导言

本参考手册针对应用开发，提供关于如何使用 WB32F10xxx 微控制器的存储器和外设的详细信息。WB32F10xxx 系列拥有不同的存储器容量、封装和外设配置。关于订货编号、电气和物理性能参数，请参考 WB32F10xxx 系列数据手册。

关于芯片内部闪存的编程，擦除和保护操作，请参考 WB32F10xxx 系列用户手册。关于 ARM Cortex™-M3 内核的具体信息，请参考 Cortex™-M3 技术参考手册。

Westberry 提供本文档仅用作基于 WB32F10xxx 系列微控制器设计的参考。Westberry 对错误和遗漏不承担任何责任。所有数据和规格如有更改，恕不另行通知。如需更多信息或问题，请联系：常州韦斯佰瑞电子科技有限公司（Westberry Technology (ChangZhou) Corp., Ltd）

参考文献

1. Cortex-M3 Technical Reference Manual (TRM) (Cortex-M3 技术参考手册):
请从 www.arm.com/documentation/ARMProcessor_Cores/index.html 下载
2. ARMv7-M Architecture Application Level Reference Manual(ARMv7-M 应用级架构参考手册)
请从 www.arm.com/products/CPUs/ARM_Cortex-M3_v7.html 下载
3. CoreSight Technology System Design Guide (CoreSight 技术系统设计指导)
请从 www.arm.com/documentation/Trace_Debug/index.html 下载
4. ARM Application Note 179: Cortex-M3 Embedded Software Development(ARM 应用笔记 179：Cortex-M3 嵌入式软件开发)
请从 www.arm.com/pdfs/DUI0205G_rvct_compiler_and_libraries_guide.pdf 下载

目录

第一章　文中的缩写

1.1　寄存器描述表中使用的缩写列表

在对寄存器的描述中使用了下列缩写：

缩写	描述
read/write (RW)	软件能读写此位。
read-only (R)	软件只能读此位。
write-only (W)	软件只能写此位，读此位将返回复位值。
read/clear (R_W1)	软件可以读此位，也可以通过写’1’清除此位，写’0’对此位无影响。
read/clear (R_W0)	软件可以读此位，也可以通过写’0’清除此位，写’1’对此位无影响。
read/clear by read (RC_R)	软件可以读此位；读此位将自动地清除它为’0’。
read/set (RS)	软件可以读也可以设置此位，写’0’对此位无影响。
set (S)	软件只能设置此位，写’0’对此位无影响。
read-only write trigger (RT_W)	软件可以读此位；写’0’或’1’触发一个事件但对此位数值没有影响。
toggle (T)	软件只能通过写’1’来翻转此位，写’0’对此位无影响。
Reserved (Res)	保留位，必须保持默认值不变

1.2　术语表

1.3　可用的外设

不同的型号可以支持的外设不同：
TIM1 TIM2 TIM3 TIM4
UART1 UART2 UART3
QSPI SPIS1 SPIM2 SPIS2
I2C1 I2C2 USB I2S
ADC

第二章　存储器和总线构架

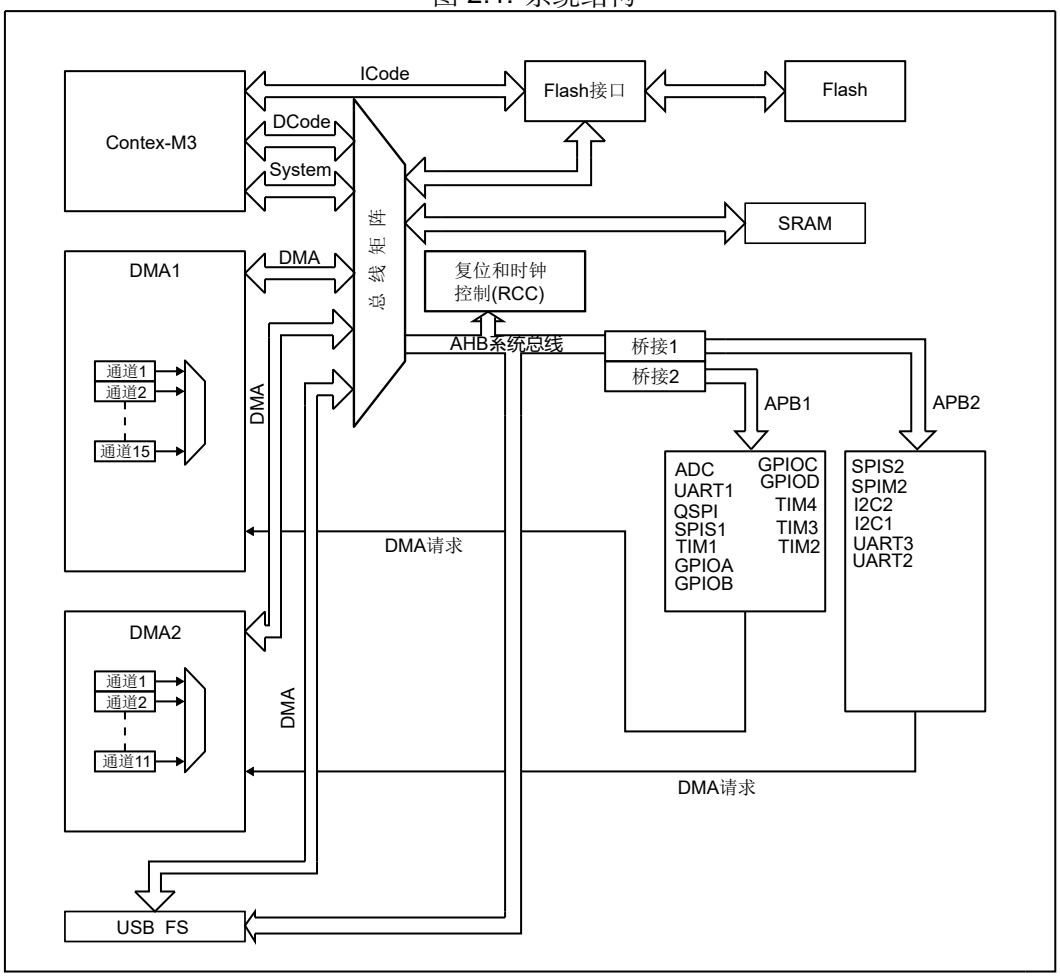
2.1　系统构架

主系统由以下部分构成:

- 五个驱动单元
  - Cortex™-M3 内核 DCode 总线 (D-bus)，和系统总线 (S-bus)
  - 通用 DMA1 和通用 DMA2
  - USB DMA
- 三个被动单元
  - 内部 SRAM
  - 内部闪存存储器
  - AHB 到 APB 的桥 (AHB2APBx)

这些都是通过一个多级的 AHB 总线构架相互连接的，如下图2.1所示：

图 2.1: 系统结构



ICode 总线

该总线将 Cortex™-M3 内核的指令总线与闪存指令接口相连接。指令预取在此总线上完成。

DCode 总线

该总线将 Cortex™-M3 内核的 DCode 总线与闪存存储器的数据接口相连接 (常量加载和调试访问)。

系统总线

此总线连接 Cortex™-M3 内核的系统总线 (外设总线) 到总线矩阵，总线矩阵协调着内核和 DMA 间的访问。

DMA 总线

此总线将 DMA 的 AHB 主控接口与总线矩阵相联，总线矩阵协调着 CPU 的 DCode 和 DMA 到 SRAM、闪存和外设的访问。

总线矩阵　总线矩阵协调内核系统总线和 DMA 主控总线之间的访问仲裁，仲裁利用轮换算法。在产品中，总线矩阵包含 5 个驱动部件 (CPU 的 DCode、系统总线、USB DMA、DMA1 总线和 DMA2 总线) 和 3 个从部件 (闪存存储器接口、SRAM 和 AHB2APB 桥)。AHB 外设通过总线矩阵与系统总线相连，允许 DMA 访问。**AHB/APB 桥 (APB)**

两个 AHB/APB 桥在 AHB 和 2 个 APB 总线间提供同步连接。APB1 和 APB2 操作于全速 (最高 128MHz)。表2.1 有关连接到每个桥的不同外设的地址映射请参考相应章节的寄存表。在每一次复位以后，所有除 SRAM 以外的外设都被关闭，在使用一个外设之前，必须设置 RCC 寄存器 AHBENR0/ AHBENR1/ AHBENR2/ APB1ENR/ APB2ENR 来打开该外设的时钟。

注意：当对 APB 寄存器进行 8 位或者 16 位访问时，该访问会被自动转换成 32 位的访问：桥会自动将 8 位或者 32 位的数据扩展以配合 32 位的向量。

2.2　存储器组织

程序存储器、数据存储器、寄存器和输入输出端口被组织在同一个 4GB 的线性地址空间内。

数据字节以小端格式存放在存储器中。一个字里的最低地址字节被认为是该字的最低有效字节，而最高地址字节是最高有效字节。

外设寄存器的映像请参考相关章节。

可访问的存储器空间被分成 8 个主要块，每个块为 512MB。

其他所有没有分配给片上存储器和外设的存储器空间都是保留的地址空间。

2.3　存储器映像

请参考相应器件的数据手册中的存储器映像图。表2.1 列出了所用 WB32F10xxx 中内置外设的起始地址。

表 2.1: 寄存器组起始地址

起始地址	外设	总线	寄存器映像
0x4001 7C00 - 0x4001 FFFF	保留	AHB	参见数据手册 4.3 节
0x4001 7800 - 0x4001 7BFF	FMC	AHB	参见数据手册 4.3 节
0x4001 6800 - 0x4001 77FF	保留	AHB	参见数据手册 4.3 节
0x4001 6400 - 0x4001 67FF	SYS	AHB	参见数据手册 4.3 节
0x4001 6000 - 0x4001 63FF	ISO	AHB	参见数据手册 4.3 节
0x4001 5C00 - 0x4001 5FFF	BKP	AHB	参见数据手册 4.3 节
0x4001 5800 - 0x4001 5BFF	RTC	AHB	参见数据手册 4.3 节
0x4001 5400 - 0x4001 57FF	CACHE	AHB	参见数据手册 4.3 节
0x4001 5000 - 0x4001 53FF	保留	AHB	参见数据手册 4.3 节
0x4001 4C00 - 0x4001 4FFF	SFM	AHB	参见数据手册 4.3 节
0x4001 4800 - 0x4001 4BFF	CRC	AHB	参见数据手册 4.3 节
0x4001 4400 - 0x4001 47FF	保留	AHB	参见数据手册 4.3 节
0x4001 4000 - 0x4001 43FF	USB	AHB	参见数据手册 4.3 节
0x4001 1000 - 0x4001 3FFF	保留	AHB	参见数据手册 4.3 节
0x4001 0C00 - 0x4001 0FFF	RCC	AHB	参见数据手册 4.3 节
0x4001 0800 - 0x4001 0BFF	IWDG	AHB	参见数据手册 4.3 节
0x4001 0400 - 0x4001 07FF	ANCTL	AHB	参见数据手册 4.3 节
0x4001 0000 - 0x4001 03FF	PWR	AHB	参见数据手册 4.3 节
0x4000 FC00 - 0x4000 FFFF	DMAC2	APB2	参见数据手册 4.3 节
0x4000 C000 - 0x4000 FBFF	保留	APB2	参见数据手册 4.3 节

0x4000 BC00 - 0x4000 BFFF	LED	APB2	参见数据手册 4.3 节
0x4000 B800 - 0x4000 BBFF	RNG	APB2	参见数据手册 4.3 节
0x4000 B400 - 0x4000 B7FF	I2S	APB2	参见数据手册 4.3 节
0x4000 9C00 - 0x4000 B3FF	保留	APB2	参见数据手册 4.3 节
0x4000 9800 - 0x4000 9BFF	WWDG	APB2	参见数据手册 4.3 节
0x4000 9400 - 0x4000 97FF	SPIS2	APB2	参见数据手册 4.3 节
0x4000 9000 - 0x4000 93FF	SPIM2	APB2	参见数据手册 4.3 节
0x4000 8C00 - 0x4000 8FFF	I2C2	APB2	参见数据手册 4.3 节
0x4000 8800 - 0x4000 8BFF	I2C1	APB2	参见数据手册 4.3 节
0x4000 8400 - 0x4000 87FF	UART3	APB2	参见数据手册 4.3 节
0x4000 8000 - 0x4000 83FF	UART2	APB2	参见数据手册 4.3 节
0x4000 7C00 - 0x4000 7FFF	DMAC1	APB1	参见数据手册 4.3 节
0x4000 4000 - 0x4000 7BFF	保留	APB1	参见数据手册 4.3 节
0x4000 3C00 - 0x4000 3FFF	ADC	APB1	参见数据手册 4.3 节
0x4000 3800 - 0x4000 3BFF	UART1	APB1	参见数据手册 4.3 节
0x4000 3400 - 0x4000 37FF	SPIS1	APB1	参见数据手册 4.3 节
0x4000 3000 - 0x4000 33FF	QSPI	APB1	参见数据手册 4.3 节
0x4000 2C00 - 0x4000 2FFF	保留	APB1	参见数据手册 4.3 节
0x4000 2800 - 0x4000 2BFF	TIM4	APB1	参见数据手册 4.3 节
0x4000 2400 - 0x4000 27FF	TIM3	APB1	参见数据手册 4.3 节
0x4000 2000 - 0x4000 23FF	TIM2	APB1	参见数据手册 4.3 节
0x4000 1C00 - 0x4000 1FFF	TIM1	APB1	参见数据手册 4.3 节
0x4000 1800 - 0x4000 1BFF	EXTI	APB1	参见数据手册 4.3 节
0x4000 1400 - 0x4000 17FF	AFIO	APB1	参见数据手册 4.3 节
0x4000 1000 - 0x4000 13FF	保留	APB1	参见数据手册 4.3 节
0x4000 0C00 - 0x4000 0FFF	GPIOD	APB1	参见数据手册 4.3 节
0x4000 0800 - 0x4000 0BFF	GPIOC	APB1	参见数据手册 4.3 节
0x4000 0400 - 0x4000 07FF	GPIOB	APB1	参见数据手册 4.3 节
0x4000 0000 - 0x4000 03FF	GPIOA	APB1	参见数据手册 4.3 节

2.3.1　嵌入式 SRAM

WB32F10xxx 内置 36K 字节的静态 SRAM。它可以以字节、半字 (16 位) 或全字 (32 位) 访问。SRAM 的起始地址是 0x2000 0000。

2.3.2　位段

Cortex™-M3 存储器映像包括两个位段 (bit-band) 区。这两个位段区将别名存储器区中的每个字映射到位段存储器区的一个位，在别名存储区写入一个字具有对位段区的目标位执行读-改-写操作的相同效果。

在 WB32F10xxx 里，外设寄存器和 SRAM 都被映射到一个位段区里，这允许执行单一的位段的写和读操作。

下面的映射公式给出了别名区中的每个字是如何对应位带区的相应位的：

bit_word_addr = bit_band_base + (byte_offset×32) + (bit_number×4)

其中：

bit_word_addr 是别名存储器区中字的地址，它映射到某个目标位。

bit_band_base 是别名区的起始地址。

byte_offset 是包含目标位的字节在位段里的序号

bit_number 是目标位所在位置 (0-31)

例子：

下面的例子说明如何映射别名区中 SRAM 地址为 0x20000300 的字节中的位 2：

0x22006008 = 0x22000000 + (0x300×32) + (2×4).

对 0x22006008 地址的写操作与对 SRAM 中地址 0x20000300 字节的位 2 执行读-改-写操作有着相同的效果。

读 0x22006008 地址返回 SRAM 中地址 0x20000300 字节的位 2 的值 (0x01 或 0x00)。

请参考《Cortex™-M3 技术参考手册》以了解更多有关位段的信息。

2.3.3　嵌入式闪存

闪存读取

闪存的指令和数据访问是通过 AHB 总线完成的。预取模块是用于通过 ICode 总线读取指令的。仲裁是作用在闪存缓存接口，并且 DCode 总线上的数据访问优先。读访问可以有以下配置选项：

- 等待时间：可以随时更改的用于读取操作的等待状态的数量。
- 预取缓冲区 (2 个 128 位)：在每一次复位以后被自动打开，由于每个缓冲区的大小 (128 位) 与闪存的带宽相同，因此只通过需一次读闪存的操作即可更新整个缓冲区的内容。由于预取缓冲区的存在，CPU 可以工作在更高的主频。CPU 每次取指最多为 32 位的字，取一条指令时，下一条指令已经在缓冲区中等待。

注：

1. 这些选项应与闪存存储器的访问时间一起使用。等待周期体现了系统时钟 (SYSCLK) 频率与闪存访问时间的关系：
   - 0 等待周期，当 $0MHz < SYSCLK \leq 32MHz$
   - 1 等待周期，当 $32MHz < SYSCLK \leq 48MHz$
   - 2 等待周期，当 $48MHz < SYSCLK \leq 72MHz$
   - 3 等待周期，当 $72MHz < SYSCLK \leq 96MHz$
   - 4 等待周期，当 $96MHz < SYSCLK \leq 128MHz$
   - 当频率大于 96MHz，必须使能高频辅助功能位 HIFREQ
2. 只有在系统时钟 (SYSCLK) 小于 48MHz 或者程序在 SRAM 执行时，才能执行预取缓冲器的打开和关闭操作。一般而言，在初始化过程中执行预取缓冲器的打开和关闭操作，这时微控制器的时钟由 8MHz 的内部 RC 振荡器 (MHSI) 提供。
3. 使用 DMA ： DMA 在 DCode 总线上访问闪存存储器。DMA 在每次传送完成后具有一个空余的周期。有些指令可以和 DMA 传输一起执行。

编程和擦除闪存

闪存编程一次写入一个页 (256 字节)。

闪存擦除操作可以按页面擦除或完全擦除 (全擦除)。全擦除不影响信息块。

为了确保正确的编程，闪存必须先完成擦除操作，同一个页不允许重复编程。

写操作 (编程或擦除) 结束时可以触发中断。仅当闪存控制器接口时钟开启时，此中断可以用来从 WFI 模式退出。

2.4　启动配置

在 WB32F10xxx 里，可以通过 BOOT[1:0] 引脚选择三种不同启动模式。

表 2.2: 启动模式

启动模式选择引脚		启动模式	说明
BOOT1	BOOT0		
X	0	主闪存存储器	主闪存存储器被选为启动区域
0	1	系统存储器	系统存储器被选为启动区域
1	1	内置 SRAM	内置 SRAM 被选为启动区域

在系统复位后，SYSCLK 的第 4 个上升沿，BOOT 引脚的值将被锁存。用户可以通过设置 BOOT1 和 BOOT0 引脚的状态，来选择在复位后的启动模式。

在从待机模式退出时，BOOT 引脚的值将被被重新锁存；因此，在待机模式下 BOOT 引脚应保持为需要的启动配置。在启动延迟之后，CPU 从地址 0x0000 0000 获取堆栈顶的地址，并从启动存储器的 0x0000 0004 指示的地址开始执行代码。

因为固定的存储器映像，代码区始终从地址 0x0000 0000 开始 (通过 ICode 和 DCode 总线访问)，而数据区 (SRAM) 始终从地址 0x2000 0000 开始 (通过系统总线访问)。Cortex-M3 的 CPU 始终从 ICode 总线获取复位向量，即启动仅适合于从代码区开始 (典型地从 Flash 启动)。WB32F10xxx 微控制器实现了一个特殊的机制，系统可以不仅仅从 Flash 存储器或系统存储器启动，还可以从内置 SRAM 启动。

根据选定的启动模式，主闪存存储器、系统存储器或 SRAM 可以按照以下方式访问：

- 从主闪存存储器启动：主闪存存储器被映射到启动空间 (0x0000 0000)，但仍然能够在它原有的地址 (0x0800 0000) 访问它，即闪存存储器的内容可以在两个地址区域访问，0x0000 0000 或 0x0800 0000。
- 从系统存储器启动：系统存储器被映射到启动空间 (0x0000 0000)，但仍然能够在它原有的地址 (原有地址为 0x1FFF E000) 访问它。
- 从内置 SRAM 启动：只能在 0x2000 0000 开始的地址区访问 SRAM。

注意：当从内置 SRAM 启动，在应用程序的初始化代码中，必须使用 NVIC 的异常表和偏移寄存器，重新映射向量表至 SRAM 中。

内嵌的自举程序

内嵌的自举程序存放在系统存储区，由 WestBerry 在生产线上写入，用于通过可用的串行接口对闪存存储器进行重新编程。

2.5　缓存寄存器

2.5.1　缓存控制寄存器 (CACHE_CR)

地址偏移量：0x00
复位值：0x0300 0000

表 2.3: 缓存控制寄存器 (CACHE_CR) 位描述

位	符号	访问	描述
31:26	-	R	保留
25:24	CHEEN	RW	缓存使能控制 00：缓存关闭 11：缓存打开 其它：保留配置
23:9	-	R	保留
8	HIFREQ	RW	高速访问辅助功能 0：高速访问辅助功能关闭 1：高速访问辅助功能打开（当系统频率大于 96MHz 必须打开）
7:6	-	R	保留
5:4	PREFEN	RW	使能指令预取功能 00: 禁用指令预取功能 01: 使能指令预取功能 其他: 保留配置
3:0	LATENCY	RW	闪存存储器访问等待周期 0x0：0 等待周期（0MHz < SYSCLK ≤ 32MHz） 0x1：1 等待周期（36HMz < SYSCLK ≤ 48MHz ） 0x2：2 等待周期（48MHz < SYSCLK ≤ 72MHz ） 0x3：3 等待周期（72MHz < SYSCLK ≤ 96MHz ） 0x4：4 等待周期（96MHz < SYSCLK ≤ 128MHz ） 0x5-0xF：5-15 等待周期（128MHz < SYSCLK)

第三章　系统配置 (SYS)

3.1　SYS 简介

系统配置模块主要是提供系统相关的信息，软件可以根据系统模块提供的信息来判断系统的功能和状况，并执行适当的操作。

用户可以得到如下信息：

- 产品的 ID；
- SRAM 和闪存容量大小；
- 系统启动的配置和状态；
- 系统的安全功能配置；
- 用户程序空间，系统程序空间和信息块的保护状态；

注：当信息块中的系统配置修改后，必须重新复位才能更新系统的配置信息。

3.2　SYS 主要功能

3.2.1　安全级别配置

WB32F10xxx 提供了不同的安全级别供用户配置。用户程序空间的保护由安全控制寄存器的不同配置决定。用户可以根据产品的开发的不同阶段和系统的需求配置不同的安全级别。

表 3.1: 安全保护级别配置

安全级别	RD_PROT	SRAM_DIS	SWD_DIS	SMEM_DIS	描述
级别 0	0	0	0	0	无安全保护
级别 1	1	1	0	1	不支持在系统升级程序，当进入 SWD 模式后，用户的程序不可读
级别 2	1	1	1	0	支持在系统升级程序
级别 3	1	1	1	1	最高优先级，升级程序由客户直接定义

除了正常的安全级别以外，当二次开发使能的时候，系统对二次开发程序空间提供额外的安全保护，详情请参考第 3.2.5 节。

注：当 *RD_PROT* 位设置后，*DMA* 将会被禁止从程序空间读取数据。

3.2.2　信息块的写保护状态

WB32F10xxx 共有 8 个信息块，每个信息块有 256 字节，设置有单独的保护状态。信息块和保护状态的对应关系如下表：

表 3.2: 信息块写保护状态

信息块编号	地址空间	保护位	描述
信息块 0	0x1FFF_F000 - 0x1FFF_F0FF	-	不可修改
信息块 1	0x1FFF_F100 - 0x1FFF_F1FF	-	不可修改
信息块 2	0x1FFF_F200 - 0x1FFF_F2FF	-	不可修改
信息块 3	0x1FFF_F300 - 0x1FFF_F3FF	-	不可修改
信息块 4	0x1FFF_F400 - 0x1FFF_F4FF	SYS_IF4LCK	启动控制配置
信息块 5	0x1FFF_F500 - 0x1FFF_F5FF	SYS_IF5LCK	主程序写保护配置
信息块 6	0x1FFF_F600 - 0x1FFF_F6FF	SYS_IF6LCK	二次开发配置
信息块 7	0x1FFF_F700 - 0x1FFF_F7FF	SYS_IF7LCK	重启功能配置及 OTP 空间

注：信息块 *0 -3* 处于写保护状态，用户不可更改。信息块 *4 -7* 的写保护状态由用户程序配置，但是一旦配置成写保护状态，就不能对该信息块擦写，否则会导致主程序损坏。

写保护未使能情况下，信息块 *7* 的 *0x1FFF_F720 - 0x1FFF_F7FF* 区域可以提供给用户作为 *OTP* 空间。

3.2.3　系统程序空间的写保护状态

WB32F10xxx 系统程序空间有 4K 字节。

系统程序空间的地址范围是：0x1FFF_E000 - 0x1FFF_EFFF

3.2.4　用户程序空间的写保护状态

WB32F10xxx 以 4K 字节为单位配置写保护功能，共有 32 个写保护控制位。当主程序空间超过 128K 字节时，保护状态由第 32 个写保护位控制。

程序空间和写保护状态的对应关系如下表3.2.4：

表 3.3: 主程序空间写保护状态（读保护关闭）

块编号	地址空间	写保护	描述
块 0	0x0800_0000 - 0x0800_0FFF	MAIN_WEN[0]	用户配置
块 1	0x0800_1000 - 0x0800_1FFF	MAIN_WEN[1]	用户配置
块 2	0x0800_2000 - 0x0800_2FFF	MAIN_WEN[2]	用户配置
块 3	0x0800_3000 - 0x0800_3FFF	MAIN_WEN[3]	用户配置
块 4	0x0800_4000 - 0x0800_4FFF	MAIN_WEN[4]	用户配置
块 5	0x0800_5000 - 0x0800_5FFF	MAIN_WEN[5]	用户配置
块 6	0x0800_6000 - 0x0800_6FFF	MAIN_WEN[6]	用户配置
块 7	0x0800_7000 - 0x0800_7FFF	MAIN_WEN[7]	用户配置
块 8	0x0800_8000 - 0x0800_8FFF	MAIN_WEN[8]	用户配置
块 9	0x0800_9000 - 0x0800_9FFF	MAIN_WEN[9]	用户配置

块 10	0x0800_A000 - 0x0800_AFFF	MAIN_WEN[10]	用户配置
块 11	0x0800_B000 - 0x0800_BFFF	MAIN_WEN[11]	用户配置
块 12	0x0800_C000 - 0x0800_CFFF	MAIN_WEN[12]	用户配置
块 13	0x0800_D000 - 0x0800_DFFF	MAIN_WEN[13]	用户配置
块 14	0x0800_E000 - 0x0800_EFFF	MAIN_WEN[14]	用户配置
块 15	0x0800_F000 - 0x0800_FFFF	MAIN_WEN[15]	用户配置
块 16	0x0801_0000 - 0x0801_0FFF	MAIN_WEN[16]	用户配置
块 17	0x0801_1000 - 0x0801_1FFF	MAIN_WEN[17]	用户配置
块 18	0x0801_2000 - 0x0801_2FFF	MAIN_WEN[18]	用户配置
块 19	0x0801_3000 - 0x0801_3FFF	MAIN_WEN[19]	用户配置
块 20	0x0801_4000 - 0x0801_4FFF	MAIN_WEN[20]	用户配置
块 21	0x0801_5000 - 0x0801_5FFF	MAIN_WEN[21]	用户配置
块 22	0x0801_6000 - 0x0801_6FFF	MAIN_WEN[22]	用户配置
块 23	0x0801_7000 - 0x0801_7FFF	MAIN_WEN[23]	用户配置
块 24	0x0801_8000 - 0x0801_8FFF	MAIN_WEN[24]	用户配置
块 25	0x0801_9000 - 0x0801_9FFF	MAIN_WEN[25]	用户配置
块 26	0x0801_A000 - 0x0801_AFFF	MAIN_WEN[26]	用户配置
块 27	0x0801_B000 - 0x0801_BFFF	MAIN_WEN[27]	用户配置
块 28	0x0801_C000 - 0x0801_CFFF	MAIN_WEN[28]	用户配置
块 29	0x0801_D000 - 0x0801_DFFF	MAIN_WEN[29]	用户配置
块 30	0x0801_E000 - 0x0801_EFFF	MAIN_WEN[30]	用户配置
块 31	0x0801_F000 - 0x0801_FFFF	MAIN_WEN[31]	用户配置
块 32	0x0802_0000 - 0x0803_FFFF	MAIN_WEN[31]	超过 128K 字节空间

当读保护功能打开时，根据启动模式，写保护的功能控制如下表3.4，3.5：

表 3.4: 主程序空间写保护状态（读保护开）（小容量产品）

启动模式	**0x0800_0000 - 0x0800_0FFF**	**其它空间**
SRAM 启动	写保护	写保护
系统程序启动	写保护	MAIN_WEN
用户程序启动	写保护	MAIN_WEN

表 3.5: 主程序空间写保护状态（读保护开）（大容量产品）

启动模式	**0x0800_0000 - 0x0800_1FFF**	**其它空间**
SRAM 启动	写保护	写保护
系统程序启动	写保护	MAIN_WEN
用户程序启动	写保护	MAIN_WEN

注：当读保护功能打开时，*192K* 字节的产品只支持 *4K* 字节写保护。

注：大容量产品是指主程序存储空间大于或等于 *128K* 字节的产品。小容量产品是指主程序存储空间小于 *64K* 字节的产品。

3.2.5　二次开发功能

WB32F10xxx 支持用户程序二次开发功能。启用二次开发功能后（SENDEV_EN = 1），Flash 存储器被分为两个部分——用户代码区域和专有代码区域。专有代码区域只能存放指令，专有代码程序需要读取的常量等数据必须存放在用户代码区域。CPU 可以从专有代码区域取指令用于执行，试图从专有代码区域读取数据将会产生硬件错误中断。

用户代码区域的大小是由 SENDEV_SIZE 的配置决定的，专有代码区域在 SENDEV_SIZE 空间的上方。例如：Flash 存储器的大小是 64KB，SENDEV_SIZE = 48，则专有代码程序存放在 48KB 到 64KB 的地址空间中。

在专有代码程序存放在 Flash 空间的上方后，用户要配置用户代码区域的大小并使能二次开发功能。配置完成后，需要复位芯片令二次开发功能有效。

3.3　SYS 寄存器描述

3.3.1　ID 寄存器 (SYS_ID)

地址偏移量：0x00
复位值：NA

表 3.6: ID 寄存器 (SYS_ID) 位描述

位	符号	访问	描述
31:28	-	R	固定值: 0x3
27:14	-	R	保留
13:0	CHIP_ID	R	芯片识别编码 固定值: 0x2980

3.3.2　存储控制寄存器 (SYS_MEMSZ)

地址偏移量：0x04
复位值：NA

表 3.7: 存储控制寄存器 (SYS_MEMSZ) 位描述

位	符号	访问	描述
31:6	-	R	保留
5:4	SRAM_SIZE	R	SRAM 容量 00：SRAM 容量是 36K 字节。 01：SRAM 容量是 28K 字节。 10：SRAM 容量是 20K 字节。 11：SRAM 容量是 12K 字节。

3:0	FLASH_SIZE	R	FLASH 容量 0000：保留配置。 0010：保留配置。 0011：FLASH 容量是 256K 字节。 0100：FLASH 容量是 128K 字节。 0110：FLASH 容量是 96K 字节。 0111：FLASH 容量是 64K 字节。 1111：FLASH 容量是 192K 字节。 其它：FLASH 容量是 32K 字节。

3.3.3　安全控制寄存器 (SYS_BTCR)

地址偏移量：0x0C

复位值：NA

表 3.8: 安全控制寄存器 (SYS_BTCR) 位描述

位	符号	访问	描述
31:25	-	R	保留
24	SMEM_DIS	R	从系统存储器启动控制 0：允许从系统存储器启动 1：禁止从系统存储器启动
23:17	-	R	保留
16	SRAM_DIS	R	从 SRAM 启动控制 0：允许从 SRAM 启动 1：禁止从 SRAM 启动
15:9	-	R	保留
8	SWD_DIS	R	SWD 使能控制 0：允许使用 SWD 1：禁止使用 SWD
7:1	-	R	保留
0	RD_PROT	R	读保护使能控制 0：读保护使能控制关闭 1：读保护使能控制打开

3.3.4　FLASH 写保护寄存器 (SYS_MEMWEN)

地址偏移量：0x10

复位值：NA

表 3.9: 存储控制寄存器 (SYS_MEMSZ) 位描述

位	符号	访问	描述
31:0	MAIN_WEN	R	FLASH 写保护控制 0：主存储块擦写禁止。(4K 字节) 1：主存储块擦写允许。(4K 字节)

注：每个控制位对应 *4K* 字节空间，对于大容量产品，位 *31* 同时控制 *128K* 字节以上的空间。

3.3.5　二次开发控制寄存器 (SYS_SENDEV)

地址偏移量：0x14
复位值：NA

表 3.10: 二次开发控制寄存器 (SYS_SENDEV) 位描述

位	符号	访问	描述
31:17	-	R	保留
16	SENDEV_EN	R	二次开发使能控制 0：二次开发功能禁止 1：二次开发功能允许
15:12	-	R	保留
11:0	SENDEV_SIZE	R	二次开发功能中用户代码区域的大小，单位是 1K 字节。 当二次开发功能使能时，大于 SENDEV_SIZE 的地址空间就是专有代码区域。

3.3.6　重启控制寄存器 (SYS_RSTCR)

地址偏移量：0x18
复位值：0x0000 000x

表 3.11: 重启控制寄存器 (SYS_RSTCR) 位描述

位	符号	访问	描述
31:3	-	R	保留
2	RST_STBY	R	进入 standby 模式时复位控制 0：进入 standby 模式时不复位 1：进入 standby 模式时复位
1	RST_STOP	R	进入 stop 模式时复位控制 0：进入 stop 模式时不复位 1：进入 stop 模式时复位
0	IWDG_EN	R	IWDG 启动控制 0：IWDG 通过软件启动 1：IWDG 自动启动，不需要软件配置

3.3.7　信息块 4 保护寄存器 (SYS_IF4LCK)

地址偏移量：0x1C
复位值：0x0000 000x

表 3.12: 信息块 4 保护寄存器 (SYS_IF4LCK) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SYS_IF4LCK	R	第 4 信息块擦写控制 1：禁止对第 4 信息块擦写 0：允许对第 4 信息块擦写

注：在信息块擦写禁止的情况下进行擦写操作会导致主程序损坏。

3.3.8　信息块 5 保护寄存器 (SYS_IF5LCK)

地址偏移量：0x20
复位值：0x0000 000x

表 3.13: 信息块 5 保护寄存器 (SYS_IF5LCK) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SYS_IF5LCK	R	第 5 信息块擦写控制 1：禁止对第 5 信息块擦写 0：允许对第 5 信息块擦写

注：在信息块擦写禁止的情况下进行擦写操作会导致主程序损坏。

3.3.9　信息块 6 保护寄存器 (SYS_IF6LCK)

地址偏移量：0x24
复位值：0x0000 000x

表 3.14: 信息块 6 保护寄存器 (SYS_IF6LCK) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SYS_IF6LCK	R	第 6 信息块擦写控制 1：禁止对第 6 信息块擦写 0：允许对第 6 信息块擦写

注：在信息块擦写禁止的情况下进行擦写操作会导致主程序损坏。

3.3.10　信息块 7 保护寄存器 (SYS_IF7LCK)

地址偏移量：0x28
复位值：0x0000 000x

表 3.15: 信息块 7 保护寄存器 (SYS_IF7LCK) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SYS_IF7LCK	R	第 7 信息块擦写控制 1：禁止对第 7 信息块擦写 0：允许对第 7 信息块擦写

注：在信息块擦写禁止的情况下进行擦写操作会导致主程序损坏。

3.3.11　启动状态寄存器 (SYS_BTSR)

地址偏移量：0x34
复位值：0x0000 0002

表 3.16: 启动状态寄存器 (SYS_BTSR) 位描述

位	符号	访问	描述
31:2	-	R	保留
1:0	BOOT_MODE	R	启动模式状态 00：保留 01：从系统程序空间启动 10：从用户程序空间启动 11：从 SRAM 空间启动

第四章　循环冗余校验 (CRC)

4.1　CRC 简介

4.2　CRC 主要特性

- 支持四个通用多项式 CRC-8、CRC-CCITT、CRC-16 和 CRC-32。
  CRC-8 : $x^8 + x^2 + x + 1$
  CRC-CCITT : $x^{16} + x^{12} + x^5 + 1$
  CRC-16 : $x^{16} + x^{15} + x^2 + 1$
  CRC-32 : $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- 针对输入数据及 CRC 和的位顺序取反和 1 的补码可编程设置。
- 可编程种子数设置
- 接受每次写操作任意大小数据宽度：8、16 或 32 位。
  8 位写操作：1 周期计算时间
  16 位写操作：2 周期计算时间
  32 位写操作：4 周期计算时间

4.3　CRC 寄存器描述

4.3.1　CRC 模式寄存器 (CRC_MODE)

地址偏移量：0x00
复位值：0x0000 0000

表 4.1: CRC 模式寄存器 (CRC_MODE) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	SEED_SET	RW	写 1 装载种子到 CRC。
6	SEED_OP	RW	CRC 种子设定选择。 0：使用多项式默认值 1：使用 CRC_SEED 寄存器作为 CRC 运算种子

5	CMPL_SUM	RW	CRC 校验和补码。 0：不对 CRC_SUM 采用 1 的补码 1：1 的补码（针对 CRC_SUM）
4	BIT_RVS_SUM	RW	CRC 校验和位顺序。 0：不对 CRC_SUM 进行位顺序取反 1：对 CRC_SUM 进行位顺序取反
3	CMPL_WR	RW	数据补码。 0：不对 CRC_WR_DATA 采用 1 的补码 1：1 的补码（针对 CRC_WR_DATA）
2	BIT_RVS_WR	RW	数据位顺序取反。 0：不对 CRC_WR_DATA 进行位顺序取反（每字节） 1：对 CRC_WR_DATA 进行位顺序取反（每字节）
1:0	CRC_POLY	RW	CRC 多项式选择。 00：选择 CRC-8 多项式（默认种子值为 0x00） 01：选择 CRC-CCITT 多项式（默认种子值为 0xFFFF） 10：选择 CRC-16 多项式（默认种子值为 0x0000） 11：选择 CRC-32 多项式（默认种子值为 0xFFFFFFFF）

4.3.2　CRC 种子寄存器 (CRC_SEED)

地址偏移量：0x04

复位值：0x0000 FFFF

表 4.2: CRC 种子寄存器 (CRC_SEED) 位描述

位	符号	访问	描述
31:0	CRC_SEED	RW	CRC 种子值

4.3.3　CRC 校验和寄存器 (CRC_SUM)

地址偏移量：0x08

复位值：0x0000 FFFF

该寄存器是只读寄存器，包含最近一次校验和。由于 CRC 校验和运算完成需要若干周期，数据写入后需要插入一个空操作才能启动对该寄存器读。

表 4.3: CRC 校验和寄存器 (CRC_SUM) 位描述

位	符号	访问	描述
31:0	CRC_SUM	R	CRC 校验和

4.3.4　CRC 数据寄存器 (CRC_WDATA)

地址偏移量：0x08

复位值：0xXXXX XXXX

该寄存器是只写寄存器，用于输入计算 CRC 和的数据块。

表 4.4: CRC 数据寄存器 (CRC_WDATA) 位描述

位	符号	访问	描述
31:0	CRC_WDATA	W	将使用写入该寄存器的数据按选定位顺序和 1 的补码预处理进行 CRC 计算。允许 8、16 或 32 位的任意写大小，并接受连续计算。

第五章　随机数发生器 (RNG)

5.1　简介

随机数发生器 (RNG) 使用 24 位 LFSR 产生 8 位随机数给其他模块使用，同时也可以从 APB2 总线读取这个随机数。

注 1：若要读写 RNG 的寄存器，必须先配置 RCC 寄存器来设置 APB2 总线时钟，并且将 RNG 的总线时钟使能。

注 2：若要 RNG 正常工作，必须配置 RCC 寄存器 RNGCLKENR 来打开 RNG 时钟。

请参考 RCC 寄存器 APB2PRE，APB2ENR 和 RNGCLKENR。

5.2　RNG 寄存器

5.2.1　随机数寄存器 (RNG_RAND)

地址偏移量：0x000

复位值：0x000

表 5.1: 随机数寄存器 (RNG_RAND) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:0	RNG_RAND	R	产生的 8 位随机数

5.2.2　算法暂停寄存器 (RNG_STOP)

地址偏移量：0x004

复位值：0x000

表 5.2: 算法暂停寄存器 (RNG_STOP) 位描述

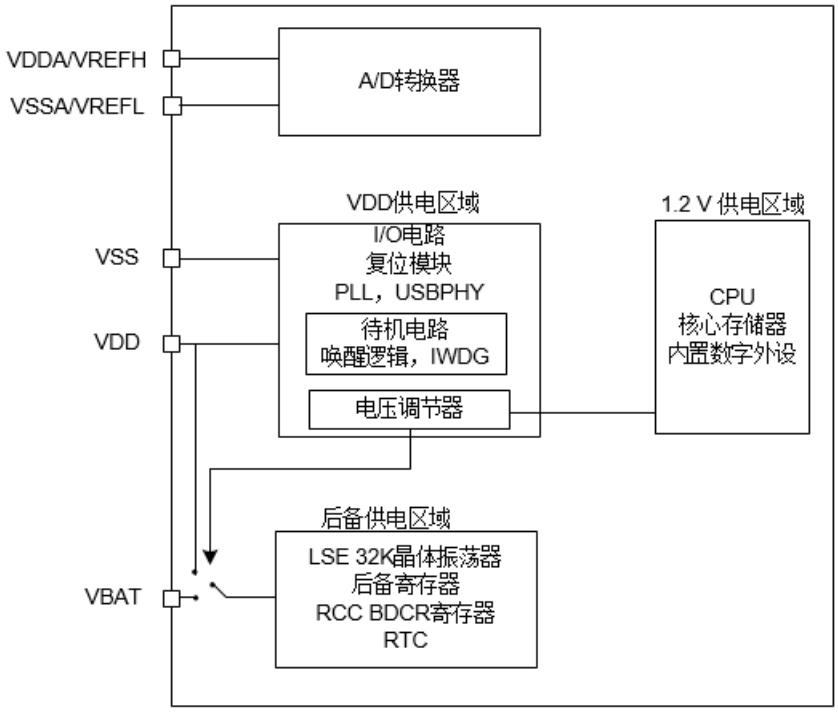
位	符号	访问	描述
31:1	-	R	保留
0	RNG_STOP	RW	算法暂停 1: LFSR 暂停 0: LFSR 继续

第六章　电源控制 (PWR)

6.1　电源

WB32F10xxx 的工作电压 (VDD) 为 2.0 ~ 3.6V。通过内置的电压调节器提供所需的 1.2V 数字电源。当主电源 VDD 掉电后，通过 VBAT 脚为实时时钟 (RTC) 和备份寄存器提供电源。备用电源电压为 1.8V ~ 3.6V。

图 6.1: 电源框图



6.1.1　独立的 A/D 转换器供电和参考电压

为了提高转换的精确度，ADC 使用一个独立的电源供电，过滤和屏蔽来自印刷电路板上的毛刺干扰。

- ADC 的电源引脚为 VDDA
- 独立的电源地 VSSA

模拟参考正极 (VREFH) 在芯片内部连接到 VDDA，模拟参考负极 (VREFL) 在芯片内部连接到 VSSA。这两个引脚不封装出去。

6.1.2　电池备份区域

使用电池或其他电源连接到 VBAT 脚上，当 VDD 断电时，可以保存备份寄存器的内容和维持 RTC 的功能。

VBAT 脚为 BKP、RTC、LSE 振荡器和 PC13 至 PC15 端口供电，可以保证当主电源被切断时 RTC 能继续工作。切换到 VBAT 供电的开关，由复位模块中的掉电复位功能控制。

警告：

- 在 VDD 上升阶段或者探测到 PDR(掉电复位) 之后，在一段时间之内（大概 50ns），VBAT 和 VDD 同时供电。这段时间之内，电流可能通过 VDD 和 VBAT 之间的内部二极管注入到 VBAT。
- 如果与 VBAT 连接的电源或者电池不能承受这样的注入电流，建议在外部 VBAT 和电源之间连接一个低压降二极管。

如果在应用中没有外部电池，建议 VBAT 在外部连接到 VDD 并连接一个 100nF 的陶瓷滤波电容。
当备份区域由 VDD(内部模拟开关连到 VDD) 供电时，下述功能可用：

- PC14 和 PC15 可以用于 GPIO 或 LSE 引脚
- PC13 可以作为通用 I/O 口、TAMPER 引脚、RTC 校准时钟、RTC 闹钟或秒输出

当后备区域由 VBAT 供电时 (VDD 消失后模拟开关连到 VBAT)，可以使用下述功能：

- PC14 和 PC15 只能用于 LSE 引脚
- PC13 可以作为 TAMPER 引脚、RTC 闹钟或秒输出 (参见第7.4.1节:RTC 时钟校准寄存器 (BKP_RTCCR))

注：因为模拟开关只能通过少量的电流 (3mA)，在输出模式下使用 PC13 至 PC15I/O 口功能是有限制的：速度必须限制在 2MHz 以下，最大负载为 30pF，而且这些 I/O 口绝对不能当作电流源 (如驱动 LED)。

6.1.3　电压调节器

复位后调节器总是使能的。根据应用方式它有以下 3 种不同的模式：

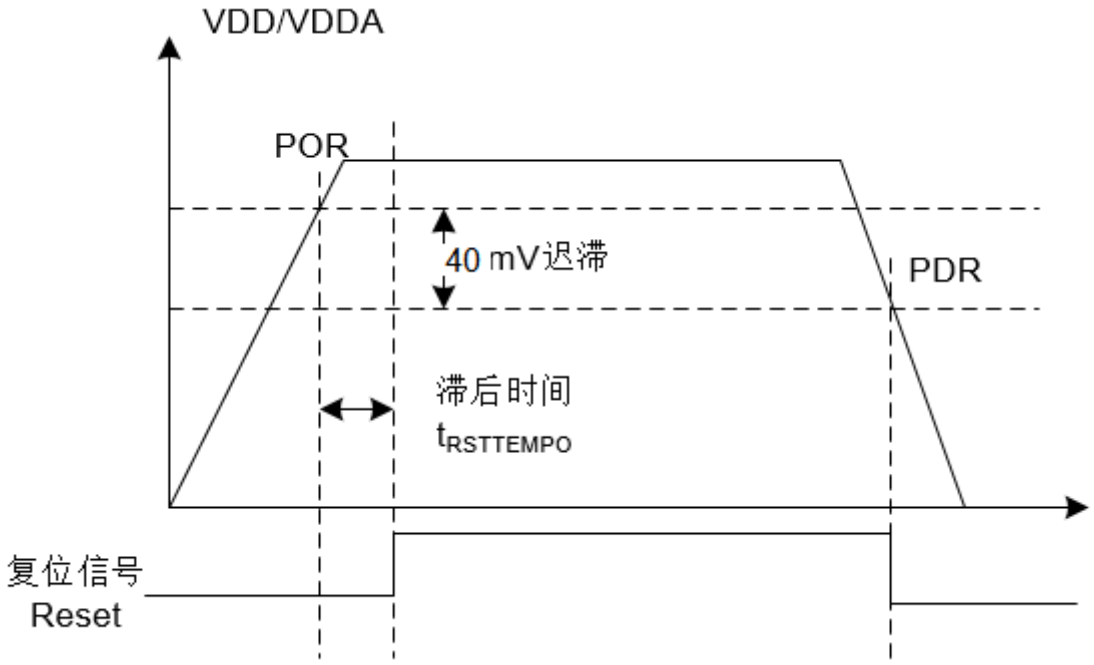
- 正常工作模式：调节器以正常功耗模式提供 1.2V 电源 (内核，内存和外设)。
- 低功耗模式：调节器以低功耗模式提供 1.2V 电源。
- 深度低功耗模式：不需要提供参考电压，仍然可以提供 1.2V 电源，以保存寄存器和 SRAM 的内容，但是驱动能力非常小。

6.2　电源管理器

6.2.1　上电复位 (POR) 和掉电复位 (PDR)

WB32F10xxx 内部有一个完整的上电复位 (POR) 和掉电复位 (PDR) 电路。当供电电压达到 2.0V 时系统即能正常工作。当 VDD/VDDA 低于指定的限位电压 V_{POR}/V_{PDR} 时，系统保持为复位状态，而无需外部复位电路。关于上电复位和掉电复位的细节请参考数据手册。

图 6.2: 上电复位和掉电复位的波形图



6.2.2　可编程电压监测器 (PVD)

用户可以利用 PVD 对 VDD 电压和一定的阈值进行比较来监控电源。模拟控制模块寄存器 PVDCR 的 PLS 位可以配置监控电压的阈值。

通过设置模拟控制模块 PVDENR 的位 PVDE 来使能 PVD。

寄存器 PWR_SR0 用来表明 VDD 是高于还是低于 PVD 的电压阀值。当 PVDO 为 1 时，表示 VDD 低于 PVD 的电压阈值。该事件连接到外部中断的第 16 线。如果该中断在外部中断寄存器中是使能的，当 VDD 下降到 PVD 阀值以下或当 VDD 上升到 PVD 阀值之上时，根据外部中断第 16 线的上升/下降边沿触发设置，就会产生 PVD 中断。该中断可用于执行一些紧急关闭任务。

6.3　低功耗模式

在系统或电源复位以后，微控制器处于运行状态。当 CPU 不需要继续运行时，可以利用多种低功耗模式来节省功耗，例如等待某个外部事件时。用户需要根据低电源消耗、快速启动时间和可用的唤醒源等条件，选定一个佳的低功耗模式。

WB32F10xxx 有三种低功耗模式：

- 睡眠模式 (Cortex™-M3 内核停止，所有外设包括 Cortex-M3 核心的外设，如 NVIC、系统时钟 (SysTick) 等仍在运行)
- 停止模式 (所有的时钟都已停止)
- 待机模式 (1.2V 电源关闭)

此外，在运行模式下，可以通过以下方式中的一种或几种降低功耗：

- 降低系统时钟的频率

- 关闭系统总线上未被使用的外设时钟

表 6.1: 低功耗模式一览

模式	进入	唤醒	对 1.2V 区域时钟的影响	对 VDD 区域时钟的影响	电压调节器
睡眠	WFI	任一中断	CPU HCLK 时钟关闭，FCLK 频率降低，对其他时钟和 ADC 时钟无影响	无	开
	WFE	唤醒事件			
停机	PDDS(0) + SLEEPDEEP 位 + WFI/WFE	任一外部中断 (在外部中断寄存器中设置)	关闭所有 1.2V 区域的时钟	HSI 和 HSE 的振荡器关闭	开启或处于低功耗模式
待机	PDDS 位 (1)+SLEEP-DEEP 位 +WFI/WFE	WKUP 引脚的上升沿、RTC 闹钟事件、NRST 引脚上的外部复位、IWDG 复位			深度低功耗模式

6.3.1　降低系统时钟频率

在运行模式下，通过对预分频寄存器进行编程，可以降低任意一个系统时钟 (SYSCLK、HCLK、FCLK) 的速度。进入睡眠模式前，也可以利用预分频器来降低外设的时钟。详情请参考 RCC 模块相关章节。

6.3.2　外部时钟的控制

在运行模式下，任何时候都可以通过停止外设和内存的时钟来减少功耗。为了在睡眠模式下更多地减少功耗，可在执行 WFI 或 WFE 指令前关闭所有外设的时钟。通过设置 AHBENR0，AHBENR1，AHBENR2，APB1ENR 和 APB2ENR 来开关各个外设模块的时钟。

6.3.3　睡眠模式

进入睡眠模式

通过执行 WFI 或 WFE 指令进入睡眠状态。根据 Cortex™-M3 系统控制寄存器中的 SLEEPONEXIT 位的值，有两种选项可用于选择睡眠模式进入机制：

- SLEEP-NOW：如果 SLEEPONEXIT 位被清除，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。
- SLEEP-ON-EXIT：如果 SLEEPONEXIT 位被置位，系统从低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

在睡眠模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。

关于如何进入睡眠模式，更多的细节参考表6.2和表6.3。

退出睡眠模式

如果执行 WFI 指令进入睡眠模式，任意一个被嵌套向量中断控制器响应的外设中断都能将系统从睡眠模式唤醒。

如果执行 WFE 指令进入睡眠模式，则一旦发生唤醒事件时，微处理器都将从睡眠模式退出。唤醒事件可以通过下述方式产生：

- 在外设控制寄存器中使能一个中断，而不是在 NVIC(嵌套向量中断控制器) 中使能，并且在 Cortex-M3 系统控制寄存器中使能 SEVONPEND 位。当 MCU 从 WFE 中唤醒后，外设的中断挂起位和外设的 NVIC 中断通道挂起位 (在 NVIC 中断清除挂起寄存器中) 必须被清除。
- 配置一个外部或内部的 EXIT 线为事件模式。当 MCU 从 WFE 中唤醒后，因为与事件线对应的挂起位未被设置，不必清除外设的中断挂起位或外设的 NVIC 中断通道挂起位。该模式唤醒所需的时间短，因为没有时间损失在中断的进入或退出上。

关于如何退出睡眠模式，更多的细节参考表6.2和表6.3。

表 6.2: SLEEP-NOW 模式

SLEEP-NOW	**说明**
进入	在以下条件执行 WFI(等待中断) 或 WFE(等待事件) 指令： –SLEEPDEEP = 0 和 –SLEEPONEXIT = 0 参考 Cortex-M3 系统控制寄存器。
退出	如果执行 WFI 进入睡眠模式： -中断：参考中断向量表 (表10.1.1) 如果执行 WFE 进入睡眠模式： -中断：参考中断向量表 (表10.1.1) -唤醒事件：参考唤醒事件管理 (第十一章)
唤醒延时	无

表 6.3: SLEEP-ON-EXIT 模式

SLEEP-ON-EXIT	**说明**
进入	在以下条件执行 WFI 指令： –SLEEPDEEP = 0 和 –SLEEPONEXIT = 1 参考 Cortex-M3 系统控制寄存器。
退出	中断：参考中断向量表 (表10.1.1)
唤醒延时	无

睡眠模式深度降低功耗

在睡眠模式下，可以通过配置进一步降低功耗：

- 将 Cortex-M3 的时钟 FCLK 频率降低一半。详细信息请参考6.4.1

6.3.4　停止模式

停止模式是在 Cortex™-M3 的深睡眠模式基础上结合了外设的时钟控制机制。在停止模式下，电压调节器可运行在正常，低功耗模式或深度低功耗模式。此时在 1.2V 供电区域的的所有时钟都被停止，PLL、MHSI、FHSI 和 HSE RC 振荡器的功能被禁止，SRAM 和寄存器内容被保留下来。

在停止模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。

进入停止模式

关于如何进入停止模式，详见表6.5。如果正在进行闪存编程，直到对内存访问完成，系统才进入停止模式。如果正在进行对 APB 的访问，直到对 APB 访问完成，系统才进入停止模式。

用户通过配置，有四种停止模式可以选择。

停止模式	进入停止模式后，各模拟部分的配置	CR0 配置	CFGR 配置
SP1	–关闭 MHSI，FHSI，HSE 和 PLL –电压调节器正常工作，输出电压值降低 4 个 step（大概降低 100mV） –闪存器进入睡眠模式 –闪存器用的电压调节器正常工作 –BandGap 正常工作	0x900	0x3F
SP2	–关闭 MHSI，FHSI，HSE 和 PLL –电压调节器进入低功耗模式 –闪存器断电 –闪存器用的电压调节器进入低功耗模式 –BandGap 正常工作	0x22002	0x3BE
SP3	–关闭 MHSI，FHSI，HSE 和 PLL –电压调节器进入超低功耗模式（不需要参考电压，仍然可以输出电压，但是驱动能力很低） –闪存器断电 –闪存器用的电压调节器关闭 –BandGap 正常工作	0x63004	0x3BF
SP4	–关闭 MHSI，FHSI，HSE 和 PLL –电压调节器进入超低功耗模式（不需要参考电压，仍然可以输出电压，但是驱动能力很低） –闪存器断电 –闪存器用的电压调节器关闭 –BandGap 关闭	0x7B004	0x3B3

Note: 寄存器 CR0 中的 DBP 位与停止模式无关。在配置时请注意保持原来状态。

通过对独立的控制位进行编程，可选择以下功能：

- 独立看门狗 (IWDG)：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。详见十九。
- 实时时钟 (RTC)：通过备份域控制寄存器 (BKP_BDCR) 的 RTCEN 位来设置。
- 内部 RC 振荡器 (LSI RC)：通过模拟控制寄存器 (ANCTL_LSICR) 的 LSION 位来设置。
- 外部 32.768kHz 振荡器 (LSE)：通过备份域控制寄存器 (BKP_BDCR) 的 LSEON 位设置。

在停止模式下，如果在进入该模式前 ADC 没有被关闭，那么这些外设仍然消耗电流。通过清除 ANCTL_SARENR 的 SAREN 位和 ADC_CR2 的 ADON 位可关闭 ADC 转换。

退出停止模式

关于如何退出停止模式，详见表6.5。当一个中断或唤醒事件导致退出停止模式时，MHSI RC 振荡器被选为系统时钟。如果电压调节器处于低功耗模式或者超低功耗模式，当系统从停止模式退出时，将会有一段

额外的启动延时。从停止模式退出后，所有的时钟配置都回到初始状态，用户需要重新配置 RCC。如果由于中断或者事件提前到来，系统没有成功进入停止模式，可以通过查询 PWR_SR1 的 CKF 标志位检查是否所有的始终配置都回到了初始状态。*Note: 退出待机模式之后，FHSI 被关闭。如果用户需要使用 FHSI，请自行打开。*

表 6.5: 停止模式

停止模式	说明
进入	在以下条件下执行 WFI(等待中断) 或 WFE(等待事件) 指令： –设置 Cortex-M3 系统控制寄存器中的 SLEEPDEEP 位 –设置电源控制寄存器 0 中的 PDDS 位为 2'b00 –通过设置电源控制寄存器 0 选择电压调节器的模式 注：为了进入停止模式，所有外部中断的请求位和 RTC 的闹钟标志都必须被清除，否则停止模式的进入流程将会被跳过，程序继续运行。
退出	如果执行 WFI 进入停止模式： - 设置任一外部中断线为中断模式 (在 NVIC 中必须使能相应的外部中断向量)。参见中断向量表10.1.1 。 如果执行 WFE 进入停止模式： - 设置任一外部中断线为事件模式。参见唤醒事件管理十一。
唤醒延时	MHSI RC 唤醒时间 + 电压调节器从低功耗唤醒的时间 + BandGap 唤醒的时间（根据配置）。

6.3.5　待机模式

待机模式可实现系统的低功耗。在该模式时，Cortex-M3 处于深睡眠模式，整个 1.2V 供电区域被断电，PLL、MHSI、FHSI 和 HSE 振荡器等也被断电。SRAM 和寄存器内容丢失。电压调节器处于深度低功耗模式，为备份域寄存器，RTC 和待机电路供电。

进入待机模式

关于如何进入待机模式，详见表6.6。可以通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗 (IWDG)：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。详见第十九章。
- 实时时钟 (RTC)：通过备用区域控制寄存器 (BKP_BDCR) 的 RTCEN 位来设置。
- 内部 RC 振荡器 (LSI RC)：通过模拟控制寄存器 (ANCTL_LSICR) 的 LSION 位来设置。
- 外部 32.768kHz 振荡器 (LSE)：通过备用区域控制寄存器 (BKP_BDCR) 的 LSEON 位设置。

退出待机模式

当一个外部复位 (NRST 引脚)、IWDG 复位、WKUP 引脚上的上升沿或 RTC 闹钟事件的上升沿发生时，微控制器从待机模式退出。从待机模式唤醒之后，产生系统复位。从待机模式唤醒后的代码执行等同于其它系统复位后的执行 (采样启动模式引脚、读取复位向量等)。电源控制状态寄存器 STA_1 将会指示内核由待机状态退出。关于如何退出待机模式，详见表6.6。

表 6.6: 待机模式

停止模式	说明
进入	在以下条件下执行 WFI(等待中断) 或 WFE(等待事件) 指令: –设置 Cortex™-M3 系统控制寄存器中的 SLEEPDEEP 位 –设置电源控制寄存器 0 中的 PDDS 位为 2'b10 –清除电源控制状态寄存器 1 中的 WUF 位
退出	WKUP 引脚的上升沿、RTC 闹钟事件的上升沿、NRST 引脚上外部复位、IWDG 复位。
唤醒延时	复位阶段时电压调节器的启动。

待机模式下的输入/输出端口状态

在待机模式下，所有的 I/O 引脚处于高阻态，除了以下的引脚:

- 复位引脚 (始终有效)
- 当被设置为防侵入或校准输出时的 TAMPER 引脚
- 被使能的唤醒引脚

调试模式

默认情况下，如果在进行调试微处理器时，使微处理器进入停止或待机模式，将失去调试连接。这是因为 Cortex™-M3 的内核失去了时钟。然而，通过设置 DBGMCU_CR 寄存器中的某些配置位，可以在使用低功耗模式下调试软件。更多的细节请参考31.13.1

6.3.6　低功耗模式下各模块状态

进入睡眠模式后，各模块的状态不变，只是 CPU 的 HCLK 关闭，FCLK 时钟频率减半;
进入停止模式后，根据 PWR_CR0 的配置，各模块可以被关闭或者进入低功耗模式;
进入待机模式后，各模块均进入低功耗模式或者停止模式。
详情见下表:

	MAIN REG	Panel	MHSI	FHSI	LSI	HSE	PLL	PVD	POR BG	POR ZP	4K SRAM	32K SRAM	内核 logic	ADC
Active	ON	ON	ON	ON	CFG	CFG	CFG	CFG	ON	ON	ON	ON	ON	CFG
睡眠模式	ON	ON	ON	ON	CFG	CFG	CFG	CFG	ON	ON	ON	ON	ON	CFG
停止模式	CFG	CFG	OFF	OFF	CFG	OFF	OFF	CFG	CFG	ON	CFG	CFG	ON	CFG
待机模式	ULP	PD	OFF	OFF	CFG	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	CFG

注意：表格中的 OFF 表示模式处于处于不工作状态；flash 的 PD 表示 power down 模式；CFG 表示用户可以配置；ULP 表示超低功耗模式。

6.3.7　低功耗模式下的自动唤醒 (AWU)

RTC 可以在不需要依赖外部中断的情况下唤醒低功耗模式下的微控制器 (自动唤醒模式)。RTC 提供一个可编程的时间基数，用于周期性从停止或待机模式下唤醒。通过对备份区域控制寄存器 (BKP_BDCR) 的 RTCSEL[1:0] 位的编程，三个 RTC 时钟源中的二个时钟源可以选作实现此功能。

- 外部晶振 LSE: 该时钟源提供了一个低功耗且精确的时间基准。
- 内部 RC 振荡器 LSI: 使用该时钟源，节省了一个 32.768kHz 晶振的成本。但是 RC 振荡器将少许增加电源消耗。

为了用 RTC 闹钟事件将系统从停止模式下唤醒，必须进行如下操作：

- 配置外部中断线 17 为上升沿触发。
- 配置 RTC 使其可产生 RTC 闹钟事件。

如果要从待机模式中唤醒，不必配置外部中断线 17。

6.4　电源控制寄存器

6.4.1　控制寄存器 0 (PWR_CR0)

地址偏移：0x00
复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留。始终读为 0。
19	S4KMODE	RW	配置 4K SRAM 在进入停止模式时的状态 1：在进入停止模式时，关掉 4K SRAM 的电源 0：在进入停止模式时，保持 4K SRAM 的电源
18	S32KMODE	RW	配置 32K SRAM 在进入停止模式时的状态 1：在进入停止模式时，关掉 32K SRAM 的电源 0：在进入停止模式时，保持 32K SRAM 的电源
17	FLASHMODE	RW	配置闪存器在进入停止模式时的状态 1：在进入停止模式时，闪存器进入关机模式 (power down mode) 0：在进入停止模式时，闪存器进入睡眠模式
16	PORMODE	RW	配置 POR 在进入停止模式时的状态 1：在进入停止模式时，关闭 POR 0：在进入停止模式时，POR 继续工作
15:14	BGMODE	RW	配置带隙基准模块在进入停止模式时的状态 如果在进入停止模式时，有部分模拟模块仍正常工作，带隙基准模块就处于正常工作状态。该位的配置只有在所有模拟部分都停止工作时起作用。 00：在进入停止模式时，带隙基准模块正常工作 10：在进入停止模式时，关闭带隙基准模块 其它：没有定义，不支持

13:12	FREGMODE	RW	配置闪存器专用电压调节器在进入停止模式时的状态 0x：在进入停止模式时，闪存器专用电压调节器正常工作 10：在进入停止模式时，闪存器专用电压调节器进入低功耗模式 11：在进入停止模式时，关闭闪存器专用电压调节器
11:9	REGLVL	RW	配置电压调节器的步进值 进入停止模式时，如果电压调节器处于正常工作模式，可以降低其输出电压，以达到省电的目的。该位用来配置调节的步进值。 000: 0 步进值　100: 4 步进值 001：1 步进值　101: 5 步进值 010: 2 步进值　110: 6 步进值 011：3 步进值　111: 7 步进值 注：1 步进值是大概 23mV。
8:7	REGCFG	RW	配置电压调节器的电压变化方向 进入停止模式时，如果电压调节器处于正常工作模式，可以降低其输出电压，以达到省电的目的。该位用来配置电压调节的方向。 0x：在进入停止模式时，电压调节器的输出电压不变 10：在进入停止模式时，降低电压调节器的输出电压 11：在进入停止模式时，升高电压调节器的输出电压 注：当升高或者降低输出电压时，具体调节的步进值由 REGLVL 配置。
6:5	PDDS	RW	当 M3 进入深度睡眠模式时，配置系统的工作状态。 00：当 M3 进入深度睡眠模式时，系统进入停止模式 10：当 M3 进入深度睡眠模式时，系统进入待机模式 01/11：没有定义，不支持
4	-	R	保留。始终读为 0。
3	FCLKSD	RW	降低 M3 FCLK 的频率 在进入睡眠模式时，可以降低 M3 FCLK 的频率，从而降低功耗。 该位仅在睡眠模式时被用到。 1: 进入睡眠模式时，如果 M3 FCLK 的分频系数小于 32 (参考 RCC 模式的寄存器描述)，M3 FCLK 的频率降低为原来的一半 0: 进入睡眠模式时，M3 FCLK 的频率不变
2:1	REGMODE	RW	配置电压调节器在进入停止模式时的状态 00：在进入停止模式时，电压调节器正常工作 01：在进入停止模式时，电压调节器进入低功耗模式 10：在进入停止模式时，电压调节器进入超低功耗模式 (这时不需要带隙基准的输出电压) 11：没有定义，不支持
0	DBP	RW	取消后备区域的写保护 在复位后，RTC 和后备寄存器处于被保护状态以防意外写入。设置该位允许写入这些寄存器。 0：禁止写入 RTC 和后备寄存器 1：允许写入 RTC 和后备寄存器 注：如果 RTC 的时钟是 HSE/128，该位必须保持为’1’。

注意：

- 如果在睡眠模式下，电压调节器的输出电压发生变化，在退出睡眠模式时，会自动恢复到正常状态。
- 如果在停止模式下，SRAM、闪存器和其他的模拟部分进入省电模式，在退出停止模式时，硬件会重新启动它们，保证它们可以正常工作；
- 在待机模式下，SRAM、闪存器和其他的模拟部分都进入省电模式。在退出停止模式时，会启动系统复位，CPU 指令从初始地址开始工作。

6.4.2　控制寄存器 1 (PWR_CR1)

地址偏移：0x04
复位值：0x0000 0000

位	符号	访问	描述
31:4	-	R	保留。始终读为 0。
3	CCKF	W	清除 CKF 标志位 该位只能写，不能读，读出时一直为 0。 软件向该位写 1 时，硬件会清除 CKF 标志位，写 0 无效
2	CSPF	W	清除 SPF 标志位 该位只能写，不能读，读出时一直为 0。 软件向该位写 1 时，硬件会清除 SPF 标志位，写 0 无效
1	CSBF	W	清除 SBF 标志位 该位只能写，不能读，读出时一直为 0。 软件向该位写 1 时，硬件会清除 SBF 标志位，写 0 无效
0	CWUF	W	清除 WUF 标志位 该位只能写，不能读，读出时一直为 0。 软件向该位写 1 时，硬件会清除 WUF 标志位，写 0 无效

6.4.3　控制寄存器 2 (PWR_CR2)

地址偏移：0x08
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留。始终读为 0。
0	EWUP	RW	使能 WKUP 引脚 0：WKUP 引脚为通用 I/O。WKUP 引脚上的事件不能将 CPU 从待机模式唤醒 1：WKUP 引脚用于将 CPU 从待机模式唤醒，WKUP 引脚被强置为输入下拉的配置，WKUP 引脚上的上升沿将系统从待机模式唤醒。 注：在系统复位时清除该位。

6.4.4　状态寄存器 0 (PWR_SR0)

地址偏移：0x10

复位值：0x0000 0000
该寄存器只能读取，写无效。

位	符号	访问	描述
31:1	-	R	保留。始终读为 0。
0	PVDO	R	PVD 输出 0：VDD 高于选定的 PVD 阀值 1：VDD 低于选定的 PVD 阀值。

注：在待机模式下 *PVD* 被停止。因此，待机模式后或复位后，直到设置 *PVDE* 位之前，该位为 *0*。

6.4.5　状态寄存器 1 (PWR_SR1)

地址偏移：0x14
复位值：0x0000 0000
该寄存器只能读取，写无效。
该寄存器的状态只在上电/掉电时复位，系统复位时内容不变。
在系统退出停止模式和待机模式时，用户需要读取该寄存器来判断系统的状态。

位	符号	访问	描述
31:4	-	R	保留。始终读为 0。
3	CKF	R	CKF 标志位 读取时如果该位是 1，表示在进入停止模式或者待机模式时，系统的时钟配置回到初始状态，用户需要做以下操作 - 写 CCKF 为 1 来清除该标志位 - 重新配置时钟
2	SPF	R	SPF 标志位 读取时如果该位是 1，表示系统从停止模式退出。用户需要写 CTL_1 的 CSPF 来清除该标志位。
1	SBF	R	SBF 标志位 读取时如果该位是 1，表示系统从待机模式退出。用户需要写 CTL_1CSBF 来清除该标志位。
0	WUF	R	WUF 标志位 读取时如果该位是 1，表示 WKUP 引脚上发生唤醒事件或者出现 RTC 闹钟事件。用户需要写 CTL_1CWUF 来清除该标志位。 注：当 WKUP 引脚已经是高电平时，在通过设置 EWUP 位使能 WKUP 引脚时，会检测到一个额外的事件。

6.4.6　通用寄存器 0 (PWR_GPREG0)

地址偏移：0x18
复位值：0x0000 0000
该寄存器的状态只在上电/掉电时复位，系统复位时内容不变。
该寄存器是 32 位通用寄存器 0。

6.4.7　通用寄存器 1 (PWR_GPREG1)

地址偏移：0x1C
复位值：0x0000 0000
该寄存器的状态只在上电/掉电时复位，系统复位时内容不变。
该寄存器是 32 位通用寄存器 1。

6.4.8　配置寄存器 (PWR_CFGR)

地址偏移：0x20
复位值：0x0000 03BE
该寄存器的状态只在上电/掉电时复位，系统复位时内容不变。

位	符号	访问	描述
31:13	-	R	保留。
12:0	LPCFG	RW	配置进入和退出低功耗模式的时间。 0x3F : 配置进入和退出停止模式 LP1 时的时间。 0x3BE: 配置进入和退出停止模式 LP2 时的时间。 0x3BF: 配置进入和退出停止模式 LP3 时的时间。 0x3B3: 配置进入和退出停止模式 LP4 时的时间。 0x13B3: 配置进入和退出待机模式的时间。 其它：保留。

6.4.9　模拟使能寄存器 1 (PWR_ANAKEY1)

地址偏移：0x28
复位值：0x0000 0000
该寄存器读取时，返回值一直是 0。
用户在写模拟控制模块的寄存器时，必须先写 ANAKEY1 为 0x3，ANAKEY2 为 0xC。

6.4.10　模拟使能寄存器 2 (PWR_ANAKEY2)

地址偏移：0x2C
复位值：0x0000 0000
该寄存器读取时，返回值一直是 0。
用户在写模拟控制模块的寄存器时，必须先写 ANAKEY1 为 0x3，ANAKEY2 为 0xC。

第七章　备份寄存器 (BKP)

7.1　BKP 简介

备份寄存器是 21 个 32 位的寄存器，可用来存储 84 个字节的用户应用程序数据。他们处在备份域里，当 VDD 电源被切断，他们仍然由 V_{BAT} 维持供电。当系统在待机模式下被唤醒，或系统复位或电源复位时，他们也不会被复位。

此外，BKP 控制寄存器用来管理侵入检测和 RTC 校准功能。

复位后，对备份寄存器和 RTC 的访问被禁止，并且备份域被保护以防止可能存在的意外的写操作。执行以下操作可以使能对备份寄存器和 RTC 的访问。

- 通过设置 RCC 寄存器 AHBENR2 的 BDI 位来打开电源和后备接口的时钟
- PWR 电源控制寄存器 CR0 的 DBP 位来使能对后备寄存器和 RTC 的访问。

7.2　BKP 特性

BKP 的主要特性如下：

- 84 字节数据后备寄存器
- 用来管理防侵入检测并具有中断功能的状态/控制寄存器
- 用来存储 RTC 校验值的校验寄存器
- 在 PC13 引脚 (当该引脚不用于侵入检测时) 上输出 RTC 校准时钟，RTC 闹钟脉冲或者秒脉冲

7.3　BKP 功能描述

7.3.1　侵入检测

当 TAMPER 引脚上的信号从’0’变成’1’或者从’1’变成’0’(取决于备份控制寄存器 BKP_CR 的 TPAL 位)，会产生一个侵入检测事件。侵入检测事件将所有数据备份寄存器内容清除。然而为了避免丢失侵入事件，侵入检测信号是边沿检测的信号与侵入检测允许位的逻辑与，从而在侵入检测引脚被允许前发生的侵入事件也可以被检测到。

- 当 TPAL=0 时：如果在启动侵入检测 TAMPER 引脚前 (通过设置 TPE 位) 该引脚已经为高电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件 (尽管在 TPE 位置’1’后并没有出现上升沿)。

- 当 TPAL=1 时：如果在启动侵入检测引脚 TAMPER 前 (通过设置 TPE 位) 该引脚已经为低电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件 (尽管在 TPE 位置’1’后并没有出现下降沿)。

设置 BKP_CSR 寄存器的 TPIE 位为’1’，当检测到侵入事件时就会产生一个中断。

在一个侵入事件被检测到并被清除后，侵入检测引脚 TAMPER 应该被禁止。然后，在再次写入备份数据寄存器前重新用 TPE 位启动侵入检测功能。这样，可以阻止软件在侵入检测引脚上仍然有侵入事件时对备份数据寄存器进行写操作。这相当于对侵入引脚 TAMPER 进行电平检测。

注：当 *VDD* 电源断开时，侵入检测功能仍然有效。为了避免不必要的复位数据备份寄存器，*TAMPER* 引脚应该在片外连接到正确的电平。

7.3.2　RTC 校准

为方便测量，RTC 时钟可以经 64 分频输出到侵入检测引脚 TAMPER 上。通过设置 RTC 校验寄存器 (BKP_RTCCR) 的 CCO 位来开启这一功能。

通过配置 CAL[6:0] 位，此时钟可以最多减慢 121ppm。

7.4　BKP 寄存器

7.4.1　RTC 时钟校准寄存器 (BKP_RTCCR)

地址偏移量：0x00

复位值：0x0000 0000

表 7.1: RTC 时钟校准寄存器 (BKP_RTCCR) 位描述

位	符号	访问	描述
31:10	-	R	保留
9	ASOS	RW	闹钟或秒输出选择 (Alarm or second output selection) 当设置了 ASOE 位，ASOS 位可用于选择在 TAMPER 引脚上输出的是 RTC 秒脉冲还是闹钟脉冲信号。 0：输出 RTC 闹钟脉冲 1：输出秒脉冲 注：该位只能被后备区的复位所清除
8	ASOE	RW	允许输出闹钟或秒脉冲 (Alarm or second output enable) 根据 ASOS 位的设置，该位允许 RTC 闹钟或秒脉冲输出到 TAMPER 引脚上。 输出脉冲的宽度为一个 RTC 时钟的周期。设置了 ASOE 位时不能开启 TAMPER 的功能。 注：该位只能被后备区的复位所清除
7	CCO	RW	校准时钟输出 (Calibration clock output) 0：无影响 1：此位置 1 可以在侵入检测引脚输出经 64 分频后的 RTC 时钟。当 CCO 位置 1 时，必须关闭侵入检测功能以避免检测到无用的侵入信号。 注：当 *VDD* 供电断开时，该位被清除。

6:0	CAL	RW	校准值 (Calibration value) 校准值表示在每 2^{20} 个时钟脉冲内将有多少个时钟脉冲被跳过。这可以用来对 RTC 进行校准，以 $1000000/2^{20}$ ppm 的比例减慢时钟。 RTC 时钟可以被减慢 0~121ppm

7.4.2　备份控制寄存器 (BKP_CR)

地址偏移量：0x04

复位值：0x0000 0000

表 7.2: 备份控制寄存器 (BKP_CR) 位描述

位	符号	访问	描述
31:2	-	R	保留
1	TPAL	RW	侵入检测 TAMPER 引脚有效电平 (TAMPER pin active level) 0：侵入检测 TAMPER 引脚上的高电平会清除所有数据备份寄存器 (如果 TPE 位为 1) 1：侵入检测 TAMPER 引脚上的低电平会清除所有数据备份寄存器 (如果 TPE 位为 1)
0	TPE	RW	启动侵入检测 TAMPER 引脚 (TAMPER pin enable) 0：侵入检测 TAMPER 引脚作为通用 IO 口使用 1：开启侵入检测引脚作为侵入检测使用

注：同时设置 *TPAL* 和 *TPE* 位总是安全的。然而，同时清除两者会产生一个假的侵入事件。因此，推荐只在 *TPE* 为 *0* 时才改变 *TPAL* 位的状态。

7.4.3　备份控制/状态寄存器 (BKP_CSR)

地址偏移量：0x08

复位值：0x0000 0000

表 7.3: 备份控制/状态寄存器 (BKP_CSR) 位描述

位	符号	访问	描述
31:10	-	R	保留
9	TIF	R	侵入中断标志 (Tamper interrupt flag) 当检测到有侵入事件且 TPIE 位为 1 时，此位由硬件置 1。通过向 CTI 位写 1 来清除此标志位 (同时也清除了中断)。如果 TPIE 位被清除，则此位也会被清除。 0：无侵入中断 1：产生侵入中断 注：仅当系统复位或由待机模式唤醒后才复位该位

8	TEF	R	侵入事件标志 (Tamper event flag) 当检测到侵入事件时此位由硬件置 1。通过向 CTE 位写 1 可清除此标志位 0：无侵入事件 1：产生侵入事件 注：侵入事件会复位所有的 *BKP_DRx* 寄存器。只要 *TEF* 为 *1*，所有的 *BKP_DRx* 寄存器就一直保持复位状态。当此位被置 *1* 时，若对 *BKP_DRx* 进行写操作，写入的值不会被保存。
7:3	-	R	保留
2	TPIE	RW	允许侵入 TAMPER 引脚中断 (TAMPER pin interrupt enable) 0：禁止侵入检测中断 1：允许侵入检测中断 (BKP_CR 寄存器的 TPE 位也必须被置 1) 注：侵入中断无法将系统内核从低功耗模式唤醒；仅当系统复位或由待机模式唤醒后才复位该位。
1	CTI	RW	清除侵入检测中断 (Clear tamper interrupt) 此位只能写入，读出值为 0。 0：无效 1：清除侵入检测中断和 TIF 侵入检测中断标志
0	CTE	RW	清除侵入检测事件 (Clear tamper event) 此位只能写入，读出值为 0。 0：无效 1：清除 TEF 侵入检测事件标志 (并复位侵入检测器)。

7.4.4　备份数据寄存器 x(BKP_DRx) (x = 1 …21)

地址偏移量：0x10 到 0x60
复位值：0x0000 0000

表 7.4: 备份数据寄存器 x(BKP_DRx) (x = 1 …21) 位描述

位	符号	访问	描述
31:0	D	RW	备份数据 这些位可以被用来写入用户数据。

注：*BKP_DRx* 寄存器不会被系统复位、电源复位、从待机模式唤醒所复位。它们可以由备份域复位来复位或 *(*如果侵入检测引脚 *TAMPER* 功能被开启时*)* 由侵入引脚事件复位。

7.4.5　备份域控制寄存器 (BKP_BDCR)

地址偏移量：0x100
复位值：0x0000 0000

表 7.5: 备份域控制寄存器 (BKP_BDCR) 位描述

位	符号	访问	描述

31:16	-	R	保留
15	RTCEN	RW	RTC 时钟使能 (RTC clock enable) 由软件置’1’或清’0’ 0：RTC 时钟关闭； 1：RTC 时钟开启。
14:10	-	R	保留
9:8	RTCSEL	RW	RTC 时钟源选择 (RTC clock source selection) 由软件设置来选择 RTC 时钟源。一旦 RTC 时钟源被选定，直到下次后备域被复位，它不能在被改变。可通过设置 BDRST 位来清除。 00：无时钟； 01：LSE 振荡器作为 RTC 时钟； 10：LSI 振荡器作为 RTC 时钟； 11：HSE 振荡器在 128 分频后作为 RTC 时钟。
7:3	-	R	保留
2	LSEBYP	RW	外部低速时钟振荡器旁路 (External low-speed oscillator bypass) 在调试模式下由软件置’1’或清’0’来旁路 LSE。只有在外部 32kHz 振荡器关闭时，才能写入该位 0：LSE 时钟未被旁路； 1：LSE 时钟被旁路。
1	LSERDY	R	外部低速 LSE 就绪 (External low-speed oscillator ready) 由硬件置’1’或清’0’来指示是否外部 32kHz 振荡器就绪。在 LSEON 被清零后，该位需要 6 个外部低速振荡器的周期才被清零 0：外部 32kHz 振荡器未就绪； 1：外部 32kHz 振荡器就绪。
0	LSEON	RW	外部低速振荡器使能 (External low-speed oscillator enable) 由软件置’1’或清’0’ 0：外部 32kHz 振荡器关闭； 1：外部 32kHz 振荡器开启。

第八章　复位与时钟控制器 (RCC)

8.1　复位

8.1.1　系统复位

除了备份域的寄存器和 RCC 复位标志寄存器 (RCC_RSTSTAT) 以外，系统复位会将其余全部寄存器复位。只要发生以下事件之一，就会产生系统复位：

- NRST 引脚低电平 (外部复位)
- 窗口看门狗计数结束 (WWDG 复位)
- 独立看门狗计数结束 (IWDG 复位)
- 软件复位 (SW 复位)
- 低功耗复位 (PWR 复位)

软件复位：

将 Cortex-M3 应用中断和复位控制寄存器中的 SYSRESETREQ 置为 1 即可实现软件复位。

低功耗复位：

当系统退出待机模式 (Standby Mode) 时会产生一个低功耗复位。

外部复位时，当 NRST 引脚处于低电平时芯片产生复位脉冲。

内部复位时，内部复位源均作用于 NRST 引脚，芯片会向 NRST 引脚输出一个至少 20us 的低电平脉冲。

8.1.2　电源复位

只要发生以下事件之一，就会产生电源复位：

- 上电复位 (POR 复位)
- 掉电复位 (PDR 复位)

除备份域内的寄存器以外，电源复位会将其它全部寄存器设置为复位值。

8.1.3　备份域复位

备份域复位会将所有 RTC 核心寄存器/计数器和备份域寄存器复位为各自的复位值。
只要发生以下事件之一，就会产生备份域复位：

- 软件将 RCC 寄存器 RCC_BDRSTR 设置为 1.
- 在电源 VDD 和 VBAT 都已掉电后，其中任何之一又再上电。

8.2　时钟

8.2.1　系统时钟源

可使用四个不同的时钟源来驱动系统时钟：

- MHSI(8MHz) 内部振荡器时钟
- FHSI(48MHz) 内部振荡器时钟
- PLL 时钟
- HSE 外部振荡器时钟

8.2.2　次级时钟源

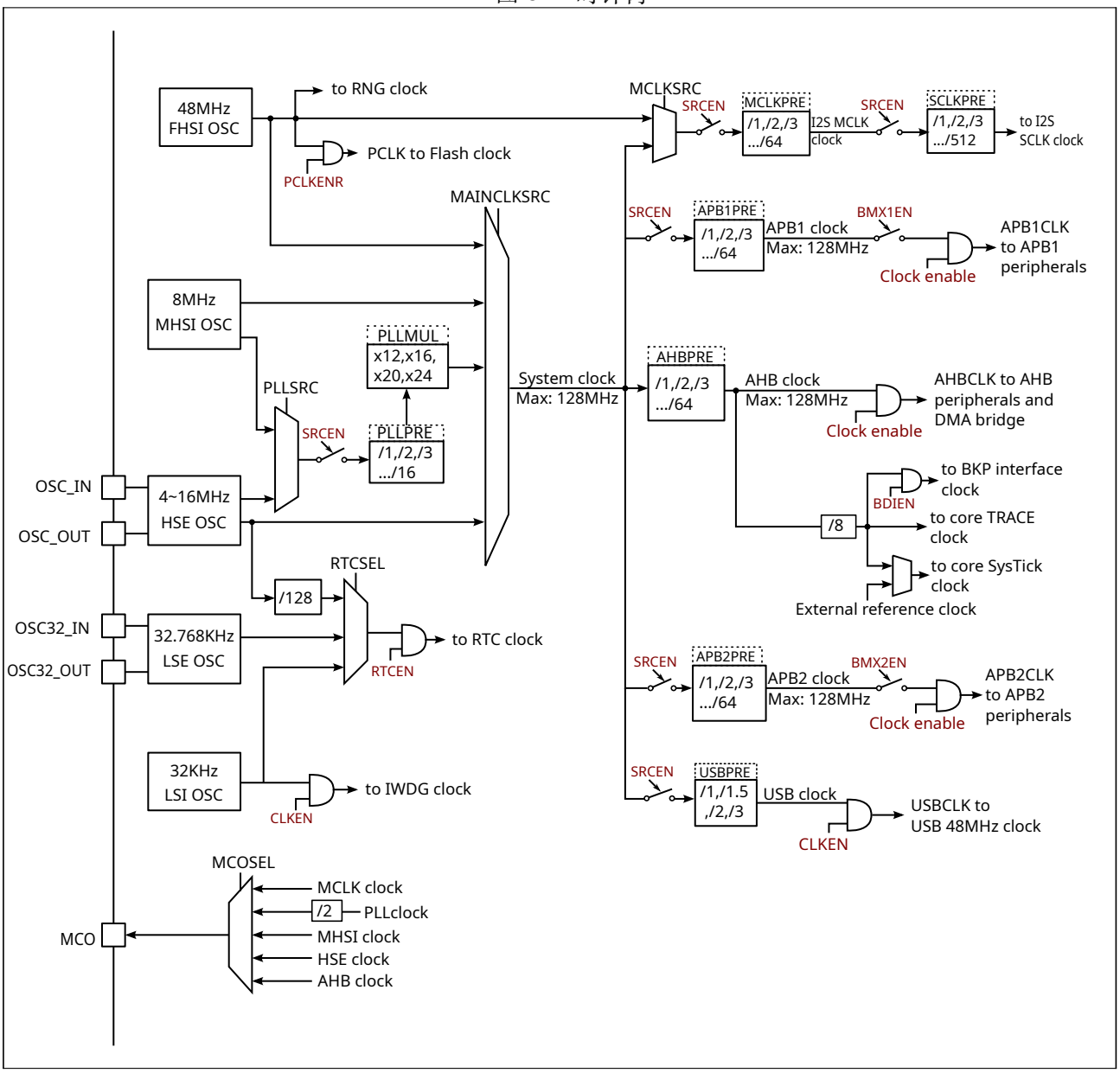
器件具有以下两个次级时钟源:

- LSI(32KHz) 内部低速振荡器时钟，用于驱动独立看门狗 (IWDG)
- LSE(32.768KHz) 外部低速振荡器时钟，用于驱动 RTC 时钟

8.2.3　时钟树

芯片的时钟树如下所示：

图 8.1: 时钟树



系统主时钟 MAINCLK 作为 AHB 总线，APB 总线，USB 和 I2S 的时钟源。Cortex 系统定时器 SysTick 和 TRACE 使用 AHB 时钟的 8 分频作为时钟源。

8.2.4　HSE 时钟

高速外部时钟信号 (HSE) 有两个时钟源：

1. 外部晶振/陶瓷谐振器

谐振器和负载电容必须尽可能地靠近振荡器的引脚，以尽量减小输出失真和起振稳定时间。
负载电容值必须根据所选振荡器的不同做适当调整。

2. 外部用户时钟

外部时钟源必须使用占空比为 50% 的外部时钟信号来驱动 OSC_IN 引脚，同时 OSC_OUT 引脚应保持高阻状态。

要使用 HSE 时钟，必须先将 GPIOD 的端口 0 和 1 设为模拟模式 (Analog Mode)，然后设置模拟控制单元 (ANCTL) 中的 HSE 控制寄存器。请参考 GPIO 和 ANCTL 章节内容。

注：必须等到模拟控制单元 (ANCTL) 中的状态寄存器 HSESR 为 1，才表示 HSE 时钟已经稳定，可以使用。

8.2.5　HSI 时钟

高速内部时钟信号 (HSI) 有两个时钟源：

- 8MHz MHSI 振荡器
- 48MHz FHSI 振荡器

HSI 时钟信号可以直接用作系统时钟，或者用做 PLL 输入信号。
用户可以通过模拟控制单元 (ANCTL) 中的 MHSI 使能控制寄存器 (MHSIENR) 来开关 MHSI 时钟。
用户可以通过模拟控制单元 (ANCTL) 中的 FHSI 使能控制寄存器 (FHSIENR) 来开关 FHSI 时钟。

注：必须等到模拟控制单元 (ANCTL) 中的状态寄存器 MHSISR 或 FHSISR 为 1，才表示 MHSI 或 FHSI 时钟已经稳定，可以使用。

8.2.6　PLL 时钟

PLL 用于产生系统时钟，最高可以产生 128MHz 时钟频率。
PLL 的输入来源可以为外部高速时钟 (HSE) 或者内部 8MHz 振荡器 (MHSI, 8MHz)。
PLL 的输入时钟源在进入 PLL 之前可以做预分频，参考寄存器 RCC_PLLPRE。
PLL 的参数设定请参考模拟控制单元 (ANCTL) 中的 PLL 寄存器。

8.2.7　LSE 时钟

LSE 时钟由 32.768KHz 外部时钟，低速外部晶振或陶瓷谐振器产生。可作为实时时钟 (RTC) 的时钟源来提供如日历等定时功能。

当外部时钟作为 LSE 时钟源时，必须使用占空比为 50% 的时钟信号来驱动 OSC32_IN 引脚，同时 OSC32_OUT 引脚应保持高阻状态。

8.2.8　LSI 时钟

LSI 时钟可作为低功耗时钟源在停机和待机模式下保持运行，供独立看门狗 (IWDG) 使用，频率为 32KHz。请参考模拟控制单元 (ANCTL) 中的 LSI 控制寄存器。

8.2.9　系统主时钟 MAINCLK 选择

系统复位后默认系统主时钟为 MHSI(8MHz)。

在直接使用 MHSI(8MHz) 或者通过 PLL 使用 MHSI(8MHz) 时，该时钟源无法停止。

只有在目标时钟源已稳定时，才可以切换时钟源，请参考 ANCTL 中的各时钟源状态寄存器来判断各时钟是否稳定。

8.2.10　时钟安全系统 CSS

时钟安全系统可通过软件激活，激活后，时钟监测器将在 HSE 稳定后开始工作，并在 HSE 停止时被关闭。

如果 HSE 时钟发生故障，HSE 振荡器将自动禁止，一个时钟故障事件将发送到高级定时器 TIM1，并且同时生成一个中断 (CSSI) 来通知 MCU。

CSSI 与 MCU 的 NMI(不可屏蔽中断) 相连接。

注意：当 CSS 使能后，如果 HSE 时钟发生故障，CSS 将生产一个中断给 NMI，应用程序必须在 NMI ISR 中将 CSS 中断清零。

当 HSE 作为系统时钟时，如果检测出故障，系统时钟将切换到 MHSI(8MHz) 振荡器，并且 HSE 振荡器将被禁止。

当 HSE 作为 PLL 时钟源并且 PLL 作为系统时钟时，如果检测出故障，PLL 也会被禁止。

8.2.11　RTC 时钟

RTC 时钟源可以选定为 HSE,LSE 或者 LSI，一旦选定便无法更改。要想修改时钟源，只能先进行备份域复位，然后修改。

当选定 HSE 作为 RTC 时钟源，HSE 会通过一个 128 倍分频单元连接到 RTC 上。

如果选定 LSE 作为 RTC 时钟，只要备份域电源 Vbat 正常，即使系统电源 VDD 关闭，RTC 仍将正常工作。

如果选定 HSE 或 LSI 作为 RTC 时钟，则系统电源 VDD 掉电时将无法保证 RTC 正常工作。

8.2.12　看门狗时钟

如果独立看门狗 (IWDG) 已经启动，则 LSI 振荡器将强制打开并且无法关闭，在 LSI 振荡器稳定后将提供时钟给 IWDG。

8.2.13　时钟输出功能

用户可以通过配置寄存器向 MCO 引脚 (PA8) 输出五个不同的时钟：

- PLL 输出时钟
- MHSI 时钟
- HSE 时钟
- I2S MCLK 时钟
- AHB 总线时钟

对于 MCO 引脚输出时钟，必须将 GPIOA 端口 8 设置为复用功能模式 (Alternate Function Mode)。

8.2.14　总线时钟开关与分频

AHB 总线时钟频率可以从系统时钟分频得到，分频比为 1 到 64。
APB 总线时钟可以控制其开启或关闭。
APB 总线时钟频率可以从系统时钟分频得到，分频比为 1 到 64。
APB 总线上各功能模块的时钟均可以单独控制其开启或关闭。
具体请参考 RCC 寄存器。

8.3　RCC 寄存器

8.3.1　PLL 预分频控制寄存器 (RCC_PLLPRE)

地址偏移：0x000
复位值：0x0000_0000
描述：可以对 PLL 的输入时钟进行预分频

位	符号	访问	描述
31:6	-	R	保留
5	SRCEN	RW	PLL 预分频器输入时钟使能 0: 禁止 PLL 预分频器输入时钟 1: 允许 PLL 预分频器输入时钟
4:1	RATIO	RW	分频比 0000: 2 分频 0001: 3 分频 ..... 1110: 16 分频
0	DIVEN	RW	PLL 预分频器使能 (Divider enable) 0：关闭 PLL 预分频器，关闭预分频器即表示输入时钟直接连到 PLL，不做预分频处理 1：开启 PLL 预分频器

8.3.2　PLL 时钟源选择寄存器 (RCC_PLLSRC)

地址偏移：0x004
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	SRC	RW	PLL 时钟源选择 0: MHSI(8MHz) 作为 PLL 输入时钟 1: HSE 作为 PLL 输入时钟

8.3.3　主时钟源选择寄存器 (RCC_MAINCLKSRC)

地址偏移：0x008

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	R	保留
1:0	SRC	RW	主时钟源选择 00: MHSI(8MHz) 作为系统主时钟 01: FHSI(48MHz) 作为系统主时钟 10: PLL 作为系统主时钟 11: HSE 作为系统主时钟

8.3.4　主时钟源更新使能寄存器 (RCC_MAINCLKUEN)

地址偏移：0x00C

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	ENA	W	更新主时钟源 对此位写 1 将使系统主时钟切换到用户选择的时钟源，系统会自动清除此位，无需用户对此位清零

8.3.5　USB 时钟预分频控制寄存器 (RCC_USBPRE)

地址偏移：0x014

复位值：0x0000_0000

描述：可以对 USB 输入时钟进行预分频

位	符号	访问	描述
31:4	-	R	保留
3	SRCEN	RW	USB 预分频器输入时钟使能 0: 禁止 USB 预分频器输入时钟 1: 允许 USB 预分频器输入时钟
2:1	RATIO	RW	分频比 00: 2 分频 10: 1.5 分频 others: 3 分频
0	DIVEN	RW	USB 预分频器使能 (Divider enable) 0：关闭 USB 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：开启 USB 预分频器

8.3.6　AHB 时钟预分频寄存器 (RCC_AHBPRE)

地址偏移：0x018

复位值：0x0000_0000

描述：可以对 AHB 总线输入时钟进行预分频

位	符号	访问	描述
31:7	-	R	保留
6:1	RATIO	RW	分频比 0x00: 2 分频 0x01: 3 分频 ..... 0x3E: 64 分频
0	DIVEN	RW	AHB 预分频器使能 (Divider enable) 0：关闭 AHB 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：开启 AHB 预分频器

8.3.7　APB1 时钟预分频控制寄存器 (RCC_APB1PRE)

地址偏移：0x01C

复位值：0x0000_0000

描述：可以对 APB1 总线输入时钟进行预分频

位	符号	访问	描述
31:8	-	R	保留
7	SRCEN	RW	APB1 预分频器输入时钟使能 0: 禁止 APB1 预分频器输入时钟 1: 允许 APB1 预分频器输入时钟
6:1	RATIO	RW	分频比 0x00: 2 分频 0x01: 3 分频 ..... 0x3E: 64 分频
0	DIVEN	RW	APB1 预分频器使能 (Divider enable) 0：关闭 APB1 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：开启 APB1 预分频器

8.3.8　APB2 时钟预分频控制寄存器 (RCC_APB2PRE)

地址偏移：0x020

复位值：0x0000_0000

描述：可以对 APB2 总线输入时钟进行预分频

位	符号	访问	描述
31:8	-	R	保留

7	SRCEN	RW	APB2 预分频器输入时钟使能 0: 禁止 APB2 预分频器输入时钟 1: 允许 APB2 预分频器输入时钟
6:1	RATIO	RW	分频比 0x00: 2 分频 0x01: 3 分频 ..... 0x3E: 64 分频
0	DIVEN	RW	APB2 预分频器使能 (Divider enable) 0：关闭 APB2 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：开启 APB2 预分频器

8.3.9　I2S MCLK 时钟预分频控制寄存器 (RCC_MCLKPRE)

地址偏移：0x024

复位值：0x0000_0000

位	符号	访问	描述
31:8	-	R	保留
7	SRCEN	RW	I2S MCLK 预分频器输入时钟使能 0: 禁止 MCLK 预分频器输入时钟 1: 允许 MCLK 预分频器输入时钟
6:1	RATIO	RW	分频比 0x00: 2 分频 0x01: 3 分频 ..... 0x3E: 64 分频
0	DIVEN	RW	I2S MCLK 预分频器使能 (Divider enable) 0：关闭 MCLK 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：打开 MCLK 预分频器

8.3.10　I2S SCLK 时钟预分频控制寄存器 (RCC_I2SPRE)

地址偏移：0x028

复位值：0x0000_0000

描述：可以对 I2S SCLK 输入时钟进行预分频

位	符号	访问	描述
31:11	-	R	保留
10	SRCEN	RW	I2S SCLK 预分频器输入时钟使能 0: 禁止 SCLK 预分频器输入时钟 1: 允许 SCLK 预分频器输入时钟

9:1	RATIO	RW	分频比 0x00: 2 分频 0x01: 3 分频 ..... 0x1FE: 512 分频
0	DIVEN	RW	I2S SCLK 预分频器使能 (Divider enable) 0：关闭 SCLK 预分频器，关闭预分频器即表示直接使用输入时钟，不做分频处理 1：打开 SCLK 预分频器

8.3.11　I2S MCLK 时钟源选择寄存器 (RCC_MCLKSRC)

地址偏移：0x02C

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	SRC	RW	I2S MCLK 时钟源选择 0: 系统主时钟 (MAINCLK) 作为 I2S MCLK 时钟源 1: FHSI(48MHz) 时钟作为 I2S MCLK 时钟源

8.3.12　USB 缓存时钟源选择寄存器 (RCC_USBFIFOCLKSRC)

地址偏移：0x034

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	SRC	RW	USB 缓存时钟源选择 0: AHB 总线时钟作为 USB 缓存时钟 1: USB 时钟作为 USB 缓存时钟

8.3.13　引脚输出时钟选择寄存器 (RCC_MCOSEL)

地址偏移：0x038

复位值：0x0000_0000

描述：选择一个时钟输出到 MCO 引脚 (PA8)，一次只能有一个控制位为 1。

改变输出时钟的时候，必须先禁止所有时钟输出 (本寄存器写为全 0)，然后再选择其中之一来输出 (某一个位写为 1)

位	符号	访问	描述
31:5	-	R	保留
4	MCLK	RW	I2S MCLK 输出到 MCO 引脚
3	PLLDIV2	RW	PLL 时钟经过一个 2 分频器输出到 MCO 引脚
2	MHSI	RW	MHSI(8MHz) 输出到 MCO 引脚

1	HSE	RW	HSE 时钟输出到 MCO 引脚
0	AHBCLK	RW	AHB 时钟输出到 MCO 引脚

8.3.14　AHB 设备的总线时钟开关寄存器 0(RCC_AHBENR0)

地址偏移：0x03C

复位值：0x0000_007B

描述：AHB 总线上的各设备总线时钟开关，控制位为 1 即开启对应设备的总线时钟，控制位为 0 即关闭对应设备的总线时钟

位	符号	访问	描述
31:3	-	R	保留
2	IWDGEN	RW	独立看门狗时钟使能
1:0	-	R	保留

8.3.15　AHB 设备的总线时钟开关寄存器 1(RCC_AHBENR1)

地址偏移：0x040

复位值：0x0000_003D

描述：AHB 总线上的各设备总线时钟开关，控制位为 1 即开启对应设备的总线时钟，控制位为 0 即关闭对应设备的总线时钟

位	符号	访问	描述
31:9	-	R	保留
8	CRCSFMEN	RW	CRC 和 SFM 模块时钟使能
7	DMAC2BREN	RW	DMAC2 与 APB2 的连接桥时钟使能
6	DMAC1BREN	RW	DMAC1 与 APB1 的连接桥时钟使能
5	SYSEN	RW	SYS 控制器时钟使能
4	CACHEEN	RW	CACHE 控制器时钟使能
3	FLASHEN	RW	FLASH 存储控制器时钟使能
2	ISOEN	RW	ISO7816 控制器时钟使能
1	USBEN	RW	USB 控制器时钟使能
0	-	R	保留

8.3.16　AHB 设备的总线时钟开关寄存器 2(RCC_AHBENR2)

地址偏移：0x044

复位值：0x0000_0003

描述：AHB 总线上的各设备总线时钟开关，控制位为 1 即开启对应设备的总线时钟，控制位为 0 即关闭对应设备的总线时钟

位	符号	访问	描述
31:3	-	R	保留

2	BDIEN	RW	备份域接口时钟使能
1:0	-	R	保留

8.3.17　APB1 设备的总线时钟开关寄存器 (RCC_APB1ENR)

地址偏移：0x048

复位值：0x0000_0000

描述：APB1 总线上的各设备总线时钟开关，控制位为 1 即开启对应设备的总线时钟，控制位为 0 即关闭对应设备的总线时钟

位	符号	访问	描述
31:16	-	R	保留
15	BMX1EN	RW	APB1 总线控制器时钟使能
14	UART1EN	RW	UART1 控制器时钟使能
13	SPIS1EN	RW	SPIS1 控制器时钟使能
12	QSPIEN	RW	QSPI 控制器时钟使能
11	ADCEN	RW	ADC 控制器时钟使能
10	AFIOEN	RW	AFIO 控制器时钟使能
9	EXTIEN	RW	EXTI 控制器时钟使能
8	GPIODEN	RW	GPIOD 控制器时钟使能
7	GPIOCEN	RW	GPIOC 控制器时钟使能
6	GPIOBEN	RW	GPIOB 控制器时钟使能
5	GPIOAEN	RW	GPIOA 控制器时钟使能
4	TIM4EN	RW	TIMER4 控制器时钟使能
3	TIM3EN	RW	TIMER3 控制器时钟使能
2	TIM2EN	RW	TIMER2 控制器时钟使能
1	TIM1EN	RW	TIMER1 控制器时钟使能
0	DMAC1EN	RW	DMAC1 控制器时钟使能

注：在 *APB1* 相关模块操作时，*BMX1EN* 必须配置成 *1*。

8.3.18　APB2 设备的总线时钟开关寄存器 (RCC_APB2ENR)

地址偏移：0x04C

复位值：0x0000_0000

描述：APB2 总线上的各设备总线时钟开关，控制位为 1 即开启对应设备的总线时钟，控制位为 0 即关闭对应设备的总线时钟

位	符号	访问	描述
31:12	-	R	保留
11	BMX2EN	RW	APB2 总线控制器时钟使能
10	LEDEN	RW	LED 控制器时钟使能
9	RNGEN	RW	RNG 控制器时钟使能

8	I2C2EN	RW	I2C2 控制器时钟使能
7	I2C1EN	RW	I2C1 控制器时钟使能
6	I2SEN	RW	I2S 控制器时钟使能
5	SPIS2EN	RW	SPIS2 控制器时钟使能
4	SPIM2EN	RW	SPIM2 控制器时钟使能
3	UART3EN	RW	UART3 控制器时钟使能
2	UART2EN	RW	UART2 控制器时钟使能
1	WWDGEN	RW	窗口看门狗时钟使能
0	DMAC2EN	RW	DMAC2 控制器时钟使能

注：在 *APB2* 相关模块操作时，*BMX2EN* 必须配置成 *1*。

8.3.19　随机数发生器时钟开关寄存器 (RCC_RNGCLKENR)

地址偏移：0x05C
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	随机数发生器时钟开关 0：关闭随机数发生器时钟 1：开启随机数发生器时钟

8.3.20　独立看门狗时钟开关寄存器 (RCC_IWDGCLKENR)

地址偏移：0x064
复位值：0x0000_0001

位	符号	访问	描述
31:3	-	R	保留
2	DCSSCLKEN	RW	DCSS 时钟开关 0：关闭 DCSS 时钟 1：开启 DCSS 时钟
1	-	R	保留
0	IWDGCLKEN	RW	独立看门狗时钟开关 0：关闭独立看门狗时钟 1：开启独立看门狗时钟 注：当独立看门狗启动时，这个时钟开关变为无效。

8.3.21　USB48MHz 时钟开关寄存器 (RCC_USBCLKENR)

地址偏移：0x06C
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	USB48MHz 时钟开关 0：关闭 USB48MHz 时钟 1：开启 USB48MHz 时钟

8.3.22　I2S SCLK 时钟开关寄存器 (RCC_I2SCLKENR)

地址偏移：0x070

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	I2S SCLK 时钟开关 0：关闭 I2S SCLK 时钟 1：开启 I2S SCLK 时钟

8.3.23　SPIS1 时钟开关寄存器 (RCC_SPIS1CLKENR)

地址偏移：0x074

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	SPIS1 时钟开关 0：关闭 SPIS1 时钟 1：开启 SPIS1 时钟

8.3.24　SPIS2 时钟开关寄存器 (RCC_SPIS2CLKENR)

地址偏移：0x078

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	SPIS2 时钟开关 0：关闭 SPIS2 时钟 1：开启 SPIS2 时钟

8.3.25　USB 缓存时钟开关寄存器 (RCC_USBFIFOCLKENR)

地址偏移：0x07C

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	USB 缓存时钟开关 0：关闭 USB 缓存时钟 1：开启 USB 缓存时钟

8.3.26　AHB 设备的复位寄存器 1(RCC_AHBRSTR1)

地址偏移：0x088

复位值：0x0000_0000

描述：AHB 总线上的各设备复位，控制位写 1 即开启对应设备的复位，控制位写 0 即结束对应设备的复位。

位	符号	访问	描述
31:9	-	R	保留
8	CRCSFMRST	RW	CRC 和 SFM 模块复位
7:6	-	R	保留
5	SYSRST	RW	SYS 控制器复位
4	CACHERST	RW	CACHE 控制器复位
3	FLASHRST	RW	FLASH 存储控制器复位
2	ISORST	RW	ISO7816 控制器复位
1	USBRST	RW	USB 控制器复位
0	-	R	保留

8.3.27　APB1 设备的复位寄存器 (RCC_APB1RSTR)

地址偏移：0x090

复位值：0x0000_0000

描述：APB1 总线上的各设备复位，控制位写 1 即开启对应设备的复位，控制位写 0 即结束对应设备的复位。

位	符号	访问	描述
31:16	-	R	保留
15	BMX1RST	RW	APB1 总线控制器复位
14	UART1RST	RW	UART1 控制器复位
13	SPIS1RST	RW	SPIS1 控制器复位
12	QSPIRST	RW	QSPI 控制器复位
11	ADCRST	RW	ADC 控制器复位
10	AFIORST	RW	AFIO 控制器复位
9	EXTIRST	RW	EXTI 控制器复位
8	GPIODRST	RW	GPIOD 控制器复位
7	GPIOCRST	RW	GPIOC 控制器复位

6	GPIOBRST	RW	GPIOB 控制器复位
5	GPIOARST	RW	GPIOA 控制器复位
4	TIM4RST	RW	TIMER4 控制器复位
3	TIM3RST	RW	TIMER3 控制器复位
2	TIM2RST	RW	TIMER2 控制器复位
1	TIM1RST	RW	TIMER1 控制器复位
0	DMAC1RST	RW	DMAC1 控制器复位

8.3.28　APB2 设备的复位寄存器 (RCC_APB2RSTR)

地址偏移：0x094

复位值：0x0000_0000

描述：APB2 总线上的各设备复位，控制位写 1 即开启对应设备的复位，控制位写 0 即结束对应设备的复位。

位	符号	访问	描述
31:12	-	R	保留
11	BMX2RST	RW	APB2 总线控制器复位
10	LEDRST	RW	LED 控制器复位
9	RNGRST	RW	RNG 控制器复位
8	I2C2RST	RW	I2C2 控制器复位
7	I2C1RST	RW	I2C1 控制器复位
6	I2SRST	RW	I2S 控制器复位
5	SPIS2RST	RW	SPIS2 控制器复位
4	SPIM2RST	RW	SPIM2 控制器复位
3	UART3RST	RW	UART3 控制器复位
2	UART2RST	RW	UART2 控制器复位
1	WWDGRST	RW	窗口看门狗复位
0	DMAC2RST	RW	DMAC2 控制器复位

8.3.29　I2S SCLK 复位寄存器 (RCC_I2SCLKRSTR)

地址偏移：0x0B8

复位值：0x0000_0000

描述：写 1 即对 I2S SCLK 时钟域复位，写 0 即结束 I2S SCLK 时钟域的复位。

位	符号	访问	描述
31:1	-	R	保留
0	SCLKRST	RW	I2S SCLK 时钟域复位

8.3.30　复位标志清除寄存器 (RCC_CLRRSTSTAT)

地址偏移：0x0C8

复位值：0x0000_0000

描述：写 1 即清除 RCC_RSTSTAT 寄存器，系统会自动将此位清零

位	符号	访问	描述
31:1	-	R	保留
0	CLR	W	清除 RCC_RSTSTAT 寄存器的内容

8.3.31　备份域复位寄存器 (RCC_BDRSTR)

地址偏移：0x0D4

复位值：0x0000_0000

描述：写 1 即复位备份域，写 0 即结束备份域的复位。

位	符号	访问	描述
31:1	-	R	保留
0	BDRST	RW	备份域复位

8.3.32　LSI 备份域时钟开关寄存器 (RCC_LSI2RTCENR)

地址偏移：0x0D8

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	备份域时钟开关 0: 禁止 LSI 作为备份域中 RTC 时钟 1: 允许 LSI 作为备份域中 RTC 时钟

8.3.33　HSE 时钟 128 倍分频开关寄存器 (RCC_HSE2RTCENR)

地址偏移：0x0DC

复位值：0x0000_0000

描述：若选择 HSE 作为 RTC 时钟，HSE 将通过一个 128 倍分频器，本寄存器用于控制这个分频器

位	符号	访问	描述
31:1	-	R	保留
0	DIVEN	RW	128 倍分频器开关 0: 关闭 128 倍分频器 1: 开启 128 倍分频器

8.3.34　复位标志寄存器 (RCC_RSTSTAT)

地址偏移：0x100

复位值：0x0000_0000

描述：本寄存器将记录发生过的复位，只能通过 RCC_CLRRSTSTAT 寄存器来清除

位	符号	访问	描述
31:6	-	R	保留
5	PINRSTF	R	在 NRST 引脚复位发生时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除
4	PORRSTF	R	在上电/掉电复位发生时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除
3	SFTRSTF	R	在软件复位发生时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除
2	IWDGRSTF	R	在独立看门狗复位发生在 VDD 区域时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除
1	WWDGRSTF	R	在窗口看门狗复位发生时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除
0	LPWRRSTF	R	在低功耗管理复位发生时由硬件置 1，由软件通过写 RCC_CLRRSTSTAT 清除

第九章　通用 I/O (GPIO) 和替换功能 I/O（AFIO）

9.1　简介

每个通用 I/O 端口包括 4 个 32 位配置寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_SMIT 和 GPIOx_PUPDR）、2 个 32 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR）和 1 个 32 位置位/复位寄存器 (GPIOx_BSRR)。此外，所有 GPIO 都包括 1 个 32 位锁定寄存器 (GPIOx_LCKR)、1 个 32 位辅助配置寄存器（GPIOx_CFGMSK）和 2 个 32 位复用功能选择寄存器（GPIOx_AFRH 和 GPIOx_AFRL）。

9.2　GPIO 主要特性

- 输出状态：推挽或开漏 + 上拉/下拉
- 从输出数据寄存器 (GPIOx_ODR) 或外设（复用功能输出）输出数据
- 可为每个 I/O 选择不同的速度
- 输入状态：浮空、上拉/下拉、模拟
- 将数据输入到输入数据寄存器 (GPIOx_IDR) 或外设（复用功能输入）
- 置位和复位寄存器 (GPIOx_BSRR)，对 GPIOx_ODR 具有按位写权限
- 锁定机制 (GPIOx_LCKR)，可冻结 I/O 端口配置
- 辅助配置机制 (GPIOx_CFGMSK)，可临时冻结 I/O 端口配置功能
- 复用功能选择寄存器
- 引脚复用非常灵活，允许将 I/O 引脚用作 GPIO 或多种外设功能中的一种

9.3　GPIO 功能描述

根据数据手册中列出的每个 I/O 端口的特性，可通过软件将通用 I/O (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
- 施密特触发器
- 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
- 具有上拉或下拉功能的复用功能开漏

每个 I/O 端口位均可自由编程，但 I/O 端口寄存器必须按 32 位字、半字或字节进行访问。GPIOx_BSRR 寄存器和 GPIOx_BRR 寄存器旨在实现对 GPIOx_ODR 寄存器进行原子读取/修改访问。这样便可确保在读取和修改访问之间发生中断请求也不会有问题。GPIOx_CFGMSK 寄存器辅助进行寄存器配置。

表 9.1: 端口位配置表

MODE[1:0]	OTYPER	OSPEED[1:0]		PUPD[1:0]		I/O 配置	
01	0	SPEED[1:0]		0	0	GP 输出	PP
	0			0	1	GP 输出	PP + PU
	0			1	0	GP 输出	PP + PD
	0			1	1	保留	
	1			0	0	GP 输出	OD
	1			0	1	GP 输出	OD + PU
	1			1	0	GP 输出	OD + PD
	1			1	1	保留 (GP 输出 OD)	
10	0	SPEED[1:0]		0	0	AF	PP
	0			0	1	AF	PP + PU
	0			1	0	AF	PP + PD
	0			1	1	保留	
	1			0	0	AF	OD
	1			0	1	AF	OD + PU
	1			1	0	AF	OD + PD
	1			1	1	保留	
00	x	x	x	0	0	输入	浮空
	x	x	x	0	1	输入	PU
	x	x	x	1	0	输入	PD
	x	x	x	1	1	保留 (输入浮空)	
11	x	x	x	0	0	输入输出	模拟
	x	x	x	0	1	保留	
	x	x	x	1	0		
	x	x	x	1	1		

9.3.1　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，大多数 I/O 端口被配置为输入浮空模式。复位后，下列引脚处于复用功能上拉/下拉状态：

- PA14：处于下拉状态
- PA13：处于上拉状态
- PA12：处于上拉状态
- PA11：处于下拉状态

当引脚配置为输出后，写入到输出数据寄存器 (GPIOx_ODR) 的值将在 I/O 引脚上输出。可以在推挽模式下或开漏模式下使用输出驱动器（仅驱动低电平，高电平为高阻态）。

输入数据寄存器 (GPIOx_IDR) 每隔 1 个 APB 时钟周期捕获一次 I/O 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。

9.3.2　I/O 引脚复用功能复用器和映射

器件 I/O 引脚通过一个复用器连接到板载外设/模块，该复用器一次仅允许一个外设的复用功能 (AF) 连接到 I/O 引脚。这可以确保共用同一个 I/O 引脚的外设之间不会发生冲突。每个 I/O 引脚都有一个复用器，该复用器采用多达 16 路复用功能输入（AF0 到 AF15），可通过 GPIOx_AFRL（针对引脚 0 到 7）和 GPIOx_AFRH（针对引脚 8 到 15）寄存器对这些输入进行配置：

- 复位后，复用器选择为复用功能 0 (AF0)。在复用模式下通过 GPIOx_MODER 寄存器配置 I/O。
- 器件数据手册中详细说明了每个引脚的特定复用功能分配。

除了这种灵活的 I/O 复用架构之外，各外设还可以将复用功能映射到不同 I/O 引脚，这可以优化小型封装中可用外设的数量。要将 I/O 配制成所需功能，必须按照以下步骤操作：

- 调试功能：每个器件复位后，立即将这些引脚分配为可由调试主机使用的复用功能引脚。
- 系统功能：必须将 MCOx 引脚配置为复用功能模式。
- GPIO ： 在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟通道。
- 外设复用功能：
  - 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需的 AFx。
  - 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDER 寄存器，分别选择类型、上拉/下拉以及输出速度。
  - 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能。
- 其它功能：
  - 对于 ADC，CMP 和 USB，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC，CMP 和 USB 寄存器中配置所需功能。
  - 对于 RTC_OUT、RTC_TS、RTC_TAMP、WKUP 和振荡器的其它功能，在相关的 RTC、PWR 和 BKP 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。

9.3.3　I/O 端口控制寄存器

每个 GPIO 端口有 4 个 32 位存储器映射的控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR），可配置多达 16 个 I/O。GPIOx_MODER 寄存器用于选择 I/O 模式（输入、输出、AF、模拟）。GPIOx_OTYPER 和 GPIOx_OSPEEDR 寄存器用于选择输出类型（推挽或开漏）和速度。无论采用哪种 I/O 方向，GPIOx_PUPDR 寄存器都用于选择上拉/下拉。

9.3.4　I/O 端口数据寄存器

每个 GPIO 都具有 2 个 16 位数据寄存器：输入和输出数据寄存器（GPIOx_IDR 和 GPIOx_ODR）。GPIOx_ODR 用于存储待输出数据，可对其进行读/写访问。通过 I/O 输入的数据存储到输入数据寄存器 (GPIOx_IDR) 中，它是一个只读寄存器。

9.3.5　I/O 数据位操作

置位复位寄存器 (GPIOx_BSRR) 是一个 32 位寄存器，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。置位复位寄存器的大小是 GPIOx_ODR 的二倍。

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) 位会复位 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 中同时尝试对某个位执行置位和复位操作，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：在一次原子写访问中，可以修改一个或多个位。

9.3.6　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_CURRENT、GPIOx_SMIT、GPIOx_AFRL 和 GPIOx_AFRH。

要对 GPIOx_LCKR 寄存器执行写操作，必须应用特定的写/读序列。当正确的 LOCK 序列应用到此寄存器的第 16 位后，会使用 LCKR[15:0] 的值来锁定 I/O 的配置（在写序列期间，LCKR[15:0] 的值必须相同）。将 LOCK 序列应用到某个端口位后，在执行下一次 MCU 复位或外设复位之前，将无法对该端口位的值进行修改。每个 GPIOx_LCKR 位都会冻结控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_CURRENT、GPIOx_SMIT、GPIOx_AFRL 和 GPIOx_AFRH）中的对应位。

LOCK 序列只能通过对 GPIOx_LCKR 寄存器进行字（32 位长）访问的方式来执行，因为 GPIOx_LCKR 的第 16 位必须与 [15:0] 位同时置位。

配置辅助寄存器 GPIOx_CFGMSK 功能类似于 GPIOx_LCKR 寄存器，可以用来快速配置寄存器的值，当 GPIOx_CFGMSK 特定的位被值 1 后，相应的 GPIO 寄存器位就不能进行写操作。配置辅助寄存器 GPIOx_CFGMSK 的读写没有任何限制。

9.3.7　I/O 复用功能输入/输出

有两个寄存器可用来从每个 I/O 可用的复用功能输入/输出中进行选择。借助这些寄存器，可根据应用程序的要求将某个复用功能连接到其它某个引脚。

这意味着可使用 GPIOx_AFRL 和 GPIOx_AFRH 复用功能寄存器在每个 GPIO 上复用多个可用的外设功能。这样一来，应用程序可为每个 I/O 选择任何一个可用功能。由于 AF 选择信号由复用功能输入和复用功能输出共用，所以只需为指定 I/O 的复用功能输入/输出选择一个通道即可。

使用 GPIOx_AFRL 和 GPIOx_AFRH 复用功能寄存器在每个 GPIO 上复用外设功能：

表 9.2:　WB32F10xxx 复用功能映射表

Port	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
	SYS	TIM1/2	TIM3/4	I2S	I2C / LED	SPI(M)	SPI(S)	UART
PA0	WKUP	TIM2_CH1_ETR						UART2_CTS
PA1		TIM2_CH2						UART2_RTS
PA2		TIM2_CH3						UART2_TX
PA3		TIM2_CH4						UART2_RX
PA4						QSPI_NSS0	SPIS1_NSS	UART2_CK
PA5						QSPI_SCK	SPIS1_SCK	
PA6		TIM1_BKIN	TIM3_CH1			QSPI_MI_IO1	SPIS1_SO	
PA7		TIM1_CH1N	TIM3_CH2			QSPI_MO_IO0	SPIS1_SI	
PA8	MCO	TIM1_CH1						UART1_CK
PA9		TIM1_CH2						UART1_TX
PA10		TIM1_CH3						UART1_RX
PA11		TIM1_CH4						UART1_CTS
PA12		TIM1_ETR						UART1_RTS
PA13	SWO_DIO					QSPI_NSS1		
PA14	SWO_CLK					QSPI_NSS2		
PA15		TIM2_CH1_ETR		I2S_WS	I2C_SMBAI	QSPI_NSS0	SPIS1_NSS	
PB0		TIM1_CH2N	TIM3_CH3	I2S_MCLK		QSPI_IO2		
PB1		TIM1_CH3N	TIM3_CH4			QSPI_IO3		
PB2	BOOT1							
PB3	SWO	TIM2_CH2		I2S_SCLK		QSPI_SCK	SPIS1_SCK	
PB4			TIM3_CH1			QSPI_MI_IO1	SPIS1_SO	
PB5			TIM3_CH2	I2S_SD1	I2C1_SMBAI	QSPI_MO_IO0	SPIS1_SI	
PB6			TIM4_CH1		I2C1_SCL	QSPI_NSS1		UART1_TX
PB7			TIM4_CH2		I2C1_SDA	SPIM2_NSS1		UART1_RX
PB8			TIM4_CH3		I2C1_SCL	SPIM2_NSS2		UART1_CTS
PB9			TIM4_CH4		I2C1_SDA			UART1_RTS
PB10		TIM2_CH3	TIM4_CH1		I2C2_SCL	QSPI_NSS2		UART3_TX
PB11		TIM2_CH4			I2C2_SDA	SPIM2_NSS1		UART3_RX
PB12		TIM1_BKIN		I2S_WS		SPIM2_NSS0	SPIS2_NSS	UART3_CK
PB13		TIM1_CH1N		I2S_SCLK		SPIM2_SCK	SPIS2_SCK	UART3_CTS
PB14		TIM1_CH2N				SPIM2_MI	SPIS2_SO	UART3_RTS
PB15		TIM1_CH3N		I2S_SD0		SPIM2_MO	SPIS2_SI	
PC0				I2S_WS		SPIM2_NSS0	SPIS2_NSS	
PC1				I2S_SCLK		SPIM2_SCK	SPIS2_SCK	
PC2				I2S_SD0		SPIM2_MI	SPIS2_SO	
PC3				I2S_SD1		SPIM2_M0	SPIS2_SI	
PC4	TRACECK							
PC5	TRACED0					SPIM2_NSS2		
PC6			TIM3_CH1	I2S_MCLK				
PC7			TIM3_CH2	I2S_MCLK				
PC8			TIM3_CH3					
PC9	TRACED1		TIM3_CH4					
PC10	TRACED2							UART3_TX

(continued)

Port	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
	SYS	TIM1/2	TIM3/4	I2S	I2C	SPI(M)	SPI(S)	UART
PC11	TRACED3							UART3_RX
PC12			TIM4_ETR					UART3_CK
PC13	TAMPER_RTC							
PC14	OSC32_IN							
PC15	OSC32_OUT							
PD0	OSC_IN							
PD1	OSC_OUT							
PD2			TIM3_ETR					

9.3.8　外部中断线/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。更多的关于外部中断的信息，参考：

- 第十一 章: 外部中断/事件控制器 (EXTI)

9.3.9　输入控制

对 I/O 端口进行编程作为输入时：

- 输出缓冲器被禁止
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 输入数据寄存器每隔 1 个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样
- 对输入数据寄存器的读访问可获取 I/O 状态

9.3.10　输出控制

对 I/O 端口进行编程作为输出时：

- 输出缓冲器被打开：
  - 开漏模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”会使端口保持高阻态 (Hi-Z)（P-MOS 始终不激活）
  - 推挽模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”可激活 P-MOS
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 输入数据寄存器每隔 1 个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样
- 对输入数据寄存器的读访问可获取 I/O 状态
- 对输出数据寄存器的读访问可获取最后的写入值

9.3.11　复用功能配置

对 I/O 端口进行编程作为复用功能时：

- 可将输出缓冲器配置为开漏或推挽模式
- 输出缓冲器由来自外设的信号驱动（发送器使能和数据）
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开弱上拉电阻和下拉电阻
- 输入数据寄存器每隔 1 个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样
- 对输入数据寄存器的读访问可获取 I/O 状态

9.3.12　模拟配置

对 I/O 端口进行编程作为模拟配置时：

- 输出缓冲器被禁止
- 施密特触发器输入停用，I/O 引脚的每个模拟输入的功耗变为零。施密特触发器的输出被强制处理为恒定值 (0)
- 弱上拉和下拉电阻被硬件关闭
- 对输入数据寄存器的读访问值为“0”

9.3.13　将 HSE 或 LSE 振荡器引脚用作 GPIO

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

当 HSE 或 LSE 振荡器打开（通过设置 ANCTL 和 BKP 中相应寄存器中的位），并且这些引脚的 GPIO 配置成模拟功能，振荡器会控制与其相关联的引脚。

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

9.3.14　在备份电源域中使用 GPIO 引脚

当内核电源域掉电时（器件进入待机模式时），PC13/PC14/PC15 GPIO 功能会丢失。在这种情况下，如果不通过 RTC 配置旁路其 GPIO 配置，这些引脚将被设置为输入模式。

9.4　GPIO 寄存器

9.4.1　GPIO 端口模式寄存器（GPIOx_MODER）

地址偏移量：0x00

复位值：

- 0x2A80 0000 (端口 A)
- 0x0000 0000 (其它端口)

表 9.3: GPIO 端口模式寄存器（GPIOx_MODER）（x=A，B，C，D）位描述

位	符号	访问	描述
2y+1:2y (y=0..15)	MODERy[1:0]	RW	端口 x 配置位 (Port x configuration bits) 这些位通过软件写入，用于配置 I/O 模式。 00：输入模式（复位状态） 01：通用输出模式 10：复用功能模式 11：模拟模式

9.4.2　GPIO 端口输出类型寄存器（GPIOx_OTYPER）

地址偏移量：0x04
复位值：0x0000 0000

表 9.4: GPIO 端口输出类型寄存器（GPIOx_OTYPER）（x=A，B，C，D）位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	OTy	RW	端口 x 配置位 (Port x configuration bits) 这些位通过软件写入，用于配置 I/O 输出类型。 0：推挽输出（复位状态） 1：开漏输出

9.4.3　GPIO 端口输出速度寄存器（GPIOx_OSPEEDR）

地址偏移量：0x08
复位值：

- 0x0F00 0000 (端口 A)
- 0x0000 0000 (其它端口)

表 9.5: GPIO 端口输出速度寄存器（GPIOx_OSPEEDR）（x=A，B，C，D）位描述

位	符号	访问	描述
2y+1:2y (y=0..15)	OSPEEDRy	RW	端口 x 配置位 (Port x configuration bits) 这些位通过软件写入，用于配置 I/O 输出速度。 00：高速（复位状态） 01：低速 其他：保留

9.4.4　GPIO 端口上拉/下拉寄存器（GPIOx_PUPDR）

地址偏移量：0x0C
复位值：

- 0x2580 0000 (端口 A)
- 0x0000 0000 (其它端口)

表 9.6: GPIO 端口上拉/下拉寄存器（GPIOx_PUPDR）（x=A，B，C，D）位描述

位	符号	访问	描述
2y+1:2y (y=0..15)	PUPDRy[1:0]	RW	端口 x 配置位 (Port x configuration bits) 这些位通过软件写入，用于配置 I/O 上拉或下拉。 00：无上拉或下拉 01：上拉 10：下拉 11：保留

9.4.5　GPIO 端口输入数据寄存器（GPIOx_IDR）

地址偏移量：0x10
复位值：0xXXXX XXXX

表 9.7: GPIO 端口输入数据寄存器（GPIOx_IDR）（x=A，B，C，D）位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	IDRy	R	端口输入数据 (Port input data)(y = 0..15) 这些位为只读。它们包含相应 I/O 端口的输入值。

9.4.6　GPIO 端口输出数据寄存器（GPIOx_ODR）

地址偏移量：0x14
复位值：0x0000 0000

表 9.8: GPIO 端口输出数据寄存器（GPIOx_ODR）（x=A，B，C，D）位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	ODRy	RW	当 ALTODR_SEL=0 时，端口输出数据 (Port output data) (y = 0..15) 这些位可通过软件读取和写入。

注：对于原子置位/复位，通过写入 *GPIOx_BSRR* 或 *GPIOx_BRR* 寄存器，可分别置位和/或复位 *ODR* 位 *(x = A..D)* 。

9.4.7　GPIO 端口置位/复位寄存器（GPIOx_BSRR）

地址偏移量：0x18
复位值：0x0000 0000

表 9.9: GPIO 端口置位/复位寄存器（GPIOx_BSRR）（x=A，B，C，D）位描述

位	符号	访问	描述
31:16	BRy	W	端口 x 复位位 y (Port x reset bit y) (y= 0..15) 这些位为只写。读取这些位可返回值 0x0000。 0：不会对相应的 ODRx 位执行任何操作 1：复位相应的 ODRx 位
15:0	BSy	W	端口 x 置位位 y (Port x set bit y) (y= 0..15) 这些位为只写。读取这些位可返回值 0x0000。 0：不会对相应的 ODRx 位执行任何操作 1：置位相应的 ODRx 位

9.4.8　GPIO 端口配置锁定寄存器（GPIOx_LCKR）

地址偏移量：0x1C
复位值：0x0000 0000

表 9.10: GPIO 端口配置锁定寄存器（GPIOx_LCKR）（x=A，B，C，D）位描述

位	符号	访问	描述
31:17	-	R	保留
16	LCKK	RW	锁定键 (Lock key) 可随时读取此位。可使用锁定键写序列对其进行修改。 0：端口配置锁定键未激活 1：端口配置锁定键已激活。在下一次 MCU 复位或外设复位之前，GPIOx_LCKR 寄存器始终处于锁定状态。锁定键写序列： WR LCKR[16] = ‘1’ + LCKR[15:0] WR LCKR[16] = ‘0’ + LCKR[15:0] WR LCKR[16] = ‘1’ + LCKR[15:0] RD LCKR RD LCKR[16] = ‘1’（此读操作为可选操作，但它可确认锁定已激活）
15:0	LCKy	RW	端口 x 锁定位 y (Port x lock bit y) (y= 0..15) 这些位都是读/写位，但只能在 LCKK 位等于“0”时执行写操作。 0：端口配置未锁定 1：端口配置已锁定

注：在锁定键写序列期间，不能更改 *LCK[15:0]* 的值。锁定序列中的任何错误都将中止锁定操作。在任一端口位上的第一个锁定序列之后，对 *LCKK* 位的任何读访问都将返回“*1*”，直到下一次 *MCU* 复位或外设复位为止。

9.4.9　GPIO 端口功能低寄存器（GPIOx_AFRL）

地址偏移量：0x20
复位值：0x0000 0000

表 9.11: GPIO 端口功能低寄存器（GPIOx_AFRL）（x=A，B，C，D）位描述

位	符号	访问	描述
4y+3:4y (y=0..7)	AFRy[3:0]	RW	端口 x 引脚 y 的复用功能选择 (Alternate function selection for port x pin y)(y = 0..7) 这些位通过软件写入，用于配置复用功能 I/O。 AFSELy 选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

9.4.10　GPIO 端口功能高寄存器（GPIOx_AFRH）

地址偏移量：0x24
复位值：0x0000 0000

表 9.12: GPIO 端口功能高寄存器（GPIOx_AFRH）（x=A，B，C，D）位描述

位	符号	访问	描述
4y-29 :4y-32 (y=8..15)	AFRy[3:0]	RW	端口 x 引脚 y 的复用功能选择 (Alternate function selection for port x pin y)(y = 8..15) 这些位通过软件写入，用于配置复用功能 I/O。 AFSELy 选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

9.4.11　GPIO 端口施密特寄存器（GPIOx_SMIT）

地址偏移量：0x28
复位值：

- 0x0000 FFFF(端口 A，B，C)
- 0x0000 0007(端口 D)

表 9.13: GPIO 端口施密特寄存器（GPIOx_SMIT）（x=A，B，C，D）位描述

位	符号	访问	描述
31:17	-	R	保留
15:0	SMITy[1:0]	RW	端口 x 配置位 (Port x configuration bits)(y=0..15) 这些位通过软件写入，用于配置 I/O 施密特功能。 0：施密特功能关闭 1：施密特功能使能

9.4.12　GPIO 端口驱动寄存器（GPIOx_CURRENT)

地址偏移量：0x2C

复位值：0x0000 0000

表 9.14: GPIO 端口驱动寄存器（GPIOx_CURRENT）（x=A，B，C，D）位描述

位	符号	访问	描述
2y+1:2y (y=0..15)	CURRENTy[1:0]	RW	端口 x 配置位 (Port x configuration bits) 这些位通过软件写入，用于配置 I/O 驱动电流大小。 00：4mA 驱动电流 01：8mA 驱动电流 10：12mA 驱动电流 11：16mA 驱动电流

9.4.13　GPIO 端口配置辅助寄存器（GPIOx_CFGMSK）

地址偏移量：0x30

复位值：0x0000 0000

表 9.15: GPIO 端口配置辅助寄存器（GPIOx_CFGMSK）（x=A，B，C，D）位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	CFGMSKy	RW	端口 x 配置锁定位 y (Port x lock bit y) (y= 0..15) 这些位都是读/写位，可以执行写操作。 0：端口配置未锁定 1：端口配置已锁定

9.5　AFIO 寄存器

9.5.1　外部中断配置寄存器 1 （AFIO_EXTICR1）

地址偏移量：0x08

复位值：0x0000 0000

表 9.16: 外部中断配置寄存器 1 （AFIO_EXTICR1）位描述

位	符号	访问	描述
31:16	-	R	保留
4y+3:4y (y=0..3)	EXTIy[3:0]	RW	EXTI x 配置（x = 0 到 3）(EXTI x configuration (x = 0 to 3)) 这些位通过软件写入，以选择 EXTIx 外部中断的源输入。 0000：PA[x] 引脚 0001：PB[x] 引脚 0010：PC[x] 引脚 0011：PD[x] 引脚 其它：保留

9.5.2　外部中断配置寄存器 2 （AFIO_EXTICR2）

地址偏移量：0x0C

复位值：0x0000 0000

表 9.17: 外部中断配置寄存器 2 （AFIO_EXTICR2）位描述

位	符号	访问	描述
31:16	-	R	保留
4y-13: 4y-16 (y=4..7)	EXTIy[3:0]	RW	EXTI x 配置（x = 4 到 7）(EXTI x configuration (x = 4 to 7)) 这些位通过软件写入，以选择 EXTIx 外部中断的源输入。 0000：PA[x] 引脚 0001：PB[x] 引脚 0010：PC[x] 引脚 0011：PD[x] 引脚 其它：保留

9.5.3　外部中断配置寄存器 3 （AFIO_EXTICR3）

地址偏移量：0x10

复位值：0x0000 0000

表 9.18: 外部中断配置寄存器 3 （AFIO_EXTICR3）位描述

位	符号	访问	描述
31:16	-	R	保留
4y-29 :4y-32 (y=8..11)	EXTIy[3:0]	RW	EXTI x 配置（x = 8 到 11）(EXTI x configuration (x = 8 to 11)) 这些位通过软件写入，以选择 EXTIx 外部中断的源输入。 0000：PA[x] 引脚 0001：PB[x] 引脚 0010：PC[x] 引脚 0011：PD[x] 引脚 其它：保留

9.5.4　外部中断配置寄存器 4 （AFIO_EXTICR4）

地址偏移量：0x14

复位值：0x0000 0000

表 9.19: 外部中断配置寄存器 4 （AFIO_EXTICR4）位描述

位	符号	访问	描述
31:16	-	R	保留
4y-45 :4y-48 (y=12..15)	EXTIy[3:0]	RW	EXTI x 配置（x = 12 到 15）(EXTI x configuration (x = 12 to 15)) 这些位通过软件写入，以选择 EXTIx 外部中断的源输入。 0000：PA[x] 引脚 0001：PB[x] 引脚 0010：PC[x] 引脚 0011：PD[x] 引脚 其它：保留

第十章　嵌套向量中断控制器 (NVIC)

10.1　NVIC 特性

嵌套向量中断控制器 NVIC 包含以下特性:

- WB32F10xxx 具有多达 38 个可屏蔽中断通道
- 16 个可编程优先级（使用了 4 位中断优先级）
- 低延迟异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

嵌套向量中断控制器 (NVIC) 和处理器内核接口紧密配合，可以实现低延迟的中断处理和晚到中断的高效处理。包括内核异常在内的所有中断均通过 NVIC 进行管理。

10.1.1　中断和异常向量

请参见表10.1.1, 了解 WB32F10xxx 器件的向量表。

表 10.1: WB32F10xxx 向量表

位置	优先级	优先级类型	缩略语	说明	地址
-	-	-	-	保留	0x0000_0000
-	-3	固定	复位	复位	0x0000_0004
-	-2	固定	NMI	不可屏蔽中断。时钟安全系统 (CSS) 连接到 NMI 向量。	0x0000_0008
-	-1	固定	HardFault	所有类型的错误	0x0000_000C
-	0	可设置	MemManage	存储器管理	0x0000_0010
-	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000_0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0000_0018
-	-	-	-	保留	0x0000_001C - 0x0000_002B
-	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000_002C
-	4	可设置	调试监控器	调试监控器	0x0000_0030

-	-	-	-	保留	0x0000_0034
-	5	可设置	PendSV	可挂起的系统服务	0x0000_0038
-	6	可设置	SysTick	系统节拍定时器	0x0000_003C
0	7	可设置	WWDG	窗口看门狗中断	0x0000_0040
1	8	可设置	PVD	连接到 EXTI 线的可编程电压检测 (PVD) 中断	0x0000_0044
2	9	可设置	TAMPER	入侵中断	0x0000_0048
3	10	可设置	RTC	RTC 全局中断	0x0000_004C
4	11	可设置	FMC	FMC 全局中断	0x0000_0050
5	12	可设置	RCC	RCC 全局中断	0x0000_0054
6	13	可设置	EXTI0	EXTI 线 0 中断	0x0000_0058
7	14	可设置	EXTI1	EXTI 线 1 中断	0x0000_005C
8	15	可设置	EXTI2	EXTI 线 2 中断	0x0000_0060
9	16	可设置	EXTI3	EXTI 线 3 中断	0x0000_0064
10	17	可设置	EXTI4	EXTI 线 4 中断	0x0000_0068
11	18	可设置	DMAC1	DMAC1 全局中断	0x0000_006C
12	19	可设置	DMAC2	DMAC2 全局中断	0x0000_0070
13	20	可设置	ADC	ADC 全局中断	0x0000_0074
14	21	可设置	USB	USB 全局中断	0x0000_0078
15	22	可设置	USB_DMA	USB_DMA 全局中断	0x0000_007C
16	23	可设置	EXTI9_5	EXTI 线 [9:5] 中断	0x0000_0080
17	24	可设置	TIM1_BRK	TIM1 刹车中断	0x0000_0084
18	25	可设置	TIM1_UP	TIM1 更新中断	0x0000_0088
19	26	可设置	TIM1_TRG_COM	TIM1 触发和换相中断	0x0000_008C
20	27	可设置	TIM1_CC	TIM1 捕获比较中断	0x0000_0090
21	28	可设置	TIM2	TIM2 全局中断	0x0000_0094
22	29	可设置	TIM3	TIM3 全局中断	0x0000_0098
23	30	可设置	TIM4	TIM4 全局中断	0x0000_009C
24	31	可设置	I2C1	I2C1 事件中断	0x0000_00A0
25	32	可设置	I2C2	I2C2 事件中断	0x0000_00A4
26	33	可设置	QSPI	QSPI 全局中断	0x0000_00A8
27	34	可设置	SPIM2	SPIM2 全局中断	0x0000_00AC
28	35	可设置	SPIS1	SPIS1 全局中断	0x0000_00B0
29	36	可设置	SPIS2	SPIS2 全局中断	0x0000_00B4
30	37	可设置	UART1	USART1 全局中断	0x0000_00B8
31	38	可设置	UART2	USART2 全局中断	0x0000_00BC
32	39	可设置	UART3	USART3 全局中断	0x0000_00C0
33	40	可设置	EXTI15_10	EXTI 线 [15:10] 中断	0x0000_00C4
34	41	可设置	RTCAlarm	连接到 EXTI 线的 RTC 闹钟（A 和 B）中断	0x0000_00C8
35	42	可设置	USBP_WKUP	USB PIN 唤醒中断	0x0000_00CC
36	43	可设置	I2S	I2S 全局中断	0x0000_00D0
37	44	可设置	ISO	ISO7816 全局中断	0x0000_00D4

第十一章　外部中断和事件控制器 (EXTI)

11.1　EXTI 简介

外部中断/事件控制器包含多达 19 个用于产生事件/中断请求的边沿检测器。每根输入线都可单独进行配置，以选择类型（中断或事件）和相应的触发事件（上升沿触发、下降沿触发或边沿触发）。每根输入线还可单独屏蔽。挂起寄存器用于保持中断请求的状态线。

11.2　EXTI 主要特性

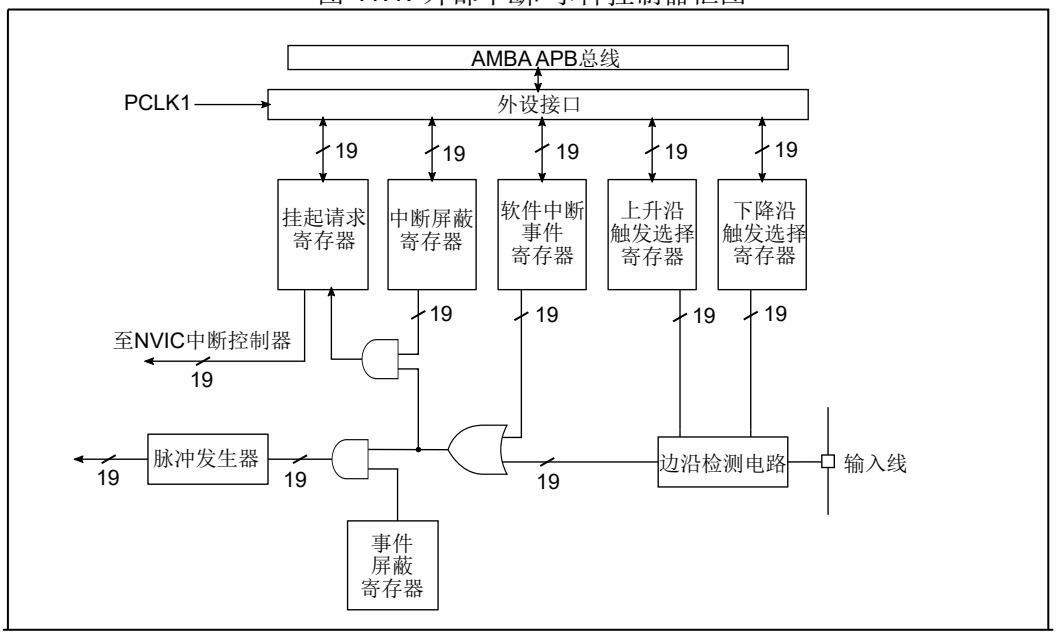
EXTI 控制器的主要特性如下：

- 每个中断/事件线上都具有独立的触发和屏蔽
- 每个中断线都具有专用的状态位
- 支持多达 19 个软件事件/中断请求
- 检测脉冲宽度低于 APB 时钟宽度的外部信号。

11.3　EXTI 框图

图 11.1 显示了框图。

图 11.1: 外部中断/ 事件控制器框图



11.4　唤醒事件管理

WB32F10xxx 器件能够处理外部或内部事件来唤醒内核 (WFE)。唤醒事件可通过以下方式产生：表10.1.1

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时使能 Cortex ® -M3 系统控制寄存器中的 SEVONPEND 位。当 MCU 从 WFE 恢复时，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部 EXTI 线为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

11.5　功能描述

要产生中断，必须先配置好并使能中断线。根据需要的边沿检测设置 2 个触发寄存器，同时在中断屏蔽寄存器的相应位写“1”使能中断请求。当外部中断线上出现选定信号沿时，便会产生中断请求，对应的挂起位也会置 1。在挂起寄存器的对应位写“1”，将清除该中断请求。

要产生事件，必须先配置好并使能事件线。根据需要的边沿检测设置 2 个触发寄存器，同时在事件屏蔽寄存器的相应位写“1”允许事件请求。当事件线上出现选定信号沿时，便会产生事件脉冲，对应的挂起位不会置 1。

通过在软件中对软件中断/事件寄存器写“1”，也可以产生中断/事件请求。

11.6　硬件中断选择

要将一根线配置为中断源，请执行以下步骤：

1. 配置相应的屏蔽位 (EXTI_IMR)

2. 配置中断线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）
3. 配置对应到外部中断控制器 (EXTI) 的 NVIC 中断通道的使能和屏蔽位，使得 19 个中断线中的请求可以被正确地响应。

11.7　硬件事件选择

要将一根线配置为中断源，请执行以下步骤：

1. 配置相应的屏蔽位 (EXTI_EMR)
2. 配置事件线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）

11.8　软件中断/ 事件选择

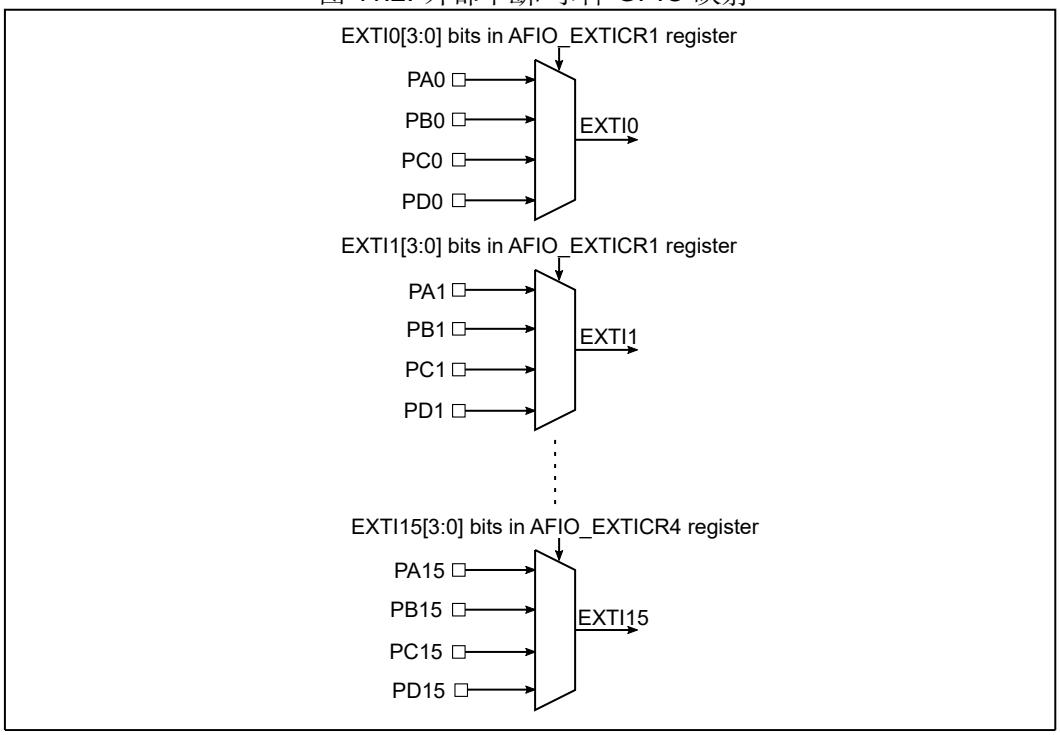
可将这根线配置为软件中断/事件线。以下为产生软件中断的步骤：

1. 配置相应的屏蔽位（EXTI_IMR、EXTI_EMR）
2. 在软件中断寄存器设置相应的请求位 (EXTI_SWIER)

11.9　外部中断/事件线映射

多达 51 个 GPIO 通过以下方式连接到 16 个外部中断/事件线：

图 11.2: 外部中断/ 事件 GPIO 映射



另外 3 根 EXTI 线连接方式如下：

- EXTI 线 16 连接到 PVD 输出
- EXTI 线 17 连接到 RTC 闹钟事件
- EXTI 线 18 连接到 USB 唤醒事件, 参考章节13.2.3。

11.10　EXTI 寄存器

11.10.1　中断屏蔽寄存器 (EXTI_IMR)

地址偏移量：0x00
复位值：0x0000 0000

表 11.1: 中断屏蔽寄存器 (EXTI_IMR) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	MRx	RW	x 线上的中断屏蔽 (Interrupt mask on line x) 0：屏蔽来自 x 线的中断请求 1：开放来自 x 线的中断请求

11.10.2　事件屏蔽寄存器 (EXTI_EMR)

地址偏移量：0x04
复位值：0x0000 0000

表 11.2: 事件屏蔽寄存器 (EXTI_EMR) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	MRx	RW	x 线上的事件屏蔽 (Event mask on line x) 0：屏蔽来自 x 线的事件请求 1：开放来自 x 线的事件请求

11.10.3　上升沿触发选择寄存器 (EXTI_RTSR)

地址偏移量：0x08
复位值：0x0000 0000

表 11.3: 上升沿触发选择寄存器 (EXTI_RTSR) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	TRx	RW	线 x 的上升沿触发事件配置位 (Rising trigger event configuration bit of line x) 0：禁止输入线上升沿触发（事件和中断） 1：允许输入线上升沿触发（事件和中断）

注：外部唤醒线配置为边沿触发时，在这些线上不能出现毛刺信号。

如果在向 *EXTI_RTSR* 寄存器写入值的同时外部中断线上产生上升沿，挂起位将被置位。在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

11.10.4　下降沿触发选择寄存器 (EXTI_FTSR)

地址偏移量：0x0C

复位值：0x0000 0000

表 11.4: 下降沿触发选择寄存器 (EXTI_FTSR) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	TRx	RW	线 x 的下降沿触发事件配置位 (Falling trigger event configuration bit of line x) 0：禁止输入线下降沿触发（事件和中断） 1：允许输入线下降沿触发（事件和中断）

注：外部唤醒线配置为边沿触发时，在这些线上不能出现毛刺信号。

如果在向 *EXTI_FTSR* 寄存器写入值的同时外部中断线上产生下降沿，挂起位不会被置位。在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

11.10.5　软件中断事件寄存器 (EXTI_SWIER)

地址偏移量：0x10

复位值：0x0000 0000

表 11.5: 软件中断事件寄存器 (EXTI_SWIER) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	SWIERx	RW	线 x 上的软件中断 (Software Interrupt on line x) 当 SWIERx 位设置为“0”时，将“1”写入该位会将 EXTI_PR 寄存器中相应挂起位置 1。 如果在 EXTI_IMR 寄存器中允许在 x 线上产生该中断，则产生中断请求。 通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

11.10.6　挂起寄存器 (EXTI_PR)

地址偏移量：0x14

复位值：0x0000 0000

表 11.6: 挂起寄存器 (EXTI_PR) 位描述

位	符号	访问	描述
31:19	-	R	保留
18:0	PRx	RC_W1	挂起位 (Pending bit) 0：未发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置“1”。 将此位编程为“1”可清除此位。

第十二章　模拟控制 (ANCTL)

12.1　模拟控制简介

WB32F10xxx 包含多个模拟功能模块，包括 ADC，CMP，USBPHY，FLASH，PVD，时钟检测模块，PLL 以及电源和时钟模块。这些模拟功能都是由模拟控制寄存器控制。

12.2　模拟控制模块主要特性

模拟功能控制模块的主要特性如下：

- 控制模拟模块的启动和关闭；
- 调整模拟模块的参数；
- 对所有的模拟功能状态进行监测；
- 支持模拟功能测试；

12.3　模拟控制模块的功能

由于模拟功能会影响产品的工作状态和性能，为防止程序的误操作，模拟控制寄存器通常处于写保护状态。用户必须对于 PWR 模块的 ANAKEY1 和 ANAKEY2 寄存器写入特定的值，才能开始改变模拟控制寄存器的状态。写操作完成后，需要清除 PWR 模块的 ANAKEY1 和 ANAKEY2 寄存器的内容，使模拟控制寄存器处于保护状态。

模拟寄存器的读操作没有任何限制。

12.3.1　Main REGU 的控制

Main Regulator 对于外部输入电压进行降压和整流，用于给整个芯片的数字逻辑提供 1.2V 工作电压。

注：*Main REGU* 控制寄存器会在上电时自动配置完成，用户程序不需要改变。

12.3.2　BandGap 的控制

BandGap 用于提供其它模拟功能模块所需要的参考电压和参考电流。

ANCTL_BGCR2 用于控制 BandGap 温度传感器的输出。

注：*BandGap* 控制寄存器会在上电时自动配置完成，用户程序不需要改变。

12.3.3　MHSI 的控制

Main OSC 是系统内部的主要 OSC，在上电启动时提供系统工作时钟，MHSI 工作频率是 8MHz。

ANCTL_MHSIENR 用于控制 8MHz OSC 的打开和关闭。

ANCTL_MHSISR 用于监测 8MHz OSC 的输出是否稳定。

注：当系统时钟选择 *MHSI* 时，用户程序不能关闭 *8MHz OSC*。

12.3.4　FHSI 的控制

Flash OSC 是系统内部的另一个 OSC，在 Flash 写操作时提供工作时钟，FHSI 也可以用作系统时钟。FHSI 工作频率是 48MHz。

ANCTL_FHSIENR 用于控制 48MHz OSC 的打开和关闭。

ANCTL_FHSISR 用于监测 48MHz OSC 的输出是否稳定。

注：当系统时钟选择 *FHSI* 时，用户程序不能关闭 *48MHz OSC*。

12.3.5　LSI 的控制

LSI 是系统内部的低速 OSC，用于给一些低速模块提供时钟，如 RTC，IWDG，LSI 工作频率是 32KHz。

ANCTL_LSIENR 用于控制 32KHz OSC 的打开和关闭。

ANCTL_LSISR 用于监测 32KHz OSC 的输出是否稳定。

12.3.6　HSE 的控制

HSE 是系统外部的高速晶振，用于给系统提供精确的工作时钟，HSE 工作频率是 8MHz。

ANCTL_HSECR0 和 ANCTL_HSECR1 用于控制 HSE 的工作模式：

- 晶振模式
- 外部输入时钟模式

HSE 的工作模式配置如下表12.1：

表 12.1: HSE 工作模式配置

HSE 工作模式	HSEON	BYPASS	PADOEN
晶振模式	1	0	1
外部时钟模式	0	1	0

ANCTL_HSESR 用于监测 HSE 的输出是否稳定。

注：当系统时钟选择 *HSE* 时，用户可以启动频率检测模块，监控 *HSE* 的输出是否稳定。当监测到 *HSE* 的输出低于配置的检测频率时，系统会产生一个不可屏蔽中断，并且系统时钟自动切换到内部的 *MHSI* 运行。

当系统时钟选择 *HSE* 时，用户程序不能改变 *HSE* 的工作状态。

12.3.7　Flash REGU 的控制

Flash Regulator 对于外部输入电压进行降压和整流，用于给 Flash 提供 2.1V 工作电压。

注：*Flash REGU* 控制寄存器会在上电时自动配置完成，用户程序不需要改变。

12.3.8　PLL 的控制

PLL 用于在系统高速运行时提供系统工作时钟，PLL 输出的最高工作频率是 128MHz。

ANCTL_PLLCR 用于控制 PLL 参数，用于调整输出频率以及倍频系数。PLL 输入时钟的预分频系数由 RCC 模块的 PLLPRE 寄存器控制.

PLL 典型工作频率配置如表12.2

表 12.2: PLL 工作频率配置表

HSE 输入频率	预分频系数	倍频系数	PLL 输出频率
8MHz	1	16	128MHz
8MHz	2	12	48MHz
8MHz	2	20	80MHz
8MHz	2	24	96MHz
6MHz	1	16	96MHz
6MHz	3	24	96MHz

ANCTL_PLLENR 用于 PLL 的使能控制。ANCTL_PLLSR 用于监测 PLL 的输出是否稳定。

注：当系统时钟选择 *PLL* 输出时，用户程序不能关闭 *PLL*。

12.3.9　PVD 的控制

PVD 用于检测芯片的输入电压，当外部输入电压低于预先配置的电压并且 PVD 使能，则检测到 PVD 事件。PVD 触发事件输入到 EXTI 的第 16 个通道，可以产生 PVD 中断。

ANCTL_PVDCR 用于配置 PVD 的检测电压, 见表12.3：

表 12.3: PVD 检测电压配置表

PLS	电压下降检测点	电压上升检测点
3’b000	2.14	2.25
3’b001	2.24	2.35
3’b010	2.34	2.45
3’b011	2.44	2.55
3’b100	2.54	2.65
3’b101	2.64	2.75
3’b110	2.74	2.85
3’b111	2.84	2.95

ANCTL_PVDENR 用于 PVD 的使能控制和测试功能。

PVD 输出状态可以参考 PWR_SR0 寄存器。

12.3.10　ADC 的控制

ANCTL_SARENR 用于控制 ADC 内部时钟延时以及的使能功能。

注：当设置 *SAREN* 为*'1'* 时，*ADC* 必须启动一次 *Calibration* 操作才能开始使用，具体操作请参考相关章节。

12.3.11　USBPHY 的控制

USBPHY 控制寄存器用于设置 USBPHY 的状态，使内部的 USB 控制模块可以通过 USBPHY 跟外部设备进行通讯。

USBPHY 的配置如下表12.4：

表 12.4: USBPHY 配置表

控制位	工作模式
USBPEN	1
DPPUEN	1
HIGHRESEN	0
DMSTEN	0
DPSTEN	0

12.3.12　POR 的控制

ANCTL_PORCR 用于控制 POR 模块的使能功能。

注：*ANCTL_PORCR* 控制寄存器会在上电时自动配置完成，用户程序不需要改变。

12.3.13　CMP 的控制

WB32F10xxx 内部配置有两个电压比较模块。每个电压比较器有两个输入端，每个输入端可以从 4 个不同的 IO 输入比较电压。

ANCTL_CMPACR 和 ANCTL_CMPBCR 用于控制 CMP 的使能和输入电压通道的选择。

ANCTL_CMPASR 和 ANCTL_CMPBSR 用于读取电压比较的结果。

CMPA 的通道选择配置表如下：

表 12.5: CMPA 负端通道配置表

NSEL	通道选择	输入电压管脚
0x00	负端输入通道 0	PA9
0x01	负端输入通道 1	PB6
0x02	负端输入通道 2	PB7
0x03	负端输入通道 3	PA14
其它	保留	-

表 12.6: CMPA 正端通道配置表

PSEL	通道选择	输入电压管脚
0x00	正端输入通道 0	PA8
0x01	正端输入通道 1	PB4
0x02	正端输入通道 2	PB5
0x03	正端输入通道 3	PA13
其它	保留	-

CMPB 的通道选择配置表如下：

表 12.7: CMPB 负端通道配置表

NSEL	通道选择	输入电压管脚
0x00	负端输入通道 0	PC12
0x01	负端输入通道 1	PB8
0x02	负端输入通道 2	PB9
0x03	负端输入通道 3	PB3
其它	保留	-

表 12.8: CMPB 正端通道配置表

PSEL	通道选择	输入电压管脚
0x00	正端输入通道 0	PD2
0x01	正端输入通道 1	PC10
0x02	正端输入通道 2	PC11
0x03	正端输入通道 3	PA15
其它	保留	-

12.3.14　CSS 的控制

输入频率检测模块 (CSS) 主要是用来检测 HSE 输入的频率是否停止，如果检测到输入频率停止，则会产生不可屏蔽中断，同时，系统自动切换到内部 MHSI 时钟运行。

ANCTL_CSSENR 用于控制时钟检测频率的使能。

ANCTL_CSSCR 需要配置成 0x3。

12.4　模拟功能控制寄存器

12.4.1　BG 控制寄存器 2 (ANCTL_BGCR2)

地址偏移量：0x1C

复位值：0x0000 0000

表 12.9: BG 控制寄存器 2 (ANCTL_BGCR2) 位描述

位	符号	访问	描述
31:2	-	R	保留
1	TEMPOUTEN	RW	Bandgap 温度传感器输出通道使能控制 0: 温度传感器输出通道关闭 1: 温度传感器输出通道打开
0	-	R	保留

12.4.2　MHSI 使能寄存器 (ANCTL_MHSIENR)

地址偏移量：0x2C

复位值：0x0000 0001

表 12.10: MHSI 使能寄存器 (ANCTL_MHSIENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	MHSION	RW	内部 OSC_8MHz 的使能控制 0: 内部 OSC_8MHz 关闭 1: 内部 OSC_8MHz 打开

12.4.3　MHSI 状态寄存器 (ANCTL_MHSISR)

地址偏移量：0x30

复位值：0x0000 0001

表 12.11: MHSI 状态寄存器 (ANCTL_MHSISR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	MHSIRDY	R	内部 OSC_8MHz 输出状态 0: 内部 OSC_8MHz 输出未稳定 1: 内部 OSC_8MHz 输出稳定

12.4.4　FHSI 使能寄存器 (ANCTL_FHSIENR)

地址偏移量：0x38

复位值：0x0000 0001

表 12.12: FHSI 使能寄存器 (ANCTL_FHSIENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	FHSION	RW	内部 OSC_48MHz 的使能控制 0: 内部 OSC_48MHz 关闭 1: 内部 OSC_48MHz 打开

12.4.5　FHSI 状态寄存器 (ANCTL_FHSISR)

地址偏移量：0x3C
复位值：0x0000 0001

表 12.13: FHSI 状态寄存器 (ANCTL_FHSISR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	FHSIRDY	R	内部 OSC_48MHz 输出状态 0: 内部 OSC_48MHz 输出未稳定 1: 内部 OSC_48MHz 输出稳定

12.4.6　LSI 使能寄存器 (ANCTL_LSIENR)

地址偏移量：0x44
复位值：0x0000 0000

表 12.14: LSI 使能寄存器 (ANCTL_LSIENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	LSION	RW	内部 OSC_32KHz 的使能控制 0: 内部 OSC_32KHz 关闭 1: 内部 OSC_32KHz 打开

12.4.7　LSI 状态寄存器 (ANCTL_LSISR)

地址偏移量：0x48
复位值：0x0000 0000

表 12.15: LSI 状态寄存器 (ANCTL_LSISR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	LSIRDY	R	内部 OSC_32KHz 输出状态 0: 内部 OSC_32KHz 输出未稳定 1: 内部 OSC_32KHz 输出稳定

12.4.8　HSE 控制寄存器 0 (ANCTL_HSECR0)

地址偏移量：0x4C
复位值：0x0000 0000

表 12.16: HSE 控制寄存器 0 (ANCTL_HSECR0) 位描述

位	符号	访问	描述
31:2	-	R	保留
1	BYPASS	RW	外部时钟输入模式使能 0: 外部时钟输入模式关闭 1：外部时钟输入模式使能
0	HSEON	RW	外部 8MHz 晶振使能 0: 外部 8MHz 晶振关闭 1：外部 8MHz 晶振使能

12.4.9　HSE 控制寄存器 1 (ANCTL_HSECR1)

地址偏移量：0x50
复位值：0x0000 0000

表 12.17: HSE 控制寄存器 1 (ANCTL_HSECR1) 位描述

位	符号	访问	描述
31:2	-	R	保留
1	PADOEN	RW	外部 8MHz 晶振输出反馈功能控制 0: 外部 8MHz 晶振输出反馈功能关闭 1：外部 8MHz 晶振输出反馈功能打开
0	-	R	保留

12.4.10　HSE 状态寄存器 (ANCTL_HSESR)

地址偏移量：0x58
复位值：0x0000 0000

表 12.18: HSE 状态寄存器 (ANCTL_HSESR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	HSERDY	R	外部 8MHz 晶振输出状态 0: 外部 8MHz 晶振输出未稳定 1: 外部 8MHz 晶振输出稳定

12.4.11　PLL 控制寄存器 (ANCTL_PLLCR)

地址偏移量：0x74

复位值：0x0000 0000

表 12.19: PLL 控制寄存器 (ANCTL_PLLCR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:6	PLLMUL	RW	配置 PLL 的倍频系数 00: 输出频率倍频系数为 24 01: 输出频率倍频系数为 20 10: 输出频率倍频系数为 16 11: 输出频率倍频系数为 12
5:0	-	R	保留

注意：PLL 倍频后的频率不能超过 128MHz。

12.4.12　PLL 使能寄存器 (ANCTL_PLLENR)

地址偏移量：0x78
复位值：0x0000 0000

表 12.20: PLL 使能寄存器 (ANCTL_PLLENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	PLLON	RW	PLL 使能控制 0: PLL 关闭 1: PLL 打开

12.4.13　PLL 状态寄存器 (ANCTL_PLLSR)

地址偏移量：0x7C
复位值：0x0000 0000

表 12.21: PLL 状态寄存器 (ANCTL_PLLSR) 位描述

位	符号	访问	描述
31:2	-	R	保留
1:0	PLLRDY	R	PLL 时钟输出状态 00: PLL 时钟输出未稳定 01: PLL 时钟输出达到目标频率的 90% 10: 非法状态 11: PLL 时钟输出稳定

12.4.14　PVD 控制寄存器 (ANCTL_PVDCR)

地址偏移量：0x80

复位值：0x0000 0004

表 12.22: PVD 控制寄存器 (ANCTL_PVDCR) 位描述

位	符号	访问	描述
31:3	-	R	保留
2:0	PLS	RW	配置 PVD 的比较电压

12.4.15　PVD 使能寄存器 (ANCTL_PVDENR)

地址偏移量：0x84
复位值：0x0000 0000

表 12.23: PVD 使能寄存器 (ANCTL_PVDENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	PVDE	RW	PVD 使能控制 0: PVD 关闭 1: PVD 打开

12.4.16　SARADC 使能寄存器 (ANCTL_SARENR)

地址偏移量：0x8C
复位值：0x0000 001E

表 12.24: SARADC 使能寄存器 (ANCTL_SARENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SAREN	RW	SARADC 使能控制 0: SARADC 关闭 1: SARADC 打开

12.4.17　USBPHY 控制寄存器 (ANCTL_USBPCR)

地址偏移量：0x90
复位值：0x0000 0002

表 12.25: USBPHY 控制寄存器 (ANCTL_USBPCR) 位描述

位	符号	访问	描述
31:5	-	R	保留
4	DPSTEN	RW	USBPHY DP 输入通道的施密特功能控制 通过 SFM 模块检测 USB 端口状态时，需要打开该功能。 0: DP 输入通道的施密特功能关闭 1: DP 输入通道的施密特功能打开

3	DMSTEN	RW	USBPHY DM 输入通道的施密特功能控制 通过 SFM 模块检测 USB 端口状态时，需要打开该功能。 0: DM 输入通道的施密特功能关闭 1: DM 输入通道的施密特功能打开
2	HIGHRESEN	RW	USBPHY 输入高阻功能控制 0: USBPHY 输入高阻功能关闭 1: USBPHY 输入高阻功能打开
1	DPPUEN	RW	DP 上拉控制 0: DP 上拉功能关闭 1: DP 上拉功能打开 DP 上拉时，USBPHY 输入高阻功能需要打开。
0	USBPEN	RW	USBPHY 使能控制 0: USBPHY 关闭 1: USBPHY 使能

12.4.18　POR 控制寄存器 (ANCTL_PORCR)

地址偏移量：0x94
复位值：0x0000 0841

表 12.26: POR 控制寄存器 (ANCTL_PORCR) 位描述

位	符号	访问	描述
31:12	-	R	保留
11:0	POREN	RW	POR 使能控制 0x7BE: POR 关闭 其它值：POR 使能

12.4.19　CMPA 控制寄存器 (ANCTL_CMPACR)

地址偏移量：0x98
复位值：0x0000 0000

表 12.27: CMPA 控制寄存器 (ANCTL_CMPACR) 位描述

位	符号	访问	描述
31:9	-	R	保留
8	CMPAEN	RW	CMPA 使能控制 0: CMPA 关闭 1: CMPA 打开

7:4	NSEL	RW	CMPA 负端输入通道选择 0000: 选择 CMPA 负端输入通道 0 0001: 选择 CMPA 负端输入通道 1 0010: 选择 CMPA 负端输入通道 2 0011: 选择 CMPA 负端输入通道 3 其它: 保留
3:0	PSEL	RW	CMPA 正端输入通道选择 0000: 选择 CMPA 正端输入通道 0 0001: 选择 CMPA 正端输入通道 1 0010: 选择 CMPA 正端输入通道 2 0011: 选择 CMPA 正端输入通道 3 其它: 保留

12.4.20　CMPB 控制寄存器 (ANCTL_CMPBCR)

地址偏移量：0x9C
复位值：0x0000 0000

表 12.28: CMPB 控制寄存器 (ANCTL_CMPBCR) 位描述

位	符号	访问	描述
31:9	-	R	保留
8	CMPBEN	RW	CMPB 使能控制 0: CMPB 关闭 1: CMPB 打开
7:4	NSEL	RW	CMPB 负端输入通道选择 0000: 选择 CMPB 负端输入通道 0 0001: 选择 CMPB 负端输入通道 1 0010: 选择 CMPB 负端输入通道 2 0011: 选择 CMPB 负端输入通道 3 其它: 保留
3:0	PSEL	RW	CMPB 正端输入通道选择 0000: 选择 CMPB 正端输入通道 0 0001: 选择 CMPB 正端输入通道 1 0010: 选择 CMPB 正端输入通道 2 0011: 选择 CMPB 正端输入通道 3 其它: 保留

12.4.21　INT 状态寄存器 (ANCTL_ISR)

地址偏移量：0xA0
复位值：0x0000 0003

表 12.29: INT 状态寄存器 (ANCTL_ISR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	CSSIS	R	HSE clock failure 检测中断标志
6	-	R	保留
5	PLLIS	R	PLL 输出稳定中断标志
4	LSEIS	R	32KHz 晶振输出稳定中断标志
3	HSEIS	R	8MHz 晶振输出稳定中断标志
2	LSIIS	R	32KHz OSC 输出稳定中断标志
1	FHSIIS	R	48MHz Flash OSC 输出稳定中断标志
0	MHSIIS	R	8MHz Main OSC 输出稳定中断标志

12.4.22　INT 使能寄存器 (ANCTL_IER)

地址偏移量：0xA4

复位值：0x0000 0000

表 12.30: INT 使能寄存器 (ANCTL_IER) 位描述

位	符号	访问	描述
31:6	-	R	保留
5	PLLIE	RW	PLL 输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生
4	LSEIE	RW	32KHz 晶振输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生
3	HSEIE	RW	8MHz 晶振输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生
2	LSIIE	RW	32KHz OSC 输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生
1	FHSIIE	RW	48MHz Flash OSC 输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生
0	MHSIIE	RW	8MHz Main OSC 输出稳定中断使能 0: 禁止中断发生 1: 允许中断发生

注：*HSE clock failure* 检测 *(CSS)* 是不可屏蔽中断。

12.4.23　INT 清除寄存器 (ANCTL_ICR)

地址偏移量：0xA8
复位值：0x0000 0000

表 12.31: INT 清除寄存器 (ANCTL_ICR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	CSSIC	RW	清除 HSE clock failure 检测中断标志 对这个位写’1‘会清除相应的中断标志。
6	-	R	保留
5	PLLIC	RW	清除 PLL 输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。
4	LSEIC	RW	清除 32KHz 晶振输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。
3	HSEIC	RW	清除 8MHz 晶振输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。
2	LSIIC	RW	清除 32KHz OSC 输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。
1	FHSIIC	RW	清除 48MHz Flash OSC 输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。
0	MHSIIC	RW	清除 8MHz Main OSC 输出稳定中断标志 对这个位写’1‘会清除相应的中断标志。

12.4.24　CMPA 状态寄存器 (ANCTL_CMPASR)

地址偏移量：0xAC
复位值：0x0000 0000

表 12.32: CMPA 状态寄存器 (ANCTL_CMPASR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	CMPAOUT	RW	CMPA 输出比较结果 0: CMP 正端输入电压小于负端输入电压 1: CMP 正端输入电压大于负端输入电压

12.4.25　CMPB 状态寄存器 (ANCTL_CMPBSR)

地址偏移量：0xB0
复位值：0x0000 0000

表 12.33: CMPB 状态寄存器 (ANCTL_CMPBSR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	CMPBOUT	RW	CMPB 输出比较结果 0: CMP 正端输入电压小于负端输入电压 1: CMP 正端输入电压大于负端输入电压

12.4.26　CSS 使能寄存器 (ANCTL_CSSENR)

地址偏移量：0xB4

复位值：0x0000 0000

表 12.34: CSS 使能寄存器 (ANCTL_CSSENR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	CSSON	RW	HSE clock failure 检测使能控制 0: HSE clock failure 检测关闭 1: HSE clock failure 检测打开

12.4.27　CSS 配置寄存器 (ANCTL_CSSCR)

地址偏移量：0xB8

复位值：0x0000 0000

表 12.35: CSS 配置寄存器 (ANCTL_CSSCR) 位描述

位	符号	访问	描述
31:12	-	R	保留
11:0	FREQCNT	RW	配置 CSS 计数值，建议配置成 0x03

第十三章　特殊功能宏 (SFM)

13.1　特殊功能宏简介

特殊功能宏 (SFM) 主要提供了几种特殊用途的功能：

1. 统计一个 WORD（32bit）中二进制数 1 的个数。
2. 实现对一个 WORD（32bit）进行倍宽的操作（最多 8 倍宽）。
3. 支持 USB 端口状态检测控制和中断控制。

13.2　功能描述

13.2.1　统计二进制数 1 的个数

该功能用于统计一个 WORD（32bit）中二进制数 1 的个数。该功能的使用流程如下：

1. EXPEN 位（SFM_CTRL[3]）设置为 0。
2. 将要计算的数据写入 SFM_DATA 寄存器中。
3. 读取 SFM_DOUT0 即可得到结果。

13.2.2　对一个 WORD（32bit）进行倍宽操作

该功能用于对一个 WORD（32bit）进行倍宽的操作（最多 8 倍宽）。该功能的使用流程如下（这里以 3 倍宽为例）：

1. EXPEN 位（SFM_CTRL[3]）设置为 1。
2. 将要操作的数据写入 SFM_DATA 寄存器中。
3. 读取 SFM_DOUT0, SFM_DOUT1, SFM_DOUT2 即可获取三倍宽后的结果。

13.2.3　USB 端口状态检测和中断控制

USB 端口状态检测支持以下功能：

- 可以检测的 USB 端口状态有：SE0/SE1/J-STAT/K-STAT 。

- 每种 USB 端口状态都有相应的状态检测使能控制。
- USB 状态事件送到 EXTI[18] 用来唤醒低功耗模式。

13.3　SFM 寄存器描述

13.3.1　控制寄存器 (SFM_CTRL)

地址偏移量：0x00
复位值：0x0000 0000

表 13.1: SFM 控制寄存器 (SFM_CTRL) 位描述

位	符号	访问	描述
31:4	-	R	保留
3	EXPEN	RW	0：统计数据寄存器中二进制位 1 的个数。 1：对数据寄存器中的数据进行相应的倍宽操作。
2:0	EXPRATE	RW	当 EXPEN=1 时，该位域指定倍宽的倍数。 000：一倍宽（结果与输入数据一致） 001：二倍宽 010：三倍宽 011：四倍宽 100：五倍宽 101：六倍宽 110：七倍宽 111：八倍宽

13.3.2　SFM 数据寄存器 (SFM_DATA)

地址偏移量：0x04
复位值：0x0000 0000

表 13.2: SFM 数据寄存器 (SFM_DATA) 位描述

位	符号	访问	描述
31:0	DATA	RW	存放需要运算的数据

13.3.3　SFM 结果寄存器 (SFM_DOUT*x*)

地址偏移量：0x08 - 0x24
复位值：0x0000 0000

表 13.3: SFM 结果寄存器 (SFM_DOUT*x*) 位描述

位	符号	访问	描述
31:0	DOUT	R	存放操作完成后的结果。

13.3.4　USB 端口状态检测控制/状态寄存器 (SFM_USBPSDCSR)

地址偏移量：0x44

复位值：0x0000 0000

表 13.4: USB 端口状态检测控制/状态寄存器 (SFM_USBPSDCSR) 位描述

位	符号	访问	描述
31:12	-	R	保留
11	SE1EN	RW	USB 端口'SE1' 状态检测使能。 需要设置 ANCTL_USBPCR 中的 DMSTEN 和 DPSTEN 位。
10	KSTATEN	RW	USB 端口'K' 状态检测使能。 需要设置 ANCTL_USBPCR 中的 DMSTEN 和 DPSTEN 位。
9	JSTATEN	RW	USB 端口'J' 状态检测使能。 需要设置 ANCTL_USBPCR 中的 DMSTEN 和 DPSTEN 位。
8	SE0EN	RW	USB 端口'SE0' 状态检测使能。 需要设置 ANCTL_USBPCR 中的 DMSTEN 和 DPSTEN 位。
7:4	-	R	保留
3	SE1F	RW	当 USB 端口状态切换到'SE1' 状态时，该为由硬件置'1'。 该位由软件清'0'。
2	KSTATF	RW	当 USB 端口状态切换到'K' 状态 (Differential "0") 时，该为由硬件置'1'。 该位由软件清'0'。
1	JSTATF	RW	当 USB 端口状态切换到'J' 状态 (Differential "1") 时，该为由硬件置'1'。 该位由软件清'0'。
0	SE0F	RW	当 USB 端口状态切换到'SE0' 状态时，该为由硬件置'1'。 该位由软件清'0'。

13.3.5　USB 端口状态寄存器 (SFM_USBPSTAT)

地址偏移量：0x48

复位值：0x0000 0008

表 13.5: USB 端口状态寄存器 (SFM_USBPSTAT) 位描述

位	符号	访问	描述
31:6	-	R	保留
5	FULL_SPEED	R	USB 处于全速模式。
4	SUSPEND	R	USB 处于 SUSPEND 状态。
3:0	-	R	保留

第十四章　DMA 控制器 (DMAC)

14.1　DMAC 简介

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

WB32F10xxx 有两个完全相同的 DMA 控制器 DMAC1 和 DMAC2。每个 DMA 控制器有 3 个通道（一共有 6 个通道），每个通道可以单独配置，管理各种类型 DMA 传输。每个 DMAC 内还有一个仲裁器来协调各个 DMA 请求的优先权。

14.2　DMAC 主要特性

- 2 个 AHB 主控模块接口，1 个 AHB 从模块接口。AHB 接口符合 AMBA 2.0 协议。
- 3 个独立的传输通道
- 16 组硬件握手信号
- 可以提前结束数据传输，而且没有数据丢失
- 支持外设与外设之间，外设与存储器之间，存储器与存储器之间的告诉数据传输
- DMA 传输的源头和目的地可以在总线矩阵的同一层，也可以在不同层。
- 可编程的数据传输数目
- 每个通道可以单独配置源头和目的地址、传输总线类型 (AHB 或者 APB)、握手接口等
- 每个通道的优先级可配，有 0~7 一共 8 个等级，7 的优先级最高，0 的优先级最低。
- 每个通道都直接连接专用的硬件 DMA 请求，但同时也支持软件触发。这些功能通过软件来配置。
- 支持单个数据块 (block) 的 DMA 传输
- 支持多个数据块 (block) 的 DMA 传输
  - 链表式
  - 自动重新装载
  - 连续数据块
- 仅支持 DMAC 作为数据传输的流量调节器 (flow controller)

- 支持分发操作和收集操作
- 支持通道锁定
- 支持总线锁定
- 每个通道含有一个数据缓冲器，支持 FIFO 模式
- 支持伪飞行操作 (Pseudo Fly-By Operation)

14.3　DMAC 使用注意事项

使用 DMAC 进行高速数据传输时，必须正确配置其他部分与它协同工作。

14.3.1　I/O 总线配置

如果需要通过 I/O 总线与片外的器件进行数据传输，用户需要通过 GPIO 和 AFIO 来正确配置 I/O 总线。

14.3.2　时钟配置

为了节省功耗，用户可以配置 RCC 的寄存器来关闭暂时不工作部分的时钟。在进行 DMAC 传输之前，用户需要保证已打开 DMAC1/2 的时钟和需要工作的总线时钟。

14.3.3　中断配置

DMAC 有一个中断申请信号送给 CPU，详细信息请参考图11.1。在默认情况下，该中断请求是禁用的。用户需要在开始 DMA 传输之前，正确配置该中断的优先级和使能该中断。

14.3.4　外设配置

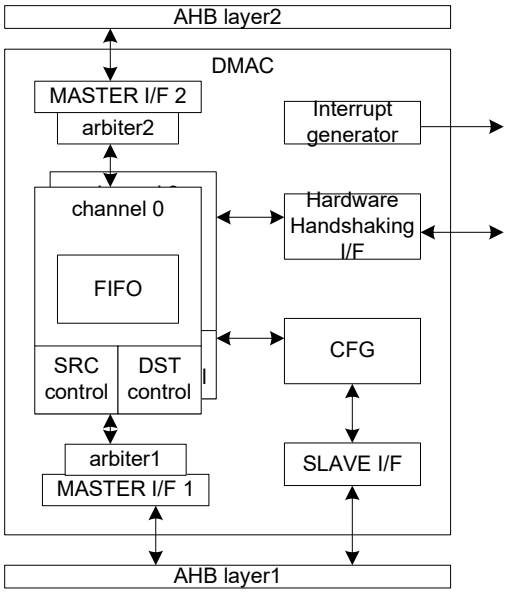
用户需要正确配置参与 DMA 传输的外设，包括时钟配置。详细信息请参考各外设相应章节的描述。

14.4　DMAC 功能描述

DMA 控制器和 Cortex™-M3 核心共享系统数据总线，执行直接存储器数据传输。当 CPU 和 DMA 同时访问同一目标 (RAM 或外设) 时，DMA 请求会暂停 CPU 访问系统总线达若干个周期，总线仲裁器执行循环调度，以保证 CPU 至少可以得到一半的系统总线 (存储器或外设) 带宽。

CPU 通过 AHB 从模块接口配置 DMAC，进行外设和外设之间、外设和存储器之间或者存储器和存储器之间的高速数据传输。DMAC 有两个主模块接口，可以支持多层总线的数据传输和伪飞行操作。每个数据传输在 DMAC 的一个通道进行，每个通道有一个 FIFO。在 FIFO 模式下，可以进行数据突发传输，从而减少 DMAC 申请系统总线的次数，加快数据传输。

图 14.1: DMA 框图



14.4.1　DMA 处理

DMA 传输由给定数目的数据传输序列组成。要传输的数据项的数目以及宽度 (8,16,32) 可以配置。

每个 DMA 传输包括三项操作：

- 通过 SARx 寄存器寻址，从外设数据寄存器或者存储器中加载数据。
- 通过 DARx 寄存器寻址，将加载的数据存储到外设数据寄存器或者存储器中。
- 数据传输结束之后，更新 SARx 和 DARx

在产生事件后，外设会向 DMA 发送请求信号。DMAC 根据通道优先级处理该请求。只要 DMA 访问外设，DMAC 就会向外设发送确认信号。外设获得 DMAC 的确认信号之后，便会立即释放其请求。一旦外设请求失败，DMAC 就会释放确认信号。如果有更多请求，外设可以启动下一个事务。

14.4.2　仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。每个通道有两根申请线，源端和目的地端传输分别申请 HSB 的使用权。

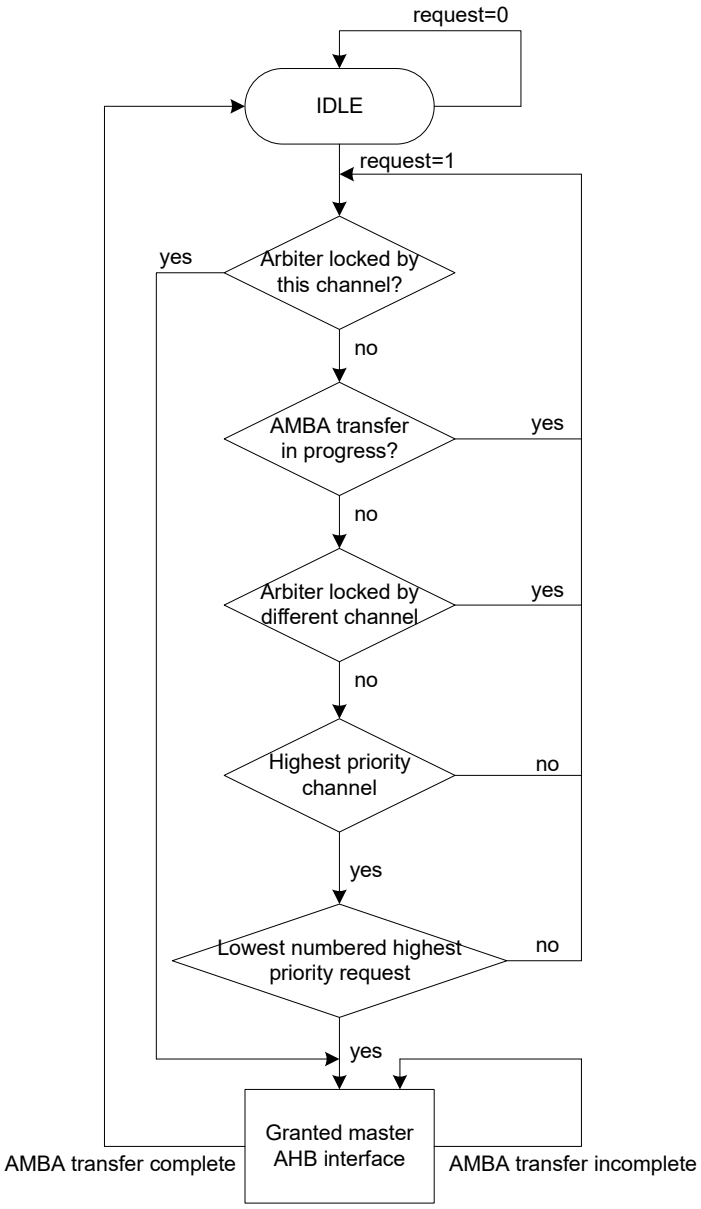
优先级管理分两个阶段：

- 软件：每个通道的优先级在 CFGx.CH_PRIOR 中配置，0 是最低优先级。
- 硬件：如果多个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有较高的优先级。例如：通道 2 优先于通道 5。

仲裁器在当前的 HSB 总线传输结束之后才会开始新的仲裁。如果当一个优先级较低的通道正在进行数据传输时，另一更高优先级的通道申请使用 HSB 总线，DMAC 等当前的传输结束之后，再响应新的申请。

下面的图详细显示了这一过程。

图 14.2: DMA 仲裁过程



14.4.3　DMA 握手接口

当 DMA 与外设之间进行数据传输时，需要握手协议来管理 DMA 传输的开始、结束等。有两种握手接口可以选择：硬件握手接口和软件握手接口。用户可以通过通道配置寄存器来配置。具体详见14.5.5。

软件握手接口

软件握手的好处是不需要占用 DMA 的硬件握手信号组，通过内存映射寄存器完成。

当外设要求 DMA 传输时，向 CPU 发送中断申请。然后中断服务程序使用软件寄存器来启动和控制 DMA。这些软件寄存器用于实现软件握手接口。

由于流控制器总是 DMAC，不需要使用最后一个事务寄存器 LstSrcReg 和 LstDstReg。

硬件握手

每个 DMAC 和外设之间有 16 个硬件握手接口。关于这些接口的设备特定映射，请参考14.4.10。每个通道的源端和目的地段可以通过 CFGxL.HS_SEL_SRC, CFGxH.SRC_PER, CFGxL.HS_SEL_DST, CFGxH.DST_PER,

分别选择接口类型和接口的号码。

当外设要求进行 DMA 传输时，升高相应的 dma_req 信号 (如果是单次交易，升高 dma_single_req)。如果 DMAC 可以响应该申请，就开始 DMA 传输，同时升高相应的 dma_ack 信号。当外设结束所有的传输时，拉低 dma_req 或者 dma_single_req。DMAC 在下一个周期拉低 dma_ack。以下两张图显示了这一过程。每次 DMA 传输的交易数量由 CTLx.SRC_MSIZE 和 CTLx.DST_MSIZE 决定。

图 14.3: DMA 单次交易

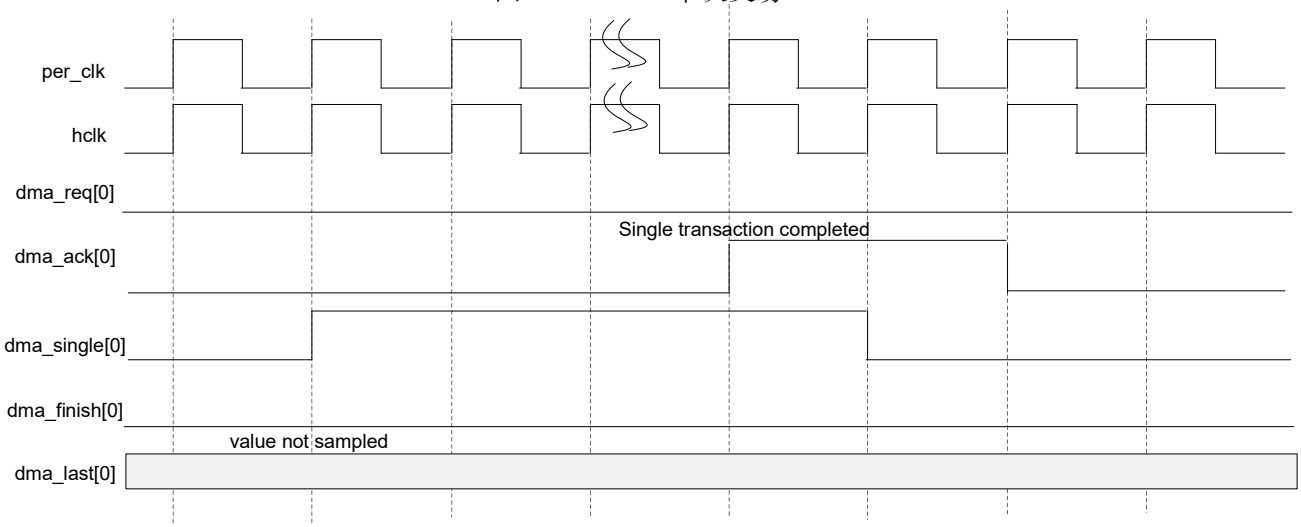
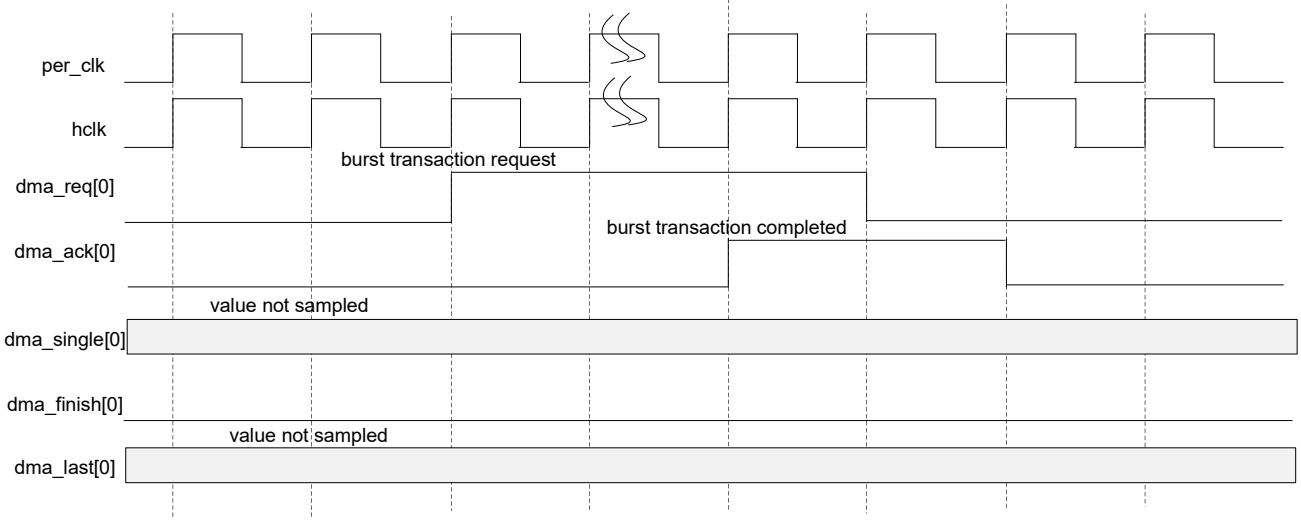


图 14.4: DMA 突发式交易



注：*DMA 与存储器之间的数据传输不需要握手协议。*

14.4.4　收集 (gather)

收集与块内的源传输相关。当到达散布边界时，从源读取数据的系统总线地址以编程量 (撒布增量) 递增或者递减。

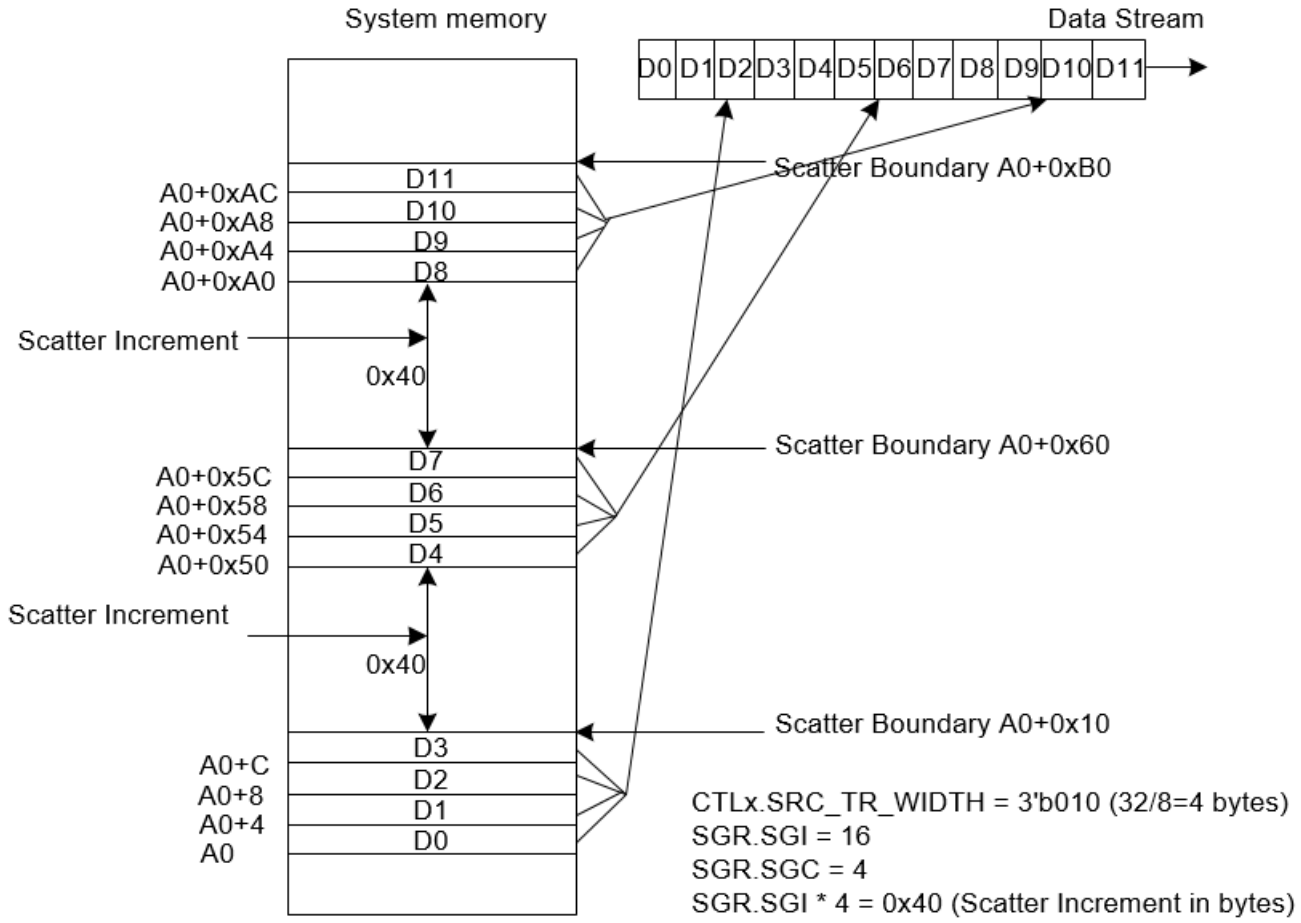
$$\text{编程量} = SGRx.SGI \times (CTLxL.SRC_TR_WIDTH\text{对应的字节数})/8$$

在连续的散布边界之间从源读取的数据量配置在 SGRx.SGC 中。

将 CTLxL.SRC_GAT_EN 值为 1, 可以使能收集功能。CTLxL.SINC 决定每次地址变化时递增，递减还是保持不变。

下图以实例说明这一功能。

图 14.5: DMA 源收集传输



14.4.5　分发 (scatter)

分发与块内目的地传输相关。当到达散布边界时，向目的地传输的系统总线地址以编程量 (撒布增量) 递增或者递减。

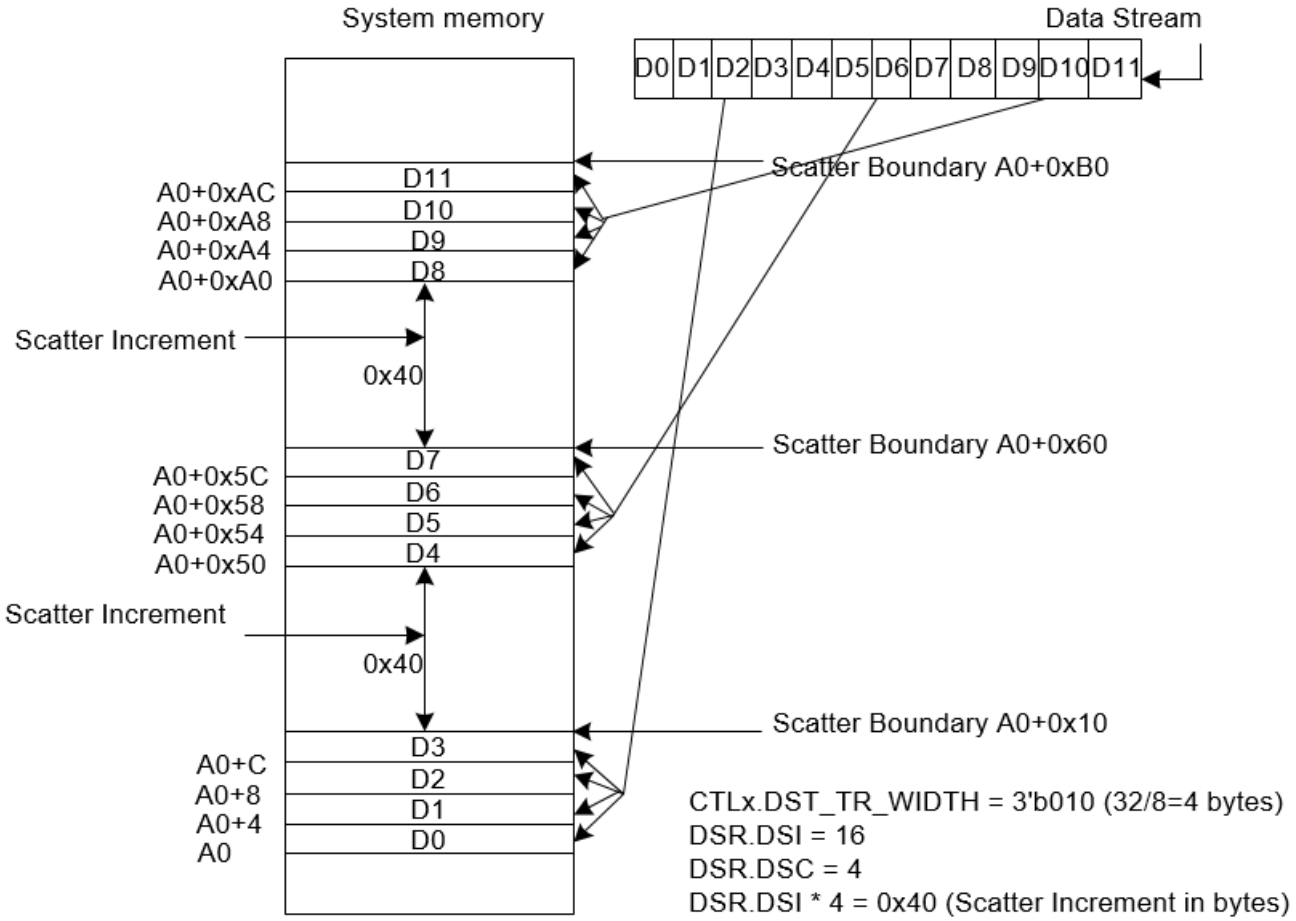
$$编程量 = DSRx.DSI \times (CTLxL.DST_TR_WIDTH对应的字节数)/8$$

在连续的散布边界之间向目的地传输的数据量配置在 DSRx.DSC 中。

将 CTLxL.DST_SCAT_EN 值为 1, 可以使能分发功能。CTLxL.DINC 决定每次地址变化时递增，递减还是保持不变。

下图以实例说明这一功能。

图 14.6: DMA 目的地分发传输



14.4.6　错误管理

如果 DMA 访问一段被保护或者被保留的地址空间，将会产生总线错误。当 DMA 读写操作中发生总线错误，硬件会立刻停止数据传输；禁止发生错误的 DMA 通道；如果中断允许且 Err 中断没有被屏蔽，则将产生 DMA 中断申请。

通道的缓存 (FIFO) 内的数据虽然没有被清除，但是不能继续使用。用户重新使能该通道开始新的 DMA 传输时，缓存钟的内容被覆盖。

发生传输错误的数据块不支持自动恢复，用户接到错误的中断后，必须重新使能该 DMA 通道，然后重新开始数据传输。

如果 DMA 传输中发生总线错误，用户接到中断后需要重新开启硬件握手接口的协议。外设需要先将 DMA 申请信号降低，然后 DMA 的通道打开之后，再将 DMA 申请拉高，开始新的握手协议。

14.4.7　提前结束 DMA 传输

在正常操作下，软件将 ChEnReg.CH_EN 写 1 来使能 DMA 通道，在传输结束时，硬件通过清除该寄存器来禁用 DMA 通道。如果用户希望提前结束 DMA 传输，推荐的方法是使用 CFGxL.CH_SUSP 和 CFGxL.FIFO_EMPTY。

1. 设置 CFGxL.CH_SUSP 为 1，停止来自源外设的所有传输。因此通道的 FIFO 不再接收新的数据；
2. 查询 CFGxL.FIFO_EMPTY 为 1，表示通道的 FIFO 已经为空，所有的数据已经写入目的地；

3. 写 ChEnReg.CH_EN 为 0，禁用该 DMA 通道。

在特殊情况下，如果源端每次传输的数据宽度比目的地端小 (CTLx.SRC_TR_WIDTH 比 CTLx.DST_TR_WIDTH)，将 CFGxL.CH_SUSP 写 1 之后，有可能出现 FIFO 不会为空的情况。因为 FIFO 内的数据不足以开始新的目的地传输。这时，用户可以将 CFGxL.CH_SUSP 写为 0，继续 DMA 传输以正常结束。

14.4.8　中断

每个通道有五个中断源：

- DMA 传输结束中断 (Tfr)
- 数据块传输结束中断 (Block)
- 源端传输结束中断 (SrcTran)
- 目的地端传输结束中断 (DstTran)
- 错误中断 (Err)

每个中断源有对应的原始状态寄存器，状态寄存器，屏蔽寄存器和清除寄存器。详见寄存器列表**??**。其中中断屏蔽寄存器和清除寄存器都有写保护，以防误操作。

每个通道的控制寄存器有中断允许配置：当 CTLxL.INT_EN 被置为 1 时，使能五种中断类型。

所有通道的中断状态被或在一起，存在中断状态寄存器中 (StatusInt)，方便用户查询。这五种中断源或在一起，产生 DMA 的中断申请信号。

14.4.9　DMA 通道配置

下面的例子具体说明如何配置 DMA 的通道以实现各种类型的数据传输。

1. 读取 ChEnReg，选择一个空闲的 DMA 通道；

2. 通过写入中断清除寄存器，清除之前 DMA 传输通道上的任何未决中断。读取中断原始状态寄存器和中断寄存器，确认所有的中断已经被清除；

3. 编写以下通道寄存器：

- 在 SARx 寄存器中写入源起始地址
- 在 DARx 寄存器中写入目的地起始地址
- 配置通道的控制寄存器和配置寄存器（详细请参考寄存器描述）

4. 配置完通道之后，将 ChEnReg 对应的 CH_EN 位写 1，使能 DMA 通道。同时需要使能 DMAC 模块 (DmaCfgReg.DMA_EN 位 1)。

5. 源外设发出请求之后，DMAC 开始读取源外设的数据寄存器，并且向源外设发出应答信号；目标外设发出请求之后，DMAC 写目标外设的数据寄存器，并且向目标外设发送应答信号。如果 DMA 传输的源或者目的地是存储器，也不需要握手协议，直接开始数据传输。

6. 传输结束后，硬件向 CPU 发出中断，清除通道使能位以禁用该通道。用户可以响应块完成或者传输完成中断，或者查询通道使能位 (ChEnReg.CH_EN) 位，直到它被硬件清除，以检测传输何时完成。

14.4.10　DMA 请求映射

每个 DMA 有 16 组硬件握手接口，与其他的外设相连。

模块名	接口数字	外设 DMA 申请
DMAC1	0	TIM1_CH1 或 TIM2_UP 或 TIM3_CH3
	1	TIM1_CH4 或 TIM1_TRIG 或 TIM1 COM 或 TIM4_CH2
	2	TIM1_UP 或 TIM2_CH1 或 TIM4_CH3
	3	TIM1_CH3 或 TIM3_CH1 或 TIM3_TRIG
	4	TIM2_CH3 或 TIM4_CH1
	5	TIM2_CH2 或 TIM2_CH4 或 TIM4_UP
	6	TIM3_CH4 或 TIM3_UP 或 TIM1 CH2
	7	QSPI RX
	8	QSPI TX
	9	SPIS1 RX
	10	SPIS1 TX
	11	UART1 RX
	12	UART1 TX
	13	ADC 规则通道转换
	14	ADC 注入通道转换
	15	-
DMAC2	0	SPIM2 RX
	1	SPIM2 TX
	2	SPIS2 RX
	3	SPIS2 TX
	4	UART2 RX
	5	UART2 TX
	6	UART3 RX
	7	UART3 TX
	8	I2C1 RX
	9	I2C1 TX
	10	I2C2 RX
	11	I2C2 TX
	12	-
	13	-
	14	-
	15	-

注：以下外设虽然有 *DMA* 申请信号，但是没有被接入 *DMA* 模块。所以他们不支持硬件接口。如果用户需要使用 *DMA* 功能，请使用软件接口。

- TIM2 的 COM 和 TRIG DMA 申请
- TIM3 的 CC2 和 COM DMA 申请
- TIM4 的 CC4, COM 和 TRIG DMA 申请

注：*ADC* 仅支持 *single DMA* 传输。

14.5 寄存器描述

14.5.1 通道源地址寄存器 (SARx) (x=0..2)

地址偏移量：x × 0x58
复位值：0x0000 0000

位	符号	访问	描述
31:0	SAR	RW	DMA 传输的当前源地址。每次从源外设读取数据之后，更新该寄存器。

14.5.2 通道目标地址寄存器 (DARx) (x=0..2)

地址偏移量：0x8 + x × 0x58
复位值：0x0000 0000

位	符号	访问	描述
31:0	DAR	RW	DMA 传输的当前目的地地址。每次向目的地外设写入数据之后，更新该寄存器。

14.5.3 通道控制寄存器低 (CTLLx) (x=0..2)

地址偏移量：0x18 + x × 0x58
复位值：0x0030 4801

位	符号	访问	描述
31:27	-	R	保留
26:25	SMS	RW	访问源外设时用到的主模块接口 0: 主模块接口 1 1: 主模块接口 2 2,3: 保留
24:23	DMS	RW	访问目标外设时用到的主模块接口 0: 主模块接口 1 1: 主模块接口 2 2,3: 保留
22:20	TT_FC	RW	传输类型和流程控制。具体请参考14.5.3
19	-	R	保留
18	DST_SCAT_EN	RW	目的地分发 (scatter) 使能。只有 DINC 为 0 或者 1 时，该位有用。
17	SRC_GAT_EN	RW	源端收集 (gather) 使能。只有 SINC 为 0 或者 1 时，该位有用。
16:14	SRC_MSIZE	RW	源端 burst transaction 的长度。每次源端发送 burst transaction 申请时，传输数据项的数目。每个数据项的宽度由 SRC_TR_WIDTH 决定。 0: 1 个数据项 1: 4 个数据项 其他：保留

13:11	DST_MSIZE	RW	目的地端 burst transaction 的长度。每次目的地端发送 burst transaction 申请时，传输数据项的数目。每个数据项的宽度由 DST_TR_WIDTH 决定。 0: 1 个数据项 1: 4 个数据项 其他：保留
10:9	SINC	RW	源端地址递增。 0：源端地址递增 1：源端地址递减 其他：源端地址不变
8:7	DINC	RW	目的地端地址递增。 0：目的地端地址递增 1：目的地端地址递减 其他：目的地端地址不变
6:4	SRC_TR_WIDTH	RW	源端传输 (transfer) 的数据宽度 0: 数据宽度是 8 位 1: 数据宽度是 16 位 2: 数据宽度是 32 位 其他: 保留
3:1	DST_TR_WIDTH	RW	目的地端传输 (transfer) 的数据宽度 0: 数据宽度是 8 位 1: 数据宽度是 16 位 2: 数据宽度是 32 位 其他: 保留
0	INT_EN	RW	中断使能位。如果置 1,5 个中断源全部使能。

TT_FC	传输类型	流程控制者
000	存储器到存储器	DMAC
001	存储器到外设	DMAC
010	外设到存储器	DMAC
011	外设到外设	DMAC
其他	保留	保留

14.5.4 通道控制寄存器高 (CTLHx) (x=0..2)

地址偏移量：0x1C + x × 0x58

复位值：0x0000 0002

位	符号	访问	描述
31:13	-	R	保留
12	DONE	RW	用户可以 polling 该位来判断块传输是否结束。
11:0	BLOCK_TS	RW	数据块传输的大小，不能超过 511。用户需要在开始 DMA 传输之前，配置该位来指定数据块的大小。单次传输的数据宽度由 SRC_TR_WIDTH。

14.5.5 通道配置寄存器低 (CFGLx) (x=0..2)

地址偏移量：0x40 + x × 0x58

复位值：0x00000C00 + x × 0x20

位	符号	访问	描述
31	RELOAD_DST	RW	目的地端地址寄存器自动重装载（仅通道 0 有此功能） 0: 禁用 DARx 寄存器自动重装载 1: 启用 DARx 寄存器自动重装载
30	RELOAD_SRC	RW	源端地址寄存器自动重装载（仅通道 0 有此功能） 0: 禁用 SARx 寄存器自动重装载 1: 启用 SARx 寄存器自动重装载
29:20	-	R	保留
19	SRC_HS_POL	RW	源端握手接口的极性 0: 高有效 1: 低有效
18	DST_HS_POL	RW	目的地端握手接口的极性 0: 高有效 1: 低有效
17:12	-	R	保留
11	HS_SEL_SRC	RW	源端握手接口选择 0: 选择硬件握手接口 1: 选择软件握手接口
10	HS_SEL_DST	RW	目的地端握手接口选择 0: 选择硬件握手接口 1: 选择软件握手接口
9	FIFO_EMPTY	R	FIFO 状态。 0: 通道 FIFO 没有空 1: 通道 FIFO 空
8	CH_SUSP	RW	通道挂起 0: 源端的 DMA 传输没有被挂起 1: 挂起源端的 DMA 传输
7:5	CH_PRIOR	RW	通道优先级选择。0 是最低优先级。
4:0	-	R	保留

14.5.6 通道配置寄存器高 (CFGHx) (x=0..2)

地址偏移量：0x44 + x × 0x58

复位值：0x0000 0002

位	符号	访问	描述
31:15	-	R	保留
14:11	DST_PER	RW	指定目的地端硬件握手接口 (1-16)
10:7	SRC_PER	RW	指定源端硬件握手接口 (1-16)
6:5	-	R	保留

4:2	PROTCTL	RW	AMBA 总线保护配置。HPROT[0] 总是 1。HPROT[3:1] 由 PROTCTL 配置。
1	FIFO_MODE	RW	FIFO 模式选择。
0	FCMODE	RW	流程控制模式选择端。该位仅在目标外设是流程控制者时有用。

14.5.7 通道收集寄存器 (SGR0)

地址偏移量：0x48

复位值：0x0000 0000

通道 1 和通道 2 没有该寄存器。

位	符号	访问	描述
31:20	SGC	RW	在连续收集时传输的数据量，以 SRC_TR_WIDTH 为单位。最大值是 511。
19:0	SGI	RW	连续收集间隔

14.5.8 通道分发寄存器 (DSR0)

地址偏移量：0x50

复位值：0x0000 0000

通道 1 和通道 2 没有该寄存器。

位	符号	访问	描述
31:20	DSC	RW	在连续的散布边界之间向目的地传输的数据量，以 SRC_TR_WIDTH 为单位。最大值是 511。
19:0	DSI	RW	目的地分发增量

14.5.9 中断原始状态寄存器 (RawTfr, RawBlock, RawSrcTran, RawDstTran, RawErr)

地址偏移量：0x2C0, 0x2C8, 0x2D0, 0x2D8, 0x2E0

复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留
2	RAW2	R	通道 2 中断的原始状态
1	RAW1	R	通道 1 中断的原始状态
0	RAW0	R	通道 0 中断的原始状态

14.5.10 中断状态寄存器 (StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr)

地址偏移量：0x2E8, 0x2F0, 0x2F8, 0x300, 0x308

复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留
2	STAT2	R	通道 2 中断状态

1	STAT1	R	通道 1 中断状态
0	STAT0	R	通道 0 中断状态

14.5.11 中断屏蔽寄存器 (MaskTfr, MaskBlock, MaskSrcTran, MaskDstTran, MaskErr)

地址偏移量：0x310, 0x318, 0x320, 0x328, 0x330
复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	MSK2_WE	RW	通道 2 中断屏蔽写允许。
9	MSK1_WE	RW	通道 1 中断屏蔽写允许。
8	MSK0_WE	RW	通道 0 中断屏蔽写允许。
7:3	-	R	保留
2	MSK2	RW	通道 2 中断屏蔽
1	MSK1	RW	通道 1 中断屏蔽
0	MSK0	RW	通道 0 中断屏蔽

14.5.12 中断清除寄存器 (ClrTfr, ClrBlock, ClrSrcTran, ClrDstTran, ClrErr)

地址偏移量：0x338, 0x340, 0x348, 0x350, 0x358
复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留
2	CLR2	W	通道 2 中断清除
1	CLR1	W	通道 1 中断清除
0	CLR0	W	通道 0 中断清除

14.5.13 中断寄存器 (StatusInt)

地址偏移量：0x360
复位值：0x0000 0000

位	符号	访问	描述
31:5	-	R	保留
4	ERR	R	三个通道的 Err 状态或
2	DSTT	R	三个通道的 DstTran 状态或
2	SRCT	R	三个通道的 SrcTran 状态或
1	BLOCK	R	三个通道的 Block 状态或
0	TFR	R	三个通道的 Trf 状态或

14.5.14 源端事务软件申请寄存器 (ReqSrcReg)

地址偏移量：0x368

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	SRC_REQ2	RW	通道 2 源端传输软件申请
1	SRC_REQ1	RW	通道 1 源端传输软件申请
0	SRC_REQ0	RW	通道 0 源端传输软件申请

14.5.15 目标端事务软件申请寄存器 (ReqDstReg)

地址偏移量：0x370

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	DST_REQ2	RW	通道 2 目标端传输软件申请
1	DST_REQ1	RW	通道 1 目标端传输软件申请
0	DST_REQ0	RW	通道 0 目标端传输软件申请

14.5.16 源端单次事务软件申请寄存器 (SglReqSrcReg)

地址偏移量：0x378

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	S_SG_REQ2	RW	通道 2 源端单次交易软件申请
1	S_SG_REQ1	RW	通道 1 源端单次交易软件申请
0	S_SG_REQ0	RW	通道 0 源端单次交易软件申请

14.5.17 目标端单次事务软件申请寄存器 (SglReqDstReg)

地址偏移量：0x380

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	D_SG_REQ2	RW	通道 2 目标端单次交易软件申请
1	D_SG_REQ1	RW	通道 1 目标端单次交易软件申请
0	D_SG_REQ0	RW	通道 0 目标端单次交易软件申请

14.5.18 源端最后一次事务软件申请寄存器 (LstSrcReg)

地址偏移量：0x378

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	LSTSRC2	RW	通道 2 源端最后一次交易软件申请
1	LSTSRC1	RW	通道 1 源端最后一次交易软件申请
0	LSTSRC0	RW	通道 0 源端最后一次交易软件申请

14.5.19 目标端最后一次事务软件申请寄存器 (LstDstReg)

地址偏移量：0x380

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	REQ_WE2	RW	通道 2 软件申请写允许。
9	REQ_WE1	RW	通道 1 软件申请写允许。
8	REQ_WE0	RW	通道 0 软件申请写允许。
7:3	-	R	保留
2	LSTDST2	RW	通道 2 目标端最后一次交易软件申请
1	LSTDST1	RW	通道 1 目标端最后一次交易软件申请
0	LSTDST0	RW	通道 0 目标端最后一次交易软件申请

14.5.20 DMA 配置寄存器 (DmaCfgReg)

地址偏移量：0x398

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	DMA_EN	RW	DMA 允许。高有效

14.5.21 通道允许寄存器 (ChEnReg)

地址偏移量：0x3A0

复位值：0x0000 0000

位	符号	访问	描述
31:11	-	R	保留
10	CHEN_WE2	RW	通道 2 写允许。
9	CHEN_WE1	RW	通道 1 写允许。
8	CHEN_WE0	RW	通道 0 写允许。
7:3	-	R	保留
2	CHEN2	RW	通道 2 允许
1	CHEN1	RW	通道 1 允许
0	CHEN0	RW	通道 0 允许

第十五章　模拟/数字转换 (ADC)

15.1　ADC 介绍

ADC 可以配置为 10 位或者 12 位，是一种逐次逼近型模拟数字转换器，转换率高达 1Msps。它有多达 18 个通道，可测量 16 个外部和 2 个内部信号源。各通道的 A/D 转换可以单次、连续、扫描或间断模式执行。ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阀值。

ADC 的输入时钟不得超过 16MHz，它是由 apb_clk 经分频产生。

15.2　ADC 主要特征

- ADC 可配置为 10 位或者 12 位分辨率
- 配置为 10 位分辨率时，支持 $1 \sim 4$ 个采样保持电路同时工作
- 配置为 12 位分辨率时，仅有一个采样保持电路工作
- 转换结束、注入转换结束、发生模拟看门狗事件、FIFO 溢出和 FIFO 空时产生中断
- 支持单次和连续转换模式
- 从通道 0 到通道 n 的自动扫描模式
- 自校准
- 带内嵌数据一致性的数据对齐
- 采样间隔可以按通道分别编程
- 规则转换和注入转换均有外部触发选项
- 间断模式
- ADC 转换时间：普通模式下
  - 10 位 ADC 的每次转换时间最快是 14 个 ADC 时钟周期
  - 12 位 ADC 的每次转换时间最快是 17 个 ADC 时钟周期
- ADC 供电要求: 2.4V 到 3.6V

- ADC 测量范围：0V ≤ VIN ≤ VDDA
- 规则通道转换支持 DMA。

15.3　ADC 功能描述

图15.1是 ADC 模块的框图，表15.1为 ADC 引脚的说明。

图 15.1: ADC 连接框图

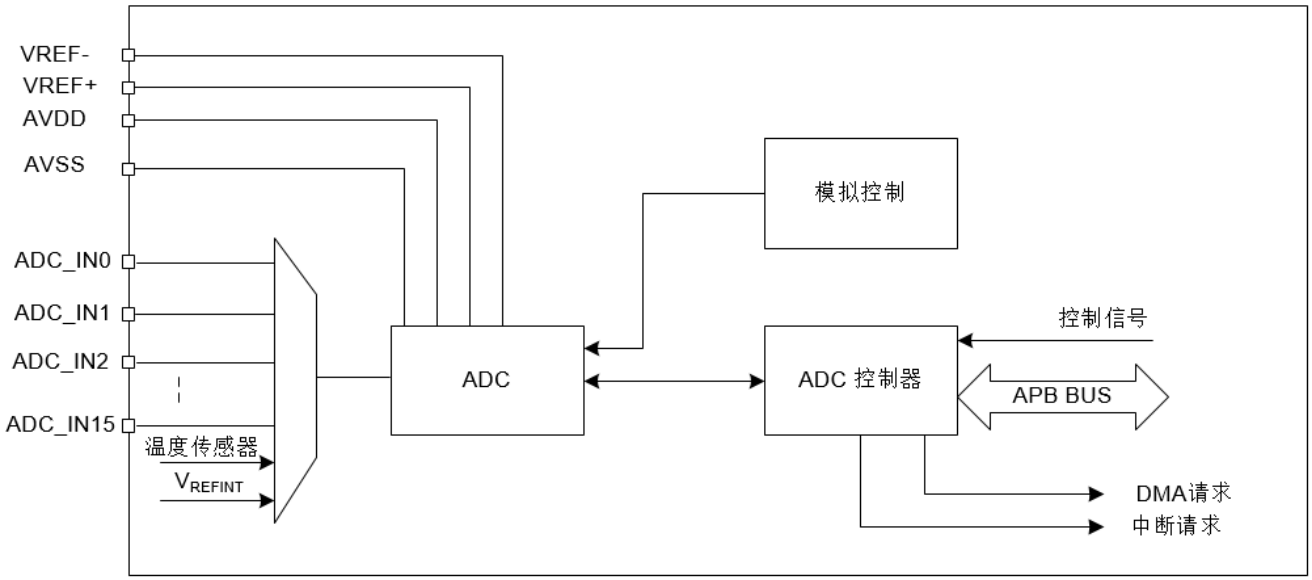


表 15.1: ADC 引脚

名称	信号类型	注解
VREF+	输入，模拟参考正极	ADC 使用的高端/正极参考电压，接在 AVDD 上
AVDD	输入，模拟电源	模拟电源，2.0V ≤ AVDD ≤ 3.6V
VREF-	输入，模拟参考负极	ADC 使用的低端/负极参考电压，接在 AVSS 上
AVSS	输入，模拟电源地	模拟电源地
ADC_IN[17:0]	模拟输入信号	18 个模拟输入通道
ADON	输入	ADC 使能信号，高有效
NPOR	输入	ADC 复位信号，低有效

[1] *ADC 在 ADON 为 1 而且 NPOR 为 1 时，正常工作。*

15.3.1　ADC 控制开关

使能 ADC 转换，需要分两步：

1. 使能 ADC 转换器: 通过设置 ANCTL 模块的 ADC_CTL_0.ADON 位可以使能 ADC(请参考十二)。如果需要 ADC 停止转换，将该位清零。

2. 使能 ADC 控制器: 系统上电之后，第一次设置 ADC_CR2 的 ADON 位使能 ADC 控制模块，再次设置 ADON 位时开始进行转换。

15.3.2　ADC 时钟

ADC 控制器模块内有一个专用的可编程预分频器，用来产生 ADC 的时钟。该时钟与控制器的时钟同步。

15.3.3　精度选择

ADC 的精度可以配置为 12 位或者 10 位。

15.3.4　通道选择

有 16 个多路通道。可以把转换组织成两组：规则组和注入组。在任意多个通道上以任意顺序进行的一系列转换构成成组转换。例如，可以如下顺序完成转换：通道 3、通道 8、通道 2、通道 2、通道 0、通道 2、通道 2、通道 15。

- 规则组由多达 16 个转换组成。规则通道和它们的转换顺序在 ADC_SQRx 寄存器中选择。规则组中转换的总数应写入 ADC_SQR1 寄存器的 L[3:0] 位中。
- 注入组由多达 4 个转换组成。注入通道和它们的转换顺序在 ADC_JSQR 寄存器中选择。注入组里的转换总数目应写入 ADC_JSQR 寄存器的 L[1:0] 位中。

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

温度传感器/ VREFINT 内部通道

温度传感器和通道 ADC1_IN16 相连接，内部参照电压 VREFINT 和 ADC1_IN17 相连接。可以按注入或规则通道对这两个内部通道进行转换。

15.3.5　单次转换模式

单次转换模式下，ADC 只执行一次转换。该模式既可通过设置 ADC_CR2 寄存器的 ADON 位 (只适用于规则通道) 启动也可通过外部触发启动 (适用于规则通道或注入通道)，这时 CONT 位为 0。一旦选择通道的转换完成：

- 如果一个规则通道被转换：
  - 转换数据被储存在 16 位 ADC_DR 寄存器中
  - EOC(转换结束) 标志被设置
  - 如果设置了 EOCIE，则产生中断。
  - 然后 ADC 停止。
- 如果一个注入通道被转换：
  - 转换数据被储存在 16 位的 ADC_DRJ1 寄存器中
  - JEOC(注入转换结束) 标志被设置
  - 如果设置了 JEOCIE 位，则产生中断。
  - 然后 ADC 停止。

15.3.6　连续转换模式

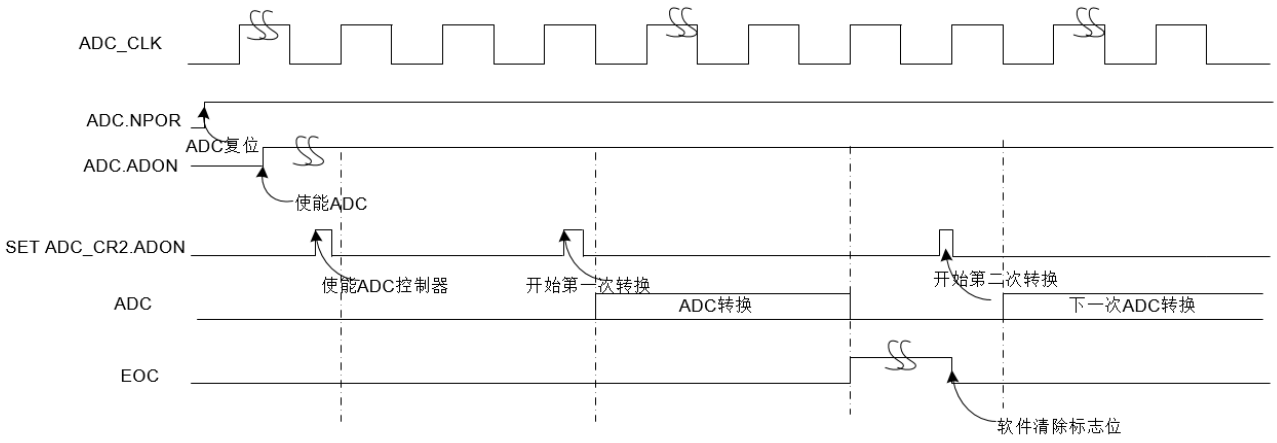
在连续转换模式中，当前的 ADC 转换一结束马上就启动另一次转换。此模式可通过外部触发启动或通过设置 ADC_CR2 寄存器上的 ADON 位启动，此时 CONT 位是 1。每个转换后：

- 如果一个规则通道被转换：
  - 转换数据被储存在 16 位的 ADC_DR 寄存器中
  - EOC(转换结束) 标志被设置
  - 如果设置了 EOCIE，则产生中断。
- 如果一个注入通道被转换：
  - 转换数据被储存在 16 位的 ADC_DRJ1 寄存器中
  - JEOC(注入转换结束) 标志被设置
  - 如果设置了 JEOCIE 位，则产生中断。

15.3.7　时序图

如图15.2所示，ADC 使能之后，开始 ADC 转换。14/17 个时钟周期之后，ECO 标志被设置，10/12 位 ADC 转换结果存在 ADC 数据寄存器中。

图 15.2: ADC 时序图



15.3.8　模拟看门狗

如果被 ADC 转换的模拟电压低于低阀值或高于高阀值，AWD 模拟看门狗状态位被设置。阀值位于 ADC_HTR 和 ADC_LTR 寄存器的低 10/12 个有效位中。通过设置 ADC_CR1 寄存器的 AWDIE 位以允许产生相应中断。

阀值独立于由 ADC_CR2 寄存器上的 ALIGN 位选择的数据对齐模式。比较是在对齐之前完成的。

通过配置 ADC_CR1 寄存器，模拟看门狗可以作用于 1 个或多个通道，如表15.2所示。

图 15.3: ADC 模拟看门狗警戒区

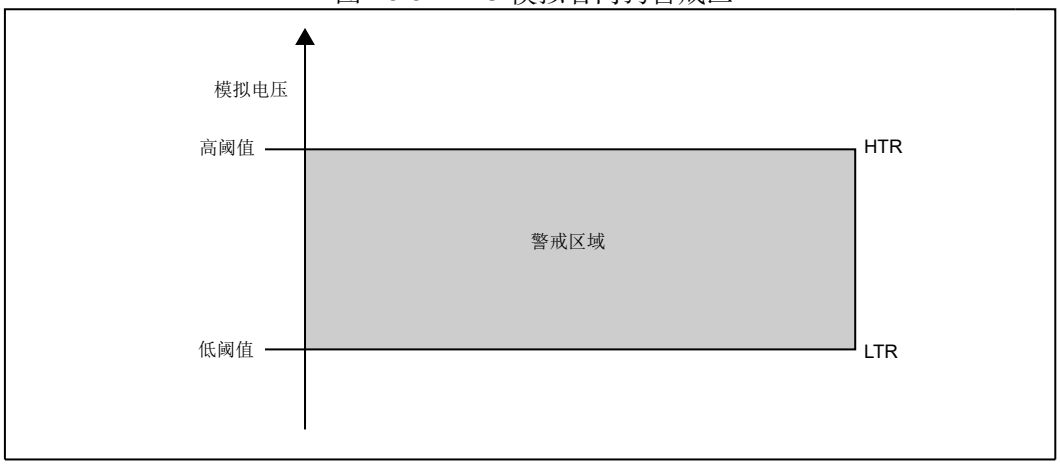


表 15.2: ADC 模拟看门狗通道选择

模拟看门狗警戒的通道	AWDSGL	AWDEN	JAWDEN
所有注入通道	0	0	1
所有规则通道	0	1	0
所有注入和规则通道	0	1	1
单一的注入通道 (1)	1	0	1
单一的规则通道 (1)	1	1	0
单一的注入和规则通道（1）	1	1	1

[1] *(1) 由 AWDCH[4:0] 位选择*

15.3.9　扫描模式

此模式用来扫描一组模拟通道。

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

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

15.3.10　注入通道管理

触发注入

清除 ADC_CR1 寄存器的 JAUTO 位，并且设置 SCAN 位，即可使用触发注入功能。

- 1. 利用外部触发或通过设置 ADC_CR2 寄存器的 ADON 位，启动一组规则通道的转换。
- 2. 如果在规则通道转换期间产生一外部触发注入，当前转换被暂停，注入通道序列被以单次扫描方式进行转换。

- 3. 然后，恢复上次被中断的规则组通道转换。如果在注入转换期间产生一规则事件，注入转换不会被中断，但是规则序列将在注入序列结束后被执行。

注意：当使用触发的注入转换时，必须保证触发事件的间隔长于注入序列。例如：注入序列长度为 *28* 个 *ADC* 时钟周期，每次触发之间的最小间隔是 *29* 个 *ADC* 时钟周期。

自动注入

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

在此模式里，必须禁止注入通道的外部触发。

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

注意：不能同时使用自动注入和间断模式。

15.3.11　间断模式

规则组

此模式通过设置 ADC_CR1 寄存器上的 DISCEN 位激活。它可以用来执行一个短序列的 n 次转换 (n≤8)，此转换是 ADC_SQRx 寄存器所选择的转换序列的一部分。数值 n 由 ADC_CR1 寄存器的 DISCNUM[2:0] 位给出。

一个外部触发信号可以启动 ADC_SQRx 寄存器中描述的下一轮 n 次转换，直到此序列所有的转换完成为止。总的序列长度由 ADC_SQR1 寄存器的 L[3:0] 定义

举例：

n=3，被转换的通道 = 0、1、2、3、6、7、9、10

第一次触发：转换的序列为 0、1、2

第二次触发：转换的序列为 3、6、7

第三次触发：转换的序列为 9、10，并产生 EOC 事件

第四次触发：转换的序列 0、1、2

注意：

1. 当以间断模式转换一个规则组时，转换序列结束后不自动从头开始。

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

注入组

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

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

例子：

n=1，被转换的通道 = 1、2、3

第一次触发：通道 1 被转换

第二次触发：通道 2 被转换

第三次触发：通道 3 被转换，并且产生 EOC 和 JEOC 事件

第四次触发：通道 1 被转换

注意：

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

1 个注入通道 *1* 。

2. 不能同时使用自动注入和间断模式。

3. 必须避免同时为规则组和注入组设置间断模式。间断模式只能用于一组转换。

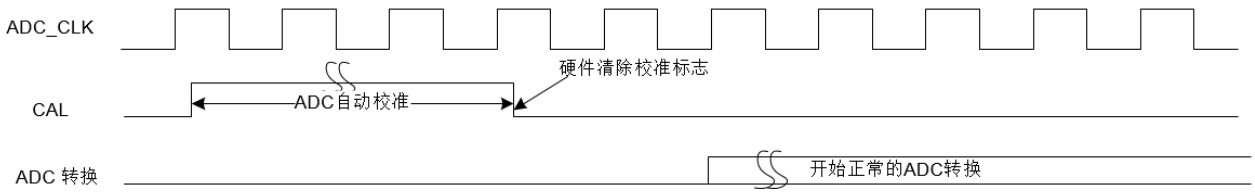
15.4　校准

ADC 有一个内置自校准模式。校准可大幅减小因内部电容器组的变化而造成的准精度误差。在校准期间，在每个电容器上都会计算出一个误差修正码 (数字值)，这个码用于消除在随后的转换中每个电容器上产生的误差。

通过设置 ADC_CR2 寄存器的 CAL 位启动校准。一旦校准结束，CAL 位被硬件复位，可以开始正常转换。建议在上电时执行一次 ADC 校准。校准阶段结束后，校准码储存在 ADC_CALL 和 ADC_CALH 中。

注意：建议在每次上电后执行一次校准。

图 15.4: ADC 校准时序图



15.5　数据对齐

ADC_CR2 寄存器中的 ALIGN 位选择转换后数据储存的对齐方式。数据可以左对齐或右对齐，如图15.5、图15.6、图15.7 和图15.8所示。

注入组通道转换的数据值已经减去了在 ADC_JOFRx 寄存器中定义的偏移量，因此结果可以是一个负值。SEXT 位是扩展的符号值。

对于规则组通道，不需减去偏移值，因此只有 10 或者 12 个位有效。

图 15.5: ADC 数据左对齐（10 位）

注入组

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

规则组

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

图 15.6: ADC 数据左对齐（12 位)

注入组

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

规则组

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

图 15.7: ADC 数据右对齐（10 位）

注入组

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

规则组

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

图 15.8: ADC 数据右对齐（12 位）

注入组

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

规则组

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

15.6　可编程的通道采样时间

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

总转换时间如下计算：T_{CONV} = 采样时间 + 12.5个周期

15.7　外部触发转换

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

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

表 15.3: 用于规则通道的外部触发

触发源	类型	EXTSEL[2:0]
TIM1_CC1 事件	定时器 TIM1 的中断信号	000
TIM1_CC2 事件	定时器 TIM1 的中断信号	001
TIM1_CC3 事件	定时器 TIM1 的中断信号	010
TIM2_CC2 事件	定时器 TIM2 的中断信号	011
TIM3_TRGO 事件	定时器 TIM3 的中断信号	100
TIM4_CC4 事件	定时器 TIM4 的中断信号	101
EXTI11	外部引脚	110
SWSTART	软件控制位	111

表 15.4: 用于注入通道的外部触发

触发源	类型	JEXTSEL[2:0]
TIM1_TRGO 事件	定时器 TIM1 的中断信号	000
TIM1_CC4 事件	定时器 TIM1 的中断信号	001
TIM2_TRGO 事件	定时器 TIM2 的中断信号	010
TIM2_CC1 事件	定时器 TIM2 的中断信号	011
TIM3_CC4 事件	定时器 TIM3 的中断信号	100
TIM4_TRGO 事件	定时器 TIM4 的中断信号	101
EXTI15	外部引脚	110
JSWSTART	软件控制位	111

15.8　DMA 请求

规则组 DMA 请求

因为规则通道转换的值存储在一个深度是 4 的 FIFO 中，所以当转换多个规则通道 (序列长度超过 4）时，需要使用 DMA，避免丢失数据。DMA 读取时的寄存器是 ADC_DR。

在规则通道的一次转换结束时（FIFO 不为空）产生 DMA 请求，并将转换的数据从 ADC_DR 寄存器传输到用户指定的目标地址。

注入组 DMA 请求

注入组转换全部结束之后，开始产生 DMA 请求。DMA 读取寄存器 ADC_JDMAR 得到数据。

15.9　温度传感器

温度传感器可以用来测量器件周围的温度（TA）。温度传感器在内部和 ADC1_IN16 输入通道相连接，此通道把传感器输出的电压转换成数字值。温度传感器模拟输入推荐采样时间是 17.1μs（TBD）。

必须同时设置模拟寄存器模块的 BGCR2.TEMPOUTEN 为 1，以打开温度传感器。温度传感器输出电压随温度线性变化，由于生产过程的变化，温度变化曲线的偏移在不同芯片上会有不同 (最多相差 45°C TBD)。

内部温度传感器更适合于检测温度的变化，而不是测量绝对的温度。如果需要测量精确的温度，应该使用一个外置的温度传感器。

使用温度传感器的步骤如下：

1. 选择 ADC1_IN16 输入通道
2. 选择采样时间为 17.1us （TBD）
3. 设置 ADC 控制寄存器 2(ADC_CR2) 的 TSVREFE 位，以唤醒关电模式下的温度传感器
4. 设置模拟寄存器模块 BGCR2.TEMPOUTEN 为 1
5. 利用下列公式得出温度:

$$温度\,(°C) = \{(V_{25} - V_{SENSE})\,/\text{Avg\_Slope}\} + 25$$

这里：

$V_{25} = V_{SENSE}$ 在 25°C 时的数值。

Avg_Slope= 温度与 V_{SENSE} 曲线的平均斜率（单位为 $mV/°C$ 或 $\mu V/°C$)

注意：传感器从关电模式唤醒后到可以输出正确水平的 *VSENSE* 前，有一个建立时间。*ADC* 在上电后也有一个建立时间，因此为了缩短延时，建议同时打开 *ADC* 和温度传感器。

15.10　高级工作模式

在高级工作模式下，WB32F10xxx 支持对规则转换进行多通道同时转换。使用该模式时需要注意：

- SARADC 必须工作在 10 位宽精度模式。12 位宽精度时，该模式不能使用。
- SARADC 的四个采样保持电路有三种配置模式：
  - 采样保持电路 0 和 A 同时工作
  - 采样保持电路 0，A 和 B 同时工作
  - 采样保持电路 0，A，B 和 C 同时工作
- SARADC 只有一个转换通道，在高级工作模式下，对各采样保持电路的结果依次转换。
- 在该模式时，规则转换的序列长度必须是 N_{sh} 的整数倍。N_{sh} 是同时工作的采样保持电路的个数。
- 在该模式时，规则转换的序列在各个通道顺序排列。例如：依次转换的序列顺序是 AN8=> AN0=> AN1=> AN2=> AN9=> AN3=> AN4=> AN5, 四个采样保持电路同时工作，那么 AN8 和 AN9 使用采样保持电路 0，AN0 和 AN3 使用采样保持电路 A，AN1 和 AN4 使用采样保持电路 B，AN2 和 AN5 使用采样保持电路 C。图15.9和15.10详细解释这一执行过程。
- 在该模式时，对转换通道的选择有一定的限制。采样保持电路 0 可以转换所有的通道，A 只能转换通道 AN0 和 AN3，B 只能转换通道 AN1 和 AN4，C 只能转换通道 AN2 和 AN5。详情参考**??**

- 在高级工作模式下，间断模式仍然有效 (ADC_CR1 的 DISCEN 位)。与普通模式的区别是，在一次触发事件发生时，N_{sh} 个序列同时开始转换。
- 如果在规则转换期间发生了注入转换，当前的规则转换全部暂停，优先响应注入转换。当注入转换结束之后，再恢复之前的 N_{sh} 个规则转换。注入转换使用采样保持电路 0，所以恢复规则转换时，需要额外的时间来重新对转换的通道进行采样。
- 在高级工作模式下，由于多个采样保持电路同时工作，大大提高了 SARADC 的转换效率，用户需要及时读出转换结果，否则新的数据将会丢失。
- 各序列的转换结果存在 FIFO 中，FIFO 的深度是 4。存储结果的序列顺序同 ADC_SQ。

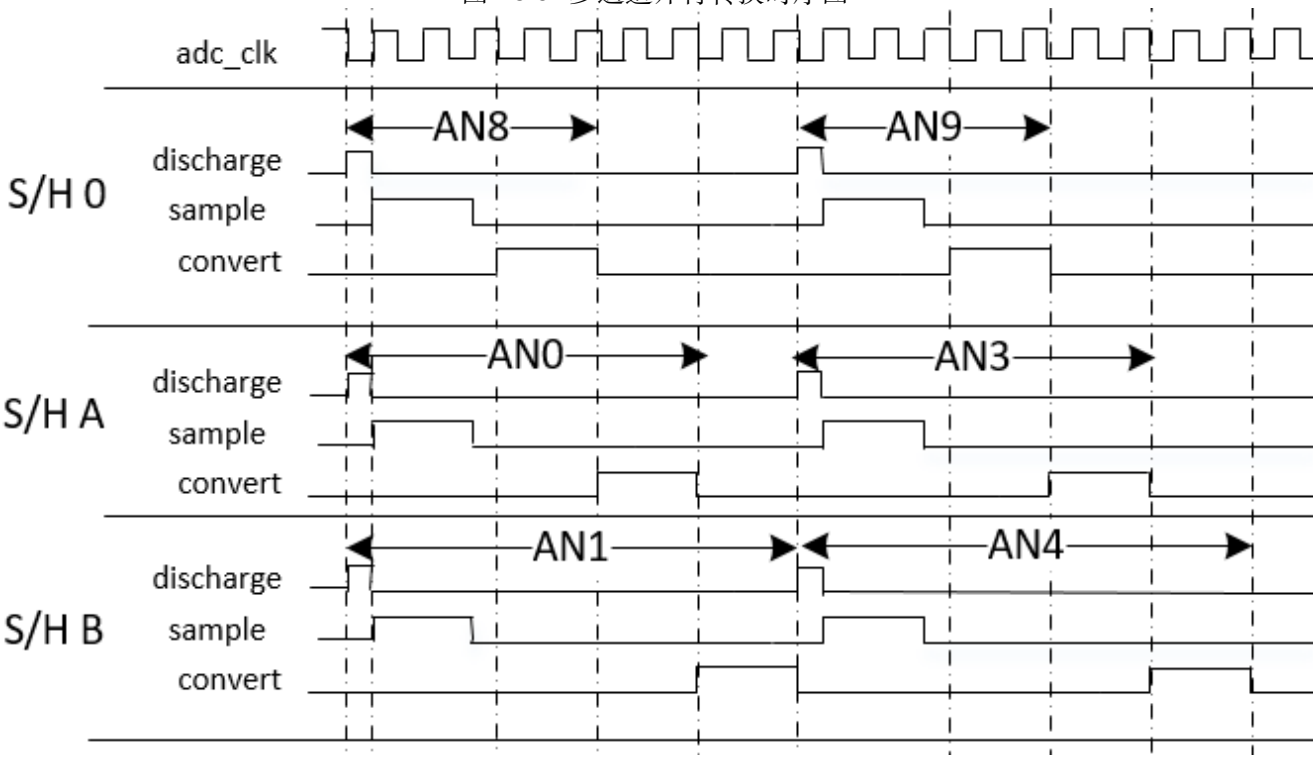
在高级工作模式下，由于多个采样保持电路的工作方式不同，可分为两个模式：多通道并行转换和多通道顺序转换。

15.10.1　多通道并行转换

在多通道并行转换时，多个采样保持电路同时工作，采样时间由各通道中的最大采样时间决定。采样时间结束之后，依次开始转换各采样保持电路的结果。

在这种模式下，节省了通道的采样时间，大大提高了 ADC 的转换效率。图15.9显示了这一模式下的时序。在这个例子中，采样保持电路 0，A 和 B 同时工作，N_{sh} 为 3，规则转换序列为 6, 转换序列为：AN8=> AN0=> AN1=> AN9=> AN3=> AN4。

图 15.9: 多通道并行转换时序图



15.10.2　多通道顺序转换

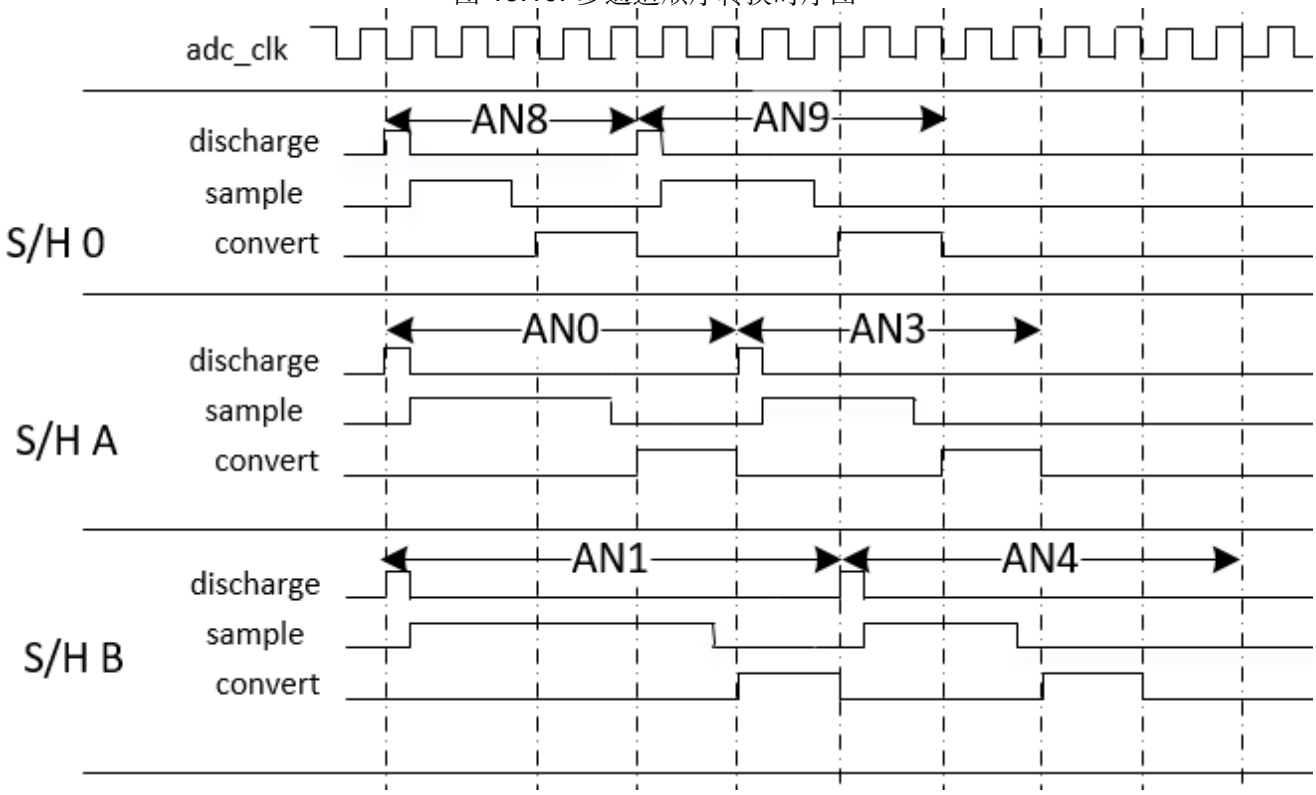
在多通道并行转换时，同样是同时启动多个采样保持电路同时工作。区别是：

- 采样时间由寄存器配置；
- 当采样保持电路 0 结束之后，立即开始对其结果进行转换。同时其他的采样保持电路继续工作。
- 对采样保持电路 0 的转换结束之后，开始对采样保持电路 A 的结果进行转换，同时采样保持电路 0 开始对下个通道进行采样。
- 然后依次顺序转换下去。

这这种模式下，对各采样保持电路的转换结束之后，马上开始对下一个通道的放电和采样，不用等到所有的转换结束之后。这样进一步节省了各通道的采样时间，提高了 ADC 的转换效率。

图15.10显示了这一模式下的时序。在这个例子中，采样保持电路 0，A 和 B 同时工作，N_{sh} 为 3，规则转换序列为 6, 转换序列为：AN8=> AN0=> AN1=> AN9=> AN3=> AN4。

图 15.10: 多通道顺序转换时序图



15.11　ADC 中断

规则和注入组转换结束时能产生中断，当模拟看门狗状态位被设置时也能产生中断。它们都有独立的中断使能位。

规则通道转换结束时，数据存在 FIFO 中。当 FIFO 为空或者溢出时会产生中断。FIFO 空和 FIFO 溢出也有独立的中断使能位。当 FIFO 非空时，如果 DMA 使能，会发出 DMA 请求。建议检查到 FIFO 非空时，读取 ADC_DR 得到规则通道的转换结果。FIFO 溢出时，说明存在转换结果丢失。用户可以在中断程序中做一些处理或者提高读取数据的速度以避免这种情况发生。

ADC_SR 寄存器中有 2 个其他标志，不产生中断：

- JSTRT(注入组通道转换的启动)

- STRT(规则组通道转换的启动)

表 15.5: ADC 中断

中断事件	事件标志	使能控制位
规则组 FIFO 溢出	OVF	OVFIE
规则组 FIFO 空	EMPF	EMPIE
规则组转换结束	EOC	EOCIE
注入组转换结束	JEOC	JEOCIE
模拟看门狗中断	AWD	AWDIE

15.12　ADC 寄存器

15.12.1　ADC 状态寄存器 (ADC_SR)

地址偏移：0x00

复位值：0x0000_0000

位	符号	访问	描述
31:15	-	R	保留。必须保持位 0
6	OVF	RC_W0	FIFO 溢出标志位 (FIFO overflow flag) 由硬件在规则通道 FIFO 溢出时设置，由硬件或者软件清除。 0：规则通道 FIFO 没有溢出； 1：规则通道 FIFO 溢出
5	EMPF	RC_W0	FIFO 空标志位 (FIFO empty flag) 由硬件在规则通道 FIFO 为空时设置，由硬件或者软件清除。 0：规则通道 FIFO 内有未读走的数据； 1：规则通道 FIFO 空
4	STRT	RC_W0	规则通道开始位 (Regular channel Start flag) 由硬件在规则通道转换开始时设置，由软件清除。 0：规则通道转换未开始； 1：规则通道转换已开始。
3	JSTRT	RC_W0	注入通道开始位 (Injected channel Start flag) 该位由硬件在注入通道组转换开始时设置，由软件清除。 0：注入通道组转换未开始； 1：注入通道组转换已开始。
2	JEOC	RC_W0	注入通道转换结束位 (Injected channel end of conversion) 该位由硬件在所有注入通道组转换结束时设置，由软件清除 0：转换未完成； 1: 转换完成。

1	EOC	RC_W0	规则通道转换结束位 (Regular channel end of conversion) 该位由硬件在规则通道或者注入通道转换结束时设置，由软件清除或者读取 ADC_DR 清除。 0：转换未完成； 1: 转换完成。
0	AWD	RC_W0	模拟看门狗标志位 (Analog watchdog flag) 该位由硬件在转换的电压值超出了 ADC_LTR 和 ADC_HTR 寄存器定义的范围时设置，由软件清除 0：没有发生模拟看门狗事件； 1：发生模拟看门狗事件。

15.12.2　ADC 控制寄存器 1(ADC_CR1)

地址偏移：0x04

复位值：0x0000_0000

位	符号	访问	描述
31:24	-	R	保留。必须保持为 0。
23	AWDEN	RW	在规则通道上开启模拟看门狗 (Analog watchdog enable on regular channels) 该位由软件设置和清除。 0：在规则通道上禁用模拟看门狗； 1：在规则通道上使用模拟看门狗。
22	JAWDEN	RW	在注入通道上开启模拟看门狗 (Analog watchdog enable on injected channels) 该位由软件设置和清除。 0：在注入通道上禁用模拟看门狗； 1：在注入通道上使用模拟看门狗。
21:16	-	R	保留。必须保持为 0。
15:13	DISCNUM[2:0]	RW	间断模式通道计数 (Discontinuous mode channel count) 软件通过这些位定义在间断模式下，收到外部触发后转换规则通道的数目 000：1 个通道 001：2 个通道 ... 111：8 个通道
12	JDISCEN	RW	在注入通道上的间断模式 (Discontinuous mode on injected channels) 该位由软件设置和清除，用于开启或关闭注入通道组上的间断模式 0：注入通道组上禁用间断模式； 1：注入通道组上使用间断模式。
11	DISCEN	RW	在规则通道上的间断模式 (Discontinuous mode on regular channels) 该位由软件设置和清除，用于开启或关闭规则通道组上的间断模式 0：规则通道组上禁用间断模式； 1：规则通道组上使用间断模式。

10	JAUTO	RW	自动的注入通道组转换 (Automatic Injected Group conversion) 该位由软件设置和清除，用于开启或关闭规则通道组转换结束后自动的注入通道组转换。 0：关闭自动的注入通道组转换； 1：开启自动的注入通道组转换。
9	AWDSGL	RW	在一个单一的通道上使用看门狗 (Enable the watchdog on a single channel in scan mode) 该位由软件设置和清除，用于开启或关闭由 AWDCH[4:0] 位指定的通道上的模拟看门狗功能 0：在所有的通道上使用模拟看门狗； 1：在单一通道上使用模拟看门狗。
8	SCAN	RW	扫描模式 (Scan mode) 该位由软件设置和清除，用于开启或关闭扫描模式。在扫描模式中，转换由 ADC_SQRx 或 ADC_JSQRx 寄存器选中的通道。 0：关闭扫描模式； 1：使能扫描模式。 注：如果分别设置了 EOCIE 或 JEOCIE 位，只在最后一个通道转换完毕后才会产生 EOC 或 JEOC 中断。
7	JEOCIE	RW	允许产生注入通道转换结束中断 (Interrupt enable for injected channels) 该位由软件设置和清除，用于禁止或允许所有注入通道转换结束后产生中断。 0：禁止 JEOC 中断； 1：允许 JEOC 中断。当硬件设置 JEOC 位时产生中断。
6	AWDIE	RW	允许产生模拟看门狗中断 (Analog watchdog interrupt enable) 该位由软件设置和清除，用于禁止或允许模拟看门狗产生中断。 0：禁止模拟看门狗中断； 1：允许模拟看门狗中断。
5	EOCIE	RW	允许产生 EOC 中断 (Interrupt enable for EOC) 该位由软件设置和清除，用于禁止或允许转换结束后产生中断。 0：禁止 EOC 中断； 1：允许 EOC 中断。当硬件设置 EOC 位时产生中断。
4:0	AWDCH[4:0]	RW	模拟看门狗通道选择位 (Analog watchdog channel select bits) 这些位由软件设置和清除，用于选择模拟看门狗保护的输入通道。 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 ..... 01111：ADC 模拟输入通道 15 10000：ADC 模拟输入通道 16 10001：ADC 模拟输入通道 17 保留所有其他数值。

15.12.3　ADC 控制寄存器 2(ADC_CR2)

地址偏移：0x08

复位值：0x0000_0000

位	符号	访问	描述
31:24	-	R	保留。必须保持为 0。
23	TSVREFE	RW	温度传感器和 VREFINT 使能 (Temperature sensor and VREFINT enable) 该位由软件设置和清除，用于开启或禁止温度传感器和 VREFINT 通道。 0：禁止温度传感器和 VREFINT 通道； 1：启用温度传感器和 VREFINT 通道。
22	SWSTART	RW	开始转换规则通道 (Start conversion of regular channels) 由软件设置该位以启动转换，转换开始后硬件马上清除此位。如果在 EXTSEL[2:0] 位中选择了 SWSTART 为触发事件，该位用于启动一组规则通道的转换。 0：复位状态； 1：开始转换规则通道。
21	JSWSTART	RW	开始转换注入通道 (Start conversion of injected channels) 由软件设置该位以启动转换，软件可清除此位或在转换开始后硬件马上清除此位。如果在 JEXTSEL[2:0] 位中选择了 JSWSTART 为触发事件，该位用于启动一组注入通道的转换。 0：复位状态； 1: 开始转换注入通道。
20	EXTTRIG	RW	规则通道的外部触发转换模式 (External trigger conversion mode for regular channels) 该位由软件设置和清除，用于开启或禁止可以启动规则通道组转换的外部触发事件。 0：不用外部事件启动转换； 1：使用外部事件启动转换。
19:17	EXTSEL[2:0]	RW	选择启动规则通道组转换的外部事件 (External event select for regular group) 这些位选择用于启动规则通道组转换的外部事件 触发事件配置如下： 000：定时器 1 的 CC1 事件　　001：定时器 1 的 CC2 事件 010：定时器 1 的 CC3 事件　　011：定时器 2 的 CC2 事件 100：定时器 3 的 TRGO 事件　　101：定时器 4 的 CC4 事件 110：EXTI11　　111：SWSTART
16	-	R	保留。必须保持为 0。
15	JEXTTRIG	RW	注入通道的外部触发转换模式 (External trigger conversion mode for injected channels) 该位由软件设置和清除，用于开启或禁止可以启动注入通道组转换的外部触发事件。 0：不用外部事件启动转换； 1：使用外部事件启动转换。

14:12	JEXTSEL[2:0]	RW	选择启动注入通道组转换的外部事件 (External event select for injected group) 这些位选择用于启动注入通道组转换的外部事件。 触发事件配置如下： 000：定时器 1 的 TRGO 事件　　001：定时器 1 的 CC4 事件 010：定时器 2 的 TRGO 事件　　011：定时器 2 的 CC1 事件 100：定时器 3 的 CC4 事件　　101：定时器 4 的 TRGO 事件 110：EXTI15　　111：JSWSTART
11	ALIGN	RW	数据对齐 (Data alignment) 该位由软件设置和清除。参考图15.5、图15.6、图15.7 和图15.8所示。 0：右对齐； 1：左对齐。
10	-	R	保留。必须保持为 0。
9	JDMAEN	RW	注入组 DMA 模式 该位由软件设置和清除。详见 DMA 控制器章节。 0：不使用 DMA 模式； 1：使用 DMA 模式。
8	DMAEN	RW	规则组 DMA 模式 该位由软件设置和清除。详见 DMA 控制器章节。 0：不使用 DMA 模式； 1：使用 DMA 模式。
7:4	-	R	保留。必须保持为 0。
3	RSTCAL	RW	复位校准 (Reset calibration) 该位由软件设置并由硬件清除。在校准寄存器被初始化后该位将被清除。 0：校准寄存器已初始化； 1：初始化校准寄存器。 注：如果正在进行转换时设置 RSTCAL，清除校准寄存器需要额外的周期。
2	CAL	RW	A/D 校准 (A/D Calibration) 该位由软件设置以开始校准，并在校准结束时由硬件清除。 0：校准完成； 1：开始校准。
1	CONT	RW	连续转换 (Continuous conversion) 该位由软件设置和清除。如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式； 1：连续转换模式。
0	ADON	RW	开/关 ADC 控制器。 该位由软件设置和清除。当该位为 0 时，写入 1 将使能 ADC 控制器。当该位为 1 时写入 1 将启动转换。 0：关闭 ADC 控制器； 1: 使能 ADC 控制器并启动转换； 注：如果在这个寄存器中与 ADON 一起还有其他位被改变，则转换不被触发。这是为了防止触发错误的转换。

15.12.4　ADC 采样时间寄存器 1(ADC_SMPR1)

地址偏移：0x0C
复位值：0x0000_0000

位	符号	访问	描述
31:24	-	R	保留。必须保持为 0。
23:0	SMPx[2:0]	RW	选择通道 x 的采样时间 (Channel x Sample time selection) 这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。 具体配置信息请参考表15.10。

表 15.10: SMPx 具体配置

SMPx	10 位	12 位
000	2	3
001	7	8
010	13	14
011	28	29
100	42	43
101	56	57
110	72	73
111	240	241

15.12.5　ADC 采样时间寄存器 2(ADC_SMPR2)

地址偏移：0x10
复位值：0x0000_0000

位	符号	访问	描述
31:30	-	R	保留。必须保持为 0。
29:0	SMPx[2:0]	RW	选择通道 x 的采样时间 (Channel x Sample time selection) 这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。 具体配置信息请参考表15.10。

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

地址偏移：0x14 - 0x20
复位值：0x0000_0000

位	符号	访问	描述
31:12	-	R	保留。必须为 0。

11:0	JOFFSETx[11:0]	RW	注入通道 x 的数据偏移 (Data offset for injected channel x) 当转换注入通道时，这些位定义了用于从原始转换数据中减去的数值。转换的结果可以在 ADC_JDRx 寄存器中读出。

15.12.7　ADC 看门狗高阈值寄存器 (ADC_HTR)

地址偏移：0x24

复位值：0x0000_0000

位	符号	访问	描述
31:12	-	R	保留。必须为 0。
11:0	HT[10:0]	RW	模拟看门狗高阀值 (Analog watchdog high threshold) 这些位定义了模拟看门狗的阀值高限。

15.12.8　ADC 看门狗低阈值寄存器 (ADC_LTR)

地址偏移：0x28

复位值：0x0000_0000

位	符号	访问	描述
31:12	-	R	保留。必须为 0。
11:0	LT[10:0]	RW	模拟看门狗低阀值 (Analog watchdog low threshold) 这些位定义了模拟看门狗的阀值低限。

15.12.9　ADC 规则序列寄存器 1 (ADC_SQR1)

地址偏移：0x2C

复位值：0x0000_0000

位	符号	访问	描述
31:24	-	R	保留。必须为 0。
23:20	L[3:0]	RW	规则通道序列长度 (Regular channel sequence length) 这些位由软件定义在规则通道转换序列中的通道数目。 0000: 1 个转换 0001：2 个转换 ..... 1111：16 个转换
19:15	SQ16[4:0]	RW	规则序列中的第 16 个转换 (16th conversion in regular sequence) 这些位由软件定义转换序列中的第 16 个转换通道的编号 (0 17)。
14:10	SQ15[4:0]	RW	规则序列中的第 15 个转换 (15th conversion in regular sequence)
9:5	SQ14[4:0]	RW	规则序列中的第 14 个转换 (14th conversion in regular sequence)
4:0	SQ13[4:0]	RW	规则序列中的第 13 个转换 (13th conversion in regular sequence)

15.12.10　ADC 规则序列寄存器 2(ADC_SQR2)

地址偏移：0x30

复值：0x0000_0000

位	符号	访问	描述
31:30	-	R	保留。必须为 0。
29:25	SQ12[4:0]	RW	规则序列中的第 12 个转换 (12th conversion in regular sequence) 这些位由软件定义转换序列中的第 12 个转换通道的编号 (0 17)。
24:20	SQ11[4:0]	RW	规则序列中的第 11 个转换 (11th conversion in regular sequence)
19:15	SQ10[4:0]	RW	规则序列中的第 10 个转换 (10th conversion in regular sequence)
14:10	SQ9[4:0]	RW	规则序列中的第 9 个转换 (9th conversion in regular sequence)
9:5	SQ8[4:0]	RW	规则序列中的第 8 个转换 (8th conversion in regular sequence)
4:0	SQ7[4:0]	RW	规则序列中的第 7 个转换 (7th conversion in regular sequence)

15.12.11　ADC 规则序列寄存器 3(ADC_SQR3)

地址偏移：0x34

复值：0x0000_0000

位	符号	访问	描述
31:30	-	R	保留。必须为 0。
29:25	SQ6[4:0]	RW	规则序列中的第 6 个转换 (6th conversion in regular sequence) 这些位由软件定义转换序列中的第 6 个转换通道的编号 (0 17)。
24:20	SQ5[4:0]	RW	规则序列中的第 5 个转换 (5th conversion in regular sequence)
19:15	SQ4[4:0]	RW	规则序列中的第 4 个转换 (4th conversion in regular sequence)
14:10	SQ3[4:0]	RW	规则序列中的第 3 个转换 (3th conversion in regular sequence)
9:5	SQ2[4:0]	RW	规则序列中的第 2 个转换 (2th conversion in regular sequence)
4:0	SQ1[4:0]	RW	规则序列中的第 1 个转换 (1th conversion in regular sequence)

15.12.12　ADC 注入序列寄存器 (ADC_JSQR)

地址偏移：0x38

复位值：0x0000_0000

位	符号	访问	描述
31:22	-	R	保留。必须为 0。
21:20	JL[1:0]	RW	注入通道序列长度 (Injected sequence length) 这些位由软件定义在规则通道转换序列中的通道数目。 00：1 个转换 01：2 个转换 10：3 个转换 11：4 个转换

19:15	JSQ4[4:0]	RW	注入序列中的第 4 个转换 (4th conversion in injected sequence) 这些位由软件定义转换序列中的第 4 个转换通道的编号 (0 17)。 注：不同于规则转换序列，如果 JL[1:0] 的长度小于 4，则转换的序列顺序是从 (4-JL) 开始。例如：ADC_JSQR[21:0] = 10 00011 00011 00111 00010，意味着扫描转换将按下列通道顺序转换：7、3、3，而不是 2、7、3。
14:10	JSQ3[4:0]	RW	注入序列中的第 3 个转换 (3rd conversion in injected sequence)
9:5	JSQ2[4:0]	RW	注入序列中的第 2 个转换 (2nd conversion in injected sequence)
4:0	JSQ1[4:0]	RW	注入序列中的第 1 个转换 (1st conversion in injected sequence)

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

地址偏移：0x3C - 0x48

复位值：0x0000_0000

位	符号	访问	描述
31:16	-	R	保留。必须为 0。
15:0	JDATA[15:0]	R	注入通道转换的数据 (Injected data) 这些位为只读，包含了注入通道的转换结果。数据是左对齐或右对齐，如图15.5、图15.6、图15.7 和图15.8所示。

15.12.14　ADC 规则数据寄存器 (ADC_DR)

地址偏移：0x4C

复位值：0x0000_0000

位	符号	访问	描述
31:16	-	R	保留。必须为 0。
15:0	DATA[15:0]	R	规则通道转换的数据 (Regular data) 该寄存器是规则通道 FIFO 的入口。读取该寄存器，可以得到规则通道的转换结果。数据是左对齐或右对齐，如图15.5、图15.6、图15.7 和图15.8所示。

15.12.15　ADC 控制寄存器 3(ADC_CR3)

地址偏移：0x54

复位值：0x0000_0000

位	符号	访问	描述
31:18	-	R	保留。必须为 0。
17	EMPIE	RW	FIFO 空中断使能位。 该位由软件设置和清除。 0：禁止 EMPF 中断； 1：允许 EMPF 中断。

16	OVFIE	RW	FIFO 溢出中断使能位。 该位由软件设置和清除。 0：禁止 OVF 中断； 1：允许 OVF 中断。
15:8	ADC_PRS[7:0]	RW	ADC 时钟预分频器配置： 8'h00 : $Tad = 2 * Tapb$ 8'h01 : $Tad = 4 * Tapb$ 8'h02 : $Tad = 6 * Tapb$ ..... 8'hFF : $Tad = 512 * Tapb$
7	-	R	保留。必须为 0。
6	12BIT	RW	配置 ADC 的精度为 10 位/12 位。 0: ADC 的精度是 10 位 1: ADC 的精度是 12 位
5:4	-	R	保留。必须为 0。
3:2	SAMCHN	RW	当 ADC 的精度是 12 位时，该位没有定义，必须为 0。当 ADC 的精度是 10 位时: 2'b00: 采样保持电路 0 工作 2'b01: 采样保持电路 0 和 A 同时工作 2'b10: 采样保持电路 0，A 和 B 同时工作 2'b11: 采样保持电路 0，A，B 和 C 同时工作
1:0	ADVMODE	RW	高级工作模式配置 2'b0x: 只有采样保持电路 0 工作 2'b10: 多个采样保持电路顺序工作 2'b11: 多个采样保持电路并行工作 注意：当 ADC 是 12 位精度时，不能工作在高级工作模式。

15.12.16　ADC 注入序列 DMA 接口 (ADC_JDMAR)

地址偏移：0x58

复位值：0x0000_0000

当注入通道 DMA 请求发出时，读取该地址可以得到转换的结果。

第十六章　高级控制定时器 (TIM1)

16.1　高级控制定时器 TIM1 简介

高级控制定时器 TIM1 由一个 20 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度 (输入捕获)，或者产生输出波形 (输出比较、PWM、嵌入死区时间的互补 PWM 等)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

高级控制定时器 TIM1 和通用定时器 (TIMx) 是完全独立的，它们不共享任何资源。它们可以同步操作，具体描述参看17.3.14。

16.2　高级控制定时器 TIM1 主要特性

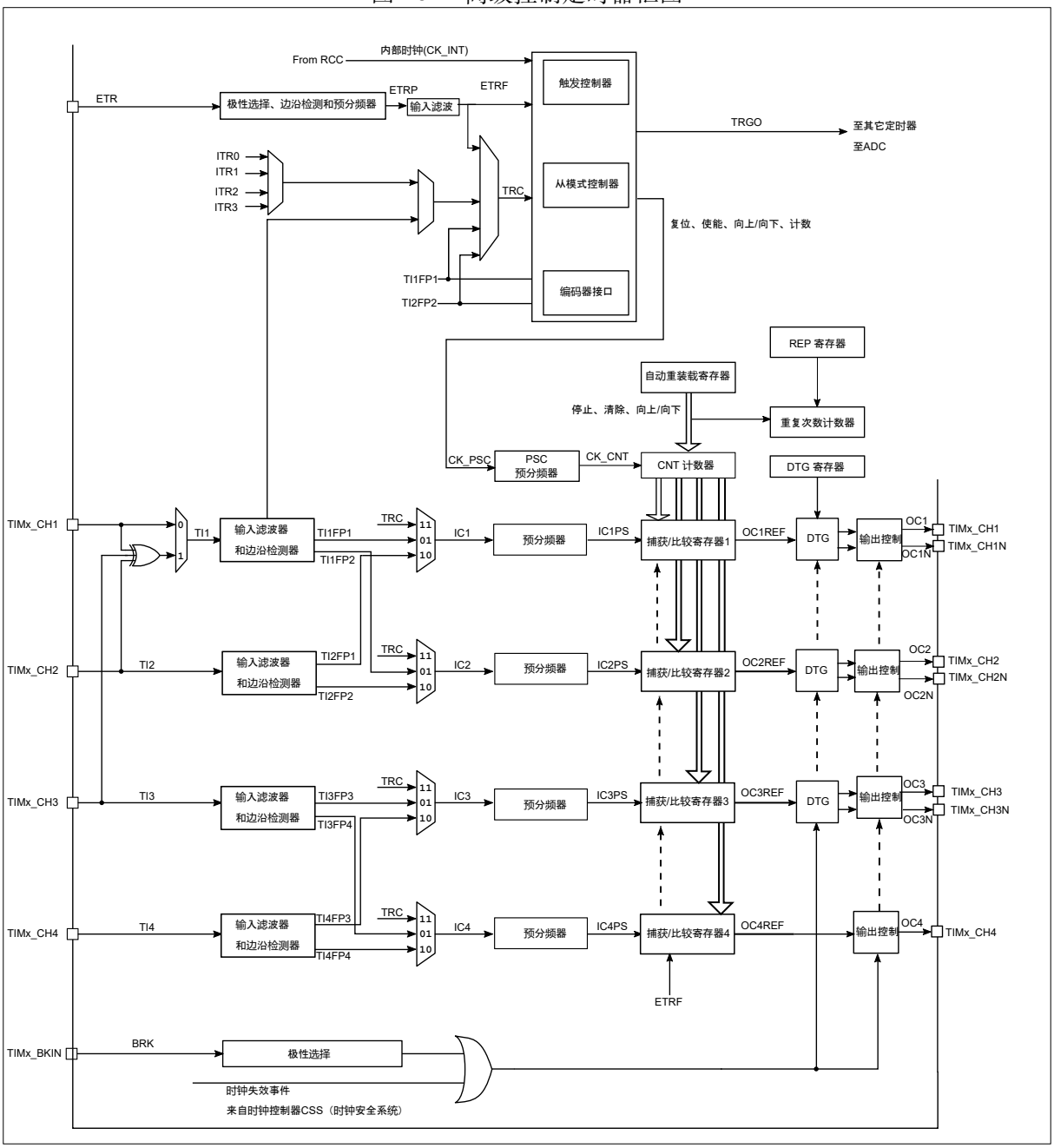
TIM1 定时器的功能包括：

- 20 位向上、向下、向上/下自动装载计数器
- 16 位可编程 (可以实时修改) 预分频器，计数器时钟频率的分频系数为 1 ~ 65535 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成 (边缘或中间对齐模式)
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化 (通过软件或者内部/外部触发)
  - 触发事件 (计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获

- 输出比较
- 刹车信号输入

- 支持针对定位的增量 (正交) 编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

图 16.1: 高级控制定时器框图



16.3　高级控制定时器 TIM1 功能描述

16.3.1　时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

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

时基单元包含：

- 计数器寄存器 (TIMx_CNT)
- 预分频器寄存器 (TIMx_PSC)
- 自动装载寄存器 (TIMx_ARR)
- 重复次数寄存器 (TIMx_RCR)

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

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位 (CEN) 时，CK_CNT 才有效。(更多有关使能计数器的细节，请参见控制器的从模式描述)。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频率按 1 到 65536 之间的任意值分频。它是基于一个 (在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。图16.2和图16.3给出了在预分频器运行时，更改计数器参数的例子。

图 16.2: 当预分频器的参数从 1 到 2 时，计数器的时序图

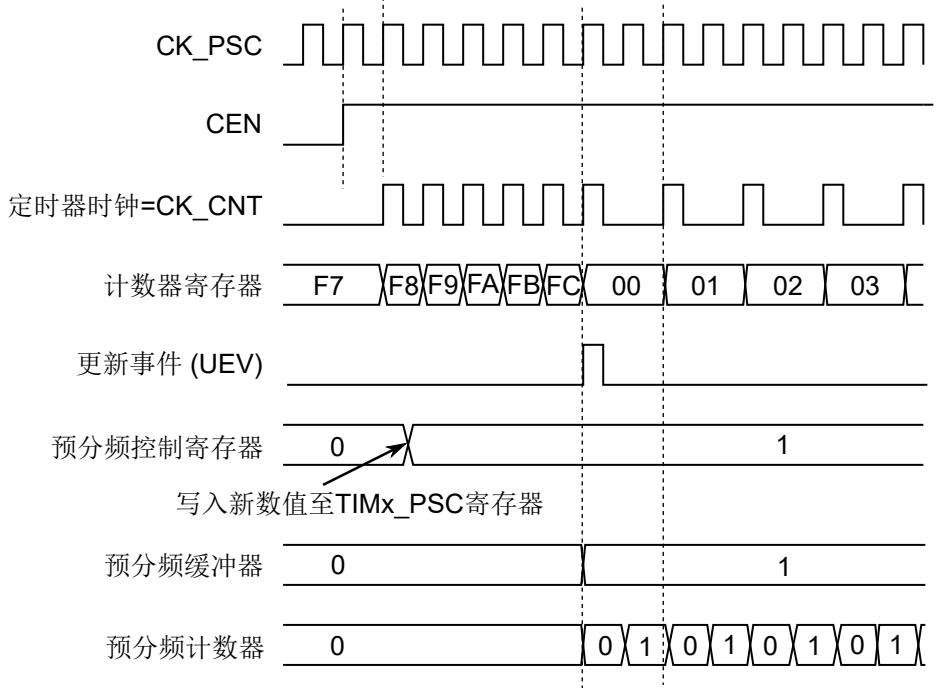
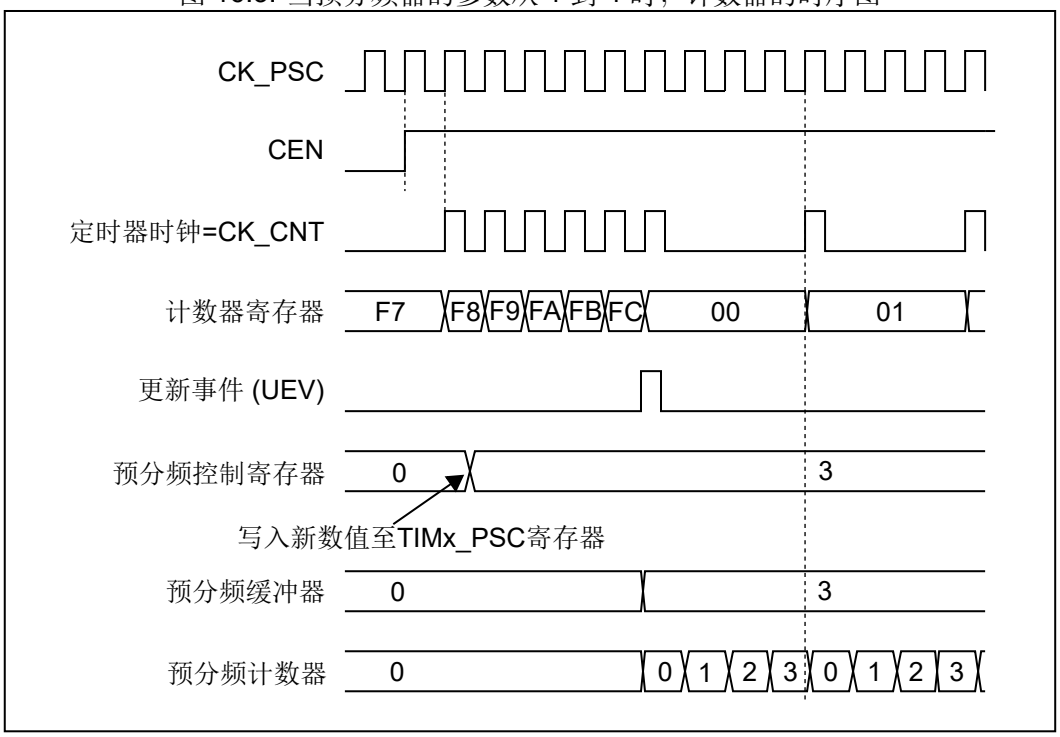


图 16.3: 当预分频器的参数从 1 到 4 时，计数器的时序图



16.3.2　计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值 (TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。如果使用了重复计数器功能，在向上计数达到设置的重复计数次数 (TIMx_RCR)

时，产生更新事件 (UEV)；否则每次计数器溢出时才产生更新事件。

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 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清’0’之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清’0’，同时预分频器的计数也被清’0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志 (即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时 (依据 URS 位) 设置更新标志位 (TIMx_SR 寄存器中的 UIF 位)。

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

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

图 16.4: 计数器时序图，内部时钟分频因子为 1

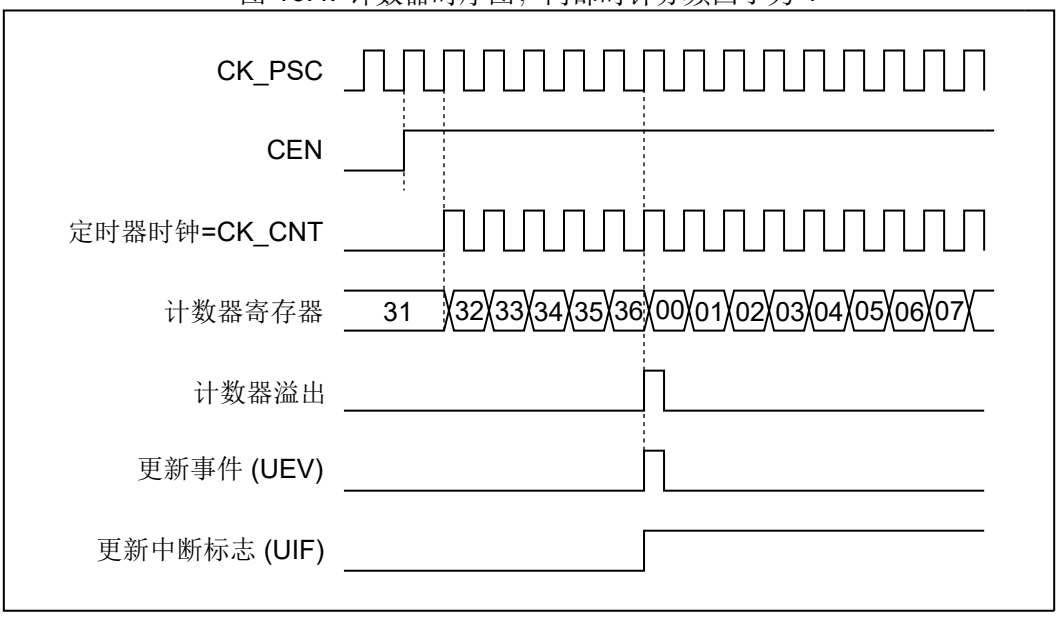


图 16.5: 计数器时序图，内部时钟分频因子为 2

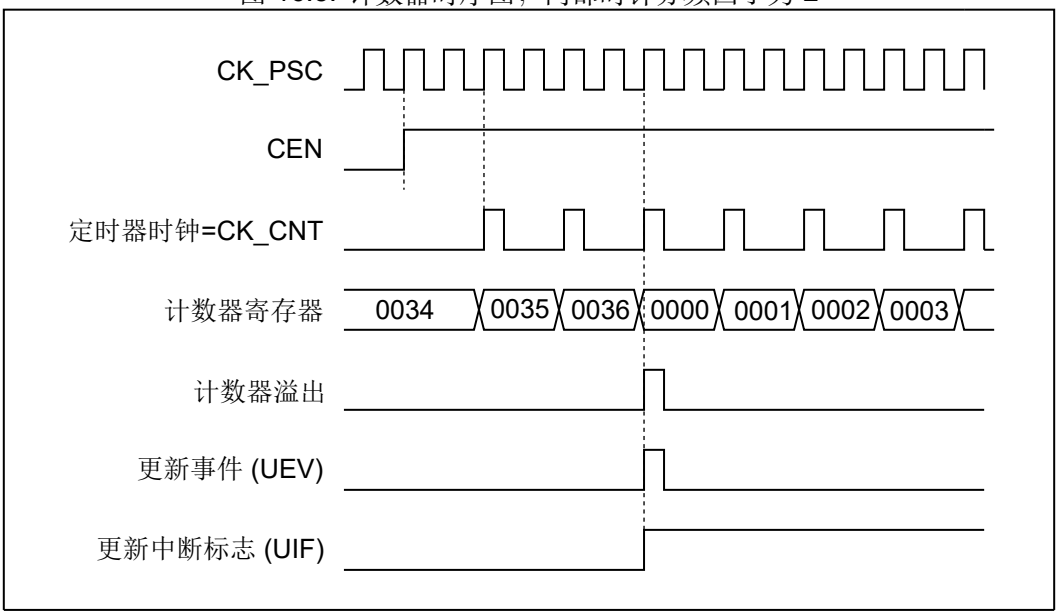


图 16.6: 计数器时序图，内部时钟分频因子为 4

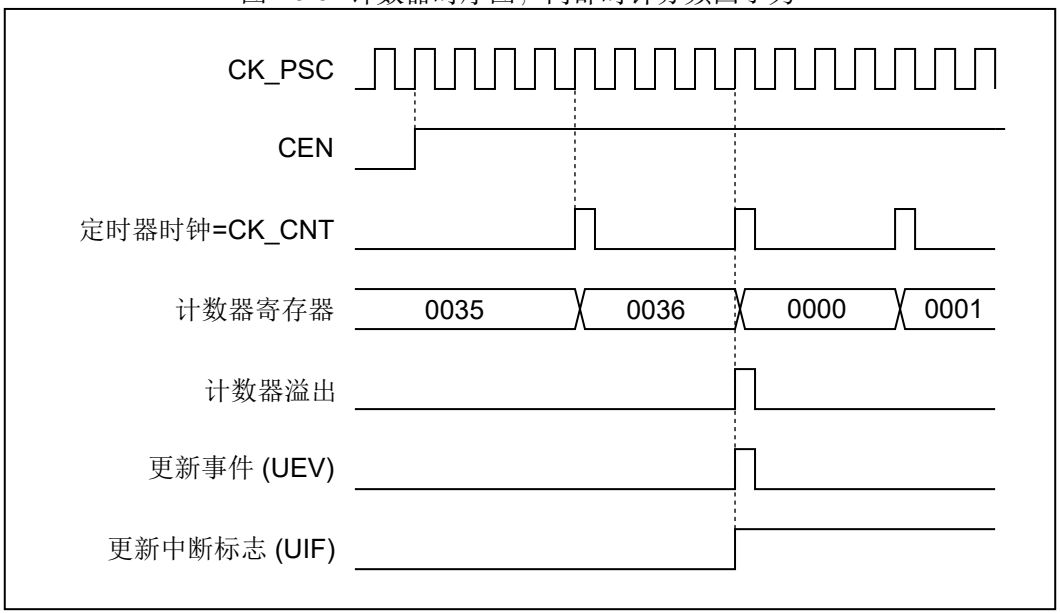


图 16.7: 计数器时序图，内部时钟分频因子为 N

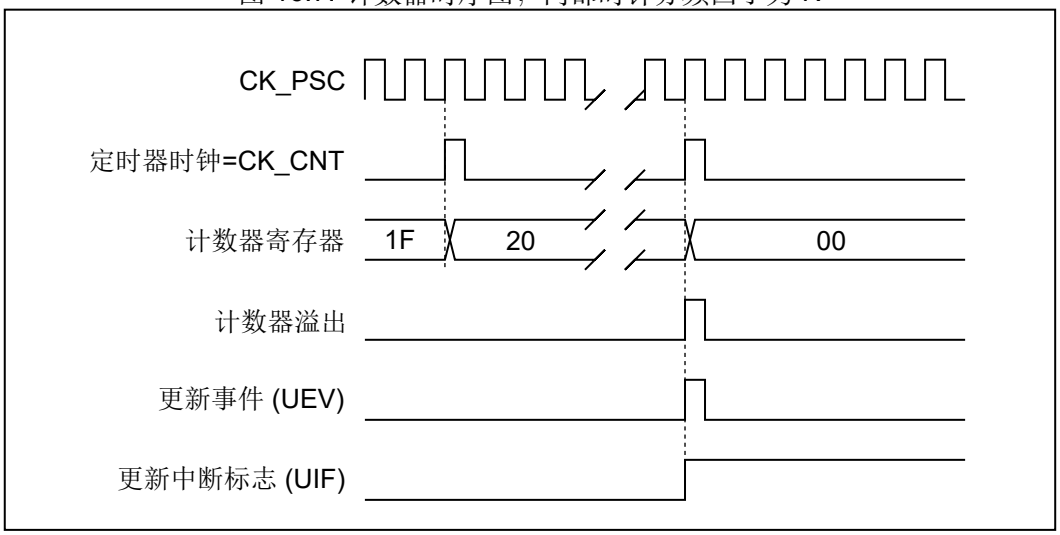


图 16.8: 计数器时序图，当 ARPE=0 时的更新事件 (TIMx_ARR 没有预装入)

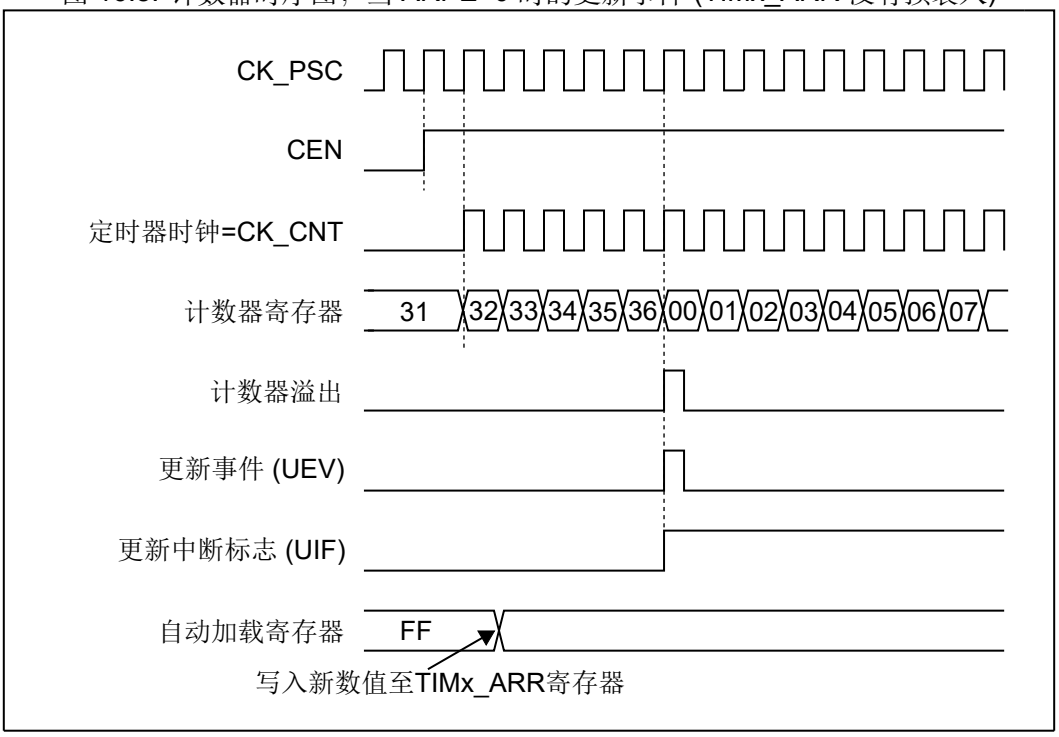
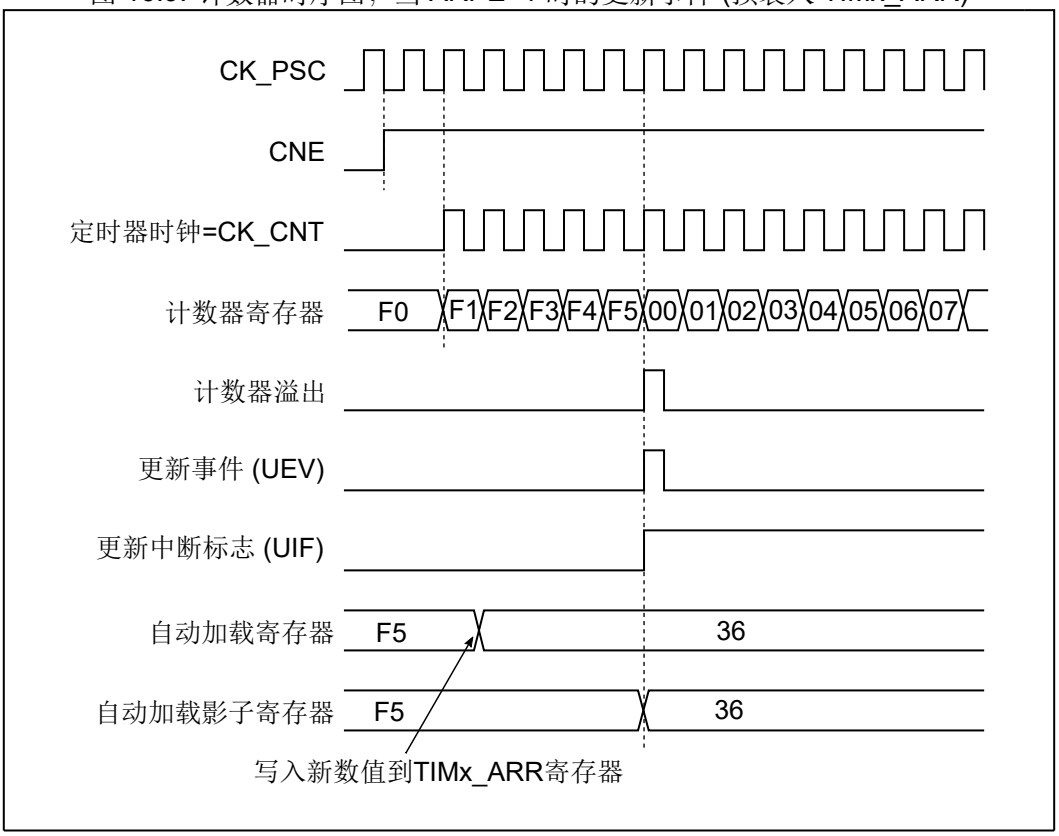


图 16.9: 计数器时序图，当 ARPE=1 时的更新事件 (预装入 TIMx_ARR)



向下计数模式

在向下模式中，计数器从自动装入的值 (TIMx_ARR 计数器的值) 开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。如果使用了重复计数器，当向下计数重复了重复计数寄存器 (TIMx_RCR) 中设定的次数后，将产生更新事件 (UEV)，否则每次计数器下溢时才产生更新事件。

在 TIMx_EGR 寄存器中 (通过软件方式或者使用从模式控制器) 设置 UG 位，也同样可以产生一个更新事件。设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始 (但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志 (因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且 (根据 URS 位的设置) 更新标志位 (TIMx_SR 寄存器中的 UIF 位) 也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载的值 (TIMx_PSC 寄存器的值)。
- 当前的自动加载寄存器被更新为预装载值 (TIMx_ARR 寄存器中的内容)。注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

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

图 16.10: 计数器时序图，内部时钟分频因子为 1

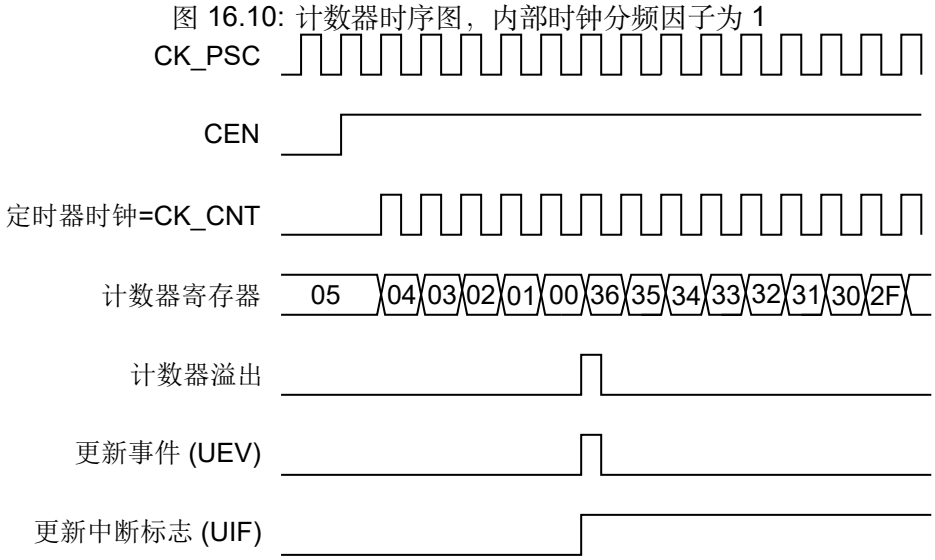


图 16.11: 计数器时序图，内部时钟分频因子为 2

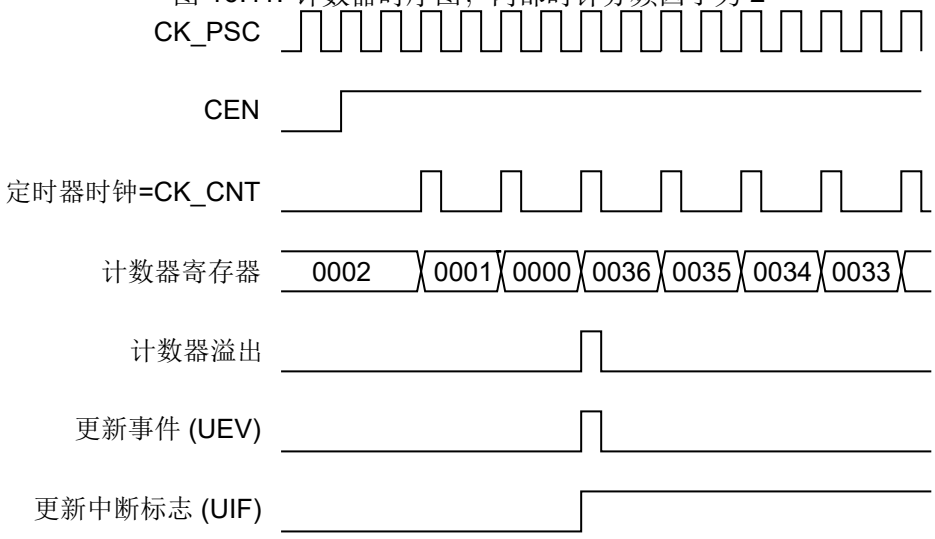


图 16.12: 计数器时序图，内部时钟分频因子为 4

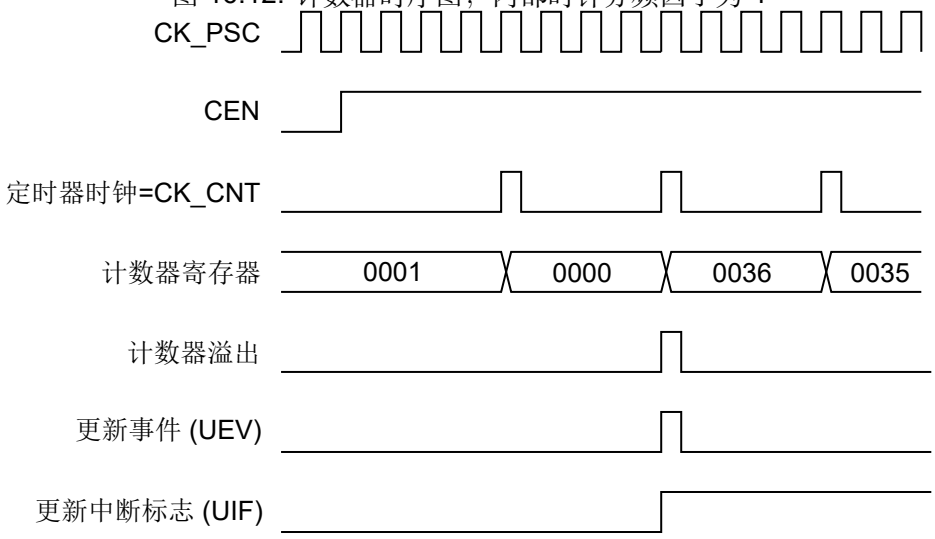


图 16.13: 计数器时序图，内部时钟分频因子为 N

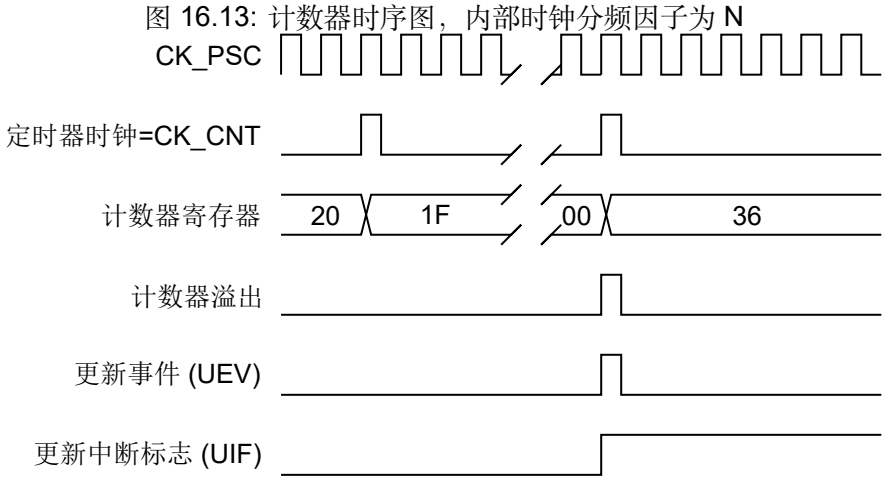
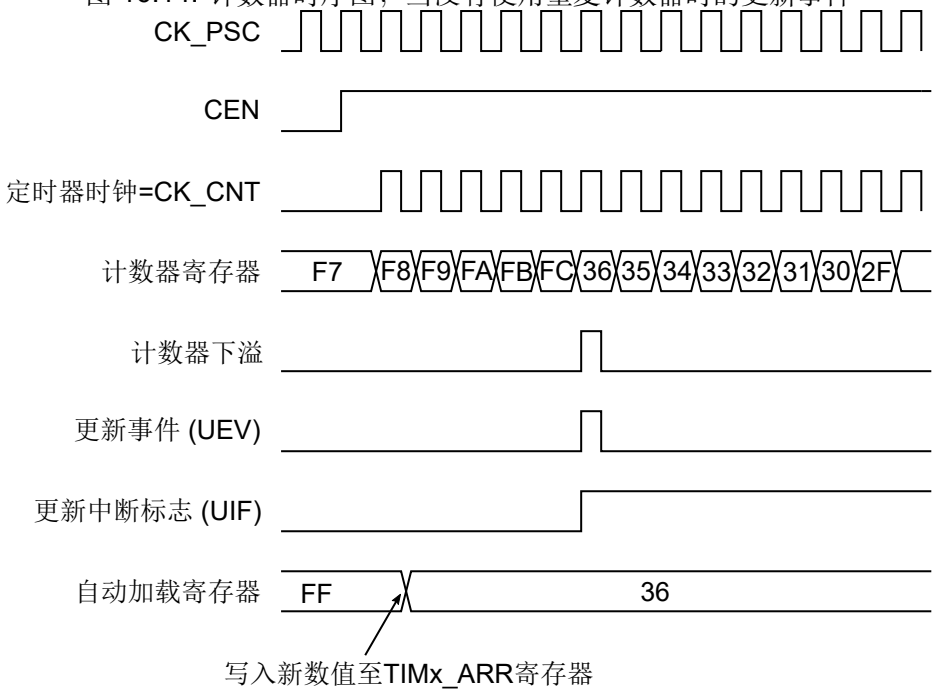


图 16.14: 计数器时序图，当没有使用重复计数器时的更新事件



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。在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过 (软件或者使用从模式控制器) 设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志 (因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且 (根据 URS 位的设置) 更新标志位 (TIMx_SR 寄存器中的 UIF 位) 也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载 (TIMx_PSC 寄存器) 的值。
- 当前的自动加载寄存器被更新为预装载值 (TIMx_ARR 寄存器中的内容)。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)。

以下是一些计数器在中央对齐模式下不同时钟频率下的操作的例子：

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

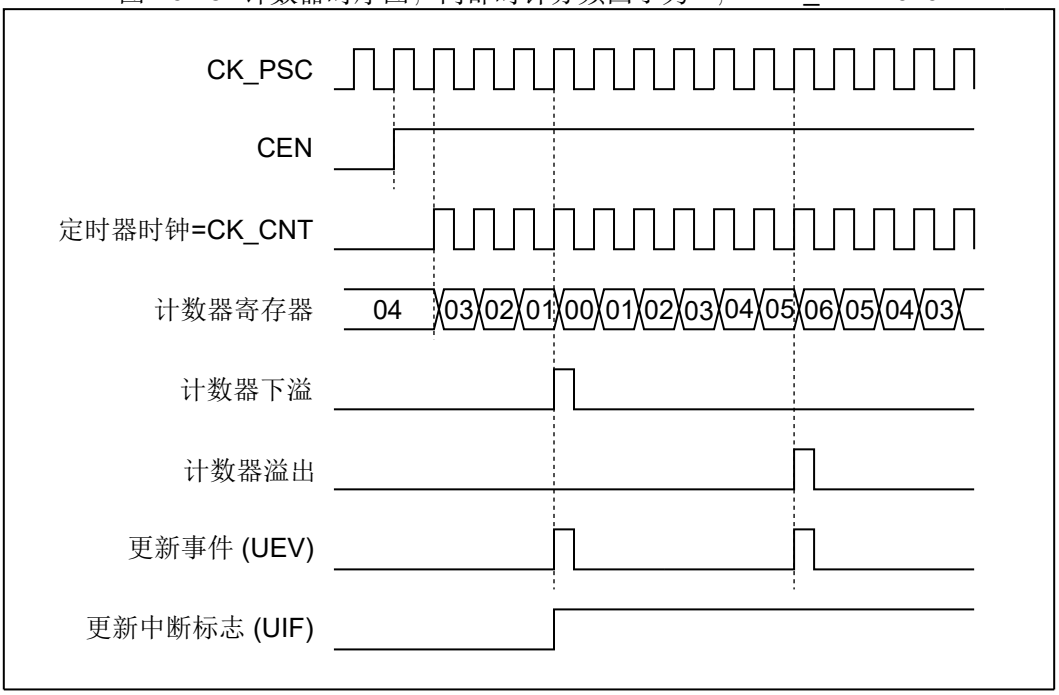


图 16.16: 计数器时序图，内部时钟分频因子为 2

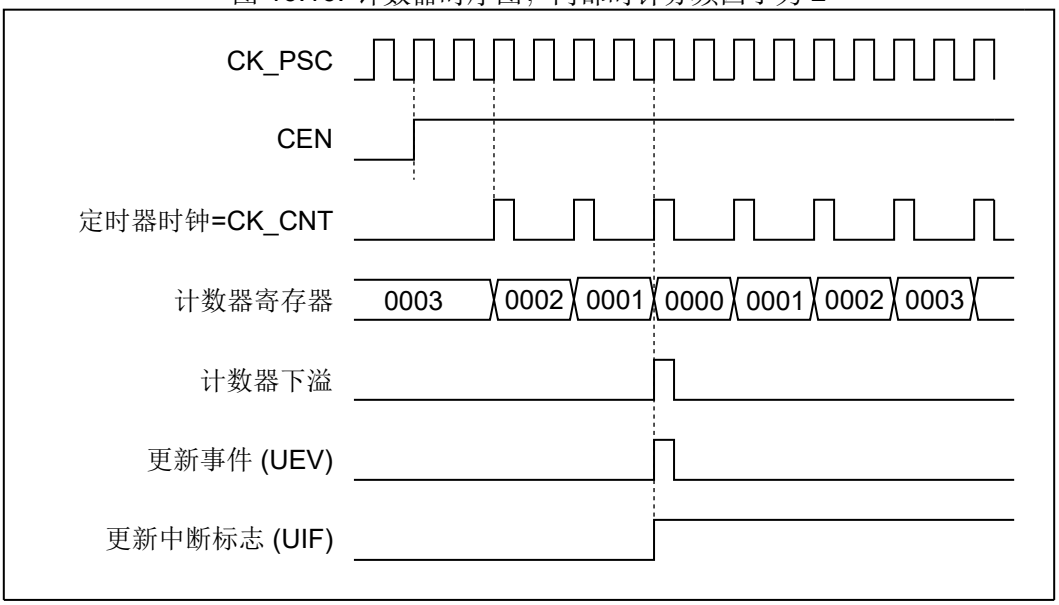
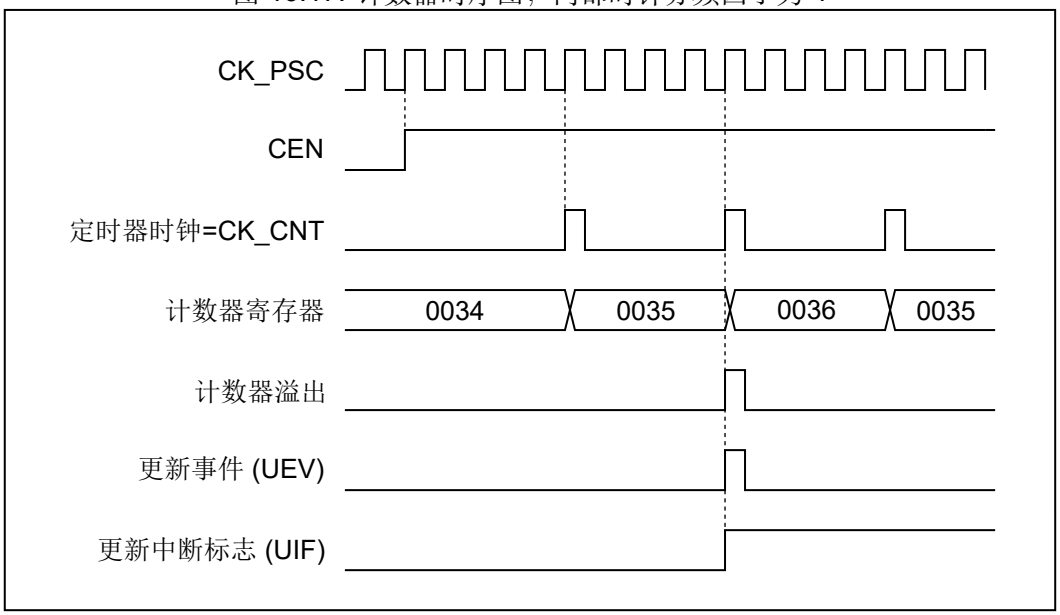


图 16.17: 计数器时序图，内部时钟分频因子为 4



注：中心对齐模式 *2* 或 *3* 是在溢出时与 *UIF* 标志一起使用。

图 16.18: 计数器时序图，内部时钟分频因子为 N

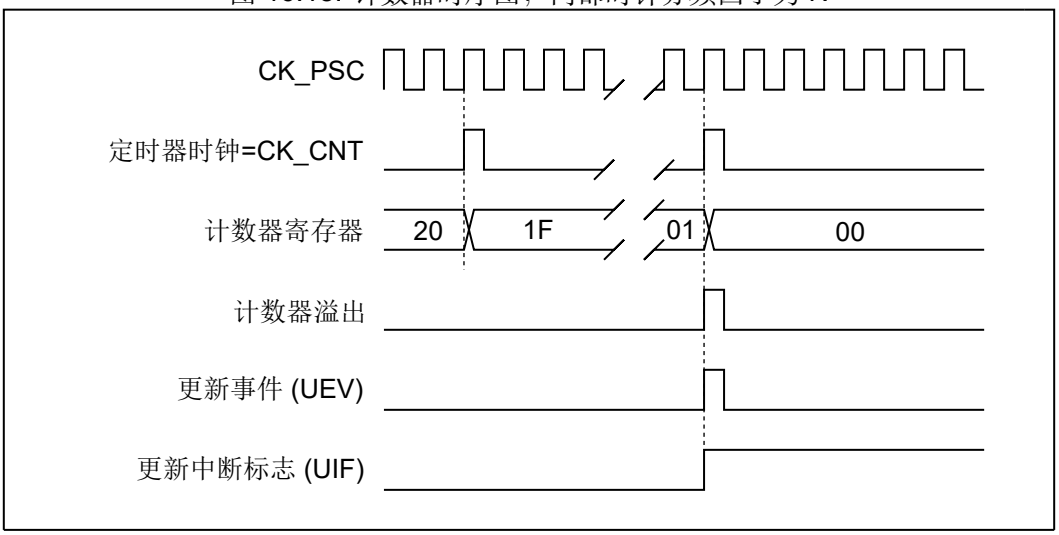


图 16.19: 计数器时序图，ARPE=1 时的更新事件 (计数器下溢)

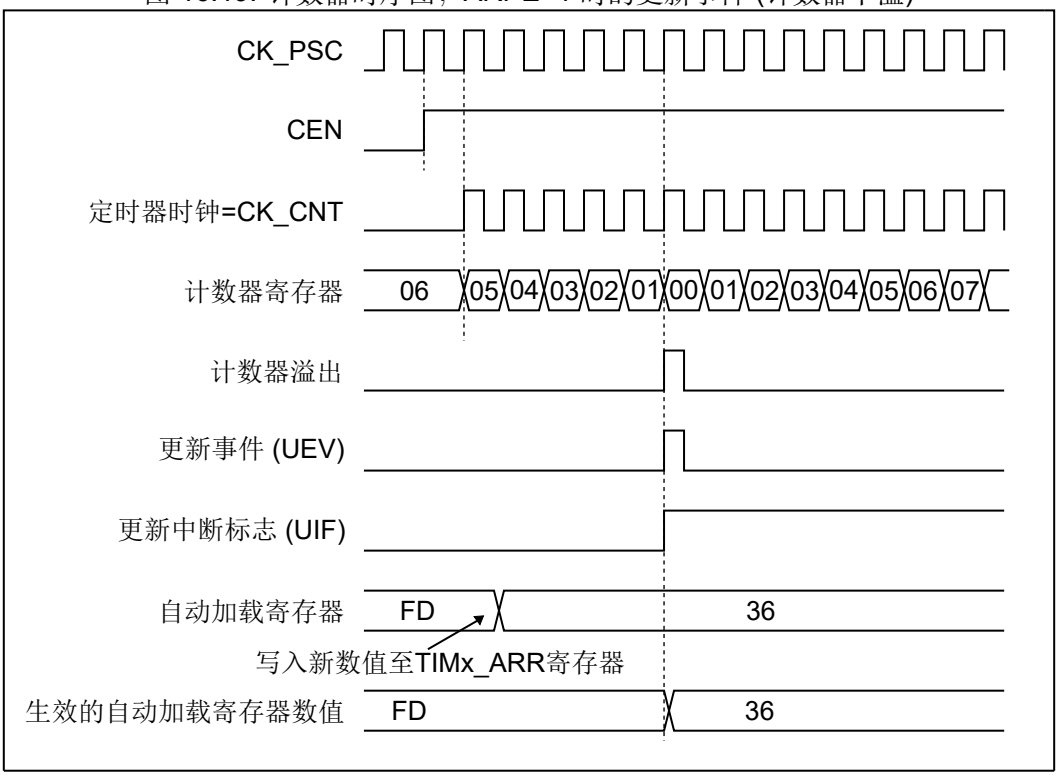
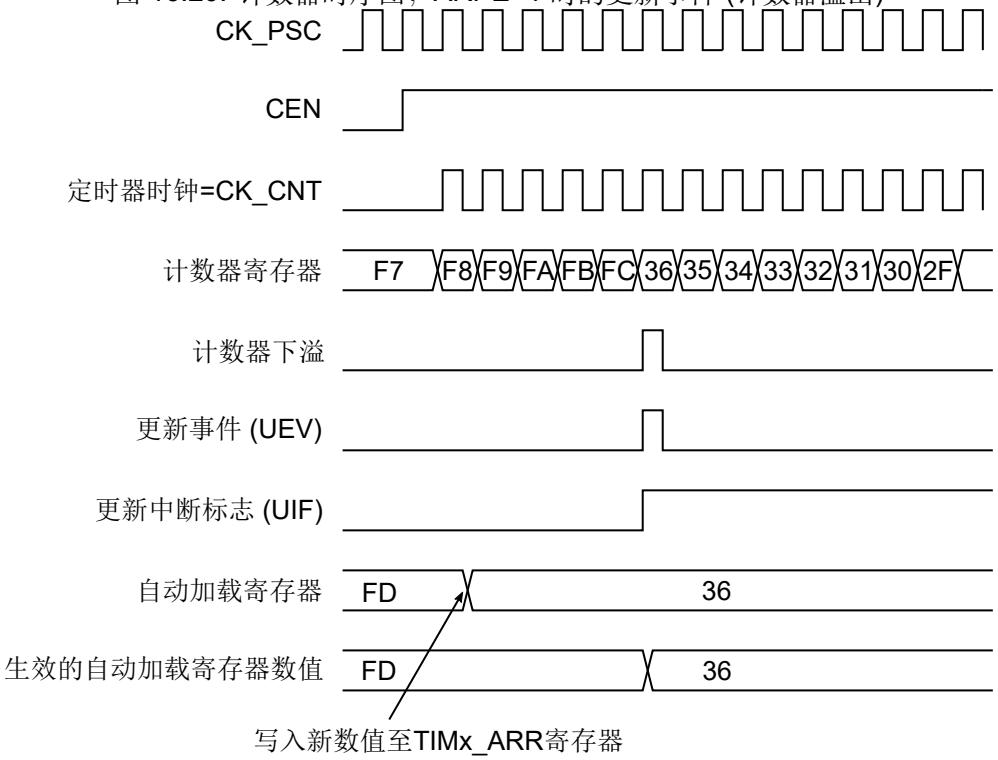


图 16.20: 计数器时序图，ARPE=1 时的更新事件 (计数器溢出)



16.3.3　重复计数器

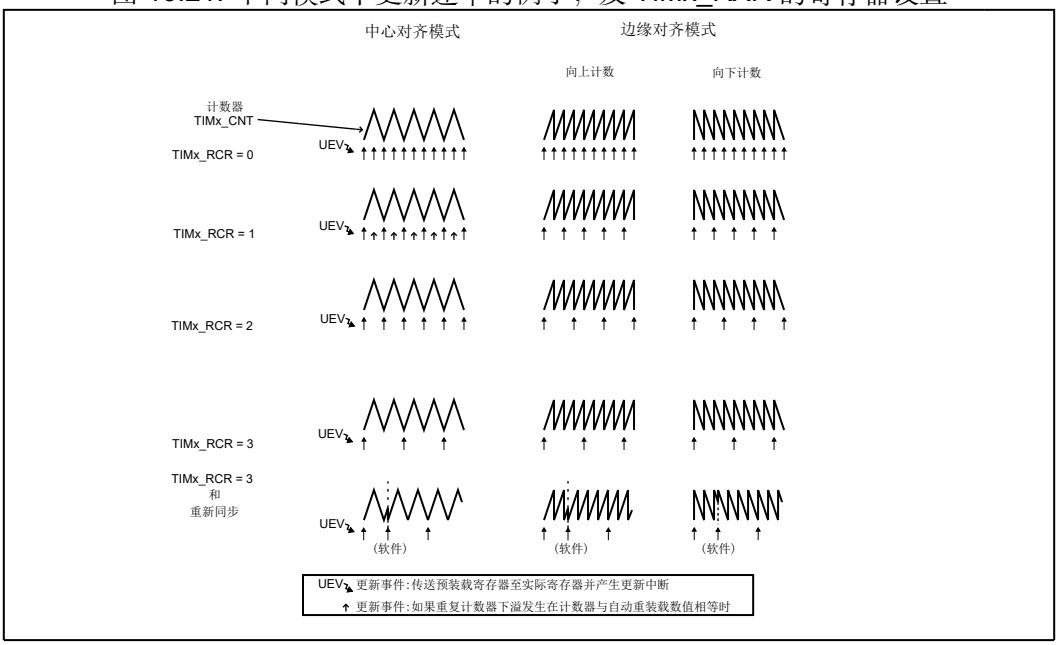
在16.3.1解释了计数器上溢/下溢时更新事件 (UEV) 是如何产生的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器 (TIMx_ARR 自动重载入寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。虽然这样限制了 PWM 的大循环周期为 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 (参看)。当更新事件由软件产生 (通过设置 TIMx_EGR 中的 UG 位) 或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

图 16.21: 不同模式下更新速率的例子，及 TIMx_RXR 的寄存器设置



16.3.4　时钟选择

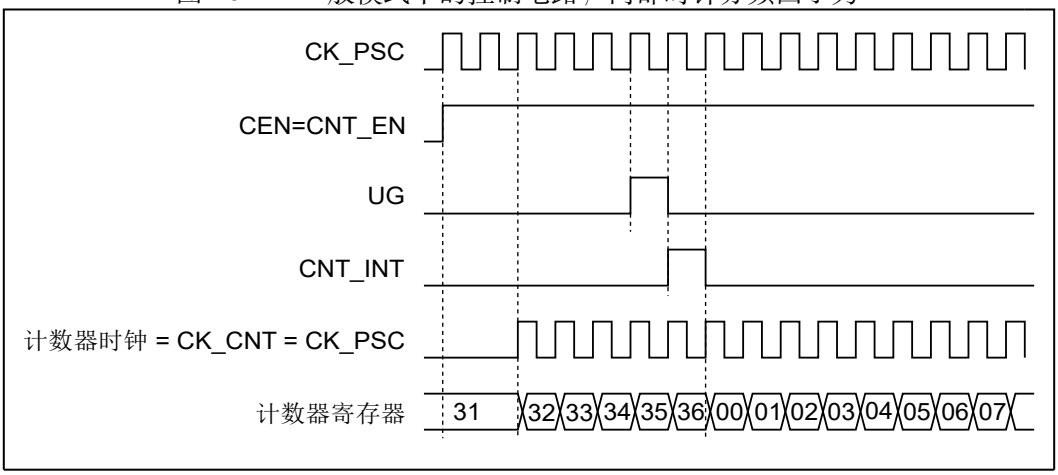
计数器时钟可由下列时钟源提供:

- 内部时钟 (CK_INT)
- 外部时钟模式 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。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。详见下一章。

内部时钟源 (CK_INT)

如果禁止了从模式控制器 (SMS=000)，则 CEN、DIR(TIMx_CR1 寄存器) 和 UG 位 (TIMx_EGR 寄存器) 是事实上的控制位，并且只能被软件修改 (UG 位仍被自动清除)。只要 CEN 位被写成’1’，预分频器的时钟就由内部时钟 CK_INT 提供。下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

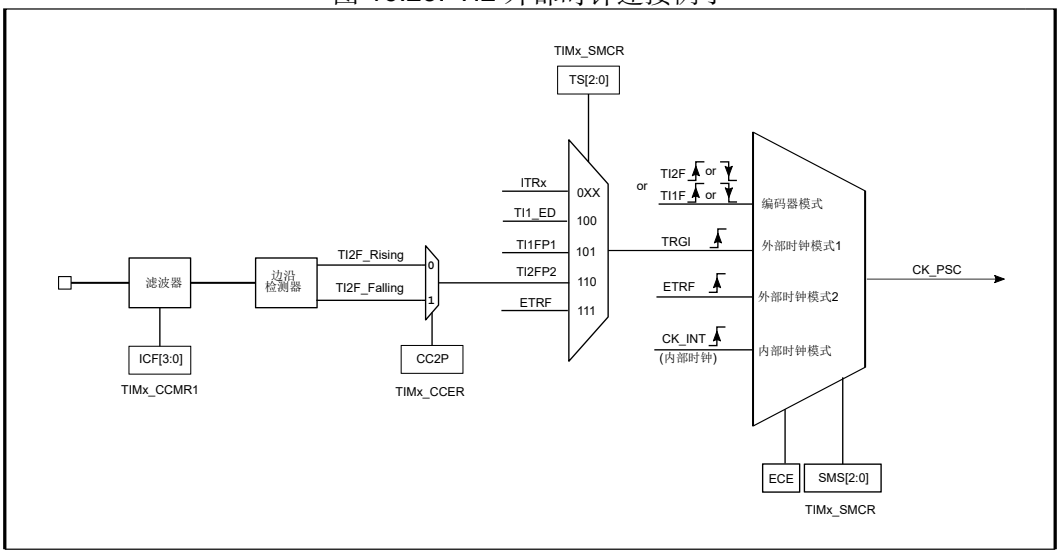
图 16.22: 一般模式下的控制电路，内部时钟分频因子为 1



外部时钟源模式 1

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

图 16.23: TI2 外部时钟连接例子



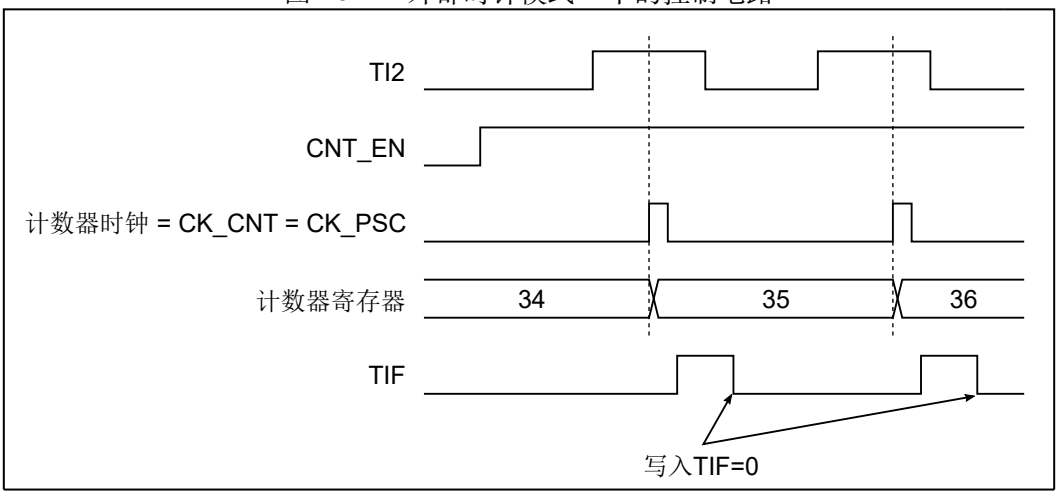
例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

1. 配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿
2. 配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽 (如果不需要滤波器，保持 IC2F=0000)
3. 配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性
4. 配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1
5. 配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源
6. 设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注：捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。

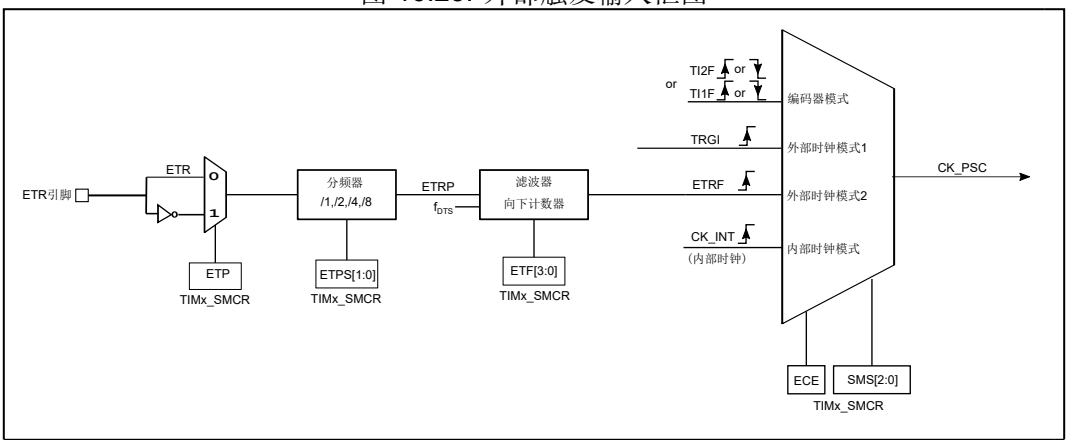
图 16.24: 外部时钟模式 1 下的控制电路



外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1。计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。下图是外部触发输入的框图

图 16.25: 外部触发输入框图



实例：配置计数器为在 ETR 下每 2 个上升沿计数一次的向上计数器

1. 在本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000
2. 设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0] = 01
3. 选择 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0
4. 开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1
5. 启动计数器，写 TIMx_CR1 寄存器中的 CEN=1

16.3.5　捕获/比较通道

TIM1 包含四个捕获/比较通道，如图16.1所示。通道 1，通道 2 和通道 3 是互补输出，通道 4 有所不同，只有正向输出。输入部分对相应的 TIx 输入信号采样，通过滤波器后产生信号 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。该信号通过预分频器之后进入捕获寄存器。

在输出时，通道 1/2/3 通过输出模式控制器，死区发生器和使能控制部分，然后互补输出。通道 4 经过输出模式控制器和使能控制部分后，只有正向输出，没有反向输出。

16.3.6　输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 (TIMx_CCRx) 中。当发生捕获事件时，相应的 CCxIF 标志 (TIMx_SR 寄存器) 被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器) 被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽 (即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以 (以 fDTS 频率) 连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止 (写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置 (中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注：设置 *TIMx_EGR* 寄存器中相应的 *CCxG* 位，可以通过软件产生输入捕获中断和/或 *DMA* 请求。

16.3.7　PWM 输入模式

PWM 输入模式是一种特殊的输入模式，它有以下两个区别：

- 两个 ICx 信号被映射至同一个 TIx 输入。

- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用通道 1 和通道 2。

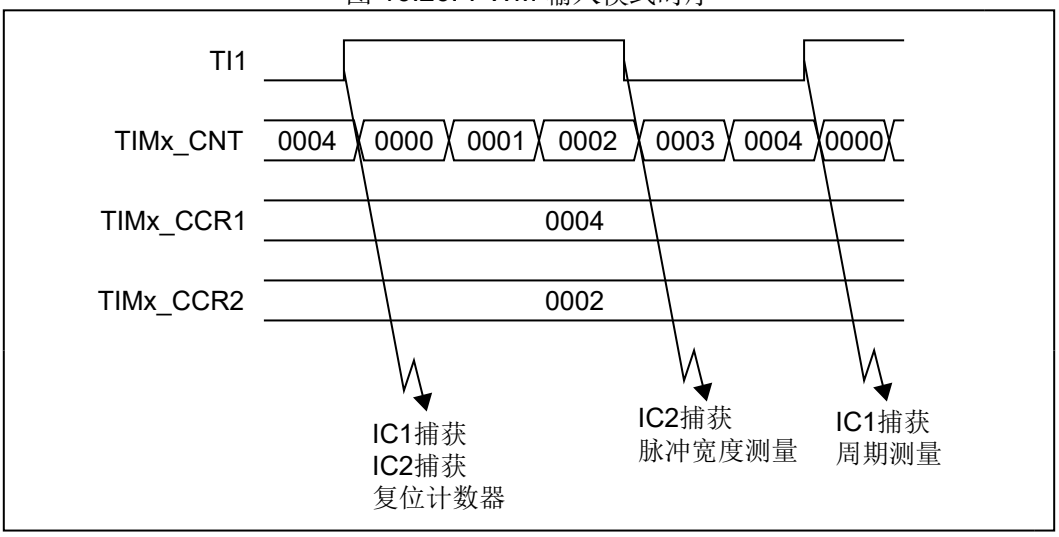
下面给出一个例子具体说明如何配置在该模式下工作。

例如，需要测量输入到 TI1 上的 PWM 信号的长度和占空比，长度存在 TIMx_CCR1 寄存器中，占空比存在 TIMx_CCR2 寄存器中。具体步骤如下：

1. 选择通道 1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
2. 选择 TI1FP1 的有效极性 (用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
3. 选择通道 2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
4. 选择 TI1FP2 的有效极性 (捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
5. 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
6. 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7. 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

下图显示了在这一过程中的时序图。

图 16.26: PWM 输入模式时序



16.3.8　强置输出模式

在输出模式下，输出比较信号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，同时 OCx 得到 CCxP 极性相反的信号。

置 TIMx_CCMRx 寄存器中相应的 OCxM=100，即可强置输出比较信号 (OCxREF/OCx) 为无效状态。这样 OCxREF 被强置为低电平，同时 OCx 得到 CCxP 极性相反的信号。

在该模式下虽然不使用输出比较器的结果，但是输出比较器仍然工作，相应的标志也会被更新，因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

注：*OCxREF 始终为高电平有效*

16.3.9　输出比较模式

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

- 将输出比较模式 (TIMx_CCMRx 寄存器中的 OCxM 位) 和输出极性 (TIMx_CCER 寄存器中的 CCxP 位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 (OCxM=000)、被设置成有效电平 (OCxM=001)、被设置成无效电平 (OCxM=010) 或进行翻转 (OCxM=011)。
- 设置中断状态寄存器中的标志位 (TIMx_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断屏蔽 (TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的使能位 (TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

使用该模式时，需要注意：

- 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；当禁用预装载寄存器时，用户可以在任意时刻写 TIMx_CCRx，并且立刻起作用。当使用预装载寄存器时，用户先写该寄存器，然后在更新事件发生时，该寄存器内容才被装载到 TIMx_CCRx，下一次比较发生时才起作用。
- 在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。
- 输出比较模式 (在单脉冲模式下) 也能用来输出一个单脉冲。
- 精度可以达到计数器的一个计数周期。

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：

- 计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
- 置 OCxPE = 0 禁用预装载寄存器
- 置 CCxP = 0 选择极性为高电平有效
- 置 CCxE = 1 使能输出

5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器

16.3.10　PWM 模式

PWM 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。
使用 PWM 模式时，需要注意：

- 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

- 设置 TIMx_CR1 寄存器的 ARPE 位，使能自动重装载的预装载寄存器。
- 仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。
- OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。
- OCx 的输出使能通过 (TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

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。

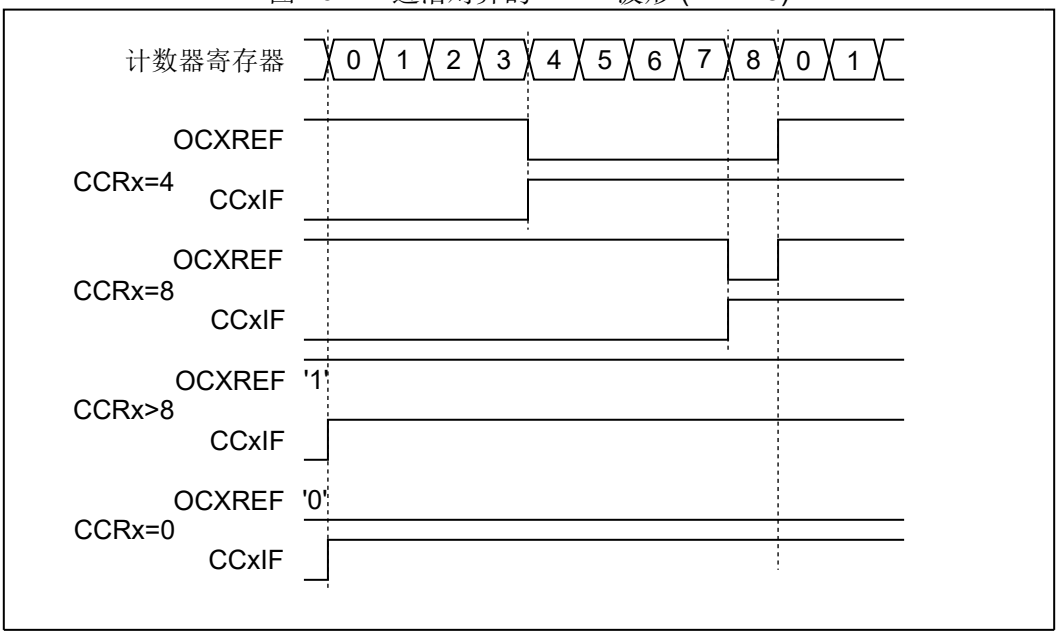
PWM 边沿对齐模式（向上计数）

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

- 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 16.27: 边沿对齐的 PWM 波形 (ARR=8)



PWM 边沿对齐模式（向下计数）

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

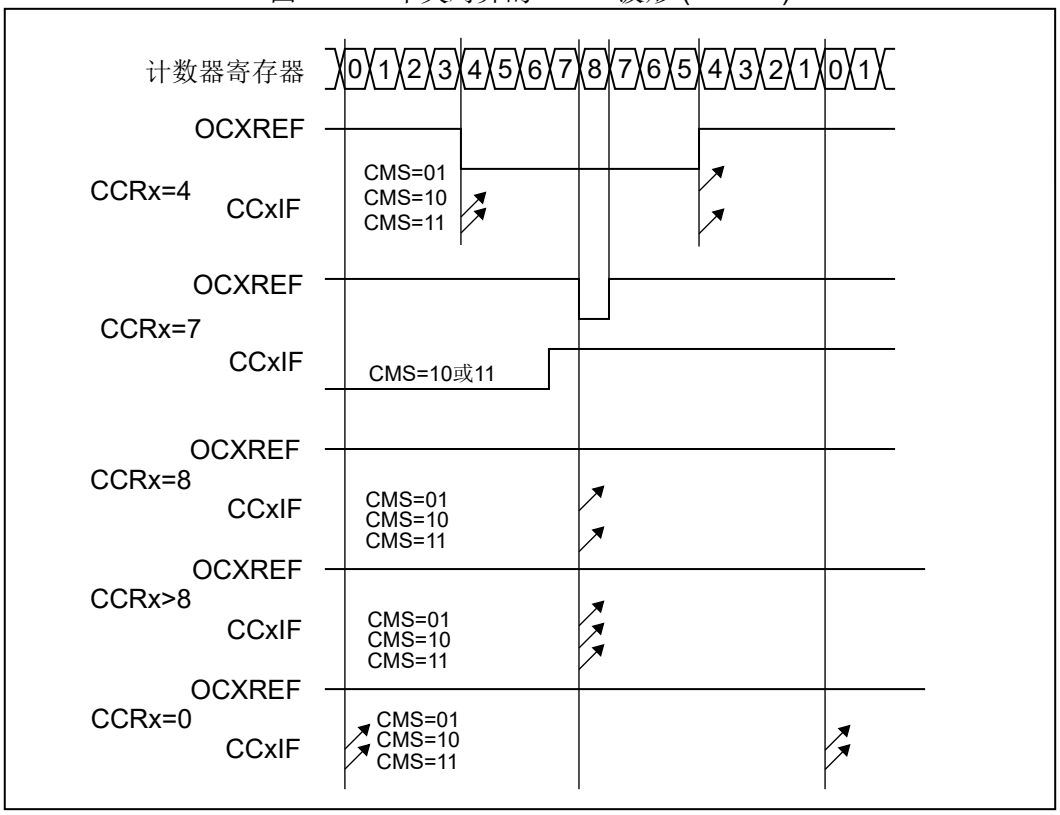
- 在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。
- 如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。
- 该模式下不能产生 0%的 PWM 波形。

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。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。参看16.3.2的中央对齐模式。

下图给出了一些中央对齐的 PWM 波形的例子: (ARR=8, PWM1, TIMx_CR1 寄存器的 CMS=01)

图 16.28: 中央对齐的 PWM 波形 (ARR=8)



使用中央对齐模式的注意:

- 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。
- 使用中央对齐模式保险的方法，就是在启动计数器之前产生一个软件更新，并且不要在计数进行过程中修改计数器的值。

16.3.11　互补输出和死区插入

TIM1 能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性 (电平转换的延时、电源开关的延时等) 来调整死区时间。

配置 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性 (主输出 OCx 或互补输出 OCxN)。互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见表16.43。特别的是，在转换到 IDLE 状态时 (MOE 下降到 0) 死区被激活。同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 10 位的死区发生器。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。

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。

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

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

图 16.29: 带死区插入的互补输出

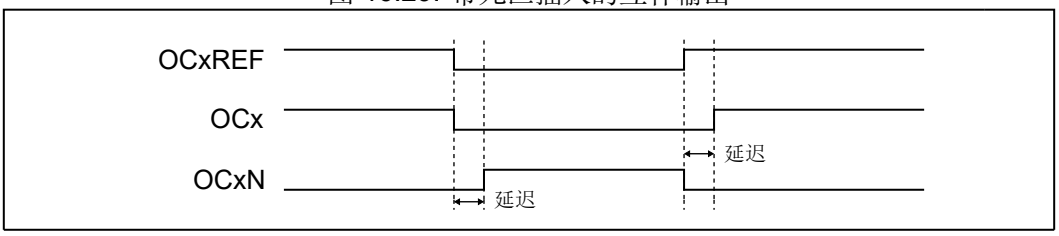


图 16.30: 死区波形延迟大于负脉冲

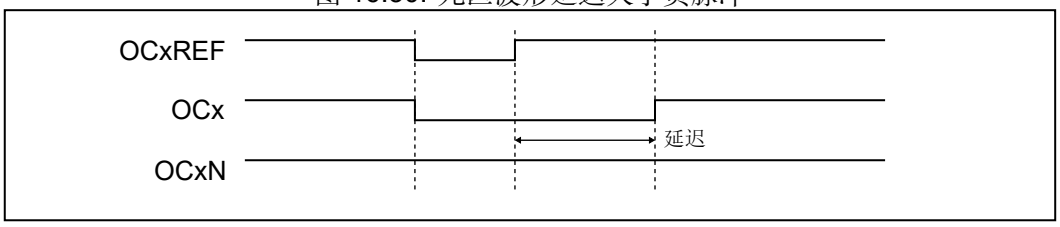
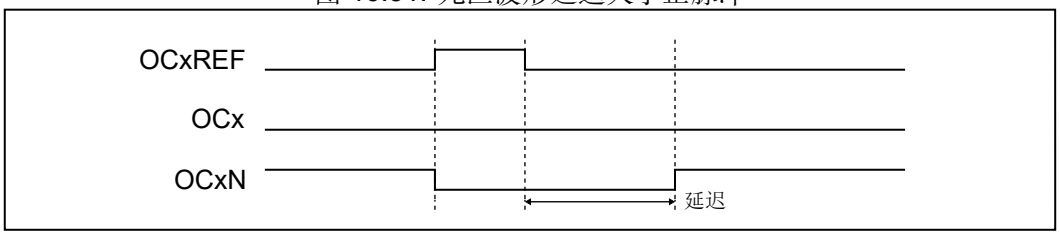


图 16.31: 死区波形延迟大于正脉冲



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。

注意：

1. 当只使能 OCxN 时，它不会反相，当 OCxREF 有效时立即变高。

2. 当 OCx 和 OCxN 都被使能时，当 OCxREF 为高时 OCx 有效；当 OCxREF 低时 OCxN 有效。

16.3.12　使用刹车功能

当使用刹车功能时，依据相应的控制位 (TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见表16.43

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生，详见8.2.10时钟安全系统 (CSS)。

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 下降沿是异步的，在实际信号 (作用在输出端) 和同步控制位 (在 TIMx_BDTR 寄存器中) 之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时 (空指令) 才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时 (在刹车输入端出现选定的电平)，有下述动作：

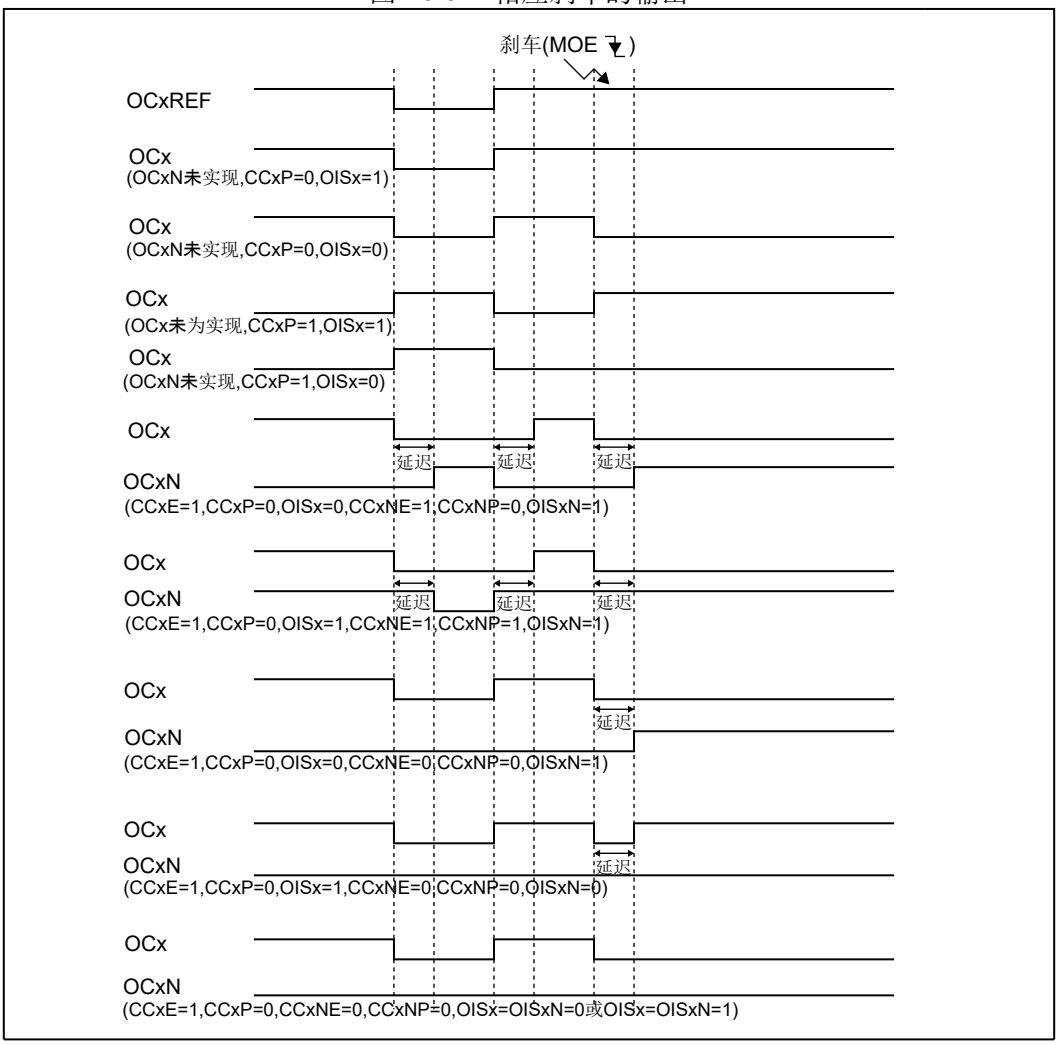
- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态 (由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态 (取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。因为重新同步 MOE，死区时间比通常情况下长一些。
  - 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，当刹车状态标志位是 1 时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，用户可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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

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

下图显示响应刹车的输出实例。

图 16.32: 相应刹车的输出



16.3.13　在外部事件时清除 OCxREF 信号

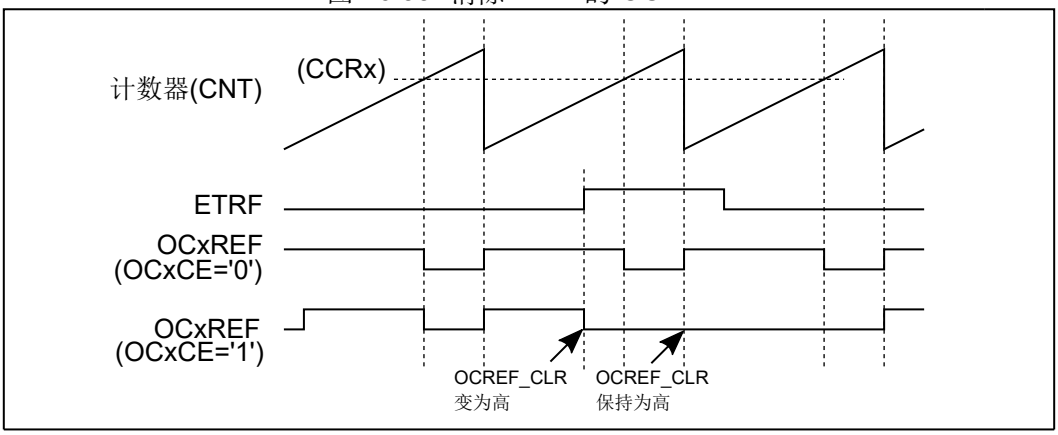
对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为’1’，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

例如，OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

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

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

图 16.33: 清除 TIMx 的 OCxREF



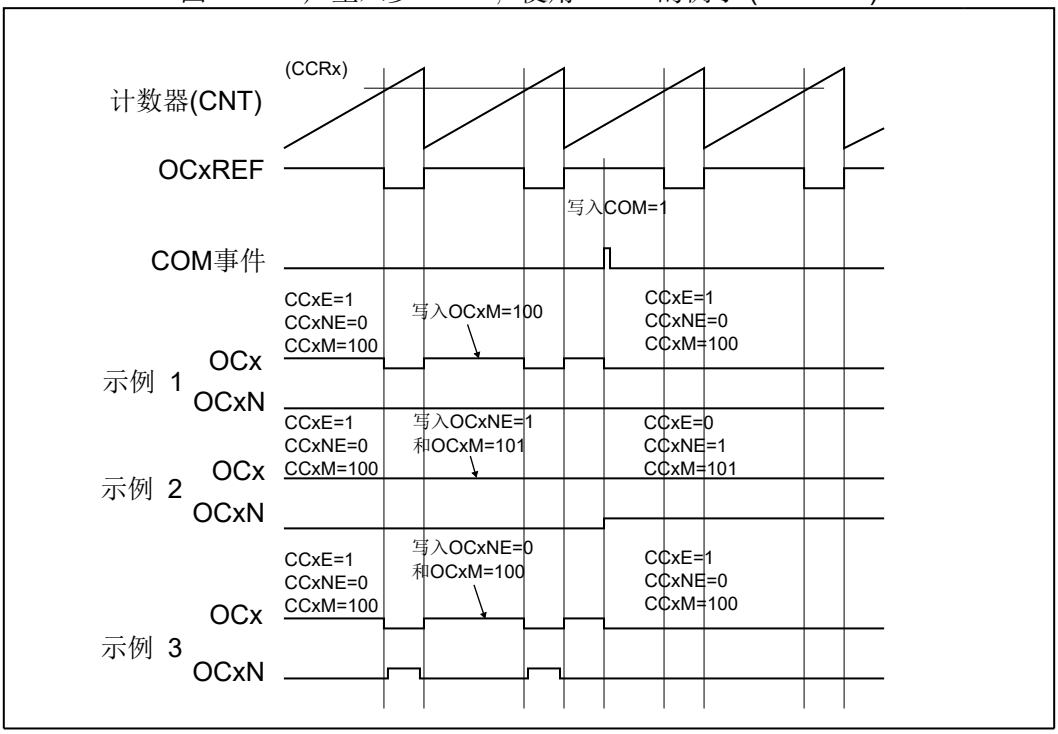
16.3.14　产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。COM 可以通过设置 TIMx_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位 (TIMx_SR 寄存器中的 COMIF 位),这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

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

图 16.34: 产生六步 PWM，使用 COM 的例子 (OSSR=1)



16.3.15　单脉冲模式

单脉冲模式 (OPM) 是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。

设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前 (当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)
- 向下计数方式：计数器 CNT > CCRx

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小延时 tDELAY。

如果要以小延时输出波形，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。

16.3.16　编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI2 的边沿计数，则置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表16.35，假定计数器已经启动，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。

图 16.35: 计数方向与编码器信号的关系

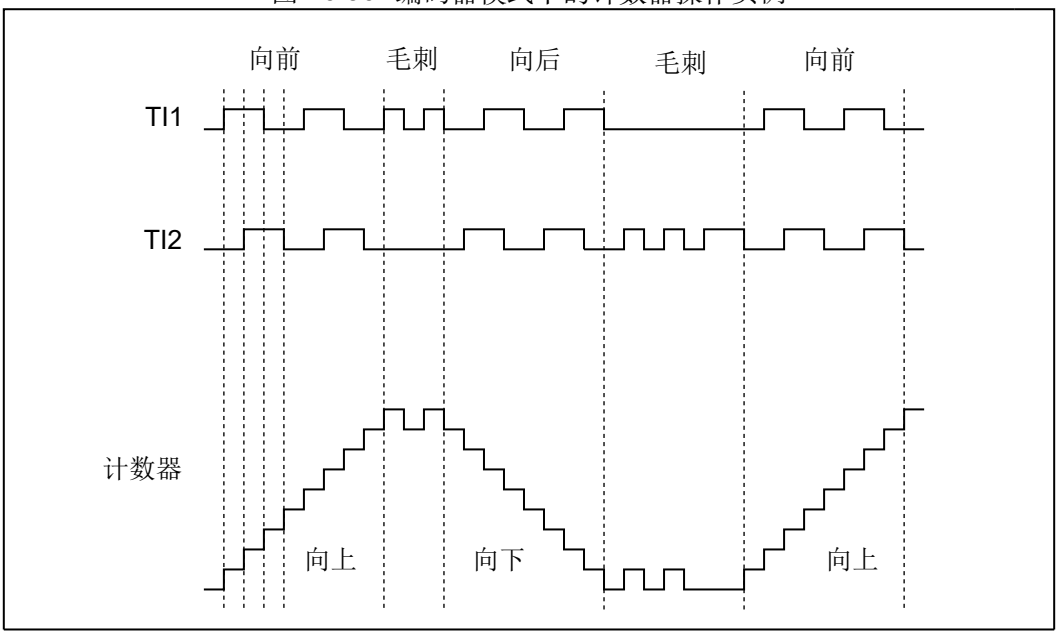
有效边沿	相对信号的电平(TI1FP1对应TI2，TI2FP2对应TI1)	TI1FP1信号		TI2FP2信号	
		上升		下降	
仅在TI1计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在TI2计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在TI1和TI2上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差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：

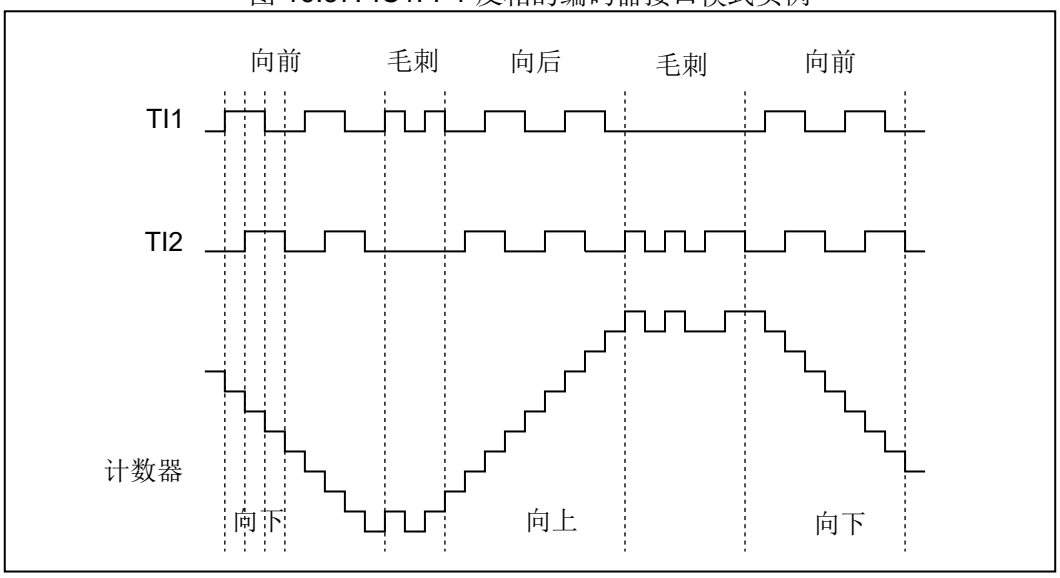
- CC1S=’01’(TIMx_CCMR1 寄存器，IC1FP1 映射到 TI1)
- CC2S=’01’(TIMx_CCMR2 寄存器，IC2FP2 映射到 TI2)
- CC1P=’0’(TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
- CC2P=’0’(TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
- SMS=’011’(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效)
- CEN=’1’(TIMx_CR1 寄存器，计数器使能)

图 16.36: 编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例 (CC1P=’1’，其他配置与上例相同)

图 16.37: IC1FP1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，提供传感器当前位置的信息。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做此目的。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，你可以把计数器的值锁存到第三个输入捕获寄存器 (捕获信号必须是周期的并且可以由另一个定时器产生)；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

16.3.17　定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 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。

16.3.18　与霍尔传感器的接口

使用高级控制定时器 (TIM1) 产生 PWM 信号驱动马达时，可以用另一个通用 TIMx(TIM2、TIM3、TIM4 或 TIM5) 定时器作为“接口定时器”来连接霍尔传感器，见图16.38，3 个定时器输入脚 (CC1、CC2、CC3) 通过一个异或门连接到 TI1 输入通道 (通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器从新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

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

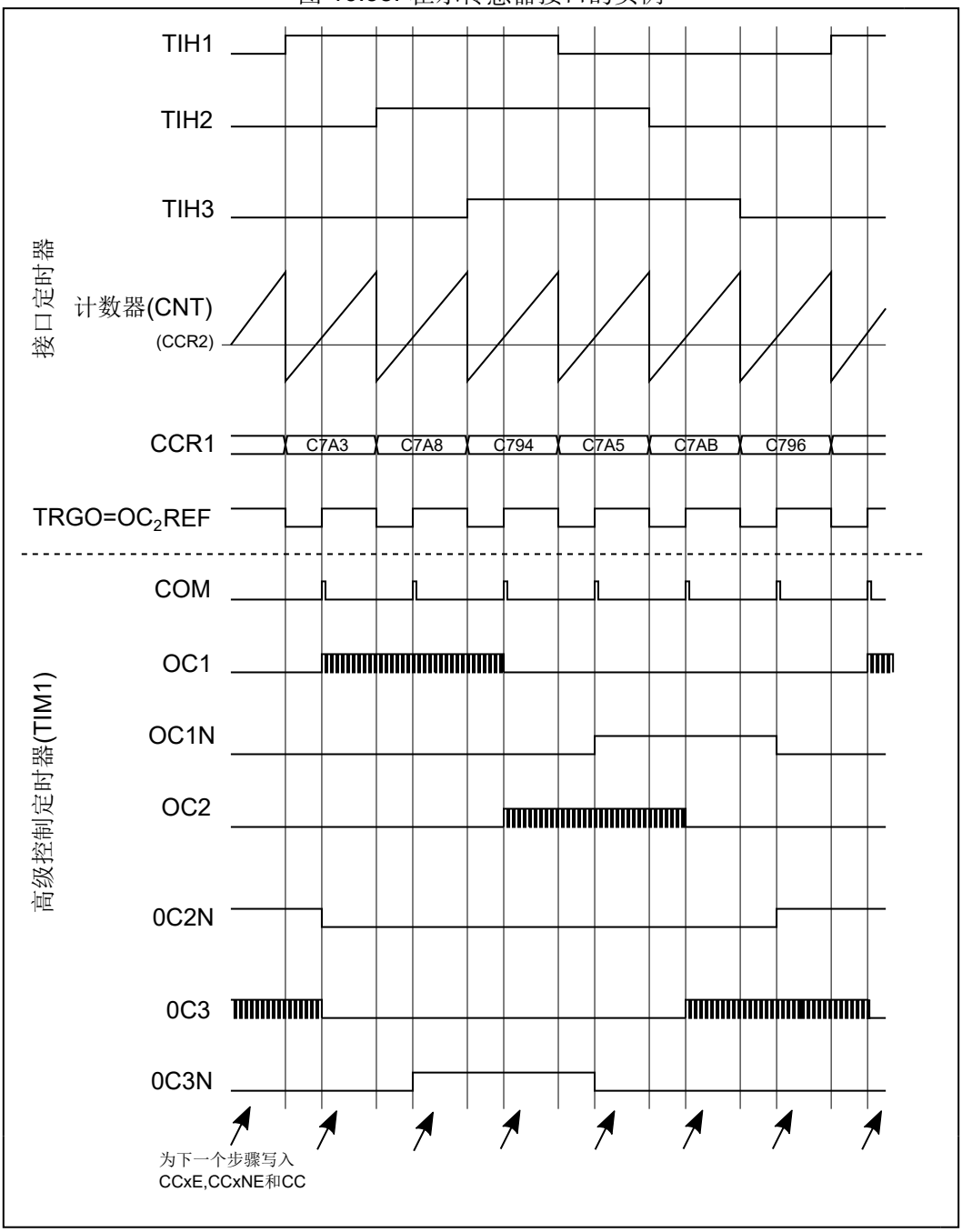
举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。

- 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。

下图显示了这个实例

图 16.38: 霍尔传感器接口的实例



16.3.19　TIM1 定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器 (TIMx_ARR，TIMx_CCRx) 都被更新了。在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽 (在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即

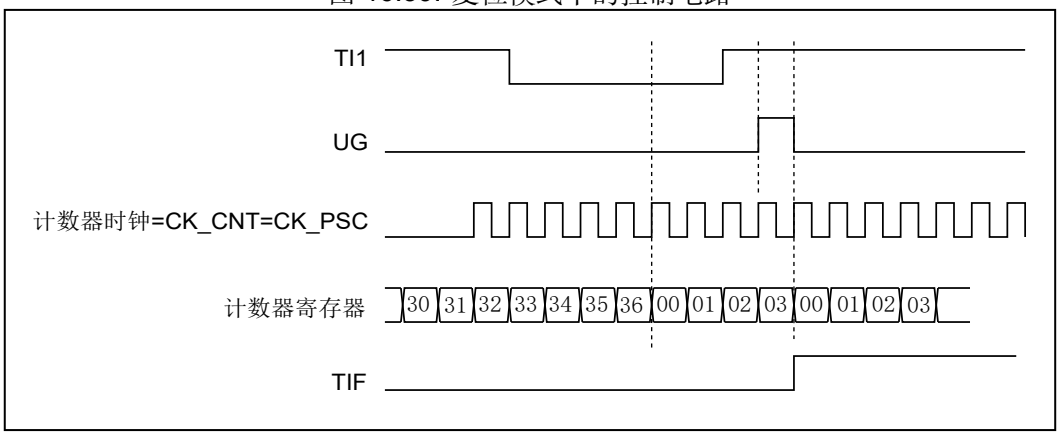
TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性 (只检测上升沿)。

- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志 (TIMx_SR 寄存器中的 TIF 位) 被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能) 位和 TDE(DMA 使能) 位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 16.39: 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。

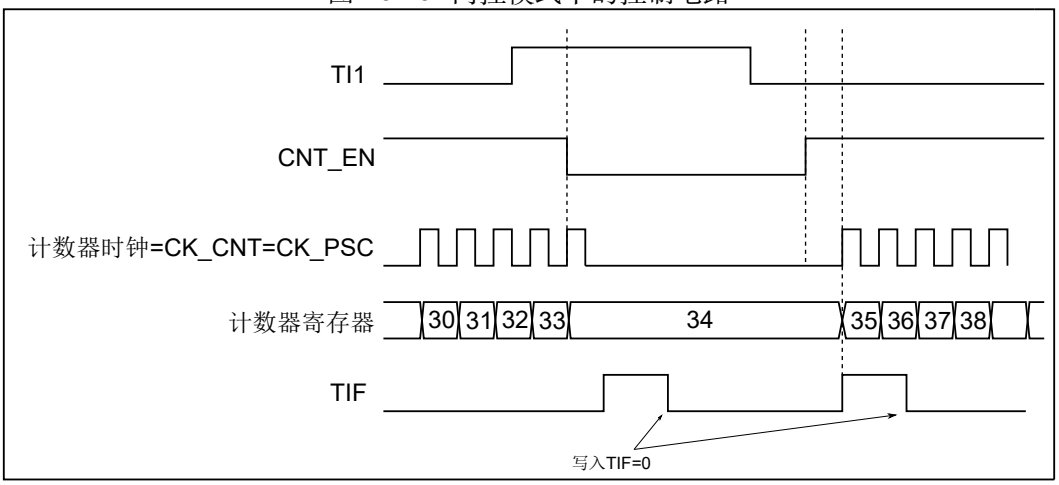
在如下的例子中，计数器只在 TI1 为低时向上计数：

- 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 16.40: 门控模式下的控制电路



textbf 从模式：触发模式

输入端上选中的事件使能计数器。

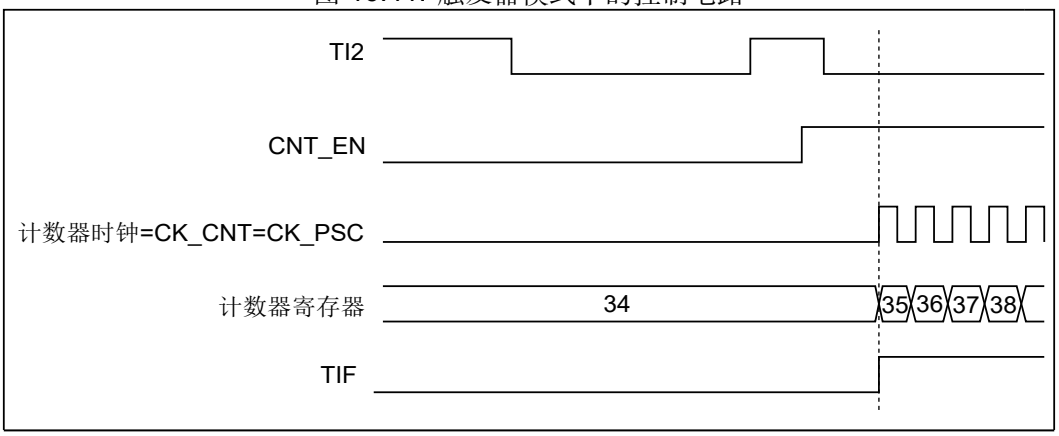
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

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

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 16.41: 触发器模式下的控制电路



textbf 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：

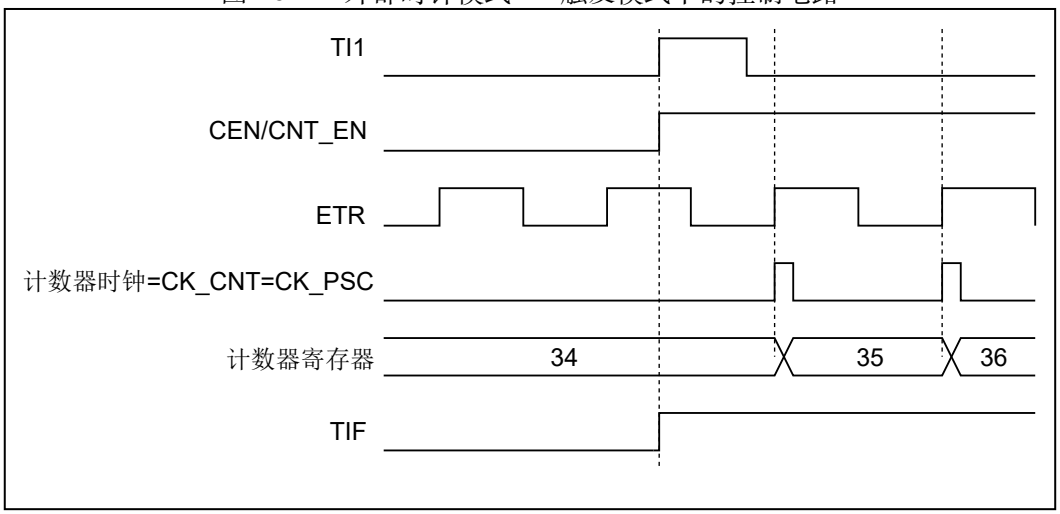
- IC1F=0000：没有滤波 ─ 触发操作中不使用捕获预分频器，不需要配置
- 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
- 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性 (只检测上升沿)

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 16.42: 外部时钟模式 2+ 触发模式下的控制电路



16.3.20　定时器同步

所有 TIM 定时器在内部相连，用于定时器同步或链接。详见17.3.15节。

16.3.21　调试模式

当微控制器进入调试模式时 (Cortex-M3 核心停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。详见31.13.2节。

16.4　高级控制定时器 TIM1 寄存器描述

16.4.1　TIM1 控制寄存器 1 (TIMx_CR1)

地址偏移：0x00

复位值：0x0000 0000

位	符号	访问	描述
31:10	-	R	保留。始终读为 0。
9:8	CKD[1:0]	RW	时钟分频因子 (Clock division) 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：$t_{DTS} = 2xt_{CK\_INT}$ 10：$t_{DTS} = 4xt_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	自动重装载预装载允许位 (Auto-reload preload enable) 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
6:5	CMS[1:0]	RW	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。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时 (CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	方向 (Direction) 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	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件 (清除 CEN 位) 时，计数器停止。
2	URS	RW	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： • 计数器溢出/下溢 • 设置 UG 位 • 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。

1	UDIS	RW	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新 (UEV) 事件由下述任一事件产生： • 计数器溢出/下溢 • 设置 UG 位 • 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	使能计数器 (Counter enable) 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。 在单脉冲模式下，当发生更新事件时，CEN 被自动清除。

16.4.2　TIM1 控制寄存器 2 (TIMx_CR2)

地址偏移：0x04

复位值：0x0000 0000

位	符号	访问	描述
31:15	-	R	保留。始终读为 0。
14	OIS4	RW	输出空闲状态 4(OC4 输出)。参见 OIS1 位。
13	OIS3N	RW	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位。
12	OIS3	RW	参见 OIS1 位。
11	OIS2N	RW	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
10	OIS2	RW	输出空闲状态 2(OC2 输出)。参见 OIS1 位。
9	OIS1N	RW	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器) 级别 1、2 或 3 后，该位不能被修改。
8	OIS	RW	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器) 级别 1、2 或 3 后，该位不能被修改。
7	TI1S	RW	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。

6:4	MMS[2:0]	RW	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下： 000：复位–TIMx_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。 001：使能–计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。 010：更新–更新事件被选为触发输入 (TRGO)。 011：比较脉冲–在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时 (即使它已经为高)，触发输出送出一个正脉冲 (TRGO)。 100：比较–OC1REF 信号被用于作为触发输出 (TRGO)。 101：比较–OC2REF 信号被用于作为触发输出 (TRGO)。 110：比较–OC3REF 信号被用于作为触发输出 (TRGO)。 111：比较–OC4REF 信号被用于作为触发输出 (TRGO)。
3	CCDS	RW	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	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 (通道 1,2,3)。
1	-	R	保留。始终读为 0。
0	CCPC	RW	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。

16.4.3　TIM1 从模式控制寄存器 (TIMx_SMCR)

地址偏移：0x08

复位值：0x0000 0000

位	符号	访问	描述
31-16	-	R	保留。始终读为 0。
15	ETP	RW	外部触发极性 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。

14	ECE	RW	外部时钟使能位 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111) 具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是’111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS[1:0]	RW	外部触发预分频 外部触发信号 ETRP 的频率不能超过 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
11:8	ETF[3:0]	RW	外部触发滤波 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。 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 $f_{SAMPLING} = fCK_INT$，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
7	MSM	RW	主/从模式 0：无作用； 1：触发输入 (TRGI) 上的事件被延迟了，以实现当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。

6:4	TS[2:0]	RW	触发选择 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0(ITR0) 100：TI1 的边沿检测器 (TI1F_ED) 001：内部触发 1(ITR1) 101：滤波后的定时器输入 1(TI1FP1) 010：内部触发 2(ITR2) 110：滤波后的定时器输入 2(TI2FP2) 011：内部触发 3(ITR3) 111：外部触发输入 (ETRF) 更多有关 ITRx 的细节，参见表16.4。注：这些位只能在未用到 (如 SMS=000) 时被改变，以避免在改变时产生错误的边沿检测。
3	-	R	保留，始终读为 0。
2:0	SMS[2:0]	RW	从模式选择 000：关闭从模式–如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 –根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 –根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 –根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式–选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式–当触发输入 (TRGI) 为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止 (但不复位)。计数器的启动和停止都是受控的。 110：触发模式–计数器在触发输入 TRGI 的上升沿启动 (但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 –选中的触发输入 (TRGI) 的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入 (TS=100) 时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表 16.4: TIMER 内部触发连接

从定时器	IRX0(TS=000)	IRX1(TS=001)	IRX2(TS=010)	IRX3(TS=011)
TIM1	0	TIM2	TIM3	TIM4
TIM2	TIM1	0	TIM3	TIM4
TIM3	TIM1	TIM2	0	TIM4
TIM4	TIM1	TIM2	TIM3	0

16.4.4　TIM1 DMA/中断使能寄存器 (TIMx_DIER)

地址偏移：0x0C

复位值：0x0000 0000

位	符号	访问	描述
31:15	-	R	保留，始终读为 0。

14	TDE	RW	允许触发 DMA 请求 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13	COMDE	RW	允许 COM 的 DMA 请求 0：禁止 COM 的 DMA 请求； 1：允许 COM 的 DMA 请求。
12	CC4DE	RW	允许捕获/比较 4 的 DMA 请求 0：禁止捕获/比较 4 的 DMA 请求； 1：允许捕获/比较 4 的 DMA 请求。
11	CC3DE	RW	允许捕获/比较 3 的 DMA 请求 0：禁止捕获/比较 3 的 DMA 请求； 1：允许捕获/比较 3 的 DMA 请求。
10	CC2DE	RW	允许捕获/比较 2 的 DMA 请求 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。
9	CC1DE	RW	允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	允许更新的 DMA 请求 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。
7	BIE	RW	允许刹车中断 0：禁止刹车中断； 1：允许刹车中断。
6	TIE	RW	触发中断使能 0：禁止触发中断； 1：使能触发中断。
5	COMIE	RW	允许 COM 中断 0：禁止 COM 中断； 1：允许 COM 中断。
4	CC4IE	RW	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	允许捕获/比较 3 中断 0：禁止捕获/比较 3 中断； 1：允许捕获/比较 3 中断。
2	CC2IE	RW	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	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	允许更新中断 0：禁止更新中断； 1：允许更新中断。

16.4.5　TIM1 状态寄存器 (TIMx_SR)

地址偏移：0x10

复位值：0x0000 0000

位	符号	访问	描述
31:13	-	R	保留，始终读为 0。
12	CC4OF	RC_W0	捕获/比较 4 重复捕获标记 参见 CC1OF 描述。
11	CC3OF	RC_W0	捕获/比较 3 重复捕获标记 参见 CC1OF 描述。
10	CC2OF	RC_W0	捕获/比较 2 重复捕获标记 参见 CC1OF 描述。
9	CC1OF	RC_W0	捕获/比较 1 重复捕获标记 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：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
8	-	R	保留，始终读为 0。
7	BIF	RC_W0	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	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，或门控模式下的任一边沿) 时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	RC_W0	COM 中断标记 一旦产生 COM 事件 (当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新) 该位由硬件置’1’。它由软件清’0’。 0：无 COM 事件产生； 1：COM 中断等待响应。
4	CC4IF	RC_W0	捕获/比较 4 中断标记 参考 CC1IF 描述。
3	CC3IF	RC_W0	捕获/比较 3 中断标记 参考 CC1IF 描述。
2	CC2IF	RC_W0	捕获/比较 2 中断标记 参考 CC1IF 描述。

1	CC1IF	RC_W0	捕获/比较 1 中断标记 **如果通道 CC1 配置为输出模式：** 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外 (参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 **如果通道 CC1 配置为输入模式：** 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	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：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： • 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时 (重复计数器 =0 时产生更新事件)。 • 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 • 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

16.4.6　TIM1 事件产生寄存器 (TIMx_EGR)

地址偏移：0x14

复位值：0x0000 0000

位	符号	访问	描述
31:8	-	R	保留，始终读为 0。
7	BG	W	产生刹车事件 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
6	TG	W	产生触发事件 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。

5	COMG	W	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：该位只对拥有互补输出的通道有效。
4	CC4G	W	产生捕获/比较 4 事件 参考 CC1G 描述。
3	CC3G	W	产生捕获/比较 3 事件 参考 CC1G 描述。
2	CC2G	W	产生捕获/比较 2 事件 参考 CC1G 描述。
1	CC1G	W	产生捕获/比较 1 事件 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出：设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入：当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	产生更新事件 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’，但是预分频系数不变。 若在中心对称模式下或 DIR=0(向上计数) 则计数器被清’0’； 若 DIR=1(向下计数) 则计数器取 TIMx_ARR 的值。

16.4.7　TIM1 捕获/比较模式寄存器 1 (TIMx_CCMR1)

地址偏移：0x18

复位值：0x0000 0000

通道可用于输入 (捕获模式) 或输出 (比较模式)，通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15	OC2CE	RW	输出比较 2 清 0 使能
14:12	OC2M[2:0]	RW	输出比较 2 模式
11	OC2PE	RW	输出比较 2 预装载使能
10	OC2FE	RW	输出比较 2 快速使能

9:8	CC2S[1:0]	RW	捕获/比较 2 选择。 该位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC2E=0) 才是可写的。
7	OC1CE	RW	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	输出比较 1 模式 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1) 相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1) 相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平 (OC1REF=0)，否则为有效电平 (OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位) 并且 CC1S=00(该通道配置成输出) 则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。

3	OC1PE	RW	输出比较 1 预装载使能 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位) 并且 CC1S=00(该通道配置成输出) 则该位不能被修改。 注 2：仅在单脉冲模式下 (TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	输出比较 1 快速使能 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S[1:0]	RW	捕获/比较 1 选择。 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。

输入捕获模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15:12	IC2F[3:0]	RW	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	输入/捕获 2 预分频器
9:8	CC2S[1:0]	RW	捕获/比较 2 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC2E=0) 才是可写的。

7:4	IC1F[3:0]	RW	输入捕获 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 $f_{SAMPLING} = f_{DTS}/8$，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	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，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	捕获/比较 1 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。

16.4.8　TIM1 捕获/比较模式寄存器 2 (TIMx_CCMR2)

地址偏移：0x1C

复位值：0x0000 0000

参看以上 CCMR1 寄存器的描述

输出比较模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15	OC4CE	RW	输出比较 4 清 0 使能

14:12	OC4M[2:0]	RW	输出比较 4 模式
11	OC4PE	RW	输出比较 4 预装载使能
10	OC4FE	RW	输出比较 4 快速使能
9:8	CC4S[1:0]	RW	捕获/比较 4 选择 该 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC4E=0) 才是可写的。
7	OC3CE	RW	输出比较 3 清 0 使能
6:4	OC3M[2:0]	RW	输出比较 3 模式
3	OC3PE	RW	输出比较 3 预装载使能
2	OC3FE	RW	输出比较 3 快速使能
1:0	CC3S[1:0]	RW	捕获/比较 3 选择 该 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC3E=0) 才是可写的。

输入捕获模式：

位	符号	访问	描述
31:16	保留，始终读为 0。		
15:12	IC4F[3:0]	RW	输入捕获 4 滤波器
11:10	IC4PSC[1:0]	RW	输入/捕获 2 预分频器
9:8	CC4S[1:0]	RW	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：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI24 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC4E=0) 才是可写的。
7:4	IC3F[3:0]	RW	输入捕获 3 滤波器
3:2	IC3PSC[1:0]	RW	输入/捕获 3 预分频器

1:0	CC3S[1:0]	RW	捕获/比较 3 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC3E=0) 才是可写的。

16.4.9　TIM1 捕获/比较使能寄存器 (TIMx_CCER)

地址偏移：0x20

复位值：0x0000 0000

位	符号	访问	描述
31:14	保留，始终读为 0。		
13	CC4P	RW	输入/捕获 4 输出极性 参考 CC1P 的描述。
12	CC4E	RW	输入/捕获 4 输出使能 参考 CC1E 的描述。
11	CC3NP	RW	输入/捕获 3 互补输出极性 参考 CC1NP 的描述。
10	CC3NE	RW	输入/捕获 3 互补输出使能 参考 CC1NE 的描述。
9	CC3P	RW	输入/捕获 3 输出极性 参考 CC1P 的描述。
8	CC3E	RW	输入/捕获 3 输出使能 参考 CC1E 的描述。
7	CC2NP	RW	输入/捕获 2 互补输出极性 参考 CC1NP 的描述。
6	CC2NE	RW	输入/捕获 2 互补输出使能 参考 CC1NE 的描述。
5	CC2P	RW	输入/捕获 2 输出极性 参考 CC1P 的描述。
4	CC2E	RW	输入/捕获 2 输出使能 参考 CC1E 的描述。
3	CC1NP	RW	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 级别设为 3 或 2 且 CC1S=00, 则该位不能被修改。

2	CC1NE	RW	输入/捕获 1 互补输出使能 0：关闭 - OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1	CC1P	RW	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：** 该位选择是 IC1 还是 IC1 的反相信号作为触发或捕获信号。 0：不反相：捕获发生在 IC1 的上升沿；当用作外部触发器时，IC1 不反相。 1：反相：捕获发生在 IC1 的下降沿；当用作外部触发器时，IC1 反相。 注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位) 设为 3 或 2，则该位不能被修改。
0	CC1E	RW	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 和 CC1NE 位的值。 1：开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

图 16.43: 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE 位	OSSI 位	OSSR 位	CCxE 位	CCxNE 位	OCx输出状态	OCxN输出状态
1	X	0	0	0	输出禁止(与定时器断开) OCx=0,OCx_EN=0	输出禁止(与定时器断开) OCxN=0,OCxN_EN=0
		0	0	1	输出禁止(与定时器断开) OCx=0,OCx_EN=0	OCxREF+极性, OCNx=OCxREF xor CCNxp, OCxN_EN=1
		0	1	0	OCxREF+极性, OCx=OCxREF xor CCxp, OCx_EN=1	输出禁止(与定时器断开) OCxN=0,OCxN_EN=0
		0	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF反相+极性+死区, OCxN_EN=1
		1	0	0	输出禁止(与定时器断开) OCx=CCxP,OCx_EN=0	输出禁止(与定时器断开) OCxN=CCxNP,OCxN_EN=0
		1	0	1	关闭状态(输出使能且无效电平) OCx=CCxP,OCx_EN=1	OCxREF+极性, OCNx=OCxREF xor CCNxp, OCxN_EN=1
		1	1	0	OCxREF+极性, OCx=OCxREF xor CCxp, OCx_EN=1	关闭状态(输出使能且无效电平) OCxN=CCxNP,OCxN_EN=1
		1	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF反相+极性+死区, OCxN_EN=1
0	0	X	0	0	输出禁止(与定时器断开) 异步的:OCx=CCxP.OCx_EN=0.OCxN=CCxNP. OCxN_EN=0; 若时钟存在:经过一个死区时间后OCx=OISx, OCxN=OISxN. 假设OISx与OISxN并不都对应OCx和OCxN的有效电平	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	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OCx=OISx. OCxN=OISxN, 假设OISx与OISxN并不都对应OCx和OCxN的有效电平。	
	1		0	1		
	1		1	0		
	1		1	1		

1.如果一个通道的2个输出都没有使用(CCxE=CCxNE=0)，那么OISx, OISxN, CCxP和CCxNP都必须清零。

16.4.10　TIM1 计数器 (TIMx_CNT)

地址偏移：0x24

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	CNT[19:0]	RW	计数器的值

16.4.11　TIM1 预分频器 (TIMx_PSC)

地址偏移：0x28

复位值：0x0000 0000

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15:0	PSC[15:0]	RW	预分频器的值 计数器的时钟频率 (CK_CNT) 等于 $f_{CK_PSC}/(PSC[15:0]+1)$。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’。

16.4.12　TIM1 自动重装载寄存器 (TIMx_ARR)

地址偏移：0x2C

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	ARR[19:0]	RW	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16.3.3节：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。

16.4.13　TIM1 重复计数寄存器 (TIMx_RCR)

地址偏移：0x30

复位值：0x0000 0000

位	符号	访问	描述
31:8	-	R	保留，始终读为 0。
7:0	REP[7:0]	RW	重复计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件，并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1) 对应着： • 在边沿对齐模式下，PWM 周期的数目； • 在中心对称模式下，PWM 半周期的数目；

16.4.14　TIM1 捕获/比较寄存器 1 (TIMx_CCR1)

地址偏移：0x34

复位值：0x0000 0000

位	符号	访问	描述
31:20	保留，始终读为 0。		
19:0	CCR1[19:0]	RW	捕获/比较通道 1 的值 **若 CC1 通道配置为输出：** CCR1 包含了装入当前捕获/比较 1 寄存器的值 (预装载值)。如果在 TIMx_CCMR1 寄存器 (OC1PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。

16.4.15　TIM1 捕获/比较寄存器 2 (TIMx_CCR2)

地址偏移：0x38

复位值：0x0000 0000

位	符号	访问	描述
31:20	保留，始终读为 0。		
19:0	CCR2[19:0]	RW	捕获/比较通道 2 的值 **若 CC2 通道配置为输出：** CCR2 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR2 寄存器 (OC2PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 **若 CC2 通道配置为输入：** CCR2 包含了由上一次输入捕获 2 事件 (IC2) 传输的计数器值。

16.4.16　TIM1 捕获/比较寄存器 3 (TIMx_CCR3)

地址偏移：0x3C

复位值：0x0000 0000

位	符号	访问	描述
31:20	保留，始终读为 0。		

19:0	CCR3[19:0]	RW	捕获/比较通道 3 的值 **若 C3 通道配置为输出：** CCR3 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR3 寄存器 (OC3PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 **若 CC3 通道配置为输入：** CCR3 包含了由上一次输入捕获 3 事件 (IC3) 传输的计数器值。

16.4.17　TIM1 捕获/比较寄存器 4 (TIMx_CCR4)

地址偏移：0x40

复位值：0x0000 0000

位	符号	访问	描述
31:20	保留，始终读为 0。		
19:0	CCR4[19:0]	RW	捕获/比较通道 4 的值 **若 C4 通道配置为输出：** CCR4 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR4 寄存器 (OC4PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 **若 CC4 通道配置为输入：** CCR4 包含了由上一次输入捕获 4 事件 (IC4) 传输的计数器值。

16.4.18　TIM1 刹车和死区寄存器 (TIMx_BDTR)

地址偏移：0x44

复位值：0x0000 0000

位	符号	访问	描述
31:16	保留，始终读为 0。		
15	MOE	RW	主输出使能 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位 (TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。

14	AOE	RW	自动输出使能 0：MOE 只能被软件置’1’； 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别设为’1’，则该位不能被修改。
13	BKP	RW	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	刹车功能使能 0：禁止刹车输入 (BRK 及 CCS 时钟失效事件)； 1：开启刹车输入 (BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。 0：当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号 =0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号 =1。 注：一旦 LOCK 级别设为 2，则该位不能被修改。
10	OSSI	RW	空闲模式下“关闭状态”选择 该位用于当 MOE=0 且通道设为输出时。 0：当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号 =0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号 =1。 注：一旦 LOCK 级别设为 2，则该位不能被修改。
9:8	LOCK[1:0]	RW	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护；01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位 (一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位) 以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位 (一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。

7:0	DTG[7:0]	RW	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：$DTG[7:5]=0xx => DT=DTG[7:0]\times T_{dtg}\square T_{dtg}=T_{DTS}$； $DTG[7:5]=10x => DT=(64+DTG[5:0])\times T_{dtg}\square T_{dtg}=2\times T_{DTS}$； $DTG[7:5]=110 => DT=(32+DTG[4:0])\times T_{dtg}\square T_{dtg}=8\times T_{DTS}$； $DTG[7:5]=111 => DT=(32+DTG[4:0])\times T_{dtg}\square T_{dtg}=16\times T_{DTS}$； 注：一旦 LOCK 级别设为 1、2 或 3，则不能修改这些位。

注释：

根据锁定设置，AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0] 位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

第十七章　通用定时器 (TIMx)

17.1　TIMx 简介

通用定时器是一个通过可编程预分频器驱动的 20 位自动装载计数器构成。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 (输出比较和 PWM)。

使用定时器预分频器和 RCC 时钟控制预分频器，脉冲长度和波形周期可以在几个微秒到几个毫秒间调整。

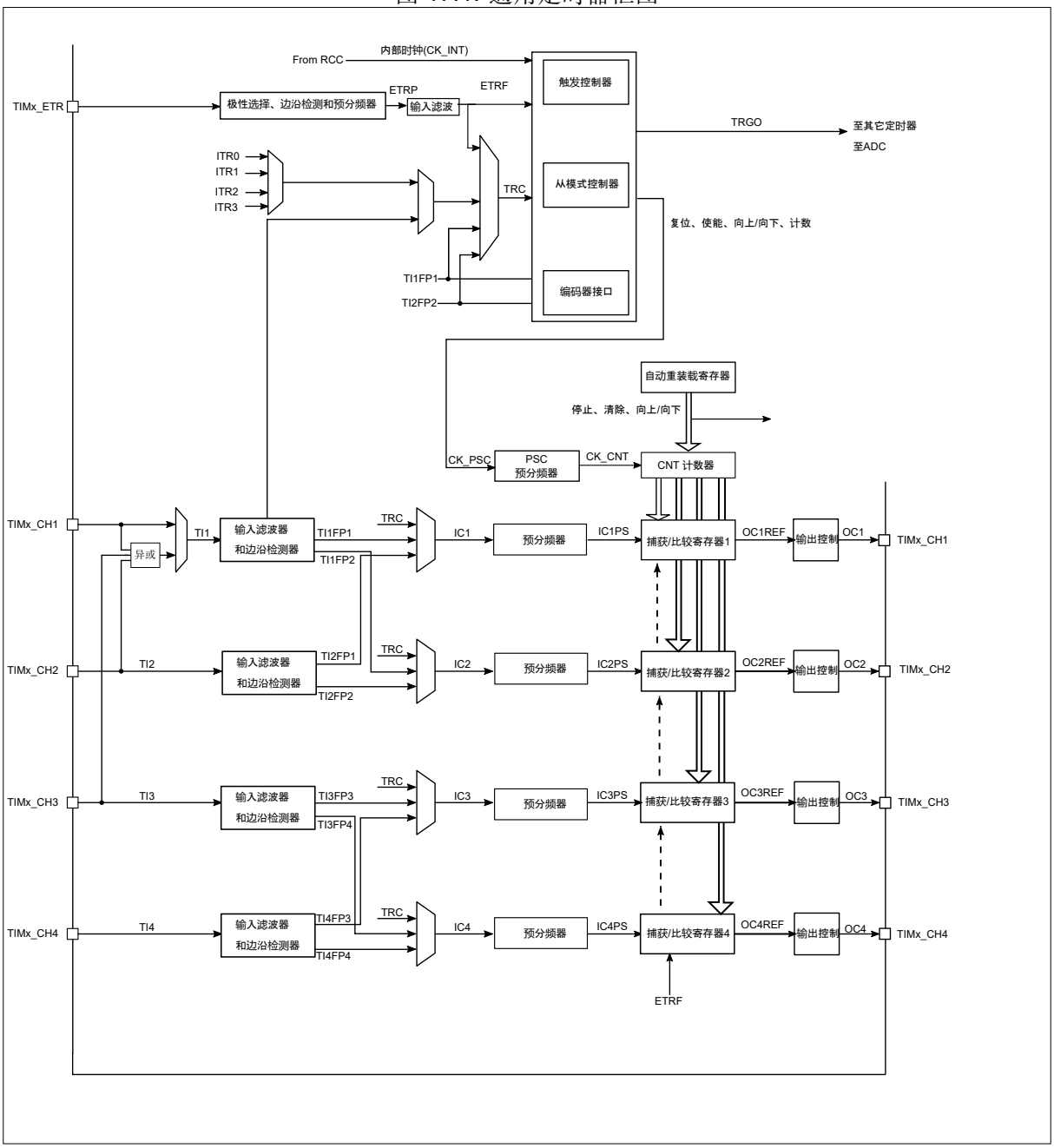
每个定时器都是完全独立的，没有互相共享任何资源。它们可以一起同步操作。

17.2　TIMx 主要功能

通用 TIMx (TIM2、TIM3、TIM4) 定时器功能包括：

- 20 位向上、向下、向上/下自动装载计数器
- 16 位可编程 (可以实时修改) 预分频器，计数器时钟频率的分频系数为 1 ~ 65535 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成 (边缘或中间对齐模式)
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化 (通过软件或者内部/外部触发)
  - 触发事件 (计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
- 支持针对定位的增量 (正交) 编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

图 17.1: 通用定时器框图



17.3　TIMx 功能描述

17.3.1　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。

时基单元包含：

- 计数器寄存器 (TIMx_CNT)

- 预分频器寄存器 (TIMx_PSC)
- 自动装载寄存器 (TIMx_ARR)
- 重复次数寄存器 (TIMx_RCR)

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

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位 (CEN) 时，CK_CNT 才有效。(更多有关使能计数器的细节，请参见控制器的从模式描述)。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频率按 1 到 65536 之间的任意值分频。它是基于一个 (在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。图17.2和图17.3给出了在预分频器运行时，更改计数器参数的例子。

图 17.2: 当预分频器的参数从 1 到 2 时，计数器的时序图

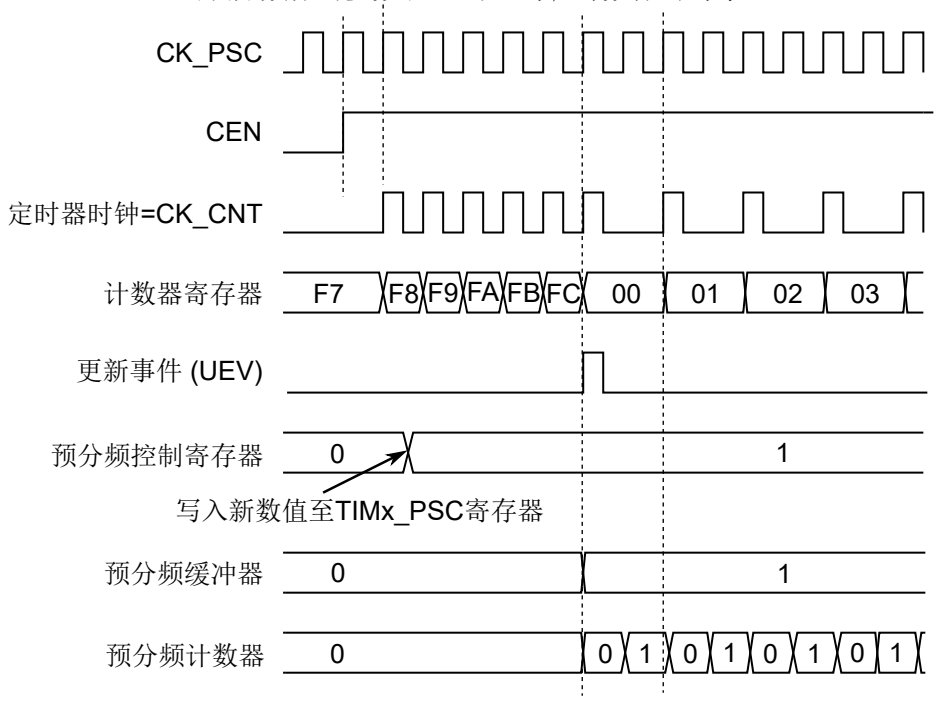
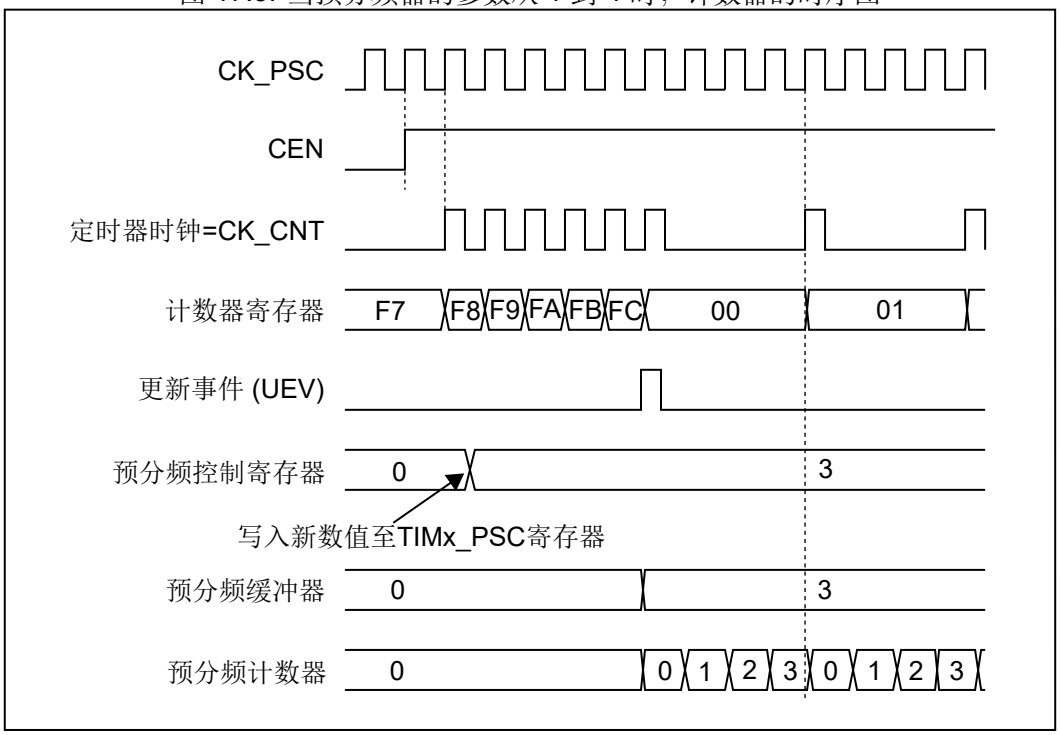


图 17.3: 当预分频器的参数从 1 到 4 时，计数器的时序图



17.3.2　计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值 (TIMx_ARR 计数器的内容)，然后重新从 0 开始计数并且产生一个计数器溢出事件。每次计数器溢出时可以产生更新事件，在 TIMx_EGR 寄存器中 (通过软件方式或者使用从模式控制器) 设置 UG 位也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清’0’之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清’0’，同时预分频器的计数也被清’0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志 (即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时 (依据 URS 位) 设置更新标志位 (TIMx_SR 寄存器中的 UIF 位)。

- 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)。
- 自动装载影子寄存器被重新置入预装载寄存器的值 (TIMx_ARR)。

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 17.4: 计数器时序图，内部时钟分频因子为 1

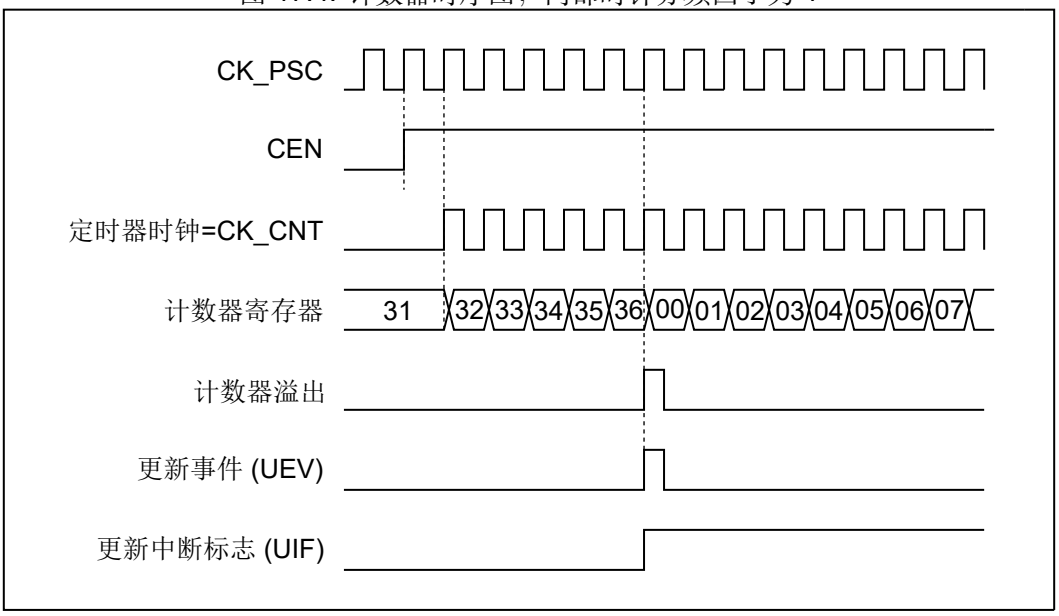


图 17.5: 计数器时序图，内部时钟分频因子为 2

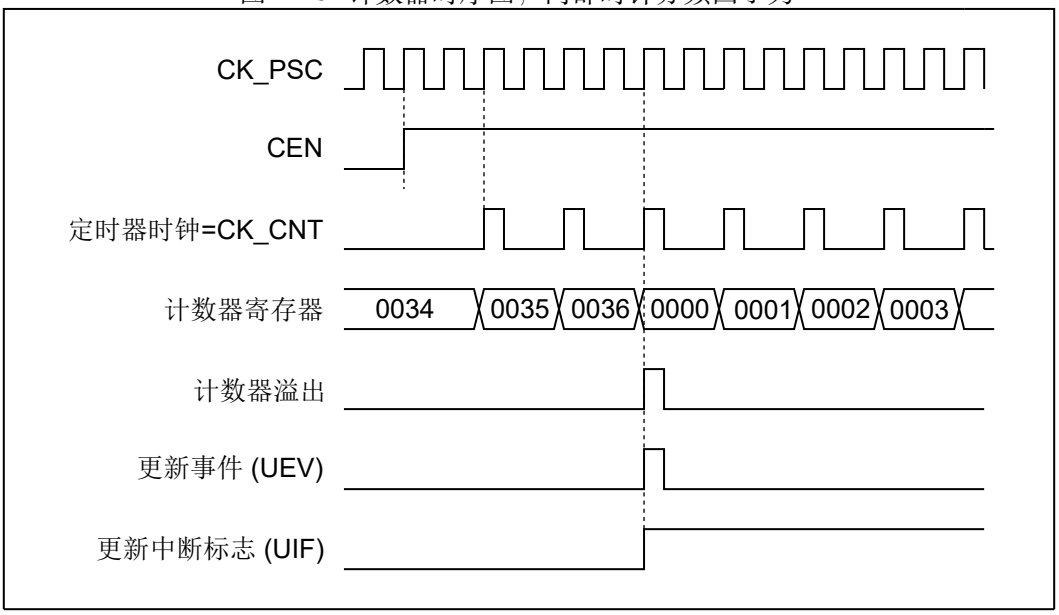


图 17.6: 计数器时序图，内部时钟分频因子为 4

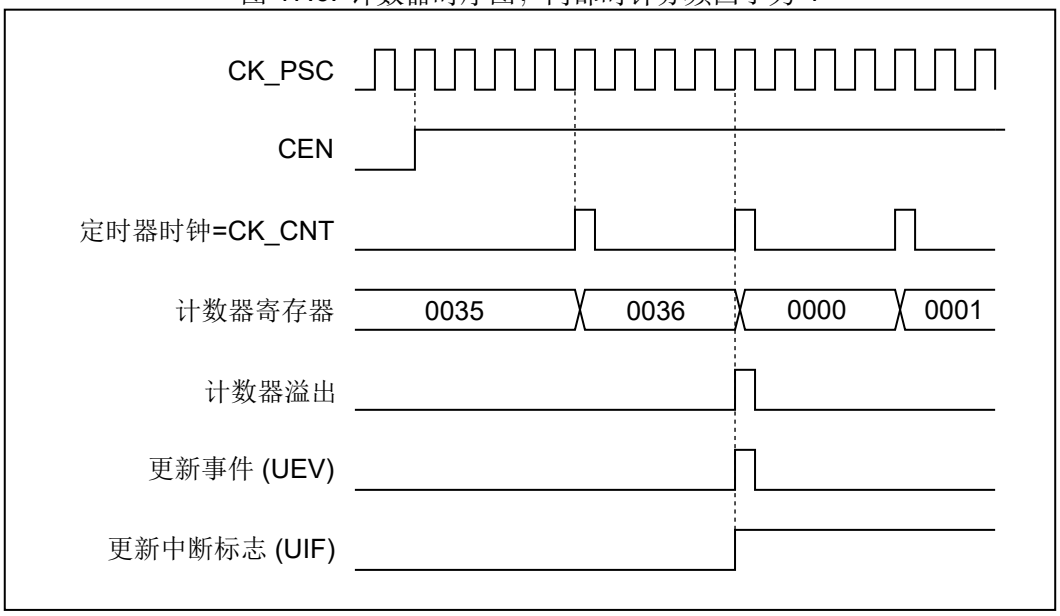


图 17.7: 计数器时序图，内部时钟分频因子为 N

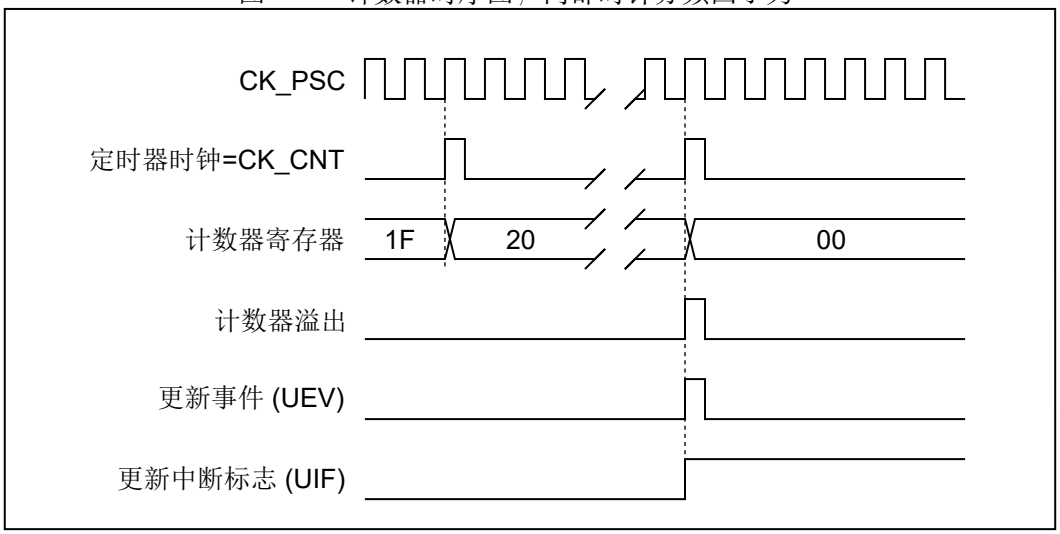


图 17.8: 计数器时序图，当 ARPE=0 时的更新事件 (TIMx_ARR 没有预装入)

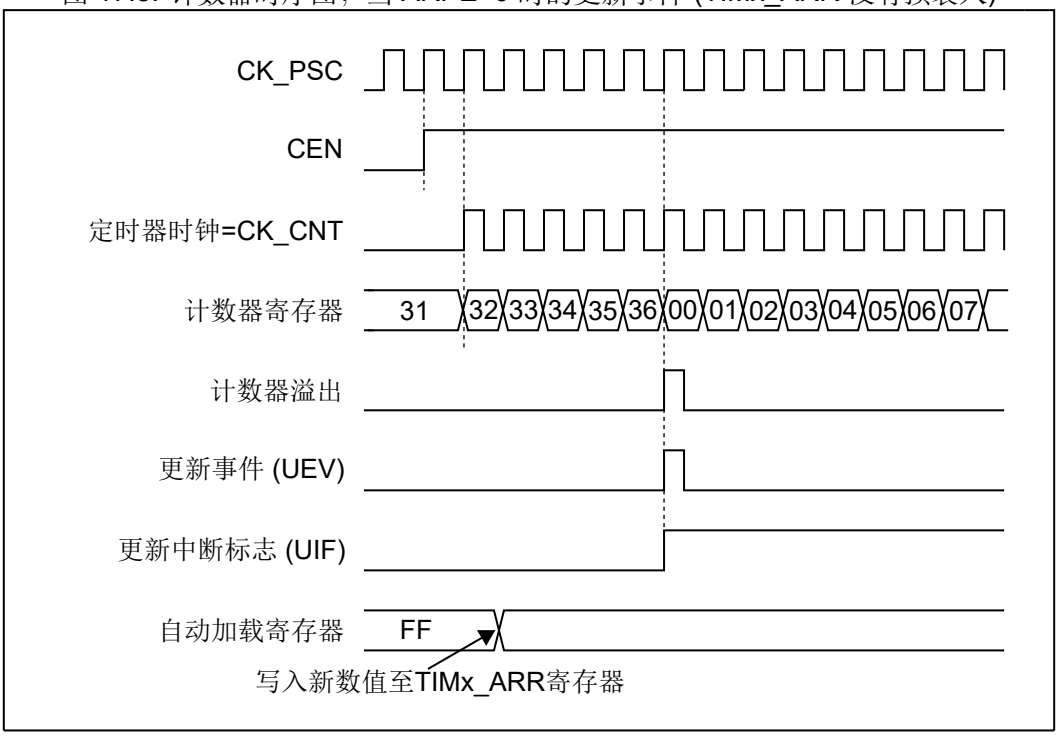
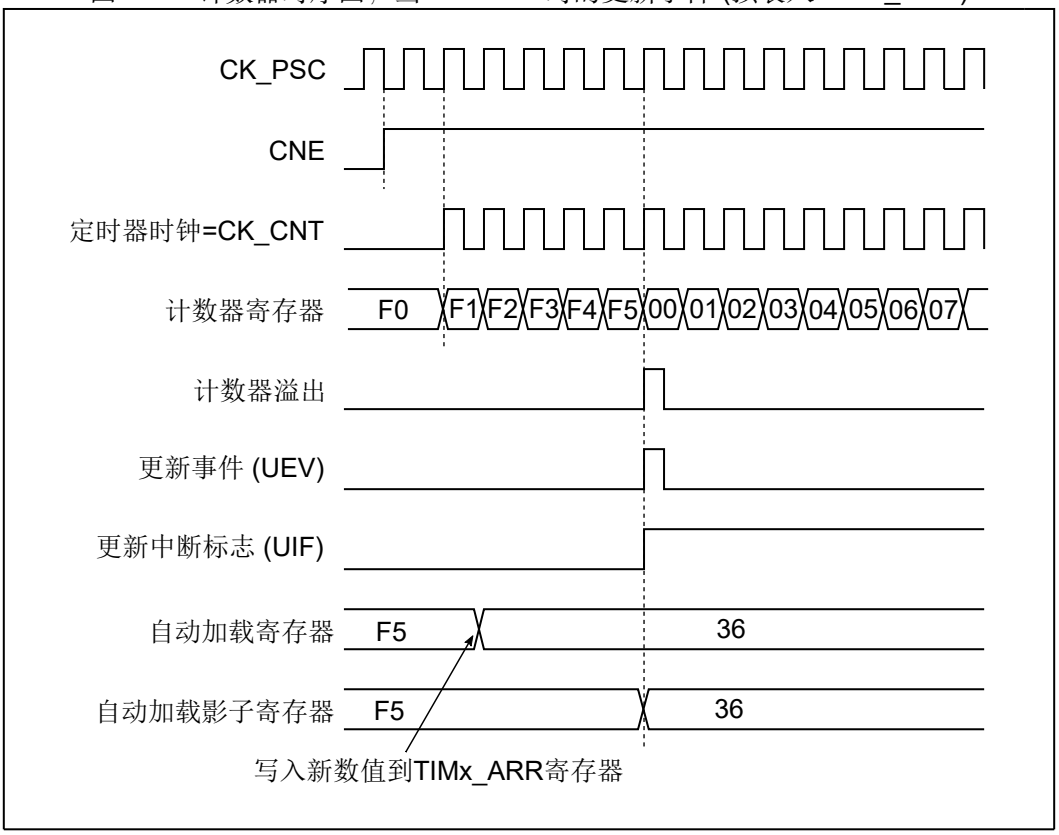


图 17.9: 计数器时序图，当 ARPE=1 时的更新事件 (预装入 TIMx_ARR)



向下计数模式

在向下模式中，计数器从自动装入的值 (TIMx_ARR 计数器的值) 开始向下计数到 0，然后从自动装入的

值重新开始并且产生一个计数器向下溢出事件。每次计数器溢出时可以产生更新事件，在 TIMx_EGR 寄存器中 (通过软件方式或者使用从模式控制器) 设置 UG 位也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始 (但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志 (因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且 (根据 URS 位的设置) 更新标志位 (TIMx_SR 寄存器中的 UIF 位) 也被设置。

- 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。

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

图 17.10: 计数器时序图，内部时钟分频因子为 1

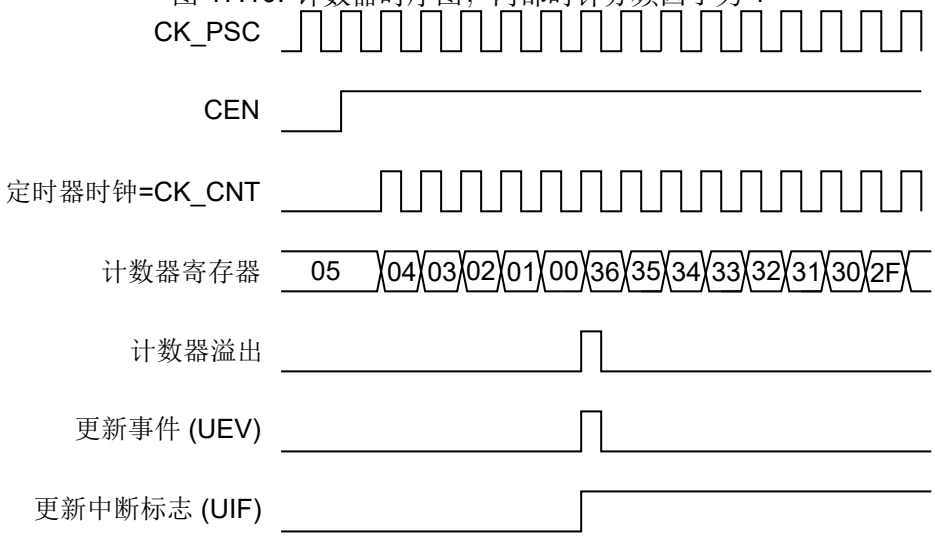


图 17.11: 计数器时序图，内部时钟分频因子为 2

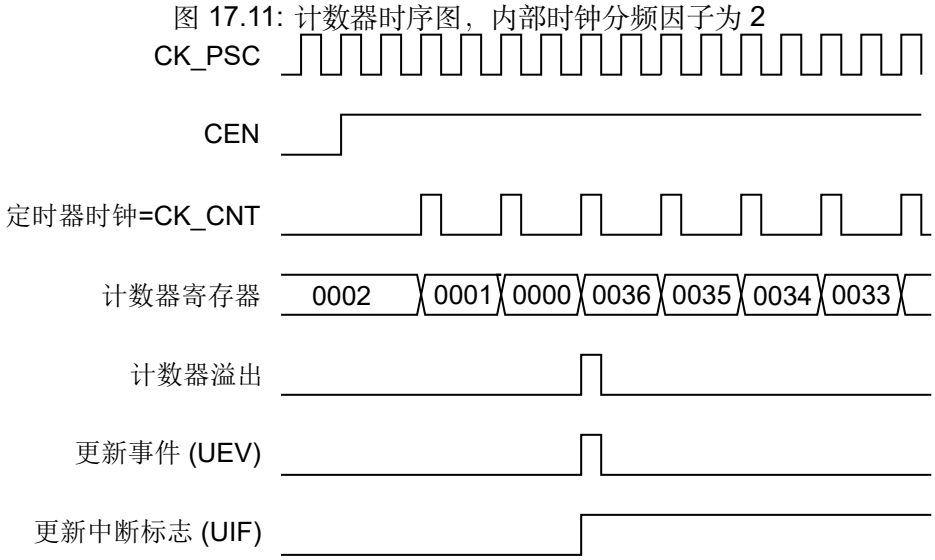


图 17.12: 计数器时序图，内部时钟分频因子为 4

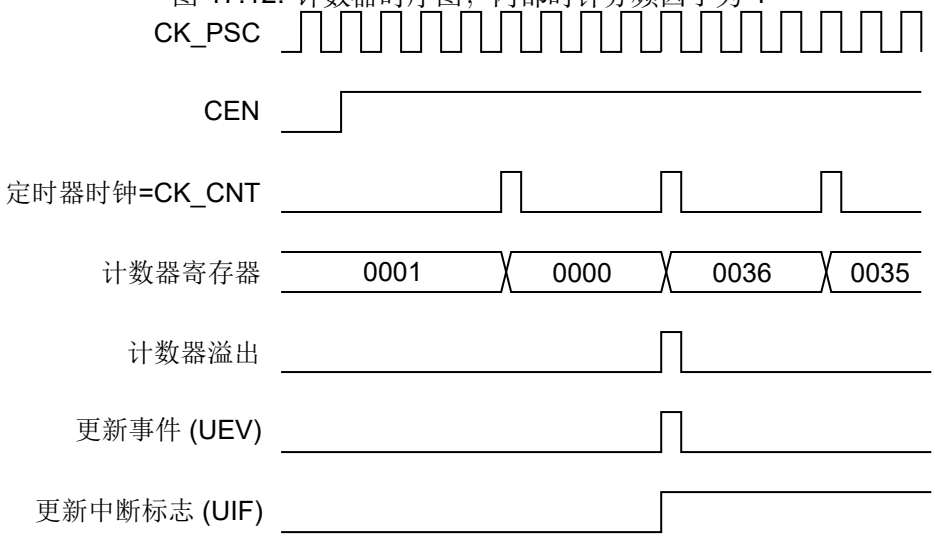


图 17.13: 计数器时序图，内部时钟分频因子为 N

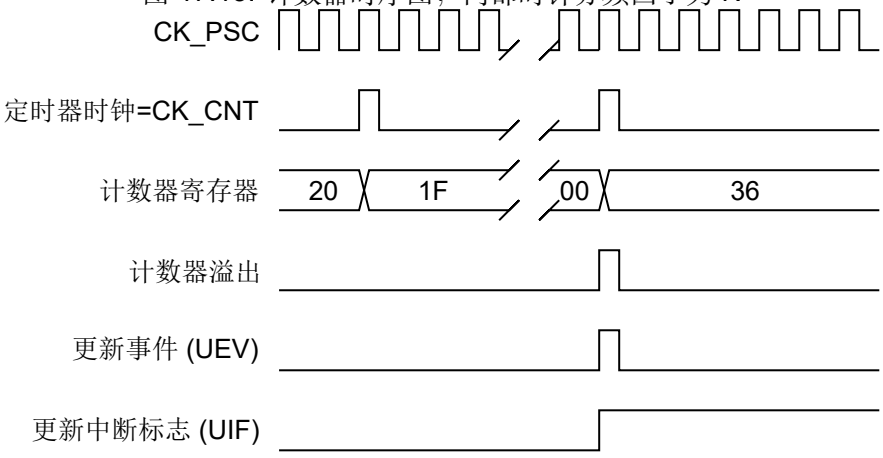
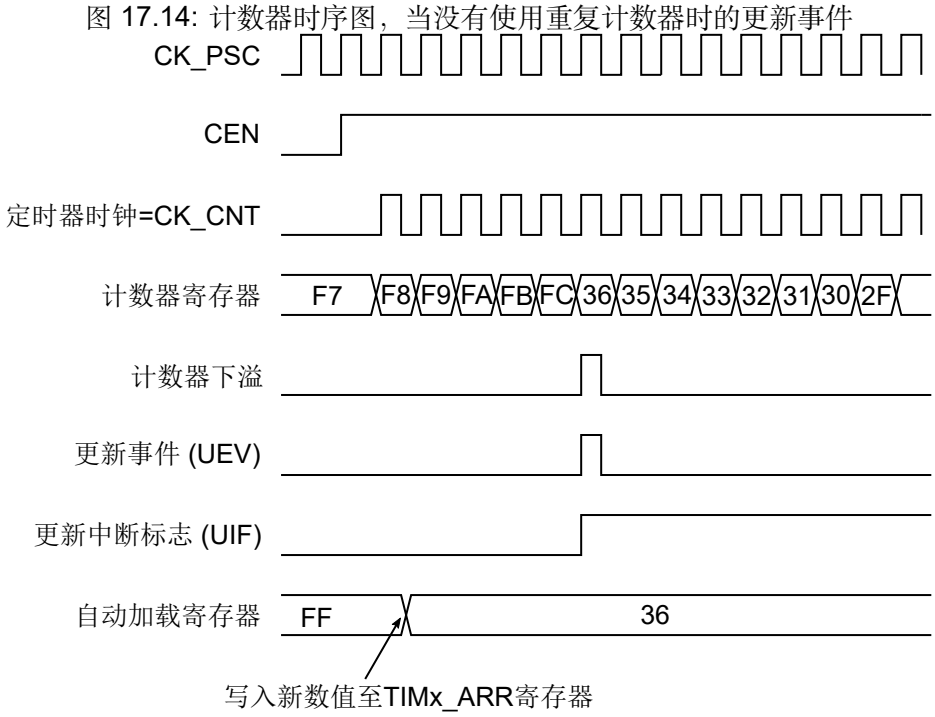


图 17.14: 计数器时序图，当没有使用重复计数器时的更新事件



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。在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过 (软件或者使用从模式控制器) 设置 TIMx_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位 (选择更新请求)，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志 (因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且 (根据 URS 位的设置) 更新标志位 (TIMx_SR 寄存器中的 UIF 位) 也被设置。

- 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)。

以下是一些计数器在中央对齐模式下不同时钟频率下的操作的例子：

图 17.15: 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

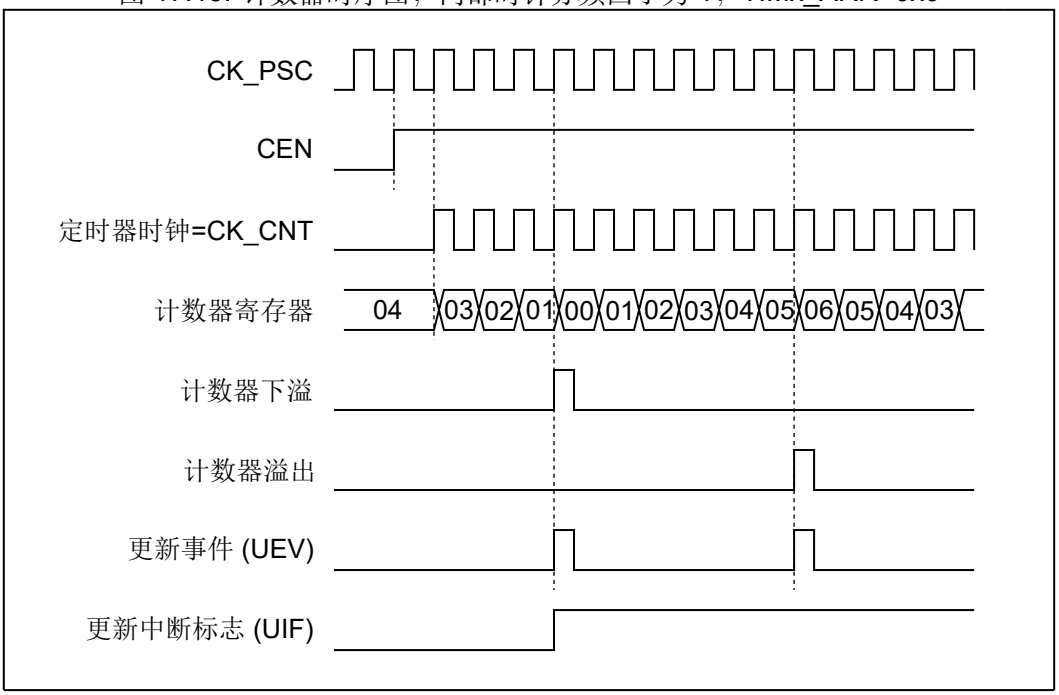


图 17.16: 计数器时序图，内部时钟分频因子为 2

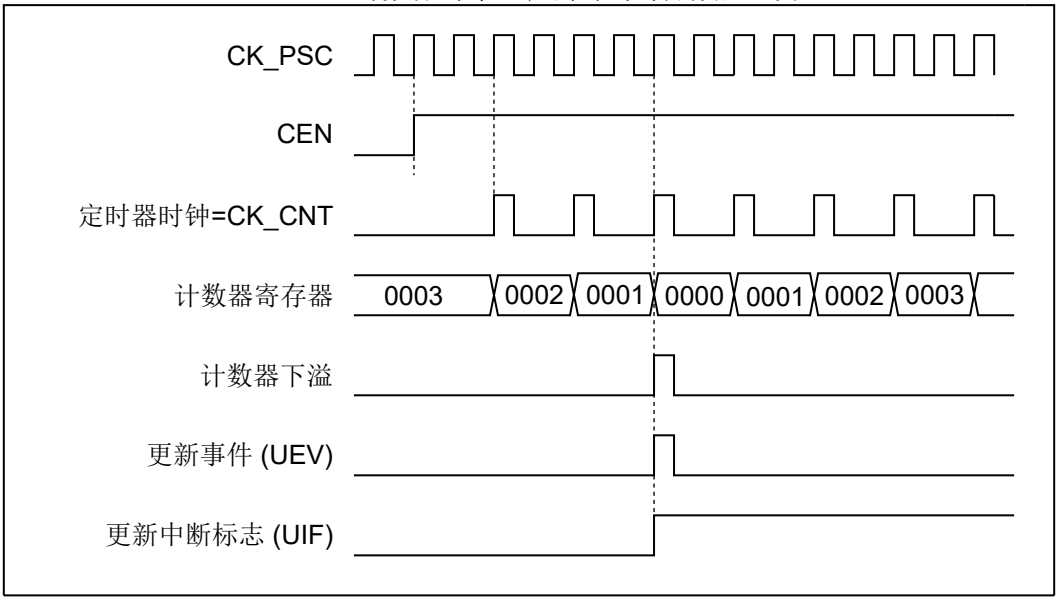
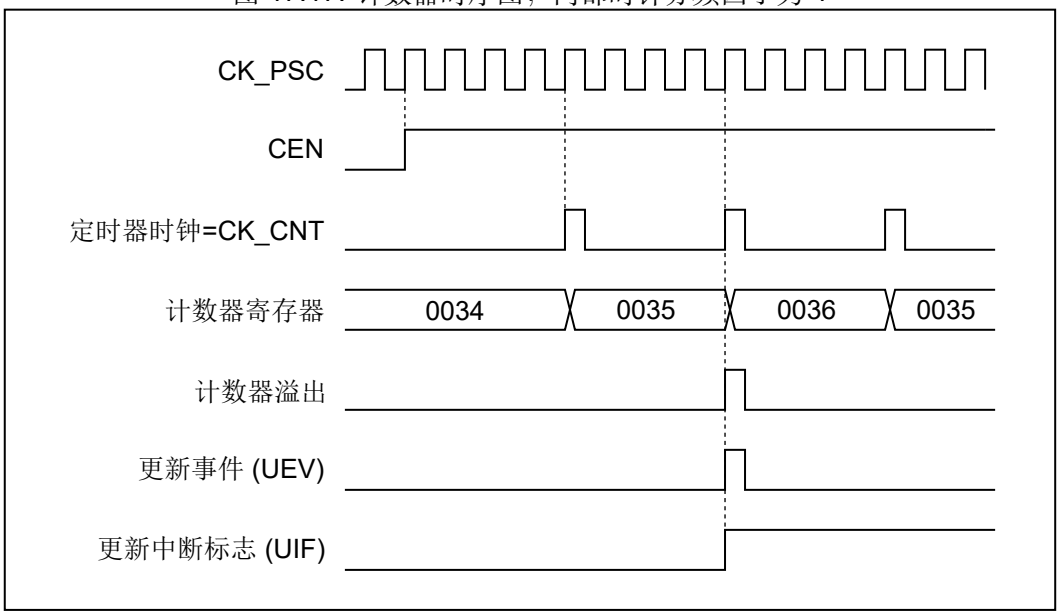


图 17.17: 计数器时序图，内部时钟分频因子为 4



注：中心对齐模式 *2* 或 *3* 是在溢出时与 *UIF* 标志一起使用。

图 17.18: 计数器时序图，内部时钟分频因子为 N

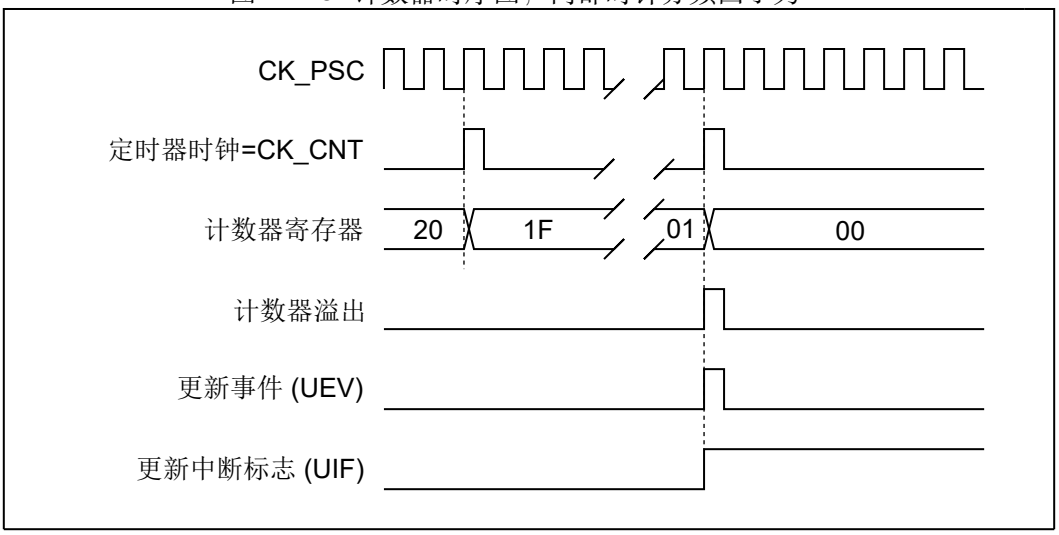


图 17.19: 计数器时序图，ARPE=1 时的更新事件 (计数器下溢)

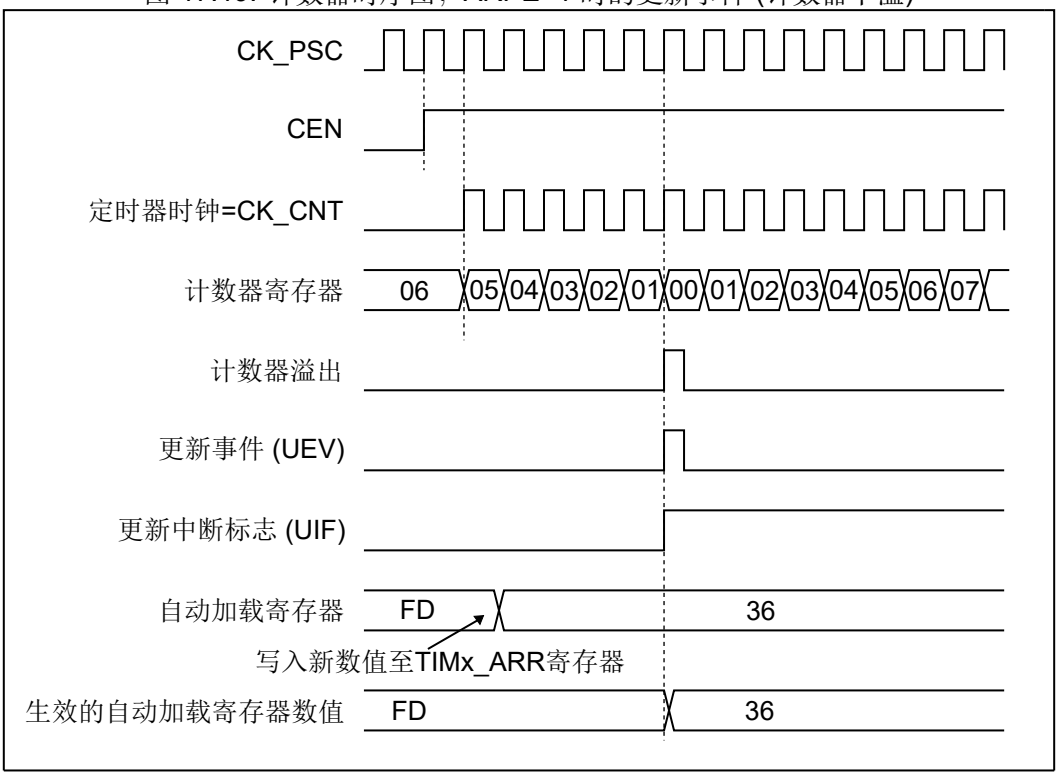
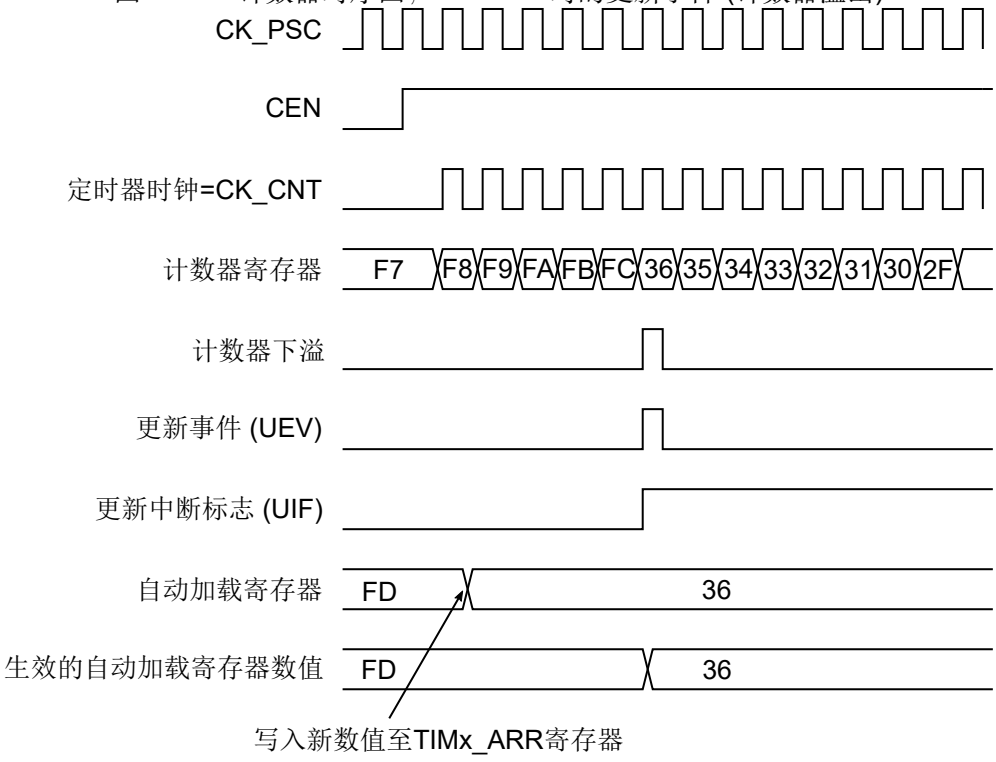


图 17.20: 计数器时序图，ARPE=1 时的更新事件 (计数器溢出)



17.3.3　时钟选择

计数器时钟可由下列时钟源提供:

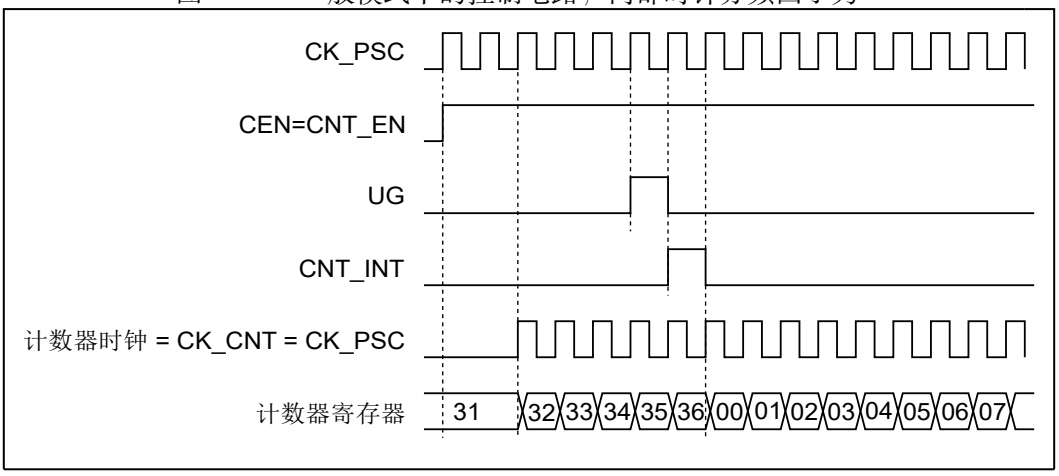
- 内部时钟 (CK_INT)
- 外部时钟模式 1：外部输入引脚
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入 (ITRx)：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。详见下一章。

内部时钟源 (CK_INT)

如果禁止了从模式控制器 (SMS=000)，则 CEN、DIR(TIMx_CR1 寄存器) 和 UG 位 (TIMx_EGR 寄存器) 是事实上的控制位，并且只能被软件修改 (UG 位仍被自动清除)。只要 CEN 位被写成’1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

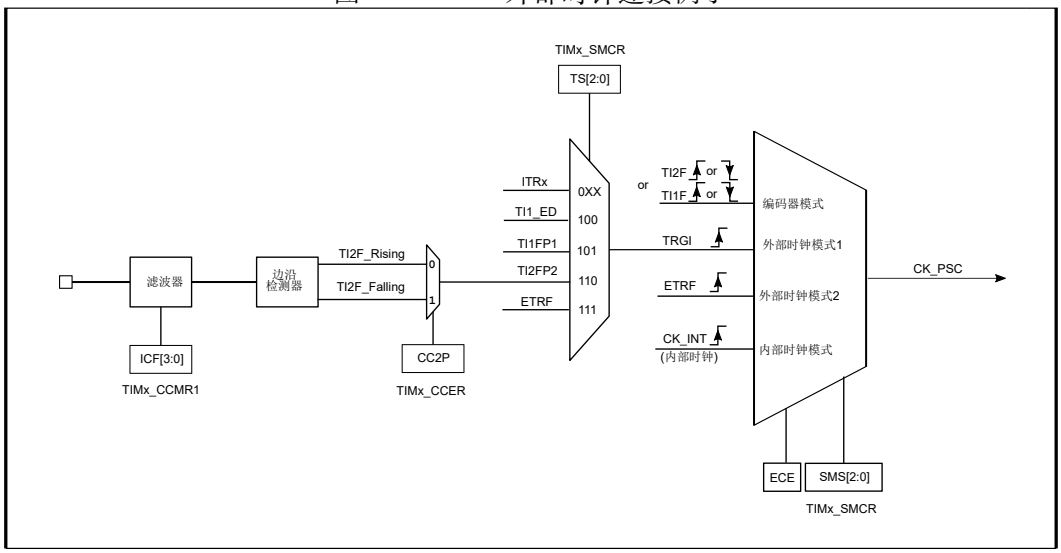
图 17.21: 一般模式下的控制电路，内部时钟分频因子为 1



外部时钟源模式 1

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

图 17.22: TI2 外部时钟连接例子



例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

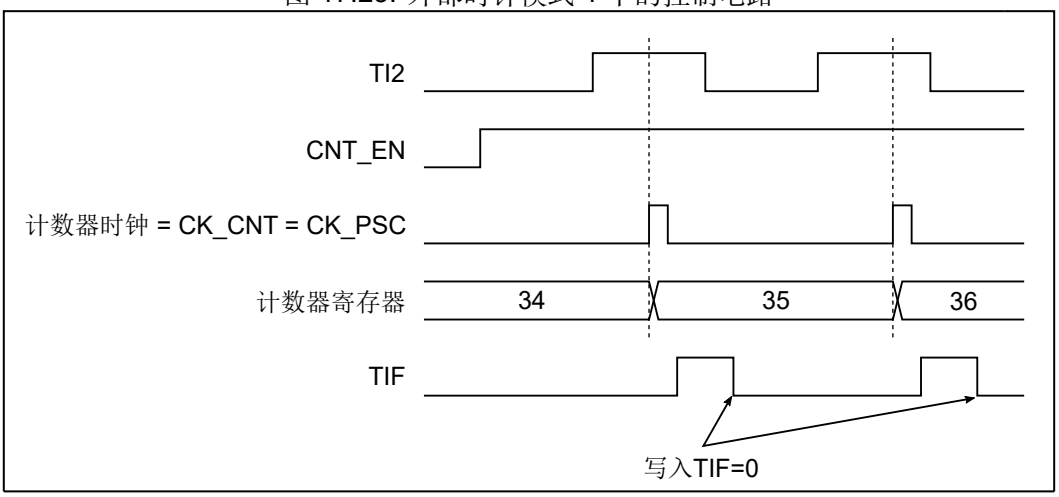
1. 配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿
2. 配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽 (如果不需要滤波器，保持 IC2F=0000)
3. 配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性
4. 配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1
5. 配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源
6. 设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注：捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

图 17.23: 外部时钟模式 1 下的控制电路



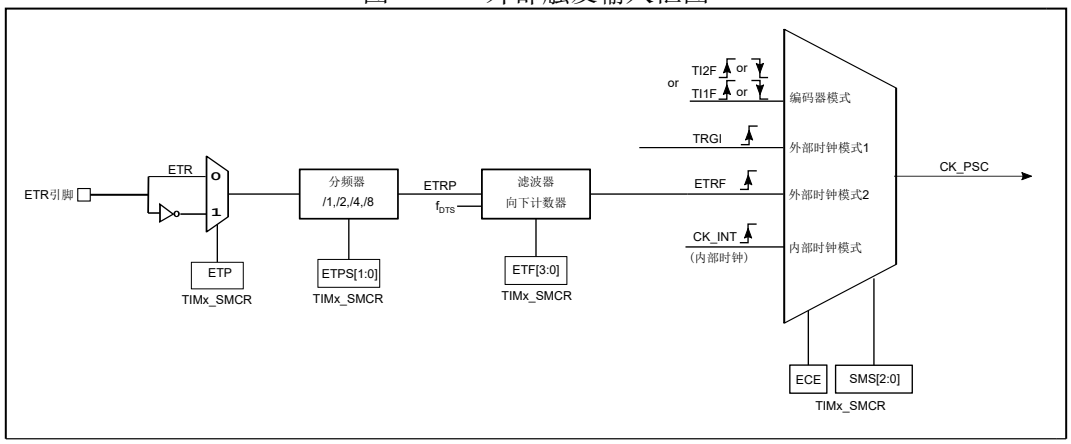
外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1

计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图

图 17.24: 外部触发输入框图



实例：配置计数器为在 ETR 下每 2 个上升沿计数一次的向上计数器

1. 在本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000

2. 设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0] = 01
3. 选择 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0
4. 开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1
5. 启动计数器，写 TIMx_CR1 寄存器中的 CEN=1

17.3.4　捕获/比较通道

TIMx 包含四个捕获/比较通道。每一个捕获/比较通道都是围绕着一个捕获/比较寄存器 (包含影子寄存器)，包括捕获的输入部分 (数字滤波、多路复用和预分频器)，和输出部分 (比较器和输出控制)。

输入部分对相应的 TIx 输入信号采样，通过滤波器后产生信号 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。该信号通过预分频器之后进入捕获寄存器。

输出部分产生一个中间波形 OCxRef(高有效) 作为基准，链的末端决定终输出信号的极性。在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

17.3.5　输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 (TIMx_CCRx) 中。当发生捕获事件时，相应的 CCxIF 标志 (TIMx_SR 寄存器) 被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器) 被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽 (即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以 (以 fDTS 频率) 连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止 (写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置 (中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。

- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注：设置 *TIMx_EGR* 寄存器中相应的 *CCxG* 位，可以通过软件产生输入捕获中断和/或 *DMA* 请求。

17.3.6　PWM 输入模式

PWM 输入模式是一种特殊的输入模式，它有以下两个区别：

- 两个 ICx 信号被映射至同一个 TIx 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用通道 1 和通道 2。

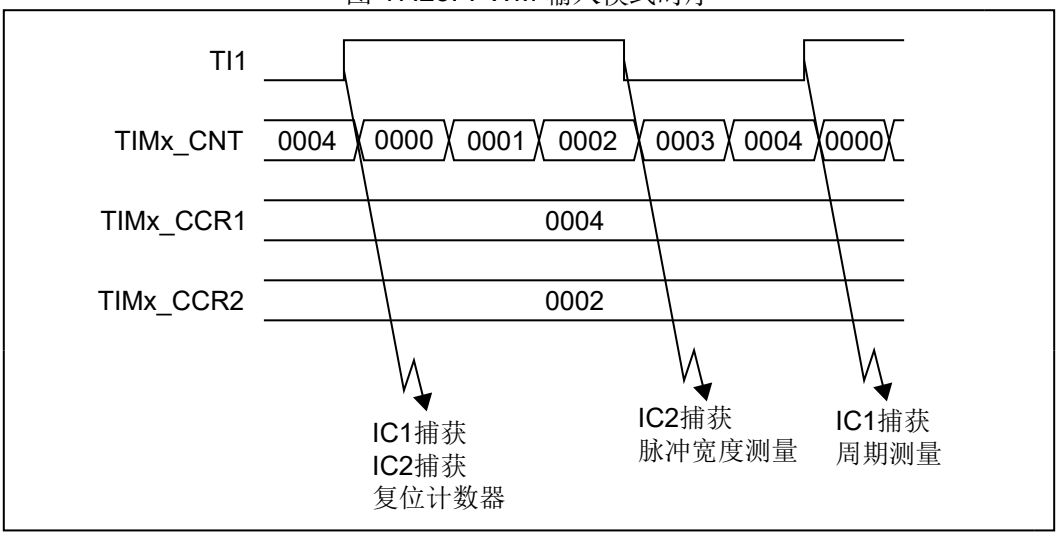
下面给出一个例子具体说明如何配置在该模式下工作。

例如，需要测量输入到 TI1 上的 PWM 信号的长度和占空比，长度存在 TIMx_CCR1 寄存器中，占空比存在 TIMx_CCR2 寄存器中。具体步骤如下：

1. 选择通道 1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
2. 选择 TI1FP1 的有效极性 (用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
3. 选择通道 2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
4. 选择 TI1FP2 的有效极性 (捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
5. 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
6. 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7. 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

下图显示了在这一过程中的时序图。

图 17.25: PWM 输入模式时序



17.3.7　强置输出模式

在输出模式下，输出比较信号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，同时 OCx 得到 CCxP 极性相反的值。

置 TIMx_CCMRx 寄存器中相应的 OCxM=100，即可强置输出比较信号 (OCxREF/OCx) 为无效状态。这样 OCxREF 被强置为低电平，同时 OCx 得到 CCxP 极性相反的值。

在该模式下虽然不使用输出比较器的结果，但是输出比较器仍然工作，相应的标志也会被更新，因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

注：*OCxREF 始终为高电平有效*

17.3.8　输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式 (TIMx_CCMRx 寄存器中的 OCxM 位) 和输出极性 (TIMx_CCER 寄存器中的 CCxP 位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 (OCxM=000)、被设置成有效电平 (OCxM=001)、被设置成无效电平 (OCxM=010) 或进行翻转 (OCxM=011)。
- 设置中断状态寄存器中的标志位 (TIMx_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断屏蔽 (TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的使能位 (TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

使用该模式时，需要注意：

- 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。
- 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：

- 计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
- 置 OCxPE = 0 禁用预装载寄存器
- 置 CCxP = 0 选择极性为高电平有效
- 置 CCxE = 1 使能输出

5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器 (OCxPE=’0’，否则 TIMx_CCRx 影子寄存器只能在发生下一次更新事件时被更新)。

17.3.9　PWM 模式

PWM 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。使用 PWM 模式时，需要注意：

- 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
- 设置 TIMx_CR1 寄存器的 ARPE 位，使能自动重装载的预装载寄存器。
- 仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。
- OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。
- OCx 的输出使能通过 (TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

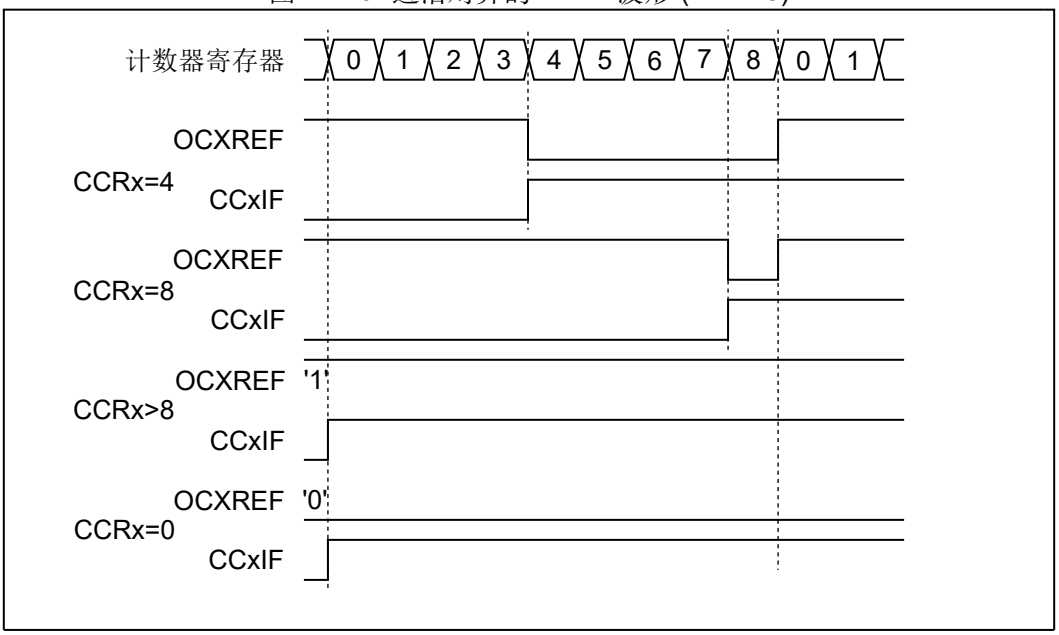
根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式（向上计数）当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

- 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 17.26: 边沿对齐的 PWM 波形 (ARR=8)



PWM 边沿对齐模式（向下计数）

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

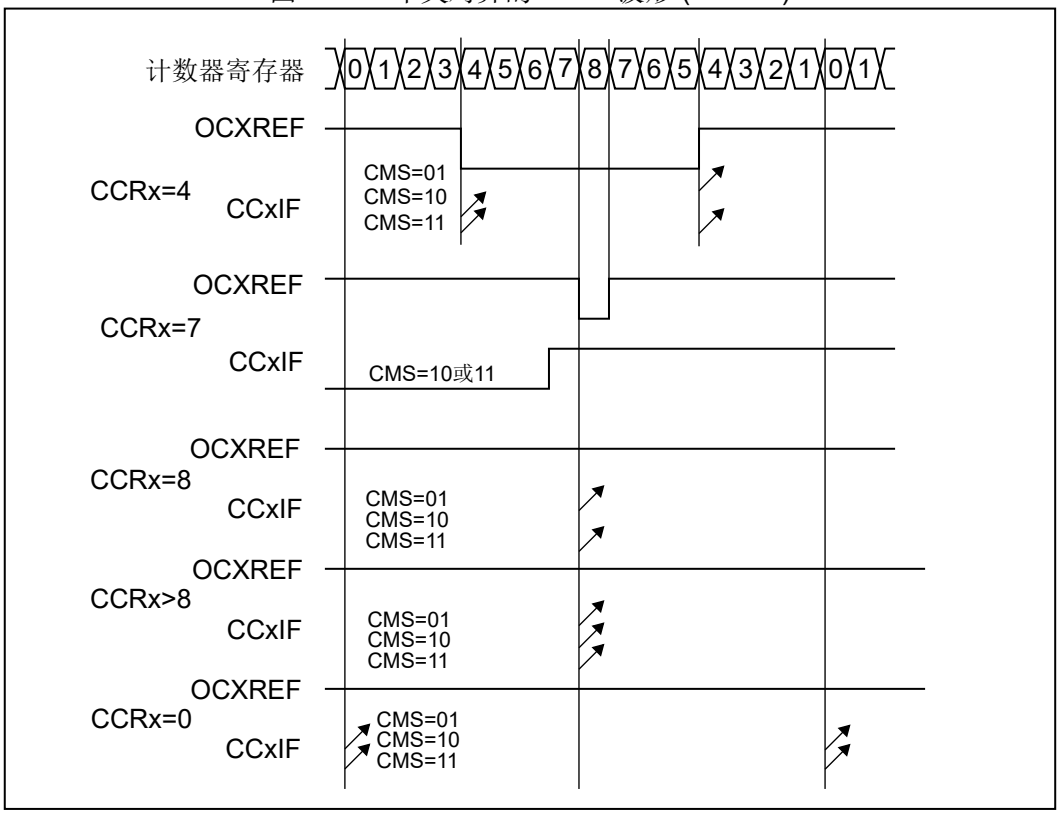
- 在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。
- 如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。
- 该模式下不能产生 0%的 PWM 波形。

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。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。参看17.3.2的中央对齐模式。

下图给出了一些中央对齐的 PWM 波形的例子: (ARR=8, PWM1, TIMx_CR1 寄存器的 CMS=01)

图 17.27: 中央对齐的 PWM 波形 (ARR=8)



使用中央对齐模式的注意：

- 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。
- 使用中央对齐模式保险的方法，就是在启动计数器之前产生一个软件更新，并且不要在计数进行过程中修改计数器的值。

17.3.10　单脉冲模式

单脉冲模式 (OPM) 是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。

设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前 (当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)
- 向下计数方式：计数器 CNT > CCRx

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小延时 tDELAY。

如果要以小延时输出波形，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。

17.3.11　在外部事件时清除 OCxREF 信号

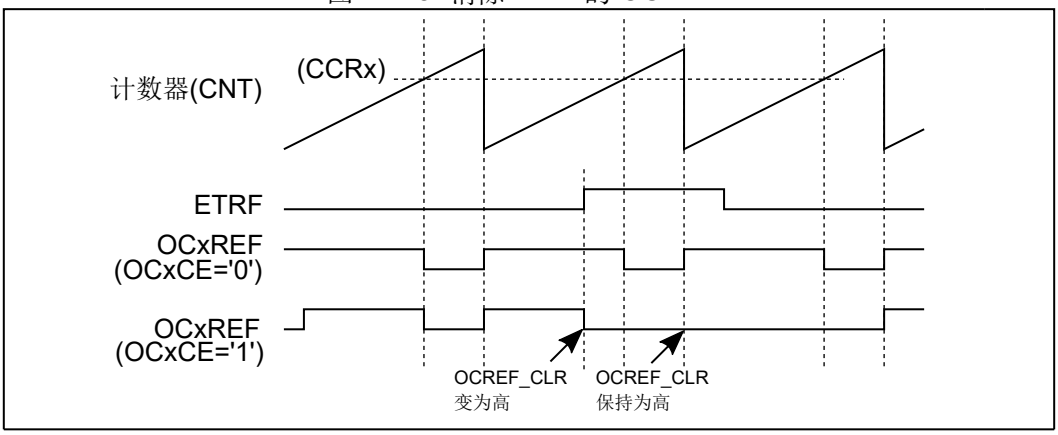
对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为’1’，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

例如，OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

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

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

图 17.28: 清除 TIMx 的 OCxREF



17.3.12　编码器接口模式

选择编码器接口模式的方法是:如果计数器只在 TI2 的边沿计数,则置 TIMx_SMCR 寄存器中的 SMS=001;如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表17.29，假定计数器已经启动，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。

图 17.29: 计数方向与编码器信号的关系

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

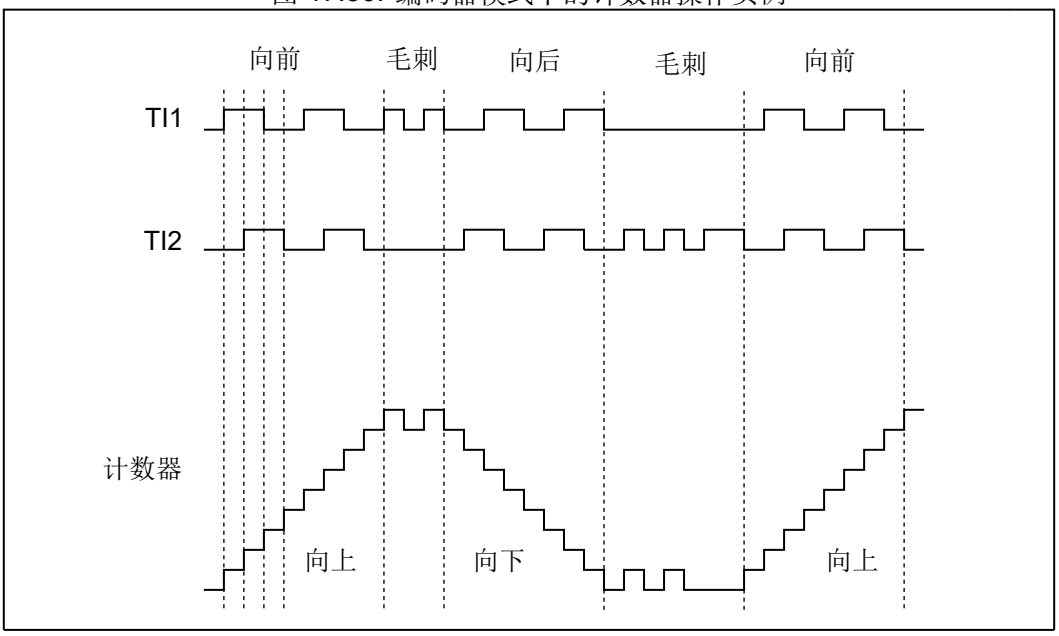
一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

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，IC1FP1 映射到 TI1)
- CC2S=’01’(TIMx_CCMR2 寄存器，IC2FP2 映射到 TI2)
- CC1P=’0’(TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
- CC2P=’0’(TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
- SMS=’011’(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效)

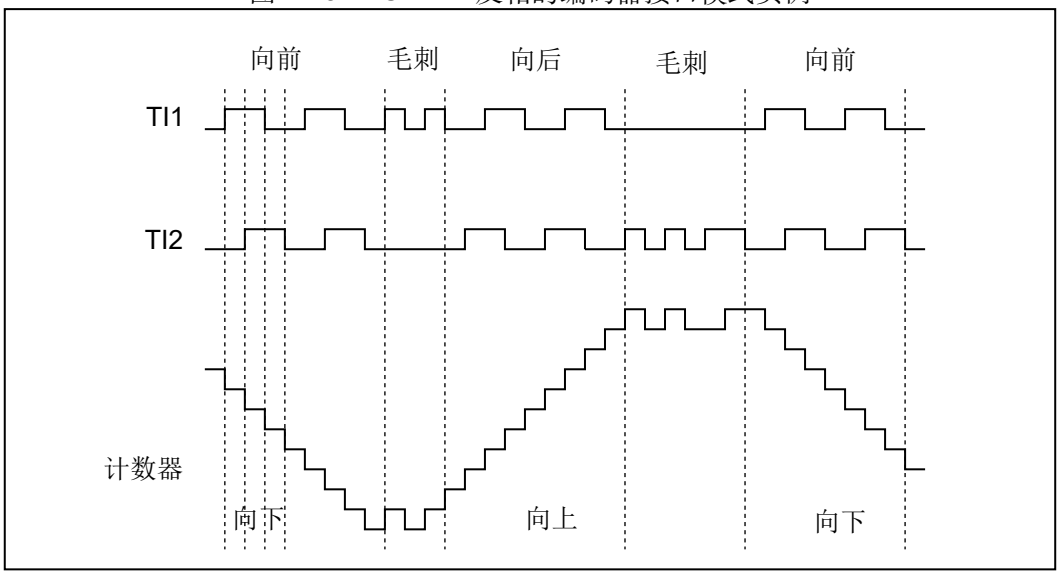
- CEN=’1’(TIMx_CR1 寄存器，计数器使能)

图 17.30: 编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例 (CC1P=’1’，其他配置与上例相同)

图 17.31: IC1FP1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，提供传感器当前位置的信息。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做此目的。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，你可以把计数器的值锁存到第三个输入捕获寄存器 (捕获信号必须是周期的并且可以由另一个定时器产生)；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

17.3.13　定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 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。

17.3.14　定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器 (TIMx_ARR，TIMx_CCRx) 都被更新了。

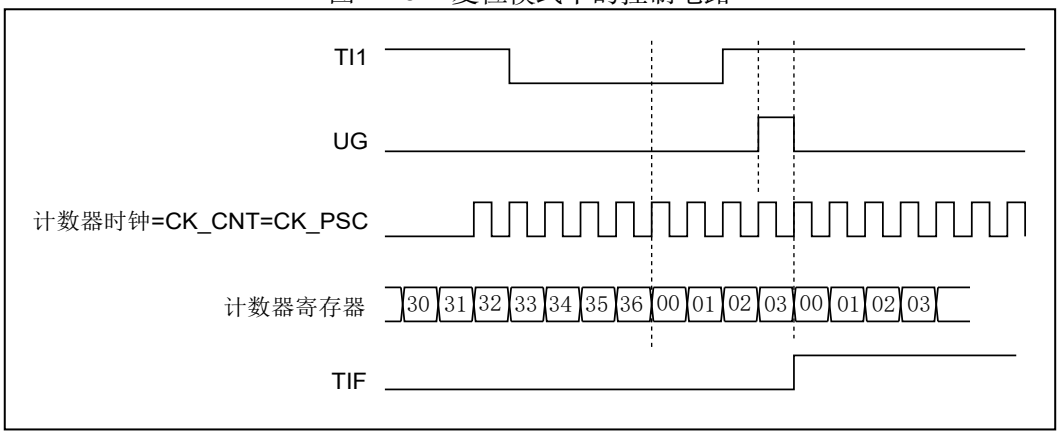
在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽 (在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性 (只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志 (TIMx_SR 寄存器中的 TIF 位) 被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能) 位和 TDE(DMA 使能) 位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图 17.32: 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。

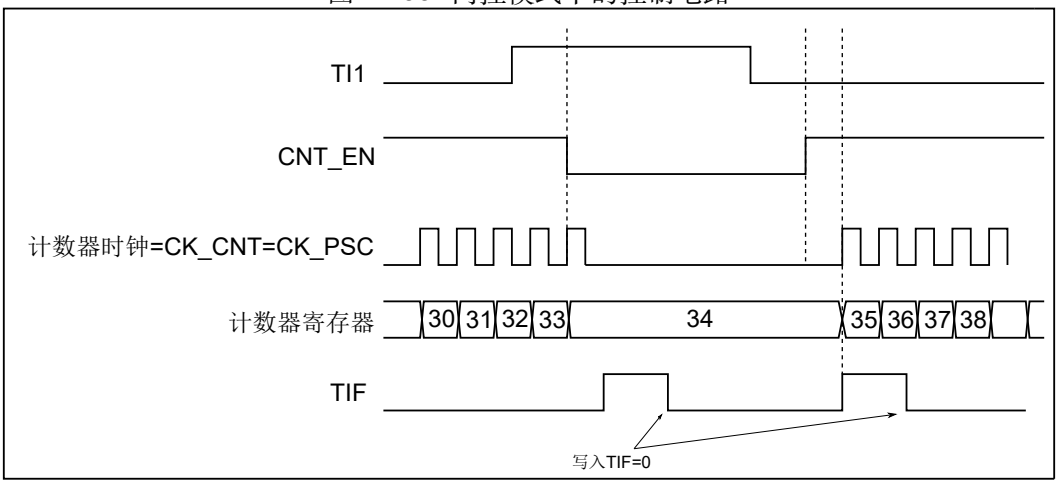
在如下的例子中，计数器只在 TI1 为低时向上计数：

- 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 17.33: 门控模式下的控制电路



textbf 从模式：触发模式

输入端上选中的事件使能计数器。

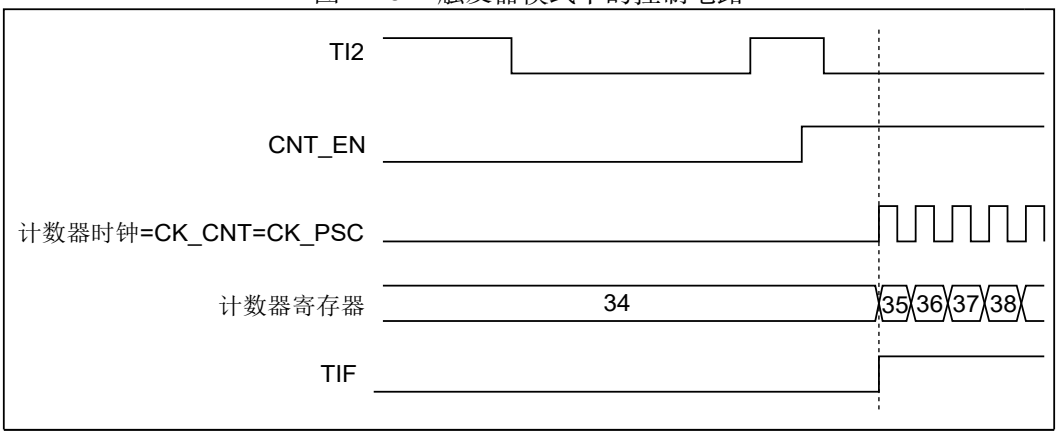
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

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

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图 17.34: 触发器模式下的控制电路



textbf 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：

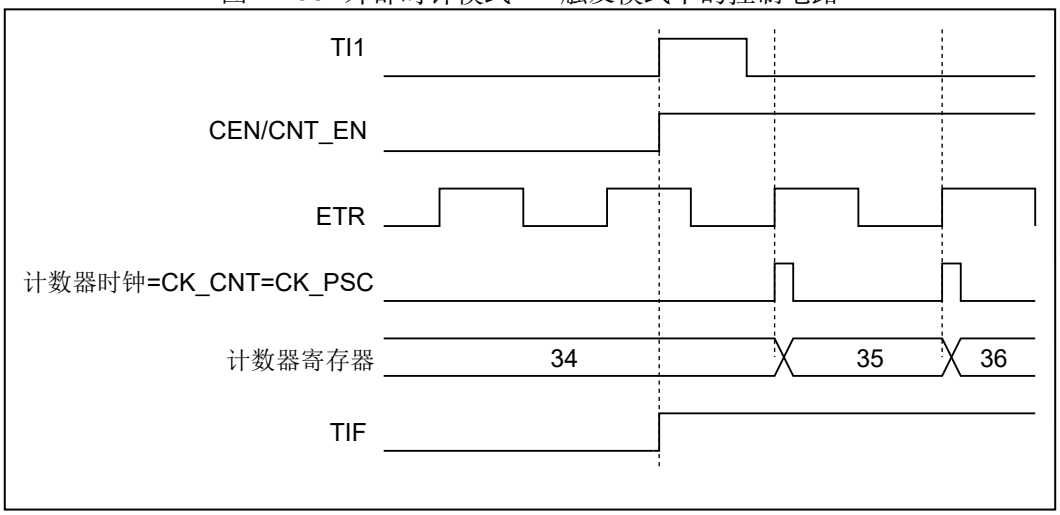
- IC1F=0000：没有滤波 ─ 触发操作中不使用捕获预分频器，不需要配置
- 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
- 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性 (只检测上升沿)

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 17.35: 外部时钟模式 2+ 触发模式下的控制电路



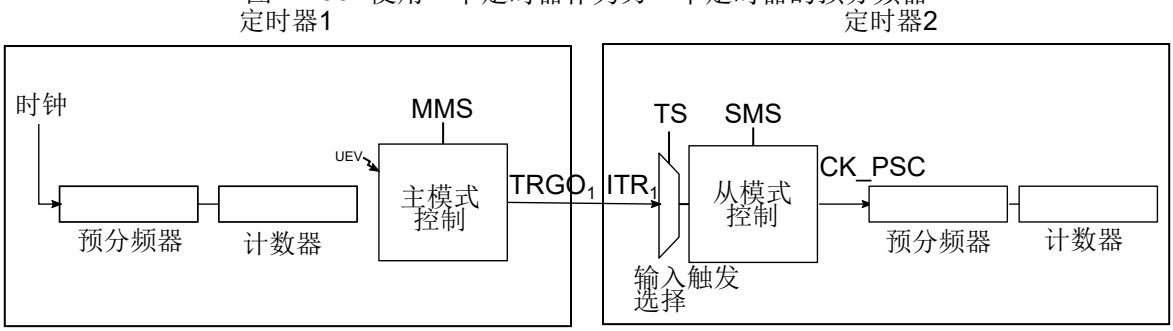
17.3.15　定时器同步

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

下图显示了触发选择和主模式选择模块的概况。

使用一个定时器作为另一个定时器的预分频器

图 17.36: 使用一个定时器作为另一个定时器的预分频器



如：可以配置定时器 1 作为定时器 2 的预分频器。参考图17.36，进行下述操作：

- 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM1_CR2 寄存器的 MMS=’010’时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。
- 连接定时器 1 的 TRGO1 输出至定时器 2，设置 TIM2_SMCR 寄存器的 TS=’000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式 1(TIM2_SMCR 寄存器的 SMS=111)；这样定时器 2 即可由定时器 1 周期性的上升沿 (即定时器 1 的计数器溢出) 信号驱动。
- 最后，必须设置相应 (TIMx_CR1 寄存器) 的 CEN 位分别启动两个定时器。

注：如果 OCx 已被选中为定时器 1 的触发输出 (MMS=1xx)，它的上升沿用于驱动定时器 2 的计数器。

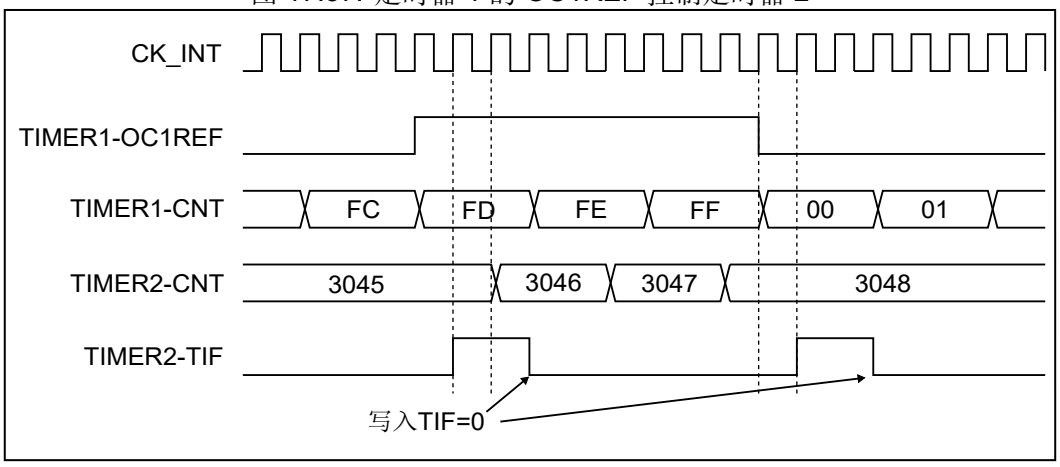
使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考图17.36的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT} = f_{CK\_INT}/3$) 得到。

- 配置定时器 1 为主模式，送出它的输出比较参考信号 (OC1REF) 为触发输出 (TIM1_CR2 寄存器的 MMS=100)
- 配置定时器 1 的 OC1REF 波形 (TIM1_CCMR1 寄存器)
- 配置定时器 2 从定时器 1 获得输入触发 (TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式 (TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

注：定时器 2 的时钟不与定时器 1 的时钟同步，这个模式只影响定时器 2 计数器的使能信号。

图 17.37: 定时器 1 的 OC1REF 控制定时器 2



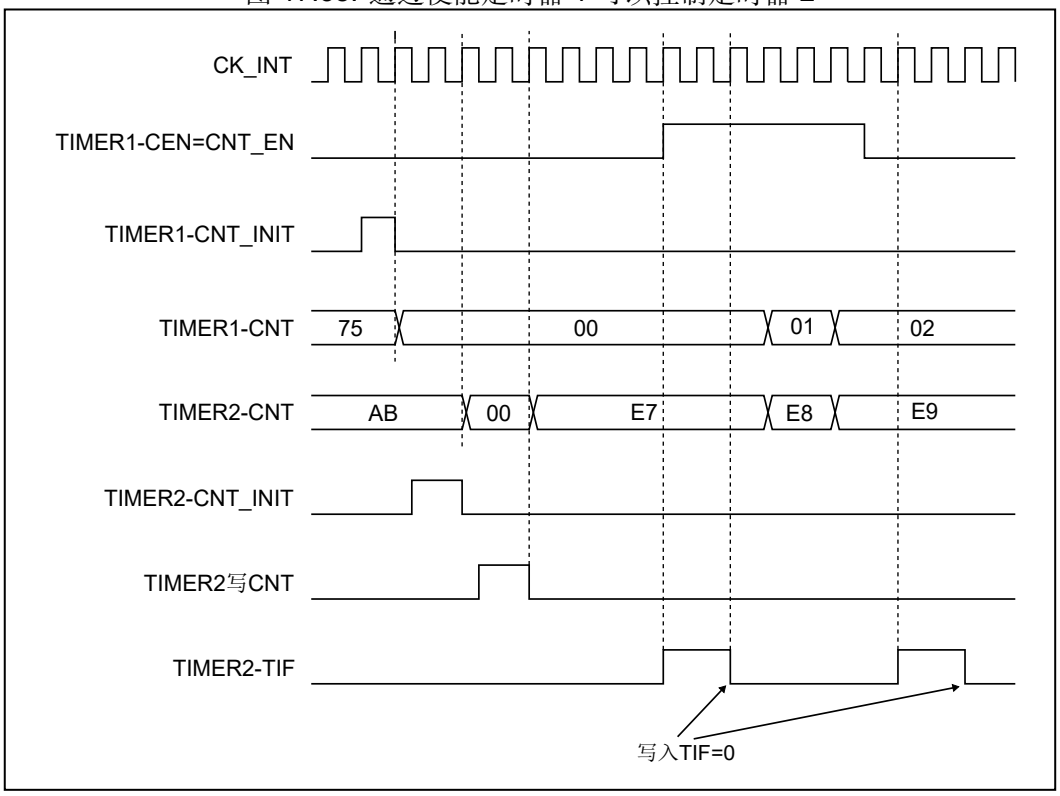
在图17.37的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 1 之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入需要的任意数值。写 TIMx_EGR 寄存器的 UG 位即可复位定时器。

在下一个例子中，需要同步定时器 1 和定时器 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写’0’到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 2 随即停止。

- 配置定时器 1 为主模式，送出输出比较 1 参考信号 (OC1REF) 做为触发输出 (TIM1_CR2 寄存器的 MMS=100)。
- 配置定时器 1 的 OC1REF 波形 (TIM1_CCMR1 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发 (TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式 (TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM1_EGR 寄存器的 UG=’1’，复位定时器 1。

- 置 TIM2_EGR 寄存器的 UG=’1’，复位定时器 2。
- 写’0xE7’至定时器 2 的计数器 (TIM2_CNTL)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN=’1’以使能定时器 2。
- 置 TIM1_CR1 寄存器的 CEN=’1’以启动定时器 1。
- 置 TIM1_CR1 寄存器的 CEN=’0’以停止定时器 1。

图 17.38: 通过使能定时器 1 可以控制定时器 2

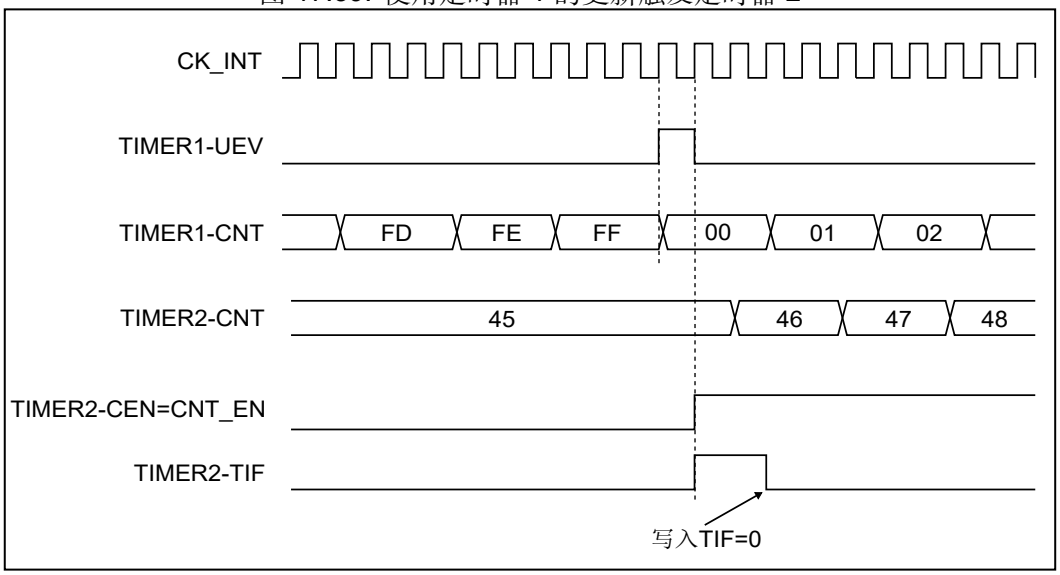


使用一个定时器启动另一个定时器

在这个例子中，使用定时器 1 的更新事件使能定时器 2。参考图17.36的连接。一旦定时器 1 产生更新事件，定时器 2 即从它当前的数值 (可以是非 0) 按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CEN 位被自动地置’1’，同时计数器开始计数直到写’0’到 TIM2_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT} = f_{CK\_INT}/3$)。

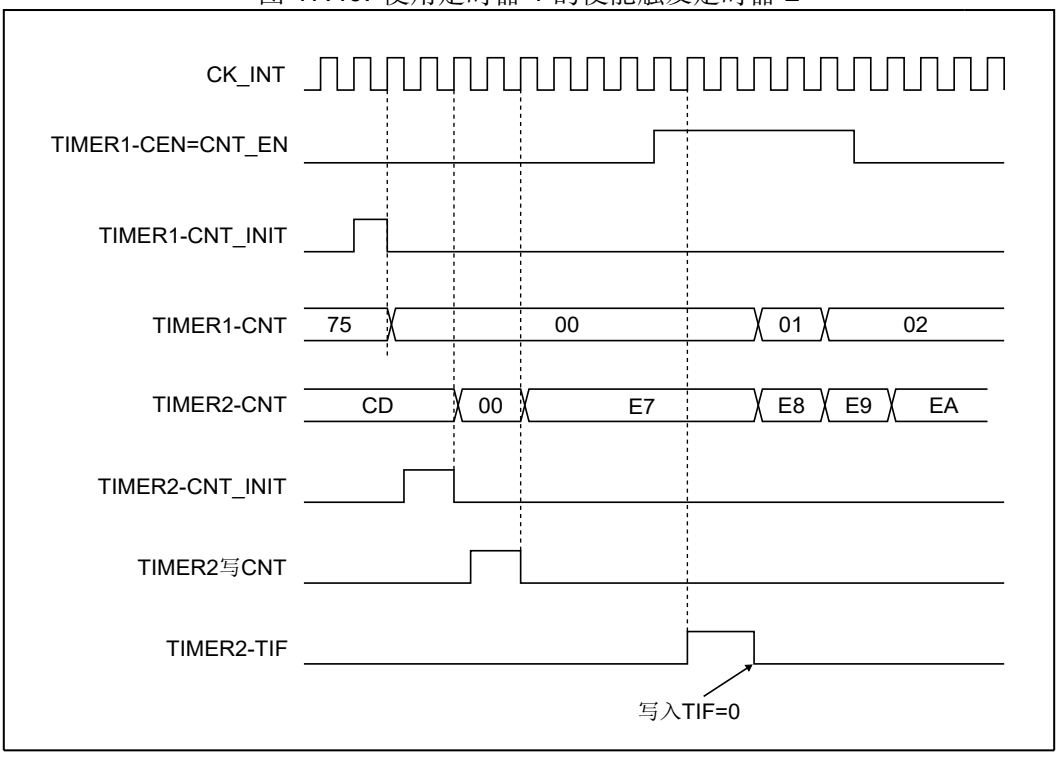
- 配置定时器 1 为主模式，送出它的更新事件 (UEV) 做为触发输出 (TIM1_CR2 寄存器的 MMS=010)。
- 配置定时器 1 的周期 (TIM1_ARR 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发 (TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式 (TIM2_SMCR 寄存器的 SMS=110)
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

图 17.39: 使用定时器 1 的更新触发定时器 2



在上一个例子中，可以在启动计数之前初始化两个计数器。图17.40显示在与 0 相同配置情况下，使用触发模式而不是门控模式 (TIM2_SMCR 寄存器的 SMS=110) 的动作。

图 17.40: 使用定时器 1 的使能触发定时器 2



使用一个定时器作为另一个定时器的预分频器

这个例子使用定时器 1 作为定时器 2 的预分频器。参考图17.36的连接，配置如下：

- 配置定时器 1 为主模式，送出它的更新事件 UEV 做为触发输出 (TIM1_CR2 寄存器的 MMS=’010’)。然后每次计数器溢出时输出一个周期信号。
- 配置定时器 1 的周期 (TIM1_ARR 寄存器)。

- 配置定时器 2 从定时器 1 获得输入触发 (TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 使用外部时钟模式 (TIM2_SMCR 寄存器的 SMS=111)
- 置 TIM1_CR2 寄存器的 CEN=1 以启动定时器 2。
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

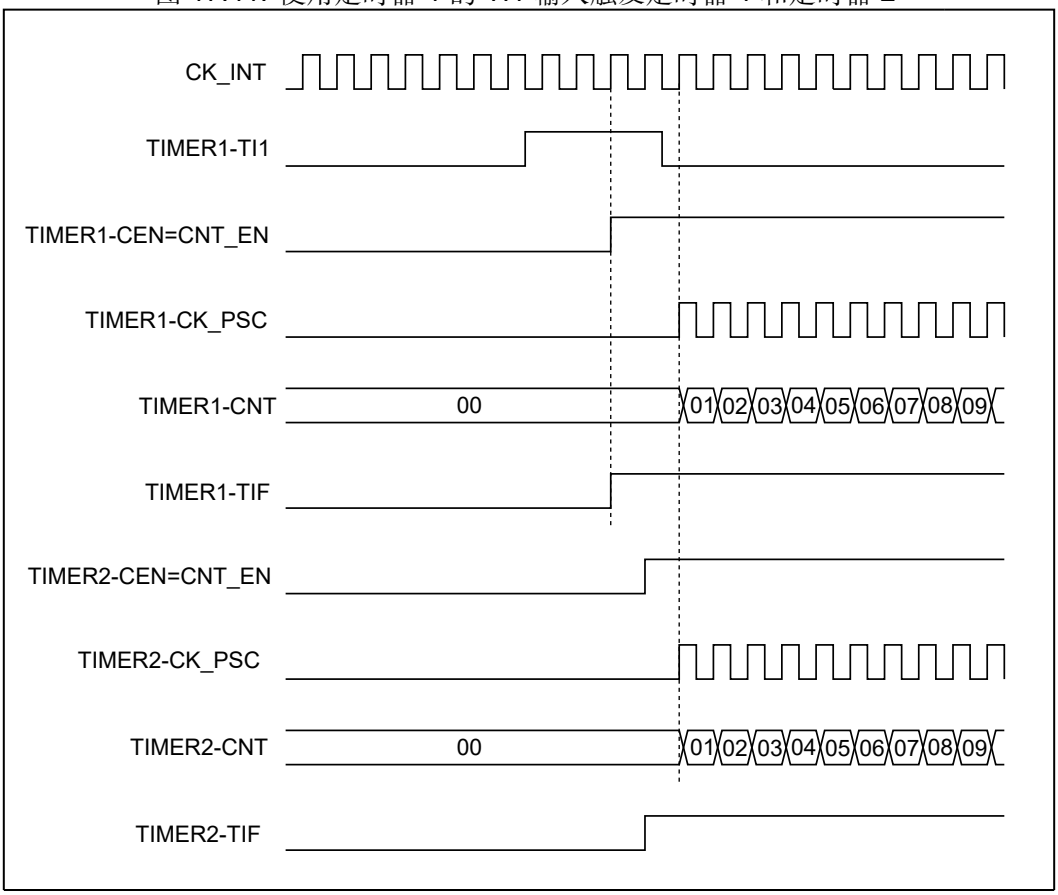
使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见图17.36。为保证计数器的对齐，定时器 1 必须配置为主/从模式 (对应 TI1 为从，对应定时器 2 为主):

- 配置定时器 1 为主模式，送出它的使能做为触发输出 (TIM1_CR2 寄存器的 MMS=’001’)。
- 配置定时器 1 为从模式，从 TI1 获得输入触发 (TIM1_SMCR 寄存器的 TS=’100’)。
- 配置定时器 1 为触发模式 (TIM1_SMCR 寄存器的 SMS=’110’)。
- 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM=’1’。
- 配置定时器 2 从定时器 1 获得输入触发 (TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式 (TIM2_SMCR 寄存器的 SMS=’110’)。当定时器 1 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被设置。

注：在这个例子中，在启动之前两个定时器都被初始化 (置相应的 UG 位)，两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器 (TIMx_CNT) 在定时器间插入一个偏移。下图中能看到主/从模式下在定时器 1 的 CNT_EN 和 CK_PSC 之间有个延迟。

图 17.41: 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2



17.3.16 调试模式

当微控制器进入调试模式时 (Cortex-M3 核心停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。详见31.13.2节。

17.4 通用定时器 TIMx 寄存器描述

17.4.1 TIMx 控制寄存器 1 (TIMx_CR1)

地址偏移：0x00

复位值：0x0000 0000

位	符号	访问	描述
31:10	-	R	保留。始终读为 0。
9:8	CKD[1:0]	RW	时钟分频因子 (Clock division) 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：$t_{DTS} = 2xt_{CK_INT}$ 10：$t_{DTS} = 4xt_{CK_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	自动重装载预装载允许位 (Auto-reload preload enable) 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
6:5	CMS[1:0]	RW	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。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道 (TIMx_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时 (CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	方向 (Direction) 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	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件 (清除 CEN 位) 时，计数器停止。

2	URS	RW	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： • 计数器溢出/下溢 • 设置 UG 位 • 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新 (UEV) 事件由下述任一事件产生： • 计数器溢出/下溢 • 设置 UG 位 • 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持它们的值。 如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	使能计数器 (Counter enable) 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。 在单脉冲模式下，当发生更新事件时，CEN 被自动清除。

17.4.2　TIMx 控制寄存器 2 (TIMx_CR2)

地址偏移：0x04

复位值：0x0000 0000

位	符号	访问	描述
31:8	-	R	保留。始终读为 0。
7	TI1S	RW	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。

6:4	MMS[2:0]	RW	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下： 000：复位–TIMx_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。 001：使能–计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。 010：更新–更新事件被选为触发输入 (TRGO)。 011：比较脉冲–在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时 (即使它已经为高)，触发输出送出一个正脉冲 (TRGO)。 100：比较–OC1REF 信号被用于作为触发输出 (TRGO)。 101：比较–OC2REF 信号被用于作为触发输出 (TRGO)。 110：比较–OC3REF 信号被用于作为触发输出 (TRGO)。 111：比较–OC4REF 信号被用于作为触发输出 (TRGO)。
3	CCDS	RW	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2:0	-	R	保留。始终读为 0。

17.4.3　TIMx 从模式控制寄存器 (TIMx_SMCR)

地址偏移：0x08

复位值：0x0000 0000

位	符号	访问	描述
31-16	-	R	保留
15	ETP	RW	外部触发极性 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
14	ECE	RW	外部时钟使能位 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111) 具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是’111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。

13:12	ETPS[1:0]	RW	外部触发预分频 外部触发信号 ETRP 的频率不能超过 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
11:8	ETF[3:0]	RW	外部触发滤波 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。 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 $f_{SAMPLING} = fCK_INT$，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
7	MSM	RW	主/从模式 0：无作用； 1：触发输入 (TRGI) 上的事件被延迟了，以实现当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	RW	触发选择 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0(ITR0) 100：TI1 的边沿检测器 (TI1F_ED) 001：内部触发 1(ITR1) 101：滤波后的定时器输入 1(TI1FP1) 010：内部触发 2(ITR2) 110：滤波后的定时器输入 2(TI2FP2) 011：内部触发 3(ITR3) 111：外部触发输入 (ETRF) 更多有关 ITRx 的细节，参见表16.4。注：这些位只能在未用到 (如 SMS=000) 时被改变，以避免在改变时产生错误的边沿检测。
3	-	R	保留，始终读为 0。

2:0	SMS[2:0]	RW	从模式选择 000：关闭从模式–如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 –根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 –根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 –根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式–选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式–当触发输入 (TRGI) 为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止 (但不复位)。计数器的启动和停止都是受控的。 110：触发模式–计数器在触发输入 TRGI 的上升沿启动 (但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 –选中的触发输入 (TRGI) 的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入 (TS=100) 时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

17.4.4 TIMx DMA/中断使能寄存器 (TIMx_DIER)

地址偏移：0x0C

复位值：0x0000 0000

位	符号	访问	描述
31:15	-	R	保留，始终读为 0。
14	TDE	RW	允许触发 DMA 请求 0：禁止触发 DMA 请求； 1：允许触发 DMA 请求。
13	-	R	保留，始终读为 0。
12	CC4DE	RW	允许捕获/比较 4 的 DMA 请求 0：禁止捕获/比较 4 的 DMA 请求； 1：允许捕获/比较 4 的 DMA 请求。
11	CC3DE	RW	允许捕获/比较 3 的 DMA 请求 0：禁止捕获/比较 3 的 DMA 请求； 1：允许捕获/比较 3 的 DMA 请求。
10	CC2DE	RW	允许捕获/比较 2 的 DMA 请求 0：禁止捕获/比较 2 的 DMA 请求； 1：允许捕获/比较 2 的 DMA 请求。
9	CC1DE	RW	允许捕获/比较 1 的 DMA 请求 0：禁止捕获/比较 1 的 DMA 请求； 1：允许捕获/比较 1 的 DMA 请求。
8	UDE	RW	允许更新的 DMA 请求 0：禁止更新的 DMA 请求； 1：允许更新的 DMA 请求。

7	-	R	保留，始终读为 0。
6	TIE	RW	触发中断使能 0：禁止触发中断； 1：使能触发中断。
5	-	R	保留，始终读为 0。
4	CC4IE	RW	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	允许捕获/比较 3 中断 0：禁止捕获/比较 3 中断； 1：允许捕获/比较 3 中断。
2	CC2IE	RW	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	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	允许更新中断 0：禁止更新中断； 1：允许更新中断。

17.4.5 TIMx 状态寄存器 (TIMx_SR)

地址偏移：0x10

复位值：0x0000 0000

位	符号	访问	描述
31:13	-	R	保留，始终读为 0。
12	CC4OF	RC_W0	捕获/比较 4 重复捕获标记 参见 CC1OF 描述。
11	CC3OF	RC_W0	捕获/比较 3 重复捕获标记 参见 CC1OF 描述。
10	CC2OF	RC_W0	捕获/比较 2 重复捕获标记 参见 CC1OF 描述。
9	CC1OF	RC_W0	捕获/比较 1 重复捕获标记 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：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
8:7	-	R	保留，始终读为 0。

6	TIF	RC_W0	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，或门控模式下的任一边沿) 时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	-	R	保留，始终读为 0。
4	CC4IF	RC_W0	捕获/比较 4 中断标记 参考 CC1IF 描述。
3	CC3IF	RC_W0	捕获/比较 3 中断标记 参考 CC1IF 描述。
2	CC2IF	RC_W0	捕获/比较 2 中断标记 参考 CC1IF 描述。
1	CC1IF	RC_W0	捕获/比较 1 中断标记 **如果通道 CC1 配置为输出模式：** 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外 (参考 TIMx_CR1 寄存器的 CMS 位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 **如果通道 CC1 配置为输入模式：** 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	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：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： • 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时 (重复计数器 =0 时产生更新事件)。 • 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 • 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

17.4.6　TIMx 事件产生寄存器 (TIMx_EGR)

地址偏移：0x14

复位值：0x0000 0000

位	符号	访问	描述
31:7	-	R	保留，始终读为 0。
6	TG	W	产生触发事件 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	-	R	保留，始终读为 0。
4	CC4G	W	产生捕获/比较 4 事件 参考 CC1G 描述。
3	CC3G	W	产生捕获/比较 3 事件 参考 CC1G 描述。
2	CC2G	W	产生捕获/比较 2 事件 参考 CC1G 描述。
1	CC1G	W	产生捕获/比较 1 事件 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出：设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入：当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	产生更新事件 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意预分频器的计数器也被清’0’，但是预分频系数不变。 若在中心对称模式下或 DIR=0(向上计数) 则计数器被清’0’； 若 DIR=1(向下计数) 则计数器取 TIMx_ARR 的值。

17.4.7　TIMx 捕获/比较模式寄存器 1 (TIMx_CCMR1)

地址偏移：0x18

复位值：0x0000 0000

通道可用于输入 (捕获模式) 或输出 (比较模式)，通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15	OC2CE	RW	输出比较 2 清 0 使能

14:12	OC2M[2:0]	RW	输出比较 2 模式
11	OC2PE	RW	输出比较 2 预装载使能
10	OC2FE	RW	输出比较 2 快速使能
9:8	CC2S[1:0]	RW	捕获/比较 2 选择。 该位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC2E=0) 才是可写的。
7	OC1CE	RW	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	输出比较 1 模式 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1) 相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1) 相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平 (OC1REF=0)，否则为有效电平 (OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位) 并且 CC1S=00(该通道配置成输出) 则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。

3	OC1PE	RW	输出比较 1 预装载使能 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位) 并且 CC1S=00(该通道配置成输出) 则该位不能被修改。 注 2：仅在单脉冲模式下 (TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	输出比较 1 快速使能 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
1:0	CC1S[1:0]	RW	捕获/比较 1 选择。 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。

输入捕获模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15:12	IC2F[3:0]	RW	输入捕获 2 滤波器
11:10	IC2PSC[1:0]	RW	输入/捕获 2 预分频器
9:8	CC2S[1:0]	RW	捕获/比较 2 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC2E=0) 才是可写的。

7:4	IC1F[3:0]	RW	输入捕获 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 1000：采样频率 $f_{SAMPLING} = f_{DTS}/8$，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	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，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	捕获/比较 1 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。

17.4.8 TIMx 捕获/比较模式寄存器 2 (TIMx_CCMR2)

地址偏移：0x1C

复位值：0x0000 0000

参看以上 CCMR1 寄存器的描述

输出比较模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15	OC4CE	RW	输出比较 4 清 0 使能

14:12	OC4M[2:0]	RW	输出比较 4 模式
11	OC4PE	RW	输出比较 4 预装载使能
10	OC4FE	RW	输出比较 4 快速使能
9:8	CC4S[1:0]	RW	捕获/比较 4 选择 该 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC4E=0) 才是可写的。
7	OC3CE	RW	输出比较 3 清 0 使能
6:4	OC3M[2:0]	RW	输出比较 3 模式
位 3	OC3PE	RW	输出比较 3 预装载使能
位 2	OC3FE	RW	输出比较 3 快速使能
1:0	CC3S[1:0]	RW	捕获/比较 3 选择 该 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC3E=0) 才是可写的。

输入捕获模式：

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15:12	IC4F[3:0]	RW	输入捕获 4 滤波器
11:10	IC4PSC[1:0]	RW	输入/捕获 2 预分频器
9:8	CC4S[1:0]	RW	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：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI24 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC4E=0) 才是可写的。
7:4	IC3F[3:0]	RW	输入捕获 3 滤波器
3:2	IC3PSC[1:0]	RW	输入/捕获 3 预分频器

1:0	CC3S[1:0]	RW	捕获/比较 3 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC3E=0) 才是可写的。

17.4.9　TIMx 捕获/比较使能寄存器 (TIMx_CCER)

地址偏移：0x20

复位值：0x0000 0000

位	符号	访问	描述
31:14	-	R	保留，始终读为 0。
13	CC4P	RW	输入/捕获 4 输出极性 参考 CC1P 的描述。
12	CC4E	RW	输入/捕获 4 输出使能 参考 CC1E 的描述。
11:10	-	R	保留，始终读为 0。
9	CC3P	RW	输入/捕获 3 输出极性 参考 CC1P 的描述。
8	CC3E	RW	输入/捕获 3 输出使能 参考 CC1E 的描述。
7:6	-	RW	保留，始终读为 0。
5	CC2P	RW	输入/捕获 2 输出极性 参考 CC1P 的描述。
4	CC2E	RW	输入/捕获 2 输出使能 参考 CC1E 的描述。
3:2	-	R	保留，始终读为 0。
1	CC1P	RW	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：** 该位选择是 IC1 还是 IC1 的反相信号作为触发或捕获信号。 0：不反相：捕获发生在 IC1 的上升沿；当用作外部触发器时，IC1 不反相。 1：反相：捕获发生在 IC1 的下降沿；当用作外部触发器时，IC1 反相。

0	CC1E	RW	输入/捕获 1 输出使能 **CC1 通道配置为输出：** 0：关闭 - OC1 禁止输出 1：开启 - OC1 信号输出到对应的输出引脚 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表 17.12: 标准 OCx 通道的输出控制位

CCxE 位	OCx 输出状态
0	禁止输出 (OCx=0，OCx_EN=0)
1	OCx = OCxREF + 极性，OCx_EN=1

17.4.10　TIMx 计数器 (TIMx_CNT)

地址偏移：0x24

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	RW	保留，始终读为 0。
19:0	CNT[19:0]	RW	计数器的值

17.4.11　TIMx 预分频器 (TIMx_PSC)

地址偏移：0x28

复位值：0x0000 0000

位	符号	访问	描述
31:16	-	R	保留，始终读为 0。
15:0	PSC[15:0]	RW	预分频器的值 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_PSC}/(PSC[15:0]+1)$。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’。

17.4.12　TIMx 自动重装载寄存器 (TIMx_ARR)

地址偏移：0x2C

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。

19:0	ARR[19:0]	RW	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16.3.3节：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。

17.4.13 TIMx 捕获/比较寄存器 1 (TIMx_CCR1)

地址偏移：0x34

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	CCR1[19:0]	RW	**CCR1[19:0]:** 捕获/比较通道 1 的值 **若 CC1 通道配置为输出：** CCR1 包含了装入当前捕获/比较 1 寄存器的值 (预装载值)。如果在 TIMx_CCMR1 寄存器 (OC1PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。

17.4.14 TIMx 捕获/比较寄存器 2 (TIMx_CCR2)

地址偏移：0x38

复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	CCR2[19:0]	RW	捕获/比较通道 2 的值 **若 CC2 通道配置为输出：** CCR2 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR2 寄存器 (OC2PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 **若 CC2 通道配置为输入：** CCR2 包含了由上一次输入捕获 2 事件 (IC2) 传输的计数器值。

17.4.15 TIMx 捕获/比较寄存器 3 (TIMx_CCR3)

地址偏移：0x3C
复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	CCR3[19:0]	RW	捕获/比较通道 3 的值 **若 C3 通道配置为输出：** CCR3 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR3 寄存器 (OC3PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 **若 CC3 通道配置为输入：** CCR3 包含了由上一次输入捕获 3 事件 (IC3) 传输的计数器值。

17.4.16 TIMx 捕获/比较寄存器 4 (TIMx_CCR4)

地址偏移：0x40
复位值：0x0000 0000

位	符号	访问	描述
31:20	-	R	保留，始终读为 0。
19:0	CCR4[19:0]	RW	捕获/比较通道 4 的值 **若 C4 通道配置为输出：** CCR4 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TIMx_CCMR4 寄存器 (OC4PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 **若 CC4 通道配置为输入：** CCR4 包含了由上一次输入捕获 4 事件 (IC4) 传输的计数器值。

第十八章　实时时钟 (RTC)

18.1　简介

实时时钟是一个独立的定时器。RTC 模块拥有一组连续计数的计数器，在相应软件配置下，可提供时钟日历的功能。修改计数器的值可以重新设置系统当前的时间和日期。

RTC 模块和时钟配置系统 (BKP_BDCR 寄存器) 位于备份域，即在系统复位或从待机模式唤醒后，RTC 的设置和时间维持不变。

系统复位后，对备份域寄存器和 RTC 的访问被禁止，这是为了防止对备份域 (BKP) 的意外写操作。执行以下操作将开启对备份域寄存器和 RTC 的访问：

- 设置 RCC 寄存器 AHBENR2 的 BDI 位，将备份域接口时钟使能
- PWR 电源控制寄存器 CR0 的 DBP 位来使能对后备寄存器和 RTC 的访问。

18.2　特性

- 可编程的预分频系数
- 2 个独立的输入时钟：AHB 总线时钟和 RTC 时钟 (RTC 时钟频率必须至少小于 AHB 总线时钟频率的四分之一)
- 3 个 RTC 时钟源：HSE/128, LSE 和 LSI。
- 2 个独立的复位：AHB 接口复位 (即系统复位)，RTC 核心复位 (即备份域复位)
- 3 个独立的可屏蔽中断：闹钟中断，秒中断和溢出中断

18.3　功能描述

18.3.1　概述

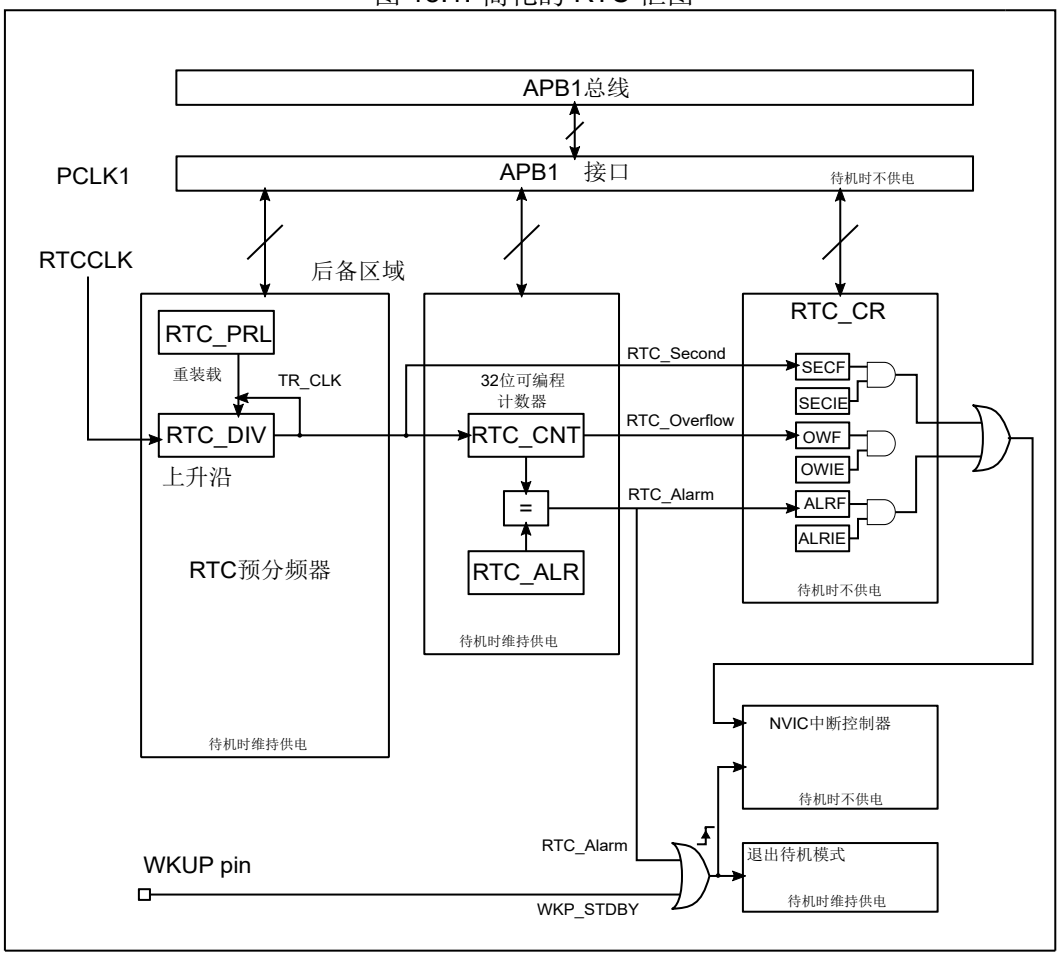
RTC 由两个主要部分组成。

第一部分为 APB 接口，用来与 AHB 总线相连，此部分包含一组寄存器，可通过 AHB 总线对其进行读写。

另一部分为 RTC 核心，由一组可编程计数器组成，分为 2 个主要模块。第一个模块是 RTC 预分频模块，它可编程产生最长为 1 秒的 RTC 时间基准 TR_CLK。RTC 的预分频模块包含了一个 20 位的可编程分频器，

如果在 RTC_CR 寄存器中设置了相应的使能，则在每个基准时间 TR_CLK 周期中产生一个秒中断。第二个模块是一个 32 位可编程计数器，可被初始化成当前的系统时间，系统时间按基准时间 TR_CLK 周期累加并与存储在 RTC_ALR 寄存器中的可编程时间相比较，如果 RTC_CR 控制寄存器中设置了相应的使能位，比较匹配时将产生一个闹钟中断。

图 18.1: 简化的 RTC 框图



18.3.2　复位过程

除了 RTC_PRL，RTC_ALR，RTC_CNT 和 RTC_DIV 寄存器外，所有的系统寄存器都由系统复位或电源复位来进行复位。

RTC_PRL，RTC_ALR，RTC_CNT 和 RTC_DIV 寄存器只能通过备份域复位来进行复位。

18.3.3　RTC 寄存器读取

RTC 核完全独立于 RTC APB 接口。

软件通过 APB 接口访问 RTC 的预分频值、计数器值和闹钟值。但是，相关的可读寄存器只在与 RTC APB 时钟进行重新同步的 RTC 时钟的上升沿被更新。RTC 标志也是如此的。

这意味着，如果 APB 接口曾经被关闭，而读操作又是在刚刚重新开启 APB 之后，则在第一次的内部寄存器更新之前，从 APB 上读出的 RTC 寄存器数值可能被破坏了 (通常读到 0)。下述几种情况下能够发生这种情形：

- 发生系统复位或电源复位
- 系统刚从待机模式唤醒
- 系统刚从停机模式唤醒

所有以上情况中，APB 接口被关闭时 (复位、无时钟或断电)RTC 核仍保持运行状态。

若 RTC 的 APB 接口时钟曾经被关闭过，则软件必须等到 RTC_CRL 寄存器中的 RSF 位被硬件置为 1 之后才能读取其余 RTC 寄存器。

注：*RTC 的 APB 接口不受 WFI 和 WFE 等低功耗模式的影响。*

18.3.4　RTC 寄存器配置

必须设置 RTC_CRL 寄存器中的 CNF 位，使 RTC 进入配置模式后，才能对 RTC_PRL，RTC_CNT 和 RTC_ALR 寄存器进行写入。另外必须注意：对 RTC 任何寄存器的写操作，都必须在前一次写操作结束后进行，可以查询 RTC_CRL 寄存器中的 RTOFF 位来判断 RTC 寄存器是否处于更新中，只有当 RTOFF 位是 1 时才能对 RTC 寄存器写入。

配置过程：

1. 查询 RTOFF，直到 RTOFF 的值为 1
2. 将 CNF 置为 1，进入配置模式
3. 对 RTC 寄存器进行写操作
4. 清除 CNF 位，退出配置模式
5. 查询 RTOFF，直到 RTOFF 的值为 1，确认写操作完成

仅当 CNF 标志位被清除时，写操作才能进行，这个过程至少需要 3 个 RTCCLK 周期。

18.3.5　RTC 标志的设置

在每一个 RTC 核心的时钟周期中，硬件会更改 RTC 计数器之前设置 RTC 秒标志 SECF。

在计数器到达 0x0000 之前的最后一个 RTC 时钟周期中，硬件会设置 RTC 溢出标志 OWF。

在计数器的值到达闹钟寄存器的值加 1(RTC_ALR+1) 之前一个时钟周期，硬件会设置 RTC 闹钟标志 ALRF。

对 RTC 闹钟的写操作必须使用下述过程之一来与 RTC 秒标志同步：

- 使用 RTC 闹钟中断，并在中断处理程序中修改 RTC 闹钟和 RTC 计数器
- 等 RTC 控制寄存器中的 SECF 位被设置，再更改 RTC 闹钟和 RTC 计数器

图 18.2: RTC 秒和闹钟波形图示例，PR=0003，ALARM=00004

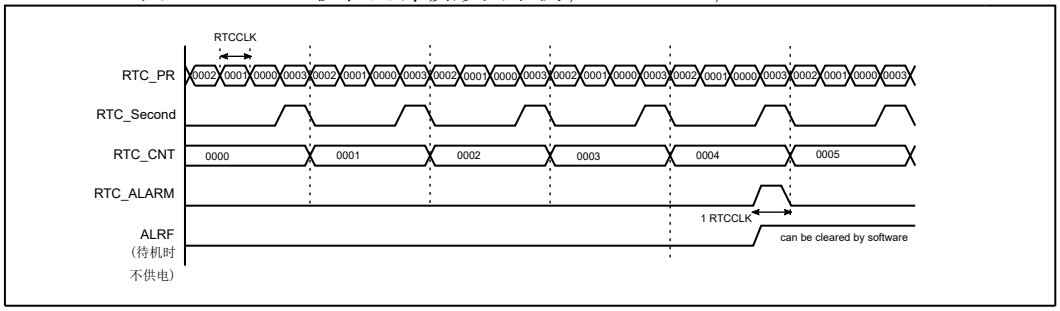
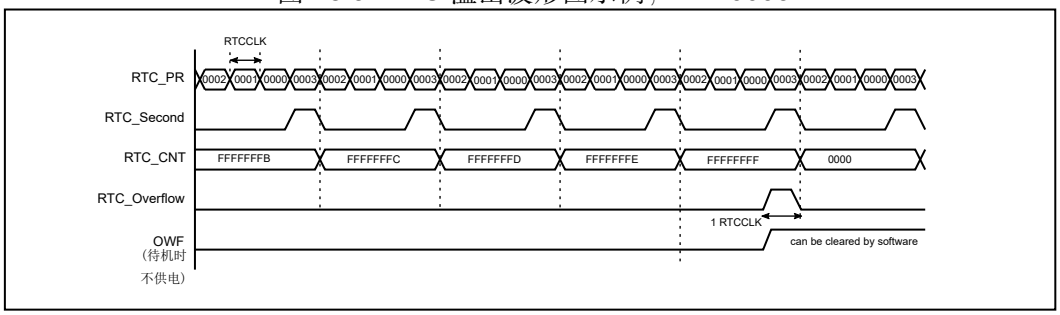


图 18.3: RTC 溢出波形图示例，PR=0003



18.4　RTC 寄存器

18.4.1　控制寄存器高位 (RTC_CRH)

地址偏移：0x00

复位值：0x0000_0000

位	符号	访问	描述
31:3	-	R	保留
2	OWIE	RW	溢出中断屏蔽 0: 屏蔽溢出中断 1: 允许溢出中断
1	ALRIE	RW	闹钟中断屏蔽 0: 屏蔽闹钟中断 1: 允许闹钟中断
0	SECIE	RW	秒中断屏蔽 0: 屏蔽秒中断 1: 允许秒中断

18.4.2　控制寄存器低位 (RTC_CRL)

地址偏移：0x04

复位值：0x0000_0020

位	符号	访问	描述
31:6	-	R	保留
5	RTOFF	R	RTC 操作关闭，若此位为 0 则无法对任何 RTC 寄存器进行写操作 0: 上一次 RTC 寄存器写操作仍在进行 1: 上一次 RTC 寄存器写操作已完成
4	CNF	RW	配置标志，此位必须由软件置 1 以进入配置模式，然后才能对 RTC 核心寄存器写入，然后软件将此位清零后，RTC 核心寄存器才会真正开始更新 0: 退出配置模式，RTC 核心寄存器开始更新为写入的值 1: 进入配置模式，软件可以对 RTC 核心寄存器写入

3	RSF	R_W0	寄存器同步标志，每当 RTC_CNT 和 RTC_DIV 寄存器由软件更新时，此位由硬件置 1。要进行任何的读操作之前，用户程序必须等待此位被硬件置 1，以确保 RTC 核心寄存器已同步 0: RTC 核心寄存器尚未同步 1: RTC 核心寄存器已经同步
2	OWF	R_W0	溢出标志，每当 RTC_CNT 溢出时，此位由硬件置 1。如果 RTC_CRH 中的 OWIE=1，则产生中断。此位只能由软件清零，对此位写 1 是无效的 0: 无溢出 1: RTC_CNT 溢出
1	ALRF	R_W0	闹钟标志，当 RTC_CNT 达到 RTC_ALR 的预设值时，此位由硬件置 1。如果 RTC_CRH 中的 ALRIE=1，则产生中断。此位只能由软件清零，对此位写 1 是无效的 0: 无闹钟 1: 有闹钟
0	SECF	R_W0	秒标志，当预分频器 RTC_PRL 溢出时，此位由硬件置 1 同时 RTC_CNT 计数器加 1。因此，此标志为分辨率可编程的 RTC 计数器提供一个周期性的信号 (通常为 1 秒)。如果 RTC_CRH 中的 SECIE=1，则产生中断。此位只能由软件清零，对此位写 1 是无效的 0: 秒标志条件不成立 1: 秒标志条件成立

注：*1.* 任何标志位都将保持挂起状态，直到适当的 *RTC_CR* 请求位被软件复位，表示所请求的中断已经被接受。

2. 在复位时禁止所有中断，无挂起的中断请求，可以对 *RTC* 寄存器进行写操作。

3. OWF 、*ALRF* 、*SECF* 和 *RSF* 位只能由硬件置位，由软件来清零。

4. 若 *ALRF=1* 且 *ALRIE=1* ，则允许产生 *RTC* 全局中断。如果在 *EXTI* 控制器中允许产生 *EXTI* 线 *17* 中断，则允许产生 *RTC* 全局中断和 *RTC* 闹钟中断。

5. 若 *ALRF=1* ，如果在 *EXTI* 控制器中设置了 *EXTI* 线 *17* 的中断模式，则允许产生 *RTC* 闹钟中断；

如果在 *EXTI* 控制器中设置了 *EXTI* 线 *17* 的事件模式，则这条线上会产生一个脉冲 *(* 不会产生 *RTC* 闹钟中断*)* 。

18.4.3　预分频寄存器高位 (RTC_PRLH)

地址偏移：0x08

复位值：0x0000_0000

位	符号	访问	描述
31:4	-	R	保留
3:0	PRL[19:16]	W	预分频高位，根据以下公式来定义计数器的时钟频率 $f_{TR\_CLK} = f_{RTCCLK}/(PRL[19:0]+1)$ 注：不推荐用 0 值，否则无法准确产生 RTC 中断和标志位

18.4.4　预分频寄存器低位 (RTC_PRLL)

地址偏移：0x0C

复位值：0x0000_8000

位	符号	访问	描述
31:16	-	R	保留
15:0	PRL[15:0]	W	预分频低位，根据以下公式来定义计数器的时钟频率 $f_{TR\_CLK} = f_{RTCCLK}/(PRL[19:0]+1)$ 注：如果输入时钟 f_{RTCCLK}=32.768KHz，这个寄存器写入 7FFF 可获得周期为 1 秒的信号

18.4.5　预分频余数寄存器高位 (RTC_DIVH)

地址偏移：0x10

复位值：0x0000_0000

描述：在 TR_CLK 的每个周期里，RTC 预分频器中计数器的值都会被重新设置为 RTC_PRL 寄存器的值。用户可通过读取 RTC_DIV 寄存器来获得预分频计数器的当前值，而不停止计数器的工作，从而获得精确的时间测量。此寄存器是只读寄存器，其值在 RTC_PRL 或 RTC_CNT 寄存器中的值发生改变后，由硬件重新装载。

位	符号	访问	描述
31:4	-	R	保留
3:0	DIV[19:16]	R	预分频余数高位

18.4.6　预分频余数寄存器低位 (RTC_DIVL)

地址偏移：0x14

复位值：0x0000_8000

描述：在 TR_CLK 的每个周期里，RTC 预分频器中计数器的值都会被重新设置为 RTC_PRL 寄存器的值。用户可通过读取 RTC_DIV 寄存器来获得预分频计数器的当前值，而不停止计数器的工作，从而获得精确的时间测量。此寄存器是只读寄存器，其值在 RTC_PRL 或 RTC_CNT 寄存器中的值发生改变后，由硬件重新装载。

位	符号	访问	描述
31:16	-	R	保留
15:0	DIV[15:0]	R	预分频余数低位

18.4.7　计数器寄存器高位 (RTC_CNTH)

地址偏移：0x18

复位值：0x0000_0000

描述：RTC 核心有一个 32 位可编程计数器，可通过 2 个 16 位寄存器访问，本寄存器即是这 2 个 16 位寄存器中的高位。计数器以预分频产生的 TR_CLK 时间基准为参考进行计数。仅当 RTC_CRL 中的 RTOFF 为 1 时才允许对本寄存器写入，写入的值能直接装载到相应计数器中，并重新装载预分频器。当进行读操作

时，直接返回计数器内的值 (系统时间)

位	符号	访问	描述
31:16	-	R	保留
15:0	CNT[31:16]	RW	计数器高位

18.4.8　计数器寄存器低位 (RTC_CNTL)

地址偏移：0x1C

复位值：0x0000_0000

描述：RTC 核心有一个 32 位可编程计数器，可通过 2 个 16 位寄存器访问，本寄存器即是这 2 个 16 位寄存器中的低位。计数器以预分频产生的 TR_CLK 时间基准为参考进行计数。仅当 RTC_CRL 中的 RTOFF 为 1 时才允许对本寄存器写入，写入的值能直接装载到相应计数器中，并重新装载预分频器。当进行读操作时，直接返回计数器内的值 (系统时间)

位	符号	访问	描述
31:16	-	R	保留
15:0	CNT[15:0]	RW	计数器低位

18.4.9　闹钟寄存器高位 (RTC_ALRH)

地址偏移：0x20

复位值：0x0000_FFFF

描述：当 32 位可编程计数器的值与 RTC_ALR 中的 32 位值相等时即触发一个闹钟事件，并可以产生闹钟中断。本寄存器即为配置闹钟的高位寄存器，仅当 RTC_CRL 中的 RTOFF 为 1 时，才允许对本寄存器写入。

位	符号	访问	描述
31:16	-	R	保留
15:0	ALR[31:16]	W	闹钟高位

18.4.10　闹钟寄存器低位 (RTC_ALRL)

地址偏移：0x24

复位值：0x0000_FFFF

描述：当 32 位可编程计数器的值与 RTC_ALR 中的 32 位值相等时即触发一个闹钟事件，并可以产生闹钟中断。本寄存器即为配置闹钟的低位寄存器，仅当 RTC_CRL 中的 RTOFF 为 1 时，才允许对本寄存器写入。

位	符号	访问	描述
31:16	-	R	保留
15:0	ALR[15:0]	W	闹钟低位

第十九章　独立看门狗 (IWDG)

19.1　IWDG 简介

WB32F10xxx 内置两个看门狗，提供了更高的安全性、时间的精确性和使用的灵活性。两个看门狗设备 (独立看门狗和窗口看门狗) 可用来检测和解决由软件错误引起的故障；当计数器达到给定的超时值时，触发一个中断 (仅适用于窗口型看门狗) 或产生系统复位。

独立看门狗 (IWDG) 由专用的低速时钟 (LSI) 驱动，即使主时钟发生故障它也仍然有效。窗口看门狗由从 APB 时钟分频后得到的时钟驱动，通过可配置的时间窗口来检测应用程序非正常的过迟或过早的操作。

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。WWDG 最适合那些要求看门狗在精确计时窗口起作用的应用程序。

19.2　IWDG 主要特性

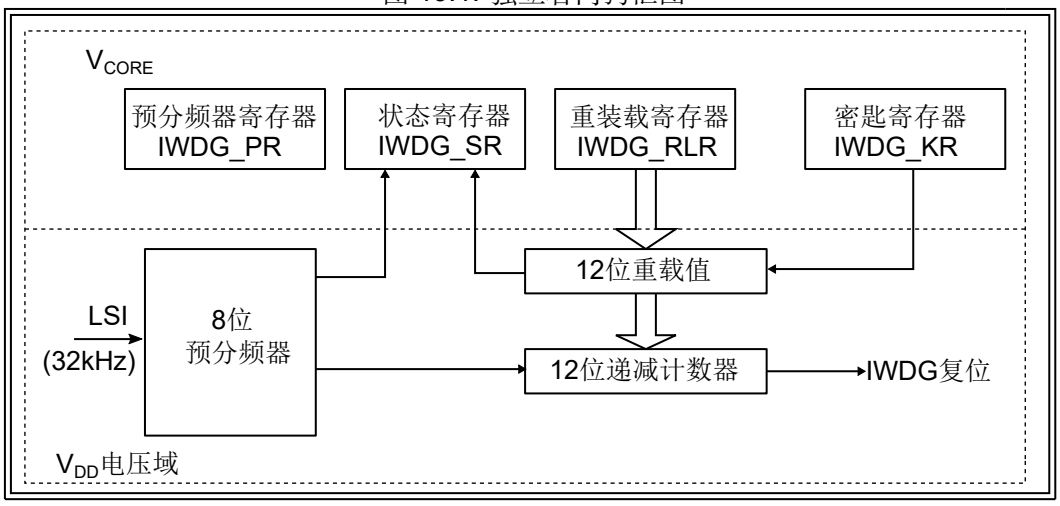
- 自由运行递减计数器
- 时钟由独立 RC 振荡器提供（可在待机和停止模式下运行）
- 看门狗被激活后，则在计数器计数至 0x000 时产生复位

19.3　IWDG 功能描述

图 19.1给出了独立看门狗模块的功能框图。

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，只要在键寄存器 IWDG_KR 中写入 0xAAAA，IWDG_RLR 中的值就会被重新加载到计数器，从而避免产生看门狗复位

图 19.1: 独立看门狗框图



看门狗功能由V_{CORE}电压域供电，在停止模式和待机模式下仍能工作。

19.3.1　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。

19.3.2　寄存器访问保护

IWDG_PR 和 IWDG_RLR 寄存器具有写保护功能。要修改这两个寄存器的值，必须先向 IWDG_KR 寄存器中写入 0x5555。以不同的值写入这个寄存器将会打乱操作顺序，寄存器将重新被保护。重装载操作 (即写入 0xAAAA) 也会启动写保护功能。

状态寄存器指示预分频值和递减计数器是否正在被更新。

19.3.3　调试模式

当微控制器进入调试模式时 (Cortex-M3 核心停止)，根据调试模块中的 DBG_IWDG_STOP 配置位的状态，IWDG 的计数器能够继续工作或停止。

表 19.1: 看门狗超时时间 (32kHz 的输入时钟 (LSI))

预分频系数	PR[2:0] 位	最短时间 (ms) RL[11:0] = 0x000	最长时间 (ms) RL[11:0] = 0xFFF
/4	0	0.125	512
/8	1	0.25	1024
/16	2	0.5	2048
/32	3	1.0	4096
/64	4	2.0	8192
/128	5	4.0	16384
/256	(6 或 7)	8.0	32768

19.4　寄存器描述

19.4.1　键寄存器 (IWDG_KR)

地址偏移量：0x00
复位值：0x0000 0000

表 19.2: 键寄存器 (IWDG_KR) 位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	KEY	W	键值 (Key value)（只能写，读为 0x0000） 必须每隔一段时间便通过软件对这些位写入键值 0xAAAA，否则当计数器计数到 0 时，看门狗会产生复位。 写入键值 0x5555 可使能对 IWDG_PR、IWDG_RLR 寄存器的访问 写入键值 0xCCCC 可启动看门狗（选中硬件看门狗选项的情况除外）

19.4.2　预分频寄存器 (IWDG_PR)

地址偏移量：0x04
复位值：0x0000 0000

表 19.3: 预分频寄存器 (IWDG_PR) 位描述

位	符号	访问	描述
31:3	-	R	保留
2:0	PR	RW	预分频因子 (Prescaler divider) 这些位具有写保护设置，参见19.3.2节。通过设置这些位来选择计数器时钟的预分频因子。要改变预分频因子，IWDG_SR 寄存器的 PVU 位必须为 0。 000: 预分频因子 =4　100: 预分频因子 =64 001: 预分频因子 =8　101: 预分频因子 =128 010: 预分频因子 =16　110: 预分频因子 =256 011: 预分频因子 =32　111: 预分频因子 =256

注：对此寄存器进行读操作，将从 *VDD* 电压域返回预分频值。如果写操作正在进行，则读回的值可能是无效的。因此，只有当 *IWDG_SR* 寄存器的 *PVU* 位为 *0* 时，读出的值才有效。

19.4.3　重装载寄存器 (IWDG_RLR)

地址偏移量：0x08
复位值：0x0000 0FFF

表 19.4: 重装载寄存器 (IWDG_RLR) 位描述

位	符号	访问	描述
31:12	-	R	保留
11:0	RL	RW	看门狗计数器重装载值 (Watchdog counter reload value) 这些位具有写保护功能。用于定义看门狗计数器的重装载值，每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此重装载值和时钟预分频值来计算。 只有当 IWDG_SR 寄存器中的 RVU 位为 0 时，才能对此寄存器进行修改。

注：对此寄存器进行读操作，将从 *VDD* 电压域返回预分频值。如果写操作正在进行，则读回的值可能是无效的。因此，只有当 *IWDG_SR* 寄存器的 *RVU* 位为 *0* 时，读出的值才有效。

19.4.4　状态寄存器 (IWDG_SR)

地址偏移量：0x0C

复位值：0x0000 0000

表 19.5: 状态寄存器（IWDG_SR）位描述

位	符号	访问	描述
31:2	-	R	保留
1	RVU	R	看门狗计数器重装载值更新 (Watchdog counter reload value update) 此位由硬件置’1’用来指示重装载值的更新正在进行中。当在 VDD 域中的重装载更新结束后，此位由硬件清’0’(最多需 5 个 40kHz 的 RC 周期)。重装载值只有在 RVU 位被清’0’后才可更新。
0	PVU	R	看门狗预分频值更新 (Watchdog prescaler value update) 此位由硬件置’1’用来指示预分频值的更新正在进行中。当在 VDD 域中的预分频值更新结束后，此位由硬件清’0’(最多需 5 个 32kHz 的 RC 周期)。预分频值只有在 PVU 位被清’0’后才可更新。

注：如果在应用程序中使用了多个重装载值或预分频值，则必须在 *RVU* 位被清除后才能重新改变预装载值，在 *PVU* 位被清除后才能重新改变预分频值。然而，在预分频和 / 或重装值更新后，不必等待 *RVU* 或 *PVU* 复位，可继续执行下面的代码。*(* 即是在低功耗模式下，此写操作仍会被继续执行完成。*)*

第二十章　窗口看门狗 (WWDG)

20.1　WWDG 简介

窗口看门狗通常被用来监测，由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。除非递减计数器的值在 T6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个 MCU 复位。在递减计数器达到窗口寄存器数值之前，如果 7 位的递减计数器数值 (在控制寄存器中) 被刷新，那么也将产生一个 MCU 复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。

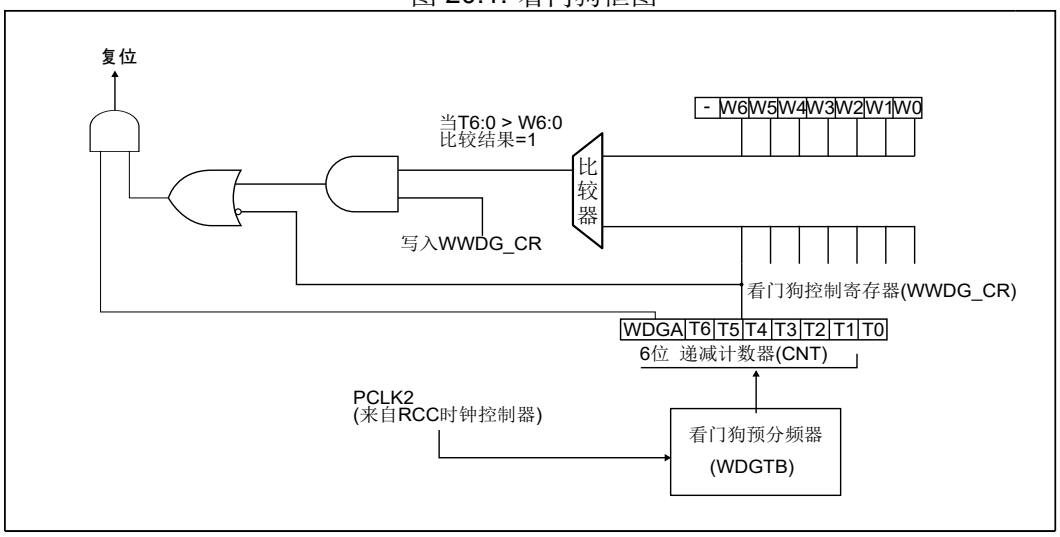
20.2　WWDG 主要特性

- 可编程的自由运行递减计数器
- 条件复位
  - 当递减计数器的值小于 0x40，(若看门狗被启动) 则产生复位。
  - 当递减计数器在窗口外被重新装载，(若看门狗被启动) 则产生复位。
- 如果启动了看门狗并且允许中断，当递减计数器等于 0x40 时产生早期唤醒中断 (EWI)，它可以被用于重装载计数器以避免 WWDG 复位。

20.3　WWDG 功能描述

如果看门狗被启动 (WWDG_CR 寄存器中的 WDGA 位被置’1’)，并且当 7 位 (T[6:0]) 递减计数器从 0x40 翻转到 0x3F(T6 位清零) 时，则产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。

图 20.1: 看门狗框图



应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在 WWDG_CR 寄存器中的数值必须在 0xFF 和 0xC0 之间：

- 启动看门狗

  在系统复位后，看门狗总是处于关闭状态，设置 WWDG_CR 寄存器的 WDGA 位能够开启看门狗，随后它不能再被关闭，除非发生复位。

- 控制递减计数器

  递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时，T6 位必须被设置，以防止立即产生一个复位。

  T[5:0] 位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频值是未知的。

  配置寄存器 (WWDG_CFR) 中包含窗口的上限值：要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载，图20.2描述了窗口寄存器的工作过程。

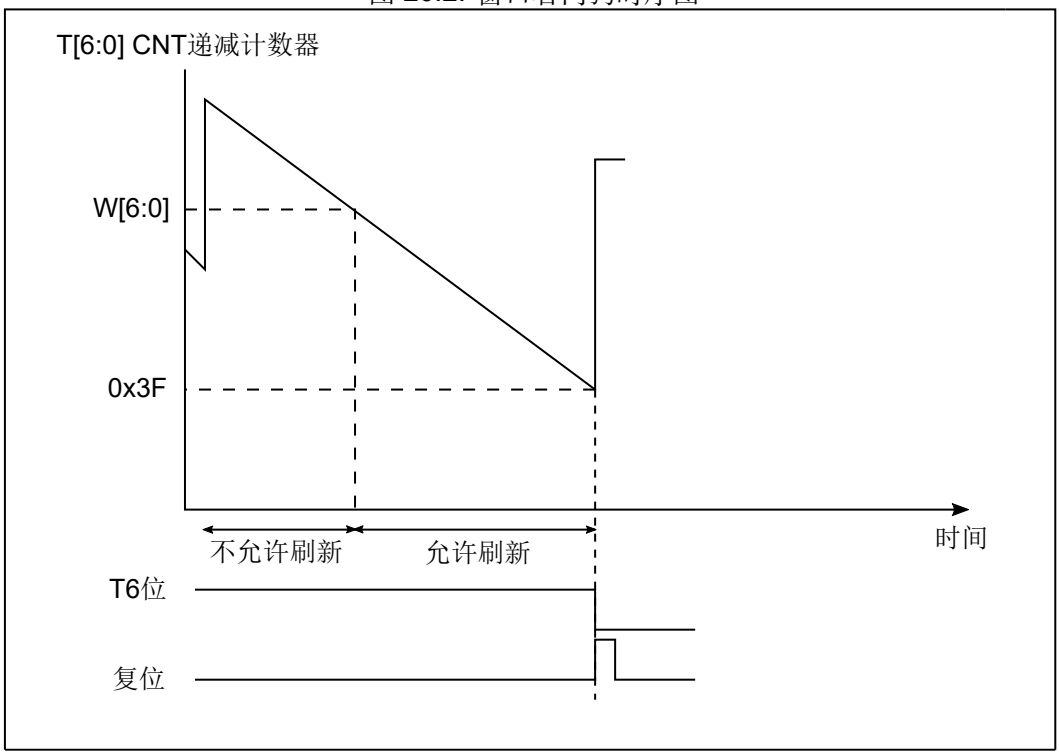
  另一个重装载计数器的方法是利用早期唤醒中断 (EWI)。设置 WWDG_CFR 寄存器中的 WEI 位开启该中断。当递减计数器到达 0x40 时，则产生此中断，相应的中断服务程序 (ISR) 可以用来加载计数器以防止 WWDG 复位。在 WWDG_SR 寄存器中写’0’可以清除该中断。

20.4　如何编写看门狗超时程序

可以使用图 20.2 提供的公式计算窗口看门狗的超时时间。

注：写入 *WWDG_CR* 寄存器时，始终将 *1* 写入 *T6* 位，以避免生成立即复位。

图 20.2: 窗口看门狗时序图



超值时的计算公式如下：

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WDGTB[1:0]} \times (T\,[5:0] + 1) \qquad (ms)$$

其中：

t_{WWDG}：WWDG 超时

t_{PCLK}：APB2 时钟周期，以 ms 为测量单位

4096: 对应于内部分频器的值

例如，假设 APB2 频率等于 48MHz，将 $WDGTB\,[1:0]$ 设置为 3 并将 $T\,[5:0]$ 设置为 63：

$$t_{WWDG} = 1/48000 \times 4096 \times 2^3 \times (63 + 1) = 43.69ms$$

有关 t_{WWDG} 的最小值和最大值，请参考数据手册。

20.5　调试模式

当微控制器进入调试模式时（Cortex ® -M3 内核停止），WWDG 计数器会根据 DBG 模块中的 DBG_WWDG_STOP 配置位选择继续正常工作或者停止工作。

20.6　寄存器描述

20.6.1　控制寄存器 (WWDG_CR)

地址偏移量：0x00

复位值：0x0000 007F

表 20.1: WWDG 控制寄存器（WWDG_CR）位描述

位	符号	访问	描述
31:8	-	R	保留
7	WDGA	RS	激活位 (Activation bit) 此位由软件置’1’，但仅能由硬件在复位后清’0’。当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗 1：启用看门狗
6:0	T	RW	7 位计数器 (MSB 至 LSB) (7-bit counter) 这些位用来存储看门狗的计数器值。每 (4096×2^{WDGTB}) 个 PCLK1 周期减 1。当计数器值从 40h 变为 3Fh 时 (T6 变成 0)，产生看门狗复位。

20.6.2　配置寄存器 (WWDG_CFR)

地址偏移量：0x04

复位值：0x0000 007F

表 20.2: WWDG 配置寄存器（WWDG_CFR）位描述

位	符号	访问	描述
31:10	-	R	保留
9	EWI	RS	提前唤醒中断 (Early wakeup interrupt) 此位若置’1’，则当计数器值达到 40h，即产生中断。 此中断只能由硬件在复位后清除。
8:7	WDGTB	RW	时基 (Timer base) 预分频器的时基可以设置如下： 00: CK 计时器时钟 (PCLK1 除以 4096) 除以 1 01: CK 计时器时钟 (PCLK1 除以 4096) 除以 2 10: CK 计时器时钟 (PCLK1 除以 4096) 除以 4 11: CK 计时器时钟 (PCLK1 除以 4096) 除以 8
6:0	W	RW	7 位窗口值 (7-bit window value) 这些位包含了用来与递减计数器进行比较用的窗口值。

20.6.3　状态寄存器 (WWDG_SR)

地址偏移量：0x00

复位值：0x0000 0000

表 20.3: 状态寄存器（WWDG_SR）位描述

位	符号	访问	描述
31:1	-	R	保留
0	EWIF	RC_W0	提前唤醒中断标志 (Early wakeup interrupt flag) 当计数器值达到 0x40 时此位由硬件置 1。它必须由软件通过写入 0 来清零。写入 1 不起作用。 如果不使能中断，此位也会被置 1。

第二十一章　闪存器控制模块 (FMC)

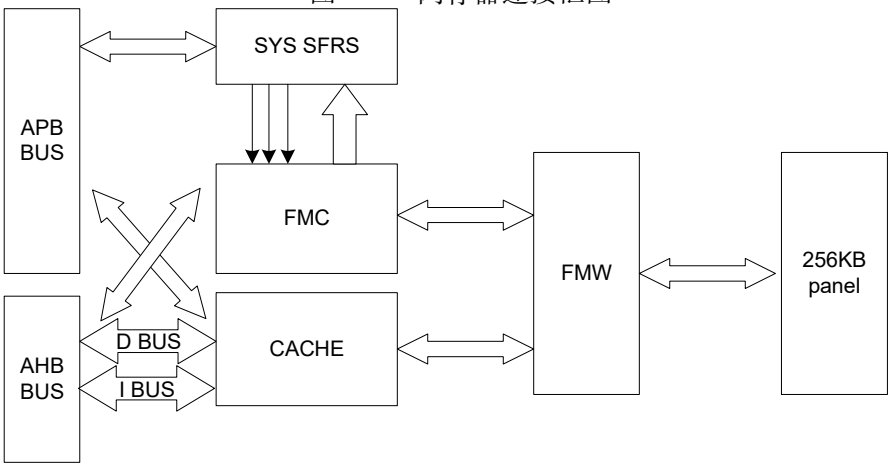
21.1　闪存器控制模块简介

闪存器控制模块连接在 AHB 总线上，实现对闪存器的自我配置、写操作控制、CRC 计算等。读取闪存器的操作则通过 cache 执行。

21.2　闪存器控制主要特性

- AHB 接口
- 持针对闪存器的以下操作：页编程（256B），页擦除（256B），块擦除（32KB），主存储器整体擦除，写页面锁存器，清除页面锁存器
- 支持块写保护（8KB）
- 支持对一段连续的闪存空间计算 CRC：$X^{16} + X^{15} + X^2 + 1$
- 芯片启动时，控制闪存器实现自我配置；同时读取信息块内容，更新芯片配置信息。
- 对闪存器的写操作正在执行时，暂停从闪存器读取指令和数据

图 21.1: 闪存器连接框图



21.3　功能描述

21.3.1　写操作

- 写操作流程
  - 擦除操作
    1. 主存储器支持页擦除，块擦除和全擦除
    2. 信息块部分仅支持页擦除
  - 编程操作
    1. 支持页编程（256B）
    2. 编程操作之前，必须先擦除之前的内容
    3. 编程操作的步骤是：
       1. 擦除页面锁存器
       2. 把准备写入闪存的内容写入页面锁存器
       3. 执行页编程命令

21.3.2　CRC 计算

- 支持对闪存一段连续的空间进行 CRC 计算，可以验证对闪存的编程操作是否正确
- CRC 计算的配置请参考寄存器 CRCON。
- 由于对闪存的读取操作是同步的，闪存支持的读周期时间是 15-20ns，需要根据系统频率适当配置 CRCON 的 Period 位。
- CRC 计算结束时，可以产生中断。

表 21.1: CRC 计算操作配置

系统频率	CRC 计算 (CRCON.PERIOD)
36MHz	0
48MHz	1
72MHz	2
96MHz	3
128MHz	4

21.4　寄存器描述

21.4.1　CRC 控制寄存器 (FMC_CRCON)

地址偏移量：0x4
复位值：0x0000 F000

位	符号	访问	描述
31-26	-	R	保留
25:16	CRCLEN	RW	CRC 计算的数据长度（以页为单位） 0:CRC 计算 1 页 1:CRC 计算 2 页 .....
15:12	PERIOD(1)	RW	CRC 计算时，每次读取闪存器的时钟数 0: 每次读取闪存器需要 1 个时钟周期 1: 每次读取闪存器需要 2 个时钟周期 ..... 15: 每次读取闪存器需要 16 个时钟周期
11:9	-	R	保留
8	CRCFIE	RW	CRC 计算完成中断允许位。为 1 时，CRC 计算结束时，产生中断
7:4	-	R	保留
3	SLOWRD	RW	控制 CRC 计算间隔 1: 相邻两次 CRC 计算之间相隔 16 个时钟 0: 相邻两次 CRC 计算之间没有间隔
2	PAUSE	RW	暂停 CRC 计算
1	CRCF	RW	CRC 计算完成标志位。 1:CRC 计算结束; 0: 如果 CRCEN 为 1，CRC 计算正在进行;
0	CRCEN	RW	写 1 启动 CRC，结束时自动清零

(1): 详细的计算方法参考表21.1。

- 注意：当闪存器的写操作正在执行时，对 CRCON 的写操作无效

21.4.2　地址寄存器 (FMC_ADDR)

地址偏移量：0x10
复位值：0x0000 0000
该寄存器的状态只在上电/掉电时复位，系统复位时内容不变。

- 整体擦除：不需要配置该寄存器
- 页擦除、页编程：页地址
- 块擦除：块地址
- 在 CRC 计算之前，配置 CRC 计算的起始地址。

21.4.3　数据寄存器 (FMC_DATA1)

地址偏移量：0x1C
复位值：0x0000 0000

- CRC 计算结束时，存储计算结果

21.4.4　缓存寄存器 (FMC_BUFx)x=0...63

地址偏移量：0x100～0x1FF

复位值：0x0000 0000

用户执行写页面锁存器的操作时，需要把准备写入页面锁存器的内容通过 APB 总线写入 BUF0 到 BUF63。每次写 BUFx，都会启动闪存器的写页面锁存器操作。

这个寄存器只能写，不能读。

在闪存器写操作正在执行时，对该寄存器的写操作无效。

第二十二章　USB 全速设备接口 (USB)

22.1　USB 简介

22.2　USB 主要特性

- 符合 USB2.0 全速设备的技术规范
- 在端点 0 之外，还支持 3 个 IN 和 3 个 OUT 端点
- 支持批量, 中断和同步传输
- 支持 USB 挂起/恢复操作

22.3　USB FIFO

1024 字节的专用 RAM 被用作 USB 的 FIFO 空间。该 FIFO 空间在端点 0~3 之间的分配如图22.1所示。

表 22.1: USB 专用 RAM 分配图

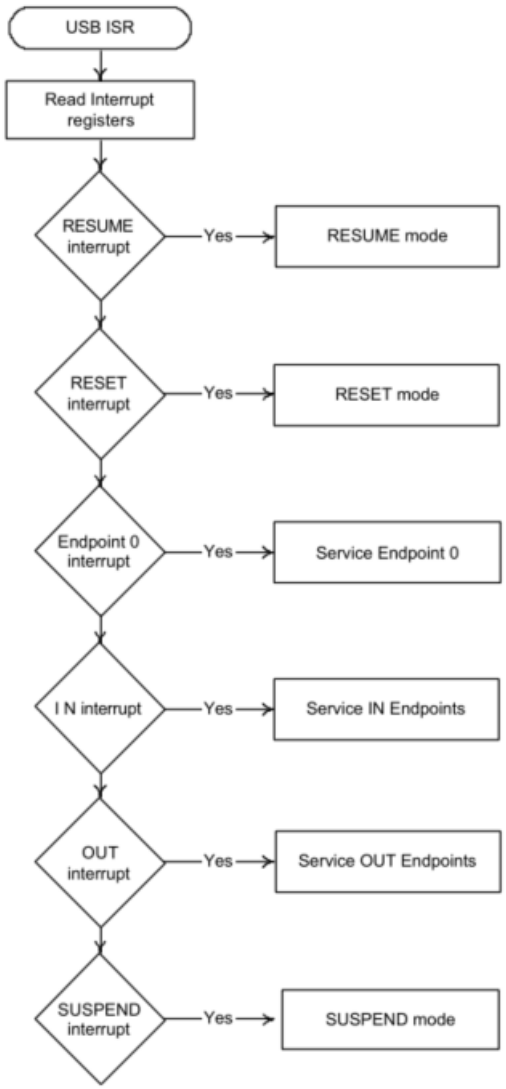
起始地址	Endpoint No.	空间大小
0x0000	Endpoint 0	64B
0x0040	Endpoint 1 IN	128B
0x00C0	Endpoint 1 OUT	128B
0x0140	Endpoint 2 IN	128B
0x01C0	Endpoint 2 OUT	128B
0x0240	Endpoint 3 IN	128B
0x02C0	Endpoint 3 OUT	128B
0x0340	未使用	192B

22.4　编程向导

22.4.1　USB 中断处理

当 CPU 被 USB 中断中断时，它需要读取中断状态寄存器以确定哪个端点引起了中断并跳转到相应的例程。如果多个端点导致中断，则应首先为端点 0 提供服务，然后再为其他端点提供服务。SUSPEND 中断应该最后处理。

图 22.1: USB 中断处理流程



22.5　USB 复位

当 USB 模块在总线上检测到 Reset 信号后，USB 模块会执行以下操作：

- 设置 FADDR 寄存器为 0。
- 设置 INDEX 寄存器为 0。

- 清空所有端点的 FIFO。
- 复位所有的控制/状态寄存器。
- 使能所有中断（除了 SUSPEND 中断）。
- 生成一个 Reset 中断。

当软件收到复位中断时，应该关闭所有打开的管道并等待总线枚举开始。

22.6 挂起/恢复

当 USB 模块检测到 USB 总线上超过 3ms 没有任何活动，将产生一个 SUSPEND 中断（如果使能）。

当 USB 处于挂起模式时，由软件决定禁用哪些东西。

22.6.1 USB 模块在 SUSPEND 期间保持活动状态

USB 协议规定可以通过在 USB 总线上发送 Resume 信号使设备退出 SUSPEND 模式。

如果在 USB 处于挂起模式时 USB 模块保持活动状态，则 USB 模块将监视 USB 总线上的 Resume 信号。当 Resume 信号出现在总线上时，USB 模块将产生一个 Resume 中断。

22.6.2 USB 模块在 SUSPEND 期间被禁用

当收到 SUSPEND 中断时，软件可以通过停止其时钟来禁用 USB 模块。但是由于 USB 模块已禁用，因此无法检测到 USB 上的恢复信号。因此，需要一些其他的措施来检测 Resume 信号，以便可以重新启动 USB 模块的时钟。

22.6.3 远程唤醒

如果 USB 模块处于挂起模式时，软件想要启动远程唤醒，则应设置 POWER 寄存器 RESUME 位为 1（如果 USB 模块的时钟已停止，则需要在发生此写操作之前重新启动）。

软件应保持该位为 1 大约 10ms（最小值为 2ms，最大值为 15ms），然后将其重置为 0。此时集线器应接管驱动总线上的 Resume 信号。

注意：软件启动远程唤醒时不会产生 Resume 中断。

22.7 端点 0 处理

端点 0 是 USB 主要的控制端点。因此，为端点 0 提供服务所需的例程比为其他端点提供服务所需的例程更复杂。

软件需要处理通过端点 0 接收的所有标准设备请求。这些在 *Universal Serial Bus Specification Revision 2.0, Chapter 9* 中有详细的描述。这些设备请求的协议涉及每次传输的不同数量和类型的事务。为了适应这种情况，CPU 需要采用状态机方法来命令解码和处理。

标准设备请求可以分为三类：零数据请求（其中所有信息都包含在命令中），写请求（在命令后跟随附加的数据）和读请求（设备需要将数据发送回主机）。

本节介绍软件处理不同类型设备请求时必须执行的事件序列。

注意：与任何标准设备请求关联的 Setup 数据包应包含 8 字节命令。USB 模块将自动拒绝任何超过 8 字节的 Setup 数据包。

22.7.1　零数据请求

零数据请求将其所有信息都包含在 8 字节命令中，所有不需要传输其他数据。零数据标准设备请求的例子有：SET_FEATURE, CLEAR_FEATURE, SET_ADDRESS, SET_CONFIGURATION, SET_INTERFACE。

当软件收到端点 0 中断时，CSR0 寄存器的 OUTPKTRDY 位被设置为 1。软件应从端点 0 FIFO 中读取 8 字节命令，解码并采取适当的操作。例如，如果命令是 SET_ADDRESS，则命令中包含的 7 位地址值应写入 FADDR 寄存器。

然后应设置 CSR0 寄存器的 SVDOUTPKTRDY 位（表示已从 FIFO 读取命令）和 DATAEND 位（表示此请求不再需要其他数据）。

当主机移动到请求的状态阶段时，将生成第二个端点 0 中断以指示请求已完成。软件无需进一步操作：第二个中断只是确认请求成功完成。

如果是无法识别的命令，或者由于某些其他原因无法执行，则在解码时，应写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位和 SENDSTALL 位。当主机移动到请求的状态阶段时，USB 模块将发送 STALL 以告知主机该请求未被执行。这将产生第二个端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

如果主机在设置 DATAEND 位后发送更多数据，则 USB 模块将发送 STALL。这将产生端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

22.7.2　写请求

写请求即在 8 字节命令之后附加从主机发送的数据包。写入标准设备请求的例子有：SET_DESCRIPTOR。

当软件收到端点 0 中断时，CSR0 寄存器的 OUTPKTRDY 位被设置为 1。软件应从端点 0 FIFO 中读取 8 字节命令并解析。

与零数据请求一样，应该写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位（表示该命令已从 FIFO 读取），但在这种情况下，不应设置 DATAEND 位（CSR0[3]）（表示预计会有更多数据）。

当收到第二个端点 0 中断时，应读取 CSR0 寄存器以检查端点状态。如果 OUTPKTRDY 位被设置为 1 表示已接收到数据包。然后应读取 COUNT0 寄存器以确定该数据包的大小。然后可以从端点 0 FIFO 读取数据包。

如果与请求关联的数据长度（由命令中的 *wLength* 指示）大于端点 0 的最大数据包大小，则将发送更多数据包。在这种情况下，应写入 CSR0 以设置 SVDOUTPKTRDY 位，但不应设置 DATAEND 位。

当收到所有预期的数据包时，应写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位和 DATAEND 位（表示不再需要数据）。

当主机移动到请求的状态阶段时，将生成另一个端点 0 中断以指示请求已完成。软件无需进一步操作，中断只是确认请求成功完成。

如果是无法识别的命令，或者由于某些其他原因无法执行，则在解码时，应写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位和 SENDSTALL 位。当主机发送更多数据时，USB 模块将发送 STALL 以告知主机该请求未被执行。这将产生端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

如果主机在设置 DATAEND 位后发送更多数据，则 USB 模块将发送 STALL。这将产生端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

22.7.3　读请求

在 8 字节命令之后，读取请求具有从该功能发送到主机的数据包。读取标准设备请求的例子有：GET_CONFIGURATION，GET_INTERFACE，GET_DESCRIPTOR，GET_STATUS，SYNCH_FRAME。

当软件收到端点 0 中断时，CSR0 寄存器的 OUTPKTRDY 位被设置为 1。软件应从端点 0 FIFO 中读取 8 字节命令并解析。然后应该写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位（表示该命令已从 FIFO 读取）。

然后，要发送到主机的数据应写入端点 0 FIFO。如果要发送的数据大于端点 0 的最大数据包大小，则只应将最大数据包大小写入 FIFO。然后应写入 CSR0 寄存器以设置 INPKTRDY 位（表示 FIFO 中有一个数据包要发送）。当数据包发送到主机时，将生成另一个端点 0 中断，并且可以将下一个数据包写入 FIFO。

当最后一个数据包写入 FIFO 时，应写入 CSR0 寄存器以设置 INPKTRDY 位和 DATAEND 位（表示该数据包后没有更多数据）。

当主机移动到请求的状态阶段时，将生成另一个端点 0 中断以指示请求已完成。软件无需进一步操作，中断只是确认请求成功完成。

如果是无法识别的命令，或者由于某些其他原因无法执行，则在解码时，应写入 CSR0 寄存器以设置 SVDOUTPKTRDY 位和 SENDSTALL 位。当主机请求数据时，USB 模块将发送 STALL 以告知主机该请求未被执行。这将产生端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

如果主机在设置 DATAEND 位后请求更多数据，则 USB 模块将发送 STALL。这将产生端点 0 中断，并将设置 SENTSTALL 位（CSR0[2]）。

22.7.4　端点 0 的状态

端点 0 控制需要三种状态：IDLE 状态，TX 状态和 RX 状态。对应于控制传输的不同阶段。

上电或复位时的默认状态应为 IDLE 状态。

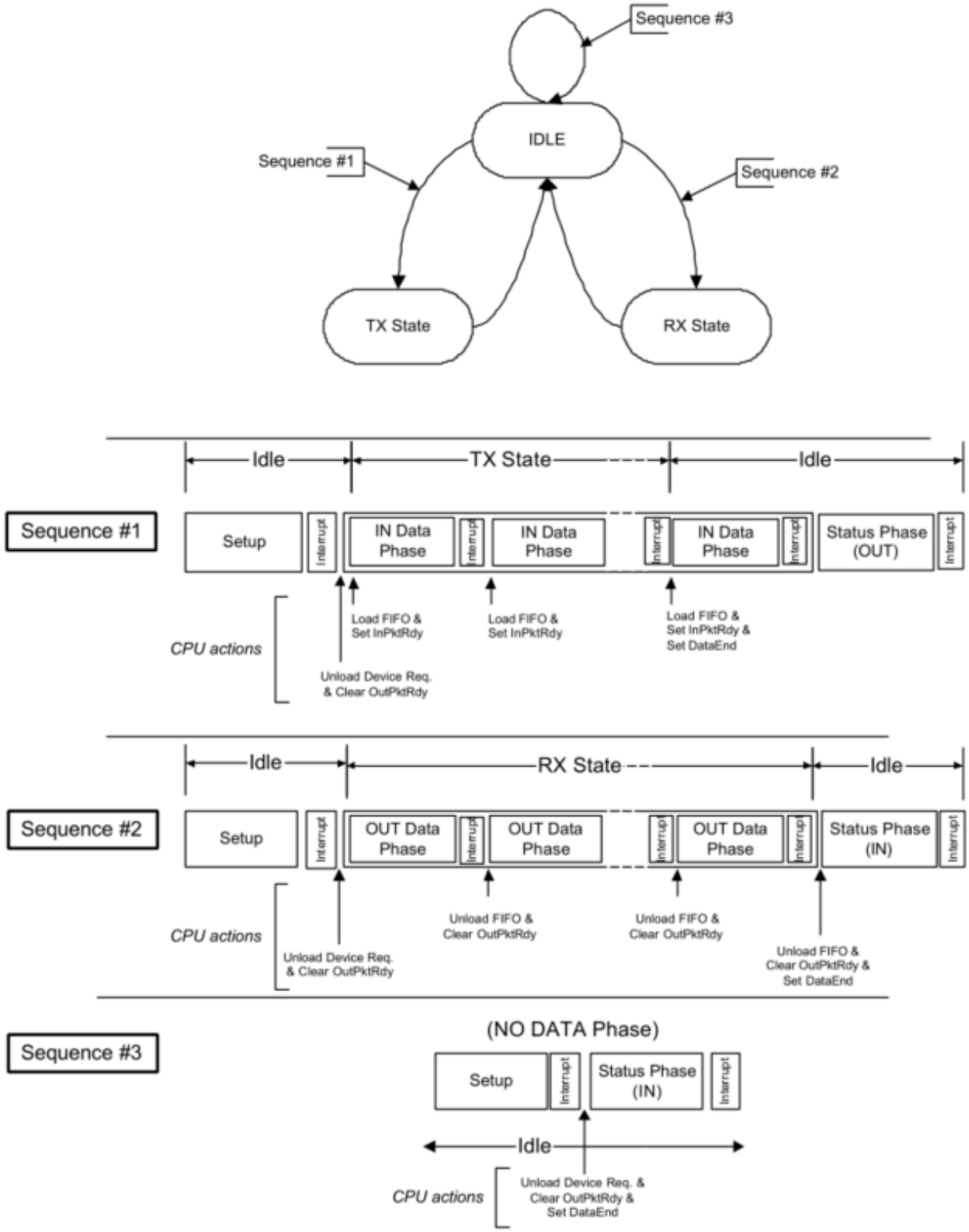
当端点 0 处于 IDLE 状态时 OUTPKTRDY（CSR0[0]）位被置位，表示新的设备请求。一旦从 FIFO 卸载了设备请求，USB 模块就对描述符进行解码，以查找是否存在数据阶段，如果存在，则查找控制传输的数据阶段的方向（以便设置 FIFO 方向）。

根据数据阶段的方向，端点 0 进入 TX 状态或 RX 状态。如果没有数据阶段，则端点 0 保持在 IDLE 状态以接受下一个设备请求。

软件在传输的不同阶段需要采取的操作（例如：加载 FIFO，设置 INPKTRDY 位），见图22.2。

请注意，USB 模块根据数据阶段的方向改变 FIFO 方向，而与软件无关。

图 22.2: 端点 0 的状态



22.7.5　端点 0 中断服务例程

当发生下列情况时产生端点 0 中断:

- 当收到一个数据包并且已写入端点 0 的 FIFO 时，OUTPKTRDY（CSR0[0]）被硬件置 1。
- 当 FIFO 中的数据包成功传输到主机后，INPKTRDY 位（CSR0[1]）被硬件清 0。
- 在控制传输因违反协议而结束后，硬件将 SENTSTALL 位（CSR0[2]）置 1。
- 因控制传输在软件设置 DATAEND 位（CSR0[3]）之前结束，导致硬件将 SETUPEND 位（CSR0[4]）置 1。

每当进入端点 0 服务例程时，固件必须首先检查当前控制传输是否由于 STALL 条件或控制传输过早结束而结束。如果控制传输由于 STALL 条件而结束，则将设置 SENTSTALL 位。如果控制传输由于控制传输过早结束而结束，则将设置 SETUPEND 位。在任何一种情况下，固件都应该中止处理当前控制传输并将状态设置为 IDLE。

一旦固件确定中断不是由非法总线状态生成的，则采取的下一个操作取决于端点状态。

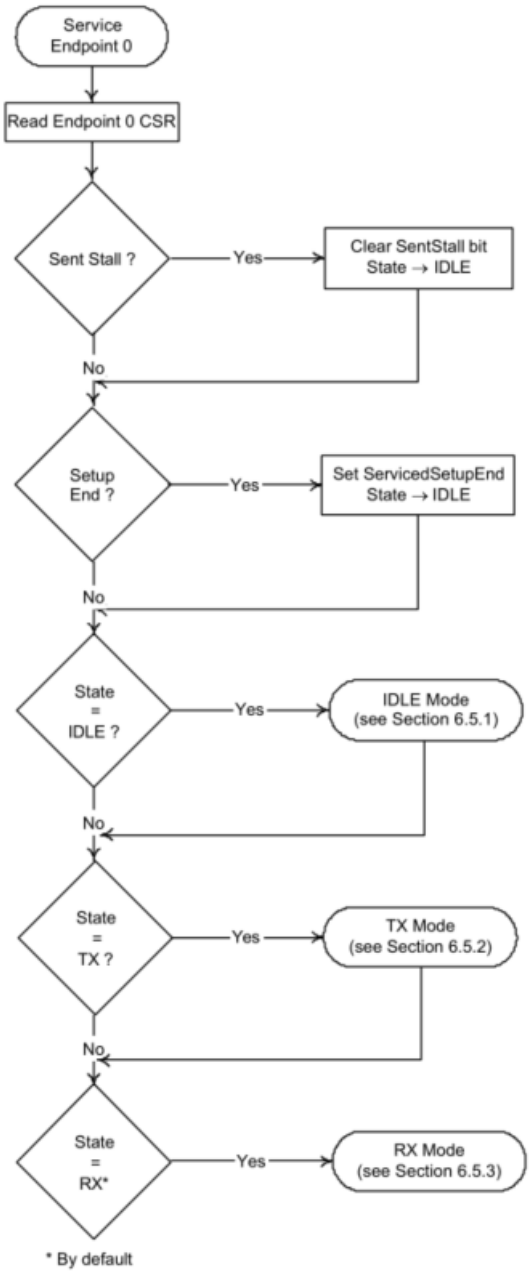
如果端点 0 处于空闲状态，则可以生成中断的唯一有效原因是模块从 USB 总线接收数据。服务例程必须通过 OUTPKTRDY 位（CSR0[0]）来检查这一点。如果该位置 1，则模块已收到 SETUP 数据包。必须从 FIFO 卸载并解码以确定 USB 模块将要采取的操作。根据 SETUP 数据包中包含的命令，端点 0 将进入以下三种状态之一：

- 如果该命令是单数据包事务（SET_ADDRESS，SET_INTERFACE 等）而没有任何数据阶段，则端点 0 将保持 IDLE 状态。
- 如果命令具有 OUT 数据阶段（SET_DESCRIPTOR 等），则端点将进入 RX 状态。
- 如果命令具有 IN 数据阶段（GET_DESCRIPTOR 等），则端点将进入 TX 状态。

如果端点处于 TX 状态，则中断指示模块已接收到 IN 令牌并且已发送 FIFO 中的数据。如果主机仍然期望更多数据，则固件必须通过在 FIFO 中放置更多的数据包来响应此操作，或者通过设置 DATAEND 位指示数据阶段完成。一旦事务的数据阶段完成，端点 0 应返回到 IDLE 状态以等待下一个控制事务。

如果端点处于 RX 状态，则中断指示已收到数据包。固件必须通过从 FIFO 卸载接收的数据来响应。然后，固件必须确定它是否已收到所有预期的数据。如果有，则固件应设置 DATAEND 位并将端点 0 返回到 IDLE 状态。如果需要更多数据，固件应设置 SVDOUTPKTRDY 位以指示它已读取 FIFO 中的数据并使端点处于 RX 状态。

图 22.3: 端点 0 中断服务例程

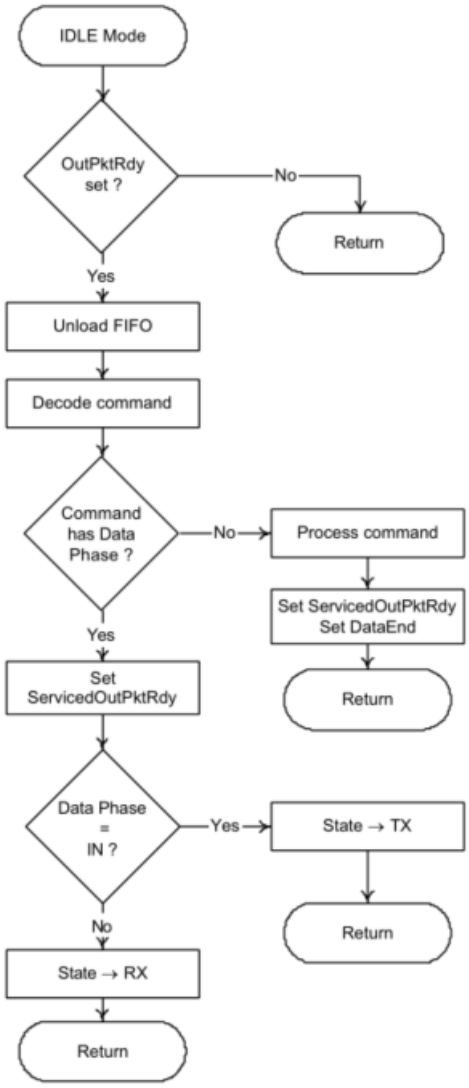


IDLE 状态

IDLE 状态是控制端点 0 在上电或复位时的默认状态，是 RX 和 TX 状态终止时控制端点 0 应返回的状态。

它也是处理控制传输 SETUP 阶段的状态。

图 22.4: 控制传输 SETUP 阶段



TX 状态

当端点处于 TX 状态时，所有到达的 IN 令牌都需要被视为数据阶段的一部分，直到所需的数据量已发送到主机。如果在端点处于 TX 状态时接收到 SETUP 或 OUT 令牌，则这将导致发生 SETUPEND 条件，因为模块仅期望 IN 令牌。

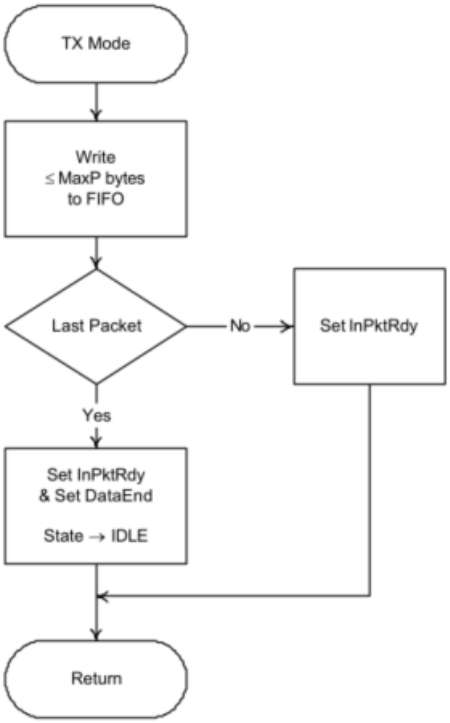
在发送预期数据量之前，三个事件可能导致 TX 状态终止：

- 主机发送无效令牌导致 SETUPEND 条件（CSR0[4] 置位）
- 固件发送包含小于端点 0 最大数据包大小的数据包
- 固件发送空数据包

在事务终止之前，固件只需在接收到指示已从 FIFO 发送数据包的中断时加载 FIFO。（清除 InPktRdy 时会产生中断）

当固件强制终止传输时（通过发送短数据包或空数据包），它应该设置 DATAEND 位（CSR0[3]）以向模块指示数据阶段已完成并且模块接下来应该接收到确认包。

图 22.5: 控制传输 IN 数据阶段



RX 状态

在 RX 状态下，所有到达的数据都应被视为数据阶段的一部分，直到收到预期的数据量。如果端点处于 RX 状态时收到 SETUP 或 IN 令牌，这将导致发送 SETUPEND 条件，因为模块只需要 OUT 令牌。

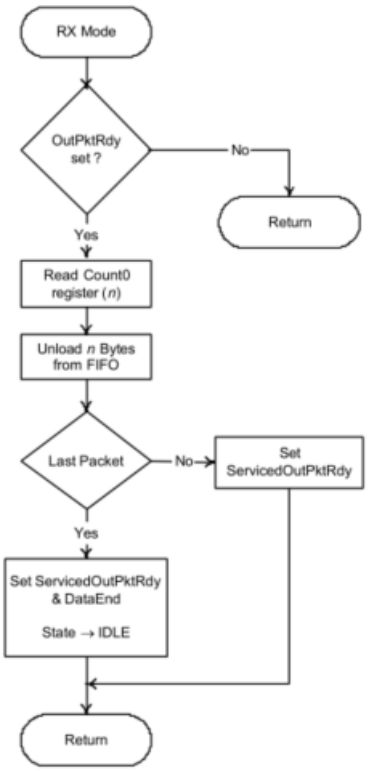
在收到预期数据量之前，三个事件可能导致 RX 状态终止：

- 主机发送无效令牌导致 SETUPEND 条件（CSR0[4] 置位）
- 主机发送包含小于端点 0 最大数据包大小的数据包
- 主机发送空数据包

在事务终止之前，固件只需在接收到指示新数据已到达的中断（OUTPKTRDY 位置 1）时卸载 FIFO，并通过设置 SVDOUTPKTRDY 位清除 OUTPKTRDY 位。

当固件检测到传输终止时（通过接收预期的数据量或空数据包），它应设置 DATAEND 位（CSR0[3]）以向模块指示数据阶段已完成并且模块接下来应该接收一个确认包。

图 22.6: 控制传输 OUT 数据阶段



22.7.6　错误处理

由于 USB 上的协议错误，主机过早地结束传输，或者如果控制器软件希望中止传输（例如因为它不能处理命令），控制传输可能会中止。

USB 模块将自动检测协议错误，并在以下条件下向主机发送 STALL 数据包：

1. 主机在写入请求的 OUT 数据阶段发送的数据多于命令中指定的数据。在设置 DATAEND 位（CSR0[3]）后主机发送 OUT 令牌时检测到此情况。
2. 主机在读取请求的 IN 数据阶段请求的数据多于命令中指定的数据。当主机在 CSR0 寄存器中的 DATAEND 位置 1 后发送 IN 令牌时，会检测到这种情况。
3. 主机在 OUT 数据包中发送超过 MaxP 数据字节。
4. 主机在读请求的 STATUS 阶段发送非零长度的 DATA1 数据包。

当 USB 模块发送 STALL 数据包时，它会设置 SENTSTALL 位（CSR0[2]）并生成中断。当软件收到设置了 SENTSTALL 位的端点 0 中断时，它应该中断当前传输，清除 SENTSTALL 位，然后返回 IDLE 状态。

如果主机通过在传输请求的所有数据之前进入 STATUS 阶段而过早地结束传输，或者在完成当前传输之前通过发送新的 SETUP 数据包，则将设置 SETUPEND 位（CSR0[4]）并且产生端点 0 中断。当软件收到设置了 SETUPEND 位的端点 0 中断时，它应该中止当前传输，设置 SVDSETUPEND 位（CSR0[7]），并返回 IDLE 状态。如果设置了 OUTPKTRDY 位（CSR0[0]），则表示主机已发送另一个 SETUP 数据包，然后软件应处理此命令。

如果软件想要中止当前传输，因为它无法处理命令或有其他内部错误，那么它应该设置 SENDSTALL 位（CSR0[5]）。然后，USB 模块将向主机发送 STALL 数据包，设置 SENTSTALL 位（CSR0[2]）并生成端点 0 中断。

22.8 Bulk IN 端点

Bulk IN 端点用于将非周期性数据从 USB 设备传输到主机。

有两个可选功能可用于 Bulk IN 端点：

- 双数据包缓冲

  如果写入 INMAXP 寄存器的值小于或等于 IN 端点 FIFO 大小的一半，则将自动启用双数据包缓冲。启用后，FIFO 中最多可存储两个数据包等待传输到主机。

- AutoSet 功能

  启用 AutoSet 功能后，当 INMAXP 字节数据包加载到 FIFO 时，将自动设置 INPKTRDY 位（INCSR1[0]）。

22.8.1 端点配置

在使用 Bulk IN 端点之前，必须将端点的最大数据包大小（以字节为单位）写入 INMAXP 寄存器。此值应与端点的标准端点描述符的 *wMaxPacketSize* 字段相同。此外，INTRINE 寄存器中相关中断使能位应设置为 1（如果此端点需要中断），并且 INCSR2 寄存器应设置如下：

7	AUTOSET	0/1	如果需要 AutoSet 功能，则设置为 1。
6	ISO	0	设置为 0 以启用 Bulk 协议。
3	FRCDATATOG	0	设置为 0 以允许正常数据切换操作。

首次配置 Bulk IN 端点时，在端点 0 上执行 SET_CONFIGURATION 或 SET_INTERFACE 命令后，应写入 INCSR1 寄存器以设置 CLRDATATOG 位。这将确保数据切换（由 USB 模块自动处理）以正确的状态启动。此外，如果 FIFO 中由任何数据包（由 FIFONE 位指示），则应通过 FLUSHFIFO 位置 1 来清空它们。注意：如果启用了双缓冲，则可能需要连续两次设置此位。

22.8.2 端点操作

当要通过 Bulk IN 管道传输数据时，需要将数据包装入 FIFO 并写入 INCSR1 寄存器以设置 INPKTRDY 位。发送数据包后，USB 模块将清除 INPKTRDY 位并产生中断，以便下一个数据包可以加载到 FIFO 中。如果启用了双数据包缓冲，则在加载第一个数据包并设置 INPKTRDY 位之后，USB 模块将立即清除 INPKTRDY 位并生成中断，以便可以将第二个数据包加载到 FIFO 中。软件应以相同的方式操作，在收到中断时加载数据包，无论是否启用双数据包缓冲。

数据包大小不得超过 INMAXP 寄存器中指定的大小。当要传输大于 INMAXP 的数据块时，必须将其作为多个数据包发送。这些数据包的大小应该是 INMAXP，除了残留的最后一个数据包。主机可以通过知道预期的数据总量来确定已经发送了用于传输的所有数据。或者，主机可以推断当它接收到小于 INMAXP 大小的数据包时已经发送了所有数据。在后一种情况下，如果数据块的总大小时 INMAXP 的倍数，则在发送所有数据之后，必须发送一个空包。这是通过在接收到下一个中断时设置 INPKTRDY 位来完成的，而不会将任何数据加载到 FIFO 中。

22.8.3 错误处理

如果软件要关闭 Bulk IN 管道，则应设置 SENDSTALL 位（INCSR1[4]）。当 USB 模块收到下一个 IN 令牌时，它会向主机发送一个 STALL，设置 SENTSTALL 位（INCSR1[5]）并产生一个中断。

当软件收到设置了 SENTSTALL 位（INCSR1[5]）的中断时，应清除 SENTSTALL 位。软件应该保留 SENDSTALL 位为 1，直到软件准备好重新启用 Bulk IN 管道。注意：如果主机由于某种原因未能收到 STALL 数据包，它将发送另一个 IN 令牌，因此建议保留 SENDSTALL 位置 1，直到软件准备好重新启用 Bulk IN 管道。重新启用管道时，应通过将 INCSR1 寄存器中的 CLRDATATOG 位置 1 来重新启动数据切换序列。

22.9 Bulk OUT 端点

Bulk OUT 端点用于将非周期性数据从主机传输到 USB 设备。

有两个可选功能可用于 Bulk OUT 端点：

- 双数据包缓冲

  如果写入 OUTMAXP 寄存器的值小于或等于 OUT 端点 FIFO 大小的一半，则将自动启用双数据包缓冲。启用后，FIFO 中最多可存储两个数据包。

- AutoClear 功能

  启用 AutoClear 功能后，当从 FIFO 卸载 OUTMAXP 字节数据包时，OUTPKTRDY 位（OUTCSR1[0]）将自动清零。

22.9.1 端点配置

在使用 Bulk OUT 端点之前，必须将端点的最大数据包大小（以字节为单位）写入 OUTMAXP 寄存器。此值应与端点的标准端点描述符的 *wMaxPacketSize* 字段相同。此外，INTROUTE 寄存器中相关中断使能位应设置为 1（如果此端点需要中断），并且 OUTCSR2 寄存器应设置如下：

7	AUTOCLR	0/1	如果需要 AutoClear 功能，则设置为 1。
6	ISO	0	设置为 0 以启用 Bulk 协议。

首次配置 Bulk OUT 端点时，在端点 0 上执行 SET_CONFIGURATION 或 SET_INTERFACE 命令后，应写入 OUTCSR1 以设置 CLRDATATOG 位。这将确保数据切换（由 USB 模块自动处理）以正确的状态启动。此外，如果 FIFO 中由任何数据包（由 OUTPKTRDY 位指示），则应通过 FLUSHFIFO 位置 1 来清空它们。注意：如果启用了双缓冲，则可能需要连续两次设置此位。

22.9.2 端点操作

当 Bulk OUT 端点接收到数据包时，OUTPKTRDY 位（OUTCSR1[0]）置 1 并产生中断。软件应读取端点的两个 OUTCOUNT 寄存器，以确定数据包的大小。软件应从 FIFO 读取数据包，然后清除 OUTPKTRDY 位。

数据包大小不应超过 OUTMAXP 寄存器中指定的大小(因为这应该是发送到主机的端点描述符的 *wMaxPacketSize* 字段中设置的值)。当要将大于 OUTMAXP 的数据块发送到设备时，它将分为多个数据包发送。除最后一个包含残留的数据包外，所有数据包的大小都是 OUTMAXP。软件可以使用特定于应用程序的方法来确定块的总大小，并因此确定何时接收到最后一个数据包。或者，当它接收到大小小于 OUTMAXP 的分组时，它可以推断已经接收到整个块。(如果数据块的总大小是 OUTMAXP 的倍数，则在数据之后将发送空数据包以表示传输已完成。)

22.9.3 错误处理

如果软件想要关闭 Bulk OUT 管道，它应该设置 SENDSTALL 位（OUTCSR1[5]）。当 USB 模块接收到下一个数据包时，它会向主机发送一个 STALL，设置 SENTSTALL 位（OUTCSR1[6]）并产生一个中断。

当软件收到设置了 SENTSTALL 位（OUTCSR1[6]）的中断时，应清除 SENTSTALL 位。软件应该保留 SENDSTALL 位为 1，直到软件准备好重新启用 Bulk OUT 管道。注意：如果主机由于某种原因未能收到 STALL 数据包，它将发送另一个数据包，因此建议保留 SENDSTALL 位置 1，直到软件准备好重新启用 Bulk OUT 管道。重新启用管道时，应通过将 OUTCSR1 寄存器中的 CLRDATATOG 位置 1 来重新启动数据切换序列。

22.10 中断 IN 端点

中断 IN 端点用于将周期性数据从 USB 设备传输到主机。

中断 IN 端点使用与 Bulk IN 端点相同的协议，并且可以以相同的方式使用。

中断 IN 端点还支持 Bulk IN 端点不支持的一个功能：支持数据切换位的连续切换。通过将 INCSR2 寄存器中的 FRCDATATOG 位置 1 来使能该功能。当此位设置为 1 时，无论是否从主机收到 ACK，USB 模块都会认为数据包已成功发送并切换端点的数据位。

22.11 中断 OUT 端点

中断 OUT 端点用于将周期性数据从主机传输到 USB 设备。

中断 OUT 端点使用与 Bulk OUT 端点几乎相同的协议，并且可以以相同的方式使用。

22.12 同步 IN 端点

同步 IN 端点用于将周期性数据从 USB 设备传输到主机。

有两个可选功能可用于同步 IN 端点：

- 双数据包缓冲

  如果写入 INMAXP 寄存器的值小于或等于 IN 端点 FIFO 大小的一半，则将自动启用双数据包缓冲。启用后，FIFO 中最多可存储两个数据包等待传输到主机。注意：对于同步 IN 端点，通常建议使用双数据包缓冲，以避免数据欠载。

- AutoSet 功能

  启用 AutoSet 功能后，当 INMAXP 字节数据包加载到 FIFO 时，将自动设置 INPKTRDY 位（INCSR1[0]）。但是，此功能对同步端点不是特别有用，因为传输的数据包通常不是最大数据包大小，并且需要在每个数据包之后访问 INCSR1 寄存器以检查欠载错误。

22.12.1 端点配置

在使用同步 IN 端点之前，必须将端点的最大数据包大小（以字节为单位）写入 INMAXP 寄存器。此值应与端点的标准端点描述符的 *wMaxPacketSize* 字段相同。此外，INTRINE 寄存器中相关中断使能位应设置为 1（如果此端点需要中断），并且 INCSR2 寄存器应设置如下：

7	AUTOSET	0/1	如果需要 AutoSet 功能，则设置为 1。
6	ISO	1	设置为 1 以启用同步协议。
3	FRCDATATOG	0	在同步模式下忽略。

22.12.2　端点操作

同步端点不支持数据重试，因此如果要避免数据欠载，则必须在接收到 IN 令牌之前将要发送到主机的数据加载到 FIFO 中。主机每帧将发送一个 IN 令牌，但帧内的时间可能会有所不同。如果在一帧结束附近然后在下一帧开始时接收到 IN 令牌，则几乎没有时间重新加载 FIFO。因此，同步 IN 端点通常需要双缓冲。

AutoSet 功能可以与同步 IN 端点一起使用，与 Bulk IN 端点的方式相同。但是，除非数据以绝对一致的速率从源到达，与主机的帧时钟同步，否则发送到主机的数据包的大小必须逐帧增加或减少以匹配源数据速率。这意味着实际的数据包大小并不总是 INMAXP 大小，表现为 AutoSet 功能无效。

每当数据包发送到主机时都会产生中断，软件可以使用此中断将下一个数据包加载到 FIFO 中，并以 Bulk IN 端点相同的方式设置 INCSR1 寄存器的 INPKTRDY 位。由于中断几乎可以在帧内的任何时间发生，取决于主机何时调度事务，这可能导致 FIFO 加载请求的不规则定时。如果端点的数据源来自某些外部硬件，则在加载 FIFO 之前等待每帧结束可能更方便，因为这将最小化对额外缓冲的要求。这可以通过使用 SOF 中断（INTRUSB[3]）或来自 USB 模块的外部 SOF_PULSE 信号来触发下一个数据包的加载来完成。当接收到 SOF 数据包时，每帧产生一次 SOF_PULSE。（USB 模块还维护一个外部帧计数器，因此当 SOF 数据包丢失时它仍然可以生成 SOF_PULSE）中断仍可用于设置 INCSR1 中的 INPKTRDY 位并检查数据溢出/欠载。

启动双缓冲同步 IN 管道可能是某些问题的根源。双缓冲要求在加载数据包之后才发送数据包。如果设备在主机设置管道之前至少加载第一个数据包（因此开始发送 IN 令牌），则没有问题。但是，如果主机在第一个数据包加载时已经开始发送 IN 令牌，则数据包可以在加载时在相同的帧中传输，具体取决于它是在接收到 IN 令牌之前还是之后加载。通过设置 POWER 寄存器中的 ISOUD 位可以避免这个潜在的问题。当该位设置为 1 时，加载到同步 IN 端点 FIFO 中的任何数据包将不会被发送，直到接收到下一个 SOF 数据包之后，从而确保数据包不会过早发送。

22.12.3　错误处理

如果端点在接收到 IN 令牌时其 FIFO 中没有数据，它将向主机发送空数据包并设置 INCSR1 寄存器中的 UNDERRUN 位。这表明软件没有为主机提供足够快的数据。由应用程序决定如何处理此错误条件。

如果软件每帧加载一个数据包，并且发现 INCSR1 寄存器中的 INPKTRDY 位在想要加载下一个数据包时已经设置，则表示尚未发送数据包（可能是因为来自主机的 IN 令牌已损坏）。由应用程序决定如何处理这种情况：可以选择通过设置 INCSR1 寄存器中的 FLUSHFIFO 位来清除未发送的数据包，或者可以选择跳过当前数据包。

22.13　同步 OUT 端点

同步 OUT 端点用于将周期性数据从主机传输到 USB 设备。

有两个可选功能可用于同步 OUT 端点：

- 双数据包缓冲

  如果写入 OUTMAXP 寄存器的值小于或等于 OUT 端点 FIFO 大小的一半，则将自动启用双数据包缓

冲。启用后，FIFO 中最多可存储两个数据包。注意：对于同步 OUT 端点，通常建议使用双数据包缓冲，以避免数据溢出。

- AutoClear 功能
  启用 AutoClear 功能后，当从 FIFO 卸载 OUTMAXP 字节数据包时，OUTPKTRDY 位（OUTCSR1[0]）将自动清零。但是，此功能对同步端点不是特别有用，因为传输的数据包通常不是最大数据包大小，并且需要在每个数据包之后访问 OUTCSR1 寄存器以检查溢出或 CRC 错误。

22.13.1 端点配置

在使用同步 OUT 端点之前，必须将端点的最大数据包大小（以字节为单位）写入 OUTMAXP 寄存器。此值应与端点的标准端点描述符的 *wMaxPacketSize* 字段相同。此外，INTROUTE 寄存器中相关中断使能位应设置为 1（如果此端点需要中断），并且 OUTCSR2 寄存器应设置如下：

7	AUTOCLR	0/1	如果需要 AutoClear 功能，则设置为 1。
6	ISO	1	设置为 1 以启用同步协议。

22.13.2 端点操作

同步端点不支持数据重试，因此如果要避免数据溢出，FIFO 中必须有空间在接收数据包时接受数据包。主机每帧将发送一个数据包，但帧内的时间可能会有所不同。如果在一帧结束附近接收到一个数据包而另一帧到达下一帧开始时，则几乎没有时间卸载 FIFO。因此，同步 OUT 端点通常需要双缓冲。

AutoClear 功能可与同步 OUT 端点一起使用，方法与 Bulk OUT 端点相同。但是，除非数据接收器以绝对一致的速率接收数据并且与主机的帧时钟同步，否则从主机发送的数据包的大小必须逐帧增加或减少以匹配所需的数据速率。这意味着实际数据包大小的大小并不总是 OUTMAXP，表现为 AutoClear 功能无效。

每当从主机接收到数据包时都会产生中断，并且软件可以使用此中断从 FIFO 卸载数据包并清除 OUTCSR1 寄存器中的 OUTPKTRDY 位，方法与 Bulk OUT 端点相同。由于中断几乎可以在帧内的任何时间发生，取决于主机何时调度事务，FIFO 卸载请求的时间可能是不规则的。如果端点的数据接收器要进入某些外部硬件，则最好在卸载 FIFO 之前等待每帧结束时最小化额外缓冲的要求。这可以通过使用 SOF 中断（INTRUSB[3]）或来自 USB 模块的外部 SOF_PULSE 信号来触发数据包的卸载来完成。当接收到 SOF 数据包时，每帧产生一次 SOF_PULSE。（USB 模块还维护一个外部帧计数器，因此当 SOF 数据包丢失时它仍然可以生成 SOF_PULSE）中断仍可用于清除 OUTCSR1 中的 OUTPKTRDY 位并检查数据溢出/欠载。

22.13.3 错误处理

如果在从主机接收数据包时 FIFO 中没有空间存储数据包，则 OUTCSR1 寄存器中的 OVERRUN 将置位。这表明软件没有足够快地为主机卸载数据。由应用程序决定如何处理此错误条件。

如果 USB 模块发现接收到的数据包有 CRC 错误，它仍会将数据包存储在 FIFO 中并设置 OUTPKTRDY 位（OUTCSR1[0]）和 DATAERROR 位（OUTCSR1[3]）。应用程序如何处理此错误情况取决于应用程序。

22.14 USB 寄存器描述

表 22.2: USB 寄存器简介

名称	偏移地址	描述	复位值
中断寄存器			
INTRIN	0x02	USB 输入端点中断标志寄存器	0x00
INTROUT	0x04	USB 输出端点中断标志寄存器	0x00
INTRUSB	0x06	USB 中断标志寄存器	0x00
INTRINE	0x07	USB 输入端点中断允许寄存器	0x00
INTROUTE	0x09	USB 输出端点中断允许寄存器	0x00
INTRUSBE	0x0B	USB 中断允许寄存器	0x00
公共寄存器			
FADDR	0x00	USB 功能地址寄存器	0x00
POWER	0x01	USB 电源控制寄存器	0x00
FRAMEL	0x0C	USB 帧号低字节寄存器	0x00
FRAMEH	0x0D	USB 帧号高字节寄存器	0x00
INDEX	0x0E	USB 端点索引寄存器	0x00
FIFO*x*	0x20 + 0x04 × *x*	USB 端点 *x*（*x*=0~3）FIFO 访问寄存器	NA
索引寄存器			
INMAXP	0x10	USB 输入端点最大包长配置寄存器	0x00
CSR0	0x11	USB 端点 0 控制/状态寄存器	0x00
INCSR1	0x11	USB 输入端点控制/状态寄存器 1	0x00
INCSR2	0x12	USB 输入端点控制/状态寄存器 2	0x20
OUTMAXP	0x13	USB 输出端点最大包长配置寄存器	0x00
OUTCSR1	0x14	USB 输出端点控制/状态寄存器 1	0x00
OUTCSR2	0x15	USB 输出端点控制/状态寄存器 2	0x00
COUNT0	0x16	USB 端点 0 数据计数寄存器	0x00
OUTCOUNTL	0x16	USB 输出端点数据计数低字节寄存器	0x00
OUTCOUNTH	0x17	USB 输出端点数据计数高字节寄存器	0x00

22.14.1 中断寄存器

这组寄存器用于 USB 模块的中断处理。

USB 输入端点中断标志寄存器 (USB_INTRIN)

偏移地址：0x02

复位值：0x00

USB_INTRIN 寄存器用于指示 IN 端点 1-3 和端点 0 的中断。

如果有中断产生，USB_INTRIN 中的相应位为 1；否则，该位为 0。读取该寄存器将清除该寄存器中的所有标志位。

表 22.3: USB 输入端点中断标志寄存器 (USB_INTRIN) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	IN3	R	IN 端点 3 中断标志。
2	IN2	R	IN 端点 2 中断标志。
1	IN1	R	IN 端点 1 中断标志。
0	EP0	R	端点 0 中断标志。

USB 输出端点中断标志寄存器 (USB_INTROUT)

偏移地址：0x04

复位值：0x00

USB_INTROUT 寄存器用于指示 OUT 端点 1-3 的中断。

如果有中断产生，USB_INTROUT 中的相应位为 1；否则，该位为 0。读取该寄存器将清除该寄存器中的所有标志位。

表 22.4: USB 输出端点中断标志寄存器 (USB_INTROUT) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	OUT3	R	OUT 端点 3 中断标志。
2	OUT2	R	OUT 端点 2 中断标志。
1	OUT1	R	OUT 端点 1 中断标志。
0	-	R	保留，始终读为 0。

USB 中断标志寄存器 (USB_INTRUSB)

偏移地址：0x06

复位值：0x00

USB_INTRUSB 寄存器用于指示 USB 的中断。

如果有中断产生，USB_INTRUSB 中的相应位为 1；否则，该位为 0。读取该寄存器将清除该寄存器中的所有标志位。

表 22.5: USB 中断标志寄存器 (USB_INTRUSB) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	SOFIS	R	帧起始中断标志。 该位在收到 SOF 令牌时被硬件置'1'。
2	RSTIS	R	复位中断标志。 当总线上检测到复位信号时该位被硬件置'1'。
1	RSUIS	R	恢复中断标志。 在 USB 模块处于挂起状态期间，当总线上检测到恢复信号时该位被硬件置'1'。
0	SUSIS	R	挂起中断标志。 当总线上检测到挂起信号时该位被硬件置'1'。

USB 输入端点中断允许寄存器 (USB_INTRINE)

偏移地址：0x07

复位值：0x0F

表 22.6: USB 输入端点中断允许寄存器 (USB_INTRINE) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	IN3E	RW	IN 端点 3 中断使能。 0：禁止 IN 端点 3 中断。 1：允许 IN 端点 3 中断。
2	IN2E	RW	IN 端点 2 中断使能。 0：禁止 IN 端点 2 中断。 1：允许 IN 端点 2 中断。
1	IN1E	RW	IN 端点 1 中断使能。 0：禁止 IN 端点 1 中断。 1：允许 IN 端点 1 中断。
0	EP0E	RW	端点 0 中断使能。 0：禁止端点 0 中断。 1：允许端点 0 中断。

USB 输出端点中断允许寄存器 (USB_INTROUTE)

偏移地址：0x09

复位值：0x0E

表 22.7: USB 输出端点中断允许寄存器 (USB_INTROUTE) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	OUT3E	RW	OUT 端点 3 中断使能。 0：禁止 OUT 端点 3 中断。 1：允许 OUT 端点 3 中断。
2	OUT2E	RW	OUT 端点 2 中断使能。 0：禁止 OUT 端点 2 中断。 1：允许 OUT 端点 2 中断。
1	OUT1E	RW	OUT 端点 1 中断使能。 0：禁止 OUT 端点 1 中断。 1：允许 OUT 端点 1 中断。
0	-	R	保留，始终读为 0。

USB 中断允许寄存器 (USB_INTRUSBE)

偏移地址：0x0B
复位值：0x06

表 22.8: USB 中断允许寄存器 (USB_INTRUSBE) 位描述

位	符号	访问	描述
7:4	-	R	保留
3	SOFIE	RW	帧起始中断使能。 0：禁止帧起始中断。 1：允许帧起始中断。
2	RSTIE	RW	复位中断使能。 0：禁止复位中断。 1：允许复位中断。
1	RSUIE	RW	恢复中断使能。 0：禁止恢复中断。 1：允许恢复中断。
0	SUSIE	RW	挂起中断使能。 0：禁止挂起中断。 1：允许挂起中断。

22.14.2 公共寄存器

USB 功能地址寄存器 (USB_FADDR)

偏移地址：0x00
复位值：0x00

表 22.9: USB 功能地址寄存器 (USB_FADDR) 位描述

位	符号	访问	描述
7	UPDATE	R	功能地址更新位。 当软件写 FADDR 寄存器时被硬件置 1。当新地址生效后，硬件将该位请 0。 0：最后写入 FADDR 的地址已经生效。 1：最后写入 FADDR 的地址尚未生效。
6:0	FADDR	RW	功能地址位。 保持 USB 的 7 位功能地址。当端点 0 收到 SET_ADDRESS 标准设备请求时，软件应写该地址。设备请求完成后新地址生效。

USB 电源控制寄存器 (USB_POWER)

偏移地址：0x01
复位值：0x00

表 22.10: USB 电源控制寄存器 (USB_POWER) 位描述

位	符号	访问	描述
7	ISOUD	RW	ISO 更新位。 该位影响所有 IN 同步端点。 0：当软件向 INRDY 写 1 时，USB 模块在收到下一个 IN 令牌后发送数据包。 1：当软件向 INRDY 写 1 时，USB 模块等待 SOF 令牌，然后发送数据包。如果在 SOF 令牌之前收到 IN 令牌，则 USB 模块将发送长度为 0 的数据包。
6:4	-	R	保留，始终读为 0。
3	USBRST	R	复位检测。 当总线上检测到复位信号时该位被硬件置位，直到总线恢复到空闲状态。 0：总线上未检测到复位信号。 1：总线上检测到复位信号。
2	RESUME	RW	强制恢复。 在挂起状态 (SUSMD=1) 向该位写 1 将强制 USB 模块在总线上产生恢复信号（一个远程唤醒事件）。软件应在 10~15ms 后向该位写 0，以结束恢复信号。
1	SUSMD	R	挂起状态。 当 USB 模块进入挂起模式时，该位被硬件置 1。当软件向 RESUME 位写 0（后面跟随一次远程唤醒）或在检测到总线上的恢复信号后读取 INTRUSB 寄存器时硬件将该位清 0。 0：USB 模块不处于挂起模式。 1：USB 模块处于挂起模式。
0	SUSEN	RW	进入挂起模式使能。 0：当检测到总线上的挂起信号时，USB 模块不进入挂起模式。 1：当检测到总线上的挂起信号时，USB 模块进入挂起模式。

USB 帧号低字节寄存器 (USB_FRAMEL)

偏移地址：0x0C
复位值：0x00

表 22.11: USB 帧号低字节寄存器 (USB_FRAMEL) 位描述

位	符号	访问	描述
7:0	FRAMEL	R	帧号低字节位。 该寄存器包含最后接收帧号的位 7-0。

USB 帧号高字节寄存器 (USB_FRAMEH)

偏移地址：0x0D
复位值：0x00

表 22.12: USB 帧号高字节寄存器 (USB_FRAMEH) 位描述

位	符号	访问	描述
7:3	-	R	保留
2:0	FRAMEH	R	帧号高字节位。 该寄存器包含最后接收帧号的位 10-8。

USB 端点索引寄存器 (USB_INDEX)

偏移地址：0x0E
复位值：0x00

每个 IN 端点和 OUT 端点都有一组它们自己的控制/状态寄存器。同一时刻只有一个 IN 端点的控制/状态寄存器和一个 OUT 端点的控制/状态寄存器能出现在存储映射上。在访问一个端点的控制/状态寄存器之前，需要将相应的端点号写入 USB_INDEX 寄存器。

表 22.13: USB 端点索引寄存器 (USB_INDEX) 位描述

位	符号	访问	描述
7:4	-	R	保留
3:0	EPSEL	RW	端点选择。 当访问需要索引的 USB 寄存器 (10h-17h) 时，这些位选择目标端点。 0000：端点 0 0001：端点 1 0010：端点 2 0011：端点 3 0100-1111：保留。

USB 端点 FIFO 访问寄存器 (USB_FIFO*x*)

偏移地址：0x20 + 0x04 × *x*

对每个端点 FIFO 的访问是通过对应的 FIFO*x* 寄存器来实现的。对一个端点 FIFO*x* 寄存器的读操作将数据从相应端点的 OUT FIFO 中卸载；对一个端点 FIFO*x* 寄存器的写操作将数据加载到该端点的 IN FIFO 中。

表 22.14: USB 端点 FIFO 访问寄存器 (USB_FIFO*x*) 位描述

位	符号	访问	描述
7:0	FIFODATA	RW	端点 FIFO 访问。

22.14.3 索引寄存器

USB 输入端点最大包长配置寄存器 (USB_INMAXP)

偏移地址：0x10

复位值：0x00

INMAXP 是一个 8 位的寄存器，用于设置当前选择的 IN 端点传输事务的最大数据包长 — 以 8 字节为单位，除了 128 这个值将最大数据包长设置为 1023（在 USB 全速操作中同步传输的最大包长）而不是 1024。在设置此值时，您应该注意 USB 规范对全速操作中批量，中断和同步传输的数据包大小的限制。

每个 IN 端点都有一个 INMAXP 寄存器（端点 0 除外）。

写入该寄存器的值应与对应端点的标准端点描述符的 *wMaxPacketSize* 字段匹配（参见 *Universal Serial Bus Specification Revision 2.0, Chapter 9*）。不匹配可能会导致意外结果。

如果将大于该端点 IN FIFO 大小的值写入该寄存器，则该值将自动更改为 IN FIFO 大小。如果写入该寄存器的值小于或等于 IN FIFO 大小的一半，则可以有两个 IN 数据包缓冲在 IN FIFO 中。

如果在从该端点发送数据包后更改此寄存器，则应在将新值写入该寄存器后完全刷新端点 IN FIFO（使用 INCSR1 中的 FlushFIFO 位（D3））。

表 22.15: USB 输入端点最大包长配置寄存器 (USB_INMAXP) 位描述

位	符号	访问	描述
7:0	INMAXP	RW	IN 端点传输事务的最大数据包长。

USB 端点 0 控制/状态寄存器 (USB_CSR0)

偏移地址：0x11

复位值：0x00

CSR0 寄存器提供了端点 0 的控制和状态位。

表 22.16: USB 端点 0 控制/状态寄存器 (USB_CSR0) 位描述

位	符号	访问	描述
7	SVDSETUPEND	S	软件向该位写 1 可清除 SETUPEND。该位被硬件自动清 0。
6	SVDOUTPKTRDY	S	软件向该位写 1 可清除 OUTPKTRDY 位。该位被硬件自动清 0。
5	SENDSTALL	S	软件可以通过向该位写 1 来结束当前的数据传输（因为错误条件、不希望的传输请求等）。当 STALL 信号被发送后，硬件将该位清 0。
4	SETUPEND	R	当一次控制传输在软件设置 DATAEND 位之前结束时，硬件将该只读位置 1。软件向 SVDSETUPEND 写 1 后，硬件将该位清 0。
3	DATAEND	S	软件应在下列情形中向该位写 1： 1. 为发送最后一个数据包，软件向 INPKTRDY 写 1 时。 2. 当接收完最后一个数据包后，软件向 SVDOUTPKTRDY 写 1 时。 3. 为发送一个零长度数据包，软件向 INPKTRDY 写 1 时。 该位被硬件自动清 0。
2	SENTSTALL	R_W0	当 STALL 握手信号发出后，硬件将该位置 1。该位必须用软件清 0。
1	INPKTRDY	RS	软件应在将一个要发送的数据包装入端点 0 FIFO 后向该位写 1。当数据包发送出去之后，硬件将该位清 0 并产生中断。
0	OUTPKTRDY	R	当收到一个数据包时，硬件将该位置 1 并产生中断。软件向 SVDOUTPKTRDY 位写 1 时该位被清 0。

USB 输入端点控制/状态寄存器 1(USB_INCSR1)

偏移地址：0x11

复位值：0x00

表 22.17: USB 输入端点控制/状态寄存器 1(USB_INCSR1) 位描述

位	符号	访问	描述
7	-	R	保留，始终读为 0。
6	CLRDATATOG	S	软件向该位写 1 将 IN 端点的 Data toggle 复位为 0。
5	SENTSTALL	R_W0	当 STALL 握手信号被发送后，硬件将该位置 1。FIFO 被清空，INPKTRDY 位被清 0。该标志必须用软件清 0。
4	SENDSTALL	RW	软件应向该位写 1 以产生 STALL 信号作为对一个 IN 令牌的应答。软件应向该位写 0 以结束 STALL 信号。该位对 ISO 方式没有影响。
3	FLUSHFIFO	S	向该位写 1 将从 IN 端点 FIFO 中清除待发送的下一个数据包。FIFO 指针被复位，INPKTRDY 位被清 0。如果 FIFO 中包含两个数据包，软件必须设置两次 FLUSHFIFO 位，以完全清空 FIFO。当 FIFO 清空完成后，硬件将该位清 0。
2	UNDERRUN	R_W0	该位的功能取决于 IN 端点的方式： ISO：在 INPKTRDY=0 并且收到一个 IN 令牌后发送了一个零长度数据包时，该位被置 1。 中断/批量：在对 IN 令牌回应时返回 NAK，则该位被置 1。 该位必须用软件清 0。
1	FIFONE	R_W0	FIFO 非空。 0：IN 端点 FIFO 为空。 1：IN 端点 FIFO 包含一个或多个数据包。
0	INPKTRDY	RS	软件应在将一个要发送的数据包装入 FIFO 后向该位写 1。当数据包发送出去之后，硬件将该位清 0 并产生中断 (如果允许)。

USB 输入端点控制/状态寄存器 2(USB_INCSR2)

偏移地址：0x12
复位值：0x20

表 22.18: USB 输入端点控制/状态寄存器 2(USB_INCSR2) 位描述

位	符号	访问	描述
7	AUTOSET	RW	如果该位设置为 1，那么当达到最大包长 (INMAXP) 的数据加载到 IN FIFO 中时，INPKTRDY 位被硬件自动置位。如果小于最大包长的数据被加载到 IN FIFO 中时，需要软件来设置 INPKTRDY 位。
6	ISO	RW	该位使能/禁止在当前端点进行同步传输 0：端点被配置为批量/中断传输。 1：端点被配置为同步传输。
5	-	R	保留。
4	-	R	保留，必须为 0。
3	FRCDATATOG	RW	强制数据切换位。 0：端点数据切换只在发送完一个数据包后收到 ACK 时切换。 1：端点数据切换在每发送完一个数据包后被强制切换，不管是否收到 ACK。
2:0	-	R	保留，始终读为 0。

USB 输出端点最大包长配置寄存器 (USB_OUTMAXP)

偏移地址：0x13

复位值：0x00

OUTMAXP 是一个 8 位的寄存器，用于设置当前选择的 OUT 端点传输事务的最大数据包长—以 8 字节为单位，除了 128 这个值将最大数据包长设置为 1023（在 USB 全速操作中同步传输的最大包长）而不是 1024。在设置此值时，您应该注意 USB 规范对全速操作中批量，中断和同步传输的数据包大小的限制。

每个 OUT 端点都有一个 OUTMAXP 寄存器（端点 0 除外）。

写入该寄存器的值应与对应端点的标准端点描述符的 *wMaxPacketSize* 字段匹配（参见 *Universal Serial Bus Specification Revision 2.0, Chapter 9*）。不匹配可能会导致意外结果。

写入该寄存器的值表示的数据总量不得超过 OUT 端点的 FIFO 大小，如果需要双缓冲，则不应超过 FIFO 大小的一半。如果将大于端点 OUT FIFO 大小的值写入该寄存器，则该值将自动更改为 OUT FIFO 大小。如果写入该寄存器的值小于或等于 OUT FIFO 大小的一半，则可以缓冲两个 OUT 数据包。

表 22.19: USB 输出端点最大包长配置寄存器 (USB_OUTMAXP) 位描述

位	符号	访问	描述
7:0	OUTMAXP	RW	OUT 端点传输事务的最大数据包长。

USB 输出端点控制/状态寄存器 1(USB_OUTCSR1)

偏移地址：0x14

复位值：0x00

表 22.20: USB 输出端点控制/状态寄存器 1(USB_OUTCSR1) 位描述

位	符号	访问	描述
7	CLRDATATOG	S	向该位写 1 将 OUT 端点的 Data toggle 复位为 0。
6	SENTSTALL	R_W0	当 STALL 握手信号被发送后，硬件将该位置 1。该标志必须用软件清 0。
5	SENDSTALL	RW	向该位写 1 以产生 STALL 握手信号。向该位写 0 以结束 STALL 信号。该位对 ISO 方式没有影响。
4	FLUSHFIFO	S	向该位写 1 将从 OUT 端点 FIFO 中清除下一个数据包。FIFO 指针被复位。OUTPKTRDY 位被清 0。如果 FIFO 包含两个数据包，则需要设置两次 FLUSHFIFO 位以完全清空 FIFO。
3	DATAERROR	R	如果数据包有 CRC 或位填充错误，则在设置 OUTPKTRDY 时设置此位。当软件清除 OUTPKTRDY 时，该位被清 0。该位仅在 ISO 模式下有效。
2	OVERRUN	R_W0	如果 OUT 数据包无法加载到 OUT FIFO，则该位置 1。该位仅在 ISO 模式下有效。该位必须用软件清 0。
1	FIFOFULL	R	当 OUT FIFO 中无法再放入一个数据包时，该位置 1。
0	OUTPKTRDY	R_W0	当收到一个数据包时，硬件将该位置 1 并产生中断。软件应在将每个数据包从 OUT 端点 FIFO 读出后将该位清 0。

USB 输出端点控制/状态寄存器 2(USB_OUTCSR2)

偏移地址：0x15

复位值：0x00

表 22.21: USB 输出端点控制/状态寄存器 2(USB_OUTCSR2) 位描述

位	符号	访问	描述
7	AUTOCLR	RW	如果该位设置为 1，那么当从 OUT FIFO 中卸载最大数据包大小 (OUTMAXP) 的数据包时，OUTPKTRDY 位被硬件自动清 0。当卸载小于最大数据包大小的数据包时，OUTPKTRDY 位必须由软件清 0。
6	ISO	RW	该位使能/禁止在当前端点进行同步传输 0：端点被配置为批量/中断传输。 1：端点被配置为同步传输。
5:4	-	R	保留，必须为 0。
3:0	-	R	保留，始终读为 0。

USB 端点 0 数据计数寄存器 (USB_COUNT0)

偏移地址：0x16 （INDEX 寄存器设置为 0 时）
复位值：0x00

表 22.22: USB 端点 0 数据计数寄存器 (USB_COUNT0) 位描述

位	符号	访问	描述
7	-	R	保留。
6:0	COUNT0	R	端点 0 OUT FIFO 中接收的数据字节数。该值只在 OUTPKTRDY(CSR0[0]) 位为 1 期间有效。

USB 输出端点数据计数低字节寄存器 (USB_OUTCOUNTL)

偏移地址：0x16
复位值：0x00

表 22.23: USB 输出端点数据计数低字节寄存器 (USB_OUTCOUNTL) 位描述

位	符号	访问	描述
7:0	OUTCOUNTL	R	当前 OUT 端点 FIFO 中最后一个接收数据包之 10 位数据字节数的低 8 位。该数值只在 OUTPKTRDY=1 时有效。

USB 输出端点数据计数高字节寄存器 (USB_OUTCOUNTH)

偏移地址：0x17
复位值：0x00

表 22.24: USB 输出端点数据计数高字节寄存器 (USB_OUTCOUNTH) 位描述

位	符号	访问	描述
7:3	-	R	保留。
2:0	OUTCOUNTH	R	当前 OUT 端点 FIFO 中最后一个接收数据包之 10 位数据字节数的高 3 位。 该数值只在 OUTPKTRDY=1 时有效。

第二十三章　串行外设接口 (SPI)

23.1　简介

串行外设接口允许芯片与外部设备以半/全双工，同步，串行方式通信。此接口可以被配置成主模式或从模式来与外部设备通信。

本芯片内部具有 2 个独立的从模式 SPI 模块和 1 个主模式 SPI 模块。

23.2　SPI 特性

各独立 SPI 模块均具有以下特性：

- 32 位 APB 总线接口
- 支持 DMA 传输
- 2 个深度为 4，宽度为 16 位的数据缓存，分别对应发送和接收
- 一个中断输出信号
- 4 种可配置的时钟模式
- 数据格式为右对齐
- 支持 Motorola SPI 协议
- 支持 Texas Instruments SSP 协议
- 支持 National Semiconductor Microwire 协议

图 23.1: SPI 结构框图

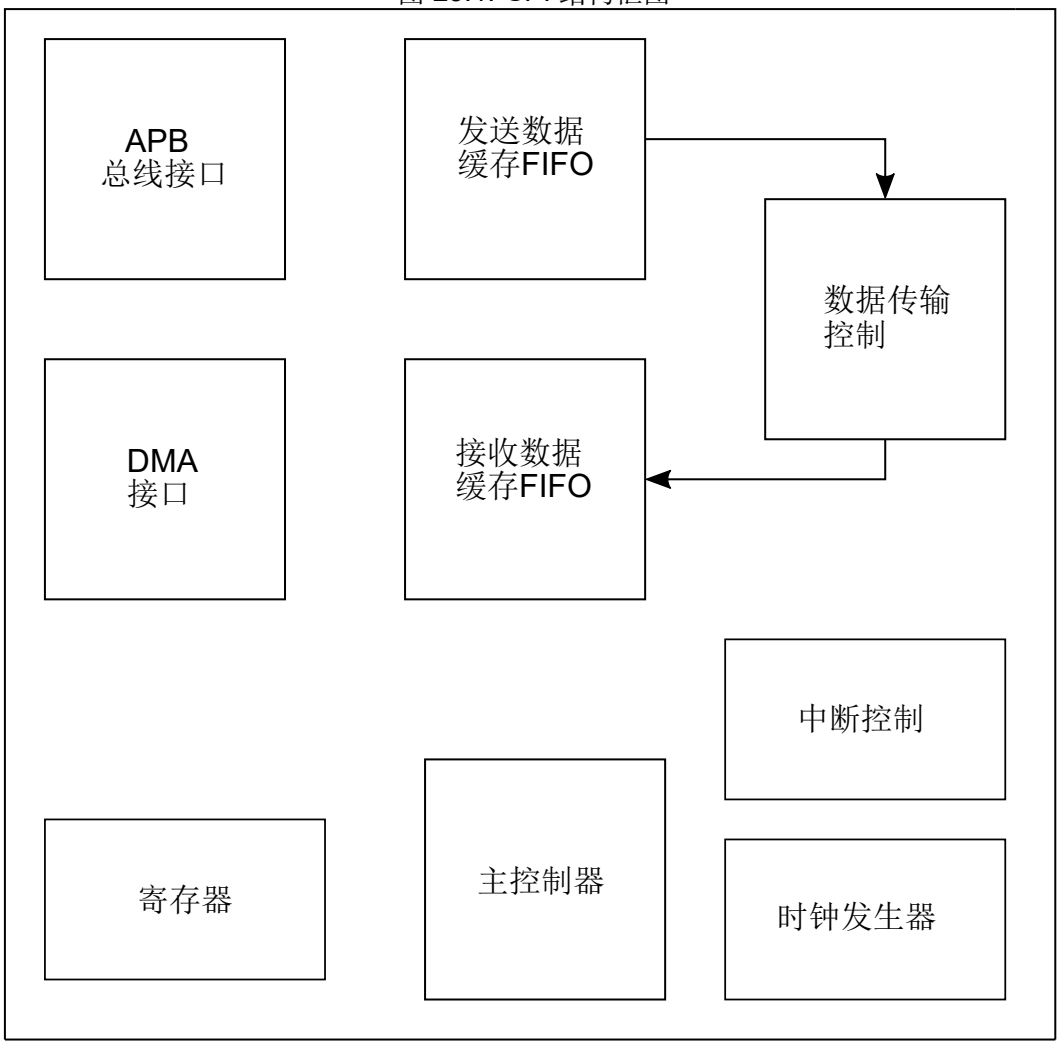
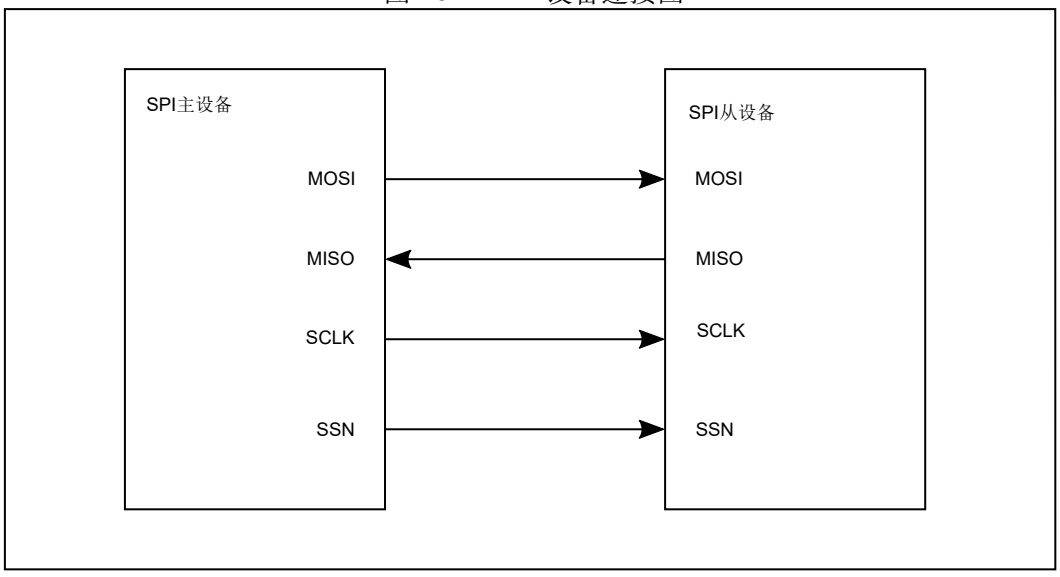


图 23.2: SPI 设备连接图



23.3　时钟模式

支持共 4 种时钟模式：

- CPOL,CPHA = 00
- CPOL,CPHA = 01
- CPOL,CPHA = 10
- CPOL,CPHA = 11

SPI_CR0 寄存器的 CPOL 和 CPHA 位，能组合成 4 种可能的时序关系。

CPOL 位控制时钟在空闲状态的电平，此位对主模式和从模式设备都有效。若 CPOL=0，SCLK 在空闲状态保持低电平；若 CPOL=1，SCLK 在空闲状态保持高电平。

CPHA 位控制时钟的相位。若 CPHA=0，时钟沿出现在数据位的中间，当数据开始传输时，SCLK 的第一个边沿用作数据采样；若 CPHA=1，时钟沿出现在数据位开头，当数据开始传输时，SCLK 的第二个边沿用作数据采样。

图 23.3: SPI 时钟模式：CPHA = 0

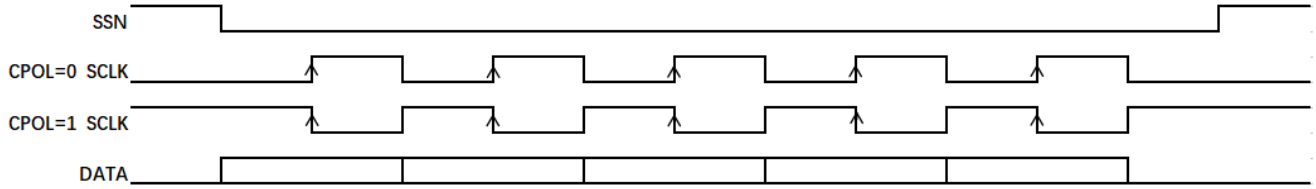
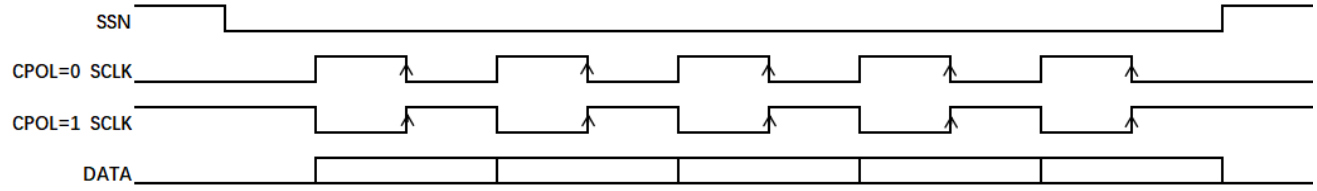


图 23.4: SPI 时钟模式：CPHA = 1



23.4　中断

各 SPI 模块都有 5 个中断源，但是只有一个中断信号输出，当 CPU 收到中断时需要读取 SPI 中断寄存器来判定中断源。

中断源:

- 发送缓存空: 当发送缓存中待发送数据等于或小于设定的阈值时产生
- 发送缓存上溢出 (overflow): 当发送缓存已满并且有新数据试图从 APB 总线写入时产生

- 接收缓存满: 当接收缓存中存储的数据等于或大于设定的阈值时产生
- 接收缓存上溢出 (overflow): 当接收缓存已满并且又接收到了新数据时产生
- 接收缓存下溢出 (underflow): 当接收缓存已空并且试图从 APB 总线读取数据时产生

缓存阈值:

可以为发送缓存和接收缓存设定阈值来更好的控制数据量，如果不设定阈值，那么默认的阈值就是发送缓存和接收缓存的最大容量。

23.5　数据传输

数据传输分为发送和接收两种，由寄存器 SPI_CR0 的 TMOD 位来控制。

- TMOD = 00: 发送和接收都有效
- TMOD = 01: 只有发送有效，接收端将不会把数据存入接收缓存，需要屏蔽掉关于接收的中断
- TMOD = 10: 只有接收有效，发送端将不会从发送缓存提取数据，需要屏蔽掉关于发送的中断
- TOMD = 11: 不允许设置为 11

23.6　DMA 传输

发送和接收都支持 DMA 传输，用户可以为发送和接收单独设定 DMA 阈值来触发 DMA 传输。

DMA 发送阈值：通过寄存器 SPI_DMATDLR 来设定，当发送缓存里的数据量等于或小于 DMA 发送阈值时，将产生一个 DMA 请求到总线上的 DMA 控制器。

DMA 接收阈值：通过寄存器 SPI_DMARDLR 来设定，当接收缓存里的数据量等于或大于 DMA 接收阈值时，将产生一个 DMA 请求到总线上的 DMA 控制器。

23.7　SPI 从模式

本芯片内部有 2 个独立的从模式 SPI 模块: SPIS1，SPIS2。

SPIS1 位于 APB1 总线，SPIS2 位于 APB2 总线。

从模式模块用于跟芯片外部主模式 SPI 设备通信。在主模式设备发送时钟之前必须配置好从模式模块，并使从模式模块的数据缓存就绪，否则有可能发生意外的数据传输错误。

SCLK 时钟由主模式设备控制，从模式设备的时钟模式 (CPOL,CPHA) 必须跟主模式设备相同。

1. SPIS1 引脚配置，若使用 SPIS1，则需先做如下 GPIO 配置 (请参考 GPIO 寄存器章节):

- SSN 对应 GPIOA 端口 4 或者 GPIOA 端口 15，将该端口的复用模式 (Alternate Mode) 配置为 6
- SCLK 对应 GPIOA 端口 5 或者 GPIOB 端口 3，将该端口的复用模式 (Alternate Mode) 配置为 6
- MISO 对应 GPIOA 端口 6 或者 GPIOB 端口 4，将该端口的复用模式 (Alternate Mode) 配置为 6
- MOSI 对应 GPIOA 端口 7 或者 GPIOB 端口 5，将该端口的复用模式 (Alternate Mode) 配置为 6

2. SPIS2 引脚配置，若使用 SPIS2，则需先做如下 GPIO 配置：

- SSN 对应 GPIOB 端口 12 或者 GPIOC 端口 0，将该端口的复用模式 (Alternate Mode) 配置为 6
- SCLK 对应 GPIOB 端口 13 或者 GPIOC 端口 1，将该端口的复用模式 (Alternate Mode) 配置为 6
- MISO 对应 GPIOB 端口 14 或者 GPIOC 端口 2，将该端口的复用模式 (Alternate Mode) 配置为 6
- MOSI 对应 GPIOB 端口 15 或者 GPIOC 端口 3，将该端口的复用模式 (Alternate Mode) 配置为 6

从模式数据接收：

主模式设备产生 SCLK 时钟并将数据串行输入到从模式设备，从模式设备根据配置好的时钟模式在适当的时钟边沿采样数据，然后将接收到的数据存入接收缓存。

从模式数据发送：

在主模式设备开始接收之前，必须将要发送的数据写入到从模式设备的数据发送缓存内。

23.8　SPI 主模式

本芯片内部有 1 个主模式 SPI 模块，位于 APB2 总线。

主模式模块需要控制 SCLK 时钟，必须先通过寄存器 SPI_BAUDR 配置好主模式的波特率。

主模式模块可以输出 3 个 SSN 信号，对应最多 3 个外部从模式设备。

主模式引脚配置，需先做如下 GPIO 配置 (请参考 GPIO 寄存器章节)：

- SSN0 对应 GPIOB 端口 12 或者 GPIOC 端口 0，将该端口配置为输出模式 (Output Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5
- SSN1 对应 GPIOB 端口 7 或者 GPIOB 端口 11，将该端口配置为输出模式 (Output Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5
- SSN2 对应 GPIOB 端口 8 或者 GPIOC 端口 5，将该端口配置为输出模式 (Output Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5
- SCLK 对应 GPIOB 端口 13 或者 GPIOC 端口 1，将该端口配置为输出模式 (Output Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5
- MISO 对应 GPIOB 端口 14 或者 GPIOC 端口 2，将该端口配置为输入模式 (Input Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5
- MOSI 对应 GPIOB 端口 15 或者 GPIOC 端口 3，将该端口配置为输出模式 (Output Mode)，将该端口的复用模式 (Alternate Mode) 配置为 5

23.9　SPI 从模式寄存器

23.9.1　SPI 控制寄存器 0(SPI_CR0)

地址偏移量：0x000

复位值：0x0100_0007

位	符号	访问	描述
31:25	-	Res	保留
24	SSTE	RW	当 CPHA=0 时，SSTE 设定了数据帧之间 SSN 信号的状态。 SSTE=1: 在两个连续的数据帧之间，SCLK 保持为空闲状态，SSN 保持为 1 SSTE=0: SSN 信号会在两个连续的数据帧之间保持为 0
23:16	-	Res	保留
15:12	CFS	RW	控制字长度，National Semiconductor Microwire 协议专用 0000: 控制字为 1 个比特 0001: 控制字为 2 个比特 0010: 控制字为 3 个比特 ..... 1111: 控制字为 16 个比特
11	-	Res	保留
10	SLVOE	RW	从模式输出控制 0: 禁止从模式设备输出 1: 允许从模式设备输出
9:8	TMOD	RW	传输控制 00: 发送和接收都有效 01: 仅发送有效 10: 仅接收有效 11: 无效设置，不允许设置为 11
7	CPOL	RW	时钟极性 (Clock polarity) 0: 空闲状态时，SCLK 保持低电平 1: 空闲状态时，SCLK 保持高电平
6	CPHA	RW	时钟相位 (Clock phase) 0: 数据采样从第一个时钟边沿开始 1: 数据采样从第二个时钟边沿开始
5:4	FRF	RW	协议选择 00: Motorola SPI 协议 01: Texas Instruments SSP 协议 10: National Semiconductor Microwire 协议 11: 无效设置，不允许设置为 11
3:0	DFS	RW	数据帧长度 0000: 无效设置，不允许设置为 0000 0001: 无效设置，不允许设置为 0001 0010: 无效设置，不允许设置为 0010 0011: 数据帧长度为 4bit 0100: 数据帧长度为 5bit 0101: 数据帧长度为 6bit ..... 1111: 数据帧长度为 16bit

23.9.2 SPI 使能寄存器 (SPI_SPIENR)

地址偏移量：0x008
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	Res	保留
0	SPI_EN	RW	SPI 使能

23.9.3 Microwire 控制寄存器 (SPI_MWCR)

地址偏移量：0x00C
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1	MDD	RW	数据传输 0: 接收数据 1: 发送数据
0	MWMOD	RW	连续传输控制 0: 单数据传输 1: 连续传输

23.9.4 发送缓存阈值寄存器 (SPI_TXFTLR)

地址偏移量：0x018
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	TFT	RW	发送缓存阈值设定

23.9.5 接收缓存阈值寄存器 (SPI_RXFTLR)

地址偏移量：0x01C
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	RFT	RW	接收缓存阈值设定

23.9.6 发送缓存状态寄存器 (SPI_TXFLR)

地址偏移量：0x020
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	TXTFL	R	显示还有多少数据存在于发送缓存中

23.9.7　接收缓存状态寄存器 (SPI_RXFLR)

地址偏移量：0x024

复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	RXTFL	R	显示还有多少空闲空间在接收缓存中

23.9.8　SPI 状态寄存器 (SPI_SR)

地址偏移量：0x028

复位值：0x0000_0006

位	符号	访问	描述
31:6	-	Res	保留
5	TXERR	RC_R	当发送缓存已空但是却启动了发送时会置 1，读此寄存器会清除掉此位
4	RFF	R	接收缓存满时置 1，接收缓存不满时置 0
3	RFNE	R	接收缓存非空时置 1，接收缓存已空时置 0
2	TFE	R	发送缓存已空时置 1，发送缓存非空时置 0
1	TFNF	R	发送缓存未满时置 1，发送缓存已满时置 0
0	BUSY	R	传输正在进行时置 1，所有传输结束时置 0

23.9.9　中断使能寄存器 (SPI_IER)

地址偏移量：0x02C

复位值：0x0000_001F

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIE	RW	写 0 即屏蔽接收缓存满中断
3	RXOIE	RW	写 0 即屏蔽接收缓存上溢出 (overflow) 中断
2	RXUIE	RW	写 0 即屏蔽接收缓存下溢出 (underflow) 中断
1	TXOIE	RW	写 0 即屏蔽发送缓存上溢出 (overflow) 中断
0	TXEIE	RW	写 0 即屏蔽发送缓存空中断

23.9.10　中断状态寄存器 (SPI_ISR)

地址偏移量：0x030

复位值：0x0000_0000

描述：此状态寄存器会受到中断使能寄存器 (SPI_IER) 影响

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIS	R	接收缓存满中断
3	RXOIS	R	接收缓存上溢出 (overflow) 中断
2	RXUIS	R	接收缓存下溢出 (underflow) 中断
1	TXOIS	R	发送缓存上溢出 (overflow) 中断
0	TXEIS	R	发送缓存空中断

23.9.11　无屏蔽中断状态寄存器 (SPI_RISR)

地址偏移量：0x034

复位值：0x0000_0000

描述：此状态寄存器将无视中断使能寄存器 (SPI_IER)

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIR	R	接收缓存满中断
3	RXOIR	R	接收缓存上溢出 (overflow) 中断
2	RXUIR	R	接收缓存下溢出 (underflow) 中断
1	TXOIR	R	发送缓存上溢出 (overflow) 中断
0	TXEIR	R	发送缓存空中断

23.9.12　发送缓存上溢出中断清除寄存器 (SPI_TXOICR)

地址偏移量：0x038

复位值：0x0000_0000

描述：读此寄存器将清除发送缓存上溢出 (overflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	TXOICR	R	清除发送缓存上溢出 (overflow) 中断

23.9.13　接收缓存上溢出中断清除寄存器 (SPI_RXOICR)

地址偏移量：0x03C

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存上溢出 (overflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXOICR	R	清除接收缓存上溢出 (overflow) 中断

23.9.14　接收缓存下溢出中断清除寄存器 (SPI_RXUICR)

地址偏移量：0x040

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXUICR	R	清除接收缓存下溢出 (underflow) 中断

23.9.15　中断清除寄存器 (SPI_ICR)

地址偏移量：0x048

复位值：0x0000_0000

描述：读此寄存器将同时清除发送缓存上溢出 (overflow) 中断，接收缓存上溢出 (overflow) 中断，接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	ICR	R	清除中断

23.9.16　DMA 使能寄存器 (SPI_DMACR)

地址偏移量：0x04C

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1	TDMAE	RW	DMA 发送使能 0: 禁止 DMA 发送 1: 允许 DMA 发送
0	RDMAE	RW	DMA 接收使能 0: 禁止 DMA 接收 1: 允许 DMA 接收

23.9.17　DMA 发送阈值寄存器 (SPI_DMATDLR)

地址偏移量：0x050

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMATDLR	RW	DMA 发送阈值设定

23.9.18　DMA 接收阈值寄存器 (SPI_DMARDLR)

地址偏移量：0x054

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMARDL	RW	DMA 接收阈值设定

23.9.19　SPI 数据寄存器 (SPI_DR)

地址偏移量：0x060 到 0x0EC

复位值：0x0000_0000

描述：DR 寄存器是一个连续的地址空间，对此段空间写入将把数据按顺序送入发送缓存；对此段空间读取将顺序读出接收缓存中的数据。

位	符号	访问	描述
31:16	-	Res	保留
15:0	DR	RW	SPI 数据 写此寄存器将把数据存入发送缓存 读此寄存器将从接收缓存中读取数据

23.9.20　SPI 从模式寄存器图

下表列出了 SPI 从模式的寄存器映像和复位值。

表 23.20: SPI 从模式的寄存器映像表

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SPI_CR0	保留							SSTE	保留								CFS				保留	SLVOE	TMOD		CPOL	CPHA	FRF		DFS			
	复位值								1									0				0	0	0		0	0	0		7			
0x08	SPI_SPIENR	保留																															SPI_EN
	复位值																																0
0x0C	SPI_MWCR	保留																														MDD	MWMOD
	复位值																															0	0
0x18	SPI_TXFTLR	保留																															TFT
	复位值																																0
0x1C	SPI_RXFTLR	保留																															RFT
	复位值																																0
0x20	SPI_TXFLR	保留																														TFTFL	
	复位值																															0	
0x24	SPI_RXFLR	保留																														RFTFL	
	复位值																															0	
0x28	SPI_SR	保留																										TXERR	RFF	RFNE	TFE	TFNF	BUSY
	复位值																											0	0	0	1	1	0
0x2C	SPI_IER	保留																											RXFIE	RXOIE	RXUIE	TXOIE	TXEIE
	复位值																												0	0	0	0	0
0x30	SPI_ISR	保留																											RXFIS	RXOIS	RXUIS	TXOIS	TXEIS
	复位值																												0	0	0	0	0
0x34	SPI_RISR	保留																											RXFIR	RXOIR	RXUIR	TXOIR	TXEIR
	复位值																												0	0	0	0	0
0x38	SPI_TXOICR	保留																															TXOICR

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	复位值																																0
0x3C	SPI_RXOICR	保留																															RXOICR
	复位值																																0
0x40	SPI_RXUICR	保留																															RXUICR
	复位值																																0
0x48	SPI_ICR	保留																															ICR
	复位值																																0
0x4C	SPI_DMACR	保留																														TDMAE	RDMAE
	复位值																															0	0
0x50	SPI_DMATDLR	保留																															DMATDL
	复位值																																0
0x54	SPI_DMARDLR	保留																															DMARDL
	复位值																																0
0x60	SPI_DR	保留																DR															
	复位值																	0															

23.10　SPI 主模式寄存器

注: 当在 SPI_SPIENR 中使能了 SPI 以后，不能再对其他寄存器写入。

23.10.1　SPI 控制寄存器 0(SPI_CR0)

地址偏移量：0x000

复位值：0x0100_0007

位	符号	访问	描述
31:25	-	Res	保留
24	SSTE	RW	当 CPHA=0 时，SSTE 设定了数据帧之间 SSN 信号的状态。 SSTE=1: 在两个连续的数据帧之间，SCLK 保持为空闲状态，SSN 保持为 1 SSTE=0: SSN 信号会在两个连续的数据帧之间保持为 0
23:16	-	Res	保留
15:12	CFS	RW	控制字长度，National Semiconductor Microwire 协议专用 0000: 控制字为 1 个比特 0001: 控制字为 2 个比特 0010: 控制字为 3 个比特 ..... 1111: 控制字为 16 个比特
11:10	-	Res	保留
9:8	TMOD	RW	传输控制 00: 发送和接收都有效 01: 仅发送有效 10: 仅接收有效 11: 无效设置，不允许设置为 11

7	CPOL	RW	时钟极性 (Clock polarity) 0: 空闲状态时，SCLK 保持低电平 1: 空闲状态时，SCLK 保持高电平
6	CPHA	RW	时钟相位 (Clock phase) 0: 数据采样从第一个时钟边沿开始 1: 数据采样从第二个时钟边沿开始
5:4	FRF	RW	协议选择 00: Motorola SPI 协议 01: Texas Instruments SSP 协议 10: National Semiconductor Microwire 协议 11: 无效设置，不允许设置为 11
3:0	DFS	RW	数据帧长度 0000: 无效设置，不允许设置为 0000 0001: 无效设置，不允许设置为 0001 0010: 无效设置，不允许设置为 0010 0011: 数据帧长度为 4bit 0100: 数据帧长度为 5bit 0101: 数据帧长度为 6bit ..... 1111: 数据帧长度为 16bit

23.10.2　SPI 控制寄存器 1(SPI_CR1)

地址偏移量：0x004

复位值：0x0000_0000

描述：此寄存器仅在 TMOD=10 时起作用，用于设置连续接收的最大数据量。

位	符号	访问	描述
31:16	-	Res	保留
15:0	NDF	RW	连续接收的最大数据量

23.10.3　SPI 使能寄存器 (SPI_SPIENR)

地址偏移量：0x008

复位值：0x0000_0000

位	符号	访问	描述
31:1	-	Res	保留
0	SPI_EN	RW	SPI 使能

23.10.4　Microwire 控制寄存器 (SPI_MWCR)

地址偏移量：0x00C

复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2	MHS	RW	握手控制 0: 禁止握手 1: 允许握手
1	MDD	RW	数据传输 0: 接收数据 1: 发送数据
0	MWMOD	RW	连续传输控制 0: 单数据传输 1: 连续传输

23.10.5　从设备选择寄存器 (SPI_SER)

地址偏移量：0x010

复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2	SE2	RW	从设备 2 选择 0: 未选中 1: 选中该从设备
1	SE1	RW	从设备 1 选择 0: 未选中 1: 选中该从设备
0	SE0	RW	从设备 0 选择 0: 未选中 1: 选中该从设备

23.10.6　波特率寄存器 (SPI_BAUDR)

地址偏移量：0x014

复位值：0x0000_0000

描述：主设备输出的 SCLK 时钟是由 SPI 主设备输入时钟 (SPI_CLK) 分频得到。

位	符号	访问	描述
31:16	-	Res	保留
15:0	SCKDV	RW	分频比，按如下公式计算输出 SCLK 时钟的频率 $F_{sclk} = F_{spi\_clk} / SCKDV$

23.10.7　发送缓存阈值寄存器 (SPI_TXFTLR)

地址偏移量：0x018

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	TFT	RW	发送缓存阈值设定

23.10.8　接收缓存阈值寄存器 (SPI_RXFTLR)

地址偏移量：0x01C
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	RFT	RW	接收缓存阈值设定

23.10.9　发送缓存状态寄存器 (SPI_TXFLR)

地址偏移量：0x020
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	TXTFL	R	显示还有多少数据存在于发送缓存中

23.10.10　接收缓存状态寄存器 (SPI_RXFLR)

地址偏移量：0x024
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	RXTFL	R	显示还有多少空闲空间在接收缓存中

23.10.11　SPI 状态寄存器 (SPI_SR)

地址偏移量：0x028
复位值：0x0000_0006

位	符号	访问	描述
31:6	-	Res	保留
5	TXERR	RC_R	当发送缓存已空但是却启动了发送时会置 1，读此寄存器会清除掉此位
4	RFF	R	接收缓存满时置 1，接收缓存不满时置 0
3	RFNE	R	接收缓存非空时置 1，接收缓存已空时置 0
2	TFE	R	发送缓存已空时置 1，发送缓存非空时置 0
1	TFNF	R	发送缓存未满时置 1，发送缓存已满时置 0

0	BUSY	R	传输正在进行时置 1，所有传输结束时置 0

23.10.12　中断使能寄存器 (SPI_IER)

地址偏移量：0x02C

复位值：0x0000_001F

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIE	RW	写 0 即屏蔽接收缓存满中断
3	RXOIE	RW	写 0 即屏蔽接收缓存上溢出 (overflow) 中断
2	RXUIE	RW	写 0 即屏蔽接收缓存下溢出 (underflow) 中断
1	TXOIE	RW	写 0 即屏蔽发送缓存上溢出 (overflow) 中断
0	TXEIE	RW	写 0 即屏蔽发送缓存空中断

23.10.13　中断状态寄存器 (SPI_ISR)

地址偏移量：0x030

复位值：0x0000_0000

描述：此状态寄存器会受到中断使能寄存器 (SPI_IER) 影响

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIS	R	接收缓存满中断
3	RXOIS	R	接收缓存上溢出 (overflow) 中断
2	RXUIS	R	接收缓存下溢出 (underflow) 中断
1	TXOIS	R	发送缓存上溢出 (overflow) 中断
0	TXEIS	R	发送缓存空中断

23.10.14　无屏蔽中断状态寄存器 (SPI_RISR)

地址偏移量：0x034

复位值：0x0000_0000

描述：此状态寄存器将无视中断使能寄存器 (SPI_IER)

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIR	R	接收缓存满中断
3	RXOIR	R	接收缓存上溢出 (overflow) 中断
2	RXUIR	R	接收缓存下溢出 (underflow) 中断
1	TXOIR	R	发送缓存上溢出 (overflow) 中断
0	TXEIR	R	发送缓存空中断

23.10.15　发送缓存上溢出中断清除寄存器 (SPI_TXOICR)

地址偏移量：0x038

复位值：0x0000_0000

描述：读此寄存器将清除发送缓存上溢出 (overflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	TXOICR	R	清除发送缓存上溢出 (overflow) 中断

23.10.16　接收缓存上溢出中断清除寄存器 (SPI_RXOICR)

地址偏移量：0x03C

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存上溢出 (overflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXOICR	R	清除接收缓存上溢出 (overflow) 中断

23.10.17　接收缓存下溢出中断清除寄存器 (SPI_RXUICR)

地址偏移量：0x040

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXUICR	R	清除接收缓存下溢出 (underflow) 中断

23.10.18　中断清除寄存器 (SPI_ICR)

地址偏移量：0x048

复位值：0x0000_0000

描述：读此寄存器将同时清除发送缓存上溢出 (overflow) 中断，接收缓存上溢出 (overflow) 中断，接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	ICR	R	清除中断

23.10.19　DMA 使能寄存器 (SPI_DMACR)

地址偏移量：0x04C

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1	TDMAE	RW	DMA 发送使能 0: 禁止 DMA 发送 1: 允许 DMA 发送
0	RDMAE	RW	DMA 接收使能 0: 禁止 DMA 接收 1: 允许 DMA 接收

23.10.20　DMA 发送阈值寄存器 (SPI_DMATDLR)

地址偏移量：0x050

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMATDL	RW	DMA 发送阈值设定

23.10.21　DMA 接收阈值寄存器 (SPI_DMARDLR)

地址偏移量：0x054

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMARDL	RW	DMA 接收阈值设定

23.10.22　SPI 数据寄存器 (SPI_DR)

地址偏移量：0x060 到 0x0EC

复位值：0x0000_0000

描述：DR 寄存器是一个连续的地址空间，对此段空间写入将把数据按顺序送入发送缓存；对此段空间读取将顺序读出接收缓存中的数据。

位	符号	访问	描述
31:16	-	Res	保留
15:0	DR	RW	SPI 数据 写此寄存器将把数据存入发送缓存 读此寄存器将从接收缓存中读取数据

第二十四章　4 线串行外设接口 (QSPI)

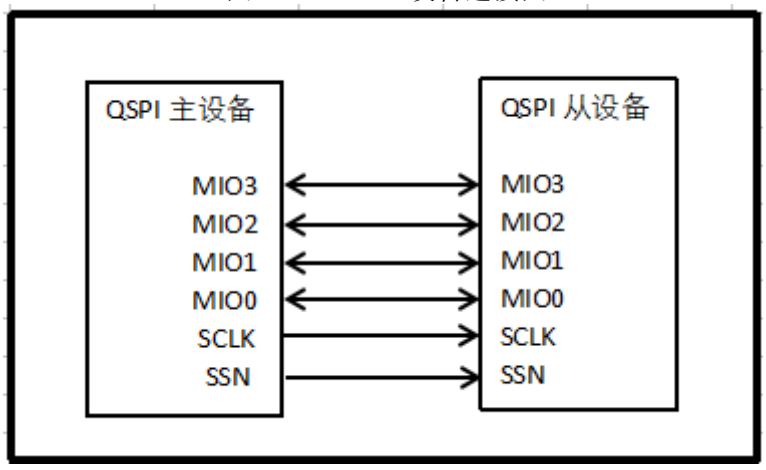
24.1　简介

QSPI 是具有 4 条数据线的 SPI 主模式设备，这是一种特殊的 SPI 协议，同时也支持标准 SPI 协议。

24.2　QSPI 特性

- 32 位 APB 总线接口
- 支持 DMA 传输
- 2 个深度为 10，宽度为 16 位的数据缓存，分别对应发送和接收
- 一个中断输出信号
- 4 种可配置的时钟模式
- 数据格式为右对齐
- 支持最多 3 个 4 线 SPI 从模式设备
- 支持标准 SPI 协议 (请参考“串行外设接口 SPI”章节)

图 24.1: QSPI 设备连接图



24.3　时钟模式

支持共 4 种时钟模式，请参考 SPI 章节。

- CPOL,CPHA = 00
- CPOL,CPHA = 01
- CPOL,CPHA = 10
- CPOL,CPHA = 11

24.4　中断

QSPI 有 5 个中断源，但是只有一个中断信号输出，当 CPU 收到中断时需要读取 QSPI 中断寄存器来判定中断源。

中断源:

- 发送缓存空: 当发送缓存中待发送数据等于或小于设定的阈值时产生
- 发送缓存上溢出 (overflow): 当发送缓存已满并且有新数据试图从 APB 总线写入时产生
- 接收缓存满: 当接收缓存中存储的数据等于或大于设定的阈值时产生
- 接收缓存上溢出 (overflow): 当接收缓存已满并且又接收到了新数据时产生
- 接收缓存下溢出 (underflow): 当接收缓存已空并且试图从 APB 总线读取数据时产生

缓存阈值:

可以为发送缓存和接收缓存设定阈值来更好的控制数据量，如果不设定阈值，那么默认的阈值就是发送缓存和接收缓存的最大容量。

24.5　数据传输

数据传输分为发送和接收两种，由寄存器 QSPI_CR0 的 TMOD 位来控制。

- TMOD = 00: 发送和接收都有效
- TMOD = 01: 只有发送有效，接收端将不会把数据存入接收缓存，需要屏蔽掉关于接收的中断
- TMOD = 10: 只有接收有效，发送端将不会从发送缓存提取数据，需要屏蔽掉关于发送的中断
- TOMD = 11: 不允许设置为 11

24.6　DMA 传输

发送和接收都支持 DMA 传输，用户可以为发送和接收单独设定 DMA 阈值来触发 DMA 传输。

DMA 发送阈值：通过寄存器 QSPI_DMATDLR 来设定，当发送缓存里的数据量等于或小于 DMA 发送阈值时，将产生一个 DMA 请求到总线上的 DMA 控制器。

DMA 接收阈值：通过寄存器 QSPI_DMARDLR 来设定，当接收缓存里的数据量等于或大于 DMA 接收阈值时，将产生一个 DMA 请求到总线上的 DMA 控制器。

24.7 QSPI 主模式

QSPI 需要控制 SCLK 时钟，必须先通过寄存器 QSPI_BAUDR 配置好主模式的波特率。

QSPI 可以输出 3 个 SSN 信号，对应最多 3 个外部从模式设备。

要使用 QSPI，需先做如下 GPIO 配置 (请参考 GPIO 寄存器章节):

- SSN0 对应 GPIOA 端口 4 或者 GPIOA 端口 15，将该端口的复用模式 (Alternate Mode) 配置为 5
- SSN1 对应 GPIOA 端口 13 或者 GPIOB 端口 6，将该端口的复用模式 (Alternate Mode) 配置为 5
- SSN2 对应 GPIOA 端口 14 或者 GPIOB 端口 10，将该端口的复用模式 (Alternate Mode) 配置为 5
- SCLK 对应 GPIOA 端口 5 或者 GPIOB 端口 3，将该端口的复用模式 (Alternate Mode) 配置为 5
- MIO0 对应 GPIOA 端口 7 或者 GPIOB 端口 5，将该端口的复用模式 (Alternate Mode) 配置为 5
- MIO1 对应 GPIOA 端口 6 或者 GPIOB 端口 4，将该端口的复用模式 (Alternate Mode) 配置为 5
- MIO2 对应 GPIOB 端口 0，将该端口的复用模式 (Alternate Mode) 配置为 5
- MIO3 对应 GPIOB 端口 1，将该端口的复用模式 (Alternate Mode) 配置为 5

QSPI 主设备会先向从设备发送 QSPI 指令，接着发送要读写的地址 (指令和地址合称控制帧)，然后才会发送数据到从设备或者由从设备接收数据。

24.8 QSPI 主模式寄存器

注: 当在 QSPI_SPIENR 中使能了 QSPI 以后，不能再对其他寄存器写入。

24.8.1 QSPI 控制寄存器 0(QSPI_CR0)

地址偏移量：0x000

复位值：0x0100_0007

位	符号	访问	描述
31:25	-	Res	保留
24	SSTE	RW	当 CPHA=0 时，SSTE 设定了数据帧之间 SSN 信号的状态。 SSTE=1: 在两个连续的数据帧之间，SCLK 保持为空闲状态，SSN 保持为 1 SSTE=0: SSN 信号会在两个连续的数据帧之间保持为 0
23	-	Res	保留

22:21	SPI_MODE	RW	SPI 模式设置 00: 标准 SPI 模式 01: 双线 SPI 模式 (Dual) 10: 四线 SPI 模式 (Quad) 11: 保留
20:10	-	Res	保留
9:8	TMOD	RW	传输控制 00: 发送和接收都有效 01: 仅发送有效 10: 仅接收有效 11: 无效设置，不允许设置为 11
7	CPOL	RW	时钟极性 (Clock polarity) 0: 空闲状态时，SCLK 保持低电平 1: 空闲状态时，SCLK 保持高电平
6	CPHA	RW	时钟相位 (Clock phase) 0: 数据采样从第一个时钟边沿开始 1: 数据采样从第二个时钟边沿开始
5:4	FRF	RW	协议选择 00: Motorola SPI 协议 01: Texas Instruments SSP 协议 10: 无效设置，不允许设置为 10 11: 无效设置，不允许设置为 11
3:0	DFS	RW	数据帧长度，必须为 4 的倍数 0000: 无效设置，不允许设置为 0000 0001: 无效设置，不允许设置为 0001 0010: 无效设置，不允许设置为 0010 0011: 数据帧长度为 4bit 0100: 数据帧长度为 5bit 0101: 数据帧长度为 6bit ..... 1111: 数据帧长度为 16bit

24.8.2 QSPI 控制寄存器 1(QSPI_CR1)

地址偏移量：0x004

复位值：0x0000_0000

描述：此寄存器仅在 TMOD=10 时起作用，用于设置连续接收的最大数据量。

位	符号	访问	描述
31:16	-	Res	保留
15:0	NDF	RW	连续接收的最大数据量

24.8.3 QSPI 使能寄存器 (QSPI_SPIENR)

地址偏移量：0x008
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	Res	保留
0	SPI_EN	RW	QSPI 使能

24.8.4 从设备选择寄存器 (QSPI_SER)

地址偏移量：0x010
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2	SE2	RW	从设备 2 选择 0: 未选中 1: 选中该从设备
1	SE1	RW	从设备 1 选择 0: 未选中 1: 选中该从设备
0	SE0	RW	从设备 0 选择 0: 未选中 1: 选中该从设备

24.8.5 波特率寄存器 (QSPI_BAUDR)

地址偏移量：0x014
复位值：0x0000_0000
描述：QSPI 输出的 SCLK 时钟是由输入时钟 (QSPI_CLK) 分频得到。

位	符号	访问	描述
31:16	-	Res	保留
15:0	SCKDV	RW	分频比，按如下公式计算输出 SCLK 时钟的频率 $F_{sclk} = F_{qspi\_clk} / SCKDV$

24.8.6 发送缓存阈值寄存器 (QSPI_TXFTLR)

地址偏移量：0x018
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	TFT	RW	发送缓存阈值设定

24.8.7 接收缓存阈值寄存器 (QSPI_RXFTLR)

地址偏移量：0x01C
复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	RFT	RW	接收缓存阈值设定

24.8.8 发送缓存状态寄存器 (QSPI_TXFLR)

地址偏移量：0x020
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	TXTFL	R	显示还有多少数据存在于发送缓存中

24.8.9 接收缓存状态寄存器 (QSPI_RXFLR)

地址偏移量：0x024
复位值：0x0000_0000

位	符号	访问	描述
31:3	-	Res	保留
2:0	RXTFL	R	显示还有多少空闲空间在接收缓存中

24.8.10 QSPI 状态寄存器 (QSPI_SR)

地址偏移量：0x028
复位值：0x0000_0006

位	符号	访问	描述
31:6	-	Res	保留
5	TXERR	RC_R	当发送缓存已空但是却启动了发送时会置 1，读此寄存器会清除掉此位
4	RFF	R	接收缓存满时置 1，接收缓存不满时置 0
3	RFNE	R	接收缓存非空时置 1，接收缓存已空时置 0
2	TFE	R	发送缓存已空时置 1，发送缓存非空时置 0
1	TFNF	R	发送缓存未满时置 1，发送缓存已满时置 0
0	BUSY	R	传输正在进行时置 1，所有传输结束时置 0

24.8.11 中断使能寄存器 (QSPI_IER)

地址偏移量：0x02C
复位值：0x0000_001F

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIE	RW	写 0 即屏蔽接收缓存满中断
3	RXOIE	RW	写 0 即屏蔽接收缓存上溢出 (overflow) 中断
2	RXUIE	RW	写 0 即屏蔽接收缓存下溢出 (underflow) 中断
1	TXOIE	RW	写 0 即屏蔽发送缓存上溢出 (overflow) 中断
0	TXEIE	RW	写 0 即屏蔽发送缓存空中断

24.8.12 中断状态寄存器 (QSPI_ISR)

地址偏移量：0x030

复位值：0x0000_0000

描述：此状态寄存器会受到中断使能寄存器 (QSPI_IER) 影响

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIS	R	接收缓存满中断
3	RXOIS	R	接收缓存上溢出 (overflow) 中断
2	RXUIS	R	接收缓存下溢出 (underflow) 中断
1	TXOIS	R	发送缓存上溢出 (overflow) 中断
0	TXEIS	R	发送缓存空中断

24.8.13 无屏蔽中断状态寄存器 (QSPI_RISR)

地址偏移量：0x034

复位值：0x0000_0000

描述：此状态寄存器将无视中断使能寄存器 (QSPI_IER)

位	符号	访问	描述
31:5	-	Res	保留
4	RXFIR	R	接收缓存满中断
3	RXOIR	R	接收缓存上溢出 (overflow) 中断
2	RXUIR	R	接收缓存下溢出 (underflow) 中断
1	TXOIR	R	发送缓存上溢出 (overflow) 中断
0	TXEIR	R	发送缓存空中断

24.8.14 发送缓存上溢出中断清除寄存器 (QSPI_TXOICR)

地址偏移量：0x038

复位值：0x0000_0000

描述：读此寄存器将清除发送缓存上溢出 (overflow) 中断

位	符号	访问	描述

31:1	-	Res	保留
0	TXOICR	R	清除发送缓存上溢出 (overflow) 中断

24.8.15 接收缓存上溢出中断清除寄存器 (QSPI_RXOICR)

地址偏移量：0x03C

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存上溢出 (overflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXOICR	R	清除接收缓存上溢出 (overflow) 中断

24.8.16 接收缓存下溢出中断清除寄存器 (QSPI_RXUICR)

地址偏移量：0x040

复位值：0x0000_0000

描述：读此寄存器将清除接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	RXUICR	R	清除接收缓存下溢出 (underflow) 中断

24.8.17 中断清除寄存器 (QSPI_ICR)

地址偏移量：0x048

复位值：0x0000_0000

描述：读此寄存器将同时清除发送缓存上溢出 (overflow) 中断，接收缓存上溢出 (overflow) 中断，接收缓存下溢出 (underflow) 中断

位	符号	访问	描述
31:1	-	Res	保留
0	ICR	R	清除中断

24.8.18 DMA 使能寄存器 (QSPI_DMACR)

地址偏移量：0x04C

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1	TDMAE	RW	DMA 发送使能 0: 禁止 DMA 发送 1: 允许 DMA 发送

0	RDMAE	RW	DMA 接收使能 0: 禁止 DMA 接收 1: 允许 DMA 接收

24.8.19 DMA 发送阈值寄存器 (QSPI_DMATDLR)

地址偏移量：0x050

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMATDL	RW	DMA 发送阈值设定

24.8.20 DMA 接收阈值寄存器 (QSPI_DMARDLR)

地址偏移量：0x054

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	Res	保留
1:0	DMARDL	RW	DMA 接收阈值设定

24.8.21 QSPI 数据寄存器 (QSPI_DR)

地址偏移量：0x060 到 0x0EC

复位值：0x0000_0000

描述：DR 寄存器是一个连续的地址空间，对此段空间写入将把数据按顺序送入发送缓存；对此段空间读取将顺序读出接收缓存中的数据。

位	符号	访问	描述
31:16	-	Res	保留
15:0	DR	RW	QSPI 数据 写此寄存器将把数据存入发送缓存 读此寄存器将从接收缓存中读取数据

24.8.22 QSPI 增强模式配置寄存器 (QSPI_ESPICR)

地址偏移量：0x0F4

复位值：0x0000_0200

位	符号	访问	描述
31:16	-	Res	保留

15:11	WCYC	RW	QSPI 等待时间 接收时，QSPI 主设备发送完控制帧 (指令和地址) 以后需要一段等待时间，待从设备准备就绪后才能由从设备接收数据，等待时间以 SCLK 周期为单位
10	-	Res	保留
9:8	INSTL	RW	指令长度 00: 0 位，即表示不需要发送指令 01: 4 位指令 10: 8 位指令 11: 16 位指令
7:6	-	Res	保留
5:2	ADDRL	RW	地址长度 0000: 0 位，即表示不需要发送地址 0001: 4 位地址 0010: 8 位地址 0011: 12 位地址 0100: 16 位地址 0101: 20 位地址 0110: 24 位地址 0111: 28 位地址 1000: 32 位地址 1001: 36 位地址 1010: 40 位地址 1011: 44 位地址 1100: 48 位地址 1101: 52 位地址 1110: 56 位地址 1111: 60 位地址
1:0	TRANST	RW	控制帧传输格式 00: 指令和地址都以标准 SPI 格式 (单线 SPI) 传送 01: 指令以标准 SPI 格式 (单线 SPI) 传送，地址以 QSPI(4 线 SPI) 传送 10: 指令和地址都以 QSPI(4 线 SPI) 传送 11: 不允许设置为 11

第二十五章　集成电路内置音频总线 (I2S)

25.1　I2S 简介

WB32F10xxx 内置 1 个 I2S 总线接口，支持多种音频传输协议，工作于主控模式，支持双通道输入和输出。提供主时钟（MCLK）和串行时钟（SCLK）以及帧时钟（WS）和串行数据（SD0/SD1）。

25.2　I2S 主要特性

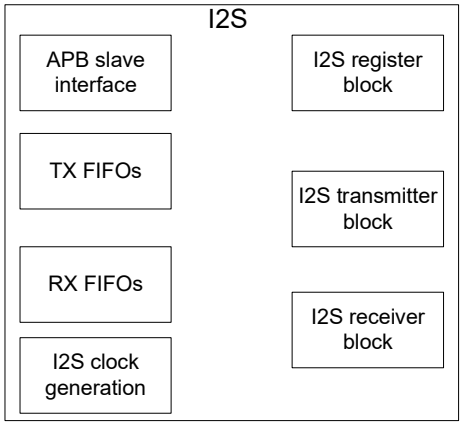
- 32 位 APB 接口
- 符合 Philips I2S 串行协议
- 除三根主要信号 sdi, sclk 和 ws 外，为了使系统间能够更好的同步，提供主时钟 mclk
- 2 个发送通道，2 个接收通道
- 支持全双工传输，因为发送器和接收器独立
- 主机工作模式，提供时钟门控和时钟使能信号
- 分辨率可配，支持 12,16,20,24,32 位
- Rx FIFO 和 Tx FIFO 深度为 4, 字大小是 32 位
- 支持 DMA 传输
- FIFO 阈值可配

25.3　I2S 功能描述

25.3.1　框图

I2C 接口如图25.1所示。

图 25.1: I2S 框图



25.3.2　I2S 传输协议

I2S 接口包括三根信号线：串行数据 sdi，串行时钟 sclk，字线 ws。为了使系统间能够更好的同步，WB32F10xxx 还提供主时钟 MCLK。

I2S 中的数据以 2 进制补码格式传输，总是从 ws 变化后的下一个 sclk 周期，开始传输 MSB。

I2S 接口总是先传输左声道数据，然后传输右声道数据。

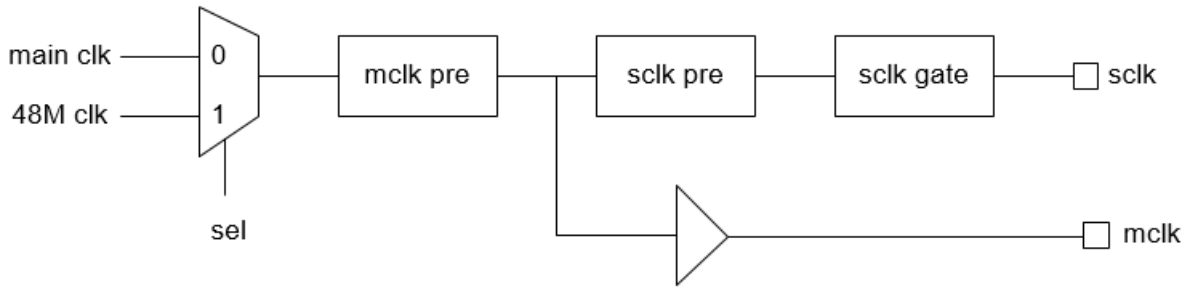
字线为低时传输左声道数据，字线为高时传输右声道数据。字线的长度可以通过 I2S_CCR.WSS 配置，支持三种配置：16，24 和 32 个 sclk 周期。

当数据分辨率小于字线长度，可以配置 I2S_CCR.SCLKG 将不用的位对应的 sclk 关闭。

25.3.3　I2S 时钟配置

I2S 的时钟如下图所示：

图 25.2: I2S 时钟框图



mclk 源可以选择 main clk 或者 48MHz 时钟，通过 RCC 模块的 MCLKSRC 配置。mclk 的分频器可以通过 RCC 模块的 MCLKPRE 配置。sclk 的分频器可以通过 RCC 模块的 I2SPRE 配置。详情请参考 RCC 模块相关章节。

当时钟发生器关闭时 (I2S_CER[0]=0), 停止产生 sclk 和 ws 信号。用户可以在时钟发生器关闭时，通过寄存器 I2S_CCR 配置 sclk 和 ws 信号。

25.3.4　I2S 接口使能

在使用 I2S 接口开始数据传输之前，需要先使能 I2S 接口。将 I2S_IER[0] 写为 1，可以使能 I2S 接口。将 I2S_IER[0] 写为 0，可以禁用 I2S 接口。

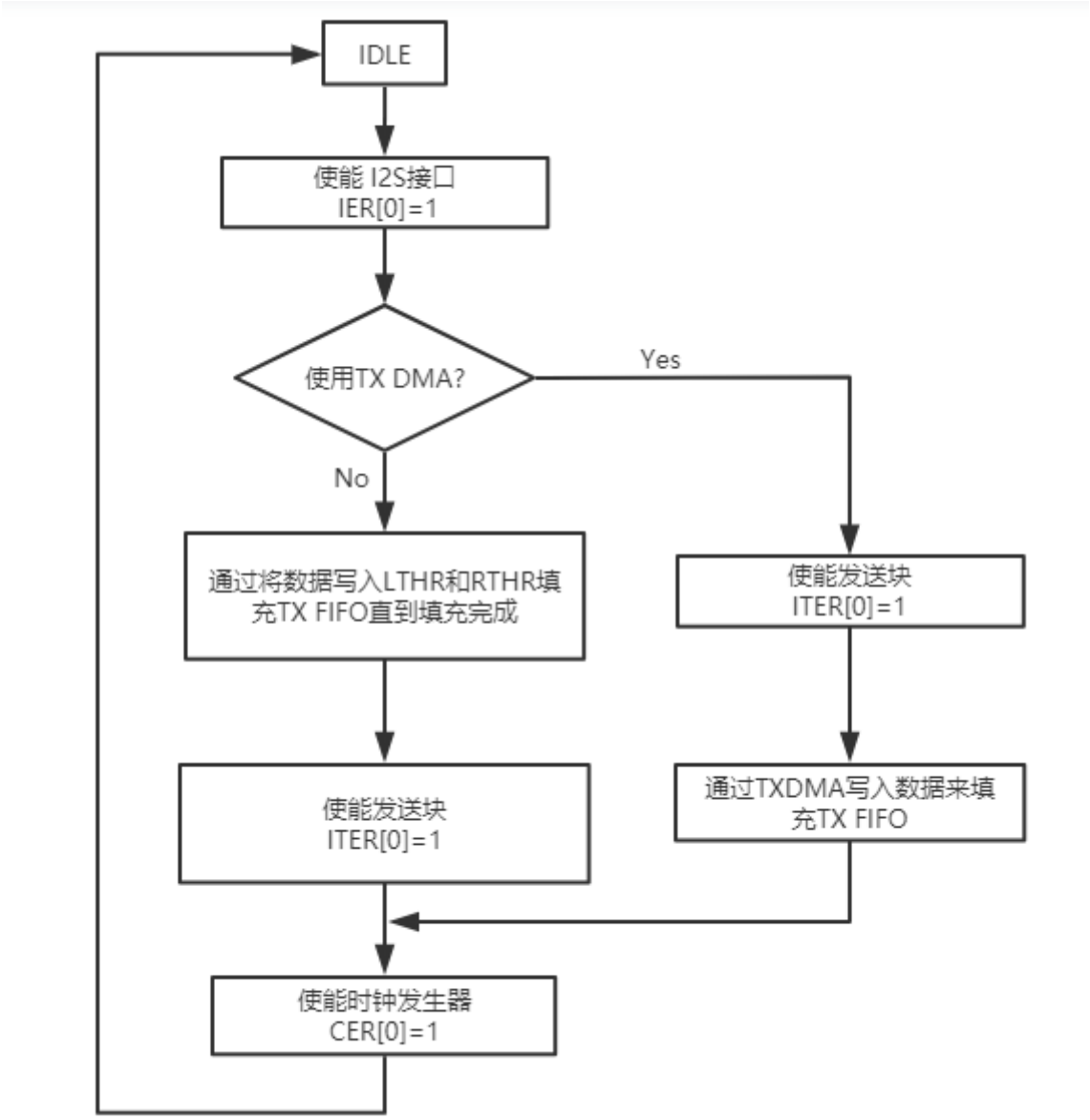
在禁用 I2S 接口之后：

- 清除 Tx FIFO 和 Rx FIFO
- 正在被传输或者接收的数据丢失
- I2S_IRER/I2S_ITER/I2S_RERx/I2S_TERx 中的使能位无效
- sclk 输出被关闭

25.3.5　发送器模式

该 I2S 接口支持 2 个发送通道，可以独立工作。下图显示了 I2S 接口做为发送器时的使用方法：

图 25.3: I2S 发送器模式基本使用流程



最后打开时钟发生器是因为发送数剧从 *ws* 信号线变化开始，在此之前需要先配置 *I2S* 接口。

发送数据

每个发送通道有两个 FIFO，分别存储左声道和右声道数据。在非 DMA 模式下，用户需要将立体声信息对的左声道数据写入 I2S_LTHRx(x=0,1)，右声道数据写入 I2S_RTHRx(x=0,1)。在 DMA 模式下，通过 I2S_TXDMA 写发送数据。需要从使能的最低通道写起，先写左声道数据，后写右声道数据。

例如：通道 0 和通道 1 同时被使能，写发送数据的顺序是：

1. Ch0 - 左声道数据
2. Ch0 - 右声道数据
3. Ch1 - 左声道数据
4. Ch1 - 右声道数据
5. Ch0 - 左声道数据
6. Ch0 - 右声道数据

.....

在写 I2S_TXDMA 的过程中如果要修改顺序，重新从最低通道写起，可以写 I2S_RTXDMA 寄存器。注意：写操作不能在左右声道对之间，否则该写操作无效。

例如：通道 0 和通道 1 同时被使能，写发送数据的顺序是：

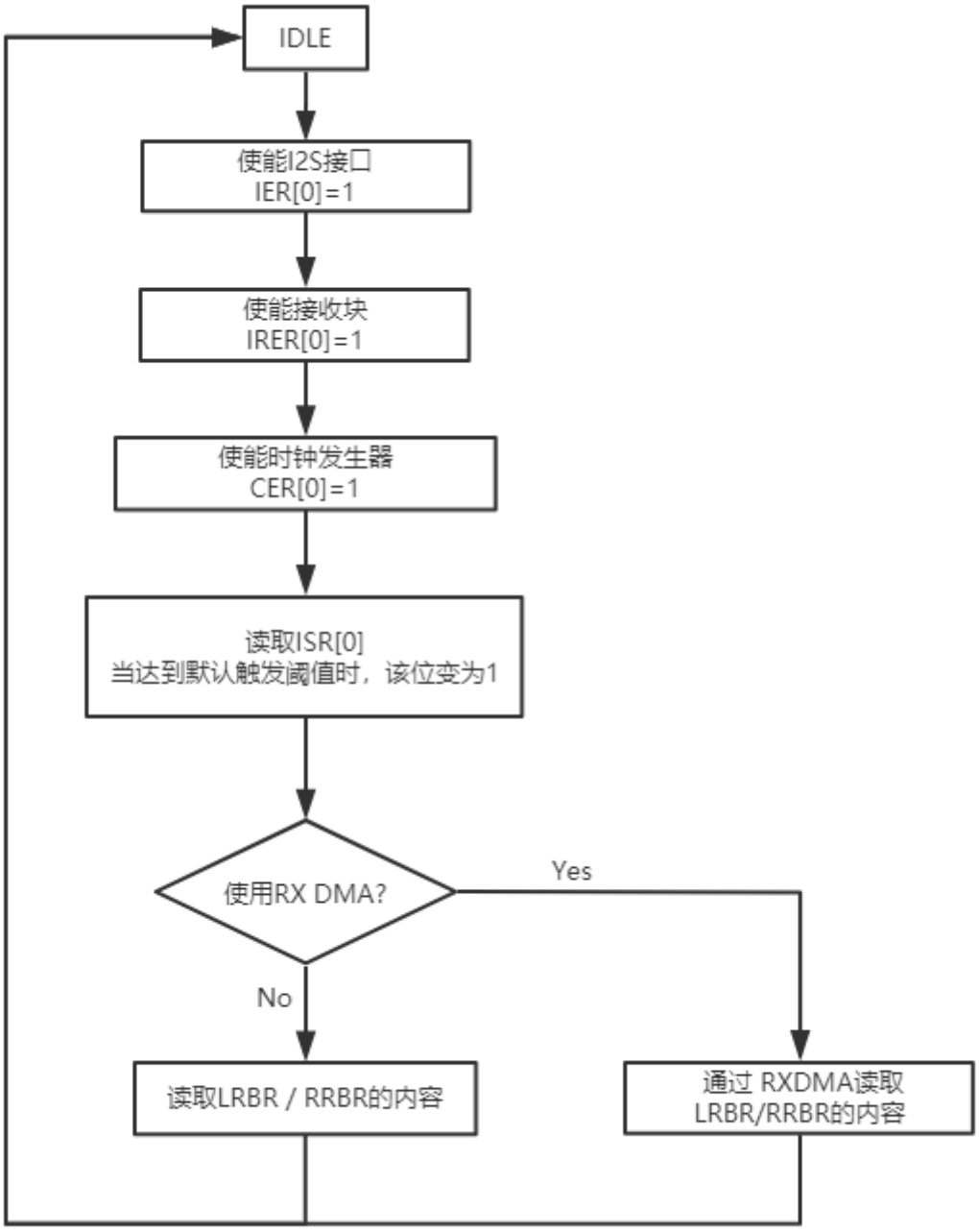
1. Ch0 - 左声道数据
2. 写 I2S_RTXDMA 复位（该写操作无效）
3. Ch0 - 右声道数据
4. Ch1 - 左声道数据
5. Ch1 - 右声道数据
6. Ch0 - 左声道数据
7. Ch0 - 右声道数据
8. 写 I2S_RTXDMA 复位（该写操作有效）
8. Ch0 - 左声道数据
9. Ch0 - 右声道数据

.....

25.3.6　接收器模式

该 I2S 接口支持 2 个接收通道，可以独立工作。下图显示了 I2S 接口做为接收器时的使用方法：

图 25.4: I2S 接收器模式基本使用流程



最后打开时钟发生器是因为数剧传输从 *ws* 信号线变化开始，在此之前需要先配置 *I2S* 接口

接收数据

每个接收通道有两个 FIFO，分别存储左声道和右声道数据。在非 DMA 模式下，用户从 I2S_LRBRx(x=0,1) 读出立体声信息对的左声道数据，从 I2S_RRBRx(x=0,1) 读出右声道数据。在 DMA 模式下，通过 I2S_RXDMA 读取接收数据。首先读到的是使能的最低通道，左声道数据在前，右声道数据在后。

例如：通道 0 和通道 1 同时被使能，读取数据的顺序是：

1. Ch0 - 左声道数据
2. Ch0 - 右声道数据

3. Ch1 - 左声道数据
4. Ch1 - 右声道数据
5. Ch0 - 左声道数据
6. Ch0 - 右声道数据

.....

在读 RXDMA 的过程中如果要修改顺序，重新从最低通道读起，可以写 I2S_RRXDMA 寄存器。注意：写操作不能在左右声道对之间，否则该写操作无效。

例如：通道 0 和通道 1 同时被使能，读取接收数据的顺序是：

1. Ch0 - 左声道数据
2. 写 I2S_RRXDMA 复位（该写操作无效）
3. Ch0 - 右声道数据
4. Ch1 - 左声道数据
5. Ch1 - 右声道数据
6. Ch0 - 左声道数据
7. Ch0 - 右声道数据
8. 写 I2S_RRXDMA 复位（该写操作有效）
8. Ch0 - 左声道数据
9. Ch0 - 右声道数据

.....

25.3.7　中断

每个接收通道有两个中断：Rx FIFO 数据有效中断和数据溢出中断。

每个发送通道有两个中断：Tx FIFO 空中断和数据溢出中断。

所有的中断源或在一起，产生统一的中断申请信号 intr 送给 CPU。

下表详细列出了所有的中断源：

表 25.1: IC 中断源

通道	中断源	中断屏蔽	中断状态	中断产生	中断清除
RX CH0	Rx FIFO 数据有效中断	IMR0[0]	ISR0[0]	Rx FIFO 中的数据大于或者等于 (RFCR0.RXCHDT)+1	通过 I2S_RXDMA 或者 I2S_LRBR/I2S_RRBR 读取 Rx FIFO 中的数据，直到 Rx FIFO 中的数据小于 (RFCR0.RXCHDT)+1
	Rx FIFO 溢出中断	IMR0[1]	ISR0[1]	当 Rx FIFO 满时，又接到新的数据需要写入 Rx FIFO 中	读 ROR0[0]

RX CH1	Rx FIFO 数据有效中断	IMR1[0]	ISR1[0]	Rx FIFO 中的数据大于或者等于(RFCR1.RXCHDT)+1	通过 I2S_RXDMA 或者 I2S_LRBR/I2S_RRBR 读取 Rx FIFO 中的数据，直到 Rx FIFO 中的数据小于(RFCR1.RXCHDT)+1
	Rx FIFO 溢出中断	IMR1[1]	ISR1[1]	当 Rx FIFO 满时，又接到新的数据需要写入 Rx FIFO 中	读 ROR1[0]
TX CH0	Tx FIFO 空中断	IMR0[4]	ISR0[4]	Tx FIFO 中的数据小于或者等于(TFCR0.TXCHET)	通过 I2S_TXDMA 或者 I2S_LTHR/I2S_RTHR 向 Tx FIFO 中写入数据，直到 Tx FIFO 中的数据大于(TFCR0.TXCHET)
	Tx FIFO 溢出中断	IMR0[5]	ISR0[5]	当 Tx FIFO 满时，用户又向 Tx FIFO 中写入新的数据	读 TOR0[0]
TX CH1	Tx FIFO 空中断	IMR1[4]	ISR1[4]	Tx FIFO 中的数据小于或者等于(TFCR1.TXCHET)	通过 I2S_TXDMA 或者 I2S_LTHR/I2S_RTHR 向 Tx FIFO 中写入数据，直到 Tx FIFO 中的数据大于(TFCR1.TXCHET)
	Tx FIFO 溢出中断	IMR1[5]	ISR1[5]	当 Tx FIFO 满时，用户又向 Tx FIFO 中写入新的数据	读 TOR1[0]

25.4　寄存器描述

25.4.1　使能寄存器 (I2S_IER)

地址偏移量：0x0

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	IEN	RW	该外设使能位，高有效。

25.4.2　接收块使能寄存器 (I2S_IRER)

地址偏移量：0x4
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	RXEN	RW	接收块使能位，高有效。

25.4.3　发送块使能寄存器 (I2S_ITER)

地址偏移量：0x8
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	TXEN	RW	发送块使能位，高有效。

25.4.4　时钟使能寄存器 (I2S_CER)

地址偏移量：0xC
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	CLKEN	RW	时钟发生器使能位，高有效。

25.4.5　时钟配置寄存器 (I2S_CCR)

地址偏移量：0x10
复位值：0x0000 0002
该寄存器仅在 I2S_CER[0] 为 0 时可以写

位	符号	访问	描述
31:5	-	R	保留
4:3	WSS	RW	配置左声道或者右声道的字节线长度，以 sclk 时钟周期为单位。 0x0: 16 sclk 周期 0x1: 24 sclk 周期 0x2: 32 sclk 周期
2:0	SCLKG	RW	配置 SCLK 时钟门控： 0x0: 时钟门控被禁用 0x1: 12 个 sclk 周期后，关掉 sclk 0x2: 16 个 sclk 周期后，关掉 sclk 0x3: 20 个 sclk 周期后，关掉 sclk 0x4: 24 个 sclk 周期后，关掉 sclk

25.4.6　接收块 FIFO 复位寄存器 (I2S_RXFFR)

地址偏移量：0x14
复位值：0x0000 0000
该寄存器仅在 I2S_IRER[0] 为 0 时可以写

位	符号	访问	描述
31:1	-	R	保留
0	RXFFR	W	向该位写 1 时，清除接收 FIFO 中的内容。

25.4.7　发送块 FIFO 复位寄存器 (I2S_TXFFR)

地址偏移量：0x18
复位值：0x0000 0000
该寄存器仅在 I2S_ITER[0] 为 0 时可以写

位	符号	访问	描述
31:1	-	R	保留
0	TXFFR	W	向该位写 1 时，清除发送 FIFO 中的内容。

25.4.8　左接收缓冲器寄存器 (I2S_LRBRx x=0,1)

地址偏移量：0x20，0x60
复位值：0x0000 0000

位	符号	访问	描述
31:0	LRBR	R	存储从 sdix 收到的左立体声数据

25.4.9　左发送维持寄存器 (I2S_LTHRx x=0,1)

地址偏移量：0x20，0x60
复位值：0x0000 0000

位	符号	访问	描述
31:0	LTHR	W	配置准备从 sdox 发送的左立体声数据

25.4.10　右接收缓冲寄存器 (I2S_RRBRx x=0,1)

地址偏移量：0x24，0x64
复位值：0x0000 0000

位	符号	访问	描述
31:0	RRBR	W	存储从 sdix 收到的右立体声数据

25.4.11　右发送维持寄存器 (I2S_RTHRx x=0,1)

地址偏移量：0x24，0x64

复位值：0x0000 0000

位	符号	访问	描述
31:0	RTHR	W	配置准备从 sdox 发送的左立体声数据

25.4.12　通道接收允许寄存器 (I2S_RERx x=0,1)

地址偏移量：0x28，0x68

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	RXCHEN	W	通道接收使能位。高有效。I2S_IER[0] 和 I2S_IRER[0] 都为 1 时，该位有效。

25.4.13　通道发送允许寄存器 (I2S_TERx x=0,1)

地址偏移量：0x2C，0x6C

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	TXCHEN	W	通道发送使能位。高有效。I2S_IER[0] 和 I2S_ITER[0] 都为 1 时，该位有效。

25.4.14　通道接收配置寄存器 (I2S_RCRx x=0,1)

地址偏移量：0x30，0x70

复位值：0x0000 0004

该寄存器仅在 I2S_RERx[0] 和 I2S_IRER[0] 都为 0 时可以写

位	符号	访问	描述
31:3	-	R	保留
2:0	WLEN	RW	配置通道接收器的分辨率： 0x0: 分辨率为 0，忽略字长 0x1: 分辨率为 12 位 0x2: 分辨率为 16 位 0x3: 分辨率为 20 位 0x4: 分辨率为 24 位 0x5: 分辨率为 32 位

25.4.15　通道发送配置寄存器 (I2S_TCRx x=0,1)

地址偏移量：0x34，0x74

复位值：0x0000 0004

该寄存器仅在 I2S_TERx[0] 和 I2S_ITER[0] 都为 0 时可以写

位	符号	访问	描述
31:3	-	R	保留
2:0	WLEN	RW	配置通道发送器的分辨率： 0x0: 分辨率为 0，忽略字长 0x1: 分辨率为 12 位 0x2: 分辨率为 16 位 0x3: 分辨率为 20 位 0x4: 分辨率为 24 位 0x5: 分辨率为 32 位

25.4.16　通道中断状态寄存器 (I2S_ISRx x=0,1)

地址偏移量：0x38，0x78

复位值：0x0000 0000

位	符号	访问	描述
31:6	-	R	保留
5	TXFO	R	发送 FIFO 溢出中断。 0x0: TX FIFO 没有溢出，写 TX FIFO 有效 0x1: TX FIFO 溢出，写 TX FIFO 无效
4	TXFE	R	发送 FIFO 空中断。 0x0: TX FIFO 没有空 0x1: TX FIFO 空
3:2	-	R	保留
1	RXFO	R	接收 FIFO 溢出中断。 0x0: RX FIFO 没有溢出，写 RX FIFO 有效 0x1: RX FIFO 溢出，写 RX FIFO 无效
0	RXDA	R	接收数据有效中断。 0x0: RX FIFO 没有空 0x1: RX FIFO 空

25.4.17　通道中断屏蔽寄存器 (I2S_IMRx x=0,1)

地址偏移量：0x3C，0x7C

复位值：0x0000 0000

位	符号	访问	描述
31:6	-	R	保留
5	TXFOM	RW	屏蔽发送 FIFO 溢出中断。高有效。
4	TXFEM	RW	屏蔽发送 FIFO 空中断。高有效。
3:2	-	R	保留
1	RXFOM	RW	屏蔽接收 FIFO 溢出中断。高有效。

0	RXDAM	RW	屏蔽接收数据有效中断。高有效。

25.4.18　通道接收超限寄存器 (I2S_RORx x=0,1)

地址偏移量：0x40，0x80
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	RXCHO	R	读取该位时，清除 RX FIFO 溢出中断。

25.4.19　通道发送超限寄存器 (I2S_TORx x=0,1)

地址偏移量：0x44，0x84
复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	TXCHO	R	读取该位时，清除 TX FIFO 溢出中断。

25.4.20　通道接收 FIFO 配置寄存器 (I2S_RFCRx x=0,1)

地址偏移量：0x48，0x88
复位值：0x0000 0000
该寄存器仅在 I2S_RERx[0] 和 I2S_IRER[0] 都为 0 时可以写

位	符号	访问	描述
31:4	-	R	保留
3:0	RXCHDT	RW	配置 RX FIFO 产生数据有效中断的水平。水平 = 配置值 + 1。可配置为 0-3。

25.4.21　通道发送 FIFO 配置寄存器 (I2S_TFCRx x=0,1)

地址偏移量：0x4C，0x8C
复位值：0x0000 0000
该寄存器仅在 I2S_TERx[0] 和 I2S_ITER[0] 都为 0 时可以写

位	符号	访问	描述
31:4	-	R	保留
3:0	TXCHET	RW	配置 TX FIFO 产生空中断的水平。水平 = 配置值。可配置为 0-3。

25.4.22　通道接收 FIFO 清除寄存器 (I2S_RFFx x=0,1)

地址偏移量：0x50，0x90
复位值：0x0000 0000

该寄存器仅在 I2S_RERx[0] 和 I2S_IRER[0] 都为 0 时可以写

位	符号	访问	描述
31:1	-	R	保留
0	RXCHFR	W	向该位写 1 时清除 RX FIFO，写 0 无效。

25.4.23　通道发送 FIFO 清除寄存器 (I2S_TFFx x=0,1)

地址偏移量：0x54，0x94

复位值：0x0000 0000

该寄存器仅在 I2S_TERx[0] 和 I2S_ITER[0] 都为 0 时可以写

位	符号	访问	描述
31:1	-	R	保留
0	TXCHFR	W	向该位写 1 时清除 TX FIFO，写 0 无效。

25.4.24　接收 DMA 寄存器 (I2S_RXDMA)

地址偏移量：0x1C0

复位值：0x0000 0000

位	符号	访问	描述
31:0	RXDMA	R	接收 DMA 寄存器，循环读取各使能通道的数据。

25.4.25　清除接收 DMA 寄存器 (I2S_RRXDMA)

地址偏移量：0x1C4

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	RRXDMA	W	向该位写 1 时，从 RXDMA 读取的数据将从最低允许的通道开始。如果写操作发生在左右声道数据之间，则无效。

25.4.26　发送 DMA 寄存器 (I2S_TXDMA)

地址偏移量：0x1C8

复位值：0x0000 0000

位	符号	访问	描述
31:0	TXDMA	R	发送 DMA 寄存器，循环配置各使能通道将要发送的数据。

25.4.27　清除发送 DMA 寄存器 (I2S_RTXDMA)

地址偏移量：0x1CC

复位值：0x0000 0000

位	符号	访问	描述
31:1	-	R	保留
0	RRXDMA	W	向该位写 1 时，向 TXDMA 写入的数据将从最低允许的通道开始。如果写操作发生在左右声道数据之间，则无效。

第二十六章　内部集成电路接口 (I2C)

26.1　I2C 简介

I2C(内部集成电路) 总线接口处理微控制器与串行 I2C 总线之间的通信。WB32F10xxx 内有两个 I2C，可以控制所有 I2C 总线特定的序列、协议、仲裁和时序，可以支持标准模式、快速模式和高速模式。I2C1 提供多主模式功能，而且支持 SMBUS(系统管理总线) 协议。

可使用 DMA 来减轻 CPU 的工作量。

26.2　I2C 主要特性

- 用户可配置为从机模式或主机模式
- 兼容 I2C 总线规范 6.0 版
- 两个接口信号：串行数据线 SDA 和串行时钟线 SCL
- 标准速度模式：$<= 100Kb/s$
- 快速模式：$<= 400Kb/s$
- 高速模式：$<= 3.4Mb/s$
- 7 位和 10 位寻址模式
- 7/10 位地址模式下均支持混合读写
- 在从发送器模式，支持批量传输
- 忽略 CBUS 地址
- 发送缓冲器和接收缓冲器
- 在所有的速度模式下，处理位等待和字节等待
- 可配置软件驱动程序支持的组件参数
- SDA 保持时间可以配置
- APB 接口
- 支持 DMA 操作

- 支持清除总线特性
- 支持设备 ID 特性
- 兼容 SMBUS/PMBUS
- 支持地址解析协议 (ARP)
- 支持 SMBUS 从机检测并响应 ARP 命令
- 支持 SMBUS 报警
- 支持 SMBUS 超时和空闲条件检测
- 独立可配时钟发生器，可使 I2C 通信速度不受 PCLK 时钟频率更改的影响

26.3　I2C 特性实现

本篇介绍了 I2C1 和 I2C2 中实现的所有特性。下表具体列出 I2C1 和 I2C2 中含有的特性。

表 26.1: I2C1 和 I2C2 特性实现

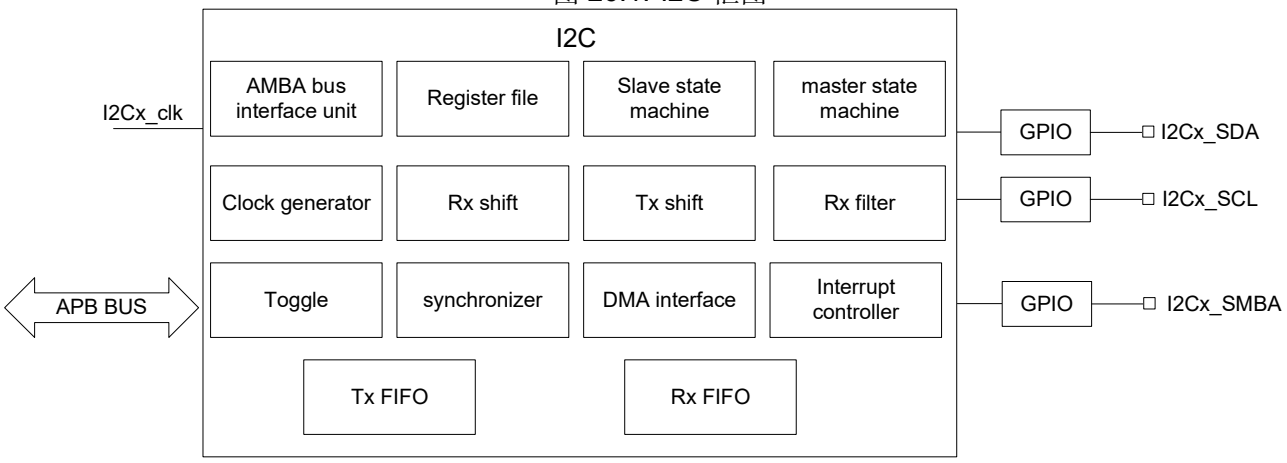
I2C 特性	I2C1	I2C2
7 位寻址模式	√	√
10 位寻址模式	√	√
标准速度模式	√	√
快速模式	√	√
高速模式	X	√
SMBUS	√	X

26.4　I2C 功能描述

26.4.1　框图

I2C 接口如图26.1所示。

图 26.1: I2C 框图



26.4.2　模式选择

该外设在工作时可选用以下四种模式之一：

- 从发送器
- 从接收器
- 主发送器
- 主接收器

在默认情况下，它以从机模式工作。用户可以通过软件配置为主机模式或者从机模式。

起始位

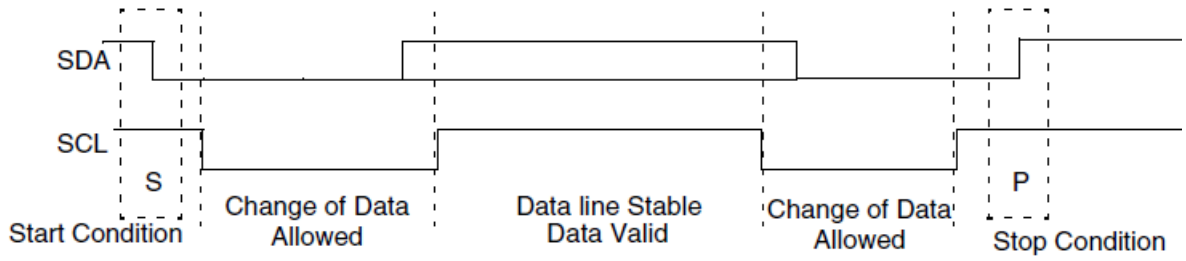
起始位是 SCL 为高时，SDA 数据线从高变为低。在 I2C 总线上出现起始位或者重新开始位时，开始串行数据传输，总线状态变为忙。在数据传输时，SDA 在 SCL 为低时变化；SCL 为高时，SDA 保持。

停止位

停止位是 SCL 为高时，SDA 数据线从低变为高。在总线上出现停止位时，停止数据传输，总线状态变为空闲。

下图26.2显示了起始位和停止位。

图 26.2: I2C 起始位和停止位



通信流程

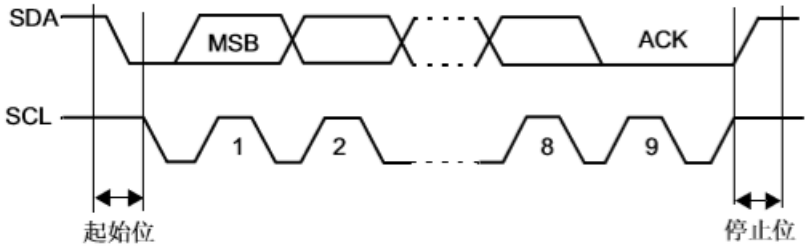
在主机模式下，I2C 接口会启动数据传输并生成时钟信号。串行数据传输始终是在出现起始位时开始，在出现停止位时结束。起始位和停止位均在主机模式下由软件生成。当用户向发送缓冲器写数据时，硬件自动产生起始位。用户可以将 IC_DATA_CMD[9] 写为 1 来产生停止位。

在从模式下，I2C 接口不会产生起始位和停止位。该外设能够识别其自身地址（7 或 10 位）

数据和地址均以 8 位字节传输，MSB 在前。起始位后紧随地址字节（7 位地址占据一个字节；10 位地址占据两个字节）。地址始终在主模式下传送。

在字节传输 8 个时钟周期后是第 9 个时钟脉冲，在此期间接收器必须向发送器发送一个应答位。请参见下图。

图 26.3: I2C 总线协议



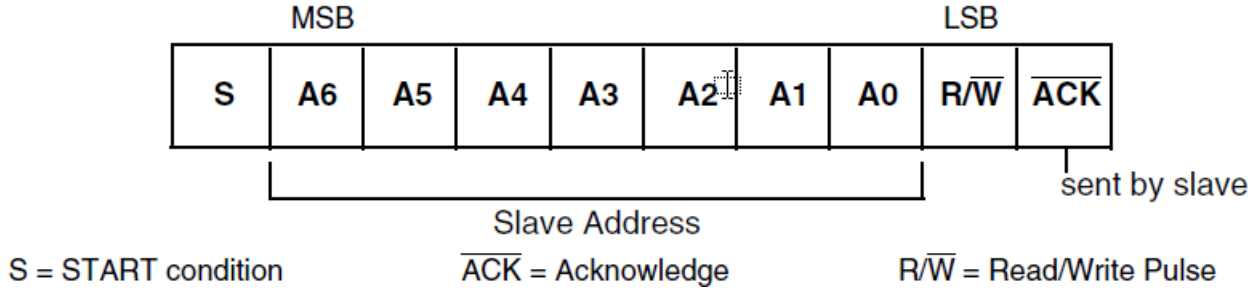
26.4.3　寻址从机协议

该外设支持两种寻址模式：7 位地址模式和 10 位地址模式

7 位寻址模式

在 7 位寻址模式中，在第一个字节中，前面 7 位表示从机地址，最后一位 (LSB, bit0) 是读写控制位。当 LSB 是 0，表示写操作，主机将向从机传输数据；当 LSB 位 1，表示读操作，主机将从从机读取数据。图26.5详细显示这一格式。

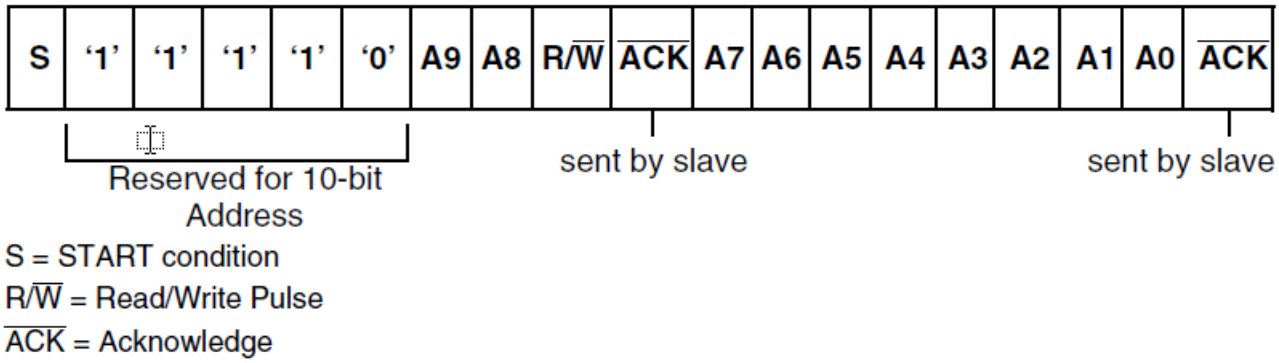
图 26.4: 10 位地址模式



10 位寻址模式

在 10 位寻址模式中，寻址操作包括两个字节。前面 5 位是 5’b11110 表示十位寻址模式，接着是高 2 位地址，然后是读写控制位。第二个字节是地址的后 8 位。图26.4详细显示这一格式。

图 26.5: 7 位地址模式



表格26.2显示了第一个字节中的特殊用途。

表 26.2: I2C/SMBUS 第一字节定义

从机地址	读写控制位	解释
0000 000	0	通用呼叫地址。将收到的数据放在接收缓冲器中，并产生通用呼叫中断。
0000 000	1	始字节。
0000 001	X	CBUS 地址，忽略
0000 010	X	保留
0000 011	X	保留
0000 1XX	X	高速主代码
1111 1XX	X	保留
1111 0XX	X	10 位寻址模式
0001 000	X	SMBUS 主机
0001 100	X	SMBUS 报警响应地址
1100 001	X	SMBUS 设备默认地址

26.4.4　收发协议

在 I2C 总线中，主机发起传输，可作为主发送器或者主接收器；从机响应主机数据传输的要求，作为从接收器或者从发送器。

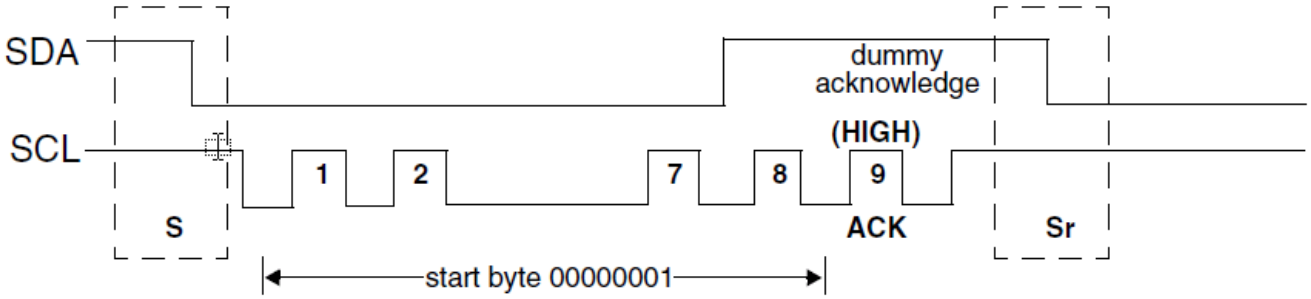
关于详细的 I2C 收发协议，请参考 I2C 相关文档。

26.4.5　始字节传输协议

始字节传输协议是为没有专用 I2C 硬件模块的系统建立的。本模块在从机模式时，总是以可支持的最高速度去采样 I2C 总线，因此不需要本协议。然而在主模式下，支持在每次传输开始之前先发送始字节，以防某些从设备需要它。

始字节是 0000 0001，如下图26.6。某些从设备以较低的频率去采样 I2C 总线，当接收到 0 之后，会调整采样频率到正常值，开始正常的数据传输。从机不需要响应始字节。

图 26.6: 始字节传输



始字节传输过程:

- 主机发起起始位
- 主机发送始字节 0000 0001
- 主机发送相应位的时钟，但是没有从机设备将 SDA 拉低，响应位始终为高
- 主机发起重新启动

26.4.6　I2C 从模式

从模式初始化配置

1. 将寄存器 IC_ENABLE 的位 0(ENABLE) 写为 0
2. 将从机地址写入 IC_SAR 寄存器
3. 设置 7 位/10 位寻址模式: IC_CON.IC_10ADDR_SLV; 配置从机模式: IC_CON.IC_SLAVE_DISABLE 和 IC_CON.MASTER_MODE 写为 0。
4. 将寄存器 IC_ENABLE 的位 0(ENABLE) 写为 1，使能该模块

从发送器——单字节

当 I2C 总线上的主机请求读操作时，该模块工作在从发送器模式。以下是具体步骤:

1. I2C 总线上的主机发起读操作要求，地址与 IC_SAR（该从机地址）匹配
2. 该模块确认发送的地址，并识别传送的方向，确定工作在从发送器模式
3. 发起 RD_REQ 中断申请，将 SCL 线拉低，等待软件响应。如果该中断源被屏蔽，用户必须定期读取 IC_RAW_INTR_STAT 的 RD_REQ 位, 响应 I2C 总线的传输要求。建议读取的周期是 SCL 最快周期的 10 倍。
4. 如果在发送缓冲器中已经有数据，将产生 TX_ABRT 中断，刷新发送缓冲器中的内容。如果 TX_ABR 中断被屏蔽，用户同样需要定期读取 IC_RAW_INTR_STAT 的 TX_ABRT 位。建议读取的周期是 SCL 最快周期的 10 倍。
5. 软件清除 RD_REQ 和 TX_ABRT 中断，将要发送的数据写入 IC_DATA_CMD, 同时位 8 写为 0(表示写操作)
6. 放开 SCL，开始传输字节
7. 主机收到数剧后，可以放开总线 (STOP) 或者继续新的传输 (RESTART)

注意:如果产生 TX_ABRT 中断,需要先读 IC_CLR_TX_ABRT 来清除中断标志,否则通过 IC_CMD_DATA 来写发送缓冲器测操作无效。

从接收器——单字节

当 I2C 总线上的主机请求写操作时，该模块工作在从接收器模式。以下是具体步骤：

1. I2C 总线上的主机发起写操作要求，地址与 IC_SAR（该从机地址）匹配
2. 该模块确认发送的地址，并识别传送的方向，确定工作在从接收器模式
3. 接收一个字节，并放在接收缓冲器中。
4. 产生 RX_FULL 中断。如果该中断被屏蔽，必须定期读取 IC_STATUS 的 RFNE 位。
5. 软件读取 IC_DATA_CMD(位 7-0)，得到读取的字节
6. 主机收到响应后，可以放开总线 (STOP) 或者继续新的传输 (RESTART)

批量传输

在从机模式下，该外设可以支持批量传输。如果总机要读取多个字节的数据，当收到 RD_REQ 中断申请之后，软件可以向发送缓冲器中写入多个数据，批量传输，不需要每次都拉低 SCL 等待软件写入下一个发送数据。这样大大提高了传输效率。

如果写入发送缓冲器中的数据过多，在收到总机的最后一个数据响应后 (NACK)，产生 TX_ABR 中断，清除发送缓冲器的内容。

注意：如果产生 *TX_ABRT* 中断，需要先读 *IC_CLR_TX_ABRT* 来清除中断标志，否则后面通过 *IC_CMD_DATA* 来写发送缓冲器测操作无效。

26.4.7　I2C 主模式

主模式初始化配置

1. 将寄存器 IC_ENABLE 的位 0(ENABLE) 写为 0。
2. 写 IC_CON 配置传输速度，地址寻址模式 (7 位/10 位)，主工作模式（位 0 和位 6 都是 1）。
3. 写 IC_TAR，配置目标地址，以及始字节，一般呼叫，设备 ID 等的发送。
4. 如果是高速传输，写 IC_HS_MADDR，配置主机编码。
5. 将寄存器 IC_ENABLE 的位 0(ENABLE) 写为 1，使能该模块
6. 写 IC_DATA_CMD，配置传输方向，要传输的数据等

注意：必须先使能该外设，然后写 *IC_DATA_CMD*。否则写 *IC_DATA_CMD* 的操作无效。

主发送器和主接收器

在主模式下，读操作和写操作可以自动转换。在读操作中，作为主接收器；在写操作中，作为主发送器。IC_DATA_CMD.READ 决定数据传输的方向。当发送缓冲器内有数据时，在总线上发起起始位，开始数据传输；当发送缓冲器位空时，拉低 SCL 保持总线，直到发送缓冲器中写入新的数据。

26.4.8　禁用 I2C 接口

将 IC_ENABLE.ENABLE 写为 0，可以禁用 I2C 接口。用户可以不断读取 IC_ENABLE_STATUS 寄存器的状态 (polling)，等到值为 0，表示 I2C 接口是否已经被禁用。

26.4.9　中止 I2C 传输

在主机模式下，用户可以写 IC_ENABLE.ABORT 为 1 来中止正在进行的数据传输，进入停止状态，放开 I2C 总线控制权，然后刷新发送缓冲器。以下是中止操作的具体过程：

1. 停止向发送缓冲器 (IC_DATA_CMD) 中写入新的命令
2. 如果 DMA 打开，将 TDMAE 写为 0，禁用发送 DMA
3. 写 IC_ENABLE.ABORT 为 1，中止传输

4. 等待 M_TX_ABRT 中断
5. 读取 IC_TX_ABRT_SOURCE 寄存器确认中止传输的源是 ABRT_USER_ABRT.

26.4.10　尖峰抑制

该外设内含有尖峰抑制电路，来过滤 I2C 总线的两个输入信号：SDA 和 SCL。用户需要根据 I2C 总线的协议和传输速度，配置 IC_FS_SPKLEN/IC_HS_SPKLEN, 指定最大尖峰宽度。

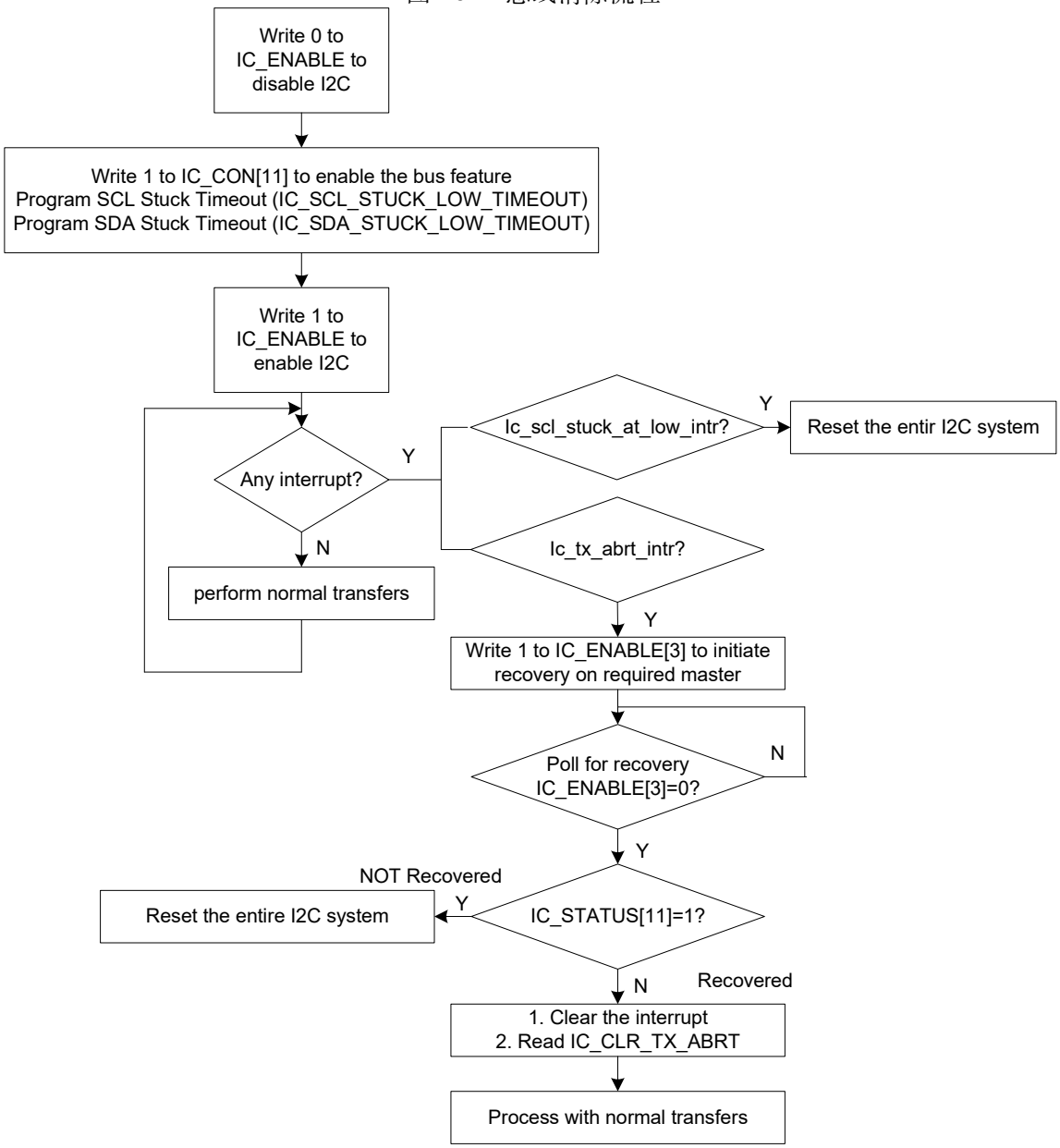
根据 *I2C* 总线协议，在 *SS/FS*，最大尖峰宽度是 *50ns*；在 *HS*，最大尖峰宽度是 *10ns*。

26.4.11　总线清除

如果 I2C 总线由于某些问题，SCL 或者 SDA 保持在低电平，用户可以使用总线清除功能恢复总线状态，避免硬件复位。

图26.7详细解释了这一过程。

图 26.7: 总线清除流程



26.4.12　读取器件 ID

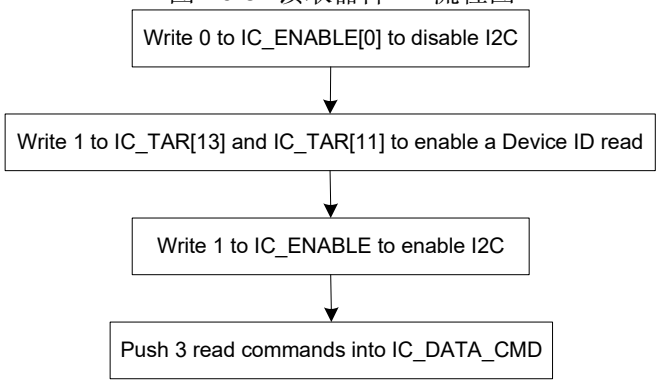
器件 ID 由 3 个字节组成:

- 独一无二的制造商名称，12 位
- 制造商指定的零件识别，9 位
- 制造商指定的 die 修订版本，3 位

在主机模式下,用户可以读取总线上其他从机的 ID。读出的数据存在接收缓冲器中,可以读取 IC_DATA_CMD 寄存器得到。具体流程参考图26.8

在从机模式下，用户可以通过寄存器 IC_DEVICE_ID 来配置器件 ID。

图 26.8: 读取器件 ID 流程图



26.4.13　SMBUS 特性

系统管理总线 (SMBus) 是一个双线制接口，各器件可通过它在彼此之间或者与系统的其余部分进行通信。它以 I2C 的工作原理为基础。SMBus 可针对系统和电源管理相关的任务提供控制总线。

系统管理总线规范涉及三类器件：

- 从器件，用于接收或响应命令。
- 主器件，用于发出命令、生成时钟和中止传输。
- 主机，专用的主器件，可提供连接系统 CPU 的主接口。主机必须具有主 - 从器件功能，并且必须支持 SMBus 主机通知协议。系统中只允许存在一个主机。

I2C1 可配置为主器件或从器件，也可配置为主机。

26.4.14　SMBUS 总线协议

任何给定器件都有十一种可用命令协议。器件既可以在这十一种协议中任选其一，也可以使用全部十一种协议进行通信。这十一种协议分别为快速命令、发送字节、接收字节、写入字节、写入字、读取字节、读取字、过程调用、块读取、块写入以及块写入-块读取过程调用。这些协议应通过用户软件实施。

在 SMBUS 主模式，所有接收到的数据都存在接受缓冲器中。在 SMBUS 从模式，所有收到的命令协议的编码和数据都存在接收缓冲器中，要求发送的数据都通过发送缓冲器。

该外设可以配置为只接收快速命令 (IC_CON.SLV_QUICK_CMD_EN)。当收到快速命令时，产生 SMBUS_QUICK_DET 中断。

26.4.15　SMBUS 地址解析协议 (ARP)

通过为各个从器件动态分配一个新的唯一地址可解决 SMBus 从地址冲突的问题。为了提供一种机制来针对地址分配隔离各个器件，各器件必须具有唯一的器件标识符 (UDID)。UDID 是 128 位的数字。在 I2C1 中，高 96 位都是 0，低 32 位可以由 IC_SMBUS_ARP_UDID_LSB 来配置。

该外设支持地址解析协议 (ARP)。通过将 IC_CON 寄存器中的 SMBUS_ARP_EN 位置 1 来使能 ARP。ARP 可有软件实现。

同时该外设支持固定从地址，可以通过 IC_CON.PERSISANT_SLV_ADDR_EN 配置。

此外，还将在从模式下执行仲裁以支持 ARP。

26.4.16　SMBUS 报警

I2C1 含有 SMBALERT 信号，支持 SMBUS 报警功能。在从模式时，软件可以设置 IC_ENABLE.SMBUS_ALERT_CTRL 产生报警，待收到主设备发来的 ARA(alert response address) 后将 IC_SAR[7:0] 发给主设备，然后解除报警。

在主模式时，如果报警中断，软件需要发送 ARA，读取发出报警的从设备地址。如果报警仍然存在，继续发送 ARA，读取下一个发出报警的从设备地址。通过 IC_STATUS[20] 可以读取报警的状态。

26.4.17　SDA 保持时间

再 I2C 协议中规定 SDA 需要满足一定的保持时间。用户可以通过配置 IC_SDA_HOLD 寄存器动态调整 SDA 的保持时间。当 I2C 总线传输速度变化时，需要同时调整 SDA 动态保持时间。

IC_SDA_HOLD[15:0] 是 IC_SDA_TX_HOLD，用来调整发送模式时 SDA 的保持时间。但在发送模式下，该外设有最低 SDA 保持时间。当作为主发送器时，最小 SDA 保持时间 1 个 ic_clk 周期；当作为从发送器时，最小 SDA 保持时间是 $SPKLEN + 7$ 个 ic_clk 周期 (SPKLEN 指 IC_FS_SPKLEN 或者 IC_HS_SPKLEN)。

26.4.18　I2C 时钟频率

当工作在主模式时，进行 I2C 总线传输之前，用户需要根据传输速度配置以下寄存器：

- IC_SS_SCL_HCNT
- IC_SS_SCL_LCNT
- IC_FS_SCL_HCNT
- IC_FS_SCL_LCNT
- IC_HS_SCL_HCNT
- IC_HS_SCL_LCNT

在配置这些寄存器时，有以下限制：

- IC_SS_SCL_LCNT 和 IC_FS_SCL_LCNT 需要大于 IC_FS_SPKLEN+7
- IC_SS_SCL_HCNT 和 IC_FS_SCL_HCNT 需要大于 IC_FS_SPKLEN+5
- IC_HS_SCL_LCNT 需要大于 IC_HS_SPKLEN+7
- IC_HS_SCL_HCNT 需要大于 IC_HS_SPKLEN+5

表26.3详细显示了各种速度模式时这些寄存器的配置。

表 26.3: 时钟频率配置

速度模式	ic_clk 频率 (MHz)	IC_*SPK_LEN 最小值	SCL 低电平时间	SCL 低电平配置	SCL 高电平时间	SCL 高电平配置
SS	2.7	1	4.7us	12	5.2us	6
FS	12	1	1.33us	15	1.16us	6
HS(400pf)	51	1	333ns	16	274ns	6
HS(100pf)	105.4	1	161ns	16	132ns	6

26.4.19　DMA 接口

该设备支持 DMA 传输，可以向 DMAC 发出 DMA 申请。用户可以通过写 IC_DMA_CR 来打开发送 DMA 和接收 DMA 功能。当发送缓冲器中的数据少于或者等于 IC_DMA_TDLR 规定的数目时，产生发送 DMA 申请；当接收缓冲器中的数据大于或者等于 IC_DMA_RDLR 规定的数目时，产生接收 DMA 申请。

在进行 DMA 传输之前，用户需要先配置 DMAC。详情参考 DMAC 相关章节。

26.4.20　中断

下表详细列出了所有的中断。

表 26.4: 中断源

工作模式	中断名称	中断产生条件	中断清除
I2C 模式	SCL_STUCK_AT_LOW	SCL 连续为低 IC_SCL_STUCK_LOW_TIMOUT 时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_SCL_STUCK_DET
	RS_DET	在从机模式，当总线上出现重新开始状态且被寻址时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_RESTART_DET
	GEN_CALL	在从机模式，收到通用呼叫且被确认时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_GEN_CALL
	START_DET	当总线上出现开始状态或者重新开始状态时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_START_DET
	STOP_DET	当总线上出现停止状态时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_STOP_DET
	ACTIVITY	当总线没有空闲，处于活动状态时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_ACTIVITY_DET
	RX_DONE	aaa	1. 读 IC_CLR_INTR 2. 读 IC_CLR_RX_DONE
	TX_ABRT	当不能发送完 TX FIFO 中的所有命令，需要中止传输时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_TX_ABRT
	RD_REQ	在从机模式下，主机要求读操作时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_RD_REQ
	TX_EMPTY	当 IC_CON.TX_EMP_CTRL=0：发送缓冲器内的命令比 IC_TX_TL 时，产生中断 当 IC_CON.TX_EMP_CTRL=1：发送缓冲器内的命令比 IC_TX_TL 而且最后一个命令的地址和数据已经发送完毕时，产生中断	1. 读 IC_CLR_INTR 2. 写 IC_DATA_CMD，使发送缓冲器内的命令数量大于 IC_TX_TL
	TX_OVER	如果发送缓冲器已满，继续写 IC_DATA_CMD 时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_TX_OVER

工作模式	中断名称	中断产生条件	中断清除
	RX_FULL	如果接收缓冲器已满 (接受缓冲器内的数据大于或者等于 IC_RX_TL)，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_DATA_CMD，是接收缓冲器的数据小于 IC_RX_TL。
	RX_OVER	如果接收缓冲器已满，又收到一个新的数据时，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_RX_OVER
	RX_UNDER	软件读取 IC_DATA_CMD 时，如果接收缓冲器为空，产生中断	1. 读 IC_CLR_INTR 2. 读 IC_CLR_RX_UNDER
SMBUS 模式	SMBUS_ALERT_DET	SMBALERT 上是低电平时候产生中断	向 IC_CLR_SMBUS_INTR[10] 写 1 清除中断。
	SMBUS_SUSPEND_DET	SMBSUS 上是低电平时候产生中断	向 IC_CLR_SMBUS_INTR[9] 写 1 清除中断。
	SLV_RX_PEC_NACK	从模式下，在 ARP 协议中，由于 PEC 不符合而返回 NACK 时产生中断	向 IC_CLR_SMBUS_INTR[8] 写 1 清除中断。
	ARP_ASSGN_ADDR_CMD_DET	收到分配地址 ARP 命令时产生中断	向 IC_CLR_SMBUS_INTR[7] 写 1 清除中断。
	ARP_GET_UDID_CMD_DET	收到读取 UDID ARP 命令时产生中断	向 IC_CLR_SMBUS_INTR[6] 写 1 清除中断。
	ARP_RST_CMD_DET	收到复位 ARP 命令时产生中断	向 IC_CLR_SMBUS_INTR[5] 写 1 清除中断。
	ARP_PREPARE_CMD_DET	收到准备 ARP 命令时产生中断	向 IC_CLR_SMBUS_INTR[4] 写 1 清除中断。
	HOST_NOTIFY_MST_DET	收到 Notify ARP Master ARP 命令时产生中断	向 IC_CLR_SMBUS_INTR[3] 写 1 清除中断。
	QUICK_CMD_DET	收到快速命令时产生中断	向 IC_CLR_SMBUS_INTR[2] 写 1 清除中断。
	MCLKEXT_TIMEOUT	时钟延伸超时，产生中断	向 IC_CLR_SMBUS_INTR[1] 写 1 清除中断。
	SCLKEXT_TIMEOUT	数据延伸超时，产生中断	向 IC_CLR_SMBUS_INTR[0] 写 1 清除中断。

26.5　寄存器描述

26.5.1　控制寄存器 (CON)

地址偏移量：0x0

复位值：0x0000 0024 (I2C1) 0x0000 003E (I2C2)

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:20	-	R	保留

19	SMBUS_PERS_SLV_ADDR_EN	RW	该位仅在从机模式有用，选择使能 PSA 模式。 1：I2C 从机工作在 PSA 模式 0: I2C 从机不在 PSA 模式，从机地址由主机分配。 *I2C1 存在，I2C2 中不存在*
18	SMBUS_ARP_EN	RW	该位仅在从机模式有用，选择是否支持 SMBUS 地址解决协议 (ARP)。 1: 支持 SMBUS 地址解决协议，允许动态分配地址 0: 不支持 SMBUS 地址解决协议 *I2C1 存在，I2C2 中不存在*
17	SMBUS_SLV_QUICK_EN	RW	该位仅在从机模式有用 1: I2C 从机仅接收 QUICK 命令，不支持的总线协议。 0：I2C 从机不接收 QUICK 命令，仅支持其他的总线协议。 *I2C1 存在，I2C2 中不存在*
16	OP_SAR_CTRL	RW	该位仅在从机模式有用，选择使能寄存器 IC_OPTIONAL_SAR。如果用户要使用附加从机地址，需要首先配置 IC_OPTIONAL_SAR。 *I2C1 存在，I2C2 中不存在*
15:12	-	R	保留
11	BUS_CLR_CTRL	RW	该位仅在主机模式有用 1: 使能总线清除 0：禁用总线清除
10	STOP_DET_IF_MST_ACTIVE	RW	该位仅在主机模式有用 1: 当主机活动时产生 STOP_DET 中断 0：产生 STOP_DET 中断与主机是否处于活动状态无关 (acitve)
9	RX_FIFO_FULL_HLD_CTRL	RW	1：当 RX FIFO 满时，保持总线； 1：当 RX FIFO 满时，溢出；
8	TX_EMP_CTRL	RW	该位控制如何产生 TX_EMPTY 中断。
7	STOP_DET_IFADDRESSED	RW	该位仅在从机模式有用。 1: 当从机被寻址时，产生 STOP_DET 中断 0：产生 STOP_DET 中断与从机是否被寻址无关
6	IC_SLV_DISABLE	RW	1：关闭从机模式 0：使能从机模式 *如果用户将该位写 0，必须同时将该寄存器的位 0 也写 0。*
5	IC_RS_EN	RW	1：使能主机重新开始 0：禁止主机重新开始
4	IC_10ADDR_MST	RW	1：主机地址 10 位 0：主机地址 7 位
3	IC_10ADDR_SLV	RW	1：从机地址 10 位 0：从机地址 7 位

2:1	SPEED	RW	该位控制 I2C 总线传输的速度 00：保留 01：标准速度 100Kb/s 10: 快速 <=400Kb/s 11: 高速 3.4Mb/s *I2C1 中只能配为 1 或者 2，I2C2 中可配为 1,2 或 3*
0	MST_MODE	RW	1: 使能主机模式 0：禁用主机模式 *如果用户将该位写 1，必须同时将该寄存器的位 6 也写 1。*

26.5.2　目标地址寄存器 (TAR)

地址偏移量：0x4

复位值：0x0000 0055

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

如果 I2C 进工作在从机模式，该寄存器不用配置

位	符号	访问	描述
31:17	-	R	保留
16	SMBUS_QUICK_CMD	RW	当 SPECIAL=1 时，该位有效。 1：使能 QUICK_CMD 0：禁止 QUICK_CMD *I2C1 存在，I2C2 中不存在*
15:14	-	R	保留
13	DEVICE_ID	RW	当 SPECIAL=1 时，该位有效。 1：使能 DEVICE_ID 0：禁止 DEVICE_ID
12	-	R	保留
11	SPECIAL	RW	1：允许 DEVICE_ID, GENERAL_CALL 和 START 传输 0：禁止 DEVICE_ID, GENERAL_CALL 和 START 传输
10	GC_OP_START	RW	当 SPECIAL=1，DEVICE_ID=0 时，该位有效。 1：发送 START 0: 发送 GENERAL_CALL
9:0	TAR	RW	主机传输的目标地址。

26.5.3　从机地址寄存器 (SAR)

地址偏移量：0x8

复位值：0x0000 03F0

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:10	-	R	保留

9:0	SAR	RW	从机地址。在 7 位地址模式下，只有 SAR[6:0] 有效。该寄存器仅当 I2C 工作在从机模式下有用。

26.5.4　高速主机编码地址寄存器 (HS_MADDR)

地址偏移量：0xC
复位值：0x0000 0001
只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。
该寄存器在 I2C1 中不存在，仅在 I2C2 中存在

位	符号	访问	描述
31:3	-	R	保留
2:0	HS_MAR	RW	高速模式下，主机编码。

26.5.5　I2C 数据与命令寄存器 (DATA_CMD)

地址偏移量：0x10
复位值：0x0000 0000
只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:12	-	R	保留
11	FIRST_DATA_BYTE	R	表示地址段结束之后，收到第一个字节的数据。
10	RESTART	W	控制在发本次数据传输之前，是否需要发 RESTART。高有效。
9	STOP	W	控制当前发送或者接收结束后，是否发 STOP。高有效。
8	READ	W	仅在主机模式下有效。 1：主机发出读命令 0：主机发出写命令
7:0	DAT	RW	读操作时，该寄存器是 I2C 总线上收到到的数据； 写操作时，该寄存器写入将要在 I2C 总线上传输的数据。

26.5.6　I2C 快速时钟高计数寄存器 (SS_SCL_HCNT)

地址偏移量：0x14
复位值：0x0000 0320

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_SS_SCL_HCNT	RW	标准速度模式时，SCL 高电平时间。以 ic_clk 周期为单位。

26.5.7　I2C 快速时钟高计数寄存器 (SS_SCL_LCNT)

地址偏移量：0x18
复位值：0x0000 03AC

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_SS_SCL_LCNT	RW	标准速度模式时，SCL 低电平时间。以 ic_clk 周期为单位。

26.5.8　I2C 快速时钟高计数寄存器 (FS_SCL_HCNT)

地址偏移量：0x1C

复位值：0x0000 0078

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_FS_SCL_HCNT	RW	快速模式时，SCL 高电平时间。以 ic_clk 周期为单位。

26.5.9　I2C 快速时钟高计数寄存器 (FS_SCL_LCNT)

地址偏移量：0x20

复位值：0x0000 0104

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_FS_SCL_LCNT	RW	快速模式时，SCL 低电平时间。以 ic_clk 周期为单位。

26.5.10　I2C 高速时钟高计数寄存器 (HS_SCL_HCNT)

地址偏移量：0x24

复位值：0x0000 000C

该寄存器在 I2C1 中不存在，仅在 I2C2 中存在

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_HS_SCL_HCNT	RW	高速模式时，SCL 高电平时间。以 ic_clk 周期为单位。

26.5.11　I2C 高速时钟低计数寄存器 (HS_SCL_LCNT)

地址偏移量：0x28

复位值：0x0000 0020

该寄存器在 I2C1 中不存在，仅在 I2C2 中存在

位	符号	访问	描述
31:16	-	R	保留
15:0	IC_HS_SCL_LCNT	RW	高速模式时，SCL 低电平时间。以 ic_clk 周期为单位。

26.5.12　I2C 中断状态寄存器 (INTR_STAT)

地址偏移量：0x2C

复位值：0x0000 0000

位	符号	访问	描述
31:15	-	R	保留
14	R_SCL_STUCK_AT_LOW	R	中断 SCL_STUCK_AT_LOW 的状态。高有效
13	-	R	保留
12	R_RS_DET	R	中断 RS_DET 的状态。高有效
11	R_GEN_CALL	R	中断 GEN_CALL 的状态。高有效
10	R_START_DET	R	中断 START_DET 的状态。高有效
9	R_STOP_DET	R	中断 STOP_DET 的状态。高有效
8	R_ACTIVITY	R	中断 ACTIVITY 的状态。高有效
7	R_RX_DONE	R	中断 RX_DONE 的状态。高有效
6	R_TX_ABRT	R	中断 TX_ABRT 的状态。高有效
5	R_RD_REQ	R	中断 RD_REQ 的状态。高有效
4	R_TX_EMPTY	R	中断 TX_EMPTY 的状态。高有效
3	R_TX_OVER	R	中断 TX_OVER 的状态。高有效
2	R_RX_FULL	R	中断 RX_FULL 的状态。高有效
1	R_RX_OVER	R	中断 RX_OVER 的状态。高有效
0	R_RX_UNDER	R	中断 RX_UNDER 的状态。高有效

26.5.13　I2C 中断屏蔽寄存器 (INTR_MASK)

地址偏移量：0x30

复位值：0x0000 0048FF

位	符号	访问	描述
31:15	-	R	保留
14	M_SCL_STUCK_AT_LOW	RW	屏蔽中断 SCL_STUCK_AT_LOW。 1: 屏蔽中断 SCL_STUCK_AT_LOW 0：使能中断 SCL_STUCK_AT_LOW
13	-	R	保留
12	M_RS_DET	RW	中断 RS_DET 的状态。 0: 屏蔽中断 RS_DET 1：使能中断 RS_DET
11	M_GEN_CALL	RW	中断 GEN_CALL 的状态。 0: 屏蔽中断 GEN_CALL 1：使能中断 GEN_CALL
10	M_START_DET	RW	中断 START_DET 的状态。 0: 屏蔽中断 START_DET 1：使能中断 START_DET

9	M_STOP_DET	RW	中断 STOP_DET 的状态。 0: 屏蔽中断 STOP_DET 1：使能中断 STOP_DET
8	M_ACTIVITY	RW	中断 ACTIVITY 的状态。 0: 屏蔽中断 ACTIVITY 1：使能中断 ACTIVITY
7	M_RX_DONE	RW	中断 RX_DONE 的状态。 0: 屏蔽中断 RX_DONE 1：使能中断 RX_DONE
6	M_TX_ABRT	RW	中断 TX_ABRT 的状态。 0: 屏蔽中断 TX_ABRT 1：使能中断 TX_ABRT
5	M_RD_REQ	RW	中断 RD_REQ 的状态。 0: 屏蔽中断 RD_REQ 1：使能中断 RD_REQ
4	M_TX_EMPTY	RW	中断 TX_EMPTY 的状态。 0: 屏蔽中断 TX_EMPTY 1：使能中断 TX_EMPTY
3	M_TX_OVER	RW	中断 TX_OVER 的状态。 0: 屏蔽中断 TX_OVER 1：使能中断 TX_OVER
2	M_RX_FULL	RW	中断 RX_FULL 的状态。 0: 屏蔽中断 RX_FULL 1：使能中断 RX_FULL
1	M_RX_OVER	RW	中断 RX_OVER 的状态。 0: 屏蔽中断 RX_OVER 1：使能中断 RX_OVER
0	M_RX_UNDER	RW	中断 RX_UNDER 的状态。 0: 屏蔽中断 RX_UNDER 1：使能中断 RX_UNDER

26.5.14　I2C 中断原始状态寄存器 (RAW_INTR_STAT)

地址偏移量：0x34

复位值：0x0000 0000

位	符号	访问	描述
31:15	-	R	保留
14	SCL_STUCK_AT_LOW	R	中断 SCL_STUCK_AT_LOW 的原始状态。SCL 连续为低 IC_SCL_STUCK_LOW_TIMOUT 个 ic_clk 时钟周期时，发生 SCL_STUCK_AT_LOW 中断。 1: 中断 SCL_STUCK_AT_LOW 有效 0: 中断 SCL_STUCK_AT_LOW 无效
13	-	R	保留

12	R_RS_DET	R	中断 RS_DET 的原始状态。工作在从机模式且正在被寻址时，在总线上有 RESTART 状态，产生 RS_DET 中断。 1: 中断 RS_DET 有效 0: 中断 RS_DET 无效
11	R_GEN_CALL	R	中断 GEN_CALL 的原始状态。接收到一般地址呼叫并且被确认时，产生 GEN_CALL 中断。 1: 中断 GEN_CALL 有效 0: 中断 GEN_CALL 无效
10	R_START_DET	R	中断 START_DET 的原始状态。不管是从机模式还是主机模式，在 I2C 总线上有开始或者重新开始的状态时，产生 START_DET 中断。 1: 中断 START_DET 有效 0: 中断 START_DET 无效
9	R_STOP_DET	R	中断 STOP_DET 的原始状态。不管是从机模式还是主机模式，在 I2C 总线上有停止的状态时，产生 STOP 中断。 1: 中断 STOP 有效 0: 中断 STOP 无效
8	R_ACTIVITY	R	中断 ACTIVITY 的原始状态。当检测到 I2C 处于活动状态时产生 ACTIVITY 中断。 1: 中断 ACTIVITY 有效 0: 中断 ACTIVITY 无效
7	R_RX_DONE	R	中断 RX_DONE 的原始状态。在从机模式下发送结束时，产生 RX_DONE 中断。 1: 中断 RX_DONE 有效 0: 中断 RX_DONE 无效
6	R_TX_ABRT	R	中断 TX_ABRT 的原始状态。当不能发送完 TX FIFO 中的全部内容，需要中止传送时，产生 TX_ABRT 中断。 1: 中断 TX_ABRT 有效 0: 中断 TX_ABRT 无效
5	R_RD_REQ	R	中断 RD_REQ 的原始状态。在从机模式下，主机要求读操作时产生 RD_REQ 中断。 1: 中断 RD_REQ 有效 0: 中断 RD_REQ 无效
4	R_TX_EMPTY	R	中断 TX_EMPTY 的原始状态。 如果 IC_CON.TX_EMP_CTRL 为 0 时：发送缓冲器中的数据比 IC_TX_TL 少时产生中断； 如果 IC_CON.TX_EMP_CTRL 为 0 时：发送缓冲器中的数据比 IC_TX_TL 少而且最新拿到的命令的地址和数据已经传送结束，产生中断； 1: 中断 TX_EMPTY 有效 0: 中断 TX_EMPTY 无效

3	R_TX_OVER	R	中断 TX_OVER 的原始状态。当发送缓冲器已满，如果继续写 IC_DATA_CMD，将产生 TX_OVER。 1: 中断 TX_OVER 有效 0: 中断 TX_OVER 无效
2	R_RX_FULL	R	中断 RX_FULL 的原始状态。当接收缓冲器已满时 (接收缓冲器的数据大于或者等于 IC_RX_TL 时)，产生 RX_FULL 中断。 1: 中断 RX_FULL 有效 0: 中断 RX_FULL 无效
1	R_RX_OVER	R	中断 RX_OVER 的原始状态。当接收缓冲器已满，又接收到一个新的字节时，产生 RX_OVER 中断。 1: 中断 RX_OVER 有效 0: 中断 RX_OVER 无效
0	R_RX_UNDER	R	中断 RX_UNDER 的原始状态。当读 IC_DATA_CMD 时，如果接收缓冲器为空，产生 RX_UNDER 中断。 1: 中断 RX_UNDER 有效 0: 中断 RX_UNDER 无效

26.5.15　I2C 接收缓冲器阈值寄存器 (RX_TL)

地址偏移量：0x38

复位值：0x0000 0005

位	符号	访问	描述
31:8	-	R	保留
7:0	RX_TL	RW	接收缓冲器的阈值，控制何时产生 RX_FULL 中断。

26.5.16　I2C 发送缓冲器阈值寄存器 (TX_TL)

地址偏移量：0x3C

复位值：0x0000 0005

位	符号	访问	描述
31:8	-	R	保留
7:0	TX_TL	RW	发送缓冲器的阈值，控制何时产生 TX_EMPTY 中断。

26.5.17　I2C 清除中断寄存器 (CLR_INTR)

地址偏移量：0x40

复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_INTR	R	当用户读取该位时，将清除所有的中断状态，以及 IC_TX_ABRT_SOURCE 寄存器 (bit9 除外)

26.5.18　I2C 清除 RX_UNDER 中断寄存器 (CLR_RX_UNDER)

地址偏移量：0x44
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_RX_UNDER	R	当用户读取该位时，将清除 RX_UNDER 中断。

26.5.19　I2C 清除 RX_OVER 中断寄存器 (CLR_RX_OVER)

地址偏移量：0x48
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_RX_OVER	R	当用户读取该位时，将清除 RX_OVER 中断。

26.5.20　I2C 清除 TX_OVER 中断寄存器 (CLR_TX_OVER)

地址偏移量：0x4C
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_TX_OVER	R	当用户读取该位时，将清除 TX_OVER 中断。

26.5.21　I2C 清除 RD_REQ 中断寄存器 (CLR_RD_REQ)

地址偏移量：0x50
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_RD_REQ	R	当用户读取该位时，将清除 RD_REQ 中断。

26.5.22　I2C 清除 TX_ABRT 中断寄存器 (CLR_TX_ABRT)

地址偏移量：0x54
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_TX_ABRT	R	当用户读取该位时，将清除 TX_ABRT 中断和 IC_TX_ABRT_SOURCE 寄存器 (bit9 除外)，同时可以继续写 TX 缓冲器开始新的传输。

26.5.23　I2C 清除 RX_DONE 中断寄存器 (CLR_RX_DONE)

地址偏移量：0x58
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_RX_DONE	R	当用户读取该位时，将清除 RX_DONE 中断。

26.5.24　I2C 清除 ACTIVITY 中断寄存器 (CLR_ACTIVITY)

地址偏移量：0x5C
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_ACTIVITY	R	当用户读取该位时，如果 I2C 不处于活动状态，将清除 ACTIVITY 中断; 如果 I2C 仍然处于活动状态，不能清除 ACTIVITY 中断。 读取时，返回 IC_RAW_INTR_STAT 内的状态。

26.5.25　I2C 清除 STOP_DET 中断寄存器 (CLR_STOP_DET)

地址偏移量：0x60
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_STOP_DET	R	当用户读取该位时，将清除 STOP_DET 中断;

26.5.26　I2C 清除 START_DET 中断寄存器 (CLR_START_DET)

地址偏移量：0x64
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_START_DET	R	当用户读取该位时，将清除 START_DET 中断;

26.5.27　I2C 清除 GEN_CALL 中断寄存器 (CLR_GEN_CALL)

地址偏移量：0x68
复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留

0	CLR_GEN_CALL	R	当用户读取该位时，将清除 GEN_CALL 中断;

26.5.28　I2C 使能寄存器 (ENABLE)

地址偏移量：0x6C
复位值：0x0000 0000

位	符号	访问	描述
31:19	-	R	保留
18	SM_ALTER_EN	RW	该位用来控制 SMBALTER 信号： 1：将 SMBALTER 信号拉高，指示 SMBUS 主机。 0：不会将 SMBALTER 信号拉高 *I2C1 存在，I2C2 中不存在*
17	SM_SUSPEND_EN	RW	该位用来控制 SMBSUS 信号： 1：将 SMBSUS 信号拉高，表示主机处于暂停状态 0：将 SMBSUS 信号拉低，表示主机没有处于暂停状态 *I2C1 存在，I2C2 中不存在*
16	SM_CLK_RST	RW	该位在主机模式下使用，用来发起主机时钟复位。 1：主机发起 SMBUS 时钟复位过程 0：主机不发起 SMBUS 时钟复位过程 *I2C1 存在，I2C2 中不存在*
15:4	-	R	保留
3	SDA_STUCK_RC_EN	RW	当传送中止由于 SDA 线固定在低电平引起时，用户可以将该位置 1 来发起恢复 SDA 的过程。该位会自动清零。
2	TX_CMD_BLOCK	RW	在主机模式下： 1：不管发送缓冲器中是否有数据，阻塞 I2C 总线上的传输 0：当发送缓冲器内有数据，会自动开始 I2C 总线上的传输
1	ABORT	RW	该位用于主机模式，用户可以将该位设置为来中止传输。关于中止操作的详细过程，请参考26.4.9。 该位会自动清除。在 ENABLE 为 1 时，置位操作才有效。 1：ABORT 操作正在进行 0: ABORT 操作已经结束或者没有发起 ABORT 操作
0	ENABLE	RW	该位用来控制该模块是否被使能，高有效。

26.5.29　I2C 状态寄存器 (STATUS)

地址偏移量：0x70
复位值：0x0000 0006
该寄存器指示当前传输的状态和 FIFO 的状态，不会产生中断申请。用户可以随时读取该寄存器。
当用户将 IC_ENABLE 写零时：

- bit 1 和 bit 2 被置为 1
- bit 3 和 bit 10 被置为 0

位	符号	访问	描述
31:21	-	R	保留
20	SM_ALTER_STATUS	R	SMBUS 警告信号控制位 1：SMBUS alert 被拉高 0：SMBUS alert 没有被拉高 该位在 *I2C2* 中不存在，仅在 *I2C1* 中存在
19	SM_SUSPEND_STATUS	R	SMBUS SUSPEND 状态位 1：SMBUS 处于暂停状态 0：SMBUS 没有处于暂停状态 该位在 *I2C2* 中不存在，仅在 *I2C1* 中存在
18	SM_SLV_ADDR_RESOLVED	R	SMBUS 从机地址解析状态位 1：SMBUS 从机地址已经解析 0：SMBUS 从机地址没有解析 该位在 *I2C2* 中不存在，仅在 *I2C1* 中存在
17	SM_SLV_ADDR_VLD	R	SMBUS 从机地址有效状态位 1：SMBUS 从机地址有效 0：SMBUS 从机地址无效 该位在 *I2C2* 中不存在，仅在 *I2C1* 中存在
16	SM_QUICK_CMD_BIT	R	SMBUS QUICK CMD 控制位 1: SMBUS QUICK CMD R/W 位被置为 1 0：SMBUS QUICK CMD R/W 位被置为 0 该位在 *I2C2* 中不存在，仅在 *I2C1* 中存在
15:12	-	R	保留
11	SDA_STUCK_NOT_RECOVERED	R	恢复机制之后，SDA 状态位 1：在恢复机制之后，信号 SDA 已经恢复 0：在恢复机制之后，信号 SDA 没有恢复，仍然保持为 0
10	SLV_HOLD_RX_FIFO_FULL	R	接收缓冲器满，从机保持总线状态位 1：由于接收缓冲器满，从机保持总线 0：从机没有保持总线或者虽然总线被保持，但不是由于接收缓冲器满引起的
9	SLV_HOLD_TX_FIFO_EMPTY	R	发送缓冲器空，从机保持总线状态位 1：由于发送缓冲器空，从机保持总线 0：从机没有保持总线或者虽然总线被保持，但不是由于发送缓冲器空引起的
8	MST_HOLD_RX_FIFO_FULL	R	接收缓冲器满，主机保持总线状态位 1：由于接收缓冲器满，主机保持总线 0：主机没有保持总线或者虽然总线被保持，但不是由于接收缓冲器满引起的
7	MST_HOLD_TX_FIFO_EMPTY	R	发送缓冲器空，主机保持总线状态位 1：由于发送缓冲器空，主机保持总线 0：主机没有保持总线或者虽然总线被保持，但不是由于发送缓冲器空引起的

6	SLV_ACTIVITY	R	从机活动状态位 1: 从机状态机不在空闲状态，从机部分处于活动状态 0：从机状态机处于空闲状态，从机部分没有处于活动状态
5	MST_ACTIVITY	R	主机活动状态位 1: 主机状态机不在空闲状态，主机部分处于活动状态 0：主机状态机处于空闲状态，主机部分没有处于活动状态
4	RFF	R	接收缓冲器满状态位: 1：接收缓冲器满 0: 接收缓冲器不满
3	RFNE	R	接收缓冲器非空状态位: 1：接收缓冲器非空 0：接收缓冲器空
2	TFE	R	发送缓冲器空状态位: 1: 发送缓冲器空 0: 发送缓冲器非空
1	TFNF	R	发送缓冲器满状态位: 1: 发送缓冲器满 0: 发送缓冲器内有满，可以继续写
0	ACTIVITY	R	I2C 活动状态位: 1: I2C 处于活动状态 0：I2C 处于空闲状态

26.5.30　I2C 发送缓冲器水平寄存器 (TXFLR)

地址偏移量：0x74
复位值：0x0000 0000
该寄存器在以下条件被清除:

- I2C 被禁用 (IC_ENABLE.ENABLE 被清零)
- 传输中止- IC_RAW_INTR_STAT.TX_ABRT 被置位
- 从机批量传输被中止

位	符号	访问	描述
31:30	-	R	保留
2:0	TXFLR	R	发送缓冲器内的数据个数。

26.5.31　I2C 接收缓冲器水平寄存器 (RXFLR)

地址偏移量：0x78
复位值：0x0000 0000
该寄存器在以下条件被清除:

- I2C 被禁用 (IC_ENABLE.ENABLE 被清零)
- 传输中止- IC_RAW_INTR_STAT.TX_ABRT 被置位

位	符号	访问	描述
31:30	-	R	保留
2:0	RXFLR	R	接收缓冲器内的数据个数。

26.5.32　I2C SDA 保持寄存器 (SDA_HOLD)

地址偏移量：0x7C
复位值：0x0000 0002
只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:24	-	R	保留
23:16	IC_SDA_RX_HOLD	RW	在接收模式时，设置 SDA 保持时间。以 ic_clk 时钟周期为单位。
15:0	IC_SDA_TX_HOLD	RW	在发送模式时，设置 SDA 保持时间。以 ic_clk 时钟周期为单位。

26.5.33　I2C 发送中止源寄存器 (TX_ABRT_SOURCE)

地址偏移量：0x80
复位值：0x0000 0000

位	符号	访问	描述
31:23	TX_FLUSH_CNT	R	由于 TX_ABRT 中断的发生, 丢掉的 Tx FIFO 中数据的个数。
22:21	-	R	保留
20	DEVICE_WRITE	R	仅在主机模式使用。当主机发起 DEVICE_ID 传输时，Tx FIFO 中含有其他的命令。
19	DEVICE_SLVADDR_NOACK	R	仅在主机模式使用。当主机发起 DEVICE_ID 传输时，被寻址的从机没有响应。
18	DEVICE_NOACK	R	仅在主机模式使用。当主机发起 DEVICE_ID 传输时，从机没有响应。
17	SDA_STUCK_AT_LOW	R	仅在主机模式使用。SDA 维持低电平 IC_SDA_STUCK_AT_LOW_TIMEOUT 个 ic_clk 周期。
16	USER_ABRT	R	仅在主机模式使用。发现传输中止。
15	SLVRD_INTX	R	仅在从机模式使用。从机收到主机的读请求，但是用户向 IC_DATA_CMD.READ 中写入 1。
14	SLV_ARBLOST	R	仅在从机模式使用。从机正在传输时，总线断开。
13	SLVFLUSH_TXFIFO	R	仅在从机模式使用。从机收到读请求时，Tx FIFO 还有残存有数据。
12	LOST	R	仲裁失败
11	MASTER_DIS	R	仅在主机模式使用。用户要求发起传输时，主机没有被使能。
10	10B_RD_NORSTRT	R	仅在主机模式使用。主机在 10 位地址模式下读取数据时，RESTART 没有被使能。

9	SBYTE_NORSTRT	R	仅在主机模式使用。主机发送 START 时，RESTART 没有被使能。
8	HS_NORSTRT	R	仅在主机模式使用。主机切换成 HS 模式时，RESTART 没有被使能。
7	SBYTE_ACKDET	R	仅在主机模式使用。主机发送 START 时，收到了 ACK。
6	HS_ACKDET	R	仅在主机模式使用。在 HS 模式下，主机发送 HS 主机码，收到 ACK。
5	GCALL_READ	R	仅在主机模式使用。GCALL 后面紧接着是读命令。
4	GCALL_NOACK	R	仅在主机模式使用。发送 GCALL 时，没有从机响应。
3	TXDATA_NOACK	R	仅在主机模式使用。发送数据时，没有收到被寻址从机的响应。
2	10ADDR2_NOACK	R	仅在主机模式使用。在 10 位地址模式下，发出的第二个地址字节没有收到任何从机的响应。
1	10ADDR1_NOACK	R	仅在主机模式使用。在 10 位地址模式下，发出的第一个地址字节没有收到任何从机的响应。
0	7B_ADDR_NOACK	R	仅在主机模式使用。在 7 位地址模式下，发出的地址字节没有收到任何从机的响应。

26.5.34　I2C 从机数据 NACK 寄存器 (SLV_DATA_NACK_ONLY)

地址偏移量：0x84

复位值：0x0000 0000

只有当 IC_ENABLE[0] 和 IC_STATUS[6] 都为 0 时，该寄存器可以写。

位	符号	访问	描述
31:1	-	R	保留
0	NACK	RW	产生 NACK，高有效。仅在从接收器模式使用。 1：收到数据时，只能回复 NACK。数据传输被中止，收到的数据不存入接收缓冲器。 0: 正常功能。

26.5.35　I2C DMA 控制寄存器 (DMA_CR)

地址偏移量：0x88

复位值：0x0000 0000

位	符号	访问	描述
31:2	-	R	保留
1	TDMAE	RW	发送 DMA 使能位。高有效。
0	RDMAE	RW	接收 DMA 使能位。高有效。

26.5.36　I2C DMA 发送数据水平寄存器 (DMA_TDLR)

地址偏移量：0x8C

复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留
2:0	DMATDL	RW	DMA 发送数据水平，控制何时产生发送 DMA 申请。当发送缓冲器内的命令少于或者等于 DMATDL 时，产生 DMA 申请。

26.5.37　I2C DMA 接收数据水平寄存器 (DMA_RDLR)

地址偏移量：0x90

复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留
2:0	DMARDL	RW	DMA 接收数据水平，控制何时产生发送 DMA 申请。当接收缓冲器内的数据大于 DMARDL 时，产生 DMA 申请。

26.5.38　I2C SDA 建立寄存器 (SDA_SETUP)

地址偏移量：0x94

复位值：0x0000 0064

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:8	-	R	保留
7:0	SDA_SETUP	RW	SDA 建立时间，以 ic_clk 周期为单位，仅在从发送器模式时使用。硬件应用时将该值 +1。

26.5.39　I2C ACK 通用呼叫 (ACK_GENERAL_CALL)

地址偏移量：0x98

复位值：0x0000 0001

位	符号	访问	描述
31:1	-	R	保留
0	ACK_GEN_CALL	RW	控制收到通用呼叫后，响应 ACK/NACK。仅在从机模式时使用。 1: 响应 ACK 0: 响应 NACK

26.5.40　I2C 使能状态寄存器 (ENABLE_STATUS)

地址偏移量：0x9C

复位值：0x0000 0000

位	符号	访问	描述
31:3	-	R	保留

2	SLV_RX_DATA_LOST	R	从机接收数据丢失。当该位为 1 时，由于将 IC_ENABLE 写 0，至少造成一个数据丢失。当 IC_EN 为 0 时，用户可以读取该位。
1	SLV_DIS_WHILE_BUSY_LOST	R	在从机模式时，当忙碌时软件禁用该外设。当 IC_EN 为 0 时，用户可以读取该位。
0	IC_EN	R	IC_ENABLE 的状态位。用户将 IC_ENABLE 从 1 写为 0 之后，一段时间之后才可以反映到该状态位。 1: 该外设处于使能状态 0：该外设处于不活动状态

26.5.41　I2C 快速尖峰长度寄存器 (FS_SPKLEN)

地址偏移量：0xA0

复位值：0x0000 00001(I2C1) 0x0000 0005(I2C2)

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:8	-	R	保留
7:0	IC_FS_SPKLEN	RW	在标准速度和快速模式时，SCL 和 SDA 上允许的最大尖峰值。以 ic_clk 周期为单位。在开始 I2C 总线传输之前需要合理配置。

26.5.42　I2C 高速尖峰长度寄存器 (HS_SPKLEN)

地址偏移量：0xA4

复位值：0x0000 00001

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:8	-	R	保留
7:0	IC_HS_SPKLEN	RW	在高速模式时，SCL 和 SDA 上允许的最大尖峰值。以 ic_clk 周期为单位。在开始 I2C 总线传输之前需要合理配置。

26.5.43　I2C 清除 RESTART_DET 中断寄存器 (CLR_RESTART_DET)

地址偏移量：0xA8

复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_RS_DET	R	当用户读取该位时，将清除 RS_DET 中断;

26.5.44　I2C SCL 低电平超时寄存器 (SCL_STUCK_AT_LOW_TIMEOUT)

地址偏移量：0xAC

复位值：0xFFFF FFFF

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:0	SCL_LOWTO	RW	当 SCL 低电平时间大于 SCL_LOWTO 时 (以 ic_clk 的周期为单位)，产生 SCL_STUCK_AT_LOW 中断。

26.5.45　I2C SDA 低电平超时寄存器 (SDA_STUCK_AT_LOW_TIMEOUT)

地址偏移量：0xB0

复位值：0xFFFF FFFF

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

位	符号	访问	描述
31:0	SDA_LOWTO	RW	当 SDA 低电平时间大于 SDA_LOWTO 时 (以 ic_clk 的周期为单位)，如果 IC_ENABLE[3] 为 1，开始恢复 SDA。

26.5.46　I2C 清除 SCL_STUCK_DET 中断寄存器 (SCL_STUCK_DET)

地址偏移量：0xB4

复位值：0x0000 0000

位	符号	访问	描述
31:30	-	R	保留
0	CLR_SCL_STUCK_DET	R	当用户读取该位时，将清除 SCL_STUCK_DET 中断;

26.5.47　SMBUS 从模式时钟延伸超时寄存器 (SMBUS_CLK_LOW_SEXT)

地址偏移量：0xBC

复位值：0xFFFF FFFF

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:0	SMBUS_CLK_LOW_SEXT_TO	RW	定义参数 tLOW:SEXT，详情请参考 SMBUS 协议。以 i2c_clk 周期为单位。

26.5.48　SMBUS 主模式时钟延伸超时寄存器 (SMBUS_CLK_LOW_MEXT)

地址偏移量：0xC0

复位值：0xFFFF FFFF

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述

31:0	SMBUS_CLK_LOW_MEXT_TO	RW	定义参数 tLOW:MEXT，详情请参考 SMBUS 协议。以 i2c_clk 周期为单位。

26.5.49　SMBUS 总线空闲寄存器 (SMBUS_THIGH_MAX_BUS_IDLE_CNT)

地址偏移量：0xC4

复位值：0x0000 FFFF

只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:16	-	R	保留
15:0	SMBUS_MBI_CNT	RW	定义总线上的最大空闲时间。以 i2c_clk 周期为单位。

26.5.50　SMBUS 中断状态寄存器 (SMBUS_INTR_STAT)

地址偏移量：0xC8

复位值：0x0000 0000

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:11	-	R	保留
10	SMBUS_ALERT_DET	R	中断 SMBUS_ALERT_DET 的状态，高有效。
9	SMBUS_SUSPEND_DET	R	中断 SMBUS_SUSPEND_DET 的状态，高有效。
8	SLV_RX_PEC_NACK	R	中断 SLV_RX_PEC_NACK 的状态，高有效。
7	ARP_ASSGN_ADDR_CMD_DET	R	中断 ARP_ASSGN_ADDR_CMD_DET 的状态，高有效。
6	ARP_GET_UDID_CMD_DET	R	中断 ARP_GET_UDID_CMD_DET 的状态，高有效。
5	ARP_RST_CMD_DET	R	中断 ARP_RST_CMD_DET 的状态，高有效。
4	ARP_PREPARE_CMD_DET	R	中断 ARP_PREPARE_CMD_DET 的状态，高有效。
3	HOST_NOTIFY_MST_DET	R	中断 HOST_NOTIFY_MST 的状态，高有效。
2	QUICK_CMD_DET	R	中断 QUICK_CMD_DET 的状态，高有效。
1	MCLKEXT_TIMEOUT	R	中断 MST_CLOCK_EXTND_TIMEOUT 的状态，高有效。
0	SCLKEXT_TIMEOUT	R	中断 SLV_CLOCK_EXTND_TIMEOUT 的状态，高有效。

26.5.51　SMBUS 中断屏蔽寄存器 (SMBUS_INTR_MASK)

地址偏移量：0xCC

复位值：0x0000 0000

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:11	-	R	保留
10	M_SMBUS_ALERT_DET	R	中断 SMBUS_ALERT_DET 的屏蔽位，写 0 屏蔽中断。

9	M_SMBUS_SUSPEND_DET	R	中断 SMBUS_SUSPEND_DET 的屏蔽位，写 0 屏蔽中断。
8	M_SLV_RX_PEC_NACK	R	中断 SLV_RX_PEC_NACK 的屏蔽位，写 0 屏蔽中断。
7	M_ARP_ASSGN_ADDR_CMD_DET	R	中断 ARP_ASSGN_ADDR_CMD_DET 的屏蔽位，写 0 屏蔽中断。
6	M_ARP_GET_UDID_CMD_DET	R	中断 ARP_GET_UDID_CMD_DET 的屏蔽位，写 0 屏蔽中断。
5	M_ARP_RST_CMD_DET	R	中断 ARP_RST_CMD_DET 的屏蔽位，写 0 屏蔽中断。
4	M_ARP_PREPARE_CMD_DET	R	中断 ARP_PREPARE_CMD_DET 的屏蔽位，写 0 屏蔽中断。
3	M_HOST_NOTIFY_MST_DET	R	中断 HOST_NOTIFY_MST 的屏蔽位，写 0 屏蔽中断。
2	M_QUICK_CMD_DET	R	中断 QUICK_CMD_DET 的屏蔽位，写 0 屏蔽中断。
1	M_MCLKEXT_TIMEOUT	R	中断 MST_CLOCK_EXTND_TIMEOUT 的屏蔽位，写 0 屏蔽中断。
0	M_SCLKEXT_TIMEOUT	R	中断 SLV_CLOCK_EXTND_TIMEOUT 的屏蔽位，写 0 屏蔽中断。

26.5.52　SMBUS 原始中断状态寄存器 (SMBUS_RAW_INTR_STAT)

地址偏移量：0xD0

复位值：0x0000 0000

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:11	-	R	保留
10	SMBUS_ALERT_DET	R	中断 SMBUS_ALERT_DET 的原始状态，高有效。
9	SMBUS_SUSPEND_DET	R	中断 SMBUS_SUSPEND_DET 的原始状态，高有效。
8	SLV_RX_PEC_NACK	R	中断 SLV_RX_PEC_NACK 的原始状态，高有效。
7	ARP_ASSGN_ADDR_CMD_DET	R	中断 ARP_ASSGN_ADDR_CMD_DET 的原始状态，高有效。
6	ARP_GET_UDID_CMD_DET	R	中断 ARP_GET_UDID_CMD_DET 的原始状态，高有效。
5	ARP_RST_CMD_DET	R	中断 ARP_RST_CMD_DET 的原始状态，高有效。
4	ARP_PREPARE_CMD_DET	R	中断 ARP_PREPARE_CMD_DET 的原始状态，高有效。
3	HOST_NOTIFY_MST_DET	R	中断 HOST_NOTIFY_MST 的原始状态，高有效。
2	QUICK_CMD_DET	R	中断 QUICK_CMD_DET 的原始状态，高有效。
1	MCLKEXT_TIMEOUT	R	中断 MST_CLOCK_EXTND_TIMEOUT 的原始状态，高有效。
0	SCLKEXT_TIMEOUT	R	中断 SLV_CLOCK_EXTND_TIMEOUT 的原始状态，高有效。

26.5.53　SMBUS(CLR_SMBUS_INTR)

地址偏移量：0xD4

复位值：0x0000 0000

该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:11	-	R	保留
10	SMBUS_ALERT_DET	W	中断 SMBUS_ALERT_DET 的写清除位。写 1 清除中断，写 0 无效。
9	SMBUS_SUSPEND_DET	W	中断 SMBUS_SUSPEND_DET 的写清除位。写 1 清除中断，写 0 无效。
8	SLV_RX_PEC_NACK	W	中断 SLV_RX_PEC_NACK 的写清除位。写 1 清除中断，写 0 无效。
7	ARP_ASSGN_ADDR_CMD_DET	W	中断 ARP_ASSGN_ADDR_CMD_DET 的写清除位。写 1 清除中断，写 0 无效。
6	ARP_GET_UDID_CMD_DET	W	中断 ARP_GET_UDID_CMD_DET 的写清除位。写 1 清除中断，写 0 无效。
5	ARP_RST_CMD_DET	W	中断 ARP_RST_CMD_DET 的写清除位。写 1 清除中断，写 0 无效。
4	ARP_PREPARE_CMD_DET	W	中断 ARP_PREPARE_CMD_DET 的写清除位。写 1 清除中断，写 0 无效。
3	HOST_NOTIFY_MST_DET	W	中断 HOST_NOTIFY_MST 的写清除位。写 1 清除中断，写 0 无效。
2	QUICK_CMD_DET	W	中断 QUICK_CMD_DET 的写清除位。写 1 清除中断，写 0 无效。
1	MCLKEXT_TIMEOUT	W	中断 MST_CLOCK_EXTND_TIMEOUT 的写清除位。写 1 清除中断，写 0 无效。
0	SCLKEXT_TIMEOUT	W	中断 SLV_CLOCK_EXTND_TIMEOUT 的写清除位。写 1 清除中断，写 0 无效。

26.5.54　I2C 可选从机地址寄存器 (OPTIONAL_SAR)

地址偏移量：0xD8
复位值：0x0000 0000
只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。
该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:7	-	R	保留
6:0	OPTIONAL_SAR	RW	SMBUS 从模式使用。

26.5.55　SMBUS(SMBUS_UDID_LSB)

地址偏移量：0xDC
复位值：0xFFFF FFFF
只有当 IC_ENABLE[0] 为 0 时，该寄存器可以写。
该寄存器在 I2C2 中不存在，仅在 I2C1 中存在

位	符号	访问	描述
31:0	SMBUS_UDID_LSB	RW	SMBUS UDID 最后 32 位。

26.5.56　I2C1 寄存器图

下表列出了 I2C1 的寄存器映像和复位值。

表 26.60: I2C1 的寄存器映像表

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CON	保留												SMBUS_PERS_SLV_ADDR_EN	SMBUS_ARP_EN	SMBUS_SLV_QUICK_EN	OP_SAR_CTRL	保留				BUS_CLR_CTRL	STOP_DET_IF_MST_ACTIVE	RX_FIFO_FULL_HLD_CTRL	TX_EMP_CTRL	STOP_DET_IFADDRESSED	IC_SLV_DISABLE	IC_RS_EN	IC_10ADDR_MST	IC_10ADDR_SLV	SPEED		MST_MODE
	复位值													0	0	0	0					0	0	0	0	0	0	1	0	0	1	0	0
0x04	TAR	保留															SMBUS_QUICK_CMD	保留		DEVICE_ID	保留	SPECIAL	GC_OP_START	TAR									
	复位值																0			0		0	0	0x55									
0x08	SAR	保留																						SAR									
	复位值																							0x3F0									
0x10	DATA_CMD	保留																				FIRST_DATA_BYTE	RESTART	STOP	READ	DAT							
	复位值																					0	0	0	0	0							
0x14	SS_SCL_HCNT	保留																IC_SS_SCL_HCNT															
	复位值																	0x0320															
0x18	SS_SCL_LCNT	保留																IC_SS_SCL_LCNT															
	复位值																	0x03AC															
0x1C	FS_SCL_HCNT	保留																IC_FS_SCL_HCNT															
	复位值																	0x0078															
0x20	FS_SCL_LCNT	保留																IC_FS_SCL_LCNT															
	复位值																	0x0104															
0x2C	IC_INTR_STAT	保留															R_SCL_STUCK_AT_LOW	保留		R_RS_DET	R_GEN_CALL	R_START_DET	R_STOP_DET	R_ACTIVITY	R_RX_DONE	R_TX_ABRT	R_RD_REQ	R_TX_EMPTY	R_TX_OVER	R_RX_FULL	R_RX_OVER	R_RX_UNDER
	复位值																0			0	0	0	0	0	0 0	0	0	0	0	0	0	
0x30	IC_INTM_MASK	保留															M_SCL_STUCK_AT_LOW	保留		M_RS_DET	M_GEN_CALL	M_START_DET	M_STOP_DET	M_ACTIVITY	M_RX_DONE	M_TX_ABRT	M_RD_REQ	M_TX_EMPTY	M_TX_OVER	M_RX_FULL	M_RX_OVER	M_RX_UNDER
	复位值																0			0	0	0	0	0	0 0	0	0	0	0	0	0	
0x34	IC_RAW_INTR_STAT	保留															SCL_STUCK_AT_LOW	保留		R_RS_DET	R_GEN_CALL	R_START_DET	R_STOP_DET	R_ACTIVITY	R_RX_DONE	R_TX_ABRT	R_RD_REQ	R_TX_EMPTY	R_TX_OVER	R_RX_FULL	R_RX_OVER	R_RX_UNDER
	复位值																0			0	0	0	0	0	0 0	0	0	0	0	0	0	
0x38	IC_RX_TL	保留																						RX_TL									
	复位值																							0x0005									
0x3C	IC_TX_TL	保留																						TX_TL									
	复位值																							0x0005									

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x40	IC_CLR_INTR	保留																															CLR_INTR
	复位值																																0
0x44	IC_CLR_RX_UNDER	保留																															CLR_RX_UNDER
	复位值																																0
0x48	IC_CLR_RX_OVER	保留																															CLR_RX_OVER
	复位值																																0
0x4C	IC_CLR_TX_OVER	保留																															CLR_TX_OVER
	复位值																																0
0x50	IC_CLR_RD_REQ	保留																															CLR_RD_REQ
	复位值																																0
0x54	IC_CLR_TX_ABRT	保留																															CLR_TX_ABRT
	复位值																																0
0x58	IC_CLR_RX_DONE	保留																															CLR_RX_DONE
	复位值																																0
0x5C	IC_CLR_ACTIVITY	保留																															CLR_ACTIVITY
	复位值																																0
0x60	IC_CLR_STOP_DET	保留																															CLR_STOP_DET
	复位值																																0
0x64	IC_CLR_START_DET	保留																															CLR_START_DET
	复位值																																0
0x68	IC_CLR_GEN_CALL	保留																															CLR_GEN_CALL
	复位值																																0
0x6C	IC_ENABLE	保留													SM_ALTER_EN	SM_SUSPEND_EN	SM_CLK_RST	保留													SDA_STUCK_RC_EN	TX_CMD_BLOCK	ABORT	ENABLE
	复位值														0	0	0													0	0	0	0
0x70	IC_STATUS	保留											SM_ALTER_STATUS	SM_SUSPEND_STATUS	SM_SLV_ADDR_RESOLVED	SM_SLV_ADDR_VLD	SM_QUICK_CMD_BIT					SDA_STUCK_NOT_RECOVERED	SLV_HOLD_RX_FIFO_FULL	SLV_HOLD_TX_FIFO_EMPTY	MST_HOLD_RX_FIFO_FULL	MST_HOLD_TX_FIFO_EMPTY	SLV_ACTIVITY	MST_ACTIVITY	REF	RFNE	TFE	TFNF	ACTIVITY

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	复位值												0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0
0x74	IC_TXFLR	保留																													TXFLR		
	复位值																														0		
0x78	IC_RXFLR	保留																													RXFLR		
	复位值																														0		
0x7C	IC_SDA_HOLD	保留								IC_SDA_RX_HOLD								IC_SDA_TX_HOLD															
	复位值									0								0x2															
0x80	IC_TX_ABRT_SOURCE	TX_FLUSH_CNT								保留			DEVICE_WRITE	DEVICE_SLVADDR_NOACK	DEVICE_NOACK	SDA_STUCK_AT_LOW	USER_ABRT	SLVRD_INTX	ALV_ARBLOST	SLVFLUSH_TXFIFO	LOST	MASTER_DIS	10B_RD_NORSTRT	SBYTE_NORSTRT	HS_NORSTRT	SBYTE_ACKDET	HS_ACKDET	GCALL_READ	GCALL_NOACK	TXDATA_NOACK	10ADDR2_NOACK	10ADDR1_NOACK	7B_ADDR_NOACK
	复位值	0											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x84	IC_SLV_DATA_NACK_ONLY	保留																															NACK
	复位值																																0
0x88	IC_DMA_CR	保留																														TDMAE	RDMAE
	复位值																															0	0
0x8C	IC_DMA_TDLR	保留																													DMATDL		
	复位值																														0		
0x90	IC_DMA_RDLR	保留																													DMARDL		
	复位值																														0		
0x94	IC_SDA_SETUP	保留																													SDA_SETUP		
	复位值																														0		
0x98	IC_ACK_GENERAL_CALL	保留																															ACK_GEN_CALL
	复位值																																0
0x9C	IC_ENABLE_STATUS	保留																													SLV_RX_DATA_LOST	SLV_DIS_WHILE_BUSY_LOST	IC_EN
	复位值																														0	0	0
0xA0	IC_FS_SPKLEN	保留																													IC_FS_SPKLEN		
	复位值																														0		
0xA4	IC_HS_SPKLEN	保留																													IC_HS_SPKLEN		
	复位值																														0		
0xA8	IC_CLR_RESTART_DET	保留																															CLR_RS_DET
	复位值																																0
0xAC	IC_SCL_STUCK_AT_LOW_TIMOUT	SCL_LOWTO																															
	复位值	0																															
0xB0	IC_SDA_STUCK_AT_LOW_TIMOUT	SDA_LOWTO																															
	复位值	0																															
0xB4	SCL_STUCK_DET	保留																															CLR_SCL_STUCK_DET
	复位值																																0
0xBC	IC_SMBUS_SCK_LOW_SEXT	SMBUS_SCK_LOW_SEXT_TO																															
	复位值	0xFFFF FFFF																															
0xC0	IC_SMBUS_SCK_LOW_MEXT	SMBUS_SCK_LOW_MEXT_TO																															
	复位值	0xFFFF FFFF																															
0xC4	IC_SMBUS_THIGH_MAX_BUS_IDLE_CNT	SMBUS_MBI_CNT																															
	复位值																	0xFFFF															

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xC8	IC_SMBUS_INTR_STAT	保留																					SMBUS_ALERT_DET	SMBUS_SUSPEND_DET	SLV_RX_PEC_NACK	ARP_ASSGN_ADDR_CMD_DET	ARP_GET_UDID_CMD_DET	ARP_RST_CMD_DET	ARP_PREPARE_CMD_DET	HOST_NOTIFY_MST_DET	QUICK_CMD_DET	MCLKEXT_TIMEOUT	SCLKEXT_TIMEOUT
	复位值																						0	0	0	0	0	0	0	0	0	0	0
0xCC	IC_SMBUS_INTR_MASK	保留																					M_SMBUS_ALERT_DET	M_SMBUS_SUSPEND_DET	M_SLV_RX_PEC_NACK	M_ARP_ASSGN_ADDR_CMD_DET	M_ARP_GET_UDID_CMD_DET	M_ARP_RST_CMD_DET	M_ARP_PREPARE_CMD_DET	M_HOST_NOTIFY_MST_DET	M_QUICK_CMD_DET	M_MCLKEXT_TIMEOUT	M_SCLKEXT_TIMEOUT
	复位值																						0	0	0	0	0	0	0	0	0	0	0
0xD0	IC_SMBUS_RAW_INTR_STAT	保留																					SMBUS_ALERT_DET	SMBUS_SUSPEND_DET	SLV_RX_PEC_NACK	ARP_ASSGN_ADDR_CMD_DET	ARP_GET_UDID_CMD_DET	ARP_RST_CMD_DET	ARP_PREPARE_CMD_DET	HOST_NOTIFY_MST_DET	QUICK_CMD_DET	MCLKEXT_TIMEOUT	SCLKEXT_TIMEOUT
	复位值																						0	0	0	0	0	0	0	0	0	0	0
0xD4	IC_CLR_SMBUS_INTR	保留																					SMBUS_ALERT_DET	SMBUS_SUSPEND_DET	SLV_RX_PEC_NACK	ARP_ASSGN_ADDR_CMD_DET	ARP_GET_UDID_CMD_DET	ARP_RST_CMD_DET	ARP_PREPARE_CMD_DET	HOST_NOTIFY_MST_DET	QUICK_CMD_DET	MCLKEXT_TIMEOUT	SCLKEXT_TIMEOUT
	复位值																						0	0	0	0	0	0	0	0	0	0	0
0xD8	IC_OPTIONAL_SAR	保留																					OPTIONAL_SAR										
	复位值																						0										
0xDC	IC_SMBUS_UDID_LSB	SMBUS_UDID_LSB																															
	复位值	0xFFFF FFFF																															

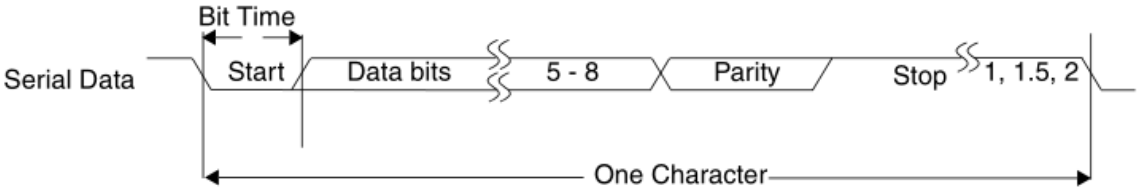
第二十七章　通用异步收发器 (UART)

27.1　简介

27.1.1　RS232 串行通讯协议

UART 端口支持 RS232 串行通信协议，协议的格式如下图，分为起始位，数据位，奇偶校验位和停止位，其中奇偶校验位是一个可选项，停止位可以是 1 个，1.5 个或者 2 个位宽。

图 27.1: RS232 串行数据格式



27.1.2　9 位数据传输

UART 支持 9 位数据传输模式，第九个位在数据的第 8 位之后，并且在奇偶校验位和停止位之前，下图显示了 9 位传输的格式。

图 27.2: 9 位传输数据格式

通过使能 9 位传输模式，系统可以支持一个主控制器跟多个从设备的连接。当主控制器需要跟一个从设备进行通讯时，它会先发一个地址数据选择需要通讯的从设备。地址和数据的区别在于第 9 位，当第 9 位设置成 “1”，这时传输的数据代表从设备的地址；当第 9 位设置成 “0”，这时传输的是数据。当接收到地址时，所有的从设备会跟自己的地址进行比较，如果地址匹配，那么该设备就会继续接收后面的数据；如果不匹配，那么从设备会忽略后面的数据直到接收到新的地址。

在接收的时候，可以通过硬件来自动匹配接收到的地址，也可以通过软件来对收到的进行匹配。

27.1.3　分数波特率

UART 支持分数波特率的配置来降低在高速率传输中的误码率，可以确保 UART 的波特率误差在 2% 以内。影响波特率的因素有：

- UART 模块时钟的工作频率 F_{uart}。
- 期望的波特率 Baud Rate。
- 波特率发生寄存器 DIVISOR 的值。
- 可接受的波特率误差

波特率的计算公式如下：

$$Baud\ Rate = \frac{F_{uart}}{16 \times DIVISOR} \tag{27.1}$$

其中，DIVISOR 是由 DLH 和 DLL 寄存器组合而成。
根据这个公式，可以得到计算 DIVISOR 的公式：

$$DIVISOR = \frac{F_{uart}}{16 \times Baud\ Rate} \tag{27.2}$$

除数的小数部分可以由保存在 DLF 寄存器的值计算出来：

表 27.1: 除数锁存器对应的小数值

DLF 值	分数	小数值
0000	0/16	0.0000
0001	1/16	0.0625
0010	2/16	0.125
0011	3/16	0.1875
0100	4/16	0.25
0101	5/16	0.3125
0110	6/16	0.375
0111	7/16	0.4375
1000	8/16	0.5
1001	9/16	0.5625
1010	10/16	0.625
1011	11/16	0.6875
1100	12/16	0.75
1101	13/16	0.8125
1110	14/16	0.875
1111	15/16	0.9375

可编程的分数波特率相对于传统的整数波特率发生器可以产生更高精度的波特率。这样可以允许对波特率的整数和分数部分继续编程。其公式如下：

$$Baud\ Rate\ Divisor = \frac{Fuart}{16 \times Baud\ Rate} = BRD_1 + BRD_2 \tag{27.3}$$

其中，

BRD_1 除数的整数部分。

BRD_2 除数的分数部分。

27.1.4　IrDA SIR 协议

UART 支持 IrDA 1.0 SIR 模式通过红外线进行双向数据传输，最高波特率可以达到 115.2Kbps。数据格式根标准的串行通讯格式一样，每个数据字节按照以下顺序发送：

1. 以一个起始位开始
2. 发送 8 位数据
3. 以一个结束位结束

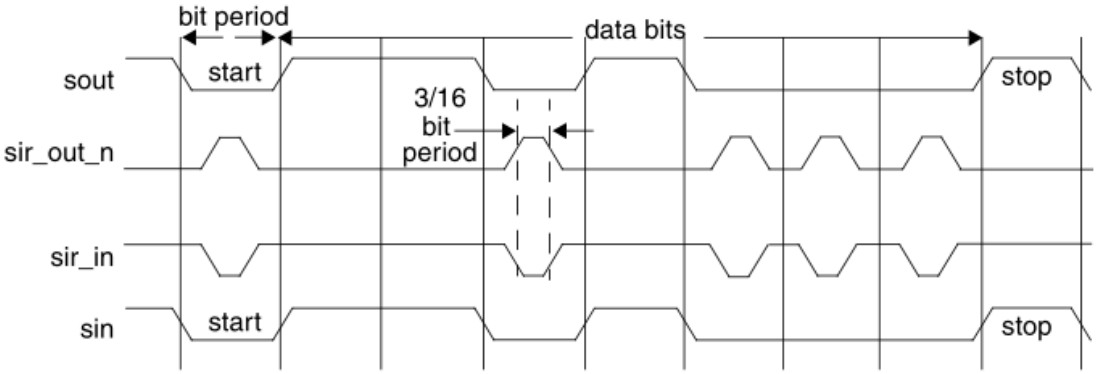
这样，可以发送的数据宽度是固定的，不支持奇偶校验位的发送，并且只有一个结束位。

发送或不发送一个红外脉冲代表的含义如下：

- 发送一个红外脉冲代表逻辑 0
- 不发送一个红外脉冲代表逻辑 1

每个脉冲的宽度是 3/16 的正常位宽，因此，每个新的字节传输是以一个红外脉冲开始的。然而，接收数据跟发送数据的极性是相反的，这个主要是由于红外接收器的特性决定的。下图显示了红外数据的格式。

图 27.3: 红外传输数据格式



27.1.5　自动流控制

如果自动 RTS 模式启用，则 UART 接收器 FIFO 硬件可控制 UART 的 RTS 输出。如果自动 CTS 模式启用，则只有在 CTS 输入信号有效时，UART TSR 硬件才启动发送。

自动 RTS

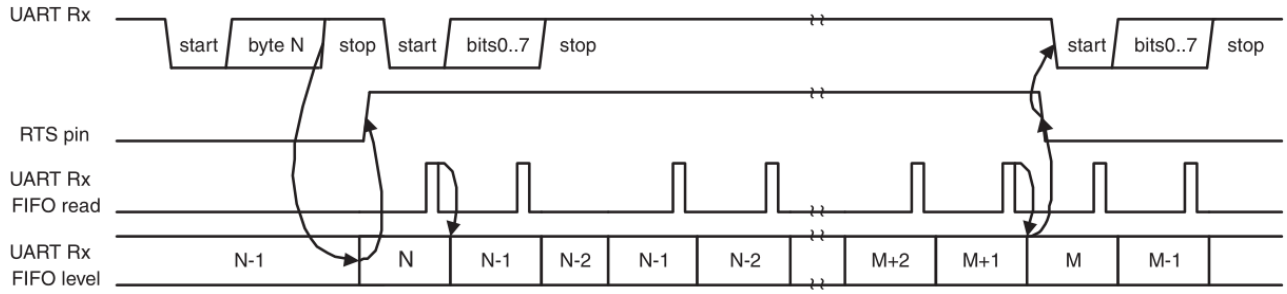
通过设置 AFCE 位可以启用自动 RTS 功能。自动 RTS 数据流控制在 RBR 模块中产生，并链接至已编程好的接收器 FIFO 触发阈值。如果自动 RTS 启用，则按照以下方式对数据流进行控制：

当接收器 FIFO 的值达到已编程好的触发阈值时，RTS 失效（变为高电平值）。发送 UART 在达到触发阈值后，可能会发送一个额外字节（假设发送 UART 有额外字节要发送），因为它可能直到开始发送额外字

节后才知道 RTS 失效。一旦接收器 FIFO 达到先前的触发阈值，RTS 就会自动重新生效（变为低电平值）。发送 UART 的 RTS 信号重新生效后可继续发送数据。

如果自动 RTS 模式被禁用，则 RTS 位可控制 UART 的 RTS 输出。如果自动 RTS 模式启用，则硬件可控制 RTS 输出，而且可以将 RTS 的实际值复制到 UART 的 RTS 控制位。如果自动 RTS 启用，则只有软件可对 RTS 控制位的值进行只读操作。

图 27.4: 自动 RTS 功能时序图



自动 CTS

通过设置 AFCE 位可以启用自动 CTS 功能。如果自动 CTS 启用，则发送器电路将在发送下一个数据字节之前检查 CTS 输入。当 CTS 有效（低电平）时，发送器发送下一个字节。为了让发送器停止发送后面的字节，必须在当前正在发送的最后一个停止位发送一半以前释放 CTS。在自动 CTS 模式下，CTS 信号的变化不会触发调制解调器状态中断，除非对 CTS 中断使能位进行设置，不过将会设置 MSR 中的 Delta CTS 位。

自动 CTS 功能可减少主机系统的中断。当流控制启用时，CTS 状态的改变不会触发主机中断，因为器件会自动控制其发送器。若不使用自动 CTS，则发送器会将发送 FIFO 中存在的所有数据都发送出去，从而导致接收器发生溢出错误。图27.5 展示了自动 CTS 功能时序。

图 27.5: 自动 CTS 功能时序图

27.1.6　DMA 操作

通过使用微型 DMA，用户可选择操作 UART 的发送和 / 或接收。DMA 模式由 FCR 寄存器中的 DMA 模式选择位决定。只有通过 FCR 寄存器中的 FIFO 使能位将 FIFO 启用时，该位才会有效。

UART 接收器 DMA

在 DMA 模式中，当接收器 FIFO 的电平等于或大于触发电平时，或者在发生字符超时的情况下，接收器 DMA 请求就会生效。请参考上文对 RX 触发电平的描述。接收器 DMA 请求由 DMA 控制器清除。

UART 发送器 DMA

在 DMA 模式中，当发送器 FIFO 变为未满时，发送器 DMA 请求就会生效。发送器 DMA 请求由 DMA 控制器清除。

27.2　UART 寄存器

27.2.1　接收器缓冲寄存器 (UART_RBR)（当 DLAB = 0 时）

地址偏移量：0x000
复位值：NA

表 27.2: 接收器缓冲寄存器 (UART_RBR) 位描述

位	符号	访问	描述
31:9	-	R	保留
8:0	RBR	R	UART 接收器缓冲寄存器包含了 UART RX FIFO 中最早接收到的字节。

RBR 是 UART RX FIFO 的最高字节。RX FIFO 的最高字节包含最早接收到的字符，并可通过总线接口进行读取。LSB （位 0）表示接收的数据位。如果接收到的字符少于 9 位，则未使用的 MSB 用 0 填充。

如果要访问 RBR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。RBR 始终为只读。

由于奇偶错误 (PE)、帧错误 (FE) 和间隔中断 (BI) 位与 RBR FIFO 顶部的字节（即下次从 RBR 读取时要读取的字节）相对应，因此，要正确提取有效的接收字节对及其状态位，应先读取 LSR 寄存器的内容，然后再读取 RBR 中的字节。

27.2.2　发送器保持寄存器 (UART_THR)（当 DLAB = 0 时）

地址偏移量：0x000
复位值：NA

表 27.3: 发送器保持寄存器 (UART_THR) 位描述

位	符号	访问	描述
31:9	-	R	保留
8:0	THR	W	写入 UART 发送保持寄存器会使数据保存到 UART 发送 FIFO 中。当字节达到 FIFO 的底部并且发送器可用时，字节就会被发送。

THR 是 UART TX FIFO 的最高字节。最高字节是 TX FIFO 中的最新字符，可通过总线接口进行写入。LSB 代表第一个将要发送的位。

如果要访问 THR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。THR 始终为只写。

27.2.3　除数锁存器 LSB 寄存器 (UART_DLL)（当 DLAB = 1 时）

地址偏移量：0x000
复位值：0x000

表 27.4: 除数锁存器 LSB 寄存器 (UART_DLL) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:0	DLLSB	RW	UART 除数锁存器 LSB 寄存器与 DLH 寄存器一起决定 UART 的波特率。

27.2.4　除数锁存器 MSB 寄存器 (UART_DLH)（当 DLAB = 1 时）

地址偏移量：0x004
复位值：0x000

表 27.5: 除数锁存器 MSB 寄存器 (UART_DLH) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:0	DLHSB	RW	UART 除数锁存器 LSB 寄存器与 DLH 寄存器一起决定 UART 的波特率。

UART 除数锁存器是 UART 波特率生成器的一部分，并保存使用的值，它与小数分频器一同使用，对 UART_PCLK 时钟进行分频，以产生波特率时钟，波特率时钟必须是所需波特率的 16 倍。DLL 和 DLH 寄存器一起构成了一个 16 位除数，其中 DLL 包含了除数的低 8 位，而 DLH 包含了除数的高 8 位。0x0000 值会被作为 0x0001 值处理，因为除数不能为 0。如果要访问 UART 除数锁存器，LCR 中的除数锁存器访问位 (DLAB) 必须为 1。

27.2.5　中断使能寄存器 (UART_IER)（当 DLAB = 0 时）

地址偏移量：0x004
复位值：0x000
描述：IER 用于启用 4 个 UART 中断源。

表 27.6: 中断使能寄存器 (UART_IER) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	PTIME	RW	控制可编程 THRE 中断。 0: 禁用可编程 THRE 中断。 1: 启用可编程 THRE 中断。
6:5	-	R	保留。
4	LSRCLRMD	RW	该位用于控制 LSR 寄存器中状态位的清除方式。 只适用于 OE, PE, FE 和 BI 状态位。0: 读取 RBR 或 LSR 寄存器时，清除 LSR 状态位。 1: 仅在读取 LSR 寄存器时，清除 LSR 状态位。
3	MSIE	RW	调制解调器中断控制。 0: 禁用调制解调器中断。 1: 使能调制解调器中断。

2	RLSIE	RW	RX 线中断启用。启用 UART RX 线状态中断。该中断的状态可从 LSR[4:1] 中读取。 0: 禁用 RX 线状态中断。 1: 启用 RX 线状态中断。
1	THREIE	RW	THRE 中断使能。启用 UART 的 THRE 中断。该中断的状态可从 LSR[5] 中读取。 0: 禁用 THRE 中断。 1: 启用 THRE 中断。
0	RDAIE	RW	RBR 中断使能。启用 UART 的接收数据可用中断。它还控制着字符接收超时中断。 0: 禁用 RDA 中断。 1: 启用 RDA 中断。

27.2.6 中断识别寄存器 (UART_IIR)

地址偏移量：0x008

复位值：0x001

表 27.7: 中断识别寄存器 (UART_IIR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:6	FIFOSE	R	FIFO 使能控制。 00: FIFO 禁用。 11: FIFO 启用。
5:4	-	R	保留，用户软件不应对保留位写入 1。
3：0	INTID	R	中断识别。IER[3:0] 可识别挂起的最高优先级中断。下面未列出的 IER[3:1] 的其他组合均为保留值（100、101、111）。 0xC：字节超时中断。 0x7：忙状态检测中断。 0x6：接收线状态中断。 0x4：接收数据可用中断。 0x2：THR 空中断。 0x1：没有挂起的中断。 0x0: 调制解调器中断。

表 27.8: UART 中断处理

IIR[3:0]	优先级	中断类型	中断源	中断复位
0001	-	无	无	-
0110	最高	RX 线状态/错误	OE、PE、FE 或 BI	LSR 读操作

0100	第二	RX 数据可用	RX 数据可用或达到 FIFO 触发电平 (FCR0=1)	RBR 读操作或 UART FIFO 低于触发电平
1100	第二	字符超时指示	RX FIFO 中至少有一个字符，并且在一段时间内没有字符输入或移出，该时间的长短取决于 FIFO 中的字符数以及在 3.5 到 4.5 字符的时间内的触发值。 实际时间为： [（字长度）× 7 - 2] × 8 + [（触发电平 - 字符数）× 8 + 1] RCLK	RBR 读操作
0010	第三	THRE	THRE	IIR 读操作或 THR 写操作

27.2.7　FIFO 控制寄存器 (UART_FCR)

地址偏移量：0x008
复位值：0x000

表 27.9: FIFO 控制寄存器 (UART_FCR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:6	RXTL	W	RX 触发电平。这两个位决定了接收器 UART FIFO 在激活中断前必须写入的字符数量。 00: 触发电平 0 （1 个字符或 0x01）。 01: 触发电平 1 （4 个字符或 0x04）。 10: 触发电平 2 （8 个字符或 0x08）。 11: 触发电平 3 （14 个字符或 0x0E）。
5:4	TXTL	W	TX 触发电平。这两个位决定了发送 UART FIFO 在激活空中断前 FIFO 的字符数量。 00: 触发电平 0 （0 个字符或 0x00）。 01: 触发电平 1 （2 个字符或 0x02）。 10: 触发电平 2 （4 个字符或 0x04）。 11: 触发电平 3 （8 个字符或 0x08）。
3	-	R	保留
2	TXFIFORS	W	TX FIFO 复位。 0: 对两个 UART FIFO 均无影响。 1: 写逻辑 1 到 FCR[2] 将会清除 UART TX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。
1	RXFIFORS	W	RX FIFO 复位。 0: 对两个 UART FIFO 均无影响。 1: 写逻辑 1 到 FCR[1] 将会清除 UART RX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。

0	FIFOEN	W	FIFO 启用。 0: UART FIFO 被禁用。 1: 高电平有效，启用对 UART RX、TX FIFO 和 FCR[7:1] 的访问。该位必须进行设置，以实现正确的 UART 操作。该位的任何变化都会自动清除 UART FIFO。

27.2.8 线路控制寄存器 (UART_LCR)

地址偏移量：0x00C
复位值：0x000

表 27.10: 线路控制寄存器 (UART_LCR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	DLAB	RW	除数锁存器访问位 (DLAB)。 0: 禁用对除数锁存器的访问。 1: 启用对除数锁存器的访问。
6	BC	RW	间隔控制。 0: 禁用间隔传输。 1: 启用间隔传输。当 LCR[6] 为高电平有效时，输出引脚 UART TXD 强制为逻辑 0。
5:4	PS	RW	奇偶检验选择。 0x0: 奇检验。发送的字符和附加检验位中逻辑 1 的个数为奇数。 0x1: 偶检验。发送的字符和附加检验位中逻辑 1 的个数为偶数。 0x2: 奇偶检验位强制为 1。 0x3: 奇偶校验位强制为 0。
3	PE	RW	奇偶检验使能。 0: 禁用奇偶检验的生成和检查。 1: 启用奇偶检验的生成和检查。
2	SBS	RW	停止位选择。 0: 1 个停止位。 1: 2 个停止位（如果 LCR[1:0]=00，则为 1.5 个停止位）。
1:0	WLS	RW	字长度选择。 00: 5 位字符长度。 01: 6 位字符长度。 10: 7 位字符长度。 11: 8 位字符长度。

27.2.9 调制解调器控制寄存器 (UART_MCR)

地址偏移量：0x010
复位值：0x000

表 27.11: 调制解调器控制寄存器 (UART_MCR) 位描述

位	符号	访问	描述
31:7	-	R	保留
6	SIRE	RW	IrDA SIR 模式使能。 0: 禁用 IrDA SIR 模式。 1: 启用 IrDA SIR 模式。
5	AFCE	RW	自动流程控制使能。 0: 禁用自动流程控制。 1: 启用自动流程控制。
4:2	-	R	保留
1	RTS	RW	调制解调器输出引脚 RTS 的源。当调制解调器回送模式激活时，该位读为 0。
0	-	R	保留。

27.2.10　线路状态寄存器 (UART_LSR)

地址偏移量：0x014

复位值：0x060

表 27.12: 线路状态寄存器 (UART_LSR) 位描述

位	符号	访问	描述
31:9	-	R	保留
8	ADDR_RCVD	R	地址接收位。在 9bit 数据模式有效时，这个位指接收的第 9 个位。这个位也可以代表接收的是地址还是数据。 0: 接收的是地址。 1: 接收到数据。
7	RFE	R	RX FIFO 错误。当一个带有 RX 错误（如：帧错误、奇偶错误或间隔中断）的字符载入到 RBR 时，LSR[7] 就会被设置。当 LSR 寄存器被读取，且 UART FIFO 中没有后续错误时，此位会被清除。 0: RBR 不包含 UART RX 错误或 FCR[0]=0。 1: UART RBR 包含至少一个 UART RX 错误。
6	TEMT	R	发送器为空。当 THR (或 TX FIFO) 和发送器移位寄存器同时为空时，TEMT 就会被设置；发送器移位寄存器或 THR (或 TX FIFO) 任意一个包含有效数据时，TEMT 就会被清零。 0: THR (或 TX FIFO) 或发送器移位寄存器包含有效数据。 1: THR (或 TX FIFO) 和发送器移位寄存器为空。

5	THRE	R	发送器保持寄存器为空。 当可编程 THRE 中断模式未启用 (IER[7] = 0) 时，不论 FIFO 是否启用，该位指示 THR 或 TX FIFO 是否为空。 0: THR 或 TX FIFO 中包含有效数据。 1: THR 或 TX FIFO 为空。 当可编程 THRE 模式和 FIFO 都启用 (IER[7] = 1 且 FCR[0] = 1) 时，该位指示发送 FIFO 是否已满。 0: TX FIFO 未满。 1: TX FIFO 已满。
4	BI	R	间隔中断。在发送整个字符（起始、数据、奇偶检验、停止）的过程中，如果 RXD1 保持在间隔状态（全 “0”），则会发生间隔中断。一旦检测到间隔条件，接收器会立即进入空闲状态，直到 RXD1 进入标记状态（全 “1”）。LSR 读操作会将该状态位清零。间隔检测的时间取决于 FCR[0]。 注：间隔中断与 UART RBR FIFO 顶部的字符相关。 0: 间隔中断状态无效。 1: 间隔中断状态有效。
3	FE	R	成帧错误。当已接收字符的停止位为逻辑 0 时，会发生成帧错误。LSR 读操作会将 LSR[3] 清零。成帧错误检测时间取决于 FCR0。当检测到有成帧错误时，RX 会尝试与数据重新同步，并假设错误的停止位实际是一个超前的起始位。但是，即使没有出现成帧错误，也无法假设下一个接收到的字节就是正确的。 注：成帧错误与 UART RBR FIFO 顶部的字符相关。 0: 成帧错误状态无效。 1: 成帧错误状态有效。
2	PE	R	奇偶检验错误。当已接收字符的奇偶检验位处于错误状态时，就会发生奇偶检验错误。LSR 读操作会将 LSR[2] 清零。奇偶检验错误检测时间取决于 FCR[0]。 注：奇偶检验错误与 UART RBR FIFO 顶部的字符相关。 0: 奇偶检验错误状态无效。 1: 奇偶检验错误状态有效。
1	OE	R	溢出错误。溢出错误条件在错误发生后立即设置。LSR 读操作会将 LSR[1] 清零。当 UART RSR 有了新的字符组合，且 UART RBR FIFO 已满时，LSR[1] 会被设置。在这种情况下，UART RBR FIFO 不会被覆盖，而 UART RSR 中的字符将会丢失。 0: 溢出错误状态无效。 1: 溢出错误状态有效。
0	DR	R	接收器数据就绪：当 RBR 包含未读字符时，LSR[0] 会被设置；当 UART RBR FIFO 为空时，LSR[0] 会被清零。 0: RBR 为空。 1: RBR 包含有效数据。

27.2.11　调制解调器状态寄存器 (UART_MSR)

地址偏移量：0x018

复位值：0x000

表 27.13: 调制解调器状态寄存器 (UART_MSR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7	DCD	R	数据载波检测状态。输入 DCD 的补码。在调制解调器回送模式下，该位连接到 MCR[3]。
6	RI	R	振铃指示器状态。输入 RI 的补码。在调制解调器回送模式下，该位连接到 MCR[2]。
5	DSR	R	数据集就绪状态。输入信号 DSR 的补码。在调制解调器回送模式下，该位连接到 MCR[0]。
4	CTS	R	清除发送状态。输入信号 CTS 的补码。在调制解调器回送模式下，该位连接到 MCR[1]。
3	DDCD	R	Delta DCD。当输入 DCD 的状态改变时，该位即被设置。MSR 读操作会清除该位。 0: 没有检测到调制解调器输入 DCD 上的状态变化。 1: 检测到调制解调器输入 DCD 上的状态变化。
2	TERI	R	后沿 RI。 当输入 RI 上低电平到高电平转换时，该位即被设置。MSR 读操作会清除该位。 0: 没有检测到调制解调器输入 RI 上的状态变化。 1: 检测到 RI 上低电平到高电平的转换。
1	DDSR	R	Delta DSR。 当输入 DSR 的状态改变时，该位即被设置。MSR 读操作会清除该位。 0: 溢出错误状态无效。 1: 溢出错误状态有效。
0	DCTS	R	Delta CTS。当输入 CTS 的状态改变时，该位被设置。读 MSR 会清除该位。 0: 没有检测到调制解调器输入 CTS 上的状态变化。 1: 检测到调制解调器输入 CTS 上的状态变化。

27.2.12　高速暂时寄存器 (UART_SCR)

地址偏移量：0x01C

复位值：0x000

表 27.14: 高速暂时寄存器 (UART_SCR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7:0	PAD	RW	一个可读、可写的字节。

27.2.13　UART 状态寄存器 (UART_USR)

地址偏移量：0x07C

复位值：0x006

表 27.15: UART 状态寄存器 (UART_USR) 位描述

位	符号	访问	描述
31:5	-	R	保留
4	RFF	R	接收 FIFO 满。用来指示接收 FIFO 是否满。
3	RFNE	R	接收 FIFO 非空。用来指示接收 FIFO 是否空。
2	TFE	R	发送 FIFO 空。用来指示发送 FIFO 是否空。
1	TFNF	R	发送 FIFO 法满。用来指示发送 FIFO 是否满。
0	BUSY	R	忙状态。用来指示串行数据传输正在进行。

27.2.14　发送 FIFO LEVEL 寄存器 (UART_TFL)

地址偏移量：0x080

复位值：0x000

表 27.16: 发送 FIFO LEVEL 寄存器 (UART_TFL) 位描述

位	符号	访问	描述
31:4	-	R	保留
3：0	TFL	R	用来指示发送 FIFO 中的数据个数。

27.2.15　接收 FIFO LEVEL 寄存器 (UART_RFL)

地址偏移量：0x084

复位值：0x000

表 27.17: 接收 FIFO LEVEL 寄存器 (UART_RFL) 位描述

位	符号	访问	描述
31:4	-	R	保留
3：0	RFL	R	用来指示接收 FIFO 中的数据个数。

27.2.16　软件重启寄存器 (UART_SRR)

地址偏移量：0x088

复位值：0x000

表 27.18: 发送 FIFO 读取寄存器 (UART_TFR) 位描述

位	符号	访问	描述
31:3	-	R	保留
2	XFR	RW	这是发送 FIFO 重启控制位 FCR[2] 的影子位。
1	RFR	RW	这是接收 FIFO 重启控制位 FCR[1] 的影子位。
0	UR	RW	这是 UART 模块重启控制位。

27.2.17　影子发送请求寄存器 (UART_SRTS)

地址偏移量：0x08C
复位值：0x000

表 27.19: 发送 FIFO 读取寄存器 (UART_TFR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SRTS	RW	这是发送请求控制位 RTS (MCR[1]) 的影子位。

27.2.18　影子中断控制寄存器 (UART_SBCR)

地址偏移量：0x090
复位值：0x000

表 27.20: 影子中断控制寄存器 (UART_SBCR) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SBCB	RW	这是中断控制位 (LCR[6]) 的影子位。

27.2.19　影子 FIFO 使能寄存器 (UART_SFE)

地址偏移量：0x098
复位值：0x000

表 27.21: 影子 FIFO 使能寄存器 (UART_SFE) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	SFE	RW	这是 FIFO 使能控制位 (FCR[0]) 的影子位。

27.2.20　影子接收触发寄存器 (UART_SRT)

地址偏移量：0x09C
复位值：0x000

表 27.22: 影子接收触发寄存器 (UART_SRT) 位描述

位	符号	访问	描述
31:0	-	R	保留
1:0	SRT	RW	这是接收触发控制位 (FCR[7：6]) 的影子位。

27.2.21　影子发送触发寄存器 (UART_STET)

地址偏移量：0x0A0

复位值：0x000

表 27.23: 影子发送触发寄存器 (UART_STET) 位描述

位	符号	访问	描述
31:0	-	R	保留
1:0	STET	RW	这是发送存储器空触发控制位 (FCR[7：4]) 的影子位。

27.2.22　DMA 软件中止寄存器 (UART_DMASA)

地址偏移量：0x0A8

复位值：0x000

表 27.24: DMA 软件中止寄存器 (UART_DMASA) 位描述

位	符号	访问	描述
31:1	-	R	保留
0	DMASA	RW	在 DMA 发生错误的时候，设置这个 bit 可以中止 DMA 传输。这个位会被自动清除。

27.2.23　分数锁存寄存器 (UART_DLF)

地址偏移量：0x0C0

复位值：0x000

表 27.25: 分数锁存寄存器 (UART_DLF) 位描述

位	符号	访问	描述
31:4	-	R	保留
3：0	DLF	RW	这个寄存器用来存放波特率发生器的分数部分。

27.2.24　接收地址寄存器 (UART_RAR)

地址偏移量：0x0C4

复位值：0x000

表 27.26: 接收地址寄存器 (UART_RAR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7：0	RAR	R	这个寄存器在接收模式中用来存放比较的地址。当第 9 位被置 “1”，接下来的 8 个位会跟 RAR 寄存器中的地址进行比较。当匹配时，接下来的第 9 位为 “0” 的数据会被接收。

27.2.25　发送地址寄存器 (UART_TAR)

地址偏移量：0x0C8

复位值：0x000

表 27.27: 发送地址寄存器 (UART_TAR) 位描述

位	符号	访问	描述
31:8	-	R	保留
7：0	TAR	R	这个寄存器在发送模式中用来存放地址。当 WLS_E 位置“1”，UART 会发送 9 位数据，第 9 位置“1”，接下来的数据会从 TAR 寄存器中取出。

27.2.26　扩展控制寄存器 (UART_EXTLCR)

地址偏移量：0x0CC

复位值：0x000

表 27.28: 扩展控制寄存器 (UART_EXTLCR) 位描述

位	符号	访问	描述
31:4	-	R	保留
3	TRANSMIT_MODE	RW	用来控制是否 9-bit 传输模式。 1：发送寄存器是 9 位宽。 0：发送寄存器是 8 位宽。
2	SEND_ADDR	RW	发送地址控制位。 1：发送 9 位宽数据，第 9 位置“1”，地址从 TAR 中取。 0：发送 9 位宽数据，第 9 位置“0”，数据从 TXFIFO 取。
1	ADDR_MATCH	RW	地址匹配使能。 1：允许地址匹配。 0：正常传输模式。
0	WLS_E	RW	WLS 扩展。用来使能 9 位数据传输

第二十八章　ISO7816 控制器 (ISO)

28.1　简介

ISO/IEC 7816 是一种串行通讯的标准协议，通常用于集成电路卡。本芯片集成了 ISO7816 协议功能。

注 1：若要读写 ISO7816 的寄存器，必须先配置 RCC 寄存器来设置 AHB 总线时钟，并且将 ISO7816 的总线时钟使能。

请参考 RCC 寄存器 AHBPRE 和 AHBENR1。

28.2　特性

主要特性如下：

- 支持 ISO7816-3 T=0 和 T=1 协议
- 当 T=0 时支持重发送 (Resend) 功能
- 支持 Inverse 和 Non-inverse 模式
- 支持波特率从 8 到 2048
- 4 个字节接收数据缓存
- 支持奇偶校验
- 支持 3 个中断源

28.3　功能描述

28.3.1　奇偶校验

复位后奇偶校验功能是关闭的。

当奇偶校验出错时硬件将设置一个标志位，只能由软件清除这个标志位。

28.3.2　中断

支持 3 种中断源：

- 接收模式，时当检测到起始位时产生中断 SBI (Start Bit Interrupt)
- 接收模式，当接收完一个字节的数据后产生中断 RXI (Recieve Interrupt)
- 发送模式，当发送完一个字节的数据后产生中断 TXI (Transmit Interrupt)

3 个中断源都可以被单独屏蔽。

28.3.3 重发送 (Resend)

当 T=1 时不支持重发送。
当 T=0 时支持重发送，但硬件不会自动重发送，需要由软件检测到重发送标志位以后来重新发送数据。

28.4 ISO7816 寄存器

28.4.1 数据发送寄存器 (ISO_TXBUF)

地址偏移量：0x00
复位值：0x0000_0000

位	符号	访问	描述
31:8	-	R	保留
7:0	WDATA	W	要发送的数据

28.4.2 数据接收寄存器 (ISO_RXBUF)

地址偏移量：0x04
复位值：0x0000_0000

位	符号	访问	描述
31:8	-	R	保留
7:0	RDATA	R	从这里读取接收数据，但实际上数据存在缓存 (FIFO) 中

28.4.3 发送结束标志寄存器 (ISO_TXDONE)

地址偏移量：0x08
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	TXDONE	R	一个字节的数据发送完毕 此标志位不能直接清零，要通过读 ISO_CLR_TXDONE 寄存器来将此标志位清零

28.4.4 接收结束标志寄存器 (ISO_RXDONE)

地址偏移量：0x0C
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	RXDONE	R	一个字节的数据接收完毕 此标志位不能直接清零，要通过读 ISO_CLR_RXDONE 寄存器来将此标志位清零

28.4.5 检测到起始位标志寄存器 (ISO_STARTDET)

地址偏移量：0x10
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	START	R	已检测到起始位 此标志位不能直接清零，要通过读 ISO_CLRSTART 寄存器来将此标志位清零

28.4.6 清除发送结束标志寄存器 (ISO_CLRTXDONE)

地址偏移量：0x14
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLR_TXDONE	R	读此寄存器将把 ISO_TXDONE 清零

28.4.7 清除接收结束标志寄存器 (ISO_CLRRXDONE)

地址偏移量：0x18
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLR_RXDONE	R	读此寄存器将把 ISO_RXDONE 清零

28.4.8 清除起始位标志寄存器 (ISO_CLRSTART)

地址偏移量：0x1C
复位值：0x0000_0000

位	符号	访问	描述
31:1	-	R	保留
0	CLR_START	R	读此寄存器将把 ISO_START 清零

28.4.9 传输状态寄存器 (ISO_TRANSR)

地址偏移量：0x20

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	R	保留
1	PERR	R	奇偶校验结果，只在 T=1 的时候有效，当 T=0 时此位永远为 0 0: 无奇偶校验错 1: 检测到奇偶校验错
0	RESEND	R	重发送标志，只在 T=0 时有效 0: 未检测到重发送信号 1: 检测到了重发送信号

28.4.10 数据缓存状态寄存器 (ISO_FIFOSR)

地址偏移量：0x24

复位值：0x0000_0000

位	符号	访问	描述
31:2	-	R	保留
1	EMPTY	R	缓存空标志 0: 缓存里有数据 1: 缓存空
0	FULL	R	缓存满标志 0: 缓存未满 1: 缓存已满

28.4.11 中断屏蔽控制寄存器 (ISO_IER)

地址偏移量：0x28

复位值：0x0000_0000

位	符号	访问	描述
31:3	-	R	保留
2	SBIEN	RW	起始位中断控制 0: 屏蔽起始位中断 1: 打开起始位中断
1	RXIEN	RW	接收中断控制 0: 屏蔽接收中断 1: 打开接收中断
0	TXIEN	RW	发送中断控制 0: 屏蔽发送中断 1: 打开发送中断

28.4.12 模式配置寄存器 (ISO_MODE)

地址偏移量：0x2C

复位值：0x0000_0008

位	符号	访问	描述
31:5	-	R	保留
4:3	GT	RW	保护时间 (Guard Time) 00: 没有保护时间 01: 保护时间为 1 个 ETU 10: 保护时间为 2 个 ETU 11: 保护时间为 4 个 ETU
2	T	RW	T 模式 0: T=0 1: T=1
1	INV	RW	Inverse 模式 0: Non-inverse 模式 1: Inverse 模式
0	PCHK	RW	奇偶校验控制 0: 关闭奇偶校验功能，发送时也不会发送奇偶校验位 1: 发送时会发送奇偶校验位，接收时会检验奇偶校验位

28.4.13 ETU 配置寄存器 (ISO_ETU)

地址偏移量：0x30

复位值：0x0000_0000

位	符号	访问	描述
31:11	-	R	保留
10:0	CYCLE	RW	ETU 时间 N: 1 个 ETU 时间为 N+1 个时钟周期 2047: 1 个 ETU 时间为 2048 个时钟周期

注意：一般设置为 371，即 1 个 ETU 时间为 372 个时钟周期。

28.4.14 无屏蔽中断标志寄存器 (ISO_RISR)

地址偏移量：0x34

复位值：0x0000_0000

描述：此中断标志寄存器无视中断屏蔽位。

位	符号	访问	描述
31:3	-	R	保留
2	SBI	R	起始位中断标志
1	RXI	R	接收中断标志
0	TXI	R	发送中断标志

28.4.15 中断标志寄存器 (ISO_ISR)

地址偏移量：0x38

复位值：0x0000_0000

描述：此中断标志寄存器会受到中断屏蔽位影响。

位	符号	访问	描述
31:3	-	R	保留
2	SBI	R	起始位中断标志
1	RXI	R	接收中断标志
0	TXI	R	发送中断标志

第二十九章　LED 驱动控制器 (LED)

29.1　LED 简介

LED 驱动控制器内含硬件 8 段 LED 驱动串行输出电路，模块时钟默认为内部系统时钟时钟。

29.2　LED 驱动控制器主要特性

LED 驱动控制器的主要特性如下：

- 采用 3 态 IO 口，IO 口可以输出“低电平 0”“高电平 1”“高阻 HI-Z”；
- 模块时钟采用 APB2_CLK；
- 扫描周期可控；
- 可以进行动态切换；

29.3　扫描宽度及周期设置

LED 驱动模块时钟 APB2_CLK 配置完成后，LED 驱动扫描的周期由寄存器 LED_CYC 决定。LED 驱动模块是串行输出，同一时刻最多点亮 1 个 LED，所以 LED 扫描的周期是：

$$56 * T_{apb2\_clk} * (LED\_CYC + 1)$$

29.4　LED 驱动寄存器

29.4.1　LED 控制寄存器 (LED_CON)

地址偏移量：0x00
复位值：0x0000 0000

表 29.1: LED 控制寄存器 (LED_CON) 位描述

位	符号	访问	描述
31:6	-	R	保留
5:4	LED_CHNUM	RW	LED 驱动通道控制 00：开启 led0、led1、led2、led3、led4、led5、led6、led7 01：开启 led0、led1、led2、led3、led4、led5； 10：开启 led0、led1、led2、led3、led4、led5、led6； 11：开启 led0、led1、led2、led3、led4、led5、led6、led7； 注：没有开启的 LED 驱动通道，作为其它功能 IO 口使用
3:1	-	R	保留
0	LED_RUN	RW	LED 使能控制 0：关闭 LED 驱动 1：打开 LED 驱动

29.4.2 LED 周期寄存器 (LED_CYC)

地址偏移量：0x04

复位值：0x0000 0000

表 29.2: LED 周期寄存器 (LED_CYC) 位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	LED_CYC	RW	LED_CYC 寄存器决定驱动两个相邻 LED 的时间间隔，而整个 56 段 LED 的扫描周期为：T= 56 * Δt；

29.4.3 LED 显示寄存器 (LED_ECO)

地址偏移量：0x08

复位值：0x0000 0000

表 29.3: LED 显示寄存器 (LED_ECO) 位描述

位	符号	访问	描述
31:16	-	R	保留
15:0	LED_ECO	RW	LED_ECO 寄存器决定驱动 LED 的时间 (亮度)，LED_ECO 必须小于 LED_CYC；

29.4.4 LED 高段控制寄存器 (LED_SEGH)

地址偏移量：0x08

复位值：0x0000 0000

表 29.4: LED 高段控制寄存器 (LED_SEGH) 位描述

位	符号	访问	描述
31:24	LED_SEGD7	RW	Segment 7 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit31 必须为 0。
23:16	LED_SEGD6	RW	Segment 6 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit22 必须为 0。
15:8	LED_SEGD5	RW	Segment 5 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit13 必须为 0。
7:0	LED_SEGD4	RW	Segment 4 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit4 必须为 0。

图 29.1: LED 段驱动控制输出

Segment*控制输出						
Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1：点亮⑦ 0：高阻 Z	1：点亮⑥ 0：高阻 Z	1：点亮⑤ 0：高阻 Z	1：点亮④ 0：高阻 Z	1：点亮③ 0：高阻 Z	1：点亮② 0：高阻 Z	1：点亮① 0：高阻 Z
1：输出高电平，对应的 LED 点亮状态 0：输出高阻，对应的 LED 灭状态 一一:表示寄存器位对应 port 输出低电平						

29.4.5 LED 低段控制寄存器 (LED_SEGL)

地址偏移量：0x10

复位值：0x0000 0000

表 29.5: LED 低段控制寄存器 (LED_SEGL) 位描述

位	符号	访问	描述
31:24	LED_SEGD3	RW	Segment 3 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit27 必须为 0。
23:16	LED_SEGD2	RW	Segment 2 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit18 必须为 0。
15:8	LED_SEGD1	RW	Segment 1 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit9 必须为 0。
7:0	LED_SEGD0	RW	Segment 0 控制输出位; 1：控制输出高电平; 0：控制输出高阻 注：bit0 必须为 0。

第三十章　器件电子签名

30.1　存储器容量寄存器

基地址：0x4001 6404
只读，它的内容在出厂时编写
详见 SYS 模块3.3.2

30.2　产品唯一身份标识寄存器 (96 位)

产品唯一的身份标识非常适合：

- 用来作为序列号 (例如 USB 字符序列号或者其他的终端应用)
- 用来作为密码，在编写闪存时，将此唯一标识与软件加解密算法结合使用，提高代码在闪存存储器内的安全性。
- 用来激活带安全机制的自举过程

96 位的产品唯一身份标识所提供的参考号码对任意一个 WB32 微控制器，在任何情况下都是唯一的。用户在何种情况下，都不能修改这个身份标识。

这个 96 位的产品唯一身份标识，按照用户不同的用法，可以以字节 (8 位) 为单位读取，也可以以半字 (16 位) 或者全字 (32 位) 读取。

该寄存器只读，它的内容在出厂时编写。

基地址：0x1FFF F204。

30.3　微控制器设备 ID 编码

微控制器 WB32F10xxx 内含一个 MCU ID 编码，定义了 WB MCU 的部件号和硅片版本。

基地址：0x4001 6400
只读，它的内容在出厂时编写
详见3.3.1

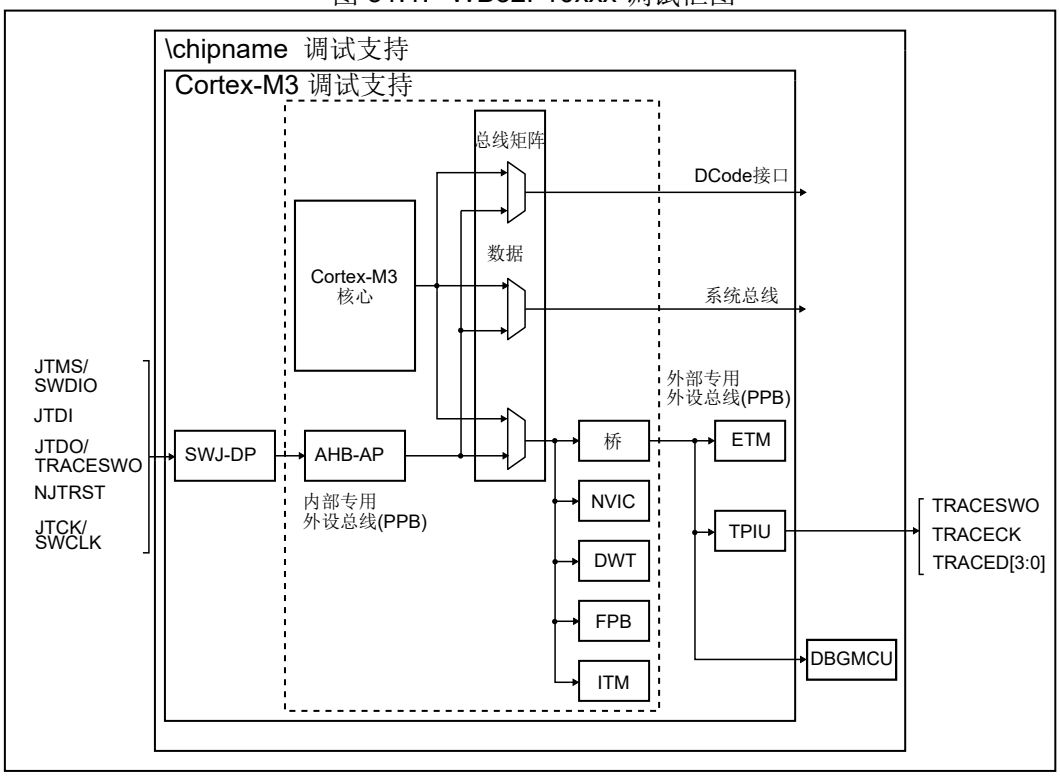
第三十一章　调试支持 (DBG)

31.1　概述

WB32F10xxx 使用 Cortex™-M3 内核，该内核内含硬件调试模块，支持复杂的调试操作。硬件调试模块允许内核在取指 (指令断点) 或访问数据 (数据断点) 时停止。内核停止时，内核的内部状态和系统的外部状态都是可以查询的。完成查询后，内核和外设可以被复原，程序将继续执行。

当 WB32F10xxx 微控制器连接到调试器并开始调试时，调试器将使用内核的硬件调试模块进行调试操作。WB32F10xxx 支持 SWD 串行调试接口。

图 31.1：WB32F10xxx 调试框图



Cortex™-M3 内核内含有的硬件调试模块是 ARM CoreSight 开发工具集的子集。

ARM Cortex™-M3 内核提供集成的片上调试功能。它由以下部分组成：

- SWJ-DP：串行/JTAG 调试端口
- AHP-AP：AHB 访问端口
- ITM：执行跟踪单元

- FPB：闪存指令断点
- DWT：数据触发
- TPUI：跟踪单元接口 (仅较大封装的芯片支持)
- ETM：嵌入式跟踪微单元 (在较大的封装上才有支持此功能的引脚)

专用于 WB32F10xxx 的调试特性:

- 灵活的调试引脚分配
- MCU 调试盒 (支持低电源模式，控制外设时钟等)

注意：更多 *ARM Cortex™-M3* 内核的调试功能信息，请参考 *Cortex™-M3(r1p1* 版*)* 技术参考手册 *(TRM)* 和 *CoreSight* 开发工具集 *(r1p0* 版*) TRM* 。

31.2　参考文献

- Cortex™-M3(r1p1 版) 技术参考手册 (TRM)
- ARM 调试接口 V5
- ARM CoreSight 开发工具集 (r1p0 版) 技术参考手册

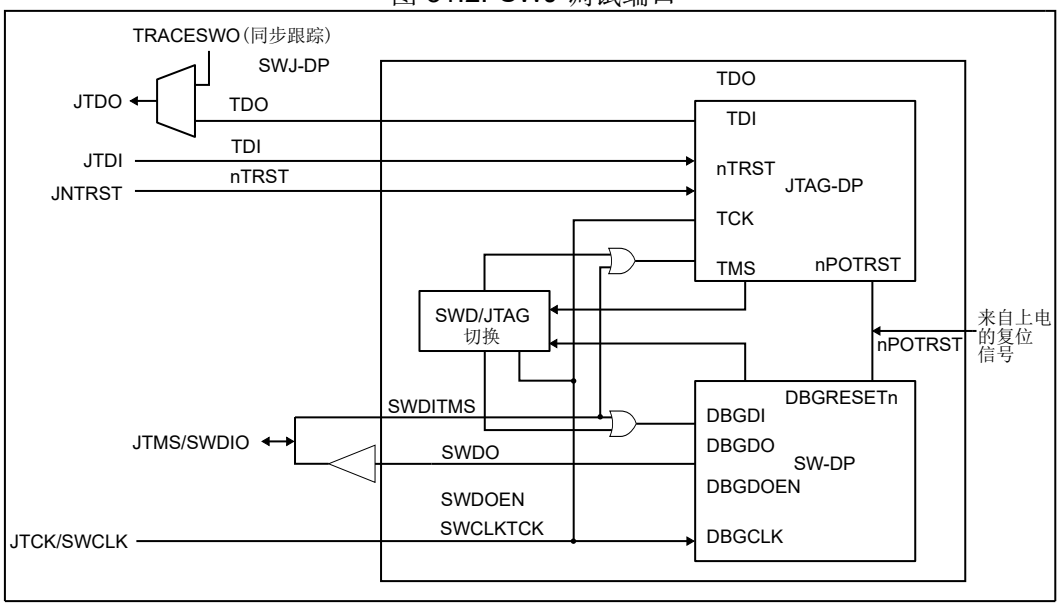
31.3　SWJ 调试接口 (serial wire and JTAG)

WB32F10xxx 内核集成了串行/JTAG 调试接口 (SWJ-DP)。这是标准的 ARM CoreSight 调试接口，包括 JTAG-DP 接口 (5 个引脚) 和 SW-DP 接口 (2 个引脚)。

- JTAG 调试接口 (JTAG-DP) 为 AHP-AP 模块提供 5 针标准 JTAG 接口。
- 串行调试接口 (SW-DP) 为 AHP-AP 模块提供 2 针 (时钟＋数据) 接口。

在 SWJ-DP 接口中，SW-DP 接口的 2 个引脚和 JTAG 接口的 5 个引脚中的一些是复用的。

图 31.2: SWJ 调试端口



上面的图显示异步跟踪输出脚 (TRACESWO) 和 TDO 是复用的，因此异步跟踪功能只能在 SWDP 调试接口上实现，不能在 JTAG-DP 调试接口上实现。

JTAG-DP 和 SW-DP 切换的机制

JTAG 调试接口是默认的调试接口。如果调试器想要切换到 SW-DP，必须在 TMS/TCK 上输出一指定的 JTAG 序列 (分别映射到 SWDIO 和 SWCLK)，该序列禁止 JTAG-DP，并激活 SW-DP。该方法可以只通过 SWCLK 和 SWDIO 两个引脚来激活 SW-DP 接口。

指定的序列是:

1. 输出超过 50 个 TCK 周期的 TMS(SWDIO) = 1 信号
2. 输出 16 个 TMS(SWDIO) 信号 0111100111100111 (MSB)
3. 输出超过 50 个 TCK 周期的 TMS(SWDIO) = 1 信号

WB32F10xxx 不支持 JTAG 调试，只支持 SWD 调试

31.4　引脚分布和调试端口脚

31.4.1　SWJ 调试端口脚

WB32F10xxx 的 2 个普通 I/O 口可用作 SWJ-DP 接口引脚（JTAG 不支持）。这些引脚在所有的封装里都存在。

表 31.1: SWJ 调试端口引脚

SWJ-DP 端口引脚名称	JTAG 调试接口 (不支持)		SW 调试接口		引脚分配
	类型	描述	类型	调试功能	
JTMS/SWDIO	输入	JTAG 模式选择	输入/输出	串行数据输入/输出	PA13
JTCK/SWCLK	输入	JTAG 时钟	输入	串行时钟	PA14
JTDI	输入	JTAG 数据输入 (接到 0)	-	-	-
JTDO/TRACESWO	输出	JTAG 数据输出 (空接)	-	跟踪时为 TRACESWO 信号	PB3
JNTRST	输入	JTAG 模块复位（接到 1）	-	-	-

31.4.2　灵活的 SWJ-DP 脚分配

复位 (SYSRESETn 或 PORESETn) 以后，属于 SWJ-DP 的 2 个引脚都立即被初始化为可被调试器使用的专用引脚 (注意，并没有初始化跟踪输出脚，除非调试器对此脚进行定义)。

然而，WB32F10xxx 微控制器可以用寄存器 GPIOx_MODER 来将这两个专用引脚释放，以用作普通 I/O 口。此寄存器被映射到和 Cortex™-M3 系统总线相连接的 APB 桥上。对此寄存器的设置将由用户代码而不是调试器完成。

31.5　ID 代码和锁定机制

在 WB32F10xxx 微控制器内部有多个 ID 编码。

31.5.1　微控制器设备 ID 编码

微控制器 WB32F10xxx 内含一个 MCU ID 编码。这个 ID 号被固定在 0x0000_0000

DBGMCU_IDCODE

地址：0xE004 2000

WB MCU 的部件号和硅片版本请参考 3.3.1

31.5.2　Cortex-M3 JEDEC-106 ID 代码

ARM 的 Cortex-M3 有一个 JEDEC-106 ID 编码。它位于映射到内部 PPB 总线地址为 0xE00FF000_0xE00FFFFF 的 4KB ROM 表中。使用 SWD 或者用户程序可以访问这一空间。

31.6　SW 调试端口

31.6.1　SW 协议介绍

此同步串行协议使用 2 个引脚：

- SWCLK：从主机到目标的时钟信号
- SWDIO：双向数据信号

协议允许读写 2 个寄存器组 (DPACC 和 APACC 寄存器组)。
数据位按 LSB 传输。
由于 SWDIO 为双向口，该引脚需有上拉 (ARM 建议使用 100KΩ 电阻)。
按协议每次 SWDIO 方向改变时，需插入一个转换时间。在该期间内主机和目标都不驱动此信号线。转换时间的默认值是 1 个比特，但可以通过配置 SWCLK 频率来调节。

31.6.2　SW 协议序列

每个序列由 3 个阶段组成：
1. 主机发送包请求 (8 位)
2. 目标发送确认响应 (3 位)
3. 主机或目标发送数据 (33 位)

表 31.2: 请求包 (8 位)

比特位	名称	描述
0	起始	必须为 1
1	APnDP	0: 访问 DP　　1：访问 AP
2	RnW	0：写请求　　1: 读请求
4:3	A(3:2)	DP 或 AP 寄存器的地址（请参考 0）
5	Parity	前面比特位的校验位
6	Stop	0
8	Park	不能由主机驱动，由于有上拉，目标永远读为 1

有关 DPACC 和 APACC 寄存器描述的详细资料，请参考 Cortex-M3 r1p1 技术参考手册。
包请求后总是跟一个 (默认为 1 位) 转换时间，此时主机和目标都不驱动线路。

表 31.3: ACK 定义 (3 比特位)

比特位	名称	描述
0：2	ACK	001：失败　010: 等待　100: 成功

当 ACK 为失败或等待，或者是一个回复读操作的 ACK，此 ACK 后有一个转换时间。

表 31.4: 数据传输 (33 比特位)

比特位	名称	描述
0：31	WDATA/RDATA	写数据/读数据
32	Parity	32 位数据的奇偶校验位

读操作的数据传输操作后有一个转换时间。

31.6.3　SW-DP 状态机 (Reset, idle states, ID code)

SW-DP 状态机有一个内部 ID 编码用来识别 SW-DP，它遵守 JEP-106 标准。此 ID 编码是 ARM 默认的编码，值为 0x1BA01477(对应于 Cortex-M3 r1p1)。

注意：在调试器读这个 *ID* 编码之前，*SW-DP* 的状态机是不工作的。

- SW-DP 状态机将处于 RESET 状态，在上电复位后，或 DP 从 JTAG 切换到 SWD 后，或有超过 50 个周期的高电平。
- 当状态机处于 RESET 状态时，如果有至少 2 个周期的低电平，状态机将切换到 IDLE 状态。
- 当状态机处于 RESET 状态后，必需首先进入 IDLE 状态，并执行一个读 DP-SW ID 寄存器的操作。否则，调试器在执行其他传输时，只能获得一个失败的 ACK 响应。

更详细的 SW-DP 状态机资料请参考 Cortex-M3 r1p1 技术参考手册和 CoreSight Design Kit r1p0 技术参考手册。

31.6.4　DP 和 AP 读/写访问

- 对 DP 的读操作没有延迟：调试器将直接获得数据 (如果 ACK ＝成功)，或者等待 (如果 ACK= 等待)。
- 对 AP 的读操作具有延迟。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 标志位会在每次 AP 读操作和 RDBUFF 读操作后更新，以通知调试器 AP 的读操作是否成功。
- SW-DP 具有写缓冲区 (DP 和 AP 都有写缓冲)，这使得其他传输在进行时，仍然可以接受写操作。如果写缓冲区满，调试器将获得一个等待的 ACK 响应。读 IDCODE 寄存器，读 CTRL/STAT 寄存器和写 ABORT 寄存器操作在写缓冲区满时仍被接受。
- 由于 SWCLK 和 HCLK 的异步性，需要在写操作后 (在奇偶校验位后) 插入 2 个额外的 SWCLK 周期，以确保内部写操作正确完成。这两个额外的时钟周期需要在线路为低时插入 (IDLE 状态下)。这个操作步骤在写 CTRL/STAT 寄存器以提出一个上电请求时尤其重要，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

31.6.5　SW-DP 寄存器

当 APnDP=0 时，可以访问以下这些寄存器。

表 31.5: SW-DP 寄存器

A(3:2)	读/写	SELECT 的 CTRLSEL 位	寄存器	描述
00	读	-	IDCODE	固定为 0x1BA01477(用于识别 SW-DP)。
00	写	-	ABORT	-
01	读/写	0	DP-CTRL/STAT	-请求一个系统或调试的上电操作; -配置 AP 访问的操作模式; -控制比较，校验操作; -读取一些状态位 (溢出，上电响应)。
01	读/写	1	WIRE CONTROL	配置串行通信物理层协议 (如转换时间长度等)。
10	读	-	READ RESEND	允许从一个错误的调试传输中恢复数据而不用重复初的 AP 传输。
10	写	-	SELECT	选择当前的访问端口和有效的 4 字长寄存器窗口。
11	读/写	-	READ BUFFER	由于 AP 的访问具有传递性 (当前 AP 读操作的结果会在下次 AP 传输时传出), 因此这个寄存器非常必要。这个寄存器会从 AP 捕获上一次读操作的数据结果，因此可以获得数据而不必再启动一个新的 AP 传输。

31.6.6　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

31.7　AHB 访问端口

功能:

- 系统访问是独立于处理器状态的。
- JTAG-DP 和 SW-DP 都可以访问 AHB-AP
- AHB-AP 是总线矩阵的 AHB 主设备。因此，它可以访问所有的数据总线 (Dcode 总线，System 总线，内部和外部 PPB 总线)，只有 ICode 总线除外。
- 支持位寻址的传输
- 旁路 FPB 的 AHB-AP 传输

32 位 AHP-AP 寄存器的地址是 6-位宽 (多 64 个字或 256 个字节)，由以下部分组成:
a) 比特位 [8:4]= DP SELECT 寄存器的位 [7:4] APBANKSEL
b) 比特位 [3:2] = 35 位 SW-DP 包请求中的 A(3:2)。

Cortex-M3 的 AHB-AP 有 9 个 32 位的寄存器

表 31.6: SW-DP 寄存器

地址偏移	寄存器名	描述
0x00	AHB-AP Control and Status Word	配置 AHB 接口的传输特性 (长度，地址自加模式，当前传输状态，特权模式等)。
0x04	AHB-AP Transfer Address	-
0x0C	AHB-AP Data Read/Write	-
0x10	AHB-AP Banked Data 0	直接访问 4 个相连的字而不用重写访问地址。
0x14	AHB-AP Banked Data 1	
0x18	AHB-AP Banked Data 2	
0x1C	AHB-AP Banked Data 3	
0xF8	AHB-AP Debug ROM Address	调试接口的基地址。
0xFC	AHB-AP ID Register	-

更多信息请参考 Cortex-M3 r1p1 技术参考手册。

31.8　内核调试

通过操作内核调试寄存器可以实行对内核的调试。对这些寄存器的访问通过先进高性能总线 (AHB-AP) 进行。处理器可以通过内部私有外设总线 (PPB) 直接访问这些寄存器。它包括 4 个寄存器。

表 31.7: 内核调试寄存器

寄存器名	描述
DHCSR	32 位的调试控制和状态寄存器。 此寄存器提供内核状态信息，允许内核进入调试模式，和提供单步功能。
DCRSR	17 位的内核寄存器调试选择寄存器 此寄存器选择需要进行读写操作的内核寄存器。
DCRDR	32 位的内核寄存器调试数据寄存器 此寄存器存放由 DCRSR 选择的内核寄存器读出的或需要写入的数据。
DEMCR	32 位异常调试和监视控制寄存器 此寄存器提供向量传输和监视调试控制功能。TRCENA 位启动 TRACE 功能。

注意：这些寄存器在系统复位时不复位，仅在上电复位时复位。

更多详细资料请参考 Cortex-M3 r1p1 技术参考手册。

为了在复位后立即使内核进入调试状态，需要：

- 使能调试和异常监视控制寄存器 (Debug and Exception Monitor Control Register) 的位 0 (VC_CORRESET)。
- 使能调试控制和状态寄存器 (Debug Halting Control and Status Register) 的位 0 (C_DEBUGEN)。

31.9　调试器主机在系统复位下的连接能力

WB32F10xxx 微控制器的复位系统由下列复位源组成：

- POR(上电复位)，在每次上电时发起一次复位
- 内部看门狗复位
- 软件复位
- 外部复位

Cortex-M3 将调试部分的复位 (通常是 PORRESETn) 和其他复位 (SYSRESETn) 区分开。因此，当内核处于系统复位状态时，调试器可以连接到内核，配置内核调试寄存器，使能调试允许位，这样操作使内核在系统复位被释放时立即进入调试状态而不执行任何指令。同样的，可以在内核处于复位状态下时配置调试特性。

注意：强烈建议调试器在系统复位时连接内核 (在复位向量处设置断点) 。

31.10　FPB (Flash patch breakpoint)

FPB 单元：

- 实现硬件断点
- 用系统区域的代码和数据取代代码区域的代码和数据。此特性可以用来纠正代码区域内的软件错误。

软件补丁功能和硬件断点功能不能同时使用。
FPB 由以下部分组成：

- 2 个内容比较器，用来比较代码区域取得的内容并重映射到系统区域的相关地址。
- 6 个指令比较器，用来比较代码区域的指令。这些比较器可用来实现软件补丁或者硬件断点功能。

31.11　DWT(数据观察点触发 data watchpoint trigger)

DWT 模块由四个比较器组成，它们分别是：

- 一个硬件数据比较器
- 一个 ETM 触发器
- 一个 PC 值取样器
- 一个数据地址取样器

DWT 还可用来获取某些侧面的信息。通过一些计数器可以获得以下数据：

- 时钟周期

- 分支指令
- 存取单元操作
- 睡眠周期
- CPI(每条指令的执行时间)
- 中断开销

31.12　ITM (指令跟踪微单元)

31.12.1　概述

ITM 是一应用驱动的跟踪源，它支持 printf 类的调试手段来跟踪操作系统 (OS) 和应用事件，并发布判定的系统信息。ITM 以包的形式发布跟踪信息，它由以下部分组成：

- 软件跟踪：软件可以通过直接写 ITM 激发寄存器来发布包信息。
- 硬件跟踪：ITM 会发布由 DWT 产生的信息包。
- 时间戳：时间戳被发布到相应的包上。ITM 包含一个 21 位的计数器以产生时间戳。CortexM3 的时钟或串行线观测器 (Serial Wire Viewer) 的位时钟率给计数器提供时钟。

由 ITM 发送的信息包输出到 TPIU(Trace Port Interface Unit)，TPIU 再添加一些额外的包 (参考 TPIU)，然后输出完整的包序列给调试器。

用户在设置或使用 ITM 之前，必需先使能异常调试和监视控制寄存器 (Debug Exception and Monitor Control Register) 的 TRCEN 位。

31.12.2　时间戳包，同步和溢出包

时间戳包包含了时间戳信息，普通的控制和同步信息。它使用一个 21 位的时间戳计数器 (及可能的预分频器)，此计数器在每个时间戳包发放时复位。计数器的时钟可以是 CPU 时钟也可以是 SWV 时钟。

同步包为 0x80_00_00_00_00_00，按 00 00 00 00 00 80 发送给 TPIU(LSB 在前)。

同步包是时间戳包的控制信号。

它也在每个 DWT 触发时发送，因此 DWT 必须配置为触发 ITM：必须设置 DWT 控制寄存器 (DWT Control Register) 的位 0(CYCCNTENA)。此外，也必须设置 ITM 跟踪控制寄存器 (Trace Control Register) 的位 2(SYNCENA)。

注意：如果 SYNENA 位没有被置起，DWT 产生给 TPIU 的同步触发，将只发送 TPIU 同步包而不发送 ITM 同步包。

溢出包是一个特殊的时间戳包，该包指示数据已经被写但是 FIFO 已满。

表 31.8: 主要的 ITM 寄存器

地址	寄存器	描述
0xE0000FB0	ITM Lock Access	写入 0xC5ACCE55 允许写其他 ITM 寄存器
0xE0000E80	ITM Trace Control	位 31-24 = 总是 0 位 23 = 忙 位 22-16 = 7 位的 ATB ID 用以识别跟踪数据源 位 15-10 = 总是 0 位 9:8 = 时间戳的预分频 位 7-5 = 未定义 位 4 = 使能 SWV 功能即时间戳计数器使用 SWV 时钟 位 3 = 使能 DWT 的激发功能 位 2 = 此位必需设为 1 来使能 DWT 的产生同步触发功能，以使 TPIU 能够发送同步包 位 1 = 时间戳使能 位 0 = ITM 的全局使能位
0xE0000E40	ITM Trace Privilege	位 3：置’1’使能跟踪端口 31:24 位 2：置’1’使能跟踪端口 23:16 位 1：置’1’使能跟踪端口 15:8 位 0：置’1’使能跟踪端口 7:0
0xE0000E00	ITM Trace Enable	每个比特位使能相应的触发端口产生跟踪
0xE0000000 - 0xE000007C	Stimulus Port Registers 0-31	向选中的产生跟踪的触发端口 (32 个) 写 32 位数据

31.13　MCU 调试模块 (MCUDBG)

MCU 调试模块协助调试器提供以下功能：

- 低功耗模式
- 在断点时提供定时器、看门狗时钟控制

31.13.1　低功耗模式的调试支持

使用 WFI 和 WFE 可以进入低功耗模式。

MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的能耗。

内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。

为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

- 在睡眠模式下，调试器必须先置位 DBGMCU_CR 寄存器的 DBG_SLEEP 位。这将为 HCLK 提供与 FCLK(由代码配置的系统时钟) 相同的时钟。

- 在停止模式下，调试器必须先置位 DBG_STOP 位。这将激活内部 RC 振荡器，在停止模式下为 FCLK 和 HCLK 提供时钟。

31.13.2　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。这在输出 PWM 控制电机时常常要用到。
- 在产生断点时，计数器停止计数。这对于看门狗的计数器是必需的。
-

31.13.3　调试 MCU 配置寄存器

此寄存器允许在调试状态下下配置 MCU。包括：

- 支持低功耗模式
- 支持定时器和看门狗的计数器

DBGMCU_CR 寄存器基地址为 0xE0042000, 由 PORESET 异步复位 (不被系统复位所复位)。当内核处于复位状态下时，调试器可写该寄存器。

如果调试器不支持这些特性，用户软件仍可写这些寄存器。

DBGMCU_CR

地址：0xE0042004

只支持 32 位访问

POR 复位：0x0000 0000(不被系统复位所复位)

位	符号	访问	描述
31:14	-	R	保留
13:10	DBG_TIMx_STOP	RW	当内核进入调试状态时，计数器停止工作 x=1..4 0: 选中定时器的计数器仍然正常工作； 1: 选中定时器的计数器停止工作
9	DBG_WWDG_STOP	RW	当内核进入调试状态时调试窗口看门狗停止工作 0: 窗口看门狗计数器仍然正常工作； 1: 窗口看门狗计数器停止工作
8	DBG_IWDG_STOP	RW	当内核进入调试状态时看门狗停止工作 0: 看门狗计数器仍然正常工作； 1: 看门狗计数器停止工作
7:6	-	R	保留
5	TRACE_IOEN	RW	跟踪引脚分配控制 0：不分配跟踪引脚 (默认状态) 1：分配跟踪引脚 (详细参见31.14.2)
4:3	-	R	保留

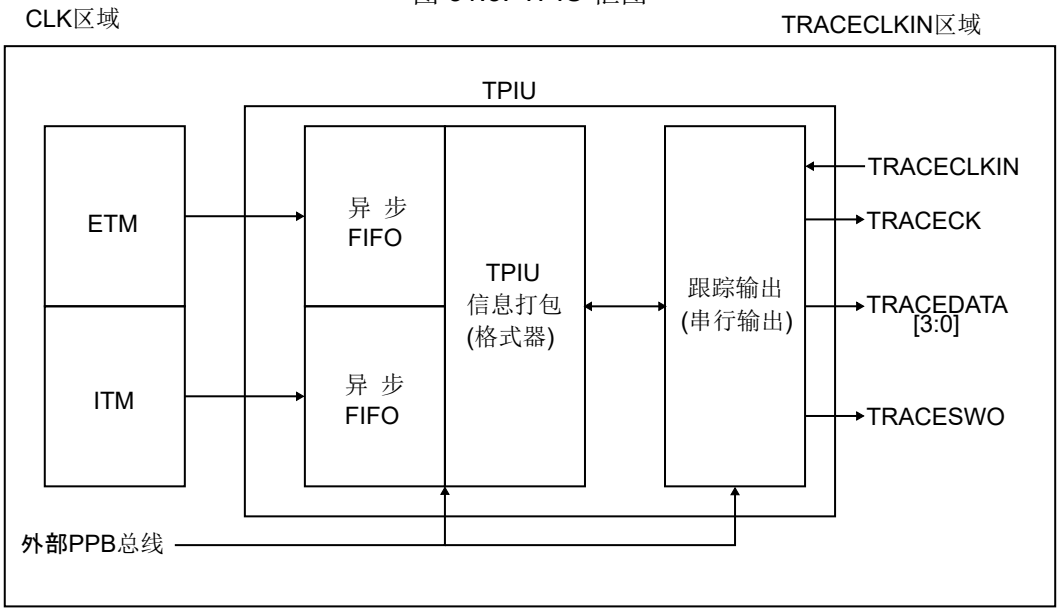
2	DBG_STANDBY	RW	调试待机模式 0:(FCLK 关，HCLK 关) 整个数字电路部分都断电。 从软件的观点看，退出 STANDBY 模式与复位是一样的 (除了一些状态位指示了微控制器刚从 STANDBY 状态退出)。 1：(FCLK 开，HCLK 开) 数字电路部分不下电，FCLK 和 HCLK 时钟由内部 RL 振荡器提供时钟。 另外，微控制器通过产生系统复位来退出 STANDBY 模式和复位是一样的。
1	DBG_STOP	RW	调试停止模式 0：(FCLK 关，HCLK 关) 在停止模式时，时钟控制器禁止一切时钟 (包括 HCLK 和 FCLK)。当从 STOP 模式退出时，时钟的配置和复位之后的配置一样 (微控制器由 8MHz 的内部 RC 振荡器 (HIS) 提供时钟)。因此，软件必需重新配置时钟控制系统启动 PLL，晶振等。 1：(FCLK 开，HCLK 开) 在停止模式时，FCLK 和 HCLK 时钟由内部 RC 振荡器提供。当退出停止模式时，软件必需重新配置时钟系统启动 PLL，晶振等 (与配置此比特位为 0 时的操作一样)。
0	DBG_SLEEP	RW	调试睡眠模式 0：(FCLK 开，HCLK 关) 在睡眠模式时，FCLK 由原先已配置好的系统时钟提供，HCLK 则关闭。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时，软件不需要重新配置时钟系统。 1：(FCLK 开，HCLK 开) 在睡眠模式时，FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。

31.14　TPIU (跟踪端口接口单元)

31.14.1　导言

TPIU 在片上数据跟踪和 ITM 之间担当桥梁的作用。
输出的数据流封装成跟踪源 ID，然后被追踪端口分析器 (Trace Port Analyzer) 采集。
内核嵌入了一个简单的专门为低价调试所设计的 TPIU(由一个特殊版本的 CoreSight TPIU 组成)。

图 31.3: TPIU 框图



31.14.2　跟踪引脚分配

异步模式

异步模式需要 1 个额外的引脚 (TRACESWO) 作为异步跟踪数据输出，并且存在于所有的封装。此模式仅在串行调试接口有效 (不支持 JTAG 调试接口)。该引脚分配的 PAD 是 PB3.

同步模式

同步模式根据跟踪数据长度使用 2-5 个额外引脚，可以提供比异步跟踪更好的数据输出量。

表 31.10: 同步跟踪引脚分配

TPIU 引脚名称	同步跟踪模式		**引脚分配**
	类型	描述	
TRACECK	输出	跟踪时钟	PC4
TRACED[0]	输出	同步跟踪数据输出，0 比特位	PC5
TRACED[1]	输出	同步跟踪数据输出，1 比特位	PC9
TRACED[2]	输出	同步跟踪数据输出，2 比特位	PC10
TRACED[3]	输出	同步跟踪数据输出，3 比特位	PC11

注意：进入调试模式之后，异步跟踪数据输出端口 (TRACESWO) 一直有效。同步模式需要的 5 个引脚在 TRACE_IOEN 为 1 时有效。如果不需要这些引脚，可以释放用作普通 IO。

模式选择

调试器可以通过写 TPIU 的 SPP_R(Selected Pin Protocol) 寄存器的 PROTOCOL [1:0] 位来配置跟踪模式。

- PROTOCOL=00: 跟踪模式 (同步)
- PROTOCOL=01 或 10：串行模式 (曼彻斯特或 NRZ 编码)。默认状态为 01

调试器可以通过写 TPIU 的 CPSPS_R(Current Sync Port Size) 寄存器的位 [3:0] 来配置跟踪端口的大小。

- 0x1：数据宽度为 1 位 (默认)
- 0x2：数据宽度为 2 位
- 0x8：数据宽度为 4 位

31.14.3　TPIU 格式器

协议格式器输出 16 个字节组成的帧：

- 7 个数据字节
- 8 个多用途字节，由以下部分组成：
  - 1 位 (LSB) 用来区分数据字节 (0) 和 ID 字节 (1)。
  - 7 位 (MSB) 可以作为数据或跟踪源 ID 的变化。
- 1 个辅助字节，其中的每个位都对应于 8 个多用途字节中的一个：
  - 如果对应的多用途字节是数据字节，那么这个位是数据的比特 0 位。
  - 如果对应字节是 ID 字节，这个位表明 ID 变化何时生效。

注意：更多信息，请参考 *ARM CoreSight Architecture Specification v1.0(ARM IHI 0029B)*

31.14.4　TPIU 帧异步包

TPUI 会产生两种类型的同步包：

- 帧同步包 (或全字同步包)

  该包为 0x7F_FF_FF_FF(LSB 先发)，这个序列只有在 0x7F 没有被作为 ID 源编码的时候才能使用。

  该包在帧之间周期性地输出。在连续模式里，一旦同步帧被发现，TPA 必须抛弃所有这些帧。

- 半字同步包

  该包为：0x7F_FF(LSB 先发)。

  它在帧之间或帧内周期性的输出。

  这些包只存在于连续模式中，并且使能 TPA 检测 IDLE 模式下的 TRACE 口 (无 TRACE 被捕捉)。当被 TPA 检测到时，必须将其抛弃。

31.14.5　同步帧包的发送

由于内核的 TPIU 内没有同步计数寄存器，因此同步的触发只能由 DWT 产生。参考 DWT Control Register 寄存器 (SYNCTAP[11:10] 位) 和 DWT Current PC Sampler Cycle Count 寄存器。TPUI 帧同步包 (0x7F_FF_FF_FF) 在下列情况时被发送：

- 在每个 TPIU 复位释放后。复位信号同步于 TRACECLKIN 时钟的上升沿释放，这意味着当 DBGMCU_CFG 寄存器的 TRACE_IOEN 位被置位时，同步包就被发送。这种情况下，包 0x7F_FF_FF_FF 后面不跟任何格式的包。

- 在每个 DWT 触发时 (假设已事先设置好 DWT)，有以下两种情况：
  - 如果 ITM 的 SYNENA 位 =0，只发送字 0x7F_FF_FF_FF，后面不跟任何格式的数据包。
  - 如果 ITM 的 SYNENA 位 =1，ITM 同步包将跟在 (0x80_00_00_00_00_00) 后面，由 TPUI 编排格式 (加上跟踪源 ID)。

31.14.6　同步模式

跟踪输出数据的引脚数可以为 4 个，2 个或者 1 个。

TRACECLKIN 频率等于系统时钟频率/8, 具体配置可以参考 RCC 模块寄存器描述。

注意：在此类同步模式中，不需要提供稳定的时钟频率。

TRACE 的 I/O 端口 (包括 TRACECK) 由 TRACLKIN 的上升沿驱动。因此，TRACECK 的输出频率等于 TRACLKIN/2。

31.14.7　异步模式

调试模块提供一个低成本的，只使用一个引脚的跟踪数据输出功能，即使用异步输出引脚 TRACESWO。但显然，这样的输出数据带宽是有限的。

TRACESWO 引脚在所有封装都存在。

异步模式需要 TRACECLKIN 引脚有平稳的频率提供 (系统时钟频率/8)。对标准的 UART(NRZ) 捕捉机制来说，需要 5% 的正确度。曼彻斯特编码可放宽到 10%。TRACECLKIN 在内部连接到 main clk，可以达到这样的精度要求。

31.14.8　TPIU 寄存器

只有当 Debug Exception and Monitor Control(DEMCR) 寄存器的 TRCENA 位被置位时，TPIU APB 寄存器才可以被读写。否则寄存器读出值为 0(这一位的输出使能 TPIU 的 PCLK)。

表 31.11: 重要的 TPIU 寄存器

地址	寄存器	模式
0xE0040004	Current Port Size	跟踪端口的长度： 位 0：端口长度为 1 位 1：端口长度为 2 位 2：端口长度为 3，不支持 位 3：端口长度为 4 四个比特中只能同时置位一个比特。 默认状态下，端口长度为 1(0x00000001)
0xE00400F0	Selected pin protocol	跟踪端口协议的选择： 位 1:0 = 00：同步跟踪模式 01：串行输出 - 曼彻斯特编码 (默认值) 10：串行输出 - NRZ 11：未定义
0xE0040304	Formatter and flush control	位 31-9：总是 0 位 8 = TrigIn：总是 1，指示触发器 位 7-4：总是 0 位 3-2：总是 0 位 1 = EnFCont：同步模式下 (Select Pin Protocol 寄存器的 Bit1：0 为 00)，此比特位强制为 1，连续模式下格式器被自动使能。异步模式下 (Select Pin Protocol 寄存器的 Bit1：0 不为 00)，此比特可以被置位或复位来选择是否使能格式器。 位 0：总是 0 默认值为 0x102 注意：在同步模式下，由于 TRACECTL 信号没有外部引脚，因此格式器会在连续模式下自动使能。这意味着格式器会插入一些控制包来识别跟踪包的源
0xE0040300	Formatter and flush status	没有在 Cortex-M3 中使用，读出值始终为 0x00000008

31.14.9　配置的例子

- 设置 Debug Exception and Monitor Control 寄存器的 TRCENA 位；
- 在 TPIU Current Port Size 寄存器中写入期望值 (默认是 0x1，指示端口长度为 1bit)；
- 向 TPIU Formatter and Flush Control 寄存器中写入 0x102(默认值)；
- 写 TPIU Select Pin Protocol 寄存器，选择同步或异步模式。例如写 0x2 选择 NRZ 编码的异步模式 (类似 URAT)；
- 向 DBGMCU_CR 寄存器写入 0x20(置位 IO_TRACEN)，为异步模式分配 TRACE 的 I/O 口。此时 TPIU 将发出一个同步包 (FF_FF_FF_7F)；

- 配置 ITM 并且写 ITMStimulus 寄存器输出数据。

31.15　DBG 寄存器映象

下表列出了 DBG 的寄存器映像和复位值。

表 31.12: DBG 的寄存器映像表

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xE0042004	DBGMCU_CR	保留																		DBG_TIM4_STOP	DBG_TIM3_STOP	DBG_TIM2_STOP	DBG_TIM1_STOP	DBG_WWDG_STOP	DBG_IWDG_STOP	保留		TRACE_IOEN	保留		DBG_STANDBY	DBG_STOP	DBG_SLEEP
	复位值																			0	0	0	0	0	0			0			0	0	0