Technology Innovator
Puya

MS32C001B 系列参考手册

32 位 ARM®Cortex®-M0+微控制器

Puya Semiconductor (Shanghai) Co., Ltd.

目录

1.文档约定

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

2. 系统框图

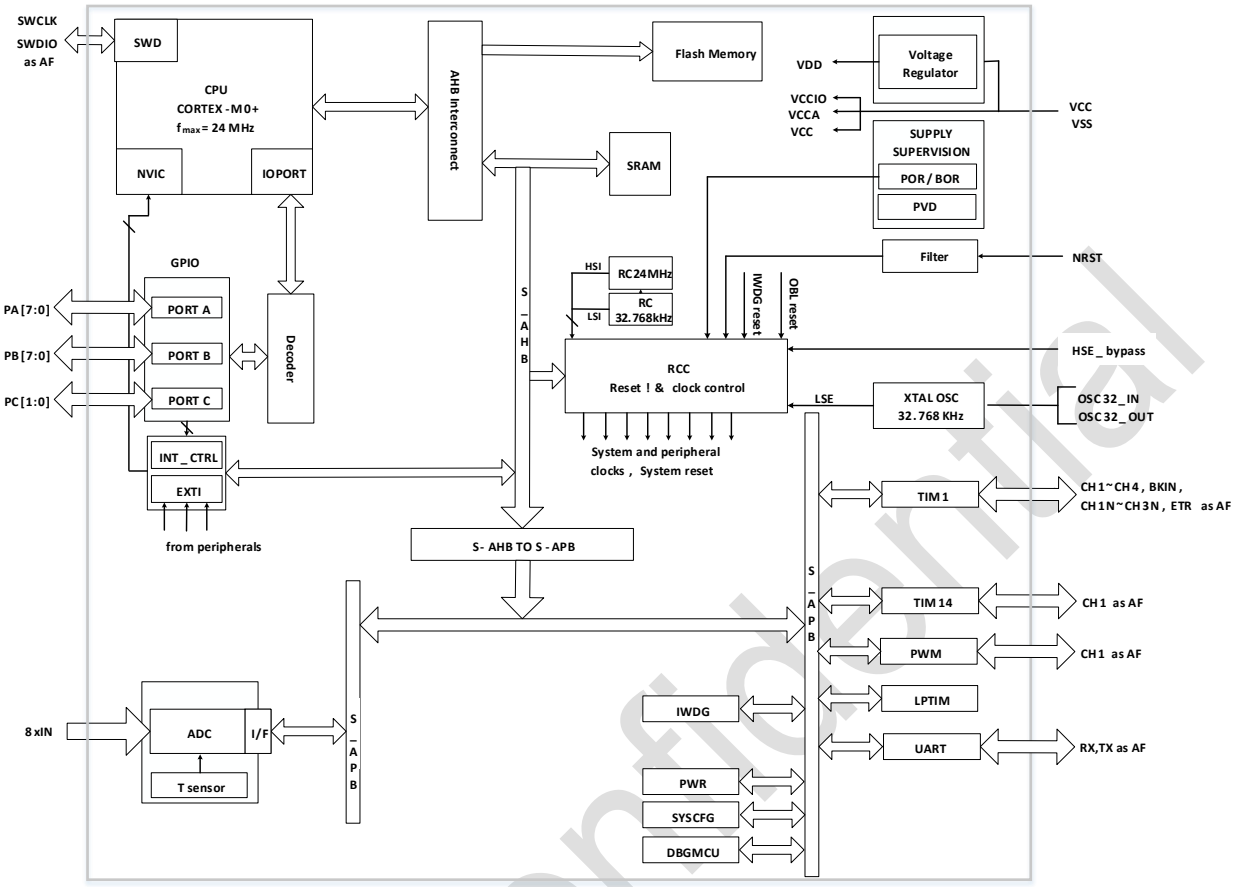


图 2-1 系统架构框图

3. 存储器和总线架构

3.1. 系统架构

系统由以下部分组成:

- 一个 Master
  - Cortex-M0+
- 三个 Slave
  - 内部 SRAM
  - 内部 Flash
  - 带 AHB-APB 总线桥的 AHB

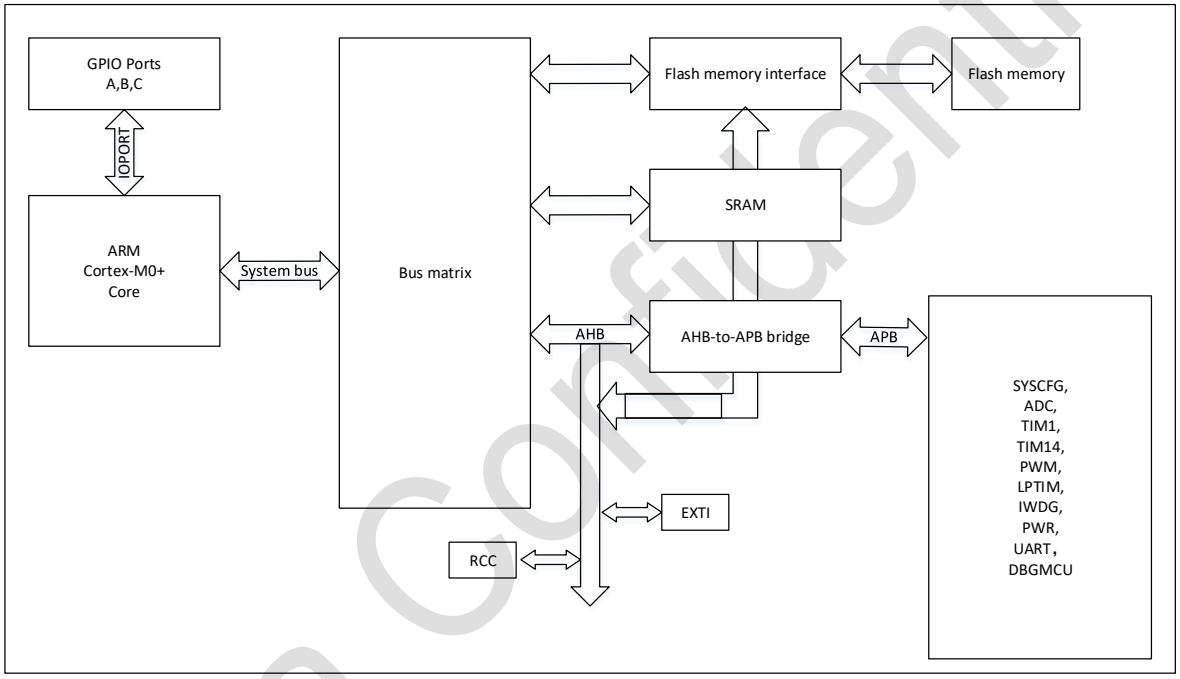


图 3-1 系统架构

- 系统总线

  该总线把 Cortex-M0+的系统总线总线矩阵（Bus matrix）。

- 总线矩阵

  总线矩阵由 Master（CPU）和 slaves（Flash memory、SRAM 和 AHB-to-APB bridge）组成。

- AHB-to-APB 总线桥

  AHB-to-APB 总线桥 提供了在 AHB 和 APB 总线之间的同步及连接到该 Bridge 的外设地址映射。

3.2. 存储器结构

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

整个寻址空间被划分成 8 个 512 MB 的 Block 区域。

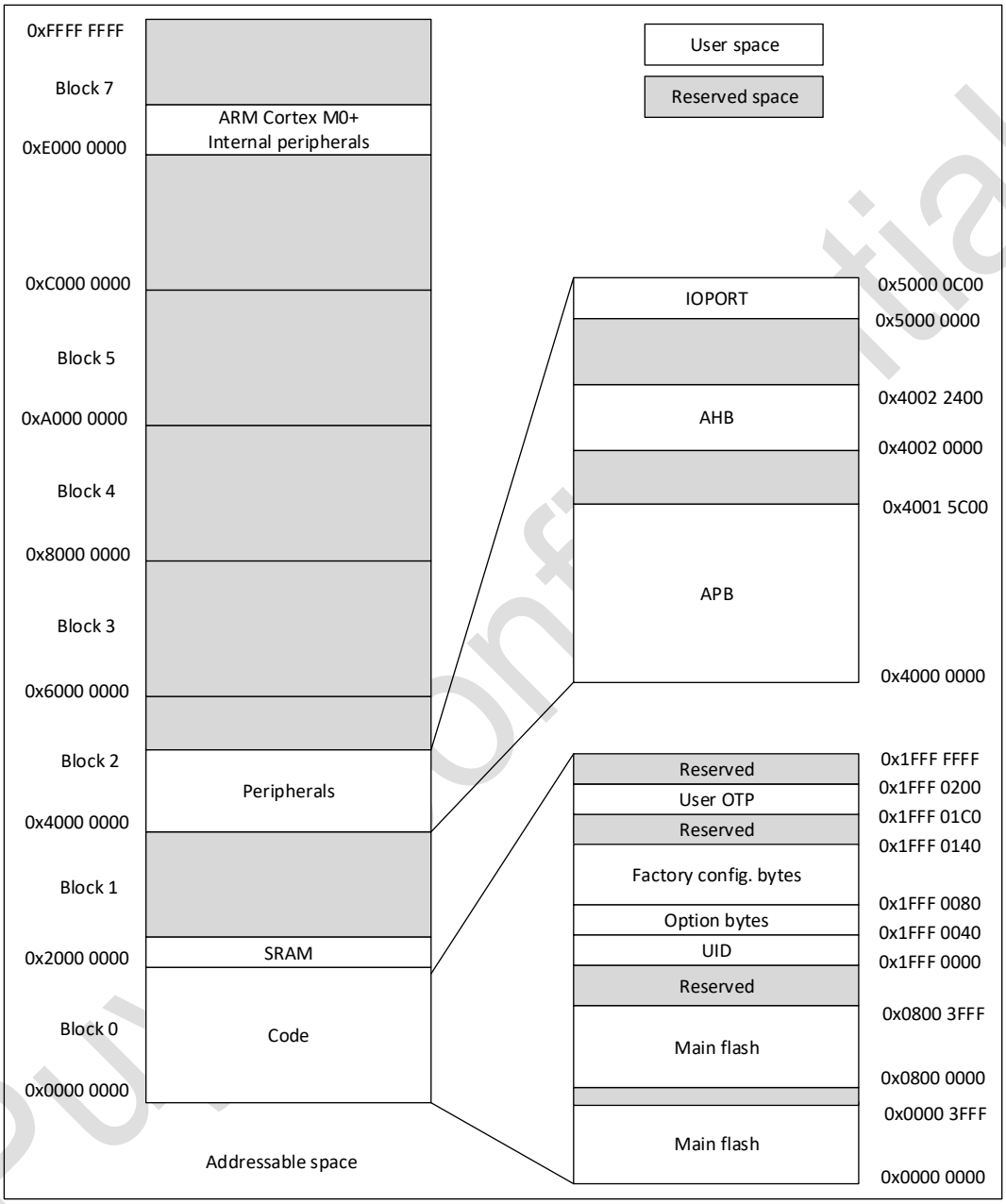


图 3-2 存储器映射

表 3-1 存储器地址

Type	Boundary Address	Size	Memory Area	Description
SRAM	0x2000 0800-0x3FFF FFFF	-	保留	-
	0x2000 0000-0x2000 07FF	2 KB	SRAM	-
Code	0x1FFF 0200-0x1FFF FFFF	-	保留	-
	0x1FFF 01C0-0x1FFF 01FF	64 bytes	USER OTP memory	存放用户数据
	0x1FFF 0100-0x1FFF 013F	64 bytes	Factory Config 2 bytes	存放 trimming 数据(含 HSI trimming 数据)、上电读校验码
	0x1FFF 00C0-0x1FFF 00FF	64 bytes	Factory Config 1 bytes	存放 trimming 数据(含 HSI trimming 数据)、上电读校验码
	0x1FFF 0080-0x1FFF 00BF	64 bytes	Factory Config 0 bytes	存放用户用到的 HSI 和 LSI trimming 数据、Flash 擦写时间配置参数
	0x1FFF 0040-0x1FFF 007F	64 bytes	Option bytes	芯片软硬件选项字节信息
	0x1FFF 0000-0x1FFF 003F	64 bytes	UID	Unique ID
	0x0800 4000-0x1FFE FFFF	-	保留	-
	0x0800 0000-0x0800 3FFF	16 KB	Main flash memory	-
	0x0000 4000-0x07FF FFFF	-	保留	-
	0x0000 0000-0x0000 3FFF	16 KB	Main flash memory	-

注：标注为**保留**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

总线	边界地址	大小	外设
	0xE000 0000-0xE00F FFFF	-	M0+
IOPORT	0x5000 0C00-0x5FFF FFFF	-	保留
	0x5000 0800-0x5000 0BFF	1 KB	GPIOC
	0x5000 0400-0x5000 07FF	1 KB	GPIOB
	0x5000 0000-0x5000 03FF	1 KB	GPIOA
AHB	0x4002 2400-0x4FFF FFFF	-	保留
	0x4002 2000-0x4002 23FF	1 KB	Flash
	0x4002 1C00-0x4002 1FFF		保留
	0x4002 1900-0x4002 1BFF	1 KB	保留
	0x4002 1800-0x4002 18FF		EXTI
	0x4002 1400-0x4002 17FF	-	保留
	0x4002 1080-0x4002 13FF	1 KB	保留
	0x4002 1000-0x4002 107F		RCC
	0x4002 0000-0x4002 0FFF	-	保留

总线	边界地址	大小	外设
	0x4002 3400-0x4FFF FFFF	-	保留
APB	0x4001 5C00-0x4001 FFFF	-	保留
	0x4001 5800-0x4001 5BFF	1 KB	DBG
	0x4001 3C00-0x4001 57FF	-	保留
	0x4001 3000-0x4001 3BFF	-	保留
	0x4001 2C00-0x4001 2FFF	1 KB	TIM1
	0x4001 2800-0x4001 2BFF	-	保留
	0x4001 2400-0x4001 27FF	1 KB	ADC
	0x4001 0400-0x4001 23FF	-	保留
	0x4001 0300-0x4001 03FF	1 KB	保留
	0x4001 0200-0x4001 02FF		保留
	0x4001 0100-0x4001 01FF		V_{REFBUF}
	0x4001 0000-0x4001 00FF		SYSCFG
	0x4000 8000-0x4000 FFFF	-	保留
	0x4000 7C00-0x4000 7FFF	1 KB	LPTIM
	0x4000 7400-0x4000 7BFF	-	保留
	0x4000 7000-0x4000 73FF	1 KB	PWR
	0x4000 4C00-0x4000 6FFF	-	保留
	0x4000 4800-0x4000 4BFF	1 KB	UART
	0x4000 3400-0x4000 47FF	-	保留
	0x4000 3000-0x4000 33FF	1 KB	IWDG
	0x4000 2C00-0x4000 2FFF	-	保留
	0x4000 2800-0x4000 2BFF	1 KB	PWM
	0x4000 2000-0x4000 27FF	-	保留
	0x4000 2000-0x4000 23FF	1 KB	TIM14
	0x4000 0000-0x4000 1FFF	-	保留

3.3. 嵌入式 SRAM

片内最大集成 2 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 hardfault。

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，16 KB，它包含应用程序和用户数据。用于存储用户程序和用户数据。
- Information 区域，0.5 KB，它包括以下部分：
  - Factory config. bytes 0：64 bytes，用于存放：
    - HSI 频率选择控制值，及对应的 Trimming 值
    - Flash 擦写时间配置参数值

  - LSI 32.768 kHz 的 Trimming 值
- Factory config. bytes 1 和 2：共计 128 bytes，用于存放：
  - 上电读校验码
  - 芯片硬件 Trimming 配置值
- UID：64 bytes，用于存放芯片的 UID
- Option byte：64 bytes，用于存放芯片硬件和存储保护的配置值
- User OTP Memory：64 bytes，用于存放用户数据

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的基本写/擦等操作。

4. 嵌入式 Flash 接口（FMC）

4.1. Flash 主要特征

- 主储存区：最大 16 KB（4K x 32 bits)
- 信息区：0.5 KB（128 x 32 bits）
- 页（Page）大小：64 bytes
- 扇区（Sector）大小：1 KB

闪存控制接口电路的主要特征如下：

- 闪存写（program）和擦除（erase）
- 写保护
- 读保护

4.2. Flash 功能介绍

4.2.1. 闪存结构

Flash 存储器由 32 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 64 bytes，Sector 大小为 1 KB。

从功能上，Flash 存储器分为 Main flash 和 Information flash，Main flash 容量最大是 16 KB，Information flash 容量为 0.5 KB。

Page erase 操作可以应用于 Main flash。

如果没有设置写保护，则全擦（Mass erase）可应用于 Main flash，否则不能应用于 Main flash。

表 4-1 闪存结构及边界地址

Block	sector	Page	Base address	Size
Main flash	Sector 0	Page 0-15	0x0800 0000 - 0x0800 03FF	1 KB
	Sector 1	Page 16-31	0x0800 0400 - 0x0800 07FF	1 KB
	Sector 2	Page 32-47	0x0800 0800 - 0x0800 0BFF	1 KB
	…	…	…	…
	Sector 13	Page 208-223	0x0800 3400 - 0x0800 37FF	1 KB
	Sector 14	Page 224-239	0x0800 3800 - 0x0800 3BFF	1 KB
	Sector 15	Page 240-255	0x0800 3C00 - 0x0800 3FFF	1 KB
UID	-	Page 0	0x1FFF 0000-0x1FFF 003F	64 bytes
选项字节		Page 1	0x1FFF 0040-0x1FFF 007F	64 bytes
Factory config 0		Page 2	0x1FFF 0080-0x1FFF 00BF	64 bytes
Factory config 1		Page 3	0x1FFF 00C0-0x1FFF 00FF	64 bytes
Factory config 2		Page 4	0x1FFF 0100-0x1FFF 013F	64 bytes
保留		Page 5	0x1FFF 0140-0x1FFF 017F	64 bytes
保留		Page 6	0x1FFF 0180-0x1FFF 01BF	64 bytes
USER OTP memory		Page 7	0x1FFF 01C0-0x1FFF 01FF	64 bytes

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。

取址和数据访问都是通过 AHB 总线进行的。

4.2.3. 闪存写操作和擦除操作

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

ICP：用来更新整个 Flash 存储器的内容，可以使用 SWD 协议，把用户应用程序装入 MCU 中。ICP 提供了快速和有效的设计迭代。

IAP：可以使用芯片支持的通讯接口，下载要写的数据到 Flash 中。IAP 允许用户在应用程序运行时，再次写 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占用总线。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。

注意：在 Flash 擦写期间，不能在 SRAM 中运行程序，否则会导致 CPU 停止工作

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（除了用作重新加载选项字节的 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，产生解锁时序，启用 FLASH_CR 寄存器的访问。

具体步骤如下：

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

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

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

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

另外，当 FLASH_SR 寄存器的 BSY 位被置位时，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 总线的占用，直到 BSY 位被清零。

4.2.3.2. 闪存写操作

Flash 存储器每次以 word（32 位）为单位进行整个页（page）的写操作。

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

当 FLASH_CR 寄存器的 PG 位被置位，CPU 向 FLASH 存储器地址空间写 32 位数据时，写操作开始启动。任何非 32 位的写入将导致 HardFault 中断。

如果要写的 flash 地址空间，是被 FLASH_WRPR 寄存器设置为保护的区域，则写操作会被忽略掉，同时 FLASH_CR 寄存器 WRPRTERR 位会被置位。写操作结束时，FLASH_CR 寄存器的 EOP 位会被置位。

具体 Flash 的写操作步骤如下所示：

1. 检查 FLASH_SR 寄存器的 BSY 位，判断是否当前没有正在继续的 Flash 操作
2. 如果没有正在进行的 Flash 擦或者写操作，则软件读出该页（Page）的 16 个字（如果该页已有数据存放，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（如果需要产生 EOP 中断）位
5. 向目标地址进行第 1 到第 15 个字的写操作（只接受 32 位的写操作）
6. 置位 FLASH_CR 寄存器的 PGSTRT
7. 写第 16 个字
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（当写操作已经成功，该位被置位），然后软件清零该位
10. 如果不再有写操作，则软件清除 PG 位
11. 当上述步骤 7）成功执行，则写操作自动启动，同时 BSY 位被硬件置位。
12. 闪存擦除操作

Flash 存储器可以按照 page 进行擦操作，或者进行扇擦（sector erase）和全擦（mass erase）（扇擦和全擦对 information memory 不起作用）。

4.2.3.3. 页擦（Page erase）

当某个页（page）被写保护，它是不会被擦的，此时 WRPERR 位被置位。当要进行页擦（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（如果需要产生 EOP 中断）位
4. 向该 Page 写任意数据（必须 32 位数据）
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. 闪存片擦（Mass erase）

片擦（Mass erase）用来对整片 Main flash 进行擦操作，但对 Information 区不起作用。另外，当 WRP 被使能，片擦功能无效，不会产生片擦操作，并且 WEPERR 位被置位。

进行片擦的步骤如下：

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（如果需要产生 EOP 中断）位
4. 向 Flash 的任意 Main flash 空间写任意数据（32 位数据）
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. 扇擦（Sector erase）

扇擦用来对 1 KB 的 Flash 主储存区进行擦除操作，但对信息区不起作用。另外，当某个扇区被 WRP 保护，它是不会被擦除的，此时 WRPERR 位被置位。

进行扇擦的步骤如下：

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

除了用户一次编程 memory 以外的 Information memory 是只读的，永远不会被 program/erase。

4.2.3.6. 写和擦除时间配置

Flash 的 program 和 erase 的时间需要进行严格的控制，否则会造成操作失败。软件需要从相应 INFO 区地址读出数据，再写入对应的寄存器，以实现所需的擦写时间的配置。

表 4-2 Program 和 erase 时间配置

寄存器	相应 INFO 区地址
FLASH_TS0	0x1FFF 009C
FLASH_TS1	0x1FFF 009C
FLASH_TS2P	0x1FFF 00A0
FLASH_TPS3	0x1FFF 00A0
FLASH_TS3	0x1FFF 009C
FLASH_ERTPE	0x1FFF 00A4
FLASH_PRGTPE	0x1FFF 00A8
FLASH_PRETPE	0x1FFF 00A8

4.3. 产品唯一身份标识码（UID）

唯一身份标识码典型应用场景：

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

产品唯一身份标识提供了一个对于任何设备都唯一的参考号码。

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

4.4. Flash 选项字节

4.4.1. Flash 选项字节描述

芯片内 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，选项字节以正码及反码形式分别存储。

表 4-3 选项字节格式

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

选项字节的内容可以从下表选项字节结构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 用户选项寄存器（FLASH_OPTR）
- Flash SDK 区域地址寄存器（FLASH_SDKR）
- Flash WRP 地址寄存器（FLASH_WRPR）

表 4-4 选项字节结构

Word Address	Description
0x1FFF 0040	用户选项字节及反码
0x1FFF 0044	SDK 区域地址选项字节及反码
0x1FFF 0048	保留
0x1FFF 004C	WRP 地址选项字节及反码
0x1FFF 0050	保留
0x1FFF 0054	保留
…	保留
…	保留
…	保留
0x1FFF 007C	保留

- **Flash 用户选项的选项字节**

Flash 地址：0x1FFF 0040

生产值：0x6F55 90AA

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 flash information memory 的选型字节区域读出相应的值，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	~NRST_ MODE	Res.	~IWDG _SW	~BOR_LEV[2:0]			~BOR_ EN	~RDP							
R	R		R	R			R	R							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IWDG_STOP	NRST_	Res.	IWDG	BOR_LEV[2:0]			BOR_	RDP							

	MODE		_SW		EN	
R	R		R	R	R	R

Bit	Name	R/W	Function
31	~ IWDG_STOP	R	IWDG_STOP 的反码
30	~NRST_MODE	R	NRST_MODE 的反码
29	Reserved	-	保留
28	~IWDG_SW	R	IWDG_SW 的反码
27：25	~BOR_LEV[2:0]	R	BOR_LEV 的反码
24	~BOR_EN	R	BOR_EN 的反码
23：16	~RDP[7:0]	R	RDP 的反码
15	IWDG_STOP	R	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
14	NRST_MODE	R	0：仅复位输入 1：GPIO 功能
13	Reserved	-	保留
12	IWDG_SW	R	0：硬件看门狗 1：软件看门狗
11：9	BOR_LEV[2:0]	R	000：BOR 上升阈值为 2.2V，下降阈值位 2.1V 001：BOR 上升阈值为 2.4V，下降阈值位 2.3V 010：BOR 上升阈值为 2.6V，下降阈值位 2.5V 011：BOR 上升阈值为 2.8V，下降阈值位 2.7V 100：BOR 上升阈值为 3.0V，下降阈值位 2.9V 101：BOR 上升阈值为 3.2V，下降阈值位 3.1V 110：BOR 上升阈值为 3.4V，下降阈值位 3.3V 111：BOR 上升阈值为 3.6V，下降阈值位 3.5V
8	BOR_EN	R	BOR 使能 0：BOR 不使能 1：BOR 使能，BOR_LEV 起作用
7：0	RDP[7:0]	R	0xAA：level 0，读保护无效 非 0xAA：level 1，读保护有效

■ Flash SDK 区域地址的选项字节

Flash 地址：0x1FFF 0044

生产值： 0xFFF0 000F

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，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	~SDK_END[3:0]				Res	Res	Res	Res	~SDK_STRT[3:0]			

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

Bit	Name	R/W	Function
31：28	Reserved	-	保留
27：24	~SDK_END[3:0]	R	SDK_END 的反码
23：20	Reserved	-	保留
19：16	~SDK_STRT[3:0]	R	SDK_STRT 的反码
15：12	Reserved	-	保留
11：8	SDK_END[3:0]	R	SDK 区域结束地址，每一位对应的 STEP 为 1 KB
7：4	Reserved	-	保留
3：0	SDK_STRT[3:0]	R	SDK 区域开始地址，每一位对应的 STEP 为 1 KB

■ **Flash 写保护地址**

Flash 地址：0x1FFF 004C

生产值：0x0000 FFFF

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，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	~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

4.4.2. 写 Flash 选项字节

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. 通过解锁时序，解锁 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 HardFault 中断。

User option（用户选项）（Information flash 的选项字节）可以通过软件写 FLASH_CR 寄存器的 OPTLOCK 位保护，以防止不想要的擦或者写操作。

如果软件置位 Lock 位，则 OPTLOCK 位也被自动置位。

修改用户的选项字节

选项字节的写操作，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：

1. 用之前描述的步骤，清零 OPTLOCK 位
2. 检查 BSY 位，确认没有正在进行的 Flash 操作
3. 向选项字节寄存器 FLASH_OPTR/FLASH_SDKR/FLASH_WRP 写期望的值（1~3 个字）
4. 置位 OPTSTRT 位
5. 向 Main flash 0x4002 2080 地址写任意 32 位数据（触发正式的写操作）
6. 等待 BSY 位被清零
7. 等待 EOP 拉高，软件清零

任何对选项字节的改动，硬件都会先把选项字节对应的整个 page 擦掉，然后用 FLASH_OPTR、FLASH_SDKR 或者 FLASH_WRPR 寄存器的值，写到选项字节中。硬件自动计算相应的反码，并把计算值写到选项字节的相应区域。

重新加载选项字节

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

选项字节的装载，在以下两种情况下进行：

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

“装载选项字节”进行的操作是：对 information memory 区域的选项字节进行读操作，再把读出的数据存储在内部选项寄存器中（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，这样选项字节的装载，才能在系统的复位下进行。

每个选项位在它相同的双字地址（下一个半字）有相应的反码。在选项字节装载期间，会对选项位和其反码进行验证，这能确保装载被正确的进行了。

如果正反码匹配，则选项字节被复制到选项寄存器中。

如果正反码不匹配，则 FLASH_SR 寄存器的 OPTVERR 状态位被置位。option 寄存器维持默认值：

- 对于用户选项
  - 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 地址选项，SDKR_STRT[3:0]= 0x0，SDKR_END[3:0]=0xF，即所有 Flash 空间都被设定为 SDK
- 对于 FLASH boot 启动选择
  - BOOT0 位写成 0（即选择 Main flash 作为启动区）
- 对于 WRP option，不匹配的值是默认值“无保护”

在系统复位后，选项字节的内容被复制到下面的选项寄存器（软件可读可写）：

- FLASH_OPTR
- FLASH_SDKR
- FLASH_WRPR

这些寄存器也被用来修改选项字节。如果这些寄存器不被用户修改，他们体现了系统选项的状态。

4.5. Flash 配置字节

芯片内的 Flash 的 information 区域的部分区间（共 3 个 page）作为 Factory config. byte 使用。

Page 2 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
- Flash 擦写时间配置参数值
- LSI 32.768 KHz 的 Trimming 值

Page 3 和 4 存放芯片硬件出厂信息（正反码存放）：

- 芯片上电读校验码
- 芯片硬件 Trimming 配置值

为了数据的安全性，Page 3 和 4 的 Factory config. byte 以正文及反码形式分别存储。

表 4-5 Factory config. byte organization

Page	Word	Address	Contents
0	0-3	0x1FFF 0000-0x1FFF 000F	UID
	4-15	0x1FFF 0010-0x1FFF 003F	保留
2	0	0x1FFF 0080	存放 HSI 8 MHz 频率选择控制及对应的 Trimming 值
	1	0x1FFF 0084	保留
	2	0x1FFF 0088	存放 HSI 24 MHz 频率选择控制及对应的 Trimming 值
	3	0x1FFF 008C	保留
	4	0x1FFF 0090	存放 LSI 32.768 kHz 的 Trimming 值
	5	0x1FFF 0094	常温 ts data
	6	0x1FFF 0098	高温 ts data
	7	0x1FFF 009C	存放 FLASH_TS0、FLASH_TS1、FLASH_TS3 寄存器的配置值
	8	0x1FFF 00A0	存放 FLASH_TS2P、FLASH_TPS3 寄存器的配置值
	9	0x1FFF 00A4	存放 FLASH_ERTPE 寄存器的配置值
	10	0x1FFF 00A8	存放 FLASH_PRGTPE、FLASH_PRETPE 寄存器的配置值
	11	0x1FFF 00AC	保留
	12	0x1FFF 00B0	保留

Page	Word	Address	Contents
	13	0x1FFF 00B4	保留
	14	0x1FFF 00B8	保留
	15	0x1FFF 00BC	保留
3	0	0x1FFF 00C0	上电读校验码 0x55AA AA55
	1	0x1FFF 00C4	上电读校验码 0xAA55 55AA
	2	0x1FFF 00C8	上电读校验码 0x55AA AA55
	3	0x1FFF 00CC	上电读校验码 0xAA55 55AA
	4	0x1FFF 00D0	PMU trimming bit 及反码
	5	0x1FFF 00D4	PMU trimming bit 及反码
	6	0x1FFF 00D8	PMU trimming bit 及反码
	7	0x1FFF 00DC	PMU trimming bit 及反码
	8	0x1FFF 00E0	保留
	9	0x1FFF 00E4	保留
	10	0x1FFF 00E8	CLK trimming bit 及反码
	11	0x1FFF 00EC	CLK trimming bit 及反码
	12	0x1FFF 00F0	CLK trimming bit 及反码
	13	0x1FFF 00F4	保留
	14	0x1FFF 00F8	Flash trimming 及反码
	15	0x1FFF 00FC	Flash trimming 及反码
4	0	0x1FFF 0100	Flash trimming 及反码
	1	0x1FFF 0104	Flash trimming 及反码
	2	0x1FFF 0108	Flash trimming 及反码
	3	0x1FFF 010C	Flash trimming 及反码
	4	0x1FFF 0110	ANA IP trimming 及反码
	5	0x1FFF 0114	ANA MUX trimming 及反码
	6	0x1FFF 0118	ANA MUX trimming 及反码
	7	0x1FFF 011C	ANA MUX trimming 及反码
	8	0x1FFF 0120	ANA MUX trimming 及反码
	9	0x1FFF 0124	ANA MUX trimming 及反码
	10	0x1FFF 0128	ANA MUX trimming 及反码
	11	0x1FFF 012C	保留
	12	0x1FFF 0130	保留
	13	0x1FFF 0134	保留
	14	0x1FFF 0138	Device ID code
	15	0x1FFF 013C	保留

4.5.1. HSI_TRIMMING_FOR_USER

Flash 地址： 0x1FFF 0080(8MHz)/ 0x1FFF 0088(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		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HSI_TC_TRIMCR[3:0]				HSI_ABS_TRIMCR[11:0]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件需要从该地址读出数据，再写入 RCC_ICSCR 寄存器对应的 HSI_TC_TRIMCR[3:0]和 HSI_ABS_TRIMCR[11:0]，以实现 HSI 频率的更改。

4.5.2. LSI_32.768K

Flash 地址： 0x1FFF 0090（32.768 kHz）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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	LSI_TRIM[8:0]								
							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 LSI_TRIM[8:0]，以实现 LSI 频率的更改

4.5.3. FLASH_EPPARA0

Flash 地址： 0x1FFF 009C

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

软件需要从相应地址读出数据，再写入 FLASH_TS0、FLASH_TS1、FLASH_TS3 寄存器，以实现所需的擦写时间的配置。

4.5.4. FLASH_EPPARA1

Flash 地址： 0x1FFF 00A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	TPS3[9:0]									
						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**
Res	Res	Res	Res	Res	Res	Res	Res	TS2P[7:0]							
								R	R	R	R	R	R	R	R

软件需要从相应地址读出数据，再写入 FLASH_TS2P、FLASH_TPS3 寄存器，以实现所需的擦写时间的配置。

4.5.5. FLASH_EPPARA2

Flash 地址： 0x1FFF 00A4

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

软件需要从相应地址读出数据，再写入 FLASH_ERTPE 寄存器中，以实现所需的擦写时间的配置。

4.5.6. FLASH_EPPARA3

Flash 地址： 0x1FFF 00A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	PRETPE[13:0]													
		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**
PRGTPE[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

软件需要从相应地址读出数据，再写入 FLASH_PRGTPE 和 FLASH_PRETPE 寄存器中，以实现所需的擦写时间的配置。

4.5.7. Flash USER OTP memory bytes

芯片内的 Flash 的 information 区域的部分区间作为 Flash USER OTP memory bytes。

表 4-6 USER OTP memory bytes organization

Page	Word	Address	Contents
7	0	00x1FFF 01C0	Bit[31:16]：存放用户数据 Bit[15:0]：USER OTP MEMORY_LOCK
	1	00x1FFF 01C4	存放用户数据
	2	00x1FFF 01C8	存放用户数据
	…	…	存放用户数据
	…	…	存放用户数据
	…	…	存放用户数据
	31	00x1FFF 01FC	存放用户数据

本 Page 配置在 information 区域，对本 Page 区域 program 和擦是按照 Main flash 的方法来处理。另外，Main flash 区域的 mass erase 对本区域无效。

设定 USER OTP MEMORY_LOCK 内容不会立刻更新，直到上电复位（POR/BOR/PDR），从会起到保护功能。 对本 Page Write 有如下保护。

表 4-7 Flash USER OTP memory bytes 的写保护状态

USER OTP MEMORY_LOCK	Write protection
0xAA55	读：可以 program 和擦操作：不可以
除(0xAA55)之外的任何值	读、program 和擦操作：可以

4.6. Flash 存储区保护

对 Flash 主储存区的保护包括以下几种机制：

- SDK（software design kit）的保护，用来对特定程序区的访问保护，大小是 1 KB。
- 写保护（WRP）控制，防止不想要的写操作（程序存储器指针的混乱）。写保护的大小设计为 1 KB。
- 选项字节写保护有专门的解锁设计。

4.6.1. 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[3:0]，SDKR_END[3:0]定义，每一个 bit 对应 1 KB。

Start address(起始地址)：

Flash memory base address + SDK_STRT[3:0] x 0x400（包含）

End address（结束地址）：

Flash memory base address + (SDK_END[3:0]+1) x 0x400（不包含）

当 SDK_STRT[3:0] 大于 SDK_END[3:0] 时，SDK 保护无效；当 SDK_STRT[3:0] 小于或等于 SDK_END[3:0]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[3:0]大于 SDK_END[3:0]），硬件会先触发全擦（mass erase）（SDK 区域被保护的程序之前已经写入，通过全擦起到了对 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。

4.6.2. 读闪存保护(RDP)

设置 RDP 选项字节，并进行系统复位（POR/BOR 或者 OBL 复位)装载新的 RDP 选项字节，可以激活读保护功能。RDP 保护 Main flash。

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 选项字节和补码成对正确存在于选项字节时，Main flash 会被保护。

表 4-8 Main flash 的 RDP 读保护状态

RDP 值	RDP 反码值	RDP 保护等级
0xAA	0x55	Level 0
除(0xAA 和 0x55)之外的任何值		Level 1

无论任何保护级别，Information memory 能都被访问，不能进行 program 和 erase 操作。

级别 0：无保护

对 Main flash 的读、program 和擦操作是可能的，对选项字节也是可以进行任何操作。

级别 1：读保护

当选项字节里的 RDP 及其反码包含任何（0xAA、0x55）之外的组合，则 level 1 read protection 生效，级别 1 是缺省的保护级别。

- Main boot：在用户模式下执行的程序（boot from Main flash），可以对 Main flash、选项字节进行所有操作。
- Debug，boot from SRAM：在 debug 模式，或者当从 SRAM 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 HardFault 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 4-9 访问状态与保护级别和执行模式的关系

区域	RDP 保护级别	SDK 区域保护等级	从 Main Flash(CPU)启动						调试/从 SRAM 启动		
			用户执行操作 (From Non SDK Area)			用户执行操作 (From SDK Area)					
			Read	Write	Erase	Read	Write	Erase	Read	Write	Erase
Main flash	0	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	1	-	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No
Non SDK Area	-	不使能	Yes	Yes	Yes	N/A	N/A	N/A	Yes	Yes	Yes
	-	使能	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SDK Area	-	不使能	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	-	使能	No	No	No	Yes	Yes	Yes	No	No	No
选项字节	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Factory bytes	-	-	Yes	No	No	Yes	No	No	Yes	No	No
UID	-	-	Yes	No	No	Yes	No	No	Yes	No	No

注：

1. 任何区域发出的全擦（mass erase）指令都会擦掉 SDK 区。
2. N/A 的含义是当 SDK 区域禁能掉，由于不存在 SDK 区域，上表 SDK 区域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 区域访问的情况。

4.6.3. 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 寄存器每位控制大小为 1 KB 的写保护（WRP）区域，即 1 个 sector 大小。

当被 WRP 的区域被激活，则不允许进行擦或者写操作。相应的，即使只有一个区域被设定为写保护，则全擦（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擦或者写操作，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。

4.6.4. 选项字节写保护

默认情况下，选项字节是可读，并进行写保护的。为获得对选项字节的擦或者写访问，需要向 OPTKEYR 寄存器写入正确的序列。

4.7. Flash 中断

表 4-10 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 HardFault：

- 解锁 Flash memory 的 FLASH_CR 寄存器的序列错误
- 解锁 Flash 选项字节的写操作序列错误
- 写 Flash 操作未进行 32 位数据的对齐
- 擦除 Flash（含页擦（page erase）、扇擦（sector erase）和全擦（mass erase））操作未进行 32 位数据对齐
- 对选项字节寄存器的写操作未进行 32 位数据的对齐

4.8. Flash 寄存器

4.8.1. Flash 密钥寄存器 (FLASH_KEYR)

偏移地址： 0x08

复位值： 0x0000 0000

所有寄存器位是 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	32'h0	下面的值必须被连续的写入，才能解锁 FLASH_CR 寄存器，并允许 Flash 的 program/erase 操作 KEY1：0x4567 0123 KEY2：0xCDEF 89AB

4.8.2. Flash 选项密钥寄存器 (FLASH_OPTKEYR)

偏移地址： 0x0C

复位值： 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	32'h0	下面的值必须被连续的写入，才能解锁 Flash 的 option 寄存器，并允许选项字节的 program/erase 操作 KEY1：0x0819 2A3B KEY2：0x4C5D 6E7F

4.8.3. Flash 状态寄存器 (FLASH_SR)

偏移地址： 0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	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	-	-	保留
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	选项和修剪位加载有效性错误 当 option 和 trimming bit 及其反码不匹配时，硬件置位该位。装载不匹配的选项字节，被强制成安全值。 软件写 1，清零。
14:5	Reserved	-	-	保留

Bit	Name	R/W	Reset Value	Function
4	WRPERR	RC_W1	0	写保护错误。 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。 写 1，清零该位。
3:1	Reserved	-	-	保留
0	EOP	RC_W1	0	当 Flash 的 program/erase 操作成功完成，硬件置位。 如果 FLASH_CR 寄存器的 EOPIE 位使能后，产生中断。 写 1，清零该位。

4.8.4. Flash 控制寄存器(FLASH_CR)

偏移地址： 0x14

复位值： 0xC000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPT LOCK	Res	Res	OBL_LAUNCH	Res	ERR IE	EOP IE	Res	Res	Res	Res	PGSTRT	Res	OPT STRT	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 寄存器锁定位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器被锁定住。当成功给出解锁时序后，该位被硬件清零解锁 FLASH_CR 寄存器。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的解锁时序给出，该位仍然保持置位状态，直到下一次系统复位。
30	OPTLOCK	RS	1	选项字节 Lock 位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器中与选项字节有关的位被锁定住。当成功给出解锁时序后，该位被硬件清零，解锁了 FLASH_CR 寄存器。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的解锁时序给出，该位仍然保持置位状态，直到下一次系统复位。
29:28	Reserved	-	-	保留
27	OBL_LAUNCH	RC_W1	0	强制选项字节加载。

Bit	Name	R/W	Reset Value	Function
				当置位时，该位强制系统进行选项字节的重装载。该位仅当选项字节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：选项字节加载完成 1：产生选项字节加载请求，系统产生复位，进行选项字节的重装载。
26	Reserved	-	-	保留
25	ERRIE	RW	0	错误中断使能位，当 FLASH_SR 寄存器的 WRPERR 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
24	EOPIE	RW	0	操作结束中断使能位 当 FLASH_SR 寄存器的 EOP 位被置位，如果该位使能，则产生中断请求。 0：EOP 中断关闭 1：EOP 中断使能
23:20	Reserved	-	-	保留
19	PGSTRT	RW	0	Flash 主储存区的编程操作的启动位。 该位启动了 Flash 主储存区的编程操作，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	-	-	保留
17	OPTSTRT	RW	0	Flash 选项字节修改的启动位 该位启动了对选项字节的修改。软件置位，在 FLASH_SR 寄存器的 BSY 位被清零后，硬件清零该位。 注意：当对 Flash 选项字节进行修改时，硬件自动把整个 128 bytes 的页进行擦除操作，再进行编程操作，其中也包括自动进行反码的写入。
16:12	Reserved	-	-	保留
11	SER	RW	0	4 KB 的扇区擦除操作 0：未选择 Flash 的扇区擦除操作 1：选择 Flash 的扇区擦除操作 注： 1. 扇区擦除不会对 Flash 信息区起作用。 2. 扇区擦除对设定为 WRP 的区域不起作用。
10:3	Reserved	-	-	保留
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：

Bit	Name	R/W	Reset Value	Function
				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	页擦除操作 0：未选择 Flash 的页擦除操作 1：选择 Flash 的页擦除操作
0	PG	RW	0	Program 操作 0：未选择 Flash 的 program 操作 1：选择 Flash 的 program 操作

4.8.5. Flash 选项寄存器 (FLASH_OPTR)

偏移地址： 0x20

复位值： 0x0000 90AA。

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash 信息区的选项字节区域读出相应的值，写入到该寄存器相应的 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	NRST_MODE	Res	IWDG_SW	BOR_LEV[2:0]			BOR_EN	RDP[7:0]							
RW	RW	-	RW	RW	RW	RW	RW	RW							

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	IWDG_STOP	RW	1	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
14	NRST_MODE	RW	0	0：仅复位输入 1：GPIO 功能
13	Reserved	-	-	保留
12	IWDG_SW	RW	1	0：硬件看门狗 1：软件看门狗
11：9	BOR_LEV[2:0]	RW	3'h0	000：BOR 上升阈值为 2.2V，下降阈值位 2.1V 001：BOR 上升阈值为 2.4V，下降阈值位 2.3V 010：BOR 上升阈值为 2.6V，下降阈值位 2.5V 011：BOR 上升阈值为 2.8V，下降阈值位 2.7V 100：BOR 上升阈值为 3.0V，下降阈值位 2.9V 101：BOR 上升阈值为 3.2V，下降阈值位 3.1V

Bit	Name	R/W	Reset Value	Function
				110：BOR 上升阈值为 3.4V，下降阈值位 3.3V 111：BOR 上升阈值为 3.6V，下降阈值位 3.5V
8	BOR_EN	RW	0	BOR 使能 0：BOR 不使能 1：BOR 使能，BOR_LEV 起作用
7：0	RDP[7:0]	RW	8’hAA	0xAA：level 0，读保护无效 非 0xAA：level 1，读保护有效

4.8.6. Flash SDK 地址寄存器 (FLASH_SDKR)

偏移地址：0x24

复位值：0x0000 000F

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，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**
Res	Res	Res	Res	SDK_END[3:0]				Res	Res	Res	Res	SDK_STRT[3:0]			
-	-	-	-	RW	RW	RW	RW	-	-	-	-	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：12	Reserved	-	-	保留
11：8	SDK_END[3:0]	RW	4’h0	SDK 区域结束地址，每一位对应的 STEP 为 1 KB
7：4	Reserved	-	-	保留
3：0	SDK_STRT[3:0]	RW	4’hF	SDK 区域开始地址，每一位对应的 STEP 为 1 KB

4.8.7. Flash WRP 地址寄存器 (FLASH_WRPR)

偏移地址： 0x2C

复位值： 0x0000 FFFF

在上电复位（POR/BOR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，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[7:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	WRP	RW	16'hFFFF	0：sector[y]被保护 1：sector[y]无保护 y=0~15

4.8.8. Flash TS0 寄存器 (FLASH_TS0)

偏移地址：0x100

复位值：0x0000 003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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	-	-	保留
7：0	TS0	RW	8'h3C	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 009C

4.8.9. Flash TS1 寄存器 (FLASH_TS1)

偏移地址：0x104

复位值：0x0000 0090

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	保留
7：0	TS1	RW	7'h90	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，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 009C

4.8.10. Flash TS2P 寄存器 (FLASH_TS2P)

偏移地址： 0x108

复位值： 0x0000 003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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	-	-	保留
7：0	TS2P	RW	8'h3C	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 00A0

4.8.11. Flash TPS3 寄存器 (FLASH_TPS3)

偏移地址：0x10C

复位值：0x0000 0240

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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	TPS3									
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	保留
9：0	TPS3	RW	10'h240	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 00A0

4.8.12. Flash TS3 寄存器 (FLASH_TS3)

偏移地址：0x110

复位值：0x0000 003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	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	-	-	保留
7：0	TS3	RW	8'h3C	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 009C

4.8.13. Flash 擦写（ERASE）TPE 寄存器（FLASH_ERTPE)

偏移地址： 0x114

复位值： 0x0000 6D60

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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
31：16	Reserved	-	-	保留
15：0	ERTPE	RW	16'h6D60	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 00A4

4.8.14. Flash PROGRAM TPE 寄存器（FLASH_PRGTPE)

偏移地址： 0x11C

复位值： 0x0000 1F40

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

Bit	Name	R/W	Reset Value	Function
15：0	PRGTPE	RW	16'h1F40	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，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 00A8

4.8.15. Flash PRE-PROGRAM TPE 寄存器 (FLASH_PRETPE)

偏移地址： 0x120

复位值： 0x0000 0640

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

Bit	Name	R/W	Reset Value	Function
31：14	Reserved	-	-	保留
13：0	PRETPE	RW	14'h640	软件通过读出存放在信息区相应地址的数据，写入对应寄存器，以实现所需的擦写时间的配置。 保存在 Flash 的如下地址内：0x1FFF 00A8

5. 电源控制（PWR）

5.1. PWR 电源

5.1.1. 电源框图

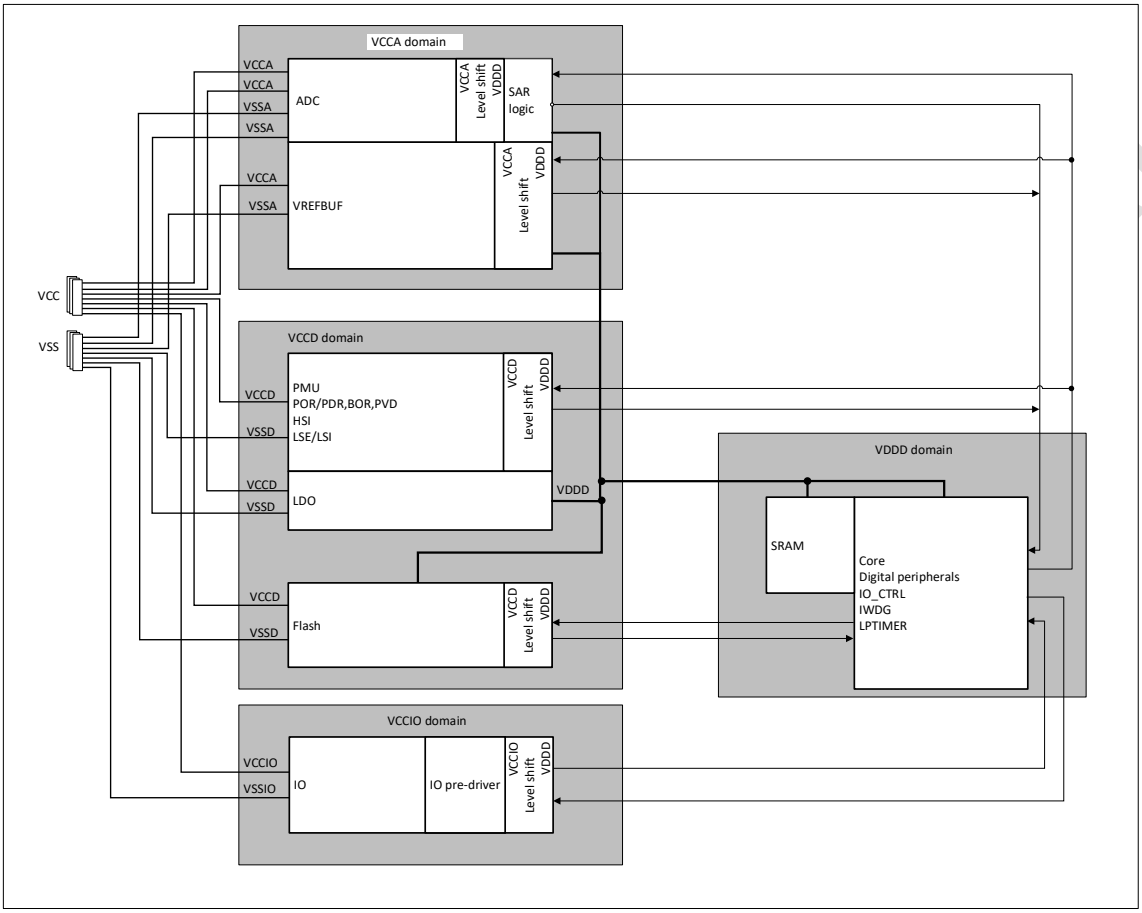


图 5-1 电源框图

表 5-1 电源框图

编号	电源	电源值	描述
1	V_{CC}	1.9 ~ 5.5 V	通过电源管脚为芯片提供电源，其供电模块为：部分模拟电路。
2	V_{CCA}	1.9 ~ 5.5 V	给大部分模拟模块供电（ADC，V_{REFBUF}），来自于 V_{CC} PAD
3	V_{CCD}	1.9 ~ 5.5 V	给大部分模拟模块供电（PMU，Flash 和时钟），来自于 V_{CC} PAD
4	V_{CCIO}	1.9 ~ 5.5 V	给 IO 供电，来自于 V_{CC} PAD
5	V_{DDD}	1.2 V	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 1.2V。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。

5.1.2. 电压调节器

芯片设计 3 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。
- DLPR（Deep low power regulator）Deep_stop 模式下，提供最低功耗的选择。

在芯片运行模式，MR 保持工作，输出 1.2 V 电压，LPR 关闭。

在 Stop 模式，可由软件决定从 MR 或 LPR 供电。在 Deep_stop 模式，由 DLPR 供电。

5.1.3. 动态电压值管理

本项目定义两种电压范围：

- 范围 1：高性能范围

MR 的输出为典型值 1.2 V，系统时钟频率可以运行在最快的 24 MHz 频率下。

- 范围 2：低功耗范围

只有当芯片处于 Stop/Deep_stop 模式时，才允许设定进入该范围。

5.2. PWR 电源监控

5.2.1. 上电复位（POR)/下电复位（PDR)/欠压复位（BOR)

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

除了 POR/PDR 外，还实现了 BOR（Brown out reset）。

当 BOR 被打开时，BOR 的阈值可以通过选项字节进行选择，上升和下降检测点都可以被单独配置，见图 5-2。

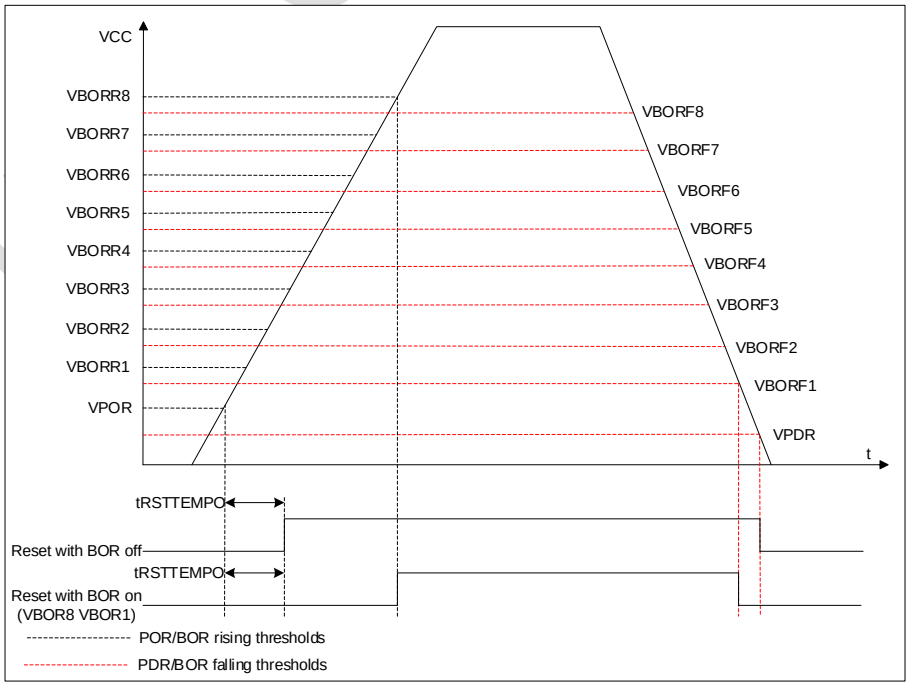


图 5-2 POR/PDR/BOR 阈值

5.2.2. 可编程电压检测器（PVD）

可编程电压检测器（PVD）模块可以用来检测 V_{CC} 电源，检测点可通过寄存器进行配置。当 V_{CC} 高于或低于 PVD 的检测点时，产生相应的复位标识。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 V_{CC} 上升超过 PVD 的检测点，或者 V_{CC} 降低到 PVD 的检测点以下，产生中断，在中断服务程序中用户可以进行紧急的 shutdown 任务。

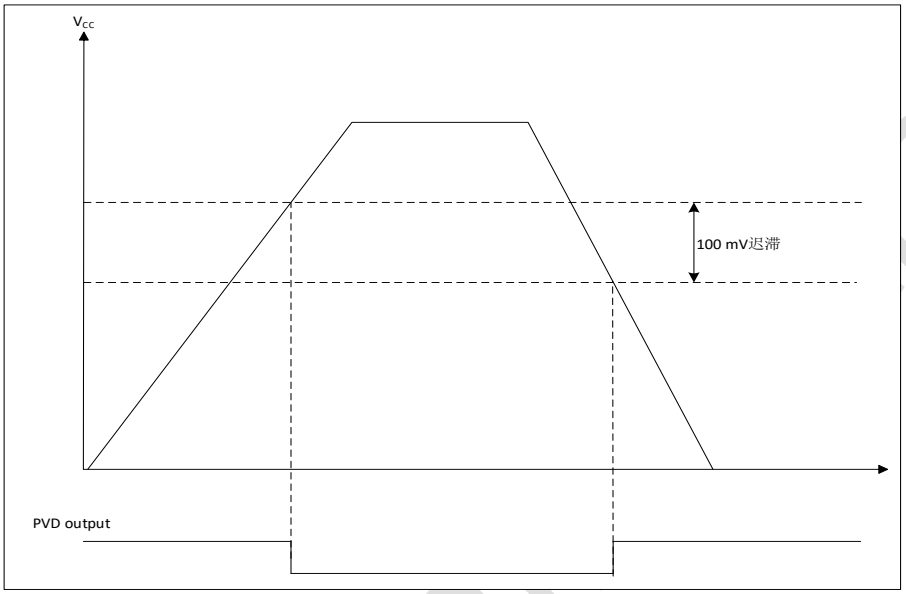


图 5-3 PVD 阈值

模拟输出的 PVD_OUT 信号，在数字端会完成如下功能：

- 数字滤波
- 根据配置，产生中断或者复位

PVD 时钟源选择说明：

RCC_CCIPR .PVDSEL 寄存器选择 PVD 检测时的时钟源。一个为 PCLK，一个为 LSC。时钟源配置有如下要求：

- 选择 LSC 时钟的步骤：
  - ✓ 配置 RCC_BDCR.LSCOSEL，选择低速时钟；
  - ✓ 使能选择的低速时钟（LSI：RCC_CSR.LSION；LSE：RCC_BDCR.LSEON）
  - ✓ 等待低速时钟稳定（LSI：RCC_CSR.LSIRDY；LSE：RCC_BDCR.LSERDY）
- 如果使能滤波（PWR_CR2.FLTEN），则时钟源必须选择 LSI/LSC
- 时钟源为 PCLK 时，如果配置 PWR_CR2.FLTEN，则滤波宽度会不准确，不建议操作
- 时钟源为 LSC 时，可以使能滤波，也可以不使能滤波

5.3. PWR 低功耗模式

默认状态下，芯片在系统或者电源复位之后，进入正常运行模式。当 CPU 不需要持续工作时，芯片可进入低功耗模式。

5.3.1. 低功耗模式介绍

芯片在正常的运行模式之外，有 3 种低功耗模式：

- **Sleep mode**：CPU 时钟关闭（NVIC，SysTick 等工作），外设可以配置为工作模式。（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**：LPR/MR 供电，该模式下 SRAM 和寄存器的内容保持，HSI 关闭。
- **Deep_stop mode**：DLPR 供电，该模式下时钟情况和 Stop 模式相同，但需要更长的唤醒时间。GPIO 和 NRST 可以唤醒 Deep_stop 模式。

  在 Stop 模式，LSI 和 LSE 可以保持工作，LPTIM 等可以保持工作。具体该模式下各模块的工作情况，参照下表。

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，但具备几个周期的快速唤醒能力。

此外，正常运行模式下可以通过下述方法降低功耗：

- 降低系统时钟频率
- 对于不使用的外设，关掉外设时钟
- **Deep Stop mode**：该模式下和 Stop 模式相同，但需要更长的唤醒时间。GPIO，nRST 可以唤醒 Deep Stop 模式。

表 5-2 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	DLPR
Sleep (sleep-now or sleep-on-exit)	WFI or Return from ISR	任何中断	与进入睡眠模式之前一样	CPU 时钟停止，对其他时钟和时钟源没影响。	开 (1)	关	关
	WFE	唤醒事件					
Stop	SLEEPDEEP bit 1. WFI 2. Return from ISR 3. WFE 注：系统时钟需要使用 HSI 8M 为时钟源	任何配置为唤醒的 EXTI 输入（EXTI 寄存器配置）、IWDG、NRST	保持进入前的频率配置	HSI 关闭； LSI 和 LSE 可选择开或者关； 其余模块的时钟关闭。	关/开	关/开	关
Deep stop	SLEEPDEEP bit 1. WFI 2. Return from ISR	任何配置为唤醒的 EXTI 输入（EXTI 寄存	保持进入前的频率配置	HSI/LSI/LSE 关闭；	关	关	开

模式	进入	唤醒源	唤醒时钟	对时钟的影响	Voltage regulator		
					MR	LPR	DLPR
	3. WFE 注：系统时钟需要使用 HSI8M 为时钟源	器配置）、NRST					

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

5.3.2. 各工作模式下的功能

表 5-3 各工作模式下的功能 (1)

外设	运行	Sleep	Stop		Deep Stop	
			工作情况	唤醒能力	工作情况	唤醒能力
CPU	Y	-	-	-	-	-
Flash memory	Y	Y	- (2)	-	- (2)	-
SRAM	Y	O (3)	- (4)	-	- (4)	-
Brown-out reset (BOR)	Y	Y	O	O	O	O
HSI	O	O	-	-	-	-
LSI	O	O	O	-	-	-
LSE	O	O	O	-	-	-
LSE Clock Security System (CSS)	O	O	O	O	-	-
ADC	O	O	-	-	-	-
PVD	O	O	O	O	-	-
Temperature sensor	O	O	-	-	-	-
UART	O	O	-	-	-	-
Timers(TIM1/TIM14)	O	O	-	-	-	-
LPTIM	O	O	O	O	-	-
IWDG	O	O	O	O	-	-
SysTick timer	O	O	-	-	-	-
GPIOs	O	O	O	O	O	O

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

注2. Flash 不下电，但无时钟提供，进入最低功耗状态。

注3. SRAM 的时钟可以被开或者关。

注4. SRAM 不下电，但无时钟提供，进入最低功耗状态。

5.3.3. Sleep 模式

5.3.3.1. 进入 Sleep 模式

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，有两种可选的进入睡眠模式的机制。

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

在睡眠模式，所有的 IO 引脚与运行模式保持相同的状态。

5.3.3.2. 退出 Sleep 模式

如果用 WFI 进入睡眠模式，被 NVIC 响应的任何外设中断可以把芯片从睡眠模式唤醒。

如果用 WFE 进入睡眠模式，当一个事件发生时，芯片退出睡眠模式。唤醒事件可以通过以下方式产生：

- 外设中断使能寄存器被使能，并使能 Cortex M0+的 SEVONPEND 位。当芯片从 WFE 唤醒后继续执行时，外设中断标志位必须被清零。
- 配置外部或者内部 EXTI 输入为事件模式。当 CPU 从 WFE 唤醒后继续执行时，不必清除外设中断标志位。该模式具有最短的唤醒时间，并且没有在中断进入和退出浪费时间。

表 5-4 Sleep-now

Sleep-now	描述
进入方式	WFI 或者 WFE，并且： - SLEEPDEEP = 0 并且 - SLEEPONEXIT = 0
退出方式	如果通过 WFI 进入的睡眠模式，则退出方式是：中断。 如果通过 WFE 进入的睡眠模式，则退出方式是：唤醒事件。
唤醒延迟	无

表 5-5 Sleep-on-exit

Sleep-on-exit	描述
进入方式	WFI，并且： SLEEPDEEP = 0 并且 SLEEPONEXIT = 1
退出方式	中断
唤醒延迟	无

5.3.4. Stop/Deep_stop 模式

Stop 模式是基于 Cortex-M0+的深度睡眠以及对外设时钟的门控，VR 可以被配置成 MR 或者 LPR 供电。在该模式下，HSI 被关闭，SRAM 和寄存器内容处于保持状态，LSI、LSE、LPTIM、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 V_{DDD} 域的数字模块的时钟输入被关闭。

Deep_stop 模式由 DLPR 供电，该模式下时钟情况和 Stop 模式相同，但需要更长的唤醒时间。GPIO，nRST 可以唤醒 Deep_stop 模式。

在 Stop 模式下，所有的 IO 引脚保持跟正常运行模式相同的状态。

注：在进入 Stop 模式前需要切换系统时钟为 HSI

5.3.4.1. 进入 Stop /Deep_stop 模式

为了进一步降低 Stop 模式的功耗，可以通过配置 PWR_CR1.VR_MODE，选择进入不同的 Stop/Deep_stop 模式。

注：Deep_stop 模式即为 PWR_CR1.VR_MODE 配置为 2'b11，选择由 DLPR 供电的 Stop 模式。

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。

5.3.4.2. 退出 Stop/Deep_stop 模式

当通过中断或者唤醒事件退出 Stop 模式时，HSI8M 被选择作为系统时钟源。

在 Stop 模式，如果 VR 处于 LPR 状态，则从 Stop 模式唤醒有额外的延迟。

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。

Deep_stop 模式功耗比 Stop 模式低，但需要更长的唤醒时间。

表 5-6 Stop /Deep_stop mode

Stop mode	描述
进入方式	WFI(wait for interrupt) 或者 WFE（wait for event），并且： - 配置设定： 1） 通过 PWR_CR1 的 VR_MODE 位，选择 VR 工作方式：MR/LPR/DLPR 2） 通过 PWR_CR1 的 FLS_SLPTIME 配置 Flash 的唤醒时间 - 置位 Cortex M0+的 SLEEPDEEP 位 **注：** 为了进入低功耗模式，所有 EXTI 输入的标志位（EXTI_PR 寄存器）、所有外设的中断标志位，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。 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，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。
退出方式	如果使用 WFI 进入 Stop/Deep_stop 模式： - 任何被配置成中断模式的 EXTI 输入 如果使用 WFE 进入 Stop/Deep_stop 模式： - 任何被配置成事件模式的 EXTI 输入 - CPU SEVONPEND 位置位情况下的中断标志位
唤醒延迟	LDO 切换的时间 HSI 唤醒时间 Flash 唤醒时间

5.3.5. 降低系统时钟频率

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

5.3.6. 外设时钟门控

在正常运行模式，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。

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

5.4. PWR 寄存器

该外设的寄存器可以通过 half-word 或者 word 访问。

5.4.1. PWR 控制寄存器 1 (PWR_CR1)

偏移地址：0x00

复位值：0x0000 2000（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.	HSION_CTRL	Res.	Res.	Res.	Res.	Res.	Res.
									RW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VR_MODE[1:0]		FLS_SLP-TIME[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW												

Bit	Name	R/W	Reset Value	Function
31:23	Reserved	-	-	保留
22	HSION_CTRL	RW	1’b0	从 Stop 模式唤醒时，HSI 使能时间控制。 0：等待 MR ready 后使能 HSI 1：唤醒时立即使能 HSI
21:16	Reserved	-	-	保留
15:14	VR_MODE[1:0]	RW	0	低功耗模式 00：MR 模式 01：LPR 模式 11：DLPR 模式 其它：保留（硬件切换为 MR 模式）
13:12	FLS_SLPTIME	RW	2’b10	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’b00：5μs 2’b01：2μs 2’b10：3μs 2’b11：0μs

				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 3μs 内访问 Flash。
11:0	Reserved	-	-	保留

5.4.2. PWR 控制寄存器 2 (PWR_CR2)

偏移地址：0x04

复位值：0x0000 0500（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.	FLT_TIME			FLTEN	PVD_OUT_SEL	PVDT			SRCSEL		Res.	PVDE
				RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11：9	FLT_TIME	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 480μs（16 个 LSI/LSE 时钟） 010：滤波时间大约为 120μs（4 个 LSI/LSE 时钟） 001：滤波时间大约为 60μs（2 个 LSI/LSE 时钟） 000：滤波时间大约为 30μs（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	PVD_OUT_SEL	RW	0	PVD 数字输出控制。 0：模拟 PVD_OUT 不经过同步和滤波输出 1：模拟 PVD_OUT 经过同步和滤波输出
6：4	PVDT[2:0]	RW	3’b000	电压上升沿检测阈值（下降沿检测阈值相应减小 0.1V）。 000：VPVD0 (around 2.2V) 001：VPVD1 (around 2.4V) 010：VPVD2 (around 2.6V) 011：VPVD3 (around 2.8 V) 100：VPVD4 (around 3.0V) 101：VPVD5 (around 3.2V) 110：VPVD6 (around 3.4V) 111：VPVD7 (around 3.6V)

3：1	Reserved	-	-	保留
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。

5.4.3. PWR 状态寄存器 (PWR_SR)

偏移地址：0x14

复位值：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	-	-	保留
11	PVDO	R		PVD 检测结果输出。 0：V_{CC} 超出 PVD 选择的比较阈值 1：V_{CC} 低于 PVD 选择的比较阈值
10:0	Reserved	-	-	保留

6.复位与时钟控制（RCC）

6.1. RCC 简介

本模块完成如下功能：

- 产生系统时钟和系统复位
- 产生各个模块时钟和复位
- 产生时钟源控制信号

6.2. RCC 复位

芯片内设计两种复位，分别是：电源复位和系统复位。

6.2.1. 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：

- 上下电复位（POR/PDR）
- 欠压复位（BOR）

6.2.2. 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 管脚的复位
- 独立看门狗复位(IWDG)
- SYSRESETREQ 软件复位
- 重新加载选项字节复位（OBL）

通过检查 RCC_CSR 寄存器的复位标识位，可以识别复位源。

6.2.2.1. NRST 管脚 (external reset)

通过选项字节（NRST_MODE 位）的装载，NRST 管脚可以被配置成下述模式（具体配置参见 Flash 选项字节）：

- 复位输入

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

在该配置模式下，GPIO 的 PC0 功能无效。

对 NRST 管脚有滤毛刺处理，设计保证 NRST 最小要满足 40μs 宽度，少于该宽度的信号将被滤除。

- GPIO

在该模式下，该管脚可以用作标准的 GPIO，即 PC0。管脚上的复位功能无效。芯片复位只会由芯片内部产生，并且不能传递到管脚上。

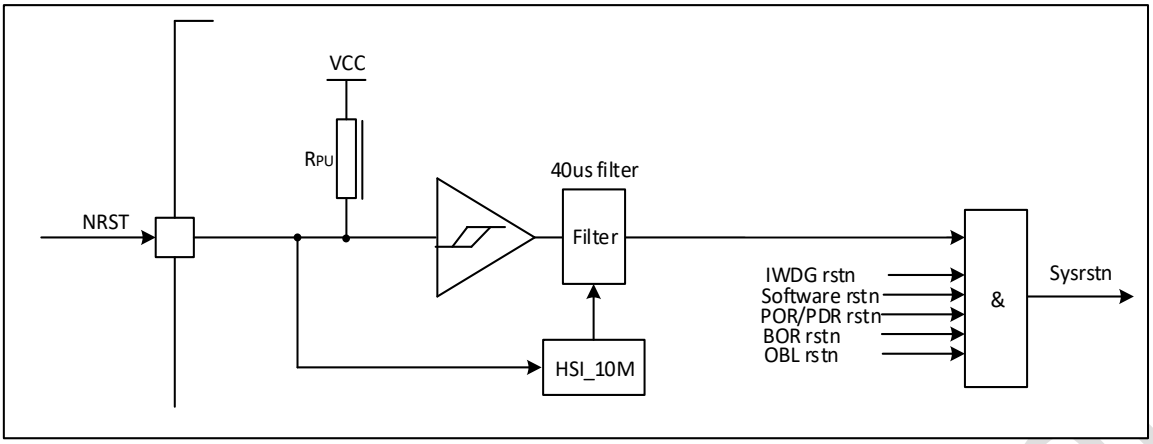


图 6-1 复位电路简化图

6.2.2.2. 看门狗复位

详见独立看门狗（IWDG）章节。

6.2.2.3. 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。

6.2.2.4. 重载选项字节复位

软件通过配置 FLASH_CR.OBL_LAUNCH=1，产生重载选项字节复位，从而启动选项字节再次加载。

6.3. RCC 时钟

6.3.1. 时钟源

6.3.1.1. 外部高速时钟(HSE)

- 外部时钟由 PA6 输入
- 当外部时钟信号使能时，PA6 自动使能该 IO 的输入使能；且 PA6 禁止作为 GPIO 使用

6.3.1.2. 外部低速时钟 LSE

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

LSE 晶体通过在备份域控制寄存器(RCC_BDCR)里的 LSEON 位启动和关闭。驱动能力可以通过 LSE_DRIVER[1:0]进行调节。

在备份域控制寄存器(RCC_BDCR)里的 LSERDY 指示 LSE 晶体振荡是否稳定。在启动阶段，直到这个位被硬件置 1 后，LSE 时钟信号才被释放出来。如果在时钟中断寄存器里被允许，可产生中断申请。

6.3.1.3. 外部时钟源（LSE bypass）

在这个模式里必须提供一个 32.768 kHz 频率的外部时钟源。可以通过设置在备份域控制寄存器(RCC_BDCR)里的 LSEBYP 和 LSEON 位来选择这个模式。具有 50%占空比的外部时钟信号必须连到 OSC32_IN 引脚，同时保证 OSC32_OUT 引脚悬空。

6.3.1.4. 内部高速时钟 HSI

内部高速时钟，作为芯片系统时钟最重要的来源。HSI 时钟源的中心频率设计成 8 MHz，可以软件选择 24 MHz。

6.3.1.5. 内部低速时钟 LSI

内部低速时钟，作为 IWDG 和 LPTIM 的时钟，以及作为芯片低速运行时的系统时钟。该时钟中心频率设计在 32.768 kHz。

6.3.2. 时钟树

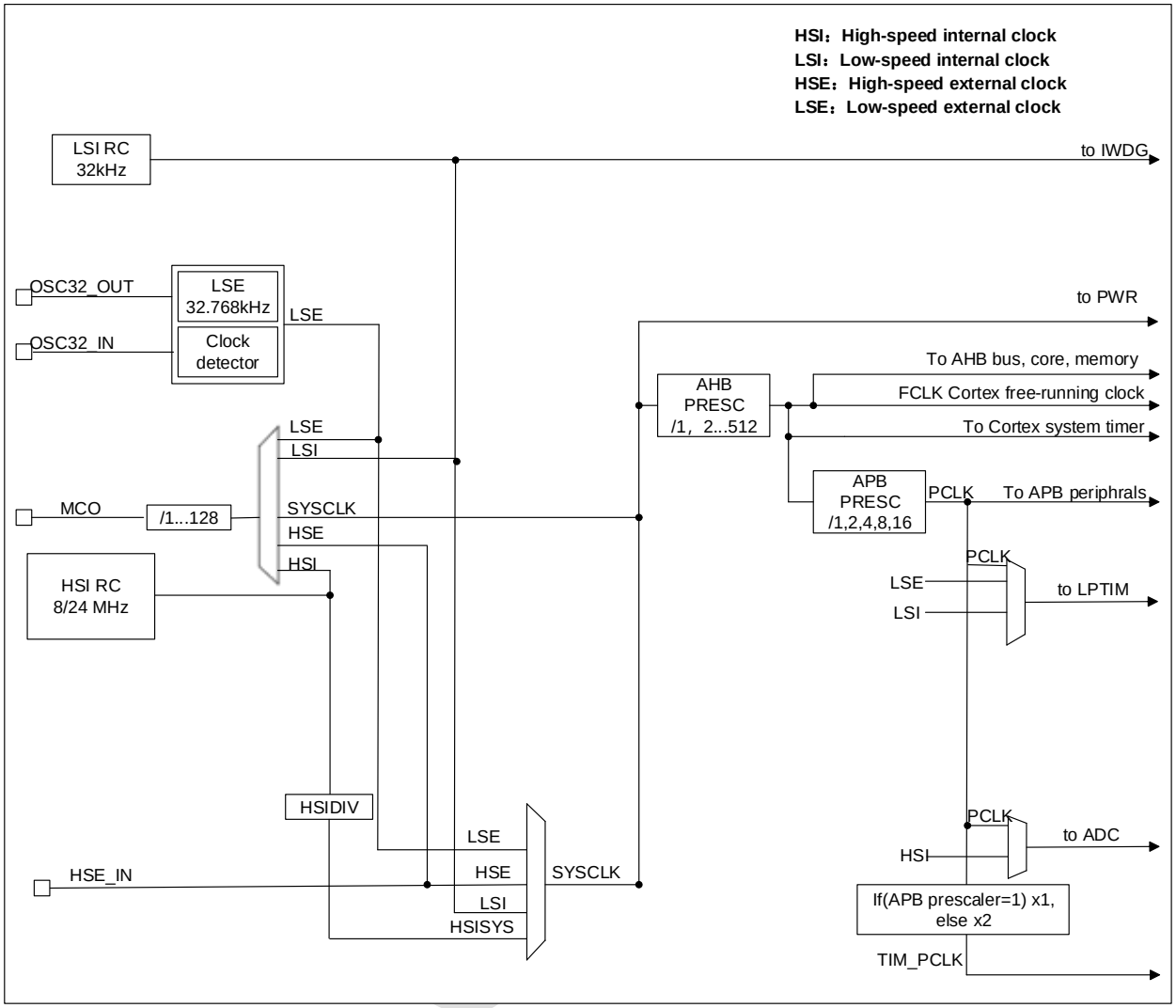


图 6-2 系统时钟结构图

6.3.3. 时钟安全系统 (LSE CSS)

时钟安全系统可以被软件激活。在这种情况下，LSE 的启动延迟后，时钟检测功能被打开。当这个 LSE 被关闭后，时钟检测功能被关闭。

如果在 LSE 上发现时钟错误，LSE 会被自动关闭，时钟错误事件被送给 TIM1（高级定时器）的刹车输入端，并产生中断 Clock Security System Interrupt（CSSI）通知软件该错误，进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex-M0+的 NMI（Non-maskable interrupt）。

注：一旦 CSS 被使能，并且如果 LSE 时钟错误，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 被直接或者间接的用作系统时钟，时钟错误将导致系统时钟自动切换到 LSI，同时关闭 LSE。

6.3.4. 输出时钟

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

表 6-1 输出时钟选择

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

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

6.3.5. 时钟校准

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

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

6.3.5.1. HSI 校准

- 时钟测量

基本原理是基于时钟源的比率，精度与两个时钟源之比紧密相关。比率越大，测量效果越好。

借助 LSE 信号连续边沿之间的 HSI 时钟计数数量，即可对内部时钟周期进行测量。利用 LSE 的高精度（ppm 级），用户能以同一分辨率测定时钟频率，并可通过对时钟源进行微调来补偿与生产、工艺、温度及电压相关的频率偏差。

HSI 振荡器均设有针对此目的的专用校准位，且支持用户访问。

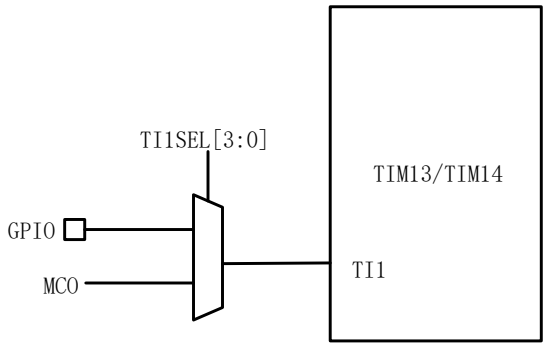


图 6-3 频率测量与 TIM14 捕获模式

Timer 14 的输入捕获通道可以是 GPIO 或者芯片内的时钟。

对于这些时钟的选择，是通过 TIM14_TISEL 的 TI1SEL[3:0]寄存器实现的。其中 2 种选择如下所示：

1. TIM14 通道 1 连接到 GPIO
2. TIM14 通道 1 连接到 MCO（Microcontroller clock output）

■ 时钟校准

一旦检测到 HSI 时钟异常，通过中断，通知软件处理。软件通过微调内部时钟校准参数，从而实现动态校准的目的。

通过 MCO multiplexer 连接 LSE 到 TIM14 通道 1 的输入捕获，其主要目的是能精准的测量 HSI（这种情况下，HSI 应该设置为系统时钟源）。对在连续两个 LSE 信号的变化沿期间的 HSI 时钟个数的计数，这样的机制提供了对内部时钟周期的度量。

这也是充分利用了外接晶振时的 LSE 高精度（ppm），才可能以相同的分辨率决定内部时钟频率，然后对时钟源进行校准，以补偿由于工艺、温度、电压相关的频率漂移。

HSI 也因此设计有专门的用户可访问的校准寄存器位。

该实现机制的基本原理就是相对的度量（比如，HSI/LSE 的比率）：准确度因而会与两个时钟源频率的比率密切相关。比率越高，度量效果越好。

6.3.5.2. LSI 校准

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

LSI 的校准是连接 LSI 的输出和 TIM14 的输入捕获

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

6.4. RCC 寄存器

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

6.4.1. RCC 时钟控制寄存器（RCC_CR）

偏移地址：0x00

复位值：0x0000 0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HSEON	Res	Res
													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
31:19	Reserved	-	-	保留
18	HSEON	RW	0	外部时钟使能 0：无 HSE 时钟

Bit	Name	R/W	Reset Value	Function
				1：管脚输入 HSE 时钟
17:14	Reserved	-	-	保留
13:11	HSIDIV[2:0]	RW	3'h0	HSI 时钟分频系数。 软件控制这些位设定 HSI 的分频系数，产生 HSISYS 时钟 000：1 001：2 010：4 011：8 100：16 101：32 110：64 111：128
10	HSIRDY	R	0	HSI 时钟准备标志。 硬件置位表明 HSI OSC 稳定。该位只有当 HSION=1 时才有效。 0：HSI OSC 没有准备好； 1：HSI OSC 准备好了； 当 HSION 清零后，HSIRDY 立即拉低。
9	Reserved	-	-	保留
8	HSION	RW	1	HSI 时钟使能位。软件可以置位和清零该位。 当进入 Stop 模式时，硬件清零该位，停止 HSI。 当 HSI 被直接或者间接用作系统时钟（也当退出 Stop 模式）。 0：HSI 关闭 1：HSI 打开
7:0	Reserved	-	-	保留

6.4.2. RCC 内部时钟源校准寄存器（RCC_ICSCR）

偏移地址：0x04

复位值：0x00FF 10FF，通过 POR/BOR 复位

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	LSI_TRIM[8:0]									Res	Res	HSI_FS_CR
				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**
HSI_TC_TRIMCR[3:0]				HSI_ABS_TRIM[12:0]											
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

该寄存器需注意：

1. HSI_FS_CR、HSI_TC_TRIMCR 、HSI_ABS_TRIMCR 需要同时配置。
2. 连续的两次配置之间不能少于 6 个 HSI。

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:19	LSI_TRIM	RW	9'hFF	内部低速时钟频率调整，通过校准，内部低速时钟可以输出 32.768 kHz。 上电后芯片硬件会把出厂信息写入该寄存器中，实现 LSI 特定输出频率下的校准。 校准值保存在 Flash 的如下地址： 32.768 kHz 校准值地址：0x1FFF 0110 软件通过对该寄存器值进行改写，每增（减）1，使 LSI 的输出频率增（减）约 0.2%。
18:17	Reserved	-	-	保留
16	HSI_FS_CR	RW	0	HSI 频率选择： 0：8MHz 1：24MHz 上电后，默认选择 8MHz。在系统复位后，硬件同样切换成 8MHz。 更新频率时必须同时更新该频率对应的 trim。
15:12	HSI_TC_TRIMCR	RW	4'h8	HSI 时钟温度系数校准值。 上电后硬件加载 HSI8M 对应的 trimming 值。 软件通过读出存放在 information 区相应地址的数据，写入该寄存器，实现 HSI 特定输出频率下的校准。 保存在 Flash 的如下地址内： 24MHz 校准值存放地址：0x1FFF 0088 8MHz 校准值存放地址：0x1FFF 0080 通过向该寄存器写入校准值，也可以该值为中心值，修改该寄存器数值。
11:0	HSI_ABS_TRIMCR	RW	12'h200	HSI 时钟频率校准值。 上电后硬件加载 HSI8M 对应的 trimming 值。 软件通过读出存放在 information 区相应地址的数据，写入该寄存器，实现 HSI 特定输出频率下的校准。 保存在 Flash 的如下地址内： 24MHz 校准值存放地址：0x1FFF 0088 8MHz 校准值存放地址：0x1FFF 0080 通过向该寄存器写入校准值后，也可以该值为中心值，修改该寄存器数值，每增（减）1，则 HSI 的输出频率增（减）约 0.1%。 HSI_ABS_TRIMCR [11:8]:粗调位，HSI_ABS_TRIMCR [7:0]细调位。在 trim 时需要分两段控制，如粗调位不变改变细调位。

6.4.3. RCC 时钟配置寄存器（RCC_CFGR）

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	MCOPRE[2:0]			Res	MCOSEL[2:0]			Res	Res	Res	Res	Res	Res	Res	Res
	RW	RW	RW		RW	RW	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	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	-	-	保留
30:28	MCOPRE[2:0]	RW	3’h0	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： 000：1 001：2 010：4 011：8 100：16 101：32 110：64 111：128 推荐在 MCO 输出使能前，设置这些位。
27	Reserved	-	-	保留
26:24	MCOSEL[2:0]	RW	3’h0	MCO 选择 000：没有时钟，MCO 输出不使能 001：SYSCLK 010：保留 011：HSI 100：HSE 101：保留 110：LSI 111：LSE 注：在时钟启动或者切换阶段可能会出现输出时钟不完整的情况。
23:15	Reserved	-	-	保留
14:12	PPRE[2:0]	RW	3’h0	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

Bit	Name	R/W	Reset Value	Function
				100：2 101：4 110：8 111：16
11:8	HPRE[3:0]	RW	3’h0	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 为保证系统正常工作，需根据 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	-	-	保留
5:3	SWS[2:0]	R	3’h0	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：保留 011：LSI 100：LSE 其它：保留
2:0	SW[2:0]	RW	3’h0	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：保留 011：LSI 100：LSE 其它：保留 硬件配置为 HSISYS 的情况包括： 1)系统从 Stop 模式退出

6.4.4. RCC 外部时钟源控制寄存器（RCC_ECSCR）

偏移地址：0x10

复位值： 0x0001 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	LSE_STARTUP		Res		LSE_DRIVER	
										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

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	LSE_STARTUP	RW	2’h0	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	-	-	保留
17:16	LSE_DRIVER	RW	2’h3	低速晶振驱动能力选择。 00：最弱驱动能力 01：弱驱动能力 10：强驱动能力 11：最强驱动能力（默认） 注：需要根据晶振特性、负载电容以及电路板的寄生参数选择适当的驱动能力。驱动能力越大则功耗越大，驱动能力越弱则功耗越小。
15:0	Reserved	-	-	保留

6.4.5. RCC 时钟中断使能寄存器（RCC_CIER）

偏移地址：0x18

复位值：0x0000 0000

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

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	HSI RDYIE	Res	LSE RDYIE	LSI RDYIE
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	HSIRDYIE	RW	0	HSI 时钟 ready 中断使能。 0：禁止 1：使能
2	Reserved	-	-	保留
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

6.4.6. RCC 时钟中断标志寄存器（RCC_CIFR)

偏移地址：0x1C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	LSECSSF	R	0	LSE 时钟安全系统（CSS）中断标志。 当硬件检测 LSE 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。
8:4	Reserved	-	-	保留
3	HSIRDYF	R	0	HSI 准备中断标识位

Bit	Name	R/W	Reset Value	Function
				当 HSI 稳定并且 HSIRDYIE 使能，该位由硬件置位。软件通过置位 HSIRDYC 位，清零该位。 0：无由 HSI 引起的时钟准备中断 1：有由 HSI 引起的时钟准备中断
2	Reserved	-	-	保留
1	LSERDYF	R	0	LSE 准备中断标识位 当 LSE 稳定并且 LSERDYIE 使能，该位由硬件置位。软件通过置位 LSERDYC 位，清零该位。 0：无由 LSE 引起的时钟准备中断 1：有由 LSE 引起的时钟准备中断
0	LSIRDYF	R	0	LSI 准备中断标识位 当 LSI 稳定并且 LSIRDYIE 使能，该位由硬件置位。软件通过置位 LSIRDYC 位，清零该位。 0：无由 LSI 引起的时钟准备中断 1：有由 LSI 引起的时钟准备中断

6.4.7. RCC 时钟中断清除寄存器（RCC_CICR)

偏移地址：0x20

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	LSECSSC	W	0	LSE 时钟安全系统（CSS）中断标志清零。 0：没有影响； 1：清零 LSECSSF 标志
8:4	Reserved	-	-	保留
3	HSIRDYC	W	0	HSI 准备标志清零。 0：没有影响。 1：清除 HSIRDYF 位。
2	Reserved	-	-	保留
1	LSERDYC	W	0	LSE 准备标志清零。 0：没有影响。

Bit	Name	R/W	Reset Value	Function
				1：清除 LSERDYF 位。
0	LSIRDYC	W	0	LSI 准备标志清零。 0：没有影响。 1：清除 LSIRDYF 位。

6.4.8. RCC GPIO 复位寄存器（RCC_IOPRSTR)

偏移地址：0x24

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	GPIOCRST	RW	0	I/O PortC 复位。 0：没有影响 1：PortC I/O 复位
1	GPIOBRST	RW	0	I/O PortB 复位。 0：没有影响 1：PortB I/O 复位
0	GPIOARST	RW	0	I/O PortA 复位。 0：没有影响 1：PortA I/O 复位

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

偏移地址：0x2C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31	LPTIMRST	RW	0	LPTIM 模块复位。 0：没有影响 1：该模块复位
30:29	Reserved	-	-	保留
28	PWRRST	RW	0	Power 接口模块复位。 0：没有影响 1：该模块复位
27	DBGRST	RW	0	MCU Debug 模块复位。 0：没有影响 1：该模块复位
26:19	Reserved	-	-	保留
18	UARTRST	RW	0	UART 模块复位。 0：没有影响 1：该模块复位
17:0	Reserved	-	-	保留

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

偏移地址：0x30

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	VREFBUF RST	Res	Res	Res	Res	Res	ADC RST	Res	Res	Res	Res
					RW						RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14 RST	Res	Res	Res	TIM1 RST	Res	PWM RST	Res	Res	Res	Res	Res	Res	Res	Res	SYS CFG RST
RW				RW		RW									RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	保留
26	VREFBUFRST	RW	0	V_{REFBUF} 模块复位。 0：没有影响 1：该模块复位
25:23	Reserved	-	-	保留
20	ADCRST	RW	0	ADC 模块复位。 0：没有影响 1：该模块复位

Bit	Name	R/W	Reset Value	Function
19:16	Reserved	-	-	保留
15	TIM14RST	RW	0	TIM14 模块复位。 0：没有影响 1：该模块复位
14:12	Reserved	-	-	保留
11	TIM1RST	RW	0	TIM1 模块复位。 0：没有影响 1：该模块复位
10	Reserved	-	-	保留
9	PWMRST	RW	0	PWM 模块复位。 0：没有影响 1：该模块复位
8:1	Reserved	-	-	保留
0	SYSCFGRST	RW	0	SYSCFG 模块复位。 0：没有影响 1：该模块复位

6.4.11. RCC GPIO 时钟使能寄存器（RCC_IOPENR）

偏移地址：0x34

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
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

Bit	Name	R/W	Reset Value	Function
				1：时钟使能

6.4.12. RCC AHB 外设时钟使能寄存器（RCC_AHBENR）

偏移地址：0x38

复位值：0x0000 0300

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	SRAMEN	RW	1	在睡眠模式下，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，在正常运行模式，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，在正常运行模式，该模块时钟不会关闭
7:0	Reserved	-	-	保留

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

偏移地址：0x3C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31	LPTIMEN	RW	0	LPTIM 模块时钟使能。 0：禁止 1：使能
30:29	Reserved	-	-	保留
28	PWREN	RW	0	低功耗控制模块时钟使能。 0：禁止 1：使能
27	DBGEN	RW	0	Debug 模块时钟使能。 0：禁止 1：使能
26:19	Reserved	-	-	保留
18	UARTEN	RW	0	UART 模块时钟使能。 0：禁止 1：使能
17:0	Reserved	-	-	保留

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

偏移地址：0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	VERFBUFEN	Res	Res	Res	Res	Res	ADC EN	Res	Res	Res	Res
					RW						RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TIM14 EN	Res	Res	Res	TIM1 EN	Res	PWM EN	Res	Res	Res	Res	Res	Res	Res	Res	SYS CFG EN
RW				RW		RW									RW

Bit	Name	R/W	Reset Value	Function
31:27	Reserved	-	-	保留
26	VREFBUFEN	RW	0	V_{REFBUF} 模块时钟使能。 0：禁止 1：使能
25:21	Reserved	-	-	保留
20	ADCEN	RW	0	ADC 模块时钟使能。 0：禁止 1：使能

Bit	Name	R/W	Reset Value	Function
19:16	Reserved	-	-	保留
15	TIM14EN	RW	0	TIM14 模块时钟使能。 0：禁止 1：使能
14:12	Reserved	-	-	保留
11	TIM1EN	RW	0	TIM1 模块时钟使能。 0：禁止 1：使能
10	Reserved	-	-	保留
9	PWMEN	RW	0	PWM 模块时钟使能。 0：禁止 1：使能
8:1	Reserved	-	-	保留
0	SYSCFGEN	RW	0	SYSCFG 模块时钟使能。 0：禁止 1：使能

6.4.15. RCC 外设独立时钟配置寄存器（RCC_CCIPR）

偏移地址：0x54

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:18	LPTIMSEL[1:0]	RW	2’h0	LPTIM1 内部时钟源选择。 00：PCLK 01：LSI 10：无时钟 11：LSE
17:18	Reserved	-	-	保留
7	PVDSEL	RW	0	PVD 检测时钟源选择。 0：PCLK

				1：LSC（RCC_BDCR.LSCOSEL 选择后的时钟） 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
6:0	Reserved	-	-	保留

6.4.16. RCC 域控制寄存器（RCC_BDCR）

偏移地址：0x5C

复位值：0x0000 0000，通过 POR/BOR 复位

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	LSCOSEL	Res	Res	Res	Res	Res	Res	Res	Res	Res
						RW									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res		Res	LSECSSD	LSECS-SON	Res		LSE BYP	LSE RDY	LSE ON
									R	RW			RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	LSCOSEL	RW	0	低速时钟选择。 0：LSI 1：LSE
24:7	Reserved	-	-	保留
6	LSECSSD	R	0	LSE CSS(时钟安全系统)检测失败。 该位由硬件置位，表明 CSS 检测 32.768 kHz OSC（LSE）失败。 0：未检到 LSE 失败 1：检测 LSE 失败
5	LSECSSON	RW	0	LSE CSS 使能 0：禁止 1：使能 必须 LSEON=1 并且 LSERDY=1 后才能使能 LSECSSON。 一旦使能该位，不能再把该位禁止，除非 LSECSSD=1。
4:3	Reserved	-	-	保留
2	LSEBYP	RW	0	LSE OSC bypass 0：没有旁路，低速外部时钟选择晶振 1：旁路，低速外部时钟选择外部接口输入时钟

Bit	Name	R/W	Reset Value	Function
				注：只有当外部 32.768 kHz OSC 禁止（LSEON=0 并且 LSERDY=0）时才能写该位。
1	LSERDY	R	0	LSE OSC 准备位。 硬件置位，硬件清零，表明当 LSE 稳定时 0：没有准备好 1：准备
0	LSEON	RW	0	LSE OSC 使能。 0：禁止 1：使能

6.4.17. RCC 控制/状态寄存器（RCC_CSR)

偏移地址：0x60

复位值：0x0800 0000

复位源如下：1）[29:25]：POR 复位；2）LSION：系统复位；3）NRST_FLTIDS 不会被系统复位

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	IWDG RSTF	SFT RSTF	PWR RSTF	PIN RSTF	OBL RSTF	Res	RMVF	Res	Res	Res	Res	Res	Res	Res
		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	NRST_FLT-IDIS	Res	Res	Res	Res	Res	Res	LSI RDY	LSION
							RW							R	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
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	1	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	-	-	保留
23	RMVF	RW	0	需通过软件置 1 来清零[29:25]的复位标志。
22:9	Reserved	-	-	保留

Bit	Name	R/W	Reset Value	Function
8	NRST_FLTDIS	RW	0	NRST 滤波禁止 0：使能滤波功能 1：禁止滤波功能
7:2	Reserved	-	-	保留
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 软件置位，软件清零。在硬件使能 IWDG（通过选项字节）和软件使能 LSECSSON 时，硬件会对该位进行置位。

7. 通用 I/O（GPIO）

7.1. GPIO 简介

每个 GPIO 端口有：

- 4 个 32 位配置寄存器 (GPIOx_MODER 、 GPIOx_OTYPER 、 GPIOx_OSPEEDR 、 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 位置位/复位寄存器(GPIOx_BSRR)
- 1 个 32 位锁定寄存器(GPIOx_LCKR)
- 1 个复用功能选择寄存器(GPIOx_AFR)。

7.2. GPIO 主要特征

- 受控 IO 最多达 18 个
- 输出状态：推挽或开漏 + 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
- 可为每个 IO 选择不同的速度
- 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）和复位寄存器（GPIOx_BRR）可对 GPIOx_ODR 按位写
- 锁定机制（GPIOx_LCKR）会冻结 I/O 口配置功能
- 模拟功能
- 复用功能选择寄存器（每个 IO 口最多 16 种复用功能）
- 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

7.3. GPIO 功能描述

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

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟功能
- 具有上拉或下拉功能的开漏输出
- 具有上拉或下拉功能的推挽输出
- 具有上拉或下拉功能的复用功能推挽输出/输入
- 具有上拉或下拉功能的复用功能开漏输出/输入

每个 IO 端口位均可自由编程，但 IO 端口寄存器必须按字、半字或字节进行访问。GPIOx_BSRR 和 GPIOx_BRR 寄存器允许对任何 GPIO 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

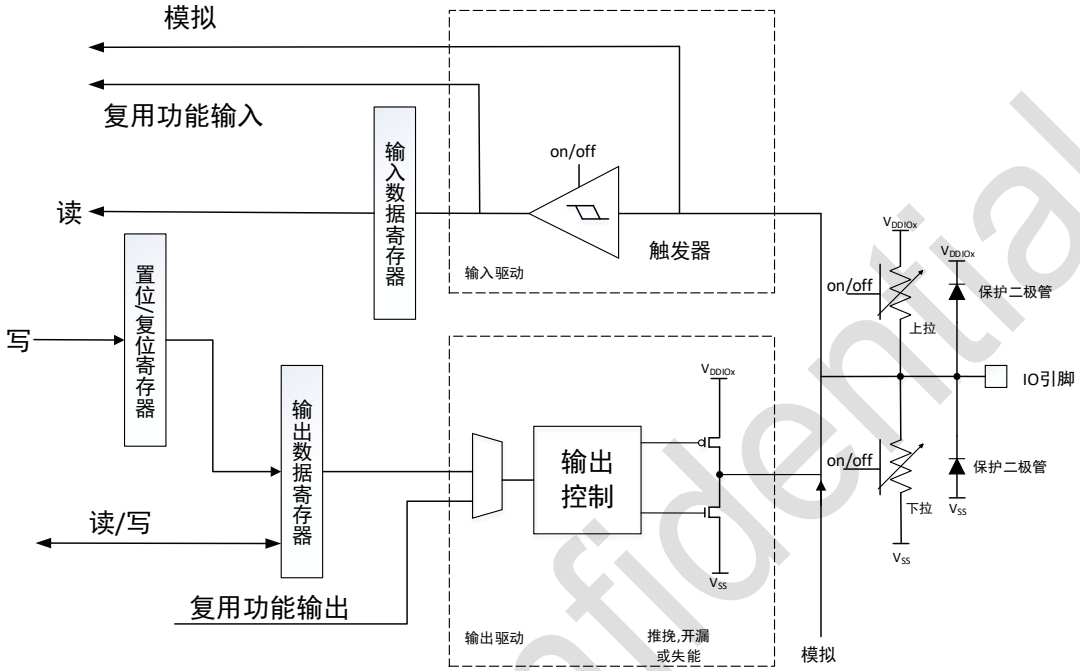


图 7-1 IO 端口位的基本结构

7.3.1. 通用 I/O(GPIO)

在复位期间及复位刚刚完成后，复用功能尚未激活，大部分 IO 端口被配置为模拟模式。复位后，调试引脚处于复用功能上拉/下拉状态：

- PA2-SWCLK：处于下拉状态
- PB6-SWDIO：处于上拉状态

当引脚被配置为输出时，写到输出数据寄存器上的值(GPIOx_ODR)输出到相应的 IO 引脚。可以在推挽模式或开漏模式（仅驱动低电平，高电平为 HI-Z）下使用输出驱动。

输入数据寄存器（GPIOx_IDR）每隔 1 个 AHB 时钟周期捕获一次 IO 引脚的数据。

所有 GPIO 引脚都具有内部弱上拉及下拉电阻，可根据 GPIOx_PUPDR 寄存器中的值来打开/关闭。

7.3.2. 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 口上的可用外设出现冲突。

每个 I/O 口上的多路选择器最多达 16 种复用功能输入（AF0 到 AF15），可通过寄存器 GPIOx_AFR 来配置。

调试功能

系统复位后，调试相关的引脚复用为调试功能，供系统调试接口使用。

GPIO

在 GPIOx_MODER 寄存器中将所需 IO 配置为输出、输入、模拟、复用中的一种

外设复用功能

- 寄存器 GPIOx_AFR 可配置每个 IO 的复用编号(AF0~AF15)
- 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器，分别选择输出类型、上拉/下拉以及输出速度
- 在 GPIOx_MODER 寄存器中将所需 IO 配置为复用模式

额外功能

对于 ADC，需要通过寄存器 GPIOx_MODER 配置相应的 IO 为模拟模式，并在 ADC 中配置相应功能；需要特别注意，对于模拟附加功能的输出，需要先配置相应 IO 为模拟模式，再使能外设的控制寄存器。对于晶振额外功能，在相应的 PWR 和 RCC 寄存器里配置各自功能。这些配置比标准的 GPIO 配置具有更高优先级。

7.3.3. I/O 控制寄存器

每个 GPIO 有 4 个控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR），可配置多达 8 个 IO。GPIOx_MODER 寄存器用于选择 IO 工作模式（输入、输出、复用、模拟）。GPIOx_OTYPER 用于选择输出类型（推挽或开漏）。GPIOx_OSPEEDR 寄存器用于设定 IO 速度 。GPIOx_PUPDR 寄存器用于选择上拉/下拉。

7.3.4. I/O 数据寄存器

每个 GPIO 有 2 个数据寄存器：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，只读寄存器。

7.3.5. 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，它允许应用程序在输出数据寄存器(GPIOx_ODR) 中对各个单独的数据位执行置位和复位操作。

GPIOx_ODR 中的每个数据位对应于 GPIOx_BSRR 中的两个控制位：BS (i)和 BR (i)。当写入 1 时，BS(i) 位会置位对应的 ODR(i) 位。当写入 1 时，BR(i)位会清零 ODR(i)对应的位。

在 GPIOx_BSRR 中向任何位写入 0 都不会对 GPIOx_ODR 中的对应位产生任何影响。如果在 GPIOx_BSRR 中同时尝试对 GPIOx_ODR 中的某个位执行置位和清零操作，则置位操作优先。

使用 GPIOx_BSRR 寄存器更改 GPIOx_ODR 中各个位的值是一个“单次”操作，不会锁定 GPIOx_ODR 位。随时都可以直接访问 GPIOx_ODR 位。GPIOx_BSRR 寄存器提供了一种执行原子按位处理的方法。在对 GPIOx_ODR 进行位操作时，软件无需禁止中断：在一次原子 AHB 写访问中，可以修改一个或多个位。

复位寄存器 (GPIOx_BRR) 是一个 32 位寄存器，它允许应用程序在输出数据寄存器(GPIOx_ODR) 中对各个单独的数据位执行复位操作。复位寄存器的作用与置位复位寄存器相似，但仅能复位输出数据寄存器的一个或多个位。

7.3.6. GPIO 锁定机制

通过将特定的写序列应用到 GPIOx_LCKR 寄存器，可以冻结 GPIO 控制寄存器。冻结的寄存器包括 GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFR

要对 GPIOx_LCKR 寄存器执行写操作，必须用特定的写/读序列，软件以正确的锁定序列操作此寄存器的 LCKR[16]后，会使用 LCKR[7:0]的值来锁定 IO 的配置（在写序列期间，LCKR[7:0]的值必须保持不变）。某个或多个端口被锁定后，无法解锁，直到发生系统复位，或 GPIO 被复位。每个 GPIOx_LCKR 位都会冻结控制寄存器（GPIOx_MODER、GPIOx_OTYPER、 GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFR）中的对应位。

锁定序列只能通过字访问对 GPIOx_LCKR 寄存器操作。

具体操作参考 GPIOx_LCKR 寄存器描述。

7.3.7. I/O 复用功能输入/输出模式配置

有 1 个寄存器可用来从每个 IO 可用的 16 个复用功能输入/输出中进行选择。

不同的复用信号有不同的输入/输出方向，具体情况由各自的外设模块决定。

7.3.8. 外部中断/唤醒线

所有端口都具有外部中断功能。要使用外部中断线，必须将端口配置为输入模式，请参见中断和事件章节。

7.3.9. I/O 输入配置

当 IO 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
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输入数据寄存器可得知引脚电平

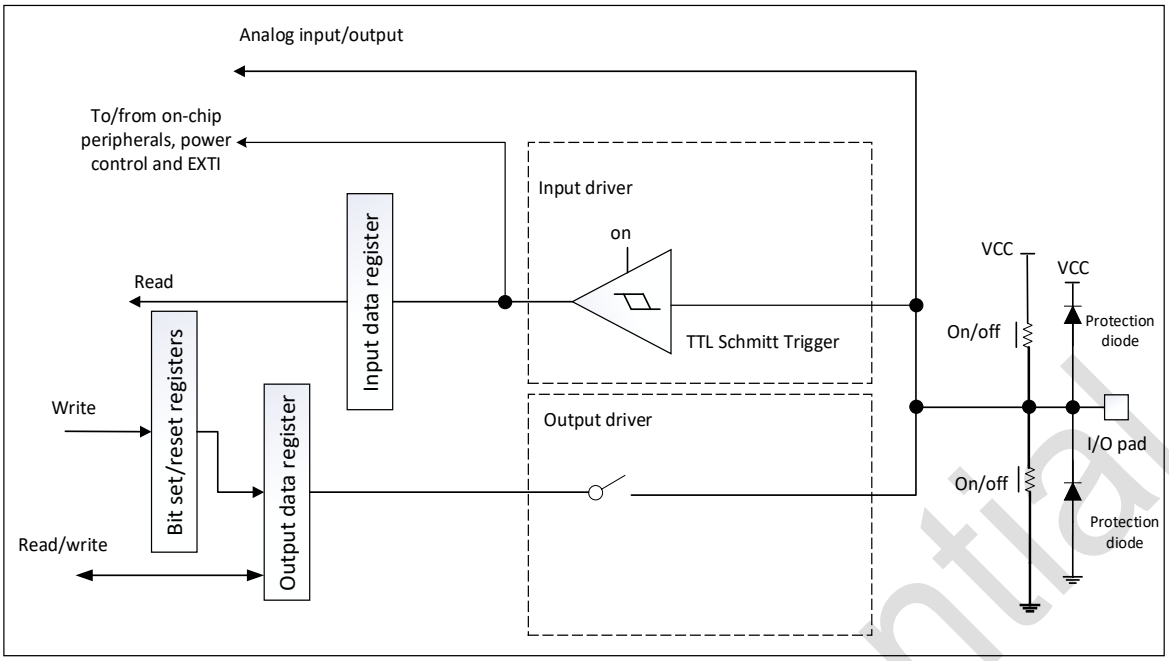


图 7-2 输入浮空/上拉/下拉配置

7.3.10. I/O 输出配置

当 IO 端口被配置为输出时：

- 输出缓冲器使能
  - 开漏模式：输出寄存器上的’0’导通 N-MOS，而输出寄存器上的’1’将端口置于高阻状态(PMOS 不导通)。
  - 推挽模式：输出寄存器上的’0’导通 N-MOS，而输出寄存器上的’1’将导通 P-MOS。
- 施密特触发输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输入数据寄存器可得知引脚电平
- 读取输出数据寄存器可得知上一次写 GPIO 的值

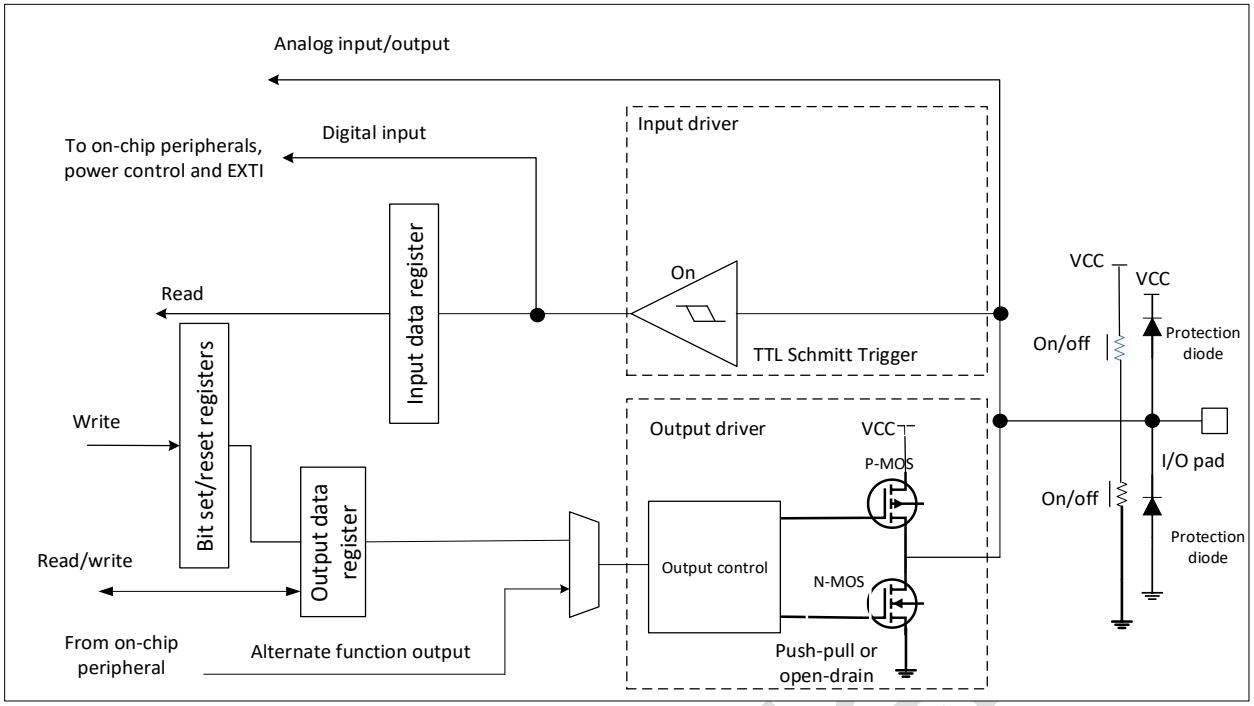


图 7-3 输出配置

7.3.11. 复用功能配置

当 IO 端口被配置为复用功能时：

- 输出缓冲器使能
- 输出缓冲器的输出数据和输出使能由外设模块驱动
- 施密特触发输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输出数据寄存器可得知上一次写 GPIO 的值

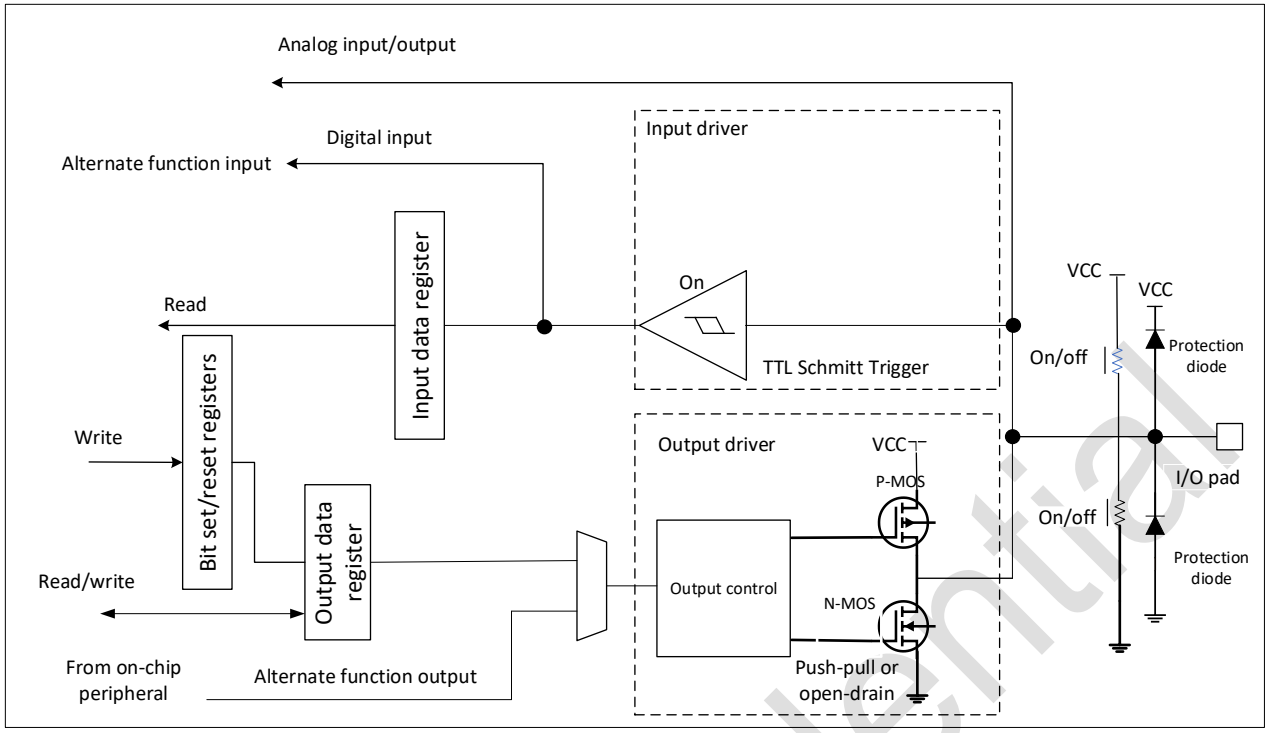


图 7-4 复用功能配置

7.3.12. 模拟配置

当 IO 端口被配置为模拟模式时：

- 输出缓冲器不使能；
- 施密特触发器输入不使能，实现了每个模拟 IO 引脚上的零功耗。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'。

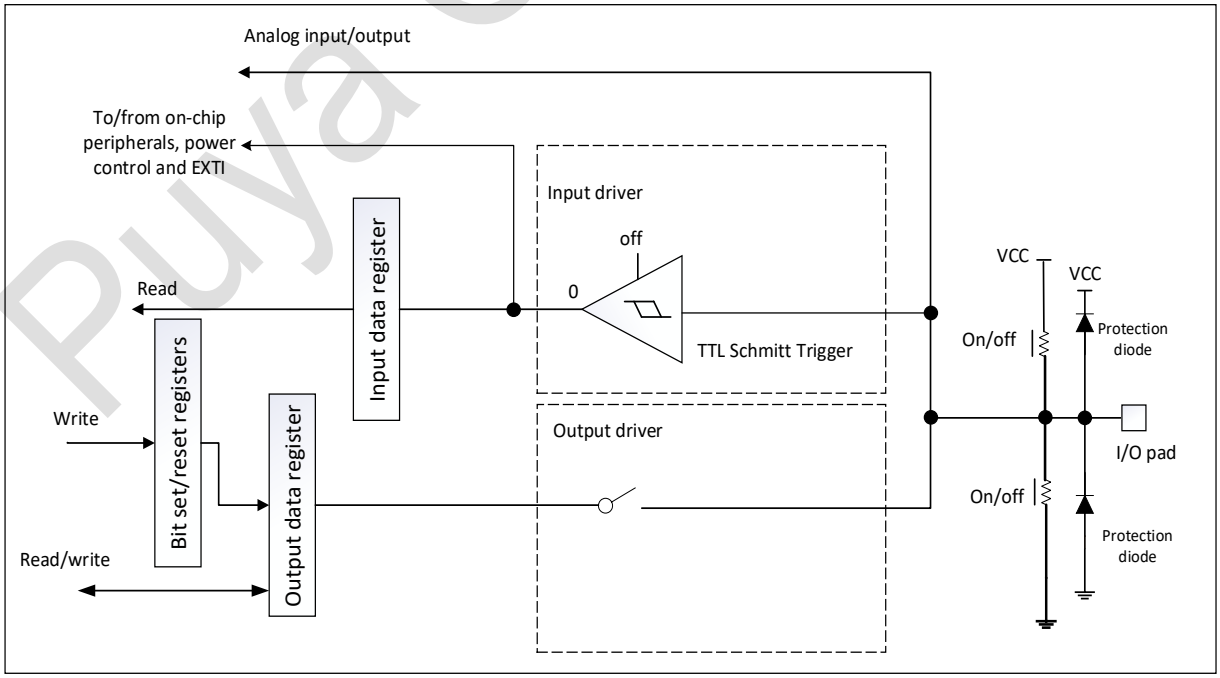


图 7-5 高阻抗模拟配置

7.3.13. 使用 PC1 作为 GPIO/LSE 输入

当 LSE 不使能（复位后的默认），相应的管脚可以当作正常的 GPIO。

当 LSE 使能（RCC 寄存器中置位 LSEON），相关引脚的功能由振荡器控制，不受这些 GPIO 寄存器的设定影响。

当晶振配置为用户外部时钟模式，只有 OSC32_IN（PC1）用于接收外部时钟输入，而 OSC32_OUT（PB7）仍然可以用作正常 GPIO。

7.3.14. 使用 PC0 作为 GPIO/NRST

PC0 可以用作复位引脚（NRST）或 GPIO。根据用户选项字节中的 NRST_MODE 位，它切换到以下模式：

外部复位模式(仅复位输入)：NRST_MODE=0

GPIO 模式：NRST_MODE=1

7.4. GPIO 寄存器

所有 GPIO 相关寄存器都可进行字节（8 位）、半字（16 位）或字（32 位）写操作。

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

偏移地址：0x00

复位值：

- GPIOA：0x0000FFEF
- GPIOB：0x0000EFFF
- GPIOC：0x0000000F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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
31:16	Reserved	-	-	保留
15：0	MODEy[1:0]	RW		y = 7..0 软件通过这些位配置相应的 IO 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(复位状态)

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

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	OTy	RW	0	y = 7..0 软件配置 IO 的输出类型 0：推挽输出 (复位状态) 1：开漏输出

7.4.3. GPIO 端口输出速度寄存器(GPIOx_OSPEEDR) (x=A~C)

偏移地址：0x08

复位值：

- GPIOB：0x00001000
- 其他：0x00000000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	OSPEEDy[1:0]	RW	PB6:2'b01 其他:2'b00	y = 7..0 软件配置 IO 口的输出速度 00：低速 01：高速 11：非常高速 其他：高速

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

偏移地址：0x0C

复位值：

- GPIOA：0x00000020
- GPIOB：0x00001000
- GPIOC：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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
31:16	Reserved	-	-	保留
15:0	PUPDy [1:0]	RW	PA2:2'b10 PB6:2'b01 其他:2'b00	y = 7..0 软件配置 IO 口上拉或者下拉 00：无上下拉 01：上拉 10：下拉 11：上拉+下拉

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

偏移地址：0x10

复位值：0x0000 XXXX

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	IDy	R		y = 7..0 这是只读的，读出值位对应 IO 口的状态

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

偏移地址：0x14

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	保留
7：0	ODy	RW	0	y = 7..0 对 GPIOx_BSRR 或 GPIOx_BRR 寄存器(x=A/B/C)， 可以分别对各个 ODR 位进行独立的设置/清除。

7.4.7. GPIO 端口位设置/复位寄存器(GPIOx_BSRR) (x=A~C)

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
								W	W	W	W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:16	BRy	W		y = 7..0 读返回值是 0 0：对应的 ODRy 位不受影响 1：清除对应的 ODRy 位 注：如果同时写 BSy 和 BRy 的对应位，BSy 位起作用
15:8	Reserved	-	-	保留
7：0	BSy	W		y = 7..0 读返回值是 0 0：对应的 ODRy 位不受影响 1：设置对应的 ODRy 位

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

当执行正确的写序列置位 LCKK 时，该寄存器用来锁定端口位的配置。LCKR[7:0]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[7:0]。对相应的端口执行锁定序列后，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。在锁定时序中只允许字访问。

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）

偏移地址：0x1C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
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 锁键值写序列: 写 1->写 0->写 1->读 0->读 1，最后一个读可省略，但可以用来确认锁键已被激活。 注：在操作锁键的写入序列时，不能改变 LCK[7:0]的值。锁键时序的任何错误都会终止锁键被激活。对端口的任何一位首次锁键时序之后，读 LCKK 位都是返回 1，直到系统复位或者 GPIO 复位。
15:8	Reserved	-	-	保留
7:0	LCKy	RW		y = 7..0 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

7.4.9. GPIO 复用功能寄存(GPIOx_AFR) (x=A~C)

偏移地址：0x20

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	AFSEL7[3:0]	RW	4'h0	Px7 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
27:24	AFSEL6[3:0]	RW	4'h0	Px6 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
23:20	AFSEL5[3:0]	RW	4'h0	Px5 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
19:16	AFSEL4[3:0]	RW	4'h0	Px4 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
15:12	AFSEL3[3:0]	RW	4'h0	Px3 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
11:8	AFSEL2[3:0]	RW	4'h0	Px2 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
7:4	AFSEL1[3:0]	RW	4'h0	Px1 配置复用功能 IO 使用方法可参考 AFSEL0[3:0]
3:0	AFSEL0[3:0]	RW	4'h0	Px0 配置复用功能 IO AFSELy 选择: 0000：AF0　1000：AF8 0001：AF1　1001：AF9 0010：AF2　1010：AF10 0011：AF3　1011：AF11 0100：AF4　1100：AF12 0101：AF5　1101：AF13 0110：AF6　1110：AF14 0111：AF7　1111：AF15

7.4.10. GPIO 端口位复位寄存器 (GPIOx_BRR) (x=A~C)

偏移地址：0x28

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	BRy	W	**0**	y =7..0 读返回值是 0 0：对应的 ODy 位受影响 1：清除对应的 ODy 位

8. 系统配置控制器(SYSCFG)

8.1. SYSCFG 主要特性

SYSCFG 模块主要完成如下功能:

- 所有 IO 噪声滤波器控制
- TIMER1 刹车输入控制

8.2. SYSCFG 寄存器

8.2.1. SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)

偏移地址: 0x18

复位值: 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	Res	PVD _LOCK	Res	LOCKUP _LOCK
													RS		RS

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	PVD_LOCK	RS	0	PVD Lock 使能位，它可以被用作使能和锁定 PVD 连接给 TIM1 的刹车输入，也锁定 PWR_CR 寄存器的 PVDE。 软件置位，系统复位清零，软件写 0 到该位不会改变该位的值 0：PVD 中断不与 TIM1 的刹车输入连接。PVDE 位可以被软件写入。 1：PVD 中断与 TIM1 的刹车输入连接。PVDE 位只读。
1	Reserved	-	-	保留
0	LOCKUP_LOCK	RS	0	Cortex-M0+ LOCKUP 使能位，可以使能和锁定 Cortex-M0+的 LOCKUP(hardfault)输出给 TIM1 的刹车输入。 软件置位，系统复位清零，软件写 0 到该位不会改变该位的值 0：Cortex-M0+的 LOCKUP 输出不与 TIM1 的刹车输入连接 1：Cortex-M0+的 LOCKUP 输出与 TIM1 的刹车输入连接

8.2.2. SYSCFG GPIO 滤波使能（SYSCFG_GPI_ENS）

偏移地址：0x1C

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PC_ENS[1:0]	
														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PB_ENS[7:0]								PA_ENS[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
31:18	Reserved	-	-	保留
17:16	PC_ENS[x]	RW	2'h0	PC0~PC1 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能
15:8	PB_ENS[x]	RW	8'h0	PB0~PB7 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能
7:0	PA_ENS[x]	RW	8'h0	PA0~PA7 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能

9. 中断和事件

9.1. 嵌套向量中断控制器(NVIC)

9.1.1. NVIC 主要特性

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

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

9.1.2. 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 6 MHz（最大 f_{HCLK}/8），给出了 1ms 的参考时钟基准。

9.1.3. 中断和异常向量

表 9-1 中断向量表

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	不可屏蔽中断 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	PVD	PVD 中断(PVD 的 EXTI 线是 16)	0x0000_0040
1	8	可设置	Flash	Flash 全局中断	0x0000_0044
2	9	可设置	RCC	RCC 全局中断	0x0000_0048
3	10	可设置	EXTI0_1	EXTI 线[1:0] 中断	0x0000_004C
4	11	可设置	EXTI2_3	EXTI 线[3:2] 中断	0x0000_0050
5	12	可设置	EXTI4_7	EXTI 线[7:4] 中断	0x0000_0054
6	13	可设置	-	保留	0x0000_0058

位置	优先级	优先级类型	名称	说明	地址
7	14	可设置	-	保留	0x0000_005C
8	15	可设置	ADC	ADC 中断	0x0000_0060
9	16	可设置	TIM1_BRK_UP_TRG_COM	TIM1 断开、更新、触发和通信中断	0x0000_0064
10	17	可设置	TIM1_CC	TIM1 捕获/比较中断	0x0000_0068
11	18	可设置	LPTIM1	LPTIM 中断	0x0000_006C
12	19	可设置	TIM14	TIM14 全局中断	0x0000_0070
13	20	可设置	-	保留	0x0000_0074
14	21	可设置	-	保留	0x0000_0078
15	22	可设置	-	保留	0x0000_007C
16	23	可设置	PWM	PWM 全局中断	0x0000_0080
17	24	可设置	UART	UART 全局中断	0x0000_0084
18	25	可设置	-	保留	0x0000_0088

注：地址小于 0x0000_0040 的灰色行是 Cortex®-M0+固有中断。

9.2. 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(line)，管理着 CPU 和系统唤醒功能，并输出下述请求信号：

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

EXTI 唤醒请求允许系统从 Stop 模式唤醒，中断请求和事件请求也可以在正常运行模式使用。

EXTI 允许管理多达 10 个 configurable/direct 事件输入（9 个 configurable 事件输入和 1 个 direct 事件输入）。

9.2.1. EXTI 主要特性

- 系统可以通过 GPIO 和指定模块（LPTIM）输入事件唤醒
- Configurable 型事件（来自 I/O，或无挂起状态位的外设，产生脉冲的外设）
  - 可选有效触发沿（上升沿/下降沿）
  - 中断挂起标志位
  - 独立中断和事件产生屏蔽位
  - 可软件触发
- Direct 型事件（具有中断相关标志和中断挂起状态位的外设）
  - 固定的上升沿触发
  - 在 EXTI 模块里没有中断挂起位
  - 独立中断和事件产生屏蔽位
  - 无软件触发

- IO 端口选择

9.2.2. EXTI 框图

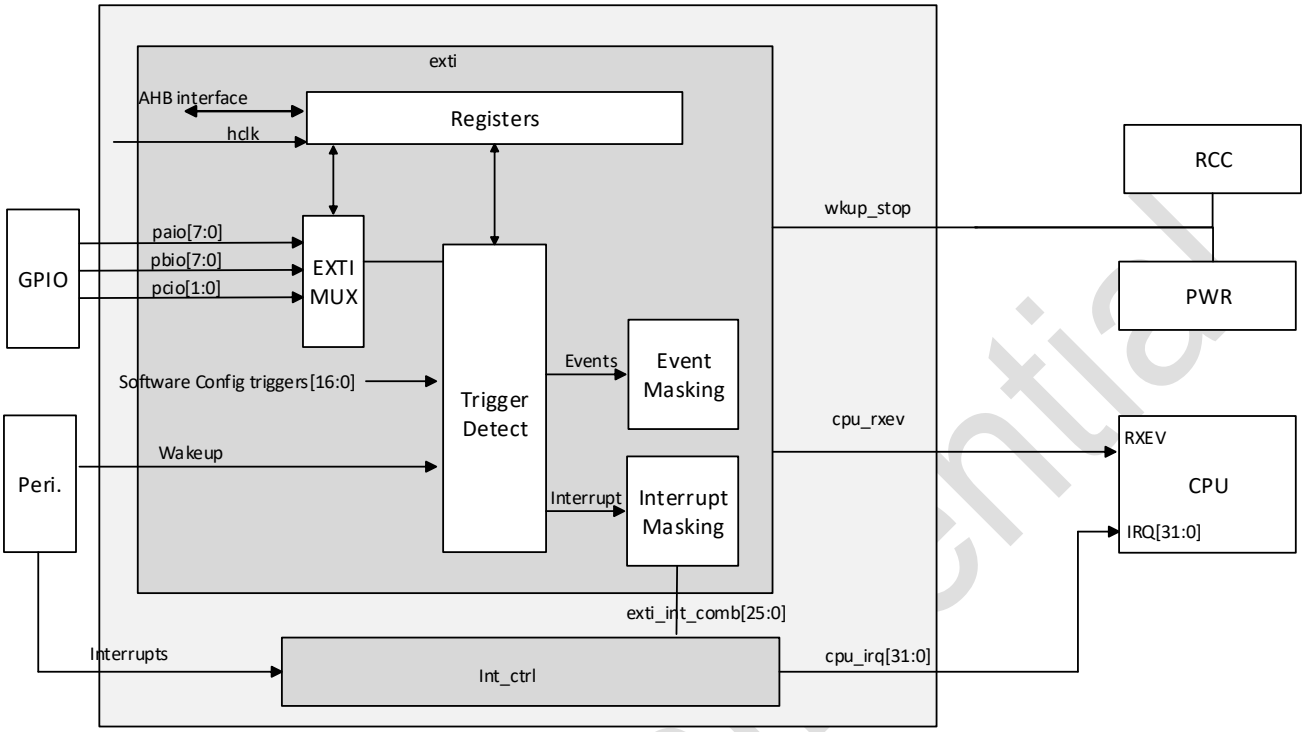


图 9-1 EXTI 框图

9.2.3. EXTI 可配置事件（configurable）触发唤醒

通过配置 EXTI_SWIER 寄存器，软件可以触发唤醒功能。

有对应寄存器配置上升沿或者下降沿触发或者双沿触发 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。所有给 CPU 的事件'或'运算后输出到 CPU 的输入信号 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 寄存器写 1 清零。

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

9.2.4. 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 的事件信号。CPU 在处理该种类型触发事件产生的中断时，软件要清零外设模块的中断状态位。

EXTI 选择器

GPIO 被用以下方式连接到 8 个外部中断/事件输入上：

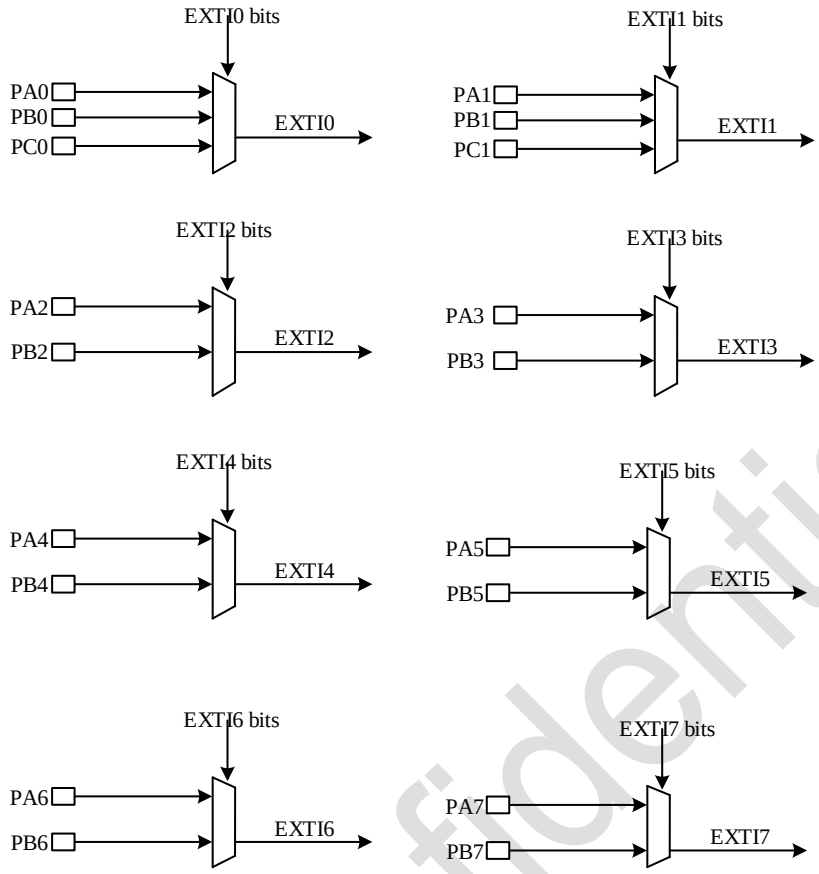


图 9-2 外部中断/事件 GPIO 映像

所有 line 连接内容如下表所示：

表 9-2 EXTI 线连接

EXTI 线	线源	线类型
Line 0-7	GPIO	可配置
Line 8-15	保留	-
Line 16	PVD	可配置
Line 17	保留	-
Line 18	保留	-
Line 19	保留	-
Line 20	保留	-
Line 21	保留	-
Line 22	保留	-
Line 23	保留	-
Line 24	保留	-
Line 25	保留	-
Line 26	保留	-
Line 27	保留	-
Line 28	保留	-

EXTI 线	线源	线类型
Line 29	LPTIM	直接类型

9.3. EXTI 寄存器

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

9.3.1. EXTI 上升沿触发选择寄存器 (EXTI_RTSR)

偏移地址： 0x00

复位值： 0x0000 0000

仅包含对 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	Res	Res	Res	Res	RT16
															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	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	RT16	RW	0	Configurable 类型 EXTI line16 上升沿触发配置。 0：禁止 1：使能
15:8	Reserved	-	-	保留
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。

Bit	Name	R/W	Reset Value	Function
				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 输入是边沿触发的，在这些输入上不能产生毛刺。如果在写 EXTI_RTSR 寄存器期间，Configurable 中断输入出现了上升沿，相关的挂起位不被置位。

在同一个输入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。

9.3.2. EXTI 下降沿触发选择寄存器 (EXTI_FTSR)

偏移地址： 0x04

复位值： 0x0000 0000

仅包含对 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	Res	Res	Res	Res	FT16
															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	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：17	Reserved	-	-	保留
16	FT16	RW	0	Configurable 类型 EXTI line16 下降沿触发配置。 0：禁止 1：使能
15：8	Reserved	-	-	保留
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

Bit	Name	R/W	Reset Value	Function
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。

9.3.3. EXTI 软件中断事件寄存器（EXTI_SWIER)

偏移地址： 0x08

复位值： 0x0000 0000

仅包含对 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	Res	Res	Res	Res	SW16
															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	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：17	Reserved	-	-	保留
16	SWI16	RW	0	Configurable 类型 EXTI line16 软件上升沿触发配置。 0：没有影响 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.

Bit	Name	R/W	Reset Value	Function
15：8	Reserved	-	-	保留
7	SWI7	RW	0	Configurable 类型 EXTI line7 软件上升沿触发配置。 0：没有影响 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：没有影响 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：没有影响 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：没有影响 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：没有影响 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：没有影响 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：没有影响 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：没有影响 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）

9.3.4. EXTI 挂起寄存器(EXTI_PR)

偏移地址： 0x0C

复位值： 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	Res	Res	Res	Res	PR16
															RC_W1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res	Res	Res	Res	Res	Res	Res	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
								RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31：17	Reserved	-	-	保留
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；
15：8	Reserved	-	-	保留
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； 1：产生上升沿/下降沿/软件触发事件请求；

Bit	Name	R/W	Reset Value	Function
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	PR2	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；

9.3.5. EXTI 外部中断选择寄存器 1 (EXTI_EXTICR1)

偏移地址：0x60

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	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	-	-	保留
25:24	EXTI3[1:0]	RW	2'h0	EXTI3 对应 GPIO port 选择。 00：PA[3] pin 01：PB[3] pin 10：保留 11：保留
23:18	Reserved	-	-	保留
17:16	EXTI2[1:0]	RW	2'h0	EXTI2 对应 GPIO port 选择。 00：PA[2] pin 01：PB[2] pin 10：保留 11：保留
15:10	Reserved	-	-	保留
9:8	EXTI1[1:0]	RW	2'h0	EXTI1 对应 GPIO 选择。 00：PA[1] pin 01：PB[1] pin 10：PC[1] pin 11：保留
7:2	Reserved	-	-	保留
1:0	EXTI0[1:0]	RW	2'h0	EXTI0 对应 GPIO 选择。 00：PA[0] pin 01：PB[0] pin 10：PC[0] pin 11：保留

9.3.6. EXTI 外部中断选择寄存器 2 (EXTI_EXTICR2)

偏移地址：0x64

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	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
31:25	Reserved	-	-	保留
25:24	EXTI7[1:0]	RW	2'h0	EXTI7 对应 GPIO port 选择。 00：PA[7] pin 01：PB[7] pin 10：保留 11：保留
23:17	Reserved	-	-	保留
17:16	EXTI6[1:0]	RW	2'h0	EXTI6 对应 GPIO port 选择。 00：PA[6] pin 01：PB[6] pin 10：保留 11：保留
15:9	Reserved	-	-	保留
9:8	EXTI5[1:0]	RW	2'h0	EXTI5 对应 GPIO port 选择。 00：PA[5] pin 01：PB[5] pin 10：保留 11：保留
7:1	Reserved	-	-	保留
0	EXTI4[1:0]	RW	2'h0	EXTI4 对应 GPIO port 选择。 00：PA[4] pin 01：PB[4] pin 10：保留 11：保留

9.3.7. EXTI 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x80

复位值：0x2000 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	Res	Res	Res	Res	IM16
		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	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
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
28:17	Reserved	-	-	保留
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
15:8	Reserved	-	-	保留
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

Bit	Name	R/W	Reset Value	Function
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

9.3.8. EXTI 事件屏蔽寄存器(EXTI_EMR)

偏移地址： 0x84

复位值： 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	Res	Res	Res	Res	EM16
		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	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
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
28:17	Reserved	-	-	保留
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
15:8	Reserved	-	-	保留
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。

Bit	Name	R/W	Reset Value	Function
				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

10. 模拟/数字转换(ADC)

10.1. ADC 简介

芯片具有 1 个 12 位的 SARADC（successive approximation analog-to-digital converter）。该模块共有 10 个要被测量的通道，包括 8 个外部通道和 2 个内部通道。

各通道的转换模式可以设定为单次、连续、非连续模式。转换结果存储在左对齐或者右对齐的 1 个 16 位数据寄存器中。

模拟看门狗允许应用检测是否输入电压超出了用户定义的高或者低阈值。

ADC 实现了在低频率下运行，可获得很低的功耗。

10.2. ADC 主要特性

- 高性能
  - 12bit、10bit、8bit 和 6bit 分辨率可配置
  - ADC 转换时间：0.75MPS@12bit (24MHz)
  - 自校准
  - 可编程的采样时间
  - 可编程的数据对齐模式
  - 支持序列配置
- 低功耗
  - 为低功耗操作，降低 PCLK 频率，而仍然维持合适的 ADC 性能
  - 自动延迟转换模式：防止以低频 PCLK 运行产生溢出
- 模拟输入通道
  - 8 个外部模拟输入通道
  - 1 个内部 temperature sensor 通道
  - 1 个内部参考电压通道（V_{REFINT}）
- 转换操作启动可以通过
  - 软件启动
  - 可配置极性的硬件启动
- 转换模式
  - 单次模式（single mode）：可以转换 1 个单通道或者可以扫描一系列通道
  - 连续模式（continuous mode）：连续转换被选择的通道
  - 非连续模式（discontinuous mode）：每次触发，转换被选择的通道 1 次
- 中断产生
  - 在单个通道采样结束
  - 在单个通道转换结束
  - 在序列转换结束
  - 模拟看门狗事件
  - 溢出事件
- 模拟看门狗

10.3. ADC 功能描述

10.3.1. ADC 框图

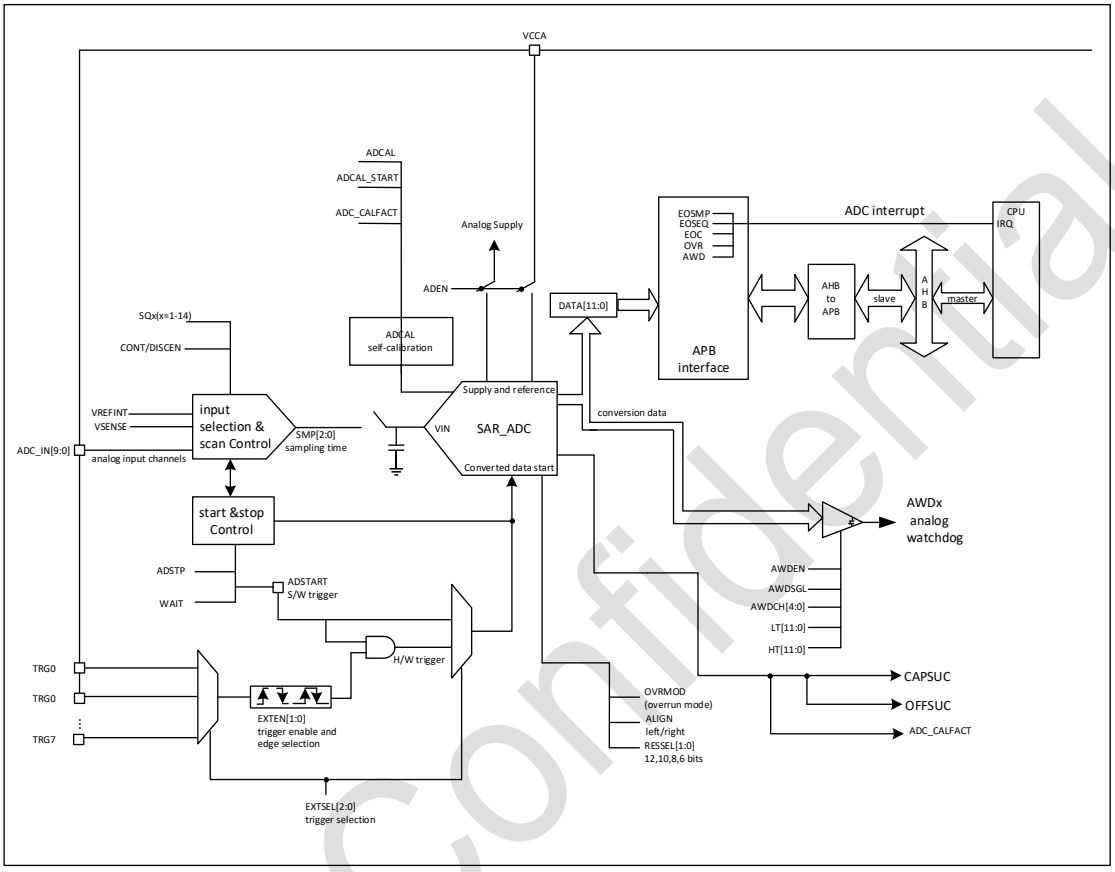


图 10-1 ADC 框图

10.3.2. ADC 校准

该 ADC 具有校准功能（软件启动），用户可通过软件使能 ADCAL 进行校准。在校准期间，ADC 计算一个用于 ADC 内部的校准因子。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，建议用户进行校准操作。校准用于消除芯片和芯片之间的，由于工艺变化引起的失调误差和失配误差。

ADC 软件校准

软件设置 ADCAL=1 使能校准功能，然后开启 ADC CLK，然后软件设置 ADCAL_START 启动校准，校准只能在 ADC 未使能时（ADEN=0）启动，且仅支持选择 PCLK 作为 ADC 的时钟。当校准完成后，在校准结束后，硬件清除 ADCAL。

当 ADC 的工作条件发生改变时（V_{CC} 改变是失调误差和失配误差的主要因素，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：

1.软件写校准目标值（该步骤是可选的：参考软件写校准因子步骤）

2.确认 ADEN=0

3.设置 RSTCAL（该步骤是可选的）；

4.设置校准次数 CALNUM（该步骤是可选的）；

5.设置 ADCAL=1，使能校准

6.设置 ADCAL_START，启动校准

7.等待 ADCAL= 0，则校准结束

8.通过查询 CAL_FAIL 标志位，判断校准成功与否（1：失败；0：成功)

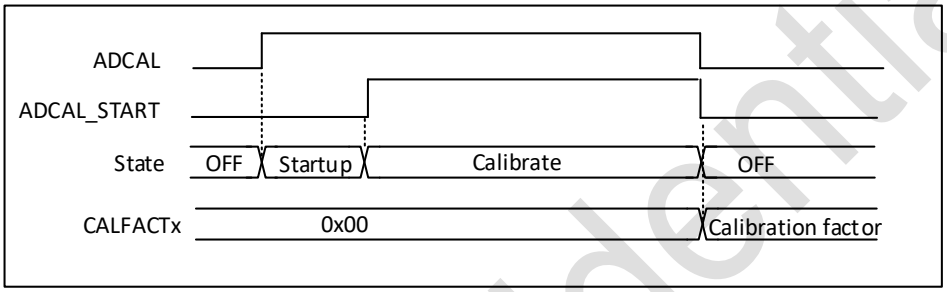


图 10-2 ADC 校准

软件写校准因子

1. 确保ADEN=1，且没有转换正在进行，同时ADCAL=0。
2. 使能WRVLD，FACTSEL[4:0]选择校准控制寄存器，设置WCALFACT [8:0]写入校准配置值。
3. ADEN使能，进行AD转换时，校准因子自动注入模拟ADC。

软件读校准因子

1. 确保 ADEN=1，且没有转换正在进行，同时ADCAL=0。
2. 使能RDVLD，FACTSEL[4:0]选择校准相关寄存器。
3. 读RCALFACT [8:0]校准因子。

10.3.3. ADC 开关控制(ADEN，ADDIS)

芯片上电复位后，ADC 模块不使能，处于不工作状态(ADEN=0)。

ADEN 位用于控制位开启或关闭 ADC。

有两个控制位用于启用或禁用 ADC：

- ADEN=1 时，启用 ADC。
- ADDIS=1 时，禁用 ADC。一旦模拟 ADC 实际上被禁用，硬件会自动清除 ADEN 和 ADDIS。

ADC 规则转换由设置 ADSTART 来启动，根据 EXTEN 的配置，可以立即开始转换（软件启动）或者等待硬件触发后开始转换。

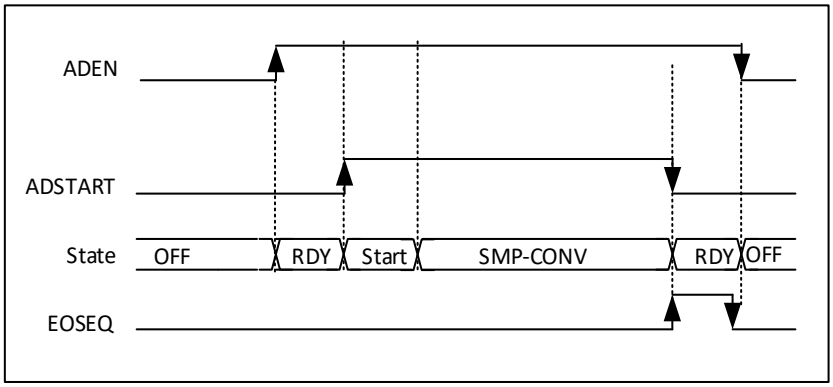


图 10-3 ADC 开关控制

软件禁用 ADC 的步骤

1. 检查 ADSTART 是否均为 0，确保没有正在进行的转换。如有需要，通过设置 ADSTP=1 停止任何正在进行的规则和注入转换，然后等待直到 ADSTP=0。
2. 设置 ADDIS=1 以禁用 ADC。
3. 等待直至 ADEN=0，即模拟 ADC 实际上被禁用（一旦 ADEN=0，ADDIS 将自动重置）。

10.3.4. ADC 时钟

ADC 具有双时钟域架构，ADC 时钟（ADC_CLK）立 APB 时钟（PCLK)。ADC_CLK 可由两种可能的时钟源产生。

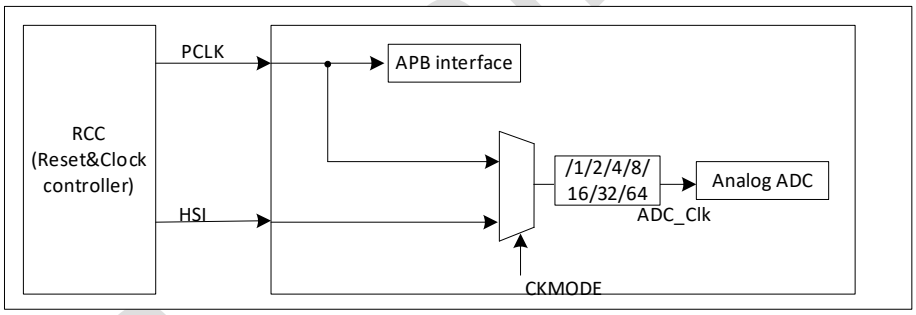


图 10-4 ADC clock scheme

表 10-1 触发器和转换开始之间的延迟

ADC clock source	CKMODE[3:0]	分频系数
PCLK	0000	1
	0001	2
	0010	4
	0011	8
	0100	16
	0101	32
	0110	64
	0111	/
HSI	1000	1
	1001	2

	1010	4
	1011	8
	1100	16
	1101	32
	1110	64
	1111	/

10.3.5. 配置 ADC

软件必须在 ADC 禁止(ADEN 必须为 0) 的情况下改写 ADC_CR 寄存器中的 ADCAL 和 ADEN 位。软件必须在 ADC 开启且没有关闭请求挂起(ADEN=1)的情况下改写 ADC_CR 寄存器中的 ADSTART。

对于以下这些 ADC_IER、ADC_CFGRi、ADC_SMPR、ADC_TR 和 ADC_CCR 寄存器，软件必须在无转换期间（ADSTART = 0)的情况下才能进行改写。ADC_CHSELR 是在 ADEN = 0 且 ADSTART = 0 的情况下改写。

软件必须在 ADC 开启且无挂起请求（ADSTART = 1）的情况下改写 ADC_CR 寄存器中的 ADSTP 位。

10.3.6. ADC 通道选择 (SQx(x=1-11))

共有 10 路复用通道：

- 8 个从 GPIO 引脚引入的模拟输入 (ADC_IN0...ADC_IN7)
- 2 个内部模拟输入(温度传感和内部参考电压)
- 温度传感器连接到 ADC_IN8 通道，
- 内部参考电压连接到 ADC_IN9 通道

ADC 可以转换一个单一通道或自动扫描一个序列通道。

被转换的通道序列必须在通道选择寄存器 ADC_SQRx(x=1-3)中配置

10.3.7. ADC 可编程采样时间 (SMP)

在启动 ADC 转换之前，ADC 需要在被测电压源和内嵌采样电容间建立一个直接连接。采样时间必须足够长以便输入电压源对内嵌电容充电到输入电压的水平。

可编程采样时间根据输入电压的输入阻抗来调整转换速度。

ADC 采样输入电压所用的 ADC 时钟个数可用 ADC_SMPR 寄存器中的 SMP[2:0]位来进行修改。可编程采样时间对所有通道都通用。如有应用需求，则可用软件改变和适应不同通道间的采样时间。

总转换时间计算如下：

t_{CONV} = 采样时间 + (转换分辨率 x2+1.5) x ADC 时钟周期

例如：

当 ADC_CLK = 24 MHz，分辨率为 12 位，且采样时间为 6.5 个 ADC 时钟周期：

t_{CONV} =（6.5+25.5）x ADC 时钟周期= 32 x ADC 时钟周期 = 1.3333 μs

EOSMP 标志位用来表明采样阶段的结束。

10.3.8. ADC 单次转换模式（CONT=0，DISCEN=0）

单次转换模式下，ADC 执行一次序列转换，转换所有被选的通道。当 ADC_CFGR1 寄存器中的 CONT=0，DISCEN=0 时，ADC 为单次转换模式。

ADC 转换可由下述两种方法启动：

- 在 ADC_CR 寄存器中设置 ADSTART 位
- 硬件触发事件

在序列通道的转换期间，每次转换完成后：

- 转换的数据结果存放到 SQx 对应的 16 位寄存器 SQx_DATA 中。
- EOC(转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断。
- 通道序列转换完成后：
- EOSEQ（序列结束）标志置位
- 若 EOSIE 位置位则产生一个中断

转换结束后，ADC 停止直到新的触发事件或 ADSTART 重新置位。

注：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

10.3.9. 连续转换模式（CONT=1)

在连续转换模式中，当软件或硬件触发事件产生，ADC 执行一个序列转换。转换所有的通道一次且自动重新开始执行相同的序列转换。当寄存器 ADC_CFGR1 中的 CONT=1 时，ADC 选择为连续转换模式。ADC 转换可由下述两种方法启动：

- 在 ADC_CR 寄存器中设置 ADSTART 位
- 硬件触发事件

在序列通道的转换期间，每次转换完成后：

- 转换的数据结果存放到 SQx 对应的 16 位寄存器 SQx_DATA 中。
- EOC (转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断。

通道序列转换完成后：

- EOSEQ(序列结束)标志置位
- 若 EOSEQIE 位置位则产生一个中断

一次序列转换结束后，ADC 立即重新转换相同的序列通道。

注：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

ADC 不能同时处于 discontinuous 转换模式和 continuous 转换模式，在这种情况下 (DISCEN=1，CONT=1)，其表现为单次转换模式。

10.3.10. 非连续转换模式（DISCEN=1)

该模式由设置 ADC_CFGR1 寄存器中的 DISCEN 位来开启。

在这个模式 (DISCEN=1)下，需要硬件或软件的触发事件去启动定义在一个序列中的每次转换。

相反，DISCEN=0 时，一个硬件或软件的触发事件，就可以启动定义在一个序列中的所有转换。

例如：

DISCEN=1，需要转换的通道为：0、3、7、8

- 1st 触发：通道 0 被转换且一个 EOC 事件产生
- 2nd 触发：通道 3 被转换且一个 EOC 事件产生
- 3rd 触发：通道 7 被转换且一个 EOC 事件产生
- 4th 触发：通道 8 被转换且产生 EOC 和 EOSEQ 事件
- 5th 触发：通道 0 被转换且一个 EOC 事件产生
- 6th 触发：通道 3 被转换且一个 EOC 事件产生
- ...

DISCEN=0，需要转换的通道为：0、3、7、8

- 1st 触发：整个完整的序列转换，依次为通道 0、3、7 和 8。

每次转换完成，产生一个 EOC 事件，转换到最后一个通道，除产生 EOC 外，还产生一个 EOSEQ 事件。

- 任何触发事件都会重新开始完整的序列转换。

注：让 ADC 同时处于连续模式和连续转换模式是不可能的事情，在这种情况下(DISCEN =1，CONT=1)，其表现为单次转换模式。

10.3.11. 启动 ADC 转换 (ADSTART)

软件用设置 ADSTART=1 启动 ADC 转换。

当 ADSTART 设置，则转换：

- 当 EXTEN=0x0(软件触发) 时，立即开始
- 当 if EXTEN ≠ 0x0 时，在下一个所选择的硬件触发有效边沿开始

ADSTART 位也用于说明目前 ADC 转换操作是否正在进行。当 ADSTART=0 时，可重新配置 ADC，说明此时 ADC 处于空闲。

ADSTART 位可由硬件清除。

- 单次转换模式由软件触发 (CONT=0，EXTSEL=0x0)
  - 在序列转换结束后 (EOSEQ=1)
- Discontinuous 转换模式由软件触发 (CONT=0，DISCEN=1，EXTSEL=0x0)
  - 在转换结束后(EOC=1)
- 在所有的情况下(CONT=X，EXTSEL=X)
  - 在软件调用并执行 ADSTP 过程后

注：在连续模式 (CONT=1) 下，ADSTART 位不能由 EOSEQ 引发的硬件清除，其原因是自动重新开始序列转换。当硬件触发选择为单次转换模式 (CONT=0 and EXTSEL =0x01)，则当 EOSEQ 标志设置后，ADSTART 不会被硬件清 0。这就避免了需要软件重新设置 ADSTART 位且要确保无硬件触发事件错过。

10.3.12. 转换时间

转换所用的时间由启动转换时间和与转换分辨率有关的逐次逼近时间组成。

$t_{ADC} = t_{SMPL} + t_{SAR} = [\ 6.5_{|min} + 25.5_{|12bit}] * t_{ADC\_CLK}$

$t_{ADC} = t_{SMPL} + t_{SAR} = \ 270.83ns_{|min} + 1062.5\ ns_{|12bit} = 1.3333\ \mu s_{|min}$ (for f_{ADC_CLK} = 24 MHz)

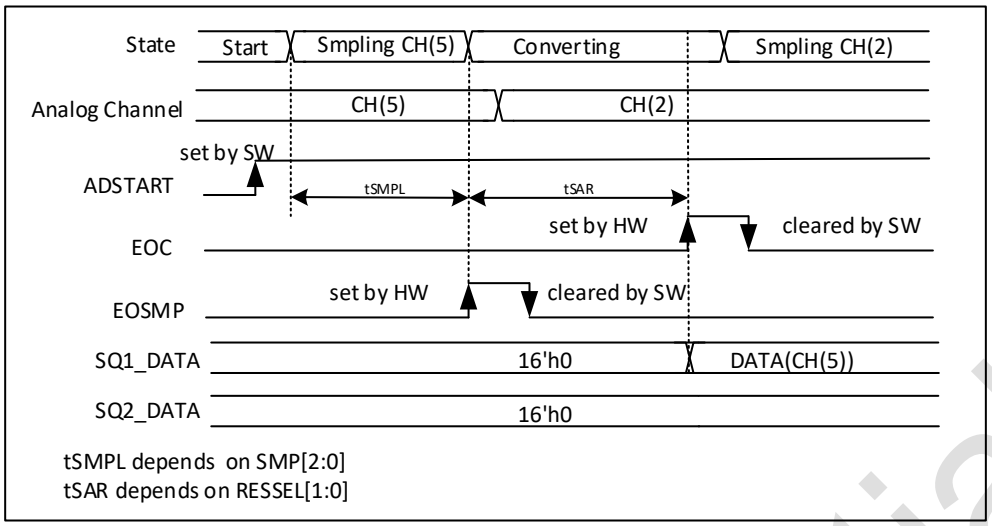


图 10-5 AD 转换时序

10.3.13. 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。

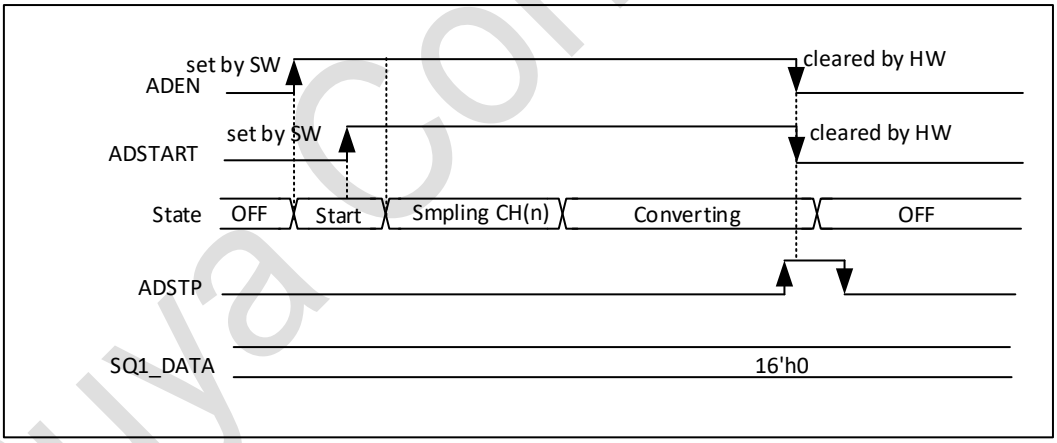


图 10-6 ADSTP 时序

10.3.14. 外部触发转换和触发极性(EXTSEL，EXTEN)

一次转换或一个序列的转换可由软件或外部事件（例如：定时器捕、输入引脚）触发。若 EXTEN[1:0] ≠ “00”，则外部事件在其所选择的极性上可以用于触发转换。当软件设置 ADSTART=1 时，触发选择有效。

当正在进行 ADC 转换时，任何硬件触发都会被忽略。

当 ADSTART=0 时，任何硬件触发都会忽略。

Source	EXTEN[1:0]
触发检测禁止	00
在上升沿检测	01
在下降沿检测	10
在上升和下降沿检测	11

注：在转换时外部触发极性不能改变。EXTSEL[2:0]控制位用于选择可触发转换的事件。

下表给出了规则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 10-2 外部触发

Name	Source	EXTSEL[2:0]
TRG0	TIM1_TRGO	000
TRG1	TIM1_CC4	001
TRG2	TIM1_CC1	010
TRG3	保留	011
TRG4	TIM14_CC1	100
TRG5	PWM_CC1	101
TRG6	保留	110
TRG7	EXTI LINE 1	111

注： 在转换时外部触发源不能改变。

10.3.15. 快速转换模式

用降低转换分辨率来获取更快的转换时间 (tSAR) 是可行的。转换分辨率可通过设置 ADC_CFGR1 寄存器中的 RESSEL[1:0]来配置为 12/10/8/6 位模式。当应用不需要高精度数据时，可用低的转换分辨率来加快转换时间。转换结果也是 12 位宽度且低位补 0。

分辨率模式减少逐次逼近的转换时间，如下表所示：

RESSEL [1:0]	t_{SAR} (ADC 时钟周期)	t_{SAR}(ns) @ f_{ADC} = 24MHz	t_{SMP} (ADC 时钟周期)	t_{ADC}(t_{SMP} = 7) (ADC 时钟周期)	t_{CONV}(ns) @ f_{ADC} = 24MHz
12	25.5	1062ns	7	32	13333ns
10	21.5	896ns	7	28	1166ns
8	17.5	728ns	7	24	1000ns
6	13.5	562ns	7	20	834ns

10.3.16. 转换结束/采样结束

ADC 通知应用每次转换结束（EOC）事件。

一旦在 ADC_DR 寄存器中的一个转换数据有效后，ADC 在 ADC_ISR 寄存器中设置 EOC 标志表明转换完成。当 ADC_IER 中的 EOCIE 置为 1 时，则会产生一个 EOC 中断。EOC 标志由软件写 1 清除或读 ADC_DRx 寄存器来清除。

ADC 同样在 ADC_ISR 寄存器中给出采样阶段结束标志 EOSMP。EOSMP 标志可写 1 请除。当在 ADC_IER 寄存器中的 EOSMPIE 置为 1 后，则会产生一个 EOSMP 中断。

10.3.17. 序列转换结束（EOSEQ flag）

ADC 通知应用每次序列转换结束（EOSEQ)事件。

一旦一个转换序列的最后一个通道转换数据有效后，ADC 在 ADC_ISR 寄存器中设置 EOSEQ 标志。当 ADC_IER 中的 EOSEQIE 位置 1 时，则会产生中断。EOSEQ 标志由软件写 1 清 0。

10.3.18. 采样时间图

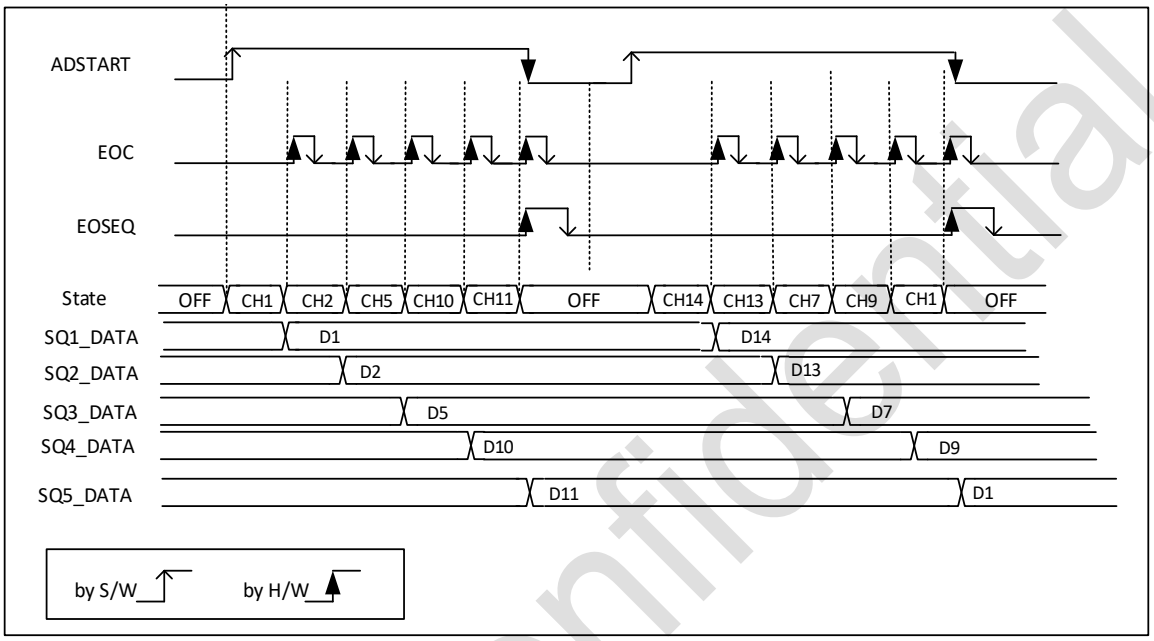


图 10-7 序列的单次转换，软件触发

1. EXTEN=0x0，CONT=0，WAIT=0

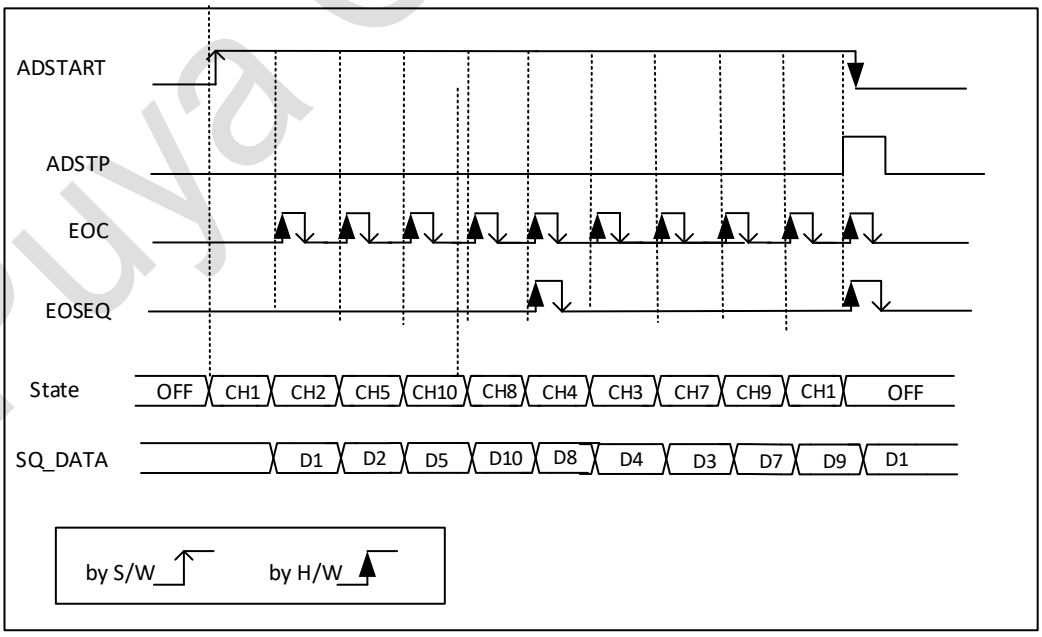


图 10-8 序列的连续转换，软件触发

1. EXTEN=0x0，CONT=1，WAIT=0

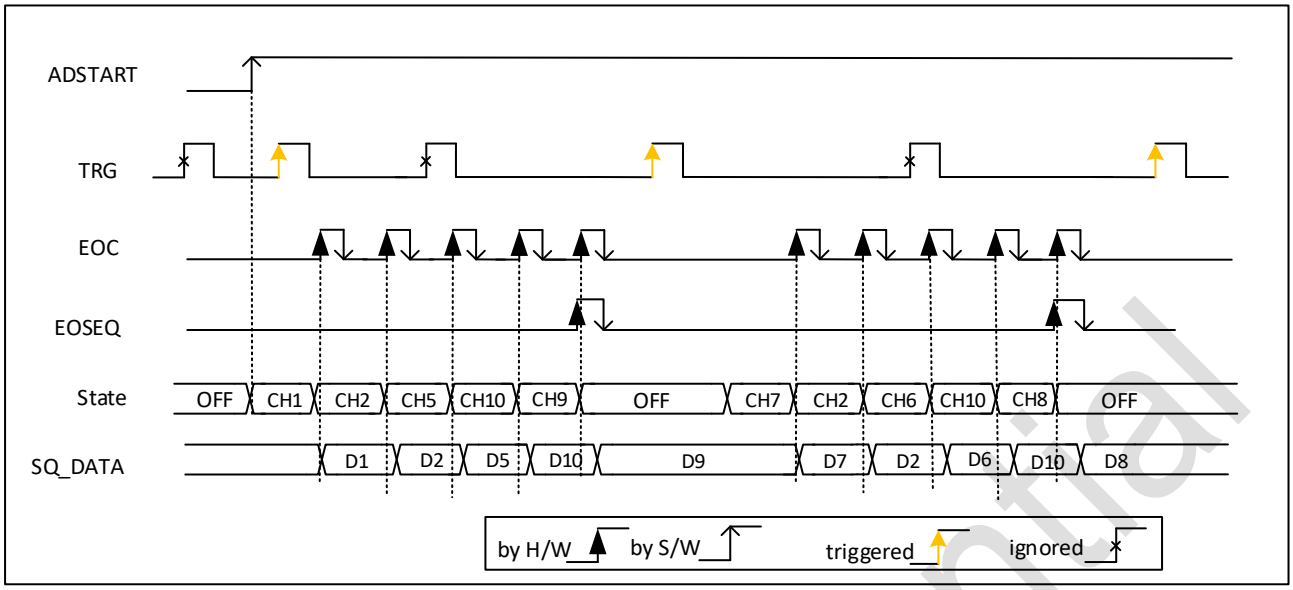


图 10-9 序列的单次转换，硬件触发

1. EXTSEL=TRGx，EXTEN=0x1（上升沿），CONT=0

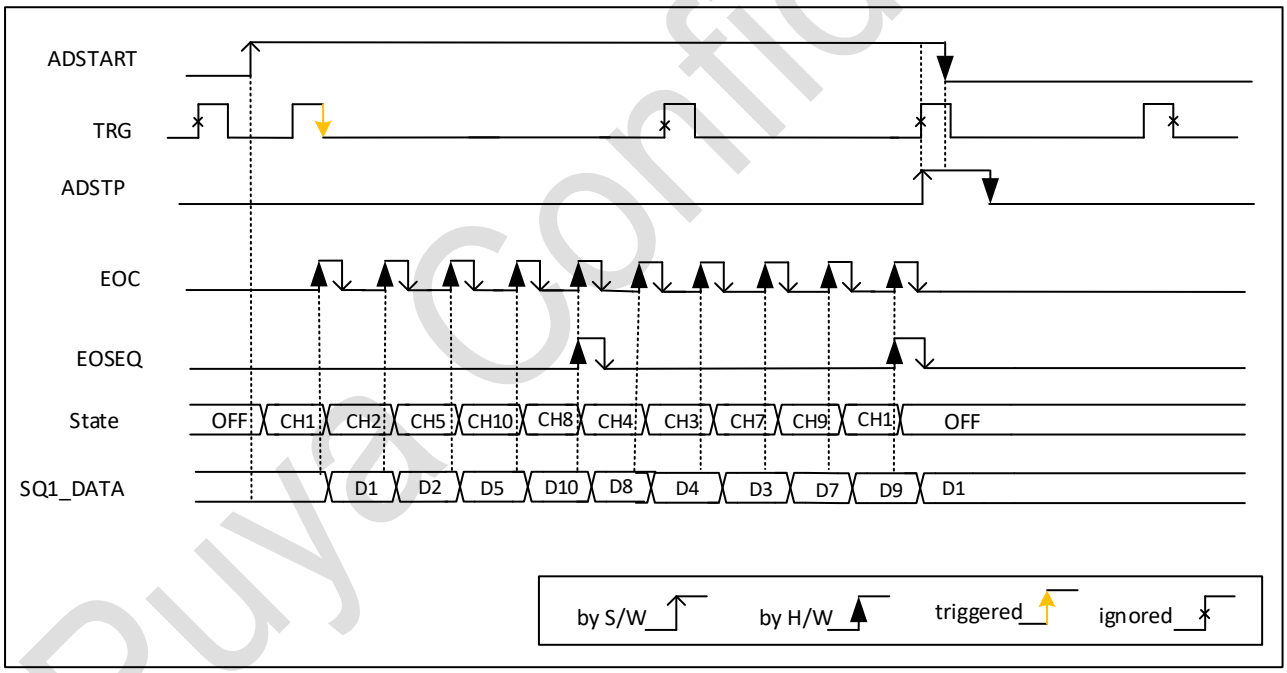


图 10-10 序列的连续转换，硬件触发

1. EXTSEL=TRGx，EXTEN=0x2（下降沿），CONT=1，WAIT=0

10.3.19. 数据管理

数据寄存器和数据对齐(ADC_DRx，ALIGN)

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 SQx 对应的 16 位寄存器 SQx_DATA 中。

ADC_DRx 数据格式与所配置的数据对齐和转换分辨率有关。ADC_CFGR1 寄存器中的 ALIGN 位用于选择数据存储的对齐方式，数据可选为右对齐（ALIGN=0）或左对齐（ALIGN=1)。

ALIGN	RESSEL	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0X0	0X0				DATA[11:0]											
	0X1	0X0				DATA[9:0]										0X0	
	0X2	0X0				DATA[7:0]								0x0			
	0X3	0X0				DATA[6:0]						0X0					
1	0X0	DATA[11:0]												0X0			
	0X1	DATA[9:0]										0X0		0X0			
	0X2	DATA[7:0]								0x0				0X0			
	0X3	DATA[6:0]						0X0						0X0			

ADC 过载（OVR，OVRMOD)

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

若 EOS 还为‘1’的情况下，这时一个新的转换已经完成，那么 CPU 就会在 ADC_ISR 寄存器中的 OVR 标志被置位，表明 ADC 过冲。当 ADC_IER 寄存器中的 OVRIE 置位时，产生一个 ADC 过冲中断。

当过冲事件发生时，ADC 会继续操作并且继续转换除非软件决定停止并复位这个序列转换，可用软件设置 ADC_CR 寄存器中的 ADSTP 为 1 来停止 ADC 转换 OVR 标志可用软件写 1 清除。

当发生过冲事件时，可通过对 ADC_CFGR1 寄存器中的 OVRMOD 位来设置 ADC 数据寄存器中的数据是被保持还是被覆盖：

- OVRMOD=0

- 一个过冲事件保持数据寄存器的值防止被覆盖：之前的数据被保持，新的转换数据丢弃。若 OVR 保持为 1，则后续的转换会被执行但结果都被丢弃。

- OVRMOD=1

- 用最近一次的转换结果覆盖数据寄存器，先前未读的数据丢失。若 OVR 保持为 1，则后续的转换被执行且 ADC_DRx 寄存器存放着最新转换的结果值。

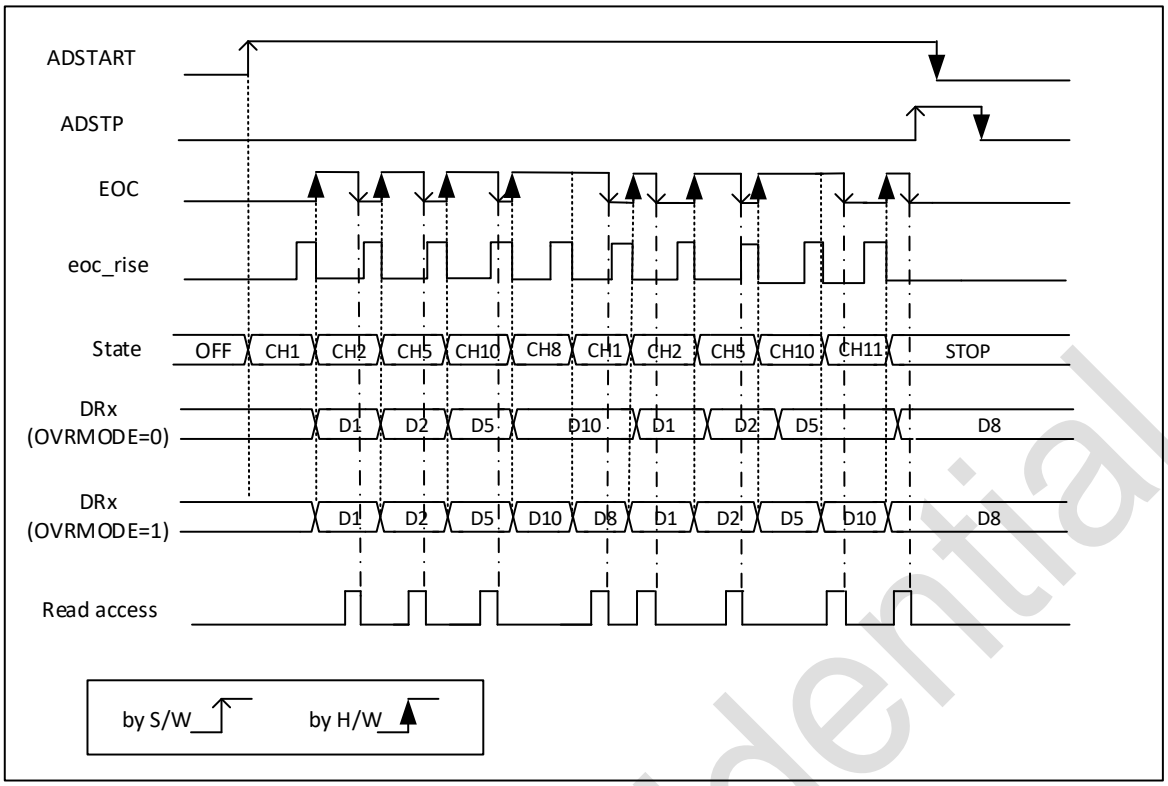


图 10-11 过载

若 ADC 的转换足够慢，转换序列可由软件来控制。这种情况下，软件应用 EOS 标志及其关联的中断去处理每个转换数据。当每次转换结束时，在 ADC_ISR 寄存器中的 EOC 位置位，此时可读 ADC_DRx 寄存器的转换值。ADC_CFGR1 寄存器中的 OVRMOD 位可配为 0 来管理过冲事件。

存在着转换一个或多个通道且不用每次转换结果都要读取的应用。这种情况下，OVRMOD 位必须置为 1 且软件应忽略 OVR 标志。当 OVRMOD=1 时，过冲事件不能阻止 ADC 继续转换且 ADC_DRx 寄存器中的数据一直为最后转换的数据。

10.3.20. 低功耗特性

自动延迟转换模式

自动延迟转换模式可用于在低速运行时简化软件以及优化应用程序的性能，当然在这种模式下不容易产生 ADC 过冲的情况。

当在 ADC_CFGR1 寄存器中设置 WAIT 为 1 时，一个新的转换只有在刚才的 ADC 数据处理完后(比如 ADC_DR x 寄存器中的数据被读取或 EOS 标志已被清除)才开始。这是一种自适应 ADC 速度和自适应系统读取 ADC 数据速度的方法。

注：当正在转换中或自动延迟产生的情况下，任一硬件产生的触发都会被忽略。

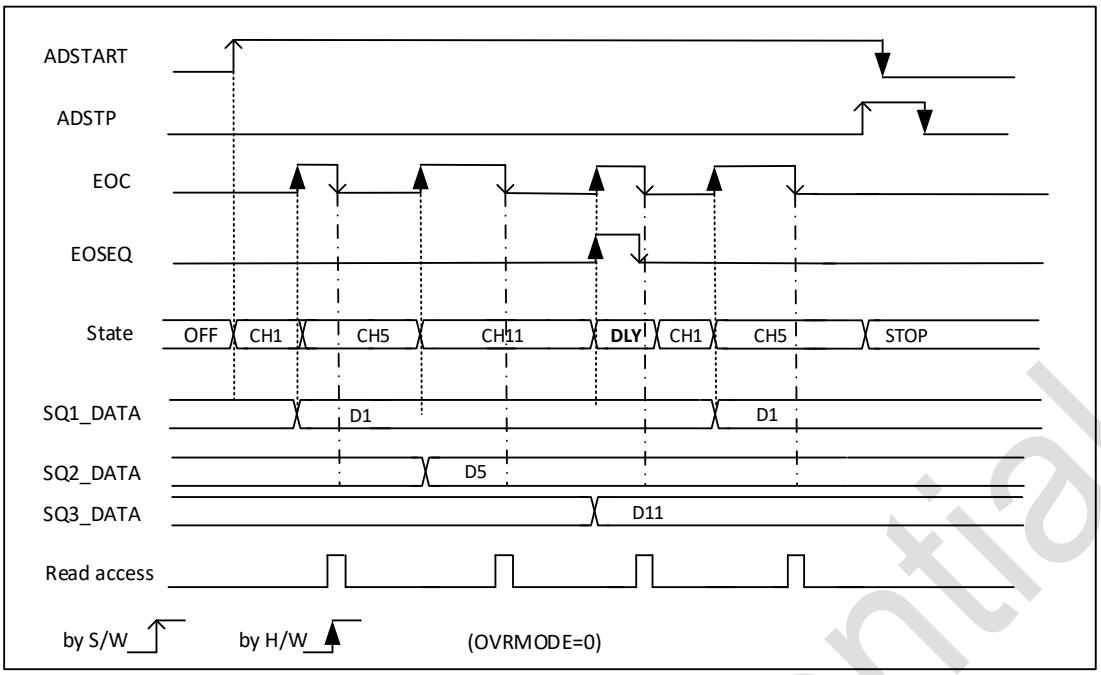


图 10-12 自动延迟转换模式

1. EXTEN=0x0，CONT=1

10.3.21. 模拟看门狗

模拟看门狗的功能由在 ADC_CFGR1 寄存器中的 AWDEN 位置位来开启。它可用于监控所选的单一通道或所有使能通道所配置电压范围(窗口)。

如果模拟电压转换由 ADC 低于低阈值或高于高阈值时，AWD 模拟看门狗的状态位被置位。阈值被编程到最多具有 12 位有效数据的 ADC_HTR 和 ADC_LTR 16 位寄存器中。模拟看门狗中断可用设置 ADC_IER 寄存器中的 AWDIE 位来使能。AWD 标志位可用软件写 1 来清除。当转换的数据分辨率小于 12 位（由 RESSEL[1:0]位来决定）被编程阀值的低位必须保持清零，因为内部转换数据的比较都是按左对齐全 12 位的方式进行比较。

表 10-3 模拟看门狗比较

分辨率位	模拟看门狗比较:		说明
	原始转换数据，左对齐	阈值	
00：12-bit	DATA[11:0]	LT[11:0] and HT[11:0]	
01：10-bit	DATA[11:2]，00	LT[11:0] and HT[11:0]	用户必须配置 LT[1:0]和 HT[1:0]为 00
10：8-bit	DATA[11:4]，0000	LT[11:0] and HT[11:0]	用户必须配置 LT[3:0]和 HT[3:0]为 0000
11：6-bit	DATA[11:6]，000000	LT[11:0] and HT[11:0]	用户必须配置 LT[5:0]和 HT[5:0]为 000000

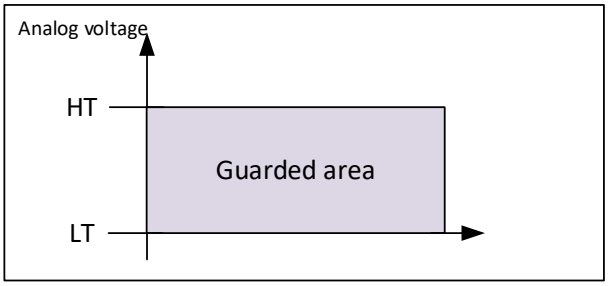


图 10-13 模拟看门狗保护区

表 10-4 模拟看门狗通道选择

看门狗监测通道	AWDSGL 位	AWDEN 位
无	x	0
All channels	0	1
Single channel	1	1

ADC_AWD_OUT 信号输出产生

模拟看门狗与一个内部硬件信号相关联，ADC_AWD_OUT 直接连接到片上定时器 TIM1 的 ETR 输入（外部触发）。

启用模拟看门狗时，将激活 ADC_AWD_OUT：

- 当经过 AWDCH 选择的通道转换超出程序阈值时，将设置 ADC_AWD_OUT。
- 在下一个经过 AWDCH 选择的通道的转换结束之后，ADC_AWD_OUT 在编程的阈值之内复位。如果下一个受保护的转换仍超出编程的阈值，则它将保持为 1。
- 禁用 ADC 时（将 ADDIS 设置为 1 时）ADC_AWD_OUT 也会复位。请注意，停止转换（ADSTP 设置为 1）可能会清除 ADC_AWDx_OUT 状态。
- 未选择为模拟看门狗的通道，不影响 ADC_AWD_OUT 状态位。

AWD 标志由硬件设置并由软件复位：AWD 标志对 ADC_AWD_OUT 的生成没有影响（例如，如果软件未清除该标志，则 ADC_AWDx_OUT 可以切换，而 AWDx 标志保持为 1）。

ADC_AWD_OUT 信号由 PCLK 域生成。

AWD 比较在每次 ADC 转换结束时执行。

10.3.22. 温度传感器和内部参考电压

温度传感器可以用来测量器件的接点温度（T_J）。

温度传感器内部连接到 ADC 输入通道，可用于转换传感器的电压值到一个数值。温度传感器的采样时间必须大于数据手册给出的 Ts_temp 的最小值。当温度传感器没被使用时，传感器可以置于断电模式。

温度传感器输出电压跟温度成线性变化关系，但是跟工艺变量有关每颗芯片会有细微差别。为了提高这个准确度，每一颗的校准值会被产品测试单独给出并且保存在系统存储区域。

内部电压参考（V_{REFINT}）提供一个稳定电压输出给 ADC。

注：必须设置 TSEN 和 VREFN 位来激活两个内部通道：温度传感器、V_{REFINT}。

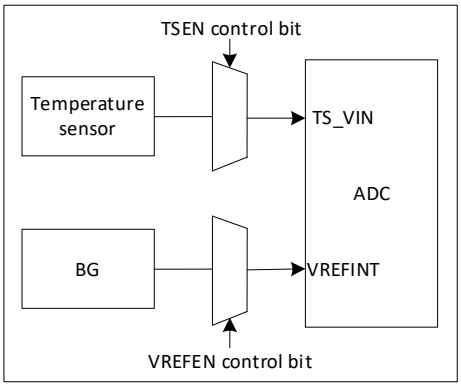


图 10-14 T_S and V_{REFINT} channel

读温度

如何用温度传感器：

1. 选择 ADC_IN8 输入通道
2. 根据器件的规格书选择一个合适的采样时间
3. 在 ADC_CCR 寄存器中设置 TSEN 位用来唤醒从断电模式下的温度传感器
4. 用设置在 ADC_CR 寄存器中的 ADSTART 位（也可用外部触发）来启动 ADC 转换
5. 从 ADC_DR 寄存器中读取 VSENSE 转换数据
6. 用下列公式计数温度：

$$Temperature(in\ ℃) = \frac{105℃ - 30℃}{TS_{CAL2} - TS_{CAL1}} \times (TS_{DATA} - TS_{CAL1}) + 30℃$$

TSCAL2 代表 105℃温度传感器的校准值，校准值存放地址：0x1FFF 0098

TSCAL1 代表 30℃温度传感器的校准值，校准值存放地址：0x1FFF 0094

TSDATA 是 ADC 转换的实际输出值

注：传感器从断电模式下唤醒时到能正确输出 VSENSE 要有一个启动时间，ADC 从上电后启动也有一个启动时间， 若要减少这个延时， 则需要同时设置 ADEN 和 TSEN 位。

利用内部的参考电压计算实际的 Vcc 电压

$$VREFINT = 1.2V = \frac{ADC\_DATAx}{4095} \times VCC$$

利用 V_{CC} 电压来计算 Vchannnel

$$VCHANNEL = \frac{ADC\_DATAx}{4095} \times VCC$$

V_{REFINT} 固定值为 1.2V；

VCHANNEL 是通道电压；

ADC_DATA 是 ADC_DRx 里面的转换数据；

4096 表示为 12 位。

微控制的 VCC 电源容易受影响或者不是很明确该值大小。内部电压参考（VREFINT）以及在生产过程中 VCC=3.3V ADC 获取的校准数据可以用来评估出真实的 VCC 的电压水平。

10.3.23. ADC 中断

ADC 中断可由以下任一事件产生：

- 任何一次的转换结束（EOC 标志）
- 序列转换结束（EOS 标志)
- 当模拟看门狗检测发生（AWD 标志)
- 当采样阶段结束发生（EOSMP 标志)
- 当数据过冲发生（OVR 标志)
- ADC 上电结束标志（ADRDY 标志)
- 半序列转换结束标志（EOH 标志)

独自的中断使能位用于灵活设置 ADC 中断

表 10-5 ADC 中断

中断事件	事件标志	使能控制
转换结束	EOC	EOCIE
序列转换结束	EOS	EOSIE
模拟看门狗状态置位	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
过冲	OVR	OVRIE
ADC 上电结束标志	ADRDY	ADRDYIE
半序列转换结束标志	EOH	EOHIE

10.4. ADC 寄存器

10.4.1. ADC 中断和状态寄存器（ADC_ISR)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	EOH	AWD	Res	Res	OVR	EOSEQ	EOC	EOSMP	Res
							RC_W1	RC_W1			RC_W1	RC_W1	RC_W1	RC_W1	

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	EOH	RC_W1	0	半序列结束标志 SQx 位选择的一半序列转换结束时硬件置位该位。软件写 1 清 0 0：一半转换序列没有完成（或者软件已经应答和清除该标志） 1：一半转换序列完成

Bit	Name	R/W	Reset Value	Function
				注意：半序列结束是基于序列长度 len/2 后四舍五入得到的。
7	AWD	RC_W1	0	模拟看门狗 当转换电压值超过 ADC_TR.HT 或者小于 ADC_TR.LT 编程的值时硬件置位。软件写 1 清零。 0：无模拟看门狗事件发生（或者软件已清除该事件标志） 1：模拟看门狗事件发生
6:5	Reserved	-	-	保留
4	OVR	RC_W1	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。该位写 1 清 0 0：无过载发生（或软件已应答和清除该位） 1：过载已发生
3	EOSEQ	RC_W1	0	序列结束标志 SQx 位选择的序列转换结束时硬件置位该位。软件写 1 清 0 0：转换序列没有完成（或者软件已经应答和清除该标志） 1：转换序列完成
2	EOC	RC_W1	0	转换结束标志 当每个通道每次转换结果后新的数据结果可以从 ADC_DR 寄存器读到时，硬件置位该位。软件写 1 清 0 或读 ADC_DR 寄存器清 0 0：通道转换没有完成（或者软件已经应答和清除该标志） 1：通道转换已完成
1	EOSMP	RC_W1	0	采样结束标志，在每次转换的采样阶段结束时，硬件置位该位，软件写 1 清 0 0：不处在采样阶段结束时（或者软件已经应答和清除该标志） 1：采样阶段结束
0	Reserved	-	-	保留

10.4.2. ADC 中断使能寄存器（ADC_IER)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	EO-HIE	AWDIE	Res	Res	OVRIE	EOSE-QIE	EO-CIE	EOSMPIE	Res
							RW	RW			RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	EOHIE	RW	0	半序列中断使能位 软件清除或置起半序列中断 0：半序列中断不使能 1：半序列中断使能
7	AWDIE	RW	0	模拟看门狗中断使能位 软件清除或置起模拟看门狗中断 0：模拟看门狗中断不使能 1：模拟看门狗中断使能
6:5	Reserved	-	-	保留
4	OVRIE	RW	0	ADC 过载中断使能位 软件清除或置起过载中断使能 0：ADC 过载中断不使能 1：ADC 过载中断使能
3	EOSEQIE	RW	0	序列结束中断使能位 软件清除或置起序列结束中断使能 0：序列结束中断不使能 1：序列结束中断使能
2	EOCIE	RW	0	转换结束中断使能位 软件清除或置起转换结束中断使能位 0：转换结束中断不使能 1：转换结束中断使能
1	EOSMPIE	RW	0	采样标志结束中断使能位 软件清除或置起转换采样标志结束中断位 0：采样标志结束中断不使能 1：采样标志结束中断使能
0	Reserved	-	-	保留

说明：当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写这些位

10.4.3. ADC 控制寄存器（ADC_CR)

偏移地址：0x08

复位值：0x0000 0000

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

Res	Res	Res	Res	Res	Res	Res	Res	Res		Res	AD-STP	Res	AD-START	AD-DIS	ADEN
											RS		RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RW	0	ADC 校准启动，软件使能 ADC 校准;
30	ADCAL_START	RW	0	0：没有正在进行的 ADC 校准 1：写 1 启动 ADC 校准，校准结束 硬件清 0.
29	RSTCAL	RS	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。
28:5	Reserved	-	-	保留
4	ADSTP	RS	0	ADC 停止转换命令 软件置位停止和丢弃正在进行的转换（ADSTP 命令） 当转换被丢弃并且准备接受新的转换命令时硬件回清除该位 0：没有正在进行的 ADC 停止转换命令 1：写 1 停止 ADC，读为 1 表明一个 ADSTP 命令正在进行中。
3	Reserved	-	-	保留
2	ADSTART	RS	0	ADC 启动命令 软件置位该位启动 ADC 转换。根据 EXTEN[1：0]的配置来决定转换是软件立即启动，还是由硬件触发事件来启动.该位由硬件清除： – 在单次转换模式（CONT=0，DISCEN=0），选择软件驱动时(EXTEN=00)：转换完成标志结束时（EOSEQ 标志） – 在非连续转换模式（CONT=0，DISCEN=1），当软件驱动时(EXTEN=00)：转换结束标志（EOC） – 其他情况下：执行 ADSTP 命令之后，同时 ADSTP 标志又被硬件清 0 之时 0：没有正在进行的 ADC 转换

Bit	Name	R/W	Reset Value	Function
				1：写 1 启动 ADC，读为 1 表明 ADC 正在操作可能正在转换。 注意：软件只能在 ADEN=1 且 ADDIS=0 时置位 ADSTART
1	ADDIS	RS		ADEN 禁止使能 软件置位禁止 ADC 并且 ADC 进入掉电。硬件清除该位，当 ADC 被禁止（ADEN 被硬件清零同时） 0：没有 ADDIS 1：写 1 禁止 ADC，读 1 表示 ADDIS 指令正在执行 注意：设置 ADDIS 为 1 有效只能在 ADEN=1 并且 ADSTART=0 时（确保没有转换进行）
0	ADEN	RS	0	ADC 使能命令 软件置位该位使能 ADC，ADC 将准备操作。 0：禁用 ADC 1：使能 ADC

10.4.4. ADC 配置寄存器 1 (ADC_CFGR1)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	AWDCH				Res	Res	AWDEN	AWDSGL	Res	Res	CALNUM			DIS-CEN
		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	WAIT	CONT	OV-RMOD	EXTEN[1:0]		Res	EXTSEL			ALIGN	RESSEL		Res	Res	Res
	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:26	AWDCH[3:0]	RW	4’h0	模拟看门狗通道选择，软件可清除和设置该位。 模拟看门狗监测选择的输入通道 0000：ADC 模拟输入通道 0 0001：ADC 模拟输入通道 1 0010：ADC 模拟输入通道 2 ….

Bit	Name	R/W	Reset Value	Function
				1001：ADC 模拟输入通道 9 其他值：保留位 说明：AWDCH[3:0] 位配置的通道也需要设置到 ADC_SQRx(x=1-3)寄存器里 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
25：24	Reserved	-	-	保留
23	AWDEN	RW	0	模拟看门狗使能 软件可设置和清除该位 0：不使能模拟看门狗 1：使能看门狗 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
22	AWDSGL	RW	0	在一个通道或者所有通道使能模拟看门狗 软件可设置和清除该位取使能模拟看门狗在 AWDCH[3：0]位设置的通道上或者所有通道 0：在所有通道上使能模拟看门狗 1：在一个通道上使能模拟看门狗 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
21：20	Reserved	-	-	保留
19:17	CALNUM	RW	3’h0	校准平均次数 0xx：1 次校准和平均 100：4 次校准和平均 101：8 次校准和平均 110：16 次校准和平均 111：32 次校准和平均
16	DISCEN	RW	0	非连续模式 软件可设置和清除该位，使能/不使能非连续模式 0：不使能非连续模式 1：使能非连续模式 不可能既使能非连续模式又使能连续模式；禁止设置 DISCEN=1 和 CONT=1. 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
15	Reserved	-	-	保留
14	WAIT	RW	0	自动延迟转换模式 软件可设置和清除该位，使能/禁止自动延迟转换模式 0：自动延迟转换模式关闭

Bit	Name	R/W	Reset Value	Function
				1：自动延迟转换模式打开 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
13	CONT	RW	0	单次/连续转换模式 软件可设置和清除该位。如果置为 1，直到该为被清除，否则会一致发生转换 不可能既使能非连续模式又使能连续模式；禁止设置 DISCEN=1 和 CONT=1. 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
12	OVRMOD	RW	0	过载管理模式 软件可设置和清除该位，配置数据过载管理的方式 0：当过载发生时，ADC_DR 寄存器保留旧值 1：当过载发生时，ADC_DR 寄存器会被上一次转换结果覆盖掉 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
11：10	EXTEN[1:0]	RW	2’h0	外部驱动使能和极性选择 软件可设置和清除该位，选择驱动极性和使能驱动 00：硬件驱动检测不使能（软件启动转换） 01：上升沿硬件驱动检测 10：下降沿硬件驱动检测 11：上升沿和下降沿硬件驱动检测 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
9	Reserved	-	-	保留
8：6	EXTSEL[2:0]	RW	3’h0	外部驱动选择 该位选择触发转换启动的外部事件 000：TRG0(TIM1_TRG0) 001：TRG1(TIM1_CH4) 010：TRG2(TIM1_CH1) 011：保留 100：TRG4(TIM14_CH1) 101：TRG5(PWM_CH1) 110：保留 111：TRG7(EXTI LINE1)
5	ALIGN	RW	0	数据对齐 软件设置和清除该位选择右对齐或左对齐 0：右对齐 1：左对齐

Bit	Name	R/W	Reset Value	Function
				仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
4：3	RESSEL[1:0]	RW	00	数据分辨率 软件设置该位选择转换分辨率 00：12 位 01：10 位 10：8 位 11：6 位 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
2：0	Reserved	-	-	保留

10.4.5. ADC 配置寄存器 2 (ADC_CFGR2)

偏移地址：0x10

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	CKMODE[3:0]	RW	4’h0	ADC 时钟模式，软件可设置和清除该位，定义模拟 ADC 的时钟源 0000：PCLK 0001：PCLK/2 0010：PCLK/4 0011：PCLK/8 0100：PCLK/16 0101：PCLK/32 0110：PCLK/64 1000：HSI 1001：HSI/2 1010：HSI/4 1011：HSI/8 1100：HSI/16 1101：HSI/32 1110：HSI/64

Bit	Name	R/W	Reset Value	Function
				其他：预留 仅当 ADC 不使能时 ADCAL=0，ADSTART=0，AD-STP=0)。软件被允许操作这些位
27:0	Reserved	-	-	保留

10.4.6. ADC 采样时间寄存器（ADC_SMPR)

偏移地址：0x14

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：3	Reserved	-	-	保留
2：0	SMP[2:0]	RW	3’h0	采样时钟选择 软件可配置该位选择所有通道的采样时间 000：6.5 ADC 时钟周期 001：9.5 ADC 时钟周期 010：14.5 ADC 时钟周期 011：22.5 ADC 时钟周期 100：49.5 ADC 时钟周期 101：99.5 ADC 时钟周期 110：199.5 ADC 时钟周期 111：519.5 ADC 时钟周期 仅当 ADEN=1，ADSART=0 时（确保没有正在进行的转换）允许软件写这些位

10.4.7. ADC 看门狗阈值寄存器（ADC_TR)

偏移地址：0x18

复位值：0x0FFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	HT[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:16	HT[11:0]	RW	12'hFFF	模拟看门狗高阈值 软件可配，定义模拟看门狗高阈值 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
15:12	Reserved	-	-	保留
11:0	LT[11:0]	RW	12'h0	模拟看门狗低阈值 软件可配，定义模拟看门狗低阈值 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位

10.4.8. ADC 通道选择寄存器（ADC_SQR1)

偏移地址：0x1C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	SQ4[3:0]	RW	4'h0	规则序列第 4 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 4 次。
23:22	Reserved	-	-	保留
21:18	SQ3[3:0]	RW	4'h0	规则序列第 3 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 3 次。
17:16	Reserved	-	-	保留
15:12	SQ2[3:0]	RW	4'h0	规则序列第 2 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 2 次。
11:10	Reserved	-	-	保留
9:6	SQ1[3:0]	RW	4'h0	规则序列第 1 次转换

				这些位由软件写入，任一通道被分配为规则转换序列中的第 1 次。
5:4	Reserved	-	-	保留
3:0	L[3:0]	RW	4'h0	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 …… 1010： 11 个转换 其他：预留

10.4.9. ADC 通道选择寄存器（ADC_SQR2)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	SQ9[3:0]				Res	Res	SQ8[3:0]				Res	Res
				RW						RW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ7[3:0]				Res	Res	SQ6[3:0]				Res	Res	SQ5[3:0]			
RW						RW						RW			

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	SQ9[3:0]	RW	4'h0	规则序列第 9 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 9 次。
23:22	Reserved	-	-	保留
21:18	SQ8[3:0]	RW	4'h0	规则序列第 8 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 8 次。
17:16	Reserved	-	-	保留
15:12	SQ7[3:0]	RW	4'h0	规则序列第 7 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 7 次。
11:10	Reserved	-	-	保留
9:6	SQ6[3:0]	RW	4'h0	规则序列第 6 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 6 次。
5:4	Reserved	-	-	保留
3:0	SQ5[3:0]	RW	4'h0	规则序列第 5 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 5 次。

10.4.10. ADC 通道选择寄存器（ADC_SQR3)

偏移地址：0x24

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:6	SQ11[3:0]	RW	4'h0	规则序列第 11 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 11 次。
5:4	Reserved	-	-	保留
3:0	SQ10[3:0]	RW	4'h0	规则序列第 10 次转换 这些位由软件写入，任一通道被分配为规则转换序列中的第 10 次。

10.4.11. ADC 数据寄存器（ADC_DR1)

偏移地址：0x28

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ1_DATA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	SQ1_DATA [15:0]	R	16'h0	SQ1 转换数据 该位时只读的。上次转换通道的转换结果放于此寄存器。 数据是左对齐或者右对齐的。

10.4.12. ADC 校准因子寄存器（ADC_CALFACT)

偏移地址：0x44

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCALFACT [8:0]									Res	CAL FAIL	Res	Res		Res	Res
R	R	R	R	R	R	R	R	R		RC_W1					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDVLD	WRVLD	FACTSEL[4:0]					WCALFACT [8:0]								
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：23	RCALFACT [8:0]	R	9’h0	溢出标志位读寄存器；(该位段开放 RCALFACT [1:0]给用户) 也是读校准结果寄存器（补码表示）。 当 RDVLD 有效时，读取 ADC 校准结果。只有 ADCAL 为 0 时，软件才能读取。 当 RDVLD 有效时，读取溢出标志位。只有 ADCAL 为 0 时，软件才能读取。
22	Reserved	-	-	保留
21	CALFAIL	RC_W1	0	电容校准状态位。 1：表示 ADC 电容校准失败。硬件置 1；软件写 1 置 0； 0：表示 ADC 电容校准成功。
20:16	Reserved	-	-	保留
15	RDVLD	RW	0	溢出标志位读使能。 1：使能读溢出标志 0：复位状态 也是校准因子读使能。 1：使能读校准因子 0：复位状态
14	WRVLD	RW	0	校准因子写使能。 1：使能写校准因子 0：复位状态
13:9	FACTSEL[4:0]	RW	5’h0	写校准值，溢出标志选择位 当 RDVLD/WRVLD 有效时，选择需要读/写入的配置。 5’d0：校准值，默认为 8。 5’d1：校准比较值，默认为 4，4-bit 5’d2：bist 值测试范围高[11:8]，默认值为 0，4-bit

				5’d3：bist 值测试范围高[7:0]，默认值为 64，8-bit 5’d4：bist 值测试范围低[11:8]，默认值为 0，4-bit 5’d5：bist 值测试范围 低[7:0]，默认值为 0，8-bit 5’d6：BIST test 使能和测试选择 测试使能: 1’b0：当 ADCAL 位被置位时，校准被执行 1’b1：当 ADCAL 位被置位时，bist test 被执行. 测试选择 6’b0：默认测试所有电容 6’b1 - 6’d33：分别测试 1~33 电容 (1-16：MDAC，17-33：REFDAC)
8:0	WCALFACT [8:0]	RW	9’h00	写校准因子寄存器（补码表示）。 当 WRVLD 有效时，WCALFACT 值加载到 ADC 中。 注意(ADCAL 为 0 时，软件才能写入)。

10.4.13. ADC 通用配置寄存器（ADC_CCR)

偏移地址：0x48

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:25	PWRMODE	RW	3’h0	ADC 参考电流选择;
24	VREFSEL	RW	0	ADC 参考电压选择 0：选择 V_{REFP} 作为参考电压 (V_{REFP} 连接到 V_{CC}) 1：选择 V_{REFBUF} 作为参考电压
23	TSEN	RW	0	温度传感器使能位，软件可设置和清除该位，使能/不使能温度传感器 0：不使能 1：使能 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位

22	VREFEN	RW	0	基准 V_{REFINT} 使能位，软件可设置和清除该位，使能/不使能基准 V_{REFINT} 0：不使能 1：使能 仅当 ADSART=0 时（确保没有正在进行的转换）允许软件写这些位
21：0	Reserved	-	-	保留

11. 高级控制定时器 (TIM1)

11.1. TIM1 简介

高级控制定时器（TIM1）由一个 16 位的自动装载计数器组成，计数器由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM)。

使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

高级控制定时器（TIM1）和通用定时器（TIMy）是完全独立的，它们不共享任何资源。它们可以同步操作。

11.2. TIM1 主要特性

- 16 位递增、递减或者递增/递减的自动重装载计数器
- 16 位可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频
- 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

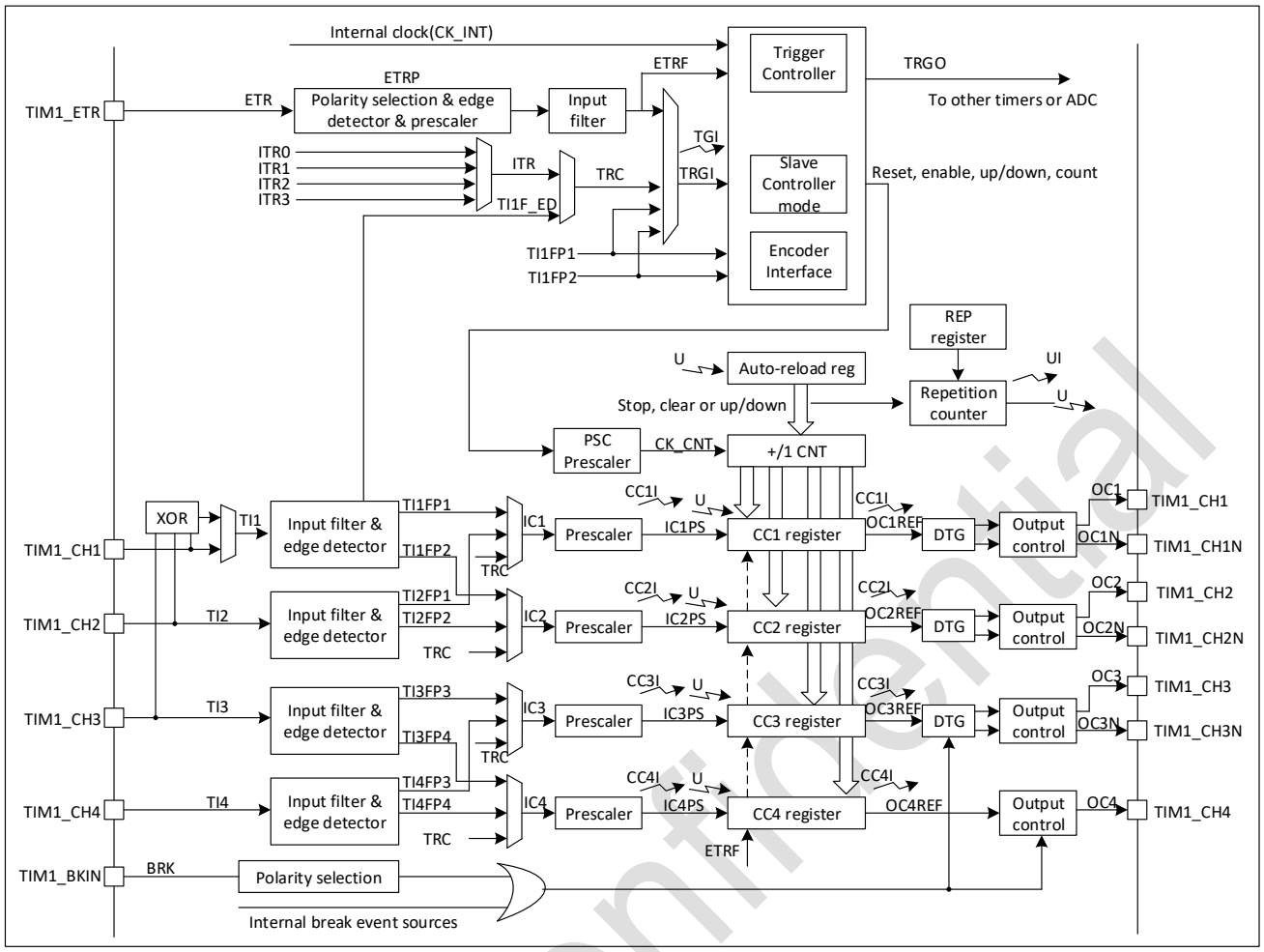


图 11-1 高级控制定时器架构框图

11.3. TIM1 功能描述

11.3.1. 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 TIM1_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 TIM1_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 65536 之间的任意值分频。它是基于一个（在 TIM1_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

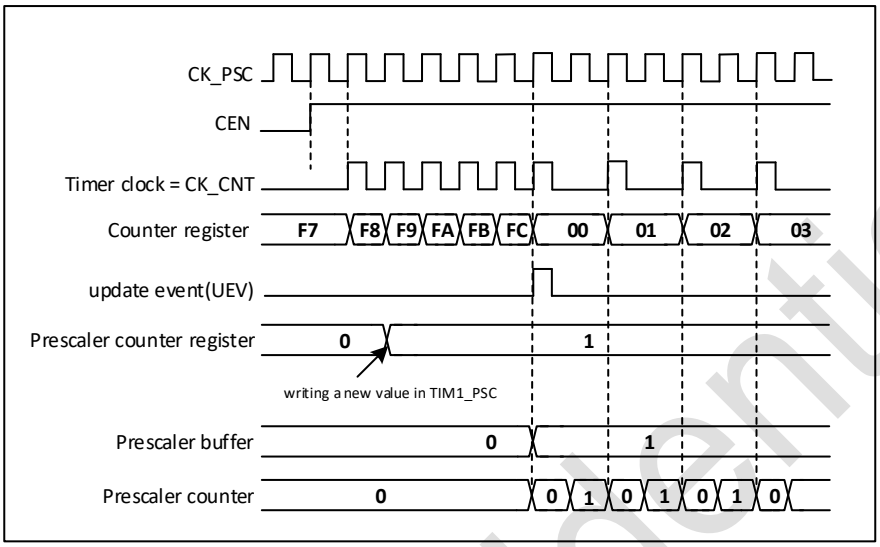


图 11-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

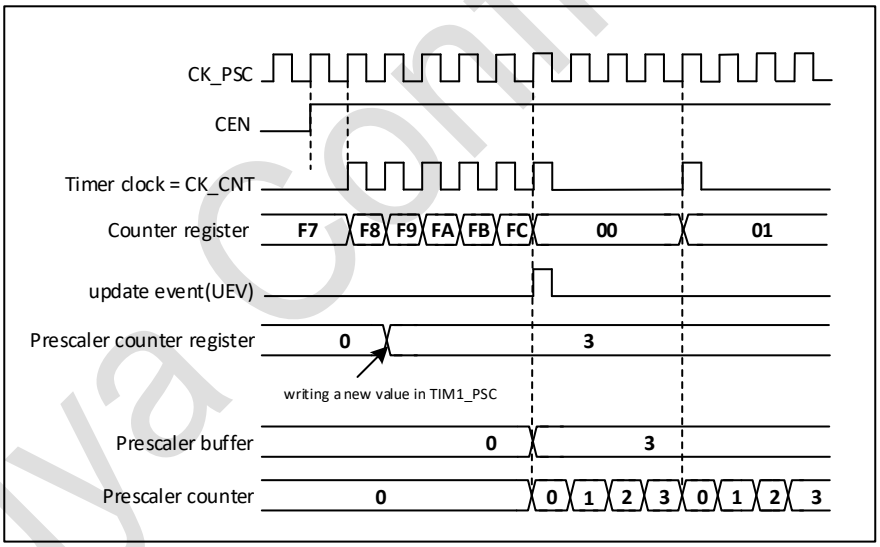


图 11-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

11.3.2. 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几次（对重复计数器可编程）后，产生更新事件。否则，在每个计数溢出时，产生更新事件。

在 TIM1_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 TIM1_CR1 寄存器中的 UDIS 位，软件可以禁止更新事件；这样是为了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，计数器从 0 开始重新启动，预分频器也从 0 开始重新启动（但预分频器的数值不变）。此外，如果设置了 TIM1_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但不设置 UIF 标志（即不产生中断）。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（TIM1_SR 寄存器中的 UIF 位)。

- 重复计数器被重新加载为 TIM1_RCR 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值（TIM1_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值（TIM1_PSC 寄存器的内容)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

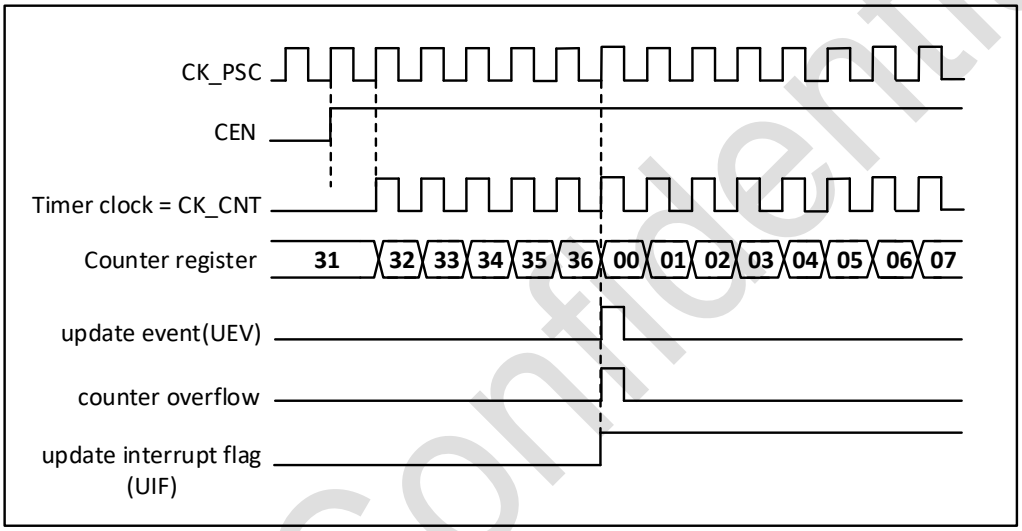


图 11-4 计数器时序图，内部时钟 1 分频

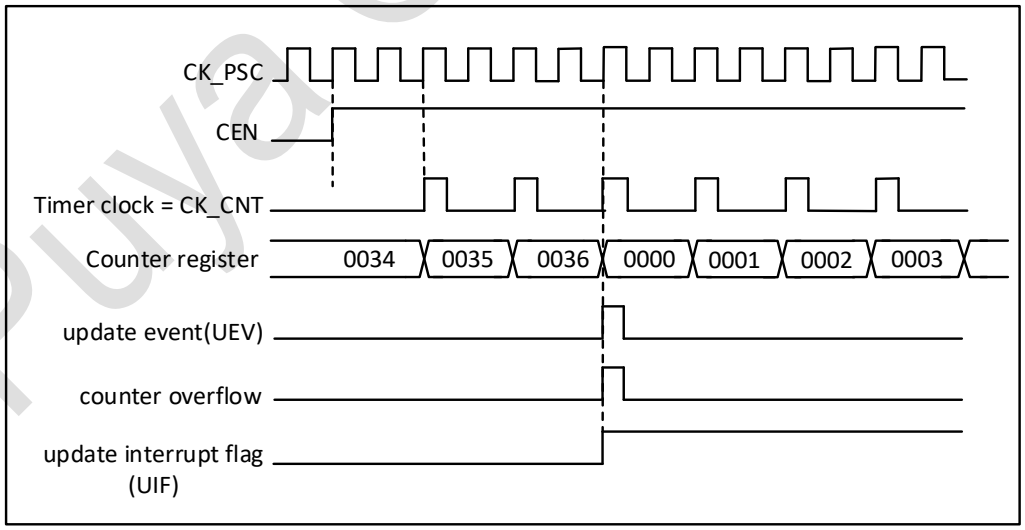


图 11-5 计数器时序图，内部时钟 2 分频

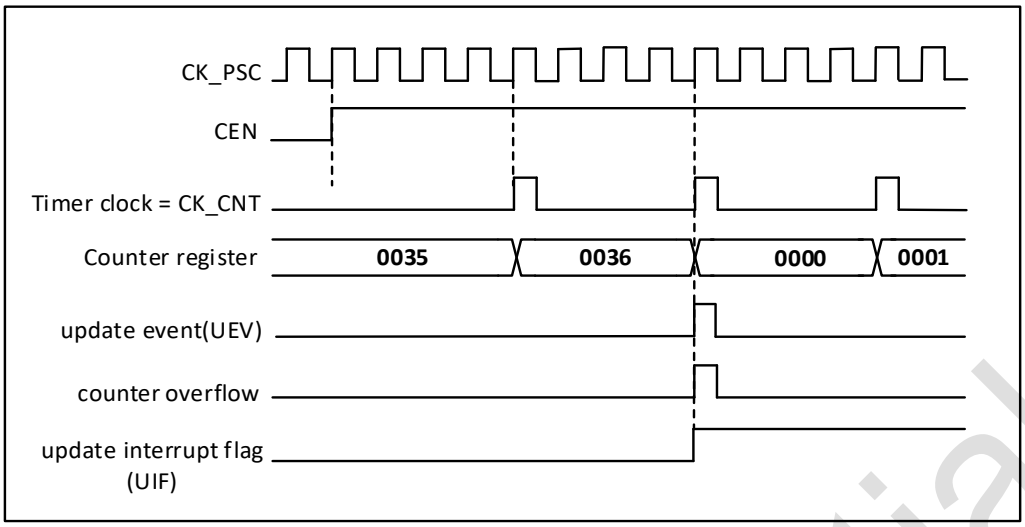


图 11-6 计数器时序图，内部时钟 4 分频

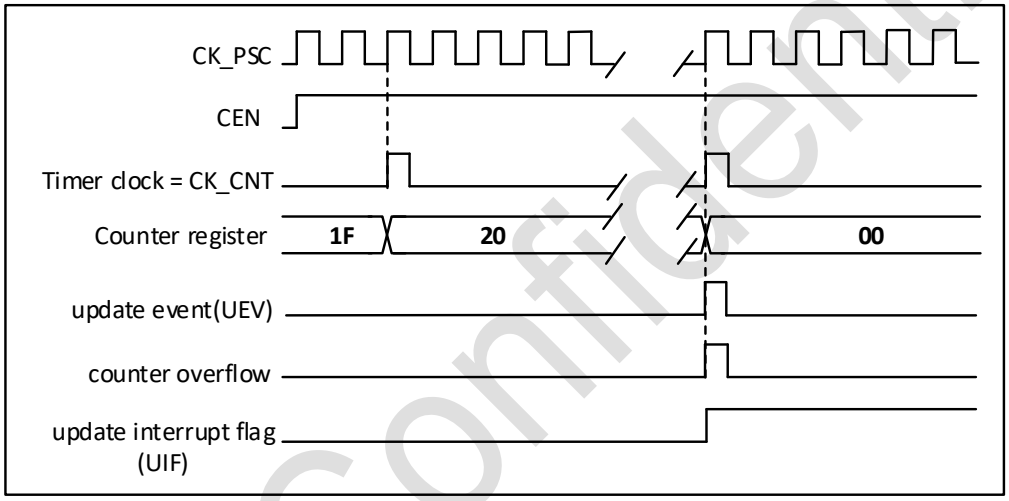


图 11-7 计数器时序图，内部时钟 N 分频

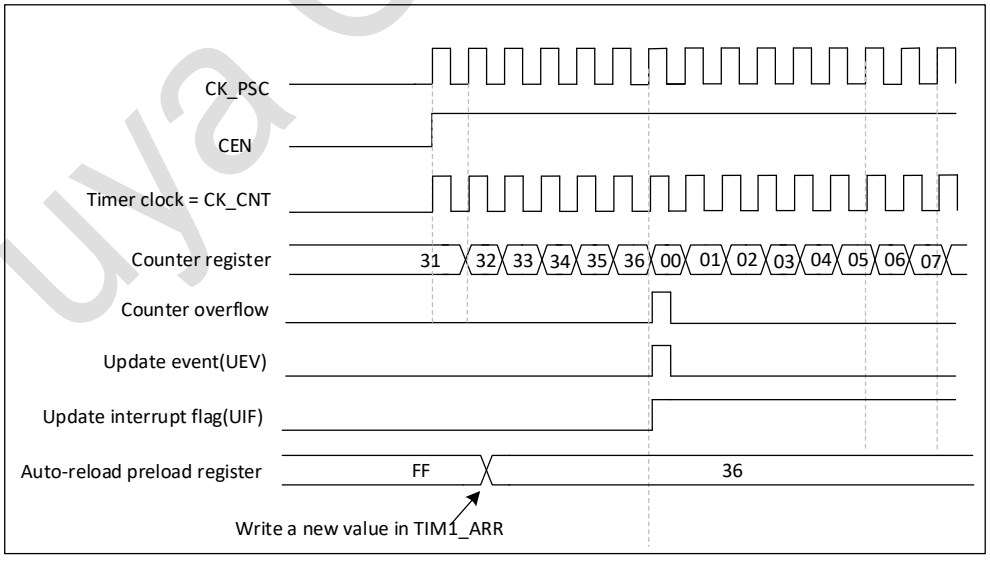


图 11-8 计数器时序图，当 ARPE=0 时的更新事件(TIM1_ARR 未预装载)

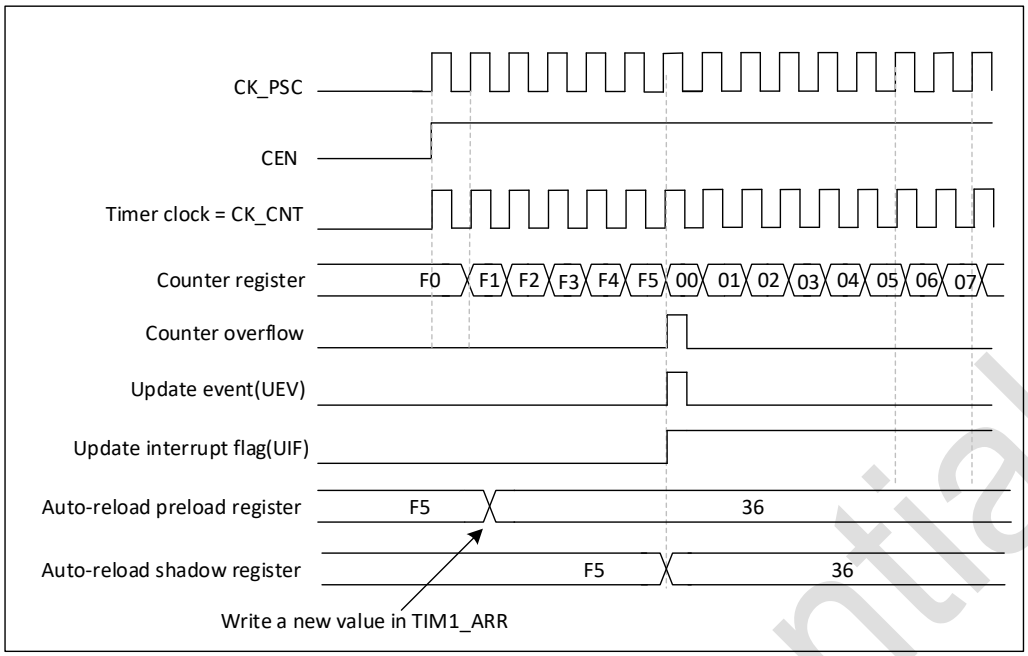


图 11-9 计数器时序图，当 ARPE=1 时的更新事件(TIM1_ARR 预装载)

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)，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。

下图显示了 TIMx_ARR=0x36 时不同时钟频率下计数器行为的一些示例。

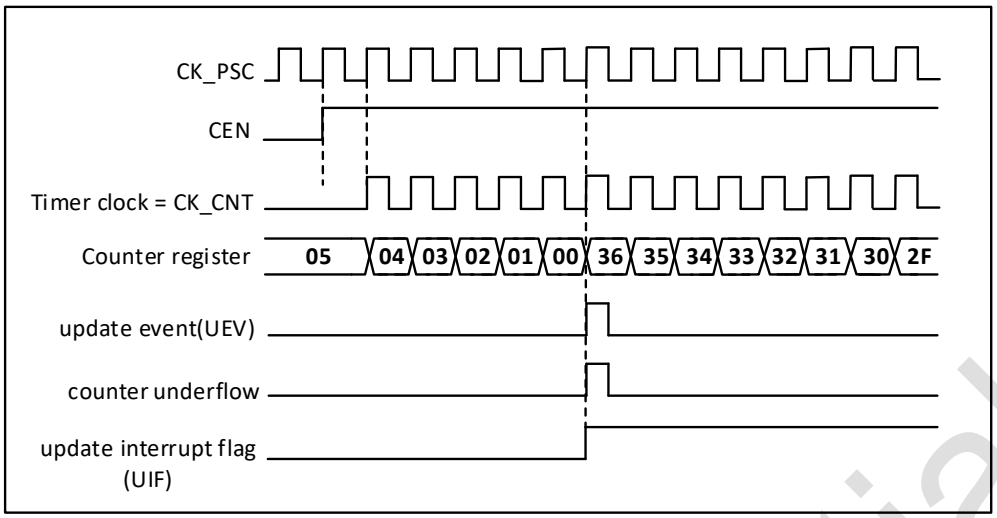


图 11-10 计数器时序图，内部时钟 1 分频

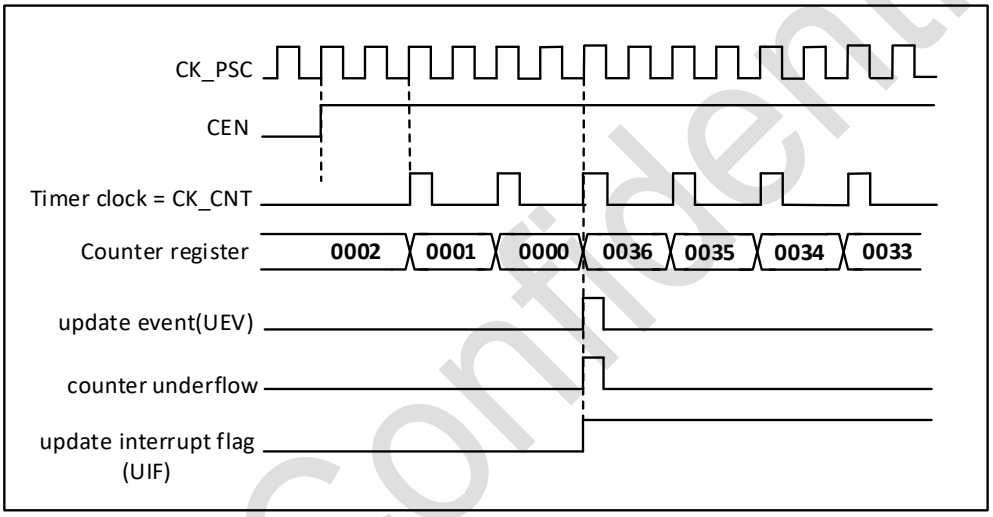


图 11-11 计数器时序图，内部时钟 2 分频

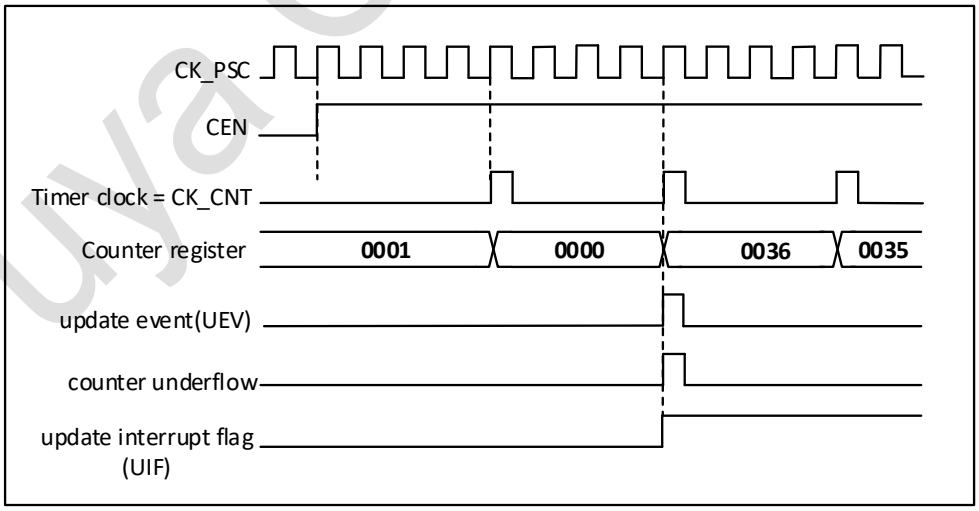


图 11-12 计数器时序图，内部时钟 4 分频

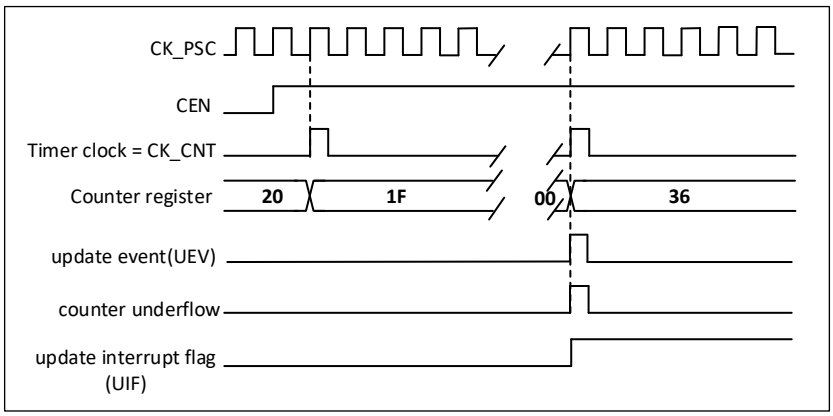


图 11-13 计数器时序图，内部时钟 N 分频

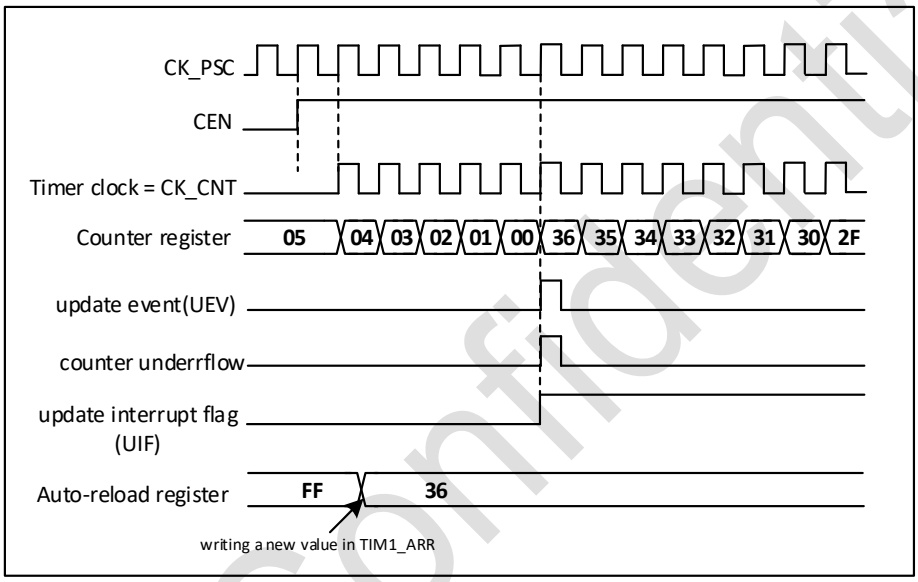


图 11-14 计数器时序图，当没有使用周期计数器时的更新事件

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，输出比较中断标志在以下情况将被置位，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 寄存器中的内容)

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)

下图显示了不同时钟频率下计数器行为的一些示例。

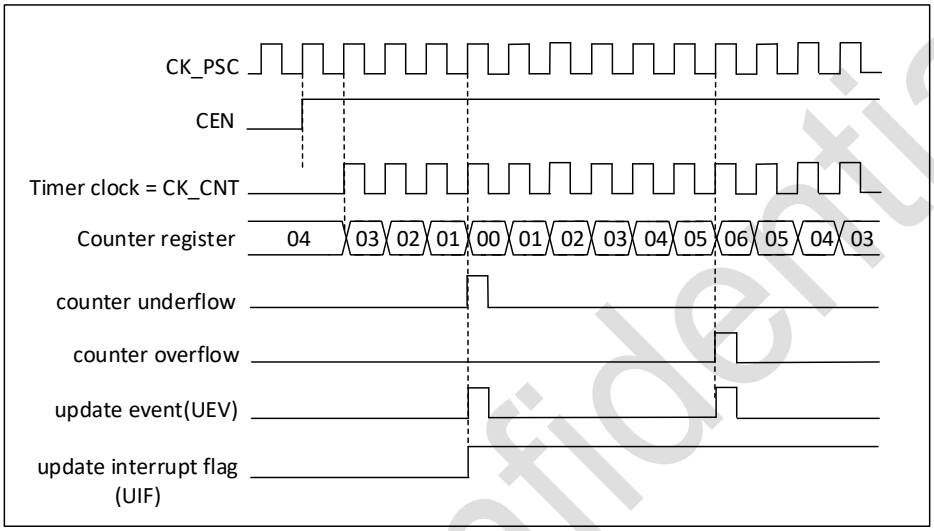


图 11-15 计数器时序图，内部时钟 1 分频，TIMx_ARR = 0x6

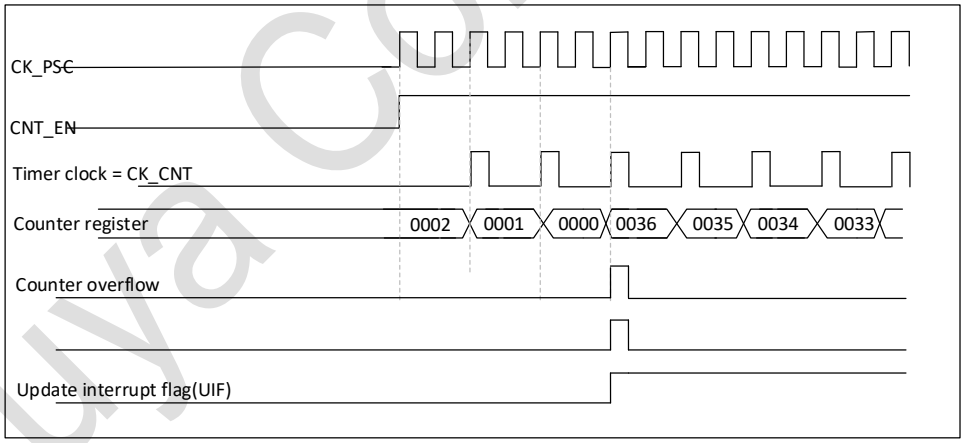


图 11-16 计数器时序图，内部时钟分频因子为 2，TIMx_ARR=0x36

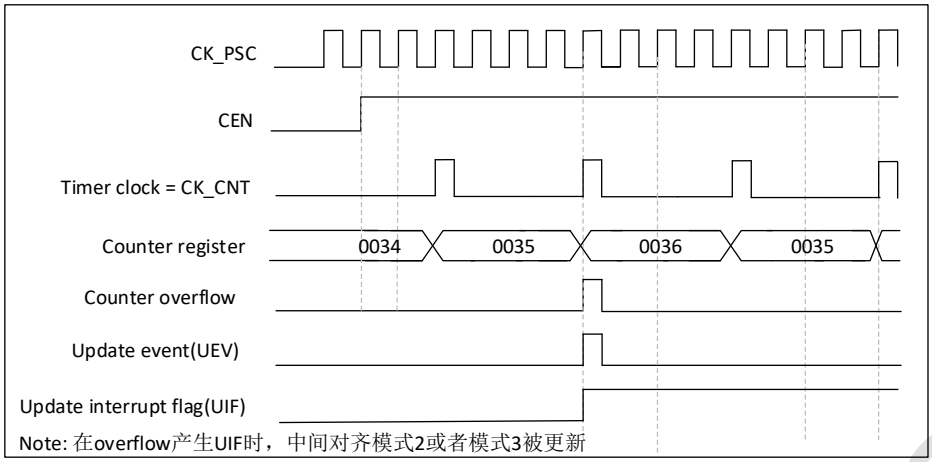


图 11-17 计数器时序图，内部时钟 4 分频，TIMx_ARR=0x36

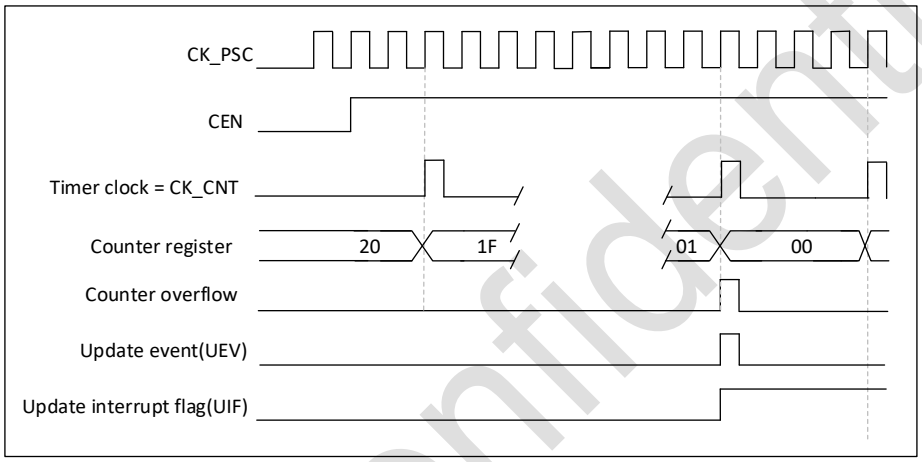


图 11-18 计数器时序图，内部时钟 N 分频

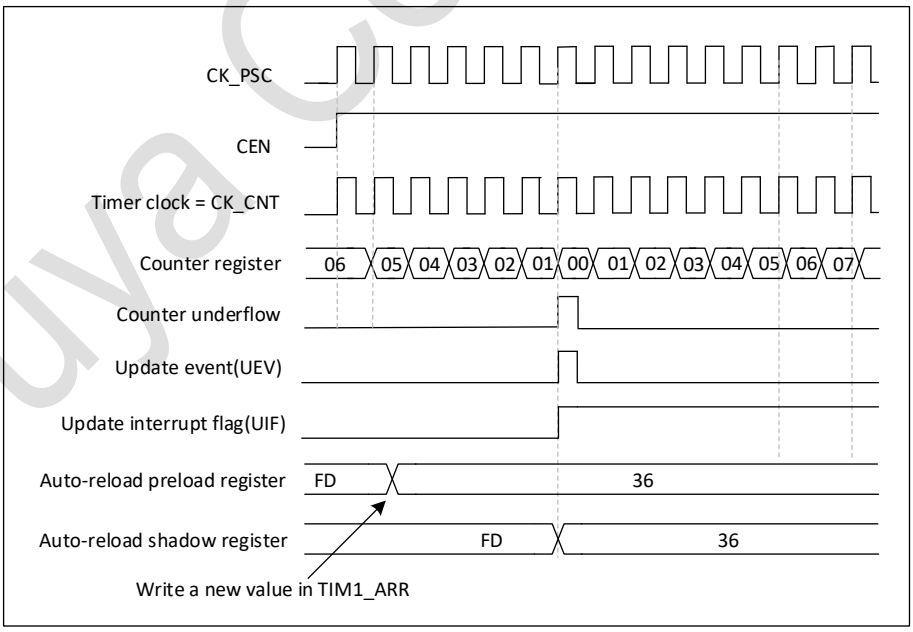


图 11-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

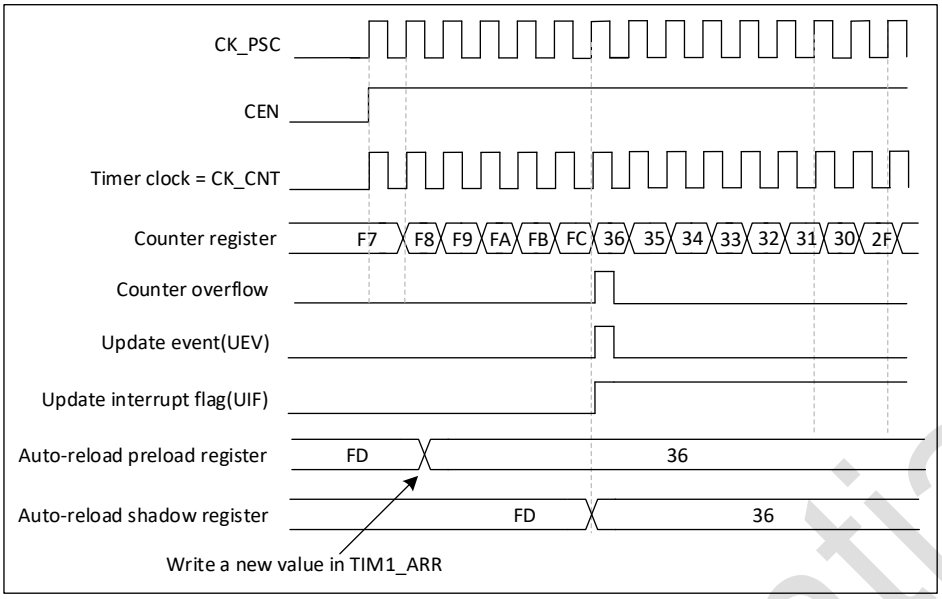


图 11-20 计数器时序图，ARPE=1 时的更新事件(计数器上溢)

11.3.3. 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如何产生的。它实际上仅当重复计数器计数到零才产生。这在产生 PMW 信号时很有用的。

这意味着在每 N+1 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 的最大循环周期为 128，但它能够在每个 PWM 周期 2 次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 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，更新事件发生在上溢、或者下溢，这取决于何时写入 RCR 寄存器以及何时计数器开始。如果在启动计数器之后写 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）。

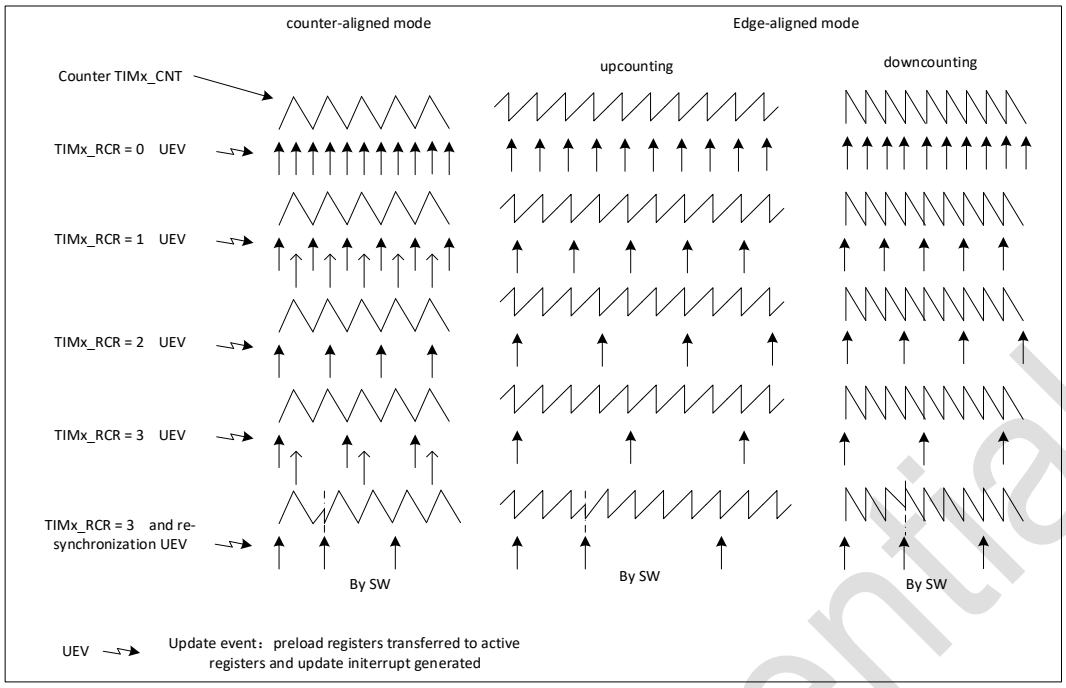


图 11-21 不同模式下更新速率的例子，及 TIM1_RCR 的寄存器设置

11.3.4. 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 Timer3 的预分频器。

内部时钟源（CK_INT）

如果从模式控制器被禁止（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_EGR 寄存器）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。

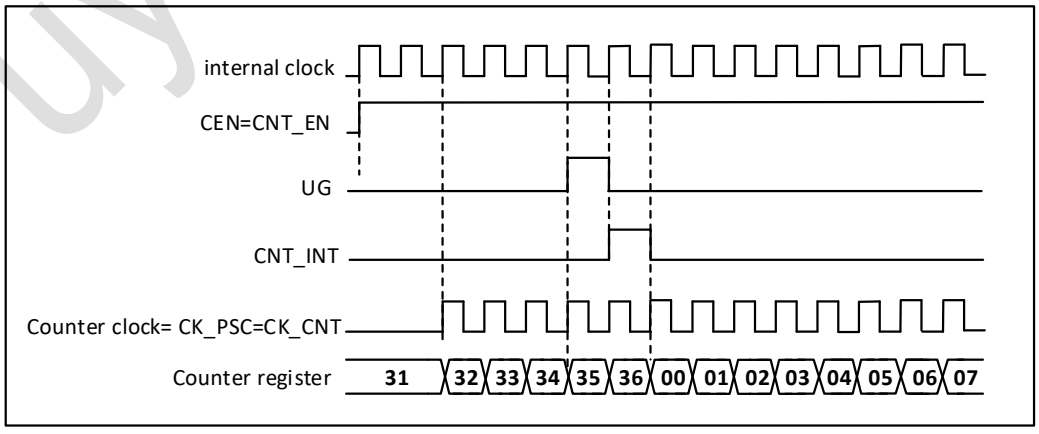


图 11-22 一般模式下的控制电路，内部时钟 1 分频

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。

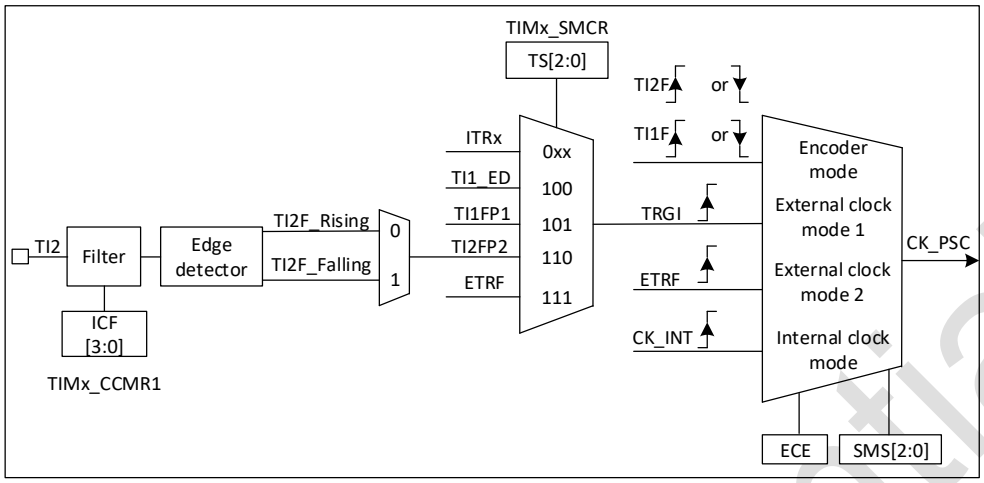


图 11-23 TI2 外部时钟连接例子

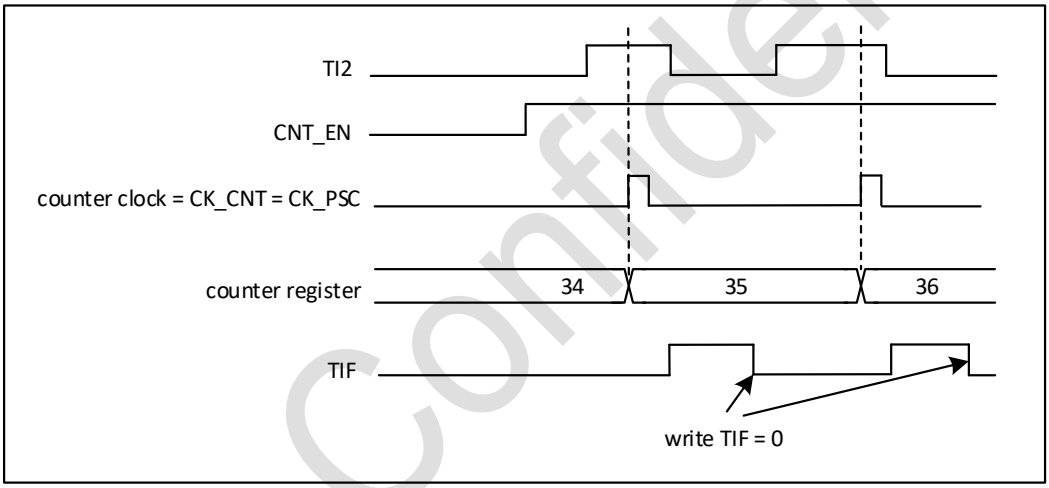


图 11-24 外部时钟模式 1 下的控制电路

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。

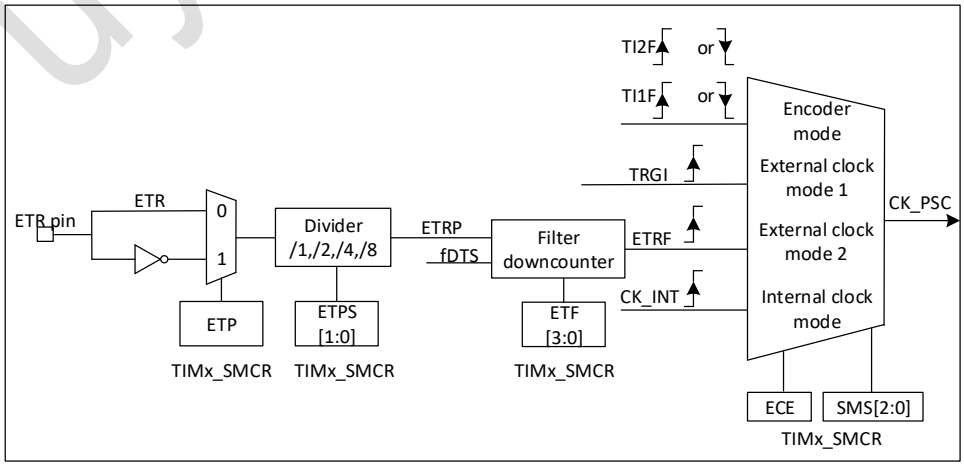


图 11-25 TI2 外部触发输入框图

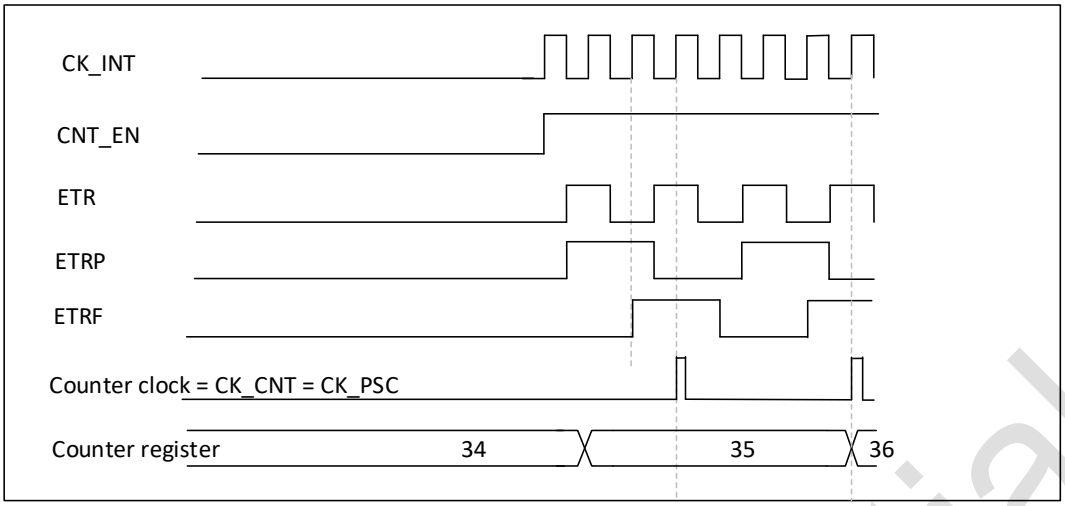


图 11-26 外部时钟模式 2 下的控制电路

11.3.5. 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）。

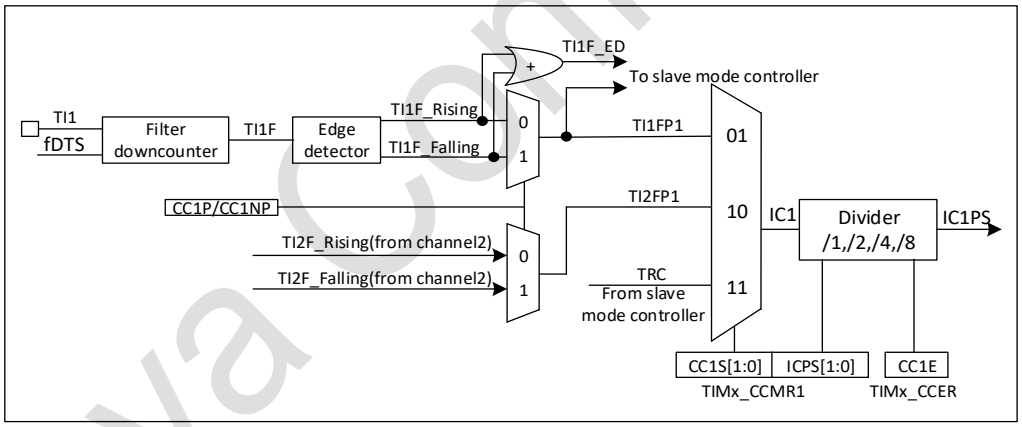


图 11-27 捕获/比较通道(如：通道 1 输入部分)

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。

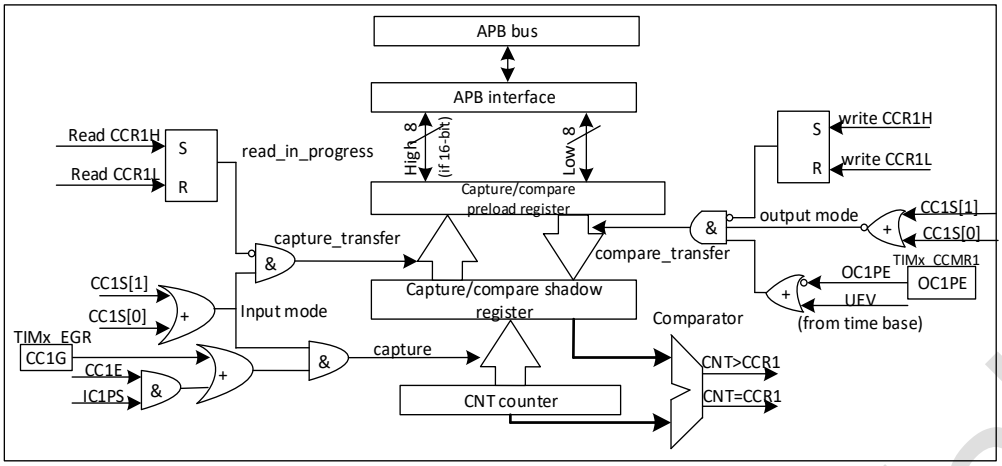


图 11-28 捕获/比较通道 1 的主电路

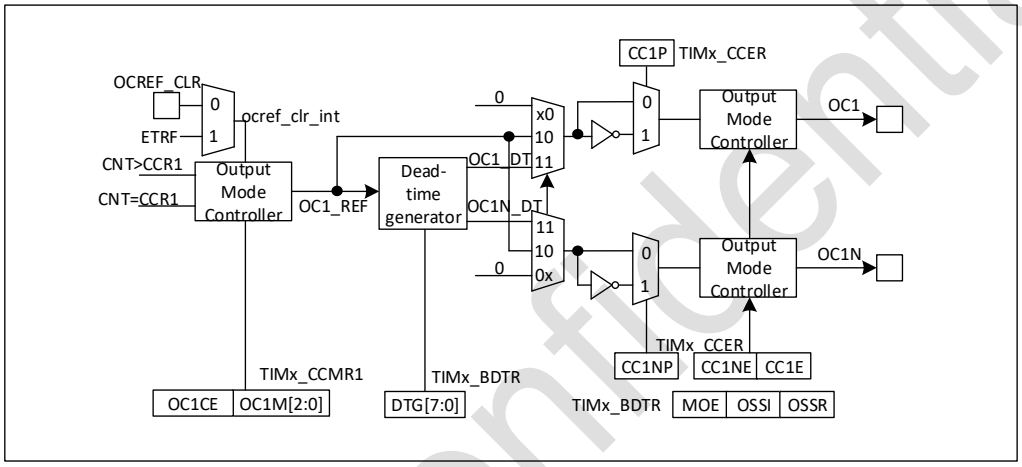


图 11-29 捕获/比较通道的输出部分(通道 1 至 3)

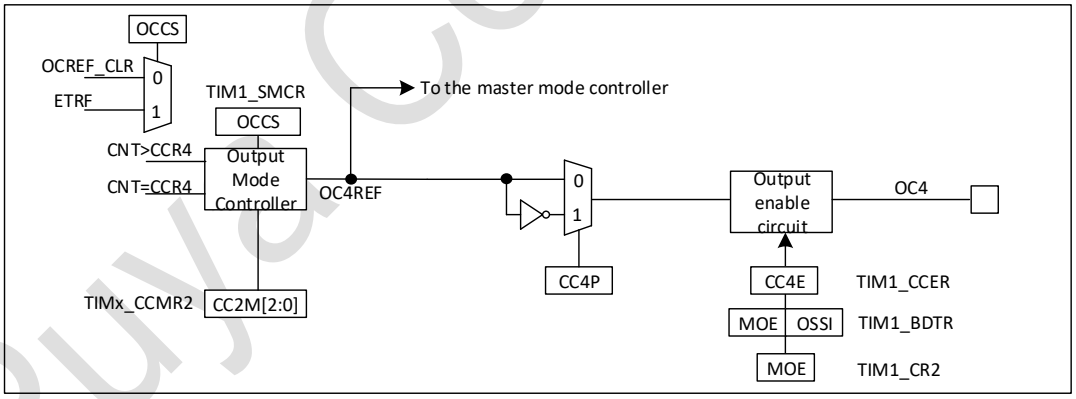


图 11-30 捕获/比较通道的输出部分(通道 4)

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。

11.3.6. 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，如果中断操作被打开，则将产生中断请求。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

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。

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请求。

- 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)

- 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。

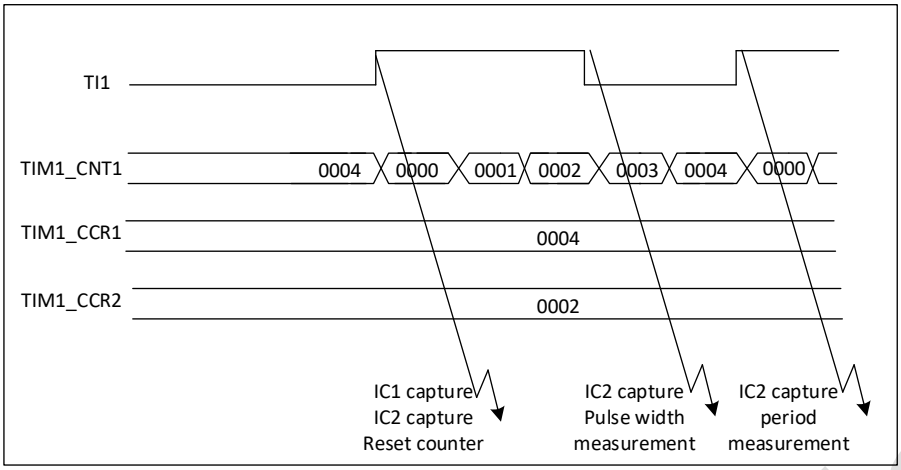


图 11-31 PWM 输入模式时序

11.3.7. 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/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请求。这将会在下面的输出比较模式一节中介绍。

11.3.8. 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

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

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

— 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。

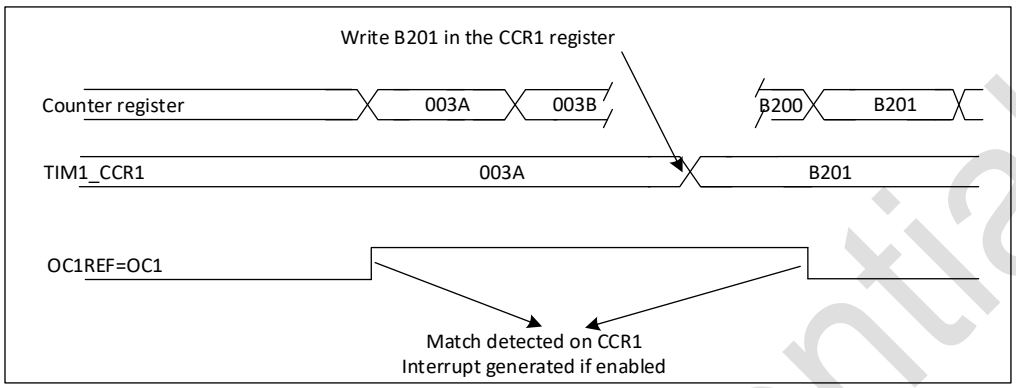


图 11-32 输出比较模式，翻转 OC1

11.3.9. 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。

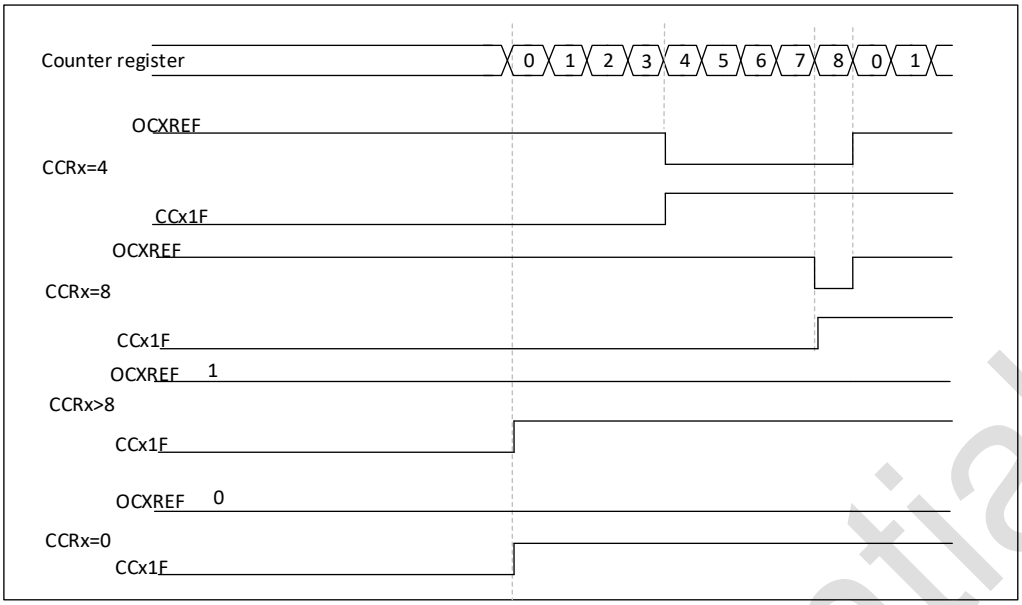


图 11-33 边沿对齐方式 PWM 输出，向上（ARR=8）

- 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

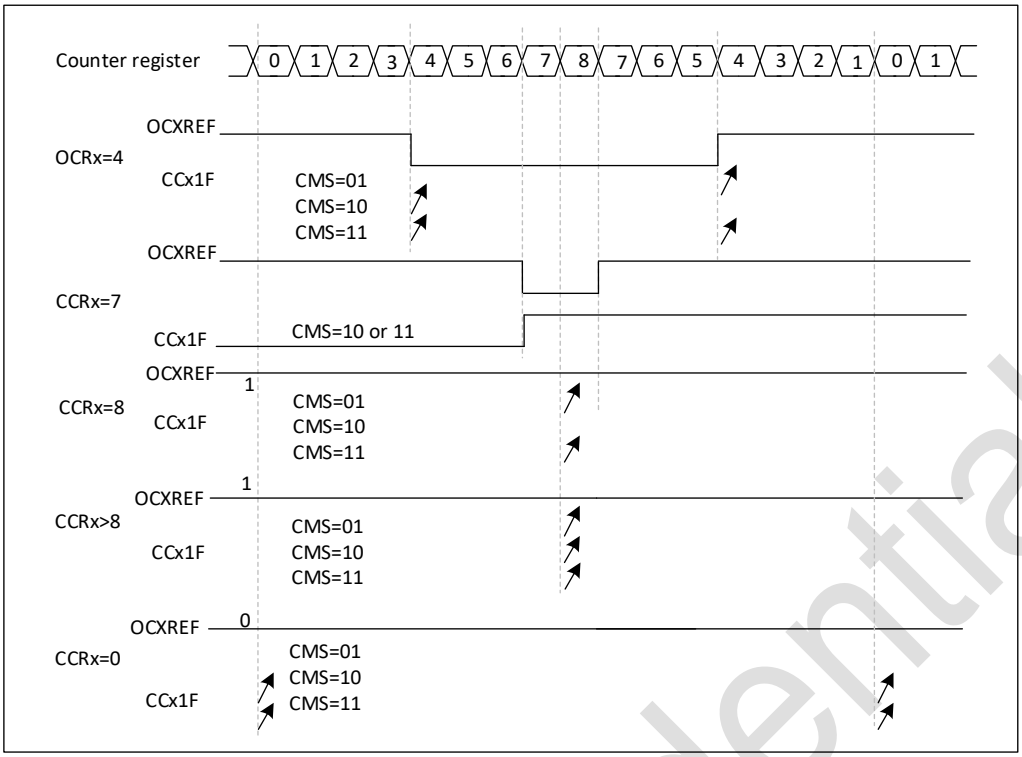


图 11-34 中央对齐的 PWM 波形(APR=8)

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。

PWM 移相模式

PWM 移相模式是产生不同的相位差，这种模式通常用于调节电机速度或实现更复杂的控制逻辑。在 PWM 中央对齐模式（CMS 不为 00）下才能使能 PWM 移相模式，通过在各 PWM 周期产生一次更新中断（根据 TIM1_AF1 中的 INTR_SEL 选择中断位置），更新寄存器输出比较值 TIM1_CCR1、TIM1_CCR2、TIM1_CCR3（其中 CCRx[15:0]为向上计数模时的比较值，CCRx_H[15:0]为向下计数模时的比较值）产生 PWM 移相功能。

下图是 PWM 移相模式结构概览。

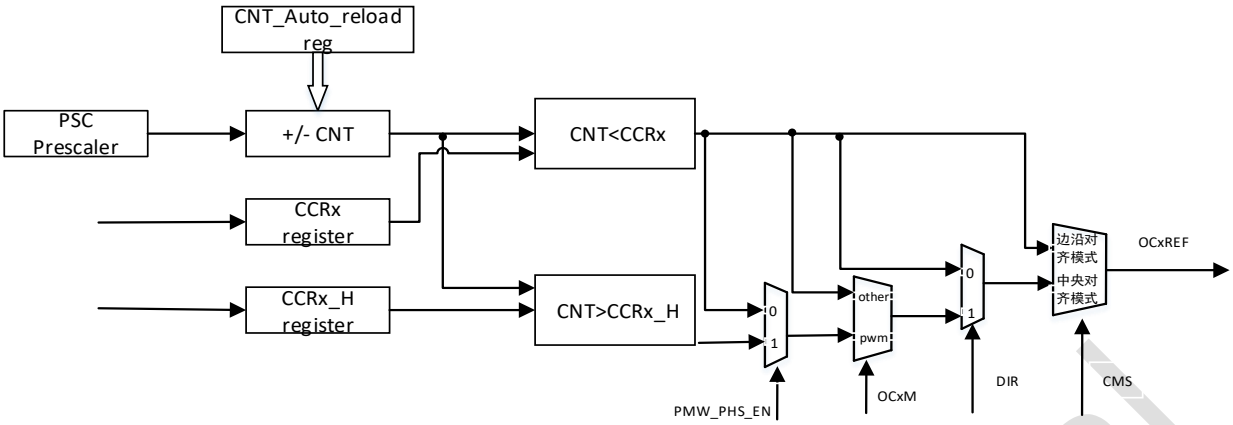


图 11-35 PWM 移相模式结构概览

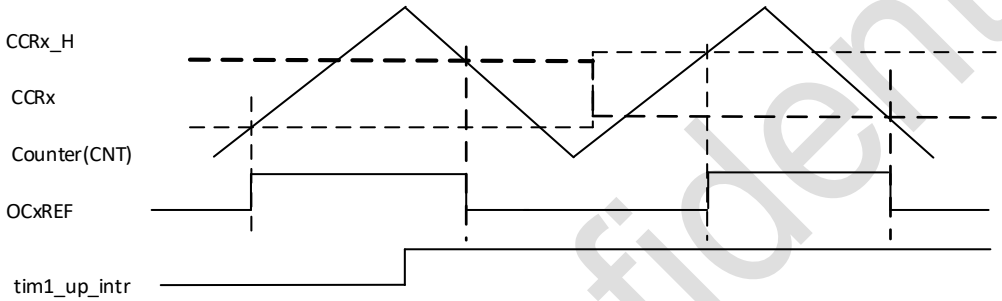


图 11-36 PWM 向右移相模式

上图为 PWM 向右移相模式，上溢时产生中断（上溢/下溢中断可用 INTR_SEL 选择），进入中断后可重新配置输出比较寄存器 CCRx，CCRx_H。

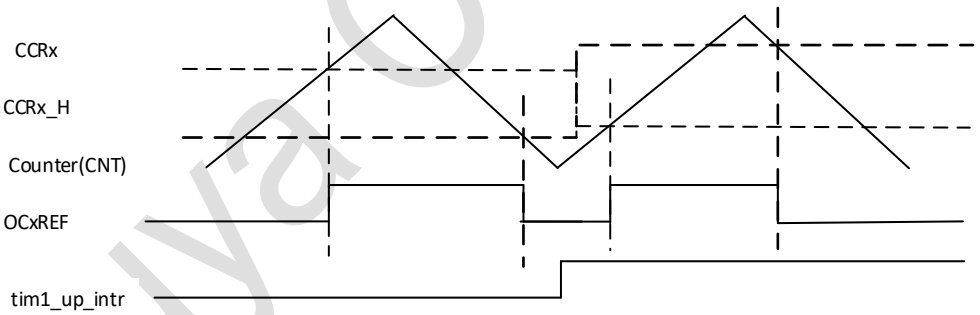


图 11-37 PWM 向左移相模式

上图为 PWM 向左移相模式，下溢时产生更新中断（上溢/下溢中断可用 INTR_SEL 选择），进入更新中断后可重新配置输出比较寄存器 CCRx，CCRx_H。

下边给出了 PWM 移相配置例子：

假定计数器选择内部时钟源:

- 配置 TIM1_CR1 寄存器中的 CMS 选择中央对齐模式
- 配置 TIM1_AF1 寄存器中的 PWM_PHS_EN=1，PWM 移相使能，ISR_SEL=01 选择波峰中断。
- 配置 TIM1_CCR1[31:0]。
- 配置 TIM1_CCR2[31:0]。
- 配置 TIM1_CCR3[31:0]。

- 配置 TIM1_ARR 自动重装载寄存器。
- 配置 TIM1_CCMR1 中的 OC1M 为 PWM2 模式，OC2M 为 PWM2 模式。
- 配置 TIM1_CCMR2 中的 OC3M 为 PWM2 模式。
- 配置 TIM1_CCER 中的 CC1E=1，CC2E=1，CC3E=1，比较输出使能。
- 配置 TIM1_BDTR 中的 MOE=1，主输出使能。
- 配置 TIM1_DIER 中的 UIE=1，允许更新中断。
- 配置 TIM1_CR1 寄存器中的 CEN=1 使能计数器。
- 产生中断后可重新配置 TIM1_CCR1/ TIM1_CCR2/ TIM1_CCR3。

11.3.10. 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。特别是，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)

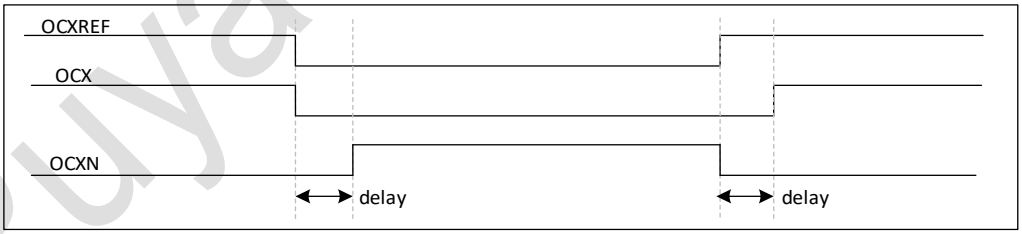


图 11-38 带死区插入的互补输出

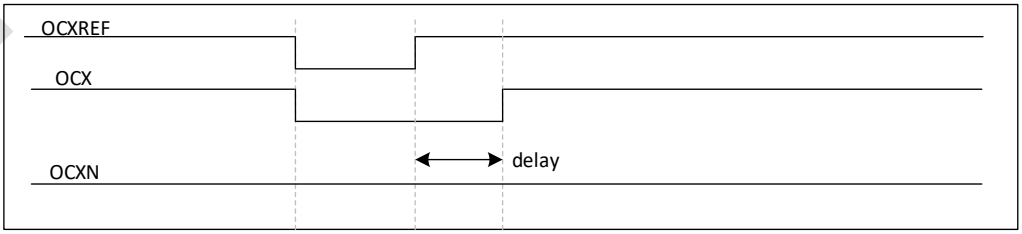


图 11-39 死区波形延迟大于负脉冲

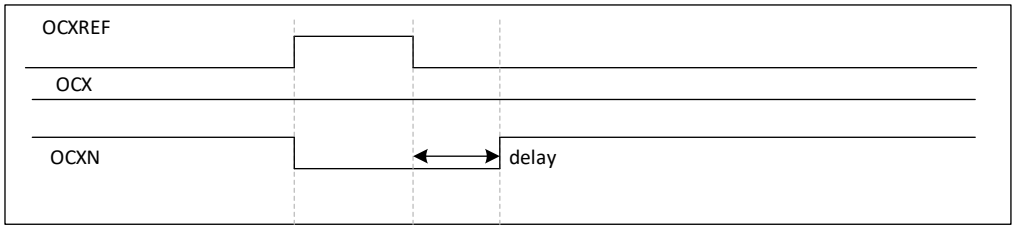


图 11-40 死区波形延迟大于正脉冲

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。

11.3.11. 使用刹车功能

刹车功能的目的是保护由定时器产生的 PWM 信号驱动的功率开关。两个断路输入通常连接到功率级和三相逆变器的故障输出。当被激活时，断开电路关闭 PWM 输出并将其强制到预定义的安全状态。也可以选择一些内部 MCU 事件来触发输出关断。

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
- SRAM 奇偶校验错误信号
- 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 BKE 位开启。

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

下图显示响应刹车的输出实例。

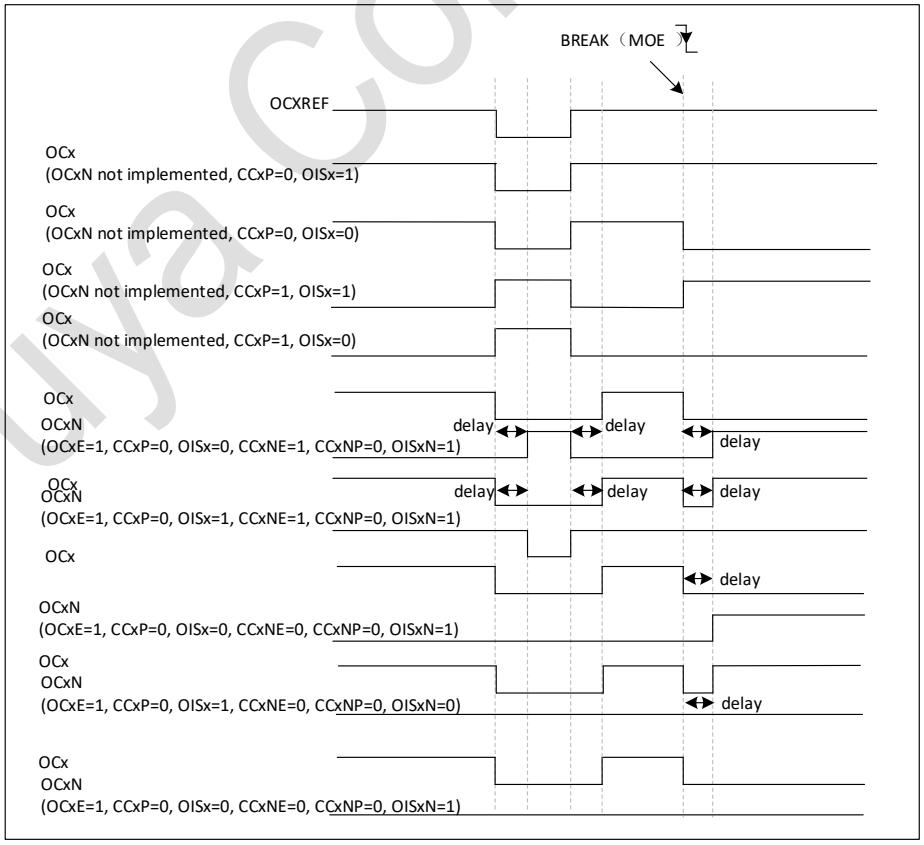


图 11-41 响应刹车的输出

11.3.12. 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 OCREF_CLR_INPUT 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次计数溢出所产生的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

而 OCREF_CLR_INPUT 可以通过配置 TIMx_SMCR 寄存器中的 OCCS 位，选择 ETRF(ETR 滤波后)。

例如，OCxREF 信号可以联到一个 ETRF。这时，ETR 必须配置如下：

1. 外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。
2. 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。
3. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要配置。

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

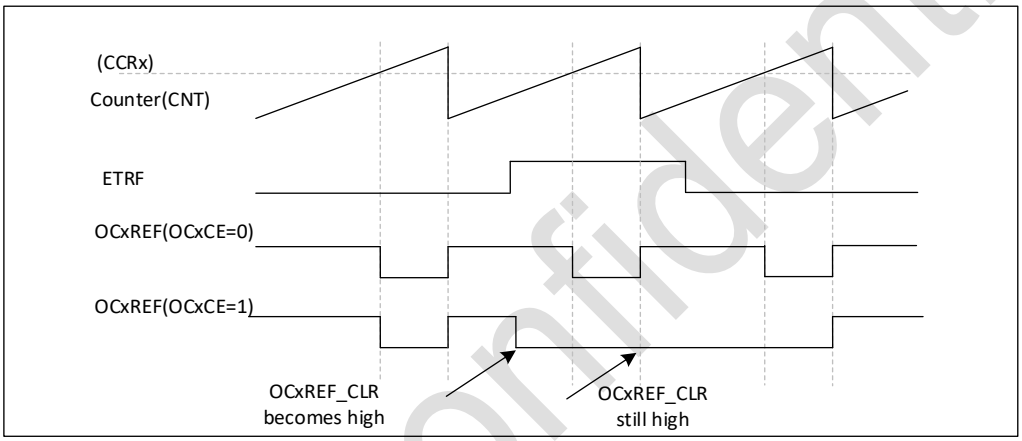


图 11-42 清除 TIM1 的 OCxREF

11.3.13. 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 commutation 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。

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

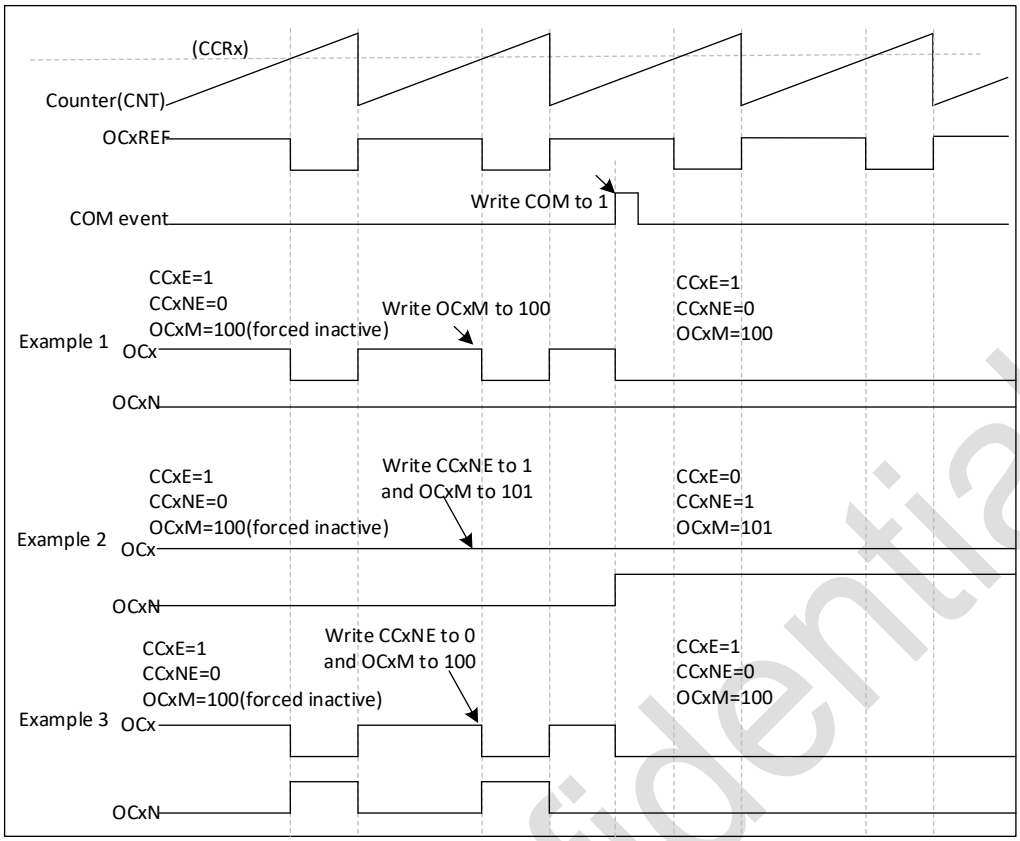


图 11-43 六步产生，COM 的例子(OSSR=1)

11.3.14. 单脉冲模式

单脉冲模式（OPM）是之前所述众多模式中的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后，产生一个脉宽可被程序控制的脉冲。

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

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。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

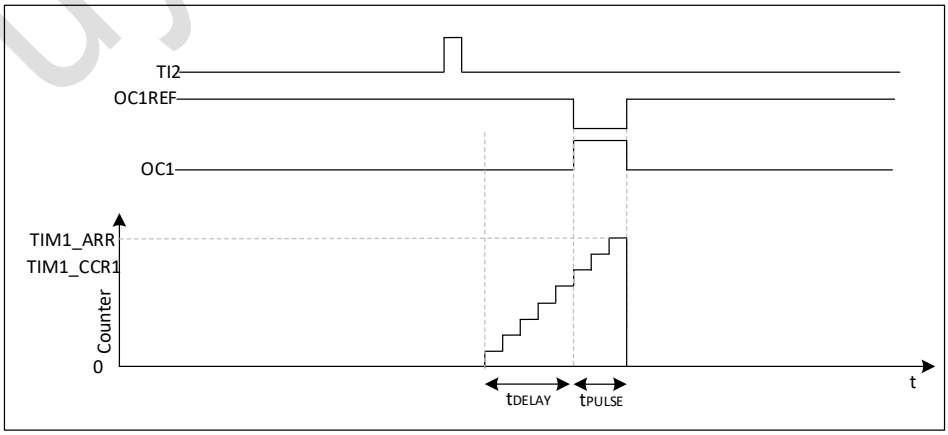


图 11-44 单脉冲模式的例子

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)

- 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。

11.3.15. 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 table 35，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 11-1 计数方向与编码器信号的关系

有效边沿	相反信号的电平 (TI1FP1 for TI2，TI2FP2 for TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

一个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)

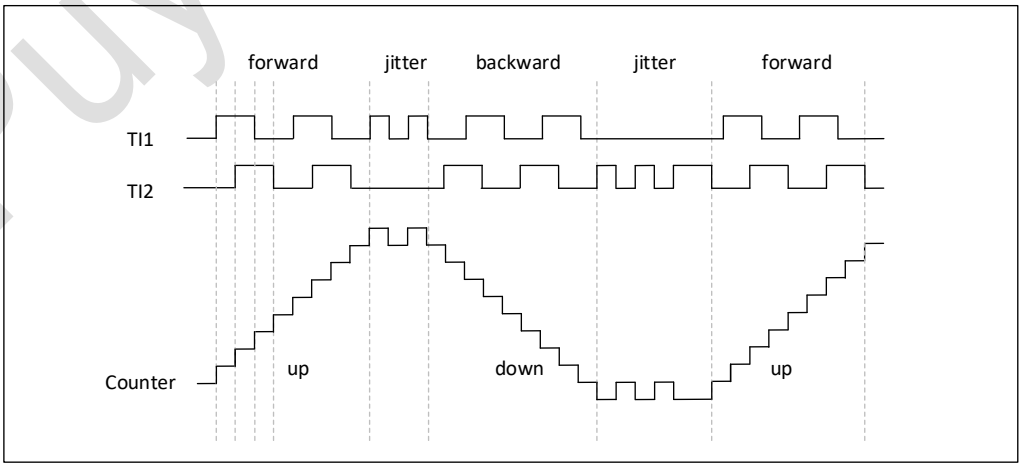


图 11-45 编码器模式下的计数器操作实例

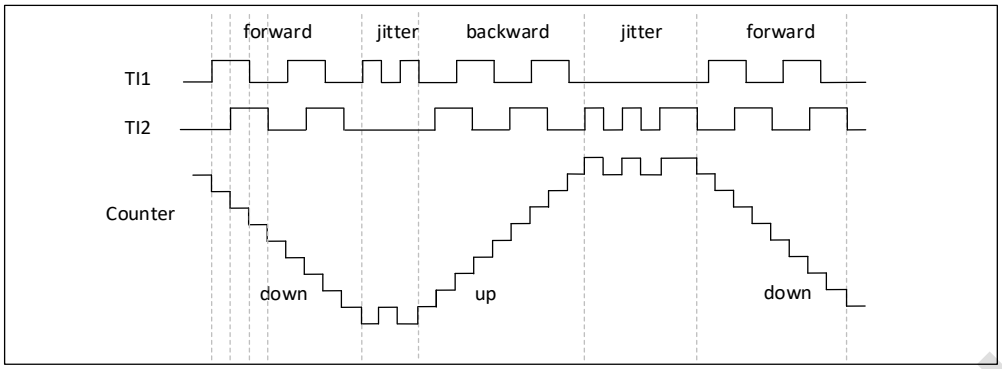


图 11-46 TI1FP1 反相的编码器接口模式实例

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）。

11.3.16. 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。

11.3.17. 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 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。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。

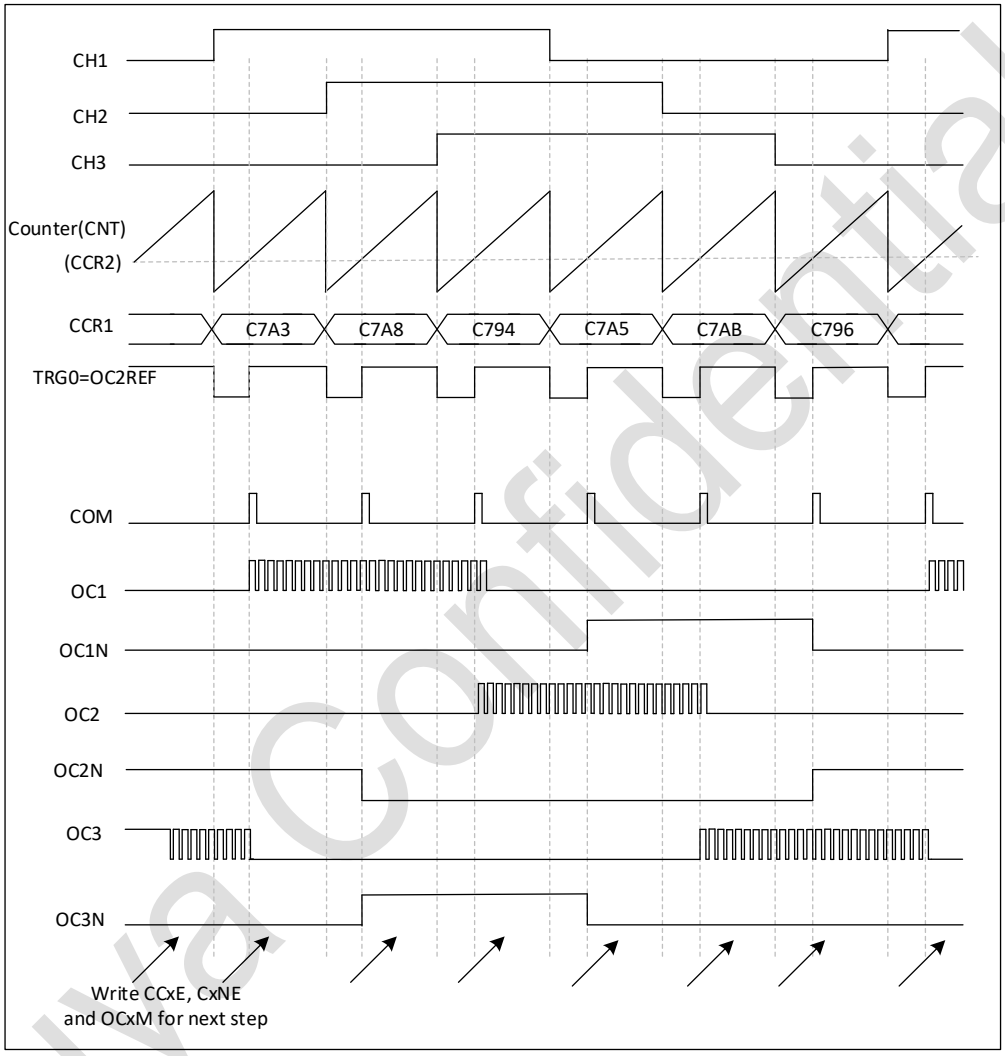


图 11-47 霍尔传感器接口的实例

11.3.18. 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)位的设置，产生一个中断请求。

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。

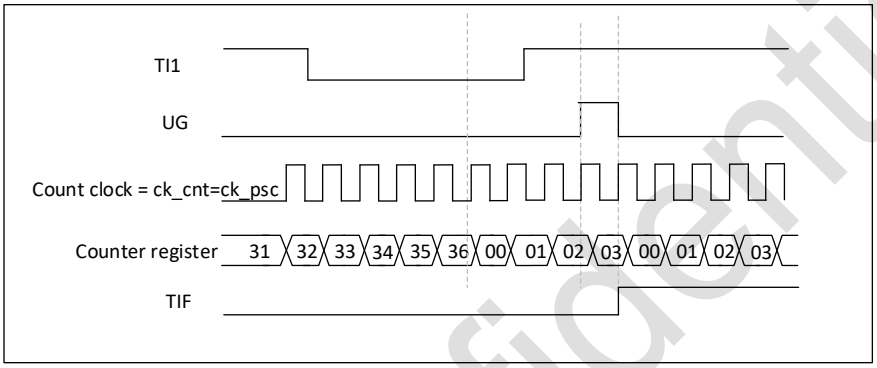


图 11-48 复位模式下的控制电路

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。

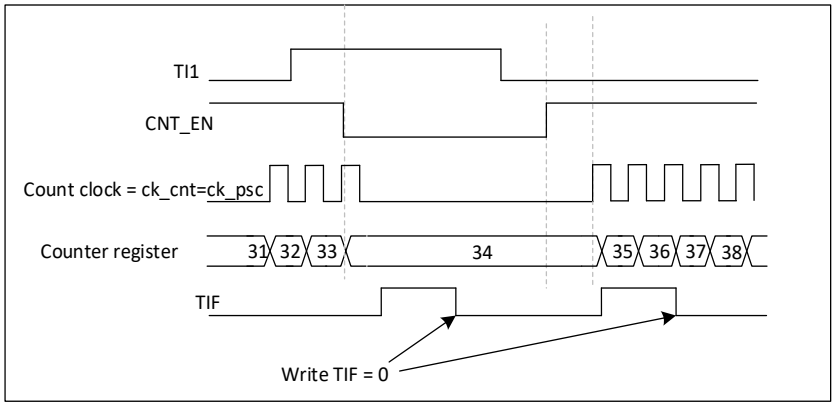


图 11-49 门控模式下的控制电路

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：

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

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。 TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

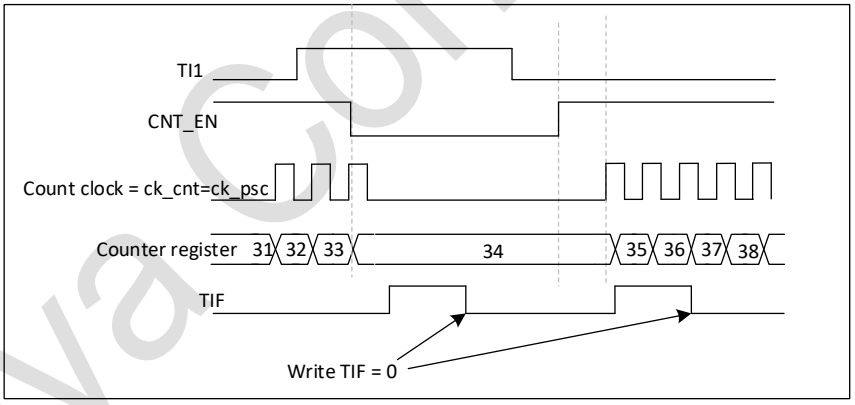


图 11-50 门控模式下的控制电路

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：

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

– IC1F=0000：没有滤波

– 触发操作中不使用捕获预分频器，不需要配置

– 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源

– 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

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

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。 ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

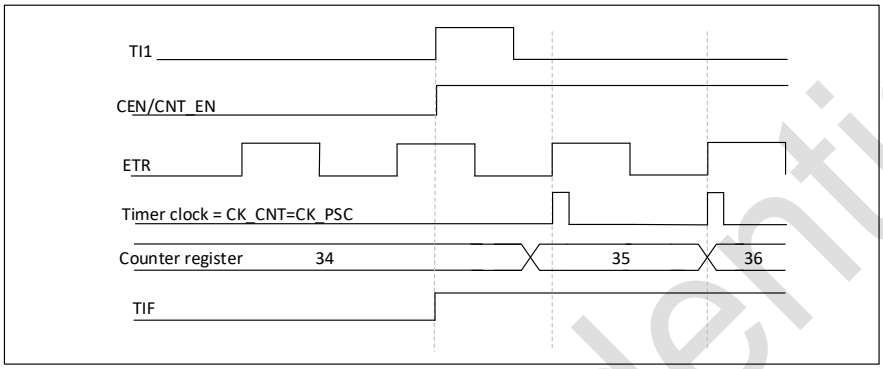


图 11-51 外部时钟模式 2 + 触发模式下的控制电路

11.3.19. 定时器同步

所有 TIMx 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况。

使用一个定时器作为另一个定时器的预分频器

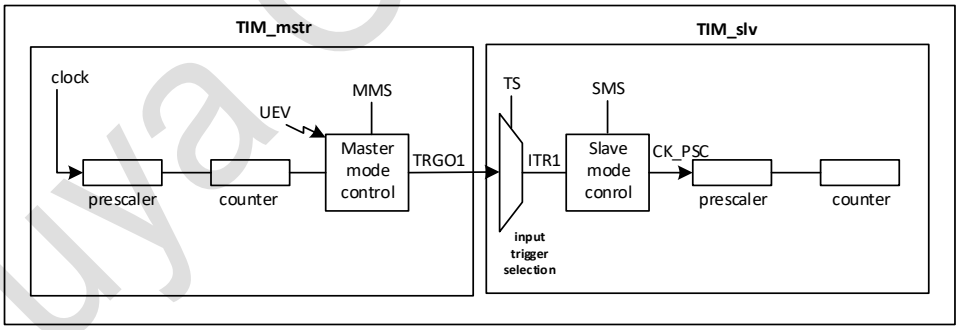


图 11-52 主/从定时器的例子

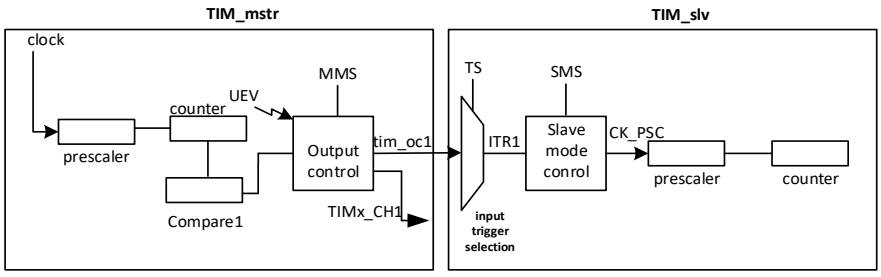


图 11-53 仅有 1 个通道定时器的主从连接示例

仅具有一个通道的计时器（见上图）不具有主模式。但是，tim_oc1 输出信号可以作为从定时器的触发器。tim_oc1 信号脉冲宽度必须编程为目标定时器的至少 2 个时钟周期，以确保从属定时器检测到触发。例如，如果目标定时器时钟比源定时器慢 4 倍，则 OC1 脉冲宽度必须为 8 个时钟周期。

如：可以配置 TIM_mstr 作为 TIM_slv 的预分频器。参考上图，进行下述操作：

- 配置 TIM_mstr 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 TIM_mst_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 TIM_mstr 的 TRGO1 输出至 TIM_slv，设置 TIM_slv_SMCR 寄存器的 TS='00000'，配置 TIM_slv 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(TIM_slv_SMCR 寄存器的 SMS=111)；这样 TIM_slv 即可由 TIM_mstr 周期性的上升沿(即 TIM_mstr 的计数器溢出)信号驱动。
- 最后，必须设置相应 CEN 位分别启动两个定时器。

注：如果 OCx 已被选中为 TIM_mstr 的触发输出(MMS=1xx)，它的上升沿用于驱动 TIM_slv 的计数器。

使用一个定时器使能另一个定时器

在这个例子中，TIM_slv 的使能由 TIM_mstr 的输出比较控制。只当 TIM_mstr 的 OC1REF 为高时，TIM_slv 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 TIM_mstr 为主模式，送出它的输出比较参考信号(OC1REF)为触发输出(TIM_mstr_CR2 寄存器的 MMS=100)
- 配置 TIM_mstr 的 OC1REF 波形(TIM_mst_CCMR1 寄存器)
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为门控模式(TIM_slv_SMCR 寄存器的 SMS=101)
- 置 TIM_slv_CR1 寄存器的 CEN=1 以使能 TIM_slv
- 置 TIM_mst_CR1 寄存器的 CEN=1 以启动 TIM_mstr

注： TIM_slv 的时钟不与 TIM_mstr 的时钟同步，这个模式只影响 TIM_slv 计数器的使能信号。

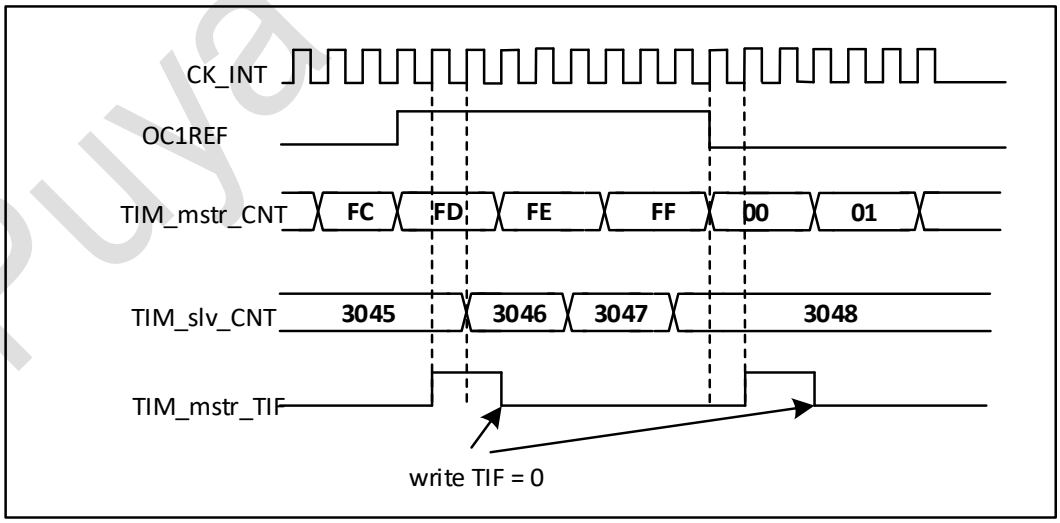


图 11-54 TIM_mstr 的 OC1REF 控制 TIM_slv

在上图的例子中，在 TIM_slv 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 TIM_mstr 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 TIM_mstr 和 TIM_slv。TIM_mstr 是主模式并从 0 开始，TIM_slv 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 TIM_mstr_CR1 的 CEN 位将禁止 TIM_mstr，TIM_slv 随即停止。

- 配置 TIM_mstr 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM_mst_CR2 寄存器的 MMS=100)。
- 配置 TIM_mstr 的 OC1REF 波形(TIM_mstr_CCMR1 寄存器)。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为门控模式(TIM_slv_SMCR 寄存器的 SMS=101)
- 置 TIM_mstr_EGR 寄存器的 UG=‘1’，复位 TIM_mstr。
- 置 TIM_slv_EGR 寄存器的 UG=‘1’，复位 TIM_slv。
- 写‘0XE7’至 TIM_slv 的计数器(TIM_slv_CNT)，初始化它为 0xE7。
- 置 TIM_slv_CR1 寄存器的 CEN=‘1’以使能 TIM_slv。
- 置 TIM_mstr_CR1 寄存器的 CEN=‘1’以启动 TIM_mstr。
- 置 TIM_mstr_CR1 寄存器的 CEN=‘0’以停止 TIM_mstr。

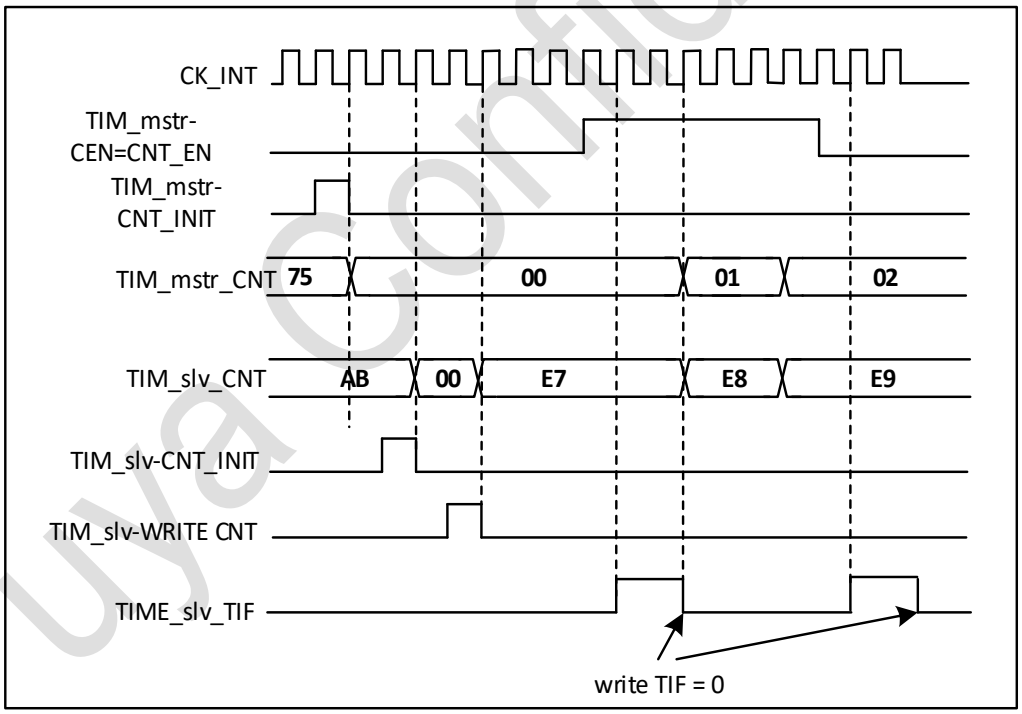


图 11-55 通过使能 TIM_mstr 可以控制 TIM_slv

使用一个定时器去启动另一个定时器

在这个例子中，使用 TIM_mstr 的更新事件使能 TIM_slv。一旦 TIM_mstr 产生更新事件，TIM_slv 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，TIM_slv 的 CEN 位被自动地置‘1’，同时计数器开始计数直到写‘0’到 TIM_slv_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 TIM_mstr 为主模式，送出它的更新事件(UEV)做为触发输出(TIM_mstr_CR2 寄存器的

MMS=010)。

- 配置 TIM_mstr 的周期(TIM_mstr_ARR 寄存器)。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为触发模式(TIM_slv_SMCR 寄存器的 SMS=110)
- 置 TIM_mstr_CR1 寄存器的 CEN=1 以启动 TIM_mstr。

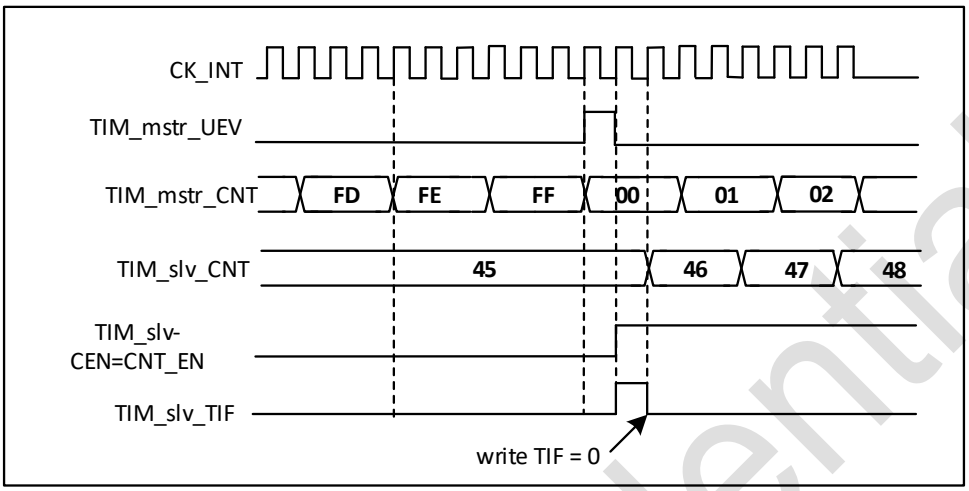


图 11-56 使用 TIM_mstr 的更新触发 TIM_slv

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 0 相同配置情况下，使用触发模式而不是门控模式(TIM_slv_SMCR 寄存器的 SMS=110)的动作。

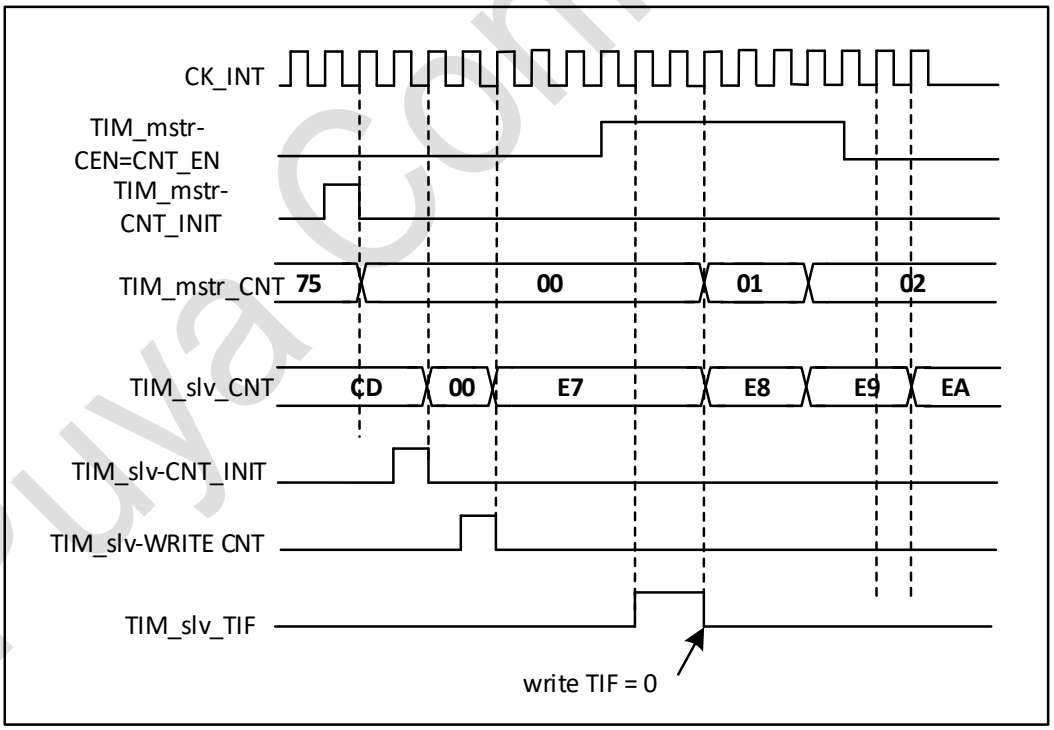


图 11-57 利用 TIM_mstr 的使能触发 TIM_slv

使用一个外部触发同步地启动 2 个定时器

这个例子中当 TIM_mstr 的 TI1 输入上升时使能 TIM_mstr，使能 TIM_mstr 的同时使能 TIM_slv。保证计数器的对齐，TIM_mstr 必须配置为主/从模式(对应 TI1 为从，对应 TIM_slv 为主)：

- 配置 TIM_mstr 为主模式，送出它的使能做为触发输出(TIM_mstr_CR2 寄存器的 MMS=001)。

- 配置 TIM_mstr 为从模式，从 TI1 获得输入触发(TIM_mstr_SMCR 寄存器的 TS=00100)。
- 配置 TIM_mstr 为触发模式(TIM_mstr_SMCR 寄存器的 SMS=110)。
- 配置 TIM_mstr 为主/从模式， TIM_mstr_SMCR 寄存器的 MSM=1。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为触发模式(TIM_slv_SMCR 寄存器的 SMS=110)。

当 TIM_mstr 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 TIM_mstr 的 CNT_EN 和 CK_PSC 之间有个延迟。

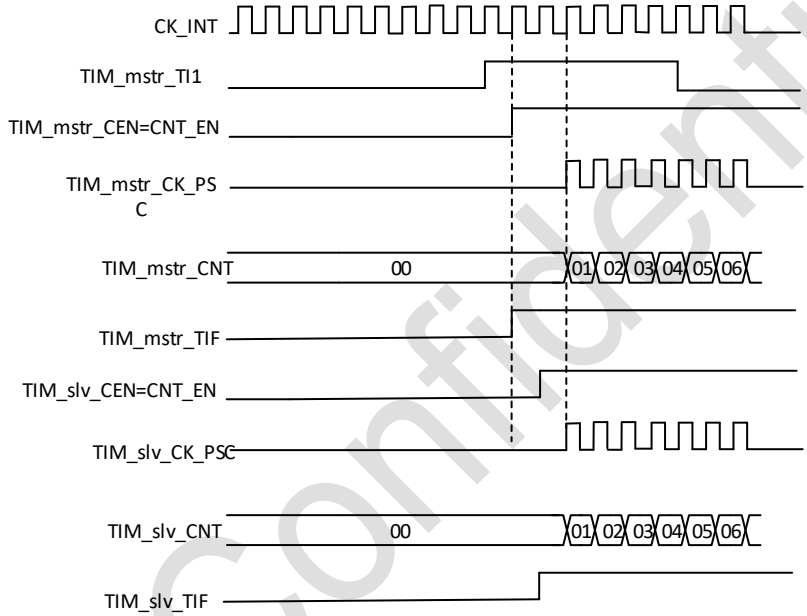


图 11-58 使用 TIM_mstr 的 TI1 输入触发 TIM_mstr 和 TIM_slv

11.3.20. 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。

11.4. TIM1 寄存器描述

11.4.1. TIM1 控制寄存器 1 (TIM1_CR1)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN

-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	00	时钟分频因子 这 2 位定义在定时器时钟(CK_INT)频率，死区时间和由死区发生器与数字滤波器(ETR，Tix)所用的采样时钟之间的分频比例 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0：TIM1_ARR 寄存器没有缓冲 1：TIM1_ARR 寄存器被装入缓冲器
6:5	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。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。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	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。

Bit	Name	R/W	Reset Value	Function
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请求，则下述任一事件产生一个更新中断请求： – 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请求，则只有计数器溢出/下溢产生一个更新中断请求
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。

11.4.2. TIM1 控制寄存器 2 (TIM1_CR2)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	MMS[3]	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	RW	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			Res	CCUS	Res	CCPC
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	-	RW

Bit	Name	R/W	Reset Value	Function
31：26	Reserved	-	-	保留

Bit	Name	R/W	Reset Value	Function
25	MMS[3]	RW	0	详见 MMS[2:0]描述
24:15	Reserved	-	-	保留
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。

Bit	Name	R/W	Reset Value	Function
				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	Reserved	-	-	保留
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	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位 0：CcxE， CcxNE 和 OCxM 位不是预装载的。 1：CcxE， CcxNE 和 OCxM 位是预装载的；设置该位后，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。

11.4.3. TIM1 从模式控制寄存器 (TIM1_SMCR)

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	TS[4:3]		Res	Res	Res	SMS[3]
-	-	-	-	-	-	-	-	-	-	RW		-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[4:0]	RW	0	详见 TS 描述
19:17	Reserved	-	-	保留
16	SMS[3]	RW	0	详见 SMS 描述
15	ETP	RW	0	外部触发极性。该位选择是否 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。这位使能外部时钟模式 2 0：外部时钟模式 2 不使能 1：外部时钟模式 2 使能，计数器工作在 ETRF 信号的有效沿
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 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 f_{DTS} 下采样 0001：$f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：$f_{SAMPLING}=f_{CK\_INT}/2$，N=6 0101：$f_{SAMPLING}=f_{CK\_INT}/2$，N=8 0110：$f_{SAMPLING}=f_{CK\_INT}/4$，N=6 0111：$f_{SAMPLING}=f_{CK\_INT}/4$，N=8 1000：$f_{SAMPLING}=f_{CK\_INT}/8$，N=6 1001：$f_{SAMPLING}=f_{CK\_INT}/8$，N=8 1010：$f_{SAMPLING}=f_{CK\_INT}/16$，N=5 1011：$f_{SAMPLING}=f_{CK\_INT}/16$，N=6 1100：$f_{SAMPLING}=f_{CK\_INT}/16$，N=8 1101：$f_{SAMPLING}=f_{CK\_INT}/32$，N=5 1110：$f_{SAMPLING}=f_{CK\_INT}/32$，N=6 1111：$f_{SAMPLING}=f_{CK\_INT}/32$，N=8 必须关注当 ETF[3:0] = 1 或者 2 或者 3 时，f_{DTS} 被方程式中的 CK_INT 代替

Bit	Name	R/W	Reset Value	Function
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：保留(ITR0) 001：保留 010：TIM14_CH1 (ITR2) 011：PWM_CH1 (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	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

Bit	Name	R/W	Reset Value	Function
				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。

TIM1 内部触发连接

Slave TIM	ITR0(TS=000)	ITR1(TS=001)	ITR2(TS=010)	ITR3(TS=011)
TIM1	TIM14	保留	保留	保留

11.4.4. TIM1 中断使能寄存器 (TIM1_DIER)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW

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

Bit	Name	R/W	Reset Value	Function
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

11.4.5. TIM1 状态寄存器(TIM1_SR)

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Res	Res.	Res.	Res	IC2IR	IC1IR	Res	Res.
-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	-	-	-	-	Rc_w0	Rc_w0	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res.	SBIF	CC4OF	CC3OF	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1F	UIF
-	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	-	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0	Rc_w0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
28	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
27	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
26	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获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：无下降沿捕获事件产生； 1：发生下降沿捕获事件。
25	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
24	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
23:20	Reserved	-	-	保留

19	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
18	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清'0'或通过读 TIMx_CCR1 清'0'。 0：无上升沿捕获事件产生； 1：发生上升沿捕获事件。
17:14	Reserved	-	-	保留
13	SBIF	RC_W0	0	系统刹车中断标志 当系统刹车输入一旦有效，该标志信号会被硬件置位。系统刹车输入无效后可通过软件清零。 该标志必须复位以使 PWM 重新开始工作。 0:无刹车事件发生。 1:系统刹车输入处于有效。如果 TIMx DIER 寄存器的 BIE 位=1 则会产生中断。
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：CC1IF 置 1 时，计数器的值被捕获到 TIM1_CCR1 寄存器。
8	Reserved	-	-	保留
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
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。或者当 TIM1_CCR1 的值大于 TIM1_ARR（向上计数和中央对齐计数的上溢时、向下计数的下溢时 CC1IF 置 1）。 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置 1，它由软件清 0 或通过读 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)。
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)）

11.4.6. TIM1 事件产生寄存器(TIM1_EGR)

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
-	-	-	--	-	-	-	-	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清 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： TIM1_SR 寄存器的 TIF=1，若开启对应的中断，则产生相应的中断。
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

Bit	Name	R/W	Reset Value	Function
				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，则产生相应的中断。 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，则产生相应的中断。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。

11.4.7. TIM1 捕获/比较模式寄存器 1(TIM1_CCMR1)

偏移地址：0x18

复位值：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	OC2M[3]	Res	Res	Res	Res	Res	Res	Res	OC1M[3]
-	-	-	-	-	-	-	RW	-	-	-	-	-	-	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OC2CE	OC2M[2:0]			OC2PE	CO2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC2M[3]	RW	0	见 OC2M 描述
23:17	Reserved	-	-	保留
16	OC1M[3]	RW	0	见 OC1M 描述
15	OC2CE	RW	0	输出比较 2 清 0 使能

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。 0000：冻结。输出比较寄存器 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用； 0001：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。 0010：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。 0011：翻转。当 TIM1_CCR1=TIM1_CNT 时，翻转 OC1REF 的电平。 0100：强制为无效电平。强制 OC1REF 为低。 0101：强制为有效电平。强制 OC1REF 为高。 0110： 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)。 0111： 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。 1000：可恢复 OPM 模式 1-在递增计数模式下，通道处于有效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且通道在下一次更新时再次有效。在递减计数模式下，通道处于无效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且信道在下一次更新时再次变为不活动。 1001：可恢复 OPM 模式 2-在递增计数模式下，信道处于无效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且通道在下一次更新时再次变为无效。在下计数模式下，信道处于有效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且信道在下一次更新时再次变为有效。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。 注 3：使用可恢复 OPM 模式时，计数模式不要配置为中央计数模式。
3	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 TIM1_CCR1 寄存器的预装载功能，可随时写入 TIM1_CCR1 寄存器，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。

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**
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
15:12	IF2F	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 f_{DTS} 采样 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2

Bit	Name	R/W	Reset Value	Function
				1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，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 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。

11.4.8. TIM1 捕获/比较模式寄存器 2(TIM1_CCMR2)

偏移地址：0x1C

复位值：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	OC4M[3]	Res	Res	Res	Res	Res	Res	Res	OC3M[3]
-	-	-	-	-	-	-	RW	-	-	-	-	-	-	-	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	CO4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

输出比较模式：

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC4M[3]	RW	0	参见 OC4M 描述
23:17	Reserved	-	-	保留
16	OC3M[3]	RW	0	参见 OC3M 描述
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； 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。

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	RW	0000	输入捕获 4 滤波器
11:10	IC4PSC	RW	00	输入/捕获 4 预分频器
9:8	CC4S	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	RW	0000	输入捕获 3 滤波器
3:2	IC3PSC	RW	00	输入/捕获 3 预分频器
1:0	OC3S	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。

11.4.9. TIM1 捕获/比较使能寄存器（TIM1_CCER)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CC4NP	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	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15	CC4NP	RW	0	捕获/比较 4 互补输出极性。参考 CC1NP 的描述。
14	Reserved	-	-	保留
13	CC4P	RW	0	输入/捕获 4 输出极性。参考 CC1P 的描述。
12	CC4E	RW	0	输入/捕获 4 输出使能。参考 CC1E 的描述。
11	CC3NP	RW	0	输入/捕获 3 互补输出极性。参考 CC1NP 的描述。
10	CC3NE	RW	0	输入/捕获 3 互补输出使能。参考 CC1NE 的描述。
9	CC3P	RW	0	输入/捕获 3 输出极性。参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能。参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	CC2NE	RW	0	输入/捕获 2 互补输出使能。参考 CC1NE 的描述。
5	CC2P	RW	0	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，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）。

Bit	Name	R/W	Reset Value	Function
				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。 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 TIM1_CCR1 寄存器。 0：捕获禁止 1：捕获使能

表 11-2 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

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 OC-REF）+ 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 OC-REF）+ polarity + dead-time OCN_EN=1
0	0	X	0	0	输出禁止(与定时器断开)，OCx=CCxP， OCx_EN=0	输出禁止(与定时器断开)， OCxN=CCxNP，OCxN_EN=0
	0		0	1	输出禁止(与定时器断开) 异步的：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		1	0		
	0		1	1		
	1		0	0	输出禁止（与定时器断开） OCx=CCxP，OCx_EN=0	输出禁止（与定时器断开）， OCxN=CCxNP，OCxN_EN=0
	1		0	1	关闭状态（输出使能且为无效电平） 异步的：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		

11.4.10. TIM1 计算器(TIM1_CNT)

偏移地址：0x24

复位值：0x0000 0000

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

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

15:0	CNT[15:0]	RW	0	计数器的值

11.4.11. TIM1 预分频器 (TIM1_PSC)

偏移地址：0x28

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	0	预分频器的值 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。

11.4.12. TIM1 自动重新加载寄存器 (TIM1_ARR)

偏移地址：0x2c

复位值：0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	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。

11.4.13. TIM1 重复计数器寄存器(TIM1_RCR)

偏移地址：0x30

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：8	Reserved	-	-	保留
7:0	REP[7:0]	RW	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；

11.4.14. TIM1 捕获/比较寄存器 1(TIM1_CCR1)

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1_H[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**
CCR1[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	CCR1_H[15:0]	RW	0	比较 1 的值

				非移相模式（PWM_PHS_EN=0）： CCR1_H 无效 移相模式（PWM_PHS_EN=1）： CCR1_H 包含了中央对齐模式中向下计数时装入比较 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 TIM1_CNT 比较的值，并且在 OC1 端口上输出信号。
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 TIM1_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

11.4.15. TIM1 捕捉/比较寄存器 2(TIM1_CCR2)

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2_H[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**
CCR2[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	CCR2_H[15:0]	RW	0	比较 2 的值 非移相模式（PWM_PHS_EN=0）： CCR2_H 无效

				移相模式（PWM_PHS_EN=1）： CCR2_H 包含了中央对齐模式中向下计数时装入比较 2 寄存器的值（预装载值）。 如果在 TIM1_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 TIM1_CNT 比较的值，并且在 OC 端口上输出信号。
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 TIM1_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 TIM1_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。

11.4.16. TIM1 捕获/比较寄存器 3 (TIM1_CCR3)

偏移地址：0x3C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3_H[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**
CCR3[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	CCR3_H[15:0]	RW	0	比较 3 的值 非移相模式（PWM_PHS_EN=0）： CCR3_H 无效 移相模式（PWM_PHS_EN=1）：

				CCR3_H 包含了中央对齐模式中向下计数时装入比较 3 寄存器的值（预装载值）。 如果在 TIM1_CCMR2 寄存器(OC3PE 位)中未选择预装载特性，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。
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。

11.4.17. TIM1 捕捉/比较寄存器 4(TIM1_CCR4)

偏移地址：0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	CCR4[15:0]	RW	0	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）。

Bit	Name	R/W	Reset Value	Function
				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。

11.4.18. TIM1 刹车和死区寄存器(TIM1_BDTR)

偏移地址：0x44

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。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；

Bit	Name	R/W	Reset Value	Function
				1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入（BRK 及 BRK_ACTH）； 1：开启刹车输入（BRK 及 BRK_ACTH）。 注：一旦 LOCK 级别(TIM1_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
11	OSSR	RW	0	运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明（捕获/比较使能寄存器(TIM1_CCER)）。 0：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)）。 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，

Bit	Name	R/W	Reset Value	Function
				TIM1_CCER 寄存器的 CCxP/CCxNP 位）以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，TIM1_CCMRx 寄存器的 OCxM/OCxPE 位）； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIM1_BDTR 寄存器，则其内容冻结直 至复位。
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 1μs； 64us 到 126us，若步长时间为 2μs； 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。

11.4.19. TIM1 输入选择寄存器（TIM1_TISEL）

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
-	-	-	-	RW	RW	RW	RW	-	-	-	-	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
-	-	-	-	RW	RW	RW	RW	-	-	-	-	RW	RW	RW	RW

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

27:24	TI4SEL	RW	0	TI4 输入选择. 0000：TIM1_CH4 其他：保留
23:20	Reserved	-	-	保留
19:16	TI3SEL	RW	0	TI3 输入选择. 0000：TIM1_CH3 其他：保留
15:12	Reserved	-	-	保留
11:8	TI2SEL	RW	0	TI2 输入选择. 0000：TIM1_CH2 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL	RW	0	TI1 输入选择. 0000：TIM1_CH1 其他：保留

11.4.20. TIM1 备用选项寄存器 1（TIM1_AF1）

偏移地址：0x60

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWM_PHS_EN	INTR_SEL		ETRSEL[3:2]	
-	-	-	-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETR-SEL[1:0]		Res.	Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW	-		RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20	PWM_PHS_EN	RW	0	PWM 移相使能 0：关 1：开 注：仅在中央对齐模式中该位可配置
19:18	INTR_SEL	RW	0	更新中断选择 00:波峰或波谷中断 01:波峰中断

				10:波谷中断 11:保留 注：仅在中央对齐模式中该位可配置
17:14	ETRSEL	RW	0	外部触发源选择（etr_in source selection） 该位段用于选择 ETR 输入源 0000：TIM1_ETR 0001：保留 0010：保留 0011：ADC_AWD_OUT 0100：LSE_CSS_OUT 0101：保留 0110：保留 0111：保留 其它：保留 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
13:12	Reserved	-	-	保留
11	BKCMP2P	RW	0	brk_cmp2 输入极性。 该位选择 brk_cmp2 输入极性，必须与 BKP 极性位同时配置 0：brk_cmp2 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：brk_cmp2 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	brk_cmp1 输入极性。 该位选择 brk_cmp1 输入极性，必须与 BKP 极性位同时配置 0：brk_cmp1 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：brk_cmp1 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效）

				注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	brk_cmp2 输入使能。 该位用于刹车输入时使能 brk_cmp2。brk_cmp2 输入与其它刹车源进行或逻辑作为刹车输入。 0：brk_cmp2 输入关闭 1：brk_cmp2 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	brk_cmp1 输入使能。 该位用于刹车输入时使能 brk_cmp1。brk_cmp1 输入与其它刹车源进行或逻辑作为刹车输入。 0：brk_cmp1 输入关闭 1：brk_cmp1 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能。 该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

12. 通用定时器 (TIM14)

12.1. TIM14 简介

通用定时器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。

12.2. 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

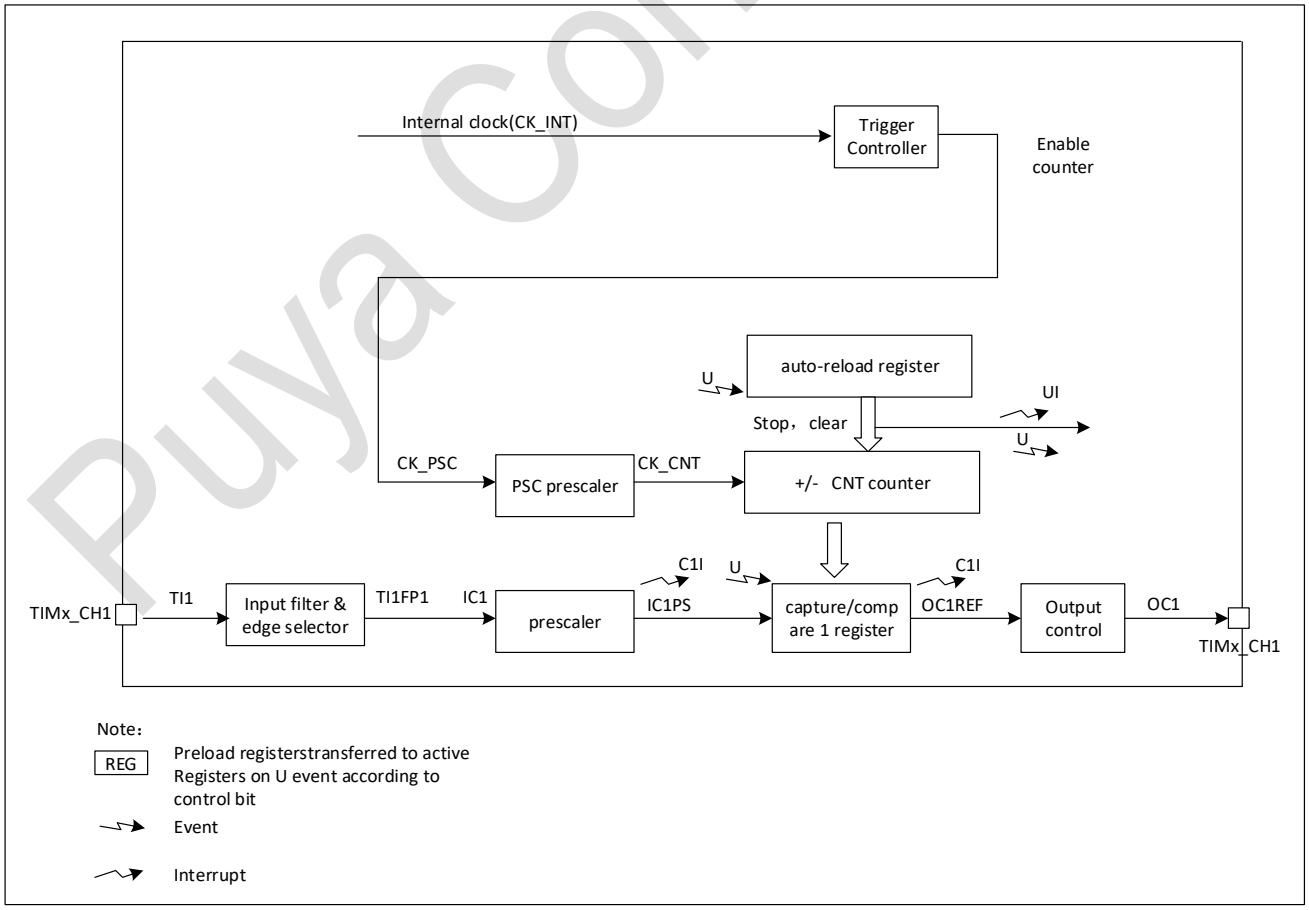


图 12-1 TIM14 架构框图

12.3. TIM14 功能描述

12.3.1. 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 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 65536 之间的任意值分频。它是基于一个（在 TIMx_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

下图给出了在计数器运行时，更改预分频器参数的例子。

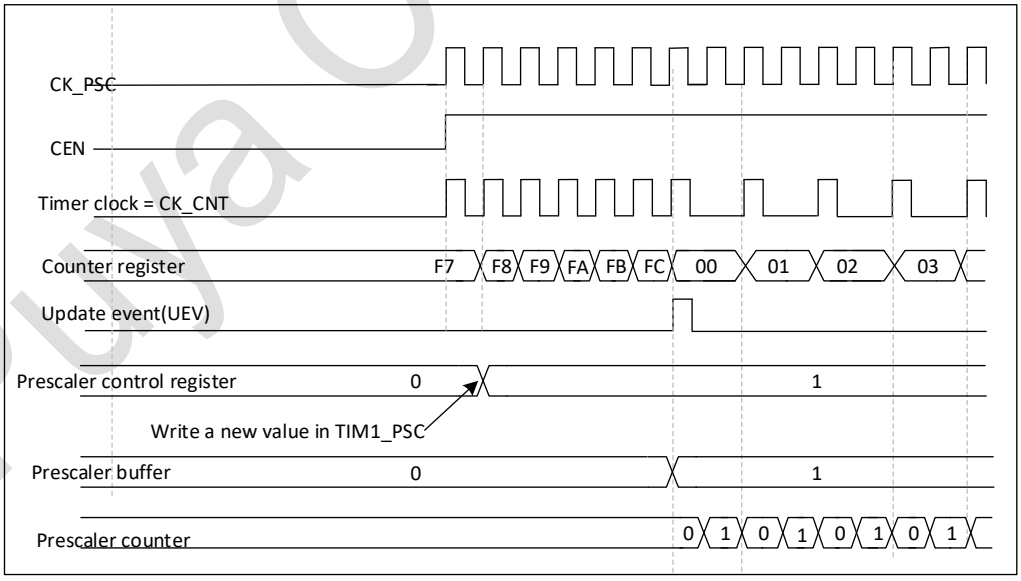


图 12-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

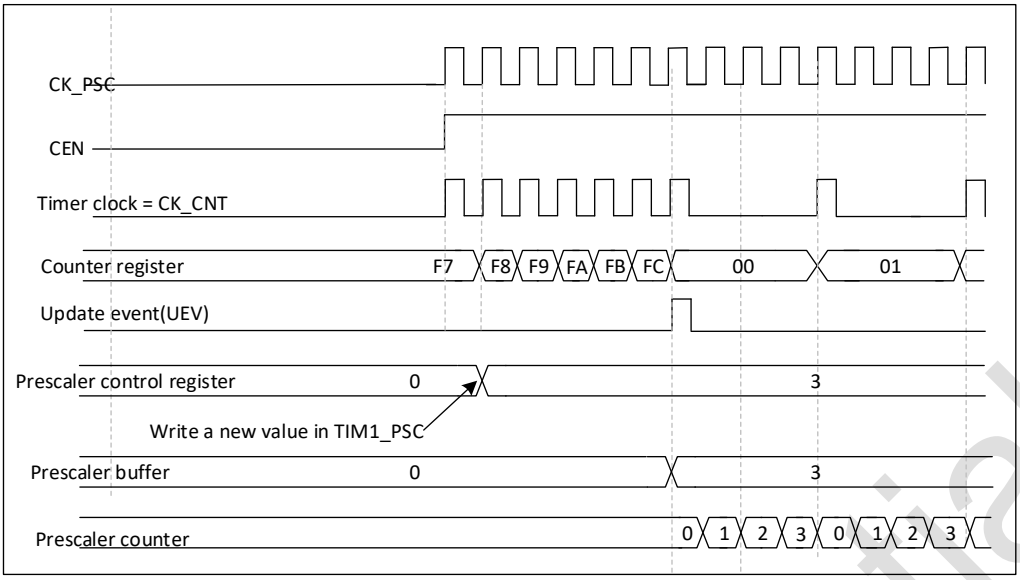


图 12-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

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，并产生一个计数器溢出事件。

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)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 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)。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.

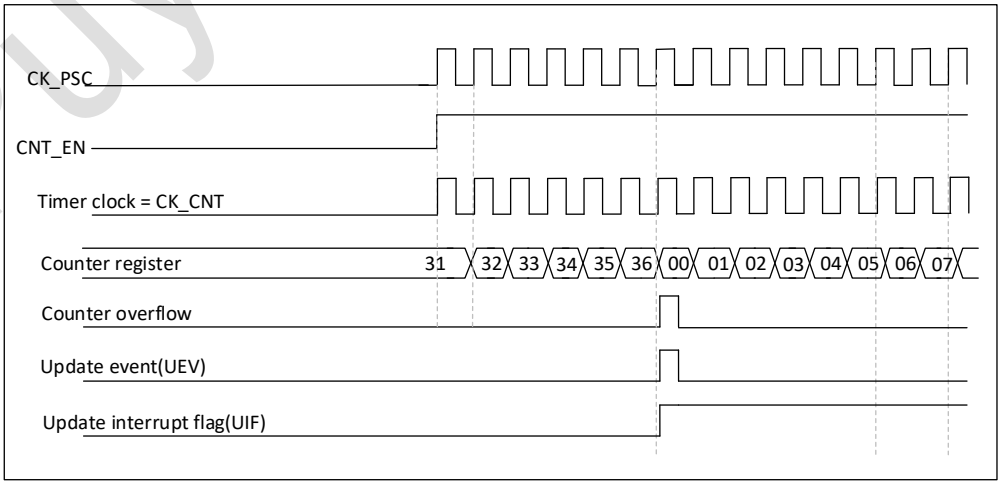


图 12-4 计数器时序图，内部时钟分频因子为 1

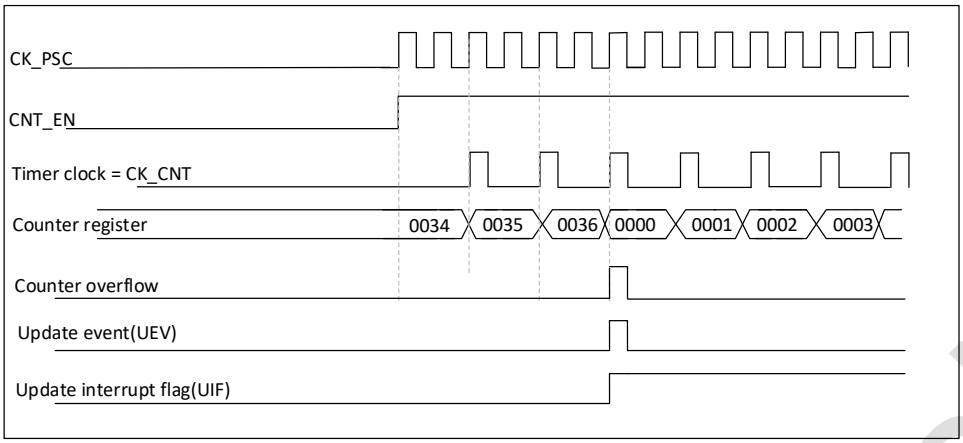


图 12-5 计数器时序图，内部时钟分频因子为 2

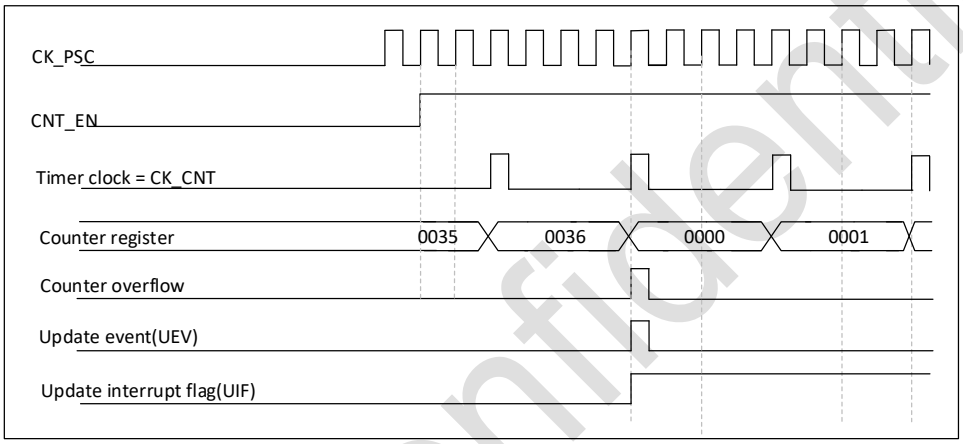


图 12-6 计数器时序图，内部时钟分频因子为 4

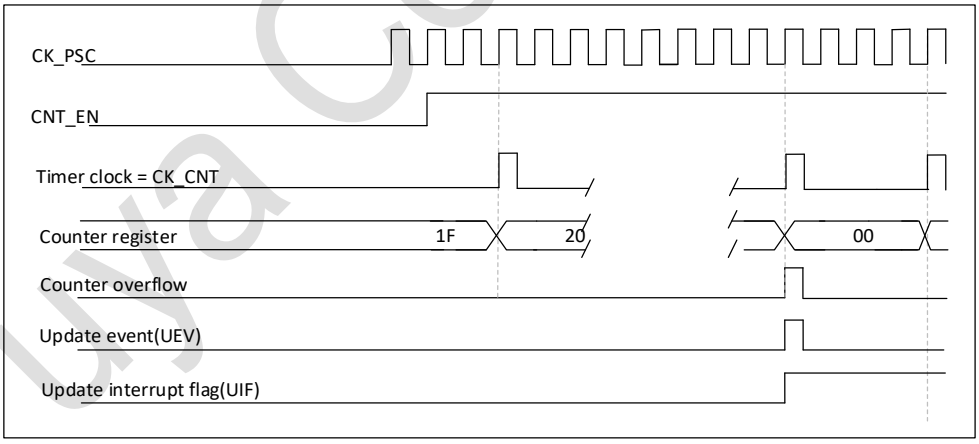


图 12-7 计数器时序图，内部时钟分频因子为 N

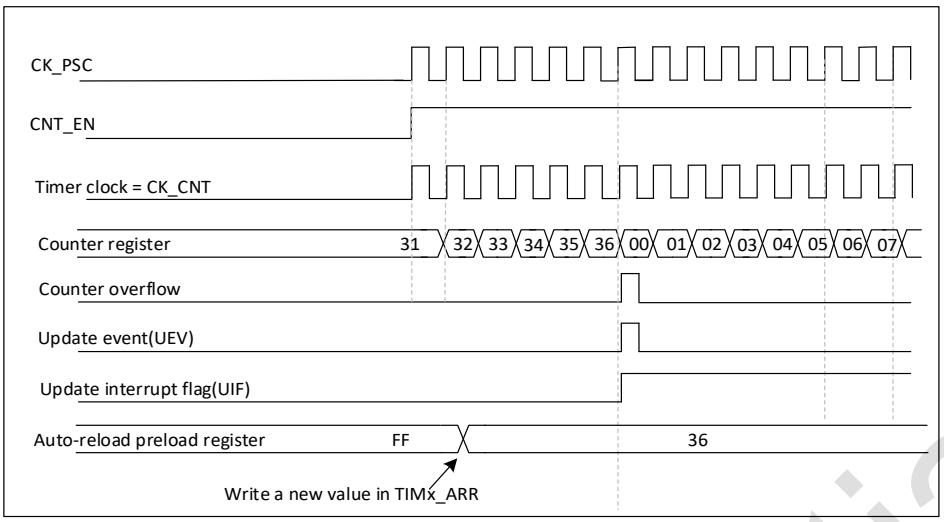


图 12-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

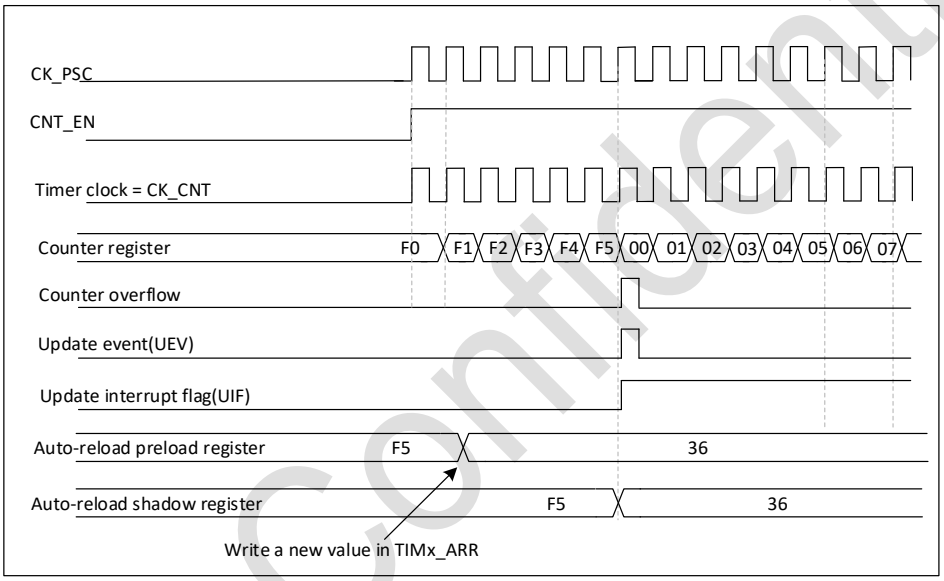


图 12-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

12.3.2. 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。

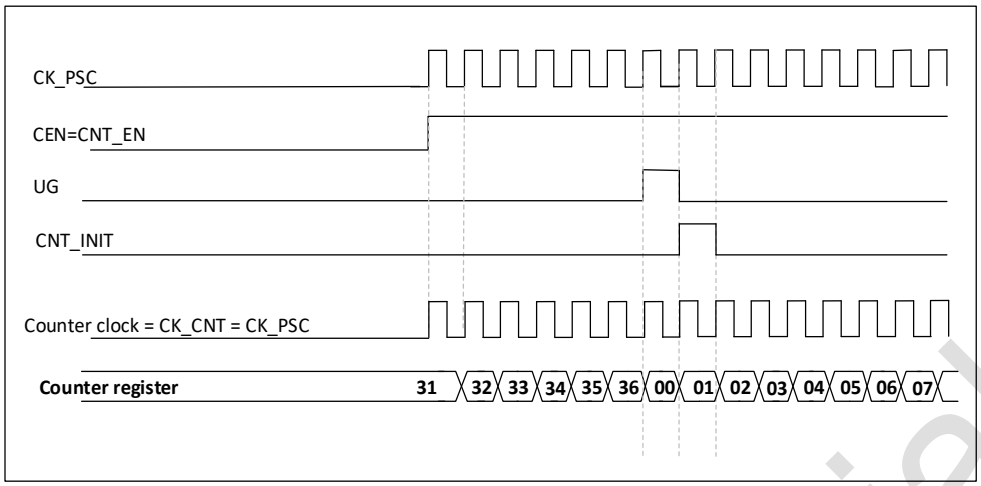


图 12-10 一般模式下的控制电路，内部时钟分频因子为 1

12.3.3. 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)。

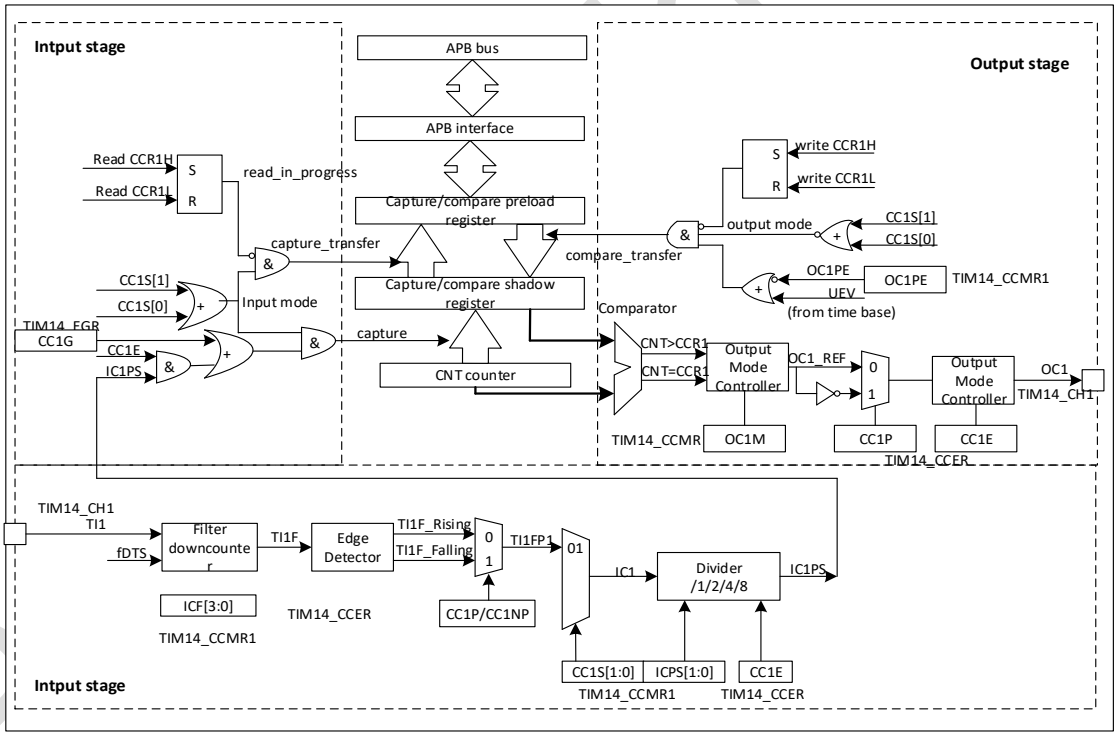


图 12-11 TIM14 捕获/比较通道图

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。

12.3.4. 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_CCR1）中。当捕获事件发生时，相应的 CcxIF 标志（TIMx_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 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输出端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为 00 时，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宽度（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 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 TIMx_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1.
- 如设置了 CC1E 位，则会产生一个中断请求。

为了处理中断溢出，建议在读出中断溢出标志之前，读取数据。这是为了避免丢失在读出中断溢出标志之后和读取数据之前可能产生的中断溢出信息。

注：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。

12.3.5. 强置输出模式

在该模式下（TIMx_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 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强制输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（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 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。这将会在下面的输出比较模式一节中介绍。

12.3.6. 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(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。

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。

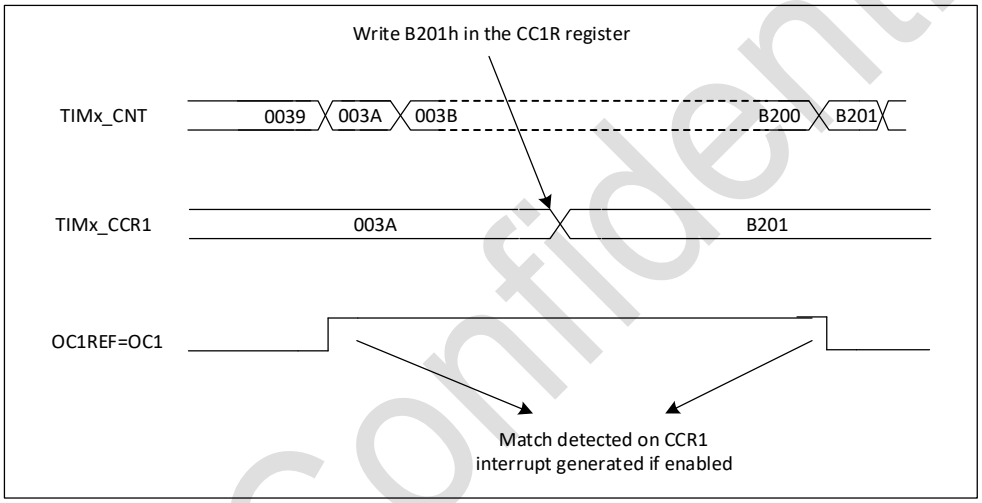


图 12-12 输出比较模式，翻转 OC1

12.3.7. 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 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。必须设置 TIMx_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 TIMx_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 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或者低电平有效。TIMx_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），TIMx_CNT 和 TIMx_CCRx 始终在进行比较，以确定是否符合 TIMx_CNT ≤ TIMx_CCRx。

定时器仅当计数器是向上计数时才能够产生边沿对齐模式的 PWM。

PWM 边沿对齐模式

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。

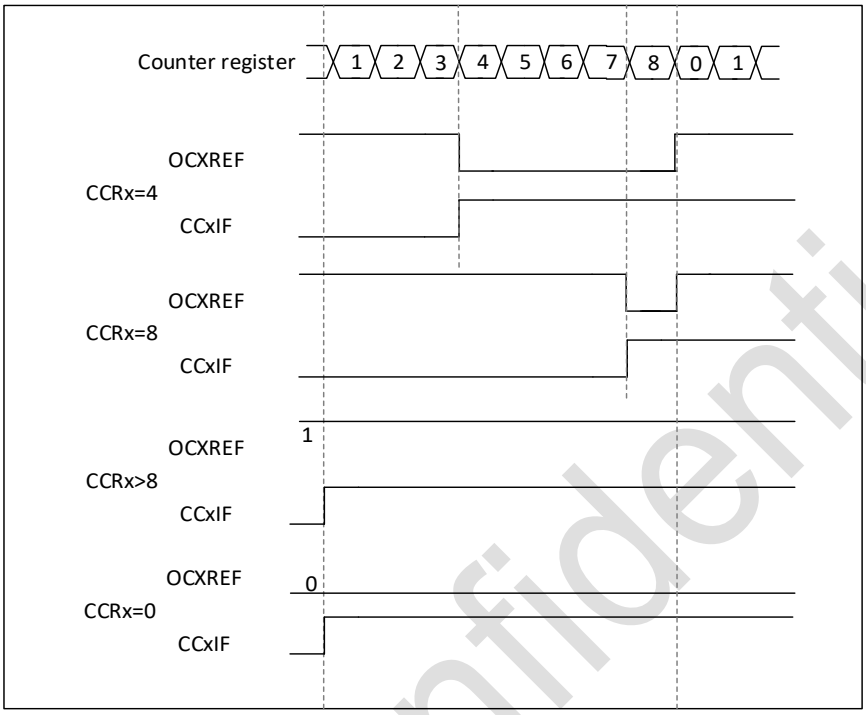


图 12-13 边沿对齐的 PWM 波形(ARR=8)

12.3.8. 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。

12.3.9. 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。

12.4. TIM14 寄存器

12.4.1. TIM14 控制寄存器 1（TIM14_CR1）

偏移地址：0x00

复位值：0x0000 0000

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

Res	Res	Res	Res	Res	Res	CKD[1:0]	ARPE	Res	Res	Res	URS	UDIS	CEN
-	-	-	-	-	-	RW	RW	-	-	-	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	00	时钟分频因子，这 2 位定义在定时器时钟(CK_INT)频率，所用的采样时钟之间的分频比例 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0：TIM14_ARR 寄存器没有缓冲 1：TIM14_ARR 寄存器被装入缓冲器
6:3	Reserved	-	0	保留，一直读为 0
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请求，则下述任一事件产生一个更新中断请求： – 计数器溢出/下溢 – 设置 UG 位 1：如果允许产生更新中断请求，则只有计数器溢出/下溢产生一个更新中断请求
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。

12.4.2. TIM14 中断使能寄存器（TIM14_DIER）

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1IE	UIE
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RW	RW

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	保留
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

12.4.3. TIM14 状态寄存器（TIM14_SR)

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	IC1IF	Res	Res	Res	IC1IR
-	-	-	-	-	-	-	-	-	-	-	RC_W0	-	-	-	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	CC1OF	Res	Res	Res	Res	Res	Res	Res	CC1IF	UIF
-	-	-	-	-	-	RC_W0	-	-	-	-	-	-	-	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31：21	Reserved	-	-	保留
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’。

Bit	Name	R/W	Reset Value	Function
				0：无重复捕获产生； 1：发生下降沿捕获事件。
19:17	Reserved	-	-	保留
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。
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	Reserved	-	-	保留
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： − 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)；

12.4.4. TIM14 事件产生寄存器（TIM14_EGR)

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1G	UG
-	-	-	-	-	-	-	-	-	-	-	-	-	-	W	W

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	保留
1	CC1G	W	0	产生捕获/比较 1 事件，该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： **若通道 CC1 配置为输出：** 设置 CC1IF=1，若开启对应的中断，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)。

12.4.5. TIM14 捕获/比较模式寄存器 1（TIM14_CCMR1)

偏移地址：0x18

复位值：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	Res	Res	Res	Res	Res	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
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 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。
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；

Bit	Name	R/W	Reset Value	Function
				01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：保留； 11：保留。 注：CC1S 仅在通道关闭时(TIM14_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**
Res	Res	Res	Res	Res	Res	Res	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
31:8	Reserved	-	-	保留
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 f_{DTS} 采样 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，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。

Bit	Name	R/W	Reset Value	Function
				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：保留 11：保留 注：CC1S 仅在通道关闭时(TIM14_CCER 寄存器的 CC1E=0)才是可写的。

12.4.6. TIM14 捕获/比较使能寄存器（TIM14_CCER)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CC1NP	Res	CC1P	CC1E
												RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31：4	Reserved	-	-	保留
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 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	-	-	保留
1	CC1P	RW	0	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效 1：OC1 低电平有效 CC1 通道配置为输入：

Bit	Name	R/W	Reset Value	Function
				CC1NP/CC1P 两位选择是 TI1FP1 还是 TI2FP1 的极性信号作为触发或捕获信号。 00：不反相/上升沿：捕获发生在 TixFP1 的上升沿(捕获，复位触发，外部时钟或触发模式)； 01：反相/下降沿：捕获发生在 TixFP1 的下降沿(捕获，复位触发，外部时钟或触发模式)； 10：保留，无效配置。 11：不反向，双边沿。
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

12.4.7. TIM14 计数器（TIM14_CNT）

偏移地址：0x24

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	0	计数器的值

12.4.8. TIM14 预分频器（TIM14_PSC）

偏移地址：0x28

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	0	预分频器的值 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。

12.4.9. TIM14 自动重装载寄存器（TIM14_ARR）

偏移地址：0x2C

复位值：0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	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。 详细参考 12.3.1：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。

12.4.10. TIM14 捕获/比较寄存器 1（TIM14_CCR1）

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	CCR1[15:0]	RW	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。

12.4.11. TIM14 输入选择寄存器（TIM14_TISEL）

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1SEL[3:0]			
-	-	-	-	-	-	-	-	-	-	-	-	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	TI1SEL	RW	0	定时器输入 1 选择 通过软件置位和清零。

				0000：TIM14 通道 1 连接到 GPIO，具体参考数据手册的复用功能。 0001：保留 0010：TIM14 通道 1 连接到 MCU（MCO）。这个配置是通过 RCC_CFG 寄存器的 MCO[2:0]的设置来决定的。 0011：LSE_CSS_OUT 0100：LSI 其它保留

13. 专用脉冲宽度调制(PWM)

13.1. PWM 简介

PWM 模块通过可编程的周期寄存器、占空比寄存器和一个 10 位主计数器来实现 1 路 PWM 的输出，计数器由一个可编程的预分频器驱动。

13.2. PWM 主要特性

- 10bit 向上、向下或者向上向下的自动重装载计数器
- 可编程周期和占空比
- 可编程分频器，允许对计数器的时钟频率进行 1 到 255 的分频
- 1 个独立的通道
- 支持 LED 级联模式
- 输出极性可配置
- 支持边沿对齐和中心对齐
- 不支持 DMA
- 中断事件

13.3. PWM 功能描述

13.3.1. PWM 模块框图

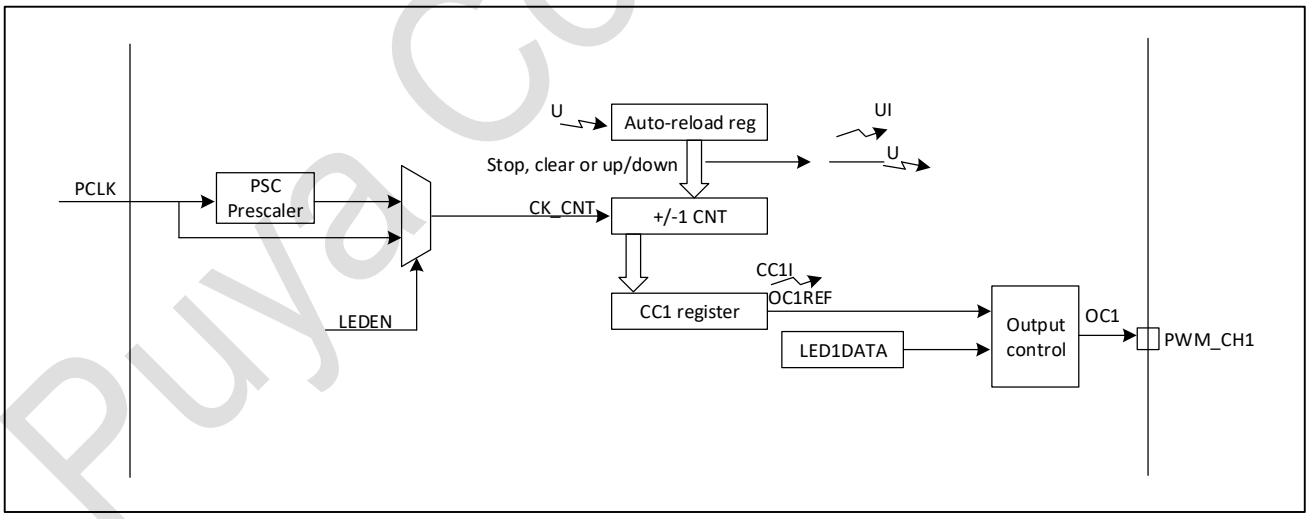


图 13-1 PWM 模块框图

13.3.2. 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 10 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（PWM_CNT）
- 预分频寄存器（PWM_PSC）
- 自动重装载寄存器（PWM_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 PWM_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 PWM_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 PWM_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 PWM_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频率按 1 到 65535 之间的任意值分频。它是基于一个(在 PWM_PSC 寄存器中的)16 位寄存器控制的 3 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频参数将在下一次更新事件到来时被采用。

下图给出了在计数器运行时，更改预分频器参数的例子。

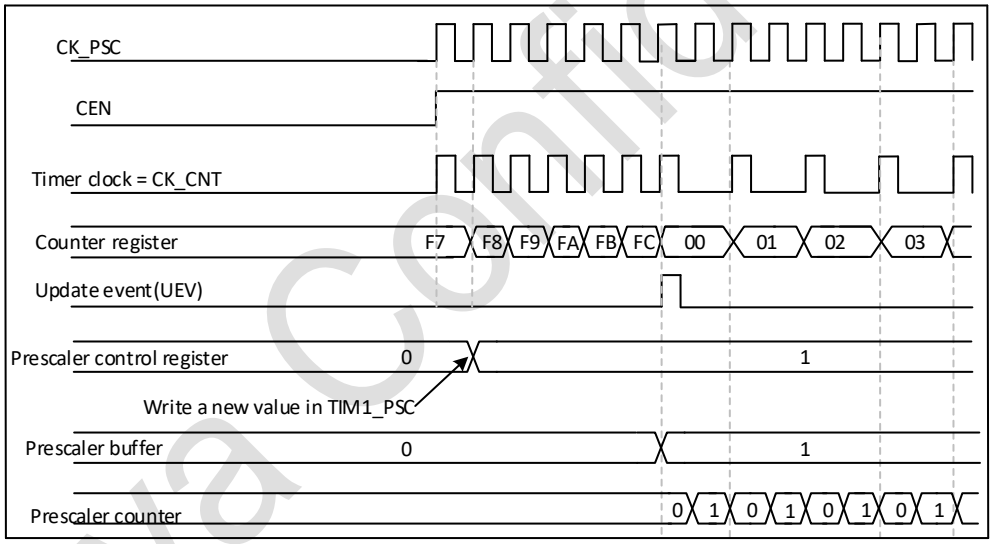


图 13-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

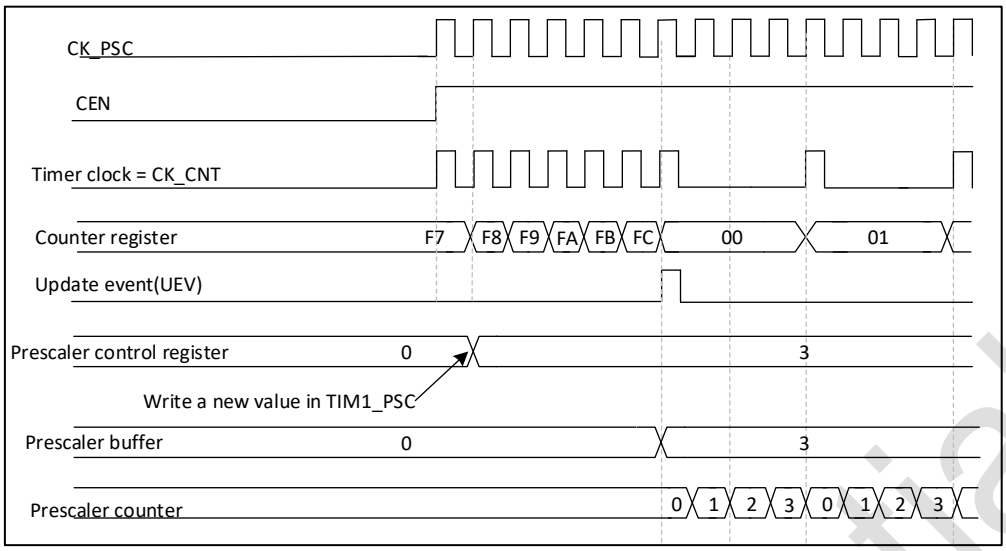


图 13-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

13.3.3. 计数器模式

向上计数模式

计数器从 0 计数到自动装载值（PWM_ARR 寄存的值），然后又从 0 重新开始计数，并产生一个计数器溢出事件。

每个计数溢出时，产生更新事件。在 PWM_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 PWM_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 PWM_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)。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(PWM_SR 寄存器中的 UIF 位)。

- 自动装载影子寄存器被重新置入预装载寄存器的值（PWM_ARR）。
- 预分频器的缓冲区被置入预装载寄存器的值（PWM_PSC 寄存器的内容）。

下面的例程显示了几个在不同频率下的计数器行为，当 PWM_ARR=0X36.

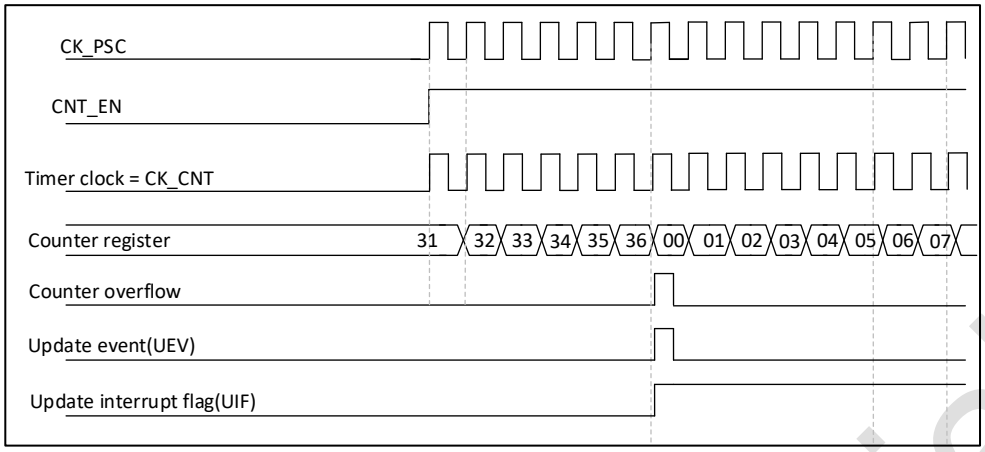


图 13-4 计数器时序图，内部时钟分频因子为 1

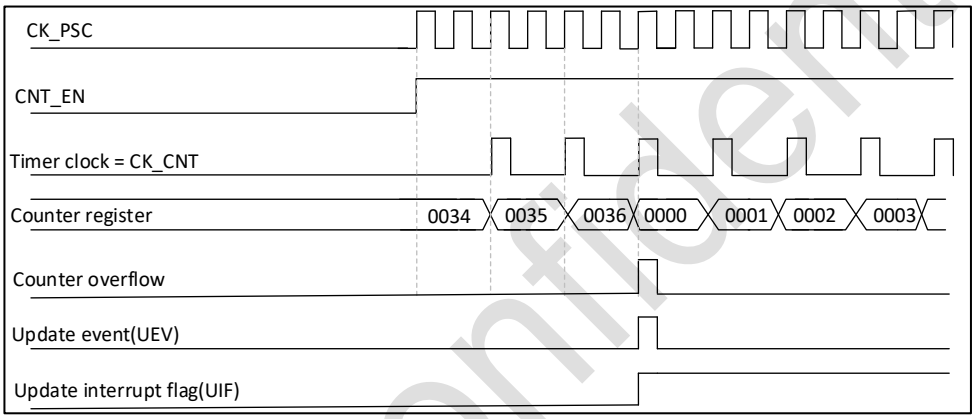


图 13-5 计数器时序图，内部时钟分频因子为 2

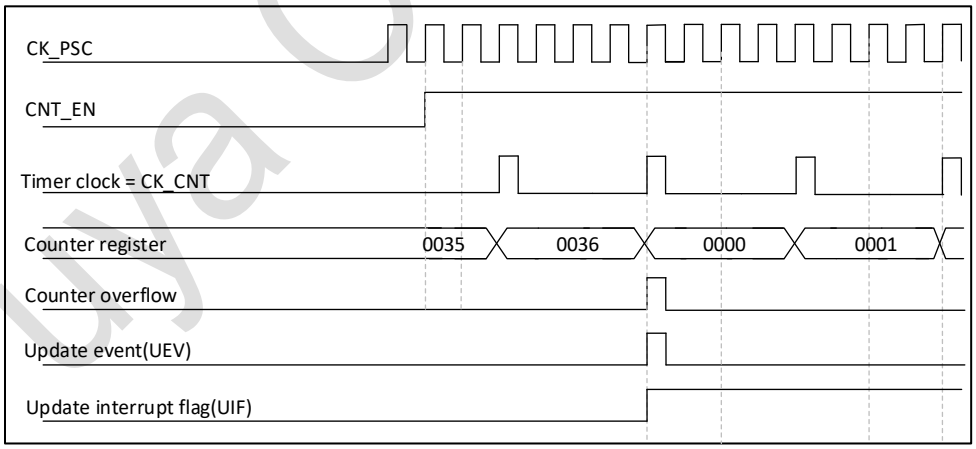


图 13-6 计数器时序图，内部时钟分频因子为 4

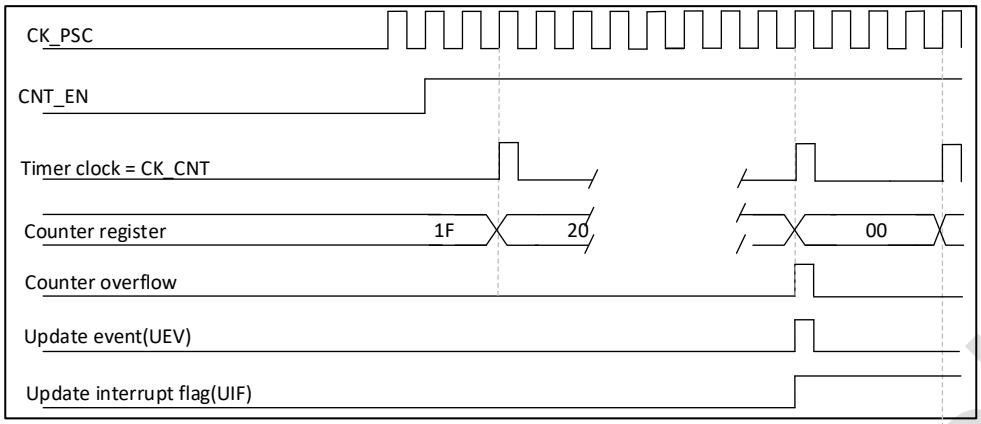


图 13-7 计数器时序图，内部时钟分频因子为 N

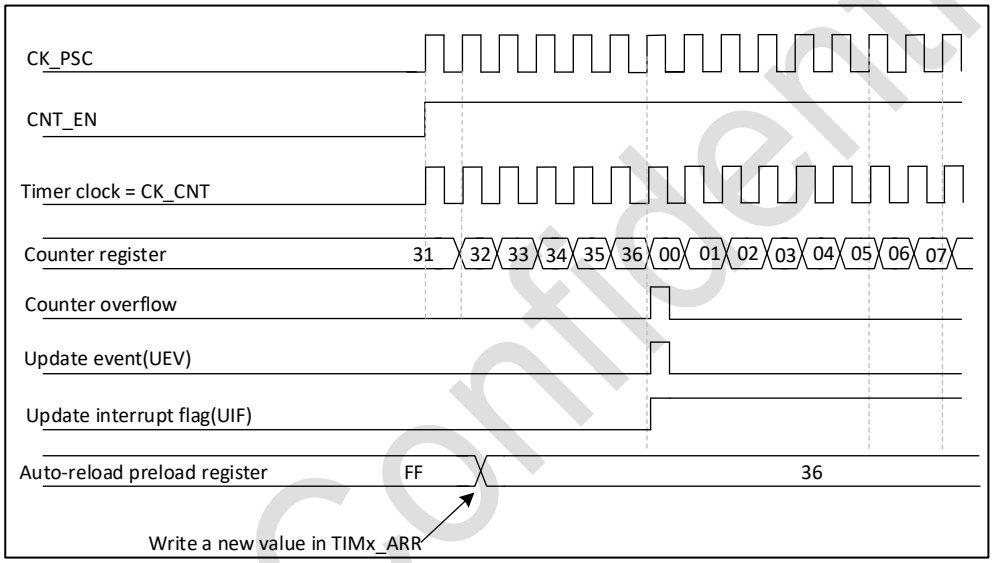


图 13-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

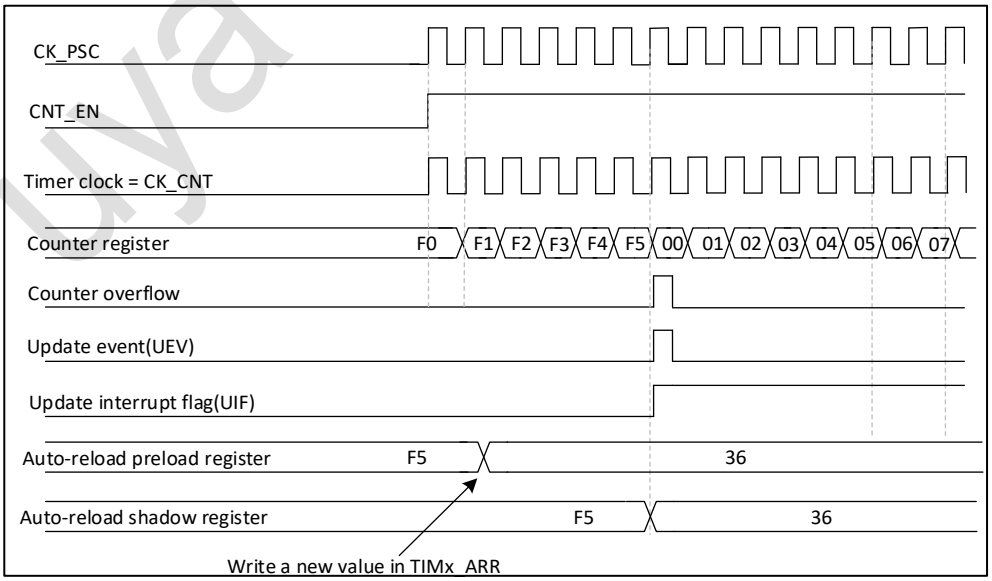


图 13-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

向下计数模式

在向下计数模式中，计数器从自动加载值(PWM_ARR 的内容)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数下溢事件。每次计数下溢都会产生更新事件。

当发生一个更新事件时，所有以下的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 占空比影子寄存器被重新置入周期寄存器的值(PWM_DTR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

以下是一些当 PWM_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

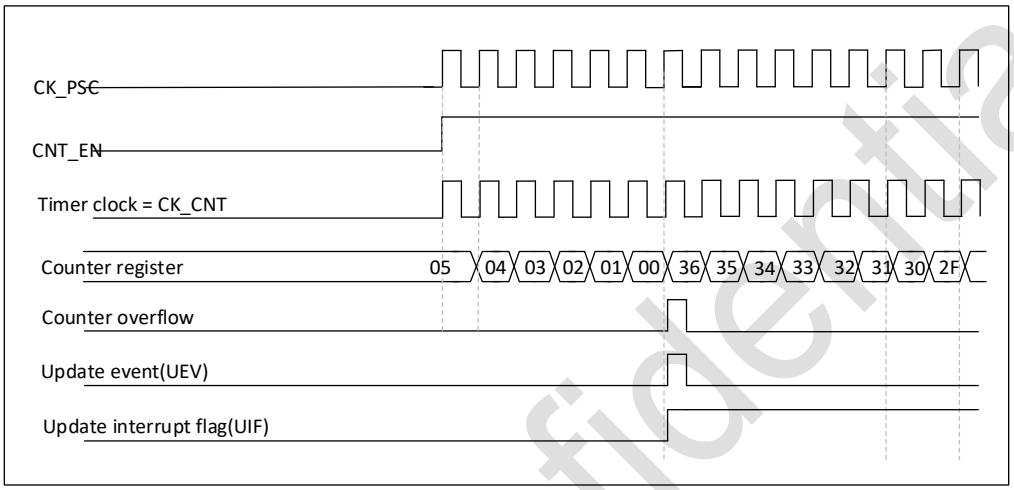


图 13-10 计数器时序图，内部时钟分频因子为 1

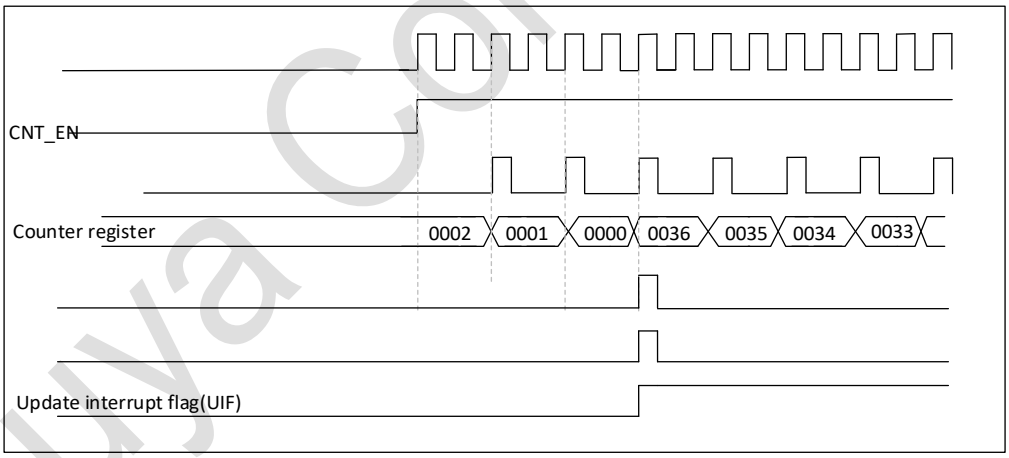


图 13-11 计数器时序图，内部时钟分频因子为 2

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(PWM_ARR 寄存器)−1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

通过配置 PWM_CR1 寄存器中的 CMS 位不为‘00’可以得到中央对齐模式。

在此模式下，不能写入 PWM_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件。然后，计数器重新从 0 开始计数，预分频器内部计数器也重新从 0 开始计数。

当发生更新事件时，所有的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 占空比影子寄存器被重新置入周期寄存器的值(PWM_DTR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

以下是一些计数器在不同时钟频率下的操作的例子：

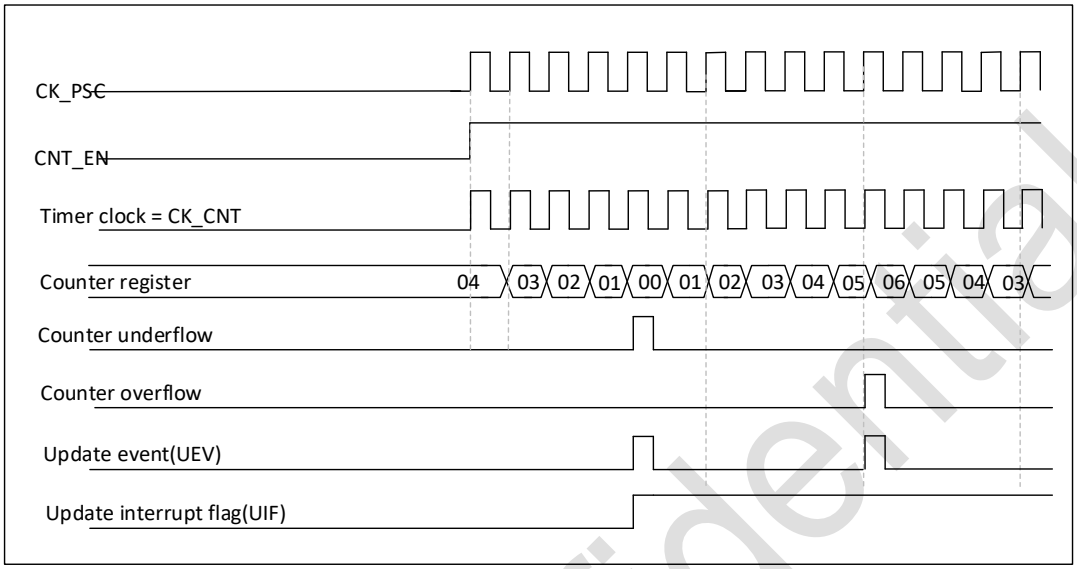


图 13-12 计数器时序图，内部时钟分频因子为 1， PWM_ARR=0x6

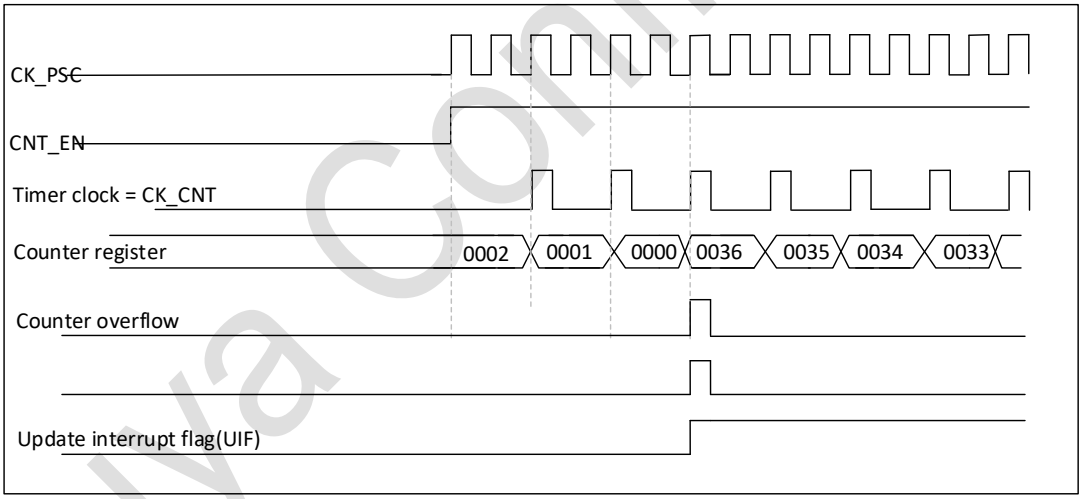


图 13-13 计数器时序图，内部时钟分频因子为 2

13.3.4. 时钟源

只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 pwm_ker_ck 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。

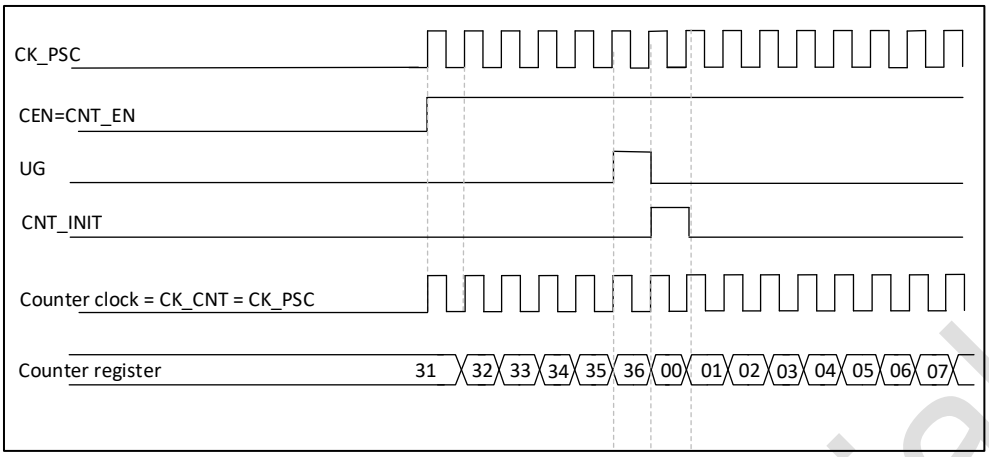


图 13-14 一般模式下的控制电路，内部时钟分频因子为 1

13.3.5. 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)。

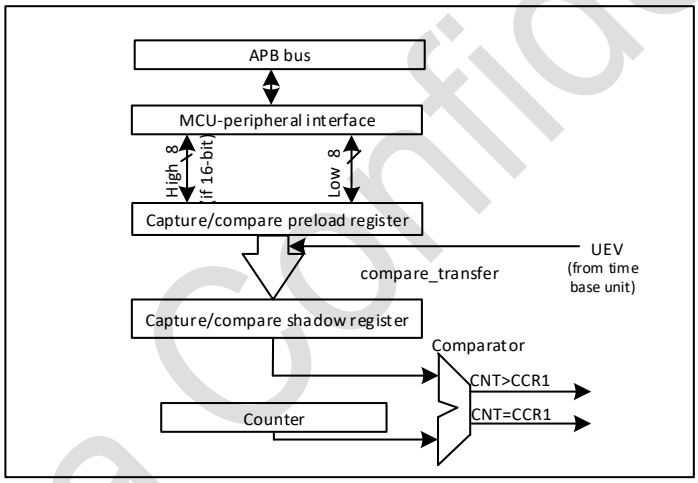


图 13-15 捕获/比较通道图

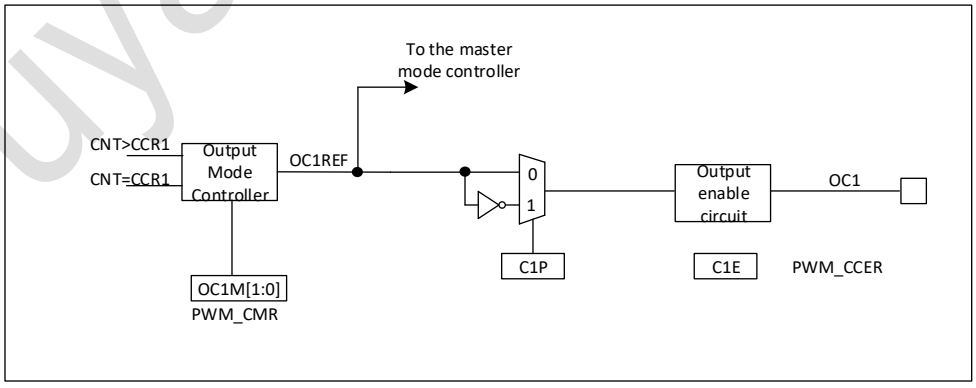


图 13-16 比较通道的输出部分(通道 1)

比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。

比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

13.3.6. PWM 模式

该模块能产生一个由 PWM_ARR 寄存器确定频率、由 PWM_CCRx 寄存器确定占空比的信号。

在 PWM_CMR 寄存器中的 OCxM 位写入“10”（PWM 模式 1）或“11”（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。

如果要改变 PWM 的周期或者占空比，修改 PWM_ARR 和 PWM_CCRx 寄存器后，在下一个周期新值才会生效。

OCx 的极性可以通过软件在 PWM_CER 寄存器中的 CxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过 PWM_CER 中 CxE 控制。

在 PWM 模式(模式 1 或模式 2)下，PWM_CNT 和 PWM_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 PWM_CCRx≤PWM_CNT 或者 PWM_CNT≤PWM_CCRx。

根据 PWR_CR 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 PWM_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 PWM_CNT<PWM_CCRx 时，PWM 为 CxP 定义的有效电平，否则为无效电平。

如果 PWM_CCRx 中的比较值大于自动重装载值(PWM_ARR)，则 PWM 输出保持为’1’。如果比较值为 0，则输出保持为’0’。

下图为 PWM_ARR=8 时边沿对齐的 PWM 波形实例。

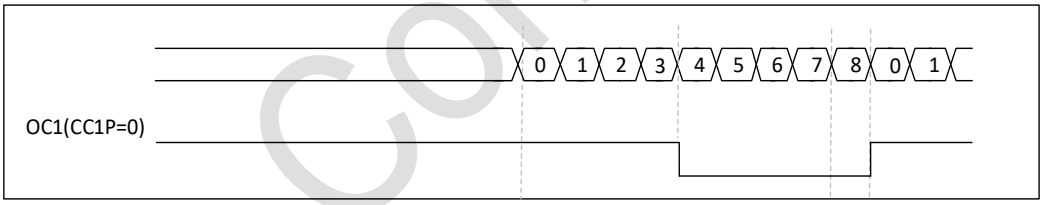


图 13-17 边沿对齐方式 PWM 输出，向上（ARR=8）

- 向下计数配置

当 PWM_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 PWM_CNT>PWM_CCRx 时输出为无效电平，否则为有效电平。如果 PWM_CCRx 中的比较值大于 PWM_ARR 中的自动重装载值，则输出保持为’1’。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 PWM_CR1 寄存器中的 CMS 位为 1 时为中央对齐模式。此时，PWM_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

下图给出一些中央对齐的 PWM 波形的例子：

- PWM_ARR = 8
- PWM 模式 1
- PWM_CR1 寄存器的 CMS=1

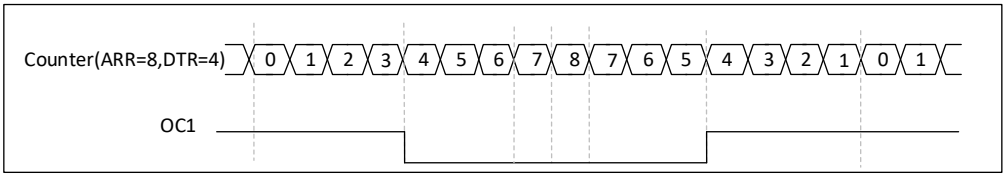


图 13-18 中央对齐方式 PWM 输出(ARR=8)

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 PWM_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中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：

1. — 如果写入计数器的值大于自动重加载的值(PWM_CNT>PWM_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
2. — 如果将0或者PWM_ARR的值写入计数器，方向被更新。

13.3.7. LED 级联控制

PWM 通过寄存器 PWM_CR1.LEDEN 来使能 LED 级联控制模式，可驱动外接 LED。

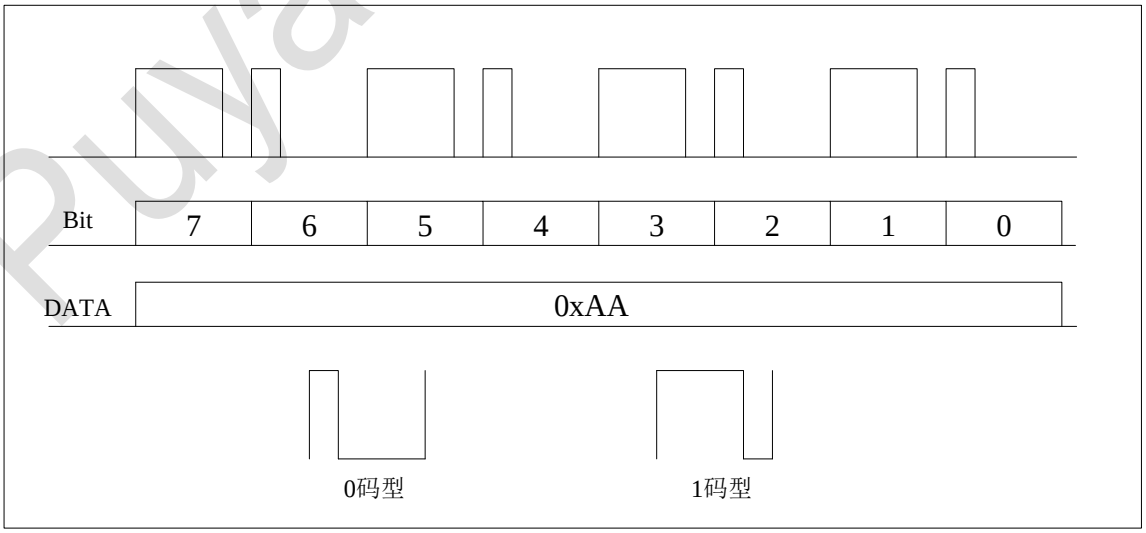
此模式下，LED 的数据通过归零码的方式传输出去，归零码的周期由 PWM_ARR 寄存器决定，归零码“0”码型的低电平时间和“1”码型的高电平时间均是由 PWM_CCRx 决定(即“0”码型的低电平时间和“1”码型的高电平时间是相同的)，需要发送的归零码数据存放在 LED_DATA 寄存器中。

通过配置 PWM_CR1.LEDWORD，可以选择一次传输数据的长度，目前有 8 位、16 位和 24 位数据长度可选。

通过配置PWM_CR1.LEDEN和PWM_CR1.LEDCEN为1后，将存放在LED_DATA寄存器中的数据，通过 PWM 端口输出。数据发送结束后，PWM_CR1.LEDCEN 自动清零，重新置位后才开始发送下一次数据。

当一笔数据发送结束后，PWM_SR. LEDx_END_IF 标志位会置起，如果使能了相对应的中断使能位，则会产生一个中断。

通过 PWM 端口输出级联控制时序如下图所示(以 8 位数据传输为例)：



13.4. PWM 寄存器描述

13.4.1. PWM 控制寄存器 1（PWM_CR1）

偏移地址：0x00

复位值：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.	LEDWORD[1:0]		LEDCEN	LEDEN
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**11**	**10**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARPE	CMS[1:0]		DIR	Res.	URS	UDIS.	CEN
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:18	LEDWORD	RW	00	LED 数据选择位 00：每次发送 8 位数据 01：每次发送 16 位数据 1x：每次发送 24 位数据
17	LEDCEN	RW	0	LED 级联数据发送控制位 0：没有数据发送，PWM 输出处于默认状态 1：使能 LED 级联数据发送功能，每次发送结束后硬件自动清零
16	LEDEN	RW	0	LED 级联功能使能位 0：关闭 LED 级联功能 1：使能 LED 级联功能
15:8	Reserved	-	-	保留，一直读为 0
7	ARPE	RW	0	自动重装载预装载允许位 0：PWM_ARR 寄存器没有缓冲 1：PWM_ARR 寄存器被装入缓冲器 注：该位在 LEDEN 位使能后硬件置 0，且写入无效。
6:5	CMS[1:0]	RW	00	选择中央对齐模式 (Center-aligned mode selection) 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。比较中断标志仅在向下计数时产生 10：中央对齐模式 2。计数器交替地向上和向下计数。比较中断标志仅在向上计数时产生 11：中央对齐模式 3。计数器交替地向上和向下计数。比较中断标志在向上和向下计数时产生

				注：1、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。 2、该位在 LEDEN 位使能后硬件置 0，且写入无效。
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：1、当计数器配置为中央对齐模式时，该位为只读。 2、该位在 LEDEN 位使能后硬件置 0，且写入无效。
3	Reserved	-	-	保留
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断请求，则下述任一事件产生更新中断请求： − 计数器溢出/下溢 − 设置 UG 位 1：如果使能了更新中断请求，则只有计数器溢出/下溢才产生更新中断请求。 注：该位在 LEDEN 位使能后硬件置 0，且写入无效。
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 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、CCRx)保持它们的值。如果设置了 UG 位，则计数器和预分频器被重新初始化。 注：该位在 LEDEN 位使能后硬件置 0，且写入无效。
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。

13.4.2. PWM 中断使能寄存器 (PWM_DIER)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LED1_END_IE
															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.	OC1IE	UIE
-	-	-												RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	LED1_ENDIE	RW	0	LED1 级联数据发送结束中断使能 0：禁止 LED 级联数据发送完成中断 1：允许 LED 级联数据发送完成中断
15:2	Reserved	-	-	保留
1	OC1IE	RW	0	OC1IE：允许比较输出 1 中断 0：禁止比较输出 1 中断 1：允许比较输出 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

13.4.3. PWM 状态寄存器（PWM_SR)

偏移地址： 0x010

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LED1_END_IF
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RC_W0
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.	OC 1IF	UIF
-	-	-	-	-	-	-	-	-	-	-	-	-	-	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	LED1_ENDIF	RC_W0	0	LED1 级联数据发送结束中断标记 0：LED 级联数据发送未结束 1：LED 级联数据发送结束
15:2	Reserved	-	-	保留
1	OC1IF	RC_W0	0	输出比较 1 中断标记 当计数器值与比较值匹配时该位由硬件置 1。 0：无匹配发生；

				1：PWM_CNT 的值与 PWM_CCR1 的值匹配。
0	UIF	RC_W0	0	更新中断标记 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：更新事件产生(计数器上溢或者下溢或者置位 UG)。该位由硬件置 1：

13.4.4. PWM 事件产生寄存器 1(PWM_EGR)

偏移地址：0x14

复位值：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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UG
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	W

Bit	Name	R/W	Reset Value	Function
15:1	Reserved	-	-	保留
0	UG	W	0	产生更新事件 该位由软件置‘1’，由硬件自动清‘0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清‘0’(但是预分频系数不变)。

13.4.5. PWM 输出比较模式寄存器 1(PWM_CMR)

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1PE	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[1:0]	
							RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 PWM_CCR1 寄存器的预装载功能，可随时写入 TIM1_CCR1 寄存器，且新值马上起作用。

				1：开启 PWM_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，PWM_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。
7:2	Reserved	-	-	保留
1:0	OC1M[1:0]	RW	2'h0	OC1PWM 模式 10：PWM 模式 1 - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为无效电平，否则为有效电平。 11：PWM 模式 2 - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为有效电平，否则为无效电平。 其它：保留 注：有效电平由 PWM_CER 寄存器配置，无效电平为取反。

13.4.6. PWM 输出比较使能寄存器 (PWM_CER)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	C1P	C1E
-	-			-	-			-	-			-	-	RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	C1P	RW	0	OC1 输出极性 0：OC1 高电平有效 1：OC1 低电平有效
0	C1E	RW	0	OC1 输出使能 0：关闭 - OC1 禁止输出，输出电平为高阻。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 C1P 的值。

13.4.7. PWM 计数寄存器(PWM_CNT)

偏移地址：0x24

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved			保留，一直为 0
9:0	CNT[9:0]	RW	10'h0	计数器的值

13.4.8. PWM 预分频器（PWM_PSC）

偏移地址：0x28

复位值：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.	PSC[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	PSC	RW	8'h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 tpwm_psc_ck/(PSC[7:0]+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 注：该位在 LEDEN 位使能后硬件置 0，且写入无效。

13.4.9. PWM 自动重载寄存器（PWM_ARR)

偏移地址：0x2C

复位值：0x0000 3FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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.	ARR[9:0]									
-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留，一直为 0
9:0	ARR[9:0]	RW	10'h3FF	输出 PWM 周期预装载值 当更新事件发生时，此预装载值才装入当前周期影子寄存器中。 当自动重装载的值为空时，计数器不工作。 注：在 LED 级联模式下，归零码的周期为 fpclk*(arr+1)。

13.4.10. PWM 比较寄存器 1(PWM_CCR1)

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	CCR1									
-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:0	CCR1[9:0]	RW	10'h0	比较 1 的值 CCR1 包含了装入当前比较 1 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 1 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC1 端口上输出 PWM 信号。 注：在 LED 级联模式下，归零码"0"码的低电平时间和"1"码的高电平时间均为(CCR1+1)个周期，归零码"0"码的高电平时间和"1"码的低电平时间为(ARR-CCR1)个周期。

13.4.11. LED1 数据寄存器（LED1_DATA）

偏移地址：0x50

复位值：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.	LED1_DATA[23:16]							
-	-	-	-	-	-	-	-	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LED1_DATA[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved			保留，一直为 0
23:0	LED1_DATA	RW	24'h0	LED1 级联发送数据寄存器 根据 LEDWORD 的值选择有效位数，即 LEDWORD = 00 时，低 8 位有效； LEDWORD = 01 时，低 16 位有效； LEDWORD = 10/11 时，低 24 位有效。

14. 低功耗定时器(LPTIM)

14.1. LPTIM 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 引入了一种灵活的时钟方案，可提供所需的功能和性能，同时将功耗降至最低。

14.2. 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 位 ARR 可重载寄存器
- 连续/单次模式

14.3. LPTIM 功能描述

14.3.1. LPTIM 框图

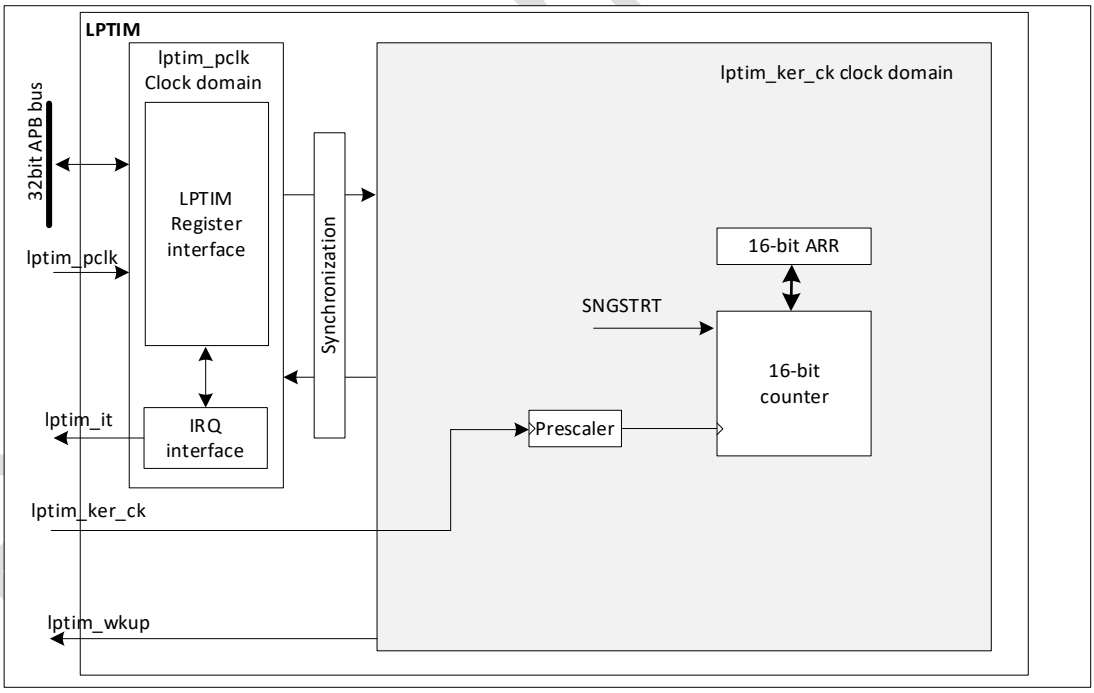


图 14-1 低功耗定时器框图

14.3.2. LPTIM 管脚和内部信号

表 14-1 LPTIM 内部信号

Names	Signal type	Description
lptim_pclk	Digital input	LPTIM APB clock domain
lptim_ker_ck	Digital input	LPTIM kernel clock
lptim_it	Digital output	LPTIM global interrupt
lptim_wakeup	Digital output	LPTIM wakeup event

14.3.3. 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）。

14.3.4. 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 14-2 预分频系数

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

14.3.5. 工作模式

LPTIM 只有一种使用 timer 模式。

- **连续模式：**计时器自由运行，从触发事件开始运行，直到计时器被禁用才停止。
- **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。

14.3.6. 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 将在当前周期结束时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用不同的时钟，因此在 APB 写入，和写入的值被应用到计数器比较器时，存在一定的延迟。在此延迟周期内，必须避免对这些寄存器进行任何额外的写操作。

14.3.7. 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。

14.3.8. 计数器复位 INDANG

为了将 LPTIM_CNT 寄存器的内容复位，提供复位机制：

异步复位机制：

异步复位由 LPTIM_CR 寄存器的 RSTARE 位控制。当该位被置为 1 时，任何读 LPTIM_CNT 寄存器的访问都将其内容复位为零。

应注意，为了可靠地读取 LPTIM_CNT 寄存器，必须进行 2 次读访问并比较其结果，结果一致，被则认为是读出值是可靠的。

需要注意的是：

- 使能异步复位时，第一次读会复位 LPTIM_CNT；第二次读才能读取 LPTIM_CNT 寄存器的计数结果。
- 在 LPTIM 计数时钟选择 PCLK/HSI 时，连续访问 2 次也不能保证读出值可靠。

14.3.9. 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。

14.4. LPTIM 低功耗模式

表 14-3 LPTIM 不同低功耗模式的区别

模式	描述
Sleep	功能没有影响。LPTIM 中断会退出睡眠模式。
Stop	LSE/LSI 时钟存在时功能没有影响。LPTIM 中断会退出停止模式。

14.5. 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。

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

14.6. LPTIM 寄存器

14.6.1. LPTIM 中断和状态寄存器 (LPTIM_ISR)

偏移地址：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	AR-ROK	Res	Res	ARRM	Res
											R			R	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	保留
4	ARROK	R	0	自动重载寄存器更新 OK。 ARROK 由硬件设置，以通知应用程序 APB 总线对 LPTIM_ARR 的写操作已成功完成。向 LPTIM_ICR.AR-ROKCF 写入 1 可清除 ARROK 标志.
3：2	Reserved	-	-	保留
1	ARRM	R	0	自动重载匹配 ARRM 由硬件设置，通知应用程序 LPTIM_CNT 寄存器值匹配 LPTIM_ARR 寄存器的值。向 LPTIM_ICR 寄存器的 ARRMCF 位写入 1 可清除 ARRM 标志
0	Reserved	-	-	保留

14.6.2. LPTIM 中断清除寄存器 (LPTIM_ICR)

偏移地址：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	AR-ROKCF	Res	Res	ARRMCF	Res
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	保留
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	-	-	保留
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	-	-	保留

14.6.3. LPTIM 中断使能寄存器（LPTIM_IER)

偏移地址：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	AR-ROKIE	Res	Res	ARRM IE	Res
											RW			RW	

Bit	Name	R/W	Reset Value	Function
31：5	Reserved	-	-	保留
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	-	-	保留
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	-	-	保留

14.6.4. LPTIM 配置寄存器（LPTIM_CFGR)

偏移地址：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	-	-	保留
22	PRELOAD	RW	0	寄存器更新模式 预加载位控制 LPTIM_ARR 寄存器更新模式 0:每次 APB 总线写访问后更新寄存器 1:寄存器在当前 LPTIM 周期结束时更新
21:12	Reserved	-	-	保留
11:9	PRESC[2:0]	RW	0	时钟预分频器 PRESC 位配置预分频器分频系数。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:/32 110:/64 111:/128
8:0	Reserved	-	-	保留

14.6.5. LPTIM 控制寄存器 (LPTIM_CR)

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RSTARE	COUN-TRST	CNTSTRT	SNGSTRT	ENABLE
											RW	RS	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	RSTARE	RW	0	读取后复位使能

				此位由软件置 1 和清 0。当 RSTARE 设置为“1”时，对 LPTIM_CNT 的任何读取访问寄存器将异步重置 LPTIM_CNT 寄存器内容。
3	COUNTRST	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

14.6.6. LPTIM 自动重装载寄存器 (LPTIM_ARR)

偏移地址：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	-	-	保留

15：0	ARR	RW	16'h0001	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

14.6.7. LPTIM 计数寄存器 (LPTIM_CNT)

偏移地址：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	-	-	保留
15：0	CNT	R	16'h0	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。

15. 独立看门狗（IWDG）

15.1. IWDG 简介

独立看门狗寄存器（简称 IWDG），该模块具有高安全级别、时序精确及灵活使用的特点。IWDG 可用来检测和解决由软件错误引起的故障，并在计数器达到指定的超时值时（TIMEOUT）触发系统复位。

独立看门狗(IWDG)由专用的低速内部时钟(LSI)驱动，即使主时钟发生故障它也仍然有效。

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。

15.2. IWDG 主要特性

- 自由运行的递减计数器
- 时钟由独立的 RC 振荡器提供(可在睡眠模式、停止模式、深度停止模式下工作)
- 看门狗被激活后，则在计数器递减至 0x000 时产生复位
- 具有低功耗冻结功能：在用户选项字节里开启 IWDG 低功耗冻结功能后，IWDG 可在停止模式和深度停止模式下保持工作状态，计数器保持递减计数
- 具有硬件看门狗功能：在用户选项字节里将 IWDG 配置成硬件启动模式后，IWDG 会在复位后自动使能
- 可自动使能 LSI：IWDG 使能后，自动使能 LSI，并作为 IWDG 的内核时钟

15.3. IWDG 功能描述

15.3.1. IWDG 框图

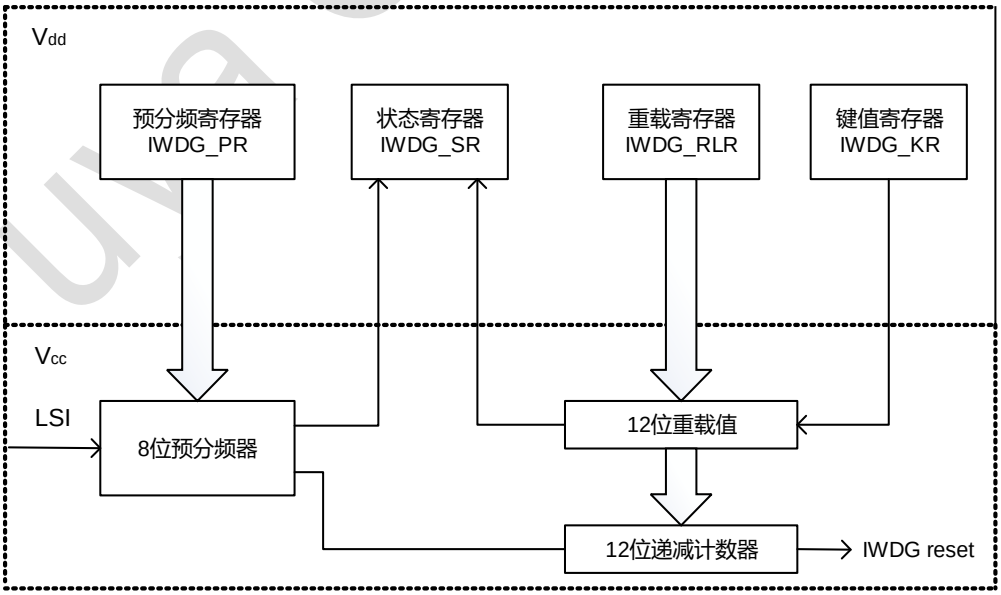


图 15-1 IWDG 框图

注： 看门狗功能处于 Vcc 供电区，即在停止模式和深度停止模式时仍能正常工作。

在键寄存器(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，只要键寄存器 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 。

一旦运行，则 IWDG 不能被停止，除非发生系统复位。

15.3.2. 硬件看门狗

如果用户在选择字节中启用了“硬件看门狗”功能，在系统上电复位后，看门狗会自动开始运行；如果在计数器计数结束前，若软件没有向键寄存器写入相应的值，则系统会产生复位

15.3.3. 硬件访问保护

对 IWDG_PR 和 IWDG_RLR 寄存器的写访问是被保护的，要修改这两个寄存器的值，IWDG_KR 必须先被写入 0x5555。对这些寄存器的写其他数将重新开启写保护，比如写入 0xAAAA，IWDG_PR 和 IWDG_RLR 将被再次保护。

IWDG_SR 指示 IWDG_PR 或 IWDG_RLR 的值是否正在更新。

15.3.4. 调试模式

如果 CPU 进入调试模式，根据调试模块中的 DBG 模块中 DBG_IWDG_STOP 配置位的状态，IWDG 的计数器继续工作或停止

15.3.5. 低功耗冻结

取决于选项字节里的 IWDG_STOP 位，决定了 IWDG 停止模式下是否继续保持计数。如果 IWDG 在停止模式和深度停止模式下保持工作，IWDG 能从当前的低功耗模式下唤醒微控制器。

15.4. IWDG 寄存器

15.4.1. IWDG 密钥寄存器 (IWDG_KR)

偏移地址：0x00

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	Reserved
15:0	KEY[15:0]	W	16’h0	键值。

				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。 写入 0x5555：允许访问 IWDG_PR 和 IWDG_RLR； 写入 0xCCCC：激活 IWDG（如果选择了硬件看门狗则不受此命令控制）。

15.4.2. IWDG 预分频寄存器（IWDG_PR)

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:0	PR[2:0]	RW	3’h0	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。 此寄存器受写保护。向 IWDG_KR 写入 0x5555 可解除写保护。 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；

15.4.3. IWDG 重装载寄存器（IWDG_RLR)

偏移地址：0x08

复位值：0x0000 0FFF

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

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	RL[11:0]	RW	12'h0	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。 此寄存器受写保护。向 IWDG_KR 写入 0x5555 可解除写保护。只有当 IWDG_SR.RVU=0 时，才能对寄存器进行修改。

15.4.4. IWDG 状态寄存器 (IWDG_SR)

偏移地址：0x0C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	RVU	R	0	看门狗计数器重装值更新。 此位由硬件置 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。 此位由硬件置 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，可继续执行后续的代码。

16. 通用异步收发器(UART)

16.1. UART 简介

UART 是一种可编程通用异步收发器。

16.2. UART 主要特征

- AMBA APB 接口
- 支持 5/6/7/8/9 位串行数据
- 支持 1/2 位 STOP 位（5 位数据时：1/1.5 位 STOP）
- 支持发送地址/数据
- 支持固定奇偶校验
- 支持断开帧
- 起始位错误检测
- 支持可编程分数波特率：可编程串行数据波特率，计算如下：波特率=（串行时钟频率）/（16*除数）
- 支持 4 位小数波特率
- 支持 Tx/Rx 引脚互换功能
- 支持大小端切换 MSB FIRST 功能

16.3. UART 功能描述

16.3.1. UART 框图

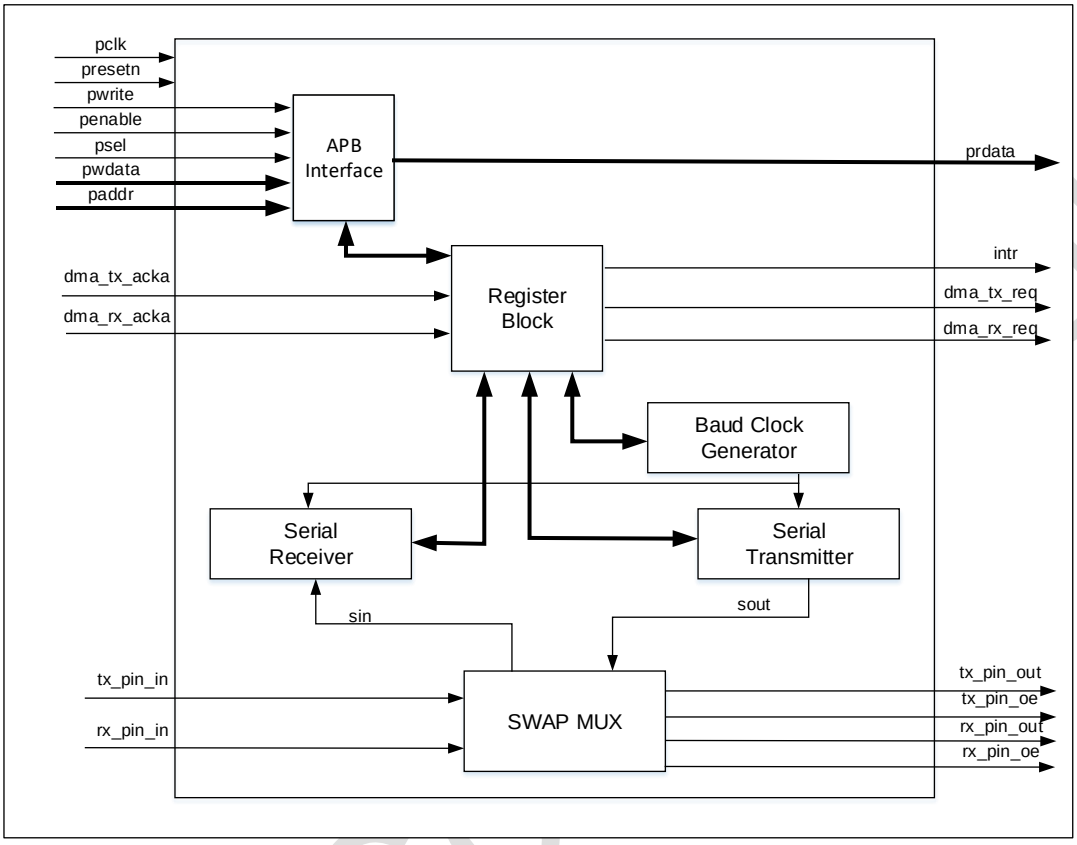


图 16-1 UART 框图

注：1.粗体线表示存在且为总线。

主要模块：

- APB 从接口（APB interface）---APB 总线接口。
- 寄存器块（Register Block）---负责 UART 的主要功能，包括控制、状态和中断生成。
- 波特时钟生成器（Baud Clock Generator）---生成发送器和接收器波特时钟以及输出参考时钟信号。
- 串行发送器（Serial Transmitter）---将写入 UART 的并行数据转换为串行形式，并添加控制寄存器指定的所有附加位，用于传输。
- 串行接收器（Serial Receiver）---将 UART 中接收的控制寄存器指定的串行数据字符转换为并行形式。此块控制：
  - 奇偶校验错误检测
  - 帧错误检测
  - 断开帧检测
- 引脚互换处理（SWAP MUX）---SWAP 使能时的引脚处理

16.3.2. UART（RS232）串行协议

因为 UART 和所选设备之间的串行通信是异步的，所以在串行数据中添加了额外的位（开始和停止）来指示开始和结束。利用这些比特可以使两个设备同步。这种由起始位和停止位组成的串行数据结构被称为一个字符，如下图所示。

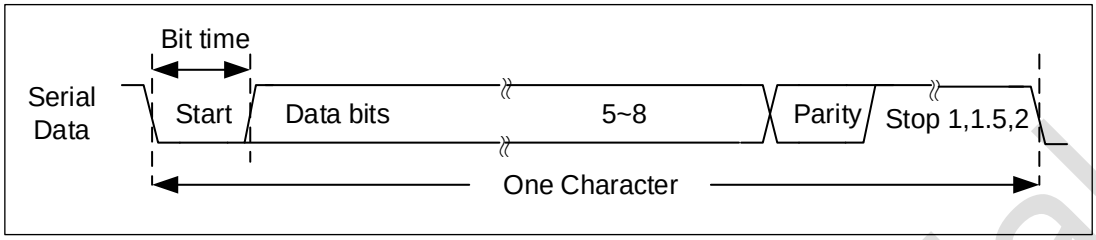


图 16-2 串行数据格式

一个额外的奇偶校验位可以添加到串行字符。该位出现在字符结构中的最后一个数据位之后和停止位之前，以便为 UART 提供对接收到的数据执行简单错误检查的能力。

UART 线路控制寄存器用于控制串行字符特性。数据字的各个位在起始位之后发送，从最低有效位（LSB）开始。它们后面是可选的奇偶校验位，后面是停止位，可以是 1、1.5 或 2。

注：UART 实现的停止位持续时间可能会更长，原因如下：

在某些配置的字符之间插入的空闲时间

传输中的所有比特都是在完全相同的持续时间内传输的；这方面的例外是当使用 1.5 个停止位时的半停止位。该持续时间被称为比特周期或比特时间；一位时间等于十六个波特时钟。

为了确保线路的稳定性，一旦检测到起始位，接收器就在位时间的大约中点对串行输入数据进行采样。因为每个比特传输的波特时钟的确切数量是已知的，所以计算采样的中点并不困难；即在起始位的中点采样之后每十六个波特时钟。

与串行输入去抖动一起，这种采样有助于避免错误起始位的检测。短故障通过去抖动被过滤掉，并且在线路上没有检测到转换。如果故障足够宽，可以通过去抖动来避免滤波，则会检测到下降沿。然而，只有当线路在半个采样周期后再次被采样为低时，才检测到起始位。

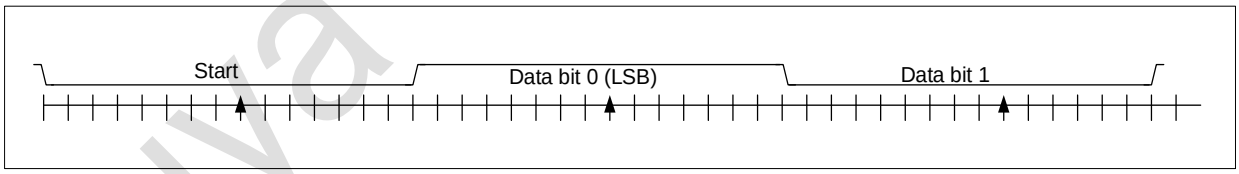


图 16-3 接收器串行数据采样点

作为 16550 标准的一部分，一个可选的波特时钟参考输出信号（baudout_n）为需要它的接收设备提供定时信息。UART 的波特率由单时钟实现中的串行时钟 pclk 以及分频锁存寄存器（BRR）控制。

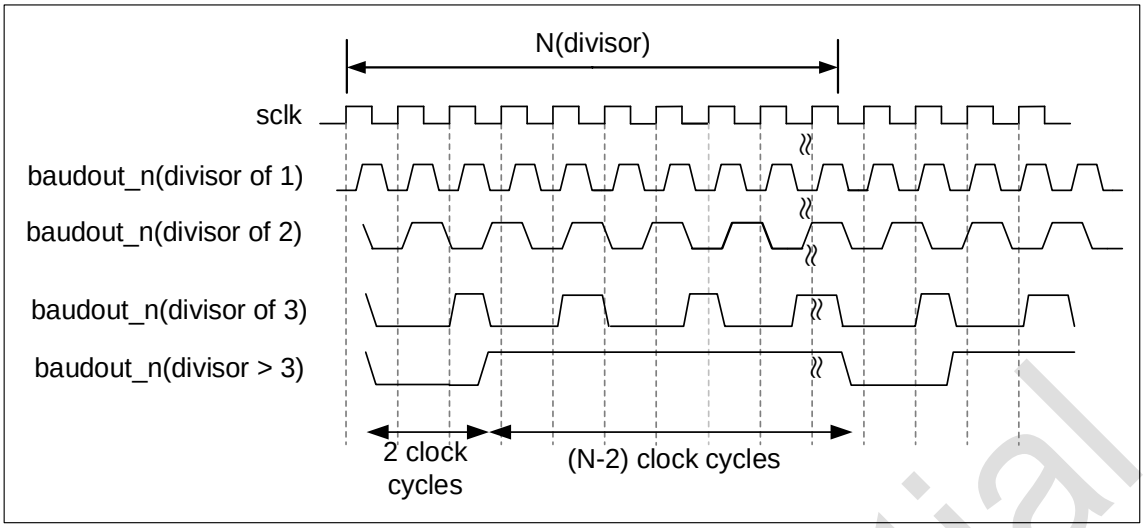


图 16-4 不同除数值的baudout_n输出的时序图

16.3.3. UART 9 位数据传输

UART 在发送和接收模式下都可以被配置为具有 9 位数据传输。字符中的第 9 位出现在字符的第 8 位之后和奇偶校验位之前。下图显示了字符的串行传输，其中 D8 表示第 9 位，还显示了 9 位模式下的一般串行传输。（此为正常小端模式，如果是大端模式则调转 9 位数据位的顺序）

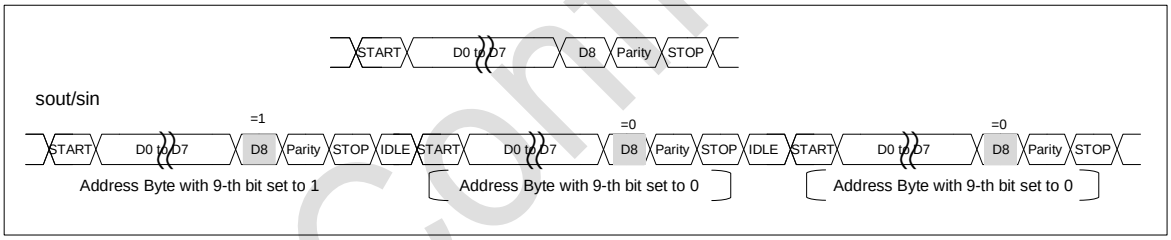


图 16-5 9 位字符

通过启用 9 位数据传输模式，UART 可以用于多点系统，其中一个主机连接到系统中的多个从机。主机与其中一个从机交流。当主机想要将数据块传输到从机时，它首先发送一个地址字节来识别目标从机。地址/数据字节之间的区分是基于输入字符中的第 9 位来完成的。如果第 9 位设置为 0，则该字符表示一个数据字节。如果第 9 位设置为 1，则该字符表示地址字节。所有从系统将地址字节与它们自己的地址进行比较，并且只有目标从机（其中地址匹配）能够从主机接收数据。主机开始向目标从机发送数据字节。未寻址的从机忽略传入的数据，直到接收到新的地址字节。

在上图中，请注意一个地址后面跟着 2 个数据字节。地址字节在第 9 位（D8）设置为 1 的情况下输出，而数据字节在第九位（D8）设置为 0 的情况下发出。奇偶校验位是一个可选字段。

用于 9 位数据传输的 UART 的配置执行以下操作：

- UART_CR3.M_E 位用于启用或禁用 9 位数据传输。
- 在接收的情况下，UART_CR3.ADDR_MATCH 用于在基于硬件和软件的地址匹配之间进行选择。
- UART_CR3.SEND_ADDR 位用于在发送的情况下使能发送地址。
- UART_CR3.TX_MODE 位用于在基于硬件和软件的地址传输之间进行选择。
- UART_TAR 和 UART_RAR 寄存器分别用于发送地址和匹配接收到的地址。
- UART_DR（TDR/RDR）寄存器为 9 位寄存器，用于在 9 位模式下进行数据传输。
- UART_SR.ADDR_RCVD 位用于指示地址接收中断。

16.3.3.1. 发送模式

UART 支持两种类型的传输模式：

- 传输模式 0（当 UART_CR3.TX_MODE 设置为 0 时）
- 传输模式 1（当 UART_CR3.TX_MODE 设置为 1 时）

1. 模式 0：

在传输模式 0 中，地址被编程在传输地址寄存器（TAR）中，数据被写入传输保持寄存器（TDR）。TDR 寄存器的第 9 位在此模式中不适用。

下图说明了基于 SEND_ADDR（CR3[2]）、TDR 空条件的地址和数据传输。

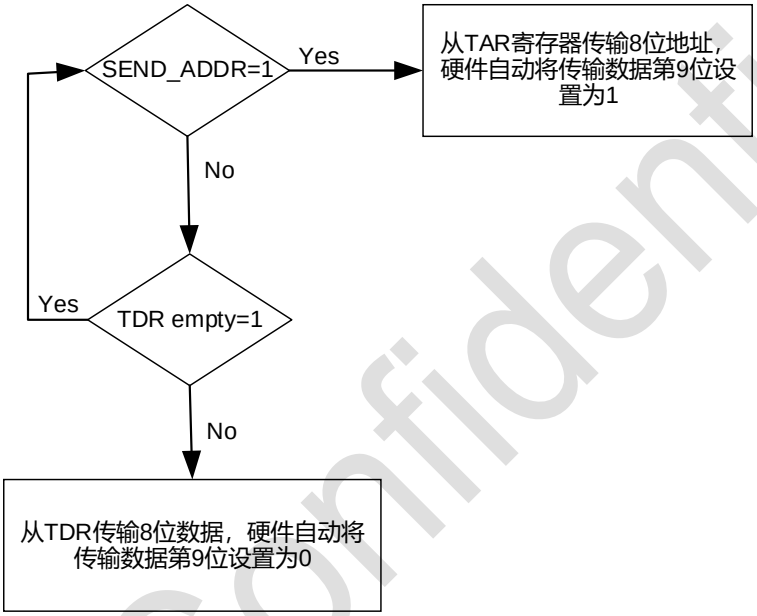


图 16-6 自动地址传输流程图

要向其传输数据的目标从机地址被编程在 TAR 寄存器中。必须启用 SEND_ADDR（CR3[2]）位来传输串行 UART 线上 TAR 寄存器中的目标从机地址，其中第 9 个数据位设置为 1，表示正在发送地址到从机。地址字符开始在线上传输后，UART 清除 SEND_ADDR 位。

传输到目标从机所需的数据通过传输保持寄存器（TDR）进行编程。数据在 UART 线上传输，第 9 个数据位设置为 0，表示正在发送数据到从机。

2. 模式 1：

在传输模式 1 中，TDR 寄存器为 9 位宽，地址和数据都通过 TDR 寄存器编程。UART 不区分地址和数据。SEND_ADDR（UART_CR3[2]）位和传输地址寄存器（UART_TAR）不适用于此模式。根据是否必须发送地址/数据，软件必须用 1/0 写入第 9 位。

表 16-1 发送配置

M_E	TX_MODE	SEND_ADDR	使用场景
0	X	X	发 8 位 TDR 发数据
1	0	0	发“0+8 位 TDR 数据“
1	0	1	发“1+8 位 TAR 地址”
1	1	X	发 “9 位 TDR 数据”

注：该表说明作为发送方可以使用的几种发送场景和对应的配置。

16.3.3.2. 接收模式

UART 支持两种接收模式：

- 硬件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 1 时）
- 软件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 0 时）

1. 硬件地址匹配接收模式：

在硬件地址匹配接收模式中，如果接收字符的第 9 位设置为 1，则 UART 将接收字符与在接收地址寄存器（UART_RAR）中编程的地址进行匹配：

如果接收到的地址与 UART_RAR 寄存器中的编程地址匹配成功，则随后接收数据字节。

如果地址匹配失败，则 UART 控制器丢弃数据字符，直到接收到匹配的地址为止。

下图显示了基于地址匹配功能的数据字节接收流程图。

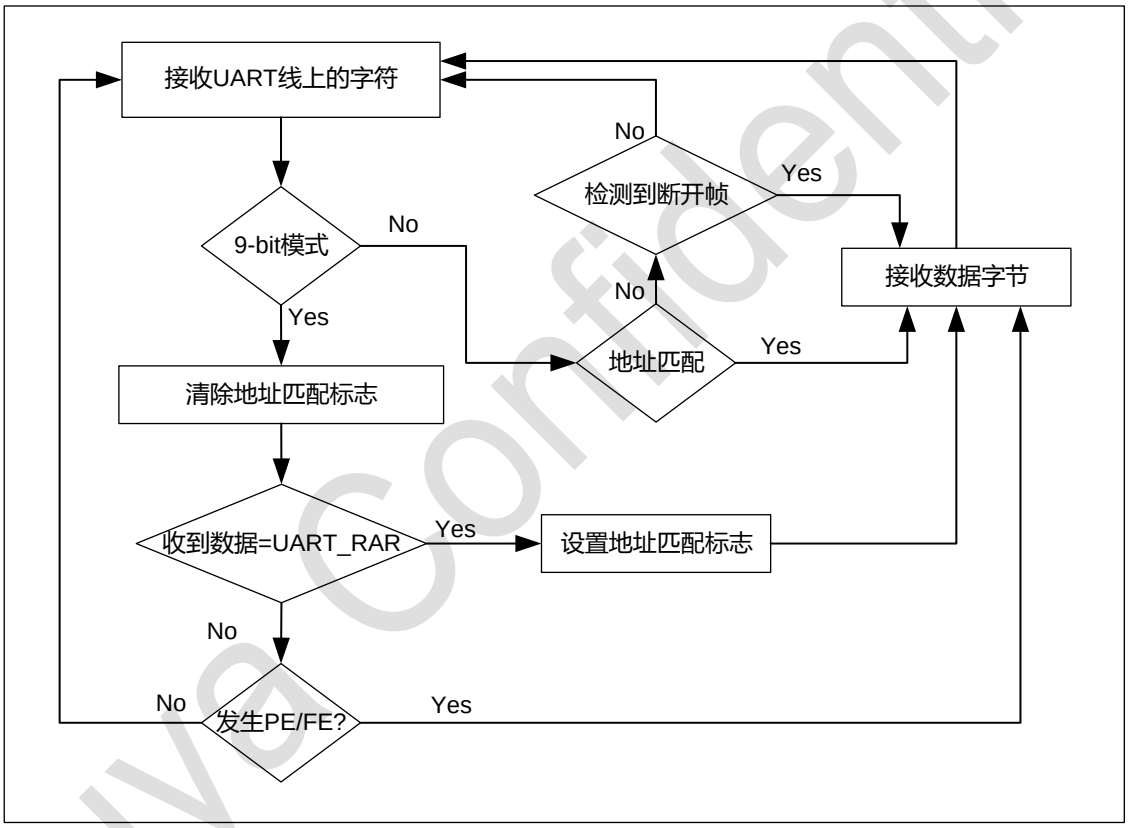


图 16-7 硬件地址匹配接收模式

UART 接收字符，而不管第 9 位数据是否设置为 1。如果接收到的字符的第 9 位被设置为 1，则它清除内部地址匹配标志，然后将接收到的 8 位字符信息与 UART_RAR 寄存器中编程的地址进行比较。

如果接收到的地址字符与 UART_RAR 寄存器中编程的地址匹配成功，则地址匹配标志被设置为 1，并且接收到的字符推送到 UART_RDR 寄存器，并且 UART_SR 寄存器中的 ADDR_RCVD 位被设置为指示地址已被接收，随后的数据字节（接收字符的第 9 位设置为 0）被推送到 UART_RDR。

如果地址与 UART_RAR 寄存器匹配失败并且（奇偶校验或者接收的地址字符中发现帧错误）的情况下，则接收的地址字符串仍然被推送到 UART_RDR 寄存器，ADDR_RCVD 和 PE/FE 错误位被设置为 1。

如果接收到任何中断字符，UART 将其视为一个特殊字符，并将其推送到 UART_RDR 寄存器，而不考虑地址匹配标志。

2. 软件地址匹配接收模式：

在这种操作模式中，UART 不执行与 UART_RAR 寄存器的接收地址字符（第 9 位数据设置为 1）的地址匹配。UART 始终接收 9 位数据，移位进 UART_RDR 寄存器。每当接收到地址字节并通过状态寄存器中的 ADDR_RCVD 位指示时，必须由用户自行比较地址。

16.3.3.3. UART 波特率

UART 的波特率由 PCLK 和分频锁存寄存器（UART_BRR）和小数波特率寄存器 UART_BRRF 控制。

波特率由以下因素决定：

- 串行时钟工作频率（PCLK）
- 波特率生成器除数值 divisor（由 UART_BRR 寄存器组成）
- 可接受的波特率误差

计算波特率的方程式如下：

波特率=串行时钟工作频率/（16* DIVISOR）——（1）

其中，DIVISOR–用于对 BRR 进行编程的数字（十六进制）。

串行时钟频率–UART PCLK 引脚处的频率。

根据等式（1），DIVISOR 可以计算为：DIVISOR=串行时钟频率/（16*波特率）

（等式算出的小数部分放在 UART_BRRF）

同样从等式（1）中，还可以示出：串行时钟频率= 波特率* 16* 除数

波特率和选定波特率定之间的误差如下所示：

百分比错误=（ |波特率-选定波特率| ）/波特率* 100%

表 16-2 设置波特率时的误差计算

波特率		fpclk = 4 MHz			fpclk = 24 MHz			fpclk = 48 MHz		
序号	kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.404	104	0.16%	2.4	625	0.00%	2.4	1250	0.00%
2	9.6	9.615	26	0.16%	9.615	156	0.16%	9.615	312	0.16%
3	19.2	19.231	13	0.16%	19.231	78	0.16%	19.231	156	0.16%
4	57.6	62.5	4	8.51%	57.692	26	0.16%	57.692	52	0.16%
5	115.2	125	2	8.51%	115.385	13	0.16%	115.385	26	0.16%
6	230.4	250	1	8.51%	250	6	8.51%	230.769	13	0.16%
7	460.8	不可能	不可能	不可能	500	3	8.51%	500	6	8.51%
8	921.6	不可能	不可能	不可能	1500	1	62.76%	1000	3	8.51%
9	2250	不可能	不可能	不可能	不可能	不可能	不可能	3000	1	33.33%
10	4500	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能

注:

1. CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

2.在配置时钟的时候，由于波特率时钟是在系统时钟上进行分频的，但得到的时钟频率与实际时钟频率会有误差。配置的时钟误差需要在 2%以内，才可以正常工作。

16.3.4. UART 接收容忍度

只有当整体的时钟系统地变化小于 UART 异步接收器能够容忍的范围， UART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的变化(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足： DTRA + DQUANT + DREC + DTCL < UART 接收器的容忍度

对于正常接收数据，UART 接收器的容忍度等于最大能容忍的变化为 96%~105%。

16.4. UART 中断

UART 中断输出信号（intr）的断言——只要几个优先中断类型中的一个被启用并激活，就会发生中断。以下中断类型可以通过 UART_CR2 寄存器启用：

- 接收器数据可用（RXNEIE）
- 发送寄存器空（TXEIE）
- 发送保持寄存器为空（在可编程 TDR 中断模式下）（TDREIE）
- 忙错误检测指示（BUSYERRIE）
- 接收线路状态（LSIE）

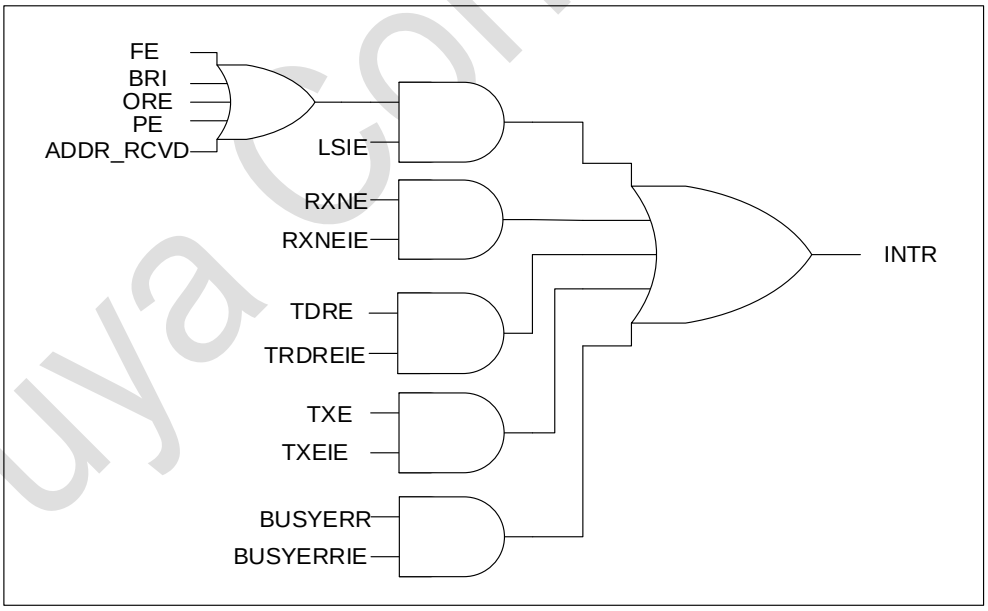


图 16-8 中断映射图

这些中断类型在下表中有详细说明。

表 16-3 中断控制功能

中断置位和清零功能		
中断类型	**中断源**	**中断清零控制**
接收线路状态	溢出/奇偶校验/帧错误中断或地址接收中断	对应位写 1 清 0
接收数据可用 RXNE	接收器数据可用	读取接收器缓冲寄存器
发送保持寄存器为空 TDRE	发送保持寄存器为空	写入 TDR
发送寄存器为空 TXE	发送寄存器为空	写入 TDR
忙错误检测 BUSYERR	UART 繁忙时（BUSY[0]设置为 1），主机尝试写入 CR1 寄存器。	对应位写 1 清 0

16.5. UART 寄存器

术语：设置=设置为 1；清除=清除为 0。

16.5.1. UART 数据寄存器 (UART_DR)

地址偏移：0x00

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	DR[8:0]	RW	9'b0	接收/发送数据寄存器。 一共是由两个寄存器组成（一个是发送的 TDR，一个是接收的 RDR），所以 DR 寄存器实现了读和写的两个功能。 RDR 寄存器是 UART 模式下串行输入端口上接收到的数据字节。只有状态寄存器（SR）中的接收非空（RXNE）位被设置时，该寄存器中的数据才有效。 必须在下一个数据到达之前读取 RDR 中的数据，否则它将被覆盖，从而导致溢出错误。 TDR 寄存器是在 UART 模式下串行输出端口上传输

				的数据。软件保证只有当 TDR 空（UART_SR.TDRE）位被设置时，数据才能再次写入 TDR。 如果 TDRE 被设置，则向 TDR 写入单个字符将清除 TDRE。在再次设置 TDRE 之前对 TDR 的任何额外写入都会导致 TDR 数据被重写。 仅当 UART_CR3.TX_MODE = 1 时，第 9 位才适用。

16.5.2. UART 波特率寄存器 (UART_BRR)

地址偏移：0x04

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BRR	RW	16'b0	该寄存器构成 16 位 divisor，divisor 包含 UART 的波特率除数。输出波特率等于串行时钟（PCLK）频率除以波特率除数值的十六倍，如下所示：波特率=（串行时钟频率）/（16*divisor）。 请注意，当除数锁存寄存器（BRR）设置为零时，波特时钟被禁用，不会发生串行通信。 此外，一旦设置了 BRR，在发送或接收数据之前，应等待至少 8 个时钟周期。

16.5.3. UART 状态寄存器 (UART_SR)

地址偏移：0x10

复位值：0x0000_0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	BUSY_ERR	BUSY	ADDR_RCVD	Res.	TXE	TDRE	BRI	FE	PE	ORE	RXNE

					RC_W1	R	RC_W1		R	O	RC_W1	RC_W1	RC_W1	RC_W1	R

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	BUSY_ERR	RC_W1	1'b0	忙检测错误，检测到忙时的误操作。 0：无忙误操作错误。 1：UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 该位写 1 清零。
9	BUSY	R	1'b0	UART 忙 这表示串行传输正在进行中，清除时表示 UART 处于空闲或非活动状态。 在以下任何一种情况下，此位都将设置为 1（忙）： ---串行接口上正在进行发送 ---串行接口上正在进行接收 ---传输 TDR 中存在的数据，波特除数为非零（BRR 不等于 0） ---接收 RDR 中存在数据 注意：即使可能从另一个设备发送了新字符，UART 忙位也可能被清除。也就是说，如果 UART 在 TDR 和 RDR 中没有数据，并且没有正在进行的传输，并且新字符的起始位刚刚到达 UART。这是由于直到位周期的中间才看到有效的开始，并且该持续时间取决于已编程的波特除数。 0（IDLE）：UART 处于空闲或非活动状态 1（忙）：UART 忙（主动传输数据）
8	ADDR_RCVD	RC_W1	1'b0	地址接收。 如果启用 9 位数据模式（UART_CR3.M_E = 1），则该位用于指示接收数据的第 9 位被设置为 1。该位还可以用于指示输入字符是地址还是数据。 0：表示字符为数据。 1：表示字符为地址。 注意：用户需要确保在下一个地址字节到达之前清除中断（该位写 1 清 0）。 如果清除中断有延迟，则软件将无法区分多个地址相关的中断。
7	Reserved	-	-	保留
6	TXE	R	1'b1	发送为空。 每当当前没有发送数据且 TDR 为空的时候，就会设置

				此位。如果 TXEIE 使能，则会产生 TXE 中断 0：发送不为空 1：发送为空
5	TDRE	R	1'b1	发送保持寄存器空。 该位指示 TDR 为空。 每当数据从 TDR 传输到发送器移位寄存器且没有新数据写入 TDR 时，该位被设置。如果 TDRIE 使能，则会产生中断。 0：TDR 非空 1：TDR 为空
4	BRI	RC_W1	1'b0	断开中断位 这用于指示在串行输入数据上检测到中断序列。 如果在 UART 模式下，每当串行输入保持在逻辑“0”状态的时间超过起始时间+数据位+奇偶校验位+停止位的总和时，就会设置它。对该位写 1 将清除 BRI 位。 0：未检测到断开序列 1：检测到断开序列
3	FE	RC_W1	1'b0	帧错误位。 这用于指示接收机中出现帧错误。当接收机在接收的数据中没有检测到有效的停止位时，就会发生帧错误。 应该注意的是，如果发生断开，也将设置帧错误（UART_SR.FE）位，如断开中断（UART_SR.BRI）位所示。之所以会发生这种情况，是因为断开字符通过将输入保持到逻辑 0 的时间长于字符的持续时间而隐式地生成帧错误。对该位写 1 将清除 FE 位。 0：无帧错误 1：帧错误
2	PE	RC_W1	1'b0	奇偶校验错误位。 如果设置了奇偶校验使能（UART_CR1.PEN）位，则该寄存器用于指示接收器发生奇偶校验错误。 应该注意的是，如果发生了断开，在这种情况下，如果奇偶校验生成和检测被启用（UART_CR1.PCE = 1）并且奇偶校验被设置为奇数（UART_CR1.PS = 0），则 PE 也会被设置。对该位写 1 将清除 PE 位。 0：无奇偶校验错误 1：奇偶校验错误
1	ORE	RC_W1	1'b0	溢出错误位。 这用于指示溢出错误的发生。 如果在读取以前的数据之前接收到新的数据字符，则

				会发生这种情况。 当新字符在从 RDR 读取前一个字符之前到达接收器时，设置 ORE 位。当这种情况发生时，RDR 中的数据将被覆盖。 对该位写 1 将清除 ORE 位。 0：无溢出错误 1：溢出错误
0	RXNE	R	1'b0	数据就绪位。 这用于指示接收器在 RDR 中至少包含一个字符。 读取 RDR 时，此位被清除。 0：数据未就绪 1：数据就绪

16.5.4. UART 控制寄存器 1 (UART_CR1)

地址偏移：0x14

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	MSBFIRST	RW	1'b0	最高有效位在前。 0：起始位后，收发第 0 位数据； 1：起始位后，收发第 5/6/7/8/9 位数据； 在数据传输过程中，不能修改这个位。
8	SWAP	RW	1'b0	TX/RX 引脚互换。 0：TX/RX 引脚按照标准引脚映射定义； 1：TX/RX 引脚互换。此时用作交叉连接其他 UART 时。
7	Reserved	-	-	保留
6	SBK	RW	1'b0	断开控制位。 这用于将断开发送到接收设备。如果设置为 1，串行输出将强制进入间隔（逻辑 0）状态。输出线被强制为低电平，直到 SBK 位被清除。 0：释放串行输出以进行数据传输

				1：串行输出被迫进入间隔状态
5	SP	RW	1'b0	固定奇偶校验。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。此位用于强制固定奇偶校验值。当 PCE、PS 和 SP 设置为 1 时，奇偶校验位被发送并检查为逻辑 0。如果 PCE 和 SP 被设置为 1，并且 PS 是逻辑 0，则奇偶校验位被发送并被检查为逻辑 1。如果此位设置为 0，则 SP（固定奇偶校验）被禁用。 0：固定奇偶校验已禁用 1：固定奇偶校验已启用
4	PS	RW	1'b0	偶数奇偶校验选择。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当奇偶校验被启用（PCE 设置为 1）时，这用于在偶数和奇数奇偶校验之间进行选择。如果设置为 1，则传输或检查偶数个逻辑“1”。如果设置为零，则传输或检查奇数个逻辑“1”。 0：奇校验 1：偶校验
3	PCE	RW	1'b0	奇偶校验启用。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。该位用于控制发送和接收的串行字符中的奇偶校验生成和检测。 0：禁用奇偶校验 1：启用奇偶校验
2	STOP	RW	1'b0	停止位的数量。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。这用于选择外围设备将发送和接收的每个字符的停止位数。如果设置为零，则在串行数据中传输一个停止位。 如果设置为 1 并且数据位设置为 5（UART_CR1.M 设置为 0），则发送一个半停止位。否则，传输两个停止位。 注意，无论所选择的停止位的数量如何，接收器将仅检查第一个停止位。 注：由于某些配置的字符之间插入的空闲时间和传输方向上的波特时钟除数值，UART 实现的停止位持续时间可能会更长； 0：1 个停止位 1：1.5/2 个停止位
1:0	M[1:0]	RW	2'b0	数据长度选择。

				仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当 UART_CR3 中的 M_E 设置为 0 时，此寄存器用于选择外围设备将发送和接收的每个字符的数据位数。 0x0：每个字符 5 个数据位 0x1：每个字符 6 个数据位 0x2：每个字符 7 个数据位 0x3：每个字符 8 个数据位

16.5.5. UART 控制寄存器 2 (UART_CR2)

地址偏移：0x18

复位值：0x0000_0008

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

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	TXEIE	RW	1'b0	启用发送空中断。这用于启用/禁用传输发送空中断的生成。 0：禁用发送空中断 1：使能发送空中断
3	BUSYERRIE	RW	1’b1	启用 BUSYERR 状态中断。这用于启用/禁用 BUSYERR 状态中断生成。UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 0：禁用 BUSYERR 状态中断 1：使能 BUSYERR 状态中断
2	LSIE	RW	1'b0	启用接收器线路状态中断。这用于启用/禁用接收器线路状态中断的生成。 该中断标志是 PE、FE、ORE、BRI、ADDR_RCVD 中断标志的组合。 0：禁用接收器线路状态中断 1：使能接收器线路状态中断
1	TDREIE	RW	1'b0	启用传输保持寄存器空中断。这用于启用/禁用传输保持寄存器空中断的生成。 0：禁用传输保持寄存器空中断

				1：使能传输保持寄存器空中断
0	RXNEIE	RW	1'b0	启用接收数据可用中断。这用于启用/禁用接收数据可用中断。 0：禁用接收数据中断 1：使能接收数据中断

16.5.6. UART 控制寄存器 3 (UART_CR3)

地址偏移：0x1C

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	TX_MODE	RW	1'b0	传输模式控制位。该比特用于控制 9-bit 数据传输期间的传输模式的类型。 数值： 1：在这种操作模式下，传输保持寄存器（TDR）为 9 位宽。 用户需要确保 TDR 寄存器的地址/数据写入正确。 地址：第 9 位设置为 1 数据：第 9 位数设置为 0 注意：传输地址寄存器（UART_TAR）不适用于此操作模式。 0：在此操作模式中，传输保持寄存器（TDR）的宽度为 8 位。 用户需要将地址编程到传输地址寄存器（UART_TAR）中，并将数据编程到 TDR 寄存器中。
2	SEND_ADDR	RW	1'b0	发送地址控制位。此位用作控制旋钮，供用户在传输模式下确定何时发送地址。 0：9 位字符内容来自 TDR 寄存器 1：9 位字符内容：第 9 位设置为 1，其余 8 位将与“传输地址寄存器”中正在编程的内容相匹配。 注：

				1.在发出地址字符后，该位由硬件自动清除。用户不应将此位编程为 0。 2.此字段仅在 M_E 位设置为 1 且 TX_MODE 设置为 0 时适用。
1	ADDR_MATCH	RW	1'b0	地址匹配模式。此位用于在接收期间启用地址匹配功能。 ■ 在地址匹配模式下，UART 将等待第 9 位设置为 1 的传入字符。并进一步检查地址是否与“接收地址匹配寄存器”中编程的地址匹配。如果匹配，则后续字符将被视为有效数据，UART 开始接收数据。 ■ 在正常模式下，UART 将开始接收数据，9 位字符将形成。用户负责读取数据并区分 b/n 地址和数据。 0：正常模式 1：地址匹配模式 注：仅当 M_E 设置为 1 时，此字段才适用。
0	M_E	RW	1'b0	9 数据使能 此位用于启用用于发送和接收传输的 9 位数据 0：禁止 9 位数据 1：使能 9 位数据

16.5.7. UART 接收地址寄存器 (UART_RAR)

地址偏移：0x20

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	RAR[7:0]	RW	8'b0	接收模式下的地址匹配寄存器。 如果在输入字符中第 9 位被设置为 1，则将对照该寄存器值检查剩余的 8 位。如果匹配成功，则第 9 位设置为 0 的后续字符将被视为数据字节，直到接收到下一个地址字节。 注： 1. 仅当“ADDR_MATCH”（UAR_CR3[1]）和“M_E”

				（UART_CR3[0]）位设置为 1 时，此寄存器才适用。2. RAR 可以在任何时间点被编程。但是，当任何接收正在进行时，用户不得更改此寄存器值。

16.5.8. UART 发送地址寄存器 (UART_TAR)

地址偏移：0x24

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	TAR[7:0]	RW	8'b0	这是传输模式下的地址匹配寄存器。 如果 M_E 位被启用，并且“SEND_ADDR”（UART_CR3[2]）位被设置为 1，则 UART 将发送第 9 位设置为 1 的 9 位字符，剩余的 8 位地址将从该寄存器发送。 注： 1. 此寄存器仅用于发送地址。正常数据应通过编程 TDR 寄存器发送。 2. 在 UART 串行通道上开始发送地址后，硬件将自动清除“SEND_ADDR”位

16.5.9. UART 波特率小数寄存器 (UART_BRRF)

地址偏移：0x28

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留

3:0	BRRF[3:0]	RW	4'b0	波特率小数部分。 小数值由（BRRF）/（2^4）确定。

17. 电压基准缓冲器(V_{REFBUF})

17.1. V_{REFBUF} 简介

内嵌的 V_{REFBUF} 被用作 ADC 参考电压。

17.2. V_{REFBUF} 功能描述

内嵌的参考电压支持三档电压，可通过 VREFBUF_CR 寄存器中的 VREFBUF_OUT_SEL 配置:

- VREFBUF_OUT_SEL=01：1.5 V
- VREFBUF_OUT_SEL=10：2.048 V
- VREFBUF_OUT_SEL=11：2.5 V

通过设置 VREFBUF_CR 中的 VREFBUF_EN 来使能 V_{REFBUF}。

17.3. V_{REFBUF} 寄存器

17.3.1. V_{REFBUF} 控制寄存器 (VREFBUF_CR)

偏移地址：0x00

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3	VREFBUF_EN	RW	0	V_{REFBUF} 使能。 0：禁止 V_{REFBUF} 1：使能 V_{REFBUF}
2	Reserved	-	-	保留
1:0	VREFBUF_OUT_SEL	RW	0	V_{REFBUF} 模块输出电压选择（不同产品该寄存器代表电压值会不同）。 00：保留 01：1.5 V 10：2.048 V 11：2.5 V

18. MCU 调试接口

18.1. DBGMCU 简介

本芯片基于 Cortex-M0+ CPU，该 CPU 核包含高级 debug 硬件扩展功能。硬件调试模块允许内核在给定指令（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 SW-DP。在 M0+ CPU 核中的调试功能是由一套 ARM CoreSight Design kit 提供。

M0+提供了集成的片上调试支持，由以下部分组成：

- SW-DP：串行调试端口（Serial wire）
- BPU：断点单元（Break point unit）
- DWT：数据触发（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，SWDIO@PB6、SWCLK@PB7
- MCU 调试盒（支持低功耗模式，控制外设时钟等）

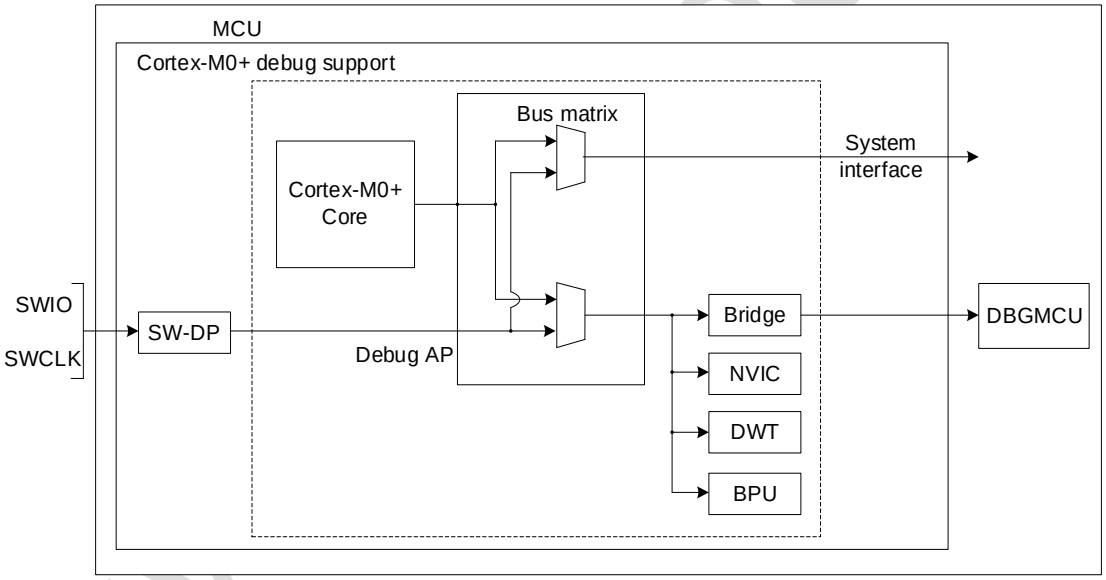


图 18-1 DBG 框图

18.2. 引脚分布和调试端口脚

18.2.1. 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 18-1 DBG 框图

SW-DP 端口引脚名称	SW 调试接口		引脚分配
	类型	调试功能	
SWDIO	输入/输出	串行数据输入/输出	PB6
SWDCLK	输入	串行时钟	PB7

18.2.2. 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用来调试主机使用的端口。

另外，芯片可以关闭 SWD 端口，并释放该端口作为 GPIO 使用。

18.2.3. 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：输入上拉
- SWCLK：输入下拉

18.3. 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 0138 地址，装载到 DBGMCU_IDCODE 寄存器中。

18.4. SWD 调试端口

18.4.1. 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 100 kΩ 的电阻）。

在协议中每次 SWDIO 方向的改变，在线上既没有被主机，也没有被芯片驱动的转向时间。默认状态下，这个时间是 1 个比特的时间，但是可以通过配置 SWCLK 频率来调整。

18.4.2. SWD 协议序列

每个序列由以下阶段组成：

- 主机发送的包请求（8bits）
- 芯片发送的应答响应（3bits）
- 主机或者芯片的数据发送阶段（33bits）

表 18-2 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

比特位	名称	描述
		1：读请求
4:3	A[3:2]	DP 或者 AP 寄存器的地址区域
5	Parity	前面比特位的校验位
6	Stop	0
7	Park	不能由主机驱动，由于有上拉，目标永远读为 1

通常转向时间（默认为 1bit）跟随着包请求，此时主机和芯片都没有驱动信号线。

表 18-3 ACK 响应（3bits）

比特位	名称	描述
[2:0]	ACK	001：失败 010：等待 100：成功

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 18-4 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。

18.4.3. 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 代码是默认的 ARM 代码，并被置位 0x0BC11477（对应 Cortex-M0+）。

18.4.4. DP and AP 读/写访问

- 对 DP 的操作没有延迟：调试器将直接获取数据（ACK=OK），或者等待（ACK=WAIT）
- 对 AP 的读操作有延迟：这意味着访问的结果被返回到下一次传输。即前一次读操作的结果只能在下一次操作时获得。如果下一次的操作不是对 AP 的访问，则必需读 DP-RDBUFF 寄存器来获得上一次读操作的结果。
- DP-CTRL/STAT 寄存器的 READOK 标志在每个 AP 读操作或者 RDBUFF 读操作（以通知调试器 APD1 读操作是成功）时被更新。
- SW-DP 实现了写缓存区（对于 DP 和 AP 写），这甚至当其他操作在进行时时，任然可以接受写操作。如果写缓存满了，调试器将获得一个等待的 ACK 响应。读 IDCODE 寄存器，读 CTRL/STAT 寄存器和写 ABORT 寄存器操作在写缓冲区满时仍被接受。
- 由于 SWCLK 和 HCLK 的异步性，需要在写操作后(在奇偶校验位后)插入 2 个额外的 SWCLK 周期，以确保内部写操作正确完成。这两个额外的时钟周期需要在线路为低时插入(IDLE 状态下)。这个操作步骤在写 CTRL/STAT 寄存器以提出一个上电请求时尤其重要，否则下一个操作(在内核上电后才有效的操作)会立即执行，这将导致失败。

18.4.5. 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	R	-	IDCODE	固定为 0x0BC11477
00	W	-	ABORT	-
01	RW	0	DP_CTRL/STAT	- 请求一个系统或调试的上电操作； - 配置 AP 访问的操作模式； - 控制比较，校验操作； - 读取状态位
01	RW	1	WIRE CONTROL	配置串行通信物理层协议
10	R	-	READ RESEND	允许从一个错误的调试传输中恢复数据而不用重复最初的 AP 传输。
10	W	-	SELECT	选择当前的访问端口和有效的 4 字长寄存器窗口。
11	RW	-	READ BUFFER	由于 AP 的访问具有传递性（当前 AP 读操作的结果会在下次 AP 传输时传出），因此这个寄存器非常必要。这个寄存器会从 AP 捕获上一次读操作的数据结果，因此可以获得数据而不必再启动一个新的 AP 传输。

18.4.6. SW-AP 寄存器

当 APnDP=1 时，可以访问以下这些寄存器。

AP 寄存器的访问地址由以下两部分组成：

- A[3:2]的值。
- DP SELECT 寄存器的当前值。

18.5. 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。由下面四个寄存器组成

表 18-5 内核调试寄存器

寄存器	描述
DHCSR	32 位的调试控制和状态寄存器
DCRSR	17 位的内核寄存器调试选择寄存器
DCRDR	32 位的内核寄存器调试数据寄存器
DEMCR	32 位异常调试和监视控制寄存器

这些寄存器不会被系统复位。他们只会被上电复位。为了在复位后立即进入调试状态，需要：

- 使能调试和异常监视控制寄存器的 bit0（VC_CORRESET）。
- 使能调试控制和状态寄存器的 bit0（VC_DEBUGEN）

18.6. BPU 断点单元(Break Point Unit)

Cortex-M0+ BPU 实现提供了 4 个断点寄存器。

18.6.1. 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 的信息。

18.7. 数据观察点 DWT (Data Watchpoint)

Cortex-M0+ DWT 实现提供了 2 个数据观察点寄存器。

18.7.1. DWT 功能

处理器的断点实现基于 PC 的断点功能。

18.7.2. 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。

CORTEX-M0+ DWT_PCSR 记录了通过了条件代码的指令和未通过的指令。

18.8. DBGMCU 调试模块

MCU 调试模块给调试者提供以下支持：

- 低功耗模式
- 对 timer、看门狗在断点期间的时钟控制

18.8.1. 低功耗模式的调试支持

使用 WFI 和 WFE 可以进入低功耗模式。

MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的能耗。

内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。

为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

- 在睡眠模式下，调试器必须先置位 DBGMCU_CR 寄存器的 DBG_SLEEP 位。这将为 HCLK 提供与 FCLK(由代码配置的系统时钟)相同的时钟。
- 在停止模式下，调试器必须先置位 DBG_STOP 位。这将激活 HSI 时钟，在停止模式下为 FCLK 和 HCLK 提供时钟。

18.8.2. 支持定时器、看门狗的调试

在产生断点时，有必要根据定时器和看门狗的不同用途选择计数器的工作模式：

- 在产生断点时，计数器继续计数。
- 在产生断点时，计数器停止计数。

18.9. DBGMCU 寄存器

18.9.1. DBGMCU ID 编码(DBGMCU_IDCODE)

偏移地址：0x00

复位值：0x2020 0061

仅支持 32-bit 地址访问，只读。

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

Bit	Name	R/W	Reset Value	Function
31：0	DBG_IDCODE[31:0]	R	32’h2020 0061	MCU 的 ID 编码寄存器

18.9.2. DBGMCU 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。

偏移地址：0x04

复位值：0x0000 0000（不会被系统复位进行复位）

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

Bit	Name	R/W	Reset Value	Function
31：2	Reserved	-	-	保留
1	DBG_STOP	RW	0	调试停止模式。

				0：（FCLK 关，HCLK 关）。在停止模式下，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。 1：（FCLK 开，HCLK 开）。当进入停止模式，HSI 不会关闭，FCLK 和 HCLK 由 HSI 提供。当退出停止模式，如果需要改变时钟控制，软件需要重新配置。
0	DBG_SLEEP	RW	0	调试睡眠模式。 0：（FCLK 开，HCLK 关）。在睡眠模式，FCLK 由原先配置好的系统时钟提供，HCLK 关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出后，软件不需要重新配置时钟。 1：（FCLK 开，HCLK 开）。在睡眠模式，FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。

18.9.3. DBGMCU APB 冻结寄存器 1(DBGMCU_APB_FZ1)

该寄存器用来配置 timer、看门狗(IWDG)在调试模式下的时钟。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

偏移地址：0x08

Power on 复位值：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	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	DBG_ IWDG_STOP	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
			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
30:13	Reserved			保留
12	DBG_IWDG_STOP	RW	0	当 CPU 停止时，IWDG 计数器的时钟控制位 0：使能 1：不使能
11:0	Reserved			保留

18.9.4. DBGMCU APB 冻结寄存器 2(DBGMCU_APB_FZ2)

该寄存器用来配置 timer 在调试模式下的时钟控制。该寄存器被上电复位进行异步复位（不是系统复位）。它可以被调试者在系统复位下进行写。

偏移地址：0x0C

Power on 复位值：0x0000 0000

仅支持 32-bit 地址访问，只读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DBG_PWM1_STOP	Res	Res	Res
												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
31：20	Reserved			保留
19	DBG_PWM1_STOP	RW	0	当 CPU 停止时，PWM1 计数器的时钟控制位 0：使能 1：不使能
18：16	Reserved			保留
15	DBG_TIM14_STOP	RW	0	当 CPU 停止时，TIM14 计数器的时钟控制位 0：使能 1：不使能
14：12	Reserved			保留
11	DBG_TIM1_STOP	RW	0	当 CPU 停止时，TIM1 计数器的时钟控制位 0：使能 1：不使能
10：0	Reserved			保留

19. 版本历史

版本	日期	更新记录
V0.1	2025.12.24	预发布版本

Puya Semiconductor Co., Ltd.