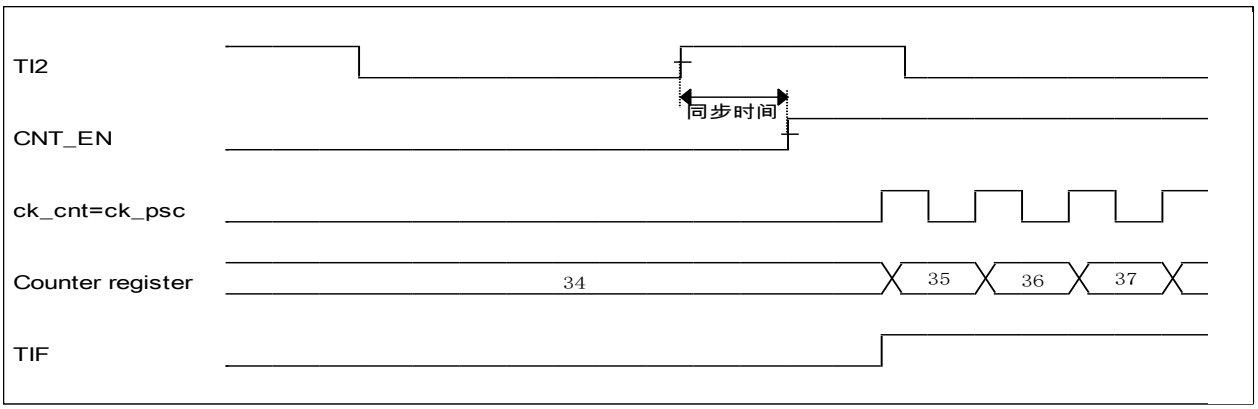


图 16-47 触发模式下的控制时序图

从模式：外部时钟模式 2 +触发模式

外部时钟模式 2 可以与另一个从模式（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，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 TIMx_SMCR 寄存器中的 TS 位来选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，向上计数器在 ETR 的每一个上升沿加 1：

1. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：
   - ETF=0000：没有滤波
   - ETPS=00：不用预分频器
   - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2
2. 按下列方式配置通道 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
   - 写 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）
3. 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。写 TIMx_SMCR 寄存器中的 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被置位，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之间的延时取决于 ETRP 输入端的再同步电路。

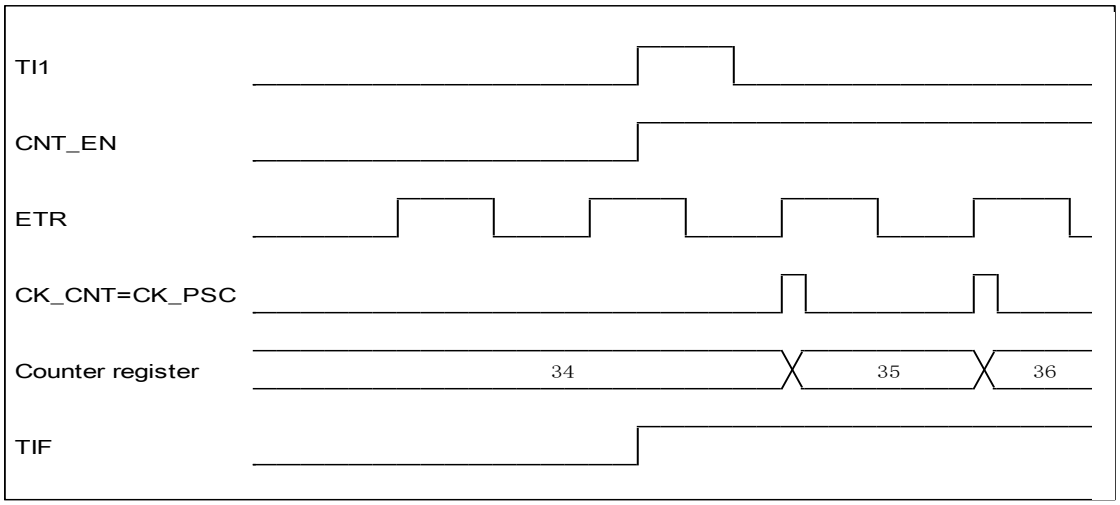


图 16.48 外部时钟模式 2 + 触发模式下的控制时序图

16.3.20. 定时器同步

TIM 定时器在内部相连，用于定时器的同步或链接。详细说明参考章节 17.3.15：定时器同步。

16.3.21. 调试模式

当微控制器进入调试模式时（Cortex_M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止。

16.4. TIM1 寄存器映射

下表给出了 TIM1 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM1_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0
0x04	TIM1_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	-	CCPC
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0
0x08	TIM1_SMCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIM1_DIER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	TIM1_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC4OF	CC3OF	CC2OF	CC1OF	-	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	0	0	0	0	0	0	0	0
0x14	TIM1_EGR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x18	TIM1_CCMR1（输出模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIM1_CCMR1（输入模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x1C	TIM1_CCMR2（输出模式）																	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
	TIM1_CCMR2（输入模式）																	IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x20	TIM1_CCER																			CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x24	TIM1_CNT																	CNT[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x28	TIM1_PSC																	PSC[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x2C	TIM1_ARR																	ARR[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1						
0x30	TIM1_RCR																									REP[7:0]													
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0						
0x34	TIM1_CCR1																	CCR1[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x38	TIM1_CCR2																	CCR2[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x3C	TIM1_CCR3																	CCR3[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x40	TIM1_CCR4																	CCR4[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x44	TIM1_BDTR																	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]													
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
0x48	TIM1_DCR																				DBL[4:0]								DBA[4:0]									
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0						
0x4C	TIM1_DMAR																	DMAB[15:0]																					
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						

16.4.1. TIM1 控制寄存器 1（TIM1_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CKD[1:0]	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	CKD[1:0]	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，不要使用此配置
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:5	CMS[1:0]	选择中央对齐模式 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向下计数时被置1。 10：中央对齐模式2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向上计数时被置1。 11：中央对齐模式3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被置1。 注意 :在计数器开启时(CEN=1) ,不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	计数方向 0：计数器向上计数； 1：计数器向下计数。 注意：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	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	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，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 – 软件设置UG位 – 复位触发事件产生的更新 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新 软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出/下溢 – 软件设置UG位 – 复位触发事件产生的更新 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果触发复位模式下触发事件到来时或软件设置UG位，计数器和预分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。 注意：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。而触发模式下可以自动地通过硬件设置CEN位。

16.4.2. TIM1 控制器 2（TIM1_CR2）

地址偏移：0x04

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
7:0	TI1S	MMS[2:0]			CCDS	CCUS	—	CCPC
类型	RW	RW	RW	RW	RW	RW	RO-0	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	OIS4	输出空闲状态4(OC4输出)。参见OIS1位。
13	OIS3N	输出空闲状态3(OC3N输出)。参见OIS1N位。
12	OIS3	输出空闲状态3(OC3输出)。参见OIS1位。
11	OIS2N	输出空闲状态2(OC2N输出)。参见OIS1N位。

10	OIS2	输出空闲状态2(OC2输出)。参见OIS1位。
9	OIS1N	输出空闲状态1(OC1N输出)。 0：当MOE=0时，则在一个死区时间后，OC1N=0； 1：当MOE=0时，则在一个死区时间后，OC1N=1。 注意：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	输出空闲状态1(OC1输出)。 0：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=0； 1：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=1。 注意：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	TI1选择 0：CC1输入管脚连到TI1(数字滤波器的输入)； 1：CC1、CC2、CC3和CC4管脚经异或后连到TI1。
6:4	MMS[2:0]	主模式选择 这3位用于选择在主模式下送到其它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)。如果触发输入(时钟/触发控制器配置为复位模式)产生复位，则TRGO上的信号相对实际的复位会有一个延迟。 001：使能 – 计数器使能信号被用于作为触发输出(TRGO)。其用于启动多个定时器以便控制在一段时间内使能从定时器。计数器使能信号是通过CEN控制位和门控模式下的触发输入信号的逻辑或产生。除非选择了主/从模式(见TIM1_SMCR寄存器中MSM位的描述)， 当计数器使能信号受控于触发输入时，TRGO上会有一个延迟。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲(MATCH1) – 一旦发生一次捕获或一次比较成功，当CC1IF标志被置1时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF信号被用于作为触发输出(TRGO)。
3	CCDS	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	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。 注意：该位只对具有互补输出的通道起作用。
1	NA	保留位，未定义
0	CCPC	捕获/比较预装载控制位 0：CCxE，CCxNE，CCxP，CCxNP和OCxM位不是预装载的； 1：CCxE，CCxNE，CCxP，CCxNP和OCxM位是预装载的；设置该位后，只在设置了COMG位或TRGI检测到上升沿（取决于CCUS位的设置）后被更新。

		注意：该位只对具有互补输出的通道起作用。

16.4.3. TIM1 从模式控制寄存器（TIM1_SMCR）

地址偏移：0x08

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	ETP	ECE	ETPS[1:0]		ETF[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	MSM	TS[2:0]			OCCS	SMS[2:0]		
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	ETP	外部触发极性 该位决定是ETR还是ETR的反相用于触发操作。 0：ETR不反相，即高电平或上升沿有效； 1：ETR反相，即低电平或下降沿有效。
14	ECE	外部时钟使能 该位用于使能外部时钟模式2。 0：禁止外部时钟模式2； 1：使能外部时钟模式2，计数器由ETRF信号上的的任意有效边沿驱动。 注1：ECE位置1的效果与选择把TRGI连接到ETRF的外部时钟模式1相同(TIM1_SMCR寄存器中，SMS=111，TS=111)。 注2：外部时钟模式2可与下列模式同时使用：触发模式；复位模式；门控模式。但是，此时TRGI决不能与ETRF相连。 注3：外部时钟模式1与外部时钟模式2同时被使能时，外部时钟输入为ETRF。
13:12	ETPS[1:0]	外部触发预分频器 外部触发信号EPRP的频率最大不能超过TIMxCLK频率的1/4。可用预分频器来降低ETRP的频率，当EPRP的频率很高时，它非常有用： 00：预分频器关闭； 01：EPRP的频率/2； 02：EPRP的频率/4； 03：EPRP的频率/8。
11:8	ETF[3:0]	外部触发滤波器选择 这些位定义了ETRP的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到N个事件后会产生一个输出的跳变： 0000：无滤波器，以$f_{SAMPLING} = f_{DTS}$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	主/从模式 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许当前定时器与它的从定时器间的完美同步(通过TRGO)。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	触发选择 这3位选择用于选择同步计数器的触发输入。 000：内部触发0（ITR0） 001：保留 010：内部触发2（ITR2） 011：内部触发3（ITR3） 100：TI1的边沿检测器(TI1F_ED) 101：滤波后的定时器输入1(TI1FP1) 110：滤波后的定时器输入2(TI2FP2) 111：外部触发输入(ETRF) 注意：这些位只能在未用到(如SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	OCCS	OCxREF清除选择 此位用于选择清除COxREF信号的清除源。 0：OCREF_CLR_INT信号连接到OCREF_CLR内部输入信号上； 1：OCREF_CLR_INT信号连接到ETRF信号上。
2:0	SMS[2:0]	时钟/触发/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：时钟/触发控制器禁止 – 如果CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式1 – 根据TI1FP1的电平，计数器在TI2FP2的边沿向上/下计数。 010：编码器模式2 – 根据TI2FP2的电平，计数器在TI1FP1的边沿向上/下计数。 011：编码器模式3 – 根据另一个输入的电平，计数器在TI1FP1和TI2FP2的边沿向上/下计数。 100：复位模式 – 在选中的触发输入(TRGI)的上升沿时重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计

		数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入TRGI的上升沿启动(但不复位) ,只有计数器的启动是受控的。 111：外部时钟模式1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果TI1F_ED被选为触发输入(TS=100)时，不要使用门控模式。这是因为TI1F_ED在每次TI1F变化时只是输出一个脉冲 ,然而门控模式是要检查触发输入的电平。

表 16.3 TIMx 内部触发连接

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

16.4.4. TIM1 DMA/中断使能寄存器（TIM1_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
7:0	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	TDE	允许触发 DMA 请求 0：触发 DMA 请求禁止 1：触发 DMA 请求允许
13	COMDE	COM 换相事件 DMA 请求使能 0：换相事件 DMA 请求禁止 1：换相事件 DMA 请求允许
12	CC4DE	捕获/比较 4 DMA 请求使能 0：CC4 DMA 请求禁止 1：CC4 DMA 请求允许
11	CC3DE	捕获/比较 3 DMA 请求使能 0：CC3 DMA 请求禁止 1：CC3 DMA 请求允许
10	CC2DE	捕获/比较 2 DMA 请求使能

		0：CC2 DMA 请求禁止 1：CC2 DMA 请求允许
9	CC1DE	捕获/比较 1 DMA 请求使能 0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
8	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允许
7	BIE	刹车中断使能 0：刹车中断禁止 1：刹车中断允许
6	TIE	触发中断使能 0：触发中断禁止 1：触发中断允许
5	COMIE	COM 换相事件中断使能 0：COM 中断禁止 1：COM 中断允许
4	CC4IE	捕获/比较 4 中断使能 0：CC4 中断禁止 1：CC4 中断允许
3	CC3IE	捕获/比较 3 中断使能 0：CC3 中断禁止 1：CC3 中断允许
2	CC2IE	捕获/比较 2 中断使能 0：CC2 中断禁止 1：CC2 中断允许
1	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

16.4.5. TIM1 状态寄存器（TIM1_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

15:8	—			CC4OF	CC3OF	CC2OF	CC1OF	—
类型	RO-0	RO-0	RO-0	RC_W0	RC_W0	RC_W0	RC_W0	RO-0
7:0	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
类型	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	Function
31:13	NA	保留位，未定义
12	CC4OF	捕获/比较4重复捕获标志 参见CC1OF描述。
11	CC3OF	捕获/比较3重复捕获标志 参见CC1OF描述。
10	CC2OF	捕获/比较2重复捕获标志 参见CC1OF描述。
9	CC1OF	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，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIM1_CCR1寄存器时，CC1IF的状态已经为1。
8	NA	保留位，未定义
7	BIF	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	TIF	触发器中断标志 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在TRGI输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置1。它由软件清0。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	COM中断标志 一旦产生COM事件(当捕获/比较控制位：CCxE、CCxNE、OCxM已被更新)该位由硬件置1。它由软件清0 。 0：无COM事件产生； 1：COM中断等待响应。
4	CC4IF	捕获/比较4中断标志 参考CC1IF描述。
3	CC3IF	捕获/比较3中断标志 参考CC1IF描述。
2	CC2IF	捕获/比较2中断标志 参考CC1IF描述。
1	CC1IF	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，但在中心对称模式下除外(参考TIM1_CR1寄存器的CMS位)。它由软件清0。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_ARR的内容时，在向上或向上/向下计数模式

		时计数器溢出，或向下计数模式时计数器下溢条件下，CC1IF位变高。 如果通道CC1配置为输入模式： 当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIMx_CCR1清0。 0：无输入捕获产生； 1：计数器值已被捕获至TIMx_CCR1(在IC1上检测到与所选极性相同的边沿)。
0	UIF	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，当计数器上溢或下溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当计数器CNT被触发事件重新初始化时 (参考从模式控制寄存器TIMx_SMCR)。

16.4.6. TIM1 事件产生寄存器（TIM1_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
类型	W	W	W	W	W	W	W	W

Bit	Name	Function
31:8	NA	保留位，未定义
7	BG	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	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，若开启对应的中断和DMA，则产生相应的中断和DMA。
5	COMG	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、CCxP、CCxNP、OCIM位。

		注意：该位只对拥有互补输出的通道有效。
4	CC4G	产生捕获/比较4事件 参考CC1G描述。
3	CC3G	产生捕获/比较3事件 参考CC1G描述。
2	CC2G	产生捕获/比较2事件 参考CC1G描述。
1	CC1G	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： 设置CC1IF=1，若开启对应的中断和DMA，则产生相应的中断和DMA。 若通道CC1配置为输入： 当前的计数器值被捕获至TIMx_CCR1寄存器，设置CC1IF=1，若开启对应的中断和DMA ,则产生相应的中断和DMA。若CC1IF已经为1 ,则设置CC1OF=1。
0	UG	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)。若在中心对称模式下或DIR=0(向上计数)则计数器被清0；若DIR=1(向下计数)则计数器取TIMx_ARR的值。

16.4.7. TIM1 捕获/比较模式寄存器（TIM1_CCMR1）

地址偏移：0x18

复位值：0x0000

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

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
	IC2F[3:0]				IC2PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15	OC2CE	输出比较2清零使能
14:12	OC2M[2:0]	输出比较2模式
11	OC2PE	输出比较2预装载使能
10	OC2FE	输出比较2快速使能
9:8	CC2S	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7	OC1CE	输出比较1清零使能 0：OC1REF不受ETRF输入的影响； 1：一旦检测到ETRF输入高电平，清除OC1REF。
6:4	OC1M[2:0]	输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1、OC1N的值。OC1REF是高电平有效，而OC1、OC1N的有效电平取决于CC1P、CC1NP位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平； 在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)， 否则为有效电平(OC1REF=1)。 111：PWM模式2 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平； 在向下计数时，一旦TIMx_CNT>TIM1_CCRx时通道1为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。 注3：在有互补输出的通道上，这些位是预装载的。如果TIMx_CR2寄存器的CCPC=1，OC1M 位只有在COM事件发生时，才从预装载位取新值。

3	OC1PE	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，并且新写入的数值立即起作用。 1：开启TIMx_CCR1寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1的预装载值在更新事件到来时被加载至当前寄存器中。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：为了操作正确，在PWM模式下必须使能预装载功能。但在单脉冲模式下(TIMx_CR1寄存器的OPM=1)，它不是必须的。
2	OC1FE	输出比较1 快速使能 该位用于加快CC输出对触发输入事件的响应。 0：根据计数器与CCR1的值，CC1正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活CC1输出的最小延时为5个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC被设置为比较电平而与比较结果无关。采样触发器的有效沿和CC1输出间的延时被缩短为3个时钟周期。 注意：OC1FE只在通道被配置成PWM1或PWM2模式时起作用。
1:0	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC2F[3:0]	输入捕获2滤波器
11:10	IC2PSC[1:0]	输入/捕获2预分频器
9:8	CC2S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7:4	IC1F[3:0]	输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，以$f_{SAMPLING} = f_{CK_INT}$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$f_{SAMPLING} = f_{D□S}/32$，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]	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]	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TIMx_SMCR寄存器的TS位选择)。 注：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

16.4.8. TIM1 捕获/比较模式寄存器 2（TIM1_CCMR2）

偏移地址：0x1C

复位值：0x0000

参考 CCMR1 寄存器的描述。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
	IC4F[3:0]				IC4PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	IC3F[3:0]				IC3PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15	OC4CE	输出比较4清零使能
14:12	OC4M[2:0]	输出比较4模式
11	OC4PE	输出比较4预装载使能
10	OC4FE	输出比较4快速使能
9:8	CC4S	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC4S仅在通道关闭时(TIMx_CCER寄存器的CC4E=0且已被更新)才是可写的。
7	OC3CE	输出比较3清零使能
6:4	OC3M[2:0]	输出比较3模式
3	OC3PE	输出比较3预装载使能
2	OC3FE	输出比较3快速使能
1:0	CC3S	捕获/比较3选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3通道被配置为输出； 01：CC3通道被配置为输入，IC3映射在TI3上； 10：CC3通道被配置为输入，IC3映射在TI4上； 11：CC3通道被配置为输入，IC3映射在TRC上。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC3S仅在通道关闭时(TIMx_CCER寄存器的CC3E=0、CC3NE=0且已被更新)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC4F[3:0]	输入捕获4滤波器
11:10	IC4PSC[1:0]	输入/捕获4预分频器
9:8	CC4S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC4S仅在通道关闭时(TIMx_CCER寄存器的CC4E=0且已被更新)才是可写的。
7:4	IC3F[3:0]	输入捕获3滤波器
3:2	IC3PSC[1:0]	输入/捕获3预分频器

1:0	CC3S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC3S仅在通道关闭时(TIMx_CCER寄存器的CC3E=0，CC3NE=0且已被更新)才是可写的。

16.4.9. TIM1 捕获/比较使能寄存器（TIM1_CCER）

地址偏移：0x20

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
类型	RO-0	RO-0	RW	RW	RW	RW	RW	RW
7:0	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:14	NA	保留位，未定义
13	CC4P	输入捕获/比较4输出极性。参考CC1P的描述。
12	CC4E	输入捕获/比较4输出使能。参考CC1E的描述。
11	CC3NP	输入捕获/比较3互补输出极性。参考CC1NP的描述。
10	CC3NE	输入捕获/比较3互补输出使能。参考CC1NE的描述。
9	CC3P	输入捕获/比较3输出极性。参考CC1P的描述。
8	CC3E	输入捕获/比较3输出使能。参考CC1E的描述。
7	CC2NP	输入捕获/比较2互补输出极性。参考CC1NP的描述。
6	CC2NE	输入捕获/比较2互补输出使能。参考CC1NE的描述。
5	CC2P	输入捕获/比较2输出极性。参考CC1P的描述。
4	CC2E	输入捕获/比较2输出使能。参考CC1E的描述。
3	CC1NP	输入捕获/比较1互补输出极性 CC1通道配置为输出： 0：OC1N高电平有效； 1：OC1N低电平有效。 CC1通道配置为输入： 本为用于和CC1P联合定义TI1FP1和TI2FP1的极性。参考CC1P的描述。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3且CC1S=00(通道配置为输出) 则该位不能被修改。

		注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1NP位才从预装载位中取新值。
2	CC1NE	输入捕获/比较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和CC1E位的值。 1：开启 - OC1N信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1E位的值。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1NE位才从预装载位中取新值。
1	CC1P	输入捕获/比较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位联合选择在触发或捕获模式下TI1FP1和TI2FP1的有效极性。 00：非反相/上升沿 电路作用于TIxFP1的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1非反相（在门控模式或编码模式）。 01：反相/下降沿 电路作用于TIxFP1的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 10：保留不用 11：非反相/上升或下降沿 电路作用于TIxFP1的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3，则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1P位才从预装载位中取新值。
0	CC1E	输入捕获/比较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和CC1NE位的值。 1：开启 - OC1信号输出到对应的输出引脚 ,其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1NE位的值 。 CC1通道配置为输入： 该位决定了计数器的值是否能捕获入TIMx_CCR1寄存器。 0：捕获禁止； 0：捕获使能。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1E位才从预装载位中取新值。

表 16.4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态（1）	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	0	1	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性 OCx=OCxREF ^ CCxNP OCx_EN=1	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
		1	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
		1	0	1	关闭状态(运行模式下输出使能) OCx=CCxP, OCx_EN=1	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		1	1	0	OCxREF + 极性 OCx=OCxREF ^ T1CCxNP OCx_EN=1	关闭状态(运行模式下输出使能) OCxN=T1CCxNP, OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
0	0	X	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	0		0	1	输出关闭(不由定时器驱动)	
	0		1	0	异步：OCx=CCxP,OCx_EN=0,OCxN=CCxNP,OCxN_EN=0	
	0		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	1		0	1	关闭状态(空闲模式下输出使能)	
	1		1	0	异步：OCx=CCxP,OCx_EN=1,OCxN=CCxNP,OCxN_EN=1	
	1		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	

（1） 如果一个通道的 2 个输出都没有使用（CCxE=CCxNE=0），那么 OISx，OISxN，CCxP 和 CCxNP 都必须清零。

注意：引脚连接到互补的 OCx 和 OCxN 通道的外部 I/O 引脚的状态，取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器。

16.4.10. TIM1 计数器（TIM1_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

16.4.11. TIM1 预分频器（TIM1_PSC）

地址偏移：0x28

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

16.4.12. TIM1 自动重装载寄存器（TIM1_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 参考章节13.3.1：时基单元中有关ARR的更新和动作的相关内容。 当自动装载值为空时，计数器不工作。

16.4.13. TIM1 重复计数寄存器（TIM1_RCR）

地址偏移：0x30

复位值：0x0000

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	REP[7:0]	重复计数器的值 预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率（即周期性的从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器REP_CNT到达0时，会产生一个更新事件并且计数器

		REP_CNT重新从REP值开始计数。由于REP_CNT只有在周期更新事件U_RC发生时才重载REP值，因此对TIMx_RCR寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在PWM模式中，（REP+1）对应着： – 在边沿对齐模式下，PWM周期的数目。 – 在中央对齐模式下，PWM半周期的数目。

16.4.14. TIM1 捕获/比较寄存器（TIM1_CCR1）

地址偏移：0x34

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR1[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR1[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR1[15:0]	捕获/比较通道1的值 若CC1通道配置为输出： CCR1决定了装入当前捕获/比较1寄存器的值（预装载值）。 如果在TIMx_CCMR1寄存器（OC1PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC1端口上输出信号。 若CC1通道配置为输出入： CCR1包含由上一次输入捕获1事件（IC1）传输的计数器值。

16.4.15. TIM1 捕获/比较寄存器 2（TIM1_CCR2）

地址偏移：0x38

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR2[15:8]							

类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR2[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR2[15:0]	捕获/比较通道1的值 若CC2通道配置为输出： CCR2决定了装入当前捕获/比较2寄存器的值（预装载值）。 如果在TIMx_CCMR2寄存器（OC2PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较2寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC2端口上输出信号。 若CC2通道配置为输出入： CCR2包含由上一次输入捕获2事件（IC2）传输的计数器值。

16.4.16. TIM1 捕获/比较寄存器 3（TIM1_CCR3）

地址偏移：0x3C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR3[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR3[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR3[15:0]	捕获/比较通道1的值 若CC3通道配置为输出： CCR3决定了装入当前捕获/比较3寄存器的值（预装载值）。 如果在TIMx_CCMR3寄存器（OC3PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较3寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC3端口上输出信号。 若CC3通道配置为输出入： CCR3包含由上一次输入捕获3事件（IC3）传输的计数器值。

16.4.17. TIM1 捕获/比较寄存器 4（TIM1_CCR4）

地址偏移：0x40

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR4[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR4[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR4[15:0]	捕获/比较通道1的值 若CC4通道配置为输出： CCR4决定了装入当前捕获/比较4寄存器的值（预装载值）。 如果在TIMx_CCMR4寄存器（OC4PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较4寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC4端口上输出信号。 若CC4通道配置为输出入： CCR4包含由上一次输入捕获4事件（IC4）传输的计数器值。

16.4.18. TIM1 刹车和死区寄存器（TIM1_BDTR）

地址偏移：0x44

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	DTG[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义

15	MOE	主输出使能 一旦刹车输入有效，该位被硬件异步清0。根据AOE位的设置值，该位可以由软件置1或被自动置1。它仅对配置为输出的通道有效。 0：禁止OCx和OCxN输出或强制为空闲状态； 1：如果设置了相应的使能位(TIM1_CCERx寄存器的CCxE位)，则使能OCx和OCxN输出。
14	AOE	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被自动置1(如果刹车输入无效)。 注意：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	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。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用
12	BKE	刹车功能使能 0：禁止刹车输入(BRK和内部刹车源)； 1：开启刹车输入(BRK和内部刹车源)。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。
11	OSSR	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，首先开启OCx/OCxN并输出无效电平，然后置OCx/OCxN使能输出信号=1。 注意：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	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，OCx/OCxN首先输出其空闲电平，然后OCx/OCxN使能输出信号=1。 注意：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	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位和TIMx_CR2寄存器的OISx/OISxN位； 10：锁定级别2，不能写入锁定级别1中的各位，也不能写入CC极性位(一旦相关通道通过CCxS位设为输出，CC极性位是TIMx_CCER寄存器的CCxP/CCxNP)以及OSSR/OSSI位； 11：锁定级别3，不能写入锁定级别2中的各位，也不能写入CC控制位(一旦相关通道通过CCxS位设为输出，CC控制位是TIMx_CCMR寄存器的

		OCxM/OCxPE位)； 注意：在系统复位后，只能写一次LOCK位，一旦写入TIMx_BDTR寄存器，则其内容保持不变直至复位。
7:0	DTG[7:0]	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}，其中：$t_{dtg}=t_{DTS}$ DTG[7:5]=10x => DT=(64+DTG[5:0]) * t_{dtg}，其中：$t_{dtg}= 2*t_{DTS}$ DTG[7:5]=110 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg=8* t_{DTS} DTG[7:5]=111 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg= 16*t_{DTS} 举列: 如果t_{DTS} =125 ns (8 MHz)，可能的死区时间为： DTG[7:0] = 0到7Fh，0到15875 ns ，步长时间为125 ns。 DTG[7:0] = 80h到BFh，16µs到31750ns，步长时间为250 ns。 DTG[7:0] = C0h到DFh，32µs到63µs，步长时间为1µs。 DTG[7:0] = E0h到FFh，64µs到126µs，步长时间为2 µs。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1、2或3，则不能修改这些位。

16.4.19. TIM1 DMA 控制寄存器（TIM1_DCR）

地址偏移：0x48

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			DBL[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
7:0	—			DBA[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:13	NA	保留位，未定义
12:8	DBL[4:0]	DMA突发传输长度 这5位定义了DMA在突发传输模式下的传输长度。（当对TIMx_DMAR寄存器进行读或写时，定时器则进行一次突发传输） 00000：1次传输 00001：2次传输 00010：3次传输 … 10001：18次传输
7:5	NA	保留位，未定义
4:0	DBA[4:0]	DMA基地址

		这5位定义了DMA传输的基地址（当对TIMx_DMAR寄存器进行读或写时）， DBA定义为TIMx_CR1寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR … 例：要完成如下的传输：DBL=7，DBA=TIMx_CR1 此时传输从TIMx_CR1的地址开始向连续7个寄存器进行操作。

16.4.20. TIM1 全部传输时 DMA 地址（TIM1_DMAR）

地址偏移：0x4C

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DMAB[15:0]	DMA突发传输寄存器 对TIMx_DMAR寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1地址）+（DBA + DMA索引）*4 其中： TIMx_CR1地址是控制器1（TIMx_CR1）所在的地址； DBA是TIMx_DCR寄存器中定义的基地址； DMA索引是由DMA自动控制的偏移量取决于TIMx_DCR寄存器中的DBL。

如何使用 DMA 突发传输的例子

本例中使用定时器 DMA 的突发传输功能，将 CCRx 寄存器（x=2，3，4）的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。

按如下步骤进行操作：

1. 配置相关的 DMA 通道：
   - DMA 通道设备地址为 DMAR 寄存器地址
   - DMA 通道存储器地址为包含要通过 DMA 传送到 CCRx 寄存器的数据 RAM 缓冲区地址
   - 传送数据数量=3
   - 循环模式禁止
2. 配置 DCR 寄存器中的 DBA 和 DBL 位：DBL=3 说明连续传输次数为 3 次；DBA=0xE，初始传输的偏移地址为 0x38（TIMx_CCR2）。
3. 使能 TIMx 更新 DMA 请求
4. 使能 TIMx
5. 使能 DMA 通道

注意：在本例中所有 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传输的数据数量应该为 6，而 RAM 缓冲区要包含 data1，data2，data3，data4，data5 和 data6。数据按如下过程被传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个更新 DMA 请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

17. 通用定时器（TIM3）

17.1. TIM3 简介

通用定时器由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。
它适合多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM）。
使用定时器预分频器和 RCC 时钟控制预分频器 ,可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。
通用定时器（TIMx）是完全独立的，它们不共享任何资源。它们可以同步操作，具体描述参考 17.3.15 章节的内容。

17.2. TIM3 主要特性

TIM3 时器的功能包括：

- 16 位向上、向下、向上/向下自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 如下事件发生时产生中断/DMA：
  - 更新事件：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 支持用于定位的增量编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

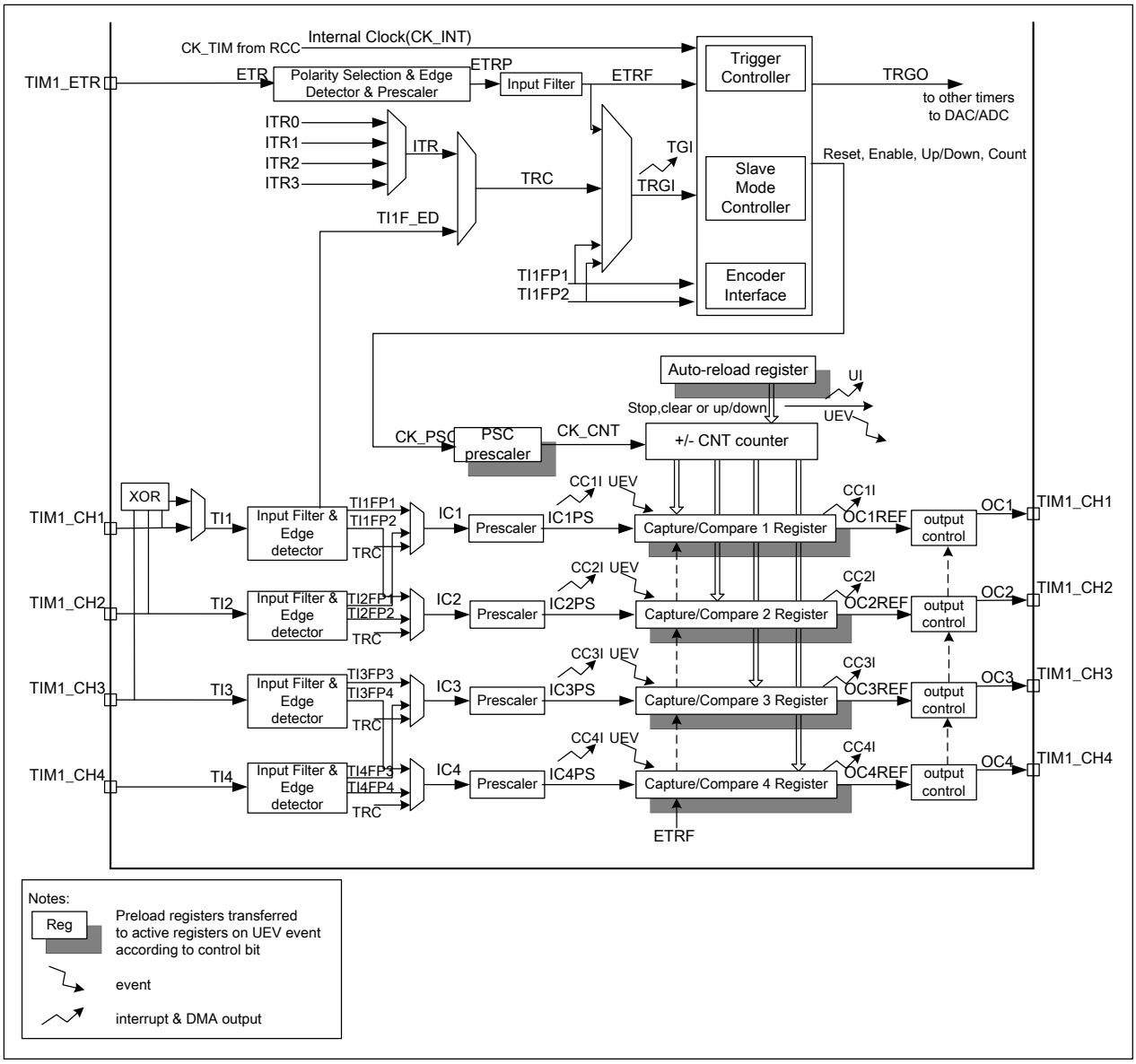


图 17.1 通用定时器框图（TIM3）

17.3. TIM3 功能描述

17.3.1. 时基单元

可编程通用定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

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

自动装载寄存器是预先装载的，写或读自动装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容会被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN) 时，CK_CNT 才有效。(更多有关使能计数器的细节，请参考控制器的从模式描述)

注意：在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

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。新的预分频器的参数在下一次更新事件到来时被采用。

图 17.2 和图 17.3 给出了在运行时更改预分频器因子，计数器的相关动作的例子。

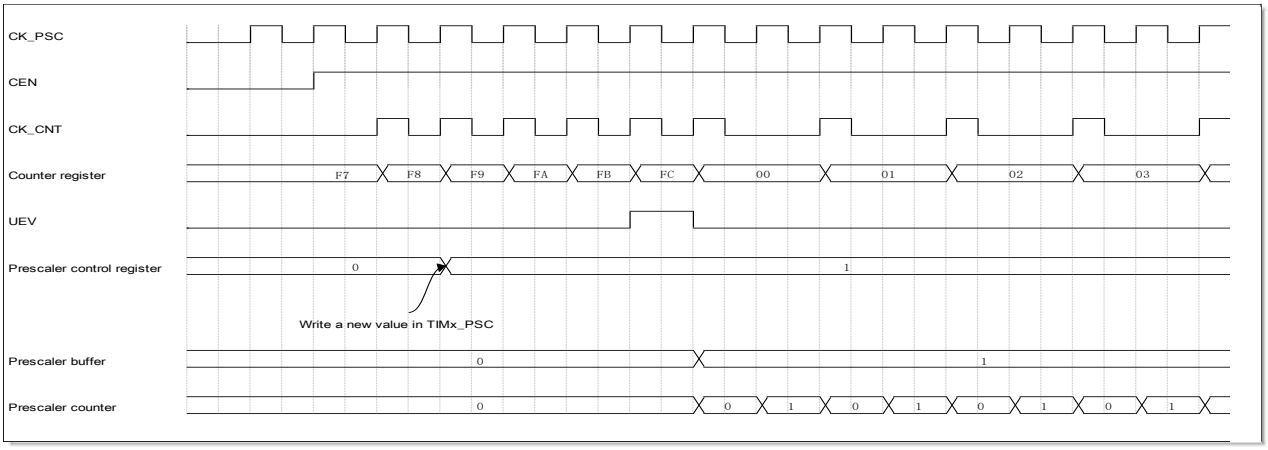


图 14-2 预分频系数从 1 变到 2 时，计数器的时序图

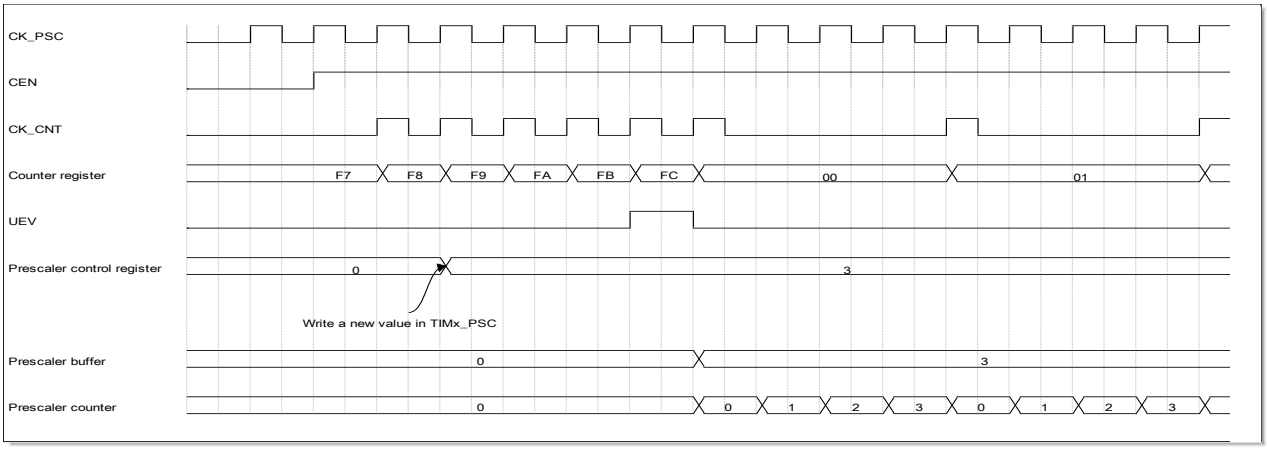


图 17.3 预分频系数从 1 变到 4 时，计数器的时序图

17.3.2. 计数器模式

向上计数模式

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

每次计数器溢出时可以产生更新事件，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。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

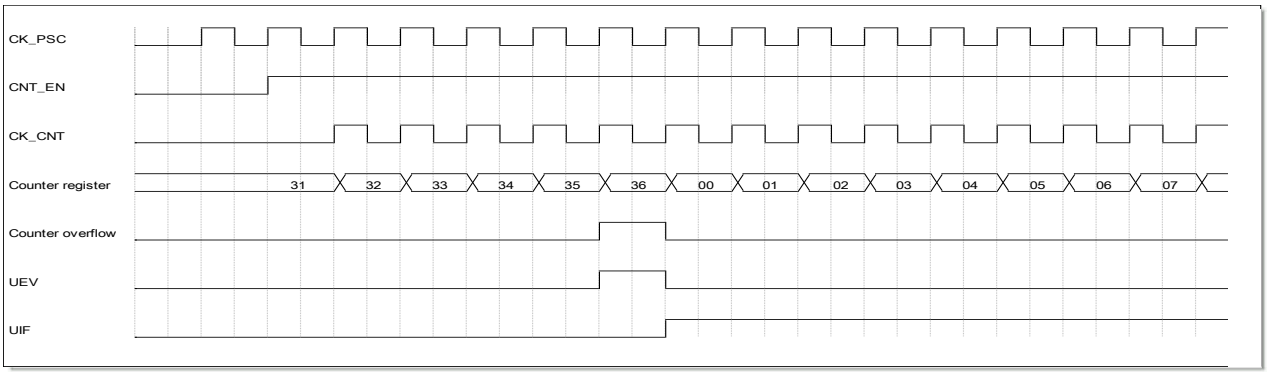


图 17.4 计数器时序图，内部时钟分频因子为 1

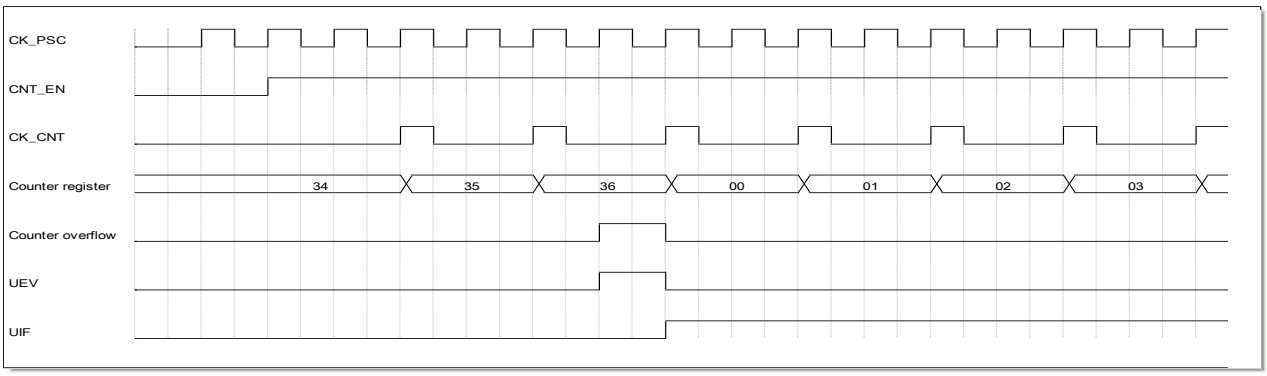


图 17.5 计数器时序图，内部时钟分频因子为 2

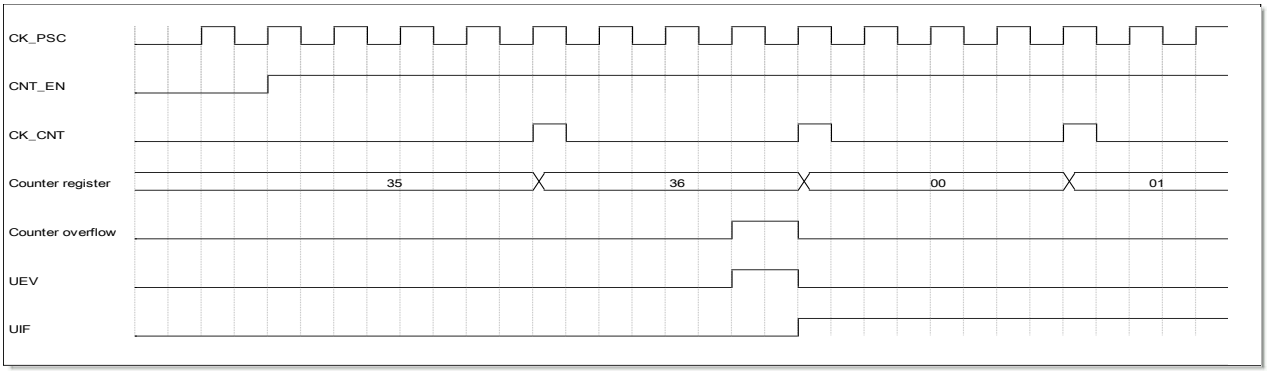


图 17.6 计时器时序图，内部时钟分频因子为 4

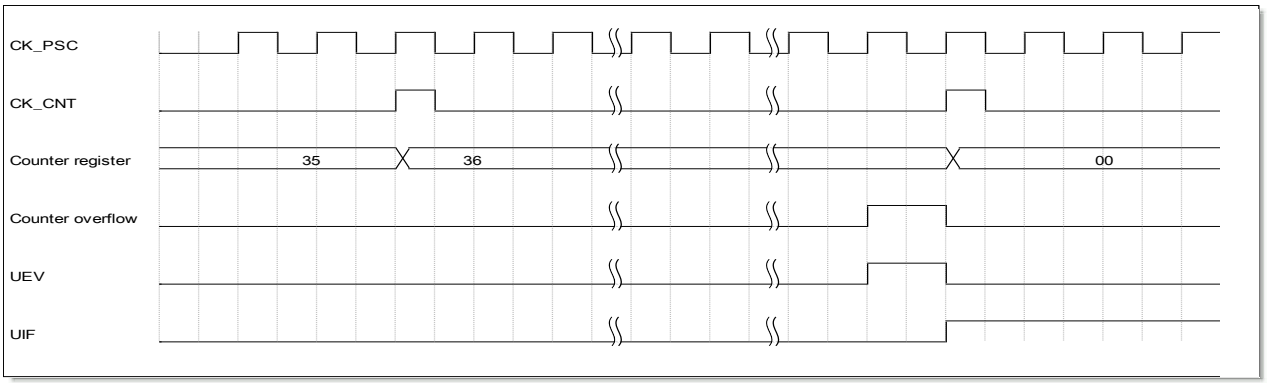


图 17.7 计数器时序图，内部时钟分频因子为 N

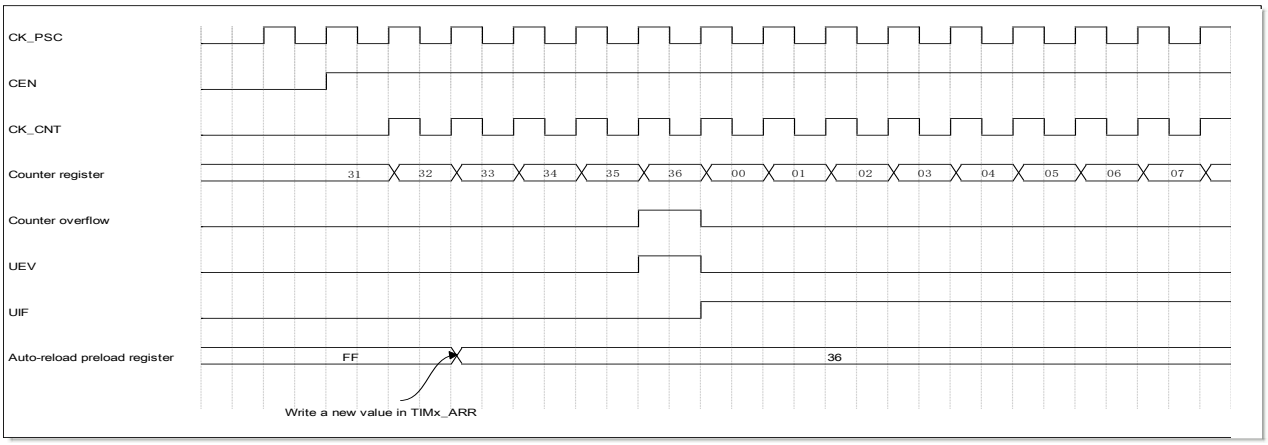


图 17.8 计数器时序图，当 ARPE=0 时的更新事件

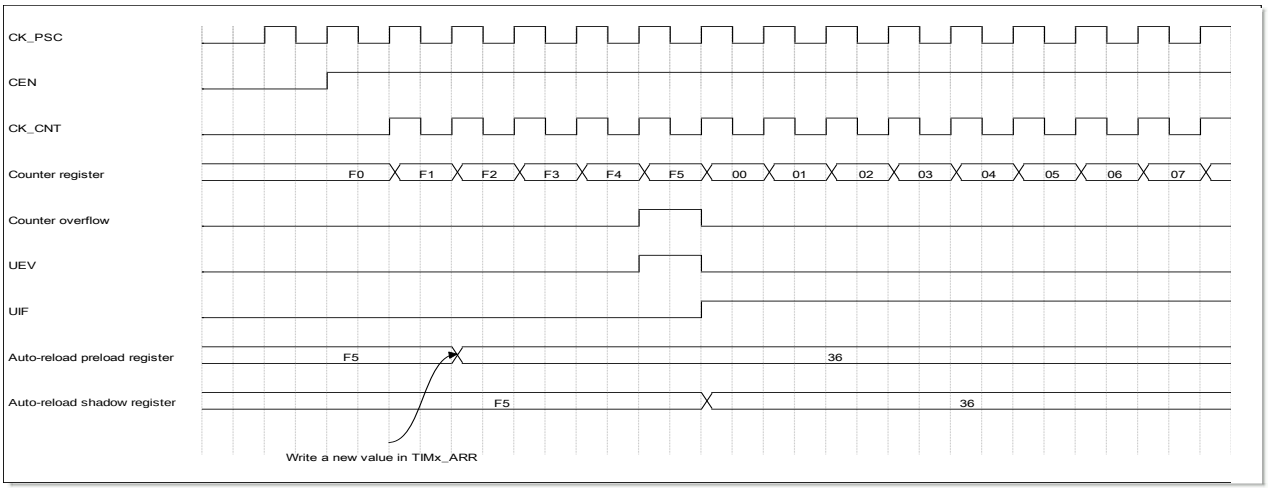


图 17.9 计数器时序图，当 ARPE=1 时的更新事件

向下计数模式

在向下计数模式中，计数器从自动装载值（TIMx_ARR 寄存器的值）开始向下计数到 0，然后又从自动装载值重新开始计数，并且产生一个计数器向下溢出事件。

每次计数器溢出时可以产生更新事件，设置 TIMx_EGR 寄存器的 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。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值；注意，如果在计数器重载之前自动装载寄存器被更新，此时在下一个周期才是预期的值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

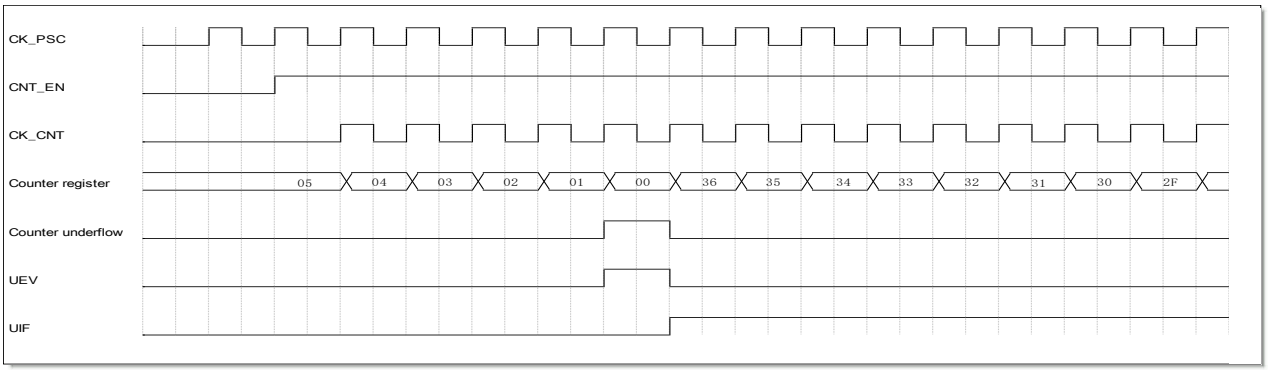


图 17.10 计数器时序图，内部时钟分频因子为 1

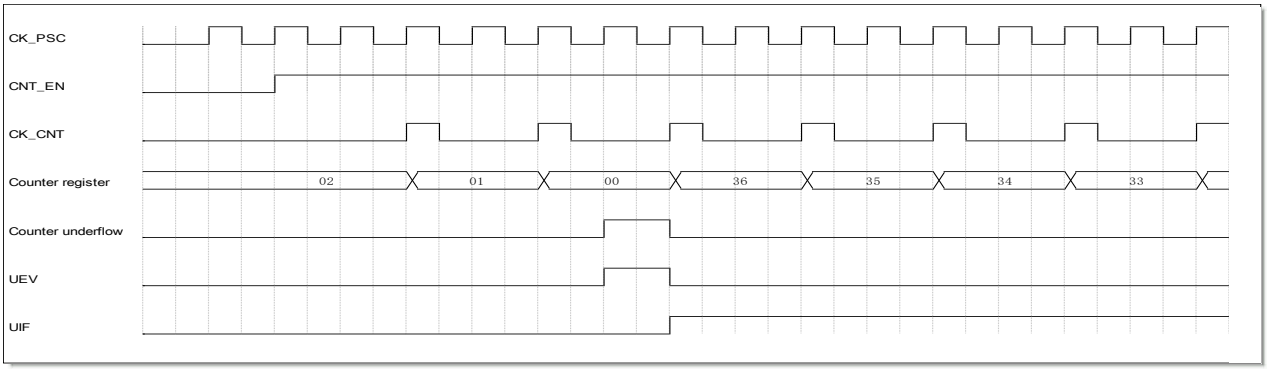


图 17.11 计数器时序图，内部时钟分频因子为 2

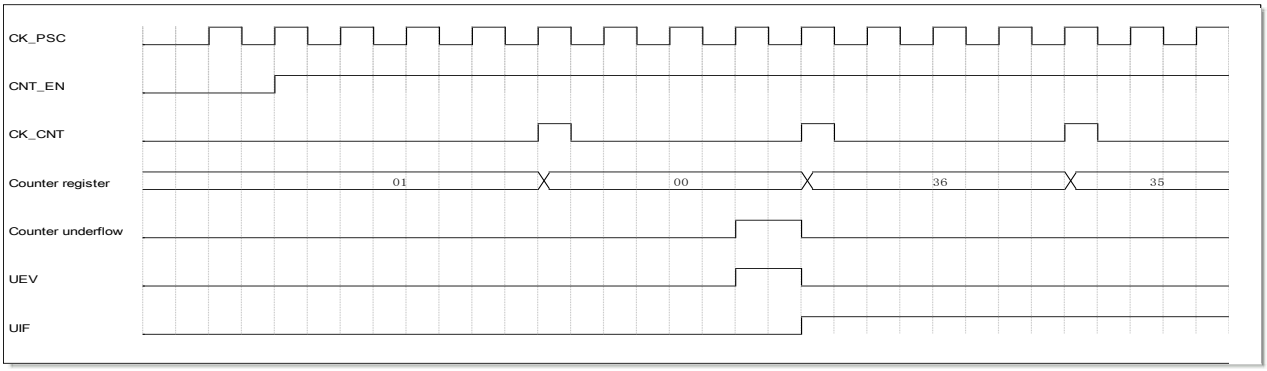


图 17.12 计数器时序图，内部时钟分频因子为 4

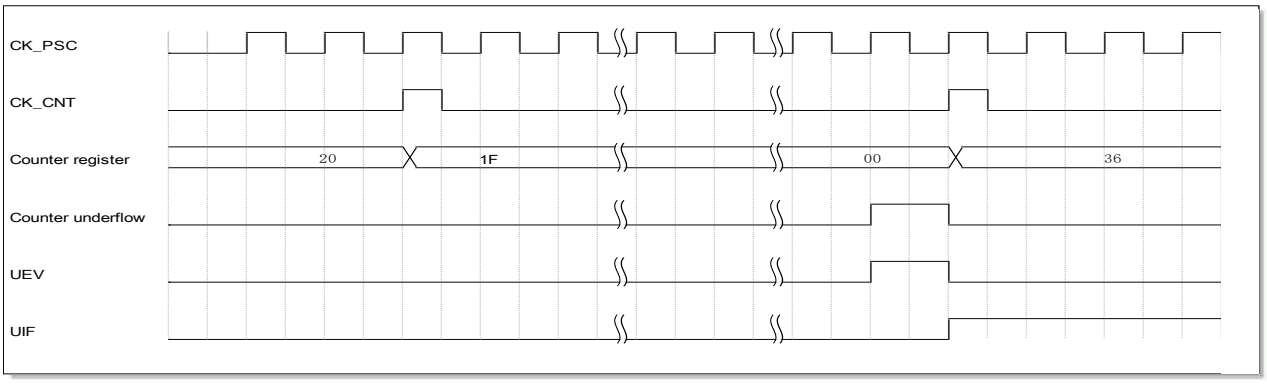


图 17.13 计数器时序图，内部时钟分频因子为 N

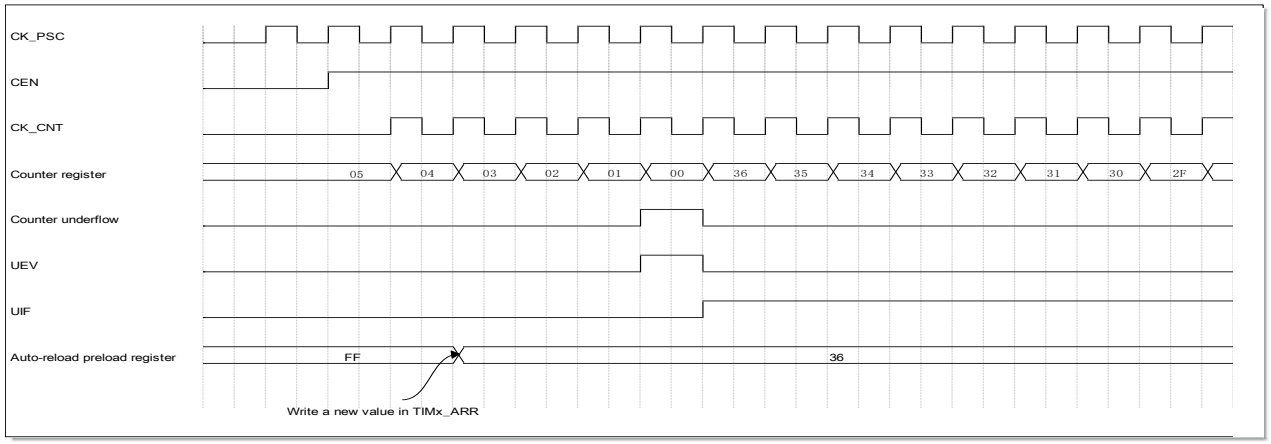


图 17.14 计数器时序图，没有使用重复计数器时的更新事件

中央对齐模式

在中央对齐模式下中，计数器从 0 开始计数到自动装载值（TIMx_ARR 寄存器）-1，产生一个计数器溢出事件，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。

当 TIMx_CR1 的 CMS 位不为 00 时，使能中央对齐模式。已配置为输出通道的输出比较中断标志在以下情况下被置位：计数器向下计数（中央对齐模式 1，CMS=01），计数器向上计数（中央对齐模式 2，CMS=10），计数器向下和向上计数（中央对齐模式 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。

在此模式下，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。此时，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。然而，计数器仍会根据当前自动装载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位（选择更新请求源）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值；注意，如果因为计数器上溢而产生更新，在计数器重载之前自动装载寄存器被更新，此时在下一个周期才是预期的值。

下面一些时序图展示了计数器在不同时钟频率下的计数情况。

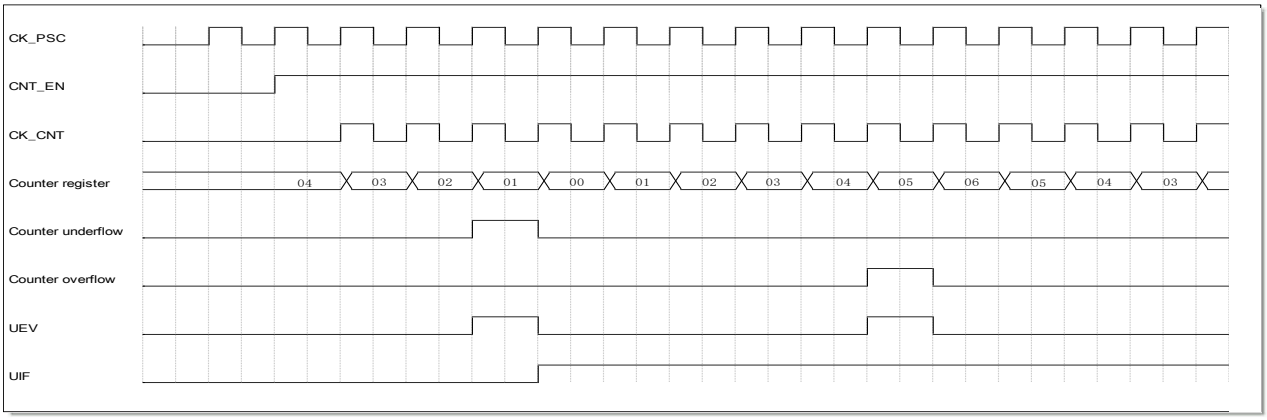


图 17.15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

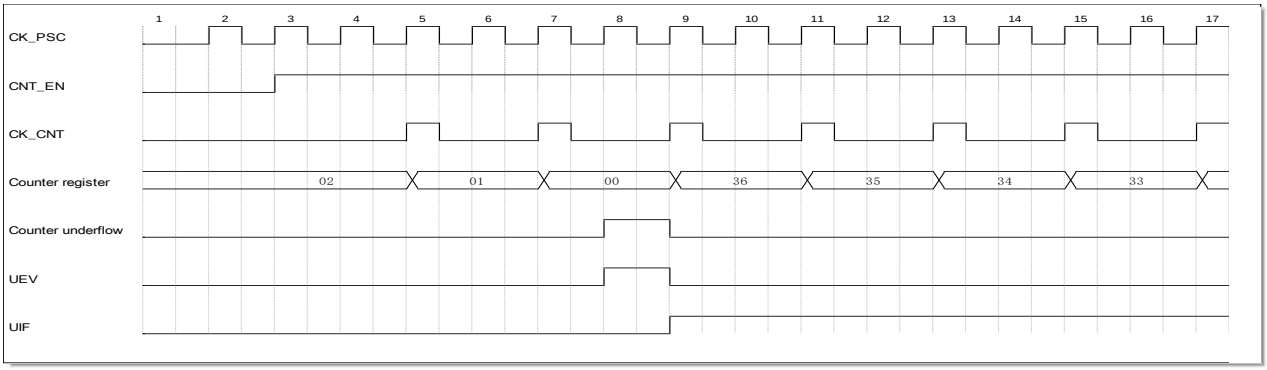


图 17.16 计数器时序图，内部时钟分频因子为 2

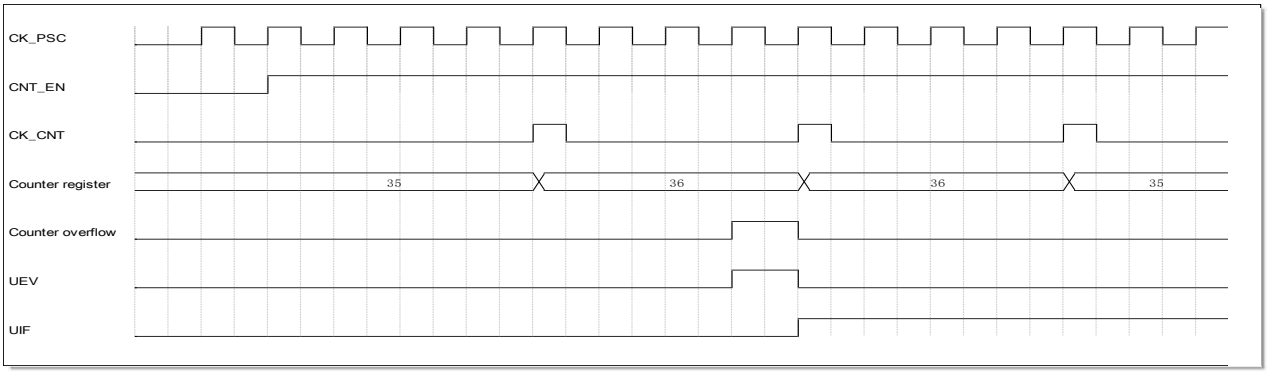


图 17.17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36

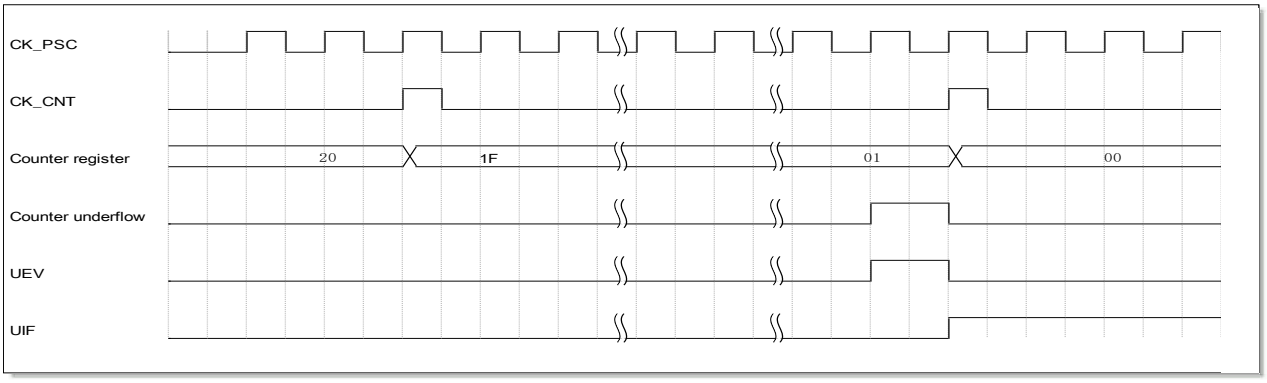


图 17.18 计数器时序图，内部时钟分频因子为 N

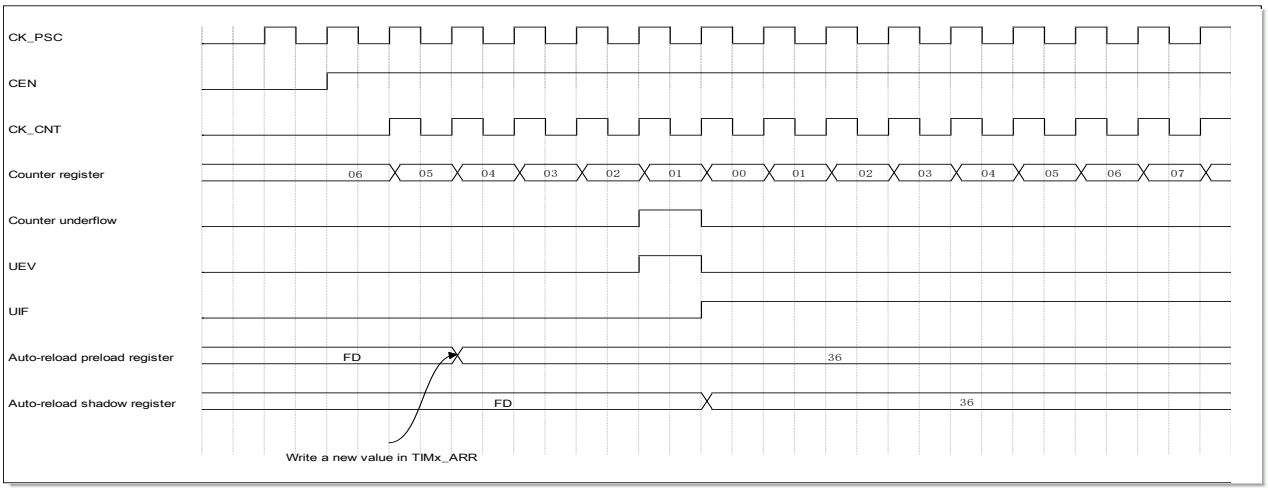


图 17.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

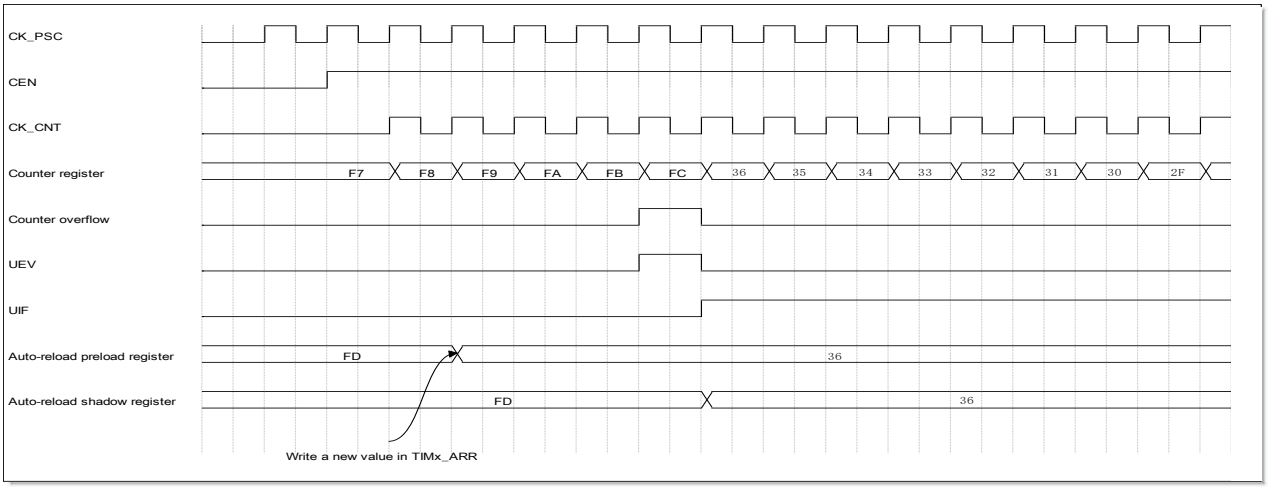


图 17.20 计数器时序图，ARPE=1 时的更新事件（计数器上溢）

17.3.3. 时钟源

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置定时器 TIM1 作为定时器 TIM2 的预分频器。

内部时钟源（CK_INT）

如果禁止了从模式控制器（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是实际上的控制位，并且只能被软件修改（除非 UG 位自动被清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示了在不带预分频器时，控制电路和向上计数器在正常模式下的动作。

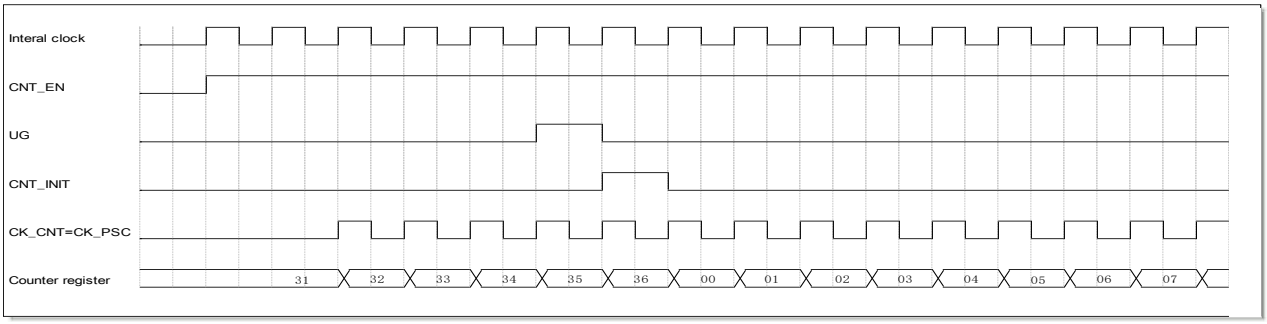


图 17.21 内部时钟分频是 1 时正常模式下的控制时序图

外部时钟源模式 1

当 TIMx_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器在选定输入端的每个上升沿或下降沿计数。

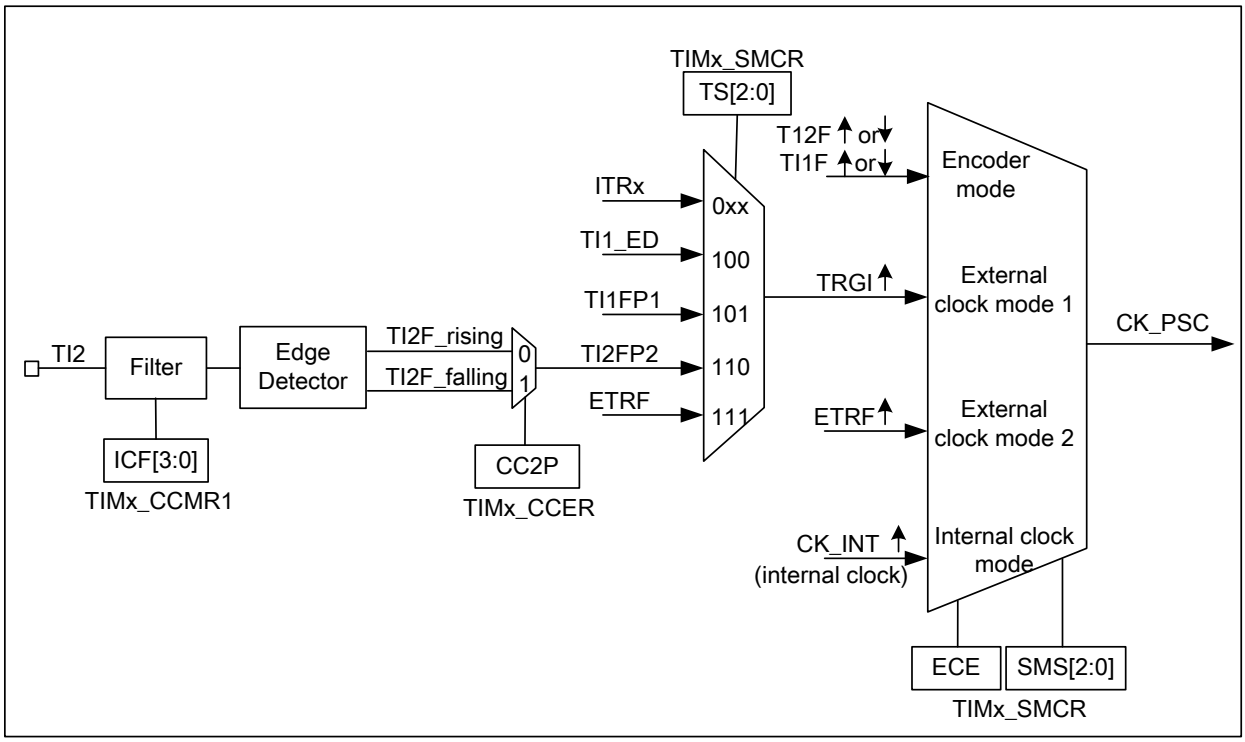


图 17.22 TI2 外部时钟连接示例

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

7. 写 TIMx_CCMR1 寄存器的 CC2S=01，配置通道 2 检测 TI2 输入的上升沿。
8. 写 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）。
9. 写 TIMx_CCER 寄存器的 CC2P=0，选择上升沿极性。
10. 写 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1。
11. 写 TIMx_SMCR 寄存器的 TS=110，选择 TI2 作为触发输入源。
12. 写 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注意： 捕获预分频器不用作触发，所以不需要对它进行配置。

当 TI2 上升沿出现时，计数器计数一次且 TIF 标志被置位。

当 TI2 的上升沿和计数器实际时钟之间的延时取决于在 TI2 输入端的重新同步电路。

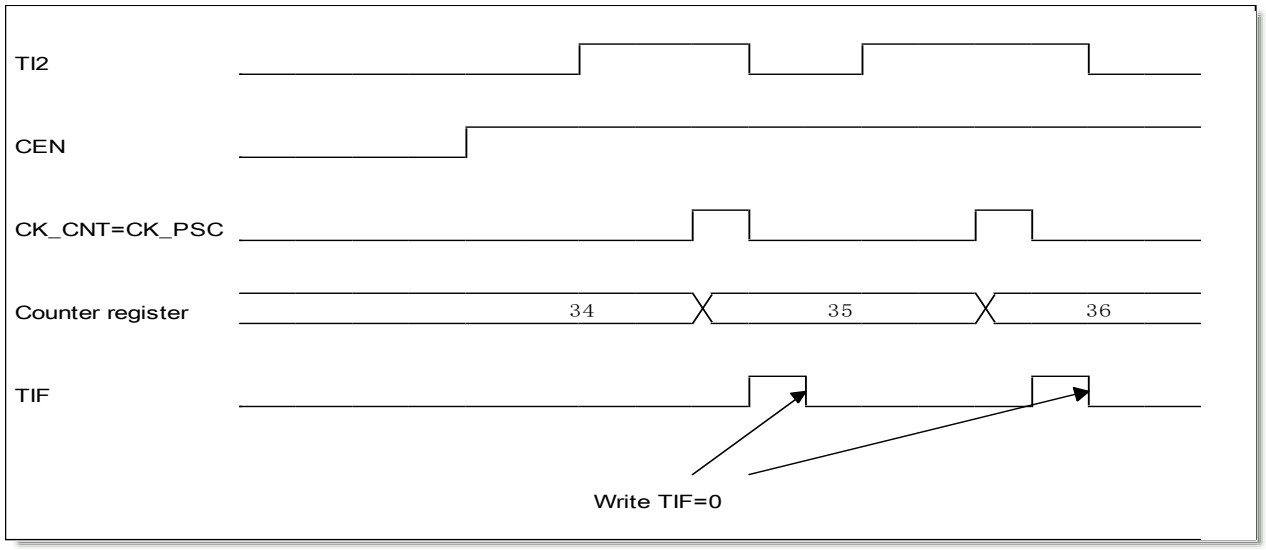


图 17.23 外部时钟模式 1 时的控制时序图

外部时钟源模式 2

当 TIMx_SMCR 寄存器的 ECE=1 时，此模式被选中。计数器在外部触发 ETR 的每个上升沿或下降沿计数。

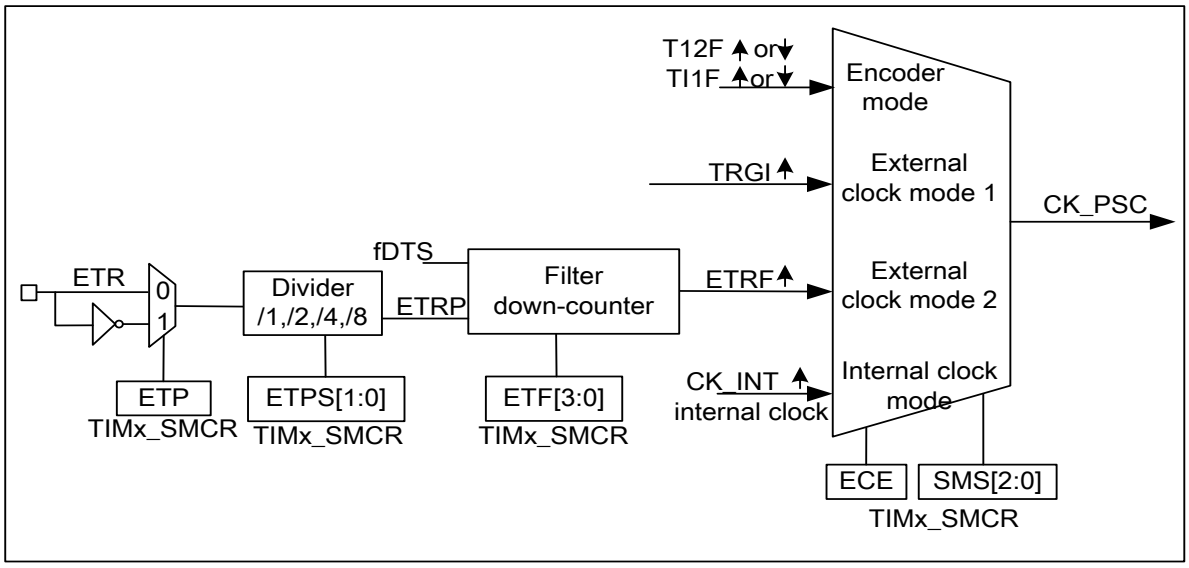


图 17.24 外部触发输入框图

例如，要配置向上计数器在 ETR 的每 2 个上升沿计数一次，使用下列步骤：

6. 本例中不需要滤波器，写 TIMx_SMCR 寄存器中的 ETF[3:0]=0000。
7. 写 TIMx_SMCR 寄存器中的 ETPS[1:0]=01，设置预分频器分频比为 2。
8. 写 TIMx_SMCR 寄存器中的 ETP=0，选择检测 ETR 的上升沿。
9. 写 TIMx_SMCR 寄存器中的 ECE=1，开启外部时钟模式 2。
10. 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。

计数器在每 2 个 ETR 上升沿计数一次。

ETR 的上升沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。

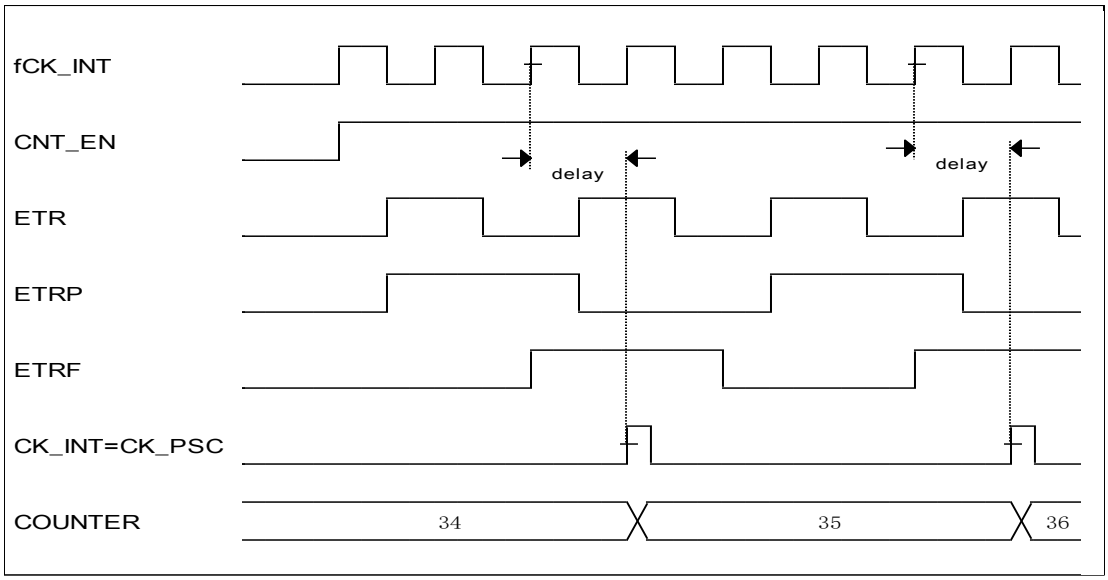


图 17.25 外部时钟模式 2 时的控制时序图

17.3.4. 捕获/比较通道

每一个捕获/比较通道都是包括一个捕获/比较寄存器（包含影子寄存器），一个捕获的输入部分（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）。

输入部分采样相应的 TIx 输入信号，产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿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。该信号紧接着被预分频产生信号 IC1PS。

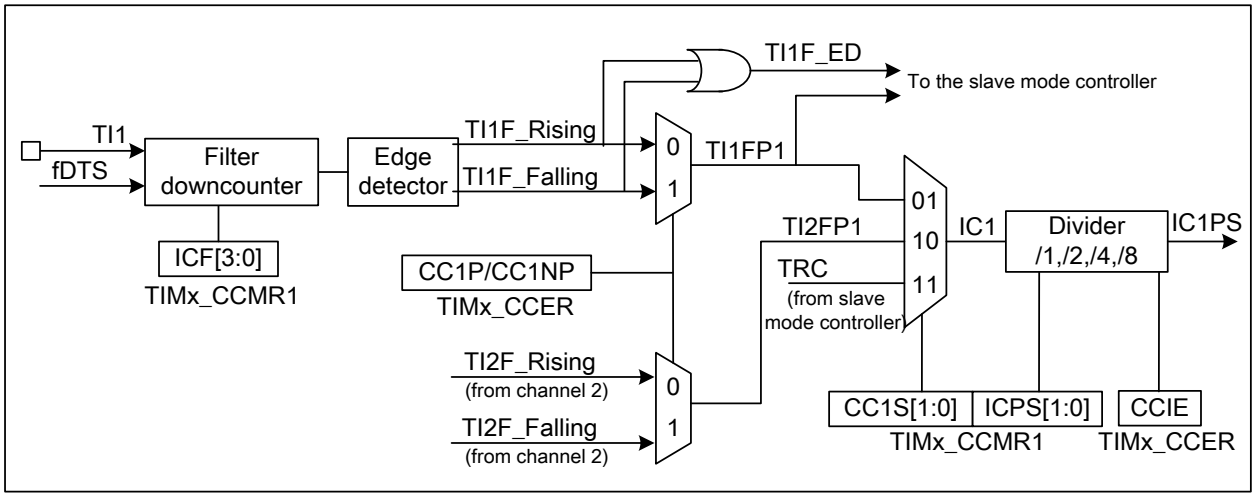


图 17.26 捕获/比较通道（如：通道 1 输入部分）

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。

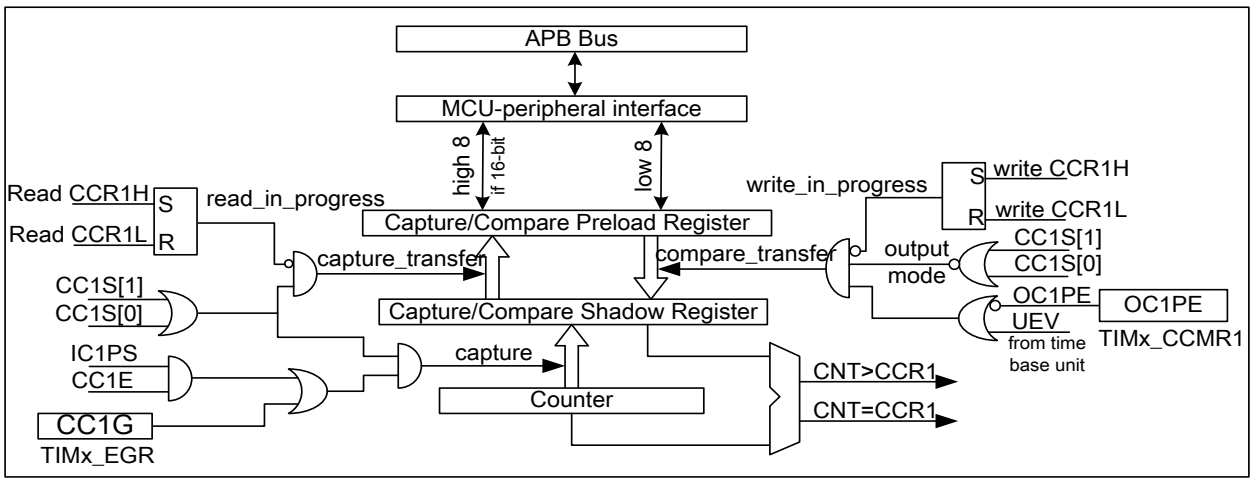


图 17.27 捕获/比较通道 1 主要框图

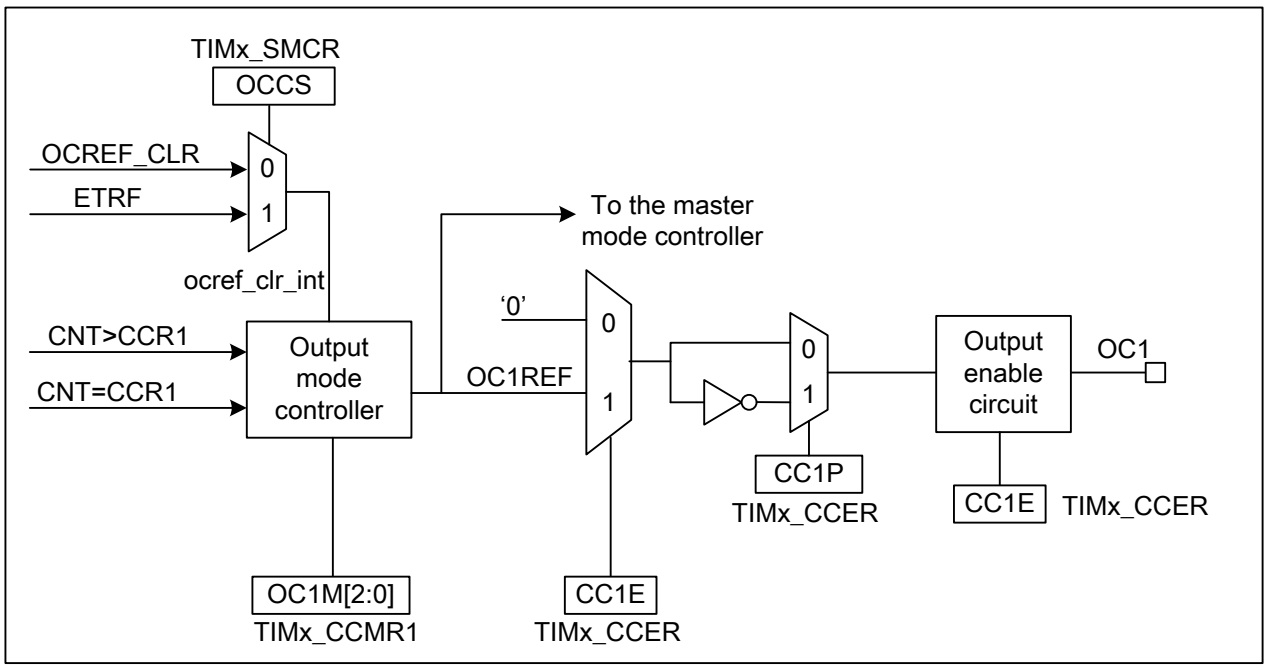


图 17.28 捕获/比较通道的输出部分（通道 1）

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。

17.3.5. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿跳变时，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIMx_SR 寄存器）被置 1，如果使能了中断或者 DMA 则产生一个相应的中断或者一个相应的 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMx_SR 寄存器）被置 1。软件写写 CCxIF=0 可清楚 CCxIF，或者读取存储在 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：

- 选择有效输入端 :TIMx_CCR1 必须连接到 TI1 输入 ,所以需要写 TIMx_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为 0，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
- 根据连接到计数器的输入信号特点，配置输入滤波器为所需的带宽（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 频率采样）连续采样 8 次，以确认在 TI1 上一次新的边沿变换。然后在 TIMx_CCMR1 寄存器中写 IC1F=0011。
- 配置输入预分频器。在本例中，希望在每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
- 写 TIMx_CCER 寄存器的 CCIE=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 标志被置位。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如果设置了 CC1IE 位，则会产生一个中断。
- 如果设置了 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。

注意：通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断或 DMA 请求。

17.3.6. 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 信号被映射至同一个 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 信号被作为处罚输入信号，而从模式控制器被配置成复位模式。

例如 ,你需要测量输入到 TI1 上的 PWM 信号的周期(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器), 具体步骤如下（取决于 CK_INT 的频率和预分频器的值）:

- 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 寄存器中的 CC2S=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TIMx_CCR2 中）: 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（选中 TI1FP1）。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 寄存器中的 SMS=100。

- 使能捕获：置 TIMx_CCER 寄存器中的 CC1E=1 且 CC2E=1。

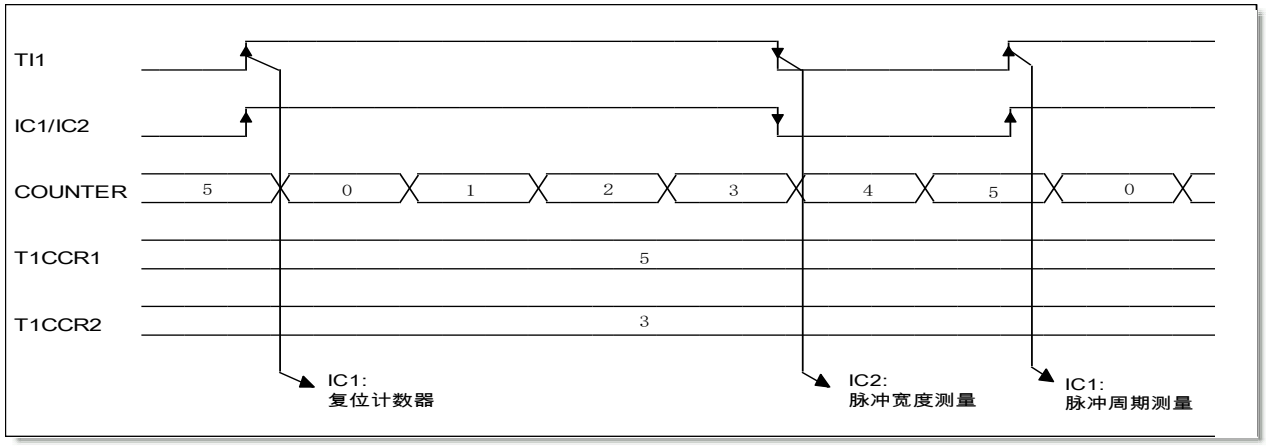


图 17.29 PWM 输入模式时序图

17.3.7. 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强制为有效或者无效状态，而不依赖于输出比较寄存器和计数器之间的比较结果。

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强制输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强制为高电平（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 被强制为高电平。

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

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

17.3.8. 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=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 ）。
- 置位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。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也可以用来输出一个单脉冲。

输出比较模式的配置步骤：

6. 选择计数器时钟（内部、外部、预分频器）。
7. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
8. 如果要产生一个中断请求，设置 CCxIE 位。
9. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   - 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
10. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

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

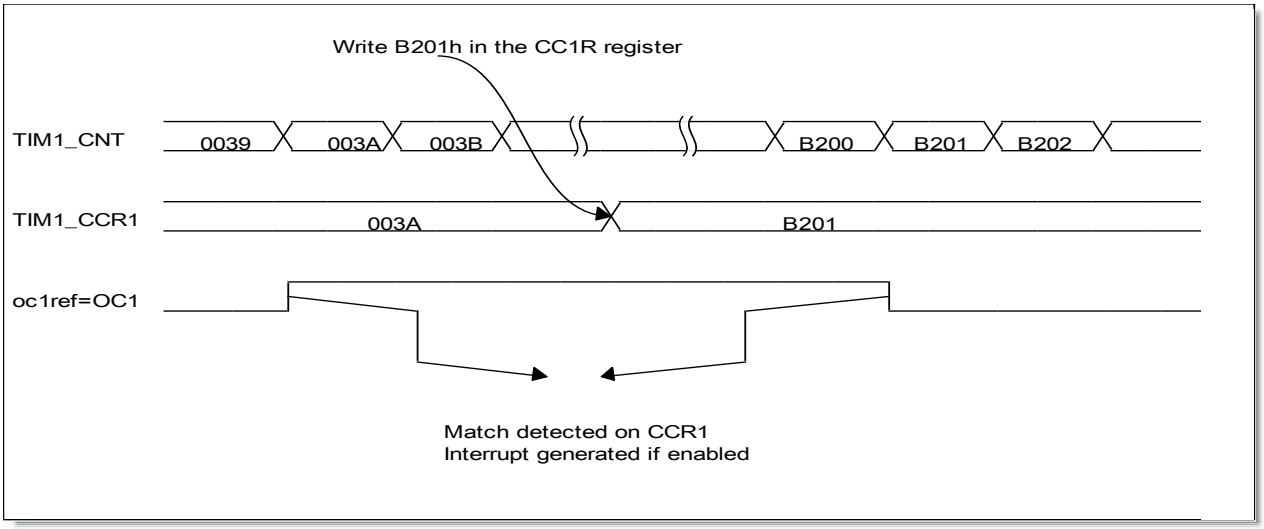


图 17.30 输出比较模式，翻转 OC1

17.3.9. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率，由 TIMx_CCRx 寄存器确定占空比的信号。在 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)

以确定是否符合 TIMx_CCRx<=TIMx_CNT 或者 TIMx_CNT<=TIMx_CCRx。根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值(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。

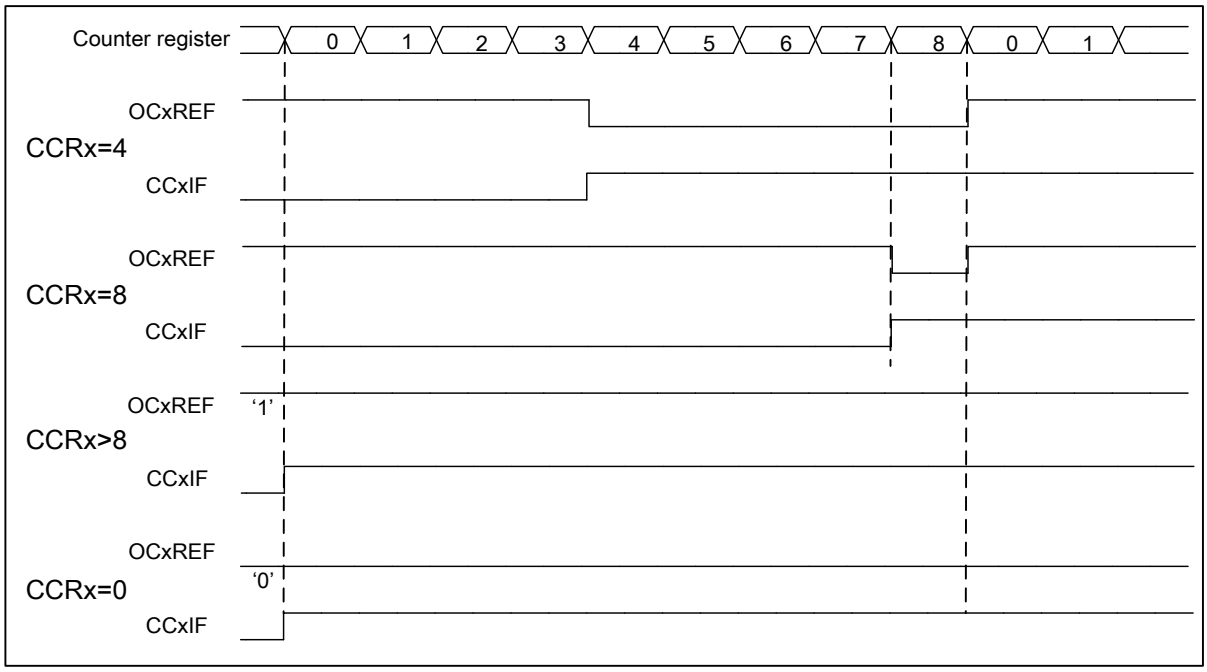


图 17.31 边沿对齐的 PWM 波形（ARR=8）

- 向下计数配置

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

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为 0 时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时，在计数器向下计数时，或在计数器向上和向下计数器时被置 1。TIMx_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。
下图给出了一些中央对齐的 PWM 波形的例子

- TIMx_ARR=8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

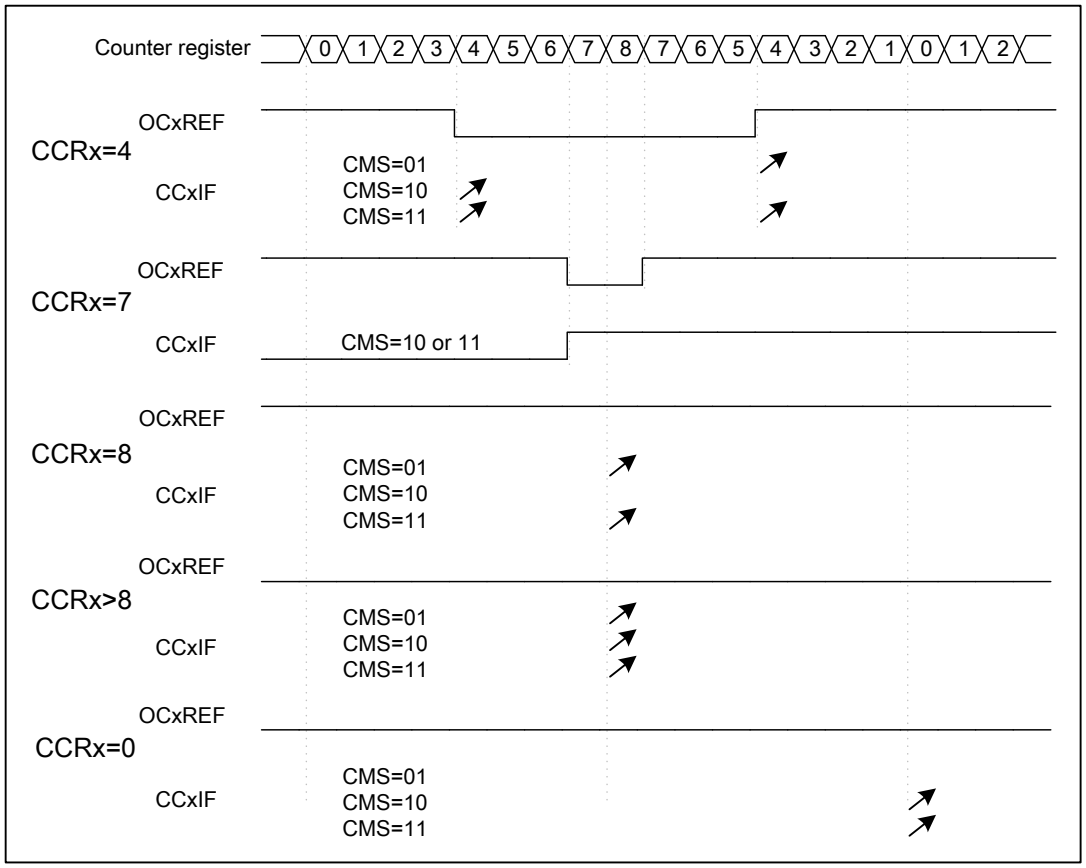


图 17.32 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

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

17.3.10. 单脉冲模式

单脉冲模式（OPM）是前述众多模式中的一个特例。这种模式允许计数器通过响应一个激励来启动，并在一个程序可控的延时之后产生一个脉宽程序可控的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器在产生下一个更新事件 UEV 时自动停止。

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

- 在向上计数模式下，计数器 CNT<CCRx<=ARR（特别的，0<CCRx）。

- 在向下计数模式下，计数器 CNT>CCRx。

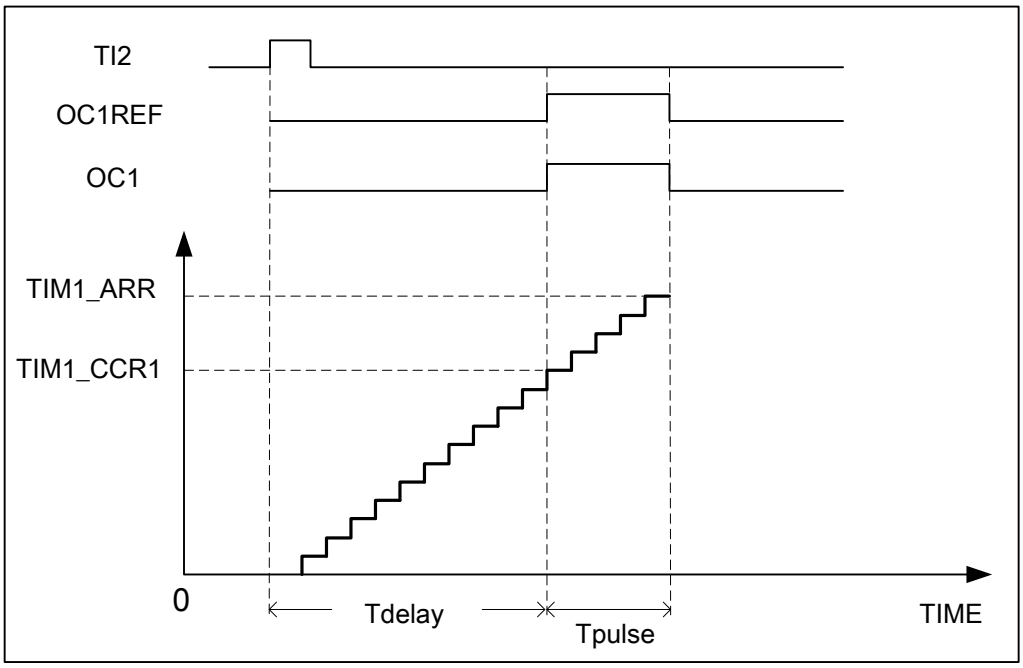


图 17.33 单脉冲模式示意图

例如 ,需要从 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。假设 TI2FP2 作为触发 1：

- 写 TIMx_CCMR1 寄存器中的 CC2S=01，将 TI2FP2 映像到 TI2 上。
- 写 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 写 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
- 写 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 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换；首先要写 TIMx_CCMR1 寄存器的 OC1M=1111，选择 PWM 模式 2；根据需要有选择的使能预装载寄存器，写 TIMx_CCMR1 寄存器中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；此时必须向 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动装载值，设置 UG 位来产生一个更新事件，然后等待 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。当 TIMx_CR1 寄存器中的 OPM=0，进入重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但这些操作需要一定的时钟周期，因此限制了可得到的最小延时T_{delay}。

如果需要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF 直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM 模式 1 和 PWM

模式 2 时起作用。

17.3.11. 外部事件时清除 OCxREF 信号

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

4. 外部触发预分频必须关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。
5. 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。
6. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要进行配置。

下图显示了当 ETRF 输入变为高时，对应不同的 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。

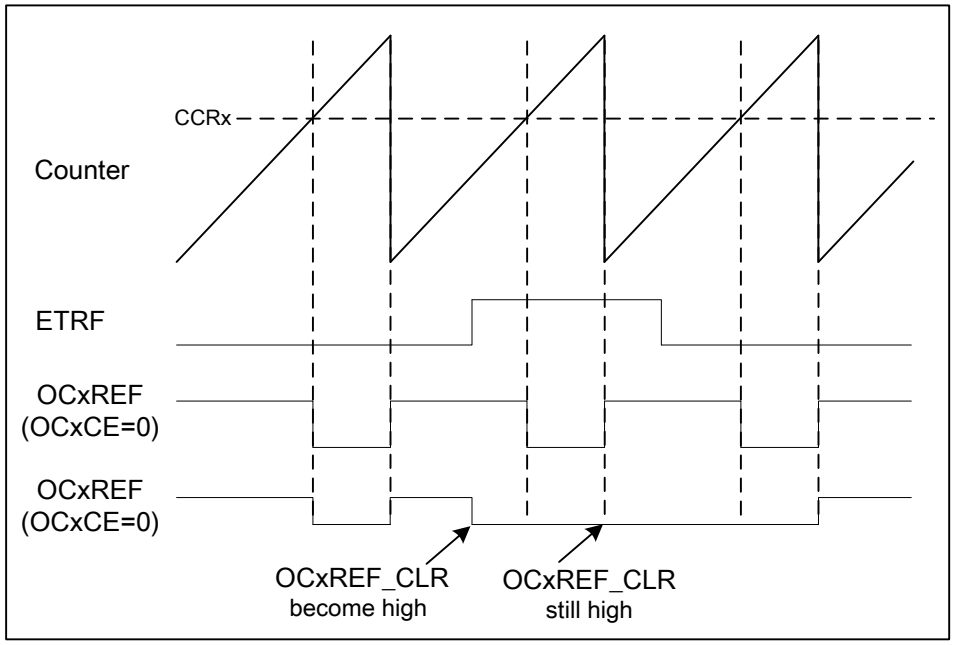


图 17.34 清除 OCxREF 的时序图

注意：在 100%的 PWM 时（如果 CCRx>ARR），OCxREF 在下一次计数溢出时被再次使能。

17.3.12. 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 TI2 的边沿计数，则写 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则写 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则写 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进行配置。CC1NP 和 CC2NP 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口，参看表格 17.1。假定计数器已经启动（TIMx_CR1 寄存器中的 CEN=1），则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 再通过输入滤波器和极性控制后产生的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位修改。不管计数器是依靠 TI1 还是 TI2 或者同时依靠 TI1 和 TI2 计数，在任一输入端的跳变都会重新计算 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、重复计数器、触发输出特性等仍能正常工作。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，计数器增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 17.1 计数方向与编码器信号的关系

有效计数沿	相对信号的电平值 (TI1FP1 for TI2) (TI2FP2 for TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
TI1	高	向下	向上	——	——
	低	向上	向下	——	——
TI2	高	——	——	向上	向下
	低	——	——	向下	向上
TI1 or TI2	高	向下	向上	向上	向下
	低	向上	向下	向下	向上

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰的能力。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，TI1FP1 映射到 TI1）。
- CC2S=01（TIMx_CCMR2 寄存器，TI2FP2 映射到 TI2）。
- CC1P=0（TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1）。
- CC2P=0（TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2=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）。

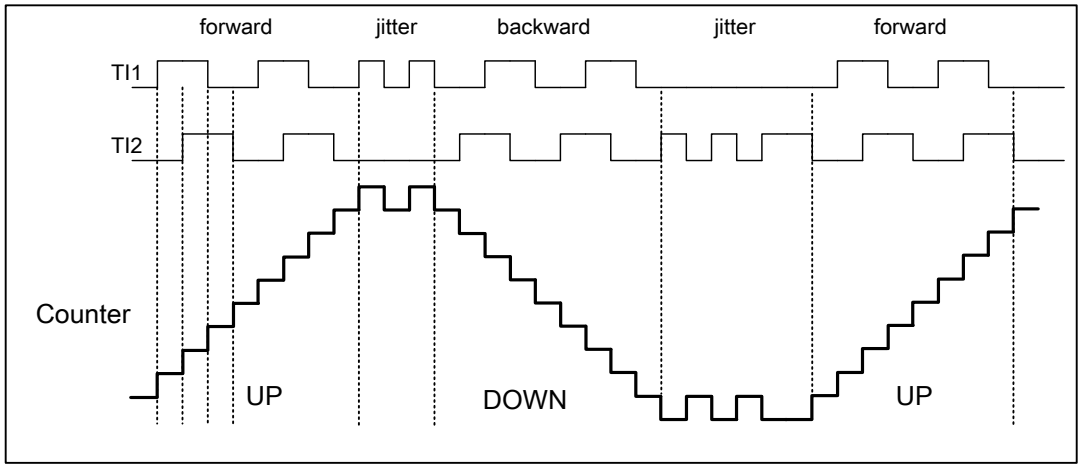


图 17.35 编码器模式下的计数器操作

下图为当 TI1FP1 极性反相时计数器的操作实例（CC1P=1，其他配置与上例相同）

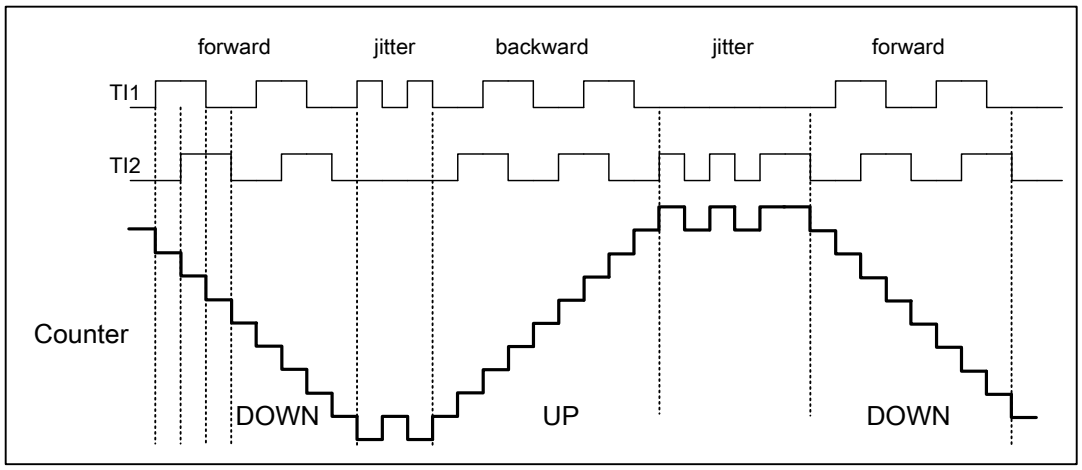


图 17.36 编码器模式下的计数器操作（TI1FP1 反相）

当定时器配置成编码器接口模式时，用于指示传感器当前位置。配合使用第二个定时器配置为捕获模式，通过测量两个编码器事件的间隔，从而获得动态信息（速度、加速度、负加速度）。编码器输出还可用来指示机械零点。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果有必要，可以把计数器的值锁存到第三个输入捕获寄存器（由另一个定时器产生的捕获信号必须是周期性）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

17.3.13. 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或端的 4 个输入端为 TIMx_CH1、TIMx_CH2、TIMx_CH3 和 TIMx_CH4。
异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

17.3.14. TIMx 定时器和外部触发的同步

TIMx 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。

从模式：复位模式

在一个触发输入事件发生时，计数器和它的预分频器被重新初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIMx_ARR、TIMx_CCRx）都被更新。
在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中的 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）。
- 写 TIMx_SMCR 寄存器中的 SMS=100，配置定时器为复位模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

- 写 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，并根据 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 上升沿河计数器的实际复位之间的延时取决于 TI1 输入端的再同步电路。

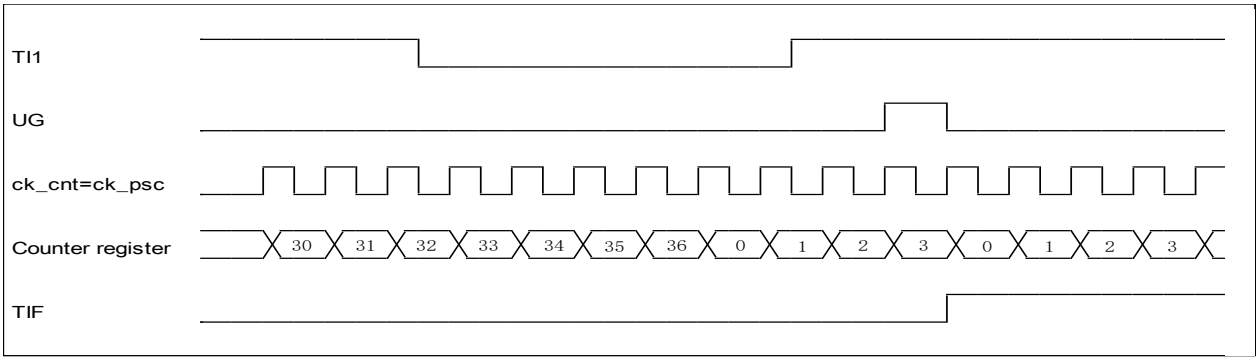


图 17.37 复位模式下的控制时序图

从模式：门控模式

可以按照选中的输入端电平使能计数器。

在如下的例子中，计数器只能在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波器，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，写 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 写 TIMx_SMCR 寄存器中的 SMS=101，配置定时器为门控模式；写 TIMx_SMCR 寄存器中的 TS=101，选择 TI1 作为输入源。
- 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都会将 TIMx_SR 寄存器中的 TIF 标志置位。

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

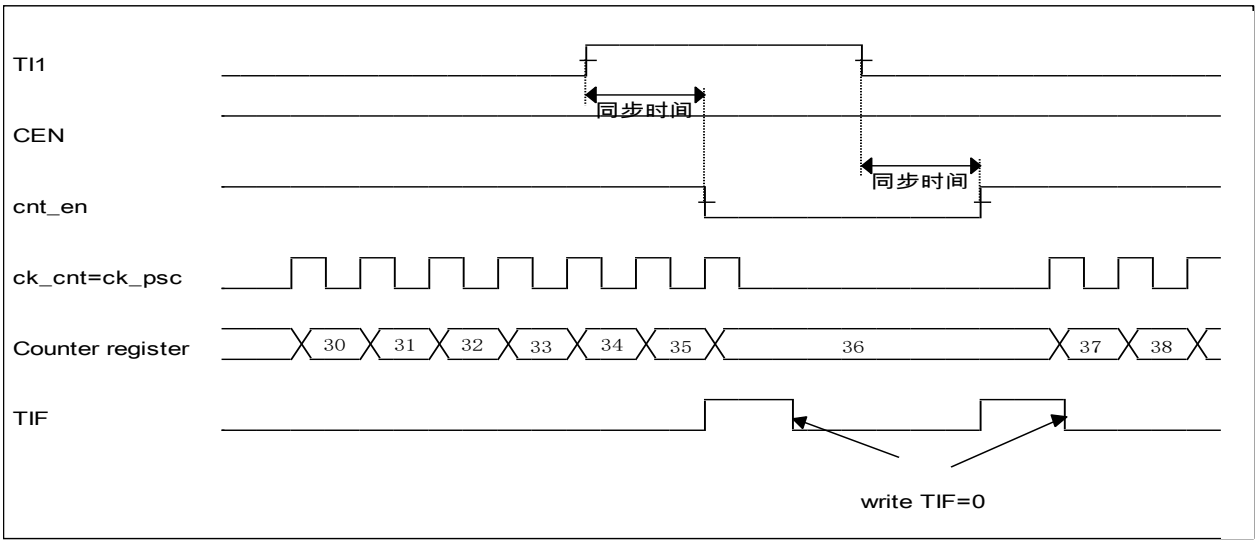


图 17.38 门控模式下的控制时序图

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 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，写 TIMx_CCMR1 寄存器中的 CC2S=01。写 TIMx_CCER 寄存器中的 CC2P=1 和 CC2NP=0 以确定极性（只检测低电平）。
- 写 TIMx_SMCR 寄存器中的 SMS=110，配置定时器为触发模式；写 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，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 输入端的再同步电路。

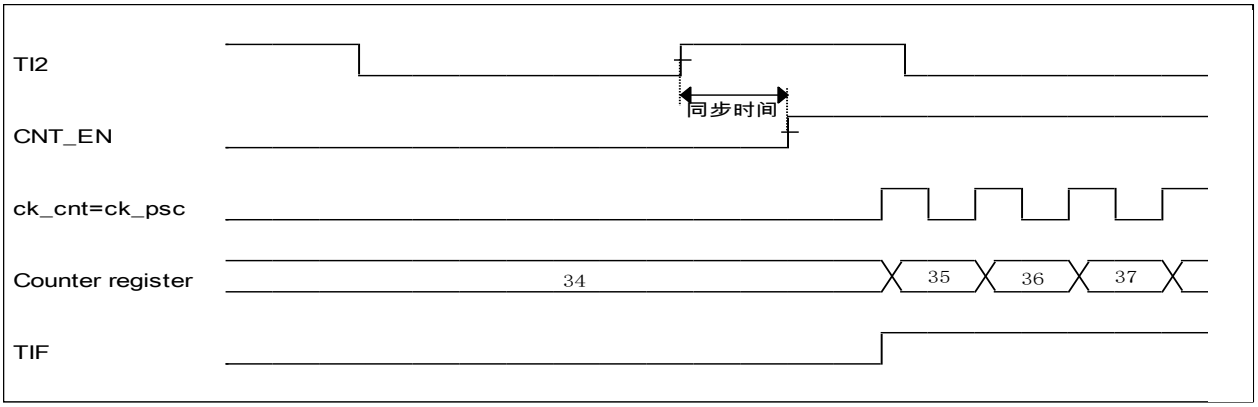


图 17.39 触发模式下的控制时序图

从模式：外部时钟模式 2 +触发模式

外部时钟模式 2 可以与另一个从模式（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，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 TIMx_SMCR 寄存器中的 TS 位来选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，向上计数器在 ETR 的每一个上升沿加 1：

4. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：
   - ETF=0000：没有滤波
   - ETPS=00：不用预分频器
   - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2
5. 按下列方式配置通道 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
   - 写 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）
6. 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。写 TIMx_SMCR 寄存器中的 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被置位，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之间的延时取决于 ETRP 输入端的再同步电路。

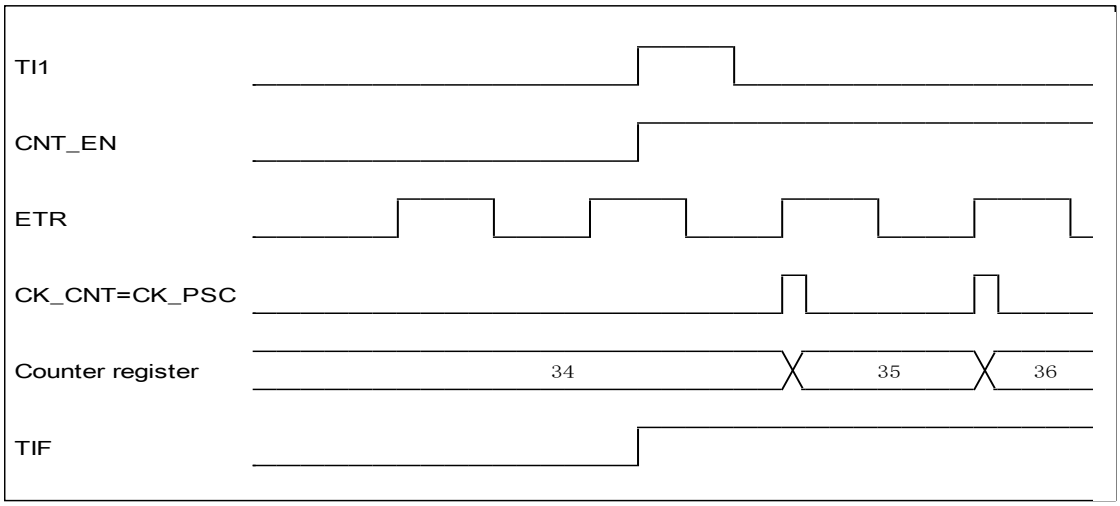


图 17.39 外部时钟模式 2 + 触发模式下的控制时序图

17.3.15. 定时器同步

所有 TIMx 定时器在内部相连，用于定时器同步或链接。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、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况。

使用一个定时器作为另一个定时器的预分频器

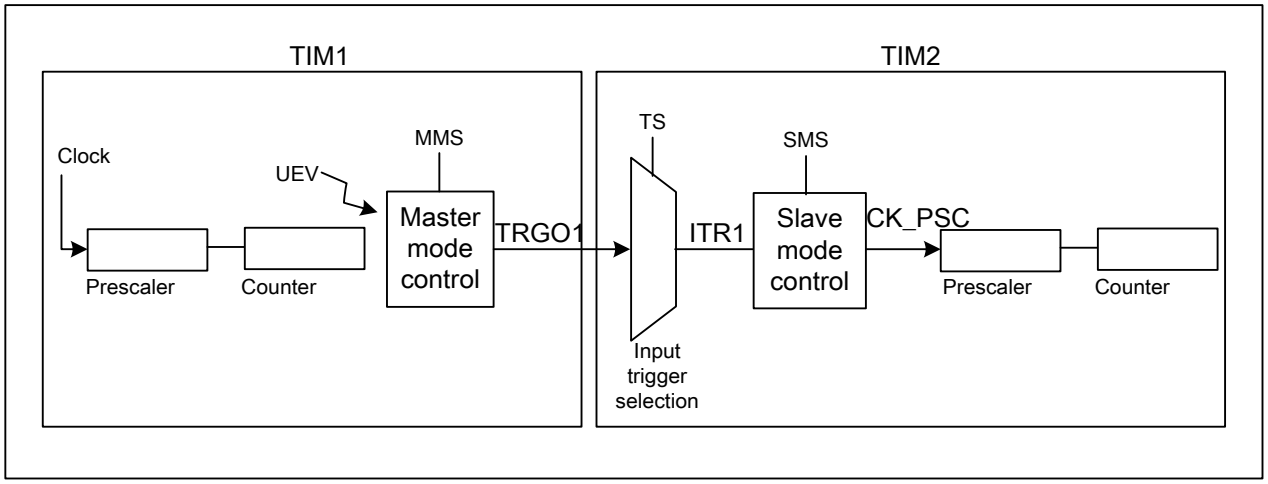


图 17.40 主/从定时器的链接

例如，可以配置定时器 1 作为定时器 3 的预分频器，参考图 17.40，进行下述操作：

- 配置定时器 1 为主模式，它可以再每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM1_CR2 寄存器的 MMS=010 时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。
- 连接定时器 1 的 TRGO1 输出至定时器 3 ,设置 TIM3_SMCR 寄存器的 TS=000 ,配置定时器 3 为使用 ITR0 作为内部触发的从模式。
- 然后把从模式控制器设置成外部时钟模式 1（TIM3_SMCR 寄存器的 SMS=111）；这样定时器 3 可以由定时器 1 周期性的上升沿（即定时器 1 的计时器溢出）信号驱动。
- 最后，必须设置相应（TIMx_CR1 寄存器）的 CEN 位分别启动两个定时器。

注意：如果 OCx 已被选中为定时器 1 的处罚输出（MMS=1xx），它的上升沿用于驱动定时器 2 的计数器。

使用一个定时器使能另一个定时器

在这个例子中 ,定时器 3 的使能由定时器 1 的输出比较控制。参考图 17.40 的连接。只当定时器 1 的 OCxREF 为高时，定时器 3 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_CNT 除以 3（f_{CK_CNT}=$f_{CK\_CNT}/3$）得到。

- 配置定时器 1 为主模式，送出它的输出比较参考信号（OC1REF）为触发输出（TIM1_CR2 寄存器的 MMS=100）
- 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器）
- 配置定时器 3 从定时器 1 获得输入触发（TIM3_SMCR 寄存器的 TS=000）
- 配置定时器 3 为门控模式（TIM3_SMCR 寄存器的 SMS=101）
- 置 TIM3_CR1 寄存器的 CEN=1 以使能定时器 3
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

注意：定时器 3 的时钟不与定时器 1 的时钟同步，这个模式只影响定时器 3 计数器的使能信号。

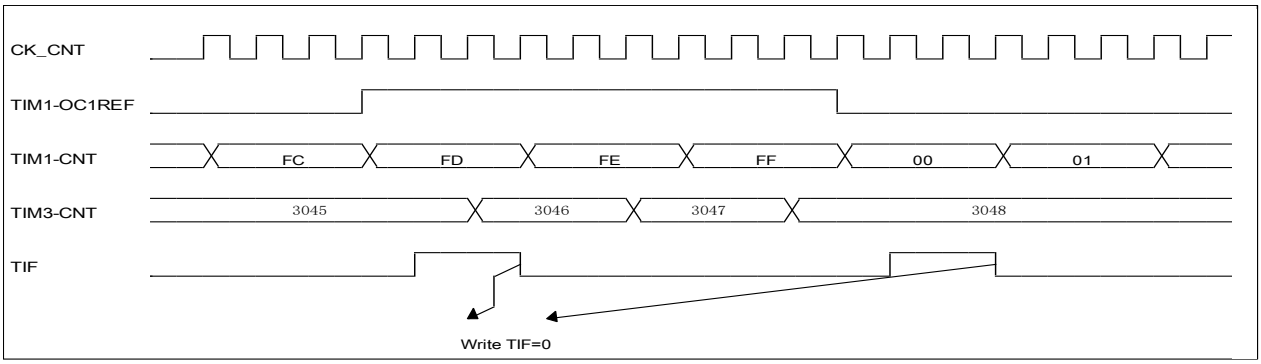


图 17.41 定时器 1 的 OC1REF 控制定时器 3

在上图的例子中，在定时器 3 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。可以再启动定时器 1 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 位即可复位定时器。

在下一个例子中 ,需要同步定时器 1 和定时器 3。定时器 1 是主模式并从 0 开始 ,定时器 3 是从模式并从 0xE7 开始计数；2 个定时器的预分频器系数相同。写 0 到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 3 也随即停止。

- 配置定时器 1 为主模式，送出输出比较 1 参考信号（OC1REF）作为触发输出（TIM1_CR2 寄存器的 MMS=100 ）。
- 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器 ）。
- 配置定时器 3 从定时器 1 获得输入触发（TIM3_SMCR 寄存器的 TS=000 ）。
- 配置定时器 3 为门控模式（TIM3_SMCR 寄存器的 SMS=101 ）。
- 置 TIM1_EGR 寄存器的 UG=1，复位定时器 1。
- 置 TIM3_EGR 寄存器的 UG=1，复位定时器 2。
- 写 0xE7 置定时器 3 的计数器（TIM3_CNT），初始化为 0xE7。
- 置 TIM3_CR1 寄存器的 CEN=1，以使能定时器 3。
- 置 TIM1_CR1 寄存器的 CEN=1，以启动定时器 1。
- 置 TIM1_CR1 寄存器的 CEN=0，以停止定时器 1。

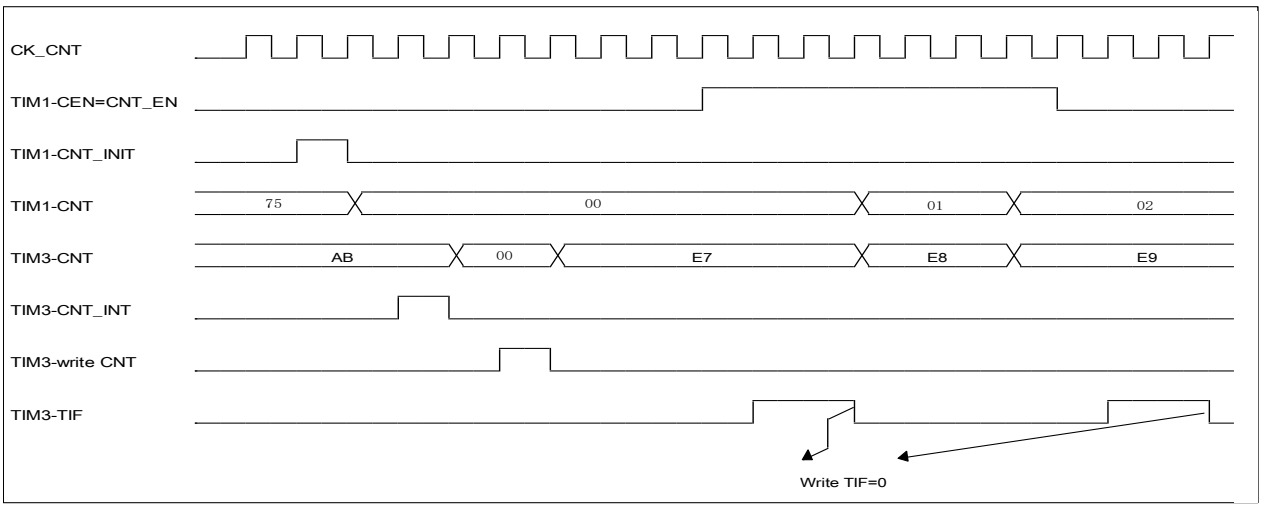


图 17.42 通过使能定时器 1 可以控制定时器 3

使用一个定时器去启动另一个定时器

在这个例子中，使用定时器 1 的更新事件去使能定时器 3。参考图 17.40。一旦定时器 1 产生更新事件，定时器 3 即从它当前的数值(可以是非 0 值)按照分频的内部时钟开始计数。在收到触发信号时，定时器 3 的 CEN 被自动的置 1，同时计数器开始计数直到写 0 到 TIM3_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_CNT}/3)。

- 配置定时器 1 为主模式，送出它的更新事件（UEV）作为触发输出（TIM1_CR2 寄存器的 MMS=010 ）。
- 配置定时器 1 的周期（TIM1_ARR 寄存器 ）。
- 配置定时器 3 从定时器 1 获得输入触发（TIM3_SMCR 寄存器的 TS=000 ）。
- 配置定时器 3 为触发模式（TIM3_SMCR 寄存器的 SMS=110 ）。
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

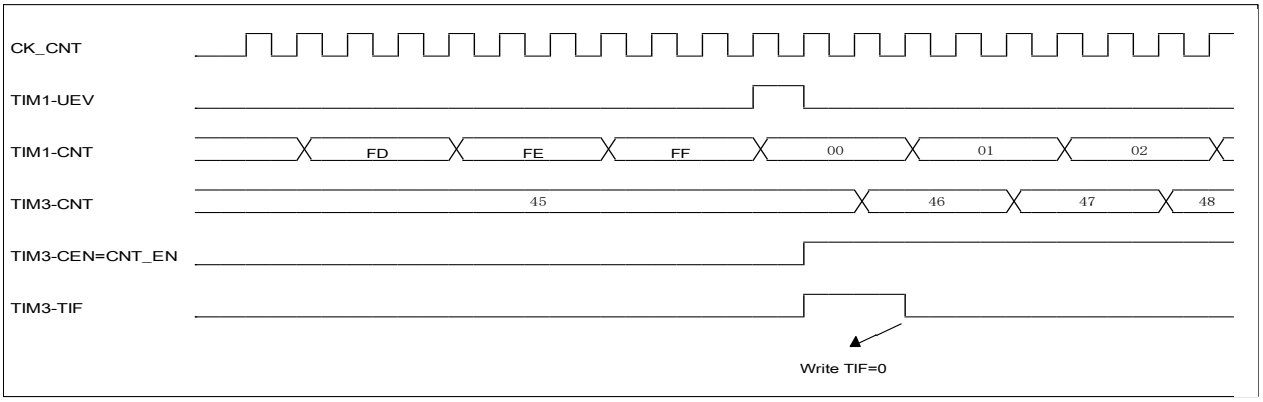


图 17.43 使用定时器 1 的更新触发定时器 2

在上一个例子中，可以再启动计数器之前初始化两个计数器。下图显示在与图 17.43 相同配置情况下，使用触发模式（TIM3_SMCR 寄存器的 SMS=110）而不是门控模式的动作。

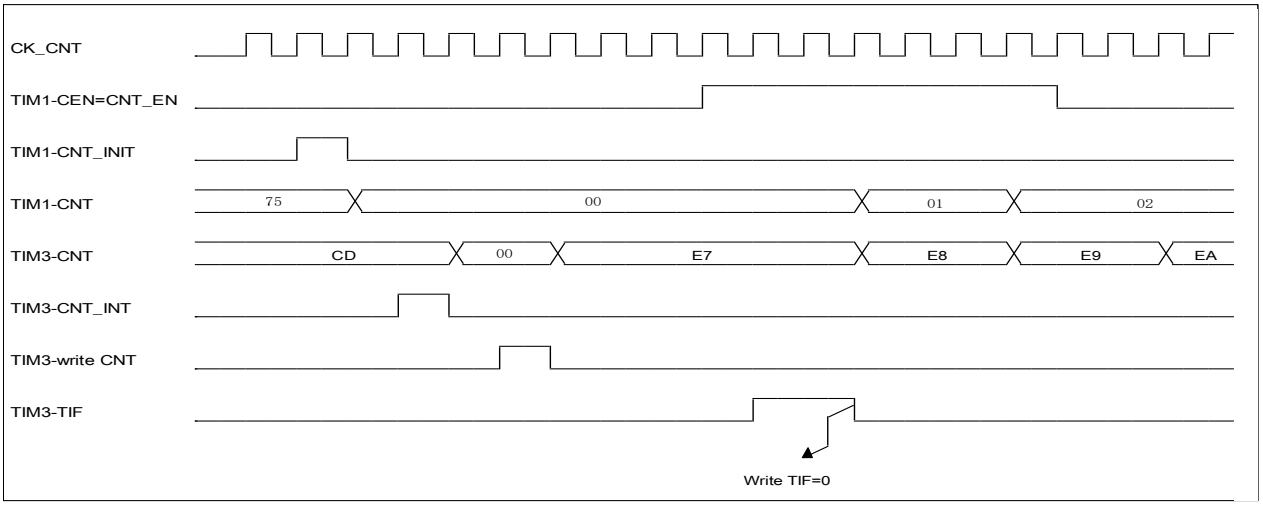


图 17.44 利用定时器 1 的使能触发定时器 3

使用一个定时器作为另一个定时器的预分频器

这个例子使用定时器 1 作为定时器 3 的预分频器。参考图 17.40 的连接，配置如下：

- 配置定时器 1 为主模式，送出它的更新事件 UEV 作为触发输出（TIM1_CR2 寄存器的 MMS=000）。然后每次计数器溢出时输出一个周期信号。
- 配置定时器 1 的周期（TIM1_ARR 寄存器）。
- 配置定时器 3 从定时器 1 获得输入触发（TIM3_SMCR 寄存器的 TS=000）。
- 配置定时器 3 使用外部时钟模式 1（TIM3_SMCR 寄存器的 SMS=111）。
- 置 TIM3_CR1 寄存器的 CEN=1 以启动定时器 3。
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 3。为保证计数器的对齐，定时器 1 必须配置为主/从模式（对应 TI1 为从模式，对应定时器 3 为主）:

- 配置定时器 1 为主模式，送出它的使能作为触发输出（TIM1_CR2 寄存器的 MMS=001）。
- 配置定时器 1 为从模式，从 TI1 获得输入触发（TIM1_SMCR 寄存器的 TS=100）。
- 配置定时器 1 为触发模式（TIM1_SMCR 寄存器的 SMS=110）。
- 配置定时器为主/从模式，TIM1_SMCR 寄存器的 MSM=1。
- 配置定时器 3 从定时器 1 获得输入触发（TIM3_SMCR 寄存器的 TS=000）。
- 配置定时器 3 为触发模式（TIM3_SMCR 寄存器的 SMS=110）。

当定时器 1 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时，两个 TIF 标志也同时被置位。

注意：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/ 从模式下的定时器 1 的 CNT_EN 和 CK_PSC 之间有个延迟。

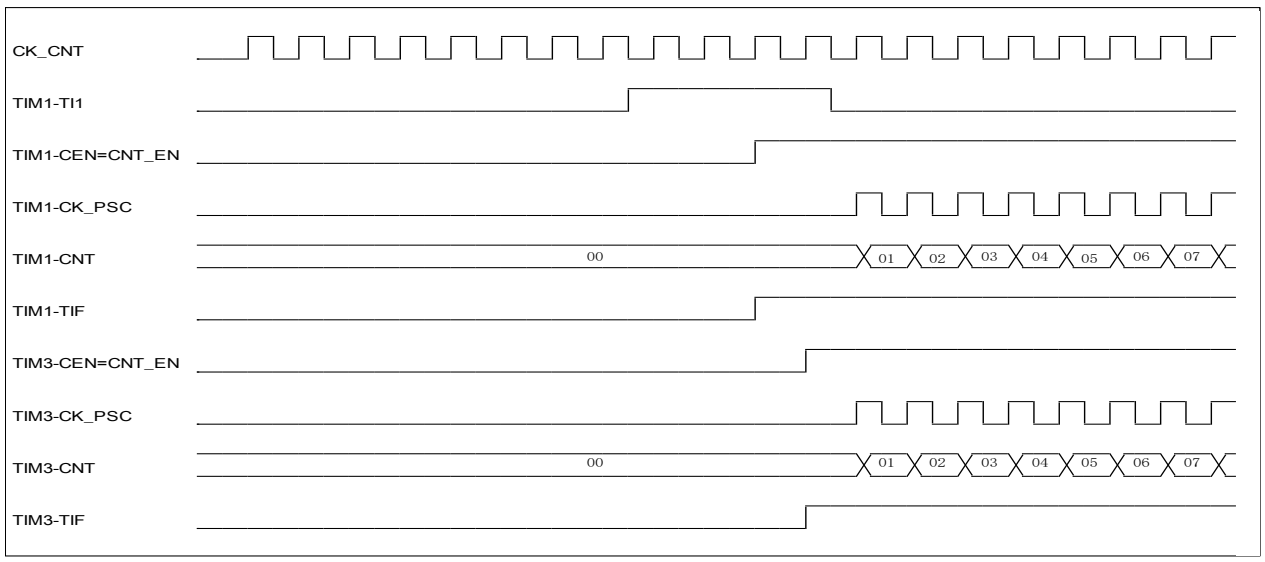


图 17.45 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 3

17.4. 调试模式

当微控制器进入调试模式时（Cortex_M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止。

17.5. TIM3 寄存器映射

下表给出了 TIM3 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM3_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0
0x04	TIM3_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TI1S	MMS[2:0]			CCDS	-	-	-
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x
0x08	TIM3_SMCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIM3_DIER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TDE	-	CC4DE	CC3DE	CC2DE	CC1DE	UDE	-	TIE	-	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	0	x	0	x	0	0	0	0	0
0x10	TIM3_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC4OF	CC3OF	CC2OF	CC1OF	-	-	TIF	-	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	x	0	x	0	0	0	0	0
0x14	TIM3_EGR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TG	-	CC4G	CC3G	CC2G	CC1G	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	0
0x18	TIM3_CCMR1（输出模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIM3_CCMR1（输入模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IC2F[3:0]				IC2PSC [1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1C	TIM3_CCMR2（输出模式）	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIM3_CCMR2（输入模式）	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIM3_CCER	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CC4NP	—	CC4P	CC4E	CC3NP	—	CC3P	CC3E	CC2NP	—	CC2P	CC2E	CC1NP	—	CC1P	CC1E
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	x	0	0	0	x	0	0	0	x	0	0
0x24	TIM3_CNT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CNT[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM3_PSC	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	PSC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM3_ARR	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ARR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIM3_CCR1	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CCR1[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIM3_CCR2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CCR2[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIM3_CCR3	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CCR3[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIM3_CCR4	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CCR4[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TIM3_DCR	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DBL[4:0]					—	—	—	DBA[4:0]				
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0
0x4C	TIM3_DMAR	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DMAB[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17.5.1. TIM3 控制寄存器 1（TIM3_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CKD[1:0]	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	CKD[1:0]	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字

		滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，不要使用此配置
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:5	CMS[1:0]	选择中央对齐模式 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向下计数时被置1。 10：中央对齐模式2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向上计数时被置1。 11：中央对齐模式3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx寄存器中CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被置1。 注意 :在计数器开启时(CEN=1) ,不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	计数方向 0：计数器向上计数； 1：计数器向下计数。 注意：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	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	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，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 – 软件设置UG位 – 复位触发事件产生的更新 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新 软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出/下溢 – 软件设置UG位 – 复位触发事件产生的更新 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果触发复位模式下触发事件到来时或软件设置UG位，计数器和预分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。

		注意：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。而触发模式下可以自动地通过硬件设置CEN位。

17.5.2. TIM3 控制器 2（TIM3_CR2）

地址偏移：0x04

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	TI1S	MMS[2:0]			CCDS	—		
类型	RW	RW	RW	RW	RW	RO-0	RO-0	RO-0

Bit	Name	Function
31:8	NA	保留位，未定义
7	TI1S	TI1选择 0：CC1输入管脚连到TI1(数字滤波器的输入)； 1：CC1、CC2、CC3和CC4管脚经异或后连到TI1。
6:4	MMS[2:0]	主模式选择 这3位用于选择在主模式下送到其它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)。如果触发输入(时钟/触发控制器配置为复位模式)产生复位，则TRGO上的信号相对实际的复位会有一个延迟。 001：使能 – 计数器使能信号被用于作为触发输出(TRGO)。其用于启动多个定时器以便控制在一段时间内使能从定时器。计数器使能信号是通过CEN控制位和门控模式下的触发输入信号的逻辑或产生。除非选择了主/从模式(见TIM1_SMCR寄存器中MSM位的描述)， 当计数器使能信号受控于触发输入时，TRGO上会有一个延迟。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲(MATCH1) – 一旦发生一次捕获或一次比较成功，当CC1IF标志被置1时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF信号被用于作为触发输出(TRGO)。
3	CCDS	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	NA	保留位，未定义

17.5.3. TIM3 从模式控制寄存器（TIM3_SMCR）

地址偏移：0x08

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	ETP	ECE	ETPS[1:0]		ETF[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	MSM	TS[2:0]			OCCS	SMS[2:0]		
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	ETP	外部触发极性 该位决定是ETR还是ETR的反相用于触发操作。 0：ETR不反相，即高电平或上升沿有效； 1：ETR反相，即低电平或下降沿有效。
14	ECE	外部时钟使能 该位用于使能外部时钟模式2。 0：禁止外部时钟模式2； 1：使能外部时钟模式2，计数器由ETRF信号上的的任意有效边沿驱动。 注1：ECE位置1的效果与选择把TRGI连接到ETRF的外部时钟模式1相同(TIM1_SMCR寄存器中，SMS=111，TS=111)。 注2：外部时钟模式2可与下列模式同时使用：触发模式；复位模式；门控模式。但是，此时TRGI决不能与ETRF相连。 注3：外部时钟模式1与外部时钟模式2同时被使能时，外部时钟输入为ETRF。
13:12	ETPS[1:0]	外部触发预分频器 外部触发信号EPRP的频率最大不能超过TIMxCLK频率的1/4。可用预分频器来降低ETRP的频率，当EPRP的频率很高时，它非常有用： 00：预分频器关闭； 01：EPRP的频率/2； 02：EPRP的频率/4； 03：EPRP的频率/8。
11:8	ETF[3:0]	外部触发滤波器选择 这些位定义了ETRP的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到N个事件后会产生一个输出的跳变： 0000：无滤波器，以$f_{SAMPLING} = f_{DTS}$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	主/从模式 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许当前定时器与它的从定时器间的完美同步(通过TRGO)。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	触发选择 这3位选择用于选择同步计数器的触发输入。 000：内部触发0（ITR0） 001：保留 010：内部触发2（ITR2） 011：内部触发3（ITR3） 100：TI1的边沿检测器(TI1F_ED) 101：滤波后的定时器输入1(TI1FP1) 110：滤波后的定时器输入2(TI2FP2) 111：外部触发输入(ETRF) 注意：这些位只能在未用到(如SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	OCCS	OCxREF清除选择 此位用于选择清除COxREF信号的清除源。 0：OCREF_CLR_INT信号连接到OCREF_CLR内部输入信号上； 1：OCREF_CLR_INT信号连接到ETRF信号上。
2:0	SMS[2:0]	时钟/触发/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：时钟/触发控制器禁止 – 如果CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式1 – 根据TI1FP1的电平，计数器在TI2FP2的边沿向上/下计数。 010：编码器模式2 – 根据TI2FP2的电平，计数器在TI1FP1的边沿向上/下计数。 011：编码器模式3 – 根据另一个输入的电平，计数器在TI1FP1和TI2FP2的边沿向上/下计数。 100：复位模式 – 在选中的触发输入(TRGI)的上升沿时重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式

		– 计数器在触发输入TRGI的上升沿启动(但不复位) ,只有计数器的启动是受控的。 111：外部时钟模式1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果TI1F_ED被选为触发输入(TS=100)时，不要使用门控模式。这是因为TI1F_ED在每次TI1F变化时只是输出一个脉冲 ,然而门控模式是要检查触发输入的电平。

从定时器	ITR0（TS=000）	ITR2（TS=010）	ITR3（TS=011）
TIM3	TIM1	TIM15	TIM14

表格 17.2 TIM3 内部触发连接

17.5.4. TIM3 DMA/中断使能寄存器（TIM3_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—	TDE	—	CC4DE	CC3DE	CC2DE	CC1DE	UDE
类型	RO-0	RW	RO-0	RW	RW	RW	RW	RW
7:0	—	TIE	—	CC4IE	CC3IE	CC2IE	CC1IE	UIE
类型	RO-0	RW	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	TDE	允许触发 DMA 请求 0：触发 DMA 请求禁止 1：触发 DMA 请求允许
13	NA	保留位，未定义
12	CC4DE	捕获/比较 4 DMA 请求使能 0：CC4 DMA 请求禁止 1：CC4 DMA 请求允许
11	CC3DE	捕获/比较 3 DMA 请求使能 0：CC3 DMA 请求禁止 1：CC3 DMA 请求允许
10	CC2DE	捕获/比较 2 DMA 请求使能 0：CC2 DMA 请求禁止 1：CC2 DMA 请求允许
9	CC1DE	捕获/比较 1 DMA 请求使能

		0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
8	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允许
7	NA	保留位，未定义
6	TIE	触发中断使能 0：触发中断禁止 1：触发中断允许
5	NA	保留位，未定义
4	CC4IE	捕获/比较 4 中断使能 0：CC4 中断禁止 1：CC4 中断允许
3	CC3IE	捕获/比较 3 中断使能 0：CC3 中断禁止 1：CC3 中断允许
2	CC2IE	捕获/比较 2 中断使能 0：CC2 中断禁止 1：CC2 中断允许
1	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

17.5.5. TIM3 状态寄存器（TIM3_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			CC4OF	CC3OF	CC2OF	CC1OF	—
类型	RO-0	RO-0	RO-0	RC_W0	RC_W0	RC_W0	RC_W0	RO-0
7:0	保留位	TIF	保留位	CC4IF	CC3IF	CC2IF	CC1IF	UIF
类型	RO-0	RC_W0	RO-0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	Function
31:13	NA	保留位，未定义

12	CC4OF	捕获/比较4重复捕获标志 参见CC1OF描述。
11	CC3OF	捕获/比较3重复捕获标志 参见CC1OF描述。
10	CC2OF	捕获/比较2重复捕获标志 参见CC1OF描述。
9	CC1OF	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，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIM1_CCR1寄存器时，CC1IF的状态已经为1。
8	NA	保留位，未定义
7	NA	保留位，未定义
6	TIF	触发器中断标志 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在TRGI输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置1。它由软件清0。 0：无触发器事件产生； 1：触发中断等待响应。
5	NA	保留位，未定义
4	CC4IF	捕获/比较4中断标志 参考CC1IF描述。
3	CC3IF	捕获/比较3中断标志 参考CC1IF描述。
2	CC2IF	捕获/比较2中断标志 参考CC1IF描述。
1	CC1IF	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，但在中心对称模式下除外(参考TIM1_CR1寄存器的CMS位)。它由软件清0。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_ARR的内容时，在向上或向上/向下计数模式时计数器溢出，或向下计数模式时计数器下溢条件下，CC1IF位变高。 如果通道CC1配置为输入模式： 当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIMx_CCR1清0。 0：无输入捕获产生； 1：计数器值已被捕获至TIMx_CCR1(在IC1上检测到与所选极性相同的边沿)。
0	UIF	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，当计数器上溢或下溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当计数器CNT被触发事件重新初始化时 (参考从模式控制寄存器TIMx_SMCR)。

17.5.6. TIM3 事件产生寄存器（TIM3_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	TG	—	CC4G	CC3G	CC2G	CC1G	UG
类型	RO-0	W	RO-0	W	W	W	W	W

Bit	Name	Function
31:7	NA	保留位，未定义
6	TG	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，若开启对应的中断和DMA，则产生相应的中断和DMA。
5	NA	保留位，未定义
4	CC4G	产生捕获/比较4事件 参考CC1G描述。
3	CC3G	产生捕获/比较3事件 参考CC1G描述。
2	CC2G	产生捕获/比较2事件 参考CC1G描述。
1	CC1G	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： 设置CC1IF=1，若开启对应的中断和DMA，则产生相应的中断和DMA。 若通道CC1配置为输入： 当前的计数器值被捕获至TIMx_CCR1寄存器，设置CC1IF=1，若开启对应的中断和DMA ,则产生相应的中断和DMA。若CC1IF已经为1 ,则设置CC1OF=1。
0	UG	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)。若在中心对称模式下或DIR=0(向上计数)则计数器被清0；若DIR=1(向下计数)则计数器取TIMx_ARR的值。

17.5.7. TIM3 捕获/比较模式寄存器（TIM3_CCMR1）

地址偏移：0x18

复位值：0x0000

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

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
	IC2F[3:0]				IC2PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15	OC2CE	输出比较2清零使能
14:12	OC2M[2:0]	输出比较2模式
11	OC2PE	输出比较2预装载使能
10	OC2FE	输出比较2快速使能
9:8	CC2S	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7	OC1CE	输出比较1清零使能 0：OC1REF不受ETRF输入的影响； 1：一旦检测到ETRF输入高电平，清除OC1REF。
6:4	OC1M[2:0]	输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1、OC1N的值。OC1REF是高电平有效，而OC1、OC1N的有效电平取决于CC1P、CC1NP位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用；

		001：匹配时设置通道1的输出为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平； 在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)， 否则为有效电平(OC1REF=1)。 111：PWM模式2 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平； 在向下计数时，一旦TIMx_CNT>TIM1_CCRx时通道1为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIMx_BKR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：在有互补输出的通道上，这些位是预装载的。如果TIMx_CR2寄存器的CCPC=1，OC1M 位只有在COM事件发生时，才从预装载位取新值。
3	OC1PE	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，并且新写入的数值立即起作用。 1：开启TIMx_CCR1寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1的预装载值在更新事件到来时被加载至当前寄存器中。 注1：一旦LOCK级别设为3(TIMx_BKR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：为了操作正确，在PWM模式下必须使能预装载功能。但在单脉冲模式下(TIMx_CR1寄存器的OPM=1)，它不是必须的。
2	OC1FE	输出比较1 快速使能 该位用于加快CC输出对触发输入事件的响应。 0：根据计数器与CCR1的值，CC1正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活CC1输出的最小延时为5个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC被设置为比较电平而与比较结果无关。采样触发器的有效沿和CC1输出间的延时被缩短为3个时钟周期。 注意：OC1FE只在通道被配置成PWM1或PWM2模式时起作用。
1:0	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC2F[3:0]	输入捕获2滤波器
11:10	IC2PSC[1:0]	输入/捕获2预分频器
9:8	CC2S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7:4	IC1F[3:0]	输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，以$f_{SAMPLING} = f_{CK_INT}$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]	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]	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TIMx_SMCR寄存器的TS位选择)。 注：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

17.5.8. TIM3 捕获/比较模式寄存器 2（TIM3_CCMR2）

偏移地址：0x1C

复位值：0x0000

参考 CCMR1 寄存器的描述。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
	IC4F[3:0]				IC4PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	IC3F[3:0]				IC3PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15	OC4CE	输出比较4清零使能
14:12	OC4M[2:0]	输出比较4模式
11	OC4PE	输出比较4预装载使能
10	OC4FE	输出比较4快速使能
9:8	CC4S	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC4S仅在通道关闭时(TIMx_CCER寄存器的CC4E=0且已被更新)才是可写的。
7	OC3CE	输出比较3清零使能
6:4	OC3M[2:0]	输出比较3模式
3	OC3PE	输出比较3预装载使能
2	OC3FE	输出比较3快速使能
1:0	CC3S	捕获/比较3选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3通道被配置为输出； 01：CC3通道被配置为输入，IC3映射在TI3上； 10：CC3通道被配置为输入，IC3映射在TI4上； 11：CC3通道被配置为输入，IC3映射在TRC上。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。

		注：CC3S仅在通道关闭时(TIMx_CCER寄存器的CC3E=0、CC3NE=0且已被更新)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC4F[3:0]	输入捕获4滤波器
11:10	IC4PSC[1:0]	输入/捕获4预分频器
9:8	CC4S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC4S仅在通道关闭时(TIMx_CCER寄存器的CC4E=0且已被更新)才是可写的。
7:4	IC3F[3:0]	输入捕获3滤波器
3:2	IC3PSC[1:0]	输入/捕获3预分频器
1:0	CC3S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC3S仅在通道关闭时(TIMx_CCER寄存器的CC3E=0，CC3NE=0且已被更新)才是可写的。

17.5.9. TIM3 捕获/比较使能寄存器（TIM3_CCER）

地址偏移：0x20

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		CC4P	CC4E	CC3NP	—	CC3P	CC3E
类型	RO-0	RO-0	RW	RW	RW	RO-0	RW	RW
7:0	CC2NP	—	CC2P	CC2E	CC1NP	—	CC1P	CC1E
类型	RW	RO-0	RW	RW	RW	RO-0	RW	RW

Bit	Name	Function

31:14	NA	保留位，未定义
13	CC4P	输入捕获/比较4输出极性。参考CC1P的描述。
12	CC4E	输入捕获/比较4输出使能。参考CC1E的描述。
11	CC3NP	输入捕获/比较3互补输出极性。参考CC1NP的描述。
10	NA	保留位，未定义
9	CC3P	输入捕获/比较3输出极性。参考CC1P的描述。
8	CC3E	输入捕获/比较3输出使能。参考CC1E的描述。
7	CC2NP	输入捕获/比较2互补输出极性。参考CC1NP的描述。
6	NA	保留位，未定义
5	CC2P	输入捕获/比较2输出极性。参考CC1P的描述。
4	CC2E	输入捕获/比较2输出使能。参考CC1E的描述。
3	CC1NP	输入捕获/比较1互补输出极性 CC1通道配置为输出： 0：OC1N高电平有效； 1：OC1N低电平有效。 CC1通道配置为输入： 本为用于和CC1P联合定义TI1FP1和TI2FP1的极性。参考CC1P的描述。 注1：一旦LOCK级别(TIMx_BKR寄存器中的LOCK位)设为2或3且CC1S=00(通道配置为输出) 则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1NP位才从预装载位中取新值。
2	NA	保留位，未定义
1	CC1P	输入捕获/比较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位联合选择在触发或捕获模式下TI1FP1和TI2FP1的有效极性。 00：非反相/上升沿 电路作用于TIxFP1的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1非反相（在门控模式或编码模式）。 01：反相/下降沿 电路作用于TIxFP1的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 10：保留不用 11：非反相/上升或下降沿 电路作用于TIxFP1的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 注1：一旦LOCK级别(TIMx_BKR寄存器中的LOCK位)设为2或3，则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1P位才从预装载位中取新值。
0	CC1E	输入捕获/比较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和CC1NE位的值。

		1：开启 - OC1信号输出到对应的输出引脚 ,其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1NE位的值 。 CC1通道配置为输入： 该位决定了计数器的值是否能捕获入TIMx_CCR1寄存器。 0：捕获禁止； 0：捕获使能。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1E位才从预装载位中取新值。

表格 17.3 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

CCxE 位	OCx 输出状态
0	禁止输出（OCx=0，OCx_EN=0）
1	OCx=OCxREF + 极性， OCx_EN=1

注意：引脚连接到互补的 OCx 通道的外部 I/O 引脚的状态，取决于 OCx 通道状态和 GPIO 寄存器。

17.5.10. TIM3 计数器（TIM3_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

17.5.11. TIM3 预分频器（TIM3_PSC）

地址偏移：0x28

复位值：0x0000

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

类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	PSC[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	PSC[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

17.5.12. TIM3 自动重装载寄存器（TIM3_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 参考章节13.3.1：时基单元中有关ARR的更新和动作的相关内容。 当自动装载值为空时，计数器不工作。

17.5.13. TIM3 捕获/比较寄存器（TIM3_CCR1）

地址偏移：0x34

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR1[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR1[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR1[15:0]	捕获/比较通道1的值 若CC1通道配置为输出： CCR1决定了装入当前捕获/比较1寄存器的值（预装载值）。 如果在TIMx_CCMR1寄存器（OC1PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC1端口上输出信号。 若CC1通道配置为输出入： CCR1包含由上一次输入捕获1事件（IC1）传输的计数器值。

17.5.14. TIM3 捕获/比较寄存器 2（TIM3_CCR2）

地址偏移：0x38

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR2[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR2[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR2[15:0]	捕获/比较通道1的值 若CC2通道配置为输出： CCR2决定了装入当前捕获/比较2寄存器的值（预装载值）。 如果在TIMx_CCMR2寄存器（OC2PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较2寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC2端口上输出信号。 若CC2通道配置为输出入： CCR2包含由上一次输入捕获2事件（IC2）传输的计数器值。

17.5.15. TIM3 捕获/比较寄存器 3（TIM3_CCR3）

地址偏移：0x3C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR3[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR3[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR3[15:0]	捕获/比较通道1的值 若CC3通道配置为输出： CCR3决定了装入当前捕获/比较3寄存器的值（预装载值）。 如果在TIMx_CCMR3寄存器（OC3PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较3寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC3端口上输出信号。 若CC3通道配置为输出入： CCR3包含由上一次输入捕获3事件（IC3）传输的计数器值。

17.5.16. TIM3 捕获/比较寄存器 4（TIM3_CCR4）

地址偏移：0x40

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR4[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR4[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义

15:0	CCR4[15:0]	捕获/比较通道1的值 若CC4通道配置为输出： CCR4决定了装入当前捕获/比较4寄存器的值（预装载值）。 如果在TIMx_CCMR4寄存器（OC4PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较4寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC4端口上输出信号。 若CC4通道配置为输出入： CCR4包含由上一次输入捕获4事件（IC4）传输的计数器值。

17.5.17. TIM3 DMA 控制寄存器（TIM3_DCR）

地址偏移：0x48

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			DBL[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
7:0	—			DBA[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:13	NA	保留位，未定义
12:8	DBL[4:0]	DMA突发传输长度 这5位定义了DMA在突发传输模式下的传输长度。（当对TIMx_DMAR寄存器进行读或写时，定时器则进行一次突发传输） 00000：1次传输 00001：2次传输 00010：3次传输 … 10001：18次传输
7:5	NA	保留位，未定义
4:0	DBA[4:0]	DMA基地址 这5位定义了DMA传输的基地址（当对TIMx_DMAR寄存器进行读或写时），DBA定义为TIMx_CR1寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR … 例：要完成如下的传输：DBL=7，DBA=TIMx_CR1 此时传输从TIMx_CR1的地址开始向连续7个寄存器进行操作。

17.5.18. TIM3 全部传输时 DMA 地址（TIM3_DMAR）

地址偏移：0x4C

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DMAB[15:0]	DMA突发传输寄存器 对TIMx_DMAR寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1地址）+（DBA + DMA索引）*4 其中： TIMx_CR1地址是控制器1（TIMx_CR1）所在的地址； DBA是TIMx_DCR寄存器中定义的基地址； DMA索引是由DMA自动控制的偏移量取决于TIMx_DCR寄存器中的DBL。

如何使用 DMA 突发传输的例子

本例中使用定时器 DMA 的突发传输功能，将 CCRx 寄存器（x=2，3，4）的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。

按如下步骤进行操作：

1. 配置相关的 DMA 通道：
   - DMA 通道设备地址为 DMAR 寄存器地址
   - DMA 通道存储器地址为包含要通过 DMA 传送到 CCRx 寄存器的数据 RAM 缓冲区地址
   - 传送数据数量=3
   - 循环模式禁止
2. 配置 DCR 寄存器中的 DBA 和 DBL 位：DBL=3 说明连续传输次数为 3 次；DBA=0xE，初始传输的偏移地址为 0x38（TIMx_CCR2）。
3. 使能 TIMx 更新 DMA 请求
4. 使能 TIMx
5. 使能 DMA 通道

注意：在本例中所有 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传输的数据数量应该为 6，而 RAM 缓冲区要包含 data1，data2，data3，data4，data5 和 data6。数据按如下过程被传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个更新 DMA 请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

18. 基本定时器（TIM6）

18.1. TIM6 简介

基本定时器 TIM6 由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

18.2. TIM6 主要特性

TIM6 定时器的功能包括：

- 16 位向上自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 计数器向上溢出时产生中断/DMA

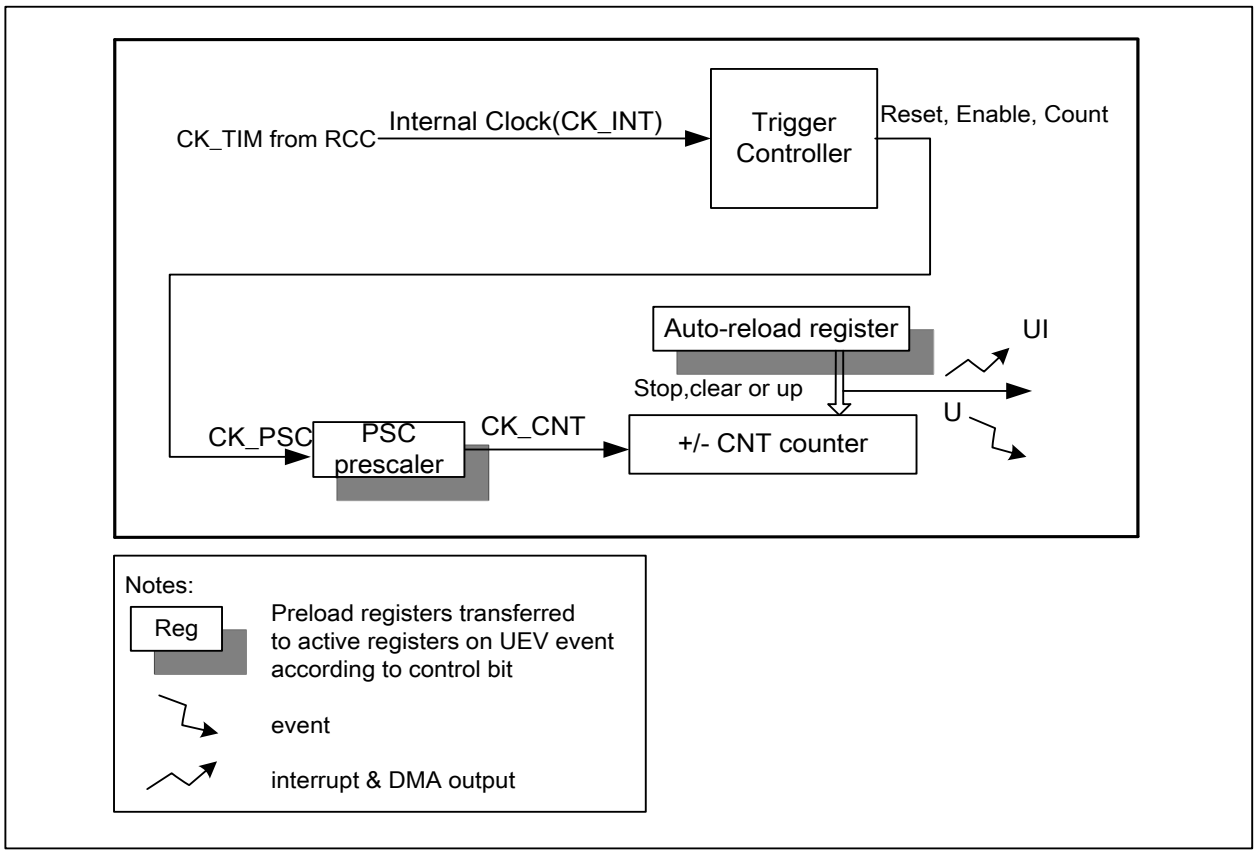


图 18.1 基本定时器框图

18.3. TIM6 功能描述

18.3.1. 时基单元

基本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。此计数器时钟由分频器分频得到。计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

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

自动装载寄存器是预先装载的，写或读自动装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容会被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。

计数器由预分频器的时钟输出 CK_CNT 驱动 ,仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN）时，CK_CNT 才有效。

注意：在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

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。新的预分频器的参数在下一次更新事件到来时被采用。

图 18.2 和图 18.3 给出了在运行时更改预分频器因子，计数器的相关动作的例子。

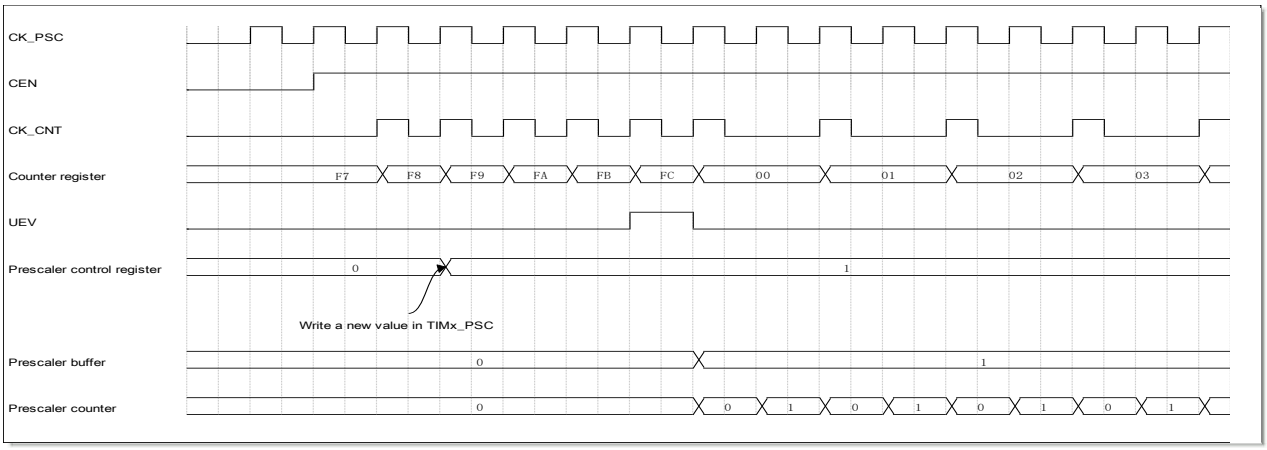


图 28.2 预分频系数从 1 变到 2 时，计数器的时序图

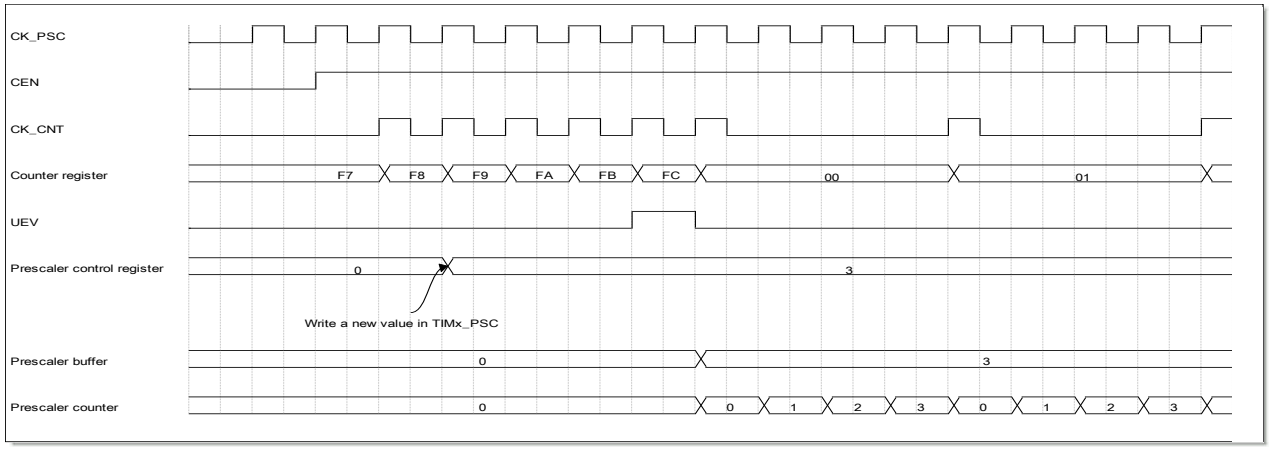


图 18.3 预分频系数从 1 变到 4 时，计数器的时序图

18.3.2. 计数器模式

计数器从 0 计数到自动加载值（TIMx_ARR 计数器的值），然后重新从 0 开始计数并且产生一个计数器溢出事件。在 TIMx_EGR 寄存器中设置 UG 位也同样可以产生一个更新事件。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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）。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

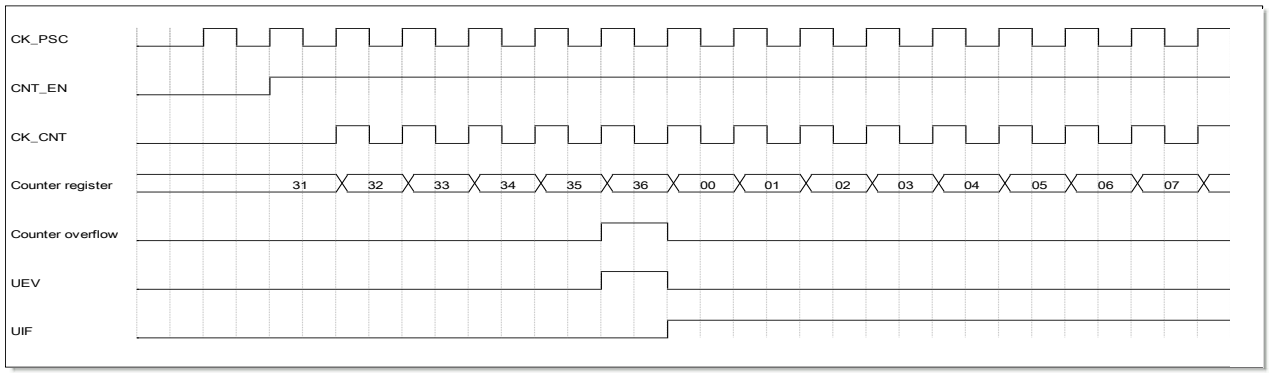


图 18.4 计数器时序图，内部时钟分频因子为 1

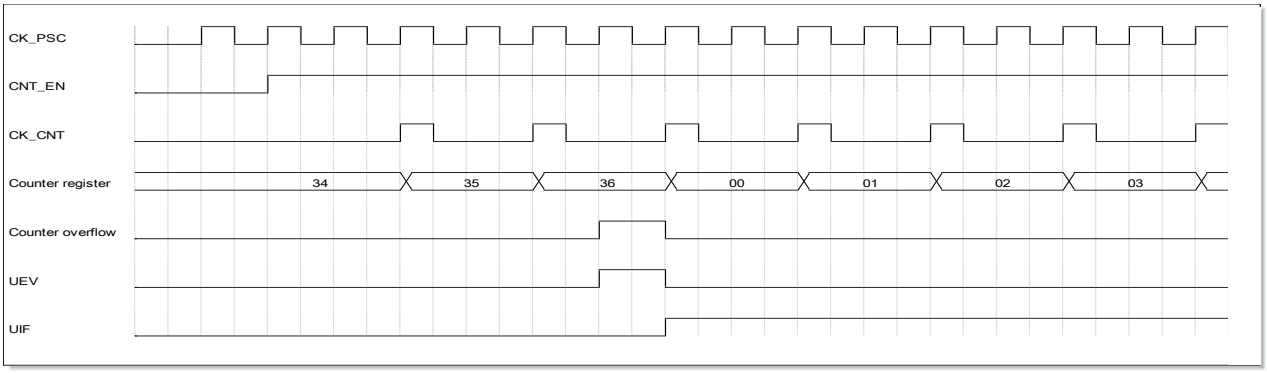


图 18.5 计数器时序图，内部时钟分频因子为 2

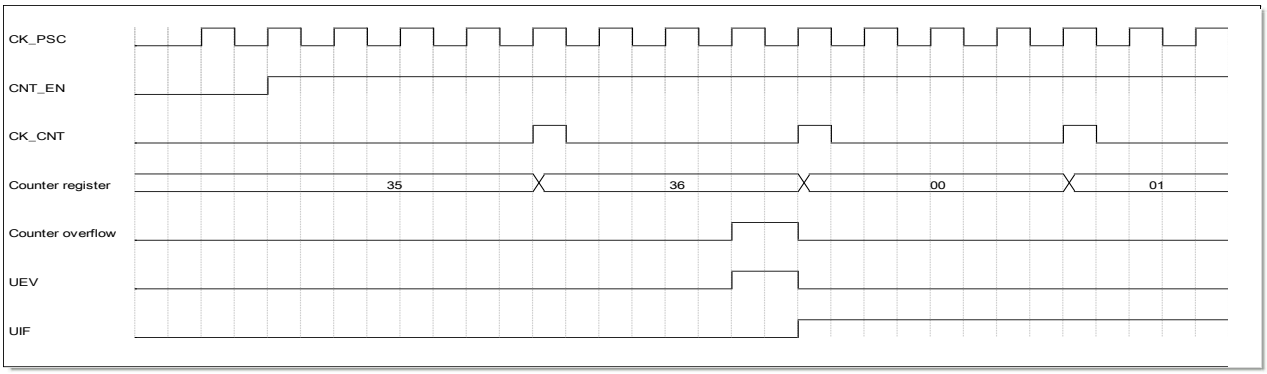


图 18.6 计时器时序图，内部时钟分频因子为 4

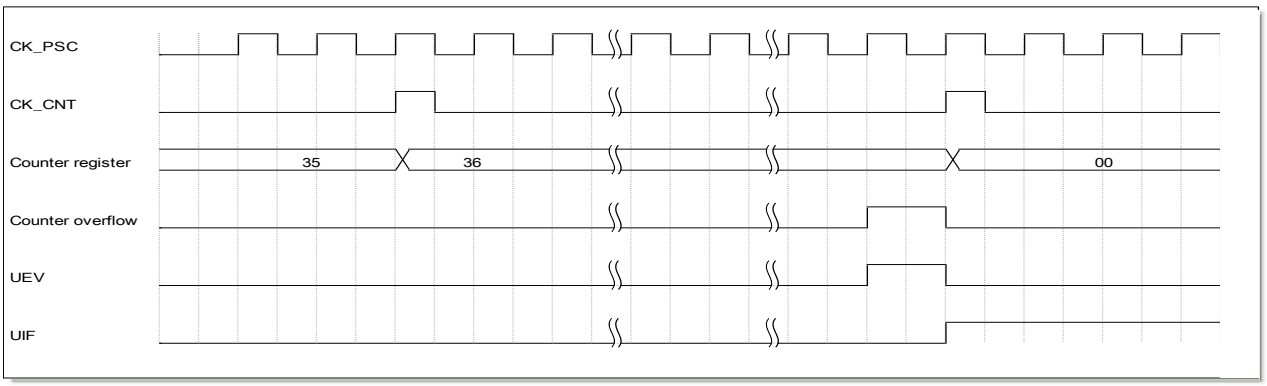


图 18.7 计数器时序图，内部时钟分频因子为 N

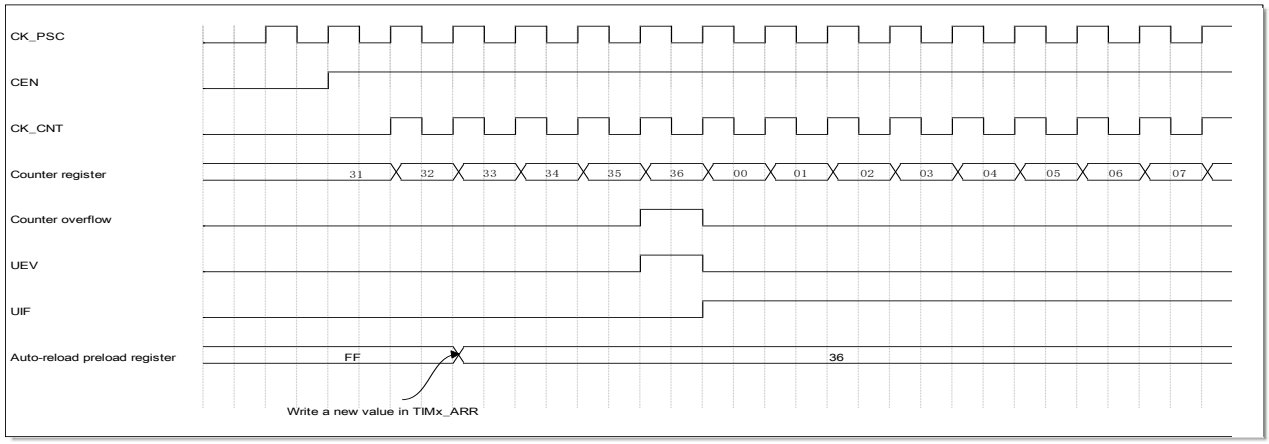


图 18.8 计数器时序图，当 ARPE=0 时的更新事件

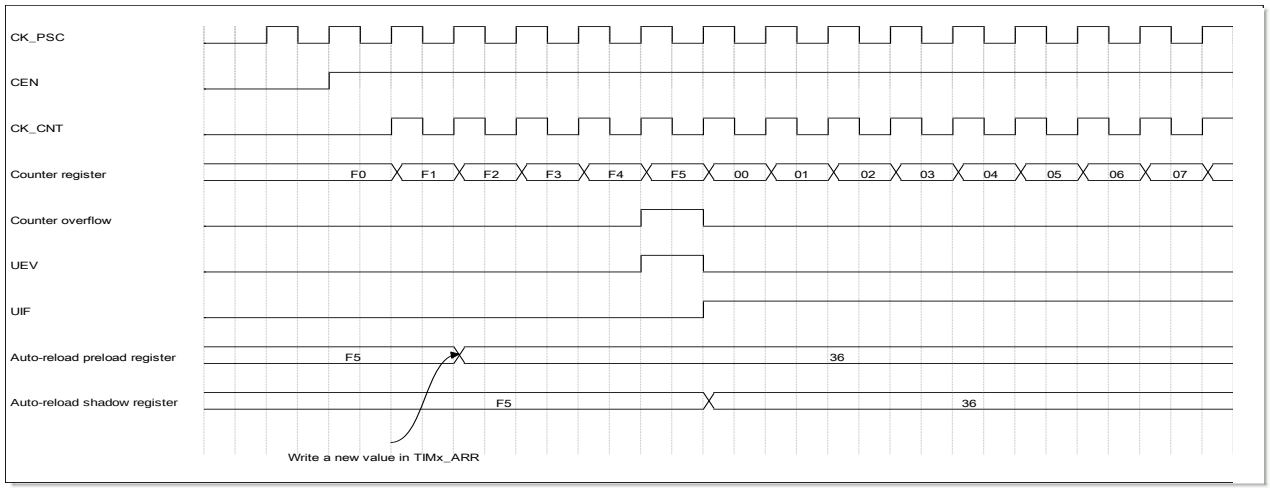


图 18.9 计数器时序图，当 ARPE=1 时的更新事件

18.3.3. 时钟源

计数器时钟可由内部时钟源提供。

CEN（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是实际上的控制位，并且只能被软件修改（除非 UG 位自动被清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示了在不带预分频器时，控制电路和向上计数器在正常模式下的动作。

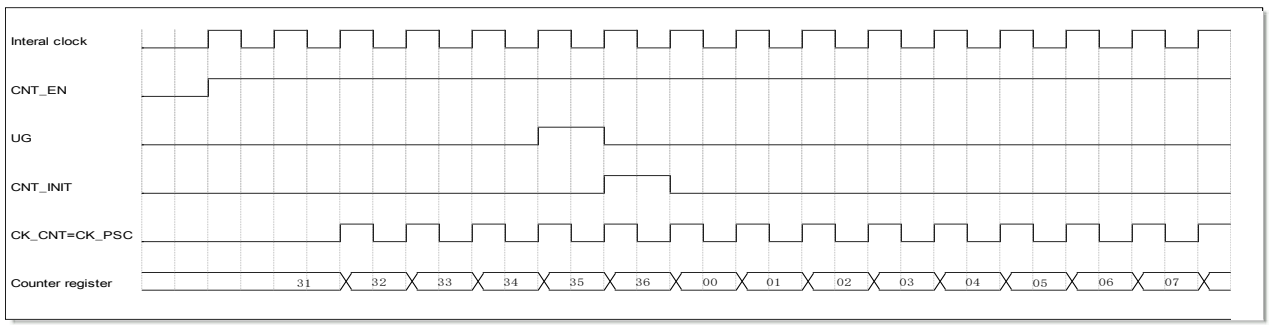


图 18.10 内部时钟分频是 1 时正常模式下的控制时序图

18.3.4. 调试模式

当微控制器进入调试模式时（Cortex_M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止。

18.4. TIM6 寄存器映射

下表给出了 TIM6 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM6_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ARPE	-	-	-	OPM	URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	0
0x0C	TIM6_DIER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	UDE	-	-	-	-	-	-	-	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0
0x10	TIM6_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0
0x14	TIM6_EGR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0
0x24	TIM6_CNT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CNT[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM6_PSC	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PSC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM6_ARR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ARR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

18.4.1. TIM6 控制寄存器 1（TIM6_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	ARPE	—			OPM	URS	UDIS	CEN
类型	RW	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:4	NA	保留位，未定义
3	OPM	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	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，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 – 软件设置UG位 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新 软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出/下溢 – 软件设置UG位 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果软件设置UG位，计数器和预分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。

18.4.2. TIM6 DMA/中断使能寄存器（TIM6_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							UDE
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
7:0	—							UIE
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
31:9	NA	保留位，未定义
8	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允许
7:1	NA	保留位，未定义
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

18.4.3. TIM6 状态寄存器（TIM6_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—							UIF
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0

Bit	Name	Function
31:1	NA	保留位，未定义
0	UIF	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，当计数器上溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时；

18.4.4. TIM6 事件产生寄存器（TIM6_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—							UG
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	W

Bit	Name	Function
31:1	NA	保留位，未定义
0	UG	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)。

18.4.5. TIM6 计数器（TIM6_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

18.4.6. TIM6 预分频器（TIM6_PSC）

地址偏移：0x28

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

18.4.7. TIM6 自动重装载寄存器（TIM6_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 参考章节15.3.1：时基单元中有关ARR的更新和动作的相关内容。 当自动装载值为空时，计数器不工作。

19. 通用定时器（TIM14）

19.1. TIM14 简介

通用定时器（TIM14）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。
它适合多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较和 PWM）。
使用定时器预分频器和 RCC 时钟控制预分频器 ,可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。
通用定时器（TIMx）是完全独立的，它们不共享任何资源。它们可以同步操作。

19.2. TIM14 主要特性

TIM14 定时器的功能包括：

- 16 位向上自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 1 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
- 如下事件发生时产生中断/DMA：
  - 更新事件：计数器向上溢出，计数器初始化（通过软件）
  - 输入捕获
  - 输出比较

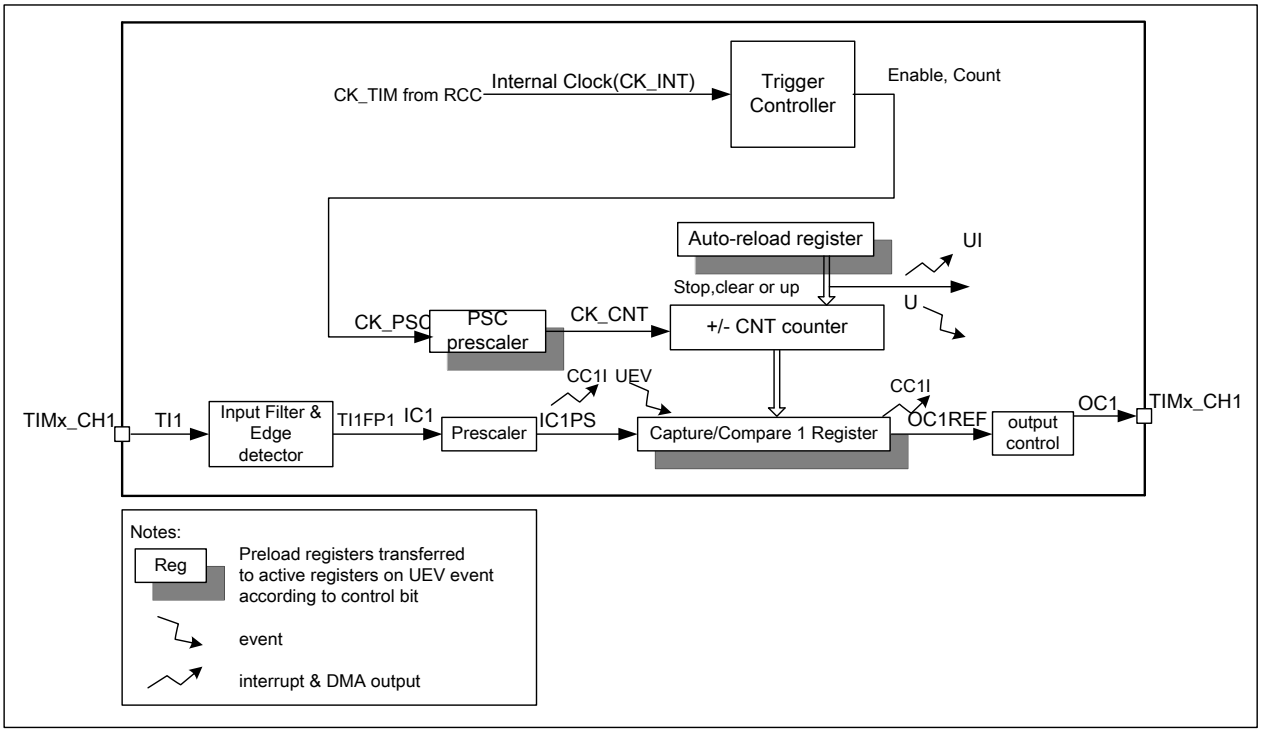


图 19.1 通用定时器框图

19.3. TIM14 功能描述

19.3.1. 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。此计数器时钟由分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

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

自动装载寄存器是预先装载的，写或读自动装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容会被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动 ,仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN) 时，CK_CNT 才有效。

注意：在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

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。新的预分频器的参数在下一次更新事件到来时被采用。

图 19.2 和图 19.3 给出了在运行时更改预分频器因子，计数器的相关动作的例子。

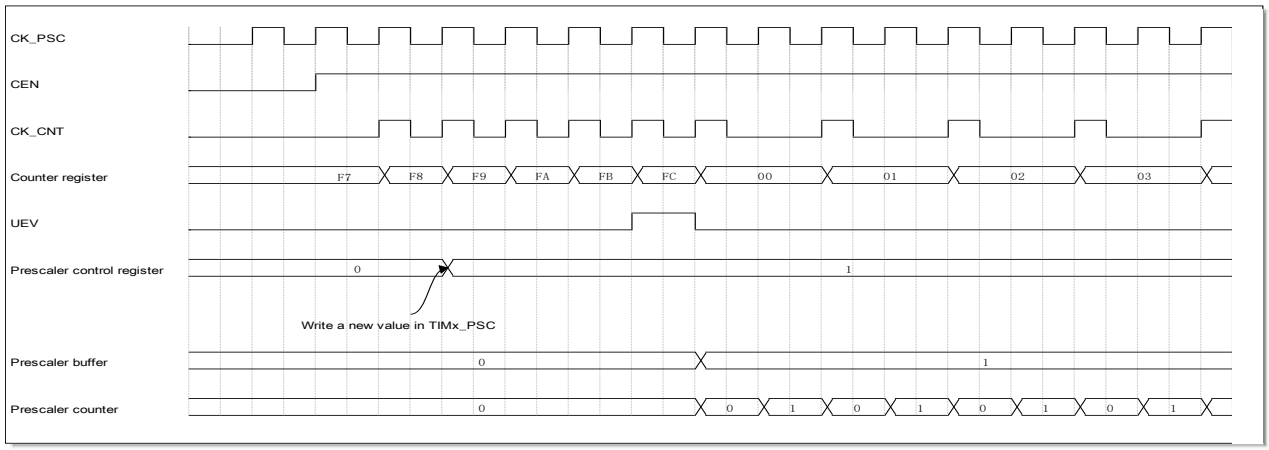


图 36-2 预分频系数从 1 变到 2 时，计数器的时序图

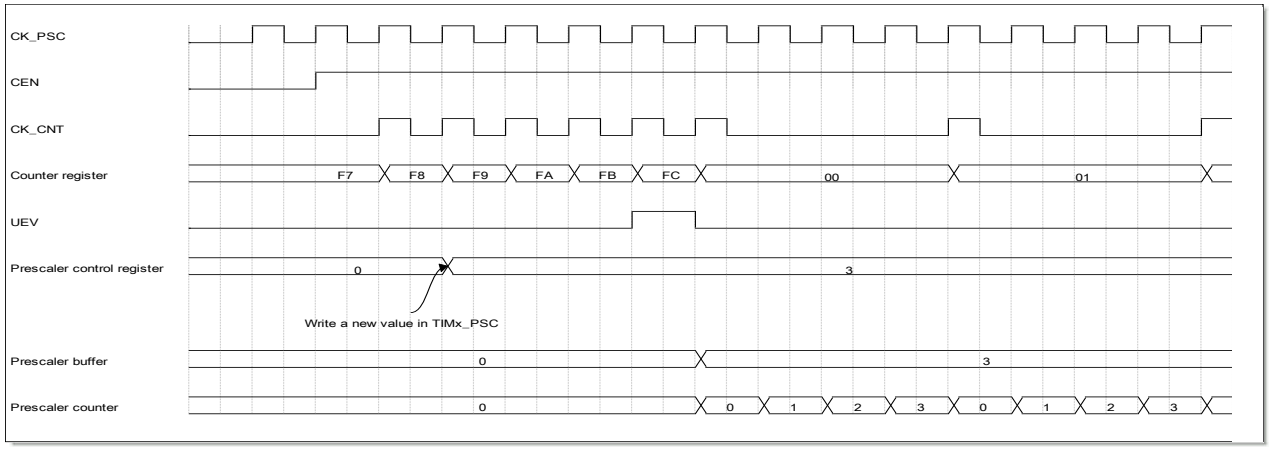


图 19.3 预分频系数从 1 变到 4 时，计数器的时序图

19.3.2. 计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值（TIMx_ARR 计数器的值），然后重新从 0 开始计数并且产生一个计数器溢出事件。在 TIMx_EGR 寄存器中（通过软件方式）设置 UG 位也同样可以产生一个更新事件。通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

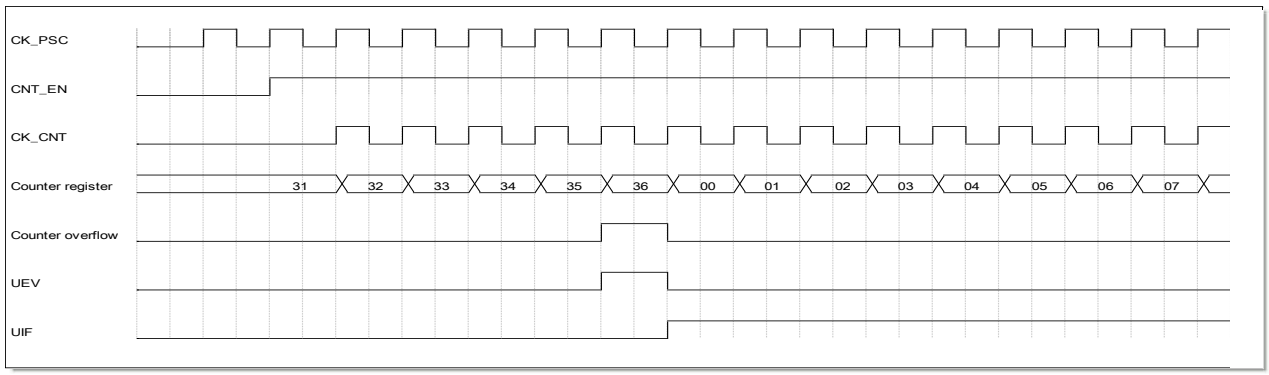


图 19.4 计数器时序图，内部时钟分频因子为 1

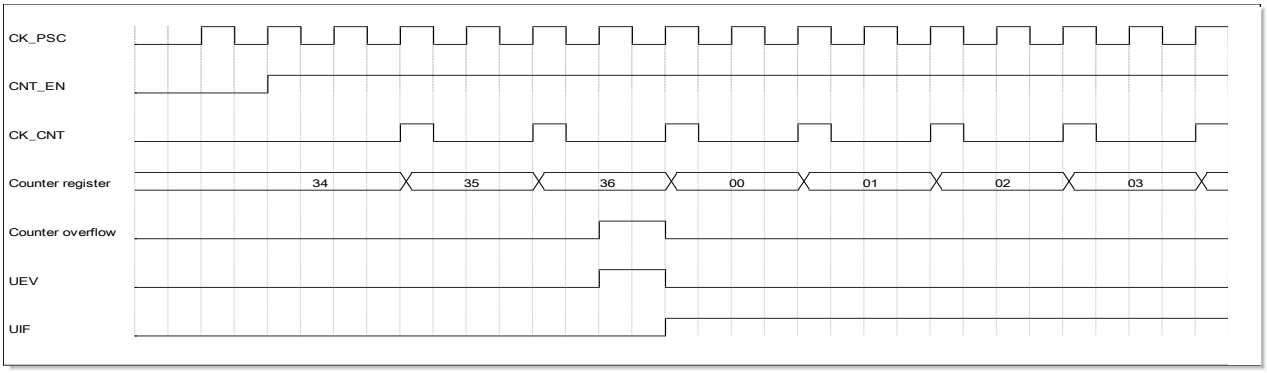


图 19.5 计数器时序图，内部时钟分频因子为 2

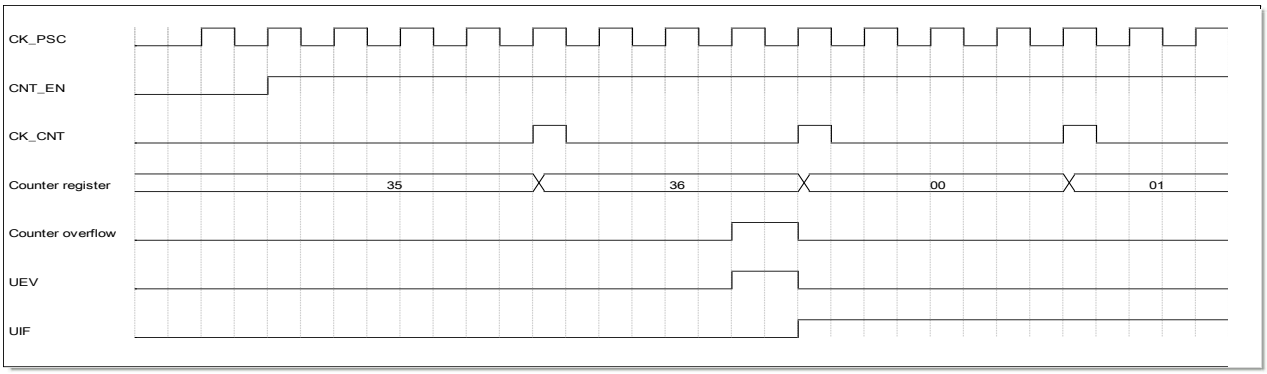


图 19.6 计时器时序图，内部时钟分频因子为 4

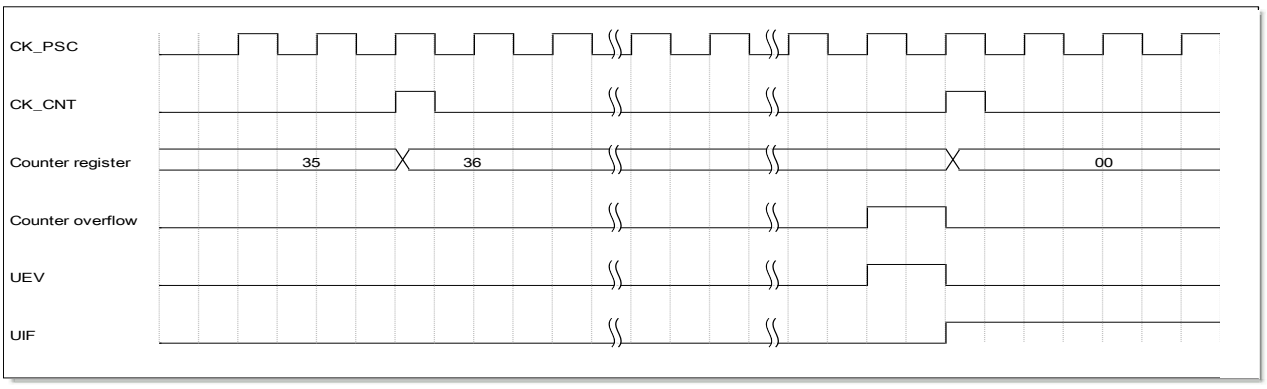


图 19.7 计数器时序图，内部时钟分频因子为 N

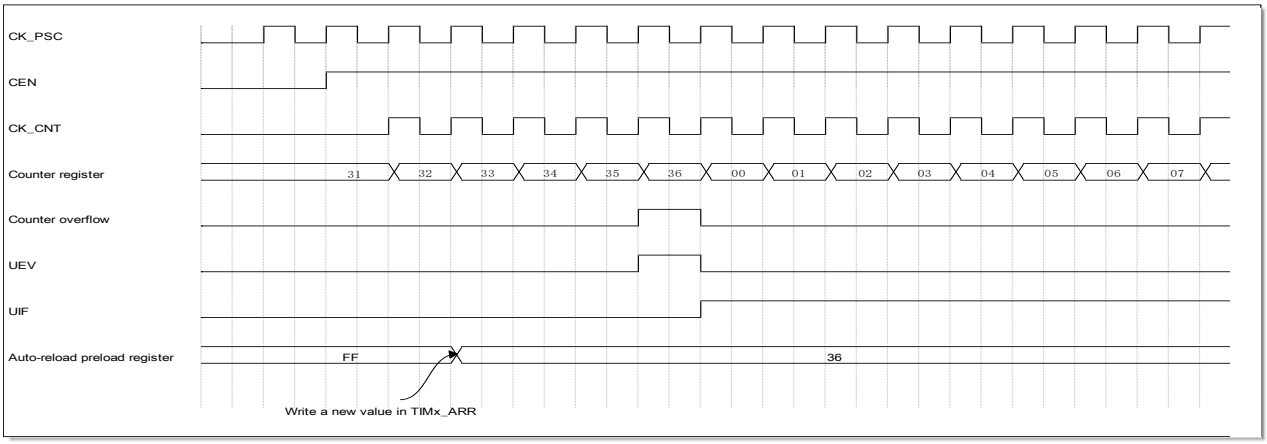


图 19.8 计数器时序图，当 ARPE=0 时的更新事件

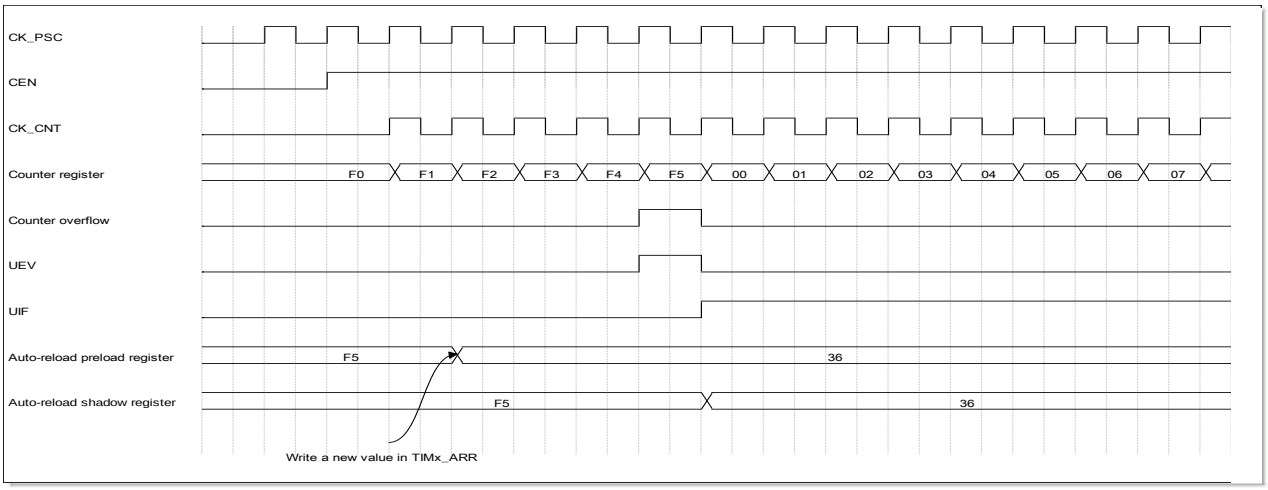


图 19.9 计数器时序图，当 ARPE=1 时的更新事件

19.3.3. 时钟源

计数器时钟可由内部时钟源提供。

CEN（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是实际上的控制位，并且只能被软件修改（除非 UG 位自动被清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示了在不带预分频器时，控制电路和向上计数器在正常模式下的动作。

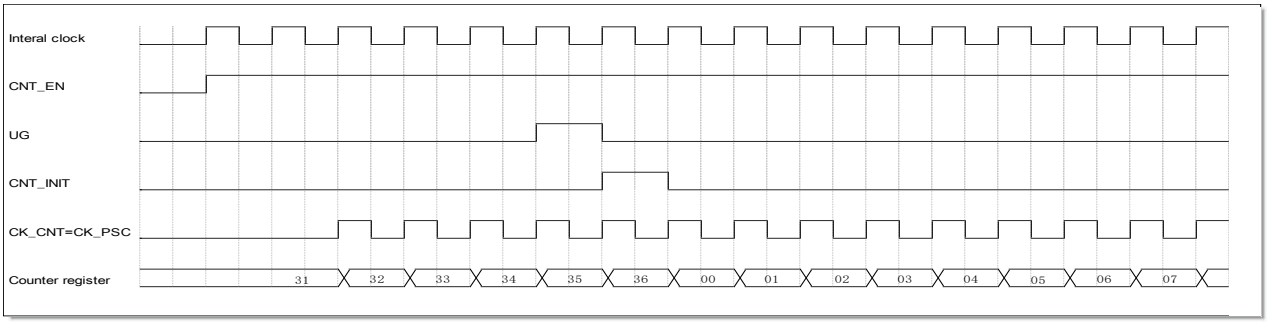


图 19.10 内部时钟分频是 1 时正常模式下的控制时序图

19.3.4. 捕获/比较通道

每一个捕获/比较通道都是包括一个捕获/比较寄存器（包含影子寄存器），一个捕获的输入部分（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）。

输入部分采样相应的 TIx 输入信号，产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿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。该信号紧接着被预分频产生信号 IC1PS。

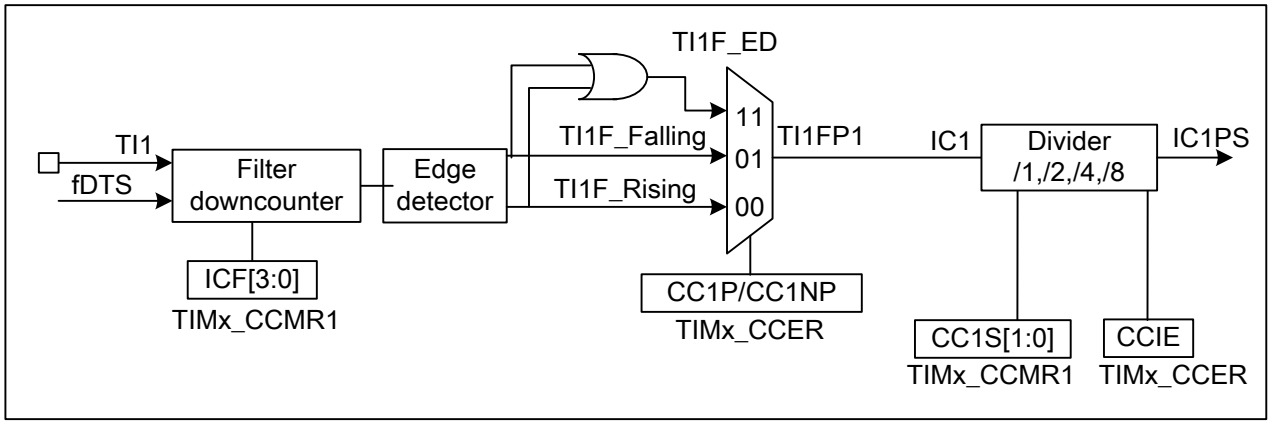


图 19.11 捕获/比较通道（如：通道 1 输入部分）

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。

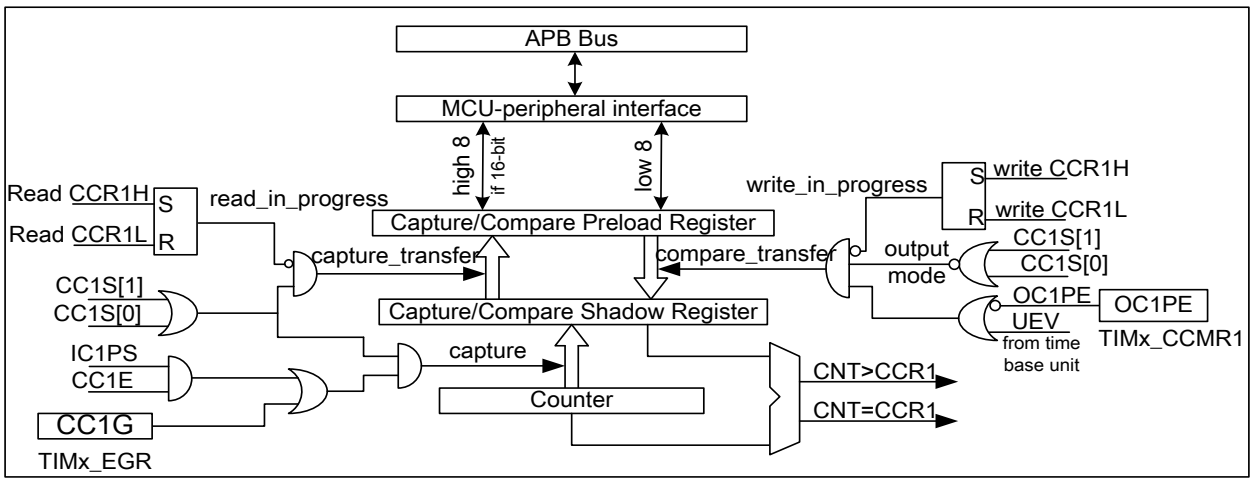


图 19.12 捕获/比较通道 1 主要框图

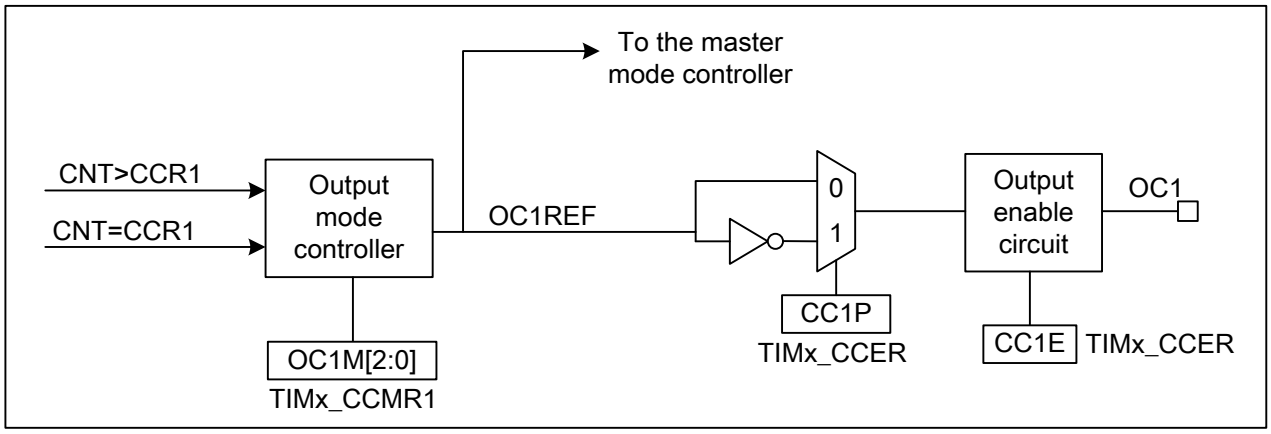


图 19.13 捕获/比较通道的输出部分（通道 1）

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。

19.3.5. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿跳变时，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIMx_SR 寄存器）被置 1，如果使能了中断则产生一个相应的中断。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMx_SR 寄存器）被置 1。软件写写 CCxIF=0 可清楚 CCxIF，或者读取存储在 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：

- 选择有效输入端 :TIMx_CCR1 必须连接到 TI1 输入 ,所以需要写 TIMx_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为 0，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
- 根据连接到计数器的输入信号特点，配置输入滤波器为所需的带宽（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 频率采样）连续采样 8 次，以确认在 TI1 上一次新的边沿变换。然后在 TIMx_CCMR1 寄存器中写 IC1F=0011。
- 配置输入预分频器。在本例中，我们希望在每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
- 写 TIMx_CCER 寄存器的 CCIE=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。

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 标志被置位。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如果设置了 CC1IE 位，则会产生一个中断。

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。

注意：通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断。

19.3.6. 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强制为有效或者无效状态，而不依赖于输出比较寄存器和计数器之间的比较结果。

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强制输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强制为高电平（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 被强制为高电平。

置 TIMx_CCMRx 寄存器中的 OCxM=100，可强制 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。

19.3.7. 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=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）。
- 置位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。

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

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也可以用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   - 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被更新）。下图给出了一个例子。

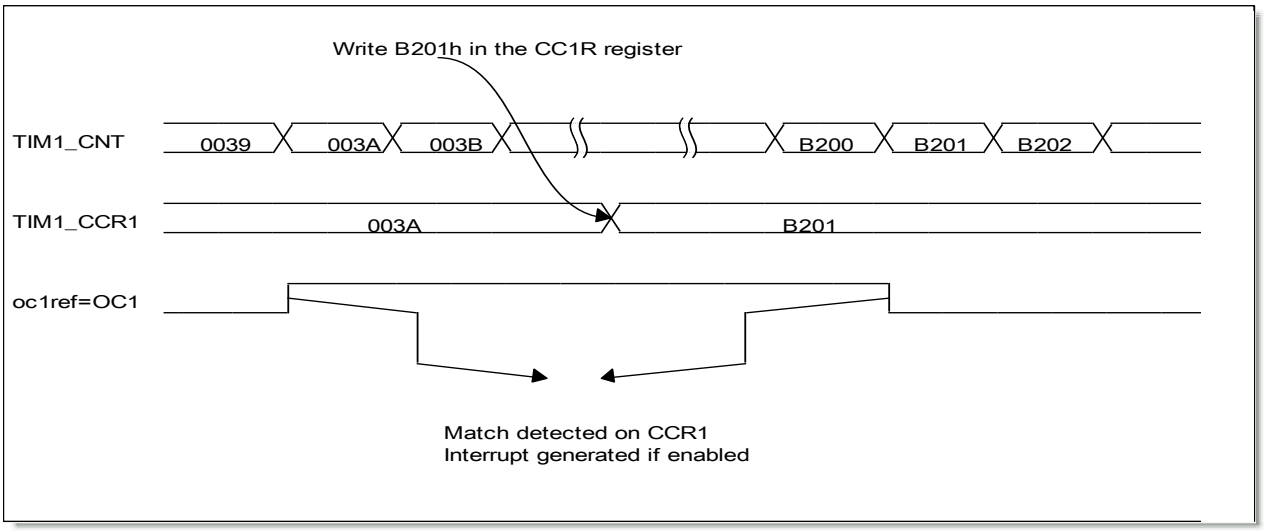


图 19.14 输出比较模式，翻转 OC1

19.3.8. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率，由 TIMx_CCRx 寄存器确定占空比的信号。在 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

在 PWM 模式（模式 1 或模式 2）下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，（依据计数器的计数方向）

以确定是否符合 TIMx_CCRx<=TIMx_CNT 或者 TIMx_CNT<=TIMx_CCRx。

PWM 边沿对齐模式

- 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值(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。

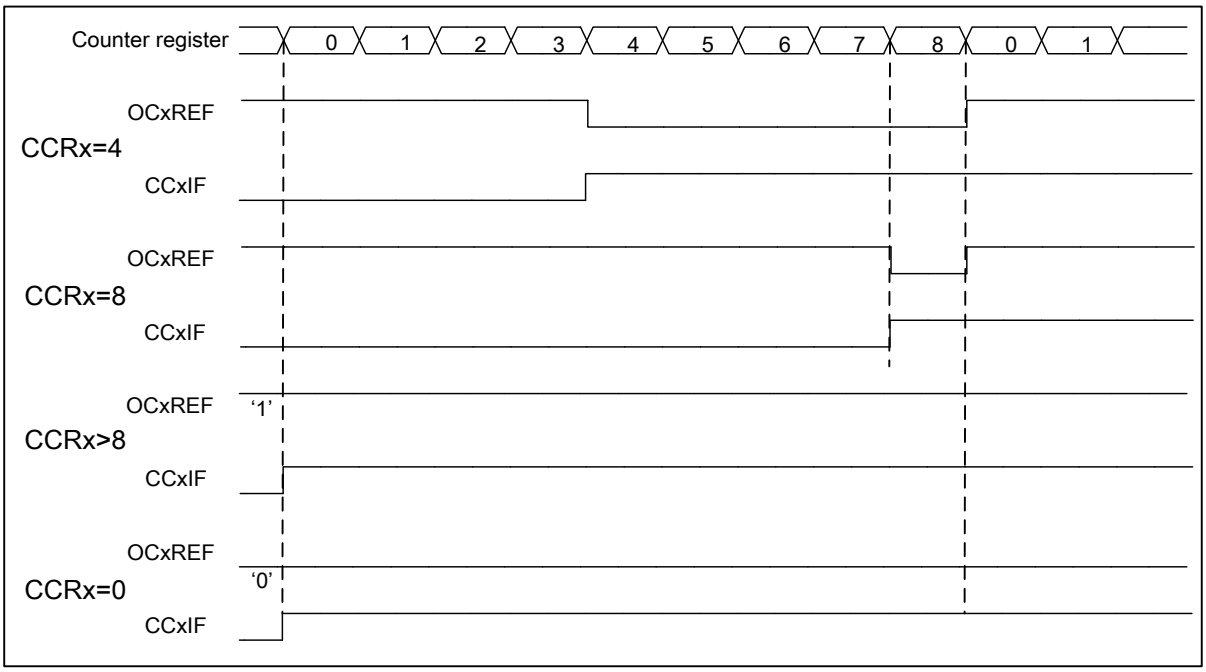


图 19.15 边沿对齐的 PWM 波形（ARR=8）

19.3.9. 调试模式

当微控制器进入调试模式时（Cortex_M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止。

19.4. TIM14 寄存器映射

下表给出了 TIM14 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM14_CR1																							CKD[1:0]		ARPE					URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0	0	0
0x0C	TIM14_DIER																															CC1IE	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0
0x10	TIM14_SR																							CC1OF								CC1IF	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	0	0
0x14	TIM14_EGR																															CC1G	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0
0x18	TIM14_ CCMR1 （输出模式）																										OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0
	TIM14_ CCMR1 （输入模式）																									IC1F[3:0]				IC1PSC [1:0]		CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x20	TIM14_CCER																													CC1NP		CC1P	CC1E
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0
0x24	TIM14_CNT																	CNT[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC																	PSC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR																	ARR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIM14_CCR1																	CCR1[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	TIM14_OR																															TI1_RMP	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0

19.4.1. TIM14 控制寄存器 1（TIM14_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CKD[1:0]	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	ARPE	—				URS	UDIS	CEN
类型	RW	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	CKD[1:0]	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，不要使用此配置
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:3	NA	保留位，未定义
2	URS	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，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 – 软件设置UG位 – 复位触发事件产生的更新 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新 软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出/下溢 – 软件设置UG位 – 复位触发事件产生的更新 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果触发复位模式下触发事件到来时或软件设置UG位，计数器和预

		分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。

19.4.2. TIM14 DMA/中断使能寄存器（TIM14_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—						CC1IE	UIE
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW

Bit	Name	Function
31:2	NA	保留位，未定义
1	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

19.4.3. TIM14 状态寄存器（TIM14_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CC1OF	—
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0	RO-0
7:0	—						CC1IF	UIF
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0	RC_W0

Bit	Name	Function
31:10	NA	保留位，未定义
9	CC1OF	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，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIM1_CCR1寄存器时，CC1IF的状态已经为1。
8:2	NA	保留位，未定义
1	CC1IF	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，但在中心对称模式下除外(参考TIM1_CR1寄存器的CMS位)。它由软件清0。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_ARR的内容时，在向上或向上/向下计数模式时计数器溢出，或向下计数模式时计数器下溢条件下，CC1IF位变高。 如果通道CC1配置为输入模式： 当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIMx_CCR1清0。 0：无输入捕获产生； 1：计数器值已被捕获至TIMx_CCR1(在IC1上检测到与所选极性相同的边沿)。
0	UIF	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，当计数器上溢或下溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时；

19.4.4. TIM14 事件产生寄存器（TIM14_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—						CC1G	UG
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	W	W

Bit	Name	Function
31:2	NA	保留位，未定义
1	CC1G	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： 设置CC1IF=1，若开启对应的中断，则产生相应的中断。 若通道CC1配置为输入： 当前的计数器值被捕获至TIMx_CCR1寄存器，设置CC1IF=1，若开启对应的中断，则产生相应的中断。若CC1IF已经为1，则设置CC1OF=1。
0	UG	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)。

19.4.5. TIM14 捕获/比较模式寄存器（TIM14_CCMR1）

地址偏移：0x18

复位值：0x0000

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

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:7	NA	保留位，未定义
6:4	OC1M[2:0]	输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1的值。OC1REF是高电平有效，而OC1的有效电平取决于CC1P、CC1NP位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。

		101：强制为有效电平。强制OC1REF为高。 110：PWM模式1 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平； 在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)， 否则为有效电平(OC1REF=1)。 111：PWM模式2 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平； 在向下计数时，一旦TIMx_CNT>TIM1_CCRx时通道1为有效电平，否则为无效电平。 注意：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	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，并且新写入的数值立即起作用。 1：开启TIMx_CCR1寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1的预装载值在更新事件到来时被加载至当前寄存器中。
2	OC1FE	输出比较1 快速使能 该位用于加快CC输出对触发输入事件的响应。 0：根据计数器与CCR1的值，CC1正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活CC1输出的最小延时为5个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC被设置为比较电平而与比较结果无关。采样触发器的有效沿和CC1输出间的延时被缩短为3个时钟周期。 注意：OC1FE只在通道被配置成PWM1或PWM2模式时起作用。
1:0	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。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

输入捕获模式：

Bit	Name	Function
31:8	NA	保留位，未定义
7:4	IC1F[3:0]	输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，以$f_{SAMPLING} = f_{CK_INT}$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]	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]	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。

19.4.6. TIM14 捕获/比较使能寄存器（TIM14_CCER）

地址偏移：0x20

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—				CC1NP	—	CC1P	CC1E
类型	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RW	RW

Bit	Name	Function
31:4	NA	保留位，未定义
3	CC1NP	输入捕获/比较1互补输出极性 CC1通道配置为输出： 0：OC1N高电平有效； 1：OC1N低电平有效。 CC1通道配置为输入： 本为用于和CC1P联合定义TI1FP1和TI2FP1的极性。参考CC1P的描述。

		注1：一旦LOCK级别(TIMx_BKR寄存器中的LOCK位)设为2或3且CC1S=00(通道配置为输出) 则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1NP位才从预装载位中取新值。
2	NA	保留位，未定义
1	CC1P	输入捕获/比较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位联合选择在触发或捕获模式下TI1FP1和TI2FP1的有效极性。 00：非反相/上升沿 电路作用于TIxFP1的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1非反相（在门控模式或编码模式）。 01：反相/下降沿 电路作用于TIxFP1的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 10：保留不用 11：非反相/上升或下降沿 电路作用于TIxFP1的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。
0	CC1E	输入捕获/比较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。 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； 0：捕获使能。

CCxE 位	OCx 输出状态
0	禁止输出（OCx=0，OCx_EN=0）
1	OCx=OCxREF + 极性， OCx_EN=1

表格 16-1 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

注意：引脚连接到互补的 OCx 通道的外部 I/O 引脚的状态，取决于 OCx 通道状态和 GPIO 寄存器。

19.4.7. TIM14 计数器（TIM14_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

19.4.8. TIM14 预分频器（TIM14_PSC）

地址偏移：0x28

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

19.4.9. TIM14 自动重装载寄存器（TIM14_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 参考章节13.3.1：时基单元中有关ARR的更新和动作的相关内容。 当自动装载值为空时，计数器不工作。

19.4.10. TIM14 捕获/比较寄存器（TIM14_CCR1）

地址偏移：0x34

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR1[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR1[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR1[15:0]	捕获/比较通道1的值 若CC1通道配置为输出： CCR1决定了装入当前捕获/比较1寄存器的值（预装载值）。 如果在TIMx_CCMR1寄存器（OC1PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输

		至当前捕获/比较1寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC1端口上输出信号。 若CC1通道配置为输出入： CCR1包含由上一次输入捕获1事件（IC1）传输的计数器值。

19.4.11. TIM14 配置选择寄存器（TIM14_OR）

地址偏移：0x50

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
7:0	—						TI1_RMP	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW

Bit	Name	Function
31:2	NA	保留位，未定义
1:0	TI1_RMP[1:0]	定时器输入通道1的重映射 00：TIM14通道1连接GPIO； 01：TIM14通道1连接RTCCLK； 10：TIM14通道1连接HSE/32； 11：TIM14通道1连接微控制器主时钟输出（MCO）。

20. 通用定时器（TIM15/16/17）

20.1. TIM15/16/17 简介

通用定时器（TIM15/16/17）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。
它适合多种用途，包括测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较和 PWM）。
使用定时器预分频器和 RCC 时钟控制预分频器 ,可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。
通用定时器（TIM15/16/17）是完全独立的，它们不共享任何资源。它们可以同步操作。

20.2. TIM15 主要特性

TIM15 定时器的功能包括：

- 16 位向上自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 2 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出（仅通道 1）
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新事件：计数器向上溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入

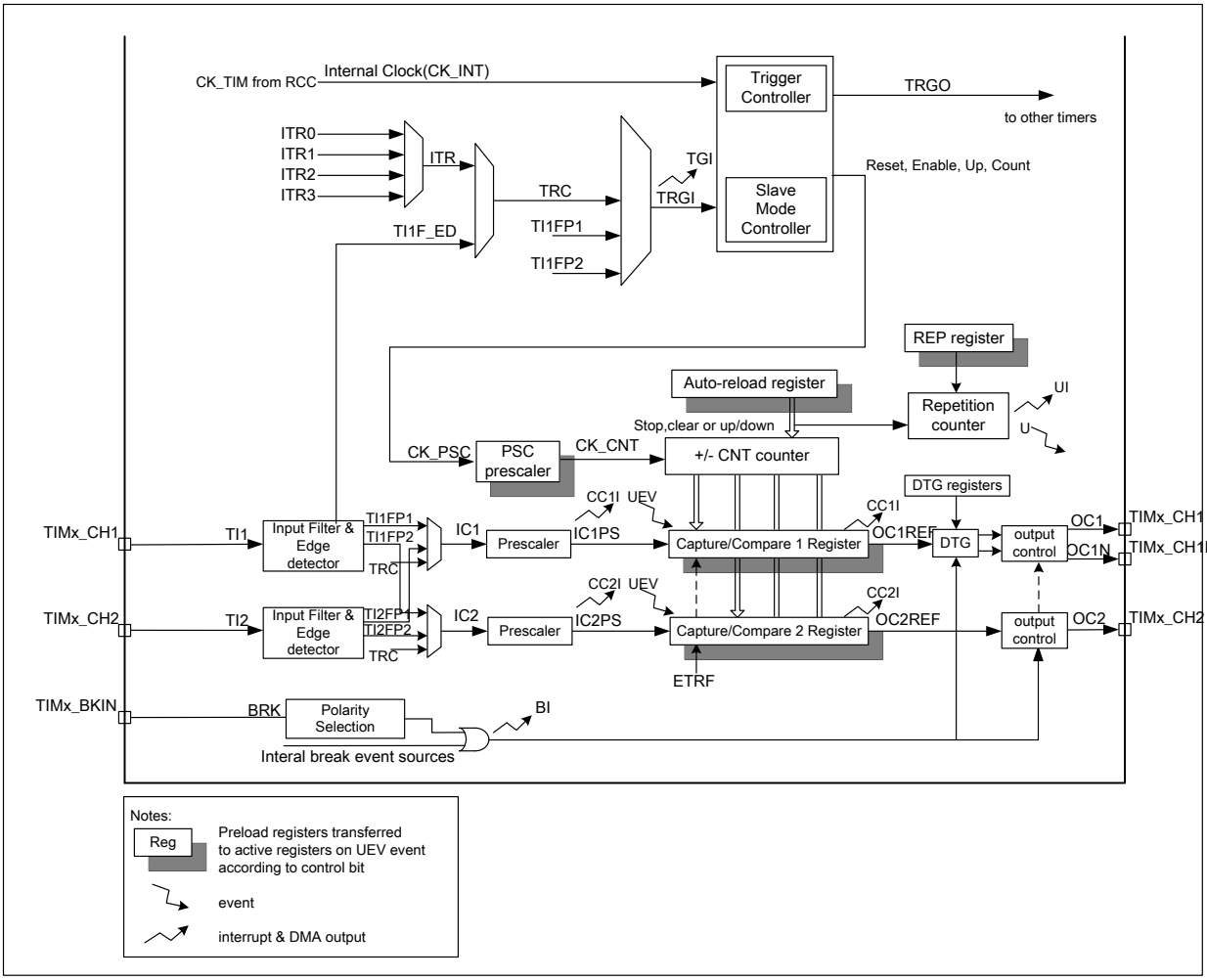


图 20.1 TIM15 定时器框图

20.3. TIM16 和 TIM17 主要特性

通用 TIM16/TIM17 定时器功能包括：

- 16 位向上自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟分频的系数为 1~65535 之间的任意数值
- 1 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或一个已知状态
- 如下事件发生时产生中断/DMA：

- 更新事件：计数器向上溢出，计数器初始化（通过软件或者内部/外部触发）
- 触发事件
- 输入捕获
- 输出比较
- 刹车信号输入

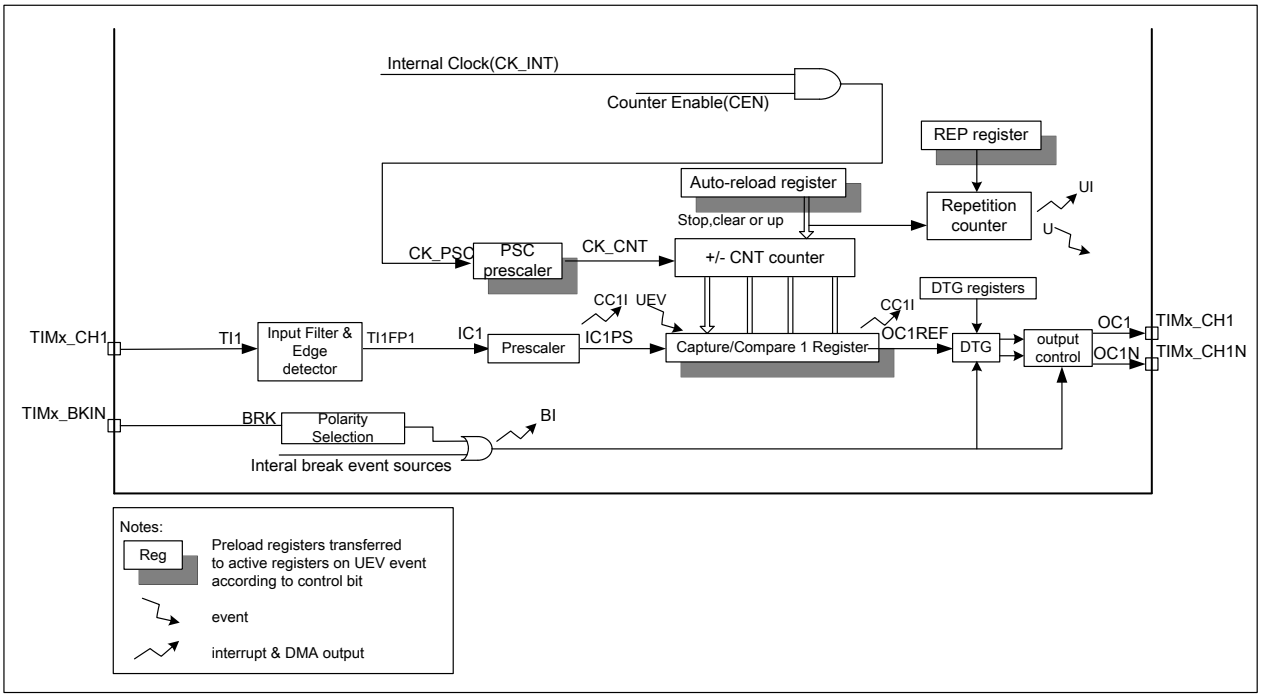


图 20.2 TIM16、TIM17 定时器框图

20.4. TIM15/16/17 功能描述

20.4.1. 时基单元

可编程通用定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数。此计数器时钟由分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

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

自动装载寄存器是预先装载的，写或读自动装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容会被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)

时，CK_CNT 才有效。(更多有关使能计数器的细节，请参考控制器的从模式描述)
注意，在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

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。新的预分频器的参数在下一次更新事件到来时被采用。
图 20.3 和图 20.4 给出了在运行时更改预分频器因子，计数器的相关动作的例子。

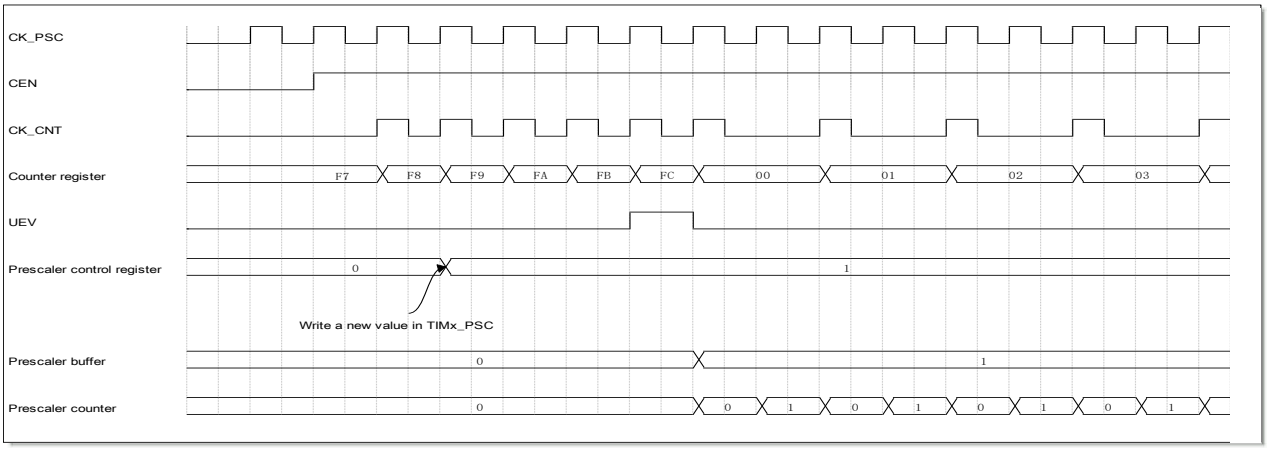


图 20.3 预分频系数从 1 变到 2 时，计数器的时序图

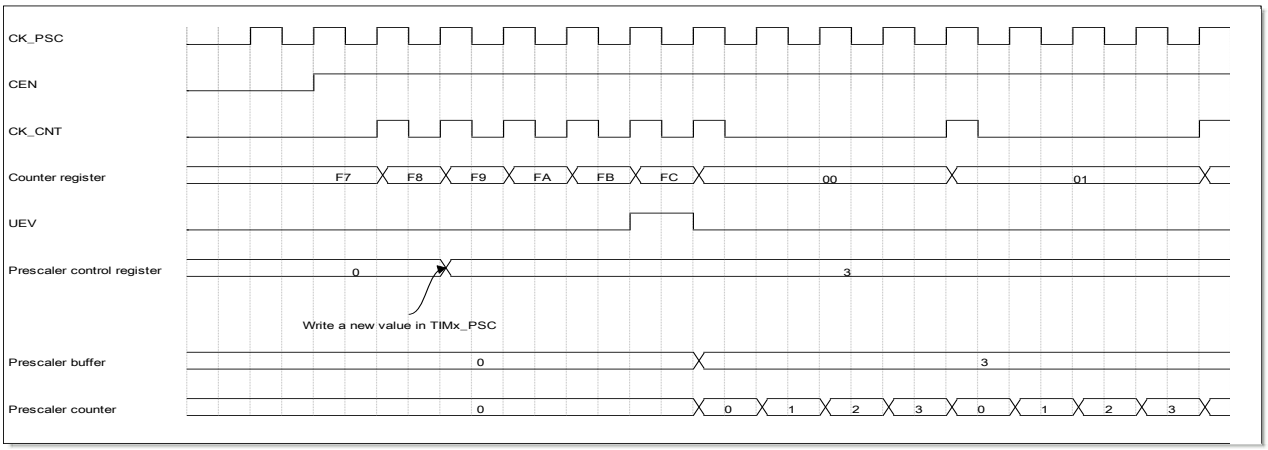


图 20.4 预分频系数从 1 变到 4 时，计数器的时序图

20.4.2. 计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值 (TIMx_ARR 计数器的值)，然后重新从 0 开始计数并且产生一个计数器溢出事件。
如果使用了重复计数器功能，在向上计数达到设置的重复计数次数(TIMx_RCR)时，才产生更新事件(UEV)；

否则每次计数器溢出都会产生更新事件。

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。

通过软件设置 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 UDSI 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清零，同时预分频器也被清零（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）为 1，置位 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寄存器的同时产生更新和捕获中断。

当产生一个更新事件时，所有寄存器都被更新，同时（根据 URS 位）设置更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- 重复计数器被重新加载为 TIMx_RCR 寄存器中的内容。
- 自动装载影子寄存器被重新加载为 TIMx_ARR 中的预装载值。
- 预分频器的缓冲区被重新加载为 TIMx_PSC 中的预装载值。

下面一些时序图展示了当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的计数情况。

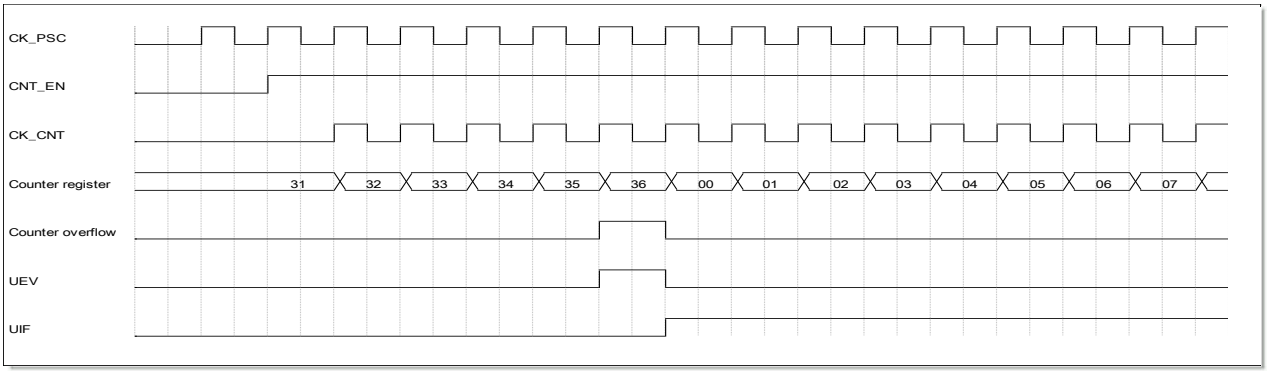


图 20.5 计数器时序图，内部时钟分频因子为 1

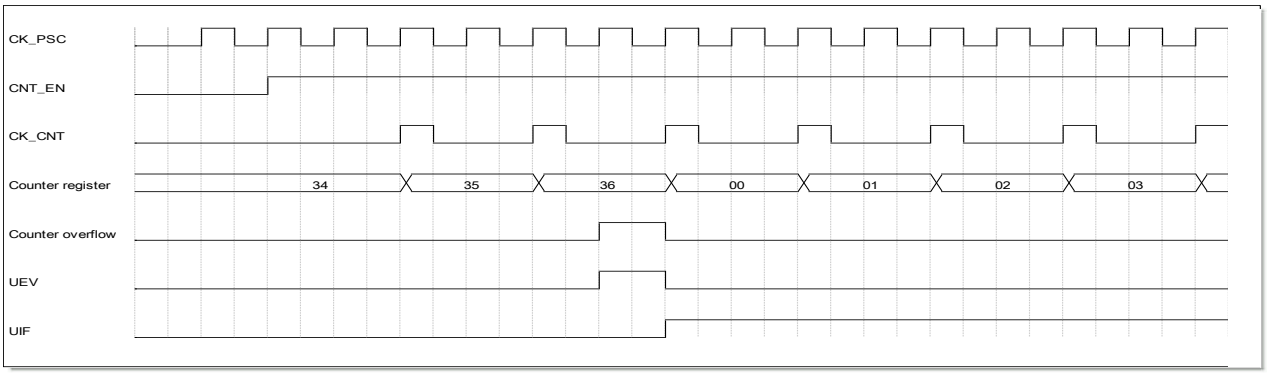


图 20.6 计数器时序图，内部时钟分频因子为 2

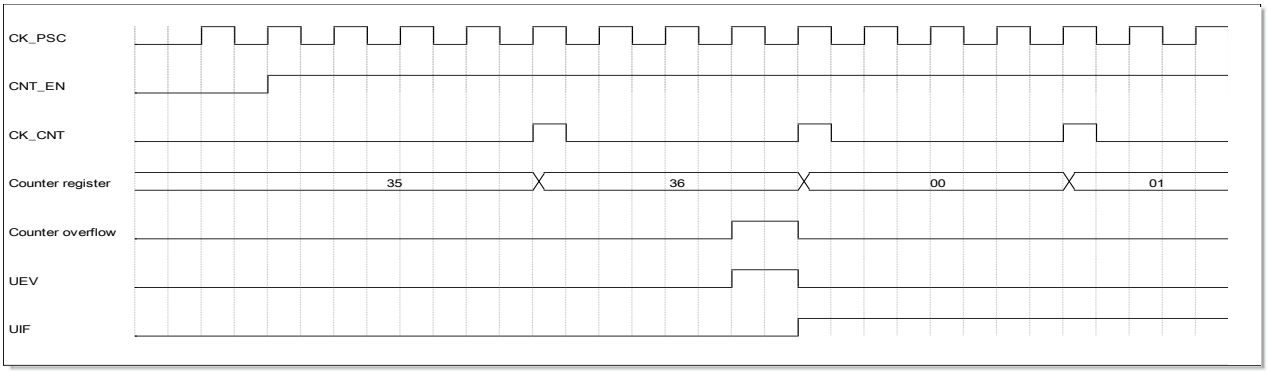


图 20.7 计时器时序图，内部时钟分频因子为 4

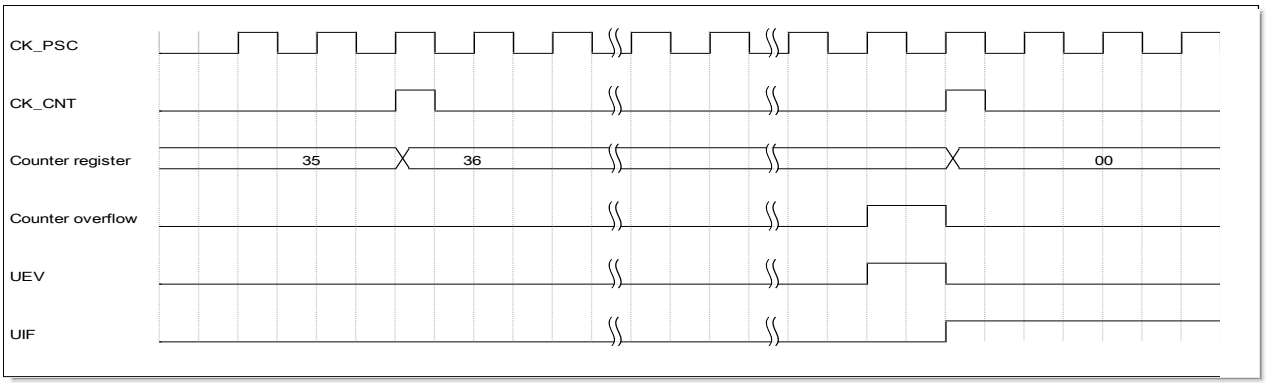


图 20.8 计数器时序图，内部时钟分频因子为 N

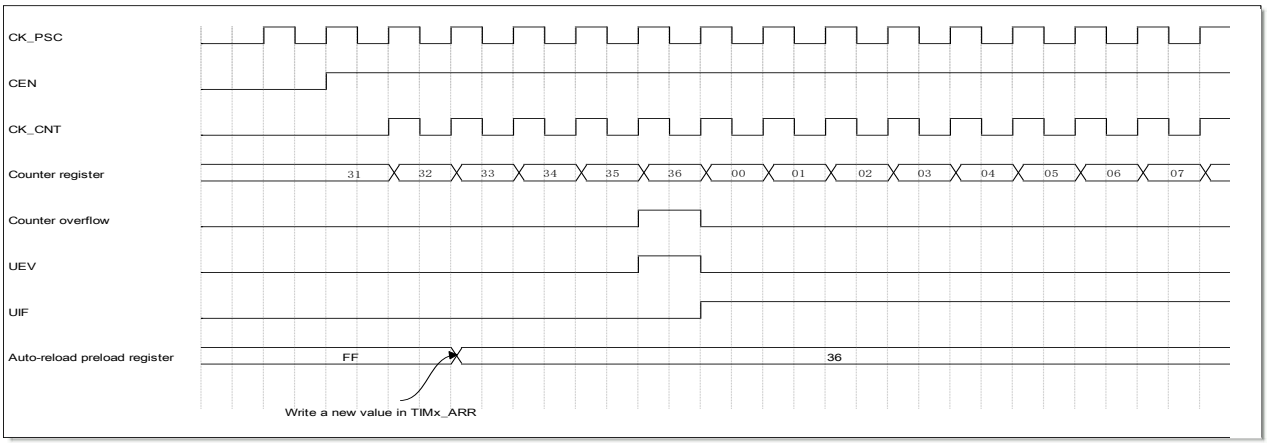


图 20.9 计数器时序图，当 ARPE=0 时的更新事件

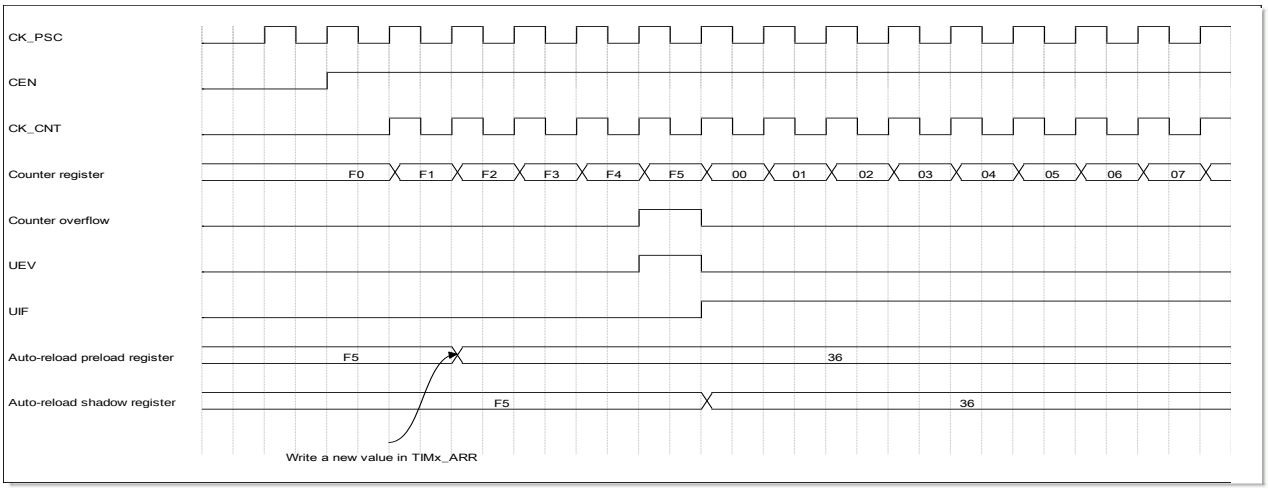


图 20.10 计数器时序图，当 ARPE=1 时的更新事件

20.4.3. 重复计数器

20.4.1 章节的时基单元解释了计数器上溢/下溢时是如何产生更新事件（UEV）的，然而事实上它只能在重复计数器计数到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器（TIMx_ARR 自动重加载寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的 TIMx_CCRx 捕获/比较寄存器），N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时

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

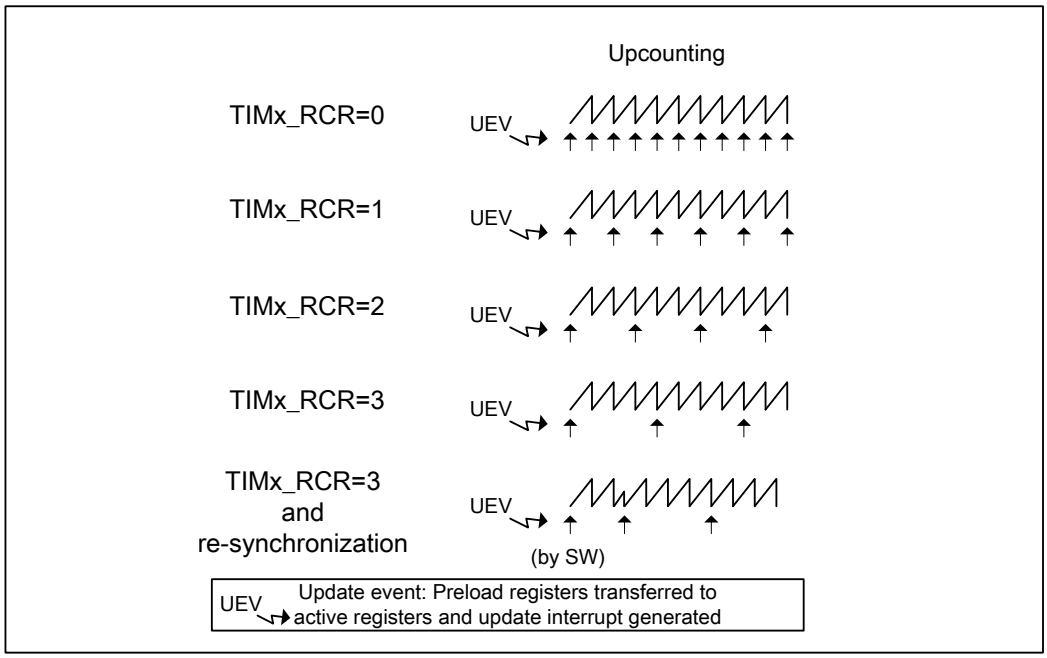


图 20.11 不同 TIMx_RCR 寄存器设置下更新速率的例子

20.4.4. 时钟源

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚
- 内部触发输入（ITRx）（仅 TIM15）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置定时器 TIM1 作为定时器 TIM5 的预分频器。

内部时钟源（CK_INT）

如果禁止了从模式控制器（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是实际上的控制位，并且只能被软件修改（除非 UG 位自动被清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示了在不带预分频器时，控制电路和向上计数器在正常模式下的动作。

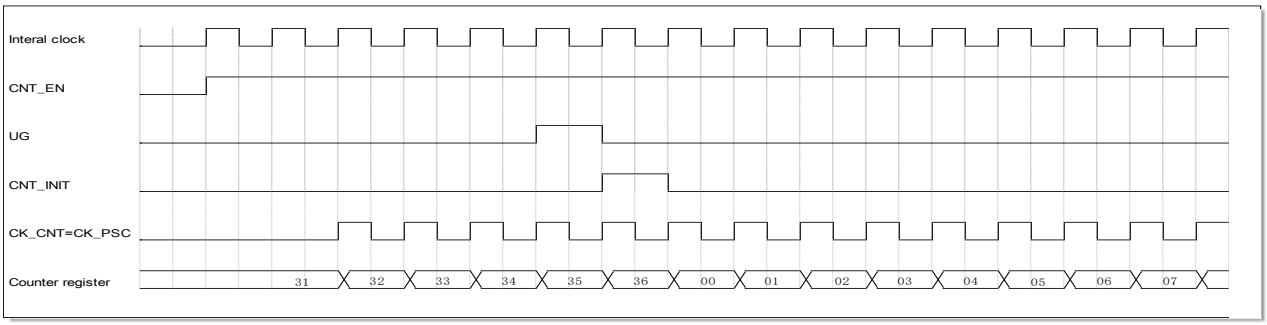


图 20.12 内部时钟分频是 1 时正常模式下的控制时序图

外部时钟源模式 1

当 TIMx_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器在选定输入端的每个上升沿或下降沿计数。

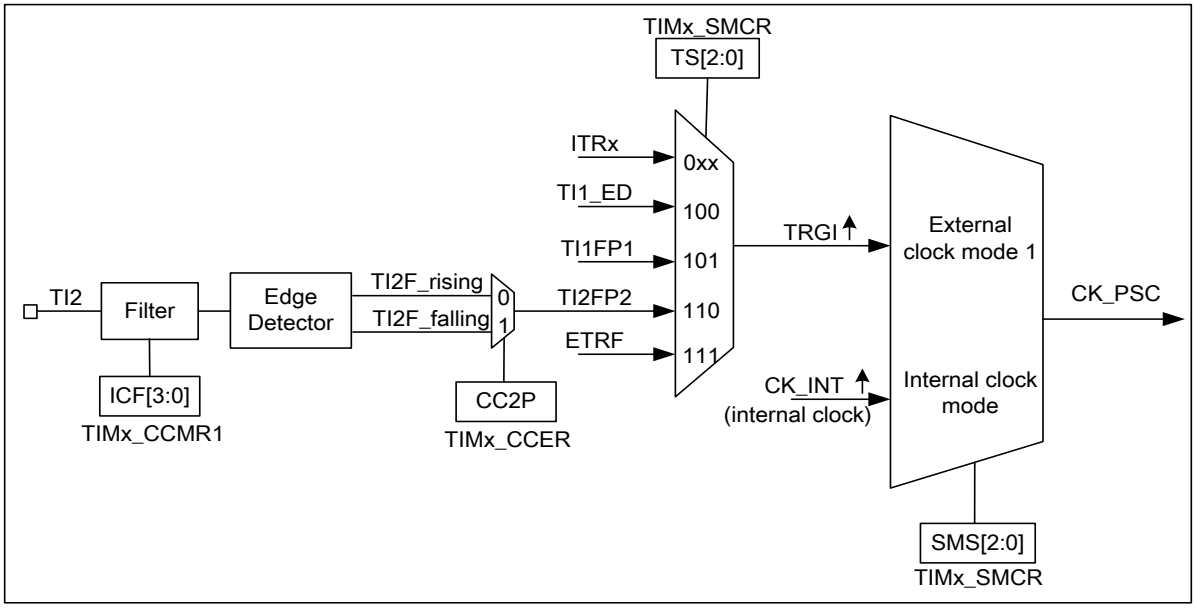


图 20.13 TI2 外部时钟连接示例

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

13. 写 TIMx_CCMR1 寄存器的 CC2S=01，配置通道 2 检测 TI2 输入的上升沿。
14. 写 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）。
15. 写 TIMx_CCER 寄存器的 CC2P=0，选择上升沿极性。
16. 写 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1。
17. 写 TIMx_SMCR 寄存器的 TS=110，选择 TI2 作为触发输入源。
18. 写 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注意： 捕获预分频器不用作触发，所以不需要对它进行配置。

当 TI2 上升沿出现时，计数器计数一次且 TIF 标志被置位。

当 TI2 的上升沿和计数器实际时钟之间的延时取决于在 TI2 输入端的重新同步电路。

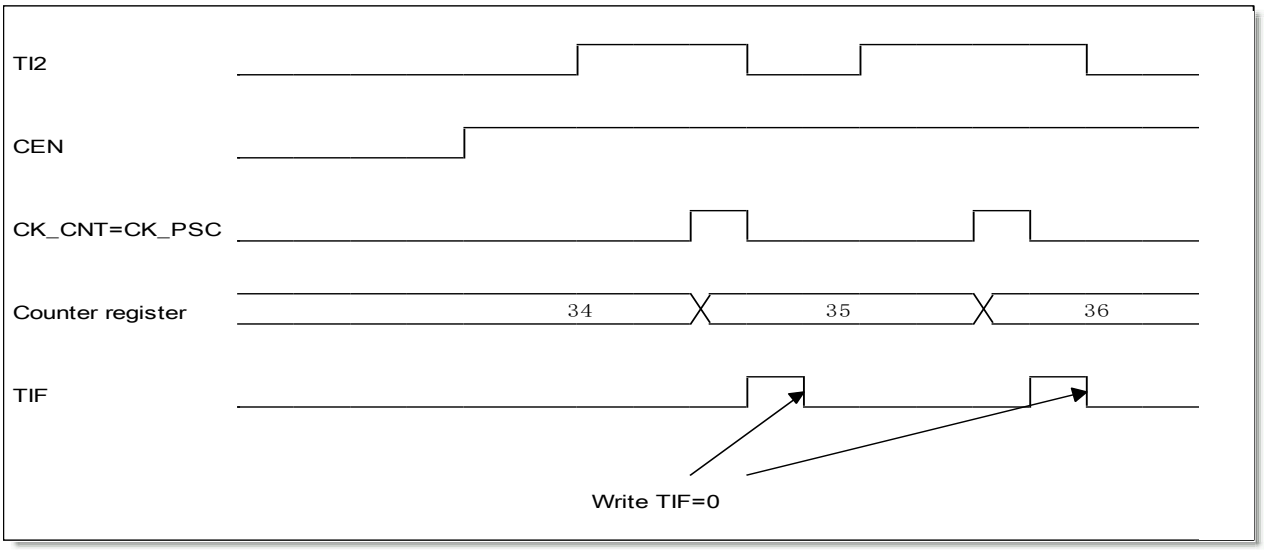


图 20.14 外部时钟模式 1 时的控制时序图

20.4.5. 捕获/比较通道

每一个捕获/比较通道都是包括一个捕获/比较寄存器（包含影子寄存器），一个捕获的输入部分（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）。

输入部分采样相应的 TIx 输入信号，产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿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。该信号紧接着被预分频产生信号 IC1PS。

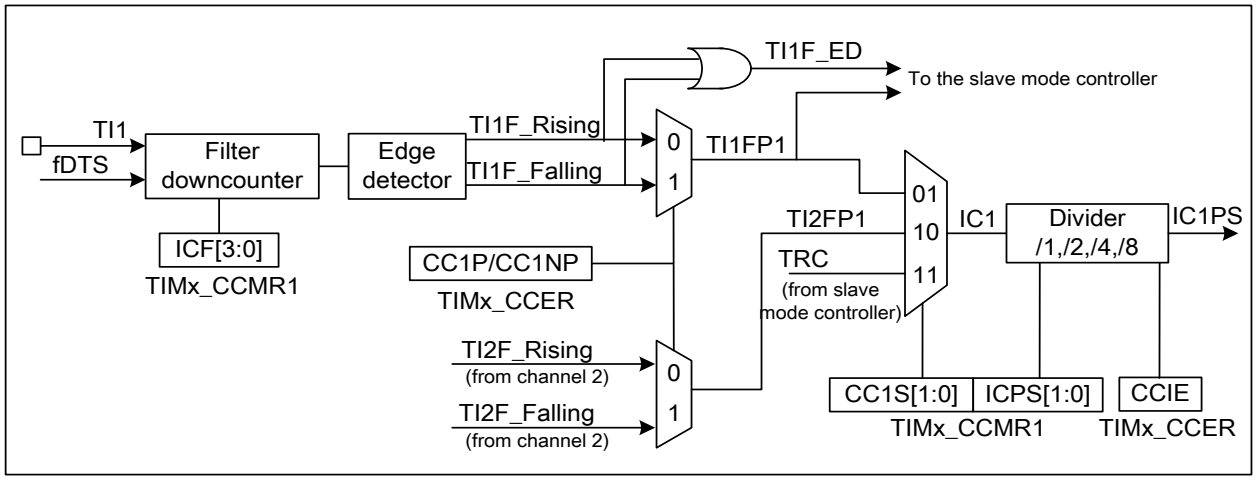


图 20.15 捕获/比较通道（如：通道 1 输入部分）

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。

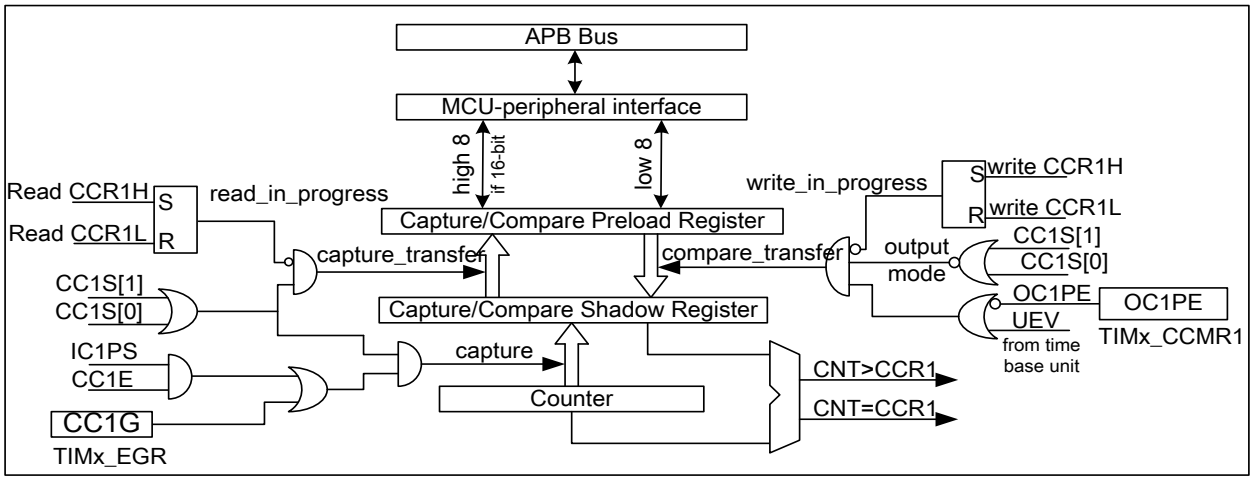


图 20.16 捕获/比较通道 1 主要框

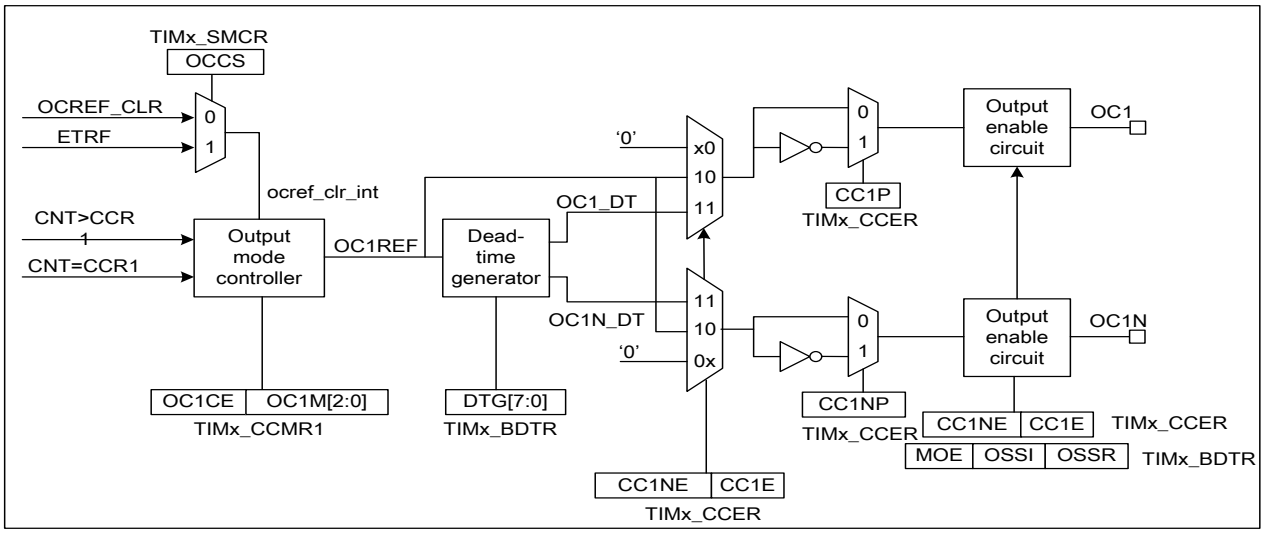


图 20.17 捕获/比较通道的输出部分（通道 1）

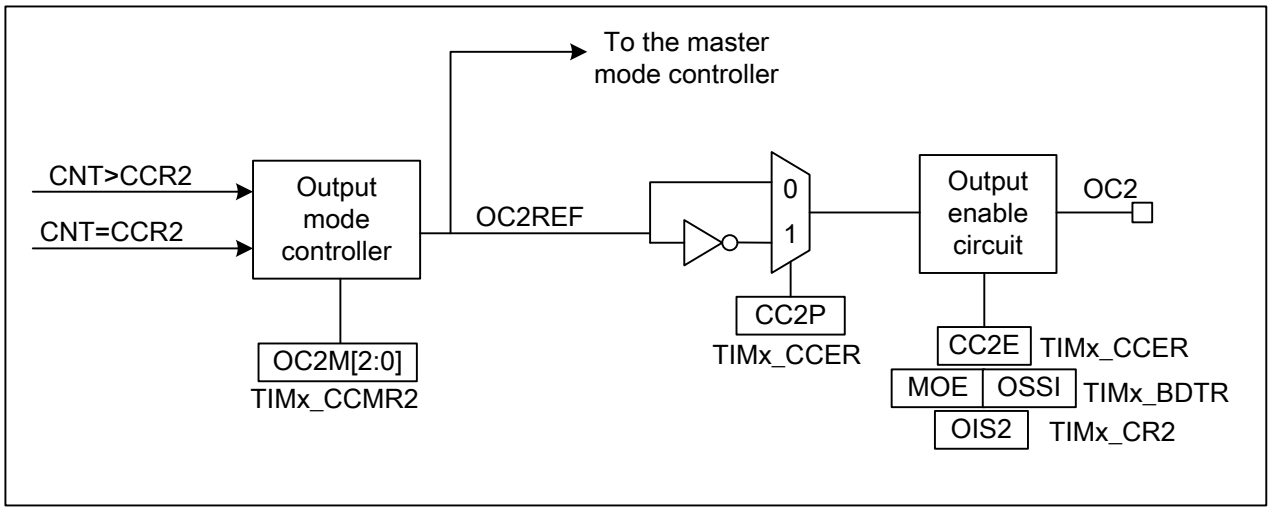


图 20.18 捕获/比较通道的输出部分（TIM15 通道 2）

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。

20.4.6. 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿跳变时，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIMx_SR 寄存器）被置 1，如果使能了中断或者 DMA 则产生一个相应的中断或者一个相应的 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMx_SR 寄存器）被置 1。软件写写 CCxIF=0 可清楚 CCxIF，或者读取存储在 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：

- 选择有效输入端 :TIMx_CCR1 必须连接到 TI1 输入 ,所以需要写 TIMx_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为 0，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
- 根据连接到计数器的输入信号特点，配置输入滤波器为所需的带宽（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 频率采样）连续采样 8 次，以确认在 TI1 上一次新的边沿变换。然后在 TIMx_CCMR1 寄存器中写 IC1F=0011。
- 配置输入预分频器。在本例中，我们希望在每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
- 写 TIMx_CCER 寄存器的 CCIE=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 标志被置位。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如果设置了 CC1IE 位，则会产生一个中断。
- 如果设置了 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。

注意：通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断或 DMA 请求。

20.4.7. PWM 输入模式（仅 TIM15）

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 信号被映射至同一个 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 信号被作为处罚输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的周期(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下（取决于 CK_INT 的频率和预分频器的值）：

- 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 寄存器中的 CC2S=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TIMx_CCR2 中）：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（选中 TI1FP1）。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 寄存器中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中的 CC1E=1 且 CC2E=1。

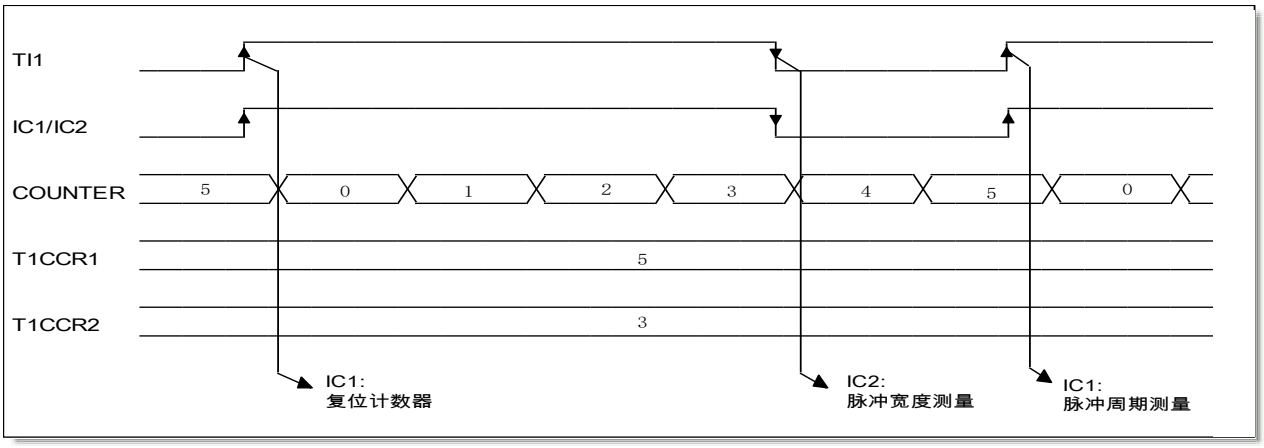


图 20.19 PWM 输入模式时序图

20.4.8. 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强制为有效或者无效状态，而不依赖于输出比较寄存器和计数器之间的比较结果。
置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强制输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强制为高电平（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 被强制为高电平。
置 TIMx_CCMRx 寄存器中的 OCxM=100，可强制 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。

20.4.9. 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=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 ）。
- 置位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。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也可以用来输出一个单脉冲。
输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器 ）。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   - 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。

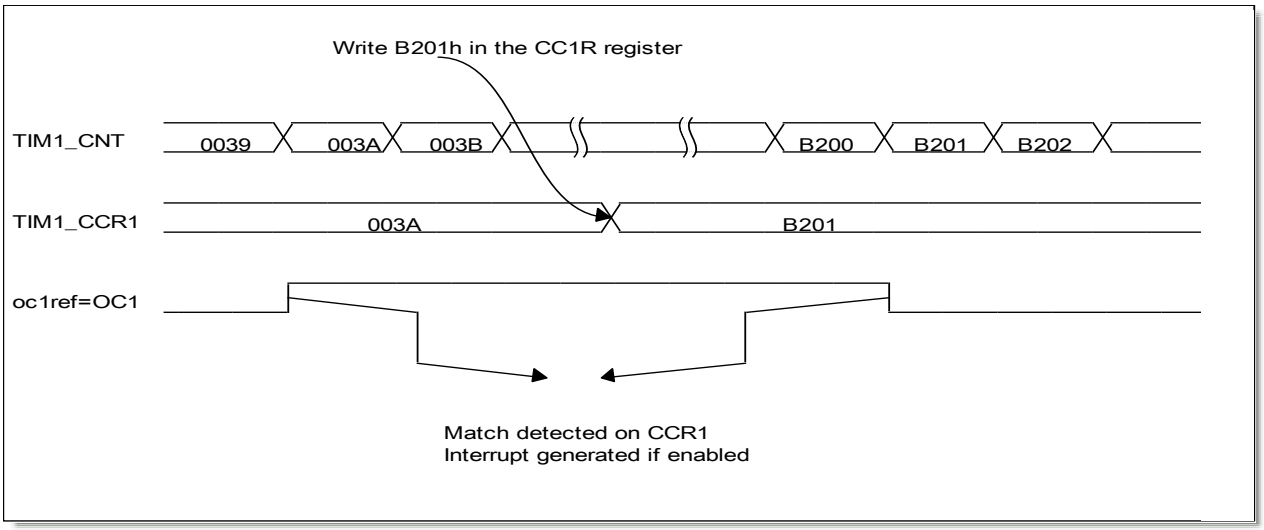


图 20.20 输出比较模式，翻转 OC1

20.4.10. PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率，由 TIMx_CCRx 寄存器确定占空比的信号。在 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

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

PWM 边沿对齐模式

- 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值(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。

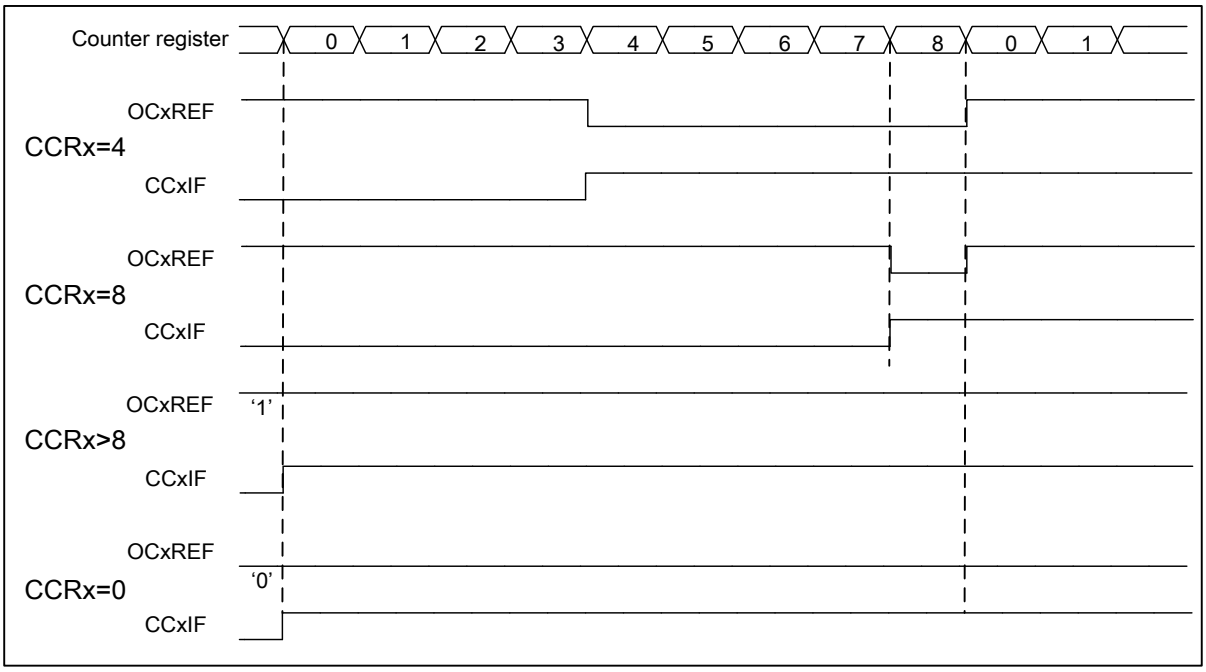


图 20.21 边沿对齐的 PWM 波形（ARR=8）

20.4.11. 互补输出和死区插入

通用定时器（TIM15/16/17）能够输出两路互补信号，并且能够管理输出的瞬时开关和接通。

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

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

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

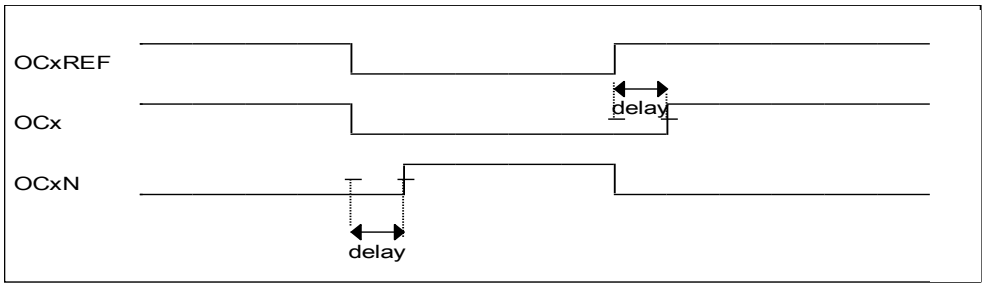


图 20.22 带死区插入的互补输出

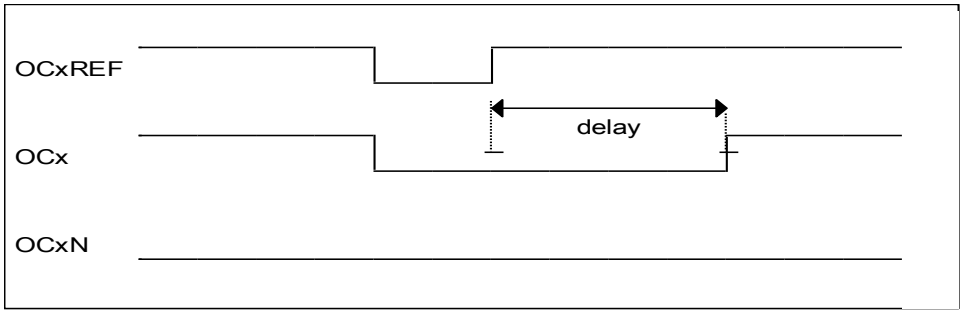


图 20.23 死区波形延迟大于负脉冲

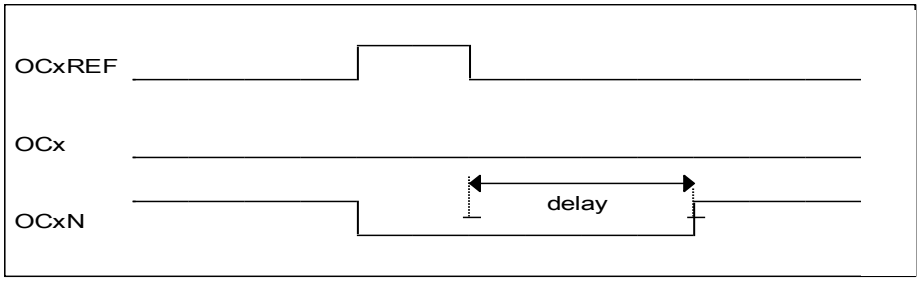


图 20.24 死区波形延迟大于正脉冲

每一个通道的死去延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。详细说明将 TIM1 刹车章节和死区寄存器（TIMx_BDTR）中的延时计算。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强制、输出比较或 PWM），通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注意：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 OCxNE=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。

20.4.12. 使用刹车功能

当使用刹车功能时，依据相应的控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位），输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详细说明见表格：带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。

刹车源既可以是外部刹车输入引脚又可以是下列任意一种内部刹车源：

- 内核的 LOCKUP 输出
- PVD 输出
- CSS 检测到的时钟缺失事件

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，才能正确的读回写入的位。

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出的是同步信号。

当发生刹车时（在刹车输入端出现选定的电平），有下列动作：

- MOE 位被异步清除，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。
- 当使能互补输出时：
  - 输出首先被置于复位状态即无效状态（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电平。注意：因为 MOE 的重新同步，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或者一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置恶劣 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志（TIMx_SR 寄存器中的 BIF 位）为 1 时，则产生一个中断。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次写入 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接到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

刹车由 BRK 输入产生，它的有效极性是可配置的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性）。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，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。

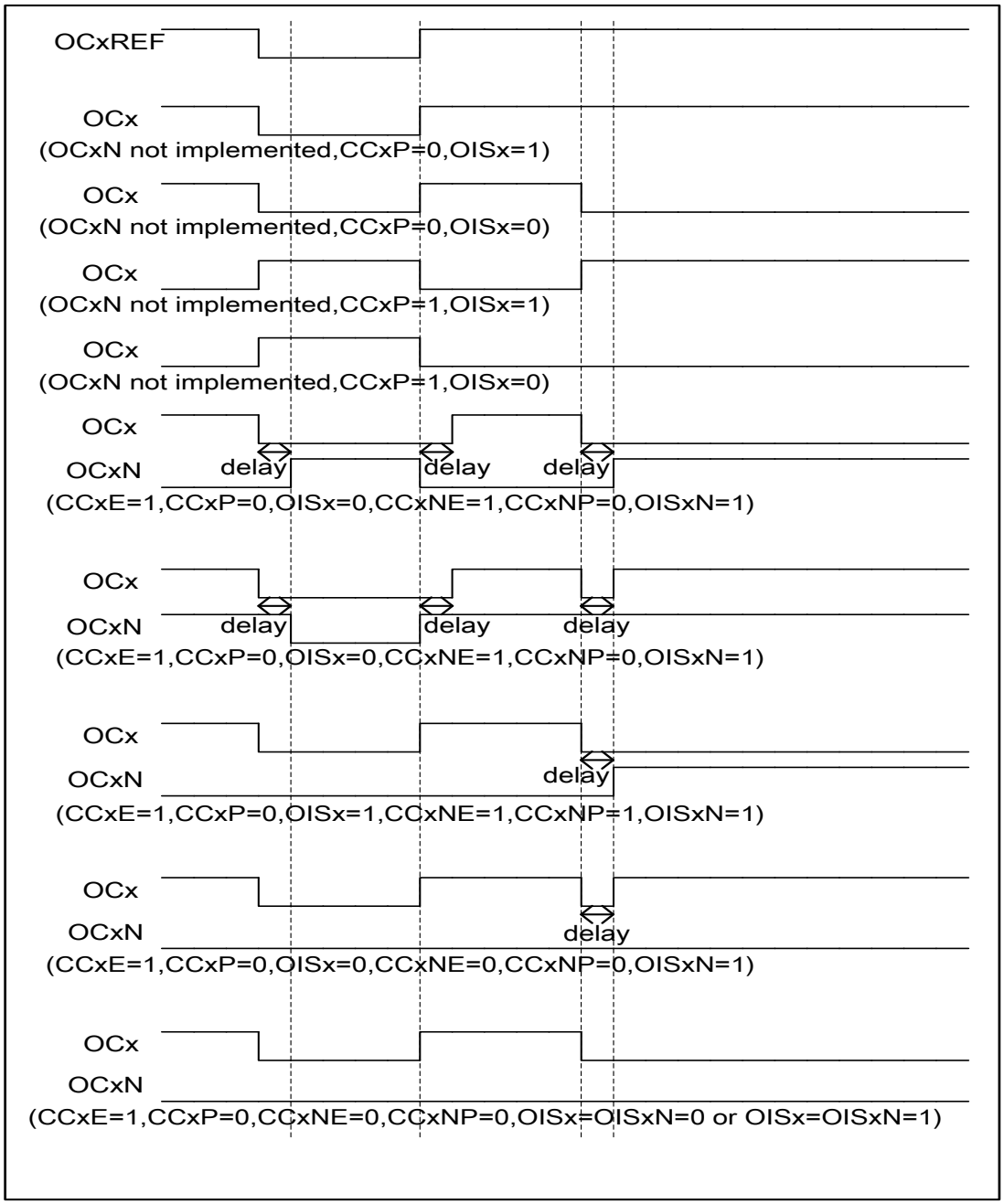


图 20.25 响应刹车的输出时序图

20.4.13. 单脉冲模式

单脉冲模式（OPM）是前述众多模式中的一个特例。这种模式允许计数器通过响应一个激励来启动，并在一个程序可控的延时之后产生一个脉宽程序可控的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器在产生下一个更新事件 UEV 时自动停止。

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

- 在向上计数模式下，计数器 CNT<CCRx<=ARR（特别的，0<CCRx）。
- 在向下计数模式下，计数器 CNT>CCRx。

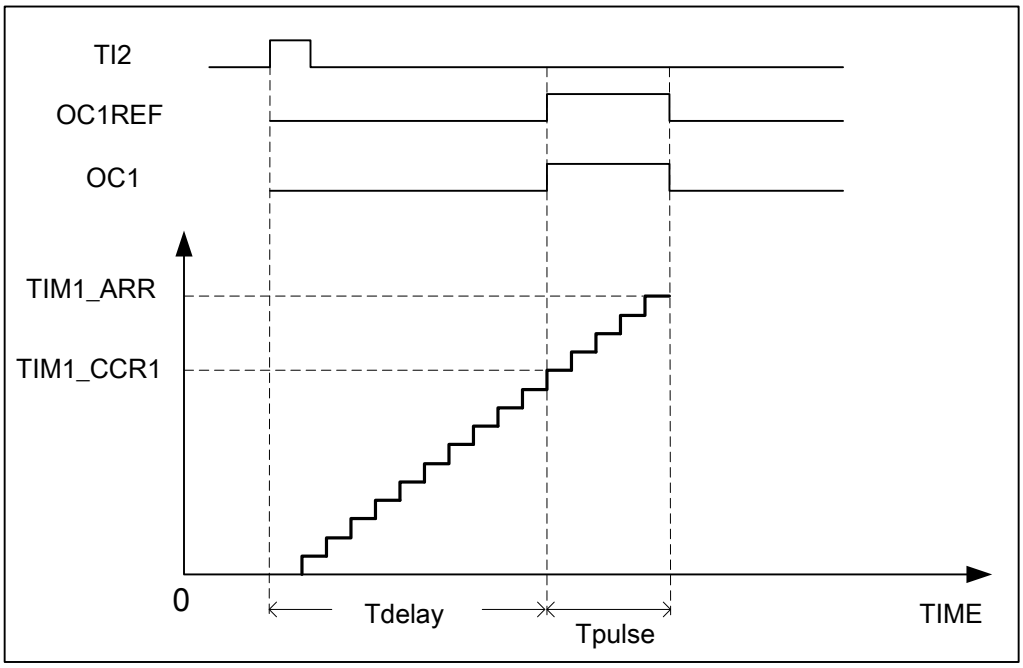


图 20.25 单脉冲模式示意图

例如，需要从 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。假设 TI2FP2 作为触发 1：

- 写 TIMx_CCMR1 寄存器中的 CC2S=01，将 TI2FP2 映像到 TI2 上。
- 写 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- 写 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
- 写 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 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换；首先要写 TIMx_CCMR1 寄存器的 OC1M=1111，选择 PWM 模式 2；根据需要有选择的使能预装载寄存器，写 TIMx_CCMR1 寄存器中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；此时必须向 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动装载值，设置 UG 位来产生一个更新事件，然后等待 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。当 TIMx_CR1 寄存器中的 OPM=0，进入重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但这些操作需要一定的时钟周期，因此限制了可得到的最小延时T_{delay}。

如果需要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF 直接响应激励

而不再依赖比较的结果 ,输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

20.4.14. TIM15 定时器和外部触发的同步

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。

从模式：复位模式

在一个触发输入事件发生时，计数器和它的预分频器被重新初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIMx_ARR、TIMx_CCRx）都被更新。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中的 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)。
- 写 TIMx_SMCR 寄存器中的 SMS=100，配置定时器为复位模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 写 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，并根据 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 上升沿河计数器的实际复位之间的延时取决于 TI1 输入端的再同步电路。

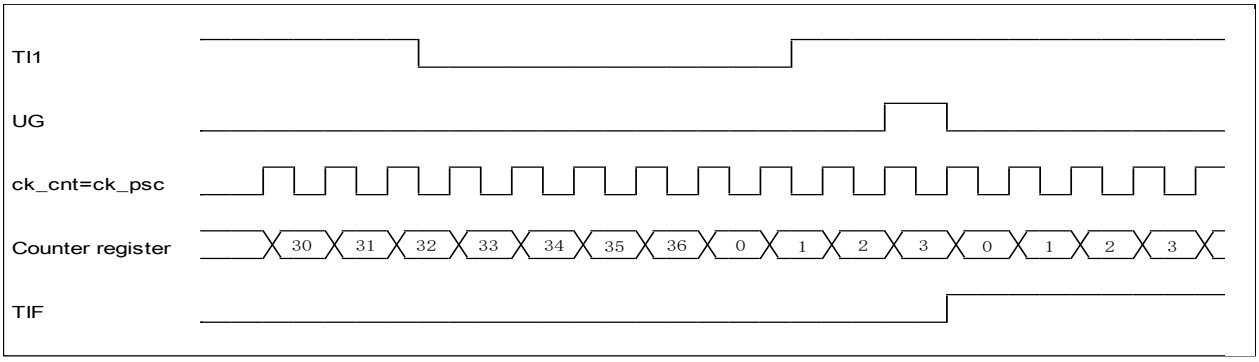


图 20.26 复位模式下的控制时序图

从模式：门控模式

可以按照选中的输入端电平使能计数器。

在如下的例子中，计数器只能在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中 ,不需要滤波器 ,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，写 TIMx_CCMR1 寄存器中的 CC1S=01。写 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 写 TIMx_SMCR 寄存器中的 SMS=101，配置定时器为门控模式；写 TIMx_SMCR 寄存器中的 TS=101，

选择 TI1 作为输入源。

- 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。(在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何)

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都会将 TIMx_SR 寄存器中的 TIF 标志置位。

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

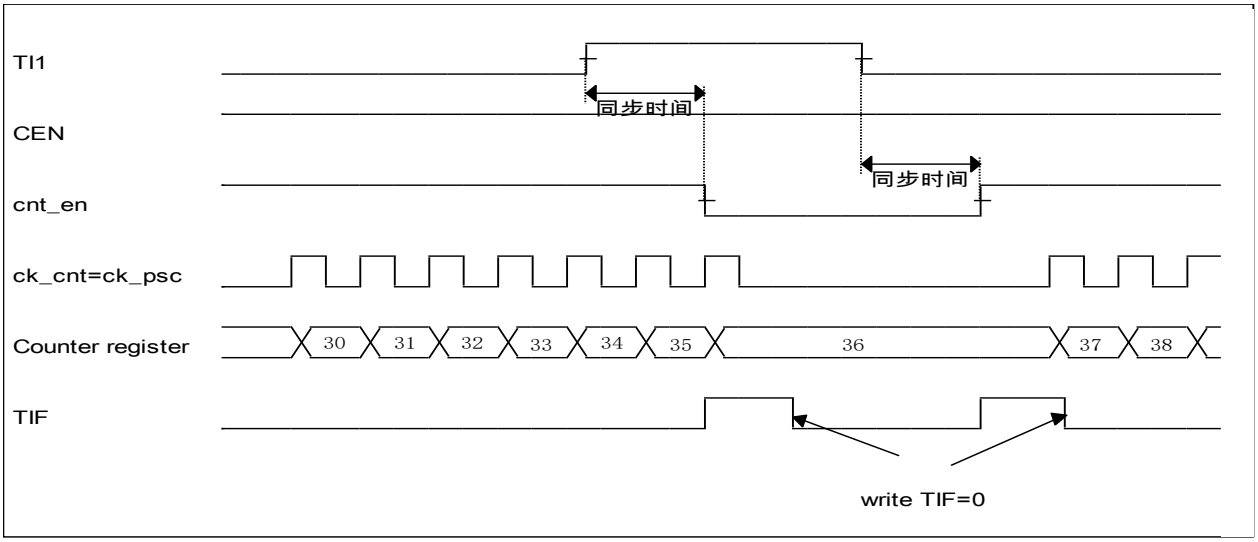


图 13-27 门控模式下的控制时序图

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 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽 (本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，写 TIMx_CCMR1 寄存器中的 CC2S=01。写 TIMx_CCER 寄存器中的 CC2P=1 和 CC2NP=0 以确定极性 (只检测低电平)。
- 写 TIMx_SMCR 寄存器中的 SMS=110，配置定时器为触发模式；写 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，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 输入端的再同步电路。

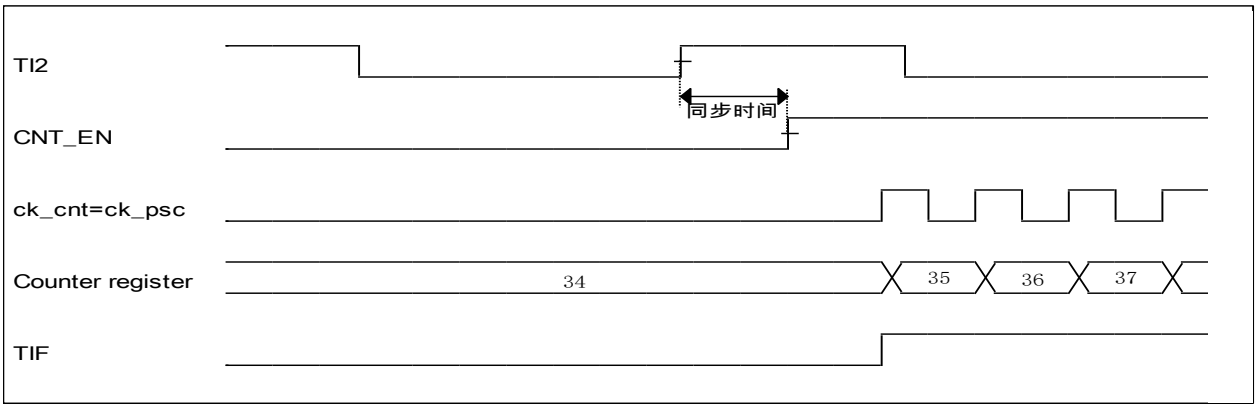


图 20.28 触发模式下的控制时序图

20.4.15. 定时器同步

TIM 定时器在内部相连，用于定时器的同步或链接。详细说明参考章节 16.3.15：定时器同步。

20.4.16. 调试模式

当微控制器进入调试模式时（Cortex_M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止。

20.5. TIM15 寄存器映射

下表给出了 TIM15 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIM15_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CKD[1:0]		ARPE	-	-	-	OPM	URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	0	0	0	0
0x04	TIM15_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OIS2	OIS1N	OIS1	-	MMS[2:0]			CCDS	CCUS	-	CCPC
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	x	0	0	0	0	0	x	0
0x08	TIM15_SMCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MSM	TS[2:0]			-	SMS[2:0]		
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	x	0	0	0
0x0C	TIM15_DIER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TDE	-	-	-	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	-	-	CC2IE	CC1IE	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	0	0	0	0	0	0	x	x	0	0	0
0x10	TIM15_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC2OF	CC1OF	-	BIF	TIF	COMIF	-	-	CC2IF	CC1IF	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x	0	0	0	x	x	0	0	0
0x14	TIM15_EGR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BG	TG	COMG	-	-	CC2G	CC1G	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	0	0	0

0x18	TIM15 _CCMR1（输出模式）																		OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]			OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0								
	TIM15 _CCMR1（输入模式）																		IC2F[3:0]				IC2PSC [1:0]		CC2S[1:0]			IC1F[3:0]			IC1PSC [1:0]			CC1S[1:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x20	TIM15_CCER																									CC2NP		CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	0	0								
0x24	TIM15_CNT	CNT[15:0]																																								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x28	TIM15_PSC	PSC[15:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x2C	TIM15_ARR	ARR[15:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1								
0x30	TIM15_RCR	REP[7:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0								
0x34	TIM15_CCR1	CCR1[15:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x38	TIM15_CCR2	CCR2[15:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x44	TIM15_BDTR	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]											DTG[7:0]																						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								
0x48	TIM15_DCR	DBL[4:0]																								DBA[4:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0								
0x4C	TIM15_DMAR	DMAB[15:0]																																							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								

20.5.1. TIM15 控制寄存器 1（TIM15_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CKD[1:0]	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	ARPE	—			OPM	URS	UDIS	CEN
类型	RW	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	CKD[1:0]	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，不要使用此配置
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:5	NA	保留位，未定义
4	DIR	计数方向 0：计数器向上计数； 1：计数器向下计数。 注意：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	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	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，则下述任一事件产生一个更新中断或DMA请求： – 计数器上溢 – 软件设置UG位 – 复位触发事件产生的更新 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新

		软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出 – 软件设置UG位 – 复位触发事件产生的更新 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果触发复位模式下触发事件到来时或软件设置UG位，计数器和预分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。 注意：在软件设置了CEN位后，外部时钟和门控模式才能工作。而触发模式下可以自动地通过硬件设置CEN位。

20.5.2. TIM15 控制器 2（TIM15_CR2）

地址偏移：0x04

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—					OIS2	OIS1N	OIS1
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
7:0	—	MMS[2:0]			CCDS	CCUS	—	CCPC
类型	RO-0	RW	RW	RW	RW	RW	RO-0	RW

Bit	Name	Function
31:11	NA	保留位，未定义
10	OIS2	输出空闲状态2(OC2输出)。参见OIS1位。
9	OIS1N	输出空闲状态1(OC1N输出)。 0：当MOE=0时，则在一个死区时间后，OC1N=0； 1：当MOE=0时，则在一个死区时间后，OC1N=1。 注意：已经设置了LOCK(TIM1_BKR寄存器)级别1、2或3后，该位不能被修改。
8	OIS1	输出空闲状态1(OC1输出)。 0：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=0； 1：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=1。 注意：已经设置了LOCK(TIM1_BKR寄存器)级别1、2或3后，该位不能被修改。
7	NA	保留位，未定义
6:4	MMS[2:0]	主模式选择 这3位用于选择在主模式下送到其它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)。如果触发输入(时钟/触发控制器配置为复位模式)产生复位，则TRGO上的信号相对实

		际的复位会有一个延迟。 001：使能 – 计数器使能信号被用于作为触发输出(TRGO)。其用于启动多个定时器以便控制在一段时间内使能从定时器。计数器使能信号是通过CEN控制位和门控模式下的触发输入信号的逻辑或产生。除非选择了主/从模式(见TIM1_SMCR寄存器中MSM位的描述)， 当计数器使能信号受控于触发输入时，TRGO上会有一个延迟。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲(MATCH1) – 一旦发生一次捕获或一次比较成功，当CC1IF标志被置1时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF信号被用于作为触发输出(TRGO)。 110：保留。 111：保留。
3	CCDS	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	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。 注意：该位只对具有互补输出的通道起作用。
1	NA	保留位，未定义
0	CCPC	捕获/比较预装载控制位 0：CCxE，CCxNE，CCxP，CCxNP和OCxM位不是预装载的； 1：CCxE，CCxNE，CCxP，CCxNP和OCxM位是预装载的；设置该位后，只在设置了COMG位或TRGI检测到上升沿（取决于CCUS位的设置）后被更新。 注意：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）

地址偏移：0x08

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	MSM	TS[2:0]			—	SMS[2:0]		
类型	RW	RW	RW	RW	RO-0	RW	RW	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	MSM	主/从模式 0：无作用； 1：触发输入(TRGI)上的事件被延迟了，以允许当前定时器与它的从定时器间的完美同步(通过TRGO)。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	触发选择 这3位选择用于选择同步计数器的触发输入。 000：保留 001：内部触发1（ITR0） 010：内部触发2（ITR2） 011：内部触发3（ITR3） 100：TI1的边沿检测器(TI1F_ED) 101：滤波后的定时器输入1(TI1FP1) 110：滤波后的定时器输入2(TI2FP2) 111：外部触发输入(ETRF) 注意：这些位只能在未用到(如SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	NA	保留位，未定义
2:0	SMS[2:0]	时钟/触发/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：时钟/触发控制器禁止 – 如果CEN=1，则预分频器直接由内部时钟驱动。 100：复位模式 – 在选中的触发输入(TRGI)的上升沿时重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入TRGI的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果TI1F_ED被选为触发输入(TS=100)时，不要使用门控模式。这是因为TI1F_ED在每次TI1F变化时只是输出一个脉冲，然而门控模式是要检查触发输入的电平。

表 20.3 TIMx 内部触发连接

从定时器	ITR0（TS=001）	ITR2（TS=010）	ITR3（TS=011）
TIM15	TIM3	TIM16_OC	TIM17_OC

20.5.4. TIM15 DMA/中断使能寄存器（TIM15_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	保留位	TDE	COMDE	保留位			CC1DE	UDE
类型	RO-0	RW	RO-0	RO-0	RO-0	RW	RW	RW
7:0	BIE	TIE	COMIE	—		CC2IE	CC1IE	UIE
类型	RW	RW	RW	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	TDE	允许触发 DMA 请求 0：触发 DMA 请求禁止 1：触发 DMA 请求允许
13	COMDE	COM 换相事件 DMA 请求使能 0：换相事件 DMA 请求禁止 1：换相事件 DMA 请求允许
12:10	NA	保留位，未定义
9	CC1DE	捕获/比较 1 DMA 请求使能 0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
8	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允许
7	BIE	刹车中断使能 0：刹车中断禁止 1：刹车中断允许
6	TIE	触发中断使能 0：触发中断禁止 1：触发中断允许
5	COMIE	COM 换相事件中断使能 0：COM 中断禁止 1：COM 中断允许
4:3	NA	保留位，未定义
2	CC2IE	捕获/比较 2 中断使能 0：CC2 中断禁止 1：CC2 中断允许

1	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

20.5.5. TIM15 状态寄存器（TIM15_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—					CC2OF	CC1OF	—
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0	RC_W0	RO-0
7:0	BIF	TIF	COMIF	—		CC2IF	CC1IF	UIF
类型	RC_W0	RC_W0	RC_W0	RO-0	RO-0	RC_W0	RC_W0	RC_W0

Bit	Name	Function
31:11	NA	保留位，未定义
10	CC2OF	捕获/比较2重复捕获标志 参见CC1OF描述。
9	CC1OF	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，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIMx_CCR1寄存器时，CC1IF的状态已经为1。
8	NA	保留位，未定义
7	BIF	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	TIF	触发器中断标志 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在TRGI输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置1。它由软件清0。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	COM中断标志 一旦产生COM事件(当捕获/比较控制位：CCxE、CCxNE、OCxM已被更新)该位由硬件置1。它由软件清0 。

		0：无COM事件产生； 1：COM中断等待响应。
4:3	NA	保留位，未定义
2	CC2IF	捕获/比较2中断标志 参考CC1IF描述。
1	CC1IF	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，但在中心对称模式下除外(参考TIM1_CR1寄存器的CMS位)。它由软件清0。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_ARR的内容时，在向上或向上/向下计数模式时计数器溢出，或向下计数模式时计数器下溢条件下，CC1IF位变高。 如果通道CC1配置为输入模式： 当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIMx_CCR1清0。 0：无输入捕获产生； 1：计数器值已被捕获至TIMx_CCR1(在IC1上检测到与所选极性相同的边沿)。
0	UIF	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，当计数器上溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当计数器CNT被触发事件重新初始化时 (参考从模式控制寄存器TIMx_SMCR)。

20.5.6. TIM15 事件产生寄存器（TIM15_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	BG	TG	COMG	—		CC2G	CC1G	UG
类型	W	W	W	RO-0	RO-0	W	W	W

Bit	Name	Function
31:8	NA	保留位，未定义
7	BG	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	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，若开启对应的中断和DMA，则产生相应的中断和DMA。
5	COMG	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、CCxP、CCxNP、OCIM位。 注意：该位只对拥有互补输出的通道有效。
4:3	NA	保留位，未定义
2	CC2G	产生捕获/比较2事件 参考CC1G描述。
1	CC1G	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： 设置CC1IF=1，若开启对应的中断和DMA，则产生相应的中断和DMA。 若通道CC1配置为输入： 当前的计数器值被捕获至TIMx_CCR1寄存器，设置CC1IF=1，若开启对应的中断和DMA ,则产生相应的中断和DMA。若CC1IF已经为1 ,则设置CC1OF=1。
0	UG	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)。若在中心对称模式下或DIR=0(向上计数)则计数器被清0；若DIR=1(向下计数)则计数器取TIMx_ARR的值。

20.5.7. TIM15 捕获/比较模式寄存器 1（TIM15_CCMR1）

地址偏移：0x18

复位值：0x0000

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

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	

	IC2F[3:0]				IC2PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	—	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:15	NA	保留位，未定义
14:12	OC2M[2:0]	输出比较2模式
11	OC2PE	输出比较2预装载使能
10	OC2FE	输出比较2快速使能
9:8	CC2S	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7	NA	保留位，未定义
6:4	OC1M[2:0]	输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1、OC1N的值。OC1REF是高电平有效，而OC1、OC1N的有效电平取决于CC1P、CC1NP位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平； 在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)， 否则为有效电平(OC1REF=1)。 111：PWM模式2 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平； 在向下计数时，一旦TIMx_CNT>TIM1_CCRx时通道1为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且

		CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。 注3：在有互补输出的通道上，这些位是预装载的。如果TIMx_CR2寄存器的CCPC=1，OC1M 位只有在COM事件发生时，才从预装载位取新值。
3	OC1PE	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，并且新写入的数值立即起作用。 1：开启TIMx_CCR1寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1的预装载值在更新事件到来时被加载至当前寄存器中。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：为了操作正确，在PWM模式下必须使能预装载功能。但在单脉冲模式下(TIMx_CR1寄存器的OPM=1)，它不是必须的。
2	OC1FE	输出比较1 快速使能 该位用于加快CC输出对触发输入事件的响应。 0：根据计数器与CCR1的值，CC1正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活CC1输出的最小延时为5个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC被设置为比较电平而与比较结果无关。采样触发器的有效沿和CC1输出间的延时被缩短为3个时钟周期。 注意：OC1FE只在通道被配置成PWM1或PWM2模式时起作用。
1:0	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC2F[3:0]	输入捕获2滤波器
11:10	IC2PSC[1:0]	输入/捕获2预分频器
9:8	CC2S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7:4	IC1F[3:0]	输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，以$f_{SAMPLING} = f_{CK_INT}$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]	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]	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TIMx_SMCR寄存器的TS位选择)。 注：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

20.5.8. TIM15 捕获/比较使能寄存器（TIM15_CCER）

地址偏移：0x20

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	CC2NP	—	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
类型	RW	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	CC2NP	输入捕获/比较2互补输出极性。参考CC1NP的描述。
6	NA	保留位，未定义
5	CC2P	输入捕获/比较2输出极性。参考CC1P的描述。
4	CC2E	输入捕获/比较2输出使能。参考CC1E的描述。
3	CC1NP	输入捕获/比较1互补输出极性 CC1通道配置为输出： 0：OC1N高电平有效； 1：OC1N低电平有效。 CC1通道配置为输入： 本为用于和CC1P联合定义TI1FP1和TI2FP1的极性。参考CC1P的描述。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3且CC1S=00(通道配置为输出) 则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1NP位才从预装载位中取新值。
2	CC1NE	输入捕获/比较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和CC1E位的值。 1：开启 - OC1N信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1E位的值。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1NE位才从预装载位中取新值。
1	CC1P	输入捕获/比较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位联合选择在触发或捕获模式下TI1FP1和TI2FP1的有效极性。

		00：非反相/上升沿 电路作用于TIxFP1的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1非反相（在门控模式或编码模式）。 01：反相/下降沿 电路作用于TIxFP1的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 10：保留不用 11：非反相/上升或下降沿 电路作用于TIxFP1的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3，则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1P位才从预装载位中取新值。
0	CC1E	输入捕获/比较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和CC1NE位的值。 1：开启 - OC1信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1NE位的值 。 CC1通道配置为输入： 该位决定了计数器的值是否能捕获入TIMx_CCR1寄存器。 0：捕获禁止； 0：捕获使能。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1E位才从预装载位中取新值。

表 20.1 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态（1）	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	0	1	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性 OCx=OCxREF ^ CCxNP OCx_EN=1	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
		1	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
		1	0	1	关闭状态(运行模式下输出使能) OCx=CCxP, OCx_EN=1	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		1	1	0	OCxREF + 极性 OCx=OCxREF ^ T1CCxNP OCx_EN=1	关闭状态(运行模式下输出使能) OCxN=T1CCxNP, OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
0	0	X	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	0		0	1	输出关闭(不由定时器驱动)	
	0		1	0	异步：OCx=CCxP,OCx_EN=0,OCxN=CCxNP,OCxN_EN=0	
	0		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	1		0	1	关闭状态(空闲模式下输出使能)	
	1		1	0	异步：OCx=CCxP,OCx_EN=1,OCxN=CCxNP,OCxN_EN=1	
	1		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	

（1） 如果一个通道的 2 个输出都没有使用（CCxE=CCxNE=0），那么 OISx，OISxN，CCxP 和 CCxNP 都必须清零。

注意：引脚连接到互补的 OCx 和 OCxN 通道的外部 I/O 引脚的状态，取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器。

20.5.9. TIM15 计数器（TIM15_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

20.5.10. TIM15 预分频器（TIM15_PSC）

地址偏移：0x28

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

20.5.11. TIM15 自动重装载寄存器（TIM15_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 当自动装载值为空时，计数器不工作。

20.5.12. TIM15 重复计数寄存器（TIM15_RCR）

地址偏移：0x30

复位值：0x0000

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	REP[7:0]	重复计数器的值 预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率（即周期性的从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器REP_CNT到达0时，会产生一个更新事件并且计数器REP_CNT重新从REP值开始计数。由于REP_CNT只有在周期更新事件U_RC发生时才重载REP值，因此对TIMx_RCR寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在PWM模式中，（REP+1）对应着在边沿对齐模式下，PWM周期的数目。

20.5.13. TIM15 捕获/比较寄存器（TIM15_CCR1）

地址偏移：0x34

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR1[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR1[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR1[15:0]	捕获/比较通道1的值 若CC1通道配置为输出： CCR1决定了装入当前捕获/比较1寄存器的值（预装载值）。 如果在TIMx_CCMR1寄存器（OC1PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC1端口上输出信号。 若CC1通道配置为输出入： CCR1包含由上一次输入捕获1事件（IC1）传输的计数器值。

20.5.14. TIM15 捕获/比较寄存器 2（TIM15_CCR2）

地址偏移：0x38

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR2[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR2[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义

15:0	CCR2[15:0]	捕获/比较通道1的值 若CC2通道配置为输出： CCR2决定了装入当前捕获/比较2寄存器的值（预装载值）。 如果在TIMx_CCMR2寄存器（OC2PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较2寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC2端口上输出信号。 若CC2通道配置为输出入： CCR2包含由上一次输入捕获2事件（IC2）传输的计数器值。

20.5.15. TIM15 刹车和死区寄存器（TIM15_BDTR）

地址偏移：0x44

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	DTG[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	MOE	主输出使能 一旦刹车输入有效，该位被硬件异步清0。根据AOE位的设置值，该位可以由软件置1或被自动置1。它仅对配置为输出的通道有效。 0：禁止OCx和OCxN输出或强制为空闲状态； 1：如果设置了相应的使能位(TIM1_CCERx寄存器的CCxE位)，则使能OCx和OCxN输出。
14	AOE	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被自动置1(如果刹车输入无效)。 注意：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	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。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用
12	BKE	刹车功能使能

		0：禁止刹车输入(BRK和内部刹车源)； 1：开启刹车输入(BRK和内部刹车源)。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。
11	OSSR	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，首先开启OCx/OCxN并输出无效电平，然后置OCx/OCxN使能输出信号=1。 注意：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	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，OCx/OCxN首先输出其空闲电平，然后OCx/OCxN使能输出信号=1。 注意：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	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位和TIMx_CR2寄存器的OISx/OISxN位； 10：锁定级别2，不能写入锁定级别1中的各位，也不能写入CC极性位(一旦相关通道通过CCxS位设为输出，CC极性位是TIMx_CCER寄存器的CCxP/CCxNP)以及OSSR/OSSI位； 11：锁定级别3，不能写入锁定级别2中的各位，也不能写入CC控制位(一旦相关通道通过CCxS位设为输出，CC控制位是TIMx_CCMR寄存器的OCxM/OCxPE位)； 注意：在系统复位后，只能写一次LOCK位，一旦写入TIMx_BDTR寄存器，则其内容保持不变直至复位。
7:0	DTG[7:0]	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}，其中：$t_{dtg}=t_{DTS}$ DTG[7:5]=10x => DT=(64+DTG[5:0]) * t_{dtg}，其中：$t_{dtg}= 2*t_{DTS}$ DTG[7:5]=110 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg=8* t_{DTS} DTG[7:5]=111 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg= 16*t_{DTS} 举列: 如果t_{DTS} =125 ns (8 MHz)，可能的死区时间为： DTG[7:0] = 0到7Fh，0到15875 ns ，步长时间为125 ns。 DTG[7:0] = 80h到BFh，16µs到31750ns，步长时间为250 ns。 DTG[7:0] = C0h到DFh，32µs到63µs，步长时间为1µs。 DTG[7:0] = E0h到FFh，64µs到126µs，步长时间为2 µs。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1、2或3，则不能修改这些位。

20.5.16. TIM15 DMA 控制寄存器（TIM15_DCR）

地址偏移：0x48

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			DBL[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
7:0	—			DBA[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:13	NA	保留位，未定义
12:8	DBL[4:0]	DMA突发传输长度 这5位定义了DMA在突发传输模式下的传输长度。（当对TIMx_DMAR寄存器进行读或写时，定时器则进行一次突发传输） 00000：1次传输 00001：2次传输 00010：3次传输 … 10001：18次传输
7:5	NA	保留位，未定义
4:0	DBA[4:0]	DMA基地址 这5位定义了DMA传输的基地址（当对TIMx_DMAR寄存器进行读或写时），DBA定义为TIMx_CR1寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR … 例：要完成如下的传输：DBL=7，DBA=TIMx_CR1 此时传输从TIMx_CR1的地址开始向连续7个寄存器进行操作。

20.5.17. TIM15 全部传输时 DMA 地址（TIM15_DMAR）

地址偏移：0x4C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	DMAB[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	DMAB[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DMAB[15:0]	DMA突发传输寄存器 对TIMx_DMAR寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1地址）+（DBA + DMA索引）*4 其中： TIMx_CR1地址是控制器1（TIMx_CR1）所在的地址； DBA是TIMx_DCR寄存器中定义的基地址； DMA索引是由DMA自动控制的偏移量取决于TIMx_DCR寄存器中的DBL。

如何使用 DMA 突发传输的例子

本例中使用定时器 DMA 的突发传输功能，将 CCRx 寄存器（x=2，3，4）的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。

按如下步骤进行操作：

1. 配置相关的 DMA 通道：
   - DMA 通道设备地址为 DMAR 寄存器地址
   - DMA 通道存储器地址为包含要通过 DMA 传送到 CCRx 寄存器的数据 RAM 缓冲区地址
   - 传送数据数量=3
   - 循环模式禁止
2. 配置 DCR 寄存器中的 DBA 和 DBL 位：DBL=3 说明连续传输次数为 3 次；DBA=0xE，初始传输的偏移地址为 0x38（TIMx_CCR2）。
3. 使能 TIMx 更新 DMA 请求
4. 使能 TIMx
5. 使能 DMA 通道

注意：在本例中所有 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传输的数据数量应该为 6，而 RAM 缓冲区要包含 data1，data2，data3，data4，data5 和 data6。数据按如下过程被传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个更新 DMA 请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

20.6. TIM16 和 TIM17 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIMx_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CKD[1:0]		ARPE	-	-	-	OPM	URS	UDIS	CEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	0	0	0	0
0x04	TIMx_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OIS1N	OIS1	-	-	-	-	CCDS	-	-	CCPC
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	x	x	x	x	0	x	x	0
0x0C	TIMx_DIER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC1DE	UDE	BIE	-	COMIE	-	-	-	CC1IE	UIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	0	x	x	x	0	0
0x10	TIMx_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC1OF	-	BIF	-	COMIF	-	-	-	CC1IF	UIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	x	0	x	x	x	0	0
0x14	TIMx_EGR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BG	-	COMG	-	-	-	CC1G	UG
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	x	x	x	0	0
0x18	TIMx_CCMR1（输出模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0
	TIMx_CCMR1（输入模式）	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x20	TIMx_CCER	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CC1NP	CC1NE	CC1P	CC1E
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0
0x24	TIMx_CNT	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CNT[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIMx_PSC	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PSC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIMx_ARR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ARR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x30	TIMx_RCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	REP[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x34	TIMx_CCR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CCR1[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TIMx_BDTR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x48	TIMx_DCR																			DBL[4:0]								DBA[4:0]				
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	x	0	0	0	0	0
0x4C	TIMx_DMAR																DMAB[15:0]														
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

20.6.1. TIM16 和 TIM17 控制寄存器 1(TIM16_CR1 和 TIM17_CR1)

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CKD[1:0]	
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	ARPE	—			OPM	URS	UDIS	CEN
类型	RW	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9:8	CKD[1:0]	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，不要使用此配置
7	ARPE	自动重装载预装载允许位 0：TIMx_ARR寄存器没有缓冲，它可以被直接写入 1：TIMx_ARR 寄存器由预装载缓冲器缓冲
6:5	NA	保留位，未定义
4	DIR	计数方向 0：计数器向上计数； 1：计数器向下计数。 注意：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	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	更新请求源 软件通过该位选择UEV事件的源 0：如果UDIS允许产生更新事件，则下述任一事件产生一个更新中断或DMA

		请求： – 计数器上溢 – 软件设置UG位 – 复位触发事件产生的更新 1：如果使能了更新中断或DMA请求，则只有计数器上溢/下溢时才能产生更新中断或DMA请求。
1	UDIS	禁止更新 软件通过该位允许/禁止UEV事件的产生 0：允许UEV事件。更新事件UEV由下述任一事件产生： – 计数器溢出 – 软件设置UG位 – 复位触发事件产生的更新 1：禁止UEV事件。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果触发复位模式下触发事件到来时或软件设置UG位，计数器和预分频器会被重新初始化。
0	CEN	允许计数器 0：禁止计数器； 1：使能计数器。 注意：在软件设置了CEN位后，外部时钟和门控模式才能工作。而触发模式下可以自动地通过硬件设置CEN位。

20.6.2. TIM16 和 TIM17 控制器 2（TIM16_CR2 和 TIM17_CR2）

地址偏移：0x04

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						OIS1N	OIS1
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	—				CCDS	—		CCPC
类型	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RO-0	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9	OIS1N	输出空闲状态1(OC1N输出)。 0：当MOE=0时，则在一个死区时间后，OC1N=0； 1：当MOE=0时，则在一个死区时间后，OC1N=1。 注意：已经设置了LOCK(TIM1_BKR寄存器)级别1、2或3后，该位不能被修改。
8	OIS1	输出空闲状态1(OC1输出)。 0：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=0； 1：当MOE=0时，如果OC1N使能，则在一个死区后，OC1=1。 注意：已经设置了LOCK(TIM1_BKR寄存器)级别1、2或3后，该位不能被修改。

7:4	NA	保留位，未定义
3	CCDS	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	NA	保留位，未定义
0	CCPC	捕获/比较预装载控制位 0：CCxE，CCxNE，CCxP，CCxNP和OCxM位不是预装载的； 1：CCxE，CCxNE，CCxP，CCxNP和OCxM位是预装载的；设置该位后，只在设置了COMG位或TRGI检测到上升沿（取决于CCUS位的设置）后被更新。 注意：该位只对具有互补输出的通道起作用。

20.6.3. TIM16 和 TIM17 DMA/中断使能寄存器（TIM16_DIER 和 TIM17_DIER）

地址偏移：0x0C

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CC1DE	UDE
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:0	BIE	—	COMIE	—			CC1IE	UIE
类型	RW	RO-0	RW	RO-0	RO-0	RO-0	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9	CC1DE	捕获/比较 1 DMA 请求使能 0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
8	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允许
7	BIE	刹车中断使能 0：刹车中断禁止 1：刹车中断允许
6	NA	保留位，未定义
5	COMIE	COM 换相事件中断使能 0：COM 中断禁止 1：COM 中断允许

4:2	NA	保留位，未定义
1	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
0	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

20.6.4. TIM16 和 TIM17 状态寄存器（TIM16_SR 和 TIM17_SR）

地址偏移：0x10

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						CC1OF	—
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0	RO-0
7:0	BIF	—	COMIF	—			CC1IF	UIF
类型	RC_W0	RO-0	RC_W0	RO-0	RO-0	RO-0	RC_W0	RC_W0

Bit	Name	Function
31:10	NA	保留位，未定义
9	CC1OF	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，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIMx_CCR1寄存器时，CC1IF的状态已经为1。
8	NA	保留位，未定义
7	BIF	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	NA	保留位，未定义
5	COMIF	COM中断标志 一旦产生COM事件(当捕获/比较控制位：CCxE、CCxNE、OCxM已被更新)该位由硬件置1。它由软件清0 。 0：无COM事件产生； 1：COM中断等待响应。
4:2	NA	保留位，未定义
1	CC1IF	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，但在中心对称模式下除外(参考TIM1_CR1寄存器的CMS位)。它由软件清0。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_ARR的内容时，在向上或向上/向下计数模式时计数器溢出，或向下计数模式时计数器下溢条件下，CC1IF位变高。 如果通道CC1配置为输入模式： 当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIMx_CCR1清0。 0：无输入捕获产生； 1：计数器值已被捕获至TIMx_CCR1(在IC1上检测到与所选极性相同的边沿)。
0	UIF	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，当计数器上溢时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当设置TIMx_EGR寄存器的UG位软件对计数器重新初始化时； – 若TIMx_CR1寄存器的UDIS=0、URS=0，当计数器CNT被触发事件重新初始化时 (参考从模式控制寄存器TIMx_SMCR)。

20.6.5. TIM16 和 TIM17 事件产生寄存器（TIM16_EGR 和 TIM17_EGR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	BG	—	COMG	—			CC1G	UG
类型	W	RO-0	W	RO-0	RO-0	RO-0	W	W

Bit	Name	Function
31:8	NA	保留位，未定义
7	BG	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	NA	保留位，未定义
5	COMG	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、CCxP、CCxNP、OCIM位。 注意：该位只对拥有互补输出的通道有效。
4:2	NA	保留位，未定义
1	CC1G	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： 设置CC1IF=1，若开启对应的中断和DMA，则产生相应的中断和DMA。 若通道CC1配置为输入： 当前的计数器值被捕获至TIMx_CCR1寄存器，设置CC1IF=1，若开启对应的中断和DMA ,则产生相应的中断和DMA。若CC1IF已经为1 ,则设置CC1OF=1。
0	UG	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)。若在中心对称模式下或DIR=0(向上计数)则计数器被清0；若DIR=1(向下计数)则计数器取TIMx_ARR的值。

20.6.6. TIM16 和 TIM17 捕获/比较模式寄存器 1（TIM16_CCMR1 和 TIM17_CCMR1）

地址偏移：0x18

复位值：0x0000

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

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	Function
31:7	NA	保留位，未定义
6:4	OC1M[2:0]	输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1、OC1N的值。OC1REF是高电平有效，而OC1、OC1N的有效电平取决于CC1P、CC1NP位。 000：冻结。输出比较寄存器TIMx_CCR1与计数器TIMx_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIMx_CNT的值与捕获/比较寄存器1(TIMx_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIMx_CCR1=TIMx_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为有效电平，否则为无效电平； 在向下计数时，一旦TIMx_CNT>TIMx_CCR1时通道1为无效电平(OC1REF=0)， 否则为有效电平(OC1REF=1)。 111：PWM模式2 - 在向上计数时，一旦TIMx_CNT<TIMx_CCR1时通道1为无效电平，否则为有效电平； 在向下计数时，一旦TIMx_CNT>TIM1_CCRx时通道1为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。 注3：在有互补输出的通道上，这些位是预装载的。如果TIMx_CR2寄存器的CCPC=1，OC1M 位只有在COM事件发生时，才从预装载位取新值。
3	OC1PE	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，并且新写入的数值立即起作用。 1：开启TIMx_CCR1寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1的预装载值在更新事件到来时被加载至当前寄存器中。 注1：一旦LOCK级别设为3(TIMx_BDTR寄存器中的LOCK位)并且CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：为了操作正确，在PWM模式下必须使能预装载功能。但在单脉冲模式下(TIMx_CR1寄存器的OPM=1)，它不是必须的。
2	OC1FE	输出比较1 快速使能 该位用于加快CC输出对触发输入事件的响应。 0：根据计数器与CCR1的值，CC1正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活CC1输出的最小延时为5个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC被设置为比较电平而与比较结果无关。采样触发器的有效沿和CC1输出间的延时被缩短为3个时钟周期。 注意：OC1FE只在通道被配置成PWM1或PWM2模式时起作用。

1:0	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。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

输入捕获模式：

Bit	Name	Function
31:16	NA	保留位，未定义
15:12	IC2F[3:0]	输入捕获2滤波器
11:10	IC2PSC[1:0]	输入/捕获2预分频器
9:8	CC2S[1:0]	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。此模式仅工作在内部触发器输入被选中时(由TIMx_SMCR寄存器的TS位选择)。 注：CC2S仅在通道关闭时(TIMx_CCER寄存器的CC2E=0，CC2NE=0且已被更新)才是可写的。
7:4	IC1F[3:0]	输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，以$f_{SAMPLING} = f_{CK_INT}$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]	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]	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。 注意：CC1S仅在通道关闭时(TIMx_CCER寄存器的CC1E=0)才是可写的。

20.6.7. TIM16 和 TIM17 捕获/比较使能寄存器（TIM16_CCER 和 TIM17_CCER）

地址偏移：0x20

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—				CC1NP	CC1NE	CC1P	CC1E
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:4	NA	保留位，未定义
3	CC1NP	输入捕获/比较1互补输出极性 CC1通道配置为输出： 0：OC1N高电平有效； 1：OC1N低电平有效。 CC1通道配置为输入： 本为用于和CC1P联合定义TI1FP1和TI2FP1的极性。参考CC1P的描述。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3且CC1S=00(通道配置为输出) 则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1NP位才从预装载位中取新值。
2	CC1NE	输入捕获/比较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和CC1E位的值。 1：开启 - OC1N信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1E位的值。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1NE位才从预装载位中取新值。

1	CC1P	输入捕获/比较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位联合选择在触发或捕获模式下TI1FP1和TI2FP1的有效极性。 00：非反相/上升沿 电路作用于TIxFP1的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1非反相（在门控模式或编码模式）。 01：反相/下降沿 电路作用于TIxFP1的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 10：保留不用 11：非反相/上升或下降沿 电路作用于TIxFP1的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作），TIxFP1反相（在门控模式或编码模式）。 注1：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2或3，则该位不能被修改。 注2：对于有互补输出的通道，该位是预装载的。如果CCPC=1（TIMx_CR2寄存器），只有在COM事件发生时，CC1P位才从预装载位中取新值。
0	CC1E	输入捕获/比较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和CC1NE位的值。 1：开启 - OC1信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1NE位的值 。 CC1通道配置为输入： 该位决定了计数器的值是否能捕获入TIMx_CCR1寄存器。 0：捕获禁止； 0：捕获使能。 注：对于有互补输出的通道，该位是预装载的。如果CCPC=1(TIMx_CR2寄存器)，只有在COM事件发生时，CC1E位才从预装载位中取新值。

表 20.2 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态（1）	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	0	1	输出关闭(不由定时器驱动) OCx=0, OCx_EN=0	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性 OCx=OCxREF ^ CCxNP OCx_EN=1	输出关闭(不由定时器驱动) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
		1	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
		1	0	1	关闭状态(运行模式下输出使能) OCx=CCxP, OCx_EN=1	OCxREF + 极性 OCxN=OCxREF ^ CCxNP OCxN_EN=1
		1	1	0	OCxREF + 极性 OCx=OCxREF ^ T1CCxNP OCx_EN=1	关闭状态(运行模式下输出使能) OCxN=T1CCxNP, OCx_EN=1
		1	1	1	OCxREF + 极性 + 死区 OCx_EN=1	OCxREF 的互补信号 + 极性 + 死区 OCxN_EN=1
0	0	X	0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	0		0	1	输出关闭(不由定时器驱动)	
	0		1	0	异步：OCx=CCxP,OCx_EN=0,OCxN=CCxNP,OCxN_EN=0	
	0		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	
	1		0	0	输出关闭(不由定时器驱动) OCx=CCxP, OCx_EN=0	输出关闭(不由定时器驱动) OCxN=CCxNP,OCxN_EN=0
	1		0	1	关闭状态(空闲模式下输出使能)	
	1		1	0	异步：OCx=CCxP,OCx_EN=1,OCxN=CCxNP,OCxN_EN=1	
	1		1	1	若时钟存在：在死区时间之后 OCx=OISx,OCxN=OISxN，假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	

（1） 如果一个通道的 2 个输出都没有使用（CCxE=CCxNE=0），那么 OISx，OISxN，CCxP 和 CCxNP 都必须清零。

注意：引脚连接到互补的 OCx 和 OCxN 通道的外部 I/O 引脚的状态，取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器。

20.6.8. TIM16 和 TIM17 计数器（TIM16_CNT 和 TIM17_CNT）

地址偏移：0x24

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CNT[15:0]	计数器值

20.6.9. TIM16 和 TIM17 预分频器（TIM16_PSC 和 TIM17_PSC）

地址偏移：0x28

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	PSC[15:0]	预分频值 计数器的时钟频率（CK_CNT）等于f_{CK_PSC}/(PSC[15:0]+1)。 每次当更新事件产生时，PSC的值被装入当前预分频器寄存器

20.6.10. TIM16 和 TIM17 自动重装载寄存器（TIM16_ARR 和 TIM17_ARR）

地址偏移：0x2C

复位值：0xFFFF

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	ARR[15:0]	自动重装载值 ARR包含了将要装载如实际的自动重装载寄存器的值。 当自动装载值为空时，计数器不工作。

20.6.11. TIM16 和 TIM17 重复计数寄存器（TIM16_RCR 和 TIM17_RCR）

地址偏移：0x30

复位值：0x0000

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	REP[7:0]	重复计数器的值 预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率（即周期

		性的从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器REP_CNT到达0时，会产生一个更新事件并且计数器REP_CNT重新从REP值开始计数。由于REP_CNT只有在周期更新事件U_RC发生时才重载REP值，因此对TIMx_RCR寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在PWM模式中，（REP+1）对应着在边沿对齐模式下，PWM周期的数目。

20.6.12. TIM16 和 TIM17 捕获/比较寄存器（TIM16_CCR1 和 TIM17_CCR1）

地址偏移：0x34

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CCR1[15:8]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	CCR1[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CCR1[15:0]	捕获/比较通道1的值 若CC1通道配置为输出： CCR1决定了装入当前捕获/比较1寄存器的值（预装载值）。 如果在TIMx_CCMR1寄存器（OC1PE位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。当前捕获/比较寄存器参与计数器TIMx_CNT的比较，并在OC1端口上输出信号。 若CC1通道配置为输出入： CCR1包含由上一次输入捕获1事件（IC1）传输的计数器值。

20.6.13. TIM16 和 TIM17 刹车和死区寄存器（TIM16_BDTR 和 TIM17_BDTR）

地址偏移：0x44

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	DTG[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	MOE	主输出使能 一旦刹车输入有效，该位被硬件异步清0。根据AOE位的设置值，该位可以由软件置1或被自动置1。它仅对配置为输出的通道有效。 0：禁止OCx和OCxN输出或强制为空闲状态； 1：如果设置了相应的使能位(TIM1_CCERx寄存器的CCxE位)，则使能OCx和OCxN输出。
14	AOE	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被自动置1(如果刹车输入无效)。 注意：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	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。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用
12	BKE	刹车功能使能 0：禁止刹车输入(BRK和内部刹车源)； 1：开启刹车输入(BRK和内部刹车源)。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1，则该位不能被修改。 注意：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。
11	OSSR	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)；

		1：当定时器不工作时，一旦CCxE=1或CCxNE=1，首先开启OCx/OCxN并输出无效电平，然后置OCx/OCxN使能输出信号=1。 注意：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	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。 0：当定时器不工作时，禁止OCx/OCxN输出(OCx/OCxN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，OCx/OCxN首先输出其空闲电平，然后OCx/OCxN使能输出信号=1。 注意：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	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位和TIMx_CR2寄存器的OISx/OISxN位； 10：锁定级别2，不能写入锁定级别1中的各位，也不能写入CC极性位(一旦相关通道通过CCxS位设为输出，CC极性位是TIMx_CCER寄存器的CCxP/CCxNP)以及OSSR/OSSI位； 11：锁定级别3，不能写入锁定级别2中的各位，也不能写入CC控制位(一旦相关通道通过CCxS位设为输出，CC控制位是TIMx_CCMR寄存器的OCxM/OCxPE位)； 注意：在系统复位后，只能写一次LOCK位，一旦写入TIMx_BDTR寄存器，则其内容保持不变直至复位。
7:0	DTG[7:0]	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}，其中：$t_{dtg}=t_{DTS}$ DTG[7:5]=10x => DT=(64+DTG[5:0]) * t_{dtg}，其中：$t_{dtg}= 2*t_{DTS}$ DTG[7:5]=110 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg=8* t_{DTS} DTG[7:5]=111 => DT=(32+DTG[4:0]) * t_{dtg}，其中：tdtg= 16*t_{DTS} 举列: 如果t_{DTS} =125 ns (8 MHz)，可能的死区时间为： DTG[7:0] = 0到7Fh，0到15875 ns ，步长时间为125 ns。 DTG[7:0] = 80h到BFh，16µs到31750ns，步长时间为250 ns。 DTG[7:0] = C0h到DFh，32µs到63µs，步长时间为1µs。 DTG[7:0] = E0h到FFh，64µs到126µs，步长时间为2 µs。 注意：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为1、2或3，则不能修改这些位。

20.6.14. TIM16 和 TIM17 DMA 控制寄存器（TIM16_DCR 和 TIM17_DCR）

地址偏移：0x48

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	保留位			DBL[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
7:0	保留位			DBA[4:0]				
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:13	NA	保留位，未定义
12:8	DBL[4:0]	DMA突发传输长度 这5位定义了DMA在突发传输模式下的传输长度。（当对TIMx_DMAR寄存器进行读或写时，定时器则进行一次突发传输） 00000：1次传输 00001：2次传输 00010：3次传输 … 10001：18次传输
7:5	NA	保留位，未定义
4:0	DBA[4:0]	DMA基地址 这5位定义了DMA传输的基地址（当对TIMx_DMAR寄存器进行读或写时），DBA定义为TIMx_CR1寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR … 例：要完成如下的传输：DBL=7，DBA=TIMx_CR1 此时传输从TIMx_CR1的地址开始向连续7个寄存器进行操作。

20.6.15. TIM16 和 TIM17 全部传输时 DMA 地址（TIM16_DMAR 和 TIM17_DMAR）

地址偏移：0x4C

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DMAB[15:0]	DMA突发传输寄存器 对TIMx_DMAR寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1地址）+（DBA + DMA索引）*4 其中： TIMx_CR1地址是控制器1（TIMx_CR1）所在的地址； DBA是TIMx_DCR寄存器中定义的基地址； DMA索引是由DMA自动控制的偏移量取决于TIMx_DCR寄存器中的DBL。

如何使用 DMA 突发传输的例子

本例中使用定时器 DMA 的突发传输功能，将 CCRx 寄存器（x=2，3，4）的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。

按如下步骤进行操作：

1. 配置相关的 DMA 通道：
   - DMA 通道设备地址为 DMAR 寄存器地址
   - DMA 通道存储器地址为包含要通过 DMA 传送到 CCRx 寄存器的数据 RAM 缓冲区地址
   - 传送数据数量=3
   - 循环模式禁止
2. 配置 DCR 寄存器中的 DBA 和 DBL 位：DBL=3 说明连续传输次数为 3 次；DBA=0xE，初始传输的偏移地址为 0x38（TIMx_CCR2）。
3. 使能 TIMx 更新 DMA 请求
4. 使能 TIMx
5. 使能 DMA 通道

注意：在本例中所有 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传输的数据数量应该为 6，而 RAM 缓冲区要包含 data1，data2，data3，data4，data5 和 data6。数据按如下过程被传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个更新 DMA 请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

21. 红外接口（IRTIM）

设备可通过红外接口进行远程操控。红外接口可实现红外 LED 的远程操控。
它利用 USART1、USART2、TIM16 和 TIM17 这 4 条内部连接线实现红外接口功能，如下图 18-1 所示。
为了产生红外遥控信号，红外遥控界面必须使能，并且 TIM16 和 TIM17 必须被正确配置以产生正确的波形。
红外接收器的实现可以通过简单的输入捕捉模式得以实现。

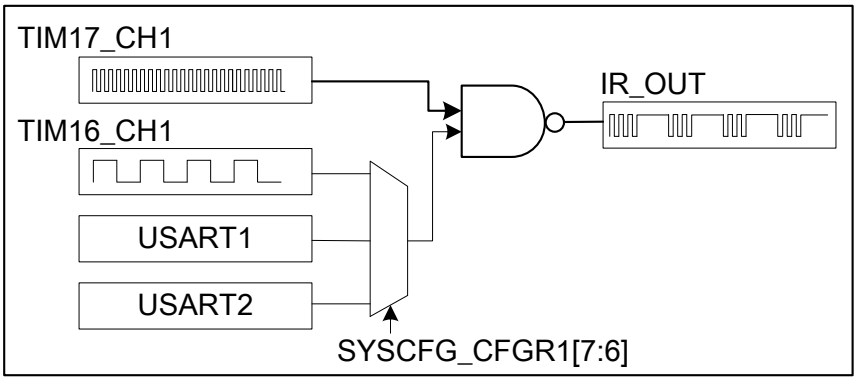


图 21.1 红外接口内部硬件信号连接图

所有标准红外接口脉冲调制模式都可通过配置两个定时器输出比较通道获得。
TIM17 产生高频载波信号，TIM16 则产生调制信号。
红外功能输出通过 IR_OUT 管脚。打开此输出功能可通过配置 GPIOx_AFRx 寄存器来使能相关的复用功能位。

22. 独立看门狗（IWDG）

22.1. 简介

本器件集成了一个内嵌的看门狗外设，用集成的方式提供了高安全水平，精准的时间控制及运用的灵活性。集成的看门狗外设是用来解决某些软件故障问题的；当它的定时计数值达到预设门限的时候，它会触发一个系统复位。

22.2. IWDG 主要功能

- 向下递减计数器
- 由一个独立的 RC 振荡器驱动（在 Standby 和 Stop 模式下仍可操作）
- 复位条件
  - 当向下递减计数器的值达到 0 时，产生复位请求
  - 当配置了窗口选项的情况下，向下递减计数器的值处于窗口之上时，进行计数器重新加载的操作就会产生复位请求

22.3. IWDG 功能描述

22.3.1. IWDG 模块框图

图 19-1 描述了独立看门狗模块的功能框图。

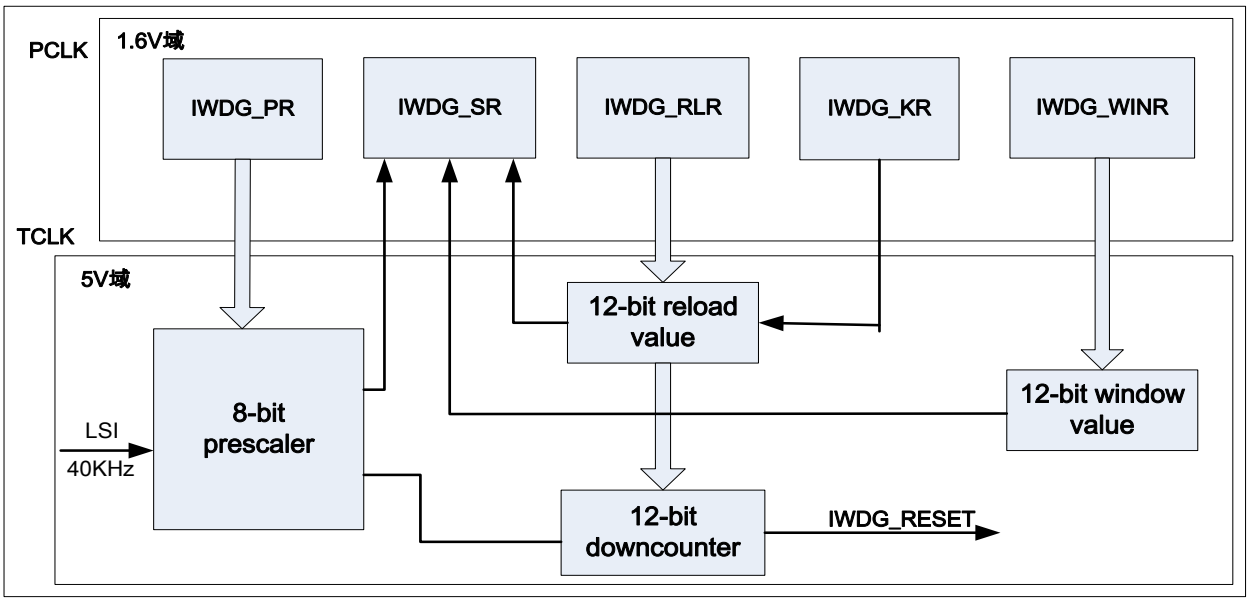


图 22.1 独立看门狗功能框图

当向独立看门狗的关键寄存器写入启动值 0x0000CCCC 的时候，计数器开始由复位值 0xFFF 向下计数。当计数值达到 0x000 的时候由独立看门狗发出复位信号（IWDG_RESET）。

任何时候将关键字 0x0000AAAA 写到关键寄存器（IWDG_KR）中，都会使得重加载寄存器（IWDG_RLR）中的值被重新加载到计数器中，从而阻止即将发生的复位动作。但两次写关键字 0x0000AAAA 之间必须要有一定的间隔时间，至少为 3 个 40kHz 的慢时钟周期。

22.3.2. 窗口选项

通过对窗口寄存器（IWDG_WINR）设置适当的值即可使独立看门狗工作在窗口看门狗模式下。

如果重加载操作执行时，计数器的值超出了窗口寄存器（IWDG_WINR）中的存储值，也会产生复位。

窗口寄存器(IWDG_WINR)的默认值是 0x00000FFF,所以如果没有对窗口寄存器(IWDG_WINR)进行改写，那么窗口选项默认是关闭的。

窗口值一旦改变，立即就会引起计数器的一次重加载动作，将其置为重加载寄存器（IWDG_RLR）中所设置的值，从而一定程度上延缓目前到下次复位所需的时间周期。

当窗口选项使能时配置 IWDG

1. 将 0x0000CCCC 写到关键寄存器（IWDG_KR），使能 IWDG。
2. 向关键寄存器（IWDG_KR）中写入 0x00005555 打开寄存器访问许可。
3. 向预分频寄存器（IWDG_PR）写入 0~7 的值，以配置 IWDG 的预分频器。
4. 配置重加载寄存器（IWDG_RLR）。
5. 等待状态寄存器（IWDG_SR）的值更新为 0x00000000。
6. 配置窗口寄存器（IWDG_WINR），这将会自动将重加载寄存器（IWDG_RLR）的值更新到计数器中。

注意：当状态寄存器（IWDG_SR）的值为 0x00000000 时，写窗口寄存器会时计数器更新为重加载寄存器（IWDG_RLR）的值。

当窗口选项被禁止时配置 IWDG

当窗口选项未被使用时，可按下列顺序配置 IWDG：

1. 将 0x0000CCCC 写到关键寄存器（IWDG_KR），使能 IWDG。
2. 向关键寄存器（IWDG_KR）中写入 0x00005555 打开寄存器访问许可。
3. 向预分频寄存器（IWDG_PR）写入 0~7 的值，以配置 IWDG 的预分频器。
4. 配置重加载寄存器（IWDG_RLR）。
5. 等待状态寄存器（IWDG_SR）的值更新为 0x00000000。
6. 向关键寄存器（IWDG_KR）中写入 0x0000AAAA，将重加载寄存器（IWDG_RLR）值载入计数器中。

22.3.3. 硬件看门狗

如果配置 DBG 模块中的 DBG_IWDG_STOP 位为 1，那么在上电的时候独立看门狗就被自动打开。如果没有在计数器计数到 0 之前或者向下计数器的值超出窗口之前向关键寄存器（IWDG_KR）写入 0x0000AAAA，那么会产生硬件复位请求。

22.3.4. Stop 模式和 Standby 模式下的行为

在 Stop 模式和 Standby 模式下 ,IWDG 的寄存器无法进行配置 ,但计数器能正常进行 ,复位信号能正常产生。

22.3.5. 寄存器访问保护

默认条件下，对预分频寄存器（IWDG_PR）、重加载寄存器（IWDG_RLR）和窗口寄存器（IWDG_WINR）的写访问操作都是受保护的。想要改变这一点 ,必须先向关键寄存器(IWDG_KR)中写入解锁码 0x00005555。如果写入别的值，将会使得对寄存器的访问保护重新生效。这意味着在做重加载操作（向关键寄存器写入 0x0000AAAA）的时候就属于这种情况.
可以通过状态寄存器(IWDG_SR)来观察预分频器的更新情况、计数器的重加载情况或窗口值的重加载情况。

22.3.6. 调试模式

当微控制器进入调试模式时（内核被暂停），计数器可以继续运行也可以被停止；这取决于 DBG 模块中的 DBG_IWDG_STOP 选项的配置。

22.4. IWDG 寄存器映射

下表给出了 IWDG 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR																	KEY[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR																														PR[2:0]		
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0
0x08	IWDG_RLR																					RL[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR																														WVU	RVU	PVU
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0
0x10	IWDG_WINR																					WIN[11:0]											
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1

22.4.1. 关键寄存器（IWDG_KR）

地址偏移：0x00

复位值：0x0000 0000（退出 Standby 模式时复位）

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	KEY[15:0]	此寄存器只能写，读数据为 0x0000 这些位必须周期性的由软件写入 0xAAAA ,否则当计数器向下计数到 0 的时候会产生硬件复位请求。 写入 0x5555 会使能对预分频寄存器（IWDG_PR）、重加载寄存器（IWDG_RLR）和窗口寄存器（IWDG_WINR）的访问许可。 写入 0xCCCC 会启动看门狗（除非硬件看门狗选项在上电时就已经使能）

22.4.2. 预分频寄存器（IWDG_PR）

地址偏移：0x04

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					PR		
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
31:3	NA	保留位，未定义
2:0	PR[2:0]	这些位平时处于写保护状态；由软件写入来选择对输入时钟的预分频系数。为了能够改变预分频器的系数，状态寄存器（IWDG_SR）中的 PVU 位必须先清零。 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 注意：读这个寄存器会返回 5V 供电区域的分频器的值。对这个寄存器进行写操作后，寄存器值不一定有那么快更新；所以只有当状态寄存器（IWDG_SR）中的 PVU 位为 0 时，读这个寄存器数据才是有效的。

22.4.3. 重加载寄存器（IWDG_RLR）

地址偏移：0x08

复位值：0x0000 0FFF（退出 Standby 模式时复位）

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

Bit	Name	Function
31:12	NA	保留位，未定义
11:0	RL[11:0]	这些位平时处于写保护状态；由软件写入来设置，并且每次向关键寄存器（IWDG_KR）写入 0xAAAA 的时候，这个值会被更新到计数器中。看门狗计数器正是从这个值开始向下计数；定时的长度由这个值和预分频器的设定值来共同决定。 注意：读这个寄存器会返回 5V 供电区域的分频器的值。对这个寄存器进行写操作后，寄存器值不一定有那么快更新；所以只有当状态寄存器（IWDG_SR）中的 RVU 位为 0 时，读这个寄存器数据才是有效的。

22.4.4. 状态寄存器（IWDG_SR）

地址偏移：0x0C

复位值：0x0000 0FFF（退出 Standby 模式时不会被复位）

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							

类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					WVU	RVU	PVU
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
31:3	NA	保留位，未定义
2	WVU	看门狗计数器窗口值更新标志位。 该位由硬件置位，用来表明正在更新窗口值。当 5V 供电区域中完成了窗口值的加载时，该位会被硬件清零。(这需要 3 个 40kHz 的 RC 振荡周期)。 只有当 WVU 的值为 0 时才能再次改写窗口值。
1	RVU	看门狗计数器重加载值更新标志位。 该位由硬件置位，用来表明正在更新重加载值。当 5V 供电区域中完成了重加载值的加载时，该位会被硬件清零。(这需要 3 个 40kHz 的 RC 振荡周期)。 只有当 RVU 的值为 0 时才能再次改写重加载值。
0	PVU	看门狗预分频器值更新标志位。 该位由硬件置位，用来表明正在更新预分频值。当 5V 供电区域中完成了预分频值的加载时，该位会被硬件清零。(这需要 3 个 40kHz 的 RC 振荡周期)。 只有当 PVU 的值为 0 时才能再次改写预分频值。

22.4.5. 窗口寄存器（IWDG_WINR）

地址偏移：0x0C

复位值：0x0000 0FFF（退出 Standby 模式时复位）

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

Bit	Name	Function
31:12	NA	保留位，未定义
11:0	WIN[11:0]	这些位平时处于写保护状态 ;这些位包含的是和向下计数器比较的上限的窗口值。 为了阻止复位信号的产生 ,必须在向下计数器递减到窗口值和 0 之间的某个值时重加载计数器。

		要想改变这个窗口值，必须先保证状态寄存器（IWDG_SR）中的 WVU 位为 0。 注意：读这个寄存器会返回 5V 供电区域的分频器的值。对这个寄存器进行写操作后，寄存器值不一定有那么快更新；所以只有当状态寄存器（IWDG_SR）中的 WVU 位为 0 时，读这个寄存器数据才是有效的。

23. 系统窗口看门狗（WWDG）

23.1. 简介

窗口看门狗通常被用来监测 ,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。除非递减计数器的值在 T6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个 MCU 复位。在递减计数器达到窗口寄存器数值之前，如果 7 位的递减计数器数值被刷新，那么也将产生一个 MCU 复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。
窗口看门狗时钟从 APB 时钟预分频，并有一个可配置的时间窗口，这可通过编程来检测异常推迟或提前的应用行为。
窗口看门狗最适合要求看门狗在一个精确时间窗口内做出反应的应用程序。

23.2. WWDG 主要功能

- 可编程的自由运行向下递减计数器
- 复位条件
  - 当向下递减计数器的值小于 0x40 时，产生复位请求。
  - 当向下递减计数器在窗口外被重新装载时，产生复位请求。
- 提前唤醒中断（EWI）: 如果启动了看门狗，当向下递减计数器等于 0x40 时产生提前唤醒中断。

23.3. WWDG 功能描述

如果看门狗复位使能被启动(控制寄存器(WWDG_CR)中的 WDGA 位被置 1),并且当 7 位递减计数器(T[6:0])从 0x40 下降到 0x3F（T6 位清零）时，将产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。
应用程序在正常运行过程中必须定期的写入控制寄存器（WWDG_CR）以防止 MCU 发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在控制寄存器（WWDG_CR）中的数值必须在 0xFF 和 0xC0 之间。

图 20-1 描述了窗口看门狗模块的功能框图。

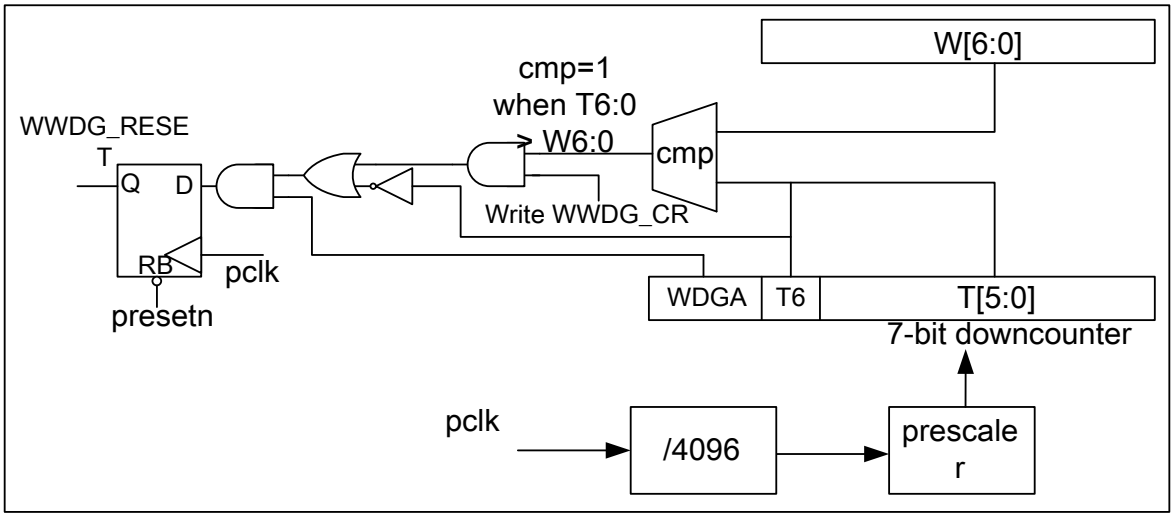


图 23.1 窗口看门狗功能框图

23.3.1. 使能看门狗复位

在系统复位之后，看门狗复位使能总是处于关闭状态，设置控制寄存器（WWDG_CR）的 WDGA 位能够开启看门狗复位使能，随后它不能被关闭，除非发生复位。

23.3.2. 控制递减计数器

递减计数器处于自由运行状态；即使看门狗复位使能被禁止，递减计数器仍继续递减计数。当看门狗复位使能位被启用时，T6 位必须被设置，以防止立即产生一个复位。

T[5:0]位包含了看门狗产生复位之前的计时数目。复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入控制寄存器（WWDG_CR）时，预分频值是未知的。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大于 0x3F 时被重新装载。图 20-2 描述了窗口寄存器的工作过程。

注意：可以用 T6 位产生一个软件复位（设置 WDGA 位为 1，T6 位为 0）。

23.3.3. 看门狗中断高级特性

如果在实际复位产生之前必须进行特定的安全操作或数据记录，可以用提前唤醒中断。设置配置寄存器（WWDG_CFR）中的 EWI 位开启该中断。在复位之前当递减计数器到达 0x40 时，则产生此中断，同时可以用相应的中断服务程序来触发特定的行为（例如通信或数据记录）。

在某些应用中，提前唤醒中断可以用来管理软件系统检测或系统恢复或故障弱化，但不产生窗口看门狗复位（WWDG_RESET）。在这种情况下，相应的中断服务程序将重加载计数器，以避免窗口看门狗复位，然后触发必要的行为。

注意：当提前唤醒中断无法启用时，例如由于系统锁定在更高优先级任务，最终将产生窗口看门狗复位。

23.3.4. 如何设置窗口看门狗超时

可以使用下面提供的公式计算窗口看门狗的超时时间。
注意：当写入控制寄存器（WWDG_CR）时，始终置 T6 位为 1，以避免立即产生一个复位。

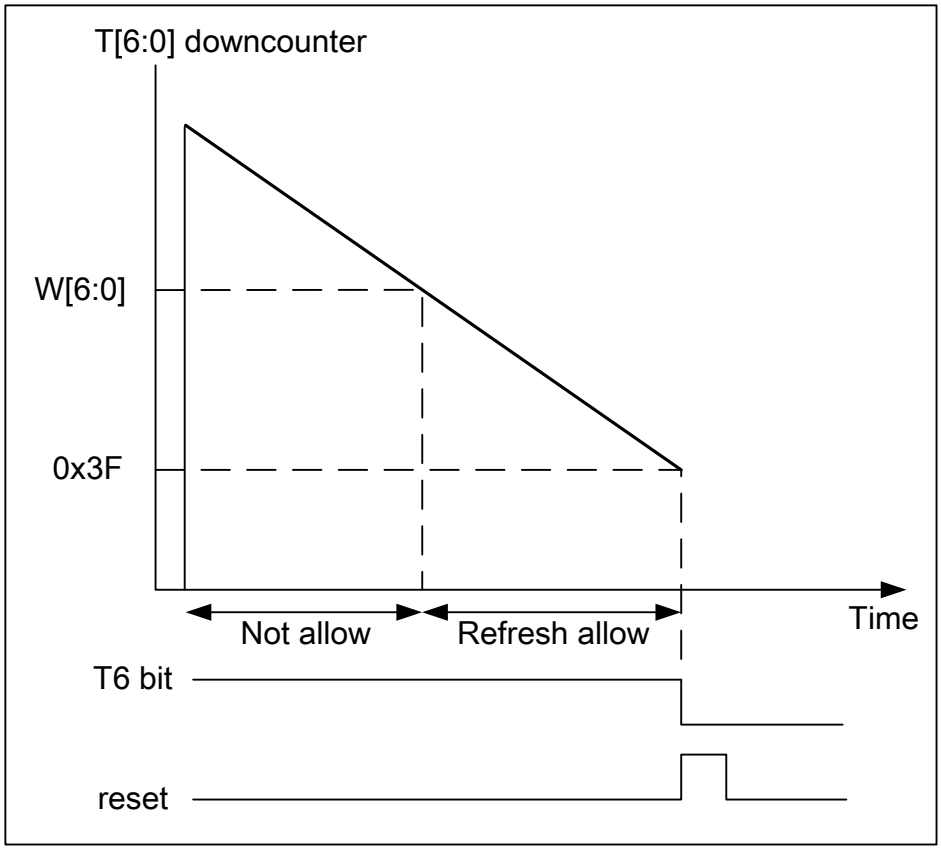


图 23.2 窗口看门狗时序图

计算超时的公式如下：

$$t_{WWDG} = t_{PCLK} * 4096 * 2^{WDGTB[1:0]} * (T[5:0] + 1) \qquad (\text{ms})$$

其中：
t_{WWDG}：WWDG 超时时间
t_{PCLK}：APB 以 ms 为单位的时钟周期
4096：内部 分频器中的固有值
假设 APB 频率为 48MHz，WDGTB[1:0]设置成 3 并且 T[5:0]设为 63，那么超时时间的计算如下：

$$t_{WWDG} = {}^{1}/_{48000} * 4096 * 2^{3} * (63 + 1) = 43.69 \qquad (\text{ms})$$

23.3.5. 调试模式

当微控制器进入调试模式时（内核被暂停），计数器可以继续运行也可以被停止；这取决于 DBG 模块中的 DBG_WWDG_STOP 选项的配置。

23.4. WWDG 寄存器映射

下表给出了 WWDG 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	WWDG_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	WDGA	T[6:0]						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	1	1	1	1	1	1	1
0x04	WWDG_CFR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EWI	WDGTB1	WDGTB0	W[6:0]						
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	1	1	1	1	1	1	1
0x08	WWDG_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EWIF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0

23.4.1. 控制寄存器（WWDG_CR）

地址偏移：0x00

复位值：0x0000 007F

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	WDGA	T[6:0]						
类型	RS	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:8	NA	保留位，未定义
7	WDGA	激活位 此位由软件置 1，但仅能由硬件在复位后清零。 当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗复位。 1：使能看门狗复位。
6:0	T[6:0]	7 位计数器（MSB 到 LSB） 这些位用来存储看门狗的计数器值。每（$4096 * 2^{WDGTB}$）个 PCLK 周期减 1。 当计数器值从 0x40 变为 0x3F 时（T6 变成 0），产生看门狗复位。

23.4.2. 配置寄存器（WWDG_CFR）

地址偏移：0x04

复位值：0x0000 007F

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—						EWI	WDGTB
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RS	RW
7:0	WDGTB	W[6:0]						
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:10	NA	保留位，未定义
9	EWI	提前唤醒中断使能 此位置为 1，则当计数器值达到 0x40 时即产生中断。此中断使能只能在硬件复位后清除。
8:7	WDGTB[1:0]	时基 预分频器的时基可以设置如下： 00：CK 计时器时钟（PCLK 除以 4096）除以 1 01：CK 计时器时钟（PCLK 除以 4096）除以 2 10：CK 计时器时钟（PCLK 除以 4096）除以 4 11：CK 计时器时钟（PCLK 除以 4096）除以 8
6:0	W[6:0]	7 位窗口值 这些位包含了用来与递减计数器进行比较用的窗口值。

23.4.3. 状态寄存器（WWDG_SR）

地址偏移：0x08

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—							EWIF
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RC_W0

Bit	Name	Function
31:1	NA	保留位，未定义
0	EWIF	提前唤醒中断标志 当计数器的值达到 0x40 时，此位由硬件置 1。它必须通过软件写 0 来清除。 对此位写 1 无效。若中断未被启用，此位也会被置 1。

24. 实时时钟（RTC）

24.1. 介绍

实时时钟(RTC)是一个独立的 BCD 定时器和计数器。RTC 提供一个带有可编程闹钟中断的当前时钟/日历。两个 32 位寄存器包含了以二进制编码的十进制格式（BCD）表示的秒、分、小时（12/24 小时制）、星期（一周中的其中一天）、日（一月中的其中一天）、月和年。亚秒也是二进制格式。

自动执行 28、29（闰年）、30 和 31 天的月份补偿，也可以进行夏令时补偿。

额外的 32 位寄存器包含了可编程的闹钟亚秒、秒、分、小时、星期和日。

数字校准功能可以补偿晶振精度上的任何偏差。

在 RTC 域复位之后，所有 RTC 寄存器都会被写保护，以防止意外的写访问。

只要电源电压保持在工作范围内，无论器件状态如何（运行模式、低功耗模式或者复位状态），RTC 都不会停止。

24.2. RTC 主要特性

RTC 单元主要特性如下：

- 日历功能，可显示亚秒、秒、分、小时（12/24 小时制）、星期（一周中的其中一天）、日（一月中的其中一天）、月和年。
- 通过软件编程实现的夏令时补偿
- 带中断功能的可编程闹钟。闹钟可由任何日历字段的组合来触发。
- 参考时钟检测：可以使用更精准的第二个时钟源（50 或 60 Hz）来提高日历精度。
- 使用亚秒移位功能可以跟外部时钟精确同步。
- 数字校准电路（周期性计数器校正）: 0.95 ppm 精度，在几秒钟的校准窗口内可获得。
- 时间戳功能，用于事件保存。
- 篡改检测事件，带可配置的滤波器和内部上拉。
- 可屏蔽的中断/事件：
  - 闹钟 A
  - 时间戳
  - 篡改检测

24.3. RTC 功能描述

24.3.1. RTC 框图

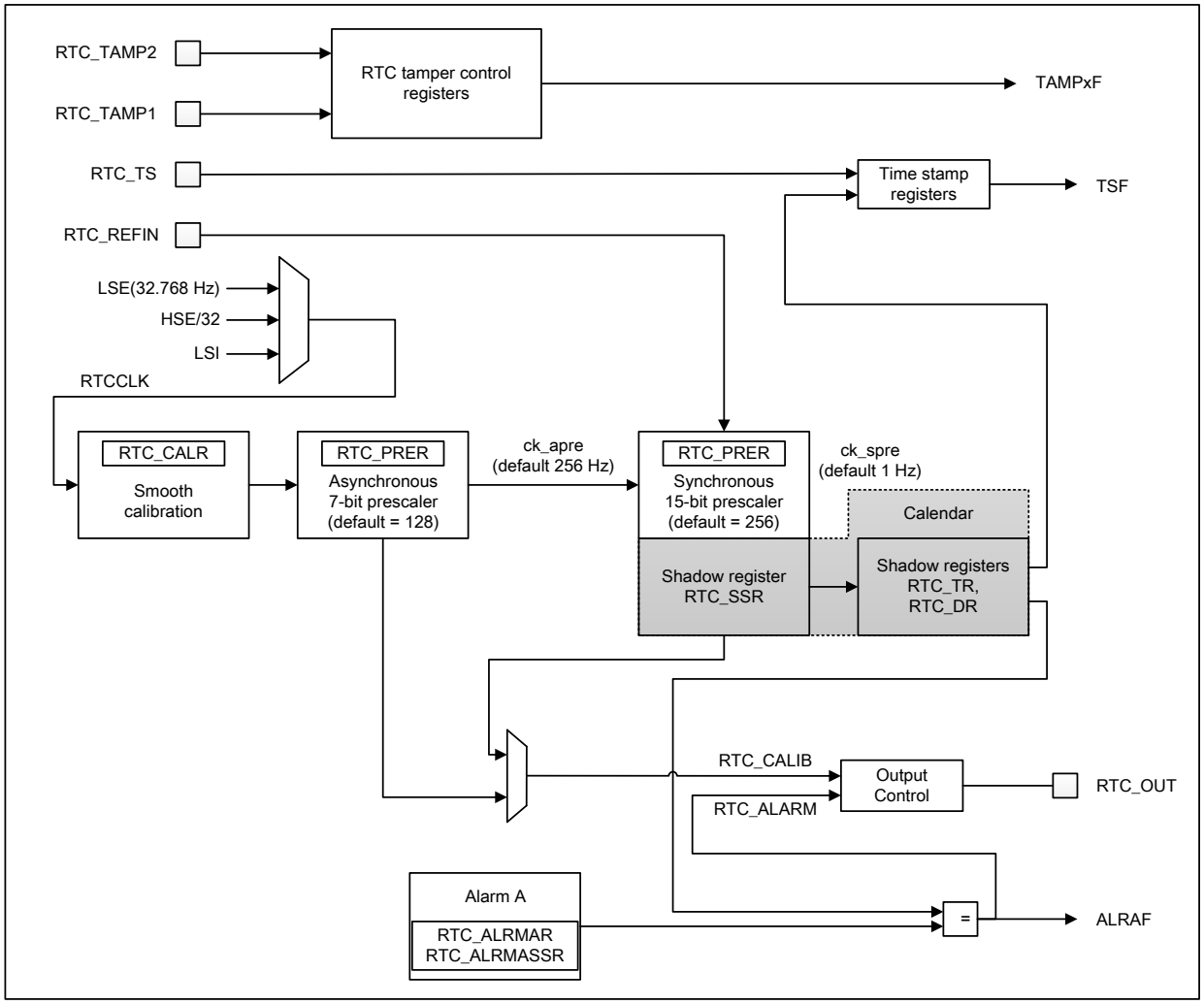


图 24.1. RTC 框图

RTC 包含：

- 一个闹钟
- 两个来自 I/O 口的篡改事件
- 一个来自 I/O 口的时间戳事件
- 篡改事件检测能产生一个时间戳事件
- 复用功能输出：RTC_OUT 选择以下两个之一输出：
  - RTC_CALIB：512 Hz 或 1 Hz 时钟输出（使用 32.768 kHz 的 LSE 频率时）。通过置位 RTC_CR 寄存器的 COE 位来使能这个输出。
  - RTC_ALARM：闹钟 A。通过配置 RTC_CR 寄存器的 OSEL[1:0]位选择这个输出。

- 复用功能输入：
  - RTC_TS：时间戳事件
  - RTC_TAMP1：篡改事件 1 检测
  - RTC_TAMP2：篡改事件 2 检测
  - RTC_REFIN：50 或 60 Hz 参考时钟输入

24.3.2. 由 RTC 控制 GPIO

RTC_OUT、RTC_TS 和 RTC_TAMP1 映射到相同的引脚（PC13）。
RTC_ALARM 输出选择是通过 RTC_TAFCR 寄存器进行的，如下表所示。使用 PC13VALUE 位选择 RTC_ALARM 输出模式是配置为推挽模式还是开漏模式。
当 PC13 没有被用作 RTC 复用功能时,可以通过置位 RTC_TAFCR 的 PC13MODE 位来强制其为输出推挽模式。输出数值由 PC13VALUE 位给出。在这种情况下，PC13 输出的推挽状态和数据都会在 Standby 模式下被保留。
输出机制遵循表 24.1 中的优先级顺序。
当 PC14 和 PC15 不被用作 LSE 振荡器功能时，可以通过分别置位 RTC_TAFCR 寄存器的 PC14MODE 和 PC15MODE 位来强制其为输出推挽模式。输出数值由 PC14VALUE 和 PC15VALUE 给出。在这种情况下，PC14 和 PC15 输出的推挽状态和数据都会在推挽模式下被保留。
输出机制遵循表 24.2 和表 24.3 的优先级顺序。

表 24.1. RTC 引脚 PC13 配置(1)

引脚配置和功能	RTC_ALARM 输出使能	RTC_CALIB 输出使能	RTC_TAMP1 输入使能	RTC_TS 输入使能	PC13MODE	PC13VALUE
RTC_ALARM output OD	1	不关心	不关心	不关心	不关心	0
RTC_ALARM output PP	1	不关心	不关心	不关心	不关心	1
RTC_CALIB output PP	0	1	不关心	不关心	不关心	不关心
RTC_TAMP1 input floating	0	0	1	0	不关心	不关心
RTC_TS and RTC_TAMP1 input floating	0	0	1	1	不关心	不关心
RTC_TS input floating	0	0	0	1	不关心	不关心
Output PP forced	0	0	0	0	1	PC13 输出数据
Wakeup pin or Standard GPIO	0	0	0	0	0	不关心

1. OD：开漏模式；PP：推挽模式。

表 24.2. LSE 引脚 PC14 配置(1)

引脚配置和功能	RCC_BDCR 寄存器的 LSEON 位	RCC_BDCR 寄存器的 LSEBYP 位	PC14MODE 位	PC14VALUE 位
LSE oscillator	1	0	不关心	不关心
LSE bypass	1	1	不关心	不关心
Output PP forced	0	不关心	1	PC14 输出数据
Standard GPIO	0	不关心	0	不关心

1. OD：开漏模式；PP：推挽模式。

表 24.3. LSE 引脚 PC15 配置(1)

引脚配置和功能	RCC_BDCR 寄存器的 LSEON 位	RCC_BDCR 寄存器的 LSEBYP 位	PC15MODE 位	PC15VALUE 位
LSE oscillator	1	0	不关心	不关心
Output PP forced	1	1	1	PC15 输出数据
	0	不关心		
Standard GPIO	0	不关心	0	不关心

1. OD：开漏模式；PP：推挽模式。

24.3.3. 时钟和预分频器

RTC 时钟源（RTCCLK）通过时钟控制器在 LSE 时钟、LSI 时钟和 HSE 时钟之间进行选择。有关 RTC 时钟源配置的更多信息，请参阅第 7 章：复位和时钟控制（RCC）。

一个可编程的预分频阶段会产生一个 1 Hz 时钟，用来更新日历。为了使功耗减到最少，该预分频器被分为 2 个可编程的预分频器（见图 24.1：RTC 框图）：

- 一个 7 位异步预分频器，通过 RTC_PRER 寄存器的 PREDIV_A 位来配置。
- 一个 15 位同步预分频器，通过 RTC_PRER 寄存器的 PREDIV_S 位来配置。

注：当所有预分频器都使用时，建议将异步预分频器配置为一个较高的值，以便使功耗减到最少。

在 LSE 频率为 32.768 kHz 时 ,将异步预分频系数设置为 128,同步预分频系数设置为 256 ,可以获得一个 1 Hz 的内部时钟频率（ck_spre）。

最小预分频系数为 1，最大预分频系数为 2^{22}，相当于最大输入频率大约为 4 MHz。

f_{CK_APRE} 由以下公式给出：

$$f_{CK\_APRE} = \frac{f_{RTCCLK}}{PREDIV\_A + 1}$$

ck_apre 时钟为二进制的 RTC_SSR 亚秒递减计数器提供时钟。当计数值到达 0 时，RTC_SSR 重新加载 PREDIV_S 的内容。

f_{CK_SPRE} 由以下公式给出：

$$f_{CK\_SPRE} = \frac{f_{RTCCLK}}{(PREDIV\_S + 1) \times (PREDIV\_A + 1)}$$

ck_spre 时钟用于更新日历。

24.3.4. 实时时钟和日历

RTC 日历的时间和日期寄存器是通过与 PCLK（APB 时钟）同步的影子寄存器来访问的。也可以直接访问它们，以避免等待同步时间。

- RTC_SSR 用于亚秒
- RTC_TR 用于时间
- RTC_DR 用于日期

每两个 RTCCLK 周期，当前日历值会被拷贝到影子寄存器中，并置位 RTC_ISR 寄存器的 RSF 位（参阅第 24.7.4 节：RTC 初始化和状态寄存器（RTC_ISR））。在 Stop 和 Standby 模式下不执行拷贝动作。当退出这些模式时，影子寄存器将在最多 2 个 RTCCLK 周期后被更新。

当应用程序读取日历寄存器时，它访问的是影子寄存器的内容。可以通过置位 RTC_CR 寄存器的 BYPSHAD 控制位来直接访问日历寄存器。默认情况下，该位是清零的，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（f_{APB}）的频率必须至少是 RTC 时钟（f_{RTCCLK}）频率的 7 倍。

影子寄存器会被系统复位信号复位。

24.3.5. 可编程闹钟

RTC 单元提供可编程闹钟：闹钟 A。

可编程闹钟功能通过 RTC_CR 寄存器的 ALRAE 位来使能。如果日历的亚秒、秒、分、小时、日或星期与闹钟寄存器 RTC_ALRMASSR 和 RTC_ALRMAR 中的编程值匹配上的话，则会将 ALRAF 设置为 1。每个日历字段可以通过 RTC_ALRMAR 寄存器的 MSKx 位和 RTC_ALRMASSR 寄存器的 MASKSSx 位来独立选择是否需要匹配。闹钟中断通过 RTC_CR 寄存器的 ALRAIE 位来使能。

注意：如果选择秒字段（清零 RTC_ALRMAR 的 MSK1 位），则设置在 RTC_PRER 寄存器中的同步预分频系数必须至少为 3，以确保行为正确。

闹钟 A（如果通过 RTC_CR 寄存器的 OSEL 位使能）可以布线到 RTC_ALARM 输出。RTC_ALARM 输出极性可以通过 RTC_CR 寄存器的 POL 位来配置。

24.3.6. RTC 初始化和配置

RTC 寄存器访问

RTC 寄存器是 32 位寄存器。除了在 BYPSHAD=0 时对日历影子寄存器的读访问，APB 接口在 RTC 寄存器访问时引入 2 个等待状态。

RTC 寄存器写保护

系统复位后，RTC 寄存器通过清零 PWR_CR 寄存器的 DBP 位（参阅电源控制章节）来防止意外的写访问。必须置位 DBP 位才能启用 RTC 寄存器的写访问。

在 RTC 域复位之后，所有 RTC 寄存器是写保护的。通过向写保护寄存器 RTC_WPR 写入密钥，可以使能 RTC 寄存器的写入。

要解锁除 RTC_TAFCR 和 RTC_ISR[13:8]外的所有 RTC 寄存器上的写保护，需要执行以下步骤：

1. 向 RTC_WPR 寄存器写入“0xCA”。

2. 向 RTC_WPR 寄存器写入“0x53”。

写入错误的密钥将会重新激活写保护机制。

写保护机制不受系统复位影响。

日历初始化和配置

要对初始时间和日期日历值（包括时间格式和预分频配置）进行编程，需要按照以下顺序：

1. 设置 RTC_ISR 寄存器的 INIT 位为 1 来进入初始化模式。在该模式下，日历计数器会被停止，并且可对其更新。
2. 轮询 RTC_ISR 寄存器的 INITF 位。初始化阶段模式在 INITF 被设置为 1 时进入。这个过程大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。
3. 要产生一个 1 Hz 时钟给日历计数器，需要编程 RTC_PRER 寄存器中的全部预分频系数。
4. 加载初始时间和日期值到影子寄存器（RTC_TR 和 RTC_DR）中，并通过 RTC_CR 寄存器的 FMT 位来配置小时格式（12 或 24 小时制）。
5. 通过清零 INIT 位来退出初始化模式。接着实际日历计数器值会被自动加载，并且在 4 个 RTCCLK 时钟周期之后重新开始计数。

初始化序列完成后，日历开始计数。

注：在系统复位之后，应用程序可以读取 RTC_ISR 寄存器的 INITS 标志来检查日历是否已经初始化。如果标志等于 0，因为年份字段还被设为 RTC 域复位默认值（0x00），所以日历还未被初始化。

要在初始化后读取日历，软件必须先检查 RTC_ISR 寄存器中的 RSF 标志是否已经被置位。

夏令时

通过 RTC_CR 寄存器的 SUB1H、ADD1H 和 BKP 位来管理夏令时。

通过设置 SUB1H 或 ADD1H 位，软件可以在不通过初始化流程的情况下将日历中的时间增加或减少一次 1 个小时。

另外，软件可通过 BKP 位来记录该操作。

编程闹钟

1. 编程或更新可编程闹钟时，必须遵循以下类似的过程：
2. 清零 RTC_CR 中的 ALRAE 来禁用闹钟 A。
3. 编程闹钟 A 寄存器（RTC_ALRMASSR / RTC_ALRMAR）
4. 再次置位 RTC_CR 中的 ALRAE 来使能闹钟 A

注：由于时钟同步，RTC_CR 寄存器的每次更改都要在大约 2 个 RTCCLK 时钟周期之后才会有效。

24.3.7. 读取日历

当 RTC_CR 寄存器中的 BYPSHAD 控制位被清零时

为了正确地读取 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。这确保同步机制的行为安全。

如果 APB 时钟频率小于 RTC 时钟频率的七倍，软件必须读取日历时间和日历日期寄存器各两次。如果 RTC_TR 的第二次读取跟第一次读取的结果一样，这就可以确保数据是正确的。否则必须执行第三次读取。

在任何情况下，APB 时钟频率必须不小于 RTC 时钟频率。

每当日历寄存器拷贝到 RTC_SSR、RTC_TR、RTC_DR 影子寄存器时，会置位 RTC_ISR 寄存器的 RSF 位。

每两次 RTCCLK 周期执行一次拷贝。为确保三者之间的一致性，读 RTC_SSR 或 RTC_TR 会锁定高阶日历影

子寄存器，直到 RTC_DR 被读取。万一软件在时间间隔小于 2 个 RTCCLK 周期的情况下执行日历读访问：RSF 必须在第一次读日历时通过软件清零，并且软件必须等到 RSF 被置位后才能再次读取 RTC_SSR、RTC_TR、RTC_DR 寄存器。

从低功耗模式（Stop 或 Standby）唤醒之后，RSF 必须通过软件清零。软件必须要等到 RSF 被再次置位后才能读取 RTC_SSR、RTC_TR、RTC_DR 寄存器。

RSF 位必须是在唤醒之后被清零，而不是在进入低功耗模式之前。

在系统复位之后，软件必须要等到 RSF 被置位后才能读取 RTC_SSR、RTC_TR、和 RTC_DR 寄存器。实际上，系统复位会复位影子寄存器到它的复位值。

在初始化后(参阅“日历初始化和配置”) :软件必须等到RSF被置位后才能读取RTC_SSR、RTC_TR和RTC_DR寄存器。

在同步之后（参阅第 24.3.9 节：RTC 同步）: 软件必须等到 RSF 被置位后才能读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

当 RTC_CR 寄存器中的 BYPSHAD 控制位被置位时（略过影子寄存器）

直接从日历计数器中读取日历寄存器的所需数值，因此无需等待 RSF 位被置位。这在退出低功耗模式之后特别有用，因为影子寄存器在这些模式中不会被更新。

当 BYPSHAD 位被置位为 1 时，如果 RTCCLK 边沿发生在两次寄存器读访问之间，那不同寄存器的结果可能会是相互不一致的。另外，如果 RTCCLK 边沿发生在读操作过程中，那其中一个寄存器的数值可能会是不正确的。软件必须要读取所有寄存器各两次，并且比较结果以确定数据是一致的和正确的。或者软件可以仅比较低阶日历寄存器的两个读取结果。

注：当 BYPSHAD=1 时，读取日历寄存器的指令需要一个额外的 APB 周期来完成操作。

24.3.8. 复位 RTC

通过所有可用的系统复位源都可以将日历影子寄存器（RTC_SSR、RTC_TR、RTC_DR）和 RTC 状态寄存器（RTC_ISR）的部分位复位为它们的默认值。

相反地，下列寄存器只会被 RTC 域复位信号复位成它们的默认值，而不受系统复位影响：RTC 当前日历寄存器、RTC 控制寄存器（RTC_CR）、预分频寄存器（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、RTC_TSDR）、RTC 篡改检测和复用功能配置寄存器（RTC_TAFCR）、闹钟 A 寄存器（RTC_ALRMASSR、RTC_ALRMAR）。

另外，当 RTC 由 LSE 时钟驱动时，如果复位源与 RTC 域复位源不同，则 RTC 在系统复位下保持运行（有关不受系统复位影响的 RTC 时钟源的详细信息，请参阅“复位和时钟控制器”的 RTC 时钟部分）。当 RTC 域发生复位，RTC 会被停止，并且所有 RTC 寄存器会被设置成它们的复位值。

24.3.9. RTC 同步

RTC 能与由高精度远程时钟进行同步。在读取亚秒字段（RTC_SSR 或 RTC_TSSSR）后，可计算出远程时钟和 RTC 中时间的精确偏差值。然后可以使用 RTC_SHIFTR 将其时钟“移位”若干分之一秒，从而调整 RTC 来消除这种偏差。

RTC_SSR 包含同步预分频计数器的数值。这允许计算 RTC 所保持的精确时间，精确到分辨率位 1/(PREDIV_S+1)秒。因此，可以通过增加同步预分频值（PREDIV_S[14:0]）来提高分辨率。允许的最大分辨率（30.52 µs / 32768 Hz 时钟）可通过 PREDIV_S 设置为 0x7FFF 来获得。

但是 ,增加 PREDIV_S 意味着必须减少 PREDIV_A ,以便将同步预分频器的输出保持在 1 Hz。通过这种方式，异步预分频器的输出频率将会增加，这可能会增加 RTC 动态功耗。

使用 RTC 移位控制寄存器（RTC_SHIFTR）可以微调 RTC。写入 RTC_SHIFTR 可以将时钟（延迟或提前）移位至多 1 秒 ,分辨率为 1/(PREDIV_S+1)秒。移位操作包括增加 SUBFS[14:0]值到同步预分频计数器 SS[15:0]：这将会延迟时钟。如果同时 ADD1S 位被置位 ,结果是增加一秒的同时减少若干分之一秒 ,所以这会提前时钟。

注：在初始化一个移位操作之前，用户必须检查 SS[15]是否等于 0，以确保不会发生溢出。

一旦移位操作通过写入 RTC_SHIFTR 寄存器来进行初始化，SHPF 标志就会通过硬件置位，以指示移位操作正在挂起。一旦移位操作完成，该位就会马上被硬件清零。

注：同步功能不兼容参考时钟检测功能：固件不可以在 REFCKON=1 时写入 RTC_SHIFTR。

24.3.10. RTC 参考时钟检测

RTC 日历的更新可以跟一个参考时钟进行自动同步 ,参考时钟 RTC_REFIN ,通常是电源频率(50 或 60 Hz)。RTC_REFIN 参考时钟的精度应该要大于 32.768 kHz LSE 时钟的精度。当使能了 RTC_REFIN 检测(RTC_CR 的 REFCKON 位被设置为 1),日历还是由 LSE 时钟驱动的 ,而 RTC_REFIN 是用于补偿日历更新频率(1 Hz)。每个 1 Hz 时钟边沿会跟最近的 RTC_REFIN 时钟边沿（在给定的时间窗口找到的）进行比较。在大多数情况下，两个时钟边沿是正确对齐的。当 1 Hz 时钟因为 LSE 时钟的不精确而变得不对齐时，RTC 移位 1 Hz 时钟一小点，以便后续的 1 Hz 时钟边沿能够被对齐。由于这个机制，日历可以变得和参考时钟一样精确。

RTC 通过使用由 32.768 kHz 晶振产生的 256 Hz 时钟来检测参考时钟源是否存在。检测是在围绕每个日历更新（每隔 1 秒）的时间窗口中执行的。当检测到第一个参考时钟边沿时，窗口等于 7 个 ck_apre 周期。3 个 ck_apre 周期的较小窗口用于后续的日历更新。

每次在窗口中检测到参考时钟时，输出 ck_apre 时钟的异步预分频器会被强制重载。当参考时钟和 1 Hz 时钟对齐时，这没有任何影响，因为预分频器的两个重载动作是在同一时刻执行的。当两个时钟未对齐时，重载可以将后续的 1Hz 时钟边沿移位一点点，以便它们可以跟参考时钟对齐。

如果参考时钟暂停（在 3 个 ck_apre 窗口中没有出现参考时钟边沿），日历只根据 LSE 时钟不断更新。然后 RTC 使用以 ck_spre 边沿为中心的 7 个 ck_apre 周期的大检测窗口等待参考时钟。

当使能 RTC_REFIN 检测时，PREDIV_A 和 PREDIV_S 必须设置为它们的默认值：

- PREDIV_A = 0x007F
- PREDIV_S = 0x00FF

注：RTC_REFIN 时钟检测在 Standby 模式下无效。

24.3.11. RTC 平滑数字校准

RTC 频率可以进行数字校准，分辨率约为 0.954 ppm，范围为-487.1 ppm 到+488.5 ppm。频率的校正是通过一系列的小调整来完成的（增加并/或减少单个 RTCCLK 脉冲）。这些调整分布得相当好，因此 RTC 会被校准得很好，即使是再观察一段时间也是一样的。

平滑数字校准在大约 2^{20} 个 RTCCLK 脉冲内进行的，若输入频率为 32768 Hz 则为 32 秒。这个周期由一个 20 位计数器 cal_cnt[19:0]来保持，其由 RTCCLK 时钟驱动。

平滑校准寄存器（RTC_CALR）指定了要在 32 秒周期内屏蔽的 RTCCLK 时钟周期数量：

- 将 CALM[0]位设置为 1，会导致在 32 秒周期内只会有一个脉冲被屏蔽。
- 将 CALM[1]位设置为 1，会导致两个额外的周期被屏蔽。
- 将 CALM[2]位设置为 1，会导致四个额外的周期被屏蔽。

- 以此类推，直到将 CALM[8]位设置为 1，会导致额外的 256 个时钟被屏蔽。

注：CALM[8:0]（RTC_CALR）指定了要在 32 秒周期内屏蔽的 RTCCLK 脉冲数。将 CALM[0]设置为 1，在 32 秒周期内的 cal_cnt[19:0]为 0x80000 的时刻，只会屏蔽这一个脉冲；CALM[1]=1 时会导致有两个其它的周期被屏蔽（在 cal_cnt 为 0x40000 和 0xC0000）；CALM[2]=1 时会导致有四个其它的周期被屏蔽（cal_cnt = 0x20000 / 0x60000 / 0xA0000 / 0xE0000）；以此类推，直到 CALM[8]=1 时会导致有 256 个其它的周期被屏蔽（cal_cnt = 0xXX800）。

CALM 可以使 RTC 的频率降低 487.1 ppm，并具有更高的分辨率，而 CALP 位可以使 RTC 的频率增加 488.5 ppm。将 CALP 设置为 1，每隔 2^{11} 个 RTCCLK 周期就会插入一个额外的 RTCCLK 脉冲，也就是说，每 32 秒增加 512 个时钟。

与 CALP 一起使用 CALM，在 32 秒周期内可以增加从-511 到+512 个 RTCCLK 周期的偏移范围，即校正范围为-487.1 ppm 到+488.5 ppm，而分辨率约为 0.954 ppm。

给定输入频率（F_{RTCCLK}）计算有效校准频率（F_{CAL}）的公式如下：

$$F_{CAL} = F_{RTCCLK} \times [1 + (CALP \times 512 - CALM) / (2^{20} + CALM - CALP \times 512)]$$

在 PREDIV_A<3 时校准

当异步预分频值（RTC_PRER 寄存器的 PREDIV_A 位）小于 3 时，CALP 位不能被设置为 1。当 CALP 已经被设置为 1 而 PREDIV_A 位被设置为小于 3 的值时，CALP 会被忽略，并且校准操作就像 CALP 等于 0 一样。

若要执行 PREDIV_A 小于 3 的校准，应该减少同步预分频值（PREDIV_S），以便每秒钟加快 8 个 RTCCLK 时钟周期，这相当于每 32 秒增加 256 个时钟周期。因此，仅使用 CALM 位就可以在每个 32 秒周期内有效地增加-255 到 256 个时钟脉冲（对应的校准范围：-243.3 到 244.1ppm）。

使用 32768 Hz 的额定 RTCCLK 频率下，当 PREDIV_A 等于 1（分频系数为 2），PREDIV_S 应该被设置为 16379 而不是 16383（要减去 4）。另一个有趣的情况是当 PREDIV_A=0 时，PREDIV_S 应该被设置为 32759 而不是 32767（要减去 8）。

若将 PREDIV_S 按此方式约简，则给定校准输入时钟有效频率的公式为：

$$F_{CAL} = F_{RTCCLK} \times [1 + (256 - CALM) / (2^{20} + CALM - 256)]$$

在这种情况下，如果 RTCCLK 恰好是 32768.00 Hz，那么 CALM[7:0]等于 0x100（CALM 范围的中点）是正确的设置。

验证 RTC 校准

通过测量 RTCCLK 的精确频率和计算正确的 CALM 值与 CALP 值来保证 RTC 的精度。提供一个可选的 1 Hz 输出，以允许应用程序来测量和验证 RTC 精度。

在一个有限的时间间隔内测量 RTC 的精确频率会导致在测量周期内有最多 2 个 RTCCLK 时钟周期的测量误差，这取决于数字校准周期如何与测量周期对齐。

但是，如果测量周期与校准周期相同，则可以消除这种测量误差。在这种情况下，唯一观察到的误差就是由于数字校准的分辨率引起的误差。

- 默认情况下，校准周期为 32 秒。

使用这种模式和测量在精确 32 秒内 1 Hz 输出的精度，可以保证测量值在 0.477 ppm 以内（由于校准分辨率的限制，在 32 秒内有 0.5 个 RTCCLK 周期误差）。

- RTC_CALR 寄存器的 CALW16 位可设置为 1，以强制使用 16 秒的校准周期。

在这种情况下，可以在 16 秒内测量 RTC 精度，最大误差为 0.954 ppm（在 16 秒内有 0.5 个 RTCCLK 周期误差）。然而，由于校准分辨率的降低，从长远来看 RTC 精度也降低到了 0.954 ppm：当 CALW16 被设置为 1 时，CALM[0]位被固定为 0。

- RTC_CALR 寄存器的 CALW8 位可设置为 1，以强制使用 8 秒的校准周期。

在这种情况下，可以在 8 秒内测量 RTC 精度，最大误差为 1.907 ppm（在 8 秒内有 0.5 个 RTCCLK 周期误差）。从长远来看 RTC 精度也降低到了 1.907 ppm：当 CALW8 被设置为 1 时，CALM[1:0]位被固定为 00。

运行中重校准

在 RTS_ISR 的 INITF=0 时，校准寄存器（RTC_CALR）可以通过使用以下流程在运行中被更新：

1. 轮询 RTC_ISR 的 RECALPF（重校准挂起标志）。
2. 如果其被设置为 0，则在必要时将新值写入 RTC_CALR。然后 RECALPF 位会被自动设置为 1。
3. 在对 RTC_CALR 的写操作后的三个 ck_apre 周期以内，新的校准设定将会生效。

24.3.12. 时间戳功能

通过设置 RTC_CR 寄存器的 TSE 位为 1 来使能时间戳。

当在 RTC_TS 引脚上检测到时间戳事件时，日历值会被保存在时间戳寄存器（RTC_TSSSR、RTC_TSTR、RTC_TSDR）中。

当有一个时间戳事件发生时，RTC_ISR 寄存器中的时间戳标志位（TSF）会被置位。

通过置位 RTC_CR 寄存器中的 TSIE 位，在时间戳事件发生时会产生一个中断。

如果新的时间戳事件在时间戳标志（TSF）已经置位的情况下被检测到，时间戳溢出标志（TSOVF）会被置位，并且时间戳寄存器（RTC_TSTR、RTC_TSDR）还会保持先前事件的结果。

注：由于同步过程，TSF 会在时间戳事件发生之后的 2 个 ck_apre 周期时被置位。

TSOVF 的置位没有延迟 ,这意味着如果两个时间戳事件靠的太近 ,在 TSF 还是 0 的时候 TSOVF 会被认为 1。因此，推荐仅在 TSF 已经被置位后轮询 TSOVF。

注意：假如在 TSF 刚好被清零时立马发生了一个时间戳事件，那么 TSF 和 TSOVF 位都会被置位。为避免在相同时刻屏蔽了时间戳事件的发生，应用程序不能向 TSF 位写入 0，除非它已经将被读为 1。

可选地，篡改事件可以引起一个时间戳以便可被记录。有关 TAMPTS 控制位的描述，请参考章节 24.6.14：RTC_TAFCR（RTC 篡改和复用功能配置寄存器）。

24.3.13. 篡改检测

RTC_TAMPx 输入事件可以被配置为边沿检测，或者带滤波的电平检测。

篡改检测可以被配置为以下用途：

- 产生一个中断
- 可以从 Stop 和 Standby 模式下唤醒。

篡改检测初始化

每一个输入都能够通过设置 RTC_TAFCR 寄存器中对应的 TAMPxE 位为 1 来使能。

每一个 RTC_TAMPx 篡改检测输入都与 RTC_ISR 寄存器的一个标志位 TAMPxF 相关联。

TAMPxF 标志会在引脚上的篡改事件发生后置位，延迟时间如下：

- 当 TAMPFLT 不为 0x0（带滤波的电平检测）时为 3 个 ck_apre 周期。
- 当 TAMPTS=1（篡改事件的时间戳）时为 3 个 ck_apre 周期。
- 当 TAMPFLT=0x0（边沿检测）并且 TAMPTS=0 时没有延迟。

在此期间，只要置位了 TAMPxF，就无法检测到在同一引脚上发生的新篡改事件。

通过置位 RTC_TAFCR 寄存器的 TAMPIE 位，在一个篡改检测事件发生时会产生一个中断。

篡改事件的时间戳

当 TAMPTS 设置为 1 时 ,任何篡改事件都会导致时间戳发生。在这种情况下 ,RTC_ISR 的 TSF 位或者 TSOVF 位都可以被置位，其方式与发生正常时间戳事件的方式相同。受影响的篡改标志寄存器 TAMPxF 与 TSF/TSOVF 同时置位。

篡改输入的边沿检测

如果 TAMPFLT 位是 00 时 , 根据相应的 TAMPxTRG 位 , 当 RTC_TAMPx 引脚检测到上升沿或者下降沿时会产生篡改检测事件。当选择边沿检测时，RTC_TAMPx 输入引脚上的内部上拉电阻将会失效。

注意：为避免丢失篡改检测事件，用于边沿检测的信号与相应的 TAMPxE 位进行逻辑或运算，以便能够检测得到发生在 RTC_TAMPx 引脚被使能之前的篡改检测事件。

- 当 TAMPxTRG=0：如果 RTC_TAMPx 复用功能在篡改检测被使能（TAMPxE 位设置为 1）之前已经是高的，一旦启用 RTC_TAMPx 输入，就会检测到篡改事件，即使在置位 TAMPxE 后 RTC_TAMPx 输入并没有上升沿。
- 当 TAMPxTRG=1：如果 RTC_TAMPx 复用功能在篡改检测被使能（TAMPxE 位设置为 1）之前已经是低的，一旦启用 RTC_TAMPx 输入，就会检测到篡改事件，即使在置位 TAMPxE 后 RTC_TAMPx 输入并没有下降沿。

带 RTC_TAMPx 输入滤波的电平检测

通过将 TAMPFLT 设置为非零值来执行带过滤的电平检测。当连续采样到 2 个、4 个或 8 个(取决于 TAMPFLT) TAMPxTRG 位指定电平时，将产生一个篡改检测事件。

RTC_TAMPx 输入引脚在采样前通过 I/O 内部上拉电阻进行预充电，可将 TAMPPUDIS 设置为 1 来禁用该功能。预充电的持续时间由 TAMPPRCH 位决定，这允许 RTC_TAMPx 输入引脚可以有更大的电容。

通过使用 TAMPFREQ 去确定电平检测的采样频率，优化篡改检测延迟和上拉功耗之间的取舍。

注：有关上拉电阻的电气特性，请参阅数据手册。

24.3.14. 校准时钟输出

当 RTC_CR 寄存器的 COE 位被设置为 1 时，一个参考时钟提供 RTC_CALIB 输出。

如果 RTC_CR 寄存器的 COSEL 位被清零，并且 PREDIV_A=0x7F，则 RTC_CALIB 频率为 $f_{RTCCLK}/64$。这对应在 RTCCLK 频率为 32.768 Hz 下 512 Hz 校准输出。RTC_CALIB 占空比是不规则的：下降沿有轻微抖动。因此推荐使用上升沿。

当 COSEL 被置位，并且“PREDIV_S+1”是 256 的非零倍数（例如：PREDIV_S[7:0]=0xFF）时，RTC_CALIB 频率是 $f_{RTCCLK}/(256*(PREDIV\_A+1))$。这对应在 RTCCLK 频率是 32.768 Hz 下预分频为默认值（PREDIV_A=0x7F，PREDIV_S=0xFF）时 1 Hz 校准输出。1 Hz 输出会在移位操作正在进行时受到影响，并且可能会在移位操作中（SHPF=1）发生翻转。

注：当选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚将会被自动配置为输出复用功能。

当 COSEL 位被清零时，RTC_CALIB 输出为异步预分频器的第 6 级输出。

当 COSEL 位被置位时，RTC_CALIB 输出为同步预分频器的第 8 级输出。

24.3.15. 闹钟输出

RTC_CR 寄存器的 OSEL 控制位被用于激活闹钟复用功能输出 RTC_ALARM。
输出的极性由 RTC_CR 中的 POL 控制位确定，因此当 POL 被设置为 1 时，输出与所选标志位为相反极性。
闹钟复用功能输出
通过使用 RTC_TAFCR 寄存器中的 PC13VALUE 控制位 ,RTC_ALARM 引脚能够被配置为开漏输出还是推挽输出。
注：一旦使能了 RTC_ALARM 输出，它的优先级就高于 RTC_CALIB（COE 位不关心，必须保持清零）。
当选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚将被自动配置为输出复用功能。

24.4. RTC 低功耗模式

表 24.4. 低功耗模式对 RTC 的影响

模式	描述
Sleep	没影响 RTC 中断可使得设备退出 Sleep 模式
Stop	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持活跃状态。RTC 闹钟、RTC 篡改事件和 RTC 时间戳事件可使得设备退出 Stop 模式。
Standby	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持活跃状态。RTC 闹钟、RTC 篡改事件和 RTC 时间戳事件可使得设备退出 Standby 模式。

24.5. RTC 中断

所有 RTC 中断都连接到 NVIC 控制器。
要使能 RTC 中断，需要按照以下顺序：
1. 配置和使能与中断模式下的 RTC 事件相对应的 NVIC 线，并且选择上升沿敏感。
2. 配置和使能 NVIC 中的 RTC IRQ 通道。
3. 配置 RTC 以产生 RTC 中断。

表 24.5. 中断控制位

中断事件	事件标志	使能控制位	退出 Sleep 模式	退出 Stop 模式	退出 Standby 模式
闹钟 A	ALRAF	ALRAIE	yes	yes(1)	yes(1)
RTC_TS 输入(时间戳)	TSF	TSIE	yes	yes(1)	yes(1)
RTC_TAMP1 输入检测	TAMP1F	TAMPIE	yes	yes(1)	yes(1)
RTC_TAMP2 输入检测	TAMP2F	TAMPIE	yes	yes(1)	yes(1)

1. 只有当 RTC 时钟源为 LSE 或 LSI 时，才能从 Stop 和 Standby 模式中唤醒。

24.6. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RTC_TR	-	-	-	-	-	-	-	-	-	PM	HT[1:0]		HU[3:0]				-	MNT[2:0]			MNU[3:0]				-	ST[2:0]			SU[3:0]			
	Reset	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0
0x04	RTC_DR	-	-	-	-	-	-	-	-	YT[3:0]				YU[3:0]				WDU[2:0]			MT	MU[3:0]				-	-	DT[1:0]		DU[3:0]			
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	x	x	0	0	0	0	0	1
0x08	RTC_CR	-	-	-	-	-	-	-	-	COE	-	OSEL	POL	COSEL	BKP	SUB1H	ADD1H	TSIE	-	-	ALRAIE	TSE	-	-	ALRAE	-	FMT	BYPSHAD	REFCKON	TSEDGE	-	-	-
	Reset	x	x	x	x	x	x	x	x	0	x	0	0	0	0	0	0	0	x	x	0	0	x	x	0	x	0	0	0	0	x	x	x
0x0C	RTC_ISR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RECALPF	-	TAMP2F	TAMP1F	TSOVF	TSF	-	-	ALRAF	INIT	INITF	RSF	INITS	SHPF	-	-	ALRAWF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	0	0	0	0	x	x	0	0	0	0	0	0	x	x	1
0x10	RTC_PRER	-	-	-	-	-	-	-	-	-	PREDIV_A[6:0]							-	PREDIV_S[14:0]															
	Reset	x	x	x	x	x	x	x	x	x	1	1	1	1	1	1	1	x	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x1C	RTC_ALRMAR	MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]				MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	RTC_WPR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	KEY[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x28	RTC_SSR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SS[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	RTC_SHIFTR	ADD1S	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SUBFS[14:0]															
	Reset	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	RTC_TSTR	-	-	-	-	-	-	-	-	-	PM	HT[1:0]		HU[3:0]				-	MNT[2:0]			MNU[3:0]				-	ST[2:0]			SU[3:0]			
	Reset	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0
0x34	RTC_TSDR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	WDU[2:0]			MT	MU[3:0]				-	-	DT[1:0]		DU[3:0]			
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0
0x38	RTC_TSSSR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SS[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	RTC_CALR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CALP	CALW8	CALW16	-	-	-	-	CALM[8:0]								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0	0	0	0	0	0	0	0	0
0x40	RTC_TAFCR	-	-	-	-	-	-	-	-	PC15MODE	PC15VALUE	PC14MODE	PC14VALUE	PC13MODE	PC13VALUE	-	-	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]			TAMPTS	-	-	TAMP2TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0
0x44	RTC_ALRMASSR	-	-	-	-	MASKSS				-	-	-	-	-	-	-	-	-	SS[14:0]														
	Reset	x	x	x	x	0	0	0	0	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

24.6.1. RTC_TR（RTC 时间寄存器）

RTC_TR 是日历时间影子寄存器。该寄存器只能在初始化模式下写入。请参阅“日历初始化和配置”以及“读日历”。

该寄存器是写保护的。写操作步骤描述在“RTC 寄存器写保护”。

地址偏移：0x00

RTC 域复位值：0x0000 0000

系统复位：当 BYPSHAD=0 时为 0x0000 0000。当 BYPSHAD=1 时没影响。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-	PM	HT[1:0]		HU[3:0]			
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
15:8	-	MNT[2:0]			MNU[3:0]			
类型	RO-0	RW	RW	RW	RW	RW	RW	RW
7:0	-	ST[2:0]			SU[3:0]			
类型	RO-0	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:23	NA	保留位，未定义
22	PM	AM/PM 标记 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	小时的十位数（BCD 格式）
19:16	HU[3:0]	小时的个位数（BCD 格式）
15	NA	保留位，未定义
14:12	MNT[2:0]	分的十位数（BCD 格式）
11:8	MNU[3:0]	分的个位数（BCD 格式）
7	NA	保留位，未定义
6:4	ST[2:0]	秒的十位数（BCD 格式）
3:0	SU[3:0]	秒的个位数（BCD 格式）

24.6.2. RTC_DR（RTC 日期寄存器）

RTC_DR 是日历日期影子寄存器。该寄存器只能在初始化模式下写入。请参阅“日历初始化和配置”以及“读日历”。

该寄存器是写保护的。写操作步骤描述在“RTC 寄存器写保护”。

地址偏移：0x04

RTC 域复位值：0x0000 2101

系统复位：当 BYPSHAD=0 时为 0x0000 2101。当 BYPSHAD=1 时没影响。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	YT[3:0]				YU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	WDU[2:0]			MT	MU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

7:0	-		DT[1:0]		DU[3:0]			
类型	RO-0	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	NA	保留位，未定义
23:20	YT[3:0]	年的十位数（BCD 格式）
19:16	YU[3:0]	年的个位数（BCD 格式）
15:13	WDU[2:0]	星期几 000：禁用 001：星期一 ... 111：星期日
12	MT	月的十位数（BCD 格式）
11:8	MU[3:0]	月的个位数（BCD 格式）
7:6	NA	保留位，未定义
5:4	DT[1:0]	日的十位数（BCD 格式）
3:0	DU[3:0]	日的个位数（BCD 格式）

24.6.3. RTC_CR（RTC 控制寄存器）

地址偏移：0x08
RTC 域复位值：0x0000 0000
系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	COE	-	OSEL	POL	COSEL	BKP	SUB1H	ADD1H
类型	RW	RO-0	RW	RW	RW	RW	WO	WO
15:8	TSIE	-		ALRAIE	TSE	-		ALRAE
类型	RW	RO	RO	RW	RW	RO-0	RO-0	RW
7:0	-	FMT	BYPSHAD	REFCKON	TSEDGE	-		
类型	RO-0	RW	RW	RW	RW	RO-0	RO-0	RO-0

Bit	Name	Function
31:24	NA	保留位，未定义
23	COE	校准输出使能 该位使能 RTC_CALIB 输出 0：禁用校准输出 1：使能校准输出
22	NA	保留位，未定义
21	OSEL	输出选择

		该位用于选择标志位是否布线到 RTC_ALARM 输出。 0：禁用输出 1：使能闹钟 A 输出
20	POL	输出极性 该位用于配置 RTC_ALARM 输出的极性 0：当 ALRAF 有效时，引脚为高电平 1：当 ALRAF 有效时，引脚为低电平
19	COSEL	校准输出选择 当 COE=1 时，该位选择哪一个信号输出到 RTC_CALIB 上。 0：校准输出为 512 Hz 1：校准输出为 1 Hz 这些频率在 RTCCLK 为 32.768 kHz 和预分频器处于默认值(PREDIV_A=127 和 PREDIV_S=255）时有效。请参阅章节 24.3.14：校准时钟输出。
18	BKP	备份 该位可由用户写入，用于记录夏令时是否已经发生变化。
17	SUB1H	减一小时（冬季时间变化） 当该位被置位时，如果当前小时位不是 0，日历时间将减去一个小时。该位读为 0。 如果当前小时位是 0，则置位该位将无效。 0：没影响 1：当前时间减去一个小时。这可用于在初始化模式以外的冬季时间变化。
16	ADD1H	加一小时（夏季时间变化） 当该位被置位时，日历时间将增加一个小时。该位读为 0。 0：没影响 1：当前时间增加一个小时。这可用于在初始化模式以外的夏季时间变化。
15	TSIE	时间戳中断使能 0：禁用时间戳中断 1：使能时间戳中断
14:13	NA	保留位，未定义
12	ALRAIE	闹钟 A 中断使能 0：禁用闹钟 A 中断 1：使能闹钟 A 中断
11	TSE	时间戳使能 0：禁用时间戳 1：使能时间戳
10:9	NA	保留位，未定义
8	ALRAE	闹钟 A 使能 0：禁用闹钟 A 1：使能闹钟 A
7	NA	保留位，未定义
6	FMT	小时制 0：24 小时制 1：AM/PM 小时制

5	BYPSHAD	略过影子寄存器 0：日历值（在读取 RTC_SSR、RTC_TR、RTC_DR 时）来自于影子寄存器，影子寄存器每两个 RTCCLK 周期更新一次。 1：日历值（在读取 RTC_SSR、RTC_TR、RTC_DR 时）直接来源于日历计数器。 注：如果 APB 时钟频率小于 RTCCLK 频率的七倍，BYPSHAD 必须要设置为 1。
4	REFCKON	RTC_REFIN 参考时钟检测使能（50 或 60 Hz） 0：禁用 RTC_REFIN 检测 1：使能 RTC_REFIN 检测 注：PREDIV_S 必须设置为 0x00FF。
3	TSEDGE	时间戳事件有效边沿 0：RTC_TS 输入上升沿产生一个时间戳事件 1：RTC_TS 输入下降沿产生一个时间戳事件 当 TSEDGE 发生改变时 TSE 必须为零，以避免不需要的 TSF 置位。
2:0	NA	保留位，未定义

注：该寄存器的第 7、6、4 位只能在初始化模式下写入（RTC_ISR/INTF=1）。
建议不要在日历小时增加期间去更改小时，因为这会屏蔽掉日历小时的增加。
ADD1H 和 SUB1H 的更改将会在下一秒生效。
这个寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。
注意：当 TSEDGE 发生改变时 TSE 必须为零，以避免 TSF 的虚假置位。

24.6.4. RTC_ISR（RTC 初始化和状态寄存器）

该寄存器是写保护的（除了 RTC_ISR[14:8]位）。写访问步骤被描述在“RTC 寄存器写保护”。
地址偏移：0x0C
RTC 域复位值：0x0000 0007
系统复位：除了 INIT、INITF 和 RSF 位会被清零外，其它的无影响。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							RECALPF
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO
15:8	-	TAMP2F	TAMP1F	TSOVF	TSF	-		ALRAF
类型	RO-0	RC_W0	RC_W0	RC_W0	RC_W0	RO-0	RO-0	RC_W0
7:0	INIT	INITF	RSF	INITS	SHPF	-		ALRAWF
类型	RW	RO	RC_W0	RO	RO	RO-0	RO-0	RO

Bit	Name	Function
31:17	NA	保留位，未定义

16	RECALPF	重校准挂起标志 当软件写 RTC_CALR 寄存器时，RECALPF 状态标志位会被自动设置为 1，表示 RTC_CALR 寄存器已被锁定。当新的校准设定生效时，该位将会回到 0。请参考“运行中重校准”。
15	NA	保留位，未定义
14	TAMP2F	RTC_TAMP2 检测标志 当在 RTC_TAMP2 输入上检测到篡改检测事件时，该标志由硬件置位。 该位由软件写 0 清零。
13	TAMP1F	RTC_TAMP1 检测标志 当在 RTC_TAMP1 输入上检测到篡改检测事件时，该标志由硬件置位。 该位由软件写 0 清零。
12	TSOVF	时间戳溢出标志 当在 TSF 已经被置位时发生一个时间戳事件，该标志会由硬件置位。 该位由软件写 0 清零。建议仅在清零 TSF 位后检查并清零 TSOVF。否则，如果在 TSF 位清零之前立即发生时间戳事件的话，则可能不会注意到溢出事件。
11	TSF	时间戳标志 当时间戳事件发生时，该标志将由硬件置位。 该标志由软件写 0 清零。
10:9	NA	保留位，未定义
8	ALRAF	闹钟 A 标志 当时间/日期寄存器（RTC_TR 和 RTC_DR）匹配闹钟 A 寄存器（RTC_ALRMAR）时，该标志将由硬件置位。 该标志由软件写 0 清零。
7	INIT	初始化模式 0：自由运行模式 1：初始化模式用于编程时间和日期寄存器（RTC_TR 和 RTC_DR），还有预分频寄存器（RTC_PRER）。当 INIT 清零时，计数器将停止并从新值开始计数。
6	INITF	初始化标志 当该位被设置为 1 时，RTC 处于初始化状态，可以更新时间、日期和预分频寄存器。 0：日历寄存器不允许更新 1：日历寄存器允许更新
5	RSF	寄存器同步标志 每次将日历寄存器复制到影子寄存器（RTC_SSRx、RTC_TRx 和 RTC_DRx）时，该位将会被硬件置位。当处于初始化模式或处于移位操作挂起状态（SHPF=1）或处于略过影子寄存器模式（BYPSHAD=1）时，硬件将清零该位。该位也可以由软件清零。 在初始化模式下，可通过软件或硬件清零该位。 0：日历影子寄存器尚未同步 1：日历影子寄存器已经同步
4	INITS	初始化状态标志 当日历年份字段不为 0（RTC 域复位状态）时，该位将由硬件置位。

		0：日历尚未被初始化 1：日历已经被初始化
3	SHPF	移位操作挂起标志 0：没有移位操作被挂起 1：有移位操作被挂起
2:1	NA	保留位，未定义
0	ALRAWF	闹钟 A 可写标志 在将 RTC_CR 中的 ALRAE 位设置为 0 之后，该位会在闹钟 A 数值可被更改时由硬件置位。 该位由硬件在初始化模式下清零。 0：闹钟 A 不可更新 1：闹钟 A 可以更新

注：ALRAF 和 TSF 位将在被编程为 0 后的 2 个 APB 时钟周期之后被清零。

24.6.5. RTC_PRER（RTC 预分频寄存器）

该寄存器仅可以在初始化模式下被写入。初始化必须由两次独立的写访问完成。参考“日历初始化和配置”。
该寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。
地址偏移：0x10
RTC 域复位值：0x007F 00FF
系统复位：没影响

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

Bit	Name	Function
31:23	NA	保留位，未定义
22:16	PREDIV_A[6:0]	异步预分频系数 ck_apre 频率=RTCCLK 频率/(PREDIV_A+1)
15	NA	保留位，未定义
14:0	PREDIV_S[14:0]	同步预分频系数 ck_spre 频率=ck_apre 频率/(PREDIV_S+1)

24.6.6. RTC_ALRMAR（RTC 闹钟 A 寄存器）

该寄存器仅在 RTC_ISR 的 ALRAWF 设置为 1 或在初始化模式下可写。
该寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。
地址偏移：0x1C
RTC 域复位值：0x0000 0000
系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	MSK4	WDSEL	DT[1:0]		DU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	MSK3	PM	HT[1:0]		HU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	MSK2	MNT[2:0]			MNU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	MSK1	ST[2:0]			SU[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31	MSK4	闹钟 A 日期屏蔽位 0：如果日/星期匹配，则闹钟 A 置位 1：在闹钟 A 比较中忽略日/星期
30	WDSEL	星期选择 0：DU[3:0]表示日 1：DU[3:0]表示星期。DT[1:0]是不关心的。
29:28	DT[1:0]	日的十位数（BCD 格式）
27:24	DU[3:0]	日的个位数（BCD 格式）
23	MSK3	闹钟 A 的小时屏蔽位 0：如果小时匹配，则闹钟 A 置位 1：在闹钟 A 比较中忽略小时
22	PM	AM/PM 标记 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	小时的十位数
19:16	HU[3:0]	小时的个位数
15	MSK2	闹钟 A 的分钟屏蔽位 0：如果分钟匹配，则闹钟 A 置位 1：在闹钟 A 比较中忽略分钟
14:12	MNT[2:0]	分的十位数（BCD 格式）
11:8	MNU[3:0]	分的个位数（BCD 格式）
7	MSK1	闹钟 A 的秒钟屏蔽位 0：如果秒钟匹配，则闹钟 A 置位

		1：在闹钟 A 比较中忽略秒钟
6:4	ST[2:0]	秒的十位数（BCD 格式）
3:0	SU[2:0]	秒的个位数（BCD 格式）

24.6.7. RTC_WPR（RTC 写保护寄存器）

地址偏移：0x24
复位值：0x0000 0000

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	KEY[7:0]	写保护密钥 该字节由软件写入。 读该字节永远返回 0x00。 有关如何解锁 RTC 寄存器写保护的描述，请参考“RTC 寄存器写保护”。

24.6.8. RTC_SSR（RTC 亚秒寄存器）

地址偏移：0x28
RTC 域复位值：0x0000 0000
系统复位：当 BYPSHAD=0 时为 0x0000 0000。当 BYPSHAD=1 时没影响。

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	SS[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	SS[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	SS[15:0]	亚秒数值 SS[15:0]是同步预分频计数器中的数值。秒的小数部分由下列公式给出： Second fraction = (PREDIV_S - SS) / (PREDIV_S + 1) 注：只有在移位操作之后，SS 才可能大于 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/日期要少一秒。

24.6.9. RTC_SHIFTR（RTC 移位控制寄存器）

该寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。

地址偏移：0x2C

RTC 域复位值：0x0000 0000

系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	ADD1S	-						
类型	WO	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	-	SUBFS[14:8]						
类型	RO-0	WO	WO	WO	WO	WO	WO	WO
7:0	SUBFS[7:0]							
类型	WO	WO	WO	WO	WO	WO	WO	WO

Bit	Name	Function
31	ADD1S	增加一秒 0：没影响 1：时钟/日历增加一秒 该位是只写的，总是读为零。当移位操作挂起时（当 RTC_ISR 中的 SHPF 为 1 时），写该位是无效的。 这个功能将于 SUBFS（参见下面的描述）一起使用，以便在原子操作中有效地向时钟添加若干分之一秒。
30:15	NA	保留位，未定义
14:0	SUBFS[14:0]	减去若干分之一秒 该位是只写的，总是读为零。当移位操作挂起时（当 RTC_ISR 中的 SHPF 为 1 时），写该位是无效的。 写 SUBFS 的数值会被加到同步预分频计数器上。因为这个计数器是倒计时的，所以这个操作可以有效地减去（延时）时钟： Delay (seconds) = SUBFS / (PREDIV_S + 1) 当 ADD1S 功能与 SUBFS 一起使用时，可以有效地将若干分之一秒添加到时

		钟中（提前时钟），从而有效地提前时钟： Advance (seconds) = (1 - (SUBFS / (PREDIV_S + 1))) 注：写 SUBFS 会导致 RSF 被清零。然后，软件可以等待 RSF=1，以确保影子寄存器已更新为移位后的时间。

24.6.10. RTC_TSTR（RTC 时间戳时间寄存器）

该寄存器的内容仅在 RTC_ISR 的 TSF 为 1 时有效。其在 TSF 位复位时清零。
地址偏移：0x30
RTC 域复位值：0x0000 0000
系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-	PM	HT[1:0]		HU[3:0]			
类型	RO-0	RO	RO	RO	RO	RO	RO	RO
15:8	-	MNT[2:0]			MNU[3:0]			
类型	RO-0	RO	RO	RO	RO	RO	RO	RO
7:0	-	ST[2:0]			SU[3:0]			
类型	RO-0	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:23	NA	保留位，未定义
22	PM	AM/PM 标记 0：AM 或 24 小时制 1：PM
21:20	HT[1:0]	小时的十位数（BCD 格式）
19:16	HU[3:0]	小时的个位数（BCD 格式）
15	NA	保留位，未定义
14:12	MNT[2:0]	分的十位数（BCD 格式）
11:8	MNU[3:0]	分的个位数（BCD 格式）
7	NA	保留位，未定义
6:4	ST[2:0]	秒的十位数（BCD 格式）
3:0	SU[3:0]	秒的个位数（BCD 格式）

24.6.11. RTC_TSDR（RTC 时间戳日期寄存器）

该寄存器的内容仅在 RTC_ISR 的 TSF 为 1 时有效。其在 TSF 位复位时清零。
地址偏移：0x34
RTC 域复位值：0x0000 0000

系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	WDU[2:0]			MT	MU[3:0]			
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	-		DT[1:0]		DU[3:0]			
类型	RO-0	RO-0	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位，未定义
15:13	WDU[2:0]	星期几
12	MT	月的十位数（BCD 格式）
11:8	MU[3:0]	月的个位数（BCD 格式）
7:6	NA	保留位，未定义
5:4	DT[1:0]	日的十位数（BCD 格式）
3:0	DU[3:0]	日的个位数（BCD 格式）

24.6.12. RTC_TSSSR（RTC 时间戳亚秒寄存器）

该寄存器的内容仅在 RTC_ISR 的 TSF 为 1 时有效。其在 TSF 位复位时清零。

地址偏移：0x38

RTC 域复位值：0x0000 0000

系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	SS[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	SS[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	SS[15:0]	亚秒数值 当发生时间戳事件时，SS[15:0]为同步预分频计数器的数值。

24.6.13. RTC_CALR（RTC 校准寄存器）

该寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。

地址偏移：0x3C

RTC 域复位值：0x0000 0000

系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	CALP	CALW8	CALW16	-				CALM[8]
类型	RW	RW	RW	RO-0	RO-0	RO-0	RO-0	RW
7:0	CALM[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	CALP	RTC 频率增加 488.5 ppm 0：没有增加 RTCCLK 脉冲 1：每 2^{11} 脉冲有效地插入一个 RTCCLK 脉冲（频率增加 488.5 ppm） 这个功能旨在与 CALM 一起使用，因为 CALM 以精细的分辨率降低了日历频率。如果输入频率是 32768 Hz，则在 32 秒窗口内添加的 RTCCLK 脉冲数计算如下：(512 * CALP) - CALM。 请参阅章节 24.3.11：RTC 平滑数字校准。
14	CALW8	使用 8 秒校准循环周期 当 CALW8 被设置为 1 时，8 秒校准循环周期将被选中。 注：当 CALW8=1 时，CALM[1:0]被锁定在“00”。请参阅章节 24.3.11：RTC 平滑数字校准。
13	CALW16	使用 16 秒校准循环周期 当 CALW16 被设置为 1 时，16 秒校准循环周期将被选中。当 CALW8=1 时不可以设置该位。 注：当 CALW16=1 时，CALM[0]被锁定在“0”。请参阅章节 24.3.11：RTC 平滑数字校准。
12:9	NA	保留位，未定义
8:0	CALM[8:0]	校准负数 通过屏蔽 2^{20} 个 RTCCLK 脉冲（如果输入频率为 32768 Hz，则为 32 秒）中的 CALM 个脉冲来降低日历的频率。这减少了日历的频率，其分辨率为 0.9537 ppm。 要增加日历的频率的话，应该与 CALP 结合使用。参阅章节 24.3.11：RTC 平滑数字校准。

24.6.14. RTC_TAFCR（RTC 篡改和复用功能配置寄存器）

地址偏移：0x40

RTC 域复位值：0x0000 0000

系统复位：没影响

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	-							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	PC15MODE	PC15VALUE	PC14MODE	PC14VALUE	PC13MODE	PC13VALUE	-	
类型	RW	RW	RW	RW	RW	RW	RO-0	RO-0
15:8	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]		
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	TAMPTS	-		TAMP2TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E
类型	RW	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	NA	保留位，未定义
23	PC15MODE	PC15 模式 0：PC15 由 GPIO 配置寄存器控制。因此 PC15 在 Standby 模式下悬空。 1：如果 LSE 被禁用，PC15 将被强制为推挽输出。
22	PC15VALUE	PC15 数值 如果 LSE 被禁用并且 PC15MODE=1，PC15VALUE 配置 PC15 输出数据。
21	PC14MODE	PC14 模式 0：PC14 由 GPIO 配置寄存器控制。因此 PC14 在 Standby 模式下悬空。 1：如果 LSE 被禁用，PC14 将被强制为推挽输出。
20	PC14VALUE	PC14 数值 如果 LSE 被禁用并且 PC14MODE=1，PC14VALUE 配置 PC14 输出数据。
19	PC13MODE	PC13 模式 0：PC13 由 GPIO 配置寄存器控制。因此 PC13 在 Standby 模式下悬空。 1：如果所有 RTC 复用功能被禁用，PC13 将被强制为推挽输出。
18	PC13VALUE	RTC_ALARM 输出类型 / PC13 数值 如果 PC13 被用于输出 RTC_ALARM，则 PC13VALUE 配置其输出配置： 0：RTC_ALARM 为开漏输出 1：RTC_ALARM 为推挽输出 如果所有 RTC 复用功能被禁用并且 PC13MODE=1，则 PC13VALUE 配置 PC13 输出数据。
17:16	NA	保留位，未定义
15	TAMPPUDIS	RTC_TAMPx 上拉禁用位

		该位决定每一个 RTC_TAMPx 引脚在每次采样之前是否被预充电。 0：在采样之前对 RTC_TAMPx 引脚进行预充电（使能内部上拉） 1：禁用 RTC_TAMPx 引脚的预充电功能
14:13	TAMPPRCH[1:0]	RTC_TAMPx 预充电持续时间 这些位决定了在每次采样前上拉/激活的持续时间。TAMPPRCH 对于每个 RTC_TAMPx 输入都是有效的。 0x0：1 个 RTCCLK 周期 0x1：2 个 RTCCLK 周期 0x2：4 个 RTCCLK 周期 0x3：8 个 RTCCLK 周期
12:11	TAMPFLT[1:0]	RTC_TAMPx 滤波计数 这些位决定了激活篡改事件所需要的连续采样到指定电平（TAMPxTRG）的数量。TAMPFLT 对于每个 RTC_TAMPx 输入都是有效的。 0x0：篡改事件在 RTC_TAMPx 输入转换的边沿被激活（RTC_TAMPx 输入没有内部上拉） 0x1：篡改事件在有效电平的 2 次连续采样之后被激活。 0x2：篡改事件在有效电平的 4 次连续采样之后被激活。 0x3：篡改事件在有效电平的 8 次连续采样之后被激活。
10:8	TAMPFREQ[2:0]	篡改采样频率 决定采样每一个 RTC_TAMPx 输入的频率。 0x0：RTCCLK / 32768（当 RTCCLK=32768 Hz 时为 1 Hz） 0x1：RTCCLK / 16384（当 RTCCLK=32768 Hz 时为 2 Hz） 0x2：RTCCLK / 8192（当 RTCCLK=32768 Hz 时为 4 Hz） 0x3：RTCCLK / 4096（当 RTCCLK=32768 Hz 时为 8 Hz） 0x4：RTCCLK / 2048（当 RTCCLK=32768 Hz 时为 16 Hz） 0x5：RTCCLK / 1024（当 RTCCLK=32768 Hz 时为 32 Hz） 0x6：RTCCLK / 512（当 RTCCLK=32768 Hz 时为 64 Hz） 0x7：RTCCLK / 256（当 RTCCLK=32768 Hz 时为 128 Hz）
7	TAMPTS	激活篡改检测事件的时间戳功能 0：篡改检测事件不会导致时间戳被保存 1：在篡改检测事件发生时保存时间戳
6:5	NA	保留位，未定义
4	TAMP2TRG	RTC_TAMP2 输入的有效电平 如果 TAMPFLT != 00： 0：RTC_TAMP2 输入保持低电平将会触发篡改检测事件。 1：RTC_TAMP2 输入保持高电平将会触发篡改检测事件。 如果 TAMPFLT == 00： 0：RTC_TAMP2 输入的上升沿将会触发篡改检测事件。 1：RTC_TAMP2 输入的下降沿将会触发篡改检测事件。
3	TAMP2E	RTC_TAMP2 输入检测使能 0：禁用 RTC_TAMP2 检测 1：使能 RTC_TAMP2 检测
2	TAMPIE	篡改中断使能

		0：禁用篡改中断 1：使能篡改中断
1	TAMP1TRG	RTC_TAMP1 输入的有效电平 如果 TAMPFLT != 00： 0：RTC_TAMP1 输入保持低电平将会触发篡改检测事件。 1：RTC_TAMP1 输入保持高电平将会触发篡改检测事件。 如果 TAMPFLT == 00： 0：RTC_TAMP1 输入的上升沿将会触发篡改检测事件。 1：RTC_TAMP1 输入的下降沿将会触发篡改检测事件。
0	TAMP1E	RTC_TAMP1 输入检测使能 0：禁用 RTC_TAMP1 检测 1：使能 RTC_TAMP1 检测

注意：当 TAMPFLT=0 时，TAMPxE 必须在 TAMPxTRG 发生改变时被复位，以避免虚假的 TAMPxF 置位。

24.6.15. RTC_ALRMASSR（RTC 闹钟 A 亚秒寄存器）

该寄存器仅在 RTC_ISR 的 ALRAWF 设置为 1 或在初始化模式下可写。
该寄存器是写保护的。写访问步骤描述在“RTC 寄存器写保护”。
地址偏移：0x44
RTC 域复位值：0x0000 0000
系统复位：没影响

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

Bit	Name	Function
31:28	NA	保留位，未定义
27:24	MASKSS[3:0]	屏蔽从该位开始的高位 0：没有比较闹钟 A 的亚秒。闹钟在秒钟单位增加时设置（假设其余字段是匹配的） 1：SS[14:1]在闹钟 A 比较中是不关心的。仅比较 SS[0]。 2：SS[14:2]在闹钟 A 比较中是不关心的。仅比较 SS[1:0]。 3：SS[14:3]在闹钟 A 比较中是不关心的。仅比较 SS[2:0]。 ... 12：SS[14:12]在闹钟 A 比较中是不关心的。仅比较 SS[11:0]。

		13：SS[14:13]在闹钟 A 比较中是不关心的。仅比较 SS[12:0]。 14：SS[14]在闹钟 A 比较中是不关心的。仅比较 SS[13:0]。 15：所有 15 个 SS 位都要比较，并且必须匹配才能激活闹钟。 同步计数器（15 位）的溢出位永远不会被比较。该位只有在移位操作之后才会不为 0。
23:15	NA	保留位，未定义
14:0	SS[14:0]	亚秒数值 该数值用于跟同步预分频计数器的内容进行比较，以决定闹钟 A 是否要被激活。仅第 0 位到第 MASKSS-1 位会被比较。

25. I2C 接口

25.1. 主要特性

- 支持主机和从机模式
- 多主机兼容性
- 标准模式（高达 100kHz）
- 快速模式（高达 400kHz）
- 快速+模式（高达 1MHz）
- 支持 7 比特地址和 10 比特地址
- 多重 7 比特地址模式（第二个地址支持配置掩码）
- 支持广播模式
- 可编程的建立时间和保持时间
- 支持时钟拉低扩展
- 支持软件复位模块
- 单个数据存储区，兼容 DMA 模式
- 可编程的数据滤波器和模拟滤波器
- 兼容 SMBus 2.0 版本
  ——硬件 PEC 生成与校验
  ——命令和数据 ACK 控制
  ——ARP 地址解析协议支持
  ——主机和设备支持
  ——SMBus 警告支持
  ——超时和空闲帧检测
  ——兼容 PMBus1.1 版本
  ——多时钟域支持，模块的工作时钟与 PCLK 时钟独立

25.2. 功能描述

除了接收和发送数据，这个接口从串行转换成并行格式，以及其逆过程。通过软件启用或禁用中断。该接口通过数据引脚（SDA）和时钟引脚（SCL）连接到 I2C 总线。它可以连接标准（高达 100kHz），快速模式（高达 400kHz）或快速+模式（高达 1MHz）I2C 总线。

此接口也可以通过数据引脚（SDA）和时钟引脚（SCL）连接到的 SMBus。如果有 SMBus 功能支持：附加的 SMBus ALERT 引脚（SMBA）也是可用的。

25.2.1. 功能比对

本文描述了 I2C1 所实现的全套功能。I2C2 功能稍有裁剪，但有的功能都和 I2C1 一样；差异如下表所示。

表 25.1 I2C 功能实现比对

I2C 特性	I2C1	I2C2
7 比特地址模式	√	√
10 比特地址模式	√	√
标准模式	√	√
快速模式	√	√
多时钟域	√	—
SMBus 模式	√	—
快速模式+20mA 驱动能力	√	—

25.2.2. 结构框图

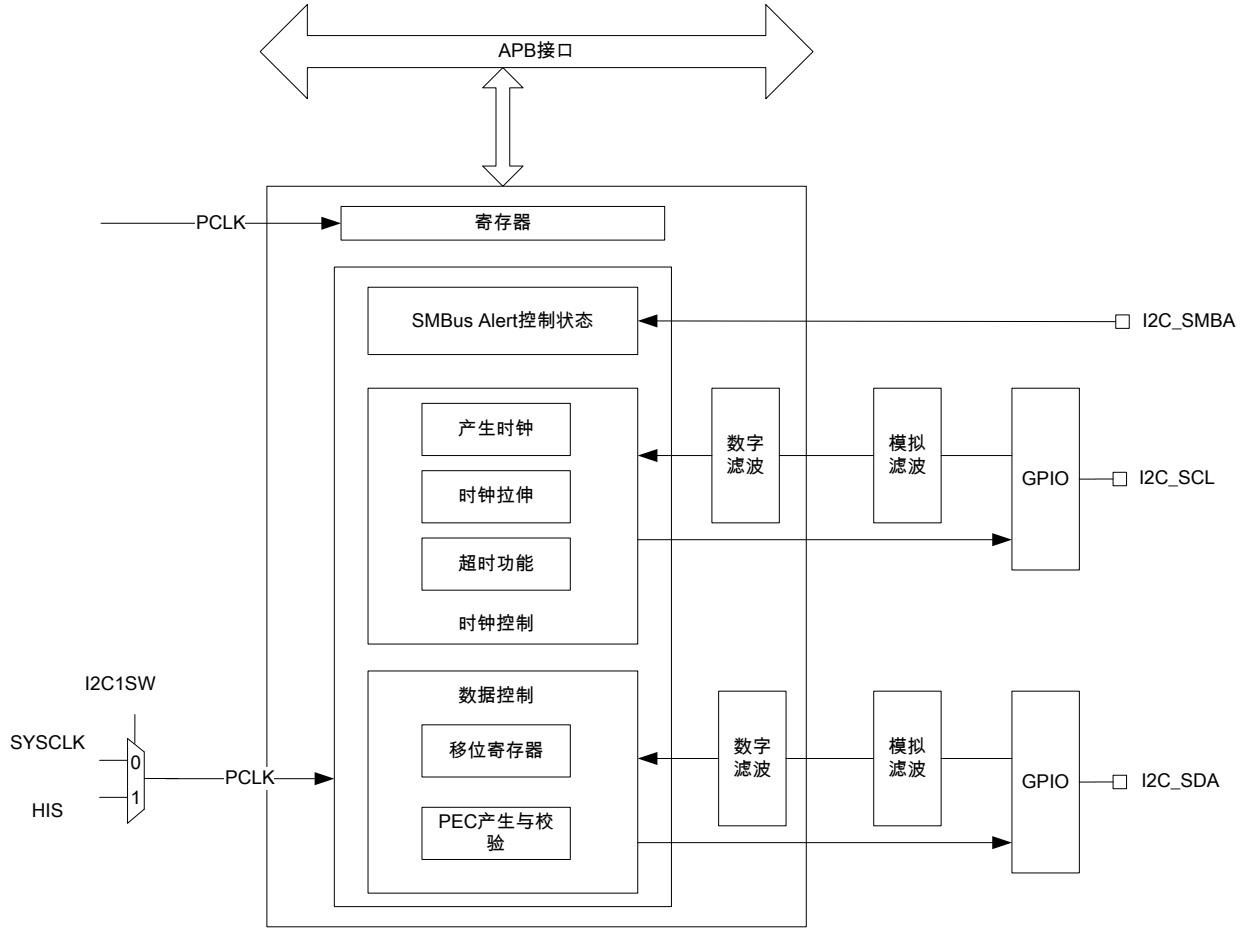


图 25.1 I2C 实现结构框图

I2C 由一个独立的时钟源驱动，它允许 I2C 相对于 PCLK 频率独立运作。这个独立的时钟源可以选择以下两个时钟源之一：

HSI：高速内部振荡器

SYSCLK：系统时钟

I2C 的 I/Os 支持 20mA 的输出电流驱动以适应超快速模式的操作。通过将 SCL 和 SDA 的驱动能力控制位置 1

来启用此设置，详见 SYSCFG 配置（SYSCFG_CFGR1）。

25.2.3. 时钟要求

I2C 内核由 I2CCLK 提供时钟。

I2CCLK 的时钟周期 T_{I2CCLK} 必须满足下列条件：
$T_{I2CCLK}<(T_{LOW}-T_{filters})/4$ and $T_{I2CCLK}<T_{HIGH}$
其中：
T_{LOW}:SCL 低电平时间和 tHIGH:SCL 高电平时间
$T_{filters}$:启用时，模拟滤波器和数字滤波器所带来的延迟的总和。模拟滤波器的延迟最大是 200 纳秒。数字滤波器延时是 DNF * T_{I2CCLK}。

PCLK 的时钟周期 T_{PCLK} 必须满足下列条件:
$T_{PCLK}<4/3T_{SCL}$
其中 T_{SCL}:SCL 周期
注意：当 I2C 内核时钟由 PLCK 主频提供时，PCLK 必须满足 T_{I2CCLK} 的限制条件。

25.2.4. 模式选择

接口可以工作在以下四个模式之一：
- 从机发送
- 从机接收
- 主机发送
- 主机接收

默认情况下，它在从机模式下运行。接口在生成起始条件后自动地从从模式切换到主模式；当仲裁丢失或产生停止信号时，则从主模式切换到从模式。允许多主机功能。

通讯流
在主控模式下，I2C 接口启动数据传输，并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。起始条件和停止条件都是在主模式下由软件控制产生。

从模式时，I2C 接口能识别它自己的地址(7 位或 10 位)和广播呼叫地址。软件能够控制开启或禁止广播呼叫地址的识别。保留的 SMBus 地址，也可以由软件启用。

数据和地址按 8 位/字节进行传输，高位在前。跟在起始条件后的 1 或 2 个字节是地址(7 位模式为 1 个字节，10 位模式为 2 个字节)。地址只在主模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位(ACK)给发送器。如下图 25.2 所示。

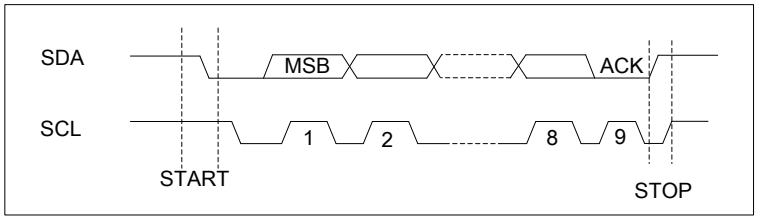


图 25.2 I2C 总线协议

软件可以开启或禁止应答(ACK)，并可以设置 I2C 接口的地址(7 位、10 位地址或广播呼叫地址)。

25.2.5. I2C 初始化

启用和禁用外设

I2C 外设时钟必须在时钟控制器中配置并启用。然后 ,I2C 可以通过设置在 I2Cx_CR1 寄存器的 PE 位来使能。当 I2C 被禁用（PE=0）的时候，I2C 执行软件复位：I2C 线（SDA 和 SCL）都被释放。内部状态机复位，所有的通信控制位和状态位回到它们的复位值。

噪声滤波器

如有必要，在通过设置在 I2Cx_CR1 寄存器的 PE 位，启用 I2C 外设之前，你必须先配置噪声滤波器。默认情况下，模拟噪声滤波器会处理 SDA 和 SCL 输入。这个模拟滤波器符合快速模式和快速模式 Plus I2C 规范的需要，来抑制 50 纳秒以内的尖峰脉冲宽度。可以通过设置的 ANFOFF 位来禁用这个模拟滤波器和/或配置 I2Cx_CR1 寄存器的 DNF[3:0]位选择数字滤波器。

当数字滤波功能选择使能时，SCL 和 SDA 总线保持的某种状态大于 DNF*I2CCLK 时钟周期数时，内部的信号状态才会发生改变；这用于对 SDA 和 SCL 进行滤波，滤波的长度是 1-15 个 I2CCLK 周期。

表 25.2 模拟滤波和数字滤波的比较

	模拟滤波	数字滤波
滤波宽度	大于等于 50ns	可编程 1-15 个工作时钟周期
滤波实现的好处	睡眠可以工作	可编程的滤波长度
滤波实现的坏处	滤波长度随着工艺电压温度改变	—

注意：当 I2C 启用时不允许更改过滤器的配置。

I2C 时序

在主从模式中必须配置时序，以保证正确的数据保持和建立时间。这通过编程 I2C_TIMINGR 寄存器中的 PRESC[3:0]，SCLDEL[3:0]和 SDADEL[3:0]位来实现。

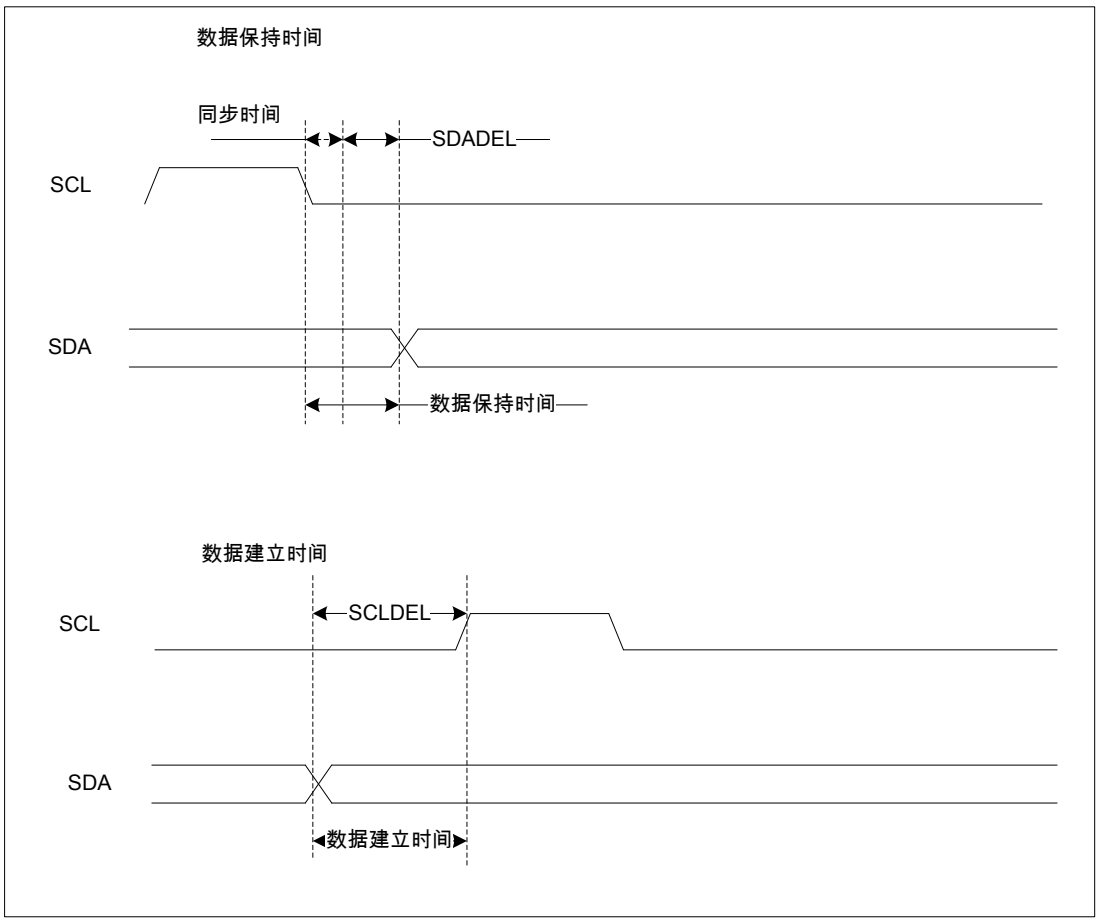


图 25.3 建立时间和保持时间

● 当内部检测 SCL 下降沿，在发送 SDA 输出前会插入一个延时。这个延迟是

T_{SDADEL}=SDADEL*T_{PRESC}，T_{PRESC}=（PRESC+1）*T_{I2C_CLK}

T_{SDADEL} 受保持时间的影响

总的 SDA 输出延迟是：

$T_{同步}$+[SDADEL*(PRESC+1)*T_{I2C_CLK}]

$T_{同步}$持续时间由下列参数决定：

- SCL 下降斜率
- 启用时，模拟滤波器所带来的输入延迟：至少为 50ns，最大 200ns
- 启用时，数字滤波器所带来的输入延迟：T_{DNF}=DNF*T_{I2C_CLK}
- 由于 SCL 的同步 I2C_CLK 时钟（2 至 3 个 I2C_CLK 周期）造成的延迟

为了弥补 SCL 下降沿的未定义区域，你必须按下列方式，对 SDADEL 编程：

{T_{fmax}+$T_{最小数据保持时间}$-50ns-[(DNF+3)*T_{I2C_CLK}]}/{(PRESC+1)*T_{I2C_CLK}}<=SDADEL

SDADEL<={$T_{最大数据保持时间}$-200ns-[(DNF+4)*T_{I2C_CLK}]}/{(PRESC+1)*T_{I2C_CLK}}

注：-50ns/-200ns 只有在启用了模拟滤波器的时候才是公式的一部分。

● 在 SDA 发送输出后，在建立时间中，SCL 线保持在低电平。这个建立时间是

T_{SCLDEL}=(SCLDEL+1)*T_{PRESC} T_{PRESC}=(PRESC+1)*T_{I2C_CLK}

T_{SCLDEL} 受建立时间的影响

为了弥补 SDA 发送（上升沿通常比较糟时）带来的的未定义区域，你必须按下列方式，对 SCLDEL 编程：

{[$T_{R\ 最大}$+$T_{\ 最小建立时间}$]/[(PRESC+1)]*T_{I2C_CLK}]}-1<=SCLDEL

表 25.2 I2C-SMBUS 建立时间和保持时间定义

符号	表示的意义	标准模式		快速模式		快速+模式		SMBUS 模式		单位
		最小	最大	最小	最大	最小	最大	最小	最大	
$T_{HD:DAT}$	数据保持时间	0	-	0	-	0	-	0.3	-	μs
$T_{VD:DAT}$	数据有效时间	-	3.45	-	0.9	-	0.45	-	-	μs
$T_{SU:DAT}$	数据建立时间	250	-	100	-	50	-	250	-	ns
T_R	SCL 和 SDA 的上升时间	-	1000	-	300	-	120	-	1000	ns
T_F	SCL 和 SDA 的下降时间	-	300	-	300	-	120	-	300	ns

在主模式下，SCL 时钟高和低电平，必须通过编程 I2Cx_TIMINGR 寄存器的 PRESC[3:0]、SCLH 的[7:0]和 SCLL[7:0]位域来配置。

- 当内部检测 SCL 下降沿，在释放 SCL 输出之前会插入一个延时。这个延迟是 T_{SCLL}=(SCLL+1)*T_{PRESC} 其中 T_{PRESC}=(PRESC+1)*T_{I2C_CLK}。T_{SCLL} 受 SCL 低时间 T_{LOW} 的影响。
- 当内部检测 SCL 上升沿，在拉低 SCL 输出之前会插入一个延时。这个延迟是 T_{SCLH}=(SCLH+1)*T_{PRESC} 其中 T_{PRESC}=(PRESC+1)*T_{I2C_CLK}。T_{SCLH} 受 SCL 高时间 T_{HIGH} 的影响。

注意：当 I2C 启用时不允许更改时序配置。

I2C 配置

I2C 从机 NOSTRETCH 模式必须在在使能 I2C 外设之前配置

注意：当 I2C 启用时不允许更改 NOSTRETCH 配置。

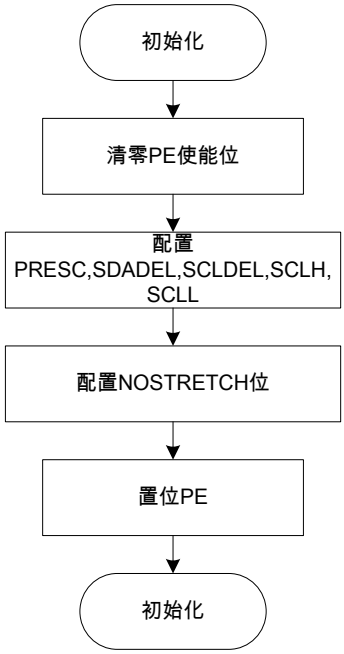


图 25.4 I2C 初始化流程

25.2.6. 软件复位

将 I2Cx_CR1 寄存器的 PE 位置 0，可以实现软件复位。在这种情况下，SCL 和 SDAI2C 线被释放。内部状态机复位，所有的通信控制位和状态位回到它们的复位值。而配置寄存器不会受到任何影响。

这是受影响的寄存器位的列表：

1. I2Cx_CR2 寄存器：START,STOP,NACK
2. I2Cx_ISR 寄存器：BSY,TXE,TXIS,RXNE,ADDR,NACKF,TCR,TC,STOPF,BERR,ARLO,OVR

另外还有支持 SMBus 功能时：

1. I2Cx_CR2 寄存器：PECBYTE
2. I2Cx_ISR 寄存器：PECERR,TIMEOUT,ALERT

PE 清零后只少要保持三个 APB 时钟周期，以便对接口进行复位；执行复位的正确顺序应该是先写 PE=0，然后读取 PE=0，最后后写 PE=1。

25.2.7. 数据传输

要通过发送和接收数据寄存器和一个移位寄存器来管理数据发送。

接收

SDA 输入会填充移位寄存器。在第 8 个 SCL 脉冲（当收到完整的数据字节）之后，如果 I2Cx_RXDR 寄存器是空的（RXNE=0）移位寄存器的内容会被复制到 I2Cx_RXDR 寄存器。如果 RXNE=1，也就是说，尚未读取之前收到的数据字节，这时 SCL 线被拉低直到 I2Cx_RXDR 被读取。这个拉伸被插入到第 8 和第 9 个 SCL 脉冲之间（应答脉冲之前）。

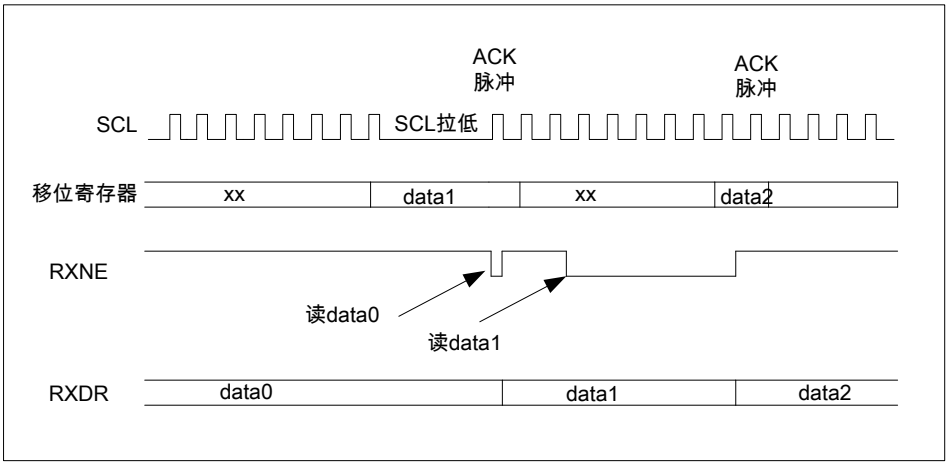


图 25.5 数据接收时序图

发送

如果 I2Cx_TXDR 寄存器不为空（的 TXE=0），其内容将在第 9 个 SCL 脉冲（应答脉冲）之后被复制到移位寄存器。然后移位寄存器的内容被移到 SDA 线上。如果 TXE=1，也就是说，I2Cx_TXDR 中没有数据被写入，SCL 线会被拉长为低直到 I2Cx_TXDR 有新数据被写入。在第 9 个 SCL 脉冲之后完成拉伸。

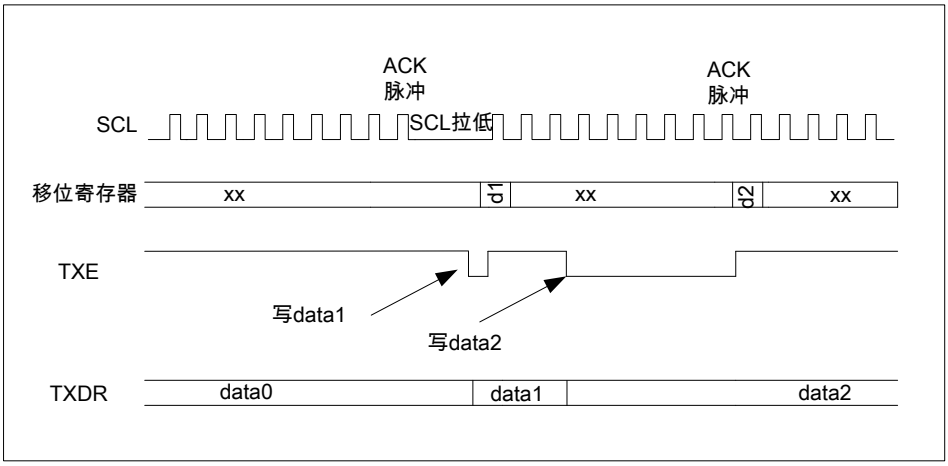


图 25.6 数据发送时序图

硬件管理

I2C 有一个内嵌的字节计数器，可以管理发送的字节数，以便在各种模式中关闭通讯：

- 主模式下生成 NACK、STOP 和 ReSTART
- 从机接收模式下的 ACK 控制
- SMBus 的功能支持时的 PEC 的生成/检查

字节计数器总是在主模式下使用。默认情况下，它在从模式下被禁用，但它可以通过软件设置 I2Cx_CR2 寄存器的 SBC 位（从字节控制）来启用。

要传输的字节数编程在 I2Cx_CR2 寄存器的 NBYTES[7:0]位域。如果要传输的字节数(NBYTES 个)大于 255，或一个接收器要控制收到的数据字节的应答值，就必须设置 I2Cx_CR2 寄存器的 RELOAD 位来选择重载模式。

在这种模式下，当 NBYTES 中编程的字节个数被发送完毕后，TCR 标志被置 1，如果 TCIE 为 1，则会产生一个中断。SCL 在 TCR 标志为 1 期间被拉伸。向 NBYTES 写入一个非零的值时，TCR 标志由软件清除。

当 NBYTES 被重载为上次的字节数时，RELOAD 位必须被清零。

当主模式下 RELOAD=0，计数器可以按下列 2 种模式工作：

- 自动结束模式(I2Cx_CR2 寄存器的 AUTOEND=1)。在这种模式下，一旦发送字节数达到了 NBYTES[7:0]位域中设置的字节数，主机会自动发送一个停止条件。
- 软件结束模式（I2Cx_CR2 寄存器的 AUTOEND=0)。在这种模式下，发送字节数达到了 NBYTES[7:0]位域中设置的字节数后需要软件的干预，这时 TC 标志会被置 1，如果 TCIE 位为 1，还会产生中断。在 TC 标志为 1 期间，SCL 信号被拉伸。在 I2Cx_CR2 寄存器的 START 或 STOP 位被置 1 时，TC 标志由软件清除。主机要发送一个 RESTART 条件时，必须使用此模式。

注意: AUTOEND 位在 RELOAD 位被置 1 时没有效果。

表 25.3 I2C 配置表

功能	SBC 位	RELOAD 位	AUTOEND 位
主机发送/接收 NBYTES+停止位	x	0	1
主机发送/接收 NBYTES+RESTART	x	0	0
从机发送/接收+ACK 回复	0	x	x
从机接收+ACK 回复控制	1	1	x

25.2.8. I2C 从机模式

I2C 从机初始化

要工作在从机模式下，您必须启用至少一个从机地址。两个寄存器 I2Cx_OAR1 和 I2Cx_OAR2 都可以用来写入从机的本机地址 OA1 和 OA2。

- 通过设置 I2Cx_OAR1 寄存器的 OA1MODE 位，可将 OA1 配置在 7 位模式（默认）或 10 位地址模式。通过设置 I2Cx_OAR1 寄存器的 OA1EN 位来启用 OA1。
- 如果需要额外的从机地址，你可以配置第二个从机地址 OA2。配置 I2Cx_OAR2 寄存器的 OA2MSK[2:0]位位域，可以最多屏蔽 OA2 的低 7 位。因此从 1 到 6 为 OA2MSK 配置，只有 OA2[7:2]，OA2[7:3]，OA2[7:4]，OA2[7:5]，OA2[7:6]或 OA2[7]参加与接收到的地址的比较操作。只要 OA2MSK 不等于 0，被 OA2 地址比较排除的 I2C 地址（0000xxx 和 1111xxx）不会被应答。如果 OA2MSK=7，收到的所有 7 位地址（保留地址除外）都会被应答。OA2 始终是一个 7 位地址，不可以 10 位。如果他们启用特定的使能位，这些保留地址也可以应答，就是如果他们被写在 I2Cx_OAR1 或 I2Cx_OAR2 寄存器中并且 OA2MSK=0。通过设置 I2Cx_OAR2 寄存器的 OA2EN 位来启用 OA2。
- I2Cx_CR1 寄存器的 GCEN 位被置 1 时，呼叫地址被启用。

当 I2C 由启用了的地址选中时，ADDR 中断状态标志被置 1，如果 ADDRIE 位为 1，就会产生一个中断。

默认情况下，从机使用它的时钟延长的功能，这意味着，它可根据需要在 SCL 信号拉低，以便执行软件动作。如果主机不支持时钟延长，I2C 的 I2Cx_CR1 寄存器的 NOSTRETCH 必须配置为 1。

收到地址中断后，如果启用了多个地址，你必须读 I2Cx_ISR 寄存器中的 ADDCODE[6:0]位以检查是哪一个地址匹配上了。还要检查 DIR 标志，以了解数据传输的方向。

从机时钟延长（NOSTRETCH=0）

在默认模式下，I2C 从机于下列情况下拉低 SCL 时钟：

- 当地址标志被置位：接收到的地址和启用了的从机地址之一匹配上。这种拖延在 ADDR 标志被软件清零后被释放。清零该位的办法是将 ADDRCF 位置 1。
- 在发送时，如果以前的数据传输完毕后，并没有新的数据被写到 I2Cx_TXDR 寄存器，或者如果在 ADDR 标志被清除（TXE=1）后还没有写一个字节。只要数据被写入 I2Cx_TXDR 寄存器，就会释放这个拉伸。
- 在接收时如果 I2Cx_RXDR 寄存器的内容还没有被读走，又有一个新的数据被收进来。这时延长在读取 I2Cx_RXDR 时被释放。
- 在从机字节控制模式，重载模式（SBC=1 和 RELOAD=1）时，如果 TCR=1，意味着最后一个数据字节已发送完毕。在向 NBYTES[7:0]位域写入一个非零值会清除 TCR 标志，这时延长也会释放。

从机时钟不延长（NOSTRETCH=1）

当 I2Cx_CR1 寄存器的 NOSTRETCH=1 时，I2C 从机不会延长 SCL 信号。

- 在 ADDR 标志置位的时候 SCL 时钟不会延长。
- 发送中，数据必须在对于发送的第一个 SCL 脉冲之前写入到 I2Cx_TXDR 寄存器。如果没有，会发生下溢，I2Cx_ISR 寄存器中的 UDR 标志被置位，如果 I2Cx_CR1 寄存器的 ERRIE 位为 1，会产生一个中断。如果首个数据发送已经开始而 STOPF 位还是 1（未被清掉）时，OVR 标志也会被设置。因此，如果你在写下次发送的首个发送数据之后才清除前一次传输的 STOPF 标志，必须先确认 OVR 状态，甚至于首个数据已经发送。
- 在接收时，必须在下一个字节的第 9 个 SCL 脉冲（ACK 脉冲）发生前，从 I2Cx_RXDR 寄存器将数据读走。如果没有，会发生溢出，I2Cx_ISR 寄存器中的 OVR 标志被置位，如果 I2Cx_CR1 寄存器的 ERRIE 位为 1，会产生一个中断。

从机字节控制模式

为了允许从机接收模式的字节 ACK 控制，从机字节控制模式必须通过设置 I2Cx_CR1 寄存器的 SBC 位来启用。这是与 SMBus 标准兼容所必需的。

必须选择重载模式，以便允许在从机接收模式下的字节 ACK 控制（RELOAD=1）。要获得每个字节的控制，必须在 ADDR 中断服务程序中将 NBYTES 初始化为 0x1，并在每接收一个字节后重新加载为 0x1。每当收到一个字节，TCR 位置 1，在第 8 和第 9 个 SCL 脉冲间，延长 SCL 信号为低。可以从 I2Cx_RXDR 寄存器读数据，然后决定要不要通过配置 I2Cx_CR2 寄存器的 ACK 位来发 ACK。对 NBYTES 写入一个非零值会释放 SCL 延长：ACK 或 NAK 被发送，然后可以接收下一字节。

可以写入一个比 0x1 更大的值到 NBYTES，在这种情况下，会连续接收 NBYTES 个数据。

注：1.SBC 位必须在 I2C 被禁用时配置，或当从机没有被寻址到时，或者当 ADDR=1 时。
2.RELOAD 位的值可以在 ADDR=1 或者当 TCR=1 的时候改变。
3.从机字节控制模式不兼容 NOSTRETCH 模式。不允许在 NOSTRETCH=1 的时候置位 SBC。

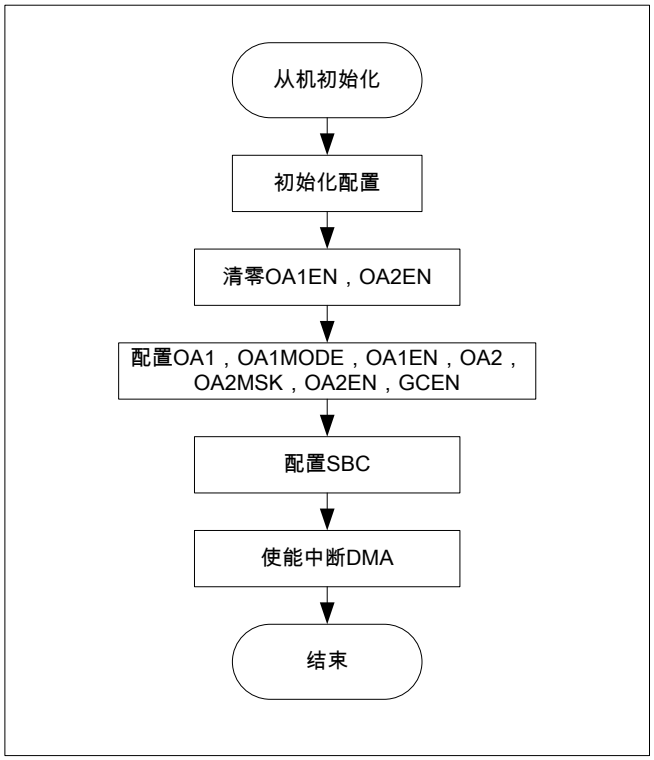


图 25.7 从机初始化流程

从机发送

当 I2Cx_TXDR 寄存器为空时会产生发送中断状态（TXIS）。如果 I2Cx_CR1 寄存器中的 TXIE 位为 1，则会产生中断。

向 I2Cx_TXDR 寄存器写入下一个发送数据时，TXIS 位被清除。

当收到一个 NACK，I2Cx_ISR 寄存器中的 NACKF 位置 1，如果 I2Cx_CR1 寄存器的 NACKIE 位为 1，会产生一个中断。从机会自动释放 SCL 和 SDA 线 ,这是为了允许主机执行停止或重新启动条件。在收到一个 NACK 时 TXIS 位不会被置 1。

当收到一个 STOP 的时候 I2Cx_ISR 寄存器中的 STOPF 标志会被置 1 ,如果这时 I2Cx_CR1 寄存器的 STOPIE 又为 1，就会产生一个中断。在大多数应用中，SBC 位通常被设定为 0。在这种情况下，在 TXE 为 0 时收到从机地址（ADDR=1），可以选择是将 I2Cx_TXDR 寄存器的内容当作第一个数据字节发送出去，还是将 TXE 位置 1，从而清空寄存器 I2Cx_TXDR 以便写一个新的数据字节进去。

在从机字节控制模式(SBC=1),要发送的字节数必须在地址匹配(ADDR=1)中断服务程序中写入到 NBYTES。在这种情况下，在传输过程中的 TXIS 事件个数与 NBYTES 中写入的值对应。

注意：当 NOSTRETCH=1，SCL 时钟在 ADDR 标志被置位的时候不延长，所以你不能够在 ADDR 子程序中清除 I2Cx_TXDR 寄存器的内容而定义第一个数据字节。要发送的第一个数据字节 ,必须预先写到 I2Cx_TXDR 寄存器：

- 这个数据可以是前一次发送消息的时候 TXIS 事件里写入的数据。
- 如果这个数据字节不是要发送的那个，I2Cx_TXDR 寄存器可以随着 TXE 位的置位而被清空，以便写入一

个新的数据字节。STOPF 位必须在这些动作之后被清除，以保证他们在第一个数据传输开始前就执行，跟着地址的 ACK。如果 STOPF 在第一个数据传输开始时仍然为 1 ,会产生下溢错误(OVR 标志被置 1)。如果需要 1 个 TXIS 事件 ,(发送中断或发送 DMA 请求), 除了置 TXE 位以外还要置 TXIS 位，这是为了产生 TXIS 的事件。

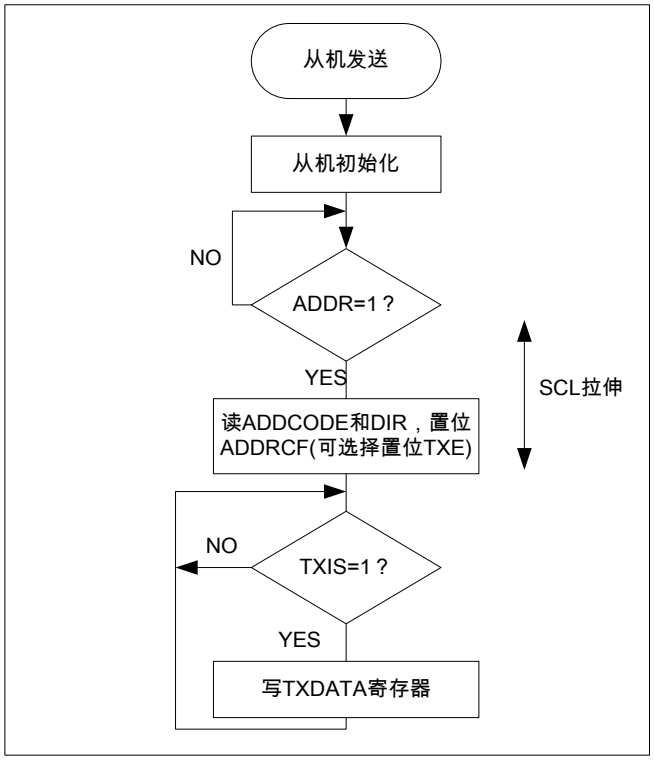


图 25.8 从机发送流程图(NOSTRETCH=0)

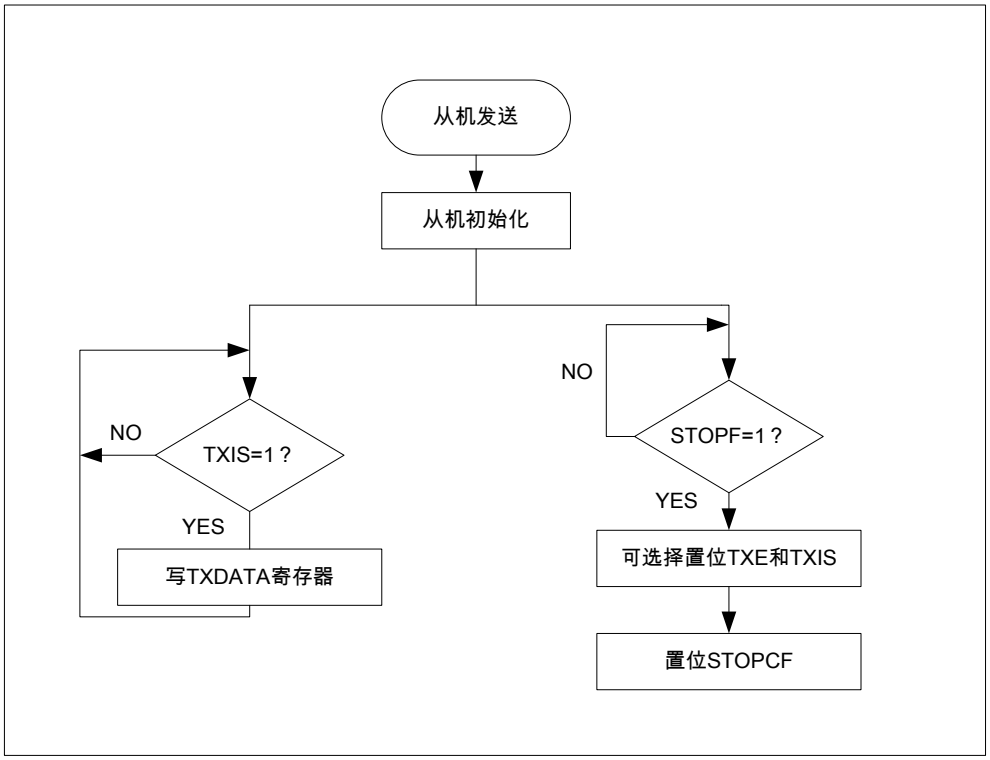


图 25.9 从机发送流程图(NOSTRETCH=1)

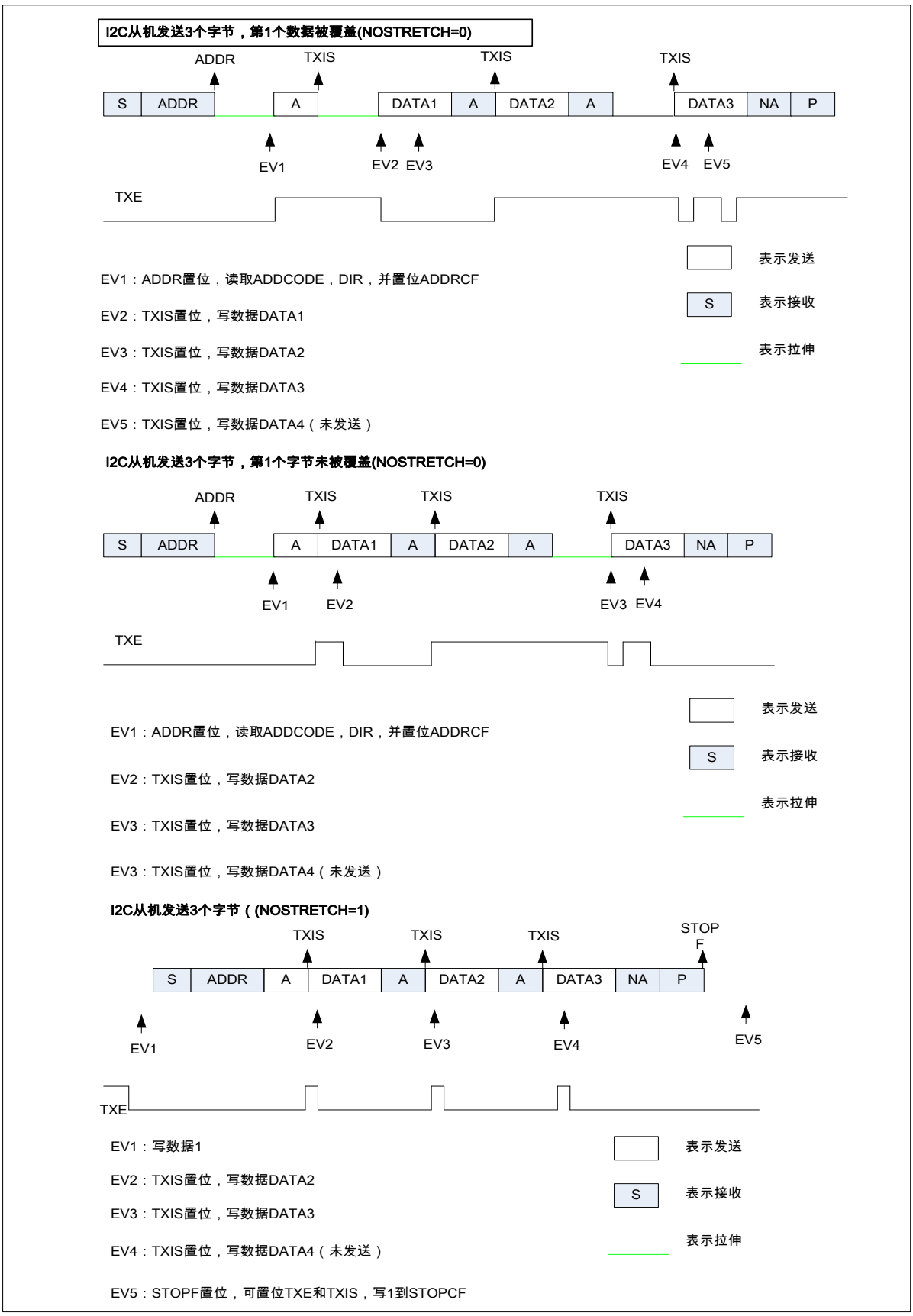


图 25.10 从机发送总线时序图

从机接收

在 I2Cx_RXDR 收满时 I2Cx_ISR 寄存器的 RXNE 被置 1，如果 I2Cx_CR1 寄存器的 RXIE 为 1，会产生一个中断。在读取 I2Cx_RXDR 时 RXNE 会被清除。

在收到一个 STOP 条件，并且 I2Cx_CR1 的 STOPIE 被置 1，I2Cx_ISR 寄存器的 STOPF 位会被置 1，并产生一个中断。

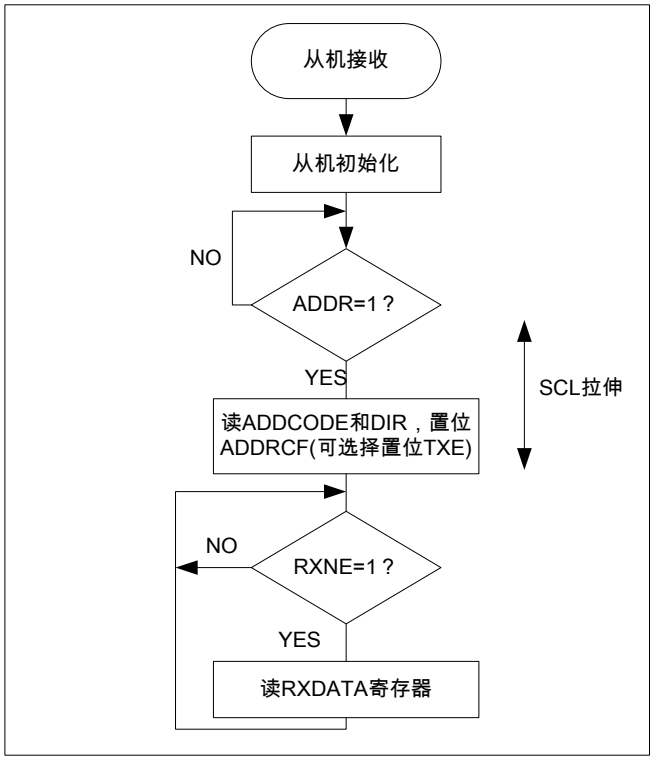


图 25.11 从机接收流程如（NOSTRETCH=0）

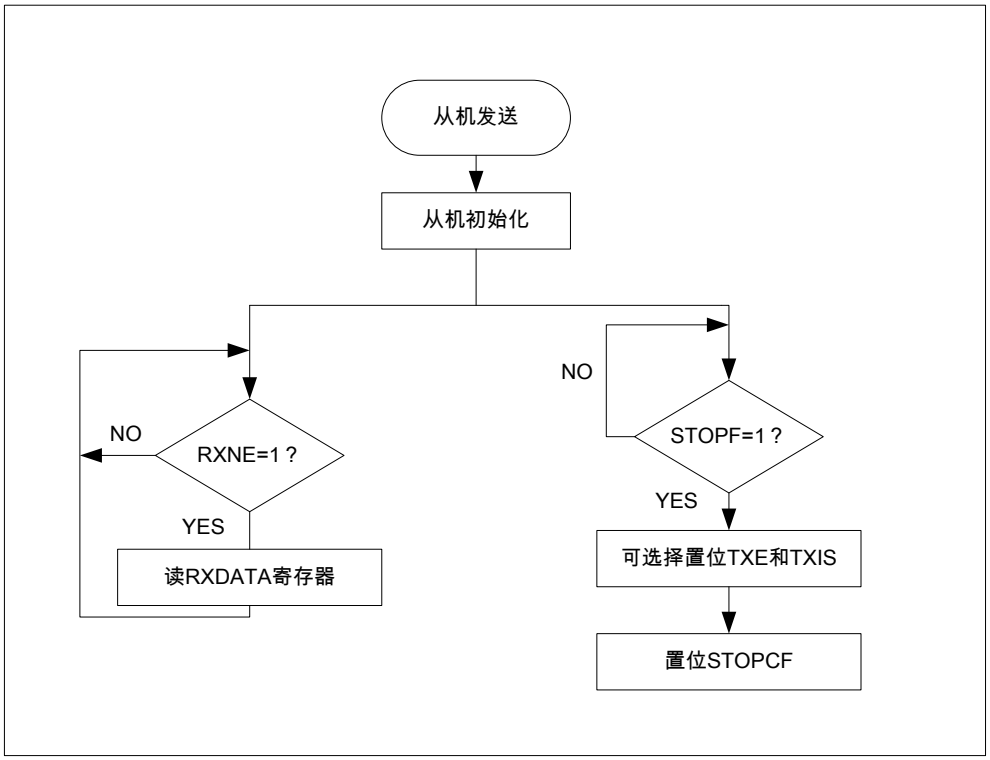


图 25.12 从机接收流程如（NOSTRETCH=1）

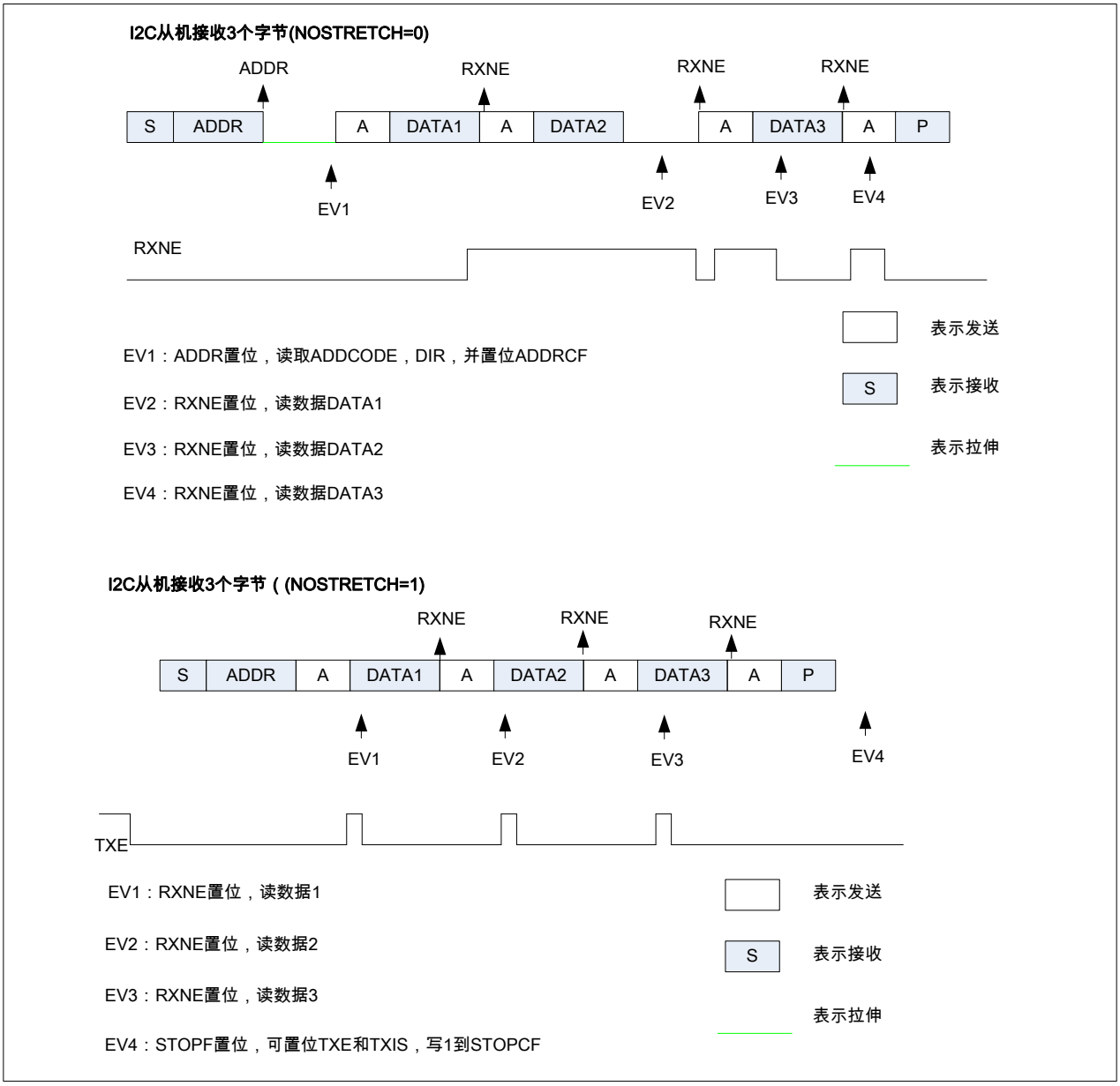


图 25.13 从机接收总线时序图

25.2.9. I2C 主机模式

主机模式初始化

在启动外设之前，必须设置 I2Cx_TIMINGR 寄存器的 SCLH 和 SCLL 的位来配置 I2C 主时钟。

这会实现一个时钟同步机制，以支持多主机环境和从机时钟延长。

为了让时钟同步：

- 从 SCL 低电平的内部检测开始用 SCLL 计数器来计数时钟低电平的个数。

- 从 SCL 高电平的内部检测开始用 SCLH 计数器来计数时钟高电平的个数。

I2C 依靠 SCL 下降沿的一个 $T_{同步}$延迟和 SCL 输入噪声滤波器（模拟+数字）来检测自己的 SCL 低电平，并将 SCL 同步到 I2Cx_CLK 时钟。一旦 SCLL 计数器达到了 I2Cx_TIMINGR 寄存器的 SCLL[7:0]位域中的编程值，I2C 释放 SCL 到高电平。

I2C 依靠 SCL 上升沿的一个 $T_{同步}$延迟和 SCL 输入噪声滤波器（模拟+数字）来检测自己的 SCL 高电平，并将 SCL 同步到 I2Cx_CLK 时钟。一旦 SCLH 计数器达到了 I2Cx_TIMINGR 寄存器的 SCLH[7:0]位域中的编程值，I2C 将 SCL 拉低到低电平。

因此，主机时钟周期是：

$T_{SCL} = T_{同步1} + T_{同步2} + \{ [(SCLH+1) + (SCLL+1)] \times (PRESC+1) * T_{I2C\_CLK} \}$,其中 $T_{同步1}$时间取决于以下参数：

- SCL 下降斜率
- 启用时，模拟滤波器所带来的输入延迟
- 启用时，数字滤波器所带来的输入延迟：DNF $*T_{I2C\_CLK}$
- 由于 SCL 的同步 I2C_CLK 时钟（2 至 3 个 I2C_CLK 周期）造成的延迟

$T_{同步2}$时间取决于以下参数：

- SCL 的上升斜率
- 启用时，模拟滤波器所带来的输入延迟
- 启用时，数字滤波器所带来的输入延迟：DNF $*T_{I2C\_CLK}$
- 由于 SCL 的同步 I2C_CLK 时钟（2 至 3 个 I2C_CLK 周期）造成的延迟

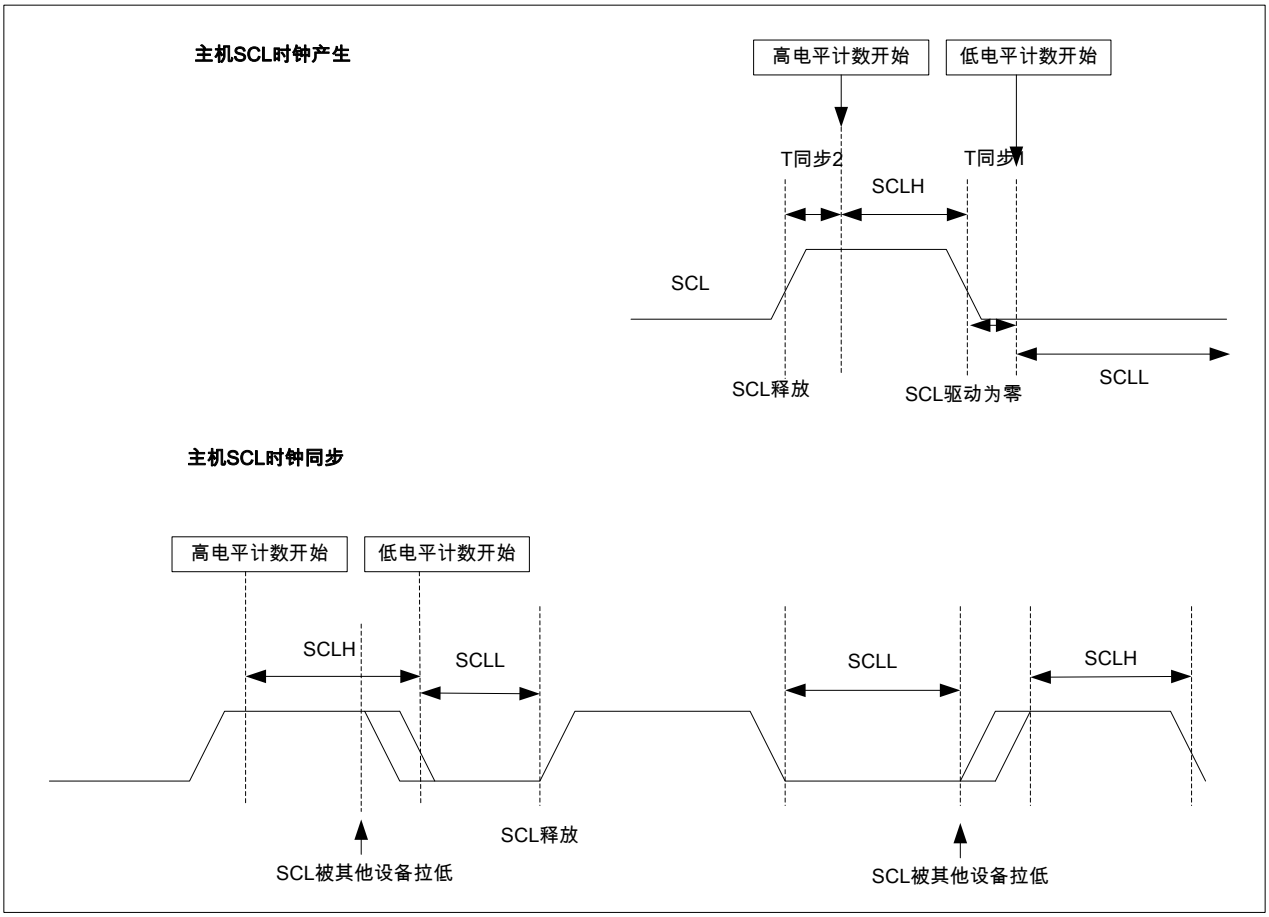


图 25.14 主机时钟的产生

注意：为了与 I2C 或 SMBus 兼容，主机时钟必须保证的时序如下：

表 25.4 I2C SMBUS 规范的时钟时序

符号	参数	标准模式		快速模式		快速+模式		SMBUS		单位
		最小	最大	最小	最大	最小	最大	最小	最大	
F_{SCL}	SCL 时钟频率	-	100	-	400	-	1000	-	100	kHz
$T_{HD:STA}$	START 保持时间	4.0	-	0.6	-	0.26	-	4.0	-	μs
$T_{SU:STA}$	RESTART 保持时间	4.7	-	0.6	-	0.26	-	4.7	-	μs
$T_{SU:STO}$	STOP 建立时间	4.0	-	0.6	-	0.26	-	4.0	-	μs
T_{BUF}	STOP 和 START 之间的空闲时间	4.7	-	1.3	-	0.5	-	4.7	-	μs
T_{LOW}	SCL 低电平时间	4.7	-	1.3	-	0.5	-	4.7	-	μs
T_{HIGH}	SCL 高电平时间	4.0	-	0.6	-	0.26	-	4.0	50	μs
T_R	SDA 和 SCL 的上上升时间	-	1000	-	300	-	120	-	1000	ns
T_F	SDA 和 SCL 的下	-	300	-	300	-	120	-	300	ns

	降时间									

注：SCLL 也被用来产生 T_{BUF} 和 $T_{SU:STA}$ 时序。
SCLH 也被用来产生 $T_{HD:STA}$ 和 $T_{SU:STO}$ 时序。

主机通信初始化（地址阶段）

为了启动通讯，必须在 I2Cx_CR2 寄存器中设置从机地址的下列参数：

- 地址模式（7 位或 10 位）：ADD10
- 要发送的从机地址：SADD[9:0]
- 传输方向：RD_WRN
- 在 10 位地址的情况下读取：HEAD10R 位。在必须发送完整的地址顺序时，HEAD10R 必须被先置位，以示在方向改变时仅需要帧头的区别。
- 要传输的字节数：NBYTES[7:0]位，如果字节数大于等于 255 个字节，NBYTES[7:0]最初必须使用 0xFF 填充。

随后必须设置 I2Cx_CR2 寄存器的 START 位。START 位一旦被置 1，就不再允许更改上述所有位。

只要检测到总线是空闲(BSY=0)的并插入一个延时后 ,主机自动发送 START 条件 ,随后就是发送从机地址。

主在仲裁丢失的情况下，主机自动切换回从模式，如果从机地址被选中，还将会自动发送 ACK 应答。

注：在从机地址在总线上面发出后，无论收到 ACK 值的内容是什么，START 位都由硬件复位。如果发生了仲裁丢失。START 位也由硬件清零。在 START 位为 1 期间，如果 I2C 作为一个从机（ADDR=1）被选中，I2C 会切换到从机模式，并且在 ADDRCF 位被置 1 时 START 位被清零。

注：相同的程序适用于重复开始条件。在这种情况下，BSY=1。

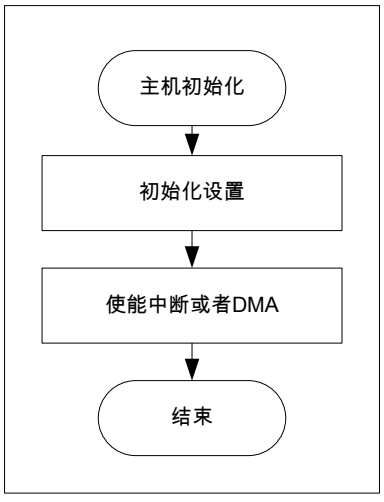


图 25.15 主机初始化流程

主机接收器寻址一个 10 位地址的从机的初始化

如果从机地址是 10 位格式，你可以选择清除 I2Cx_CR2 寄存器的 HEAD10R 位以发送一个完整的读序列。在

这种情况下，主机会在 START 位被置 1 后，自动发送下面的完整序列：(Re)Start+从机地址 10 位头的写操

作+从机地址的第二字节+ReStart+从机地址 10 位头的读操作。

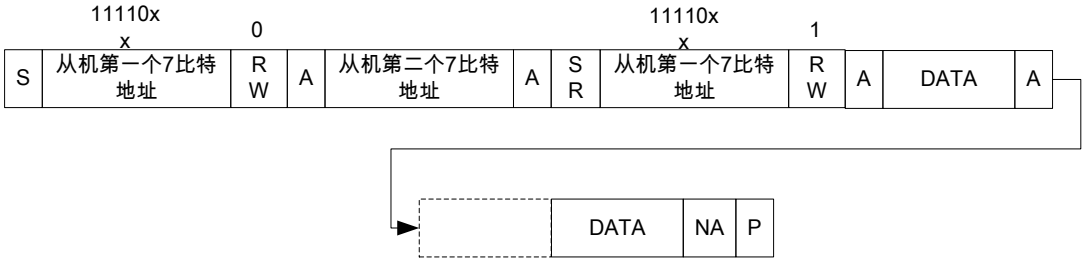


图 25.16 10 比特主机读（HEAD10R=0）

- 如果主机要寻址到一个 10 位地址的从机，将数据发送给这个从机，然后从同一个从机读取数据，主机发送流程必须首先完成。然后以 HEAD10R=1 的状态重复开始条件和 10 位从机地址。在这种情况下，主机发送序列如下：ReStart+从机地址 10 位头的读操作。

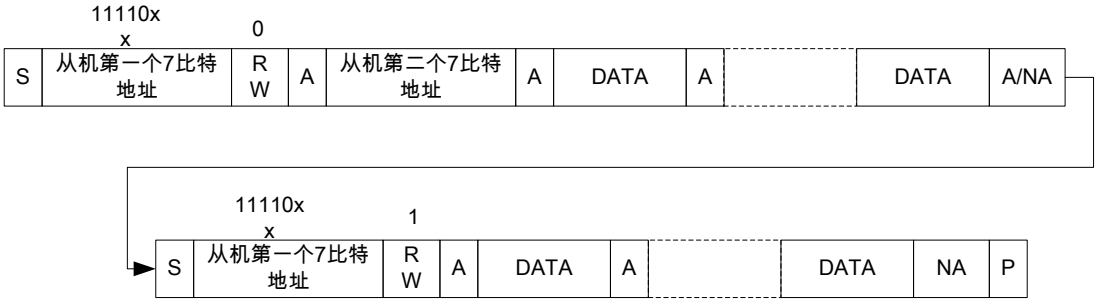


图 25.17 10 比特主机读（HEAD10R=1）

主机发送

在写传输的情况下，每个字节发送完之后，TXIS 标志会被置 1，也就是在第 9 个 SCL 脉冲的时候收到一个 ACK。

如果 I2Cx_CR1 寄存器的 TXIE 位为 1，则会由 TXIS 事件产生一个中断。向 I2Cx_TXDR 寄存器写入下一个发送数据时，这个标志位被清除。

在传输过程中的 TXIS 事件个数与 NBYTES 中写入的值对应。如果数据要发送的字节总数大于 255，必须将 I2Cx_CR2 寄存器的 RELOAD 位置 1 以选择重载模式。这种情况下，发送了 NBYTES 个字节之后，TCR 标志被置位，并且 SCL 线被拉低直至 NBYTES[7:0]中被写入一个非零值。

在收到一个 NACK 时 TXIS 位不会被置 1。

- 当 RELOAD=0 以及 NBYTES 个数据已传输完：
  - 自动结束模式（AUTOEND=1）下，会自动发送一个 STOP 条件。
  - 在软件结束模式下（AUTOEND=0），TC 标志被置 1，并且 SCL 线被拉低，这时要软件执行下一步的动作：这时如果已经配置好从机地址和要传送的字节数，就可以将 I2Cx_CR2 中的 START 位置 1，再次发出一个 START 条件。将 START 位置 1 的操作将会清除 TC 标志，并在总线上发出 START 条件。可以将 I2Cx_CR2 寄存器的 STOP 位置 1 来发出停止条件。将 STOP 位置 1 的操作将会清除 TC 标志，并在总线上发出 STOP 条件。
- 如果收到一个 NACK：TXIS 标志不会被置 1，并在收到 NACK 之后自动发送一个 STOP 条件。I2Cx_ISR 寄存器的 NACKF 标志会被置 1，这时如果 NACKIE 位为 1，就会产生中断。

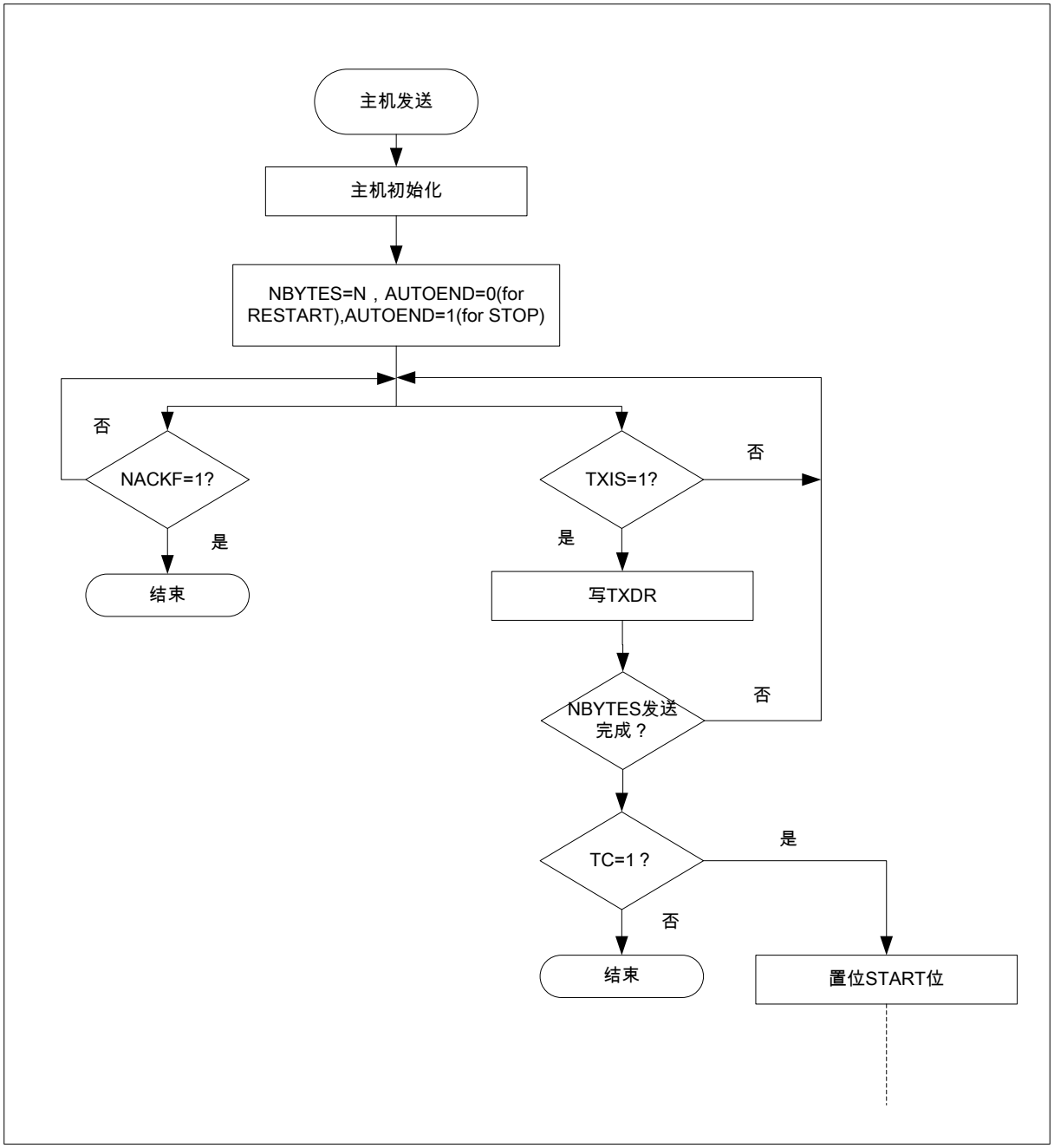


图 25.18 主机发送流程（发送数据小于等于 255 个字节）

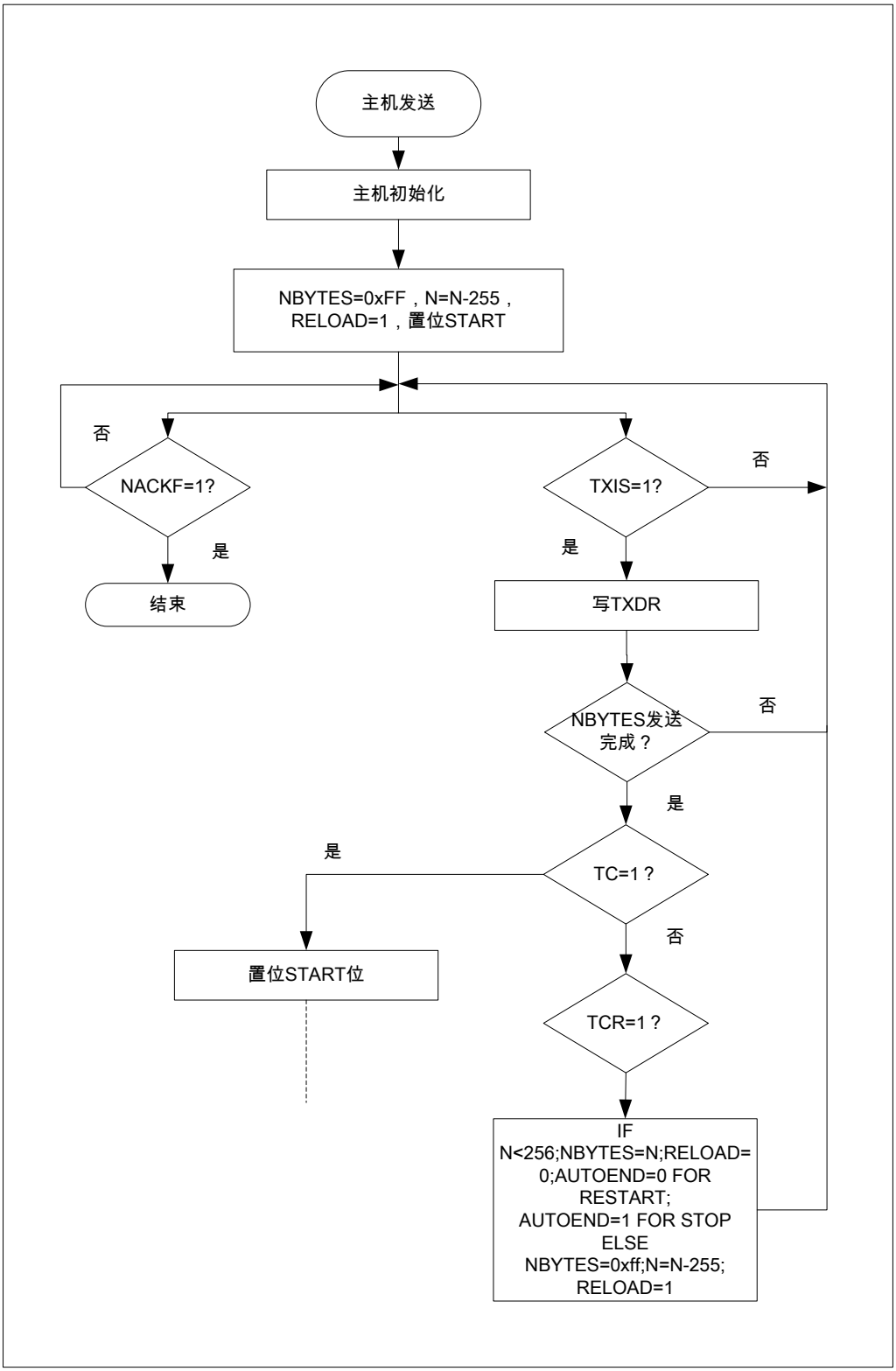


图 25.19 主机发送流程（发送数据 N 大于 255 个字节）

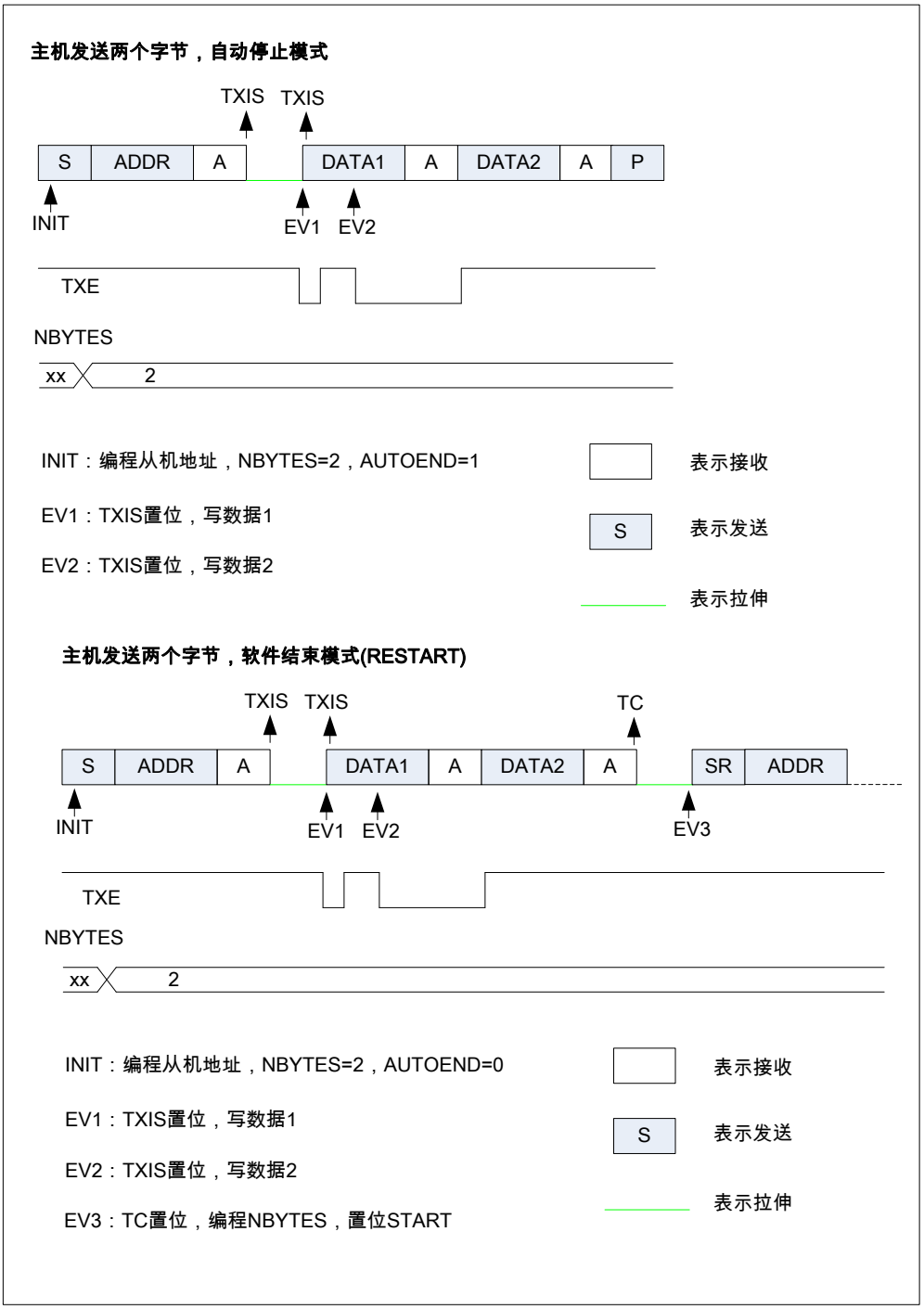


图 25.20 主机发送总线时序

主机接收

在读传输的时 ,在每个字节接收完后 ,第 8 个 SCL 脉冲时 ,RXNE 标志会置 1。如果 I2Cx_CR1 寄存器的 RXIE 位为 1，则会由 RXNE 事件产生一个中断。在读取 I2Cx_RXDR 时 RXNE 会被自动清零。

如果数据要接收的字节总数大于 255，必须将 I2Cx_CR2 寄存器的 RELOAD 位置 1 以选择重加载模式。这种情况下，发送了 NBYTES 个字节之后，TCR 标志被置位，并且 SCL 线被拉低直至 NBYTES[7:0]被写入一个非零值。

●当 RELOAD=0 以及 NBYTES 个数据已传输完：

- 自动结束模式(AUTOEND=1)下 ,在收到最后一个字节后会自动发送一个 NACK 和一个 STOP 条件。
- 在软件结束模式 (AUTOEND=0) 下，在收到最后一个字节后会自动发送一个 NACK，然后 TC 标志被置 1，并且 SCL 线被拉低，这时要软件来执行后一步的操作：这时如果已经配置好从机地址和要传送的字节数，就可以将 I2Cx_CR2 中的 START 位置 1，再次发出一个 START 条件。将 START 位置 1 的操作将会清除 TC 标志 ,并在总线上发出 START 条件及从机地址。可以将 I2Cx_CR2 寄存器的 STOP 位置 1 来发出停止条件。将 STOP 位置 1 的操作将会清除 TC 标志，并在总线上发出 STOP 条件。

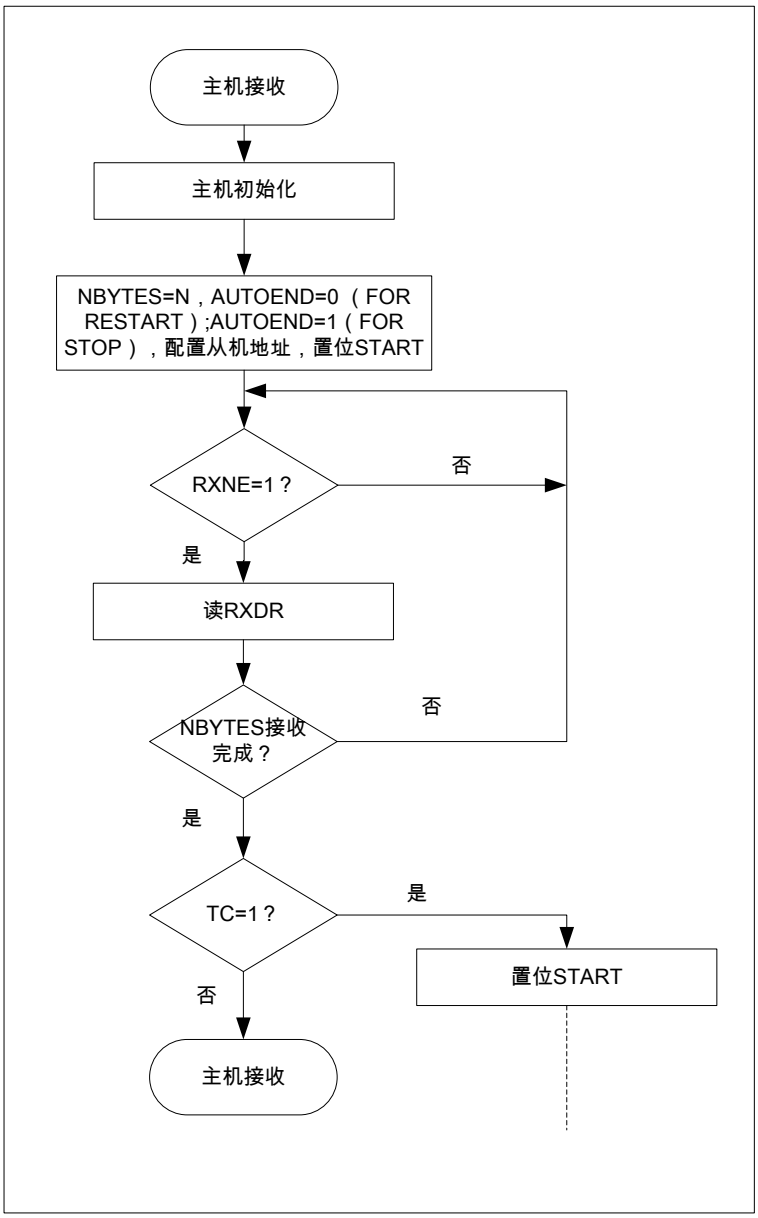


图 25.21 主机接收流程图（小于等于 255 个字节）

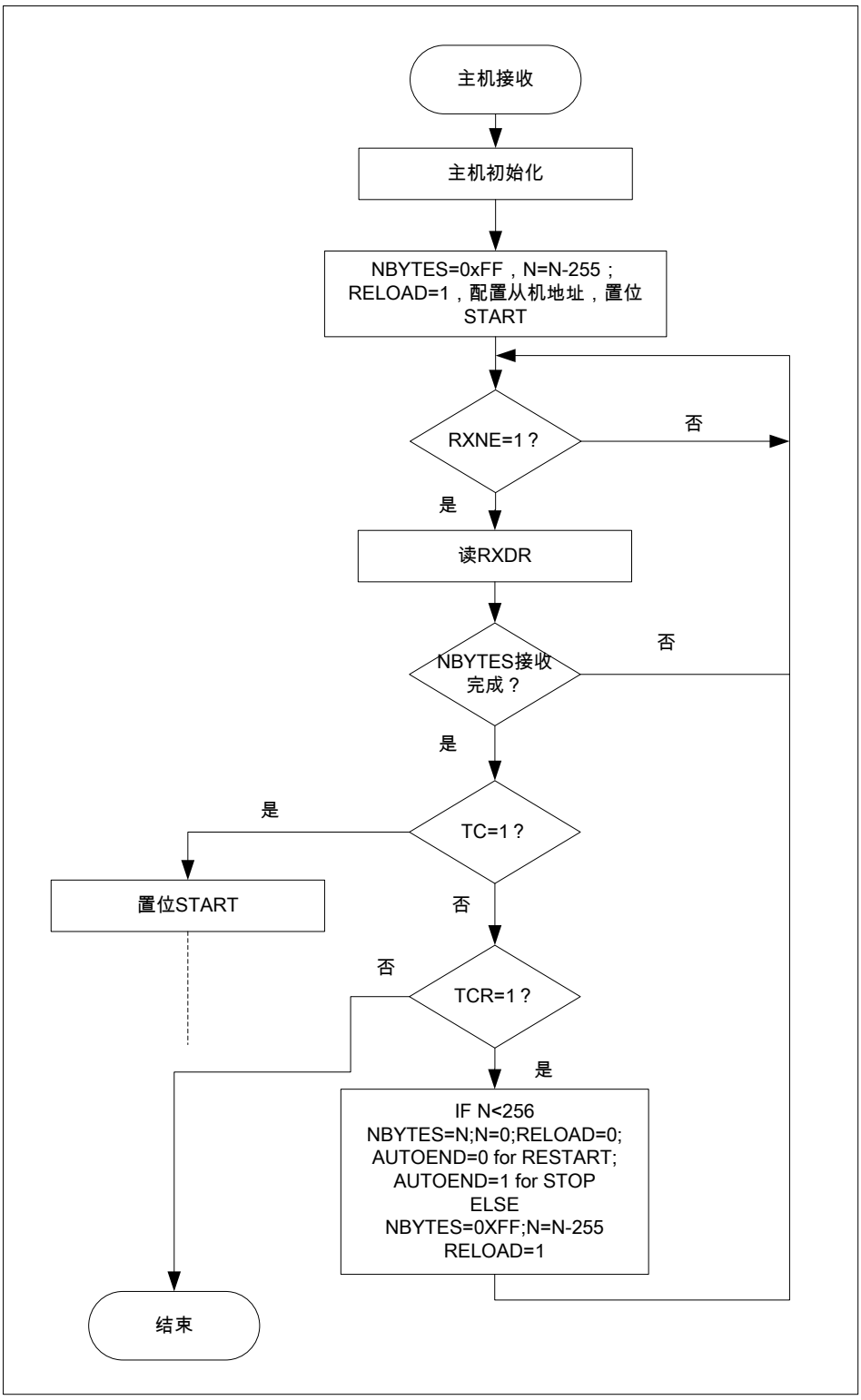


图 25.22 主机接收流程图（N>255 个字节）

图 25.23 主机接收总线时序

25.2.10. I2Cx_TIMINGR 寄存器配置举例

下面的表中是配置 I2C 时序的参考配置，配置的结果根据实际的同步结果会有一定的误差。

表 25.5 I2C 工作时钟是 F_{I2CCLK}=8M 时的时序配置

参数	标准速度		快速模式	快速+模式
	10kHz	100kHz	400kHz	500kHz
PRESC	1	1	0	0
SCLL	0xC7	0x13	0x9	0x6
T_{SCLL}	200*250ns=50μs	20*250ns=5μs	10*125ns=1250ns	7*125ns=875ns

SCLH	0xC3	0xF	0x3	0x3
T_{SCLH}	196*250ns=49μs	16*250ns=4μs	4*125ns=500ns	4*125ns=500ns
T_{SCL}	~100μs	~10μs	~2500ns	~2000ns
SDADEL	0x2	0x2	0x1	0x0
T_{SDADEL}	2*250ns=500ns	2*250ns=500ns	1*125ns=125ns	0ns
SCLDEL	0x4	0x4	0x3	0x1
T_{SCLDEL}	5*250ns=1250ns	5*250ns=1250ns	4*125ns=500ns	2*125ns=250ns

表 25.6 I2C 工作时钟是 F_{I2CCLK}=16M 时的时序配置

参数	标准速度		快速模式	快速+模式
	10kHz	100kHz	400kHz	1000kHz
PRESC	3	3	1	0
SCLL	0xC7	0x13	0x9	0x4
T_{SCLL}	200*250ns=50μs	20*250ns=5μs	10*125ns=1250ns	5*62.5ns=312.5ns
SCLH	0xC3	0xF	0x3	0x2
T_{SCLH}	196*250ns=49μs	16*250ns=4μs	4*125ns=500ns	3*62.5ns=187.5ns
T_{SCL}	~100μs	~10μs	~2500ns	~1000ns
SDADEL	0x2	0x2	0x2	0x0
T_{SDADEL}	2*250ns=500ns	2*250ns=500ns	2*125ns=125ns	0ns
SCLDEL	0x4	0x4	0x3	0x2
T_{SCLDEL}	5*250ns=1250ns	5*250ns=1250ns	4*125ns=500ns	3*62.5ns=187.5ns

表 25.6 I2C 工作时钟是 F_{I2CCLK}=48M 时的时序配置

参数	标准速度		快速模式	快速+模式
	10kHz	100kHz	400kHz	1000kHz
PRESC	0xB	0xB	5	5
SCLL	0xC7	0x13	0x9	0x3
T_{SCLL}	200*250ns=50μs	20*250ns=5μs	10*125ns=1250ns	4*125ns=500ns
SCLH	0xC3	0xF	0x3	0x1
T_{SCLH}	196*250ns=49μs	16*250ns=4μs	4*125ns=500ns	2*125ns=250ns
T_{SCL}	~100μs	~10μs	~2500ns	~875ns
SDADEL	0x2	0x2	0x3	0x0
T_{SDADEL}	2*250ns=500ns	2*250ns=500ns	3*125ns=375ns	0ns
SCLDEL	0x4	0x4	0x3	0x1
T_{SCLDEL}	5*250ns=1250ns	5*250ns=1250ns	4*125ns=500ns	2*125ns=250ns

25.2.11. SMBus 特定功能

系统管理总线（SMBus）是一个种两线接口，通过它可以与各种设备和系统的其余部分互相通讯。它基于 I2C 操作原理。SMBus 提供了一种针对系统和电源管理相关任务的控制总线。

该外设兼容的 SMBus 规范 2.0 版（http://smbus.org/specs/）。

系统管理总线规范是指三种类型的设备。

- 从机是指接收或相应命令的设备。
- 主机是下达命令，产生时钟和终止传输的设备。
- HOST 是一个特殊的主机，它向系统 CPU 提供主接口。HOST 必须具备主机和从机的双重功能，并且必须支持 SMBus HOST 通知协议。一个系统中只可以有一个 HOST。

本外设可以配置为主机或从机，当然也可以作为 HOST。SMBus 是基于 I2C 规范修订版 2.1。

总线协议

对于任何给定的设备有 11 个可能的命令协议。一个设备可以使用任何或全部的 11 个协议进行通信。协议是快速命令，发送字节，接收字节，写字节，写字，读字节，读字，过程调用，块读，块写和块写块读过程调用。这些协议是由用户软件实现的。

对于这些协议的更多细节，请参考 SMBus 规范版本.2.0(http://smbus.org/specs/)。

地址解析协议 ARP

SMBus 从机地址冲突的问题可以通过给每个从设备动态标定一个新的独特的地址的方式解决。为了分配地址，需要一种区分每个设备的机制，每个设备必须拥有一个唯一的设备标识符。这个 128 位的数字是由软件实现的。

本外设支持地址解析协议（ARP）。将 I2Cx_CR1 寄存器的 SMBDEN 位置 1，会启用 SMBus 设备的默认地址（0b1100001）。ARP 命令由用户软件实现。

ARP 支持的仲裁动作也是在从机模式下完成。

SMBus 地址解析协议的更多细节，请参考 SMBus 规范版本 2.0(http://smbus.org/specs/)。

命令的接收和数据应答的控制

一个 SMBus 的接收器必须能够对每个收到的命令或数据回应 NACK。为了允许从机模式下的 ACK 控制，必须将 I2Cx_CR1 寄存器的 SBC 位置 1，以启用从机字节控制模式。

HOST 通知协议

设置 I2Cx_CR1 寄存器的 SMBHEN 位，使得本外设支持 HOST 通知协议。在这种情况下，HOST 会应答 SMBus 主机地址（0b0001000）。

使用此协议时，本设备会作为主机，而 HOST 则会作为一个从机。

SMBus 报警

本外设可选 SMBus 提醒信号支持。一个仅作为从机的设备在想要发起通讯的时候，可以通过 SMBALERT 引脚通知 HOST。HOST 会处理这个中断，并且随即通过提醒响应地址(0b0001100)来访问全部的 SMBALERT 设备。只有将 SMBALERT 引脚拉低的设备会回应提醒响应地址。

当被配置为从机设备（SMBHEN=0）时，将 I2Cx_CR1 寄存器的 ALERTEN 位置 1，会导致 SMBA 引脚被拉低。提醒响应地址则会同时启用。

当被配置为 HOST 设备（SMBHEN=1）时，只要发现 ALERTEN=1，并且在 SMBA 引脚上检测到一个下降沿，I2Cx_ISR 寄存器中的提醒标志就会被置位。如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。当 ALERTEN=0 时，即使是 SMBA 外部引脚为低，提醒线也会被认为是高。

如果不需要使用 SMBus 提醒引脚，只要是 ALERTEN=0，SMBA 脚就可以被用来作为一个标准的 GPIO。

包错误检查

SMBus 规范中介绍到的包错误检查机制可以提高通信可靠性。包错误检查是通过在每个消息后面附带一个包错误检查码来实现的。PEC 码通过对全部的消息字节(包括地址和读写位)使用多项式 $C(x)=x^8+x^2+x+1$ CRC8 计算得来。

本外设内嵌硬件 PEC 计算单元 ,在接收数据的时候如果发现 PEC 结果不匹配 ,会允许自动发送一个 NACK。

超时

本外设内嵌一个硬件定时器，以便和 SMBus 规范 V2.0 中定义的 3 个超时相符。

表 25.7 SMBus 超时定义

符号	参数	限制		单位
		最小	最大	
T_{TIMOUT}	SCL 低电平超时	25	35	ms
$T_{LOW:SEXT}$	累计的时钟低电平时间（从机模式）	-	25	ms
$T_{LOW:MEXT}$	累计的时钟低电平时间（主机模式）	-	10	ms

$T_{LOW:SEXT}$ 是一个给定的从机设备从 START 到 STOP 之间可以延长的时钟周期的累计。另一个从机设备或主机也可能对时钟进行占用从而导致总的时钟低的占用时间大于 $T_{LOW:SEXT}$。因此，这个参数的测量条件是从机作为一个全速的主机的唯一通讯目标。

$T_{LOW:MEXT}$ 是一个主机设备按照 START 到 ACK ,ACK 到 ACK 或者 ACK 到 STOP 的方式发送一个字节所允许的时钟周期的累计。另一个从机设备或主机也可能对时钟进行占用从而导致总的时钟低的占用时间大于 $T_{LOW:MEXT}$。因此，这个参数的测量条件是有只一个全速的从机作为唯一的通讯目标。

总线空闲检测

如果主机发现时钟和数据信号保持为高的时间 T_{IDLE} 大于 $T_{HIGH,MAX}$，就可以认为目前总线处于空闲状态。

这个时序参数覆盖了那种主机被动态的加入到总线上，不一定检测到了 SMBCLK 或者 SMBDAT 上的状态转换的条件。在这种情况下，主机必须等待足够长的时间，以确保传输是否正在进行中。本外设支持硬件总线空闲检测。

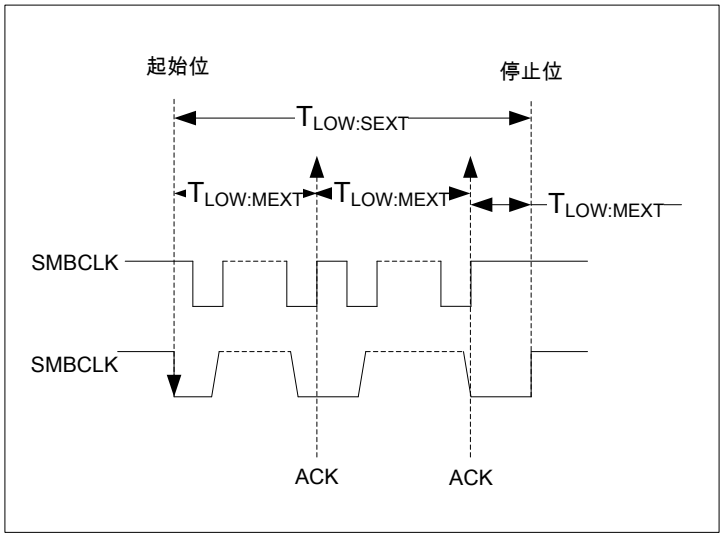


图 25.24 $T_{LOW:MEXT}$ 和 $T_{LOW:SEXT}$ 超时时序

25.2.12. SMBus 初始化

本节只针对支持 SMBus 功能的部分。为了正确执行 SMBus 通讯，在 I2C 初始化的基础上还有一些其它的初始化动作要做：

命令的接收和数据应答控制

一个 SMBus 的接收器必须能够对每个收到的命令或数据回应 NACK。为了允许从机模式下的 ACK 控制，必须将 I2Cx_CR1 寄存器的 SBC 位置 1，以启用从机字节控制模式。

特定地址（从机模式）

如有必要，需要启用特定的 SMBus 地址。

- 将 I2Cx_CR1 寄存器的 SMBDEN 位置 1，会启用 SMBus 设备的默认地址（0b1100001）。
- 将 I2Cx_CR1 寄存器的 SMBHEN 位置 1 会启用 SMBus 主机地址（0b0001000）。
- I2Cx_CR1 寄存器的 ALERTEN 位被置 1 时，会启用通知响应地址。

包错误检查

将 I2Cx_CR1 寄存器的 PECEN 位置 1，会启用的 PEC 计算这时的 PEC 传送由硬件计数器位（I2Cx_CR2 寄存器中的 NBYTES）来配合管理。PECEN 必须在使能 I2C 之前设置好。

既然 PEC 传输是有硬件字节计数器管理的，所以在 SMBus 工作于从机模式时，SBC 位必须先置 1。PEC 数据会在传送了 NBYTES-1 个数据被传送完毕之后发送，当然这时 PECBYTE 位要求是 1，并且 RELOAD 位要求是 0。如果 RELOAD 位是 1，那 PECBYTE 位就没有作用了。

注意：当 I2C 启用时不允许更改 PECEN 的配置。

表 25.8 SMBus 带 PEC 的的配置表

模式	SBC	RELOAD	AUTOEND	PECBYTE
主机发送/接收 NBYTES+PEC+STOP	x	0	1	1

主机发送/接收 NBYTES+PEC+RESTART	x	0	0	1
从机发送/接收+PEC	1	0	x	1

超时检测

启用超时检测功能，须将 I2Cx_TIMEOUTR 寄存器的 TIMOUTEN 位和 TEXTEN 位置 1。定时器须用某种方式编程，以实现在 SMBus 规范 2.0 给出的最大时间之前检测到超时。

- $T_{TIMEOUT}$ 检查

要启用 $T_{TIMEOUT}$ 检查,先要将要检查的 $T_{TIMEOUT}$ 参数对应的定时器重载值写入到 12 位的 TIMEOUTA[11:0]。要检查 SCL 低电平超时，还需要保证 TIDLE 位为 0。在 I2Cx_TIMEOUTR 寄存器的 TIMOUTEN 被置 1 使能了定时器后。如果 SCL 被拉低的时间大于(TIMEOUTA+1)*2048*T_{I2C_CLK}，I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

注意：在 TIMEOUTEN 位为 1 的时候不允许改变 TIMEOUTA[11:0]位域和 TIDLE 位的配置。

- $T_{LOW:SEXT}$ 和 $T_{LOW:MEXT}$ 检查

必须配置 12 位定时器 TIMEOUTB，从机模式会检查 $T_{LOW:SEXT}$，而主机模式会检查 $T_{LOW:MEXT}$。标准只规定了最大值 ,可选择与之相同的值。在 I2Cx_TIMEOUTR 寄存器的 TIMOUTEN 被置 1 使能了定时器后。如果 SMBus 外设将 SCL 拉低的时间达到了(TIMEOUTB+1)*2048*T_{I2C_CLK},并且又没有达到总线空闲检查中描述的时间，I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

注意：当 TEXTEN 位为 1 时，不允许改变 TIMEOUTB 的配置。

总线空闲检测

要使能 T_{IDLE} 检查,必须根据要实现的 T_{IDLE} 参数向 TIMEOUTA[11:0]位域写入预装载值。要同时检测 SCL 和 SDA 上的高电平超时，TIDLE 须写为 1。将 I2Cx_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1，使能了定时器后。如果 SCL 和 SDA 线同时保持高的时间大于(TIMEOUTA+1)*4*T_{I2C_CLK} ,I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

25.2.13. I2C_TIMEOUTR 寄存器配置实例

本节只针对支持 SMBus 功能的部分。

- 配置 TIMEOUT 最长到 25 毫秒：

表 25.9 各种工作频率下 TIMEOUTA 配置举例

F_{I2CCLK}	TIMEOUTA	TIDLE	TIMEOUTEN	$T_{TIMEOUT}$
8MHz	0x61	0	1	98*2048*125ns=25ms
16MHz	0xC3	0	1	196*2048*62.5ns=25ms
48MHz	0x249	0	1	586*2048*20.08ns=25ms

- $T_{LOW:SEXT}$ 和 $T_{LOW:MEXT}$ 最长为 8ms：

表 25.10 各种工作频率下 TIMEOUTB 配置举例

F_{I2CCLK}	TIMEOUTB	TEXTEN	$T_{LOW:EXT}$
8MHz	0x1F	1	32*2048*125ns=8ms
16MHz	0x3F	1	64*2048*62.5ns=8ms

48MHz	0xBB	1	188*2048*20.08ns=8ms

- T_{IDLE}最长为50μs：

表 25.11 各种工作频率下 TIMEOUTA 配置举例（T_{IDLE}）

F_{I2CCLK}	TIMEOUTA	TIDLE	TIMEOUTEN	T_{IDLE}
8MHz	0x63	1	1	100*4*125ns=50μs
16MHz	0xC7	1	1	200*4*62.5ns=50μs
48MHz	0x257	1	1	600*4*20.08ns=50μs

25.2.14. SMBus 从机模式

本节只针对支持 SMBus 功能的部分。

除了 I2C 的从机传输管理还需要执行一些额外的软件流程以支持 SMBus。

SMBus 从机发送

在 SMBus 模式下，SBC 必须为 1，以允许在发送了足够数量的字节之后跟上 PEC 字节。当 PECBYTE 位被置 1，NBYTES[7:0]中编程的字节数中要包括 PEC 字节。在这种情况下 TXIS 中断总数将为 NBYTES-1 个，如果主机在 NBYTES-1 次传送之后还要求更多的字节，I2Cx_PECR 寄存器的内容就会自动被发送出去。

注意：RELOAD 位被置 1 时，PECBYTE 位没有作用。

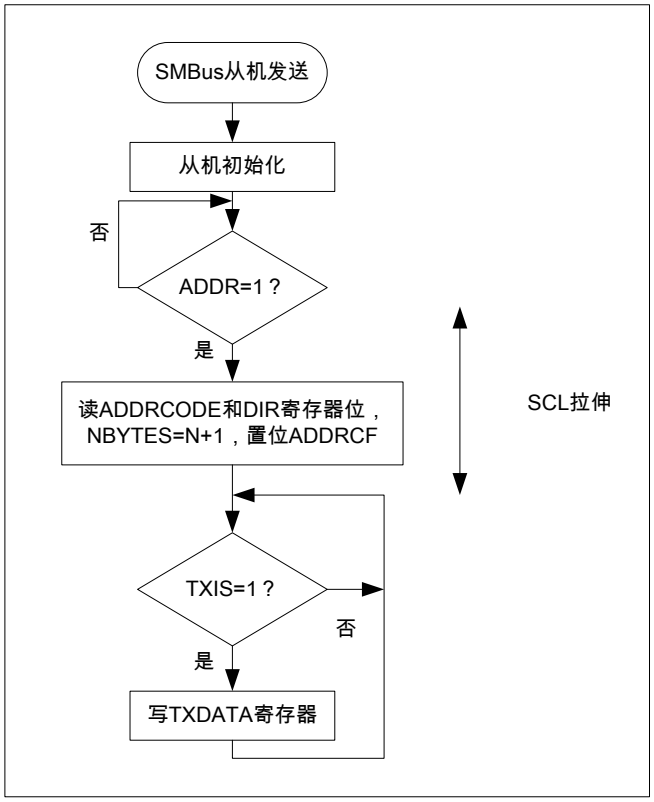


图 25.25 SMBus 发送 N 个字节+PEC 流程图

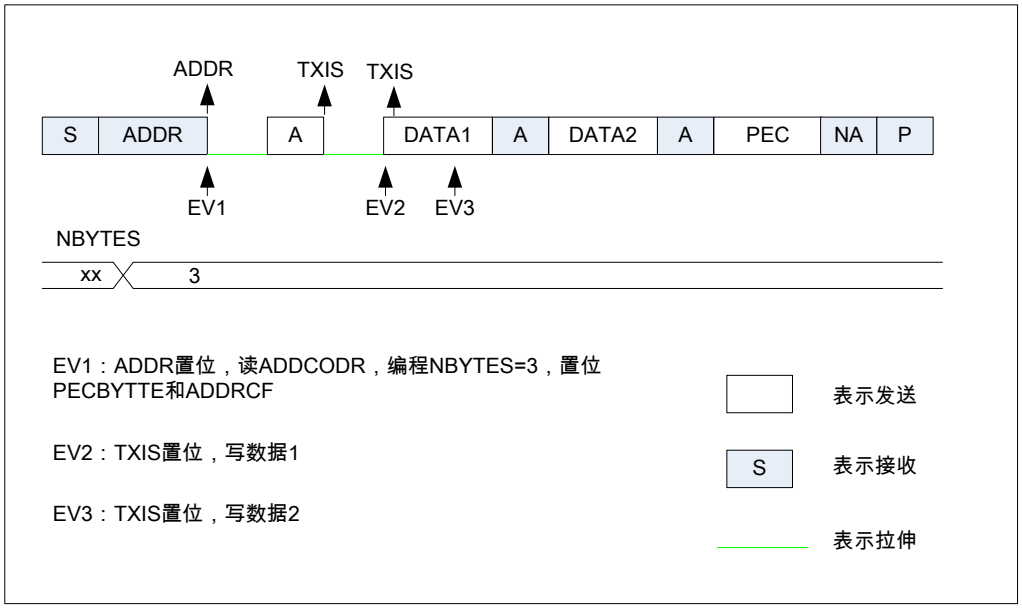


图 25.26 SMBus 从机发送总线时序（SBC=1）

SMBus 从机接收

在工作于 SMBus 模式时，SBC 必须为 1，以允许在发送了足够数量的字节之后跟上 PEC 字节。为了实现每个字节的 ACK 控制，必须选择重载模式（RELOAD=1）。

为了检查 PEC 字节，必须清除 RELOAD 位，而且必须置 PECBYTE 位为 1。在这种情况下，已收到 NBYTES-1 个数据后，下一个收到的字节会与内部 I2Cx_PECR 寄存器的内容进行比较。如果比较不匹配，自动生成一个 NACK，如果比较匹配，自动生成一个 ACK，这时 ACK 位的值是不是 1 都没所谓。PEC 字节一旦被收到，它像任何其他数据样被复制到 I2Cx_RXDR 寄存器，并且 RXNE 标志被置 1。

在 PEC 不匹配的情况下，PECERR 标志被置 1，如果 I2Cx_CR1 寄存器的 ERRIE 位为 1，会产生一个中断。

如果 ACK 由软件控制，可以令 PECBYTE=1，并在连续传送中使用同样的写操作，将 NBYTES 的值写成实际要接收的值。在 NBYTES-1 个字节收到后，下一个收到的字节被当做 PEC 来检查。

注意：RELOAD 位被置 1 时，PECBYTE 位没有作用。

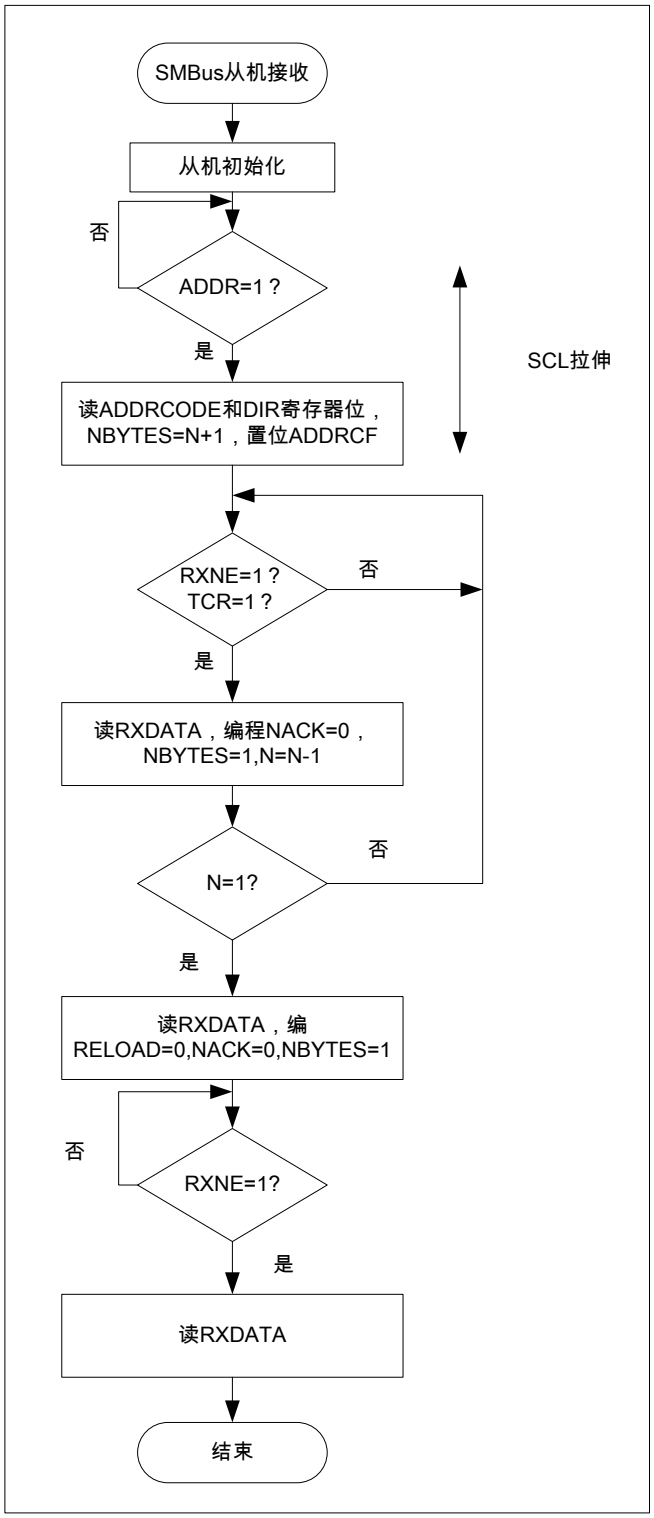


图 25.27 SMBus 从机接收流程

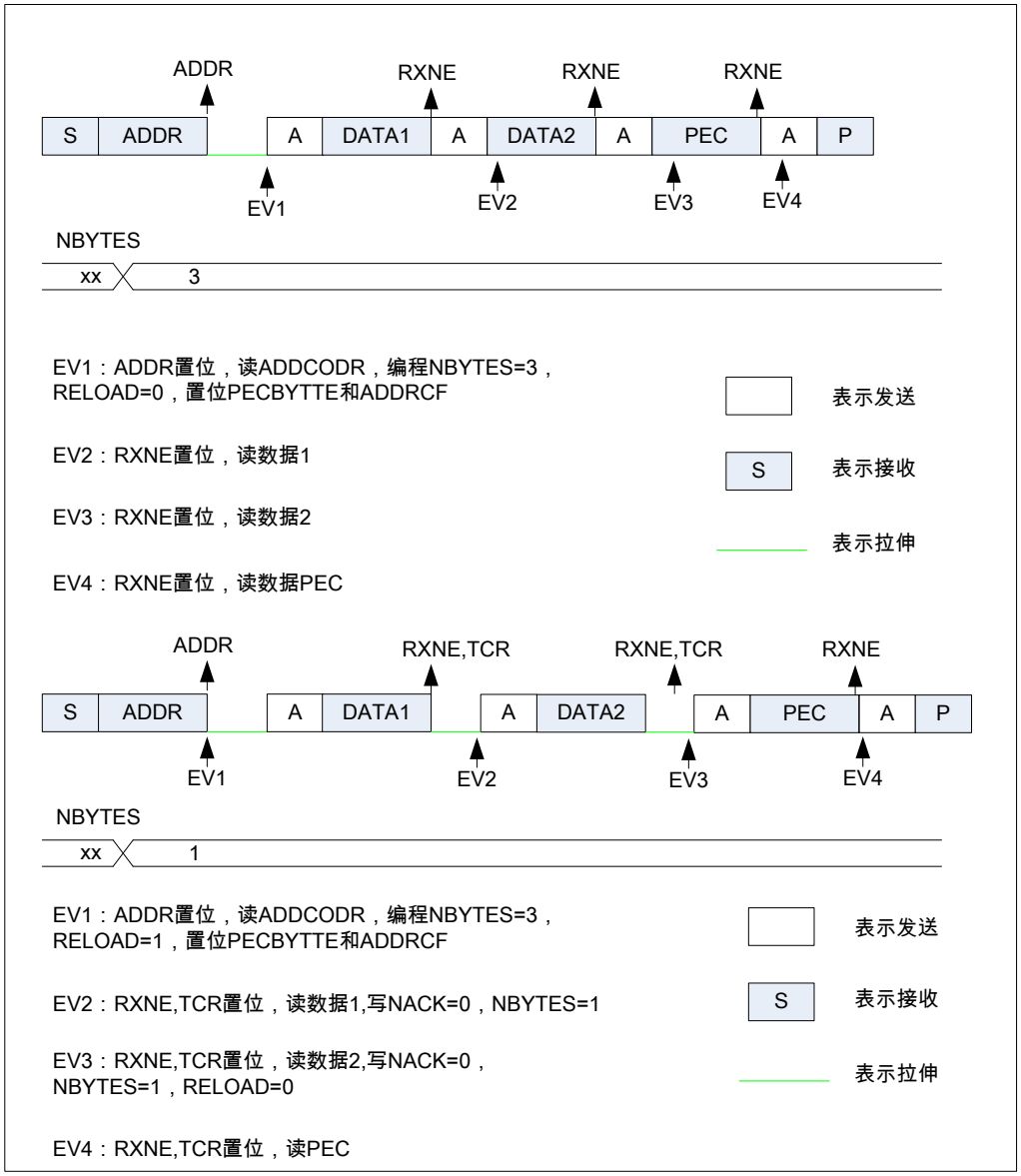


图 25.28 SMBus 从机接收总线时序（SBC=1）

本节只针对支持 SMBus 功能的部分。除了 I2C 的从机传输管理还需要执行一些额外的软件流程以支持 SMBus。

SMBus 主机发送

当 SMBus 主机要发送 PEC 时，在将 START 置 1 之前，PECBYTE 位必须置 1，并且要发送的字节数必须写到 NBYTES[7:0]位域中。在这种情况下将 TXIS 中断总数为 NBYTES-1。所以，如果在 NBYTES=0x1 的时候将 PECBYTE 位置 1，I2Cx_PECR 寄存器的内容就会被自动发送出去。

如果 SMBus 主机要在 PEC 数据之后自动发送一个 STOP 条件，则应该选择自动结束方式（AUTOEND=1）。在这种情况下，PEC 数据发送过后会自动跟上一个 STOP 条件。

当 SMBus 主机要在 PEC 传输之后发送 RESTART 条件，则必须选择软件结束模式（AUTOEND=0）。在这种情况下，一旦 NBYTES-1 个字节被传送完，I2Cx_PECR 寄存器的内容会被发送，PEC 发送完后，TC 标志会被置 1，并且 SCL 线会被拉低。RESTART 条件必须在 TC 中断服务程序中编程实现。

注意：RELOAD 位被置 1 时，PECBYTE 位没有作用。

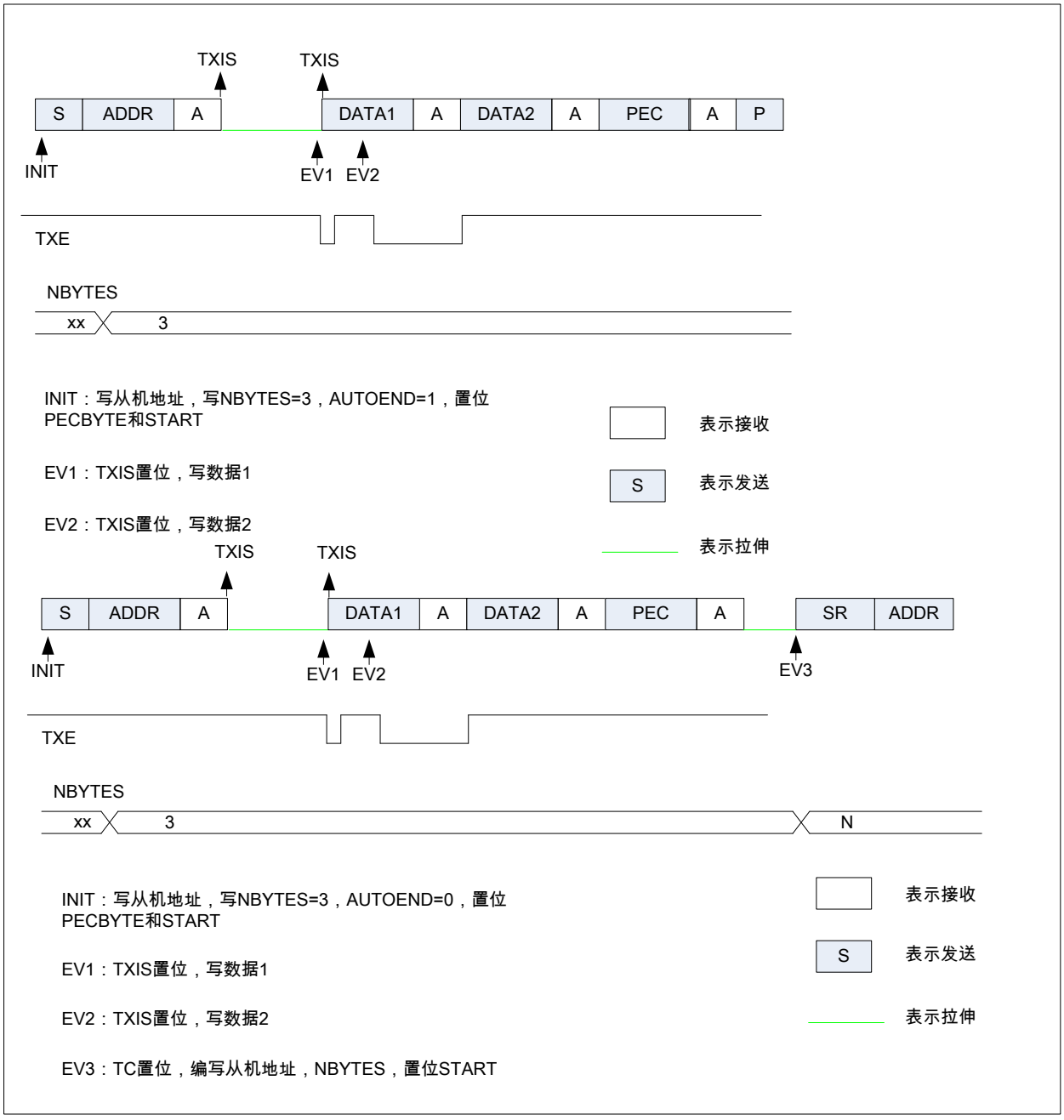


图 25.29 SMBus 主机发送总线时序

SMBus 主机接收

当 SMBus 主机要在传送末尾接收 PEC 之后自动发出停止条件，须选择自动结束模式（AUTOEND=1）。PECBYTE 位必须先置 1 ,以及从机地址也要在置 START 位之前预先设置。在这种情况下 ,已收到 NBYTES-1 个数据后，下一个收到的字节会与内部 I2Cx_PECR 寄存器的内容进行比较。对 PEC 字节会给出一个 NACK 响应后面再跟着一个 STOP 条件。

当 SMBus 主机要在传送末尾接收 PEC 之后发出 RESTART 条件，须选择软件结束模式（AUTOEND=0）。PECBYTE 位必须先置 1 ,以及从机地址也要在置 START 位之前预先设置。在这种情况下 ,已收到 NBYTES-1 个数据后，下一个收到的字节会与内部 I2Cx_PECR 寄存器的内容进行比较。在收到 PEC 字节后，TC 标志被置位，同时 SCL 线被拉低。RESTART 条件可以在 TC 中断服务程序中编程实现。

注意：RELOAD 位被置 1 时，PECBYTE 位没有作用。

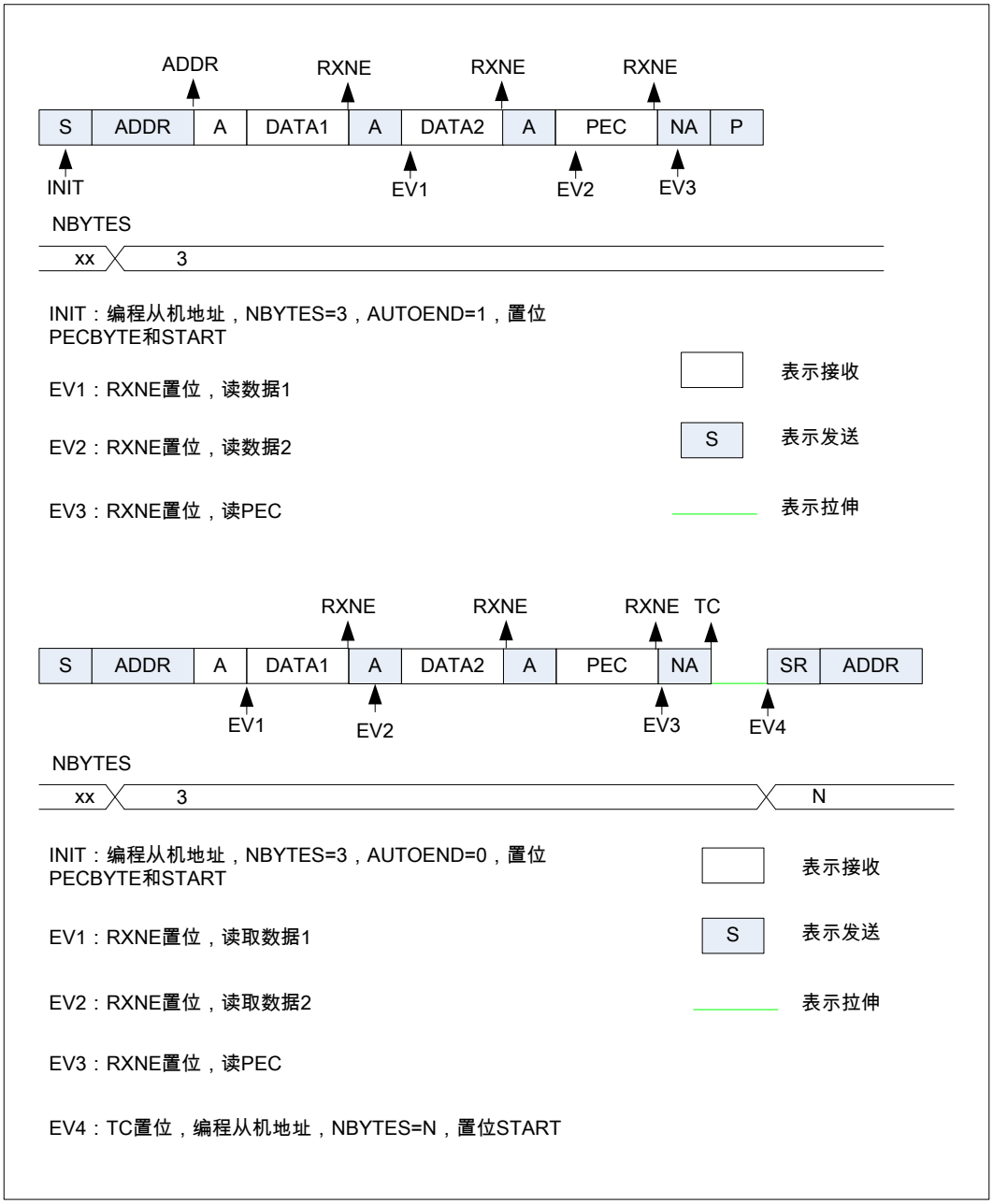


图 25.30 SMBus 主机接收总线时序

25.2.15. 错误条件

以下是可能导致通信失败的错误条件。

总线错误(BERR)

如果在 9 个 SCL 时钟脉冲以内检测到一个 START 或者 STOP 条件，会发生一个总线错误。当 SCL 为高的时候在 SDA 上出现上升沿或者下降沿，会检测到 START 或 STOP 条件。

只有在 I2C 处于数据传输状态下（作为主机在发送或作为从机被寻址到）才会有总线错误标志置位，例如在

从机模式下的寻址阶段就不会。

对于从机模式下，一个错位的 START 或者 RESTART 条件会被从机当成正常的 START 条件来处理。

当检测到总线错误，I2Cx_ISR 寄存器中的 BEER 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

仲裁丢失（ARLO）

在往 SDA 线上发高电平，但在 SCL 的上升沿却从 SDA 采样得到低电平时，就会检测到一次仲裁丢失错误。

- 在主机模式下，仲裁丢失在地址阶段、数据阶段以及数据确认阶段进行检测。在这种情况下，SDA 和 SCL 线被释放，START 控制位由硬件清零，主机自动切换到从机模式。
- 在从机模式下，仲裁丢失在数据阶段和数据确认阶段进行检测。在这种情况下，传输被终止，SCL 和 SDA 线被释放。

当检测到仲裁丢失错误，I2Cx_ISR 寄存器中的 ARLO 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

溢出/下溢错误（OVR）

如果 NOSTRETCH=1，在从机模式下只要下列条件发生，就会检测到下溢或溢出错误：

- 在接收时，尚未读取 RXDR 寄存器的时候又有一个新的字节接收到。

新收到的字节会丢失，并且会有一个 NACK 自动发送出去，当作是对这个新的字节的响应。

- 在发送时：

  —当应该发送第一个数据字节时却有 STOPF=1。如果 TXE=0 ,那么 I2Cx_TXDR 寄存器的值会发送出去，如果不是 0，那么就会发送 0xFF。

  —在一个新的字节应该被写入到 I2Cx_TXDR 寄存器，但却没有写，那就会将 0xFF 发送出去。

当检测到下溢/溢出错误，I2Cx_ISR 寄存器中的 OVR 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

包错误检查错误（PECERR）

本节只针对支持 SMBus 功能的部分。

在收到的 PEC 字节与 I2Cx_PECR 寄存器的内容不匹配时会检测到 PEC 错误。错误的 PEC 接收后，会自动发送一个 NACK。

当检测到 PEC 错误，I2Cx_ISR 寄存器中的 PECERR 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

超时错误（超时）

本节只针对支持 SMBus 功能的部分。

这些条件中的任何一个都会导致超时错误发生：

- TIDLE=0 并且 SCL 保持低的时间达到了在 TIMEOUTA[11:0]位域所定义的时间：这是用来检测 SMBus

超时。

- TIDLE=1，SDA 和 SCL 都保持高电平并达到了在 TIMEOUTA[11:0]位域所定义的时间：这是用来检测总线空闲状态。
- 主机时钟低延长累计时间达到了在 TIMEOUTB[11:0]位域规定的时间（SMBus 的 $T_{LOW:MEXT}$ 参数）
- 从机时钟低延长累计时间达到了在 TIMEOUTB[11:0]位域规定的时间（SMBus 的 $T_{LOW:SEXT}$ 参数）

当在主机模式下检测到超时，会自动发送一个 STOP 条件。

当在从机模式下检测到超时，SDA 线和 SCL 线会自动释放。

当检测到超时错误，I2Cx_ISR 寄存器中的 TIMEOUT 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

警告（ALERT）

本节只针对支持 SMBus 功能的部分。

当 I2C 接口配置为 HOST（SMBHEN=1），SMBA 引脚检测功能被启用（ALERTEN=1），并且在 SMBA 脚上检测到一个下降沿时，ALERT 标志会被置 1。如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

25.2.16. DMA 请求

利用 DMA 发送

通过设置在 I2Cx_CR1 寄存器 TXDMAEN 位，可以启用 DMA（直接存储器存取）发送。数据被预先放到 DMA 外设所设定的 SRAM 区域发往 I2Cx_TXDR 寄存器，而不管 TXIS 位是不是 1。

只使用 DMA 传输数据。

- 在主机模式下：初始化，从机地址，方向，字节数和起始位都由软件设置（已经发送了的从机地址，不能使用 DMA 传输）。当所有数据都使用 DMA 传输，必须在设置 START 位之前初始化 DMA。传输的结束由 NBYTES 计数器来管理。
- 在从机模式下：
  - NOSTRETCH=0 时，所有数据都使用 DMA 传输时，DMA 必须在地址匹配事件之前初始化好，或者中断服务程序中，清除 ADDR 标志之前完成这个任务。
  - 当 NOSTRETCH=1，DMA 必须在地址匹配事件之前初始化好。
- SMBus 支持的实例：PEC 的传送由 NBYTES 计数器管理。

注意：如果使用 DMA 发送，TXIE 位则不需要启用。

利用 DMA 接收

通过设置在 I2Cx_CR1 寄存器 RXDMAEN 位，可以启用 DMA（直接存储器存取）接收。数据会从 I2Cx_RXDR 寄存器取出来，转移到 DMA 外设中指向的 SRAM 区域而不管 RXNE 位是不是 1。只有数据（包括 PEC）被 DMA 传输。

- 在主机模式下，初始化，从机地址，方向，字节数和起始位，都要通过软件设置。当所有数据都使用 DMA

传输完后，必须在设置 START 位之前初始化 DMA。传输的结束由 NBYTES 计数器来管理。

- 在从机模式中如果 NOSTRETCH=0，所有数据都使用 DMA 传输后，DMA 必须在地址匹配事件之前初始化好，或者在中断服务程序中，清除 ADDR 标志之前完成这个任务。
- 如果 SMBus 支持：PEC 的传送由 NBYTES 计数器管理。

注意：如果使用 DMA 发送，RXIE 位则不需要启用。

25.2.17. 调试模式

当 CPU 进入调试模式时（核停止），SMBus 相关的超时功能是否工作取决于 DEBUG 模块中 DBG_I2Cx_SMBUS_TIMEOUT 寄存器位的配置。

25.2.18. 低功耗模式

表 25.13 低功耗模式

模式	描述
睡眠模式	可以正常工作，产生中断后可以唤醒 CPU
停止模式	模块相关寄存器状态保持，因为没有时钟，所以不能工作
待机模式	模块关闭，下次上电必须重新初始化

25.2.19. 中断

表 25.14 中断请求

中断事件	中断标志	中断清零	中断使能位
接收非空	RXNE	读 I2C_RXDR 寄存器	RXIE
发送缓冲器中断	TXIS	写 I2C_TXDR 寄存器	TXIE
检测到停止位	STOPF	写 STOPCF=1	STOPIE
RELOAD 模式传输完成	TCR	写 NBYTES!=0	TCIE
传输完成	TC	写 START=1 或者 STOP=1	
地址匹配	ADDR	写 ADDRCF=1	ADDRIE
接收到 NACK	NACKF	写 NACKCF=1	NACKIE
总线错误	BERR	写 BERRCF=1	ERRIE
溢出/下溢错误	OVR	写 OVRCF=1	
包校验错误	PECERR	写 PECERRCF=1	
超时错误	TIMEOUT	写 TIMEOUTCF=1	
SMBus 警告	ALERT	写 ALERTCF=1	

根据产品的实际功能，所有这些中断事件可以共享同一个中断向量（I2C 全局中断），或组合成 2 个（I2C 事件中断和 I2C 错误中断）中断向量。

要使能 I2C 中断，有下列顺序要求：

1. 在 NVIC 中配置和启用 I2C 中断请求通道。
2. 配置 I2C 以产生中断。

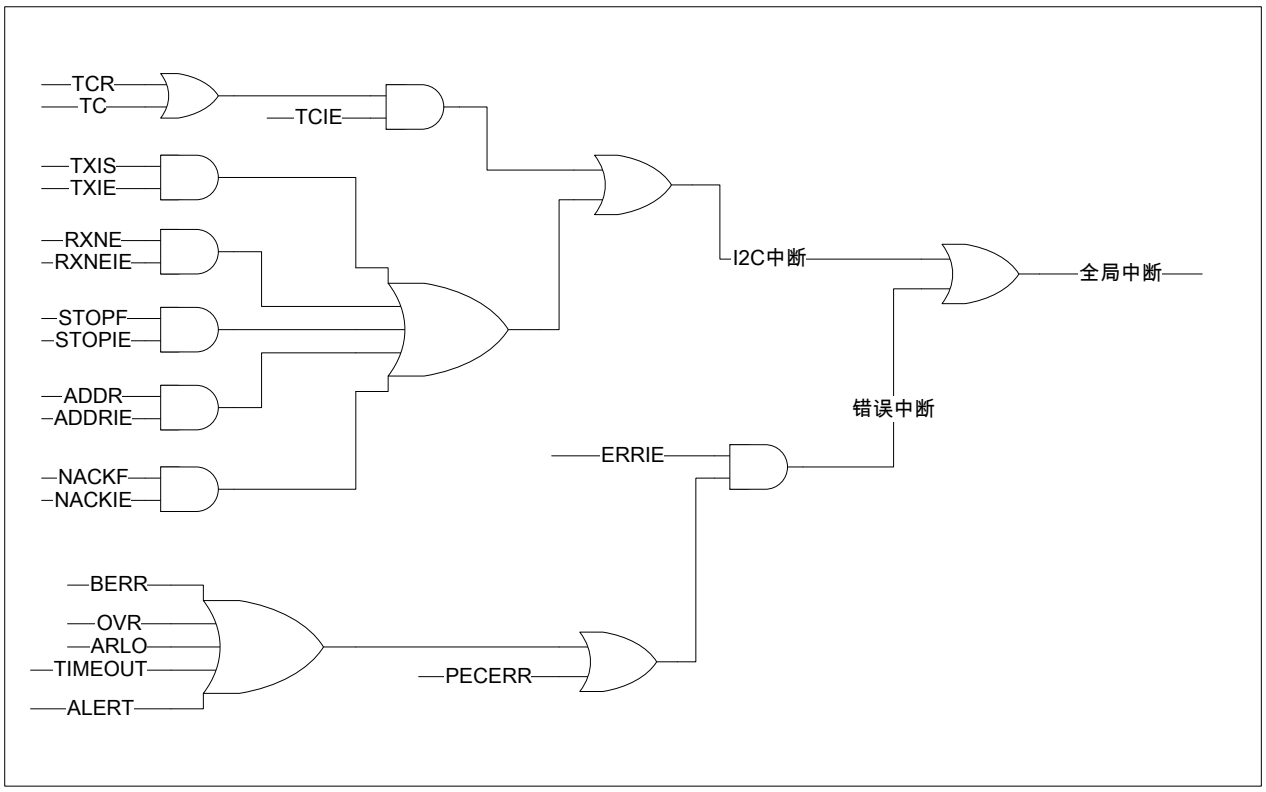


图 25.31 I2C 中断映射

25.3. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	I2C_CR1	–	–	–	–	–	–	–	–	PECEN	ALERTEBN	SMBDEN	SMBHEN	GCEN	–	NOSTRETCH	SBC	RXDMAEN	TXDMAEN	–	ANOFF	DNF[3:0]				ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	x	0	x	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	I2C_CR2	–	–	–	–	–	PECBYTE	AUTOEND	RELOAD	NBYTE[7:0]								NACK	STOP	START	HEAD10R	ADD10	RD_WRN	SADD[9:0]										
	Reset	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	I2C_OAR1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	OA1EN	–	–	–	–	OA1MODE	OA1[9:0]										
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0
0x0C	I2Cx_OAR2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	OA2EN	–	–	–	–	OA2MSK[2:0]			OA2[7:1]							–
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x
0x10	I2Cx_TIMINGR	PRESC[3:0]				–	–	–	–	SCLDEL[3:0]				SDADEL[3:0]				SCLH[7:0]								SCLL[7:0]							
	Reset	0	0	0	0	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	I2Cx_TIMEOUTR	TEXTEN	–	–	–	TIMEOUTB[11:0]												TIMOUTEN	–	–	TIDLE	TIMEOUTA[11:0]											
	Reset	0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	I2Cx_ISR	–	–	–	–	–	–	–	–	ADDCODE[6:0]							DIR	BUSY	–	ALERT	TIMEOUT	PECERR	OVR	ARLO	BERR	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	I2Cx_ICR	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	ALERTCF	TIMEOUTCF	PECCF	OVRCF	ARLOCF	BERRCF	–	–	STOPCF	NACKCF	ADDRCF	–	–	–
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	x	x	x
0x20	I2Cx_PECR	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	PECR[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x24	I2Cx_RXDR	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	RXDATA[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0
0x28	I2Cx_TXDR	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	TXDATA[7:0]							
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0

25.3.1. I2Cx_CR1

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	—	NOSTRETCH	SBC
类型	RW	RW	RW	RW	RW	RO-0	RW	RW
15:8	RXDMAEN	TXDMAEN	—	ANFOFF	DNF[3:0]			
类型	RW	RW	RO-0	RW	RW	RW	RW	RW
7:0	ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:24	NA	保留位，未定义
23	PECEN	包错误检查使能，用于 SMBus 功能 1：使能包错误检测 0：包错误检查未使能
22	ALERTEN	SMBus 警告使能 1：使能 SMBus 警告使能 0： SMBus 警告使能未使能
21	SMBDEN	SMBus 设备默认地址使能 1：设备地址使能，地址是 0b1100001x 0：设备地址不使能
20	SMBHEN	SMBus 主机地址使能 1：使能主机地址，地址是 0b0001000x 0：主机地址不使能
19	GCEN	广播地址使能 1：使能广播地址，地址是 0b00000000 0：广播地址未使能
18	NA	保留位，未定义
17	NOSTRETCH	时钟拉低扩展使能 1：使能时钟拉低使能 0：时钟拉低功能未使能
16	SBC	从机字节控制 1：使能从机字节控制 0：从机字节控制未使能
15	RXDMAEN	DMA 接收使能 1：使能 DMA 接收模式 0：DMA 接收模式未使能
14	TXDMAEN	DMA 发送使能 1：使能 DMA 发送模式

		0：DMA 发送模式未使能
13	NA	保留位，未定义
12	ANFOFF	模拟滤波器使能关闭 1：模拟滤波器未使能 0：使能模拟滤波器
11:8	DNF	数字滤波器，滤波时钟周期数 0：禁止数字滤波 x：滤除小于等于 x 个模块时钟长度的信号变化
7	ERRIE	错误中断使能 1：使能错误中断 0：错误中断未使能
6	TCIE	传输完成中断使能 1：使能传输完成中断 0：传输完成中断未使能
5	STOPIE	STOP 中断使能 1：使能 STOP 中断 0：STOP 中断未使能
4	NACKIE	收到 NACK 中断使能 1：使能收到 NACK 中断 0：收到 NACK 中断未使能
3	ADDRIE	地址匹配中断使能 1：使能地址匹配中断 0：地址匹配中断未使能
2	RXIE	接收非空中断使能 1：接收非空中断使能 0；接收非空中断未使能
1	TXE	发送未空中断使能 1：使能发送非空中断 0：发送非空中断未使能
0	PE	接口使能 1：接口使能 0；接口未使能

25.3.2. I2Cx_CR2

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—					PECBYTE	AUTOEND	RELOAD
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
23:16	NBYTES[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	NACK	STOP	START	HEAD10R	ADD10	RD_WRN	SADD[9:8]	

类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	SADD[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:27	NA	保留位，未定义
26	PECBYTE	包错误校验字节请求发送，PE 为 0 时，该位为零 1：请求发送/接收包错误校验字节，传输完成后该位自动清零 0：包错误校验字节未传输
25	AUTOEND	自动结束模式 1：使能自动结束模式，当 NBYTES 个数据传输完成后，STOP 条件自动发送 0：软件结束模式，NBYTES 个字节传输完成后 TC 标志位置 1，拉低 SCL
24	RELOAD	NBYTES 重载模式 1：NBYTES 个字节传输完成后，当前传输未完成，TCR 标志位置 1，拉低 SCL 0：NBYTES 个字节传输完成后，当前传输完成。
23:16	NBYTES	字节数，将要传输的字节数；从机模式 SBC=0 时，该为无效；START 位置 1 时，该位的值不允许修改
15	NACK	NACK 产生，从机模式 1：在当前数据接收完成后回复 NACK 0：在当前数据接收完成后回复 ACK
14	STOP	STOP 产生，主机模式 1：在当前字节传输完成后，发送 STOP 0：当前字节传输完成后，不发送 STOP
13	START	开始产生，改位在地址发送完成、仲裁丢失、超时和 PE=0 时清零。 1：RESTART/START 产生 0：没有 START 产生
12	HEAD10R	读方向 10 比特地址头，主机接收模式 1：主机只发送 10 比特地址低 7 比特地址+READ 0：主机发送完整的开始+完整的地址+RESTART+低 7 比特地址+READ
11	ADD10	10 比特地址，主机模式 1：主机工作在 10 比特模式 0：主机工作在 7 比特地址模式
10	RD_WRN	表示主机发起的是读操作还是写操作 1：主机发起读传输 0：主机发起写传输
9:0	SADD	从机地址，主机模式 SADD[9:0]，10 比特地址，在 7 比特地址时 SADD[9:8]和 SADD[0]不关心 SADD[7:1]，7 比特地址

25.3.3. I2Cx_OAR1

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OA1EN	—				OA1MODE	OA1[9:8]	
类型	RW	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
7:0	OA1[7:0]							
类型	RW							

Bit	Name	Function
31:16	NA	保留位，未定义
15	OA1EN	从机地址（本机地址）1 使能 1：从机地址 1 使能 0：从机地址 1 未使能
14:11	NA	保留位，未定义
10	OA1MODE	本机 10 比特地址模式 1：本机 10 比特地址模式 0：本机 7 比特地址模式
9:0	OA1	本地地址配置 10 比特地址时，OA1[9:8]用于本机 10 比特地址 7 比特地址时，OA1[7:1]用于本机 7 比特模式，OA1[9:8]和 OA1[0]，不关心

25.3.4. I2Cx_OAR2

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	OA2EN	—				OA2MSK[2:0]		
类型	RW	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
7:0	OA1[7:1]							—
类型	RW	RW	RW	RW	RW	RW	RW	RO-0

Bit	Name	Function
31:16	NA	保留位，未定义
15	OA2EN	从机地址（本机地址）2 使能 1：使能本机地址 2

		0：本机地址 2 未使能
14:11	NA	保留位，未定义
10:8	OA2MSK	地址 2 掩码 000：没有掩码 001：地址匹配时，OA2[1]不关心 010：地址匹配时，OA2[2:1]不关心 011：地址匹配时，OA2[3:1]不关心 100：地址匹配时，OA2[4:1]不关心 101：地址匹配时，OA2[5:1]不关心 110：地址匹配时，OA2[6:1]不关心 111：地址匹配时，OA2[7:1]不关心，所有 7 比特地址匹配(除了保留地址)
7:1	OA2	本机 7 位接口地址配置
0	NA	保留位，未定义

25.3.5. I2Cx_TIMINGR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	PRESC[3:0]				—			
类型	RW				RO-0	RO-0	RO-0	RO-0
23:16	SCLDEL[3:0]				SDADEL[3:0]			
类型	RW				RW			
15:8	SCLH[7:0]							
类型	RW							
7:0	SCLL[7:0]							
类型	RW							

Bit	Name	Function
31:28	PRESC	时钟预分频 T_{PRESC}=（PRESC+1）* T_{I2CCLK}
27:24	NA	保留位，未定义
23:20	SCLDEL	数据建立时间 T_{SCLDEL}=（SCLDEL+1）*T_{PRESC}
29:16	SDADEL	数据保持时间 T_{SDADEL}=SDADEL*T_{PRESC}
15:8	SCLH	SCL 高电平周期，主机模式 T_{SCLH}=（SCLH+1）*T_{PRESC}
7:0	SCLL	SCL 低电平时间，主机模式 T_{SCLL}=（SCLL+1）*T_{PRESC}

25.3.6. I2Cx_TIMEOUTR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	TEXTEN	—			TIMOUTB[11:8]			
类型	RW	RO-0	RO-0	RO-0	RW			
23:16	TIMOUTB[7:0]							
类型	RW							
15:8	TIMOUTEN	—		TIDLE	TIDLE[11:8]			
类型	RW	RO-0	RO-0	RW	RW			
7:0	TIDLE[7:0]							
类型	RW							

Bit	Name	Function
31	TEXTEN	扩展时钟超时使能 1：扩展时钟超时使能，当 SCL 被拉低超过 $T_{LOW:EXT}$ 时，TIMEOUT 会置 1 0：扩展时钟超时未使能
30:28	NA	保留位，未定义
27:16	TIMEOUTB	总线超时 B，$T_{LOW:EXT}$=(TIMEOUTB+1)*2048*T_{I2CCLK}
15	TIMOUTEN	时钟超时使能 1：SCL 被拉低超过设定的 T_{IDLE}/T_{TIMOUT} 时置 1 0：SCL 拉低检测关闭
14:13	NA	保留位，未定义
12	TIDLE	空闲时钟超时使能 1：TIMEOUTA 被用作检测 SCL 和 SDA 高电平超时（总线空闲） 0：TIMEOUTA 被用作 SCL 低电平超时检测
11:0	TIMEOUTA	TIDLE=0 时用于 SCL 拉低检测，T_{TIMOUT}=（TIMEOUTA+1）*2048*T_{I2CCLK} TIDLE=1 时用于总线空闲检测，T_{IDLE}=（TIMEOUTA+1）*4*T_{I2CCLK}

25.3.7. I2Cx_ISR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	ADDCODE[6:0]							DIR
类型	RO							RO
15:8	BUSY	—	ALERT	TIMEOUT	PECERR	OVR	ARLO	BERR
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:24	NA	保留位，未定义
23:17	ADDCODE	从机模式匹配到的地址 ADDR=1 时更新该字段的值；10 比特地址时，这里更新的是低 7 位的值
16	DIR	表示传输方向，从机模式 1：读方向，从机进入发送状态 0：写方向，从机进入接收状态
15	BUSY	忙标志位，PE=0 时自动清零 1：表示通信正在进行中 0：通信未正在进行
14	NA	保留位，未定义
13	ALERT	SMBus 警告标志位，PE=0 时自动清零 1：SMBHEN=1 和 ALERTEN=1 时，SMBALERT 引脚发生了低电平翻转，写 1 到 ALERTCF 清零 0：未产生 SMBus 警告标志位
12	TIMEOUT	超时表示位，PE=0 时自动清零 1：发送了超时，写 1 到 TIMEOUTCF 清零 0：未发生超时
11	PECERR	接收时发现了包检验错误，用于 SMBus 功能，PE=0 时自动清零 1：接收到的包校验错误码不匹配，写 1 到 PECCF 清零 0：未发生包校验码错误
10	OVR	从机模式发生溢出或者下溢状态，PE=0 时自动清零 1：从机模式 NOSTRETCH=1 时，发生了溢出或下溢，写 1 到 OVRCF 后清零 0：未发生溢出或者下溢
9	ARLO	仲裁丢失，PE=0 时自动清零 1：通信中发生了总线仲裁失败，写 1 到 ARLOCF 清零 0：未发生总线仲裁失败
8	BERR	总线错误，PE=0 时，自动清零 1：总线检测到了错误位置的停止位或者起始位，写 1 到 BERRCF 清零 0：未发生总线错误
7	TCR	RELOAD=1 时，NBYTES 个字节已经传输完成，PE=0 时自动清零 1：RELOAD 模式下，NBYTES 个字节已经传输完成，向 NBYTES 写入非零数据后，清零 0：RELOAD 模式下，未发生传输完成
6	TC	主机软件模式下，发送完成（RELOAD=0 和 AUTOEND=0）标志，PE=0 时自动清零 1：传输完成标志，软件置位 START 和 STOP 位时，清零 0：发送未完成
5	STOPF	停止位标志，PE=0 时自动清零 1：总线上检测到了 STOPF 标志位，写 1 到 STOPCF 清零 0：总线上未检查到 STOPF 标志
4	NACKF	表示没有接收到应答标志位，PE=0 时自动清零

		1：没有接收到应答标志 0：接收到了应答标志
3	ADDR	从机地址匹配到了标志，PE=0 时自动清零 1：从机匹配到了地址，写 1 到 ADDRCF 后清零 0：从机未匹配到地址
2	RXNE	接收非空标志，PE=0 时清零 1：RXDR 寄存器接收到了新的数据，读取 RXDR 寄存器后清零 0：未接收到新的数据
1	TXIS	发送数据为空，传输未结束，需要立即写入数据，PE=0 时，自动清零 1：发送寄存器为空，数据需要写入 TXDR，写入完成后自动清零；NOSTRETCH=1 时，向该位写 1 产生 TXIS 事件 0：数据发送寄存器不需要写入数据
0	TXE	发送数据寄存器为空 1：发送数据寄存器未空；向该位写 1 产生 TXE 事件 0：发送数据寄存器中有数据

25.3.8. I2Cx_ICR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—		ALERTCF	TIMEOUTCF	PECCF	OVRCF	ARLOCF	BERRCF
类型	RO-0	RO-0	WO	WO	WO	WO	WO	WO
7:0	—		STOPCF	NACKCF	ADDRCF	—		
类型	RO-0	RO-0	WO	WO	WO	RO-0	RO-0	RO-0

Bit	Name	Function
31:14	NA	保留位，未定义
13	ALERTCF	警告标志位清零 1：清零 ALERT 标志位 0：不对 ALERT 标志位进行操作
12	TIMOUTCF	超时标志位清零 1：清零 TIMEOUT 标志 0：不对 TIMEOUT 标志进行操作
11	PECCF	包错误标志位清零 1：清零 PECERR 标志 0：不对 PECERR 标志进行操作
10	OVRCF	清零溢出或者下溢标志位

		1：清零 OVR 标志位 0：不对 OVR 标志位进行操作
9	ARLOCF	清零仲裁丢失标志位 1：清零 ARLO 标志位 0：不对 ARLO 标志进行清零
8	BERRCF	清零总线错误标志 1：清零 BERRF 标志位 0：不对 BERRF 标志位进行操作
7:6	NA	保留位，未定义
5	STOPCF	清零停止标志位 1：清零 STOPF 标志 0：不对 STOPF 标志进行操作
4	NACKF	清零未收到应答标志位 1：清零 NACKF 标志位 0：不对 NACKF 标志位进行操作
3	ADDRCF	清零地址匹配标志 1：清零 ADDRF 标志 0：不对 ADDRF 进行操作
2:0	NA	保留位，未定义

25.3.9. I2Cx_PECR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	PEC[7:0]							
类型	RO							

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	PEC	包错误校验码，PECEN=1 时，内部计算的 CRC 校验码

25.3.10. I2Cx_RXDR

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

31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	RXDATA[7:0]							
类型	RO							

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	RXDATA	接收数据寄存器

25.3.11. I2Cx_TXDR

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

Bit	Name	Function
31:8	NA	保留位，未定义
7:0	TXDATA	发送数据寄存器

26. 异步同步通信接口（USART）

26.1. 主要特性

- 支持全双工模式
- 非归零(NRZ)标准格式
- 可配置的16或8倍过采样方法提供速度和时钟容忍度间的灵活选择
- 一个公共的可编程收发波特率高达9Mbit/s（时钟频率为72MHz，过采样率为8倍时）
- 方便的波特率改变
- 自动波特率检测
- 数据字长可编程（8或9位）
- 高位在前或低位在前可设置
- 停止位个数可设置，支持1个或2个停止位
- 同步模式下时钟输出功能，实现同步通讯
- 单线半双工通讯
- DMA（直接内存访问）支持下的连续数据通讯
  ——利用DMA功能将收/发字节缓冲到保留的SRAM空间
- 针对接收器和发送器的单独使能
- 针对发送和接收的单独的信号极性控制
- 可配置为发送/接收引脚互换
- 用于MODEM的硬件流控制和RS-485发送使能控制
- 通信中的控制/错误标志位产生
- 奇偶校验控制
  ——发送奇偶校验位
  ——接收时校验奇偶校验位
- 十四个中断源和中断标志
- 多机通信，如果地址不匹配则进入静默模式
- 从静默模式唤醒（检测到线路空闲或检测到地址标记）

26.2. 功能描述

26.2.1. 功能实现比对

本文描述了 USART1 所实现的全套功能。USART2 功能稍有裁剪，但有的功能都和 USART1 一样；差异如下表所示。

表 26.1 USART 功能实现比对

USART 功能	USART1	USART2
MODEM 所需的硬件流控制	√	√

用 DMA 实现连续通讯	√	√
多机通信	√	√
同步模式	√	√
半双工模式	√	√
接收超时中断	√	-
自动波特率检测	√	-
RS485 用的驱动使能信号	√	√

26.2.2. 功能描述

接口与外部设备通过三个引脚相连（如图 26.1），任何 USART 双向通讯要求最少有两个引脚：接收数据输入（RX）和发送数据输出（TX）

RX:接收数据输入是串行数据的输入口。使用过采样技术来完成数据恢复，以区别输入数据和噪声。

TX:数据发送输出。当发送器被禁止，输出脚回到其 I/O 口配置状态。当发送器被使能，但不发送数据时，TX 脚为高电平输出。在单线半双工模式中，这个口线既用于发送数据也用于接收数据。

通过这些引脚，异步串行数据用数据帧的形式发送和接收：

- 在发送和接收之前为空闲状态
- 起始位
- 数据字（8或9位）最低有效位在前
- 1,1.5,2个停止位表明帧的结束
- 采用小数波特率发生器，整数12位小数4位
- 一个状态寄存器（USART_ISR）
- 分开的接收和发送数据寄存器（USART_RDR，USART_TDR）
- 一个波特率寄存器（USART_BRR）12位整数和4位小数

下面的引脚在同步模式中会用到：

- SCLK:时钟输出。该引脚会输出发送数据的同步时钟，发送功能和SPI主模式一致（起始位和停止位上没有时钟脉冲，在最后一位数据上可选择有无时钟脉冲）。同时，数据可经由RX引脚同步接收，这可以被用来连接那种通过移位寄存器相连的外设（例如：LCD驱动器）。时钟相位和极性可软件设定。

下列引脚用于支持硬件流控制模式：

- nCTS:低发送，当高电平时作为发送阻塞信号。
- nRTS:请求发送，表明USART已经准备好接收数据（低电平的时候）

下列引脚在RS485驱动使能控制的时候会用到：

- DE:驱动使能将外部收发器的发送模式激活。

注：DE 和nRTS 共用同一个外部引脚。

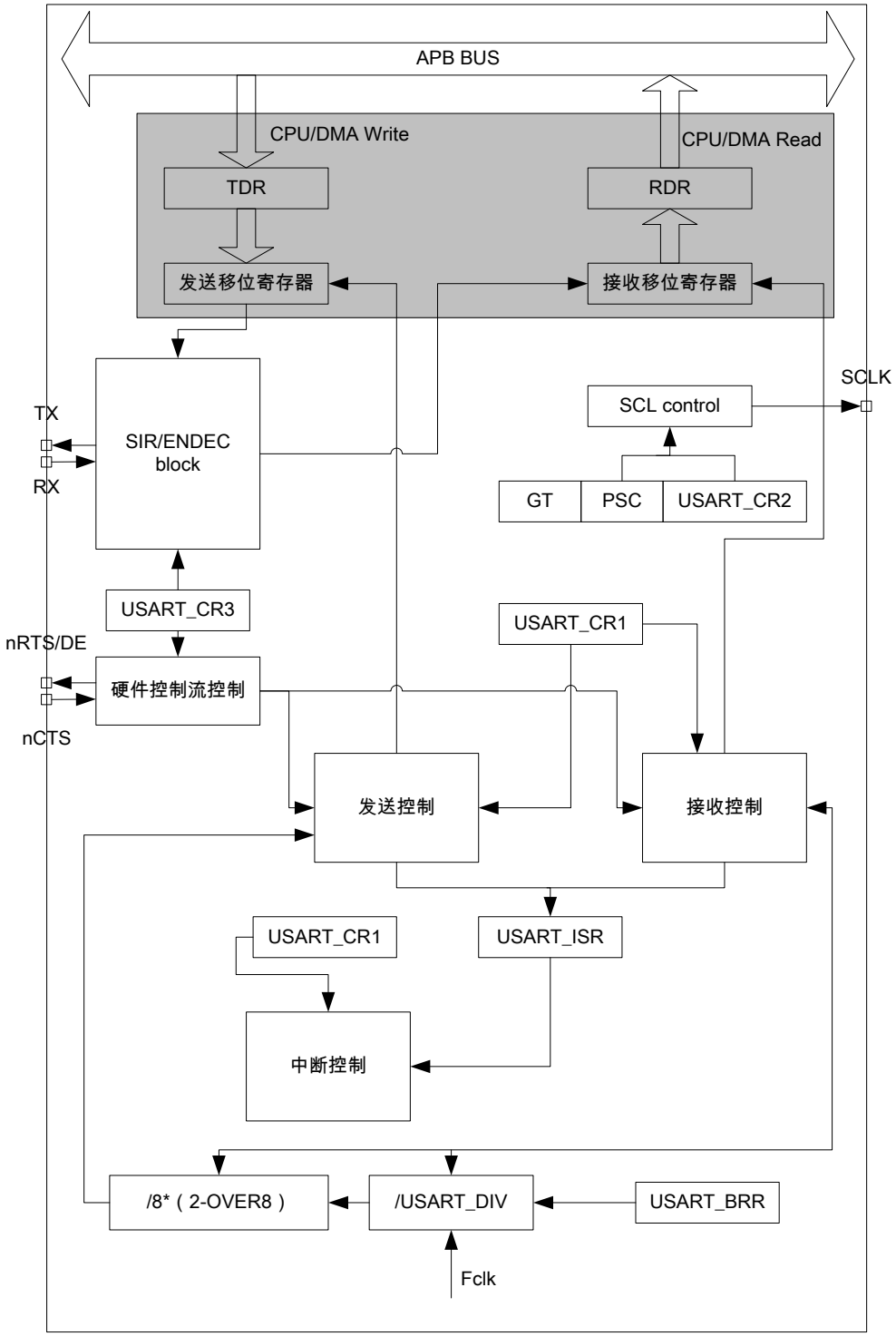


图26.1 USART框图

26.2.3. USART 符号描述

配置USART_CR1寄存器中的M0位可选择8位或9位字长。

- 8比特字符长度：M0=0
- 9比特字符长度：M0=1

默认设置中 ,发送和接收的起始位都是低电平。而停止位都是高电平。默认逻辑可以在极性控制中设置为反向。

空闲符号被视为完全由‘1’组成的完整的数据帧（‘1’的个数也包括了停止位的位数)。

断开符号被视为在一个帧周期内全部收到‘0’包括停止位期间)。在断开帧结束时，发送器会再插入2个停止位。

发送和接收由一个共用的波特率发生器驱动，当发送器和接收器的使能位分别置1时，分别为其产生时钟。

下面是每个模块的详细说明。

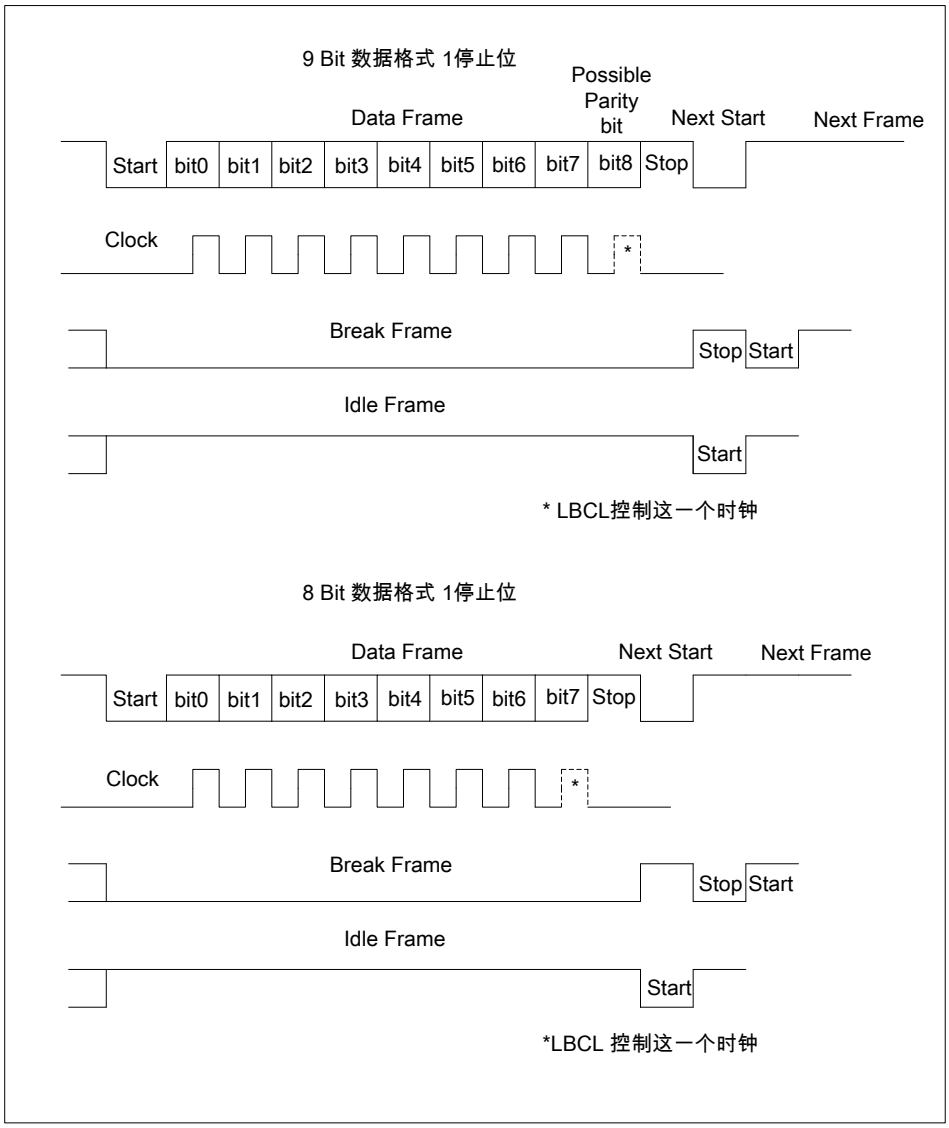


图26.2 字符长度设置

26.2.4. 发送器

发送器根据M 位的状态发送8位或9位的数据字。发送使能位必须置1以打开发送功能。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。

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。在此模式里，USART_TDR寄存器充当了一个内部总线和发送移位寄存器之间的缓冲器(TDR)。每个字符之前都有一个低电平的起始位. 之后跟着停止位，停止位的数目是可选择的。USART支持多种停止位的选择：1、1.5 和2个停止位。

注：1. 在向USART_TDR写数据之前必须先值TE位为1，在发送数据时，禁止清零TE位，清零TE会导致数据发送不完整。

2. 在TE位被置1后将会发送一个空闲帧。

可配置的停止位

随每个字符发送的停止位的位数可以通过控制寄存器CR2的第12和13位进行编程。

1. 1个停止位：停止位的位数的默认值。
2. 2个停止位：可用于常规USART 模式、单线半双工模式以及调制解调器模式。

空闲帧包括了停止位。

断开帧是10位低电平(当m=0时)；或者11位低电平(m=1时)，后跟2个停止位。没办法传输更长的断开帧(长度大于10或者11位的那种)。

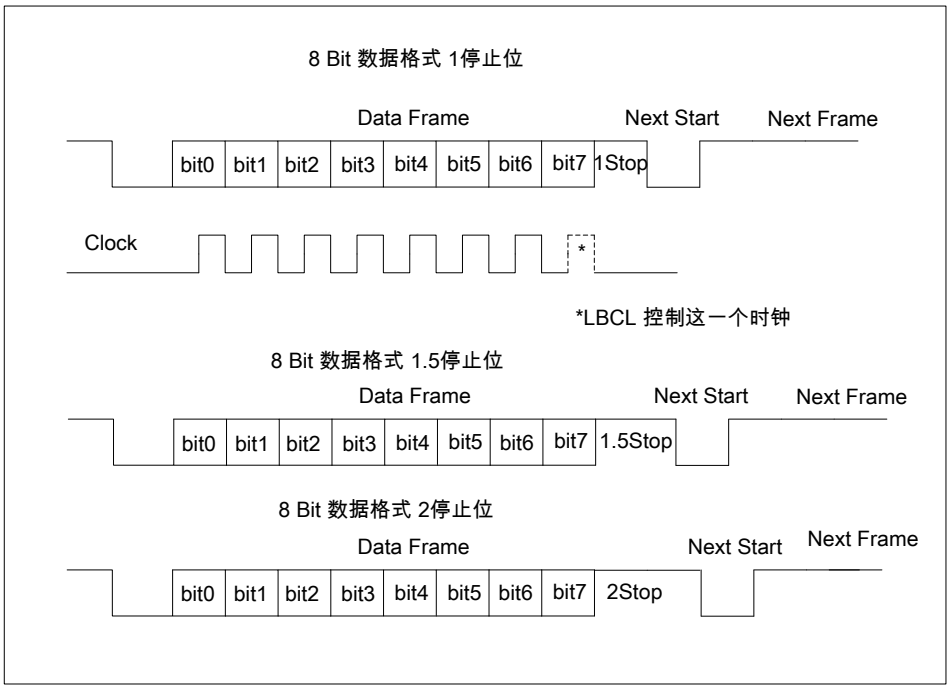


图26.3 停止位长度设置

配置步骤：

1. 设置USART_CR1的M0位来定义字长。
2. 利用USART_BRR寄存器选择希望的波特率。
3. 在USART_CR2中设置停止位的位数。
4. 通过将USART_CR1寄存器的UE位置1来使能USART。
5. 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。
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_TDR寄存器(此动作将清除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。重复步骤7之前应等待TXE变成1。
8. 在USART_TDR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或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次传输。

单字节通信

清零TXE位总是通过对数据寄存器的写操作来完成的。TXE位由硬件来设置，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。
- USART_TDR寄存器被清空。
- 下一个数据可以被写进USART_TDR寄存器而不会覆盖先前的数据。

如果TXEIE 位被设置，此事件将产生一个中断请求。

如果此时USART正在发送数据，对USART_TDR寄存器的写操作把数据存进TDR寄存器，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_TDR寄存器的写操作将导致直接把数据放进移位寄存器，数据传输开始，TXE位则立即被置起。

当一个字节发送完成时(停止位发送后)并且之前TXE位已经置1时，TC位会被置1。如果USART_CR1寄存器中的TCIE位是1时，则会产生中断。

在USART_TDR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（见图26.4）。

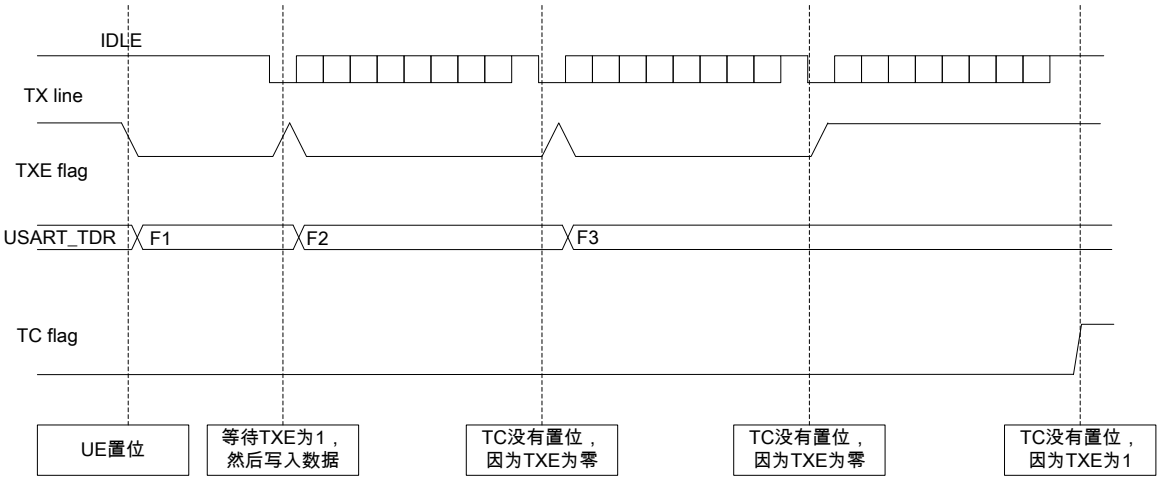


图26.4 发送时TC/TXE的时序

断开符号

向SBKRQ位写1可发送一个断开符号。断开符号的长度取决于M0位(见图26.2)。

如果设置SBK=1，在完成当前数据发送后，将在TX线上发送一个断开符号。SBKF位会在写操作发生时置1，在断开符号发送完成时(在断开符号的停止位发出时)被硬件清零。USART在最后一个断开帧的结束处插入一个逻辑‘1’并保持2位时长作为停止，以保证能识别下一帧的起始位。

空闲符号

将TE置1将使得USART在第一个数据前发送一空闲符号。

26.2.5. 接收器

接收器根据USART_CR1寄存器中M0位的状态接收8位或9位的数据字。

起始位侦测

过采样率设置成16或8，不影响起始位侦测的顺序。

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：1110X0X0X0X0X0X0。

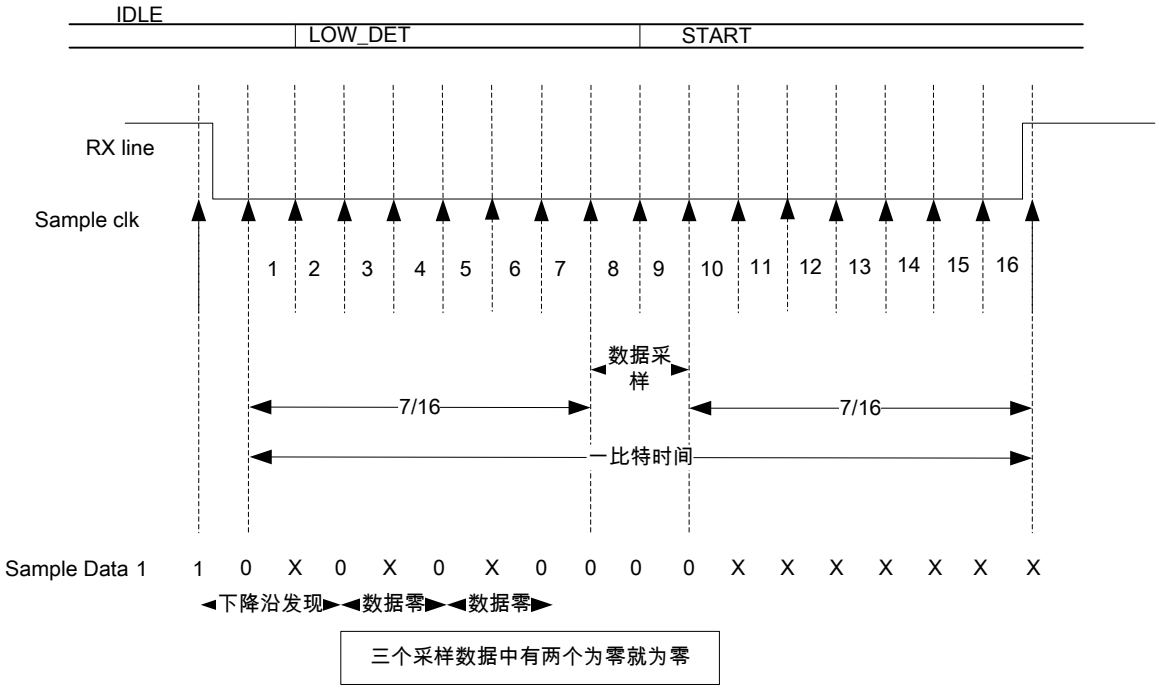


图26.5 接收起始位的侦测时序图

注：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)开始等待下降沿。

字符接收

在USART接收期间，数据的最低有效位（默认情况下）首先从RX脚移进。在此模式里，USART_RDR寄存器充当了一个位于内部总线和接收移位寄存器之间的缓冲器。

配置步骤：

1. 设置USART_CR1的M0位来定义字长。
2. 利用波特率寄存器USART_BRR选择希望的波特率。
3. 在USART_CR2中设置停止位的位数。
4. 通过将USART_CR1寄存器的UE位置1来激活USART。
5. 如果采用多缓冲器通信，配置USART_CR3中的DMA使能位(DMAR)。按多缓冲器通信中的描述配置DMA寄存器。
6. 将USART_CR1的RE位置1。这将激活接收器，使它开始寻找起始位。

当一个字符被接收到时

- RXNE位被置1。它表明移位寄存器的内容被转移到RDR。换句话说，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)。
- 这时如果RXNEIE位是1，将会引起中断请求。
- 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。PE标志也会和RXNE一起被置1。
- 在多缓冲器通信时，RXNE在每个字节接收后被置起，并由DMA对数据寄存器的读操作而清零。
- 在单缓冲器模式里，由软件读USART_RDR寄存器完成对RXNE位清除。RXNE标志也可以通过对USART_RQR寄存器中的RXFRQ位写1来清除。RXNE位必须在下一字符接收结束前被清零，以避免溢出错误。

断开符号

当接收到一个断开帧时，USART会像处理帧错误一样处理它。

空闲符号

当一空闲帧被检测到时，其处理步骤和接收到普通数据帧一样，但如果IDLEIE位为1，将产生一个中断请求。

溢出错误

如果RXNE还没有被复位，而又接收到了一个字符，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标记是接收到每个字节后被置1的。如果下一个数据已被收到或先前DMA请求还没被服务时，RXNE标志仍是置起的，也会产生溢出错误。当溢出错误产生时：

- ORE位被置1。
- RDR内容将不会丢失。读USART_RDR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位被置为1或EIE位被置为1，会产生中断。
- ORE位可以通过将ICR寄存器中的ORECF位置1的方式清除。

注：当ORE位置1时，表明至少有1个数据已经丢失。有两种可能性：

如果RXNE=1，尚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已经没有东西可读。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，此种情况是可能发生的。

选择时钟源和适当的过采样率的方法

时钟源的选择要通过时钟控制系统（参见第七章：复位和时钟控制系统）。

必须在使能USART之前（将UE位置1）从而打开USART的时钟源。

时钟源的选择需要依据两个标准：

- 在低功耗模式下使用串口的可能性
- 通讯速度

时钟源的频率是f_{CK}。

通讯速度范围（特别是最大通讯速度）是由所选的时钟源来决定的。

接收器根据用户设定的过采样率来执行数据恢复，从而将接收数据和噪声区分开来。如果设成同步模式则不会是这样。这需要在最大通讯速度和噪声抗扰度及对波特率偏差的敏感度之间做取舍。

通过USART_CR1寄存器中的OVER8位来选择过采样率是波特率时钟的8倍或者是16倍（如图26.6和26.7所示）。

根据应用：

- 选择8倍过采样以实现较高速度（高至$f_{CK}/8$）。这种情况下最大的接收器允许的波特率偏差要小一些
- 选择16倍过采样率(OVER8=0)可提高时钟偏差容忍度。这种情况下，最高通讯速度就会被限制在$f_{CK}/16$，

其中，f_{CK}就是时钟源的频率。

USART_CR3中的ONEBIT位用来选择判断逻辑电平的方法。有两种选择：

- 根据接收数据位中央的三个采样结果进行多数票决。这种情况下，当发现三个参与票决的采样结果不等时，NF位会被置1
- 根据接收数据位中央的单个置采样结果来直接裁决

根据应用：

- 在噪声干扰环境中应该选择三个采样多数票决的方式，可以排除检测到噪声的数据，因为那说明在数据采样的时候有干扰。
- 在不担心噪声的情况下，应选择单采样裁决（ONEBIT=1），可以提高接收器对时钟偏差的容忍度这种情况下NF位永远都不会置1。

当一帧数据中检测到噪声时，NF位会被置1；只要USART_CR3 中的EIE位是高就会产生中断，将ICR寄存器中的NFCF位置1 就可以清除NF标志位。

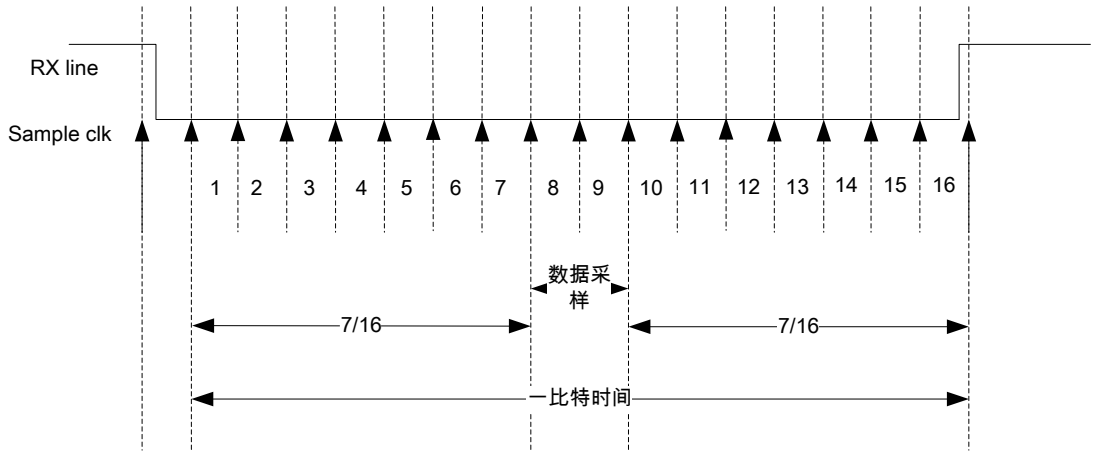


图26.6 16倍过采样时的数据采样

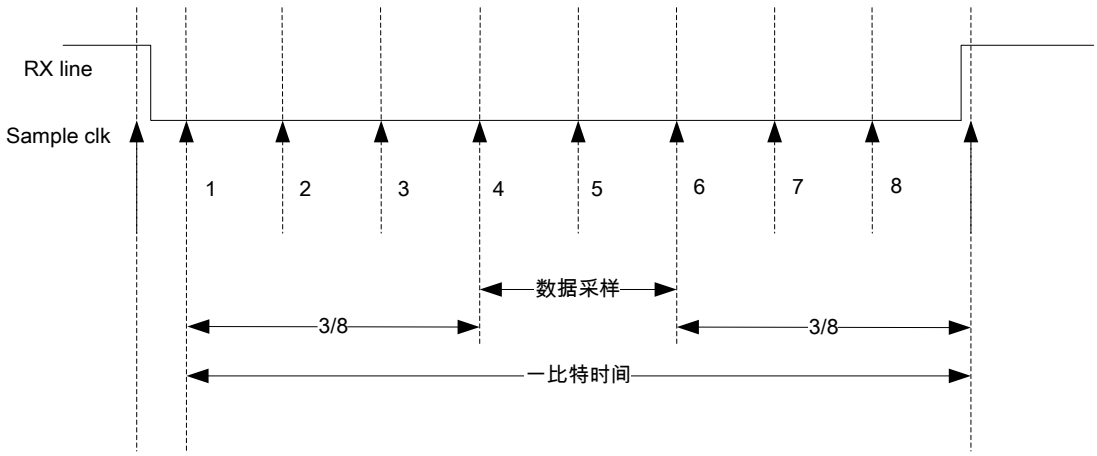


图26.7 8倍过采样时的数据采样

表26.2 从采样数据中检测噪声

采样值	NF位	采到的值
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

帧错误

当以下情况发生时检测到帧错误：

由于同步亚稳态或大量噪音的原因，停止位没有在预期的时间上接收和识别出来。

当帧错误被检测到时，FE位被硬件置1；只要USART_CR3中的EIE位是高就会产生中断。USART_ICR寄存器中的FECF置1就可以清除FE标志位。

接收期间的可配置的停止位

被接收的停止位的个数可以通过USART_CR2的控制位来配置，在正常模式时，可以是1或2个。

- 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。
- 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。在第一个停止位期间RXNE标志将被置位。

26.2.6. 波特率的产生

接收器和发送器的波特率在 USARTDIV 的整数和小数寄存器中的设置的值是相同的。

标准 USART（包括 SPI 模式）的波特率

$$\text{Tx or Rx Braud} = \frac{f_{\text{ck}}}{8 \times (2 - \text{OVER8}) \times \text{USARTDIV}}$$

USARTDIV 是一个无符号的定点数。这 12 位的值设置在 USART_BRR 寄存器。

- 当 OVER8=0，小数部分按 4 位计算，设置在 USART_BRR 寄存器的低四位中，此时 USARTDIV=USART_BRR。
- 当 OVER8=1 时，小数部分只有 3 位有效，设置在 USART_BRR 寄存器的低三位中，第四位必须保持为 0，此时 USARTDIV[11:4]=USART_BRR[11:4]，USART_BRR[2:0]=USARTDIV[3:1]，USART_BRR[3]固定为零。

注：在写入 USART_BRR 之后，波特率计数器会立即被波特率寄存器的新值替换。因此，不要在通信过程中改变波特率寄存器的数值。同时 USARTDIV 配置的值要不小于 16。

如何从 USART_BRR 寄存器值得到 USARTDIV

例 1：8M 工作时钟下配置 9600 波特率

- 过采样频率是 16 时：

  USARTDIV=8000000/9600

  BRR=USARTDIV=800d=0341h

- 过采样频率是 8 时：

  USARTDIV=2*8000000/9600=1666.66(1667d=683h)

  BRR[3:0]=3h>>1=1h

  BRR=0x681

例 2：48M 工作时钟下配置 921600 波特率

- 过采样频率是 16 时：

  USARTDIV=48000000/921600

  BRR=USARTDIV=52d=34h

- 过采样频率是 16 时：

  USARTDIV=2*48000000/921600=104(104d=68h)

  BRR[3:0]= USARTDIV[3:0]>>1 = 8h>>1=4h

  BRR=0x64

表26.3 f_{ck}=48MHz,过采样率是16和8时，设置波特率的误差计数

波特率		过采样为16			过采样为8		
序号	目标值	实际值	BRR	误差%	实际值	BRR	误差%
1	2.4kBps	2.4kBps	0x4E20	0	2.4kBps	0x9C40	0
2	9.6kBps	9.6kBps	0x1388	0	9.6kBps	0x2710	0
3	19.2kBps	19.2kBps	0x9C4	0	19.2kBps	0x1384	0
4	38.4kBps	38.4kBps	0x4E2	0	38.4kBps	0x9C2	0
5	57.6kBps	57.62kBps	0x341	0.03	57.59kBps	0x681	0.02
6	115.2kBps	115.11kBps	0x1A1	0.08	115.25kBps	0x340	0.04
7	230.4kBps	230.76kBps	0xD0	0.16	230.21kBps	0x1A0	0.08
8	460.8kBps	461.54kBps	0x68	0.16	461.54kBps	0xD0	0.16
9	921.6kBps	923.07kBps	0x34	0.16	923.07kBps	0x64	0.16
10	2MBps	2MBps	0x18	0	2MBps	0x30	0
11	3MBps	3MBps	0x10	0	3MBps	0x20	0
12	4Mbps	NA	NA	NA	4MBps	0x14	0
13	5MBps	NA	NA	NA	5052.63kBps	0x11	1.05
14	6MBps	NA	NA	NA	6MBps	0x10	0

注：CPU 的时钟频率越低，则某一特定波特率的精度也越低。

26.2.7. 接收器对时钟的容忍程度

只有当整体的时钟系统的变化小于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<USART接收器的容忍度
对于正常接收数据，USART接收器的容忍度等于最大能容忍的变化，它依赖于下述选择：

- 由USART_CR1寄存器的M0位定义的10或11位字符长度
- USART_CR1寄存器的OVER8位定义的，过采样率8位16位的选择
- 小数波特率的使用
- USART_CR3寄存器的ONEBIT位定义的，是1位采样还是3位采样

表26.4 小数波特率(BRR[3:0]=0)为零时的，串口接收容忍度

M0位	OVER8=0		OVER8=1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.75%	4.375%	2.5%	3.75%
1	3.41%	3.97%	2.27%	3.41%

表26.5 小数波特率(BRR[3:0]!=0)不为零时的，串口接收容忍度

M0位	OVER8=0		OVER8=1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.33%	3.88%	2%	3%
1	3.03%	3.53%	1.82%	2.73%

注：在接收数据格式为10 位时长（M=0时）或11位时长（M=1时）时，表26.3和表26.4中的数据可能有一些轻微的偏差。

26.2.8. 自动波特率检测

USART可以根据接收到的一个字符来检测和自动设置USART_BRR寄存器的值。自动波特率检测在两种情况下很有用：

- 通讯速度不可知的情况下
- 使用低精度时钟源，需要在不测量时钟偏差的条件下纠正波特率的时候

时钟源的频率必须和预期的波特率保持相对的稳定（过采样率为16，并且波特率处于f_{CK}/65535和f_{CK}/16之间）。

在打开自动波特率检测之前，字符的内容必须先确认。有两个可能的字符内容，能够通过USART_CR2寄存器的ABRMOD[1:0]域进行选择。具体是：

- 以1开头的任何字符。这种情况下USART将测量起始位的长度（下降沿到上升沿）。
- 以10xx开头的任何字符。这种情况下USART将测量第一个数据位的长度。测量下降沿到下降沿的时长，在小信号摆率的时候可以确保更好的测量精度。

在打开波特率自动检测之前，USART_BRR寄存器必须先初始化为一个不为零的波特率值。

如果线路噪声严重，不能保证得到的波特率是准确的。这时BRR值可能是错的或者ABRE错误标志会被置1。在通讯速度超出自动波特率检测范围（位长度不在8(OVER8=1)/16(0VER8=0)到65536个时钟周期之间）时也会发生这种情况。

RXNE的中断也会在数据接收完成之后产生。

随后，可以将ABRF标志清除掉，以重启自动波特率检测（如果不需要重新检测波特率可以不用清除ABRF标志位）。

注：如果在自动波特率操作期间将USART关掉（UE=0），那BRR的检测结果可能就不可信了。

26.2.9. 多机通信

可以将多个 USART 连接成一个网络来实现多机通讯。例如某个 USART 设备可以是主，它的 TX 输出和其他 USART 从设备的 RX 输入相连接；USART 从设备各自的 TX 输出按逻辑与在一起，并且和主设备的 RX 输入相连接。

在多处理器配置中，我们通常希望只有被寻址的接收者才被激活，来接收随后的数据，这样就可以减少由未被寻址的接收器的参与带来的多余的 USART 服务开销。

未被寻址的设备可启用其静默功能进入静默模式。要使用静默模式功能，USART_CR1 寄存器的 MME 位必须被置 1。

在静默模式里：

- 任何接收状态位都不会被置 1。
- 所有接收中断事件不会产生。
- 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通过 USART_RQR 寄存器的 MMRQ 位写入。

根据 USART_CR1 寄存器中的 WAKE 位状态，USART 可以用二种方法退出静默模式。

- 如果 WAKE 位为 0：进行空闲总线检测。
- 如果 WAKE 位为 1：进行地址标记检测。

空闲总线检测(WAKE=0)

当 MMRQ 位被置 1，RWU 会被自动置 1 时，USART 进入静默模式。当检测到空闲帧时，会唤醒（如图 26.8 所示）；

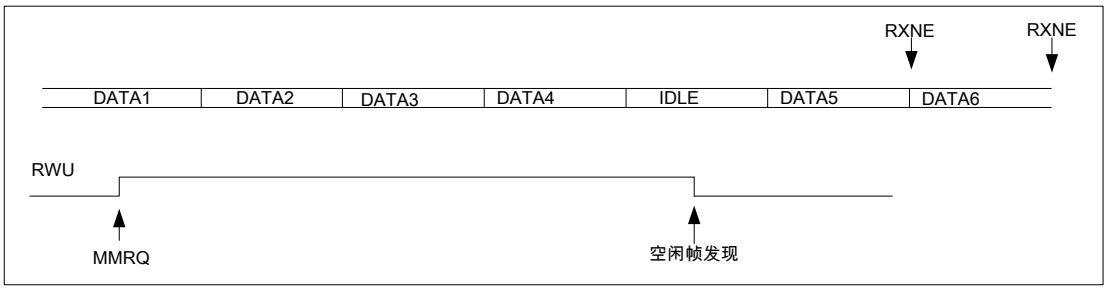


图 26.8 利用空闲帧从静默模式唤醒

4-bit/7-bit 地址标记检测 (WAKE=1)

在这个模式里，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。在一个地址字节中，目标收器的地址被放在 4 个或 7 个位的位域中。用 ADDM7 位来选择是用 4 位地址还是用 7 位地址。这个 4 位或 7 位地址被接收器同它自己地址做比较，接收器的地址被设置在 USART_CR2 寄存器的 ADD 域。

注：在 7 位和 9 位数据模式下，地址检测分别按 6 位和 8 位地址(ADD[5:0]和 ADD[7:0])操作。

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，硬件将 RWU 位置 1。当 USART 进到静默模式后，接收字节时既不会动 RXNE 标志也不会产生中断或发出 DMA 请求。

将 MMRQ 置 1 也会令 USART 进入静默模式。这时 RWU 位也被自动置 1。如果接收到的字节与它的编程地址匹配，USART 将退出静默模式。然后 RWU 位被清零，后续的字节会被正常接收。从 RWU 位被清零开始，RXNE 位会因为地址字节的接收而被置 1（如图 26.9 所示）。

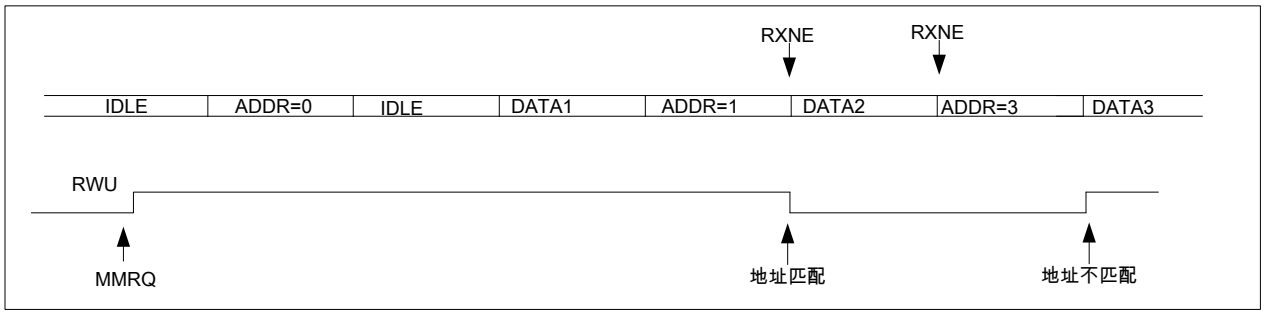


图 26.9 利用地址匹配从静默模式唤醒

26.2.10. 校验控制

设置 USART_CR1 寄存器上的 PCE 位，可以使能校验控制(发送时生成一个校验位，接收时进行校验检查)。根据 M0 位定义的帧长度，可能的 USART 帧格式列在下表中。

表 26.6 帧格式

M0 位	PCE	USART 帧格式
0	0	起始位+8 位数据+停止位
0	1	起始位+7 位数据+奇偶位+停止

		位
1	0	起始位+9 位数据+停止位
1	1	起始位+8 位数据+奇偶位+停止位

偶校验

校验位的数据使得一帧中的数据和校验位中的‘1’的个数为偶数个（PS 置 0）。

奇校验

校验位的数据使得一帧中的数据和校验位中的‘1’的个数为奇数个（PS 置 1）。

接收时的校验检查

如果校验检查失败，USART_ISR 寄存器中的 PE 标志会被置 1，如果 USART_CR1 寄存器中的 PEIE 为 1，将引发相应中断。由软件向 USART_ICR 寄存器的 PECF 位写 1，可以清除 PE 标志。

发送时的校验生成

如果 USART_CR1 的 PCE 位被置 1，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去。

26.2.11. 同步模式

通过在 USART_CR2 寄存器的 CLKEN 位上写 1 来选择同步模式在同步模式下，下列位必须保持为 0：

- USART_CR3 寄存器中的 HDSEL 位

USART 允许用户以主模式方式控制双向同步串行通信。SCLK 脚是 USART 发送器时钟的输出。在起始位和停止位期间，SCLK 脚上没有时钟脉冲。USART_CR2 寄存器中 LBCL 位的状态决定了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（如图 26.10 和 26.11 所示）。

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 SCLK 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 SCLK 与 TX 同步(根据 CPOL 和 CPHA)，所以 TX 上的数据是随 SCLK 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 SCLK 上采样(根据 CPOL 和 CPHA 决定在上升沿还是下降沿)，不需要任何的过采样。但必须考虑建立时间和持续时间。

注：SCLK 脚同 TX 脚是协同工作的。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被禁止时（UE=0）去改变 LBCL,CPOL 和 CPHA 位的配置.这能保证时钟脉冲功能的正确性。

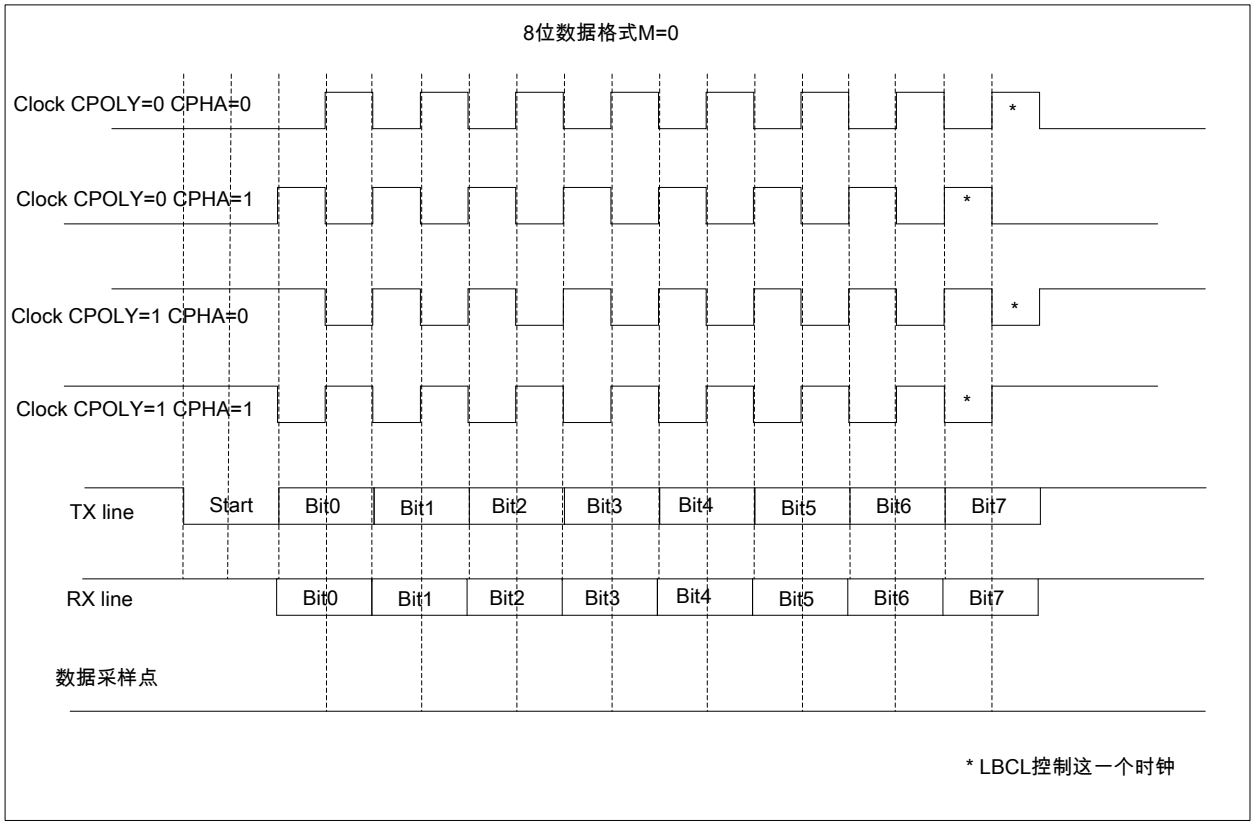


图 26.10 同步模式时序图 M=0

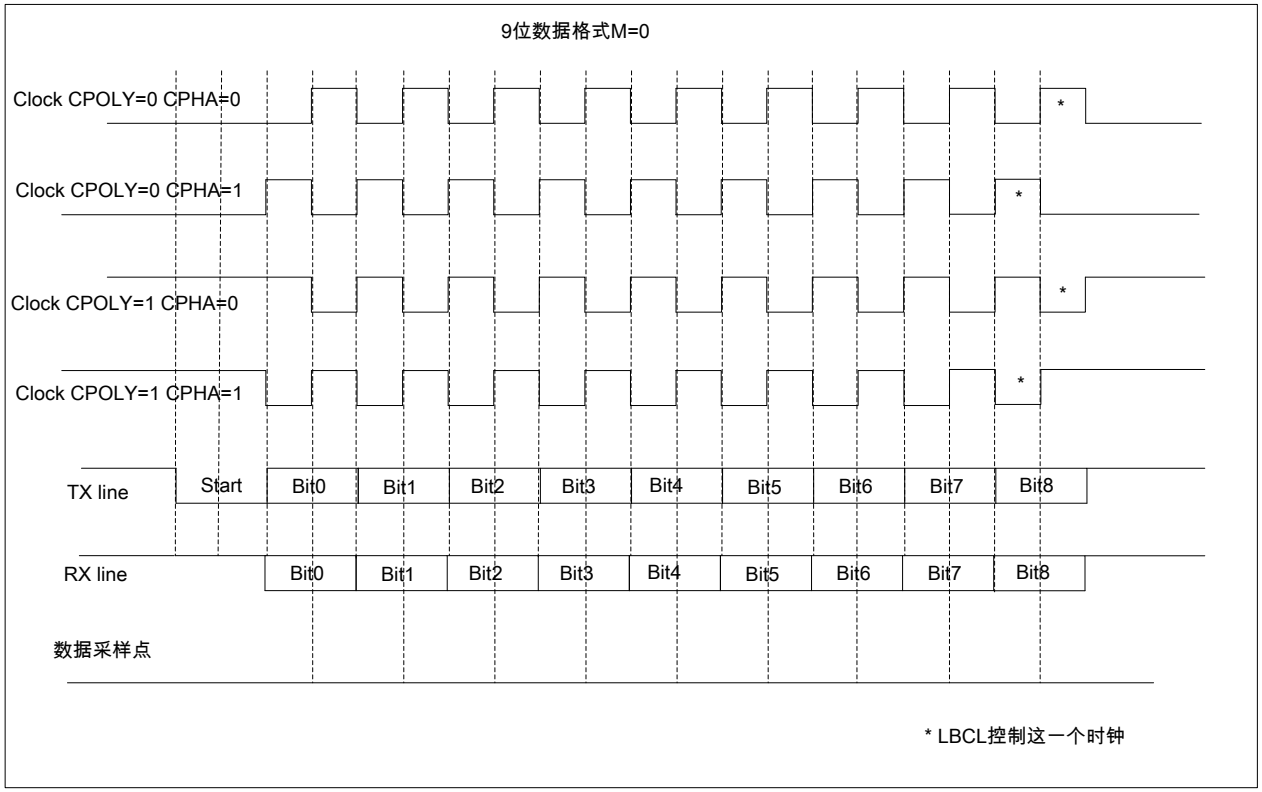


图 26.11 同步模式时序图 M=1

26.2.12. 单线半双工模式

单线半双方模式通过设置 USART_CR3 寄存器的 HDSEL 位选择。在本模式下，下列位必须保持为 0：

- USART_CR2 寄存器 CLKEN 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 引脚在芯片在内部是连在一起的。使用控制位 HDSEL(USART_CR3 中的 HDSEL 位)选择半双工通信。

当 HDSEL 为‘1’时

- TX 和 RX 引脚在芯片在内部是连在一起的
- RX 不再被使用
- 当没有数据传输时，TX 总是被释放。因此，它在空闲状态的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 ,必须配置成悬空输入(或开漏的输出高)。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上，发送就会开始。

26.2.13. 用 DMA 实现连续通讯

USART 可以利用 DMA 连续通信。RX 缓冲器和 TX 缓冲器的 DMA 请求是分别产生的。

利用 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。在设置 TXE 位之前，数据被预先放到 DMA 外设所设定的 SRAM 区域设置一个 DMA 通道给 USART 发送，要使用下列步骤：

1. 在 DMA 控制寄存器上将 USART_TDR 寄存器地址配置成 DMA 传输的目的地址。在每个 TXE 事件后，数据将被传送到这个地址。
2. 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的源地址。在每个 TXE 事件后，将从此存储器区读出数据并传送到 USART_TDR 寄存器。
3. 在 DMA 控制寄存器中配置要传输的总的字节数。
4. 在 DMA 寄存器上配置通道优先级。
5. 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。
6. 将 USART_ICR 寄存器的 TCCF 位置 1 以清除 USART_ISR 寄存器的 TC 标志。
7. 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；监视 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 或进入停机模式之前避免破坏最后一次传输的数据。软件必须等待 TC 被置 1。TC 标志在全部数据发送期间会是零，并且在最后一帧数据发送出去之后会由硬件置 1（如图 26.12 所示）。

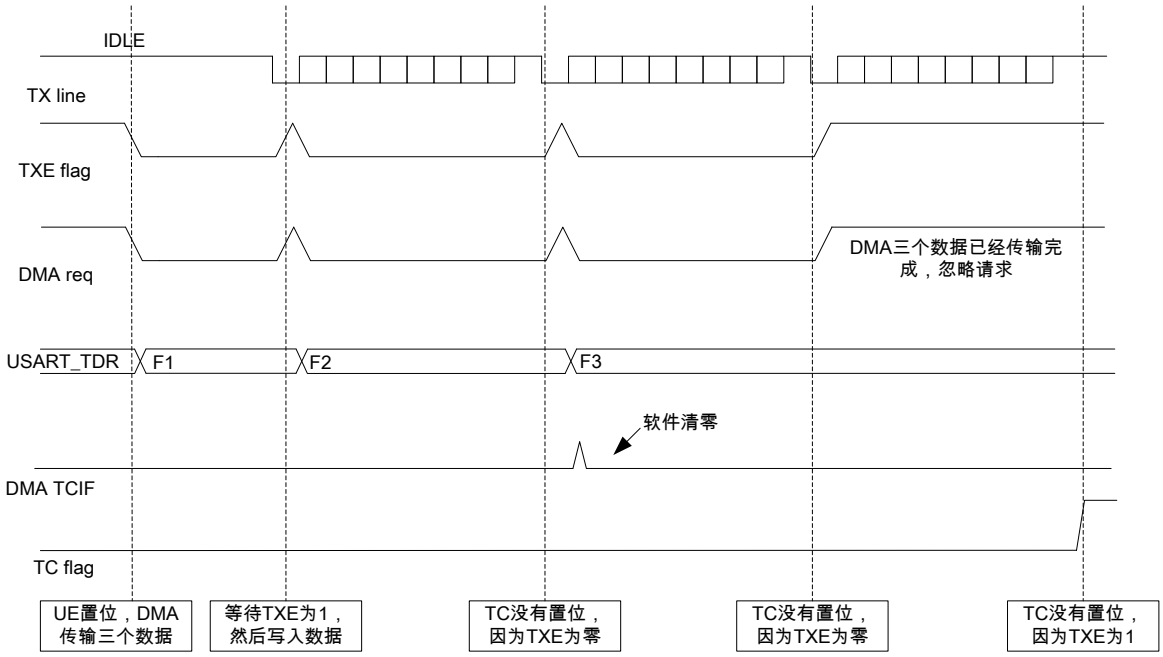


图 26.12 DMA 发送时序图

利用 DMA 接收

使用 DMA 进行接收，可以通过设置 USART_CR3 寄存器上的 DMAR 位激活。当收到一个字节数据时，从 USART_RDR 寄存器取出来的数据会被转移到 DMA 外设中指向的 SRAM 区域。

将一个 DMA 通道设置给 USART 接收，要按照下列步骤：

1. 在 DMA 控制寄存器上将 USART_RDR 配置成 DMA 传输的源地址。在每个 RXNE 事件后，数据将从这个地址取走。
2. 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的目标地址。在每个 RXNE 事件后，数据将从 USART_RDR 去往这个目标地址。
3. 在 DMA 控制寄存器中配置要传输的总的字节数。
4. 在 DMA 寄存器上配置通道优先级。
5. 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。
6. 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（如图 26.13 所示）。

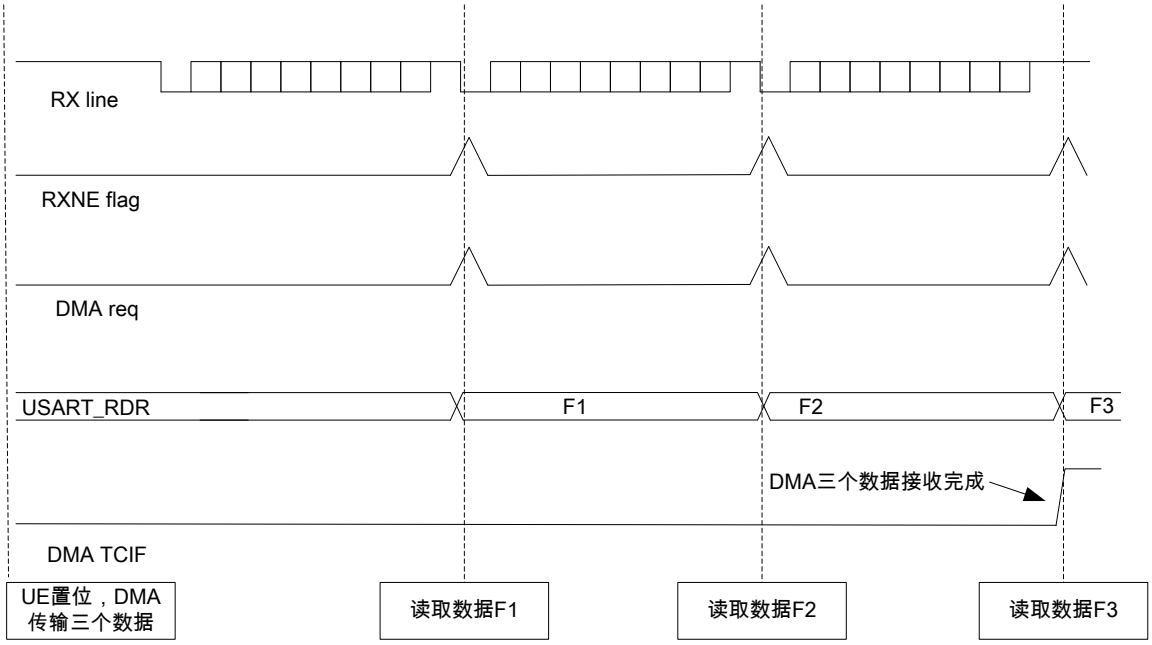


图 26.13 DMA 接收时序图

多缓冲器通信中的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，会在当前字节传输后将错误标志置 1。如果中断使能位被置 1，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 EIE，如果 EIE 被置 1 了，会在当前字节传输结束后，产生中断。

26.2.14. 硬件控制流和 RS485 驱动使能

利用 nCTS 输入和 nRTS 输出可以控制 2 个设备间的串行数据流。图 26.13 显示了这种模式下如何连接两个设备：

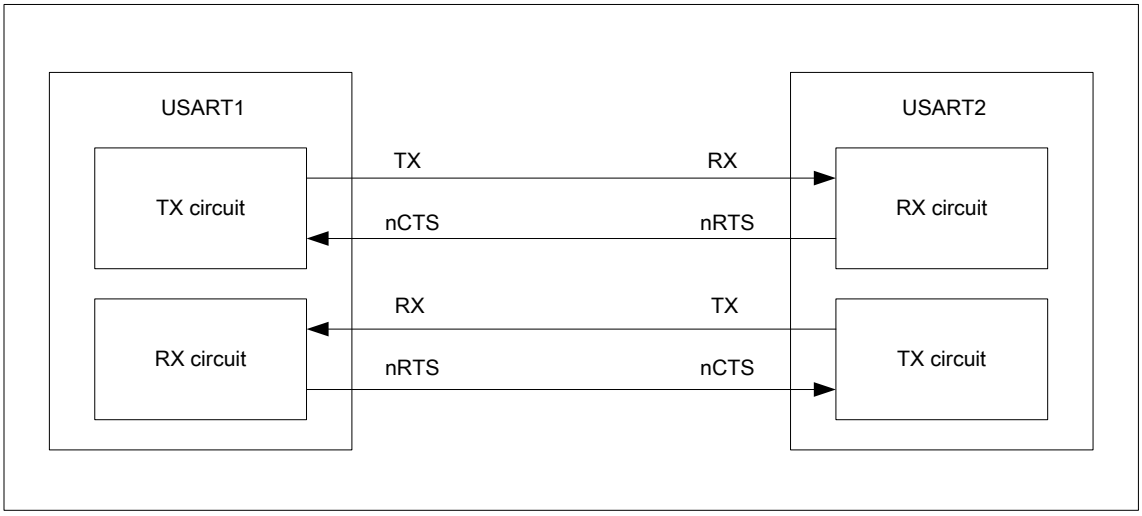


图 26.13 两个 USART 接口之间的硬件控制流连接

通过将 UASRT_CR3 中的 RTSE 和 CTSE 置 1，可以分别独立地使能 RTS 和 CTS 流控制。

RTS 流控制

如果 RTS 流控制被使能(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)。当接收寄存器内的数据未被取走时，nRTS 被释放，由此表明希望在当前帧结束时停止数据传输。图 26.14 是一个启用 RTS 流控制的通信的例子。

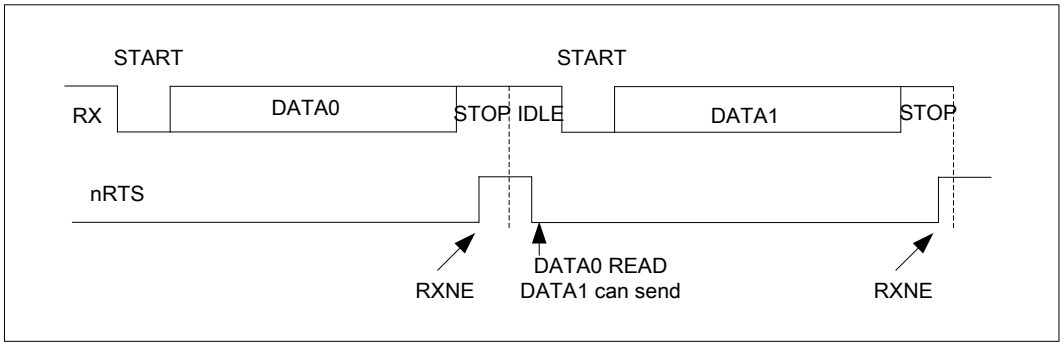


图 26.14 RTS 流控图

CTS 流控制

如果 CTS 流控制被使能(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)，则下一个数据被发送(假设那个数据是准备发送的 ,也就是如果 TXE=0) ,否则下一帧数据不被发出去。若 nCTS 在传输期间被变成无效，当前的传输完成后才停止发送。

当 CTSE=1 时，只要 nCTS 输入变换状态，硬件就自动设置 CTSIF 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 26.15 是一个启用 CTS 流控制的通信的例子。

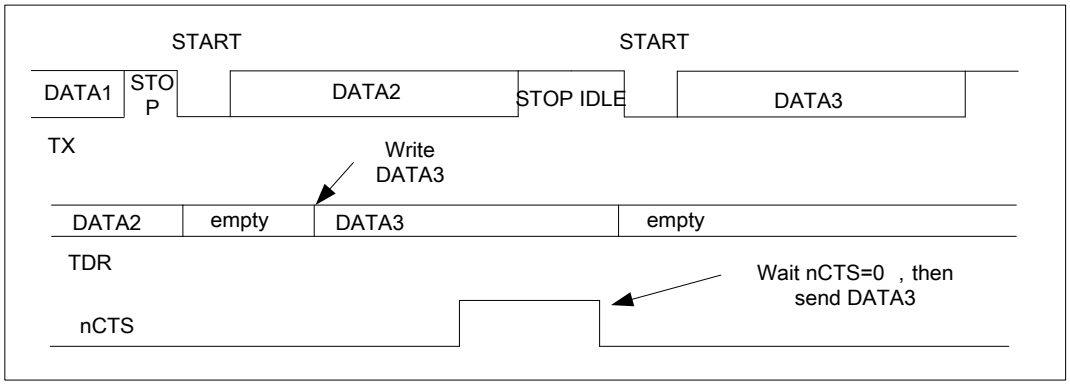


图 26.14 CTS 流控图

RS485 驱动使能

驱动使能功能用 USART_CR3 控制寄存器的 DEM 位置 1 来打开。它允许用户通过 DE (驱动使能) 信号来激活外部收发器的控制端。提前时间的意思是驱动使能信号和第一个字节的起始位之间的时间间隔。这个时间可以在 USART_CR1 控制寄存器的 DEAT[4:0]域中设置。滞后时间是一个发送消息的最后一个字节的停止位和释放 DE 信号之间的时间间隔。这个时间可以在 USART_CR1 控制寄存器的 DEDT[4:0]域中设置。DE 信号的极性则可以通过 USART_CR3 控制寄存器中的 DEP 位进行选择。

26.2.15. 低功耗模式

表 26.7 低功耗模式对 USART 影响

模式	描述
睡眠模式	睡眠模式下，USART 模块可以正常的接收数据，产生中断时可以唤醒 CPU
停止模式	停止模式下，USART 模块没有时钟，不能工作，也不能唤醒 CPU
待机模式	待机模式下，USART 模块断电，不能工作，在退出待机模式时必须重新初始化

26.2.16. 中断

表 26.8 USART 中断请求

中断事件	中断标志	中断使能控制位
数据寄存器为空	TXE	TXEIE
CTS 中断	CTSIF	CTSIE
发送完成	TC	TCIE
接收数据寄存器非空	RXNE	RXNEIE
溢出标志位	ORE	
检测到空闲帧	IDLE	IDLEIE
奇偶校验错误	PE	PEIE
噪声标志位、溢出表示位和帧错误标志位	NF、ORE 和 FE	EIE
字符匹配标志位	CMF	CMIE
接收超时错误	RTOF	RTOIE

USART 中断事件全部连接到同一个中断向量(见图 26.15)

- 发送期间：发送完成，发送数据寄存器空。
- 接收期间：空闲帧检测，溢出错误，接收数据寄存器非空，校验错误，噪声标志，帧错误，字符匹配等等。

如果设置了相应的使能控制位，这些事件都可以引起中断。

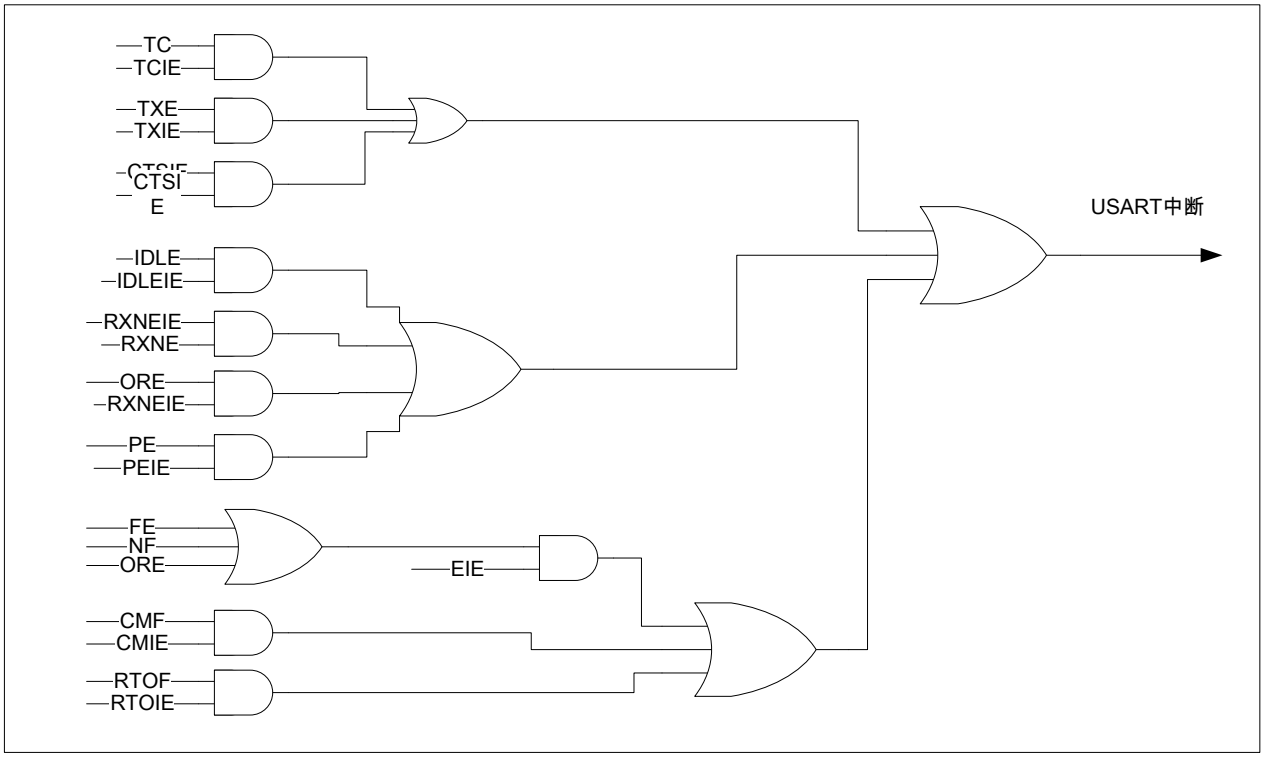


图 26.15 USART 中断映射图

26.3. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	USARTx_CR1	-	-	-	M1	-	RTIOE	DEAT[4:0]					DEDT[4:0]					OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	-	UE
	Reset	x	x	x	0	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0
0x04	USARTx_CR2	ADD[7:4]				ADD[3:0]				RTOEN	ABRMOD[1:0]		ABREN	MSBFIRST	DATAINV	TXINV	RXINV	SWAP	-	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	-	-	-	ADDM7	-	-	-	-
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	0	0	0	0	0	0	x	x	x	0	x	x	x	x
0x08	USARTx_CR3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DEP	DEM	DDRE	OVRDIS	ONEBIT	CTSIE	CTSE	RTSE	DMAT	DMAR	-	-	-	HDSEL		EIE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	x	x	x	0	x	0
0x0C	USARTx_BRR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BRR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	USARTx_RTOR	-	-	-	-	-	-	-	-	RTOR[23:0]																							
	Reset	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	USARTx_RQR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXFRQ	MMRQ	SBKRQ	ABRRQ
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0
0x1C	USARTx_ISR	-	-	-	-	-	-	-	-	-	-	-	-	-	SBKF	CMF	BUSY	ABRF	ABRE	-	-	RTOF	CTS	CTSIF	-	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	x	x	0	0	0	x	1	1	0	0	0	0	0	0
0x20	USARTx_ICR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CMCF	-	-	-	-	-	RTOCF	-	CTSCF	-	-	TCCF	-	IDLECF	ORECF	NCF	FECF	PECF
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	0	x	0	x	x	0	x	0	0	0	0	0
0x24	USARTx_RDR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RDR[8:0]								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0
0x28	USARTx_TDR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TDR[8:0]								
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0

26.3.1. USARTx_CR1

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—			M1	—	RTOIE	DEAT[4:3]	
类型	RO-0	RO-0	RO-0	RW	RO-0	RW	RW	
23:16	DEAT[2:0]			DEDT[4:0]				
类型	RW			RW				
15:8	OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:29	NA	保留位，未定义
28	M1	字长 这位寄存器跟第 12 位（M0）共同决定字长。通过软件置位或复位 M[1:0]=00: 1 位 start 位，8 位数据， n 位 stop M[1:0]=01: 1 位 start 位，9 位数据， n 位 stop M[1:0]=10: 1 位 start 位，7 位数据， n 位 stop 此位寄存器只能在 usart 关闭（UE=0）的时候写
27	NA	保留位，未定义
26	RTOIE	接收超时中断使能 1：接收超时中断使能打开 0：接收超时中断使能关闭
25:21	DEAT	驱动使能提前时间，这个 5 位数字定义 DE（驱动器使能）信号激活和第一个发送字节的起始位的时间间隔。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定）
20:16	DEDT	驱动使能滞后时间 ,这个 5 位数字是一个发送消息的最后一个字节的停止位和释放 DE 信号之间的时间间隔。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定）
15	OVER8	过采样模式 1：8 倍过采样 0：16 倍过采样
14	CMIE	字符匹配中断使能 1：使能字符匹配中断 0：禁用字符匹配中断
13	MME	静默模式使能 1：使能静默模式 0：禁用静默模式
12	M0	字长，这个位设置串口模式的字长 1：9 位数据位

		0：8 位数据位
11	WAKE	接收器唤醒方式 1：由地址匹配唤醒 0：由空闲帧唤醒
10	PCE	奇偶校验位使能 1：使能奇偶校验位 0：禁用使能校验位
9	PS	校验位选择 1：偶校验 0：奇校验
8	PEIE	校验错误中断使能 1：使能奇偶校验位错误中断 0：禁止奇偶校验位错误中断
7	TXEIE	发送寄存器空中断使能 1：使能发送为空寄存器中断 0：禁用发送为空寄存器中断
6	TCIE	发送完毕中断使能 1：使能发送完毕中断 0：禁止发送完毕中断
5	RXNEIE	接收寄存器非空中断使能 1：使能接收非空中断 0：禁止接收非空中断
4	IDLEIE	空闲中断使能 1：使能空闲帧中断 0：禁止空闲帧中断
3	TE	发送器使能 1：发送器打开 0：发送器关闭
2	RE	接收器使能 1：接收器打开 0：接收器关闭
1	NA	保留位，未定义
0	UE	USART 接口使能 1：使能 USART 接口 0：关闭 USART 接口

26.3.2. USARTx_CR2

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	ADD[7:0]							
类型	RW							
23:16	RTOEN	ABRMOD[1:0]		ABRE	MSBFIRST	DATAINV	TXINV	RXINV

类型	RW	RW		RW	RW	RW	RW	RW
15:8	SWAP	—	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL
类型	RW	RO-0	RW		RW	RW	RW	RW
7:0	—			ADDM	—			
类型	RO-0	RO-0	RO-0	RW	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
31:24	ADD	USART 的节点地址，该地址在多处理器模式时，用作地址检测；在正常接收模式时用作，可以用作字符匹配检测
23	RTOEN	接收器超时检测功能使能 1：使能接收器超时检测功能 0：关闭接收器超时检测功能
22:21	ABRMOD	00：对起始位长度进行测量，用作波特率检测 01：使用下降沿对下降沿长度的测量，用作波特率检测 1x：保留
20	ABREN	自动波特率检测使能 1：使能自动波特率检测 0：关闭自动波特率检测
19	MSBFIRST	高位在前 1：起始位之后从高位到低位的顺序发送数据 0：起始位之后从低位高高位的顺序发送数据
18	DATAINV	总线数据反向，包括奇偶校验位 1：发送或者接收时，数据采用的反向逻辑，低电平代表 1 逻辑，高电平代表 0 逻辑 0：发送或者接收时，数据采用的正向逻辑，低电平表示 0 逻辑，高电平代表 1 逻辑
17	TXINV	TX 脚有效电平反向 1：TX 脚逻辑电平反向，低电平表示空闲 0：TX 脚工作在正常逻辑电平，高电平表示空闲
16	RXINV	RX 脚有效电平反向 1：RX 脚逻辑电平反向，低电平表示空闲 0：RX 脚逻辑工作在正常电平，高电平表示空闲
15	SWAP	交换 TX/RX 引脚 1：使能 TX/RX 引脚交换功能 0：禁用 TX/RX 引脚交换功能
14	NA	保留位，未定义
13:12	STOP	停止位配置 00：1 个停止位 01：保留 10：2 个停止位 11：1.5 个停止位
11	CLKEN	时钟使能

		1：使能 SCLK 引脚 0：禁止 SCLK 引脚
10	CPOL	时钟极性 1：在没有数据传输的时候保持高电平 0：在没有数据传输的时候保持低电平
9	CPHA	时钟相位 1：第二个时钟沿采样第一个数据 0：第一个时钟沿采样第一个数据
8	LBCL	末位时钟脉冲控制 1：SCLK 在传输末尾位数据的时候有脉冲 0：SCLK 在传输末尾位数据的时候没有脉冲
7:5	NA	保留位，未定义
4	ADDM	7 位地址检测或者 4 位地址检测 1：7 位地址检测（8 位数据模式下） 0：4 位地址检测
3:0	NA	保留位，未定义

26.3.3. USARTx_CR3

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	DEP	DEM	DDRE	OVRDIS	ONEBIT	CTSIE	CTSE	RTSE
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	DMAT	DMAR	—		HDSEL	—		EIE
类型	RW	RW	RO-0	RO-0	RW	RO-0	RO-0	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	DEP	驱动使能输出脚极性选择 1：DE 信号低有效 0：DE 信号高有效
14	DEM	驱动器使能模式 1：使能 DE 功能 0：关闭 DE 功能
13	DDRE	在接收错误时禁止 DMA 1 :接收错误时禁止 DMA 请求 ,软件必须清零帧错误/奇偶校验错误/噪声错误，然后 DMA 的请求才会恢复。 0：接收错误时不禁止 DMA
12	OVRDIS	溢出检查禁止

		1：禁止溢出检测错误，在数据接收非空时，数据未读走时，新接收的数据会被覆盖掉 0：使能溢出检测
11	ONEBIT	单次采用使能 1：使能单次采样模式 0：使能三次采样模式
10	CTSIE	CTS 中断使能 1：使能 CTS 中断 0：关闭 CTS 中断
9	CTSE	CTS 功能使能 1：打开 CTS 流控 0：关闭 CTS 流控
8	RTS	RTS 使能 1：打开 RTS 流控 0：关闭 RTS 流控
7	DMAT	DMA 发送使能 1：使能 DMA 数据发送模式 0：关闭 DMA 数据发送模式
6	DMAR	DMA 接收使能 1：使能 DMA 数据接收模式 0：关闭 DMA 数据发送模式
5:4	NA	保留位，未定义
3	HDSEL	半双工模式 1：使能半双工模式 0：关闭半双工模式
2:1	NA	保留位，未定义
0	EIE	错误中断使能 1：使能错误中断，错误包括帧错误、噪声错误和奇偶校验错误 0：关闭错误中断

26.3.4. USARTx_BRR

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	BRR	波特率寄存器 BRR[15:4] USARTDIV 的整数部分 BRR[3:0] USARTDIV 的小数部分

26.3.5. USARTx_RTOR

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

Bit	Name	Function
31:24	NA	保留位，未定义
23:0	NA	接收超时配置，单位是波特时钟的长度

26.3.6. USARTx_RQR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—				RXFRQ	MMRQ	SBKRQ	ABRRQ
类型	RO-0	RO-0	RO-0	RO-0	W-R0	W-R0	W-R0	W-R0

Bit	Name	Function
31:4	NA	保留位，未定义
3	RXFRQ	接收数据清空请求 1：清空接收数据状态，RXNE 标志位清零 0：未进行接收数据清空请求
2	MMRQ	静默模式请求

		1：请求进入静默模式 0：未请求进入静默模式
1	SBKRQ	请求发送断开帧字符 1：请求发送断开帧字符 0：未请求发送断开帧字符
0	ABRRQ	自动波特率检测请求 1：请求波特率检测，ABRF 标志位为 1 时，会清掉 ABRF 标志位 0：未请求进行波特率检测

26.3.7. USARTx_ISR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—				RWU	SBKF	CMF	BUSY
类型	RO-0	RO-0	RO-0	RO-0	RO	RO	RO	RO
15:8	ABRF	ABRE	—		RTOF	CTS	CTSIF	—
类型	RO	RO	RO-0	RO-0	RO	RO	RO	RO-0
7:0	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:18	NA	保留位，未定义
19	RWU	接收器从静默模式唤醒 1：接收器处于静默模式 0：接收器处于活动模式
18	SBKF	断开帧发送标志 1：表示当前要发送断开帧，发送完成后自动清零 0：未发送断开帧字符
17	CMF	字符匹配标志 1：接收的字符与 ADD[7:0]字符相匹配 0：未接收到匹配的字符
16	BUSY	忙标志 1：正在接受数据，空闲时自动清零 0：接收器处于空闲状态
15	ABRF	自动波特率检测标志 1：检测到通信的波特率 0：未检测到波特率
14	ABRE	自动波特率检测出错标志 1：自动波特率检测失败，向 ABRRQ 写 1 后清零 0：波特率检测未出错

13:12	NA	保留位，未定义
11	RTOF	接收超时标志 1：接收数据中超过了 RTOR 中设定的值 0：接收数据未出现超时
10	CTS	CTS 引脚电平的反向 1：CTS 引脚处于低电平 0：CTS 引脚处于高电平
9	CTSIF	CTS 中断标志 1：当 CTSE 为 1，CTS 引脚电平发生变化时，该位置 1 0：CTS 引脚电平未发生变化
8	NA	保留位，未定义 1：检测到断开帧标志 0：未检测到断开帧标志
7	TXE	发送数据为空标志 1：数据寄存器为空(数据被传送到了移位寄存器) 0：数据寄存器非空
6	TC	发送完成标志，写 TXDR 寄存器或者写 1 到 TCCF 会清零该寄存器 1：数据发送完成 0：数据发送未完成
5	RXNE	接收非空，数据读走之后自动清零 1：接收数据寄存器中有数据 0：接收数据寄存器中为空
4	IDLE	空闲帧检测标志，写 IDLECF 为 1 后自动清零 1：检测到空闲帧标志 0：未检测到空闲帧标志
3	ORE	溢出错误标志 ,在 RXNE 为 1 时 ,又接收到新的数据时 ,该位置 1 ,写 ORECF 位后清零 1：检测到接收溢出标志 0：未检测到接收溢出标志
2	NF	数据采样期间，噪声检测标志 1：检测到噪声检测标志 0：未检测到噪声标志
1	FE	帧错误标志，写 FECF 位为 1 清零该标志位 1：接收过程中检测到帧错误或者断开帧 0：未接收到帧错误标志位
0	PE	校验错误标志，写 PECF 位清零 1：检测到就校验错误 0：未检测到奇偶校验错误

26.3.8. USARTx_ICR

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

31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—						CMCF	—
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	W-R0	RO-0
15:8	—				RTOCF	—	CTSCF	—
类型	RO-0	RO-0	RO-0	RO-0	W-R0	RO-0	W-R0	W-R0
7:0	—	TCCF	—	IDLECF	ORECF	NCF	FECF	PECF
类型	RO-0	W-R0	RO-0	W-R0	W-R0	W-R0	W-R0	W-R0

Bit	Name	Function
31:18	NA	保留位，未定义
17	CMCF	写 1 清零 CMF 标志位 1：清零 CMF 标志位 0：未对 CMF 标志位进行清零操作
16:12	NA	保留位，未定义
11	RTOCF	写 1 清零 RTOF 标志位 1：清零 RTOF 标志位 0：未对 RTOF 标志位进行清零操作
10	NA	保留位，未定义
9	CTSCF	写 1 清零 CTSIF 标志位 1：清零 CTSIF 标志位 0：未对 CTSIF 标志位进行清零操作
8:7	NA	保留位，未定义
6	TCCF	写 1 清零 TC 标志位 1：清零 TC 标志位 0：未对 TC 标志位进行清零操作
5	NA	保留位，未定义
4	IDLECF	写 1 清零 IDLE 标志位 1：清零 IDLE 标志位 0：未对 IDLE 标志位进行清零操作
3	ORE	写 1 清零 ORECF 标志位 1：清零 ORECF 标志位 0：未对 ORE 标志位进行清零操作
2	NCF	写 1 清零 NF 标志位 1：清零 NF 标志位 0：未对 NF 标志位进行操作
1	FECF	写 1 清零 FE 标志位 1：清零 FE 标志位 0：未对 FE 标志位进行操作
0	PECF	写 1 清零 PE 标志位 1：清零 PE 标志位 0：未对 PE 标志位进行操作

26.3.9. USARTx_RDR

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							RDR[8]
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO
7:0	RDR[7:0]							
类型	RO							

Bit	Name	Function
31:9	NA	保留位，未定义
8:0	RDR	数据接收寄存器

26.3.10. USARTx_TDR

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

Bit	Name	Function
31:9	NA	保留位，未定义
8:0	TDR	数据寄发送存器

27. 串行外设接口（SPI）

27.1. 简介

可以通过 SPI 接口和外部设备进行基于 SPI 协议的通信。SPI 模式可通过软件选择。器件复位后默认选中 SPI

模式。

串行外设接口（SPI）协议支持半双工、全双工和简单同步等方式与外部设备进行串行通信。该接口可配置为主机模式，在这种情况下，它向外部的从设备提供通信时钟（SCK）。界面也能够以多主机配置的方式操作。

27.2. SPI 主要功能

- 主设备或从设备模式
- 三线全双工同步传输
- 两线半双工同步传输（双向数据线）
- 两线简单同步传输（单向数据线）
- 4 位到 16 位数据的大小选择
- 多主机模式的能力
- 8 个主模式波特率分频器，波特率高达$f_{PCLK}/2$
- 从模式频率高达$f_{PCLK}/4$
- 主机和从机模式下都可以由硬件或软件管理 NSS 管脚：主/从模式操作的动态变化
- 可编程时钟极性和相位
- 高位在前或低位在前可设置
- 专用的发送和接收状态标志，全部支持中断触发
- SPI 总线忙状态标志
- SPI 摩托罗拉模式支持
- 硬件 CRC 功能实现可靠的通信
  - CRC 值可作为 Tx 模式的最后一个字节发送
  - 自动 CRC 错误检查上次接收到的字节
- 主模式故障、溢出等标志具备中断触发能力
- CRC 错误标志
- 2 个 32 位嵌入式 Rx 和 Tx FIFO 带 DMA 功能
- SPI TI 模式支持

27.3. SPI 功能描述

27.3.1. SPI 概括说明

SPI 允许 MCU 和外部设备之间的同步串行通信。应用软件可以通过状态标志或使用专用的 SPI 中断来管理通信过程。SPI 和它们之间的相互作用的主要内容如下图 27.1 所示。

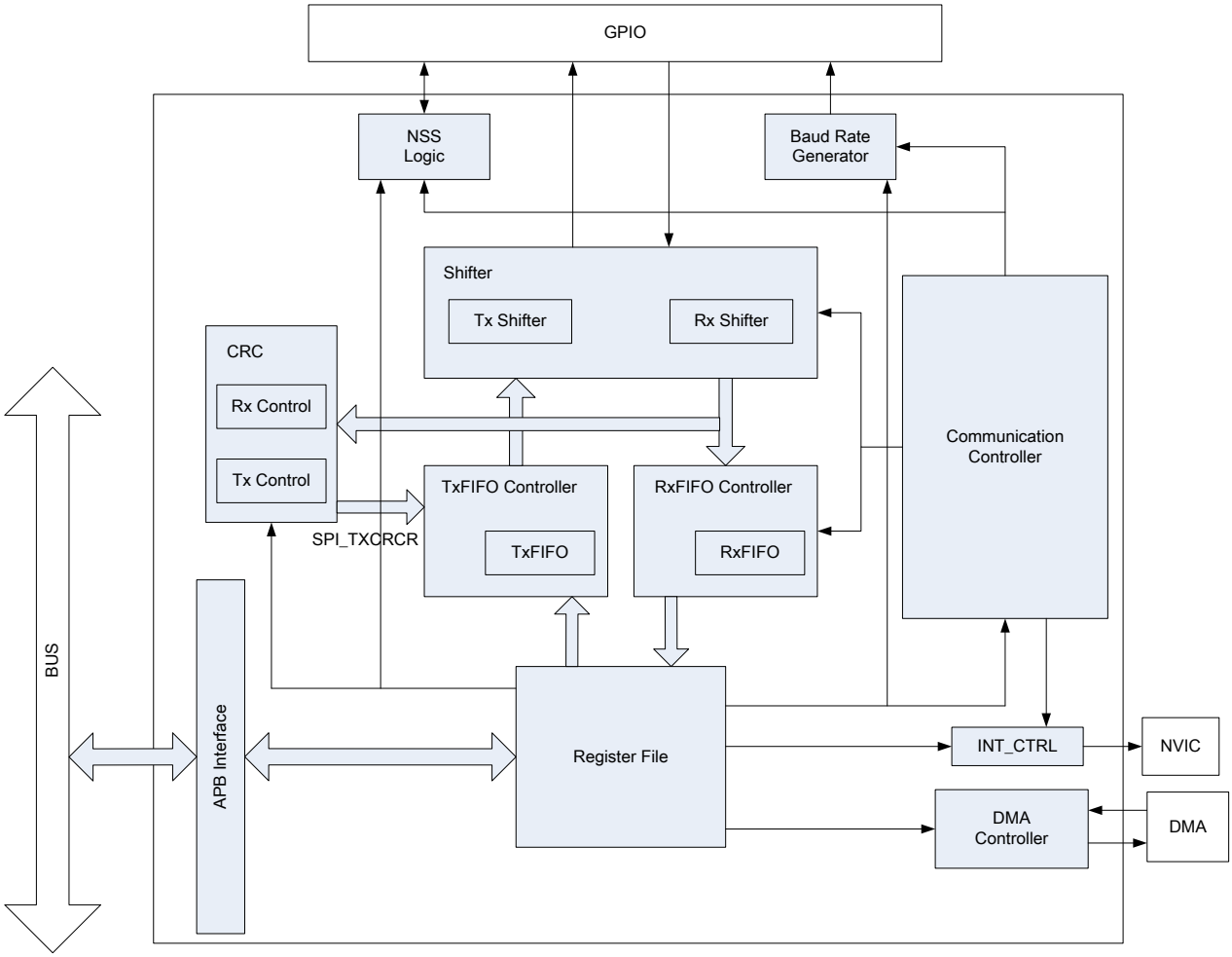


图 27.1 SPI 框图

通常 SPI 通过 4 个引脚与外部设备相连。

- MISO：主设备输入/从设备输出引脚。一般情况下，此引脚用于在从模式下的数据发送和主模式下的数据接收。
- MOSI：主设备输出/从设备输入引脚。一般情况下，此引脚用于在从模式下的数据接收和主模式下的数据发送。
- SCK：SPI 串行时钟输出引脚，作为主设备输出和从设备输入。
- NSS：从机片选脚。根据 SPI 和 NSS 设置，此引脚可用于：
  - 选择一个从机来进行通信
  - 同步数据帧
  - 检测多个主机之间的冲突

  参考 24.3.4 节：从机片选（NSS）引脚管理。

SPI 总线允许一个主设备和一个或多个从设备之间的通信。总线由至少两条线-时钟信号和同步数据传输。其他信号基于 SPI 节点和主机间的数据交换目的来添加。

27.3.2. 一个主设备和一个从设备之间的通信

SPI 允许 MCU 根据目标的设备和应用程序的要求 ,使用不同的配置进行通信。这些配置使用 2/3 线(软件 NSS 管理) 或 3/4 线 (硬件 NSS 管理)。通信总是由主机发起的。

全双工通信

默认情况下 ,SPI 被配置为全双工通信。在此配置中 ,主机和从机的移位寄存器通过两个单向线 MOSI 和 MISO 引脚进行连接。在 SPI 通信时，按照主机提供的 SCK 时钟沿进行同步数据传输。主机的数据经由 MOSI 发送到从机，从机的数据经由 MISO 发送到主机。当数据帧传输完成 (所有位转移) 时，主机和从机间的信息交换就完成了。

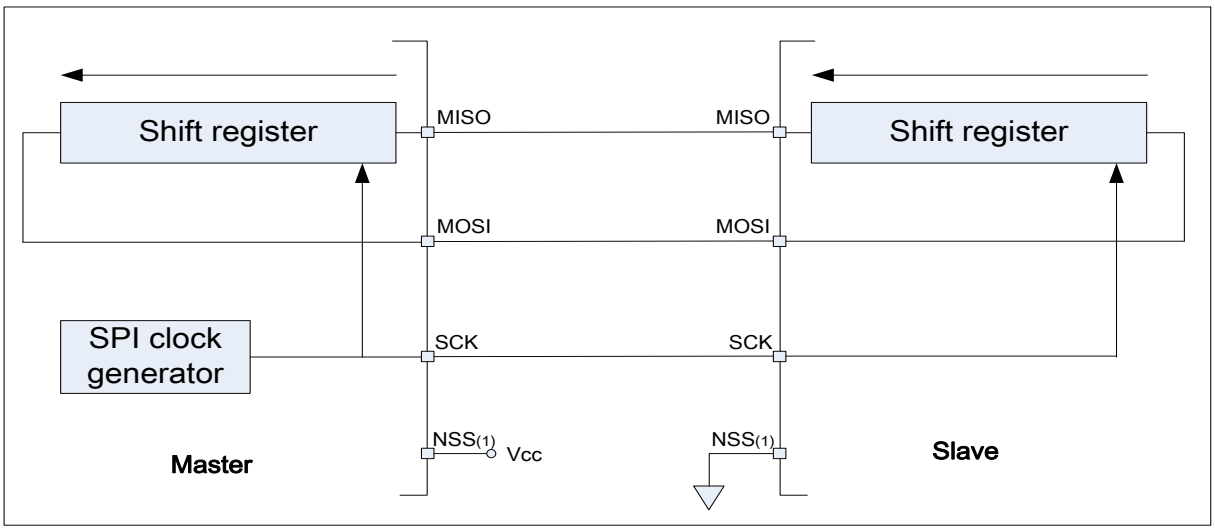


图 27.2 全双工单主/单从应用

1. 在这种情况下，NSS 引脚被配置为输入。

半双工通信

通过设置在控制寄存器 1 (SPIx_CR1) 中的 BIDIMODE 位可以令 SPI 工作在半双工模式下。在此配置中由一个单一的跨接线连接主机和从机。在通信器件 ,数据按照控制寄存器 1 (SPIx_CR1) 中的 BIDIOE 位的设定，由主机移位寄存器同步于 SCK 时钟沿传输到从机。在此配置中，主机的 MISO 引脚和从机的 MOSI 引脚都是自由的，可作为 GPIO 供其他应用程序使用。

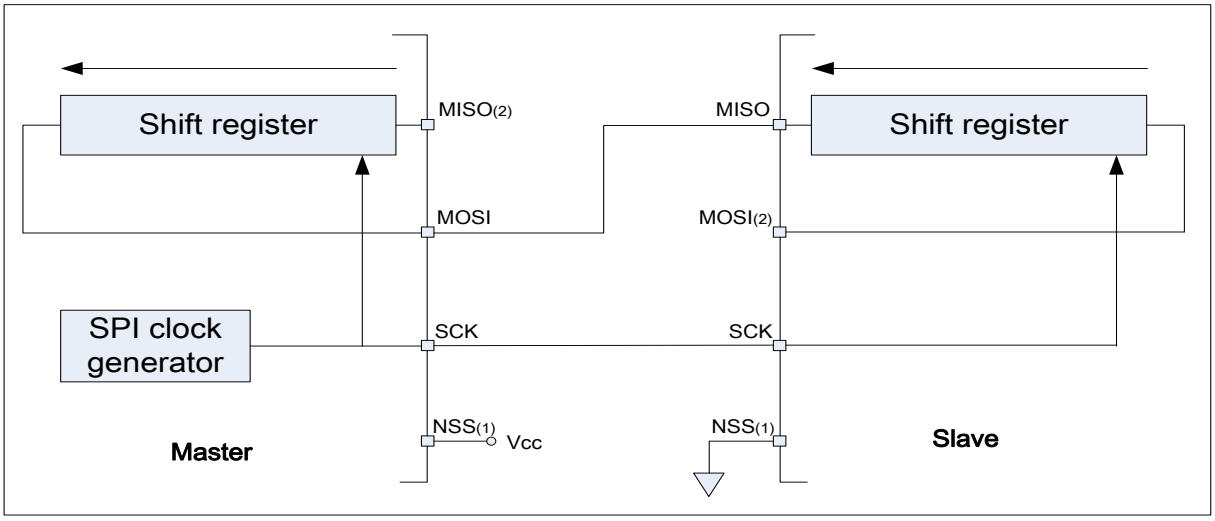


图 27.3 半双工单主/单从应用

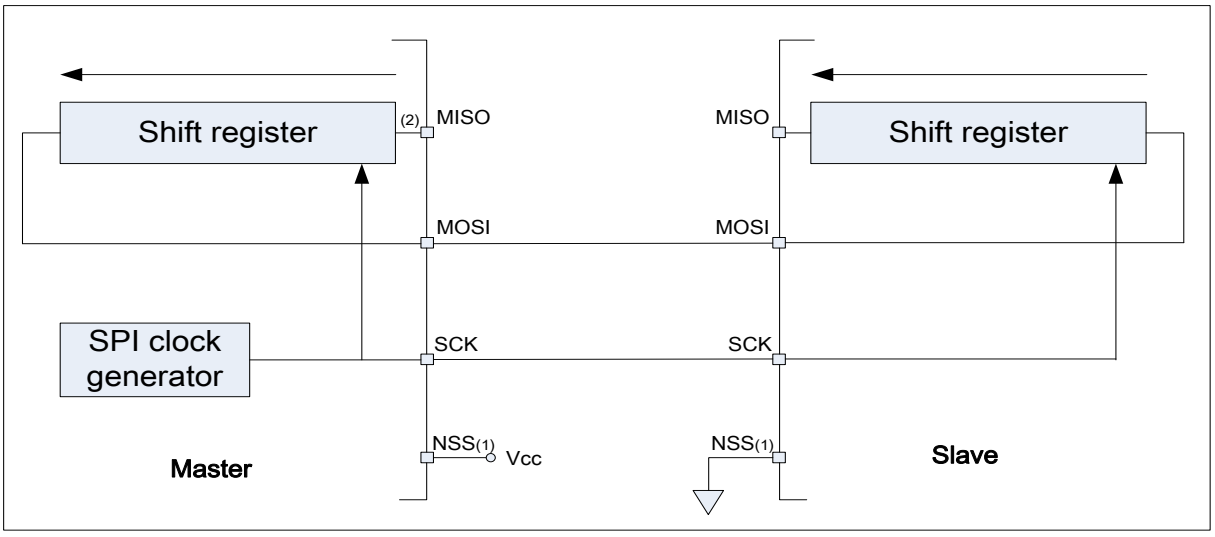
1. 在这种情况下，NSS 引脚被配置为输入。
2. 在此配置中，主机的 MISO 引脚和从机的 MOSI 引脚可以作为 GPIO 使用。

简单通信

可以通过控制寄存器 2(SPIx_CR2)中 RXONLY 位选择 SPI 工作在单工模式下，实现单发送或单接收通信。在此配置中，由一个单一的跨接线连接主机和从机。其余 MISO 和 MOSI 引脚不用于通信，并可以作为标准的 GPIO 使用。

- 只发送模式 (RXONLY=0) : 配置设置和全双工相同。应用程序必须忽略在未使用的输入引脚上捕获的信息。这个脚可以作为一个标准的 GPIO 引脚。
- 只接收模式(RXONLY=1) :应用程序可以设置 RXONLY 位以禁用 SPI 输出功能。在配置为从机时，MISO 输出被禁用，并且可以当作一个 GPIO 引脚来使用。在从机选择信号处于激活状态时，从机将从 MOSI 引脚连续接收数据。根据数据缓冲区的配置，会出现数据接收事件。

  在配置为主机时，MOSI 输出被禁用，并且可以当作一个 GPIO 引脚来使用。SPI 使能时，时钟信号会不断产生。要停止时钟输出，只有清除 RXONLY 位或 SPE 位，并等待直到从 MISO 引脚的输出样式完成并填充数据缓冲区结构，这取决于其具体配置。



图

27.4 简单的单主/单从应用（主机只发送/从机只接收）

1. 在这种情况下，NSS 引脚被配置为输入。
2. 在移位寄存器中输入的信息被捕获，但在标准只发送模式下必须丢弃（例如，OVF 格式标志）。
3. 在此配置中，只要是 MISO 引脚都可用作 GPIO。

注意：简单通信模式可以被一个固定传输方向的半双工通信模式替代。

27.3.3. 标准多从机通信

在有两个或两个以上独立的从机的配置时，主机用 GPIO 引脚来管理每个从机的片选线。主机必须通过 GPIO 拉低其中一个从机的 NSS 引脚。一旦做到这一步，就会建立一个标准的主机和专用的通信渠道。

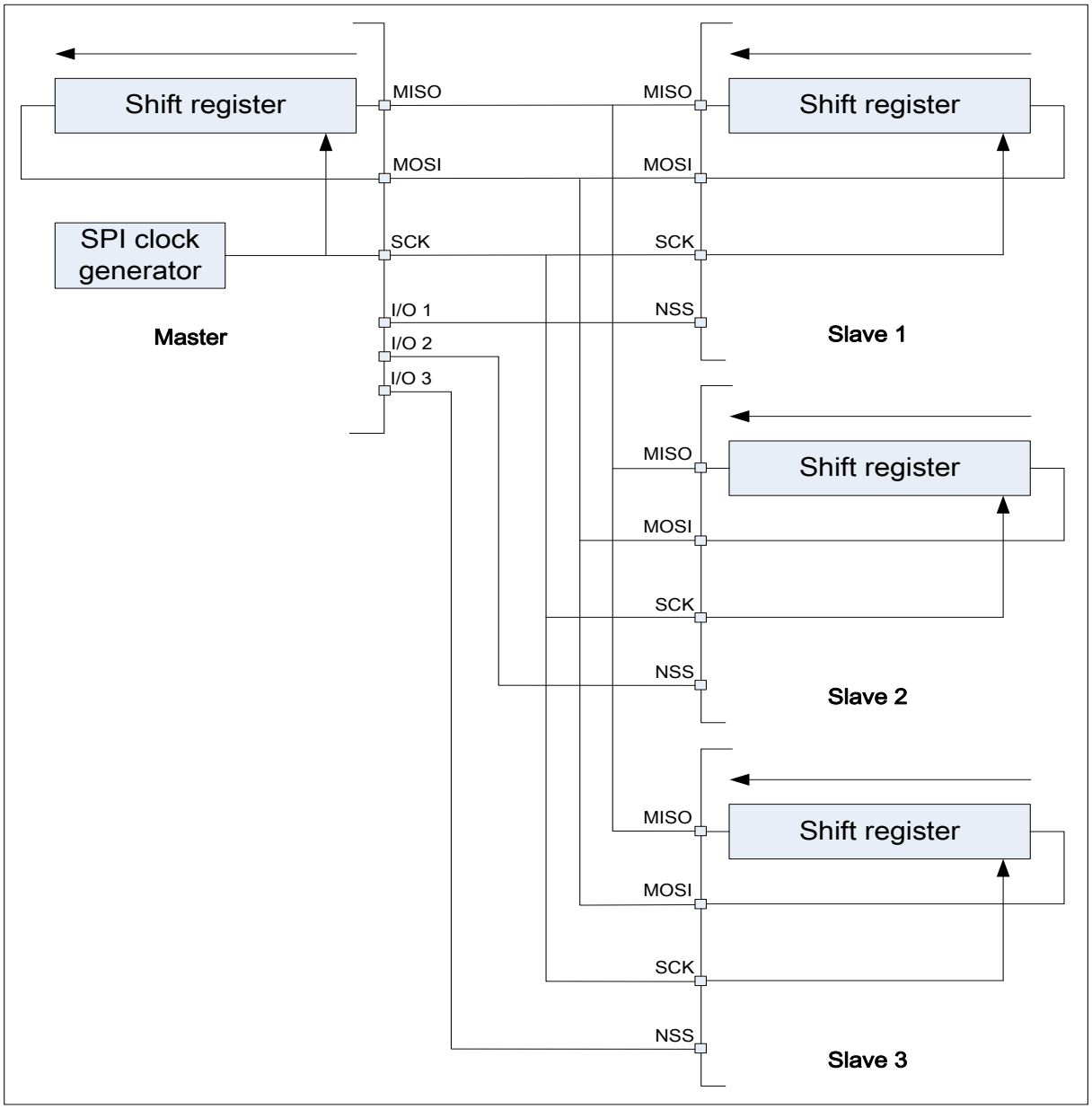


图 27.5 主机和 3 个独立从机

注意：由于从机的 MISO 引脚连接在一起，所有的从机必须将它们的 MISO 引脚设置为备用功能漏极开路状态。

27.3.4. 从机片选引脚（NSS）管理

在从机模式下，NSS 作为一个标准的片选输入工作，并允许主从通信。在主机模式下，NSS 可以用来作为输入或输出。作为输入，它可以防止多主机总线冲突；作为输出，它可以驱动一个单一的从机片选信号。
可以设置控制寄存器 1（SPIx_CR1）中的 SSM 位来选择使用硬件或软件的从机选择管理：

- 软件 NSS 管理（SSM=1）：在此配置中，从机选择信息由内部寄存器控制寄存器 1（SPIx_CR1）中的 SSI 位的值来驱动。外部 NSS 引脚可以由其他应用程序自由支配。
- 硬件 NSS 管理（SSM=0）：在这种情况下，有两种可能的配置。这种设置由控制寄存器 1（SPIx_CR1）中的 SSOE 位来决定 NSS 的输出。
  - NSS 输出使能（SSM=0，SSOE=1）：此配置仅用于当 MCU 作为主机时。NSS 引脚由硬件管理。主机模式下，在 SPI 使能（SPE=1）的同时 NSS 信号被拉低，并保持低直到 SPI 被禁止（SPE=0）。如果 NSS 脉冲模式被打开（NSSP=1），在连续通信间隔期间可以产生一个 NSS 脉冲。在这种 NSS 设置中，SPI 不能支持多重主机工作。
  - NSS 输出禁止（SSM=0，SSOE=0）：如果在总线上 MCU 作为主机，那么此配置就允许多主机通信。如果在这种模式下 NSS 引脚被拉低，SPI 进入主机模式故障状态，并自动重新配置为从机模式。在从机模式下，NSS 引脚作为标准的片选输入来工作；当 NSS 引脚被拉低时，从机被选中。

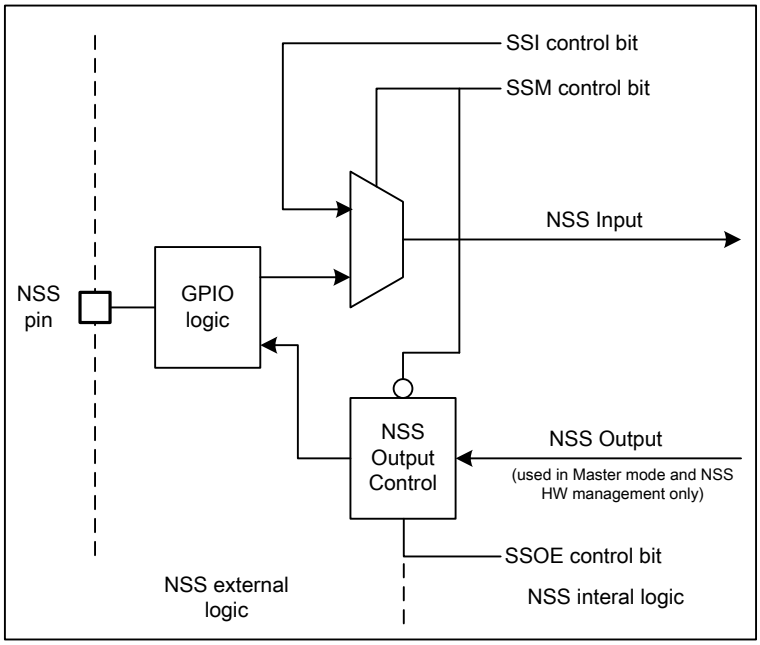


图 27.6 硬件/软件从机选择管理

27.3.5. 通信格式

在 SPI 通信中，接收和发送操作同时进行。串行时钟（SCK）同步发送并同时采样数据线上的信息。通信格式取决于时钟相位、时钟极性和数据帧格式。为了能够正常通信，主机和从机必须遵循相同的通信格式。

时钟相位和极性控制

可以通过控制寄存器 1（SPIx_CR1）中的 CPOL 和 CPHA 位用软件选择 4 种可能的时序关系。时钟极性位（CPOL）控制着在没有数据发送时的空闲状态的时钟输出电平。该位既针对主机模式也针对从机模式。如果 CPOL 被清零，SCK 引脚在闲置状态输出低电平；如果 CPOL 置 1，SCK 引脚在闲置状态输出高电平。

如果 CPHA 位被置 1，SCK 引脚上的第二个沿对准的是第一位的捕获时机（如果 CPOL 位是 0 则为下降沿；如果 CPOL 位为 1 则为上升沿）。每次发生这个时钟切换时，数据被锁存。如果 CPHA 位为 0，SCK 引脚上的第一个沿对准第一位的捕获时机（如果 CPOL 位是 1 则为下降沿；如果 CPOL 位为 0 则为上升沿）。每次发生这个时钟切换时，数据被锁存。

时钟极性（CPOL）和时钟相位（CPHA）的选择共同决定数据采样时的时钟边沿。

图 27.7 显示 CPHA 和 CPOL 的四种组合的 SPI 全双工传输。

注意：1. 改变 CPOL/CPHA 之前，SPI 必须通过清零 SPE 位关闭。

3. 在空闲状态下的 SCK 必须符合在控制寄存器 1(SPIx_CR1)中对极性的选择(如果 CPOL=1 上拉 SCK；如果 CPOL=0 下拉 SCK)。

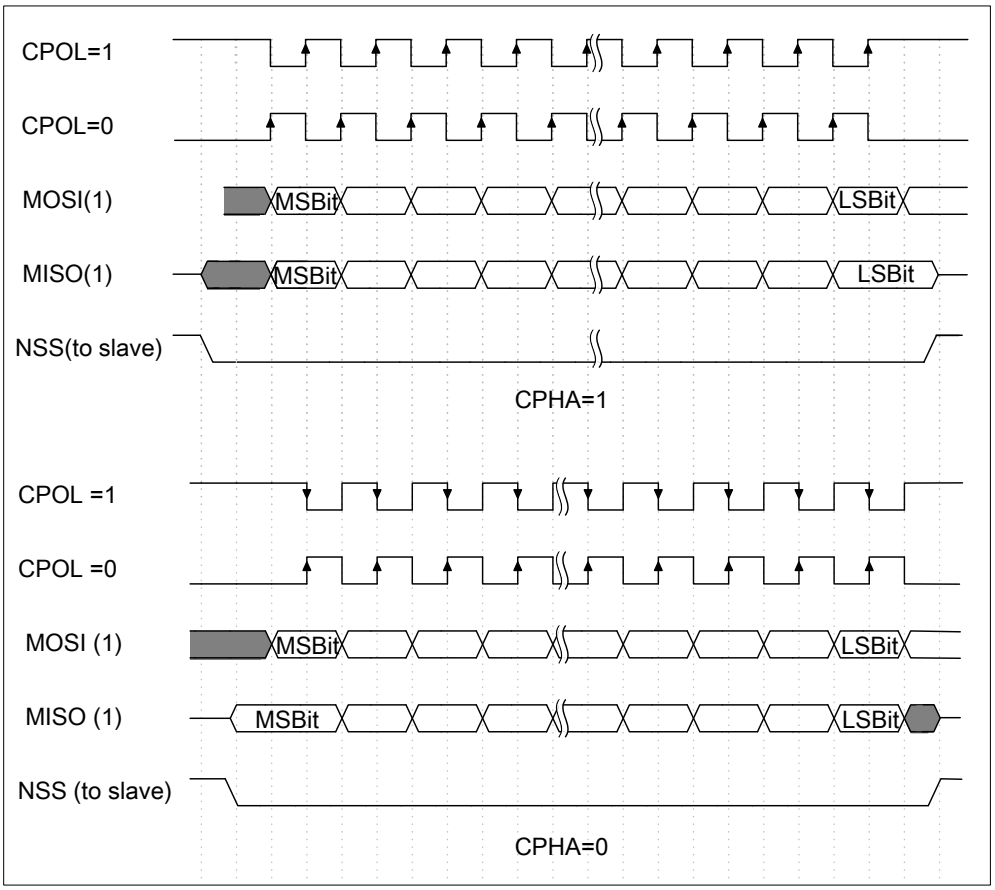


图 27.7 数据时钟时序图

1. 数据传输的顺序根据 LSBFIRST 位的设定而变化。

数据帧格式

SPI 移位寄存器可以设置 MSB 在前或 LSB 在前，取决于 LSBFIRST 位的值。数据字长使用 DS 位选择。它可以设定从 4 位到 16 位的长度 ,同时适用于发送和接收。无论选定多大字长 ,对 FIFO 的读访问必须与 FRXTH 水平对齐。当访问 SPIx_DR 寄存器时，无论是一个字节还是一个字，数据帧总是右对齐。通信时，只有数据

字长范围内的位会随时钟输出。

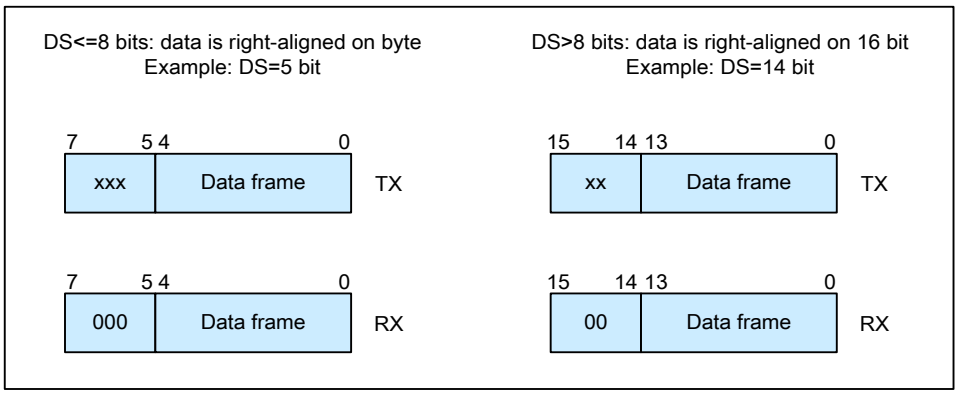


图 27.8 当数据长度不等于 8 位或 16 位时的数据对齐

注意：最小的数据长度为 4 位。如果选择数据长度小于 4 位，会强制按照 8 位的数据字长传输。

27.3.6. SPI 的配置

主机和从机的配置过程几乎是相同的。详细模式的设置可查看接下来的章节。当 SPI 被初始化成一个标准模式进行通信时，可参考下面配置流程：

1. 正确配置 GPIO 相关寄存器：配置 MOSI、MISO 和 SCK 相关的 GPIO 口。
2. 写 SPI_CR1 寄存器：
   a) 设置 BR[2:0]位选择串行时钟的波特率。（注 4）
   b) 设置 CPOL 和 CPHA 位的组合，选择数据传输和串行时钟之间的四种时序中的其中一种（在 NSSP 模式下，CPHA 必须清零）。（注 2）
   c) 配置 RXONLY、BIDIOE 和 BIDIMODE 位选择传输模式（RXONLY 和 BIDIMODE 不能同时为 1）。
   d) 配置 LSBFIRST 位，定义帧格式。（注 2）
   e) 如果需要 CRC 校验，配置 CRCL 和 CRCEN 位（在 SCK 处于闲置状态时进行配置）。
   f) 配置 SSM 和 SSI，选择 NSS 管脚管理模式。（注 2&3）
   g) 配置 MSTR 位选择主从模式（在多重主机的 NSS 配置中，需要在内部 NSS 为高时配置为主机以避免 MODF 错误的产生）。
3. 写 SPI_CR2 寄存器：
   a) 设置 DS 位选择用于传送的数据长度。
   b) 配置 SSOE 位。（注 1&2&3）
   c) 如果需要 TI 协议，设置 FRF 位（TI 模式下，保持 NSSP 为 0）。
   d) 如果在两个数据传输中间需要 NSS 脉冲，则需要设置 NSSP（在 NSSP 模式下，保持 CPHA 和 FRF 位为 0）。
   e) 设置 FRXTH 位。RXFIFO 的阈值必须和对 SPIx_DR 寄存器的读访问的字长对齐。
   f) 在 DMA 传输并且数据打包模式下，需要设置 LDMA_TX 和 LDMA_RX 位。
4. 写 SPI_CRCPR 寄存器：如果需要，可配置 CRC 校验的多项式。
5. 正确配置 DMA 中的寄存器：如果使用 DMA 通信，需要正确的配置 DMA 的工作模式。

注意：

1. 注 1 在从机模式下不需要。
2. 注 2 在 TI 模式下不需要。

3. 注 3 在 NSSP 模式下不需要。
4. 注 4 在从机模式下不需要，除非从机工作在 TI 模式下。

27.3.7. 使能 SPI 的步骤

为了确保传输的正确，需要在主机发送 SCK 时钟之前使能 SPI 从机；如果没有在发送 SCK 时钟之前使能 SPI 从机，传输数据可能会出现错误。在主机发起传输之前，从机的数据寄存器（SPI_DR）中要提前写入需发送的数据（在连续通信的情况下，需要在第一个传输时钟或在正在进行的传输结束之前继续向从机的数据寄存器写入数据）。在从机使能之前，SCK 时钟一定要处于闲置电平。
当主机在全双工模式下（或其他模式作为发送端），SPI 使能并且 TXFIFO 不为空时或 TXFIFO 正在写入数据时，主机自动发起传输。
当主机在各种只接收模式下（RXONLY=1 或 BIDIMODE=1&BIDIOE=0），SPI 使能后，主机自动发起传输并且一直发送传输时钟。

27.3.8. 数据发送和接收流程

RXFIFO 和 TXFIFO

所有 SPI 数据交换都通过 32 位的嵌入式 FIFO。这使 SPI 可以连续工作，防止配置为短数据帧时出现数据断流。每个传输方向都有它自己的 FIFO 称为 TXFIFO 和 RXFIFO。
对 FIFO 的处理的选择会根据数据交换模式（双工、单工）、数据帧格式（字长）、对 FIFO 数据寄存器访问的大小（8 位访问或 16 位访问）以及是否按照数据包访问 FIFO。
对 SPIx_DR 寄存器的读访问，会返回存储在 RXFIFO 中但未读的最老的数据。对 SPIx_DR 的写访问会将新的数据放在发送队列的尾部。读访问必须总是和 SPIx_CR2 寄存器中的 FRXTH 位设置的 RXFIFO 门限对齐。FTLVL[1:0]和 FRLVL[1:0]位表明两个 FIFO 目前的缓冲存储程度。
对 SPIx_DR 寄存器的读访问必须由 RXNE 事件触发。这个事件在数据被存入 RXFIFO 并到达门限（由 FRXTH 位定义）的时候被触发；在 RXNE 被清除时，认为 RXFIFO 是空的。类似的，写访问要由 TXE 事件触发。当 TXFIFO 的存储状况少于或等于满额的一半的时候，将触发此事件；否则 TXE 被清零，并且 TXFIFO 被认为有数据存在。用这种方式，RXFIFO 可以存储多达 4 个数据帧，而 TXFIFO 在字长不大于 8 的时候最多也只能存储 3 个数据帧。这种差异可以防止在已经有 3 个 8 位数据存在 TXFIFO 中的时候，软件试图在 16 位数据帧模式下向 TXFIFO 写入更多的数据而造成数据破坏。TXE 和 RXNE 事件都可通过查询方式和中断方式进行处理。
管理数据交换的另一种方法是使用 DMA。
如果接收到下一个数据时 RXFIFO 是满的，将导致发生溢出事件。溢出事件可以查询处理或中断处理。
正在执行数据传输的时候，硬件会将 BSY 位置 1 来指示这个状态。当时钟信号连续产生时，如果在主机模式下，BSY 标志在帧与帧之间一直保持为 1，而在从机模式下，BSY 标志在帧与帧之间会有一小段时间（一个 SPI 时钟周期）为 0。

序列处理

多个数据字节可以顺序发送来组成一个消息。在启用发送后，在主机 TXFIFO 中的数据会开始发送并连续发完。主机会连续输出时钟信号直至 TXFIFO 变为空，然后停下来等待新增的数据。
在只接收模式（RXONLY=1 或 BIDIMODE=1&BIDIOE=0），只要 SPI 和只接收模式被使能，主机会立即开始接收数据序列。主机连续提供时钟信号，直到主机关闭 SPI 或者关闭只接收模式；主机会连续接收数据直到

这个时刻。

数据帧开始后，从机无法控制或延时数据序列。出于这个原因，从机必须在开始传输之前准备好数据，并总是一直保持有数据待传；主机必须在每个序列之间给从机保留足够的时间以准备数据。需要注意的一点是 SPI 的主机和从机都没有下溢标志，即使从机没有准备好数据，传输也会在主机的控制下进行。如果可能的话，序列中的字节数量应得到限制，以便从机完成自动的数据处理（通过 DMA），尤其是在数据帧较短并且传输速度较快的时候。

在多从机并行的系统中，每个序列应该由 NSS 脉冲来分隔，以将每个序列对应到不同的从机。在单从机系统中通过 NSS 控制从机就显得没有那么必要了，但这里有个脉冲还是更好，这可以令从机和每个数据序列完成同步。NSS 引脚既可以由软件管理也可以由硬件管理。

BSY 位被置 1 标志传输正在进行中。这一点结合 FTLVL[1:0]位，可以用于检查传输是否完成。在系统进入 HALT 模式前这是很有必要的，过早的进入 HALT 模式可能导致数据破坏。判断 BSY 位的另外一个目的是可以用软件来管理 NSS 引脚。当 RXNE 标志被置 1，它意味着对当前的传输结束了；即最后一位刚刚采样完毕以及完整的数据帧存储在 RXFIFO 中。

禁用 SPI 的步骤

当需要关闭 SPI 时，必须要遵守下面内容所描述的关闭流程。这么做对于系统进入低功耗模式是非常重要的。这时正在进行的传输会被打断。在一些模式下，关闭流程是唯一停止 SPI 连续传输的方式。

当停止提供传输数据后，主机会完成当前数据的传输。在这种情况下，最后一个数据传输结束后时钟输出就会停止。在数据打包模式中，奇数个数据传输完毕后要特别注意防止出现空的字节（参考数据打包章节）。如果主机发送器正在发送数据或者下一个数据已经被写入 TXFIFO 时禁用 SPI，那么 SPI 接下来的行为可能是不可保证的。

当主机处于只接收模式下，只有禁用 SPI 或只接收模式才能停止时钟。为了收到正确数量的字节并且阻止任何无效的空数据，就得在最后一个字节正在传输的时候，把握正确的关闭接收的时机。关闭动作必须发生在首位的采样时间和最后一位数据传输开始之前。

如果接收到了数据，但仍然遗留在 RXFIFO 中未及时读取；此时关闭了 SPI 的话，那么在下一次打开 SPI 开始新的传输序列之前一定要先处理历史数据。为了防止这种情况，请确保 RXFIFO 为空的时候再去禁用 SPI。

这个过程可以通过正确的流程来实现或者通过专门的外设复位控制寄存器来执行软件复位命令，从而全面初始化 SPI 的寄存器（见 RCC_APBiRSTR 寄存器中的 SPIiRST 位）。

标准的关闭流程判断 BSY 标志和 FTLVL[1:0]来查看传输是否完成。特殊的情况下，检查传输是否完成也可以由以下的方式去完成，例如：

- 当 NSS 信号由软件控制时，主机必须在传输完成时给从机提供正确的 NSS 脉冲
- 当最后一帧数据正在传输或者 CRC 数据正在传输时，表明 DMA 或 FIFO 中的数据流已经传输结束。

正确的关闭流程（除了使用只接收模式）：

1. 等到 FTLVL[1:0]=00（没有更多的数据需要被传输）。
2. 等待 BSY=0（最后一个数据帧处理完毕）。
3. 禁止 SPI(SPE=0)。
4. 读取数据直到 FRLV[1：0]=00（读取所有接收到的数据）。

在特定的只接收模式下，正确的关闭流程：

1. 在最后一个数据的传输期间的特定时间窗口中禁用 SPI。
2. 等待 BSY=0（最后一个数据帧处理完毕）。
3. 读取数据直到 FRLV[1：0]=00（读取所有接收到的数据）。

注意：如果用了数据打包模式，并且收到一个格式小于等于 8 位的奇数个数据的数据帧，当 FRLVL[1:0]=1 时，FRXTH 必须被置 1，为的是产生 RXNE 事件以便去读取最后的奇数数据。

数据打包

当数据帧的大小适合一个字节（小于或等于 8 位），并且对 SPIx_DR 寄存器执行任何 16 位的读写访问时，数据会自动打包在一起。在这种情况下，可以并行处理双数据。SPI 会先操作低 8 位数据，然后操作高 8 位数据。图 27.9 提供了打包方式顺序处理数据的一个例子。在一次对 SPIx_DR 的 16 位写访问后，就会有 2 个字节的数据被发送出去。如果 RXFIFO 的阈值设置是 16 位（FRXTH=0），该序列则只会生成一个 RXNE 事件，而不是两个。针对这种单个的 RXNE 事件的响应，接收器必须对 SPIx_DR 寄存器作一次 16 位的读访问才能够把数据全部读取出来。RXFIFO 的阈值跟数据访问的位宽必须保持一致，否则就会丢失数据。

如果出现奇数个字节数据，那就会出现特别的问题，这是一定要解决的。在发送端，用 8 位方式访问 SPIx_DR 将最后一个字节发出来就够了。在接收端必须改变 RXFIFO 的门限，以便在传输计数字节的数据帧的最后字节时能够产生 RXNE 事件。

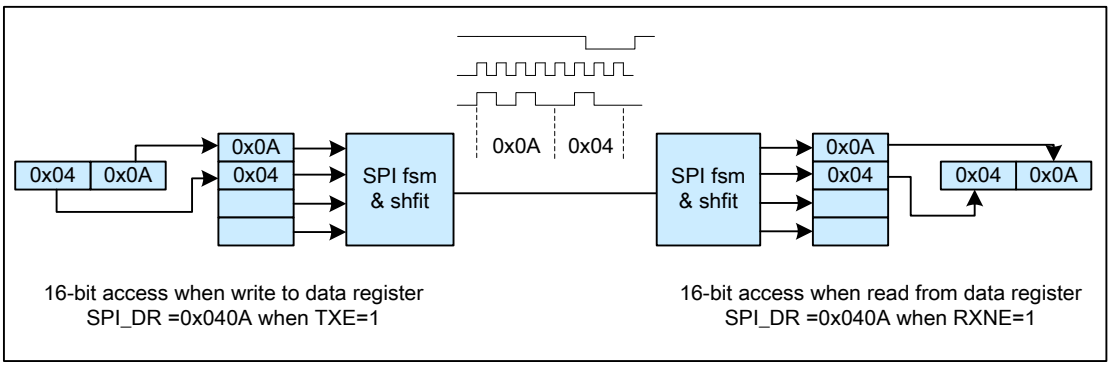


图 27.9 发送和接收 FIFO 中的数据打包

使用 DMA（直接内存访问）通信

为了使 SPI 运行在其最高通信速度，并且在方便数据寄存器读/写过程的同时避免溢出，SPI 需要用 DMA 支持，它实现了一个简单的请求/应答协议。

当在 SPIx_CR2 寄存器的 TXE 或 RXNE 位被置 1 时，可以产生一个 DMA 访问请求。TX 和 RX 缓冲区的 DMA 请求是单独的。

- 在发送中，每次的 TXE 被置为 1，发出 DMA 请求；然后 DMA 向 SPIx_DR 寄存器写数据。
- 在接收中，每次的 RXNE 被置为 1，发出 DMA 请求；然后 DMA 向 SPIx_DR 寄存器读数据。

当 SPI 仅用于发送数据时，可以只打开 SPI TX DMA 通道。在这种情况下，溢出标志会不断置 1，因为收到的数据不会被读取。当 SPI 仅用于接收数据时，可以只打开 SPI RX DMA 通道。

在发送模式下，当 DMA 传输完所有要发送的数据时，DMA 控制器中 DMA_ISR 寄存器的 TCIF 标志会被置 1；监视 BSY 标志可以确认 SPI 通信是否结束。这样可以在关闭 SPI 或进入停机模式之前避免破坏最后一次传输的数据。软件必须先等待，直到 FTLVL[1:0]=00，然后等待 BSY=0。

当 DMA 开始通信时，为了避免发生 DMA 通道管理错误事件，必须遵循以下步骤：

1. 如果用 DMA 接收数据，那么配置 SPI_CR2 寄存器中的 RXDMAEN，使能 DMA 接收缓冲区。
2. 通过 DMA 中的寄存器开启 TX 和 RX 的 DMA 数据流。
3. 如果用 DMA 发送数据，那么配置 SPI_CR2 寄存器中的 TXDMAEN，使能 DMA 发送缓冲区。
4. 通过置位 SPE，使能 SPI。

关闭 DMA 通信，也必须遵循以下步骤：

1. 通过 DMA 中的寄存器关闭 TX 和 RX 的 DMA 数据流。
2. 通过 SPI 关闭流程正确的禁用 SPI。
3. 清零 SPI_CR2 中的 TXDMAEN 和 RXDMAEN，以关闭 DMA 发送缓冲区和接收缓冲区。

DMA 与打包传输

如果传输由 DMA 来管理(TXDMAEN 和 RXDMAEN 在 SPIx_CR2 寄存器设置),会根据 SPI 发送和接收 DMA 通道的 SPI 配置状态来自动启用/禁用数据打包功能。如果 DMA 通道设置按照 16 位访问，而 SPI 数据的位宽小于等于 8 位，那么数据打包模式会被启用。DMA 会自动管理对 SPIx_DR 寄存器的访问操作。

如果启用了数据打包模式，而传输的数据个数又不是偶数，则 LDMA_TX/LDMA_RX 位必须置 1。这样 SPI 才会认为在最后的 DMA 传输中只有一个有效的发送或接收字节。

通信图解

一些典型的时序组合在这个章节会有详细解释。不管 SPI 是由查询方式处理、中断方式处理还是 DMA 方式处理 ,这些时序组合都是有效的。在下面的图解中 ,为了方便说明 ,都假设 LSBFIRST=0 ,CPOL=0 和 CPHA=1。

对于图 27.10 和图 27.11 而言，有以下几点需要注意：

1. 当 NSS 有效以及 SPI 使能后，从机开始控制 MISO 引脚；如果其中一个信号是无效的，那么 MISO 就会断开连接。在主机开始发起传输之前，需要有足够的时间给从机准备数据。
2. 在主机模式下，连续通信时（SCK 时钟持续输出），BSY 标志在帧与帧之间一直为高。BSY 标志在帧与帧之间至少会有一个 SPI 时钟周期为 0。
3. 只有在 TXFIFO 满的时候，TXE 信号才会被清零。
4. 在 TXDMAEN 置位后，DMA 仲裁处理才开始工作。在 TXEIE 置位后，TXE 中断才能产生。如果 TXE 处于有效电平，数据会一直传输到 TXFIFO 中，直到 TXFIFO 满或 DMA 传输结束。
5. 如果所有数据能被放入 RXFIFO 中，那么在 SPI 开始传输之前，DMA 发送 TCIF 标志就会被置位。这个标志通常会在 SPI 完成传输之前置位。
6. 一个数据包中，CRC 值会被一帧一帧的进行计算。CRC 信息会在整个数据包完成后被处理 – 通过 DMA 的方式自动处理 (发送端必须设置正确的需要被处理的数据数量) 或通过软件的方式处理 (用户必须在最后一帧数据期间设置 CRCNEXT 位)。

   存储在 SPIx_TXCRC 中的 CRC 值通过发送端发送出去后，接收到的 CRC 信息被存放入 RXFIFO 中，并且会与 SPIx_RXCRC 中的内容进行比较（如果两个值不同，CRC 错误标志会被置位)。这就需要用户小心的去处理在 FIFO 中的信息 – 通过软件的方式读取所有存储在 RXFIFO 中的数据 ,或者通过 DMA 设置正确的数据帧数目（数据帧数目+CRC 的帧数目）去读取。
7. 在数据打包模式下， TXE 和 RXNE 是成对出现的并且每次对 FIFO 的读写访问都是 16 位宽的，在数据帧数目是偶数的时候 ,甚至 TXFIFO 满的时候是 TXFIFO 3/4 的水平 ;这就是为什么不能在 1/2TXFIFO 满之前存入一个字节数据的原因 。这个数据帧存入 TXFIFO 中需要通过 8 位数据访问的方式 – 通过软件的方式写入，或通过 DMA 方式时设置 LDMA_TX 位自动写入。
8. 在数据打包模式下，接收最后一个奇数数据帧时必须配置 RXFIFO 阈值为 8 位 – 通过软件的方式设置 FRXTH=1，或通过 DMA 方式时设置 LDMA_RX 位自动变为 8 位访问。

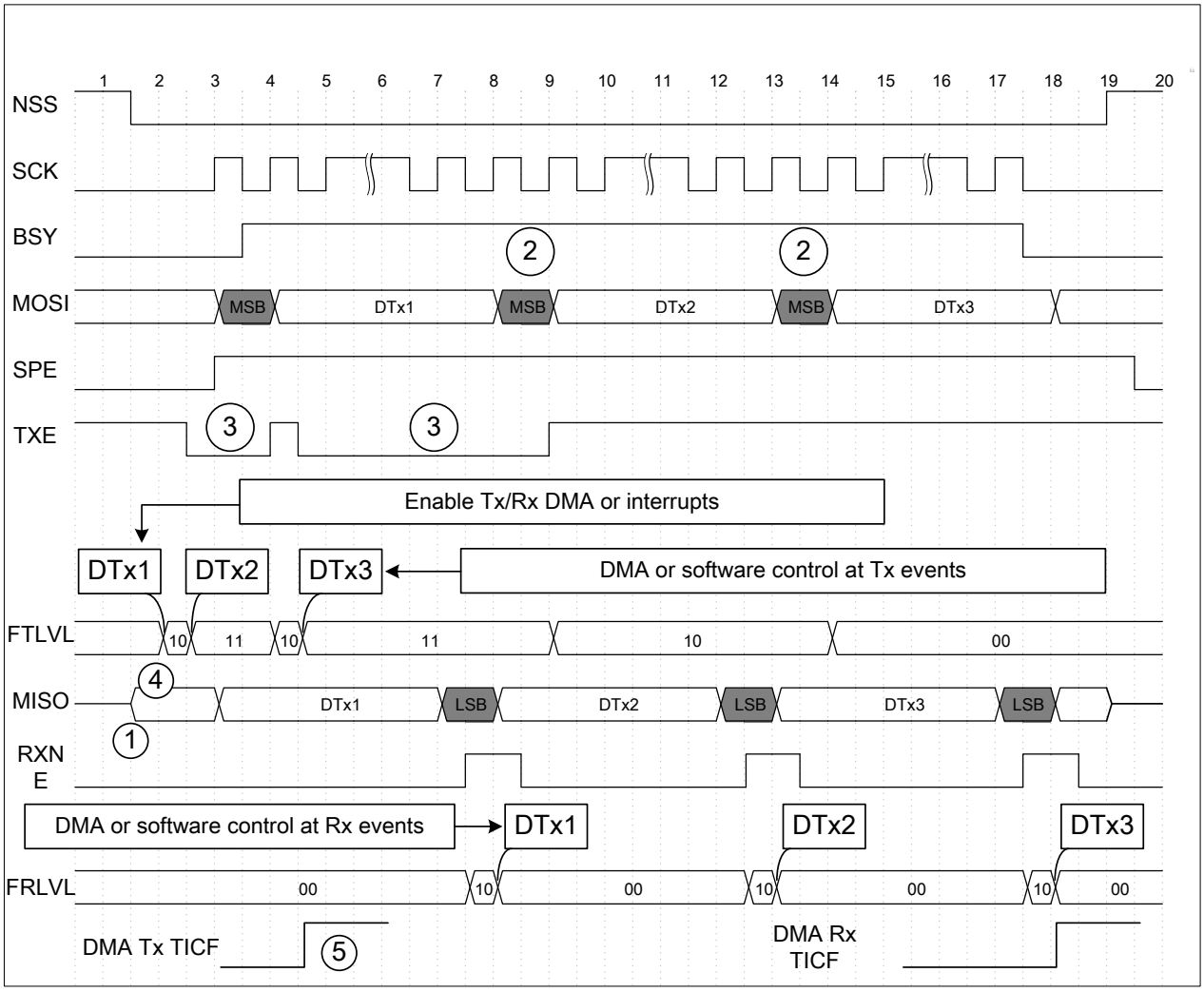


图 27.10 主机全双工通信

主机全双工通信所有的假设如下：

- 数据帧字长 > 8 位

如果使用 DMA：

- DMA 中发送数据帧数目设置为 3
- DMA 中接收数据帧数目设置为 3

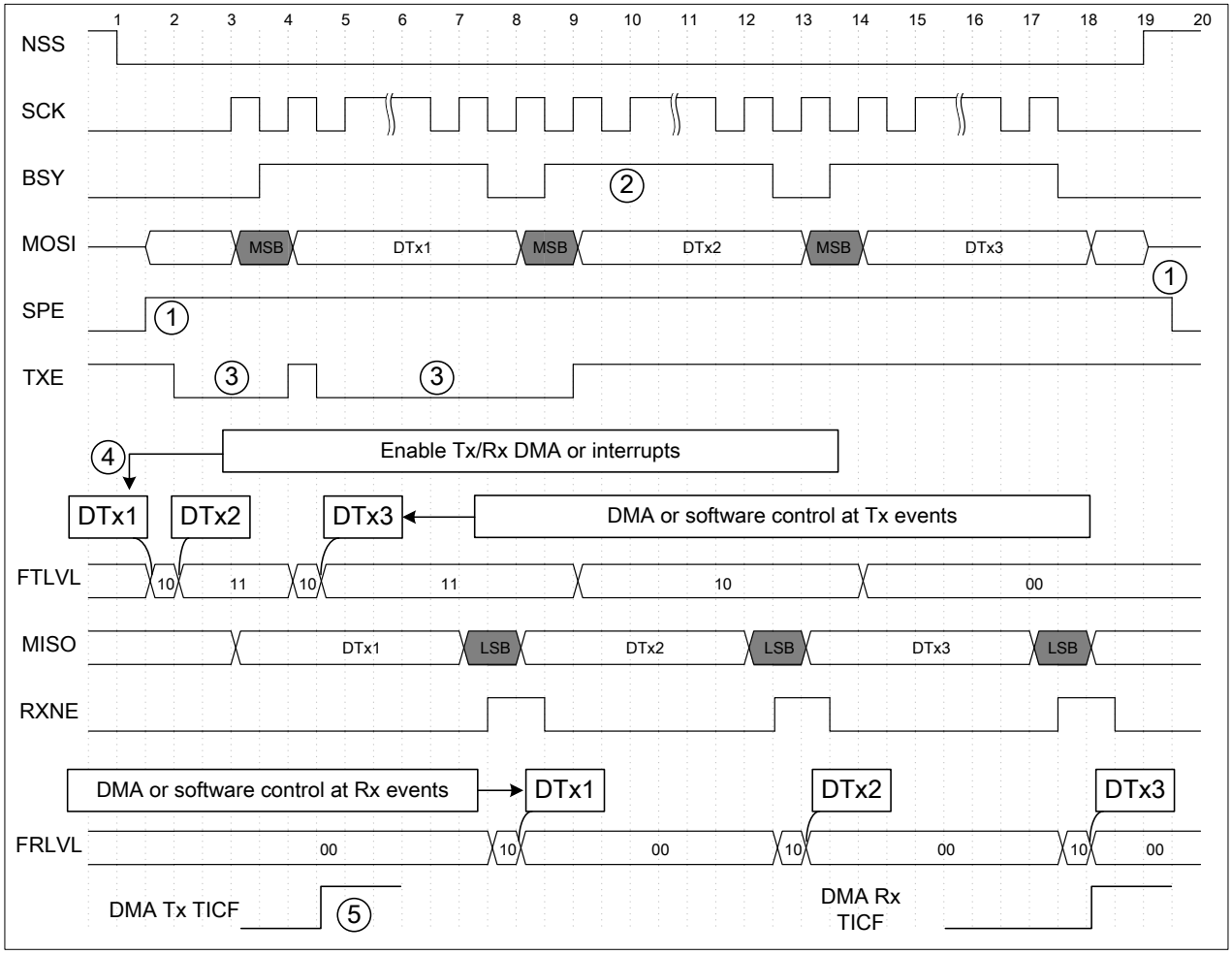


图 27.11 从机全双工通信

从机全双工通信所有的假设如下：

- 数据帧字长 > 8 位

如果使用 DMA：

- DMA 中发送数据帧数目设置为 3
- DMA 中接收数据帧数目设置为 3

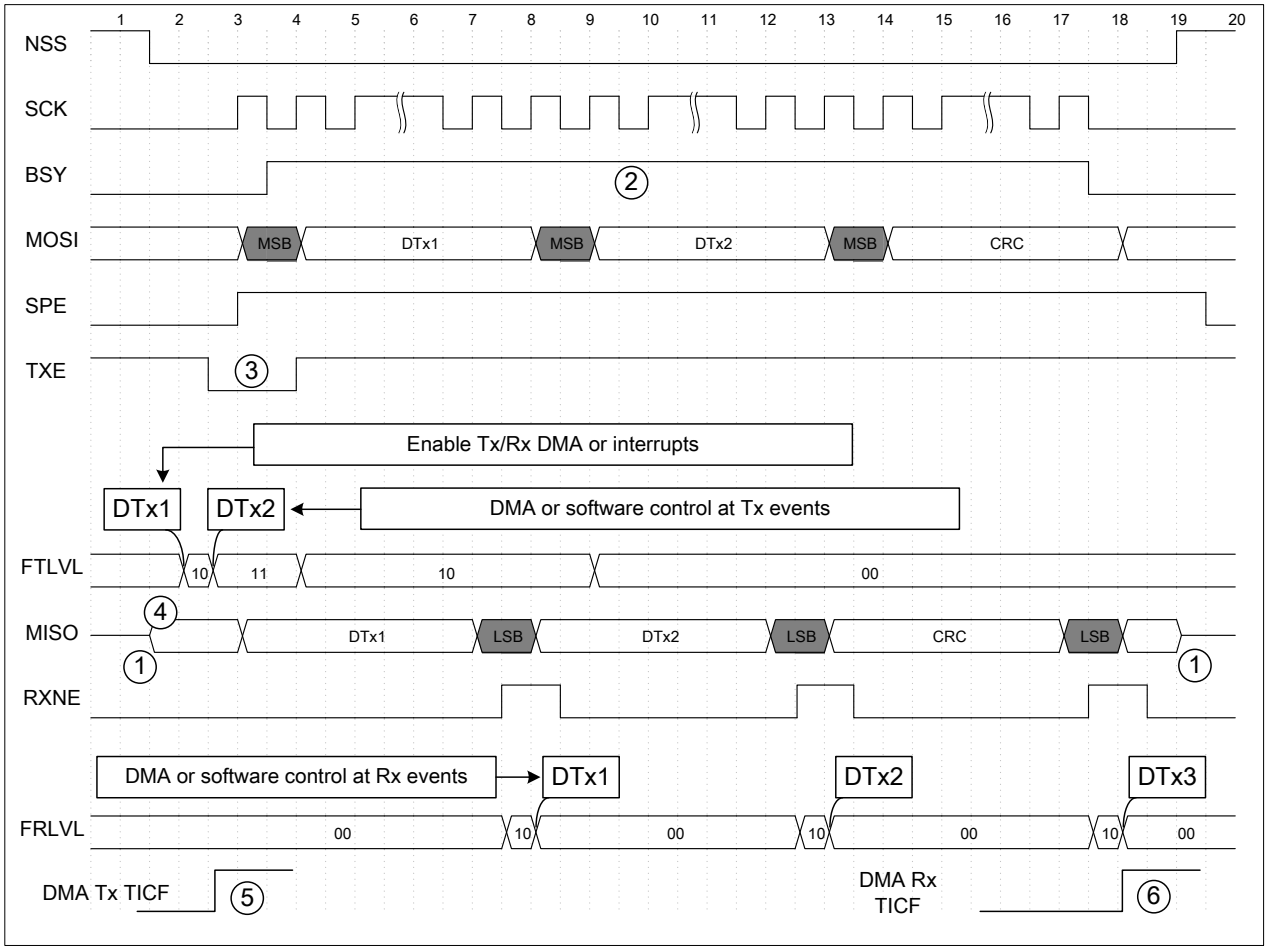


图 27.12 带 CRC 功能的主机全双工通信

主机全双工通信所有的假设如下：

- 数据帧字长 = 16 位
- 开启 CRC 功能

如果使用 DMA：

- DMA 中发送数据帧数目设置为 2
- DMA 中接收数据帧数目设置为 3

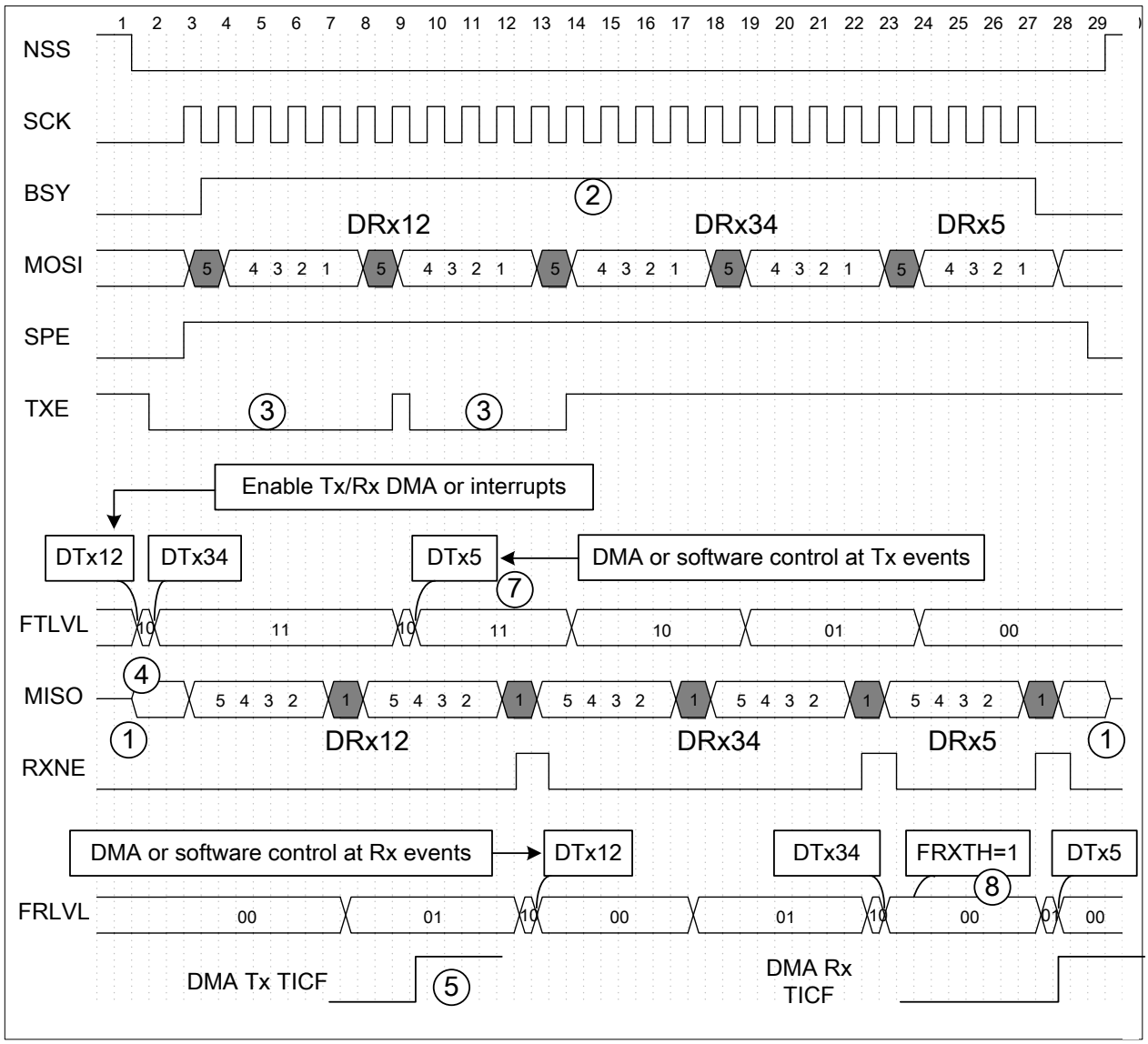


图 27.13 带数据打包功能的主机全双工通信

主机全双工通信所有的假设如下：

- 数据帧字长 = 5 位
- 对 FIFO 的读写访问大部分是 16 位的
- FRXTH=0

如果使用 DMA：

- DMA 中发送数据帧数目设置为 3
- DMA 中接收数据帧数目设置为 3
- 发送通道和接收通道的 PSIZE 都设置为 16 位
- LDMA_TX=1 和 LDMA_RX=1

27.3.9. SPI 的状态标志

应用程序可通过下面 3 个状态标志来了解 SPI 总线的全部状态。

TX 缓冲空标志（TXE）

当 TXFIFO 中有足够的空间来存储发送的数据时，TXE 标志被置 1。TXE 标志是跟 TXFIFO 水平相关联的。在 TXFIFO 的存储水平低于或等于整个 FIFO 深度的 1/2 时，这个标志会置高并且保持为高。如果 SPIx_CR2 寄存器中的 TXEIE 位是 1，还会产生一个中断。如果 TXFIFO 的存储水平又高于 FIFO 深度的 1/2 了，那么这位会自动的清零。

RX 缓冲非空标志（RXNE）

RXNE 标志的置位取决于 SPIx_CR2 寄存器中 FRXTH 位的设置值：

- 如果 FRXTH 为 1，RXNE 会在 RXFIFO 的存储水平大于或等于 1/4(8 位)的时候被置高并且保持为高。
- 如果 FRXTH 为 0，RXNE 会在 RXFIFO 的存储水平大于或等于 1/2(16 位)的时候被置高并且保持为高。

如果这是 SPIx_CR2 寄存器中的 RXNEIE 位是 1，那就会产生一个中断请求。

当上述的条件不再成立时，RXNE 由硬件自动清零。

忙标志（BSY）

BSY 标志由硬件置位和清零（软件无法改写这个标志）。

当 BSY 为 1 时，它表示 SPI 正处于数据传输过程中（SPI 总线忙）。

在某些模式下可以使用 BSY 标志来检测传输结束，从而软件可以在进入 HALT 模式之前禁用 SPI 及其外设时钟。这就避免了破坏最后的传输字节。

BSY 标志在防止多主机系统中的写冲突也是很有用的。

BSY 标志在下列条件之一达成时被清除：

- 当 SPI 被正确禁用时
- 当在主机模式中检测到故障时（MODF 位被硬件置 1）
- 在主机模式下，当完成了数据发送，并且没有新的数据要发送时
- 在从机模式下，当两个数据帧传输之间间隔至少要有一个 SPI 时钟周期时

注意：当下次传输可立即由主机来处理时（例如：如果主机是只接收模式或它的 TXFIFO 不为空），在主机这边的发送数据之间通信不会中断并且 BSY 标志仍然保持为 1。尽管从机不会有这种情况，总是建议使用 TXE 和 RXNE 标志（而不是 BSY 标志）来处理数据发送或接收的操作。

27.3.10. SPI 错误标志

在将错误中断使能位 ERRIE 置 1 后，如果下列错误标志其中一个被置 1，就会产生 SPI 中断。

溢出标志（OVR）

当主机或从机接收数据时，RXFIFO 没有足够的空间存储新收到的数据，那么溢出的情况就会发生。如果软件或者 DMA 没有足够的时间去读取 RXFIFO 之前接收到的数据来为后面接收新数据腾出足够的空间时，这种情况就会发生。例如，在只接收模式下并且 CRC 功能打开时，接收缓冲区 RXFIFO 会被限制为深度为 1 的单缓冲区，这是不读取之前收到的数据，在接收新数据时就会发生溢出。

溢出的情况发生时，新收到的数据不会覆盖 RXFIFO 中的前一个数据。新收到的数据将被丢弃，随后传输的

所有数据都将丢失。在读一次 SPIx_DR 寄存器后，紧随着读一次 SPIx_SR 寄存器，就会清除 OVR 标志。

模式故障（MODF）

主机得知其内部 NSS 信号（NSS 引脚为硬件模式或在 NSS 软件模式 SSI 位）被拉低时，发生模式故障，这将自动置起 MODF 位。主机模式故障对 SPI 接口的影响包括以下方面：

- MODF 位被置 1；另外如果 ERRIE 位被置 1，会产生 SPI 中断。
- SPE 位被清零。这将阻止主机的数据输出，同时禁用 SPI 接口。
- MSTR 位被清除，从而迫使设备转到从机模式。

使用下面的软件序列，以清除 MODF 位：

1. 在 MODF 位被置起后，对 SPIx_SR 寄存器进行一次读或写访问。
2. 然后写一下 SPIx_CR1 寄存器。

为了避免系统中的多个从机发生冲突，NSS 引脚必须在 MODF 位清零过程中拉高。作为一种安全措施，硬件不允许在 MODF 位为 1 期间将 SPE 的和 MSTR 位置 1。在从机模式下，MODF 位永远都不可能被置 1，除非是以前的多主机冲突的结果。

CRC 错误（CRCERR）

在 SPIx_CR1 寄存器中的 CRCEN 位为 1 时，这个标志用来确认收到的数据的有效性。如果接收移位寄存器中的值和接收端本身的 SPIX_RXCRCR 寄存器的值不匹配，SPIx_SR 寄存器中的 CRCERR 标志会被置 1。该标志由软件清除。

TI 模式帧格式错误（FRE）

当 SPI 工作在从机模式并且配置为 TI 协议时，如果数据通信期间，在 NSS 上出现一个脉冲，会引起一次 TI 模式帧格式错误。发生此错误时，SPIx_SR 寄存器中的 FRE 标志会被置 1。在发生错误时，SPI 不会被禁止而 NSS 脉冲会被忽略，SPI 在开始新的传输前需要等待下一个 NSS 脉冲。帧格式错误的产生可能会导致两个字节的数据丢失，数据可能会损坏。

读 SPIx_SR 寄存器，FRE 标志会被清除。如果 ERRIE 位为 1，会在 NSS 错误检测后产生一个中断。在这种情况下，SPI 应该被禁用，因为数据的完整性将不能保证了；应该通过先重新使能从机再重新初始化主机的方式重新开始通信。

27.3.11. NSS 脉冲模式

在 SPIx_CR1 寄存器的 NSSP 位被置 1 时打开这个模式，只有在 SPI 接口被配置为摩托罗拉主模式(FRF=0) 并且捕获第一个沿（CPHA=0）时才会生效。这时，NSS 脉冲会在两个连续的数据帧之间产生并且 NSS 至少会在高电平保持一个时钟周期。这个模式允许从机锁存数据。NSSP 脉冲模式为单一的主从应用而设计。

图 27.14 说明 NSSP 脉冲模式时启用的 NSS 引脚管理情况。

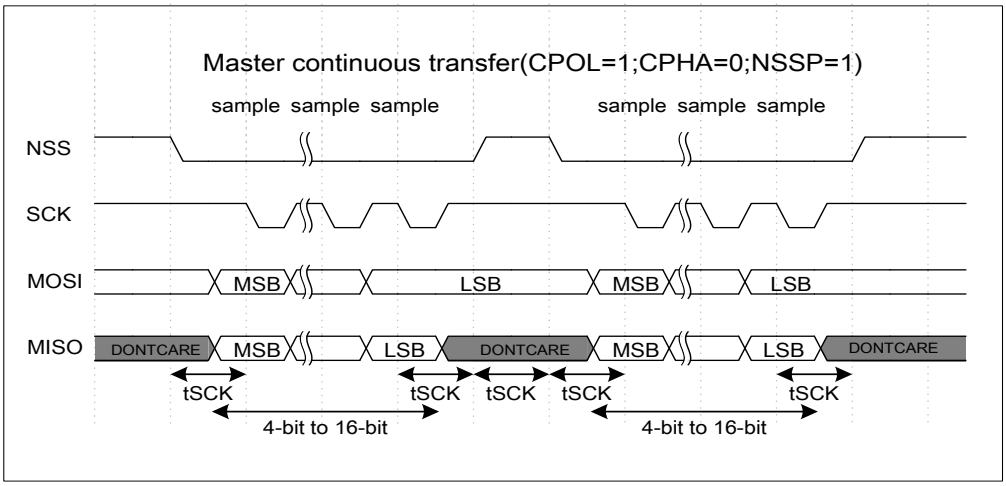


图 27.14 摩托罗拉 SPI 主模式下 NSSP 脉冲的产生

注意：当 CPOL=0 时动作也是类似的。这时，采样边沿是在 SCK 的上升沿，对 NSS 的采样也发生在这些沿。

27.3.12. TI 模式

TI 协议主模式

SPI 接口兼容 TI 协议。SPIx_CR2 寄存器中的 FRF 位可以用来令 SPI 兼容这个协议。

无论 SPIx_CR1 寄存器怎么设置，时钟极性和相位都会强制符合 TI 协议要求。NSS 管理也按照 TI 协议，这使得在这种情况下通过 SPIx_CR1 和 SPIx_CR2 寄存器（SSM，SSI，SSOE）不可能配置 NSS 管理。

在从机模式下，SPI 波特率分频器用于控制 MISO 引脚状态变化为高阻的时刻。任何波特率都可用，这使其能够确定最佳的灵活性。然而，波特率一般设置为外部主时钟的波特率。MISO 变为高阻的延时取决于信号的内部同步过程和通过 SPIx_CR1 寄存器中的 BR[2:0]所设置的波特率值，如下面公式所示：

$$\frac{t_{baud\_rate}}{2} + 4 * t_{pclk} < t_{release} < \frac{t_{baud_{rate}}}{2} + 6 * t_{pclk}$$

如果在数据通信过程中，从机检测到错位的 NSS 脉冲则会使 FRE 标志置位。

协议规定，如果数据字长为 4 位或 5 位时并且在主机全双工模式或只发送模式下时，传输会在最后一位数据后增加一个冗余位。此时，NSS 脉冲会在最后一个冗余位时产生，而不是在最后一个数据位上。

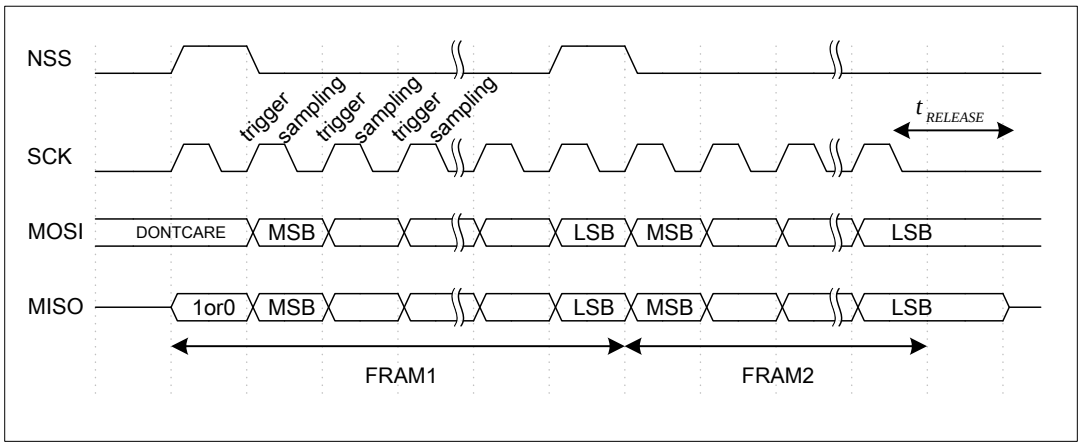


图 27.15 TI 模式传输

27.3.13. CRC 计算

有两个独立的 CRC 计算器，分别用以检查发送和接收的数据的可靠性。SPI 提供 CRC8 或 CRC16 计算，独立于帧的数据位宽，但数据位宽只能固定设置为 8 位或 16 位。对于其他所有的数据位宽，CRC 无效。
在使能 SPI 之前（SPE=0）通过设置 SPIx_CR1 寄存器中的 CRCEN 位，使能 CRC 计算。CRC 值由一个奇次可编程多项式按数据帧计算，是并行 CRC 计算，而不是串行 CRC 计算。在发送端，CRC 在 TXFIFO 推出数据到移位寄存器时进行计算；而在接收端，CRC 在数据从移位寄存器推进 RXFIFO 时进行计算。在数据块传输结束的同时，计算出的 CRC 结果自动参与检查，不管是由 CPU 传输还是由 DMA 控制传输。当发现 CRC 不匹配时，CRCERR 标志被置 1，表示数据传输出错。对 CRC 计算的正确流程取决于 SPI 配置和所选择的传输管理。
注意：多项式的值应该是奇数，不支持偶数值。

由 CPU 管理时的 CRC 传输

在最后一个数据发送或接收完之前，SPI 的启动和传输都没什么特别的。要在当前传输的数据之后跟上 CRC 数据，必须将 SPIx_CR1 中的 CRCNEXT 位设置为 1.设置 CRCNEXT 位必须在最后一个数据帧交易结束之前完成。在 CRC 数据传输期间，CRC 计算会被冻结。
接收到的 CRC 数据向正常的数据字节或字存储在 RXFIFO 中。这就是在 CRC 模式并且只接收模式时，任何时候接收缓冲区都必须被认为是单个 16 位的缓冲区并被用来接收单个数据的原因。
CRC 数据交换，通常需要在数据序列结束后再占用一个或多个数据通信的时间。当设置为 8 位的数据宽度并且做 16 位 CRC 检查时，发送完整的 CRC 数据就要两帧。
当收到最后的 CRC 数据后，会自动将收到的值和在 SPIx_RXCRCR 寄存器的值进行比较。软件必须监测 SPIx_SR 寄存器中的 CRCERR 标志以判断传输过程中数据是否被破坏。软件对 CRCERR 标志写 0 就可清除它。
CRC 数据收到后，被存储在 RXFIFO 中，必须读 SPIx_DR 寄存器以清除 RXNE 标志。

由 DMA 管理时的 CRC 传输

当 SPI 通信功能中的 CRC 功能和 DMA 功能同时打开后，通信尾部的 CRC 数据的发送和接收都是全自动的。CRCNEXT 位不再由软件处理。SPI 发送 DMA 通道计数器必须设置为不包含 CRC 数据的数量。而接收 DMA 通道计数器则要多包含一个 CRC 数据的长度，例如在 8 位数据宽度传输并使用 16 位 CRC 计算时：

$$\mathrm{DMA_{RX}} = \text{数据字节数} + 2$$

在只接收模式下，接收 DMA 通道计数器只包含数据传输数量，而不包括 CRC 数据。基于这种配置，所有 CRC 值必须通过软件的方式从 RXFIFO 中取出。
在全部传输完成后，如果发现有错误，SPIx_SR 寄存器中的 CRCERR 标志会被置 1。

SPIx_TXCRCR 和 SPIx_RXCRCR 值的复位

在 CRC 传输过后，如果有新的数据进行传输，SPIx_TXCRCR 和 SPIx_RXCRCR 值会被自动清零。这就允许使用 DMA 循环模式（只接收模式除外）来实现没有任何间断的数据传输（几个数据中间涉及 CRC 检查阶段）。
如果在通信过程中禁用了 SPI，那么需要遵循以下顺序来重新开启：

1. 禁止 SPI。
2. 清除 CRCEN 位。
3. 使能 CRCEN 位。

4. 使能 SPI。

注意：为了避免任何错误的 CRC 计算，软件必须在时钟稳定的条件下启用 CRC 计算。当 SPI 配置为从机接口时，数据阶段和 CRC 阶段之间，NSS 内部信号需要保持低。

27.4. SPI 中断

在 SPI 通信期间，中断可以由以下事件来产生：

- 发送 TXFIFO 为空，待填装
- 在接收 RXFIFO 中有收到的数据
- 主模式故障
- 溢出错误
- TI 帧格式错误
- CRC 错误

中断可以分别的启用和禁用。

表 27.1 SPI 中断请求

中断事件	事件标志	使能控制位
发送 TXFIFO 待填装	TXE	TXEIE
在接收 RXFIFO 中有收到的数据	RXNE	RXNEIE
主模式故障	MODF	ERRIE
溢出错误	OVR	
TI 帧格式错误	FRE	
CRC 错误	CRCERR	

27.5. SPI 寄存器映射

可以用半字(16 位)或字(32 位)的方式操作这些外设寄存器。此外 SPI_DR 寄存器可以用 8 位的方式访问。

下表给出了 SPI 寄存器的映射以及复位值。

0x00	SPIx_CR1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	CRCL	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPIx_CR2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LDMA_TX	LDMA_RX	FRXTH	DS[3:0]				TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	TXDMAEN	RXDMAEN
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0x08	SPIx_SR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	FTLVL[1:0]		FRLVL[1:0]		FRE	BSY	OVR	MODF	CRCERR	-	-	TXE	RXNE
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	x	x	1	0
0x0C	SPIx_DR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DR[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SPIx_CRCPR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CRCPOLY[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0x14	SPIx_RXCRCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RXCRC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	SPIx_TXCRCR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TXCRC[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

27.5.1. SPI 控制寄存器 1（SPIx_CR1）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	CRCL	RXONLY	SSM	SSI
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:16	NA	保留位，未定义
15	BIDIMODE	双向数据模式使能 此位使能只使用一条双向数据线的半双工通信方式。使能半双工模式时，应该保持 RXONLY 位为 0. 0：选择 2 线的单向数据模式（全双工模式）。 1：选择单线双向数据模式（半双工模式）。
14	BIDIOE	双向模式输出使能

		和 BIDIMODE 位一起决定在半双工模式下的输出方向 0：输出禁止（只接收模式）。 1：输出使能（只发送模式）。
13	CRCEN	硬件 CRC 计算使能 0：CRC 计算禁用 1：CRC 计算使能
12	CRCNEXT	下一个发送 CRC 0：下一个发送的值来自发送缓冲区 TXFIFO。 1：下一个发送的值来自发送 CRC 寄存器 SPIx_TXCRCR。
11	CRCL	CRC 长度 由软件设置和清零，用于选择 CRC 长度。 0：8 位 CRC 长度。 1：16 位 CRC 长度。
10	RXONLY	只接收模式使能（简单模式下） 此位使能用单向单线的简单传输去接收数据。在此只接收模式使能的时候需要保持 BIDIMODE 位为 0。在多从设备的配置中，在未被访问的从设备上设置该位为 1，使得只有被访问的从设备有输出，从而不会造成数据线上的数据冲突。 0：全双工模式（发送和接收）。 1：输出禁止（只接收模式）。
9	SSM	软件从机管理 当 SSM 被置位时，NSS 引脚上的电平由 SSI 位的值决定。 0：禁止软件从设备管理。 1：启用软件从设备管理。 注意：在 SPI TI 模式下，此位不可用。
8	SSI	内部从设备选择 此位只有当 SSM 位为 1 的时候才有效果。它决定了 NSS 上的电平，在 NSS 引脚上的 I/O 操作无效。 注意：在 SPI TI 模式下，此位不可用。
7	LSBFIRST	帧格式 0：先发送 MSB。 1：先发送 LSB。 注意：1. 当通信在进行时不能改变该位的值。 2. 在 SPI TI 模式下，此位不可用。
6	SPE	SPI 使能 0：禁止 SPI 设备。 1：开启 SPI 设备。 注意：当关闭 SPI 设备时，请按照前面章节所述的操作流程关闭 SPI。
5:3	BR[2:0]	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：$f_{PCLK}/256$ 注意：当通信在进行时不能改变该位的值。
2	MSTR	主设备选择 0：配置为从设备。 1：配置为主设备。 注意：当通信在进行时不能改变该位的值。
1	CPOL	时钟极性 0：空闲状态时，SCK 保持为低电平。 1：空闲状态时，SCK 保持为高电平。 注意：1. 当通信在进行时不能改变该位的值。 2. 在 SPI TI 模式下，此位不可用。
0	CPHA	时钟相位 0：第一个时钟沿对准第一位数据。 1：第二个时钟沿对准第一位数据。 注意：1. 当通信在进行时不能改变该位的值。 2. 在 SPI TI 模式下，此位不可用。

27.5.2. SPI 控制器 2（SPIx_CR2）

地址偏移：0x04

复位值：0x0700

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	保留位	LDMA_TX	LDMA_RX	FRXTH	DS[3:0]			
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	TXDMAEN	RXDMAEN
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:15	NA	保留位，未定义
14	LDMA_TX	最后一次 DMA 发送 此位是在数据打包模式中，用来定义 DMA 数据传输的总数是奇数还是偶数。它只有在 SPIx_CR2 寄存器中的 TXDMAEN 位被置位，并且数据打包模式已开启（数据长度小于等于 8 位和写入访问 SPIx_DR 是 16 位宽）时才有意义。它必须在 SPI 被禁止（SPE=0）时写入。 0：传输的数据数量为偶数。

		1：传输的数据数量为奇数。 注意：如果 CRCEN 位为 1 时，请按照前面章节所述的操作流程关闭 SPI。
13	LDMA_RX	最后一次 DMA 接收 此位是在数据打包模式中，用来定义 DMA 数据接收的总数是奇数还是偶数。它只有在 SPIx_CR2 寄存器中的 RXDMAEN 位被置位，并且数据打包模式已开启（数据长度小于等于 8 位和写入访问 SPIx_DR 是 16 位宽）时才有意义。它必须在 SPI 被禁止（SPE=0）时写入。 0：传输的数据数量为偶数。 1：传输的数据数量为奇数。 注意：如果 CRCEN 位为 1 时，请按照前面章节所述的操作流程关闭 SPI。
12	FRXTH	FIFO 接收门限 此位用来设置触发 RXNE 事件时的 RXFIFO 的阈值。 0：如果 FIFO 的存储水平大于或等于 1/2（16 位），产生 RXNE 事件。 1：如果 FIFO 的存储水平大于或等于 1/4（8 位），产生 RXNE 事件。
11:8	DS[3:0]	数据位宽 这些位配置 SPI 传输数据的位宽： 0000：不使用 0001：不使用 0010：不使用 0011：4 位 0100：5 位 0101：6 位 0110：7 位 0111：8 位 1000：9 位 1001：10 位 1010：11 位 1011：12 位 1100：13 位 1101：14 位 1110：15 位 1111：16 位 如果软件试图写入一个不使用的值，那 DS 会被迫赋值为 0111（8 位数据）。
7	TXEIE	发送缓冲区 TXFIFO 空中断使能 0：TXE 中断屏蔽。 1：TXE 中断没有被屏蔽。用于在 TXE 标志置 1 的时候产生一个中断请求。
6	RXNEIE	接收缓冲区 RXFIFO 非空中断使能 0：RXNE 中断屏蔽。 1 :RXNE 中断没有被屏蔽。用于在 RXNE 标志置 1 的时候产生一个中断请求。
5	ERRIE	错误中断使能 此位控制在出现错误事件（CRCERR、OVR、MODF 和 FRE）时是否产生中断。 0：错误中断屏蔽。

		1：错误中断使能。
4	FRF	帧格式 0：SPI 摩托罗拉模式 1：SPI TI 模式 注意：该位必须在 SPI 被禁止（SPE=0）时写入。
3	NSSP	NSS 脉冲管理 该位在主设备中配置为 1，使能 NSSP 模式；而从设备中可配置为 1，也可配置为 0。它允许 SPI 在连续传输时，两个数据传输之间产生一个 NSS 脉冲。在单个数据传输的情况下，它会在传输结束后将 NSS 脚强制为高电平。 在 CPHA=1 或 FRF=1 的时候，此位置位没有意义。 0：没有 NSS 脉冲。 1：产生 NSS 脉冲。 注意：1. 该位必须在 SPI 被禁止（SPE=0）时写入。 2. 在 SPI TI 模式下，此位不可用。
2	SSOE	SS 输出使能 0：在主机模式下 SS 输出被禁用，SPI 接口可以工作在多主机的配置下。 1 :SPI 接口启用的同时在主机模式下启用 SS 输出。SPI 接口不能工作在多主机环境下。 注意：在 SPI TI 模式下，此位不可用。
1	TXDMAEN	发送缓冲区 TXFIFO 的 DMA 请求使能 当此位被置位并且 TXE 标志变高时，产生一个 DMA 请求。 0：发送缓冲区 TXFIFO 的 DMA 请求禁止。 1：发送缓冲区 TXFIFO 的 DMA 请求使能。
0	RXDMAEN	接收缓冲区 RXFIFO 的 DMA 请求使能 当此位被置位并且 RXNE 标志变高时，产生一个 DMA 请求。 0：接收缓冲区 RXFIFO 的 DMA 请求禁止。 1：接收缓冲区 RXFIFO 的 DMA 请求使能。

27.5.3. SPI 状态寄存器（SPIx_SR）

地址偏移：0x08

复位值：0x0002

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			FTLVL[1:0]		FRLVL[1:0]		FRE
类型	RO-0	RO-0	RO-0	RW	RO	RO	RO	RO
7:0	BSY	OVR	MODF	CRCERR	—		TXE	RXNE
类型	RO	RO	RO	RC_W0	RO-0	RO-0	RO	RO

Bit	Name	Function
31:13	NA	保留位，未定义
12:11	FTLVL[1:0]	发送缓冲区 TXFIFO 存储水平 由硬件置位或清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满（当 FIFO 存储水平大于 1/2 时认为是满）
10:9	FRLVL[1:0]	接收缓冲区 RXFIFO 存储水平 由硬件置位或清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满 注意：在 CRC 计算使能并且 SPI 只接收模式时，这些标志不使用。
8	FRE	TI 帧格式错误 此标志只用在从机的 TI 模式下。此位由硬件置位，通过软件读 SPIx_SR 寄存器清零。 0：没有发生帧格式错误。 1：发生了一个帧格式错误。
7	BSY	忙标志 0：SPI 不忙。 1：SPI 通信忙或者 TXFIFO 发送不为空。 注意：必须谨慎使用 BSY 标志：参考 SPI 的状态标志章节和 SPI 的关闭流程章节。
6	OVR	溢出标志 0：没有发生溢出。 1：发生溢出。 此标志由硬件置位，由软件序列复位。软件序列请参考 SPI 错误标志章节。
5	MODF	模式故障 0：无模式故障发生。 1：模式故障发生。 此标志由硬件置位，由软件序列复位。软件复位请参考 SPI 错误标志章节。
4	CRCERR	CRC 错误标志 0：接收到的 CRC 值和 SPIx_RXCRCR 的值是匹配的。 1：接收到的 CRC 值和 SPIx_RXCRCR 的值不匹配。
3:2	NA	保留位，未定义
1	TXE	发送缓冲区 TXFIFO 为空标志 0：TXFIFO 非空。 1：TXFIFO 空。
0	RXNE	接收缓冲区 RXFIFO 非空标志 0：RXFIFO 空。 1：RXFIFO 非空。

27.5.4. SPI 数据寄存器（SPIx_DR）

地址偏移：0x0C

复位值：0x0000

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	DR[15:0]	数据寄存器 待发送或已经接收到的数据 数据寄存器作为 RXFIFO 和 TXFIFO 的接口。当读取数据寄存器时，RXFIFO 被访问，而写入数据寄存器则会访问 TXFIFO（见数据发送和接收流程章节）。 注意：数据始终是右对齐的。当写寄存器时，没有用到的位被忽略；读的时候，没有用到的位会是 0。接收门限的设置必须始终符合目前使用的读访问方式。

27.5.5. SPI 的 CRC 多项式寄存器（SPIx_CRCPR）

地址偏移：0x10

复位值：0x0007

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

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	CRCPOLY[15:0]	CRC 多项式寄存器 该寄存器包含 CRC 计算多项式。

		该寄存器的复位值是 0007H。根据需要，可以设置成另一个多项式。

注意：多项式的值应该是奇数，不支持偶数值。

27.5.6. SPI 接收 CRC 寄存器（SPIx_RXCRCR）

地址偏移：0x14

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	RXCRC[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	RXCRC[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	RXCRC[15:0]	RXCRC 寄存器 当启用 CRC 计算功能 ,RXCRC[15:0]位包含根据收到的字节计算出来的 CRC 值。当 SPIx_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPIx_CRCPR 寄存器中的多项式。 当数据帧格式被设置为 8 位（SPIx_CR1 寄存器中的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 当数据帧格式被设置成 16 位（SPIx_CR1 寄存器中的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

27.5.7. SPI 发送 CRC 寄存器（SPIx_TXCRCR）

地址偏移：0x18

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	TXCRC[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	TXCRC[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
31:16	NA	保留位，未定义
15:0	TXCRC[15:0]	TXCRC 寄存器 当启用 CRC 计算功能 ,TXCRC[15:0]位包含根据发送的字节计算出来的 CRC 值。当 SPIx_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPIx_CRCPR 寄存器中的多项式。 当数据帧格式被设置为 8 位 (SPIx_CR1 寄存器中的 CRCL 位为 0) 时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 当数据帧格式被设置成 16 位 (SPIx_CR1 寄存器中的 CRCL 位为 1) 时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

28. 触摸传感控制器（TSC）

仅 FT32F072 系列芯片拥有此模块。

28.1. 简介

触摸传感控制器提供给应用一个增加电容式触摸传感功能的简单的解决方案。电容式触摸解决方案可以检测与电极接近的手指，从而避免了触摸按键设计中出现的电气直接接触现象。通过检测有手指 (或其它导体) 所引入的电容变化的方法来判断是否存在触摸按键。

28.2. TSC 主要特性

触摸传感控制器包含下列主要功能：

- 可靠的电荷转移采集检测原理
- 支持多达 24 个电容传感通道
- 多达 3 个模拟 I/O 口组，可同时检测 2 组
- 全硬件管理充放电次序
- 充放电频率可配置
- I/O 引脚采样功能可配置
- 通道 I/O 可选择

28.3. TSC 功能描述

28.3.1. 表面电荷迁移采样概述

表面电荷迁移采样是一套经过验证的，可靠有效的测量电容量的方法。利用单端极性和最小数量的外围器件。

该方法利用带有模拟功能的 I/O 来设计完成。若干模拟 I/O 口组可以同时为多个电容传感通道服务从而能够支持更多的电容传感通道。对于同一个模拟 I/O 口组，各个电容传感通道是按顺序采集信号的。
每一个采样电容通道占用一个 GPIO 口。每一个模拟 I/O 口组只能使用一个采样电容 I/O。其余的 GPIO 被用来连接电极，就算是一个通道。

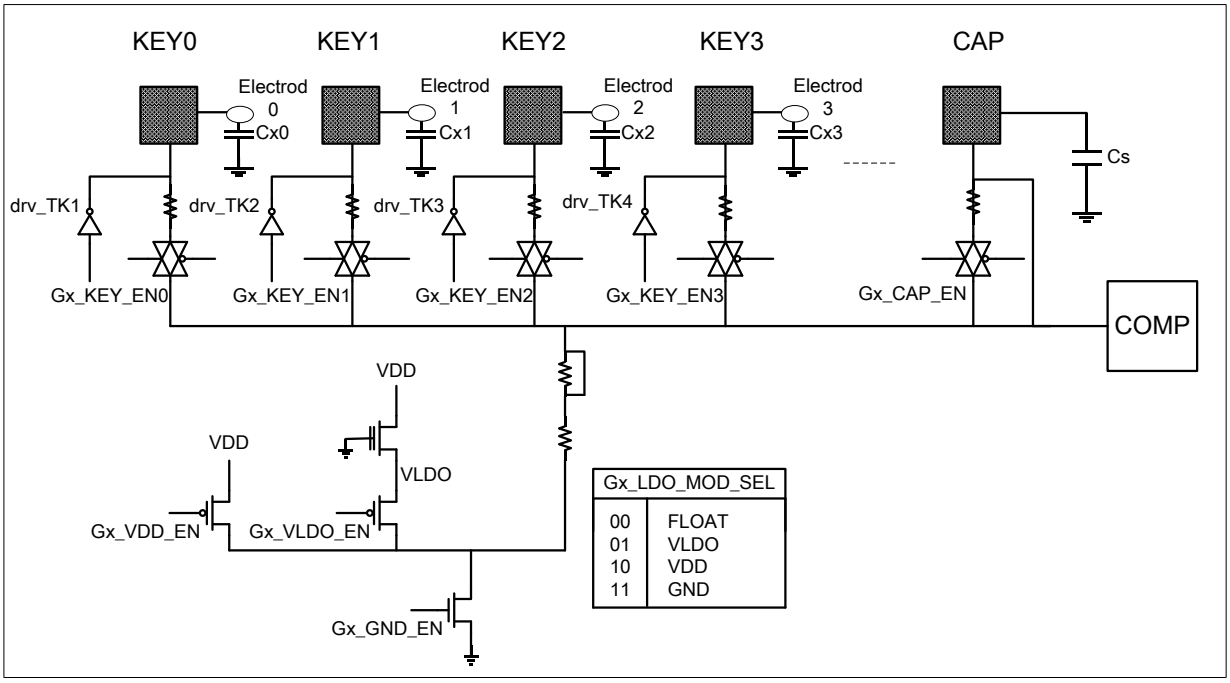


图 28.1 触摸检测原理图

表面电荷迁移检测原理由一个电极电容和一个采样电容转移电荷的通道组成。当采样电容的电压达到一个预设的门限后顺序执行下一个通道的转移过程。充电达到门限所需的电荷量和电极电容量直接相关。

28.3.2. 电荷迁移采集顺序

下表详细列出了电容传感通道的电荷迁移收集顺序。重复执行状态 3 到状态 7 的步骤直到 Cs 电容上的电压达到指定门限。电极串联电阻可以提高该方案的 ESD 免疫力。

表 28.1 采集顺序一览

状态	TSC_Gx_IOy	TSC_Gx_CAP	描述
#1	输入悬空 模拟开关关闭	输出开漏低 模拟开关关闭	放空全部的 Cx 和 Cs 电量
#2	输入悬空	输入悬空	死区时间
#3	输出推拉高	输入悬空	充电 Cx
#4	输入悬空	输入悬空	死区时间
#5	输入悬空 模拟开关关闭	输入悬空 模拟开关关闭	电荷转移从 Cx 到 Cs
#6	输入悬空	输入悬空	死区时间

#7	输入悬空	输入悬空	测量 Cs 电压

时间段内采样电容上的电压变化情况如下：

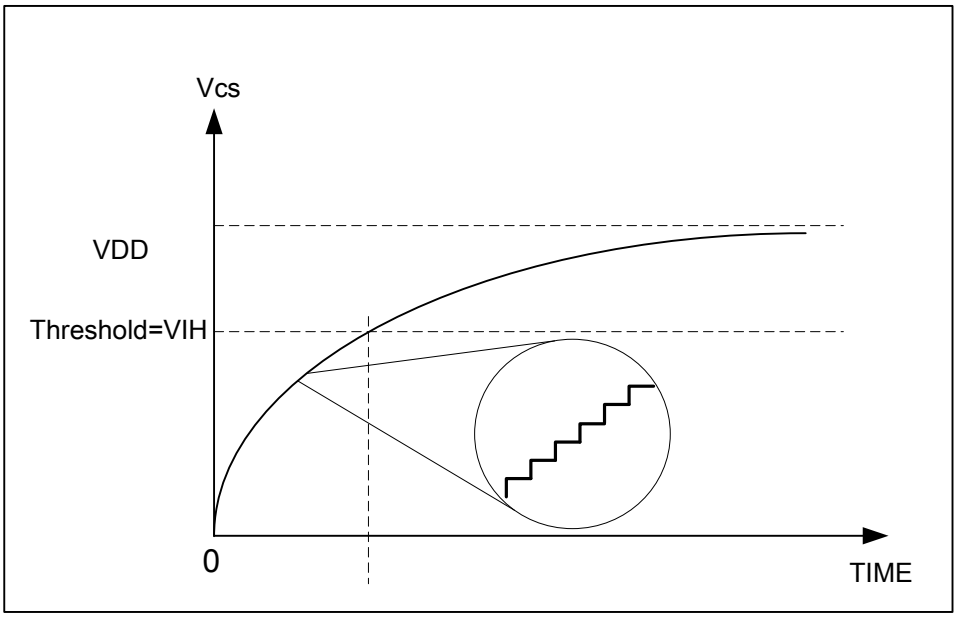


图 28.2 采样电容电压变化

下图显示了电荷迁移采集的顺序的例子。

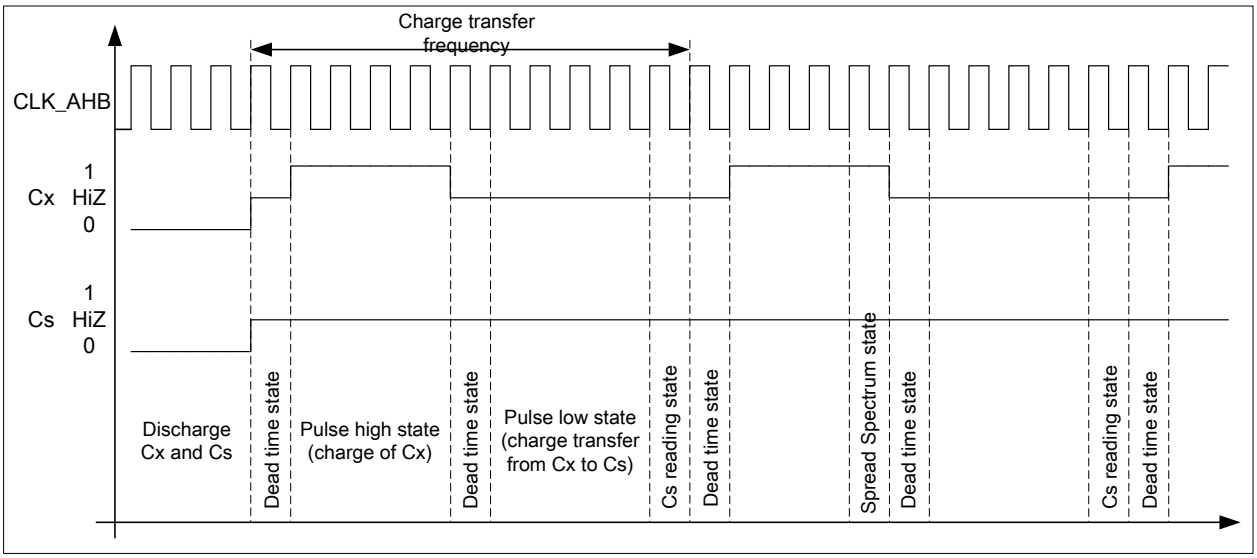


图 28.3 电荷转移采集时序图

为了更高的灵活性，电荷迁移的频率是可以设置的。脉冲的高电平（Cx 的充电）和脉冲的低电平（从 Cx 转移电荷到 Cs）的持续时间都可以通过软件或定时器产生的 PWM 来进行调整。为了确保对电极电容的测量准确度，脉冲高电平持续时间必须确保能够将 Cx 充满；脉冲低电平持续时间必须确保 Cx 中的电荷已充分进行转移。

死区时间是插在充电和电荷采集过程之间的一段时间。这段时间内充电开关和电荷迁移开关都处于断开状态，

用来保证稳定正确的电荷迁移采集顺序。

28.3.3. 跳频功能

跳频功能允许产生可变的充放电频率。这可以用来提高在噪声环境下电荷迁移采集检测的鲁棒性，同时可以限制感应信号扩散。跳频功能的实现可以通过两种方式：

1. 软件不断地去改变充电时间的长短和电荷迁移时间的长短。
2. 在硬件 PWM 模式下，不断地去改变定时器的 PWM 输出。

28.3.4. TSC 工作模式

软件 GPIO 模式

通过软件控制 GPIO 对触摸感应通道进行充放电并控制电荷迁移。根据电荷迁移采集的时序进行操作，操作流程如下：

1. 利用比较器来检测采集电容上的电压；配置比较器一端接入采集电容相对应的 I/O 口，另一端接入 DAC 的电压输出口。
2. 对 TSC 通道进行放电初始化；配置 TSC 通道相对应的 I/O 为输出端口，并输出数据 0。
3. 进入死区；配置 TSC 通道相对应的 I/O 口为模拟输入端口，相当于采样电容通道和触摸感应通道都浮空。
4. 对触摸感应通道进行充电；配置触摸感应通道相对应的 I/O 口为输出端口，并输出数据 1。
5. 再次进入死区。
6. 进行电荷迁移采集；开启采集电容通道和触摸感应通道，触摸感应通道上的电荷会迁移到采集电容上。
7. 不断地重复步骤 3~6，直到采集电容上的电压超过阈值。

软件 LDO 模式

这种模式不需要通过转换 GPIO 的模式来对通道进行充放电；而是通过配置 TSC 相关寄存器来变换 TSC 专用 LDO 的模式，从而控制通道的充放电过程。具体操作流程如下：

1. 利用比较器来检测采集电容上的电压；配置比较器一端接入采集电容相对应的 I/O 口，另一端接入 DAC 的电压输出口。
2. 对 TSC 通道进行放电初始化；开启采集电容通道和触摸感应通道，配置 Gx_LDO_MOD_SEL=11，选择为 GND 放电模式对 TSC 通道进行放电初始化。
3. 进入死区；关闭采集电容通道和触摸感应通道并配置 Gx_LDO_MOD_SEL=00，选择触摸通道浮空。
4. 对触摸感应通道进行充电；先开启触摸感应通道并配置 Gx_LDO_MOD_SEL=01，选择 VLDO 电压对触摸感应通道充电。
5. 再次进入死区。
6. 进行电荷迁移采集；开启采集电容通道和触摸感应通道，触摸感应通道上的电荷会迁移到采集电容上。
7. 不断地重复步骤 3~6，直到采集电容上的电压超过阈值。

硬件 PWM 模式

硬件 PWM 模式需要结合定时器（TIM1/TIM15/TIM16/TIM17）共同作用。需要定时器提供 2 个互补的 PWM 输出，并且此 PWM 还需要插入相应的死区时间。PWM 波形应该与电荷迁移采集时序存在一一对应的关系：

- PWM 中正向输出为低电平，反向输出为高电平时，对应电荷迁移采集时序中的 Cx 充电状态。
- PWM 中正向输出为低电平，反向输出也为低电平（死区）时，对应电荷迁移采集时序中的死区状态。

- PWM 中反向输出为高电平时，反向输出为低电平时，对应电荷迁移采集时序中的电荷转移状态。

硬件 PWM 模式下具体操作流程如下：

1. 利用比较器来检测采集电容上的电压；配置比较器一端接入采集电容相对应的 I/O 口，另一端接入 DAC 的电压输出口。
2. 对 TSC 通道进行放电初始化；开启采集电容通道和触摸感应通道，配置 Gx_LDO_MOD_SEL=11，选择为 GND 放电模式对 TSC 通道进行放电初始化。
3. 配置 TSC 为硬件 PWM 模式（Gx_PWM_EN=1）；如有需要，可以设置 HW_MOD_SEL，选择硬件 PWM 控制 GPIO 还是硬件 PWM 控制 LDO 进行充放电。设置为硬件 PWM 模式之前，必须开启采集电容通道和触摸感应通道。
4. 配置定时器产生 TSC 所需的 PWM；配置 TIM1/TIM15/TIM16/TIM17 产生所需的 PWM 波形。

28.3.5. 电容传感通道

下表所示为此器件中可以用作电容传感的 GPIO 一览。

表 28.2 通道对应的 GPIO

组别	电容感应信号名称	端口名称
1	TSC_G1_CAP	PA4
	TSC_G1_IO1	PA0
	TSC_G1_IO2	PA1
	TSC_G1_IO3	PA2
	TSC_G1_IO4	PA3
	TSC_G1_IO5	PA5
	TSC_G1_IO6	PA6
	TSC_G1_IO7	PA7
	TSC_G1_IO8	PB1
2	TSC_G2_CAP	PB12
	TSC_G2_IO1	PB0
	TSC_G2_IO2	PB2
	TSC_G2_IO3	PB10
	TSC_G2_IO4	PB11
	TSC_G2_IO5	PB13
	TSC_G2_IO6	PB14
	TSC_G2_IO7	PB15
	TSC_G2_IO8	PA8
3	TSC_G3_CAP	PA13
	TSC_G3_IO1	PA9
	TSC_G3_IO2	PA10
	TSC_G3_IO3	PA11
	TSC_G3_IO4	PA12
	TSC_G3_IO5	PA14
	TSC_G3_IO6	PA15

	TSC_G3_IO7	PB3
	TSC_G3_IO8	PB4

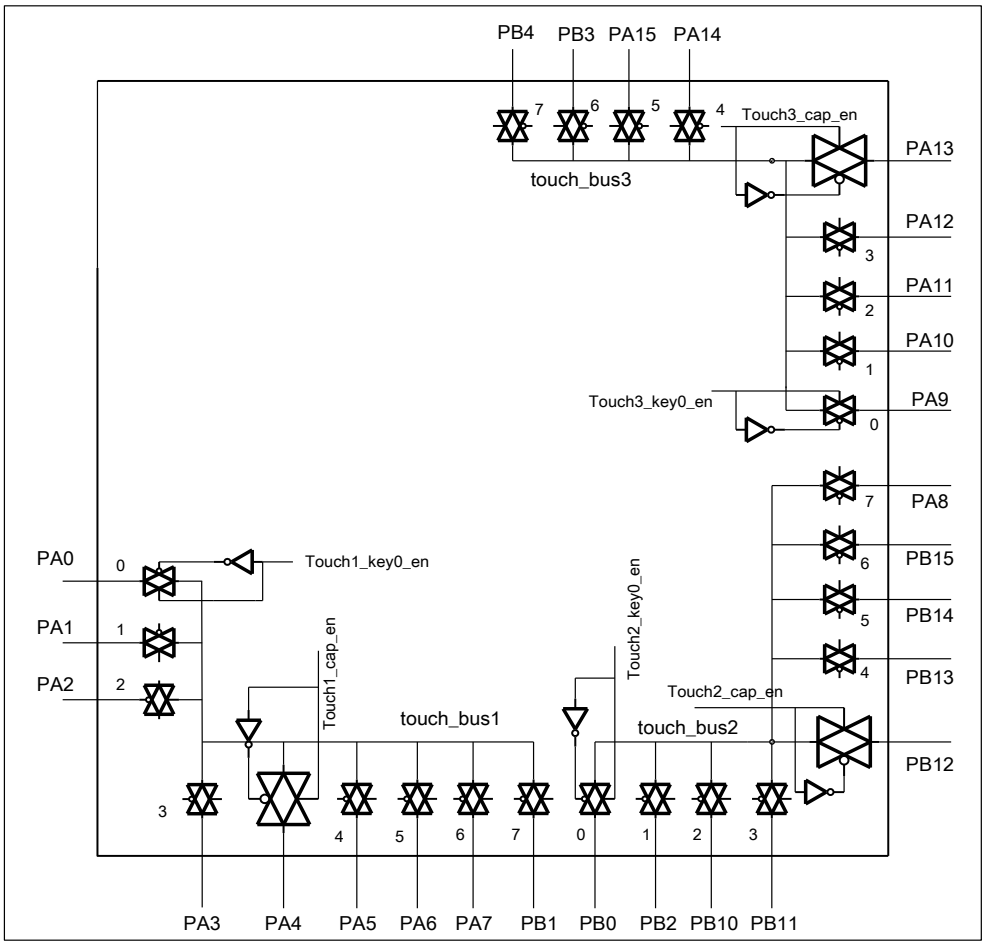


图 28.4 TSC 通道分布图

28.3.6. TSC 低功耗模式

表 28.3 低功耗模式对 TSC 的影响

模式	描述
Sleep	硬件 PWM 模式下，TSC 能正常工作。
Stop	TSC 寄存器处于冻结状态，定时器产生的 PWM 也处于冻结状态。
Standby	TSC 停止操作直到 MCU 退出 Stop 或 Standby 模式。

28.4. TSC 寄存器映射

下表给出了 TSC 寄存器的映射以及复位值。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TSC_CR	-	-	-	LDO_PWR_SEL		G3_CAP_EN	G2_CAP_EN	G1_CAP_EN	G3_KEY_EN[7:0]								G2_KEY_EN[7:0]								G1_KEY_EN[7:0]							
	Reset	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	TSC_CFGR	-	-	-	-	-	-	-	-	-	-	-	G3_LDO_MOD_SEL		G3_CG_MOD	G3_PWM_SEL	G3_PWM_EN	-	-	-	G2_LDO_MOD_SEL		G2_CG_MOD	G2_PWM_SEL	G2_PWM_EN	-	-	-	G1_LDO_MOD_SEL		G1_CG_MOD	G1_PWM_SEL	G1_PWM_EN
	Reset	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	X	X	X	0	0	0	0	0	X	X	X	0	0	0	0	0

28.4.1. TSC 控制寄存器（TSC_CR）

地址偏移：0x00

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—			LDO_PWR_SEL[1:0]		G3_CAP_EN	G2_CAP_EN	G1_CAP_EN
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
23:16	G3_KEY_EN[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	G2_KEY_EN[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW
7:0	G1_KEY_EN[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
31:29	NA	保留位，未定义
28:27	LDO_MODE[1:0]	VLDO 充电电压档位选择 三组通道共用一个电压档位选择。当 LDO 充放电模式选择为通过 VLDO 对触摸感应通道进行充放电（Gx_LDO_MOD_SEL 为 01）时，此位才有作用；否则，此位无作用。 00：2V 01：2V 10：3V 11：4V
26	G3_CAP_EN	第 3 组的采样电容通道使能 0：G3 采样电容通道关闭 1：G3 采样电容通道开启
25	G2_CAP_EN	第 2 组的采样电容通道使能 0：G2 采样电容通道关闭 1：G2 采样电容通道开启
24	G1_CAP_EN	第 1 组的采样电容通道使能 0：G1 采样电容通道关闭

		1：G1 采样电容通道开启
23:16	G3_KEY_EN[7:0]	第3组第x个触摸感应通道使能 0：第x个触摸感应通道关闭 1：第x个触摸感应通道开启
15:8	G2_KEY_EN[7:0]	第2组第x个触摸感应通道使能 0：第x个触摸感应通道关闭 1：第x个触摸感应通道开启
7:0	G1_KEY_EN[7:0]	第1组第x个触摸感应通道使能 0：第x个触摸感应通道关闭 1：第x个触摸感应通道开启

28.4.2. TSC 配置寄存器（TSC_CFGR）

地址偏移：0x04

复位值：0x0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							HW_MOD_SEL
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
23:16	—			G3_LDO_MOD_SEL		G3_CG_MOD	G3_PWM_SEL	G3_PWM_EN
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
15:8	—			G2_LDO_MOD_SEL		G2_CG_MOD	G2_PWM_SEL	G2_PWM_EN
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW
7:0	—			G1_LDO_MOD_SEL		G1_CG_MOD	G1_PWM_SEL	G1_PWM_EN
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:25	NA	保留位，未定义
24	HW_MOD_SEL	硬件 PWM 控制模式选择 此位是用于选择在硬件 PWM 模式下的控制模式的选择，可以选择通过 GPIO 对触摸感应通道进行充放电，也可以选择通过内部专用的触摸 LDO 对触摸感应通道进行充放电。当硬件 PWM 模式开启（Gx_PWM_EN=1）时，此位才有作用。 0：PWM 控制 GPIO 进行充放电。 1：PWM 控制 LDO 进行充放电。
23:21	NA	保留位，未定义
20:19	G3_LDO_MOD_SEL	G3 的 LDO 充放电模式的选择。参考 G1_LDO_MOD_SEL 的描述。
18	G3_CG_MOD	G3 的触摸通道充放电方向控制（触摸感应模式选择）。参考 G1_CG_MOD 的描述。
17	G3_PWM_SEL	G3 的 PWM 来源的选择 硬件 PWM 模式开启（G3_PWM_EN=1）时，此位才有作用。 0： TIM17 的通道 1 作为硬件 PWM 的来源。 1： TIM1 的通道 3 作为硬件 PWM 的来源。

16	G3_PWM_EN	G3 的硬件 PWM 模式使能。参考 G1_PWM_EN 的描述。
15:13	NA	保留位，未定义
12:11	G2_LDO_MOD_SEL	G2 的 LDO 充放电模式的选择。参考 G1_LDO_MOD_SEL 的描述。
10	G2_CG_MOD	G2 的触摸通道充放电方向控制（触摸感应模式选择）。参考 G1_CG_MOD 的描述。
9	G2_PWM_SEL	G2 的 PWM 来源的选择 硬件 PWM 模式开启（G2_PWM_EN=1）时，此位才有作用。 0： TIM16 的通道 1 作为硬件 PWM 的来源。 1： TIM1 的通道 2 作为硬件 PWM 的来源。
8	G2_PWM_EN	G2 的硬件 PWM 模式使能。参考 G1_PWM_EN 的描述。
7:5	NA	保留位，未定义
4:3	G1_LDO_MOD_SEL	G1 的 LDO 充放电模式的选择 00：浮空模式，LDO 不对通道进行充放电。 01：VLDO 充电模式，选择 LDO 中 VLDO 电压对通道进行充电。 10：VDD 充电模式，选择 LDO 中 VDD 电压对通道进行充电。 11：GND 放电模式，选择 LDO 中 GND 电压对通道进行放电。
2	G1_CG_MOD	G1 的触摸通道充放电方向控制（触摸感应模式选择） 0：采样电容通道向触摸感应通道转移电量。 1：触摸感应通道向采样电容通道转移电量。
1	G1_PWM_SEL	G1 的 PWM 来源的选择 硬件 PWM 模式开启（G1_PWM_EN=1）时，此位才有作用。 0： TIM15 的通道 1 作为硬件 PWM 的来源。 1： TIM1 的通道 1 作为硬件 PWM 的来源。
0	G1_PWM_EN	G1 的硬件 PWM 模式使能 此位为 0 时，需要用户操作相关的 TSC 寄存器或 GPIO 寄存器对 TSC 通道进行充放电；为 1 时，可以通过定时器产生的 PWM 波形来对 TSC 通道进行充放电。 0：硬件 PWM 模式禁止。 1：硬件 PWM 模式开启。

29. 通用串行总线（USB）

仅 FT32F072 系列芯片拥有此模块。

29.1. 主要特性

- 支持 USB1.1 协议，兼容 USB2.0 全速协议
- 支持 7 个通用端点和一个控制端点 0
- FIFO 存储器大小为 1KB
- 支持同步模式、块模式和中断模式传输
- 支持双缓冲区的块模式和、中断模式和同步模式传输
- 支持接口挂起和恢复操作
- 支持与主机连接和断开的能力(可控制的上拉电阻使能)

29.2. 功能描述

29.2.1. 结构框图

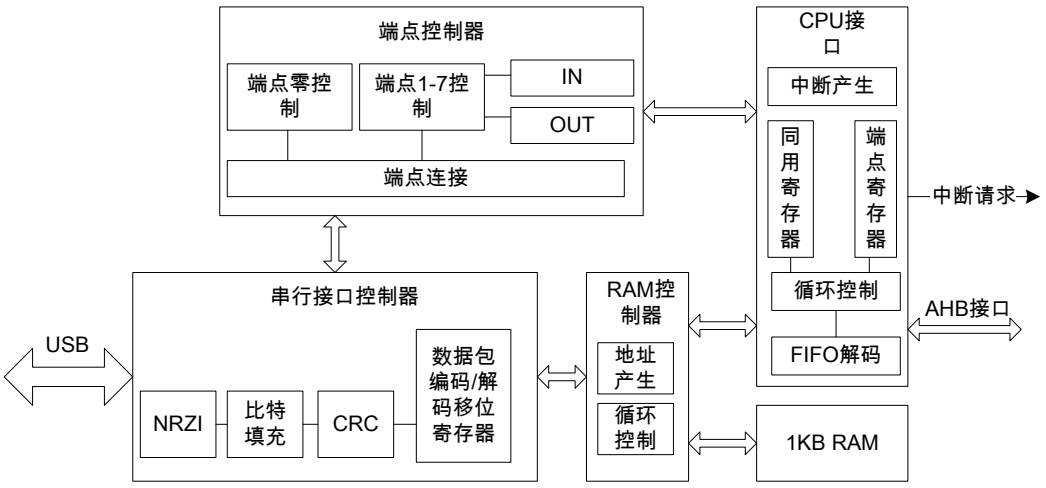


图 29.1 USB 实现结构框图

USB 接口实现了所有与 USB 通信相关的模块，包括：

串行接口控制器：这个模块功能包括同步模式识别，比特填充功能，CRC 校验码的生成与校验，PID 的接收与生成，开始帧信号的产生，数据的解码与编码等。

端点控制器：控制每个端点的传输状态，端点的传输方向；该模块总共具有 8 个端点，一个是控制端点 0，另外是端点 1-7，因为 USB 模块是半双工收发数据，因此 1-7 中的端点每个时刻只能用做 IN 或 OUT。
端点 0 只能用控制方式进行数据传输，端点 1-7 可以选择使用块模式、中断模式和同步模式进行传输。

RAM 控制器：根据每个端点的配置分配每个端点占用的存储器空间，包括存储区地址边界的处理和双缓冲区的处理；

CPU 接口：负责读写通用寄存器和端点寄存器；通过该接口读写 SRAM 中的数据（SRAM 中的数据是通过 FIFO 控制器进行控制的，不能直接读写 SRAM 中的某个地址的数据），产生中断等。

29.2.2. 数据存储区

每个端点都可以向主机收发数据。接收/发送的数据存储在对应的 FIFO 缓存区，每个端点发送的数据区域与接收的数据区域公用一个数据区域，因此每个端点发送数据与接收数据不能同时进行。当要向缓存区写入数据时，首先要确定当前的端点模式是否用作 OUT；反之当前的端点模式应该用作 IN。每个端点的缓存区分配在 1KB SRAM 中不同的位置，如图 29.2 所示。每个端点 INMAXP/OUTMAP 寄存器的配置不能超过各个端点的最大值。

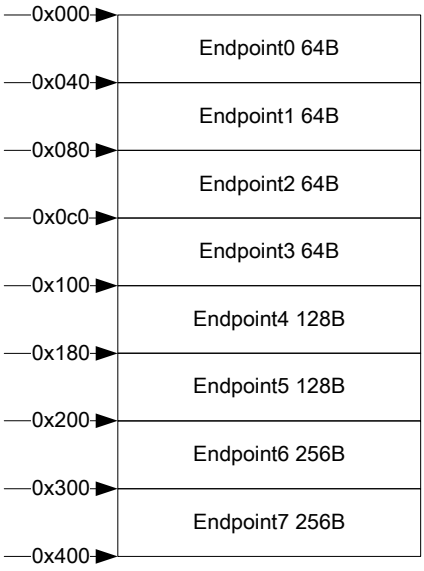


图 29.2 各个端点物理存储区地址和大小分配

除了端点 0 以外其他端点都可以使用双缓存区特性，在配置 IN/OUTMAXP 寄存器时，只要配置的数值不大于对应物理存储区的一半就可以了。例如端点 1 的物理存储区大小是 64B，配置的值不大于 32B 时，双缓存区的特性就会自动启用。

29.2.3. 传输概述

端点 0 只支持控制传输，端点 1-7 支持同步传输、块传输和中断模式传输。

基本的 USB 包如图 29.3 所示，如令牌包，数据包以及握手包。包是 USB 系统中信息传输的基本单元，所有数据都是经过打包后在总线上传输的。在 USB 上的数据信息的一次接收或发送的处理过程称为事务。事务处理的类型包括输入(IN)事务处理、输出(OUT)事务处理、设置(SETUP)事务处理等。

字段	PID	DATA	CRC16
字节数	1	0-1024	2

图 29.3 包的基本格式

控制传输

主机以控制传输的方式，通过端点 0 对设备发送各种请求，设备收到主机发来的请求后回复相应的信息，进行相应的操作。

端点 0 的 OUTPKTRDY 位置 1 时，表示端点 0 接收到了一个数据包，CPU 把这个数据包从 FIFO 中读取出来时，根据数据的内容来执行相应的操作；如果端点零需要发送数据，则需要切换端点零的 FIFO 方向为 IN，然后向 FIFO 中写入需要传送的数据，最后置位对应的 INPKTRDY 位；控制器端在接收到 IN 令牌时，数据就会自动的发向主机；数据通信过程中的接收发方在数据接收完成校验正确后会自动回复 ACK，否则会回复 NACK。

端点 0 工作在空闲状态时，在接收到 SETUP 包时，端点 0 的 OUTPKTRDY 位置 1，相应中断使能时就会进如中断，在中断中读取 FIFO 中的数据，然后根据相应的命令(设置描述符、获取描述符等)，切换 FIFO 的方向（INCSR*寄存器中的 MODE 位）；在 SETUP 包是设置地址时和设置接口等命令时，不会进行数据传输，如下图 29.4 和 29.5 所示。

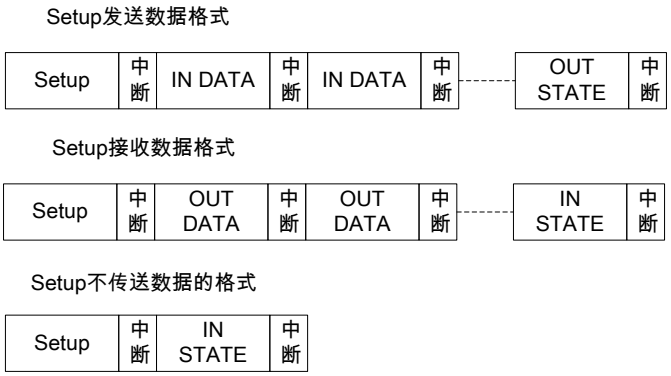


图 29.4 端点 0 数据传输格式

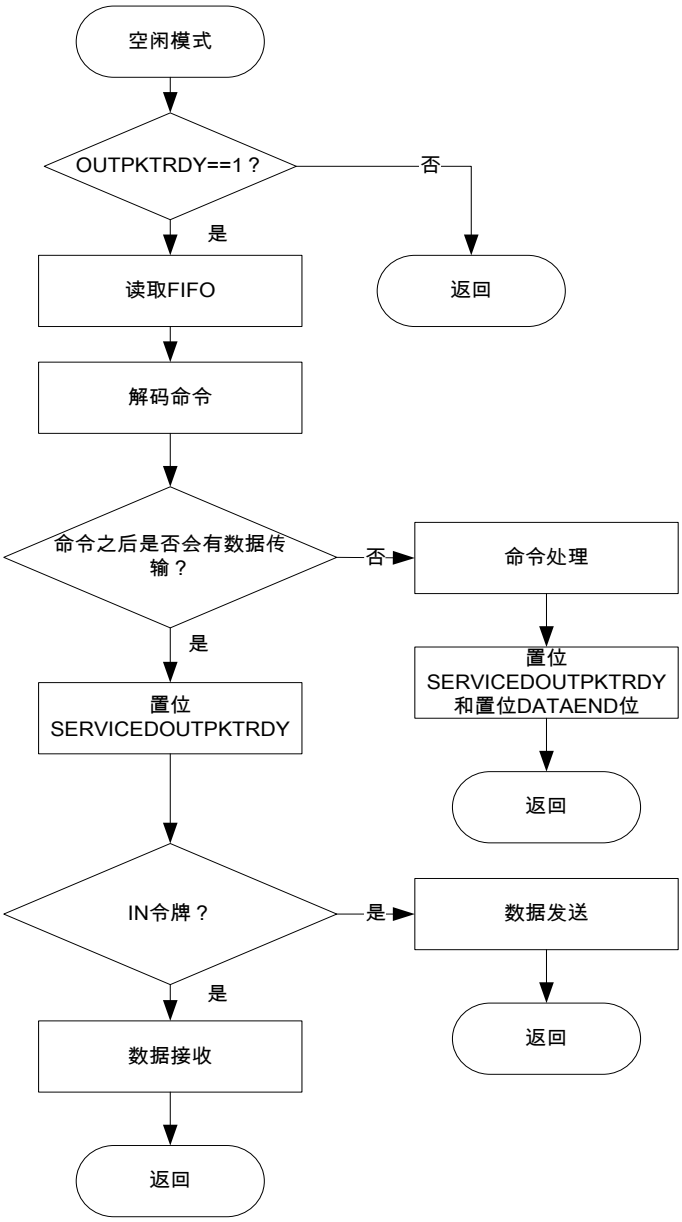


图 29.5 端点 0 的数据处理流程

块传输

块传输模式可以非周期性地跟主机交换交换数据 ;块传输模式中在 FIFO 接口配置为输出模式时 ,可以像 FIFO 中写入数据，接口控制器在接收到 IN 令牌时会自动的把 FIFO 中的数据传送给主机，主机校验接收完成后，回复 ACK ; 类似的接口 FIFO 配置为输入模式时，接口控制器在接收到 OUT 令牌时，会自动的把数据接收到 FIFO 中，接收校验正确后回复 ACK，否则回复 NACK，表示接收出现了错误。

块模式传输可以启用双缓冲区模式 ;在 INMAXP/OUTMAXP 寄存器设置的值小于等于接口实际的物理大小时，双缓存区的特性自动使能 , 这时候 FIFO 中可以存储两个数据包的数据 ; 对于接收来说在没有读取上一包的数据时 , 还可以接收新一包的数据 , 接收完成后两包的数据都不会丢失 ; 同样的对于发送来说，可以提前写入两包的数据存储在 FIFO 中。

通常对于发送数据时，在向 FIFO 中写入一包数据后，需要置位相应的 INPKTRDY 位，来告诉控制器数据写

入完毕 ;单在设置了 AUTOSET 位以后 ,只要写入 FIFO 中的数据等于了 INMAXP 设置的大小时 ,INPKTRDY 位就会自动置 1。类似的读取 FIFO 中的数据后，需要置位 SERVICEDOUTPKTRDY，来清除 OUTPKTRDY 标志，在使能了 AUTOCLEAR 以后，只要 FIFO 中的数据读取完毕后，OUTPKTRDY 标志位就自动清零了。

端点 1-7 块模式的使用需要配置一下寄存器：

- 配置 INMAXP/OUTMAXP 寄存器，设置数据包的大小，配置的值需要跟端点描述符中的最大包大小字段相同
- 使能寄存器 INTRIN1E 中相应端点的中断控制位
- 清零 ISO 位，使能块传输协议
- 置位 CLRDATATOG 位来清除端点的数据状态

相应端点的传输状态可以通过置位 SENDSTALL 位来关闭，在软件接收到 SENTSTALL 标志位时，需要清零 SENTSTALL 标志位；如果一个端点不想起用，应该一直保持 SENDSTALL 为 1。在清零了 SENDSTALL 位以后，相应的端点就可以重新使能，这时应该置位 CLRDATATOG 位，来恢复数据的翻转状态。

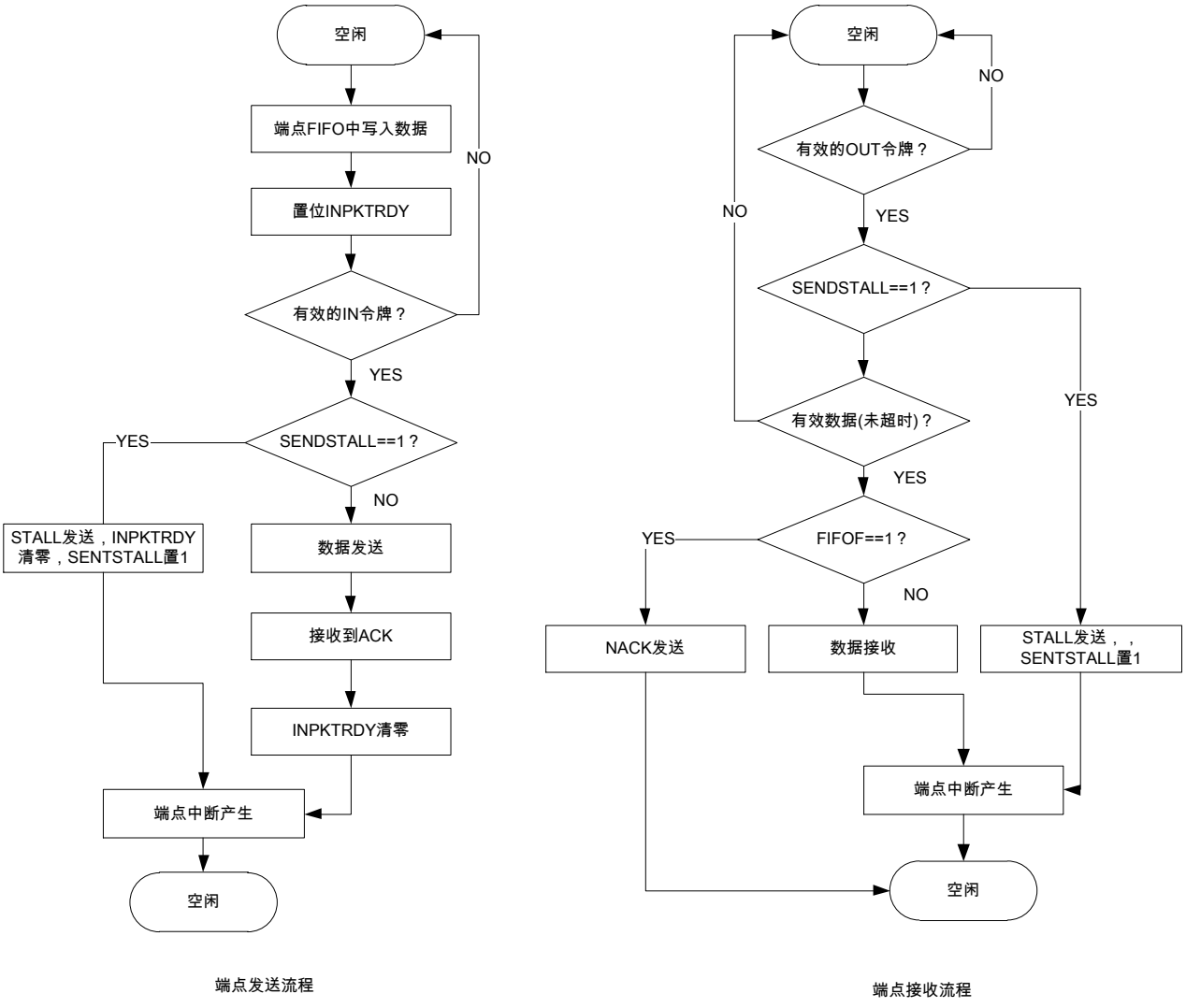


图 29.6 块模式端点数据发送和接收流程

中断模式

中断模式用来周期性的跟主机交换数据,其工作模式跟块模式基本相同;中断模式中通过设置 FRCDATATOG 位来强制发送数据状态进行翻转，每次传输完数据后都会认为传输成功，而不会考虑主机是否回复了 ACK。

同步模式

同步模式用于周期性的与主机交换大量数据;数据接收方不会回复 ACK,该方式适用于允许一定的数据误差。

同步模式中双缓存区的特性依然支持，只要配置 INMAXP/OUTMAXP 寄存器值不大于相应端点物理存储器值的一半，该特性就会自动启动。

在置位了 AUTOSET 位以后，相应端点的 FIFO 中写入的数据达到了 INMAXP 设置的包大小时，INPKTRDY 的值就会自动置 1。在 AUTOCLEAR 位置 1 以后，端点接收到数据后，只要 CPU 读取了 OUTMAXP 设置的包大小的数据时，OUTPKTRDY 的值就会自动清零。

同步模式的配置：

- 配置 INMAXP/OUTMAXP 寄存器，设置数据包的大小，配置的值需要跟端点描述符中的最大包大小字段相同
- 使能寄存器 INTRIN1E 中相应端点的中断控制位
- 置位 ISO 位，使能同步传输协议

在发送数据的过程中，假如 FIFO 中没有数据（CPU 没有足够的时间写入数据），端点会发送一个零字节的包给主机，同步会置位 UNDERRUN 标志位；在接收数据的过程中，假如 FIFO 中没有足够的空间(CPU 没有足够的时间去读取数据)存储接收的数据,OVERRUN 标志位会置 1,通信中出现 OVERRUN/UNDERRUN 标志位时，需要软件根据情况做出处理，例如可以提高 CPU 的工作时钟等；同步模式工作中即使接收到了错误（例如 CRC 校验错误），也会置位 OUTPKTRDY 标志位，同时 DATAERROR 标志位也会置 1，发现这种错误时也需要软件根据实际进行处理。

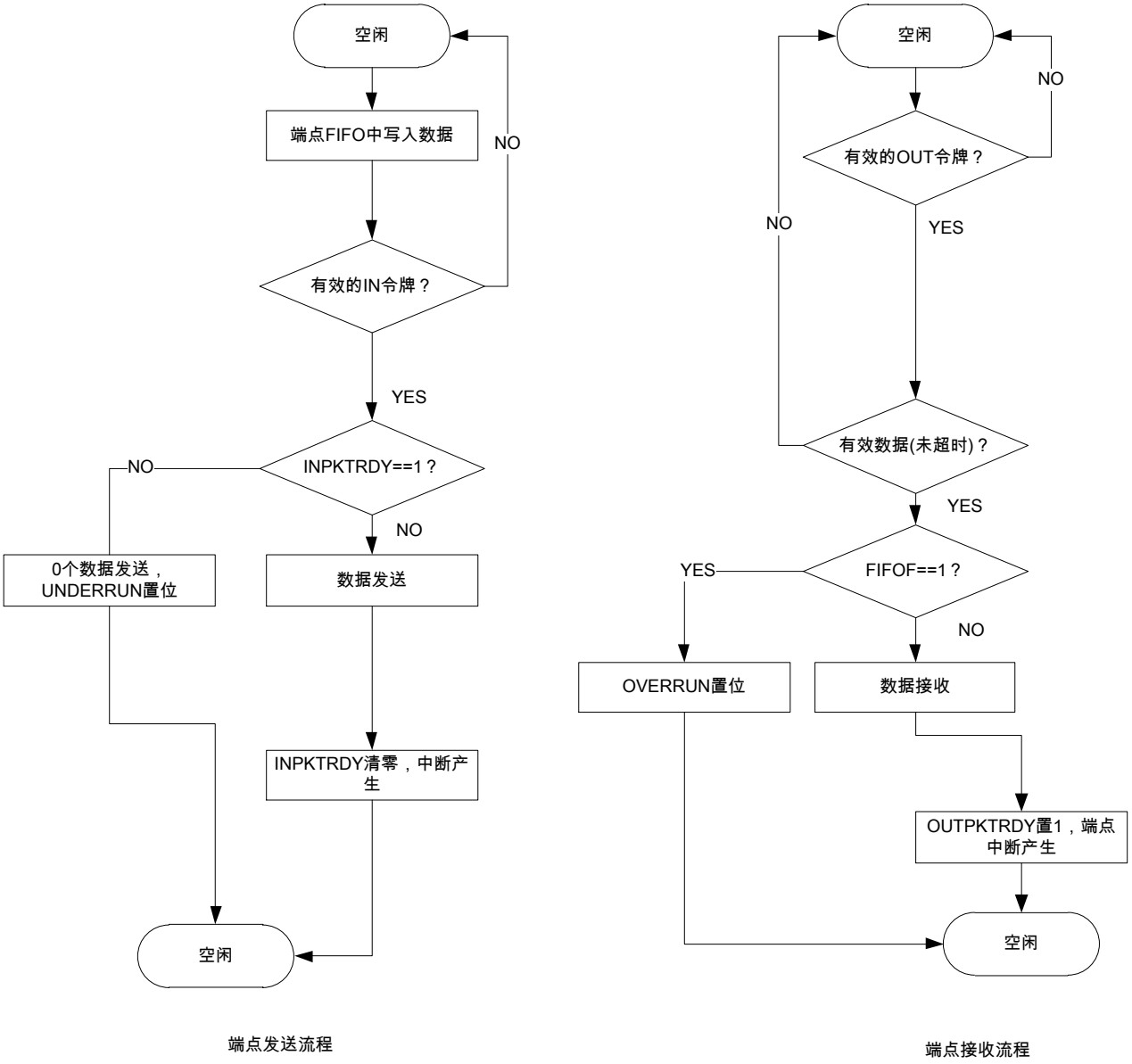


图 29.7 同步模式端点数据发送和接收流程

29.2.4. 挂起模式

USB 总线保持空闲时间超过 3ms ,同时 SUSPENDE 值为 1 时 ,接口控制器会进入挂起状态 ;在 SUSPENDIE 置 1 时，会进入中断。

挂起模式的退出可以通过两种方式实现。进入挂起模式时，当检测到总线上产生了恢复信号时，就可以退出挂起状态；另外一种方式是，通过软件置位 RESUME，来强制接口退出挂起状态。

29.3. 寄存器映射

地址偏移	名称	7	6	5	4	3	2	1	0	描述
0x00	USB_FADDR	FADDR[7:0]								通用寄存器
	Reset	0	0	0	0	0	0	0	0	
0x01	USB_POWER	ISOUPDATE				RESET	RESUME	SUSPENDM	SUSPENDE	
	Reset	0	x	x	x	0	0	0	0	
0x02	USB_INTRIN	EP7INF	EP6INF	EP5INF	EP4INF	EP3INF	EP2INF	EP1INF	EP0F	
	Reset	0	0	0	0	0	0	0	0	
0x04	USB_INTROUT	EP7OUTF	EP6OUTF	EP5OUTF	EP4OUTF	EP3OUTF	EP2OUTF	EP1OUTF		
	Reset	0	0	0	0	0	0	0	x	
0x06	USB_INTRUSB					SOF	RESET	RESUME	SUSPEND	
	Reset	x	x	x	x	0	0	0	0	
0x07	USB_INTRINE	EP7INE	EP6INE	EP5INE	EP4INE	EP3INE	EP2INE	EP1INE	EP0E	
	Reset	1	1	1	1	1	1	1	1	

地址偏移	名称	7	6	5	4	3	2	1	0	描述
0x09	USB_INTROUTE	EP7OUTIE	EP6OUTIE	EP5OUTIE	EP4OUTIE	EP3OUTIE	EP2OUTIE	EP1OUTIE	_	通用寄存器
	Reset	1	1	1	1	1	1	1	x	
0x0b	USB_INTRUSBE	_	_	_	_	SOFIE	RESETIE	RESUMEIE	SUSPENDIE	
	Reset	x	x	x	x	0	1	1	0	
0x0c	USB_FRAM1	FRAME[7:0]								
	Reset	0	0	0	0	0	0	0	0	
0x0d	USB_FRAM2	FRAME[10:8]								
	Reset									
0x0e	USB_INDEX	_	_	_	_	_	INDEX[2:0]			
	Reset	x	x	x	x	x	0	0	0	
0x0f	USB_PDCTRL	_	_	_	_	_	_	PDEN	PUEN	
	Reset	x	x	x	x	x	0	0	0	

地址偏移	名称	7	6	5	4	3	2	1	0	描述
0x10	USB_INMAXP	INMAXP[7:0]								端点寄存器(访问这些寄存器之前，请先配置INDEX寄存器的值)
	Reset	0	0	0	0	0	0	0	0	
0x11	USB_CSR0（端点0）	SSETUPEND	SOUTPKTRDY	SENDSTALL	SETUPEND	DATAEND	SENTSTALL	INPKTRDY	OUTPKTRDY	
	Reset	0	0	0	0	0	0	0	0	
0x11	USB_INCSR1（端点1-7）		CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	UNDERRUN	FIFONE	INPKTRDY	
	Reset	0	0	0	0	0	0	0	0	
0x12	USB_INCSR2	AUTOSET	ISO	MODE	-	FRCDATATOG	-	-	-	
	Reset	0	0	1	x	0	x	x	x	
0x13	USB_OUTMAXP	OUTMAXP[7:0]								
	Reset	0	0	0	0	0	0	0	0	
0x14	USB_OUTCSR1	CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOF	OUTPKTRDY	
	Reset	0	0	0	0	0	0	0	0	
0x15	USB_OUTCSR2	AUTOCLEAR	ISO	-	-	-	-	-	-	
	Reset	0	0	x	x	x	x	x	x	
0x16	USB_OUTCOUNTER	OUTCOUNTER[7:0]								
	Reset	0	0	0	0	0	0	0	0	

地址偏移	名称	7	6	5	4	3	2	1	0	描述
0x20	USB_FIFO0	FIFO0[7:0]								数据寄存器
	Reset	x	x	x	x	x	x	x	x	
0x24	USB_FIFO1	FIFO1[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x28	USB_FIFO2	FIFO2[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x2c	USB_FIFO3	FIFO3[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x30	USB_FIFO4	FIFO4[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x34	USB_FIFO5	FIFO5[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x38	USB_FIFO6	FIFO6[7:0]								
	Reset	x	x	x	x	x	x	x	x	
0x3c	USB_FIFO7	FIFO7[7:0]								
	Reset	x	x	x	x	x	x	x	x	

29.3.1. USB_FADDR

位地址	7	6	5	4	3	2	1	0
7:0	UPDATE	FADDR[6:0]						
类型	RO	RW						

Bit	Name	Function
7	UPDATE	当寄存器位 FADDR 写入的时候置位，当新地址生效时自动清零。
6:0	FADDR	功能地址，用来配置 USB 功能端的地址

29.3.2. USB_POWER

位地址	7	6	5	4	3	2	1	0
7:0	ISOUPDATE	—			RESET	RESUME	SUSPENDM	SUSPENDE
类型	RW	RO-0	RO-0	RO-0	RO	RW	RO	RW

Bit	Name	Function
7	ISOUPDATE	同步传输中等待 SOF 令牌后，在发送数据 1：置位 INPKTRDY 后，等待 SOF 令牌后发送数据；加入在 SOF 令牌之前接收到了 IN 令牌，则发送长度为零的数据包。 0：置位 INPKTRDY 后，不用等待 SOF 令牌，在接收到 IN 令牌后直接发送数据
6:4	NA	保留位，未定义
3	RESET	USB 总线上接收到了复位信号标识 1：USB 总线上接收到了复位信号 0：USB 总线上未接收到复位信号
2	RESUME	模块处于挂起状态时，恢复挂起使能 1：挂起时产生恢复信号，该位在置位超过 10ms（小于 15ms）后需要手动清零 0：不对模块进行恢复操作
1	SUSPENDM	指示模块进入挂起模式，该位在读取 USB_INTRUSB 或者置位 RESUME 后清零 1：模块进入了挂起模式 0：模块没有进入挂起模式
0	SUSPENDE	使能模块在接收到挂起信号时进入挂起模式 1：允许模块在接收到挂起信号时，进入挂起模式 0：不允许模块在接收到挂起信号时，进入挂起状态

29.3.3. USB_INTRIN

位地址	7	6	5	4	3	2	1	0
7:0	EP7INF	EP6INF	EP5INF	EP4INF	EP3INF	EP2INF	EP1INF	EP0F
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7	EP7INF	指示端点 7 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
6	EP6INF	指示端点 6 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
5	EP5INF	指示端点 5 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
4	EP4INF	指示端点 4 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
3	EP3INF	指示端点 3 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
2	EP2INF	指示端点 2 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
1	EP1INF	指示端点 1 IN 模式产生了中断 1：IN 模式产生了中断 0：IN 模式未产生中断
0	EP0F	指示端点 0 产生了中断 1：产生了中断 0：未产生中断

29.3.4. USB_INTROUT

位地址	7	6	5	4	3	2	1	0
7:0	EP7OUTF	EP6OUTF	EP5OUTF	EP4OUTF	EP3OUTF	EP2OUTF	EP1OUTF	—
类型	RO	RO	RO	RO	RO	RO	RO	RO-0

Bit	Name	Function
7	EP7OUTF	指示端点 7 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
6	EP6OUTF	指示端点 6 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
5	EP5OUTF	指示端点 5 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
4	EP4OUTF	指示端点 4 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
3	EP3OUTF	指示端点 3 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
2	EP2OUTF	指示端点 2 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
1	EP1OUTF	指示端点 1 OUT 模式产生了中断 1：OUT 模式产生了中断 0：OUT 模式未产生了中断
0	NA	保留位，未定义

29.3.5. USB_INTRUSB

位地址	7	6	5	4	3	2	1	0
7:0	—				SOF	RESET	RESUME	SUSPEND
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
7:4	NA	保留位，未定义
3	SOF	检测到每帧的起始，读取 USB_INTRUSB 后清零 1：检测到了一帧的起始 0：位检测到一帧的起始

2	RESET	USB 总线检测到了复位信号，读取 USB_INTRUSB 后清零 1：检测到了复位信号 0：未检测到复位信号
1	RESUME	当模块处于挂起模式时，USB 总线检测到了恢复信号，读取 USB_INTRUSB 后清零 1：挂起模式时检测到了恢复信号 0：未检测到恢复信号
0	SUSPEND	USB 总线检测到了 USB 挂起信号，读取 USB_INTRUSB 后清零 1：总线上检测到了挂起信号 0：未检测到挂起信号

29.3.6. USB_INTRINE

位地址	7	6	5	4	3	2	1	0
7:0	EP7INE	EP6INE	EP5INE	EP4INE	EP3INE	EP2INE	EP1INE	EP0IE
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	EP7INE	端点 7 IN 模式中断使能 1：使能端点 7 IN 模式中断 0：禁止端点 7 IN 模式中断
6	EP6INE	端点 6 IN 模式中断使能 1：使能端点 6 IN 模式中断 0：禁止端点 6 IN 模式中断
5	EP5INE	端点 5 IN 模式中断使能 1：使能端点 5 IN 模式中断 0：禁止端点 5 IN 模式中断
4	EP4INE	端点 4 IN 模式中断使能 1：使能端点 4 IN 模式中断 0：禁止端点 4 IN 模式中断
3	EP3INE	端点 3 IN 模式中断使能 1：使能端点 3 IN 模式中断 0：禁止端点 3 IN 模式中断
2	EP2INE	端点 2 IN 模式中断使能 1：使能端点 2 IN 模式中断 0：禁止端点 2 IN 模式中断
1	EP1INE	端点 1 IN 模式中断使能 1：使能端点 1 IN 模式中断 0：禁止端点 1 IN 模式中断
0	EP0E	端点 0 中断使能 1：使能端点 0 中断 0：禁止端点 0 中断

29.3.7. USB_INTROUTE

位地址	7	6	5	4	3	2	1	0
7:0	EP7OUTE	EP6OUTE	EP5OUTE	EP4OUTE	EP3OUTE	EP2OUTE	EP1OUTE	—
类型	RW	RW	RW	RW	RW	RW	RW	RO-0

Bit	Name	Function
7	EP7OUTE	端点 7 OUT 模式中断使能 1：使能端点 7 OUT 模式中断 0：禁止端点 7 OUT 模式中断
6	EP6OUTE	端点 6 OUT 模式中断使能 1：使能端点 6 OUT 模式中断 0：禁止端点 6 OUT 模式中断
5	EP5OUTE	端点 5 OUT 模式中断使能 1：使能端点 5 OUT 模式中断 0：禁止端点 5 OUT 模式中断
4	EP4OUTE	端点 4 OUT 模式中断使能 1：使能端点 4 OUT 模式中断 0：禁止端点 4 OUT 模式中断
3	EP3OUTE	端点 3 OUT 模式中断使能 1：使能端点 3 OUT 模式中断 0：禁止端点 3 OUT 模式中断
2	EP2OUTE	端点 2 OUT 模式中断使能 1：使能端点 2 OUT 模式中断 0：禁止端点 2 OUT 模式中断
1	EP1OUTE	端点 1 OUT 模式中断使能 1：使能端点 1 OUT 模式中断 0：禁止端点 1 OUT 模式中断
0	NA	保留位，未定义

29.3.8. USB_INTRUSBE

位地址	7	6	5	4	3	2	1	0
7:0	—				SOFIE	RESETIE	RESUMEIE	SUSPENDIE
类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
7:4	NA	保留位，未定义
3	SOFIE	帧开始中断使能 1：使能帧开始中断 0：禁止帧开始中断

2	RESETIE	复位信号中断使能 1：使能复位信号中断 0：禁止复位信号中断
1	RESUMEIE	恢复信号中断使能 1：使能复位信号中断 0：禁止复位信号中断
0	SUSPENDIE	挂起信号中断使能 1：使能挂起信号中断 0：禁止挂起信号中断

29.3.9. USB_FRAM1

位地址	7	6	5	4	3	2	1	0
7:0	FRAM1[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	FRAM1	上次数据接收到的帧号码低 8 位

29.3.10. USB_FRAM2

位地址	7	6	5	4	3	2	1	0
7:0	—					FRAM2[2:0]		
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO		

Bit	Name	Function
7:3	NA	保留位，未定义
2:0	FRAM2	上次数据接收到的帧号码高 3 位

29.3.11. USB_INDEX

位地址	7	6	5	4	3	2	1	0
7:0	—				INDEX[3:0]			
类型	RO-0	RO-0	RO-0	RO-0	RW			

Bit	Name	Function
7:4	NA	保留位，未定义
3:0	INDEX	配置选择端点号，例如值位零，选择端点 0

29.3.12. USB_PDCTRL

位地址	7	6	5	4	3	2	1	0
7:0	—						PDEN	PUEN
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW

Bit	Name	Function
7:2	NA	保留位，未定义
1	PDEN	USB DM 下拉电阻使能 1：使能 DM 下拉电阻 0：禁止 DP 线上拉电阻
0	PUEN	USB DP 线上拉电阻使能 1：使能 DP 线上拉电阻 0：禁止 DP 线上拉电阻

29.3.13. USB_CSR0

位地址	7	6	5	4	3	2	1	0
7:0	SSETUPEND	SOUTPKTRDY	SENDSTALL	SETUPEND	DATAEND	SENTSTALL	INPKTRDY	OUTPKTRDY
类型	W1	W1	W1	RO	W1	RC_W0	RS	RO

Bit	Name	Function
7	SSETUPEND	写 1 清零 SETUPEND 位 1：清零 SETUPEND 位 0：不对 SETUPEND 位进行清零操作
6	SOUTPKTRDY	写 1 清零 OUTPKTRDY 位 1：清零 OUTPKTRDY 位 0：不对 OUTPKTRDY 位清零
5	SENDSTALL	写 1 终止当前传输，STALL 握手信号自动发送，发送完成后自动清零 1：终止当前传输 0：不对当前传输进行终止操作
4	SETUPEND	DATAEND 未置 1 时，在接收到空值传输时，自动置 1，同时会产生中断并且清空对应 FIFO 1：表示接收到 SETUP 令牌接收完成 0：未接收到 SETUP 令牌
3	DATAEND	表示将要传输的是最后一包数据 1：在发送时表示，将要发送的时最后一包数据；接收时将要接收的是最后一包数据；可以置位 INPKTRDY 和 DATAEND 发送 0 字节长度的包 0：将要传输的不是最后一包数据

2	SENTSTALL	表示接口已经发送过 STALL 握手信号了 1：接口发送过 STALL 握手信号 0：清零 SENTSTALL 位
1	INPKTRDY	表示数据已经写入 FIFO 中，可以进行发送，发送完成后改位自动清零 1：数据已经写入 FIFO 中 0：待发送的数据未准备好
0	OUTPKTRDY	表示接收已经完成，写 SOUTPKTRDY 清零

29.3.14. USB_OUTCOUNTER

位地址	7	6	5	4	3	2	1	0
7:0	OUTCOUNTER1[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	OUTCOUNTER	相应端点上次接收的数据大小

29.3.15. USB_INMAXP

位地址	7	6	5	4	3	2	1	0
7:0	INMAXP[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	INMAXP	配置 IN 端点模式包的大小，单位是 8Byte，例如设置 1 表示包的大小是 8 字节，配置的值不能超过实际相应 FIFO 的物理值

29.3.16. USB_INCSR1

位地址	7	6	5	4	3	2	1	0
7:0	—	CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	OVERRUN	FIFONE	INPKTRDY
类型	RO-0	W1	RC_W0	RW	W1	RC_W0	RC_W0	RS

Bit	Name	Function
7	NA	保留位，未定义
6	CLRDATATOG	写 1 复位相应端点 IN 模式的数据翻转到零 1：清零相应端点的数据翻转状态 0：不对端点的数据状态进行清零

5	SENTSTALL	指示 STALL 握手信号已经发送 1：表示 STALL 握手信号已经发送 0：清零该位
4	SENDSTALL	当收到 IN 令牌时，发送 STALL 握手信号，同步模式中该位没有影响 1：接收到 IN 令牌时，发送 STALL 握手信号 0：不发送 STALL 握手信号
3	FLUSHFIFO	清空 IN 模式下的 FIFO 1：清空 FIFO 中要发送的数据 0：不对 IN 模式 FIFO 中的数据进行操作
2	UNDERRUN	同步模式中表示一个数据长度为零的包已经发送，块模式中表示 NAK 信号已经发送 1：同步模式中 INPKTRDY 未置 1 时，发送了一帧长度为零的包；块模式中表示 NAK 信号已经发送 0：清零该标志位
1	FIFONE	IN 模式中，FIFO 中数据为非空 1：FIFO 中只少有一包数据 0：FIFO 中数据为空
0	INPKTRDY	要发送的数据已经写入到 FIFO 中 1：表示要发送的数据已经写入到 FIFO 中，该位在发送完成后自动清零 0：表示数据未准备好

29.3.17. USB_INCSR2

位地址	7	6	5	4	3	2	1	0
7:0	AUTOSET	ISO	MODE	—	FRCDATATOG	—		
类型	RW	RW	RW	RO-0	RW	RO-0	RO-0	RO-0

Bit	Name	Function
7	AUTOSET	写入相应的 INMAXP 配置的数据后 INPKTRDY 自动置 1 1：使能自动置位 INPKTRDY 功能 0：禁用 INPKTRDY 功能
6	ISO	同步传输使能 1：使能同步传输模式 0：使能块模式/中断模式
5	MODE	设置端点的工作模式 1：设置端点工作模式为 IN 模式 0：设置端点工作模式为 OUT 模式
4	NA	保留位，未定义
3	FRCDATATOG	强制数据翻转 1：发送数据时，无论 ACK 信号是否收到，强制反转数据状态 0：数据状态反转按照正常模式
2:0	NA	保留位，未定义

29.3.18. USB_OUTMAXP

位地址	7	6	5	4	3	2	1	0
7:0	OUTMAXP[7:0]							
类型	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	OUTMAXP	相应端点 OUT 模式数据包的大小设置 ,单位是 8 字节 ,例如配置为 8 字节时，包的大小时 64 字节，配置的值不能超过实际相应 FIFO 的物理值

29.3.19. USB_OUTCSR1

位地址	7	6	5	4	3	2	1	0
7:0	CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOF	OUTPKTRDY
类型	W1	RC_W0	RW	W1	RO	RC_W0	RO	RC_W0

Bit	Name	Function
7	CLRDATATOG	复位端点的数据翻转状态 1：复位端点的数据反转状态 0：不对端点的复位状态进行翻转
6	SENTSTALL	STALL 握手信号已经发送 1：STALL 握手信号已经发生完成 0：清零该位
5	SENDSTALL	发送 STALL 握手信号，该位在同步模式下无效 1：发送 STALL 握手信号 0：不发送 STALL 握手信号
4	FLUSHFIFO	清空接收 FIFO 中的数据 1：清空接收 FIFO 中的数据 0：不对 FIFO 中的数据进行清空
3	DATAERROR	当 OUTPKTRDY 置 1 时，表示接收出现了 CRC 错误或者比特填充错误，这一位仅在同步模式有效 1：数据传输发生了 CRC 错误或者比特填充错误 0：数据传输未发生错误
2	OVERRUN	FIFO 存储区溢出，该位在同步模式下有效 1：表示接收发生了溢出 0：清零该位
1	FIFOF	FIFO 中数据已满 1：FIFO 中没有多余的空间存储数据 0：FIFO 中还有空间可以存储数据

0	OUTPKTRDY	端点接收到了数据 1：端点 FIFO 中有数据，需要读取 0：端点 FIFO 中没有数据

29.3.20. USB_OUTCSR2

位地址	7	6	5	4	3	2	1	0
7:0	AUTOCLEAR	ISO	—					
类型	RW	RW	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

Bit	Name	Function
7	AUTOCLEAR	OUTPKTRDY 自动清零 1：当从 FIFO 中读取字节数等于 OUTMAXP 数据时，OUTPKTRDY 自动清零 0：不启用 OUTPKTRDY 自动清零特性
6	ISO	端点 OUT 模式，同步传输使能 1：启用同步传输模式 0：启用块模式/中断模式
5:0	NA	保留位，未定义

29.3.21. USB_FIFO0

位地址	7	6	5	4	3	2	1	0
7:0	FIFO0[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO0	端点 0 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

29.3.22. USB_FIFO1

位地址	7	6	5	4	3	2	1	0
7:0	FIFO1[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO1	端点 1 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

29.3.23. USB_FIFO2

位地址	7	6	5	4	3	2	1	0
7:0	FIFO2[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO2	端点 2 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器的接口用来与内部的 FIFO 进行数据交互

29.3.24. USB_FIFO3

位地址	7	6	5	4	3	2	1	0
7:0	FIFO3[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO3	端点 3 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

29.3.25. USB_FIFO4

位地址	7	6	5	4	3	2	1	0
7:0	FIFO4[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO4	端点 4 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器的接口用来与内部的 FIFO 进行数据交互

29.3.26. USB_FIFO5

位地址	7	6	5	4	3	2	1	0
7:0	FIFO5[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO5	端点 5 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

29.3.27. USB_FIFO6

位地址	7	6	5	4	3	2	1	0
7:0	FIFO6[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO6	端点 6 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

29.3.28. USB_FIFO7

位地址	7	6	5	4	3	2	1	0
7:0	FIFO7[7:0]							
类型	RW							

Bit	Name	Function
7:0	FIFO7	端点 7 的数据寄存器 ,端点工作于 IN 模式时 ,用作写入数据 ;端点工作于 OUT 模式时，用于读取数据。该数据寄存器接口用来与内部的 FIFO 进行数据交互

30. 调试接口（Debug）

30.1. 概述

FT32F0XX 内核内含硬件调试模块，支持复杂的调试操作。硬件调试模块允许内核在取指（指令断点）或访问数据（数据断点）时停止。内核停止时，内核的内部状态和系统的外部状态都是可以查询的。完成查询后，内核和外设可以被复原，程序将继续执行。

当 FT32F0XX 微控制器连接到调试器并开始调试时，调试器将使用内核的硬件调试模块进行调试操作。

支持：

- 串行调试接口

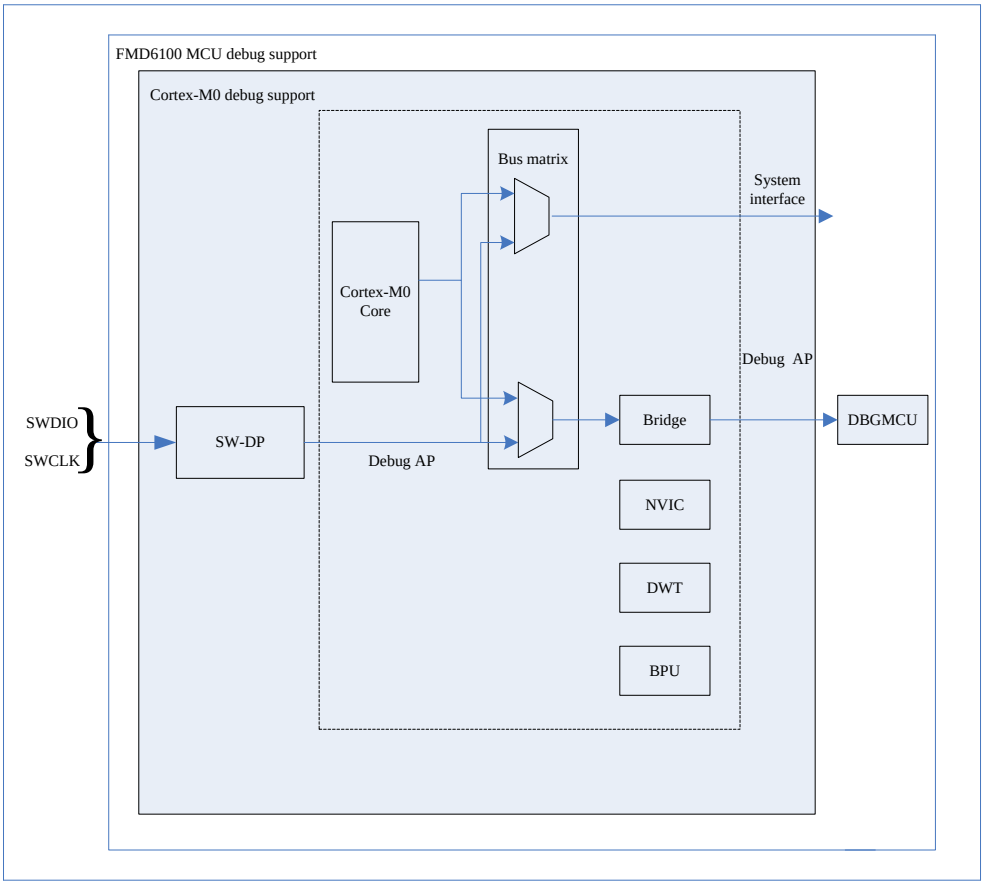


图30.1 串行调试接口结构

CPU 内核提供集成的片上调试功能。它由以下部分组成：

- SW-DP：串行调试端口
- BPU：断点调试单元
- DWT：数据触发
- 灵活的调试管脚输出分配
- MCU调试箱（支持低功耗模式），由串行时钟控制。

30.1.1. 相关 ARM 文档

- ARM® Cortex®-M0 Technical Reference Manual (TRM) It is available from: http://infocenter.arm.com
- ARM Debug Interface V5
- ARM CoreSight Design Kit revision r1p1 Technical Reference Manual

30.2. 功能描述

30.2.1. 管脚分布和调试端口脚

FT32F0XX 微控制器的不同封装有不同的有效引脚数。因此，某些与引脚相关的功能可能随封装而不同。

30.2.2. SWD 调试端口脚

FT32F0XX 的 2 个普通 I/O 口可用作 SW-DP 接口引脚。这些引脚在所有的封装里都存在。

表 30.1 管脚分配

SWJ-DP 端口引脚名称	SW 调试接口		引脚分配
	类型	调试功能	
SWDIO	输入/输出	串行数据输入/输出	PA13
SWCLK	输入	串行时钟	PA14

30.2.3. 脚上的内部上拉和下拉

保证 SWD 的输入引脚不是悬空的是非常必要的，因为他们直接连接到 D 触发器控制着调试模式。必须特别注意 SWCLK 引脚，因为他们直接连接到一些 D 触发器的时钟端。
为了避免任何未受控制的I/O电平，FT32F0XX在SWD输入脚上嵌入了内部上拉和下拉。

- SWDIO：内部上拉
- SWCLK：内部下拉

一旦SWD I/O被用户代码释放，GPIO控制器再次取得控制。这些I/O口的状态将恢复到复位时的状态。

- SWDIO：带上拉的输入
- SWCLK：带下拉的输入

软件可以把这些 I/O 口作为普通的 I/O 口使用。

30.2.4. ID 代码和锁定机制

在 FT32F0XX 内部有多个 ID 编码。

30.2.5. 微控制器设备 ID 编码

微控制器 FT32F0XX 内含一个 MCU ID 编码。这个 ID 定义了 MCU 的部件号和硅片版本。SW 调试口（2 个引脚）或通过用户代码都可以访问此编码。
DBGMCU_IDCODE 只支持 32 位访问并且为只读位。

30.2.6. SWD 协议介绍

此同步串行协议使用 2 个引脚：
SWCLK：从主机到目标的时钟信号
SWDIO：双向数据信号
协议允许读写 2 个寄存器组（DPACC 和 APACC 寄存器组）。
数据位按 LSB 传输。
由于 SWDIO 为双向口，该引脚需有上拉（建议使用 100K 电阻）。
按协议每次 SWDIO 方向改变时，需插入一个转换时间。在该期间内主机和目标都不驱动此信号线。转换时间的默认值是 1 个比特时间，但可以通过配置 SWCLK 频率来调节。

30.2.7. SWD 协议序列

每个序列由 3 个阶段组成：

1. 主机发送包请求（8位）
2. 目标发送确认响应（3位）
3. 主机或目标发送数据（33位）

比特位	名称	描述
0	起始	必须为 1
1	ApnDP	0:访问 DP 1:访问 AP
2	RnW	0:写请求 1:读请求
4:3	A(3:2)	DP 或 AP 寄存器的地址
5	Parity	前面比特位的校验位
6	Stop	0
7	Park	不能由主机驱动，由于有上拉，目标永远读为 1

有关 DPACC 和 APACC 寄存器描述的详细资料，请参考 CPU 技术参考手册。
包请求后总是跟一个（缺省为 1 位）转换时间，此时主机和目标都不驱动线路。

比特位	名称	描述
0..2	ACK	001:失败 010:等待 100:成功

当 ACK 为失败或等待，或者是一个回复读操作的 ACK，此 ACK 后有一个转换时间。

比特位	名称	描述
0..31	WDATA/RDATA	写或读的数据

32	Parity	32 位数据的奇偶校验位

读操作的数据传输操作后有一个转换时间。

30.2.8. SW-DP 状态机（Reset，idle states，ID code）

SW-DP 状态机有一个内部 ID 编码用来识别 SW-DP，它遵守 JEP-106 标准。
注：在调试器读这个 ID 编码之前，SW-DP 的状态机是不工作的。
SW-DP 状态机将处于 RESET 状态，在上电复位后，或 DP 从 JTAG 切换到 SWD 后，或有超过 50 个周期的高电平。
当状态机处于 RESET 状态时，如果有至少 2 个周期的低电平，状态机将切换到 IDLE 状态。
当状态机处于 RESET 状态后，必须首先进入 IDLE 状态，并执行一个读 DP-SW ID 寄存器的操作。否则，调试器在执行其他传输时，只能获得一个失败的 ACK 响应。

30.2.9. DP 和 AP 读/ 写访问

对 DP 的读操作没有传递性：调试器将直接获得数据（如果 ACK = 成功），或者等待（如果 ACK = 等待）。
对 AP 的读操作具有传递性。这意味着前一次读操作的结果只能在下一次操作时获得。如果下一次的操作不是对 AP 的访问，则必须读 DP-RDBUFF 寄存器来获得上一次读操作的结果。
DP-CTRL/STAT 寄存器的 READOK 标志位会在每次 AP 读操作和 RDBUFF 读操作后更新 ,以通知调试器 AP 的读操作是否成功。
SW-DP 具有写缓冲区（DP 和 AP 都有写缓冲），这使得其他传输在进行时，仍然可以接受写操作。如果写缓冲区满，调试器将获得一个等待的 ACK 响应。读 IDCODE 寄存器，读 CTRL/STAT 寄存器和写 ABORT 寄存器操作在写缓冲区满时仍被接受。
由于 SWCLK 和 HCLK 的异步性，需要在写操作后（在奇偶校验位后）插入 2 个额外的 SWCLK 周期，以确保内部写操作正确完成。这两个额外的时钟周期需要在线路为低时插入（IDLE 状态下）。这个操作步骤在写 CTRL/STAT 寄存器以提出一个上电请求时尤其重要 ,下一个操作(在内核上电后才有效的操作)会立即执行，这将导致失败。

30.2.10. SW-DP 寄存器

当 ApnDP = 0 时，可以访问以下这些寄存器。

A[3:2]	读/写	SELECT寄存器的CTRLSEL位	寄存器	描述
00	读		IDCODE	固定为0x1BA01477(用于识别SW-DP)。
00	写		ABORT	
01	读/写	0	DP-CTRL/ STAT	请求一个系统或调试的上电操作； 配置AP访问的操作模式； 控制比较，校验操作； 读取一些状态位(溢出 ,上电响应)。
01	读/写	1	WIRE CONTROL	配置串行通信物理层协议(如转换时间长度等)。
10	读		READ	允许从一个错误的调试传输中恢复数据

			RESEND	而不用重复最初 的AP传输。
10	写		SELECT	选择当前的访问端口和有效的4字长寄存器窗口。
11	读/写		READ BUFFER	由于AP的访问具有传递性（当前AP读操作的结果会在 下次AP传输时传出），因此这个寄存器非常必要。这 个寄存器会从AP捕获上一次读操作的数据结果，因此 可以获得数据而不必再启动一个新的AP传输。

30.2.11. SW-AP 寄存器

当 ApnDP = 1 时，可以访问以下这些寄存器。

AP 寄存器的访问地址由以下两部分组成：

1. A [3:2]的值
2. DP SELECT 寄存器的当前值

地址	A[3:2]	描述
0x00	00	保留位
0x04	01	DP CTRL/STAT 寄存器被使用需要： – 要求系统或者调试模块上电 – 配置AP访问的传输操作 – 控制压入比较压入验证操作 – 读一些状态标志位(溢出，上电响应)
0x08	10	DP SELECT 寄存器: 使用选择当前访问端口和实际的4字寄存器窗口。 – [31:24]: APSEL: 选择当前AP – [23:8]: 保留位 – [7:4]: APBANKSEL: 选择当前AP的4字寄存器窗口 – [3:0]: 保留位
0x0C	11	DP RDBUFF 寄存器: 在一系列操作之后允许调试接口得到最后的结果（没有请求新的JTAG-DP操作）

30.2.12. Core Debug 寄存器

芯片core中包含了许多供debug使用的寄存器。DFSR寄存器（0xE000ED30）Debug Fault Status Register（DFSR）包含了每一种debug事件的状态位，可以使调试者获知core 被暂停的原因。当debug事件使core暂停时，寄存器中相应位被置位，且寄存器中每一位都为写1清零。

地址	寄存器名字	描述

0xE000EDF0	DHCSR 寄存器	Debug Halting Control and Status Register（DHCSR）为暂停core的控制寄存器，调试者可以通过该寄存器进行暂停core，单步调试等操作。
0xE000EDF4	DCRSR 寄存器	Debug Core Register Selector Register（DCRSR）为Debug core寄存器选择寄存器，当core被暂停时，调试者可以通过该寄存器进行core寄存器的读写。
0xE000EDF8	DCRDR 寄存器	Debug Core Register Data Register（DCRDR）与DCRSR协同使用，可以用来访问core中的通用寄存器和特殊功能寄存器。DCRSR寄存器对core寄存器进行选择与读写控制 ,DCRDR寄存器为要传输的数据。
0xE000EDFC	DEMCR 寄存器	Debug Exception and Monitor Control Register（DEMCR）用来在debug状态下进行中断的控制，也用来使能DWT模块。

30.2.13. MCU 调试模块（MCUDBG）

MCU 调试模块协助调试器提供以下功能：

- 低功耗模式
- 在断点时提供定时器，看门狗的时钟控制
- 对跟踪脚分配的控制

30.2.13.1.低功耗模式的调试支持

使用 WFI 和 WFE 可以进入低功耗模式。MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降 低 CPU 的能耗。内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。
为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

在睡眠模式下，调试器必须先置位 DBGMCU_CR 寄存器的 DBG_SLEEP 位。这将为 HCLK 提供与 FCLK（由代码配置的系统时钟）相同的时钟。

停机模式下，调试器必须先置位 DBG_STOP 位。这将激活内部振荡器，在停机模式下为 FCLK 和 HCLK 提供时钟。

30.2.13.2.支持定时器和看门狗的调试

在产生断点时，有必要根据定时器和看门狗的不同用途选择计数器的工作模式：

- 在产生断点时，计数器继续计数。这在输出 PWM 控制电机时常常要用到。
- 在产生断点时，计数器停止计数。这对于看门狗的计数器是必需的。

30.2.13.3. 调试 MCU 配置寄存器

此寄存器允许在调试状态下配置 MCU。包括：

- 支持低功耗模式
- 支持定时器和看门狗的计数器
- 分配跟踪引脚

DBGMCU_CR 寄存器被映射到外部 APB 总线，基地址为 0x40013404。寄存器由 PORESET 异步复位(不被系统复位所复位)。当内核处于复位状态下时，调试器可写该寄存器。 如果调试器不支持这些特性，用户软件仍可写这些寄存器。

30.3. Core Debug 寄存器映射

Core Debug 寄存器的基地址为 0xe000ed00。

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30	DFSR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EXTERNAL	VCATCH	DWTTRAP	BKPT	HALTED
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0
0xF0	DHCSR	DBGKEY[15:0]																															
		-	-	-	-	-	-	S_RESET_ST	ST_RETIRE_ST	-	-	-	-	S_LOCKUP	S_SLEEP	S_HALT	S_READY	-	-	-	-	-	-	-	-	-	-	-	-	C_MASKINTS	C_STEP	C_HALT	C_DEBUGEN
	Reset	x	x	x	x	x	x	1	0	x	x	x	x	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0
0xF4	DCRSR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	REGWnR	-	-	-	-	-	-	-	-	-	-	-	REGSEL [4:0]				
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0
0xF8	DCRDR	DBGTMP[31:0]																															
	Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xFC	DEMCR	-	-	-	-	-	-	-	DWTENA	-	-	-	-	-	-	-	-	-	-	-	-	-	VC_HARDERR	-	-	-	-	-	-	-	-	-	VC_CORERESET
	Reset	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	x	0

30.3.1. DFSR

偏移地址：0x30
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—			EXTERNAL	VCATCH	DWTTRAP	BKPT	HALTED
类型	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
31:5	NA	保留位
4	EXTERNAL	EDBGRQ被置位（芯片中一直为0，没有外部的debug请求）
3	VCATCH	发生Vector catch
2	DWTTRAP	Data watchpoint匹配
1	BKPT	Breakpoint匹配
0	HALTED	暂停调试或单步调试

30.3.2. DHCSR

偏移地址：0xF0
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	DBGKEY[15:8]							
	—						S_RESET_ST	S_RETIRE_ST
类型	RW	RW	RW	RW	RW	RW	RW	RW
23:16	DBGKEY[7:0]							
	—				S_LOCKUP	S_SLEEP	S_HALT	S_READY
类型	RW	RW	RW	RW	RW	RW	RW	RW
15:8	保留位							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—				C_MASKINTS	C_STEP	C_HALT	C_DEBUGEN

类型	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW

Bit	Name	Function
31:26,23:20,15:4	NA	保留位
31:16	DBGKEY	Debug Key：写操作时高16位必须为0xA05F，否则写操作被忽略
25	S_RESET_ST	复位状态标志位：core已经被复位或正在复位，读清零。
24	S_RETIRE_ST	指令完成标志位，读清零
19	S_LOCKUP	Core lockup状态标志位，通过暂停core清零
18	S_SLEEP	Core sleep标志位
17	S_HALT	Core halt标志位
16	S_READY	handshake标志位，core完成一次寄存器的读写操作后置1
3	C_MASKINTS	中断屏蔽位，当debug使能时有效
2	C_STEP	单步调试控制位，当debug使能时有效
1	C_HALT	暂停core控制位，当debug使能时有效
0	C_DEBUGEN	debug使能位，为1时使能debug。

注意：写操作时，DHCSR[31 :16]必须为 0xA05F；
读操作时，DHCSR[31 :26]为保留位，DHCSR[25 :16]为对应状态标志位。

30.3.3. DCRSR

偏移地址：0xF4
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							REGWnR
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	W
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—			REGSEL[4:0]				
类型	RO-0	RO-0	RO-0	WO	WO	WO	WO	WO

Bit	Name	Function
31:17,15:5	NA	保留位
16	REGWnR	core 寄存器读写位。1：写；　　0：读
4:0	REGSEL	core 寄存器选择 REGSEL: 5’b00000：R0； 5’b00001：R1；

		… 5’b01100：R12； 5’b01101：current SP； 5’b01110：LR； 5’b01111：DebugReternAddrese（退出 debug 状态时将要执行的指令的地址）; 5’b10000：xPSR； 5’b10001：MSP； 5’b10010：PSP； 5’b10100：CONTROL（DCRDR[25:24]），PRIMASK（DCRDR[0]）

30.3.4. DCRDR

偏移地址：0xF8
复位值：0x0000 0000

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

Bit	Name	Function
31:0	DBGTMP	core 寄存器要传输的数据

30.3.5. DEMCR

偏移地址：0xFC
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							DWTENA
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—					VC_HARD ERR	—	

类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RO-0
7:0	—							VC_CORE RESET
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
31:25	NA	保留位
24	DWTENA	DWT 模块配置和控制使能位。
23:11	NA	保留位
10	VC_HARDERR	进入 HardFault 中断错误
9:1	NA	保留位
0	VC_CORERESET	system reset 后暂停 core

30.4. MCU 调试模块寄存器映射

MCU 调试模块寄存器映射基地址为 0x40015800

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IDCODE	VERSION [31:28]				PARTNO[15:0]																DESIGNER[10:0]											-
	Reset	0	0	0	0	1	0	1	1	1	0	1	1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	1	1	0	1	1	1
0x04	DBGMCU_CR	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	DBG_STANDBY	DBG_STOP	-
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	x
0x08	DBGMCU_APB1 _FZ	-	-	-	-	-	-	-	-	-	-	DBG_I2C1_SMBUS_TIMEOUT	-	-	-	-	-	-	-	-	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	-	DBG_TIM14_STOP	-	-	DBG_TIM7_STOP	DBG_TIM6_STOP	-	-	DBG_TIM3_STOP	-
	Reset	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x	x	x	0	0	0	x	0	x	x	0	0	x	x	0	x
0x0C	DBGMCU_APB2 _FZ	-	-	-	-	-	-	-	-	-	-	-	-	-	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP	-	-	-	-	DBG_TIM1_STOP	-	-	-	-	-	-	-	-	-	-	-
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	x	x	x	x	0	x	x	x	x	x	x	x	x	x	x	x

30.4.1. 调试 MCU 配置寄存器 (IDCODE)

偏移地址：0x00

复位值：0x0BB11477

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	VERSION[3:0]				PARTNO[15:12]			
类型	RO	RO	RO	RO	RO	RO	RO	RO
23:16	PARTNO[11:4]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
15:8	PARTNO[3:0]				DESIGNER[11:8]			

类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	DESIGNER[7:1]							—
类型	RO	RO	RO	RO	RO	RO	RO	RO-1

Bit	Name	Function
0	NA	保留位，总是 1
31:28	VERSION	ECO 版本
27:12	PARTNO	DP 的型号；芯片使用 SW-DP ：0xBB11
11:1	DESIGNER	Designer ID；ARM 默认值 0x23B

30.4.2. 调试 MCU 配置寄存器 (DBGMCU_CR)

偏移地址：0x04
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
7:0	—					DBG_ STANDBY	DBG_ STOP	—
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RO-0

Bit	Name	Function
31:24,23:16, 15:8,7:3, 0	NA	保留位
2	DBG_STANDBY	0 :(FCLK 关 , HCLK 关) 整个数字电路部分都断电。从软件的观点看，退出 Standby 模式与复位是一样的（除了一些状态位指示了微控制器刚从 Standby 状态退出）。 1 :(FCLK 开，HCLK 开) 数字电路部分不下电，FCLK 和 HCLK 时钟由内部 RL 振荡器提 供时钟。另外，微控制器通过产生系统复位来退出 Standby 模式和复位是一样的。
1	DBG_STOP	0 :(FCLK 关，HCLK 关) 在停机模式时，时钟控制器禁止一切时钟（包括 HCLK 和 FCLK）。当从 STOP 模式退出时，时钟的配置和复

		位之后的配置一样（微控制器由 8MHz 的内 部振荡器（I）提供时钟）。因此，软件必须重新配置时钟控制系统启动 PLL，晶振等。 1：（FCLK 开，HCLK 开）在停机模式时，FCLK 和 HCLK 时钟由内部振荡器提供。当退出停机模式时，软件必须重新配置时钟系统启动 PLL，晶振等（与配置此比特位为 0 时的操作一样）。

30.4.3. 调试 MCU APB1 停止寄存器（DBGMCU_APB1_FZ）

偏移地址：0x08
复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	保留位							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—		DBG_I2C1_SMBUS_TIMEOUT	—				
类型	RO-0	RO-0	RW	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	—			DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	—	DBG_TIM14_STOP
类型	RO-0	RO-0	RO-0	RW	RW	RW	RO-0	RW
7:0	—		DBG_TIM7_STOP	DBG_TIM6_STOP	—		DBG_TIM3_STOP	—
类型	RO-0	RO-0	RW	RW	RO-0	RO-0	RW	RO-0

Bit	Name	Function
31:22, 20:13, 9,7:6,3:2, 0	NA	保留位
21	DBG_I2C1_SMBUS_TIMEOUT	当核暂停的时候 SMBUS 超时模式停止 0：正常模式的行为 1：SMBUS 超时模式不能使用
12	DBG_IWDG_STOP	当核暂停的时候，调试独立看门狗停止。 0：尽管核暂停，独立看门狗的计数器时钟在工作。 1：当核暂停的时候，独立看门狗的计数器时钟也停止了。
11	DBG_WWDG_STOP	当核暂停的时候，调试窗口看门狗停止。 0：尽管核暂停，窗口看门狗的计数器时钟在工作。 1：当核暂停的时候，窗口看门狗的计数器时钟也停止了。

10	DBG_RTC_STOP	当核暂停的时候，RTC 停止。 0：尽管核暂停，RTC 的计数器时钟在工作。 1：当核暂停的时候，RTC 的计数器时钟也停止了。
8	DBG_TIM14_STOP	当核暂停的时候，TIM14 计数器停止。 0：尽管核暂停，TIM14 的计数器时钟在工作。 1：当核暂停的时候，TIM14 的计数器时钟也停止了。
5	DBG_TIM7_STOP	当核暂停的时候，TIM7 计数器停止。 0：尽管核暂停，TIM7 的计数器时钟在工作。 1：当核暂停的时候，TIM7 的计数器时钟也停止了。
4	DBG_TIM6_STOP	当核暂停的时候，TIM6 计数器停止。 0：尽管核暂停，TIM6 的计数器时钟在工作。 1：当核暂停的时候，TIM6 的计数器时钟也停止了。
1	DBG_TIM3_STOP	当核暂停的时候，TIM3 计数器停止。 0：尽管核暂停，TIM3 的计数器时钟在工作。 1：当核暂停的时候，TIM3 的计数器时钟也停止了。

30.4.4. 调试 MCU APB2 停止寄存器 (DBGMCU_APB2_FZ)

偏移地址：0x0C

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	保留位							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—					DBG_ TIM17_ STOP	DBG_ TIM16_ STOP	DBG_ TIM15_ STOP
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
15:8	—				DBG_ TIM1_ STOP	—		
类型	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RO-0	RO-0
7:0	保留位							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0

31:19, 15:12, 10:0	NA	保留位
18	DBG_TIM17_STOP	当核暂停的时候，TIM17 计数器停止。 0：尽管核暂停，TIM17 的计数器时钟在工作。 1：当核暂停的时候，TIM17 的计数器时钟也停止了。

17	DBG_TIM16_STOP	当核暂停的时候，TIM16 计数器停止。 0：尽管核暂停，TIM16 的计数器时钟在工作。 1：当核暂停的时候，TIM16 的计数器时钟也停止了。
16	DBG_TIM15_STOP	当核暂停的时候，TIM15 计数器停止。 0：尽管核暂停，TIM15 的计数器时钟在工作。 1：当核暂停的时候，TIM15 的计数器时钟也停止了。
11	DBG_TIM1_STOP	当核暂停的时候，TIM1 计数器停止。 0：尽管核暂停，TIM1 的计数器时钟在工作。 1：当核暂停的时候，TIM1 的计数器时钟也停止了。

31. 器件电子签名

31.1. 概述

器件电子签名存储在闪存存储模块的系统存储区并且可以使用调试接口或者 CPU 来读取。它所包含的芯片识别信息在出厂时编写，用户固件或者外部设备可以读取电子签名，用以自动匹配不同配置的 FT32F0XX 系列微控制器。表示存储器容量的数据寄存器的地址为 0x1FFFF7CC。

31.2. 寄存器映射

地址偏移	名称	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	Flash_size	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Flash_size[15:0]															
	Reset	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.2.1. FLASH_SIZE

地址偏移: 0x00

复位值：0x0000 0000

位地址	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0
31:24	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
23:16	—							
类型	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0
15:8	FLASH_SIZE[15:8]							
类型	RO	RO	RO	RO	RO	RO	RO	RO
7:0	FLASH_SIZE[7:0]							
类型	RO	RO	RO	RO	RO	RO	RO	RO

31:16	NA	保留位
15:0	FLASH_SIZE	识别闪存存储器的尺寸，单位是 k 字节。 例如，0x40 相当于 64k 字节。

文档更改历史

日期	版本	内容
2019-10-30	0.1	初版 表格文字居中
2019-12-10	0.2	更正一些笔误
2020-5-13	0.3	修改格式
2020-11-24	0.4	添加 IO 采样保持电路应用的电路图和使用步骤说明，修改其相应时序图。
2020-12-04	0.4	修改 TIM3/TIM14/TIM15/TIM16/TIM17 中寄存器 IcxF 的相关描述
2021-1-11	0.5	添加注意事项：HCLK 低于 125K 时，Flash 无法完成擦写 改变 USER 字节地址 修改 Flash 寄存器名字和位名字 增加 7.2.8 时钟校准小节 更新 7.3 小节寄存器映射表格 修改 27.3.1 小节中写关键字相关的描述 增加 6.3.2 小节 PWR_CSR 寄存器的最高频率注意事项 增加 7.3.9 小节 RCC_BDCR 寄存器的最高频率注意事项 增加 7.3.10 小节 RCC_CSR 寄存器的最高频率注意事项 修改 7.3.10 小节 RCC_CSR 寄存器的 LSIRDY 位注意事项 增加 7.2.5 小节 LSI 时钟的控制注意事项 增加 26.3.1 小节 M1 寄存器位以及其描述 删除 RAM_PARITY_CHECK 位 修改 VDD_MONITOR 为 VDDA_MONITOR 修改 LATENCY 复位值为 1 更新 7.3 小节复位和时钟模块寄存器映射 修改 TIM1/TIM3/TIM14/TIM15/TIM16/TIM17 中寄存器 IcxF 的相关描述
2021-2-22	0.6	删除”3.2.3Flash 写和擦除操作”中 I 在 Flash 做写/擦除操作时的状态描述 修改 PWR 寄存器位笔误 修改”RCC_BDCR”寄存器复位值 修改 ADC 章节对于掉电模式的描述，修改 13.9 小节”读温度”的公式 修改比较器章节笔误 修改 I2C, SPI, USART, USB 寄存器描述笔误 修改 FLASH, RCC, TIMERx, RTC 寄存器映射表 修改 GPIO 寄存器映射表笔误和一些漏掉的地方 修改 9.3.2 章节中关于附加功能中比较器实际不需要配置 GPIO 寄存器的错误 更新 15.1.1 校准输入失调电压步骤
2021-4-21	0.7	修改 ADC 章节对于内部参考电压的描述，改为仅可选 2.5V
2021-4-21	0.8	更新 29.3 小节寄存器映射表格
2021-9-16	0.9	更正 RCC_CFGR3 寄存器 I2C1SW 位描述 更正 TSC_CFGR 寄存器 G1/G2/G3_PWM_SEL 位描述 更正 SPIx_CR1 寄存器 BR[2:0]位描述