PY32F040 系列

32 位 ARM® Cortex®-M0+ 微控制器

参考手册

目录

1. 寄存器描述中使用的缩写列表

缩写	描述
read/write (rw)	软件能读写此位
read-only (r)	软件只能读此位
write-only (w)	软件只能写此位，读此位将返回复位值
read/clear write0 (rc_w0)	软件可以读此位，也可以通过写 0 清除此位，写 1 对此位无影响
read/clear write1 (rc_w1)	软件可以读此位，也可以通过写 1 清除此位，写 0 对此位无影响
read/clear write (rc_w)	软件可以通过写入寄存器来读取和清除该位，写入该位的值并不重要
read/clear by read (rc_r)	软件可以读取这个位。读取此位会自动将其清除为 0，写入此位不会影响位值
read/set by read (rs_r)	软件可以读取这个位。读取此位会自动将其设置为 1，写入此位不会影响位值
read/set (rs)	软件可以读此位，也可以设置此位为 1，写 0 对此位无影响
toggle (t)	软件可以通过写入 1 来切换此位，写入 0 无效
Reserved (Res)	保留位，必须保持在重置值

2. 系统架构框图

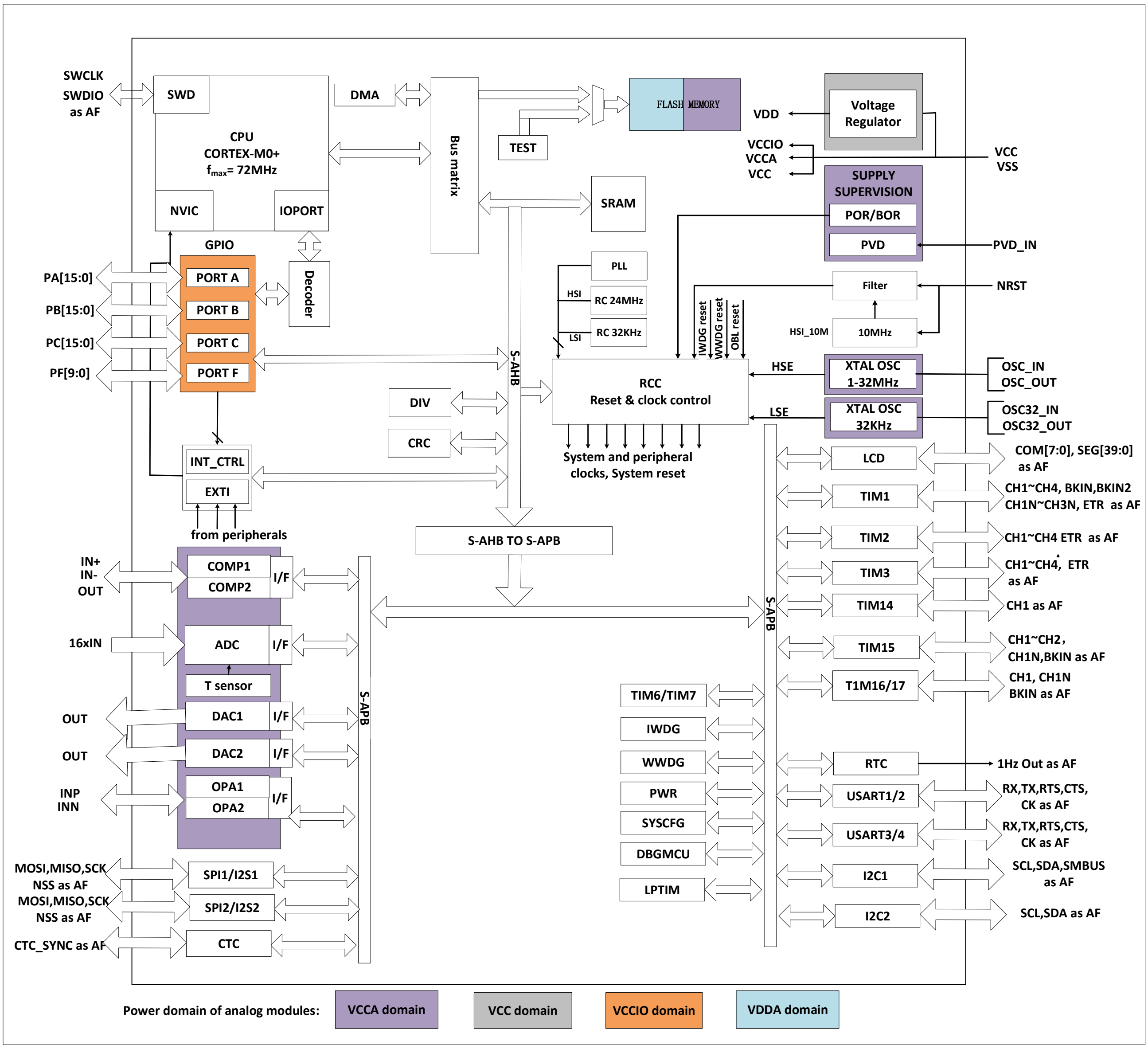


图 2-1 系统架构框图

3. 存储器和总线架构

3.1. 系统架构

系统由以下部分组成：

- 两个 Master
  - Cortex-M0+
  - 通用 DMA
- 三个 Slave
  - 内部 SRAM
  - 内部 Flash
  - 带 AHB-APB Bridge 的 AHB

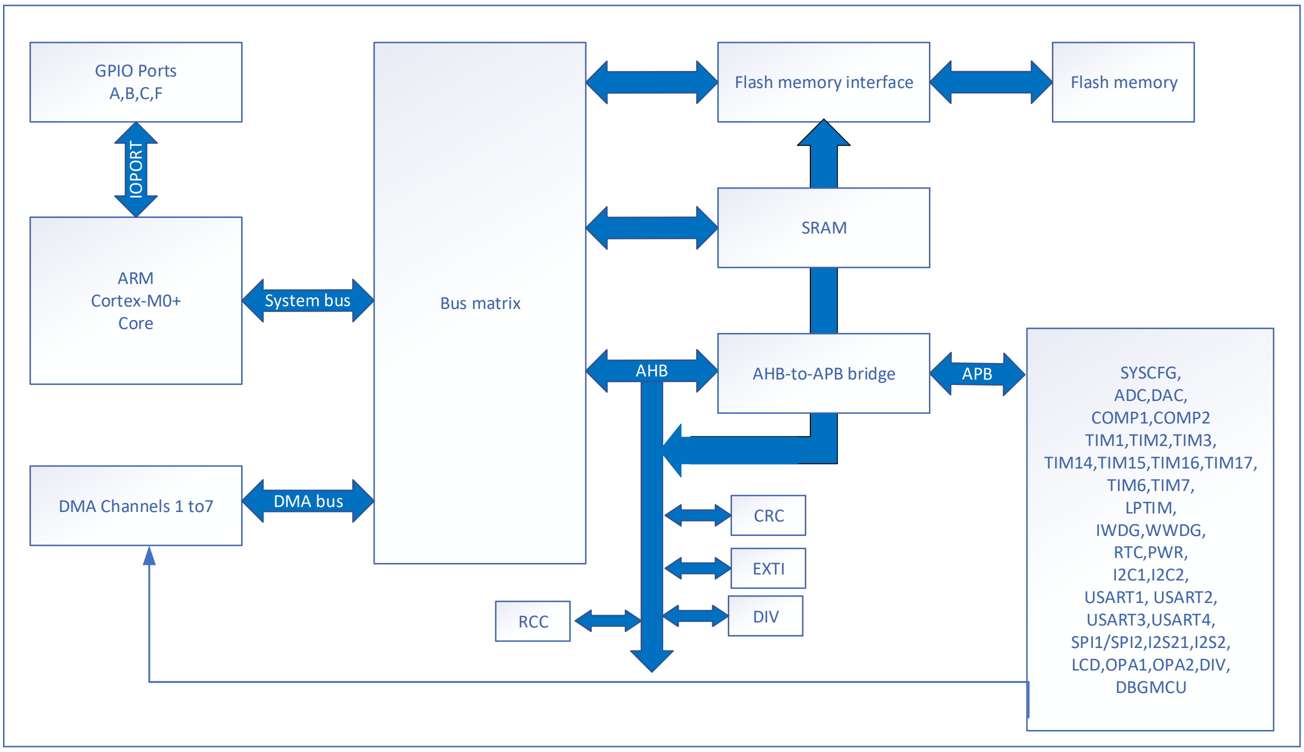


图 3-1 系统架构

- 系统总线

该总线把 Cortex-M0+的系统总线连接到总线 Matrix， 后者用来管理 CPU 和 DMA 的仲裁。

- DMA 总线

该总线把 DMA 的 AHB master 接口连接到总线 Matrix，由总线 Matrix 管理 CPU 和 DMA 对 SRAM、Flash 存储器和 AHB/APB 的外设访问。

- 总线 Matrix

总线 Matrix 负责 CPU 总线和 DMA 总线的总线仲裁。该仲裁使用 Round Robin 算法。总线 Matrix 由 Masters（CPU、DMA）和 slaves（Flash memory、SRAM 和 AHB-to-APB bridge）组成。

- AHB-to-APB bridge（APB）

The AHB-to-APB bridge负责AHB和APB总线之间的同步以及外设地址的映射。

3.2. 存储器结构简介

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

整个寻址空间被划分成 8 个 512 Mbyte 的 Block 区域。

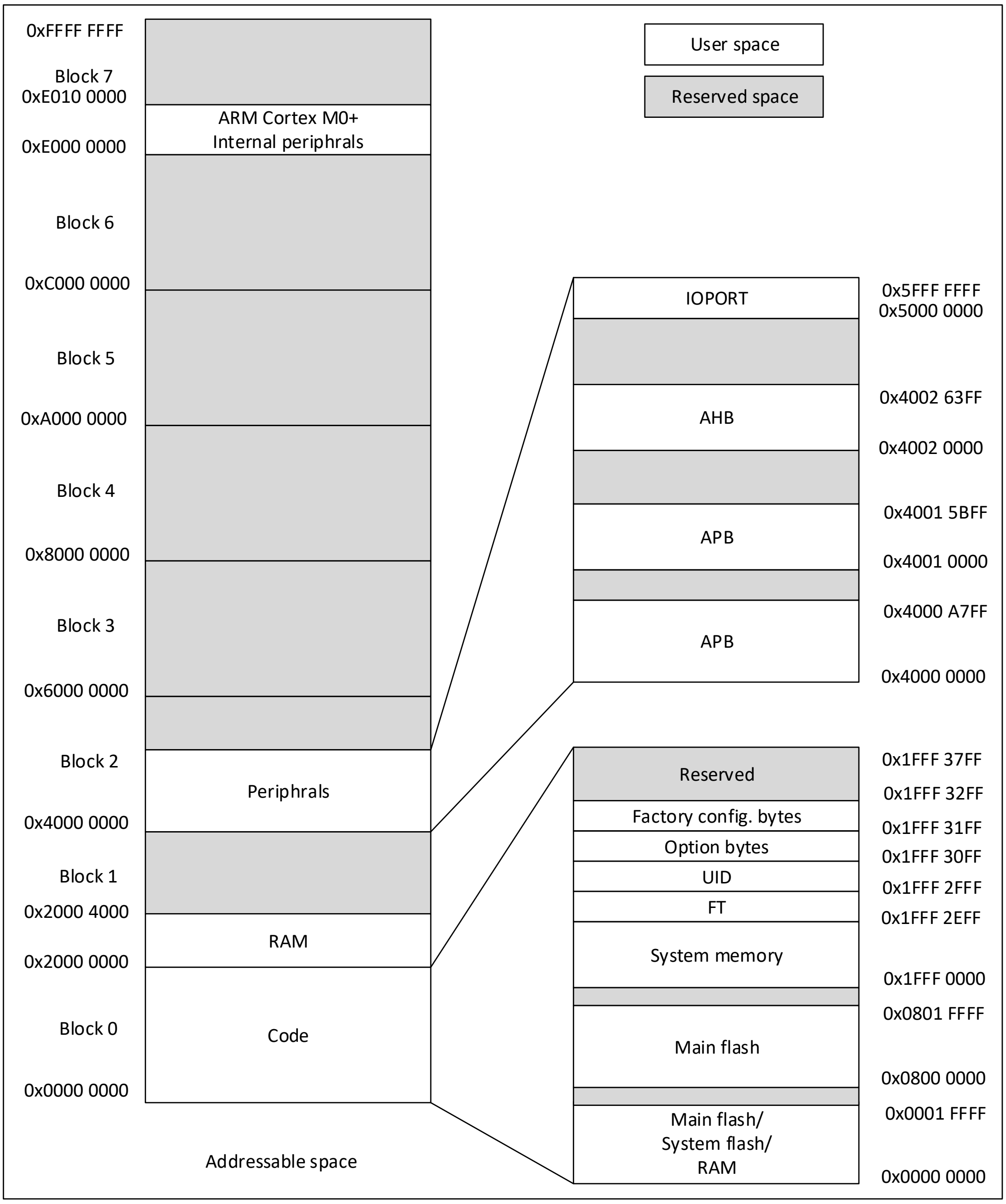


图 3-2 存储器映射

表 3-1 存储器地址

Type	Boundary Address	Size	Memory Area	Description
SRAM	0x2000 4000-0x3FFF FFFF	-	Reserved	-
	0x2000 0000-0x2000 3FFF	16 KBytes	SRAM	SRAM 最大为 16 KBytes
Code	0x1FFF 3400-0x1FFF FFFF	-	Reserved	-
	0x1FFF 3300-0x1FFF 33FF	-	Reserved	Reserved

Type	Boundary Address	Size	Memory Area	Description
	0x1FFF 3200-0x1FFF 32FF	256 Bytes	FT infor0 bytes	Factory config
	0x1FFF 3100-0x1FFF 31FF	256 Bytes	Option bytes	芯片软硬件 Option bytes 信息
	0x1FFF 3000-0x1FFF 30FF	256 Bytes	UID bytes	Unique ID
	0x1FFF 2F00-0x1FFF 2FFF	256 Bytes	FT bytes	FT bytes
	0x1FFF 0000-0x1FFF 2EFF	11.75 KBytes	System memory	存放 boot loader
	0x0802 0000-0x1FFE FFFF	-	Reserved	-
	0x0800 0000-0x0801 FFFF	128 KBytes	Main flash memory	-
	0x0002 0000-0x07FF FFFF	-	Reserved	-
	0x0000 0000-0x0001 FFFF	128 KBytes	根据 Boot 配置选择： 1）Main flash memory 2）System memory 3）SRAM	-

Note:

上述空间标注为 reserved 的空间，无法进行写操作，读为 0，且产生 response error。

表 3-2 外设寄存器地址

Bus	Boundary Address	Size	Peripheral
	0xE000 0000-0xE00F FFFF	1 MBytes	M0+
IOPORT	0x5000 1800-0x5FFF FFFF	256 MBytes	Reserved(1)
	0x5000 1400-0x5000 17FF	1 KBytes	GPIOF
	0x5000 1000-0x5000 13FF	1 KBytes	Reserved
	0x5000 0C00-0x5000 0FFF	1 KBytes	Reserved
	0x5000 0800-0x5000 0BFF	1 KBytes	GPIOC
	0x5000 0400-0x5000 07FF	1 KBytes	GPIOB
	0x5000 0000-0x5000 03FF	1 KBytes	GPIOA
AHB	0x4002 4000-0x4FFF FFFF	256 MBytes	Reserved
	0x4002 3C00-0x4002 3FFF	1 KBytes	Reserved
	0x4002 3800-0x4002 3BFF	1 KBytes	DIV
	0x4002 3400-0x4002 37FF	1 KBytes	Reserved
	0x4002 3000-0x4002 33FF	1 KBytes	CRC
	0x4002 2400-0x4002 2FFF	3 KBytes	Reserved
	0x4002 2000-0x4002 23FF	1 KBytes	FLASH
	0x4002 1C00-0x4002 1FFF	1 KBytes	Reserved
	0x4002 1800-0x4002 1BFF	1 KBytes	EXTI
	0x4002 1400-0x4002 17FF	1 KBytes	Reserved
	0x4002 1000-0x4002 13FF	1 KBytes	RCC(2)
	0x4002 0400-0x4002 0FFF	3 KBytes	Reserved
	0x4002 0000-0x4002 03FF	1 KBytes	DMA
APB	0x4001 5C00 - 0x4001 FFFF	41 KBytes	Reserved
	0x4001 5800 - 0x4001 5BFF	1 KBytes	DBG
	0x4001 4C00 - 0x4001 57FF	3 KBytes	Reserved
	0x4001 4800 - 0x4001 4BFF	1 KBytes	TIM17
	0x4001 4400 - 0x4001 47FF	1 KBytes	TIM16

Bus	Boundary Address	Size	Peripheral
	0x4001 4000 - 0x4001 43FF	1 KBytes	TIM15
	0x4001 3C00 - 0x4001 3FFF	1 KBytes	Reserved
	0x4001 3800 - 0x4001 3BFF	1 KBytes	USART1
	0x4001 3400 - 0x4001 37FF	1 KBytes	Reserved
	0x4001 3000 - 0x4001 33FF	1 KBytes	SPI1/I2S1
	0x4001 2C00 - 0x4001 2FFF	1 KBytes	TIM1
	0x4001 2800 - 0x4001 2BFF	1 KBytes	Reserved
	0x4001 2400 - 0x4001 27FF	1 KBytes	ADC
	0x4001 0400 - 0x4001 23FF	8 KBytes	Reserved
	0x4001 0300 - 0x4001 03FF	256 Bytes	OPA
	0x4001 0200 - 0x4001 02FF	256 Bytes	COMP
	0x4001 0000 - 0x4001 01FF	512 Bytes	SYSCFG
	0x4000 8000- 0x4000 FFFF	32 KBytes	Reserved
	0x4000 7C00 - 0x4000 7FFF	1 KBytes	LPTIM1
	0x4000 7800 - 0x4000 7BFF	1 KBytes	Reserved
	0x4000 7400 - 0x4000 77FF	1 KBytes	Reserved
	0x4000 7000 - 0x4000 73FF	1 KBytes	PWR(3)
	0x4000 6C00 - 0x4000 6FFF	1 KBytes	CTC
	0x4000 6800 - 0x4000 6BFF	1 KBytes	Reserved
	0x4000 6400 - 0x4000 67FF	1 KBytes	Reserved
	0x4000 6000 - 0x4000 63FF	1 KBytes	SRAM
	0x4000 5C00 - 0x4000 5FFF	1 KBytes	Reserved
	0x4000 5800 - 0x4000 5BFF	1 KBytes	I2C2
	0x4000 5400 - 0x4000 57FF	1 KBytes	I2C1
	0x4000 5000 - 0x4000 53FF	1 KBytes	Reserved
	0x4000 4C00 - 0x4000 4FFF	1 KBytes	USART4
	0x4000 4800 - 0x4000 4BFF	1 KBytes	USART3
	0x4000 4400 - 0x4000 47FF	1 KBytes	USART2
	0x4000 3C00 - 0x4000 43FF	2 KBytes	Reserved
	0x4000 3800 - 0x4000 3BFF	1 KBytes	SPI2/I2S2
	0x4000 3400 - 0x4000 37FF	1 KBytes	Reserved
	0x4000 3000 - 0x4000 33FF	1 KBytes	IWDG
	0x4000 2C00 - 0x4000 2FFF	1 KBytes	WWDG
	0x4000 2800 - 0x4000 2BFF	1 KBytes	RTC
	0x4000 2400 - 0x4000 27FF	1 KBytes	LCD
	0x4000 2000 - 0x4000 23FF	1 KBytes	TIM14
	0x4000 1800 - 0x4000 1FFF	2 KBytes	Reserved
	0x4000 1400 - 0x4000 17FF	1 KBytes	TIM7
	0x4000 1000 - 0x4000 13FF	1 KBytes	TIM6
	0x4000 0800 - 0x4000 0FFF	2 KBytes	Reserved
	0x4000 0400 - 0x4000 07FF	1 KBytes	TIM3
	0x4000 0000 - 0x4000 03FF	1 KBytes	TIM2

Note:

（1） IOPORT、AHB、APB 标注为 Reserved 的地址空间，无法写操作，读回为 0，不会产生 hardfault。

（2） 不仅支持 32 位字访问，还支持半字和字节访问。

（3） 不仅支持 32 位字访问，还支持半字访问。

3.3. 嵌入式 SRAM

片内最大集成 16 KB SRAM。通过 bytes、half-word（16 位）或者 word（32 位）的方式可访问 SRAM。软件对设定范围外空间的读写操作，会产生 hard fault。

3.4. Flash 存储器

Flash 存储器有两个不同的物理区域组成：

- Main flash 区域，128 KBytes，它包含应用程序和用户数据。软件对设定范围外空间的访问会产生 hard fault。
- Information 区域，14 Kbytes，它包括以下部分：
  - FT infor0 bytes：256 Bytes，用于存放 Normal 和 High TS DATA，HSI Re-trimming 数据；
  - Option bytes：256 Bytes，用于存放芯片软硬件 Option Bytes 信息;
  - UID：256 Bytes，用于存放芯片的 UID；
  - System memory（系统存储器）：11.75 KBytes，用于存放 Boot loader。

Flash 接口实现基于 AHB 协议的指令读取和数据访问，它也通过寄存器实现了 flash 的基本 program/erase 等操作。

3.5. Boot 模式

通过 BOOT0 pin 和 boot 配置位 nBOOT1（存放于 Option bytes 中），可选择三种不同的启动模式，如下表所示：

表 3-3 Boot 配置

Boot mode configuration		Mode
nBOOT1 bit	BOOT0 pin	
X	0	选择 Main flash 作为启动区
1	1	选择 System memory 作为启动区
0	1	选择 SRAM 作为启动区

启动模式配置在系统复位后的第 4 个 SYSCLK 进行锁存。由用户按照上表决定选择哪种启动模式。

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

- Boot from main flash：main flash 跟启动存储器空间的 0x0000 0000 对齐，但是仍然可以按其本来的存储器空间（0x0800 0000）进行访问。也就是说，Flash 空间可以从地址 0x0000 0000 或者 0x0800 0000 访问到。
- Boot from system memory：system memory 对齐在启动存储器空间 0x0000 0000，但是仍然可以从它本来的地址空间 0x1FFF 0000 访问到。
- Boot from SRAM：SRAM 对齐在启动存储器空间的 0x0000 0000，但是仍然可以通过 0x2000 0000 地址访问到。

3.5.1. 存储器物理映像

如果 boot mode 被选择，应用软件可以修改在程序空间可被访问的存储器。这个修改通过 SYS-CFG_CFGR1 寄存器的 MEM_MODE 位选择决定（详见 SYSCFG 章节）。

3.5.2. 内嵌的自举程序

Bootloader 在芯片生产阶段被写入，并存放在 system memory 中。它用来使用下面串行接口进行对 flash 存储器的再次写入：

- USART1，PA9/PA10；USART2，PA14/PA15；USART3，PB10/PB11；USART4，PC10/PC11；

4. 嵌入式闪存

4.1. 闪存主要特性

- Main flash block: 最大 128 kBytes
- Information block: 14 kBytes
- Page size: 256 Bytes
- Sector size: 8 kBytes

闪存控制接口电路的主要特征如下:

- 闪存写和擦除
- 选项字节的编程操作
- 读保护
- 写保护
- SDK 保护

4.2. 闪存功能介绍

4.2.1. 闪存结构

Flash 存储器由 64 位宽的存储单元组成，可以用作程序和数据的存储，Page 大小为 256 Bytes，Sector 大小为 8 KBytes。

从功能上，Flash 存储器分为 Main flash 和 information flash，前者容量最大是 128 Kbytes，后者容量为 14 KBytes。

表 3-4 闪存结构及边界地址

Block	扇区 sector	页 Page	Base address	Size
Main flash	Sector 0	Page 0-31	0x0800 0000-0x0800 1FFF	8Kbytes
	Sector 1	Page 32-63	0x0800 2000-0x0800 3FFF	8Kbytes
	Sector 2	Page 64-95	0x0800 4000-0x0800 5FFF	8Kbytes
	…	…	…	…
	Sector 14	Page 448-479	0x0801 C000-0x0801 DFFF	8Kbytes
	Sector 15	Page 480-511	0x0801 E000-0x0801 FFFF	8Kbytes
System flash	INFO	Page 0-46	0x1FFF 0000-0x1FFF 2EFF	11.75Kbytes
FT		Page 47	0x1FFF 2F00-0x1FFF 2FFF	256bytes
UID		Page 48	0x1FFF 3000-0x1FFF 30FF	256bytes
Option bytes		Page 49	0x1FFF 3100-0x1FFF 31FF	256bytes
Factory config		Page 50	0x1FFF 3200-0x1FFF 32FF	256bytes
Reserved		Page 51-55	0x1FFF 3300-0x1FFF 37FF	1280bytes

4.2.2. 闪存读操作和访问延迟

Flash 可以被作为一个通用的存储器空间，被直接寻址访问。通过专门的读控制时序，可以对 flash 存储器的内容进行读取。取指和数据访问都是通过 AHB 总线进行的。读操作可以被 FLASH_ACR 寄存器的 Latency 位控制，即读取 flash 增加一个或者不增加等待状态。

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

4.2.3. 闪存写操作和擦除操作

通过 ICP（In-circuit programming）或者 IAP（In-application programming）可以对 flash 进行 program 操作。

ICP：用来更新整个 Flash 存储器的内容，可以使用 SWD 协议或者 boot loader，把用户应用装入 MCU 中。ICP 提供了快速和有效的设计迭代，并消除了不必要的包处理或者 socketing。

IAP：可以使用芯片支持的通讯接口，下载要 program 的数据到 flash 中。IAP 允许用户在应用运行时，再次 program flash 存储器。然后，此时 flash 存储器中已有了之前使用 ICP 编程进去的部分应用程序。

如果在进行闪存写和擦除操作时，发生了复位，则闪存存储器的内容是不被保护的。

在闪存写和擦除操作期间，任何读闪存的操作都会拖延总线。写或擦除操作一结束，读操作就可以正确的进行。这也就意味着，当正在进写和擦除操作时，不能进行代码和数据的读取。

对于写和擦除操作，必须打开 HSI。

4.2.3.1. 闪存解锁

在复位后，flash 存储器会被保护，防止不想要的（比如电干扰引起的）写和擦除操作。写 FLASH_CR 寄存器是不被允许的（除了用作 reload option bytes 的 OBL_LAUNCH 位）。每次对 flash 的写和擦除操作，都必须通过写 FLASH_KEYR 寄存器，产生 Unlock 时序，启用 FLASH_CR 寄存器的访问。

具体步骤如下：

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

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

任何错误的时序都会锁住 FLASH_CR 寄存器，直到下一次复位。在错误的 KEY 时序时，总线错误被发现，并产生 Hard Fault 中断。这样的错误包括第一个写周期的 KEY1 不匹配，或者 KEY1 匹配，但第二个写周期的 KEY2 不匹配。

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

另外，当 FLASH_SR 寄存器的 BSY 位被置位时，FLASH_CR 寄存器不能被写。此时，任何尝试进行写该寄存器（FLASH_CR）的操作会引起 AHB 总线的拖延，直到 BSY 位被清零。

4.2.3.2. 闪存写操作

Flash 存储器每次以 32bit word 为单位（进行 half word 或者 byte 操作会产生 hardfault）进行整个 page 的 program 操作。当 FLASH_CR 寄存器的 PG 位被置位，CPU 向 FLASH 存储器地址空间写 32bit 数据时，program 操作开始启动。任何非 32bit 的写入将导致 hard fault 中断。

如果要 program 的 flash 地址空间，是被 FLASH_WRPR 寄存器设置为保护的区域，则 program 操作会被忽略掉，同时 FLASH_SR 寄存器 WRPERR 位会被置位。Program 操作的结束，FLASH_SR 寄存器的 EOP 位会被置位。

具体 flash program 的操作步骤如下所示：

1) 检查 FLASH_SR 寄存器的 BSY 位，判断是否当前没有正在继续的 flash 操作
2) 如果没有正在进行的 flash erase 或者 program 操作，则软件读出该 Page 的 64 个 word（如果该 page 已有数据存放，则进行该步骤，否则跳过该步骤）
3) 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护

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

当上述步骤 7）成功执行，则 program 操作自动启动，同时 BSY 位被硬件置位。

闪存擦除操作

Flash 存储器可以按照 page 进行 erase 操作，或者进行 sector 和 mass erase。

4.2.3.3. 页擦除

当某个 page 被 WRP 保护，它是不会被 erase 的，此时 WRPERR 位被置位。当要进行 page erase 操作时，要进行以下步骤:

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

4.2.3.4. 片擦除

Mass erase 用来对整片 main flash 进行 erase 操作。另外，当 WRP 被使能，mass erase 功能无效，不会产生 mass erase 操作，并且 WRPERR 位被置位。

进行 mass erase 的步骤如下:

1) 检查 BSY 位，确认是否没有正在进行的 Flash 操作
2) 向 FLASH_KEYR 寄存器依次写 KEY1,KEY2，解除 FLASH_CR 寄存器保护
3) 置位 FLASH_CR 寄存器的 MER 位和 EOPIE 位
4) 向 flash 的任意 main flash 空间写任意数据（32bit 数据）
5) 等待 BSY 位被清零
6) 检查 EOP 标志位被置位
7) 清零 EOP 标志

4.2.3.5. 扇区擦除

Sector erase 用来对 8Kbytes 的 main flash 进行 erase 操作。另外，当某个 sector 被 WRP 保护，它是不会被 erase 的，此时 WRPERR 位被置位。

进行 sector erase 的步骤如下:

- 检查 BSY 位，确认是否没有正在进行的 Flash 操作
- 向 FLASH_KEYR 寄存器依次写 KEY1、KEY2，解除 FLASH_CR 寄存器保护
- 置位 FLASH_CR 寄存器的 SER 位和 EOPIE 位
- 向该 sector 写任意数据
- 等待 BSY 位被清零
- 检查 EOP 标志位被置位
- 清零 EOP 标志

4.2.3.6. 写和擦除时间配置

Flash 的写和擦的时间需要进行严谨的控制，否则会造成操作失败。如果需要对 Flash 进行写和擦的操作，需要根据 HSI 输出频率，参考 FLASH_TS0, FLASH_TS1, FLASH_TS2P, FLASH_TPS3, FLASH_TS3, FLASH_PERTPE, FLASH_SMERTPE, FLASH_PRGTPE, FLASH_PRETPE 的描述对 Flash 写和擦时间控制寄存器进行正确的配置。

4.3. Flash 选项字节

4.3.1. Flash 选项字

芯片内的 flash 的 information 区域的部分区间作为选项字节使用，用来存放芯片或者用户针对应用需要对硬件进行的配置。比如，看门狗可以选择为硬件或者软件模式。

为了数据的安全性，选项字节以正文及反码形式分别存储。

表 3-5 选项字节格式

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

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

- FLASH user option 寄存器（FLASH_OPTR）
- FLASH SDK area address 寄存器（FLASH_SDKR）
- FLASH WRP address 寄存器（FLASH_WRPR）

表 3-6 选项字节结构

word 地址	描述
0x1FFF 3100	Option byte for Flash User option and its complemented
0x1FFF 3108	Option byte for Flash SDK area address and its complemented
0x1FFF 3110	Reserved
0x1FFF 3118	Option byte for Flash WRP address and its complemented
0x1FFF 3120	Reserved
0x1FFF 3128	Reserved
…	Reserved
…	Reserved
…	Reserved
0x1FFF 31F8	Reserved

1) Option byte for Flash User option

Flash memory address: 0x1FFF 3100

Production value:0x2755 D8AA

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

Bit	Name	R/W	Function
31	~IWDG_STOP	R	IWDG_STOP 的反码
30	~ nBOOT1	R	nBOOT1 的反码
29	~ NRST_MODE	R	NRST_MODE 的反码
28	~ WWDG_SW	R	WWDG_SW 的反码
27	~ IWDG_SW	R	IWDG_SW 的反码
26:24	Resrved	-	Resrved
23：16	~RDP	R	RDP 的反码
15	IWDG_STOP	R	与设置 iwdg 在 stop 模式下定时器运行状态 0：freze 定时器 1：正常运行
14	nBOOT1	R	与 BOOT PIN 一起，选择芯片启动模式
13	NRST_MODE	R	0：仅复位输入 1：GPIO 功能
12	WWDG_SW	R	0：硬件 watchdog 1：软件 watchdog
11	IWDG_SW	R	0：硬件 watchdog 1：软件 watchdog
10：8	Resrved	-	
7：0	RDP	R	0xAA：level 0, read protection inactive 非 0xAA：level 1, read protection active

2) Option byte for flash SDK area address

Flash memory address: 0x1FFF 3108

Production value: 0xFFE0 001F

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

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

Bit	Name	R/W	Function
31：29	~BOR_LEV[2:0]	R	BOR_LEV[2:0]的反码
28：24	~SDK_END[4:0]	R	SDK_END 的反码
23：22	Reserved	-	-
21	~BOR_EN	R	BOR_EN 的反码
20：16	~SDK_STRT[4:0]	R	SDK_STRT 的反码
15：13	BOR_LEV[2:0]	R	000：BOR 上升阈值为 1.8 V，下降阈值为 1.7 V 001：BOR 上升阈值为 2.0 V，下降阈值为 1.9 V 010：BOR 上升阈值为 2.2 V，下降阈值为 2.1 V 011：BOR 上升阈值为 2.4 V，下降阈值为 2.3 V 100：BOR 上升阈值为 2.6 V，下降阈值为 2.5 V 101：BOR 上升阈值为 2.8 V，下降阈值为 2.7 V 110：BOR 上升阈值为 3.0 V，下降阈值为 2.9 V 111：BOR 上升阈值为 3.2 V，下降阈值为 3.1 V
12：8	SDK_END[4:0]	R	SDK area end address，每一位对应的 STEP 为 4Kbytes
7：6	Reserved	-	-
5	BOR_EN	R	BOR enable 0：BOR 不使能 1：BOR 使能，BOR_LEV 起作用
4：0	SDK_STRT[4:0]	R	SDK area start address，每一位对应的 STEP 为 4 Kbytes

3) Option byte for Flash WRP address

Flash memory address: 0x1FFF 3118

Production value: 0x0000 FFFF

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

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

4.3.2. Flash 选项字节写

复位后，FLASH_CR 寄存器中与选项字节相关的位是被写保护的。当对选项字节进行相关操作前，FLASH_CR 寄存器中的 OPTLOCK 位必须被清零。

以下步骤用来解锁该寄存器：

1) 通过 Unlock 时序，unlock FLASH_CR 寄存器的写保护
2) 向 FLASH_OPTKEYR 寄存器，写 OPTKEY1=0x0819 2A3B
3) 向 FLASH_OPTKEYR 寄存器，写 OPTKEY2=0x4C5D 6E7F

任何错误的时序都会 lock 住 FLASH_CR 寄存器，直到下一次复位。在错误的 KEY 时序时，总线错误被发现，并产生 Hard Fault 中断。

User option（information flash 的 option bytes）可以通过软件写 FLASH_CR 寄存器的 OPTLOCK 位，被保护住，以防止不想要的 erase/program 操作。

如果软件置位 Lock 位，则 OPTLOCK 位也被自动置位。

Modifying user option bytes

选项字节的写操作（program），跟对 Main flash 的操作不一样。为修改选项字节，需要进行如下步骤：

- 用之前描述的步骤，清零 OPTLOCK 位
- 检查 BSY 位，确认没有正在进行的 Flash 操作
- 向 option bytes 寄存器 FLASH_OPTR/FLASH_SDKR/FLASH_WRPR 写期望的值（1~3 个 word）
- 置位 OPTSTRT 位
- 向 main flash 0x4002 2080 地址写任意 32bit 数据（触发正式的写操作）
- 等待 BSY 位被清零
- 等待 EOP 拉高，软件清零

任何对 Option bytes 的改动，硬件都会先把 option byte 对应的整个 page erase 掉，然后用 FLASH_OPTR、FLASH_SDKR 或者 FLASH_WRPR 寄存器的值，写到 option bytes 中。并且，硬件自动计算相应的补码，并把计算值写到 option bytes 的相应区域。

Option byte loading

在 BSY 位被清零后，所有新的 option bytes 被写入了 flash information 存储器中，但是未应用于芯片系统。对 option bytes 寄存器进行读操作，仍然返回上一次被装载的 option bytes 里的值。仅当他们（新值）被装载后，才对芯片系统起作用。

Option bytes 的装载，在以下两种情况下进行：

a) 当 FLASH_CR 寄存器中的 OBL_LAUNCH 位被置位
b) 在上电复位后（POR、BOR）

“装载 option bytes”进行的操作是：对 information memory 区域的 option bytes 进行读操作，再把读出的数据存储在内部 option 寄存器中（FLASH_OPTR、FLASH_SDKR 和 FLASH_WRPR）。这些内部寄存器配置系统，并可以被软件读。置位 OBL_LAUNCH 位，产生了一个复位，这样 option bytes 的装载，才能在系统的复位下进行。

每个 option 位在它相同的双字地址（下一个 half word）有相应的补码。在 option bytes 装载期间，会对 option bit 和其补码的验证，这能确保装载被正确地进行了。

如果正补码匹配，则 option bytes 被 copy 到 option 寄存器中。

如果正补码不匹配，则 FLASH_SR 寄存器的 OPTVERR 状态位被置位。不匹配的值被写入 option 寄存器：

- 对于 user option
  - BOR_LEV 写成 000（最低阈值）
  - BOR_EN 位写成 0（BOR 不使能）
  - NRST_MODE 位写成 0（仅复位输入）
  - RDP 位写成 0xff（即 level 1）
  - 其余不匹配的值都写成 1

- 对于 SDK area option，SDKR_STRT[4:0]= 0x00，SDKR_END[4:0]=0x1F，即所有 flash 空间都被设定为 SDK
- 对于 WRP option，不匹配的值是缺省值“无保护”

在系统复位后，option bytes 的内容被 copy 到下面的 option 寄存器（软件可读可写）：

- FLASH_OPTR
- FLASH_SDKR
- FLASH_WRPR

这些寄存器也被用来修改 option bytes。如果这些寄存器不被用户修改，他们体现了系统 option 的状态。

4.4. Flash 配置字节

芯片内的 flash 的 information 区域的部分区间（共 1 个 page）作为 Factory config. byte 使用。

Page 0 存放供软件读取信息（仅有正码，无反码存放）：

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

表 3-7 Factory config. byte organization

Page	Word	Address	Contents
0	0	0x1FFF 3200	存放 HSI 4 MHz 频率选择控制及对应的 Trimming 值
	1	0x1FFF 3208	存放 HSI 8 MHz 频率选择控制及对应的 Trimming 值
	2	0x1FFF 3210	存放 HSI 16 MHz 频率选择控制及对应的 Trimming 值
	3	0x1FFF 3218	存放 HSI 22.12 MHz 频率选择控制及对应的 Trimming 值
	4	0x1FFF 3220	存放 HSI 24 MHz 频率选择控制及对应的 Trimming 值
	5	0x1FFF 3228	TS_CAL1， 30℃温度传感器的校准值
	6	0x1FFF 3230	TS_CAL2， 85℃温度传感器的校准值
	7	0x1FFF 3238	存放 HSI 4 MHz 频率下对应的 FLASH_TS0、FLASH_TS1、 FLASH_TS3 寄存器的配置值
	8	0x1FFF 3240	存放 HSI 4 MHz 频率下对应的 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	9	0x1FFF 3248	存放 HSI 4 MHz 频率下对应的 FLASH_PERTPE 寄存器的配置值
	10	0x1FFF 3250	存放 HSI 4 MHz 频率下对应的 FLASH_SMERTPE 寄存器的配置值
	11	0x1FFF 3258	存放 HSI 4 MHz 频率下对应的 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值
	12	0x1FFF 3260	存放 HSI 8 MHz 频率下对应的 FLASH_TS0、FLASH_TS1、 FLASH_TS3 寄存器的配置值
	13	0x1FFF 3268	存放 HSI 8 MHz 频率下对应的 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	14	0x1FFF 3270	存放 HSI 8 MHz 频率下对应的 FLASH_PERTPE 寄存器的配置值
	15	0x1FFF 3278	存放 HSI 8 MHz 频率下对应的 FLASH_SMERTPE 寄存器的配置值
	16	0x1FFF 3280	存放 HSI 8 MHz 频率下对应的 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值
	17	0x1FFF 3288	存放 HSI 16 MHz 频率下对应的 FLASH_TS0、FLASH_TS1、 FLASH_TS3 寄存器的配置值
	18	0x1FFF 3290	存放 HSI 16 MHz 频率下对应的 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	19	0x1FFF 3298	存放 HSI 16 MHz 频率下对应的 FLASH_PERTPE 寄存器的配置值
	20	0x1FFF 32A0	存放 HSI 16 MHz 频率下对应的 FLASH_SMERTPE 寄存器的配置值

	21	0x1FFF 32A8	存放 HSI 16 MHz 频率下对应的 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值
	22	0x1FFF 32B0	存放 HSI 22.12 MHz 频率下对应的 FLASH_TS0、FLASH_TS1、 FLASH_TS3 寄存器的配置值
	23	0x1FFF 32B8	存放 HSI 22.12 MHz 频率下对应的 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	24	0x1FFF 32C0	存放 HSI 22.12 MHz 频率下对应的 FLASH_PERTPE 寄存器的配置值
	25	0x1FFF 32C8	存放 HSI 22.12 MHz 频率下对应的 FLASH_SMERTPE 寄存器的配置值
	26	0x1FFF 32D0	存放 HSI 22.12 MHz 频率下对应的 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值
	27	0x1FFF 32D8	存放 HSI 24 MHz 频率下对应的 FLASH_TS0、FLASH_TS1、 FLASH_TS3 寄存器的配置值
	28	0x1FFF 32E0	存放 HSI 24 MHz 频率下对应的 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	29	0x1FFF 32E8	存放 HSI 24 MHz 频率下对应的 FLASH_PERTPE 寄存器的配置值
	30	0x1FFF 32F0	存放 HSI 24 MHz 频率下对应的 FLASH_SMERTPE 寄存器的配置值
	31	0x1FFF 32F8	存放 HSI 24MHz 频率下对应的 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值

4.4.1. HSI_TRIMMING_FOR_USER

Address: 0x1FFF 3200~0x1FFF 3220

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

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

4.4.2. 温度传感器的校准值

Address: 0x1FFF 3228(30℃)、0x1FFF 3230(85℃)

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

软件需要从该地址读出数据。

4.4.3. HSI_4M/8M/16M/22.12M/24M_EPPARA0

Address: 0x1FFF 3238(4MHz)、0x1FFF 3260(8MHz)、0x1FFF 3288(16MHz)、0x1FFF 32B0(22.12MHz)、0x1FFF 32D8(24MHz)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	TS1[8:0]								
							R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TS3[7:0]								TS0[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件需要根据需要设定的 HSI 时钟频率，选择从相应地址读出数据，再写入 FLASH_TS0、FLASH_TS1、FLASH_TS3 寄存器，以实现对应 HSI 频率所需的擦写时间的配置。

4.4.4. HSI_4M/8M/16M/22.12M/24M_EPPARA1

Address: 0x1FFF 3240(4MHz)、0x1FFF 3268(8MHz)、0x1FFF 3290(16MHz)、0x1FFF 32B8(22.12MHz)、0x1FFF 32E0(24MHz)

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

软件需要根据需要设定的 HSI 时钟频率，选择从相应地址读出数据，再写入 FLASH_TS2P、FLASH_TPS3 寄存器，以实现对应 HSI 频率所需的擦写时间的配置。

4.4.5. HSI_4M/8M/16M/22.12M/24M_EPPARA2

Address: 0x1FFF 3248(4MHz)、0x1FFF 3270(8MHz)、0x1FFF 3298(16MHz)、0x1FFF 32C0(22.12MHz)、0x1FFF 32E8(24MHz)

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

软件需要根据需要设定的 HSI 时钟频率，选择从相应地址读出数据，再写入 FLASH_PERTPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.4.6. HSI_4M/8M/16M/22.12M/24M_EPPARA3

Address: 0x1FFF 3250(4MHz)、0x1FFF 3278(8MHz)、0x1FFF 32A0(16MHz)、0x1FFF 32C0(22.12MHz)、0x1FFF 32F0(24MHz)

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

软件需要根据需要设定的 HSI 时钟频率，选择从相应地址读出数据，再写入 FLASH_SMERTPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.4.7. HSI_4M/8M/16M/22.12M/24M_EPPARA4

Address: 0x1FFF 3258(4MHz)、0x1FFF 3280(8MHz)、0x1FFF 32A8(16MHz)、0x1FFF 32D0(22.12MHz)、0x1FFF 32F8(24MHz)

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

软件需要根据需要设定的 HSI 时钟频率，选择从相应地址读出数据，再写入 FLASH_PRGTPE 和 FLASH_PRETPE 寄存器中，以实现对应 HSI 频率所需的擦写时间的配置。

4.5. 闪存保护

对 Flash main memory 的保护包括以下几种机制：

(1) SDK（software design kit）的保护，用来对特定程序区的访问保护，粒度是 4Kbytes。
(2) read protection(RDP)，防止来自外部的访问。
(3) wrtie protection（WRP）控制，以防止不想要的写操作（由于程序存储器指针 PC 的混乱）。写保护的粒度设计为 8Kbytes。
(4) Option byte 写保护，专门的解锁设计。

4.5.1. 闪存软件开发包(SDK)区域保护

保护区域由 FLASH_SDKR 寄存器的 SDKR_STRT[4:0],SDKR_END[4:0]定义，每一个 bit 对应 4Kbytes。

Start address

FLASH memory base address + SDK_STRT[4:0] x 0x1000(included)

End address

FLASH memory base address + (SDK_END[4:0]+1) x 0x1000(excluded)

当 SDK_STRT[4:0]大于 SDK_END[4:0]时，SDK 保护无效；当 SDK_STRT[4:0]小于或等于 SDK_END[4:0]时，SDK 保护有效。

在保护生效状态下，对 FLASH_SDKR 寄存器解除保护时（写 SDK_STRT[4:0]大于 SDK_END[4:0]），硬件会先触发 mass erase（SDK 区域被保护的程序之前已经写入，通过 mass erase 起到了对 SDK 区域程序保护的作用），然后再更新 flash option byte 中的 SDK option 的值（此时更新的值是 SDK 保护无效）。

此时，FLASH_SDKR 寄存器的内容不会更新，直到上电复位（POR/BOR/PDR）或者 OBL 复位，寄存器内容才会被从 flash option byte 中的 SDK option 装载到寄存器中。

4.5.2. 闪存读保护

通过设置 RDP option byte，并进行 system reset(POR/BOR 或者 OPL 复位)装载新的 RDP option byte，可以激活读保护功能。RDP 保护 flah main memory、option byte、SRAM。

如果通过 SWD 的 debug 仍在连接时，读保护被设置，需要进行上电复位而不是系统复位。

当 RDP option byte 和补码成对正确存在于 option byte 时，Flash memory 会被保护。

表 3-8 闪存读保护状态

RDP byte value	RDP complemented byte value	Read protection level
0xAA	0x55	Level 0
除(0xAA 和 0x55)组合的任何值		Level 1

无论任何保护级别，System memory 只能读访问，不能进行擦写操作。

Level 0: no protection

对 main flash 的读、program 和擦操作是可能的，对 option byte 也是可以进行任何操作。

Level 1: Read protection

当 option byte 里的 RDP 及其补码包含任何（0xAA、0x55）之外的组合，则 level 1 read protection 生效，Level 1 是缺省的保护级别。

- User mode：在用户模式下执行的程序（boot from main flash），可以对 main flash、option byte 进行所有操作。
- Debug, 从 SRAM 启动以及从 system memory 启动（Boot loader）:在 debug 模式，或者当从 SRAM 或者 system memory（Boot loader）启动，main flash 是不能被访问的。在这些模式下，对 main flash

读或者写访问产生一个 bus error，以及产生一个 hard fault 中断。

当已处于 Level 1（0xAA 之外任何数），如果要修改为 Level 0（写 0xAA），硬件会对 main flash 进行 mass erase 操作。

表 3-9 访问状态与保护级别和执行模式的关系

Area	READ Protection level	SDK Area Protection level	Boot From Main Flash(CPU)						Debug/ excuted From RAM/ excuted From System memory			DMA		
			User execution (From Non SDK Area)			User execution (From SDK Area)								
			Read	Write	Erase	Read	Write	Erase	Read	Write	Erase	Read	Write	Erase
Non SDK Area	0	Disable	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes	Yes	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
SDK Area		Disable	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		Enable	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No
Non SDK Area	1	Disable	Yes	Yes	Yes	N/A	N/A	N/A	No	No	No	No	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
SDK Area		Disable	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
		Enable	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No
System memory	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No
Option bytes area	x	Disable	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes	No	No	No
		Enable	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
Factory bytes	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No
UID	x	Disable	Yes	No	No	N/A	N/A	N/A	Yes	No	No	No	No	No
		Enable	Yes	No	No	Yes	No	No	Yes	No	No	No	No	No

Note:

1. 任何区域发出的 mass erase 指令都会 erase 掉 SDK 区。
2. 任何对 level 1 修改为 level 0，都会触发硬件对 main flash 的 mass erase。
3. N/A 的含义是当 SDK Area disable 掉，由于不存在 SDK Area，上表 SDK Area 不存在读出程序的情况，也不存在从其他区域读出程序对 SDK Area 访问的情况。
4. 对于从 SRAM 或者 system memory 执行程序包括两种情况：一个是 Boot from，另一个是从别的存储器 boot，程序跳转到 SRAM 或者 system memory。

4.5.3. 闪存写保护

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

当被 WRP 的区域被激活，则不允许进行 erase 或者 program 操作。相应的，即使只有一个区域被设定为写保护，则 mass erase 功能不起作用。

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

注：写保护仅对 main flash 起作用，读对 system memory 不起作用。

4.5.4. 选项字节写保护

缺省情况下，Option bytes 是可读，并进行写保护的。为获得对 option bytes 的 erase 或者 program 访问，需要向 OPTKEYR 寄存器写入正确的序列。

4.6. 闪存中断

表 3-10 闪存中断请求

中断事件	事件标志	时间标志/中断清除方法	控制位使能
End of operation	EOP	Write EOP=1	EOPIE
Write protection	WRPERR	Write WRPERR=1	ERRIE

注：以下事件没有单独的中断标识，但会产生 Hard fault：

1） Unlock flash memory 的 FLASH_CR 寄存器的序列错误
2） Unlock flash option bytes 的写操作序列错误
3） FLASH program 操作未进行 32 位数据的对齐
4） Flash erase（含 page erase、sector erase 和 mass erase）操作未进行 32 位数据对齐
5） 对 option byte 寄存器的写操作未进行 32 位数据的对齐

4.7. 闪存寄存器描述

4.7.1. FLASH 访问控制寄存器 (FLASH_ACR)

Address offset: 0x00

Reset value: 0x0000 0700

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

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	-
1:0	LATENCY[1:0]	RW	0	Flash 读操作对应的等待状态： 00：flash 读操作没有等待状态（SYSCLK<=24 MHz） 01：flash 读操作有 1 个等待状态，即每次读 flash 需要两个系统时钟周期（24MHz <SYSCLK<=48MHz） 10：flash 读操作有 2 个等待状态，即每次读 flash 需要四个系统时钟周期（48MHz <SYSCLK<=72MHz） 11：Reserved

4.7.2. FLASH 密钥寄存器 (FLASH_KEYR)

Address offset: 0x08

Reset value: 0x0000 0000

所有寄存器位是 write-only，读出返回 0。

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

Bit	Name	R/W	Reset Value	Function
31：0	KEY[31:0]	W	0x0000 0000	下面的值必须被连续的写入，才能 unlock FLASH_CR 寄存器，并使能了 flash 的 program/erase 操作 KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.7.3. FLASH 选项密钥寄存器 (FLASH_OPTKEYR)

Address offset: 0x0C

Reset value: 0x0000 0000

所有寄存器位是 write-only，读出返回 0。

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

Bit	Name	R/W	Reset Value	Function
31：0	OPTKEY[31:0]	W	0x0000 0000	下面的值必须被连续的写入，才能 unlock flash 的 option 寄存器，并使能了 option byte 的 program/erase 操作 KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F

4.7.4. FLASH 状态寄存器 (FLASH_SR)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BSY
															R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTV ERR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	WRP ERR	Res	Res	Res	EOP
RC_W1											RC_W1				RC_W1

Bit	Name	R/W	Reset Value	Function
31：17	Reserved	-		
16	BSY	R	0	Busy 位

				该位表示 flash 的操作正在进行。该位在 flash 操作的开始被硬件置位，当操作完成或者错误产生时由硬件清零。
15	OPTVERR	RC_W1	0	Option and trimming bits loading validity error 当 option 和 trimming bit 及其反码不匹配时，硬件置位该位。装载不匹配的 option bytes，被强制成安全值。 软件写 1，清零。
14:5	reserved	-		
4	WRPERR	RC_W1	0	Write protection error 当要被 program/erase 的地址处于被写保护的 flash 区域时（WRP），硬件置位该位。 软件写 1，清零该位。
3：1	Reserved			
0	EOP	RC_W1	0	当 flash 的 program/erase 操作成功完成，硬件置位。该位仅当如果 FLASH_CR 寄存器的 EOPIE 位使能才会被置位。 软件写 1，清零该位。

4.7.5. FLASH 控制寄存器(FLASH_CR)

Address offset: 0x14

Reset value: 0xC000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPTLOCK	Res	Res	OBL_LAUNCH	Res	ERRIE	EOPIE	Res	Res	Res	Res	PGSTRT	Res	OPTSTRT	Res
RS	RS			RC_W1		RW	RW					RW		RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	SER	Res	Res	Res	Res	Res	Res	Res	Res	MER	PER	PG
				RW									RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	FLASH_CR Lock 位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器被 Lock 住。当成功给出 unlock 时序后，该位被硬件清零，unlock 了 FLASH_CR 寄存器。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的 unlock 时序给出，该位仍然保持置位状态，直到下一次系统复位。
30	OPTLOCK	RS	1	Option bytes Lock 位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器中与 option bytes 有关的位被 Lock 住。当成功给出 unlock 时序后，该位被硬件清零，unlock 了 FLASH_CR 寄存器。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的 unlock 时序给出，该位仍然保持置位状态，直到下一次系统复位。
29：28	Reserved	-		
27	OBL_LAUNCH	RC_W1	0	Force the option bytes loading。

Bit	Name	R/W	Reset Value	Function
				当置位时，该位强制系统进行 option bytes 的重装载。该位仅当 option byte 装载被完成后被硬件清零。如果 OPTLOCK 位被置位，该位不能被写. 0：Option byte loading 完成 1：产生 Option byte loading 请求，系统产生复位，进行 option byte 的重装载。
25	ERRIE	RW	0	Error interrupt enable 位，当 FLASH_SR 寄存器的 WRPERR 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
24	EOPIE	RW	0	End of operation interrupt enable 当 FLASH_SR 寄存器的 EOP 位被置位，该位使能中断的产生。 0：EOP 中断关闭 1：EOP 中断使能
23：18	Reserved	RW		
19	PGSTRT	RW	0	Flash main memory 的 program 操作的启动位。 该位启动了 Flash main memory 的 program 操作，软件置位，在 FLASH_SR 寄存器的 BSY 位被清零后，硬件清零该位。
18	Reserved	-		
17	OPTSTRT	RW	0	Flash option bytes 修改的启动位 该位启动了对 option bytes 的修改。软件置位，在 FLASH_SR 寄存器的 BSY 位被清零后，硬件清零该位。 注意：当对 flash option bytes 进行修改时，硬件自动把整个 128Bytes 的 page 进行 erase 操作，再进行 program 操作，其中也包括自动进行补码的写入。
16:12	Reserved	-		
11	SER	RW	0	4kByte 的 Sector erase 操作 0：未选择 flash 的 sector erase 操作 1：选择 flash 的 sector erases 操作 注： ■ Sector erase 不会对 flash information memory 起作用。 ■ Sector erase 对设定为 WRP 的区域不起作用。
10：3	Reserved			
2	MER	RW	0	Mass erase 操作 0：未选择 flash 的 mass erase 操作 1：选择 flash 的 mass erases 操作 注： Mass erase 不会对 flash information memory 起作用。当有 WRP 设定时，Mass erase 不起作用
1	PER	RW	0	Page erase 操作 0：未选择 flash 的 page erase 操作 1：选择 flash 的 page erase 操作
0	PG	RW	0	Program 操作 0：未选择 flash 的 program 操作

Bit	Name	R/W	Reset Value	Function
				1：选择 flash 的 program 操作

4.7.6. FLASH 选项寄存器 (FLASH_OPTR)

Address offset: 0x20

Reset value: 0x0000 xxxx。在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-		
15	IWDG_STOP	RW	1	设置 iwdg 在 stop 模式下定时器运行状态 0：freze 定时器 1：正常运行
14	nBOOT1	RW	1	与 BOOT PIN 一起，选择芯片启动模式
13	NRST_MODE	RW	0	0：仅复位输入 1：GPIO：GPIO 功能
12	WWDG_SW	RW	1	0：硬件 watchdog 1：软件 watchdog
11	IWDG_SW	RW	1	0：硬件 watchdog 1：软件 watchdog
10:8	Reserved			
7：0	RDP	RW	0xAA	0xAA：level 0, read protection inactive 非 0xAA：level 1, read protection active

4.7.7. FLASH SDK 地址寄存器 (FLASH_SDKR)

Address offset: 0x24

Reset value: 0xxxx xxxx。在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			
15:13	BOR_LEV[2:0]	RW		000：BOR 上升阈值为 1.8V，下降阈值位 1.7V

Bit	Name	R/W	Reset Value	Function
				001：BOR 上升阈值为 2.0V，下降阈值位 1.9V 010：BOR 上升阈值为 2.2V，下降阈值位 2.1V 011：BOR 上升阈值为 2.4V，下降阈值位 2.3V 100：BOR 上升阈值为 2.6V，下降阈值位 2.5V 101：BOR 上升阈值为 2.8V，下降阈值位 2.7V 110：BOR 上升阈值为 3.0V，下降阈值位 2.9V 111：BOR 上升阈值为 3.2V，下降阈值位 3.1V
12：8	SDK_END[4:0]	RW		SDK area end addres，每一位对应的 STEP 为 4Kbytes
7:6	Reserved	-		
5	BOR_EN	RW		BOR enable 0：BOR 不使能 1：BOR 使能，BOR_LEV 起作用
4：0	SDK_STRT[4:0]	RW		SDK area start address，每一位对应的 STEP 为 4Kbytes

4.7.8. FLASH WRP 地址寄存器 (FLASH_WRPR)

Address offset: 0x2C

Reset value: 0x0000 XXXX

在上电复位（POR/BOR/OBL_LAUNCH）释放后,从 flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的 option bit。

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			
15	WRP	RW	1	0：sector 15，有写保护，不允许进行 program 和 erase 1：sector 15，无写保护
14	WRP	RW	1	0：sector 14，有写保护，不允许进行 program 和 erase 1：sector 14，无写保护
13	WRP	RW	1	0：sector 13，有写保护，不允许进行 program 和 erase 1：sector 13，无写保护
12	WRP	RW	1	0：sector 12，有写保护，不允许进行 program 和 erase 1：sector 12，无写保护
11	WRP	RW	1	0：sector 11，有写保护，不允许进行 program 和 erase 1：sector 11，无写保护
10	WRP	RW	1	0：sector 10，有写保护，不允许进行 program 和 erase 1：sector 10，无写保护
9	WRP	RW	1	0：sector 9，有写保护，不允许进行 program 和 erase 1：sector 9，无写保护
8	WRP	RW	1	0：sector 8，有写保护，不允许进行 program 和 erase 1：sector 8，有写保护
7	WRP	RW	1	0：sector 7，有写保护，不允许进行 program 和 erase

Bit	Name	R/W	Reset Value	Function
				1：sector 7，无写保护
6	WRP	RW	1	0：sector 6，有写保护，不允许进行 program 和 erase 1：sector 6，无写保护
5	WRP	RW	1	0：sector 5，有写保护，不允许进行 program 和 erase 1：sector 5，无写保护
4	WRP	RW	1	0：sector 4，有写保护，不允许进行 program 和 erase 1：sector 4，无写保护
3	WRP	RW	1	0：sector 3，有写保护，不允许进行 program 和 erase 1：sector 3，无写保护
2	WRP	RW	1	0：sector 2，有写保护，不允许进行 program 和 erase 1：sector 2，无写保护
1	WRP	RW	1	0：sector 1，有写保护，不允许进行 program 和 erase 1：sector 1，无写保护
0	WRP	RW	1	0：sector 0，有写保护，不允许进行 program 和 erase 1：sector 0，无写保护

4.7.9. FLASH 睡眠时间配置寄存器(FLASH_STCR)

Address offset: 0x90

Reset value: 0x0000 6400

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved			
15：8	SLEEP_TIME	RW	0x64	FLASH 睡眠时间计数(基于 HSI_10M 时钟的计数器) 当系统时钟选择 LSI 或者 LSE 时，为获得更优化的 Run 模式功耗，可选择使用该寄存器的功能（仅推荐在 LSI 或者 LSE 为系统时钟时，使用该功能）。 当使能该功能时，每半个系统时钟低电平周期内 Flash 处于 Sleep 状态的时间宽度为： tHSI_10M * SLEEP_TIME Note: tHSI_10M 为 HSI_10M 的周期；
7：1	Reserved			
0	SLEEP_EN	RW	0	FLASH Sleep enable 1：enable flash sleep 0：disable flash sleep

4.7.10. FLASH TS0 寄存器 (FLASH_TS0)

Address offset: 0x100

Reset value: 0x0000 00B4

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

Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	TS0							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：8	Reserved			
7：0	TS0	RW	0xB4	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3238 8MHz 校准值存放地址：0x1FFF 3260 16MHz 校准值存放地址：0x1FFF 3288 22.12MHz 校准值存放地址：0x1FFF 32B0 24MHz 校准值存放地址：0x1FFF 32D8

4.7.11. FLASH TS1 寄存器 (FLASH_TS1)

Address offset: 0x104

Reset value: 0x0000 01B0

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

Bit	Name	R/W	Reset Value	Function
31：9	Reserved			
8：0	TS1	RW	0x1B0	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3238 8MHz 校准值存放地址：0x1FFF 3260 16MHz 校准值存放地址：0x1FFF 3288 22.12MHz 校准值存放地址：0x1FFF 32B0 24MHz 校准值存放地址：0x1FFF 32D8

4.7.12. FLASH TS2P 寄存器 (FLASH_TS2P)

Address offset: 0x108

Reset value: 0x0000 00B4

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved			
7：0	TS2P	RW	0xB4	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3240 8MHz 校准值存放地址：0x1FFF 3268 16MHz 校准值存放地址：0x1FFF 3290 22.12MHz 校准值存放地址：0x1FFF 32B8 24MHz 校准值存放地址：0x1FFF 32E0

4.7.13. FLASH TPS3 寄存器 (FLASH_TPS3)

Address offset: 0x10C

Reset value: 0x0000 06C0

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

Bit	Name	R/W	Reset Value	Function
31：11	Reserved			
10：0	TPS3	RW	0x6C0	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3240 8MHz 校准值存放地址：0x1FFF 3268 16MHz 校准值存放地址：0x1FFF 3290 22.12MHz 校准值存放地址：0x1FFF 32B8 24MHz 校准值存放地址：0x1FFF 32E0

4.7.14. FLASH TS3 寄存器 (FLASH_TS3)

Address offset: 0x110

Reset value: 0x0000 00B4

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved			

Bit	Name	R/W	Reset Value	Function
7：0	TS3	RW	0xB4	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3238 8MHz 校准值存放地址：0x1FFF 3260 16MHz 校准值存放地址：0x1FFF 3288 22.12MHz 校准值存放地址：0x1FFF 32B0 24MHz 校准值存放地址：0x1FFF 32D8

4.7.15. FLASH 页擦写（PAGE ERASE）TPE register (FLASH_PERTPE)

Address offset: 0x114

Reset value: 0x0001 4820

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

Bit	Name	R/W	Reset Value	Function
31：17	Reserved			
16：0	PERTPE	RW	0x14820	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3248 8MHz 校准值存放地址：0x1FFF 3270 16MHz 校准值存放地址：0x1FFF 3298 22.12MHz 校准值存放地址：0x1FFF 32C0 24MHz 校准值存放地址：0x1FFF 32E8

4.7.16. FLASH SECTOR/MASS ERASE TPE 寄存器 (FLASH_SMERTPE)

Address offset: 0x118

Reset value: 0x0001 4820

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

Bit	Name	R/W	Reset Value	Function
31：17	Reserved			
16：0	SMERTPE	RW	0x14820	软件通过读出存放在 information 区相应地址的数据，写入 对应寄存器，以实现对应 HSI 频率所需的擦写时间的配

Bit	Name	R/W	Reset Value	Function
				置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3250 8MHz 校准值存放地址：0x1FFF 3278 16MHz 校准值存放地址：0x1FFF 32A0 22.12MHz 校准值存放地址：0x1FFF 32C8 24MHz 校准值存放地址：0x1FFF 32F0

4.7.17. FLASH PROGRAM TPE register (FLASH_PRGTPE)

Address offset: 0x11C

Reset value: 0x0000 5DC0

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			
15：0	PRGTPE	RW	0x5DC0	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3258 8MHz 校准值存放地址：0x1FFF 3280 16MHz 校准值存放地址：0x1FFF 32A8 22.12MHz 校准值存放地址：0x1FFF 32D0 24MHz 校准值存放地址：0x1FFF 32F8

4.7.18. FLASH PRE-PROGRAM TPE 寄存器 (FLASH_PRETPE)

Address offset: 0x120

Reset value: 0x0000 12C0

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

Bit	Name	R/W	Reset Value	Function
31：14	Reserved			
13：0	PRETPE	RW	0x12C0	软件通过读出存放在 information 区相应地址的数据，写入对应寄存器，以实现对应 HSI 频率所需的擦写时间的配置。 保存在 Flash 的如下地址内： 4MHz 校准值存放地址：0x1FFF 3258 8MHz 校准值存放地址：0x1FFF 3280

				16MHz 校准值存放地址：0x1FFF 32A8 22.12MHz 校准值存放地址：0x1FFF 32D0 24MHz 校准值存放地址：0x1FFF 32F8

4.7.19. FLASH 寄存器映像

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	FLASH_ACR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LA-
	Reset value																																0
0x08	FLASH_KEYR	KEY[31:16]																KEY[15:0]															
	Reset value	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_OPTKEYR	OPTKEY[31:16]																OPTKEY[15:0]															
	Reset value	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	FLASH_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY	OPTVER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRPER	Res.	Res.	Res.	EOP
	Reset value																0	0											0				0
0x14	FLASH_CR	LOCK	OPTLOCK	Res.	Res.	OBL_LAUNC	Res.	ERRIE	EOPIE	Res.	Res.	Res.	Res.	PGSTRT	Res.	OPTSTRT	Res.	Res.	Res.	Res.	Res.	SER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MER	PER	PG
	Reset value	0	0			0								0		0						0									0	0	0
0x20	FLASH_OPTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	nBOOT1	NRST_MODE	WWDG_SW	IWDG_SW	BOR_LEV[2:0]			BOR_EN	RDP[7:0]							
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x24	FLASH_SDKR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SA_END[4:0]					Res.	Res.	Res.	SA_STRT[4:0]				
	Reset value																				X	X	X	X	X				X	X	X	X	X
0x2C	FLASH_WRPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP[15:0]															
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x90	FLASH_STCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_TIME[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SLEEP_E
	Reset value																	0	1	1	0	0	1	0	0								0
0x100	FLASH_TS0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS0[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x104	FLASH_TS1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS1[8:0]								
	Reset value																								1	1	0	1	1	0	0	0	0

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
0x108	FLASH_TS2P	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS2P[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x10C	FLASH_TPS3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLASH_TPS3[10:0]										
	Reset value																						1	1	0	1	1	0	0	0	0	0	0
0x110	FLASH_TS3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS3[7:0]							
	Reset value																									1	0	1	1	0	1	0	0
0x114	FLASH_PERTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERTPE	WRP[15:0]															
	Reset value																0	1	1	1	0	1	0	1	0	0	1	1	0	0	0	0	0
0x118	FLASH_SMERTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMERTPE[16:0]																
	Reset value																0	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	0
0x11C	FLASH_PRGTPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRGTPE[15:0]															
	Reset value																	1	0	0	0	1	1	0	0	1	0	1	0	0	0	0	0
0x120	FLASH_PRETPE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRETPE[13:0]													
	Reset value																			0	1	0	0	1	0	1	1	0	0	0	0	0	0

5. 电源控制

5.1. 电源

5.1.1. 电源框图

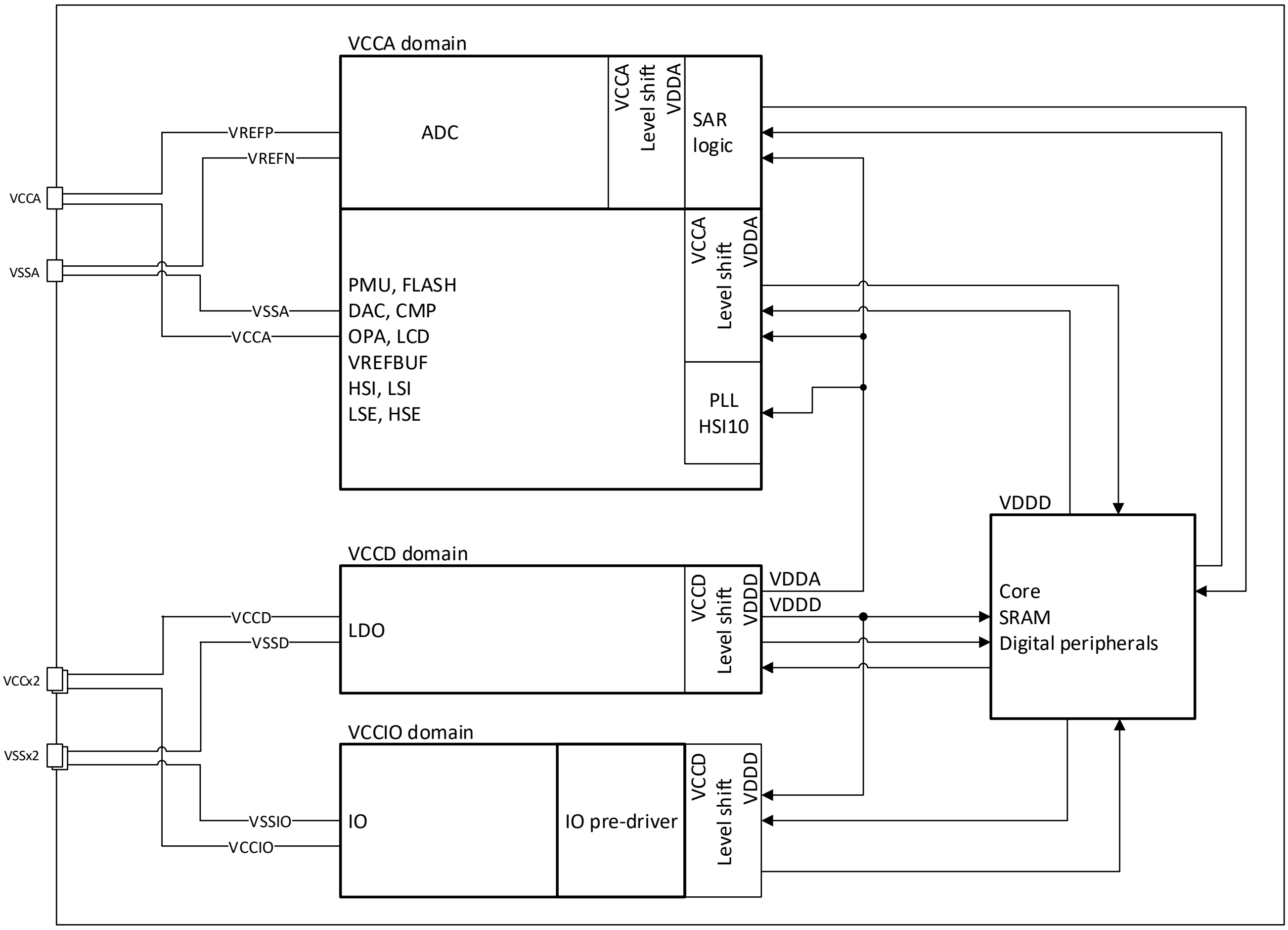


图 5-1 电源框图

表 5-1 电源框图

编号	电源	电源值	描述
1	VCC	1.7 V~5.5 V	通过电源管脚为芯片提供电源，其供电模块为：部分模拟 IP 和 IO 电路。
2	VCCA	1.7 V~5.5 V	给大部分模拟模块供电，来自于 VCCA PAD。
3	VCCIO	1.7 V~5.5 V	给 IO 供电，来自于 VCC PAD
4	VDDx (VDDD/VDDA)	1.2 V/1.0 V/0.9 V/0.8 V± 10%	来自于 VR 的输出，为芯片内部主要逻辑电路、SRAM 供电。当 MR 供电时，输出 1.2 V。当进入 stop 模式时，根据软件配置，可以由 MR 或者 LPR 供电，并根据软件配置决定 LPR 输出是 1.2 V/1.0 V/0.9 V/0.8 V。

5.2. 电压调节器

芯片设计两个电压调节器:

- MR（Main regulator）在芯片正常运行状态时保持工作。
- LPR（low power regulator）在 stop 模式下，提供更低功耗的选择。

VDDx 的电源根据芯片的工作模式，来自于 MR 或 LPR。

在芯片 run 模式，MR 保持工作，输出 1.2v 电压，LPR 关闭。

在 stop 模式，可由软件决定从 MR 或 LPR 供电。同样，由软件决定进入 stop 后，LPR 供电情况下的 VDDx 是 1.2 V/1.0 V/0.9 V/0.8 V。

5.3. 动态电压值管理

动态电压值管理指的是对 VR 的输出 VDDx 电压进行调节，使芯片可以根据应用要求运行在不同的电压下，从而获得相应的性能及功耗。

本项目定义两种电压范围:

- **Range 1：高性能 Range**

MR 的输出为典型值 1.2 V（VDDx），系统时钟频率可以运行在最快的 72 MHz 下

- **Range 2：低功耗 Range**

只有当芯片处于 stop 模式时，才允许设定进入该 range，且该 range 只针对 LPR 起作用。

默认情况，LPR 的输出为典型值 1.2 V（VDDx），当置位寄存器 VOS 位时，芯片进入 stop 模式时，MR 切换成 LPR 供电（如果软件选择 stop 模式由 LPR 供电），且 LPR 被切换成典型值 1.0 V（VDDx）。此时，部分处于工作状态的逻辑电路（LPTIMER）可以运行在 LSI 下。为了在 stop 模式获得更低功耗，可以将 LPR 换成更低值 0.9 V/0.8 V（VDDx）。此时，建议唤醒方式采用 IO。

当芯片退出 stop 模式时，芯片恢复 MR 供电，VOS 位也由硬件清零。下次进入 stop 模式，如果想要获得更低的功耗，仍要软件置位 VOS 位，使芯片进入 stop 模式后的 LPR 供电为 1.0 V，及更低的 0.9 V/0.8 V。

5.4. 电源监控

5.4.1. 上电复位 (POR)/下电复位 (PDR)/欠压复位 (BOR)

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

除了 POR/PDR 外，还实现了 BOR（brown out reset）。BOR 仅可以通过 option byte，进行使能和关闭操作。

当 BOR 被打开时，BOR 的阈值可以通过 Option byte 进行选择。

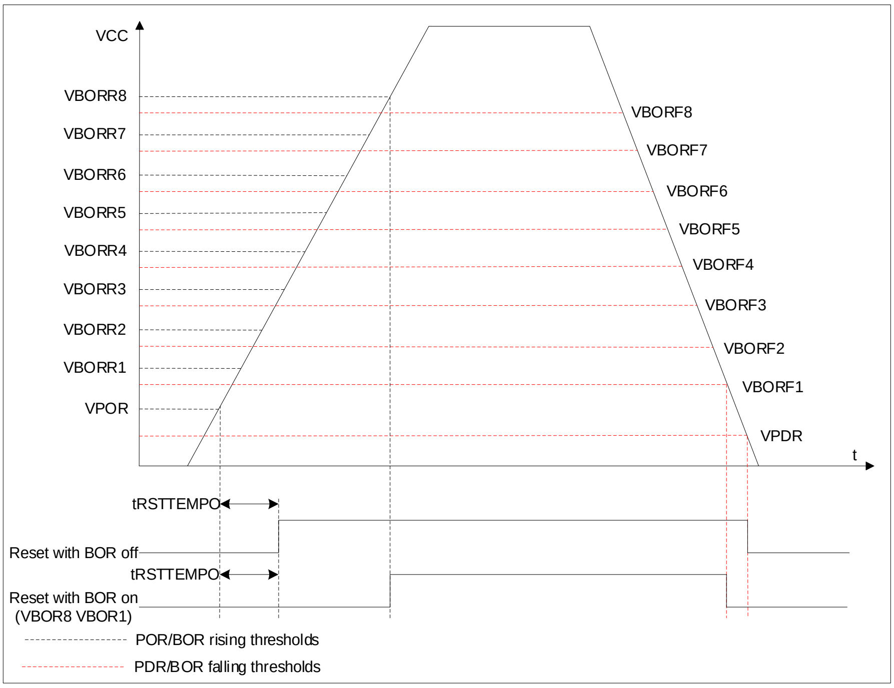


图 5-2 POR/PDR/BOR 阈值

5.4.2. 可编程电压检测器 (PVD)

该模块可以用来检测 VCC 电源（也可以检测 PB7 引脚的电压），检测点可通过寄存器进行配置。当 VCC 高于或者低于 PVD 的检测点时，产生相应的标识。

该事件内部连接到 EXTI 的 line 16，取决于 EXTI line 16 上升/下降沿配置，当 VCC 上升超过 PVD 的检测点，或者 VCC 降低到 PVD 的检测点以下，产生中断，在中断服务程序中用户可以进行紧急的 shutdown 任务。

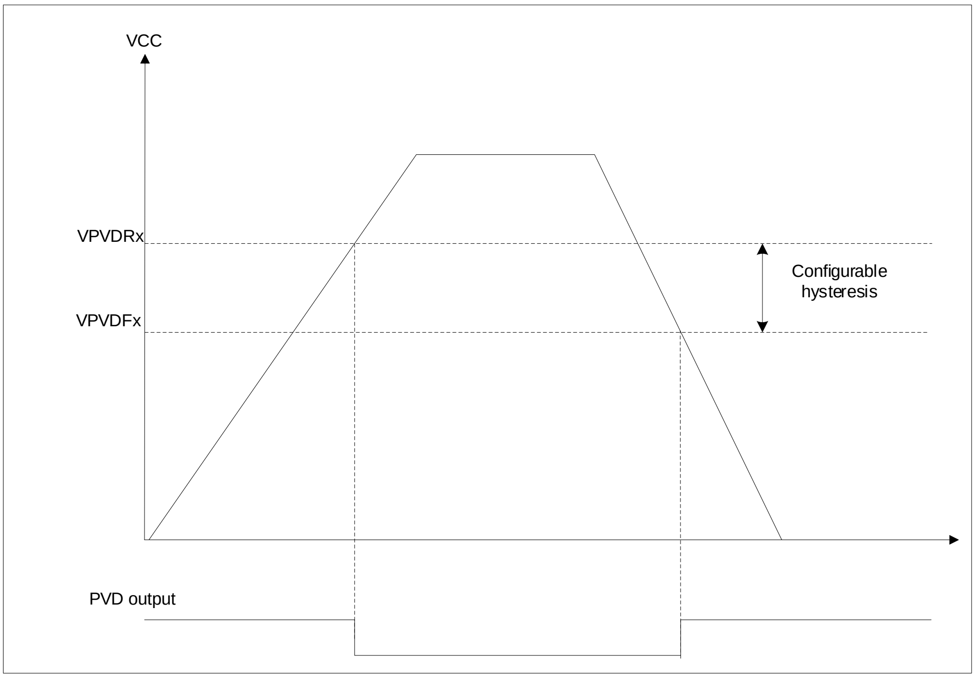


图 5-3 PVD 阈值

6. 低功耗控制

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

6.1. 低功耗模式

6.1.1. 低功耗模式介绍

芯片在正常的 run 模式之外，有 2 个低功耗模式：

- **Sleep mode**：CPU Core 时钟关闭（NVIC，SysTick 等工作），外设可以配置为保持工作。（建议只使能必须工作的模块，在模块工作结束后关闭该模块）。
- **Stop mode**：该模式下 SRAM 和寄存器的内容保持，HSI 和 HSE 关闭，VDD 域下大部分模块的时钟都被停掉。

在 stop 模式，LSI 和 LSE 可以保持工作，RTC、LPTIMER 等可以保持工作。具体该模式下各模块的工作情况，参照表 6-2。

在 stop 模式下，对应的 VR 状态可由软件控制，设成 MR 或者 LPR 供电。当 LPR 供电时，芯片功耗大大降低，但唤醒时间较长；当保持 MR 供电的情况，芯片功耗较大，但具备较快的唤醒能力。

此外，正常 run 模式下可以通过下述方法降低功耗：

1. 降低系统时钟频率
2. 对于不使用的外设，关掉外设时钟（系统时钟和模块时钟）

综上分析，本项目的低功耗模式转换图如下所述。

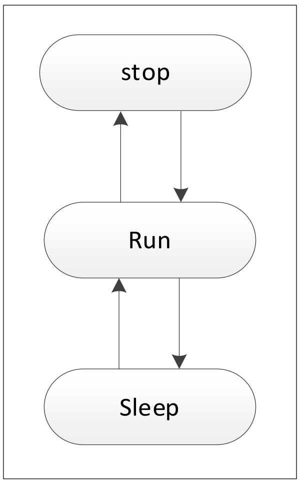


图 6-1 电源模式

6.1.2. 低功耗模式开关

表 6-1 低功耗模式开关

模式	进入	唤醒源	唤醒时钟	对时钟的影响	Voltage regulator	
					MR	LPR
Sleep (sleep-now or sleep-on-exit)	WFI or Return from ISR	任何中断	与进入 sleep 之前一样	CPU core 时钟停止，对其他时钟和时钟源没影响。	开 (1)	关
	WFE	唤醒事件				

模式	进入	唤醒源	唤醒时钟	对时钟的影响	Voltage regulator	
					MR	LPR
Stop	SLEEPDEEP bit 注1. WFI or 注2. Return from ISR or 注3. WFE Note: 进入 stop 时系统时钟切换为 HSI	任何配置为唤醒的 EXTI Line（EXTI 寄存器配置）、IWDG 复位、NRST	HSISYS (HSI 保持进入 stop 前的频率配置，不分频)	HSI、HSE、PLL 关闭； LSI、LSE 可选择开或者关； LPTIMER、RTC、IWDG：由软件配置是否工作； 低功耗唤醒和部分 RCC 等模块保持工作； 其余模块的时钟关闭。	软件配置开关	软件配置开关，如果开，输出电压 1.2/1.0v/0.9v/0.8v 可配置

注 1：软件要配置 VR 的状态为 MR 模式，才能进入 sleep 模式。

6.1.3. 各工作模式下的功能

表 6-2 各工作模式下的功能 (1)

Peripheral	Run	Sleep	Stop	
			VR@LPR or VR@MR	Wakeup ability
CPU	Y	-	-	-
Flash memory	Y	Y	- (2)	-
SRAM	Y	O (3)	- (4)	-
Brown-out reset (BOR)	Y	Y	O	O
PVD	O	O	O	O
DMA	O	O	-	-
HSI	O	O	-	-
HSE	O	O	-	-
PLL	O	O	-	-
LSI	O	O	O	-
LSE	O	O	O	-
HSE Clock Security System (CSS)	O	O	-	-
LSE Clock Security System (CSS)	O	O	O	O
RTC	O	O	O	O
USART1/USART2/USART3/USART4	O	O	-	-
I2C1/I2C2	O	O	-	-
SPI1/SPI2	O	O	-	-
ADC	O	O	-	-
COMP1/COMP2	O	O	O	O
OPA1/OPA2	O	O	-	-
Temperature sensor	O	O	-	-
LCD	O	O	O	-
Timers(TIM1/TIM2/TIM3/TIM6/TIM7/TIM14/TIM15/TIM16/TIM17)	O	O	-	-
LPTIM	O	O	O	O
IWDG	O	O	O	O
WWDG	O	O	-	-
SysTick timer	O	O	-	-
CRC	O	O	-	-
GPIOs	O	O	O	O

1. Y = Yes (使能)；O = Optional (默认关闭，可以软件使能)；- = Not available
2. Flash 不下电，但无时钟提供，进入最低功耗状态。
3. SRAM 的时钟可以被开或者关。
4. SRAM 不下电，但无时钟提供，进入最低功耗状态。
5. 进入 stop 模式之前，如果使能了 LSE CSS，则当 LSE CSS 出现问题时，会唤醒系统，并进入 NMI 中断。

6.2. Sleep mode

6.2.1. 进入 sleep mode

通过执行 WFI(wait for interrupt)或者 WFE(wait for event)指令，进入 sleep 模式。取决于 Cortex M0+的系统控制寄存器的 SLEEPONEXIT 位，有两种可选的进入 sleep 模式的机制。

- Sleep-now:如果 SLEEPONEXIT 位是 0，则执行 WFI 或者 WFE 后，立即进入 sleep 模式。
- Sleep-on-exit:如果 SLEEPONEXIT 位是 1，则当退出低优先级中断 ISR 时，进入 sleep 模式。

在 sleep 模式，所有的 IO pin 与 run 模式保持相同的状态。

6.2.2. 退出 sleep mode

如果用 WFI 进入 sleep 模式，被 NVIC 获得的任何外设中断可以把芯片从 sleep 模式唤醒。

如果用 WFE 进入 sleep 模式，当一个事件发生时，芯片退出 sleep 模式。Wakeup 事件可以通过以下方式产生：

- 在外设控制寄存器使能中断，而不是在 NVIC，并使能 Cortex M0+的 SEVONPEND 位。当芯片从 WFE 唤醒后继续执行时，外设中断 pending 位和外设 NVIC IRQ 通道 pending 位（在 NVIC 的中断清除 pending 寄存器）必须被清零。
- 或者，配置外部或者内部 EXTI line 为事件模式。当 CPU 从 WFE 唤醒后继续执行时，不必清除外设中断 pending 位，或者对应到事件 Line 的 NVIC IRQ 通道 pending 位没有被置位。

该模式具有最短的 wakeup 时间，并且没有在中断进入和退出浪费时间。

表 6-3 Sleep-now

Sleep-now mode	Description
Mode entry	WFI 或者 WFE，并且： - SLEEPDEEP = 0 并且 - SLEEPONEXIT = 0
Mode exit	如果通过 WFI 进入的 sleep 模式，则退出方式是：中断。 如果通过 WFE 进入的 sleep 模式，则退出方式是：wakeup 事件。
Wakeup latency	无

表 6-4 Sleep-on-exit

Sleep-on-exit	Description
Mode entry	WFI，并且： - SLEEPDEEP = 0 并且 - SLEEPONEXIT = 1
Mode exit	中断
Wakeup latency	无

6.3. Stop mode

Stop 模式是基于 Cortex-M0+的 deep sleep 以及对外设时钟的 gating，VR 可以被配置成 MR 或者 LPR 供电。在该模式下， HSI、HSE、PLL 被关闭，SRAM 和寄存器内容处于保持状态，LSI、LSE、LPTIMER、RTC、IWDG 可由软件配置是否工作，低功耗唤醒和部分 RCC 逻辑等保持工作，其余 VCORE 域的数字模块的时钟输入被关闭。

在 stop 模式下，所有的 IO pin 保持跟 Run 模式相同的状态。

6.3.1. 进入 stop mode

为了进一步降低 stop 模式的功耗，配置 PWR_CR.LPR=1 时，VR 可以进入 LPR 供电。

如果正在进行 flash 的擦写操作，则 stop 模式的进入会被延迟，直到存储器访问结束（由软件读 FLASH_SR 寄存器的 BSY 位判断当前是否已完成擦、写操作）。

如果 APB 总线上的操作正在进行，则 stop 模式的进入也会被延迟，直到 APB 访问结束（由软件控制）。

6.3.2. 退出 stop mode

当通过中断或者 wakeup 事件退出 stop 模式时，HSISYS 被选择作为系统时钟。

在 stop 模式，如果 VR 处于 LPR 状态，则从 stop 模式唤醒有额外的稳定延迟。

在 stop 模式，如果 VR 处于 MR 状态，电流消耗会大，但唤醒时间会被减少。

表 6-5 stop mode

Stop mode	Description
Mode entry	WFI(wait for interrupt) 或者 WFE（wait for event），并且: 1） 配置设定: - 通过 PWR_CR 的 LPR 位，选择 VR 工作在 MR 或者 LPR 下 - 通过 PWR_CR 的 VOS 位，选择 LPR 模式提供 1.2V、1.0V、0.9V、0.8V - 通过 PWR_CR 的 FLS_SLPTIME 配置 FLASH 的唤醒时间 2） 置位 Cortex M0+的 SLEEPDEEP 位 **Note:** 为了进入 stop 模式，所有 EXTI line 的 pending 位（EXTI_PR 寄存器）、所有外设的中断 pending 位、RTC alarm 标志位，必须被复位。否则，进入 stop 模式的流程将被忽略掉，程序继续执行。 如果应用需要在进入 stop 模式前关闭 HSE，系统时钟源必须首先切换到 HSI，然后清除 HSEON 位。 为使芯片功耗变化尽可能均衡，软件需要遵循逐步关闭的原则：逐步关闭各个模块的时钟，选择 HSI 作为系统时钟，关闭 HSE。 为缩短唤醒时间，在进入 stop 模式前，系统时钟应该配置为选择 HSI 高频时钟，RCC_CFGR 寄存器的 HPRE 设为 0，否则在唤醒后硬件切换时钟会消耗额外的时钟。
Mode exit	如果使用 WFI 进入 stop 模式: - 任何被配置成中断模式的 EXTI line（相应的 EXTI 中断向量必须被在 NVIC 中使能） 如果使用 WFE 进入 stop 模式: ● 任何被配置成事件模式的 EXTI line ● CPU SEVONPEND 位置位情况下的中断 pending 位
Wakeup latency	LPR to MR wakeup time + HSI wakeup time + flash wakeup time

6.4. 降低系统时钟频率

在 run 模式下，系统时钟的频率（SYSCLK, HCLK, PCLK）可以通过预分频寄存器配置分频去降低。这些预分频器也可以被用来在进入 sleep 模式前，降低外设的频率。

6.5. 外设时钟门控

在 run 模式，可以在任何时间停止单个外设和存储器的 AHB 时钟（HCLK）和 APB 时钟（PCLK），以降低功耗。

为了进一步降低在 sleep 模式的功耗，外设的时钟可以在执行 WFI 或者 WFE 指令之前被停掉。

6.6. 电源管理寄存器

该外设的寄存器可以通过 half-word 或者 word 访问。

6.6.1. 电源控制寄存器 1 (PWR_CR1)

Address offset: 0x00

Reset value: 0x0000 0000(reset by POR)

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

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19	HSION_CTRL	RW	0	从 Stop 模式唤醒时，HSI 打开时间控制。 0：等待 MR 稳定后，使能 HSI； 1：唤醒时立刻使能 HSI。
18:15	Reserved			
14	LPR	RW	0	Low power regulator 0：Main regulator 工作在 stop 模式 1：Low power regulator 工作在 stop 模式
13:12	FLS_SLPTIME[1:0]	RW	2'b00	Stop 模式唤醒时序中，在 HSI 稳定后，在 FLASH 操作前需要等待时间。 2'b00: 1us 2'b01: 2us 2'b10: 3us 2'b11: 0us 注：当该寄存器设置为 2'b11 时，表明唤醒后是从 SRAM 执行程序，而非 FLASH。并且程序保证在唤醒执行程序后不会在 3us 内访问 FLASH。
11	Reserved	-	-	
10:9	VOS[1:0]	RW	0	Voltage scaling range selection 00：进入 stop 模式后, VCORE=1.2V 01：进入 stop 模式后, VCORE=1.0V 10：进入 stop 模式后, VCORE=0.9V 11：进入 stop 模式后, VCORE=0.8V
8	DBP	RW	0	RTC 写保护禁止 在复位后，RTC 处于写保护状态以防意外写入。要访问 RTC 该位必须设置为 1。 0：禁止访问 RTC 1：可以访问 RTC
7:0	Reserved	-	-	Reserved

6.6.2. 电源控制寄存器 2 (PWR_CR2)

Address offset: 0x04

Reset value: 0x0000 0500(reset by POR)

注：该寄存器是与 PVD 功能相关寄存器。

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11：9	FLT_TIME[2:0]	RW	3'b010	数字滤波时间配置 110：滤波时间大约为 30.7ms（1024 个 LSI/LSE 时钟） 101：滤波时间大约为 3.8ms（128 个 LSI/LSE 时钟） 100：滤波时间大约为 1.92ms（64 个 LSI/LSE 时钟） 011：滤波时间大约为 480us（16 个 LSI/LSE 时钟） 010：滤波时间大约为 120us（4 个 LSI/LSE 时钟） 001：滤波时间大约为 60us（2 个 LSI/LSE 时钟） 000：滤波时间大约为 30us（1 个 LSI/LSE 时钟）
8	FLTEN	RW	1	数字滤波功能使能控制 0：禁止 1：使能
7	Reserved	-	-	
6：4	PVDT[2:0]	RW	000	电压上升沿检测阈值（下降沿检测阈值相应减小 0.1V）及 PVDIN 检测控制。 000： VPVD0 (around 1.8V) 001： VPVD1 (around 2.0V) 010： VPVD2 (around 2.2V) 011： VPVD3 (around 2.4V) 100： VPVD4 (around 2.6V) 101： VPVD5 (around 2. 8V) 110： VPVD6 (around 3.0V) 111： VPVD7 (around 3.2V)
3	Reserved	-	-	
2	SRCSEL	RW	0	PVD 检测电源选择。 0：VCC 1：检测 PB7 pin 如果该位置为 1，PB7 上的电压会在内部与 VREFINT 进行比较(包括上升和下降阈值)。这种情况下 PVDT 寄存器的设定无效。
1	Reserved	-	-	Reserved
0	PVDE	RW	0	电压检测使能位 0：电压检测不使能 1：电压检测使能 如果 SYSCFG_CFG2.PVD_LOCK=1,则 PVDE 写保护。只有当系统复位后，写保护才被复位。

6.6.3. 电源状态寄存器(PWR_SR)

Address offset: 0x14

Reset value: 0x0000 0000(reset by POR)

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	PVDO	R		PVD 检测结果输出。 0：被检测的 VCC 或者 PB7 超出 PVD 选择的比较阈值 1：被检测的 VCC 或者 PB7 低于 PVD 选择的比较阈值
10:0	Reserved	-	-	Reserved

6.6.4. PWR 寄存器映像

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_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSION_CTRL	Res.	Res.	Res.	Res.	LPR	FLS_SLP-TIME[1:0]		Res.	VOS[1:0]		DBP	Res.	Res.	Res.	Res.	Res.			
	Reset value													0					0	0	0	0	0	0	0				0	0	0	0	0
0x04	PWR_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLT_TIME[2:0]			FLTEN	Res.	PVDT[2:0]			Res.	SRCSEL	Res.	PVDE
	Reset value																					0	1	0	0	1	0	0	0		0		0
0x08	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x0C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x10	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	
0x14	PWR_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PVDO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																					0											

7. 复位

芯片内设计两种复位，分别是：电源复位和系统复位。

7.1. 复位源

7.1.1. 电源复位

电源复位把所有寄存器都复位掉，在以下几种情况下产生：

- 模拟电路产生的 POR/ BOR，实现对 VCC 的检测。当 VCC 电压上升到 trigger 值时释放复位；当 VCC 电压下降到某一 trigger 值时产生复位；
- 模拟电路产生的 PORI，实现对 VR 输出的检测。当 Vcore 电压上升到 trigger 值时释放复位；当 Vcore 电压下降到某一 trigger 值时产生复位；

7.1.2. 系统复位

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

当产生以下事件时，产生系统复位：

- NRST pin 的复位
- 窗口看门狗复位(WWDG)
- 独立看门狗复位(IWDG)
- Cortex-M0+SYSRESETREQ 软件复位
- option byte load 复位（OBL）
- 电源复位（POR、BOR）

7.1.3. NRST 管脚 (external reset)

通过 option byte(NRST_MODE 位)的装载，NRST pin 可以被配置成下述模式（具体配置参见 option byte 描述）：

- Reset input

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

在该配置模式下，GPIO 的 PF2 功能无效。

NRST pin 输入后，经过去毛刺电路（去毛刺可配置为禁止）后产生芯片的外部复位。

- GPIO

在该模式下，该 PIN 可以用作标准的 GPIO，即 PF2。Pin 上的 reset 功能无效。芯片复位只会由芯片内部产生，并且不能传递到 pin 上。

注：上电复位后，NRST pin 默认配置为复位输入模式。

7.1.4. 看门狗复位

参见 Independent watchdog 和 system windows watchdog。

7.1.5. 软件复位

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

7.1.6. Option byte loader 复位

软件通过配置 FLASH_CR.OBL_LAUNCH=1,产生 option byte load 复位, 从而启动 option byte 再次 load。

8. 时钟

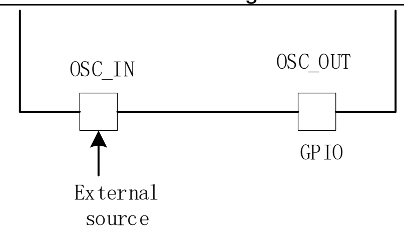
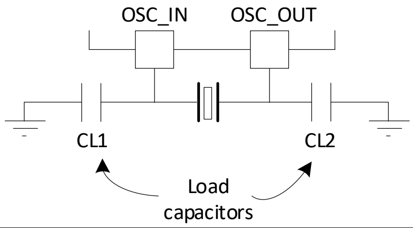
8.1. 时钟源

8.1.1. 外部高速时钟 HSE

外部高速时钟（HSE）来自两个来源：

- 外部 XTAL OSC+内部起振电路
- 通过 OSC_IN 结构输入的外部时钟（HSEBYP=1）

表 8-1 HSE 时钟来源

Clock source	Hardware configuration
外部时钟	
外接晶体	

外部高频 OSC. 频率范围 4~32 MHz.

HSE 时钟的稳定时间由寄存器配置。当 HSE 从 OFF 到 ON 时，需要等待稳定时间，稳定后，硬件置位 RCC_CR.HSERDY 寄存器。当 HSEBYP=1 时，稳定时间比非 bypass 模式减半。

HSE 时钟相关寄存器参考 RCC_ ECSCR。

8.1.2. 内部高速时钟 HSI

内部 RC 振荡器，基准频率可以为 4MHz，8MHz，16MHz，22.12MHz 和 24MHz。相较于 XTAL OSC，RC OSC 功耗低，稳定时间短，但精度低。HSI 模拟模块计数稳定时间，HSI 同时输出给系统数字模块

上电复位后， HSI 校准值需软件加载到 RCC_ ICSCR.HSITRIM 寄存器。系统复位时，该寄存器会随之复位。

从 stop 模式唤醒后，只能 HSI 作为系统时钟源。

8.1.3. 内部低速时钟 LSI

内部低频 32KHz 时钟。

8.1.4. HSI10M 时钟

该时钟作为低精度时钟，用作 nRST pin 的滤波计数，以及 FLASH 低速 run 时低功耗处理。

8.1.5. PLL

PLL 模块参考时钟是 HSI 或者 HSE， PLL 输入时钟频率范围要求为 12MHz~24MHz，不在此范围不能保证输出时钟频率和稳定性。

PLL 支持 2 倍频或者 3 倍频。

8.1.6. LSE 时钟

外部 32.76KHz OSC，用作低功耗时钟。

可以通过配置 LSEDRV 在稳定时间和功耗之间做平衡。LSE 稳定时间由寄存器配置。

与 HSE 来源类似，LSE 也有两个来源：

- 32.768K XTAL+内部起振电路
- 通过 OSC32_IN 输入的外部时钟（LSEBYP=1）

在 LSE bypass 情况下，稳定时间比非 bypass 模式减半。

8.2. 时钟树

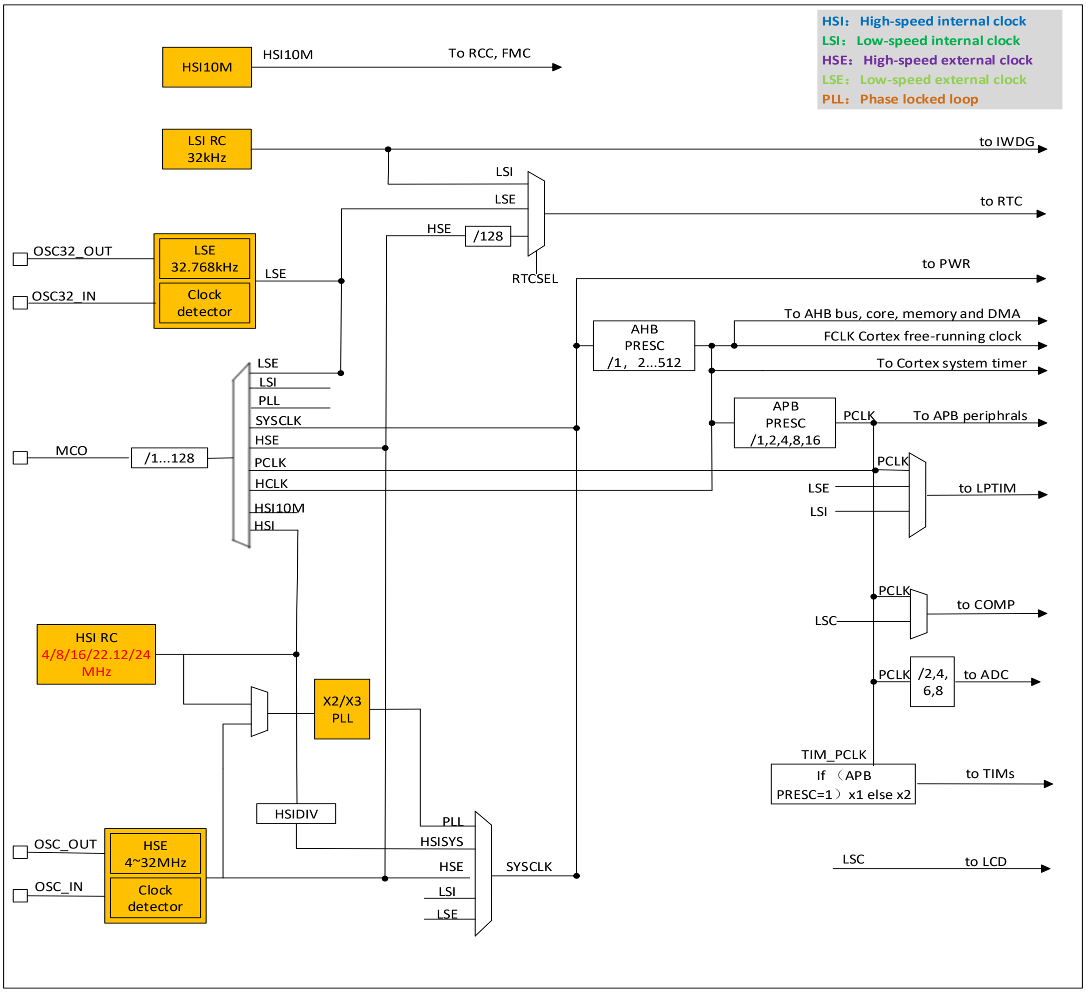


图 8-1 系统时钟结构图

8.3. 时钟安全系统（CSS）

时钟安全主要包括如下方面：

- 时钟配置和状态安全
- 时钟源 HSE 安全

- 时钟源 LSE 安全
- 基于 IWDG 的时钟安全
- 基于 Timer 的时钟安全
- 时钟频率校准（TEST 模块实现）

8.3.1. 时钟配置和状态安全

通过软件周期性地回读时钟配置和状态寄存器，获取系统当前时钟信息，并判断是否与预期一致。

8.3.1.1. 时钟源 HSE 监测

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

如果在 HSE 上发现时钟 failure，HSE 会被自动关闭，时钟 failure 事件被送给 TIM1（高级 timer）和 TIM15/TIM16/TIM17（通用 timer）的刹车输入端，并产生中断通知软件该 failure（Clock Security System Interrupt CSSI），进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex-M0+的 NMI（Non-maskable interrupt）exception 向量。

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

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

8.3.1.2. 时钟源 LSE 监测

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

如果在 LSE 上发现时钟 failure，LSE 会被自动关闭，时钟 failure 事件被送给 TIM1（高级 timer）和 TIM15/TIM16/TIM17（通用 timer）的刹车输入端，并产生中断通知软件该 failure（Clock Security System Interrupt CSSI），进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex-M0+的 NMI（Non-maskable interrupt）exception 向量。

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

如果 LSE 被用作系统时钟，时钟 Failure 将导致系统时钟自动切换到 LSI，同时关闭 LSE。同时，如果 LPTIM 和 RTC 计数时钟选择 LSE，也会自动切换到 LSI。

8.4. 输出时钟能力

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

表 8-2 输出时钟选择

时钟源	MCO 可输出的时钟源
HSI	√
SYSCLK	√
HSE	√
LSI	√
PLL	√
LSE	√

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

8.5. 复位/时钟寄存器

该模块的寄存器可以用 word(32bit)、half-word（16bit）和 byte（8bit）访问。

8.5.1. 时钟控制寄存器（RCC_CR）

Address offset:0x00

Reset value:0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	PLLRDY	PLLON	Res	ADC_DIV		Res	CSS ON	HSE BYP	HSE RDY	HSE ON
						R	RW		RW			RS	RW	R	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	HSIDIV[2:0]			HSI RDY	Res	HSION	Res	Res	Res	Res	Res	Res	Res	Res
		RW			R		RW								

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	PLLRDY	R	0	PLL 时钟 ready 标志。 硬件置位，表明 PLL 时钟 locked。 0：PLL unlocked； 1：PLL locked；
24	PLLON	RW	0	PLL 使能。 当系统进入 stop 模式时，硬件会清零该位。当 PLL 时钟作为系统时钟时，该位不能被清零。 0：PLL OFF； 1：PLL ON；
23	Reserved	-		Reserved
22:21	ADC_DIV	RW	0	ADC 分频系数 00：2 分频 01：4 分频 10：6 分频 11：8 分频
20	Reserved	-	-	Reserved
19	CSSON	RS	0x0	HSE 时钟安全系统使能。 当该位为 1 时，如果 HSE OSC ready 则硬件会使能时钟检测模块；如果 HSE 检测失败，则关闭时钟检测模块。 0：时钟安全系统关闭（时钟检测关闭）；

Bit	Name	R/W	Reset Value	Function
				1：时钟安全系统开启（如果 HSE 时钟稳定则时钟检测开启，否则关闭时钟检测）
18	HSEBYP	RW	0	HSE 屏蔽晶振，选择管脚输入时钟。 该位只有当 HSEON=0 时才能写. 0：HSE 晶振不屏蔽，外部高速时钟选择外部晶振； 1：HSE 晶振屏蔽，外部高速时钟选择外部管脚输入时钟源；
17	HSERDY	R	0	HSE 晶振时钟 ready 标志。 该位由硬件置位表明 HSE 晶振稳定。 0：HSE 晶振没有 ready； 1：HSE 晶振 ready； 注：当 HSEON 清零后，HSERDY 在 6 个 HSE 时钟周期后清零。
16	HSEON	RW	0	HSE 晶振使能。 当系统进入 stop 模式时，硬件会清零该位，关闭 HSE 晶振。当 HSE 作为系统时钟源时，该位不能被置 0. 0：HSE 晶振 OFF； 1：HSE 晶振 ON；
15:14	Reserved	-	-	Reserved
13:11	HSIDIV	RW	0x0	HSI 产生 HSISYS 时钟时的分频系数。 000：1； 001：2； 010：4； 011：8； 100：16； 101：32； 110：64； 111：128；
10	HSIRDY	R	0	HSI 时钟 ready 标志。 硬件置位表明 HSI OSC 稳定。该位只有当 HSION=1 时才有效。 0：HSI OSC not ready； 1：HSI OSC ready；
9	Reserved	-		Reserved
8	HSION	RW	1	HSI 时钟使能。 硬件在进入 stop 模式时，会根据需要清零该寄存器停止 HSI. 0：HSI OSC OFF； 1：HSI OSC ON；
7:0	Reserved	-		Reserved

8.5.2. 内部时钟源校准寄存器（RCC_ICSCR）

Address offset:0x04

Reset value:0x00FF_1080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res		Res	LSI_TRIM[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24:16	LSI_TRIM	RW	9'h0FF	内部低速时钟频率调整，通过校准，内部低速时钟可以输出 32.768KHz。 校准值保存在 Flash 的如下地址： 32.768KHz 校准值地址：0x1FFF 3348
15:13	HSI_FS	RW	3'b000	HSI 频率选择: 000: 4MHz 001: 8MHz 010: 16MHz 011: 22.12Mhz 100: 24MHz >=101: 4MHz
12:0	HSI_TRIM	RW	13'h1080	时钟频率调整，更改该寄存器的数值可以调整 HSI 的输出频率。寄存器数值每增加 1 则 HSI 的输出频率增加约 0.2%，总调整范围为 4~24 MHz。 24MHz/22.12MHz/16MHz/8MHz/4MHz 对应的校准值保存在 Flash 的如下地址内： 24MHz 校准值地址：0x1FFF 3220 22.12MHz 校准值地址：0x1FFF 3218 16MHz 校准值地址：0x1FFF 3210 8MHz 校准值地址：0x1FFF 3208 4MHz 校准值地址：0x1FFF 3200

8.5.3. 时钟配置寄存器（RCC_CFGR）

Address offset:0x08

Reset value:0x0000 0000

当时钟源切换时，访问该寄存器有 1 或者 2 个时钟的等待周期。

当 APH 或者 AHB 分频值更新时，访问该寄存器可能有 0~15 个时钟的等待周期。

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

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30:28	MCOPRE[2:0]	RW	0	MCO（microcontroller clock output）分频系数。软件控制这些位，设置 MCO 输出的分频系数： 000：1 001：2 010：4 011：8 100：16 101：32 110：64

Bit	Name	R/W	Reset Value	Function
				111：128 在 MCO 输出使能前要配置该分频系数。
27:24	MCOSEL[3:0]	RW	0	MCO 选择 0000: no clock, MCO output disabled 0001: SYSCLK 0010: HSI10M 0011: HSI 0100: HSE 0101: PLL CLK 0110: LSI 0111: LSE 1000: HCLK 1001: PCLK Others: no clock 注：在时钟启动或者切换阶段可能会出现输出时钟不完整的情况。
23:15	Reserved	-	-	Reserved
14:12	PPRE[2:0]	RW	0	该位由软件控制。为了产生 PCLK 时钟，它设置 HCLK 的分频系数如下： 0xx：1 100：2 101：4 110：8 111：16
11:8	HPRE[3:0]	RW	0	AHB 时钟分频系数。 软件控制该位。为了产生 HCLK 时钟，它设置 SYSCLK 的分频系数如下： 0xxx: 1 1000: 2 1001: 4 1010: 8 1011: 16 1100: 64 1101: 128 1110: 256 1111: 512 为了保证系统正常工作，需要根据 VR 电源情况配置合适频率。 注：建议逐级切换分频系数。
7:6	Reserved	-	-	Reserved
5:3	SWS[2:0]	R	0	系统时钟切换状态位 这些位由硬件控制，表明当前哪个时钟源被用作系统时钟： 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE Others: Reserved
2:0	SW[2:0]	RW	0	系统时钟源选择位。 这些位由软件和硬件控制，用来选择系统时钟： 000: HSISYS 001: HSE 010: PLL CLK 011: LSI 100: LSE Others: Reserved

Bit	Name	R/W	Reset Value	Function
				硬件配置为 HSISYS 的情况包括： 1）系统从 stop 模式退出 2）软件配置 001(HSE)，出现 HSE failure（HSE 为系统时钟源）

8.5.4. PLL 配置寄存器（RCC_PLLCFGR）

Address offset:0x0C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	Reserved
3:2	PLLMUL[1:0]	RW	2'b0	PLL 倍频系数 00：x2 01：x3 11：reserved
1:0	PLLSRC[1:0]	RW	0	PLL 时钟源选择。 00: No clock 01: Reserved 10: HSI 11: HSE

8.5.5. 外部时钟源控制寄存器（RCC_ECSCR）

Address offset:0x10

Reset value: 0x0003 0003

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

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	Reserved
21:20	LSE_STARTUP	RW	0x0	LSE 晶振稳定时间选择。 LSEBYP=0： 00：4096 个 LSE 时钟周期； 01：2048 个 LSE 时钟周期； 10：8192 个 LSE 时钟周期； 11：不计稳定时间，直接输出； LSEBYP=1： 00：2048 个 LSE 时钟周期； 01：1024 个 LSE 时钟周期； 10：4096 个 LSE 时钟周期； 11：不计稳定时间，直接输出；
19:18	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
17:16	LSE_DRV	RW	0x3	LSE drive capability setting, default 11 00: reserved 01: Idd 315nA, gm 3.5uA/V 10: Idd 500nA, gm 7.5uA/V 11: Idd 630nA, gm 10uA/V
15:5	Reserved	RES	-	Reserved
4:3	HSE_STARTUP	RW	0x0	HSE 稳定时间选择。 HSEBYP=0: 00：4096 个 HSE 时钟； 01：2048 个 HSE 时钟； 10：8192 个 HSE 时钟； 11：不计稳定时间，直接输出； HSEBYP=1: 00：2048 个 HSE 时钟； 01：1024 个 HSE 时钟； 10：4096 个 HSE 时钟； 11：不计稳定时间，直接输出；
2	Reserved	-	-	Reserved
1:0	HSE_DRV	RW	0x3	HSE drive capability setting, default 11 00: reserved 01: gm 3.5mA/V 10: gm 7.5mA/V 11: gm 10mA/V

8.5.6. 时钟中断使能寄存器（RCC_CIER）

Address offset:0x18

Reset value:0x0000 0000

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

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

8.5.7. 时钟中断标志寄存器（RCC_CIFR）

Address offset:0x1C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSF	R		LSE 时钟安全系统（CSS）中断标志。 当硬件检测 LSE OSC 时钟失败时置位该寄存器。 0：LSE 时钟检测失败中断未产生； 1：LSE 时钟检测失败中断产生； 写 LSECSSC 寄存器 1 清零该位。
8	CSSF	R	0	HSE 时钟安全系统中断标识位。 当硬件检测 HSE OSC 时钟失败时置位该寄存器。 0：HSE 时钟检测失败中断未产生； 1：HSE 时钟检测失败中断产生； 写 CSSC 寄存器 1 清零该位。
7:6	Reserved	-	-	Reserved
5	PLLRDYF	R	0	PLL ready 中断标志。 当 PLL lock 并且 PLLRDYDIE=1 时，硬件置位该寄存器。 0：PLL lock 中断未产生； 1：PLL lock 中断产生； 写 PLLRDYC 寄存器 1 清零该位。
4	HSERDYF	R	0	HSE ready 中断标识位 当 HSE 稳定并且 HSERDYIE 使能，该位由硬件置位。软件通过置位 HSERDYC 位，清零该位。 0：无由 HSE 引起的时钟 ready 中断 1：有由 HSE 引起的时钟 ready 中断 写 HSERDYC 寄存器 1 清零该位
3	HSIRDYF	R	0	HSI 时钟 ready 中断标志。 HSI 时钟稳定并且 HSIRDYIE=1 时，硬件置位该寄存器。 0：HSI 时钟 ready 中断未产生； 1：HSI 时钟 ready 中断产生； 写 HSIRDYC 寄存器 1 清零该位。
2	Reserved	-	-	Reserved
1	LSERDYF	R	0	LSERDY 时钟 ready 中断标志。 LSE 时钟稳定并且 LSERDYDIE=1 时，硬件置位该寄存器。 0：LSERDY 时钟 ready 中断未产生； 1：LSERDY 时钟 ready 中断产生； 写 LSERDYC 寄存器 1 清零该位。
0	LSIRDYF	R	0	LSI ready 中断标识位

Bit	Name	R/W	Reset Value	Function
				当 LSI 稳定并且 LSIRDYIE 使能，该位由硬件置位。软件通过置位 LSIRDYC 位，清零该位。 0：无由 LSI 引起的时钟 ready 中断 1：有由 LSI 引起的时钟 ready 中断 写 LSIRDYC 寄存器 1 清零该位。

8.5.8. 时钟中断清除寄存器（RCC_CICR）

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	LSECSSC	CSSC	Res	Res	PLLRDYC	HSERDYC	HSIRDYC	Res	LSERDYC	LSIRDYC
						W	W			W	W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	Reserved
9	LSECSSC	W	0	LSE 时钟安全系统（CSS）中断标志清零。 0：No effect； 1：清零 LSECSSF 标志
8	CSSC	W	0	时钟安全中断清零位。 0：没影响。 1：清除 CSSF 标志位。
7:6	Reserved	-	-	Reserved
5	PLLRDYC	W	0	PLL ready 中断标志清零。 0：No effect； 1：清零 PLLRDYF 标志；
4	HSERDYC	W	0	HSE ready 标志清零。 0：没影响。 1：清除 HSERDYF 位。
3	HSIRDYC	W	0	HSI ready 标志清零。 0：没影响。 1：清除 HSIRDYF 位。
2	Reserved	-	-	Reserved
1	LSERDYC	W	0	LSE ready 中断标志清零。 0：No effect； 1：清零 LSERDYF 标志；
0	LSIRDYC	W	0	LSI ready 标志清零。 0：没影响。 1：清除 LSIRDYF 位。

8.5.9. I/O 接口复位寄存器（RCC_IOPRSTR）

Address offset:0x24

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	GPIOF RST	Res	Res	GPIO CRST	GPIOB RST	GPIOA RST
										RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	GPIOFRST	RW	0	I/O PortF 复位。 0：no effect； 1：PortF I/O 复位
4:3	Reserved	-	-	Reserved
2	GPIOCRST	RW	0	I/O PortC 复位。 0：no effect； 1：PortC I/O 复位
1	GPIOBRST	RW	0	I/O PortB 复位。 0：no effect； 1：PortB I/O 复位
0	GPIOARST	RW	0	I/O PortA 复位。 0：no effect； 1：PortA I/O 复位

8.5.10. AHB 外设复位寄存器（RCC_AHBRSTR）

Address offset:0x28

Reset value:0x0000 0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件将复位清零。

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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	DIVRST	RW	0	除法器模块复位。 0：no effect； 1：除法器模块复位；
23:13	Reserved			Reserved
12	CRCRST	RW	0	CRC 模块复位。 0：no effect； 1：CRC 模块复位；
11:9	Reserved	-	-	Reserved
8:1	Reserved	-	-	Reserved
0	DMARST	RW	0	DMA 复位。 0：no effect； 1：DMA 模块复位

8.5.11. APB 外设复位寄存器 1（RCC_APBRSTR1）

Address offset:0x2C

Reset value:0x0000 0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件将复位清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPTIMRST	OPARST	Res	PWRRST	CTCRST	Res	Res	Res	Res	I2C2RST	I2C1RST	Res	USART4RST	USART3RST	USART2RST	Res
RW	RW		RW	RW					RW	RW		RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	SPI2RST	Res	Res	WWDGRST	RTCAPBRST	Res	Res	Res	Res	TIM7RST	TIM6RST	Res	Res	TIM3RST	TIM2RST
	RW			RW	RW					RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMRST	RW	0	LP Timer 模块复位。 0：no effect； 1：该模块复位；
30	OPARST	RW	0	OPA 模块复位。 0：no effect； 1：该模块复位；
29	Reserved	-	-	Reserved
28	PWRRST	RW	0	Power 接口模块复位。 0：no effect； 1：该模块复位；
27	CTCRST	RW	0	CTC 模块复位。 0：no effect； 1：该模块复位；
26:23	Reserved	-	-	Reserved
22	I2C2RST	RW	0	I2C2 模块复位。 0：no effect； 1：该模块复位
21	I2C1RST	RW	0	I2C1 模块复位。 0：no effect； 1：该模块复位；
20	Reserved	-	-	Reserved
19	USART4RST	RW	0	USART4 模块复位。 0：no effect； 1：该模块复位
18	USART3RST	RW	0	USART3 模块复位。 0：no effect； 1：该模块复位
17	USART2RST	RW	0	USART2 模块复位。 0：no effect； 1：该模块复位

Bit	Name	R/W	Reset Value	Function
16:15	Reserved	-	-	Reserved
14	SPI2RST	RW	0	SPI2 模块复位。 0：no effect; 1：该模块复位
13:12	Reserved	-	-	Reserved
11	WWDGRST	RW	0	WWDG 模块复位。 0：no effect; 1：该模块复位
10	RTCAPBRST	RW	0	RTC 模块 APB 复位。 0：no effect; 1：该模块复位
9:6	Reserved	-	-	Reserved
5	TIM7RST	RW	0	TIM7 模块复位。 0：no effect; 1：该模块复位
4	TIM6RST	RW	0	TIM6 模块复位。 0：no effect; 1：该模块复位
3:2	Reserved	-	-	Reserved
1	TIM3RST	RW	0	TIM3 模块复位。 0：no effect; 1：该模块复位
0	TIM2RST	RW	0	TIM2 模块复位。 0：no effect; 1：该模块复位

8.5.12. APB 外设复位寄存器 2（RCC_APBRSTR2）

Address offset:0x30

Reset value:0x0000 0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件将复位清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	LCD RST	Res	COMP2 RST	COMP1 RST	Res	TIM17 RST	TIM16 RST	TIM15 RST
								RW		RW	RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14 RST	USART1 RST	Res	SPI1 RST	TIM1 RST	MCUDBG RST	ADC RST	Res	Res	Res	Res	Res	Res	Res	Res	SYSCFG RST
RW	RW		RW	RW	RW	RW									RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23	LCDRST	RW	0	LCD 模块复位。 0：no effect; 1：该模块复位
22	Reserved	-	-	Reserved
21	COMP2RST	RW	0	COMP2 模块复位。 0：no effect;

Bit	Name	R/W	Reset Value	Function
				1：该模块复位；
20	COMP1RST	RW	0	COMP1 模块复位。 0：no effect； 1：该模块复位；
19	Reserved	-	-	Reserved
18	TIM17RST	RW	0	TIM17 模块复位。 0：no effect； 1：该模块复位；
17	TIM16RST	RW	0	TIM16 模块复位。 0：no effect； 1：该模块复位；
16	TIM15RST	RW	0	TIM15 模块复位。 0：no effect； 1：该模块复位
15	TIM14RST	RW	0	TIM14 模块复位。 0：no effect； 1：该模块复位；
14	USART1RST	RW	0	USART1 模块复位。 0：no effect； 1：该模块复位；
13	Reserved	-	-	Reserved
12	SPI1RST	RW	0	SPI1 模块复位。 0：no effect； 1：该模块复位；
11	TIM1RST	RW	0	TIM1 模块复位。 0：no effect； 1：该模块复位；
10	MCUDBGRST	RW	0	MCU Debug 模块复位。 0：no effect； 1：该模块复位；
9	ADCRST	RW	0	ADC 模块复位。 0：no effect； 1：该模块复位；
8:1	Reserved	-	-	Reserved
0	SYSCFGRST	RWs	0	SYSCFG、COMP 模块复位。 0：no effect； 1：该模块复位；

8.5.13. I/O 接口时钟使能寄存器（RCC_IOPENR）

Address offset:0x34

Reset value:0x0000 0000

该寄存器由软件置位和清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	GPIOF EN	Res	Res	GPIOC EN	GPIOB EN	GPIOA EN

										RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	GPIOFEN	RW	0	I/O PortF 时钟使能。 0：时钟禁止； 1：时钟使能
4:3	Reserved	-	-	Reserved
2	GPIOCEN	RW	0	I/O PortC 时钟使能。 0：时钟禁止； 1：时钟使能
1	GPIOBEN	RW	0	I/O PortB 时钟使能。 0：时钟禁止； 1：时钟使能
0	GPIOAEN	RW	0	I/O PortA 时钟使能。 0：时钟禁止； 1：时钟使能

8.5.14. AHB 外设时钟使能寄存器（RCC_AHBENR）

Address offset:0x38

Reset value:0x0000 0300

该寄存器由软件置位和清零。

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

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	Reserved
24	DIVEN	RW	0	除法器模块时钟使能。 0：禁止 1：使能
23:13	Reserved	-	-	Reserved
12	CRCEN	RW	0	CRC 模块时钟使能。 0：禁止 1：使能
11:10	Reserved	-	-	Reserved
9	SRAMEN	RW	1	在 sleep 模式下，SRAM 的时钟使能控制 0：在 sleep 模式该模块时钟关闭 1：在 sleep 模式该模块时钟使能 注：该位仅影响 sleep 模式该模块的时钟使能，在 run 模式，该模块时钟不会关闭
8	FLASHEN	RW	1	在 sleep 模式下，FLASH 的时钟使能控制 0：在 sleep 模式该模块时钟关闭 1：在 sleep 模式该模块时钟使能 注：该位仅影响 sleep 模式该模块的时钟使能，在 run 模式，该模块时钟不会关闭

Bit	Name	R/W	Reset Value	Function
7:1	Reserved	-	-	Reserved
0	DMAEN	RW	0	DMA 模块时钟使能。 0：禁止 1：使能

8.5.15. APB 外设时钟使能寄存器 1（RCC_APBENR1）

Address offset:0x3C

Reset value:0x0000 0000

该寄存器由软件置位和清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPT IM EN	OPA EN	Re s	PW R EN	CT CE N	Res	Res	Res	Res	I2C 2EN	I2C1 EN	Res	USA RT4E N	USA RT3E N	USA RT2 EN	Res
RW	RW		RW	RW					RW	RW		RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	SPI2 EN	Re s	Res	WW DG EN	RTC APB EN	Res	Res	Res	Res	TIM7 EN	TIM 6EN	Res	Res	TIM3 EN	TIM2 EN
	RW			RW	RW		RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31	LPTIMEN	RW	0	LP Timer1 模块时钟使能。 0：禁止 1：使能
30	OPAEN	RW	0	OPA 模块时钟使能。 0：禁止 1：使能
29	Reserved	-	-	Reserved
28	PWREN	RW	0	低功耗控制模块时钟使能。 0：禁止 1：使能
27	CTCEN	RW	0	CTC 模块时钟使能。 0：禁止 1：使能
26:23	Reserved	-	-	Reserved
22	I2C2EN	RW	0	I2C2 模块时钟使能。 0：禁止 1：使能
21	I2C1EN	RW	0	I2C1 模块时钟使能。 0：禁止 1：使能
20	Reserved	-	-	Reserved
19	USART4EN	RW	0	USART4 模块时钟使能。 0：禁止 1：使能
18	USART3EN	RW	0	USART3 模块时钟使能。 0：禁止 1：使能
17	USART2EN	RW	0	USART2 模块时钟使能。

Bit	Name	R/W	Reset Value	Function
				0：禁止 1：使能
16:15	Reserved	-	-	Reserved
14	SPI2EN	RW		SPI2 模块时钟使能。 0：禁止 1：使能 该寄存器由硬件系统复位清零。
13:12	Reserved	-	-	Reserved
11	WWDGEN	RW	0	Window WDG 模块时钟使能。 0：禁止 1：使能 该寄存器由硬件系统复位清零。
10	RTCAPBEN	RW	0	RTC 模块 APB 时钟使能。 0：禁止 1：使能
9:6	Reserved	-	-	Reserved
5	TIM7EN	RW	0	TIM7 模块时钟使能。 0：禁止 1：使能
4	TIM6EN	RW	0	TIM6 模块时钟使能。 0：禁止 1：使能
3:2	Reserved	-	-	Reserved
1	TIM3EN	RW	0	TIM3 模块时钟使能。 0：禁止 1：使能
0	TIM2EN	RW	0	TIM2 模块时钟使能。 0：禁止 1：使能

8.5.16. APB 外设时钟使能寄存器 2（RCC_APBENR2）

Address offset:0x40

Reset value:0x0000 0001

该寄存器由软件置位和清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	LCDEN	Res	COMP2EN	COMP1EN	Res	TIM17EN	TIM16EN	Res
								RW		RW	RW		RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14EN	USART1EN	Res	SPI1EN	TIM1EN	MCUDBGEN	ADCEN	Res	Res	Res	Res	Res	Res	Res	Res	SYSCFGEN
RW	RW		RW	RW	RW	RW									RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
23	LCDEN	RW	0	LCD 模块时钟使能。 0：禁止 1：使能
22	Reserved	-	-	Reserved
21	COMP2EN	RW	0	COMP2 模块时钟使能。 0：禁止 1：使能
20	COMP1EN	RW	0	COMP1 模块时钟使能。 0：禁止 1：使能
19	Reserved	-	-	Reserved
18	TIM17EN	RW	0	TIM17 模块时钟使能。 0：禁止 1：使能
17	TIM16EN	RW	0	TIM16 模块时钟使能。 0：禁止 1：使能
16	TIM15EN	RW	0	TIM15 模块时钟使能。 0：禁止 1：使能
15	TIM14EN	RW	0	TIM14 模块时钟使能。 0：禁止 1：使能
14	USART1EN	RW	0	USART1 模块时钟使能。 0：禁止 1：使能
13	Reserved	-	-	Reserved
12	SPIEN	RW	0	SPI1 模块时钟使能。 0：禁止 1：使能
11	TIM1EN	RW	0	TIM1 模块时钟使能。 0：禁止 1：使能
10	MCUDBGEN	RW-	0	MCUDBG 模块时钟使能。 0：禁止 1：使能
9	ADCEN	RW	0	ADC 模块时钟使能。 0：禁止 1：使能
8:1	Reserved	-	-	Reserved
0	SYSCFGEN	RW	1	SYSCFG、COMP 和 VREFBUF 模块时钟使能。 0：禁止 1：使能

8.5.17. 外设独立时钟配置寄存器（RCC_CCIPR）

Address offset:0x54

Reset value:0x0000 0000

该寄存器由软件置位和清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	LPTIM1SEL [1:0]		Res	Res
												RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res		Res		Res	Res	COMP2 SEL	COMP1 SEL	PVD SEL	Res	Res	Res	Res	Res	Res	Res
						RW	RW	RW							

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	Reserved
19:18	LPTIMSEL[1:0]	RW	2'b00	LPTIM1 内部时钟源选择。 00: PCLK 01: LSI 10: No clock 11: LSE
17:10	Reserved	-	-	Reserved
9	COMP2SEL	RW	0	COMP2 模块时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） 注：在使能 COMP2_FR2.FLTEN 之前先配置选择 LSC 时钟。
8	COMP1SEL	RW	0	COMP1 模块时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） 注：在使能 COMP1_FR1.FLTEN 之前先配置该寄存器选择时钟。
7	PVDSEL	RW	0	PVD detect 时钟源选择。 0: PCLK 1: LSC（RCC_BDCR.LSCOSEL 选择后的时钟） 注：当时钟源选择 PCLK 时，需要将 PWR_CR1.FLTEN 配置为 0. 若使能 FLTEN，必须选择 LSC 时钟，且在使能 FLTEN 之前先配置选择 LSC 时钟。
6:0	Reserved	-	-	Reserved

8.5.18. RTC 域控制寄存器（RCC_BDCR）

Address offset:0x5C

Reset value:0x0000 0000，reset by POR/BOR

当连续访问该寄存器时，0 ≤ wait state ≤ 3。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	R es	Res	Res	LSC OSEL	LSC OE N	Res	Res	Res	Res	Res	Res	Res	BDRST
						RW	RW	RW							RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RT C EN	Res	Res	R es	Res	Res	RTCSEL[1: 0]		Res	LSE CS SD	LSEC SSO N	Res	Res	LSE-BYP	LSER DY	LSEON
RW						RW	RW		R	RW			RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25	LSCOSEL	RW	0	低速时钟选择。

Bit	Name	R/W	Reset Value	Function
				0：LSI 1：LSE
24	LSCOEN	RW	0	低速时钟使能。 0：禁止 1：使能
23:17	Reserved	-	-	Reserved
16	BDRST	RW	0	RTC domain 软复位。 0：no effect 1：复位
15	RTCEN	RW		RTC 时钟使能。软件设置或者清零。 0：禁止 1：使能
14:10	Reserved	-	-	Reserved
9:8	RTCSEL[1：0]	RW	0	RTC 时钟源选择。 00: No clock 01:LSE 10: LSI 11: HSE divided by 128 一旦 RTC 时钟源选择后不能再改变，除非以下情况： ■ RTC domain 被复位为 00 ■ 选择 LSE(LSECSSD=1)但没有 LSE
7	Reserved	-	-	Reserved
6	LSECSSD	R	0	CSS 检测 LSE 失败。 该位由硬件置位，表明 CSS 检测 32KHz OSC（LSE）失败。 0：未检到 LSE 失败 1：检测 LSE 失败
5	LSECSSON	RW		CSS 使能 LSE 时钟。 0：禁止； 1：使能； 注：必须 LSEON=1 并且 LSERDY=1 后才能使能 LSECSSON。 一旦使能该位，不能再把该位禁止，除非 LSECSSD=1.
4:3	Reserved	-	-	Reserved
2	LSEBYP	RW		LSE OSC bypass 0: Not bypassed，低速外部时钟选择晶振； 1: Bypassed，低速外部时钟选择外部接口输入时钟； 注：只有当外部 32KHz OSC 禁止（LSEON=0 并且 LSERDY=0）时才能写该位。
1	LSERDY	R		LSE OSC ready. 硬件配置该位为 1 表明 LSE 时钟 ready。
0	LSEON	RW		LSE OSC 使能。 0：禁止； 1：使能；

8.5.19. 控制/状态寄存器 (RCC_CSR)

Address offset:0x60

Reset value:0x0000 0000

该寄存器中复位标志位只能由 power reset 复位，其他由 system reset 复位。

当连续访问该寄存器时，0 ≤ wait state ≤ 3。

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

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	Reserved
30	WWDGRSTF	R	0	Window WDG 复位标志。 RMVF 置 1 会清零该位。
29	IWDGRSTF	R	0	IWDG 复位标志。 RMVF 置 1 会清零该位。
28	SFTRSTF	R	0	软复位标志。 RMVF 置 1 会清零该位。
27	PWRRSTF	R	0	BOR/POR/PDR 复位标志。 RMVF 置 1 会清零该位。
26	PINRSTF	R	0	外部 NRST 管脚复位标志。 RMVF 置 1 会清零该位。
25	OBLRSTF	R	0	Option byte loader 复位标志。 RMVF 置 1 会清零该位。
24	Reserved	-	-	Reserved
23	RMVF	RW	0	需通过软件置 1 来清零[30:25]的复位标志。
22:9	Reserved	-	-	Reserved
8	PINRST_FLTDIS	RW	0	NRST 滤波宽度 20us 禁止 0：使能 HSI_10M，且滤波 20us 宽度功能使能 1：滤波功能禁止，同步到系统时钟产生系统复位
7:2	Reserved	-	-	Reserved
1	LSIRDY	R	0	LSI OSC 稳定标志。 0：LSI 未稳定 1：LSI 已稳定
0	LSION	RW	0	LSI OSC 使能。 0：禁止 1：使能 硬件开启模拟 LSI 的情况： 1) 硬件 IWDG 使能； 2) LSECSS 使能；

8.5.20. RCC 寄存器地址映像

Offset	Bit Width	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	32	RCC_CR	Reserved						PLL_RDY	PLLON	Reserved	ADC_DIV		Reserved	CSSON	HSEBYP	HSERDY	HSEON	Reserved		HSIDIV[2:0]			HSIRDY	Reserved	HSION	Reserved							
		Read/Write							R	RW		RW			RW	RW	R	RW			RW	RW	RW	R		RW								
		Reset Value	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0x04	32	RCC_ICSCR	Reserved							LSI_TRIM[8:0]									HSI_FS[2:0]			HSI_TRIM[12:0]													
		Read/Write								RW									RW																
		Reset Value	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	1
0x08	32	RCC_CFGR	Reserved	MCOPRE[2:0]			MCOSEL[3:0]				Reserved									PPRE[2:0]			HPRE[3:0]				Reserved		SW[2:0]			SW[2:0]		
		Read/Write		RW			RW													RW			RW						R			RW		
		Reset Value	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	32	RCC_PLLCFGR	Reserved																												PLLMUL[1:0]		PLLSRC[1:0]	
		Read/Write																													RW		RW	
		Reset Value	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
0x010	32	RCC_ECSCR	Reserved										LSE_STARTUP[1:0]		Reserved		LSE_DRV[1:0]		Reserved											HSE_STARTUP[1:0]		Reserved	HSE_DRV[1:0]	
		Read/Write											RW				RW													RW			RW	
		Reset Value	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
0x18	32	RCC_CIER	Reserved																										PLLRDYI	HSER-	HSIRDYI	Reserved	LSER-	LSIRDYIE
		Read/Write																											RW	RW	RW		RW	RW

Offset	Bit Width	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
		Reset Value	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	32	RCC_CIFR	Reserved																						LSECSSF	CSSF	Reserved		PLLRDYF	HSERDYF	HSIRDYF	Reserved	LSERDYF	LSIRDYF
		Read/Write																							R	R			R	R	R		R	R
		Reset Value	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
0x20	32	RCC_CICR	Reserved																						LSECSSC	CSSC	Reserved		PLLRDYC	HSERDYC	HSIRDYC	Reserved	LSERDYC	LSIRDYC
		Read/Write																							W	W			W	W	W		W	W
		Reset Value	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	32	RCC_IOPRSTR	Reserved																										GPIOFRST	Reserved		GPIOCRST	GPIOBRST	GPIOARST
		Read/Write																											RW			RW	RW	RW
		Reset Value	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	32	RCC_AHBRSTR	Reserved							DIVRST	Reserved											CRCRST	Reserved											DMARST
		Read/Write								RW												RW												RW
		Reset Value	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
0x2C	32	RCC_APBRSTR1	LPTIMRST	OPARST	Reserved	PWRRST	CTCRST	Reserved	Reserved	Reserved	Reserved	I2C2RST	I2C1RST	Reserved	USART4RST	USART3RST	USART2RST	Reserved		SPI2RST	Reserved		WWDGRST	RTCAPBRST	Reserved				TIM7RST	TIM6RST	Reserved		TIM3RST	TIM2RST
		Read/Write	RW	RW		RW	RW					RW	RW		RW	RW	RW			RW			RW	RW					RW	RW			RW	RW
		Reset Value	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

Offset	Bit Width	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	32	RCC_APBRSTR2	Reserved								LCDRST	Reserved	COMP2RST	COMP1RST	Reserved	TIM17RST	TIM16RST	TIM15RST	TIM14RST	USART1RST	Reserved	SPI1RST	TIM1RST	MCUDBGRST	ADCRST	Reserved								SYSCFGRST
		Read/Write									RW		RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW									RW
		Reset Value	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
0x34	32	RCC_IOPENR	Reserved																										GPIOFEN	Reserved		GPIOCEN	GPIOBEN	GPIOAEN
		Read/Write																											RW			RW	RW	RW
		Reset Value	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
0x38	32	RCC_AHBENR	Reserved							DIVEN	Reserved											CRCEN	Reserved		SRAMEN	FLASHEN	Reserved							DMAEN
		Read/Write								RW												RW			RW	RW								RW
		Reset Value	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	32	RCC_APBENR1	LPTIMEN	OPAEN	Reserved	PWREN	CTCEN	Reserved	Reserved	Reserved	Reserved	I2C2EN	I2C1EN	Reserved	USART4EN	USART3EN	USART2EN	Reserved		SPI2EN	Reserved		WWDGEN	RTCAPBEN	Reserved				TIM7EN	TIM6EN	Reserved		TIM3EN	TIM2EN
		Read/Write	RW	RW		RW	RW					RW	RW		RW	RW	RW			RW			RW	RW					RW	RW			RW	RW
		Reset Value	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
0x40	32	RCC_APBENR2	Reserved								LCDEN	Reserved	COMP2EN	COMP1EN	Reserved	TIM17EN	TIM16EN	TIM15EN	TIM14EN	USART1EN	Reserved	SPI1EN	TIM1EN	MCUDBGEN	ADCEN	Reserved								SYSCFGEN
		Read/Write									RW		RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW									RW
		Reset Value	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	1
0x54	32	RCC_CCIPR	Reserved												LPTIMSEL[1:0]		Reserved							Reserved	COMP2SEL	COMP1SEL	PVDSEL	Reserved	Reserved					
		Read/Write													RW										RW	RW	RW							

Offset	Bit Width	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
		Re-set Value	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
0x5C	32	RCC_BDCR	Reserved						LSCOSEL	LSCOEN	Reserved							BDRST	RTCEN	Reserved					RTCSEL[1:0]		Reserved	LSECSSD	LSECSSON	Reserved		LSEBYP	LSERDY	LSEON
		Read/Write							RW	RW								RW	RW						RW			R	RW			RW	R	RW
		Re-set Value	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
0x60	32	RCC_CSR	Reserved	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLSTF	Reserved	RMVF	Reserved														PINRST_FLTDIS	Reserved						LSIRDY	LSION
		Read/Write		R	R	R	R	R	R		RW															RW							R	RW
		Re-set Value	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

9. 通用 I/O（GPIO）

9.1. 通用 IO 简介

GPIO 包含 PA[15:0], PB[15:0], PC[15:0]和 PF[9:0],每个 GPIO 端口有:

- 4 个 32 位配置寄存器(GPIOx_MODER,GPIOx_OTYPER,GPIOx_OSPEEDR, GPIOx_PUPDR)
- 2 个 32 位数据寄存器 (GPIOx_IDR 和 GPIOx_ODR)
- 1 个 32 位置位/复位寄存器(GPIOx_BSRR)
- 1 个 32 位锁定寄存器(GPIOx_LCKR)
- 2 个复用功能选择寄存器(GPIOx_AFRH 和 GPIOx_AFRL)
- 1 个 32 位复位寄存器(GPIOx_BRR)

9.2. 通用 IO 功能描述

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

9.3. 通用 IO 功能描述

每个 GPIO 的每个位，可以通过软件编程，进行几种模式的配置:

1) 输入浮空
2) 输入上拉
3) 输入下拉
4) 模拟输入
5) 开漏输出，带上拉或者下拉
6) 推挽输出，带上拉或者下拉
7) 带上拉或者下拉的复用功能的推挽
8) 带上拉或者下拉的复用的开漏

每个 I/O 口可以自由编程，然而 I/O 端口寄存器必须按 32 位字、半字或者字节被访问。GPIOx_BSRR 和 GPIOx_BRR 寄存器允许对任何 GPIOx_ODR 寄存器的读/更改的独立访问。这样，在读和更改访问之间产生 IRQ 时不会发生危险。

下图给出了一个 I/O 端口（1bit）的基本结构

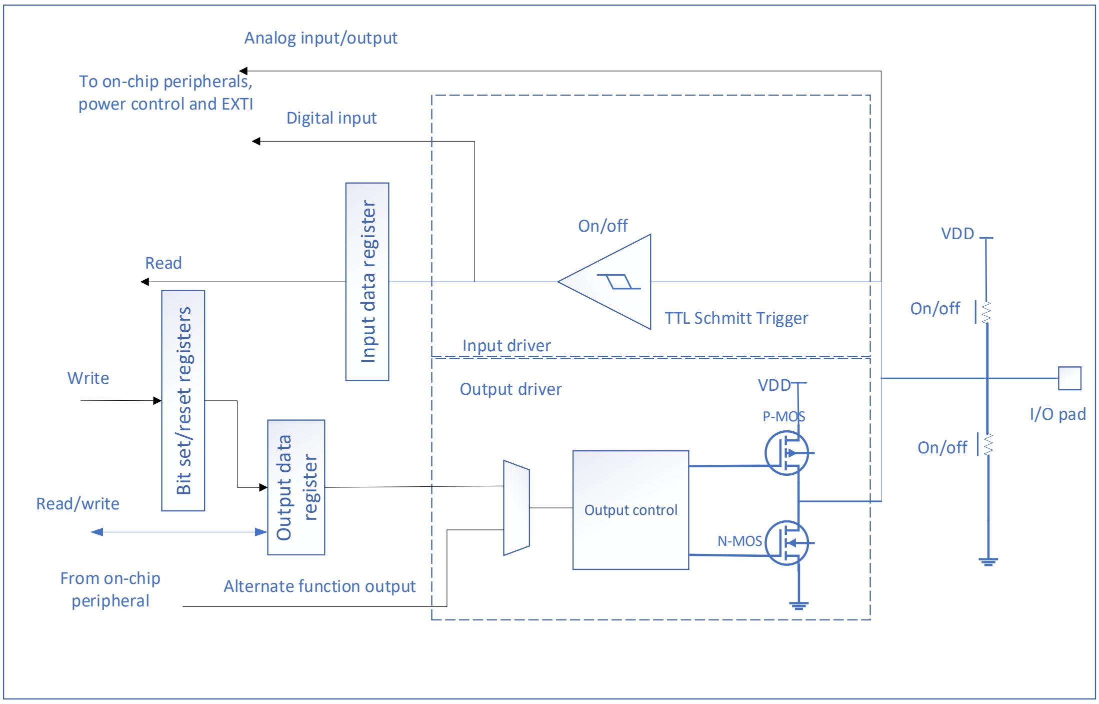


图 9-1 IO 端口位的基本结构

9.3.1. 通用 I/O(GPIO)

复位期间和复位后，复用功能未被激活，大多数 IO 被配置为模拟模式。

Debug 引脚默认被置于复用功能上拉或下拉模式：

—PA14-SWCLK：置于下拉模式

—PA13-SWDIO：置于上拉模式

Boot 引脚默认置于输入模式，下拉模式

—PF8-Boot：置于下拉模式

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

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

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

9.3.2. I/O 管脚复用功能多路选择和映射

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

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

- 刚复位后，多路选择器默认为 AF0。I/O 口的复用功能模式通过寄存器 GPIOx_MODER 配置
- 每个脚的复用功能分布在对应的章节

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

用户按照如下说明去配置 IO：

- 调试功能：每次复位后，这些调试功能脚就是默认为调试器立即可用的复用功能脚
- GPIO：在 GPIOx_MODER 将对应 I/O 口配置为输出、输入或者模拟模式
- 外设复用功能：

■ 寄存器 GPIOx_AFRL 或者 GPIOx_AFRH 配置对应的 I/O 为复用功能 x(x=0…15)

■ 寄存器 GPIOx_OTYPER, GPIOx_PUPDR 和 GPIOX_OSPEEDER 分别配置类型，上拉/下拉以及输出速度

■ 寄存器 GPIOx_MODER 是配置对应 I/O 为复用功能

- 总线读写功能：GPIO 模块除了支持 IO port 读写 GPIOx 寄存器外，也支持通过 AHB 总线读写寄存器，寄存器访问方式由 SYSCFG 模块的 GPIO_AHB_SEL bit 选择。当 GPIO_AHB_SEL bit 为 0 时，只能通过 IO port 访问 GPIOx 寄存器；当 GPIO_AHB_SEL 为 1 时，只有通过 AHB 总线访问 GPIOx 寄存器。
- AHB 总线访问 GPIO 寄存器的方式，除了支持 CPU 外，还支持 DMA，即 DMA 可以通过 AHB 总线直接访问 GPIO 寄存器。

■ 额外功能

- 无论 IO 口配置成任何模式，ADC 和 COMP 功能均在 ADC 和 COMP 模块的寄存器中使能。当 IO 口用做 ADC 或者 COMP 使用时，推荐通过寄存器 GPIOx_MODER 将该口配置为模拟模式
- 对于晶振额外功能，在相应的 PWR 与 RCC 模块寄存器里配置各自功能。这些配置比标准的 GPIO 配置具有更高优先级。

9.3.3. I/O 控制寄存器

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

9.3.4. I/O 数据寄存器

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

9.3.5. I/O 数据按位处理

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

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

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

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

在对 GPIOx_ODR 进行位操作时，软件无需禁止中断：在一次原子 AHB1 写访问中，可以修改一个或多个位。

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

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

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

9.3.6. I/O 复用功能输入/输出模式配置

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

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

9.3.7. 外部中断/唤醒线

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

9.3.8. I/O 输入配置

当 I/O 口配置为输入：

- 输出缓冲器不使能
- 施密特触发器输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可得到 I/O 状态

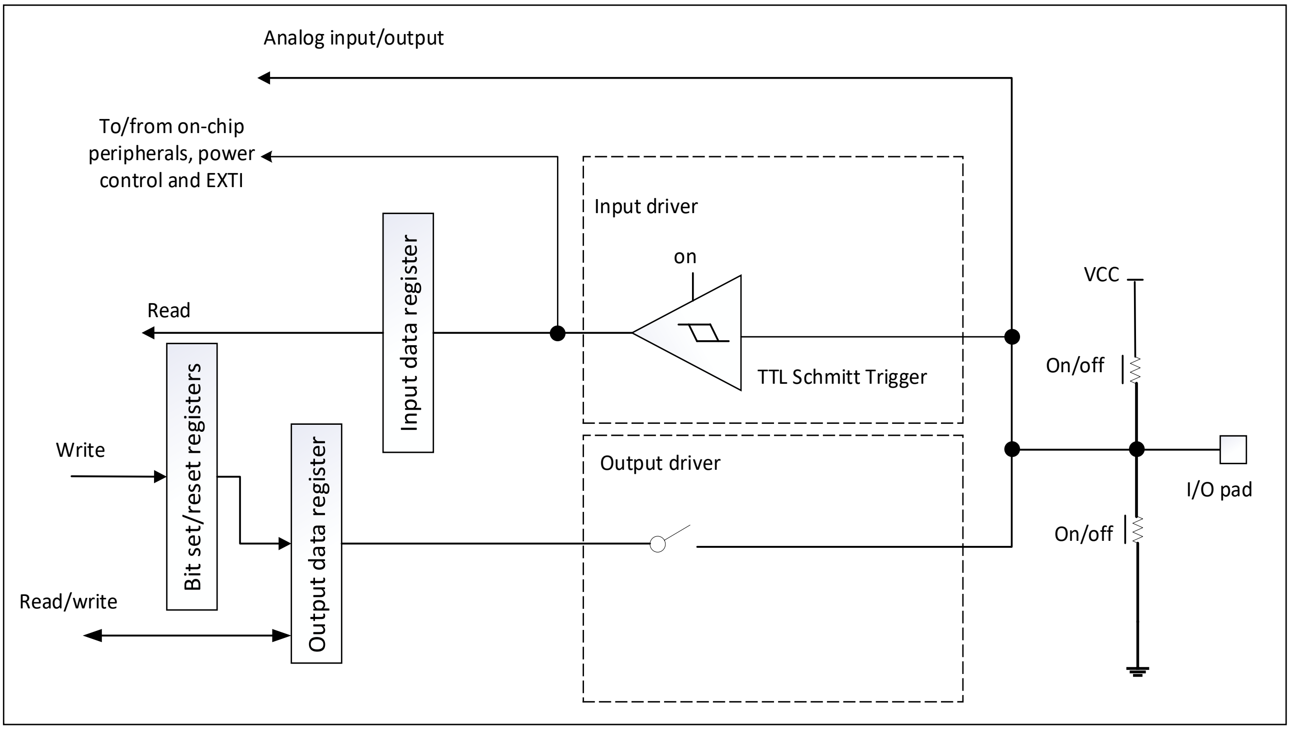


图 9-2 输入浮空/上拉/下拉配置

9.3.9. I/O 输出配置

当 I/O 端口被配置为输出时：

- 输出缓冲器被激活

  —开漏模式：输出寄存器上的'0'激活 N-MOS，而输出寄存器上的'1'将端口置于高阻状态(PMOS 从不被激活)。

  —推挽模式：输出寄存器上的'0'激活 N-MOS，而输出寄存器上的'1'将激活 P-MOS。

- 施密特触发输入被激活
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可得到 I/O 状态
- 对输出数据寄存器的读访问得到后一次写的值

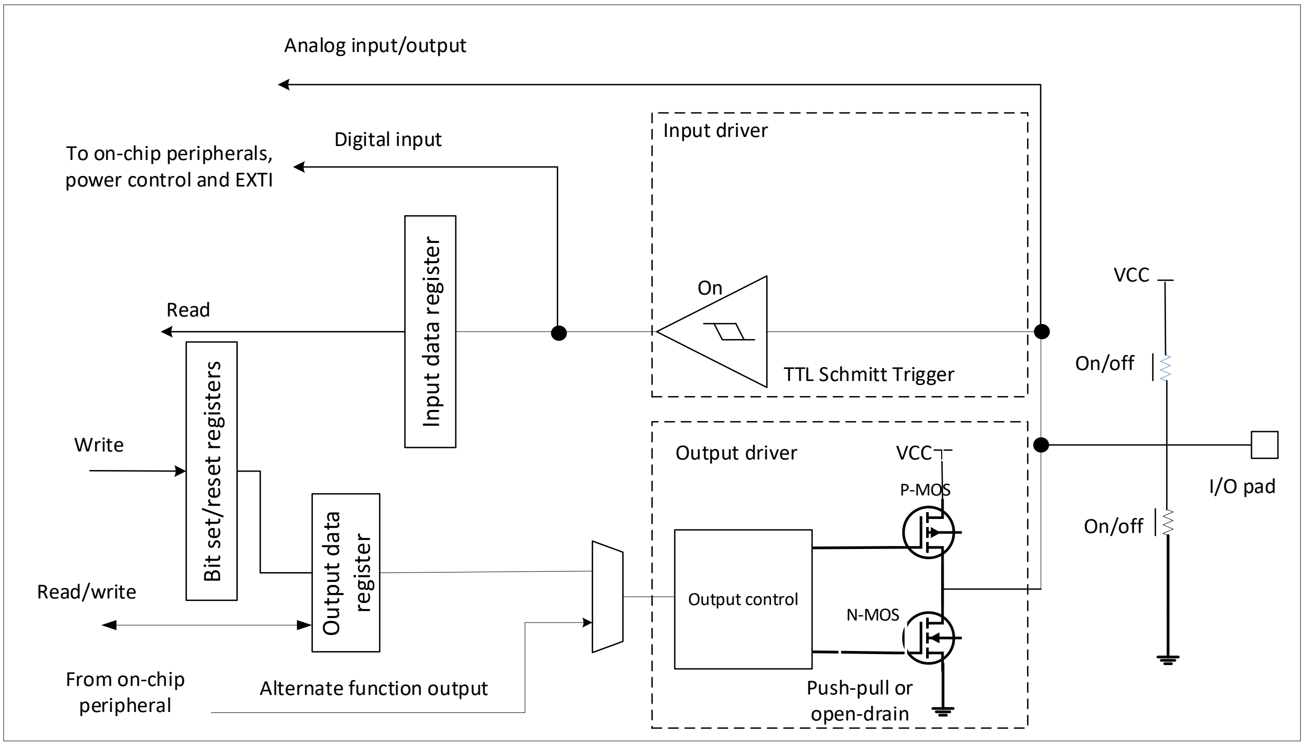


图 9-3 输出配置

9.3.10. 复用功能配置

当 I/O 端口被配置为复用功能时:

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

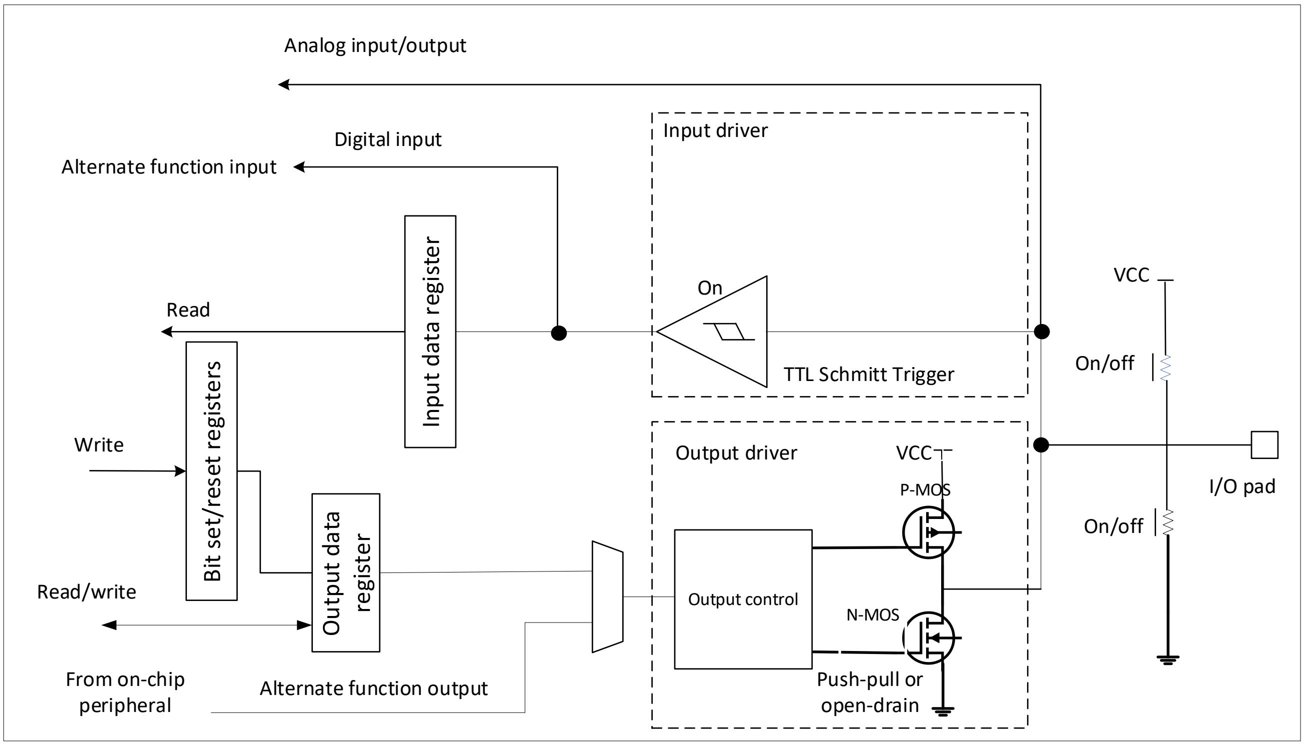


图 9-4 复用功能配置

9.3.11. 模拟配置

当 I/O 端口被配置为模拟配置时：

- 输出缓冲器被禁止；
- 禁止施密特触发输入，实现了每个模拟 I/O 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（需要软件设定）；
- 读取输入数据寄存器时数值为'0'。

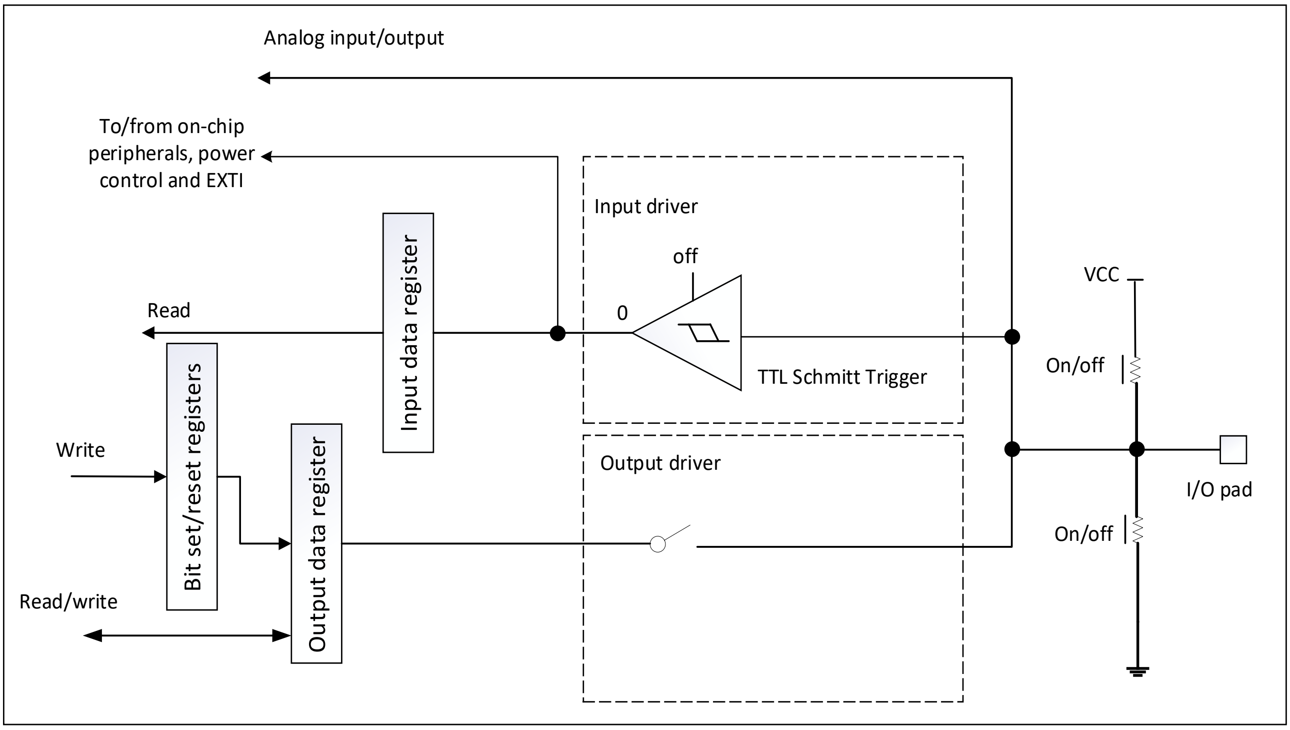


图 9-5 高阻抗模拟配置

9.3.12. 使用 HSE/LSE 管脚作为 GPIO

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

当 HSE 或 LSE 功能打开（RCC_CSR 寄存器中设置 HSEON or LSEON），需要软件配置对应的端口为模拟端口。

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

9.4. GPIO 寄存器

所有 GPIO 相关寄存器都可进行 word、half word 和 byte 写操作。

9.4.1. GPIO 端口模式寄存器 (GPIOx_MODER) (x=A, B, F)

Address offset: 0x00

Reset value:

0xEBFF FFFF for GPIOA

0xFFFF FFFF for GPIOB

0xFFFF FFFF for GPIOC

0x000C FFFF For GPIOF

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

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

9.4.2. GPIO 端口输出类型寄存器(GPIOx_OTYPER) (x = A, B, F)

Address offset: 0x04

Reset value: 0x0000 0000

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

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

9.4.3. GPIO 端口输出速度寄存器(GPIOx_OSPEEDR) (x = A, B, F)

Address offset: 0x08

Reset value: 0x0C00 0000(for port A)

Reset value: 0x0000 0000(for other ports)

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

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

9.4.4. GPIO 端口上下拉寄存器(GPIOx_PUPDR) (x = A, B, F)

Address offset: 0x0C

Reset value:

- 0x2400 0000(for port A)
- 0x0000 0000(for port B,C)

■ 0x0002 0000(for port F)

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

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

9.4.5. GPIO 端口输入数据寄存器(GPIOx_IDR) (x = A, B, F)

Address offset: 0x10

Reset value: 0x0000 XXXX

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

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

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

Address offset: 0x14

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31: 16	Reserved			
15: 0	Ody[1:0]	RW		y = 15..0 软件可读可写。 说明：对 GPIOx_BSRR or GPIOx_BRR registers. (x=A,B,C,F)，可以分别对各个 ODR 位进行独立的设置/清除。

9.4.7. GPIO 端口位设置/复位寄存器(GPIOx_BSRR) (x = A, B, C, F)

Address offset: 0x18

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	BRy	W		y = 15..0 软件可写，读出来返回值是 0 0: 对对应的 ODRy 位不产生影响 1: 清除对应的 ODRy 位 注：如果同时设置 Bsy 和 Bry 的对应位，Bsy 位起作用
15: 0	BSy	W		y = 15..0 软件可写， 读出来返回值是 0 0: 对对应的 ODRy 位不产生影响 1: 设置对应的 ODRy 位

9.4.8. GPIO 端口配置锁定寄存器(GPIOx_LCKR) (x = A, B, F)

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

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

每个锁定位冻结一种特定的配置寄存器（控制和复用功能寄存器）

Address offset: 0x1C

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:17	Reserved			
16	LCKK	RW		该位可随时读出，它只能通过锁键写入序列修改 0: 端口配置锁键位未激活 1: 端口配置锁键位被激活，下次系统复位前 GPIOx_LCKR 寄存器被锁定 LOCK key write sequence: 锁键的写入时序: WR LCKR[16] = ‘1’ + LCKR[15:0] WR LCKR[16] = ‘0’ + LCKR[15:0] WR LCKR[16] = ‘1’ + LCKR[15:0] RD LCKR[16] = ‘0’ RD LCKR[16] = ‘1’（此读操作为可选操作，但它可确认锁定已激活） 注：在操作锁键的写入时序是，不能改变 LCK[15:0]的值。锁键时序的任何错误都会终止锁键被激活。对端口的

Bit	Name	R/W	Reset Value	Function
				任何一位首次锁键时序之后，读 LCKK 位都是返回 1,直接 MCU 复位或者外围复位。
15: 0	LCKy	RW		y = 15..0 这些位可读可写但只能在 LCKK 位为 0 时写入。 0: 不锁定端口的配置 1: 锁定端口配置

9.4.9. GPIO 复用功能寄存器（low）(GPIOx_AFRL) (x = A, B, F)

Address offset: 0x20

Reset value: 0x0000 0000

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

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

9.4.10. GPIO 复用功能寄存器（high）(GPIOx_AFRH) (x = A, B, F)

Address offset: 0x24

Reset value: 0x0000 0000

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

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

9.4.11. GPIO 端口位复位寄存器 (GPIOx_BRR) (x = A, B, F)

Address offset: 0x28

Reset value: 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	Bry	W		y = 15..0 这些位软件可写，读出来返回值是 0 0: 对对应的 ODRy 位不产生影响 1: 清除对应的 ODRy 位

9.4.12. GPIO 寄存器映像

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	GPIOA_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]		MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
	Reset value	1	1	1	0	1	0	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
	GPIOB_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]		MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	GPIOF_MODER	MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]		MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0x04	GPIOF_OTYPER (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	GPIOA_OSPEEDR	OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]		OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
	Reset value	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 (x=B, F)	OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]		OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
	Reset value	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	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]		PUPD7[1:0]		PUPD6[1:0]		PUPD5[1:0]		PUPD4[1:0]		PUPD3[1:0]		PUPD2[1:0]		PUPD1[1:0]		PUPD0[1:0]	
	Reset value	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

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
0x0C	GPIOB_PUPDR	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]		PUPD7[1:0]		PUPD6[1:0]		PUPD5[1:0]		PUPD4[1:0]		PUPD3[1:0]		PUPD2[1:0]		PUPD1[1:0]		PUPD0[1:0]	
	Reset value	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	GPIOF_PUPDR	PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]		PUPD7[1:0]		PUPD6[1:0]		PUPD5[1:0]		PUPD4[1:0]		PUPD3[1:0]		PUPD2[1:0]		PUPD1[1:0]		PUPD0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0x10	GPIOx_IDR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
	Reset value																	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
0x14	GPIOx_ODR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	GPIOx_BSRR (x=A, B, 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 value	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 (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	GPIOx_AFRL (x=A, B, F)	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 value	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 (x=A, B, F)	AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]				AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]			
	Reset value	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	GPIOx_BRR (x=A, B, F)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10. 系统配置控制器(SYSCFG)

芯片内有一套配置寄存器，系统配置控制器的主要目的是：

- IO pin 接口控制
- 重映射位于代码区间开始区域的存储器
- 管理 TIMERs ETR 或者刹车输入
- DMA 外设通道选择控制

10.1. 系统配置寄存器

10.1.1. SYSCFG 配置寄存器 1(SYSCFG_CFGR1)

有两位用作配置存储器地址 0x0000 0000 访问的种类。这两位用来选择软件的物理重映射，并 bypass 掉硬件 BOOT 选择。在复位后，这些位使用被实际 boot 模式配置的值。

Address offset:0x00

Reset value:0x0000 000x(x 是被实际 boot 模式配置选择的存储器模式)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES							GPIO_AHB_SEL	RES					ETR_SRC_TIM3[2:0]		
							RW						RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RES	ETR_SRC_TIM2[2:0]			RES	ETR_SRC_TIM1[2:0]			TIM3_IC1_SRC		TIM2_IC4_SRC		TIM1_IC1_SRC		MEM_MODE	
	RW				RW			RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31:25	RES	RES	-	RES
24	GPIO_AHB_SEL	RW	0	CPU FASTIO 或 AHB 总线访问 GPIO 寄存器控制。 0：FASTIO 总线访问； 1：AHB 总线访问；
23:19	RES	RES	-	RES
18:16	ETR_SRC_TIM3[2:0]	RW	0	TIMER3 ETR 输入源选择。 3’b000: ETR 来源于 GPIO； 3’b001: ETR 来源于 COMP1； 3’b010: ETR 来源于 COMP2； 3’b011: ETR 来源于 ADC； others: reserved
15	RES	RES	-	RES
14:12	ETR_SRC_TIM2[2:0]	RW	0	TIMER2 ETR 输入源选择。 3’b000: ETR 来源于 GPIO； 3’b001: ETR 来源于 COMP1； 3’b010: ETR 来源于 COMP2； 3’b011: ETR 来源于 ADC； others: reserved
11	RES	RES	-	RES
10:8	ETR_SRC_TIM1[2:0]	RW	0	TIMER1 ETR 输入源选择。 3’b000: ETR 来源于 GPIO； 3’b001: ETR 来源于 COMP1； 3’b010: ETR 来源于 COMP2； 3’b011: ETR 来源于 ADC；

Bit	Name	R/W	Reset Value	Function
				others: reserved
7:6	TIM3_IC1_SRC	RW	0	TIM13 CH1 输入来源 00：from TIM3_CH1 IO； 01：from comp1； 10：from comp2；
5:4	TIM2_IC4_SRC	RW	0	TIM2 CH4 输入来源 00：from TIM2_CH4 IO； 01：from comp1； 10：from comp2；
3:2	TIM1_IC1_SRC	RW	0	TIM1 CH1 输入来源 00：from TIM1_CH1 IO； 01：from comp1； 10：from comp2；
1:0	MEM_MODE	RW	x	存储区映射选择。 x0: Main Flash memory 映射在 0x0000 0000 01: System Flash memory 映射在 0x0000 0000 11: Embedded SRAM 映射在 0x0000 0000

10.1.2. SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)

Address offset:0x04

Reset value： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	COMP2_Oc-ref_CLR_TIM3	COMP2_Oc-ref_CLR_TIM2	COMP2_Oc-ref_CLR_TIM1	COMP1_Oc-ref_CLR_TIM3	COMP1_Ocref_CLR_TIM2
											RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
COMP1_Oc-ref_CLR_TIM1	RES	COMP2_BRK_TIM17	COMP1_BRK_TIM17	RES	COMP2_BRK_TIM16	COMP1_BRK_TIM16	RES	COMP2_BRK_TIM15	COMP1_BRK_TIM15	RES	COMP2_BRK_TIM1	COMP1_BRK_TIM1	PVD_LOCK	RES	LOCKUP_LOCK
RW		RW	RW		RW			RW	RW		RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:21	RES	RES	-	RES
20	COMP2_Ocref_CLR_TIM3	RW	0	1：COMP2 输出作为 TIM3 ocref_clr 输入； 0：COMP2 输出不作为 TIM3 ocref_clr 输入
19	COMP2_Ocref_CLR_TIM2	RW	0	1：COMP2 输出作为 TIM2 ocref_clr 输入； 0：COMP2 输出不作为 TIM2 ocref_clr 输入
18	COMP2_Ocref_CLR_TIM1	RW	0	1：COMP2 输出作为 TIM1 ocref_clr 输入； 0：COMP2 输出不作为 TIM1 ocref_clr 输入
17	COMP1_Ocref_CLR_TIM3	RW	0	1：COMP1 输出作为 TIM3 ocref_clr 输入； 0：COMP1 输出不作为 TIM3 ocref_clr 输入

Bit	Name	R/W	Reset Value	Function
16	COMP1_Ocref_CLR_TIM2	RW	0	1：COMP1 输出作为 TIM2 ocref_clr 输入； 0：COMP1 输出不作为 TIM2 ocref_clr 输入
15	COMP1_Ocref_CLR_TIM1	RW	0	1：COMP1 输出作为 TIM1 ocref_clr 输入； 0：COMP1 输出不作为 TIM1 ocref_clr 输入
14	RES	RES	-	RES
13	COMP2_ BRK_TIM17	RW	0	COMP2 作为 TIM17 break 输入使能。 0：COMP2 输出不与 TIM17 break input 相连； 1：COMP2 输出作为 TIM17 break input；
12	COMP1_BRK_TIM17	RW	0	COMP1 作为 TIM1 break 输入使能。 0：COMP1 输出不与 TIM1 break input 相连； 1：COMP1 输出作为 TIM1 break input；
11	RES	RES	-	RES
10	COMP2_ BRK_TIM16	RW	0	COMP2 作为 TIM16 break 输入使能。 0：COMP2 输出不与 TIM16 break input 相连； 1：COMP2 输出作为 TIM16 break input；
9	COMP1_BRK_TIM16	RW	0	COMP1 作为 TIM16 break 输入使能。 0：COMP1 输出不与 TIM16 break input 相连； 1：COMP1 输出作为 TIM16 break input；
8	RES	RES	-	RES
7	COMP2_ BRK_TIM15	RW	0	COMP2 作为 TIM15 break 输入使能。 0：COMP2 输出不与 TIM15 break input 相连； 1：COMP2 输出作为 TIM15 break input；
6	COMP1_BRK_TIM15	RW	0	COMP1 作为 TIM15 break 输入使能。 0：COMP1 输出不与 TIM15 break input 相连； 1：COMP1 输出作为 TIM15 break input；
5	RES	RES	-	RES
4	COMP2_ BRK_TIM1	RW	0	COMP2 作为 TIM1 break 输入使能。 0：COMP2 输出不与 TIM1 break input 相连； 1：COMP2 输出作为 TIM1 break input；
3	COMP1_BRK_TIM1	RW	0	COMP1 作为 TIM1 break 输入使能。 0：COMP1 输出不与 TIM1 break input 相连； 1：COMP1 输出作为 TIM1 break input；
2	PVD_LOCK	RW	0	PVD lock 使能位。 0：PVD 中断不与 TIM1/15/16/17 break input 相连，由 PVDE 和 PLS 配置 PVD 功能； 1：PVD 中断与 TIM1/15/16/17 break input 相连， PVDE 和 PLS 只读；
1	RES	RES	-	RES
0	LOCKUP_LOCK	RW	0	Cortex-M0+ LOCK ″ U ″ P 位 lock 使能位。 0：Cortex-M0+ LOCKUP 位不与 TIM1/15/16/17 break input 相连； 1：Cortex-M0+ LOCKUP 位与 TIM1/15/16/17 break input 相连；

10.1.3. SYSCFG 配置寄存器 3 (SYSCFG_CFGR3)

Address offset:0x08

Reset value:0x3F3F 3F3F

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

Bit	Name	R/W	Reset Value	Function
31:30	RES	RES	-	RES
29:24	DMA4_MAP	RW	0x3F	DMA1 通道 4 映射。 见 DMA1_MAP 描述。
23:22	RES	RES	-	RES
21:16	DMA3_MAP	RW	0x3F	DMA1 通道 3 映射。 见 DMA1_MAP 描述。
15:14	RES	RES	-	RES
13:8	DMA2_MAP	RW	0x3F	DMA1 通道 2 映射。 见 DMA1_MAP 描述。
7:6	RES	RES	-	RES
5:0	DMA1_MAP	RW	0x3F	DMA1 通道 1 映射。 000000: ADC1; 000011: SPI1_RD; 000100: SPI1_WR; 000101: SPI2_RD; 000110: SPI2_WR; 000111: USART1_RD; 001000: USART1_WR; 001001: USART2_RD; 001010: USART2_WR; 001011: USART3_RD; 001100: USART3_WR; 001101: USART4_RD; 001110: USART4_WR; 001111: I2C1_RD; 010000: I2C1_WR; 010001: I2C2_RD; 010010: I2C2_WR; 010011: TIM1_CH1; 010100: TIM1_CH2; 010101: TIM1_CH3; 010110: TIM1_CH4; 010111: TIM1_COM; 011000: TIM1_TRG; 011001: TIM1_UP; 011010: TIM2_CH1; 011011: TIM2_CH2; 011100: TIM2_CH3; 011101: TIM2_CH4; 011110: TIM2_UP; 011111: TIM2_TRG; 100000: TIM3_CH1; 100001: TIM3_CH2; 100010: TIM3_CH3; 100011: TIM3_CH4; 100100: TIM3_UP; 100101: TIM3_TRG; 100110: TIM6_UP; 100111: TIM7_UP; 101000: TIM15_CH1; 101001: TIM15_CH2; 101010: TIM15_UP; 101011: TIM15_TRG; 101100: TIM15_COM, 101101: TIM16_CH1;

Bit	Name	R/W	Reset Value	Function
				101110: TIM16_UP; 101111: TIM17_CH1; 110000: TIM17_UP; 110010：LCD others：保留

10.1.4. SYSCFG 配置寄存器 4 (SYSCFG_CFGR4)

Address offset:0x0C

Reset value:0x003F 3F3F

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

Bit	Name	R/W	Reset Value	Function
31:22	RES	RES	-	RES
21:16	DMA7_MAP	RW	0x3F	DMA1 通道 7 映射。 见 DMA1_MAP 描述。
15:14	RES	RES	-	RES
13:8	DMA6_MAP	RW	0x3F	DMA1 通道 6 映射。 见 DMA1_MAP 描述。
7:6	RES	RES	-	RES
5:0	DMA5_MAP	RW	0x3F	DMA1 通道 5 映射。 见 DMA1_MAP 描述。

10.1.5. GPIOA 滤波使能（PA_ENS）

Address offset:0x10

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RES	-	RES
15:0	PA_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

10.1.6. GPIOB 滤波使能（PB_ENS）

Address offset:0x14

Reset value:0x0000 0000

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

PB_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	RES	-	RES
15:0	PB_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

10.1.7. GPIOC 滤波使能（PC_ENS）

Address offset:0x18

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15:0	PC_ENS[x]	RW	0x0000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

10.1.8. GPIOF 滤波使能（PF_ENS）

Address offset:0x1C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	RES	-	RES
9:0	PF_ENS[x]	RW	0x000	Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

10.1.9. I2C 类型 IO 配置寄存器（SYSCFG_EIIC）

Address offset:0x20

Reset value:0x0000 0000

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="6">RES</td><td colspan="2">PF_EIIC[1:0]</td><td colspan="5">RES</td><td colspan="3">PB_EIIC[10:0]</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8"></td><td colspan="6">RES</td><td colspan="2">PA_EIIC[1:0]</td></tr>
<tr><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td></tr>
</table>

Bit	Name	R/W	Reset Value	Function
31:26	Reserved			
25:24	PF_EIIC[x]			PF I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled
23:19	Reserved			
18:8	PB_EIIC[x]			PB I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled
7:2	Reserved			
1:0	PA_EIIC[x]	RW	0	PA I2C IO Noise filter enable, active high 0: noise filter bypassed 1: noise filter enabled

10.1.10. SYSCFG 寄存器映像

<table>
<tr><th>Offset</th><th>Register</th><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="2">0x00</td><td>SYSCFG_CFGR1</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>GPIO_AHB_SEL</td><td>I2C_PA10_ANF</td><td>Res.</td><td>Res.</td><td>Res.</td><td>GPIO_AHB_SEL</td><td colspan="3">ETR_SRC_TIM3[2:0]</td><td>Res.</td><td colspan="3">ETR_SRC_TIM2[2:0]</td><td>Res.</td><td colspan="3">ETR_SRC_TIM1[2:0]</td><td colspan="2">TIM3_IC1_SRC</td><td colspan="2">TIM2_IC4_SRC</td><td colspan="2">TIM1_IC1_SRC</td><td colspan="2">MEM_MODE[1:0]</td></tr>
<tr><td>Reset value</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td>0</td><td>0</td><td>0</td><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>X</td><td>X</td></tr>
<tr><td rowspan="2">0x04</td><td>SYSCFG_CFGR2</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>COMP2_Ocref_CLR_TIM3</td><td>COMP2_Ocref_CLR_TIM2</td><td>COMP2_Ocref_CLR_TIM1</td><td>COMP1_Ocref_CLR_TIM3</td><td>COMP1_Ocref_CLR_TIM2</td><td>COMP1_Ocref_CLR_TIM1</td><td>Res.</td><td>COMP2_BRK_TIM17</td><td>COMP1_BRK_TIM17</td><td>Res.</td><td>COMP2_BRK_TIM16</td><td>COMP1_BRK_TIM16</td><td>Res.</td><td>COMP2_BRK_TIM15</td><td>COMP1_BRK_TIM15</td><td>COMP1_BRK_TIM16</td><td>COMP2_BRK_TIM1</td><td>COMP1_BRK_TIM1</td><td>PVD_LOCK</td><td>Res.</td><td>LOCKUP_LOCK</td></tr>
<tr><td>Reset value</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td><td>0</td></tr>
<tr><td rowspan="2">0x08</td><td>SYSCFG_CFGR3</td><td>Res.</td><td>Res.</td><td colspan="6">DMA4_MAP</td><td>Res.</td><td>Res.</td><td colspan="6">DMA3_MAP</td><td>Res.</td><td>Res.</td><td colspan="6">DMA2_MAP</td><td>Res.</td><td>Res.</td><td colspan="6">DMA1_MAP</td></tr>
<tr><td>Reset value</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="6">0x3F</td><td></td><td></td><td colspan="6">0x3F</td><td></td><td></td><td colspan="6">0x3F</td></tr>
<tr><td>0x0C</td><td>SYSCFG_CF</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td>Res.</td><td colspan="6">DMA7_MAP</td><td>Res.</td><td>Res.</td><td colspan="6">DMA6_MAP</td><td>Res.</td><td>Res.</td><td colspan="6">DMA5_MAP</td></tr>
</table>

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
	GR4																																
	Reset value													0x3F								0x3F								0x3F			
0x10	PA_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								PA_ENS[x]								
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	PB_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								PB_ENS[x]								
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	PC_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								PC_ENS[x]								
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	PF_ENS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								PF_ENS[x]								
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	SYSCFG_EIIC	Res.	Res.	Res.	Res.	Res.	Res.	PF_IECC[x]												PB_IECC[x]												PA_IECC[x]	
	Reset value														0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.DMA

11.1. 简介

直接存储器存取(DMA)用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。搬移数据无需 CPU 干预，数据可以通过 DMA 快速地移动，这就节省了 CPU 的资源来做其他操作。DMA 控制器有 7 个通道，每个通道专门用来管理来自于一个或多个外设对存储器访问的请求。还有一个仲裁器来协调各个 DMA 请求的优先权。

11.2. DMA 主要特性

- 单 AHB master
- 支持外设到存储器，存储器到外设，存储器到存储器和外设到外设的数据传输
- 片上存储器设备，如 FLASH, SRAM, AHB 和 APB 外设，作为源和目标
- 所有 DMA 通道均可独立配置：
  - 每个通道要么与来自外设的 DMA 请求信号相关联，要么与存储器到存储器传输中的软件触发器相关联。这个配置是由软件完成的。
  - 请求之间的优先级由软件可编程(每个通道 4 级：非常高、高、中、低)，在相等的情况下由硬件可编程(例如对通道 1 的请求比对通道 2 的请求优先)。
  - 源和目标的传输大小是独立的(字节，半字，字)，模拟打包和拆包。源地址和目标地址必须按数据大小对齐。
  - 可编程传输数据数：0 ~65535
- 每个通道生成一个中断请求。每个中断请求都是由三个 DMA 事件中的任何一个引起的：传输完成、半传输或传输错误。

11.3. DMA 功能描述

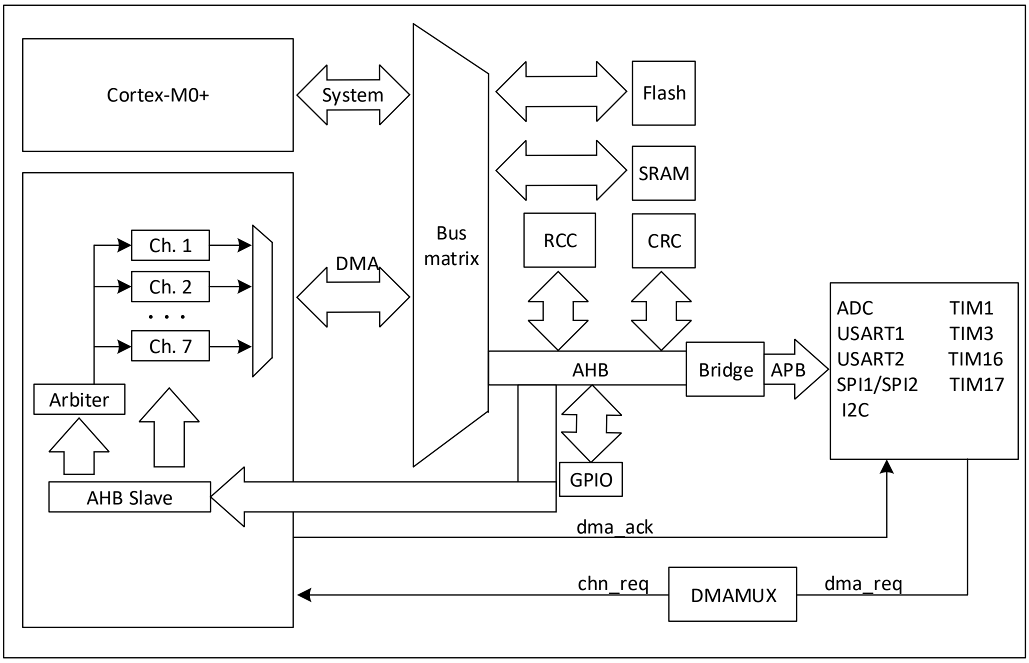


图 11-1 DMA 框图

11.3.1. DMA 处理

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

总之，每次 DMA 传送由三个操作组成：

- 从外设数据寄存器或者从当前外设/存储器地址寄存器取数，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设基地址或存储器单元。
- 存数据到外设寄存器或者当前外设/存储器地址寄存器指示的存储器地址，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设及地址或存储器单元。
- 执行一次 DMA_CNDTRx 寄存器的递减操作，该寄存器表明未完成的操作数目。

11.3.2. 仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。

优先权管理分 2 个阶段：

- 软件：每个通道的优先权可以在 DMA_CCRx 寄存器中设置，有 4 个等级
  - 最高优先级
  - 高优先级
  - 中等优先级
  - 低优先级
- 硬件：如果 2 个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有较高的优先权。比如，通道 2 优先于通道 4。

11.3.3. DMA 通道

每个通道都可以在有固定地址的外设寄存器和存储器地址之间执行 DMA 传输。DMA 的传输数据量是可编程的，最大为 65535 字节，该寄存器值在每次数据传输后递减。

传输数据量可编程

外设和存储器的传输量可以通过 DMA_CCRx 寄存器中的 PSIZE 和 MSIZE 位编程。

地址指针增量

通过设置 DMA_CCRx 寄存器中的 PINC 和 MINC 标志位，外设和存储器的指针在每次传输后可以有选择地完成自动增量。

当设置为增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决与所选的数据宽度为 1、 2 或 4。第一个传输的地址是存放在 DMA_CPARx/DMA_CMARx 寄存器中地址。在传输过程中，这些寄存器保持它们初始的数值，软件不能改变和读出当前正在传输的地址(它在内部的当前外设/存储器地址寄存器中)。

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

在循环模式下，最后一次传输结束时， DMA_CNDTRx 寄存器的内容会自动地被重新加载为其初始数值，内部的当前外设/存储器地址寄存器也被重新加载为 DMA_CPARx/DMA_CMARx 寄存器设定的初始基地址。

通道配置过程

按如下步骤配置外设请求时 DMA 通道：

- 在 DMA_CPARx 寄存器中设置外设寄存器的地址。发生外设数据传输请求时，这个地址将是数据传输的源或目标。
- 在 DMA_CMARx 寄存器中设置数据存储器的地址。发生外设数据传输请求时，传输的数据将从这个地址读出或写入这个地址。
- 在 DMA_CNDTRx 寄存器中设置要传输的数据量。在每个数据传输后，这个数值递减。
- 配置 DMA_CCRx 寄存器如下参数：
  - 通道的优先级。
  - 数据传输方向
  - 循环模式
  - 外设和存储器增量模式
  - 外设和存储器数据大小
  - 中断使能
- 设置 DMA_CCRx 寄存器的 ENABLE 位，启动该通道。

一旦启动了 DMA 通道，即可响应连到该通道上的外设的 DMA 请求。

当传输一半的数据后，半传输标志(HTIF)被置 1，当设置了允许半传输中断位(HTIE)时，将产生一个中断请求。在数据传输结束后，传输完成标志(TCIF)被置 1，当设置了允许传输完成中断位(TCIE)时，将产生一个中断请求。

通道状态及禁止通道

处于激活状态的通道 x 是一个启用的通道(读取 DMA_CCRx.EN=1)。一个活动通道 x 是一个必须已经被软件启用的通道(DMA_CCRx. EN = 1)，然后没有发生传输错误(DMA_ISR.TEIFx = 0).如果出现传输错误，通道会被硬件自动禁用(DMA_CCRx.EN = 0)。

以下 3 种情况可能会发生：

- 暂停和恢复通道

这对应以下两个动作:

--激活的通道被软件禁用(写入 DMA_CCRx.EN = 0)。

--软件重新启用通道(DMA_CCRx.EN=1)，但没有重新配置其他通道寄存器(如 DMA_CNDTRx、DMA_CPARx 和 DMA_CMARx)；或者软件禁用时未完成的传输挂起了总线。

DMA 硬件不支持这种情况，因此不能保证正确地执行剩余的数据传输。

- 停止和中止一个通道

如果应用程序不再需要该通道，则可以通过软件禁用该活动通道。通道被停止并中止，但是 DMA_CNDTRx 寄存器内容可能不能正确地反映剩余的数据传输。

- 中止并重新启动通道

这对应于软件顺序:禁用活动通道，然后重新配置通道并再次启用它。

满足以下条件时，硬件支持:

--应用程序保证，当软件禁用通道时，DMA 没有正在进行（还未完成的）传输。例如，应用程序可以首先在 DMA 模式下禁用外设，以确保该外设没有挂起的硬件 DMA 请求。

--软件必须对同一个 DMA_CCRx 寄存器进行独立的写访问:首先关闭通道，其次，如果需要更改配置，则重新配置通道，以便进行下一次块传输，包括 DMA_CCRx。最后再次启用通道。

当发生通道传输错误时，硬件清除 DMA_CCRx 寄存器的 EN 位。在 DMA_ISR 寄存器的 TEIFx 位被设置之前，这个 EN 位不能被软件再次设置以重新激活通道 x。

DMA 循环模式

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

注：在存储器到存储器模式中不能使用循环模式。在循环模式(CIRC = 1)使能通道前，软件必须清除 DMA_CCRx 寄存器的 MEM2MEM 位。当循环模式被激活时，要传输的数据量将在通道配置阶段使用编程的初始值自动重新加载，DMA 请求将继续得到响应。

为了停止循环传输，软件需要在禁用 DMA 通道之前停止外设生成 DMA 请求(例如退出 ADC 扫描模式)。

在开始/启用一个传输之前，以及在停止一个循环传输之后，软件必须明确地对 DMA_CNDTRx 值进行编程。

存储器到存储器模式

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

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

存储器到存储器模式不能与循环模式同时使用。

外设到外设模式

任何 DMA 通道都可以在外设到外设模式下运行:

- 当来自外设的硬件请求被选择触发 DMA 通道时

这个外设是 DMA 启动器，在这个外设和属于另一个存储器映射外设(这个外设没有配置为 DMA 模式)的寄存器之间进行数据传输。

- 当没有选择外设请求并连接到 DMA 通道时

该软件通过设置 DMA_CCRx 寄存器的 MEM2MEM 位来配置寄存器到寄存器的传输。

配置传输方向，指定源/目标

DMA_CCRx 寄存器的 DIR 位的值设置了传输的方向，因此，它标识了源和目标，而不管源/目标类型(外设或存储器):

- DIR = 1 通常定义存储器到外设的传输。更一般地，如果 DIR = 1:
  - 源属性由 DMA_MARx 寄存器、MSIZE[1:0]字段和 DMA_CCRx 寄存器的 MINC 位定义。
    不管它们通常的命名是什么，这些“存储器”寄存器、字段和位被用来在外设到外设模式下定义源外设。
  - 目标属性由 DMA_PARx 寄存器、DMA_CCRx 寄存器的 PSIZE[1:0]字段和 PINC 位定义。
    不管它们通常的命名，这些“外设”寄存器、字段和位被用来在存储器到存储器模式下定义目标存储器。
- DIR = 0 通常定义一个外设到存储器的传输。更一般地，如果 DIR = 0:
  - 源属性由 DMA_PARx 寄存器、DMA_CCRx 寄存器的 PSIZE[1:0]字段和 PINC 位定义。
    不管它们通常的命名是什么，这些“外设”寄存器、字段和位被用来在存储器到存储器模式下定义源存储器。
  - 目标属性由 DMA_MARx 寄存器、DMA_CCRx 寄存器的 MSIZE[1:0]字段和 MINC 位定义。

不管它们通常的命名，这些“存储器”寄存器、字段和位被用来以外设到外设模式定义目标地外设。

11.3.4. 数据传输宽度/对齐方式/大小端

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

表 11-1 数据宽度和大小端（PINC=MINC=1）

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

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
8	32	4	0x0/B0	1:在 0x0 读 B0[7:0],在 0x0 写 000000B0[31:0]	0x0/000000B0
			0x1/B1	2:在 0x1 读 B1[7:0],在 0x4 写 000000B1[31:0]	0x4/000000B1
			0x2/B2	3:在 0x2 读 B2[7:0],在 0x8 写 000000B2[31:0]	0x8/000000B2
			0x3/B3	4:在 0x3 读 B3[7:0],在 0xC 写 000000B3[31:0]	0xC/000000B3
16	8	4	0x0/B1B0	1:在 0x0 读 B1B0[15:0],在 0x0 写 B0[7:0]	0x0/B0
			0x2/B3B2	2:在 0x2 读 B3B2[15:0],在 0x1 写 B2[7:0]	0x1/B2
			0x4/B5B4	3:在 0x4 读 B5B4[15:0],在 0x2 写 B4[7:0]	0x2/B4
			0x6/B7B6	4:在 0x6 读 B7B6[15:0],在 0x3 写 B6[7:0]	0x3/B6
16	16	4	0x0/B1B0	1:在 0x0 读 B1B0[15:0],在 0x0 写 B1B0[15:0]	0x0/B1B0
			0x2/B3B2	2:在 0x2 读 B3B2[15:0],在 0x2 写 B3B2[15:0]	0x2/B3B2

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
			0x4/B5B4	3:在 0x4 读 B5B4[15:0],在 0x4 写 B5B4[15:0]	0x4/B5B4
			0x6/B7B6	4:在 0x6 读 B7B6[15:0],在 0x6 写 B7B6[15:0]	0x6/B7B6
16	32	4	0x0/B1B0	1:在 0x0 读 B1B0[7:0],在 0x0 写 0000B1B0[31:0]	0x0/0000B1B0
			0x2/B3B2	2:在 0x2 读 B3B2[7:0],在 0x4 写 0000B3B2[31:0]	0x4/000B3B2
			0x4/B5B4	3:在 0x4 读 B5B4[7:0],在 0x8 写 0000B5B4[31:0]	0x8/0000B5B4
			0x6/B7B6	4:在 0x6 读 B7B6[7:0],在 0xC 写 0000B7B6[31:0]	0xC/0000B7B6
32	8	4	0x0/B3B2B1B0	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B0[7:0]	0x0/B0
			0x4/B7B6B5B4	2:在 0x4 读 B7B6B5B4 [31:0],在 0x1 写 B4[7:0]	0x1/B4

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
			0x8/BBBAB9B8	3:在 0x8 读 BBBAB9B8 [31:0],在 0x2 写 B8[7:0]	0x2/B8
			0xC/BFBEBDBC	4:在 0xc 读 BFBEBDBC [31:0],在 0x3 写 BC[7:0]	0x3/BC
32	16	4	0x0/B3B2B1B0	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B1B0[7:0]	0x0/B1B0
			0x4/B7B6B5B4	2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B5B4[7:0]	0x2/B5B4
			0x8/BBBAB9B8	3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 B9B8[7:0]	0x4/B9B8
			0xC/BFBEBDBC	4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BDBC[7:0]	0x6/BDBC
32	32	4	0x0/B3B2B1B0	1:在 0x0 读 B3B2B1B0 [31:0],在 0x0 写 B3B2B1B0[7:0]	0x0/B3B2B1B0
			0x4/B7B6B5B4	2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B7B6B5B4[7:0]	0x4/B7B6B5B4

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
			0x8/BBBAB9B8	3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 BBBAB9B8[7:0]	0x8/BBBAB9B8
			0xC/BFBEBDBC	4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BFBEBDBC[7:0]	0xC/BFBEBDBC

操作不支持字节或半字写的 AHB 设备

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

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

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

- 一个 AHB 上对地址 0x0(或 0x1、0x2 或 0x3)的写字节数据'0xB0'操作，将转换到 APB 上对地址 0x0 的写字数据'0xB0B0B0B0'操作。
- 一个 AHB 上对地址 0x0(或 0x2)的写半字数据'0xB1B0'操作，将转换到 APB 上对地址 0x0 的写字数据'0xB1B0B1B0'操作。

11.3.5. 错误管理

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

DMA_CCRx 寄存器的 EN 位不能被软件再次设置(通道 x 重新激活)，直到 DMA_ISR 寄存器的 TEIFx 位被清除(通过设置 DMA_IFCR 寄存器的 CTEIFx 位)。

当软件通过一个涉及外设的通道收到一个传输错误的通知时，软件首先在 DMA 模式下停止这个外设，以便禁用任何等待或将来的 DMA 请求。然后，软件通常会在 DMA 模式下重新配置 DMA 和外设，以进行新的传输。

11.3.6. DMA 中断

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

表 11-2 DMA 中断请求

中断事件	事件标志位	使能控制位
传输过半	HTIFx	HTIEx
传输完成	TCIFx	TCIEx
传输错误	TEIFx	TEIEx
传输过半/传输完成/传输错误	GIFx	-

注：1. 当 DMA_CNDTRx 寄存器为 1 时，不会置 HTIFx 位，传输完成会置 TCIFx 位。

2. 当传输长度 NDT 为奇数时（大于 1），HTIF 和 TCIF 标志都会产生。内部信号 TCIF 会在 NDT=1 时产生；HTIF 会在（NDT-（NDT/2（取整）-1））时产生。如 NDT=5 时，TCIF 在 NDT 减至 1 时产生；HTIF 在 NDT 减至 4 时产生。

3. 当传输长度 NDT 为偶数时（大于 1），HTIF 和 TCIF 标志都会产生。内部信号 TCIF 会在 NDT=1 时产生；HTIF 会在（NDT-（NDT/2（取整）-1））时产生。如 NDT=10 时，TCIF 在 NDT 减至 1 时产生；HTIF 在 NDT 减至 6 时产生。

11.3.7. DMA 外设请求映射

DMA 外设请求映射到 DMA 的各个通道，由 SYSCFG 两个寄存器（SYSCFG_CFGR3 和 SYS-CFGR_CFGR4）的 DMAx_MAP 寄存器位控制，每个外设请求通过配置可以映射到 7 个通道中的任意一个。

表 11-3 每个通道的 DMA 请求

请求 MUX 输入序号	源	请求 MUX 输入序号	源	请求 MUX 输入序号	源
0	ADC1	17	I2C2_RD	34	TIM3_CH3
1	保留	18	I2C2_WR	35	TIM3_CH4
2	保留	19	TIM1_CH1	36	TIM3_UP
3	SPI1_RD	20	TIM1_CH2	37	TIM3_TRIG
4	SPI1_WR	21	TIM1_CH3	38	TIM6_UP
5	SPI2_RD	22	TIM1_CH4	39	TIM7_UP
6	SPI2_WR	23	TIM1_COM	40	TIM15_CH1
7	USART1_RD	24	TIM1_TRIG	41	TIM15_CH2
8	USART1_WR	25	TIM1_UP	42	TIM15_UP
9	USART2_RD	26	TIM2_CH1	43	TIM15_TRIG
10	USART2_WR	27	TIM2_CH2	44	TIM15_COM
11	USART3_RD	28	TIM2_CH3	45	TIM16_CH1
12	USART3_WR	29	TIM2_CH4	46	TIM16_UP
13	USART4_RD	30	TIM2_UP	47	TIM17_CH1
14	USART4_WR	31	TIM2_TRG	48	TIM17_UP
15	I2C1_RD	32	TIM3_CH1	49	保留
16	I2C1_WR	33	TIM3_CH2	50	LCD

11.4. DMA 寄存器

11.4.1. DMA 中断状态寄存器（DMA_ISR）

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5
				R	R	R	R	R	R	R	R	R	R	R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
27	TEIF7	R	0	通道 7 传输错误标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 7 传输错误（TE）；
26	HTIF7	R	0	通道 7 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 7 发生半传输事件;
25	TCIF7	R	0	通道 7 传输完成标志。 0：无传输完成（TC）； 1：通道 7 传输完成（TC）；
24	GIF7	R	0	通道 7 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 7 发生 TE/HT/TC 事件;
23	TEIF6	R	0	通道 6 传输错误标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 6 传输错误（TE）；
22	HTIF6	R	0	通道 6 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 6 发生半传输事件;
21	TCIF6	R	0	通道 6 传输完成标志。 0：无传输完成（TC）； 1：通道 6 传输完成（TC）；
20	GIF6	R	0	通道 6 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 6 发生 TE/HT/TC 事件;
19	TEIF5	R	0	通道 5 传输错误标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 5 传输错误（TE）；
18	HTIF5	R	0	通道 5 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 5 发生半传输事件;
17	TCIF5	R	0	通道 5 传输完成标志。 0：无传输完成（TC）； 1：通道 5 传输完成（TC）；
16	GIF5	R	0	通道 5 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 5 发生 TE/HT/TC 事件;
15	TEIF4	R	0	通道 4 传输错误标志。

Bit	Name	R/W	Reset Value	Function
				硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 4 传输错误（TE）；
14	HTIF4	R	0	通道 4 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 4 发生半传输事件;
13	TCIF4	R	0	通道 4 传输完成标志。 0：无传输完成（TC）； 1：通道 4 传输完成（TC）；
12	GIF4	R	0	通道 4 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 4 发生 TE/HT/TC 事件;
11	TEIF3	R	0	通道 3 传输错误标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 3 传输错误（TE）；
10	HTIF3	R	0	通道 3 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 3 发生半传输事件;
9	TCIF3	R	0	通道 3 传输完成标志。 0：无传输完成（TC）； 1：通道 3 传输完成（TC）；
8	GIF3	R	0	通道 3 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 3 发生 TE/HT/TC 事件;
7	TEIF2	R	0	通道 2 传输错误标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 2 传输错误（TE）；
6	HTIF2	R	0	通道 2 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 2 发生半传输事件;
5	TCIF2	R	0	通道 2 传输完成标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输完成（TC）； 1：通道 2 传输完成（TC）；
4	GIF2	R	0	通道 2 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 2 发生 TE/HT/TC 事件;
3	TEIF1	R	0	通道 1 传输错误标志。

Bit	Name	R/W	Reset Value	Function
				硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输错误（TE）； 1：通道 1 传输错误（TE）；
2	HTIF1	R	0	通道 1 半传输标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无半传输事件; 1：通道 1 发生半传输事件;
1	TCIF1	R	0	通道 1 传输完成标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无传输完成（TC）； 1：通道 1 传输完成（TC）；
0	GIF1	R	0	通道 1 全局中断标志。 硬件置位，软件写 DMA_IFCR=1 清零。 0：无 TE/HT/TC 事件; 1：通道 1 发生 TE/HT/TC 事件;

11.4.2. DMA 中断标志位清除寄存器（DMA_IFCR）

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	CTEIF7	CHTIF7	CTCIF7	CGIF7	CTEIF6	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5	CTCIF5	CGIF5
				W	W	W	W	W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27	CTEIF7	W	0	通道 7 传输错误标志清零。 0：No effect; 1：清零 TEIF7;
26	CHTIF7	W	0	通道 7 半传输标志清零。 0：No effect; 1：清零 HTIF7;
25	CTCIF7	W	0	通道 7 传输完成标志清零。 0：No effect; 1：清零 TCIF7;
24	CGIF7	W	0	通道 7 全局中断标志清零。 0：No effect; 1：清零通道 7 的 GIF/TEIF/HTIF/TCIF;
23	CTEIF6	W	0	通道 6 传输错误标志清零。 0：No effect;

Bit	Name	R/W	Reset Value	Function
				1：清零 TEIF6；
22	CHTIF6	W	0	通道 6 半传输标志清零。 0：No effect； 1：清零 HTIF6；
21	CTCIF6	W	0	通道 6 传输完成标志清零。 0：No effect； 1：清零 TCIF6；
20	CGIF6	W	0	通道 6 全局中断标志清零。 0：No effect； 1：清零通道 6 的 GIF/TEIF/HTIF/TCIF；
19	CTEIF5	W	0	通道 5 传输错误标志清零。 0：No effect； 1：清零 TEIF5；
18	CHTIF5	W	0	通道 5 半传输标志清零。 0：No effect； 1：清零 HTIF5；
17	CTCIF5	W	0	通道 5 传输完成标志清零。 0：No effect； 1：清零 TCIF5；
16	CGIF5	W	0	通道 5 全局中断标志清零。 0：No effect； 1：清零通道 5 的 GIF/TEIF/HTIF/TCIF；
15	CTEIF4	W	0	通道 4 传输错误标志清零。 0：No effect； 1：清零 TEIF4；
14	CHTIF4	W	0	通道 4 半传输标志清零。 0：No effect； 1：清零 HTIF4；
13	CTCIF4	W	0	通道 4 传输完成标志清零。 0：No effect； 1：清零 TCIF4；
12	CGIF4	W	0	通道 4 全局中断标志清零。 0：No effect； 1：清零通道 4 的 GIF/TEIF/HTIF/TCIF；
11	CTEIF3	W	0	通道 3 传输错误标志清零。 0：No effect； 1：清零 TEIF3；
10	CHTIF3	W	0	通道 3 半传输标志清零。 0：No effect； 1：清零 HTIF3；
9	CTCIF3	W	0	通道 3 传输完成标志清零。 0：No effect； 1：清零 TCIF3；
8	CGIF3	W	0	通道 3 全局中断标志清零。 0：No effect； 1：清零通道 3 的 GIF/TEIF/HTIF/TCIF；

Bit	Name	R/W	Reset Value	Function
7	CTEIF2	W	0	通道 2 传输错误标志清零。 0：No effect； 1：清零 TEIF2；
6	CHTIF2	W	0	通道 2 半传输标志清零。 0：No effect； 1：清零 HTIF2；
5	CTCIF2	W	0	通道 2 传输完成标志清零。 0：No effect； 1：清零 TCIF2；
4	CGIF2	W	0	通道 2 全局中断标志清零。 0：No effect； 1：清零通道 2 的 GIF/TEIF/HTIF/TCIF；
3	CTEIF1	W	0	通道 1 传输错误标志清零。 0：No effect； 1：清零 TEIF1；
2	CHTIF1	W	0	通道 1 半传输标志清零。 0：No effect； 1：清零 HTIF1；
1	CTCIF1	W	0	通道 1 传输完成标志清零。 0：No effect； 1：清零 TCIF1；
0	CGIF1	W	0	通道 1 全局中断标志清零。 0：No effect； 1：清零通道 1 的 GIF/TEIF/HTIF/TCIF；

11.4.3. DMA 通道 1 配置寄存器（DMA_CCR1）

Address offset:0x08

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道 1 存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道 1 优先级配置。 00：低；

Bit	Name	R/W	Reset Value	Function
				01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道 1 存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道 1 外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道 1 存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道 1 外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道 1 循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道 1 数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道 1 传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道 1 半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道 1 传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道 1 使能. 0：禁止； 1：通道 1 使能；

11.4.4. DMA 通道 1 数据传输数量寄存器（DMA_CNDTR1）

Address offset:0x0C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道 1 数据传输数量。 数据传输数量为 0~65535。该寄存器只在通道不工作（DMA_CCR1.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.5. DMA 通道 1 外设地址寄存器（DMA_CPAR1）

Address offset:0x10

Reset value:0x0000 0000

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

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

11.4.6. DMA 通道 1 存储地址寄存器（DMA_CMAR1）

Address offset:0x14

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道 1 存储器地址。 通道 1 存储器地址，作为数据传输的源或目标。 当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。

Bit	Name	R/W	Reset Value	Function
				当 MSIZE=2'b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.7. DMA 通道 2 配置寄存器（DMA_CCR2）

Address offset:0x1C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道 2 存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道 2 优先级配置。 00：低； 01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道 2 存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道 2 外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道 2 存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道 2 外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道 2 循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道 2 数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道 2 传输错误中断（TE）使能。

Bit	Name	R/W	Reset Value	Function
				0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道 2 半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道 2 传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道 2 使能. 0：禁止； 1：通道 1 使能；

11.4.8. DMA 通道 2 数据传输数量寄存器（DMA_CNDTR2）

Address offset:0x20

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道 2 数据传输数量。 数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR2.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.9. DMA 通道 2 外设地址寄存器（DMA_CPAR2）

Address offset:0x24

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	PA[31:0]	RW	0	通道 2 外设地址。

Bit	Name	R/W	Reset Value	Function
				通道 2 外设数据寄存器的基址，作为数据传输的源或目标。 当 PSIZE=2'b01，不使用 PA[0]位。操作自动与半字地址对齐。 当 PSIZE=2'b10，不使用 PA[1:0]位。操作自动与字地址对齐。

11.4.10. DMA 通道 2 存储地址寄存器（DMA_CMAR2）

Address offset:0x28

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道 2 存储器地址。 通道 2 存储器地址，作为数据传输的源或目标。 当 MSIZE=2'b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2'b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.11. DMA 通道 3 配置寄存器（DMA_CCR3）

Address offset:0x30

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道 3 存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道 3 优先级配置。 00：低； 01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道 3 存储器数据宽度。 00：8 位；

Bit	Name	R/W	Reset Value	Function
				01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道 3 外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道 3 存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道 3 外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道 3 循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道 3 数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道 3 传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道 3 半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道 3 传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道 3 使能. 0：禁止； 1：通道 1 使能；

11.4.12. DMA 通道 3 数据传输数量寄存器（DMA_CNDTR3）

Address offset:0x34

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道 3 数据传输数量。

Bit	Name	R/W	Reset Value	Function
				数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR3.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.13. DMA 通道 3 外设地址寄存器（DMA_CPAR3）

Address offset:0x38

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	PA[31:0]	RW	0	通道 3 外设地址。 通道 3 外设数据寄存器的基址，作为数据传输的源或目标。 当 PSIZE=2’b01，不使用 PA[0]位。操作自动与半字地址对齐。 当 PSIZE=2’b10，不使用 PA[1:0]位。操作自动与字地址对齐。

11.4.14. DMA 通道 3 存储器地址寄存器（DMA_CMAR3）

Address offset:0x3C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道 3 存储器地址。 通道 3 存储器地址，作为数据传输的源或目标。 当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2’b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.15. DMA 通道 4 配置寄存器（DMA_CCR4）

Address offset:0x44

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道优先级配置。 00：低； 01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道传输错误中断（TE）使能。 0：禁止；

Bit	Name	R/W	Reset Value	Function
				1：TE 中断使能；
2	HTIE	RW	0	通道半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道使能. 0：禁止； 1：通道使能；

11.4.16. DMA 通道 4 数据传输个数寄存器（DMA_CNDTR4）

Address offset:0x48

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道数据传输数量。 数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR3.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.17. DMA 通道 4 外设地址寄存器（DMA_CPAR4）

Address offset:0x4C

Reset value:0x0000 0000

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

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

11.4.18. DMA 通道 4 存储器地址寄存器（DMA_CMAR4）

Address offset:0x50

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道存储器地址。 通道存储器地址，作为数据传输的源或目标。 当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2’b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.19. DMA 通道 5 配置寄存器（DMA_CCR5）

Address offset:0x58

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道优先级配置。 00：低；

Bit	Name	R/W	Reset Value	Function
				01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道使能. 0：禁止； 1：通道使能；

11.4.20. DMA 通道 5 传输个数寄存器（DMA_CNDTR5）

Address offset:0x5C

Reset value:0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDT[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道数据传输数量。 数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR3.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.21. DMA 通道 5 外设地址寄存器（DMA_CPAR5）

Address offset:0x60

Reset value:0x0000 0000

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

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

11.4.22. DMA 通道 5 存储器地址寄存器（DMA_CMAR5）

Address offset:0x64

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道存储器地址。

Bit	Name	R/W	Reset Value	Function
				通道存储器地址，作为数据传输的源或目标。 当 MSIZE=2'b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2'b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.23. DMA 通道 6 配置寄存器（DMA_CCR6）

Address offset:0x6C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；
13：12	PL[1:0]	RW	0	通道优先级配置。 00：低； 01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道循环模式。 0：禁止；

Bit	Name	R/W	Reset Value	Function
				1：循环模式使能；
4	DIR	RW	0	通道数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道使能. 0：禁止； 1：通道使能；

11.4.24. DMA 通道 6 传输个数寄存器（DMA_CNDTR6）

Address offset:0x70

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道数据传输数量。 数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR3.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.25. DMA 通道 6 外设地址寄存器（DMA_CPAR6）

Address offset:0x74

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PA[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

11.4.26. DMA 通道 6 存储器地址寄存器（DMA_CMAR6）

Address offset:0x78

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道存储器地址。 通道存储器地址，作为数据传输的源或目标。 当 MSIZE=2'b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2'b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.27. DMA 通道 7 配置寄存器（DMA_CCR7）

Address offset:0x80

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	-	Reserved
14	MEM2MEM	RW	0	通道存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能；

Bit	Name	R/W	Reset Value	Function
13：12	PL[1:0]	RW	0	通道优先级配置。 00：低； 01：中等； 10：高； 11：很高；
11：10	MSIZE[1:0]	RW	0	通道存储器数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
9：8	PSIZE[1:0]	RW	0	通道外设数据宽度。 00：8 位； 01：16 位； 10：32 位； 11：保留。
7	MINC	RW	0	通道存储器地址增量模式。 0：禁止； 1：存储器地址增量模式使能；
6	PINC	RW	0	通道外设地址增量模式。 0：禁止； 1：外设地址增量模式使能；
5	CIRC	RW	0	通道循环模式。 0：禁止； 1：循环模式使能；
4	DIR	RW	0	通道数据传输方向。 0：从外设读； 1：从存储器读；
3	TEIE	RW	0	通道传输错误中断（TE）使能。 0：禁止； 1：TE 中断使能；
2	HTIE	RW	0	通道半传输中断（HT）使能。 0：禁止； 1：HT 中断使能；
1	TCIE	RW	0	通道传输完成中断（TC）使能。 0：禁止； 1：TC 中断使能；
0	EN	RW	0	通道使能. 0：禁止； 1：通道使能；

11.4.28. DMA 通道 7 传输个数寄存器（DMA_CNDTR7）

Address offset:0x84

Reset value:0x0000 0000

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

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	NDT[15:0]	RW	0	通道数据传输数量。 数据传输数量为 0~65535.该寄存器只在通道不工作（DMA_CCR3.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。

11.4.29. DMA 通道 7 外设地址寄存器（DMA_CPAR7）

Address offset:0x88

Reset value:0x0000 0000

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

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

11.4.30. DMA 通道 7 存储器地址寄存器（DMA_CMAR7）

Address offset:0x8C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：0	MA[31:0]	RW	0	通道存储器地址。 通道存储器地址，作为数据传输的源或目标。 当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2’b10，不使用 MA[1:0]位。操作自动与字地址对齐。

11.4.31. DMA 寄存器映像

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	DMA_ISR	Res.	Res.	Res.	Res.	TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
	Reset value					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	DMA_IFCR	Res.	Res.	Res.	Res.	CTEIF7	CHTIF7	CTCIF7.	CGIF7	CTEIF6.	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5.	CTCIF5	CGIF5	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
	Reset value					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	DMA_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DMA_CNDTR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	DMA_CPAR1	PA[31:0]																															
	Reset value	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 value	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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	DMA_CNDTR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															

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
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	DMA_CPAR2	PA[31:0]																															
	Reset value	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 value	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
0x2C	Reserved	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
0x30	DMA_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	DMA_CNDTR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	DMA_CPAR3	PA[31:0]																															
	Reset value	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 value	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
0x44	DMA_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	DMA_CNDTR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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
0x4C	DMA_CPAR4	PA[31:0]																															
	Reset value	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 value	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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	DMA_CNDTR5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	DMA_CPAR5	PA[31:0]																															
	Reset value	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 value	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
0x6C	DMA_CCR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x70	DMA_CNDTR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x74	DMA_CPAR6	PA[31:0]																															
	Reset value	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

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
0x78	DMA_CMAR6	MA[31:0]																															
	Reset value	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
0x80	DMA_CCR7	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[9:8]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x84	DMA_CNDTR7	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NDT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x88	DMA_CPAR7	PA[31:0]																															
	Reset value	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
0x8C	DMA_CMAR7	MA[31:0]																															
	Reset value	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

12. 中断和事件

12.1. 嵌套向量中断控制器(NVIC)

12.1.1. 主要特性

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

NVIC 和 CPU 接口是紧密耦合的，这使得低延迟中断处理和晚到达中断的高效处理成为可能。包括 CPU 的 exception，所有中断都被 NVIC 管理。

12.1.2. 系统嘀嗒（SysTick）校准值寄存器

系统嘀嗒校准值被设为 6000，通过 SysTick 时钟置为 6MHz（max f_{HCLK}/8），给出了 1ms 的参考 time base。

12.1.3. 中断和异常向量

位置	优先级	优先级类型	名称	说明	地址
-	-	-	-	保留	0x0000_0000
-	-3	固定	复位	复位	0x0000_0004
-	-2	固定	NMI_Handler	RCC 时钟安全系统(CSS)联接到 NMI 向量	0x0000_0008
-	-1	固定	HardFualt_Handler	所有类型的失效	0x0000_000C
-	3	可设置	SVCall	通过 SWI 指令的系统服务调用	0x0000_002C
-	5	可设置	PendSV	可挂起的系统服务	0x0000_0038
	6		SysTick	系统嘀嗒定时器	0x0000_003C
0	7	可设置	WWDG	窗口定时器中断	0x0000_0040
1	8	可设置	PVD	电源电压检测中断(EXTI line 16)	0x0000_0044
2	9	可设置	RTC	RTC 中断(combined EXTI lines 19)	0x0000_0048
3	10	可设置	Flash	Flash 全局中断	0x0000_004C
4	11	可设置	RCC_CTC	RCC 和 CTC 全局中断	0x0000_0050
5	12	可设置	EXTI0_1	EXTI line[1:0] interrupt	0x0000_0054
6	13	可设置	EXTI2_3	EXTI line[3:2] interrupt	0x0000_0058
7	14	可设置	EXTI4_15	EXTI line[15:4] interrupt	0x0000_005C
8	15	可设置	LCD	LCD 全局中断	0x0000_0060
9	16	可设置	DMA_Channel1	DMA 通道 1 中断	0x0000_0064
10	17	可设置	DMA_Channel2_3	DMA 通道 2 & 3 中断	0x0000_0068
11	18	可设置	DMA_Channel4_5_6_7	DMA 通道 4 & 5 & 6 & 7 中断	0x0000_006C
12	19	可设置	ADC_COMP	ADC and COMP interrupts (COMP combined with EXTI 17 & 18 & 20)	0x0000_0070
13	20	可设置	TIM1_BRK_UP_TRG_COM	TIM1 断开、更新、触发和通信中断	0x0000_0074

位置	优先级	优先级类型	名称	说明	地址
14	21	可设置	TIM1_CC	TIM1 捕获/比较中断	0x0000_0078
15	22	可设置	TIM2	TIM2 全局中断	0x0000_007C
16	23	可设置	TIM3	TIM3 全局中断	0x0000_0080
17	24	可设置	TIM6/LPTIM	TIM6/LPTIM 全局中断	0x0000_0084
18	25	可设置	TIM7	TIM7 全局中断	0x0000_0088
19	26	可设置	TIM14	TIM14 全局中断	0x0000_008C
20	27	可设置	TIM15	TIM15 全局中断	0x0000_0090
21	28	可设置	TIM16	TIM16 全局中断	0x0000_0094
22	29	可设置	TIM17	TIM17 全局中断	0x0000_0098
23	30	可设置	I2C1	I2C1 全局中断	0x0000_009C
24	31	可设置	I2C2	I2C2 全局中断	0x0000_00A0
25	32	可设置	SPI1	SPI1 全局中断	0x0000_00A4
26	33	可设置	SPI2	SPI2 全局中断	0x0000_00A8
27	34	可设置	USART1	USART1 全局中断	0x0000_00AC
28	35	可设置	USART2	USART2 全局中断	0x0000_00B0
29	36	可设置	USART3_4	USART3_4 全局中断	0x0000_00B4

- The grayed cells（the address less than 0x0000 0040） correspond to the Cortex®-M0+ interrupts.

12.2. 外部中断/事件控制器(EXTI)

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

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

EXTI 唤醒请求允许系统从 stop 模式唤醒，中断请求和事件请求也可以在 run 模式使用。

EXTI 允许管理多达 22 个 configurable/direct 事件 line（20 个 configurable 事件 Line 和 2 个 direct 事件 line）。

12.2.1. EXTI 主要特性

- 系统可以通过 GPIO 和指定模块（PVD/COMP/RTC/LPTIM）输入事件唤醒
- Configurable 型事件（来自 I/O，或无状态 pending 位的外设，产生脉冲的外设）
- 可选有效触发沿（上升沿/下降沿）
- 中断挂起标志位
- 独立中断和事件产生屏蔽位
- 可软件触发
- Direct 型事件（具有关联标志和中断 pending 状态位的外设）
- 固定的上升沿触发
- 在 EXTI 模块里没有中断 pending 位
- 独立中断和事件产生屏蔽位
- 无软件触发
  - ✓ IO 端口选择

12.2.2. EXTI 框图

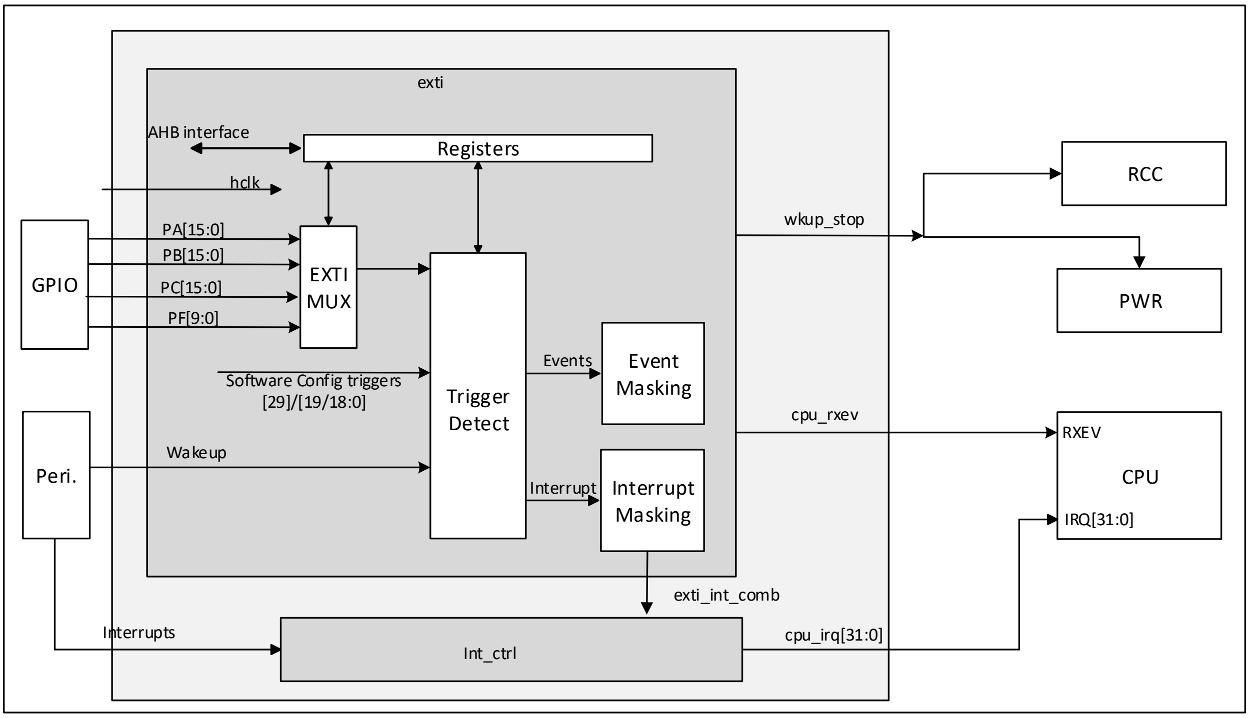


图 12-1 EXTI 框图

12.2.3. 外设和 CPU 的 EXTI 连接

在 stop 模式下能产生唤醒或者中断事件信号的外设，连接至 EXTI 模块。

- 产生一个脉冲的，或者外设内部没有中断状态位的唤醒信号，被连接到 EXTI 模块的 configurable line。此时 EXTI 模块产生一个中断挂起位（该位需要被清零），该 EXTI 中断会作为 CPU 的中断信号。
- 有关联的状态位（该位在外设被清零）的外设的中断和唤醒信号，连接到 EXTI 模块的唤醒触发信号线。
- 所有 GPIO port 输入到 EXTI MUX 模块，通过 configurable 的配置，允许选中后作为系统唤醒信号。

12.2.4. EXTI 可配置事件（configurable）触发唤醒

通过配置 EXTI_SWIER1 寄存器，软件可以触发唤醒功能。

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

CPU 有专用中断屏蔽寄存器和事件屏蔽寄存器。事件使能后产生给 CPU 的事件。所有给 CPU 的事件‘或’运算后输出到 CPU 的唯一事件输入信号 rxev。

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

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

12.2.5. EXTI 直接类型事件输入唤醒

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

12.2.6. EXTI 选择器

GPIO 被用以下方式连接到 16 个外部中断/事件 line 上：

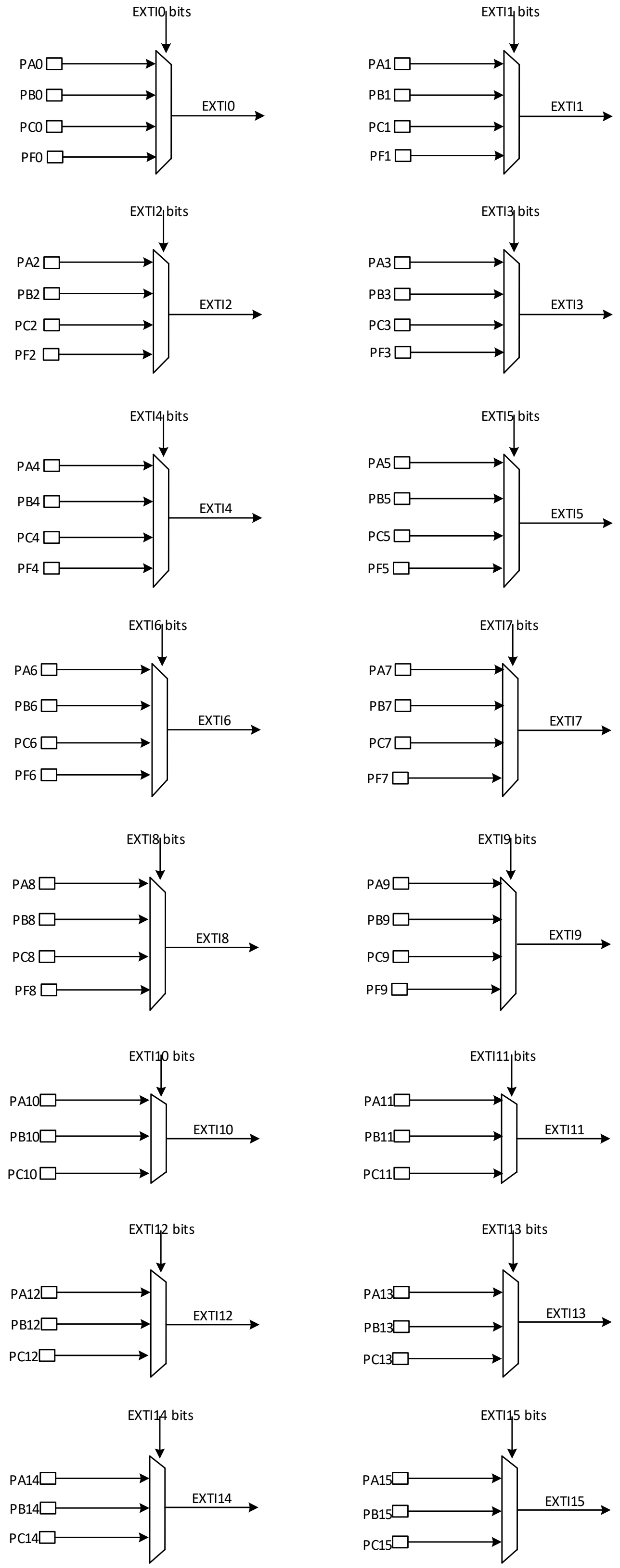


图 12-2 外部中断/事件 GPIO 映像

所有 line 连接内容如下表所示：

EXTI line	Line source	Line type
Line 0-15	GPIO	configurable
Line 16	PVD output	Configurable
Line 17	COMP 1 output	Configurable
Line 18	COMP 2 output	Configurable
Line 19	RTC	Direct
Line 20	Reserved	-
Line 21	Reserved	-
Line 22	Reserved	-
Line 23	Reserved	-
Line 24	Reserved	-
Line 25	Reserved	-
Line 26	Reserved	-
Line 27	Reserved	-
Line 28	Reserved	-
Line 29	LPTIM	Direct
Line 30~33	Reserved	

12.3. EXTI 寄存器

该外设的寄存器可以用 word(32bit)、half-word（16bit）和 byte（8bit）访问。

12.3.1. 上升沿触发选择寄存器 (EXTI_RTSR)

Address offset: 0x00

Reset value: 0x0000 0000

仅包含对 configurable 事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RT20	Res	RT18	RT17	RT16
											RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved			
20	RT20	RW	0	Configurable 类型 EXTI line20 上升沿触发配置。 0：禁止 1：使能
19	Reserved	-	-	-
18	RT18	RW	0	Configurable 类型 EXTI line18 上升沿触发配置。 0：禁止 1：使能
17	RT17	RW	0	Configurable 类型 EXTI line17 上升沿触发配置。 0：禁止 1：使能
16	RT16	RW	0	Configurable 类型 EXTI line16 上升沿触发配置。 0：禁止 1：使能
15	RT15	RW	0	Configurable 类型 EXTI line15 上升沿触发配置。 0：禁止 1：使能
14	RT14	RW	0	Configurable 类型 EXTI line14 上升沿触发配置。

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

configurable line 是边沿触发的，在这些 Line 上不能产生毛刺。如果在写 EXTI_RTSR 寄存器期间，configurable 中断线出现了上升沿，相关的 Pending 位不被置位。

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

12.3.2. 下降沿触发选择寄存器 (EXTI_FTSR)

Address offset: 0x04

Reset value: 0x0000 0000

仅包含对 configurable 事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FT20	Res	FT18	FT17	FT16
											RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	-
20	FT20	RW	0	Configurable 类型 EXTI line20 下降沿触发配置。 0：禁止 1：使能
19	Reserved	-	-	-
18	FT18	RW	0	Configurable 类型 EXTI line18 下降沿触发配置。 0：禁止 1：使能
17	FT17	RW	0	Configurable 类型 EXTI line17 下降沿触发配置。 0：禁止 1：使能
16	FT16	RW	0	Configurable 类型 EXTI line16 下降沿触发配置。 0：禁止 1：使能
15	FT15	RW	0	Configurable 类型 EXTI line15 下降沿触发配置。 0：禁止 1：使能
14	FT14	RW	0	Configurable 类型 EXTI line14 下降沿触发配置。 0：禁止 1：使能
13	FT13	RW	0	Configurable 类型 EXTI line13 下降沿触发配置。 0：禁止 1：使能
12	FT12	RW	0	Configurable 类型 EXTI line12 下降沿触发配置。 0：禁止 1：使能
11	FT11	RW	0	Configurable 类型 EXTI line11 下降沿触发配置。 0：禁止 1：使能
10	FT10	RW	0	Configurable 类型 EXTI line10 下降沿触发配置。 0：禁止 1：使能

Bit	Name	R/W	Reset Value	Function
9	FT9	RW	0	Configurable 类型 EXTI line9 下降沿触发配置。 0：禁止 1：使能
8	FT8	RW	0	Configurable 类型 EXTI line8 下降沿触发配置。 0：禁止 1：使能
7	FT7	RW	0	Configurable 类型 EXTI line7 下降沿触发配置。 0：禁止 1：使能
6	FT6	RW	0	Configurable 类型 EXTI line6 下降沿触发配置。 0：禁止 1：使能
5	FT5	RW	0	Configurable 类型 EXTI line5 下降沿触发配置。 0：禁止 1：使能
4	FT4	RW	0	Configurable 类型 EXTI line4 下降沿触发配置。 0：禁止 1：使能
3	FT3	RW	0	Configurable 类型 EXTI line3 下降沿触发配置。 0：禁止 1：使能
2	FT2	RW	0	Configurable 类型 EXTI line2 下降沿触发配置。 0：禁止 1：使能
1	FT1	RW	0	Configurable 类型 EXTI line1 下降沿触发配置。 0：禁止 1：使能
0	FT0	RW	0	Configurable 类型 EXTI line0 下降沿触发配置。 0：禁止 1：使能

Configurable line 是边沿触发的，在这些 Line 上不能产生毛刺。如果在写 EXTI_FTSR 寄存器期间，configurable line 出现了下降沿，相关的 Pending 位不被置位。

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

12.3.3. 软件中断事件寄存器 (EXTI_SWIER)

Address offset: 0x08

Reset value: 0x0000 0000

仅包含对 configurable 事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SW18	Res	SW18	SW17	SW16
											RW		RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SW15	SW14	SW13	SW12	SW11	SW10	SW9	SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved			
20	SWI20	RW	0	Configurable 类型 EXTI line20 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
19	Reserved			
18	SWI18	RW	0	Configurable 类型 EXTI line18 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
17	SWI17	RW	0	Configurable 类型 EXTI line17 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件自清。读返回 0.
16	SWI16	RW	0	Configurable 类型 EXTI line16 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
15	SWI15	RW	0	Configurable 类型 EXTI line15 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
14	SWI14	RW	0	Configurable 类型 EXTI line14 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件自清。读返回 0.
13	SWI13	RW	0	Configurable 类型 EXTI line13 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件自清。读返回 0.
12	SWI12	RW	0	Configurable 类型 EXTI line12 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
11	SWI11	RW	0	Configurable 类型 EXTI line11 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
10	SWI10	RW	0	Configurable 类型 EXTI line10 软件上升沿触发配置。

Bit	Name	R/W	Reset Value	Function
				0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
9	SWI9	RW	0	Configurable 类型 EXTI line9 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件自清。读返回 0.
8	SWI8	RW	0	Configurable 类型 EXTI line8 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
7	SWI7	RW	0	Configurable 类型 EXTI line7 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
6	SWI6	RW	0	Configurable 类型 EXTI line6 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
5	SWI5	RW	0	Configurable 类型 EXTI line5 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
4	SWI4	RW	0	Configurable 类型 EXTI line4 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
3	SWI3	RW	0	Configurable 类型 EXTI line3 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
2	SWI2	RW	0	Configurable 类型 EXTI line2 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
1	SWI1	RW	0	Configurable 类型 EXTI line1 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断

Bit	Name	R/W	Reset Value	Function
				该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）
0	SWI0	RW	0	Configurable 类型 EXTI line0 软件上升沿触发配置。 0：No effect 1：产生上升沿触发事件，进而产生中断 该位由硬件清零，读返回 0（硬件清零后）或者配置值（硬件清零前）

12.3.4. 挂起寄存器(EXTI_PR)

Address offset: 0x0C

Reset value: undefined

仅包含对 configurable 事件的寄存器控制位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PR20	Res	PR18	PR17	PR16
											rc_w1		rc_w1	rc_w1	rc_w1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	-
20	PR20	RC_W1	0	Configurable 类型 EXTI line20 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
19	Reserved	-	-	-
18	PR18	RC_W1	0	Configurable 类型 EXTI line18 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
17	PR17	RC_W1	0	Configurable 类型 EXTI line17 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
16	PR16	RC_W1	0	Configurable 类型 EXTI line16 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
15	PR15	RC_W1	0	Configurable 类型 EXTI line15 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。

Bit	Name	R/W	Reset Value	Function
				0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
14	PR14	RC_W1	0	Configurable 类型 EXTI line14 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
13	PR13	RC_W1	0	Configurable 类型 EXTI line13 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
12	PR12	RC_W1	0	Configurable 类型 EXTI line12 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
11	PR11	RC_W1	0	Configurable 类型 EXTI line11 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
10	PR10	RC_W1	0	Configurable 类型 EXTI line10 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
9	PR9	RC_W1	0	Configurable 类型 EXTI line9 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
8	PR8	RC_W1	0	Configurable 类型 EXTI line8 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
7	PR7	RC_W1	0	Configurable 类型 EXTI line7 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
6	PR6	RC_W1	0	Configurable 类型 EXTI line6 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求；

Bit	Name	R/W	Reset Value	Function
				1：产生上升沿/下降沿/软件触发事件请求；
5	PR5	RC_W1	0	Configurable 类型 EXTI line5 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
4	PR4	RC_W1	0	Configurable 类型 EXTI line4 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
3	PR3	RC_W1	0	Configurable 类型 EXTI line3 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
2	RPIF2	RC_W1	0	Configurable 类型 EXTI line2 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
1	PR1	RC_W1	0	Configurable 类型 EXTI line1 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；
0	PR0	RC_W1	0	Configurable 类型 EXTI line0 事件挂起标志。软件或者硬件产生上升沿/下降沿触发事件时，该位置位。软件写 1 清零。 0：未产生事件请求； 1：产生上升沿/下降沿/软件触发事件请求；

12.3.5. 外部中断选择寄存器 1 (EXTI_EXTICR1)

Address offset:0x60

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI3[1:0]	RW	0	EXTI3 对应 GPIO port 选择。 2'b00: PA[3] pin 2'b01: PB[3] pin 2'b10: PC[3] pin 2'b11: PF[3] pin

Bit	Name	R/W	Reset Value	Function
23:18	Reserved	-	-	Reserved
17:16	EXTI2[1:0]	RW	0	EXTI2 对应 GPIO port 选择。 2’b00: PA[2] pin 2’b01: PB[2] pin 2’b10: PC[2] pin 2’b11: PF[2] pin
15:10	Reserved	-	-	Reserved
9:8	EXTI1[1:0]	RW	0	EXTI1 对应 GPIO port 选择。 2’b00: PA[1] pin 2’b01: PB[1] pin 2’b10: PC[1] pin 2’b11: PF[1] pin
7:2	Reserved	-	-	Reserved
1:0	EXTI0[1:0]	RW	0	EXTI0 对应 GPIO port 选择。 2’b00: PA[0] pin 2’b01: PB[0] pin 2’b10: PC[0] pin 2’b11: PF[0] pin

12.3.6. 外部中断选择寄存器 2 (EXTI_EXTICR2)

Address offset:0x64

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI7[1:0]	RW	0	EXTI7 对应 GPIO port 选择。 2’b00: PA[7] pin 2’b01: PB[7] pin 2’b10: PC[7] pin 2’b11: PF[7] pin
23:18	Reserved	-	-	Reserved
17:16	EXTI6[1:0]	RW	0	EXTI6 对应 GPIO port 选择。 2’b00: PA[6] pin 2’b01: PB[6] pin 2’b10: PC[6] pin 2’b11: PF[6] pin
15:10	Reserved	-	-	Reserved
9:8	EXTI5[1:0]	RW	0	EXTI5 对应 GPIO port 选择。 2’b00: PA[5] pin 2’b01: PB[5] pin 2’b10: PC[5] pin 2’b11: PF[5] pin
7:2	Reserved	-	-	Reserved
1:0	EXTI4[1:0]	RW	0	EXTI4 对应 GPIO port 选择。 2’b00: PA[4] pin 2’b01: PB[4] pin 2’b10: PC[4] pin 2’b11: PF[4] pin

12.3.7. 外部中断选择寄存器 3 (EXTI_EXTICR3)

Address offset： 0x68

Reset value： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	EXTI11[1:0]		Res	Res	Res	Res	Res	Res	EXTI10[1:0]	
						RW	RW							RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	EXTI9[1:0]		Res	Res	Res	Res	Res	Res	EXTI8[1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI11[1:0]	RW	0	EXTI11 对应 GPIO port 选择。 2’b00: PA[11] pin 2’b01: PB[11] pin 2’b10: PC[11] pin 2’b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI10[1:0]	RW	0	EXTI11 对应 GPIO port 选择。 2’b00: PA[10] pin 2’b01: PB[10] pin 2’b10: PC[10] pin 2’b11: reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI9[1:0]	RW	0	EXTI11 对应 GPIO port 选择。 2’b00: PA[9] pin 2’b01: PB[9] pin 2’b10: PC[9] pin 2’b11: PF[9] pin
7:2	Reserved	-	-	Reserved
1:0	EXTI8[1:0]	RW	0	EXTI8 对应 GPIO port 选择。 2’b00: PA[8] pin 2’b01: PB[8] pin 2’b10: PC[8] pin 2’b11: PF[8] pin

12.3.8. 外部中断选择寄存器 4 (EXTI_EXTICR4)

Address offset:0x6C

Reset value:0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	EXTI15[1:0]		Res	Res	Res	Res	Res	Res	EXTI14[1:0]	
						RW	RW							RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	EXTI13[1:0]		Res	Res	Res	Res	Res	Res	EXTI12[1:0]	
						RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	Reserved
25:24	EXTI15[1:0]	RW	0	EXTI15 对应 GPIO port 选择。 2’b00: PA[15] pin 2’b01: PB[15] pin 2’b10: PC[15] pin 2’b11: reserved
23:18	Reserved	-	-	Reserved
17:16	EXTI14[1:0]	RW	0	EXTI14 对应 GPIO port 选择。 2’b00: PA[14] pin 2’b01: PB[14] pin 2’b10: PC[14] pin 2’b11: reserved
15:10	Reserved	-	-	Reserved
9:8	EXTI13[1:0]	RW	0	EXTI13 对应 GPIO port 选择。 2’b00: PA[13] pin 2’b01: PB[13] pin 2’b10: PC[13] pin

Bit	Name	R/W	Reset Value	Function
				2’b11: reserved
7:2	Reserved	-	-	Reserved
1:0	EXTI12[1:0]	RW	0	EXTI12 对应 GPIO port 选择。 2’b00: PA[12] pin 2’b01: PB[12] pin 2’b10: PC[12] pin 2’b11: reserved

12.3.9. 中断屏蔽寄存器（EXTI_IMR）

Address offset:0x80

Reset value:0x2008 0000

注意： Direct 类型 line 的中断 mask bit 默认为 1，即允许该 line；configurable line 的 mask 位，默认为 0，即屏蔽该 line。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	IM29	Res	Res	Res	Res	Res	Res	Res	Res	IM20	IM19	IM18	IM17	IM16
		RW									RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved			
29	IM29	RW	1	EXTI line29 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
28:21	Reserved			
20	IM20	RW	1	EXTI line20 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
19	IM19	RW	1	EXTI line19 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
18	IM18	RW	0	EXTI line18 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
17	IM17	RW	0	EXTI line17 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
16	IM16	RW	0	EXTI line16 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
15	IM15	RW	0	EXTI line15 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
14	IM14	RW	0	EXTI line14 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
13	IM13	RW	0	EXTI line13 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽

Bit	Name	R/W	Reset Value	Function
				1：中断唤醒未屏蔽
12	IM12	RW	0	EXTI line12 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
11	IM11	RW	0	EXTI line11 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
10	IM10	RW	0	EXTI line10 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
9	IM9	RW	0	EXTI line9 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
8	IM8	RW	0	EXTI line8 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
7	IM7	RW	0	EXTI line7 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
6	IM6	RW	0	EXTI line6 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
5	IM5	RW	0	EXTI line5 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
4	IM4	RW	0	EXTI line4 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
3	IM3	RW	0	EXTI line3 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
2	IM2	RW	0	EXTI line2 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
1	IM1	RW	0	EXTI line1 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽
0	IM0	RW	0	EXTI line0 作为中断唤醒 CPU 屏蔽控制。 0：中断唤醒屏蔽 1：中断唤醒未屏蔽

12.3.10. 事件屏蔽寄存器(EXTI_EMR)

Address offset: 0x84

Reset value: 0x0000 0000

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

Res	Res	EM29	Res	Res	Res	Res	Res	Res	Res	Res	EM20	EM19	EM18	EM17	EM16
		RW									RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved			
29	EM29	RW	0	EXTI line29 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
28:21	Reserved			
20	EM20	RW	0	EXTI line20 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
19	EM19	RW	0	EXTI line19 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
18	EM18	RW	0	EXTI line18 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
17	EM17	RW	0	EXTI line17 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
16	EM16	RW	0	EXTI line16 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
15	EM15	RW	0	EXTI line15 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
14	EM14	RW	0	EXTI line14 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
13	EM13	RW	0	EXTI line13 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
12	EM12	RW	0	EXTI line12 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
11	EM11	RW	0	EXTI line11 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
10	EM10	RW	0	EXTI line10 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
9	EM9	RW	0	EXTI line9 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽

Bit	Name	R/W	Reset Value	Function
				1：事件唤醒未屏蔽
8	EM8	RW	0	EXTI line8 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
7	EM7	RW	0	EXTI line7 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
6	EM6	RW	0	EXTI line6 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
5	EM5	RW	0	EXTI line5 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
4	EM4	RW	0	EXTI line4 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
3	EM3	RW	0	EXTI line3 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
2	EM2	RW	0	EXTI line2 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
1	EM1	RW	0	EXTI line1 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽
0	EM0	RW	0	EXTI line0 作为事件唤醒 CPU 屏蔽控制。 0：事件唤醒屏蔽 1：事件唤醒未屏蔽

12.3.11. EXTI 寄存器映像

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_RTSR	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RT20		RT18	RT17	RT16	RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	EXTI_FTSR	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FT20		FT18	FT17	FT16	FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0x08	EXTI_SWIER	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWI20		SWI18	SWI17	SWI16	SWI15	SWI14	SWI13	SWI12	SWI11	SWI10	SWI9	SWI8	SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	EXTI_PR	Res.	Res.		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR20		PR18	PR17	PR16	PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
	Reset value												0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	EXTI_EXTICR1	Res.	Res.	Res.	Res.	Res.	Res.	EXTI3[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI2[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI1[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI0[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x64	EXTI_EXTICR2	Res.	Res.	Res.	Res.	Res.	Res.	EXTI7[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI6[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI5[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI4[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x68	EXTI_EXTICR3	Res.	Res.	Res.	Res.	Res.	Res.	EXTI11[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI10[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI10[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI8[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x6C	EXTI_EXTICR4	Res.	Res.	Res.	Res.	Res.	Res.	EXTI15[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI14[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI12[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	EXTI11[1:0]	
	Reset value							0	0							0	0							0	0							0	0
0x84	EXTI_EMR	Res.	Res.	EM29	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EM20	EM19	EM18	EM17	EM16	EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
	Reset value			0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.循环冗余校验(CRC)

13.1. 简介

根据生成多项式，CRC 计算单元将输入的 32 位数据，运算产生一个 CRC 结果。

13.2. CRC 主要特点

- 使用 CRC-32（以太网）多项式：0x4C11DB7
  - $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$
- 支持 32 位数据输入
- 单个输入/输出 32 数据和结果输出共用一个寄存器
- 通用 8 位寄存器（可用于存放临时数据）
- 计算时间：32bits 数据 4 个 AHB 时钟

13.3. CRC 功能描述

13.3.1. CRC 框图

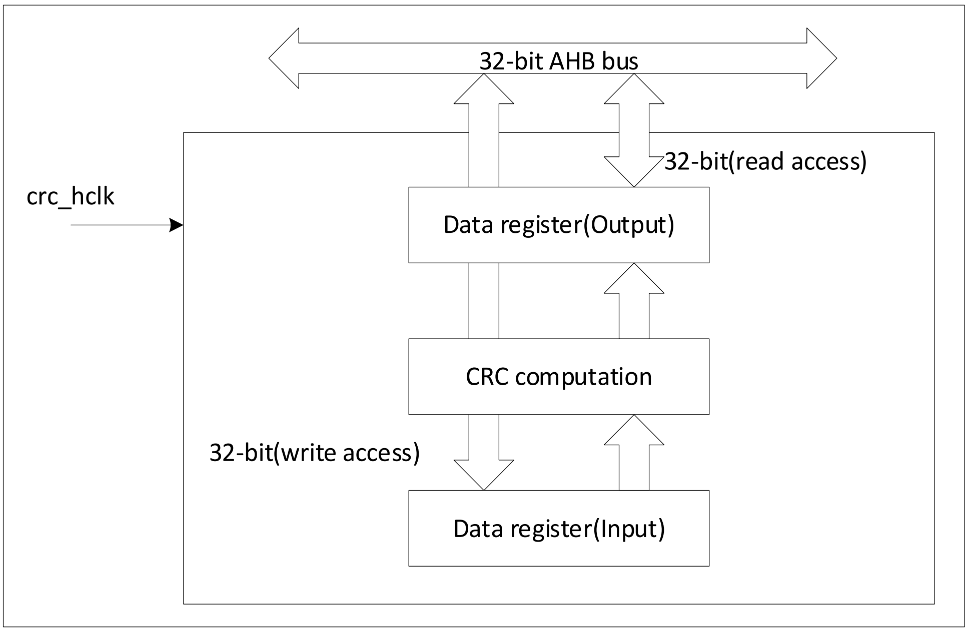


图 13-1 CRC 计算单元框图

CRC 计算单元含有 1 个 32 位数据寄存器：

- 对该寄存器进行写操作时，作为输入寄存器，可以输入要进行 CRC 计算的新数据。
- 对该寄存器进行读操作时，返回上一次 CRC 计算的结果。

每一次写入数据寄存器，其计算结果是前一次 CRC 计算结果和新计算结果的组合(对整个 32 位字进行 CRC 计算，而不是逐字节地计算)。

支持配置 CRC 初始值。

可以通过设置寄存器 CRC_CR 的 RESET 位来重置寄存器 CRC_DR 为 0xFFFF FFFF。该操作不影响寄存器 CRC_IDR 内的数据。

13.4. CRC 寄存器

13.4.1. 数据寄存器 (CRC_DR)

Address offset:0x00

Reset value:0xFFFF FFFF

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

Bit	Name	R/W	Reset Value	Function
31:0	DR	RW	0xFFFFFFFF	数据寄存器。 当写新数据时，作为输入寄存器。当被读时，保持之前 CRC 计算结果。

13.4.2. 独立数据寄存器(CRC_IDR)

Address offset:0x04

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	RES	-	保留
7:0	IDR[7:0]	RW	0	通用 8 位数据寄存器位。 可用于临时存放 1 字节数据。 寄存器 CRC_CR 的 RESET 位产生的 CRC 复位对本寄存器无影响。 注：此寄存器不参与 CRC 计算，可以存放任何数据。

13.4.3. 控制寄存器(CRC_CR)

Address offset:0x08

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	RES	-	保留

Bit	Name	R/W	Reset Value	Function
0	RESET	W	0	软件置位后将会复位 CRC 模块，数据寄存器会加载 CRC_INIT 寄存器的值。软件只能写 1，由硬件清零。

13.4.4. CRC 寄存器映像

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	CRC_DR	DR[31:0]																															
	Reset value	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	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IDR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x08	CRC_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RESET
	Reset value																																0

14. 模拟/数字转换(ADC)

14.1. 简介

12 位 ADC 是一种逐次逼近型模拟数字转换器。它有多达 24 个通道，可测量 16 个外部和 5 个内部信号源。各通道的 A/D 转换可以单次、连续、扫描或间断模式执行。 ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阀值。

14.2. ADC 主要特性

- 高性能
  - 12bit、10bit、8bit 和 6bit 分辨率可配置
  - ADC 转换时间：1us@12bit（1MHz）
  - 自校准
  - 可编程的采样时间
  - 可编程的数据对齐模式
  - 规则组支持 DMA
- 模拟输入通道
  - 16 个外部模拟输入通道
  - 1 个内部温度传感器通道（T_{SENSOR}）
  - 1 个内部参考电压通道（V_{REFINT}）
  - 1 个内部参考电压输入通道（VERF Buffer）
  - 2 个内部 OPA 输入电压通道（OPA）
- 转换操作启动可以通过
  - 软件启动
  - 硬件启动（TIM1、TIM2、TIM3、TIM15 或者 GPIO）
- 转换模式
  - 单次模式：可以转换 1 个单通道
  - 扫描模式：可以扫描一系列通道
  - 连续模式：连续转换被选择的通道
  - 间断模式：每次触发转换子序列通道，多次触发直到完整序列被转换
- 中断产生
  - 在转换结束
  - 模拟看门狗事件
- 模拟看门狗
- ADC 供电要求：1.7 V 到 5.5 V
- ADC 输入范围：$V_{REF-} \leq V_{IN} \leq V_{REF+}$

14.3. ADC 功能描述

14.3.1. ADC 框图

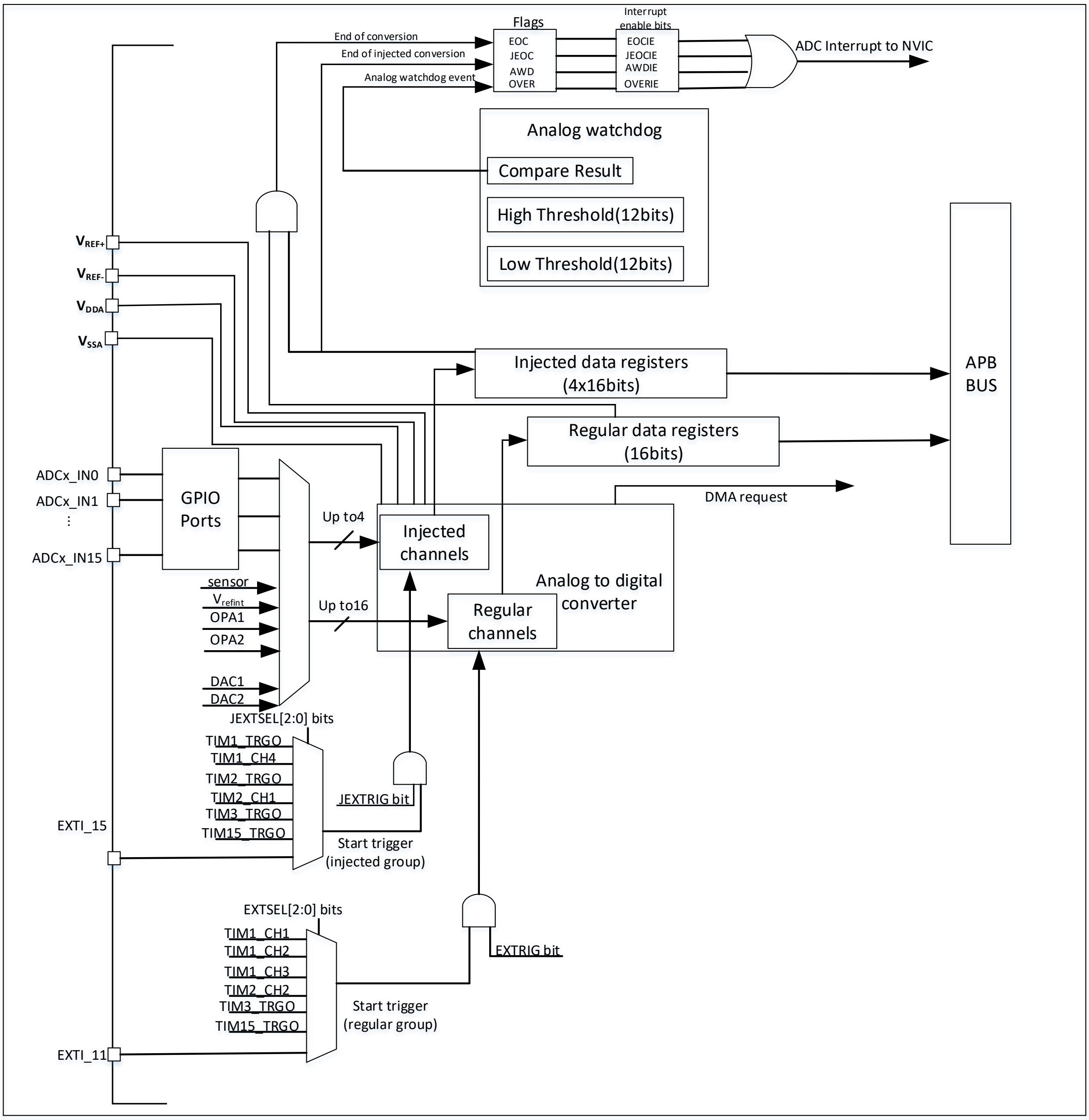


图 14-1 ADC 框图

14.3.2. 校准

该 ADC 具有软件校准功能。在校准期间，ADC 计算一个用于 ADC 内部的校准因子（ADC 断电后丢失）。在 ADC 校准期间、未完成校准前，应用不能使用 ADC 模块。

在使用 ADC 转换前，要进行校准操作。校准用于消除芯片和芯片之间的，由于工艺变化引起的 offset error。

软件设置 CAL=1 可启动校准，校准只能在 ADC 未使能时 (ADON=0)启动，且仅支持选择系统时钟作为 ADC 的时钟。当校准完成后，CAL 被硬件清 0。

当 ADC 的工作条件发生改变时（VCC 改变是 ADC offset 偏移的主要因素，温度改变次之），推荐进行再次校准操作。

校准的软件操作过程：

- 确认 ADON=0
- 设置 CAL=1
- 等待到 CAL=0

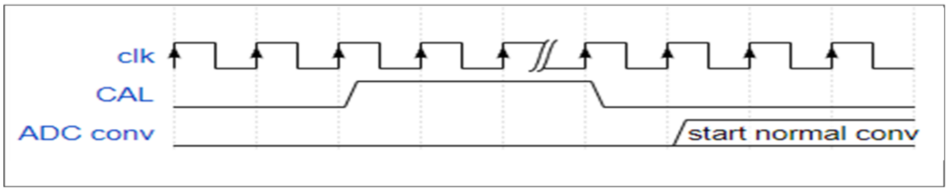


图 14-1 ADC 校准时序图

14.3.3. ADC 开关控制

通过设置 ADC_CR2 寄存器的 ADON 位可给 ADC 上电。当第一次设置 ADON 位时，它将 ADC 从断电状态下唤醒。

ADC 上电延迟一段时间后(tSTAB，不少于 20us)开始进行转换。

为了省电，当 ADON 为 0 时， ADC 模拟子模块将会进入掉电模式。通过清除 ADON 位可以停止转换，并将 ADC 置于断电模式。

14.3.4. ADC 时钟

由 RCC 控制提供的 ADCCLK 时钟和 PCLK(APB 时钟)同步。RCC 控制器(CLK 控制器)为 ADC 时钟提供一个专用的可编程预分频器，ADCCLK 时钟源有 RCC 可配来自 PCLK 的分频，相见 RCC_ADC_DIV。

14.3.5. 通道选择

有 16 个外部通道和 5 个内部通道，其中内部通道有：

温度传感器/V_{REFINT} 内部通道

温度传感器和通道 ADC1_IN16 相连接，内部参照电压 V_{REFINT} 和 ADC1_IN17 相连接。

VCC/3

VCC/3 和 通道 ADC1_IN18 相连接。

OPA

OPA1_VIN 和通道 ADC1_IN21 相连接，OPA2_VIN 和通道 ADC1_IN22 相连接。

可以把转换组织成两组：规则组和注入组。在任意多个通道上以任意顺序进行的一系列转换构成成组转换。例如，可以如下顺序完成转换：通道 3、通道 8、通道 2、通道 2、通道 0、通道 2、通道 2、通道 15。

规则组由多达 16 个转换组成。规则通道和它们的转换顺序在 ADC_SQRx 寄存器中选择。规则组中转换的总数应写入 ADC_SQR1 寄存器的 L[3:0]位中。

注入组由多达 4 个转换组成。注入通道和它们的转换顺序在 ADC_JSQR 寄存器中选择。注入组里的转换总数目应写入 ADC_JSQR 寄存器的 JL[1:0]位中。

如果 ADC_SQRx 或 ADC_JSQR 寄存器在转换期间被更改，当前的转换被清除，一个新的启动脉冲将发送到 ADC 以转换新选择的组。

14.3.6. 可编程采样时间

ADC 使用若干个 ADC_CLK 周期对输入电压采样，采样周期数可以通过 ADC_SMPR1、ADC_SMPR2 和 ADC_SMPR3 寄存器中的 SMP[2:0]位更改。每个通道可以分别用不同的时间采样。

总转换时间如下计算：

T_{CONV} = 采样时间+ 12.5 个周期

例如：

当 ADCCLK=14MHz，采样时间为 3.5 周期

T_{CONV} = 3.5 + 12.5 = 16 周期 = 1.14μs

14.3.7. 可配置的分辨率

可通过降低 ADC 分辨率来执行快速转换。ADC_CR1 寄存器的 RESSEL 位用于选择数据寄存器中可用的位数。每种分辨率的最小转换时间如下,采样时间+转换时间：

- 12 位： 3.5 + 12.5 = 16 ADCCLK 周期
- 10 位： 3.5 + 10.5 = 14 ADCCLK 周期
- 8 位： 3.5 + 8.5 = 12 ADCCLK 周期
- 6 位： 3.5 + 6.5 = 10 ADCCLK 周期

14.3.8. 单次转换模式

单次转换模式下， ADC 只执行一次转换。该模式既可通过设置 ADC_CR2 寄存器的 ADON 位(只适用于规则通道)启动也可通过外部触发启动(适用于规则通道或注入通道)，这时 CONT 位为 0。

一旦选择通道的转换完成：

- 如果一个规则通道被转换：
  - 转换数据被储存在 16 位 ADC_DR 寄存器中
  - EOC(转换结束)标志被设置
  - 如果设置了 EOCIE，则产生中断。
- 如果一个注入通道被转换：
  - 转换数据被储存在 16 位的 ADC_JDRx 寄存器中
  - JEOC(注入转换结束)标志被设置
  - 如果设置了 JEOCIE 位，则产生中断。

然后 ADC 停止。

14.3.9. 连续转换模式

在连续转换模式中，当前面 ADC 转换一结束马上就启动另一次转换。此模式可通过外部触发启动或通过设置 ADC_CR2 寄存器上的 ADON 位启动，此时 CONT 位是 1。

每个转换后：

- 如果一个规则通道被转换：
  - 转换数据被储存在 16 位的 ADC_DR 寄存器中
  - EOC(转换结束)标志被设置
  - 如果设置了 EOCIE，则产生中断。

- 如果一个注入通道被转换：
  - EOC (转换结束标志)标志置位
- 转换数据被储存在 16 位的 ADC_JDRx 寄存器中
- JEOC(注入转换结束)标志被设置
- 如果设置了 JEOCIE 位，则产生中断。

14.3.10. 扫描模式

此模式用来扫描一组模拟通道。

扫描模式可通过设置 ADC_CR1 寄存器的 SCAN 位来选择。一旦这个位被设置， ADC 扫描所有被 ADC_SQRx 寄存器(对规则通道)或 ADC_JSQR(对注入通道)选中的所有通道。在每个组的每个通道上执行单次转换。在每个转换结束时，同一组的下一个通道被自动转换。如果设置了 CONT 位，转换不会在选择组的最后一个通道上停止，而是再次从选择组的第一个通道继续转换。

如果设置了 DMA 位，DMA 控制器把规则组通道的转换数据传输到 SRAM 中。而注入通道转换的数据总是存储在 ADC_JDRx 寄存器中。

14.3.11. 间断转换模式

规则组

此模式通过设置 ADC_CR1 寄存器上的 DISCEN 位激活。它可以用来执行一个短序列的 n 次转换(n<=8)，此转换是 ADC_SQRx 寄存器所选择的转换序列的一部分。数值 n 由 ADC_CR1 寄存器的 DISCNUM[2:0]位给出。一个外部触发信号可以启动 ADC_SQRx 寄存器中描述的下一轮 n 次转换，直到此序列所有的转换完成为止。总的序列长度由 ADC_SQR1 寄存器的 L[3:0]定义。

举例：

n=3，被转换的通道 = 0、 1、 2、 3、 6、 7、 9、 10

第一次触发：转换的序列为 0、 1、 2

第二次触发：转换的序列为 3、 6、 7

第三次触发：转换的序列为 9、 10，并产生 EOC 事件

第四次触发：转换的序列 0、 1、 2

注意： 当以间断模式转换一个规则组时，转换序列结束后不自动从头开始。

当所有子组被转换完成，下一次触发启动第一个子组的转换。在上面的例子中，第四次触发重新转换第一子组的通道 0、 1 和 2。

注入组

此模式通过设置 ADC_CR1 寄存器的 JDISCEN 位激活。在一个外部触发事件后，该模式按通道顺序逐个转换 ADC_JSQR 寄存器中选择的序列。

一个外部触发信号可以启动 ADC_JSQR 寄存器选择的下一个通道序列的转换，直到序列中所有的转换完成为止。总的序列长度由 ADC_JSQR 寄存器的 JL[1:0]位定义。

例子：

n=1，被转换的通道 = 1、 2、 3

第一次触发：通道 1 被转换

第二次触发：通道 2 被转换

第三次触发：通道 3 被转换，并且产生 EOC 和 JEOC 事件

第四次触发：通道 1 被转换

注意： 1 当完成所有注入通道转换，下个触发启动第 1 个注入通道的转换。在上述例子中，第四个触发重新转换第 1 个注入通道 1。

2 不能同时使用自动注入和间断模式。

14.3.12. 注入通道管理

注入通道外部触发优先级高于规则通道外部触发，即注入通道外部触发可以中断正在进行中的规则通道转换。注入通道有两种中断方式：触发注入和自动注入。

14.3.12.1. 触发注入

若清除 ADC_CR1 寄存器的 JAUTO 位，并且设置 SCAN 位。

- 利用外部触发或通过设置 ADC_CR2 寄存器的 ADON 位，启动一组规则通道的转换。
- 如果在规则通道转换期间产生一外部注入触发，当前转换被复位，注入通道序列被以单次扫描方式进行转换。
- 然后，恢复上次被中断的规则组通道转换。如果在注入转换期间产生一规则事件，注入转换不会被中断，但是规则序列将在注入序列结束后被执行。

注： 当使用触发的注入转换时，必须保证触发事件的间隔长于注入序列。例如：序列长度为 28 个 ADC 时钟周期(即 2 个具有 1.5 个时钟间隔采样时间的转换)，触发之间最小的间隔必须是 29 个 ADC 时钟周期。

14.3.12.2. 自动注入

如果设置了 JAUTO 位，在规则组通道之后，注入组通道被自动转换。这可以用来转换在 ADC_SQRx 和 ADC_JSQR 寄存器中设置的多至 20 个转换序列。

在此模式里，必须禁止注入通道的外部触发。

如果除 JAUTO 位外还设置了 CONT 位，规则通道至注入通道的转换序列被连续执行。

注意： 不可能同时使用自动注入和间断模式

14.3.13. 停止进行中的转换(ADSTP)

用软件设置 ADC_CR 寄存器中的 ADSTP=1 可以停上当前正在进行的转换，复位 ADC 的操作并让 ADC 进入空闲状态，为下次转换作好准备。

当 ADSTP 由软件设置为 1，任何当前的转换中止且转换结果丢弃(ADC_DR 寄存器不用当前的转换值进行更新)。

扫描序列也被中止并复位 (即重新启动 ADC 时会用新的序列进行转换)

一旦结束该过程 ADSTP 和 ADSTART 位都由硬件清 0。

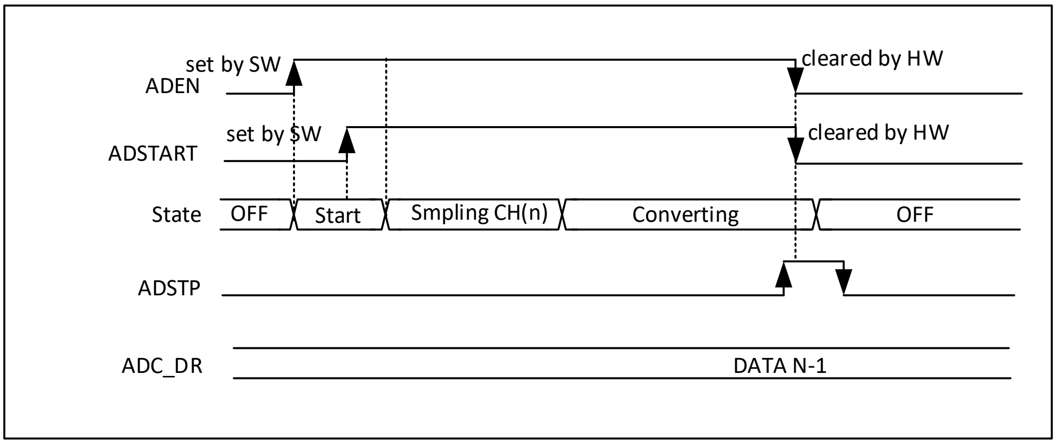


图 14-2 Stop timing

14.4. 模拟看门狗

如果 ADC 转换的模拟电压低于阈值下限或高于阈值上限，则 AWD 模拟看门狗状态位会置 1。这些阈值在 ADC_HTR 和 ADC_LTR 寄存器的 12 个最低有效位中设置。通过设置 ADC_CR1 寄存器中的 AWDIE 位产生中断。

阈值与 ADC_CR2 寄存器中的 ALIGN 位的所选对齐方式无关。阈值比较是在对齐之前完成的（注入通道减去偏移值之前）。

通过配置 ADC_CR1 寄存器，模拟看门狗可以作用于 1 个或多个通道，具体如下表所示：

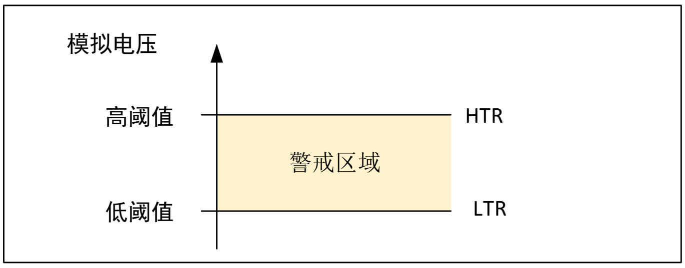


图 14-3 模拟看门狗保护区

表 14-2 模拟看门狗通道选择

模拟看门狗保护通道	ADC_CR1 寄存器控制位		
	AWDSGL	AWDEN	JAWDEN
无	X	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有注入和规则通道	0	1	1
单一注入通道	1	0	1
单一规则通道	1	1	0

模拟看门狗保护通道	ADC_CR1 寄存器控制位		
	AWDSGL	AWDEN	JAWDEN
单一注入或规则通道	1	1	1

14.5. 外部触发转换

可以通过外部事件（例如定时器捕获、 EXTI 中断）触发转换。如果设置了 EXTTRIG 或 JEXTTRIG 控制位，则外部事件就能够触发转换。EXTSEL[2:0]和 JEXTSEL2:0]控制位允许应用程序选择 8 个可能的事件中的某一个，可以触发规则和注入组的采样。

注：当外部触发信号被选为 ADC 规则或注入转换时，只有上升沿可以启动转换。

下表给出了转换可能的外部触发。软件源触发事件可由设置 ADC_CR 寄存器中的 ADSTART 位来产生。

表 14-3 ADC 用于规则通道的外部触发

触发源	类型	EXTSEL[2:0]
定时器 1 的 CH1 输出	片上定时器的内部信号	000
定时器 1 的 CH2 输出		001
定时器 1 的 CH3 输出		010
定时器 2 的 TRGO 输出		011
定时器 3 的 TRGO 输出		100
定时器 15 的 TRGO 输出		101
EXTI 线 11	外部管脚	110
SWSTART	软件控制位	111

表 14-4 ADC 用于注入通道的外部触发

触发源	类型	JEXTSEL[2:0]
定时器 1 的 TRGO 输出	片上定时器的内部信号	000
定时器 1 的 CH4 输出		001
定时器 2 的 TRGO 输出		010
定时器 2 的 CH1 输出		011
定时器 3 的 CH4 输出		100
定时器 15 的 TRGO 输出		101
EXTI 线 15 输出	外部管脚	110

触发源	类型	JEXTSEL[2:0]
JSWSTART	软件控制位	111

14.6. 数据对齐

在每次转换结束(当 EOC 事件产生时)，转换的结果数据被存放到 16 位宽 ADC_DR 数据寄存器中。ADC_CR2 寄存器中的 ALIGN 位选择转换后数据储存的对齐方式。数据可以左对齐或右对齐，如注入组通道转换的数据值已经减去了在 ADC_JOFRx 寄存器中定义的偏移量，因此结果可以是一个负值。 SEXT 位是扩展的符号值。

对于规则组通道，不需减去偏移值，因此只有 12 个位有效。

数据右对齐示例如下：

注入组

SEXT	SEXT	SEXT	SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

规则组

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

数据左对齐示例如下：

注入组

SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0

规则组

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

14.7. 数据过载

ADC 过冲标志(OVER) 是指一个缓冲区过冲事件，规则组当转换好的数据未被 CPU 或 DMA 及时读取时，另一个转换数据已经有效时，就发生了 ADC 过冲。

14.8. DMA 请求

因为规则通道转换的值储存在一个仅有的数据寄存器中，所以当转换多个规则通道时需要使用 DMA，这可以避免丢失已经存储在 ADC_DR 寄存器中的数据。

只有在规则通道的转换结束时才产生 DMA 请求，并将转换的数据从 ADC_DR 寄存器传输到用户指定的目的地址。

14.9. 温度传感器和内部参考电压

温度传感器可用于测量器件周围的温度 (TA)。温度传感器内部连接到 ADC1_IN16 通道，此通道把传感器输出的电压转换成数字值。温度传感器模拟输入推荐采样时间是 17.1us。不使用时，可将传感器置于掉电模式。

温度传感器输出电压随温度线性变化，由于生产过程的变化，温度变化曲线的偏移在不同芯片上会有不同(最多相差 45°C)。内部温度传感器更适合于检测温度的变化，而不是测量绝对的温度。如果需要测量精确的温度，应该使用一个外置的温度传感器。

注意：必须设置 TSVREFE 位激活内部通道： ADC1_IN16(温度传感器)和 ADC1_IN17(VREFINT)的转换。

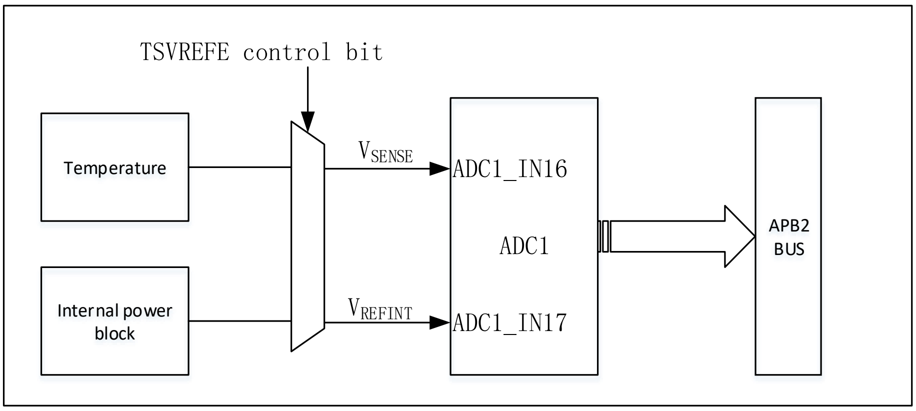


图 14-4 温度传感器通道框图

读温度主要特性

- 支持的温度范围： -40 °C 到 125 °C
- 精度： ±1.5 °C

读温度

要使用传感器，请执行以下操作：

1. 选择 ADC1_IN16
2. 选择一个采样时间，该采样时间要大于数据手册中所指定的最低采样时间。
3. 在 ADC_CR2 寄存器中将 TSVREFE 位置 1，以便将温度传感器从掉电模式中唤醒。
4. 通过将 ADON 位置 1，并使能外部触发，开始 ADC 转换
5. 读取 ADC 数据寄存器中生成的 VSENSE 数据
6. 使用以下公式计算温度：

   Temperature(in ℃)=(85℃-25℃)/(TS_CAL2-TS_CAL1)×(TS_DATA-TS_CAL1)+25℃

   TS_CAL2 代表 85℃温度传感器的校准值，校准值存放地址：0x1FFF 3230

   TS_CAL1 代表 25℃温度传感器的校准值，校准值存放地址：0x1FFF 3228

   TS_DATA 是 ADC 转换的实际输出值

注意： 传感器从掉电模式中唤醒需要一个建立时间，启动时间过后输出正确水平的 VSENSE。 ADC 在上电后同样需要一个建立时间，因此为了缩短延时，应同时设置 ADON 和 TSVREFE。

14.10. ADC 中断

规则和注入组转换结束时能产生中断，当模拟看门狗状态位被设置时能产生中断，当规则组转换数据未被及时读取时也能产生中断。它们都有独立的中断使能位。

ADC_SR 寄存器中有 2 个其他标志，但是它们没有相关联的中断：

- JSTRT(注入组通道转换的启动)
- STRT(规则组通道转换的启动)

表 14-5 ADC 中断

中断事件	事件标志	使能控制
规则组转换结束	EOC	EOCIE
注入组转换结束	JEOC	JEOCIE
设置了模拟看门狗状态位	AWD	AWDIE
溢出标志	OVER	OVERIE

14.11. ADC 寄存器

14.11.1. ADC 状态寄存器 (ADC_SR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVER	STRT	JSTRT	JEOC	EOC	AWD
										RC	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
5	OVER	RC	0	ADC 过载 当过载发生时，硬件置位该位。当 EOC 标志已置起表明一次新的转换已完成。 0：无过载发生（DMA 或 CPU 已读取上一次转换结果） 1：过载已发生
4	STRT	RC_W0	0	规则通道开始状态位 该位由硬件在规则通道转换开始时设置，由软件清除。 0：规则通道转换未开始 1：规则通道转换已开始
3	JSTRT	RC_W0	0	注入通道开始状态位 该位由硬件在注入通道组转换开始时设置，由软件清除。 0：注入通道转换未开始 1：注入通道转换已开始
2	JEOC	RC_W0	0	注入通道转换结束状态位 该位由硬件在所有注入通道组转换结束时设置，由软件清除。 0：转换未完成 1：转换完成
1	EOC	RC_W0	0	转换结束状态位 该位由硬件在(规则或注入)通道组转换结束时设置，由软件清除或由读 ADC_DR 时清除。 0：转换未完成 1：转换完成
0	AWD	RC_W0	0	模拟看门狗标志位

Bit	Name	R/W	Reset Value	Function
				该位由硬件在转换的电压值超出了 ADC_LTR 或低于 ADC_HTR 寄存器定义的范围时设置 1，由软件清除。 0：没有发生模拟看门狗事件 1：发生模拟看门狗事件

14.11.2. ADC 控制寄存器 1 (ADC_CR1)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	OVRIE	Res	ADSTP	Res	RESSEL[1:0]		AWDEN	JAWDEN	Res	Res	Res	Res	Res	Res
		RW		R_W1		RW		RW	RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
RW			RW	RW	RW	RW	RW	RW	RW	RW	RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29	OVRIE	RW	0	OVER FLAG 中断使能 0：禁止过载中断 1：允许过载中断
28	Reserved	-	-	Reserved
27	ADSTP	R_W1	0	ADC 转换停止使能。软件写 1 置 1。当接收到 ADC 停止信号时，硬件置 0。 1: 停止 ADC 转换。 0: 不停止 ADC 转换。
26	Reserved	-	-	Reserved
25:24	RESSEL[1:0]	RW	0	分辨率 (Resolution)控制位。通过软件写入这些位可选择转换的分辨率。 00： 12 位（15 ADCCLK 周期) 01： 10 位（13 ADCCLK 周期) 10： 8 位（11 ADCCLK 周期) 11： 6 位（9 ADCCLK 周期)
23	AWDEN	RW	0	规则通道模拟看门狗使能。在规则通道上开启模拟看门狗。该位由软件写 1 置 1 和写 0 置 0。 0：在规则通道上禁用模拟看门狗 1：在规则通道上使用模拟看门狗
22	JAWDEN	RW	0	注入通道模拟看门狗使能。在注入通道上开启模拟看门狗。该位由软件写 1 置 1 和写 0 置 0。 0：在注入通道上禁用模拟看门狗 1：在注入通道上使用模拟看门狗
21:16	Reserved	-	-	Reserved
15:13	DISCNUM[2:0]	RW	0	间断模式通道计数。在间断模式下，收到外部触发后，规则通道组的转换通道数。

Bit	Name	R/W	Reset Value	Function
				软件写操作设置这些位。 000： 1 个通道 001： 2 个通道 ...... 111： 8 个通道
12	JDISCEN	RW	0	注入通道间断模式使能。 该位由软件写 1 置 1 和写 0 置 0，用于开启或关闭注入通道组上的间断模式。 0：注入通道组上禁用间断模式 1：注入通道组上使用间断模式
11	DISCEN	RW	0	规则通道的间断模式使能 该位由软件写 1 置 1 和写 0 置 0，用于开启或关闭规则通道组上的间断模式 0：规则通道组上禁用间断模式 1：规则通道组上使用间断模式
10	JAUTO	RW	0	自动注入使能 该位由软件写 1 置 1 和写 0 置 0，用于开启或关闭规则通道组转换结束后自动进行注入通道组转换 0：关闭自动的注入通道组转换 1：开启自动的注入通道组转换
9	AWDSGL	RW	0	单一通道看门狗使能。 该位由软件写 1 置 1 和写 0 置 0，用于开启或关闭由 AWDCH[4:0]定义的通道上的模拟看门狗。 0：在所有的通道上使用模拟看门狗 1：在单一通道上使用模拟看门狗
8	SCAN	RW	0	扫描模式使能 该位由软件写 1 置 1 和写 0 置 0，用于开启或关闭扫描模式。在扫描模式中，由 ADC_SQRx 或 ADC_JSQRx 寄存器选中的通道被转换。 0：关闭扫描模式 1：使用扫描模式
7	JEOCIE	RW	0	注入通道转换结束中断使能 该位由软件写 1 置 1 和写 0 置 0。该 bit 使能后，在 JEOC 有效时，产生中断请求。 0：禁止 JEOC 中断 1：允许 JEOC 中断。
6	AWDIE	RW	0	模拟看门狗中断使能 该位由软件写 1 置 1 和写 0 置 0。该 bit 使能后，在 AWD 有效时，产生中断请求。 0：禁止模拟看门狗中断 1：允许模拟看门狗中断
5	EOCIE	RW	0	规则通道转换结束中断使能

Bit	Name	R/W	Reset Value	Function
				该位由软件写 1 置 1 和写 0 置 0。该 bit 使能后，在 EOC 有效时，产生中断请求。 0：禁止 EOC 中断 1：允许 EOC 中断。
4:0	AWDCH[4:0]	RW	0	模拟看门狗通道选择位 该位由软件写设置，用于选择模拟看门狗的输入通道。 00000： ADC 模拟输入通道 0 00001： ADC 模拟输入通道 1 ...... 01111： ADC 模拟输入通道 15 10000： TS_VIN 输入 10001： VREFINT 输入 10010: VCCA/3 10101: OPA1_VIN 10110: OPA2_VIN 保留所有其他数值。

14.11.3. ADC 控制寄存器 (ADC_CR2)

Address offset:0x08

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	VREF-BUFF_SEL		VREFBUFFERE	Res	TSVREFE	SWSTART	JSWSTART	EXTTRIG	EXTSEL[2:0]			Res
				RW	RW	RW		RW	R_W1	R_W1	RW	RW	RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JEXTTRIG	JEXTSEL[2:0]			ALIGN	Res	Res	DMA	Res	Res	Res	Res	RSTCAL	CAL	CONT	ADON
RW	RW	RW	RW	RW			RW					R_W1	R_W1	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	Reserved
27:26	VREFBUFF_SEL	RW	0	VREFBUF output voltage select 00: 1.5V 01: 2.048V 10: 2.5V 11: reserved
25	VREF BUFFERE	RW	0	VerfBuffer enable 软件写 0 置 0，写 1 置 1， 0：disable VerfBuffer 1：enable VerfBuffer
24	Reserved			
23	TSVREFE	RW	0	温度传感器和 VREFINT 使能 通过软件写 1 置 1 和写 0 置 0，用于开启或禁止温度传感器和 VREFINT 通道。 ADC1 中可使能，。 0：禁止温度传感器和 VREFINT 1：启用温度传感器和 VREFINT

Bit	Name	R/W	Reset Value	Function
22	SWSTART	R_W1	0	开始转换规则通道使能 通过软件写 1 置 1 启动转换，启动转换后立即被硬件清除置 0。 0：复位状态 1：开始转换规则通道
21	JSWSTART	R_W1	0	开始转换注入通道使能 由软件写 1 置 1 启动转换，启动转换后立即被硬件清除置 0。 0：复位状态 1：开始转换注入通道
20	EXTTRIG	RW	0	规则通道外部触发使能 该位由软件写 1 置 1 和写 0 置 0 清除，用于开启或禁止可以启动规则通道组转换的外部触发信号。 0：禁止规则通道外部触发 1：使能规则通道外部触发
19:17	EXTSEL[2:0]	RW	0	规则通道外部触发事件选择位。选择启动规则通道组转换的外部事件 ADC1,ADC2 外部触发事件如下： 000：定时器 1 的 CH1 事件 001：定时器 1 的 CH2 事件 010：定时器 1 的 CH3 事件 011：定时器 2 的 CH2 事件 100：定时器 3 的 TRGO 事件 101： 触发 timer15_TRGO 事件 110：EXTI11 111：SWSTART
16	Reserved			
15	JEXTTRIG	RW	0	注入通道外部触发使能 0：禁止注入通道外部触发 1：使能注入通道外部触发
14:12	JEXTSEL[2:0]	RW	0	注入通道组外部触发事件选择位 ADC1 和 ADC2 的外部触发事件如下： 000：定时器 1 的 TRGO 事件 001：定时器 1 的 CH4 事件 010：定时器 2 的 TRGO 事件 011：定时器 2 的 CH1 事件 100：定时器 3 的 TRGO 事件 101：定时器 15 的 TRGO 事件 110：EXTI15 111：JSWSTART
11	ALIGN	RW	0	数据对齐控制位 该位由软件写 1 置 1 和写 0 置 0 清除。 0：右对齐 1：左对齐
10:9	Reserved			

Bit	Name	R/W	Reset Value	Function
8	DMA	RW	0	DMA 使能位 该位由软件写 1 置 1 和写 0 置 0 清除。 0：禁止 DMA 模式 1：使能 DMA 模式
7:4	Reserved			
3	RSTCAL	RES	0	校准复位使能位 该位由软件写 1 置 1，由硬件清除置 0。在校准寄存器被初始化后(即 RSTCAL 置 1 后)，该位即被清除。 0：校准寄存器已初始化 1：初始化校准寄存器 注：当正在进行转换时，如果设置 RSTCAL，清除校准寄存器需要额外的周期。
2	CAL	R_W1	0	校准使能 该位由软件写 1 置 1 开始校准，并在校准失败或者校准成功时由硬件清除。当 ADON 无效时，且 SWSTART，JSWSTART 无效时；软件启动校准。 0：校准完成 1：使能校准
1	CONT	RW	0	连续转换使能 该位由软件写 1 置 1 和写 0 置 0 清除。如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式 1：连续转换模式
0	ADON	RW	0	开/关 A/D 转换器 ADC 转换器工作使能，当置 1 后 ADC 转换器唤醒，在转换器上电到开始转换有一个延迟 tSTAB。当置 0 后 ADC 转换器处于掉电状态。 0：禁止 ADC 转换，ADC 转换器进入断电模式 1：使能 ADC 转换器。 注：如果在该寄存器中 SWSTART,JSWSTART 与 ADON 一起被改变，则转换不被触发。这是为了防止触发错误的转换。

14.11.4. ADC 采样时间寄存器 1 (ADC_SMPR1)

Address offset:0x0C

Reset value:0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				SMP23[2:0]			SMP22[2:0]			SMP21[2:0]			SMP20[2:0]		
				RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	SMPx[2:0]	RW	0	选择通道 x 的采样时间 通过软件设置这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。 000： 3.5 周期　100： 28.5 周期 001： 5.5 周期　101： 41.5 周期 010： 7.5 周期　110： 134.5 周期 011： 13.5 周期　111： 239.5 周期

14.11.5. ADC 采样时间寄存器 2 (ADC_SMPR2)

Address offset:0x10

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
		RW			RW			RW			RW			RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP15[0]	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
RW	RW			RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	0	选择通道 x 的采样时间 通过软件设置这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。 000： 3.5 周期　100： 28.5 周期 001： 5.5 周期　101： 41.5 周期 010： 7.5 周期　110： 134.5 周期 011： 13.5 周期　111： 239.5 周期

14.11.6. ADC 采样时间寄存器 3 (ADC_SMPR3)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		RW			RW			RW			RW			RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMP5[0]	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
RW	RW			RW			RW			RW			RW		

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:0	SMPx[2:0]	RW	0	选择通道 x 的采样时间 通过软件设置这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。

Bit	Name	R/W	Reset Value	Function
				000：3.5 周期 100：28.5 周期 001：5.5 周期 101：41.5 周期 010：7.5 周期 110：134.5 周期 011：13.5 周期 111：239.5 周期

14.11.7. ADC 注入通道数据偏移寄存器 x (ADC_JOFRx)(x=1..4)

Address offset:0x18-0X24

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	JOFF-SETx[11:0]	RW	0	注入通道第 x 次转换数据偏移 软件配置这些位的值。当转换注入通道时，这些位定义了用于从原始转换数据中减去的数值。最终转换结果可以在 ADC_JDRx 寄存器中读出。

14.11.8. ADC 看门狗高阈值寄存器(ADC_HTR)

Address offset:0x28

Reset value:0x0000 0FFF

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

Bit	Name	R/W	Reset Value	Function
31：11	Reserved	-	-	Reserved
11:0	HT[11:0]	RW	0xFFF	模拟看门狗高阈值 软件配置这些位的值。这些位定义了模拟看门狗的阈值高限。

14.11.9. ADC 看门狗低阈值寄存器(ADC_LRT)

Address offset:0x2C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function

31：11	Reserved	-	-	Reserved
11:0	LT[11:0]	RW	0x000	模拟看门狗低阈值 软件配置这些位的值。这些位定义了模拟看门狗的阈值低限。

14.11.10. ADC 规则序列寄存器 1(ADC_SQR1)

Address offset:0x30

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								L[3:0]				SQ16[4:1]			
								RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ16[0]	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23:20	L[3:0]	RW	0	规则通道序列长度 软件配置这些位的值。这些位定义了在规则通道转换序列中通道数目。 0000： 1 个转换 0001： 2 个转换 ...... 1111： 16 个转换
19:15	SQ16[4:0]	RW	0	软件配置这些位的值。规则序列中的第 16 个转换，这些位定义了转换序列中的第 16 个转换通道的编号(0~23)。
14:10	SQ15[4:0]	RW	0	软件配置这些位的值。规则序列中的第 15 个转换，这些位定义了转换序列中的第 15 个转换通道的编号(0~23)。
9:5	SQ14[4:0]	RW	0	软件配置这些位的值。规则序列中的第 14 个转换，这些位定义了转换序列中的第 14 个转换通道的编号(0~23)。
4:0	SQ13[4:0]	RW	0	软件配置这些位的值。规则序列中的第 13 个转换，这些位定义了转换序列中的第 13 个转换通道的编号(0~23)。

14.11.11. ADC 规则序列寄存器 2(ADC_SQR2)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SQ12[4:0]					SQ11[4:0]					SQ10[4:1]			
		RW					RW					RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ10[0]	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:25	SQ12[4:0]	RW	0	软件配置这些位的值。规则序列中的第 12 个转换，这些位定义了转换序列中的第 12 个转换通道的编号(0~23)。
24:20	SQ11[4:0]	RW	0	软件配置这些位的值。规则序列中的第 11 个转换，这些位定义了转换序列中的第 11 个转换通道的编号(0~23)。

19:15	SQ10[4:0]	RW	0	软件配置这些位的值。规则序列中的第 10 个转换，这些位定义了转换序列中的第 10 个转换通道的编号(0~23)。
14:10	SQ9[4:0]	RW	0	软件配置这些位的值。规则序列中的第 9 个转换，这些位定义了转换序列中的第 9 个转换通道的编号(0~23)。
9:5	SQ8[4:0]	RW	0	软件配置这些位的值。规则序列中的第 8 个转换，这些位定义了转换序列中的第 8 个转换通道的编号(0~23)。
4:0	SQ7[4:0]	RW	0	软件配置这些位的值。规则序列中的第 7 个转换，这些位定义了转换序列中的第 7 个转换通道的编号(0~23)。

14.11.12. ADC 规则序列寄存器 3(ADC_SQR3)

Address offset:0x38

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		RW					RW					RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ4[0]	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	Reserved
29:25	SQ6[4:0]	RW	0	软件配置这些位的值。规则序列中的第 6 个转换，这些位定义了转换序列中的第 6 个转换通道的编号(0~23)。
24:20	SQ5[4:0]	RW	0	软件配置这些位的值。规则序列中的第 5 个转换，这些位定义了转换序列中的第 5 个转换通道的编号(0~23)。
19:15	SQ4[4:0]	RW	0	软件配置这些位的值。规则序列中的第 4 个转换，这些位定义了转换序列中的第 4 个转换通道的编号(0~23)。
14:10	SQ3[4:0]	RW	0	软件配置这些位的值。规则序列中的第 3 个转换，这些位定义了转换序列中的第 3 个转换通道的编号(0~23)。
9:5	SQ2[4:0]	RW	0	软件配置这些位的值。规则序列中的第 2 个转换，这些位定义了转换序列中的第 2 个转换通道的编号(0~23)。
4:0	SQ1[4:0]	RW	0	软件配置这些位的值。规则序列中的第 1 个转换，这些位定义了转换序列中的第 1 个转换通道的编号(0~23)。

14.11.13. ADC 注入序列寄存器(ADC_JSQR)

Address offset:0x3C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								JL[3:0]				JSQ4[4:1]			
								RW				RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JSQ4[0]	JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
RW	RW					RW					RW				

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
23:20	JL[3:0]	RW	0	注入通道序列长度 软件配置这些位的值。这些位定义了在注入通道转换序列中通道数目。

				00： 1 个转换 01： 2 个转换 10： 3 个转换 11： 4 个转换
19:15	JSQ4[4:0]	RW	0	注入序列中的第 4 个转换 软件配置这些位的值。这些位定义了转换序列中的第 4 个转换通道的编号(0~23)。 注：不同于规则转换序列，如果 JL[1:0]的长度小于 4，则转换的序列顺序是从(4-JL)开始。 例如： ADC_JSQR[21:0] = 10 00011 00011 00111 00010，意味着扫描转换将按下列通道顺序转换： 7、 3、 3，而不是 2、 7、 3
14:10	JSQ3[4:0]	RW	0	软件配置这些位的值。注入序列中的第 3 个转换
9:5	JSQ2[4:0]	RW	0	软件配置这些位的值。注入序列中的第 2 个转换
4:0	JSQ1[4:0]	RW	0	软件配置这些位的值。注入序列中的第 1 个转换

14.11.14. ADC 注入数据寄存器 x (ADC_JDRx) (x= 1..4)

Address offset:0x40-4C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
15:0	JDATA[15:0]	R	0	注入通道转换结果 软件可读这些位的值。这些位为只读，包含了注入通道的转换结果。数据是左对齐或右对齐

14.11.15. ADC 规则数据寄存器(ADC_DR)

Address offset:0x40-4C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	Reserved
15:0	DATA[15:0]	R	0	规则通道转换结果 软件可读这些位的值。这些位为只读，包含了规则通道的转换结果。数据是左或右对齐

14.11.16. ADC 校准配置和状态寄存器(ADC_CCSR)

Address offset:0x44

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CALON.	CAP-SUC	OFFSUC	Res												
RO	RC_W1	RC_W1													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALSET	CAL-BYP	CALSMP[1:0]		CALSEL	Reserved										
R_W1	R_W1	RW		RW											

Bit	Name	R/W	Reset Value	Function
31	CALON	RO	0	Calibration flag，标志 ADC 校准正在进行。 1：ADC 校准正在进行 0：ADC 校准已结束或未启动 ADC 校准
30	CAPSUC	RC_W1	0	电容校准状态位。 表示 ADC 电容校准是否成功。硬件置 1；软件写 1 置 0； CALON=0, CALSEL=0,CALSUC=1： 无效状态 CALON=0, CALSEL=0, CALSUC=0： 未进行 CAPs 校准 CALON=0, CALSEL=1, CALSUC =1： ADC CAPs 校准成功 CALON=0, CALSEL=1, CALSUC =0： ADC CAPs 校准失败
29	OFFSUC	RC_W1	0	Offset 校准状态位。 表示 ADC offset 校准是否成功。硬件置 1；软件写 1 置 0； CALON=0, CALSEL=0,OFFSUC=0： ADC OFFSET 校准失败 CALON=0, CALSEL=0, OFFSUC=1：ADC OFFSET 校准成功 CALON=0, CALSEL=1,OFFSUC=1： ADC OFFSET 校准成功 CALON=0, CALSEL=1, OFFSUC=0：ADC OFFSET 校准失败
28:16	Reserved	-	-	Reserved
15	CALSET	RC_W1	0	校准因子选择。当 CAL 为 0 时，软件写 1 置 1。CAL 有效或注入/规则通道 SWSTART，JWSTART 有效时，硬件置 0。 1：设置 CAL_CXIN 数据作为最终的校准数据 0：关闭 CAL_CXIN 到 CAL_CXOUT 的通路，选择校准电路内部产生的结果。
14	CALBYP	R_W1	0	校准因子旁路。当 CAL 为 0 时，软件写 1 置 1。CAL 有效或注入/规则通道 SWSTART，JWSTART 有效时，硬件置 0。

Bit	Name	R/W	Reset Value	Function
				1：校准结果为复位值 0：校准结果为自校准结果或者校准因子输入值
13:12	CALSMP[1:0]	RW	0	Calibration sample time seletion 根据以下信息， 配置校准采样阶段的时钟周期个数: 00： 1 个 ADC 时钟周期 01： 2 个 ADC 时钟周期 10： 4 个 ADC 时钟周期 11： 8 个 ADC 时钟周期 校准时配置 SMP 的周期越长，校准结果更精确，但该配置会带来校准周期延长的问题
11	CALSEL	RW	0	Calibration 内容选择位，用于选择需要校准的内容 1：校准 OFFSET 以及线性度 0：只校准 OFFSET
10:0	Reserved	-	-	Reserved

14.11.17. ADC 寄存器映像

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	ADC_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OVER	STRT	JSTRT	JEOC	EOC	AWD
	Reset value																											0	0	0	0	0	0
0x04	ADC_CR1	Res.	Res.	OVERIE	Res.	ADSTP	Res.	RES-SEL[1:0]		AWDEN	JAWDEN	Res.	Res.	Res.	Res.	Res.	Res.	DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
	Reset value			0		0		0		0	0							0			0	0	0	0	0	0	0	0	0				
0x08	ADC_CR2	Res.	Res.	Res.	Res.	Verfbuff_sel.		Verfbuffe	Res.	TSVREF	SWSTAR	JSWSTA	EXTTRIG	EXTSEL[2:0]			Res.	JEXTTRI	JEXTSEL[2:0]			ALIGN	Res.	Res.	DMA	Res.	Res.	Res.	Res.	RSTCAL	CAL	CONT	ADON
	Reset value					0		0		0	0	0	0	0				0	0			0			0					0	0	0	0
0x0C	ADC_SMPR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMP23[2:0]			SMP22[2:0]			SMP21[2:0]			SMP20[2:0]		
	Reset value																					0			0			0			0		
0x10	ADC_SMPR2	Res.	Res.	SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:0]			SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		

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
	Reset value			0			0			0			0			0			0			0			0			0			0		
0x14	ADC_SMPR3	Res.	Res.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:0]			SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
	Reset value			0			0			0			0			0			0			0			0			0			0		
0x18-0x24	ADC_JOFRx	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JOFFSETx[11:0]											
	Reset value																					0											
0x28	ADC_HTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT[11:0]											
	Reset value																					0xFFF											
0x2C	ADC_LRT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT[11:0]											
	Reset value																					0											
0x30	ADC_SQR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	L[3:0]				SQ16[4:0]					SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
	Reset value									0				0					0					0					0				
0x34	ADC_SQR2	Res.	Res.	SQ12[4:0]					SQ11[4:0]					SQ10[4:0]					SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
	Reset value			0					0					0					0					0					0				
0x38	ADC_SQR3	Res.	Res.	SQ6[4:0]					SQ5[4:0]					SQ4[4:0]					SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
	Reset value			0					0					0					0					0					0				
0x3C	ADC_JSQR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JL[3:0]				JSQ4[4:0]					JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
	Reset value									0				0					0					0					0				
0x40-0x4C	ADC_JDRx	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JDATA[15:0]															
	Reset																	0															

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
	value																																
0x50	ADC_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DATA[15:0]															
	Reset value																	0															
0x54	ADC_CCSR	CALON.	CAPSUC.	OFFSUC.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CALSET.	CALBYP.	CALSMP		CALSEL.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	0	0	0														0	0	0		0											

15.液晶控制器 (LCD)

15.1. 简介

LCD 控制器是一款适用于单色无源液晶显示器(LCD)的数字控制器/驱动器，最多具有8 个公用端子（COM）和 40 个区段端子（SEG），用以驱动 160 (4x40)或 288 (8x36)个 LCD 图像元素。端子的确切数量取决于数据手册中所述的器件引脚。LCD 由若干区段（像素或完整符号）组成，这些区段均可点亮或熄灭。每个区段都包含一层在两根电极之间对齐的液晶分子。当向液晶施加高于阈值电压的电压时，相应的区段可见。区段电压必须为交流，以避免液晶中出现电泳效应（这将影响显示效果）。之后，必须在区段两端生成波形以避免出现直流。

词汇表

液晶（ LCD）： 无源显示面板，带有直接引向区段的端子。

公用（ COM）： 连接到多个区段的电气连接端子。

偏置（ BIAS）： 驱动 LCD 时使用的电压等级，定义为 1/(驱动 LCD 显示的电压等级数−1)。

区段（ SEG）： 最小可视单元（ LCD 显示器上的最小组成元素，线条或点）。

占空比（ DUTY）： 定义为 1/(LCD 显示器上的公用端子数)的数字。

帧：写入区段的波形的一个周期。

帧速率：每秒帧数，即每秒激励 LCD 区段的次数。

15.2. LCD 主要特性

1） 高度灵活的帧速率控制。
2） 支持静态、 1/2、 1/3、 1/4、 1/6 和 1/8 占空比。
3） 支持 1/2、 1/3 偏置电压。
4） 多达 16 个寄存器的 LCD 数据 RAM。
5） 可通过软件配置 LCD 的对比度。
6） 2 种驱动波形生成方式
   – 内部电阻分压、外部电阻分压，
   – 可通过软件配置内部电阻分压方式的功耗，从而匹配 LCD 面板所需的电容电荷
7） 支持低功耗模式： LCD 控制器可在 run、 sleep、 stop 模式下进行显示。
8） 可配置帧中断。
9） 支持 LCD 闪烁功能且可配置多种闪烁频率
10） 未使用的 LCD 区段和公共引脚可配置为数字或模拟功能。

15.3. LCD 框图

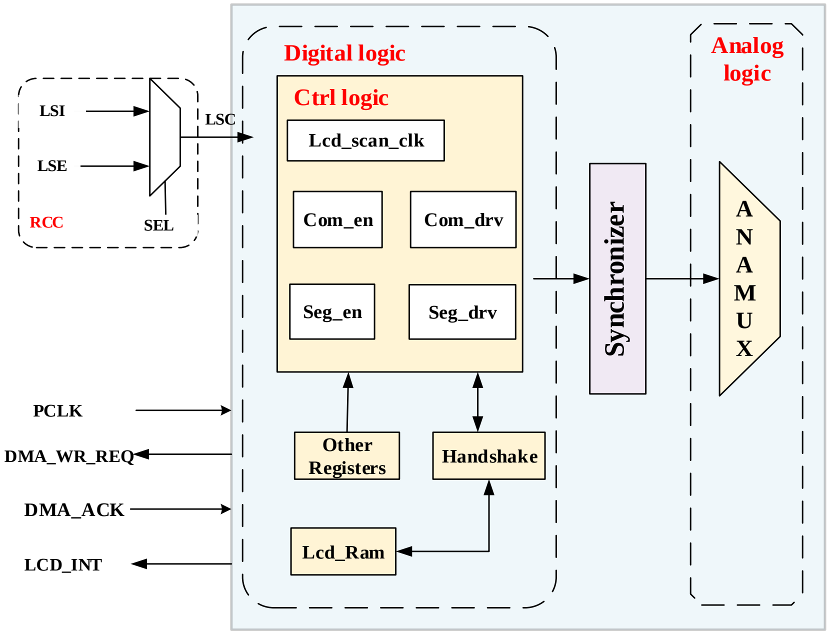


图 15-1 LCD 框图

15.4. LCD 时钟

LCD 时钟可选择为 LSI 或 LSE，可通过 RCC_BDCR 寄存器的 LSCOSEL 及 LSCOEN 确定输入时钟。

15.5. LCD 驱动波形

LCD 支持 5 种占空比（ Duty）的驱动波形：静态、 1/2、 1/3、 1/4、 1/6 和 1/8，由 LCD_CR0.Duty 进行设置。 LCD 支持 2 种偏置（ Bias）的驱动波形： 1/2、 1/3，由 LCD_CR0.Bias 进行设置。 建议的组合方式如下表所示：

表 15-1 LCD 支持的驱动波形

	1/2 Duty	1/3 Duty	1/4 Duty	1/6 Duty	1/8 Duty
1/2 Bias	✓	✓	不推荐	不推荐	不推荐
1/3 Bias	不推荐	不推荐	✓	✓	✓

各模式下的驱动波形如下方所示：

15.5.1. 静态驱动波形

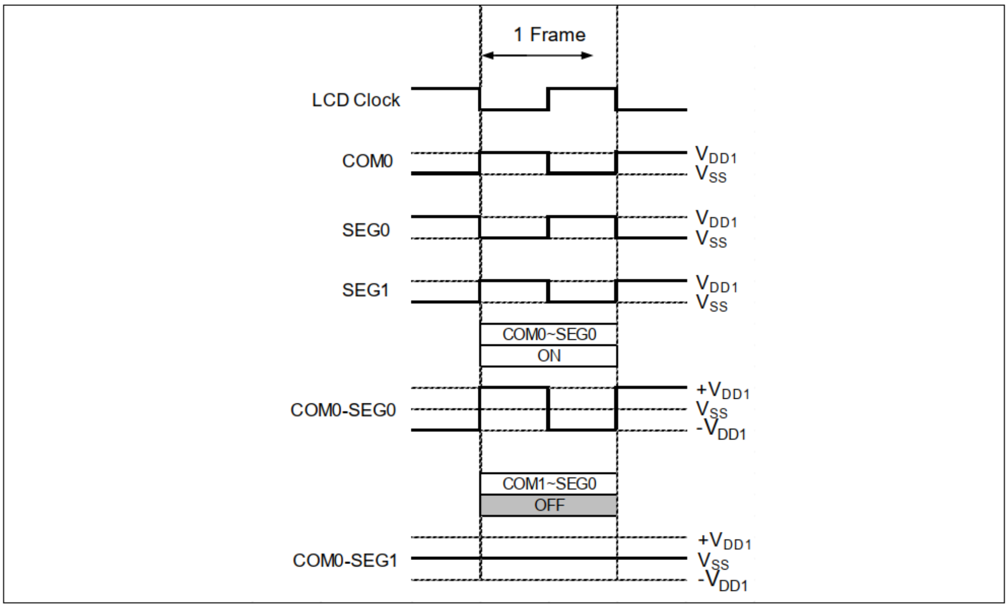


图 15-2 静态驱动波形

15.5.2. 1/2Duty 1/2Bias 驱动波形

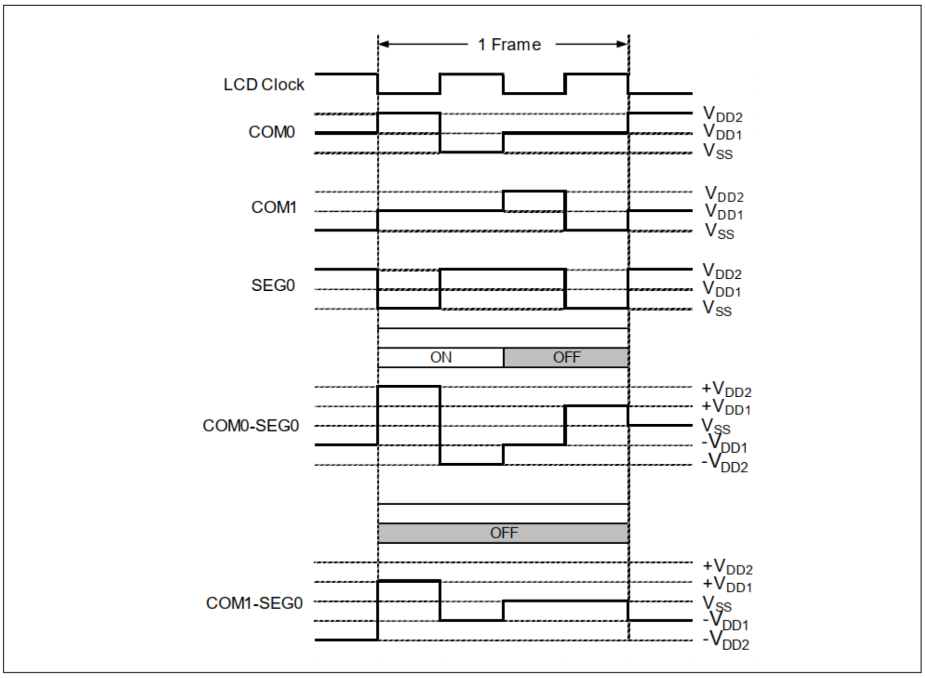


图 15-3 1/2Duty 1/2Bias 驱动波形

15.5.3. 1/2Duty 1/3Bias 驱动波形

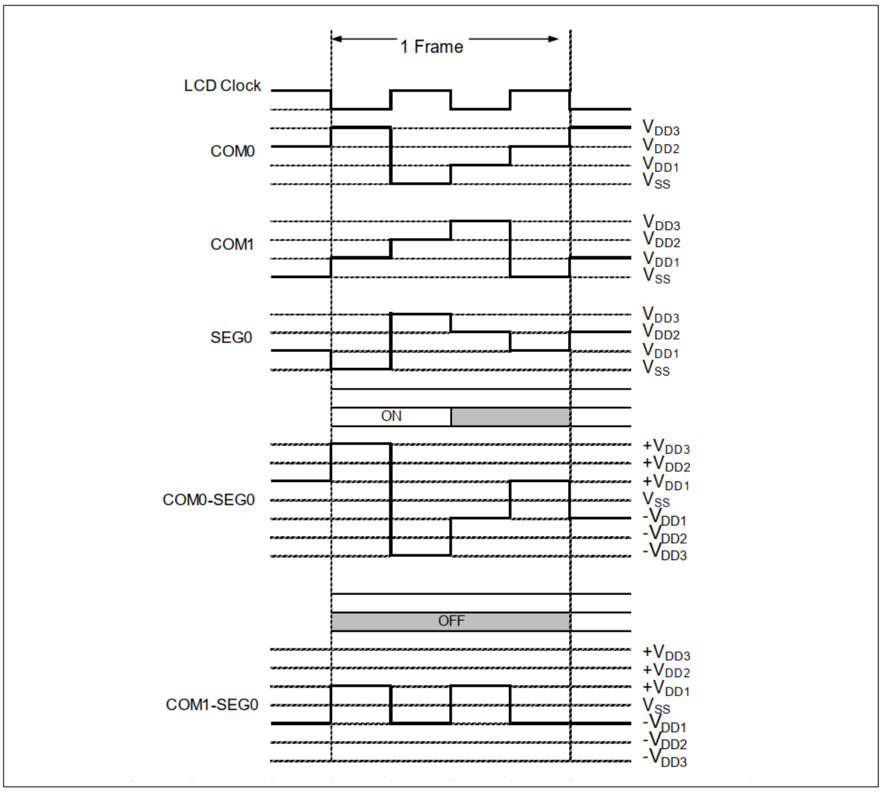


图 15-4 1/2Duty 1/3Bias 驱动波形

15.5.4. 1/3Duty 1/2Bias 驱动波形

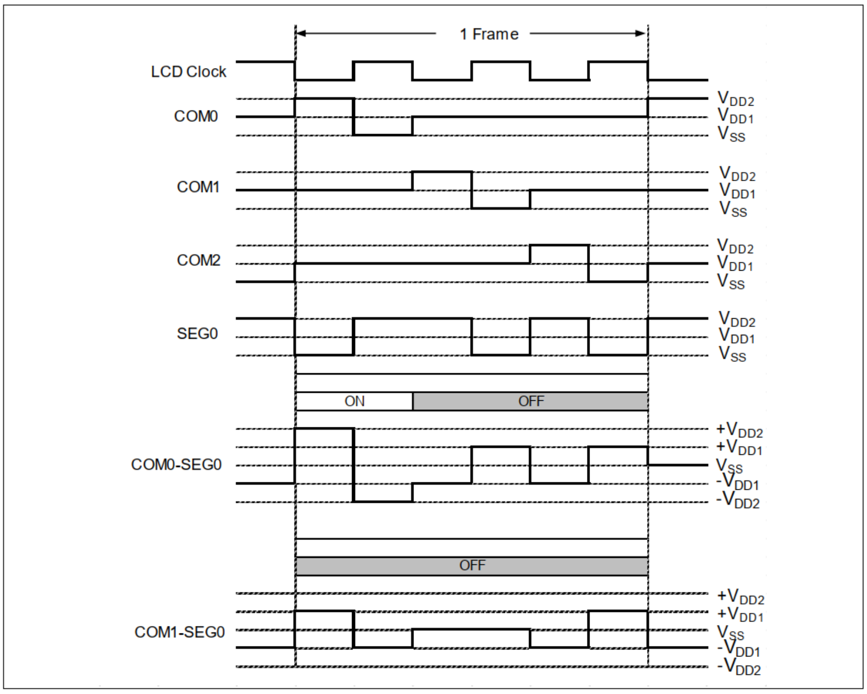


图 15-5 1/3Duty 1/2Bias 驱动波形

15.5.5. 1/3Duty 1/3Bias 驱动波形

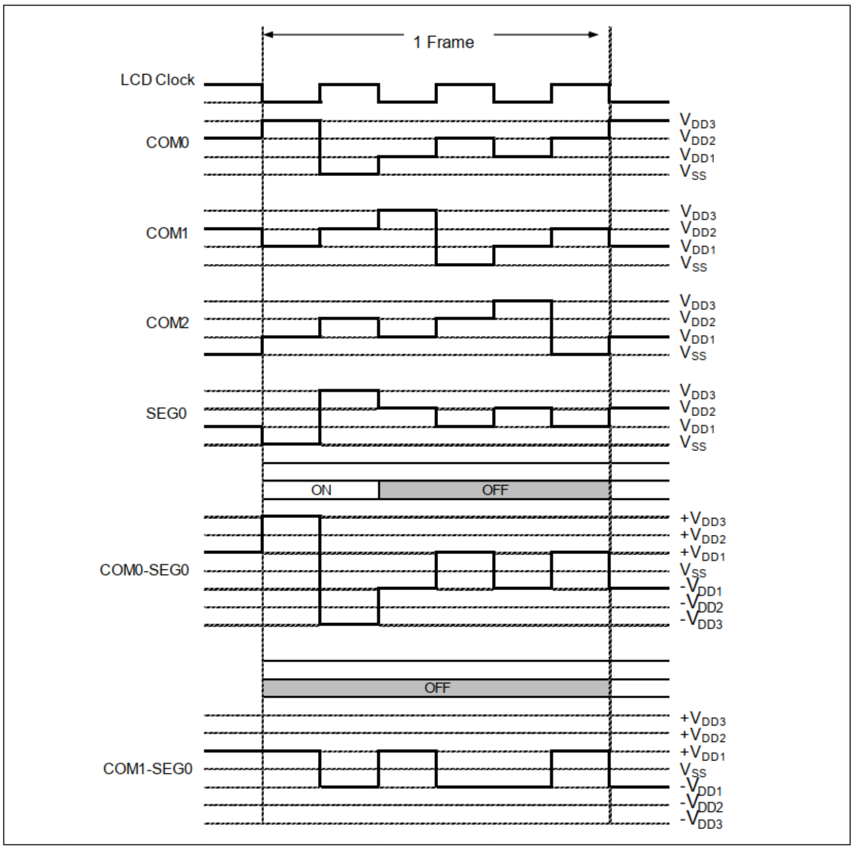


图 15-6 1/3Duty 1/3Bias 驱动波形

15.5.6. 1/4Duty 1/2Bias 驱动波形

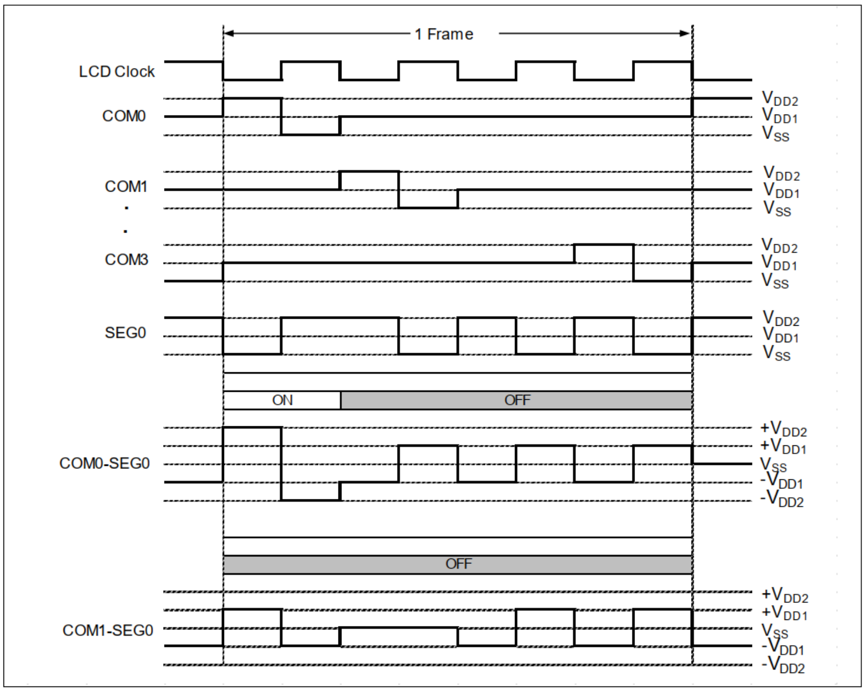


图 15-7 1/4Duty 1/2Bias 驱动波形

15.5.7. 1/4Duty 1/3Bias 驱动波形

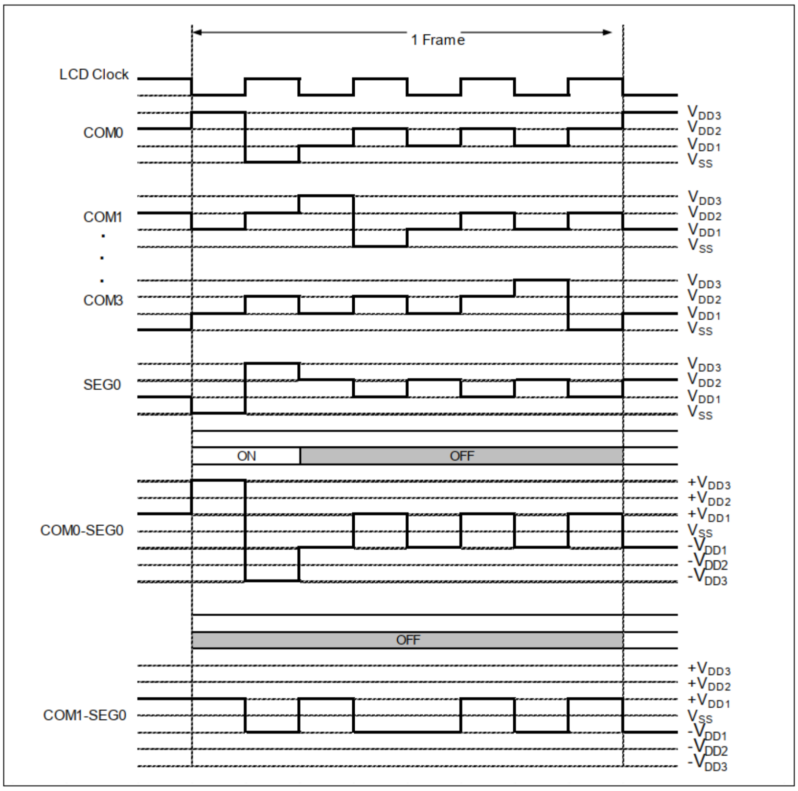


图 15-8 1/4Duty 1/3Bias 驱动波形

15.5.8. 1/6Duty 1/3Bias 驱动波形

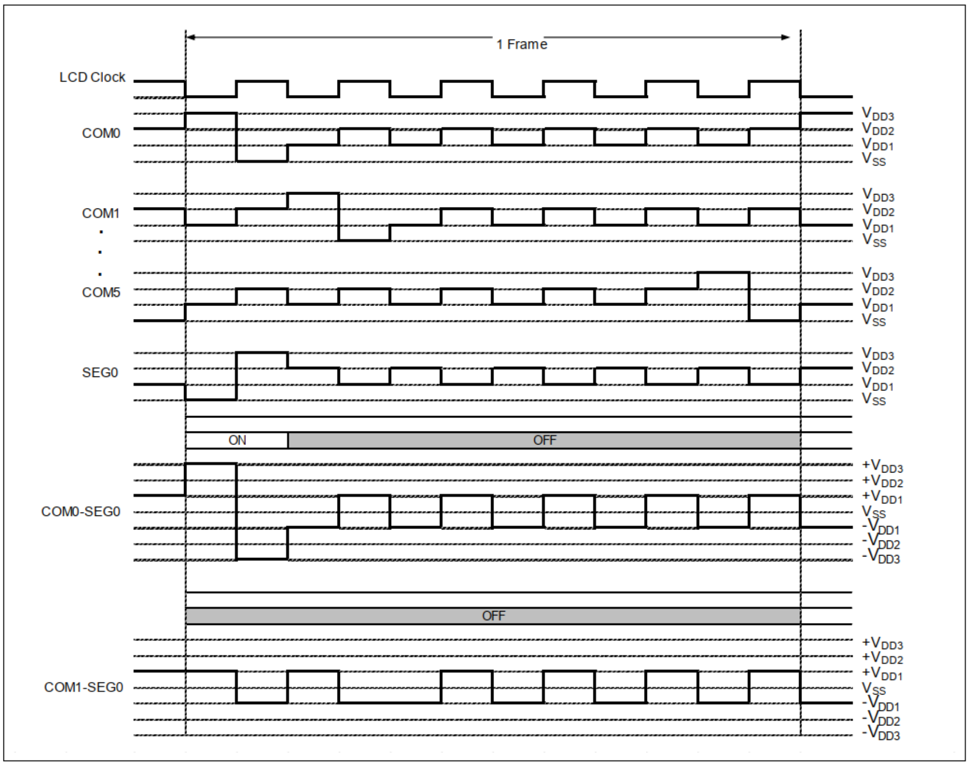


图 15-9 1/6Duty 1/3Bias 驱动波形

15.5.9. 1/8Duty 1/3Bias 驱动波形

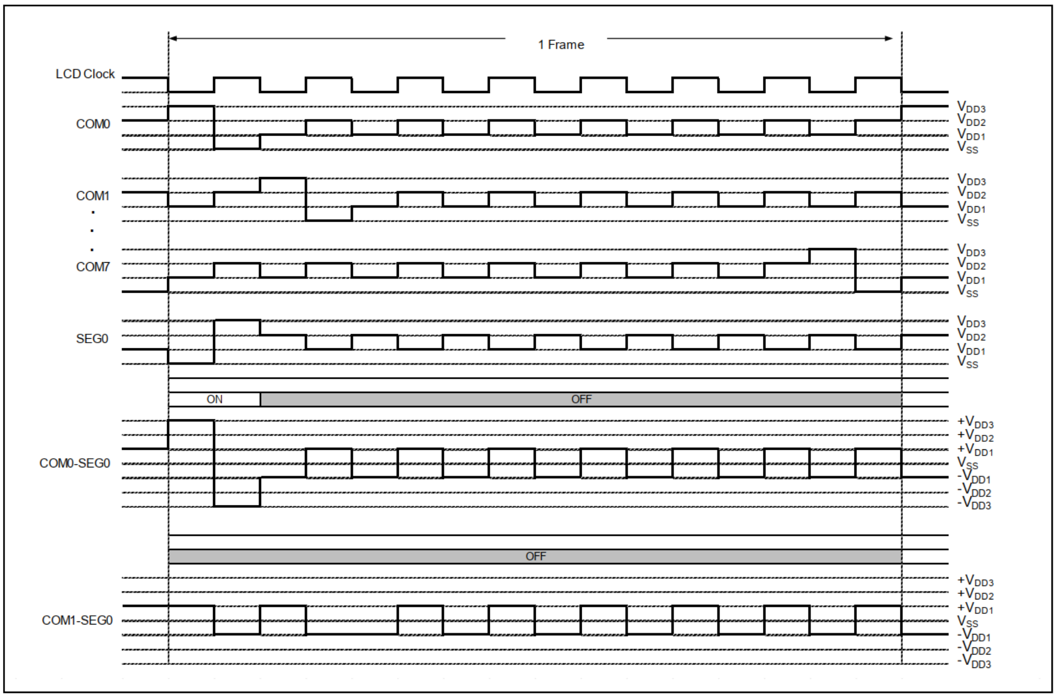


图 15-10 1/8Duty 1/3Bias 驱动波形

15.6. LCD Bias 产生电路

LCD 的 Bias 电压具有 2 种来源：内部电阻分压、外部电阻分压。当选择内部电阻分压时，芯片会自动切换内部的电路以产生符合 Bias 和 Duty 的电压。当选择外部电阻分压时，需要用户在芯片的外围引脚搭建相关电路。

15.6.1. 内部电阻模式

内部电阻模式 VLCDH,VLCD1~VLCD3 可以作为 LCD SEG 输出或者 IO 端口使用。

内部电阻模式， LCD 的驱动电压由 CR0.Contrast 控制， 如下表所示：

表 15-2 内部电阻模式

CR0.Contrast	VLCD(1/3 bias)	VLCD(1/2 bias)
0	1.00 * VCC	1.00 * VCC
1	0.94* VCC	0.92* VCC
2	0.9 * VCC	0.85 * VCC
3	0.85* VCC	0.8* VCC
4	0.81 * VCC	0.75 * VCC
5	0.78 * VCC	0.7 * VCC
6	0.75 * VCC	0.66 * VCC
7	0.72 * VCC	0.63 * VCC
8	0.70 * VCC	0.61 * VCC
9	0.67 * VCC	0.58 * VCC
10	0.65 * VCC	0.55 * VCC
11	0.63 * VCC	0.53 * VCC

CR0.Contrast	VLCD(1/3 bias)	VLCD(1/2 bias)
12	0.61 * VCC	0.51 * VCC
13	0.59 * VCC	0.48 * VCC
14	0.57 * VCC	0.47 * VCC
15	0.55 * VCC	0.45 * VCC

15.6.2. 外部电阻模式

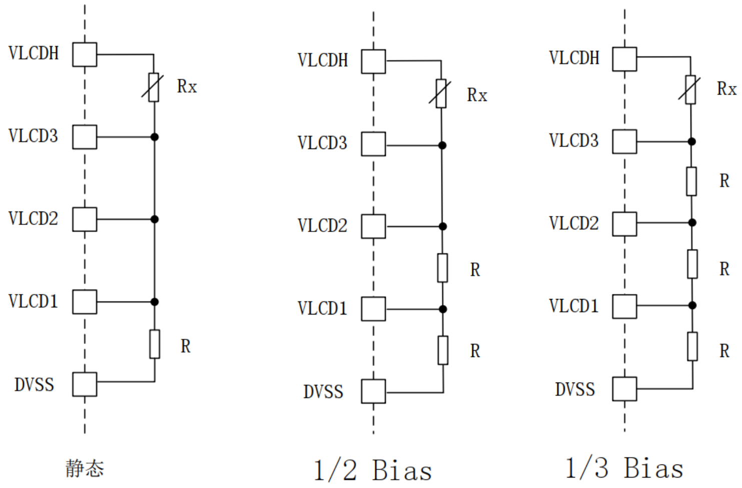


图 15-11 外部电阻模式

注意：

– Rx 为可调电阻，用于调节 LCD 显示对比度。

– 根据使用 LCD 屏幕选择合适的电阻 R。

15.7. DMA

LCD 支持软件和硬件触发 DMA 数据传输，可以将需要显示的内容从 RAM 或 ROM 中自动搬到 LCD 显示 RAM 中。硬件触发使用的是帧中断信号。 LCD 使用的 DMA 通道号为[8]。

DMA 数据传送配置流程：

1. 使能 DMA
2. 选择 LCD DMA
3. 设置传输类型、传输长度、传输方式
4. 设置源起始地址，目标起始地址
5. 设置源地址、目标地址的递增方式
6. 根据需要使能 DMA 中断
7. 使能 LCD DMA 触发

15.8. 中断

当 LCD 设置有效时， LCD 中断可以配置为帧数产生中断。

15.9. LCD 显示模式

LCD 支持两种显示模式。一种以 COM 为显示单元，同一个 SEG 的所有 COM 段在同一字节中（模式 0）。另外一种为同一个 COM 的不同 SEG 在同一个字节中（模式 1）。

根据 LCD 面板选择合适的显示方式可以简化程序操作。

15.9.1. LCD 显示模式 1 (MODE = 1)

1/8 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7
																													SEG35	SEG34	SEG33	SEG32	
																							COM0									LCDRAM8
																							COM1									LCDRAM9
																							COM2									LCDRAMA
																							COM3									LCDRAMB
																							COM4									LCDRAMC
																							COM5									LCDRAMD
																							COM6									LCDRAME
																							COM7									LCDRAMF

图 15-12 1/8 Duty LCD 显示模式 1

1/6 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
																										SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																							COM0									LCDRAM8
																							COM1									LCDRAM9
																							COM2									LCDRAMA
																							COM3									LCDRAMB
																							COM4									LCDRAMC
																							COM5									LCDRAMD
																																	LCDRAME
																																	LCDRAMF

图 15-13 1/6 Duty LCD 显示模式 1

1/4 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
																																	LCDRAM4
																																	LCDRAM5
																																	LCDRAM6
																																	LCDRAM7
																								SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																							COM0									LCDRAM8
																							COM1									LCDRAM9
																							COM2									LCDRAMA
																							COM3									LCDRAMB
																																	LCDRAMC
																																	LCDRAMD
																																	LCDRAME
																																	LCDRAMF

图 15-14 1/4 Duty LCD 显示模式 1

1/3 Duty 1/2 Duty

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
	SEG31	SEG30	SEG29	SEG28	SEG27	SEG26	SEG25	SEG24	SEG23	SEG22	SEG21	SEG20	SEG19	SEG18	SEG17	SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
COM0																																	LCDRAM0
COM1																																	LCDRAM1
COM2																																	LCDRAM2
COM3																																	LCDRAM3
COM4																																	LCDRAM4
COM5																																	LCDRAM5
COM6																																	LCDRAM6
COM7																																	LCDRAM7
																									SEG39	SEG38	SEG37	SEG36	SEG35	SEG34	SEG33	SEG32	
																								COM0									LCDRAM8
																								COM1									LCDRAM9
																								COM2									LCDRAMA
																								COM3									LCDRAMB
																								COM4									LCDRAMC
																								COM5									LCDRAMD
																								COM6									LCDRAME
																								COM7									LCDRAMF

图 15-15 1/3 1/2 Duty LCD 显示模式 1

15.9.2. LCD 显示模式 0 (MODE = 0)

1/8 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG29								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																																LCDRAMC
																																LCDRAMD
																																LCDRAME
																																LCDRAMF

图 15-16 1/8 Duty LCD 显示模式 0

1/6 Duty

Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
		COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0			COM5	COM4	COM3	COM2	COM1	COM0	
							SEG3								SEG2								SEG1								SEG0	LCDRAM0
							SEG7								SEG6								SEG5								SEG4	LCDRAM1
							SEG11								SEG10								SEG9								SEG8	LCDRAM2
							SEG15								SEG14								SEG13								SEG12	LCDRAM3
							SEG19								SEG18								SEG17								SEG16	LCDRAM4
							SEG23								SEG22								SEG21								SEG20	LCDRAM5
							SEG29								SEG26								SEG25								SEG24	LCDRAM6
							SEG31								SEG30								SEG29								SEG28	LCDRAM7
																															SEG32	LCDRAM8
																															SEG33	LCDRAM9
																															SEG34	LCDRAMA
																															SEG35	LCDRAMB
																															SEG36	LCDRAMC
																															SEG37	LCDRAMD
																																LCDRAME
																																LCDRAMF

图 15-17 1/6 Duty LCD 显示模式 0

1/4 Duty

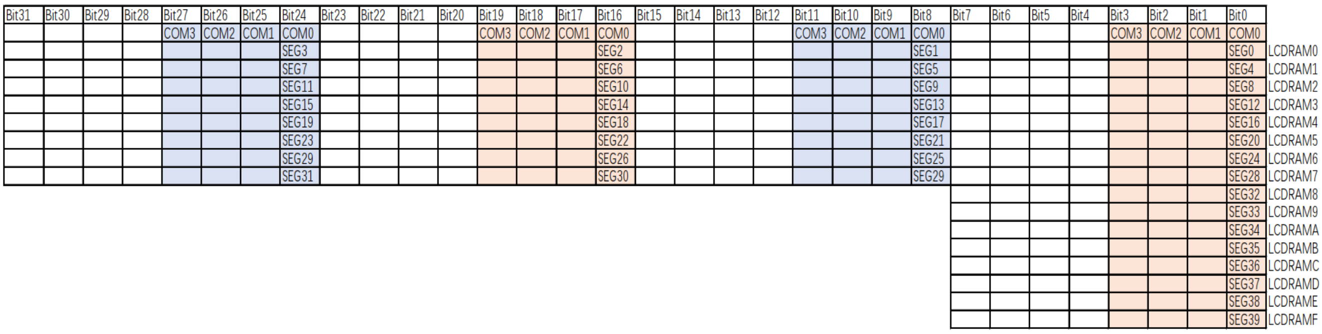


图 15-18 1/4 Duty LCD 显示模式 0

1/3 Duty 1/2 Duty

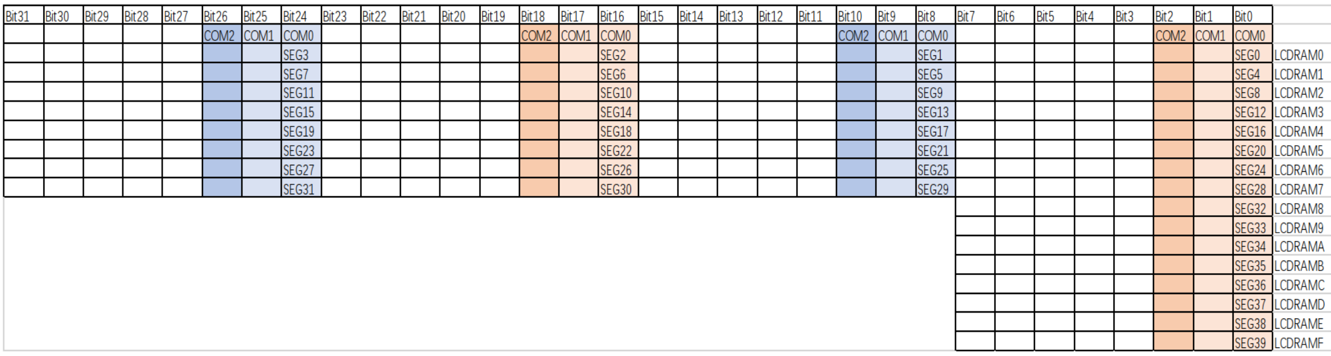


图 15-19 1/3 1/2 Duty LCD 显示模式 0

15.10. LCD 寄存器

15.10.1. 配置寄存器 0 (LCD_CR0)

Address offset:0x00

Reset value:0x00C2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Contrast				BSEL			DUTY			BIAS	Res	Res	LCDCLK		EN
RW				RW			RW			RW			RW		RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:12	Contrast	RW	3’b000	LCD 对比度调整 注：仅当 Bias 电压来源选择内部电阻分压时有效。 Constrast 值越大， LCD 波形的幅度越小。 0x0 时， LCD 波形幅度最大，对比度最大； ...... 0xF 时， LCD 波形幅度最小，对比度最小；
11:9	BSEL	RW	3’b000	Bias 电压来源选择

Bit	Name	R/W	Reset Value	Function
				111：Reserved 110：内部电阻分压，大功耗模式 101：Reserved 100：内部电阻分压，小功耗模式 011：Reserved 010：内部电阻分压，中功耗模式 001：Reserved 000：外部电阻模式，需要外部电路配合
8:6	DUTY	RW	3’b011	LCD duty 配置 000: 静态 001: 1/2 duty 010: 1/3 duty 011: 1/4 duty 100: Reserved 101: 1/6 duty 110: Reserved 111: 1/8 duty
5	BIAS	RW	0	LCD Bias 配置 0: 1/3 偏压（初始值） 1: 1/2 偏压
4:3	Reserved	-	-	Reserved
2:1	LCDCLK	RW	2’b01	LCD 扫描频率选择 00: 64Hz 01: 128Hz 10: 256Hz 11: 512Hz 注： LCD 帧频率 = LCD 扫描频率×Duty
0	EN	RW	0	LCD 使能控制 1：使能 0：禁止

15.10.2. 配置寄存器 1 (LCD_CR1)

Address offset:0x04

Reset value:0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	INTF	DMAEN	IE	MODE	Res	BLINKEN	BLINKCNT					
				RO	RW	RW	RW		RW	RW					

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11	INTF	RO	0	LCD 中断标志 1：中断 0：无中断
10	DMAEN	RW	0	DMA 硬件触发使能 1：使能 LCD 中断触发 DMA 0：禁止 LCD 中断触发 DMA
9	IE	RW	0	中断使能 1：使能

Bit	Name	R/W	Reset Value	Function
				0：禁止
8	MODE	RW	0	LCD RAM 显示模式选择 0: 模式 0 1: 模式 1
7	Reserved	-	-	Reserved
6	BLINKEN	RW	0	LCD 闪屏配置 1：使能 0：禁止
5:0	BLINKCNT	RW	0	闪屏频率与 LCD 中断间隔设置 注： LCD 闪烁频率为 = LCD 帧频率 / (BlinkCnt+1) LCD 中断间隔= (BlinkCnt+1)*(1/LCD 帧频率)

15.10.3. 中断清除寄存器 (LCD_INTCLR)

Address offset:0x08

Reset value:0x0400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	INTF_CLR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
					R1W0										

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	Reserved
10	INTF_CLR	R1W0	1	中断标志清除，写 0 清除， 写 1 无效
9:0	Reserved	-	-	Reserved

15.10.4. 输出配置寄存器 (LCD_POEN0)

Address offset:0x0C

Reset value:0xFFFFFFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Sx	RW	0xFFFFFFFF	Segx 输出控制位 0: SEG 输出使能 1: SEG 输出关闭，可以使用其他功能，如 IO,模拟输入输出

15.10.5. 输出配置寄存器 1 (LCD_POEN1)

Address offset:0x10

Reset value:0x1FFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	MUX	Cx				SxCy				S			
			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12	MUX	RW	1	SEG32~SEG35 端口功能选择，详细参考选择表格
11:8	Cx	RW	0xF	COMx 输出控制位（com0_com3） 0: COM 输出使能 1: COM 输出关闭，可以使用其他功能，如 IO,模拟输入输出
7:4	SxCy	RW	0xF	Segx/COMy 输出控制位 0: SEG/COM 输出使能 1: SEG/COM 输出关闭，可以使用其他功能，如 IO,模拟输入输出 SEG COM 引脚功能选择由 CR0.DUTY 决定
3:0	S	RW	0xF	Segx 输出控制位 0: SEG 输出使能 1: SEG 输出关闭，可以使用其他功能，如 IO,模拟输入输出

VLCDxSEGxPAD			寄存器如何配置				
			MUX	S<35:32>	BSEL		
VLCDXSEGXPAD 选择 GPIO，LCD disable（LCD_ON =0）时		VLCDHSEG35=IO VLCDHSEG34=IO VLCDHSEG33=IO VLCDHSEG32=IO	1	1111	X	X	X
VLCDXSEGXPAD 选择IO，LCD enable（LCD_ON =1）时，这时只能选择内部电阻工作模式	小电阻（大电流）模式	VLCDHSEG35=IO VLCDHSEG34=IO VLCDHSEG33=IO VLCDHSEG32=IO	1	1111	1	1	0
	中电阻（中电流）模式	VLCDHSEG35=IO VLCDHSEG34=IO VLCDHSEG33=IO VLCDHSEG32=IO	1	1111	0	1	0
	大电阻（小电流）模式	VLCDHSEG35=IO VLCDHSEG34=IO VLCDHSEG33=IO VLCDHSEG32=IO	1	1111	1	0	0
选择内部电阻工作模式	小电阻（大电流）模式	VLCDHSEG35=SEG35/IO VLCDHSEG34=SEG34/IO VLCDHSEG33=SEG33/IO VLCDHSEG32=SEG32/IO	0	XXXX	1	1	0
	中电阻（中电流）模式	VLCDHSEG35=SEG35/IO VLCDHSEG34=SEG34/IO VLCDHSEG33=SEG33/IO VLCDHSEG32=SEG32/IO	0	XXXX	0	1	0
	大电阻（小电流）模式	VLCDHSEG35= SEG35/IO VLCDHSEG34= SEG34/IO VLCDHSEG33= SEG33/IO VLCDHSEG32= SEG32/IO	0	XXXX	1	0	0
选择外部电阻工作模式，内部电阻短路		VLCDHSEG35=VOUT VLCDHSEG34=VLCD3 VLCDHSEG33=VLCD2 VLCDHSEG32=VLCD1	0	1111	0	0	0

图 15-20 寄存器配置

15.10.6. LCD_RAM0~7

Address offset:0x14~0x30

Reset value:0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	Dx	RW	0	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG COM 交叉点不亮； 1 对应的 SEG COM 交叉点亮；

15.10.7. LCD_RAM8~F

Address offset:0x34~0x50

Reset value:0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	D7	D6	D5	D4	D3	D2	D1	D0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7:0	Dx	RW	0	LCD 点输出，显示参考 LCD 显示模式 0 对应的 SEG COM 交叉点不亮； 1 对应的 SEG COM 交叉点亮；

15.10.8. LCD 寄存器映像

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	LCD_CR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CONTRAST				BSEL			DUTY			BIAS	Res.	Res.	LCD_CLK		EN
	Reset value																	0	0	0	0	0	0	0	0	1	1	0			0	1	0
0x04	LCD_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTF	DMAEN	IE	MODE	Res.	BLINKEN	BLINKCNT					
	Reset value																					0	0	0	0		0	0	0	0	0	0	0
0x08	LCD_INTCLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res	INTF_CLR	Res.									
	Reset value																						1										
0x0C	LCD_POEN0	S31	S30	S29	S28	S27	S26	S25	S24	S23	S22	S21	S20	S19	S18	S17	S16	S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
	Reset value	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
0x10	LCD_POEN1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MUX	C3	C2	C1	C0	S39/C4	S38/C5	S37/C6	S36/C7	S35	S34	S33	S32
	Reset value																				1	1	1	1	1	1	1	1	1	1	1	1	1
0x14 - 0x30	LCD_RAM0~7	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

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
	Reset value	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
0x34 - 0x50	LCD_RAM8~F	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	D7	D6	D5	D4	D3	D2	D1	D0
	Reset value																									0	0	0	0	0	0	0	0

16. 比较器 (COMP)

16.1. 简介

芯片内集成 2 个通用比较器（general purpose comparators）COMP，分别是 COMP1 和 COMP2。这 2 个模块可以作为单独的模块，也可以与 timer 组合在一起使用。

比较器可以被如下使用：

- 被模拟信号触发，产生低功耗模式唤醒功能
- 模拟信号调节
- 当与来自 timer 的 PWM 输出连接时，Cycle by cycle 的电流控制回路

16.2. COMP 主要特性

- 每个比较器有可配置的正或者负输入，以实现灵活的电压选择
  - 多路 I/O pin
  - 电源 VCC
  - 温度传感器的输出
  - 内部参考电压和通过分压提供的 3 个分数值（1/4、1/2、3/4）
- 迟滞功能可配置
- 可编程的速度和功耗
- 输出可以被连接到 I/O 或者 timer 的输入作为触发
  - OCREF_CLR 事件（cycle by cycle 的电流控制）
  - 为快速 PWM shutdown 的刹车
- COMP1 和 COMP2 可以组合成 window COMP
- 每个 COMP 具有中断产生能力，用作芯片从低功耗模式（sleep 和 stop 模式）的唤醒（通过 EXTI）

16.3. COMP 功能描述

16.3.1. COMP 框图

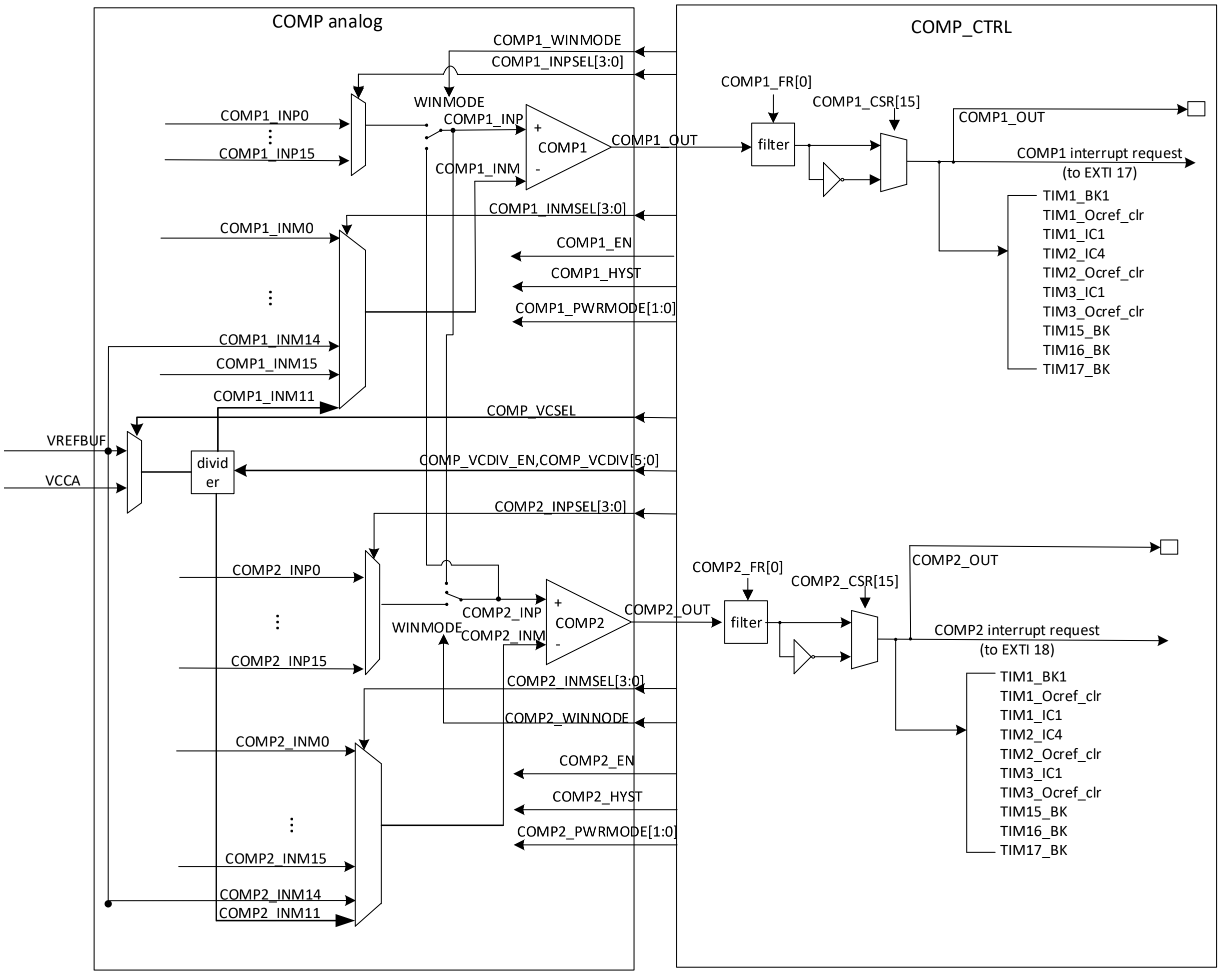


图 16-1 COMP 框图

16.3.2. COMP 管脚和内部信号

用作比较器输入的 I/O，必须在 GPIO 寄存器中被配置成模拟模式。

比较器输出可以通过在 GPIO 的复用功能通道（alternate function）连接到 I/O pin。

输出也可以在内部连接到各种 timer 的输入，达到以下目的:

- 连接刹车输入时，PWM 信号的紧急 shut-down
- 使用 OCREF_CLR 输入的 Cycle-by-cycle 电流控制
- 时序测量的输入捕获

16.3.3. COMP 复位和时钟

COMP 模块有两个时钟源:

- PCLK（APB clock），用于给配置寄存器提供时钟
- COMP 时钟，用于模拟比较器输出后的电路（模拟输出的锁存电路、滤毛刺电路等）的时钟，可选择为 PCLK 或者 LSI。当需要在 stop 模式下工作时，选择 LSI。

注意:

1) PCLK 和 COMP CLK 由 RCC_APBENR2 控制 bit 同时使能，该使能关闭则 PCLK 和 COMP CLK 关闭，若使能则 PCLK 和 COMP CLK 同时开启。

进入 Stop 模式前，建议先配置 COMP CLK 为 LSI 或 LSE，然后进入 stop 模式。

COMP 模块的复位信号包含 APB 复位源和 COMP 模块软件复位源：

1) APB 复位，用于 COMP 寄存器的复位
2) COMP 软件复位，用于模拟比较器输出后电路（模拟输出的锁存电路、滤毛刺电路等）的复位

注意：

1) 当 RCC_APBRSTR2 中的复位信号使能后，COMP 模块 PRESETn 和 COMP_RSETn 信号都会被复位。

16.3.4. Window 比较器

Window 比较器的作用是监测模拟电压是否在低和高阈值范围内。

可以使用两个比较器创建 window 比较器。被监测的模拟电压同时连接到两个比较器的 non-inverting（+端）输入，高阈值和低阈值分别连接到两个比较器的 inverting 输入端（-端）。

通过使能 WINMODE 位，可以将两个比较器的 non-inverting（+输入端）连接到一起，起到节省一个 I/O pin 的作用。

注意：两个 COMP 的 WINMODE 模式不能同时使能。

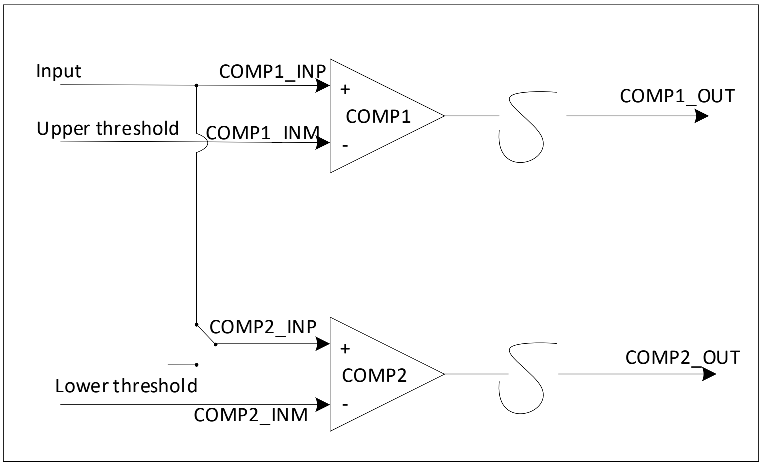


图 16-2 window comparator

16.3.5. 迟滞

为避免在噪声信号情况产生假的输出转换，比较器可以使能带有迟滞的功能（COMP1、COMP2 有独立的迟滞功能使能信号 COMP1_HYST,COMP2_HYST）。

16.3.6. 功耗模式

比较器的功耗和传输延迟可以通过 COMPx_CSR 寄存器的 PWRMODE[1:0]位来选择不同模式，以实现在特定应用的最适合的 trade-off。可选的模式包括 high speed 和 medium speed 两种，相对而言 high speed mode 下的功耗更大，传输延迟也更小。注意，当进入 stop 之前，如果选择 PWR_CR2 寄存器 LPR=1（即选择用 low power regulator 供电），则需要首先设定 COMP 在 Medium speed(PWRMODE=01)。

此外，为降低功耗，APB 时钟和 COMP 时钟被 RCC_APBENR2.COMP1EN，RCC_APBENR2.COMP2EN 控制，软件可在使用 COMP 模块时，才使能该寄存器。

16.3.7. 比较器滤波

可以通过设定 COMP_FR 寄存器，使能 COMP 的输出滤波功能及相应的滤波宽度。注意该设定应在 COMP_EN 使能前完成。

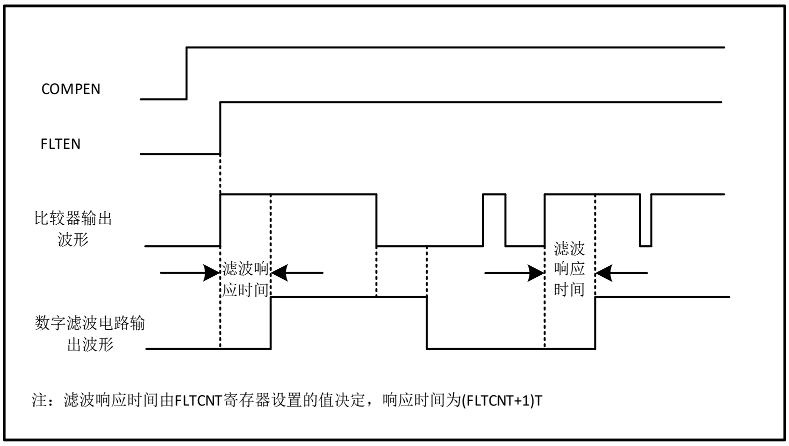


图 16-3 COMP filter

16.3.8. COMP 中断

比较器输出在芯片内部连接到 EXTI 控制器（extended interrupts and events）。每个比较器有单独的 EXTI line（17 和 18），并能够产生中断或者事件。相同的机制被用作从低功耗的唤醒。

16.4. COMP 寄存器

16.4.1. COMP1 控制和状态寄存器(COMP1_CSR)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	COMP_OUT	Res	Res	COMP_VCSEL	COMP_VCDIV_EN	COMP_VCDIV[5:0]						PWR-MODE[1:0]		Res	COMP1_HYST
	R			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	Res	Res	Res	WINMODE	Res	INPSEL[3:0]				INNSEL[3:0]				Res	COMP1_EN
RW				RW		RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31	Reserved		0	
30	COMP_OUT	R	0	COMP1 输出状态 该位只读，它反映了 COMP1 在经过极性选择的输出电平。
29：28	Reserved		0	
27	COMP_VCSEL	RW	0	COMP1,COMP2 参考电压 Vref 选择 0：VCC 1：ADC 参考电压

Bit	Name	R/W	Reset Value	Function
26	COMP_VCDIV_EN	RW	0	分压使能 1：使能 0：不使能
25:20	COMP_VCDIV[5:0]	RW	0	分压选择 00_0000: 1/64 Vref 00_0001: 2/64 Vref 00_0010: 3/64 Vref … 11_1110: 63/64 Vref 11_1111: Vref
19:18	PWRMODE[1:0]	RW	0	COMP1 功耗模式选择 选择了功耗和由此而来的 COMP1 的速度 00：High speed(250uA) 01：Medium speed(5uA) 10：reserved 11：reserved
17	Reserved			
16	COMP1_HYST	RW	0	COMP1 迟滞功能使能控制 0：无迟滞 1：迟滞电压约 20mV
15	POLARITY	RW	0	COMP1 输出极性选择 软件可读可写 0：不反向 1：反向
14:12	Reserved		0	
11	WINMODE	RW	0	COMP1 不反向的输出选择（window 模式） 软件可读可写 0：信号被 INPSEL[3:0]选择 1：COMP2 的 COMP2_INP 信号 注意两个 COMP 的 WINMODE 模式不能同时使能。
10	Reserved		0	
9:6	INPSEL[3:0]	RW	0	COMP1 非反相输入的信号选择 0000：COMP1_INP0 from PC0 0001：COMP1_INP1 from PC1 0010：COMP1_INP2 from PC2 0011：COMP1_INP3 from PC3 0100：COMP1_INP4 from PA0 0101：COMP1_INP5 from PA1 0110：COMP1_INP6 from PA2 0111：COMP1_INP7 from PA3 1000：COMP1_INP8 from PA4 1001：COMP1_INP9 from PA5 1010：COMP1_INP10 from PA6 1011：COMP1_INP11 from PA7 1100：COMP1_INP12 from PB4 1101：COMP1_INP13 from PB5 1110：COMP1_INP14 from PB6
5:2	INNSEL[3:0]	RW	0	COMP1 反相输入的信号选择 0000：COMP1_INM0 from PA0

Bit	Name	R/W	Reset Value	Function
				（注：IO 映射表中 COMP1_INM0 对应 COMP1_INN0） 0001：COMP1_INM1 from PA1 （注：IO 映射表中 COMP1_INM1 对应 COMP1_INN1） 0010：COMP1_INM2 from PA2 （注：IO 映射表中 COMP1_INM2 对应 COMP1_INN2） 0011：COMP1_INM3 from PA3 （注：IO 映射表中 COMP1_INM3 对应 COMP1_INN3） 0100：COMP1_INM4 from PA4 （注：IO 映射表中 COMP1_INM4 对应 COMP1_INN4） 0101：COMP1_INM5 from PA5 （注：IO 映射表中 COMP1_INM5 对应 COMP1_INN5） 0110：COMP1_INM6 from PA6 （注：IO 映射表中 COMP1_INM6 对应 COMP1_INN6） 0111：COMP1_INM7 from PA7 （注：IO 映射表中 COMP1_INM7 对应 COMP1_INN7） 1000：COMP1_INM8 from PC4 （注：IO 映射表中 COMP1_INM8 对应 COMP1_INN8） 1001：COMP1_INM9 from PC5 （注：IO 映射表中 COMP1_INM9 对应 COMP1_INN9） 1011：COMP1_INM11 from 电阻分压电压 1100：COMP1_INM12 from TS_VIN (温度传感器电压) 1101：COMP1_INM13 from VREF1P2 (内部基准 1.2V 输出电压) 1110：COMP1_INM14 from VREFBUF (需要使能 ADC 模块的 VREFBUFFERE bit 和 Verfbuff_sel[1:0]) 1111：COMP1_INM15 from OPA1_VIN
1	Reserved		0	
0	COMP1_EN	RW	0	COMP1 使能位 软件可读可写（如果没有被锁定） 0：Disable 1：Enable

16.4.2. COMP1 滤波寄存器(COMP1_FR)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT1[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1[15:0]	RW	0	比较器 1 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved		0	

0	FLTEN1	RW	0	比较器 1 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注: 该位必须在 COMP1_EN 为 0 时置位

16.4.3. COMP2 控制和状态寄存器(COMP2_CSR)

Address offset:0x10

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	COMP_OUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PWR-MODE[1:0]		Res	Res
	R											RW	RW		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POLARITY	Res	Res	Res	WINMODE	Res	INPSEL[3:0]				INNSEL[3:0]				COMP2_HYST	COMP2_EN
RW				RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved		0	
30	COMP_OUT	R	0	COMP2 输出状态 该位只读，它反映了 COMP2 在经过极性选择的输出电平。
29：20	Reserved		0	
19:18	PWRMODE[1:0]	RW	0	COMP2 功耗模式选择 选择了功耗和由此而来的 COMP2 的速度 00：High speed 01：Medium speed 10：reserved 11：reserved
17:16	Reserved			
15	POLARITY	RW	0	COMP2 输出极性选择 软件可读可写 0：不反向 1：反向
14:12	Reserved		0	
11	WINMODE	RW	0	COMP2 不反向的输出选择 软件可读可写 0：信号被 INPSEL[1:0]选择 1：COMP2 的 COMP2_INP 信号 注意两个 COMP 的 WINMODE 模式不能同时使能。
10	Reserved		0	
9:6	INPSEL[3:0]	RW	0	COMP2 非反相输入的信号选择，软件可读可写 0000：COMP2_INP0 from PA0 0001：COMP2_INP1 from PA1 0010：COMP2_INP2 from PA2 0011：COMP2_INP3 from PA3

Bit	Name	R/W	Reset Value	Function
				0100：COMP2_INP4 from PA4 0101：COMP2_INP5 from PA5 0110：COMP2_INP6 from PB1 0111：COMP2_INP7 from PB2 1000：COMP2_INP8 from PB10 1001：COMP2_INP9 from PB12 1010：COMP2_INP10 from PB13 1011：COMP2_INP11 from PB14 1100：COMP2_INP12 from PB4 1101：COMP2_INP13 from PB6 （注：IO 映射表中 COMP2_INP13 对应 COMP2_INP14） 1110：COMP2_INP14 from PB7 （注：IO 映射表中 COMP2_INP14 对应 COMP1_INP15）
5:2	INNSEL[3:0]	RW	0	COMP2 反相输入的信号选择 0000：COMP2_INM0 from PC0 （注：IO 映射表中 COMP2_INM0 对应 COMP2_INN0） 0001：COMP2_INM1 from PC1 （注：IO 映射表中 COMP2_INM1 对应 COMP2_INN1） 0010：COMP2_INM2 from PC2 （注：IO 映射表中 COMP2_INM2 对应 COMP2_INN2） 0011：COMP2_INM3 from PC3 （注：IO 映射表中 COMP2_INM3 对应 COMP2_INN3） 0100：COMP2_INM4 from PA0 （注：IO 映射表中 COMP2_INM4 对应 COMP2_INN4） 0101：COMP2_INM5 from PA1 （注：IO 映射表中 COMP2_INM5 对应 COMP2_INN5） 0110：COMP1_INM6 from PB0 （注：IO 映射表中 COMP2_INM6 对应 COMP2_INN6） 0111：COMP1_INM7 from PB1 （注：IO 映射表中 COMP2_INM7 对应 COMP2_INN7） 1000：COMP2_INM8 from PB2 （注：IO 映射表中 COMP2_INM8 对应 COMP2_INN8） 1001：COMP2_INM9 from PB3 （注：IO 映射表中 COMP2_INM9 对应 COMP2_INN9） 1011：COMP2_INM11 from 电阻分压电压 1100：COMP2_INM12 from TS_VIN (温度传感器电压) 1101：COMP2_INM13 from VREF1P2 (内部基准 1.2V 输出电压) 1110：COMP2_INM14 from VREFBUF (需要使能 ADC 模块的 VREFBUFFERE bit 和 Verfbuff_sel[1:0]) 1111：COMP2_INM15 from OPA2_VIN (OPA2 输出电压)
1	COMP2_HYST	RW	0	COMP2 迟滞功能使能控制

Bit	Name	R/W	Reset Value	Function
				0：无迟滞 1：迟滞电压约 20mV
0	COMP2_EN	RW	0	COMP2 使能位 软件可读可写 0：Disable 1：Enable

16.4.4. COMP2 滤波寄存器(COMP2_FR)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT2[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2[15:0]	RW	0	比较器 2 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved		0	
0	FLTEN2	RW	0	比较器 2 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注: 该位必须在 COMP2_EN 为 0 时置位

16.4.5. COMP 寄存器映像

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	COMP1_CSR	Res.	COMP_OUT	Res.	Res.	COMP_VCSEL	COMP_VCDIV_EN	COPM_VCDIV[5:0]						PWRMODE[1:0]		Res.	COMP1_HYST	POLARITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL[3:0]				INNSEL[3:0]				Res.	COMP1_EN
	Reset value		0			0	0							0	0		0	0				0		0	0	0	0	0	0	0	0		0
0x04	COMP1_FR	FLTCNT1[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																0
0x10	COMP2_CSR	Res.	COMP_OUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWR-MODE[1:0]		Res.	Res.	POLARITY	Res.	Res.	Res.	WINMODE	Res.	INPSEL[3:0]				INNSEL[3:0]				COMP2_HYST	COMP2_EN

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
	Reset value		0											0	0			0				0		0	0	0	0	0	0	0	0	0	0
0x14	COMP2_FR	FLTCNT2[15:0]																Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																0

17. 运算放大器(OPA)

17.1. OPA 简介

OPA 模块适用于简易放大器应用。内部的 2 个运放可以使用外部电阻进行级联。 OPA 的输入范围是 0V 到 VCCA，输出范围是 0.1V 到 VCCA-0.2V。

17.2. OPA 主要特性

- 2 个独立配置运放
- OPA 的输入范围是 0 到 VCCA，输出范围是 0.1V 到 VCCA-0.2V 可编程增益
- 可配置为以下模式
  - 通用运放模式

17.3. OPA 功能描述

2 个 OPA 可以通过使用外部元件组成放大器来放大小信号模拟输入信号，输出为放大后的信号。

17.3.1. OPA 框图

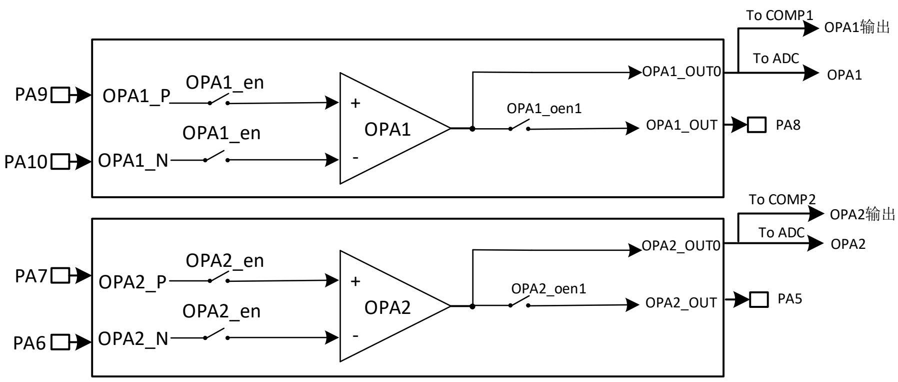


图 17-1 OPA 架构框图

17.4. OPA 寄存器

17.4.1. OPA 输出使能寄存器(OPA_CR0)

Address offset:0x30

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res				OP3OEN1	Res				OP2OEN1	Res				OP1OEN1	Res
				RW					RW					RW	

Bit	Name	R/W	Reset Value	Function
31:12	Res	-		
11	OP3OEN1	RW	0	OP3 输出 1 使能

10:7	Res	-		
6	OP2OEN1	RW	0	OP2 输出 1 使能
5:2	Res	-		
1	OP1OEN1	RW	0	OP1 输出 1 使能
0	Res	-		

17.4.2. OPA 控制寄存器(OPA_CR1)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res									EN2	EN1	Res				
									RW	RW					

Bit	Name	R/W	Reset Value	Function
31:7	Res	-		
6	EN2	RW	0	OPA2 使能
5	EN1	RW	0	OPA1 使能
4:0	Res	-		

17.4.3. OPA 寄存器映像

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	OPA_CR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OP3OEN1.	Res.	Res.	Res.	Res.	OP2OEN1.	Res.	Res.	Res.	Res.	OP1OEN1.	Res.
	Reset value																					0					0					0	
0x34	OPA_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EN3.	EN2.	EN1.	Res.	Res.	Res.	Res.	Res.
	Reset value																									0	0	0					

18.硬件除法器(DIV)

18.1. DIV 简介

DIV（Divider）是一个 32 位有符号/无符号整数硬件除法器。

18.2. DIV 主要特性

- 支持 32 位除法
- 当前除法未运行完毕时，寄存器中的数据不可改变
- 可配置有符号/无符号整数除法计算
- 32 位被除数， 32 位除数
- 输出 32 位商和 32 位余数
- 除数为零警告标志位，除法运算结束标志位
- 8 个时钟周期完成一次除法运算
- 写除数寄存器触发除法运算开始
- 写完除数后，读商寄存器和余数等寄存器时，需要先等待计算完成标志 DIV_END
- 除数为 0 时，商和余数结果为 0

18.3. DIV 功能描述

18.3.1. DIV 操作流程

a) 在 RCC 系统时钟控制器里打开硬件除法器的模块时钟使能位。
b) 配置寄存器 DIV_SIGN 设置有符号/无符号除法运算。
c) 配置寄存器 DIV_DEND 设置被除数。
d) 配置寄存器 DIV_SOR 设置除数，除法运算开始。
e) 查询寄存器 DIV_STAT 中的运算结束标志位 DIV_END， DIV_END 为 1 标志运算结束。读寄存器 DIV_QUOT 得到商，读寄存器 HDIV_REMD 得到余数。
f) 当除数为零时，除法运算立即结束，商和余数同时置为 0，同时除数为零警告标志位 DIV_ZERO 被置起。
g) 在除法运算结束之前，读寄存器 DIV_QUOT/DIV_REMD 时， CPU 将读出上一次计算值

举例：计算一个无符号除法，被除数为 1917887483(0x7250A3FB)，除数为 9597(0x257D)
步骤一，配置寄存器 DIV_SIGN 为 0，即无符号除法运算
步骤二，配置寄存器 DIV_DEND 为 0x7250A3FB， 即设置被除数
步骤三，配置寄存器 DIV_SOR 为 0x257D， 即设置除数，计算开始
步骤四，查询寄存器 DIV_STAT 运算结束标志位 DIV_END， DIV_END 为 1 标志，
运算结束。
读寄存器 DIV_QUOT 得到商 199842(0x30CA2)
读寄存器 DIV_REMD 得到余数 3809(0xEE1)

18.4. DIV 寄存器

18.4.1. DIV 被除数寄存器(DIV_DEND)

Address offset:0x00

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_DEND	RW	0	被除数寄存器

18.4.2. DIV 除数寄存器(DIV_SOR)

Address offset:0x04

Reset value:0x0000 0001

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_SOR	RW	1	除数寄存器（写此寄存器自动触发除法运算）

18.4.3. DIV 商寄存器(DIV_QUOT)

Address offset:0x08

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_QUOT	RW	0	存储除法器计算得到的商

18.4.4. DIV 余数寄存器(DIV_REMD)

Address offset:0x0C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DIV_REMD	RW	0	存储除法器计算得到的余数

18.4.5. DIV 符号寄存器(DIV_SIGN)

Address offset:0x10

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved			
0	SIGN	RW	0	符号选择寄存器 0: 无符号除法运算 1: 有符号除法运算

18.4.6. DIV 状态寄存器(DIV_STAT)

Address offset:0x14

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved			
1	DIV_ZERO	R	0	除数为零警告标志位 0: 除数不为零 1: 除数为零
0	DIV_END	R	0	除法运算结束标志位 0: 运算进行中 1: 运算结束

18.4.7. DIV 寄存器映像

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	DIV_DEND	DIV_DEND[31:0]																															
	Reset value	0																															
0x	DIV_SOR	DIV_SOR[31:0]																															

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
04	Reset value	1																															
0x08	DIV_QUOT	DIV_QUOT[31:0]																															
	Reset value	0																															
0x0C	DIV_REMD	DIV_REMD [31:0]																															
	Reset value	0																															
0x10	DIV_SIGN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SIGN
	Reset value																																0
0x14	DIV_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_ZERO.	DIV_END.
	Reset value																															0	0

19.高级控制定时器（TIM1）

19.1. TIM1 简介

高级 timer（TIM1）由 16 位被可编程分频器驱动的自动装载计数器组成。它可以被用作各种场景，包括：输入信号（输入捕获）的脉冲长度测量，或者产生输出波形（输出比较、输出 PWM、带死区插入的互补 PWM）。

脉冲长度和波形周期可以使用定时器分频器和 RCC 时钟控制分频器，实现从微秒到毫秒的调制。高级 timer（TIM1）和通用（TIMx）timer 是完全独立的，不共享任何资源。他们可以同步操作。

19.2. TIM1 主要特性

- 16bit 向上、向下或者向上向下的自动重装载计数器
- 16bit 可以实时修改的可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频（on the fly）
- 多达 4 个独立的通道
  - 输入捕获
  - 输出比较
  - PWM 产生（边缘或者中心对齐模式）
  - 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连的同步电路
- 重复计数器，在计数指定周期数后，才更新时间寄存器
- 刹车输入可以将定时器的输出信号置为复位状态和已知状态
- 中断/DMA 产生在以下事件
  - 更新：计数器向上、向下溢出，计数器初始化（通过软件或者内外部触发）
  - 触发事件
  - 输入捕获
  - 输出比较
  - 刹车输入
- 支持增量式的（正交）编码器和为定位用的霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

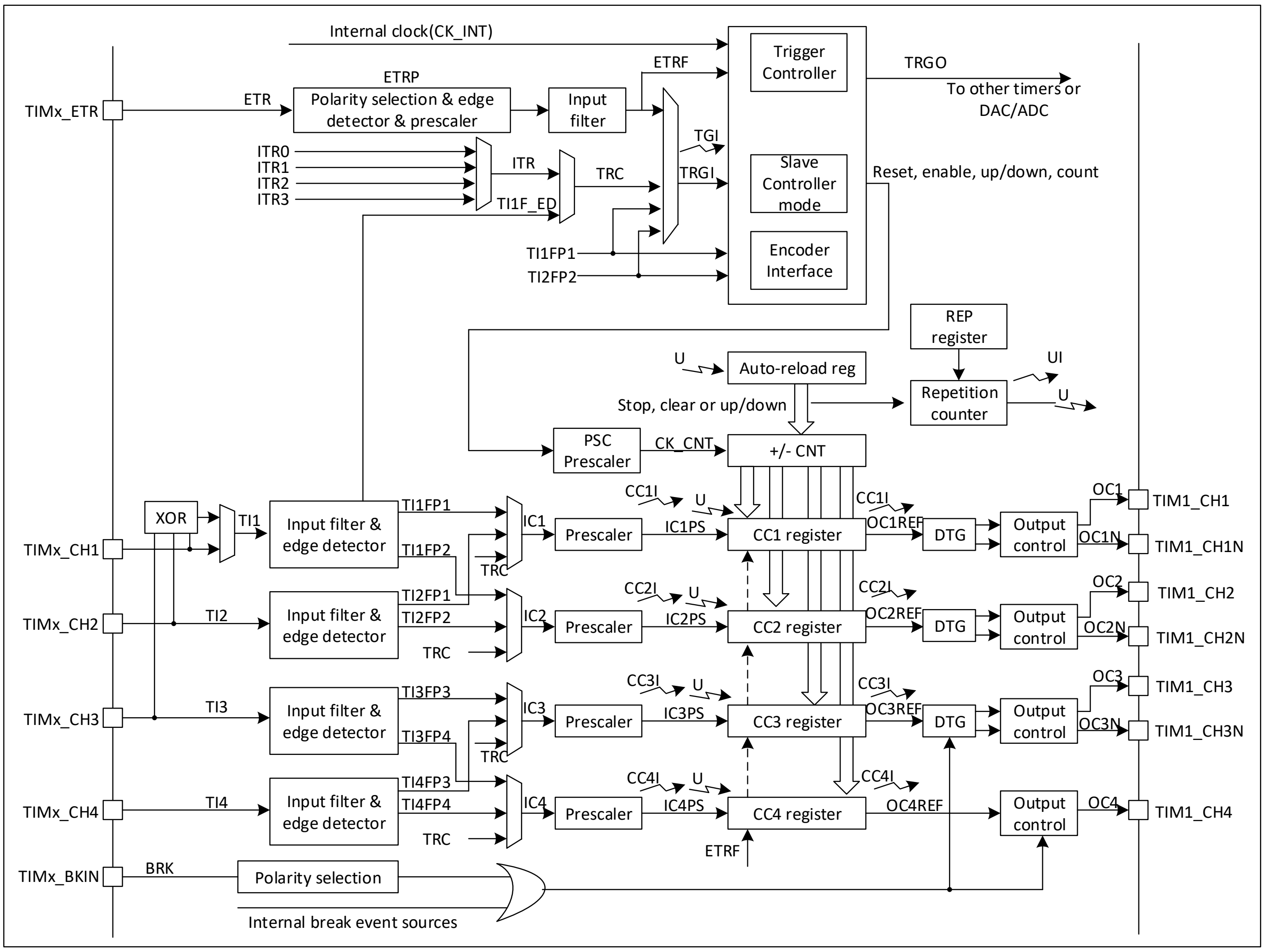


图 19-1 高级控制定时器架构框图

19.3. TIM1 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。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

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

时基单元包括：

- 计数器寄存器（TIM1_CNT）
- 预分频寄存器（TIM1_PSC）
- 自动装载寄存器（TIM1_ARR）
- 重复计数寄存器（TIM1_RCR）

自动装载寄存器是预先装载的，写或者读自动重载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到上溢出（向下计数器时的下溢条件）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。

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 TIM1_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 TIM1_CR 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按 1 到 65535 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 给出了在预分频器运行时，更改计数器参数的例子。

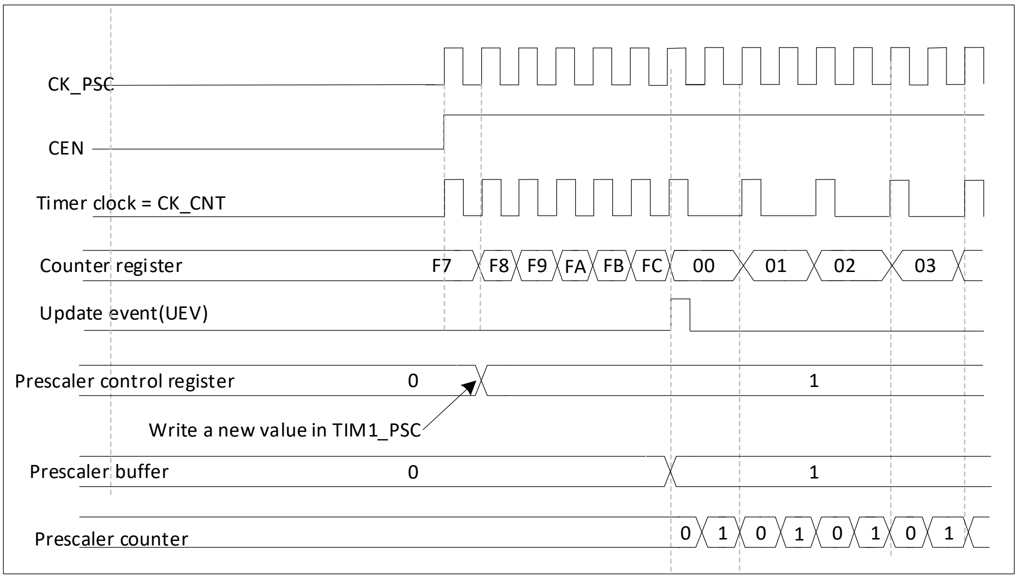


图 19-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

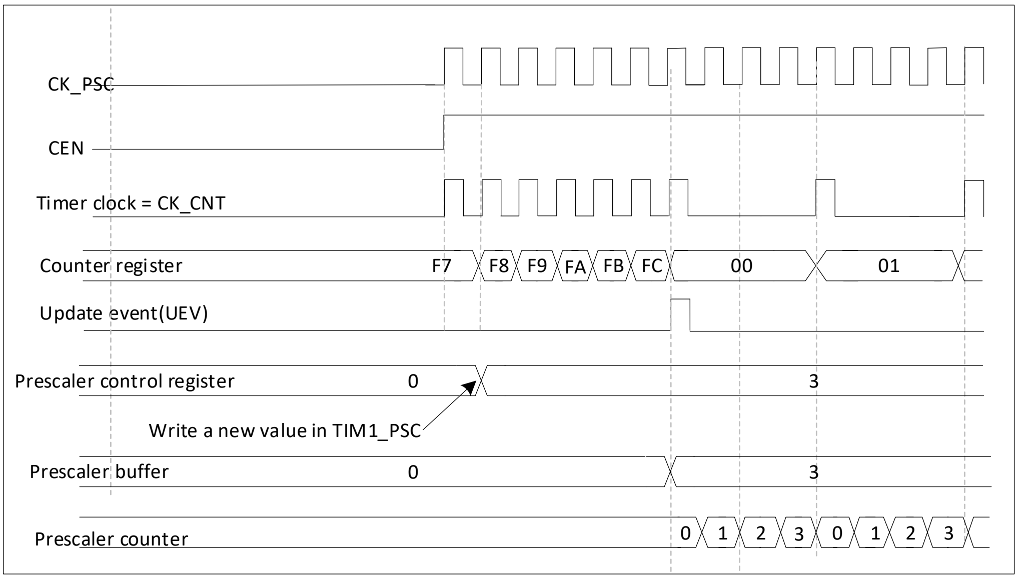


图 19-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

19.3.2. 计数器模式

向上计数模式

向上计数模式，是从 0 到自动装载值的计数器，然后又从 0 重新开始计数，并产生一个计数的溢出事件。

如果重复计数器被使用，则在向上计数器重复了重复计数寄存器(TIMx_RCR)中设定的次数后，将产生更新事件(UEV)，否则每次计数器上溢都会产生更新事件。

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。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 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 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

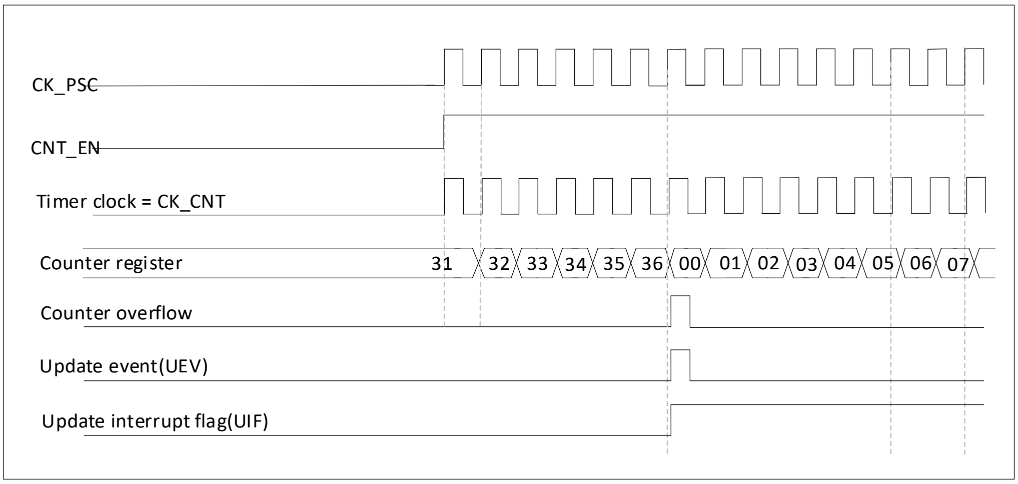


图 19-4 计数器时序图，内部时钟分频因子为 1

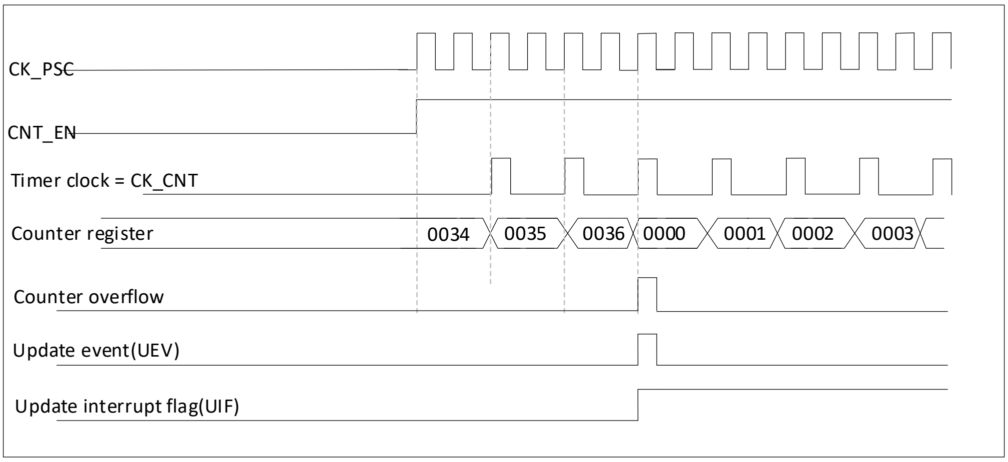


图 19-5 计数器时序图，内部时钟分频因子为 2

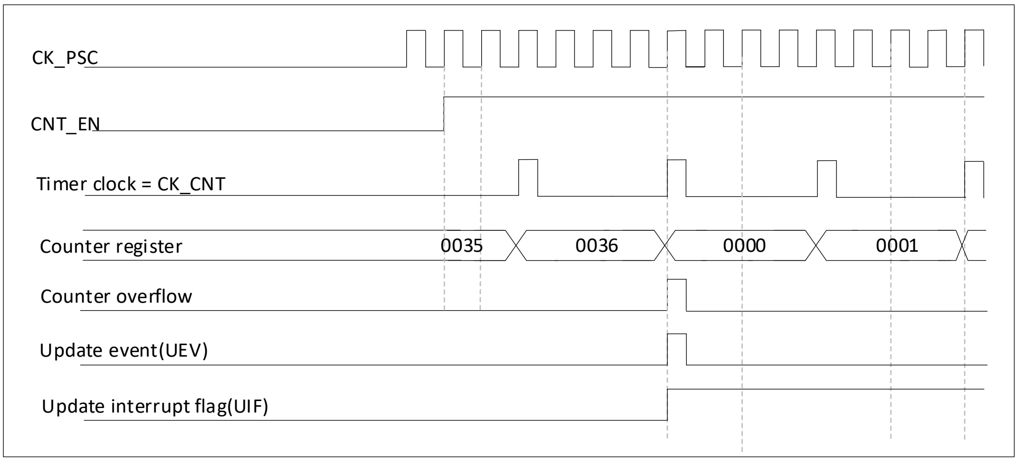


图 19-6 计数器时序图，内部时钟分频因子为 4

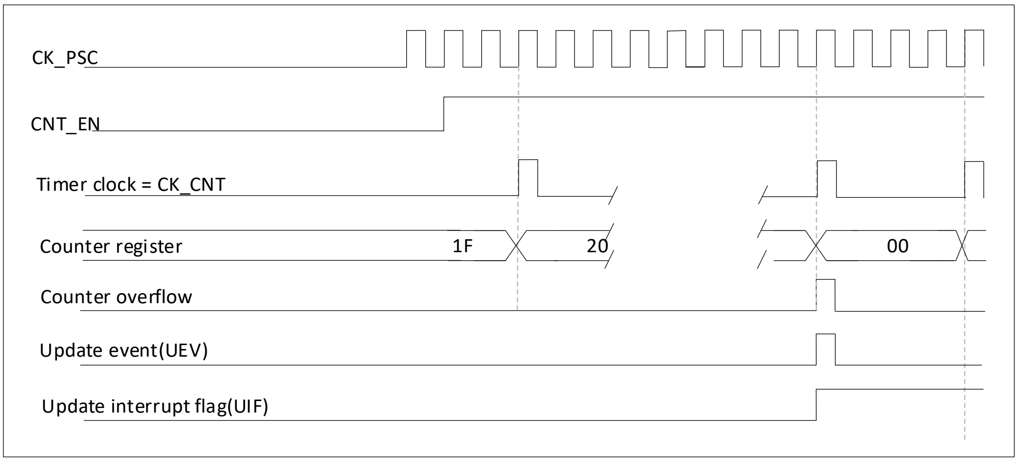


图 19-7 计数器时序图，内部时钟分频因子为 N

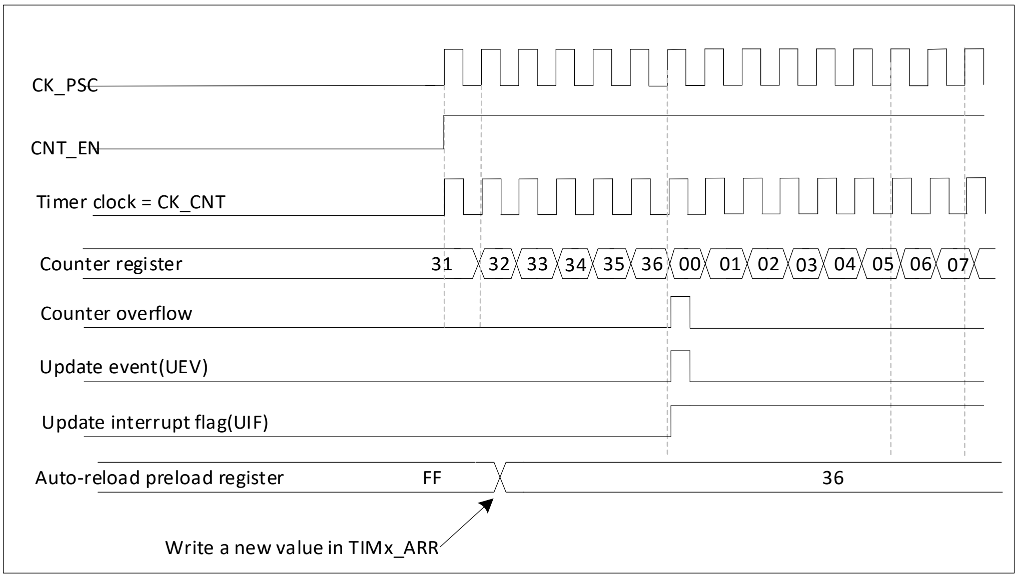


图 19-8 计数器时序图，当 ARPE=0 时的更新事件(TIM1_ARR 没有预装入)

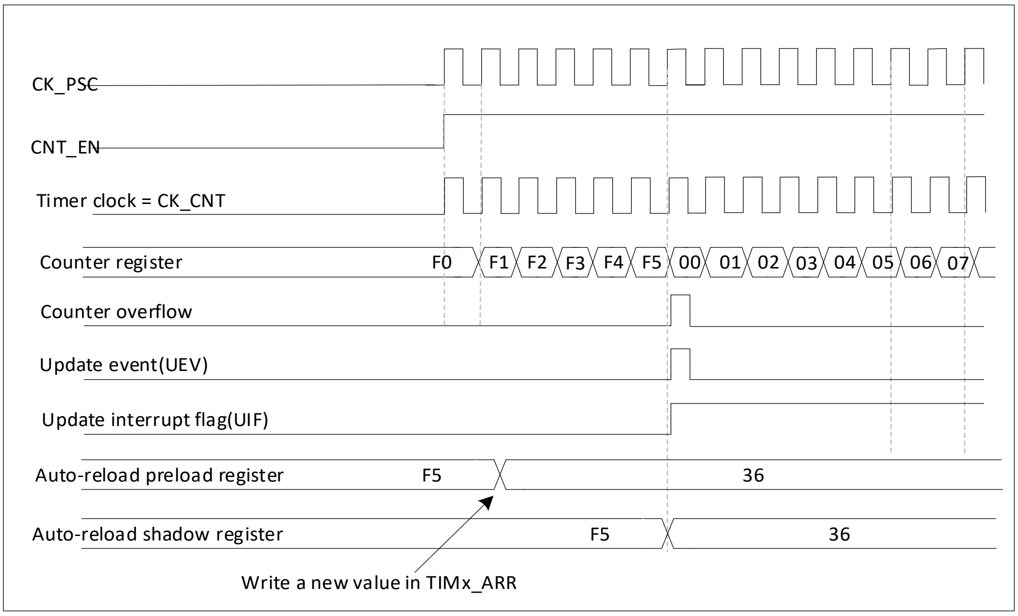


图 19-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIM1_ARR)

向下计数模式

向下计数模式，从自动装载的值开始向下计数到 0，然后重新开始从自动装载的值向下计数，并产生一个向下溢出事件。

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

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。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始(但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

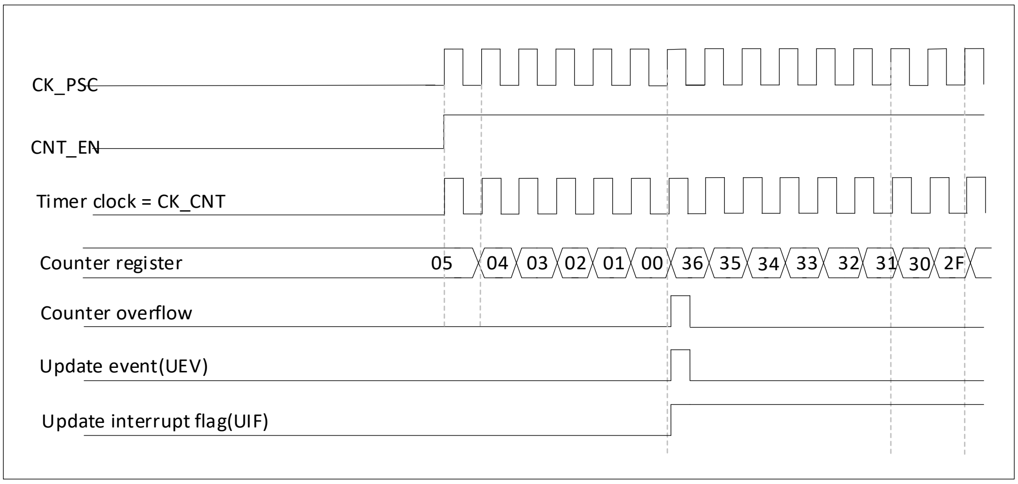


图 19-10 计数器时序图，内部时钟分频因子为 1

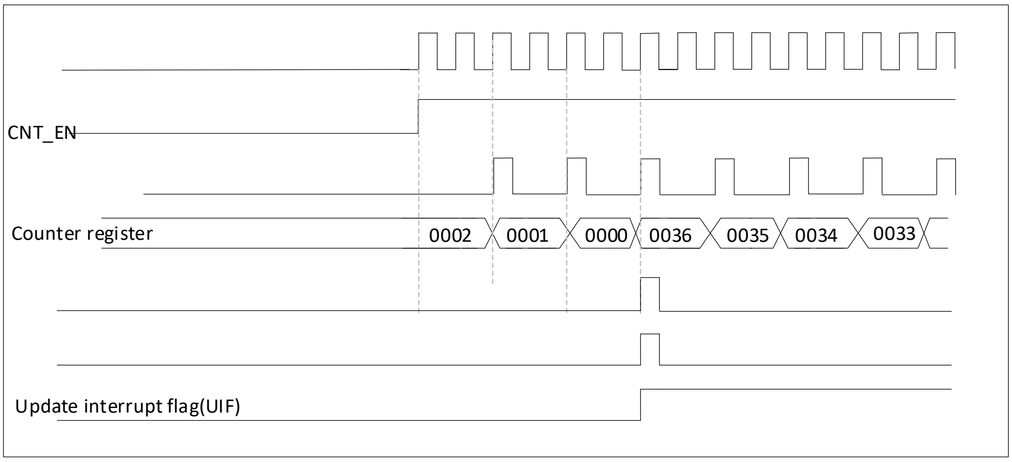


图 19-11 计数器时序图，内部时钟分频因子为 2

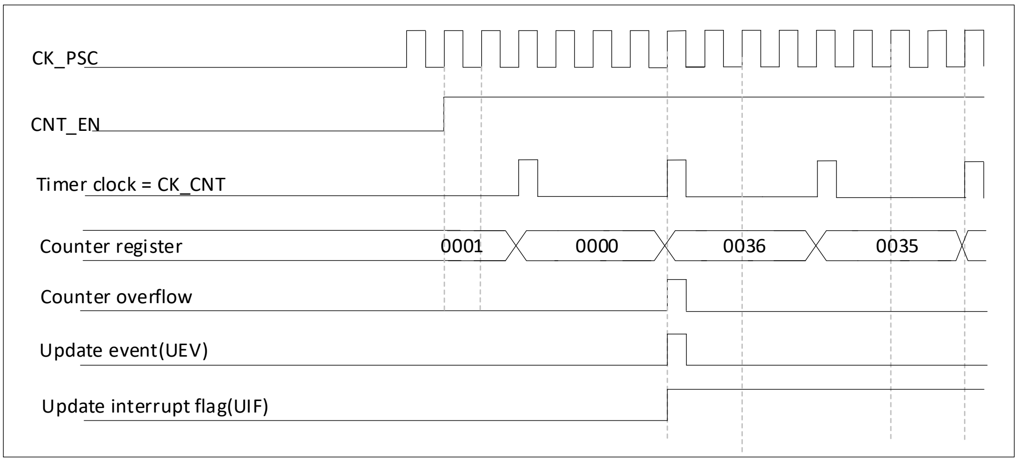


图 19-12 计数器时序图，内部时钟分频因子为 4

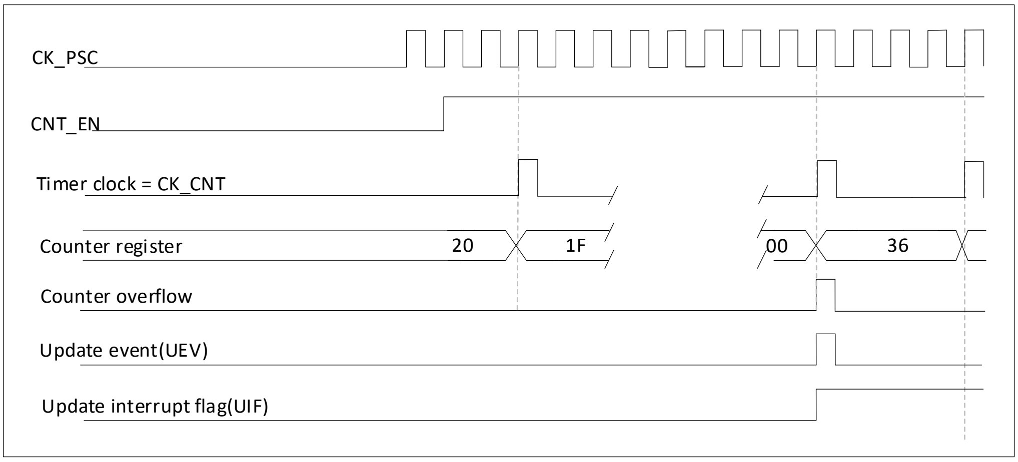


图 19-13 计数器时序图，内部时钟分频因子为 N

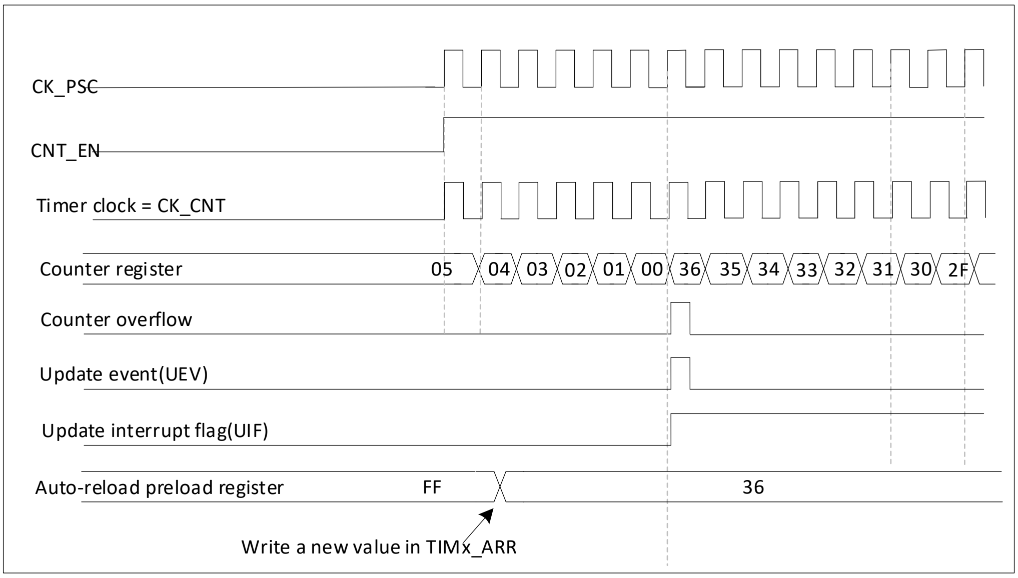


图 19-14 计数器时序图，当没有使用周期计数器时的更新事件

中央对齐模式 (向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

中央对齐模式在 TIMx_CR1 寄存器中的 CMS 不等于 0 时有效。通道在配置成输出模式时，输出比较中断标志将被置位，当：向下计数时（中央对齐模式 1，CMS=”01”）,向上计数时（中央对齐模式 2，CMS=”10”）向上向下计数（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。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)， 设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内
  注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)

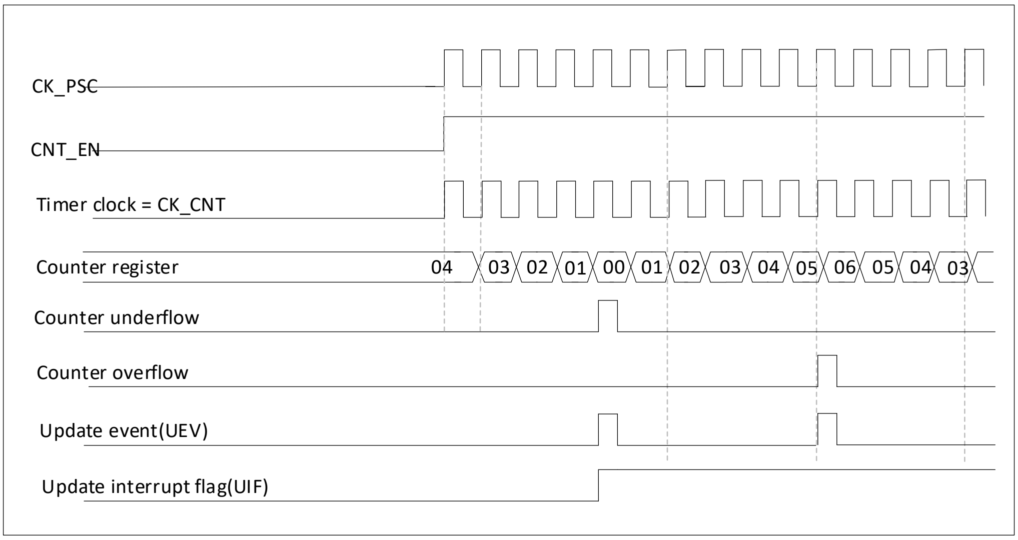


图 19-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR = 0x6

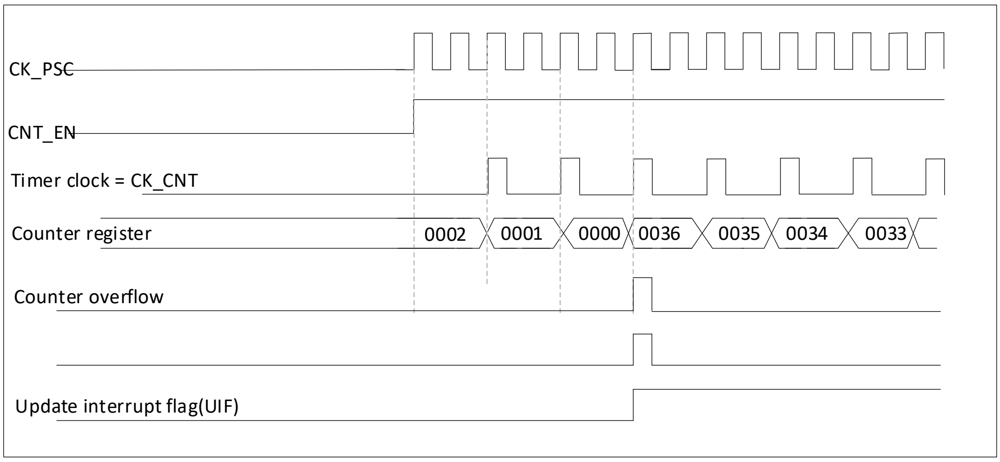


图 19-16 计数器时序图，内部时钟分频因子为 2, TIMx_ARR=0x36

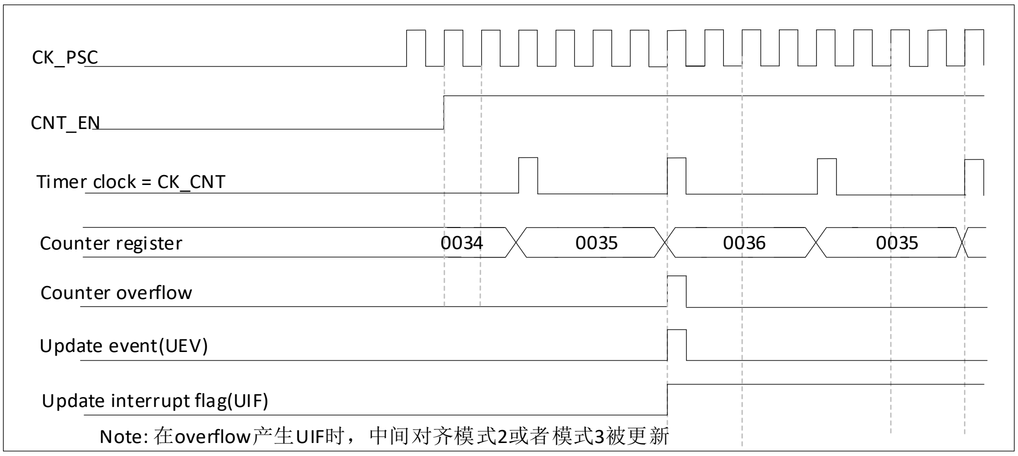


图 19-17 计数器时序图，内部时钟分频因子为 4, TIMx_ARR=0x36

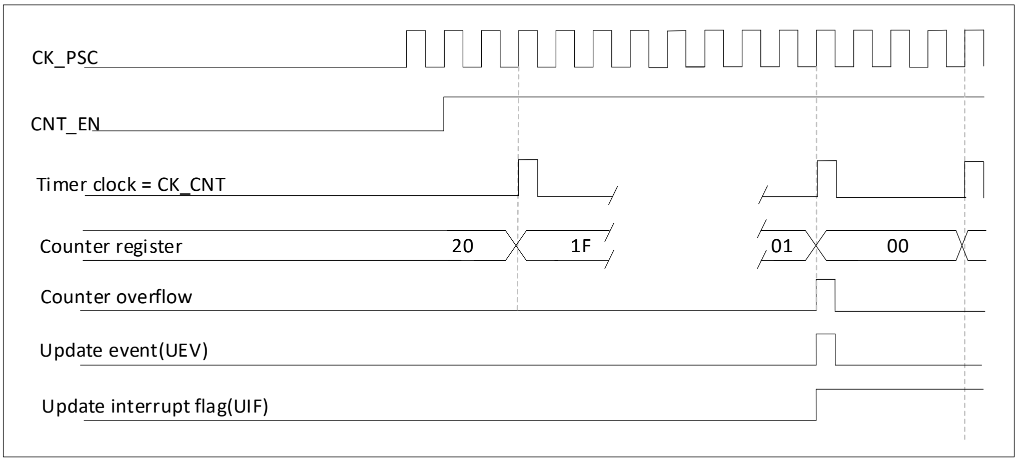


图 19-18 计数器时序图，内部时钟分频因子为 N

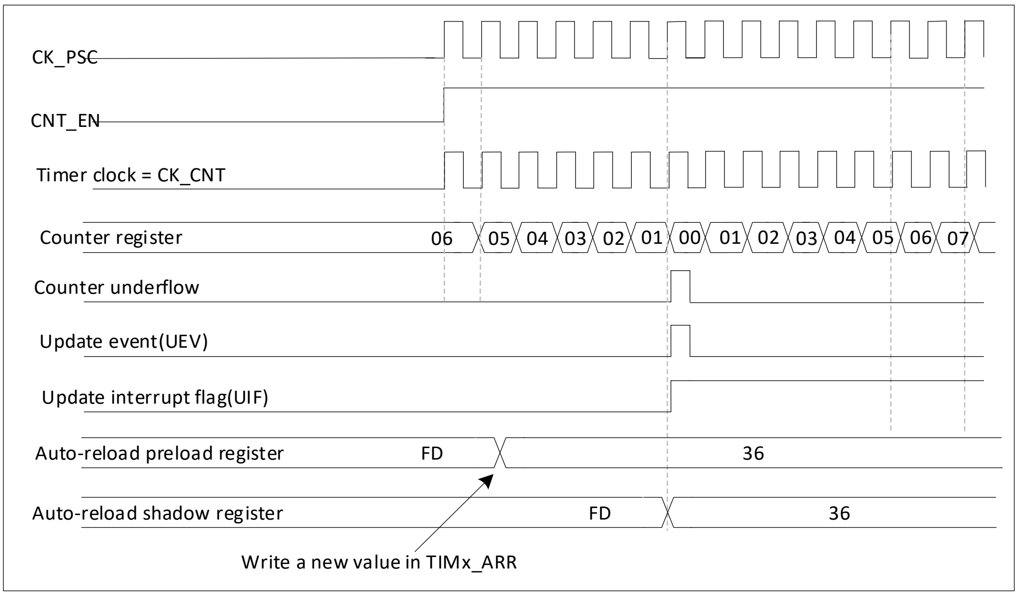


图 19-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

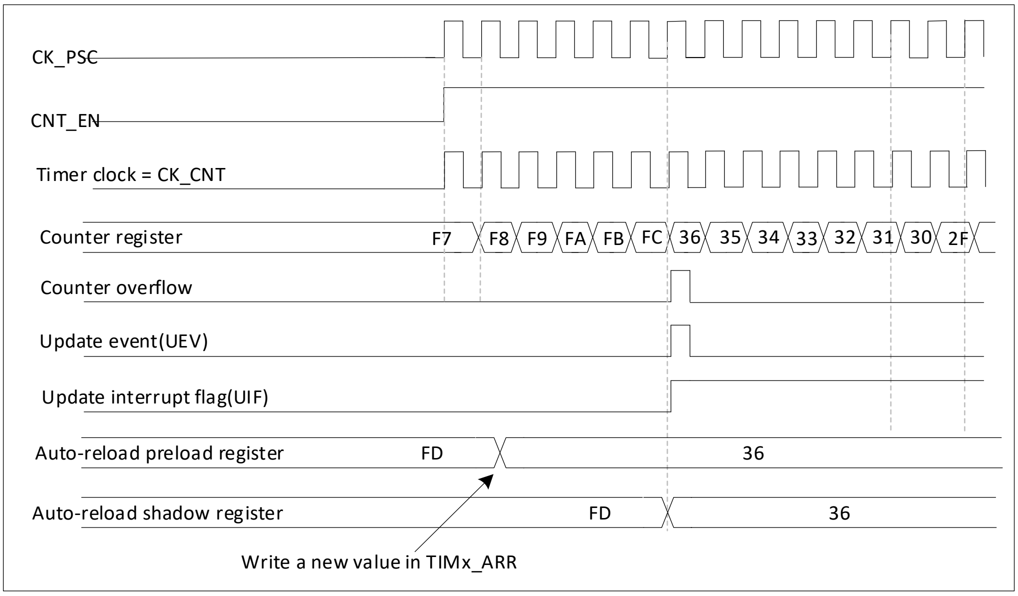


图 19-20 计数器时序图，ARPE=1 时的更新事件(计数器溢出)

19.3.3. 重复计数器

时基单元描述了关于计数器向上、向下溢出的更新事件（UEV）是如何产生的。它实际上仅当重复计数器计数到零才产生。这个特性对产生 PMW 信号非常有用。

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

重复计数器在下述任何一条件成立时递减：

- 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下溢时
- 中央对齐模式下每次上溢和每次下溢时。

虽然这样限制了 PWM 的最大循环周期位 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 2xTck。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值定义。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如到重复计数器。

在中央对齐模式下，对于 RCR 的奇数值，取决于当 RCR 寄存器被写入和当计数器开始，出现上溢、或者下溢，则更新事件产生。如果在启动计数器之前写 RCR，在上溢时产生更新事件。例如，对于 RCR = 3 时，更新事件被产生在第 4 个上溢或者下溢事件（取决于 RCR 被写入的值）。

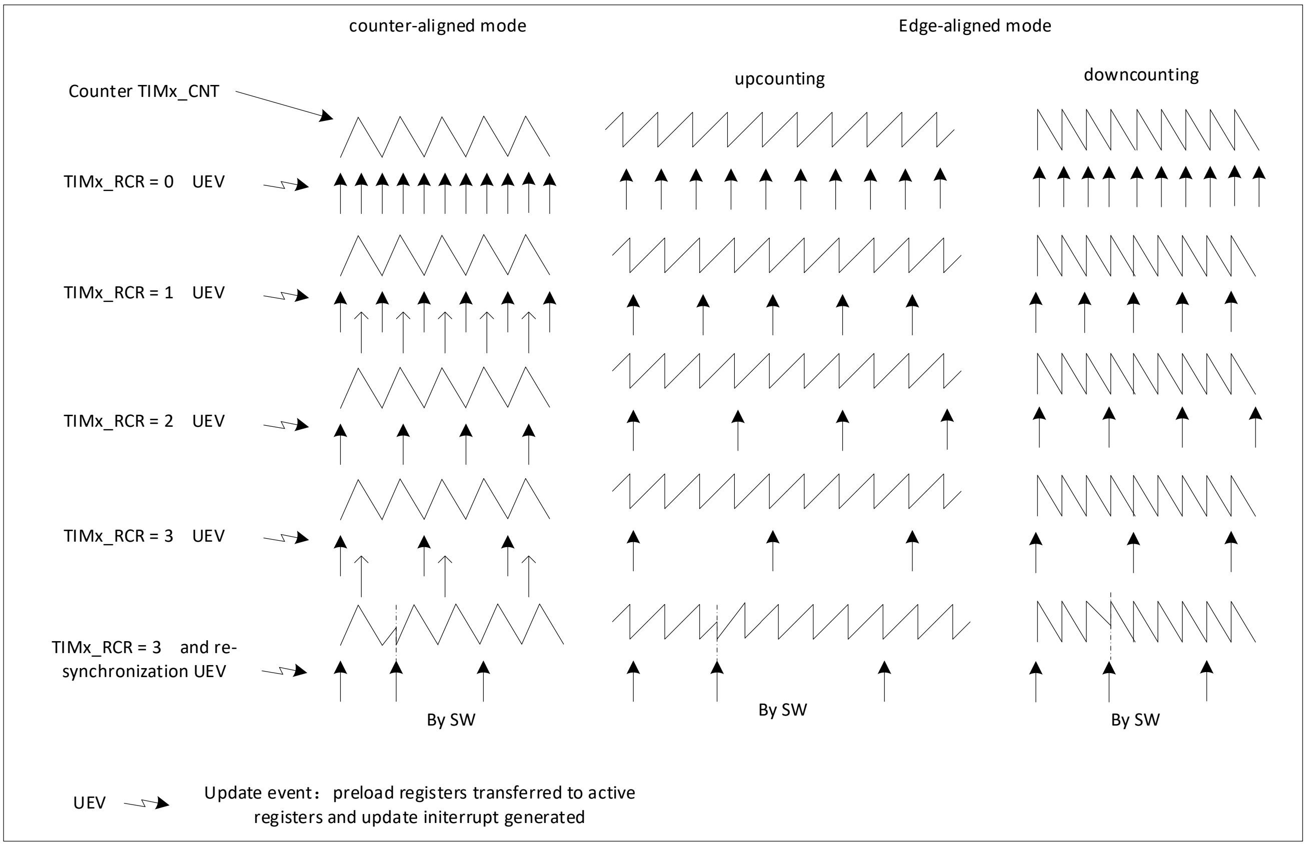


图 19-21 不同模式下更新速率的例子，及 TIM1_RCR 的寄存器设置

19.3.4. 时钟源

计数器的时钟可以由以下时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置一个定时器 Timer1 作为另一个定时器 Timer2 的预分频器。

内部时钟源（CK_INT）

如果从模式控制器被禁止，则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是事实上的控制位，并且只能被软件修改。只要 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作

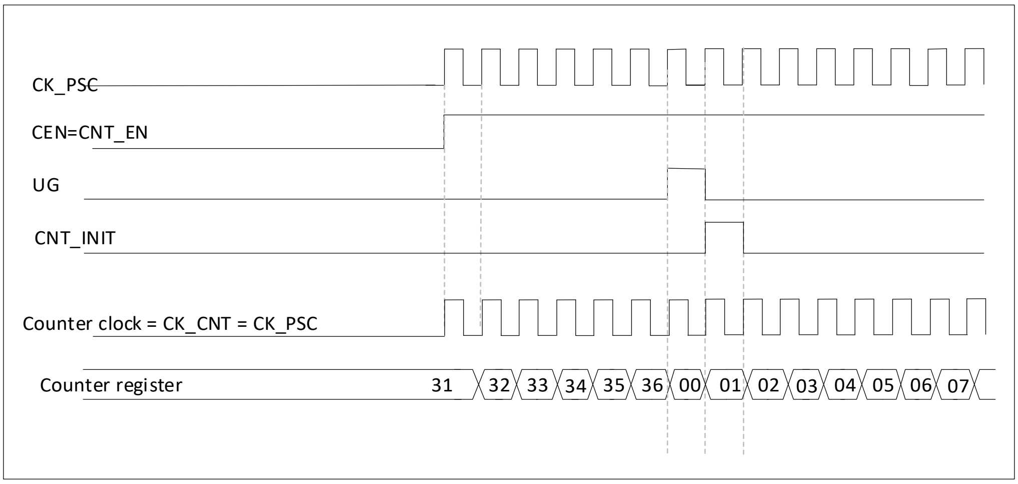


图 19-22 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

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

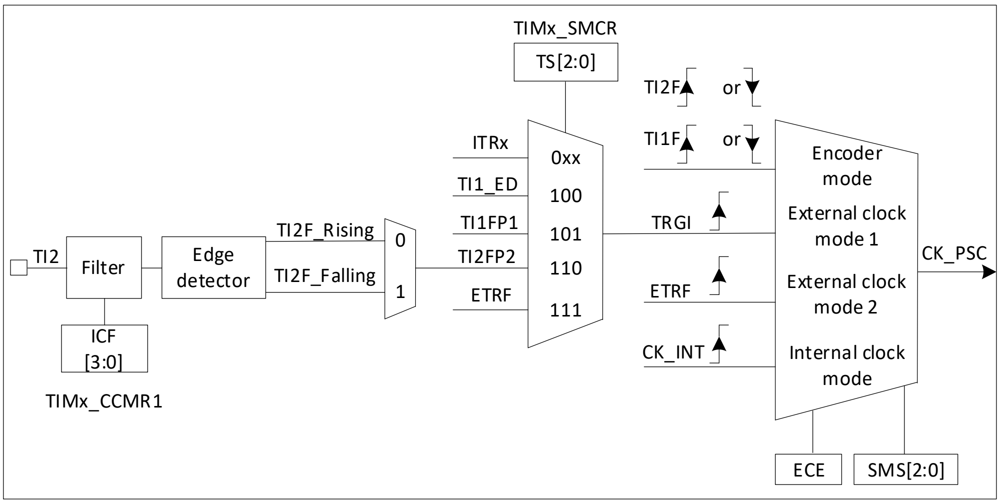


图 19-23 TI2 外部时钟连接例子

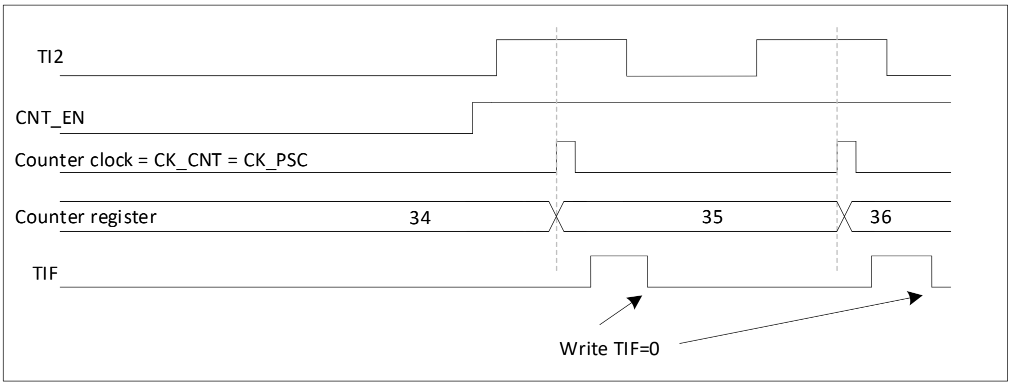


图 19-24 外部时钟模式 1 下的控制电路

外部时钟源模式 2

通过写 TIMx_SMCR 寄存器的 ECE 为 1，选定此模式。计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

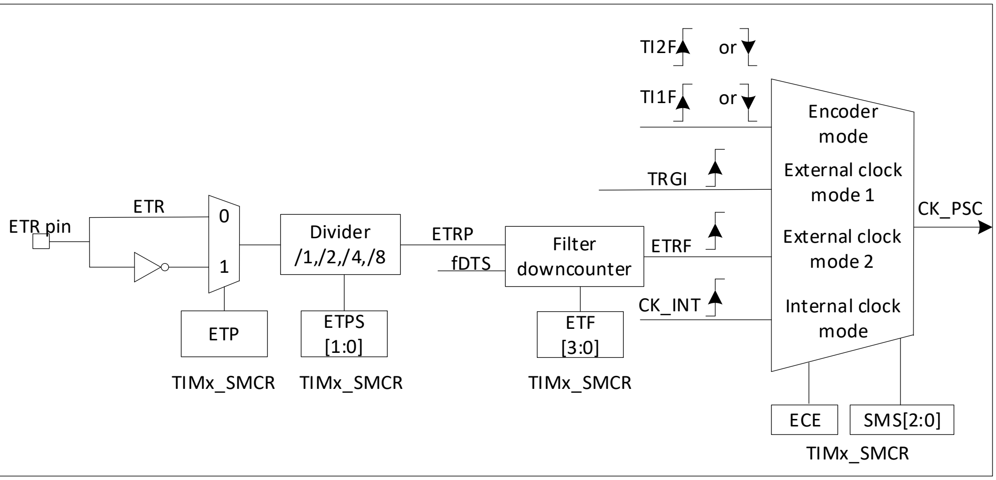


图 19-25 TI2 外部触发输入框图

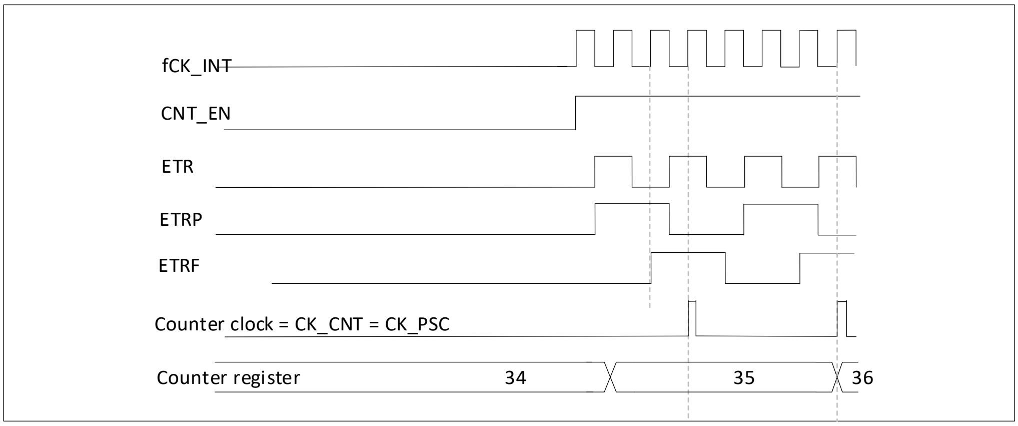


图 19-26 外部时钟模式 2 下的控制电路

19.3.5. 捕获/比较通道

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

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

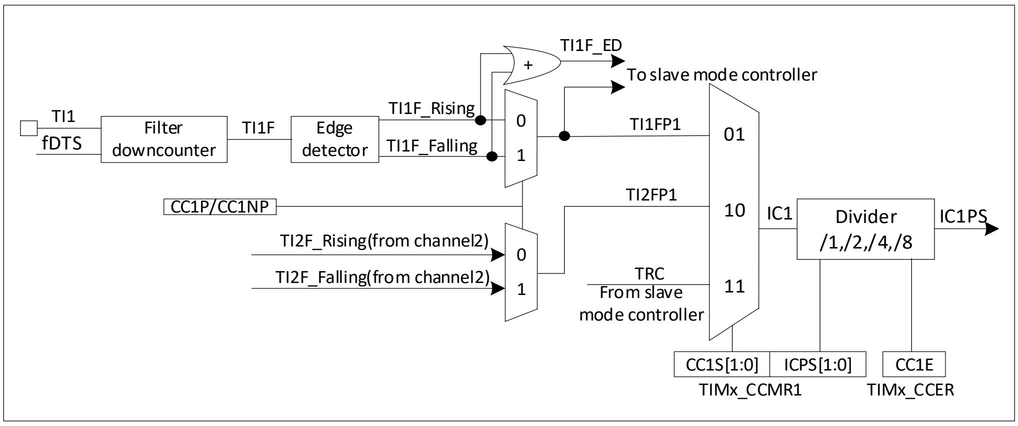


图 19-27 捕获/比较通道(如：通道 1 输入部分)

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。

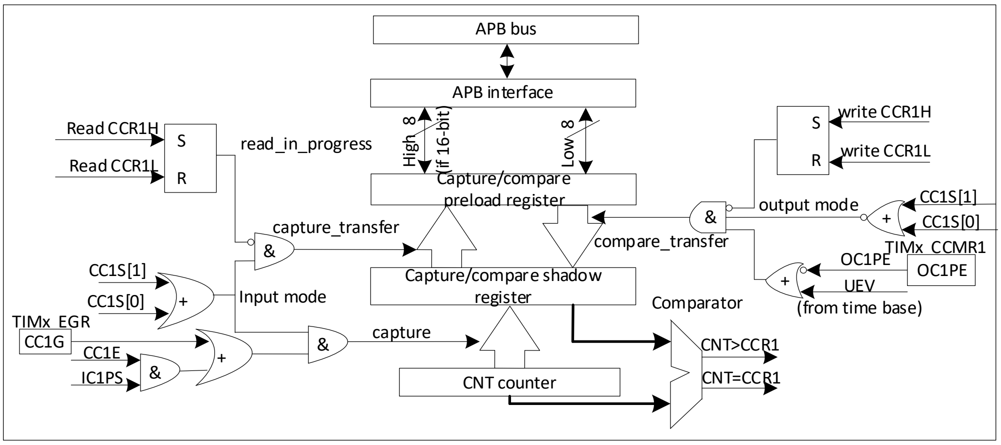


图 19-28 捕获/比较通道 1 的主电路

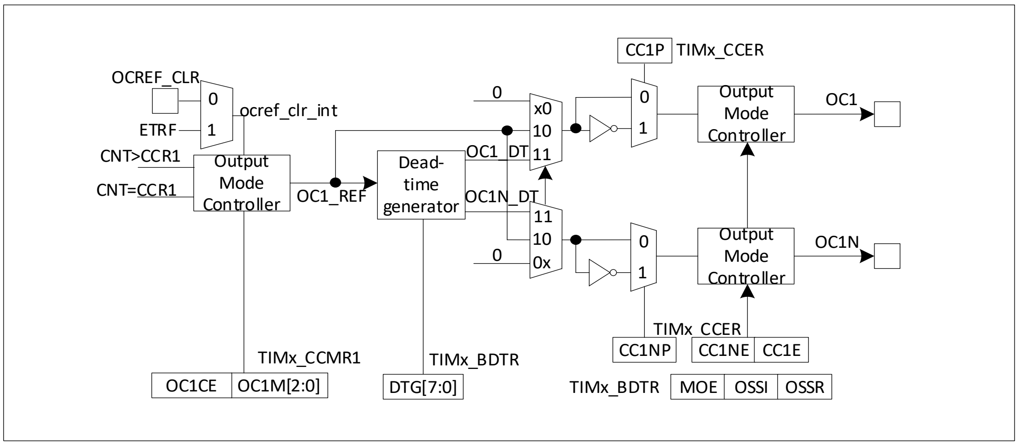


图 19-29 捕获/比较通道的输出部分(通道 1 至 3)

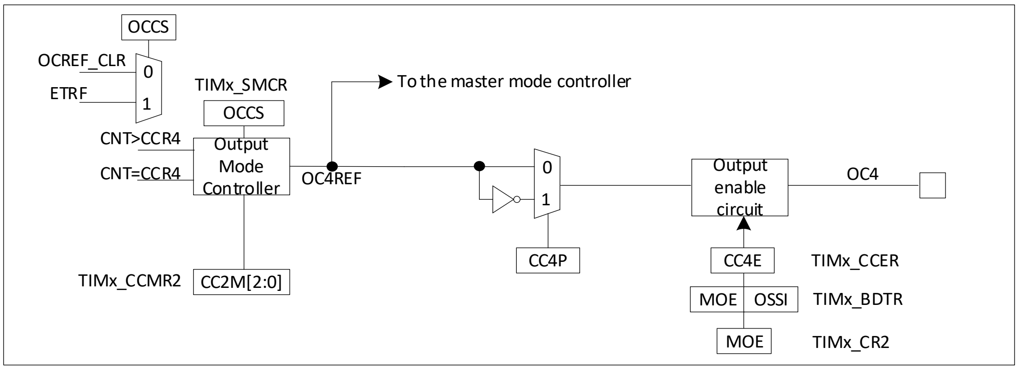


图 19-30 捕获/比较通道的输出部分(通道 4)

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。

19.3.6. 输入捕获模式

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

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

1) 选择有效输入端：TIMx_CCMR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为'00'，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
2) 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 Tix 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 IcxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 Fck_int 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
3) 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)
4) 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
5) 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
6) 如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

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

19.3.7. 输入捕获模式（PWM input mode）

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- 两个 Icx 信号被映射到同一个 Tix 输入。
- 这 2 个 Icx 信号为边沿有效，但是极性相反。
- 其中一个 TixFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

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

1) 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
2) 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)。
3) 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
4) 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
5) 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
6) 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7) 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

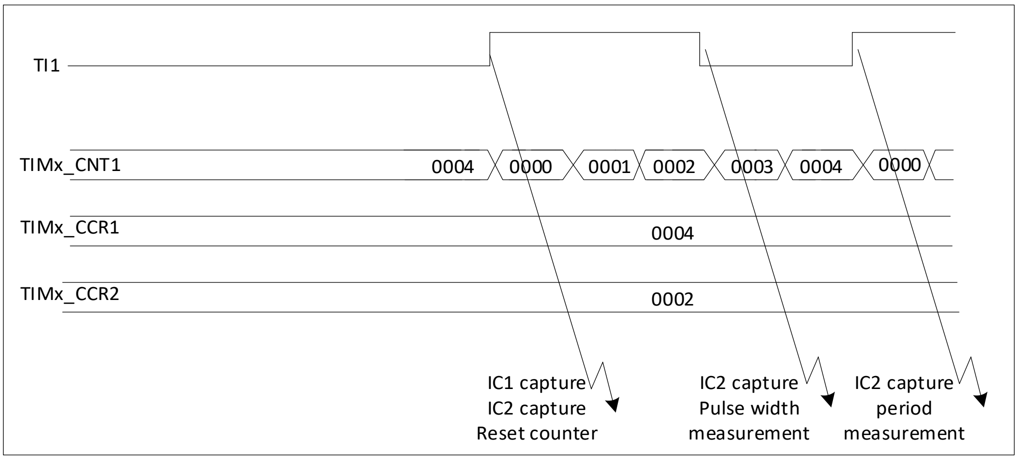


图 19-31 PWM 输入模式时序

19.3.8. 强置输出模式

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

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。 置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

19.3.9. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

1) 将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
2) 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CcxIF 位)。
3) 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CcxIE 位)，则产生一个中断。
4) 若设置了相应的使能位(TIMx_DIER 寄存器中的 CcxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。 在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

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

输出比较模式的配置步骤：

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CcxIE 位。
4. 选择输出模式，例如：
   — 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。

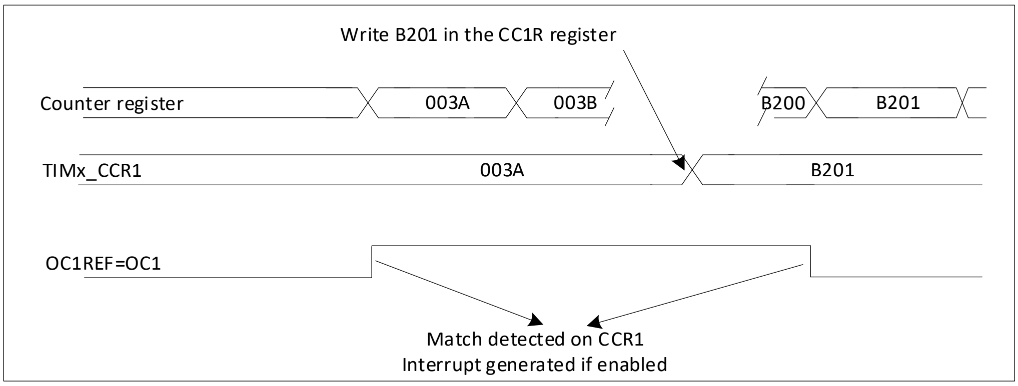


图 19-32 输出比较模式，翻转 OC1

19.3.10. PWM 模式

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

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

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

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

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- **向上计数配置**

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参看下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

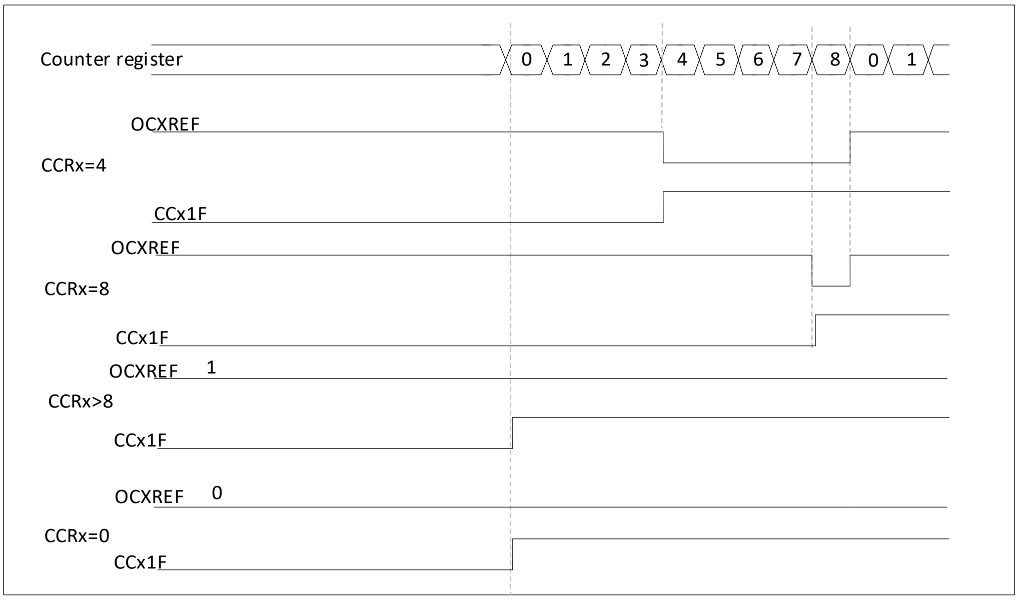


图 19-33 边沿对齐方式 PWM 输出，向上（ARR=8）

- **向下计数的配置**

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为'1'。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

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

下图给出一些中央对齐的 PWM 波形的例子

- TIMx_ARR = 8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中央对齐模式下，当计数器向下计数时设置比较标志

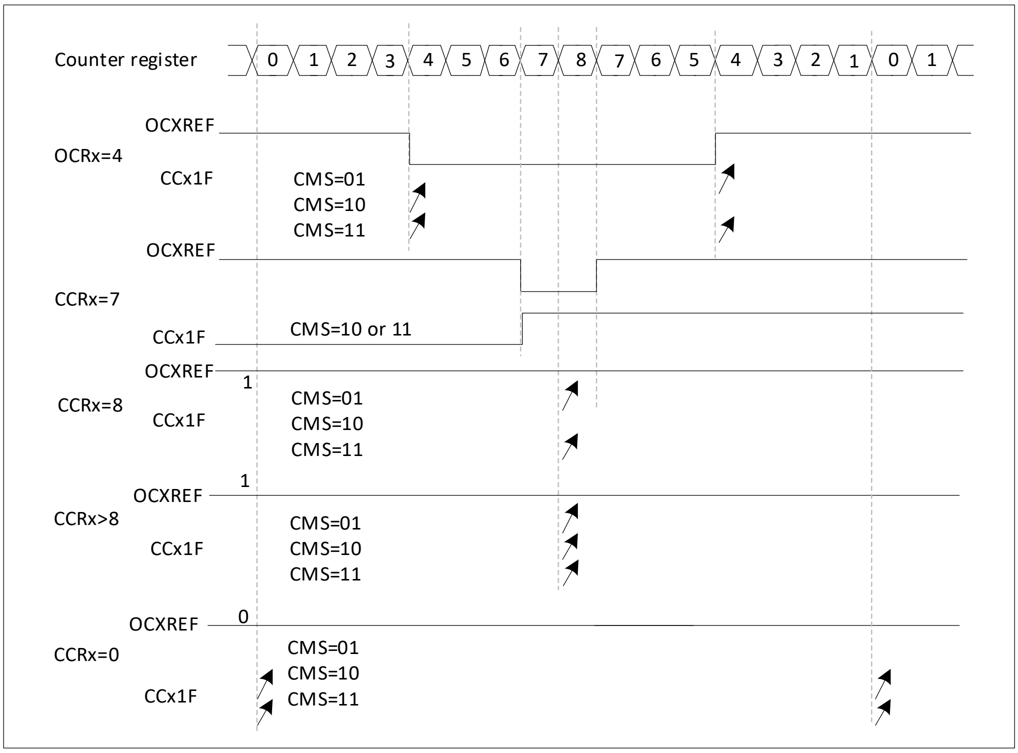


图 19-34 中央对齐的 PWM 波形(APR=8)

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。特别地： — 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。 — 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

19.3.11. 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关断和接通。 这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

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

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

同时设置 CcxE 和 CcxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 8 位的死区发生器 DTG[7:0]。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

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

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

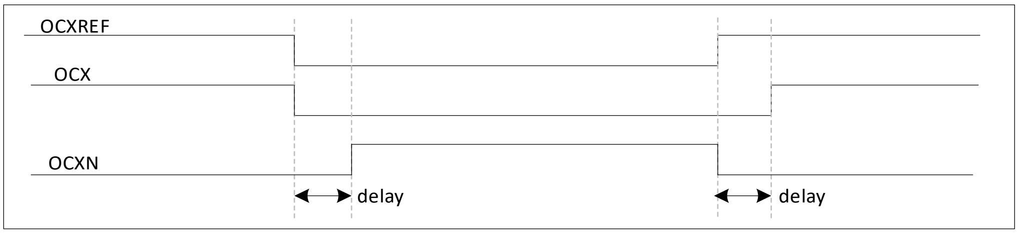


图 19-35 带死区插入的互补输出

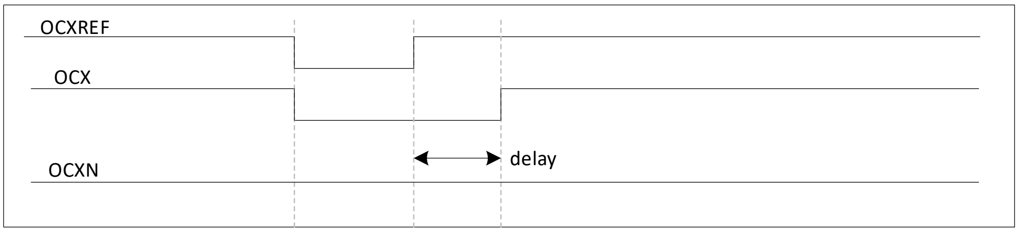


图 19-36 死区波形延迟大于负脉冲

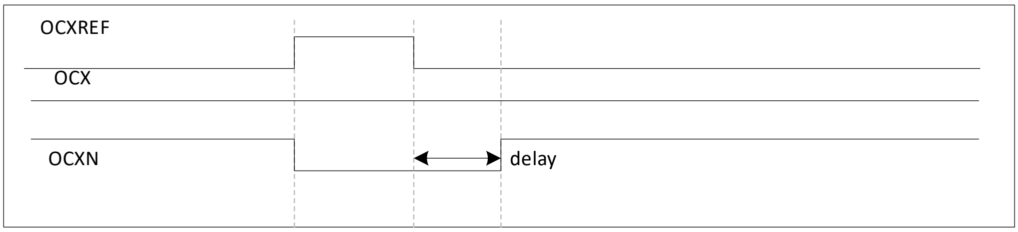


图 19-37 死区波形延迟大于正脉冲

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

重定向 OCxREF 到 OCx 或 OCxN

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

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

19.3.12. 使用刹车功能

当使用刹车功能时，依据额外的控制位，输出使能信号和无效电平信号都会被修改。无论什么情况下，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源既可以是刹车输入引脚，或者以下内部源：

- CPU LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟 failure 事件
- 来自比较器的输出

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，才能正确地读回写入的位。

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

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CcxE 与 CcxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’1’时，则产生一个中断。

■ 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置'1'；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BG 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。

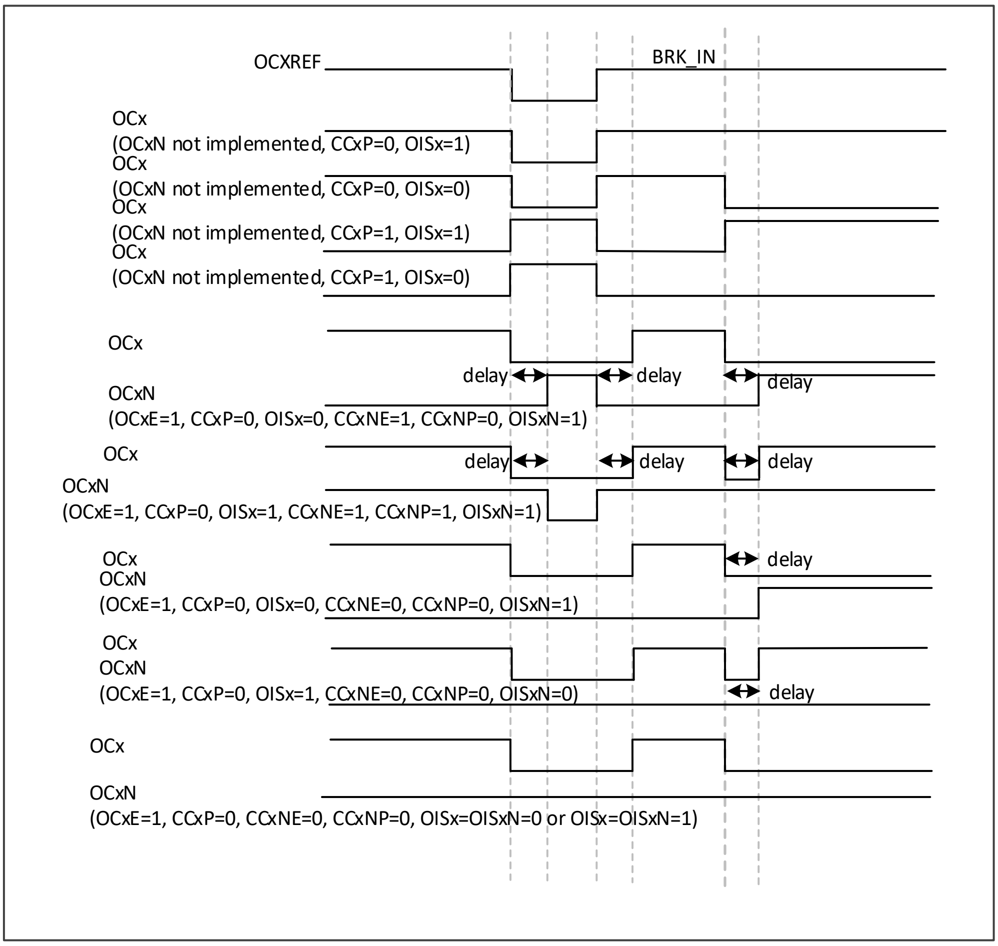


图 19-38 响应刹车的输出

19.3.13. 在外部事件时清除 OCxREF 信号

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

而 OCREF_CLR_INPUT 可以通过配置 TIMx_SMCR 寄存器中的 OCCS 位，在 OCREF_CLR 和 ETRF(ETR 滤波后)之间选择。

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

a) 外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。
b) 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。
c) 外部触发极性(ETP)和外部触发滤波器(ETF)可以根据需要配置。

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

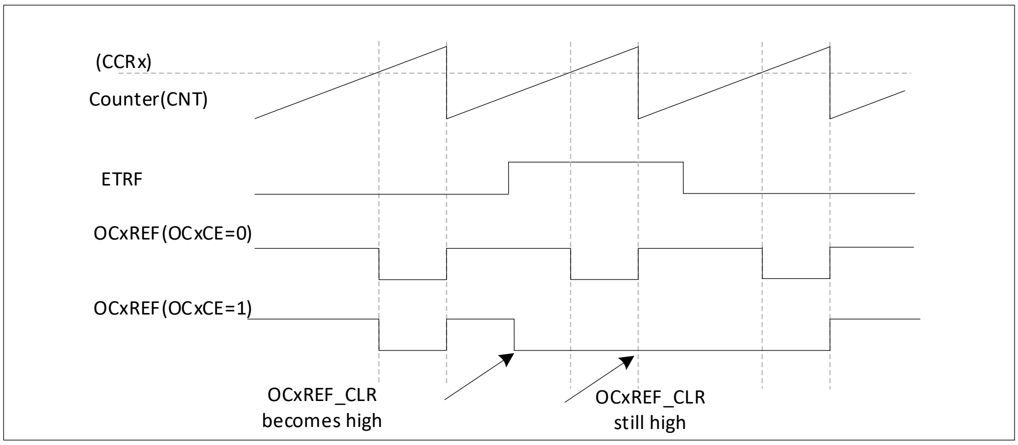


图 19-39 清除 TIM1 的 OCxREF

19.3.14. 六步 PWM 的产生

当在一个通道上需要互补输出时，预装载位有 OCxM、CcxE 和 CcxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样就可以预先设置好下一步骤配置，并在同一个时刻同时修改所有通道的配置。COM 可以通过设置 TIMx_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位(TIMx_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

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

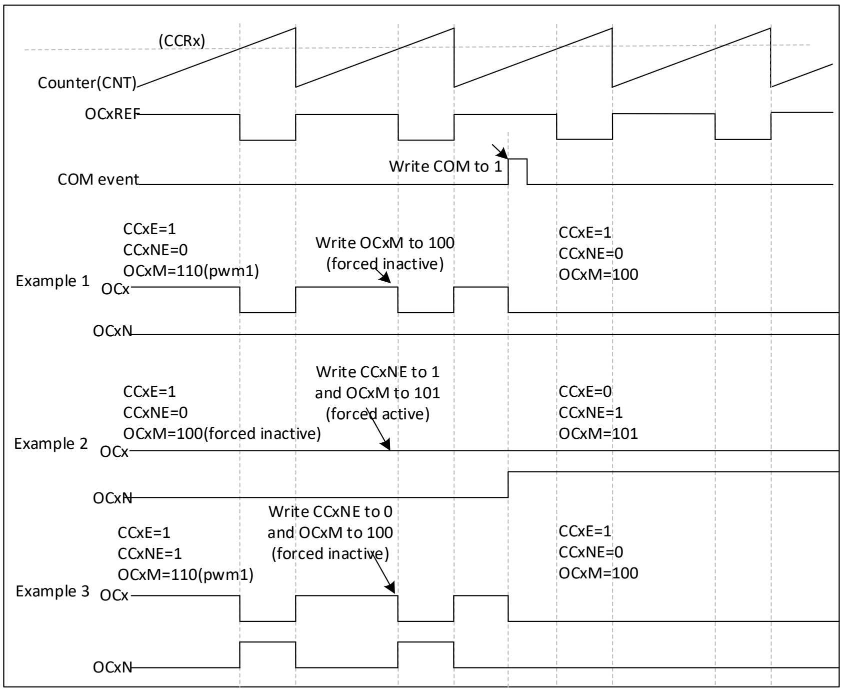


图 19-40 六步产生，COM 的例子(OSSR=1)

19.3.15. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后，产生一个脉宽可被程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器的 OPM 位将选择单脉冲模式，这样可以让计数器自动的在产生下一次计数器溢出时停止。

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

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

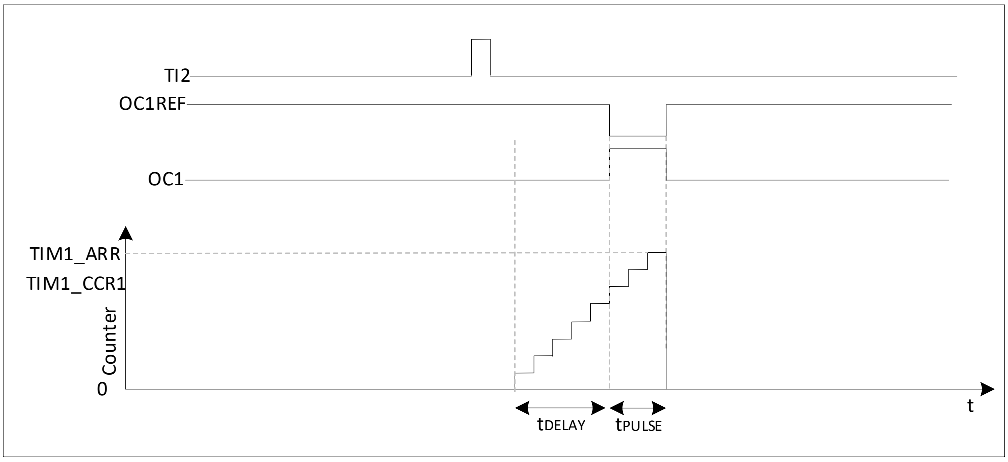


图 19-41 单脉冲模式的例子

例如，当需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

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

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

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 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果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(和 OCx)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 PWM1 和 PWM2 模式时起作用。

19.3.16. 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，则置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

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

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、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。 在这个模式下，计数器依照增量编码器的速度和方向被自

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

表 19-1 计数方向与编码器信号的关系

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No count	No count
	Low	Up	Down	No count	No count
Counting on TI2 only	High	No count	No count	Up	Down
	Low	No count	No count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。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，TI1FP2 映射到 TI2)
- CC1P='0'(TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1)
- CC2P='0'(TIMx_CCER 寄存器，TI1FP2 不反相，TI1FP2=TI2)
- SMS='011'(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效)
- CEN='1'(TIMx_CR1 寄存器，计数器使能)

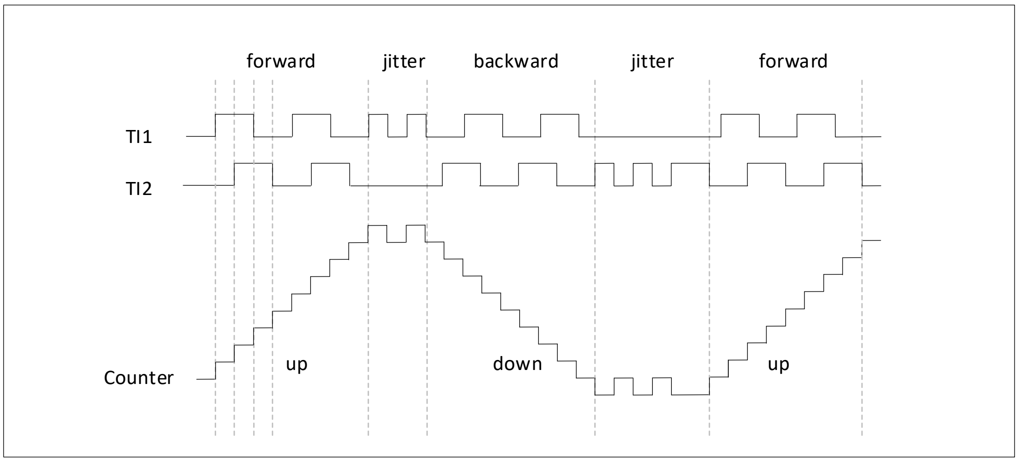


图 19-42 编码器模式下的计数器操作实例

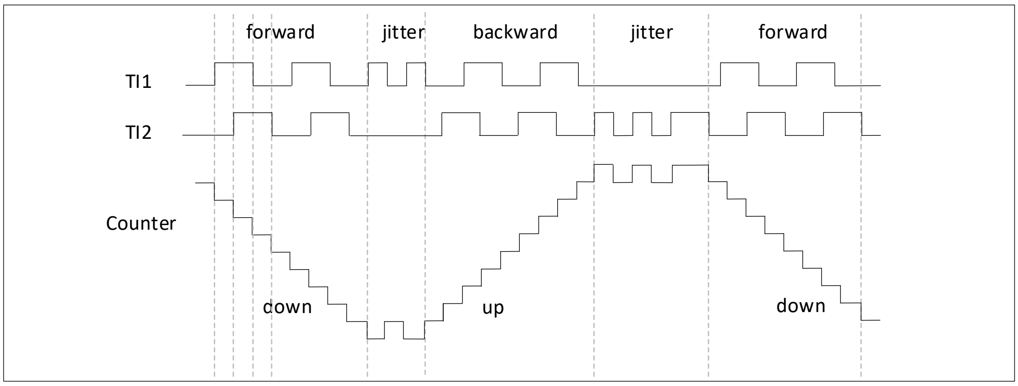


图 19-43 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前的位置信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度、加速度、减速度）。指示机械零点的编码器输出可被用作此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的，并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

19.3.17. 定时器输入异或功能

TIM_CR2 寄存器的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

19.3.18. 与霍尔传感器的接口

使用高级定时器（TIM1）产生 PWM 信号驱动马达时，可以使用另一个通用 timer（TIM3）作为“接口定时器”来连接霍尔传感器。3 个定时器输入脚（CC1、CC2、CC3）通过一个异或门连接到 TI1 输入通道（通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择），“接口定时器”捕获这个信号。

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

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

接口定时器可以用来在输出模式产生一个脉冲，这个脉冲可以（通过触发一个 COM 事件）用于改变高级定时器 TIM1 各个通道的属性，而高级定时器产生 PWM 信号驱动马达。因此接口定时器通道必须编程为一个指定的延迟（输出比较或 PWM 模式）之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级定时器 TIM1。

举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。

- 置 TIMx_CR2 寄存器的 TI1S 位为’1’，配置三个定时器输入逻辑或到 TI1 输入。
- 时基编程：置 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
- 设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。

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

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

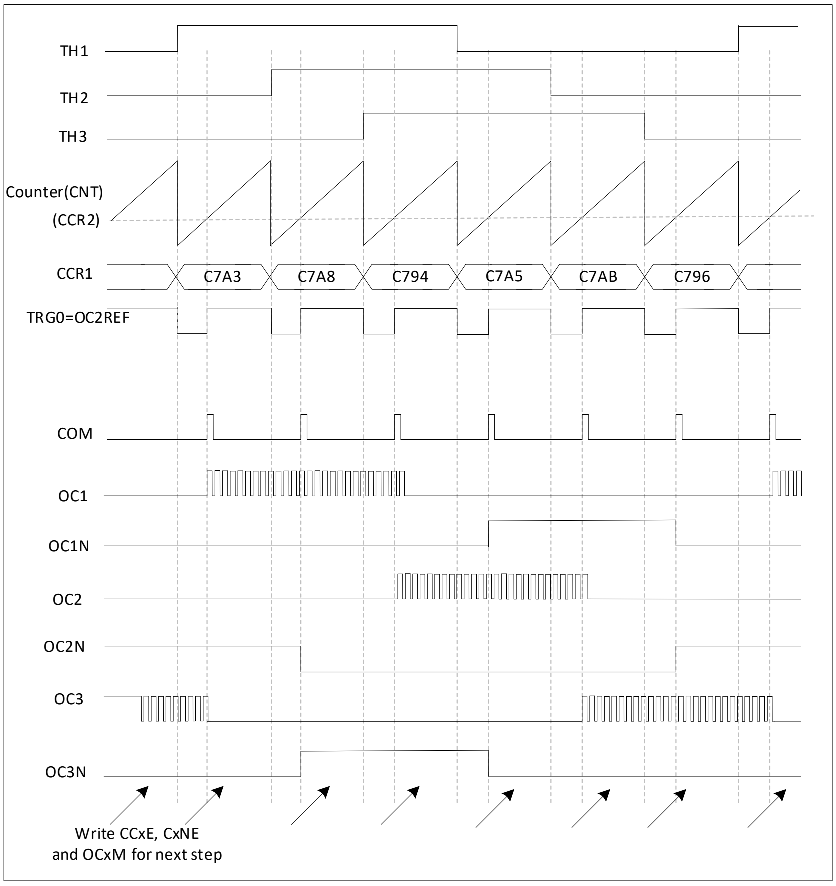


图 19-44 霍尔传感器接口的实例

19.3.19. TIM 和外部的触发同步

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

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

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

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

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

■ 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

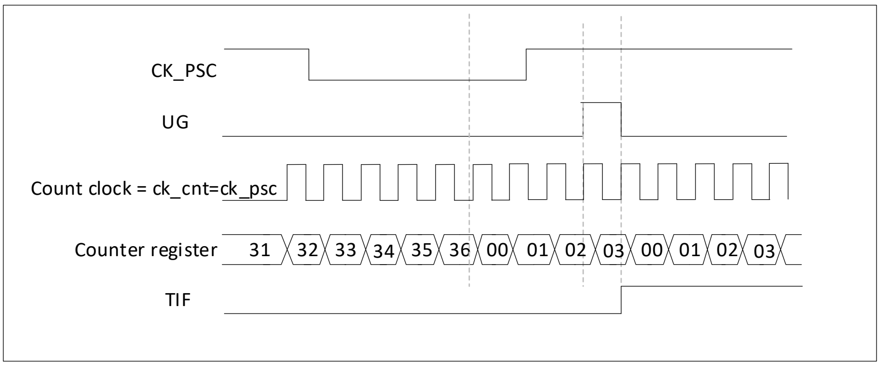


图 19-45 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

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

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

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

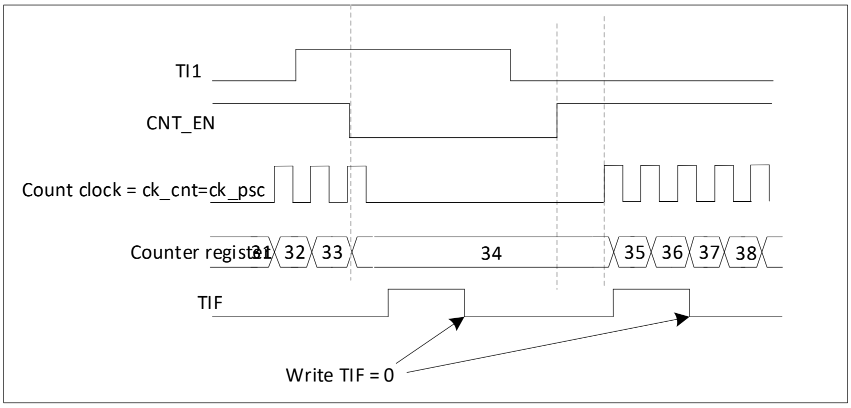


图 19-46 门控模式下的控制电路

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

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

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。 TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

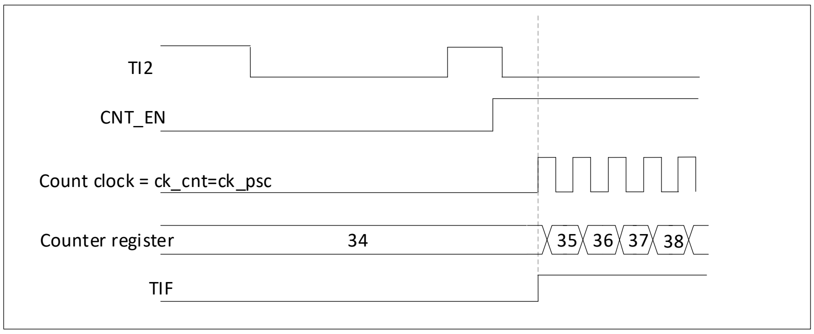


图 19-47 门控模式下的控制电路

从模式：外部时钟模式 2 ＋ 触发模式

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

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

- 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。
- 按如下配置通道 1，检测 TI 的上升沿：
  - IC1F=0000：没有滤波
  - 触发操作中不使用捕获预分频器，不需要配置
  - 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
  - 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。 ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

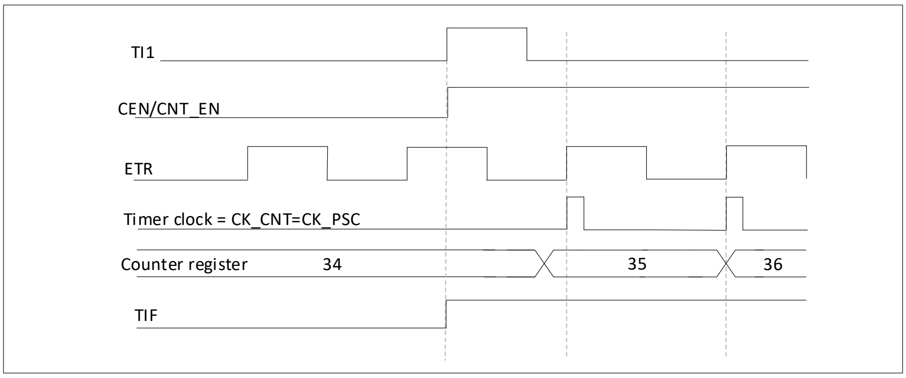


图 19-48 外部时钟模式 2 + 触发模式下的控制电路

19.3.20. 定时器同步

TIM 定时器在内部相连，用于 timer 的同步或者链接功能。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或时钟等操作。

19.3.21. 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止工作。

19.4. TIM1 寄存器描述

19.4.1. TIM1 控制寄存器 1 (TIM1_CR1)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved			
9:8	CKD[1:0]	RW	00	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： tDTS = tCK_INT 01： tDTS = 2 x tCK_INT 10： tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： TIM1_ARR 寄存器没有缓冲 1： TIM1_ARR 寄存器被装入缓冲器

Bit	Name	R/W	Reset Value	Function
6:5	CMS[1:0]	RW	00	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(TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIM1_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	计数方向 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读
3	OPM	RW	0	单脉冲模式 0：在发生更新事件时，计数器不停止 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： －计数器溢出/下溢 －设置 UG 位 －从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	禁止更新(Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： －计数器溢出/下溢 －设置 UG 位 －从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	使能计数器 0：禁止计数器 1：开启计数器

Bit	Name	R/W	Reset Value	Function
				注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

19.4.2. TIM1 控制寄存器 2 (TIM1_CR2)

Address offset:0x04

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res	CCPC
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	-	0	保留，始终为 0
14	OIS4	RW	0	输出空闲状态 4(OC4 输出)。参见 OIS1 位。
13	OIS3N	RW	0	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位
12	OIS3	RW	0	输出空闲状态 3(OC3 输出)。参见 OIS1 位。
11	OIS2N	RW	0	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。参见 OIS1 位
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出)。 0：当 MOE=0 时，死区后 OC1N=0 1：当 MOE=0 时，死区后 OC1N=1 注：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	RW	0	输出空闲状态 1(OC1 输出)。 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1 注：已经设置了 LOCK(TIM1_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7	TI1S	RW	0	TI1 选择 0： TIM1_CH1 管脚连到 TI1 输入。 1： TIM1_CH1、TIM1_CH2 和 TIM1_CH3 管脚经异或后连到 TI1 输入。
6:4	MMS[2:0]	RW	000	主模式选择 这两位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位－TIM1_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：允许－计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制从定时器的一个窗口。计数器使能信号是通过 CEN 控制位和

Bit	Name	R/W	Reset Value	Function
				门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIM1_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 一旦发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即是它已经为高)，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)。 注意： 1.从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。 2.若主从定时器不在同一总线上，主模式应该配置为能被从定时器采到的宽度。
3	CCDS	RW	0	捕获/比较的 DMA 选择 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们。 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	Res	-	0	保留，始终读为 0。
0	CCPC	RW	0	捕获/比较预装载控制位 0： CcxE， CcxNE 和 OCxM 位不是预装载的。 1： CcxE， CcxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新。 注：该位只对具有互补输出的通道起作用。

19.4.3. TIM1 从模式控制寄存器 (TIM1_SMCR)

Address offset:0x08

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

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

Bit	Name	R/W	Reset Value	Function
15	ETP	RW	0	外部触发极性。该位选择是否 ETR 或者 ETR 的反向被用作触发操作。 0：ETR 不进行反向，高电平或者上升沿有效 1：ETR 反向，低电平或者下降沿有效
14	ECE	RW	0	外部时钟使能位。该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是‘111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13：12	ETPS[1:0]	RW	00	外部触发预分频器。外部触发信号 ETRP 频率必须至多是 TIM1CLK 频率的 1/4。一个预分频器可以被使能，以降低 ETRP 的频率。当输入快速外部时钟是有效的。 00：预分频器关闭 01：ETRP 频率的 2 分频 10：ETRP 频率的 4 分频 11：ETRP 频率的 8 分频
11：8	ETF[3:0]	RW	0000	外部触发滤波。这些位定义采样 ETRP 信号的频率和应用在 ETRP 的数字滤波长度。这个数字滤波由一个事件计数器组成，在改计数器里，N 个连续的事件被需要使输出的边沿有效。 0000：没有滤波器，在 fDTS 下采样 0001：fSAMPLING=fCK_INT, N=2 0010：fSAMPLING=fCK_INT, N=4 0011：fSAMPLING=fCK_INT, N=8 0100：fSAMPLING=fCK_INT/2, N=6 0101：fSAMPLING=fCK_INT/2, N=8 0110：fSAMPLING=fCK_INT/4, N=6 0111：fSAMPLING=fCK_INT/4, N=8 1000：fSAMPLING=fCK_INT/8, N=6 1001：fSAMPLING=fCK_INT/8, N=8 1010：fSAMPLING=fCK_INT/16, N=5 1011：fSAMPLING=fCK_INT/16, N=6 1100：fSAMPLING=fCK_INT/16, N=8 1101：fSAMPLING=fCK_INT/32, N=5 1110：fSAMPLING=fCK_INT/32, N=6 1111：fSAMPLING=fCK_INT/32, N=8
7	MSM	RW	0	主/从模式 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的当前定时器和从定时器间的同步（通过

Bit	Name	R/W	Reset Value	Function
				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]	RW	000	触发选择，这 3 位选择用于同步计数器的触发输入。 000：Reserved(ITR0) 001：Reserved(ITR1) 010：TIM3(ITR2) 011：TIM17(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：为避免在信号转变时产生错误的边沿检测，必须在未使用这些位（比如 SMS=000）时修改它们
3	OCCS	RW	0	OCREF 清除选择位。该位用于选择 OCREF 的清除源。 0：OCREF_CLR_INT 连接到 OCREF_CLR 输入 1：OCREF_CLR_INT 连接到 ETRF
2：0	SMS[2:0]	RW	000	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：关闭从模式 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 根据 TI1FP2 的电平，计数器在 TI2FP1 的边沿向上/下计数。 010：编码器模式 2 根据 TI2FP1 的电平，计数器在 TI1FP2 的边沿向上/下计数。 011：编码器模式 3 模式 1 和模式 2 的综合 100：复位模式 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。 注：在编码器模式下，不要使用 UEV 作为 TRGO 输出信号，（即 MMS 不能配置为 010)

TIM1 内部触发连接

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	TIM15_TRGO	TIM2_TRGO	TIM3_TRGO	TIM17_TRGO

19.4.4. TIM1 DMA/中断使能寄存器 (TIM1_DIER)

Address offset:0x0C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	**TDE**	**COMDE**	**CC4DE**	**CC3DE**	**CC2DE**	**CC1DE**	**UDE**	**BIE**	**TIE**	**COMIE**	**CC4IE**	**CC3IE**	**CC2IE**	**CC1IE**	**UIE**
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved			保留，一直为 0
14	TDE	RW	0	TDE：允许触发 DMA 请求 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	COMDE：允许 COM 的 DMA 请求 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
12	CC4DE	RW	0	CC4DE：允许捕获/比较 4 的 DMA 请求 0：禁止捕获/比较 4 的 DMA 请求 1：允许捕获/比较 4 的 DMA 请求
11	CC3DE	RW	0	CC3DE：允许捕获/比较 3 的 DMA 请求 0：禁止捕获/比较 3 的 DMA 请求 1：允许捕获/比较 3 的 DMA 请求
10	CC2DE	RW	0	CC2DE：允许捕获/比较 2 的 DMA 请求 0：禁止捕获/比较 2 的 DMA 请求 1：允许捕获/比较 2 的 DMA 请求
9	CC1DE	RW	0	CC1DE：允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	BIE：允许刹车中断 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	TIE：允许触发中断 0：禁止触发中断 1：允许触发中断
5	COMIE	RW	0	COMIE：允许 COM 中断 0：禁止 COM 中断 1：允许 COM 中断

Bit	Name	R/W	Reset Value	Function
4	CC4IE	RW	0	CC4IE：允许捕获/比较 4 中断 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	RW	0	CC3IE：允许捕获/比较 3 中断 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	CC2IE：允许捕获/比较 2 中断 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	CC1IE：允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断 1：允许更新中断

19.4.5. TIM1 状态寄存器(TIM1_SR)

Address offset:0x010

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	ic4if	ic3if	ic2if	ic1if	ic4ir	ic3ir	ic2ir	ic1ir
-	-	-	-	-	-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
-	-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31：13	Reserved	-	0	保留，一直为 0
23	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
22	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
21	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
20	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生下降沿捕获事件。
19	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
18	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。

Bit	Name	R/W	Reset Value	Function
17	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
16	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清 ‘0’ 或通过读 TIMx_CCR1 清 ‘0’ 。 0：无重复捕获产生； 1：发生上升沿捕获事件。
15:13	Reserved			保留，始终为 0
12	CC4OF	RC_W0	0	捕获/比较 4 过捕获标记 参见 CC1OF 描述
11	CC3OF	RC_W0	0	捕获/比较 3 过捕获标记 参见 CC1OF 描述
10	CC2OF	RC_W0	0	捕获/比较 2 过捕获标记 参见 CC1OF 描述
9	CC1OF	RC_W0	0	捕获/比较 1 过捕获标记 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：无过捕获产生； 1： CC1OF 置 1 时，计数器的值已经被捕获到 TIM1_CCR1 寄存器。
8	Res	RC_W0	0	保留，始终读为 0。
7	BIF	RC_W0	0	刹车中断标记 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	RC_W0	0	触发器中断标记 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：触发器中断等待响应
5	COMIF	RC_W0	0	COM 中断标记 一旦产生 COM 事件（当 CcxE、 CcxNE、 OCxM 已被更新）该位由硬件置 1。它由软件清 0。 0：无 COM 事件产生； 1： COM 中断等待响应
4	CC4IF	RC_W0	0	捕获/比较 4 中断标记 参考 CC1IF 描述
3	CC3IF	RC_W0	0	捕获/比较 3 中断标记 参考 CC1IF 描述
2	CC2IF	RC_W0	0	捕获/比较 2 中断标记 参考 CC1IF 描述

Bit	Name	R/W	Reset Value	Function
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 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： TIM1_CNT 的值与 TIM1_CCR1 的值匹配。 如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置 1，它由软件清 0 或通过读 TIM1_CCR1 清 0。 0：无输入捕获产生； 1：输入捕获产生并且计数器值已装入 TIM1_CCR1(在 IC1 上检测到与所选极性相同的边沿)。 注：当 CEN 打开，该位也会被置位。
0	UIF	RC_W0	0	更新中断标记 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 TIM1_CR1 寄存器的 UDIS=0，当 REP_CNT=0 时产生更新事件(重复向下计数器上溢或下溢时)； – 若 TIM1_CR1 寄存器的 UDIS=0、 URS=0，当 TIM1_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)； – 若 TIM1_CR1 寄存器的 UDIS=0、 URS=0，当 CNT 被触发事件重初始化时产生更新事件。 (参考从模式控制寄存器(TIM1_SMCR))

19.4.6. TIM1 事件产生寄存器(TIM1_EGR)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	--	-	-	-	-	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	0	保留，一直为 0
7	BG	W	0	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清 0。 0：无动作；

Bit	Name	R/W	Reset Value	Function
				1：产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断，则产生相应的中断。
6	TG	W	0	产生触发事件 该位由软件置 1，用于产生一个触发事件，由硬件自动清 0。 0：无动作； 1： TIM1_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	W	0	捕获/比较事件，产生控制更新 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：当 CCPC=1，允许更新 CcxE、 CcxNE、 OCxM 位。 注： 该位只对有互补输出的通道有效。
4	CC4G	W	0	产生捕获/比较 4 事件 参考 CC1G 描述
3	CC3G	W	0	产生捕获/比较 3 事件 参考 CC1G 描述
2	CC2G	W	0	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	W	0	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 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： 当前的计数器值捕获至 TIM1_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	W	0	产生更新事件。该位由软件置 1，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)则计数器装载 TIM1_ARR 的值。

19.4.7. TIM1 捕获/比较模式寄存器 1(TIM1_CCMR1)

Address offset:0x18

Reset value:0x0000 0000

Output compare mode:

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

-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-		保留，一直为 0
15	OC2CE	RW	0	输出比较 2 清 0 使能
14:12	OC2M[2:0]	RW	000	输出比较 2 模式选择
11	OC2PE	RW	0	输出比较 2 预装载使能
10	OC2FE	RW	0	输出比较 2 快速使能
9:8	CC2S[1:0]	RW	00	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 TIM1_SMCR 寄存器的 TS 位选择）。 注： CC2S 仅在通道关闭时(TIM1_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0	输出比较 1 清 0 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=0。
6:4	OC1M[2:0]	RW	00	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 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010： 匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIM1_CCR1=TIM1_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 TIM1_CNT<TIM1_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 TIM1_CNT>TIM1_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 TIM1_CNT<TIM1_CCR1 时通道 1 为无效电平，否则为有

Bit	Name	R/W	Reset Value	Function
				效电平；在向下计数时，一旦 TIM1_CNT>TIM1_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIM1_CCR1 寄存器的预装载功能，可随时写入 TIM1_CCR1 寄存器，且新值马上起作用。 1：开启 TIM1_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作， TIM1_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 该位用于加快 CC 输出对触发器输入事件的响应。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而 与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 的只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIM1_SMCR 寄存器的 TS 位选择）。 注： CC1S 仅在通道关闭时(TIM1_CCER 寄存器的 CC1E=0)才是可写的。

Input Capture mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-		保留，一直为 0
15:12	IC2F	RW	0000	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	00	输入/捕获 2 预分频器
9:8	CC2S[1:0]	RW	0	捕获/比较 2 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIM1_SMCR 寄存器的 TS 位选择）。 注： CC2S 仅在通道关闭时(TIM1_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	0000	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， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8， N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC[1:0]	RW	00	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S[1:0]	RW	00	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；

Bit	Name	R/W	Reset Value	Function
				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 TIM1_SMCR 寄存器的 TS 位选择）。 注： CC1S 仅在通道关闭时(TIM1_CCER 寄存器的 CC1E=0)才是可写的。

19.4.8. TIM1 捕获/比较模式寄存器 2(TIM1_CCMR2)

Address offset:0x1C

Reset value:0x0000 0000

Output compare mode:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-		保留，一直为 0
15	OC4CE	RW	0	输出比较 4 清 0 使能
14:12	OC4M[2:0]	RW	000	输出比较 4 模式
11	OC4PE	RW	0	输出比较 4 预装载使能
10	OC4FE	RW	0	输出比较 4 快速使能
9:8	CC4S[1:0]	RW	00	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 TIM1_SMCR 寄存器的 TS 位选择）。 注：CC4S 仅在通道关闭时(TIM1_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0	输出比较 3 清 0 使能
6:4	OC3M[2:0]	RW	00	输出比较 3 模式
3	OC3PE	RW	0	输出比较 3 预装载使能
2	OC3FE	RW	0	输出比较 3 快速使能
1:0	CC3S[1:0]	RW	00	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；

Bit	Name	R/W	Reset Value	Function
				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 TIM1_SMCR 寄存器的 TS 位选择）。 注：CC3S 仅在通道关闭时(TIM1_CCER 寄存器的 CC3E=0)才是可写的。

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-		保留，一直为 0
15:12	IC4F[3:0]	RW	0000	输入捕获 4 滤波器
11:10	IC4PSC[1:0]	RW	00	输入/捕获 4 预分频器
9:8	CC4S[1:0]	RW	00	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 TIM1_SMCR 寄存器的 TS 位选择）。 注： CC4S 仅在通道关闭时(TIM1_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F[3:0]	RW	0000	输入捕获 3 滤波器
3:2	IC3PSC[1:0]	RW	00	输入/捕获 3 预分频器
1:0	CC3S[1:0]	RW	00	捕获/比较 3 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIM1_SMCR 寄存器的 TS 位选择）。 注：CC3S 仅在通道关闭时(TIM1_CCER 寄存器的 CC3E=0)才是可写的。

19.4.9. TIM1 捕获/比较使能寄存器 (TIM1_CCER)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：14	Reserved	-	0	保留，一直为 0
13	CC4P	RW	0	输入/捕获 4 输出极性。参考 CC1P 的描述。

Bit	Name	R/W	Reset Value	Function
12	CC4E	RW	0	输入/捕获 4 输出使能。参考 CC1E 的描述。
11	CC3NP	RW	0	输入/捕获 3 互补输出极性。参考 CC1NP 的描述。
10	CC3NE	RW	0	输入/捕获 3 互补输出使能。参考 CC1NE 的描述。
9	CC3P	RW	0	输入/捕获 3 输出极性。参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能。参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	CC2NE	RW	0	输入/捕获 2 互补输出使能。参考 CC1NE 的描述。
5	CC2P	RW	0	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LCCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE，OSSI， OSSR，OIS1，OIS1N，CC1E 位的值。
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。 注： 1.对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1P 的实际有效位只有在 com 事件发生时才会加载预载值。

Bit	Name	R/W	Reset Value	Function
				2.一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LCCK 位)设为 3 或 2，则该位不能被修改
0	CC1E	RW	0	输入/捕获 1 输出使能 CC1 通道配置为输出： 0：关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N、CC1NE 位的值。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、 OSSR、OIS1、OIS1N、CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIM1_CCR1 寄存器。 0：捕获禁止 0：捕获使能 注： 对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

Table19-2 具有中断功能的互补 OCx 和 OCxN 通道的输出控制

<table>
<tr><th colspan="5">控制位</th><th colspan="2">输出状态</th></tr>
<tr><th>MOE</th><th>OSSI</th><th>OSSR</th><th>CcxE</th><th>CcxNE</th><th>OCx output state</th><th>OCxN output state</th></tr>
<tr><td rowspan="8">1</td><td rowspan="8">x</td><td>0</td><td>0</td><td>0</td><td>输出禁止（与定时器断开）
OCx=0,
OCx_EN=0</td><td>输出禁止（与定时器断开）
OCxN=0,
OCxN_EN=0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>输出禁止（与定时器断开）
OCx=0,
OCx_EN=0</td><td>OCxREF+极性,
OCxN=OCREF xor CCxNP,
OCxN_EN=1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>OCxREF+极性,
OCx=OCREF xor CCxP,
OCx_EN=1</td><td>输出禁止（与定时器断开）
OCxN=0,
OCxN_EN=0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>OCxREF+极性+死区，,
OCx_EN=1</td><td>OCxREF+极性+死区,
OCxN_EN=1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>输出禁止（与定时器断开）
OCx=CCxP,
OCx_EN=0</td><td>输出禁止（与定时器断开）
OCxN=CCxNP,
OCxN_EN=0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>关闭状态（输出使能且为无效电平）
OCx=CCxP,
OCx_EN=1</td><td>OCxREF+极性,
OCxN=OCREF xor CCxNP,
OCxN_EN=1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>OCxREF+极性,
OCx=OCREF xor CCxP,
OCx_EN=1</td><td>关闭状态（输出使能且为无效电平）
OCxN=CCxNP,
OCxN_EN=1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>OCxREF+极性+死区，,
OCx_EN=1</td><td>OCxREF+极性+死区,
OCxN_EN=1</td></tr>
<tr><td rowspan="6">0</td><td>0</td><td rowspan="6">x</td><td>0</td><td>0</td><td colspan="2" rowspan="5">输出禁止（与定时器断开）
异步的：OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCx_EN=0;
若时钟存在：经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td colspan="2">关闭状态（输出使能且为无效电平）</td></tr>
</table>

控制位					输出状态	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
	1		1	0	异步的：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCx_EN=1;	
	1		1	1	若时钟存在：经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平	

19.4.10. TIM1 计算器(TIM1_CNT)

Address offset:0x24

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CNT[15:0]	RW	0	计数器的值

19.4.11. TIM1 预分频器 (TIM1_PSC)

Address offset:0x28

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	PSC[15:0]	RW	0	预分频器的值 计数器的时钟频率（CK_CNT）等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器 被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

19.4.12. TIM1 自动重新加载寄存器 (TIM1_ARR)

Address offset:0x2C

Reset value:0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	ARR[15:0]	RW	0xFFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

19.4.13. TIM1 重复计数器寄存器(TIM1_RCR)

Address offset:0x30

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved			保留，一直为 0
7:0	REP[7:0]	RW	0	周期计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIM1_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下， PWM 周期的数目； - 在中心对称模式下， PWM 半周期的数目；

19.4.14. TIM1 捕获/比较寄存器 1(TIM1_CCR1)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15：0	CCR1[15:0]	RW	0	捕获/比较 1 的值 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有

Bit	Name	R/W	Reset Value	Function
				当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

19.4.15. TIM1 捕捉/比较寄存器 2(TIM1_CCR2)

Address offset:0x38

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR2[15:0]	RW	0	捕获/比较通道 1 的值 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载功能，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/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

19.4.16. TIM1 捕获/比较寄存器 3 (TIM1_CCR3)

Address offset:0x3C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR3[15:0]	RW	0	捕获/比较 3 的值 若 CC3 通道配置为输出：

				CCR3 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIM1_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM1_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件（IC3）传输的计数器值。

19.4.17. TIM1 捕捉/比较寄存器 4(TIM1_CCR4)

Address offset:0x40

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR4[15:0]	RW	0	捕获/比较 4 的值 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIM1_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM1_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件（IC4）传输的计数器值。

19.4.18. TIM1 刹车和死区寄存器(TIM1_BDTR)

Address offset:0x44

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	**AOE**	**BKP**	**BKE**	**OSSR**	**OSSI**	**LOCK[1:0]**		**DTG[7:0]**							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	RW	0	保留，一直为 0
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的值，可由软件清 0 或自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位（TIM1_CCER 寄存器的 CcxE、 CcxNE 位），则开启 OC 和 OCN 输出。
14	AOE	RW	0	自动输出使能 0： MOE 只能被软件置 1； 1： MOE 能被软件置 1 或在下一个更新事件自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注： 1、一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 2、任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入（BRK 及 BRK_ACTH）； 1：开启刹车输入（BRK 及 BRK_ACTH）。 注： 1、一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 2、任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIM1_CCER)）。 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：当定时器不工作时，一旦 CcxE=1 或 CcxNE=1，开启 OC/OCN 输出并输出无效电平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIM1_CCER)）。

Bit	Name	R/W	Reset Value	Function
				0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1： 当 定 时器不 工 作时， 一旦 CcxE=1 或 CcxNE=1，OC/OCN 首 先 输出 其 空闲 电 平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	00	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIM1_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、 TIM1_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，TIM1_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，TIM1_CCMRx 寄存器的 OCxM/OCxPE 位）； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIM1_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	0000 0000	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] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg， Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1、 2 或 3，则这些位不能被修改。

19.4.19. TIM1 DMA 控制寄存器(TIM1_DCR)

Address offset:0x48

Reset value:0x0000 0000

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

Res	Res	Res	DBL[4:0]					Res			DBA[4:0]				
-	-	-	RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：13	Reserved			保留，一直为 0
12:8	DBL[4:0]	RW	0 0000	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIM1_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送），即：定义被传送的字节数目： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 10001：18 次传输
7:5	Reserved	RW	0	保留，始终读为 0
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIM1_DMAR 寄存器的地址进行读或写时），DBA 定义为从 TIM1_CR1 寄存器所在地址开始的偏移量： 00000： TIM1_CR1， 00001： TIM1_CR2， 00010： TIM1_SMCR， ......

19.4.20. TIM1 连续模式的 DMA 地址(TIM1_DMAR)

Address offset:0x4C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			
15：0	DMAB[15:0]	RW	0	DMA 连续传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIMx_CR1 地址 + DBA + DMA 指针，其中： “TIMx_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 指针”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

19.4.21. TIM1 寄存器映像

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	TIM1_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x04	TIM1_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS [2:0]			CCDS	CCUS	Res.	CCPC
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0		0
0x08	TIM1_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS [2:0]		
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIM1_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	COMD	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	TIM1_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4IR	IC3IR	IC2IR	IC1IR	IC4IF	IC3IF	IC2IF	IC1IF	Res.	Res.	Res.	CC4O	CC3O	CC2O	CC1O	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	0	0		0	0	0	0	0	0	0	0
0x14	TIM1_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																									0	0	0	0	0	0	0	0
0x18	TIM1_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M [2:0]			OC2PE	CO2FE	CC2S [1:0]		OC1CE	OC1M [2:0]			OC1PE	OC1FE	CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIM1_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC [1:0]		CC2S [1:0]		IC1F[3:0]				IC1PSC [1:0]		CC1S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIM1_CCMR2(output capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M [2:0]			OC4PE	CO4FE	CC4S [1:0]		OC3CE	OC3M [2:0]			OC3PE	OC3FE	CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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
0x1C	TIM1_CCMR2(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC [1:0]		CC4S [1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S [1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIM1_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	Reset value																									0	0	0	0	0	0	0	0
0x24	TIM1_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM1_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM1_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x30	TIM1_RCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	REP[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x34	TIM1_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TIM1_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIM1_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIM1_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TIM1_BDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK [1:0]		DTG[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	TIM1_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.			DBA[4:0]				

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
48	Reset value																				0	0	0	0	0				0	0	0	0	0
0x4C	TIM1_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

20.通用定时器(TIM2/3)

20.1. TIM2/TIM3 简介

TIM3 通用定时器是由可编程分频器驱动的 16 位自动重装载计数器构成。TIM2 通用定时器是由可编程分频器驱动的 32 位自动重装载计数器构成。

适合多种场合，包括测量输出入信号的脉冲长度（输入捕获）或者产生输出波形（输出比较和 PWM）。

使用定时器预分频器和 RCC 时钟控制器预分频器，脉冲长度和波形周期可以在几个微妙到几个毫秒间调整。

每个定时器都是完全独立的，没有互相共享任何资源。他们可以一起同步操作。

20.2. TIM2/3 主要特性

通用 TIM2/3 定时器功能包括：

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

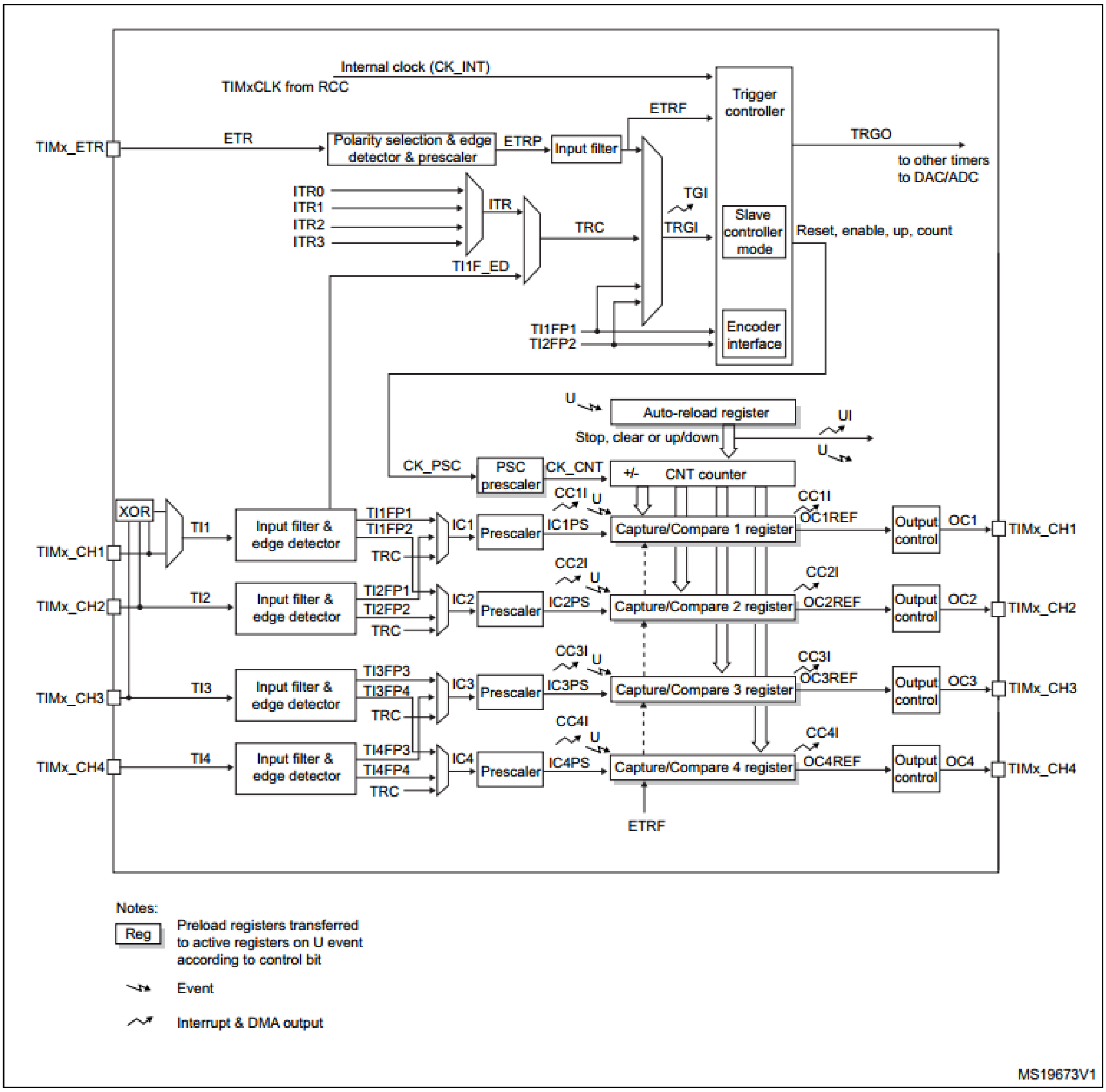


图 20-1 通用定时器架构图

20.3. TIM2/3 功能描述

20.3.1. 时基单元

TIM2/3 定时器的主要部分是一个 16 位(TIM3)或者 32 位（TIM2）计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

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

Time-base unit 包括：

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

自动装载寄存器是预先装载的，写或者读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被一直或在每次的更新事件 UEV 时传送

到影子寄存器。当计数器达到上溢出（向下计数器时的下溢条件）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。

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_CR 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按 1 到 65535 之间的任意值分频。它是基于一个（在 TIMx_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在预分频器运行时，更改计数器参数的例子。

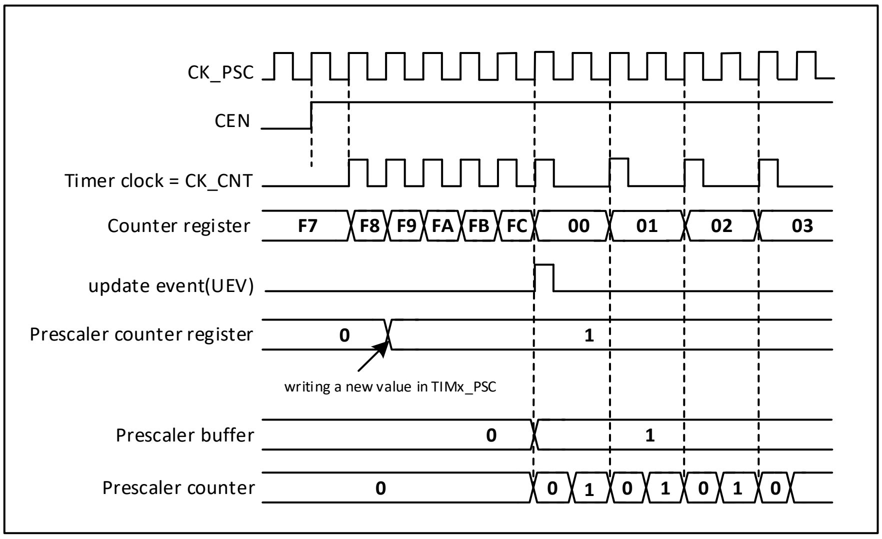


图 20-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

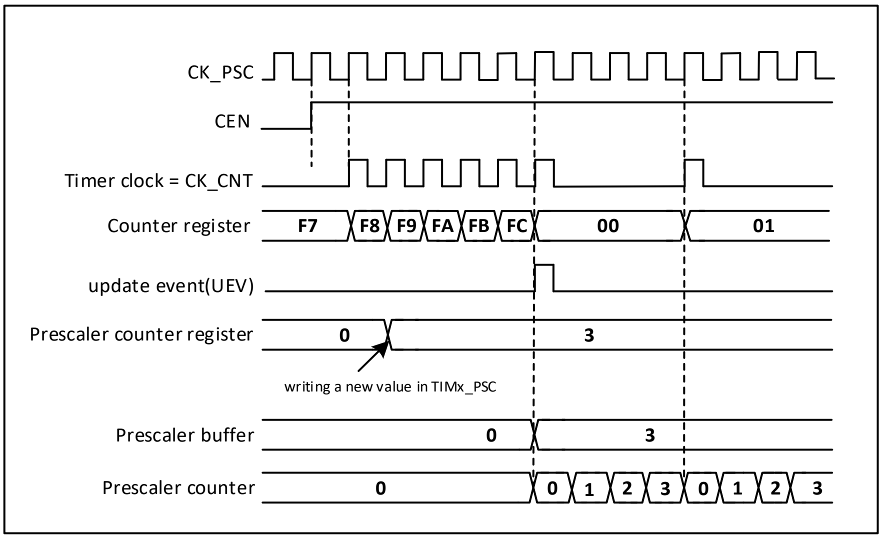


图 20-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

20.3.2. 计数器模式

向上计数模式

向上计数模式，是从 0 计数到自动装载值的计数器，然后又从 0 重新开始计数，并产生一个计数的溢出事件。

每次计数上溢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。

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

当发生一个更新事件时，所有以下的寄存器都被更新，硬件同时(依据 URS 位)设置更新标志位(TIMx_SR 寄存器中的 UIF 位)。

- 自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图给出一些例子，当 TIM3_ARR=0x36 时计数器在不同时钟频率下的动作。

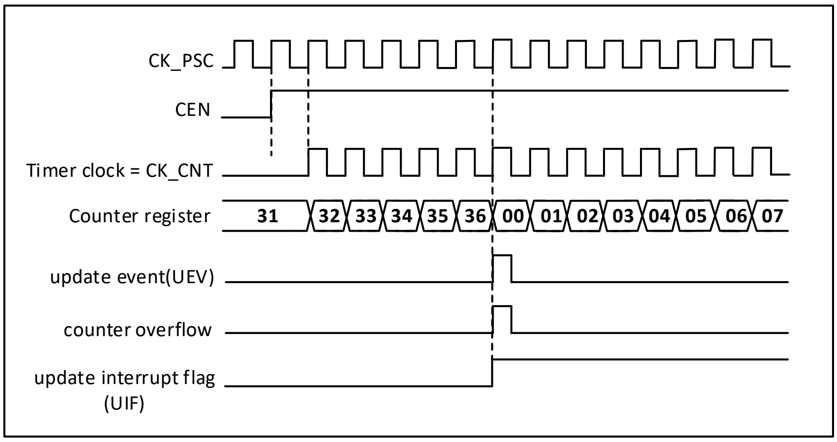


图 20-4 计数器时序图，内部时钟分频因子为 1

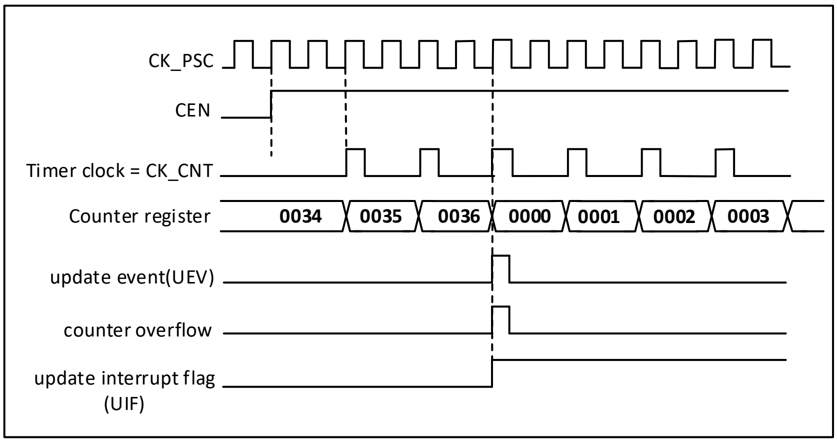


图 20-5 计数器时序图，内部时钟分频因子为 2

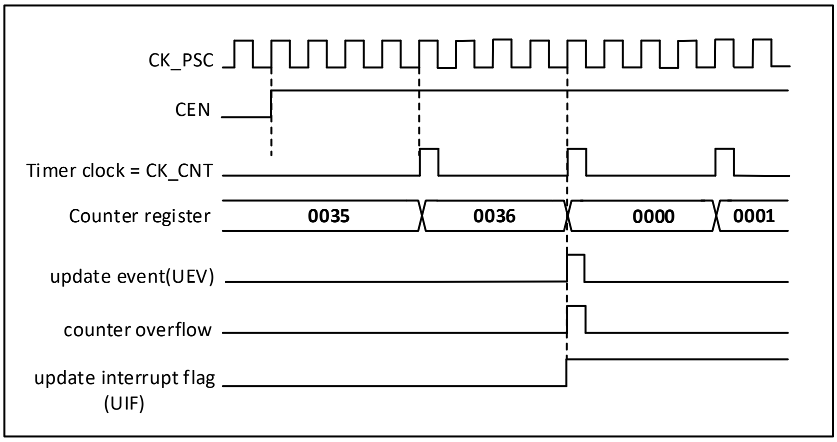


图 20-6 计数器时序图，内部时钟分频因子为 4

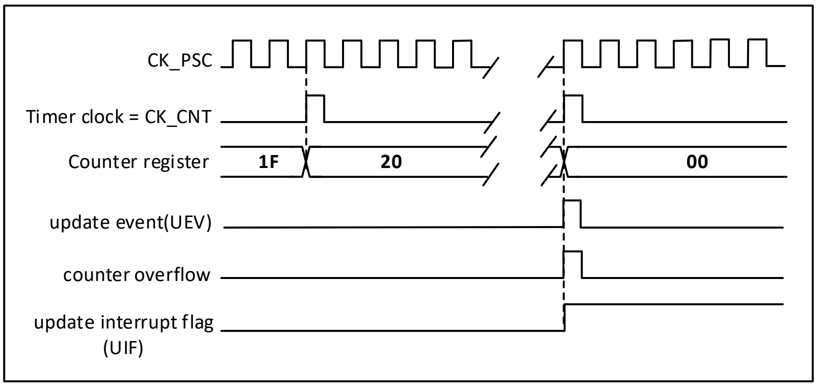


图 20-7 计数器时序图，内部时钟分频因子为 N

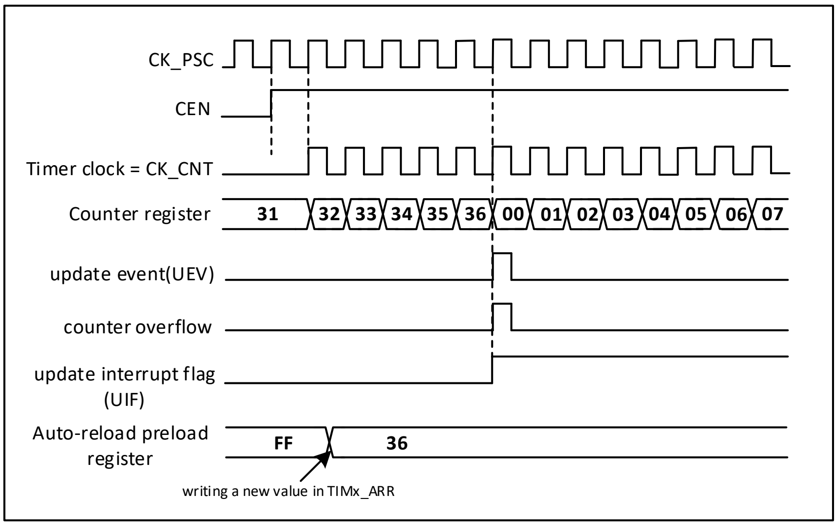


图 20-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

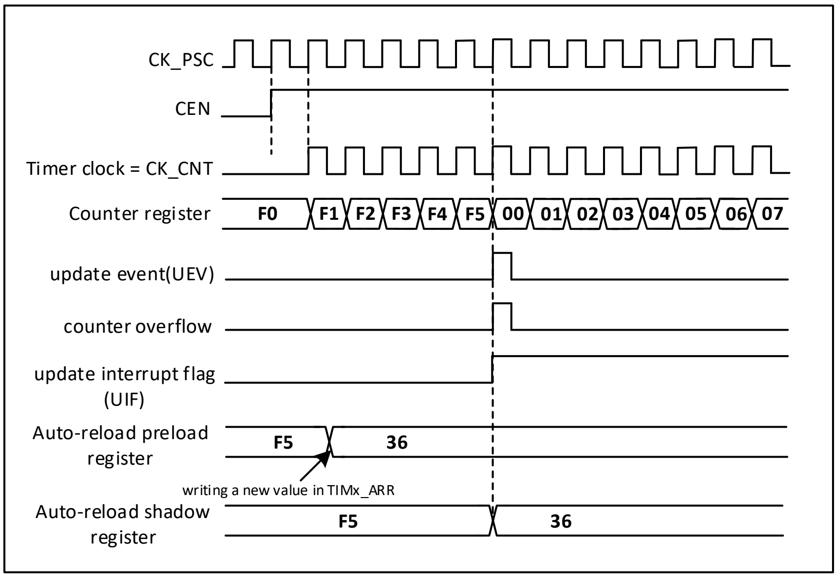


图 20-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

向下计数模式

向下计数模式，从自动装载的值开始向下计数到 0，然后重新开始从自动装载的值向下计数，并产生一个向下溢出事件。

每次计数下溢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。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止更新事件。这样可以避免向预装载寄存器中写入新值时更新寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器被清零(但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，设置 UG 位将产生一个更新事件 UEV，但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有以下的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

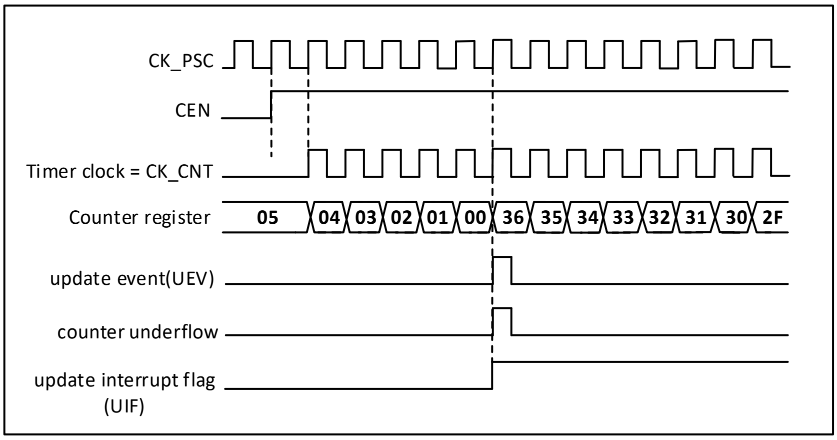


图 20-10 计数器时序图，内部时钟分频因子为 1

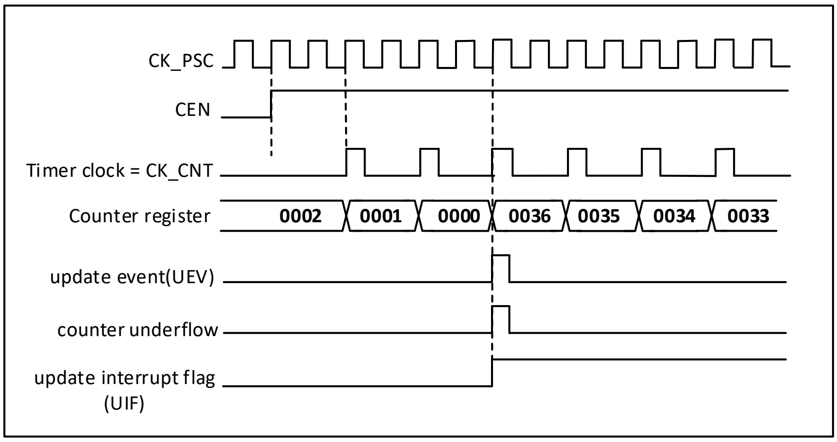


图 20-11 计数器时序图，内部时钟分频因子为 2

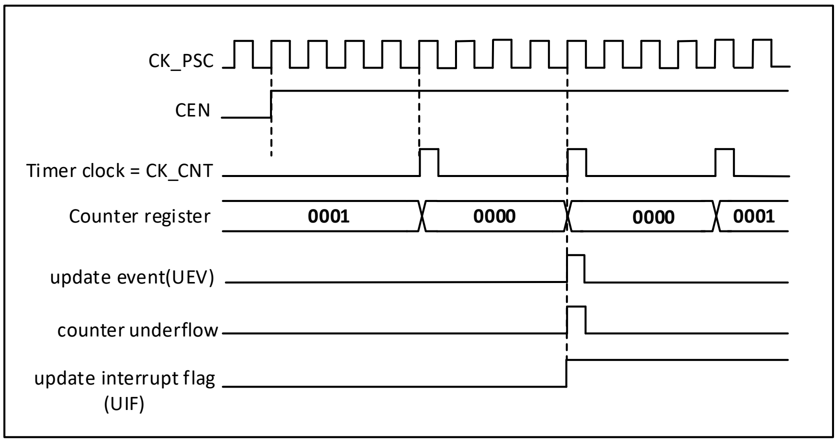


图 20-12 计数器时序图，内部时钟分频因子为 4

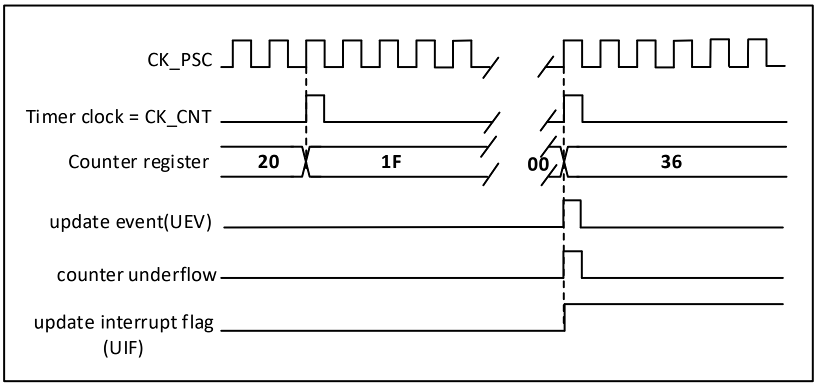


图 20-13 计数器时序图，内部时钟分频因子为 N

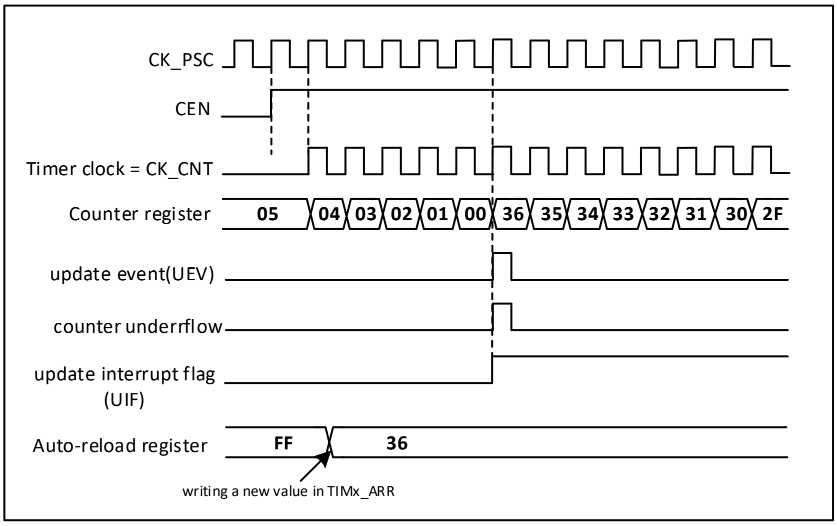


图 20-14 计数器时序图，当没有使用周期计数器时的更新事件

中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器溢出事件，然后向下计数到 1，并产生一个计数器下溢事件，然后再从 0 开始重新计数。

中央对齐模式在 TIMx_CR1 寄存器中的 CMS 不等于 0 时有效。通道在配置成输出模式时，输出比较中断标志将被置位，当：向下计数时（中央对齐模式 1，CMS=”01”）,向上计数时（中央对齐模式 2，CMS=”10”）向上向下计数（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。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIM3_SR 寄存器中的 UIF 位)也被设置。

- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR)

注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)。

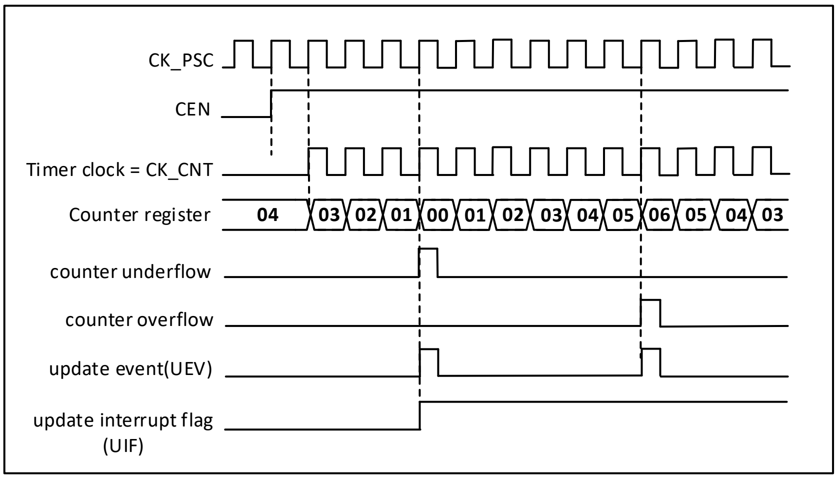


图 20-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR = 0x6

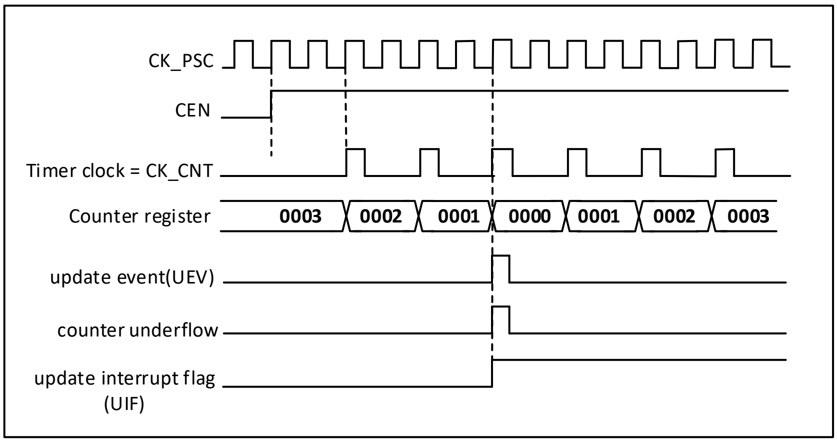


图 20-16 计数器时序图，内部时钟分频因子为 2

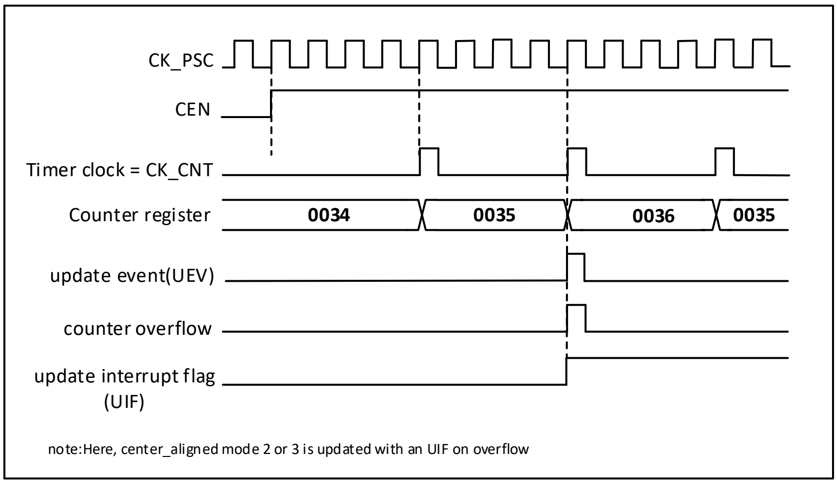


图 20-17 计数器时序图，内部时钟分频因子为 4, TIMx_ARR=0x36

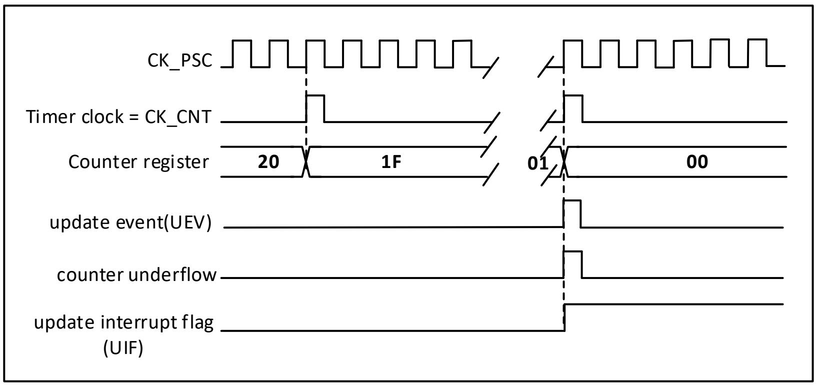


图 20-18 计数器时序图，内部时钟分频因子为 N

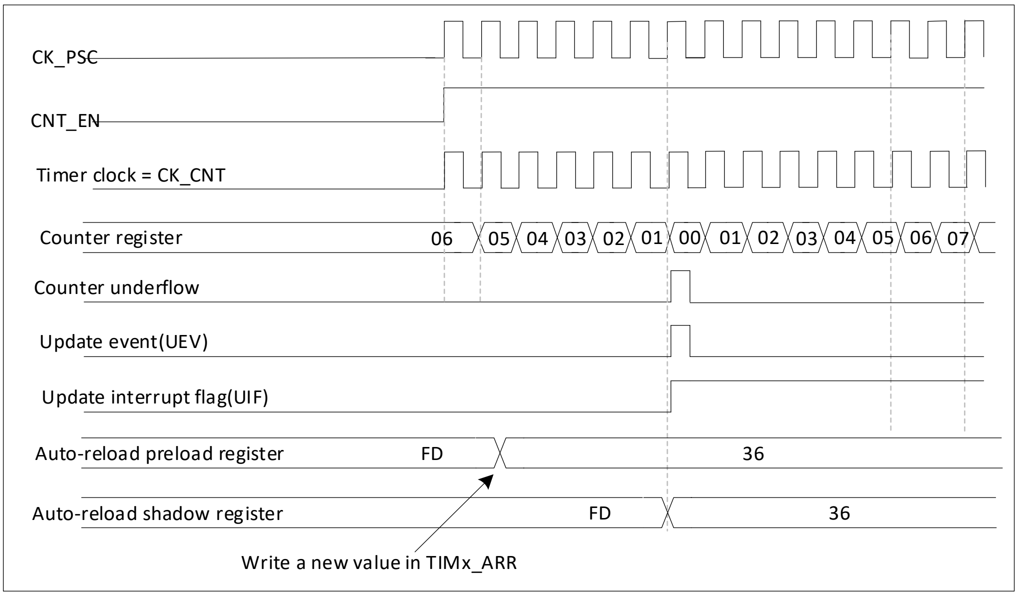


图 20-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

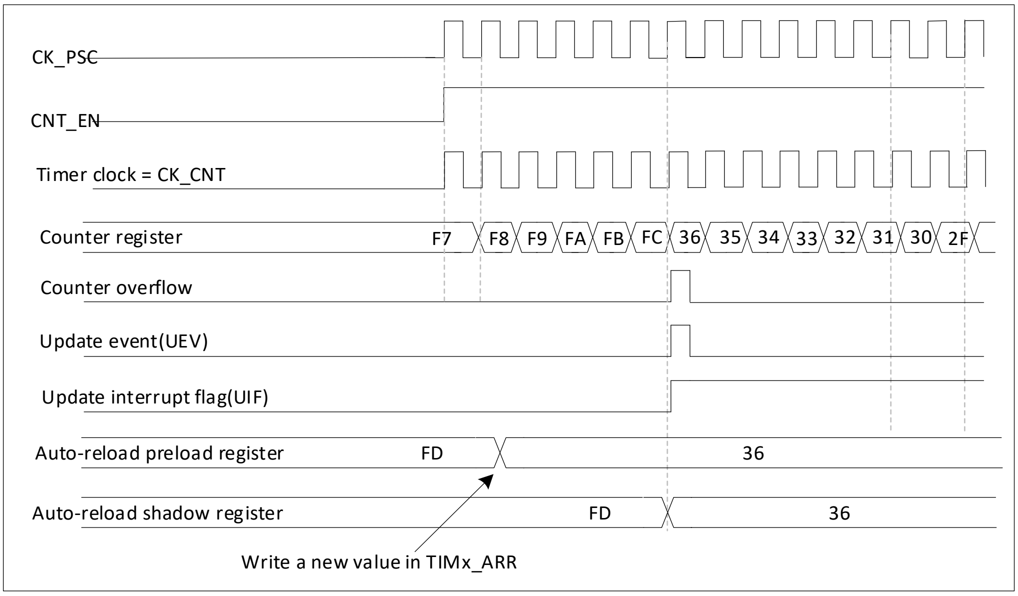


图 20-20 计数器时序图，ARPE=1 时的更新事件(计数器溢出)

20.3.3. 时钟源

计数器的时钟可以由以下时钟源提供:

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置一个定时器 Timer1 作为另一个定时器 Timer3 的预分频器。

内部时钟源（CK_INT）

如果从模式控制器被禁止，则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是事实上的控制位，并且只能被软件修改。只要 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

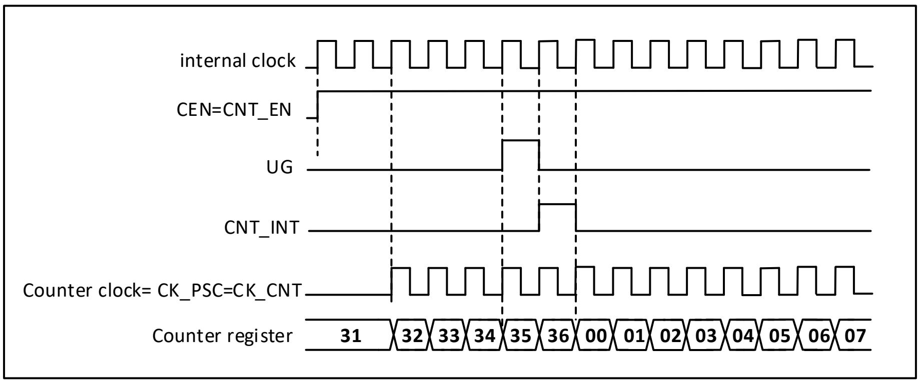


图 20-21 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

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

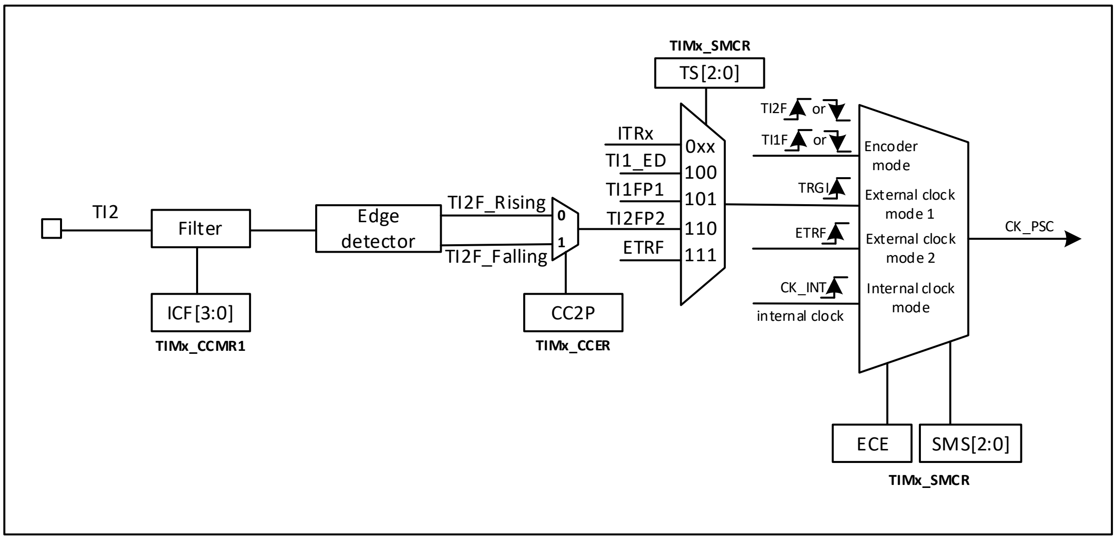


图 20-22 TI2 外部时钟连接示例

例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿

配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)

配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性

配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1

配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源

设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注： 捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

在 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。

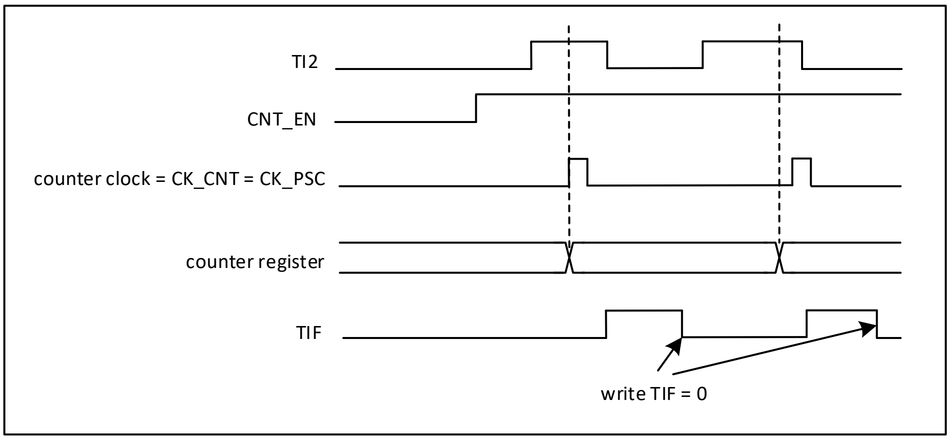


图 20-23 外部时钟模式 1 下的控制电路

20.3.4. 捕获/比较通道

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

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

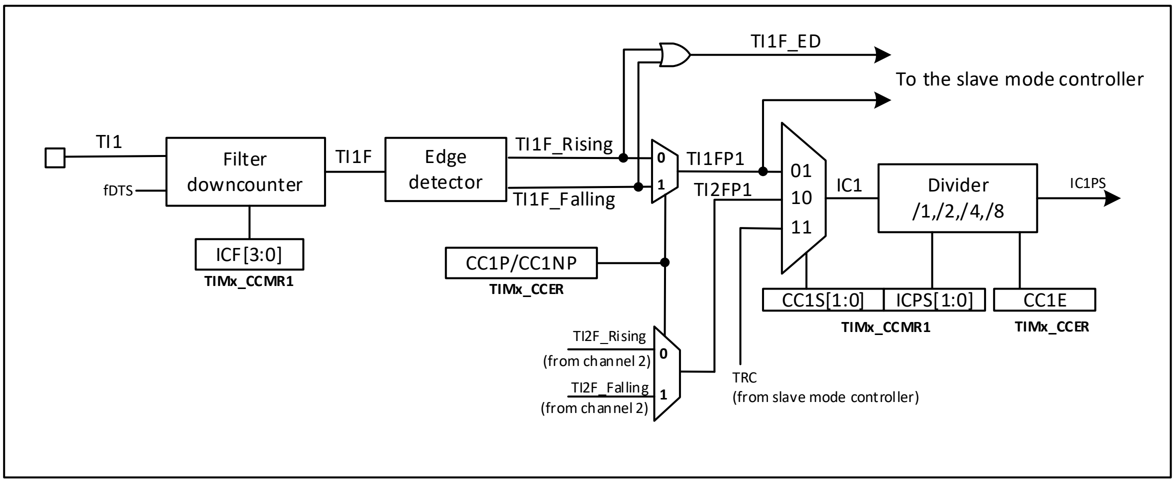


图 20-24 捕获/比较通道(如：通道 1 输入部分)

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。

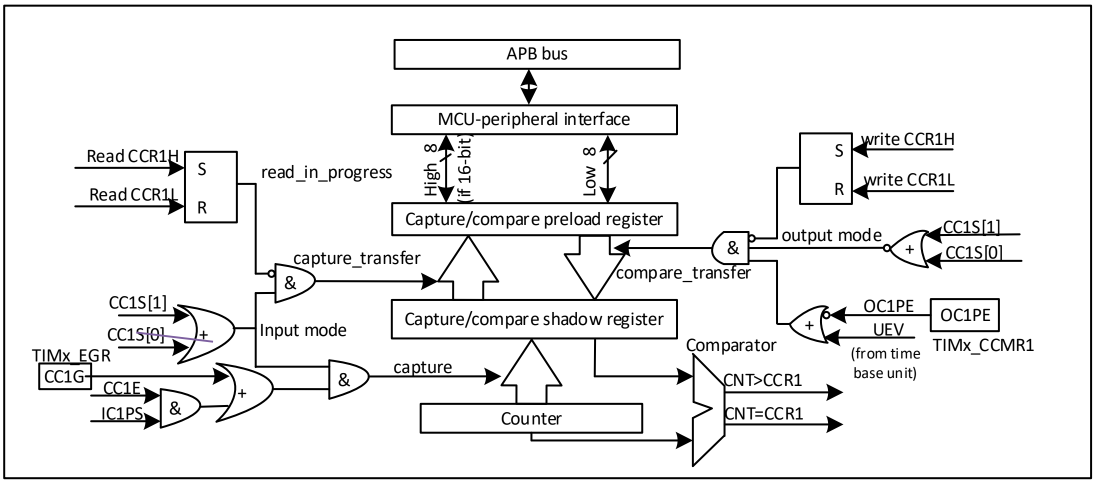


图 20-25 捕获/比较通道 1 的主电路

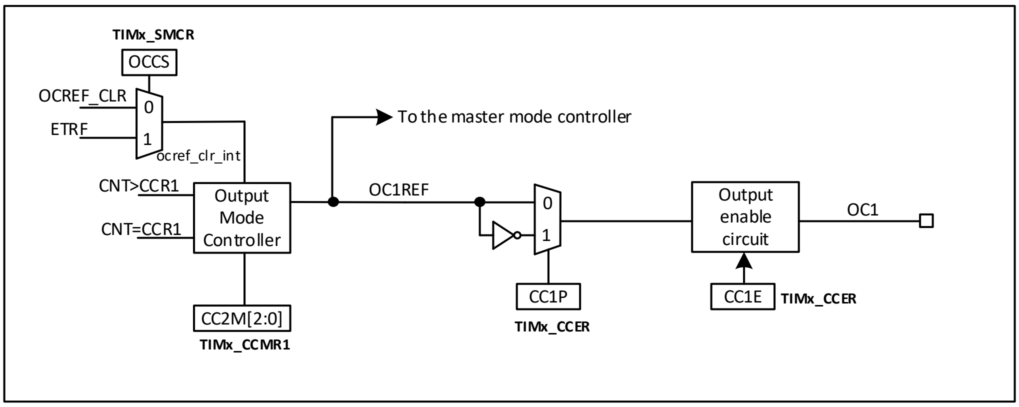


图 20-26 捕获/比较通道的输出部分(通道 1)

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。

20.3.5. 输入捕获模式

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

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

- 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 Tix 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 IcxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)（和 CC1NP=0）
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

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

20.3.6. 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 信号被映射到同一个 Tix 输入。
- 这 2 个 Icx 信号为边沿有效，但是极性相反。
- 其中一个 TixFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

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

- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

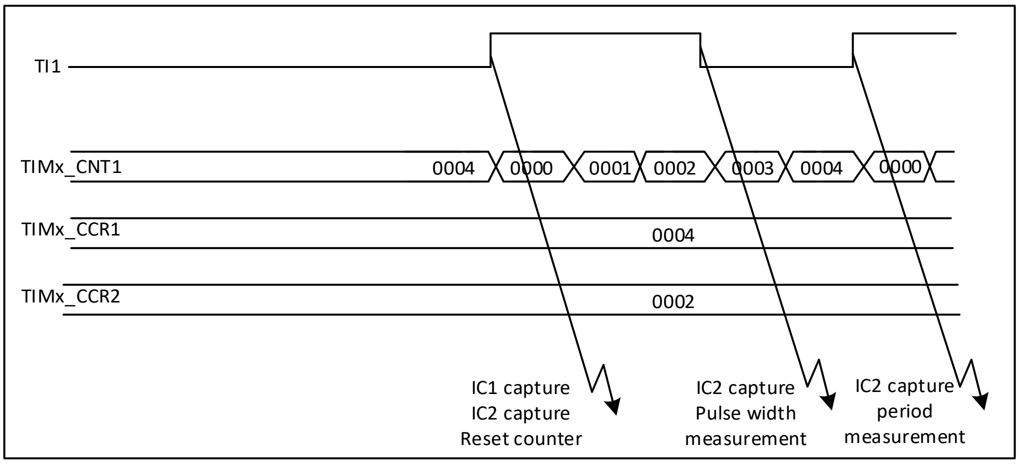


图 20-27 PWM 输入模式时序

20.3.7. 强置输出模式

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

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。 置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

20.3.8. 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)。
- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CcxIF 位)。
- 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CcxIE 位)，则产生一个中断。
- 若设置了相应的使能位(TIMx_DIER 寄存器中的 CcxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

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

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CcxIE 位。
4. 选择输出模式，例如：
   – 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。

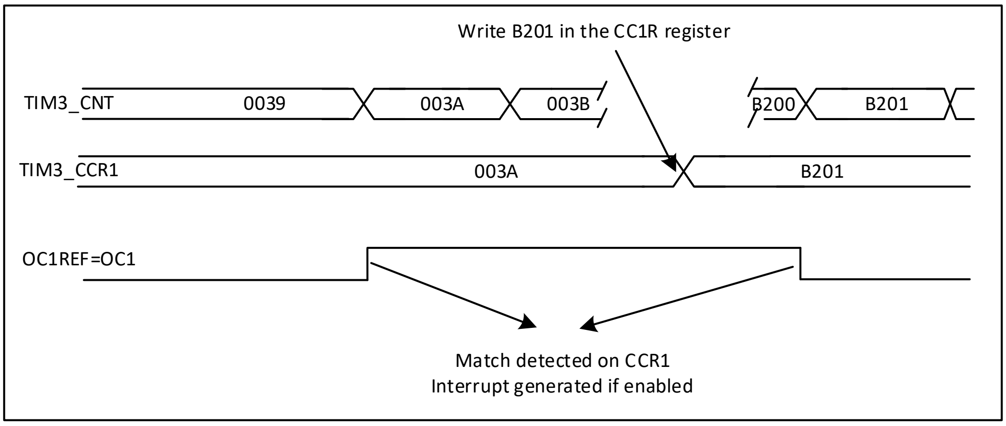


图 20-28 输出比较模式，翻转 OC1

20.3.9. 脉宽调制（PWM）模式

脉宽调制模式可以允许产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

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

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

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

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

虽然这样，为了遵守 OCREF_CLR 功能，OCREF_CLR 仅可以通过如下方式给出：

1：当比较结果改变。

2：在输出比较模式（由 frozen 配置（OCxM=000）切换到任意 PWM 模式（OCxM=”110”或”111”）,通过设置（TIMx_CCMRx 寄存器的 OCxM）。

在定时器运行时，这是通过软件强制切换到 PWM 模式。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 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。参看下面是一个 PWM 模式 1 的例子。当 TIMx_CNT < TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

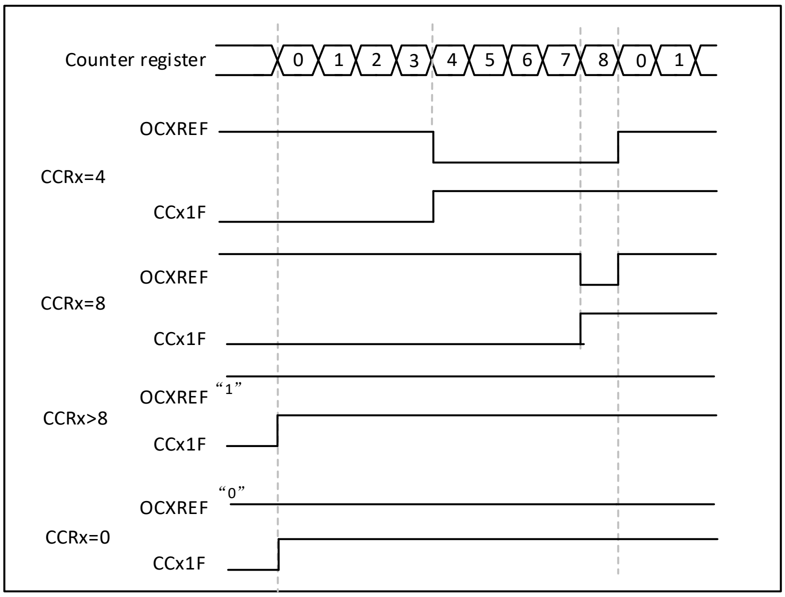


图 20-29 边沿对齐的 PWM 波形 (ARR=8)

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

下图给出一些中央对齐的 PWM 波形的例子

- TIMx_ARR = 8
- PWM 模式 1

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

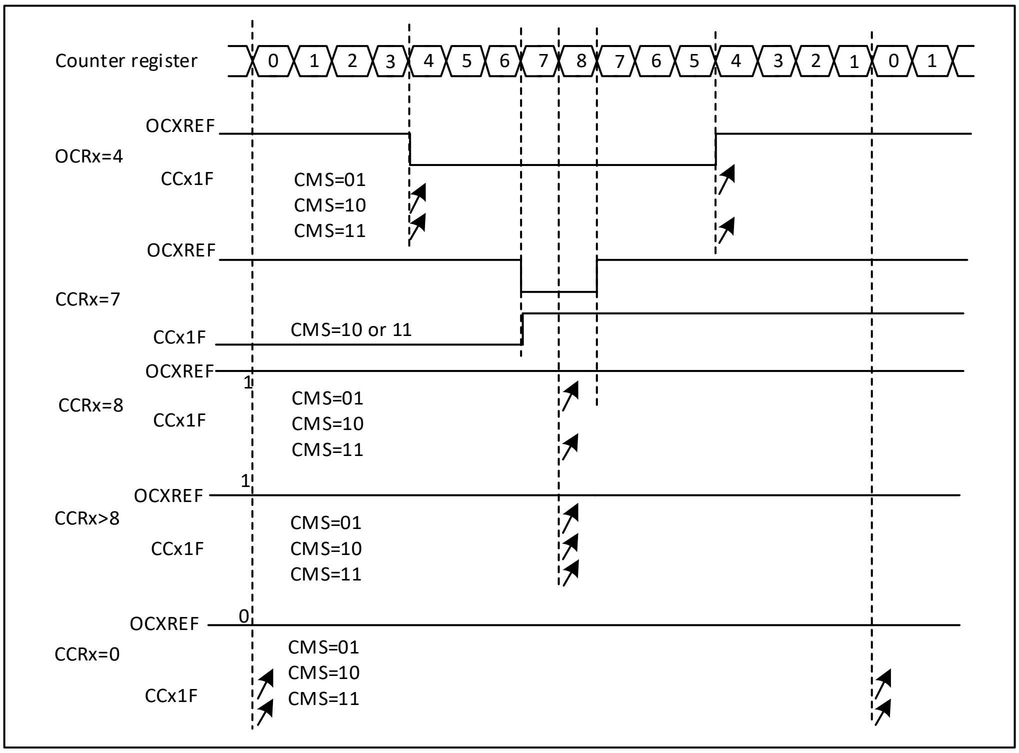


图 20-30 中央对齐的 PWM 波形(APR=8)

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。特别地：

– 如果写入计数器的值大于自动重加载的值(TIMx_CNT > TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。

– 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

■ 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

20.3.10. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后，产生一个脉宽可被程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIM3_CR1 寄存器的 OPM 位将选择单脉冲模式，这样可以让计数器自动的在产生下一个更新事件 UEV 时停止。

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

■ 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)

■ 向下计数方式：计数器 CNT > CCRx

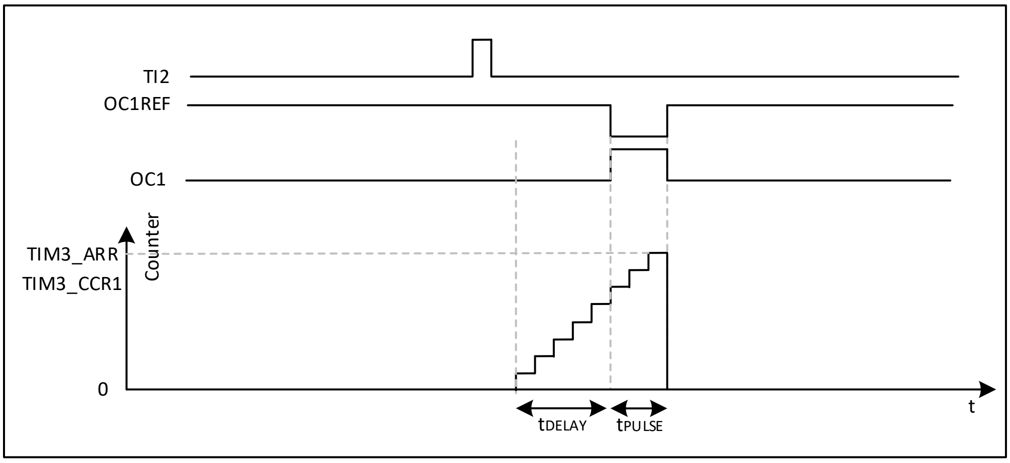


图 20-31 单脉冲模式的例子

例如，当需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 tPULSE 的正脉冲。

使用 TI2FP2 作为触发 1:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

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

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

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 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果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(和 OCx)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 PWM1 和 PWM2 模式时起作用。

20.3.11. 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，则置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 73，假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。 在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 20-1 计数方向与编码器信号的关系

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

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分输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。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 IC1)
- CC2S='01'(TIMx_CCMR2 寄存器，TI2FP2 映射到 IC2)
- 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)

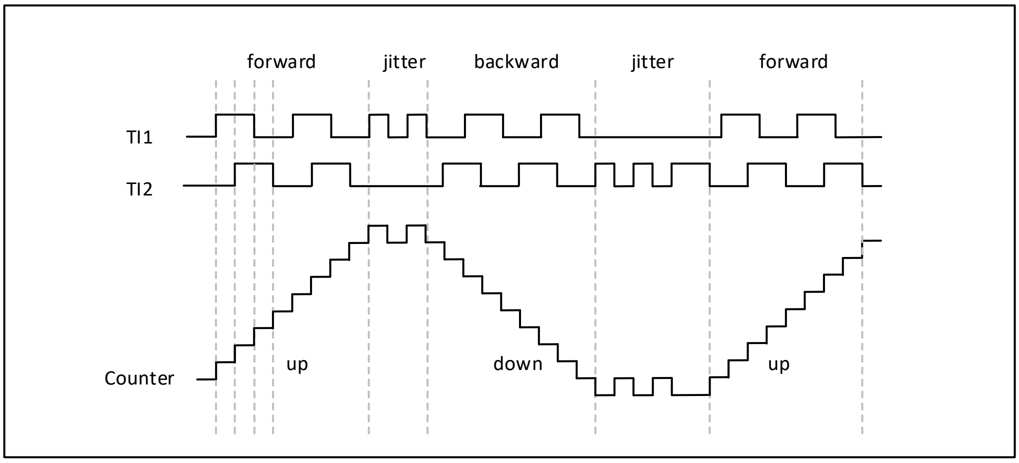


图 20-32 编码器模式下的计数器操作实例

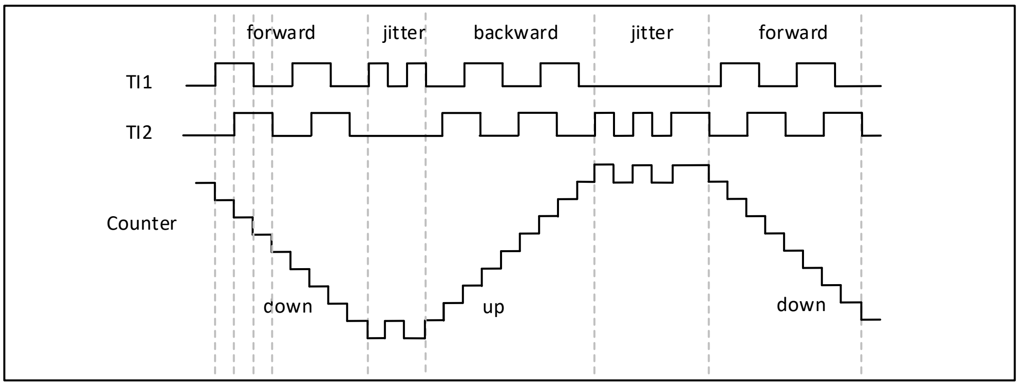


图 20-33 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前的位置信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度、加速度、减速度）。指示机械零点的编码器输出可被用作此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的，并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

20.3.12. 定时器输入异或功能

TIM3_CR2 寄存器的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

一个应用例子是霍尔传感器器接口.

20.3.13. 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

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

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0（和 CC1NP=0）以确定极性(只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIM3_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

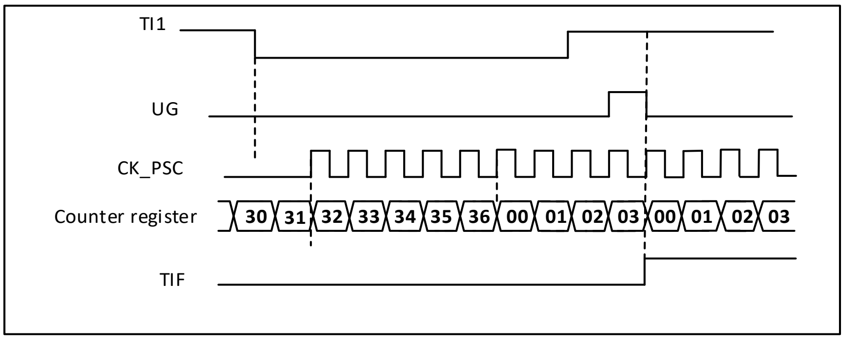


图 20-34 复位模式下的控制电路

Slave mode: Gated mode

按照选中的输入端的电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIM3_CCER 寄存器中 CC1P=1(和 CC1NP=0)以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

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

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

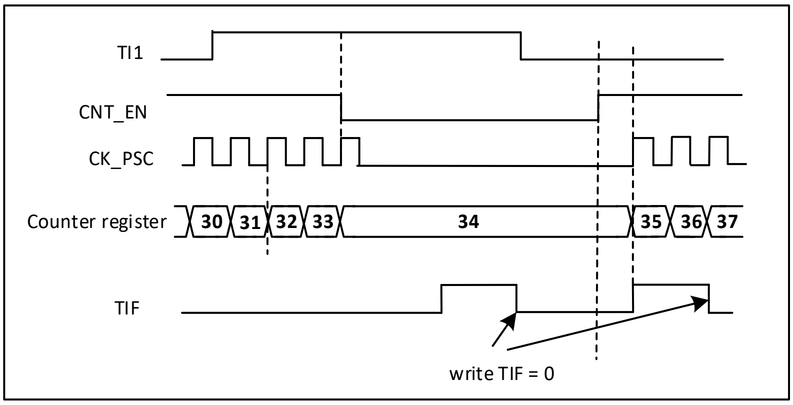


图 20-35 门控模式下的控制电路

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：

1) 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIM3x_CCMR1 寄存器中 CC2S=01。置 TIM3_CCER 寄存器中 CC2P=1(和 CC2NP=0)以确定极性(只检测低电平)。
2) 置 TIM3_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM3_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。 TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

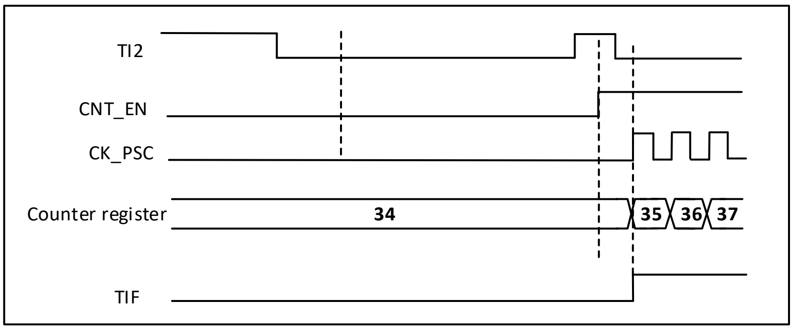


图 20-36 触发器模式下的控制电路

从模式：外部时钟模式 2 ＋ 触发模式

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

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

■ 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。

- 按如下配置通道 1，检测 TI 的上升沿：
  - IC1F=0000：没有滤波
  - 触发操作中不使用捕获预分频器，不需要配置
  - 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
  - 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

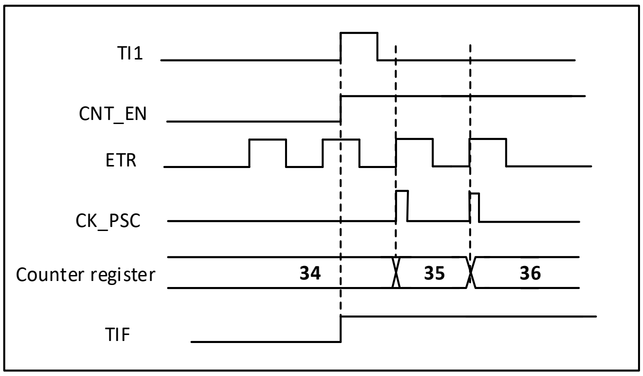


图 20-37 外部时钟模式 2 + 触发模式下的控制电路

20.3.14. 定时器同步

所有 TIM 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的预分频器

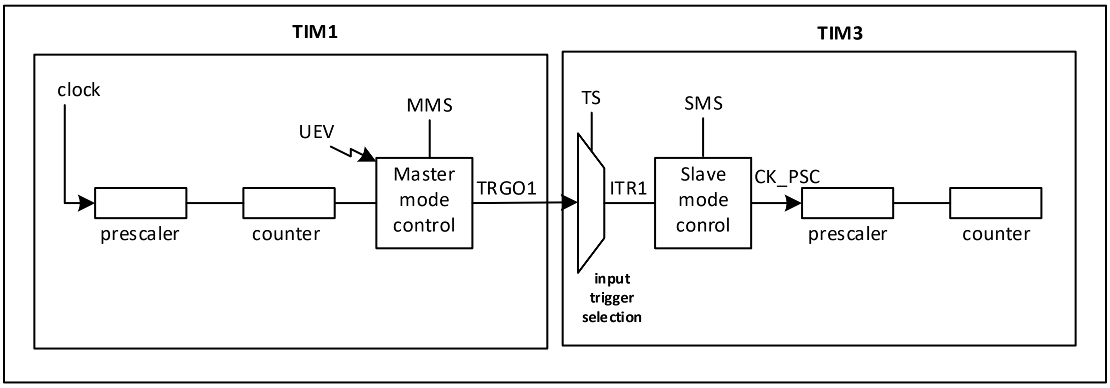


图 20-38 主/从定时器的例子

如：可以配置定时器 1 作为定时器 3 的预分频器。进行下述操作：

- 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。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 ITR1 作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式 1(TIM3_SMCR 寄存器的 SMS=111)；这样定时器 3 即可由定时器 1 周期性的上升沿(即定时器 1 的计数器溢出)信号驱动。
- 最后，必须设置相应(TIM3_CR1 寄存器)的 CEN 位分别启动两个定时器，确保先启动 Timer3，后启动 Timer1。

注： 如果 *OCx* 已被选中为定时器 *1* 的触发输出*(MMS=1xx)*，它的上升沿用于驱动定时器 *2* 的计数器。

使用一个定时器去使能另一个定时器

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。参考上图的连接。只当定时器 1 的 OC1REF 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_INT 除以 3(fCK_CNT=fCK_INT/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： 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。

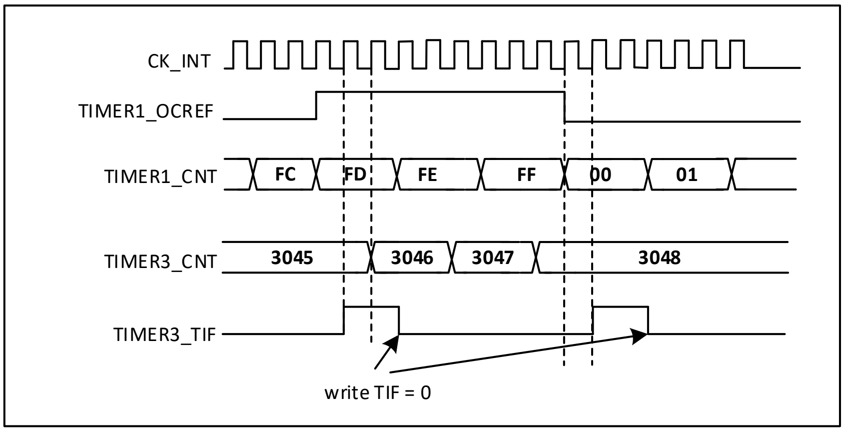


图 20-39 定时器 1 的 OC1REF 控制定时器 3

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。可以在启动定时器 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 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；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 2 随即停止。

- 配置定时器 1 为主模式，送出定时器使能信号（CNT_EN）做为触发输出(TIM1_CR2 寄存器 MMS=001 的)。
- 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。

- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM1_EGR 寄存器的 UG='1'，复位定时器 1。
- 置 TIM2_EGR 寄存器的 UG='1'，复位定时器 2。
- 写'0xE7'至定时器 2 的计数器(TIM2_CNT)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN='1'以使能定时器 2。
- 置 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。

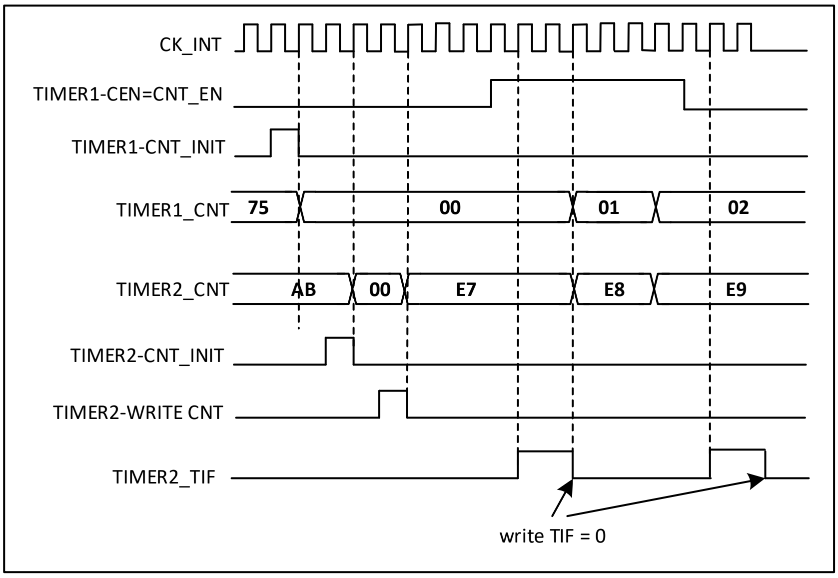


图 20-40 通过使能定时器 1 可以控制定时器 2

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 的更新事件使能定时器 3。参考 figure 132 的连接。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)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 位被自动地置'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(fCK_CNT=fCK_INT/3)。

- 配置定时器 1 为主模式，送出它的更新事件(UEV)做为触发输出(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

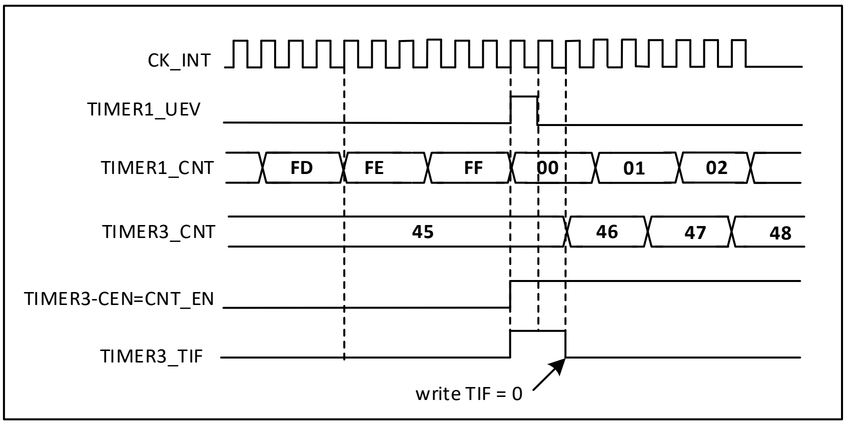


图 20-41 使用定时器 1 的更新触发定时器 3

在上一个例子中，可以在启动计数之前初始化两个计数器。下图显示在与上图相同配置情况下，使用触发模式而不是门控模式(TIM3_SMCR 寄存器的 SMS=110)的动作。

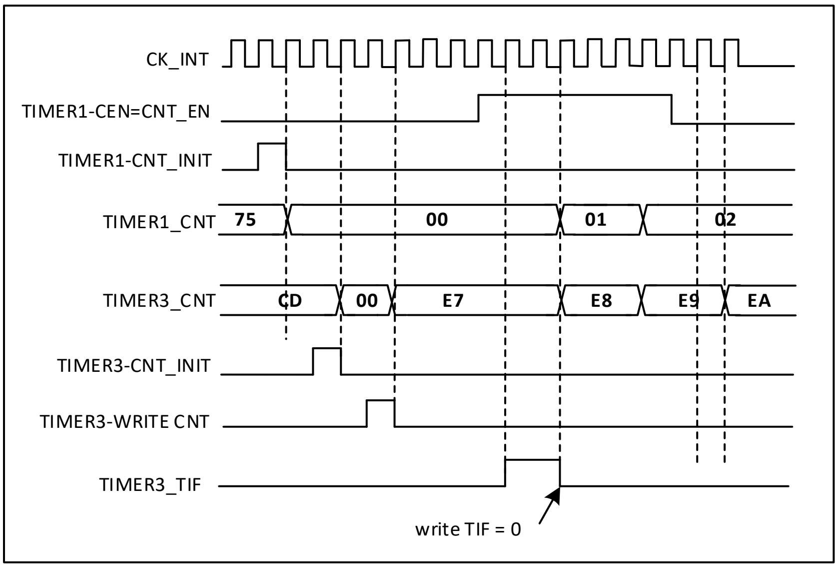


图 20-42 利用定时器 1 的使能触发定时器 3

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3，参见图18-38的连接方式。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，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1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数，两个TIF标志也同时被设置。

注：*在这个例子中，在启动之前两个定时器都被初始化(设置相应的UG位)，两个计数器都从0开始，但可以*

通过写入任意一个计数器寄存器(*TIMx_CNT*)在定时器间插入一个偏移。下图中能看到主/从模式下在定时器*1*的*CNT_EN*和*CK_PSC*之间有个延迟。

20.3.15. 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止工作。

20.4. 寄存器描述

TIM2 寄存器基地址：0x4000 0000

TIM3 寄存器基地址：0x4000 0400

20.4.1. TIM2/3 控制寄存器 1 (TIMx_CR1)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved			
9:8	CKD[1:0]	RW	00	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： tDTS = tCK_INT 01： tDTS = 2 x tCK_INT 10： tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： TIM3_ARR 寄存器没有缓冲 1： TIM3_ARR 寄存器被装入缓冲器
6:5	CMS[1:0]	RW	00	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 (TIM3_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计 数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。计数器交替地向上和向下计数。配置为输出的通道 (TIM3_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。

Bit	Name	R/W	Reset Value	Function
				11：中央对齐模式 3。计数器交替地向上和向下计数。计数器交替地向上和向下计数。配置为输出的通道(TIM3_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	方向 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读
3	OPM	RW	0	单脉冲模式 0：在发生更新事件时，计数器不停止 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

20.4.2. TIM2/3 控制寄存器 2 (TIMx_CR2)

Address offset:0x04

Reset value:0x0000 0000

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

-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	TI1S	MMS[2:0]			CCDS	Res	Res	Res
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	-	-	-

Bit	Name	R/W	Reset Value	Function
31:8	Reserved			保留，一直为 0
7	TI1S	RW	0	TI1 选择 0：TIMx_CH1 管脚连到 TI1 输入。 1：TIMx_CH1、 TIMx_CH2 和 TIMx_CH3 管脚经异或后连到 TI1 输入。
6:4	MMS[2:0]	RW	000	主模式选择 这两位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIM1_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：允许 – 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制从定时器的一个窗口。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIM1_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 一旦发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即是它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF 信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF 信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF 信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
3	CCDS	RW	0	捕获/比较的 DMA 选择 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2:0	Reseved	-	0	保留，始终读为 0

20.4.3. TIM2/3 从模式控制寄存器 (TIMx_SMCR)

Address offset:0x08

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW		RW				RW	RW			RW	RW		

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15	ETP	RW	0	外部触发极性 (External trigger polarity) 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0： ETR 不反相，高电平或上升沿有效； 1： ETR 被反相，低电平或下降沿有效。
14	ECE	RW	0	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是 ‘111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS[1:0]	RW	0	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01： ETRP 频率除以 2； 10： ETRP 频率除以 4； 11： ETRP 频率除以 8。
11:8	ETF[3:0]	RW	0	外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 fDTS 采样 0001：采样频率 fSAMPLING=fCK_INT， N=2 0010：采样频率 fSAMPLING=fCK_INT， N=4 0011：采样频率 fSAMPLING=fCK_INT， N=8 0100：采样频率 fSAMPLING=fDTS/2， N=6 0101：采样频率 fSAMPLING=fDTS/2， N=8 0110：采样频率 fSAMPLING=fDTS/4， N=6 0111：采样频率 fSAMPLING=fDTS/4， N=8 1000：采样频率 fSAMPLING=fDTS/8， N=6 1001：采样频率 fSAMPLING=fDTS/8， N=8 1010：采样频率 fSAMPLING=fDTS/16， N=5 1011：采样频率 fSAMPLING=fDTS/16， N=6 1100：采样频率 fSAMPLING=fDTS/16， N=8 1101：采样频率 fSAMPLING=fDTS/32， N=5 1110：采样频率 fSAMPLING=fDTS/32， N=6 1111：采样频率 fSAMPLING=fDTS/32， N=8
7	MSM	RW	0	主/从模式 (Master/slave mode) 0：无作用；

Bit	Name	R/W	Reset Value	Function
				1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS	RW	0	触发选择 (Trigger selection) 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0(ITR0) 001：内部触发 1(ITR1) 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) 更多有关 ITRx 的细节，参见表 5-1。 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	OCCS	RW	0	OCREF 清除选择位（OCREF clear selection） 这位选择了 OCREF 的清除源。 0：OCREF_CLR_INPUT 连接 OCREF_CLR 输入 1：OCREF_CLR_INPUT 连接 ETRF
2:0	SMS	RW	0	从模式选择 (Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表 20-2 TIMx 内部触发连接

从定时器	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM2	TIM1	TIM15	TIM3	TIM14_OC
TIM3	TIM1	TIM2	TIM15	TIM14_OC

20.4.4. TIM2/3 DMA/中断使能寄存器 (TIMx_DIER)

Address offset:0x0C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-	RW	-	RW	RW	RW	RW	RW	-	RW	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved			保留，一直读为 0
14	TDE	RW	0	TDE：允许触发 DMA 请求 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	Reserved	-	0	保留，一直读为 0
12	CC4DE	RW	0	CC4DE：允许捕获/比较 4 的 DMA 请求 0：禁止捕获/比较 4 的 DMA 请求 1：允许捕获/比较 4 的 DMA 请求
11	CC3DE	RW	0	CC3DE：允许捕获/比较 3 的 DMA 请求 0：禁止捕获/比较 3 的 DMA 请求 1：允许捕获/比较 3 的 DMA 请求
10	CC2DE	RW	0	CC2DE：允许捕获/比较 2 的 DMA 请求 0：禁止捕获/比较 2 的 DMA 请求 1：允许捕获/比较 2 的 DMA 请求
9	CC1DE	RW	0	CC1DE：允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	Reserved	-	0	保留，一直读为 0
6	TIE	RW	0	TIE：允许触发中断 0：禁止触发中断 1：允许触发中断
5	Reserved	-	0	保留，一直读为 0
4	CC4IE	RW	0	CC4IE：允许捕获/比较 4 中断 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	RW	0	CC3IE：允许捕获/比较 3 中断 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	CC2IE：允许捕获/比较 2 中断 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	CC1IE：允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断

Bit	Name	R/W	Reset Value	Function
				1：允许捕获/比较 1 中断
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断 1：允许更新中断

20.4.5. TIM2/3 状态寄存器(TIMx_SR)

Address offset:0x10

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	IC2IR	IC1IR
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	CC4OF	CC3OF	CC2OF	CC1OF	Res	Res	TIF	Res	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	-	RC_W0	RC_W0	RC_W0	RC_W0	-	-	RC_W0	-	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:2	Reserved			保留，一直读为 0
23	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
22	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
21	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
20	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生下降沿捕获事件。
19	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
18	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
17	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
16	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生上升沿捕获事件。

Bit	Name	R/W	Reset Value	Function
15:13	Reserved			保留，一直读为 0
12	CC4OF	RC_W0	0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。
11	CC3OF	RC_W0	0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。
10	CC2OF	RC_W0	0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时， CC1IF 的状态已经为‘1’。
8:07	Reserved			保留，一直读为 0
6	TIF	RC_W0	0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置‘1’。它由软件清‘0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	Reserved			保留，一直读为 0
4	CC4IF	RC_W0	0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
3	CC3IF	RC_W0	0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
2	CC2IF	RC_W0	0	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配后一个时钟周期，该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清‘0’。 0：无匹配发生； 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下， CC1IF 位变高

Bit	Name	R/W	Reset Value	Function
				如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置‘1’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生; 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生; 1：更新中断等待响应。当寄存器被更新时该位由硬件置‘1’： – 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 – 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当计数器 CNT 被触发事件重新初始化时。 (参考 TIM2/3 从模式控制寄存器(TIMx_SMCR))。

20.4.6. TIM2/3 事件产生寄存器(TIMx_EGR)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	TG	Res	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	--	-	-	-	-	-	W	-	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:7	Reserved	-	0	保留，一直为 0
6	TG	W	0	产生触发事件 该位由软件置 1，用于产生一个触发事件，由硬件自动清 0。 0：无动作; 1：TIM3_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	Reserved	-	0	保留，一直为 0
4	CC4G	W	0	产生捕获/比较 4 事件 参考 CC1G 描述
3	CC3G	W	0	产生捕获/比较 3 事件 参考 CC1G 描述
2	CC2G	W	0	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	W	0	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。

Bit	Name	R/W	Reset Value	Function
				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： 当前的计数器值捕获至 TIM1_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	W	0	产生更新事件 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清 0(但是预分频 系数不变)。若在中心对称模式下或 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 TIM3_ARR 的值。

20.4.7. TIM2/3 捕获/比较模式寄存器 1(TIMx_CCMR1)

Address offset:0x18

Reset value:0x0000 0000

Output compare mode:

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td></tr>
<tr><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>OC2CE</td><td colspan="3">OC2M[2:0]</td><td>OC2PE</td><td>CO2FE</td><td colspan="2" rowspan="2">CC2S[1:0]</td><td>OC1CE</td><td colspan="3">OC1M[2:0]</td><td>OC1PE</td><td>OC1FE</td><td colspan="2" rowspan="2">CC1S[1:0]</td></tr>
<tr><td colspan="4">IC2F[3:0]</td><td colspan="2">IC2PSC[1:0]</td><td colspan="4">IC1F[3:0]</td><td colspan="2">IC1PSC[1:0]</td></tr>
<tr><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td></tr>
</table>

Output compare mode

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15	OC2CE	RW	0	输出比较 2 清 0 使能
14:12	OC2M[2:0]	RW	000	输出比较 2 模式选择
11	OC2PE	RW	0	输出比较 2 预装载使能
10	OC2FE	RW	0	输出比较 2 快速使能
9:8	CC2S[1:0]	RW	00	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 TIM1_SMCR 寄存器的 TS 位选择）。

Bit	Name	R/W	Reset Value	Function
				注： CC2S 仅在通道关闭时(TIM1_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0	输出比较 1 清 0 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=0。
6:4	OC1M[2:0]	RW	00	输出比较 1 模式 该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 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用； 001： 匹配时设置通道1为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010： 匹配时设置通道1为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIM1_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 TIM3_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平 (OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIM3_CCR1 寄存器的预装载功能，可随时写入 TIM3_CCR1 寄存器，且新值马上起作用。 1：开启 TIM3_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM3_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。

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

输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:12	IC2F[3:0]	RW	0000	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	00	输入/捕获 2 预分频器
9:8	CC2S[1:0]	RW	0	捕获/比较 2 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入， IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIMx_SMCR 寄存器的 TS 位选择）。 注： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	0000	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， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8， N=6

Bit	Name	R/W	Reset Value	Function
				0001：采样频率 fSAMPLING=fCK_INT， N=2 1001：采样频率 fSAMPLING=fDTS/8， N=8 0010：采样频率 fSAMPLING=fCK_INT， N=4 1010：采样频率 fSAMPLING=fDTS/16， N=5 0011：采样频率 fSAMPLING=fCK_INT， N=8 1011：采样频率 fSAMPLING=fDTS/16， N=6 0100：采样频率 fSAMPLING=fDTS/2， N=6 1100：采样频率 fSAMPLING=fDTS/16， N=8 0101：采样频率 fSAMPLING=fDTS/2， N=8 1101：采样频率 fSAMPLING=fDTS/32， N=5 0110：采样频率 fSAMPLING=fDTS/4， N=6 1110：采样频率 fSAMPLING=fDTS/32， N=6 0111：采样频率 fSAMPLING=fDTS/4， N=8 1111：采样频率 fSAMPLING=fDTS/32， N=8
3:2	IC1PSC[1:0]	RW	00	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S[1:0]	RW	00	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 TIM3_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。

20.4.8. TIM2/3 捕获/比较模式寄存器 2(TIMx_CCMR2)

Address offset:0x1C

Reset value:0x0000 0000

Output compare mode:

<table>
<tr><th>31</th><th>30</th><th>29</th><th>28</th><th>27</th><th>26</th><th>25</th><th>24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th></tr>
<tr><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td><td>Res</td></tr>
<tr><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>OC4CE</td><td colspan="3">OC4M[2:0]</td><td>OC4PE</td><td>CO4FE</td><td colspan="2" rowspan="2">CC4S[1:0]</td><td>OC3CE</td><td colspan="3">OC3M[2:0]</td><td>OC3PE</td><td>OC3FE</td><td colspan="2" rowspan="2">CC3S[1:0]</td></tr>
<tr><td colspan="4">IC4F[3:0]</td><td colspan="2">IC4PSC[1:0]</td><td colspan="4">IC3F[3:0]</td><td colspan="2">IC3PSC[1:0]</td></tr>
<tr><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td><td>RW</td></tr>
</table>

输出比较模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15	OC4CE	RW	0	输出比较 4 清 0 使能
14:12	OC4M[2:0]	RW	000	输出比较 4 模式
11	OC4PE	RW	0	输出比较 4 预装载使能
10	OC4FE	RW	0	输出比较 4 快速使能
9:8	CC4S[1:0]	RW	00	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(TIM1_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0	输出比较 3 清 0 使能
6:4	OC3M[2:0]	RW	00	输出比较 3 模式
3	OC3PE	RW	0	输出比较 3 预装载使能
2	OC3FE	RW	0	输出比较 3 快速使能
1:0	CC3S[1:0]	RW	00	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)才是可写的。

输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:12	IC4F[3:0]	RW	0000	输入捕获 4 滤波器
11:10	IC4PSC[1:0]	RW	00	输入/捕获 4 预分频器
9:8	CC4S[1:0]	RW	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 TIM3_SMCR 寄存器的 TS 位选择）。

Bit	Name	R/W	Reset Value	Function
				注： CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F[3:0]	RW	0000	输入捕获 3 滤波器
3:2	IC3PSC[1:0]	RW	00	输入/捕获 3 预分频器
1:0	CC3S[1:0]	RW	00	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 TI3 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 TIM3_SMCR 寄存器的 TS 位选择）。 注： CC3S 仅在通道关闭时(TIM3_CCER 寄存器的 CC3E=0)才是可写的。

20.4.9. TIM2/3 捕获/比较使能寄存器(TIMx_CCER)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CC4NP	Res	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW	RW	-	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15	CC4NP	RW	0	输入/捕获 4 互补输出极性。参考 CC1NP 的描述。
14	Reserved	-	0	保留，一直读为 0
13	CC4P	RW	0	输入/捕获 4 输出极性。参考 CC1P 的描述。
12	CC4E	-	0	输入/捕获 4 输出使能。参考 CC1E 的描述。
11	CC3NP	RW	0	输入/捕获 3 互补输出极性。参考 CC1NP 的描述。
10	Reserved	-	0	保留，一直读为 0
9	CC3P	RW	0	输入/捕获 3 输出极性。参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能。参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	Reserved	-	0	保留，一直读为 0
5	CC2P	RW	0	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0： OC1N 高电平有效 1： OC1N 低电平有效 这位是和 CC1P 联合使用定义 TI1FP1/TI2FP1 极性，参考 CC1P 描述
2	Reserved	-	0	保留，一直读为 0

Bit	Name	R/W	Reset Value	Function
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。
0	CC1E	RW	0	输入/捕获 1 输出使能 CC1 通道配置为输出： 0：关闭 - OC1 禁止输出， 1：开启 - OC1 信号输出到对应的输出引脚， CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 0：捕获使能

标准 OCx 通道的输出控制

CcxE 位	OCx output State
0	输出禁止（OCx=0,OCx_EN=0）
1	OCx=OCxREF+Polarity,OCx_EN=1

20.4.10. TIM2/3 计数器(TIMx_CNT)

Address offset:0x24

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0

Bit	Name	R/W	Reset Value	Function
15:0	CNT[15:0]	RW	0	计数器的值

20.4.11. TIM2/3 预分频器(TIMx_PSC)

Address offset:0x28

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:0	PSC[15:0]	RW	0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清 ‘0’ 或被工作在复位模式的从控制器清 ‘0’ 。

20.4.12. TIM2/3 自动重装载寄存器 (TIMx_ARR)

Address offset:0x2C

Reset value:0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:0	ARR[15:0]	RW	FFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 12.4.1：时基单元有关 ARR 的更新和动作。 当自动重装载的值为空时，计数器不工作。

20.4.13. TIM2/3 捕获/比较寄存器 1(TIMx_CCR1)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:0	CCR1[15:0]	RW	0	捕获/比较 1 的值 若 CC1 通道配置为输出: CCR1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIM3_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM3_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1 通道配置为输入: CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

20.4.14. TIM2/3 捕获/比较寄存器 2(TIMx_CCR2)

Address offset:0x38

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR2[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31:16	Reserved			保留，一直为 0
15:0	CCR2[15:0]	RW	0	捕获/比较 2 的值 若 CC2 通道配置为输出: CCR2 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIM3_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM3_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC2 通道配置为输入: CCR2 包含了由上一次输入捕获 2 事件（IC2）传输的计数器值。

20.4.15. TIM2/3 捕获/比较寄存器 3(TIMx_CCR3)

Address offset:0x3C

Reset value:0x0000 0000

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

CCR3[15:0]
RW/RO

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR3[15:0]	RW	0	捕获/比较 3 的值 **若 CC3 通道配置为输出:** CCR3 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIM3_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM3_CNT 比较的值，并且在 OC 端口上输出信号。 **若 CC3 通道配置为输入:** CCR3 包含了由上一次输入捕获 3 事件（IC3）传输的计数器值。

20.4.16. TIM2/3 捕获/比较寄存器 4(TIMx_CCR4)

Address offset:0x40

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR4[15:0]	RW	0	捕获/比较 4 的值 **若 CC4 通道配置为输出:** CCR4 包含了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIM3_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIM3_CNT 比较的值，并且在 OC 端口上输出信号。 **若 CC4 通道配置为输入:** CCR4 包含了由上一次输入捕获 4 事件（IC4）传输的计数器值。

20.4.17. TIM2/3 DMA 控制寄存器(TIMx_DCR)

Address offset:0x48

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：13	Reserved			保留，一直为 0
12:8	DBL[4:0]	RW	0 0000	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器的地址进行读或写 时，定时器则进行一次连续传送），即：定义被传送的字 节数目： 00000： 1 字节 00001： 2 字节 00010： 3 字节 .... 10001： 18 字节 例：我们考虑这样的传输：DBL=7，DBA=TIM2_CR1 - 如果 DBL=7，DBA=TIM2_CR1 表示待传输数据的地址， 那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引), 其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入 或者读出数据的地址，这样数据的传输将发生在从地址 (TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存 器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包 含第一个 LSB 字节，以此类推。因此对于定时器，用户必 须指定由 DMA 传输的数据宽度。
7:5	Reserved	RW	0	保留，始终读为 0
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIMx_DMAR 寄存器的地址进行读或写 时）， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏 移量： 00000： TIMx_CR1， 00001： TIMx_CR2， 00010： TIMx_SMCR， ......

20.4.18. TIM2/3 连续模式的 DMA 地址 (TIMx_DMAR)

Address offset:0x4C

Reset value:0x0000 0000

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

DMAB[31:16]															
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAB[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
31：0	DMAB[31:0]	RW	0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作: TIMx_CR1 地址 + DBA + DMA 索引，其中: “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址; “DBA”是 TIMx_DCR 寄存器中定义的基地址; “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

20.4.19. TIM2/3 寄存器映像

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	TIMx_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x04	TIMx_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
	Reset value																									0	0	0	0	0			
0x08	TIMx_SMCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETP.	ECE.	ETPS.		ETF.				MSM	TS[2:0]			OCCS	SMS[2:0]		
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIMx_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	Reset value																		0		0	0	0	0	0		0		0	0	0	0	0
0x10	TIMx_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4IF.	IC3IF.	IC2IF.	IC1IF.	IC4IR.	IC3IR.	IC2IR.	IC1IR.	Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	Reset value																				0	0	0	0			0		0	0	0	0	0
0x14	TIMx_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
	Reset value																										0		0	0	0	0	0
0x18	TIMx_CCMR1(output compare	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	

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
	mode)																																
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIMx_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIMx_CCMR2(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	TIMx_CCMR2(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC4F[3:0]				IC4PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3PSC [1:0]		CC3S[1:0]	
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TIMx_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC4NP	Res.	CC4P	CC4E	CC3NP	Res.	CC3P	CC3E	CC2NP	Res.	CC2P	CC2E	CC1NP	Res.	CC1P	CC1E
	Reset value																									0		0	0	0		0	0
0x24	TIMx_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIMx_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIMx_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIMx_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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
0x38	TIMx_CCR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR2[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TIMx_CCR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR3[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TIMx_CCR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR4[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TIMx_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBL[4:0]					Res.			DBA[4:0]					
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	TIMx_DMAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

21.基本定时器 (TIM6/7)

21.1. TIM6 和 TIM7 简介

基本定时器 TIM6 和 TIM7 各包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。

TIM6 和 TIM7 都是完全独立的，没有互相共享任何资源。

21.2. TIM6 和 TIM7 的主要特性

- 16 位自动装载向上计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 在更新事件发生时产生中断/DMA

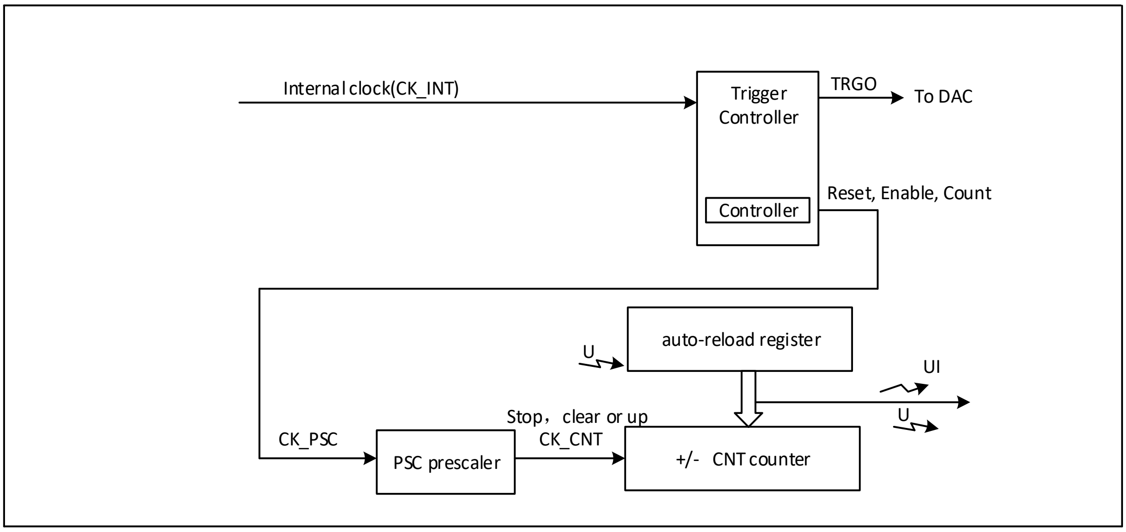


图 21-1 TIM14 架构框图

21.3. TIM6 和 TIM7 功能描述

21.3.1. 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位向上计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包括：

- 计数器寄存器（TIMx_CNT）
- 预分频寄存器（TIMx_PSC）
- 自动重装载寄存器（TIMx_ARR）

自动重装载寄存器是预先装载的，写或者读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容一直或在每次的更新事件 UEV 时传送

到影子寄存器。当计数器达到上溢出并当 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 TIM14_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_CR 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按 1 到 65535 之间的任意值分频。它是基于一个（在 TIMx_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在计数器运行时，更改预分频器参数的例子。

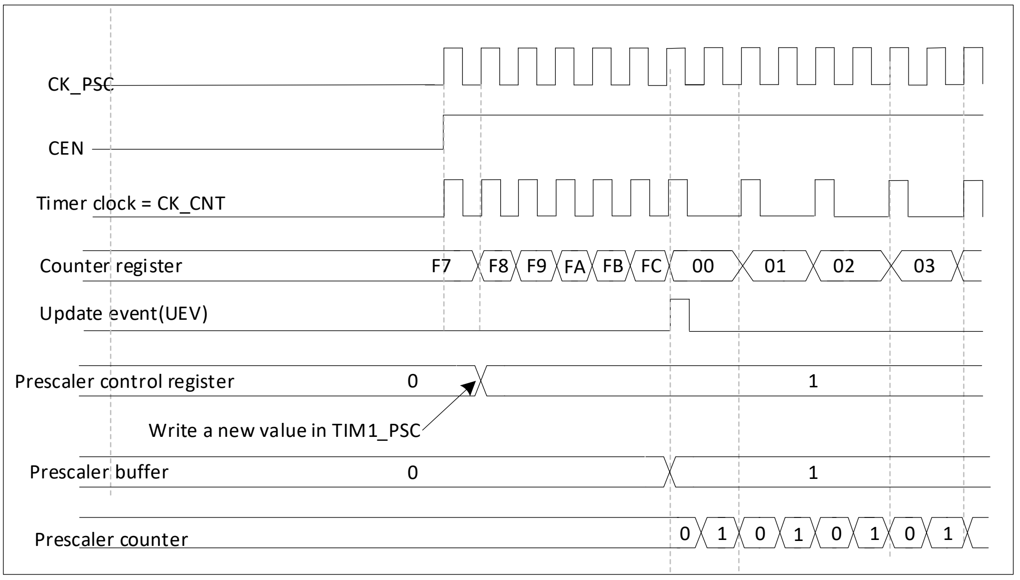


图 21-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

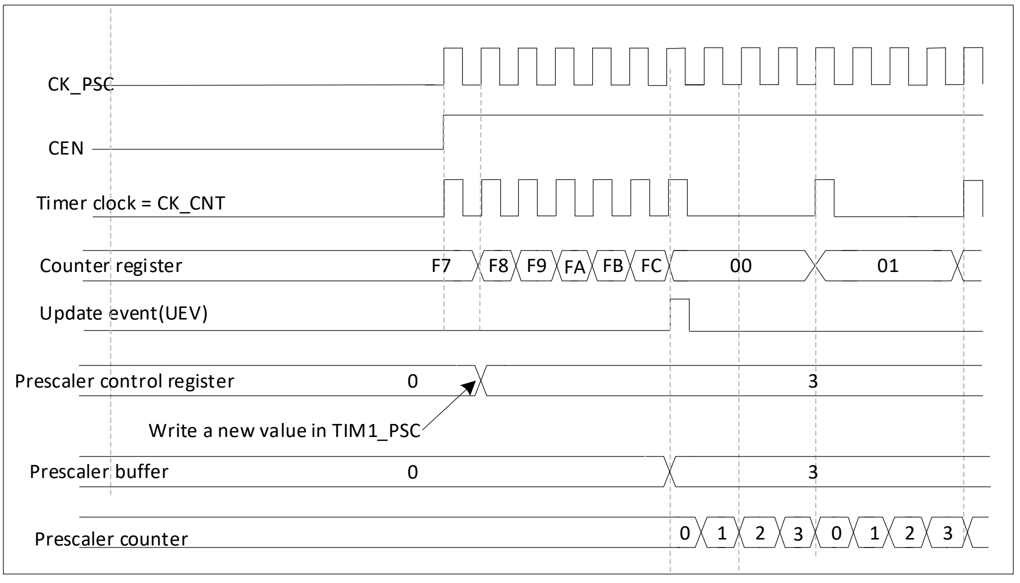


图 21-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

向上计数模式

计数器从 0 计数到自动装载值（TIMx_ARR 寄存的值），然后又从 0 重新开始计数，并产生一个计数器上溢事件。

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。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。虽然如此，但是计数器依旧从 0 开始，同时预分频器的计数也被清 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 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置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下面的例程显示了几个在不同频率下的计数器行为，当 TIMx_ARR=0X36.

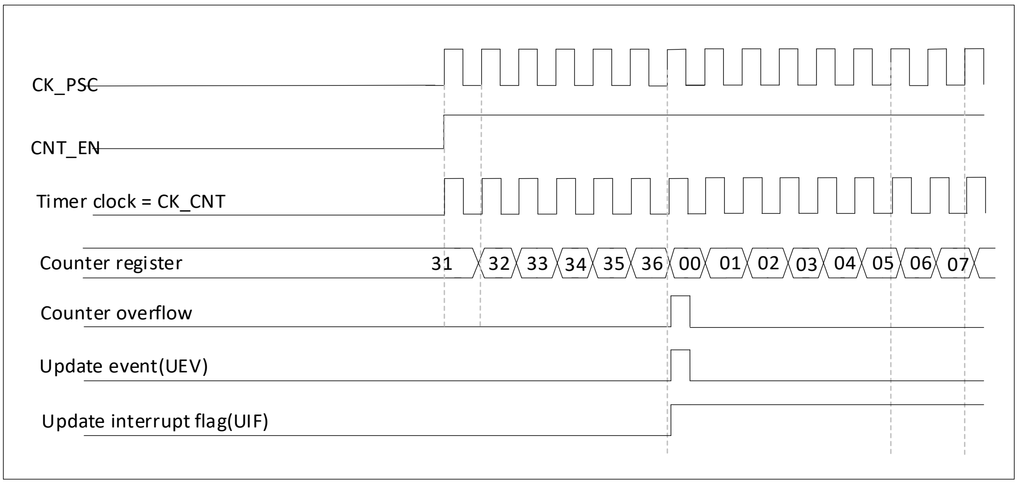


图 21-4 计数器时序图，内部时钟分频因子为 1

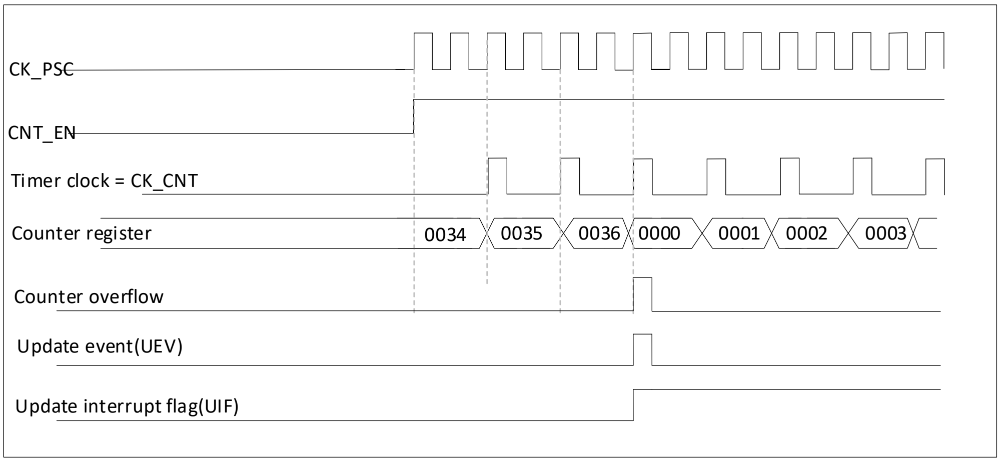


图 21-5 计数器时序图，内部时钟分频因子为 2

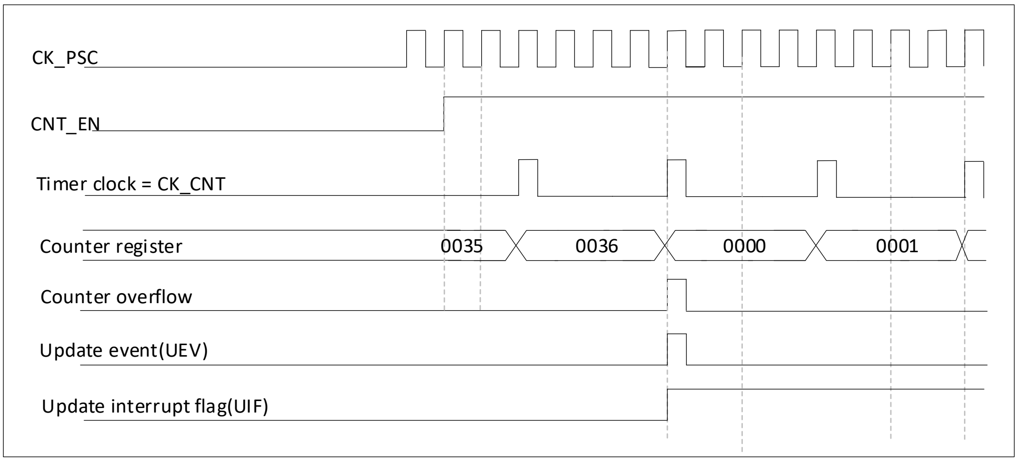


图 21-6 计数器时序图，内部时钟分频因子为 4

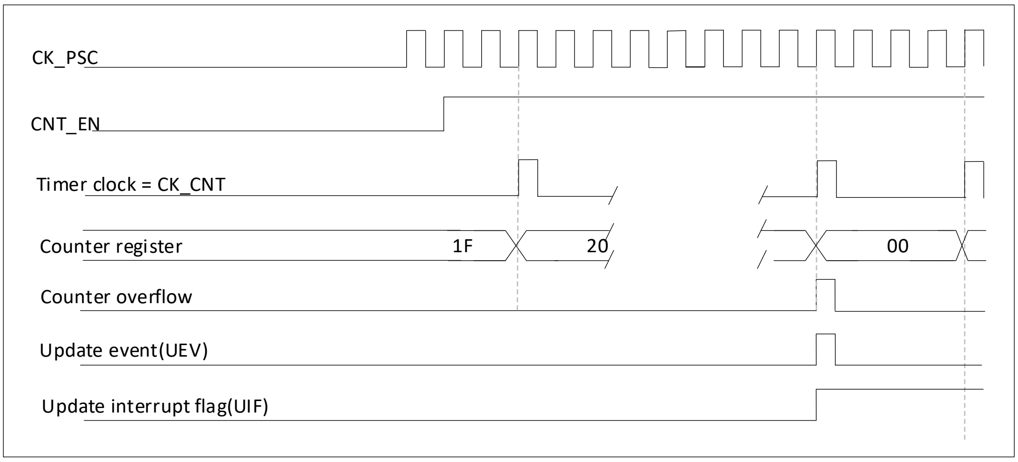


图 21-7 计数器时序图，内部时钟分频因子为 N

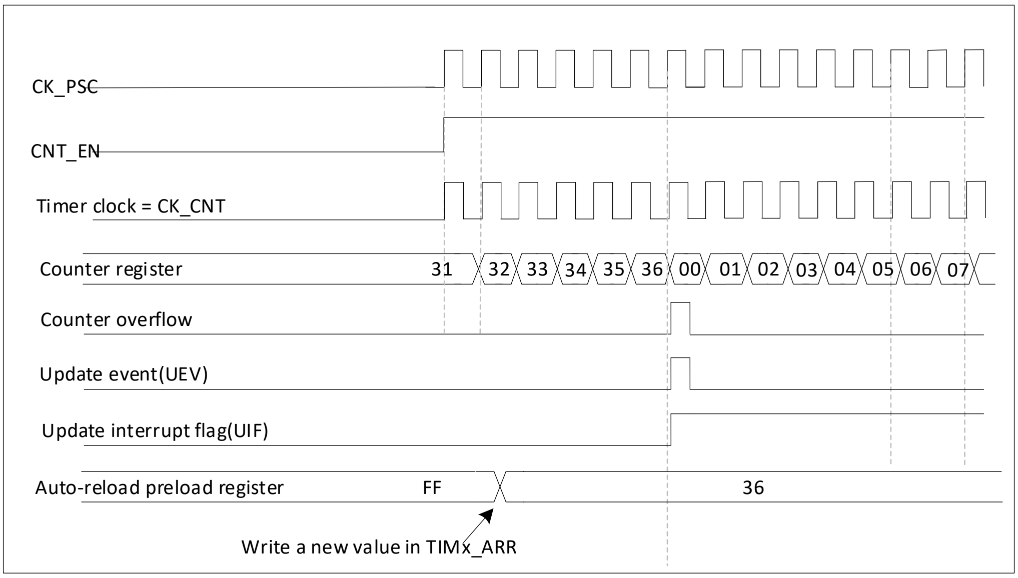


图 21-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

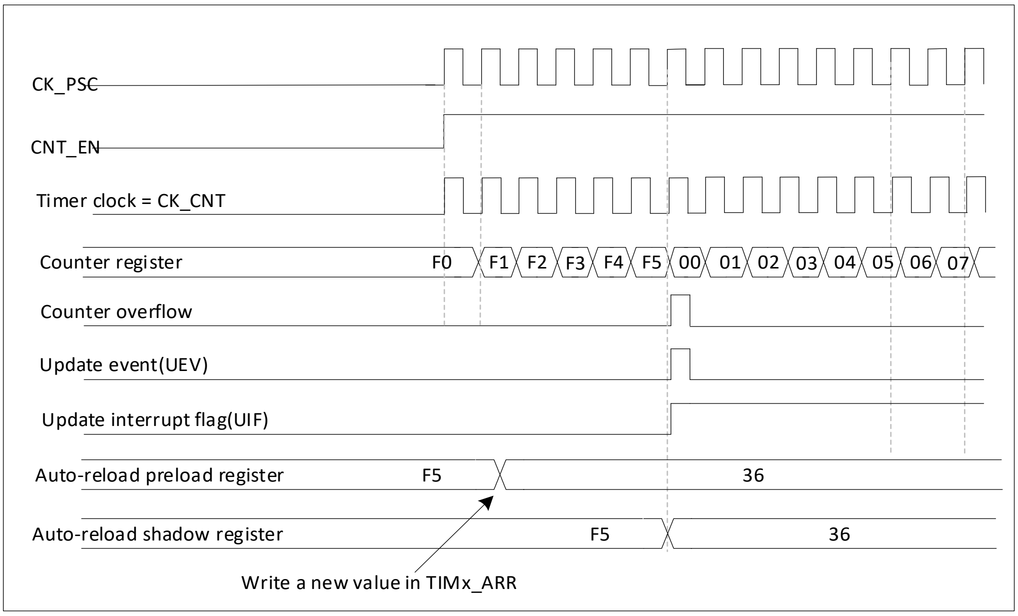


图 21-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

21.3.2. 时钟源

计数器的时钟由内部时钟（CK_INT）提供。TIMx_CR1 寄存器的 CEN 位和 TIMx_EGR 寄存器的 UG 位是实际的控制位（除了 UG 位被自动清除外），只能通过软件改变他们。一旦置 CEN 位为 1，内部时钟即向分频器提供时钟。

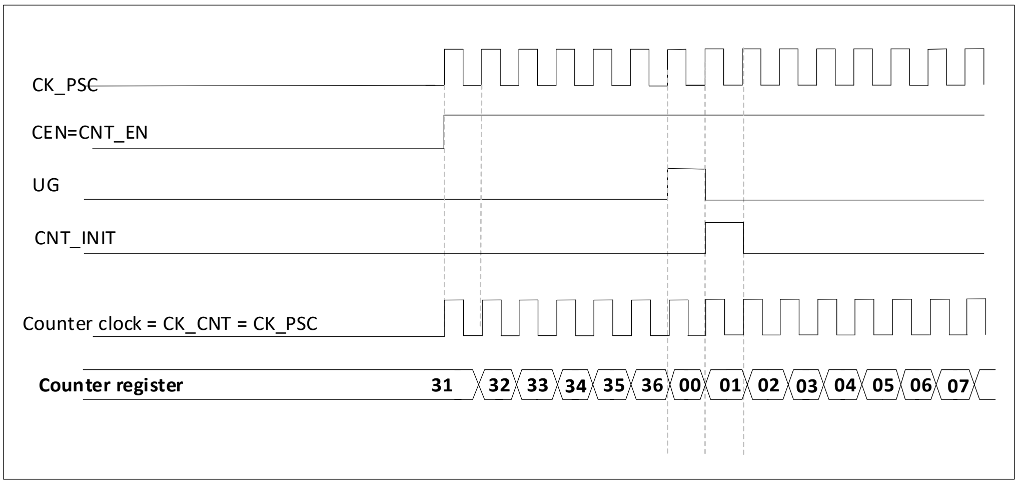


图 21-10 一般模式下的控制电路，内部时钟分频因子为 1

21.3.3. 调试模式

当芯片进入调试模式（M0+停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器或者继续正常操作，或者停止。

21.4. TIM6 和 TIM7 寄存器

21.4.1. TIM6 和 TIM7 控制寄存器 1 (TIMx_CR1)

Address offset:0x00

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res		ARPE	Res	Res	Res	OPM	URS	UDIS	CEN
-	-	-	-	-	-	-		RW	-	-	-	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	0	保留，一直读为 0
7	ARPE	RW	0	自动重装载预装载允许位 0： TIMx_ARR 寄存器没有缓冲 1： TIMx_ARR 寄存器被装入缓冲器
6:4	Reserved	-	0	保留，一直读为 0
3	OPM	RW	0	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： －计数器溢出/下溢 －设置 UG 位 －从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： －计数器溢出/下溢 －设置 UG 位 －从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

21.4.2. TIM6 和 TIM7 控制寄存器 2（TIMx_CR2）

Address offset:0x04

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	0	保留，一直读为 0
6:4	MMS[2:0]	RW	0	主模式选择 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能 – 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 注意：从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。
3:0	Reserved	-	0	保留，一直读为 0

21.4.3. TIM6 和 TIM7 DMA/中断使能寄存器 (TIM14_DIER)

Address offset:0x0C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved			保留，一直为 0
8	UDE	RW	0	UDE:允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7:1	Reserved			保留，一直为 0
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断

				1：允许更新中断

21.4.4. TIM6 和 TIM7 状态寄存器(TIMx_SR)

Address offset:0x010

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-		保留，一直为 0
0	UIF	RC_W0	0	更新中断标记，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，产生更新事件上溢； －若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

21.4.5. TIM6 和 TIM7 事件产生寄存器(TIMx_EGR)

Address offset:0x14

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：1	Reserved	-	0	保留，一直为 0
0	UG	W	0	产生更新事件，该位由软件置 1，由硬件自动清 0。 0：无动作； 1：重新初始化计数器，并产生一个寄存器的更新事件。注意预分频器的计数器也被清 0(但是预分频系数不变)。

21.4.6. TIM6 和 TIM7 计数器(TIMx_CNT)

Address offset:0x24

Reset value:0x0000 0000

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

RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CNT[15:0]	RW	0	计数器的值

21.4.7. TIM6 和 TIM7 预分频器(TIMx_PSC)

Address offset:0x28

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	PSC[15:0]	RW	0	预分频器的值 计数器的时钟频率（CK_CNT）等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器 被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

21.4.8. TIM6 和 TIM7 自动重装载寄存器 (TIMx_ARR)

Address offset:0x2C

Reset value:0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	ARR[15:0]	RW	FFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 12.4.1：时基单元有关 ARR 的更新和动作。 当自动重装载的值为空时，计数器不工作。

21.4.9. TIM6 和 TIM7 寄存器映像

Offset	Reg-ister	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
0x	TIMx_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		ARPE	Res.		Res.	OPM.	URS	UDIS	CEN

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
00	Reset value																									0				0	0	0	0
0x04	TIMx_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MMS [2:0]			Res.	Res.	Res.	Res.
	Reset value																										0	0	0				
0x0C	TIMx_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIE
	Reset value																								0								0
0x10	TIMx_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
	Reset value																																0
0x14	TIMx_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
	Reset value																																0
0x18	TIM14_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	OC1FE	CC1S[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x24	TIM14_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

22.通用定时器 (TIM14)

22.1. TIM14 简介

通用定时器 TIM14 由可编程预分频器驱动的 16 位自动装载计数器构成。

它适用于多种场合，包括测量输入信号的脉冲长度(输入捕获)或者产生输出波形(输出比较和 PWM)。

使用定时器预分频器和 RCC 时钟控制器预分频器，脉冲长度和波形周期可以在几个微秒到几个毫秒间调整。

通用定时器（TIMx）都是完全独立的，没有互相共享任何资源。它们可以一起同步操作，参见 TIM3 的同步章节。

22.2. TIM14 主要特性

- 16 位自动装载向上计数器
- 16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 1 个独立通道，作为：
  - 输入捕获
  - 输出比较
  - PWM 生成（边缘对齐模式）
- 如下事件发生时产生中断
  - 更新：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

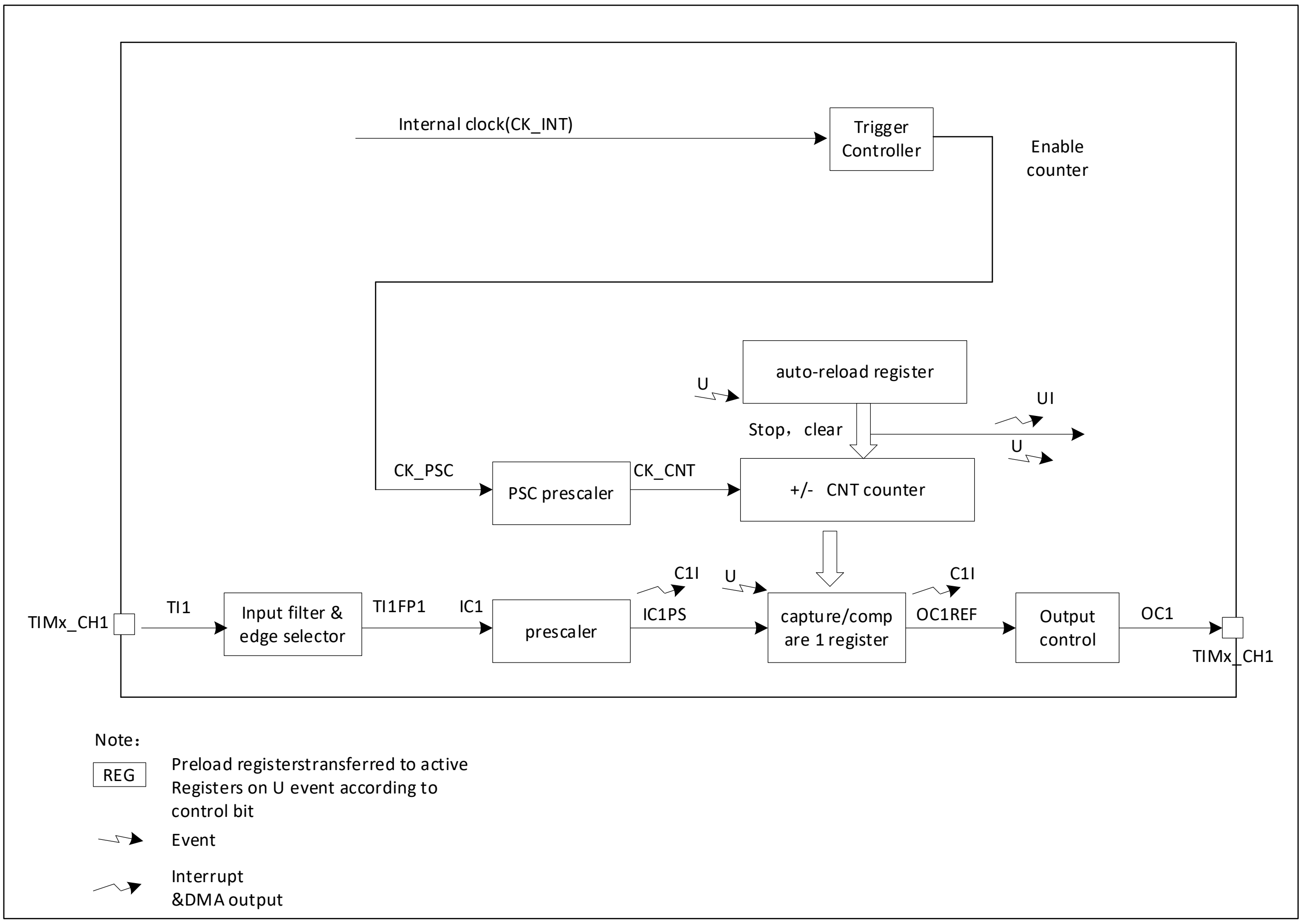


图 22-1 TIM14 架构框图

22.3. TIM14 功能描述

22.3.1. 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位向上计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包括：

- 计数器寄存器（TIM14_CNT）
- 预分频寄存器（TIM14_PSC）
- 自动重装载寄存器（TIM14_ARR）

自动重装载寄存器是预先装载的，写或者读自动重装载寄存器将访问预装载寄存器。根据在 TIM14_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一直或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到上溢出并当 TIM14_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 TIM14_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 TIM14_CR 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按 1 到 65535 之间的任意值分频。它是基于一个（在 TIM14_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在计数器运行时，更改预分频器参数的例子。

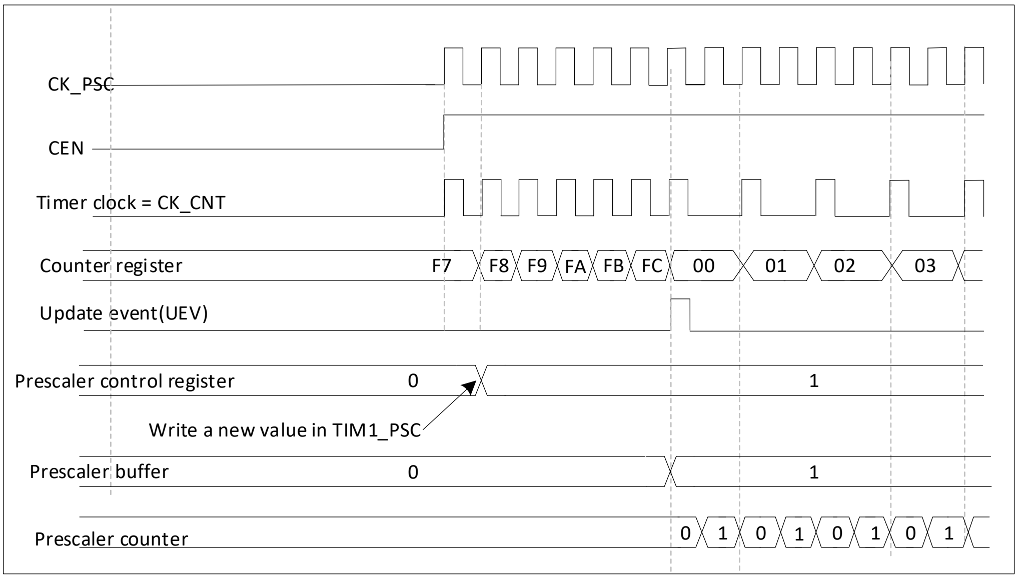


图 22-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

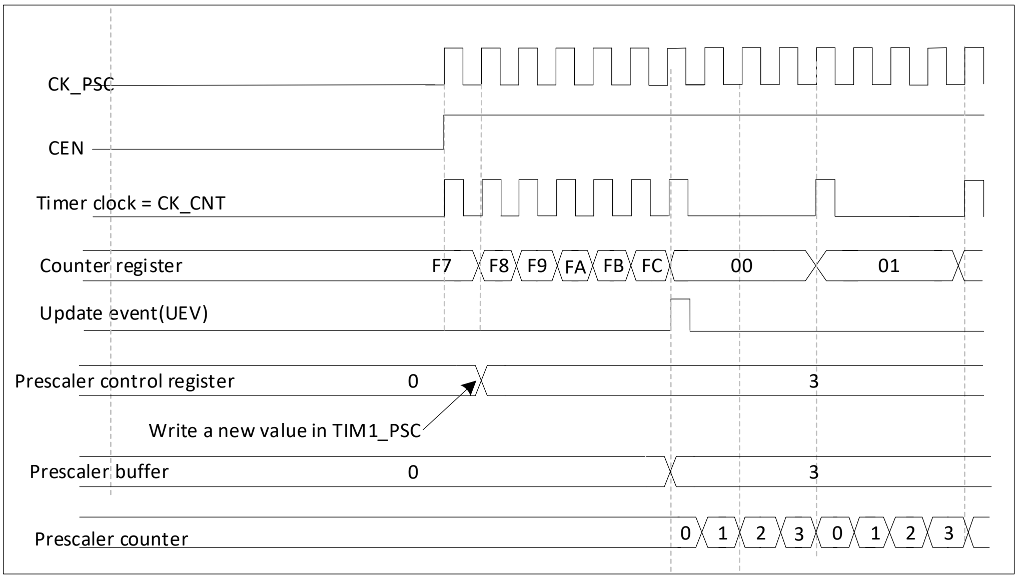


图 22-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

向上计数模式

计数器从 0 计数到自动装载值（TIM14_ARR 寄存的值），然后又从 0 重新开始计数，并产生一个计数器上溢事件。

在 TIM14_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。

设置 TIM14_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。虽然如此，但是计数器依旧从 0 开始，同时预

分频器的计数也被清 0(但预分频器的数值不变)。此外，如果设置了 TIM14_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 URS 位)设置更新标志位(TIM14_SR 寄存器中的 UIF 位)。

- 自动装载影子寄存器被重新置入预装载寄存器的值(TIM14_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIM14_PSC 寄存器的内容)。

下面的例程显示了几个在不同频率下的计数器行为，当 TIMx_ARR=0X36.

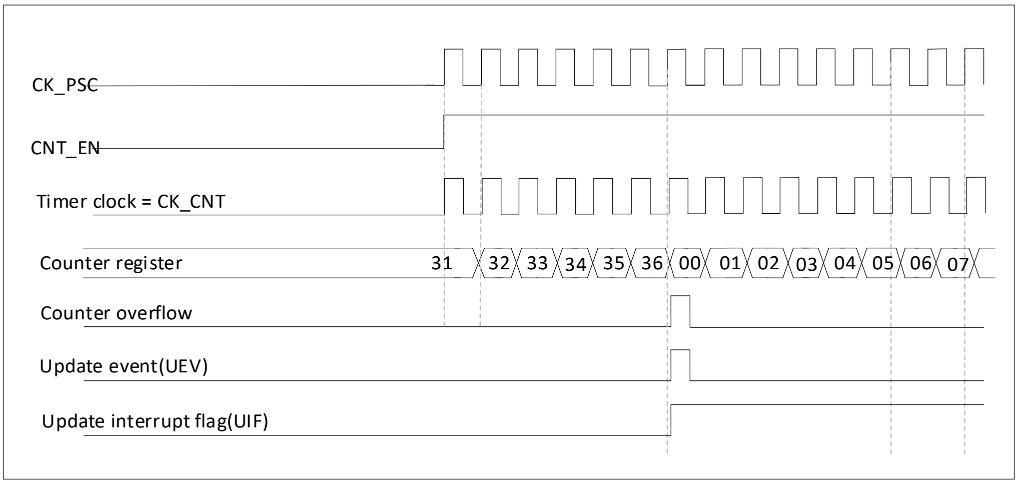


图 22-4 计数器时序图，内部时钟分频因子为 1

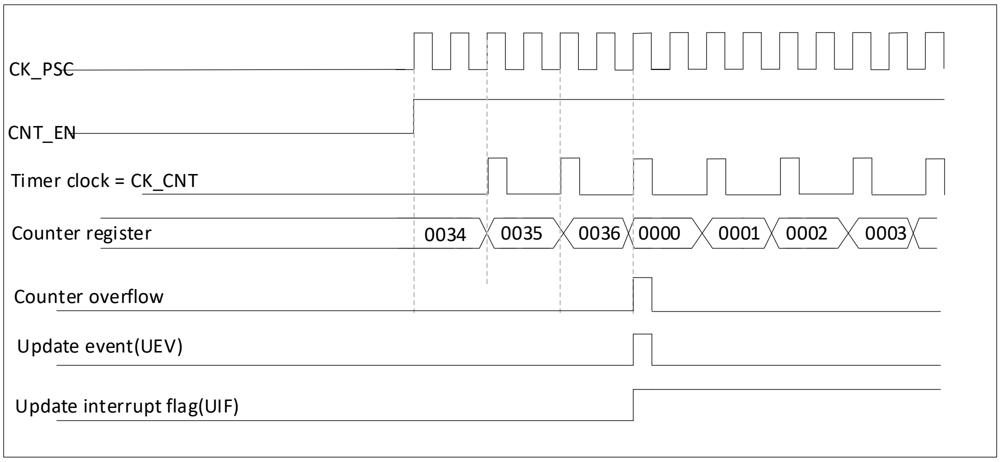


图 22-5 计数器时序图，内部时钟分频因子为 2

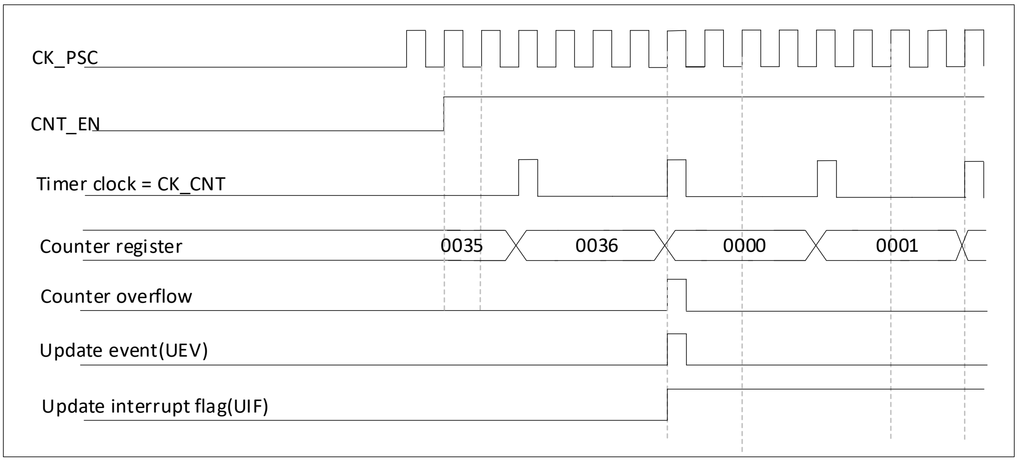


图 22-6 计数器时序图，内部时钟分频因子为 4

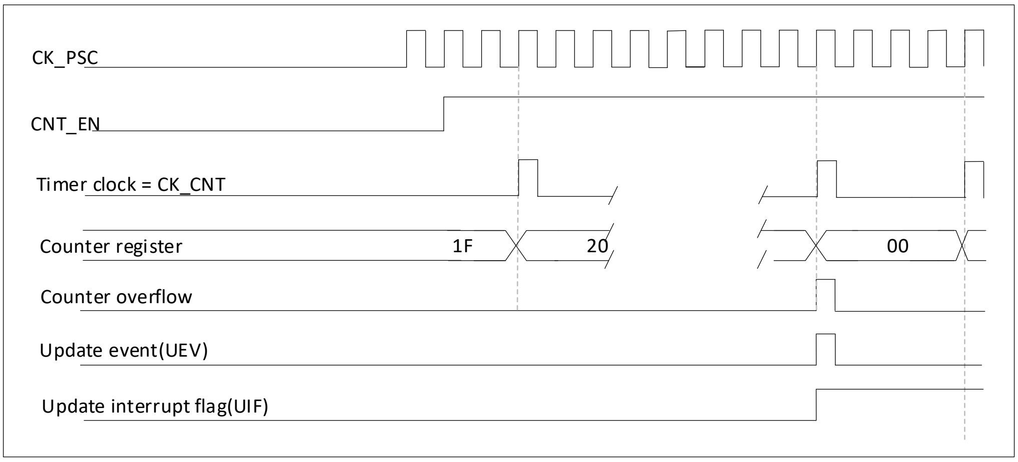


图 22-7 计数器时序图，内部时钟分频因子为 N

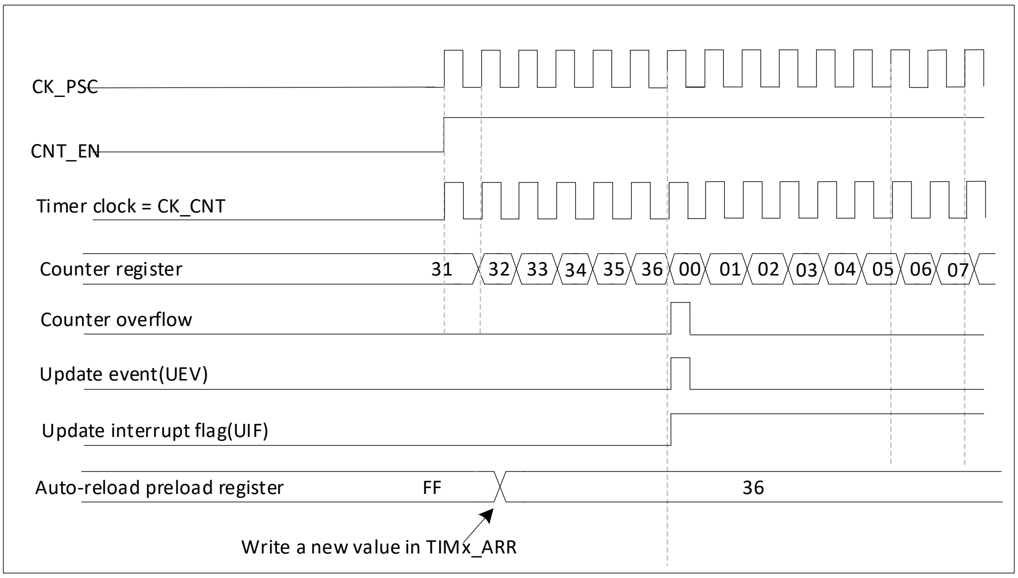


图 22-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

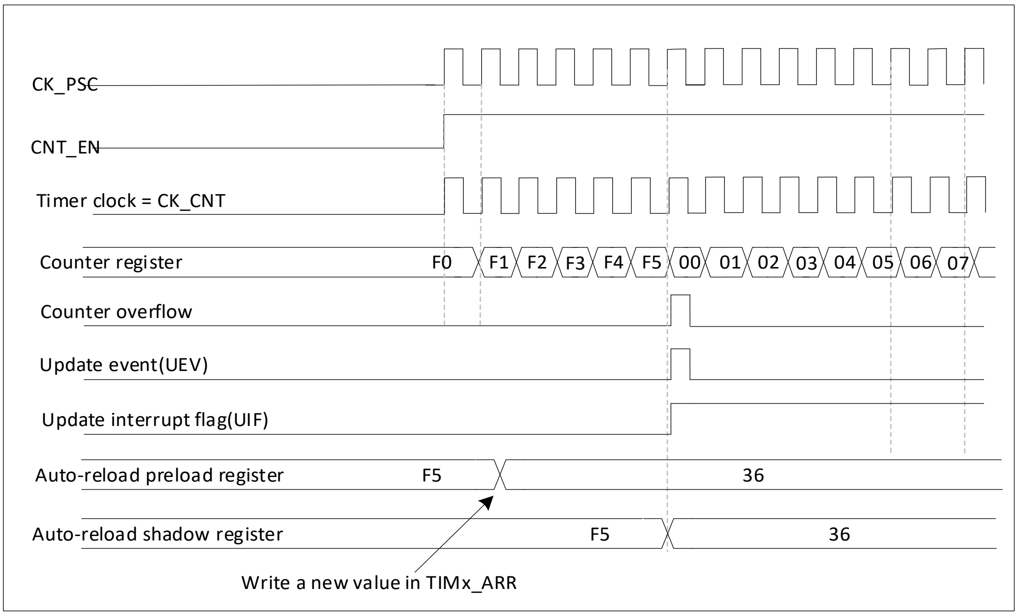


图 22-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

22.3.2. 时钟源

计数器的时钟由内部时钟（CK_INT）提供。TIMx_CR1 寄存器的 CEN 位和 TIM14_EGR 寄存器的 UG 位是实际的控制位（除了 UG 位被自动清除外），只能通过软件改变他们。一旦置 CEN 位为 1，内部时钟即向分频器提供时钟。

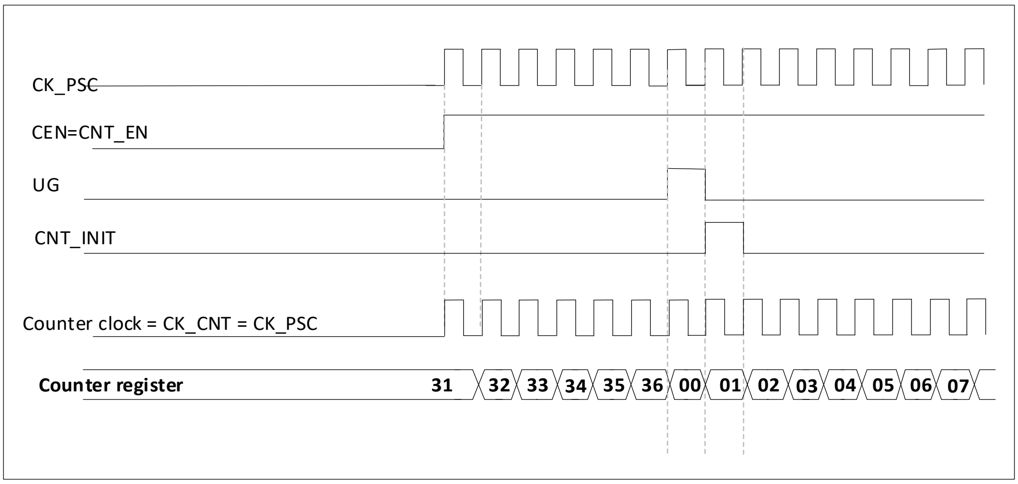


图 22-10 一般模式下的控制电路，内部时钟分频因子为 1

22.3.3. 捕获/比较通道

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

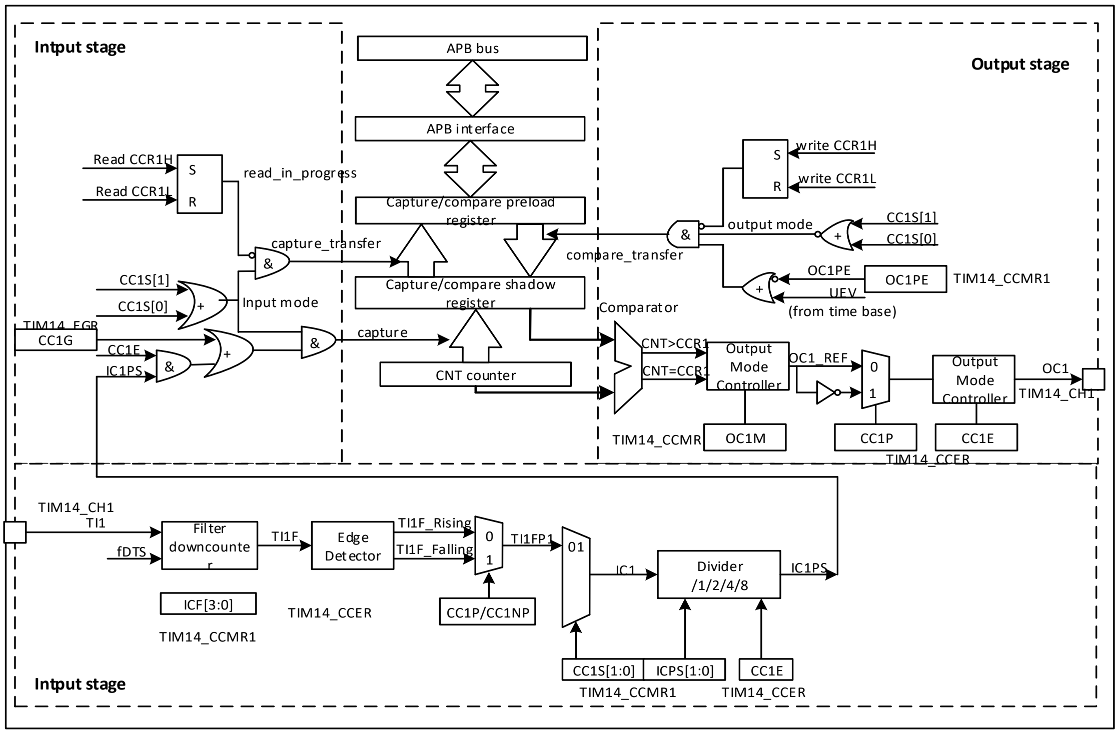


图 22-11 TIM14 捕获/比较通道图

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

输出部分产生一个中间波形（高有效）作为基准，链的末端决定最终输出信号的极性。

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。

22.3.4. 输入捕获模式

在输入捕获模式下，当检测到 Icx 信号相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当捕获事件发生时，相应的 CcxIF 标志（TIM14_SR 寄存器）被置 1。如果捕获事件发生时，CcxIF 标志已经为高，那么重复捕获标志 CcxOF（TIMx_SR 寄存器）被置 1。写 CcxIF 可清除 CcxIF，或读取存储中的捕获数据也可清除 CcxIF。写 CcxOF=0 可清除 CcxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIM14_CCR1 寄存器中，步骤如下:

- 选择有效输出端：TIM14_CCR1 必须连接到 TI1 输入，所以写入 TIM14_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为 00 时，通道被配置为输入，并且 TIM14_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的宽度（即输入为 Tix 时，输入滤波器控制位是 TIM14_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，我们必须配置滤波器的宽度长于 5 个时钟周期。因此，我们可(以 fDTS 频率)连续采样 8 次，已确认在 TI1 上一次真是的边沿变化，然后再 TIM14_CCMR1 寄存器中写入 IC1F=0011.
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0（上升沿）（和 CC1NP=0）

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

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

如果需要，通过设置 TIMx_DIER 寄存器中的 CC1E 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 TIM14_CCR1 寄存器。

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

■ 如设置了 CC1IE 位，则会产生一个中断请求。

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

注：输入捕获中断请求能通过软件设置在 TIMx_EGR 中相应的 CCxG 位来产生。

22.3.5. 强置输出模式

在该模式下（TIM14_CCMRx 寄存器中 CCxS bits = 00）下，输出比较信号（OCxREF 和相应的 OCx）能够直接由软件置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

写 TIM14_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 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性位相反的值。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

置 TIM14_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIM14_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。这将会在下面的输出比较模式一节中介绍。

22.3.6. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较工作做如下操作：

■ 将输出比较模式(TIM14_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)。

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

■ 若设置了相应的中断屏蔽(TIM14_DIER 寄存器中的 CcxIE 位)，则产生一个中断。

可通过配置 TIM14_CCMRx 中的 OCxPE 位选择 TIM14_CCRx 寄存器是否需要使用预装载寄存器。在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

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。

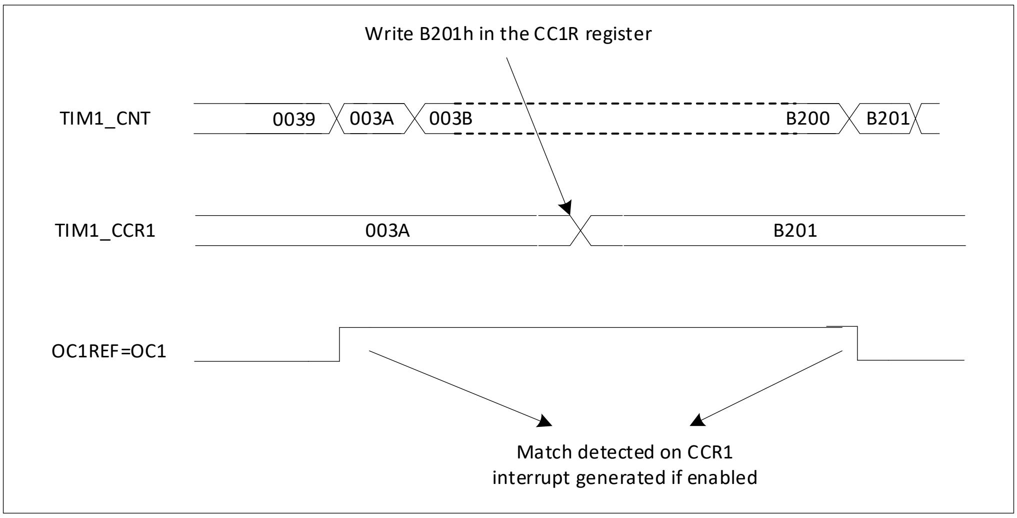


图 22-12 输出比较模式，翻转 OC1

22.3.7. 脉冲宽度调节（PWM）模式

脉冲宽度调节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 TIM14_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。必须设置 TIM14_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 TIM14_CR1 寄存器中的 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 TIM14_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 TIM14_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或者低电平有效。TIM14_CCER 寄存器中的 CcxE 位控制 OCx 输出使能。

在 PWM 模式（模式 1 或者模式 2），TIM14_CNT 和 TIM14_CCRx 始终在进行比较，以确定是否符合 TIM14_CNT ≤ TIM14_CCRx。

定时器能够产生边沿对齐模式的 PWM 信号。

PWM 边沿对齐模式

下面是一个 PWM 模式 1 的例子。当 TIM14_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM14_CCRx 中的比较值大于自动重装载值(TIM14_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。

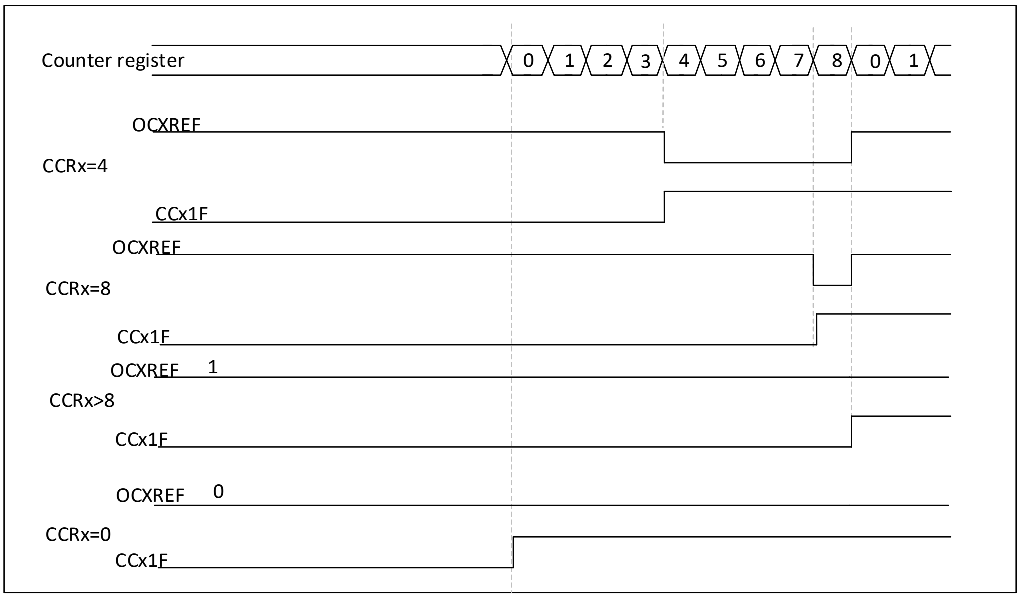


图 22-13 边沿对齐的 PWM 波形(ARR=8)

22.3.8. 单脉冲模式

单脉冲模式(OPM)是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。

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

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

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

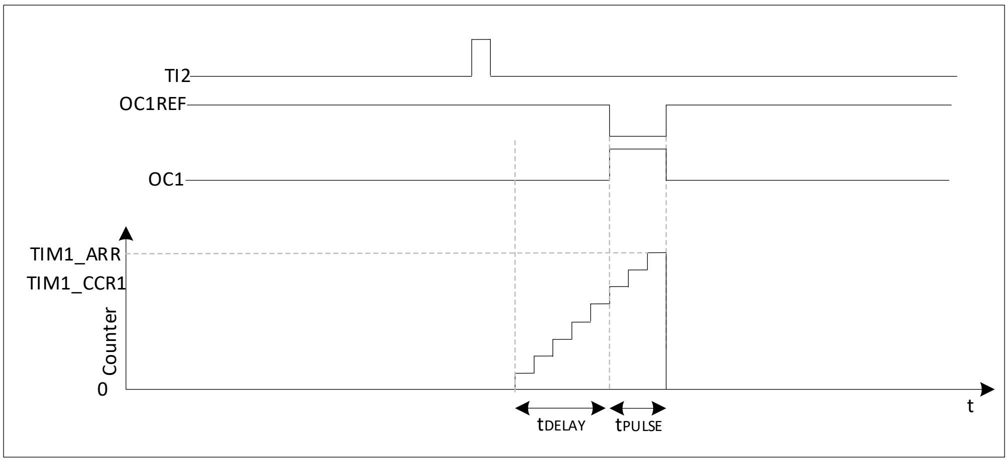


图 19-14 单脉冲模式的例子

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

假定 TI2FP2 作为触发:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110， TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)， TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

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

在这个例子中， TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 OPM=0 时，重复模式被选中。

22.3.9. 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。TIM10/11/13/14 作为主定时器输出 oc 给相应配置下的从定时器 TIM9/12。

22.3.10. 调试模式

当芯片进入调试模式（M0+停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器或者继续正常操作，或者停止。

22.4. TIM14 寄存器

22.4.1. TIM14 控制寄存器 1 (TIM14_CR1)

Address offset:0x00

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	0	保留，一直读为 0
9:8	CKD[1:0]	RW	00	时钟分频因子，这 2 位定义在定时器时钟(CK_INT)频率，所用的采样时钟之间的分频比例 00： tDTS = tCK_INT 01： tDTS = 2 x tCK_INT

Bit	Name	R/W	Reset Value	Function
				10： tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： TIM14_ARR 寄存器没有缓冲 1： TIM14_ARR 寄存器被装入缓冲器
6:4	Reserved	-	0	保留，一直读为 0
3	OPM	RW	0	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： − 计数器溢出/下溢 − 设置 UG 位 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中断或 DMA 请求
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： − 计数器溢出/下溢 − 设置 UG 位 被缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被 重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

22.4.2. TIM14 DMA/中断使能寄存器 (TIM14_DIER)

Address offset:0x0C

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														**CC1IE**	**UIE**
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31：2	Reserved			保留，一直为 0

Bit	Name	R/W	Reset Value	Function
1	CC1IE	RW	0	CC1IE：允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断 1：允许更新中断

22.4.3. TIM14 状态寄存器(TIM14_SR)

Address offset:0x010

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	**IC1IF**	Res	Res	Res	**IC1IR**
-	-	-	-	-	-	-	-	-	-	-	RW	-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res						**CC1OF**	**Res**							**CC1F**	**UIF**
-						Rc_w0	-							Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-		保留，一直为 0
20	IC1IF	RW	0	
19:17	Reserved	-		保留，一直为 0
16	IC1IR	RW	0	
15:10	Reserved	-		保留，一直为 0
9	CC1OF	Rc_w0	0	捕获/比较 1 过捕获标记 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：无过捕获产生； 1：CC1IF 置 1 时，计数器的值已经被捕获到 TIM14_CCR1 寄存器。
8:2	Res	Rc_w0	0	保留，始终读为 0。
1	CC1IF	Rc_w0	0	捕获/比较 1 中断标记 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配后一个时钟周期时，该位由硬件置 1，它由软件清 0。 0：无匹配发生； 1：TIM14_CNT 的值与 TIM14_CCR1 的值匹配。 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置 1，它由软件清 0 或通过读 TIM14_CCR1 清 0。 0：无输入捕获产生； 1：输入捕获产生并且计数器值已装入 TIM14_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	Rc_w0	0	更新中断标记，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：

Bit	Name	R/W	Reset Value	Function
				– 若 TIMx_CR1 寄存器的 UDIS=0，产生更新事件上溢； – 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

22.4.4. TIM14 事件产生寄存器(TIM14_EGR)

Address offset:0x14

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														CC1G	UG
-														W	W

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	0	保留，一直为 0
1	CC1G	W	0	产生捕获/比较 1 事件，该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断请求。 若通道 CC1 配置为输入： 当前的计数器值捕获至 TIM14_CCR1 寄存器，设置 CC1IF=1，若开启对应的中断，则产生相应的中断请求。 若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件，该位由软件置 1，由硬件自动清 0。 0：无动作； 1：重新初始化计数器，并产生一个寄存器的更新事件。注意预分频器的计数器也被清 0(但是预分频系数不变)。

22.4.5. TIM14 捕获/比较模式寄存器 1(TIM14_CCMR1)

Address offset:0x18

Reset value:0x0000 0000

输出比较模式:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								Res	OC1M[2:0]			OC1PE	Res	CC1S[1:0]	
-								-	RW	RW	RW	RW	-	RW	RW

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-		保留，一直为 0
6:4	OC1M[2:0]	RW	00	输出比较 1 模式

Bit	Name	R/W	Reset Value	Function
				该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 TIM1_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 一旦 TIMx_CNT<TIMx_CCR1 时，通道 1 为无效电平，否则为有效电平。 注：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIM14_CCR1 寄存器的预装载功能，可随时写入 TIM14_CCR1 寄存器，且新值马上起作用。 1：开启 TIM14_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM14_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	Reserved	-	0	保留，一直为 0
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：Reserved； 11：Reserved。 注：CC1S 仅在通道关闭时(TIM14_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res								IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
-								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-		保留，一直为 0
7:4	IC1F[3:0]	RW	0000	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， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8，N=6 0001：采样频率 fSAMPLING=fCK_INT， N=2 1001：采样频率 fSAMPLING=fDTS/8， N=8 0010：采样频率 fSAMPLING=fCK_INT， N=4 1010：采样频率 fSAMPLING=fDTS/16， N=5 0011：采样频率 fSAMPLING=fCK_INT， N=8 1011：采样频率 fSAMPLING=fDTS/16， N=6 0100：采样频率 fSAMPLING=fDTS/2， N=6 1100：采样频率 fSAMPLING=fDTS/16， N=8 0101：采样频率 fSAMPLING=fDTS/2， N=8 1101：采样频率 fSAMPLING=fDTS/32， N=5 0110：采样频率 fSAMPLING=fDTS/4， N=6 1110：采样频率 fSAMPLING=fDTS/32， N=6 0111：采样频率 fSAMPLING=fDTS/4， N=8 1111：采样频率 fSAMPLING=fDTS/32， N=8
3:2	IC1PSC[1:0]	RW	00	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S[1:0]	RW	00	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：Reserved 11：Reserved 注： CC1S 仅在通道关闭时(TIM14_CCER 寄存器的 CC1E=0)才是可写的。

22.4.6. TIM14 捕获/比较使能寄存器(TIM14_CCER)

Address offset:0x20

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31：4	Reserved	-	0	保留，一直为 0
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 CC1 通道配置成输出： CC1NP 必须保持 0. CC1 通道配置成输入： CC1NP 和 CC1P 联合使用来定义 TI1FP1 极性（参考 CC1P 描述）
2	Reserved	-	0	保留，一直为 0
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0： OC1 高电平有效； 1： OC1 低电平有效。 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获模式下触发）； 01：反相/下降沿： TIxFP1 下降沿有效（捕获模式下触发）； 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获模式下触发）；
0	CC1E	RW	0	输入/捕获 1 输出使能 **CC1 通道配置为输出：** 0：关闭 - OC1 禁止输出 1：开启 - OC1 信号输出到对应的输出引脚 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 **1**：捕获使能

CcxE 位	OCx output State
0	输出禁止（OCx=0,OCx_EN=0）
1	OCx=OCxREF+Polarity,OCx_EN=1

22.4.7. TIM14 计数器(TIM14_CNT)

Address offset:0x24

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CNT[15:0]	RW	0	计数器的值

22.4.8. TIM14 预分频器(TIM14_PSC)

Address offset:0x28

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	PSC[15:0]	RW	0	预分频器的值 计数器的时钟频率（CK_CNT）等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器 被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

22.4.9. TIM14 自动重装载寄存器 (TIM14_ARR)

Address offset:0x2c

Reset value:0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	ARR[15:0]	RW	FFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 12.4.1：时基单元有关 ARR 的更新和动作。

Bit	Name	R/W	Reset Value	Function
				当自动重装载的值为空时，计数器不工作。

22.4.10. TIM14 捕获/比较寄存器 1(TIM14_CCR1)

Address offset:0x34

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
31：16	Reserved			保留，一直为 0
15:0	CCR1[15:0]	RW	0	捕获/比较 1 的值 若 CC1 通道配置为输出: CCR1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIM1_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIMx_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1 通道配置为输入: CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

22.4.11. TIM14 选项寄存器(TIMx_OR)

Address offset:0x50

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res															
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res														TI1_RMP	
-														RW	RW

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	0	保留，一直读为 0。
1：0	TI1_RMP	RW	0	定时器输入 1 重映射 通过软件置位和清零。 00:TIM14 通道 1 连接到 GPIO,具体参考数据手册的复用功能。 01：TIM14 通道 1 连接到 RTCCLK. 10：TIM14 通道 1 连接到 HSE/32 时钟 11： TIM14 通道 1 连接到 MCU 时钟输出（MCO）.这个配置是通过 RCC_CFG 寄存器的 MCO[2:0]的设置来决定的。

22.4.12. TIM14 寄存器映像

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	TIM14_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.		Res.	OPM.	URS	UDIS	CEN
	Reset value																							0	0	0				0	0	0	0
0x0C	TIM14_DIER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IE	UIE
	Reset value																															0	0
0x10	TIM14_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1IF	Res.	Res.	IC1IR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1O	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1IF	UIF
	Reset value												0			0								0								0	0
0x14	TIM14_EGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1G	UG
	Reset value																															0	0
0x18	TIM14_CCMR1(output compare mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M [2:0]			OC1PE	Res.	CC1S[1:0]	
	Reset value																										0	0	0	0		0	0
0x18	TIM14_CCMR1(Input Capture mode)	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
	Reset value																									0	0	0	0	0	0	0	0
0x20	TIM14_CCER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	Res.	CC1P	CC1E
	Reset value																													0		0	0
0x24	TIM14_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TIM14_PSC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PSC[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TIM14_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x34	TIM14_CCR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CCR1[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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
0x50	TIMx_OR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1_RMP[1:0]	
	Reset value																															0	0

23.通用定时器 (TIM15/16/17)

23.1. 简介

通用 timer（TIM15_16_17）由 16 位被可编程分频器驱动的自动装载计数器组成。它可以被用作各种场景，包括：输入信号（输入捕获）的脉冲长度测量，或者产生输出波形（输出比较、输出 PWM、带死区插入的互补 PWM）。

脉冲长度和波形周期可以使用定时器分频器和 RCC 时钟控制分频器，从微秒到毫秒的调制。通用 timer（TIM15_16_17）和通用（TIMx）timer 是完全独立的，不共享任何资源。他们可以同步起来。

23.2. TIM15 主要特性

- 16bit 向上、向下或者向上向下的自动重装载计数器
- 16bit 可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频（on the fly）
- 多达 2 个独立的通道
  - 输入捕获
  - 输出比较
  - PWM 产生（边缘对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出(只有通道 1)
- 使用外部信号控制定时器和定时器互连的同步电路
- 重复计数器，在计数指定周期数后，才更新时间寄存器
- 刹车输入可以将定时器的输出信号置为复位状态或者已知状态
- 中断/DMA 产生在以下事件
  - 更新：计数器向上、向下溢出，计数器初始化（通过软件或者内外部触发）
  - 触发事件
  - 输入捕获
  - 输出比较
  - 刹车输入

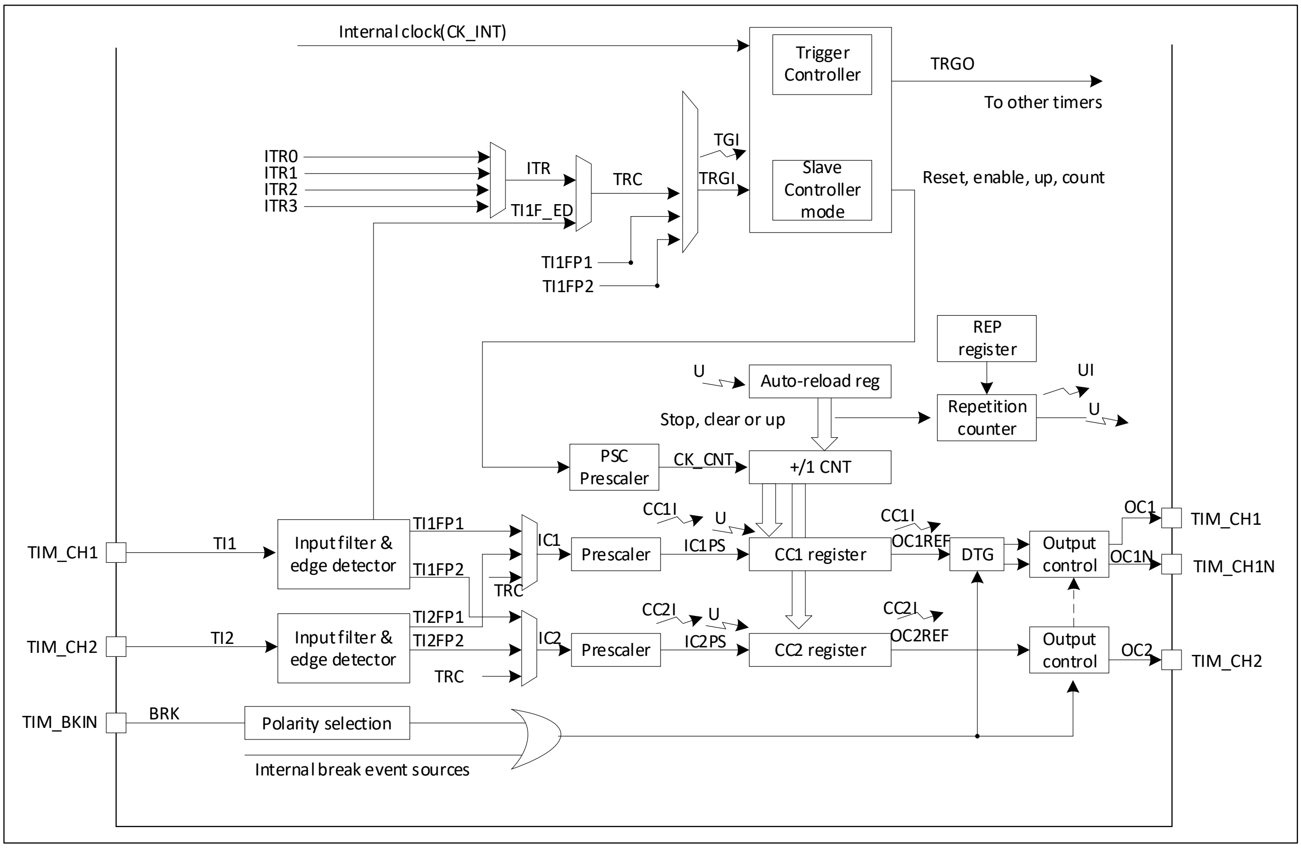


图 23-1 通用控制定时器(TMI15)架构框图

23.3. TIM16_17 主要特性

- 16bit 向上、向下或者向上向下的自动重装载计数器
- 16bit 可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频（on the fly）
- 1 个独立的通道
  - 输入捕获
  - 输出比较
  - PWM 产生（边缘对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 重复计数器，在计数指定周期数后，才更新时间寄存器
- 刹车输入可以将定时器的输出信号置为复位状态或者已知状态
- 中断/DMA 产生在以下事件
  - 更新：计数器向上、向下溢出
  - 输入捕获
  - 输出比较
  - 刹车输入

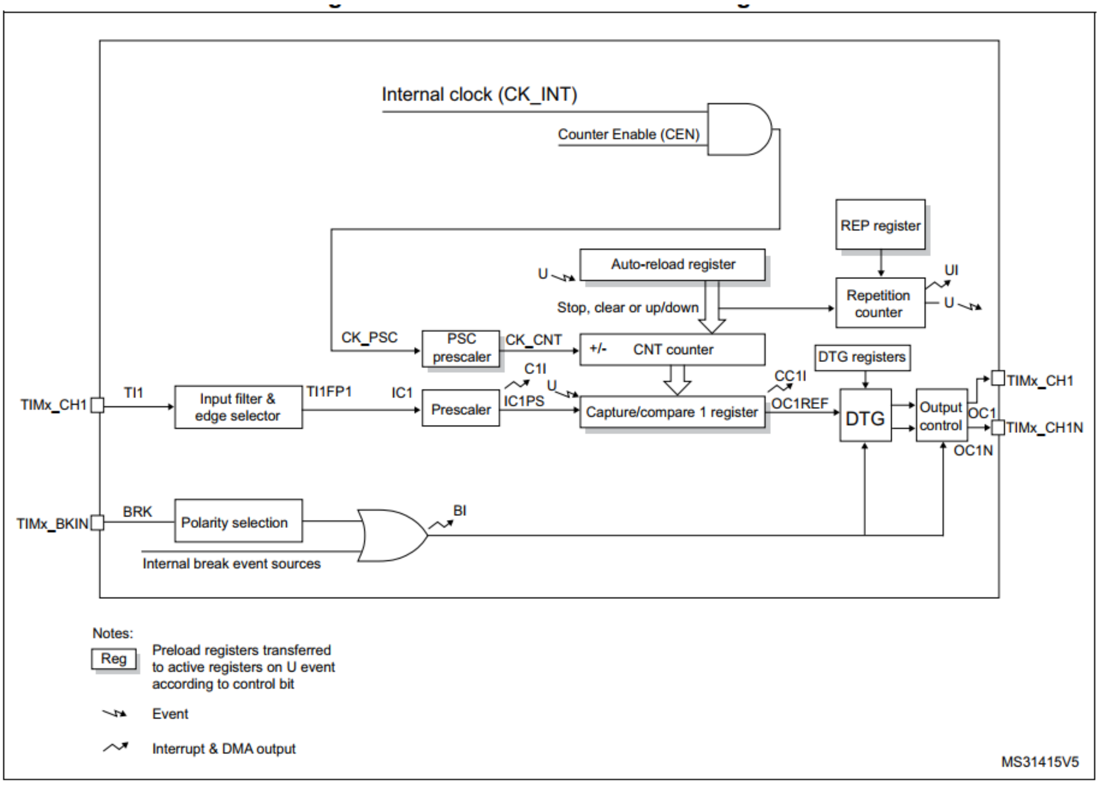


图 23-2 通用控制定时器(TMI16_17)架构框图

23.4. TIM15_16_17 功能描述

23.4.1. 时基单元

可编程通用控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

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

时基单元包括：

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

自动装载寄存器是预先装载的，写或者读自动重载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到上溢出（向下计数器时的下溢条件）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。

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_CR 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按 1 到 65535 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 xx 和图 xx 给出了在预分频器运行时，更改计数器参数的例子。

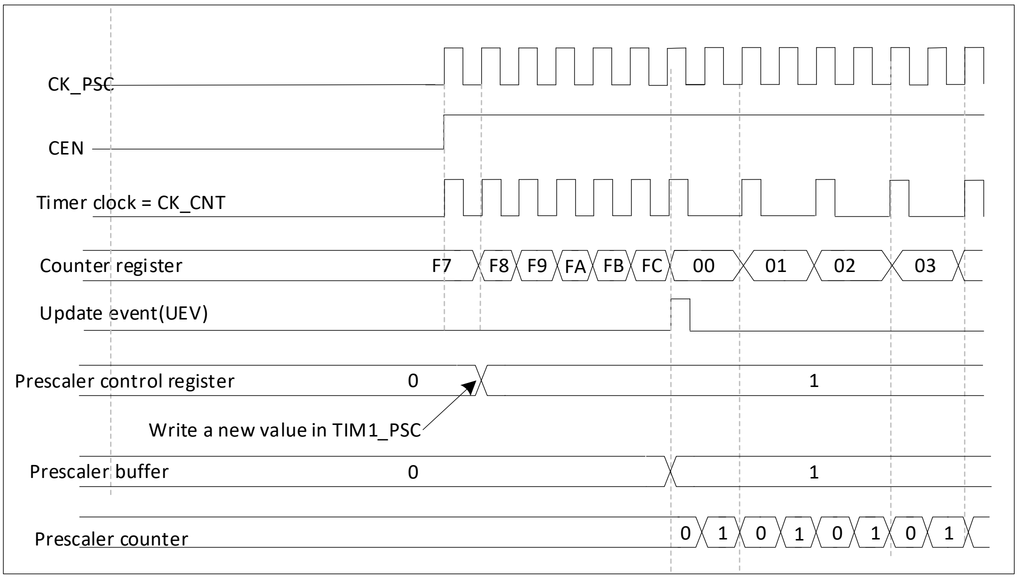


图 23-3 当预分频器的参数从 1 变到 2 时，计数器的时序图

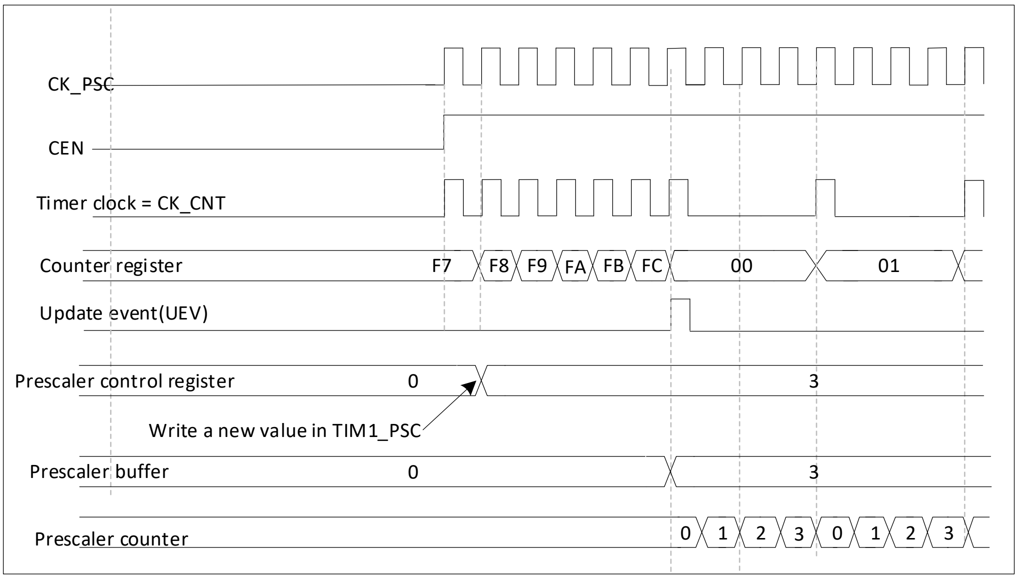


图 23-4 当预分频器的参数从 1 变到 4 时，计数器的时序图

23.4.2. 计数器模式

向上计数模式

向上计数模式，是从 0 到自动装载值的计数器，然后又从 0 重新开始计数，并产生一个计数的溢出事件。

如果重复计数器被使用，则在上溢次数达到所配置的重复计数寄存器的值加一（即 TIMx_RCR+1）后，产生更新事件。否则，在每个计数溢出时，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。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样也可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清零之前，将不产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 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 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

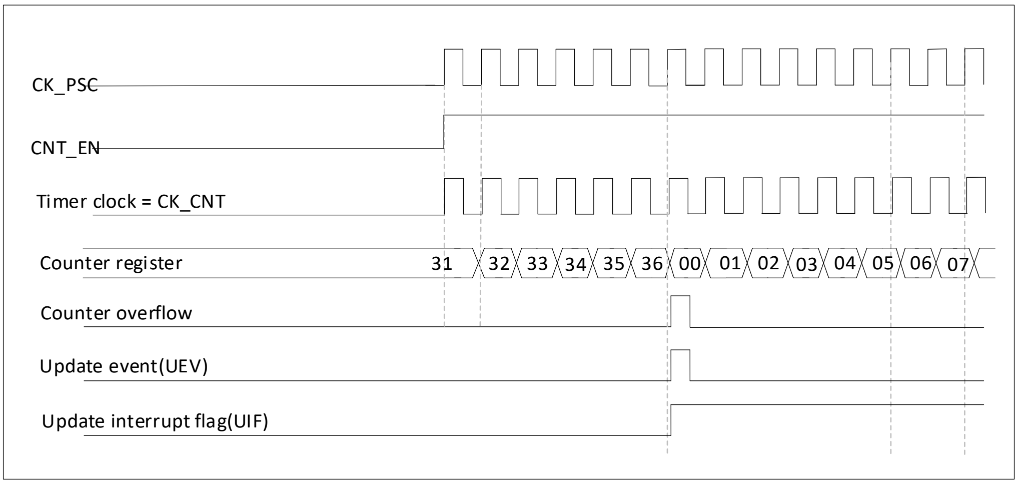


图 23-5 计数器时序图，内部时钟分频因子为 1

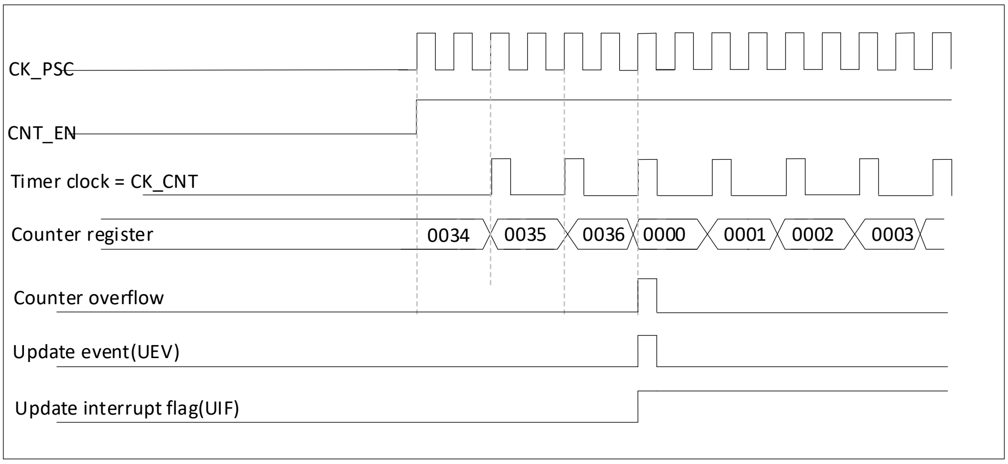


图 23-6 计数器时序图，内部时钟分频因子为 2

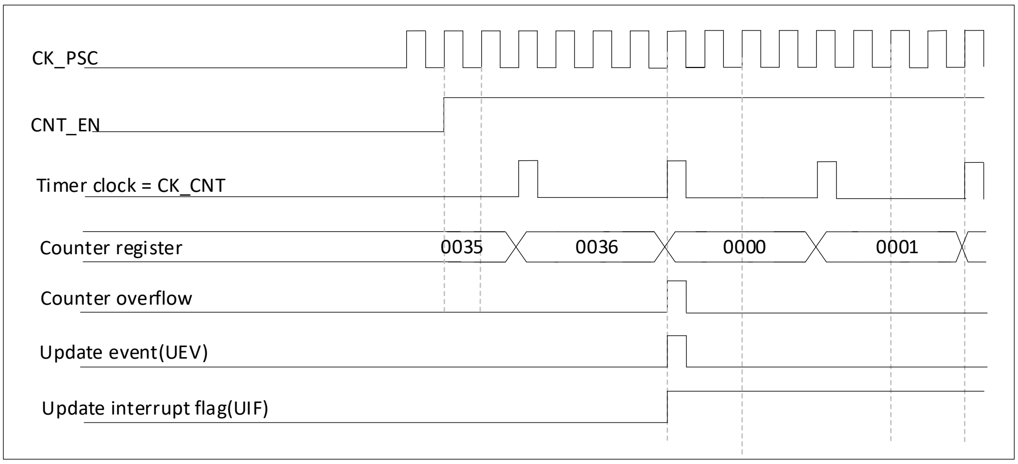


图 23-7 计数器时序图，内部时钟分频因子为 4

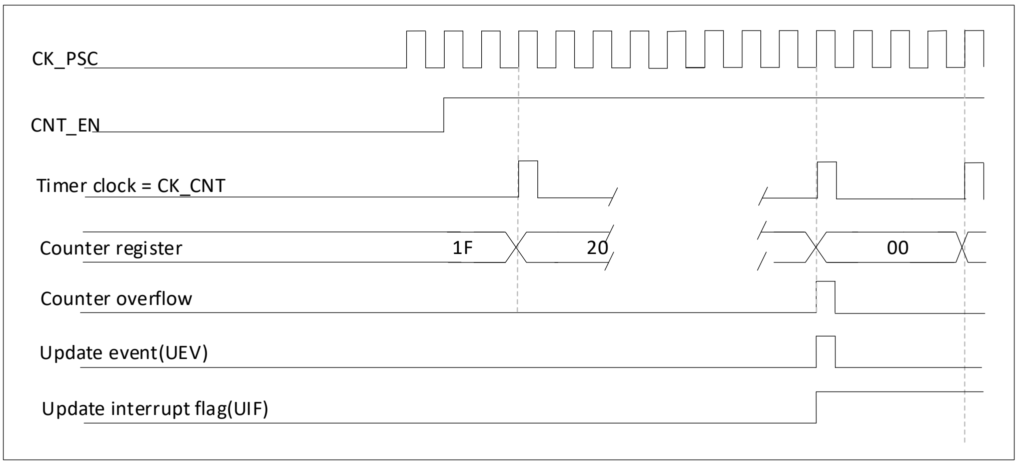


图 23-8 计数器时序图，内部时钟分频因子为 N

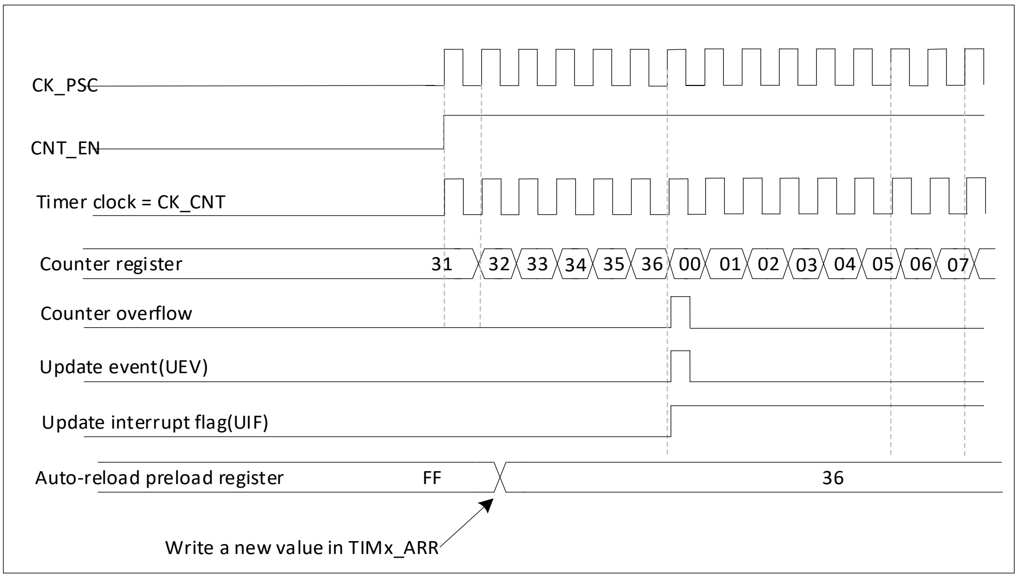


图 23-9 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

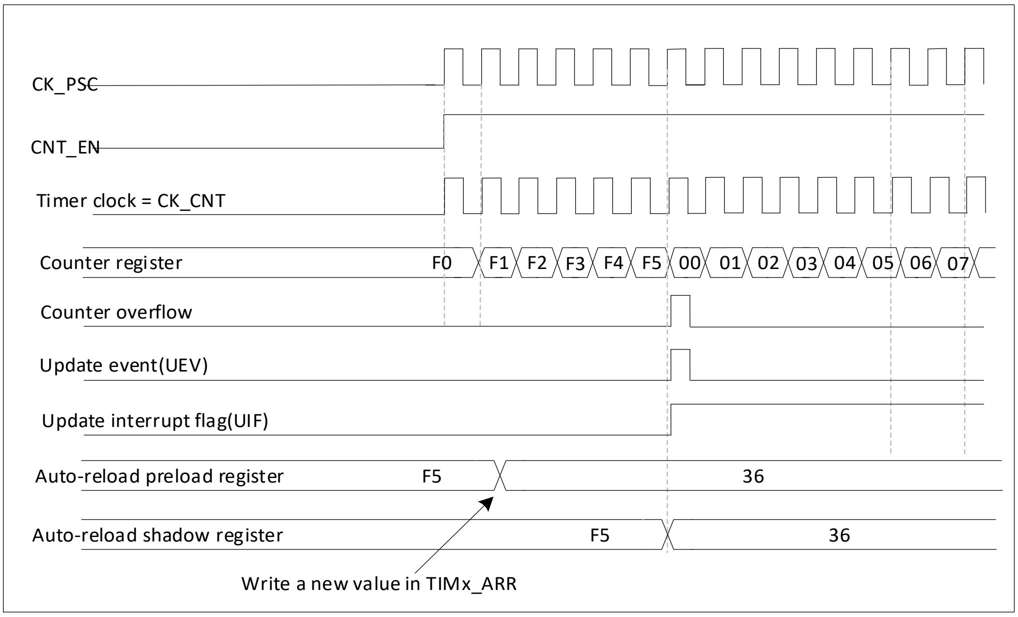


图 23-10 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

向下计数模式

向下计数模式，从自动装载的值开始向下计数到 0，然后重新开始从自动装载的值向下计数，并产生一个向下溢出事件。

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

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。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始(但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)， 设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

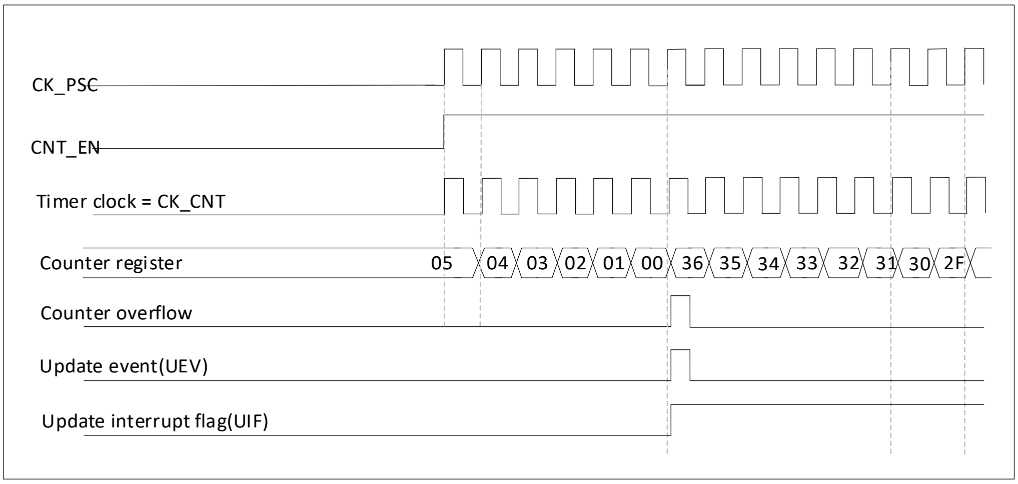


图 23-11 计数器时序图，内部时钟分频因子为 1

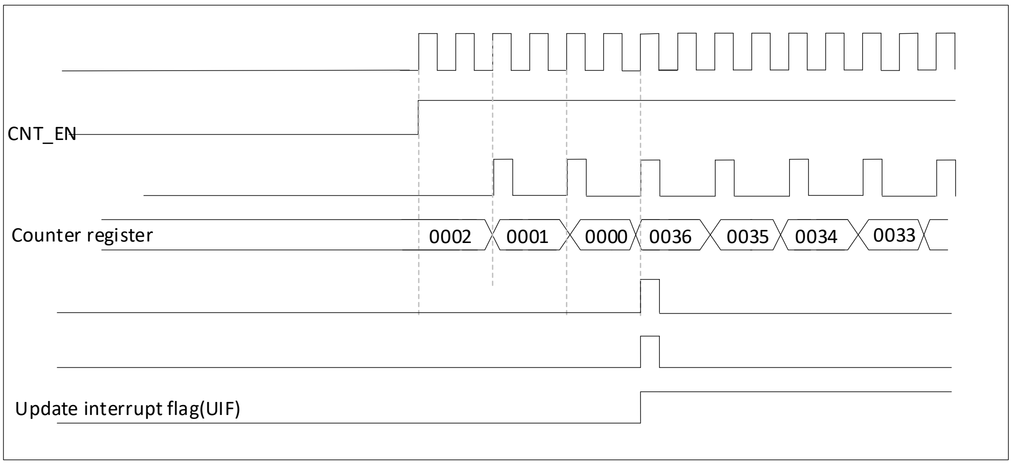


图 23-12 计数器时序图，内部时钟分频因子为 2

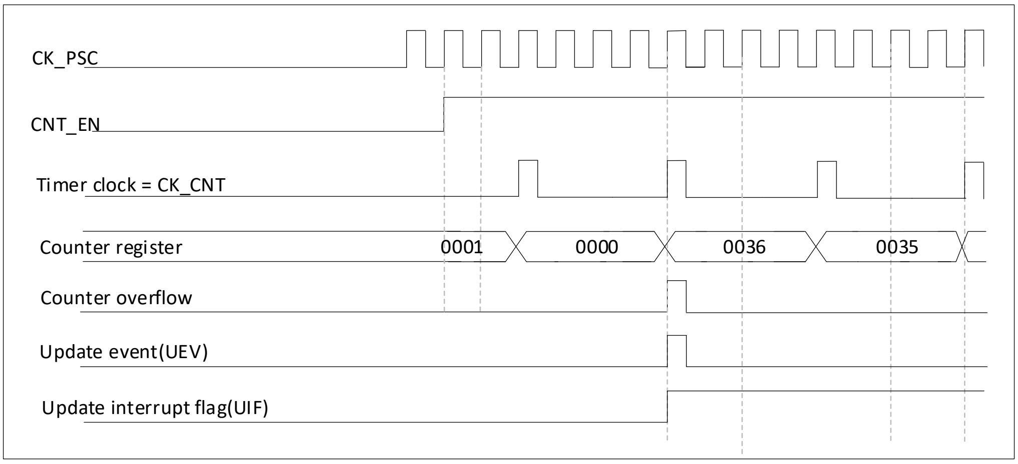


图 23-13 计数器时序图，内部时钟分频因子为 4

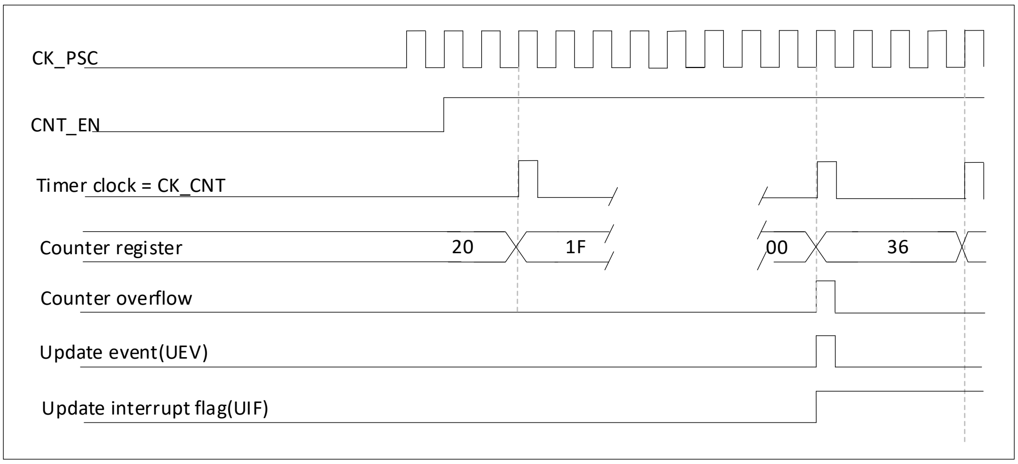


图 23-14 计数器时序图，内部时钟分频因子为 N

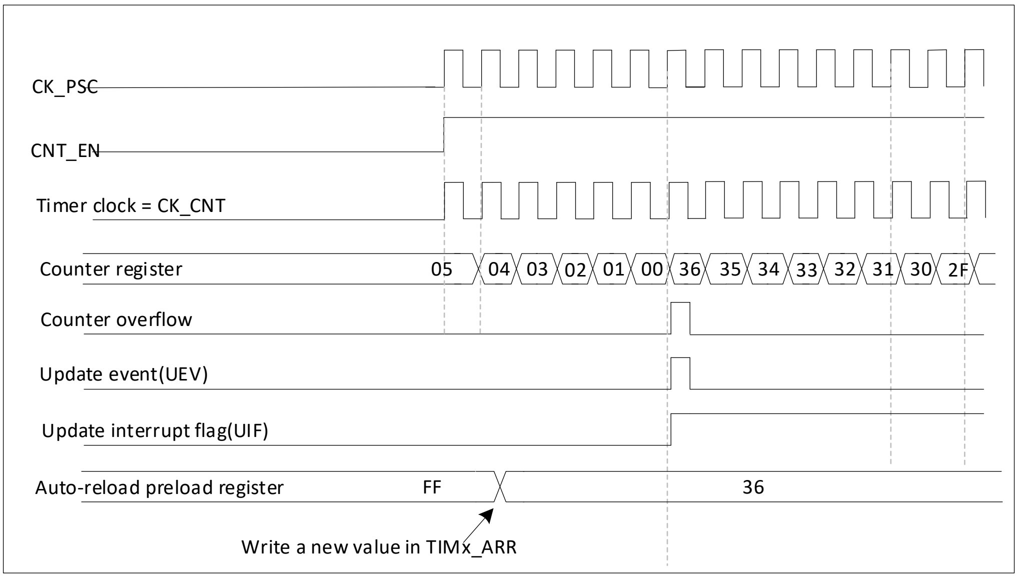


图 23-15 计数器时序图，当没有使用周期计数器时的更新事件

中央对齐模式 (向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器溢出事件，然后向下计数到 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。

中央对齐模式在 TIMx_CR1 寄存器中的 CMS 不等于 0 时有效。通道在配置成输出模式时，输出比较中断标志将被置位，当：向下计数时（中央对齐模式 1，CMS=”01”）,向上计数时（中央对齐模式 2，CMS=”10”）向上向下计数（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。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内
  注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)

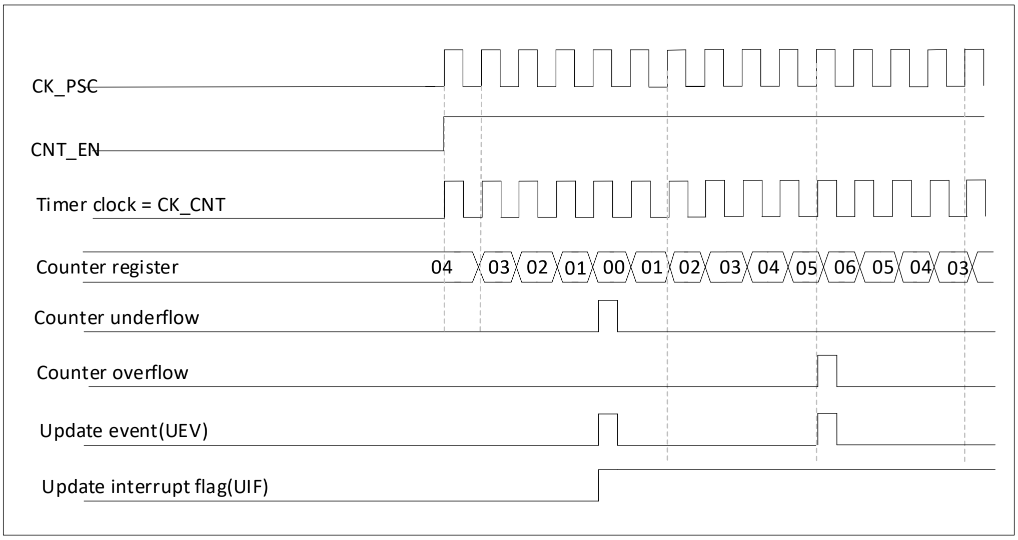


图 23-16 计数器时序图，内部时钟分频因子为 1，TIMx_ARR = 0x6

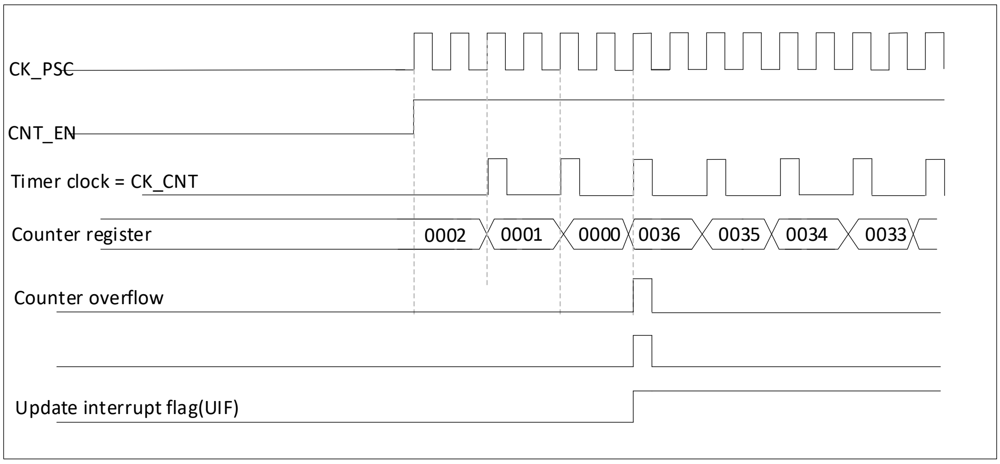


图 23-17 计数器时序图，内部时钟分频因子为 2, TIMx_ARR=0x36

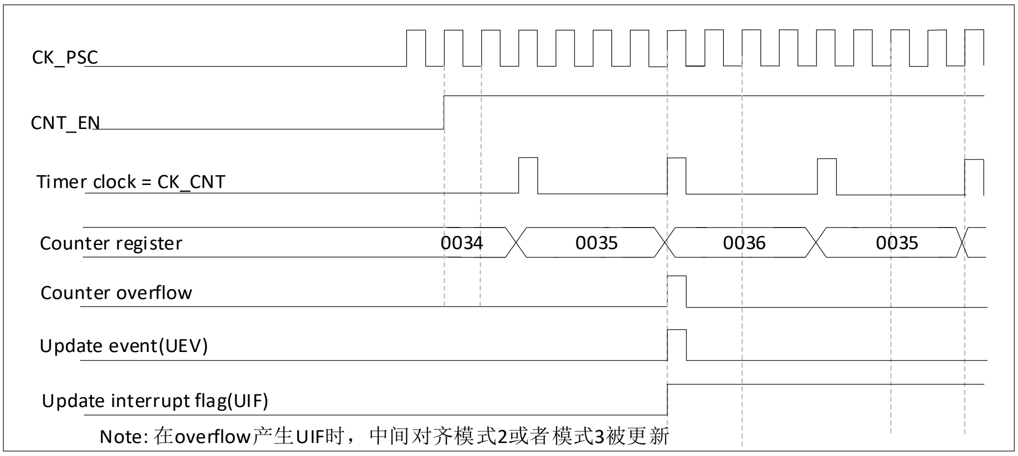


图 23-18 计数器时序图，内部时钟分频因子为 4, TIMx_ARR=0x36

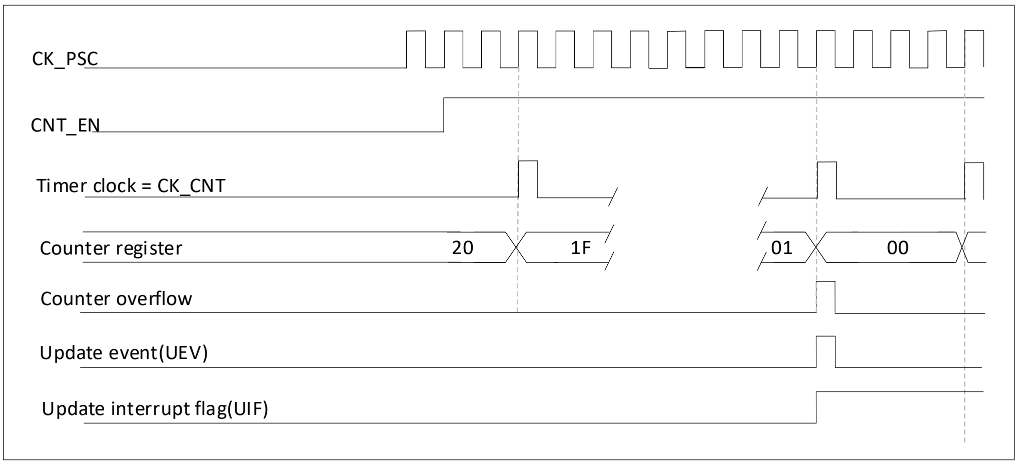


图 23-19 计数器时序图，内部时钟分频因子为 N

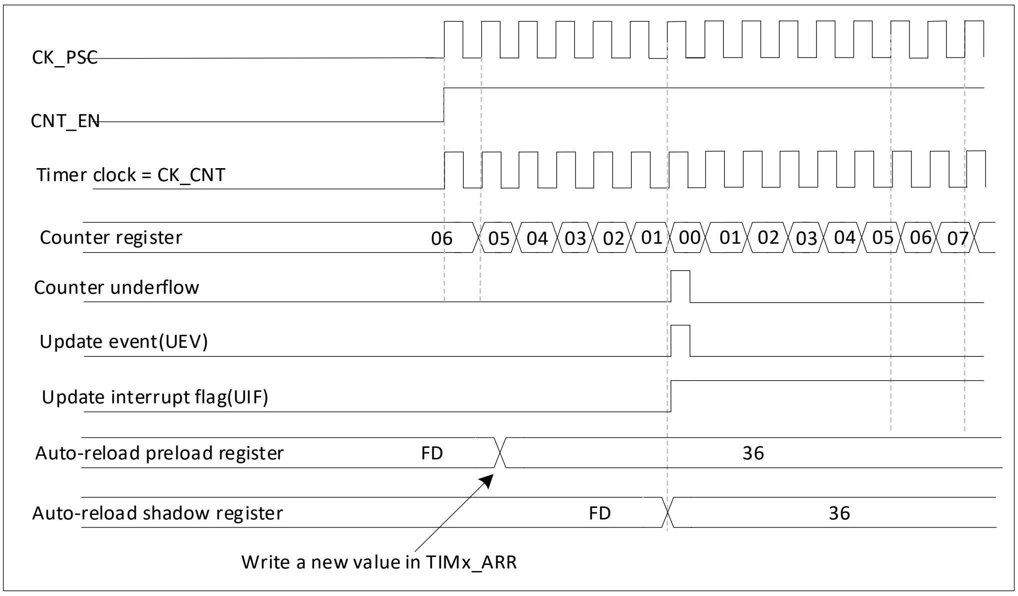


图 23-20 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

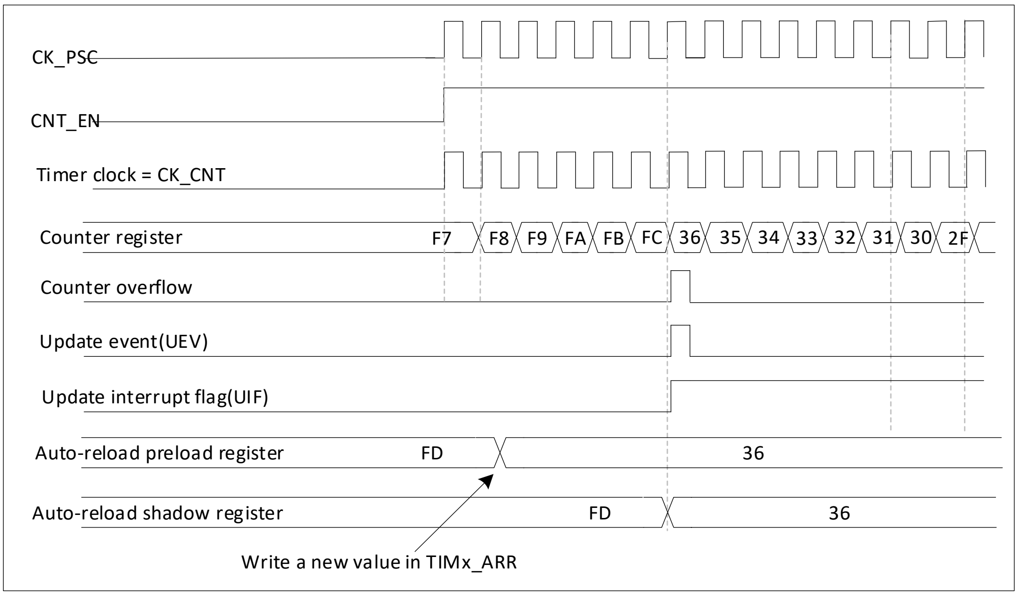


图 23-21 计数器时序图，ARPE=1 时的更新事件(计数器溢出)

23.4.3. 重复向下计数器

时基单元描述了关于计数器向上、向下溢出的更新事件如何产生的。它实际上仅当重复计数器计数到零才产生。这也是当产生 PMW 信号时很有用的。

这意味着在每 N+1(N)次计数上溢或下溢时，数据被从预装载寄存器传送到影子寄存器（TIMx_ARR 自动重载入寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx），N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任何一条件成立时递减：

- 向上计数模式下每次计数上溢时
- 向下计数模式下每次计数下溢时
- 中央对齐模式下每次上溢和每次下溢时。

虽然这样限制了 PWM 的最大循环周期位 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 2xTck。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值定义。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如到重复计数器。

在中央对齐模式下，对于 RCR 的奇数值，取决于当 RCR 寄存器被写入和当计数器开始，出现上溢、或者下溢，则更新事件产生。如果在启动计数器之前写 RCR，在上溢时产生更新事件。例如，对于 RCR = 3 时，更新事件被产生在第 4 个上溢或者下溢事件（取决于 RCR 被写入的值）。

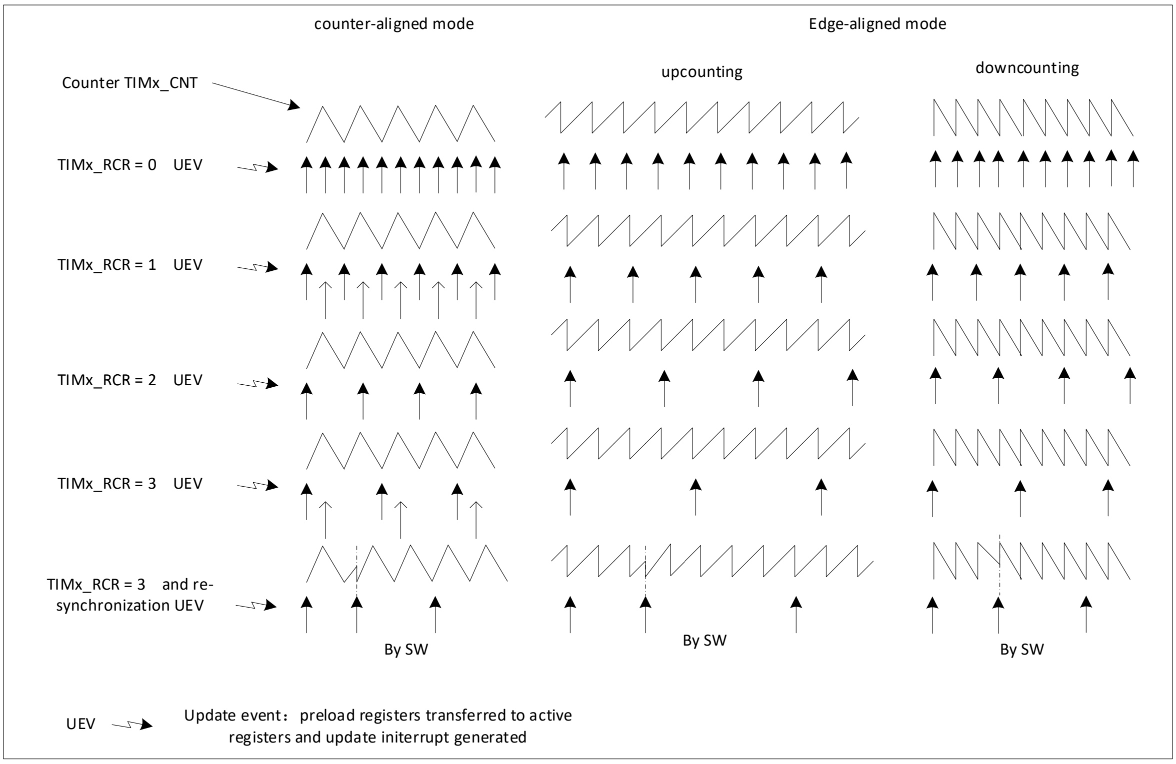


图 23-22 不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置

23.4.4. 时钟源

计数器的时钟可以由以下时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚(仅 TMI15)
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。例如，可以配置一个定时器 Timer1 作为另一个定时器 Timer2 的预分频器。(仅 TMI15)

内部时钟源（CK_INT）

如果从模式控制器被禁止，则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）是事实上的控制位，并且只能被软件修改。只要 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

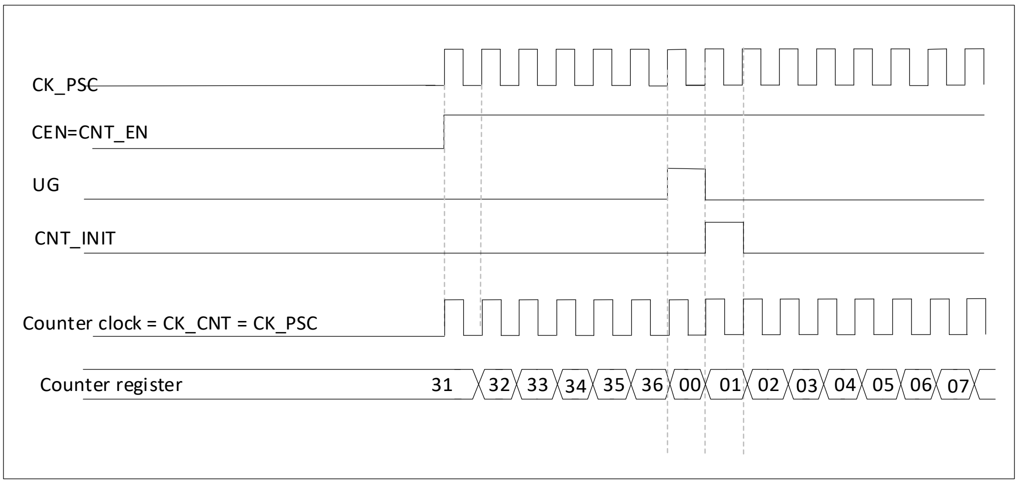


图 23-23 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1（仅 TIM15）

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

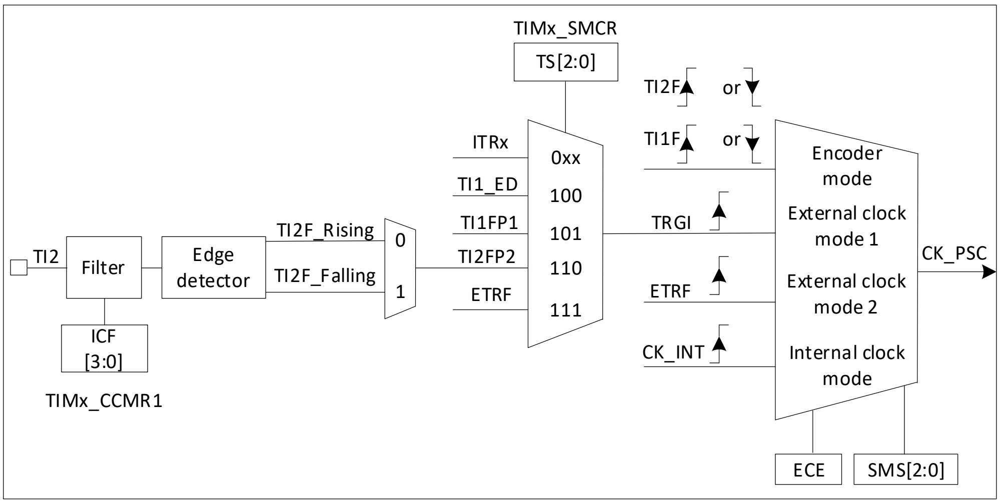


图 23-24 TI2 外部时钟连接例子

例如，要配置计数器在 T12 输入端的上升沿向上计数，使用下列步骤：

1.配置 TIMx_CCMR1 寄存器 CC2S=01，使得通道 2 检测 TI2 输入端的上升沿；

2.配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽 (如果不需要滤波器， 保持 IC2F=0000)；

3.配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；

4.配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器为外部时钟模式 1；

5.配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源；

6.设置 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

在 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。

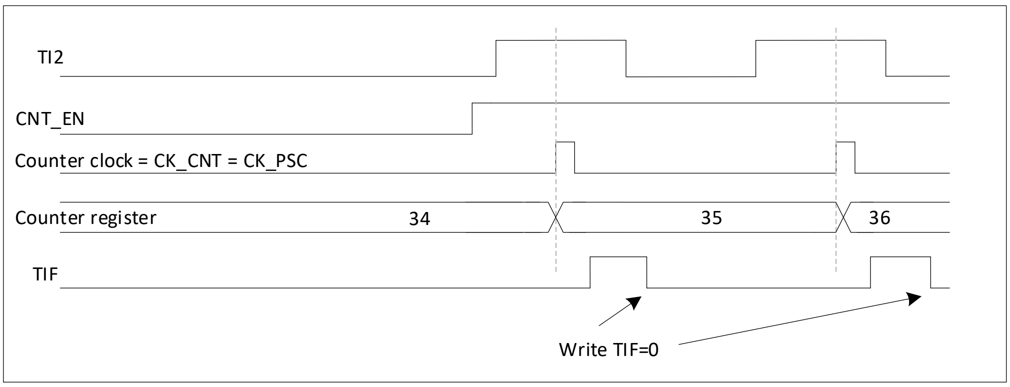


图 23-25 外部时钟模式 1 下的控制电路

23.4.5. 捕获/比较通道

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

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

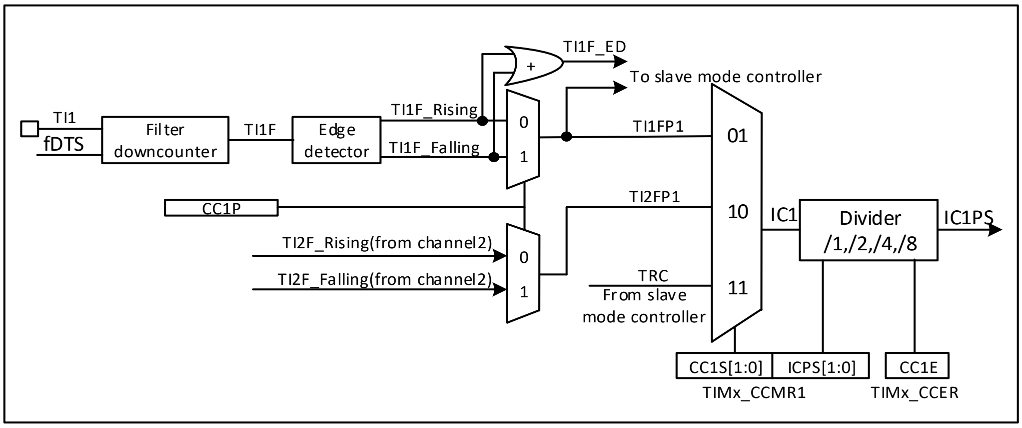


图 23-26 捕获/比较通道(如：通道 1 输入部分)

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。

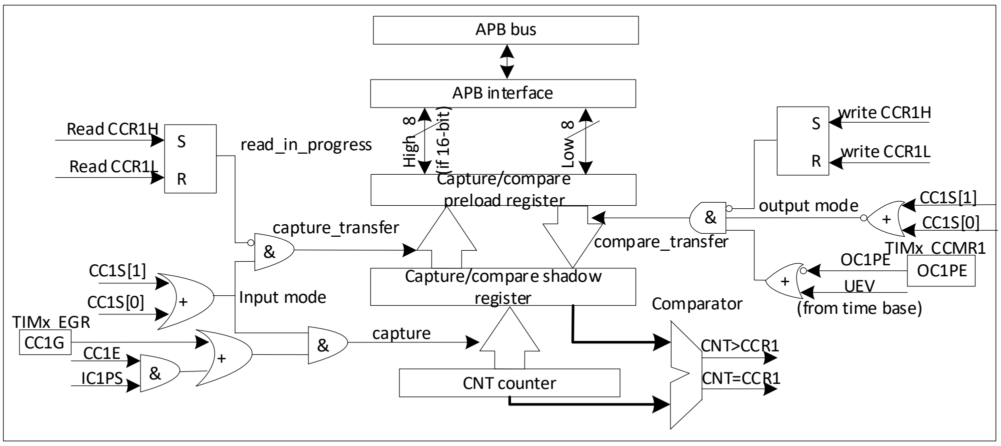


图 23-27 捕获/比较通道 1 的主电路

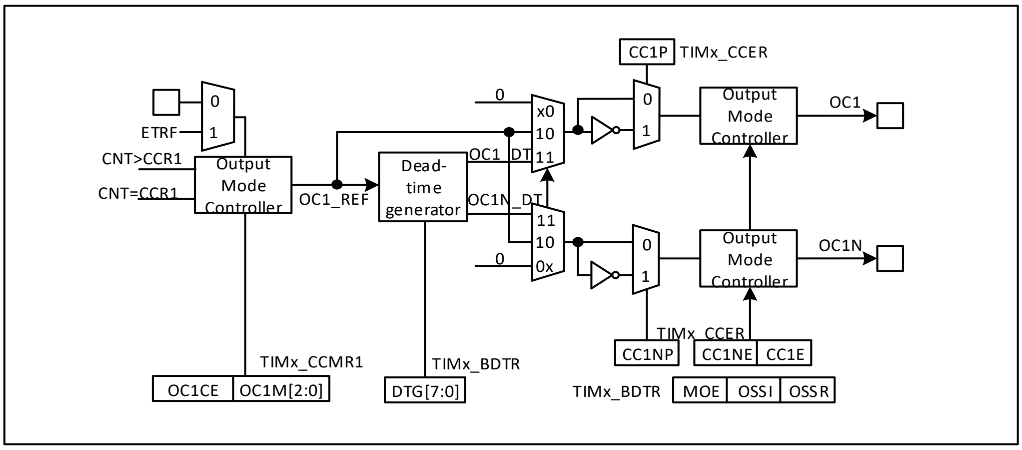


图 23-28 捕获/比较通道的输出部分(通道 1 至 3)

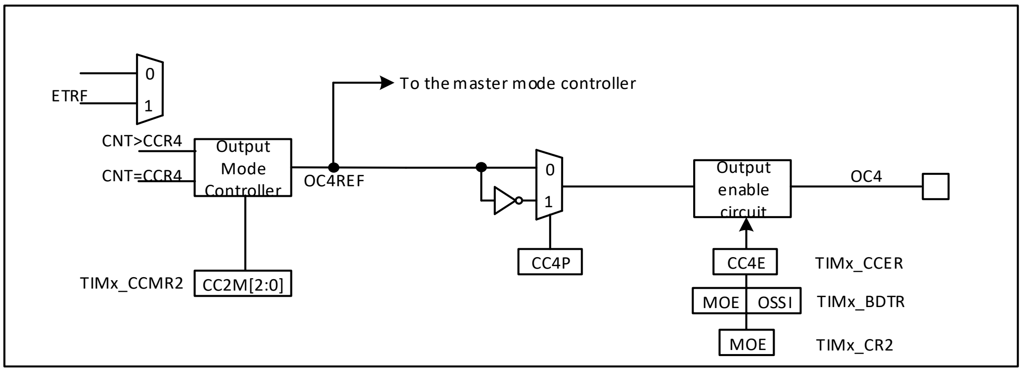


图 23-29 捕获/比较通道的输出部分(通道 4)

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。

23.4.6. 输入捕获模式

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

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

- 选择有效输入端：TIMx_CCMR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为'00'，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 Tix 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 IcxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 Fck_int 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)

- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

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

23.4.7. 输入捕获模式（PWM input mode）（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 信号被映射到同一个 Tix 输入。
- 这 2 个 Icx 信号为边沿有效，但是极性相反。
- 其中一个 TixFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

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

- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

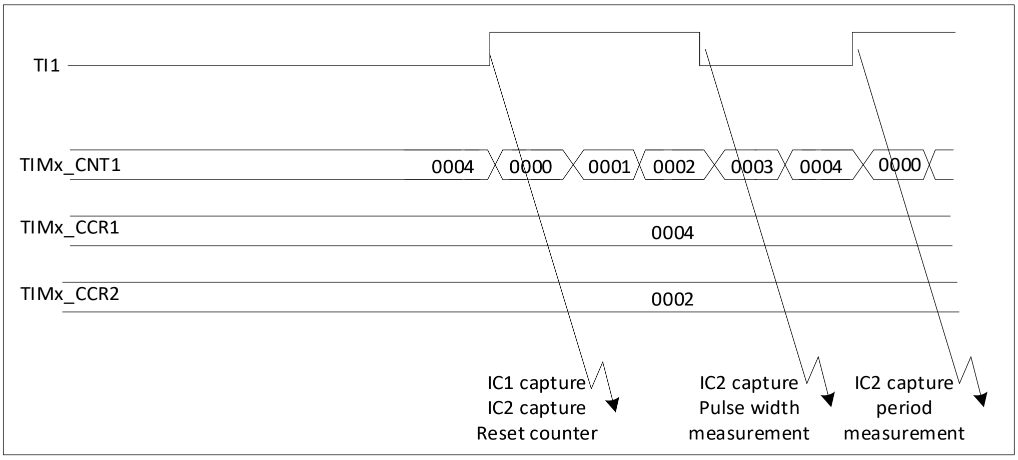


图 23-30 PWM 输入模式时序

23.4.8. 强置输出模式

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

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。 置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

23.4.9. 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)。
- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CcxIF 位)。
- 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CcxIE 位)，则产生一个中断。
- 若设置了相应的使能位(TIMx_DIER 寄存器中的 CcxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。 在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

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

输出比较模式的配置步骤：

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CcxIE 位。

4. 选择输出模式，例如：

— 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。

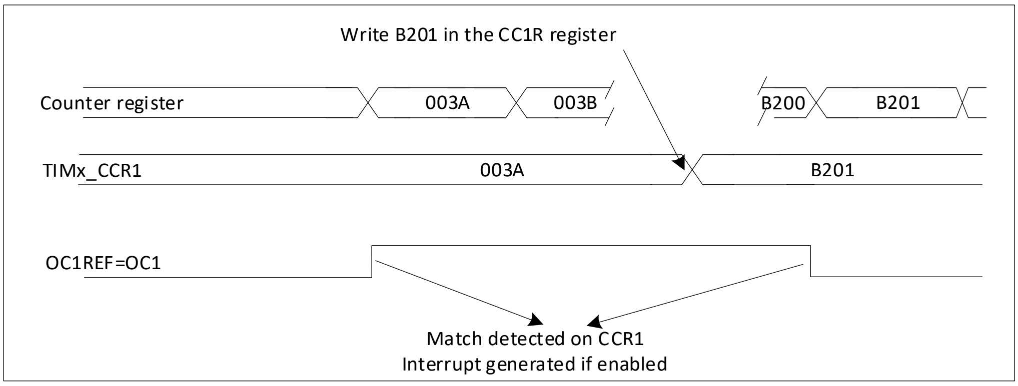


图 23-31 输出比较模式，翻转 OC1

23.4.10. PWM 模式

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

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

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

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

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

■ **向上计数配置**

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

动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

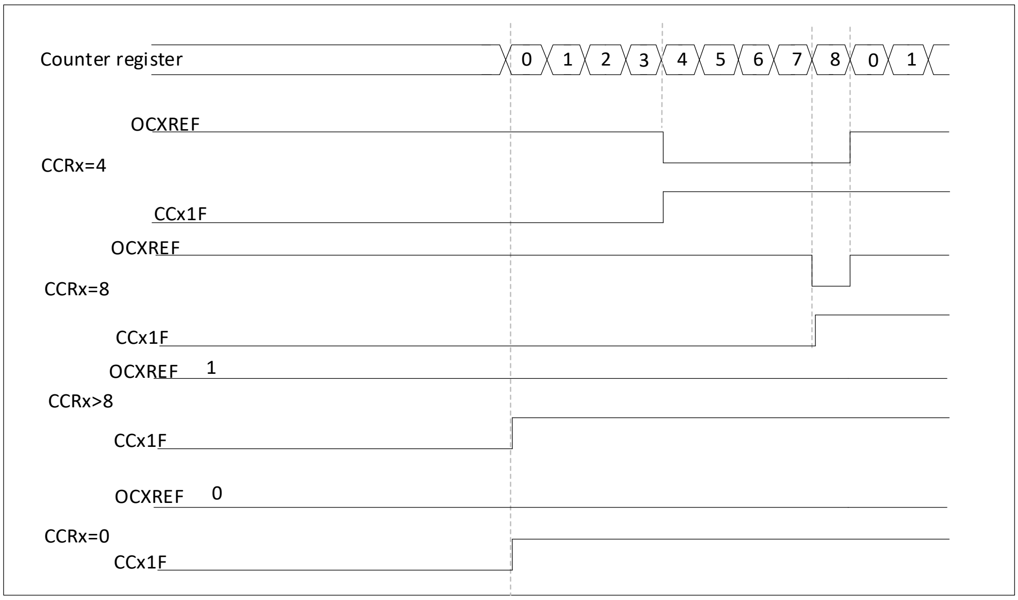


图 23-32 边沿对齐方式 PWM 输出，向上（ARR=8）

- **向下计数的配置**

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

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

下图给出一些中央对齐的 PWM 波形的例子

- TIMx_ARR = 8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中央对齐模式下，当计数器向下计数时设置比较标志

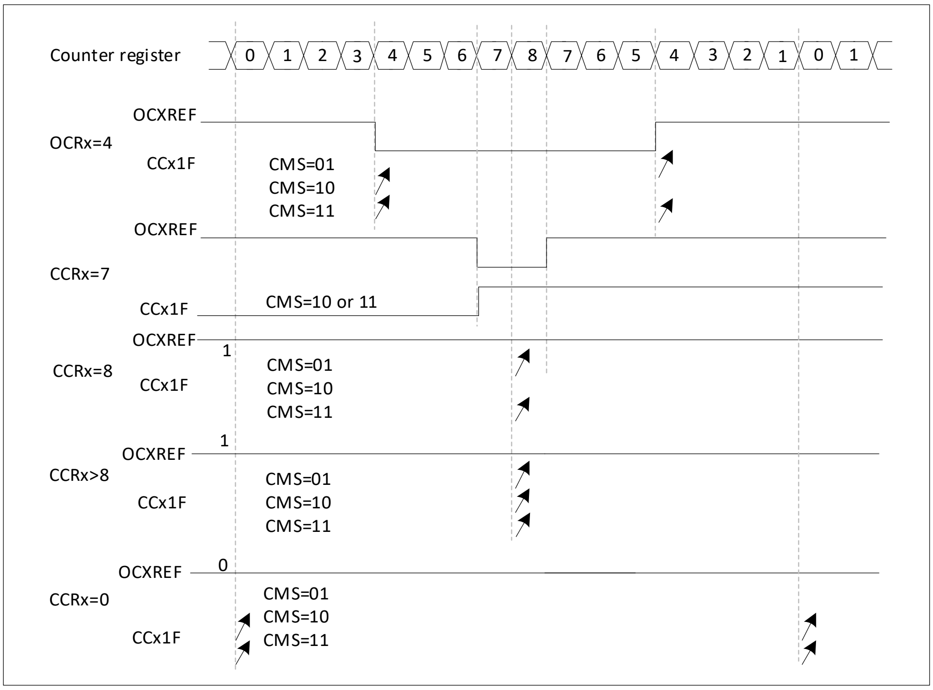


图 23-33 中央对齐的 PWM 波形(APR=8)

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。特别地： — 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。 — 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

23.4.11. 互补输出和死区插入

通用控制定时器(TIM15_16_17)能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。 这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

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

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

同时设置 CcxE 和 CcxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 8 位的死区发生器 DTG[7:0]。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

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

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

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

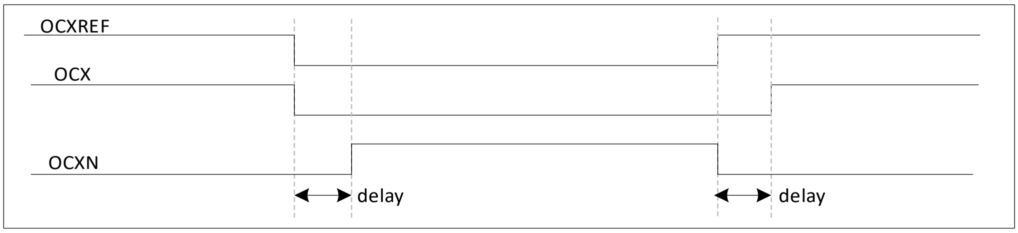


图 23-34 带死区插入的互补输出

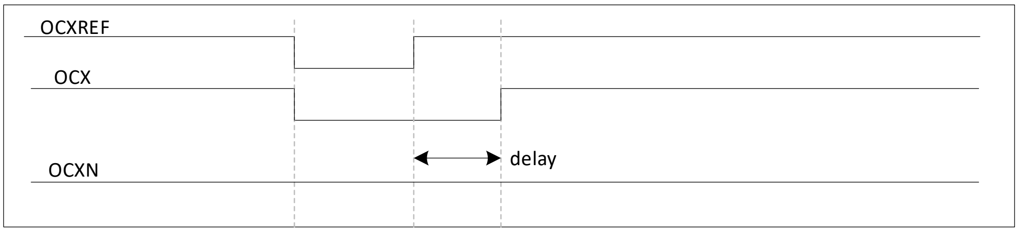


图 23-35 死区波形延迟大于负脉冲

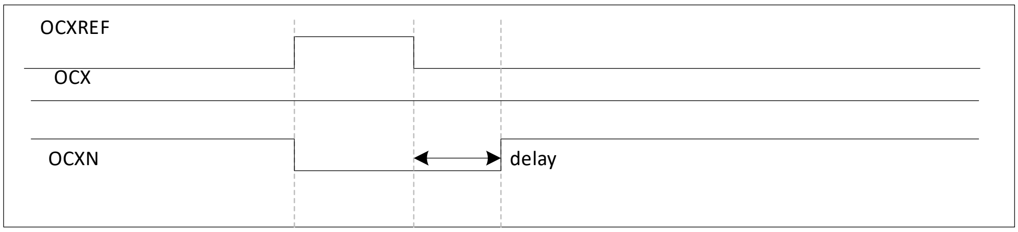


图 23-36 死区波形延迟大于正脉冲

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

重定向 OCxREF 到 OCx 或 OCxN

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

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

23.4.12. 使用刹车功能

当使用刹车功能时，依据额外的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平信号都会被修改。无论什么情况下，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源既可以是刹车输入引脚，或者以下内部源：

- 内核 LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟 failure 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，才能正确地读回写入的位。

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

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CcxE 与 CcxNE 之一变高时，使能输出变为高。
  - 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’1’时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
  - 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。刹车也可以通过软件设置 TIMx_EGR 寄存器中的 BG 位来产生。

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。

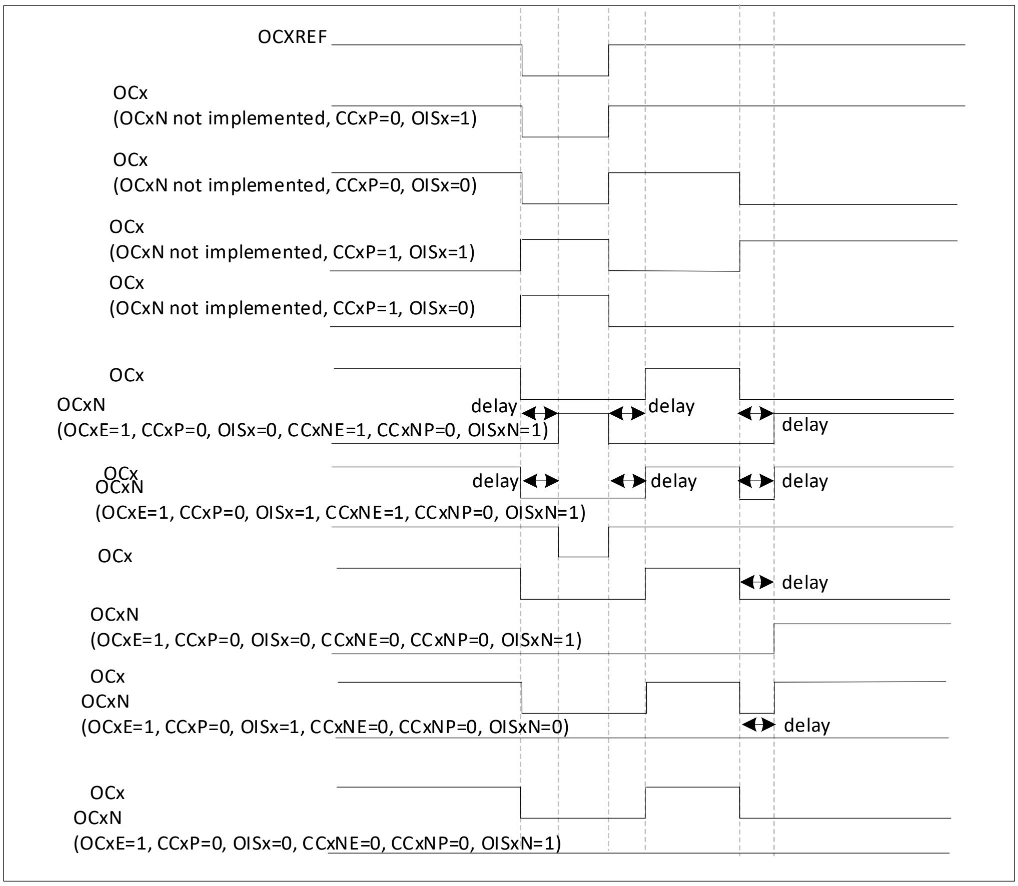


图 23-37 响应刹车的输出

23.4.13. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后，产生一个脉宽可被程序控制的脉冲。

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

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

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

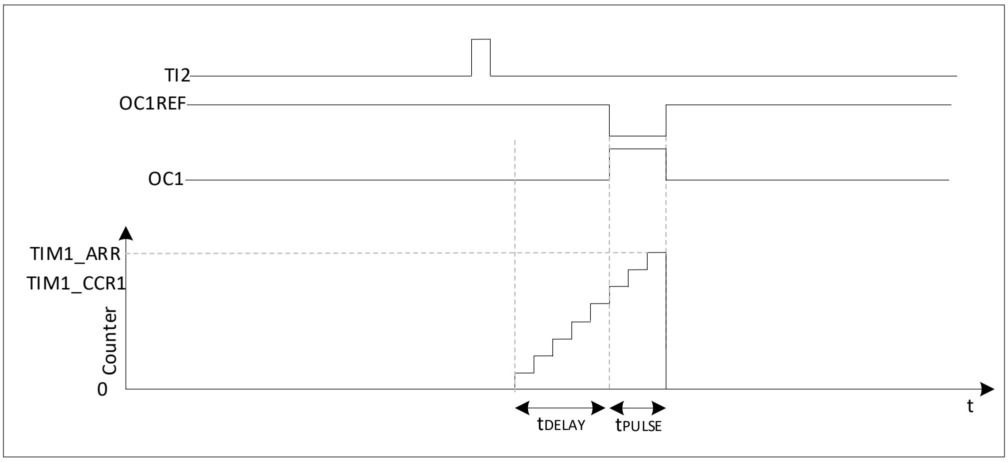


图 23-38 单脉冲模式的例子

例如，当需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 TI2FP2 作为触发 1:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

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

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

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 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果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(和 OCx)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 PWM1 和 PWM2 模式时起作用。

23.5. TIMx 定时器和外部触发的同步(仅 TIM15)

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。

23.5.1. 从模式：复位模式

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

在以下的例子中， TI1 输入端的上升沿导致向上计数器被清零：

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

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

● 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

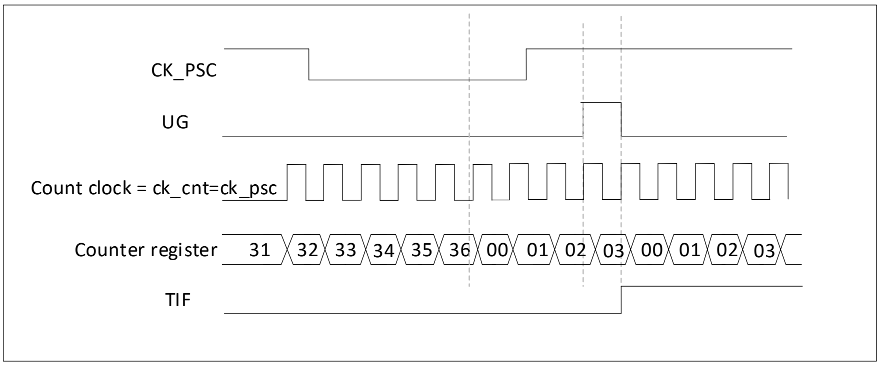


Figure 23-39 复位模式下的控制电路

23.5.2. 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

● 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。 CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

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

● 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

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

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

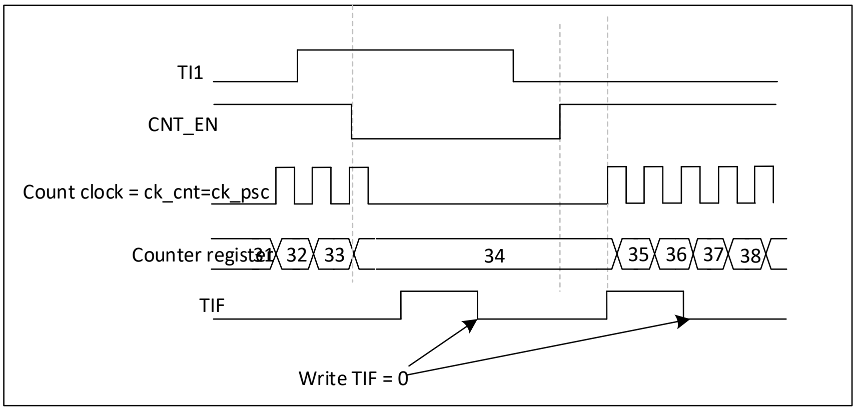


Figure 23-40 门控模式下的控制电路

23.5.3. 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

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

● 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

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

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

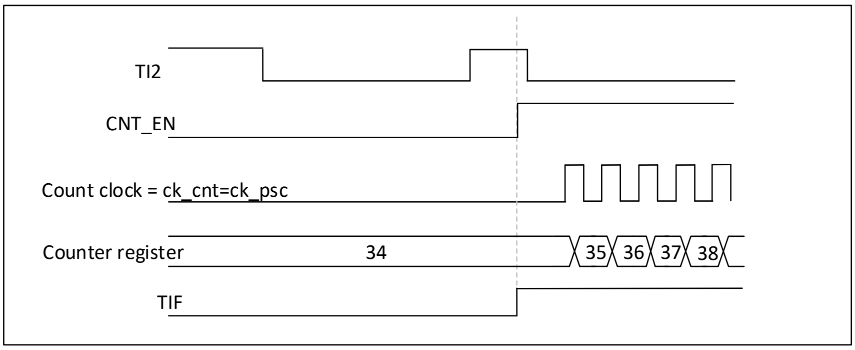


Figure 23-41 触发器模式下的控制电路

23.5.4. 从模式：外部时钟模式 2 + 触发模式

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

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

1. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：

一 ETF=0000：没有滤波

一 ETPS=00：不用预分频器

一 ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。

2. 按如下配置通道 1，检测 TI 的上升沿：

一 IC1F=0000：没有滤波

一 触发操作中不使用捕获预分频器，不需要配置

一 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源

一 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

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

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

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

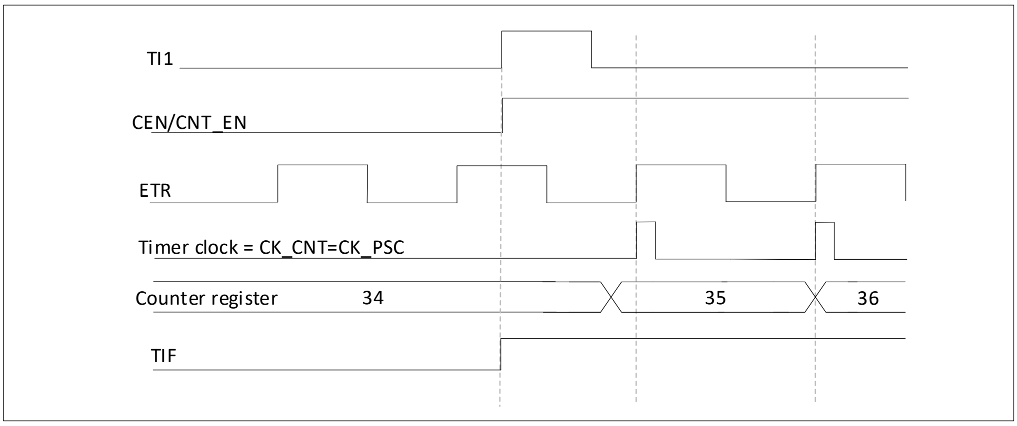


Figure 23-42 外部时钟模式 2 + 触发模式下的控制电路

23.5.5. TIM 和外部的触发同步

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

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

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

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

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

■ 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

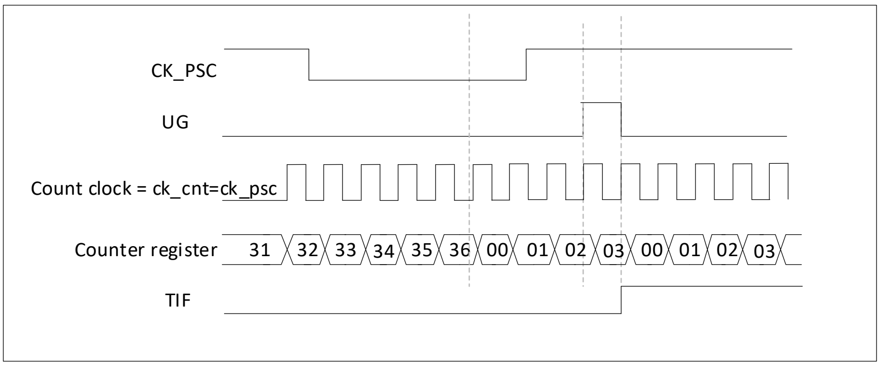


图 23-43 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

■ 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

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

■ 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

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

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

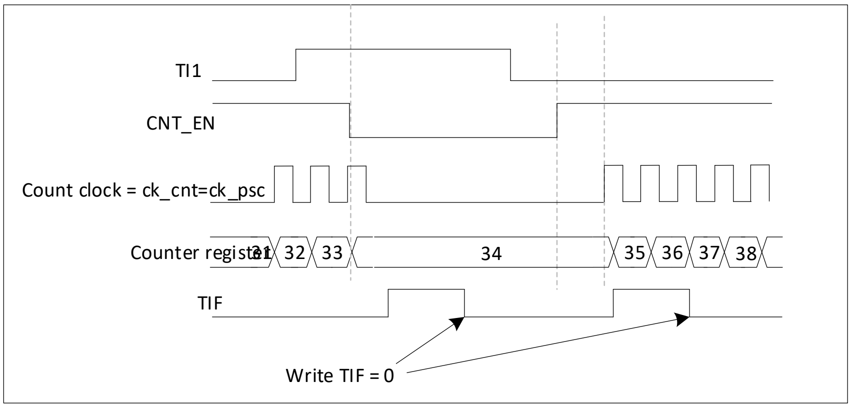


图 23-44 门控模式下的控制电路

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

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

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。 TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

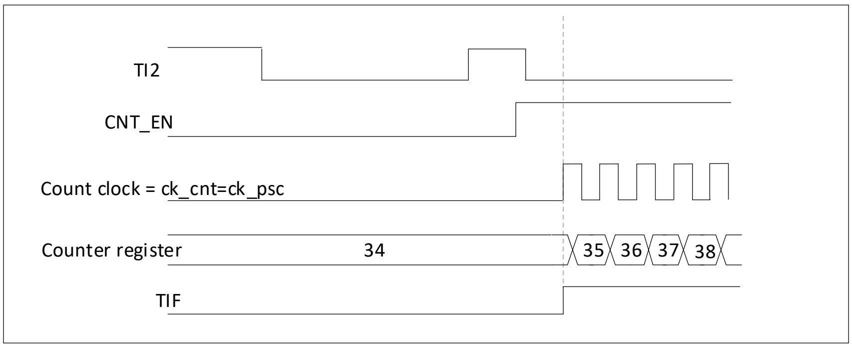


图 23-45 门控模式下的控制电路

从模式：外部时钟模式 2 ＋ 触发模式

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

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

- 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。
- 按如下配置通道 1，检测 TI 的上升沿：

– IC1F=0000：没有滤波

– 触发操作中不使用捕获预分频器，不需要配置

– 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源

– 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

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

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。 ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

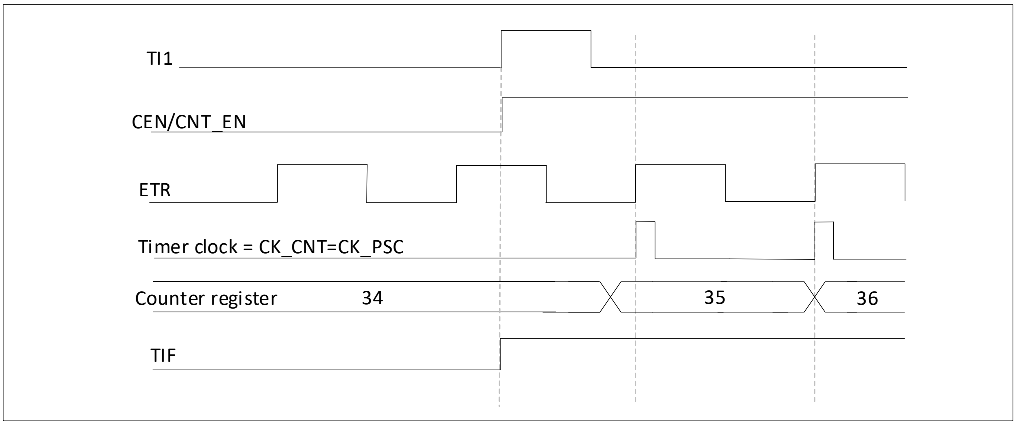


图 23-46 外部时钟模式 2 + 触发模式下的控制电路

23.6. 定时器同步(仅 TIM15)

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。

23.6.1. 使用一个定时器作为另一个定时器的预分频器

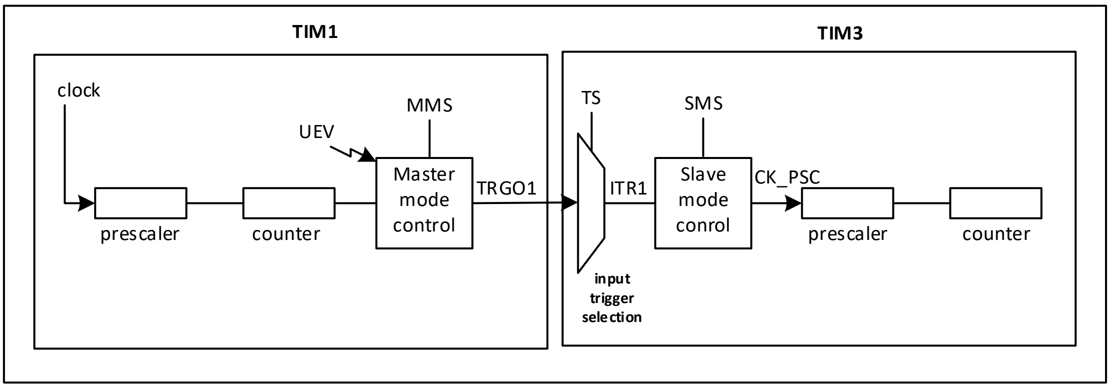


Figure 23-47 主/从定时器的例子

如：可以配置定时器 1 作为定时器 2 的预分频器。参考图 4-48，进行下述操作：

● 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM15_16_17_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 2，设置 TIM2_SMCR 寄存器的 TS=‘000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。
● 然后把从模式控制器置于外部时钟模式 1(TIM2_SMCR 寄存器的 SMS=111)；这样定时器 2 即可由定时器 1 周期性的上升沿(即定时器 1 的计数器溢出)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：如果 OCx 已被选中为定时器 1 的触发输出(MMS=1xx)，它的上升沿用于驱动定时器 2 的计数器。

23.6.2. 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考图 4-48 的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/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(TIM15_16_17_CR2 寄存器的 MMS=100)
● 配置定时器 1 的 OC1REF 波形(TIM15_16_17_CCMR1 寄存器)
● 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
● 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
● 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
● 置 TIM15_16_17_CR1 寄存器的 CEN=1 以启动定时器 1
注： 定时器 2 的时钟不与定时器 1 的时钟同步，这个模式只影响定时器 2 计数器的使能信号。

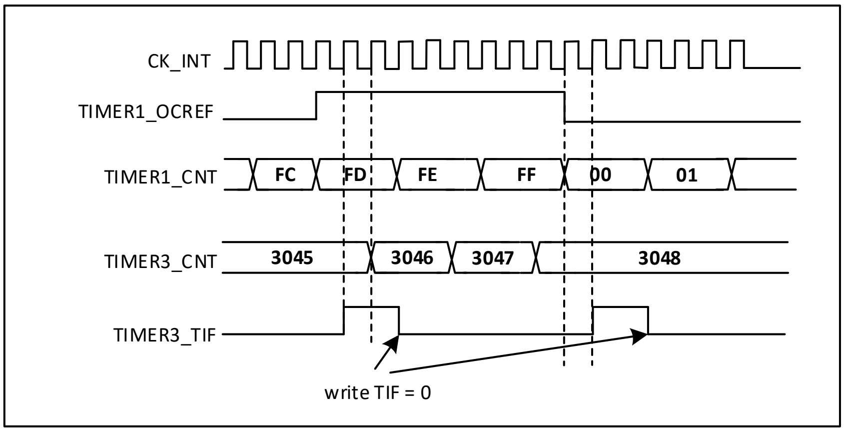


Figure 23-48 定时器 1 的 OC1REF 控制定时器 2

在图 4-47 的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 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 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写‘0’到 TIM15_16_17_CR1 的 CEN 位将禁止定时器 1，定时器 2 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)做为触发输出(TIM15_16_17_CR2 寄存器的 MMS=100)。
● 配置定时器 1 的 OC1REF 波形(TIM15_16_17_CCMR1 寄存器)。
● 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)

- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM15_16_17_EGR 寄存器的 UG=‘1’，复位定时器 1。
- 置 TIM2_EGR 寄存器的 UG=‘1’，复位定时器 2。
- 写‘0XE7’至定时器 2 的计数器(TIM2_CNT)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN=‘1’以使能定时器 2。
- 置 TIM15_16_17_CR1 寄存器的 CEN=‘1’以启动定时器 1。
- 置 TIM15_16_17_CR1 寄存器的 CEN=‘0’以停止定时器 1。

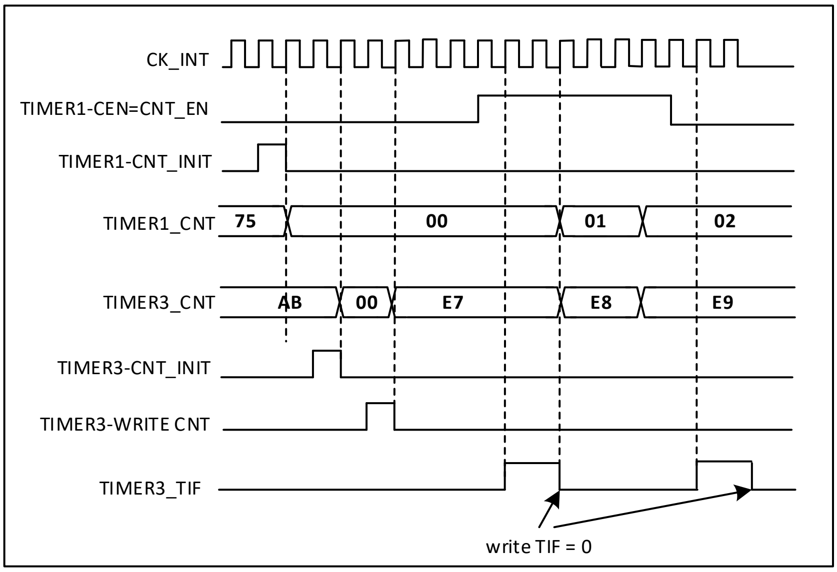


Figure 23-49 通过使能定时器 1 可以控制定时器 2

23.6.3. 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的更新事件使能定时器2。参考图4-47的连接。一旦定时器1产生更新事件，定时器 2 即从它当前的数值(可以是非 0)按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CEN 位被自动地置‘1’，同时计数器开始计数直到写‘0’到 TIM2_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/3)。

- 配置定时器 1 为主模式，送出它的更新事件(UEV)做为触发输出(TIM15_16_17_CR2 寄存器的 MMS=010)。
- 配置定时器 1 的周期(TIM15_16_17_ARR 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=110)
- 置 TIM15_16_17_CR1 寄存器的 CEN=1 以启动定时器 1。

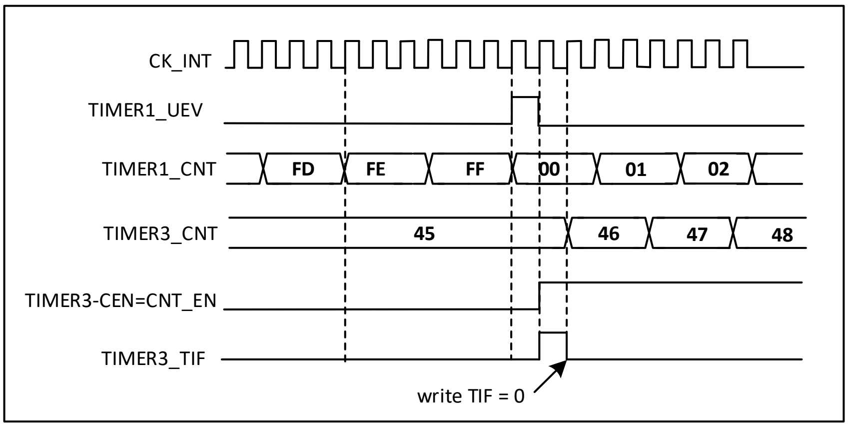


Figure 23-50 使用定时器 1 的更新触发定时器 2

在上一个例子中，可以在启动计数之前初始化两个计数器。显示在与 0 相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

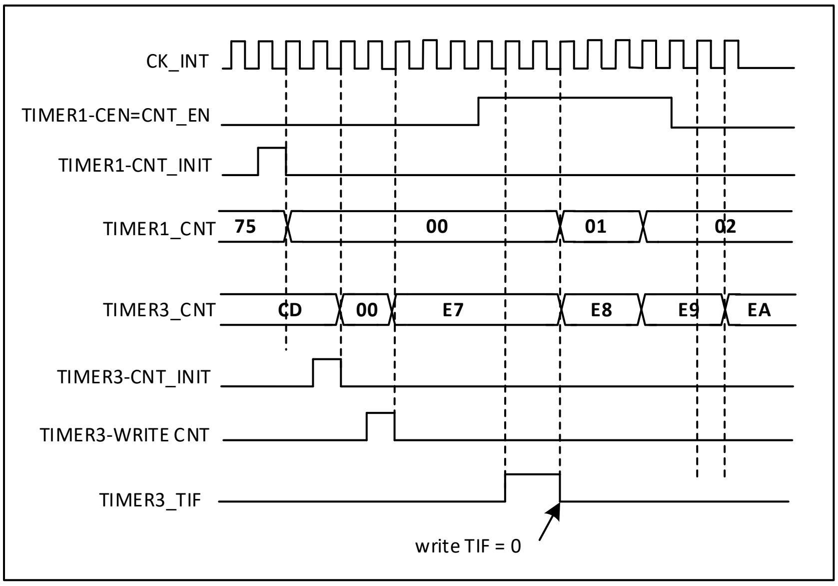


Figure 23-51 利用定时器 1 的使能触发定时器 2

23.6.4. 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 2，参见图 4-47。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，对应定时器 2 为主)：

- 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM15_16_17_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(TIM15_16_17_SMCR 寄存器的 TS=100)。
- 配置定时器 1 为触发模式(TIM15_16_17_SMCR 寄存器的 SMS=110)。
- 配置定时器 1 为主/从模式， TIM15_16_17_SMCR 寄存器的 MSM=1。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_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数，两个 TIF 标志也同时被设置。

注： 在这个例子中，在启动之前两个定时器都被初始化(设置相应的 UG 位)，两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器(TIMx_CNT)在定时器间插入一个偏移。下图中能看到主/从模式下在定时器 1 的 CNT_EN 和 CK_PSC 之间有个延迟。

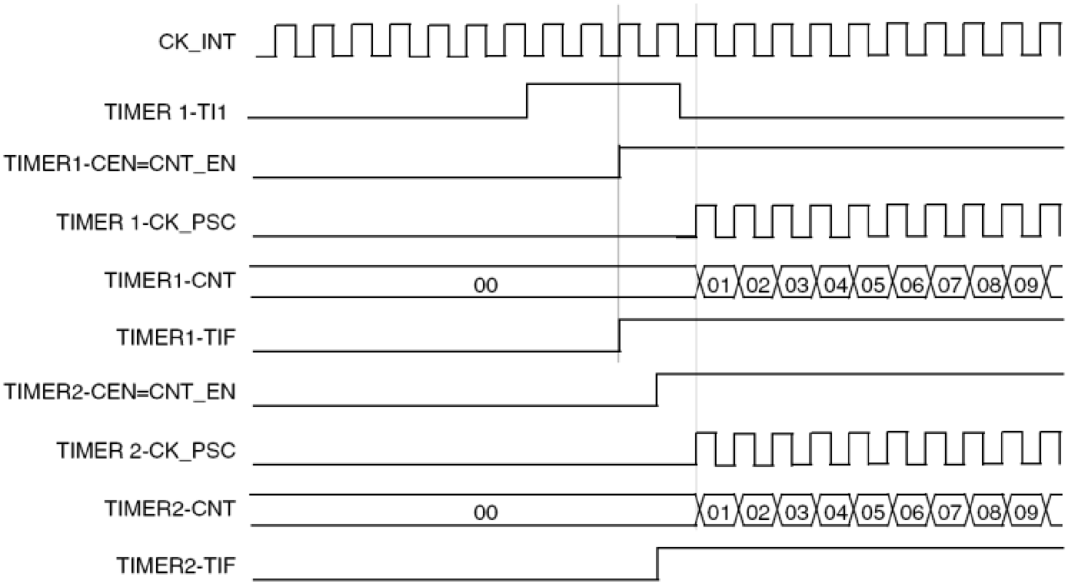


Figure 23-52 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2

23.6.5. 调试模式

当芯片进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以继续正常工作或者停止工作。

23.7. TIM15 寄存器描述

0x4001 4800 - 0x4001 4BFF TIM17
0x4001 4400 - 0x4001 47FF TIM16
0x4001 4000 - 0x4001 43FF TIM15

23.7.1. TIM15 控制寄存器 1 (TIMx_CR1)

Address offset:0x00

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE				OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	0	保留，始终为 0
9:8	CKD[1:0]	RW	00	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： tDTS = tCK_INT 01： tDTS = 2 x tCK_INT 10： tDTS = 4 x tCK_INT 11：保留，不要使用这个配置

Bit	Name	R/W	Reset Value	Function
7	ARPE	RW	0	自动重装载预装载允许位 0： TIMx_ARR 寄存器没有缓冲 1： TIMx_ARR 寄存器被装入缓冲器
6:4	-	-	-	-
3	OPM	RW	0	单脉冲模式 0：在发生更新事件时，计数器不停止 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中 断或 DMA 请求
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

23.7.2. TIM15 控制寄存器 2 (TIMx_CR2)

Address offset:0x04

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					OIS2	OIS1N	OIS1		MMS[2:0]			CCDS	CCUS		CCPC
					RW	RW	RW		RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
15:11	Reserved	-	0	保留，始终为 0
10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。参见 OIS1 位

Bit	Name	R/W	Reset Value	Function
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出)。 0：当 MOE=0 时，死区后 OC1N=0 1：当 MOE=0 时，死区后 OC1N=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出)。 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7	-	-	-	-
6:4	MMS[2:0]	RW	000	主模式选择 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如 下： 000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果触发输入(复位模式下的从模式控制器)产生复位，则 TRGO 上的信号相对实际的复位 会有一个延迟。 001：允许 – 计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要 在同一时间启动多个定时器或控制从定时器的一个窗口。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控 于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 – 一旦发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即是它已经为高)，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)。 注意： 1.从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。 2.若主从定时器不在同一总线上，主模式应该配置为能被从定时器采到的宽度。
3	CCDS	RW	0	捕获/比较的 DMA 选择 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们。 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上

Bit	Name	R/W	Reset Value	Function
				的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	Res	-	0	保留，始终读为 0。
0	CCPC	RW	0	捕获/比较预装载控制位 0： CcxE， CcxNE 和 OCxM 位不是预装载的。 1： CcxE， CcxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新。 注：该位只对具有互补输出的通道起作用。

23.7.3. TIM15 从模式控制寄存器 (TIMx_SMCR)

Address offset:0x08

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								MSM	TS[2:0]				SMS[2:0]		
								RW	RW				RW		

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	保留，始终为 0
7	MSM	RW	0	主/从模式 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的当前定时器和从定时器间的同步（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]	RW	000	触发选择，这 3 位选择用于同步计数器的触发输入。 000：Reserved(ITR0) 001：Reserved(ITR1) 010：TIM3(ITR2) 011：TIM17(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：为避免在信号转变时产生错误的边沿检测，必须在未使用这些位时修改它们
3	Reserved	-	0	保留，始终为 0
2：0	SMS[2:0]	RW	000	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：关闭从模式 如果 CEN=1，则预分频器直接由内部时钟驱动。 100：复位模式 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式

Bit	Name	R/W	Reset Value	Function
				当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。 注：在编码器模式下，不要使用 uev 作为 trgo 输出信号，（即 mms 不能配置为 010）

TIM1 内部触发连接

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	reserved	reserved	TIM3	TIM17 OC1

23.7.4. TIM15 DMA/中断使能寄存器 (TIMx_DIER)

Address offset:0x0C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TDE	COMDE			CC2DE	CC1DE	UDE	BIE	TIE	COMIE			CC2IE	CC1IE	UIE
-	RW	RW			RW	RW	RW	RW	RW	RW			RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved		0	保留，一直为 0
14	TDE	RW	0	TDE：允许触发 DMA 请求 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	COMDE：允许 COM 的 DMA 请求 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
11:12	-	-	-	-
10	CC2DE	RW	0	CC2DE：允许捕获/比较 2 的 DMA 请求 0：禁止捕获/比较 2 的 DMA 请求 1：允许捕获/比较 2 的 DMA 请求
9	CC1DE	RW	0	CC1DE：允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	BIE：允许刹车中断 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	TIE：允许触发中断

Bit	Name	R/W	Reset Value	Function
				0：禁止触发中断 1：允许触发中断
5	COMIE	RW	0	COMIE：允许 COM 中断 0：禁止 COM 中断 1：允许 COM 中断
4:3	Reserved	-	0	保留，始终为 0
2	CC2IE	RW	0	CC2IE：允许捕获/比较 2 中断 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	CC1IE：允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断 1：允许更新中断

23.7.5. TIM15 状态寄存器(TIMx_SR)

Address offset:0x010

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res			CC2OF	CC1OF	Res	BIF	TIF	COMIF			CC2IF	CC1F	UIF
-	-	-			Rc_w0	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0			Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
15:11	Reserved	-	0	保留，一直为 0
10	CC2OF	Rc_w0	0	捕获/比较 2 过捕获标记 参见 CC1OF 描述
9	CC1OF	Rc_w0	0	捕获/比较 1 过捕获标记 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：无过捕获产生； 1： CC1OF 置 1 时，计数器的值已经被捕获到 TIMx_CCR1 寄存器。
8	Res	Rc_w0	0	保留，始终读为 0。
7	BIF	Rc_w0	0	刹车中断标记 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	Rc_w0	0	触发器中断标记 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

Bit	Name	R/W	Reset Value	Function
				效边沿，或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：触发器中断等待响应
5	COMIF	Rc_w0	0	COM 中断标记 一旦产生 COM 事件（当 CcxE、 CcxNE、 OCxM 已被更新）该位由硬件置 1。它由软件清 0。 0：无 COM 事件产生； 1： COM 中断等待响应
4:3	Reserved	-	0	保留，一直为 0
2	CC2IF	Rc_w0	0	捕获/比较 2 中断标记 参考 CC1IF 描述
1	CC1IF	Rc_w0	0	捕获/比较 1 中断标记 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存 器的 CMS 位)。它由软件清 0。 0：无匹配发生； 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置 1，它由软件清 0 或通过读 TIMx_CCR1 清 0。 0：无输入捕获产生； 1：输入捕获产生并且计数器值已装入 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。 注：当 CEN 打开，该位也会被置位。
0	UIF	Rc_w0	0	更新中断标记 当产生更新事件时该位由硬件置 1。它由软件清 0。 0：无更新事件产生； 1：更新事件等待响应。当寄存器被更新时该位由硬件置 1： – 若 TIMx_CR1 寄存器的 UDIS=0，当 REP_CNT=0 时产生更新事件(重复向下计数器上溢或下溢时)； – 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事 件(软件对 CNT 重新初始化)； – 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 CNT 被触发事件重初始化时产生更新事 件。（参考从模式控制寄存器(TIMx_SMCR)）

23.7.6. TIM15 事件产生寄存器(TIMx_EGR)

Address offset:0x14

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	BG	TG	COMG			CC2G	CC1G	UG

-	-	-	--	-	-	-	-	W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	保留，一直为 0
7	BG	W	0	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清 0。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断，则产生相应的中断。
6	TG	W	0	产生触发事件 该位由软件置 1，用于产生一个触发事件，由硬件自动清 0。 0：无动作； 1： TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	W	0	捕获/比较事件，产生控制更新 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：当 CCPC=1，允许更新 CcxE、 CcxNE、 OCxM 位。 注： 该位只对有互补输出的通道有效。
4:3	Reserved	-	0	保留，一直为 0
2	CC2G	W	0	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	W	0	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 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	W	0	产生更新事件。该位由软件置 1，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)则计数器装载 TIMx_ARR 的值。

23.7.7. TIM15 捕获/比较模式寄存器 1(TIMx_CCMR1)

Address offset:0x18

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]			OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式

Output compare mode:

Bit	Name	R/W	Reset Value	Function
15	Reserved	-	0	保留，一直为 0
14:12	OC2M[2:0]	RW	0	输出比较 2 模式选择
11	OC2PE	RW	0	输出比较 2 预装载使能
10	OC2FE	RW	0	输出比较 2 快速使能
9:8	CC2S[1:0]	RW	00	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： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	Reserved	-	0	保留，一直为 0
6:4	OC1M[2:0]	RW	00	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 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有

Bit	Name	R/W	Reset Value	Function
				效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，且新值马上起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作， TIMx_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 该位用于加快 CC 输出对触发器输入事件的响应。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而 与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 的只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIMx_SMCR 寄存器的 TS 位选择）。 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
15:12	IC2F	RW	0	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	0	输入/捕获 2 预分频器

Bit	Name	R/W	Reset Value	Function
9:8	CC2S[1:0]	RW	0	捕获/比较 2 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIMx_SMCR 寄存器的 TS 位选择）。 注： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	0000	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， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8， N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC[1:0]	RW	00	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S[1:0]	RW	00	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）。

Bit	Name	R/W	Reset Value	Function
				注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

23.7.8. TIM15 捕获/比较使能寄存器 (TIMx_CCER)

Address offset:0x20

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res							CC2NP		CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
-	-							RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:14	Reserved	-	0	保留，一直为 0
13:8	Reserved	-	0	保留，一直为 0
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	Reserved	-	0	保留，一直为 0
5	CC2P	RW	0	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0： OC1N 高电平有效 1： OC1N 低电平有效 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LCCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE，OSSI， OSSR，OIS1，OIS1N，CC1E 位的值。
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿

Bit	Name	R/W	Reset Value	Function
				TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。 注： 1.对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1P 的实际有效位只有在 com 事件发生时才会加载预载值。 2.一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改
0	CC1E	RW	0	输入/捕获 1 输出使能 CC1 通道配置为输出： 0：关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N、CC1NE 位的值。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、 OSSR、OIS1、OIS1N、CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 0：捕获使能 注： 对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

Table23-53 具有中断功能的互补 OCx 和 OCxN 通道的输出控制

<table>
<tr><th colspan="5">Control bits</th><th colspan="2">Output state</th></tr>
<tr><th>MOE</th><th>OSSI</th><th>OSSR</th><th>CcxE</th><th>CcxNE</th><th>OCx output state</th><th>OCxN output state</th></tr>
<tr><td rowspan="7">1</td><td rowspan="7">X</td><td>0</td><td>0</td><td>0</td><td>输出禁止(与定时器断开)，OCx=0, OCx_EN=0</td><td>输出禁止(与定时器断开)，OCxN=0, OCxN_EN=0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>输出禁止(与定时器断开)，OCx=0, OCx_EN=0</td><td>OCxREF + Polarity OCxN=OCxREF 异或 CCxNP, OCxN_EN=1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>OCxREF + Polarity OCx=OCREF 异或 CCxP, OCx_EN=1</td><td>输出禁止(与定时器断开)，OCxN=0, OCxN_EN=0</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>OCREF + Polarity + dead-time OCx_EN=1</td><td>OCREF 的互补（not OCREF）+ Polarity + dead-time OCxN_EN=1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>输出禁止(与定时器断开)，OCx=CCxP, OCx_EN=0</td><td>输出禁止(与定时器断开)，OCxN=CCxNP, OCxN_EN=0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>输出禁止(与定时器断开), OCx=CCxP, OCx_EN=1</td><td>OCxREF+Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>OCxREF+Polarity OCx=OCxREF xor CCxP, OCx_EN=1</td><td>关闭状态（输出使能且为无效电平），</td></tr>
</table>

<table>
<tr><th colspan="5">Control bits</th><th colspan="2">Output state</th></tr>
<tr><th>MOE</th><th>OSSI</th><th>OSSR</th><th>CcxE</th><th>CcxNE</th><th>OCx output state</th><th>OCxN output state</th></tr>
<tr><td rowspan="2"></td><td rowspan="2"></td><td></td><td></td><td></td><td></td><td>OCxN=CCxNP, OCxN_EN=1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>OCREF+Polarity + dead-time OCx_EN=1</td><td>OCREF 的互补（not OCREF）+ polarity + dead-time OCN_EN=1</td></tr>
<tr><td rowspan="8">0</td><td>0</td><td rowspan="8">X</td><td>0</td><td>0</td><td rowspan="5" colspan="2">输出禁止(与定时器断开)
异步的: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0
如果时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，OCx=OISx 和 OCxN=OISxN。</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td rowspan="3" colspan="2">关闭状态（输出使能且为无效电平）
异步的：OCx=CCxP, OCx_EN=1，OCxN=CCxNP, OCxN_EN=1
若时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，OCx=OISx 和 OCxN=OISxN</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>

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： 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 以及 AFIO 寄存器。

23.7.9. TIM15 计算器(TIMx_CNT)

Address offset:0x24

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	CNT[15:0]	RW	0	计数器的值

23.7.10. TIM15 预分频器 (TIMx_PSC)

Address offset:0x28

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	PSC[15:0]	RW	0	预分频器的值 计数器的时钟频率（CK_CNT）等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器 被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

23.7.11. TIM15 自动重新加载寄存器 (TIMx_ARR)

Address offset: 0x2c

Reset value:0x0000 FFFF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	ARR[15:0]	RW	0	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

23.7.12. TIM15 重复计数器寄存器(TIMx_RCR)

Address offset： 0x30

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	**REP[7:0]**							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved			保留，一直为 0
7:0	REP[7:0]	RW	0	周期计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄 存器传输到当前寄存器）；如允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着: - 在边沿对齐模式下，PWM 周期的数目; - 在中心对称模式下，PWM 半周期的数目;

23.7.13. TIM15 捕获/比较寄存器 1(TIMx_CCR1)

Address offset： 0x34

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
15：0	CCR1[15:0]	RW/RO	0	捕获/比较 1 的值 若 CC1 通道配置为输出: CCR1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。

Bit	Name	R/W	Reset Value	Function
				当前捕获/比较寄存器包含了与计数器 TIMx_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

23.7.14. TIM15 捕捉/比较寄存器 2(TIMx_CCR2)

Address offset： 0x38

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
15:0	CCR2[15:0]	RW/RO	0	捕获/比较 2 的值 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIMx_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件（IC2）传输的计数器值。

23.7.15. TIM15 刹车和死区寄存器(TIMx_BDTR)

Address offset： 0x44

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

注释：根据锁定设置，AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0]位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

Bit	Name	R/W	Reset Value	Function
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的值，可由软件清 0 或自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位（TIMx_CCER 寄存器的 CcxE、 CcxNE 位），则开启 OC 和 OCN 输出。

Bit	Name	R/W	Reset Value	Function
14	AOE	RW	0	自动输出使能 0： MOE 只能被软件置 1； 1： MOE 能被软件置 1 或在下一个更新事件自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入（BRK 及 BRK_ACTH）； 1：开启刹车输入（BRK 及 BRK_ACTH）。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIM1_CCER)）。 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：当定时器不工作时，一旦 CcxE=1 或 CcxNE=1，开启 OC/OCN 输出并输出无效电平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIMx_CCER)）。 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1： 当 定 时器不 工 作时， 一旦 CcxE=1 或 CcxNE=1，OC/OCN 首 先 输出 其 空闲 电 平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	00	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护；

Bit	Name	R/W	Reset Value	Function
				01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，TIMx_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，TIMx_CCMRx 寄存器的 OCxM/OCxPE 位）； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	0000 0000	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] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg， Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、 2 或 3，则这些位不能被修改。

23.7.16. TIM15 DMA 控制寄存器(TIMx_DCR)

Address offset： 0x48

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	DBL[4:0]					Res			DBA[4:0]				
-	-	-	RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:13	Reserved			保留，一直为 0
12:8	DBL[4:0]	RW	0 0000	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送），即：定义被传送的字节数目：

Bit	Name	R/W	Reset Value	Function
				00000：1 次传输 00001：2 次传输 00010：3 次传输 …… …… 10001：18 次传输 例：我们考虑这样的传输： DBL=7， DBA=TIM2_CR1 - 如果 DBL=7， DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引), 其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	RW	0	保留，始终读为 0
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIMx_DMAR 寄存器的地址进行读或写时）， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000： TIMx_CR1， 00001： TIMx_CR2， 00010： TIMx_SMCR， ……

23.7.17. TIM15 连续模式的 DMA 地址(TIMx_DMAR)

Address offset： 0x4C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB[15:0]															

Bit	Name	R/W	Reset Value	Function
15：0	DMAB[31:0]	RW	0	DMA 连续传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIMx_CR1 地址 + DBA + DMA 指针，其中： “TIMx_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 指针”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

注：在使用 DMA 连续传输功能时，必须将 DMA 中对应通道的 CNDTR 寄存器的值与 TIMx_DCR 寄存器中 DBL 的值对应起来，否则该将不能正常使用。

23.7.18. TIM15 寄存器映像

偏移	寄存器	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
000h	TIMx_CR1	保留																						CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	复位值																							0	0	0	0	0	0	0	0	0	0
004h	TIMx_CR2	保留																	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	保留	CCPC
	复位值																		0	0	0	0	0	0	0	0	0	0	0	0	0		0
008h	TIMx_SMCR	保留																ETP	ECE	ETPS[1:0]		EFT[3:0]				MSM	TS[2:0]			保留	SMS[2:0]		
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
00Ch	TIMx_DIER	保留																	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	复位值																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	TIMx_SR	保留																			CC4OF	CC3OF	CC2OF	CC1OF	保留	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	复位值																				0	0	0	0		0	0	0	0	0	0	0	0
014h	TIMx_EGR	保留																								BG	TG	COM	CC4G	CC3G	CC2G	CC1G	UG
	复位值																									0	0	0	0	0	0	0	0
018h	TIMx_CCMR1 输出比较模式	保留																OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR1 输入捕获模式	保留																IC2F[3:0]				IC2 PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1 PSC[1:0]		CC1S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01Ch	TIMx_CCMR2 输出比较模式	保留																OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR2 输入捕获模式	保留																IC4F[3:0]				IC4 PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3 PSC[1:0]		CC3S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
020h	TIMx_CCER	保留																		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	复位值																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
024h	TIMx_CNT	保留																CNT[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
028h	TIMx_PSC	保留																PSC[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

偏移	寄存器	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
02Ch	TIMx_ARR	保留																ARR[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
030h	TIMx_RCR	保留																								REP[7:0]							
	复位值																									0	0	0	0	0	0	0	0
034h	TIMx_CCR1	保留																CCR1[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
038h	TIMx_CCR2	保留																CCR2[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
03Ch	TIMx_CCR3	保留																CCR3[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
040h	TIMx_CCR4	保留																CCR4[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
044h	TIMx_BDTR	保留																MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DT[7:0]							
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
048h	TIMx_DCR	保留																			DBL[4:0]					保留			DBA[4:0]				
	复位值																				0	0	0	0	0				0	0	0	0	0
04Ch	TIMx_DMAR	保留																DMAB[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.8. TIM16_17 寄存器描述

0x4001 4800 - 0x4001 4BFF TIM17
0x4001 4400 - 0x4001 47FF TIM16
0x4001 4000 - 0x4001 43FF TIM15

23.8.1. TIM16_17 控制寄存器 1 (TIMx_CR1)

Address offset: 0x00

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE				OPM	URS	UDIS	CEN
-	-	-	-	-	-	RW		RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	0	保留，始终为 0
9:8	CKD[1:0]	RW	00	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： tDTS = tCK_INT 01： tDTS = 2 x tCK_INT 10： tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： TIMx_ARR 寄存器没有缓冲 1： TIMx_ARR 寄存器被装入缓冲器
6:4	-	-	-	-
3	OPM	RW	0	单脉冲模式 0：在发生更新事件时，计数器不停止

Bit	Name	R/W	Reset Value	Function
				1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断或 DMA 请求，则下述任一事件产生一个更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果允许产生更新中断或 DMA 请求，则只有计数器溢出/下溢产生一个更新中 断或 DMA 请求
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

23.8.2. TIM16_17 控制寄存器 2 (TIMx_CR2)

Address offset： 0x04

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						OIS1N	OIS1					CCDS	CCUS	Res	CCPC
Rw						RW	RW					Rw	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved	-	0	保留，始终为 0
14:10	-	-	-	
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出)。 0：当 MOE=0 时，死区后 OC1N=0 1：当 MOE=0 时，死区后 OC1N=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出)。 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0

Bit	Name	R/W	Reset Value	Function
				1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7:4	-	-	-	-
3	CCDS	RW	0	捕获/比较的 DMA 选择 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们。 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上 的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	Res	-	0	保留，始终读为 0。
0	CCPC	RW	0	捕获/比较预装载控制位 0： CcxE， CcxNE 和 OCxM 位不是预装载的。 1： CcxE， CcxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新。 注：该位只对具有互补输出的通道起作用。

23.8.3. TIM16_17 DMA/中断使能寄存器 (TIMx_DIER)

Address offset： 0x0C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CC1DE	UDE	BIE		COMIE				CC1IE	UIE
-						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
15	Reserved		0	保留，一直为 0
14:10	-	-	-	-
9	CC1DE	RW	0	CC1DE：允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	BIE：允许刹车中断 0：禁止刹车中断 1：允许刹车中断
6	-	-	-	-
5	COMIE	RW	0	COMIE：允许 COM 中断 0：禁止 COM 中断 1：允许 COM 中断
4:2	-	-	-	-
1	CC1IE	RW	0	CC1IE：允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断

Bit	Name	R/W	Reset Value	Function
				1：允许捕获/比较 1 中断
0	UIE	RW	0	UIE：允许更新中断 0：禁止更新中断 1：允许更新中断

23.8.4. TIM16_17 状态寄存器(TIMx_SR)

Address offset： 0x010

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res				CC1OF		BIF		COMIF				CC1F	UIF
-	-	-				Rc_w0		Rc_w0		Rc_w0				Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
15：13	Reserved	-	0	保留，一直为 0
12:10	-	-	-	-
9	CC1OF	Rc_w0	0	捕获/比较 1 过捕获标记 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：无过捕获产生； 1： CC1OF 置 1 时，计数器的值已经被捕获到 TIMx_CCR1 寄存器。
8	Res	Rc_w0	0	保留，始终读为 0。
7	BIF	Rc_w0	0	刹车中断标记 一旦刹车输入有效，由硬件对该位置 1。如果刹车输入无效，则该位可由软件清 0。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
6	-	-	-	-
5	COMIF	Rc_w0	0	COM 中断标记 一旦产生 COM 事件（当 CcxE、 CcxNE、 OCxM 已被更新）该位由硬件置 1。它由软件清 0。 0：无 COM 事件产生； 1： COM 中断等待响应
4:2	-	-	-	-
1	CC1IF	Rc_w0	0	捕获/比较 1 中断标记 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 TIMx_CR1 寄存 器的 CMS 位)。它由软件清 0。 0：无匹配发生； 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 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；

Bit	Name	R/W	Reset Value	Function
				1：输入捕获产生并且计数器值已装入 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	Rc_w0	0	更新中断标记 当产生更新事件时该位由硬件置 1。它由软件清 0。 0：无更新事件产生； 1：更新事件等待响应。当寄存器被更新时该位由硬件置 1： – 若 TIMx_CR1 寄存器的 UDIS=0，当 REP_CNT=0 时产生更新事件(重复向下计数器上溢或下溢时)； – 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)； – 若 TIMx_CR1 寄存器的 UDIS=0、 URS=0，当 CNT 被触发事件重初始化时产生更新事件。（参考从模式控制寄存器(TIMx_SMCR)）

23.8.5. TIM16_17 事件产生寄存器(TIMx_EGR)

Address offset： 0x14

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	BG		COMG				CC1G	UG
-	-	-	--	-	-	-	-	W		W				W	W

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	0	保留，一直为 0
7	BG	W	0	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清 0。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断，则产生相应的中断。
6	-	-	-	-
5	COMG	W	0	捕获/比较事件，产生控制更新 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：当 CCPC=1，允许更新 CcxE、 CcxNE、 OCxM 位。 注： 该位只对有互补输出的通道有效。
4:2	-	-	-	-
1	CC1G	W	0	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。

Bit	Name	R/W	Reset Value	Function
				若通道 CC1 配置为输入： 当前的计数器值捕获至 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	W	0	产生更新事件。该位由软件置 1，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)则计数器装载 TIMx_ARR 的值。

23.8.6. TIM16_17 捕获/比较模式寄存器 1(TIMx_CCMR1)

Address offset： 0x18

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式

Output compare mode:

Bit	Name	R/W	Reset Value	Function
15:7	-	-	-	-
6:4	OC1M[2:0]	RW	00	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 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1) 相同时，强制 OC1REF 为高。 010： 匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。

Bit	Name	R/W	Reset Value	Function
				111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，且新值马上起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作， TIMx_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 该位用于加快 CC 输出对触发器输入事件的响应。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而 与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 的只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S[1:0]	RW	00	捕获/比较 1 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIMx_SMCR 寄存器的 TS 位选择）。 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

输入捕获模式

Input Capture mode:

Bit	Name	R/W	Reset Value	Function
15:12	IC2F	RW	0	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	0	输入/捕获 2 预分频器
9:8	CC2S[1:0]	RW	0	捕获/比较 2 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 （由 TIMx_SMCR 寄存器的 TS 位选择）。 注： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F[3:0]	RW	0000	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， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 fSAMPLING=fDTS/8， N=6 0001：采样频率 fSAMPLING=fCK_INT，N=2 1001：采样频率 fSAMPLING=fDTS/8，N=8 0010：采样频率 fSAMPLING=fCK_INT，N=4 1010：采样频率 fSAMPLING=fDTS/16，N=5 0011：采样频率 fSAMPLING=fCK_INT，N=8 1011：采样频率 fSAMPLING=fDTS/16，N=6 0100：采样频率 fSAMPLING=fDTS/2，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 0101：采样频率 fSAMPLING=fDTS/2，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 0110：采样频率 fSAMPLING=fDTS/4，N=6 1110：采样频率 fSAMPLING=fDTS/32，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1111：采样频率 fSAMPLING=fDTS/32，N=8
3:2	IC1PSC[1:0]	RW	00	输入/捕获 1 预分频器 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0(TIM1_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S[1:0]	RW	00	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；

Bit	Name	R/W	Reset Value	Function
				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）。

23.8.7. TIM16_17 捕获/比较使能寄存器 (TIMx_CCER)

Address offset： 0x20

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	CC1NE	CC1P	CC1E
-	-	-	-	-	-	-	-	-	-	-	-	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:4	Reserved	-	0	保留，一直为 0
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0： OC1N 高电平有效 1： OC1N 低电平有效 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LCCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE，OSSI， OSSR，OIS1，OIS1N，CC1E 位的值。
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。 注：

Bit	Name	R/W	Reset Value	Function
				一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改
0	CC1E	RW	0	输入/捕获 1 输出使能 CC1 通道配置为输出： 0：关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N、CC1NE 位的值。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、 OSSR、OIS1、OIS1N、CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 0：捕获使能

Table23-54 具有中断功能的互补 OCx 和 OCxN 通道的输出控制

Control bits					Output state	
MOE	OSSI	OSSR	CcxE	CcxNE	OCx output state	OCxN output state
1	X	0	0	0	输出禁止(与定时器断开)，OCx=0, OCx_EN=0	输出禁止(与定时器断开)，OCxN=0, OCxN_EN=0
		0	0	1	输出禁止(与定时器断开)，OCx=0, OCx_EN=0	OCxREF + Polarity OCxN=OCxREF 异或 CCxNP, OCxN_EN=1
		0	1	0	OCxREF + Polarity OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(与定时器断开)，OCxN=0, OCxN_EN=0
		0	1	1	OCREF + Polarity + dead-time OCx_EN=1	OCREF 的互补（not OCREF）+ Polarity + dead-time OCxN_EN=1
		1	0	0	输出禁止(与定时器断开)，OCx=CCxP, OCx_EN=0	输出禁止(与定时器断开)，OCxN=CCxNP, OCxN_EN=0
		1	0	1	输出禁止(与定时器断开), OCx=CCxP, OCx_EN=1	OCxREF+Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+Polarity OCx=OCxREF xor CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平）， OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF+Polarity + dead-time OCx_EN=1	OCREF 的互补（not OCREF）+ polarity + dead-time OCN_EN=1
0	0	X	0	0	输出禁止(与定时器断开) 异步的: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 如果时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，OCx=OISx 和 OCxN=OISxN。	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	关闭状态（输出使能且为无效电平） 异步的：OCx=CCxP, OCx_EN=1，OCxN=CCxNP, OCxN_EN=1 若时钟存在：经过一个死区时间后，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平，OCx=OISx 和 OCxN=OISxN	
	1		1	0		
	1		1	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： 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 以及 AFIO 寄存器。

23.8.8. TIM16_17 计算器(TIMx_CNT)

Address offset： 0x24

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	CNT[15:0]	RW	0	计数器的值

23.8.9. TIM16_17 预分频器 (TIMx_PSC)

Address offset： 0x28

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	PSC[15:0]	RW	0	预分频器的值 计数器的时钟频率（CK_CNT）等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器 被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

23.8.10. TIM16_17 自动重新加载寄存器 (TIMx_ARR)

Address offset： 0x2c

Reset value:0x0000 FFFF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	ARR[15:0]	RW	FFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

23.8.11. TIM16_17 重复计数器寄存器(TIMx_RCR)

Address offset： 0x30

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	REP[7:0]							

-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved			保留，一直为 0
7:0	REP[7:0]	RW	0	周期计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；

23.8.12. TIM16_17 捕获/比较寄存器 1(TIMx_CCR1)

Address offset： 0x34

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/RO															

Bit	Name	R/W	Reset Value	Function
15:0	CCR1[15:0]	RW/RO	0	捕获/比较 1 的值 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载特性，其始终装入当前寄存器中。 否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器包含了与计数器 TIMx_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

23.8.13. TIM16_17 刹车和死区寄存器(TIMx_BDTR)

Address offset： 0x44

Reset value: 0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	**AOE**	**BKP**	**BKE**	**OSSR**	**OSSI**	**LOCK[1:0]**		**DTG[7:0]**							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

注释：根据锁定设置，AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0]位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

Bit	Name	R/W	Reset Value	Function
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的值，可由软件清 0 或自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位（TIMx_CCER 寄存器的 CcxE、 CcxNE 位），则开启 OC 和 OCN 输出。
14	AOE	RW	0	自动输出使能 0： MOE 只能被软件置 1； 1： MOE 能被软件置 1 或在下一个更新事件自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入（BRK 及 BRK_ACTH）； 1：开启刹车输入（BRK 及 BRK_ACTH）。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIM1_CCER)）。 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：当定时器不工作时，一旦 CcxE=1 或 CcxNE=1，开启 OC/OCN 输出并输出无效电平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIMx_CCER)）。

Bit	Name	R/W	Reset Value	Function
				0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1： 当 定 时器不 工 作时， 一旦 CcxE=1 或 CcxNE=1，OC/OCN 首 先 输出 其 空闲 电 平。 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	00	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，TIMx_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，TIMx_CCMRx 寄存器的 OCxM/OCxPE 位）； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	0000 0000	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] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg， Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、 2 或 3，则这些位不能被修改。

23.8.14. TIM16_17 DMA 控制寄存器(TIMx_DCR)

Address offset： 0x48

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Res	Res	Res	DBL[4:0]					Res			DBA[4:0]				
-	-	-	RW	RW	RW	RW	RW	-	-	-	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:13	Reserved			保留，一直为 0
12:8	DBL[4:0]	RW	0 0000	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器的地址进行读或写时，定时器则进行一次连续传送），即：定义被传送的字节数目： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 10001：18 次传输 例：我们考虑这样的传输： DBL=7， DBA=TIM2_CR1 - 如果 DBL=7， DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引), 其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	RW	0	保留，始终读为 0
4:0	DBA[4:0]	RW	0 0000	DBA[4:0]: DMA 基地址 这些位定义了 DMA 在连续模式下的基地址（当对 TIMx_DMAR 寄存器的地址进行读或写时）， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000： TIMx_CR1， 00001： TIMx_CR2， 00010： TIMx_SMCR， ......

23.8.15. TIM16_17 连续模式的 DMA 地址(TIMx_DMAR)

Address offset: 0x4C

Reset value:0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAB[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15：0	DMAB[31:0]	RW	0	DMA 连续传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址的寄存器的存取操作： TIMx_CR1 地址 + DBA + DMA 指针，其中： “TIMx_CR1 地址”是控制寄存器 1 的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 指针”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

注：在使用 DMA 连续传输功能时，必须将 DMA 中对应通道的 CNDTR 寄存器的值与 TIMx_DCR 寄存器中 DBL 的值对应起来，否则该将不能正常使用。

23.8.16. TIM16_17 寄存器映像

偏移	寄存器	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
000h	TIMx_CR1	保留																						CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
	复位值																							0	0	0	0	0	0	0	0	0	0
004h	TIMx_CR2	保留																	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	保留	CCPC
	复位值																		0	0	0	0	0	0	0	0	0	0	0	0	0		0
008h	TIMx_SMCR	保留																ETP	ECE	ETPS[1:0]		EFT[3:0]				MSM	TS[2:0]			保留	SMS[2:0]		
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
00Ch	TIMx_DIER	保留																	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	复位值																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	TIMx_SR	保留																			CC4OF	CC3OF	CC2OF	CC1OF	保留	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
	复位值																				0	0	0	0		0	0	0	0	0	0	0	0
014h	TIMx_EGR	保留																								BG	TG	COM	CC4G	CC3G	CC2G	CC1G	UG
	复位值																									0	0	0	0	0	0	0	0
018h	TIMx_CCMR1 输出比较模式	保留																OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR1 输入捕获模式	保留																IC2F[3:0]				IC2 PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1 PSC[1:0]		CC1S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01Ch	TIMx_CCMR2 输出比较模式	保留																OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	TIMx_CCMR2 输入捕获模式	保留																IC4F[3:0]				IC4 PSC[1:0]		CC4S[1:0]		IC3F[3:0]				IC3 PSC[1:0]		CC3S[1:0]	
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
020h	TIMx_CCER	保留																		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
	复位值																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
024h	TIMx_CNT	保留																CNT[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
028h	TIMx_PSC	保留																PSC[15:0]															
	复位值																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

24.低功耗定时器(LPTIM)

24.1. 简介

LPTIM 是一款 16 位定时器。LPTIM 将系统从低功耗模式中唤醒的能力使得它适合于实现低功耗应用。

LPTIM 可以使用高频时钟（PCLK）和低频时钟（LSI/LSE）计数，可提供所需的功能和性能，同时将功耗降至最低。

24.2. LPTIM 主要特性

- 16 位向上计数器
- 3 位预分频器，具有 8 个可能的分频因子（1、2、4、8、16、32、64、128）
- 可选时钟
  - 内部时钟源:LSE，LSI 或 APB 时钟
- 16 BIT ARR 可重载寄存器
- 连续/单次模式

24.3. 低功耗定时器（LPTIM）功能描述

24.3.1. LPTIM 框图

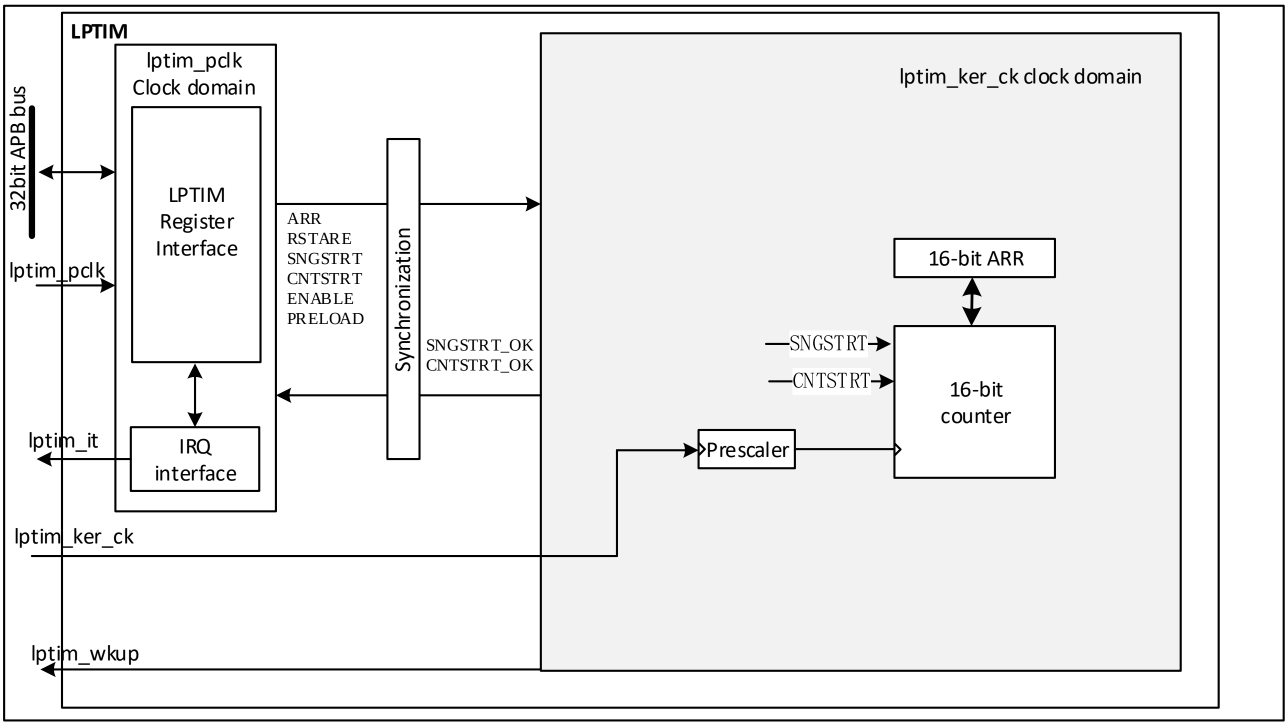


图 24-1 低功耗定时器框图

24.3.2. LPTIM 复位和时钟

LPTIM 可以使用多个时钟源进行计时。

通过 RCC 模块，可以使用内部时钟信号对其进行时钟控制（该时钟信号可以在 APB、LSI、LSE 源中进行选择）。

24.3.3. 预分频器

LPTIM 16 位计数器，由一个可配置的 2 次方预分频器控制驱动。预分频器分频比由 PRESC[2:0]控制。

下表列出了所有情况：

表 24-24 预分频系数

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

24.3.4. 工作模式

LPTIM 具有两种如下工作模式（仅支持单次计数）。

- **单次模式：**定时器从一个触发事件开始，当达到 ARR 值时停止。

要使能单次计数，SNGSTRT 位必须置 1。

一个新的触发事件将重新启动计时器。在计数器启动之后，并到达 ARR 之前的任何触发事件都将被忽略。

- **连续模式：**计时器自由运行，从触发事件开始运行，直到计时器被 disable 掉才停止。

要使能连续计数，LPTIM_CR.CNTSTRT 位必须置 1。

设置 LPTIM_CR.CNTSTRT 将启动计数器进行连续计数。

可以“on the fly”地从单次模式转变为连续模式：

- 如果之前选择了连续模式，则置位 LPTIM_CR.SNGSTRT 会将 LPTIM 切换到单次模式。计数器（如果激活）一到达 ARR 就会停止。
- 如果先前选择了单次模式，则置位 LPTIM_CR.CNTSTRT 会将 LPTIM 切换到连续模式。计数器（如果激活）一到达 ARR 就会重新启动计数。

24.3.5. 寄存器更新

PRELOAD 位控制 LPTIM_ARR 寄存器的更新方式：

- 当 PRELOAD 位被复位为“0”：LPTIM_ARR 寄存器在任何写访问后立即更新。
- 当 PRELOAD 位被设为“1”时：如果定时器已经启动，则 LPTIM_ARR 将在下一个 LPTIM 更新事件同步进行更新。

LPTIM APB 接口和 LPTIM Kernel 逻辑使用不同的时钟，因此在 APB 写入,和写入的值被应用到计数器比较器时，存在一定的延迟。在此延迟周期内，必须避免对这些寄存器进行任何额外的写操作。

LPTIM_ISR 寄存器中的 ARROK 标志，指示对 LPTIM_ARR 寄存器的写操作是否完成。

对 LPTIM_ARR 寄存器进行写操作后，只有在前一次写操作完成后，才能对同一寄存器执行新的写操作。在 ARROK 标志位置位之前的任何连续写入都将导致不可预测的结果。

24.3.6. 计数器模式

LPTIM 仅支持内部时钟计数,使能计时器。

LPTIM_CR 寄存器中的 ENABLE 位用于使能/不使能 LPTIM 内核逻辑。置位 ENABLE 位后，需要延迟两个计数器时钟才能使能 LPTIM。

仅当 LPTIM 禁用时，才能修改 LPTIM_CFGR 和 LPTIM_IER 寄存器。

24.3.7. 计数器复位

为了将 LPTIM_CNT 寄存器的内容复位，提供复位机制：

同步复位机制：

同步复位由 LPTIM_CR.CONURST 位控制。将 COUNTRST 位置 1 后，复位信号送给 LPTIM 的 Kernel 时钟域。所以需要注意在复位起作用前，LPTIM Kernel 时钟域的逻辑电路已过去几个时钟周期了。这将使从复位触发到复位生效，LPTIM 计数器多计了额外几个数。

由于 COUNTRST 是 APB 时钟域的，而 LPTIM 计数器位于 LPTIM Kernel 时钟域，所以当向 COUNTRST 位写入 1 时，需要 Kernel 时钟的 3 个时钟周期的延迟来同步来自 APB 时钟域的复位信号。

24.3.8. 调试模式（debug mode）

当芯片进入 debug 模式，取决于 DBG 模块的 DBG_LPTIM_STOP 位的设定，LPTIM 或者继续正常工作，或者停止工作。

24.4. LPTIM 低功耗模式

表 24-24 LPTIM 不同低功耗模式的区别

模式	描述
Sleep	功能无影响. LPTIM 中断（使能后）会退出 sleep 模式.
Stop	LSE/LSI 时钟存在时功能无影响。 LPTIM 中断（使能后）会退出 stop 模式.

24.5. LPTIM 中断

如果下列事件在 LPTIM_IER 寄存器内使能，则这些事件将生成中断/唤醒事件：

- 自动重新加载匹配

注意:如果在 LPTIM_ISR 寄存器（状态寄存器）中的相应标志置 1 后，LPTIM_IER 寄存器（中断使能寄存器）中的相应位被置 1，则不产生中断。

表 24-24 LPTIM 中断事件

中断事件	**描述**
自动重载匹配	当计数器寄存器的内容(LPTIM_CNT)与自动重新加载寄存器的内容匹配(LPTIM_ARR)，中断标志置位

中断事件	描述
自动重载寄存器更新 OK	当对LPTIM_ARR寄存器的写操作完成，中断标志被置位

24.6. LPTIM 寄存器

24.6.1. LPTIM 中断和状态寄存器 (LPTIM_ISR)

地址偏移：0x000

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	0	
4	ARROK	R	0	自动重载寄存器更新 OK。 ARROK 由硬件设置，以通知应用程序 APB 总线对 LPTIM_ARR 的写操作已成功完成。向 LPTIM_ICR.ARROKCF 写入 1 可清除 ARROK 标志.
1	ARRM	R	0	自动重载匹配 ARRM 由硬件设置，通知应用程序 LPTIM_CNT 寄存器值匹配 LPTIM_ARR 寄存器的值。向 LPTIM_ICR 寄存器的 ARRMCF 位写入 1 可清除 ARRM 标志
0	Reserved			

24.6.2. LPTIM 中断清除寄存器 (LPTIM_ICR)

地址偏移：0x004

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：7	Reserved	-	-	Reserved
4	ARROKCF	RW	0	自动重载寄存器更新 OK 清除标志 向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARROK 标志
3：2	Reserved	-	-	Reserved
1	ARRMCF	RW	0	自动重载匹配清除标志 向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARRM 标志
0	Reserved	-	-	

24.6.3. LPTIM 中断使能寄存器 (LPTIM_IER)

地址偏移：0x008

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	Reserved
4	ARROKIE	RW	0	自动重载寄存器更新 OK 中断使能。 0:ARROK 中断禁用 1:ARROK 中断使能
3：2	Reserved	-	-	Reserved
1	ARRMIE	RW	0	自动重载匹配中断使能 0:ARRM 中断禁用 1:ARRM 中断使能
0	Reserved	-	-	Reserved

24.6.4. LPTIM 配置寄存器 (LPTIM_CFGR)

地址偏移：0x00C

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	Reserved
22	PRELOAD	RW	0	寄存器更新模式 预加载位控制 LPTIM_ARR 寄存器更新模式 0:每次 APB 总线写访问后更新寄存器 1:寄存器在当前 LPTIM 周期结束时更新
21:12	Reserved	-	-	Reserved
11:9	PRESC[2:0]	RW	0	时钟预分频器 PRESC 位配置预分频器分频系数。它可以是下列分部中的一个因素: 000:/1 001:/2 010:/4 011:/8 100:/16 101:/32

Bit	Name	R/W	Reset Value	Function
				110:/64 111:/128
8:0	Reserved	-	-	Reserved

24.6.5. LPTIM 控制寄存器 (LPTIM_CR)

地址偏移：0x010

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R STARE	COUNT RST	CNT STRT	SNG STRT	EN ABLE
											RW	RS	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	Reserved
4	RSTARE	RW	0	读取后复位使能 此位由软件置 1 和清 0。当 RSTARE 设置为“1”时，对 LPTIM_CNT 的任何读取访问寄存器将异步重置 LPTIM_CNT 寄存器内容。
3	COUNTRST	RS	0	计数器复位。 该位由软件置 1，硬件清 0。设置为“1”时，此位将触发 LPTIM_CNT 计数寄存器同步复位。由于此复位的同步特性，它只需要在同步延迟 3 个 LPTIM 内核时钟周期之后释放（LPTIM 内核时钟可能是与 APB 时钟不同）。 注：在 COUNTRST 已被硬件清除为“0”之前，软件绝不能将其设置为“1”.因此，软件在尝试将其设置为“1”之前，应检查 COUNTRST 位是否已清零为“0”
2	CNTSTRT	RW	0	定时器启动连续模式。 该位由软件置位，该位置 1 将启动连续模式下的 LPTIM。如果在进行单次计数模式计数时该位被置 1，则定时器不会在下一个 LPTIM_ARR 和 LPTIM_CNT 寄存器匹配的脉冲模式计数时停止。LPTIM 计数器保持在连续模式下计数。 注：仅当 LPTIM 使能时，此位才能置 1。它将由硬件自动重置。
1	SNGSTRT	RW	0	LPTIM 启动单次模式。 该位由软件置位，由硬件清零。该位置 1 将以单脉冲模式启动 LPTIM。 注：仅当 LPTIM 使能时，此位才能置 1。它将由硬件自动复位。
0	ENABLE	RW	0	LPTIM 使能位,由软件设置和清零 0:LPTIM 禁用 1:LPTIM 使能

24.6.6. LPTIM 自动重装载寄存器 (LPTIM_ARR)

地址偏移：0x018

复位值： 0x0000 0001

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15：0	ARR	RW	0x0001	自动重新加载值 ARR 是 LPTIM 的自动重载值 当 LPTIM 使能后才能更新该寄存器

24.6.7. LPTIM 计数寄存器 (LPTIM_CNT)

地址偏移：0x01C

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15：0	CNT	R	0	计数器值 当 LPTIM 以异步时钟运行时，读取 LPTIM_CNT 寄存器可能返回不可靠的值。因此在这种情况下，有必要执行两次连续的读访问并验证返回的两个值是否相同。当两次连续读取访问的值相等时，可以认为读取访问是可靠的。

24.6.8. LPTIM 寄存器映像

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	LPTIM_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARROK	Res.	Res.	ARRM	Res.
	Reset value																												0			0	
0x04	LPTIM_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARROKCF	Res.	Res.	ARRMCF	Res.
	Reset value																												0			0	
0x08	LPTIM_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARROKIE	Res.	Res.	ARRMIE	Res.

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
	Reset value																												0			0	
0x0C	LPTIM_CFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRELOAD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESC [2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value										0											0	0	0									
0x10	LPTIM_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	COUNTRST	CNTSTRT	SNGSTRT	ENABLE
	Reset value																												0	0	0	0	0
0x18	LPTIM_ARR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	LPTIM_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25. 独立看门狗（IWDG）

25.1. 简介

芯片内集成了一个 Independent watchdog（简称 IWDG），该模块具有提高安全级别、时序精确及灵活使用的特点。IWDG 检测并解决由于软件失效造成的功能混乱，并在计数器达到指定的 timeout 值时触发系统复位。

IWDG 由 LSI 提供时钟，这样即使主时钟 Fail，也能保持工作。

IWDG 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。

25.2. IWDG 主要特性

- Free-running 向下计数器
- 时钟由独立的 RC 振荡器提供(可在 STOP 模式下工作)
- 支持 CPU 配置 IWDG 模块 start 后，RCC 模块自动使能 LSI 作为 IWDG 时钟。
- 看门狗被激活后，则在计数器计数至 0x000 时产生复位

25.3. IWDG 功能描述

25.3.1. IWDG 框图

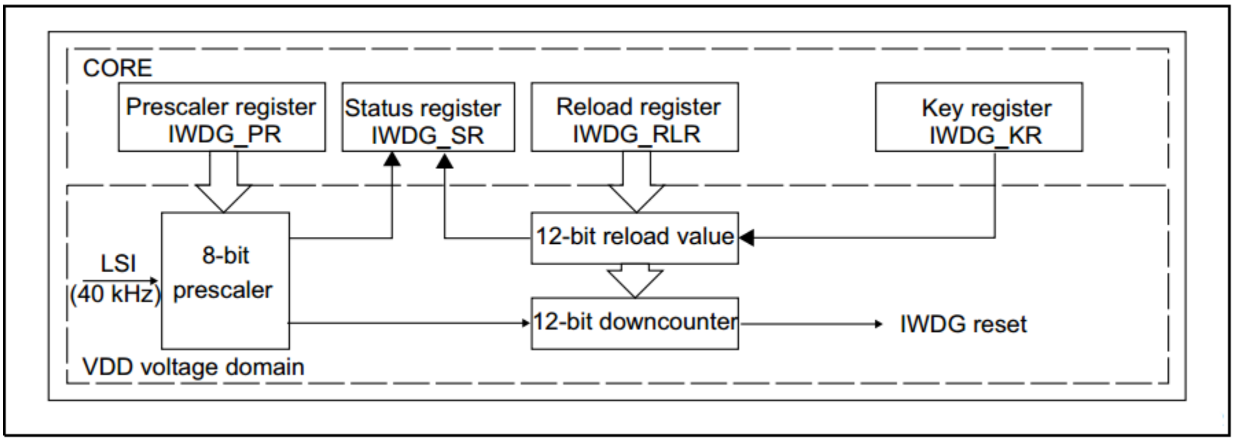


图 25-1 IWDG 框图

注：看门狗功能处于 VDD 供电区，即在停机和待机模式时仍能正常工作。

在 KEY 寄存器(IWDG_KR)中写入 0xCCCC，开始启用独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号(IWDG_RESET)。

无论何时，只要 KEY 寄存器(IWDG_KR) 中被写入 0xAAAA， IWDG_RLR 中的值就会被重新加载到计数器中从而避免产生 IWDG 复位 。

表 25-1 看门狗超时时间 32kHz 的输入时钟(LSI)

预分频系数	PR[2:0]位	最短时间(ms) RL[11:0]=0x000	最长时间(ms) RL[11:0]=0xFFF
/4	0	0.125	511.875
/8	1	0.25	1023.75
/16	2	0.5	2047.5
/32	3	1	4095
/64	4	2	8190
/128	5	4	16380
/256	（6 或 7）	8	32760

注：这些时间是按照 32kHz 时钟给出。实际上，MCU 内部的 RC 频率会在 25kHz 到 40kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。通过对 LSI 进行校准可获得相对精确的看门狗超时时间。

25.3.2. 硬件看门狗

如果上电装载的选项字节（option bytes）设置了打开硬件 watchdog，则 IWDG 上电被自动使能，并且如果在计数器计数到终值之前，IWDG key 寄存器没被软件改写，则产生复位信号。

25.3.3. 寄存器保护

对寄存器 IWDG 预分频、IWDG 重装载的写访问是被保护的。为了修改他们，用户必须先向 IWDG Key 寄存器写 0x0000 5555。对这些寄存器的写其他数将破坏时序，如写 0x0000AAAA 加载，寄存器将被再次保护。

如果预分频寄存器、重装载寄存器的值正在更新，状态寄存器是会体现出来的。

25.3.4. 调试模式与 STOP 模式

本功能为系统支持 DBG_MCU 时才存在。

如果 CPU 进入调试模式，IWDG 继续计数还是进入 stop 模式，取决于 DBG 模块中 DBG_IWDG_STOP 的配置。

STOP 模式为在 option byte 中有 IWDG_STOP 位，可以控制住 CPU 在进入 deepsleep 模式时。IWDG 继续正常计数还是进入 deepsleep 模式，取决于 FMC 中的 option byte 里的 IWDG_STOP 的配置。

Option byte 中关于 IWDG_STOP 的配置如下：

设置 iwdg 在 stop 模式下定时器运行状态

0： freeze 定时器

1： 正常运行

Option byte 中默认 IWDG_STOP 为 “ 1 ”

25.4. IWDG 寄存器

可以用半字(16 位)或字(32 位)的方式操作这些外设寄存器。

25.4.1. 密钥寄存器 (IWDG_KR)

Address offset:0x00

Reset value:0x0000 0000（由 Standby 模式复位）

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	KEY[15:0]	W	0x00	Key 值。 软件必须以一定的时间间隔向该寄存器写入 0xAAAA，否则，当计数器计数到 0 时，看门狗会产生复位。 0x5555：表示允许访问 IWDG_PR、IWDG_RLR 寄存器； 0xCCCC：表示启动 IWDG（如果选择了硬件看门狗则不受此命令字限制）。

25.4.2. 预分频寄存器 (IWDG_PR)

Address offset:0x04

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2:0	PR[2:0]	RW	0	预分频值。 通过配置该寄存器选择计数器时钟的预分频值。 要改变该寄存器，IWDG_SR 寄存器的 PVU 必须为 0。 000：4 分频； 001：8 分频； 010：16 分频； 011：32 分频； 100：64 分频； 101：128 分频； 110：256 分频； 111：256 分频；

25.4.3. 重装载寄存器 (IWDG_RLR)

Address offset:0x08

Reset value:0x0000 0FFF（由 Standby 模式复位）

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	RL[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	Reserved
11:0	RL[11:0]	RW	1	IWDG 计数器重装载值。 当向 IWDG_KR 寄存器写入 0xAAAA 时，RL 值会传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此 RL 值和时钟预分频值来计算。 只有当 IWDG_SR.RVU=0 时，才能对寄存器进行修改。 此外需要注意，当向 RLR 寄存器写入重装载值后需要等待三个 LSI 时钟才能读出。

25.4.4. 状态寄存器 (IWDG_SR)

Address offset:0x0C

Reset value:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	Reserved
1	RVU	R	0	看门狗计数器重装值更新。 必须要软硬件启动 IWDG 后才可让 RVU 正常工作 该位由硬件置 1，表明重装载值正在更新。当重装载值更新结束后，此位由硬件清零。
0	PVU	R	0	看门狗预分频值更新。 必须要软硬件启动 IWDG 后才可让 PVU 正常工作 该位由硬件置 1，表明预分频值正在更新。当预分频值更新结束后，此位由硬件清零。

注：在更新 IWDG_PR、IWDG_SR.RLR 前，要分别等待 IWDG_PVU、IWDG_SR.RVU 为 0。但在更新 IWDG_PR、IWDG_RLR 后，不必再等待 IWDG_SR.PVU、IWDG_SR.RVU 为 0，可继续执行下面的代码。

25.4.5. IWDG 寄存器映像

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	IWDG_KR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	KEY[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR[2:0]		

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
																															0	0	0
0x08	IWDG_RLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RL[11:0]											
																						1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RVU	PVU
																															0	0	0

26. 窗口看门狗（WWDG）

26.1. 简介

System window watchdog(WWDG)被用作发现一个软件 fault 的出现。通常这类软件 fault 产生于外部干扰，或者未预测到的逻辑条件，该类 fault 引起应用程序终止正常的操作顺序。Watchdog 在一个编程的时间周期到期时，产生一个复位信号，除非程序在 T6 位清除之前，刷新向下计数器的内容。

如果 7 位向下计数器的值在向下计数器已经达到 window 寄存器值之前被刷新，也会产生复位。这暗示了计数器必须在一个被限制的窗口内被刷新。

WWDG 时钟来自 APB 时钟的分频，有一个可配置的时间窗口（该窗口可编程，用来发现异常的早或者晚的应用行为）。

WWDG 最适合于需要 Watchdog 在一个精准的时序窗口响应的场景。

26.2. WWDG 主要特性

- 可编程的自由运行递减计数器
- 有条件的复位
  - 当递减计数器的值小于 0x40，(若看门狗被启动)则产生复位。
  - 当递减计数器在窗口外被重新装载，(若看门狗被启动)则产生复位。
- 早唤醒中断（Early wakeup interrupt）：当向下计数器等于 0x40 时被触发（如果该功能 enable，并且 watchdog 被激活）

26.3. WWDG 功能描述

如果 WWDG 被激活（WDGA bit 被置位），并且当 7 位向下计数器（T[6:0]位）从 0x40 减小到 0x3F（T6 被清掉），则引起复位。如果软件再装载计数器时，计数器值大于存储器 window 寄存器的值，产生复位。

应用程序必须在正常操作期间以例行的间隔，写 WWDG_CR 寄存器，以防止复位。仅当计数器值小于 window 寄存器的值并且高于 0x3F 时，上述写操作必须发生。存在 WWDG_CR 寄存器的值必须在 0xFF 和 0xC0 之间。

26.3.1. WWDG 架构框图

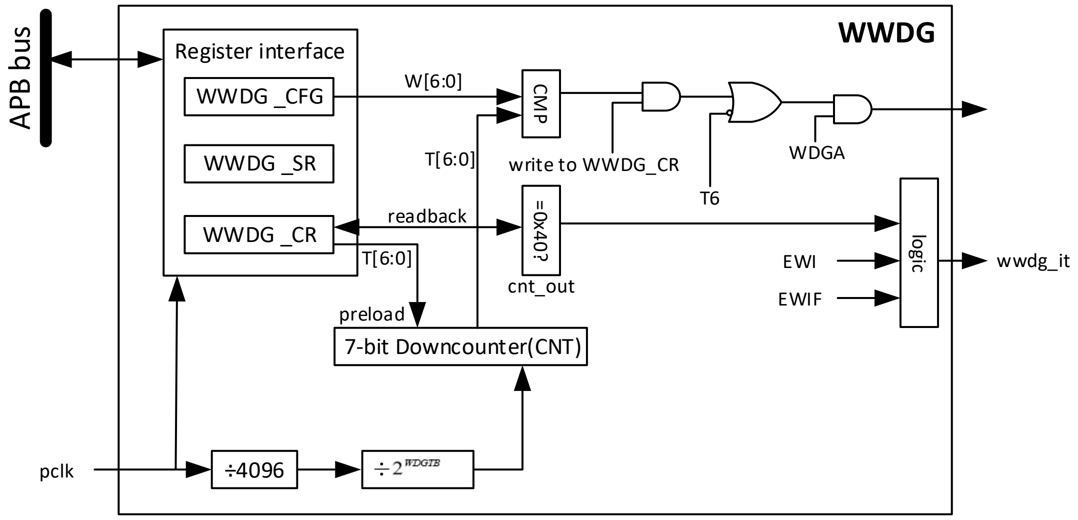


图 26-1 window watchdog 架构框图

26.3.2. 启动看门狗

在系统复位后，看门狗总是处于关闭状态，设置 WWDG_CR 寄存器的 WDGA 位或者硬件 option byte 都能能够开启看门狗，随后它不能再被关闭，除非发生复位。

在 option byte 中有 wwdg_sw 寄存器，也可启动看门狗，其值为：

0：硬件 watchdog

1：软件 watchdog

另外硬件启动与软件启动只要设置了其中一种，便会启动看门狗。

26.3.3. 控制递减计数器

递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时， 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 WWDG_CR 寄存器时，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且大于 0x3F 时被重新装载， 图 23-2 描述了窗口寄存器的工作过程。

另一个重装载计数器的方法是利用早期唤醒中断(EWI)。设置 WWDG_CFR 寄存器中的 WEI 位开启该中断。当递减计数器到达 0x40 时，则产生此中断，相应的中断服务程序(ISR)可以用来加载计数器以防止 WWDG 复位。在 WWDG_SR 寄存器中写’0’可以清除该中断。

注： 可以用 T6 位产生一个软件复位(设置 WDGA 位为’1’， T6 位为’0’)。

26.3.4. 高级看门狗中断功能

提前唤醒中断（EWI）可以用在特定的安全操作，或者在复位前做数据日志的情况。通过配置 WWDG_CFR.EWI 寄存器，可以使能 EWI 功能。当递减计数器的值达到 0x40 时，EWI 中断产生，对应的中断处理程序 ISR 可以在产生复位前执行特定操作。

26.3.5. 如何编写看门狗超时程序

对 WWDG_CR 寄存器通常通过向 T6 位写 1，以避免产生复位。

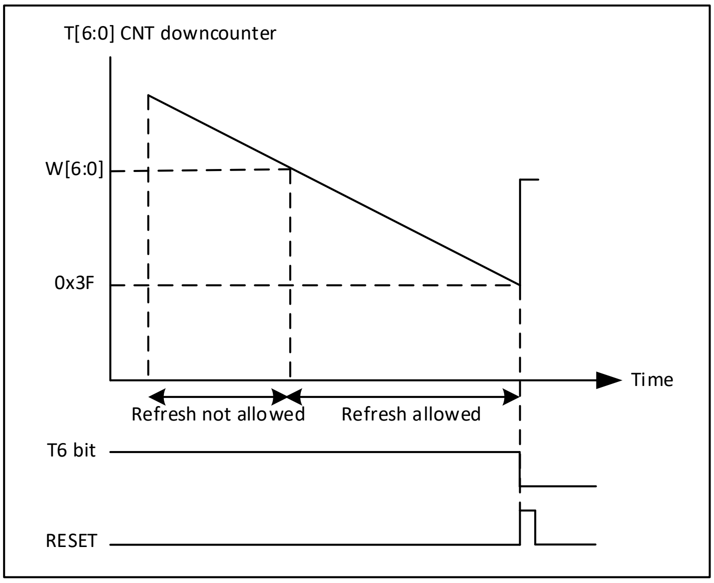


图 26-2 窗口看门狗时序图

计算 WWDG 超时值的公式如下:

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WWDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

26.3.6. 调试模式

当微控制器进入调试模式时(Cortex-M4 核心停止)，根据调试模块中的 DBG_WWDG_STOP 配置位的状态， WWDG 的计数器能够继续工作或停止。详见调试模式节。

26.4. WWDG 寄存器

26.4.1. 控制寄存器 (WWDG_CR)

Address offset:0x00

Reset value:0x0000 007F

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	Reserved
7	WDGA	RS	0	WDGA: 激活位 (Activation bit)。 此位由软件置'1'，但仅能由硬件在复位后清'0'。 当 WDGA=1 时，看门狗可以产生复位。 0：禁止; 1：使能;
6:0	T[6:0]	RW	7F	7 位计数器（MSB 至 LSB）。

Bit	Name	R/W	Reset Value	Function
				该寄存器用来存储看门狗的计数值。每（4096x2^WDGTB）个 PCLK 周期减 1.当计数值从 40h 变为 3Fh 时（T[6]变为 0），产生看门狗复位。

26.4.2. 配置寄存器 (WWDG_CFR)

Address offset:0x04

Reset value:0x0000 007F

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

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9	EWI	RS	0	提前唤醒中断。 该位置 1，则当计数器值达到 40h 时，即产生中断。 此中断只能由硬件在复位后清除。
8：7	WDGTB[1:0]	RW	0	时基（timer base）。 预分频器的时基设置如下: 00：CK 计数器时钟（PCLK 除以 4096）除以 1； 01：CK 计数器时钟（PCLK 除以 4096）除以 2； 10：CK 计数器时钟（PCLK 除以 4096）除以 4； 11：CK 计数器时钟（PCLK 除以 4096）除以 8；
6：0	W[6:0]	RW	7F	7 位窗口值。 该寄存器包含了用来与递减计数器进行比较用的窗口值。

26.4.3. 状态寄存器 (WWDG_SR)

Address offset:0x08

Reset value:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31：1	Reserved	-	-	Reserved
0	EWIF	RC_W0	0	提前唤醒中断标志。

Bit	Name	R/W	Reset Value	Function
				当计数器值达到 40h，此位由硬件置 1.软件写 0 清零，写 1 无效。 当中断未被使能时，该位也置 1.

26.4.4. WWDG 寄存器映像

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	WWDG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDGA	T[6:0]						
	Reset value																									0	1	1	1	1	1	1	1
0x04	WWDG_CFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWI	WDGTB[1:0]		W[6:0]						
	Reset value																							0	0	0	1	1	1	1	1	1	1
0x08	WWDG_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EWIF
	Reset value																																0

27.实时时钟(RTC)

27.1. 简介

实时时钟（real time clock）是一个独立的定时器。RTC 模块拥有一组连续计数的计数器，在相应软件配置下，可提供时钟日历的功能。修改计数器的值可以重新设置系统当前的时间和日期。

27.2. RTC 主要特性

- 可编程的预分频系数：分频系数最高为 2^{20}
- 32 位的可编程计数器，可用于较长时间段的测量
- 2 个单独的时钟：用于 APB1 接口的 PCLK1 和 RTC 时钟(RTC 时钟的频率必须小于 PCLK1 时钟频率的四分之一以上)
- 可以选择以下三种 RTC 的时钟源：
  - HSE 时钟除以 128
  - LSI 振荡器时钟
  - LSE 振荡器时钟
- 2 个独立的复位类型：
  - APB1 接口由系统复位；
  - RTC 核心(预分频器、闹钟、计数器和分频器)只能由备份域复位。
- 3 个专门的可屏蔽中断：
  - 闹钟中断，用来产生一个软件可编程的闹钟中断
  - 秒中断，用来产生一个可编程的周期性中断信号(最长可达 1 秒)
  - 溢出中断，指示内部可编程计数器溢出并回转为 0 的状态

27.3. RTC 功能描述

27.3.1. 总览

RTC 由两个主要部分组成(参见下图)。第一部分(APB1 接口)用来和 APB1 总线相连。此单元还包含一组 16 位寄存器（RTC_CR，实际分散在两个地址的寄存器），可通过 APB1 总线对其进行读写操作。APB1 接口由 APB1 总线时钟驱动，用来连接 APB1 总线。

另一部分(RTC 核心)由一组可编程计数器组成，分成两个主要模块。

第一个模块是 RTC 的预分频模块，它可编程产生最长为 1 秒的 RTC 时间基准 TR_CLK。RTC 的预分频模块包含了一个 20 位的可编程分频器(RTC 预分频器)。如果在 RTC_CR 寄存器中设置了相应的允许位，则在每个 TR_CLK 周期中 RTC 产生一个中断(秒中断)。

第二个模块是一个 32 位的可编程计数器，可被初始化为当前的系统时间。系统时间按 TR_CLK 周期累加并与存储在 RTC_ALR 寄存器中的可编程时间相比较，如果 RTC_CR 控制寄存器中设置了相应允许位，比较匹配时将产生一个闹钟中断。

所有中断都可以作为停止模式的唤醒信号，如下图所示，在停止模式下，RTC 对应中断作为唤醒需要预先配置对应的中断使能位以及 EXTI line19 对应寄存器。

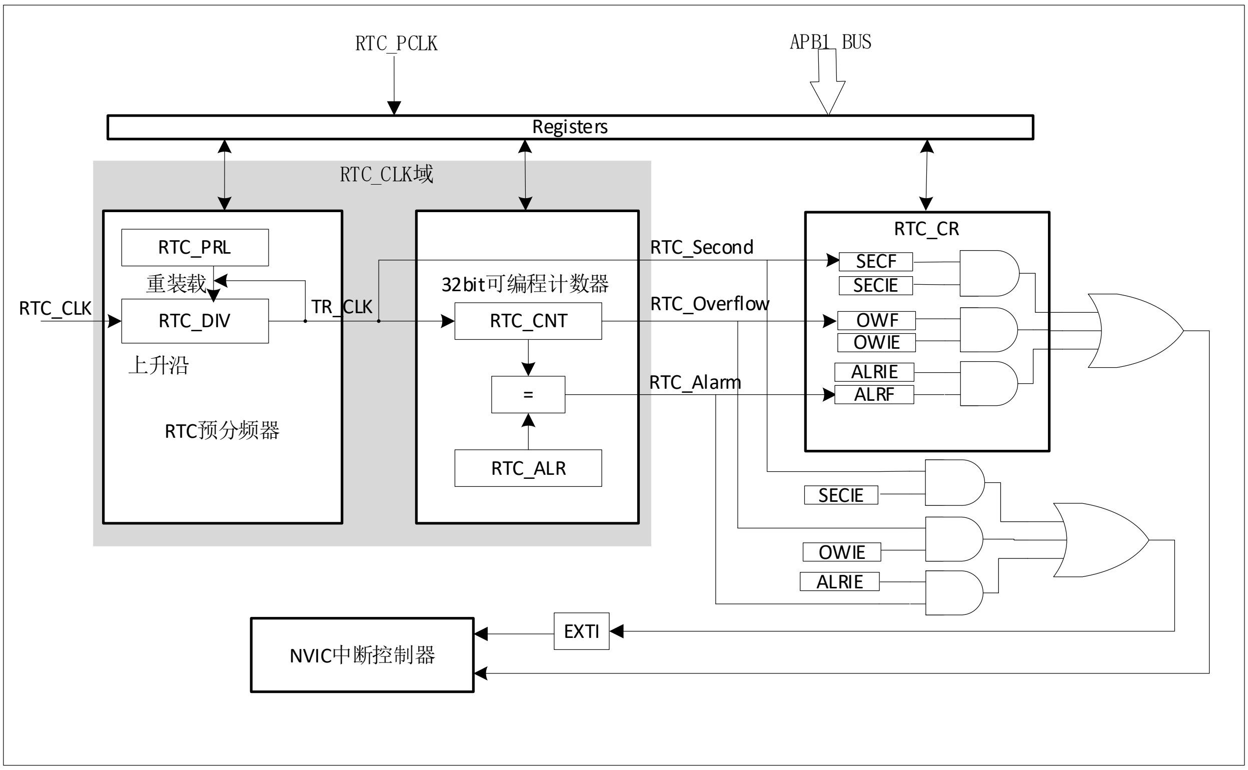


图 27-1 RTC 框图

PWR_CR1 寄存器的 DBP 位用来控制对 RTC 寄存器的写保护禁止。默认情况 DBP=0，不能对 RTC 寄存器进行写访问，软件置位 DBP 后，才能实现对 RTC 寄存器的写操作。

27.3.2. 复位 RTC 寄存器

RTC_PRL，RTC_ALR，RTC_CNT 和 RTC_DIV 寄存器仅能通过备份域复位信号复位。其他系统寄存器（RTC_CR）由系统复位或电源复位进行异步复位。

注意，当不能对 RTC 模块产生复位的复位源，不仅不能复位 RTC 模块，也不能复位送给 RTC 模块的时钟源使能信号，也不能复位送给 RTC 模块的时钟源选择控制。

27.3.3. 读 RTC 寄存器

RTC 核完全独立于 RTC APB1 接口。软件通过 APB1 接口访问 RTC 的预分频值、计数器值和闹钟值。但是，相关的可读寄存器只在于 RTC 时钟的上升沿同步到 RTC APB1 时钟后的信号有效时更新。RTC 标志也是如此的。

这意味着，如果 APB1 接口曾经被关闭，而读操作又是在刚刚重新开启 APB1 之后，则在第一次的内部寄存器更新之前，从 APB1 上读出的 RTC 寄存器数值可能被破坏了(通常读到 0)。下述几种情况下能够发生这种情形：

1） 发生系统复位或电源复位
2） 系统刚从待机模式唤醒
3） 系统刚从停机模式唤醒(这种情况计数值只是不会更新，因为 CPU 不工作，系统时钟停止，但 RTC 正常计数，计数值不会同步到 VDDD 区)

所有以上情况中， APB1 接口被禁止时(复位、无时钟或断电)RTC 核仍保持运行状态。

因此，若在读取 RTC 寄存器时，RTC 的 APB1 接口曾经处于禁止状态，则软件首先必须等待 RTC_CRL 寄存器中的 RSF 位(寄存器同步标志)被硬件置'1'。

注：RTC 的 APB1 接口不受 WFI 和 WFE 等低功耗模式的影响。

说明：

1. CPU 可读的寄存器包括 RTC_CR，RTC_CNT 和 RTC_DIV；
2. RTC_CR 寄存器为 RTC_PCLK 域，CPU 任何时候读能读到稳定值；
3. RTC_CNT 和 RTC_DIV 来源于 RTC_CLK 域。在 RTC 工作后，RTC_DIV 寄存器在每个 RTC_CLK 的上升沿更新；RTC_CNT 和来源于 RTC_CLK 时钟域的标志位同样采用跟 RTC_DIV 寄存器同样的更新信号，虽然这样不是每次更新时 RTC_CNT 的值都改变；
4. RSF 实现在 RTC_PCLK 域，在 RTC_CLK 同步到 RTC_PCLK 后的脉冲信号有效时置位；
5. RSF 仅控制 RTC_CNT 和 RTC_DIV 的读取时机（硬件不会控制）

27.3.4. 配置 RTC 寄存器

必须通过置位 RTC_CRL 寄存器的 CNF 位，进入配置模式，才能写入 RTC_PRL、RTC_CNT、RTC_ALR、BKP_RTCCR 寄存器。

另外，对 RTC 任何寄存器的写操作，都必须在前一次写操作结束后进行。可以通过查询 RTC_CR 寄存器中的 RTOFF 状态位，判断 RTC 寄存器是否处于更新中。仅当 RTOFF 状态位是’1’时，才可以写入 RTC 寄存器。

配置过程：

1) 查询 RTOFF 位，直到 RTOFF 的值变为’1’；（表明前面配置已经完成）
2) 置 CNF 值为 1，进入配置模式；（此时 RTOFF 位仍为 1，保证 CPU 在写寄存器时 RTOFF=1）
3) 对一个或多个 RTC 寄存器进行写操作；（RTOFF 在此过程仍为 1，RTC_CLK 域寄存器此步骤操作写入 buffer 寄存器）
4) 清除 CNF 标志位，退出配置模式；（硬件检测到 CNF 清零后，开始执行上一步骤中的写寄存器操作，开始写入 RTC_CLK 域的寄存器；同时 RTOFF 被清零）
5) 查询 RTOFF，直至 RTOFF 位变为’1’以确认写操作已经完成。（buffer 寄存器写入 RTC_CLK 域后置位 RTOFF）

注：仅当 CNF 标志位被清除时，写操作才能进行，这个过程至少需要 3 个 RTC_CLK 周期。(在清除 CNF 标志位后 3 个 RTC_CLK 不能重新启动配置，否则会出现配置错误（此时通过 RTOFF=0 控制））

说明：

1. 在此过程中，CPU 写寄存器时 RTOFF=1；
2. CPU 的写周期为从 CNF=1 到 CNF=0，配置其他寄存器在这两个操作之间；
3. 先将 CNF 写 1 后将 CNF 清零，这个操作清零 RTOFF；只写 CNF=0 或者写 CNF=1 后不清零不会清零 RTOFF；
4. 先将 CNF 写 1 后将 CNF 清零，这个操作启动 buffer 寄存器写 RTC_CLK 域寄存器的过程；
5. RTOFF 实现在 RTC_PCLK 域；

27.3.5. RTC 标志的设置

在每一个 RTC CLOCK 时钟周期，更改 RTC 计数器之前,硬件置位 RTC 秒标志(SECF)。在计数器到达 0x0000 之前的最后一个 RTC 时钟周期，RTC 溢出标志(OWF)被置位。

在计数器的值到达闹钟寄存器的值加 1(RTC_ALR+1)之前的 RTC 时钟周期，置位 RTC_Alarm 和 RTC 闹钟标志(ALRF)。对 RTC 闹钟的写操作必须使用下述过程之一与 RTC 秒标志同步：

（1）使用 RTC 闹钟中断，并在中断处理程序中修改 RTC 闹钟和/或 RTC 计数器。

（2）等待 RTC 控制寄存器中的 SECF 位被设置，再更改 RTC 闹钟和/或 RTC 计数器。

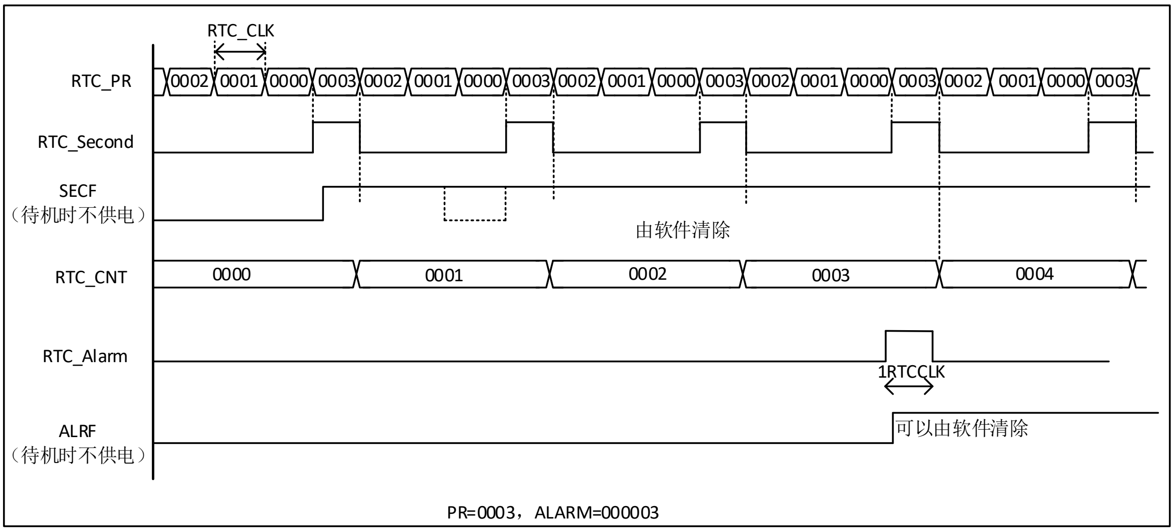


图 27-2 RTC 秒和闹钟波形图示例，PR=0003，ALARM=00004

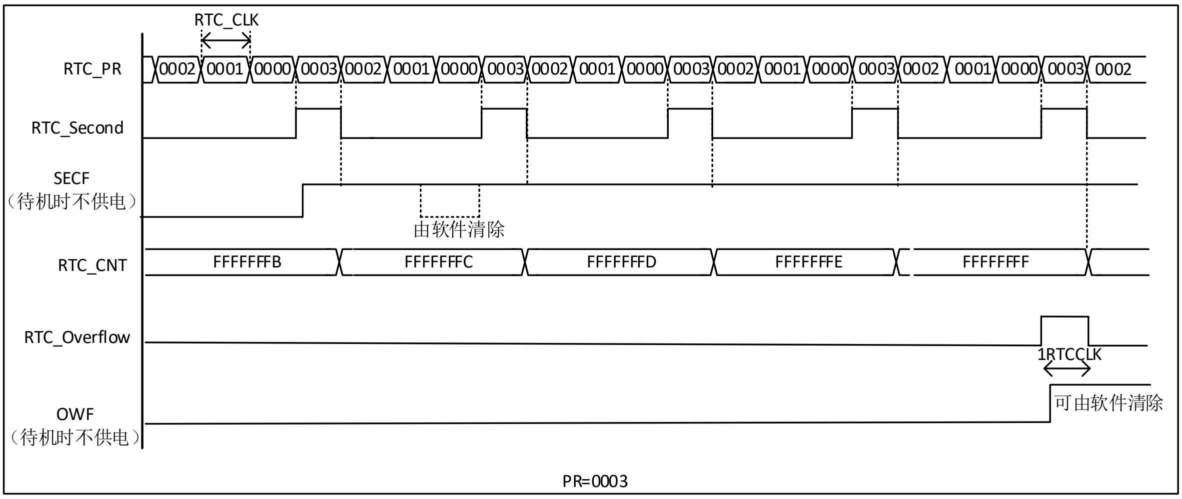


图 27-3 RTC 溢出波形图示例，PR=0003

27.3.6. RTC 校准

为了测量目的，RTC 时钟的 64 分频可以输出到 IO pin 上（PF5）。该功能是通过置位 CCO bit（RTCCR 寄存器）实现的。

通过配置 CAL[6:0] bit，时钟可以被减慢到 121 个 PPM。

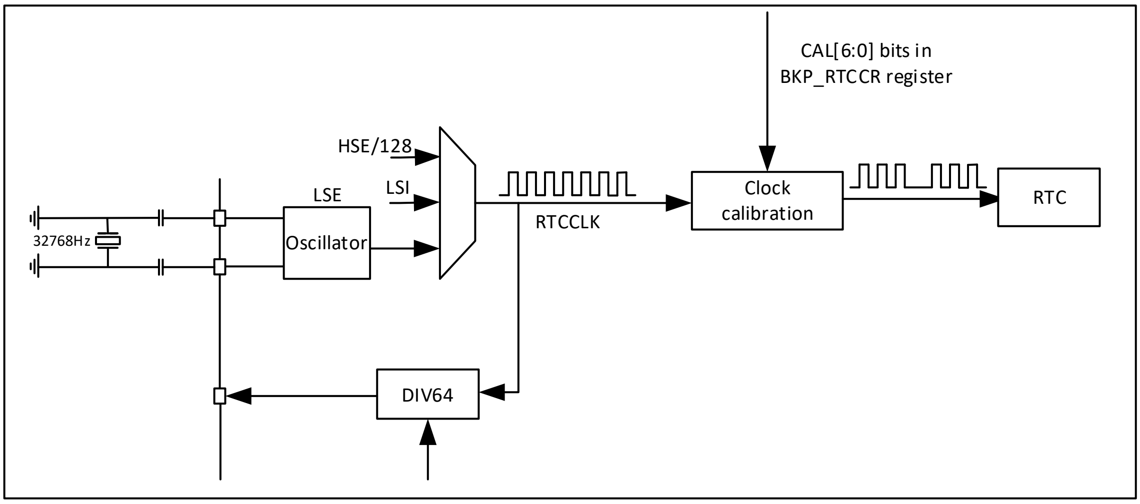


图 27-4 RTC 校准图

27.4. RTC 寄存器

27.4.1. RTC 控制寄存器 (RTC_CRH)

Address offset:0x00
Reset value:0x0000

该寄存器由系统复位复位。

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	Reserved
2	OWIE	RW	0	溢出中断允许位 0：不允许溢出中断 1：允许溢出中断
1	ALRIE	RW	0	闹钟中断允许位 0：不允许闹钟中断 1：允许闹钟中断
0	SECIE	RW	0	秒中断允许位 0：不允许秒中断 1：允许秒中断

这些位用于屏蔽中断请求。注意：在复位后，所有中断是未使能的，所以在初始化后，写 RTC 寄存器以确保没有正在挂起的中断请求是可能的。但是当外设正在完成前一次的写操作（RTOFF=0）时，是不能写 RTC_CRH 寄存器的。

该控制寄存器控制着 RTC 的功能。某些位必须使用专门的配置流程才能进行写操作。

27.4.2. RTC 控制寄存器 (RTC_CRL)

Address offset:0x04

Reset value:0x0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RTOFF	CNF	RSF	OWF	ALRF	SECF
										R	RW	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	Reserved
5	RTOFF	R	1	RTC 操作关闭 (RTC operation OFF)，该位只读。 RTC 模块利用该位来指示对其寄存器进行的最后一次操作的状态（指示操作是否完成）。 若此位为'0'，则表示无法对任何的 RTC 寄存器进行写操作。 0：上一次对 RTC 寄存器的写操作仍在进行 1：上一次对 RTC 寄存器的写操作已经完成
4	CNF	RW	0	配置标志 (Configuration flag) 此位必须由软件置'1'以进入配置模式，从而允许向 RTC_CNT、RTC_ALR 或 RTC_PRL 寄存器写入新值。 只有当此位在被置'1'，并重新由软件清'0'后，才会执行写操作。 0：退出配置模式(开始更新 RTC 寄存器) 1：进入配置模式
3	RSF	RC_W0	0	寄存器同步标志 (Registers synchronized flag)当 RTC_CNT 寄存器和 RTC_DIV 寄存器更新时，硬件置'1'该位，软件清零该位。 在 APB1 复位后，或 APB1 时钟停止后，此位必须由软件清'0'。 要进行任何的读操作之前，用户程序必须等待该位被硬件置'1'，以确保 RTC_CNT、RTC_ALR 或 RTC_PRL 已经被同步。 0：寄存器尚未被同步 1：寄存器已经被同步
2	OWF	RC_W0	0	溢出标志 (Overflow flag) 当 32 位可编程计数器溢出时，此位由硬件置'1'。如果 RTC_CRH 寄存器中 OWIE=1，则产生中断。 此位只能由软件清'0'，写'1'无效。 0：无溢出； 1：32 位可编程计数器溢出
1	ALRF	RC_W0	0	闹钟标志 (Alarm flag) 当 32 位可编程计数器达到 RTC_ALR 寄存器所设置的预定值，此位由硬件置'1'。如果 RTC_CRH 寄存器中

Bit	Name	R/W	Reset Value	Function
				ALRIE=1，则产生中断。此位只能由软件清'0'，写'1'无效。 0：无闹钟； 1：有闹钟。
0	SECF	RC_W0	0	秒标志 (Second flag) 当 32 位可编程预分频器溢出时，此位由硬件置'1'，同时 RTC 计数器加 1。 因此，此标志为分辨率可编程的 RTC 计数器提供一个周期性的信号(通常为 1 秒)。如果 RTC_CRH 寄存器中 SECIE=1，则产生中断。此位只能由软件清除，写'1'无效。 0：秒标志条件不成立 1：秒标志条件成立

RTC 的功能是被该控制寄存器控制的。当外设正在继续上一次写操作时（RTOFF=0），是不能写 RTC_CR 寄存器的。

注：

1. 任何标志位都将保持挂起状态，直到适当的 RTC_CR 请求位被软件复位，表示所请求的中断已经被接受。
2. 在复位时禁止所有中断，无挂起的中断请求，可以对 RTC 寄存器进行写操作（指从 buffer 寄存器向 RTC_CLK 域寄存器的写操作）。
3. 当 APB1 时钟不运行时， OWF、 ALRF、 SECF 和 RSF 位不被更新（无法同步）。
4. OWF、 ALRF、 SECF 和 RSF 位只能由硬件置位，由软件来清零。
5. 若 ALRF=1 且 ALRIE=1，则允许产生 RTC 全局中断。如果在 EXTI 控制器中允许产生 EXTI 线 17 中断，则允许产生 RTC 全局中断和 RTC 闹钟中断。
6. 若 ALRF=1，如果在 EXTI 控制器中设置了 EXTI 线 17 的中断模式，则允许产生 RTC 闹钟中断；如果在 EXTI 控制器中设置了 EXTI 线 17 的事件模式，则这条线上会产生一个脉冲(不会产生 RTC 闹钟中断)。

27.4.3. RTC 重装载寄存器高位 (RTC_PRLH)

PRL 寄存器保持 RTC 预分频器周期性的计数值。该寄存器是被 RTC_CR 寄存器的 RTOFF 位写保护的，只有 RTOFF=1，才允许 CPU 进行写操作（写入到 buffer 寄存器）。

Address offset:0x08

Write only

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31：4	Reserved	-	-	Reserved

Bit	Name	R/W	Reset Value	Function
3：0	PRL[19:16]	W	0	RTC 预分频装载值高位 (RTC prescaler reload value high)根据以下公式，这些位用来定义计数器的时钟频率： fTR_CLK = fRTCCLK/(PRL[19:0]+1) 注：不推荐使用 0 值，否则无法正确的产生 RTC 中断和标志位

27.4.4. RTC 重装载寄存器低位 (RTC_PRLL)

Address offset:0x0C

Write only

Reset value:0x8000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	PRL	W	0x8000	RTC 预分频装载值高位 (RTC prescaler reload value high) 根据以下公式，这些位用来定义计数器的时钟频率： fTR_CLK = fRTCCLK/(PRL[19:0]+1) 注：不推荐使用 0 值，否则无法正确的产生 RTC 中断和标志位

27.4.5. RTC 预分频分频因子寄存器高位 (RTC_DIVH)

在每个 TR_CLK 周期，RTC_PRL 寄存器的值被重装载到 RTC 预分频计数器里。用户可以通过读取 RTC_DIV 寄存器，以获得预分频计数器的当前值，而不停止分频计数器的工作，从而获得精确的时间测量。

该寄存器只读属性，当 RTC_PRL 或者 RTC_CNT 寄存器的值发生任何变化，该寄存器值将由硬件重新装载。

Address offset:0x10

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31：4	Reverved			
3：0	RTC_DIV[19:16]	R	0	RTC 时钟分频器余数高位。

27.4.6. RTC 预分频分频因子寄存器低位 (RTC_DIVL)

Address offset:0x14

Reset value:0x8000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	DIV[15:0]	R	0x8000	RTC 时钟分频器

27.4.7. RTC 计数寄存器高位 (RTC_CNTH)

RTC 模块有个 32bit 可编程的计数器，该寄存器通过两个 16bit 的寄存器访问，计数基于预分频器产生的 TR_CLK 时间基准为参考进行计数。

RTC_CNT 寄存器保持该计数器的计数值。寄存器是被写保护的，仅当 RTOFF=1 时才能进行写操作。对高 16bit 的 RTC_CNTH 或者低 16bit 的 RTC_CNTL 寄存器进行写操作，直接装载到相应的可编程计数器里，并重装载了 RTC prescaler。当读操作发生，返回计数器的当前值（系统时间）。

Address offset:0x18

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	RTC_CNT[31:16]	RW	0x0000	RTC core counter 的高 16bit 当读 RTC_CNTH 寄存器时，返回 RTC 计数器寄存器的当前值的高 16bit。只有进入配置模式才能对该寄存器进行写操作。

27.4.8. RTC 计数寄存器低位 (RTC_CNTL)

Address offset:0x1C

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	RTC_CNT[15:0]	RW	0x0000	RTC core counter 低 16bit

Bit	Name	R/W	Reset Value	Function
				当读 RTC_CNTL 寄存器时，返回 RTC 计数器寄存器当前值的低 16bit。只有进入配置模式才能对该寄存器进行写操作。

27.4.9. RTC 闹钟寄存器高位 (RTC_ALRH)

当可编程计数器（计数）达到存储在 RTC_ALR 寄存器的 32bit 值时，并产生 alarm 中断请求。该寄存器是被 RTOFF 位写保护的，只有 RTOFF=1，才允许写访问。

Address offset:0x20

Reset value:0xFFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	ALR[31:16]	RW	0xFFFF	RTC alarm 高 16bit 软件可写 Alarm 时间的高 16bit。写该寄存器必须进入配置模式。

27.4.10. RTC 闹钟寄存器高位 (RTC_ALRL)

Address offset:0x24

Reset value:0xFFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	Reserved
15：0	ALR[15:0]	RW	0xFFFF	RTC alarm 低 16bit 软件可写 Alarm 时间的低 16bit。写该寄存器必须进入配置模式。

27.4.11. RTC 时钟校准及输出配置寄存器(BKP_RTCCR)

Address offset： 0x2C

Reset value： 0x0000（仅能被 por 及 bdcr 软复位复位）

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

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	Reserved
9	ASOS	RW	0	秒/闹钟脉冲输出选择位 当 ASOE 位被置位，ASOS 位可以被用作选择 Pin 上输出是 RTC second pulse 还是 Alarm pulse 信号 0：RTC Alarm pulse 信号 1：RTC second pulse 信号
8	ASOE	RW	0	秒/闹钟脉冲输出使能位 当置位该位，则由 ASOS 位决定 pin 上输出是 RTC second pulse 还是 Alarm pulse 信号。
7	CCO	RW	0	校准时钟输出位 当 ASOE 没有置位时，可以置位 CCO 输出 RTC 时钟的 64 分频 0：无作用 1：pin 上输出 RTC 时钟的 64 分频
6：0	CAL[6:0]	RW	0	校准值 该值显示了每 2^{20} 个时钟可忽略的时钟脉冲个数，这允许 RTC 进行校准，以 1000000/2^{20}PPM 的 step 减慢时钟。 RTC clock 可以被减慢的从 0 到 121PPM。

27.4.12. RTC 寄存器映像

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	RTC_CRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OWIE	ALRIE	SECIE
	Reset value																														0	0	0
0x04	RTC_CRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	SECF
	Reset value																											1	0	0	0	0	0
0x08	RTC_PRLH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[19:16]			
	Reset value																													0	0	0	0
0x0C	RTC_PRLL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x	RTC_	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_DIV[19:16]			

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
10	DIVH																																
	Reset value																													0	0	0	0
0x14	RTC_DIVL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV[15:0]															
	Reset value																	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	RTC_CNTH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[31:16]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	RTC_CNTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	RTC_ALRH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[31:16]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x24	RTC_ALRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTC_ALR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x2C	BKP_RTCCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASOS	ASOE	CCO	CAL[6:0]						
	Reset value																							0	0	0	0	0	0	0	0	0	0

28.I2C 接口

28.1. 介绍

I2C(inter-integrated circuit)总线接口连接微控制器和串行I2C总线。它提供多主机功能，控制所有I2C总线特定的顺序、协议、仲裁和时序。支持标准（Sm）、快速（Fm）。

根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

28.2. I2C 主要特点

- Slave 和 master 模式
- 多主机功能：可以做 master，也可以做 slave
- 支持不同通讯速度
  - 标准模式（Sm）：高达 100kHz
  - 快速模式（Fm）：高达 400kHz
- 作为 Master
  - Clock 产生
  - Start 和 Stop 的产生
- 作为 slave
  - 可编程的 I2C 地址检测
  - 可响应 2 个从地址的双地址能力
  - Stop 位的发现
- 7 位/10 位寻址模式
- 通用广播（General call）
- 状态标志位
  - 发送/接收模式标志位
  - 字节传输完成标志位
  - I2C busy 标志位
- 错误标志位
  - 主机仲裁丢失
  - 地址/数据传输后的 ACK failure
  - Start/Stop 错误
  - Overrun/Underrun(时钟拉长功能禁止)
- 可选的时钟拉长功能
- 具备 DMA 能力的单字节 buffer
- 软件复位
- 模拟噪声滤波功能
- 可配置的 PEC（packet error checking）产生和验证
  - PEC 值可以在 Tx 模式下的最后一个 bytes 发送
  - 接收最后 byte 做 PEC 错误检查
- 兼容 SMBus

- 25ms 时钟低超时延时
- 10ms 主设备累积时钟低扩展时间
- 25ms 从设备累积时钟低扩展时间
- 带 ACK 控制的硬件 PEC 产生/校验
- 支持地址分辨协议(ARP)

28.3. I2C 功能描述

28.3.1. I2C 框图

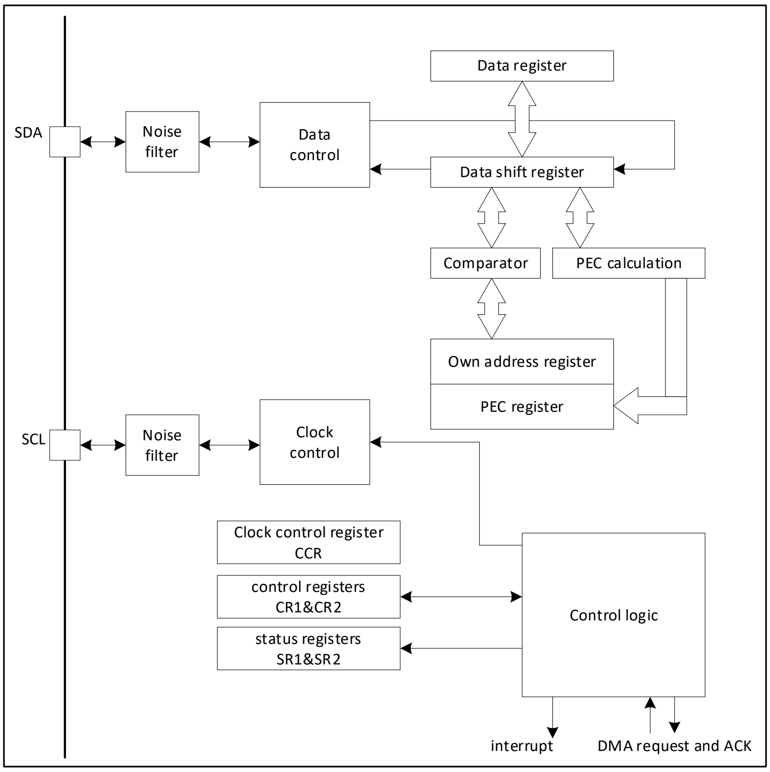


图 28-1 I2C 框图

28.3.2. 模式选择

I2C 支持以下四种模式：

1. 从发送器模式 （Slave transmitter）
2. 从接收器模式 （Slave receiver）
3. 主发送器模式（Master transmitter）
4. 主接收器模式（Master receiver）

默认模式为从模式。接口在生成起始条件后自动从从模式切换到主模式；当仲裁丢失或产生停止信号，则从主模式切换到从模式。允许多主机功能。

28.3.2.1. 通信流

作为 master，I2C 接口启动数据传输，并产生时钟信号。串行数据的传输总是以起始条件开始，并以停止条件结束。起始条件和停止条件都是在 master 模式下由软件控制产生。

作为 slave，I2C 接口能识别自己的地址(7 位)和 general call 地址。软件能够控制开启或禁止对 general call 地址的识别。

数据和地址按 8 位（字节）进行传输，高位在前。跟在 Start 条件后的 1 个字节是地址。地址只在 master 模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收方必须回送一个应答位(ACK)给发送方。参加下图。

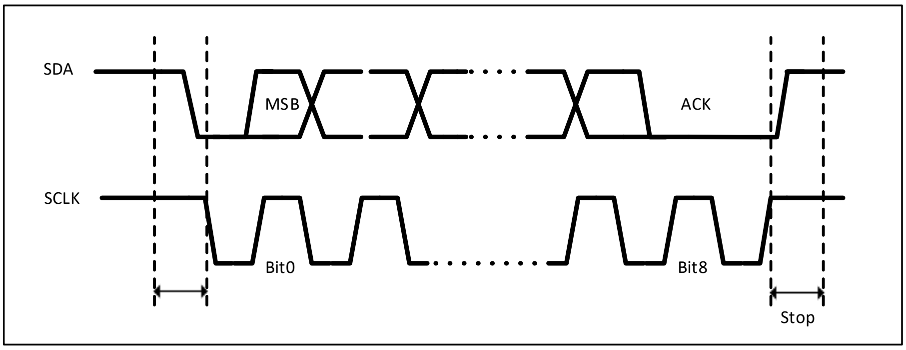


图 28-2 I2C 总线协议

软件可使能或不使能 ACK（应答）位。软件也可以选择 I2C 接口地址（7-bit、10-bit/或 general call 地址）。

28.3.3. I2C 初始化

28.3.3.1. 使能/关闭 I2C 模块

I2C 的时钟模块先要通过 RCC_APBENR1 寄存器的 I2C_EN 位打开，然后通过设定 I2C_CR1 的 PE 位使能 I2C 模块。

28.3.3.2. I2C 时序设定

为满足在 master 和 slave 模式下，准确的数据 hold 和 setup 时间，需要进行 I2C 时序的设定。这是通过写 I2C_CCR 和 I2C_TRISE 寄存器实现的。

28.3.4. I2C 从模式

默认情况下，I2C 接口总是工作在 slave 模式。从 slave 模式切换到 master 模式，需要产生一个起始条件。

为了产生正确的时序，必须在 I2C_CR2 寄存器中设定该模块的输入时钟。输入时钟的频率必须至少是：

标准模式下为：2MHz

快速模式下为：4MHz

一旦检测到起始条件，在 SDA 线上接收到的地址，被送到 shift 寄存器，并与芯片的地址 OAR1 或者 general call 地址(如果 ENGC=1)相比较。

头段或地址不匹配：

I2C 接口将其忽略并等待另一个起始条件。

地址匹配：

I2C 接口产生以下时序：

● 如果 ACK 被软件置'1'，则产生一个应答脉冲

● 硬件置位 ADDR 位，如果设置了 ITEVTEN 位，则产生中断

在从模式下 TRA 位指示当前是处于接收器模式还是发送器模式。

28.3.4.1. 从发送器

在接收到地址并清除 ADDR 位后，（如果地址字节的最低位是 1）Slave 将数据（字节）从 DR 寄存器，经由内部 shift 寄存器发送到 SDA 上。

Slave 拉低 SCL，直到 ADDR 位被清除，并且待发送数据已写入 DR 寄存器。

当收到应答脉冲时：TxE 位被硬件置位，如果设置了 ITEVTEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 位被置位，但在下一个数据发送结束之前，没有新数据写入到 I2C_DR 寄存器，则 BTF 位被置位。Slave 拉低 SCL，直到 BTF 位被软件清零（读 I2C_SR1 之后，再写入 I2C_DR 寄存器）。

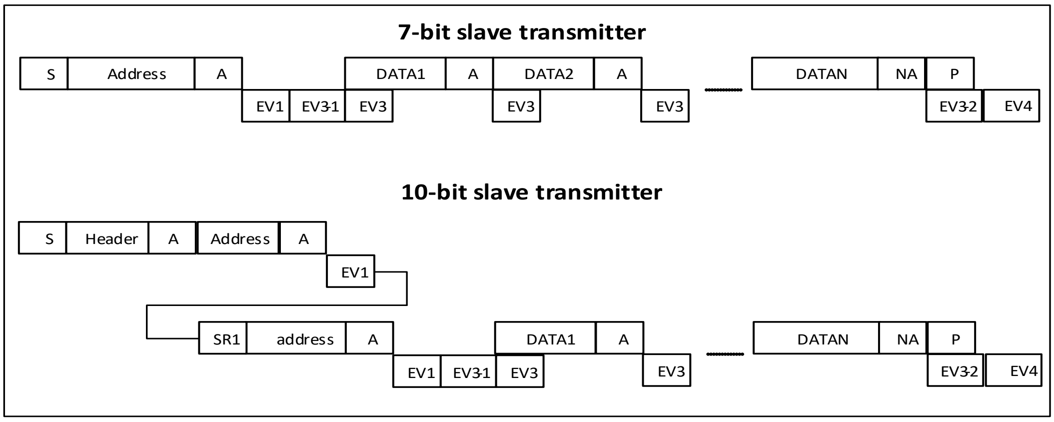


图 28-3 从发送器的传送序列图

Legend: S= Start（起始条件）, Sr= Repeated Start（重复的起始条件）, P= Stop（停止条件）, A= Acknowledge（响应）, NA= Non-acknowledge（不响应）, EVx= Event(ITEVFEN= 1 时产生中断)

EV1: ADDR=1, 通过先读 SR1 寄存器，再读 SR2 寄存器清零 ADDR 位

EV3-1： TxE=1, shift 寄存器 empty, 数据寄存器 empty, 向 DR 寄存器写 Data1

EV3： TxE=1, shift 寄存器不 empty, 数据寄存器 empty，向 DR 寄存器写（Data2）清零 TxE

EV3-2: AF=1; 软件向 AF 位写 0 清零该位

EV4: STOPF=1，通过先读 SR1 寄存器，后写 CR1 寄存器实现对该位的清零。

28.3.4.2. 从接收器

在接收到地址并清除 ADDR 后，（如果地址字节的最低位是 0）slave 将通过内部移位寄存器把从 SDA 线接收到的字节存进 DR 寄存器。I2C 接口在接收到每个字节后都执行下列操作：

● 如果设置了 ACK 位，则产生一个应答脉冲

● 硬件设置 RxNE=1。如果设置了 ITEVTEN 和 ITBUFEN 位，则产生一个中断。

如果 RxNE 被置位，并且在接收新的数据结束之前，DR 寄存器未被读出，则 BTF 位被置位，在清除 BTF（读出 I2C_SR1 之后再 I2C_DR 寄存器）之前，slave 一直拉低 SCL。(见下图)。

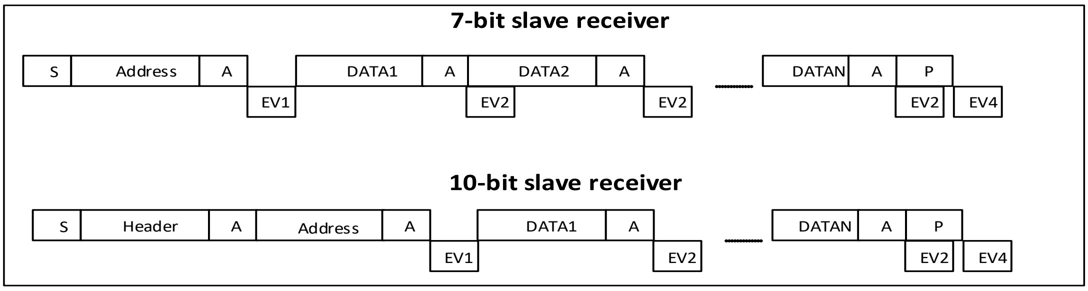


图 28-4 从接收器的传送序列图

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledged, EVx= Event(with interrupt if ITEVFEN=1)

EV1: ADDR=1，通过先读 SR1，后读 SR2，实现 ADDR 的清零

EV2: RxNE=1, 读 DR 寄存器清零该位

EV4: STOPF=1, 通过先读 SR1 寄存器，后写 CR1 寄存器实现对该位的清零。

Note:

— EV1 event 拉低 SCL，直到相应软件 sequence 的结束。

— EV2 软件 sequence 必须在当前 byte 传输完成前即完成。

— 当用户检查 SR1 寄存器内容后，应该对每个发现置位的标志位，进行完整的清除 sequence。比如 ADDR 和 STOPF 标志位，需要用以下 sequence：

如果 ADDR=1，先读 SR，再读 SR2；如果 STOPF=1，先读 SR1，再写 CR1。这样做的目的是确保如果 ADDR 和 STOPF 两位都被发现置位，都能被清除掉。

28.3.4.3. 关闭通信

在传输完最后一个数据字节后，master 产生一个停止条件，slave 检测到该条件时：

硬件置位 STOPF，如果设置了 ITEVTEN 位，则产生一个中断。

通过先读 SR1，后写 CR1，实现对 STOPF 位的清零。

28.3.5. I2C 主模式

在 Master 模式时，I2C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始，并以停止条件结束。

当通过 START 位在总线上产生了起始条件，设备就进入了 master 模式。

以下是 master 模式所要求的操作顺序：

1- 在 I2C_CR2 寄存器中设定该模块的输入时钟以产生正确的时序

2- 配置时钟控制寄存器

3- 配置上升时间寄存器

4- 编程 I2C_CR1 寄存器启动外设

5- 置 I2C_CR1 寄存器中的 START 位为 1，产生起始条件

I2C 模块的输入时钟频率必须至少是：

6- 标准模式下为：2MHz

7- 快速模式下为：4MHz

28.3.5.1. 主机产生 clock

CCR 寄存器以上升沿或者下降沿，产生 SCL 的高电平和低电平。由于 slave 可能拉长 SCL 信号，在 SCL 上升沿产生后，master 在 TRISE 寄存器编程的时间到达时，检查来自总线的 SCL 信号。

1） 如果 SCL 是低电平，意味着 slave 正在拉长 SCL 总线，高电平计数器停止计数，直到 SCL 被检测到高电平。这是为了确保 SCL 参数的最小高电平时间。
2） 如果 SCL 是高电平，高电平计数器保持计数。

实际上，即使 slave 不拉长 SCL，从 SCL 上升沿产生，到 SCL 上升沿被发现，这样的反馈环路也是要花费些时间的。这个回路的时间与 SCL 的上升时间（SCL 的 VIH 数据检测）有关系，再加上 SCL 输入路径的模拟噪声滤波，以及芯片内部由于用 APB 时钟进行的 SCL 同步。反馈回路的最大时间编程在 TRISE 寄存器中，所以无论 SCL 上升时间如何，SCL 的频率保持稳定。

28.3.5.2. 开始条件

当 BUSY=0 时，设置 START=1，I2C 接口将产生一个 Start 条件，并切换至 master 模式(MSL 被置位)。

注：在 master 模式下设置 START 位，将在当前字节传输完后，由硬件产生一个 ReStart 条件。

一旦发出 Start 条件：

● SB 位被硬件置位，如果设置了 ITEVTEN 位，则会产生一个中断。

master 读 SR1 寄存器，再把 slave 地址写入 DR 寄存器。（Transfer sequence EV5）

28.3.5.3. 从机地址发送

slave 地址通过内部移位寄存器被送到 SDA 线上。

● 在 10 位地址模式时，发送一个头端序列产生以下事件:

— ADDR10 位被硬件置位，如果设置了 ITEVTEN 位，则产生一个中断。

然后主设备等待一次读 SR1 寄存器，跟着第二个地址字节写入 DR 集寄存器。

ADDR 位被硬件置位，如果设置了 TEVFEN 位，则产生一个中断。

随后主设备等待一次读 SR1 寄存器，跟着读 SR2 寄存器。

● 在 7 位地址模式时，送出一个地址字节。

该地址字节一旦被送出，

— ADDR 位被硬件置位，如果设置了 ITEVTEN 位，则产生一个中断。

随后 Master 读 SR1 寄存器，跟着读 SR2 寄存器。

根据送出 slave 地址的最低位，master 决定进入发送器模式，还是进入接收器模式。

● 在 7 位地址模式时，

— 要进入发送器模式，主设备发送从地址时置最低位为’0’。

— 要进入接收器模式，主设备发送从地址时置最低位为’1’。

TRA 位指示主设备是在接收器模式还是发送器模式。

● 在 10 位地址模式时，

— 要进入发送器模式，主设备先送头字节(11110xx0)，然后送 LSB 位等于 0 的从地址。(头段字节中的 xx 是 10 位地址中的最高 2 位）。

– 要进入接收器模式，主设备先送头字节(11110xx0)，然后送 LSB 位等于 0 的从地址。然后再重新发送一个开始条件，后面跟着头字节(11110xx1)。

(头字节中的 xx 是 10 位地址中的最高 2 位）。TRA 位指示主设备是在接收器模式还是发送器模式。

28.3.5.4. 主机发送

在发送了地址和清除了 ADDR 位后, 主设备 master 通过内部移位寄存器将字节从 DR 寄存器发送到 SDA 线上。

Master 等待，直到第一个数据字节被写入 DR 寄存器（参见 EV8_1）。

当收到 ACK 脉冲时，TxE 位被硬件置位，如果设置了 INEVFEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 被置位，且在上一次数据发送结束之前，没有写新的数据字节到 DR 寄存器，则 BTF 被硬件置位。在清除 BTF（读 I2C_SR1 之后，再写 I2C_DR 寄存器）之前，I2C 接口将保持 SCL 为低电平。

关闭通信

在 DR 寄存器中写入最后一个字节后，通过设置 STOP 位产生一个停止条件(见图的 EV8_2)，然后 I2C 接口将自动回到从模式(MSL 位清除)。

注：当 *TxE* 或 *BTF* 位置位时，停止条件应安排在出现 *EV8_2* 事件时。

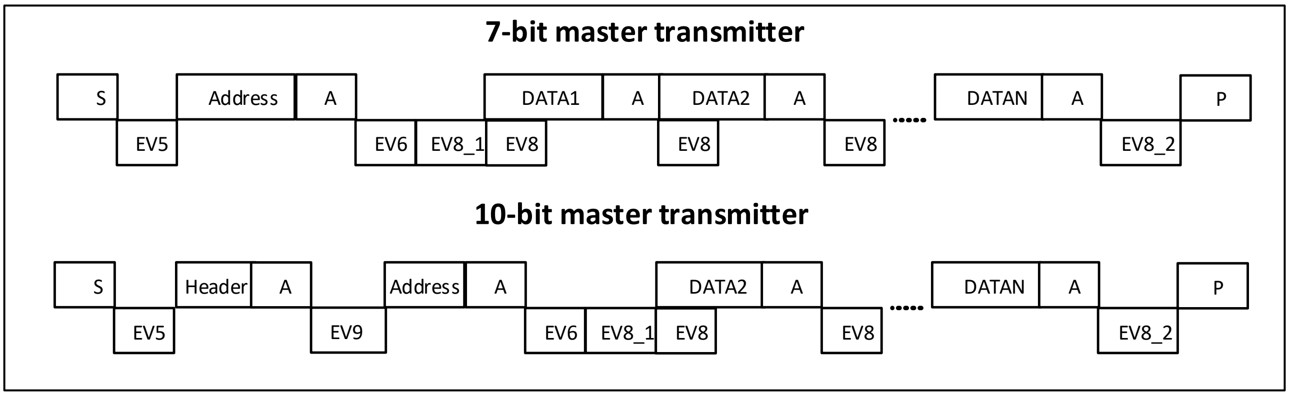


图 28-5 主发送器传送序列图

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1，读 SR1 然后将地址写入 DR 寄存器将清除该事件

EV6: ADDR=1,读 SR1 然后读 SR2 清除该事件

EV8_1: TXE=1，移位寄存器空

EV8: TXE=1，写 DR 寄存器清除该事件

EV8_2: TXE=1，BTF=1，产生停止条件时由硬件清除

EV9: ADDR10=1,读 SR1 然后写 DR 寄存器清除该事件

Note:

1） EV5, EV6, EV8_1 和 EV8_2 事件，拉长 SCL 的低电平，直到相应的软件 sequence 执行结束

2） EV8 软件 sequence 必须在当前字节发送完成前执行完毕。若 EV8 的软件 sequence 不能在当前传输的字节结束前完成，则推荐使用 BTF 代替 TxE，这产生的不利是减慢了通讯。

28.3.5.5. 主接收器

在发送地址和清除 ADDR 之后，I2C 接口进入主接收器模式。在此模式下，I2C 接口从 SDA 线接收数据字节，并通过内部移位寄存器送至 DR 寄存器。在每个字节后，I2C 接口依次执行以下操作：

- 如果 ACK 位被置位，发出一个应答脉冲。
- 硬件设置 RxNE=1，如果设置了 INEVFEN 和 ITBUFEN 位，则会产生一个中断。

如果 RxNE 位被置位，并且在接收新数据结束前， DR 寄存器中的数据没有被读走，硬件将设置 BTF=1，在清除 BTF 之前 I2C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再读出 I2C_DR 寄存器将清除 BTF 位。

关闭通信

Method 1:该方法的应用场景是：当 I2C 被设成应用程序中最高优先级的中断

Master 在从 Slave 接收到最后一个字节后，发送一个 NACK。接收到 NACK 后，Slave 释放对 SCL 和 SDA 线的控制。Master 就可以发送一个 Stop/Restart 条件。

1） 为了在收到最后一个字节后产生一个 NACK 脉冲，在读倒数第二个数据字节之后(在倒数第二个 RxNE 事件之后)必须清除 ACK 位。
2） 为了产生一个停止/重起始条件，软件必须在读倒数第二个数据字节之后(在倒数第二个 RxNE 事件之后)设置 STOP/START 位。
3） 当接收单个字节时，关闭应答和停止条件的产生位要刚好在 EV6 之后(EV6_1 时，清除 ADDR 之后)。
在产生了停止条件后，I2C 接口自动回到从模式(MSL 位被清除)。

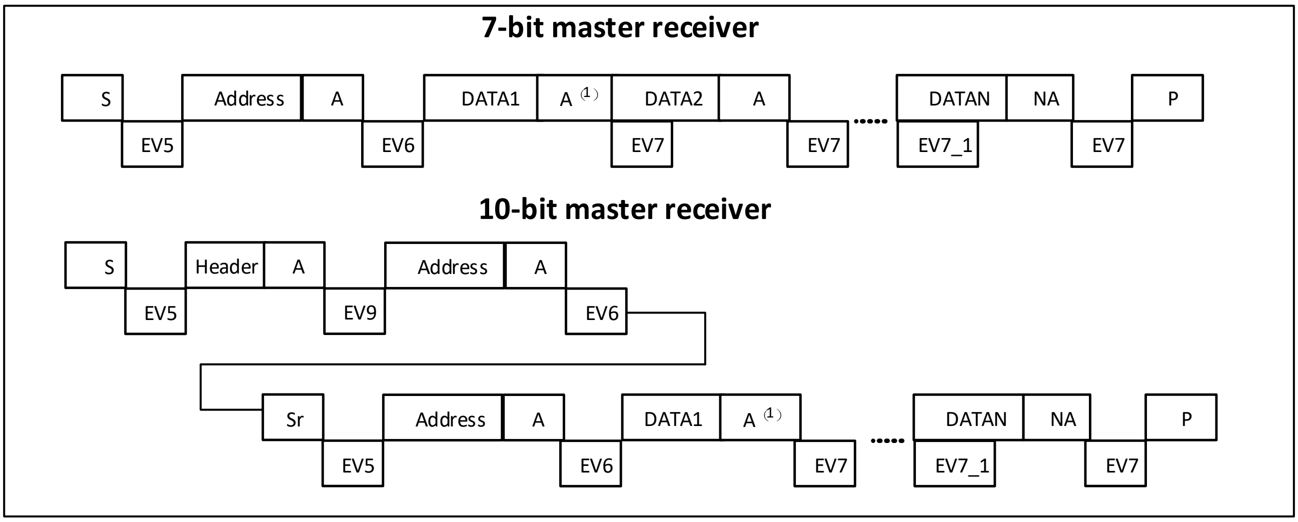


图 28-6 方法 1: 主模式发送时的时序

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5：SB=1，读 SR1，再写 DR 寄存器，该位被清零

EV6：ADDR=1，读 SR1，再读 SR2，该位被清零

EV6_1：无相关的标志事件，仅用作 1 个字节的接收

EV7：RxNE=1，读 DR 寄存器，该位被清零

EV7_1：RxNE=1, 读 DR 寄存器，写 ACK=0 并置位 STOP

EV9：ADDR10=1，读 SR1 然后写 DR 寄存器清除该事件

1) 如果是单个字节接收，则上述标注为（1）的地方会是 NA

2) EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件 sequence 执行结束

3) EV7 软件 sequence 必须在当前字节发送完成前执行完毕。在 EV7，软件 sequence 不能在当前传输的字节传输完成前，被管理。推荐使用 BTF 代替 RXNE，这产生的不利是减慢了通讯。
4) EV6_1 或者 EV7_1 的软件 sequence 必须在当前字节传输的 ACK 之前完成。

Method 2: 这个方法的应用场景是：I2C 的中断在应用中不是最高优先级，或者使用查询方式

用这个方法，DataN-2 没有被读，因此在 DataN-1 之后，通讯被拉长（RxNE 和 BTF 都被置位）。然后，在读 DR 寄存器的 DataN-2 前，清 ACK 位，以确保在 DataN ACK 之前被清掉。在此之后，在读 DataN-2 之后，置位 STOP/START 位，并读 DataN-1。在 RxNE 置位后，读 DataN

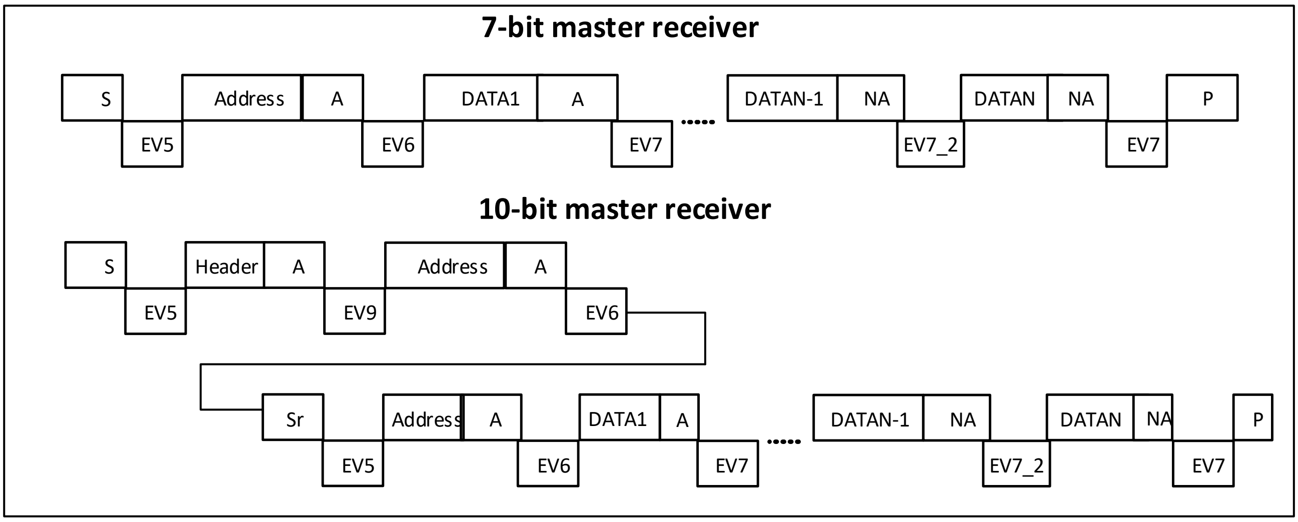


图 28-7 方法 2: N>2 时主模式发送时的时序

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, 先读 SR1 寄存器，再写 DR 寄存器，清零该位

EV6: ADDR，先读 SR1，再读 SR2，清零该位

EV7: RxNE=1, 读 DR 寄存器清零该位

EV7_2: BTF=1, DataN-2 存在 DR 寄存器中，DataN-1 存在 shift 寄存器中，写 ACK=0，读 DR 寄存器中的 DataN-2。置位 STOP，读 DataN-1

EV9：ADDR10=1，读 SR1 然后写 DR 寄存器清除该事件

Note:

- EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件 sequence 执行结束
- EV7 软件 sequence 必须在当前字节发送完成前执行完毕。在 EV7，软件 sequence 不能在当前传输的字节传输完成前，被管理。推荐使用 BTF 代替 RXNE，这产生的不利是减慢了通讯。

•当 3 个字节要被读走：

— RxNE=1 = > Nothing(DataN-2 not read)。
  - DataN-1 received
  - BTF=1, shift 和 data 寄存器都是 full：DR 寄存器存放了 DataN-2，shift 寄存器存放了 DataN-1 = > SCL 拉低：总线上没有其他要被接收的数据
  - 清零 ACK 位
  - 读 DR 寄存器中的 DataN-2 = > 这将启动 shift 寄存器对 DataN 的接收
  - DataN 接收完成（with a NACK）
  - 写 START 或者 STOP 位
  - 读 DataN-1

- RxNE=1
- 读 DataN

以上流程是针对 N > 2 的描述。1 个字节和 2 个字节的接收，要用不同的处理方式，参见以下描述：

• 2 个字节接收的情况

— 置位 POS 和 ACK 位

- 等待 ADDR 置位
- 清零 ADDR 位
- 清零 ACK 位
- 等待 BTF 被置位
- 写 STOP 位
- 读 DR 两次

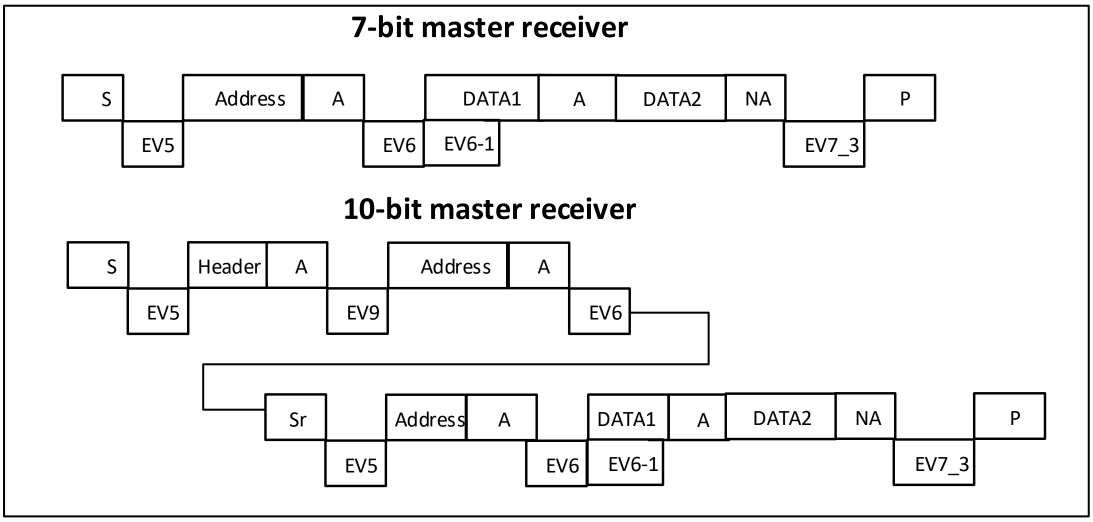


图 28-8 方法 2: N=2 时主模式发送时的时序

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(with interrupt if ITEVFEN= 1)

EV5: SB=1, 先读 SR1 寄存器，再写 DR 寄存器，清零该位

EV6: ADDR=1，先读 SR1 寄存器，后读 SR2 寄存器，清零 ADDR 位

EV6_1: 无相关的标志位事件。在 EV6 后，也就是地址被清零后，ACK 应该被清零

EV7_3: BTF=1，写 STOP=1，之后读两次 DR（Data1 和 Data2）

EV9：ADDR10=1，读 SR1 然后写 DR 寄存器清除该事件 **Note:**

- EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件 sequence 执行结束
- EV6_1 的软件 sequence 必须在当前字节传输的 ACK 之前完成

• 单个字节接收的情况

— 在 ADDR 事件里，清零 ACK 位

- 清零 ADDR
- 写 STOP 或者 START 位
- 在 RxNE 标志置位后，读数据

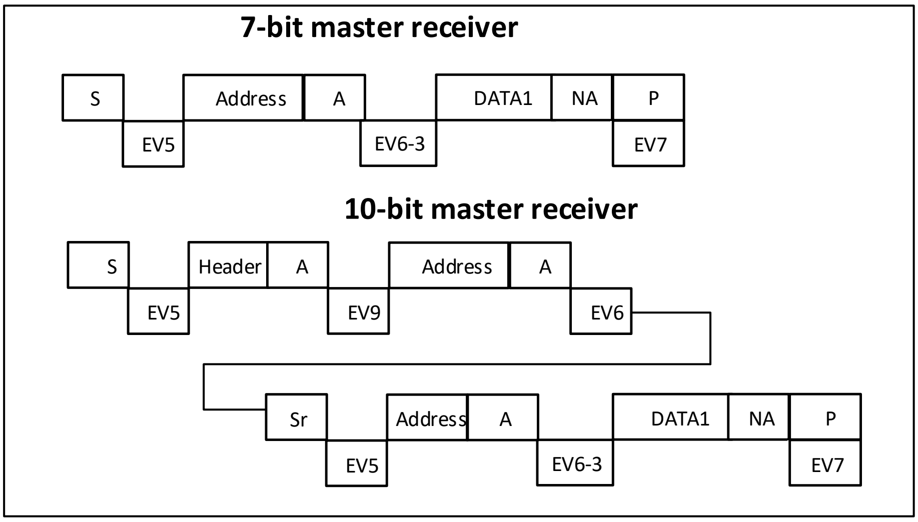


图 28-9 方法 2: N=1 时主模式发送时的时序

Legend: S= Start, Sr= Repeated Start, P= Stop, A= Acknowledge, EVx= Event(当 ITEVFEN= 1 时产生中断)

EV5: SB=1, 先读 SR1 寄存器，再写 DR 寄存器，清零该位

EV6_3: ADDR=1，写 ACK=0。先读 SR1 寄存器，后读 SR2 寄存器，清零 ADDR 位。在 ADDR 被清零后，写 STOP=1

EV7: RxNE=1, 读 DR 寄存器清零该位

EV9：ADDR10=1，读 SR1 然后写 DR 寄存器清除该事件

Note:

EV5，EV6，EV8_1 和 EV8_2 事件会拉长 SCL 的低电平，直到相应的软件 sequence 执行结束。

28.3.6. 错误状态

28.3.6.1. 总线错误

在一个地址或数据字节传输期间，当 I2C 接口检测到一个外部的停止或起始条件则产生总线错误。此时：

● BERR 位被置位为’1’；如果设置了 ITERREN 位，则产生一个中断；

● 在 slave 模式：数据被丢弃，硬件释放总线：

– 如果是错误的 Start 条件，slave 认为是一个 Restart，并等待地址或停止条件

– 如果是错误的 Stop 条件，slave 按正常的停止条件操作，同时硬件释放总线

● 在 master 模式：硬件不释放总线，同时不影响当前的传输状态。此时由软件决定是否要中止当前的传输。此时由软件决定是否要中止当前的传输。

28.3.6.2. 应答失败(AF)

当接口检测到一个无应答位时，产生应答错误。此时：

● AF 位被置位，如果设置了 ITERREN 位，则产生一个中断

● 当发送器接收到一个 NACK 时，必须复位通讯：

– 如果是处于 slave 模式，硬件释放总线。

– 如果是处于 master 模式，软件必须生成一个停止条件或者 repeated start。

28.3.6.3. 仲裁丢失 (ARLO)

当 I2C 接口检测到仲裁丢失时产生仲裁丢失错误，此时：

● ARLO 位被硬件置位，如果设置了 ITERREN 位，则产生一个中断

● I2C 接口自动回到从模式(MSL 位被清除)。当 I2C 接口丢失了仲裁，则它无法在同一个传输中响应它的从地址，但它可以在赢得总线的 master 发送 repeated start 条件之后响应

● 硬件释放总线

28.3.6.4. 过载 overrun/欠载 underrun (OVR)

在 slave 模式下，如果禁止时钟延长，I2C 接口正在接收数据时，当它已经接收到一个字节(RxNE=1)，但在 DR 寄存器中前一个字节数据还没有被读出，则发生 overrun 错误。

此时：

● 最后接收的数据被丢弃

● 在 overrun 错误时，软件应清除 RxNE 位，发送器应该重新发送最后一次发送的字节

在 slave 模式下，如果禁止时钟延长，I2C 接口正在发送数据时，在下一个字节的时钟到达之前，新的数据还未写入 DR 寄存器(TxE=1)，则发生欠载错误。此时：

● 在 DR 寄存器中的前一个字节将被重复发出

● 用户应该确定在发生 underrun 错时，接收端应丢弃重复接收到的数据。发送端应按 I2C 总线标准在规定的时间更新 DR 寄存器

在发送第一个字节时，必须在清除 ADDR 之后且在第一个 SCL 上升沿之前写入 DR 寄存器；如果不能做到这点，则接收方应该丢弃第一个数据。

28.3.7. SDA/SCL 控制

● 如果允许时钟延长：

– 发送器模式：如果 TxE=1 且 BTF=1：I2C 接口在传输前保持时钟线为低，以等待软件读取 SR1，然后把数据写进数据寄存器(DR 和 shift 寄存器都是空的)。

– 接收器模式：如果 RxNE=1 且 BTF=1：I2C 接口在接收到数据字节后保持时钟线为低，以等待软件读 SR1，然后读数据寄存器 DR(DR 和 shift 寄存器都是满的)。

● 如果在 slave 模式中禁止时钟延长：

– 如果 RxNE=1，在接收到下个字节前 DR 还没有被读出，则发生 overun。接收到的最后一个字节丢失。

– 如果 TxE=1，在必须发送下个字节之前却没有新数据写进 DR，则发生 underrun。相同的字节将被重复发出。

– 硬件未实现对写冲突的控制。

28.3.8. DMA 请求

DMA 请求(当被使能时)仅用于数据传输。发送时数据寄存器变空，或接收时数据寄存器变满，则产生 DMA 请求。DMA 必须在当前字节传输结束之前被初始化和使能。DMAEN 位（I2C_CR2 寄存器中）必须在 ADDR 事件发生前使能。

在 master 或者 slave 模式，当时钟延展功能使能，DMAEN 位可以在清零 ADDR 之前的 ADDR 事件期间置位。DMA 请求必须在当前字节传输完成之前响应。当 DMA 传输数据长度达到 DMA 设定的值时，DMA controller 向 I2C 发送 EOT（End of transfer），并产生 transfer complete 中断（如果中断使能位有效）：

● Master transmitter：在 EOT 中断服务程序中，需禁止 DMA 请求，然后在等到 BTF 事件后，置位 stop 条件。

● Master receiver：当要接收的数据数目大于或等于 2 时，DMA controller 发送一个硬件信号 EOT_1，它对应 DMA 传输(字节数 - 1)。如果在 I2C_CR2 寄存器中设置了 LAST 位，硬件在发送完 EOT_1 后的下一个字节，将自动发送 NACK。在中断允许的情况下，用户可以在 DMA 传输完成的中断服务程序中产生一个停止条件。

28.3.8.1. DMA 传输

通过置位 I2C_CR2 寄存器中的 DMAEN 位，可以使能 DMA 模式。只要 TxE 位被置位，数据将由 DMA 从预置的存储区，装载进 I2C_DR 寄存器。为 I2C 分配一个 DMA 通道，须执行以下步骤(x 是通道号)：

1. 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器地址。数据将在每个 TxE 事件后，从存储器传送至这个地址。
2. 在 DMA_CMARx 寄存器中设置存储器地址。数据在每个 TxE 事件后从这个存储区传送至 I2C_DR。
3. 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 TxE 事件后，此值将被递减。
4. 利用 DMA_CCRx 寄存器中的 PL[0:1]位配置通道优先级。
5. 设置 DMA_CCRx 寄存器中的 DIR 位，并根据应用要求可以配置在整个传输完成一半或全部完成时发出中断请求。
6. 通过设置 DMA_CCTx 寄存器上的 EN 位激活通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I2C 接口发送一个传输结束的 EOT/EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：如果使用 *DMA* 进行发送时，不要设置 *I2C_CR2* 寄存器的 *ITBUFEN* 位。

28.3.8.2. DMA 接收

通过设置 I2C_CR2 寄存器中的 DMAEN 位可以激活 DMA 接收模式。每次接收到数据字节时，将由 DMA 把 I2C_DR 寄存器的数据传送到设置的存储区(参考 DMA 说明)。设置 DMA 通道进行 I2C 接收，须执行以下步骤(x 是通道号)：

1. 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器的地址。数据将在每次 RxNE 事件后从此地址传送到存储区。
2. 在 DMA_CMARx 寄存器中设置存储区地址。数据将在每次 RxNE 事件后从 I2C_DR 寄存器传送到此存储区。
3. 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 RxNE 事件后，此值将被递减。
4. 用 DMA_CCRx 寄存器中的 PL[0:1]配置通道优先级。
5. 清除 DMA_CCRx 寄存器中的 DIR 位，根据应用要求可以设置在数据传输完成一半或全部完成时发出中断请求。
6. 设置 DMA_CCRx 寄存器中的 EN 位激活该通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I2C 接口发送一个传输结束的 EOT/EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：如果使用 *DMA* 进行接收时，不要设置 *I2C_CR2* 寄存器的 *ITBUFEN* 位。

28.3.9. SMBus

系统管理总线(SMBus)是一个两线接口。通过它，各设备之间以及设备与系统的其他部分之间可以互相通信。它基于 I2C 操作原理。SMBus 为系统和电源管理相关的任务提供一条控制总线。一个系统利用 SMBus 可以和多个设备互传信息，而不需使用独立的控制线路。

系统管理总线(SMBus)标准涉及三类设备。从设备，接收或响应命令的设备。主设备，用来发布命令，产生时钟和终止发送的设备。主机，是一种专用的主设备，它提供与系统 CPU 的主接口。主机必须具有主-从机功能，并且必须支持 SMBus 通报协议。在一个系统里只允许有一个主机。

本项目的 I2C1 模块支持 SMbus/PMbus 功能

28.3.9.1. SMBus 和 I2C 之间的相似点

1) 2 条线的总线协议(1 个时钟，1 个数据) + 可选的 SMBus 提醒线
2) 主-从通信，主设备提供时钟
3) 多主机功能
4) SMBus 数据格式类似于 I2C 的 7 位地址格式

28.3.9.2. SMBus 和 I2C 之间的不同点

下表列出来 SMBus 和 I2C 的不同点:

表 28-1 SMBus 与 I2C 的比较

SMBus	I2C
最大传输速度 100kHz	最大传输速度 400kHz
最小传输速度 10kHz	无最小传输速度
35ms 时钟低超时	无时钟超时
固定的逻辑电平	逻辑电平由 VDD 决定
不同的地址类型(保留、动态等)	7 位、 10 位和广播呼叫从地址类型
不同的总线协议（快速命令、处理呼叫等)	无总线协议

28.3.9.3. SMBus 应用用途

利用系统管理总线，设备可提供制造商信息，告诉系统它的型号/部件号，保存暂停事件的状态，报告不同类型的错误，接收控制参数，和返回它的状态。SMBus 为系统和电源管理相关的任务提供控制总线。

28.3.9.4. 地址解析协议(ARP)

SMBus 从地址冲突可以通过给每个从设备动态分配一个新的唯一地址来解决。
ARP 有以下的属性:

- 地址分配利用标准 SMBus 物理层仲裁机制
- 当设备维持供电期间，分配的地址仍保持不变，允许设备在断电时保留其地址。

- 在地址分配后，没有额外的 SMBus 的打包开销(也就是说访问分配地址的设备与访问固定地址的设备所用时间是一样的）。
- 任何一个 SMBus 主设备可以遍历总线。

28.3.9.5. SMBus 提醒模式(ALERT)

SMBus 提醒是一个带中断线的可选信号，用于那些希望扩展他们的控制能力而牺牲一个引脚的设备。SMBALERT 和 SCL 和 SDA 信号一样，是一种线与信号。SMBALERT 通常和 SMBus 广播呼叫地址一起使用。与 SMBus 有关的消息为 2 字节。

单一的从设备可以通过 SMBALERT 发信号给主机表示它希望进行通信，这可通过设置 I2C_CR1 寄存器上的 ALERT 位实现。主机处理该中断并通过提醒响应地址 ARA(Alert Response Address，地址值为 0001100x)访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应答 ARA。此状态是由 I2C_SR1 寄存器中的 SMBALERT 状态标记来标识的。主机执行一个修改过的接收字节操作。由从发送设备提供的 7 位设备地址被放在字节的 7 个最高位上，第八个位可以是 0 或 1。

如果多个设备把 SMBALERT 拉低，最高优先级设备(最小的地址)将在地址传输期间通过标准仲裁赢得通信权。在确认从地址后，此设备不得再拉低它的 SMBALERT，如果当信息传输完成后，主机仍看到 SMBALERT 低，就知道需要再次读 ARA。没有执行 SMBALERT 信号的主机可以定期访问 ARA。有关 SMBus 提醒模式的更多详细资料，请参考 2.0 版的 SMBus 规范。

28.3.9.6. 总线协议

SMBus 技术规范支持 9 个总线协议。有关这些协议的详细资料和 SMBus 地址类型，请参考 2.0 版的 SMBus 规范。这些协议由用户的软件来执行。

28.3.9.7. 超时错误(TIMEOUT)

在定时规范上 I2C 和 SMBus 之间有很多差别。

SMBus 定义一个时钟低超时，35ms 的超时。SMBus 规定 TLOW:SEXT 为从设备的累积时钟低扩展时间。SMBus 规定 TLOW:MEXT 为主设备的累积时钟低扩展时间。更多超时细节请参考 2.0 版的 SMBus 规范。

I2C_SR1 中的状态标志 Timeout 或 Tlow 错误表明了这个特征的状态。

28.3.9.8. PMBus

PMbus 是基于 SMbus 而来的，传输逻辑与 SMbus 完全一致，区别在于 PMbus 定义了一些与电源管理相关的功能（由软件完成）。

28.4. I2C 中断

表 28-28 I2C 中断请求

中断事件	事件标志	开启控制位
起始位已发送(Master)	SB	ITEVTEN
地址已发送(Master) 或 地址匹配(Slave)	ADDR	
10 位头段已发送	ADDR10	
已收到停止(Slave)	STOPF	
数据字节传输完成	BTF	

中断事件	事件标志	开启控制位
接收缓冲区非空	RxNE	ITEVTEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失(Master)	ARLO	
响应失败	AF	
过载/欠载	OVR	
PEC 错误	PECERR	

28.5. I2C 寄存器

寄存器可以 half-word 或者 word 访问。

28.5.1. I2C 控制寄存器 1 (I2C_CR1)

Address offset:0x00

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	Res.	SMBUS	PE
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
15	SWRST	RW	0	软件复位。 当被置位时，I2C 处于复位状态。在复位释放前，要确保 I2C 的引脚被释放，总线是空闲状态。 0：I2C 模块不处于复位状态 1：I2C 模块处于复位状态 注：该位可以用于 error 或 locked 状态时重新初始化 I2C。如 BUSY 位为 1，在总线上又没有检测到停止条件时。
14	Reserved	-	-	Reserved
13	ALERT	RW	0	SMBus 提醒。 0：释放 SMBAlert 引脚使其变高。提醒响应地址头紧跟在 NACK 信号后面； 1：驱动 SMBAlert 引脚使其变低。提醒响应地址头紧跟在 ACK 信号后面； PE=0 时，由硬件清除。
12	PEC	RW	0	数据包出错检查 软件可置位和清零该位，硬件可以在以下情况下清零该位：当传送 PEC 后，起始条件、或者停止条件、或者当 PE=0 时； 0：无 PEC 传输 1：PEC 传输（在发送或接收模式）

Bit	Name	R/W	Reset Value	Function
				注：仲裁丢失时，PEC 的计算失效。
11	POS	RW	0	ACK/PEC 位置（用于数据接收），软件可置位/清零该寄存器，或 PE=0 时由硬件清零。 0：ACK 位控制当前移位寄存器内正在接收的字节的(N)ACK。PEC 位表明当前移位寄存器内的字节是 PEC 1：ACK 位控制在移位寄存器里接收的下一个字节的(N)ACK。PEC 位表明在移位寄存器里接收的下一个字节是 PEC 注：POS 位只能用在 2 字节的接收配置中，必须在接收数据之前配置。 为了 NACK 第 2 个字节，必须在清除 ADDR 之后清除 ACK 位。 为了检测第 2 个字节的 PEC，必须在配置了 POS 位之后，ADDR stretch 事件时设置 PEC 位。
10	ACK	RW	0	应答使能。软件可置位/清零该寄存器，或 PE=0 时由硬件清零。 0：无应答返回 1：在接收到一个字节后返回一个应答。（匹配的地址或数据）
9	STOP	RW	0	停止条件产生，软件可以置位/清零该寄存器，或者当检测到停止条件时，由硬件清除；当检测到超时错误时，硬件置位。 在主模式下: 0：无停止条件产生 1：在当前字节传输或在当前起始条件发出后产生停止条件 在从模式下: 0：无停止条件产生 1：在当前字节传输后释放 SCL 和 SDA 线
8	START	RW	0	起始条件产生。 软件可置位/清零该寄存器，或当起始条件发出后或 PE=0 时由硬件清零。 主模式: 0：无起始条件产生 1：重复产生起始条件 从模式: 0：无起始条件产生 1：当总线空闲时，产生起始条件（并由硬件自动切换到 master mode）
7	NOSTRETCH	RW	0	禁止时钟延长（Slave）。 当 ADDR 或 BTF 标志被置位时，该位用于 slave 禁止时钟延长，直到被软件复位。 0：允许时钟延长 1：禁止时钟延长

Bit	Name	R/W	Reset Value	Function
6	ENGC	RW	0	广播呼叫使能。 0：禁止广播呼叫。以 NACK 响应地址 00h 1：允许广播呼叫。以 ACK 响应地址 00h
5	ENPEC	RW	0	PEC 使能。 0：禁止 PEC 计算 1：开启 PEC 计算
4	ENARP	RW	0	ARP 使能。 0：禁止 ARP； 1：使能 ARP； 如果 SMBTYPE=0，使用 SMBus 设备的默认地址； 如果 SMBTYPE=1，使用 SMBus 的主地址。
3	SMBTYPE	RW	0	SMBus 类型。 0：SMBus 设备； 1：SMBus 主机；
2	Reserved	-	-	Reserved
1	SMBUS	RW	0	SMBus 模式。 0：I2C 模式； 1：SMBus 模式；
0	PE	RW	0	I2C 模块使能。 0：禁止 1：I2C 使能 注：如果清除该位时通讯正在进行，在当前通讯结束后，I2C 模块被禁用并返回空闲状态。 由于在通讯结束后 PE=0，所有的位被清除。 在主模式下，通讯结束之前，绝不能清除该位。

28.5.2. I2C 控制寄存器 2 (I2C_CR2)

Address offset:0x04

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	LAST	DMAEN	IT-BUFEN	ITEV-TEN	ITER-REN	Res	Res	FREQ[5:0]					
			RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:13	Reserved	-	-	Reserved
12	LAST	RW	0	DMA 最后一次传输。 0：下一次 DMA 的 EOT 不是最后的传输 1：下一次 DMA 的 EOT 是最后的传输 注：该位在主接收模式使用，使在最后一次接收数据时可以产生一个 NACK。
11	DMAEN	RW	0	DMA 请求使能。 0：禁止 DMA 请求； 1：当 TxE=1 或 RxNE=1 时，允许 DMA 请求。
10	ITBUFEN	RW	0	缓冲器中断使能。 0：当 TxE=1 或 RxNE=1 时，不产生中断

Bit	Name	R/W	Reset Value	Function
				1：当 TxE=1 或 RxNE=1 时，产生事件中断（不管 DMAEN 是何值）
9	ITEVTEN	RW	0	事件中断使能。 0：禁止 1：允许事件中断 在下列条件下，将产生该中断： 1) SB=1（主模式）； 2) ADDR=1（主/从模式） 3) STOPF=1（从模式） 4) BTF=1，但没有 TxE 或 RxNE 事件 5) 如果 ITBUFFEN=1，TxE 事件为 1 6) 如果 ITBUFEN=1，RxNE 事件为 1
8	ITERREN	RW	0	出错中断使能。 0：禁止出错中断； 1：允许出错中断； 在下列条件下，将产生该中断： — BERR=1 — ARLO=1 — AF=1 — OVR=1 — PECERR=1 — TIMEOUT=1 — SMBAlert=1
7:6	Reserved	-	-	Reserved
5:0	FREQ	RW	0	I2C 模块时钟频率。 必须用 APB 时钟频率的值配置该寄存器，以产生与 I2C 协议兼容的数据 setup 和 hold 时间。 最小允许可设定的频率是 4MHz（标准模式，即 100k）、8MHz（400k），最大频率是芯片最高的 APB 时钟频率。 000000：禁止 000001：禁止 000100：4MHz ….. 100100：36MHz ….. 110010：50MHz 大于 100100：禁止。

28.5.3. I2C 自身地址寄存器 1 (I2C_OAR1)

Address offset:0x08

Reset value:0x4000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD-MODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	ADDMODE	RW	0	寻址模式（从模式）。 0：7 位从地址（不响应 10 位地址）； 1：10 位从地址（不响应 7 位地址）；
14:10	Reserved	-	-	Reserved
9:8	ADD[9:8]	RW	0	接口地址。 7 位地址模式该寄存器无关。 10 位地址模式位地址的 9~8 位。
7:1	ADD[7:1]	RW	0	接口地址的 7~1 位。
0	Reserved	-	-	Reserved

28.5.4. I2C 自身地址寄存器 2 (I2C_OAR2)

Address offset:0x0C

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	OA2MSK[2:0]			ADD2[7:1]							ENDUAL
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	-	Reserved
7:1	ADD2[7:1]	RW	0	接口地址的 7~1 位。 双地址模式下地址的 7~1 位。
0	ENDUAL	RW	0	双地址模式使能位。 0：在 7 位地址模式下，只有 OAR1 被识别； 1：在 7 位地址模式下，OAR1 和 OAR2 都被识别。

28.5.5. I2C 数据寄存器 (I2C_DR)

Address offset:0x10

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	DR[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	-	Reserved
7:0	DR[7:0]	RW	0	8 位数据寄存器，芯片内部实际是两个独立的 buffer 共用一个地址，分别用于存放接收到的数据（RX_DR）、放置要发送到总线的数据（TX_DR）。 **发送器模式：**

Bit	Name	R/W	Reset Value	Function
				当写一个字节至 DR 寄存器时（实际写到 TX_DR），自动启动数据传输。一旦传输开始（TxE=1），如果能及时把下一个需传输的数据写入 DR 寄存器，I2C 模块将保持连续的数据流。 **接收器模式：** 接收到的字节被拷贝到 DR 寄存器（实际是 RX_DR）（RxNE=1）。在接收到下一个字节（RxNE=1）之前读出数据寄存器，即可实现连续的数据接收。 注： 1) 在 slave 模式下，地址不会被 copy 进数据寄存器 DR 2) 硬件不处理写冲突（如果 TxE=0，仍能写入数据寄存器） 3) 如果在处理 ACK 脉冲时发生 ARLO 事件，接收到的字节不会被 copy 到数据寄存器里，因此不能读到

28.5.6. I2C 状态寄存器(I2C_SR1)

Address offset:0x14

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMBALERT	TIMEOUT	Res.	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
15	SMBALERT	RC_W0	0	SMBus 提醒状态。 SMBus 主机模式： 0：无 SMBus 提醒； 1：在引脚产生 SMBAlert 提醒事件； SMBus 从机模式： 0：没有 SMBAlert 响应地址头序列； 1：收到 SMBAlert 响应地址头序列直到 SMBAlert 变低。 该位由软件写 0 或 PE=0 时由硬件清除。
14	TIMEOUT	RC_W0	0	超时或 Tlow 错误。 0：无超时错误； 1：SCL 为低的持续时间已达到 25ms（超时）；或者主机低电平累计时钟扩展时间超过 10ms（Tlow：next）；或从设备低电平累积时钟扩展时间超过 25ms（Tlow:sext）。 当在从模式下设置该位：从设备复位通讯，硬件释放总线； 当在主模式下设置该位：硬件发出停止条件； 该位由软件写 0 清除，或当 PE=0 时由硬件清除。

Bit	Name	R/W	Reset Value	Function
				Note: 该功能仅在 SMBUS 模式有作用。
13	Reserved	-	-	Reserved
12	PECERR	RC_W0	0	在接收时发生 PEC 错误。 0：无 PEC 错误，接收到 PEC 后返回 ACK（如果 ACK=1） 1：有 PEC 错误，接收到 PEC 后返回 NACK（不管 ACK 为何值） 该位由软件写 0 清除，或当 PE=0 时由硬件清除。
11	OVR	RC_W0	0	过载/欠载标志。 0：无过载/欠载； 1：出现过载/欠载。 当 NOSTRETCH=1 时，在从模式下该位被硬件置位； 在接收模式中当收到一个新的字节时（包括 ACK 应答脉冲），数据寄存器里的内容还未被读出，则新接收的字节将丢失。 在发送模式中当要发送一个新的字节时，却没有新的数据写入数据寄存器，同样的字节将被发送两次。 该位由软件写 0 清除，或当 PE=0 时由硬件清除。 注：如果数据寄存器的写操作发生时间非常接近 SCL 的上升沿，发送的数据是不确定的，并发生保持时间错误。
10	AF	RC_W0	0	应答失败标志。 0：没有应答失败； 1：应答失败。 当没有返回应答时，硬件将置位该寄存器。 该位由软件写 0 清除，或当 PE=0 时由硬件清除。
9	ARLO	RC_W0	0	仲裁丢失（主模式）。 0：没有检测到仲裁丢失； 1：检测到仲裁丢失。 当接口失去对总线的控制给另一个主机时，硬件将置位该寄存器。 该位由软件写 0 清除，或在 PE=0 时由硬件清除。 在 ARLO 事件之后，I2C 接口自动切换回从模式（M/SL=0）。
8	BERR	RC_W0	0	总线出错标志。 0：无起始或者停止条件出错； 1：起始或者停止条件出错。 当接口检测到错误的起始或者停止条件，硬件将该位置 1。

Bit	Name	R/W	Reset Value	Function
				该位由软件写 0 清除，或者在 PE=0 时由硬件清除。
7	TxE	R	0	数据寄存器为空（发送时）标志。 0：数据寄存器非空； 1：数据寄存器为空。 在发送数据时，数据寄存器为空时该位被置 1，在发送地址阶段不设置该位。 软件写数据到 DR 寄存器可清除该位，或在发生一个起始或停止条件后，或当 PE=0 时由硬件自动清除。 如果收到一个 NACK，或下一个要发送的字节时 PEC（PEC=1），该位不被置位。 注：在写入第 1 个要发送的数据后，或设置了 BTF 时写入数据，都不能清除 TxE 位，因为此时数据寄存器为空。
6	RxNE	R	0	数据寄存器非空（接收时）标志。 0：数据寄存器为空； 1：数据寄存器非空。 在接收时，当数据寄存器不为空，置位该寄存器。在接收地址阶段，该寄存器不置位。 软件对数据寄存器的读写操作会清除该寄存器，或当 PE=0 时由硬件清除。 注：当设置了 BTF 时，读取数据不能清除 RxNE 位，因为此时数据寄存器仍为满。
5	Reserved	-	-	Reserved
4	STOPF	R	0	停止条件检测位（从模式）。 0：没有检测到停止位； 1：检测到停止条件。 在一个应答之后（如果 ACK=1），当从设备在总线上检测到停止条件时，硬件将该位置 1。 软件读取 I2C_SR1 寄存器后，对 I2C_CR1 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。
3	ADD10	R	0	10 位地址头序列已发送（主模式）。 0：没有 ADD10 事件发生。 1：主设备已经将第一个地址字节发送出去。 在 10 位地址模式下，当主设备将已将第一个字节发送出去时，硬件将该位置 1. 软件读取 I2C_SR1 寄存器后，对 I2C_DR 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。 主：收到 NACK 后，该寄存器不被置位。
2	BTF	R	0	字节传输结束标志位。 0：字节传输未完成 1：字节传输成功结束

Bit	Name	R/W	Reset Value	Function
				在下列情况下硬件将置位该寄存器（当 slave 模式，NOSTRETCH=0 时；master 模式，与 NOSTRETCH 无关）： 1) 接收时，当收到一个新字节（包括 ACK 脉冲）且数据寄存器还未被读取（RxNE=1）。 2) 发送时，当一个新数据应该被发送，且数据寄存器还未被写入新的数据（TxE=1）。 软件读取 I2C_SR1 寄存器后，对数据寄存器的读或写操作将清除该位；或发送一个起始或停止条件后，或当 PE=0 时，由硬件清除。 注: 在收到一个 NACK 后，BTF 位不会被置位。
1	ADDR	R	0	地址已被发送（主模式）/地址匹配（从模式）。 软件读取 I2C_SR2 寄存器后，再读 I2C_SR1 寄存器将清除该位；或在传输中发送一个起始或停止条件后，或当 PE=0 时，由硬件清除。 地址匹配（Slave）： 0：地址不匹配或没有收到地址； 1：收到的地址匹配。 当收到的从地址与 OAR 寄存器或 general call 地址匹配，或 SMBus 设备默认地址、或 SMBus 主机识别除 SMBus 提醒时，硬件将置位该位。 Note: 在 slave 模式下，推荐进行完整的清零 sequece，即在 ADDR 被置位后，先读 SR1 寄存器，再读 SR2 寄存器。 地址已发送（Master）： 0：地址发送没有结束； 1：地址发送结束。 10 位地址时，当收到地址的第二个字节的 ACK 后置位。 7 位地址时，当收到 ACK byte 后置位。 Note: 在收到 NACK 后，该寄存器不会被置位。
0	SB	R	0	起始位标志（主模式）。 0：未发送起始条件； 1：起始条件已发送； —当发送起始条件时，置位该寄存器。 —软件读取 I2C_SR1 寄存器后，对数据寄存器的读或写操作将清除该位；或当 PE=0 时，由硬件清除。

28.5.7. I2C 状态寄存器 2 (I2C_SR2)

Address offset:0x18

Reset value:0x0000

Note: 即使 ADDR 标志位在读 I2C_SR1 寄存器后被置位，在读 I2C_SR1 之后再读 I2C_SR2 寄存器，也会清零 ADDR 标志位。因此，仅在发现 I2C_SR1 寄存器的 ADDR 位被置位或者 STOPF 位被清零时，I2C_SR2 寄存器才必须被读。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUALF	SMBHOST	SMBDE-FAULT	GEN-CALL	Res.	TRA	BUSY	MSL
R	R	R	R	R	R	R	R	R	R	R	R		R	R	R

Bit	Name	R/W	Reset Value	Function
15:8	PEC	R	0	数据包出错检测寄存器。 当 ENPEC=1 时，该寄存器存放内部的 PEC 的值。
7	DUALF	R	0	双地址标志（从模式）. 0：接收到的地址与 OAR1 匹配； 1：接收到的地址与 OAR2 匹配。 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
6	SMBHOST	R	0	接收到 SMBus 主机头序列（从模式）标志。 0：未收到 SMBus 主机的地址； 1：当 SMBTYPE=1 且 ENARP=1 时，收到 SMBus 主机地址。 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
5	SMBDEFAULT	R	0	SMBus 从设备默认地址（从模式）。 0：未收到 SMBus 设备的默认地址； 1：当 ENARP=1 时，收到 SMBus 设备的默认地址。 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
4	GENCALL	R	0	广播呼叫地址（从模式）。 0：未收到广播呼叫地址； 1：当 ENGC=1 时，收到广播呼叫的地址。 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
3	Reserved	-	-	Reserved
2	TRA	R	0	发送/接收标志。 0：接收到数据 1：数据已发送 在整个地址传输阶段的结尾，该寄存器根据地址字节的 R/W 位来设定。 当检测到停止条件（STOPF=1），或者重复的起始条件、或者总线仲裁丢失（ARLO=1），或当 PE=0 时，硬件清除该寄存器。
1	BUSY	R	0	总线忙标志。 0：在总线上无数据通讯 1：在总线上正在极性数据通讯

Bit	Name	R/W	Reset Value	Function
				当检测到 SDA 或 SCL 为低电平时，硬件置位。 当检测到一个停止条件时，硬件清零。 该寄存器指示当前正在进行的总线通讯，当接口被禁用（PE=0）时该信息仍然被更新。
0	MSL	R	0	主从模式。 0：slave 1：master —当接口处于主模式（SB=1）时，硬件置位； —当总线上检测到一个停止条件（STOPF=1）、仲裁丢失（ARLO=1）、或当 PE=0 时，硬件清零。

28.5.8. I2C 时钟控制寄存器(I2C_CCR)

Address offset:0x1C

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	Res	Res	CCR[11:0]											
RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	F/S	RW	0	I2C 主模式选择。 0：标准模式 1：快速模式
14	DUTY	RW	0	快速模式时的占空比。 0：快速模式下：T_{low}/T_{high}=2 1：快速模式下：T_{low}/T_{high}=16/9
13:12	Reserved	-	-	Reserved
11:0	CCR[11:0]	RW	0	快速/标准模式下的时钟控制分频系数（主模式）。 该分频系数用于设置主模式下的 SCL 时钟。 1. 标准模式: ■ T_{high}=CCR x Tpclk ■ T_{low} =CCR x Tpclk 2. 快速模式: ■ DUTY=0: T_{high}=CCR x Tpclk T_{low} =2 x CCR x Tpclk ■ DUTY=1(为达到 400KHz): T_{high}=9 x CCR x Tpclk T_{low} =16 x CCR x Tpclk 注: ■ 允许设定的最小值为 0x04，在快速 DUTY 模式下允许的最小值为 0x01 ■ $T_{high}=t_{r(SCL)}+t_{w(SCLH)}$ ■ $T_{low}=t_{r(SCL)}+t_{w(SCLL)}$ ■ 这些延时没有过滤器 ■ 只有当 PE=0 时才能配置该寄存器; ■

28.5.9. I2C TRISE 寄存器 (I2C_TRISE)

Address offset:0x20

Reset value:0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TRISE[5:0]					
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:6	Reserved	-	-	Reserved
5:0	TRISE	RW	0x0002	在快速/标准模式下的最大上升时间（主模式）。 这些位应该提供在 master mode 下， SCL 反馈回路的最大持续时间。这样做的目的是无论 SCL 上升沿持续时间多少，SCL 都能保持一个稳定的频率。 这些位必须设置为 I2C 总线规范里给出的最大的 SCL 上升时间，增长步幅为 1。 例如：标准模式中最大允许 SCL 上升时间为 1000ns。如果在 I2C_CR2 寄存器中 FREQ[5:0] 中的值等于 0x08，Tpclk=125ns，则 TRISE 中配置为 0x09（1000ns/125ns = 8 + 1 = 9）。 滤波器的值也可以加到 TRISE 内。 如果结果不为整数，则将整数部分写入 TRISE，以确保 t_{HIGH} 参数。 注：当 PE=0 时才能设置该寄存器。

28.5.10. I2C 寄存器映像

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	I2C_CR1	Reserved																SWRST	Reserved	ALERT	PEC	POS	ACK	STOP	START	NOS-	ENGC	ENPEC	ENARP	SMBTYPE	Reserved	SMBUS	PE
	Reset value																	0		0	0	0	0	0	0	0	0	0	0	0		0	0
0x04	I2C_CR2	Reserved																Reserved			LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Reserved		FREQ[5:0]					
	Reset value																				0	0	0	0	0			0	0	0	0	0	0
0x08	I2C_OAR1	Reserved																ADD-	Reserved					ADD[9:8]		ADD[7:1]							ADD0
	Reset value																	0						0	0	0	0	0	0	0	0	0	0
0x0C	I2C_OAR2	Reserved																								ADD2[7:1]							ENDUAL
	Reset value																									0	0	0	0	0	0	0	0

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
0x10	I2C_DR	Reserved																Reserved								DR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x14	I2C_SR1	Reserved																SMBAL-	TIMEOUT	Reserved	PECERR	OVR	AF	ARLO	BERR	TXE	RXNE	Reserved	STOPF	ADD10	BTF	ADDR	SB
	Reset value																	0	0		0	0	0	0	0	0	0		0	0	0	0	0
0x18	I2C_SR2	Reserved																PEC[7:0]								DUALF	SMBHOS	SMBDE-	GEN-	Reserved	TRA	BUSY	MSL
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
0x1C	I2C_CCR	Reserved																F/S	DUTY	Reserved		CCR[11:0]											
	Reset value																	0	0			0	0	0	0	0	0	0	0	0	0	0	0
0x20	I2C_TRISE	Reserved																Reserved										TRISE[5:0]					
	Reset value																											0	0	0	0	1	0

29. 串行外设接口 (SPI)

本项目设计实现了 2 个 SPI 模块，两者的功能完全一样。

29.1. 简介

SPI 接口可以配置为支持 SPI 协议或者支持 I2S 音频协议。 SPI 接口默认工作在 SPI 方式，可以通过软件把功能从 SPI 模式切换到 I2S 模式。

串行外设接口(SPI)允许芯片与外部设备以半/全双工、同步、串行方式通信。此接口可以被配置成主模式，并为外部从设备提供通信时钟(SCK)。接口还能以多主配置方式工作。它可用于多种用途，包括使用一条双向数据线的双线单工同步传输，还可使用 CRC 校验的可靠通信。

I2S 也是一种 3 引脚的同步串行接口通讯协议。它支持四种音频标准，包括飞利浦 I2S 标准， MSB 和 LSB 对齐标准，以及 PCM 标准。它在半双工通讯中，可以工作在主和从 2 种模式下。当它作为主设备时，通过接口向外部的从设备提供时钟信号。

29.2. SPI 主要特征

29.2.1. SPI 主要特征

- 支持 SPI 主机模式和 SPI 从机模式
- 3 线全双工同步传输
- 2 线半双工同步传输（有双向数据线）
- 2 线单工同步传输（无双向数据线）
- 8 位或者 16 位传输帧选择
- 支持多主模式
- 8 个主模式波特率预分频系数（最大为 $f_{PCLK}/2$）
- 从模式频率（最大为 $f_{PCLK}/2$）
- 主模式和从模式下均可以由软件或硬件进行 NSS 管理：主/从操作模式的动态改变
- 可编程的时钟极性和相位
- 可编程的数据顺序，MSB 在前或 LSB 在前
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 支持可靠通讯硬件 CRC
  - 在发送模式下，CRC 值可以被作为最后一个字节发送
  - 在全双工模式中对接收到的最后一个字节自动进行 CRC 校验
- 可引起中断的主模式故障、过载以及 CRC 错误标志
- 2 个具备 DMA 能力的深度为 4，宽度为 16 位（当数据帧设置为 8 位时，宽度为 8 位）的 Rx 和 Tx FIFO

29.2.2. IIS 主要特征

- 单工通讯（仅发送或接收）
- 主或者从操作
- 8 位线性可编程分频器，获得精确的音频采样频率（8KHz 到 96KHz）

- 数据格式可以是 16 位，24 位或者 32 位
- 音频信道固定数据包格式为 16 位（16 位数据帧）或 32 位（16 位、24 位 32 位数据帧）
- 可编程的时钟极性（稳定态）
- 从发送模式下的下溢标志位和主/从接收模式下的溢出标志位
- 16 位数据寄存器用来发送和接收，在通道两端各存在一个寄存器
- 支持 I2S 协议：
  - I2S 飞利浦标准
  - MSB 对齐标准（左对齐）
  - LSB 对齐标准（右对齐）
  - PCM 标准（16 位通道帧上带长帧或者短帧同步或者 16 位数据帧扩展为 32 位通道帧）
- 数据反向总是 MSB 在先
- 发送和接收都有 DMA 能力
- 主时钟可以输出到外部音频设备，比率固定为 256xFs(Fs 为音频采样频率)

29.3. SPI 功能描述

29.3.1. 概述

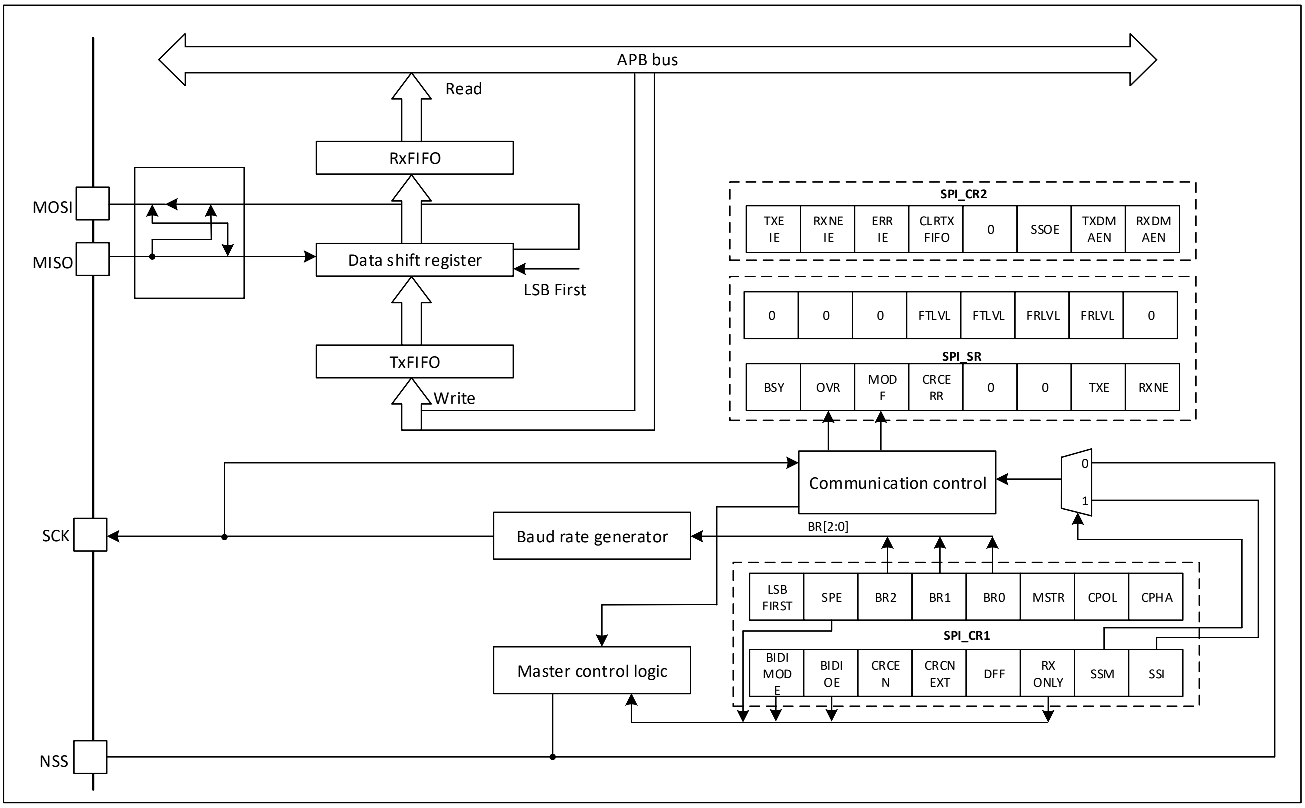


图 29-1 SPI 框图

SPI 通过 4 个引脚与外部器件相连：

MISO：主设备输入/从设备输出引脚。该引脚在从模式下发送数据，在主模式下接收数据。

MOSI：主设备输出/从设备输入引脚。该引脚在主模式下发送数据，在从模式下接收数据。

SCK：串口时钟，作为主设备的输出，从设备的输入。

NSS： 从设备选择。取决于 SPI 和 NSS 的设定，该 pin 可以用作：

- 选择要通讯的从机
- 同步数据帧
- 发现多主机间的冲突

SPI 总线允许在一个主机和一个或者多个从机之间的通讯。总线由至少两根线组成：一个是时钟，另一个是被同步传输的数据。根据应用场景，可以选择增加另外一根数据线和从机 NSS 信号。

29.3.2. 单主机和单从机通信

针对不同的应用场景，SPI 可以使用几种不同的配置进行通讯。这些配置使用 2 线、3 线（软件 NSS）或者 4 线（硬件 NSS）。通讯通常都被主机启动。

29.3.2.1. 全双工通信

缺省情况，SPI 被配置成全双工通讯。在这种配置下，主机和从机的 shift 寄存器，在 MOSI 和 MISO 之间，使用两个单向的线连到一起。在 SPI 通讯期间，数据在主机提供的时钟沿同步的被移位。主机通过 MOSI 发送数据，从 MISO 接收来自从机的数据。当数据帧传输完成（所有 bit 被 shift 完成），在主机和从机之间的信息就被交互过了。

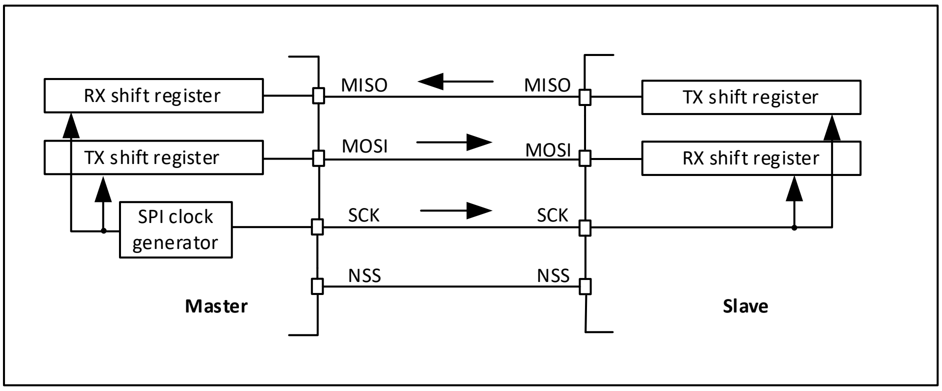


图 29-2 全双工单主机/单从机应用

29.3.2.2. 半双工通信

通过设定 BIDIMODE bit（SPI_CR1 寄存器），SPI 可以工作在半双工模式。在这种配置下，用 1 根数据线完成主机和从机 shift 寄存器的连接。在通讯过程中，在 SCK 的时钟沿，数据在两个 shift 寄存器之间以 BIDIOE（SPI_CR1 寄存器）选择的方向，同步移位。在该配置下，主机的 MISO 和从机的 MOSI 被释放作为通用端口给其他应用使用。

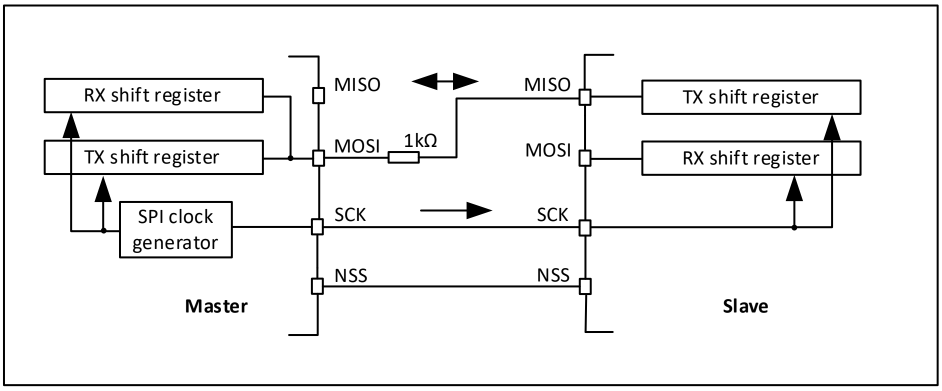


图 29-3 半双工单主机/单从机应用

NSS 可以被使用在主机和从机之间进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

在该配置下，主机的 MISO 引脚和从机的 MOSI 引脚可以用作 GPIO。

29.3.2.3. 单工通信

通过使用 RXONLY（SPI_CR1 寄存器），设定 SPI 在只发送模式或者只接收模式，使 SPI 工作在单工模式下。在这个配置下，在主机和从机的 shift 寄存器之间只使用 1 根线。另一对 MISO 和 MOSI 不被使用，可以被用作标准的 GPIO。

- 只发送模式（RXONLY=0）：配置与全双工相同。应用忽略在未使用的端口上的信息。这个端口可以被用作标准的 GPIO。
- 只接收模式（RXONLY=1）：通过置位 RXONLY，应用可以禁能 SPI 输出功能。在从机配置，MISO 输出被禁能，该端口被用作 GPIO。当他的从机 NSS 信号有效时，从机继续从 MOSI 接收数据。接收到的数据事件是否发生取决于数据 buffer 的配置。在主机配置下，MOSI 输出被禁能，该端口可以用作 GPIO。只要 SPI 被使用，时钟信号就被连续的产生。停止时钟的唯一方法是清零 RXONLY 或者 SPE，直到来自 MISO 的输入完成。并填入数据 buffer。

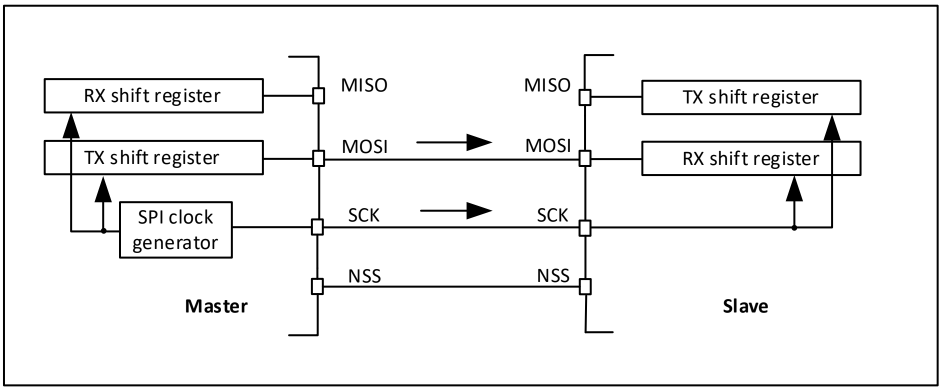


图 29-4 单工单从机/单主机应用

(主设备处于仅发送模式/从设备处于仅接收模式)

（1）在主机和从机之间可以使用 NSS 进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

（2）在 Rx shift 寄存器的输入捕获意外的输入信息。在标准的只发送模式下，所有与传输接收相关的事件都被必须被忽略。

（3）在该配置下，两边的 MISO 引脚都被用作 GPIO。

通过用传输方向的设定（在 BIDIOE bit 未发生改变时，双向模式被使能），任何单工通讯可以被半双工通讯代替。

29.3.3. 多从机通信

在一个有两个或者更多独立从机的配置里，主机为每个从机，使用 GPIO 来管理 NSS。主机必须通过拉低连接从机 NSS 选择某个从机。当完成这个，标准的主机和专门的从机通讯就被建立了。

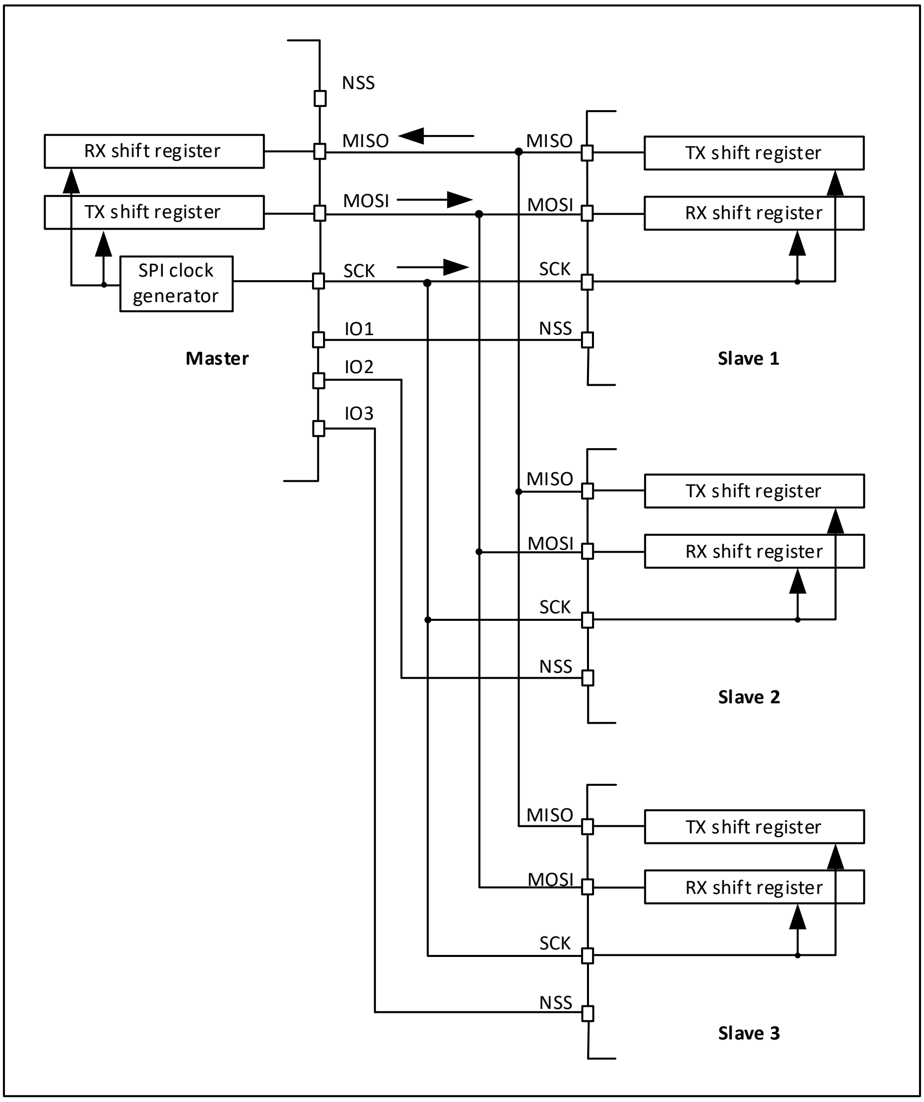


图 29-5 主机与三个独立的从机通信

NSS 在这种配置下在主机端未被使用。必须通过 SSM=1, SSI=1 来防止任何 MODF 错误。

由于从机的 MISO 连接到一起，所有从机必须把他们 MISO 的 GPIO 配置作为开漏模式。

29.3.4. 多主机通信

除非 SPI 总线不是被设计成具备多主机功能，否则用户可以使用其内嵌 feature，该 feature 可以发现两个试图同时控制总线的节点存在的潜在冲突。当需要用到这种检测时，要使用配置为硬件输入模式的 NSS 引脚。

在这种模式下有两个以上 SPI 节点工作的连接是不可能的，因为单次只有一个节点可以在公共数据线上传输。

当节点无效，缺省下两个都保持从机模式。一旦节点要控制总线，它自己切换成主机模式，并把有效的电平经过专门的 GPIO 给予其余节点的从机 NSS 输入。在该进程完成后，有效的从机 选择信号被释放，控制总线的节点暂时返回被动模式，并等待新的进程开始。

如果两个节点在同一时间都给出控制请求，总线冲突事件就会产生（查看 MODF 事件）。然后用户可以应用一些简单的仲裁过程（例如：通过提前定义的给两个节点的不同的超时推迟下一次尝试）

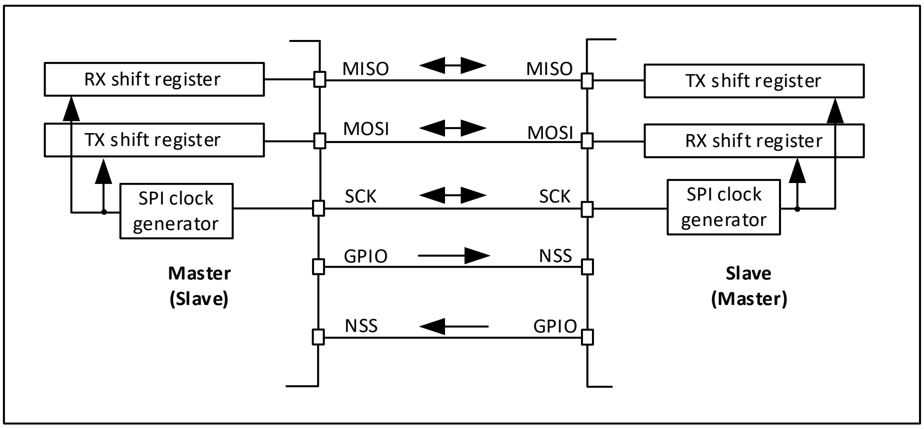


图 29-6 多主机应用

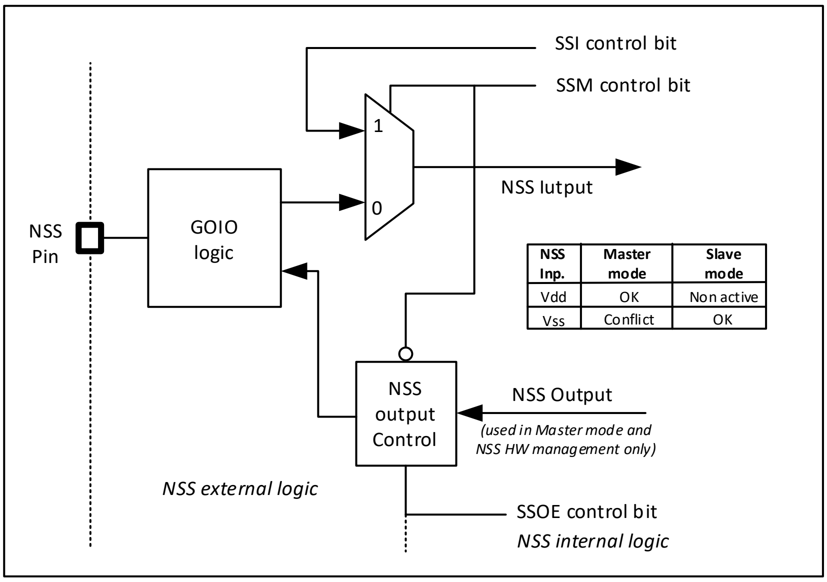
NSS 在两个节点都被配置成硬件输入模式。他的有效电平使能了 MISO 输出控制，而被动节点被配置成从机。

29.3.5. 从选择(NSS)脚管理

在从机模式，NSS 作为标准的片选输入，使从机能与主机通讯。在主机模式，NSS 既可以作为输出又可以作为输入。当作为输入时，它可以防止多主机的总线冲突，当作为输出时，它可以驱动单个从机的从机选择信号。

通过 SPI_CR1 寄存器的 SSM 位，可以选择硬件或者软件从机 NSS 管理：

1. 软件 NSS 管理（SSM=1）：在这个配置下，从机选择信号被内部的 SSI 位（SPI_CR1 寄存器）值驱动。外部 NSS 引脚被释放给其他应用使用。
2. 硬件 NSS 管理（SSM=0）：在这个情况下，有几个可能的配置。
   1）NSS 输出使能（SSM=0, SSOE=1）：这个配置仅在作为主机时使用。硬件管理 NSS 引脚。当 SPI 在主机模式被使能（SPE=1），NSS 信号就被拉低并保持低电平，直到 SPI 被 disable（SPE=0）。在多主机应用中，SPI 不能进行这种 NSS 配置。
   2）NSS 输出 disable（SSM=0, SSOE=0）：如果 MCU 在总线上作为主机，这个配置允许进行多主机能力。如果 NSS 引脚此时被拉低，SPI 进入主机 mode fault 状态，芯片自动被重配置为从机模式。在从机模式，NSS 引脚作为标准的片选输入，当 NSS 为低时，从机被选中。



NSS Inp.	Master mode	Slave mode
Vdd	OK	Non active
Vss	Conflict	OK

图 29-7 硬件/软件从机 NSS 管理

29.3.6. 通讯格式

在 SPI 通讯期间，接收和发送操作同时进行。SCK（serial clock）将数据线上的信息移位和采样操作同步。通讯格式取决于时钟相位、时钟极性和数据帧格式。为了能够进行通讯，主机和从机必须遵循相同的通讯格式。

29.3.6.1. 时钟相位 和极性控制

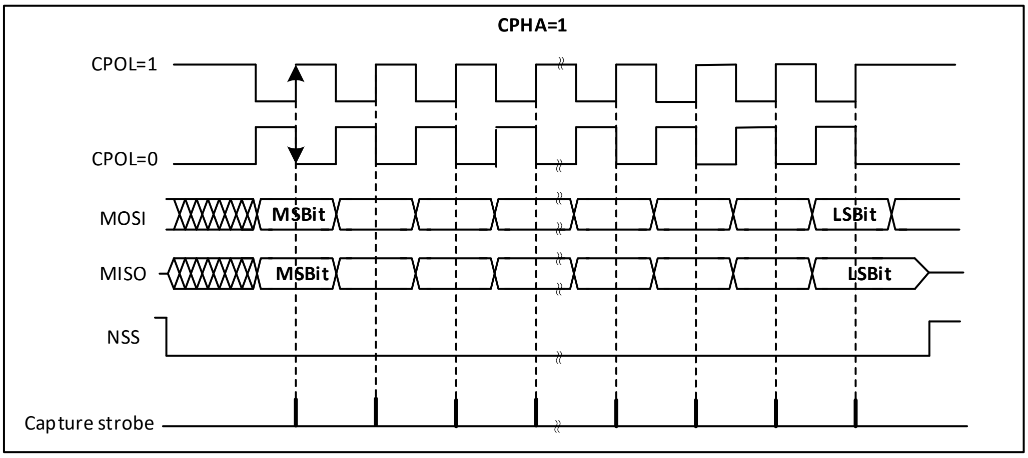
通过 CPOL 和 CPHA 位（SPI_CR1 寄存器），软件可以配置 4 种可能的时序。CPOL（clock polarity）控制当没有数据传输时的 clock 的 IDLE 状态。该位对主机和从机都有影响。如果 CPOL 被复位，SCK 引脚有低电平的状态。如果 CPOL 被置位，SCK 引脚有高电平的 IDLE 状态。

如果 CPHA 被置位，SCK 的第二个边沿捕获传输的第一个数据位（如果 CPOL 被复位，是下降沿，否则是上升沿）。在时钟变化类型的出现，数据被锁存。如果 CPHA 被复位，SCK 的第一个边沿捕获第一个传输的数据位（如果 CPOL 被置位，是下降沿，否则是上升沿）。在该时钟变化类型出现时，数据被锁存。

CPOL 和 CPHA 的组合选择了数据捕获时钟边沿。

在 CPOL/CPHA 改变之前，SPI 必须被 disable（SPE=0）。

SCK 的 IDLE 状态必须对应被 SPI_CR1 寄存器选择的极性。



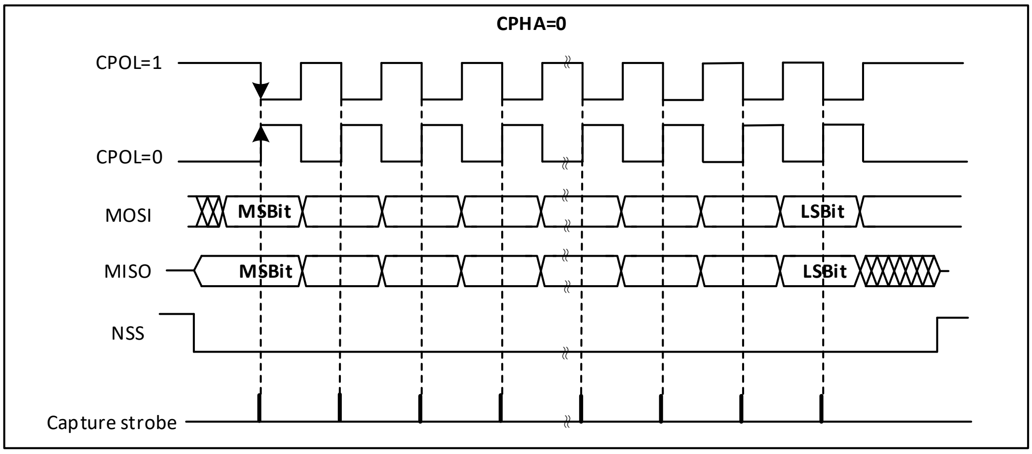


图 29-8 数据时钟时序图

数据位的顺序取决于 LSBFIRST 位的设定。

29.3.6.2. 数据帧格式

通过 LSBFIRST 位(SPI_CR1 寄存器)，SPI shift 寄存器可以设定为 MSB-FIRST 或者 LSB-FIRST。通过使用 DFF 位(SPI_CR2 寄存器)，选择数据帧的位数。可选择为 8 位或者 16 位长度，该设置对于发送和接收都适用。

29.3.7. SPI 配置

对于主机和从机，SPI 的配置流程几乎一样。对于具体的模式建立，遵循专门的章节介绍。当进行标准的通讯，进行以下步骤：

1） 写相关的 GPIO 寄存器：配置 MOSI、MISO 和 SCK 引脚
2） 写 SPI_CR1 寄存器
   - 通过 BR[2:0]配置时钟波特率（从机模式不需要）
   - 配置 CPOL 和 CPHA
   - 通过 RXONLY 或者 BIDIMODE 和 BIDIOE（RXONLY 和 BIDIMODE 不能同时有效），选择单工或者半双工模式
   - 配置 LSBFIRST

- 配置 SSM 和 SSI
- 配置 MSTR 位（在多主机 NSS 配置中，如果主机被配置防止 MODF 错误，要避免 NSS 的冲突状态）
- 配置 DFF，选择数据帧位数

3） 写 SPI_CR2 寄存器

配置 SSOE（从机模式不需要）

写相应的 DMA 寄存器：配置 DMA 的 SPI Tx 和 Rx 通道

29.3.8. SPI 使能流程

推荐在主机发送时钟之前使能 SPI 从机。如果不这样处理，不期望的数据传输可能会发生。从机的数据寄存器必须在开始与主机通讯之前，已经包含要被发送的数据（或者在通讯时钟的第一个沿，或者如果时钟信号是连续的情况，要在正在进行的通讯结束之前）。SCK 信号必须在 SPI 从机被使能之前，固定在 IDLE 状态电平（相应的被选择极性）。

全双工模式（或者只发送模式），当 SPI 被使能并且 TXFIFO 不空，或者向 TXFIFO 进行下一个写， 主机开始通讯。

在任何主机只接收 模式（RXONLY=1，或者 BIDIMODE=1 且 BIDIOE=0），在 SPI 被使能后，主机开始通讯，时钟立即被提供。

对于 DMA 处理，遵从具体的章节内容。

29.3.9. 数据传输和接收流程

29.3.9.1. RXFIFO and TXFIFO

SPI 所有数据通讯都通过 FIFO，深度为 4，宽度为 16 位/8 位。该特性使 SPI 能够以连续数据流进行工作，并防止由于 CPU 来不及处理数据导致的通讯问题。发送和接收有独立的 FIFO，叫做 TXFIFO 和 RXFIFO。这些 FIFO 被用在所有的 SPI 模式。

FIFO 的处理取决于多种参数，包括：数据交换模式（全双工、半双工）、数据帧格式。

读 SPI_SR 寄存器会得到最早存放在 RXFIFO 中还未被读走的数据结果。写 SPI_DR 寄存器，会在 FIFO 发送队列的最后位置，存入被写的数据。读写访问通常必须与数据宽度对齐。 FTLVL[1:0]和 FRLVL[1:0]位显示了两个 FIFO 当前的占用级别。

对 SPI_DR 寄存器的都访问必须通过 RXNE 事件管理。当有数据存储在 RXFIFO 时，该事件被触发。当 RXNE 被清零，RXFIFO 就被认为是空的。

相似地，写要发送的数据帧，通过 TXE 事件管理。当 TXFIFO Level 小于或者等于总容量的一半时，该事件就会被触发。否则，TXE 被清零，并且 TXFIFO 被认为是满的。

用这样的方式，RXFIFO 和 TXFIFO 都可以存 4 个数据帧。

TXE 和 RXNE 事件都可以通过查询、中断和 DMA 方式处理。

当 RXFIFO 满时，如果下一个数据被接收，则 overrun 事件产生。Overrun 事件可以通过查询和中断的方式处理。

被置位的 BSY 位显示了 1 个当前数据帧的通讯正在进行。当时钟信号连续的提供，在主机端的两个数据帧之间，BSY 标志保持置位。但在从机端的每个数据帧传输之间，BSY 会保持最小 1 个 SPI Clock 宽度的低电平。

某些应用场景下，当向 TXFIFO 中写入数据，可以通过设置 CLRTXFIFO 位，清空 TXFIFO 数据，从而重新往 TXFIFO 里写新的数据重新进行通信。

29.3.9.2. 序列处理

一些数据帧可以通过单序列传递来完成一条信息。当发送被使能，当主机的 TXFIFO 里有任何数据，序列开始并继续进行。时钟信号被主机连续的提供，直到 TXFIFO 空，然后停止等待额外的数据。

在只接收模式，即半双工（BIDIMODE=1, BIDIOE=0）或者单工模式（BIDIMODE=0，RXONLY=1），在 SPI 被使能并且只接收模式被激活的时候，主机就立即开始接收。主机一直会提供时钟并连续地接收数据，直到主机停止了 SPI 或者关闭了只接收 模式。

当主机能够以连续的模式（SCK 信号是连续的）提供所有通讯，主机必须要考虑从机处理数据流的能力。当有必要时，主机必须降低通讯速度，并提供或者更慢的时钟，或者分开的帧，或者重组 delay 的数据包。要注意的是，对于主机或者从机来说，没有 underflow 错误信号，来自于从机的数据通常被主机交互和处理（即使从机不能及时准备好数据）。对于从机，更好的方法是使用 DMA，尤其当数据帧小，且波特率高的情况。

每个序列都必须被 NSS 脉冲包住，同时在多从机系统钟选择要进行通讯的其中的一个从机。在一个单从机系统，没有必要用 NSS 去控制从机，但通常也最好提供脉冲，使从系统与每个数据序列的开头同步。NSS 可以由软件和硬件两种方式管理。

当 BSY 被置位，它显示了正在进行的数据帧交互。当专门的帧交互被完成时，RXNE 标志置位。最后一个 bit 被采样，并且整个数据帧被存在 RXFIFO 中。

29.3.9.3. 关闭 SPI 流程

当 SPI 被关闭掉，必须按照特定的关闭 流程。对于系统关闭 SPI 的流程是很重要的，因为此后应用上，外设时钟会被停掉，系统进入低功耗模式。这种情况下（关闭），正在进行的交互会被破坏。在一些模式下，关闭流程是唯一停止连续通讯的办法。

全双工或者只发送模式下，主机可以当停止提供要发送的数据时完成交互。在这种情况下，在最后的数据交互后，时钟被停止。在这些模式下，SPI 被关闭之前，用户必须使用标准的关闭 流程。当 SPI 被关闭在主机发送时，如果此时一个帧交互正在进行，或者下一个数据帧存在 TXFIFO 中，SPI 的功能是不能被保证的。

当主机处在任何只接收模式，停止连续时钟的唯一方法是停止外设（SPE=0）。该模式下，要进行专门的 SPI 关闭流程。

当 SPI 被关闭，接收到未读走的数据存放在 RXFIFO 中，这些数据必须在下一次 SPI 使能要开始新的序列之前被处理掉。为防止有未读的数据，要确保当 SPI 被关时，RXFIFO 是空的（使用正确的关闭 流程，或者通过用软件复位以初始化所有的 SPI 寄存器）。

标准的关闭流程是基于 BSY 状态，并查看 FTLVL[1:0]，以确保传输彻底完成。也可以通过特定别的检查来鉴别正在进行交互的结束，例如：

1. 当 NSS 信号被软件管理，主机要向从机提供正确的 NSS 脉冲。 或者
2. 当完成来自 DMA 或者 FIFO 的交互数据流时，此时最后的数据帧仍在传输过程中。

正确的关闭流程是（只接收模式除外）：

- 等待 FTLVL[1:0]=00（没有数据要发送）
- 等待 BSY=0（最后的数据被处理完成）
- 关闭 SPI（SPE=0）

- 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

对于特定只接收模式，正确的关闭流程是：

1. 在最后一个数据帧传输过程中，通过关闭 SPI（SPE=0），打断接收流程
2. 等待 BSY=0（最后的数据帧已被处理）
3. 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

29.3.9.4. 使用 DMA 通讯

为以最大的速度工作，并加快数据的读/写过程，以避免 overrun，SPI 具备简单请求/应答协议的 DMA 能力。

当 TXE 或者 RXNE 被置位，会产生 DMA 请求。Tx 和 Rx buffer 有独立的请求。

1. 发送时，每次 TXE 置为 1，则产生 DMA 请求。然后 DMA 会向 SPI_DR 寄存器写入数据。
2. 接收时，每次 RXNE 置为 1，则产生 DMA 请求。然后 DMA 读 SPI_DR 寄存器的数据。

当 SPI 被仅用作发送数据，可以只使能 SPI Tx DMA 通道。在这个情况下，因为被接收到的数据没有被读走，OVR 标志位被置位。当 SPI 被仅用作接收数据，可以使能 SPI Rx DMA 通道。

在发送时，当 DMA 已经写入了所有要发送的数据（DMA_ISR 寄存器的 TCIF 标志位被置位），可以通过监测 BSY 标志来确保 SPI 通讯已完成。这是用来避免当关闭 SPI 或者进入 stop 模式时，最后的传输被破坏。软件必须先等待 FTLVL[1:0]=00，然后再等待 BSY=0。

当开始使用 DMA 通讯时，为避免 DMA 通道管发生错误事件，必须遵从以下步骤：

1. 使能 DMA Rx buffer（SPI_CR2 的 RXDMAEN 位）（如果 Rx DMA 被使用）
2. 使能 Tx Rx DMA streams（在 DMA 寄存器里）（如果 steams 被用到）
3. 使能 DMA Tx buffer（在 SPI_CR2 寄存器的 TXDMAEN 位）（如果 Tx DMA 被使用）
4. 通过 SPE 位使能 SPI

强制用以下步骤关闭通讯：

- 关闭 DMA Tx Rx streams（在 DMA 寄存器里）（如果 stream 被使用）
- 通过 SPI 关闭流程关闭 SPI
- 通过清除 TXDMAEN 和 RXDMAEN（SPI_CR2 寄存器），关闭 DMA Tx 和 Rx buffer（如果 DMA Tx 和 Rx 被使用）

29.3.9.5. 通讯时序

本节介绍一些典型的时序，这些时序对于查询、中断或者 DMA 都是有效的。为了简化，假定 LSBFIRST=0, CPOL=0, CPHA=1。也不提供完整的 DMA 操作配置。

- 当 NSS 有效，SPI 被使能，从机开始控制 MISO；当 NSS 被释放或者 SPI 关闭时从机失去对 MISO 的控制。对于从机，在传输开始前必须提供充足的时间给主机，以便提前准备数据。

在主机端，仅在 SPI 被使能时，SPI 外设会控制 MOSI 和 SCK 信号（也包括 NSS 信号）。如果 SPI 被关闭，SPI 外设就从 GPIO 断开，因此在这些线上的电平值取决于 GPIO 的设定。

- 在主机端，如果通讯是连续的，则 BSY 在帧之间保持有效。在从机端，BSY 信号在数据帧之间通常变低至少一个时钟周期。
- 只有当 TXFIFO 是满的，TXE 信号才被清零。
- 在 TXDMAEN bit 一被置位，DMA 仲裁过程即开始。在 TXEIE 被置位后，产生 TXE 中断。当 TXE 信

号有效时，开始向 TxFIFO 传输数据，直到 TxFIFO 变满，或者 DMA 传输完成。

- 如果所有要被发送的数据能装进 TxFIFO，在 SPI 总线传输前 DMA Tx TCIF 标志就会被拉高。这个标志在 SPI 交互完成之前，一直都为高。

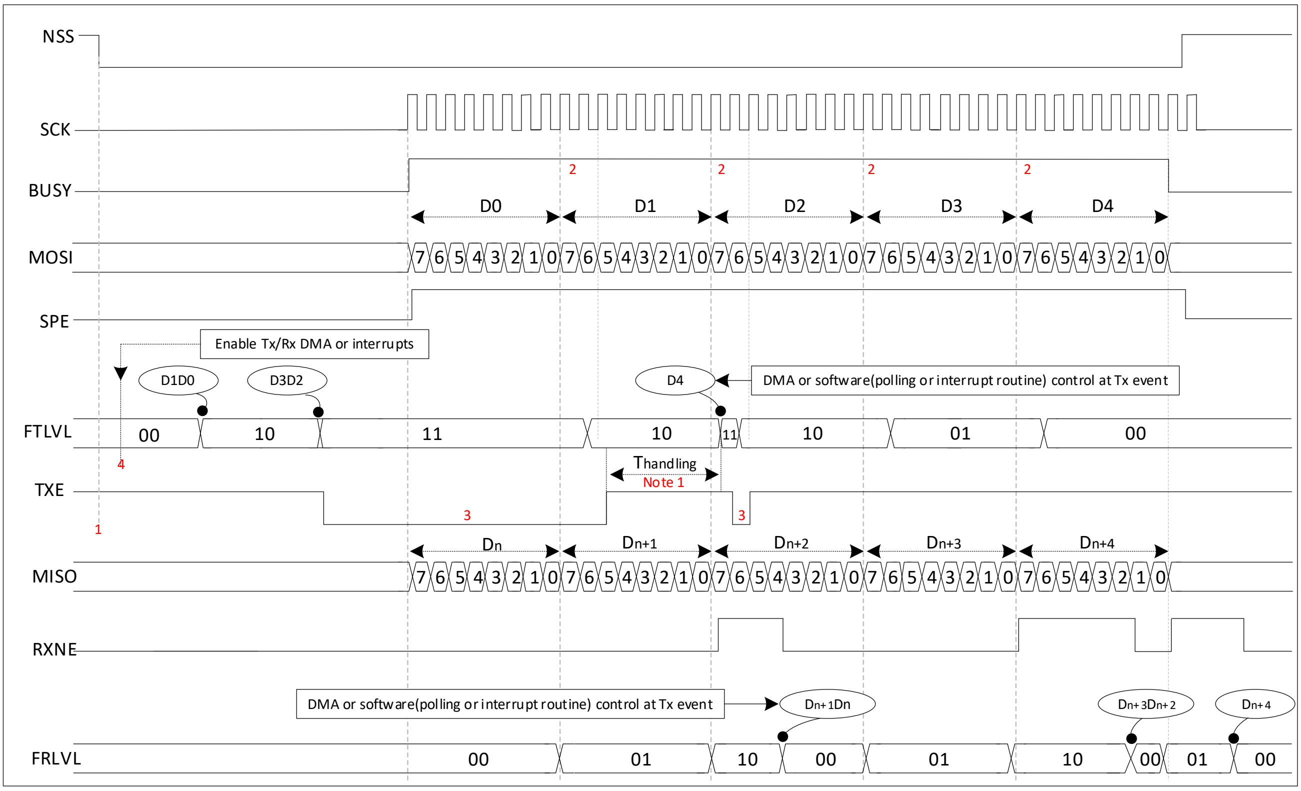


图 29-9 主机全双工通讯时序 (数据宽度=8, FRXTH=0)

29.3.10. 状态标志

应用程序通过 3 个状态标志可以完全监控 SPI 总线的状态。

29.3.10.1. 发送缓冲空标志 (TXE)

当 TXFIFO 有足够的空间存放要发送的数据时，TXE 标志位被置位。TXE 标志位与 TXFIFO level 有关。该标志位变高并保持高电平，直到 TXFIFO level 小于等于 1/2 FIFO 深度才会被硬件清零。如果 TXEIE（SPI_CR2）被置位，则会产生中断请求。当 TXFIFO level 大于 1/2，该位被自动清零。

29.3.10.2. 接收缓冲非空标志(RXNE)

RXNE 标志位被置位:

RXNE 变高并保持高电平，直到 RXFIFO level 大于或者等于 1/4FIFO。

当上述条件不再成立，则 RXNE 被硬件自动清零。

29.3.10.3. 忙标志(BSY)

BSY 标志由硬件设置与清除(写入此位无效果)，此标志表明 SPI 通信层的状态。

当它被设置为'1'时，表明 SPI 正忙于通信，但有一个例外：在主模式的双向接收模式下(MSTR=1、BDM=1 并且 BDOE=0)，在接收期间 BSY 标志保持为低。

在软件要关闭 SPI 模块并进入停机模式(或关闭设备时钟)之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输，因此需要严格按照下述过程执行。

BSY 标志还可以用于在多主机系统中避免写冲突。

除了主模式的双向接收模式(MSTR=1、BDM=1 并且 BDOE=0)，当传输开始时， BSY 标志被置'1'。

以下情况该标志将被清除为'0'：

1. 当 SPI 被正确的关闭掉
2. 主机模式，当产生 MODF=1
3. 主机模式，当传输完成，不再有效数据要发送
4. 从机模式，在每个数据传输之间，BSY 标志置为 0，并保持至少一个 SPI 时钟周期

注意: 不要使用 BSY 标志处理每个数据发送和接收。使用 TXE 和 RXNE 更合适。

29.3.11. 错误标志

29.3.11.1. 主模式失效(MODF)

主模式失效（MODF）仅发生在：当 NSS 作为输入信号（SSOE=0），NSS 引脚硬件模式管理下，主设备的 NSS 脚被拉低；或者在 NSS 引脚软件模式管理下，SSI 位被置为'0'时。此时，MODF 位被自动置位。主模式失效对 SPI 设备有以下影响：

5. MODF 位被置为'1'，如果设置了 ERRIE 位，则产生 SPI 中断；
6. SPE 位被清为'0'。这将停止一切输出，并且关闭 SPI 接口；
7. MSTR 位被清为'0'，因此强迫此设备进入从模式。

下面的步骤用于清除 MODF 位：

1. 当 MODF 位被置为'1'时，执行一次对 SPI_SR 寄存器的读或写操作；
2. 然后写 SPI_CR1 寄存器。

在有多个 MCU 的系统中，为了避免出现多个从设备的冲突，必须先拉高该主设备的 NSS 脚，再对 MODF 位进行清零。在完成清零之后， SPE 和 MSTR 位可以恢复到它们的原始状态。

出于安全的考虑，当 MODF 位为'1'时，硬件不允许设置 SPE 和 MSTR 位。

通常配置下，从设备的 MODF 位不能被置为'1'。然而，在多主配置里，一个设备可以在设置了 MODF 位的情况下，处于从设备模式；此时，MODF 位表示可能出现了多主冲突。中断程序可以执行一个复位或返回到默认状态来从错误状态中恢复。

29.3.11.2. 过载模式

当数据被主机或者从机接收，并且 RXFIFO 没有足够的空间存接收到的数据时，产生 overrun 情况。如果软件或者 DMA 没有足够的时间读走以前接收到的数据（RXFIFO 中存放），该情况就会发生。

当 overrun 情况发生，新收到的数据不会覆盖以前存放在 RXFIFO 的数据。接收到的新数据被忽略，并且所有接下来发送的数据丢失。

依次读出 SPI_DR 寄存器后 SPI_SR 寄存器可将 OVR 清除。

29.3.12. SPI 中断

表 29-1 SPI 中断请求

中断事件	事件标志	使能控制位
TXFIFO 等待被装载	TXE	TXEIE
数据接收到 RXFIFO 中	RXNE	RXNEIE
主模式失效事件	MODF	ERRIE
溢出错误	OVR	ERRIE

29.3.13. CRC 计算

CRC 校验用于保证通信的可靠性，数据发送和数据接收分别使用单独的 CRC 计算器。通过对每一个接收位进行可编程的多项式运算来计算 CRC。SPI 提供独立于帧数据长度的 CRC8 或 CRC16 计算，可固定为 8 位或 16 位。 对于所有其他数据帧长度，没有可用的 CRC。

29.3.13.1. CRC 原理

在 SPI 启用（SPE = 1）之前，通过设置 SPIx_CR1 寄存器中的 CRCEN 位来启用 CRC 计算。 CRC 值是使用每个位上的奇数可编程多项式计算的。 该计算在由 SPIx_CR1 寄存器中的 CPHA 和 CPOL 位定义的采样时钟边沿上进行处理。 计算出的 CRC 值在数据块结束时自动检查，并且由 CPU 或 DMA 管理其传输。 当检测到根据接收数据内部计算的 CRC 与发送器发送的 CRC 不匹配时，设置 CRCERR 标志以指示数据错误。 处理 CRC 计算的正确程序取决于 SPI 配置和选择的传输管理方式。

29.3.13.2. CPU 控制的 CRC 传输

开始并连续通信直到发送或接收 SPIx_DR 寄存器中最后一个数据帧。然后必须在 SPIx_CR1 寄存器中设置 CRCNEXT 位，用于指示在当前处理的数据帧后开始传输 CRC 帧。 CRCNEXT 位必须在最后一个数据帧传输结束之前置位。 在 CRC 传输期间停止 CRC 计算。

接收到的 CRC 数据以字节或字的形式存储在 RXFIFO 中。这就是为什么仅在 CRC 模式下，必须将接收缓冲区视为单个 16 位缓冲区，用于一次仅接收一个数据帧。CRC 格式的传输通常在数据传输结束时需要多出一个数据帧。但是，当设置一个通过 16 位 CRC 校验的 8 位数据帧时，需要多两帧才能发送完整的 CRC 值。

当接收到最后一个 CRC 数据时，将执行自动校验，比较接收到的值和 SPIx_RXCRC 寄存器。软件必须检查 SPIx_SR 寄存器中的 CRCERR 标志以确定数据传输是否出错。软件通过向其写入“0”来清除 CRCERR 标志。CRC 接收后，CRC 值存储在 RXFIFO 中，必须读 SPIx_DR 寄存器以清除 RXNE 标志

29.3.13.3. DMA 控制的 CRC 传输

当使用带 CRC 的 DMA 模式启用 SPI 通信时，通信结束时 CRC 的发送和接收是自动完成的（在仅接收模式下读取 CRC 数据除外）， CRCNEXT 位不必由软件处理。 SPI 传输 DMA 通道的计数器必须设置为要传输的数据帧数，其中不包括 CRC 帧。在接收端，接收到的 CRC 值在传输结束时由 DMA 自动处理，但用户必须注意从 RXFIFO 中清除接收到的 CRC 信息，因为它总是存放到 RXFIFO 中。在全双工模式下，接收 DMA 通道的计数器可以设置为要接收的包括 CRC 在内的数据帧的数量。

在只接收模式下，DMA 接收通道计数器应该只包含传输的数据量，而不包括 CRC 计算。然后基于 DMA 的完整传输，因为它在此模式下作为单个缓冲区工作，所以所有 CRC 值必须由软件从 FIFO 读回。在数据和 CRC 传输结束时，如果在传输过程中出错，SPIx_SR 寄存器中的 CRCERR 标志会被置位。

29.4. IIS 功能描述

29.4.1. 概述

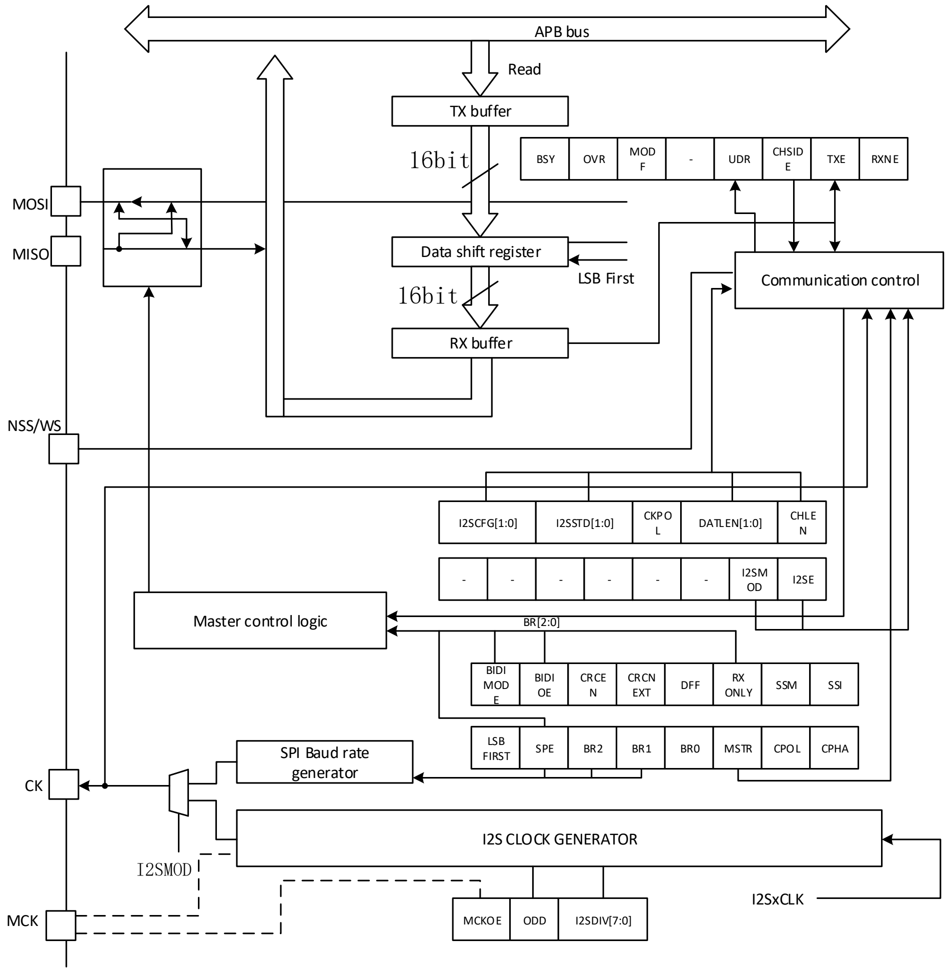


图 29-10 IIS 框图

通过将寄存器 SPI_I2SCFGR 的 I2SMOD 位置为‘1’，即可使能 I2S 功能。此时，可以把 SPI 模块用作 I2S 音频接口。 I2S 接口与 SPI 接口使用大致相同的引脚、标志和中断。

I2S 与 SPI 共用 3 个引脚：

- ✓ SD：串行数据(映射至 MOSI 引脚)，用来发送和接收 2 路时分复用通道的数据；
- ✓ WS：字选(映射至 NSS 引脚)，主模式下作为数据控制信号输出，从模式下作为输入；
- ✓ CK：串行时钟(映射至 SCK 引脚)，主模式下作为时钟信号输出，从模式下作为输入。

在某些外部音频设备需要主时钟时，可以另有一个附加引脚输出时钟：

- ✓ MCK：主时钟(独立映射)，在 I2S 配置为主模式，寄存器 SPI_I2SPR 的 MCKOE 位为‘1’时，作为输出额外的时钟信号引脚使用。输出时钟信号的频率预先设置为 256 × Fs，其中 Fs 是音频信号的采样频率。

设置成主模式时， I2S 使用自身的时钟发生器来产生通信用的时钟信号。这个时钟发生器也是主时钟输出的时钟源。 I2S 模式下有 2 个额外的寄存器，一个是与时钟发生器配置相关的寄存器 SPI_I2SPR，另一个是 I2S 通用配置寄存器 SPI_I2SCFGR(可设置音频标准、从/主模式、数据格式、数据包帧、时钟极性等参数)。

在 I2S 模式下不使用寄存器 SPI_CR1 和所有的 CRC 寄存器。同样， I2S 模式下也不使用寄存器 SPI_CR2 的 SSOE 位，和寄存器 SPI_SR 的 MODF 位和 CRCERR 位。

I2S 使用与 SPI 相同的寄存器 SPI_DR 用作 16 位宽模式数据传输。

29.4.2. 音频协议

三线总线支持 2 个声道上音频数据的时分复用：左声道和右声道，但是只有一个 16 位寄存器用作发送或接收。因此，软件必须在对数据寄存器写入数据时，根据当前传输中的声道写入相应的数据；同样，在读取寄存器数据时，通过检查寄存器 SPI_SR 的 CHSIDE 位来判明接收到的数据属于哪个声道。左声道总是先于右声道发送数据(CHSIDE 位在 PCM 协议下无意义)。有四种可用的数据和包帧组合。可以通过以下四种数据格式发送数据：

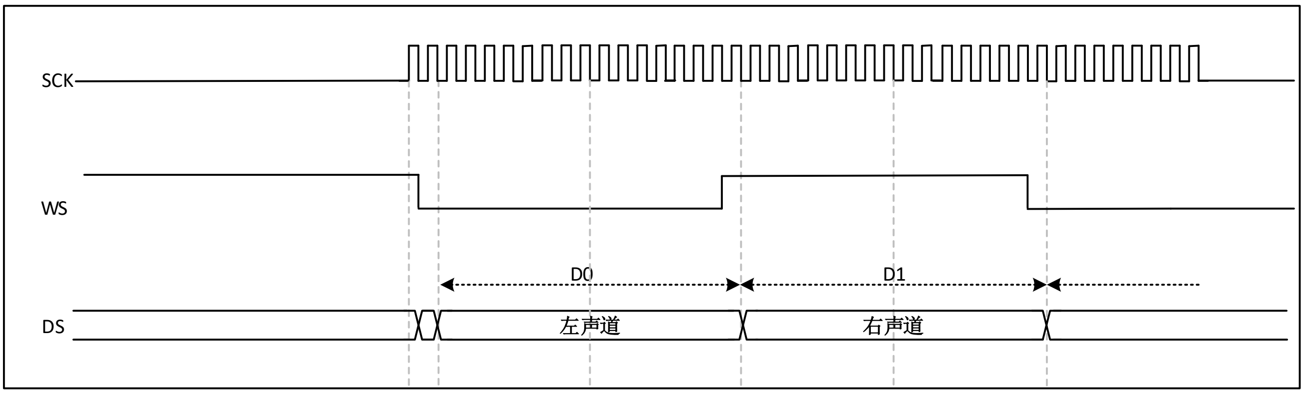
- ✓ 16 位数据打包进 16 位帧
- ✓ 16 位数据打包进 32 位帧
- ✓ 24 位数据打包进 32 位帧
- ✓ 32 位数据打包进 32 位帧

在使用 16 位数据扩展到 32 位帧时，前 16 位(MSB)是有意义的数据，后 16 位(LSB)被强制为 0，该操作不需要软件干预，也不需要有 DMA 请求(仅需要一次读/写操作)。24 位和 32 位数据帧需要 CPU 对寄存器 SPI_DR 进行 2 次读或写操作，在使用 DMA 时，需要 2 次 DMA 传输。对于 24 位数据，扩展到 32 位后，最低 8 位由硬件置 0。对于所有的数据格式和通讯标准，总是先发送最高位(MSB)。

I2S 接口支持四种音频标准，可以通过设置寄存器 SPI_I2SCFGR 的 I2SSTD[1:0]位和 PCMSYNC 位来选择。

29.4.2.1. I2S 飞利浦协议

在此标准下，引脚 WS 用来指示正在发送的数据属于哪个声道。在发送第一位数据(MSB)前 1 个时钟周期，该引脚即为有效。



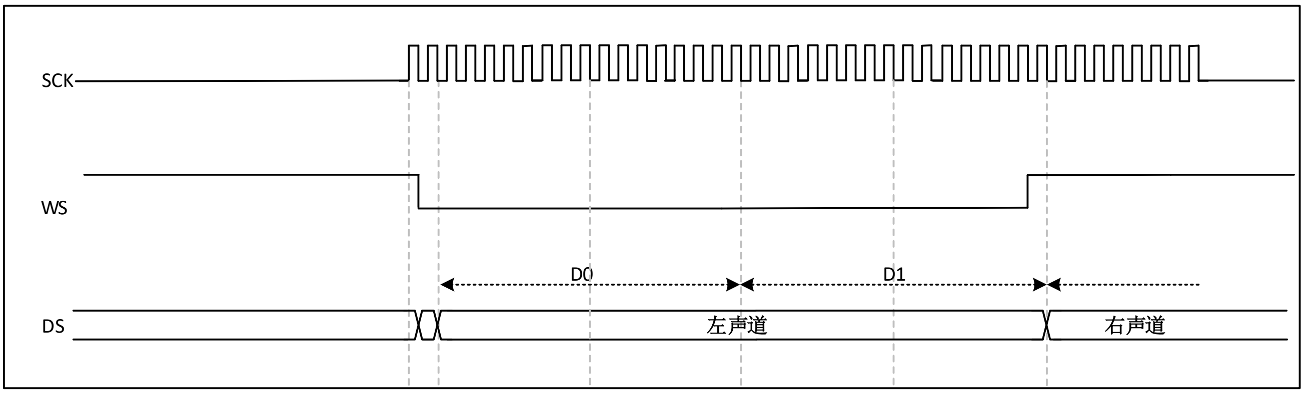
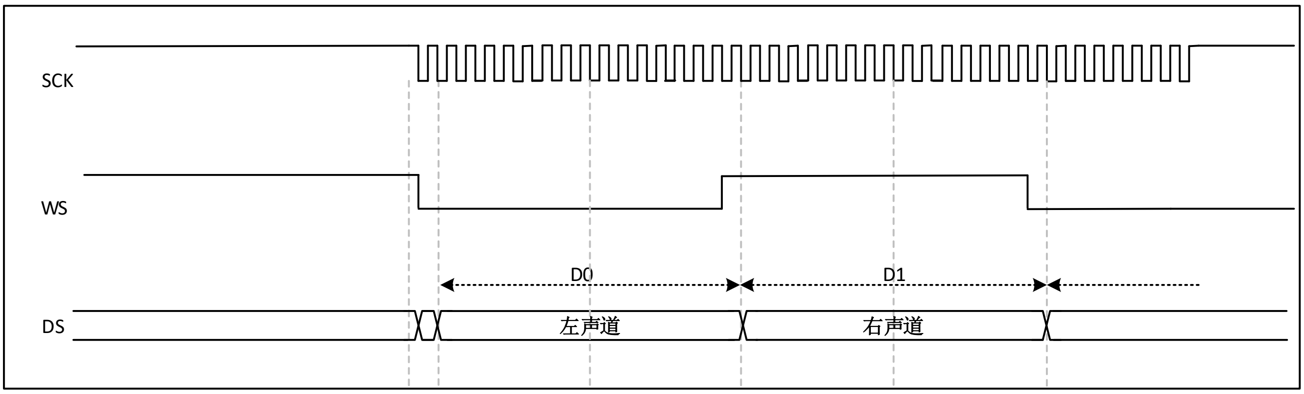


图 29-11 I2S 飞利浦协议波形(16/32 位全精度， CKPOL = 0)



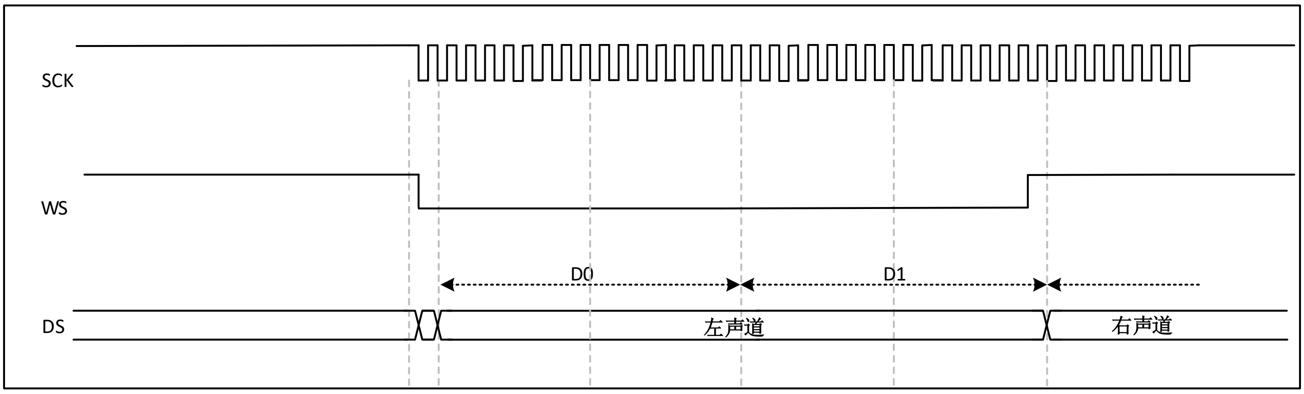


图 29-12 I2S 飞利浦协议波形(16/32 位全精度， CKPOL = 1)

发送方在时钟信号(CK)的下降沿改变数据，接收方在上升沿读取数据。 WS 信号也在时钟信号的下降沿变化。

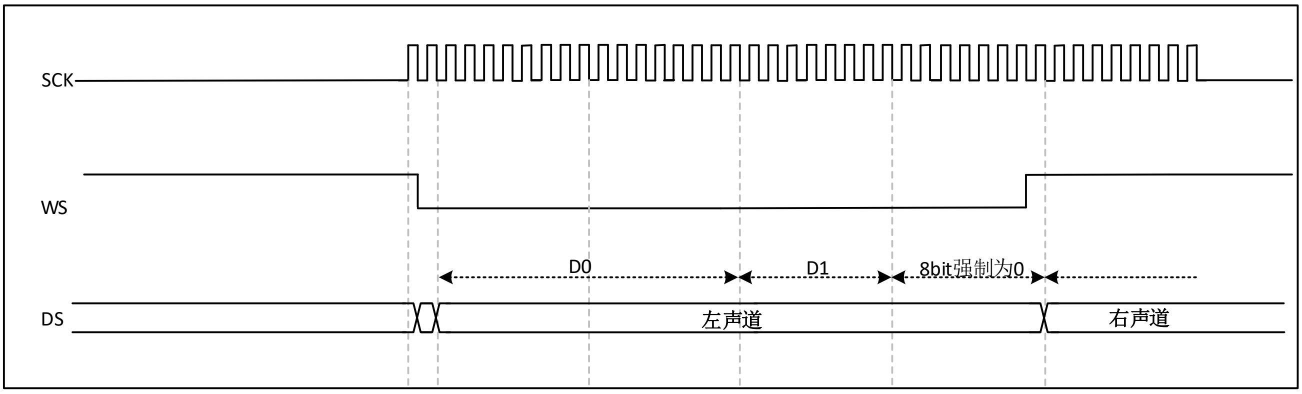


图 29-13 I2S 飞利浦协议标准波形(24 位帧， CKPOL = 0)

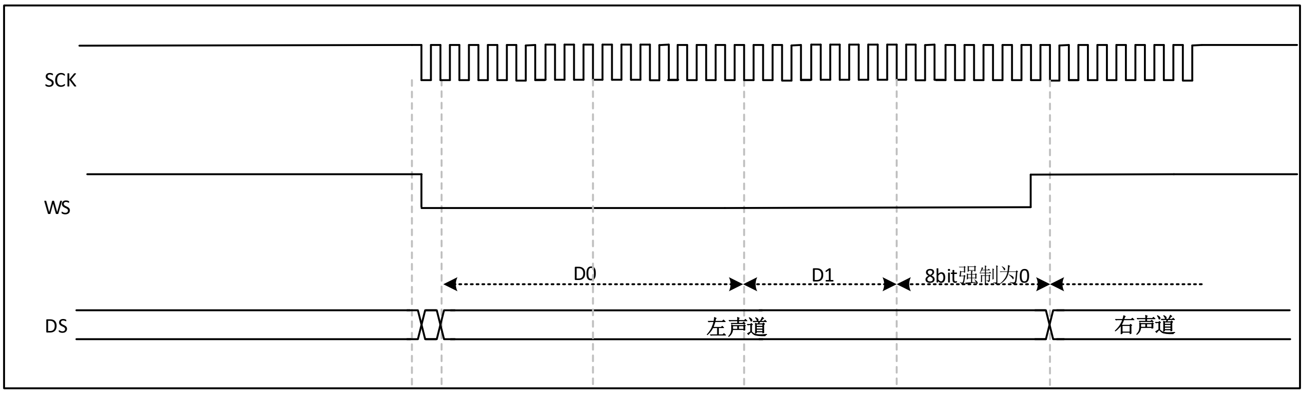
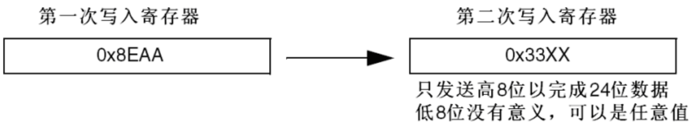


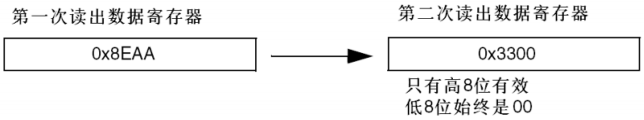
图 29-14 I2S 飞利浦协议标准波形(24 位帧， CKPOL = 1)

此模式需要对寄存器 SPI_DR 进行 2 次读或写操作。

✓ 在发送模式下：如果需要发送 0x8EAA33(24 位)：



✓ 在接收模式下：如果接收 0x8EAA33：



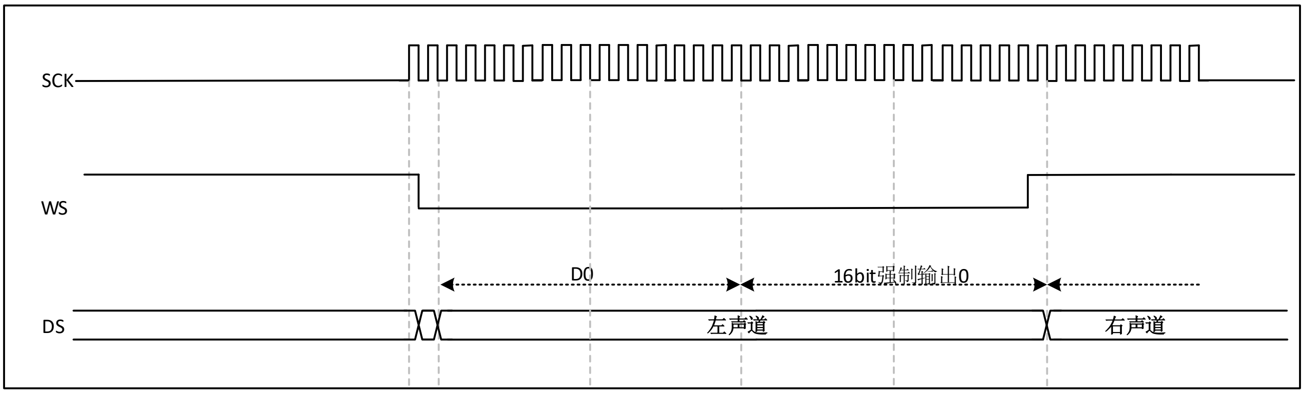


图 29-15 I2S 飞利浦协议标准波形(16 位扩展至 32 位包帧， CKPOL = 0)

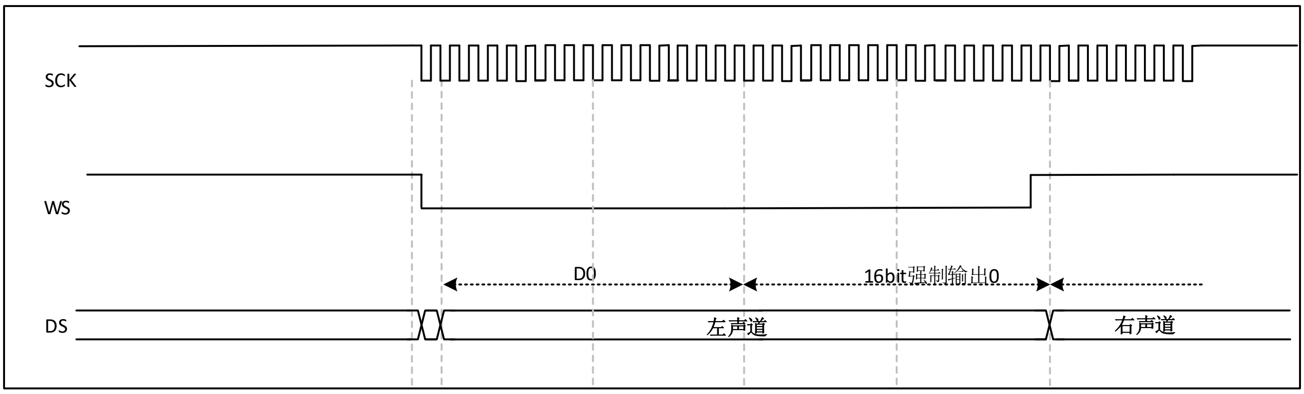


图 29-16 I2S 飞利浦协议标准波形(16 位扩展至 32 位包帧， CKPOL = 1)

在 I2S 配置阶段，如果选择将 16 位数据扩展到 32 位声道帧，只需要访问一次寄存器 SPI_DR。用来扩展到 32 位的低 16 位被硬件置为 0x0000。

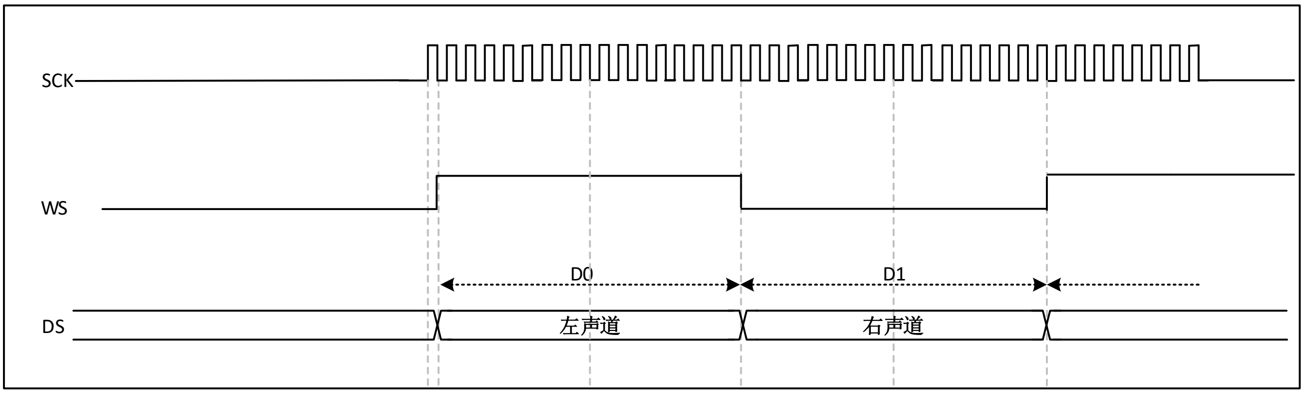
如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x76A30000)，需要的操作如下图所示。

在发送时需要将 MSB 写入寄存器 SPI_DR；标志位 TXE 为‘1’表示可以写入新的数据，如果允许了相应的中断，则可以产生中断。发送是由硬件完成的，即使还未发送出后 16 位的 0x0000，也会设置 TXE 并产生相应的中断。接收时，每次收到高 16 位半字(MSB)后，标志位 RXNE 置‘1’，如果允许了相应的中断，则可以产生中断。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

29.4.2.2. MSB 对齐标准

在此标准下， WS 信号和第一个数据位，即最高位(MSB)同时产生。



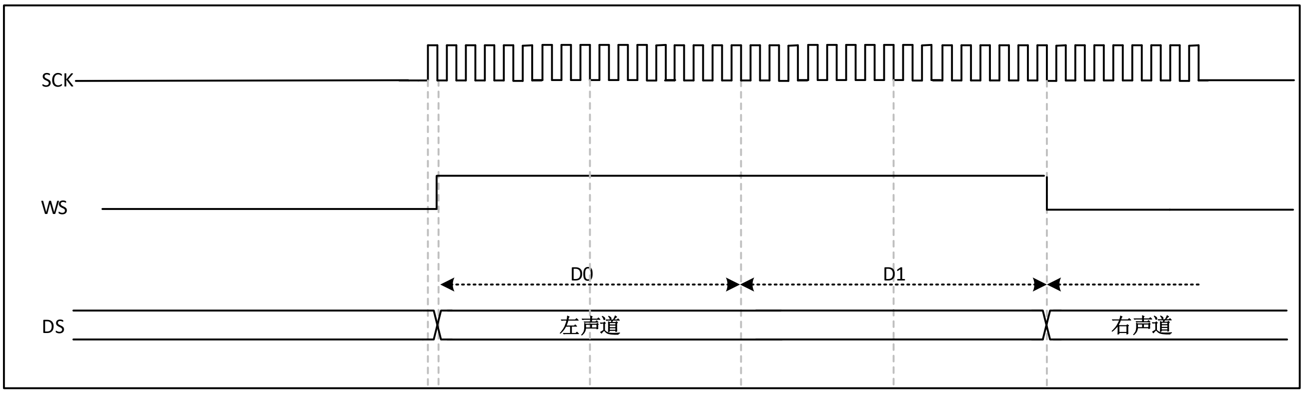


图 29-17 MSB 对齐 16 位或 32 位全精度， CPKOL = 0

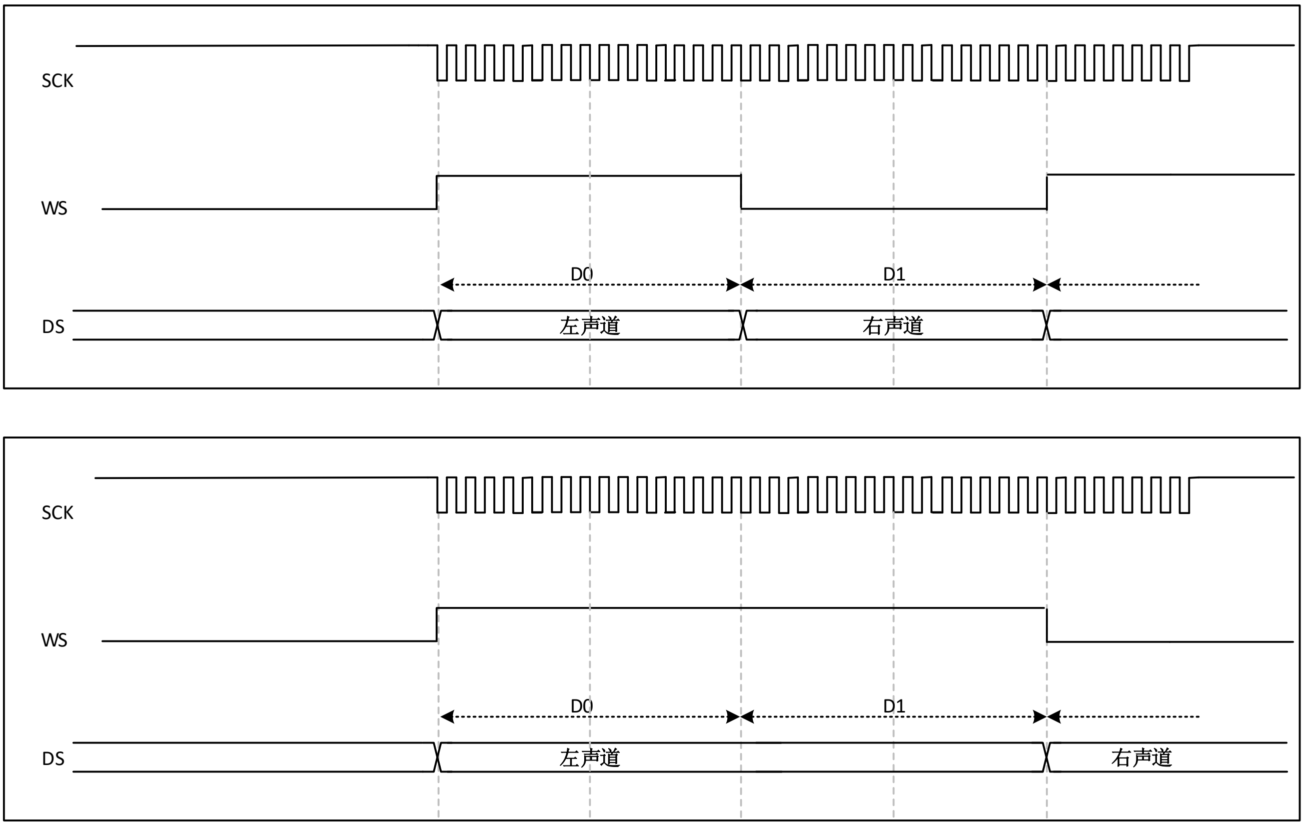


图 29-18 MSB 对齐 16 位或 32 位全精度， CPKOL = 1

发送方在时钟信号(CK)的下降沿改变数据；接收方是在上升沿读取数据。

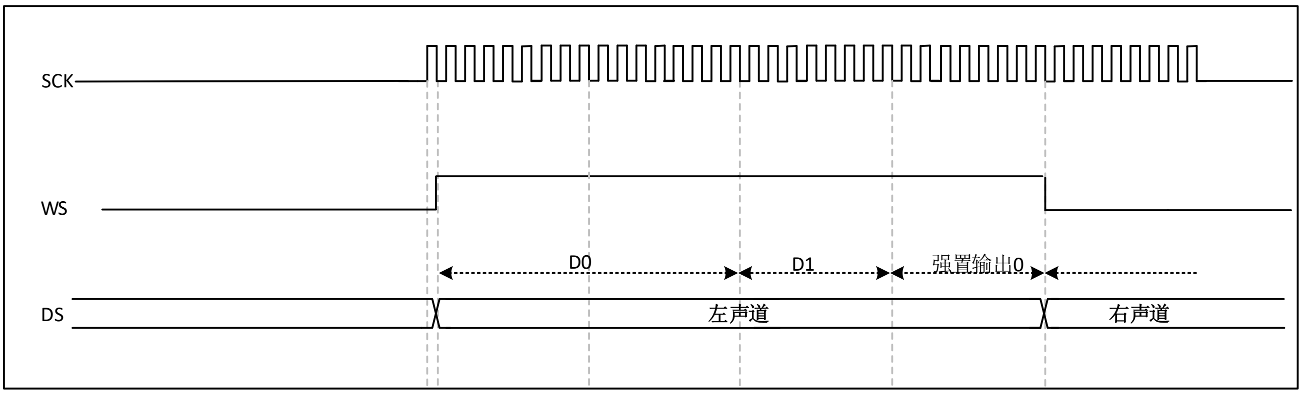


图 29-19 MSB 对齐 24 位数据， CKPOL = 0

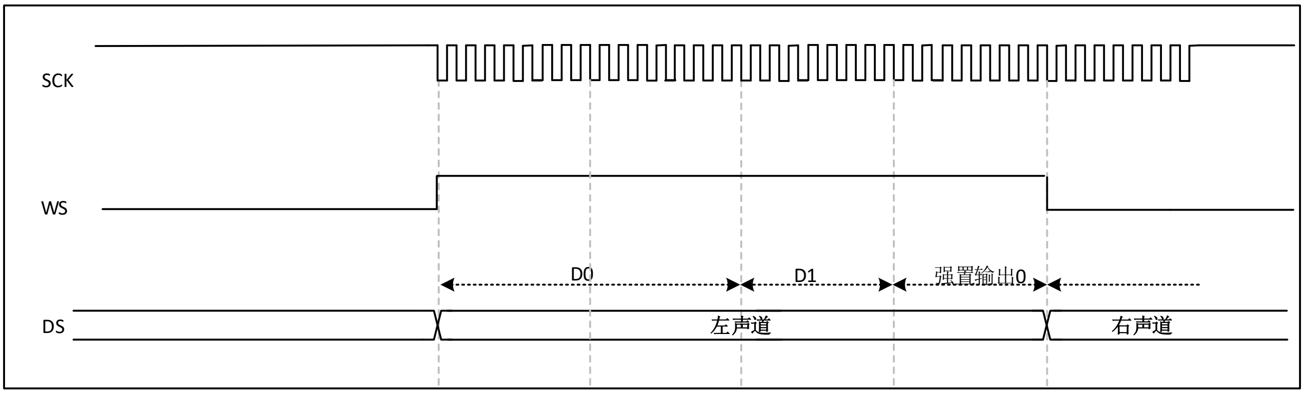


图 29-20 MSB 对齐 24 位数据， CKPOL = 1

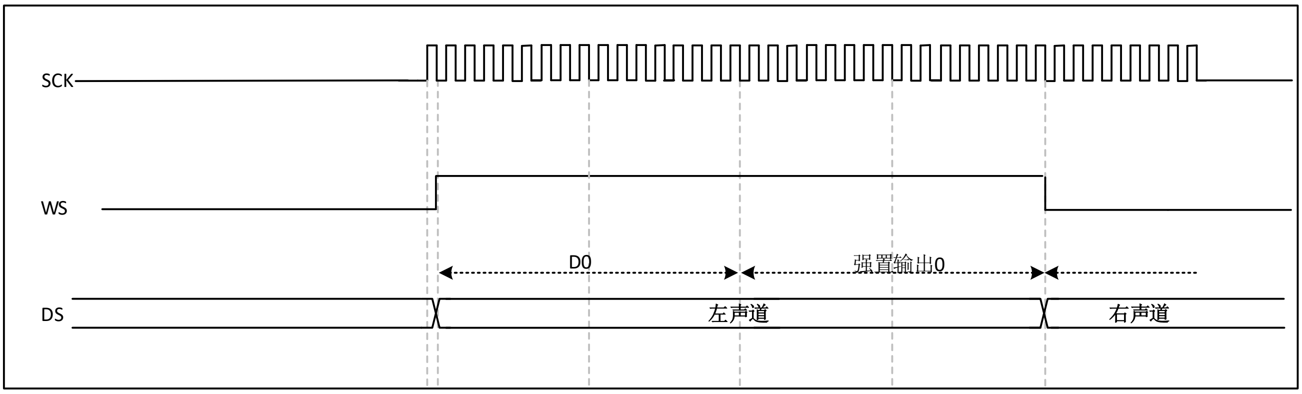


图 29-21 MSB 对齐 16 位数据扩展到 32 位包帧， CKPOL = 0

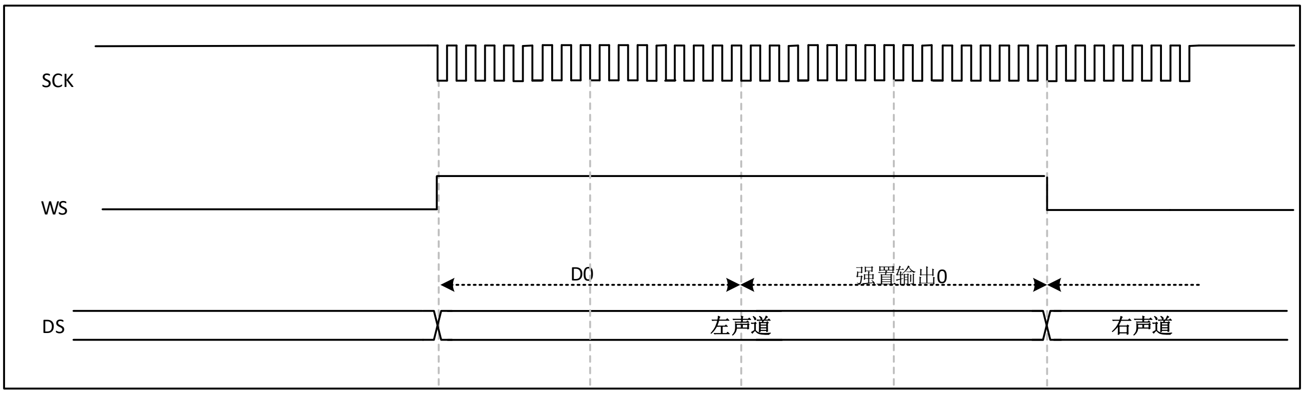


图 29-22 MSB 对齐 16 位数据扩展到 32 位包帧， CKPOL = 1

一旦有效数据开始从 SD 引脚送出，即发生下一次 TXE 事件。在接收时，一旦接收到有效数据(而不是 0x0000 部分)，即发生 RXNE 事件。

29.4.2.3. LSB 对齐标准

此标准与 MSB 对齐标准类似(在 16 位或 32 位全精度帧格式下无区别)。

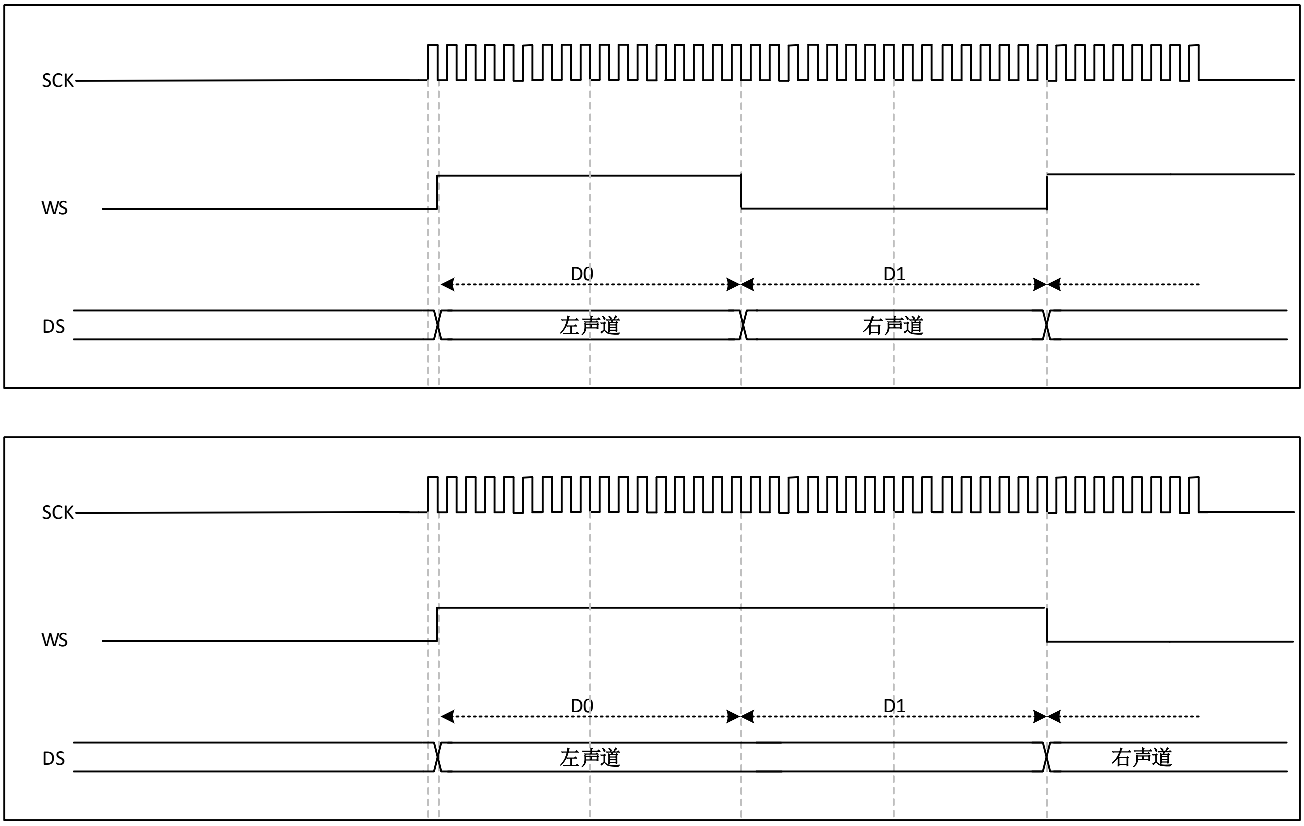


图 29-23 LSB 对齐 16 位或 32 位全精度， CKPOL = 0

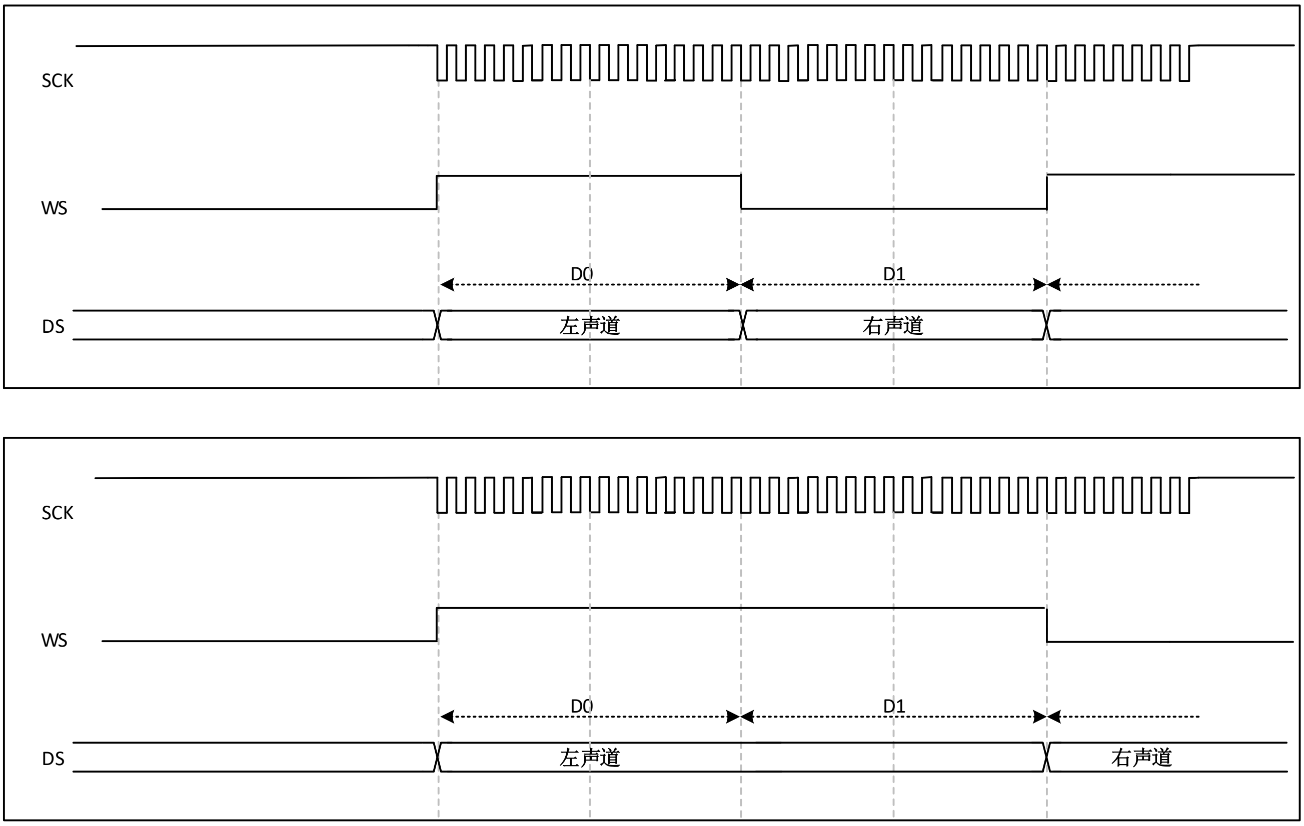


图 29-24 LSB 对齐 16 位或 32 位全精度， CKPOL = 1

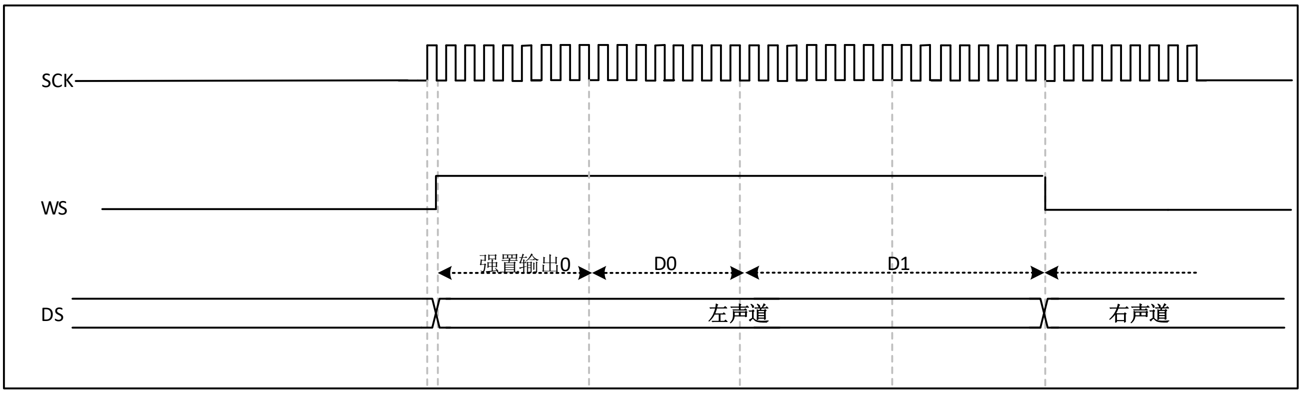


图 29-25 LSB 对齐 24 位数据， CKPOL = 0

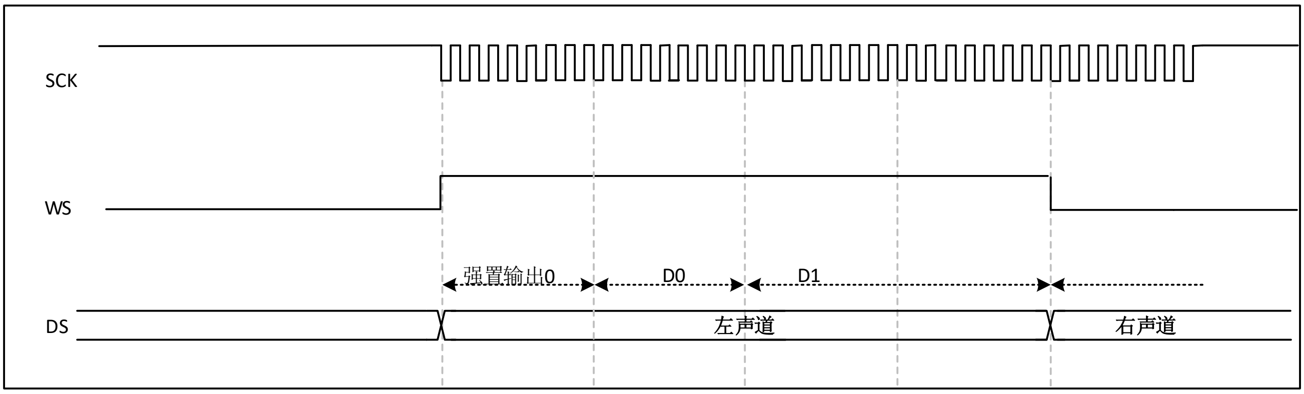
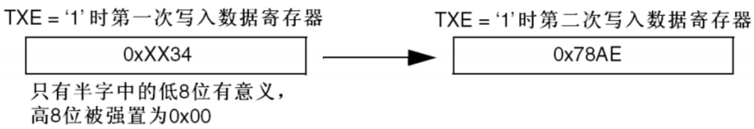


图 29-26 LSB 对齐 24 位数据， CKPOL = 1

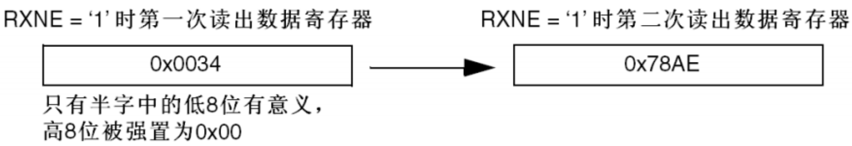
- ✓ 在发送模式下

如果要发送数据 0x3478AE，需要通过软件或者 DMA 对寄存器 SPI_DR 进行 2 次写操作。操作流程如下图所示。要求发送 0x3478AE 的操作



- ✓ 在接收模式下

如果要接收数据 0x3478AE，需要在 2 个连续的 RXNE 事件发生时，分别对寄存器 SPI_DR 进行 1 次读操作。要求接收 0x3478AE 的操作



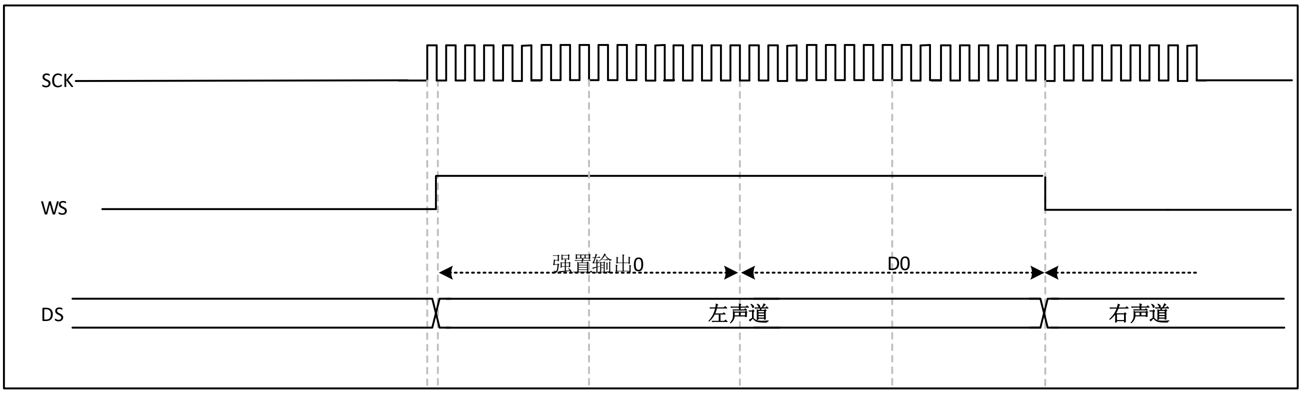


图 29-27 LSB 对齐 16 位数据扩展到 32 位包帧， CKPOL = 0

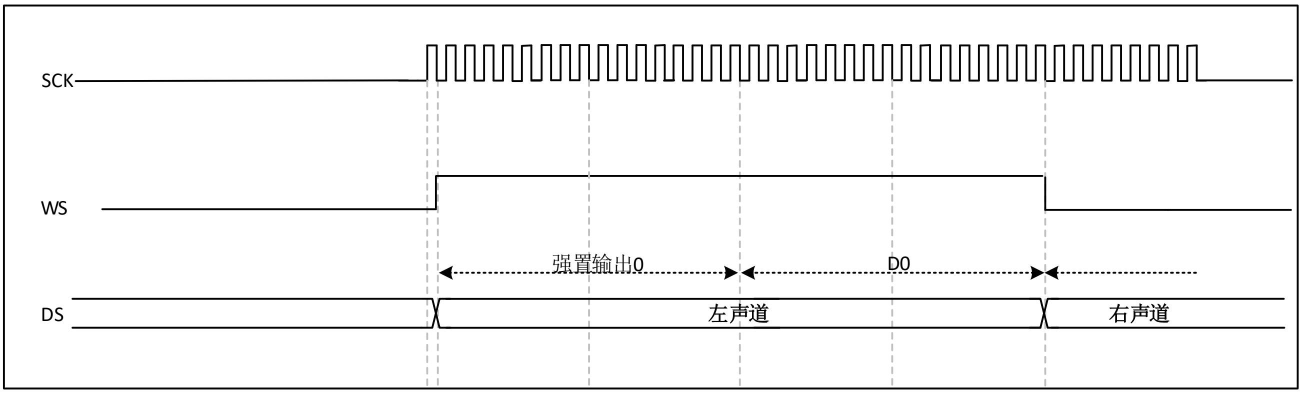


图 29-28 LSB 对齐 16 位数据扩展到 32 位包帧， CKPOL = 1

在 I2S 配置阶段，如果选择将 16 位数据扩展到 32 声道帧，只需要访问一次寄存器 SPI_DR。此时，扩展到 32 位后的高半字(16 位 MSB)被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x0000 76A3)，需要的操作如下图所示。

在发送时，如果 TXE 为‘1’，用户需要写入待发送的数据(即 0x76A3)。用来扩展到 32 位的 0x0000，部分由硬件首先发送出去，一旦有效数据开始从 SD 引脚送出，即发生下一次 TXE 事件。

在接收时，一旦接收到有效数据(而不是 0x0000 部分)，即发生 RXNE 事件。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

29.4.2.4. PCM 标准

在 PCM 标准下，不存在声道选择的信息。 PCM 标准有 2 种可用的帧结构，短帧或者长帧，可以通过设置寄存器 SPI_I2SCFGR 的 PCMSYNC 位来选择。

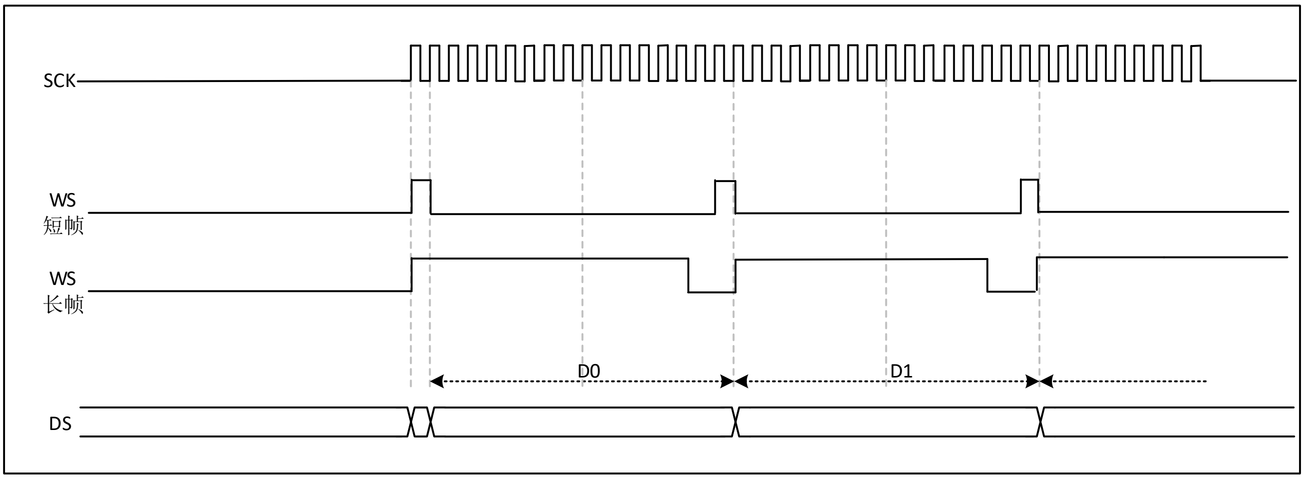


图 29-29 PCM 标准波形(16 位 CKPOL=0)

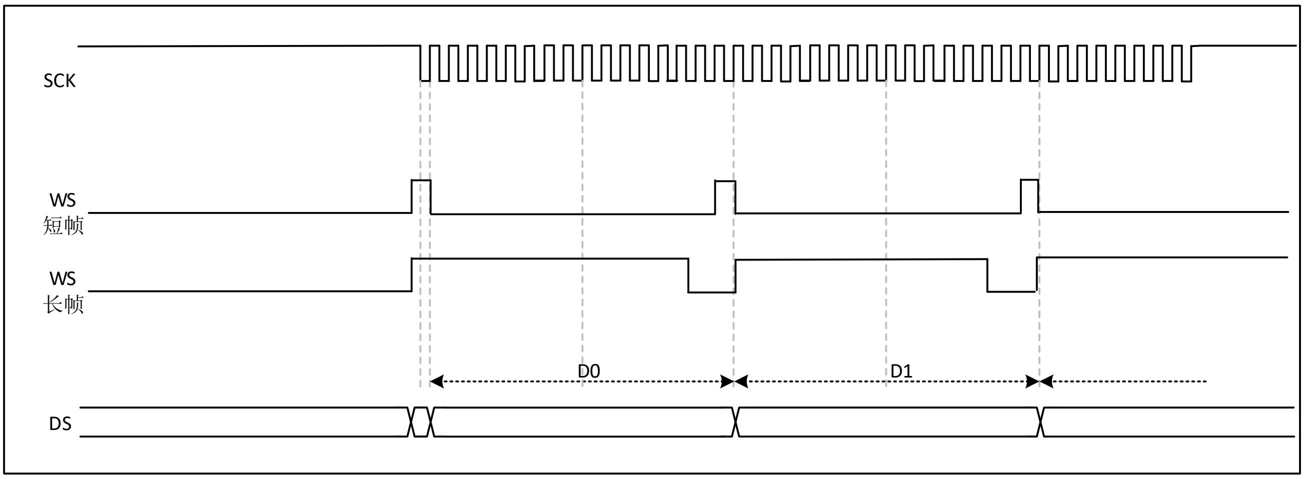


图 29-30 PCM 标准波形(16 位 CKPOL=1)

对于长帧，主模式下，用来同步的 WS 信号有效的时间固定为 13 位。

对于短帧，用来同步的 WS 信号长度只有 1 位。

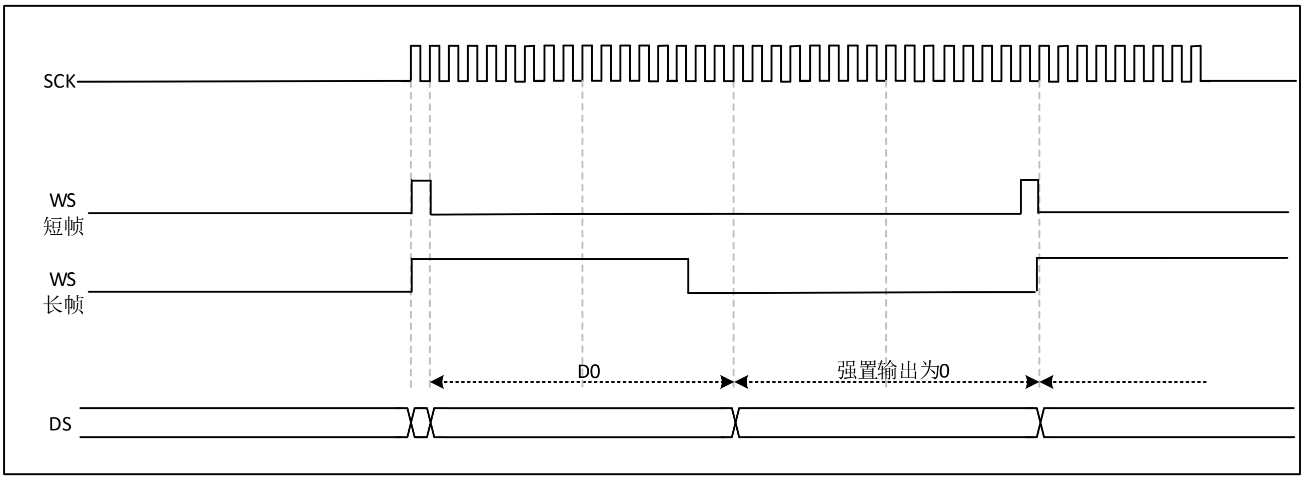


图 29-31 PCM 标准波形(16 位扩展到 32 位包帧 CKPOL=0)

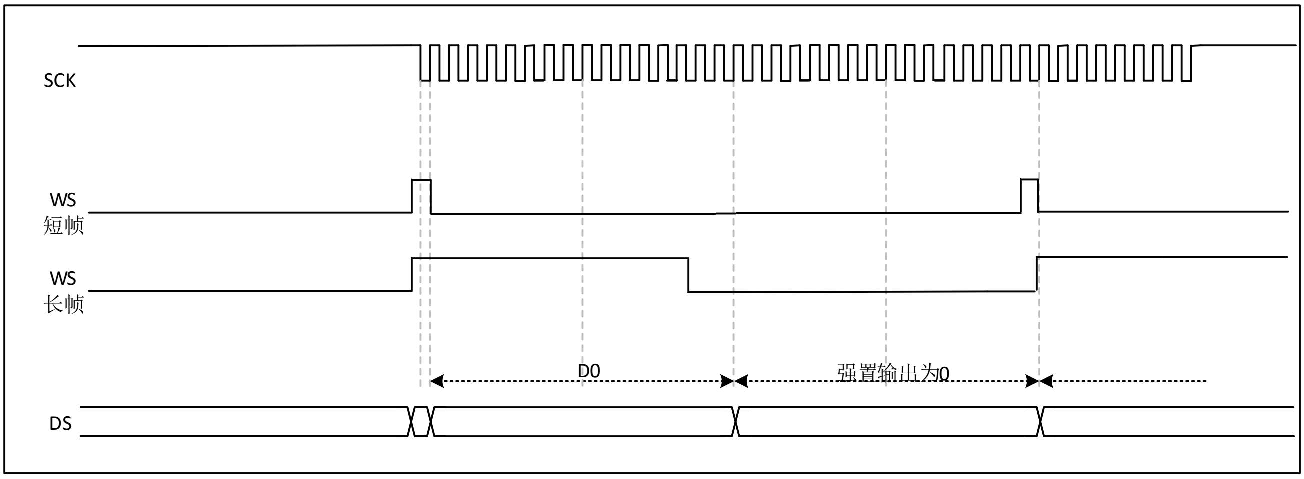


图 29-32 PCM 标准波形(16 位扩展到 32 位包帧 CKPOL=1)

注意：无论哪种模式(主或从)、哪种同步方式(短帧或长帧)，连续的 2 帧数据之间和 2 个同步信号之间的时间差，(即使是从模式)需要通过设置 SPI_I2SCFGR 寄存器的 DATLEN 位和 CHLEN 位来确定。

29.4.3. 时钟发生器

I2S 的比特率即确定了在 I2S 数据线上的数据流和 I2S 的时钟信号频率。即

I2S 比特率 = 每个声道的比特数×声道数目×音频采样频率

对于一个具有左右声道和 16 位音频信号，I2S 比特率计算如下：

I2S 比特率 = 16×2×Fs;

如果包长为 32 位，则有：I2S 比特率=32×2×Fs

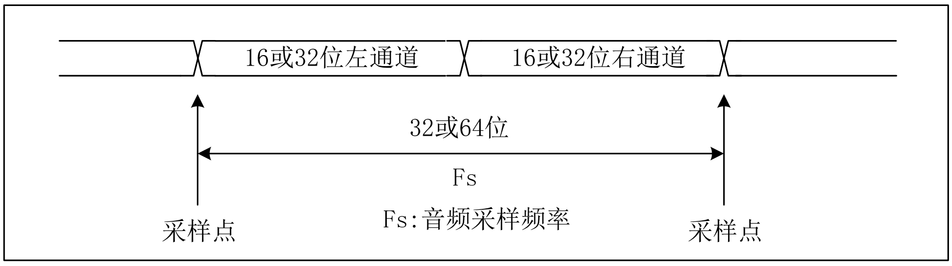


图 29-33 音频采样定义

在主模式下，为了获得需要的音频频率，需要正确地对线性分频器进行设置。

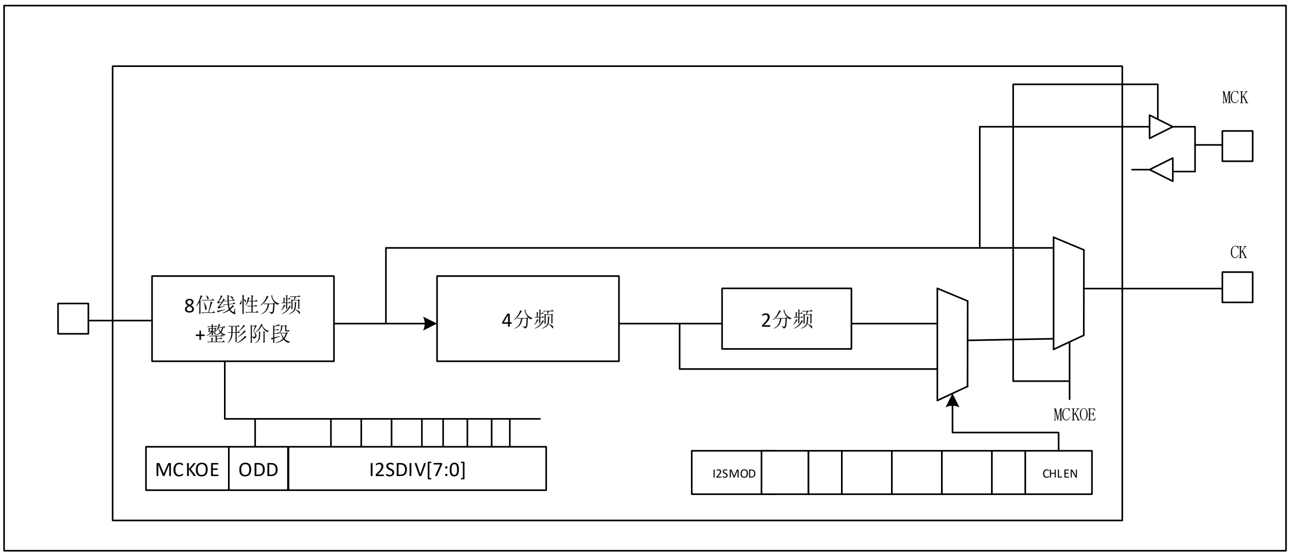


图 29-34 I2S 时钟发生器结构

音频的采样频率可以是 96kHz、 48kHz、 44.1kHz、 32kHz、 22.05kHz、 16kHz、 11.025kHz 或者 8kHz(或任何此范围内的数值)。为了获得需要的频率，需按照以下公式设置线性分频器而获得，当需要生成主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为‘1’)：

声道的帧长为 16 位时，Fs=I2SxCLK/[(16*2)*((2*I2SDIV)+ODD)*8]

声道的帧长为 32 位时，Fs=I2SxCLK/[(32*2)*((2*I2SDIV)+ODD)*4]

当关闭主时钟时(MCKOE 位为‘0’)：

声道的帧长为 16 位时，Fs=I2SxCLK/[(16*2)*((2*I2SDIV)+ODD)]

声道的帧长为 32 位时，Fs=I2SxCLK/[(32*2)*((2*I2SDIV)+ODD)]

使用标准的 8MHz HSE 时钟得到精确的音频频率，见下表：

表 29-35 音频频率

SYS-CLK(MHZ)	I2S_DIV		I2S_ODD		MCLK	期望 Fs(Hz)	实际 Fs(Hz)		误差	
	16 位	32 位	16 位	32 位			16 位	32 位	16 位	32 位
72	11	6	1	0	无	96000	97826.09	93750	1.90%	2.34%
72	23	11	1	1	无	48000	47872.34	48913.04	0.27%	1.90%
72	25	13	1	0	无	44100	44117.65	43269.23	0.04%	1.88%
72	35	17	0	1	无	32000	32142.86	32142.86	0.45%	0.45%
72	51	25	0	1	无	22050	22058.82	22058.82	0.04%	0.04%
72	70	35	1	0	无	16000	15957.45	16071.43	0.27%	0.45%
72	102	51	0	0	无	11025	11029.41	11029.41	0.04%	0.04%
72	140	70	1	1	无	8000	8007.117	7978.723	0.09%	0.27%
72	2	2	0	0	有	96000	70312.5	70312.5	26.76%	26.76%
72	3	3	0	0	有	48000	46875	46875	2.34%	2.34%
72	3	3	0	0	有	44100	46875	46875	6.29%	6.29%
72	4	4	1	1	有	32000	31250	31250	2.34%	2.34%
72	6	6	1	1	有	22050	21634.62	21634.62	1.88%	1.88%

SYS-CLK(MHZ)	I2S_DIV		I2S_ODD		MCLK	期望 Fs(Hz)	实际 Fs(Hz)		误差	
	16 位	32 位	16 位	32 位			16 位	32 位	16 位	32 位
72	9	9	0	0	有	16000	15625	15625	2.34%	2.34%
72	13	13	0	0	有	11025	10817.31	10817.31	1.88%	1.88%
72	17	17	1	1	有	8000	8035.714	8035.714	0.45%	0.45%

29.4.4. I2S 主模式

设置 I2S 工作在主模式，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 SPI_I2SPR 的 MCKOE 位来选择输出或者不输出主时钟(MCK)。

1. 设置寄存器 SPI_I2SPR 的 I2SDIV[7:0]定义与音频采样频率相符的串行时钟波特率。同时也要定义寄存器 SPI_I2SPR 的 ODD 位。

2. 设置 CKPOL 位定义通信用时钟在空闲时的电平状态。如果需要向外部的 ADC 音频器件提供主时钟 MCK，将寄存器 SPI_I2SPR 的 MCKOE 位置为‘1’，并按照不同的 MCK 输出状态，计算 I2SDIV 和 ODD 的值。

3. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位为‘1’激活 I2S 功能，设置 I2SSTD[1:0]和 PCMSYNC 位选择所用的 I2S 标准，设置 CHLEN 选择每个声道的数据位数。还要设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I2S 主模式和方向。

4. 如果需要，可以通过设置寄存器 SPI_CR2 来打开所需的中断功能和 DMA 功能。

5. 必须将寄存器 SPI_I2SCFGR 的 I2SE 位置为‘1’。

6. 引脚 WS 和 CK 需要配置为输出模式。如果寄存器 SPI_I2SPR 的 MCKOE 位为‘1’，引脚 MCK 也要配置成输出模式。

1）发送流程

当写入 1 个半字(16 位)的数据至发送缓存，发送流程开始。

假设第一个写入发送缓存的数据对应的是左声道数据。当数据从发送缓存移到移位寄存器时，标志位 TXE 置‘1’，这时，要把对应右声道的数据写入发送缓存。标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。标志位 CHSIDE 的值在 TXE 为‘1’时更新，因此它在 TXE 为‘1’时有意义。在先左声道后右声道的数据都传输完成后，才能被认为是一个完整的数据帧。不可以只传输部分数据帧，如仅有左声道的数据。

当发出第一位数据的同时，半字数据被并行地传送至 16 位移位寄存器，然后后面的位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓存移至移位寄存器时，标志位 TXE 置为‘1’，如果寄存器 SPI_CR2 的 TXEIE 位为‘1’，则产生中断。

写入数据的操作取决于所选择的 I2S 标准。为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。

建议在要关闭 I2S 功能时，等待标志位 TXE=1 及 BSY=0，再将 I2SE 位清‘0’。

2）接收流程

接收流程的配置步骤除了第 3 点外，与发送流程的一致(参见前述的”发送流程”)，需要通过配置 I2SCFG[1:0]来选择主接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收。即每次填满接收缓存后，标志位 RXNE 置‘1’，如果寄存器 SPI_CR2 的 RXNEIE 位为‘1’，则产生中断。根据配置的数据和声道长度，收到左声道或右声道的数据会需要 1 次或者 2 次把数据传送到接收缓存的过程。对寄存器 SPI_DR 进行读操作即可清除 RXNE 标志位。每次接收以后即更新 CHSIDE。它的值取决于 I2S 单元产生的 WS 信号。读取数据的操作取决于所选择的 I2S 标准。

如果前一个接收到的数据还没有被读取，又接收到新数据，即发生上溢，标志位 OVR 被置为‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，表示发生了错误。

若要关闭 I2S 功能，需要执行特别的操作，以保证 I2S 模块可以正常地完成传输周期而不会开始新的数据传输。操作过程与数据配置和通道长度、以及音频协议的模式相关：

- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 LSB(低位)对齐模式(I2SSTD=10)
  a) 等待倒数第二个(n-1)RXNE=1；
  b) 等待 17 个 I2S 时钟周期(使用软件延迟)；
  c) 关闭 I2S(I2SE=0)。
- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 MSB(高位)对齐、 I2S 或 PCM 模式(分别为 I2SSTD=00， I2SSTD=01 或 I2SSTD=11)
  a) 等待最后一个 RXNE=1；
  b) 等待 1 个 I2S 时钟周期(使用软件延迟)；
  c) 关闭 I2S(I2SE=0)。
- 所有其它 DATLEN 和 CHLEN 的组合， I2SSTD 选择的任意音频模式，使用下述方式关闭 I2S：
  a) 等待倒数第二个(n-1)RXNE=1；
  b) 等待一个 I2S 时钟周期(使用软件延迟)；
  c) 关闭 I2S(I2SE=0)。

注：在传输期间 BSY 标志始终为低。

29.4.5. I2S 从模式

在从模式下，I2S 可以设置成发送和接收模式。从模式的配置方式基本遵循和配置主模式一样的流程。在从模式下，不需要 I2S 接口提供时钟。时钟信号和 WS 信号都由外部主 I2S 设备提供，连接到相应的引脚上。因此用户无需配置时钟。

1. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位激活 I2S 功能；设置 I2SSTD[1:0]来选择所用的 I2S 标准；设置 DATLEN[1:0]选择数据的比特数；设置 CHLEN 选择每个声道的数据位数。设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I2S 从模式的数据方向。
2. 根据需要，设置寄存器 SPI_CR2 打开所需的中断功能和 DMA 功能。
3. 必须设置寄存器 SPI_I2SCFGR 的 I2SE 位为‘1’。

1）发送流程

当外部主设备发送时钟信号，并且当 NSS_WS 信号请求传输数据时，发送流程开始。必须先使能从设备，并且写入 I2S 数据寄存器之后，外部主设备才能开始通信。

对于 I2S 的 MSB 对齐和 LSB 对齐模式，第一个写入数据寄存器的数据项对应左声道的数据。当开始通信时，数据从发送缓冲器传送到移位寄存器，然后标志位 TXE 置为‘1’；这时，要把对应右声道的数据项写入 I2S 数据寄存器。

标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。与主模式的发送流程相比，在从模式中，CHSIDE 取决于来自外部主 I2S 的 WS 信号。这意味着从 I2S 在接收到主机生成的时钟信号之前，就要准备好第一个要发送的数据。 WS 信号为‘1’表示先发送左声道。

注：设置 I2SE 位为‘1’的时间，应当比 CK 引脚上的主 I2S 时钟信号早至少 2 个 PCLK 时钟周期。

当发出第一位数据的时候，半字数据并行地通过 I2S 内部总线传输至 16 位移位寄存器，然后其它位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓冲器传送至移位寄存器时，标志位 TXE 置‘1’，如果寄存器 SPI_CR2 的 TXEIE 位为‘1’，则产生中断。注意，在对发送缓冲器写入数据前，要确认标志位 TXE 为‘1’。写入数据的操作取决于所选中的 I2S 标准。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。如果在代表下一个数据传输的第一个时钟边沿到达之前，新的数据仍然没有写入寄存器 SPI_DR，下溢标志位会置‘1’，并可能产生中断；它指示软件发送数据错误。如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，在寄存器 SPI_SR 的标志位 UDR 为高时，就会产生中断。建议在这时关闭 I2S，然后重新从左声道开始发送数据。

建议在清除 I2SE 位关闭 I2S 之前，先等待 TXE=1 并且 BSY=0。

2）接收流程

配置步骤除了第 1 点外，与发送流程一致。需要通过配置 I2SCFG[1:0]来选择主接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收，即每次填满接收缓存，标志位 RXNE 置‘1’，如果寄存器 SPI_CR2 的 RXNEIE 位为‘1’，则产生中断。按照不同的数据和声道长度设置，收到左声道或者右声道数据会需要 1 次或者 2 次传输数据至接收缓冲器的过程。每次接收到数据(将要从 SPI_DR 读出)以后即更新 CHSIDE，它对应 I2S 单元产生的 WS 信号。读取 SPI_DR 寄存器，将清除 RXNE 位。读取数据的操作取决于所选中的 I2S 标准，详见 5.2 节。

在还没有读出前一个接收到的数据，又接收到新数据时，即产生上溢，并设置标志位 OVR 为‘1’；如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，指示发生了错误。

要关闭 I2S 功能时，需要在接收到最后一次 RXNE=1 时将 I2SE 位清‘0’。

注意：外部主 I2S 器件需要有通过音频声道发送/接收 16 位或 32 位数据包的功能。

29.4.6. 状态标志位

有 3 个状态标志位供用户监控 I2S 总线的状态。

29.4.6.1. 忙标志位(BSY)

BSY 标志由硬件设置与清除(写入此位无效果)，该标志位指示 I2S 通信层的状态。该位为‘1’时表明 I2S 通讯正在进行中，但有一个例外：主接收模式(I2SCFG=11)下，在接收期间 BSY 标志始终为低。在软件要关闭 SPI 模块之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输，因此需要严格按照下述过程执行。当传输开始时， BSY 标志被置为‘1’，除非 I2S 模块处于主接收模式。

下述情况时，该标志位被清除：

- ✓ 当传输结束时(除了主发送模式，这种模式下通信是连续的)；
- ✓ 当关闭 I2S 模块时。

当通信是连续的时候：

- ✓ 在主发送模式时，整个传输期间， BSY 标志始终为高；
- ✓ 在从模式时，每个数据项传输之间， BSY 标志在 1 个 I2S 时钟周期内变低。

29.4.6.2. 发送缓存空标志位(TXE)

该标志位为‘1’表示发送缓冲器为空，可以对发送缓冲器写入新的待发送数据。在发送缓冲器中已有数据时，标志位清‘0’。在 I2S 被关闭时(I2SE 位为‘0’)，该标志位也为‘0’。

29.4.6.3. 接收缓存非空标志位(RXNE)

该标志位置‘1’表示在接收缓存里有接收到的有效数据。在读取寄存器 SPI_DR 时，该位清‘0’。

29.4.6.4. 声道标志位(CHSIDE)

在发送模式下，该标志位在 TXE 为高时刷新，指示从 SD 引脚上发送的数据所在的声道。如果在从发送模式下发生了下溢错误，该标志位的值无效，在重新开始通讯前需要把 I2S 关闭再打开。在接收模式下，该标志位在寄存器 SPI_DR 接收到数据时刷新，指示接收到的数据所在的声道。如果发生错误(如上溢 OVR)，该标志位无意义，需要将 I2S 关闭再打开(同时，如果必要修改 I2S 的配置)。

在 PCM 标准下，无论短帧格式还是长帧格式，这个标志位都没有意义。

如果寄存器 SPI_SR 的标志位 OVR 或 UDR 为‘1’，且寄存器 SPI_CR2 的 ERRIE 位为‘1’，则会产生中断，而后可以通过读寄存器 SPI_SR 来清除中断标志。

29.4.7. 错误标志位

I2S 单元有 2 个错误标志位。

29.4.7.1. 下溢标志位(UDR)

在从发送模式下，如果数据传输的第一个时钟边沿到达时，新的数据仍然没有写入 SPI_DR 寄存器，该标志位会被置‘1’。在寄存器 SPI_I2SCFGR 的 I2SMOD 位置‘1’后，该标志位才有效。如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，就会产生中断。通过对寄存器 SPI_SR 进行读操作来清除该标志位。

29.4.7.2. 上溢标志位(OVR)

如果还没有读出前一个接收到的数据时，又接收到新的数据，即产生上溢，该标志位置‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断指示发生了错误。这时，接收缓存的内容，不会刷新为从发送设备送来的新数据。对寄存器 SPI_DR 的读操作返回最后一个正确接收到的数据。其他所有在上溢发生后由发送设备发出的 16 位数据都会丢失。通过先读寄存器 SPI_SR 再读寄存器 SPI_DR，来清除该标志位。

29.4.8. I2S 中断

表 29-36 I2S 中断请求

中断事件	事件标志位	使能标志位
发送缓冲器空标志位	TXE	TXEIE
接收缓冲器非空标志位	RXNE	RXNEIE

中断事件	事件标志位	使能标志位
下溢标志位	OVR	ERRIE
上溢标志位	UDR	ERRIE

29.4.9. DMA 功能

除了 CRC 功能外，同 SPI。因为在 I2S 模式下没有数据传输保护系统。

29.5. SPI 和 I2S 寄存器

SPI 对应的寄存器可以进行 16-bit 访问，。

29.5.1. SPI 控制寄存器 1 (SPI_CR1)(I2S 模式下不使用)

Address offset:0x00

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BI-DIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW	~~RW~~	~~RW~~	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15	BIDIMODE	RW	0	双向数据模式使能 0：选择“双线双向”模式； 1：选择“单线双向”模式。 注： I2S 模式下不使用。
14	BIDIOE	RW	0	双向模式下的输出使能 和 BIDIMODE 位一起决定在“单线双向”模式下数据的输出方向 0：输出禁止(只收模式)； 1：输出使能(只发模式)。 这个“单线”数据线在主设备端为 MOSI 引脚，在从设备端为 MISO 引脚。 注： I2S 模式下不使用。
13	CRCEN	RW	0	硬件 CRC 校验使能 0：禁止 CRC 计算； 1：启动 CRC 计算。 注：只有在禁止 SPI 时(SPE=0)，才能写该位，否则出错。 该位只能在全双工模式下使用。 注： I2S 模式下不使用。
12	CRCNEXT	RW	0	下一个发送 CRC 0：下一个发送的值来自发送缓冲区。 1：下一个发送的值来自发送 CRC 寄存器。

Bit	Name	R/W	Reset Value	Function
				注：在 SPI_DR 寄存器写入最后一个数据后应马上设置该位。 注： I2S 模式下不使用。
11	DFF	RW	0	数据帧格式 0：使用 8 位数据帧格式进行发送/接收； 1：使用 16 位数据帧格式进行发送/接收。 注：只有当 SPI 禁止(SPE=0)时，才能写该位，否则出错。 注：I2S 模式下不使用。
10	RXONLY	RW	0	只接收 该位和 BIDIMODE 位一起决定在“双线单向”模式下的传输方向。在多个从设备的配置中，在未被访问的从设备上该位置 1，使得只有被访问的从设备才有输出，因而不会造成数据线上有数据冲突。 0：全双工（发送和接收） 1：禁止输出（只接收模式） 注： I2S 模式下不使用。
9	SSM	RW	0	软件从设备管理 当 SSM 置位，NSS 引脚上的电平由 SSI 位的值决定。 0：禁止软件从设备管理 1：使能软件从设备管理 注： I2S 模式下不使用。
8	SSI	RW	0	内部从设备选择 (Internal slave select) 该寄存器只有当 SSM=1 时才有效。该寄存器决定了 NSS 上的电平，在 NSS 引脚上的 I/O 操作无效。 注： I2S 模式下不使用。
7	LSBFIRST	RW	0	帧格式 0：先发送 MSB 1：先发送 LSB 通讯进行时不能改变该寄存器的值。 注： I2S 模式下不使用。
6	SPE	RW	0	SPI 使能 0：禁止 SPI 1：使能 SPI 注： I2S 模式下不使用。
5:3	BR[2:0]	RW	0	波特率控制 000：$f_{PCLK}/2$ 001：$f_{PCLK}/4$ 010：$f_{PCLK}/8$ 011：$f_{PCLK}/16$ 100：$f_{PCLK}/32$ 101：$f_{PCLK}/64$ 110：$f_{PCLK}/128$

Bit	Name	R/W	Reset Value	Function
				111：$f_{PCLK}/256$ 通讯进行时不能改变该寄存器的值。 注： I2S 模式下不使用。
2	MSTR	RW	0	主设备选择 0：配置为从设备 1：配置为主设备 通讯进行时不能改变该寄存器的值。 注： I2S 模式下不使用。
1	CPOL	RW	0	时钟极性 0：空闲状态时，SCK 保持低电平 1：空闲状态时，SCK 保持高电平 通讯进行时不能改变该寄存器的值。 注： I2S 模式下不使用。
0	CPHA	RW	0	时钟相位 0：数据采样从第一个时钟边沿开始 1：数据采样从第二个时钟边沿开始 通讯进行时不能改变该寄存器的值。 注： I2S 模式下不使用。

29.5.2. SPI 控制寄存器 2 (SPI_CR2)

Address offset:0x04

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res	SSOE	TXDMAEN	RXDMAEN
								RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
15:8	Reserved	-	-	Reserved
7	TXEIE	RW	0	发送缓冲区空中断使能 0：禁止 TXE 中断 1：使能 TXE 中断。TXE=1 时产生中断请求。
6	RXNEIE	RW	0	接收缓冲区非空中断使能 0：禁止 RXNE 中断 1：使能 RXNE 中断。RXNE=1 时产生中断请求。
5	ERRIE	RW	0	错误中断使能 当错误(CRCERR、 OVR、 MODF)产生时，该位控制是否产生中断 0：禁止错误中断 1：使能错误中断。
4	CLRTXFIFO	RW	0	清空 TXFIFO 软件置位，硬件复位 0：没作用 1：清空 TXFIFO

Bit	Name	R/W	Reset Value	Function
				注：只有当 SPI 禁止(SPE=0)时，才能写该位， 否则无效。
3	Reserved	-	-	Reserved
2	SSOE	RW	0	SS 输出使能 0：禁止在主模式下 SS 输出，该设备可以工作在多主设备模式 1：开启主模式下 SS 输出，该设备不能工作在多主设备模式。
1	TXDMAEN	RW	0	发送缓冲区 DMA 使能 当该位被设置时， TXE 标志一旦被置位就发出 DMA 请求 0：禁止发送缓冲区 DMA 1：使能发送缓冲区 DMA。
0	RXDMAEN	RW	0	接收缓冲区 DMA 使能 当该位被设置时， RXNE 标志一旦被置位就发出 DMA 请求 0：禁止接收缓冲区 DMA 1：使能接收缓冲区 DMA。

29.5.3. SPI 状态寄存器 (SPI_SR)

Address offset:0x08

Reset value:0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			FTLVL [1:0]		FRLVL [1:0]		Res	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
			R	R	R	R		R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	Reserved
12:11	FTLVL	R	0	FIFO 发送 level。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO full(当 FIFO 阈值大于 1/2，即认为是满) 注意：该位在 I2S 模式下不使用。
10:9	FRLVL	R	0	FIFO 接收 level。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满 注意：这些位在 I²S 模式和 带 CRC 校验的 SPI 仅接收模式下不使用。
8	Reserved	-	-	Reserved
7	BSY	R	0	忙标志。 0：SPI 不忙； 1：SPI 处于通讯，或者发送缓冲非空。

Bit	Name	R/W	Reset Value	Function
				该位由硬件置位或者复位。
6	OVR	R	0	溢出标志 0：没有出现溢出错误； 1：出现溢出错误。 该位由硬件置位，由软件序列复位。（上溢和下溢序列不同）。
5	MODF	R	0	模式错误 0：没有出现模式错误； 1：出现模式错误。 该位由硬件置位，由软件序列复位。 注： I2S 模式下不使用。
4	CRCERR	RC_W0	0	CRC 错误标志 (CRC error flag) 0：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值匹配； 1：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值不匹配。 该位由硬件置位，由软件写 ‘0’ 而复位。 注： I2S 模式下不使用。
3	UDR	R	0	下溢标志位 (Underrun flag) 0：未发生下溢； 1：发生下溢。 该标志位由硬件置 ‘1’ ，由一个软件序列清 ‘0’ 。 注：在 SPI 模式下不使用。
2	CHSIDE	R	0	声道 0：需要传输或者接收左声道； 1：需要传输或者接收右声道。 注：在 SPI 模式下不使用。在 PCM 模式下无意义。
1	TXE	R	1	发送缓冲空。 0：发送缓冲非空 1：发送缓冲为空
0	RXNE	R	0	接收缓冲非空。 0：接收缓冲非空 1：接收缓冲为空

29.5.4. SPI 数据寄存器 (SPI_DR)

Address offset:0x0C

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	DR[15:0]	RW	0	数据寄存器。 要发送或者接收到的数据。

Bit	Name	R/W	Reset Value	Function
				数据寄存器作为 RxFIFO 和 TxFIFO 的接口。当要读数据，实际访问 RxFIFO，而要写数据，实际访问 TxFIFO。 注意：数据始终是右对齐的。 写入寄存器时忽略未使用的位，读取寄存器时读取为零。 Rx 阈值设置必须始终与当前使用的读取访问相对应。

29.5.5. SPI CRC 多项式寄存器 (SPI_CRCPR)

Address offset:0x10

Reset value:0x0007

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCPOLY[15:0]															
RW															

Bit	Name	R/W	Reset Value	Function
15:0	CRCPOLY[15:0]	RW	0x7	CRC 多项式寄存器 (CRC polynomial register) 该寄存器包含了 CRC 计算时用到的多项式。 其复位值为 0x0007，根据应用可以设置其他数值。 注：在 I2S 模式下不使用。 注：多项式值只能是奇数，不支持偶数值。

29.5.6. SPI Rx CRC 寄存器 (SPI_RXCRCR)

Address offset:0x14

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
15:0	RxCRC[15:0]	R	0	接收 CRC 寄存器 在启用 CRC 计算时， RXCRC[15:0]中包含了依据收到的字节计算的 CRC 数值。当在 SPI_CR1 的 CRCEN 位写入 ‘1’ 时，该寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的标准。 注：当 BSY 标志为 ‘1’ 时读该寄存器，将可能读到不正确的数值。 注：在 I2S 模式下不使用。

29.5.7. SPI Tx CRC 寄存器 (SPI_TXCRCR)

Address offset:0x18

Reset value:0x0000

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

Bit	Name	R/W	Reset Value	Function
15:0	TxCRC[15:0]	R	0	发送 CRC 寄存器 在启用 CRC 计算时， TXCRC[15:0]中包含了依据将要发送的字节计算的 CRC 数值。当在 SPI_CR1 中的 CRCEN 位写入‘1’时，该寄存器被复位。 CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 个位都参与计算，并且按照 CRC16 的标准。 注：当 BSY 标志为‘1’时读该寄存器，将可能读到不正确的数值。 注：在 I2S 模式下不使用。

29.5.8. SPI_I2S 配置寄存器 (SPI_I2S_CFGR)

Address offset:0x1C

Reset value:0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				I2SMOD	I2SE	I2SCFG		PCMSYNC	Res	I2SSTD		CKPOL	DATLEN		CHLEN
				RW	RW	RW		RW		RW		RW	RW		RW

Bit	Name	R/W	Reset Value	Function
15:12	Reserved	-	-	Reserved
11	I2SMOD	RW	0	I2S 模式选择 0: 选择 SPI 模式; 1: 选择 I2S 模式。 注：该位只有在关闭了 SPI 或者 I2S 时才能设置。
10	I2SE	RW	0	I2S 使能 0:关闭 I2S; 1: I2S 使能。 注：在 SPI 模式下不使用。
9:8	I2SCFG	RW	0	I2S 模式设置 00: 从设备发送; 01: 从设备接收; 10: 主设备发送; 11: 主设备接收。 注：该位只有在关闭了 I2S 时才能设置。 在 SPI 模式下不使用。
7	PCMSYNC	RW	0	PCM 帧同步 0: 短帧同步; 1: 长帧同步。

Bit	Name	R/W	Reset Value	Function
				注: 该位只在 I2SSTD = 11 (使用 PCM 标准)时有意义。 在 SPI 模式下不使用。
6	Reserved	-	-	Reserved
5:4	I2SSTD	RW	0	I2S 标准选择 00: I2S 飞利浦标准; 01: 高字节对齐标准 (左对齐); 10: 低字节对齐标准(右对齐); 11: PCM 标准。 注：为了正确操作，只有在关闭了 I2S 时才能设置该位。 在 SPI 模式下不使用。
3	CKPOL	RW	0	静止态时钟极性 0: I2S 时钟静止态为低电平; 1: I2S 时钟静止态为高电平。 注：为了正确操作，该位只有在关闭了 I2S 时才能设置。 在 SPI 模式下不使用。
2:1	DATLEN	RW	0	待传输数据长度 00: 16 位数据长度; 01: 24 位数据长度; 10: 32 位数据长度; 11: 不允许。 注：为了正确操作，该位只有在关闭了 I2S 时才能设置。 在 SPI 模式下不使用。
0	CHLEN	RW	0	声道长度 (每个音频声道的数据位数) 0: 16 位宽; 1: 32 位宽。 只有在 DATLEN = 00 时该位的写操作才有意义，否则声道长度都由硬件固定为 32 位。 注：为了正确操作，该位只有在关闭了 I2S 时才能设置。 在 SPI 模式下不使用。

29.5.9. SPI_I2S 预分频寄存器 (SPI_I2SPR)

Address offset:0x20

Reset value:0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						MCKOE	ODD	I2SDIV							
						RW	RW	RW							

Bit	Name	R/W	Reset Value	Function
15:10	Reserved	-	-	Reserved
9	MCKOE	RW	0	主设备时钟输出使能 0: 关闭主设备时钟输出; 1: 主设备时钟输出使能。

Bit	Name	R/W	Reset Value	Function
				注：为了正确操作，该位只有在关闭了 I2S 时才能设置。仅在 I2S 主设备模式下使用该位。 在 SPI 模式下不使用。
8	ODD	RW	0	奇系数预分频 0: 实际分频系数 = I2SDIV *2； 1: 实际分频系数 = (I2SDIV * 2)+1。 注：为了正确操作，该位只有在关闭了 I2S 时才能设置。仅在 I2S 主设备模式下使用该位。 在 SPI 模式下不使用。
7:0	I2SDIV	RW	0	I2S 线性预分频 禁止设置 I2SDIV [7:0] = 0 或者 I2SDIV [7:0] = 1 注：为了正确操作，该位只有在关闭了 I2S 时才能设置。仅在 I2S 主设备模式下使用该位。 在 SPI 模式下不使用。

29.5.10. SPI 寄存器映像

Off-set	Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SPI_CR1	BI-DIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPI_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res.	SSOE	TXDMAEN	RXDMAEN
	Reset value									0	0	0	0		0	0	0
0x08	SPI_SR	Res.	Res.	Res.	FTLVL[1:0]		FRLVL[1:0]		Res.	BSY	OVR	MODEF	CRCERR.	UDR.	CHSIDE.	TXE	RXNE
	Reset value				0	0	0	0		0	0	0	0	0	0	1	0
0x0C	SPI_DR	DR[15:0]															
	Reset value	0															
0x10	SPI_CRCPR	CRCPOLY[15:0]															
	Reset value	0															
0x14	SPI_RxCRC	RxCRC[15:0]															
	Reset value	0															
0x18	SPI_TxCRC	TxCRC[15:0]															
	Reset value	0															
0x1C	SPI_CFGR	Res.	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG		PCM-SYNC	Res.	I2SSTD		CKPOL	DATLEN		CHLEN
	Reset value					0	0	0		0		0		0	0		0
0x20	SPI_I2SPR	Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV							
	Reset value							0	0	0							

30.通用同步异步收发器 (USART)

30.1. 介绍

通用同步异步收发器(USART)提供了一种灵活的方法与使用工业标准NRZ异步串行数据格式的外部设备之间进行全双工数据交换。USART利用分数波特率发生器提供宽范围的波特率选择。

它支持同步单向通信和半双工单线通信也支持LIN(局部互联网)，智能卡协议和IrDA(红外数据组织)ENDEC规范，以及调制解调器（CTS/RTS）操作。它还允许多处理器通信。

使用多缓冲器配置的 DMA 方式，可以实现高速数据通信。

30.2. USART 主要特性

- 2 个支持全功能的 USART(USART1 和 USART2)，两个不支持 lin，scen，irda 的 USART(USART3 和 USART4)
- 全双工异步通信
- 发送和接收共用的可编程波特率，最高达 4.5Mbit/s
- 可配置的数据字长度（8 位或者 9 位）
- 可配置的停止位— 支持 0.5，1，1.5 或 2 个停止位
- 发送方为同步传输提供时钟
- 单线半双工通信
- 单独的发送器和接收器使能位
- 奇偶校验控制
  - 发送校验位
  - 对接收数据进行校验
- 检测标志
  - 接收缓冲器满
  - 发送缓冲器空
  - 传输结束标志
- Lin 主发送同步断开符的能力以及 Lin 从检测断开符的能力
  - 当 USART 硬件配置成 LIN 时，生成 13 位断开符，检测 10/11 断开符
- IRDA SIR 编码器解码器
  - 在正常模式下支持 3/16 位的持续时间
- 智能卡模拟功能
  - 智能卡接口支持 ISO7816-3 标准里定义的异步智能卡协议
  - 智能卡用到的 0.5 和 1.5 个停止位
- 四个错误检测标志
  - 溢出错误
  - 噪音错误
  - 帧错误
  - 校验错误
- 10 个带标志的中断源

- ➢ CTS 改变
- ➢ LIN 断开符检测
- ➢ 发送数据寄存器为空
- ➢ 发送完成
- ➢ 接收数据寄存器满
- ➢ 检测到总线为空闲
- ➢ 溢出错误
- ➢ 帧错误
- ➢ 噪音错误
- ➢ 校验错误

- ■ 多处理器通信。如果地址不匹配，则进入静默模式
- ■ 从静默模式中唤醒（通过空闲总线检测或地址标志检测）

两种唤醒接收器的方式：地址位（MSB,第 9 位），总线空闲

30.3. USART 功能描述

USART 接口通过三个引脚与其他设备连接在一起。任何 USART 双向通信至少需要两个脚：接收数据输入(RX)和发送数据输出(TX)。

RX: 接收数据串行输。通过过采样技术来区别数据和噪音，从而恢复数据。

TX: 发送数据输出。当发送器被禁止时，输出引脚恢复到它的 I/O 端口配置。当发送器被激活，并且不发送数据时，TX 引脚处于高电平。在单线和智能卡模式里，此 I/O 口被同时用于数据的发送和接收。

1) 总线在发送或接收前应处于空闲状态
2) 一个起始位
3) 一个数据字(8 或 9 位)，最低有效位在前
4) 0.5， 1.5， 2 个的停止位，由此表明数据帧的结束
5) 使用分数波特率发生器：12 位整数和 4 位小数的表示方法
6) 一个状态寄存器(USART_SR)
7) 数据寄存器(USART_DR)
8) 一个波特率寄存器(USART_BRR)，12 位的整数和 4 位小数
9) 一个智能卡模式下的保护时间寄存器(USART_GTPR)

在同步模式中需要下列引脚：

CK：发送器时钟输出。

此引脚输出用于同步传输的时钟，(在 Start 位和 Stop 位上没有时钟脉冲，软件可选地，可以在最后一个数据位送出一个时钟脉冲)。数据可以在 RX 上同步被接收。这可以用来控制带有移位寄存器的外部设备(例如 LCD 驱动器)。时钟相位和极性都是软件可编程的。

下列引脚在硬件流控模式中需要：

- ■ **nCTS:** 清除发送，若是高电平，在当前数据传输结束时阻断下一次的数据发送。
- ■ **nRTS:** 发送请求，若是低电平，表明 USART 准备好接收数据。

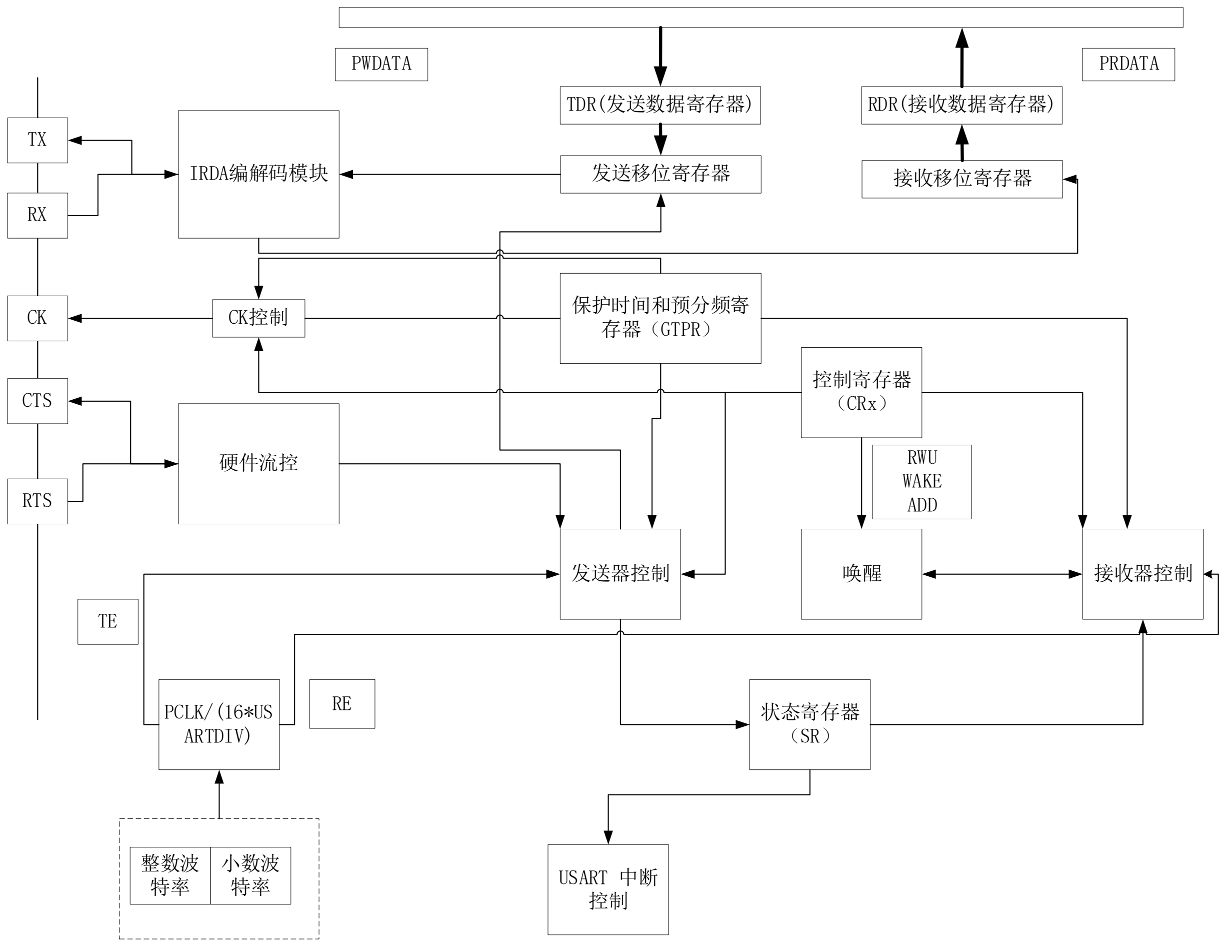


图 30-1 USART 框图

30.3.1. USART 特征描述

字长可以通过编程 USART_CR1 寄存器中的 M 位，选择成 8 或 9 位。在起始位期间，TX 脚处于低电平，在停止位期间处于高电平。

空闲符号被视为完全由’1’组成的一个完整的数据帧，后面跟着包含了数据的下一帧的开始位(‘1’的位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，也是’0’)。在断开符号结束时，发送器再插入 1 或 2 个停止位(‘1’)来应答起始位。

发送和接收由一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位时，分别为其产生时钟。

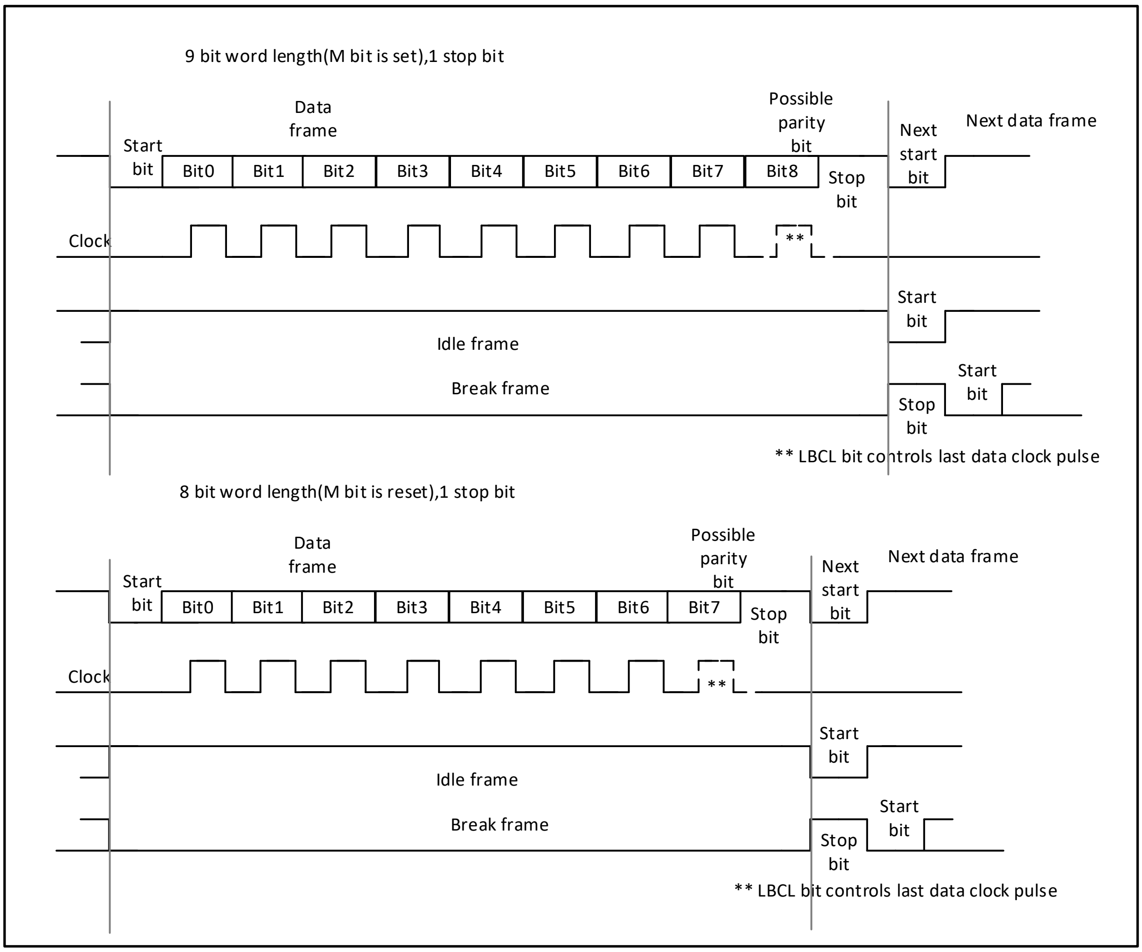


图 30-2 字长设置

30.3.2. 发送器

发送器根据 M 位的状态发送 8 位或 9 位的数据字。当发送使能位(TE)被设置时，发送移位寄存器中的数据在 TX 脚上输出，相应的时钟脉冲在 CK 脚上输出。

30.3.2.1. 字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式 USART_DR 寄存器包含了一个内部总线和发送移位寄存器之间的缓冲器。

每个字符之前都有一个低电平的起始位；之后跟着的停止位，其数目可配置。USART 支持多种停止位的配置： 0.5、 1、 1.5 和 2 个停止位。

注：

在数据传输期间不能复位 *TE* 位，否则将破坏 *TX* 脚上的数据，因为波特率计数器停止计数。正在传输的当前数据将丢失。

TE 位被激活后将发送一个空闲帧。

30.3.2.2. 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 2 的位 13、12 进行编程。

1) 1 个停止位：停止位位数的默认值。

2) 2 个停止位：可用于常规 USART 模式、单线模式以及调制解调器模式。

空闲帧包括了停止位。

断开帧是 10 位低电平，后跟停止位(当 m=0 时)；或者 11 位低电平，后跟停止位(m=1 时)。不可能传输更长的断开帧(长度大于 10 或者 11 位)。

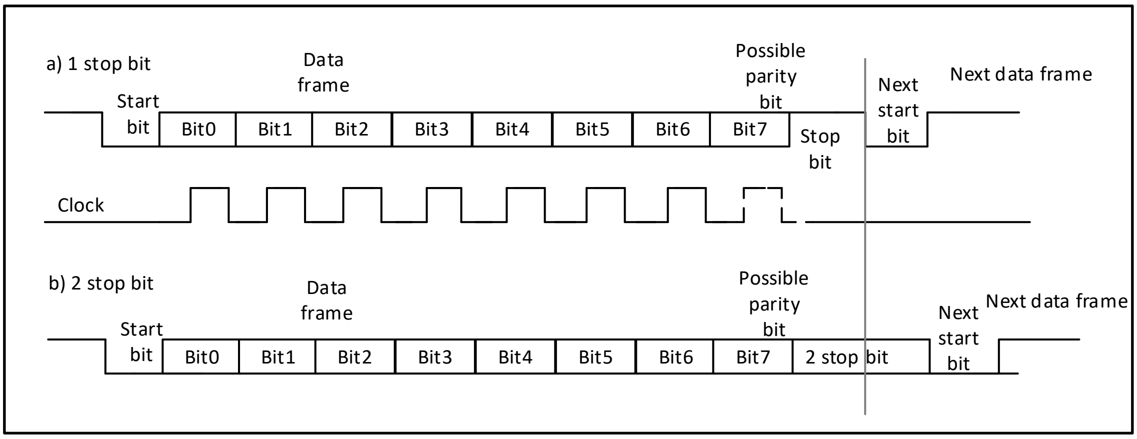


图 30-3 配置停止位

配置步骤：

1) 通过在 USART_CR1 寄存器上置位 UE 位来激活 USART

2) 编程 USART_CR1 的 M 位来定义字长。

3) 在 USART_CR2 中编程停止位的位数。

4) 如果采用多缓冲器通信，配置 USART_CR3 中的 DMA 使能位(DMAT)。按多缓冲器通信中的描述配置 DMA 寄存器。

5) 利用 USART_BRR 寄存器选择要求的波特率。

6) 设置 USART_CR1 中的 TE 位，发送一个空闲帧作为第一次数据发送。

7) 把要发送的数据写进 USART_DR 寄存器(此动作清除 TXE 位)。在只有一个缓冲器的情况下，对每个待发送的数据重复步骤 7。

8) 在 USART_DR 寄存器中写入最后一个数据字后，要等待 TC=1，它表示最后一个数据帧的传输结束。当需要关闭 USART 或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次传输。

30.3.2.3. 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
- TDR 寄存器被清空
- 下一个数据可以被写进 USART_DR 寄存器而不会覆盖先前的数据

如果 TXEIE 位被设置，此标志将产生一个中断。

如果此时 USART 正在发送数据，对 USART_DR 寄存器的写操作把数据存进 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_DR 寄存器的写操作直接把数据放进移位寄存器，数据传输开始，TXE 位立即被置起。

当一帧发送完成时(停止位发送后)并且设置了 TXE 位，TC 位被置起，如果 USART_CR1 寄存器中的 TCIE 位被置起时，则会产生中断。

在 USART_DR 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式(详见下图)之前，必须先等待 TC=1。

使用下列软件过程清除 TC 位：

1．读一次 USART_SR 寄存器；

2．写一次 USART_DR 寄存器。

注：*TC* 位也可以通过软件对它写*'0'*来清除。此清零方式只推荐在多缓冲器通信模式下使用。

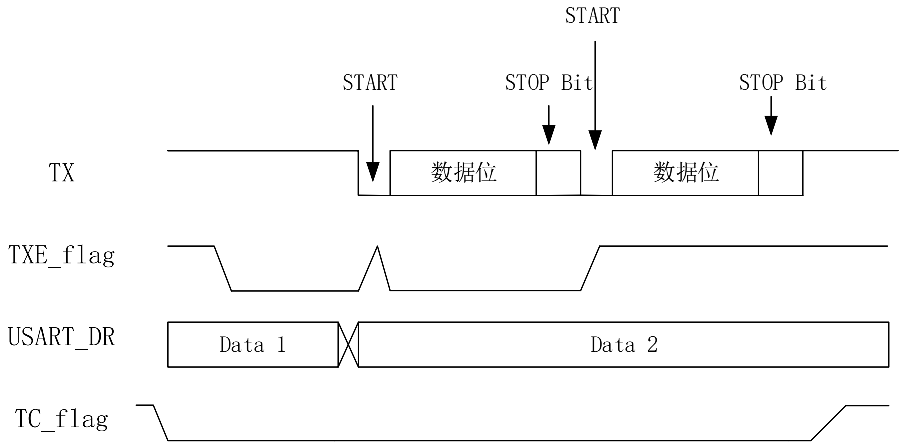


图 30-4 USART 发送时 TC/TXE 的变化

30.3.2.4. 断开符号

设置 SBK 可发送一个断开符号。断开帧长度取决 M 位。如果设置 SBK=1，在完成当前数据发送后，将在 TX 线上发送一个断开符号。断开字符发送完成时(在断开符号的停止位时)SBK 被硬件复位。USART 在最后一个断开帧的结束处插入一逻辑'1'，以保证能识别下一帧的起始位。

注意：如果在开始发送断开帧之前，软件又复位了 SBK 位，断开符号将不被发送。如果要发送两个连续的断开帧， SBK 位应该在前一个断开符号的停止位之后置起。

30.3.2.5. 空闲符号

置位 TE 将使得 USART 在第一个数据帧前发送一空闲帧。

30.3.3. 接收器

USART 可以根据 USART_CR1 的 M 位接收 8 位或 9 位的数据字。

30.3.3.1. 开始位检测

在 USART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。该序列为： 1 1 1 0 X 0 X 0 X 0 0 0 0

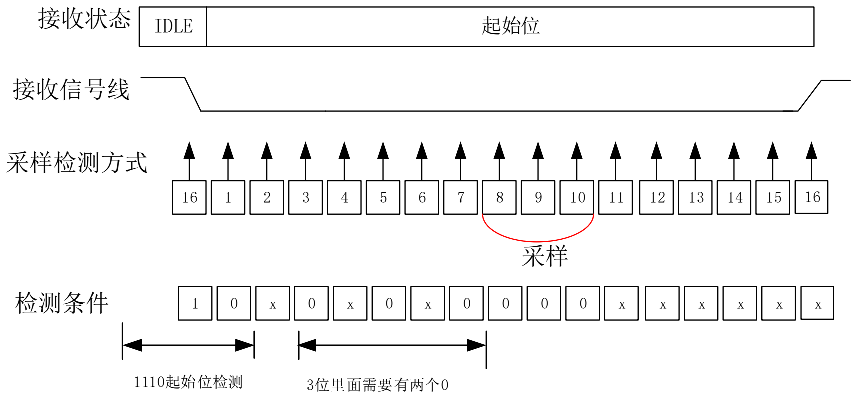


图 30-5 开始位检测

如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态*(*不设置标志位*)*等待下降沿。如果 *3* 个采样点都为*'0'(*在第 3、5、*7* 位的第一次采样，和在第 *8*、*9*、*10* 的第二次采样都为*'0')*，则确认收到起始位，这时设置 *RXNE* 标志位，如果 *RXNEIE=1*，则产生中断。

如果两次 *3* 个采样点上仅有 *2* 个是*'0'(*第 3、5、*7* 位的采样点和第 *8*、*9*、*10* 位的采样点*)*，那么起始位仍然是有效的，但是会设置 *NE* 噪声标志位。如果不能满足这个条件，则中止起始位的侦测过程，接收器会回到空闲状态*(*不设置标志位*)*。

如果有一次 *3* 个采样点上仅有 *2* 个是*'0'(*第 3、5、*7* 位的采样点或第 *8*、*9*、*10* 位的采样点*)*，那么起始位仍然是有效的，但是会设置 *NE* 噪声标志位。

30.3.3.2. 字符接收

在 USART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里， USART_DR 寄存器包含的缓冲器位于内部总线和接收移位寄存器之间。

配置步骤：

1. 将 USART_CR1 寄存器的 UE 置 1 来激活 USART。
2. 编程 USART_CR1 的 M 位定义字长
3. 在 USART_CR2 中编写停止位的个数
4. 如果需多缓冲器通信，选择 USART_CR3 中的 DMA 使能位(DMAR)。按多缓冲器通信所要求的配置 DMA 寄存器。
5. 利用波特率寄存器 USART_BRR 选择希望的波特率。
6. 设置 USART_CR1 的 RE 位。激活接收器，使它开始寻找起始位。

当一字符被接收到时：

- RXNE 位被置位。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)。
- 如果 RXNEIE 位被设置，产生中断。
- 在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起

- 在多缓冲器通信时， 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_DR 寄存器完成对 RXNE 位清除。 RXNE 标志也可以通过对它写 0 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。

  注意： 在接收数据时，*RE* 位不应该被复位。如果 *RE* 位在接收时被清零，当前字节的接收被丢失。

30.3.3.3. 断开符号

当接收到一个断开帧时，USART 像处理帧错误一样处理它。

30.3.3.4. 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 位被设置将产生一个中断。

30.3.3.5. 溢出错误

如果 RXNE 还没有被复位，又接收到一个字符，则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。 RXNE 标记是接收到每个字节后被置位的。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，溢出错误产生。

当溢出错误产生时：ORE 位被置位。

- RDR 内容将不会丢失。读 USART_DR 寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。随后接收到的数据都将丢失。
- 如果 RXNEIE 位被设置或 EIE 和 DMAR 位都被设置，中断产生。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 ORE 位

注意：当 *ORE* 位置位时，表明至少有 *1* 个数据已经丢失。有两种可能性：

- 如果 RXNE=1，上一个有效数据还在接收寄存器 RDR 上，可以被读出。
- 如果 RXNE=0，这意味着上一个有效数据已经被读走，RDR 已经没有东西可读。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，此种情况可能发生。在读序列期间(在 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据，此种情况也可能发生。

30.3.3.6. 噪音错误

使用过采样技术(同步模式除外)，通过区别有效输入数据和噪音来进行数据恢复。

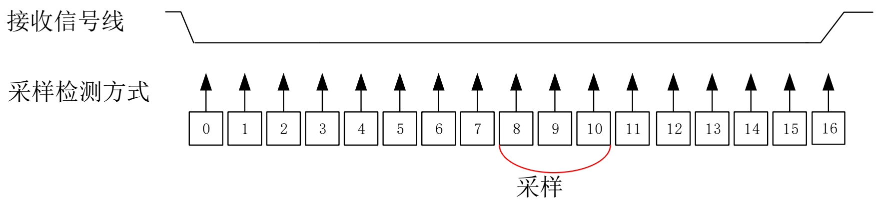


图 30-6 检测噪声的数据采样

表 30-1 检测噪声的数据采样

采样值	NE 状态	接收的位值	数据有效性
000	0	0	有效

采样值	NE 状态	接收的位值	数据有效性
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效
111	0	1	有效

当在接收帧中检测到噪音时:

- 在 RXNE 位的上升沿设置 NE 标志。
- 无效数据从移位寄存器传送到 USART_DR 寄存器。
- 在单个字节通信情况下，没有中断产生。然而，因为 NE 标志位和 RXNE 标志位是同时被设置，RXNE 将产生中断。在多缓冲器通信情况下，如果已经设置了 USART_CR3 寄存器中 EIE 位，将产生一个中断。
- 先读出 USART_SR，再读出 USART_DR 寄存器，将清除 NE 标志位

30.3.3.7. 帧错误

当以下情况发生时检测到帧错误:

由于没有同步上或大量噪音的原因，停止位没有在预期的时间上接和收识别出来。

当帧错误被检测到时:

- FE 位被硬件置起
- 无效数据从移位寄存器传送到 USART_DR 寄存器。
- 在单字节通信时，没有中断产生。然而，这个位和 RXNE 位同时置起，后者将产生中断。在多缓冲器通信情况下，如果 USART_CR3 寄存器中 EIE 位被置位的话，将产生中断。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 FE 位。

30.3.3.8. 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控制寄存器 2 的控制位来配置，在正常模式时，可以是 1 或 2 个，在智能卡模式里可能是 0.5 或 1.5 个。

- 0.5 个停止位(智能卡模式中的接收)：不对 0.5 个停止位进行采样。因此，如果选择 0.5 个停止位则不能检测帧错误和断开帧。
- 1 个停止位：对 1 个停止位的采样在第 8，第 9 和第 10 采样点上进行。
- 1.5 个停止位(智能卡模式)：当以智能卡模式发送时，器件必须检查数据是否被正确的发送出去。所以接收器功能块必须被激活(USART_CR1 寄存器中的 RE =1)，并且在停止位的发送期间采样数据线上的信号。如果出现校验错误，智能卡会在发送方采样 NACK 信号时，即总线上停止位对应的时间时，拉低数据线，以此表示出现了帧错误。 FE 在 1.5 个停止位结束时和 RXNE 一起被置起。对 1.5 个停止位的采样是在第 16，第 17 和第 18 采样点进行的。 1.5 个的停止位可以被分成 2 部分：一个是 0.5 个时钟周期，期间不做任何事情 。随后是 1 个时钟周期的停止位，在这段时间的中点处采样。

- 2 个停止位：对 2 个停止位的采样是在第一停止位的第 8，第 9 和第 10 个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被设置。第二个停止位不再检查帧错误。在第一个停止位结束时 RXNE 标志将被设置。

30.3.4. 分数波特率的产生

分数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值应设置成相同。

Tx / Rx 波特率 ＝*fCK* /(16**USARTDIV*)

这里的 *fCK* 是给外设的时钟 USARTDIV 是一个无符号的定点数。这 12 位的值设置在 USART_BRR 寄存器。

注：在写入 *USART_BRR* 之后，波特率计数器会被波特率寄存器的新值替换。因此，不要在通信进行中改变波特率寄存器的数值。

如何从 USART_BRR 寄存器值得到 USARTDIV

例 1:

如果 DIV_Mantissa = 27，DIV_Fraction = 12 (USART_BRR=0x1BC),

则:

Mantissa (USARTDIV) = 27

Fraction (USARTDIV) = 12/16 = 0.75

所以 USARTDIV = 27.75

例 2:

要求 USARTDIV = 25.62,

就有:

DIV_Fraction = 16*0.62 = 9.92

最接近的整数是： 10 = 0x0A

DIV_Mantissa = mantissa (25.620) = 25 = 0x19

于是， USART_BRR = 0x19A

例 3:

要求 USARTDIV = 50.99

就有:

DIV_Fraction = 16*0.99 = 15.84

最接近的整数是：16 = 0x10 => DIV_frac[3:0]溢出 => 进位必须加到小数部分

DIV_Mantissa = mantissa (50.990 + 进位) = 51 = 0x33

于是：USART_BRR = 0x330，USARTDIV=51

波特率		F_{PCLK}=36MHz			F_{PCLK}=72MHz		
序号	Kbps	实际	置于波特率寄存器中的值	误差（%）	实际	置于波特率寄存器中的值	误差（%）
1	2.4	2.400	937.5	0%	2.4	1875	0%
2	9.6	9.600	234.375	0%	9.6	468.75	0%
3	19.2	19.2	117.1875	0%	19.2	234.375	0%
4	57.6	57.6	39.0625	0%	57.6	78.125	0%
5	115.2	115.384	19.5	0.15%	115.2	39.0625	0%
6	230.4	230.769	9.75	0.16%	230.769	19.5	0.16%
7	460.8	461.538	4.875	0.16%	461.538	9.75	0.16%
8	921.6	923.076	2.4375	0.16%	923.076	4.875	0.16%

波特率		F_{PCLK}=36MHz			F_{PCLK}=72MHz		
序号	Kbps	实际	置于波特率寄存器中的值	误差（%）	实际	置于波特率寄存器中的值	误差（%）
9	2250	2250	1	0%	2250	2	0%
10	4500	不可能	不可能	不可能	4500	1	0%

注： *CPU* 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

30.3.5. USART 接收器容忍度

只有当整体的时钟系统地变化小于 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 M 位定义的 10 或 11 位字符长度
- 是否使用分数波特率产生

表 30-2 DIV_Fraction 为 0 时 USART 接收器的容忍度

M bit	NF is an error	NF is don't care
0	3.75%	4.375%
1	3.41%	3.97%

表 30-3 DIV_Fraction 非 0 时 USART 接收器容忍度

M bit	NF is an error	NF is don't care
0	3.33%	3.88%
1	3.03%	3.53%

30.3.6. USART 自动波特率检测

USART 能够基于一个字符的接收，检测并自动设定 USARTx_BRR 寄存器的值。自动波特率检测在以下场景是有用处的：

- 系统通讯速率提前未确认
- 系统正在使用相对低精度的时钟源，该机制允许在不测量时钟偏差的情况下，获得正确的波特率

时钟源频率必须与被期望的通讯速率兼容。（必须选择 16 倍过采样，并从 fCK/65535 和 fCK/16 之间选择）

在激活自动波特率检测之前，自动波特率检测模式必须被选择（通过 USARTx_CR3 寄存器的 ABRMOD[1:0]位被选择）。基于不同的字符模式，有各种模式。

在这些自动波特率模式下，波特率在同步数据接收期间会被测量几次，每次测量都会跟之前进行对比。

这些模式是：

MODE 0: 任何以 1 开始的字符。在这种情况下，USART 测量起始位的宽度（下降沿到上升沿）

MODE 1: 任何以 10xx 位开始的字符。在这种情况下，USART 测量起始位和第一个数据位的宽度。该测量在下降沿到下降沿之间进行，以确保在慢速信号上升情况下更好的精度。

另一个对每个 RX 的 transition 的检查也会并行进行。如果 RX 上的 transition 没有与接收器充分的同步（接收器基于 bit 0 计算的波特率），则会产生错误。

在激活自动波特率检测之前，USARTx_BRR 寄存器必须通过写一个 non-zero 波特率值进行初始化。

通过置位 USARTx_CR3 寄存器的 ABREN 位，可以激活自动波特率检测功能。USART 将等待 RX 上的第一个字符。置位 USARTx_ISR 寄存器的 ABRF 标志，显示了自动波特率检测的完成。如果通讯线上是有噪声的，则自动波特率检测的正确进行不能被保证。在这种情况下，BRR 的值可能被破坏，ABRE 错误标志位将被置位。如果通讯速度与自动波特率检测范围不兼容（位宽不在 16 和 65535 个时钟周期之间@16 位过采样），ABRE 错误也会发生。

RXNE 中断将显示了操作的完成。在以后的任何时刻，自动波特率检测可能通过复位 ABRF 标志（通过写 0）再次启动。

Note：如果在自动波特期间，清零 UE，BRR 值可能被破坏。

30.3.7. 多处理器通信

通过 USART 可以实现多处理器通信(将几个 USART 连在一个网络里)。例如某个 USART 设备可以是主，它的 TX 输出和其他 USART 从设备的 RX 输入相连接；USART 从设备各自的 TX 输出逻辑地与在一起，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置于静默模式。在静默模式里：

- 任何接收状态位都不会被设置。
- 所有接收中断被禁止。
- USARTx_CR1 寄存器中的 RWU 位被置 1。RWU 可以被硬件自动控制或在某个条件下由软件写入。

根据 USARTx_CR1 寄存器中的 WAKE 位状态，USARTx 可以用二种方法进入或退出静默模式。

- 如果 WAKE 位被复位：进行空闲总线检测。
- 如果 WAKE 位被设置：进行地址标记检测。

30.3.7.1. 空闲总线检测(WAKE=0)

当 RWU 位被写 1 时， USART 进入静默模式。当检测到一空闲帧时，它被唤醒。然后 RWU 被硬件清零，但是 USART_SR 寄存器中的 IDLE 位并不置起。 RWU 还可以被软件写 0。下图给出利用空闲总线检测来唤醒和进入静默模式的一个例子

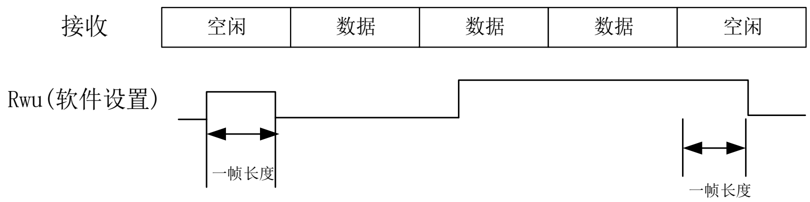


图 30-7 利用空闲总线检测的静默模式

30.3.7.2. 地址标记（Address mark）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，目标接收器的地址被放在 4 个 LSB 中。这个 4 位地址被接收器同它自己地址做比较，接收器的地址被编程在 USART_CR2 寄存器的 ADD。

如果接收到的字节与它的编程地址不匹配时，USART 进入静默模式。此时，硬件设置 RWU 位。

接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 USART 已经在静默模式。

当接收到的字节与接收器内编程地址匹配时，USART 退出静默模式。然后 RWU 位被清零，随后的字节被正常接收。收到这个匹配的地址字节时将设置 RXNE 位，因为 RWU 位已被清零。

当接收缓冲器不包含数据时(USART_SR 的 RXNE=0)，RWU 位可以被写 0 或 1。否则，该次写操作被忽略。下图给出利用地址标记检测来唤醒和进入静默模式的例子。

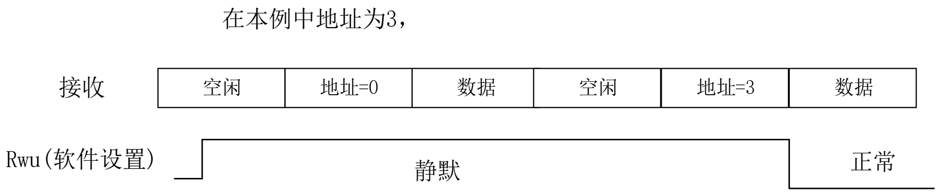


图 30-8 利用地址标记检测的静默模式

30.3.7.3. 校验控制

设置 USART_CR1 寄存器上的 PCE 位，可以使能奇偶控制(发送时生成一个奇偶位，接收时进行奇偶校验)。根据 M 位定义的帧长度，可能的 USART 帧格式列在下表中。

表 30-4 帧格式

M bit	PCE bit	USART fram
0	0	SB—8 bit data—STB
0	1	SB—7 bit data—PB—STB
1	0	SB—9 bit data—STB
1	1	SB—8 bit data—PB—STB

在用地址标记唤醒设备时，地址的匹配只考虑到数据的 *MSB* 位，而不用关心校验位。 *(MSB* 是数据位中最后发出的，后面紧跟校验位或者停止位*)*

30.3.7.4. 偶校验

校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中’1’的个数为偶数。

例如：数据=00110101，有 4 个’1’，如果选择偶校验(在 USARTx_CR1 中的 PS＝0)，校验位将是’0’。

30.3.7.5. 奇校验

此校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中’1’的个数为奇数。

例如：数据=00110101，有 4 个’1’，如果选择奇校验(在 USARTx_CR1 中的 PS = 1)，校验位将是’1’。

30.3.7.6. 传输模式

如果 USARTx_CR1 的 PCE 位被置位，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去(如果选择偶校验偶数个’1’，如果选择奇校验奇数个’1’)。如果奇偶校验失败，USART_SR 寄存器中的 PE 标志被置’1’，并且如果 USART_CR1 寄存器的 PEIE 在被预先设置的话，中断产生。

30.3.8. LIN(局域互联网)模式

配置 USART_CR2.LINEN=1 选择 LIN 模式。在 LIN 模式下，下列位必须保持位 0：

- USART_CR2 寄存器的 CLKEN;
- USART_CR3 寄存器的 STOP/SCEN/HDSEL/IREN.

30.3.8.1. 发送

与一般 USART 发送相比，LIN 发送有如下区别：

M=0，数据长度为 8bit；

需要配置 USART_CR2.LINEN=1。置位 SBK 后将发送 13bit 0 作为 break 帧。然后发送一位“1”，以允许对下一个 start 位检测。

30.3.8.2. 接收

当 LIN 模式被使能时，断开符号检测电路被激活。该检测完全独立于 USART 接收器。断开符号只要一出现就能检测到，不管是在总线空闲时还是在发送某数据帧其间，数据帧还未完成，又插入了断开符号的发送。

当接收器被激活时(USART_CR1 的 RE=1)，电路监测 RX 上的起始信号。监测起始位的方法同检测断开符号或数据是一样的。当起始位被检测到后，电路对每个接下来的位，在每个位的第 8，9，10 个过采样时钟点上进行采样。如果 10 个(当 USART_CR2 的 LBDL = 0) 或 11 个(当 USART_CR2 的 LBDL = 1)连续位都是 0，并且又跟着一个定界符， USART_SR 的 LBD 标志被设置。如果 LBDIE 位=1，中断产生。在确认断开符号前，要检查定界符，因为它意味 RX 线已经回到高电平。

如果在第 10 或 11 个采样点之前采样到了 1，检测电路取消当前检测并重新寻找起始位。如果 LIN 模式被禁止，接收器继续如正常 USART 那样工作，不需要考虑检测断开符号。

如果 LIN 模式没有被激活(LINEN=0)，接收器仍然正常工作于 USART 模式，不会进行断开检测。

如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误(也就是停止位检测到’0’，这种情况出现在断开帧)，接收器就停止，直到断开符号检测电路接收到一个 1(这种情况发生于断开符号没有完整的发出来)，或一个定界符(这种情况发生于已经检测到一个完整的断开符号)。

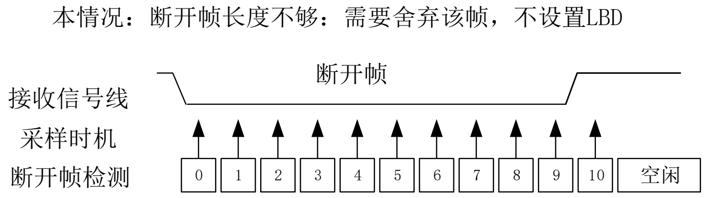


图 30-9 LIN 模式下的断开帧长度不够的情况

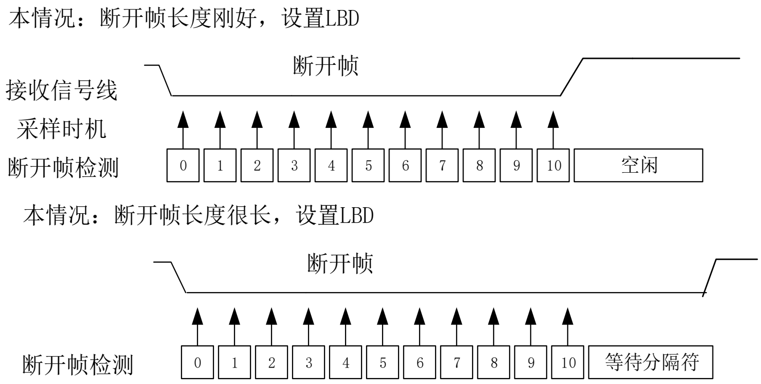


图 30-10 LIN 模式下的断开帧长度够的情况

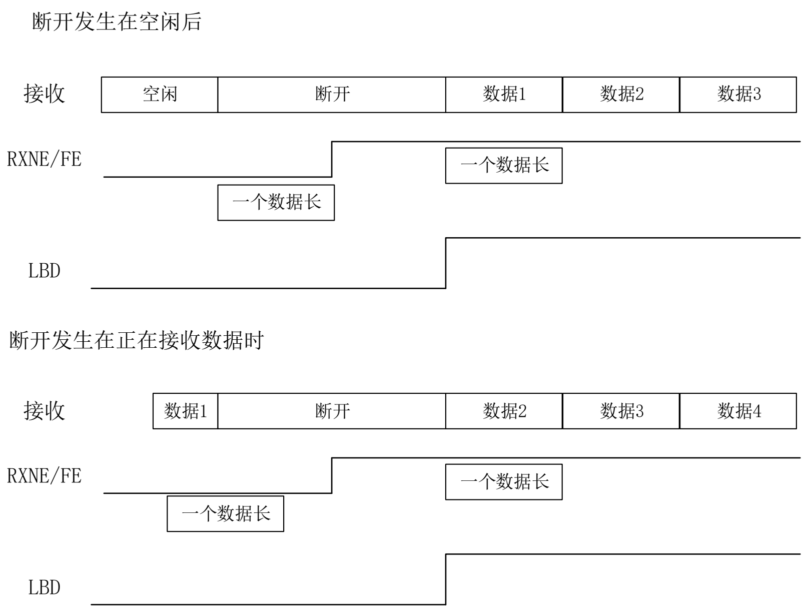


图 30-11 LIN 模式下的断开检测与帧错误的检测

30.3.9. USART 同步模式

通过写 USART_CR2 寄存器的 CLKEN 位为 1，选择同步模式。在同步模式里，下列位必须保持清零状态：

USART_CR2 寄存器中的 LINEN 位

USART_CR3 寄存器中的 SCEN,HDSEL 和 IREN 位

USART 允许用户以主模式方式控制双向同步串行通信。CK 脚是 USART 发送器时钟的输出。在起始位和停止位期间，CK 脚上没有时钟脉冲。根据 USART_CR2 寄存器中 LBCL 位的状态，决定在最后一个有效数据位期间产生或不产生时钟脉冲。USART_CR2 寄存器的 CPOL 位允许用户选择时钟极性，USART_CR2 寄存器上的 CPHA 位允许用户选择外部时钟的相位。

在总线空闲期间，实际数据到来之前以及发送断开符号的时候，外部 CK 时钟不被激活。

同步模式时，USART 发送器和异步模式里工作一模一样。但是因为 CK 是与 TX 同步的(根据 CPOL 和 CPHA)，所以 TX 上的数据是随 CK 同步发出的。

同步模式的 USART 接收器工作方式与异步模式不同。如果 RE=1，数据在 CK 上采样(根据 CPOL 和 CPHA 决定在上升沿还是下降沿)，不需要任何的过采样。但必须考虑建立时间和持续时间(取决于波特率，1/16 位时间)。

注意：

CK 脚同 *TX* 脚一起联合工作。因而，只有在使能了发送器*(TE = 1)*，并且发送数据时*(*写入数据至 *USART_DR* 寄存器*)*才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。

LBCL,CPOL 和 *CPHA* 位的正确配置，应该在发送器和接收器都被禁止时；当使能了发送器或接收器时，这些位不能被改变。

建议在同一条指令中设置 *TE* 和 *RE*，以减少接收器的建立时间和保持时间。

USART 只支持主模式：它不能用来自其他设备的输入时钟接收或发送数据*(CK* 永远是输出*)*。

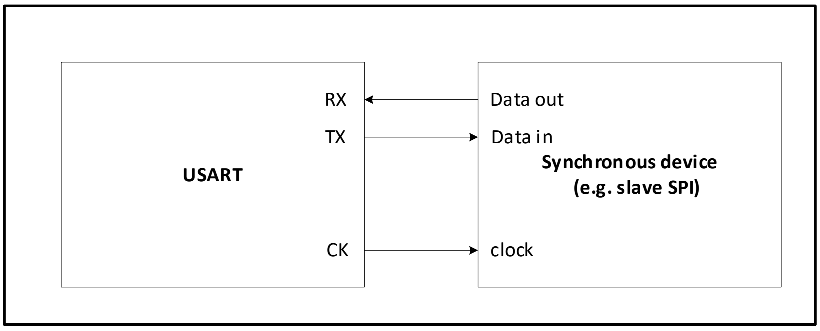


图 30-12 USART 同步传输的例子

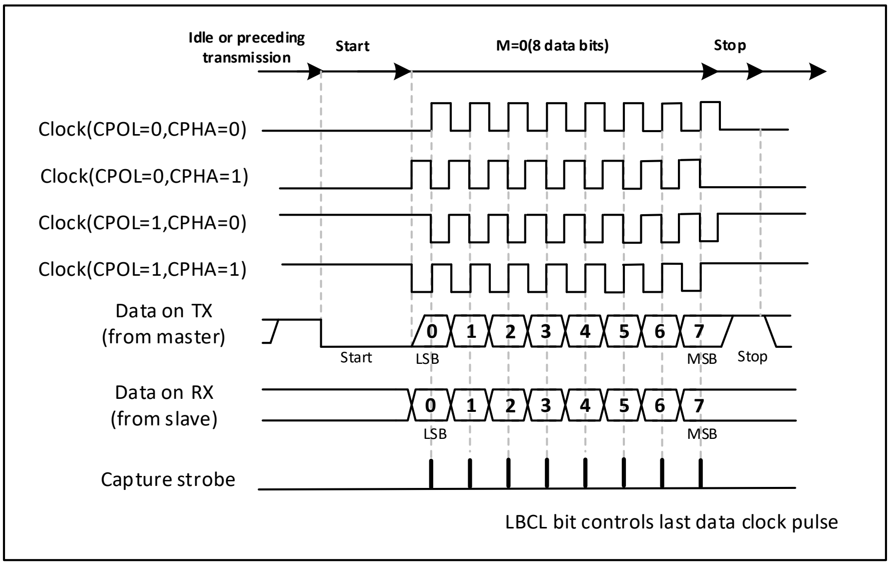


图 30-13 USART 数据时钟时序示例(M=0)

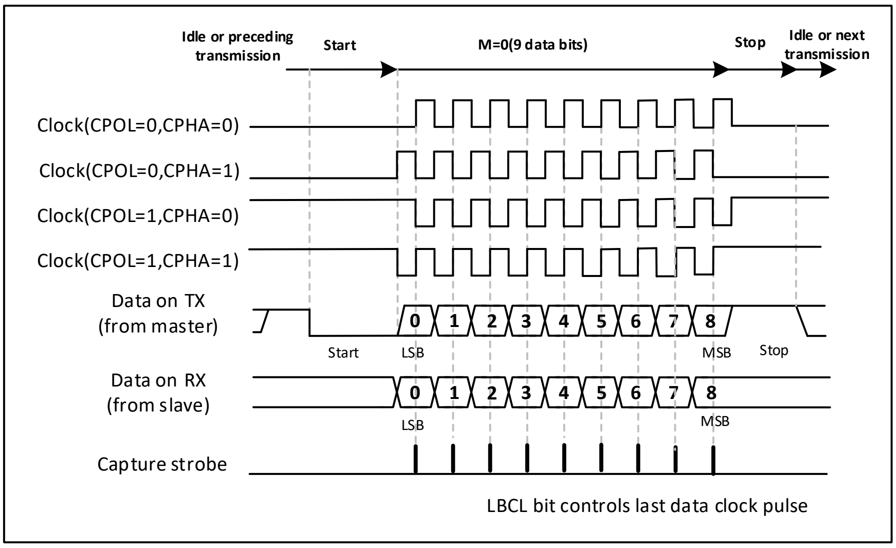


图 30-14 USART 数据时钟时序示例(M=1)

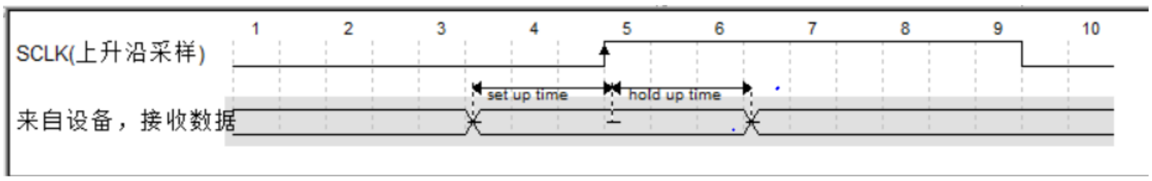


图 30-15 RX 数据采样/保持时间

30.3.10. 单线半双工通信

单线半双方模式通过设置 USARTx_CR3 寄存器的 HDSEL 位选择。在这个模式里，下面的位必须保持清零状态：

- USARTx_CR2 寄存器的 CLKEN 位

■ USART_CR3 寄存器的 SCEN 和 IREN 位

USART 可以配置成遵循单线半双工协议。在单线半双工模式下，TX 和 RX 引脚在芯片内部互连。使用控制位"HALF DUPLEX SEL"(USARTx_CR3 中的 HDSEL 位)选择半双工和全双工通信。

当 HDSEL 为'1'时

■ 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，发送就继续。

30.3.11. 智能卡

设置 USART_CR3 寄存器的 SCEN 位选择智能卡模式。在智能卡模式下，下列位必须保持清零：

■ USART_CR2 寄存器的 LINEN 位

■ USART_CR3 寄存器的 HDSEL 位和 IREN 位

此外， CLKEN 位可以被设置，以提供时钟给智能卡。

该接口符合 ISO7816-3 标准，支持智能卡异步协议。 USART 应该被设置为：

1. 8 位数据位加校验位： 此时 USART_CR1 寄存器中 M=1、 PCE=1

2. 发送和接收时为 1.5 个停止位：即 USART_CR2 寄存器的 STOP=11

注： 也可以在接收时选择 0.5 个停止位，但为了避免在 2 种配置间转换，建议在发送和接收时使用 1.5 个停止位。

下图给出的例子说明了数据线上，在有校验错误和没校验错误两种情况下的信号。

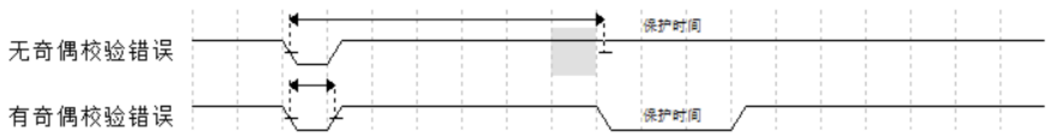


图 30-16 ISO7816-3 异步协议

当与智能卡相连接时， USART 的 TX 驱动一根智能卡也驱动的双向线。为了做到这点， SW_RX 必须和 TX 连接到相同的 I/O 口。在发送开始位和数据字节期间，发送器的输出使能位 TX_EN 被置起，在发送停止位期间被释放(弱上拉)，因此在发现校验错误的情况下接收器可以将数据线拉低。如果 TX_EN 不被使用，在停止位期间 TX 被拉到高电平：这样的话，只要 TX 配置成开漏，接收器也可以驱动这根线。

智能卡是一个单线半双工通信协议

1) 从发送移位寄存器把数据发送出去，要被延时最小 1/2 波特时钟。在正常操作时，一个满的发送移位寄存器将在下一个波特时钟沿开始向外移出数据。在智能卡模式里，此发送被延迟 1/2 波特时钟。

2) 如果在接收一个设置为 0.5 或 1.5 个停止位的数据帧期间，检测到一奇偶校验错误，在完成接收该帧后(即停止位结束时)，发送线被拉低一个波特时钟周期。这是告诉智能卡发送到 USART 的数据没有被正确地接收到。此 NACK 信号(拉低发送线一个波特时钟周期)在发送端将产生一个帧错误(发送端被配置成 1.5 个停止位)。应用程序可以根据协议处理重新发送数据。如果设置了 NACK 控制位，发生校验错误时接收器会给出一个 NACK 信号；否则就不会发送 NACK。

3) TC 标志的置起可以通过编程保护时间寄存器得以延时。在正常操作时，当发送移位寄存器变空并且没有新的发送请求出现时， TC 被置起。在智能卡模式里，空的发送移位寄存器将触发保护时间计数器

开始向上计数，直到保护时间寄存器中的值。 TC 在这段时间被强制拉低。当保护时间计数器达到保护时间寄存器中的值时， TC 被置高。

4) TC 标志的撤销不受智能卡模式的影响。

5) 如果发送器检测到一个帧错误(收到接收器的 NACK 信号)，发送器的接收功能模块不会把 NACK 当作起始位检测。根据 ISO 协议，接收到的 NACK 的持续时间可以是 1 或 2 波特时钟周期。

6) 在接收器这边，如果一个校验错误被检测到，并且 NACK 被发送，接收器不会把 NACK 检测成起始位。

注意:

1) 断开符号在智能卡模式里没有意义。一个带帧错误的 00h 数据将被当成数据而不是断开符号。

2) 当来回切换 TE 位时，没有 IDLE 帧被发送。 ISO 协议没有定义 IDLE 帧。

下图详述了 USART 是如何采样 NACK 信号的。在这个例子里， USART 正在发送数据，并且被配置成 1.5 个停止位。为了检查数据的完整性和 NACK 信号， USART 的接收功能块被激活。

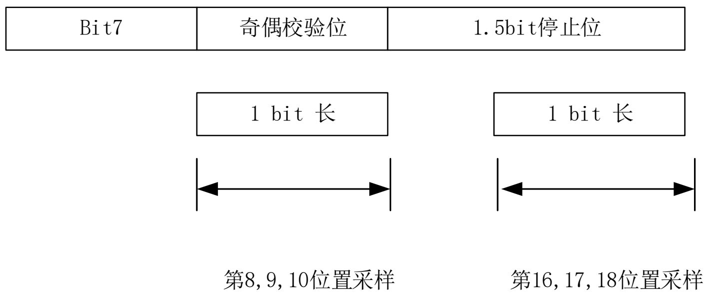


图 30-17 使用 1.5 停止位检测奇偶校验错

USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式里， CK 不和通信直接关联，而是先通过一个 5 位预分频器简单地用内部的外设输入时钟来驱动智能卡的时钟。分频率在预分频寄存器 USART_GTPR 中配置。 CK 频率可以从 fCK/2 到 fCK/62，这里的 fCK 是外设输入时钟。

30.3.12. IrDA SIR ENDEC 功能模块

通过设置 USART_CR3 寄存器的 IREN 位选择 IrDA 模式。在 IRDA 模式里，下列位必须保持清零:

- USART_CR2 寄存器的 LINEN,STOP 和 CLKEN 位
- USART_CR3 寄存器的 SCEN 和 HDSEL 位。

30.3.12.1. IrDA 正常模式

IrDA SIR 物理层规定使用反相归零调制方案(RZI)，该方案用一个红外光脉冲代表逻辑'0'(见图 4-12)。 SIR 发送编码器对从 USART 输出的 NRZ(非归零)比特流进行调制。输出脉冲流被传送到一个外部输出驱动器和红外 LED。 USART 为 SIR ENDEC 最高只支持到 115.2Kbps 速率。在正常模式里，脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 USART。在空闲状态里，解码器输入通常是高(标记状态 marking state)。发送编码器输出的极性和解码器的输入相反。当解码器输入低时，检测到一个起始位。

- IrDA 是一个半双工通信协议。如果发送器忙(也就是 USART 正在送数据给 IrDA 编码器)，IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙(也就是 USART 正在接收从 IrDA 解码器来的解码数据)，从

USART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。

- SIR 发送逻辑把’0’作为高脉冲发送，把’1’作为低电平发送。脉冲的宽度规定为正常模式时位周期的 3/16。
- SIR 接收逻辑把高电平状态解释为’1’，把低脉冲解释为’0’。
- 发送编码器输出与解码器输入有着相反的极性。当空闲时， SIR 输出处于低状态。
- SIR 解码器把 IrDA 兼容的接收信号转变成给 USART 的比特流。
- IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收器端的尖峰脉冲检测逻辑滤除宽度小于 2 个 PSC 周期的脉冲(PSC 是在 IrDA 低功耗波特率寄存器 USART_GTPR 中编程的预分频值)。宽度小于 1 个 PSC 周期的脉冲一定被滤除掉，但是那些宽度大于 1 个而小于 2 个 PSC 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。当 PSC=0 时， IrDA 编码器/解码器不工作。
- 接收器可以与一低功耗发送器通信。
- 在 IrDA 模式里， USART_CR2 寄存器上的 STOP 位必须配置成 1 个停止位。

30.3.12.2. IrDA 低功耗模式

发送器:

在低功耗模式，脉冲宽度不再持续 3/16 个位周期。取而代之，脉冲的宽度是低功耗波特率的 3 倍，它最小可以是 1.42MHz。通常这个值是 1.8432MHz(1.42 MHz < PSC < 2.12 MHz)。一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。

接收器:

低功耗模式的接收类似于正常模式的接收。为了滤除尖峰干扰脉冲， USART 应该滤除宽度短于 1 个 PSC 的脉冲。只有持续时间大于 2 个周期的 IrDA 低功耗波特率时钟(USART_GTPR 中的 PSC)的低电平信号才被接受为有效的信号。

注意:

1. 宽度小于 2 个大于 1 个 PSC 周期的脉冲可能会也可能不会被滤除。
2. 接收器的建立时间应该由软件管理。 IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时(IrDA 是一个半双工协议)。

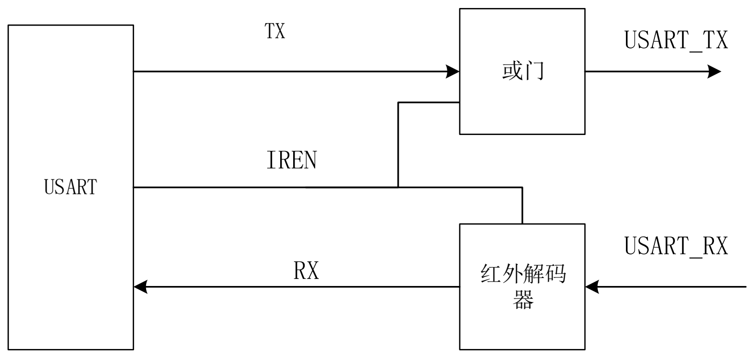


图 30-18 IrDA SIR ENDEC 框图

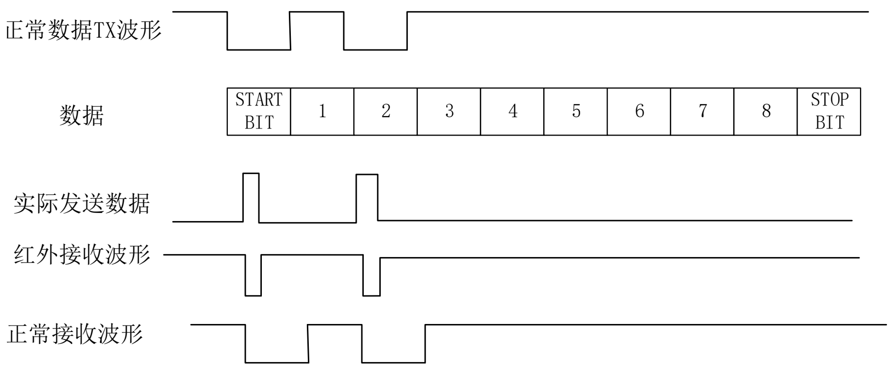


图 30-19 IrDA 数据调制(3/16)-普通模式

30.3.13. 用 DMA 连续通信

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。

30.3.13.1. 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。当 TXE 位被置为’1’时，DMA 就从指定的 SRAM 区传送数据到 USART_DR 寄存器。为 USART 的发送分配一个 DMA 通道的步骤如下(x 表示通道号)：

1）在 DMA 控制寄存器上将 USART_DR 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 控制寄存器上将存储器地址配置成 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_DR 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）通过写 0，清 TC 位。

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 USARTx_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；软件需要先等待 TXE=1，再等待 TC=1。

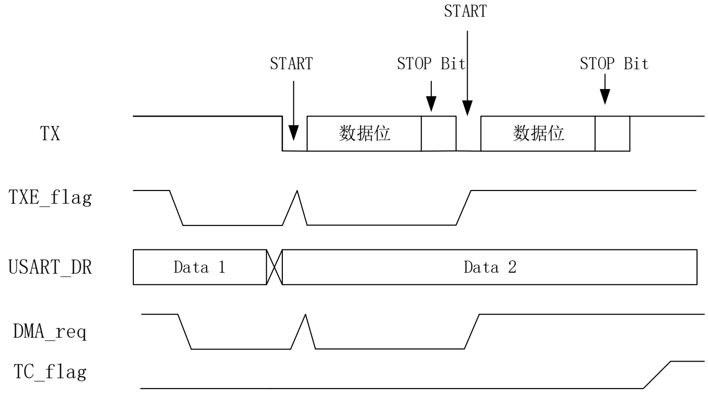


图 30-20 利用 DMA 发送

30.3.13.2. 利用 DMA 接收

可以通过设置 USART_CR3 寄存器的 DMAR 位激活使用 DMA 进行接收，每次接收到一个字节，DMA 控制器就就把数据从 USART_DR 寄存器传送到指定的 SRAM 区(参考 DMA 相关说明)。为 USART 的接收分配一个 DMA 通道的步骤如下(x 表示通道号)：

1）通过 DMA 控制寄存器把 USART_DR 寄存器地址配置成传输的源地址。在每个 RXNE 事件后，将从此地址读出数据并传输到存储器。

2）通过 DMA 控制寄存器把存储器地址配置成传输的目的地址。在每个 RXNE 事件后，数据将从 USART_DR 传输到此存储器区。

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 控制寄存器上激活该通道。

当接收完成 DMA 控制器指定的传输量时，DMA 控制器在该 DMA 通道的中断矢量上产生一中断。

30.3.13.3. 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，在当前字节传输后将置起错误标志。如果中断使能位被设置，将产生中断。在单个字节接收的情况下，和 RXNE 一起被置起的帧错误、溢出错误和噪音标志，有单独的错误标志中断使能位；如果设置了，会在当前字节传输结束后，产生中断。

30.3.14. 硬件流控制

利用 nCTS 输入和 nRTS 输出可以控制 2 个设备间的串行数据流。下图表明在这个模式里如何连接 2 个设备。

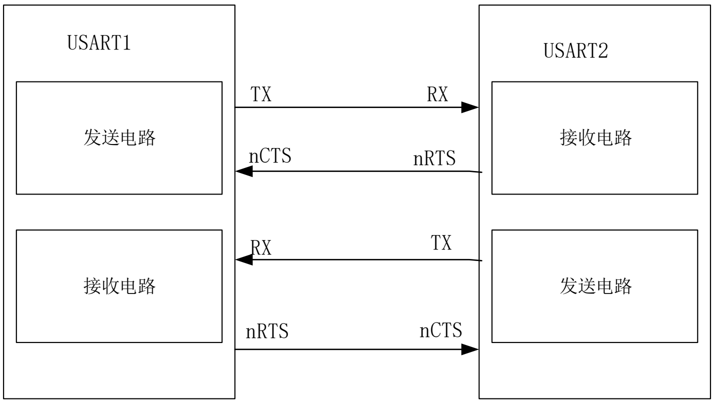


图 30-21 两个 USART 间的硬件流控制

30.3.14.1. 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)。当接收寄存器内有数据到达时，nRTS 被释放，由此表明希望在当前帧结束时停止数据传输。

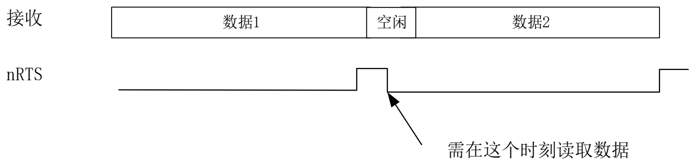


图 30-22 RTS 流控制

30.3.14.2. 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 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，当前的传输完成后停止发送。

当 CTSE=1 时，只要 nCTS 输入一变换状态，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。

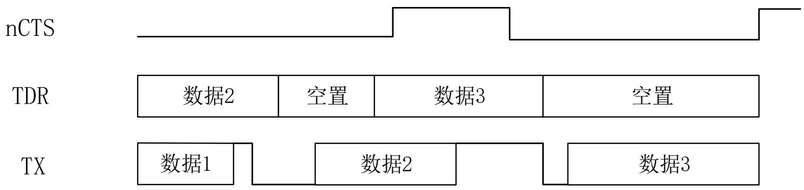


图 30-23 CTS 流控制

30.4. USART 中断请求

序号	中断事件	事件标志	使能位	发送/接收
1	发送数据寄存器空	TXE	TXEIE	发送
2	CTS（Clear to Send）中断	CTSIF	CTSIE	发送

序号	中断事件	事件标志	使能位	发送/接收
3	传送完成	TC	TCIE	发送
4	接收寄存器非空（读数据准备好）	RXNE	RXNEIE	接收
5	Overrun 错误	ORE		接收
6	空闲帧	IDLE	IDLEIE	接收
7	奇偶校验错误	PE	PEIE	接收
8	断开标志	LBD	LBDIE	接收
9	多处理器通讯时，噪声、overrun 和帧错误	NR/ORE/FE	EIE	接收

所有 USART 中断共用同一个中断向量。

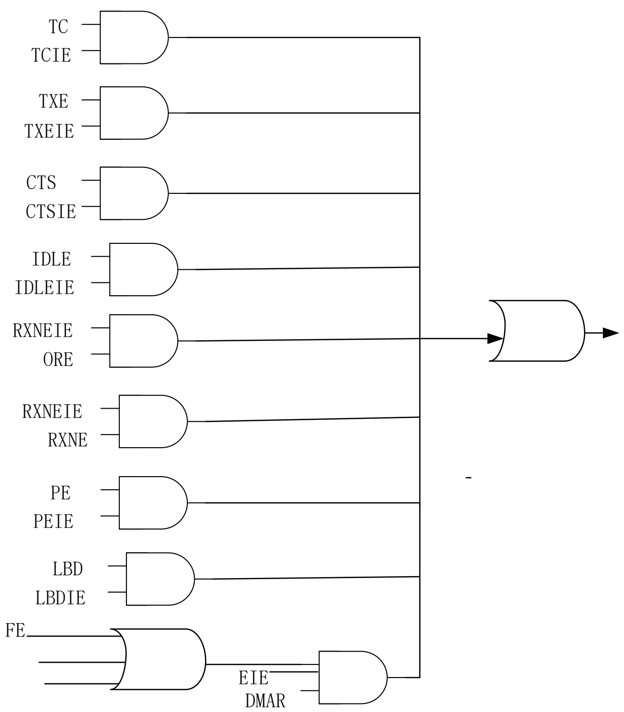


图 30-24 USART 中断映像图

30.5. USART 寄存器

30.5.1. 状态寄存器 (USART_SR)

Address offset:0x00

Reset value:0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	ABRRQ	ABRE	ABRF	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
			W	R	R	RC_W0	Rc_W0	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	RES	-	Reserved
12	ABRRQ	W	0	自动波特率请求。 该位写 1 会复位 ABRF 标志位，并且请求下一帧的自动波特率检测。
11	ABRE	R	0	自动波特率错误标志。 当自动波特率检测出错（波特率超出范围或者字符比较错误）时，硬件置位该寄存器。 软件通过写 1 到 ABRRQ 寄存器清零该位。
10	ABRF	R	0	自动波特率检测标志。 当自动波特率设置（同时设置 RXNE=1，当中断使能后产生中断），或者自动波特率检测操作出错（ABRE=1，RXNE=1，FE=1）时该位由硬件置 1。 软件通过写 1 到 USART_RQR 寄存器的 ABRRQ 位清零该位。
9	CTS	RC_W0	0	CTS 标志。 当 CTS 输入 toggle，别 CTSE=1 时，该寄存器为 1.软件写 0 清零。当 CTSIE=1 时，产生 CTS 中断。 0：CTS line 值未改变 1：CTS line 值改变
8	LBD	Rc_W0	~~0~~	LBD: LIN 断开检测标志 (LIN break detection flag) 当探测到 LIN 断开时，该位由硬件置‘1’，由软件清‘0’(向该位写 0)。如果 USART_CR3 中的 LBDIE = 1，则产生中断。 0：没有检测到 LIN 断开； 1：检测到 LIN 断开。 注意：若 LBDIE=1，当 LBD 为’1’时要产生中断
7	TXE	R	1	传输寄存器空标志。 当 USART_DR 寄存器数据传送到移位寄存器，硬件置位该寄存器。当 TXEIE=1 时，产生中断。写 USART_DR 寄存器会清零该位。 0：数据未传送到移位寄存器 1：数据传送到移位寄存器
6	TC	RC_W0	1	传送完成标志。 传送数据帧完成后，且 TXE=1，则硬件置位该寄存器。TCIE=1 时产生中断。 软件先读 USART_SR 寄存器然后写 USART_DR 寄存器会清零该位（针对多处理器通讯）。软件同时可以写 0 清零。 0：传送未完成 1：传送完成
5	RXNE	RC_W0	0	读数据寄存器不空标志。

Bit	Name	R/W	Reset Value	Function
				当移位寄存器值传送到 USART_DR 寄存器，硬件置位该寄存器。 软件读 USART_DR 寄存器、或者写 0 清零该位。 当 RXNEIE=1 时，产生中断。 0：未收到数据 1：接收数据准备好
4	IDLE	R	0	空闲标志。 检测 IDLE line，硬件置位该寄存器。当 IDLEIE=1 时产生中断。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 0：未检测到 IDLE line 1：检测到 IDLE line
3	ORE	R	0	Overrun 错误标志。 当 RXNE=1 时，在移位寄存器中接收到的数据正准备转移到 RDR 寄存器时，硬件设置该位。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 当 RXNEIE=1 时，产生中断。 0：未产生 Overrun 错误 1：产生 Overrun 错误 注：该寄存器置位时，RDR 寄存器内容不会丢失，但移位寄存器内容被覆盖。 当 EIE=1 时，产生 ORE 中断。
2	NE	R	0	噪声错误标志。 在数据帧接收到噪声时，硬件置位该寄存器。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 0：未检测到噪声错误 1：检测到噪声错误 注：当 RXNE 与 NE 同时产生时，NE=1 时不产生中断，而在设置 RXNE 标志时产生中断。在多缓冲器通讯模式下，当 EIE=1 时 NE=1 会产生中断。
1	FE	R	0	帧错误标志。 当检测到不同步、过多的噪声或中止字符时，由硬件设置此位。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。 0：未检测到帧错误 1：检测到帧错误或 break 字符 注：当 RXNE 与 FE 同时产生时，FE=1 时不产生中断，而在设置 RXNE 标志时产生中断。如果当前传输的数据既产生了帧错误，又产生了过载错误，硬件还是会继续该数据的传输，并且只设置

Bit	Name	R/W	Reset Value	Function
				ORE 标志位。在多缓冲器通讯模式下，当 EIE=1 时 FE=1 会产生中断。
0	PE	R	0	校验值错误。 当接收时校验值错误时，硬件置位该寄存器。 软件先读 USART_SR 寄存器后读 USART_DR 寄存器可以清零该位。但软件在清该位前必须等待 RXNE=1. 当 PEIE 时，产生中断。 0：未产生奇偶校验错误 1：产生奇偶校验错误

30.5.2. 数据寄存器 (USART_DR)

Address offset: 0x04

Reset value: undefined

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

Bit	Name	R/W	Reset Value	Function
31：9	Reserved	RES	-	Reserved
8：0	DR[8:0]	RW	undefined	接收/发送数据寄存器。 取决于读还是写操作，前者是接收到的数据，后者是发送的数据。 DR 寄存器物理上由两个寄存器组成（一个是发送的 TDR，一个是接收的 RDR），所以 DR 寄存器实现了读和写的两个功能。 TDR 寄存器在内部总线和输出移位寄存器之间提供了并行的接口，RDR 寄存器在输入移位寄存器和内部总线之间提供了并行接口。 当奇偶校验使能打开进行发送操作时，写 MSB 位（bit7 或者 bit8）是无效的，因为已被校验位代替了。 当奇偶校验使能打开进行接收操作时，读出的 MSB 位是被接收到的校验位。

30.5.3. 波特率寄存器 (USART_BRR)

Address offset:0x08

Reset value:0x0000_0000

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

在自动波特率检测模式，硬件更新该寄存器。

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	RES	-	Reserved
15：4	DIV_Mantissa[15:4]	RW	0	12bit 整数
3：0	DIV_Fraction[3:0]	RW	0	4bit 小数

30.5.4. 控制寄存器 1 (USART_CR1)

Address offset:0x0C

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：14	Reserved	RES	-	Reserved
13	UE	RW	0	USART 使能。当该位清零后，USART 模块会立即停止当前操作。该位由软件置位和清零。 0：USART prescaler 和 output 禁止，low-power 模式 1：USART 使能 软件需要等待 USART_ISR.TC 置位后，才能清零 UE 位，进入低功耗模式； 同时，在清零 UE 位之前 DMA 通道需要禁止。
12	M	RW	0	0：1 start bit，8 data bits，n stop bit 1：1 start bit，9 data bit，n stop bit
11	WAKE	RW	0	接收唤醒方式。 从 mute 模式唤醒方式。由软件置位或者清零。 0：Idle line 唤醒 1：地址唤醒
10	PCE	RW	0	奇偶校验控制。 0：奇偶校验禁止 1：奇偶校验使能 奇偶校验位：9bit 的第 9 位；8bit 的第 8 位。
9	PS	RW	0	奇偶校验选择。由软件置位和清零。 0：偶校验 1：奇校验
8	PEIE	RW	0	PE 中断使能。由软件置位和清零。 0：禁止 1：PE 中断使能
7	TXEIE	RW	0	TXE 中断使能。由软件置位和清零。 0：禁止 1：TXE 中断使能
6	TCIE	RW	0	传送结束中断使能。由软件置位和清零。

Bit	Name	R/W	Reset Value	Function
				0：禁止 1：TC 中断使能
5	RXNEIE	RW	0	RXNE 中断使能；由软件置位和清零。 0：禁止 1：ORE 或者 RXNE 中断使能
4	IDLEIE	RW	0	IDLE 中断使能。由软件置位和清零。 0：禁止 1：IDLE 中断使能
3	TE	RW	0	传送使能。 0：传送禁止 1：传送使能
2	RE	RW	0	接收使能。 0：接收禁止 1：接收使能，开始检测 start 位
1	RWU	RW	0	接收唤醒。 该位表明 USART 是否为 mute 模式。 当接收到 mute 模式序列，该寄存器置位；如果接收到唤醒序列，该寄存器清零。具体哪种唤醒序列（地址或者 IDLE）由寄存器 USART_CR1.WAKEbit 控制。 0：接收器为工作模式 1：接收器为静默模式 注 1：在设置该位进入 mute 模式前，USART 要已经先接收了一个数据字节，否则在 mute 模式下，不能被 idle 总线检测唤醒。 注 2：当配置成地址标记检测唤醒（WAKE=1），在 RXNE 被置位时，不能用软件修改 RWU 位。
0	SBK	RW	0	发送 break 帧。 软件置位该寄存器，发送 break 字节。Break 帧的 stop 位发送后，硬件清零该寄存器。 0：不发送 break 字节 1：发送 break 字节

30.5.5. 控制寄存器 2 (USART_CR2)

Address offset:0x10

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	Res	ADD[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW	RW		RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	RES	-	Reserved

Bit	Name	R/W	Reset Value	Function
14	LINEN	RW	0	LIN 模式使能。 软件置位和清零。 0：LIN 模式； 1：使能 LIN 模式； LIN 模式下，通过使能 SBK 位，发送 LIN 同步 breaks（13 低位），以及检测 Lin 同步断开符。
13：12	STOP[1:0]	RW	0	Stop 位配置。 00：1 stop bit； 01：0.5stop； 10：2 stop 位； 11：1.5stop
11	CLKEN	RW	0	CK pin 使能。 0：禁止； 1：CK pin 使能； 不支持同步模式时，该位保留。
10	CPOL	RW	0	时钟极性。 同步模式，CK pin 输出时钟极性。 0：传输窗外，CK pin 为稳定低值； 1：传输窗外，CK pin 为稳定高值；
9	CPHA	RW	0	该位在同步模式下用于选择 CK pin 输出时钟的相位。它与 CPOL 位一起工作，以产生所需的时钟/数据关系。 0：第一个时钟传输是首个数据捕获沿； 1：第二个时钟传输是首个数据捕获沿；
8	LBCL	RW	0	最后一位数据的时钟脉冲是否在 CK pin 输出。 0：最后一位数据的时钟脉冲不在 CK pin 输出； 1：最后一位数据的时钟脉冲在 CK pin 输出；
7	Reserved	RES	-	Reserved
6	LBDIE	RW	0	LIN break 中断使能。 0：禁止； 1：中断产生； 通过这去控制 USART_SR 寄存器中的 LBD,使得 LBD 为 1，产生中断
5	LBDL	RW	0	LIN break 检测长度。 0：10bits 检测 break； 1：11bits 检测 break；
4	Reserved	RES		Reserved
3:0	ADD[3:0]	RW	4’b0	USART 地址。 该寄存器用于多处理器 mute 模式，用作 4bit 地址唤醒时的地址。

30.5.6. 控制寄存器 3 (USART_CR3)

Address offset:0x14

Reset value: 0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ABR-MOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	Res	NACK	HDSEL	IRLP	IREN	EIE
	RW		RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：15	Reserved	RES	-	Reserved
14：13	ABRMOD[1:0]	RW	2’b0	自动波特率检测模式。 00：从 start 位开始测量波特率 01：下降沿到下降沿测量 10： Reserved 11： Reserved 当 ABREN=0 或者 UE=0 时，该寄存器只写。
12	ABREN	RW	0	自动波特率使能。 0：禁止 1：自动波特率使能
11	OVER8	RW	0	Oversampling 模式。 0：Oversampling by 16 1：Oversampling by 8 该位仅在 UE=0 时可被写。
10	CTSIE	RW	0	CTS 中断使能。 0：禁止； 1：CTSIF 中断使能；
9	CTSE	RW	0	CTS 使能。 0：CTS 硬件流控制禁止； 1：CTS 模式使能。当有当 CTS 输入为 0 时，才会传输数据。此时，当数据写入数据寄存器后，要等待 CTS 有效后才会启动传输。
8	RTSE	RW	0	RTS 使能。 0：RTS 硬件流控制禁止； 1：RTS 输出使能，只有当接收 buffer 未满时才会请求下一个数据。当前数据发送完成后，发送操作暂停。如果可以接收数据了，将 RTS 置为有效（0）。
7	DMAT	RW	0	传送时使能 DMA。 0：禁止； 1：传送时使能 DMA；
6	DMAR	RW	0	接收时使能 DMA。 0：禁止； 1：接收时使能 DMA；
5	SCEN	RW	0	智能卡模式使能。 0：禁止； 1：使能；
4	NACK	RW	0	智能卡 NACK 使能。 0：奇偶校验错误时发送 NACK 禁止； 1：奇偶校验错误时发送 NACK 使能；

Bit	Name	R/W	Reset Value	Function
3	HDSEL	RW	0	半双工选择。 0：非半双工模式； 1：半双工模式选择；
2	IRLP	RW	0	IrDA 低功耗。 0：normal 模式； 1：IrDA 低功耗模式；
1	IREN	RW	0	IrDA 模式使能。 软件使能和清零该寄存器。 0：IrDA 禁止； 1：IrDA 使能；
0	EIE	RW	0	错误中断使能。 0：禁止； 1：帧错误 FE、overrun 错误 ORE、噪声 NF 中断使能。

30.5.7. 保护时间和预分频器(USART_GTPR)

Address offset:0x18

Reset value: 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	RES	0	Reserved
15：8	GT[7:0]	RW	0	保护时间值（Guard time value）。 该域定义了以波特时钟为单位的保护时间。在智能卡模式，需要此功能。当保护时间过去后，才会设置发送完成标志。
7：0	PSC[7:0]	RW	0	预分频器值 (Prescaler value)。 ● 在红外(IrDA)低功耗模式下： PSC[7:0]=红外低功耗波特率。 对系统时钟分频以获得低功耗模式下的频率： 源时钟被寄存器中的值(仅有 8 位有效)分频 00000000：保留 – 不要写入该值； 00000001：对源时钟 1 分频； 00000010：对源时钟 2 分频； …… ● 在红外(IrDA)的正常模式下： PSC 只能设置为 00000001。 ● 在智能卡模式下： PSC[4:0]：预分频值 对系统时钟进行分频，给智能卡提供时钟。

Bit	Name	R/W	Reset Value	Function
				寄存器中给出的值(低 5 位有效)乘以 2 后，作为对源时钟的分频因子。 00000：保留 – 不要写入该值； 00001：对源时钟进行 2 分频； 00010：对源时钟进行 4 分频； 00011：对源时钟进行 6 分频； …… 注意：位[7:5]在智能卡模式下没有意义。

30.5.8. USART 寄存器映像

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	USART_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABRR	ABRE	ABRF	CTS	LBD.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
	Reset value																				0	0	0	0	0	1	1	0	0	0	0	0	0
0x04	USART_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[8:0]								
	Reset value																								0	0	0	0	0	0	0	0	0
0x08	USART_BRR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DIV_Mantissa[11:0]												DIV_Faction[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	USART_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEI	IDLEIE	TE	RE	RWU	SBK
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	USART_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LINEN.	STOP		CLKEN	CPOL	CPHA	LBCL	Res.	LBDIE.	LBDL.	Res.	ADD[3:0]			
	Reset value																		0	0		0	0	0	0		0	0		0	0	0	0
0x14	USART_CR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ABR-MOD[1:0]		ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	USART_GTPR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GT[7:0]							PSC[7:0]						
	Reset value																			0	0	0	0	0	0	0	0	0	0	0	0	0	0

31. 调试支持

31.1. 概况

本芯片基于 Cortex-M0+ CPU，该 CPU Core 包含高级 debug 硬件扩展功能。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。

调试功能在由调试主机在连接和调试 MCU 时使用，调试的接口是 serial wire。在 M0+ CPU Core 中的调试功能是一套 ARM CoreSight Design kit。

M0+提供了集成的片上调试支持，由以下部分组成：

- SW-DP：serial wire
- BPU：Break point unit
- DWT：Data watchpoint trigger

调试支持也包括了本芯片的调试集成功能：

- 灵活的调试引脚分配，SWIO@PA13、SWCLK@PA14
- MCU 调试盒（支持低功耗模式，控制外设时钟等

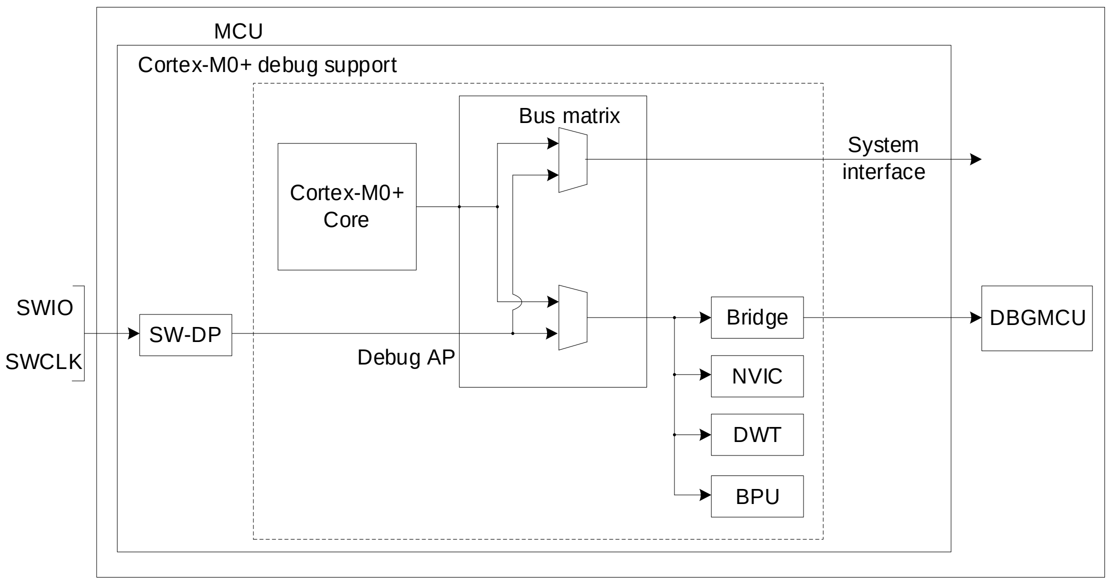


图 31-1 DBG 框图

31.2. 引脚分布和调试端口脚

31.2.1. SWD 调试端口

调试功能相关的端口有两个，在所有封装形式都可见。

表 31-1 DBG 框图

<table>
<tr><th rowspan="2">SW-DP
端口引脚名称</th><th colspan="2">SW 调试接口</th><th rowspan="2">引脚分配</th></tr>
<tr><th>类型</th><th>调试功能</th></tr>
<tr><td>SWDIO</td><td>输入/输出</td><td>串行数据输入/输出</td><td>PA13</td></tr>
<tr><td>SWDCLK</td><td>输入</td><td>串行时钟</td><td>PA14</td></tr>
</table>

31.2.2. 灵活的 SW-DP 脚分配

在芯片复位后（系统复位或者上电复位），用作 SW-DP 的端口被分配作为专门被调试主机立即使用的 pin。

然而，芯片提供了关闭 SWD 端口的可能性，并释放该端口作为 GPIO 用。

31.2.3. SWD 脚上的内部上拉和下拉

一旦 SWD 端口被软件释放，则 GPIO 控制器控制了这两个端口。GPIO 控制寄存器的复位状态把 IO 置为同等的状态：

- SWDIO：input pull-up
- SWCLK：input pull-down

片内的上拉和下拉电阻为外围节省了增加电阻的需求。

31.3. ID 代码和锁定机制

芯片内存放 ID code。推荐 Keil、IAR 等工具使用该 ID Code（位于 0x4001 5800 地址）锁住调试。

芯片上电后，硬件读取 flash 的 factory config. byte 的 0x1FFF 0FF8 地址，装载到 DBG_IDCODE 寄存器中。

31.4. SWD 调试端口

31.4.1. SWD 协议介绍

这是个同步的串行通讯协议，使用以下两个端口：

- SWCLK：来自主机给芯片的 clock 信号
- SWDIO：双向数据信号

该协议允许两个 bank 的寄存器（DPACC 寄存器和 APACC 寄存器）被读和写入。数据位是按照在线上的 LSB-first 传输。对于 SWDIO 的双向管理，线上必须在板级上拉（推荐 100k 欧的电阻）。

在协议中每次 SWDIO 方向的改变，转向时间被插入在线上既没有被主机，也没有被芯片驱动的情况。缺省状态下，这个转向时间是 1 个位的时间，然而整个可以通过配置 SWCLK 频率来调整。

31.4.2. SWD 协议序列

每个序列由以下阶段组成：

- 主机发送的包请求（8 bits）
- 芯片发送的应答响应（3 bits）
- 主机或者芯片的数据发送阶段（33 bits）

表 31-2 请求包(8-bits)

比特位	名称	描述
0	Start	必须为“1”
1	ApnDP	0：DP 访问 1：AP 访问
2	RnW	0：写请求 1：读请求
4:3	A[3:2]	DP 或者 AP 寄存器的地址区域
5	Parity	以前位的校验位
6	Stop	0
7	Park	没有被主机驱动。由于上拉属性，会被芯片读出 1。

通常转向时间（缺省为 1bit）跟随着包请求，此时主机和芯片都没有驱动信号线。

表 31-3 ACK 响应（3 bits）

比特位	名称	描述
[2:0]	ACK	001：FAULT 010：WAIT 100：OK

如果一个读操作或者如果 1 个 wait 或者 FAULT 应答被接收到，则转向时间必须跟随 ACK 响应。

表 31-4 DATA 传输（33bits）

比特位	名称	描述
[31:0]	WDATA 或者 RDATA	写或者读数据
32	校验位	对[31:0]的奇偶校验位

如果是个读操作时，转向时间必须跟随着数据传输。

31.4.3. SW-DP 状态机(reset, idle states, ID code)

SW-DP 的状态机有个定义了 SW-DP 的内部 ID 代码。它遵循 JEP-106 标准。这个 ID 代码是缺省的 ARM 代码，并被置位 0x0BB11477（对应 Cortex-M0）。

31.4.4. DP and AP 读/写访问

- 读 DP 的操作不会被 posted: 芯片响应可以被立即（ACK=OK），或者可以被延迟（ACK=WAIT）
- 读 AP 的操作被 posted：这意味着访问的结果被返回到下一次传输。如果下一次要进行的访问不是 AP 访问，则 DP-RDBUFF 寄存器必须被地呼出获得该结果。

  DP-CTRL/STAT 寄存器的 READOK 标志在每个 AP 读访问或者 RDBUFF 读请求（知道是否 AP 读访问是成功的）时被更新。
- SW-DP 实现了写 buffer（对于 DP 和 AP 写），这甚至当其他操作仍未完成时，接收一个写操作。如果写 buffer 满了，芯片应答响应是“WAIT”。IDCODE 读、CTRL/STAT 读或者 ABORT 写，是例外（甚至当如果写 buffer 是满的）

■ 由于 SWCLK 和 HCLK 是异步时钟，在写操作后（校验位之后）需要两个额外的 SWCLK 周期，用来确保写的内部有效性。当驱动信号线为低时，这几个周期应该被应用。

当为上电请求写 CTRL/STAT 时，以上尤其重要。如果下个操作（需要上电）立即出现，则会 fail。

31.4.5. SW-DP 寄存器

当 ApnDP=0 时，可以访问这些寄存器。

A[3:2]	R/W	CTRLSEL 位或者 SELECT 寄存器	Register	Notes
00	Read		IDCODE	
00	Write		ABORT	
01	Read/Write	0	DP-CTRL/STAT	
01	Read/Write	1	WIRE CONTROL	
10	Read		READ RESEND	
10	Write		SELECT	
11	Read/Write		READ BUFFER	

31.4.6. SW-AP 寄存器

Address	A[3:2] value	Description
0x0	00	Reserved
0x4	01	DP CTRL/STAT 寄存器，用作 ■ 请求一个系统或者调试的 power-up ■ 为 AP 访问配置传输操作 ■ 控制被 pushed 比较和被 pushed 验证操作 ■ 读一些状态标志（溢出、power-up 应答）
0x8	10	DP SELECTRION 寄存器：用作选择当前访问端口和 active 4 个 word 的寄存器在窗口。 ■ Bit 31:24：APSEL：选择当前 AP ■ Bit 23:8：reserved ■ Bit 7:4：APBANKSEL：在当前 AP,选择 active 4 个 word 寄存器窗口 ■ Bit 3:0：reserved
0xC	11	DP RDBUFF 寄存器：用于提供调试者在一个操作序列后，得到最终的结果（不用请求新的 JTAG-DP 操作）

31.5. 内核调试

通过 core debug 寄存器，可以访问 Core debug。Debug 访问这些寄存器是通过 debug 访问端口。他由下面四个寄存器组成

表 31-5 内核调试寄存器

寄存器	描述
DHCSR	32bit Debug halting control and status register
DCRSR	17bit Debug Core register selector register
DHCDR	32bit debug Core register Data register
DEMCR	32bit debug exception and monitor control register

这些寄存器不会被系统复位，复位掉。他们进会被上电复位，复位掉。为了在复位时 Hart，需要：

- 调试和例外监视控制寄存器的 bit0（VC_CORRESET），被使能
- 调试停止控制和状态寄存器，被使能

31.6. BPU 断点单元(Break Point Unit)

Cortex-M0+ BPU 实现提供了 4 个断点寄存器。BPU 是一套 ARMv7-M 的 flash 补丁和断点（FPB）Block（Cortex-M3 & Cortex-M4）。

31.6.1. BPU 功能

处理器断点实现基于 PC 的断点功能。

参考 ARMv6-M ARM 和 ARM Coresight Components Technical Reference Manual，以获得更多关于 BPU Coresight 的身份寄存器和他们的地址和访问种类。

31.7. 数据观察点 DWT (Data Watchpoint)

Cortex-M0 DWT 实现提供了 2 个 watchpoint 寄存器。

31.7.1. DWT 功能

处理器的断点实现基于 PC 的断点功能。

31.7.2. DWT 程序计数器样本寄存器

实现数据 watchpoint 单元的处理器，也实现了 ARMv6-M 可选的 DWT Program Counter Sample register(DWT_PCSR)。该寄存器允许调试者周期性的采样 PC，而不用停止处理器。这个机制提供了粗粒度分析。

CORTEX-M0+ DWT_PCSR 记录了通过了条件代码的指令和未通过的指令。

31.8. MCU 调试模块 (DBGMCU)

MCU debug component 帮助调试者提供以下支持：

- 低功耗模式
- 对 timer、watchdog、I2C 在 breakpoint 期间的时钟控制
- 对跟踪脚分配的控制

MCUDBG 寄存器还提供芯片 ID 编码。使用 JTAG 或者 SW 调试接口，或者用户程序都可以访问此 ID 编码。

31.8.1. 低功耗模式的调试支持

为进入低功耗模式，要执行 WFI 或者 WFE 指令。MCU 进入低功耗模式，或者是将 CPU Clock 停止掉，或者是减少 CPU 的功耗。

CPU 不允许在 debug 期间，停掉 FCLK 或者 HCLK。由于这些是调试者连接的需要，在一个调试期间，他们必须保持开启。MCU 集成了特殊的方法，允许用户在低功耗模式下调试软件。

因此，调试者主机必须先置某些调试配置寄存器的内容，以改变低功耗行为：

- 在 sleep 模式：FCLK 和 HCLK 仍然有效。相应的，该模式不能引起任何对于标准调试功能的限制。
- 在 stop 模式：DBG_STOP 位必须被调试者提前置位。

31.8.2. 支持定时器、看门狗和 I2C 的调试

在一个 breakpoint 期间，是有必要选择 timer 的计数器和 watchdog 要怎样的行为：

- 他们可以继续在 breakpoint 里计数。例如，这是当一个 PWM 正在控制电机时通常被需要的。
- 他们可以停下来在 breakpoint 内部计数。这是 watchdog 的特性决定的。

对于 I2C，用户可以选择在断点期间阻止 SMBUS 超时

31.9. DBG 寄存器

31.9.1. DBG 设备 ID 代码寄存器(DBG_IDCODE)

Address offset: 0x00

仅支持 32-bit 地址访问，只读。

该寄存器可以通过软件 debug 端口（2 pin）或者用户软件访问。

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

Bit	Name	R/W	Reset Value	Function
31：0	REV_ID	R		Revision ID

31.9.2. 调试 MCU 配置寄存器 (DBGMCU_CR)

该寄存器配置在 debug 状态下的 MCU 低功耗模式。

该寄存器会被上电复位进行异步复位（不是系统复位）。它可以在系统复位下被调试者进行写操作。

如果调试者主机不支持该功能，对于软件使用者来说，写这些寄存器仍然是可能的。

Address offset: 0x04

Reset value: 0x0000 0000（不会被系统复位进行复位）

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

Bit	Name	R/W	Reset Value	Function
31：2	Reserved			
1	DBG_STOP	RW	0	Debug stop 模式。 0：（FCLK=off，HCLK=off）。在 STOP 模式，HCLK 和 FCLK 都会关闭。当从 STOP 模式退出时，时钟配置与上电复位后相同（系统时钟为 HSI）。随后，软件需要重新配置时钟控制器。

Bit	Name	R/W	Reset Value	Function
				1：（FCLK=on，HCLK=on）。当进入 STOP 模式，HSI 不会关闭，FCLK 和 HCLK 由 HSI 产生。当退出 STOP 模式，如果需要改变时钟控制，软件需要重新配置。
0	DBG_SLEEP	RW	0	调试睡眠模式。 0：（FCLK 开，HCLK 关）。在睡眠模式，FCLK 由原先配置好的系统时钟提供，HCLK 关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出后，软件不需要重新配置时钟。 1：（FCLK 开，HCLK 开）。在睡眠模式，FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。

31.9.3. DBG APB freeze register 1 (DBG_APB_FZ1)

该寄存器用来配置 timer、RTC、IWDG、WWDG 在 debug 下的时钟。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

Address offset: 0x08

Power on Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DBG_LPTIM_STOP	Res	Res	Res	Res	Res	Res	Res	Res	DBG_I2C2_SMBUS_TIMEOUT	DBG_I2C1_SMBUS_TIMEOUT	Res	Res	Res	Res	Res
RW									R	RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	Res	Res	Res	Res	DBG_TIM7_STOP	DBG_TIM6_STOP	Res	Res	DBG_TIM3_STOP	DBG_TIM2_STOP
			RW	RW	RW					rw				RW	RW

Bit	Name	R/W	Reset Value	Function
31	DBG_LPTIM_STOP	RW	0	当 CPU 停止时，LPTIM 的计数器时钟控制位 0：使能 1：不使能
30：23	Reserved			
22	DBG_I2C2_SMBUS_TIMEOUT	R	0	固定为 0
21	DBG_I2C1_SMBUS_TIMEOUT	RW	0	当 CPU 核处于 halt 状态时，控制 I2C1 SMBUS 超时是否 frozen。 0：与 normal 模式同样处理； 1：SMBUS 超时计数冻结。
20:19	reserved			
12	DBG_IWDG_STOP	RW	0	当 CPU 停止时，IWDG 计数器的时钟控制位 0：使能 1：不使能
11	DBG_WWDG_STOP	RW	0	当 CPU 停止时，WWDG 计数器的时钟控制位 0：使能 1：不使能

Bit	Name	R/W	Reset Value	Function
10	DBG_RTC_STOP	RW	0	当 CPU 停止时，RTC 计数器的时钟控制位 0：使能 1：不使能
9：6	Reserved			
5	DBG_TIM7_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM7 的计数时钟。 0：时钟使能； 1：时钟关闭；
4	DBG_TIM6_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM6 的计数时钟。 0：时钟使能； 1：时钟关闭；
3:2	reserved			
1	DBG_TIM3_STOP	RW	0	当 CPU 停止时，TIM3 计数器的时钟控制位 0：使能 1：不使能
0	DBG_TIM2_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM2 的计数时钟。 0：时钟使能； 1：时钟关闭；

31.9.4. DBG APB freeze register 2(DBG_APB_FZ2)

该寄存器用来配置 timer 在 debug 下的时钟控制。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

Address offset: 0x0C

Power on Reset value: 0x0000 0000

仅支持 32-bit 地址访问，只读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP
													RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DBG_TIM14_STOP	Res	Res	Res	DBG_TIM1_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
RW				RW											

Bit	Name	R/W	Reset Value	Function
31：19	Reserved			
18	DBG_TIM17_STOP	RW	0	当 CPU 停止时，TIM17 计数器的时钟控制位 0：使能 1：不使能
17	DBG_TIM16_STOP	RW	0	当 CPU 停止时，TIM16 计数器的时钟控制位 0：使能 1：不使能
16	DBG_TIM15_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM15 的计数时钟。 0：时钟使能； 1：时钟关闭；
15	DBG_TIM14_STOP	RW	0	当 CPU 停止时，TIM14 计数器的时钟控制位 0：使能

Bit	Name	R/W	Reset Value	Function
				1：不使能
14：12	Reserved			
11	DBG_TIM1_STOP	RW	0	当 CPU 停止时，TIM1 计数器的时钟控制位 0：使能 1：不使能
10：0	Reserved			

31.9.5. DBG 寄存器映像

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	DBG_IDCODE	REV_ID																															
	Reset value	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	DBG_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STO	DBG_SLE
	Reset value																															0	0
0x08	DBG_APB_FZ1	DBG_LPTIM_STO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_I2C2_SMBU	DBG_I2C1_SMBU	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_IWDG_STO	DBG_WWDG_ST	DBG_RTC_STOP	Res.	Res.	Res.	Res.	DBG_TIM7_STOP	Res.	Res.	Res.	DBG_TIM3_STOP	DBG_TIM2_STOP
	Reset value	0									0	0		0							0	0	0					0				0	0
0x0C	DBG_APB_FZ2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIM17_ST	DBG_TIM16_ST	DBG_TIM1_STO	DBG_TIM14_ST	Res.	Res.	Res.	DBG_TIM1_STO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value														0	0	0	0				0											

32.版本历史

版本	日期	更新记录
V0.1	2022.09.27	初版
V0.2	2023.03.27	更新第 8 章
V0.3	2024.01.05	1. 更新 DAC 相关模块功能 2. 更新第 23 章功能描述
V0.4	2024.03.06	1. 更新 TIM15 框图 2. 更新 ADC SMPR 寄存器描述 3. 更新 15.5.5 小节标题

Puya Semiconductor Co., Ltd.

声 明

普冉半导体(上海)股份有限公司（以下简称：“Puya”）保留更改、纠正、增强、修改 Puya 产品和/或本文档的权利，恕不另行通知。用户可在下单前获取产品的最新相关信息。

Puya 产品是依据订单时的销售条款和条件进行销售的。

用户对 Puya 产品的选择和使用承担全责，同时若用于其自己或指定第三方产品上的，Puya 不提供服务支持且不对此类产品承担任何责任。

Puya 在此不授予任何知识产权的明示或暗示方式许可。

Puya 产品的转售，若其条款与此处规定不一致，Puya 对此类产品的任何保修承诺无效。

任何带有 Puya 或 Puya 标识的图形或字样是普冉的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代并替换先前版本中的信息。

普冉半导体(上海)股份有限公司 - 保留所有权利