HK32F0301MxxxxC 用户手册

版本：1.5

发布日期：2024-01-04

深圳市航顺芯片技术研发有限公司

http://www.hsxp-hk.com

前言

编写目的

本文档介绍了 HK32F0301MxxxxC 系列芯片的功能框图、存储器映射、Flash、中断和事件等功能以及各功能模块的寄存器描述，旨在帮助用户快速开发 HK32F0301MxxxxC 的应用及产品。

读者对象

本文适用于以下读者：

- 开发工程师
- 芯片测试工程师

版本说明

本文档对应的产品系列为 HK32F0301MxxxxC 系列芯片。

修订记录

版本	日期	修订内容
0.99	2022/03/09	Alpha 版本发布
1.0	2022/11/21	官网版本发布
1.1	2023/05/17	1. 章节“11.7 内部参考电压”中增加了表格“内部电压基准测量值”。 2. 更正了描述中的一些细小的问题。
1.2	2023/09/08	1. 章节“3.2.4 写保护”中增加了保护位按位取反才生效的描述。 2. 章节“3.3.1 选项字节擦除”中将 OPTER 位清零步骤挪到最后。 3. 更新了“6.1.1 系统复位”中的复位电路图。 4. 更正了“18.3.4 UART 波特率生成”中的公式。
1.3	2023/11/08	1. 更新了“3.4.1 Flash 访问控制寄存器（FLASH_ACR）”中 LATENCY 位的描述信息。 2. 更新了选项字 WDG_SW 的描述。 3. 更新了 TIM 章节的“PWM 输入模式时序图”“计数方向与编码器信号的关系表”等。
1.4	2023/12/20	1. 更新了“6.1.1 系统复位”中的“图 6-1 复位电路”。 2. 更新了“10.2.3 EXTI 与周边模块关系”中的“图 10-2 EXTI 与周边模块关系框图”：增加了 I2C 到 NVIC 的单箭头。
1.5	2024/01/04	1. 更新了章节“6.1.1 系统复位”：删除了可查询复位标志位的描述。 2. 更新了章节“6.3.9 控制/状态寄存器（RCC_CSR）”：删除了寄存器位 LPWRRSTF、WWDGRSTF 和 SFTRSTF。 3. 更新了章节“8.2.1 通用 I/O（GPIO）”：增加了关于 PB5/PD5 的使用注意信息。

目录

1 简介

本文档为HK32F0301MxxxxC系列芯片的用户手册。本系列芯片是由深圳市航顺芯片技术研发有限公司研发的经济型MCU芯片，包括以下型号：

- HK32F0301MF4x7C
  - HK32F0301MF4U7C（QFN20封装）
  - HK32F0301MF4N7C（QFN20封装）
- HK32F0301MF4P7C（TSSOP20封装）
- HK32F0301MD4P7C（TSSOP16封装）
- HK32F0301MJ4M7C（SOP8封装）

用户可以查看《HK32F0301MxxxxC数据手册》，进一步了解该系列MCU的功能特性，如外设接口、电气特性、管脚封装等。

2 系统及存储器概述

本章介绍了 HK32F0301MxxxxC MCU 的系统架构和内部存储器。

2.1 系统架构

HK32F0301MxxxxC MCU 主要包括以下几个模块：

- 主模块：
  - Cortex®-M0 内核和 AHB-Lite 总线
- 从模块：
  - 内部 SRAM
  - 内部 Flash 存储器
  - AHB-Lite 到 APB 的桥，所有的外设都挂在 APB 总线上
  - 连接于 AHB-Lite 总线的 GPIO 口

HK32F0301MxxxxC MCU 系统架构如下图所示：

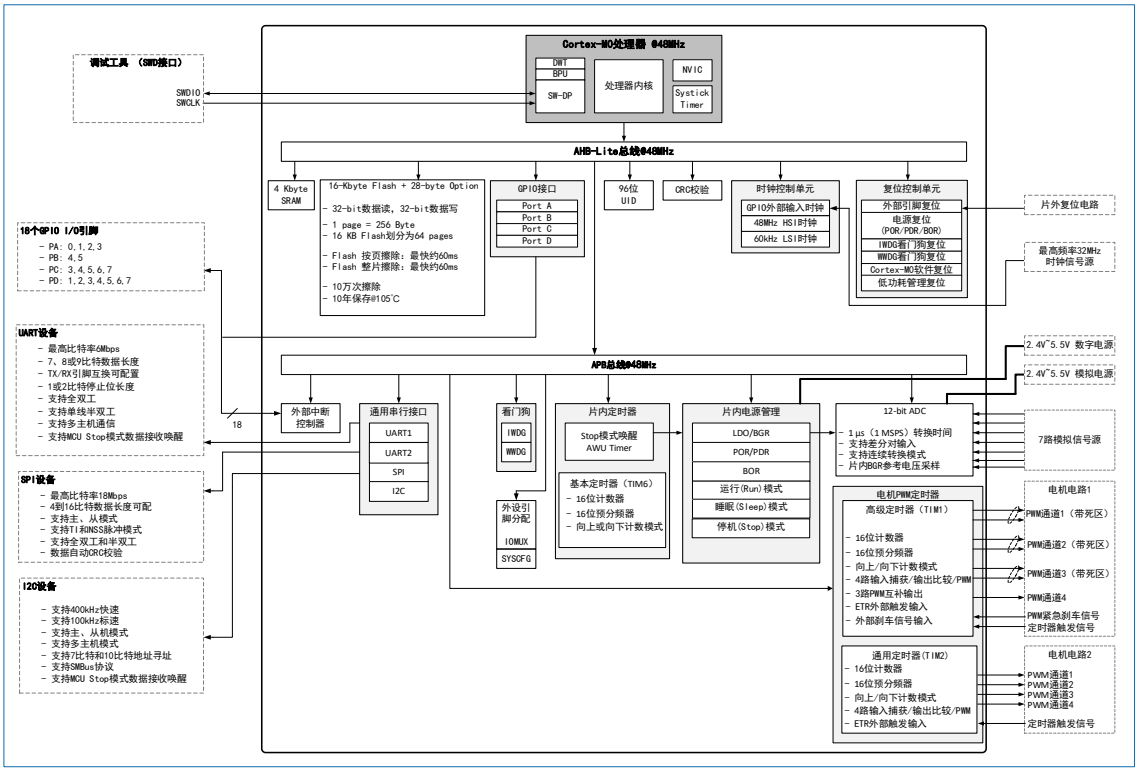

图 2-1 HK32F0301MxxxxC 系统架构图

2.1.1 总线架构

AHB-lite 到 APB 桥

AHB-lite 到 APB 桥在 AHB-lite 与 APB 总线间提供同步连接。

有关连接到桥的不同外设的地址映射请参见图 2-2。

在每次复位之后，所有的外设时钟都关闭（除了 SRAM 及 FLITF 外）。在使用一个外设前，你必须打开相应的 RCC_AHBENR、RCC_APBENRx 寄存器中时钟使能位。

说明：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 16 位或者 8 位的数据扩展以配合 32 位的宽度。

2.2 存储器映射及寄存器编址

程序存储器、数据存储器、寄存器及 I/O 口统一编址，其线性地址空间达到 4G Byte。HK32F0301MxxxxC MCU 支持小端模式的数据存储，即数据的低字节存放于低地址，数据的高字节存放于高地址。

存储器的寻址空间可分成 8 块，每块512M Byte。存储器内的保留区是指暂时未分配给片上存储器和外设的地址空间。

HK32F0301MxxxxC MCU 存储器映射如下图所示：

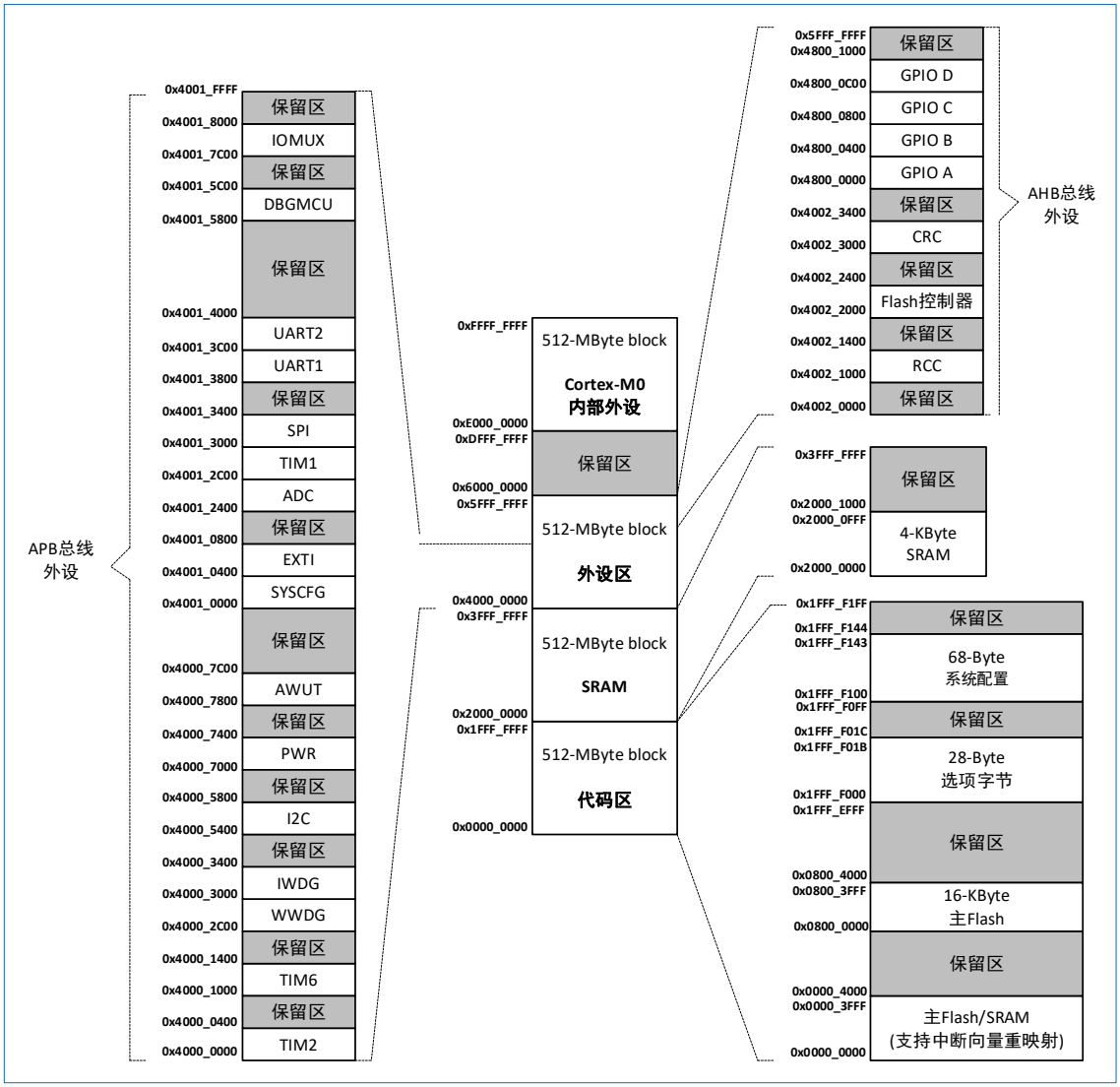

图 2-2 HK32F0301MxxxxC 存储器映射

2.3 SRAM

芯片内置 4 Kbyte 的 SRAM，起始地址是 0x2000 0000。SRAM 可以字节（8 位）、半字（16 位）或字（32 位）方式进行访问。CPU 可使用最快的系统时钟且不插入等待周期访问 SRAM。

SRAM 不支持硬件奇偶校验。

2.4 启动配置

系统复位等待至启动延迟时间结束之后，CPU 获取 Flash 地址 0x0000 0000 中存储的堆栈顶地址，然后从 Flash 地址 0x0000 0004 处开始执行代码。

系统启动后，应用程序可通过修改 SYSCFG_CFGR1 寄存器中的 MEM_MODE 位来重新映射存储器地址。与 Cortex-M3®和 Cortex®-M4 内核不同，Cortex®-M0 内核不支持中断向量表（Interrupt Vector Table，IVT）重映射，但 HK32F0301MxxxxC MCU 支持通过配置 INT_VEC_OFFSET 寄存器实现 IVT 重映射功能。

例如，实现在线升级（In Application Programming，IAP）功能，一种方法是用户将中断向量表重映射到 SRAM：

1. 将中断向量表从 Flash 拷贝到 SRAM 的起始地址 0x2000 0000。
2. 配置 SYSCFG_CFGR1 寄存器的 MEM_MODE[1:0]，以实现 SRAM 映射到 Flash 的地址 0x0000 0000。
3. 当有中断发生时，CPU 从重映射到 SRAM 中的 IVT 取中断服务程序地址，然后跳转到 Flash 中执行中断服务程序。

另外一种方法是通过配置 INT_VEC_OFFSET 寄存器将 IVT 重映射到 Flash 地址。配置完成之后，若有中断发生，CPU 从重映射到 Flash 中的 IVT 取中断服务程序地址，然后跳到 Flash 中执行中断服务程序。

3 Flash

Flash 接口遵循 AHB 协议执行指令和数据存取。

3.1 Flash 特性

- Flash 结构
  - 最高 16K 字节主 Flash 块
    - 应用程序区
    - 用户数据区
  - 信息块，其包括：
    - 选项字节（Option byte）：内含硬件及存储保护用户配置选项。
- Flash 编程/擦除操作
- 访问/写保护

3.2 Flash 功能

3.2.1 Flash 结构

Flash 空间由 32 位宽的存储单元组成，可存储代码和数据。

主 Flash 块可分为 64 页，每页 256 字节。

Flash 选项字节域共有 28 个字节。

表 3-1 Flash 结构

地址	大小（字节）	页号	描述
0x0800 0000-0x0800 00FF	256 Byte	Page0	主 Flash 块
0x0800 0100-0x0800 01FF	256 Byte	Page1	
0x0800 0200-0x0800 02FF	256 Byte	Page2	
0x0800 0300-0x0800 03FF	256 Byte	Page3	
……	……	……	
0x0800 3C00-0x0800 3CFF	256 Byte	Page60	
0x0800 3D00-0x0800 3DFF	256 Byte	Page61	
0x0800 3E00-0x0800 3EFF	256 Byte	Page62	
0x0800 3F00-0x0800 3FFF	256 Byte	Page63	
0x1FFF F000-0x1FFF F01B	28 Byte	-	选项字节
0x1FFF F100-0x1FFF F143	68 Byte	-	系统配置

3.2.2 读操作

嵌入式 Flash 模块可以直接寻址访问。任何对该 Flash 内容的读操作都必须经过专门的判断过程。

指令和数据的访问都是通过 AHB 总线完成，并按照 Flash 访问控制寄存器（FLASH_ACR）所指定的

等待周期进行访问。

3.2.3 读保护

将选项字节中的RDP 字节置位，然后重新复位，读保护功能则被激活。系统存储区不受读保护字节的影响，但该区域不允许编程和擦除操作。Flash 存储器的读保护级别和 RDP 选项字节及其按位取反之后的内容的对应关系，如下表：

表 3-2 读保护级别和 RDP 字节及其按位取反之后的值的对应关系

RDP 字节值	RDP 按位取反之后的值	读保护级别
0xAA	0x55	Level 0
任意值，除 0xAA 和 0xCC 外	任意值（不要求互补），除 0x55 和 0x33 外	Level 1（默认）
0xCC	0x33	Level 2

读保护状态包括三个级别：

- Level 0：无保护

允许对主 Flash 区域和选项字节进行读写和擦除操作。

- Level 1：读保护

这是 RDP 选项字节被擦除之后的默认保护级别。对应的 RDP 值为除 0xAA 和 0xCC 以外的任意值或者其按位取反之后的值。

  - 用户模式：在用户模式下执行的代码允许对主 Flash 和选项字节做全部操作。
  - 调试/从 RAM 或从系统区域启动：包括 boot RAM。在调试模式下或运行在 boot RAM 状态下，不允许访问主 Flash 区和备份寄存器。在调试模式下，任何简单的读访问都会引起总线错误并引发硬件错误中断。主 Flash 区也禁止写和擦除操作，以防范恶意程序修改代码。当 RDP 字节的内容被重新改为 Level 0（0xAA）的级别时，会先执行整片擦除操作，并且备份寄存器也会被复位。

- Level 2：不支持调试

Level 2 包含了 Level 1 的保护功能，且 Cortex M0 的调试接口被禁止了，也不支持从 RAM 启动、系统区启动等功能。

在用户模式下，允许对主 Flash 区进行读写和擦除操作；但选项字节区仅支持读取和写入操作，不支持擦除操作。

在 Level 2 级别时，不能改写 RDP 字节，因此 Level 2 保护级别不能被清除。设置为 Level2 是不可恢复的操作。当试图改写 RDP 字节时，FLASH_SR 寄存器中的保护错误标志 WRPRTERR 会被置位并引发一个中断。

表 3-3 不同工作模式下保护级别和保护状态的对应关系

区域	保护级别	用户代码执行			调试/从 RAM 或从系统区域启动		
		读	写	擦除	读	写	擦除
主 Flash 区域	1	允许	允许	允许	禁止	禁止	禁止 (3)
	2	允许	允许	允许	禁止 (1)	禁止 (1)	禁止 (1)
系统区域	1	允许	禁止	禁止	允许	禁止	禁止
	2	允许	禁止	禁止	禁止 (1)	禁止 (1)	禁止 (1)

区域	保护级别	用户代码执行			调试/从 RAM 或从系统区域启动		
		读	写	擦除	读	写	擦除
选项字节	1	允许	允许 (2)	允许	允许	允许 (3)	允许
	2	允许	允许 (3)	允许	禁止 (1)	禁止 (1)	禁止 (1)

(1). 当使能 Level 2 保护级别，调试口被禁止从 RAM/系统启动。

(2). 当 RDP 被改成不保护时，主 Flash 会被擦除。

(3). 除 RDP 以外的其他选项字节均能被再次编程。

3.2.3.1 改变读保护级别

修改 RDP 的值（除 0xCC 以外的值）就能将读保护级别从 Level 0 级迁移到 Level 1 级别。将 RDP 写入 0xCC，就可以直接进入 Level 2 级别。从 level 1 进入到 Level 0，一定会经过整片擦除阶段。因为在修改 RDP 成功进入到 Level 0 之前，MCU 已经启动了整片擦除。

3.2.4 写保护

写保护是通过配置选项字节中的 WRP 位，然后重新复位系统以使能写保护功能。一共 32 位写保护控制位，每个 WRP bit 置位对应保护 512 byte 主 Flash。

说明：仅当 WRP 和 nWRP 具有按位互反关系时，写保护才能生效。

如果写入或擦除一个受写保护的扇区，会引起 FLASH_SR 中的 WRPRTERR 标志位被置位。

写保护的解除

以下是解除写保护操作的实例：

1. 置位 FLASH_CR 中的 OPTER 位，擦除整个选项字节区域。
2. 向 RDP 写入 0xAA 从而解除所有保护，这会引起整片擦除。

选项字节的写保护

选项字节默认被写保护且随时可读。必须先向 FLASH_OPTKEYR 寄存器顺序写入关键字，才能对选项字节进行写/擦除操作。填入正确的关键字会引起 FLASH_CR 中的 OPTWRE 置位，表明解锁成功；通过对 OPTWRE 位清零，能够禁止对选项字节的写操作。

3.2.5 主 Flash 写和擦除操作

电路编程（In Circuit Programing，ICP）使用 SWD 或 Bootloader 的方法在线改变 Flash 的内容，将用户代码烧录到 MCU 中。ICP 提供了一种简单高效的方法，免除了烧写芯片时的芯片装夹等问题。

与 ICP 方法不同的是，IAP 能够使用 MCU 支持的任何通信接口下载程序或者数据。IAP 允许用户在运行程序的过程中重写应用程序，前提是 IAP 的烧录引导程序已经写入 MCU。

写和擦除操作在整个产品工作电压范围内都可以完成。该操作涉及以下寄存器的配置：

- Flash 关键字寄存器（FLASH_KEYR）
- Flash 状态寄存器（FLASH_SR）
- Flash 控制寄存器（FLASH_CR）
- Flash 地址寄存器（FLASH_AR）
- 写保护寄存器（FLASH_WRPR）

在写/擦除 Flash 时，不能对 Flash 进行取指或数据访问。只要 CPU 不访问 Flash 空间，进行中的 Flash 写操作不会影响 CPU 的运行。即在对 Flash 进行写/擦除操作时，任何对 Flash 的访问都会令总线停顿，直到写/擦操作完成后才会继续执行 Flash 的访问。

在对 Flash 进行写/擦除操作时，内部 RC 振荡器（HSI）必须处于开启状态。

3.2.5.1 主 Flash 空间的解锁

复位后，Flash 存储器默认处于受保护状态，以避免意外擦除。FLASH_CR 寄存器的值通常不允许改写。只有对 FLASH_KEYR 寄存器进行解锁操作后，才具有对 FLASH_CR 寄存器的访问权限。FLASH_KEYR 寄存器的解锁操作包括以下步骤：

1. 向 FLASH_KEYR 寄存器写入关键字 KEY1=0x4567 0123；
2. 向 FLASH_KEYR 寄存器写入关键字 KEY2=0xCDEF 89AB。

错误的操作顺序将会锁死 FLASH_CR 直至下次复位。当写入关键字错误时，会由总线错误触发一次硬件错误中断。

- 如果 KEY1 出错，就会立即触发中断。
- 如果 KEY1 正确但 KEY2 错误时，就会在 KEY2 错误的时刻触发中断。

3.2.5.2 主 Flash 擦除

主 Flash 存储器可以按页为单位擦除，也可以整片擦除。

注意：HK32F0301MxxxxC 系列芯片擦除后为全 F。

擦除页

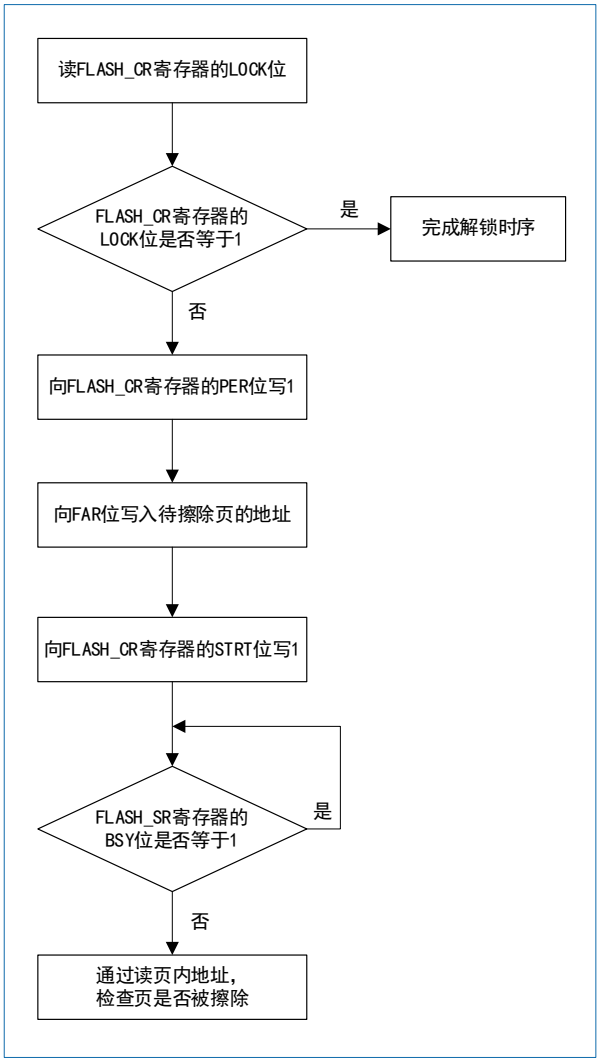

图 3-1 Flash 擦除页的流程

擦除页的操作步骤：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确认上次操作已经结束。
2. 将 FLASH_CR 寄存器中的 PER 位置为 1，以选择按页擦除。
3. 写 FLASH_AR 寄存器的 FAR 位，写入待擦除页的地址。
4. 将 FLASH_CR 寄存器中的 STRT 位置为 1，以启动擦除操作。
5. 等待 FLASH_SR 中的 BSY 变为 0，表明擦除操作完成。
6. 检查 FLASH_SR 寄存器的 EOP 标志（若 Flash 擦除成功会置位 EOP），然后软件清除该标志位。
7. 将 FLASH_CR 寄存器中的 PER 位置 0，以恢复默认值。

整片擦除

整片擦除命令可以一次擦除整个 Flash 扇区。整片擦除的具体步骤如下：

1. 检查 FLASH_SR 寄存器的 BSY 位，以确认上次操作已经结束。
2. 将 FLASH_CR 寄存器中的 MER 位置 1，以选择整片擦除。
3. 将 FLASH_CR 寄存器中的 STRT 位置为 1，以启动擦除操作。

4. 等待 FLASH_SR 中的 BSY 位置 0，表明整片擦除操作结束。

5. 检查 FLASH_SR 寄存器的 EOP 标志位（如果 Flash 擦除成功会置位 EOP），然后软件清除该标志位。

6. 将 FLASH_CR 寄存器中的 MER 位置 0，以禁能整片擦除功能。

说明：整片擦除命令对信息块不起作用。

3.2.5.3 主 Flash 编程

主 Flash 一次可以编程 32 位（字）。解锁 Flash 后，当 FLASH_CR 寄存器中的 PG 位为 1 时，则向指定的地址写入 1 个字的数据，即完成一次 Flash 编程操作。

注意：

若 Flash 处于解锁状态，FLASH_CR 寄存器中的 PG 位为 1 时，写入的数据长度不是 1 个字，则会引起硬件错误中断。

主 Flash 编程的流程如下图所示：

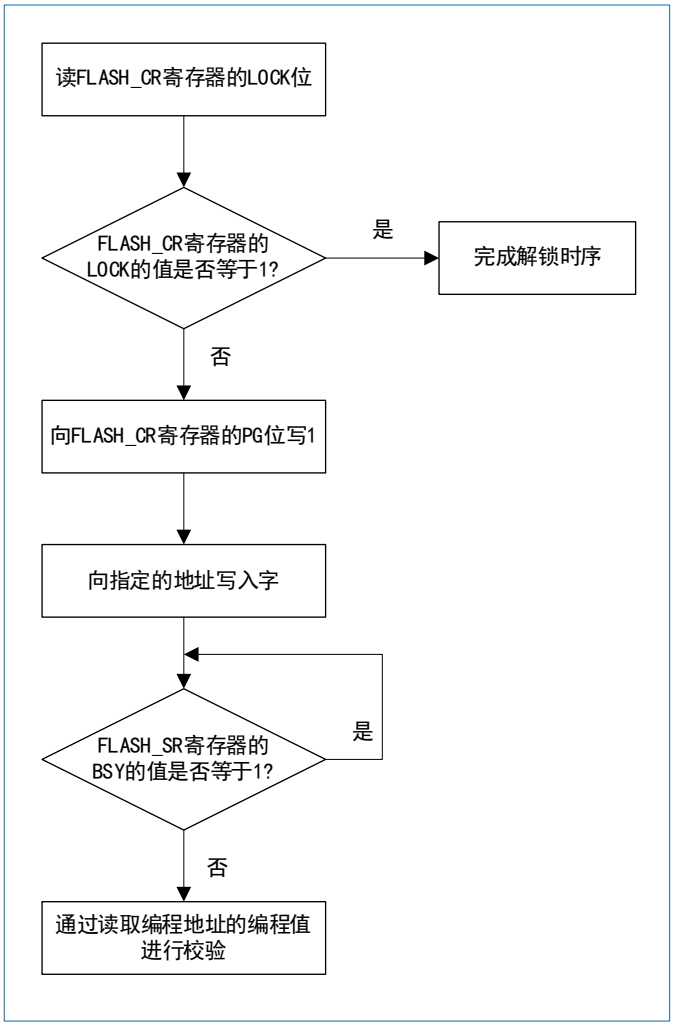

图 3-2 对 Flash 的编程

当待编程地址所对应的 FLASH_WRPR 寄存器的写保护位生效时，不允许编程操作，否则会产生编程错误告警。编程操作结束后，FLASH_SR 寄存器的 EOP 位会提示该操作是否成功。

主 Flash 存储器的标准编程流程如下：

1. 检查 FLASH_SR 中的 BSY 位，以确认上次操作已经结束。

2. 将 FLASH_CR 寄存器中的 PG 位置为 1，以写入 Flash。
3. 根据配置，以字为单位向目标地址写入数据。
4. 在完成 Flash 的数据写入后，将 FLASH_CR 寄存器中的 PG 位置为 0。
5. 等待 FLASH_SR 寄存器中的 BSY 变为 0。
6. 检查 FLASH_SR 寄存器的 EOP 标志位（如果 Flash 编程成功会置位 EOP），然后软件清除该标志位。

3.2.6 Flash 中断

表 3-4 Flash 中断事件和事件标志

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

3.3 Flash 选项字节

Flash 选项字节域共有 28 个字节，根据用户的应用需求进行配置。

一个 32 位选项字可划分成如下表所示的选项字节：

表 3-5 选项字划分成选项字节

[31:24]	[23:16]	[15:8]	[7:0]
选项字节 1 按位取反	选项字节 1	选项字节 0 按位取反	选项字节 0

选项字若有内容更新，需复位系统才能生效。前面 4 个选项字是以按位取反之后和选项字节组合的形式存储，如下表所示。

表 3-6 选项字节结构

地址	[31:24]	[23:16]	[15:8]	[7:0]
0X1FFF F000	nUSER	USER	nRDP	RDP
0X1FFF F004	nDATA1	DATA1	nDATA0	DATA0
0X1FFF F008	nWRP1	WRP1	nWRP0	WRP0
0X1FFF F00C	nWRP3	WRP3	nWRP2	WRP2
0X1FFF F010	IWDG_INI_KEY[15:0]		保留	IWDG_RL_IV[11:0]
0X1FFF F014	DBG_CLK_CTL[15:0]		LSI_LP_CTL[15:0]	
0X1FFF F018	NRST_IOEN		nBOR_LEVEL	BOR_LEVEL

表 3-7 选项字节描述

地址	位域	选项字节描述
0x1FFF F000	31:24	nUSER：USER 按位取反

地址	位域	选项字节描述

	23:16	USER：用户选项字节
		USER 内容存储于 FLASH_OBR[15:8]，用于配置如下特性：
		● 选择硬件或软件看门狗事件。
		● 当进入停止模式时，产生复位事件。
		  ○ 位[23:18]：保留
		  ○ 位 17：nRST_STOP
		  - 0：当进入停机模式产生复位
		  - 1：不产生复位
		  ○ 位 16：WDG_SW
		  - 0：IWDG 硬件看门狗
		  - 1：IWDG 软件看门狗

	15:8	nRDP：RDP 按位取反
	7:0	RDP：Flash 读保护选项字节 该字节的值定义了 Flash 读保护级别。 ● 0xAA：级别 0 ● 0xXX（除 0xAA 和 0xCC 取值外）：级别 1 ● 0xCC：级别 2
0x1FFF F004	31:0	DATAx：用户数据 ● 位[31:24]：nDATA1 ● 位[23:16]：DATA1（存于 FLASH_OBR[31:24]） ● 位[15:8]：nDATA0 ● 位[7:0]：DATA0（存于 FLASH_OBR[23:16]）

0x1FFF F008	31:0	WRPx：Flash 写保护选项字节
		● 位[31:24]：nWRP1
		● 位[23:16]：WRP1（存于 FLASH_WRPR[15:8]）
		● 位[15:8]：nWRP0
		● 位[7:0]：WRP0（存于 FLASH_WRPR[7:0]）
		  ○ 0：写保护使能
		  ○ 1：写保护禁能
		Flash 的写保护范围是按照每一位（Bit）对应 2（Page）进行控制。WRP0作用于0 ~ 15 页；WRP1作用于16~31 页。

0x1FFF F00C	31:0	WRPx：Flash 写保护选项字节
		● 位[31:24]：nWRP3
		● 位[23:16]：WRP3（存于 FLASH_WRPR[31:24]）
		● 位[15:8]：nWRP2
		● 位[7:0]：WRP2（存于 FLASH_WRPR[23:16]）
		  ○ 0：写保护使能
		  ○ 1：写保护禁能
		Flash 的写保护范围是按照每一位（Bit）对应 2（Page）进行控制。WRP2 作用于32～47 页。WRP3 作用于48～63 页。

0x1FFF F010	31:16	IWDG_INI_KEY：决定 IWDG_RL_IV 是否生效。 当 IWDG_INI_KEY[31:16]为0x5B1E 时，IWDG_RL_IV 配置有效，否则无效。
	15:12	保留值

地址	位域	选项字节描述
	11:0	IWDG_RL_IV[11:0]：IWDG 重载值。 当 IWDG_INI_KEY[15:0]为0x5B1E 时，IWDG_RL_IV 配置有效，否则无效。 IWDG 重载值计算公式参见“16 独立看门狗（IWDG）”。 *需要注意的是：* *1、 使用该功能，必须程序中要对 IWDG 喂狗；* *2、 如果没有对 IWDG 喂狗，且该值设置太小，IWDG 复位间隔时间短，可能会导致芯片无法正常烧录。*
0x1FFF F014	31:16	DBG_CLK_CTL：关闭或打开 CPU 内部 Debug 时钟。 当存储的值为0x12DE 时，关闭 CPU 内部 Debug 时钟，SWD 将不能访问 MCU；否则保持 Debug 时钟为打开状态。
	15:0	LSI_LP_CTL：决定 MCU 在使能 IWDG 后再进入停机模式时，是否需要被 IWDG 周期唤醒。 ● 该位域配置为 0x369C 时： MCU 进入停机（Stop）模式后，可根据 LSION 的设置关闭 LSI。若 LSION 为开启状态，则 MCU 被 AWUT 或 IWDG 周期唤醒，反之，则不会被唤醒。MCU 被唤醒后，LSI 恢复为进入停机模式之前的状态。 ● 若未配置该位域： 在使能 IWDG 后再进入停机模式，系统会被 AWUT 或 IWDG 周期唤醒。
0x1FFF F018	31:16	NRST_IOEN：当其为 0x3456 时，NRST Pin 可通过 IOMUX 被配置为替换为 GPIO PA0，当期为其他值时，NRST 引脚不可被配置为 GPIO PA0。
	15:8	nBOR_LEVEL：BOR_LEVEL 的取反，硬件不会被自动取反。
	7:0	BOR_LEVEL：存储 BOR 阈值，nBOR_LEVEL 和 BOR_LEVEL 为取反关系时，BOR_LEVEL 的低 4 位有效，如果该位域与其反码在选项字节加载期间发生不匹配，则 BOR 不会加载该位域值。 ● 0xxx：BOR 关闭。 ● 1000：保留 ● 1001：BOR 级别 1（VBOR1）：约 2.6 V 的复位阈值级别。 ● 1010：BOR 级别 2（VBOR2）：约 3.0V 的复位阈值级别。 ● 1011：BOR 级别 3（VBOR3）：约 3.4 V 的复位阈值级别。 ● 1100：BOR 级别 4（VBOR4）：约 3.8 V 的复位阈值级别。 ● 1101：BOR 级别 5（VBOR5）：约 4.2 V 的复位阈值级别。 ● 1110：BOR 级别 6（VBOR6）：约 4.6V 的复位阈值级别。 ● 1111：BOR 级别 7（VBOR7）：约 5.0V 的复位阈值级别。

每次系统复位后，选项字节装载器（OBL）读取和存储信息块数据到相应的选项字节寄存器（FLASH_OBR）和闪存保护寄存器（FLASH_WRPR）中。

每个选项字节都有其值按位取反之后的值存放在信息块中，其目的用于校验选项字节的正确性。当选项字节装载后，CPU 会检查选项字节的正确性。若选项字节与其按位取反之后的值比较不一致时，会产生选项字节校验错（OPTERR）信息。当产生 OPTERR 后，CPU 会强制将相应的选项字节值变为 0xFF。当选项字节与其按位取反之后的值都为 0xFF（擦除状态）时，CPU 不会比较其与按位取反之后的值的差异。

注意：选项字节的擦除和编程操作在整个产品工作电压范围内都可以完成。

3.3.1 选项字节擦除

选项字节擦除操作涉及以下寄存器的配置：

- Flash 关键字寄存器（FLASH_OPTKEYR）
- Flash 状态寄存器（FLASH_SR）
- Flash 控制寄存器（FLASH_CR）
- Flash 地址寄存器（FLASH_AR）

擦除选项字节前的准备：

需要先对 FLASH_KEYR 和 FLASH_OPTKEYR 写入关键字 KEY 以解锁 Flash，然后进行选项字节擦除操作。

选项字节是按页擦除的，步骤如下：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确保上次操作结束。
2. 将 FLASH_CR 寄存器中的 OPTWRE 位置 1，以允许改写选项字节。
3. 置 FLASH_CR 寄存器中的 OPTER 位为 1，选择擦除选项字。
4. 将要擦除地址写入 FLASH_AR 寄存器中。
5. 将 FLASH_CR 寄存器中的 STRT 位置为 1，以启动擦除操作。
6. 等待 FLASH_SR 中的 BSY 位变成 0，表明擦除操作结束。
7. 将 FLASH_CR 寄存器中的 OPTER 位置为 0，以恢复默认值。

3.3.2 选项字节编程

选项字节区按照字为单位进行编程。该区大小总共 7 个字，包括：

- 读保护
- 硬件配置
- 用户数据
- 写保护
- IWDG 配置
- 调试时钟控制

选项字节编程前的准备：

需要先对 FLASH_KEYR 和 FLASH_OPTKEYR 写入关键字 KEY 以解锁 Flash，然后进行选项字节编程操作。LSB 值会自动转化为 MSB，以适应选项字节的位定义。

选项字节编程的步骤如下：

1. 检查 FLASH_SR 寄存器中的 BSY 位，以确保上次操作结束。
2. 将 FLASH_CR 寄存器中的 OPTPG 位置为 1，以选择字方式写入。
3. 写数据（字）到目标地址。
4. 等待 FLASH_SR 中的 BSY 位为 0，表示编程操作结束。
5. 将 FLASH_CR 寄存器中的 OPTPG 位置为 0，以恢复默认值。

当读保护选项字节由保护状态变成非保护状态时，会执行一次整片擦除，然后才允许改写读保护位数据。若用户仅想改写选项字节区以外的数据，则不会引起整片擦除，该机制用于保护 Flash 的内容。

3.4 Flash 寄存器

基地址：0x4002 2000

空间大小：0x400

3.4.1 Flash 访问控制寄存器（FLASH_ACR）

偏移地址：0x00

复位值：0x0000 0000

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

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

位 31:3	Res：保留 必须保持复位值。

位 2:0	LATENCY[2:0]：等待周期（Latency）
	本位预设 HCLK 周期和 Flash 访问时间的比率关系。
	• 当 4.5V＜V_{DD}≤5.5V：
	  ○ 000：0 等待周期（适用于 HCLK≤24MHz）
	  ○ 001：1 个等待周期（适用于 24 MHz＜HCLK ≤48 MHz）
	• 当 2.7V＜V_{DD}≤4.5V：
	  ○ 000：0 等待周期（适用于 HCLK≤19MHz）
	  ○ 001：1 个等待周期（适用于 19 MHz＜HCLK ≤38 MHz）
	  ○ 010：2 个等待周期（适用于 38 MHz＜HCLK ≤58 MHz）
	• 当 2.4V＜V_{DD}≤2.7V：
	  ○ 000：0 等待周期（适用于 HCLK≤14MHz）
	  ○ 001：1 个等待周期（适用于 14 MHz＜HCLK ≤28 MHz）
	  ○ 010：2 个等待周期（适用于 28 MHz＜HCLK ≤42 MHz）
	  ○ 011：3 个等待周期（适用于 42 MHz＜HCLK ≤56 MHz）
	通过配置 LATENCY，能使 CPU 在极低的频率下运行应用程序；配置的等待周期越大，芯片的功耗越低。

3.4.2 Flash 关键字寄存器（FLASH_KEYR）

偏移地址：0x04

复位值：0xXXXX XXXX

说明：x 表示不定值。

该寄存器仅支持写。若读该寄存器，返回值为 0。

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

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

FKEYR[15:0]
w

位 31:0	FKEYR[31:0]：Flash 关键字（Flash key） 该位域用于存储解锁 Flash 的关键字。 • KEY1：0x4567 0123 • KEY2：0xCDEF 89AB

3.4.3 Flash 选项关键字寄存器（FLASH_OPTKEYR）

偏移地址：0x08

复位值：0xXXXX XXXX

说明：X 表示不定值。

该寄存器仅支持写。若读该寄存器，返回值为 0。

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

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

位 31:0	OPTKEYR[31:0]：选项字节关键字（Option byte key） 该位域用于存储可解锁 OPTWRE 的关键字。 • KEY1：0x4567 0123 • KEY2：0xCDEF 89AB

3.4.4 Flash 状态寄存器（FLASH_SR）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res										EOP	WRPRTERR	Res	PGERR	Res	BSY
										rc_w1	rc_w1		rc_w1		r

位 31:6	Res：保留 必须保持复位值。
位 5	EOP：操作结束（End of operation） 当 Flash 操作（写/擦除）完成时，由硬件置位。该位由软件写 1 清零。 *注意：写或擦除操作成功后，硬件才会置位 EOP。*
位 4	WRPRTERR：写保护错误标志（Write protection error） 当出现对写保护区域的写操作时，该位被硬件置位。该位由软件写 1 清零。

位 3	Res：保留 必须保持复位值。
位 2	PGERR：编程错误（Programming error） 当被编程区域的状态不为'0xFFFF'的情况下，执行写入操作时被硬件置位。 软件写 1 复位。 *注意：在写操作之前 FLASH_CR 寄存器中的 STRT 位应该先被清零。*
位 1	Res：保留 必须保持复位值。
位 0	BSY：忙标志（Busy） 该位标明 Flash 操作正在进行。当开始 Flash 操作时，该位被硬件置为“1”。当操作结束时或发生错误时，该位由硬件清零。

3.4.5 Flash 控制寄存器（FLASH_CR）

偏移地址：0x10

复位值：0x0000 0080

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			EOPIE	Res	ERRIE	OPTWRE	Res	LOCK	STRT	OPTER	OPTPG	Res	MER	PER	PG
			rw		rw	rw		rw	rw	rw	rw		rw	rw	rw

位 31:13	Res：保留 必须保持复位值。
位 12	EOPIE：操作结束中断使能（End of operation interrupt enable） 该位可使 FLASH_SR 中的 EOP 位变为 1 时，产生中断请求。 • 0：中断禁用 • 1：中断使能
位 11	Res：保留 必须保持复位值。
位 10	ERRIE：操作错误中断使能（Error interrupt enable） 该位使 FLASH_SR 中的 PGERR/WRPRTERR 位变为 1 时，产生中断请求。 • 0：中断禁用 • 1：中断使能
位 9	OPTWRE：选项字节写使能（Option byte write enable） 该位为 1 时，允许改写选项字节。FLASH_OPTKEYR 寄存器写入正确的关键字序列，该位被置 1。

	该位可由软件清零。
位 8	Res：保留 必须保持复位值。
位 7	LOCK：锁定 Flash 标志（Lock） 当该位为 1时，表明 Flash 为锁定状态。通过解锁时序将该位清零。当解锁不成功时，该位就一直为1了，除非下次复位重新操作。 *注意：该位只能写 1。*
位 6	STRT：启动（Start） 该位会触发一个擦除操作，仅由软件置 1，仅在 BSY 为 0 时被清零。
位 5	OPTER：选项字节擦除（Option byte erase） 擦除整个选项字节。
位 4	OPTPG：选项字节写入（Option byte programming） 只能按字的方式写入。
位 3	Res：保留 必须保持复位值。
位 2	MER：主 Flash 整片擦除（Mass erase） 主 Flash 整片擦除时选择。
位 1	PER：主 Flash 页擦除（Page erase） 主 Flash 页擦除时选择。
位 0	PG：主 Flash 按字写入（Programming） 主 Flash 写入时可配置该位。

3.4.6 Flash 地址寄存器（FLASH_AR）

偏移地址：0x14

复位值：0x0000 0000

该寄存器通过硬件根据当前和上一次操作进行更新。对于页擦除操作，该寄存器该由软件来更新以便瞄准要擦除的页。

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

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

位 31:0	FAR[31:0]：Flash 地址（Flash Address）

	当 PG 位使能时，该位域为写入 Flash 的地址；或当 PER 位使能时，选择待擦除的页。 *注意：当 FLASH_SR 中的 BSY 为 1 时，禁止写该寄存器。*

3.4.7 Flash 选项字节寄存器（FLASH_OBR）

偏移地址：0x1C

复位值：0xXXXX XX0X

选项字节的写入值决定了该寄存器的复位值；复位时，选项字节加载环节中比较选项字节及其按位取反之后的值的结果决定了 OPTERR 位的复位值。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA1[7:0]								DATA0[7:0]							
r								r							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						nRST_STOP	WDG_SW	Res					RDPRT[1:0]		OPTERR
						r	r						r		r

位 31:24	DATA1[7:0]：用户数据（User data） 该位属于用户选项字节，数据由用户提供，默认是 0。
位 23:16	DATA0[7:0]：用户数据（User data） 该位属于用户选项字节，数据由用户提供，默认是 0。
位 15:10	Res：保留 必须保持复位值。
位 9	nRST_STOP：进入停机模式是否产生复位（Enter the stop mode for a reset） 该位属于用户选项字节。 • 0：当进入停机模式产生复位。 • 1：不产生复位。
位 8	WDG_SW：选择软件或硬件看门狗（Select a software or hardware watchdog） 该位属于用户选项字节。 • 0：IWDG 硬件看门狗 • 1：IWDG 软件看门狗
位 7:3	Res：保留 必须保持复位值。
位 2:1	RDPRT[1:0]：读保护状态（Read protection level status） • 00：当前处于 Level 0 读保护状态。 • 01：当前处于 Level 1 读保护状态。 • 11：当前处于 Level 2 读保护状态。
位 0	OPTERR：选项字节错误（Option byte error）

	当该位置 1 时，表明加载选项字节互补关系不成立。

3.4.8 Flash 写保护寄存器（FLASH_WRPR）

偏移地址：0x20

复位值：0xXXXX XXXX

选项字节的写入值决定了该寄存器的复位值。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRP3[7:0]								WRP2[7:0]							
rw								rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRP1[7:0]								WRP0[7:0]							
rw								rw							

位 31:24	WRP3[7:0]：写保护（Write protect） 该寄存器包含保持由 OBL 载入的写保护选项字节。
位 23:16	WRP2[7:0]：写保护（Write protect） 该寄存器包含保持由 OBL 载入的写保护选项字节。
位 15:8	WRP1[7:0]：写保护（Write protect） 该寄存器包含保持由 OBL 载入的写保护选项字节。
位 7:0	WRP0[7:0]：写保护（Write protect） 该寄存器包含保持由 OBL 载入的写保护选项字节。

3.4.9 中断向量表偏移寄存器（FLASH_INT_VEC_OFFSET）

偏移地址：0x74

复位值：0x0000 0004

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

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

位 31:14	Res：保留 必须保持复位值。
位 13:0	INT_VEC_OFFSET[13:0]：中断向量表重映射偏移地址（Interrupt Vector table address Offset） *注意：INT_VEC_OFFSET[13:0]的最低两位必须配置为0，可设置的最小偏移量为4。*

若 INT_VEC_OFFSET 设置为 Y_Addr，则：

- 中断向量表重映射后，CPU 访问 0x0800 0000~0x0800 00FF 这一段地址时，实际访问到的地址是:（Y_Addr + 0x0800 0000）~（Y_Addr + 0x0800 00FF）；

- 当 INT_VEC_OFFSET 的值大于或等于 0x100 时，原本物理地址 0x0800 0000 到 0x0800 00FF 中的值将不能被访问到。

注意：若开发一个带 Bootloader 和应用程序（APP）的项目，需要在 APP 工程内调用 Boot 工程中的函数 fun_y，那么函数 fun_y 必须定义在地址 0x80000FF 以后。

下图说明了中断向量表地址映射及偏移的关系：

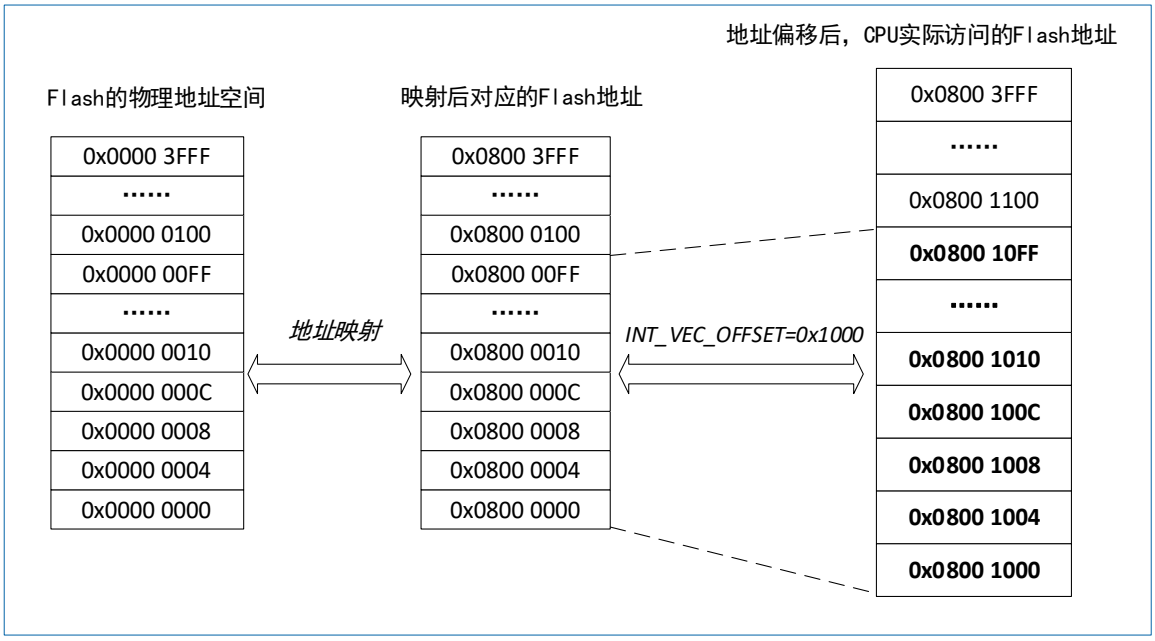

图 3-3 中断向量表地址映射及偏移

CPU 访问 Flash 的地址映射规则：规定了 CPU 每次启动，将访问 0x0000 0000 地址，通过地址映射，实则访问 Flash 地址 0x8000 0000。

INT_VEC_OFFSET = 0x1000：Cortex-M0 内核把中断向量表固定到了 0x0000 0000 地址，设置中断向量表偏移量后，CPU 到 0x0000 0000 地址取中断向量表入口地址时，实际取到的地址是:（0x1000+ 0x0000 0000）。由于默认有地址重映射的，所以取到地址的是（0x1000+ 0x0800 0000）。如此，则不需要通过操作 SYSCFG_CFG1 寄存器中的 MEM_MODE 位来重映射中断向量表了。这在 IAP 升级应用中很方便。

4 CRC 计算单元（CRC）

循环冗余校验（CRC）计算单元用于验证数据传输和数据存储的完整性。CRC 计算单元在运行期间计算出软件的签名，并将其和链接时所产生并存储于在指定存储地址的参考签名进行比较。

4.1 CRC 主要功能

- 采用的 CRC-32（与以太网标准相同）多项式 0x4C11DB7：

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

- 能处理 8 位、16 位和 32 位数据宽度
- 可编程 CRC 初始值
- 单输入/输出 32 位数据寄存器
- 输入缓冲器可避免计算期间发生总线阻塞
- 8 位通用寄存器（可用于临时存储）
- I/O 数据的可反转性选项 CRC 功能说明

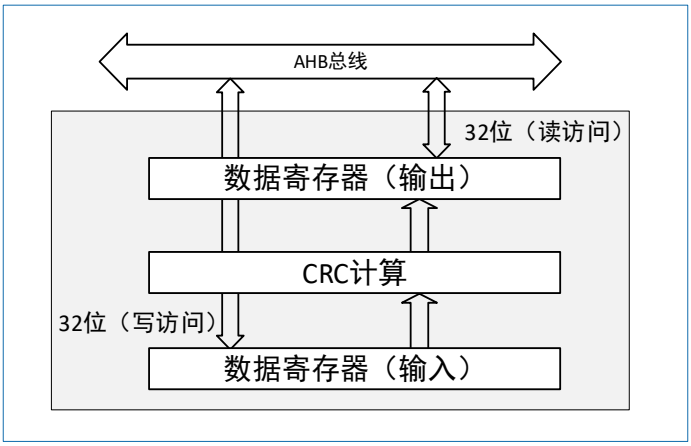

图 4-1 CRC 计算单元框图

对于 32 位数据大小，CRC 计算在 4 个 AHB 时钟周期（HCLK）内完成。

4.2 CRC 功能描述

CRC 计算单元具有单个 32 位读/写数据寄存器（CRC_DR）。CRC_DR 用于保存输入的新数据（写访问）和之前 CRC 计算的结果（读访问）。

对 CRC_DR 寄存器的每次写操作都会对之前的 CRC 值（存于 CRC_DR 中）和新值做一次 CRC 计算。CRC 计算支持整个 32 位数据字或逐个字节计算，具体取决于写入数据的位宽。

CRC_DR 寄存器可按字、右对齐半字和右对齐字节进行访问。对于其他 CRC 寄存器，只支持 32 位访问。

计算时间取决于数据宽度：

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

输入缓冲器中可立即写入第二个数据，无需因之前的 CRC 计算而等待。

CRC 计算单元可动态调整数据大小，从而能最大程度地减少给定字节数的写访问次数。例如，对 5

个字节进行 CRC 计算时，可先写入一个字，然后写入一个字节。

输入数据的顺序可反转，以管理各种数据存放方式（双字/单字/字节、大端/小端等）。可对 8 位、16 位和 32 位数据执行反转操作，具体取决于 CRC_CR 寄存器中的 REV_IN[1:0]位。

例如，输入数据 0x1A2B3C4D 在 CRC 计算中用作：

- 按字节执行位反转后的 0x58D4 3CB2
- 按半字执行位反转后的 0xD458 B23C
- 按全字执行位反转后的 0xB23C D458

通过将 CRC_CR 寄存器中 REV_OUT 位置 1，也可以将输出数据反转。该操作按位进行：例如，输出数据 0x1122 3344 将转换为 0x22CC 4488。配置 CRC_CR 寄存器中的 RESET 控制位可将 CRC 计算器初始化为可编程值（默认值为 0xFFFF FFFF）。

可使用 CRC_INIT 寄存器对 CRC 初始值进行编程。对 CRC_INIT 寄存器进行写访问时，会自动初始化 CRC_DR 寄存器。

CRC_IDR 寄存器可用于保存与 CRC 计算相关的临时值。CRC_IDR 不受 CRC_CR 寄存器中的 RESET 位影响。

4.3 CRC 寄存器

基地址：0x4002 3000

空间大小：0x400

4.3.1 数据寄存器（CRC_DR）

偏移地址：0x00

复位值：0xFFFF FFFF

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

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

位 31:0	DR[31:0]：数据寄存器（Data register） 该寄存器用于存放待计算的新数据，直接将其写入即可。读取该寄存器得到的是上次 CRC 计算的结果。如果读出或写入的数据不足 32 位，则表示该数据仅针对有意义的位。

4.3.2 独立数据寄存器（CRC_IDR）

偏移地址：0x04

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	IDR[7:0]：通用目的 8 位数据寄存器（General purpose 8-bit data register） 该寄存器可用作 1 个字节的临时存储。CRC_CR 寄存器中的 RESET 位引起的复位操作不会影响该寄存器。

4.3.3 控制寄存器（CRC_CR）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								REV_OUT	REV_IN[1:0]		Res				RESET
								rw	rw						rs

位 31:8	Res：保留 必须保持复位值。
位 7	REV_OUT：翻转输出数据（Output data revert） 该位控制输出数据的翻转。 • 0：不翻转 • 1：翻转
位 6:5	REV_IN[1:0]：翻转输入数据（Input data revert） 该位域控制输入数据的翻转。 • 00：不翻转 • 01：按字节为单位翻转 • 10：按半字为单位翻转 • 11：按字为单位翻转
位 4:1	Res：保留 必须保持复位值。
位 0	RESET：复位控制（Reset control） 该位用于复位整个 CRC 计算单元，并将 CRC_INIT 寄存器中的值更新到 CRC_DR 寄存器。 该位由软件置位，由硬件清零。

4.3.4 CRC 初值寄存器（CRC_INIT）

偏移地址：0x10

复位值：0xFFFF FFFF

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

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

位 31:0	CRC_INIT[31:0]：CRC 预置的初值（CRC initiate value） 该寄存器用于设置 CRC 的初值。

5 电源控制（PWR）

5.1 电源

芯片的工作电压（V_{DD}）为 2.4~5.5 V。通过内置的电压调节器提供所需的 1.5 V 电源。

芯片片内数字逻辑的电源由片内 LDO 提供。

片内 LDO 的输出电压可通过寄存器设置，以便于软件根据应用场景最大程度地优化芯片的电流功耗。芯片运行（Run）模式和停机（Stop）模式的 LDO 输出电压可以分别独立设置。

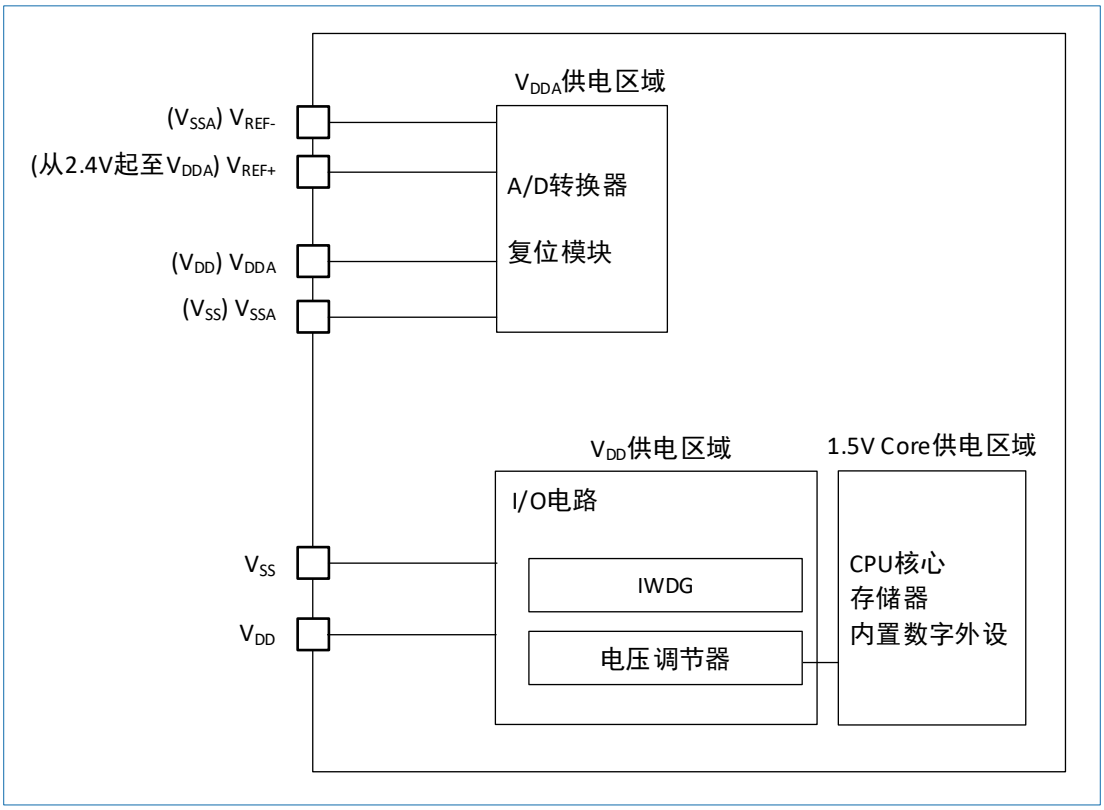

图 5-1 电源框图

注意：V_{DDA} 和 V_{SSA} 必须分别连到 V_{DD} 和 V_{SS}。

5.1.1 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，以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 V_{DDA}
- 独立的电源地 V_{SSA}

没有 V_{REF+}和 V_{REF-}引脚，它们在芯片内部与 V_{DD} 和 V_{SS} 相连。

5.1.2 电压调节器

复位后，电压调节器总是使能的。根据应用方式，调节器工作在以下不同的模式。

- 运行（Run）模式：调节器以正常功耗模式提供 1.5 V 电源（内核、内存和外设）。
- 停机（Stop）模式：调节器以低功耗模式提供 1.5 V 电源，以保存寄存器和 SRAM 的内容。

5.2 电源监控器

该器件集成了一个与欠压复位（BOR）电路耦合的上电复位（POR）/掉电复位（PDR）电路。

- 对于工作在 2.6 V 和 5.5 V 之间的器件，BOR 可配置为上电时使能，以确保器件从 2.6V 开始正常工作。器件达到 BOR 阈值（如 2.6V）后，选项字节加载过程开始启动以确认或修改默认阈值，或永久禁止 BOR。当 BOR 禁能时，POR/PDR 则会使能，V_{DD} 和 POR/PDR 阈值的比较结果决定了器件是否复位。
- 对于工作在 2.4 V 和 5.5 V 之间的器件，BOR 也可选择为禁能（器件上电时间可减少到 10µs）。

可以通过选项字节配置不同的 BOR 阈值，从 2.6 到 5.0V。当 V_{DD} 低于指定阈值 V_{POR}、V_{PDR} 或 V_{BOR} 时，器件无需任何外部复位电路便可保持复位模式。

图 5-2 中说明了不同的电源监控器（POR、PDR、BOR）。

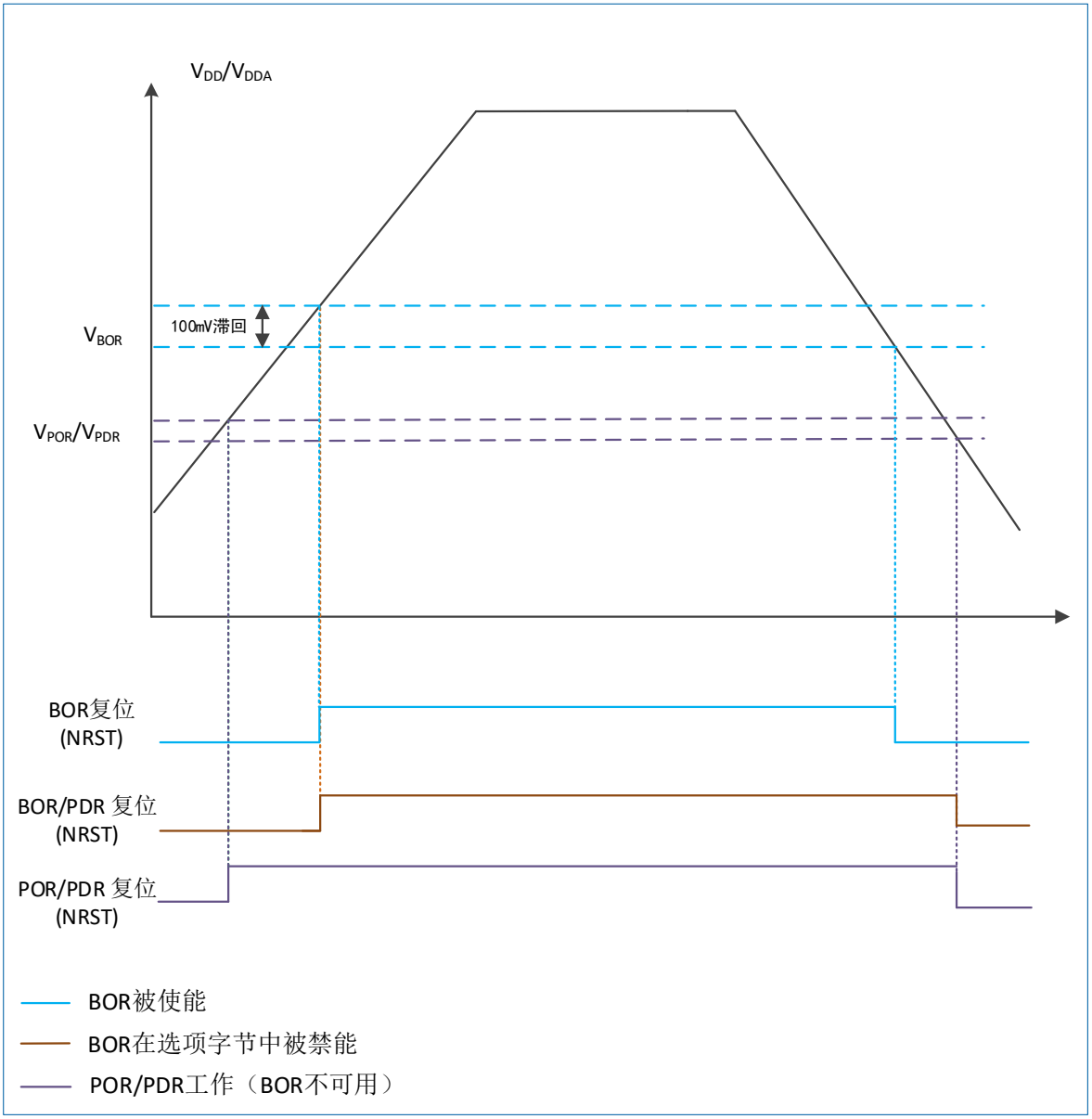

图 5-2 电源监控器

说明：

(1). BOR 功能在工作电压从2.6 到5.5V 的器件上可配置为使能，此时，它会掩盖POR/PDR 阈值。

(2). 如果通过选项字节禁止BOR，当V_{DD} 低于PDR 电平时，会发生复位。

(3). 工作于2.4 到5.5 V 的器件可通过选项字节禁能 BOR，当V_{DD} 高于 POR 电平时，会释放复位，当V_{DD} 低于 PDR 电平时，会发生复位。

5.2.1 上电/掉电复位（POR/PDR）

芯片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 POR/PDR 阈值时，系统即能正常工作。

当 V_{DD}/V_{DDA} 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更多细节，请参考数据手册的电气特性部分。

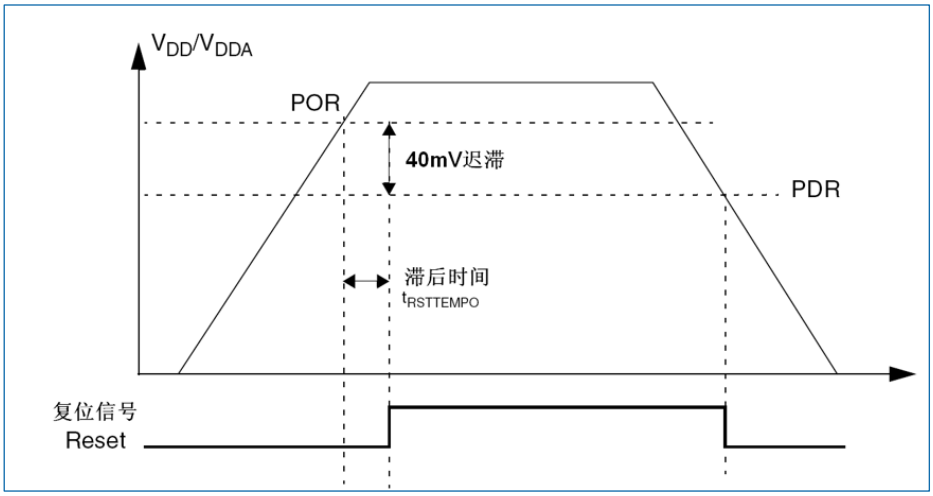

图 5-3 上电复位和掉电复位的波形图

5.2.2 欠压复位（BOR）

上电期间，欠压复位（BOR）将使器件保持复位状态，直到电源电压达到指定的V_{BOR} 阈值。对于工作于2.4V 到5.5 V 间的器件，BOR 可配置为禁能，电源供电由POR/PDR 监控。由于 POR/PDR 阈值低于2.4V，因此在V_{POR}/V_{PDR} 阈值和最低产品工作电压 2.4 V 之间存在一个“盲区”。

当释放系统复位时，可通过加载选项字节重新配置 BOR 阈值级别或禁止BOR。

当通过选项字节禁止BOR 选项时，掉电复位由PDR 控制。

V_{BOR} 通过器件选项字节进行配置。默认情况下，BOR 关闭。可以选择多个可编程 V_{BOR} 阈值：

- BOR 级别 1（VBOR1）：约 2.6 V 的复位阈值级别。
- BOR 级别 2（VBOR2）：约 3.0V 的复位阈值级别。
- BOR 级别 3（VBOR3）：约 3.4 V 的复位阈值级别。
- BOR 级别 4（VBOR4）：约 3.8 V 的复位阈值级别。
- BOR 级别 5（VBOR5）：约 4.2 V 的复位阈值级别。
- BOR 级别 6（VBOR6）：约 4.6V 的复位阈值级别。
- BOR 级别 7（VBOR7）：约 5.0V 的复位阈值级别。

当电源电压（V_{DD}）降至所选V_{BOR} 阈值以下时，将使器件复位。当V_{DD} 高于 V_{BOR} 上限时，释放器件复位，系统可以启动。

通过对器件选项字节进行编程可以禁止BOR。要禁止BOR 功能，V_{DD} 必须高于V_{BOR1}，以启动器件选项字节编程序列。POR 和PDR 将在随后监视上电和掉电（请参见“5.2.1 上电/掉电复位（POR/PDR）”）。BOR 阈值滞回电压约为100 mV（电源电压的上升沿与下降沿之间）。

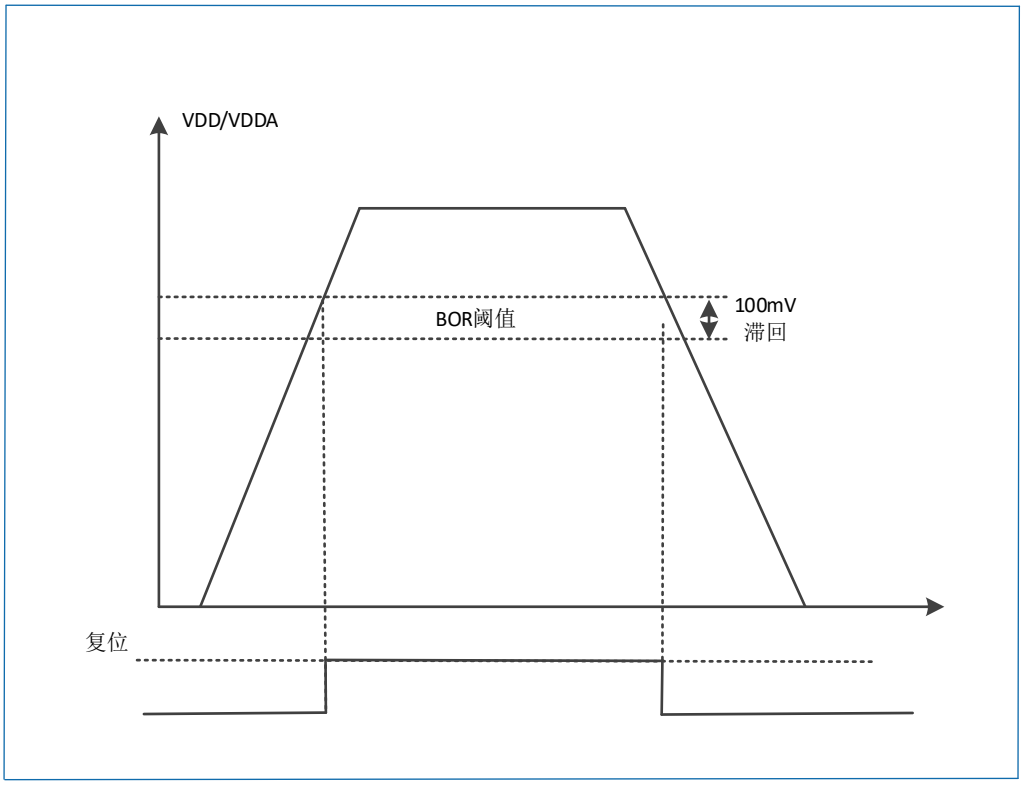

图 5-4 BOR 阈值

5.3 低功耗模式

在系统或电源复位以后，MCU 处于运行状态。当 CPU 不需继续运行时（例如等待某个外部事件时），可以利用多种低功耗模式来节省功耗。用户需要根据最低电源消耗、最短启动时间和可用的唤醒源等条件，选定一个最佳的低功耗模式。

芯片支持以下低功耗模式：

- 睡眠（Sleep）模式

Cortex®-M0 内核停止，所有外设包括 Cortex-M0 核心的外设，如 NVIC、系统时钟（SysTick）等仍在运行。

- 停机（Stop）模式

在停机模式下，所有内核被关闭，HSI 振荡器被关闭。可以通过任一配置成 EXTI 的信号把 MCU 从停机模式中唤醒。

在运行模式下，可以通过以下任意方式降低功耗：

- 降低系统时钟频率。
- 关闭 APB 和 AHB 总线上未被使用的外设时钟。

表 5-1 低功耗模式的进入/唤醒条件

工作模式	进入条件	唤醒条件	内部核电源时钟状态	V_{DD} 主区域时钟状态	电压调节器状态
睡眠模式（Sleep）	1. 设置 PWR_CR:LPDS = 0。 2. 软件执行 WFI/WFE 指令进入。	由任何一个普通 IRQ 中断事件唤醒，包括 SystemTicker。	CPU 时钟关闭，对其他时钟和 ADC 时钟无影响	开启	开启
停机模式	1. 设置 PWR_CR:LPDS = 0。 2. 设置 CM0 系统控制寄存	支持任何一个 EXTI 外	所有时钟停	HSI 关闭	开启或者处于低功耗模

工作模式	进入条件	唤醒条件	内部核电源时钟状态	V_{DD} 主区域时钟状态	电压调节器状态
（Stop）	器的 SLEEPDEEP 位。 3. 软件执行 WFI/WFE 指令进入。	部中断线唤醒。 如果执行 WFI 进入停机模式：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）。可参考章节：“10.1.3 中断和异常向量”。 如果执行 WFE 进入停机模式：设置任一外部中断线为事件模式。可参考章节：“10.2.4 唤醒事件管理”。 支持 AWUT 唤醒。	止		式（在 PWR_CR 中设置）

5.3.1 降低系统时钟

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、PCLK）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频率。参见章节：“6.3.2 时钟配置寄存器（RCC_CFGR）”。

5.3.2 外部时钟的控制

在运行模式下，随时可以通过停止为各外设和内存提供时钟（HCLK 和 PCLK）来减少功耗。在睡眠模式下为了进一步减少功耗，可在执行 WFI 或 WFE 指令前关闭所有外设的时钟。

通过设置 AHB 外设时钟使能寄存器（参见“6.3.6 AHB 外部时钟使能寄存器（RCC_AHBENR）”）APB 外设时钟使能寄存器（参见“6.3.7 APB 外设时钟使能寄存器 2（RCC_APBENR2）和 6.3.8 APB 外设时钟使能寄存器 1（RCC_APBENR1）”）来开关各个外设模块的时钟。

5.3.3 睡眠（Sleep）模式

5.3.3.1 进入睡眠模式

通过执行 WFI 或 WFE 指令进入睡眠状态。本系列芯片可以在进入睡眠模式前，将时钟切换到 LSI。从而进一步降低功耗。根据 Cortex®-M0 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，MCU 立即进入睡眠模式。
- SLEEP-ON-EXIT：如果 SLEEPONEXIT 位被置位，系统从最低优先级的中断处理程序中退出时，MCU 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 5-2 和表 5-3。

5.3.3.2 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（NVIC）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，MCU 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 中使能，并且在 Cortex-M0 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 EXTI 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 NVIC 中断请求（IRQ）通道挂起位。该模式唤醒所需的时间最短，因为中断的进入或退出没有消耗时间。关于如何退出睡眠模式，更多的细节参考表 5-2 和表 5-3。

表 5-2 SLEEP-NOW 模式

SLEEP-NOW 模式	说明
进入条件	在以下条件下，执行 WFI（等待中断）或 WFE（等待事件）指令： ● SLEEPDEEP = 0 ● SLEEPONEXIT = 0 参考 Cortex-M0 系统控制寄存器。
退出条件	● 如果执行 WFI 进入睡眠模式： 中断：参考“10.1.3 中断和异常向量”。 ● 如果执行 WFE 进入睡眠模式： 唤醒事件：参考“10.2.4 唤醒事件管理”。
唤醒延时	无

表 5-3 SLEEP-ON-EXIT 模式

SLEEP-ON-EXIT 模式	说明
进入	在以下条件下，执行 WFI 指令： ● SLEEPDEEP = 0 ● SLEEPONEXIT = 1 参考 Cortex-M0 系统控制寄存器。
退出	中断：参考“10.1.3 中断和异常向量”。
唤醒延时	无

5.3.4 停机（Stop）模式

停机模式是在 Cortex®-M0 的深睡眠模式基础上结合了外设的时钟控制机制。在停机模式下，电压调节器可运行在正常或低功耗模式。此时，在 1.5V 供电区域的所有时钟都停止，HSI 振荡器的功能禁用，SRAM 和寄存器内容被保存下来。

在停机模式下，部分的 I/O 引脚都保持它们在运行模式时的状态。

5.3.4.1 进入停机模式

关于如何进入停机模式，参见表 5-1。

在停机模式下，通过设置电源控制寄存器（PWR_CR）的 LPDS 位使内部调节器进入低功耗模式，能够降低更多的功耗。

如果正在进行 Flash 编程，直到对 Flash 访问完成，系统才进入停机模式。如果正在进行对 APB 的访问，直到对 APB 访问完成，系统才进入停机模式。停机模式下，可以通过设置独立的控制位，选择以下功能：

- 独立看门狗（IWDG）：可通过写入看门狗的关键字寄存器或硬件选择来启动 IWDG。独立看门狗一旦启动，除非系统复位，否则它不能被停止。参见 IWDG 功能描述章节。
- 内部 RC 振荡器（LSI RC）：通过控制/状态寄存器（RCC_CSR）的 LSION 位来设置。

在停机模式下，如果在进入该模式前 ADC 没有被关闭，那么这些外设仍然耗电。可在进入该模式前，通过设置寄存器 ADC_CR 的 ADEN 位为 0，可关闭该外设。

5.3.4.2 退出停机模式

关于如何退出停机模式，可参见表 5-1。当一个中断或唤醒事件导致退出停机模式时，HSI RC 振荡器被选为系统时钟。

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

5.3.5 调试模式

默认情况下，如果在进行调试 MCU 时，使 MCU 进入停机模式，将断开调试连接。这是因为 Cortex®-M0 的内核的时钟被禁用。

然而，通过设置 DBGMCU_CR 寄存器中的某些配置位，可以在低功耗模式下调试软件。更多的细节请参考章节：“22.8.1 对低功耗模式的调试支持”。

5.4 PWR 寄存器

基地址：0x4000 7000

空间大小：0x400

5.4.1 电源控制寄存器（PWR_CR）

偏移地址：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															LPDS
															rw

位 31:1	Res：保留 必须保持复位值。
位 0	LPDS：电压调节器状态（Low power deep sleep） 该位用软件设置或清除。 • 0：在停机模式下，电压调节器开启（处于正常功耗模式）。 • 1：在停机模式下，电压调节器处于低功耗模式。

5.4.2 电源控制/状态寄存器（PWR_CSR）

偏移地址：0x04

复位值：0x0000 000X

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												LDORDY	Res		
												r			

位 31:4	Res：保留 必须保持复位值。
位 3	LDORDY：LDO 状态标志（LDO status flag） • 0：表示内部 LDO 不能带动重负载，不允许主频切换至高速时钟。 • 1：表示内部 LDO 稳定，可带动重负载。此时允许主频切换至高速时钟。
位 2:0	Res：保留 必须保持复位值。

6 复位和时钟控制（RCC）

6.1 复位

芯片有两种复位方式：系统复位、电源复位。

6.1.1 系统复位

系统复位将复位除时钟控制寄存器 CSR 中的复位标志和备份寄存器以外的所有寄存器为它们的复位值。

当以下任一事件发生时，将产生系统复位：

- NRST 引脚上的低电平（外部复位）
- 窗口看门狗事件（WWDG 复位）
- 独立看门狗事件（IWDG 复位）
- 电源复位
- 软件复位（SW 复位）
- 低功耗管理复位

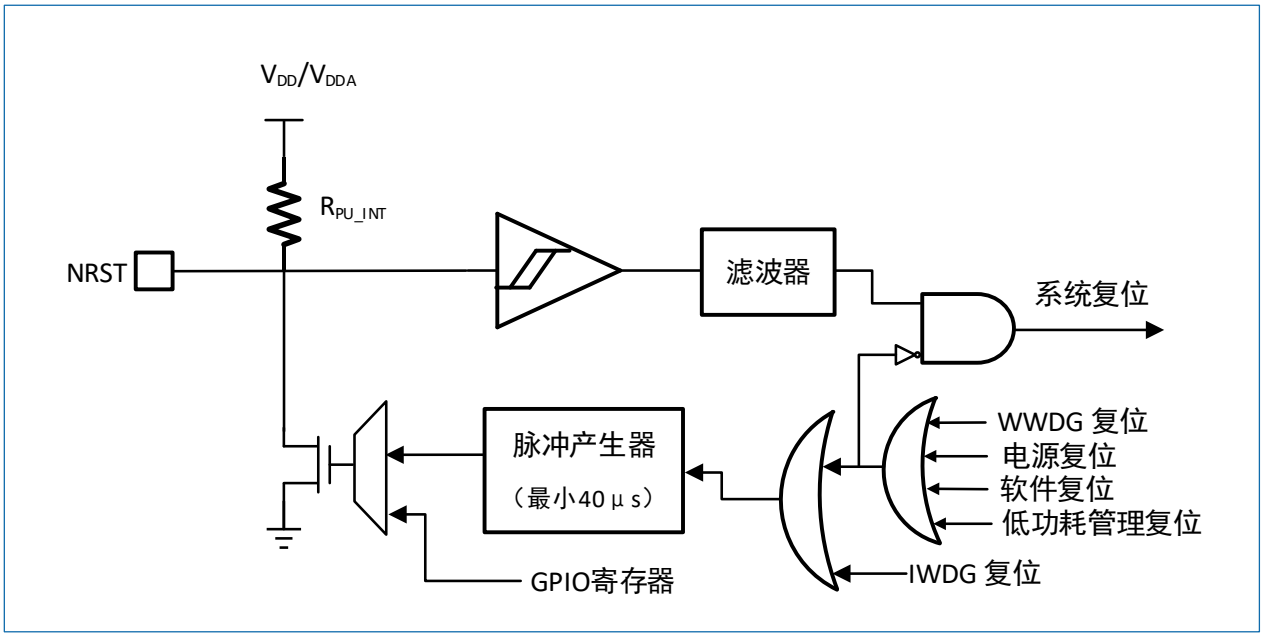

图 6-1 复位电路

电源复位将复位除了备份区域外的所有寄存器。复位源将最终作用于 NRST 引脚，并在复位过程中保持低电平。复位入口矢量被固定在地址 0x0000 0004。

芯片内部的复位信号会在 NRST 引脚上输出。脉冲发生器保证每一个（外部或内部）复位源都能有至少 40 μs 的脉冲延时。当 NRST 引脚被拉低产生外部复位时，它将产生复位脉冲。

软件复位

通过将 Cortex-M0 应用中断和复位控制寄存器中的 SYSRESETREQ 位置 1，可实现软件复位。请参考 Cortex-M0 技术参考手册获得进一步信息。

低功耗管理复位

在进入停机模式时产生低功耗管理复位：

通过将用户选择字节中的 nRST_STOP 位置 0，使能该复位。这样，即使器件正处于进入停机模式的进程，系统也将被复位而不是进入停机模式。

6.1.2 电源复位

当以下任一事件发生时，将产生电源复位：

- 上电/掉电复位（POR/PDR）
- 欠压复位（BOR）

6.2 时钟

支持多种时钟源驱动系统时钟：

- 内部高速时钟（HSI）：48MHz
- 内部低速时钟（LSI）：60kHz
- GPIO 外部输入时钟：（最高频率 32 MHz）

每种时钟源都可以独立地打开或关断。当它们不用时，可以将其关断来降低功耗。有多个分频器可用于配置 AHB 和 APB 时钟域，AHB 和 APB 域的最高时钟频率为 48 MHz。Cortex 系统定时器由 AHB 时钟（HCLK）驱动，其可由 AHB/8 时钟频率直接驱动（通过 Cortex Systick 位来配置）。

所有的外设时钟由其所在的总线时钟（HCLK 或 PCLK）驱动，以下除外：

- 闪存编程接口时钟（FLITFCLK）由 HSI 时钟或者 SYSCLK 驱动（由软件选择）。
- I2C 的时钟为下列的时钟源之一（由软件选择）：
  - SYSCLK
  - HSI 48 MHz 通过 I2C-HSI 分频器分频
  - APB 时钟（PCLK）
- Cortex 的 FCLK 由 AHB 时钟（HCLK）直接提供。
- AHB 总线、ARM 内核、存储器由 AHB 时钟（HCLK）直接提供。
- ADC 时钟由下列之一时钟得到（由软件选择 ）：
  - APB 时钟（PCLK）由 ADC 分频器 2/4 分频
  - HSI 48 MHz 由 ADC-HSI 分频器分频
- UART1/UART2 的时钟为下列的时钟源之一（由软件选择）：
  - PCLK
  - SYSCLK
  - HSI 48 MHz RC 振荡器时钟通过 UART-HSI 分频器的分频时钟
- Timer 时钟由 APB 时钟（PCLK）或者 APB 时钟（PCLK）2 倍频提供（由软件设置，硬件判定执行）。
- RCC 将 AHB 时钟（HCLK）8 分频后作为 Cortex 系统定时器（SysTick）的外部时钟。通过对 SysTick 控制与状态寄存器的设置，可选择 HCLK/8 时钟作为 SysTick 时钟。

以下是 HK32F0301MxxxxC 系列芯片的时钟树：

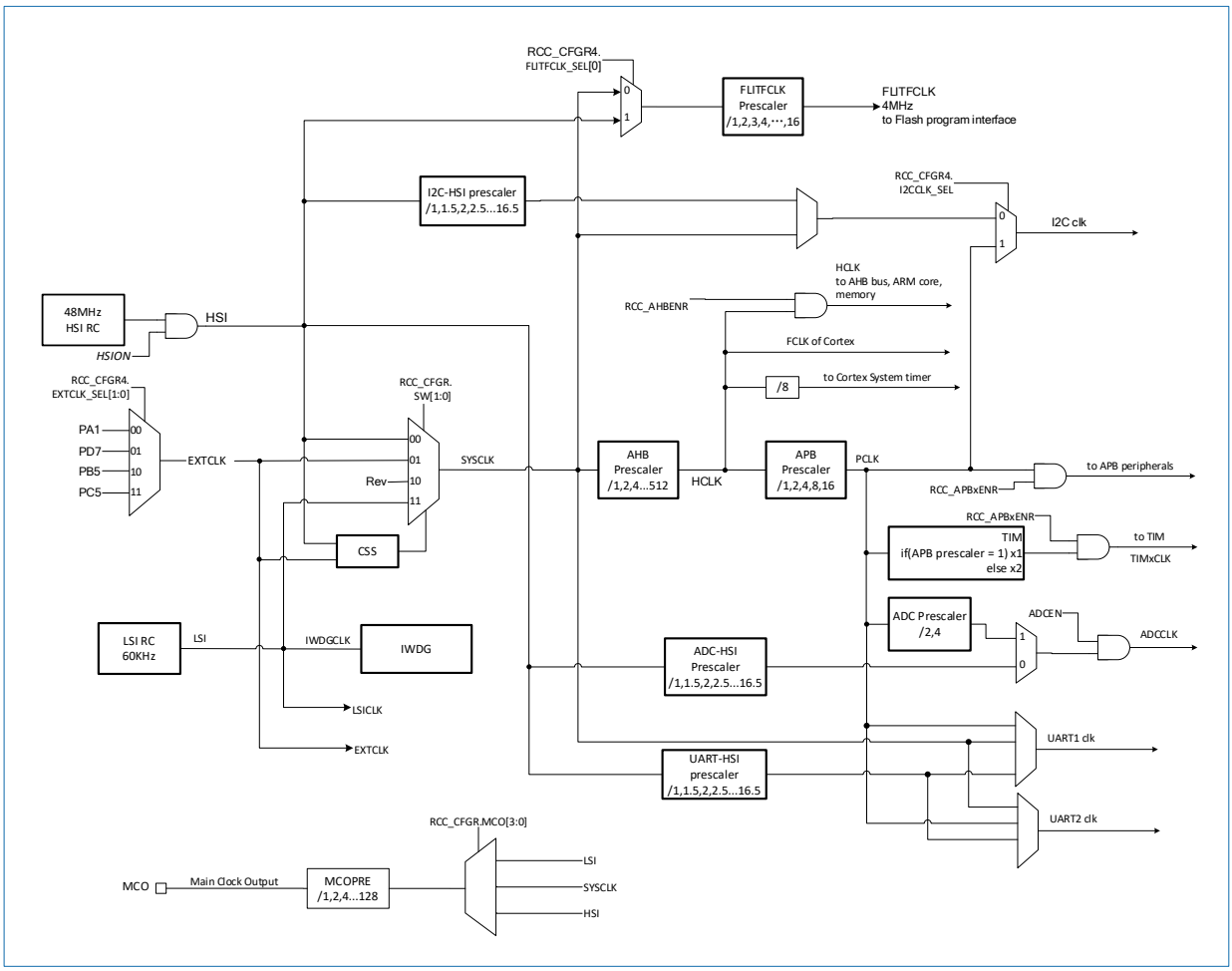

图 6-2 HK32F0301MxxxxC 时钟树

6.2.1 HSI 时钟

HSI 时钟信号由 48 MHz 内部 RC 振荡器生成的，可直接作为系统时钟。从复位重启后，HSI 时钟始终用作系统时钟。

HSI RC 振荡器能够在不需要任何外部器件的条件下提供系统时钟。

校准

制造工艺决定了不同芯片的 RC 振荡器频率会不同，这就是为什么每个芯片的 HSI 时钟频率在出厂前已经被校准到 2%（25°C）的原因。系统复位时，工厂校准值被装载到时钟控制寄存器的 HSICAL[5:0]位和 HSITRIM[5:0]（HSICAL 和 HSITRIM 在时钟控制寄存器 RCC_CR 中）。

如果用户的应用基于不同的电压或环境温度，这会影响 RC 振荡器的精度。可以通过时钟控制寄存器（RCC_CR）里的 HSITRIM[5:0]位来调整 HSI 频率。

时钟控制寄存器（RCC_CR）中的 HSIRDY 位用来指示 HSI RC 振荡器是否稳定。在时钟启动过程中，需等待此位被硬件置 1 后，才可以使用 HSI RC 输出时钟。

HSI RC 可由时钟控制寄存器（RCC_CR）中的 HSION 位来启动和关闭。

6.2.2 GPIO 外部时钟输入

在这个模式里，必须提供外部时钟。用户可以通过外部 GPIO 输入时钟作为 SYSCLK。通过 RCC_CFGR4.EXTCLK_SEL[1:0]选择作为外部时钟输入的 IO。用户可通过设置在时钟控制寄存器（RCC_CR）中的 EXTCLK 位来选择这一模式。占空比为 45%~55%的外部时钟信号（方波、正弦波或三角波）必须连到对应的外部 GPIO 引脚上。

用户还可配置时钟控制寄存器（RCC_CR）中的 CSSON 位来打开/关闭 GPIO 时钟检测功能。

6.2.3 LSI 时钟

LSI RC 可作为低功耗时钟源在停机模式下保持运行，供独立看门狗（IWDG）使用。时钟频率在 60 kHz 左右。由于 LSI 的特性，会有±10%的误差。

LSI RC 振荡器可通过控制/状态寄存器（RCC_CSR）中的 LSION 位打开或关闭。

控制/状态寄存器（RCC_CSR）中的 LSIRDY 标志指示低速内部振荡器是否稳定。在其启动后，需等待硬件将该位置 1 后，才能使用此时钟。如果在 RCC_CIR 中使能中断，则可产生中断。

自 IWDG 激活后，LSI 振荡器不能通过 LSION=0 停止。LSI 振荡器通过系统复位停止（但通过硬件使用 FLASH_OBR 寄存器中的 WDG_SW 位使能 IWDG 的情况除外）。如果 IWDG 已通过软件使能，则必须在系统复位后再次使能 LSI 振荡器，以确保 IWDG 正常工作。

可通过测量 LSI 振荡器的频散来获得精度水平可接受的 IWDG 超时。

6.2.4 系统时钟（SYSCLK）选择

可以使用几种不同的时钟源来驱动系统时钟（SYSCLK）：

- HSI 振荡器
- LSI 振荡器
- GPIO 外部输入时钟

系统复位后，HSI 振荡器被选为系统时钟。当时钟源被直接作为系统时钟时，它将不能被停止。

时钟源的切换只有在目标时钟源可用的情况下，通过设置寄存器位 RCC_CFGR.SW SW[1:0]来配置。假如系统选择了未准备好的时钟源作为当前系统时钟，那么只有在目标时钟源准备好之后才真正执行切换时钟源的操作。时钟配置寄存器（RCC_CFGR）中的 SWS[1:0]指示当前系统时钟采用哪个时钟源作为系统时钟。

6.2.5 看门狗时钟

如果独立看门狗（Independent watchdog，IWDG）已通过硬件选项或软件访问的方式启动，则 LSI 振荡器将被强制打开，且不能禁止。在 LSI 振荡器稳定后，时钟将提供给 IWDG。

如果已通过软件使能 IWDG，则系统复位后会禁止 LSI。之后，必须再次使能 LSI 振荡器，以确保 IWDG 正常工作。

6.2.6 时钟输出功能（MCO）

MCU 时钟输出（Microcontroller clock output，MCO）功能允许使用可配置的预分频值（1、2、4、8、16、32、64、128）将时钟输出到外部 MCO 引脚上。必须在复用功能模式下对相应 GPIO 端口的配置寄存器进行编程。可选择以下 3 种时钟信号之一作为 MCO 时钟：

- LSI
- SYSCLK
- HSI

时钟信号选择由时钟配置寄存器（RCC_CFGR）中的 MCO[3:0]位控制。

6.3 RCC 寄存器

基地址：0x4002 1000

空间大小：0x400

6.3.1 时钟控制寄存器（RCC_CR）

偏移地址：0x00

复位值：0x0000 XXX3

访问：无等待状态，字、半字和字节访问。

说明：x 表示不定值。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CSSON	Res	EXTCLKRDY	EXTCLKON
												rw		r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		HSITRIM[5:0]						HSICAL[5:0]						HSIRDY	HSION
		rw						rw						r	rw

位 31:20	Res：保留 必须保持复位值。
位 19	CSSON：打开/关闭时钟检测（Clock Security System enable） 当 EXTCLKON 被置位后，等待 EXTCLKRDY 变为 1，CSS 硬件就开始检测 GPIO 输入时钟频率。如果检测到 GPIO 输入时钟频率低于 CSS_THRESHOLD 设定阈值，则置位 CSSF 并产生 NMI，然后停止检测 GPIO 输入时钟。该位由软件置位和清零。 • 0：关闭时钟检测。 • 1：开启时钟检测（如果 GPIO 输入时钟没有稳定，则自动关闭时钟检测）。
位 18	Res：保留 必须保持复位值。
位 17	EXTCLKRDY：指示 GPIO 输入时钟是否稳定（External clock ready flag） • 0：GPIO 输入时钟未稳定 • 1：GPIO 输入时钟稳定
位 16	EXTCLKON：GPIO 输入时钟使能（External clock enable） • 0：关闭 GPIO 输入时钟 • 1：使能 GPIO 输入时钟
位 15:14	Res：保留 必须保持复位值。
位 13:8	HSITRIM[5:0]：HSI 精调（HSI clock trimming） 精调步进为0.2%，值越大，设置的频率越高。
位 7:2	HSICAL[5:0]：HSI 粗调（HSI clock calibration） 该位的复位值由出厂设置而定。粗调步进为2%，值越大，设置的频率越高。
位 1	HSIRDY：HSI 就绪标志（HSI clock ready flag） • 0：HSI 未稳定 • 1：HSI 已稳定

位 0	HSION：HSI 使能（HSI clock enable） 该位可由软件置位或清除。 当从 Stop 模式唤醒时，硬件自动将该位置 1；当 GPIO 被用作 SYSCLK 且 CSS 检测到 GPIO 输入时钟异常，硬件也自动将该位置 1。 选择 HSI 作为 SYSCLK 时，软件不能清除该位。 • 0：关闭 HSI • 1：打开 HSI

6.3.2 时钟配置寄存器（RCC_CFGR）

偏移地址：0x04

复位值：0x0000 0010

访问：无等待周期，支持字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	MCOPRE[2:0]			MCO[3:0]				Res							
	rw			rw											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res					PPRE[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
					rw			rw				r		rw	

位 31	Res：保留 必须保持复位值。
位 30:28	MCOPRE[2:0]：MCU 时钟输出（MCO）分频系数（Microcontroller clock output prescaler） • 000：MCO/1 • 001：MCO/2 • 010：MCO/4 • … • 111：MCO/128
位 27:24	MCO[3:0]：MCU 时钟输出选择（Microcontroller clock output） 该位由软件设置和清除。 • 0000：MCO 无时钟输出 • 0010：MCO 输出 LSI • 0100：MCO 输出 SYSCLK • 0101：MCO 输出 HSI • 其他值：MCO 无时钟输出
位 23:11	Res：保留 必须保持复位值。
位 10:8	PPRE[2:0]：PCLK 的分频系数（PCLK prescaler） • 0xx：HCLK/1 • 100：HCLK/2

	• 101：HCLK/4 • 110：HCLK/8 • 111：HCLK/16
位 7:4	HPRE[3:0]：HCLK 的分频系数（HCLK prescaler） • 0001：SYSCLK/6 • 1000：SYSCLK/2 • 1001：SYSCLK/4 • 1010：SYSCLK/8 • 1011：SYSCLK/16 • 1100：SYSCLK/64 • 1101：SYSCLK/128 • 1110：SYSCLK/256 • 1111：SYSCLK/512 • 其他值：SYSCLK/1
位 3:2	SWS[1:0]：系统时钟切换状态（System clock switch status） 该位域由硬件自动改写值。指示当前 SYSCLK 所选择的时钟源。 • 00：HSI 用作 SYSCLK • 01：GPIO 输入时钟用作 SYSCLK • 10：保留 • 11：LSI 用作 SYSCLK
位 1:0	SW[1:0]：系统时钟切换（System clock switch） • 00：选择 HSI 作为 SYSCLK • 01：选择 GPIO 输入时钟作为 SYSCLK • 10：保留 • 11：选择 LSI 作为 SYSCLK

6.3.3 时钟中断寄存器（RCC_CIR）

偏移地址：0x08

复位值：0x0000 0000

访问：无等待状态，字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								CSSC	Res			EXTRDYC	HSIRDYC	Res	LSIRDYC
								w				w	w		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				EXTRDYIE	HSIRDYIE	Res	LSIRDYIE	CSSF	Res			EXTRDYF	HSIRDYF	Res	LSIRDYF
				rw	rw		rw	r				r	r		r

位 31:24	Res：保留 必须保持复位值。
位 23	CSSC：清除安全时钟标志（Clock security system interrupt clear）

	• 0：无作用 • 1：清除 CSSF 标志
位 22:20	Res：保留 必须保持复位值。
位 19	EXTRDYC：清除 EXTRDYF 的标志（External clock ready interrupt clear） 该位由软件设置。 • 0：无作用 • 1：清除 EXTRDYF 标志
位 18	HSIRDYC：清除 HSIRDYF 的标志（HSI ready interrupt clear） 该位由软件设置。 • 0：无作用 • 1：清除 HSIRDYF 标志
位 17	Res：保留 必须保持复位值。
位 16	LSIRDYC：清除 LSIRDYF 的标志（LSI ready interrupt clear） 该位由软件设置。 • 0：无作用 • 1：清除 LSIRDYF 标志
位 15:12	Res：保留 必须保持复位值。
位 11	EXTRDYIE：关闭/使能 GPIO 时钟稳定中断（External clock ready interrupt enable） • 0：关闭 GPIO 时钟稳定中断 • 1：使能 GPIO 时钟稳定中断
位 10	HSIRDYIE：关闭/使能 HSI 时钟稳定中断（HSI ready interrupt enable） • 0：关闭 HSI 时钟稳定中断 • 1：使能 HSI 时钟稳定中断
位 9	Res：保留 必须保持复位值。
位 8	LSIRDYIE：关闭/使能 LSI 时钟稳定中断（LSI ready interrupt enable） • 0：关闭 LSI 时钟稳定中断 • 1：使能 LSI 时钟稳定中断
位 7	CSSF：时钟安全系统中断标志（Clock security system interrupt flag） 当检测到 GPIO 输入时钟频率异常，硬件置位 CSSF；软件对 CSSC 写 1 时，清除 CSSF。 • 0：没有触发 GPIO 输入频率异常 • 1：检测到 GPIO 输入频率异常

位 6:4	Res：保留 必须保持复位值。
位 3	EXTRDYF：外部时钟稳定中断标志（External clock ready interrupt flag） 当 GPIO 输入时钟变为稳定态时，由硬件置位；软件写 EXTRDYC，则将该位清零。 • 0：没有触发 GPIO 输入时钟稳定中断 • 1：触发 GPIO 输入时钟稳定中断
位 2	HSIRDYF：HSI 时钟稳定中断标志（HSI ready interrupt flag） 当打开 HSI 时钟后，HSI 时钟变为稳定态时由硬件置位。只有通过软件配置 HSION 使 HSI 变为稳定态，硬件才会置 HSIRDYF 为 1。软件对 HSIRDYC 写 1 时，清零 HSIRDYF。 • 0：没有触发 HSI 时钟稳定中断 • 1：触发 HSI 时钟稳定中断
位 1	Res：保留 必须保持复位值。
位 0	LSIRDYF：LSI 时钟稳定中断（LSI ready interrupt flag） 当 LSI 变为稳定态时，由硬件置位；软件对 LSIRDYC 写 1 时，清零 LSIRDYF。 • 0：没有触发 LSI 时钟稳定中断 • 1：触发 LSI 时钟稳定中断

6.3.4 APB 外设复位寄存器 2（RCC_APBRSTR2）

偏移地址：0x0C

复位值：0x0000 0000

访问：无等待周期，支持字、半字和字节访问

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	UART1RST	Res	SPIRST	TIM1RST	Res	ADCRST	Res								SYSCFGRST
	rw		rw	rw		rw									rw

位 31:23	Res：保留 必须保持复位值。
位 22	DBGMCURST：调试 MCU 复位（Debug MCU reset） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位调试 MCU
位 21:15	Res：保留 必须保持复位值。

位 14	UART1RST：将 UART1 复位（Reset UART1） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 UART1
位 13	Res：保留 必须保持复位值。
位 12	SPIRST：将 SPI 复位（Reset SPI） 该位由软件置1 或清0。 • 0：无作用 • 1：复位 SPI
位 11	TIM1RST：将 TIM1 定时器复位（Reset TIM1） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 TIM1 定时器
位 10	Res：保留 必须保持复位值。
位 9	ADCRST：将 ADC 接口复位（Reset ADC interface） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 ADC 接口
位 8:1	Res：保留 必须保持复位值。
位 0	SYSCFGRST：将 SYSCFG 复位（Reset SYSCFG） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 SYSCFG

6.3.5 APB 外设复位寄存器 1（RCC_APBRSTR1）

偏移地址：0x10

复位值：0x0000 0000

访问：无等待周期，支持字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	IOMUXRST	Res	PWRRST	Res						I2CRST	Res			UART2RST	AWURST
	rw		rw							rw				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				WWDGRST	Res						TIM6RST	Res			TIM2RST
				rw							rw				rw

位 31	Res：保留 必须保持复位值。
位 30	IOMUXRST：将引脚功能多重映射复位（Reset IOMUX） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位引脚功能多重映射
位 29	Res：保留 必须保持复位值。
位 28	PWRRST：将电源接口复位（Reset power interface） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位电源接口
位 27:22	Res：保留 必须保持复位值。
位 21	I2CRST：将 I2C 复位（Reset I2C） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 I2C
位 20:18	Res：保留 必须保持复位值。
位 17	UART2RST：将 UART2 复位（Reset UART2） • 0：无作用 • 1：复位 UART2
位 16	AWURST：停机模式下的自动唤醒复位（Auto_wake up reset in stop mode） • 0：无作用 • 1：复位自动唤醒
位 15:12	Res：保留 必须保持复位值。
位 11	WWDGRST：将窗口看门狗复位（Reset Window watchdog） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位窗口看门狗

位 10:5	Res：保留 必须保持复位值。
位 4	TIM6RST：将 TIM6 定时器复位（Reset TIM6） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 TIM6
位 3:1	Res：保留 必须保持复位值。
位 0	TIM2RST：将 TIM2 定时器复位（Reset TIM2） 该位由软件置 1 或清 0。 • 0：无作用 • 1：复位 TIM2

6.3.6 AHB 外部时钟使能寄存器（RCC_AHBENR）

偏移地址：0x14

复位值：0x0000 0014

访问：无等待周期，支持字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res											IOPDEN	IOPCEN	IOPBEN	IOPAEN	Res
											rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									CRCEN	Res	FLITFEN	Res	SRAMEN	Res	
									rw		rw		rw		

位 31:21	Res：保留 必须保持复位值。
位 20	IOPDEN：GPIOD 时钟使能（I/O port D clock enable） 该位由软件置 1 或清 0。 • 0：GPIOD 时钟关闭 • 1：GPIOD 时钟开启
位 19	IOPCEN：GPIOC 时钟使能（I/O port C clock enable） 该位由软件置 1 或清 0。 • 0：GPIOC 时钟关闭 • 1：GPIOC 时钟开启
位 18	IOPBEN：GPIOB 时钟使能（I/O port B clock enable） 该位由软件置 1 或清 0。 • 0：GPIOB 时钟关闭

	• 1：GPIOB 时钟开启
位 17	IOPAEN：GPIOA 时钟使能（I/O port A clock enable） 该位由软件置 1 或清 0。 • 0：GPIOA 时钟关闭 • 1：GPIOA 时钟开启
位 16:7	Res：保留 必须保持复位值。
位 6	CRCEN：CRC 时钟使能（CRC clock enable） 该位由软件置 1 或清 0。 • 0：CRC 时钟关闭 • 1：CRC 时钟开启
位 5	Res：保留 必须保持复位值。
位 4	FLITFEN：FLITF 时钟使能（Flash interface clock enable） 该位由软件置 1 或清 0 来开启/关闭在睡眠模式下的 FLITF 时钟。 • 0：在睡眠模式下 FLITF 时钟关闭 • 1：在睡眠模式下 FLITF 时钟开启
位 3	Res：保留 必须保持复位值。
位 2	SRAMEN：SRAM 接口时钟使能（SRAM interface clock enable） 该位由软件置 1 或清 0 来开启/关闭在睡眠模式下的 SRAM 时钟。 • 0：在睡眠模式下 SRAM 接口时钟关闭 • 1：在睡眠模式下 SRAM 接口时钟开启
位 1:0	Res：保留 必须保持复位值。

6.3.7 APB 外设时钟使能寄存器 2（RCC_APBENR2）

偏移地址：0x18

复位值：0x0000 0000

访问：支持字、半字和字节访问；无等待周期，除了上一次的 APB 访问未完成的情况下，必须插入等待周期直到该次访问完成。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	UART1EN	Res	SPIEN	TIM1EN	Res	ADCEN	Res								SYSCFGEN
	rw		rw	rw		rw									rw

位 31:23	Res：保留 必须保持复位值。
位 22	DBGMCUEN：MCU 调试模块时钟使能（MCU debug module clock enable） 该位由软件置1 或清0。 • 0：MCU 调试模块时钟关闭 • 1：MCU 调试模块时钟开启
位 21:15	Res：保留 必须保持复位值。
位 14	UART1EN：UART1 时钟使能（UART1 clock enable） 该位由软件置1 或清0。 • 0：UART1 时钟关闭 • 1：UART1 时钟开启
位 13	Res：保留 必须保持复位值。
位 12	SPIEN：SPI 时钟使能（SPI clock enable） 该位由软件置1 或清0。 • 0：SPI 时钟关闭 • 1：SPI 时钟开启
位 11	TIM1EN：TIM1 定时器时钟使能（TIM1 timer clock enable） 该位由软件置1 或清0。 • 0：TIM1 定时器时钟关闭 • 1：TIM1 定时器时钟开启
位 10	Res：保留 必须保持复位值。
位 9	ADCEN：ADC 接口时钟使能（ADC interface clock enable） 该位由软件置1 或清0。 • 0：ADC 接口时钟关闭 • 1：ADC 接口时钟开启
位 8:1	Res：保留 必须保持复位值。
位 0	SYSCFGEN：SYSCFG 时钟使能（SYSCFG clock enable）

	该位由软件置1 或清0。 • 0：SYSCFG 时钟关闭 • 1：SYSCFG 时钟开启

6.3.8 APB 外设时钟使能寄存器 1（RCC_APBENR1）

偏移地址：0x1C

复位值：0x0000 0000

访问：支持字、半字和字节访问；无等待周期，除了上一次的 APB 访问未完成的情况下，必须插入等待周期直到该次访问完成。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	IOMUXEN	Res	PWREN	Res						I2CEN	Res			UART2EN	AWUEN
	rw		rw							rw				rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				WWDGEN	Res						TIM6EN	Res			TIM2EN
				rw							rw				rw

位 31	Res：保留 必须保持复位值。
位 30	IOMUXEN：引脚功能多重映射时钟使能（IOMUX clock enable） 该位由软件置 1 或清 0。 • 0：引脚功能多重映射功能时钟关闭 • 1：引脚功能多重映射功能时钟开启
位 29	Res：保留 必须保持复位值。
位 28	PWREN：电源控制接口时钟使能（Power interface clock enable） 该位由软件置 1 或清 0。 • 0：电源控制接口时钟关闭 • 1：电源控制接口时钟开启
位 27:22	Res：保留 必须保持复位值。
位 21	I2CEN：I2C 时钟使能（I2C clock enable） 该位由软件置 1 或清 0。 • 0：I2C 时钟关闭 • 1：I2C 时钟开启
位 20:18	Res：保留 必须保持复位值。
位 17	UART2EN：UART2 时钟使能（UART2 clock enable）

	该位由软件置 1 或清 0。 • 0：UART2 时钟关闭 • 1：UART2 时钟开启
位 16	AWUEN：Stop 模式的自动唤醒使能（Auto-wake up clock enable in stop mode） 该位由软件置 1 或清 0。 • 0：自动唤醒关闭 • 1：自动唤醒开启
位 15:12	Res：保留 必须保持复位值。
位 11	WWDGEN：窗口看门狗时钟使能（Window watchdog clock enable） 该位由软件置 1 或清 0。 • 0：看门狗时钟关闭 • 1：看门狗时钟开启
位 10:5	Res：保留 必须保持复位值。
位 4	TIM6EN：TIM6 定时器时钟使能（TIM6 timer clock enable） 该位由软件置 1 或清 0。 • 0：TIM6 时钟关闭 • 1：TIM6 时钟开启
位 3:1	Res：保留 必须保持复位值。
位 0	TIM2EN：TIM2 定时器时钟使能（TIM2 timer clock enable） 该位由软件置 1 或清 0。 • 0：TIM2 时钟关闭 • 1：TIM2 时钟开启

6.3.9 控制/状态寄存器（RCC_CSR）

偏移地址：0x24

复位值：0x0000 0000

除了复位标志（RMVF）外，RCC_CSR 寄存器的其他位域由系统复位进行复位，RMVF 位由电源复位清除。

访问：0 到3 等待周期，支持字、半字和字节访问；当连续对该寄存器进行访问时，将插入等待状态。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		IWDGRSTF	Res	PORRSTF	PINRSTF	Res	RMVF	Res							
		r		r	r		r								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														LSIRDY	LSION
														r	rw

位 31:30	Res：保留 必须保持复位值。
位 29	IWDGRSTF：独立看门狗复位标志（Independent watchdog reset flag） 在独立看门狗复位发生时，该位由硬件置 1。 该位由软件通过写 RMVF 位清除。 • 0：无看门狗复位发生 • 1：发生看门狗复位
位 28	Res：保留 必须保持复位值。
位 27	PORRSTF：上电/掉电复位标志（POR/PDR reset flag） 在 NRST 引脚复位发生时，该位由硬件置 1。 该位由软件通过写 RMVF 位清除。 • 0：无 NRST 引脚复位发生 • 1：发生 NRST 引脚复位
位 26	PINRSTF：NRST 引脚复位标志（PIN reset flag） 在 NRST 引脚复位发生时，该位由硬件置 1。 该位由软件通过写 RMVF 位清除。 • 0：无 NRST 引脚复位发生 • 1：发生 NRST 引脚复位
位 25	Res：保留 必须保持复位值。
位 24	RMVF：清除复位标志（Remove reset flag） 由软件置 1 来清除复位标志。 • 0：无作用 • 1：清除复位标志
位 23:2	Res：保留 必须保持复位值。
位 1	LSIRDY：LSI 振荡器就绪（LSI oscillator ready） 通过硬件置 1 或清 0 来指示内部 LSI 振荡器是否就绪。在 LSION 清零后，等待 3 个 LSI 振荡器的周期后 LSIRDY 被清零。 • 0：LSI 振荡器未就绪 • 1：LSI 振荡器就绪

位 0	LSION：LSI 振荡器使能（LSI oscillator enable） 该位由软件置 1 或清 0。 • 0：LSI 振荡器关闭 • 1：LSI 振荡器开启

6.3.10 AHB 外设复位寄存器（RCC_AHBRSTR）

偏移地址：0x28

复位值：0x0000 0000

访问：无等待周期，支持字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res											IOPDRST	IOPCRST	IOPBRST	IOPARST	Res
											rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									CRCRST	Res					
									rw						

位 31:21	Res：保留 必须保持复位值。
位 20	IOPDRST：将 GPIOD 口复位（Reset I/O port D） 该位由软件置1 或清0。 • 0：无作用 • 1：复位 GPIOD 口
位 19	IOPCRST：将 GPIOC 口复位（Reset I/O port C） 该位由软件置1 或清0。 • 0：无作用 • 1：复位 GPIOC 口
位 18	IOPBRST：将 GPIOB 口复位（Reset I/O port B） 该位由软件置1 或清0。 • 0：无作用 • 1：复位 GPIOB 口
位 17	IOPARST：将 GPIOA 口复位（Reset I/O port A） 该位由软件置1 或清0。 • 0：无作用 • 1：复位 GPIOA 口
位 16:7	Res：保留 必须保持复位值。
位 6	CRCRST：将 CRC 复位（Reset CRC）

	该位由软件置1 或清0。 • 0：无作用 • 1：复位 CRC
位 5:0	Res：保留 必须保持复位值。

6.3.11 时钟配置寄存器 3（RCC_CFGR3）

偏移地址：0x30

复位值：0x0000 0000

访问：无等待周期，支持字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														UART2SW[1:0]	
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res											I2CSW	Res		UART1SW[1:0]	
											rw			rw	

位 31:18	Res：保留 必须保持复位值。
位 17:16	UART2SW[1:0]：UART2 时钟源选择（UART2 clock source selection） 该位由软件置 1 或清 0。 • 00：PCLK 被选为 UART2 的时钟源 • 01：系统时钟（SYSCLK）被选为 UART2 的时钟源。 • 10：保留 • 11：HSI 分频时钟被选为 UART2 的时钟源。
位 15:5	Res：保留 必须保持复位值。
位 4	I2CSW：选择 I2C 时钟源（I2C clock source selection） 该位和 RCC_CFGR4.I2CCLK_SEL 共同决定 I2C 时钟源，由软件置 1 和清0。 当 RCC_CFGR4.I2CCLK_SEL 为 0 时，I2C 的时钟源为： • 0：HSI 时钟被选为 I2C 时钟源 • 1：系统时钟（SYSCLK）被选为 I2C 的时钟源。
位 3:2	Res：保留 必须保持复位值。
位 1:0	UART1SW[1:0]：UART1 时钟源选择（UART1 clock source selection） 由软件置 1 和清0 来选择 UART1 的时钟源。 • 00：PCLK 被选为 UART1 的时钟源

	• 01：系统时钟（SYSCLK）被选为 UART1 的时钟源。 • 10：保留 • 11：HSI 时钟被选为 UART1 的时钟源。

6.3.12 控制寄存器（RCC_CSS）

偏移地址：0xE0

复位值：0x1E00 0000

访问：无等待周期，支持字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSS_THRESHOLD[6:0]							Res								
rw															

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

位 31:25	CSS_THRESHOLD[6:0]：控制 CSS 计数器的阈值（Clock security system threshold value） 当 CSS 功能开启后： RCC 使用 HSI 时钟来采样 GPIO 外部输入时钟分频后的波形。 如果 GPIO 输入频率非常低，则即使 GPIO 输入时钟正常也可能触发 CSS 中断，而通过配置 CSS_THRESHOLD 的值可避免这种情况。产生 CSS 中断的最低 GPIO 输入频率大约为 16M/CSS_THRESHOLD。因此若 CSS_THRESHOLD 等于复位值，当 GPIO 输入小于 1 MHz 时就会产生 NMI 中断。
位 24:0	Res：保留 必须保持复位值。

6.3.13 时钟配置寄存器 4（RCC_CFGR4）

偏移地址：0xE8

复位值：0x1806 1806

访问：无等待周期，支持字、半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	ADCHSIPRE[4:0]					EXTCLK_SEL[1:0]		Res			I2CHSIPRE[4:0]				
	rw					rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2CCLK_SEL	FLITFCLK_PRE[3:0]				FLITFCLK_SEL[1:0]		Res				UARTHSIPRE[4:0]				
rw	rw				rw						rw				

位 31	Res：保留 必须保持复位值。
位 30:26	ADCHSIPRE[4:0]：ADC 时钟相对于 HSI 的分频系数（ADC_HSI prescaler） • 00：HSI/1 • 01：HSI/1.5 • 02：HSI/2

	• 03：HSI/2.5 • … • 1F：HSI/16.5
位 25:24	EXTCLK_SEL[1:0]：外部时钟输入管脚选择（External clock io selection） • 00：外部时钟管脚 1（PA1）选择作为输入源 • 01：外部时钟管脚 2（PD7）选择作为输入源 • 10：外部时钟管脚 3（PB5）选择作为输入源 • 11：外部时钟管脚 4（PC5）选择作为输入源
位 23:21	Res：保留 必须保持复位值。
位 20:16	I2CHSIPRE[4:0]：I2C 时钟相对于 HSI 的分频系数（I2C_HSI prescaler） 分频设置和 ADCHSIPRE 相同。
位 15	I2CCLK_SEL：选择 I2C 的时钟（I2C clock selection） • 0：I2C 时钟由 RCC_CFGR3.I2CSW 决定 • 1：PCLK 作为 I2C 时钟
位 14:11	FLITFCLK_PRE[3:0]：FLITFCLK 分频系数（FLITFCLK prescaler） 分频后的 FLITFCLK 必须满足：2 MHz ≤ FLITFCLK ≤ 6 MHz，其值越大，Flash 擦除所需时间越短。 分频系数等于 FLITFCLK_PRE + 1。 *注意：不能在执行 Flash 编程和擦除的时候，更改 FLITFCLK_PRE 寄存器的值。*
位 10:9	FLITFCLK_SEL[1:0]：FLITFCLK 分频时钟源选择（FLITFCLK clock selection） • 00：选择 HSI 的分频时钟作为 FLITFCLK 分频时钟源。 • 01：选择 SYSCLK 作为 FLITFCLK 分频时钟源。 • 其他：保留 *注意：不能在执行 Flash 编程和擦除的时候更改 FLITFCLK_SEL 寄存器的值。*
位 8:5	Res：保留 必须保持复位值。
位 4:0	UARTHSIPRE[4:0]：UART1, UART2 时钟相对于 HSI 的分频系数（UART_HSI prescaler） 分频设置和 ADCHSIPRE 相同。

7 系统配置控制器（SYSCFG）

HK32F0301MxxxxC 有一组系统配置寄存器，系统配置寄存器的主要目的如下：

- 系统启动区的重映射。
- 管理外部中断与 GPIO 的连接。
- 管理系统的可靠性特性。

7.1 SYSCFG 寄存器

基地址：0x4001 0000

空间大小：0x400

7.1.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）

偏移地址：0x00

复位值：0x0000 0000

这个寄存器用于存储的特别配置（配置地址 0x0000 0000 可访问的存储类型，通过软件选择物理重映射），控制特殊 IO 特性。

复位后，这些位的值为实际启动模式配置的值。

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

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

位 31	LOCKUP_LOCK：Cortex-M0 LOCKUP 使能位（LOCKUP bit enable） 该位通过软件设置，系统复位。它能用于使能和锁定 Cortex-M0 LOCKUP（Hardfault）到 TIM1 Break 输入之间的连接。 • 0：Cortex-M0 LOCKUP 输出到 TIM1 刹车输入断开。 • 1：Cortex-M0 LOCKUP 输出到 TIM1 刹车输入连接。
位 30:2	Res：保留 必须保持为复位值。
位 1:0	MEM_MODE[1:0]：存储映射选择位（Memory mapping selection） 由软件设置和清除这些位。它控制存储器内部映射到地址 0x0000 0000。 • x0：主 Flash 存储器映射到 0x0000 0000 • 01：无效 • 11：SRAM 映射到 0x0000 0000

7.1.2 SYSCFG 外部中断配置寄存器 1（SYSCFG_EXTICR1）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]			
rw				rw				rw				rw			

位 31:16	Res：保留 必须保持复位值。
位 4y+3:4y （y=3..0）	EXTIy[3:0]：选择 EXTIy 的外部中断源（EXTI y configuration bits） 该位域由软件设置。 • x000：PA[y]引脚 • x001：PB[y]引脚 • x010：PC[y]引脚 • x011：PD[y]引脚 • 其他值：保留

7.1.3 SYSCFG 外部中断配置寄存器 2（SYSCFG_EXTICR2）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI7[3:0]				EXTI6[3:0]				EXTI5[3:0]				EXTI4[3:0]			
rw				rw				rw				rw			

位 31:16	Res：保留 必须保持复位值。
位 4(y-4)+3:4(y-4) (y=7..4)	EXTIy[3:0]：选择 EXTIy 的外部中断源（EXTI y configuration bits） 该位域由软件设置。 • x000：PA[y]引脚 • x001：PB[y]引脚 • x010：PC[y]引脚 • x011：PD[y]引脚 • 其他值：保留

8 通用 I/O（GPIO）

本章介绍芯片的 GPIO 功能和寄存器描述。

8.1 GPIO 的主要特性

- 输出状态：推挽或开漏（均带上拉/下拉功能）
- 输入状态：浮空、上拉/下拉
- 模拟功能
- 复用功能选择
- 从输出数据寄存器（GPIOx_ODR）或外设（复用功能输出）输出数据。
- 可以为每个 I/O 口选择不同的速度。
- 将数据输入到输入寄存器（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。
- 快速翻转，每次翻转最快只需要两个时钟周期。
- 引脚复用灵活，允许将 I/O 引脚用作 GPIO 或者多种外设功能中的一种。
- 所有 I/O 都可以选择打开或关闭施密特特性。

8.2 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以下任意模式：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟输入
- 具有上拉或下拉的开漏输出
- 具有上拉或下拉的推挽输出
- 具有上拉或下拉的复用功能推挽
- 具有上拉或下拉的复用功能开漏
- 所有 I/O 可以打开或关闭施密特特性

每个 I/O 端口均可自由编程，但 I/O 端口寄存器必须按字（32 位）、半字（16 位）、或者是字节进行访问。GPIOx_BSRR 寄存器和 GPIOx_BRR 寄存器旨在实现对 GPIOx_ODR 寄存器的原子读写操作，以确保在读取和修改访问之间发生中断请求也不会有问题。

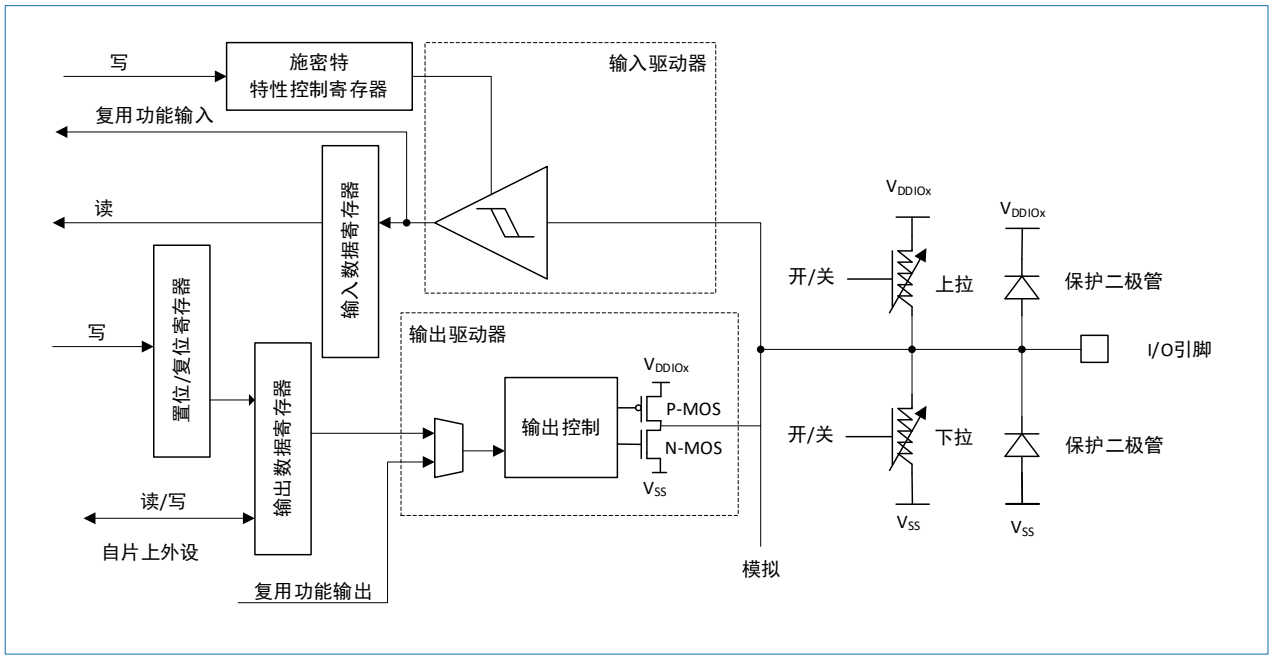

图 8-1 I/O 端口位的基本结构

表 8-1 端口位配置表

MODER（i）[1:0]	OT（i）	OSPEEDR（i）[1:0]	PUPDR（i）[1:0]		I/O 配置	
01	0	OSPEEDER[1:0]	0	0	通用输出	推挽
	0		0	1	通用输出	推挽+上拉
	0		1	0	通用输出	推挽+下拉
	0		1	1	保留	
	1		0	0	通用输出	开漏
	1		0	1	通用输出	开漏+上拉
	1		1	0	通用输出	开漏+下拉
	1		1	1	保留（通用输出开漏）	
10	0	OSPEEDER[1:0]	0	0	复用功能	推挽
	0		0	1	复用功能	推挽+上拉
	0		1	0	复用功能	推挽+下拉
	0		1	1	保留	
	1		0	0	复用功能	开漏
	1		0	1	复用功能	开漏+上拉
	1		1	0	复用功能	开漏+下拉
	1		1	1	保留	
00	X（1）	X	0	0	输入	浮空

MODER（i）[1:0]	OT（i）	OSPEEDR（i）[1:0]	PUPDR（i）[1:0]		I/O 配置	
	X	X	0	1	输入	上拉
	X	X	1	0	输入	下拉
	X	X	1	1	保留（输入浮空）	
11	X	X	0	0	输入/输出	模拟
	X	X	0	1	保留	
	X	X	1	0		
	X	X	1	1		

（1）“X”表示无影响，不起作用。

8.2.1 通用 I/O（GPIO）

在复位期间及复位刚完成时，复用功能尚未激活，大多数 I/O 端口都会默认为输入模式，但调试引脚例外：

- PB5：SWCLK 处于下拉状态
- PD5：SWDIO 处于上拉状态

当引脚配置为输出后，写入到输出寄存器（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）每个 AHB 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。

注意：当 PB5/PD5 用作普通 GPIO 或者模拟模式即 GPIOx_MODER.MODERy!=10 时，需要配置 RCC_APBENR1.IOMUXEN 为 1。

8.2.2 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）连接到同一个 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 8 路复用功能输入（AF0 到 AF7），可通过 GPIOx_AFR（引脚 0 至 7）。

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 IO。

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 复用架构外，各外设还可以将复用功能映射到不同 I/O 引脚，以优化小型封装中可用外设的数量。

要在指定配置下使用 I/O，用户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。
- GPIO：在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟。
- 外设复用功能：
  A. 在 GPIOx_AFR 寄存器中，将 I/O 连接到所需的 AFx。
  B. 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器，分别选择类型、上拉/下拉以及输出速度。

C. 在 GPIOx_MODER 寄存器中将所需的 I/O 配置为复用功能。

- 其他功能：

对于 ADC，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC 寄存器中配置所需功能。

对于 WKUP 和振荡器的其他功能，在相关的 PWR 和 RCC 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。

8.2.3 I/O 端口控制寄存器

每个通用 I/O 端口包括：

- 7 个 32 位的控制寄存器，可配置多达 18 个 IO。
  - 端口模式寄存器 GPIOx_MODER，用于选择 I/O 模式（输入、输出、AF 或模拟）。
  - 端口输出类型寄存器 GPIOx_OTYPER，用于选择输出类型（推挽或开漏）。
  - 端口输出速度寄存器 GPIOx_OSPEEDR，用于选择速度。
  - 端口上下拉寄存器 GPIOx_PUPDR，用于选择上拉/下拉。
  - 端口置位/复位寄存器 GPIOx_BSRR，用于端口置位/复位。
  - 端口复位寄存器 GPIOx_BRR，用于端口复位。
  - 端口施密特寄存器 GPIOx_IOSR，用于配置 I/O 的施密特特性。
    器件复位后 I/O 的施密特特性默认开启（包括 SWDIO 和 SWCLK 两个引脚）。
- 1 个 32 位锁定寄存器 GPIOx_LCKR
- 1 个 32 位复用功能选择寄存器 GPIOx_AFR
- 2 个 32 位的数据寄存器：
  - 端口输入数据寄存器 GPIOx_IDR，通过 I/O 输入的数据存储到该寄存器。
  - 端口输出数据寄存器 GPIOx_ODR，用于存储待输出数据，支持读/写访问。

8.2.4 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。GPIOx_ODR 中的每个数据位对应 GPIOx_BSRR 中的两个控制位：BSy 和 BRy（y=7..0）。当将 BSy 置位时，会置位对应的 ODRy；当将 BRy 置位时会复位对应的 ODRy。向 GPIOx_BSRR 中任何位写 0 都不会对 GPIOx_ODR 中的对应位产生任何的影响。如果在 GPIOx_BSRR 中同时尝试对 BSy 和 BRy 写'1'时，对 BSy 写'1'的操作优先，对 BRy 写'1'的操作被忽略掉。

使用 GPIOx_BSRR 寄存器更改 GPIOx_ODR 中各位的值是一个“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 寄存器只是提供了一种对 GPIOx_ODR 寄存器执行原子按位处理的方法。

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

8.2.5 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_PUPDR、GPIOx_AFR 和 GPIOx_IOSR。

对 GPIOx_LCKR 寄存器执行写操作必须写入特定的写序列。当正确的 LOCK 序列写到此寄存器的第 16 位后，会使用 LCK[7:0]的值来锁定对应 I/O 的配置（在写序列期间，LCK[7:0]的值必须保持不变）。将 LOCK 序列写到某个端口位后，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更

改。每个 GPIOx_LCKR 位都对应决定着 GPIO 控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFR 和 GPIOx_IOSR）中的对应位。

8.2.6 I/O 复用功能输入输出

每个通用 I/O 端口包括 1 个 32 位复用功能选择寄存器 GPIOx_AFR。这个寄存器用于从每个 I/O 可用的复用功能输入/输出中进行选择。根据应用程序的要求，用户可将某个复用功能连接到指定的 I/O 引脚上。由于 AF 选择信号由复位功能输入和复用功能输出共用，所以只需要为指定 I/O 的复用功能输入/输出选择一个通道即可。

8.2.7 外部中断线/唤醒线

所有 I/O 端口都具有外部中断功能。如果使用外部中断线，必须将端口配置为输入模式。

8.2.8 输入配置

对 I/O 端口进行编程作为输入时：

- 输出驱动器被禁用。
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉或和下拉电阻。
- 每个 AHB 时钟周期，输入数据寄存器对 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。

8.2.9 输出配置

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，输出寄存器中的'1'可激活 P-MOS。
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻。
- 输入数据寄存器每个 AHB 时钟周期对 I/O 引脚上的数据采样一次。
- 在开漏模式时，对输入数据寄存器的读访问可获取 I/O 的状态。
- 在推挽模式时，对输出数据寄存器的读访问可获取最后的写入值。

8.2.10 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（发送使能和数据）。
- 根据 GPIOx_IOSR 寄存器的配置使能或禁用施密特触发器。
- 根据 GPIOx_PUPDR 寄存器的配置决定是否打开上拉电阻和下拉电阻。
- 输入数据寄存器每个 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。

8.2.11 模拟配置

对 I/O 端口进行编程作为模拟配置时：

- 输出缓冲器被禁用。

- 施密特触发器被强制停用，I/O 引脚每个模拟输入的功耗变为零。施密特触发器的输出被强制为恒定值’0’。
- 弱上拉和下拉电阻被硬件强制关闭。
- 对输入数据寄存器的读访问值为’0’。

8.2.12 施密特功能配置

当 I/O 工作在模拟模式下时，施密特触发器被强制关闭。除此之外的其他模式下，I/O 的施密特触发器状态由 GPIOx_IOSR 寄存器中的位决定。在数字信号采样时，建议开启施密特触发器，这样可以增强数据采样的搞干扰能力。

8.3 GPIO 寄存器

基地址：（GPIOA; GPIOB, GPIOC, GPIOD）=（0x4800 0000; 0x4800 0400, 0x4800 0800, 0x4800 0C00）

空间大小：（GPIOA; GPIOB, GPIOC, GPIOD）=（0x400; 0x400, 0x400, 0x400）

8.3.1 GPIO 端口模式寄存器（GPIOx_MODER）（x=A..D）

偏移地址：0x00

复位值：0x0000 FBFF（端口 B/端口 D）/ 0x0000 FFFF（其他端口）

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

位 31:16	Res：保留 必须保持复位值。
位 2y+1:2y（y=7..0）	MODERy[1:0]：端口 x 的 y 引脚工作模式配置位（Port x pin y mode configuration bits） 该位域可由软件配置 I/O 口的工作模式。 • 00：输入模式 • 01：通用输出模式 • 10：复用功能模式 • 11：模拟模式（复位状态）

8.3.2 GPIO 端口输出类型寄存器（GPIOx_OTYPER）（x=A..D）

偏移地址：0x04

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8	Res：保留

	必须保持复位值。
位 y （y=7..0）	OTy：端口 x 的 y 引脚输出类型配置位（Port x pin y output type configuration bits） 该位由软件配置 I/O 口的输出类型。 • 0：推挽输出（复位的默认状态） • 1：开漏输出

8.3.3 GPIO 口输出速度寄存器（GPIOx_OSPEEDR）（x=A..D）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]		OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

位 31:16	Res：保留 必须保持复位值。
位 2y+1:2y （y=7..0）	OSPEEDRy[1:0]：端口 x 的 y 引脚的输出速度配置位（Port x pin y output speed configuration bits） 该位域由软件配置 I/O 口的速度。 • 00：Level1 • 01：Level2 • 10：保留 • 11：Level3 Level1、Level2、Level3 对应的具体输出电流值，参见数据手册中的说明。

8.3.4 GPIO 口上拉/下拉寄存器（GPIOx_PUPDR）（x=A..D）

偏移地址：0x0C

复位值：0x0000 0800（端口 B）/ 0x0000 0400（端口 D）/0x0000 FFFF（其他端口）

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]		PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

位 31:16	Res：保留 必须保持复位值。
位 2y+1:2y （y=7..0）	PUPDRy[1:0]：端口 x 引脚 y 配置为上拉或下拉（Port x pin y pull-up/down configuration bits） 该位域由软件配置 I/O 口为上拉或下拉。

	• 00：无上拉和下拉 • 01：上拉 • 10：下拉 • 11：保留

8.3.5 GPIO 端口输入数据寄存器（GPIOx_IDR）（x=A..D）

偏移地址：0x10

复位值：0x0000 XXXX

说明：X 表示不定值。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
								r	r	r	r	r	r	r	r

位 31:8	Res：保留 必须保持复位值。
位 y （y=7..0）	IDy：端口 x 引脚 y 的输入数据（Port x pin y input data） 该位包含相应 I/O 口的输入值，只能读。

8.3.6 GPIO 端口输出数据寄存器（GPIOx_ODR）（x=A..D）

偏移地址：0x14

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8	Res：保留 必须保持复位值。
位 y （y=7..0）	ODy：端口 x 的引脚 y 输出值（Port x pin y output data） 该位用于配置端口的输出状态。 • 0：端口 x 的引脚 y 输出低电平。 • 1：端口 x 的引脚 y 输出高电平。

8.3.7 GPIO 端口置位/复位寄存器（GPIOx_BSRR）（x=A..D）

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0

								w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
								w	w	w	w	w	w	w	w

位 31:24	Res：保留 必须保持复位值。
位 16+y （y=7..0）	BRy：端口 x 的引脚 y 复位控制位（Port x reset bit y） 该位只能写。 读该位时返回值为0。若 BSx 和 BRx 同时设置，BSx 有优先权。 • 0：对端口 x 引脚 y 的 ODy 位无影响。 • 1：复位端口 x 引脚 y 的 ODy 位。
15:8	Res：保留 必须保持复位值。
位 y （y=7..0）	BSy：端口 x 的引脚 y 设置控制位（Port x set bit y） 该位只能写。 读该位时返回值为0。 • 0：对端口 x 引脚 y 的 ODy 位无影响。 • 1：置位端口 x 引脚 y 的 ODy 位。

8.3.8 GPIO 端口配置锁定寄存器（GPIOx_LCKR）（x=A..D）

偏移地址：0x1C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:17	Res：保留 必须保持复位值。
位 16	LCKK：锁定键（Lock key） 该位可随时读取，仅能由锁定键写序列来改写。 • 0：端口配置锁定键不激活 • 1：端口配置锁定键激活。GPIOx_LCKR 寄存器保持锁定，直到 MCU 复位产生才解除锁定。 锁定键写序列： 1. 写 LCKR[16] = ‘1’ + LCKR[15:0] 2. 写 LCKR[16] = ‘0’ + LCKR[15:0] 3. 写 LCKR[16] = ‘1’ + LCKR[15:0]

	4. 读 LCKR 5. 可选操作：读LCKR[16] = ‘1’（确认锁定是否激活） *注意：在锁定键写序列时，不能更改 LCK[15:0]的值。锁定序列中的任何错误操作都将中止锁定操作。在任一端口位上的第一个锁定序列后，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。*
位 15:8	Res：保留 必须保持复位值。
位 y （y=7..0）	LCKy：端口x 引脚 y 锁定位（Port x pin y lock bit y） 该位可读/写，但仅LCKK 为 0 时写。 • 0：端口配置未锁定 • 1：端口配置锁定

8.3.9 GPIO 复用功能寄存器（GPIOx_AFR）（x=A..D）

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[3:0]			
rw				rw				rw				rw			

位 4y+3:4y （y= 7..0）	AFRy[3:0]：端口 x 引脚 y 的复用功能选择（Alternate function selection bits for port x pin y） 可由软件来配置 I/O 口的复用功能。 AFRy 的选择： • 0000：AF0 • 0001：AF1 • 0010：AF2 • 0011：AF3 • 0100：AF4 • 0101：AF5 • 0110：AF6 • 0111：AF7 • 其他值：保留

8.3.10 GPIO 端口位复位寄存器（GPIOx_BRR）（x=A..D）

偏移地址：0x28

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
								w	w	w	w	w	w	w	w

位 31:8	Res：保留 必须保持复位值。
位 y （y=7..0）	BRy：端口 x 的引脚 y 的复位控制位（Port x pin y reset bit y） 该位只能写，读该位的返回值为 0。 • 0：对应端口 x 引脚 y 的 ODy 位无影响 • 1：复位端口 x 引脚 y 的 ODy 位

8.3.11 GPIO 端口输入输出施密特寄存器（GPIOx_IOSR）（x=A..D）

偏移地址：0x30

复位值：0x0000 FFFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								SEN7	SEN6	SEN5	SEN4	SEN3	SEN2	SEN1	SEN0
								rw	rw	rw	rw	rw	rw	rw	rw

位 31:8	Res：保留 必须保持复位值。
位 y （y=7..0）	SENy：端口 x 的 y 引脚的施密特特性开关（Port x pin y Schmitt switch） • 0：使能 IO 施密特功能 • 1：禁能 IO 施密特功能

9 引脚选择功能（IOMUX）

9.1 功能介绍

器件的引脚支持通用功能输入输出（GPIO）或复用功能输入输出（IOMUX）。

通过软件配置 IOMUX 寄存器，以设置 GPIO 的引脚功能。

每个复用的引脚和通用引脚一样，都支持 EXTI 中断功能。

9.2 功能描述

当引脚通过 IOMUX 被复用为其他引脚时，该引脚具有与通用 IO 一样的功能。

需要注意的是，一个引脚可能支持多个外设引脚，需要通过使能 IOMUX 和对应外设的 RCC 时钟，然后配置 IOMUX 相关寄存器才可激活；否则，复用不生效。

当系统复位之后，IOMUX 失效，此时引脚为通用 GPIO 功能，直到再次配置 IOMUX 功能，引脚才会被再次复用为其他引脚。

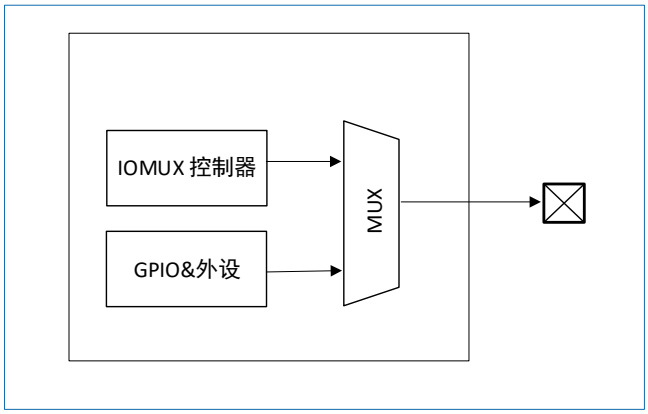

图 9-1 IOMUX 概述框图

9.3 举例说明

HK32F0301MxxxxC 的小型封装（如 SOP8/TSSOP16）产品，可通过 IOMUX 实现单根引脚对应多个 GPIO 或外设 IO 的映射控制。以 SOP8 封装产品为例进一步说明。

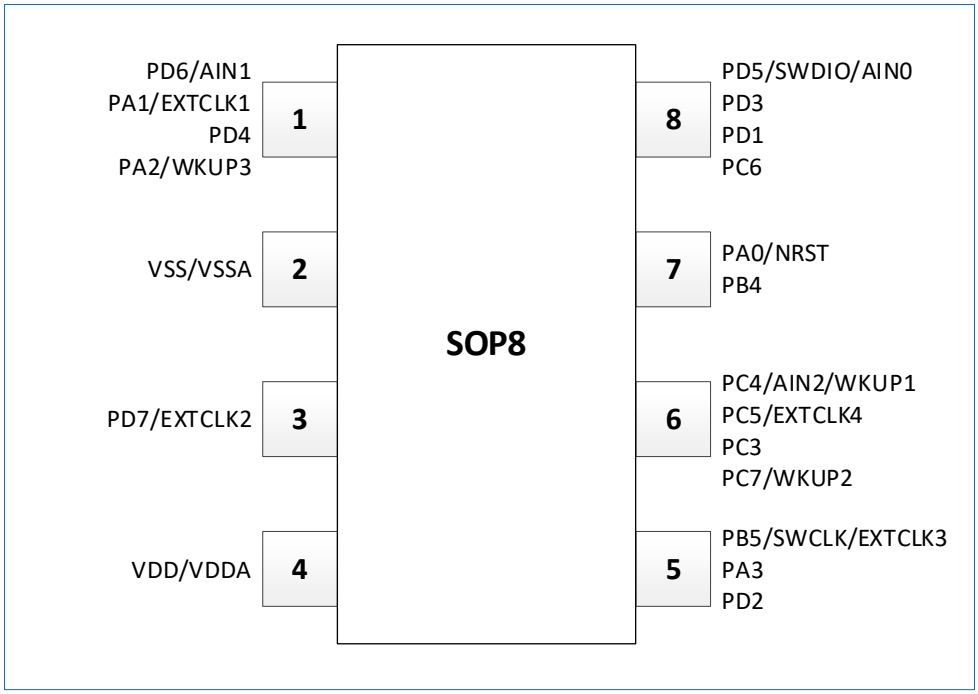

图 9-2 S0P8 封装的管脚排列

如图 9-2 所示，芯片初始复位后，第 8 引脚的功能为“PD5（及 SYSCFG 配置中对应的外设 IO）”；通过配置 IOMUX 寄存器，可以将第 8 脚功能重映射到 PD3（及 SYSCFG 配置中对应的外设 IO）、PD1（及 SYSCFG 配置中对应的外设 IO）或 PC6（及 SYSCFG 配置中对应的外设 IO）。

通过 IOMUX 配置，SOP8/TSSOP16 封装产品可以使用 18 个 GPIO 以及片内所有外设 IO。

9.4 IOMUX 寄存器

基地址：0x4001 7C00

空间大小：0x400

9.4.1 IOMUX 引脚功能选择寄存器（IOMUX_PIN_FUNC_SEL）

偏移地址：0x000

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												PD7_BKIN_SEL	PB5_I2C_SEL	PC4_TIM1_SEL	PC3_TIM1_SEL
												rw	rw	rw	rw

位 31:4	Res：保留 必须保持复位值。
位 3	PD7_BKIN_SEL：PD7 引脚的功能选择（Pin functional selection for PD7） • 0：当 PD7_AF 设置为 AF3 时，PD7 作为 CH3 引脚（系统复位时为此设置）。 • 1：当 PD7_AF 设置为 AF3 时，PD7 作为 BKIN 引脚。
位 2	PB5_I2C_SEL：PB5 引脚的功能选择（Pin functional selection for PB5）

	当 PB5_AF 设置为 AF0 时，PB5_I2C_SEL 为： • 0：PB5 作为 SWCLK 输入引脚。 • 1：PB5 作为 I2C 的 SDA 引脚。
位 1	PC4_TIM1_SEL：PC4 引脚的功能选择（Pin functional selection for PC4） 当 PC4_AF 设置为 AF3 时，PC4_TIM1_SEL 为： • 0：PC4 作为 TIM1 的 CH4 引脚。 • 1：PC4 作为 TIM1 的 CH2N 引脚。
位 0	PC3_TIM1_SEL：PC3 引脚的功能选择（Pin functional selection for PC3） 当 PC3_AF 设置为 AF3 时，PC3_TIM1_SEL 为： • 0：PC3 作为 TIM1 的 CH3 引脚。 • 1：PC3 作为 TIM1 的 CH1N 引脚。

9.4.2 IOMUX 引脚选择寄存器（IOMUX_PKG_PIN_SEL）

偏移地址：0x004

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							PD6_PIN_SEL[1:0]		PD5_PIN_SEL[1:0]		PC4_PIN_SEL[1:0]		PB5_PIN_SEL[1:0]		NRSTPA0_PIN_SEL
							rw		rw		rw		rw		rw

位 31:9	Res：保留 必须保持复位值。
位 8:7	PD6_PIN_SEL[1:0]：PD6 引脚选择（Pin MUX for PD6） • 00：PD6 引脚 • 01：PA1 引脚（16 和 20 引脚封装产品禁止此设置） • 10：PD4 引脚 • 11：PA2 引脚（16 和 20 引脚封装产品禁止此设置）
位 6:5	PD5_PIN_SEL[1:0]：PD5 引脚选择（Pin MUX for PD5） • 00：PD5 引脚 • 01：PD3 引脚（16 和 20 引脚封装产品禁止此设置） • 10：PD1 引脚 • 11：PC6 引脚（16 和 20 引脚封装产品禁止此设置）
位 4:3	PC4_PIN_SEL[1:0]：PC4 引脚选择（Pin MUX for PC4） • 00：PC4 引脚 • 01：PC5 引脚（16 和 20 引脚封装产品禁止此设置）

	• 10：PC3 引脚 • 11：PC7 引脚（16 和 20 引脚封装产品禁止此设置）
位 2:1	PB5_PIN_SEL[1:0]：PB5 引脚选择（Pin MUX for PB5） • 00/11：PB5 引脚 • 01：PA3 引脚（16 和 20 引脚封装产品禁止此设置） • 10：PD2 引脚
位 0	NRSTPA0_PIN_SEL：NRSTPA0 引脚选择（Pin MUX for NRST） • 0：NRST/PA0 引脚 • 1：PB4 引脚（16 和 20 引脚封装产品禁止此设置） *注意：* *NRSTPA0_PIN_SEL 只能通过上电复位。* *只有当 NRST_PIN_KEY[15:0]=0x5AE1 时，才能写入 NRSTPA0_PIN_SEL。*

9.4.3 IOMUX 功能关键字寄存器（IOMUX_NRST_PIN_KEY）

偏移地址：0x008

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	NRST_PIN_KEY[15:0]：复位引脚功能复用关键字（The key of pin NRST change to PA0） NRST_PA0_SEL 的功能从 NRST 改变到 PA0 之前，必须先将该寄存器设置为 0x5AE1。 *注意：* *当 CPU 改变寄存器位 IOMUX_PKG_PIN_SEL.NRSTPA0_PIN_SEL 或 IOMUX_NRST_PA0_SEL.NRST_PA0_SEL 的值后，NRST_PIN_KEY[15:0]会被系统硬件复位。*

9.4.4 IOMUX 引脚功能控制寄存器（IOMUX_NRST_PA0_SEL）

偏移地址：0x00C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															NRST_PA0_SEL
															rw

位 31:1	Res：保留 必须保持复位值。
位 0	NRST_PA0_SEL：选择 NRST/PA0 引脚的功能（Pin Functional Selection for NRST/PA0） • 0：NRST 功能 • 1：PA0 功能 *注意：* *NRST_PA0_SEL 只能上电复位。* *只有在 NRST_PIN_KEY[15:0]=0x5AE1 时，才能写 NRST_PA0_SEL。*

10 中断和事件（NVIC 和 EXTI）

本章主要介绍了 NVIC 和 EXTI 的功能特性。

10.1 嵌套向量中断控制器（NVIC）

10.1.1 NVIC 主要特性

嵌套向量中断控制器（NVIC）和处理器核的接口紧密相连，可以实现低延迟的中断处理和高效地处理晚到的中断。嵌套向量中断控制器管理着包括中断在内的所有异常。

- 23 个可屏蔽中断通道（不包括 16 个 Cortex-M0 的中断线）
- 4 个可编程的中断优先级（2 位中断优先级）
- 低延迟的异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

10.1.2 系统嘀嗒校准值寄存器

对于 HK32F0301MxxxxC 来说，系统嘀嗒（SysTick）校准值设置为 6000，当 SysTick 时钟分别设置为 6 MHz（f_{HCLK}/8 的最大值）时，会产生 1 ms 时间基准。

说明：HK32F0301MxxxxC 的 f_{HCLK} 最大值为 48MHz。

10.1.3 中断和异常向量

表 10-1 NVIC 表

位置	优先级		名称	描述	地址
-	-	-	-	保留	0x0000 0000
-	-3	固定	Reset	复位（Reset）	0x0000 0004
-	-2	固定	NMI	非屏蔽中断（Non-maskable interrupt）	0x0000 0008
-	-1	固定	HardFault	所有类型错误（All class of fault）	0x0000 000C
-	3	可配置	SVCall	通过 SWI 指令进行的系统服务调度（System service call via SWI instruction）	0x0000 002C
-	5	可配置	PendSV	可挂起的系统服务请求（Pendable request for system service）	0x0000 0038
-	6	可配置	SysTick	系统嘀嗒定时器（System tick timer）	0x0000 003C
0	7	可配置	WWDG	窗口看门狗中断（Window Watchdog interrupt）	0x0000 0040
1	8	-	-	保留	0x0000 0044
2	9	可配置	EXTI11	外部线 11 的自动唤醒中断（AWUT WKP interrupt, EXTI 11）	0x0000 0048
3	10	可配置	FLASH	Flash 全局中断（Flash global interrupt）	0x0000 004C

位置	优先级		名称	描述	地址
4	11	可配置	RCC	RCC 全局中断（RCC global interrupt）	0x0000 0050
5	12	可配置	EXTI0	EXTI 线 0 中断（EXTI Line0 interrupt）	0x0000 0054
6	13	可配置	EXTI1	EXTI 线 1 中断（EXTI Line1 interrupt）	0x0000 0058
7	14	可配置	EXTI2	EXTI 线 2 中断（EXTI Line2 interrupt）	0x0000 005C
8	15	可配置	EXTI3	EXTI 线 3 中断（EXTI Line3 interrupt）	0x0000 0060
9	16	可配置	EXTI4	EXTI 线 4 中断（EXTI Line4 interrupt）	0x0000 0064
10	17	可配置	EXTI5	EXTI 线 5 中断（EXTI Line5 interrupt）	0x0000 0068
11	18	可配置	TIM1_BRK	TIM1 刹车中断（TIM1 Break interrupt）	0x0000 006C
12	19	可配置	ADC1	ADC 中断（和 EXTI 线 8 共用）（ADC interrupt, combined with EXTI Line8）	0x0000 0070
13	20	可配置	TIM1_UP_TRG_COM	TIM1 更新/触发中断（TIM1 Update/Trigger Com interrupt）	0x0000 0074
14	21	可配置	TIM1_CC	TIM1 捕获/比较中断（TIM1 Capture/Compare interrupt）	0x0000 0078
15	22	可配置	TIM2	TIM2 全局中断（TIM2 global interrupt）	0x0000 007C
16	23	-	-	保留	0x0000 0080
17	24	可配置	TIM6	TIM6 全局中断（TIM6 global interrupt）	0x0000 0084
18	25	-	-	保留	0x0000 0088
19	26	-	-	保留	0x0000 008C
20	27	-	-	保留	0x0000 0090
21	28	可配置	EXTI6	EXTI 线 6 中断（EXTI Line6 interrupt）	0x0000 0094
22	29	可配置	EXTI7	EXTI 线 7 中断（EXTI Line7 interrupt）	0x0000 0098
23	30	可配置	I2C	I2C 全局中断（和 EXTI 线 10 共用）（I2C global interrupt, combined with EXTI Line 10）	0x0000 009C
24	31	-	-	保留	0x0000 00A0
25	32	可配置	SPI	SPI 全局中断（SPI global interrupt）	0x0000 00A4
26	33	-	-	保留	0x0000 00A8
27	34	可配置	UART1	UART1 全局中断（UART1 global interrupt）	0x0000 00AC
28	35	可配置	UART2	UART2 全局中断（UART2 global interrupt）	0x0000 00B0
29	36	-	-	保留	0x0000 00B4
30	37	-	-	保留	0x0000 00B8
31	38	-	-	保留	0x0000 00BC

10.2 扩展中断和事件控制器（EXTI）

扩展中断及事件控制器（EXTI）负责管理内、外异步中断和事件：向 CPU 输出事件请求，向中断控制器输出中断请求，向电源控制模块输出唤醒请求。

根据中断/事件触发沿是否可配置，可将 EXTI 分为两类：触发沿可配 EXTI（简称可配 EXTI）和触发沿固定 EXTI（简称固定 EXTI）。

可配 EXTI 特性：

- 可选择上升沿/下降沿触发。
- 拥有挂起状态寄存器以记录中断状态。
- 可通过写软件中断事件寄存器 EXTI_SWIER 对应位来模拟生成中断/事件。

固定 EXTI 特性：

- 采用上升沿触发。
- 仅工作在停机模式，用于从停机模式唤醒内核。
- 挂起状态寄存器中无法查询固定 EXTI 中断状态，需由对应 IP 提供。

EXTI 管理多达 11 个中断/事件。各中断或事件线均可独立屏蔽或使能。

10.2.1 主要特性

EXTI 控制器的主要特性如下：

- 支持多达 11 个事件/中断请求线
  - 9 条可配置线
    - 触发沿上升沿或下降沿可选
    - 有专用的中断状态位标记
    - 可通过软件方式触发中断、事件
  - 2 条固定线
- 每根中断/事件线都可单独触发和屏蔽
- 能够检测到比 APB 时钟宽度还小的脉冲

10.2.2 框图

EXTI 结构框图如下所示：

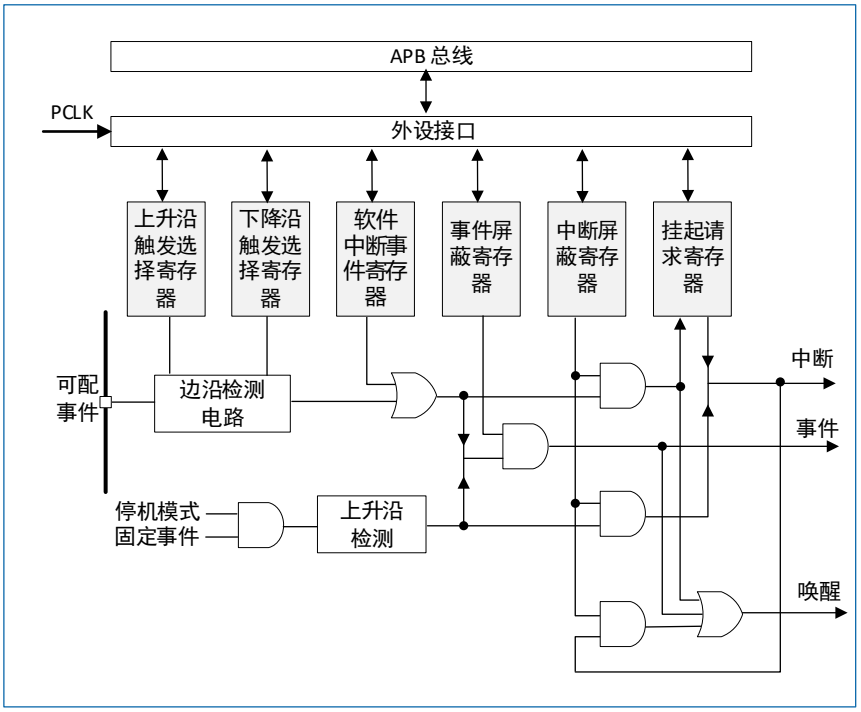

图 10-1 EXTI 框图

10.2.3 EXTI 与周边模块关系

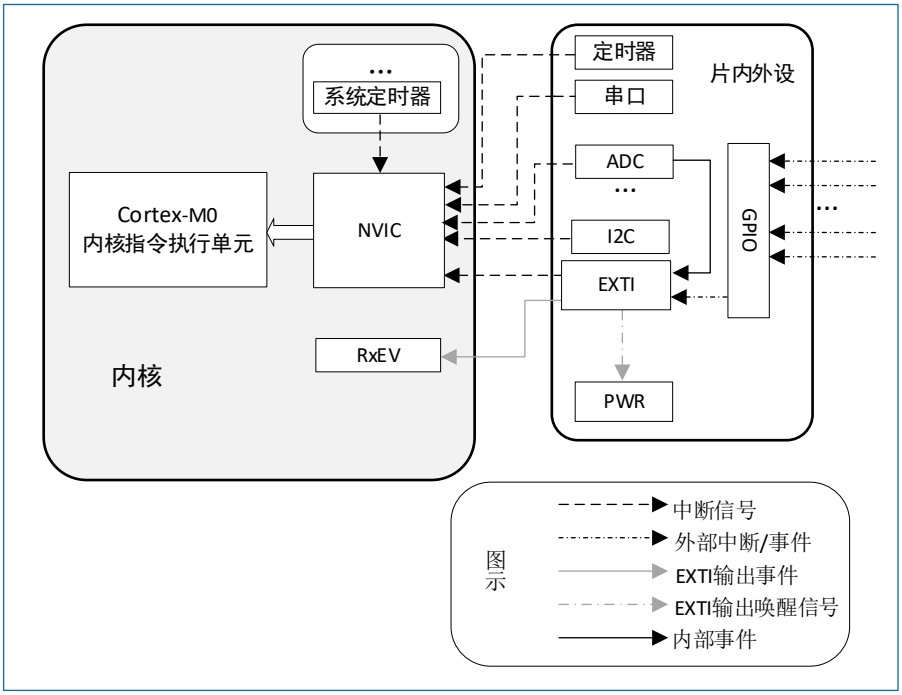

图 10-2 EXTI 与周边模块关系框图

如图 10-2 所示，EXTI 的中断输出与芯片内其他中断源一起汇总于 NVIC；EXTI 输出的事件，输入到 RxEV 模块，用于管理内核唤醒相关操作；EXTI 输出的唤醒信号，输出给 PWR 单元。

10.2.4 唤醒事件管理

芯片可以处理外部或内部事件来唤醒内核（WFE）。唤醒事件可以通过下述方式产生：

- 在外设的控制寄存器中使能一个中断，但不在 NVIC 中使能，同时在 Cortex®-M0 的系统控制寄存器中使能 SEVONPEND 位。当 CPU 从 WFE 恢复后，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 恢复后，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。

使用外部 I/O 端口作为唤醒事件，请参考“10.2.5 功能说明”。

10.2.5 功能说明

可配 EXTI 如果要产生中断，必须先配置并使能中断线。根据所需的边沿检测条件来配置上升沿/下降沿触发寄存器，并且通过向中断屏蔽寄存器的相应位写’1’来使能中断请求。当外部中断线上发生了所设置的边沿时，将产生一个中断请求，对应的挂起位也随之被置’1’。向挂起寄存器的对应位写’1’，将清除该中断请求。

固定 EXTI 的挂起状态寄存器中无法查询中断状态，需在对应的 IP 寄存器中查询。

如果需要产生事件，必须先配置并使能事件线。根据所需的边沿检测条件来配置上升沿/下降沿触发寄存器，并且通过向事件屏蔽寄存器的相应位写’1’来使能事件请求。当事件线上发生了需要的边沿时，将产生一个事件请求脉冲，对应的挂起位不被置’1’。

对于可配 EXTI，通过软件向软件中断/事件寄存器写’1’，可产生中断/事件请求。

注意：固定 EXTI 仅工作在停机模式下。

10.2.5.1 硬件中断选择

配置 EXTI 线作为中断源的步骤如下：

1. 配置所选中断线的屏蔽位（EXTI_IMR）。
2. 配置所选中断线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）。
3. 配置该外部中断控制器（EXTI）对应的 NVIC 中断通道的使能和屏蔽位，使 EXTI 线的中断能被正确响应。

10.2.5.2 硬件事件选择

配置 EXTI 线作为事件源的步骤如下：

1. 配置所选事件线的屏蔽位（EXTI_EMR）。
2. 配置相应事件线的触发选择位（EXTI_RTSR 和 EXTI_FTSR）。

10.2.5.3 软件中断/事件的选择

可配 EXTI 线可以被配置成软件中断/事件线。下面是产生软件中断的步骤：

1. 配置相应中断/事件线的屏蔽位（EXTI_IMR，EXTI_EMR）。
2. 设置软件中断寄存器的请求位（EXTI_SWIER）。

10.2.6 外部中断/事件线映射

芯片 GPIO 口以下图的方式连接到 8 根外部中断/事件线上：

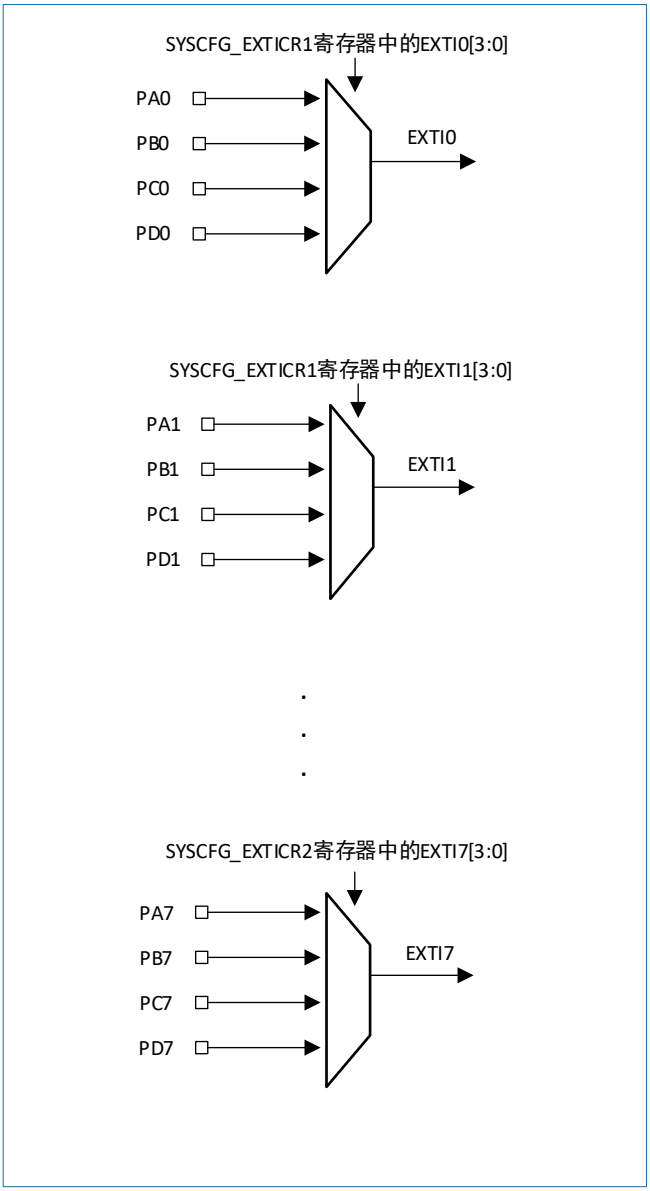

图 10-3 外部中断 GPIO 映射

说明：图10-3 为GPIO 与EXTI 映射示意图。不同型号的芯片可用管脚数目可能存在区别。具体芯片各端口可用IO，请查询对应芯片数据手册。

如图 10-3 所示，通过 SYSCFG_EXTICRx 配置 GPIO 线上的外部中断/事件，为使外部中断/事件正常工作，除使能 GPIO 时钟外，还需配置 RCC_APBENR2.SYSCFGEN 位（参见 6.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR））。

表 10-2 EXTI 中断事件映射关系表

EXTI 序号	信号	EXTI 类型
0-7	GPIO	可配
8	ADC AWD（Analog watchdog detection）	固定
10	I2C Wakeup	固定
11	AWU Wakeup	可配

其中 EXTI8 和 EXTI10 作为固定事件，不带 RTSR、FTSR、SWIER 和 PR 寄存器。这 2 个 EXTI 口仅能在停机模式下采集事件的上升沿以产生 ERQ 和 IRQ 唤醒系统。其相应的中断控制和状态位都存储于产生事

件源的外设模块内。

10.3 EXTI 寄存器

基地址：0x4001 0400

空间大小：0x400

10.3.1 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				IM11	IM10	Res	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
				rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 x （x= 11..10）	IMx：外部/内部线 x 的中断屏蔽位（Interrupt mask on line x） • 0：屏蔽来自线 x 上的中断请求 • 1：允许来自线 x 上的中断请求
位 9	Res：保留 必须保持复位值。
位 x （x= 8..0）	IMx：外部/内部线 x 的中断屏蔽位（Interrupt mask on line x） • 0：屏蔽来自线 x 上的中断请求 • 1：允许来自线 x 上的中断请求

10.3.2 事件屏蔽寄存器（EXTI_EMR）

偏移地址：0x04

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				EM11	EM10	Res	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
				rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 x （x= 11..10）	EMx：外部/内部线 x 的事件屏蔽位（Event mask on line x） • 0：屏蔽来自线 x 上的事件请求 • 1：允许来自线 x 上的事件请求

位 9	Res：保留 必须保持复位值。
位 x （x= 8..0）	EMx：外部/内部线 x 的事件屏蔽位（Event mask on line x） • 0：屏蔽来自线 x 上的事件请求 • 1：允许来自线 x 上的事件请求

10.3.3 上升沿触发选择寄存器（EXTI_RTSR）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				RT11	Res			RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
				rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 11	RT11：线 11 的上升沿触发事件配置位（Rising trigger event configuration bit of line 11） • 0：禁止输入线 11 上的上升沿触发（中断和事件） • 1：启用输入线 11 上的上升沿触发（中断和事件）
位 10:8	Res：保留 必须保持复位值。
位 x （x= 7..0）	RTx：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 x 上的上升沿触发（中断和事件） • 1：启用输入线 x 上的上升沿触发（中断和事件）

10.3.4 下降沿触发选择寄存器（EXTI_FTSR）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				FT11	Res			FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
				rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 11	FT11：线11 的下降沿触发事件配置位（Falling trigger event configuration bit of line 11） • 0：禁止输入线11 上的下降沿触发（中断和事件）

	• 1：启用输入线11 上的下降沿触发（中断和事件）
位 10:8	Res：保留 必须保持复位值。
位 x （x= 7..0）	FTx：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 x 上的下降沿触发（中断和事件） • 1：启用输入线 x 上的下降沿触发（中断和事件）

10.3.5 软件中断事件寄存器（EXTI_SWIER）

偏移地址：0x10

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				SWI11	Res			SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
				rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 11	SWI11：线 11 的软件中断（Software interrupt on line 11） 当该位为 0 时，将其写 1 会设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。 通过清除 EXTI_PR 的对应位（写入 1），可以将该位清零。
位 10:8	Res：保留 必须保持复位值。
位 x （x= 7..0）	SWIx：线 x 的软件中断（Software interrupt on line x） 当该位为 0 时，将其写 1 会设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。 通过清除 EXTI_PR 的对应位（写入 1），可以将该位清零。

10.3.6 请求挂起寄存器（EXTI_PR）

偏移地址：0x14

复位值：0xXXXX XXXX

说明：X 表示不定值。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				PIF11	Res			PIF7	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
				rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31:12	Res：保留 必须保持复位值。
位 11	PIF11：线 11 挂起位（Pending interrupt flag on line 11） • 0：没有发生触发请求 • 1：发生了选择事件的触发请求。当外部中断线上发生了所选择的边沿事件，该位被置 1。 向该位写入 1 或改变边沿检测的极性进行清除。
位 10:8	Res：保留 必须保持复位值。
位 x （x= 7..0）	PIFx：线 x 挂起位（Pending interrupt flag on line x） • 0：没有发生触发请求 • 1：发生了选择事件的触发请求。当外部中断线上发生了所选择的边沿事件，该位被置 1。 向该位写入 1 或改变边沿检测的极性进行清除。

11 模拟数字转换器（ADC）

芯片内置的 12 位 ADC 是逐次趋近型模数转换器。它具有多达 9 个复用通道，可测量来自 7 个外部和 2 个内部的信号。其中 AIN0 ~AIN6 为外部通道接 IO，AIN7 为内部通道，连接内部 PMU（电源管理单元，仅用于芯片内部电源电压校准），AIN8 为内部通道，连接内部参考电压。支持差分输入模式，AIN0 和 AIN1，AIN2 和 AIN3，AIN4 和 AIN5 组成三组差分输入（当 ADC 配置为差分输入模式时，AIN7 和 AIN8 为采样内部 PMU、参考电压的 ADC 通道，通道 6、7、8 不可用）。各个通道的 A/D 转换支持单次、连续、扫描或不连续转换模式。ADC 的结果可以左对齐或右对齐地存储在一个 16 位数据寄存器中。

ADC 具有模拟看门狗特性，支持应用检测输入电压是否超过了用户自定义的上限或下限阈值。

11.1 ADC 主要特性

- 高性能
  - 支持 12 位分辨率。
  - ADC 转换时间：12 位分辨率对应的转换时间为 1 μs（1 MHz）。
  - 可编程采样时间
  - 内置数据一致性，可确保数据对齐。
- 低功耗
  - 应用可降低 PCLK 频率从而以低功耗运行，同时仍可保持最优的 ADC 性能。例如，无论 PCLK 的频率如何，都保持 1.0 μs 的转换时间。
  - 等待模式：防止 ADC 在低频 PCLK 应用中溢出。
  - 自动关闭模式：ADC 会自动断电，但正在进行转换时除外。这可大幅降低 ADC 的功耗。
- 模拟输入通道
  - 7 路外部模拟输入
  - 1 条用于内部电源管理单元（PMU）的通道
  - 1 条用于内部参考电压（VREFINT）的通道
- 可通过以下方式启动 A/D 转换：
  - 通过软件
  - 通过极性可配置的硬件触发器（来自 TIM1、TIM2、TIM6 的内部定时器事件 GPIO 输入事件）
- 转换模式

可转换单条通道，也可扫描一系列通道。

  - 单次模式：会在每次触发时对选定的输入通道执行一次转换
  - 连续模式：可对选定的输入通道进行连续转换
  - 不连续转换模式
- 在发生 ADC 就绪、转换结束、转换序列结束、模拟看门狗检测或上溢等事件时产生中断。
- 带模拟看门狗功能
- ADC 电源要求：2.4V~5.5 V
- ADC 电压输入范围：$V_{SSA} \le V_{IN} \le V_{DDA}$

11.2 ADC 功能描述

ADC 功能框图如下：

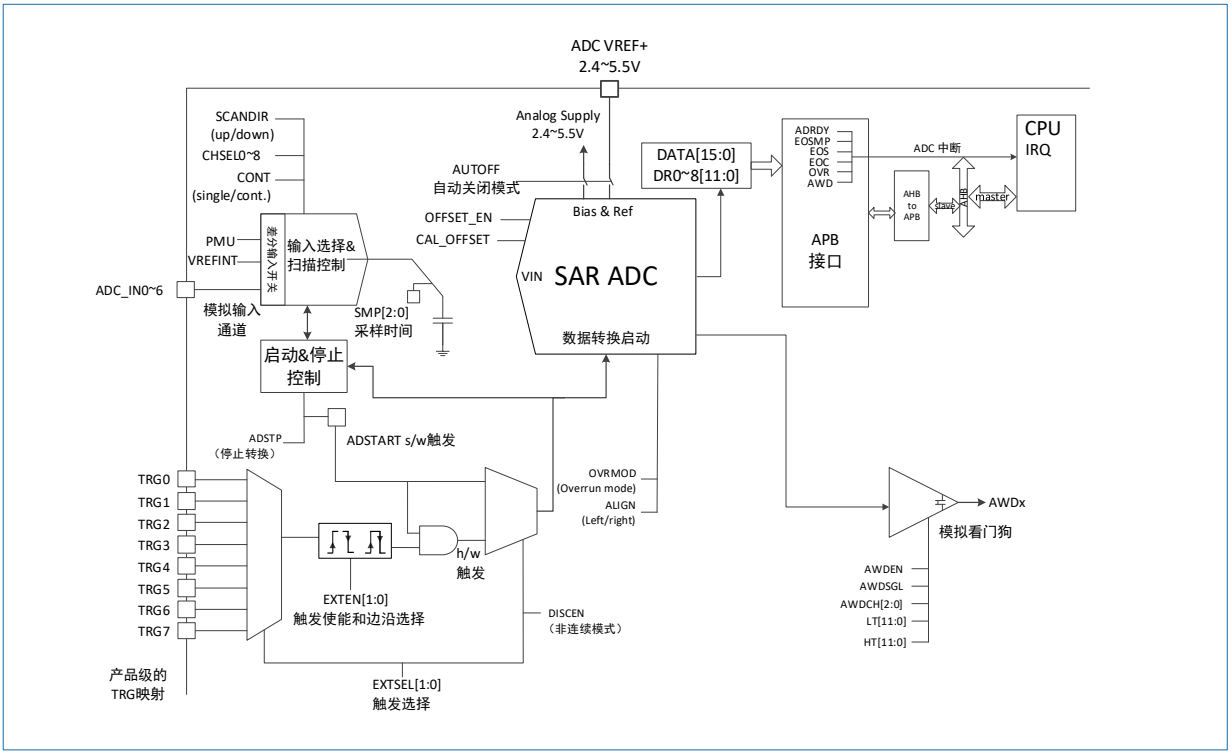

图 11-1 ADC 功能框图

11.2.1 ADC 引脚和内部信号

表 11-1 ADC 引脚

名称	信号类型	备注
V_{DDA}	模拟电源输入	ADC 的模拟电源和正参考电压
V_{SSA}	模拟电源地输入	模拟电源接地引脚。电压必须等于 V_{SS}。
ADC_IN0~6	模拟输入信号	7 个模拟输入通道

表 11-2 ADC 内部信号

内部信号名称	信号类型	说明
TRGx	输入	ADC 转换触发信号
PMU	输入	内部 PMU 输出电压
VREFINT	输入	内部参考输出电压

11.2.2 校准误差控制（ADC_OFFSET, CAL_OFFSET）

偏移误差校准使能：

将 OFFSET_EN 置 1，将使能 ADC Offset 误差校准；置 0，则不会进行 Offset 误差校准。

ADC 偏移误差校准值：

将 OFFSET_EN 置 1 后，通过配置 CAL_OFFSET，来配置 ADC 偏移误差校准值。

这些值表示 Offset 的大小，最高位 CAL_OFFSET[5]是符号位，0 表示 OFFSET data 为正，1 表示 OFFSET data 为负，CAL_OFFSET[4:0]是数据位；OFFSET data 如果是正值，则 CAL_OFFSET[4:0]是正常数据，OFFSET data 如果是负值，则 CAL_OFFSET[4:0]是补码数据。

如果芯片采样外部信号存在偏移误差，可以用此功能来修正这个偏移误差。

例如：

- 偏移误差是 2，配置 offset =2；

  OFFSET_EN=1；

  CAL_OFFSET[5]=0，CAL_OFFSET[4:0]=0x02。

- 偏移误差是-2，配置 offset =-2；

  OFFSET_EN=1；

  CAL_OFFSET[5]=1，CAL_OFFSET[4:0]=0x1E。

11.2.3 ADC 开关控制（ADEN, ADDIS, ADRDY）

MCU 上电时，ADC 被禁用并进入掉电模式（ADEN=0）。

如图 11-2 所示，ADC 在开始精确转换之前需要一段稳定时间 t_{STAB}。

以下两个控制位可用于使能或禁止 ADC：

- 将 ADEN 置 1 可使能 ADC。ADC 准备就绪后，ADRDY 标志会立即置 1。
- 将 ADDIS 置 1 可禁用 ADC 并使 ADC 进入掉电模式。随后，ADC 被完全禁止后，ADEN 位和 ADDIS 位会自动由硬件清零。

可通过将 ADSTART 置 1（请参见“11.3 外部触发转换和触发极性（EXTSEL, EXTEN）”）开始进行转换；如果触发器已使能，也可在发生外部触发事件时开始进行转换。

使能 ADC 的流程如下：

1. 对 ADC_ISR 寄存器中的 ADRDY 位写 1，将此位清零。
2. 将 ADC_CR 寄存器中的 ADEN 位置 1。
3. 等待直至 ADC_ISR 寄存器中的 ADRDY=1（ADRDY 会在 ADC 启动时间后置 1）。如果已通过将 ADC_IER 寄存器中的 ADRDYIE 位置 1 来使能中断，可通过中断进行处理。

禁用 ADC 的流程如下：

1. 检查 ADC_CR 寄存器中的 ADSTART 为 0，以确保当前未执行任何转换。如有需要，可向 ADC_CR 寄存器中的 ADSTP 位写入 1 并等待此位读取值为 0，以此停止正在进行的转换。
2. 将 ADC_CR 寄存器中的 ADDIS 位置 1。
3. 如果应用要求，可等待 ADC_CR 寄存器中的 ADEN=0，这表明 ADC 已完全禁止（ADEN=0 后，ADDIS 会自动复位）。
4. 将 ADC_ISR 寄存器中的 ADRDY 位编程为 1，将此位清零（可选步骤）。

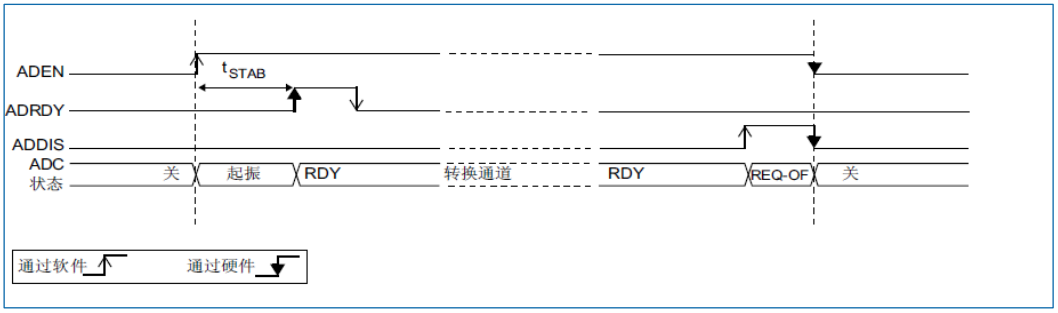

图 11-2 使能/禁用 ADC

说明：在自动关闭模式下（AUTOFF=1），上电/掉电阶段是由硬件自动执行的，不会将ADRDY 置1。

11.2.4 ADC 时钟（CKMODE）

ADC 采用双时钟域架构，因此，ADC 可由独立于 APB 时钟（PCLK）的时钟提供时钟（ADC 异步时钟）。要选择该时钟，必须将 ADC_CFGR2 寄存器的 CKMODE[1:0]位复位。

选择该时钟的优势在于：无论选择哪种 APB 时钟，都可以达到最大 ADC 时钟频率。

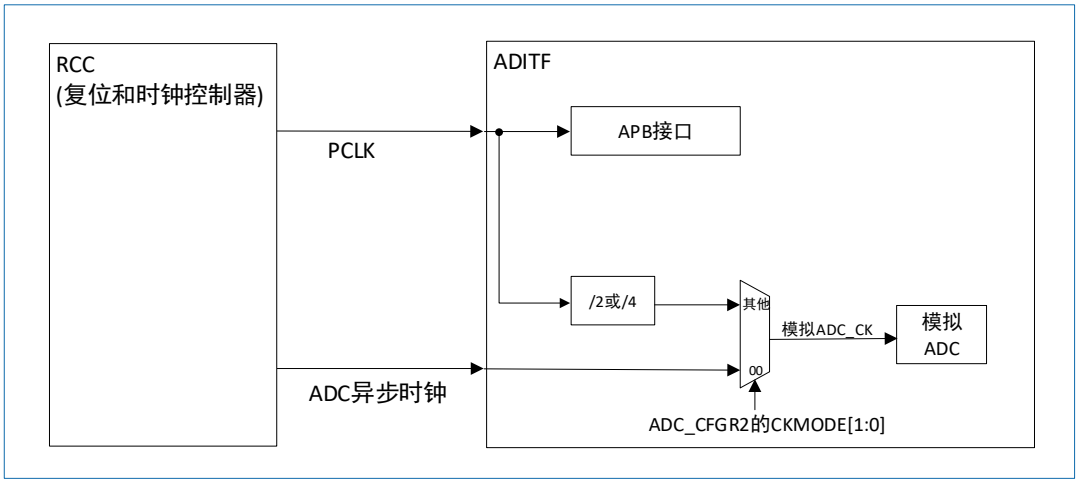

图 11-3 ADC 时钟图

说明：了解 PCLK 和 ADC 异步时钟的使能方式参见“6 复位和时钟控制（RCC）”。

ADC 时钟也可由 ADC 总线接口的 APB 时钟除以一个可编程因子（2 或 4，由 CKMODE[1:0]位而定）来提供。要选择该时钟，ADC_CFGR2 寄存器的 CKMODE[1:0]位需要设置为“01”或“10”。

选择该时钟的优势在于：不用重新同步时钟域。如果 ADC 由定时器触发，并且应用要求 ADC 精确触发（不存在任何不确定性），可使用此选项。否则，重新同步两个时钟域会为触发时刻带来不确定性。

表 11-3 触发与转换开始之间的延迟

ADC 时钟源	CKMODE[1:0]	触发事件与转换开始之间的延迟
HSI 时钟	00	延迟是不确定的（存在抖动）
PCLK 2 分频	01	延迟是确定的（无抖动），等于 4.25 个 ADC 时钟周期
PCLK 4 分频	10	延迟是确定的（无抖动），等于 4.125 个 ADC 时钟周期

11.2.5 配置 ADC

如果 ADC 已禁用（ADEN=0），必须通过软件写 ADC_CR 寄存器中的 ADEN 位来配置 ADC。

仅当 ADC 已使能且没有待处理的禁用 ADC 的请求时（ADEN=1 且 ADDIS=0），软件才能写 ADC_CR 寄存器中的 ADSTART 位和 ADDIS 位。

对于 ADC_IER、ADC_CFGRx、ADC_SMPR、ADC_TR、ADC_CHSELR 和 ADC_CCR 寄存器中的其他控制位，只有在 ADC 已使能（ADEN=1）且没有转换正在进行（ADSTART=0），软件才可以执行写操作。

如果 ADC 已使能（可能正在进行转换）、并且没有待处理的禁用 ADC 的请求时（ADSTART=1 且 ADDIS=0），软件必须只写 ADC_CR 寄存器中的 ADSTP 位才能停止 ADC。

说明：未采取硬件保护机制来防止软件执行上述规则禁止的写操作。如果发生此类禁止的写访问，ADC 可能会进入未定义状态。要在这种情况下恢复正确的操作，必须禁止 ADC（将 ADEN 以及 ADC_CR 寄存器中的所有位都清零）。

11.2.6 通道选择

复用通道多达 9 条：

- 7 路来自 GPIO 引脚（ADC_IN0~ADC_IN6）的模拟输入。
- 1 路内部模拟输入（内部参考电压 VREFINT）。
- 1 路内部模拟输入（内部 PMU 电压）。

可转换单条通道，也可以自动扫描一系列通道。

待转换通道的顺序必须在 ADC_CHSELR 通道选择寄存器中进行编程，每条模拟输入通道都有专用的选择位（CHSEL0~CHSEL8）。

要配置通道的扫描方式，可对 ADC_CFGR1 寄存器中的 SCANDIR 位进行编程：

- SCANDIR=0：正向扫描通道 0 到通道 8
- SCANDIR=1：反向扫描通道 8 到通道 0

V_{REFINT} 内部通道

内部参考电压 VREFINT 连接至通道 ADC_IN8。

差分输入通道

差分模式输入使能后通道 0 和 1、2 和 3、4 和 5 组成差分输入（因为通道 7、8 接内部信号，所以差分输入模式下通道 6、7、8 不可用）。

- 差分通道 0，即差分信号输入是模拟通道 0 和 1，通道配置为 CHSEL0；
- 差分通道 1，即差分信号输入是模拟通道 2 和 3，通道配置为 CHSEL4；
- 差分通道 2，即差分信号输入是模拟通道 4 和 5，通道配置为 CHSEL8；

11.2.7 可编程采样时间（SMP）

开始转换之前，ADC 需要在待测量电压源与 ADC 内置采样电容之间建立直接连接。该采样时间必须足以使输入电压源为采样电容充电并将电容保持在输入电压水平。

使用可编程采样时间后，可根据输入电压源的输入电阻调整转换速度。

ADC 会在数个 ADC 时钟周期内对输入电压进行采样，时钟周期数可使用 ADC_SMPR 寄存器中的 SMP[2:0]位进行配置。

此可编程采样时间是所有通道共用的。如果应用要求，可通过软件在两次转换之间更改和调整此采样时间。

总转换时间的计算公式如下：

$$t_{CONV} = （采样时间 + 12.5）x\ ADC\ 时钟周期$$

示例：如果 ADC_CLK=16 MHz，采样时间为 1.5 个 ADC 时钟周期，则总转换时间为：

t_{CONV}=（1.5 + 12.5）*x ADC 时钟周期* = 14 个 ADC 时钟周期 = 0.875 µs

ADC 通过将 EOSMP 标志置 1 来指示采样阶段结束。

11.2.8 单次转换模式（CONT=0）

在单次转换模式下，ADC 会执行单次转换序列，对所有通道进行一次转换。当 ADC_CFGR1 寄存器中的 CONT=0 时，会选择此模式。可通过以下方式开始转换：

- 将 ADC_CR 寄存器中的 ADSTART 位置 1。
- 硬件触发事件

在序列中，每次转换完成后：

- 转换后的数据会存储在 16 位 ADC_DR 寄存器中
- EOC（转换结束）标志置 1
- EOCIE 位置 1 时将产生中断

转换序列完成后：

- EOS（序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

随后，ADC 会停止工作，直至发生新的外部触发事件或 ADSTART 位再次置 1。

说明：要转换单个通道，可将序列长度编程为 1。

11.2.9 连续转换模式（CONT=1）

在连续转换模式下，如果发生软件或硬件触发事件，ADC 会执行转换序列，对所有通道进行一次转换，随后会自动重启并持续执行相同的转换序列。当 ADC_CFGR1 寄存器中的 CONT=1 时，会选择此模式。可通过以下方式开始转换：

- 将 ADC_CR 寄存器中的 ADSTART 位置 1
- 硬件触发事件

在序列中，每次转换完成后：

- 转换后的数据会存储在 16 位 ADC_DR 寄存器中
- EOC（转换结束）标志置 1
- EOCIE 位置 1 时将产生中断

转换序列完成后：

- EOS（序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

随后，会立即重启新序列，ADC 会继续重复执行转换序列。

说明：要转换单个通道，可将序列长度编程为 1。

不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

11.2.10 开始转换（ADSTART）

软件通过将 ADSTART 置 1 的方式开始进行 ADC 转换。

ADSTART 置 1 后：

- 在 EXTEN=00 时，会立即开始转换（软件触发）。
- 在 EXTEN≠00 时，会在所选硬件触发器的下一有效边沿开始转换。

ADSTART 位还用于指示当前是否正在进行 ADC 操作。可以在 ADSTART=0 时重新配置 ADC，表明 ADC 处于空闲状态。

ADSTART 位由硬件清零：

- 在使用软件触发的单次模式下（CONT=0 且 EXTEN=00）
  - 只要转换序列结束（EOS=1）就清零
- 在使用软件触发的不连续模式下（CONT=0、DISCEN=1 且 EXTEN=00）
  - 转换结束时（EOC=1）清零

- 在所有情况下（CONT=X、EXTEN=XX）
  - 执行由软件调用的 ADSTP 程序之后（参见“11.2.12 停止正在进行的转换（ADSTP）”）清零。

说明：在连续模式下（CONT=1），由于序列会自动重新启动，因此，当 EOS 标志置 1 时，ADSTART 位不会清零。

如果在单次模式下选择了硬件触发（CONT=0，且 EXTEN≠00），当 EOS 置 1 时，ADSTART 不会由硬件清零。这样，便无需通过软件将 ADSTART 再次置 1，并可确保不会错过下一触发事件。

11.2.11 时序

从转换开始到转换结束所经过的时间是配置的采样时间与逐次趋近时间的总和：

- $t_{ADC}=t_{SMPL}+t_{SAR}=[1.5_{|min}+12.5_{|12bit}]\times t_{ADC\_CLK}$
- $t_{ADC}=t_{SMPL}+t_{SAR}=93.8ns_{|min}+781.3ns_{|12bit}=0.875\mu s_{|min}$（对于 $f_{ADC\_CLK}=16$ MHz）

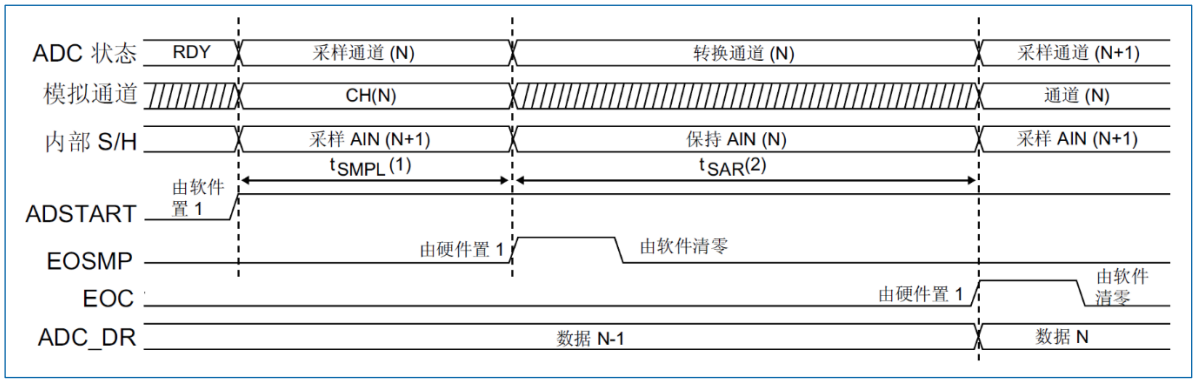

图 11-4 模数转换时间

上图的说明：

(1). t_{SMPL} 由 SMP[2:0]决定。

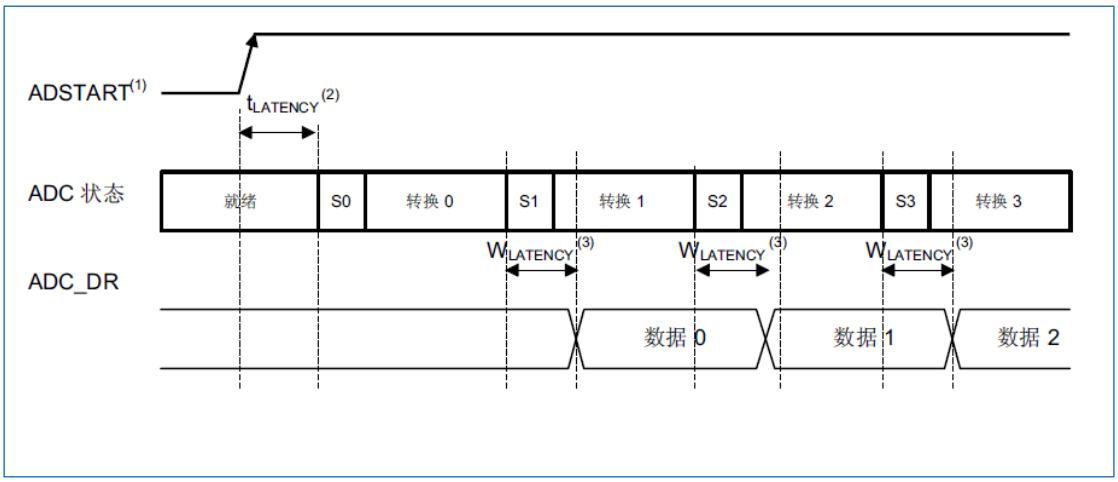

图 11-5 模数转换时序

(1). EXTEN = 00 或 EXTEN≠00

(2). 触发延迟（更多详细信息，请参见数据手册。）

(3). ADC_DR 寄存器写入延迟（更多详细信息，请参见数据手册。）

11.2.12 停止正在进行的转换（ADSTP）

可通过软件将 ADC_CR 寄存器中的 ADSTP 置 1，停止任何正在进行的转换。这会复位 ADC 操作，ADC 将处于空闲状态，准备好进行新操作。

如果 ADSTP 位由软件置 1，则会中止任何正在进行的转换，并会丢弃转换结果（ADC_DR 寄存器不会更新为当前转换结果）。

扫描序列也会中止并复位（这意味着重启 ADC 将重新开始新的序列）。一旦转换操作完成后，ADSTP 位和 ADSTART 位均由硬件清零，软件必须等待 ADSTART=0，然后才能开始进行新的转换。

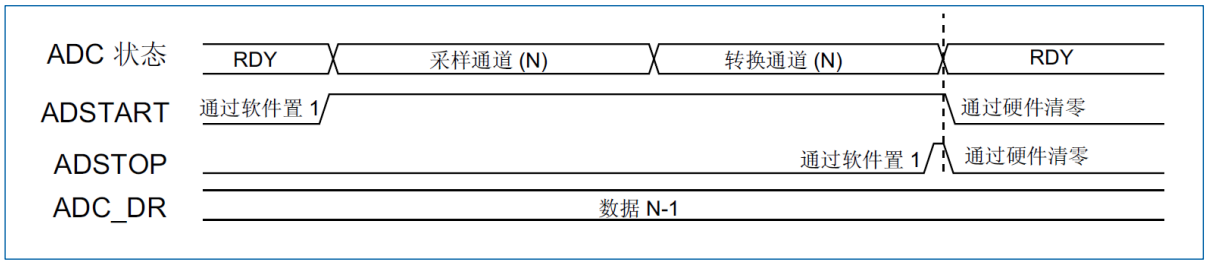

图 11-6 停止正在进行的转换

11.3 外部触发转换和触发极性（EXTSEL, EXTEN）

可通过软件或外部事件（定时器捕获事件）触发转换或转换序列。如果 EXTEN[1:0]控制位不等于“0b00”，那么外部事件能够触发所选极性的转换。软件将 ADSTART 位置 1 后，触发选择将立即生效。

在转换进行时发生的硬件触发会被忽略。

如果位 ADSTART=0，则会忽略发生的任何硬件触发。

表 11-4 提供 EXTEN[1:0]值与触发极性之间的对应关系。

表 11-4 配置触发极性

触发源检测设置	EXTEN[1:0]
禁止触发检测	00
在上升沿检测	01
在下降沿检测	10
在上升沿和下降沿均检测	11

说明：仅当 ADC 未进行转换（ADSTART= 0）时，才可以更改外部触发的极性。

EXTSEL[2:0]控制位用于选择 8 个可能的事件中哪一事件可触发转换。

表 11-5 给出了可用于常规转换的外部触发。

可将 ADC_CR 寄存器中的 ADSTART 位置 1，从而生成软件源触发事件。

表 11-5 ADC 外部触发源

名称	触发源	EXTSEL[2:0]
TRG0	TIM1_TRGO	000
TRG1	TIM1_CC4	001
TRG2	TIM2_TRGO	010
TRG3	TIM6_TRGO	011
TRG4	TIM1_CC1	100
TRG5	TIM1_CC2	101

名称	触发源	EXTSEL[2:0]
TRG6	TIM1_CC3	110
TRG7	IO_TRIG	111

说明：仅当 ADC 未进行转换（ADSTART= 0）时，才可以更改触发选择。

11.3.1 不连续模式（DISCEN）

可将 ADC_CFGR1 寄存器中的 DISCEN 位置 1，以使能此模式。

在该模式下（DISCEN=1），需要通过硬件或软件触发事件启动序列中定义的各个转换。相反，如果 DISCEN=0，单个硬件或软件触发事件会连续启动序列中定义的所有转换。

示例：

- DISCEN=1，待转换通道=0、1、3、5
  - 第一次触发：转换通道 0 并生成 EOC 事件
  - 第二次触发：转换通道 1 并生成 EOC 事件
  - 第三次触发：转换通道 3 并生成 EOC 事件
  - 第四次触发：转换通道 5 并同时生成 EOC 和 EOS 事件
  - 第五次触发：转换通道 0 并生成 EOC 事件
  - 第六次触发：转换通道 1 并生成 EOC 事件
- DISCEN=0，待转换通道=0、1、3、5
  - 第一次触发：转换整个序列：通道 0，然后是通道 1、3 和 5。每次转换都会生成 EOC 事件，最后一次转换还会生成 EOS 事件。
  - 任何后续触发事件都将重启整个序列。

说明：不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。

11.3.2 转换结束、采样阶段结束（EOC, EOSMP 标志）

ADC 指示每个转换结束（EOC）事件。

新的转换数据结果出现在 ADC_DR 寄存器中后，ADC 会立即将 ADC_ISR 寄存器中的 EOC 标志置 1。如果 ADC_IER 寄存器中的 EOCIE 位置 1，可产生中断。EOC 标志可通过软件向其写入 1 或读取 ADC_DR 寄存器的方式来清零。

ADC 还通过将 ADC_ISR 寄存器中的 EOSMP 标志置 1 来指示采样阶段结束。EOSMP 标志可通过软件向其写入 1 来清零。如果 ADC_IER 寄存器中的 EOSMPIE 位置 1，可产生中断。此中断用于处理与转换进行同步。通常情况下，可在转换阶段的隐藏时间内访问模拟复用器，这样在下次采样开始时复用器已放置好。

说明：由于采样结束与转换结束之间只有非常短的时间，因此建议使用轮询或 WFE 指令，而不建议使用中断和 WFI 指令。

11.3.3 转换序列结束（EOS 标志）

每次转换序列结束（EOS）时，ADC 都会通知应用。

转换序列的上一数据结果出现在 ADC_DR 寄存器中时，ADC 会立即将 ADC_ISR 寄存器中的 EOS 标志置 1。如果 ADC_IER 寄存器中的 EOSIE 位置 1，可产生中断。EOS 标志可通过软件向其写入 1 的方式来清零。

11.3.4 时序图示例（单次/连续模式硬件/软件触发）

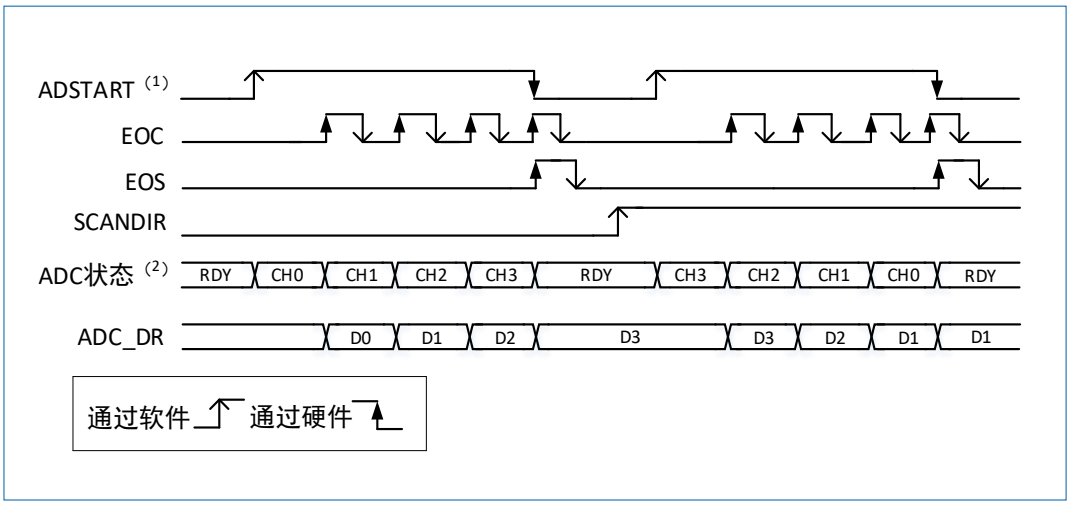

图 11-7 单次序列转换，软件触发

上图说明：

(1). EXTEN=00，CONT=0。

(2). CHSEL=0xF，WAIT=0，AUTOFF=0。

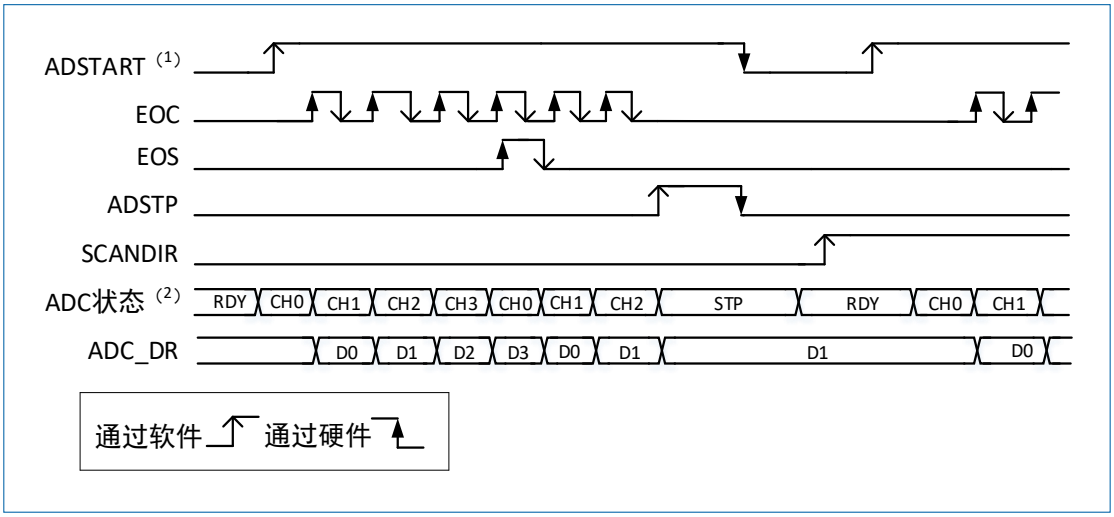

图 11-8 连续序列转换，软件触发

上图说明：

(1). EXTEN=00，CONT=1。

(2). CHSEL=0xF，WAIT=0，AUTOFF=0。

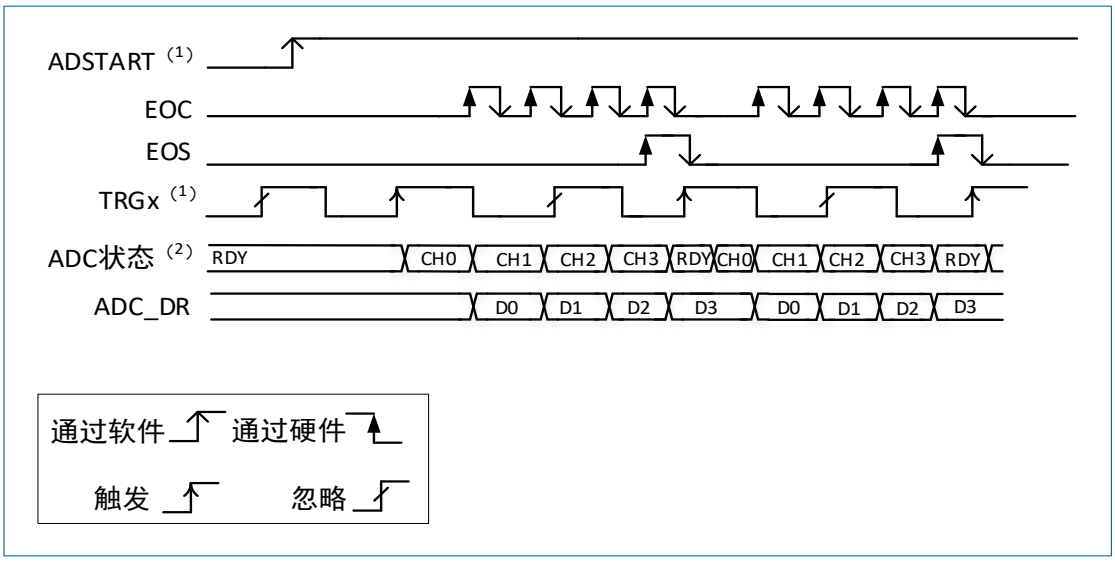

图 11-9 单次序列转换，硬件触发

上图说明：

(1). EXTSEL=TRGx（过频），EXTEN=01（上升沿）。

(2). CONT=0，CHSEL=0xF，SCANDIR=0，WAIT=0，AUTOFF=0。

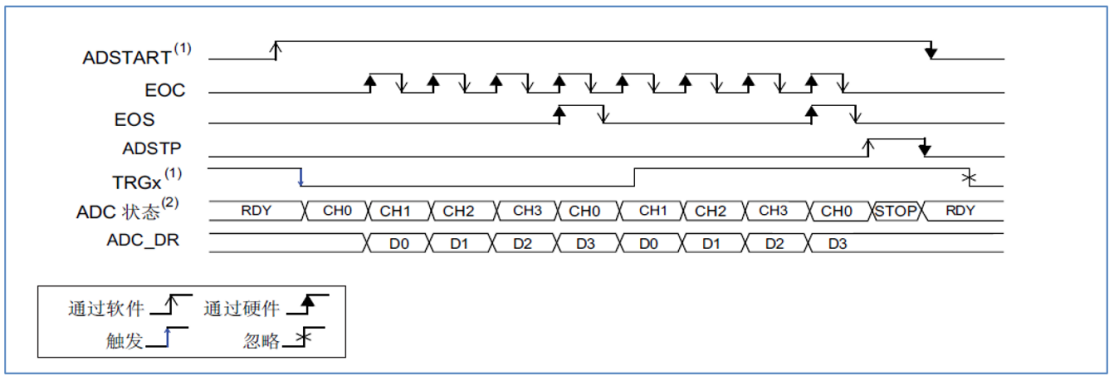

图 11-10 连续序列转换，硬件触发

上图说明：

（1）EXTSEL=TRGx，EXTEN=10（下降沿），CONT=1。

（2）CHSEL=0xF，SCANDIR=0，WAIT=0，AUTOFF=0。

11.4 数据管理

11.4.1 数据管理和数据对齐（ADC_DR, ADC_DRx, ALIGN）

每次转换结束时（发生 EOC 事件时），转换后数据的结果都会存储在宽度为 16 位的 ADC_DR 数据寄存器中。

ADC_DR 的格式取决于配置的数据对齐方式和分辨率。

ADC_CFGR1 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。数据可右对齐（ALIGN=0）或左对齐（ALIGN=1），如图 11-11 所示。

ALIGN	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0				DR[11:0]											
1	DR[11:0]												0x0			

图 11-11 数据对齐方式

通过读取寄存器 ADC_DRx 可获得对应通道 x 上最新的转换结果，可以在每一次序列转换结束的时候读取该序列的多个通道的转换结果。

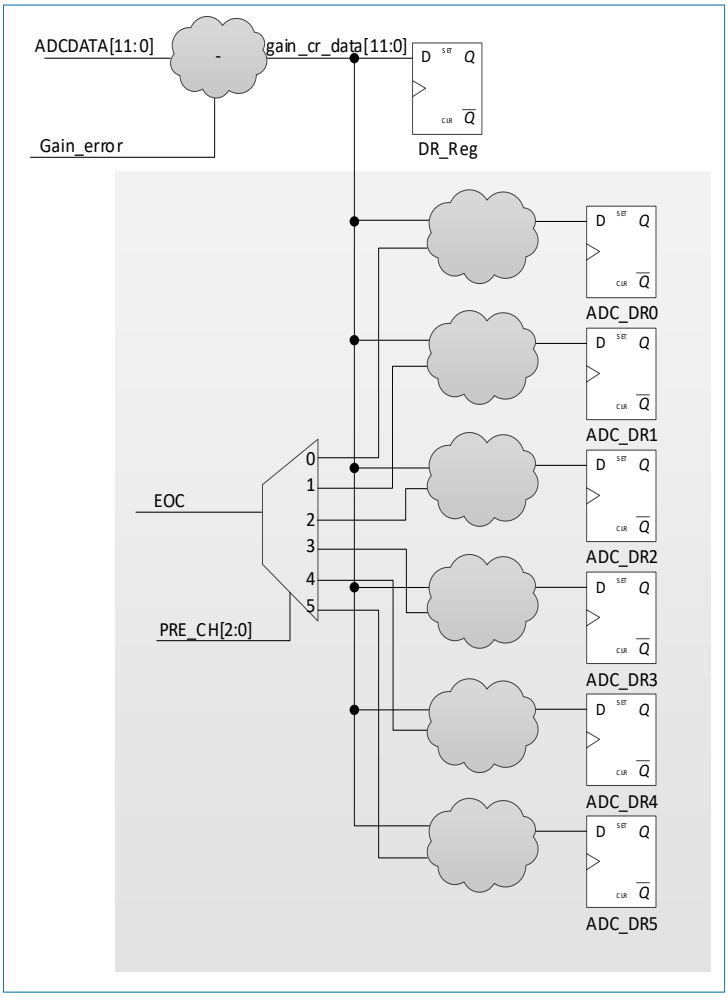

图 11-12 ADC 数据管理

11.4.2 ADC 溢出（OVR, OVRMOD）

如果转换后的数据未被及时读取，在新转换生成数据之前，会由溢出标志（OVR）指示数据溢出事件。

如果新转换完成时 EOC 标志仍为“1”，则 ADC_ISR 寄存器中的 OVR 标志会置 1。

如果 ADC_IER 寄存器中的 OVRIE 位置 1，可产生中断。

如果发生溢出情况，ADC 会保持工作状态并可继续进行转换，除非通过软件将 ADC_CR 寄存器中的 ADSTP 位置 1，从而停止并复位序列。

OVR 标志可通过软件向其写入 1 的方式来清零。

可对 ADC_CFGR1 寄存器中的 OVRMOD 位进行编程，从而配置发生溢出事件时是保留数据还是覆盖数据：

- OVRMOD=0

溢出事件会保留数据寄存器的数据，防止其被覆盖：会保留原数据，并会丢弃新的转换结果。如果

OVR 保持为 1，可继续进行转换，但会丢弃所得的数据。

- OVRMOD=1

数据寄存器会将上一次转换结果覆盖，之前未读取的数据会丢失。如果 OVR 保持为 1，可继续进行转换，ADC_DR 寄存器始终包含最新转换得出的数据。

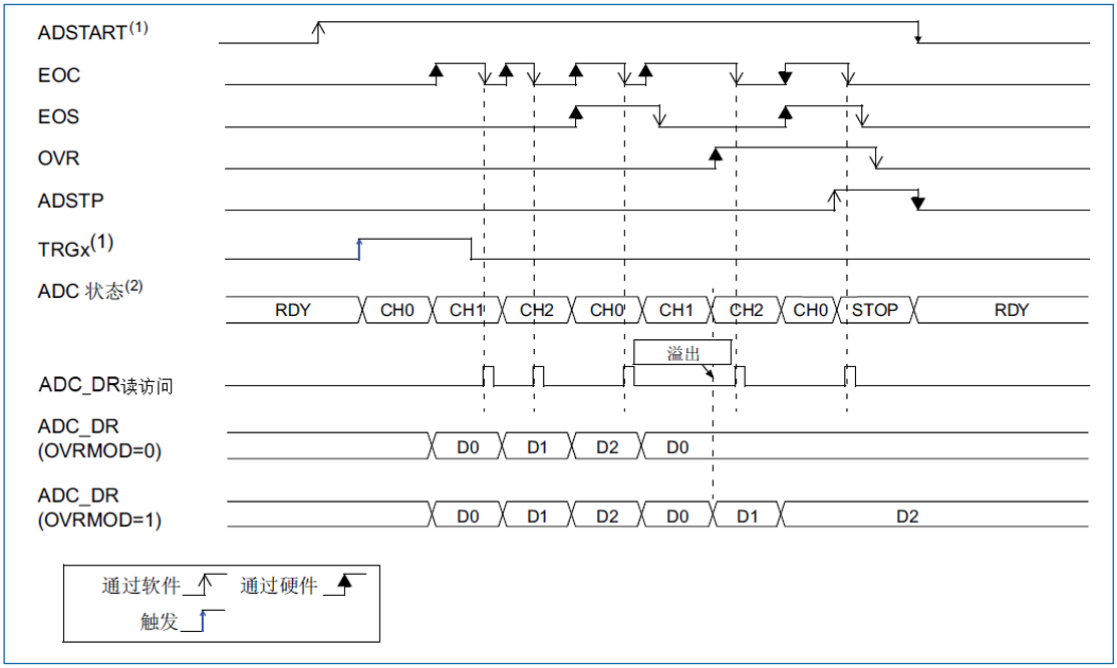

图 11-13 溢出示例（0VR）

11.5 功耗特性

11.5.1 等待模式转换（WAIT）

等待模式转换可用于简化软件，并可优化采用低频时钟的应用（此类应用可能存在 ADC 溢出的风险）的性能。

如果 ADC_CFGR1 寄存器中的 WAIT 位置 1，则仅当之前的数据已完成处理、ADC_DR 寄存器已被读取或者 EOC 位已清零，才会开始新的转换。

通过这种方式，可自动调整 ADC 的速度，使其适应系统读取数据的速度。

说明：转换进行时发生的或在读访问之前的等待时间内发生的硬件触发会被忽略。

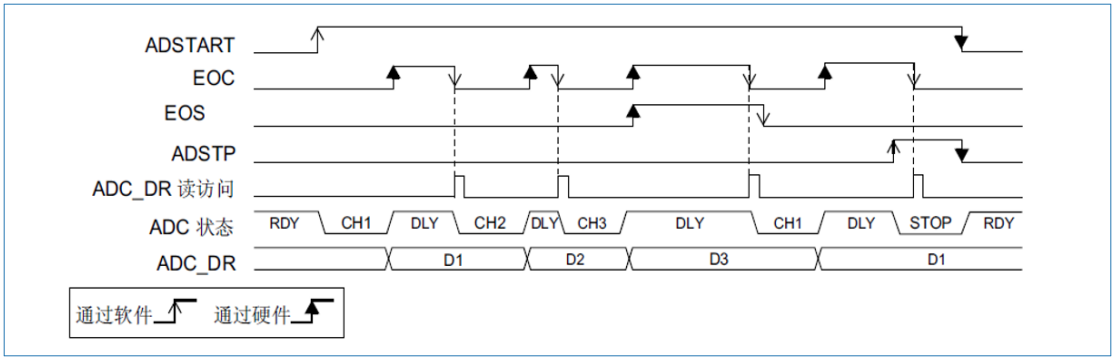

图 11-14 等待模式转换（连续模式，软件触发）

(1) EXTEN=00，CONT=1

(2) CHSEL=0x7，SCANDIR=0，WAIT=1，AUTOFF=0

11.5.2 自动关闭模式（AUTOFF）

ADC 具有自动电源管理功能，也称为自动关闭模式，将 ADC_CFGR1 寄存器中的 AUTOFF 位置 1 可使能此模式。

如果 AUTOFF=1，ADC 始终会在未进行转换时关闭，并会在转换开始后自动唤醒（通过软件或硬件触发）。在启动转换的触发事件和 ADC 的采样时间之间，会自动插入启动时间。随后，转换序列完成后，ADC 会自动禁止。

如果应用需要进行的转换次数相对较少，或者为了合理调整开关 ADC 使用的额外功率和时间而将转换请求的间隔时间设定得足够长（例如采用低频硬件触发），使用自动关闭模式可显著降低应用的功耗。

对于采用低频时钟的应用，可将自动关闭模式与等待模式转换（WAIT=1）结合使用，如果 ADC 在等待过程中自动掉电，将会在应用读取 ADC_DR 寄存器后立即重启。这种组合可显著降低功耗（请参见图 11-15。

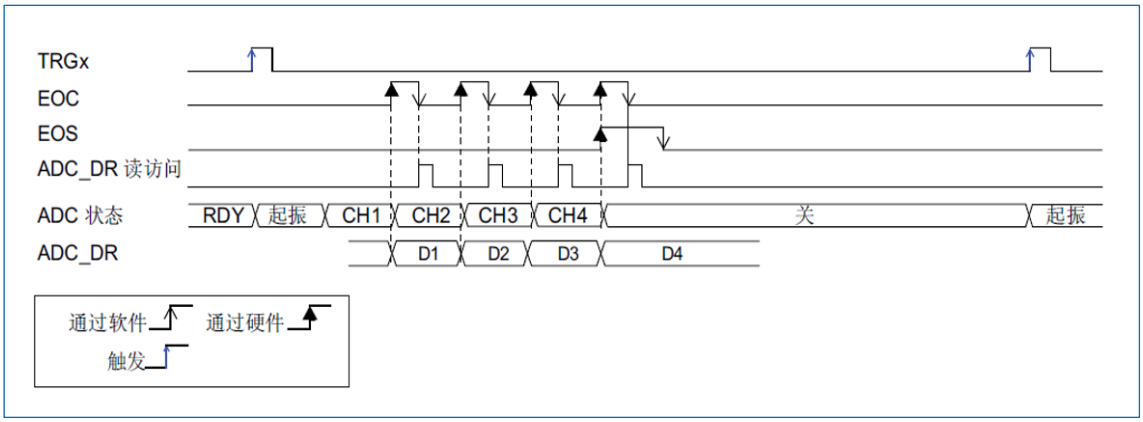

图 11-15 WAIT=0、AUTOFF=1 时的行为

上图的说明：EXTSEL=TRGx，EXTEN=01（上升沿），CONT=x，ADSTART=1，CHSEL=0x0E，SCANDIR=0，WAIT=1，AUTOFF=1。

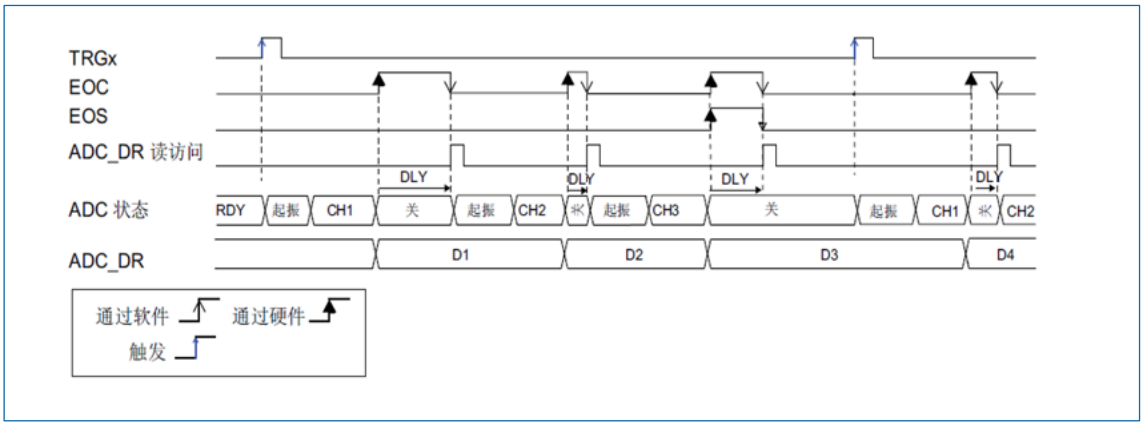

图 11-16 WAIT=1、AUTOFF=1 时的行为

上图的说明：EXTSEL=TRGx，EXTEN=01（上升沿），CONT=x，ADSTART=1，CHSEL=0x0E，SCANDIR=0，WAIT=1，AUTOFF=1。

11.6 模拟窗口看门狗（AWDEN, AWDSGL, AWDCH, HT/LT, AWD）

将 ADC_CFGR1 寄存器中的 AWDEN 位置 1，可使能 AWD 模拟看门狗功能。此功能用于监控一条选定的通道或所有已使能的通道（参见表 11-6）是否仍处于所配置的电压范围（窗口）内，如图 11-17 所示。

如果 ADC 转换的模拟电压低于阈值下限或高于阈值上限，则会将 AWD 模拟看门狗状态位置 1。这些阈值在 ADC_TR 寄存器中的 HT 位和 LT 位进行编程。可以通过将 ADC_IER 寄存器中的 AWDIE 位置 1 来

使能中断。

AWD 标志可通过软件向其写入 1 的方式来清零。

表 11-6 介绍了如何配置 ADC_CFGR1 寄存器中的 AWDSGL 位和 AWDEN 位，以使能一条或多条通道上的模拟看门狗。

表 11-6 模拟看门狗通道选择

模拟看门狗保护的通道	AWDSGL 位	AWDEN 位
无	X	0
所有通道	0	1
单通道(1)	1	1

(1). 通过 AWDCH[3:0]位选择模拟看门狗监控的输入通道。

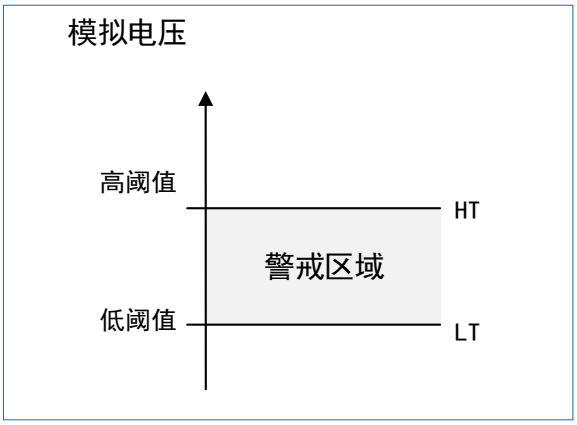

图 11-17 模拟看门狗的保护区域

11.7 内部参考电压

内部参考电压（V_{REFINT}）为 ADC 提供了一个稳定的（带隙基准）电压。V_{REFINT} 内部连接到 ADC_IN8 输入通道。

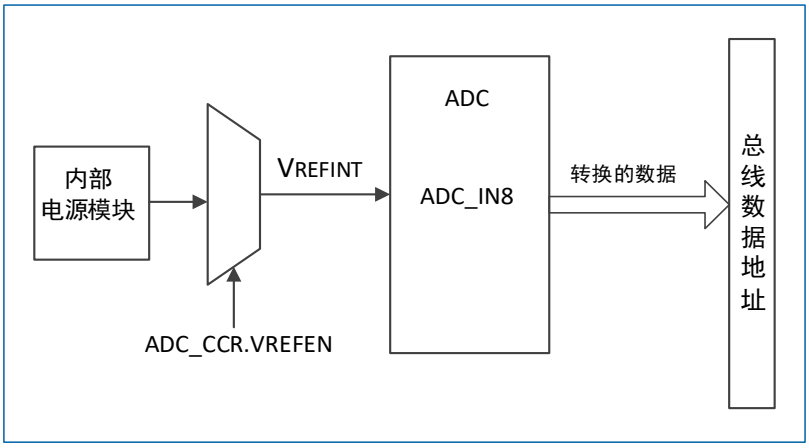

图 11-18 V_{REFINT} 通道框图

使用内部参考电压计算实际的 V_{DDA} 电压

施加给 MCU 的 V_{DDA} 电源电压可能会有变化，或无法获得准确值。在制造过程中由 ADC 在 V_{DDA}=5V 的条件下获得的内置内部参考电压（V_{REFINT}）及其校准数据可用于评估实际的 V_{DDA} 电压水平。

以下公式可求得为器件供电的实际的 V_{DDA} 电压：

$V_{DDA}=5 * V_{REFINT\_CAL}/V_{REFINT\_S}$

其中：

- V_{REFINT_S} 表示内部参考电压的实际采样值。
- V_{REFINT_CAL} 是内部参考电压 V_{REFINT} 校准值。V_{REFINT} 的精确电压在生产测试期间由航顺测量，并存储在系统内存区域，见下表。

表 11-7 内部电压基准测量值

校准值名称	描述	存储器地址
V_{REFINT_CAL}	在 25°C 温度下获得的原始数据，V_{DDA} = 5V	0x1FFF F124-0x1FFF F125

将电源相关的 ADC 测量值转换为绝对电压值

ADC 用于提供对应于模拟电源与施加给转换通道的电压之比的数字值。对于大部分应用用例，需要将该比值转换成与 V_{DDA} 无关的电压。对于 V_{DDA} 已知、ADC 转换值进行了右对齐的应用，可使用以下公式得到该绝对值：

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

对于 V_{DDA} 值未知的应用，必须使用内部参考电压，V_{DDA} 可替换为使用内部参考电压计算实际的 V_{DDA} 电压部分提供的表达式，从而得出以下公式：

$$V_{CHANNELx}=\frac{5V\times V_{REFINT\_CAL}\times ADC\_DATA_x}{V_{REFINT\_DATA}\times FULL\_SCALE}$$

其中：

- V_{REFINT_CAL} 是 V_{REFINT} 校准值。
- ADC_DATA_x 是由 ADC 在通道 x 上测得的值（右对齐）。
- V_{REFINT_DATA} 是由 ADC 转换得到的实际 V_{REFINT} 输出值。
- FULL_SCALE 是 ADC 输出的最大数字值。分辨率为 12 位，该值为 2^{12} - 1 = 4095。

说明：如果执行 ADC 测量时使用的是输出格式而非 12 位右对齐格式，那么必须先将所有参数转换为兼容格式，然后再进行计算。

11.8 ADC 中断

发生下列任一事件均可生成中断：

- ADC 使能后，ADC 就绪（ADRDY 标志）
- 任何转换结束（EOC 标志）
- 转换序列结束（EOS 标志）
- 进行模拟看门狗检测时（AWD 标志）
- 采样阶段结束时（EOSMP 标志）
- 发生数据溢出时（OVR 标志）

可以使用单独的中断使能位以提高灵活性。

表 11-8 ADC 中断

中断事件	事件标志	使能控制位
ADC 就绪	ADRDY	ADRDYIE

中断事件	事件标志	使能控制位
转换结束	EOC	EOCIE
转换序列结束	EOS	EOSIE
模拟看门狗状态位置 1	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
上溢	OVR	OVRIE

11.9 ADC 增益补偿功能

该功能是用于修正板级和封装等原因造成的内部 ADC 模拟接口上的 V_{REFP} < 5 V 和 V_{REFN} > 0 V 导致的转换结果大于实际值的增益偏差。

假设 ADC 转换结果换算出的电压值为 V_{ADC}，实际输入的电压大小为 V_{AIN}，那么 V_{AIN}/（V_{REFP}-V_{REFN}） = V_{ADC}/5V，所以修正方法是 V_{AIN} = V_{ADC}*（（V_{REFP}-V_{REFN}）/5V）=V_{ADC} - X，X=V_{ADC} *（5-（V_{REFP}-V_{REFN}））/5。ADC 控制电路增加了两个增益因子寄存器得到这个 X 值，由硬件逻辑自动消除增益误差。X=V_{gain1}+V_{gain2}。实际使用 gain1，gain2 配置增益偏差最大值。

该功能有两个局限性，一是只能修正转换结果大于实际值的情况，二是修正后的最高电压值不能到满幅 5 V，所以 ADC 的转换结果范围就限制在了 0~（4095-（gain1+gain2））。

11.9.1 增益补偿因子

增益补偿因子配置到 0，采样电压输入 0~VREFP 的条件下，得到增益线性误差最大值。根据这个最大值配置增益补偿因子的值。

配置方式举例：

- GAIN_SEL[2:0]=000：表示不调整。
- GAIN_SEL[2:0]=001：表示调整倍数为 2^11 = 2048，调整大小最大为 4095>>11=1LSB，即调整大小变为 DR 寄存器值=采样值-采样值>>11。
- GAIN_SEL[2:0]=010：表示调整倍数为 2^10 = 1024，调整大小最大为 4095/1024=3LSB，即调整大小变为 DR 寄存器值=采样值-采样值>>10。
- GAIN_SEL[2:0]=011：表示调整倍数为 2^9 = 512，调整大小最大为 4095/512=7LSB，即调整大小变为 DR 寄存器值=采样值-采样值>>9。
- GAIN_SEL[2:0]=100：表示调整倍数为 2^8 = 256，调整大小最大为 4095/256=15LSB，即调整大小变为 DR 寄存器值=采样值-采样值>>8。
- GAIN_SEL[2:0]=101：表示调整倍数为 2^7 = 128，调整大小最大为 4095/128=31LSB，即调整大小变为 DR 寄存器值=采样值-采样值>>7。

通过配合 ADC_GAIN 寄存器中的两个 GAIN_SEL1/2 使用，就能配置成 0~62 的最大线性增益误差修正值。

11.10 ADC 寄存器

基地址：0x4001 2400

空间大小：0x400

11.10.1 ADC 中断和状态寄存器（ADC_ISR）

偏移地址：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								AWD	Res		OVR	EOS	EOC	EOSMP	ADRDY
								rw			rw	rw	rw	rw	rw

位 31:8	Res：保留 必须保持复位值。
位 7	AWD：模拟看门狗标志（End of calibration flag） 当转换后的电压超过 ADC_TR 寄存器编程设定的电压时，该位由硬件置位。用软件对该位写 1 清除。 • 0：未发生模拟看门狗事件（或该事件标志由软件获取并清零） • 1：发生了模拟看门狗事件
位 6:5	Res：保留 必须保持复位值。
位 4	OVR：ADC 溢出（ADC overrun） 当溢出产生时该位由硬件置位，置位说明新的转换结束但 EOC 位还是为 1。 该位可由软件写 1 清零。 • 0：未发生溢出事件（或该事件标志由软件获取并清零） • 1：发生了溢出事件
位 3	EOS：序列转换结束标志（End of sequence flag） 由 CHSEL 位所选的通道序列转换结束后，该位由硬件置位。其由软件对该位写 1 清零。 • 0：序列转换未完成（或该事件标志由软件获取并清零） • 1：序列转换已完成
位 2	EOC：转换结束标志（End of conversion flag） 当每个通道新转换结果有效时（存放在 ADC_DR 中）该位由硬件置位。可由软件对该位写 1 清零或读取 ADC_DR 寄存器来清零。 • 0：通道转换未结束（或该事件标志由软件获取并清零或由读 ADC_DR 寄存器清零） • 1：通道转换已结束
位 1	EOSMP：采样结束标志（End of sampling flag） 在转换期间的采样阶段结束时该位由硬件置 1。 • 0：采样阶段未结束（或该事件标志由软件确认并清零） • 1：采样阶段已结束
位 0	ADRDY：ADC 就绪（ADC ready） ADC 使能后（位 ADEN=1）以及 ADC 达到准备好接收转换请求的状态时，会通过硬件将该位置 1。该位用软件写 1 清除。 • 0：ADC 未准备好（或该标志事件由软件确认并清零）

	• 1：ADC 已准备好开始转换

11.10.2 ADC 中断使能寄存器（ADC_IER）

偏移地址：0x04

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								AWDIE	Res		OVRIE	EOSIE	EOCIE	EOSMPIE	ADRDYIE
								rw			rw	rw	rw	rw	rw

位 31:8	Res：保留 必须保持复位值。
位 7	AWDIE：模拟看门狗中断使能（Analog watchdog interrupt enable） 该位由软件设置和清除来开启/关闭模拟看门狗中断。 • 0：模拟看门狗中断禁止 • 1：模拟看门狗中断使能
位 6:5	Res：保留 必须保持复位值。
位 4	OVRIE：溢出中断使能（Overrun interrupt enable） 该位由软件设置和清除来开启/关闭溢出中断。 • 0：溢出中断关闭。 • 1：溢出中断开启。当 OVR 位置位时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 3	EOSIE：序列转换结束中断使能（End of conversion sequence interrupt enable） 该位由软件设置和清除来开启/关闭序列转换结束中断 • 0：EOS 中断关闭 • 1：EOS 中断开启。当 EOS 置位时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 2	EOCIE：转换结束中断使能（End of conversion interrupt enable） 该位由软件设置和清除来开启/关闭转换结束中断。 • 0：EOC 中断关闭 • 1：EOC 中断开启。当 EOC 置位时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 1	EOSMPIE：采样结束中断使能（End of sampling flag interrupt enable）

	该位由软件设置和清除来开启/关闭采样阶段结束中断。 • 0：EOSMP 中断关闭 • 1：EOSMP 中断开启。当 EOSMP 置位时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 0	ADRDYIE：ADC 准备好中断使能（ADC ready interrupt enable） 该位由软件设置和清除来开启/关闭 ADC 准备好中断 • 0：ADRDY 中断关闭 • 1：ADRDY 中断开启。当 ADRDY 置位时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*

11.10.3 ADC 控制寄存器（ADC_CR）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res											ADSTP	Res	ADSTART	ADDIS	ADEN
											rs		rs	rs	rs

位 31:5	Res：保留 必须保持复位值。
位 4	ADSTP：ADC 停止转换命令（ADC stop conversion command） 该位由软件设置来停止和丢弃正在进行中的转换。当转换停止结束时，该位由硬件清零且 ADC 已准备好接受新的转换命令。 • 0：不发 ADC 停止转换命令。 • 1：写 1 用来停止 ADC，读为 1 时表明 ADSTP 命令正在执行中。 *注意：仅当 ADSTART=1 且 ADDIS=0 时（ADC 已使能，可能正在进行转换，并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 ADSTP 置 1。*
位 3	Res：保留 必须保持复位值。
位 2	ADSTART：ADC 开始转换命令（ADC start conversion command） 该位由软件设置来启动 ADC 转换。一次转换可由立即启动（由软件配置）或硬件触发产生（硬件触发配置）两种方式来启动，启动方式由 EXTEN[1:0]位的配置来决定。 • 0：无进行中的 ADC 转换。 • 1：写 1 开始 ADC 转换。读为 1 表明 ADC 正在进行转换。 其位由硬件清零： • 在单次转换模式中（CONT=0、DISCEN=0），当选择为软件触发 （EXTSEL=0x0）时：

	序列转换结束（EOS 置位）后该位清零。 • 在不连续转换模式下（CONT=0、DISCEN=1），如果选择了软件触发（EXTEN=00）：出现转换结束（EOC）标志时清零。 *在所有其他情况下：执行完 ADSTP 命令后，由硬件将 ADSTP 位清零的同时清零该位。* *注意：仅当 ADEN=1 且 ADDIS=0 时（ADC 已使能，并且没有任何待处理的禁止 ADC 的请求），才允许软件将 ADSTART 置 1。*
位 1	ADDIS：ADC 禁止命令（ADC disable command） 该位由软件设置来禁止 ADC（ADDIS 命令）并让 ADC 处于掉电状态（关断状态）。一旦 ADC 有效关闭（ADEN 同时被硬件清零）后由硬件清除该位。 • 0：无 ADDIS 命令进行中 • 1：写 1 为关闭 ADC。读为 1 时表明 ADDIS 命令正在执行中。 *注意：仅当 ADEN=1 且 ADSTART=0 时（确保当前未进行任何转换），才允许软件将 ADDIS 置 1。*
位 0	ADEN：ADC 使能命令（ADC enable command） 由软件设置该位来使能 ADC。一旦 ADRDY 标志置为 1 时表明 ADC 可供使用了。执行 ADDIS 命令后，ADC 关断且该位被硬件清零。 • 0：禁用 ADC（关断状态） • 1：使能 ADC *注意：仅当 ADC_CR 寄存器的所有位均为 0 时（ADSTP=0、ADSTART=0、ADDIS=0 且 ADEN=0），才允许软件将 ADEN 位置 1。*

11.10.4 ADC 配置寄存器 1（ADC_CFGR1）

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res		AWDCH[3:0]				Res		AWDEN	AWDSGL	Res					DISCEN
		rw						rw	rw						rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AUTOFF	WAIT	CONT	OVRMOD	EXTEN[1:0]		Res	EXTSEL[2:0]			ALIGN	Res		SCANDIR	Res	
rw	rw	rw	rw	rw			rw			rw			rw		

位 31:30	Res：保留 必须保持复位值。
位 29:26	AWDCH[3:0]：模拟看门狗通道选择（Analog watchdog channel selection） 这些位由软件设置和清除。它们设置模拟看门狗监控的输入通道。 • 0000：模拟看门狗监控 ADC 模拟输入通道 0 • 0001：模拟看门狗监控 ADC 模拟输入通道 1 • 0010：模拟看门狗监控 ADC 模拟输入通道 2

	• 0011：模拟看门狗监控 ADC 模拟输入通道 3 • 0100：模拟看门狗监控 ADC 模拟输入通道 4 • 0101：模拟看门狗监控 ADC 模拟输入通道 5 • 0110：模拟看门狗监控 ADC 模拟输入通道 6 • 0111：模拟看门狗监控 ADC 模拟输入通道 7 • 1000：模拟看门狗监控 ADC 模拟输入通道 8 • 其他值：保留，不会被使用 *注意：由 AWDCH[3:0]位选择的通道也必须设置到 CHSELR 寄存器中。* *仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 25:24	Res：保留 必须保持复位值。
位 23	AWDEN：模拟看门狗使能位（Analog watchdog enable） 该位由软件设置和清除。 • 0：模拟看门狗关闭 • 1：模拟看门狗开启 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 22	AWDSGL：在单一通道或所有通道使能看门狗（Enable the watchdog on a single channel or on all channels） 该位由软件设置和清除来使能由 AWDCH[3:0]位指定的通道或所有通道。 • 0：在所有通道上使能模拟看门狗 • 1：在单一通道上使能模拟看门狗 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 21:17	Res：保留 必须保持复位值。
位 16	DISCEN：断续模式（Discontinuous mode） 该位由软件设置和清除来开启/禁止断续模式。 • 0：断续模式禁止 • 1：断续模式开启 *注意：不能同时使能不连续模式和连续模式：禁止将 DISCEN 和 CONT 位同时置 1。* *仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 15	AUTOFF：自动关断模式（Auto-off mode） 该位由软件设置和清除来开启/禁止自动关断模式。 • 0：自动关断模式禁止

	• 1：自动关断模式开启 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 14	WAIT：等待转换模式（Wait conversion mode） 此位由软件置 1 和清零，以开启/禁用等待转换模式。 • 0：等待转换模式关闭 • 1：等待转换模式开启 *注意：只有当 ADSTART= 0 时，软件才允许写入此位（为了确保没有转换正在进行）。*
位 13	CONT：单次/连续转换模式（Single/continuous conversion mode） 该位由软件设置和清除。若该位置位，转换为连续模式直到该位清零。 • 0：单次转换模式 • 1：连续转换模式 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 12	OVRMOD：溢出管理模式（Overrun management mode） 该位由软件设置和清除来配置数据溢出管理。OVR 标志可通过软件向其写入 1 的方式来清零。 • 0：当检测到溢出事件时，ADC_DR 寄存器保持为原有数据。 • 1：当检测到溢出事件时，ADC_DR 寄存器上一次的转换结果覆盖。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 11:10	EXTEN[1:0]：外部触发使能和极性选择（External trigger enable and polarity selection） 这些位可由软件设置和清除来选择外部触发的极性并使能触发器。 • 00：硬件触发检测关闭（可由软件启动转换） • 01：在上升沿进行硬件触发检测 • 10：在下降沿进行硬件触发检测 • 11：在上升和下降沿进行硬件触发检测 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 9	Res：保留 必须保持复位值。
位 8:6	EXTSEL[2:0]：外部触发选择（External trigger selection） 这些位用于选择触发 ADC 转换的外部事件： • 000：事件 0（TIM1_TRGO） • 001：事件 1（TIM1_CC4） • 010：事件 2（TIM2_TRGO） • 011：事件 3（TIM6_TRGO）

	• 100：事件 4（TIM1_CC1） • 101：事件 5（TIM1_CC2） • 110：事件 6（TIM1_CC3） • 111：事件 7（IO_TRIG） *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 5	ALIGN：数据对齐（Data alignment） 该位由软件设置和清除用来选择数据的左或右对齐能 • 0：右对齐 • 1：左对齐 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 4:3	Res：保留 必须保持复位值。
位 2	SCANDIR：扫描序列方向（Scan sequence direction） 该位由软件设置和清除来选择通道序列中的通道扫描方向。 • 0：向前扫描（从 CHSEL0 到 CHSEL8） • 1：向后扫描（从 CHSEL8 到 CHSEL0） *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 1:0	Res：保留 必须保持复位值。

11.10.5 ADC 配置寄存器 2（ADC_CFGR2）

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CKMODE[1:0]		Res													
rw															

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

位 31:30	CKMODE[1:0]：ADC 时钟模式（ADC clock mode） 软件设置或清零。 • 00：ADCCLK 异步时钟工作模式 • 01：PCLK/2 同步时钟工作模式 • 10：PCLK/4 同步时钟工作模式 • 11：保留

	在所有异步时钟模式下，从定时器触发到转换开始的延迟过程中，不存在抖动。 *注意：仅当 ADC 已禁止时（ADSTART=0、ADSTP=0、ADDIS=0 且 ADEN=0），才允许通过软件对该位域执行写操作。*
位 29:0	Res：保留 必须保持复位值。

11.10.6 ADC 采样时间寄存器（ADC_SMPR）

偏移地址：0x14

复位值：0x0000 0000

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

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

位 31:3	Res：保留 必须保持复位值。
位 2:0	SMP[2:0]：采样时间选择（Sampling time selection） 该位域由软件写入，用于选择应用于所有通道的采样时间。 • 000：1.5 ADC 时钟周期 • 001：7.5 ADC 时钟周期 • 010：13.5 ADC 时钟周期 • 011：28.5 ADC 时钟周期 • 100：41.5 ADC 时钟周期 • 101：55.5 ADC 时钟周期 • 110：71.5 ADC 时钟周期 • 111：239.5 ADC 时钟周期 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*

11.10.7 ADC 看门狗阈值寄存器（ADC_TR）

偏移地址：0x20

复位值：0x00000FFF

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

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

位 31:28	Res：保留 必须保持复位值。
位 27:16	HT[11:0]：模拟看门狗的高阈值（Analog watchdog higher threshold） 这些位由软件改写，用来定义模拟看门狗的高阈值。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 15:12	Res：保留 必须保持复位值。
位 11:0	LT[11:0]：模拟看门狗低阈值（Analog watchdog lower threshold） 这些位由软件改写，用来定义模拟看门狗的低阈值。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*

11.10.8 ADC 通道选择寄存器（ADC_CHSELR）

偏移地址：0x28

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							CHSEL8	CHSEL7	CHSEL6	CHSEL5	CHSEL4	CHSEL3	CHSEL2	CHSEL1	CHSEL0
							rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:9	Res：保留 必须保持复位值。
位 x (x=8..0)	CHSELx：通道选择（Channel-x selection） 这些位可由软件改写，用来定义所要转换序列的通道。 • 0：输入通道 x 不被选为转换通道 • 1：输入通道 x 被选为转换通道 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*

11.10.9 ADC 数据寄存器（ADC_DR）

偏移地址：0x40

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	DATA[15:0]：转换数据（Converted data） 这些位只读。 其包含最后转换通道的转换结果值。数据可以采用左对齐和右对齐，参见图 11-11。

11.10.10 ADC 通道 x 数据寄存器（ADC_DRx）（x=0..8）

偏移地址：0x3C0+0x4*x

复位值：0x0000 0000

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

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

位 31:12	Res：保留 必须保持复位值。
位 11:0	DRx[11:0]：通道 x 最新一次的转换结果（Channel x latest converted data） 这些位只读。 该寄存器只支持数据左对齐，且新的数据会覆盖未读取的数据（OVRMOD=0 不起作用）。

11.10.11 ADC 通用配置寄存器（ADC_CCR）

偏移地址：0x308

复位值：0x0000 0000

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

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

位 31:23	Res：保留 必须保持复位值。
位 22	VREFEN：VREFINT 使能（VREFINT enable）

	由软件设置和清除来打开/关闭 VREFINT 通道。 • 0：VREFINT 通道关闭 • 1：VREFINT 通道开启
位 21:0	Res：保留 必须保持复位值。

11.10.12 ADC 控制寄存器 2（ADC_CR2）

偏移地址：0x3F0

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														SDIF	GCMP
														rw	rw

位 31:2	Res：保留 必须保持复位值。
位 1	SDIF：差分输入使能（Differential input enable） 使能后通道 0 和 1、2 和 3、4 和 5 组成差分输入（因为通道 7 连接内部 PMU，通道 8 接内部参考电压 V_{REFINT}，所以差分输入模式下通道 6、7、8 不可用）。 • 0：ADC 通道单端输入模式（默认） • 1：ADC 通道差分输入模式
位 0	GCMP：ADC 内部延迟控制（ADC internal delay control） • 0：比较器低增益模式（默认） • 1：比较器高增益模式

11.10.13 ADC 校准误差寄存器（ADC_OFFSET）

偏移地址：0x3F4

复位值：0x0000 0000

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

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

位 31:7	Res：保留 必须保持复位值。
位 6	OFFSET_EN：偏移误差校准使能（Calibration Offset enable） 使能后将 ADC 将会进行 Offset 误差校准，置 0，则不会进行 Offset 误差校准。

	• 0：不进行 Offset 误差校准 • 1：进行 Offset 误差校准
位 5:0	CAL_OFFSET[5:0]：ADC 偏移误差校准值（ADC Calibration Offset data） 这些位可读可写。 在校准完成时，这些值表示 Offset 的大小，CAL_OFFSET[5]最高位是符号位，0 表示 OFFSET data 为正，1 表示 OFFSET data 为负，CAL_OFFSET[4:0]是数据位；OFFSET data 如果是正值 CAL_OFFSET[4:0]是正常数据，OFFSET data 如果是负值 CAL_OFFSET[4:0]是补码数据。

11.10.14 ADC 增益补偿寄存器（ADC_GAIN）

偏移地址：0x3F8

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res										GAIN_SEL2[2:0]			GAIN_SEL1[2:0]		
										rw			rw		

位 31:6	Res：保留 必须保持复位值。
位 5:3	GAIN_SEL2[2:0]：增益修正因子 2 大小配置（Gain calibration factor 2 selection） 此位由软件置 1 和清零，用于增益修正因子 2 的大小配置。 • 000：增益修正因子 2 大小为 0（默认） • 001：增益修正因子 2 大小为 ADC_DR[11] • 010：增益修正因子 2 大小为 ADC_DR[11:10] • 011：增益修正因子 2 大小为 ADC_DR[11:9] • 100：增益修正因子 2 大小为 ADC_DR[11:8] • 101：增益修正因子 2 大小为 ADC_DR[11:7] • 其他值：保留
位 2:0	GAIN_SEL1[2:0]：增益修正因子 1 大小配置（Gain calibration factor 1 selection） 此位由软件置 1 和清零，用于增益修正因子 1 的大小配置。 • 000：增益修正因子 1 大小为 0（默认） • 001：增益修正因子 1 大小为 ADC_DR[11] • 010：增益修正因子 1 大小为 ADC_DR[11:10] • 011：增益修正因子 1 大小为 ADC_DR[11:9] • 100：增益修正因子 1 大小为 ADC_DR[11:8] • 101：增益修正因子 1 大小为 ADC_DR[11:7] • 其他值：保留

12 高级控制定时器（TIM1）

高级控制定时器（TIM1）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

高级控制定时器适合多种用途，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、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）和其他定时器完全独立，不共享任何资源。

表 12-1 TIM1 特性

符号	参数	条件	最小值	典型值	最大值	单位
$t_{res（TIM）}$	定时器分辨时间	$f_{TIMxCLK}$=48 MHz	-	20.8	-	ns
f_{EXT}	定时器的 CH1 至 CH4，外部输入的时钟频率	-	-	$f_{TIMxCLK}/2$	-	MHz
		$f_{TIMxCLK}$=48 MHz	-	24	-	MHz
t_{MAX_COUNT}	当选择内部时钟时，16 位计数器的时钟周期	-	-	2^{16}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48 MHz	-	1365	-	μs

12.1 TIM1 主要功能

TIM1 定时器的功能包括：

- 16 位向上、向下、向上/向下自动装载计数器
- 16 位可编程（可实时修改）预分频器，计数器时钟频率的分频系数为 1~65536 之间的任意数值。
- 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 以下事件发生时产生中断：
  - 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

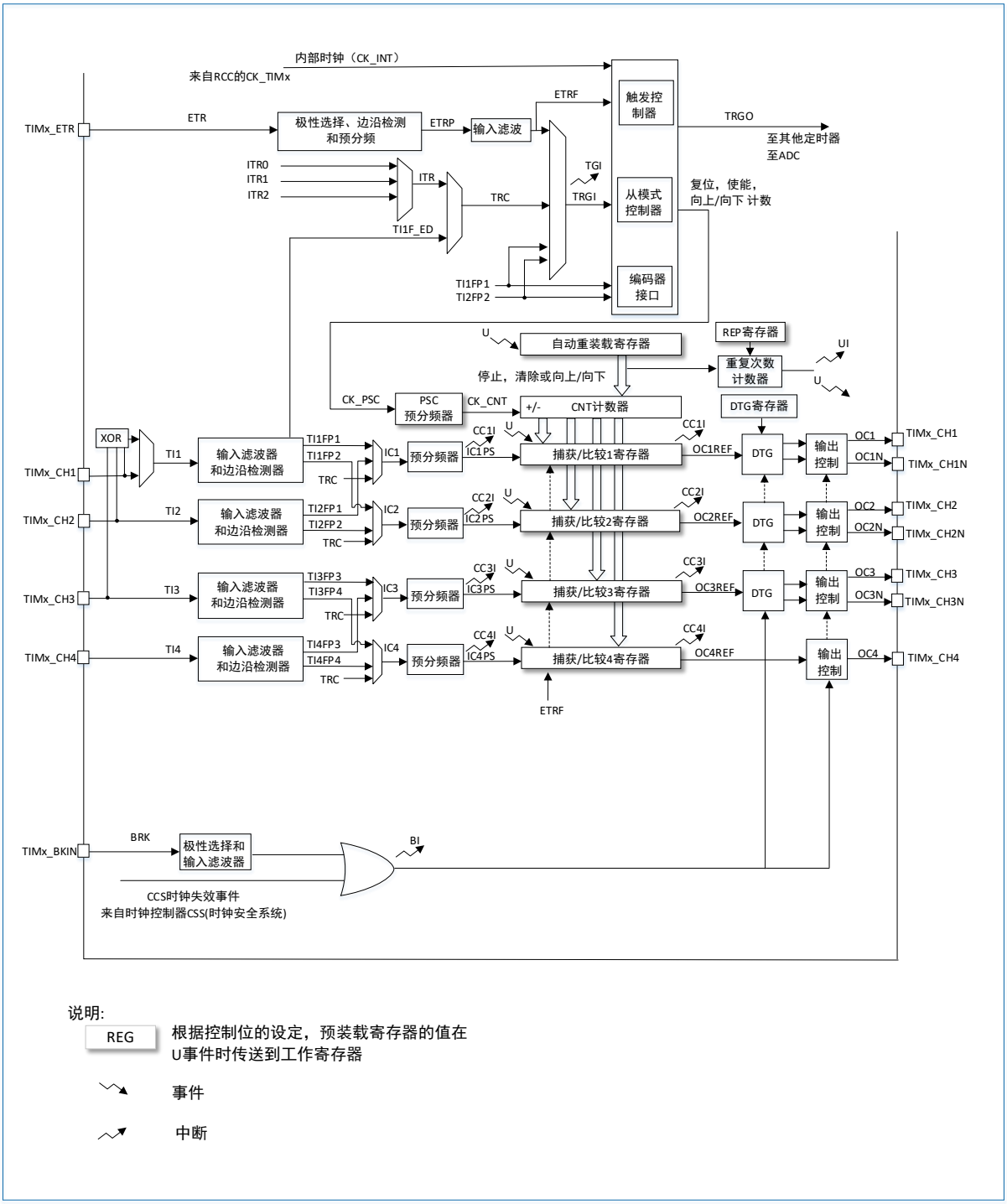

图 12-1 高级控制定时器框图

12.2 TIM1 功能描述

12.2.1 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（TIM1_CNT）

- 预分频器寄存器（TIM1_PSC）
- 自动装载寄存器（TIM1_ARR）
- 重复次数寄存器（TIM1_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 TIM1_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TIM1_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。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 TIM1_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“12.2.19 TIM1 定时器和外部触发的同步”）。

注意：在设置了 TIM1_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TIM1_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。预分频器的新参数在下一次更新事件到来时被采用。

图 12-2 和图 12-3 给出了在预分频器运行时，更改预分频器参数的例子。

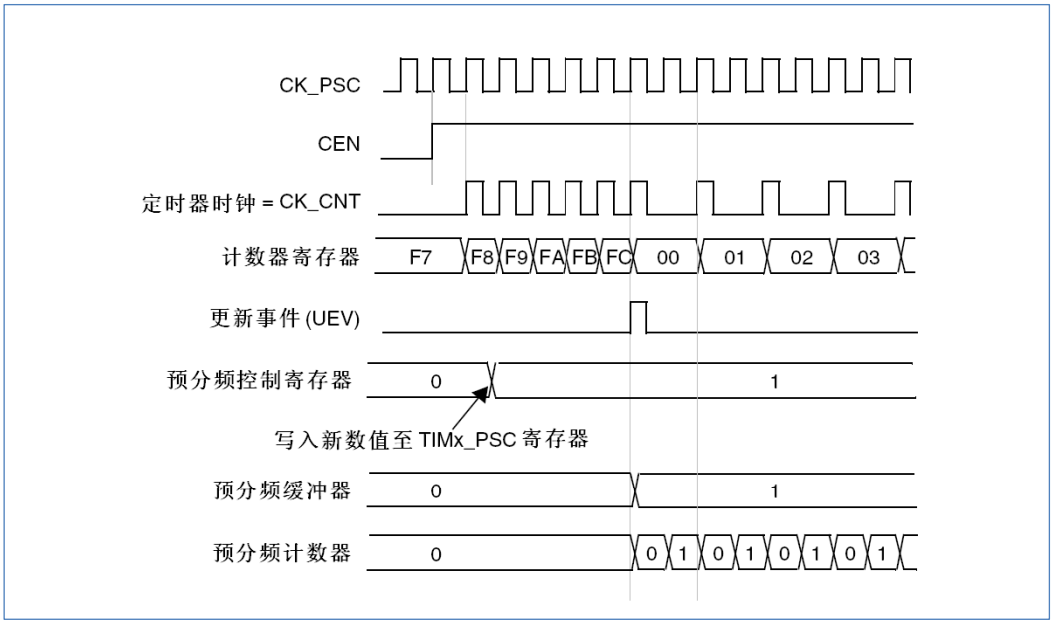

图 12-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

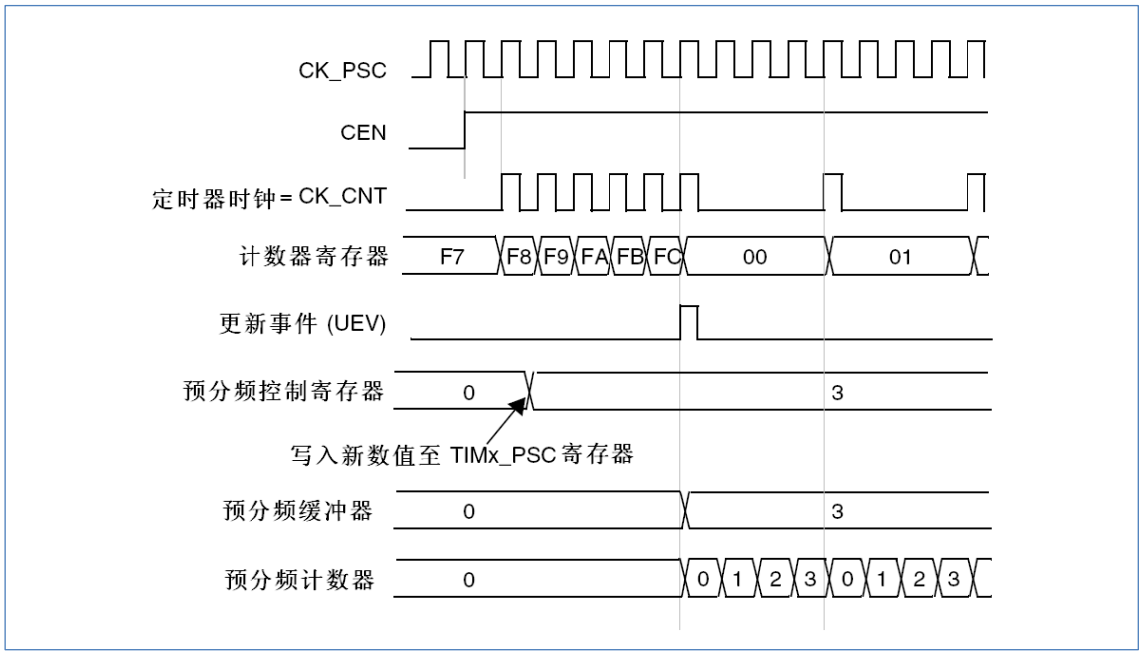

图 12-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

12.2.2 计数器模式

12.2.2.1 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（TIM1_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（TIM1_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 TIM1_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位也同样可以产生一个更新事件。

设置 TIM1_CR1 寄存器中的 UDIS 位，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零（但预分频器的数值不变）。此外，如果设置了 TIM1_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 URS 位）设置更新标志位（TIM1_SR 寄存器中的 UIF 位）。

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

下面给出一些例子，当 TIM1_ARR=0x36 时计数器在不同时钟频率下的动作。

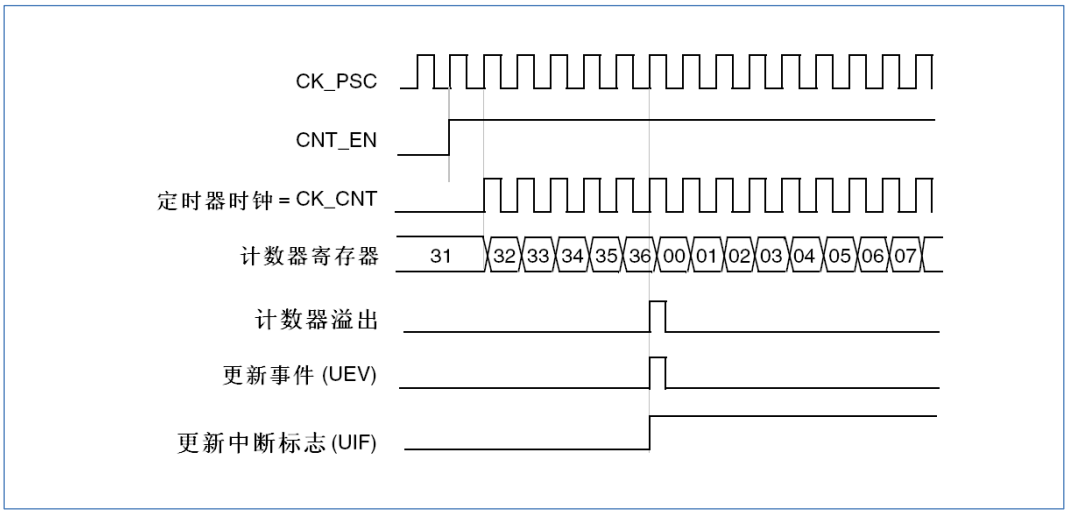

图 12-4 计数器时序图，内部时钟分频因子为 1

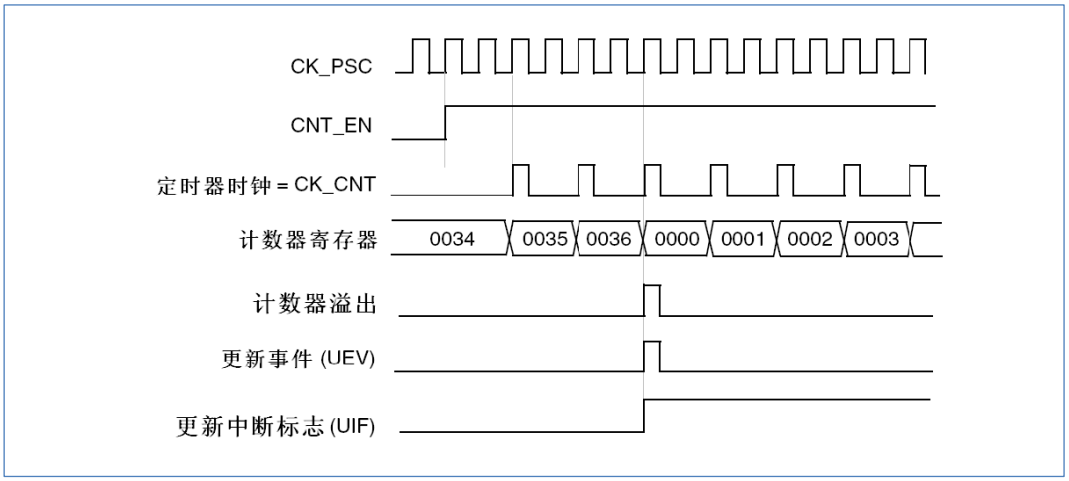

图 12-5 计数器时序图，内部时钟分频因子为 2

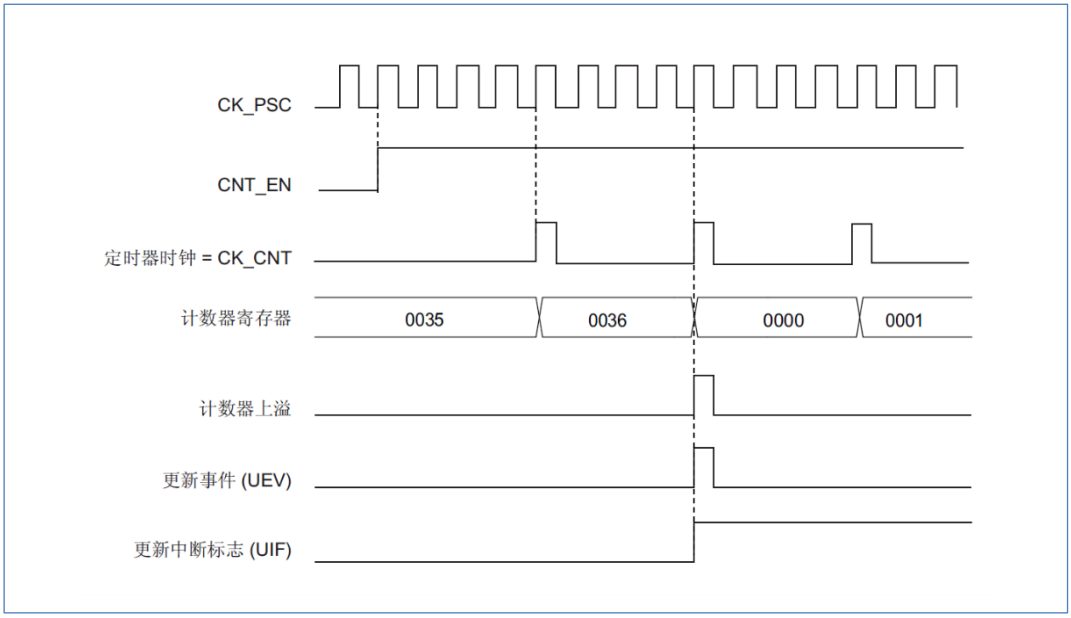

图 12-6 计数器时序图，内部时钟分频因子为 4

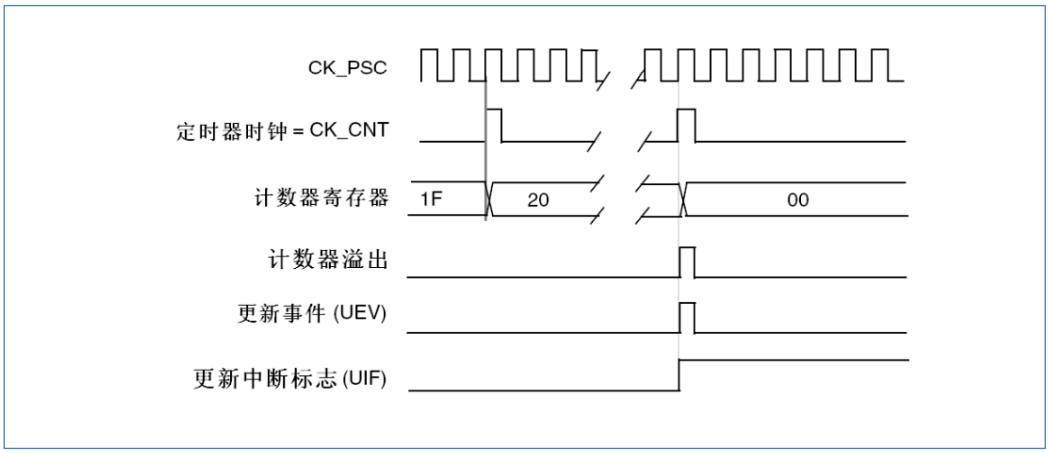

图 12-7 计数器时序图，内部时钟分频因子为 N

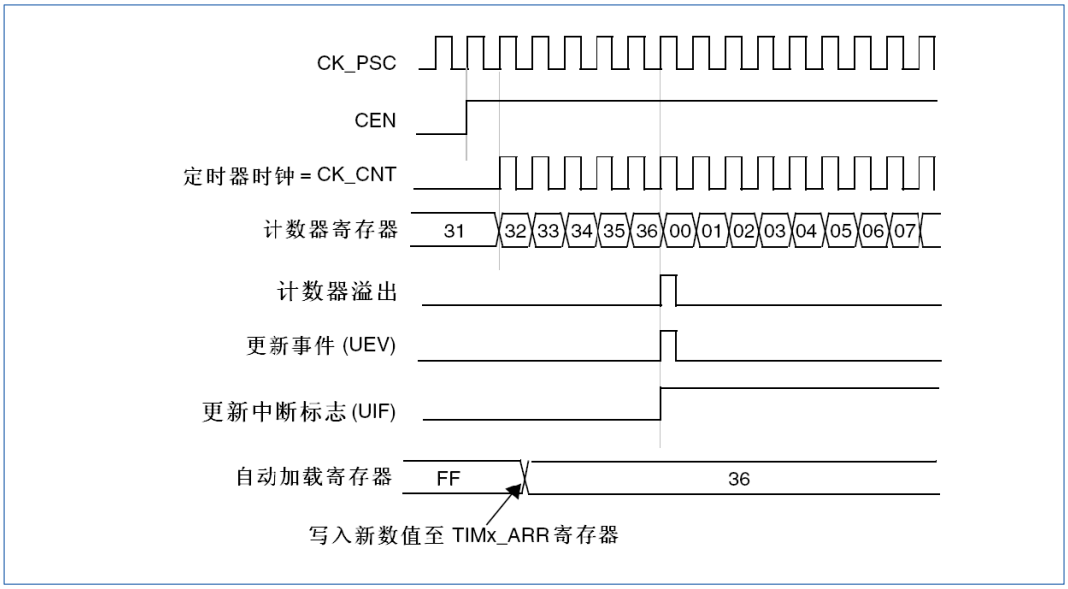

图 12-8 计数器时序图，当 ARPE=0 时的更新事件（TIM1_ARR 没有预装入）

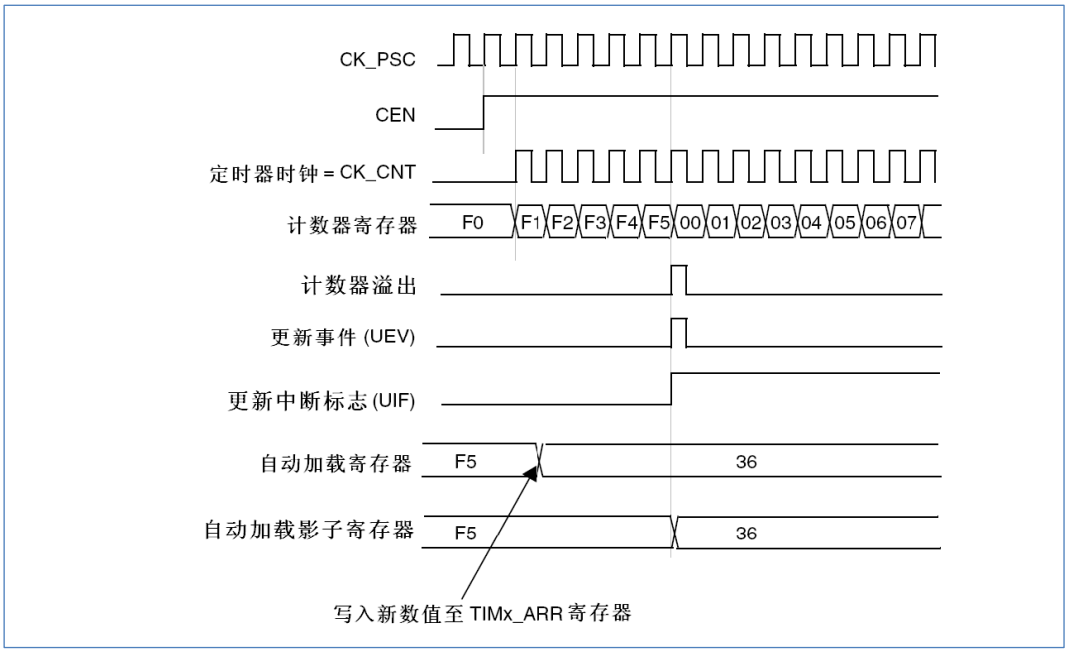

图 12-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM1_ARR）

12.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（TIM1_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（TIM1_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 TIM1_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位，也同样可以产生一个更新事件。

设置 TIM1_CR1 寄存器的 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 TIM1_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），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（TIM1_SR 寄存器中的 UIF 位）也被设置。

- 重复计数器被重置为 TIM1_RCR 寄存器中的内容。
- 预分频器的缓存器被加载为预装载的值（TIM1_PSC 寄存器的值）。
- 当前的自动加载寄存器被更新为预装载值（TIM1_ARR 寄存器中的内容）。

说明：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 TIM1_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

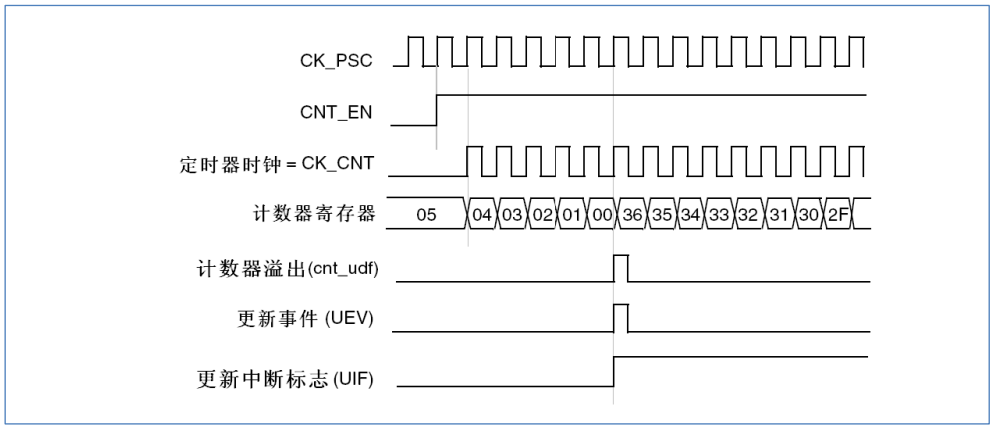

图 12-10 计数器时序图，内部时钟分频因子为 1

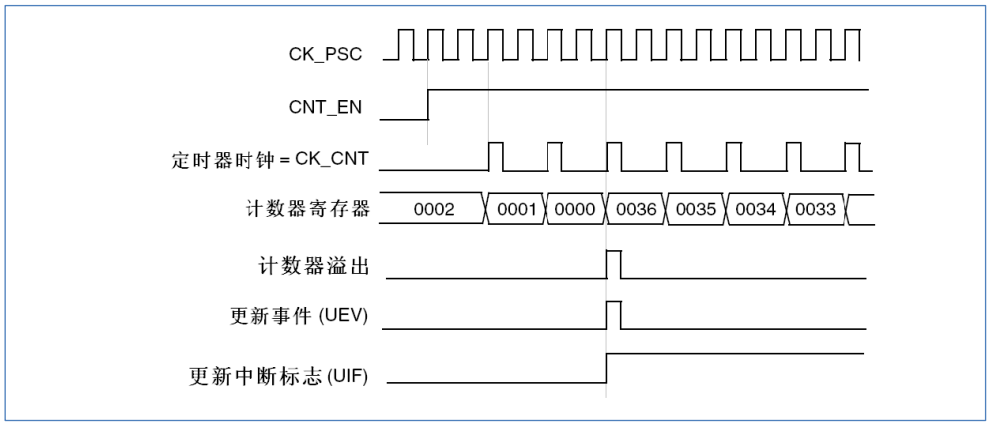

图 12-11 计数器时序图，内部时钟分频因子为 2

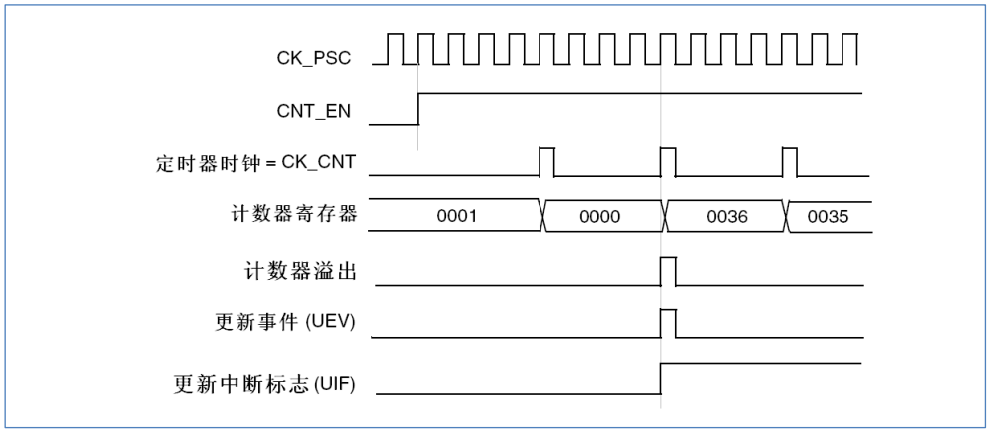

图 12-12 计数器时序图，内部时钟分频因子为 4

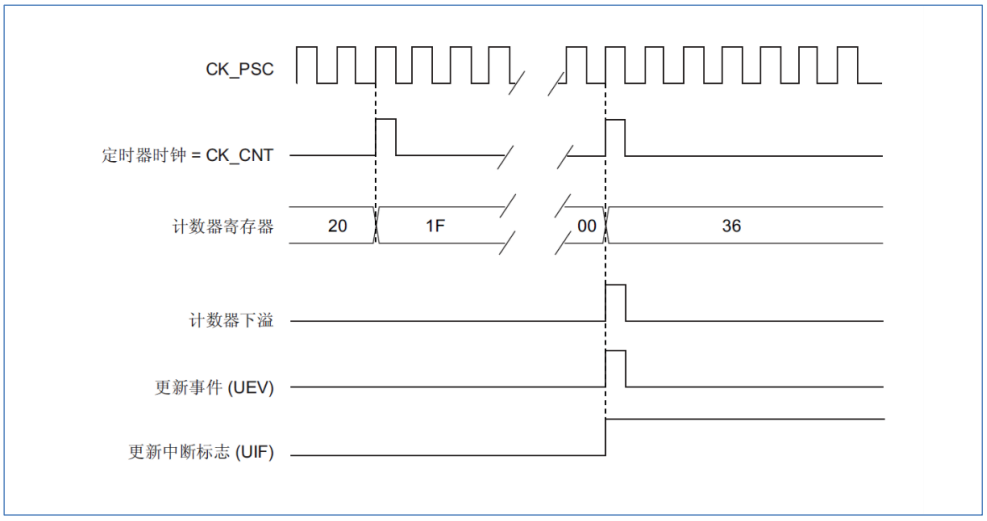

图 12-13 计数器时序图，内部时钟分频因子为 N

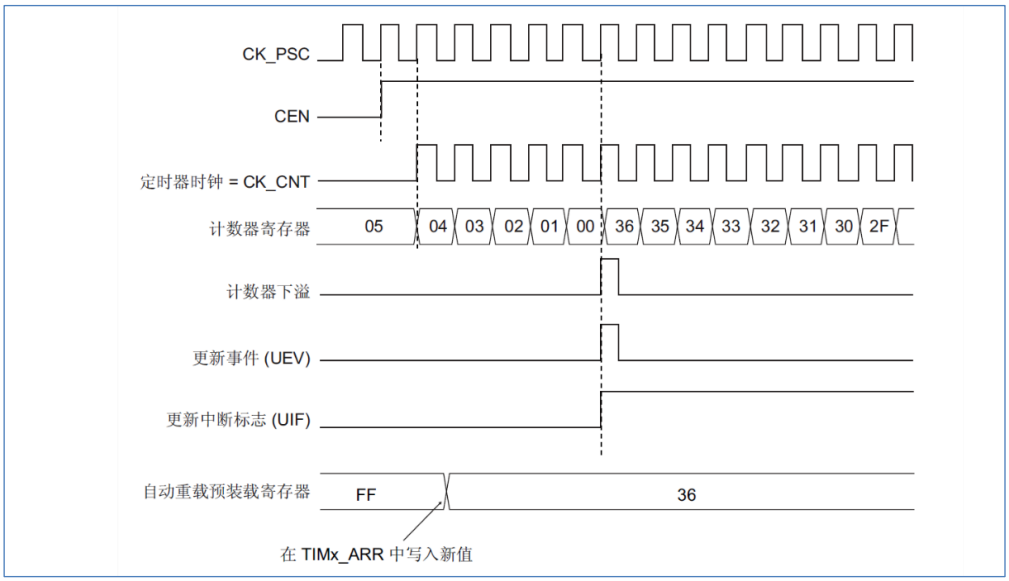

图 12-14 计数器时序图，当没有使用重复计数器时的更新事件

12.2.2.3 中央对齐模式（向上/向下计数）

在中央对齐模式，计数器从 0 开始计数到自动重载值（TIM1_ARR 寄存器的值）减 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 TIM1_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过（软件或者使用从模式控制器）设置 TIM1_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIM1_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIM1_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），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（TIM1_SR 寄存器中的 UIF 位）也被设置。

- 重复计数器被重置为 TIM1_RCR 寄存器中的内容。
- 预分频器的缓存器被加载为预装载（TIM1_PSC 寄存器）的值。
- 当前的自动加载寄存器被更新为预装载值（TIM1_ARR 寄存器中的内容）。

说明：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不同时钟频率下的操作的例子：

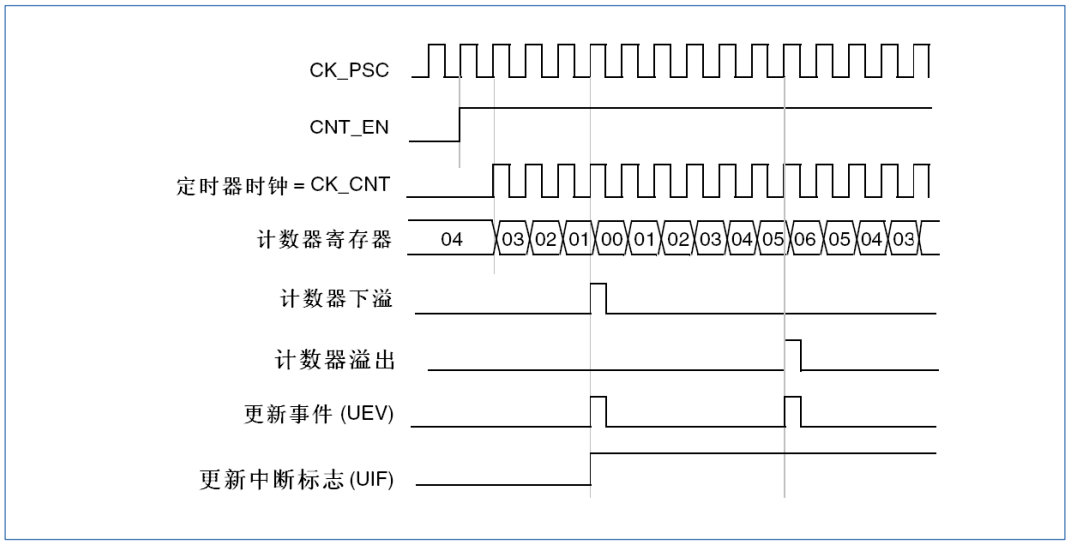

图 12-15 计数器时序图，内部时钟分频因子为 1，TIM1_ARR=0x6（这里使用了中央对齐模式 1）

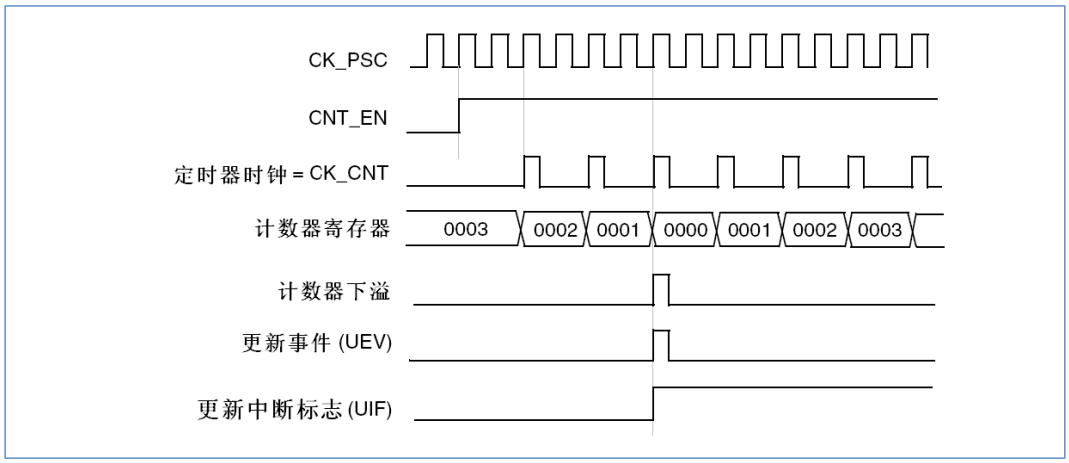

图 12-16 计数器时序图，内部时钟分频因子为 2（这里使用了中央对齐模式 1）

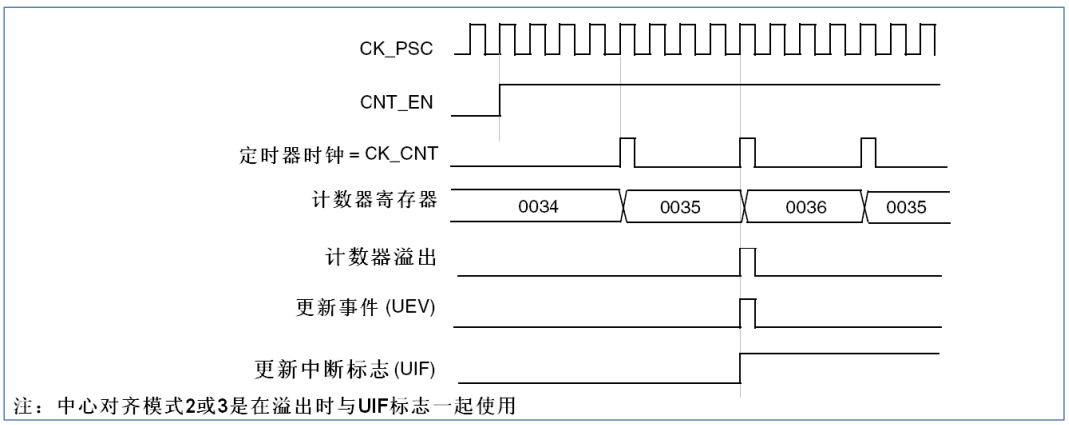

图 12-17 计数器时序图，内部时钟分频因子为 4，TIM1_ARR=0x36（这里使用了中央对齐模式 2 或 3）

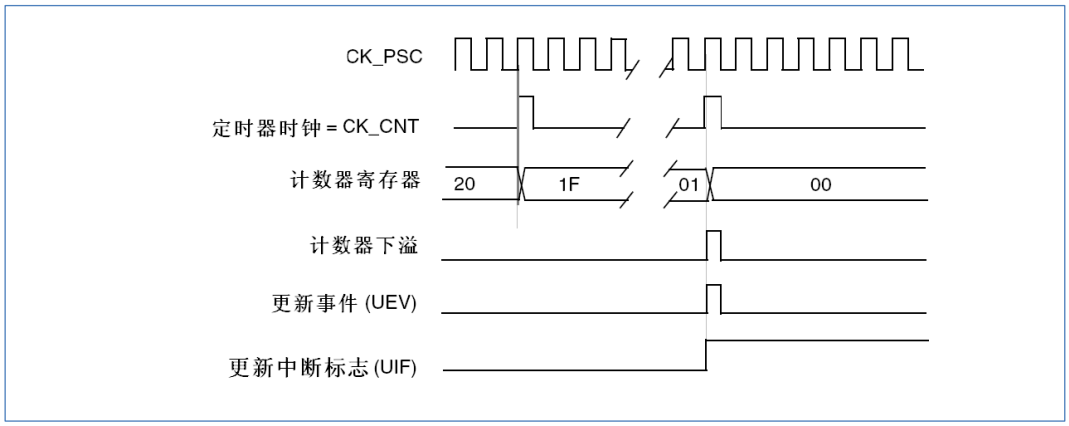

图 12-18 计数器时序图，内部时钟分频因子为 N

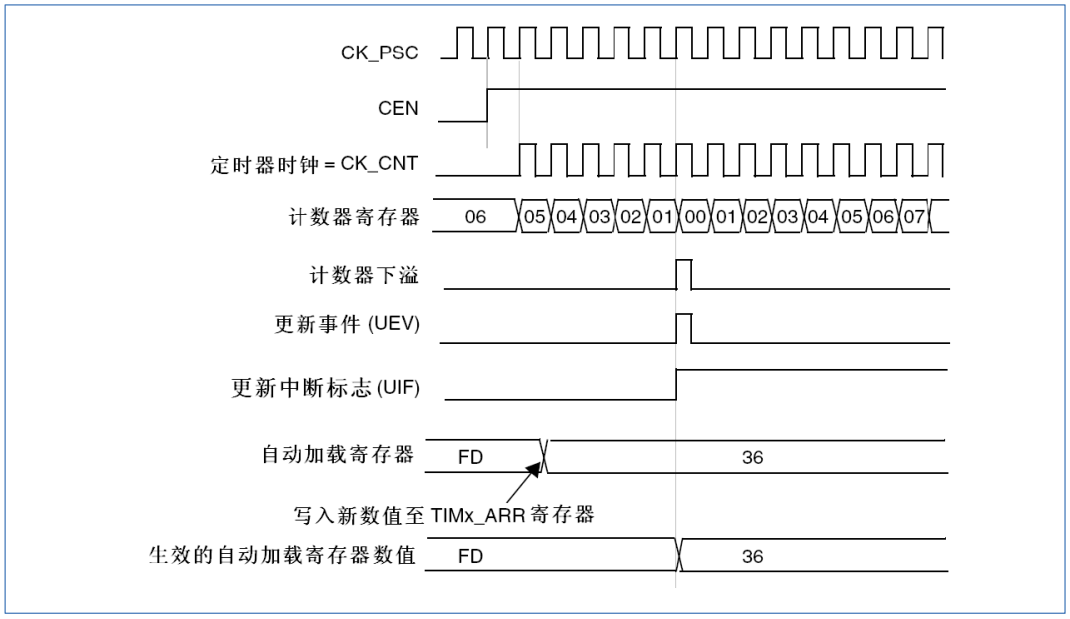

图 12-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

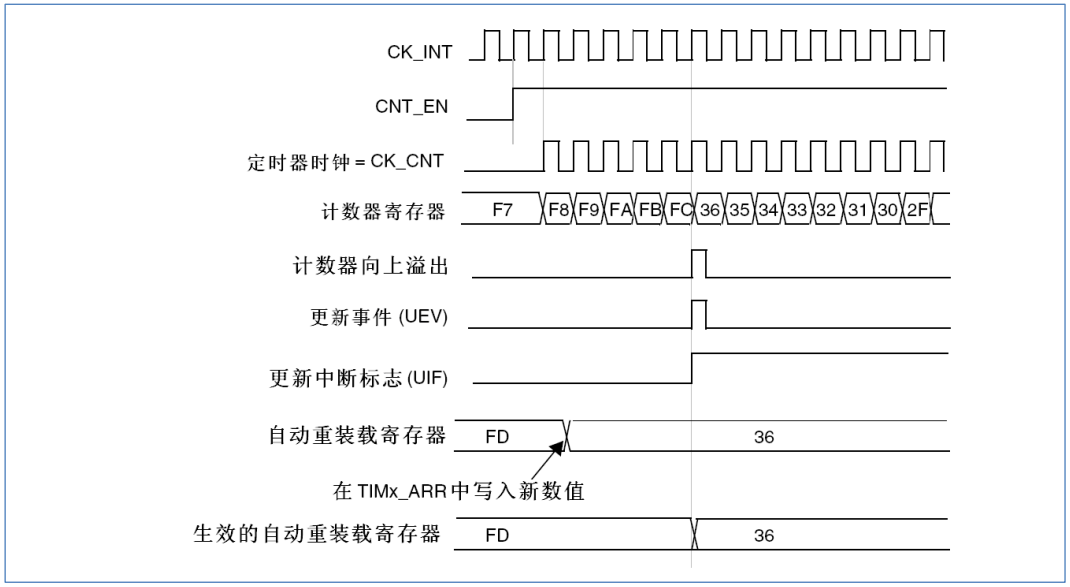

图 12-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

12.2.3 重复计数器

“时基单元”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（TIM1_ARR 自动重载入寄存器，TIM1_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIM1_CCRx），N 是 TIM1_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 的最大循环周期为 128，但它能够在每个 PWM 周期 2 次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 2 x T_{ck}。

重复计数器是自动加载的，重复速率是由 TIM1_RCR 寄存器的值定义（参见图 12-21）。当更新事件由软件产生（通过设置 TIM1_EGR 中的 UG 位）或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIM1_RCR 寄存器中的内容被重载入到重复计数器。

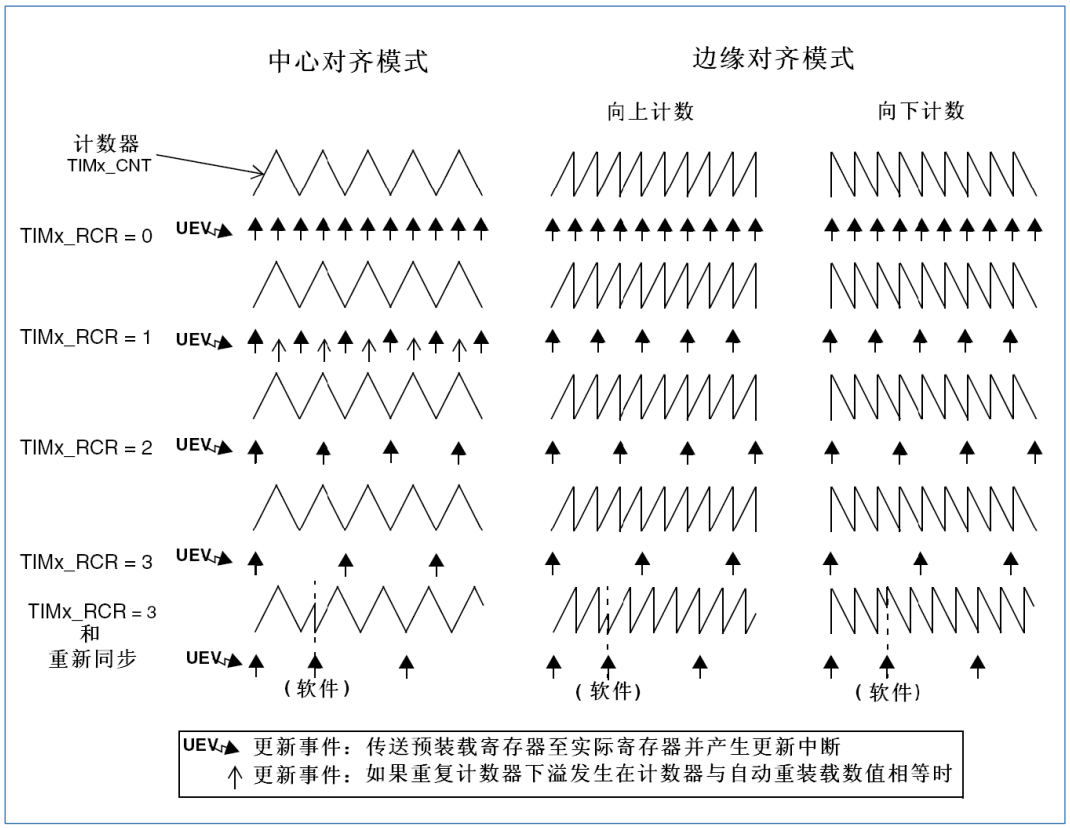

图 12-21 不同模式下更新速率的例子，及 TIM1_RCR 的寄存器设置

12.2.4 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（TIx）
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 作为另一个定时器 Timer2 的预分频器。详细信息，参见“13.2.15.1 使用一个定时器作为另一个定时器的预分频器”。

内部时钟源（CK_INT）

如果禁止了从模式控制器（SMS=000），则 CEN、DIR（TIM1_CR1 寄存器）和 UG 位（TIM1_EGR 寄存器）是实际的控制位，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。

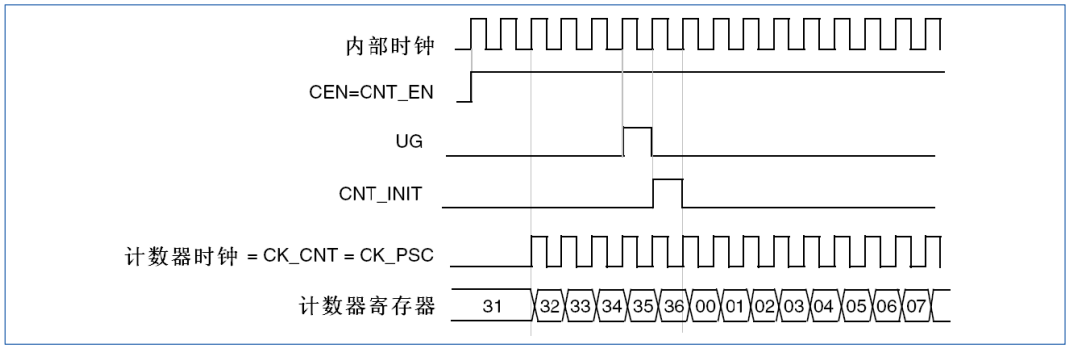

图 12-22 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

当 TIM1_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

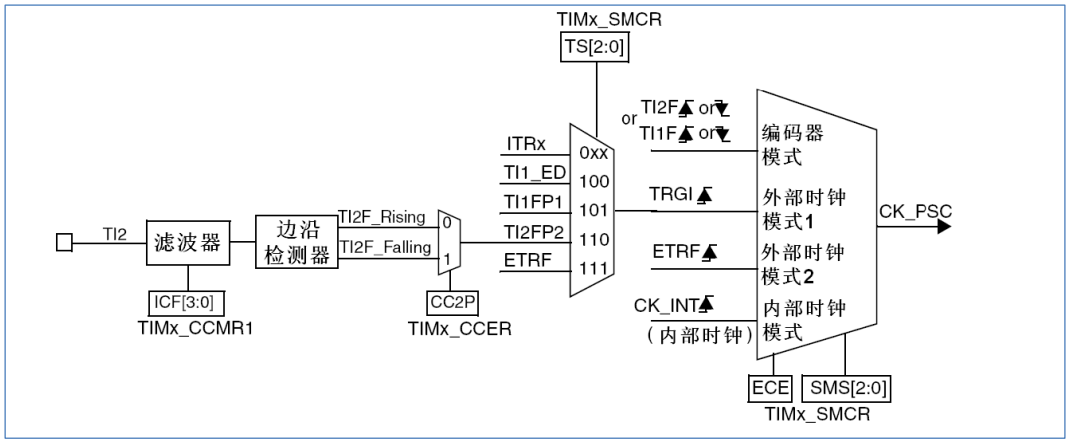

图 12-23 TI2 外部时钟连接例子

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 配置 TIM1_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TIM1_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 TIM1_CCER 寄存器的 CC2P=0，选定上升沿极性。
4. 配置 TIM1_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1。
5. 配置 TIM1_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源。
6. 设置 TIM1_CR1 寄存器的 CEN=1，启动计数器。

说明：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。

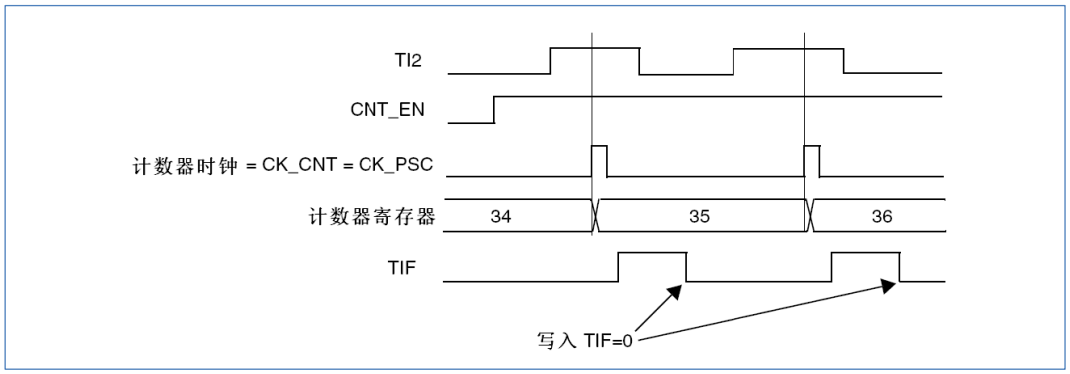

图 12-24 外部时钟模式 1 下的控制电路

外部时钟源模式 2

选定此模式的方法为：令 TIM1_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。

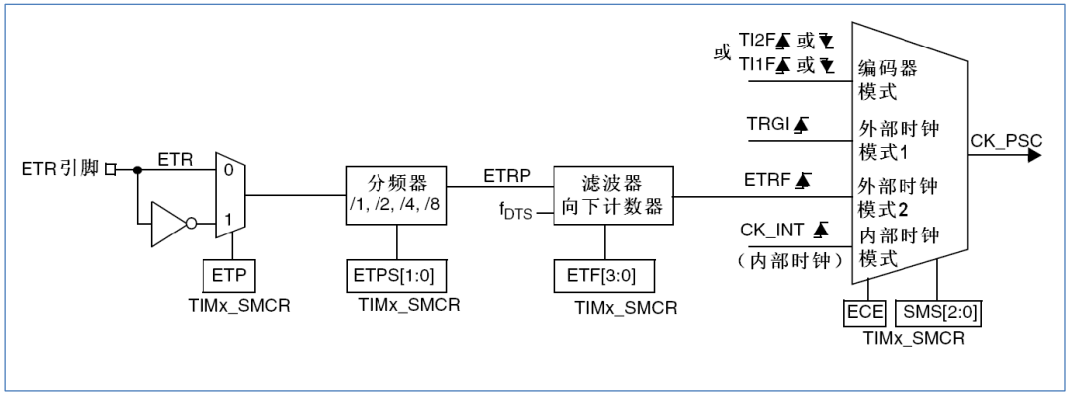

图 12-25 外部触发输入框图

例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：

1. 置 TIM1_SMCR 寄存器中的 ETF[3:0]=0000。
2. 设置预分频器，置 TIM1_SMCR 寄存器中的 ETPS[1:0]=01。
3. 选择 ETR 的上升沿检测，置 TIM1_SMCR 寄存器中的 ETP=0。
4. 开启外部时钟模式 2，写 TIM1_SMCR 寄存器中的 ECE=1。
5. 启动计数器，写 TIM1_CR1 寄存器中的 CEN=1。计数器在每 2 个 ETR 上升沿计数一次。

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

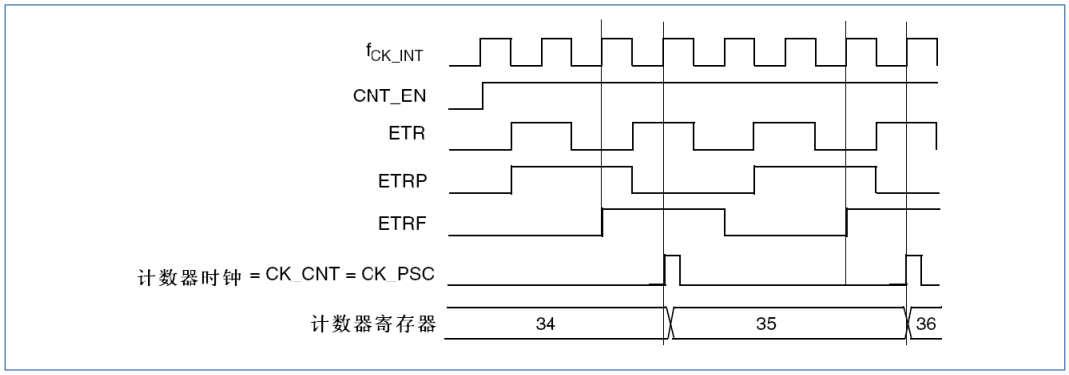

图 12-26 外部时钟模式 2 下的控制电路

12.2.5 捕获/比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器（包含影子寄存器），包括捕获的输入部分（数字滤波、多路复用和预分频器）和输出部分（比较器和输出控制）。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿监测器产生一个信号（TIxFPx），它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器（ICxPSC）。

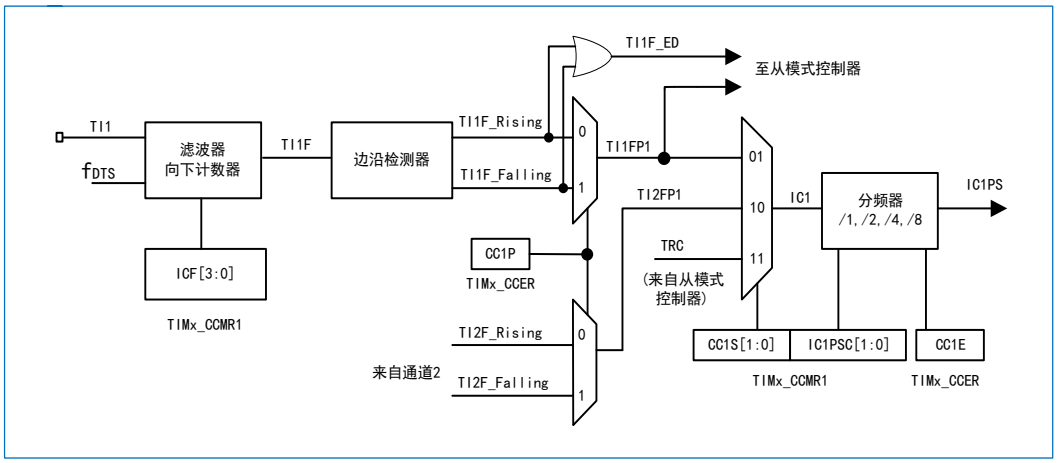

图 12-27 捕获/比较通道（如：通道 1 输入部分）

输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

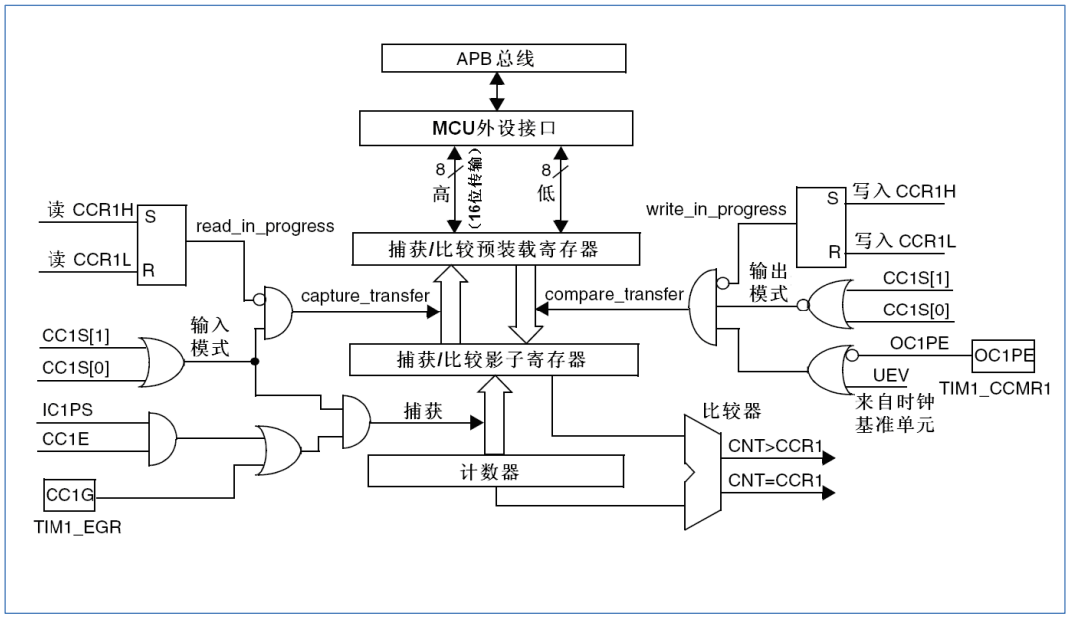

图 12-28 捕获/比较通道 1 的主电路

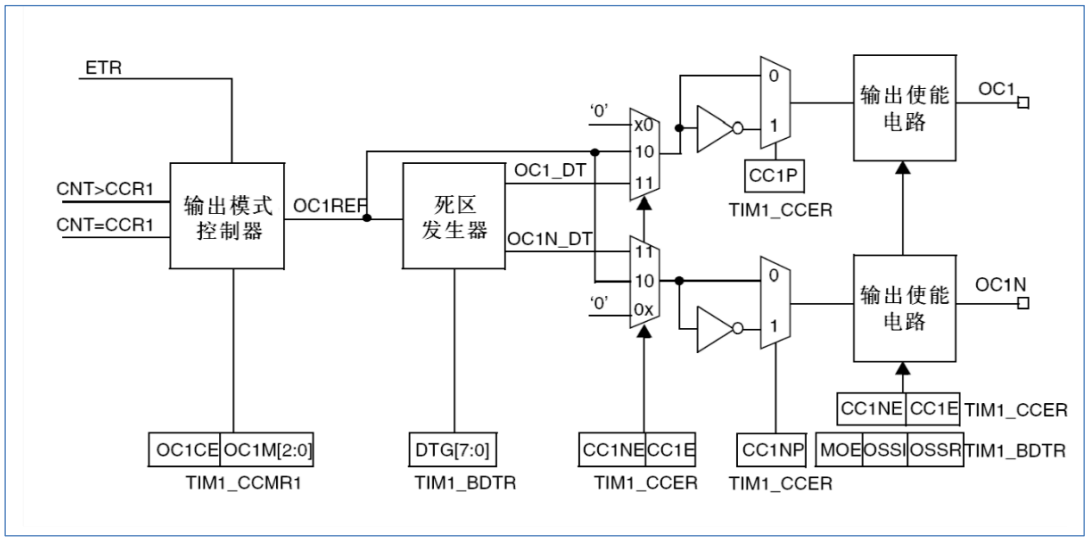

图 12-29 捕获/比较通道的输出部分（通道 1 至 3）

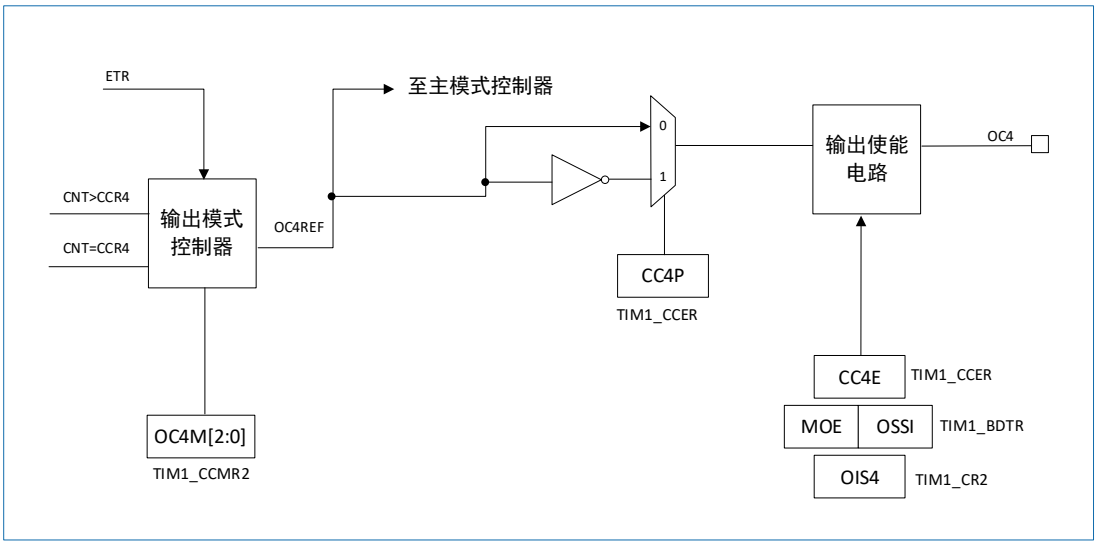

图 12-30 捕获/比较通道的输出部分（通道 4）

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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

12.2.6 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（TIM1_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIM1_SR 寄存器）被置 1，如果开放了中断，则将产生中断。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIM1_SR 寄存器）被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIM1_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIM1_CCR1 寄存器中，步骤如下：

1. 选择有效输入端：TIM1_CCR1 必须连接到 TI1 输入，所以写入 TIM1_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIM1_CCR1 寄存器变为只读。

2. 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIM1_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我

们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以 f_{DTS} 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIM1_CCMR1 寄存器中写入 IC1F=0011。

3. 选择 TI1 通道的有效转换边沿，在 TIM1_CCER 寄存器中写入 CC1P=0（上升沿）。
4. 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TIM1_CCMR1 寄存器的 IC1PSC=00）。
5. 设置 TIM1_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
6. 如果需要，通过设置 TIM1_DIER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM1_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。

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

说明：设置 TIM1_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断。

12.2.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- 两个 ICx 信号被映射至同一个 TIx 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的长度（TIM1_CCR1 寄存器）和占空比（TIM1_CCR2 寄存器），具体步骤如下（取决于 CK_INT 的频率和预分频器的值）：

1. 选择 TIM1_CCR1 的有效输入：置 TIM1_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 TIM1_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。
3. 选择 TIM1_CCR2 的有效输入：置 TIM1_CCMR1 寄存器的 CC2S=10（选中 TI1）。
4. 选择 TI1FP2 的有效极性（捕获数据到 TIM1_CCR2）：置 CC2P=1（下降沿有效）。
5. 选择有效的触发输入信号：置 TIM1_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 TIM1_SMCR 中的 SMS=100。
7. 使能捕获：置 TIM1_CCER 寄存器中 CC1E=1 且 CC2E=1。

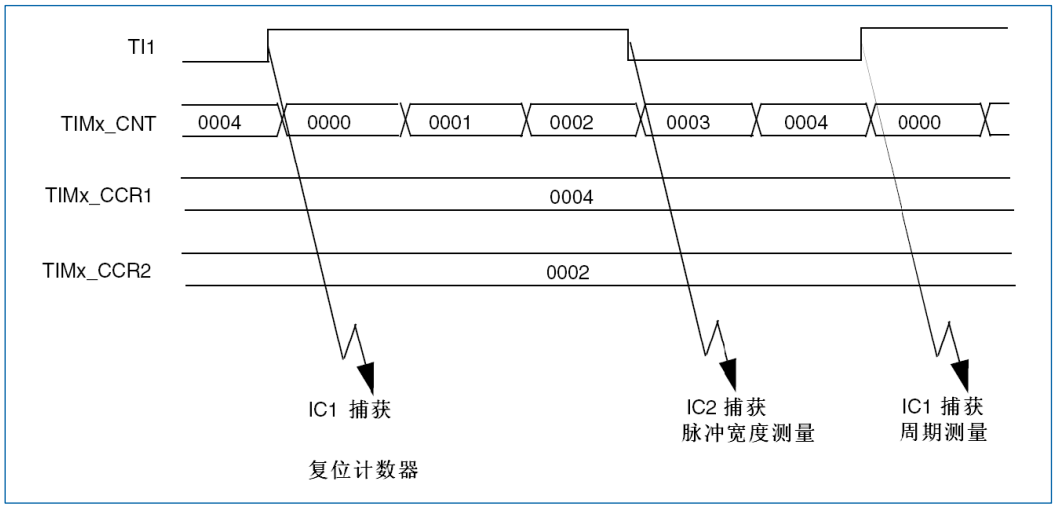

图 12-31 PWM 输入模式时序

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 TIM1_CH1/TIM1_CH2 信号。

12.2.8 强制输出模式

在输出模式（TIM1_CCMRx 寄存器中 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。置 TIM1_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0（OCx 高电平有效），则 OCx 被强置为高电平。

置 TIM1_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIM1_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断请求。这将会在下面的输出比较模式一节中介绍。

12.2.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式（TIM1_CCMRx 寄存器中的 OCxM 位）和输出极性（TIM1_CCER 寄存器中的 CCxP 位）定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平（OCxM=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（TIM1_SR 寄存器中的 CCxIF 位）。
- 若设置了相应的中断使能（TIM1_DIER 寄存器中的 CCxIE 位），则产生一个中断。

TIM1_CCMRx 中的 OCxPE 位选择 TIM1_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 将相应的数据写入 TIM1_ARR 和 TIM1_CCRx 寄存器中。

3. 如果要产生一个中断请求，设置 CCxIE 位。

4. 选择输出模式，例如：

   A. 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011。

   B. 置 OCxPE=0 禁用预装载寄存器。

   C. 置 CCxP=0 选择极性为高电平有效。

   D. 置 CCxE=1 使能输出。

5. 设置 TIM1_CR1 寄存器的 CEN 位启动计数器。

TIM1_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器（OCxPE=‘0’，否则 TIM1_CCRx 的影子寄存器只能在发生下一次更新事件时被更新）。下图给出了一个例子。

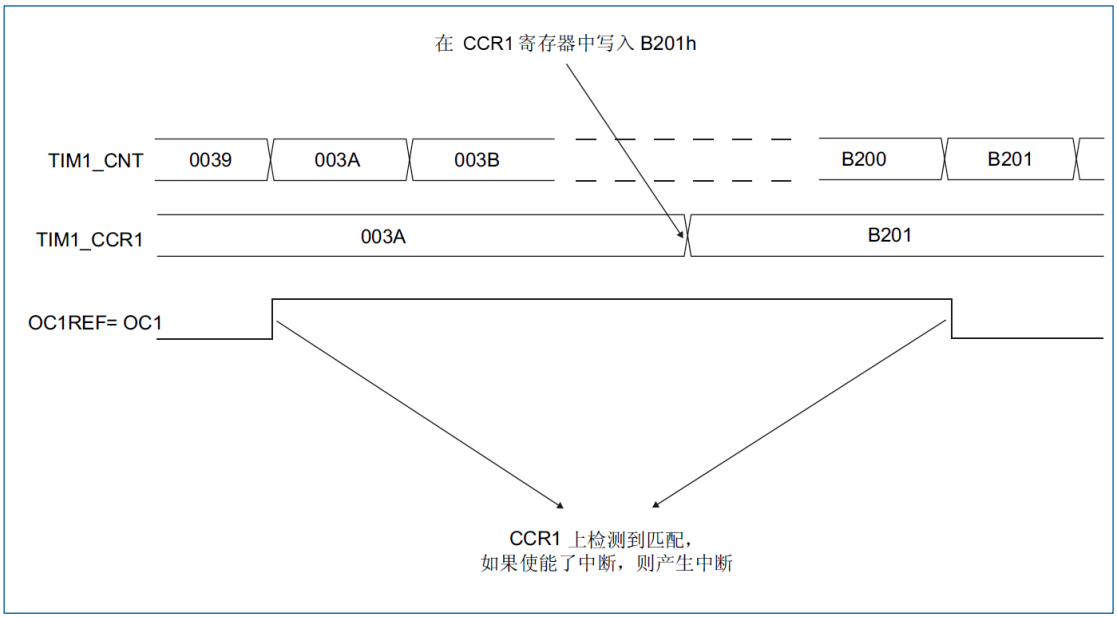

图 12-32 输出比较模式，翻转 OC1

12.2.10 PWM 模式

脉冲宽度调制模式可以产生一个由 TIM1_ARR 寄存器确定频率、由 TIM1_CCRx 寄存器确定占空比的信号。

在 TIM1_CCMRx 寄存器中的 OCxM 位写入’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 TIM1_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIM1_CR1 寄存器的 ARPE 位，（在向上计数或中央对齐模式中）使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIM1_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIM1_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过 CCxE、CCxNE、MOE、OSSI 和 OSSR 位（TIM1_CCER 和 TIM1_BDTR 寄存器中）的组合控制。参见 TIM1 捕捉/比较使能寄存器（TIM1_CCER）的描述。

在 PWM 模式（模式 1 或模式 2）下，TIM1_CNT 和 TIM1_CCRx 始终在进行比较，（依据计数器的计数方向）以确定是否符合 TIM1_CCRx≤TIM1_CNT 或者 TIM1_CNT≤TIM1_CCRx。

根据 TIM1_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- 向上计数配置

当 TIM1_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

下面是一个 PWM 模式 1 的例子。当 TIM1_CNT<TIM1_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM1_CCRx 中的比较值大于自动重装载值（TIM1_ARR），则 OCxREF 保持为'1'。如果比较值为 0，则 OCxREF 保持为'0'。下图为 TIM1_ARR=8 时边沿对齐的 PWM 波形实例。

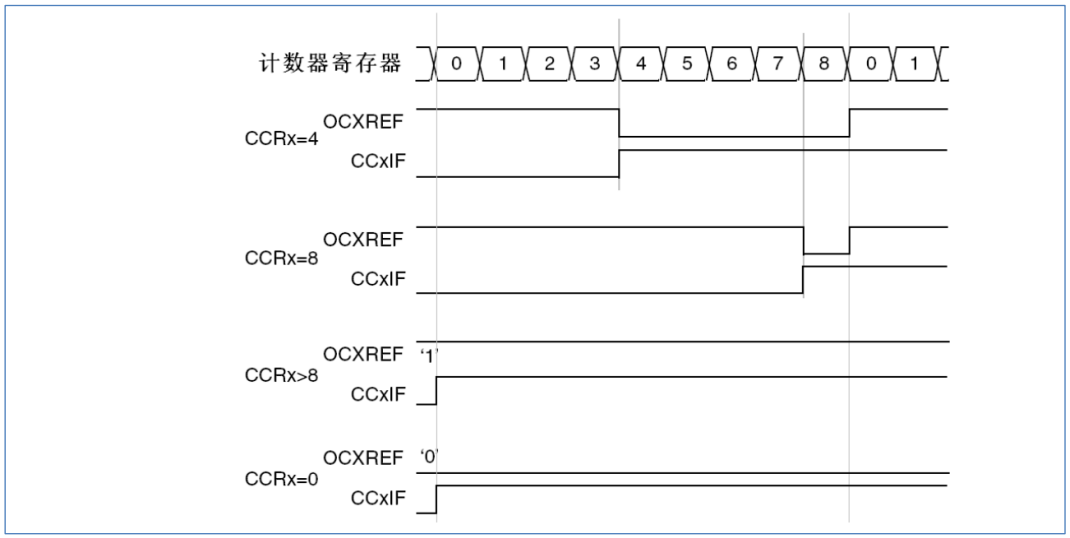

图 12-33 边沿对齐的 PWM 波形（ARR=8）

- 向下计数的配置

当 TIM1_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIM1_CNT>TIM1_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIM1_CCRx 中的比较值大于 TIM1_ARR 中的自动重装载值，则 OCxREF 保持为'1'。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 TIM1_CR1 寄存器中的 CMS 位不为'00'时，为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIM1_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。

下图给出了一些中央对齐的 PWM 波形的例子。

- TIM1_ARR=8
- PWM 模式 1
- TIM1_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

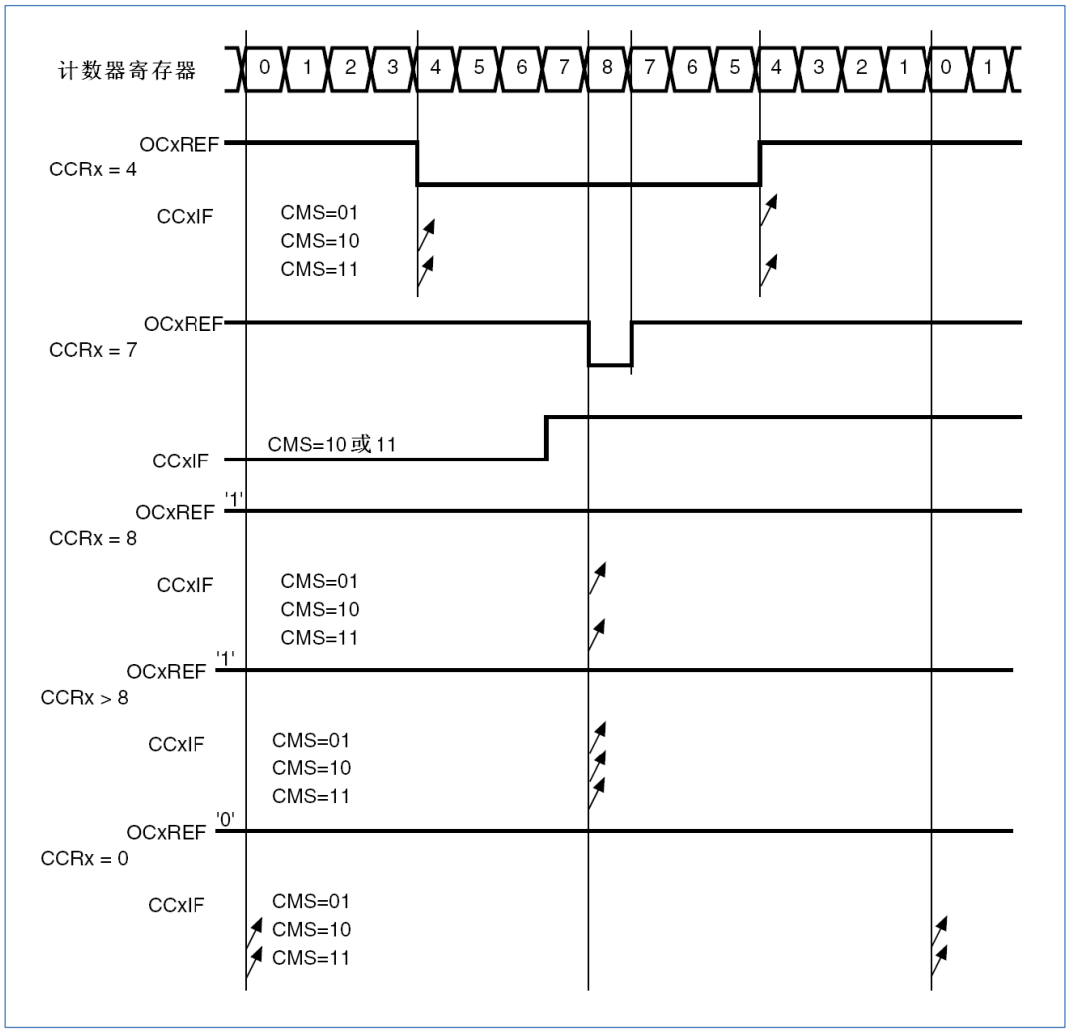

图 12-34 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIM1_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中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（TIM1_CNT>TIM1_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 TIM1_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置 TIM1_EGR 位中的 UG 位），并且不要在计数进行过程中修改计数器的值。

12.2.11 互补输出和死区插入

高级控制定时器（TIM1）能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。

这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

配置 TIM1_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性（主输出 OCx 或互补输出 OCxN）。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIM1_CCER 寄存器的 CCxE 和 CCxNE 位，TIM1_BDTR 和 TIM1_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，参见表 12-4。

特别的是，在转换到 IDLE 状态时（MOE 下降到 0）死区被激活。

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

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

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

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

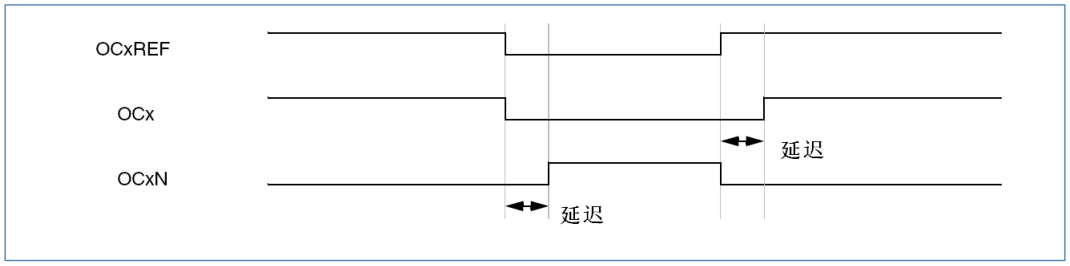

图 12-35 带死区插入的互补输出

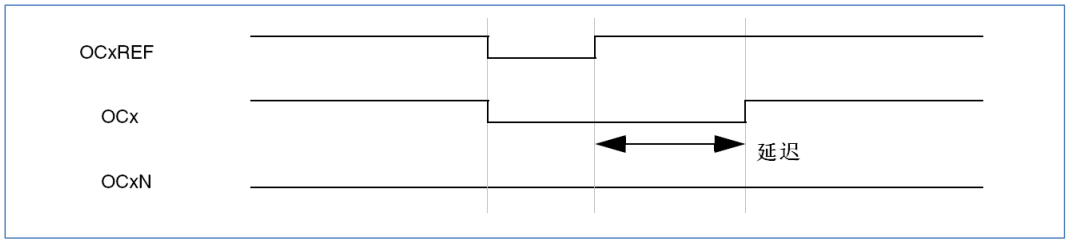

图 12-36 死区波形延迟大于负脉冲

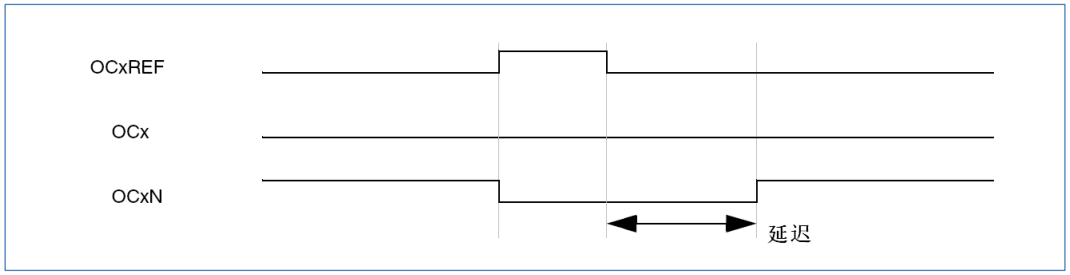

图 12-37 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIM1_BDTR 寄存器中的 DTG 位编程配置。参见 TIM1 刹车和死区寄存器（TIM1_BDTR）中的延时计算。

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 TIM1_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。

说明：当只使能 OCxN（CCxE=0、CCxNE=1）时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时（CCxE=CCxNE=1），当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

12.2.12 使用刹车功能

当使用刹车功能时，依据相应的控制位（TIM1_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIM1_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，参见表 12-4。

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIM1_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的，在实际信号（作用在输出端）和同步控制位（在 TIM1_BDTR 寄存器中）之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时（空指令）才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

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 TIM1_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。*注意：因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。*
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIM1_DIER 寄存器中的 BIE 位，当刹车状态标志（TIM1_SR 寄存器中的 BIF 位）为’1’时，则产生一个中断。
- 如果设置了 TIM1_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

说明：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 BKIN 输入产生，它的有效极性是可编程的，且由 TIM1_BDTR 寄存器中的 BKE 位开启。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性）。用户可以通过 TIM1_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 TIM1 刹车和死区寄存器（TIM1_BDTR）。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

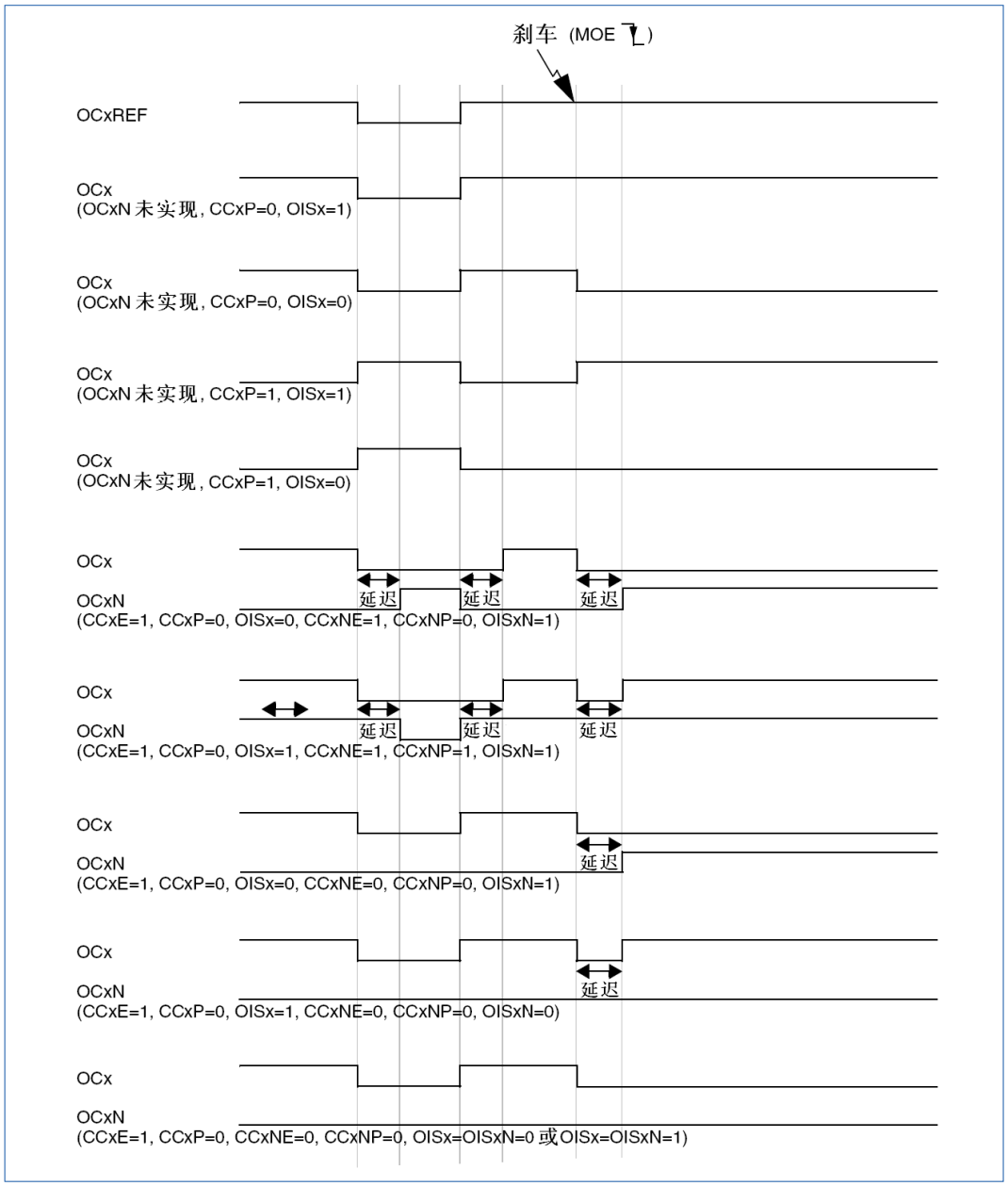

图 12-38 响应刹车的输出

12.2.13 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIM1_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 信号可以连到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

1. 外部触发预分频器必须处于关闭：TIM1_SMCR 寄存器中的 ETPS[1:0]=00。
2. 必须禁止外部时钟模式 2：TIM1_SMCR 寄存器中的 ECE=0。
3. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作。在这个例子中，定时器 TIM1 被置于 PWM 模式。

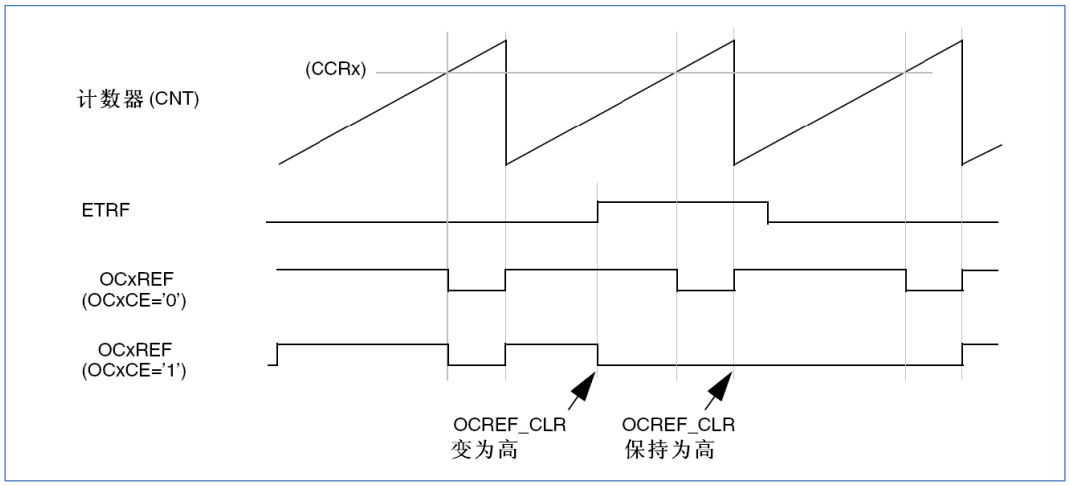

图 12-39 清除 TIM1 的 OCxREF

12.2.14 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。这样用户就可以预先设置好下一步骤配置，并在同一个时刻同时更改所有通道的配置。COM 可以通过设置 TIM1_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位（TIM1_SR 寄存器中的 COMIF 位），这时如果已设置了 TIM1_DIER.COMIE 位，则产生一个中断。

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

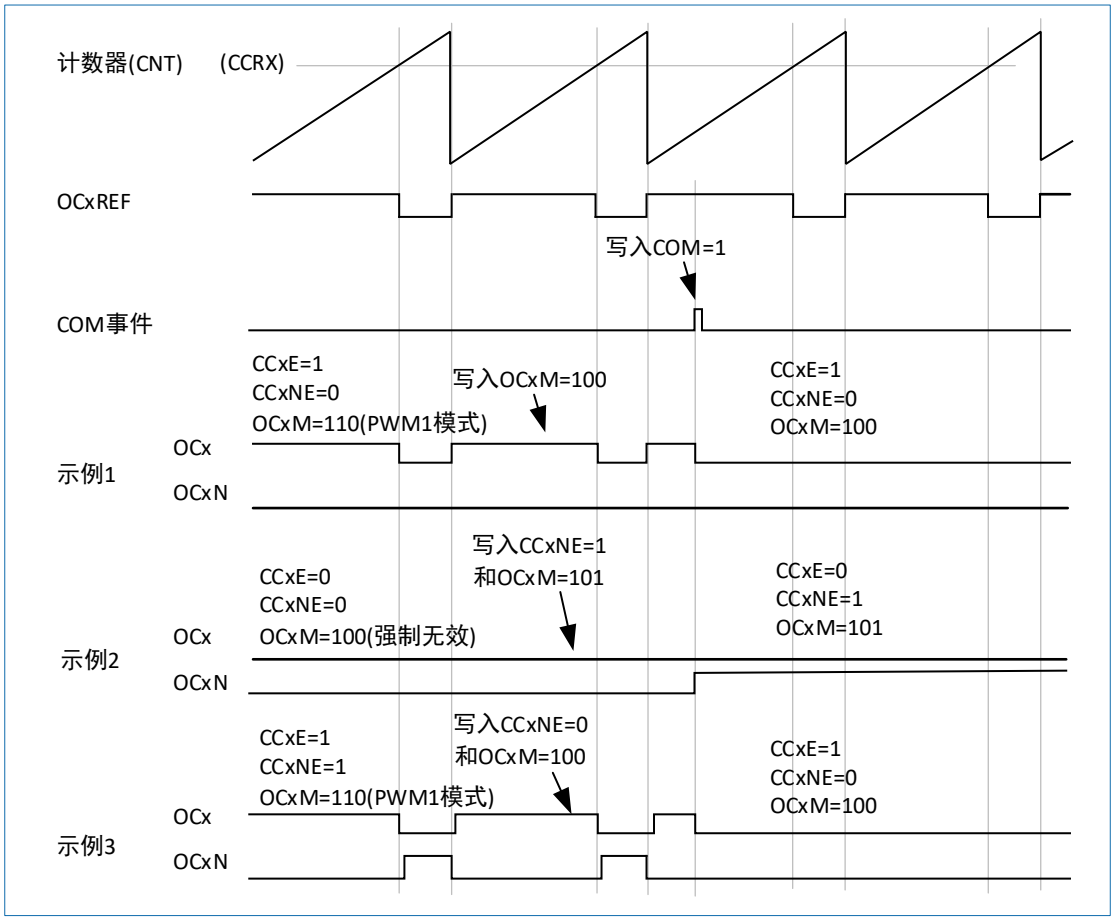

图 12-40 产生六步 PWM，使用 COM 的例子（OSSR=1）

12.2.15 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 TIM1_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

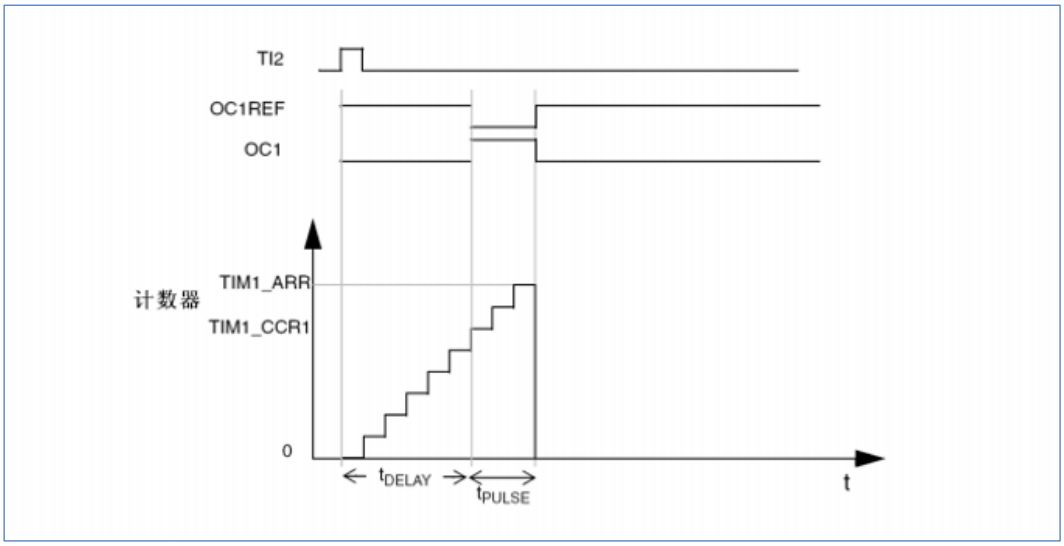

图 12-41 单脉冲模式的例子

例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

假定 TI2FP2 作为触发 1：

1. 置 TIM1_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
2. 置 TIM1_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
3. 置 TIM1_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
4. 置 TIM1_SMCR 寄存器中的 SMS=110（触发模式），TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定（要考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由 TIM1_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义（TIM1_ARR-TIM1_CCR1）。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIM1_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIM1_CCMR1 中的 OC1PE=1 和 TIM1_CR1 寄存器中的 ARPE；然后在 TIM1_CCR1 寄存器中填写比较值，在 TIM1_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIM1_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIM1_CR1 寄存器中的 OPM=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

特殊情况：OCx 快速使能：

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 t_{DELAY}。

如果要以最小延时输出波形，可以设置 TIM1_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。

12.2.16 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 TIM1_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 TIM1_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参见表 12-2，假定计数器已经启动（TIM1_CR1 寄存器中的 CEN=1），则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIM1_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端（TI1 或者 TI2）的跳变都会重新计算 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 TIM1_ARR 寄存器的自动装载值之间连续计数（根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数）。所以在开始计数之前必须配置 TIM1_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 12-2 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）	有效边沿在 TI1 上时，使用 TI1FP1 信号计数		有效边沿在 TI2 上时，使用 TI2FP2 信号计数	
		TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	TI2FP2 为高	向下计数	向上计数	不计数	不计数
	TI2FP2 为低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	TI1FP1 为高	不计数	不计数	向上计数	向下计数
	TI1FP1 为低	不计数	不计数	向下计数	向上计数
在 TI1（和 TI2）上计数	TI2FP2（和 TI1FP1）为高	向下计数	向上计数	向上计数	向下计数
	TI2FP2（和 TI1FP1）为低	向上计数	向下计数	向下计数	向上计数

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

编码器的差分输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

- CC1S=’01’（TIM1_CCMR1 寄存器，IC1FP1 映射到 TI1）
- CC2S=’01’（TIM1_CCMR2 寄存器，IC2FP2 映射到 TI2）
- CC1P=‘0’（TIM1_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1）
- CC2P=‘0’（TIM1_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2）
- SMS=’011’（TIM1_SMCR 寄存器，所有的输入均在上升沿和下降沿有效）
- CEN=‘1’（TIM1_CR1 寄存器，计数器使能）

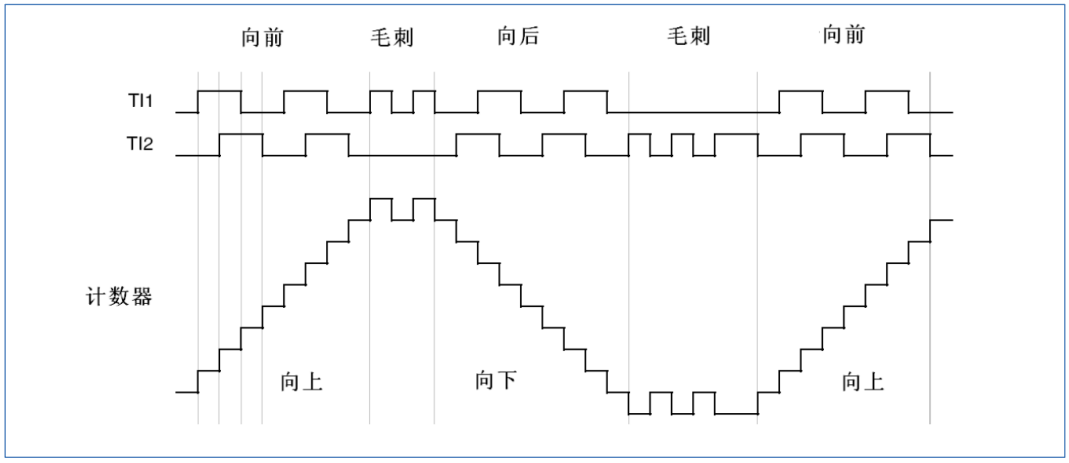

图 12-42 编码器模式下的计数器操作实例

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）。

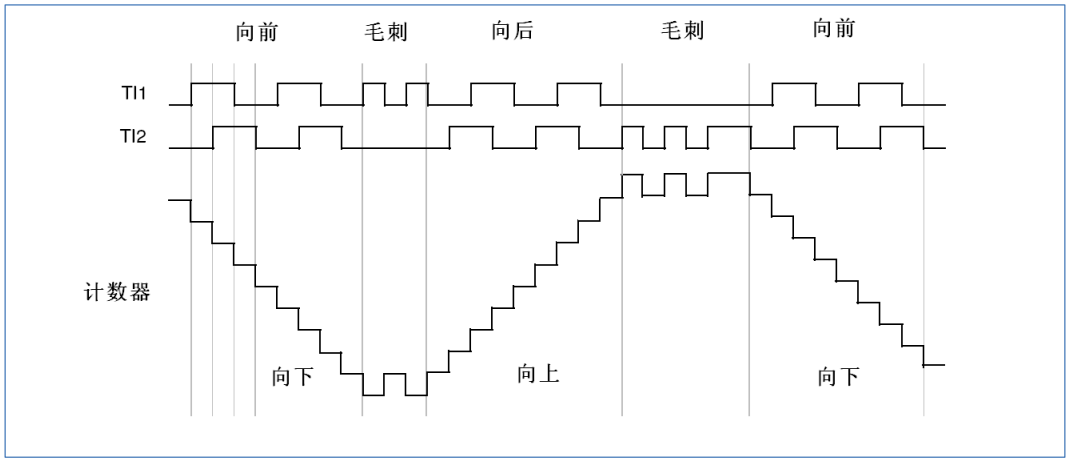

图 12-43 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前位置的信息。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，用户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）。

12.2.17 定时器输入异或功能

TIM1_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 TIM1_CH1、TIM1_CH2 和 TIM1_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。下节给出了此特性用于连接霍尔传感器的例子。

12.2.18 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 定时器 TIMx（如 TIM2）作为“接口定时器”来连接霍尔传感器，见图 12-44。TIMx 的 3 个定时器输入脚（TIMx_CH1、TIMx_CH2、TIMx_CH3）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，输入是 TI1F_ED。每当 3 个输入之一变化时，计数器重新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC（图 12-27）。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 TIM1 的 PWM 配置。

- 置 TIMx_CR2 寄存器的 TI1S 位为’1’，配置三个定时器输入逻辑或到 TI1 输入。
- 时基编程：置 TIMx_ARR 为其最大值（计数器必须通过 TI1 的变化清零）。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
- 设置通道 1 为捕获模式（选中 TRC）：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的（TIM1_CR2 寄存器中 CCPC=1），同时触发输入控制 COM 事件（TIM1_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。

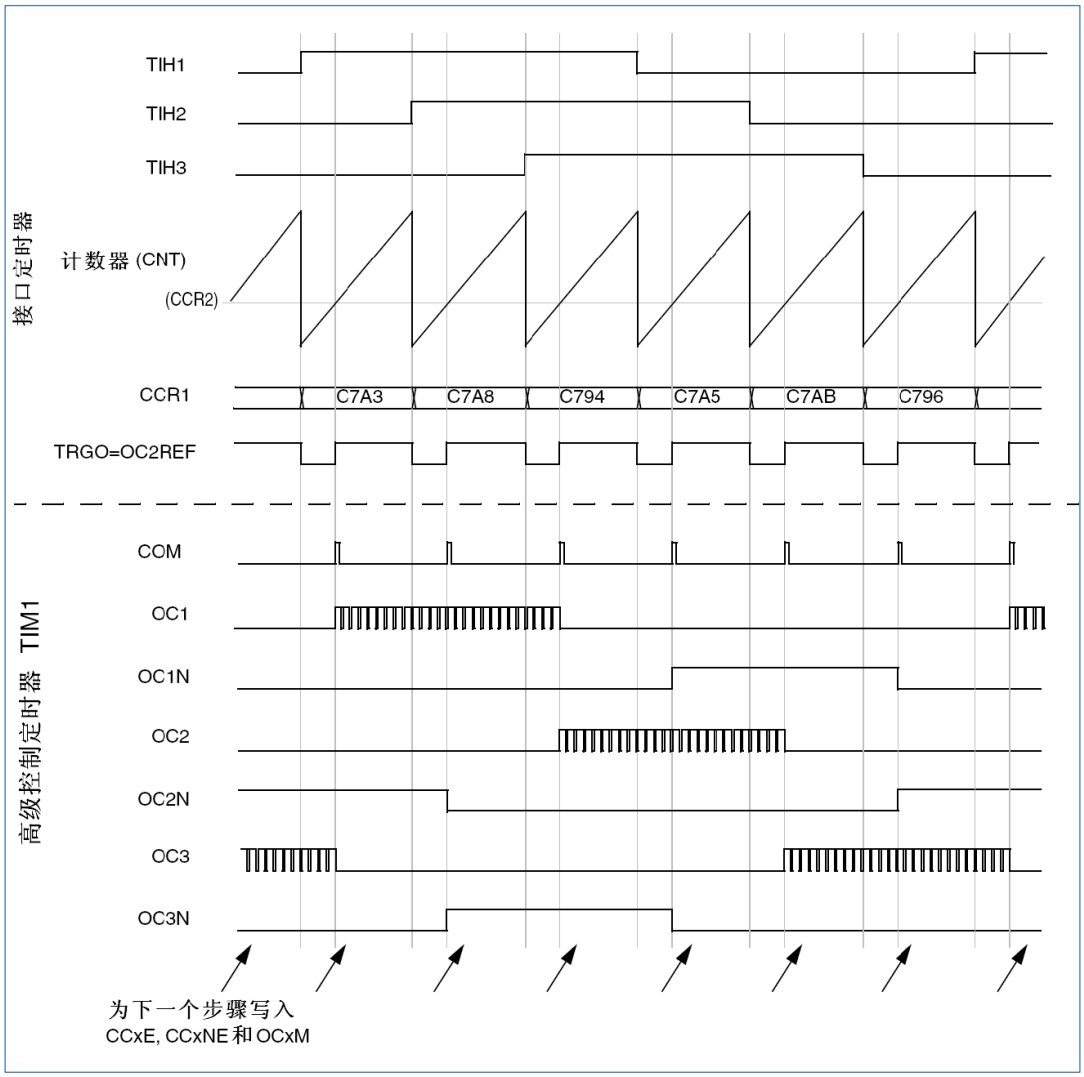

图 12-44 霍尔传感器接口的实例

12.2.19 TIM1 定时器和外部触发的同步

TIM1 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

12.2.19.1 从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIM1_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIM1_ARR，TIM1_CCRx）都被更新了。

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 TIM1_CCMR1 寄存器中 CC1S=01。置 TIM1_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。
- 置 TIM1_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM1_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM1_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIM1_SR 寄存器中的 TIF 位）被设置，根据 TIM1_DIER 寄存器中 TIE（中断使能）位的设置，产生一个中断请求。

下图显示当自动重装载寄存器 TIM1_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

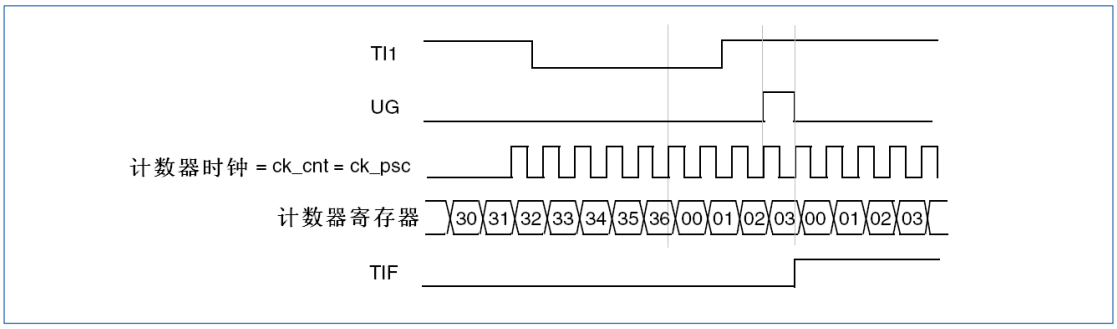

图 12-45 复位模式下的控制电路

12.2.19.2 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIM1_CCMR1 寄存器中 CC1S=01。置 TIM1_CCER 寄存器中 CC1P=1 以确定极性（只检测低电平）。
- 置 TIM1_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM1_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM1_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，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 TIM1_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。

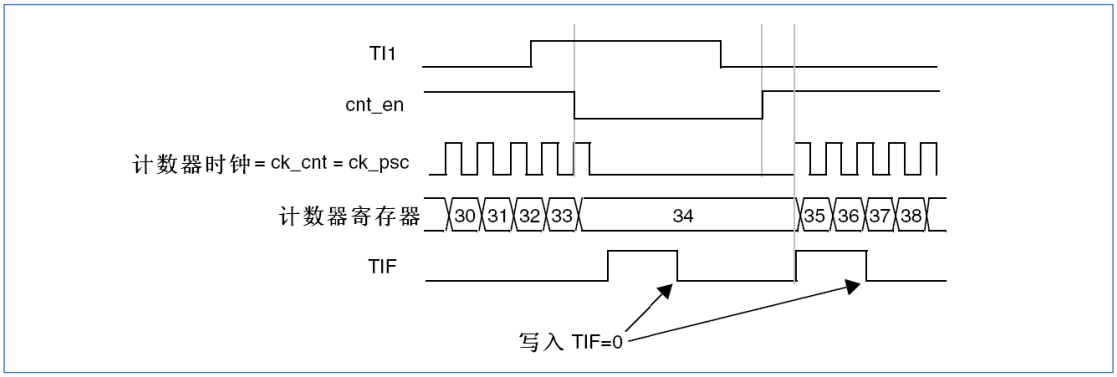

图 12-46 门控模式下的控制电路

12.2.19.3 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIM1_CCMR1 寄存器中 CC2S=01。置 TIM1_CCER 寄存器中 CC2P=1 以确定极性（只检测低电平）。

- 置 TIM1_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM1_SMCR 寄存器中 TS=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。

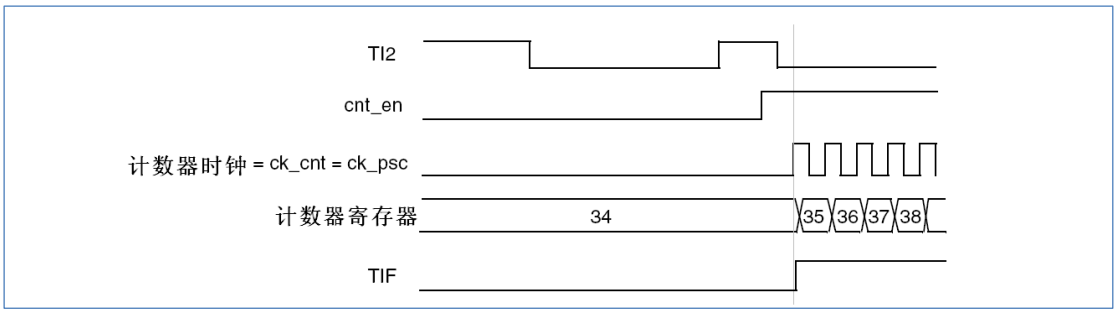

图 12-47 触发器模式下的控制电路

12.2.19.4 从模式：外部时钟模式 2＋触发模式

外部时钟模式 2 可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIM1_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1. 通过 TIM1_SMCR 寄存器配置外部触发输入电路：
   - ETF=0000：没有滤波
   - ETPS=00：不用预分频器
   - ETP=0：检测 ETR 的上升沿，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 TIM1_CCMR1 寄存器中 CC1S=01，选择输入捕获源。
   - 置 TIM1_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。
3. 置 TIM1_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIM1_SMCR 寄存器中 TS=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。

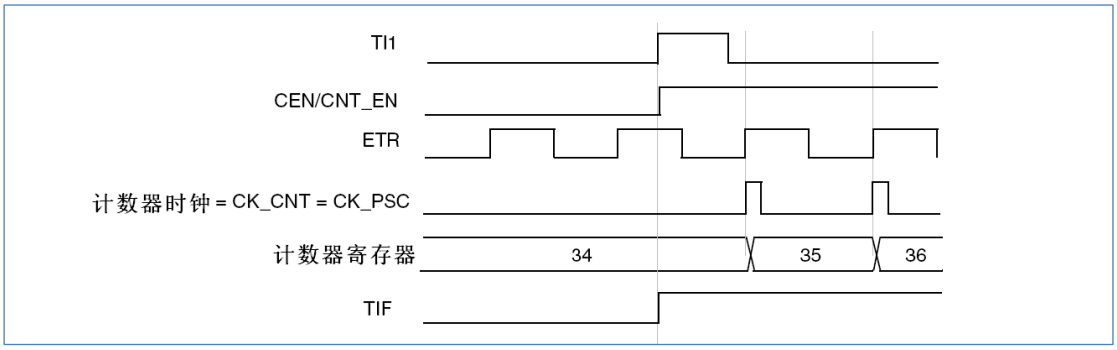

图 12-48 外部时钟模式 2＋触发模式下的控制电路

12.2.20 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。参见“13.2.15 定时器同步”。

12.2.21 调试模式

当 MCU 进入调试模式时（Cortex-M0 内核停止），根据 DBG 模块中 DBG_TIM1_STOP 的设置，TIM1 计数器可以或者继续正常操作，或者停止。参见“22.8.2 对定时器、看门狗和 I2C 的调试支持”。

12.3 TIM1 寄存器

基地址：0x4001 2C00

空间大小：0x400

12.3.1 TIM1 控制寄存器 1（TIM1_CR1）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CC4_ADC_SEL	CC3_ADC_SEL	CC2_ADC_SEL	CC1_ADC_SEL	Res											
rw	rw	rw	rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

位 31	CC4_ADC_SEL：ADC 的 CC4 触发信号的产生选择（ACD triger CC4 generation source selection） • 0：默认值；ADC 的 CC4 触发信号由端口 CH4 信号产生 • 1：ADC 的 CC4 触发信号改由 Timer 内部 OC4REF 信号产生
位 30	CC3_ADC_SEL：ADC 的 CC3 触发信号的产生选择（ACD triger CC3 generation source selection） • 0：默认值；ADC 的 CC3 触发信号由端口 CH3 信号产生 • 1：ADC 的 CC3 触发信号改由 Timer 内部 OC3REF 信号产生
位 29	CC2_ADC_SEL：ADC 的 CC2 触发信号的产生选择（ACD triger CC2 generation source selection） • 0：默认值；ADC 的 CC2 触发信号由端口 CH2 信号产生 • 1：ADC 的 CC2 触发信号改由 Timer 内部 OC2REF 信号产生
位 28	CC1_ADC_SEL：ADC 的 CC1 触发信号的产生选择（ACD triger CC1 generation source selection） • 0：默认值；ADC 的 CC1 触发信号由端口 CH1 信号产生 • 1：ADC 的 CC1 触发信号改由 Timer 内部 OC1REF 信号产生
位 27:10	Res：保留 必须保持复位值。

位 9:8	CKD[1:0]：时钟分频因子（Clock division factor） 该位域定义了定时器时钟频率（CK_INT）与死区时间、死区发生器及数字滤波器（ETR,TIx）所用的采样时钟之间的分频比例。 • 00：$t_{DTS} = t_{CK\_INT}$ • 01：$t_{DTS} = 2 \times t_{CK\_INT}$ • 10：$t_{DTS} = 4 \times t_{CK\_INT}$ • 11：保留，不使用该配置
位 7	ARPE：自动重装载预装载允许（Auto-reload preload enable） • 0：TIM1_ARR 寄存器没有缓冲 • 1：TIM1_ARR 寄存器有缓冲
位 6:5	CMS[1:0]：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、向下计数。配置为输出通道（TIM1_CCMRx 寄存器的 CCxS=00）的输出比较中断标志位，仅在计数器向下计数时设置。 • 10：中央对齐模式2 计数器交替地向上、向下计数。配置为输出通道（TIM1_CCMRx 寄存器的 CCxS=00）的输出比较中断标志，仅在计数器向上计数时设置。 • 11：中央对齐模式3 计数器交替地向上、向下计数。配置为输出通道（TIM1_CCMRx 寄存器的 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可设置。
位 4	DIR：方向（Direction） • 0：计数器向上计数 • 1：计数器向下计数
位 3	OPM：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。
位 2	URS：更新请求源（Update request source） 软件通过该位选择更新事件（UEV）的源。 • 0：如果使能了更新中断请求，则下述任一事件可产生更新中断请求： o 计数器溢出或下溢 o 设置 UG 位 o 从模式控制器产生的更新 • 1：如果使能了更新中断请求，则只有计数器溢出/下溢才产生更新中断请求。
位 1	UDIS：禁止更新（Update disable） 软件通过该位允许/禁止 UEV 的产生。

	• 0：允许 UEV 更新（UEV）事件由下述任一事件产生： o 计数器上溢/下溢 o 设置 UG 位 o 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）保持原来的值。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。
位 0	CEN：使能计数器（Update disable） • 0：禁止计数器 • 1：使能计数器

12.3.2 TIM1 控制寄存器 2（TIM1_CR2）

偏移地址：0x04

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			Res	CCUS	Res	CCPC
	rw	rw	rw	rw	rw	rw	rw	rw	rw				rw		rw

位 15	Res：保留 必须保持复位值。
位 14	OIS4：输出空闲状态4（Output Idle state 4） 参考 OIS1 位。
位 13	OIS3N：输出空闲状态3（OC3N 输出）（Output Idle state 3） 参考 OIS1N 位。
位 12	OIS3：输出空闲状态3（OC3 输出）（Output Idle state 3） 参考 OIS1 位。
位 11	OIS2N：输出空闲状态2（OC2N 输出）（Output Idle state 2） 参考 OIS1N 位。
位 10	OIS2：输出空闲状态2（OC2 输出）（Output Idle state 2） 参考 OIS1 位。
位 9	OIS1N：输出空闲状态1（OC1N 输出）（Output Idle state 1） 当 MOE=0 时，OIS1N 为： • 0：死区后 OC1N=0 • 1：死区后 OC1N=1

位 8	OIS1：输出空闲状态1（OC1 输出）（Output Idle state 1） 当 MOE=0 时，如果完成了 OC1N，OIS1 为： • 0：死区后 OC1=0 • 1：死区后 OC1=1
位 7	TI1S：TI1 选择 （TI1 selection） • 0：TIM1_CH1 引脚连到 TI1 输入。 • 1：TIM1_CH1、TIM1_CH2 和 TIM1_CH3 引脚经异或运算后连到 TI1 输入。
位 6:4	MMS[2:0]：主模式选择（Master mode selection） 该位域用于选择在主模式下送到从定时器的同步信息（TRGO）。 • 000：复位 TIM1_EGR 寄存器的 UG 位被作为触发输出（TRGO）。如果是触发输入产生的复位（模式控制器处于复位模式），则 TRGO 上的信号相对实际的复位会有延迟。 • 001：使能 计数器使能信号 CNT_EN 被作为触发输出（TRGO），用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下，计数器使能信号是 CEN 控制位和触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式。 • 010：更新 更新事件被选为触发输入（TRGO）。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 • 011：比较脉冲 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志（即使它为高），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）。
位 3	Res：保留 必须保持复位值。
位 2	CCUS：捕获/比较控制更新选择（Capture/compare control update selection） • 0：如果捕获/比较控制位是预装载的（CCPC=1），只能通过设置 COMG 位更新捕获比较寄存器。 • 1：如果捕获/比较控制位是预装载的（CCPC=1），可以通过设置 COMG 位或 TRGI 上的一个上升沿来更新捕获比较寄存器。

位 1	Res：保留 必须保持复位值。
位 0	CCPC：捕获/比较预装载控制位（Capture/compare preloaded control） • 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，只能在发生捕获/比较更新事件（COMG 设置或 TRGI 是检测到上升沿，取决于 CCUS 位）时被更新。

12.3.3 TIM1 从模式控制寄存器（TIM1_SMCR）

偏移地址：0x08

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			OCCS	SMS[2:0]		
rw	rw	rw		rw				rw	rw			rw	rw		

位 15	ETP：外部触发极性（External trigger polarity） 该位选择 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。
位 14	ECE：外部时钟使能位（External clock enable） 该位启用外部时钟模式 2。 • 0：禁止外部时钟模式 2。 • 1：使能外部时钟模式 2，计数器由 ETRF 信号上的任意有效边沿驱动。
位 13:12	ETPS[1:0]：外部触发预分频（External trigger filter） 外部触发信号 ETRP 的频率最高是 CK_INT 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/2 • 10：ETRP 频率/4 • 11：ETRP 频率/8
位 11:8	ETF[3:0]：外部触发滤波（External trigger filter） 该位域定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 • 0000：无滤波器，以 $f_{SAMPLING}=f_{DTS}$ 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8

	• 0110：采样频率 $f_{SAMPLING=}$ f_{DTS} /4，N=6 • 0111：采样频率 $f_{SAMPLING=}$ f_{DTS} /4，N=8 • 1000：采样频率 $f_{SAMPLING=}$ f_{DTS} /8，N=6 • 1001：采样频率 $f_{SAMPLING=}$ f_{DTS} /8，N=8 • 1010：采样频率 $f_{SAMPLING=}$ f_{DTS} /16，N=5 • 1011：采样频率 $f_{SAMPLING=}$ f_{DTS} /16，N=6 • 1100：采样频率 $f_{SAMPLING=}$ f_{DTS} /16，N=8 • 1101：采样频率 $f_{SAMPLING=}$ f_{DTS} /32，N=5 • 1110：采样频率 $f_{SAMPLING=}$ f_{DTS} /32，N=6 • 1111：采样频率 $f_{SAMPLING=}$ f_{DTS} /32，N=8
位 7	MSM：主/从模式（Master/slave mode） • 0：无作用 • 1：触发输入（TRGI）上的事件被延迟，以允许在当前定时器与它的从定时器间的同步（通过 TRGO）。这对于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。
位 6:4	TS[2:0]：触发选择（Trigger selection） 该位域选择同步计数器的触发输入。 • 000：内部触发 0（ITR0） • 001：内部触发 1（ITR1） • 010：内部触发 2（ITR2） • 011：保留 • 100：TI1 的边沿检测器（TI1F_ED） • 101：滤波后的定时器输入 1（TI1FP1） • 110：滤波后的定时器输入 2（TI2FP2） • 111：外部触发输入（ETRF）
位 3	OCCS：OCREF 清除选择（OCREF clear selection） 该位选择 OCREF 清除的源。 • 0：OCREF_CLR_INT 被连接到 OCREF_CLR 输入 • 1：OCREF_CLR_INT 被连接到 ETRF
位 2:0	SMS[2:0]：从模式选择（Slave mode selection） 当选择了外部信号，触发信号（TRGI）的有效边沿与选中的外部输入极性相关。 • 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/向下计数。 • 010：编码器模式 2 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/向下计数。 • 011：编码器模式 3

	根据 TI1FP1（和 TI2FP2）的电平，计数器在 TI2FP2（和 TI1FP1）的边沿向上/下计数。 • 100：复位模式 选中的触发输入信号（TRGI）的上升沿重新初始化计数器，并且触发一次更新寄存器。 • 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（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（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。

表 12-3 TIM1 内部触发连接

从定时器	ITR0（TS=000）	ITR1（TS=001）	ITR2（TS=010）
TIM1	TIM2	TIM6	ADC_AWD

12.3.4 TIM1 中断使能寄存器（TIM1_DIER）

偏移地址：0x0C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
								rw	rw	rw	rw	rw	rw	rw	rw

位 15:8	Res：保留 必须保持复位值。
位 7	BIE：刹车中断使能（Break interrupt enable） • 0：刹车中断禁用 • 1：刹车中断允许
位 6	TIE：触发中断使能（Trigger interrupt enable） • 0：触发中断禁用 • 1：触发中断允许
位 5	COMIE：COM 中断使能（COM interrupt enable） • 0：COM 中断禁用 • 1：COM 中断允许
位 4	CC4IE：捕捉/比较 4 中断使能（Capture/Compare 4 interrupt enable） • 0：CC4 中断禁用 • 1：CC4 中断允许

位 3	CC3IE：捕捉/比较 3 中断使能（Capture/Compare 3 interrupt enable） • 0：CC3 中断禁用 • 1：CC3 中断允许
位 2	CC2IE：捕捉/比较 2 中断使能（Capture/Compare 2 interrupt enable） • 0：CC2 中断禁用 • 1：CC2 中断允许
位 1	CC1IE：捕捉/比较 1 中断使能（Capture/Compare 1 interrupt enable） • 0：CC1 中断禁用 • 1：CC1 中断允许
位 0	UIE：更新中断使能（Update interrupt enable） • 0：更新中断禁用 • 1：更新中断允许

12.3.5 TIM1 状态寄存器（TIM1_SR）

偏移地址：0x10

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			CC4OF	CC3OF	CC2OF	CC1OF	Res	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

位 15:13	Res：保留 必须保持复位值。
位 12	CC4OF：捕捉/比较4 重复捕捉标志（Capture/Compare 4 overcapture flag） 请参考 CC1OF 位的定义。
位 11	CC3OF：捕捉/比较3 重复捕捉标志（Capture/Compare 3 overcapture flag） 请参考 CC1OF 位的定义。
位 10	CC2OF：捕捉/比较2 重复捕捉标志（Capture/Compare 2 overcapture flag） 请参考 CC1OF 位的定义。
位 9	CC1OF：捕捉/比较1 重复捕捉标志 （Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该位由硬件置1。软件写0 可清除该位。 • 0：无重复捕获产生。 • 1：当CC1IF 的状态已经为 1，计数器的值被捕获到TIM1_CCR1 寄存器。
位 8	Res：保留

	必须保持复位值。
位 7	BIF：刹车中断标志（Break interrupt flag） 一旦刹车输入有效，由硬件对该位置 1。 如果刹车输入无效，则该位由软件清 0。 • 0：无刹车事件产生 • 1：刹车输入信号上检测到有效电平
位 6	TIF：触发器中断标志（Trigger interrupt flag） 当发生触发事件（从模式控制器处于门控模式外的其他模式时 TRGI 输入端检测到有效边沿，或者在门控模式下 TRGI 输入端检测到任一边沿）时由硬件对该位置 1。 该位由软件清 0。 • 0：无触发器事件产生 • 1：触发中断等待响应
位 5	COMIF：COM 中断标志（COM interrupt flag） 一旦产生 COM 事件（当捕获/比较控制位：CCxE、CCxNE、OCxM 被更新） 该位由硬件置 1 或软件清 0。 • 0：无COM 事件产生 • 1：COM 中断等待响应
位 4	CC4IF：捕捉/比较4 中断标志（Capture/Compare 4 interrupt flag）
位 3	CC3IF：捕捉/比较 3 中断标志 （Capture/Compare 3 interrupt flag）
位 2	CC2IF：捕捉/比较 2 中断标志（Capture/Compare 2 interrupt flag）
位 1	CC1IF：捕捉/比较1 中断标志（Capture/Compare 1 interrupt flag） • 如果通道CC1 配置为输出模式：当计数器值与比较值匹配时，该位由硬件置1；但在中央对齐模式下除外。 该位由软件清 0。 ○ 0：无匹配发生。 ○ 1：TIM1_CNT 的值与 TIM1_CCR1 的值匹配。 当 TIM1_CCR1 > TIM1_APR 时，满足以下任一条件，CC1IF 变为 1： - 在向上或向上/向下计数模式下，计数器溢出。 - 向下计数模式下，计数器溢出。 • 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。通过软件或读 TIM1_CCR1 寄存器清 0。 ○ 0：无输入捕获产生； ○ 1：计数器值被捕获至 TIM1_CCR1（ 在 IC1 上检测到与所选极性相同的边沿）。
位 0	UIF：更新中断标志（Update interrupt flag） 当产生更新事件时，该位由硬件置 1。该位由软件清 0。

	• 0：无更新事件产生； • 1：更新中断等待响应。 当发生以下更新寄存器事件时，该位由硬件置 1。 • 若 TIM1_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 TIM1_CR1 寄存器的 URS=0 且 UDIS=0，当设置 TIM1_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。 • 若 TIM1_CR1 寄存器的 URS=0 且 UDIS=0，当计数器 CNT 被触发事件重新初始化时。

12.3.6 TIM1 事件产生寄存器（TIM1_EGR）

偏移地址：0x14

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								w	w	w	w	w	w	w	w

位 15:8	Res：保留 必须保持复位值。
位 7	BG：产生刹车事件（Break generation） 该位由软件置 1，用于产生一个刹车事件。该位由硬件自动清 0。 • 0：无动作 • 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断，则产生相应的中断。
位 6	TG：触发产生（Trigger generation） 该位由软件置 1，用于产生一个事件，由硬件自动清 0。 • 0：无动作 • 1：TIM1_SR 中 TIF=1，若开启对应的中断，则产生相应的中断。
位 5	COMG：捕捉/比较控制更新产生（Capture/Compare control update generation） 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。
位 4	CC4G：捕捉/比较 4 发生（Capture/Compare 4 generation）
位 3	CC3G：捕捉/比较 3 发生（Capture/Compare 3 generation）
位 2	CC2G：捕捉/比较 2 发生（Capture/Compare 2 generation）
位 1	CC1G：捕捉/比较 1 发生（Capture/Compare 1 generation） 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 1 上产生一个捕获/比较事件。
	  ○ 若通道 CC1 配置为输出：设置 CC1IF=1，若开启对应的中断，则产生相应的中断。
	  ○ 若通道 CC1 配置为输入：当前的计数器值被捕获至 TIM1_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断，则产生相应的中断。若 CC1IF 已经为 1，则设置 CC1OF=1。

位 0	UG：产生更新事件（Update generation） 该位由软件置 1，由硬件自动清 0。 • 0：无动作； • 1：重新初始化计数器，并产生一个（寄存器）更新事件。 *注意：* *预分频器的计数器清 0，但预分频系数不变。若在中央对齐模式下或 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 TIM1_ARR 的值。*

12.3.7 TIM1 捕捉/比较模式寄存器 1（TIM1_CCMR1）

偏移地址：0x18

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
rw	rw			rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 15	OC2CE：使能输出比较 2 清 0（Output Compare 2 clear enable） 参见 OC1CE 位。
位 14:12	OC2M[2:0]：输出比较模式 2（Output Compare 2 mode） 参见 OC1M 位。
位 11	OC2PE：使能预装输出比较 2（Output Compare 2 preload enable） 参见 OC1PE 位。
位 10	OC2FE：使能快速输出比较 2（Output Compare 2 fast enable） 参见 OC1FE 位。
位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/Compare 2 selection） 该位域定义通道的方向（输入/输出），及选择输入信号。 • 00：CC2 通道配置为输出。 • 01：CC2 通道配置为输入，IC2 映射在 TI2 上。 • 10：CC2 通道配置为输入，IC2 映射在 TI1 上。 • 11：CC2 通道配置为输入，IC2 映射在 TRC 上。此模式仅在内部触发器输入被选中时工作（由 TIM1_SMCR 寄存器的 TS 位选择）。

位 7	OC1CE：允许输出比较 1 清零 （Output Compare 1 clear enable） • 0：OC1REF 不受 ETRF 输入的影响。 • 1：一旦检测到 ETRF 输入高电平，OC1REF 清 0。

位 6:4	OC1M[2:0]：输出比较模式 1（Output Compare 1 mode）
	该位域定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1 和 OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平分别取决于 CC1P、CC1NP。
	• 000：冻结。输出比较寄存器 TIM1_CCR1 与计数器 TIM1_CNT 间的比较，对 OC1REF 不起作用。
	• 001：匹配时设置通道 1 为有效电平。当计数器 TIM1_CNT 的值与捕获/比较寄存器 1（TIM1_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 TIM1_CNT 的值与捕获/比较寄存器 1（TIM1_CCR1）相同时，强制 OC1REF 为低。
	• 011：翻转。当 TIM1_CCR1=TIM1_CNT 时，翻转 OC1REF 的电平。
	• 100：强制为无效电平。强制 OC1REF 为低。
	• 101：强制为有效电平。强制 OC1REF 为高。
	• 110：PWM 模式 1
	  ○ 在向上计数时若 TIM1_CNT < TIM1_CCR1，则通道 1 为有效电平，否则为无效电平。
	  ○ 在向下计数时若 TIM1_CNT > TIM1_CCR1，则通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。
	• 111：PWM 模式 2
	  ○ 在向上计数时若 TIM1_CNT < TIM1_CCR1，则通道 1 为无效电平，否则为有效电平。
	  ○ 在向下计数时若 TIM1_CNT > TIM1_CCR1，则通道 1 为有效电平，否则为无效电平。

位 3	OC1PE：使能输出比较 1 预装载（Output Compare 1 preload enable） • 0：禁用 TIM1_CCR1 寄存器的预装载功能。可随时写入 TIM1_CCR1 寄存器，并且新写入的数值立即生效。 • 1：开启 TIM1_CCR1 寄存器的预装载功能。读写操作仅针对预装载寄存器，TIM1_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。
位 2	OC1FE：输出比较 1 快速使能（Output Compare 1 fast enable） 该位可加快 CC 输出对触发输入事件的响应。 • 0：CC1 的正常操作依赖于计数器与 CCR1 的值，即使触发器处于工作状态。当触发器的输入产生一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 • 1：输入到触发器的有效沿的作用就像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 模式时生效。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare 1 selection） 该位域定义通道的方向（输入/输出），及所选择的输入信号。

	• 00：CC1 通道被配置为输出。 • 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 • 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 • 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅在内部触发器输入被选中时（通过 TIM1_SMCR 寄存器的 TS 位选择）工作。

输入捕捉模式

位 15:12	IC2F[3:0]：输入捕捉 2 滤波器（Input capture 2 filter） 该位域定义了 TI2 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 11:10	IC2PSC[1:0]：输入捕捉 2 预分频器 （Input capture 2 prescaler）
位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/Compare 2 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC2 通道被配置为输出； • 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； • 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； • 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。
位 7:4	IC1F[3:0]：输入捕捉 1 滤波器（Input capture 1 filter） 该位域定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样

	• 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 3:2	IC1PSC[1:0]：输入捕捉 1 预分频器（Input capture 1 prescaler） 该位域定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0（在 TIM1_CCER 寄存器中），则预分频器复位。 • 00：无预分频器，捕获输入口上检测到的每一个边沿都会触发一次捕获。 • 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。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare 1 Selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC1 通道被配置为输出。 • 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 • 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 • 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）工作。

12.3.8 TIM1 捕捉/比较模式寄存器 2（TIM1_CCMR2）

偏移地址：0x1C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
rw	rw			rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

位 15	OC4CE：输出比较4 清除使能（Output compare 4 clear enable）
位 14:12	OC4M[2:0]：输出比较 4 模式（Output compare 4 mode）
位 11	OC4PE：输出比较4 预分频使能（Output compare 4 preload enable）
位 10	OC4FE：输出比较4 快速使能（Output compare 4 fast enable）
位 9:8	CC4S[1:0]：捕捉/比较4 选择（Capture/Compare 4 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC4 通道被配置为输出。 • 01：CC4 通道被配置为输入，IC4 映射在 TI4 上。 • 10：CC4 通道被配置为输入，IC4 映射在 TI3 上。 • 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）工作。
位 7	OC3CE：输出比较3 清除使能（Output compare 3 clear enable）
位6:4	OC3M[2:0]：输出比较3 模式（Output compare 3 mode）
位 3	OC3PE：输出比较3 预分频使能（Output compare 3 preload enable）
位 2	OC3FE：输出比较3 快速使能 （Output compare 3 fast enable）
位 1:0	CC3S[1:0]：捕捉/比较 3 选择（Capture/Compare 3 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC3 通道被配置为输出。 • 01：CC3 通道被配置为输入，IC4 映射在 TI3 上。 • 10：CC3 通道被配置为输入，IC4 映射在 TI4 上。 • 11：CC3 通道被配置为输入，IC4 映射在 TRC 上。

输入捕捉模式

位 15:12	IC4F[3:0]：输入捕捉4 滤波器（Input capture 4 filter） 该位域定义了 TI4 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8

	• 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 11:10	IC4PSC[1:0]：输入捕捉4 预分频器 （Input capture 4 prescaler）
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare 4 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC4 通道被配置为输出。 • 01：CC4 通道被配置为输入，IC3 映射在 TI4 上。 • 10：CC4 通道被配置为输入，IC3 映射在 TI3 上。 • 11：CC4 通道被配置为输入，IC3 映射在 TRC 上。
位 7:4	IC3F[3:0]：输入捕捉3 滤波器（Input capture 3 filter） 该位域定义了 TI3 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 3:2	IC3PSC[1:0]：输入比较 3 预分频器（Input capture 3 prescaler）
位 1:0	CC3S[1:0]：捕捉/比较 3 选择（Capture/compare 3 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择：

	• 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。

12.3.9 TIM1 捕捉/比较使能寄存器（TIM1_CCER）

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15:14	Res：保留 必须保持复位值。
位 13	CC4P：捕捉/比较 4 输出极性（Capture/Compare 4 output polarity）
位 12	CC4E：捕捉/比较 4 输出使能（Capture/Compare 4 output enable）
位 11	CC3NP：捕捉/比较 3 互补输出极性（Capture/Compare 3 complementary output polarity）
位 10	CC3NE：捕捉/比较 3 互补输出使能（Capture/Compare 3 complementary output enable）
位 9	CC3P：捕捉/比较 3 输出极性（Capture/Compare 3 output polarity）
位 8	CC3E：捕捉/比较 3 输出使能（Capture/Compare 3 output enable）
位 7	CC2NP：捕捉/比较 2 互补输出极性（Capture/Compare 1 complementary output polarity）
位 6	CC2NE：捕捉/比较 2 互补输出使能（Capture/Compare 2 complementary output enable）
位 5	CC2P：捕捉/比较 2 输出极性（Capture/Compare 2 output polarity）
位 4	CC2E：捕捉/比较 2 输出使能（Capture/Compare 2 output enable）
位 3	CC1NP：捕捉/比较 1 互补输出极性（Capture/Compare 1 complementary output polarity）
位 2	CC1NE：捕捉/比较 1 互补输出使能（Capture/Compare 1 complementary output enable） • 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。
位 1	CC1P：捕捉/比较 1 输出极性（Capture/Compare 1 output polarity） • CC1 通道配置为输出： o 0：OC1 高电平有效。 o 1：OC1 低电平有效。 • CC1 通道配置为输入： CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 o 00：非反相/上升沿 电路作用于 TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 非反相。 o 01：反相/下降沿 电路作用于 TIxFP1 的下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作）TIxFP1 反相。 o 00：保留不用 o 11：非反相/上升或下降沿 电路作用于 TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 非反相（在门控模式）。在编码模式下不能使用此配置。
位 0	CC1E：捕捉/比较1 输出使能（Capture/Compare 1 output enable） • CC1 通道配置为输出： o 0：关闭 OC1 禁止输出，因此 OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 o 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 CC1E 位的值。 • CC1 通道配置为输入： 该位决定是否将一个定时器值的捕捉装载到捕捉/比较寄存器 1（TIM1_CCR1）。 o 0：捕捉禁止 o 1：捕捉使能

表 12-4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态 (1)	
MOE 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

控制位					输出状态 (1)	
		0	0	1	输出禁止（与定时器断开） OCx=0，OCx_EN=0	OCxREF + 极性， OCxN= OCxREF xor CCxNP， 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（输出使能且为无效电平）OCx=CCxP，OCx_EN=1	OCxREF + 极性， OCxN= OCxREF xor CCxNP， OCxN_EN=1
		1	1	0	OCxREF + 极性， OCx= OCxREF xor CCxP， OCx_EN=1	关闭状态（输出使能且为无效电平） 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） 异步地：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） 异步地： 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。

说明：引脚连接到互补的 OCx 和 OCxN 通道的外部 I/O 引脚的状态，取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器。

12.3.10 TIM1 计数寄存器（TIM1_CNT）

偏移地址：0x24

复位值：0x0000

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

位 15:0	CNT[15:0]：计数器值（Counter value）

12.3.11 TIM1 预分频寄存器（TIM1_PSC）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频值（Prescaler value） 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/（PSC[15:0]+1）。每次当更新事件产生时，PSC 的值被装入当前预分频器寄存器。 更新事件包括计数器被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

12.3.12 TIM1 自动重装载寄存器（TIM1_ARR）

偏移地址：0x2C

复位值：0x0000

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

位 15:0	ARR[15:0]：自动重装载的值（Auto-reload value） ARR 包含了将要装载实际的自动重装载寄存器的值。

12.3.13 TIM1 重复计数寄存器（TIM1_RCR）

偏移地址：0x30

复位值：0x0000

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

位 15:8	Res：保留 必须保持复位值。
位 7:0	REP[7:0]：重复计数器的值（Repetition counter value） 使能预装载寄存器后，该位域允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如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 TIM1_RCR 寄存器写入的新值仅在下次周期更新事件发生时才起作用。这意味着在 PWM 模式中，（REP+1）对应着： • 在边沿对齐模式下，PWM 周期的数目。

	• 在中央对齐模式下，PWM 半周期的数目。

12.3.14 TIM1 捕捉/比较寄存器 1（TIM1_CCR1）

偏移地址：0x34

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw															

位 15:0	CCR1[15:0]：捕捉/比较通道1 的值（Capture/Compare 1 value） • 若 CC1 通道配置为输出： CCR1 决定了装入当前捕获/比较1 寄存器的值（预装载值）。如果在 TIM1_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 TIM1_CCR1 寄存器。否则只有当更新事件发生时，此预装载值才传输至 TIM1_CCR1 寄存器。TIM1_CCR1 寄存器参与同计数器 TIM1_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。

12.3.15 IM1 捕捉/比较寄存器 2（TIM1_CCR2）

偏移地址：0x38

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw															

位 15:0	CCR2[15:0]：捕捉/比较通道 2 的值（Capture/Compare 2 value） • 若 CC2 通道配置为输出： CCR2 决定了装入 TIM_CCR2 寄存器的值（预装载值）。如果在 TIM1_CCMR2 寄存器（OC2PE 位）中未选择预装载功能，写入的数值会立即传输至 TIM_CCR2 寄存器。否则只有当更新事件发生时，此预装载值才传输至 TIM_CCR2 寄存器。TIM1_CCR2 寄存器参与同计数器 TIM1_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。

12.3.16 TIM1 捕捉/比较寄存器 3（TIM1_CCR3）

偏移地址：0x3C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw															

位 15:0	CCR3[15:0]：捕捉/比较通道 3 的值（Capture/Compare 3 value）

	• 若 CC3 通道配置为输出： CCR3 决定了装入 TIM1_CCR3 寄存器的值（预装载值）。如果在 TIM1_CCMR3 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 TIM1_CCR3 寄存器。否则只有当更新事件发生时，此预装载值才传输至 TIM1_CCR3 寄存器。TIM1_CCR3 寄存器参与同计数器 TIM1_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。

12.3.17 TIM1 捕捉/比较寄存器 4（TIM1_CCR4）

偏移地址：0x40

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw															

位 15:0	CCR4[15:0]：捕捉/比较通道 4 的值（Capture/Compare 4 value） • 若 CC4 通道配置为输出： CCR4 决定了装入 TIM1_CCR4 寄存器的值（预装载值）。如果在 TIM1_CCMR4 寄存器（OC4PE 位）中未选择预装载功能，写入的数值会立即传输至 TIM1_CCR4 寄存器。否则只有当更新事件发生时，此预装载值才传输至 TIM1_CCR4 寄存器。TIM1_CCR4 寄存器参与同计数器 TIM1_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。

12.3.18 TIM1 刹车和死区寄存器（TIM1_BDTR）

偏移地址：0x44

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
rw	rw	rw	rw	rw	rw	rw		rw							

位15	MOE：主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的值，该位可由软件清 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（TIM1_CCER 寄存器的 CCxE、CCxNE 位），则开启 OC 和 OCN 输出。
位 14	AOE：自动输出使能（Automatic output enable） • 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）。

位 13	BKP：刹车输入极性（Break polarity） • 0：刹车输入低电平有效 • 1：刹车输入高电平有效
位 12	BKE：刹车使能（Break enable） • 0：刹车输入禁止（BRK、CCS 时钟失效事件及比较器输出结果） • 1：刹车输入允许（BRK、CCS 时钟失效事件及比较器输出结果）
位 11	OSSR：运行模式下“关闭状态”选择（Off-state selection for Run mode） 该位用于当 MOE=1 且通道为互补输出时，没有互补输出的定时器中不存在 OSSR 位的情况。 • 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 使能并输出无效电平，然后置 OC/OCN 使能输出信号=1。
位 10	OSSI：空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOE=0 时通道为输出。 • 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 被强制输出空闲电平，且置 OC/OCN 使能输出信号=1。
位 9:8	LOCK[1:0]：锁定设置（Lock configuration） 该位域为防止软件错误而提供写保护。 • 00：锁定关闭，寄存器无写保护。 • 01：锁定级别 1，不能写入 TIM1_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIM1_CR2 寄存器的 OISx/OISxN 位。 • 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIM1_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位。 • 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIM1_CCMRx 寄存器的 OCxM/OCxPE 位）（
位 7:0	DTG[7:0]：死区发生器设置（Dead-time generator setup） 该位域定义了插入互补输出之间的死区持续时间。

13 通用定时器（TIM2）

通用定时器是一个由可编程预分频器驱动的 16 位自动装载计数器构成。它适用于多种场合，包括测量输入信号的脉冲长度（输入捕获）或者产生输出波形（输出比较和 PWM）。

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

每个通用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。TIM2 的特性如下：

表 13-1 TIM2 特性

符号	参数	条件	最小值	典型值	最大值	单位
$t_{res（TIM）}$	定时器分辨时间	$f_{TIMxCLK}$=48MHz	-	20.8	-	ns
f_{EXT}	定时器的 CH1 至 CH4，外部输入的时钟频率	-	-	$f_{TIMxCLK}/2$	-	MHz
		$f_{TIMxCLK}$=48 MHz	-	24	-	MHz
t_{MAX_COUNT}	当选择内部时钟时，16 位计数器的时钟周期	-	-	2^{16}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48 MHz	-	1365	-	μs

13.1 TIM2 主要功能

通用 TIM2 定时器功能包括：

- 四路输入通道都支持上升沿、下降沿触发和双沿触发功能
- 16 位向上、向下、向上/向下自动装载计数器
- 16 位可编程（可实时修改）预分频器，计数器时钟频率的分频系数为 1~65536 之间的任意数值。
- 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互连的同步电路
- 以下事件发生时产生中断：
  - 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

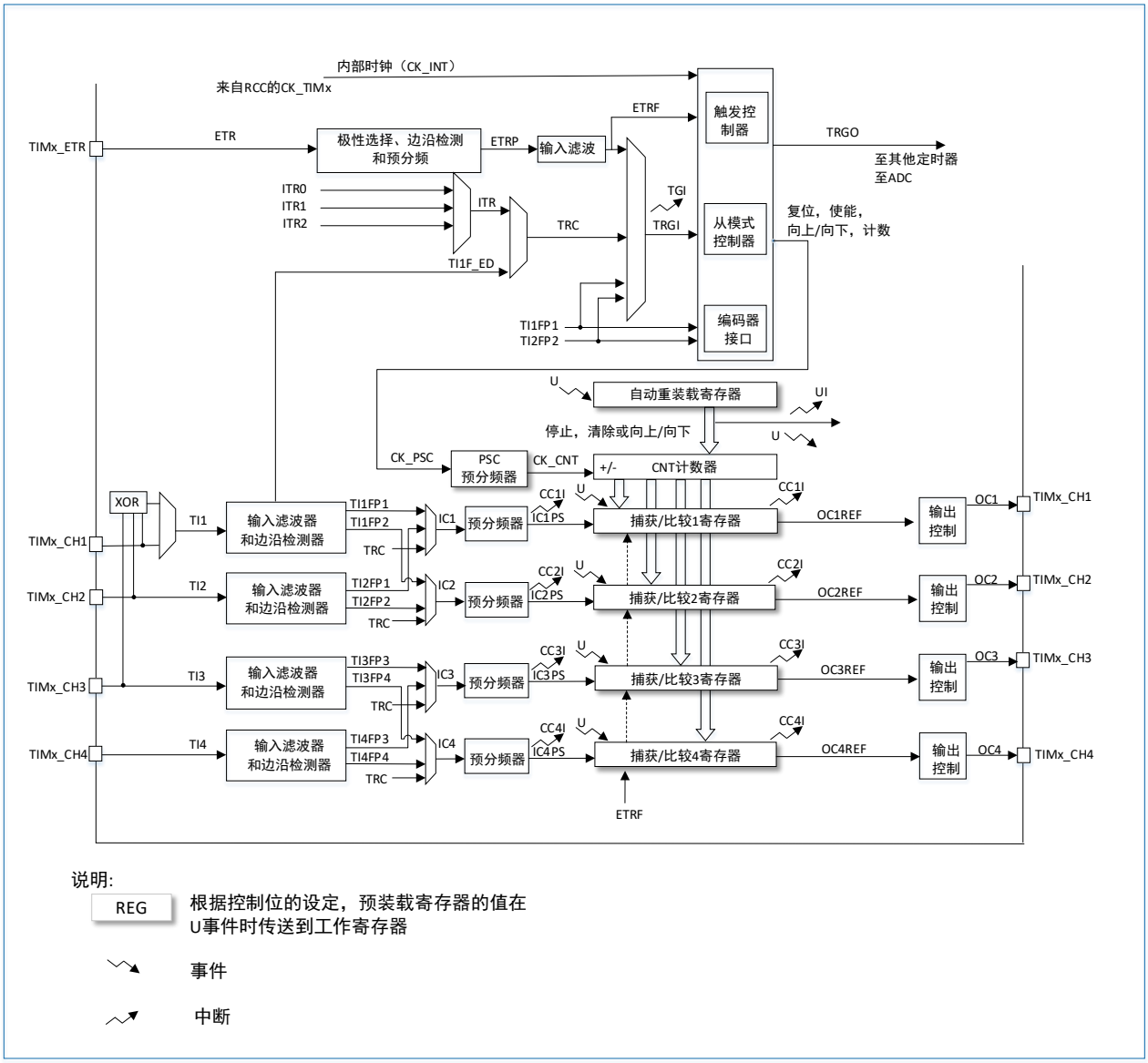

图 13-1 通用定时器框图

13.2 TIM2 功能描述

13.2.1 时基单元

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

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，且在计数器运行时仍可以读写。

时基单元包含：

- 计数器寄存器（TIM2_CNT）
- 预分频器寄存器（TIM2_PSC）
- 自动装载寄存器（TIM2_ARR）

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 TIM2_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 到来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 TIM2_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 TIM2_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效。

说明：真正的计数器使能信号 CNT_EN 是在 CEN 的一个时钟周期后被设置。

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 TIM2_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器参数在下一次更新事件到来时被采用。

下面两个图给出了在预分频器运行时，更改计数器参数的例子。

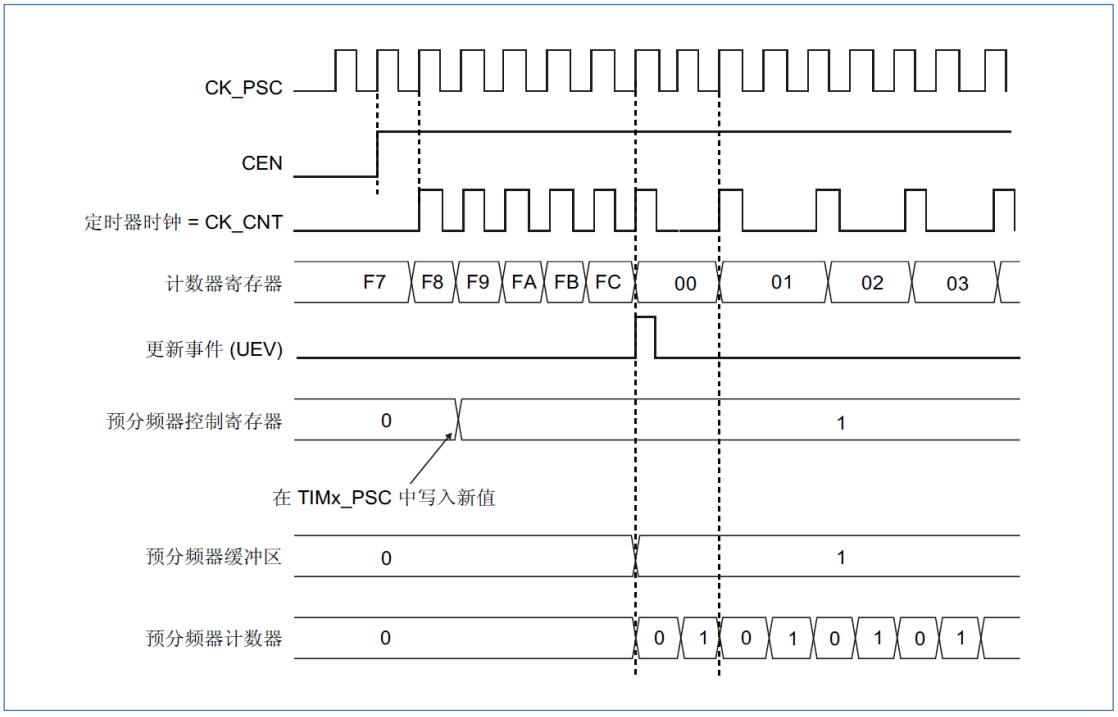

图 13-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

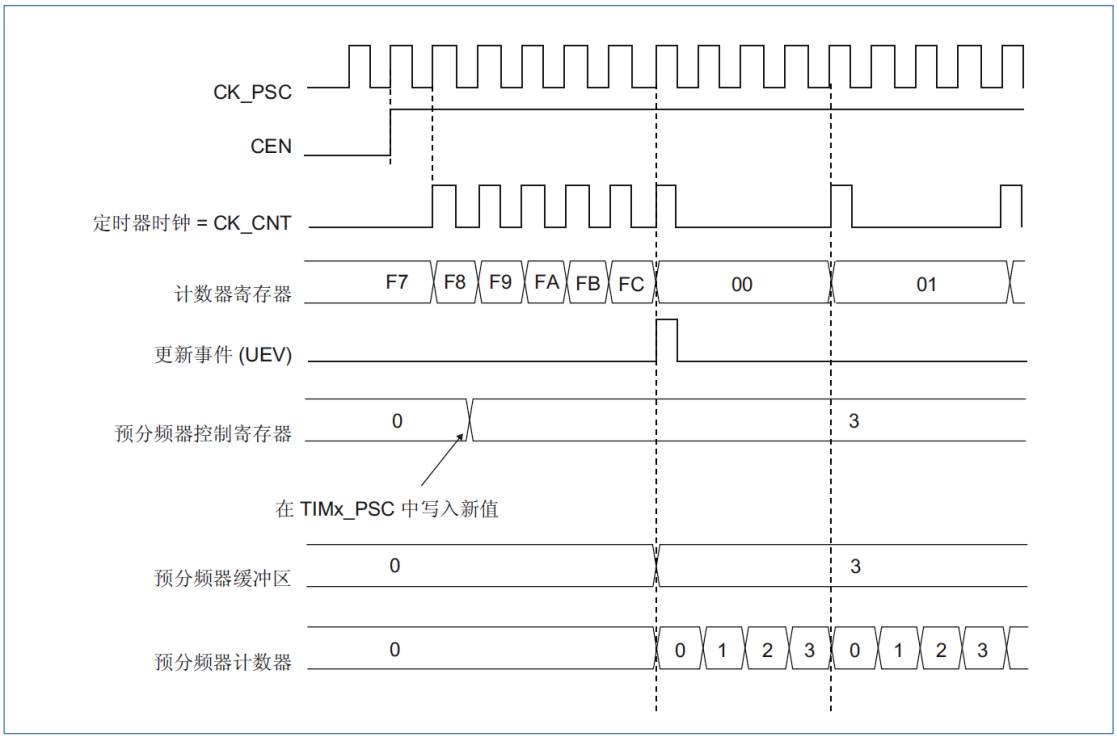

图 13-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

13.2.2 计数器模式

13.2.2.1 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（TIM2_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 TIM2_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 TIM2_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清零（但预分频系数不变）。此外，如果设置了 TIM2_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断）；这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 URS 位）设置更新标志位（TIM2_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（TIM2_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（TIM2_ARR）。

下图给出一些例子，当 TIM2_ARR=0x36 时计数器在不同时钟频率下的动作。

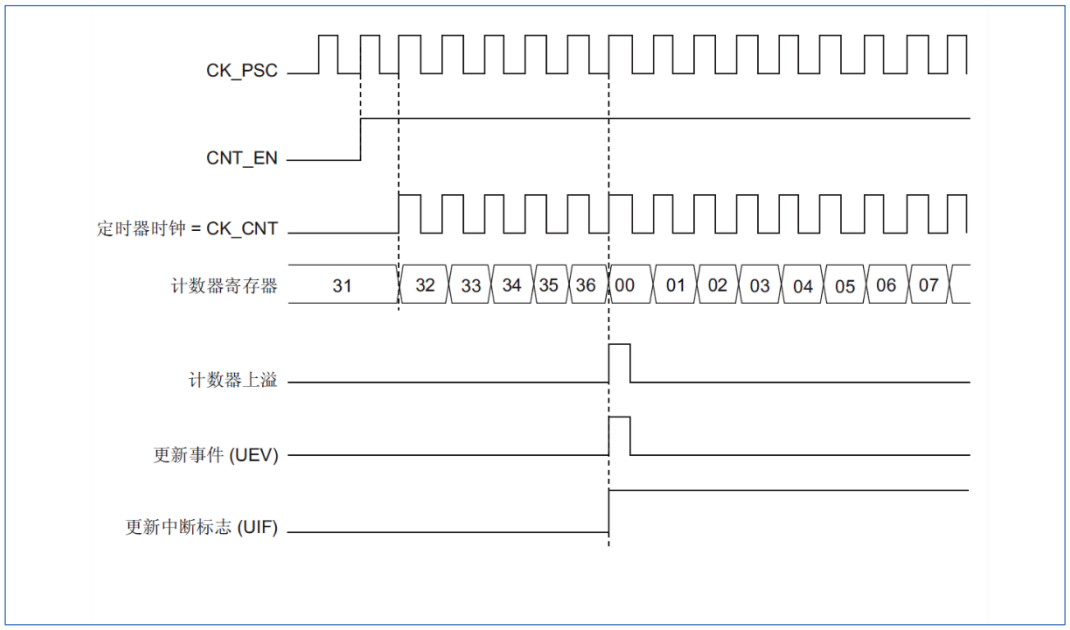

图 13-4 计数器时序图，内部时钟分频因子为 1

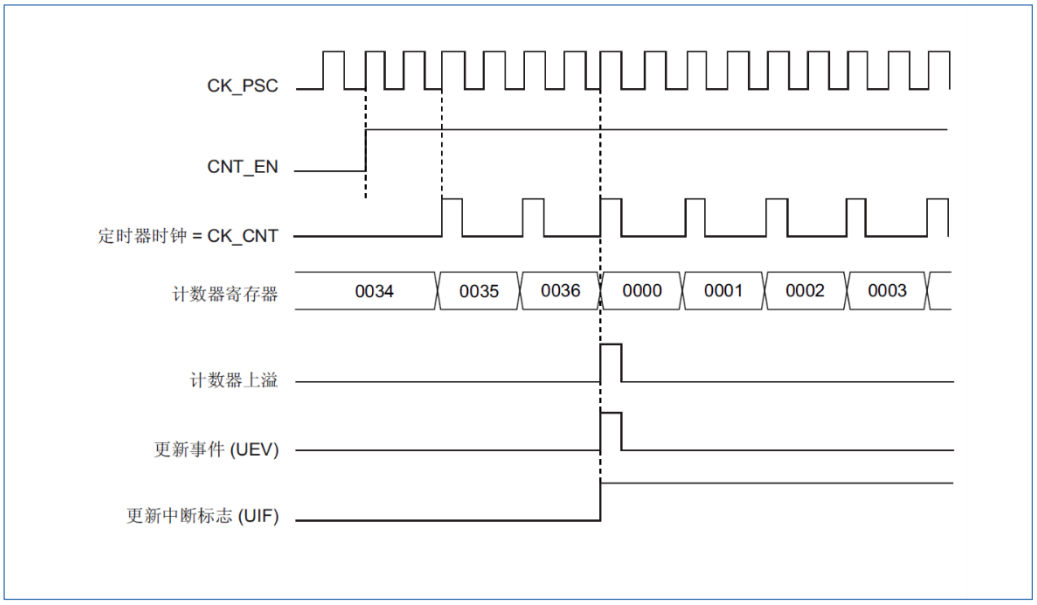

图 13-5 计数器时序图，内部时钟分频因子为 2

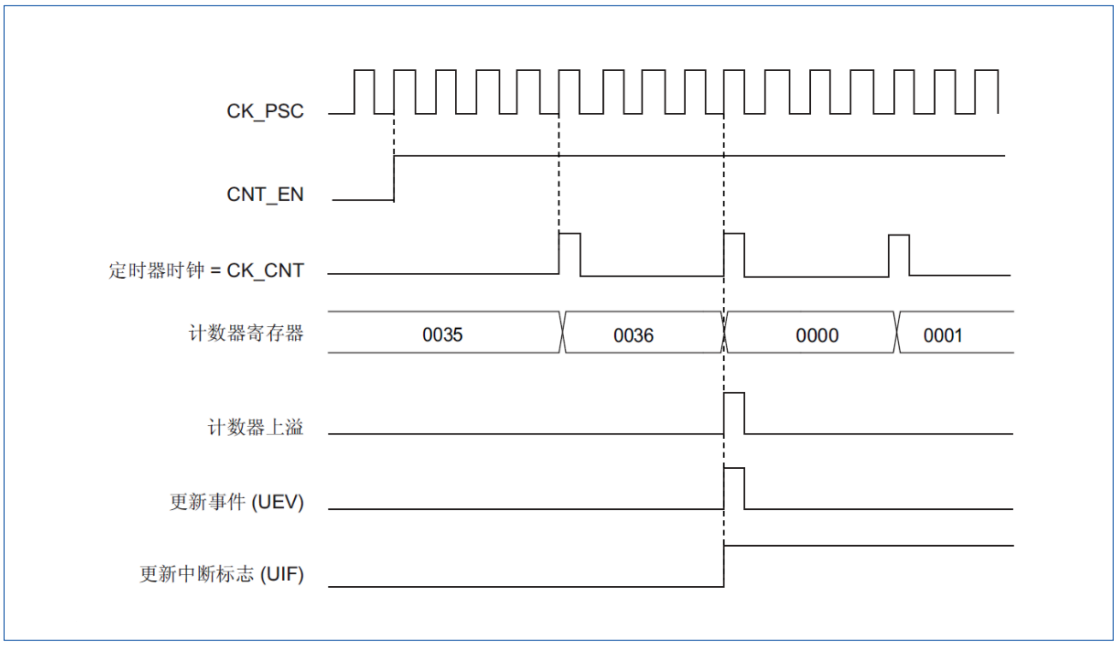

图 13-6 计数器时序图，内部时钟分频因子为 4

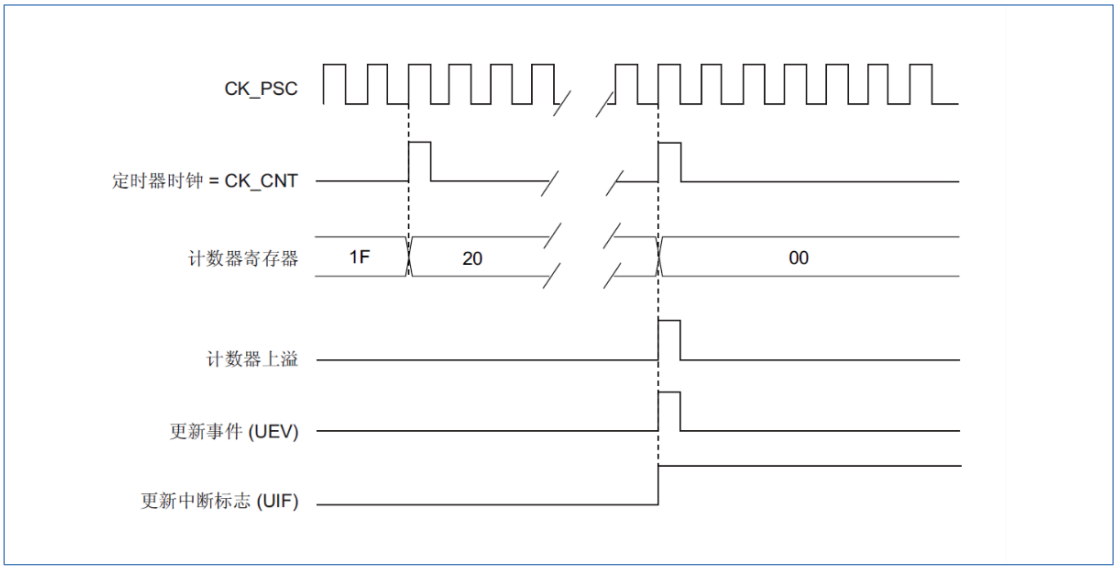

图 13-7 计数器时序图，内部时钟分频因子为 N

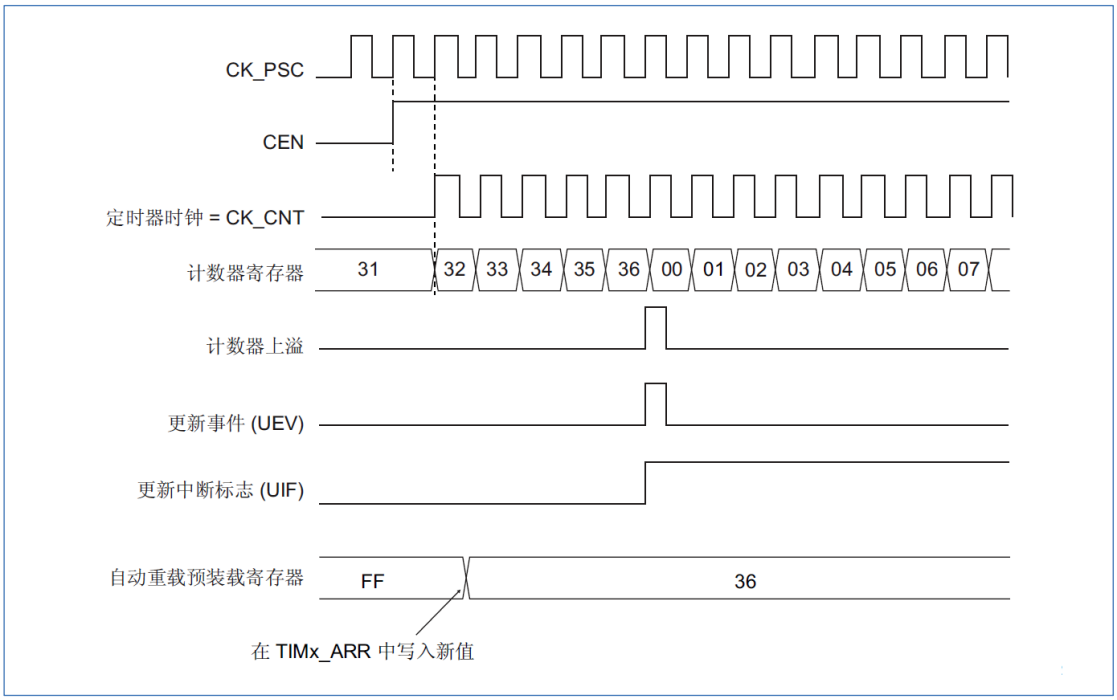

图 13-8 计数器时序图，当 ARPE=0 时的更新事件（TIM2_ARR 没有预装入）

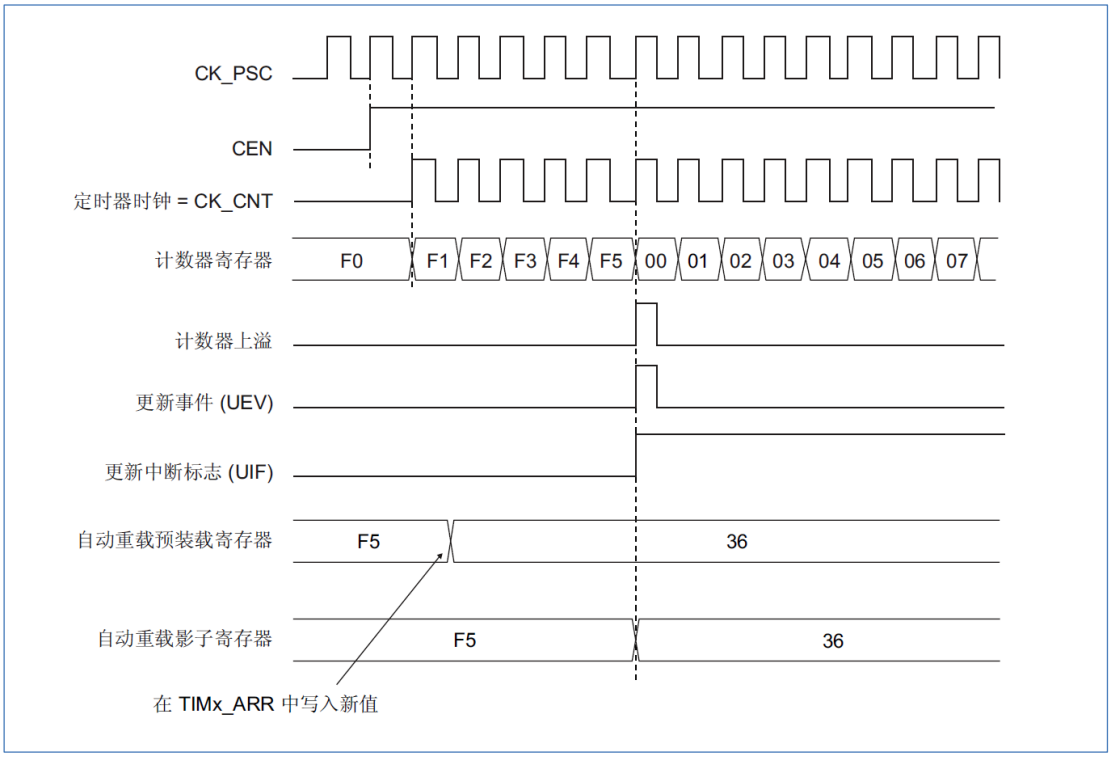

图 13-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM2_ARR）

13.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（TIM2_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 TIM2_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 TIM2_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，同时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 TIM2_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM2_SR 寄存器中的 UIF 位）也被设置。

- 预分频器的缓存器被置入预装载寄存器的值（TIM2_PSC 寄存器的值）。
- 当前的自动加载寄存器被更新为预装载值（TIM2_ARR 寄存器中的内容）。

说明：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 TIM2_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

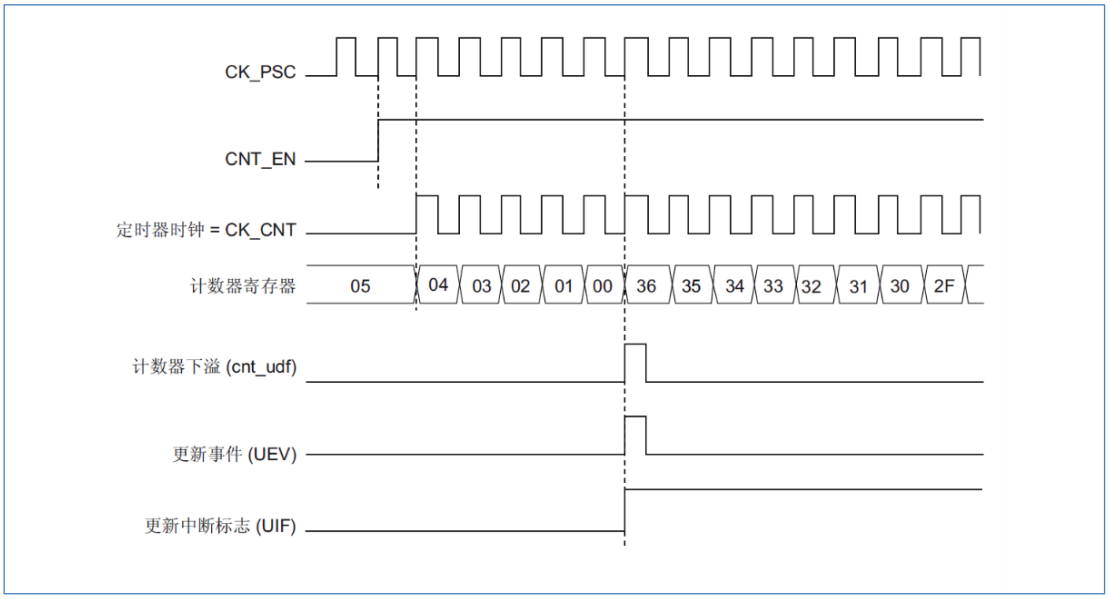

图 13-10 计数器时序图，内部时钟分频因子为 1

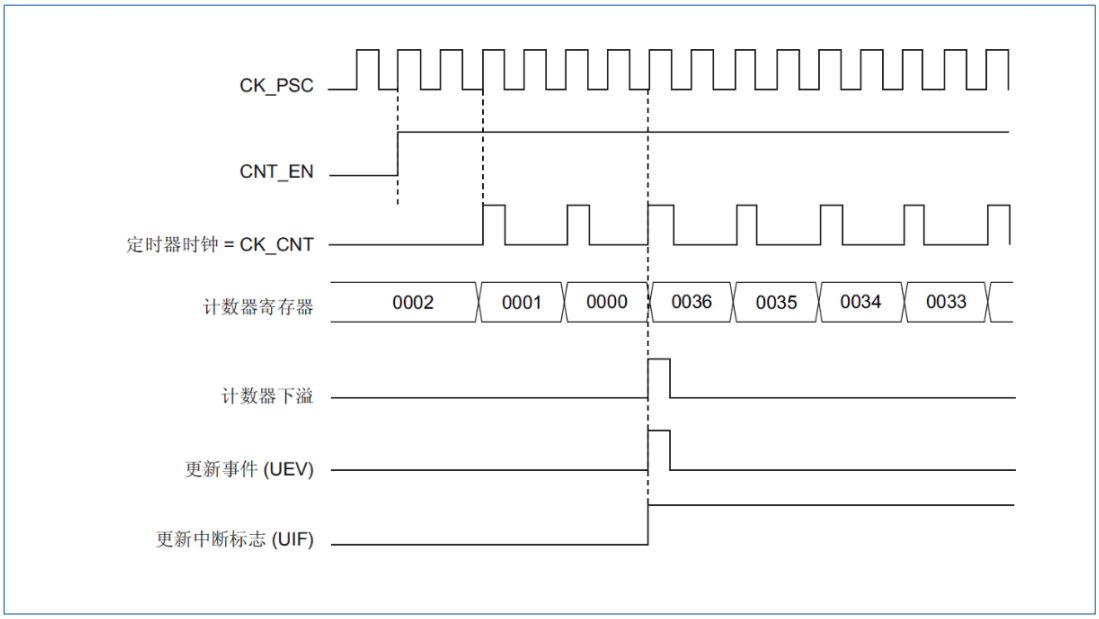

图 13-11 计数器时序图，内部时钟分频因子为 2

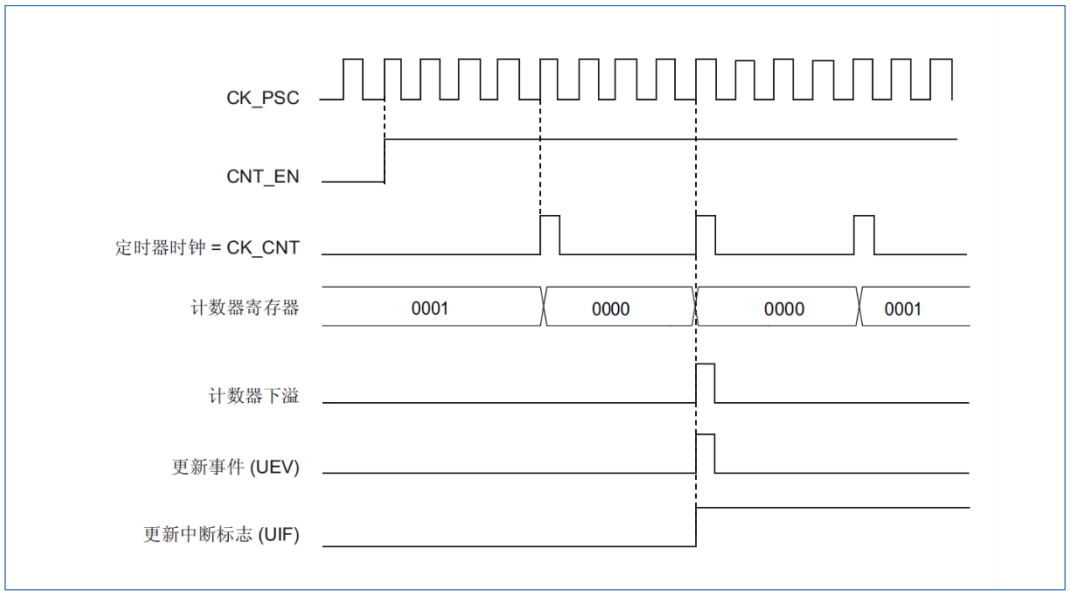

图 13-12 计数器时序图，内部时钟分频因子为 4

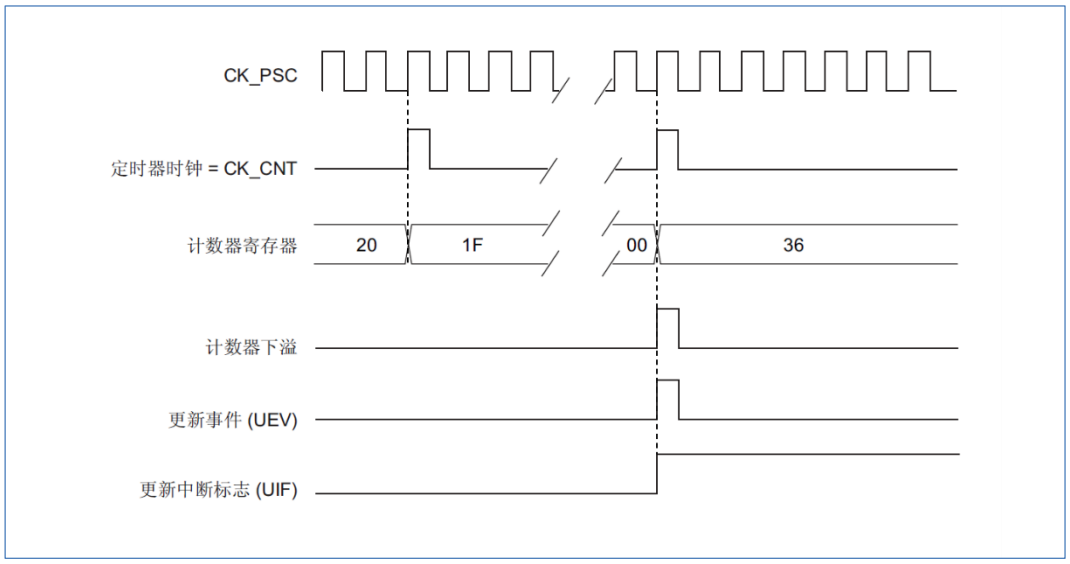

图 13-13 计数器时序图，内部时钟分频因子为 N

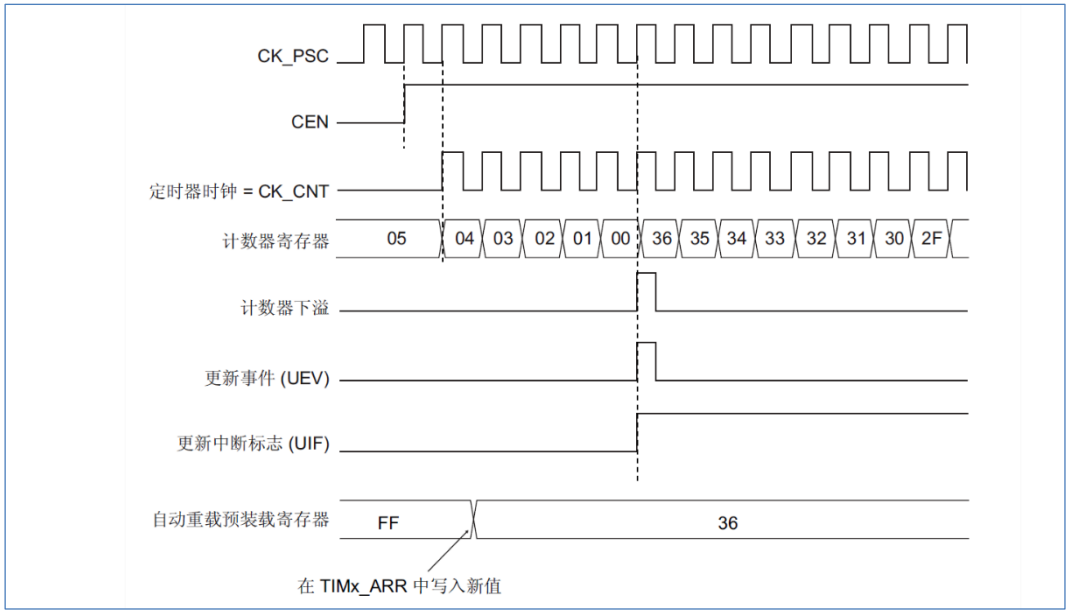

图 13-14 计数器时序图，当没有使用重复计数器时的更新事件

13.2.2.3 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（TIM2_ARR 寄存器）减 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。

在这个模式，不能写入 TIM2_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 TIM2_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 TIM2_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 TIM2_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM2_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（TIM2_PSC 寄存器）的值。
- 当前的自动加载寄存器被更新为预装载值（TIM2_ARR 寄存器中的内容）。

说明：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不同时钟频率下的操作的例子：

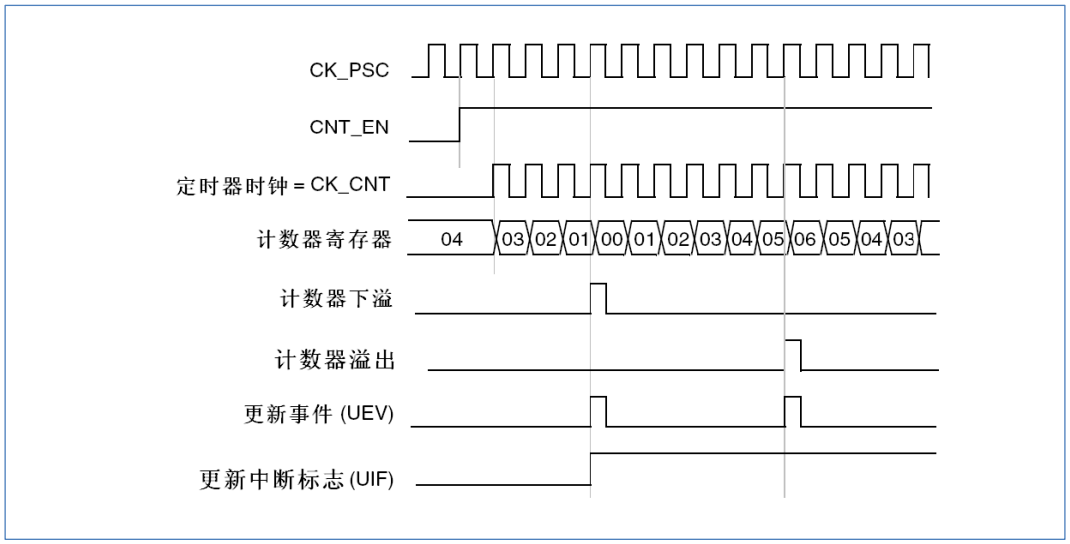

图 13-15 计数器时序图，内部时钟分频因子为 1，TIM2_ARR=0x6（这里使用了中央对齐模式 1）

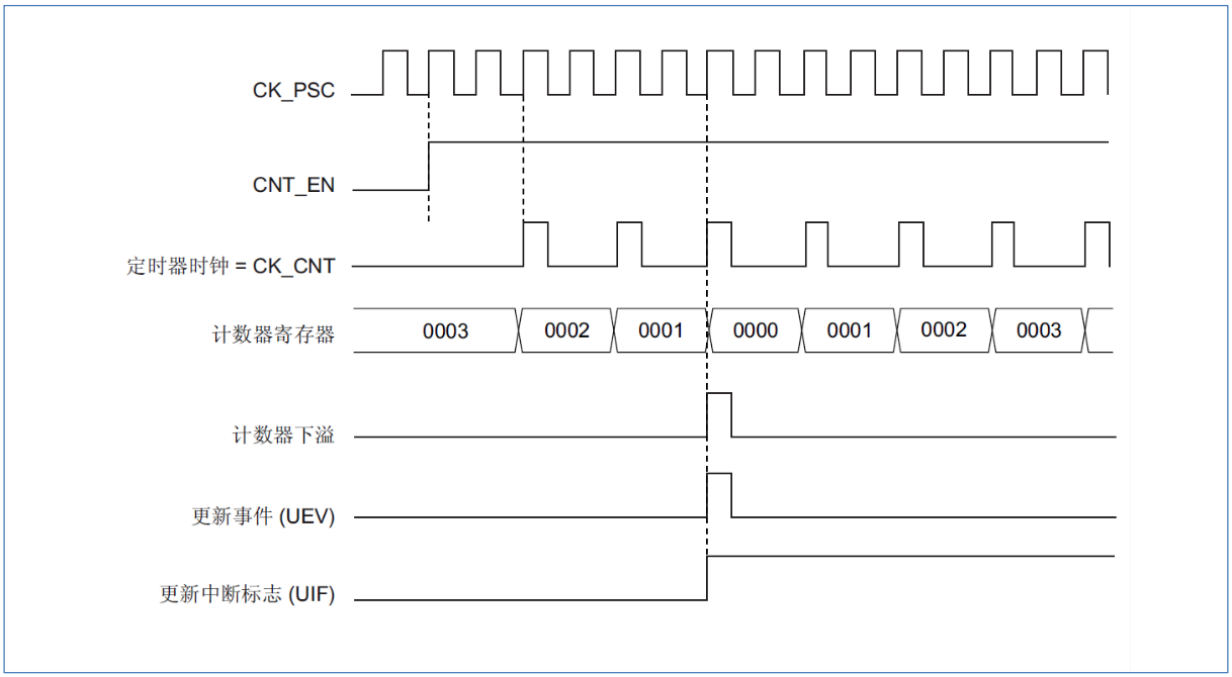

图 13-16 计数器时序图，内部时钟分频因子为 2（这里使用了中央对齐模式 1）

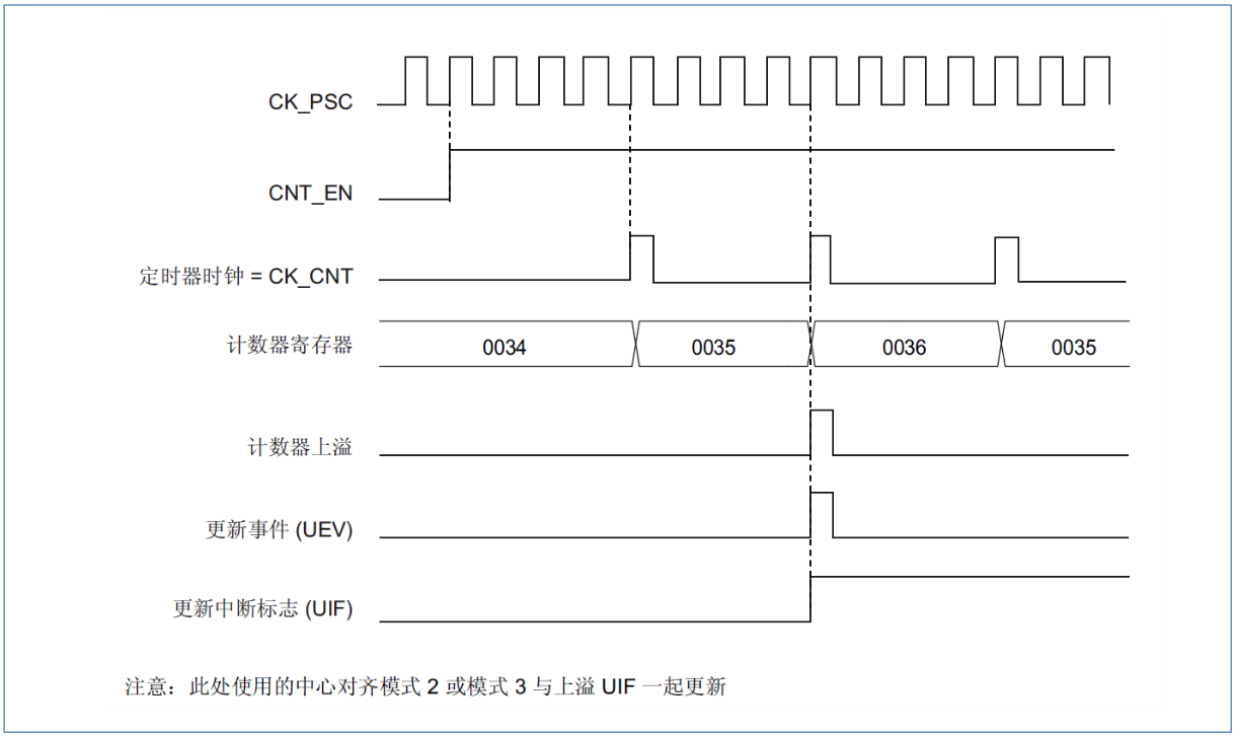

图 13-17 计数器时序图，内部时钟分频因子为 4，TIM2_ARR=0x36（这里使用了中央对齐模式 2 或 3）

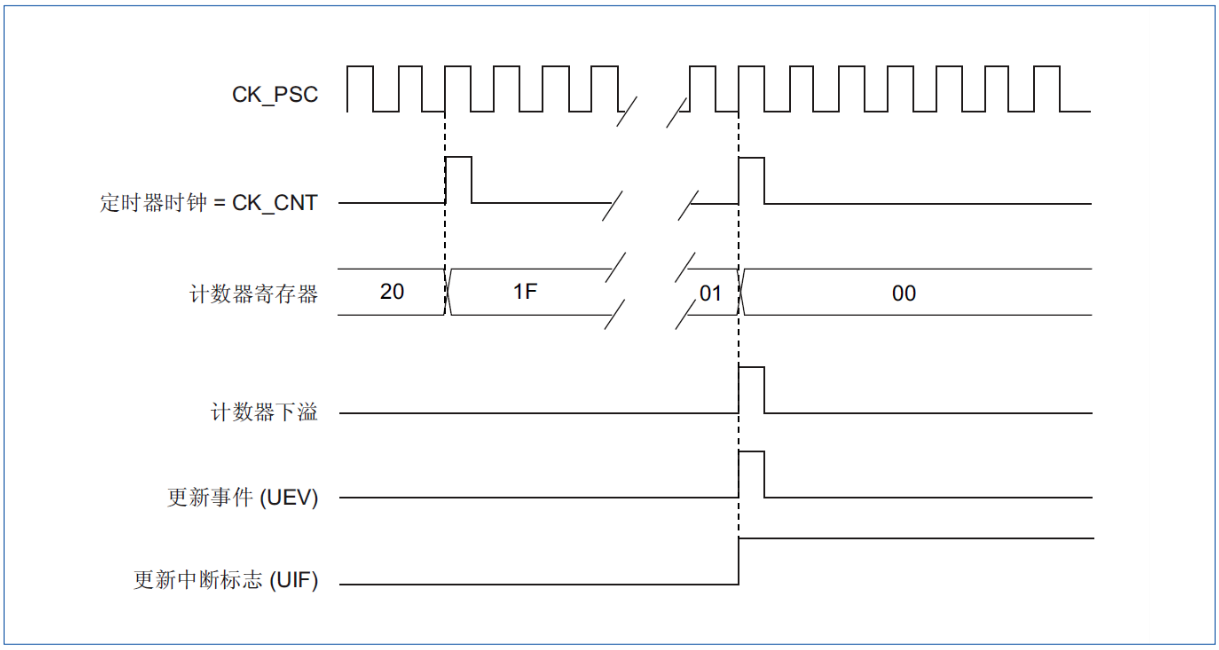

图 13-18 计数器时序图，内部时钟分频因子为 N（这里使用了中央对齐模式 1）

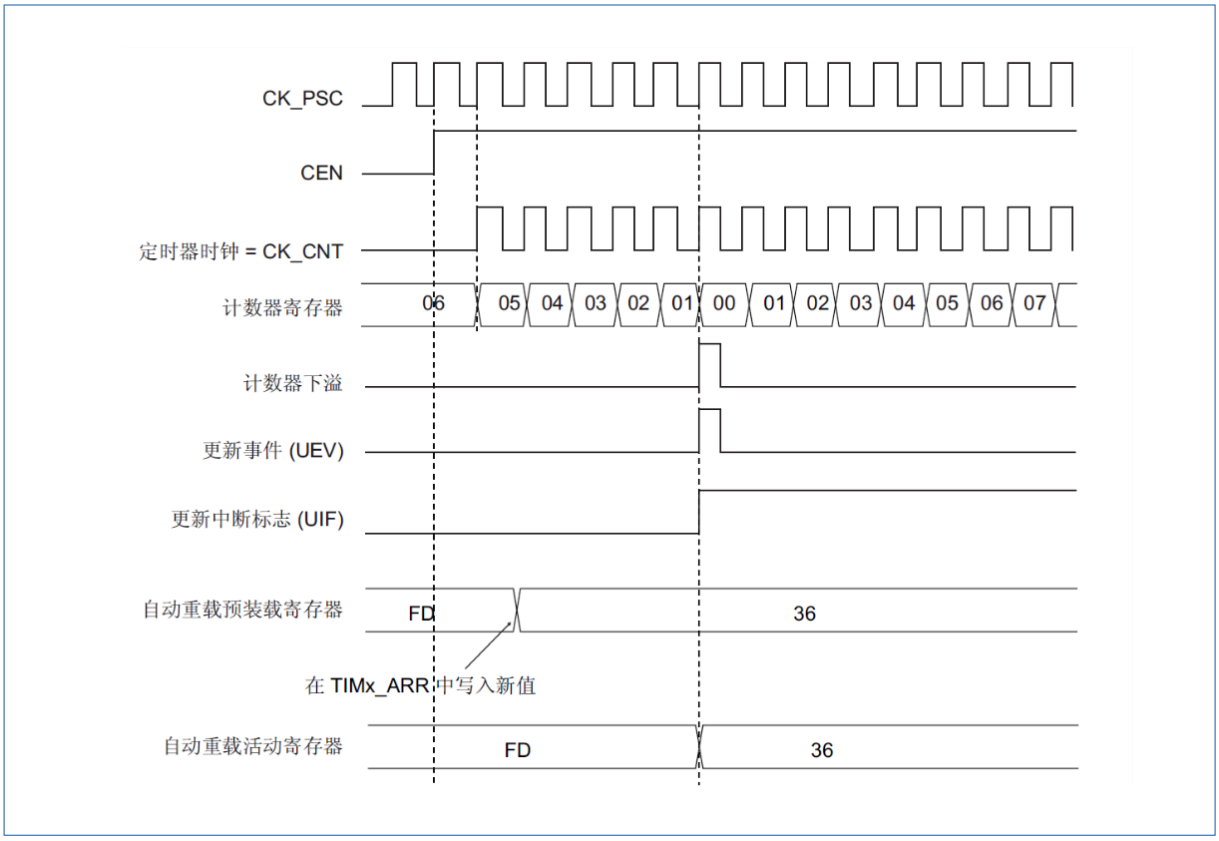

图 13-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

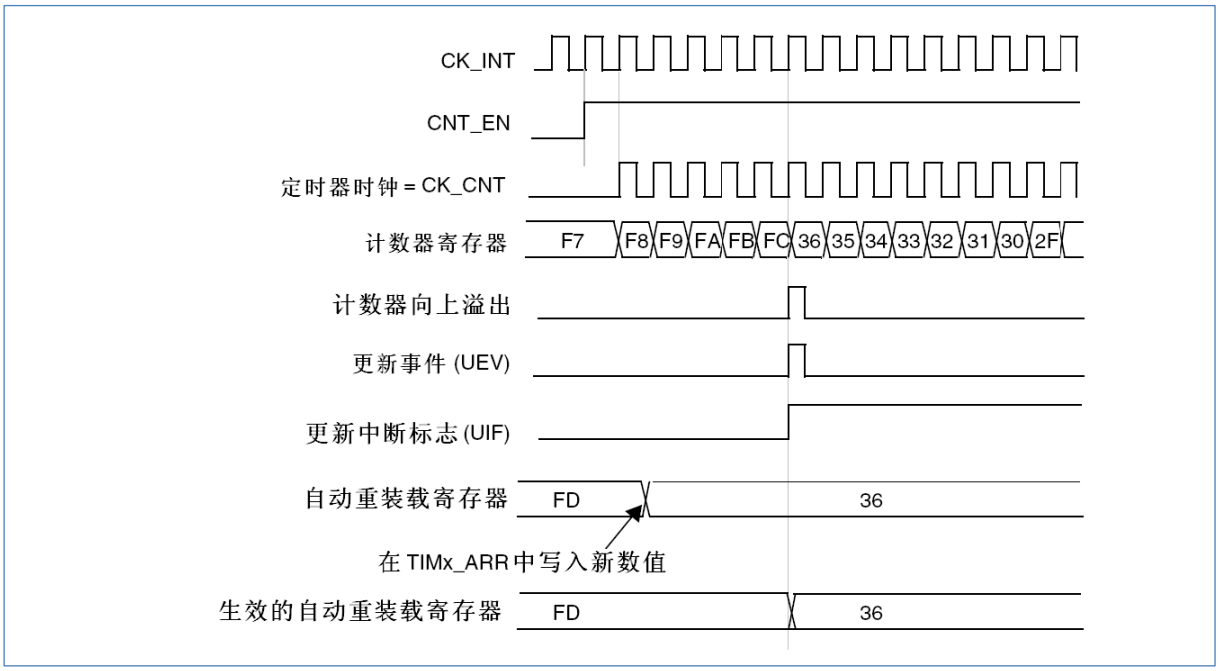

图 13-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

13.2.3 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 1：外部输入引脚（TIx）
- 外部时钟模式 2：外部触发输入（ETR）
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器，如配置一个定时器 Timer1 作为另一个定时器 Timer2 的预分频器。详细信息，参见“13.2.15.1 使用一个定时器作为另一个定时器的预分频器”。

内部时钟源（CK_INT）

如果禁止了从模式控制器（TIM2_SMCR 寄存器的 SMS=000），则 CEN、DIR（TIM2_CR1 寄存器）和 UG 位（TIM2_EGR 寄存器）是实际的控制位，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。

下图显示了控制电路和向上计数器在一般模式下，不带预分频器时的操作。

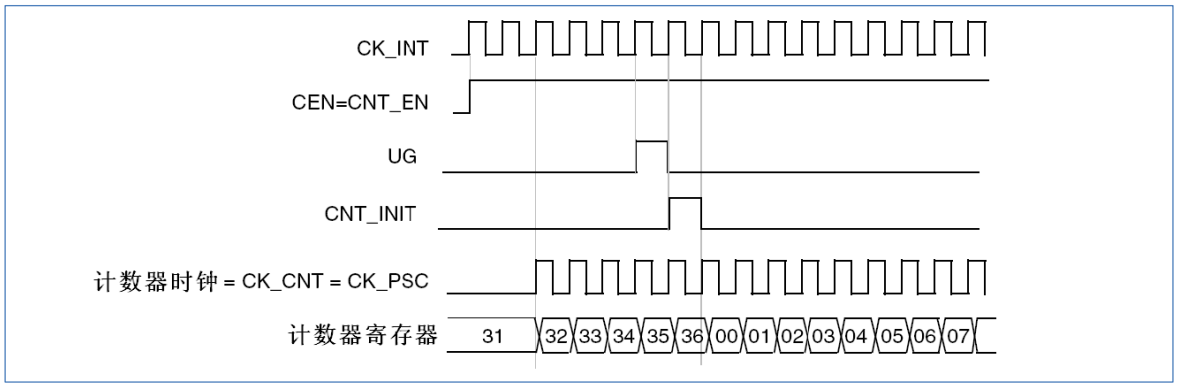

图 13-21 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

当 TIM2_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。

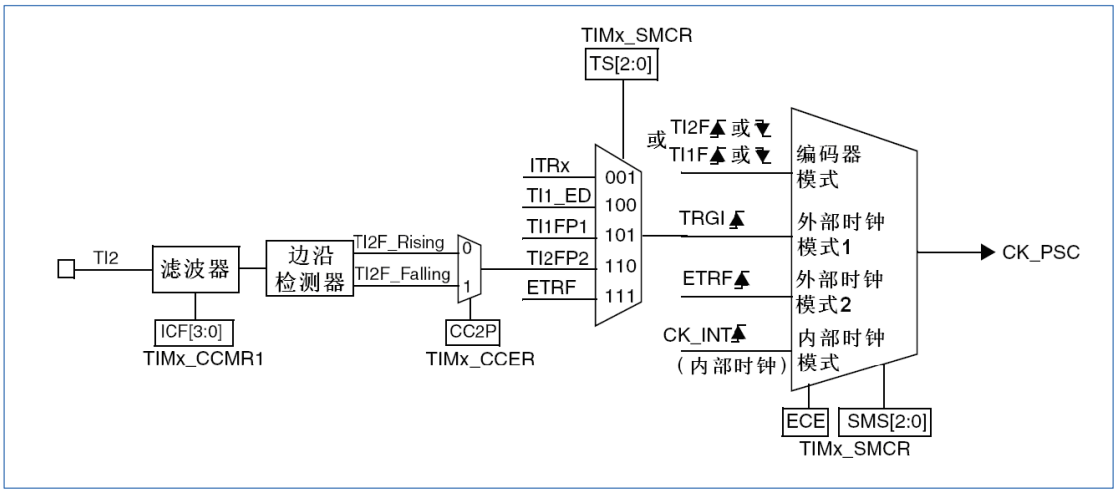

图 13-22 TI2 外部时钟连接例子

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 配置 TIM2_CCMR1 寄存器 CC2S=‘01’，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TIM2_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）。

说明：捕获预分频器不用作触发，所以不需要对它进行配置。

3. 配置 TIM2_CCER 寄存器的 CC2P=’0’，选定上升沿极性。
4. 配置 TIM2_SMCR 寄存器的 SMS=’111’，选择定时器外部时钟模式 1。
5. 配置 TIM2_SMCR 寄存器中的 TS=’110’，选定 TI2 作为触发输入源。
6. 设置 TIM2_CR1 寄存器的 CEN=’1’，启动计数器。
7. 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。

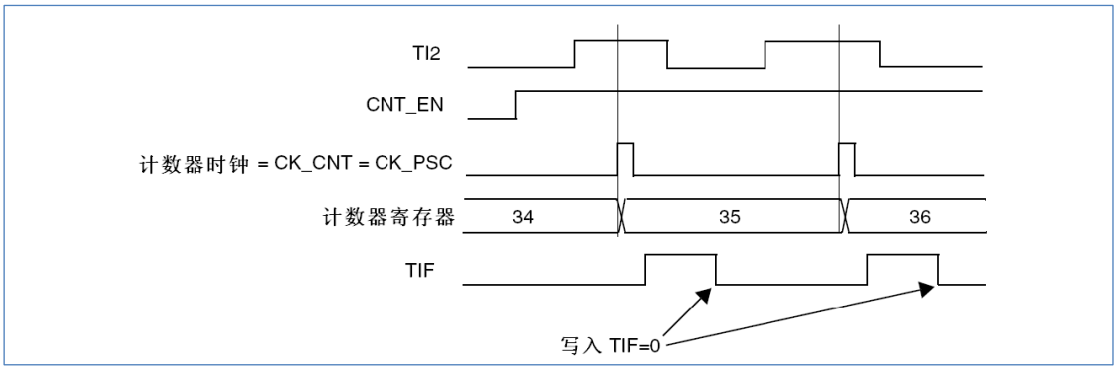

图 13-23 外部时钟模式 1 下的控制电路

外部时钟源模式 2

选定此模式的方法为：令 TIM2_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。

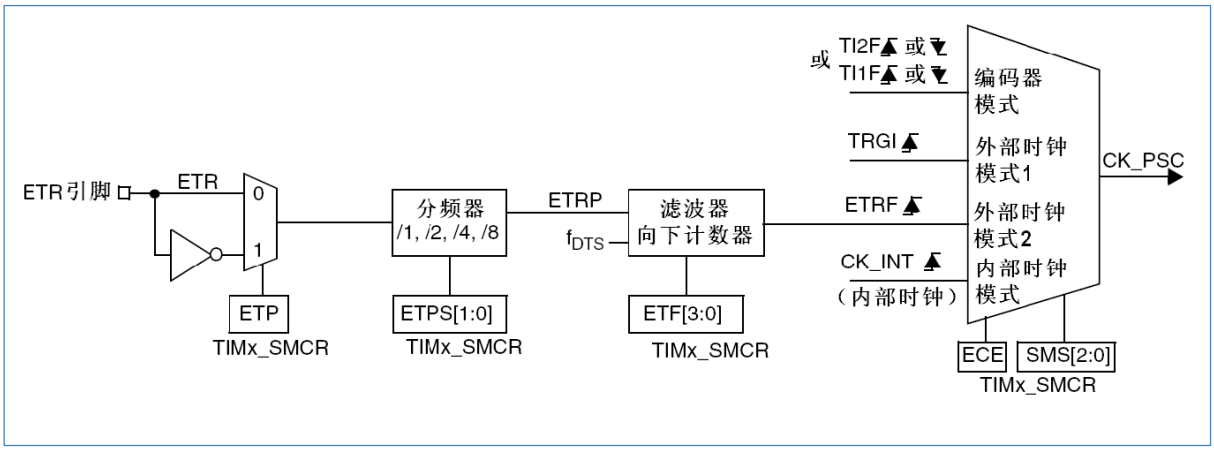

图 13-24 外部触发输入框图

例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：

1. 本例中不需要滤波器，置 TIM2_SMCR 寄存器中的 ETF[3:0]=0000。
2. 设置预分频器，置 TIM2_SMCR 寄存器中的 ETPS[1:0]=01。
3. 设置在 ETR 的上升沿检测，置 TIM2_SMCR 寄存器中的 ETP=0。
4. 开启外部时钟模式 2，置 TIM2_SMCR 寄存器中的 ECE=1。
5. 启动计数器，置 TIM2_CR1 寄存器中的 CEN=1。

计数器在每 2 个 ETR 上升沿计数一次。在 ETR 的上升沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。

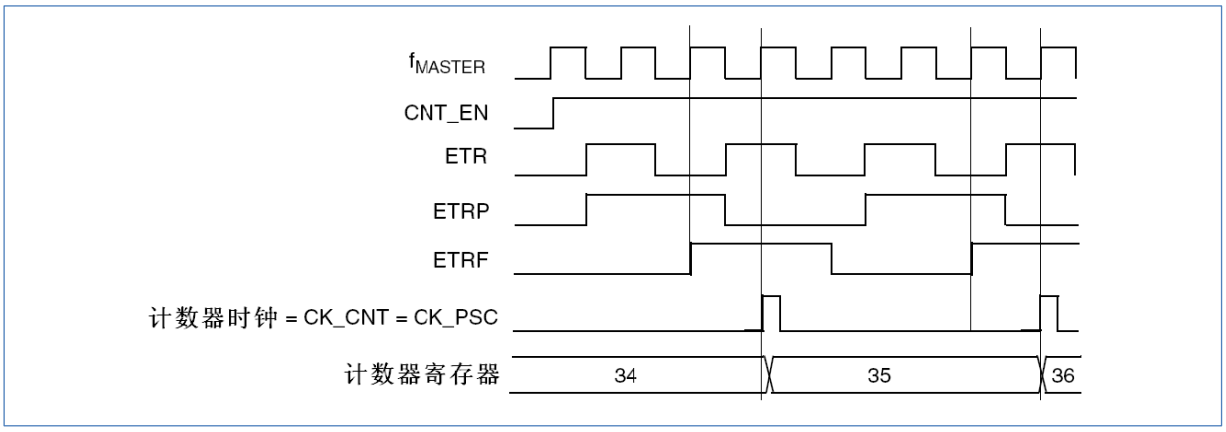

图 13-25 外部时钟模式 2 下的控制电路

13.2.4 捕获/比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器（包含影子寄存器），包括捕获的输入部分（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）。

下面几张图是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边沿检测器产生一个信号（TIxFPx），它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器（ICxPS）。

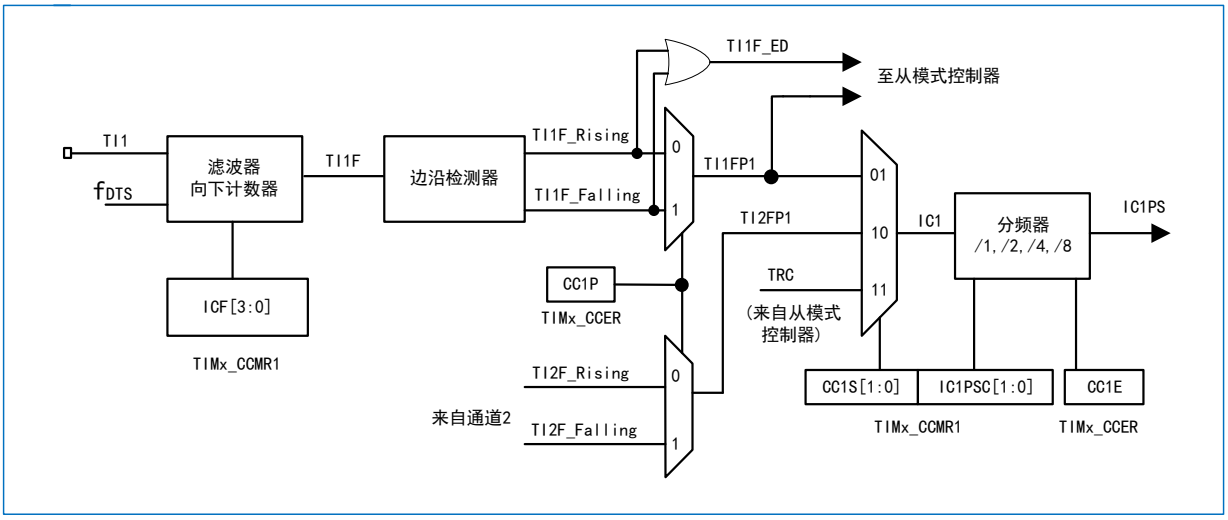

图 13-26 捕获/比较通道（如：通道 1 输入部分）

输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

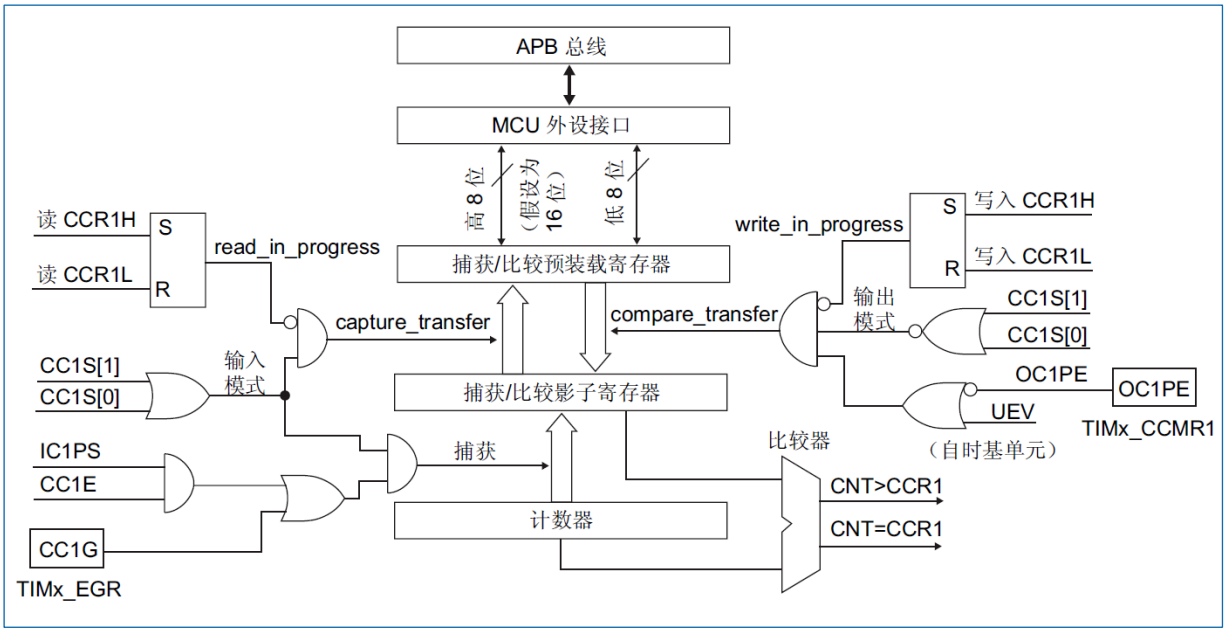

图 13-27 捕获/比较通道 1 的主电路

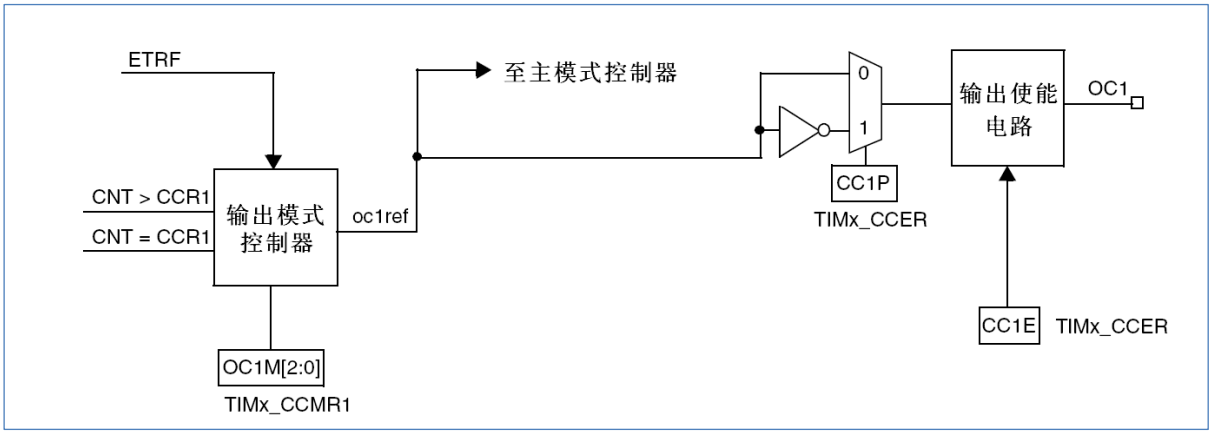

图 13-28 捕获/比较通道的输出部分（通道 1）

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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

13.2.5 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（TIM2_CCRx）中。当捕获事件发生时，相应的 CCxIF 标志（TIM2_SR 寄存器）被置'1'，如果使能了中断，则将产生中断。如果捕获事件发生时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIM2_SR 寄存器）被置'1'。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIM2_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 TIM2_CCR1 寄存器中，步骤如下：

1. 选择有效输入端：TIM2_CCR1 必须连接到 TI1 输入，所以写入 TIM2_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为'00'，通道将被配置为输入并且 TM1_CCR1 寄存器变为只读。
2. 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIM2_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽大于 5 个时钟周期。因此我们可以（以 f_{DTS} 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIM2_CCMR1 寄存器中写入 IC1F=0011。
3. 选择 TI1 通道的有效转换边沿，在 TIM2_CCER 寄存器中写入 CC1P=0（上升沿）。
4. 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TIM2_CCMR1 寄存器的 IC1PSC=00）。
5. 设置 TIM2_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
6. 如果需要，通过设置 TIM2_DIER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM2_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置'1'。
- 如设置了 CC1IE 位，则会产生一个中断。

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

说明：设置 TIM2_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断。

13.2.6 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- 两个 ICx 信号被映射至同一个 TIx 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的长度（TIM2_CCR1 寄存器）和占空比（TIM2_CCR2 寄存器），具体步骤如下（取决于 CK_INT 的频率和预分频器的值）。

1. 选择 TIM2_CCR1 的有效输入：置 TIM2_CCMR1 寄存器的 CC1S=01（选择 TI1）。
2. 选择 TI1FP1 的有效极性（用来捕获数据到 TIM2_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。

3. 选择 TIM2_CCR2 的有效输入：置 TIM2_CCMR1 寄存器的 CC2S=10（选择 TI1）。
4. 选择 TI1FP2 的有效极性（捕获数据到 TIM2_CCR2）：置 CC2P=1（下降沿有效）。
5. 选择有效的触发输入信号：置 TIM2_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 TIM2_SMCR 中的 SMS=100。
7. 使能捕获：置 TIM2_CCER 寄存器中 CC1E=1 且 CC2E=1。

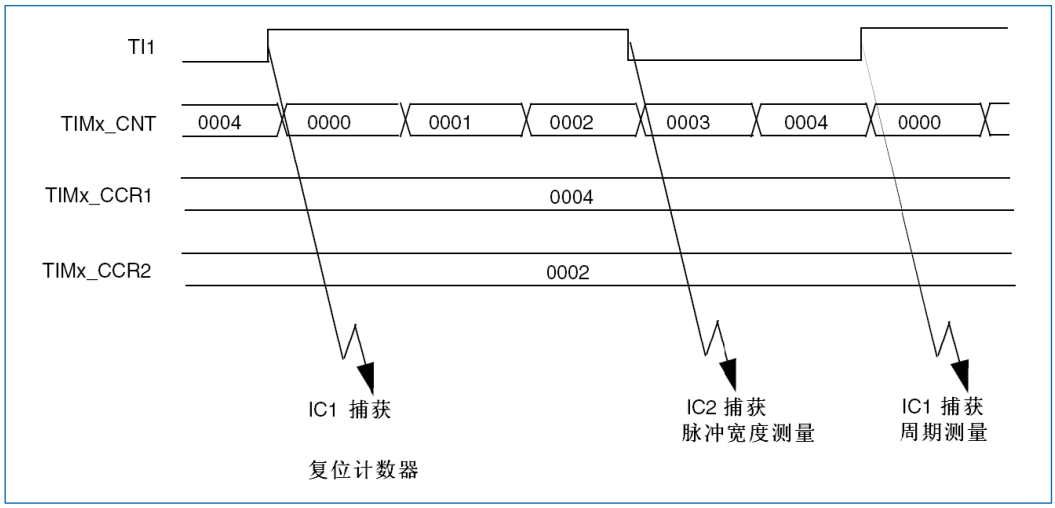

图 13-29 PWM 输入模式时序

由于只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIM2_CH1/TIM2_CH2 信号。

13.2.7 强置输出模式

在输出模式（TIM2_CCMRx 寄存器中 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx）能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 TIM2_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性位相反的值。

例如：CCxP=0（OCx 高电平有效），则 OCx 被强置为高电平。

置 TIM2_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIM2_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断。这将会在下面的输出比较模式一节中介绍。

13.2.8 输出比较模式

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

将输出比较模式（TIM2_CCMRx 寄存器中的 OCxM 位）和输出极性（TIM2_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（TIM2_SR 寄存器中的 CCxIF 位）。
- 若设置了相应的中断屏蔽（TIM2_DIER 寄存器中的 CCxIE 位），则产生一个中断。

TIM2_CCMRx 中的 OCxPE 位选择 TIM2_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 将相应的数据写入 TIM2_ARR 和 TIM2_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，例如当计数器 CNT 与 CCRx 匹配时翻转 OCx 的输出引脚，CCRx 预装载未用，开启 OCx 输出且高电平有效，则必须设置 OCxM=’011’、OCxPE=‘0’、CCxP=‘0’ 和 CCxE=‘1’。
5. 设置 TIM2_CR1 寄存器的 CEN 位启动计数器。

TIM2_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 TIM2_CCRx 影子寄存器只能在发生下一次更新事件时被更新）。下图给出了一个例子。

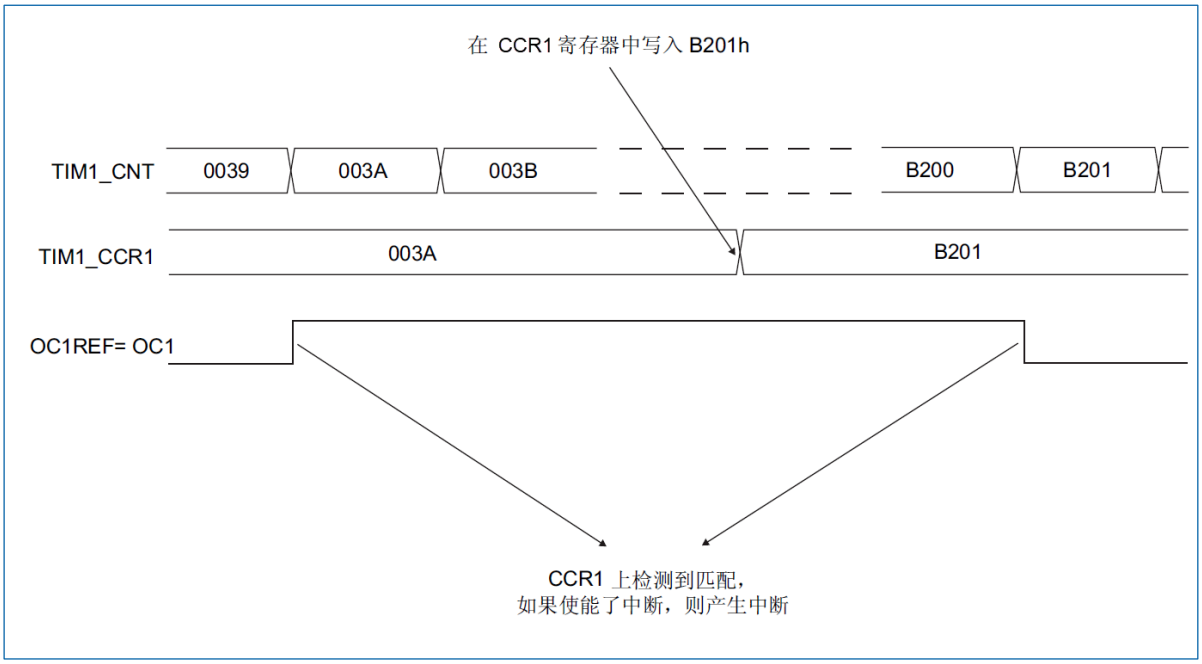

图 13-30 输出比较模式，翻转 0C1

13.2.9 PWM 模式

脉冲宽度调制模式可以产生一个由 TIM2_ARR 寄存器确定频率、由 TIM2_CCRx 寄存器确定占空比的信号。

在 TIM2_CCMRx 寄存器中的 OCxM 位写入’110’（PWM 模式 1）或’111’（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须设置 TIM2_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 TIM2_CR1 寄存器的 ARPE 位，（在向上计数或中央对齐模式中）使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIM2_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 TIM2_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。TIM2_CCER 寄存器中的 CCxE 位控制 OCx 输出使能。参见 TIM2 捕捉/比较使能寄存器（TIM2_CCER）的描述。

在 PWM 模式（模式 1 或模式 2）下，TIM2_CNT 和 TIM2_CCRx 始终在进行比较，（依据计数器的计数方向）以确定是否符合 TIM2_CCRx≤TIM2_CNT 或者 TIM2_CNT≤TIM2_CCRx。然而为了与 OCREF_CLR 的功能（在下一个 PWM 周期之前，ETR 信号上的一个外部事件能够清除 OCxREF）一致，OCxREF 信号只能在下述条件下产生：

- 当比较的结果改变。
- 当输出比较模式（TIM2_CCMRx 寄存器中的 OCxM 位）从“冻结”（无比较，OCxM=’000’）切换到某个 PWM 模式（OCxM=’110’或’111’）。

这样在运行中可以通过软件强置 PWM 输出。

根据 TIM2_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。

13.2.9.1 PWM 边沿对齐模式

向上计数配置

当 TIM2_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参看章节计数器模式。

下面是一个 PWM 模式 1 的例子。当 TIM2_CNT<TIM2_CCRx 时 PWM 信号参考 OCxREF 为高，否则为低。如果 TIM2_CCRx 中的比较值大于自动重装载值（TIM2_ARR），则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。下图为 TIM2_ARR=8 时边沿对齐的 PWM 波形实例。

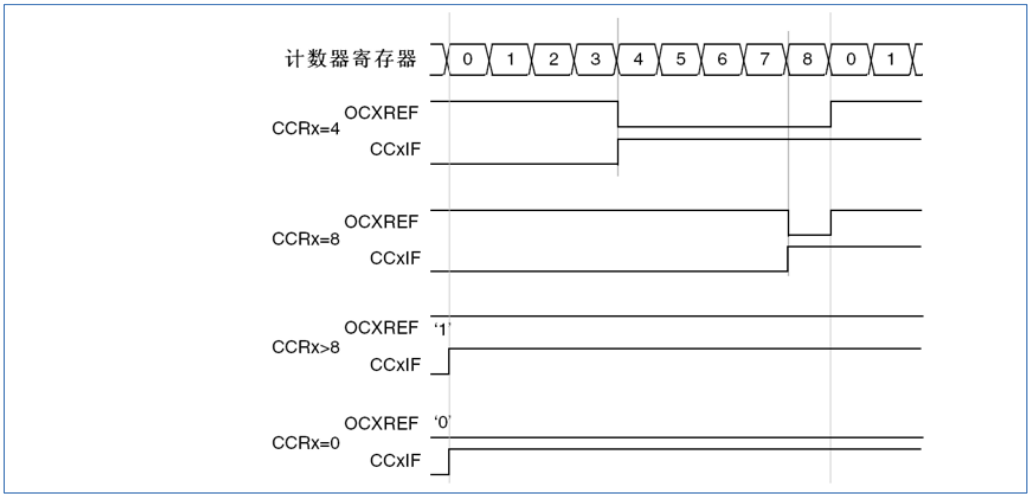

图 13-31 边沿对齐的 PWM 波形（ARR=8）

向下计数的配置

当 TIM2_CR1 寄存器的 DIR 位为高时执行向下计数。参见“13.2.2 计数器模式”。

在 PWM 模式 1，当 TIM2_CNT>TIM2_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIM2_CCRx 中的比较值大于 TIM2_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。

13.2.9.2 PWM 中央对齐模式

当 TIM2_CR1 寄存器中的 CMS 位不为’00’时，为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置’1’、在计数器向下计数时被置’1’、或在计数器向上和向下计数时被置’1’。TIM2_CR1 寄存器中的计数方向位（DIR）由硬件更新，不用软件修改。参见“13.2.2 计数器模式”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。

- TIM2_ARR=8
- PWM 模式 1

- TIM2_CR1 寄存器中的 CMS=01，在中央对齐模式 1 时，当计数器向下计数时设置比较标志。

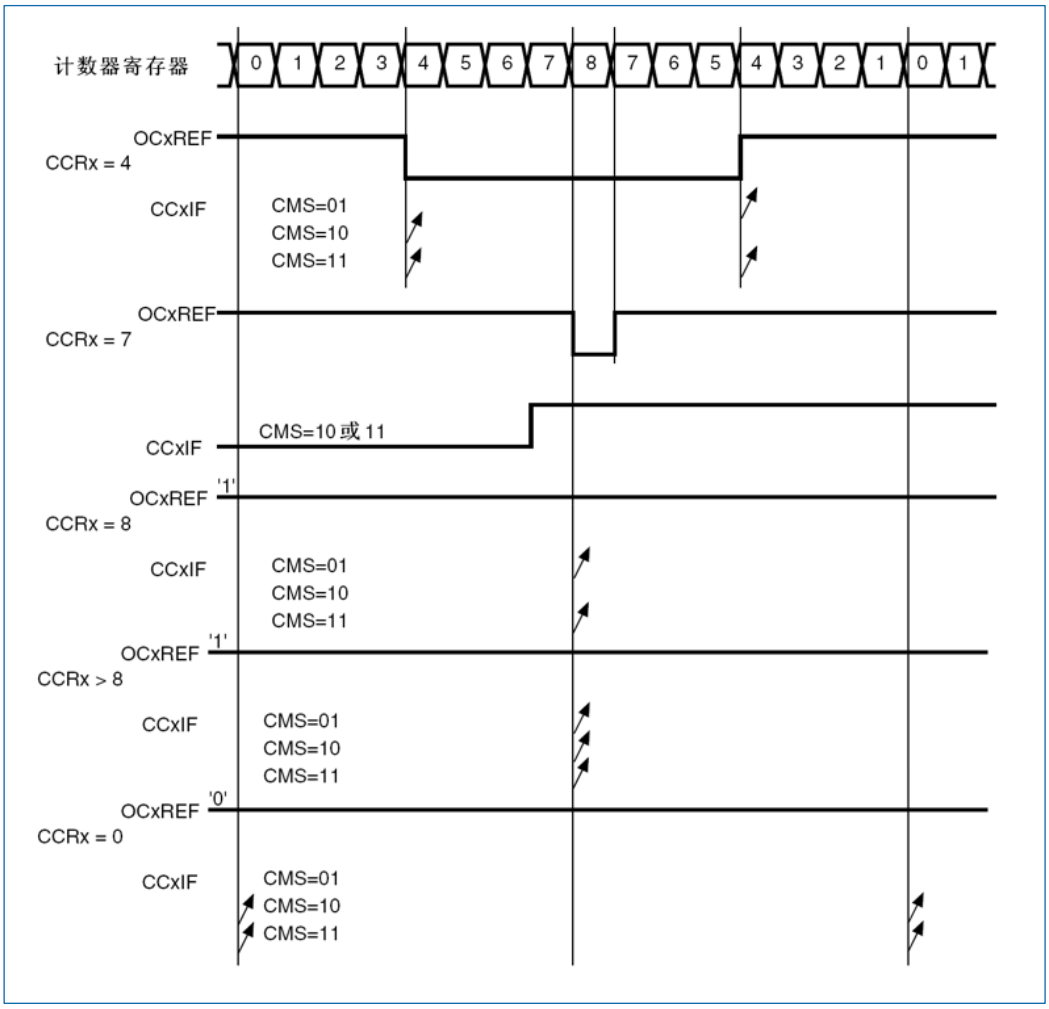

图 13-32 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIM2_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。特别是：
  - 如果写入计数器的值大于自动重加载的值（TIM2_CNT>TIM2_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 TIM2_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置 TIM2_EGR 位中的 UG 位），不要在计数进行时修改计数器的值。

13.2.10 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 TIM2_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：CNT<CCRx≤ARR（特别地，0<CCRx）

- 向下计数方式：CNT>CCRx

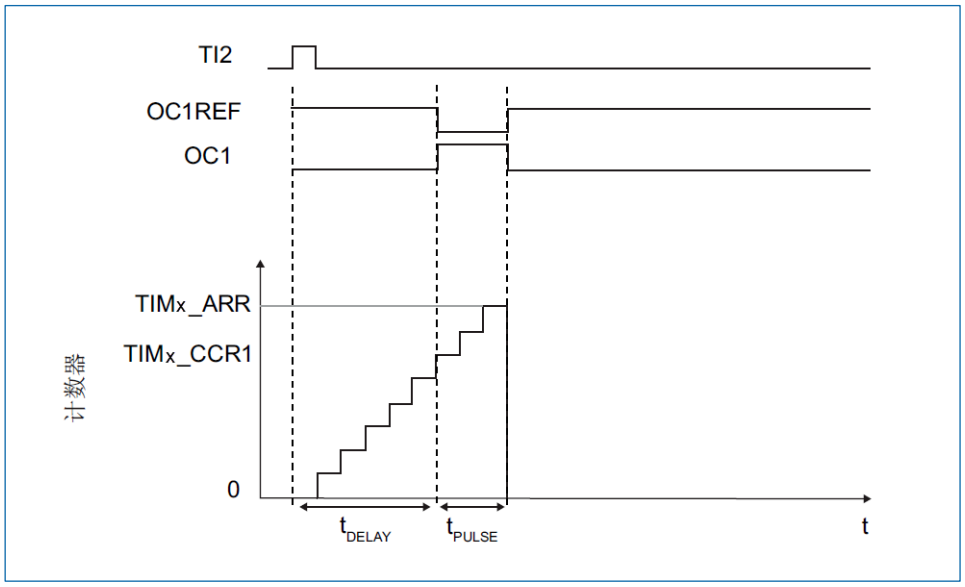

图 13-33 单脉冲模式的例子

例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

假定 TI2FP2 作为触发 1：

1. 置 TIM2_CCMR1 寄存器中的 CC2S='01'，把 TI2FP2 映射到 TI2。
2. 置 TIM2_CCER 寄存器中的 CC2P=‘0’，使 TI2FP2 能够检测上升沿。
3. 置 TIM2_SMCR 寄存器中的 TS='110'，TI2FP2 作为从模式控制器的触发（TRGI）。
4. 置 TIM2_SMCR 寄存器中的 SMS='110'（触发模式），TI2FP2 被用来启动计数器。

OPM 波形由写入比较寄存器的数值决定（要考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIM2_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义（TIM2_ARR-TIM2_CCR1）。
- 假定当发生比较匹配时要产生从'0'到'1'的波形，当计数器到达预装载值时要产生一个从'1'到'0'的波形；首先要置 TIM2_CCMR1 寄存器的 OC1M='111'，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIM2_CCMR1 中的 OC1PE=‘1’和 TIM2_CR1 寄存器中的 ARPE；然后在 TIM2_CCR1 寄存器中填写比较值，在 TIM2_ARR 寄存器中填写自动装载值，修改 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=‘0’。

在这个例子中，TIM2_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需一个脉冲，所以必须设置 TIM2_CR1 寄存器中的 OPM=‘1’，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

特殊情况：OCx 快速使能：

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 t_{DELAY}。

如果要以最小延时输出波形，可以设置 TIM2_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF（和 OCx）被强制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。

13.2.11 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIM2_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 信号可以连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：TIM2_SMCR 寄存器中的 ETPS[1:0]='00'。
2. 必须禁止外部时钟模式 2：TIM2_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 TIM2 被置于 PWM 模式。

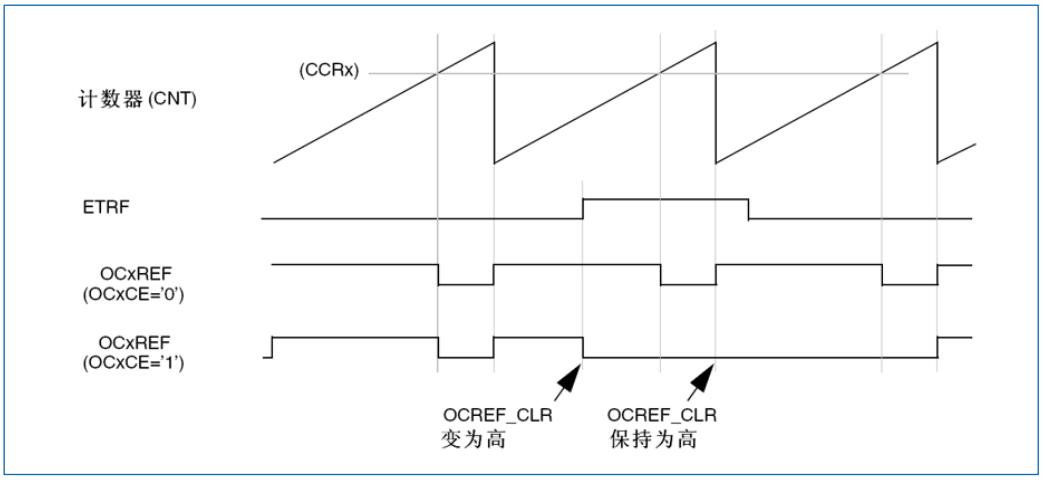

图 13-34 清除 TIM2 的 OCxREF

13.2.12 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 TIM2_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 TIM2_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。参见表 13-2，假定计数器已经启动（TIM2_CR1 寄存器中的 CEN= '1'），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 TIM2_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 TIM2_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 TIM2_ARR；同样，捕获器、比较器、预分频器、触发输出特性等仍工作如常。

在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2

不同时变换。

表 13-2 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）	有效边沿在 TI1 上时，使用 TI1FP1 信号计数		有效边沿在 TI2 上时，使用 TI2FP2 信号计数	
		TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	TI2FP2 为高	向下计数	向上计数	不计数	不计数
	TI2FP2 为低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	TI1FP1 为高	不计数	不计数	向上计数	向下计数
	TI1FP1 为低	不计数	不计数	向下计数	向上计数
在 TI1（和 TI2）上计数	TI2FP2（和 TI1FP1）为高	向下计数	向上计数	向上计数	向下计数
	TI2FP2（和 TI1FP1）为低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差分输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

- CC1S=’01’（TIM2_CCMR1 寄存器，IC1FP1 映射到 TI1）
- CC2S=’01’（TIM2_CCMR2 寄存器，IC2FP2 映射到 TI2）
- CC1P=‘0’（TIM2_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1）
- CC2P=‘0’（TIM2_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2）
- SMS=’011’（TIM2_SMCR 寄存器，所有的输入均在上升沿和下降沿有效）.
- CEN=‘1’（TIM2_CR1 寄存器，计数器使能）

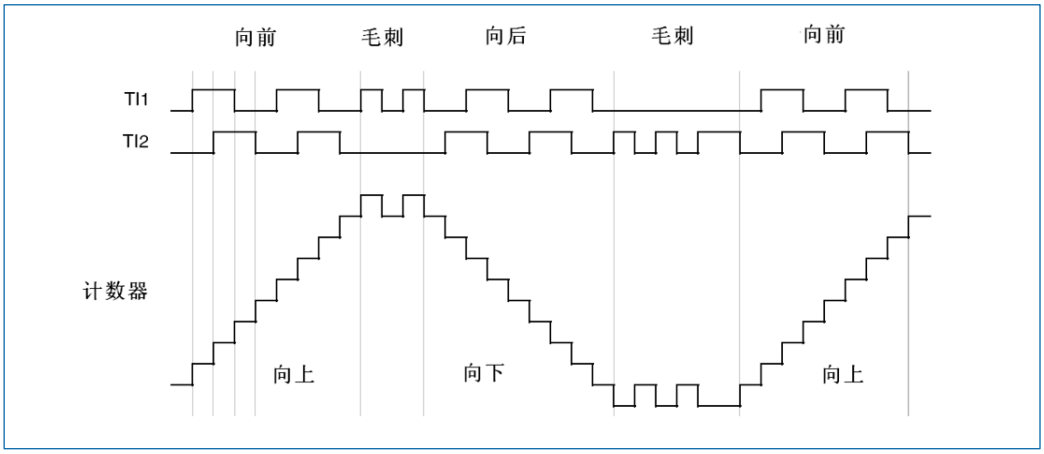

图 13-35 编码器模式下的计数器操作实例

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）。

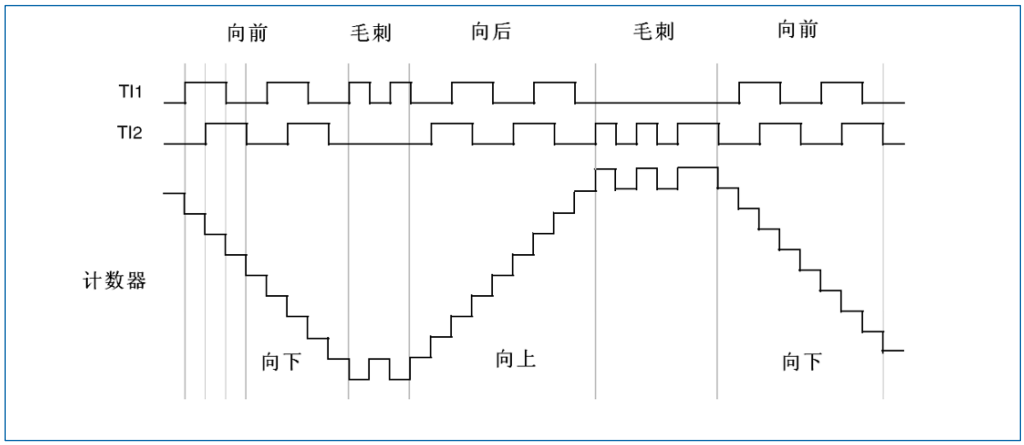

图 13-36 IC1FP1 反相的编码器接口模式实例

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）。

13.2.13 定时器输入异或功能

TIM2_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 TIM2_CH1、TIM2_CH2 和 TIM2_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。

13.2.14 定时器和外部触发的同步

TIM2 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。

13.2.14.1 从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIM2_CR1 寄存器的 URS 位为低，还会产生一个更新事件 UEV；然后所有的预装载寄存器（TIM2_ARR，TIM2_CCRx）都会被更新。

在下面的例子中，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，所以不需要配置它。CC1S 位只选择输入捕获源，即 TIM2_CCMR1 寄存器中 CC1S=01。置 TIM2_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。
- 置 TIM2_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM2_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM2_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（TIM2_SR 寄存器中的 TIF 位）被设置，根据 TIM2_DIER 寄存器中 TIE（中断使能）位的设置，产生一个中断请求。

下图显示当自动重装载寄存器 TIM2_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。

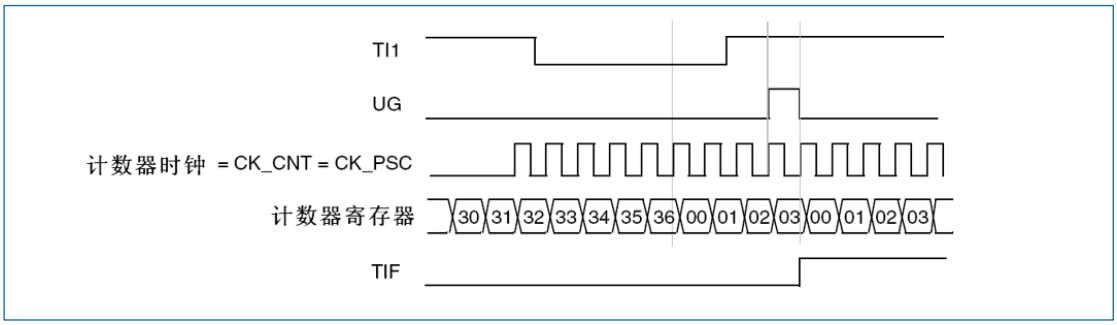

图 13-37 复位模式下的控制电路

13.2.14.2 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIM2_CCMR1 寄存器中 CC1S=01。置 TIM2_CCER 寄存器中 CC1P=1 以确定极性（只检测低电平）。
- 置 TIM2_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM2_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM2_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，在 TI1 变高时停止计数。当计数器开始或停止时都设置 TIM2_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。

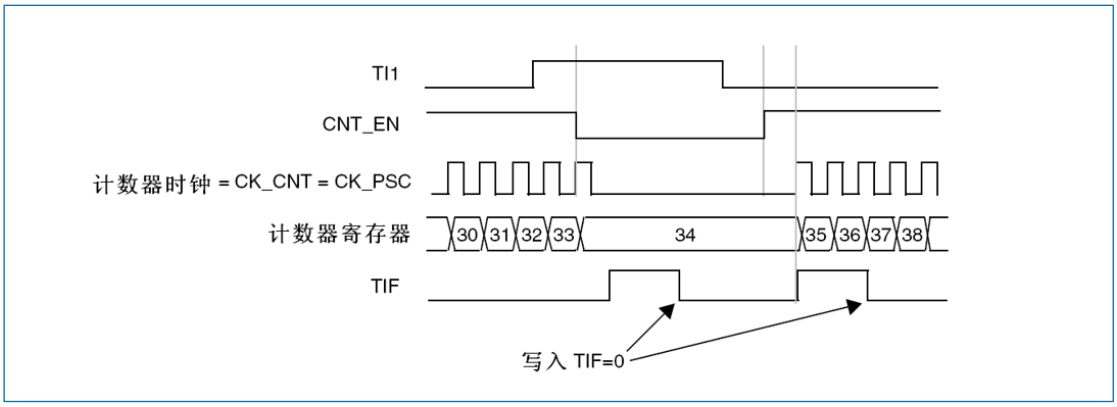

图 13-38 门控模式下的控制电路

13.2.14.3 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIM2_CCMR1 寄存器中 CC2S=01。置 TIM2_CCER 寄存器中 CC2P=1 以确定极性（只检测低电平）。
- 置 TIM2_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM2_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。

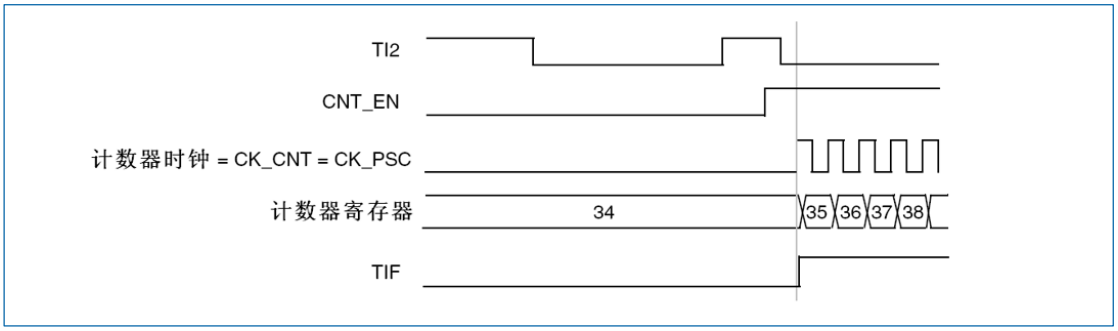

图 13-39 触发器模式下的控制电路

13.2.14.4 从模式：外部时钟模式 2＋触发模式

外部时钟模式 2 可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式时可以选择另一个输入作为触发输入。不建议使用 TIM2_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

下面的例子中，TI1 上出现一个上升沿之后，计数器即在 ETR 的每一个上升沿向上计数一次：

- 通过 TIM2_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。
- 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 TIM2_CCMR1 寄存器中 CC1S=01，选择输入捕获源。
  - 置 TIM2_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。
- 置 TIM2_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIM2_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。

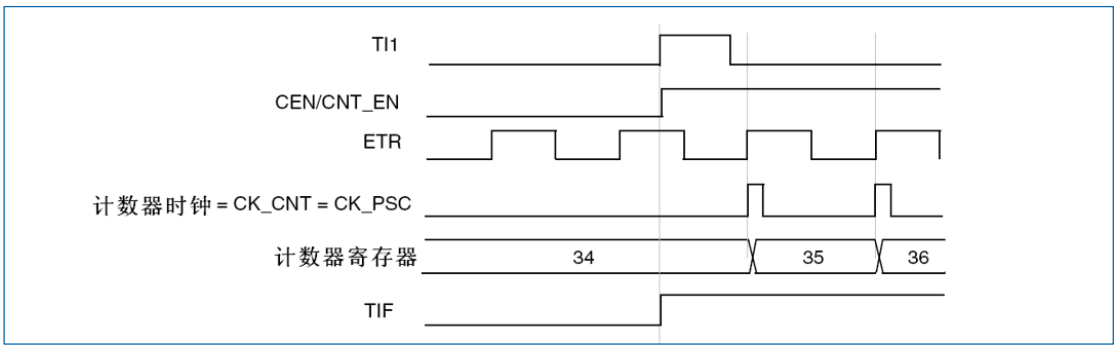

图 13-40 外部时钟模式 2＋触发模式下的控制电路

13.2.15 定时器同步

所有 TIMx 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另

一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况。

13.2.15.1 使用一个定时器作为另一个定时器的预分频器

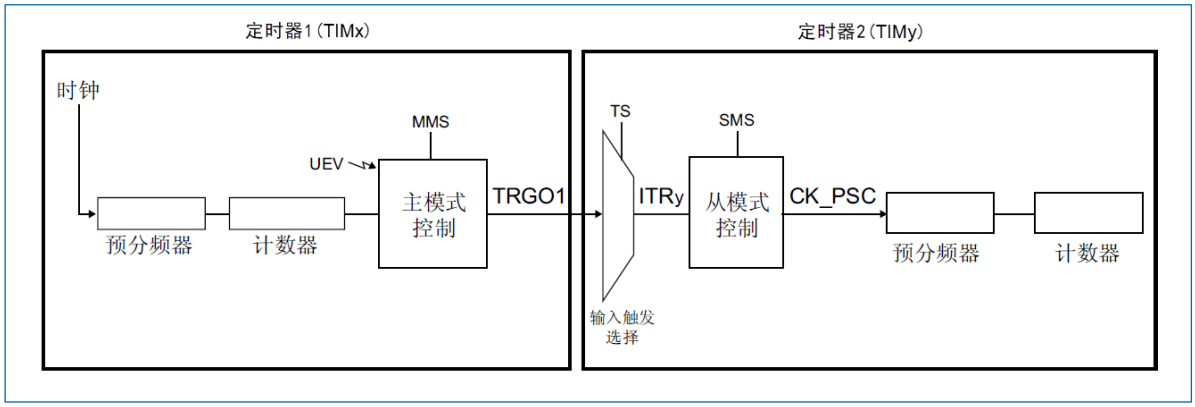

图 13-41 主/从定时器的例子

如：可以配置定时器 1（TIMx）作为定时器 2（TIMy）的预分频器。参考上图，进行下述操作：

1. 配置 TIMx 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 TIMx_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。
2. 连接 TIMx 的 TRGO1 输出至 TIMy，设置 TIMy_SMCR 寄存器的 TS 位，配置 TIMy 使用相应的 ITRy 作为内部触发的从模式。
3. 然后把从模式控制器置于外部时钟模式 1（TIMy_SMCR 寄存器的 SMS=111）；这样 TIMy 即可由 TIMx 周期性的上升沿（即 TIMx 的计数器溢出）信号驱动。
4. 最后，必须设置相应（TIMy 的 TIMy_CR1 寄存器）的 CEN 位分别启动两个定时器。

说明：如果 OCx 已被选中为定时器 1 TIMx 的触发输出（MMS=1xx），它的上升沿用于驱动定时器 2 TIMy 的计数器。

13.2.15.2 使用一个定时器使能另一个定时器

在这个例子中，定时器 2（TIMy）的使能由定时器 1（TIMx）的输出比较控制。参考图主/从定时器的例子的连接。仅当 TIMx 的 OC1REF 为高时，TIMy 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。

1. 配置 TIMx 为主模式，送出它的输出比较参考信号（OC1REF）为触发输出（TIMx 的 TIMx_CR2 寄存器的 MMS=100）。
2. 配置 TIMx 的 OC1REF 波形（TIMx 的 TIMx_CCMR1 寄存器）。
3. 配置 TIMy 从 TIMx 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=000）。
4. 配置 TIMy 为门控模式（TIMy_SMCR 寄存器的 SMS=101）。
5. 置 TIMy 的 TIMy_CR1 寄存器的 CEN=1 以使能 TIMy。
6. 置 TIMx 的 TIMx_CR1 寄存器的 CEN=1 以启动 TIMx。

说明：定时器 2 TIMy 的时钟不与定时器 1 TIMx 的时钟同步，这个模式只影响定时器 2 TIMy 计数器的使能信号。

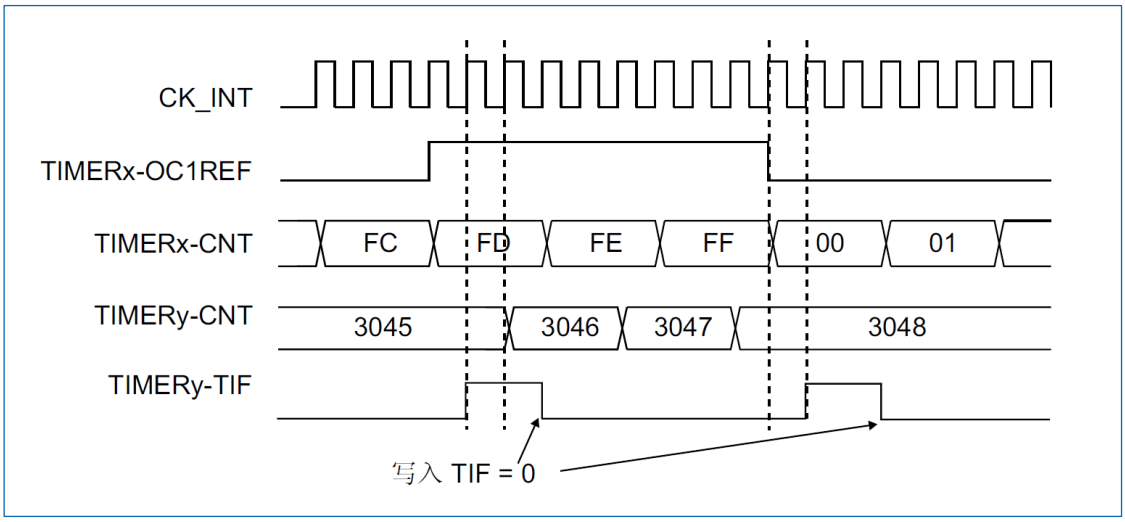

图 13-42 定时器 1（TIMx）的 OC1REF 控制定时器 2（TIMy）

在上图的例子中，在定时器 2（TIMy）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（TIMx）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 TIMy_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（TIMx）和定时器 2（TIMy）。TIMx 是主模式并从 0 开始，TIMy 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 TIMx 的 TIMx_CR1 的 CEN 位将禁止 TIMx，TIMy 随即停止。

1. 配置 TIMx 为主模式，送出输出比较 1 参考信号（OC1REF）做为触发输出（TIMx_CR2 寄存器的 MMS=100）。
2. 配置 TIMx 的 OC1REF 波形（TIMx 的 TIMx_CCMR1 寄存器）。
3. 配置 TIMy 从 TIMx 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=000）。
4. 配置 TIMy 为门控模式（TIMy_SMCR 寄存器的 SMS=101）。
5. 置 TIMx 的 TIMx_EGR 寄存器的 UG='1'，复位 TIMx。
6. 置 TIMy 的 TIMy_EGR 寄存器的 UG='1'，复位 TIMy。
7. 写'0xE7'至 TIMy 的计数器（TIMy_CNTL），初始化它为 0xE7。
8. 置 TIMy_CR1 寄存器的 CEN='1'以使能 TIMy。
9. 置 TIMx_CR1 寄存器的 CEN='1'以启动 TIMx。
10. 置 TIMx 的 TIMx _CR1 寄存器的 CEN='0'以停止 TIMx。

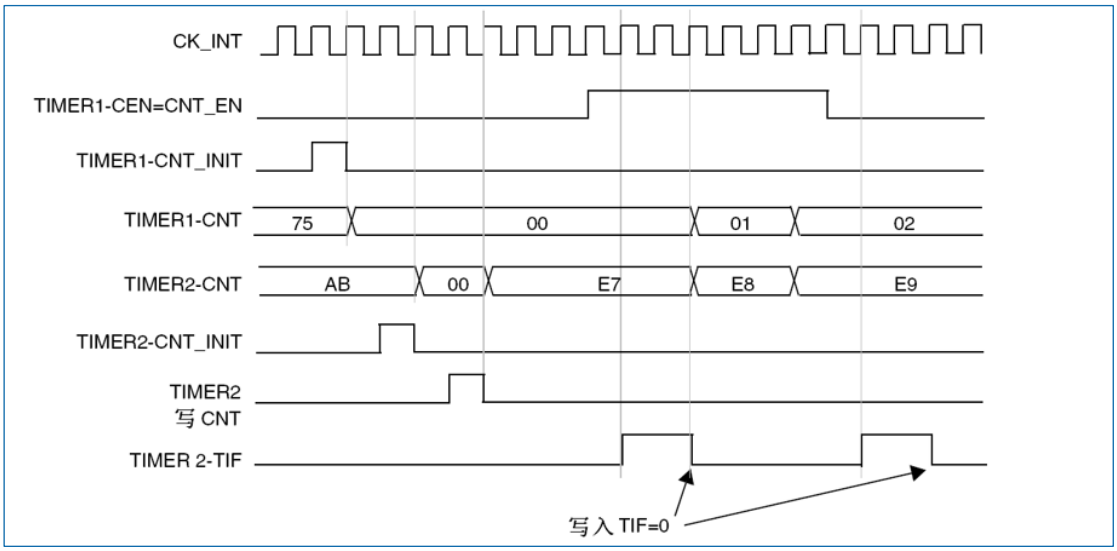

图 13-43 通过使能 TIMx 可以控制 TIMy

13.2.15.3 使用一个定时器去启动另一个定时器

在这个例子中，使用定时器 1（TIMx）的更新事件使能定时器 2（TIMy）。参考图主/从定时器的例子的连接。一旦 TIMx 产生更新事件，TIMy 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，TIMy 的 CEN 位被自动地置'1'，同时计数器开始计数直到写'0'到 TIMy_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（$f_{CK\_CNT}=f_{CK\_INT}/3$）。

1. 配置 TIMx 为主模式，送出它的更新事件（UEV）作为触发输出（TIMx_CR2 寄存器的 MMS=010）。
2. 配置 TIMx 的周期（TIMx 的 TIMx_ARR 寄存器）。
3. 配置 TIMy 从 TIMx 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=000）。
4. 配置 TIMy 为触发模式（TIMy_SMCR 寄存器的 SMS=110）。
5. 置 TIMx_CR1 寄存器的 CEN=1 以启动 TIMx。

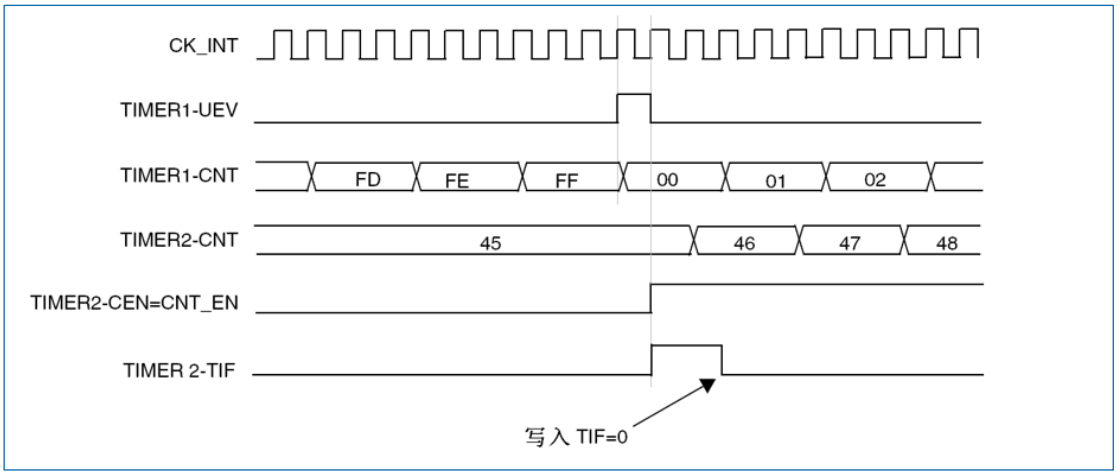

图 13-44 使用 TIMx 的更新触发 TIMy

在上一个例子中，可以在启动计数之前初始化两个计数器。下图显示在与 0 相同配置情况下，使用触发模式而不是门控模式（TIMy 的 TIMy_SMCR 寄存器的 SMS=110）的动作。

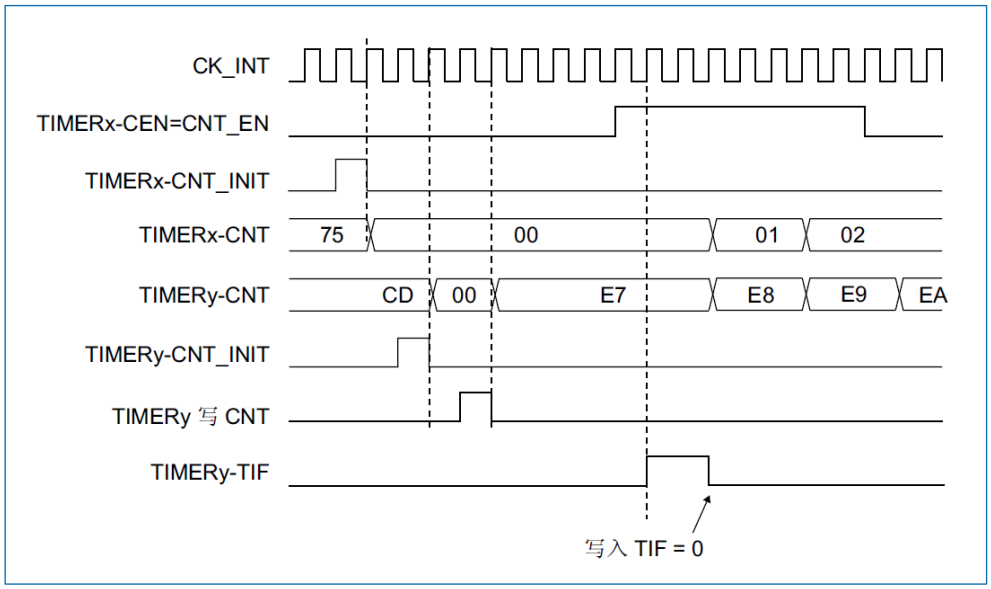

图 13-45 利用 TIMx 的使能触发 TIMy

13.2.15.4 使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1（TIMx）的 TI1 输入上升沿时使能 TIMx，使能 TIMx 的同时使能定时器 2（TIMy），参见图 13-41。为保证计数器的对齐，TIMx 必须配置为主/从模式（对应 TI1 为从，对应 TIMy 为主）：

1. 配置 TIMx 为主模式，送出它的使能作为触发输出（TIMx_CR2 寄存器的 MMS=’001’）。
2. 配置 TIMx 为从模式，从 TI1 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=’100’）。
3. 配置 TIMx 为触发模式（TIMx 的 TIMx_SMCR 寄存器的 SMS=’110’）。
4. 配置 TIMx 为主/从模式，TIMx 的 TIMx_SMCR 寄存器的 MSM=‘1’。
5. 配置 TIMy 从 TIMx 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=000）。
6. 配置 TIMy 为触发模式（TIMy 的 TIMy_SMCR 寄存器的 SMS=’110’）。

当 TIMx 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。

说明：在这个例子中，在启动之前两个定时器都被初始化（设置相应的 UG 位），两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器（TIM2_CNT）在定时器间插入一个偏移。下图中能看到主/从模式下在 TIMx 的 CNT_EN 和 CK_PSC 之间有个延迟。

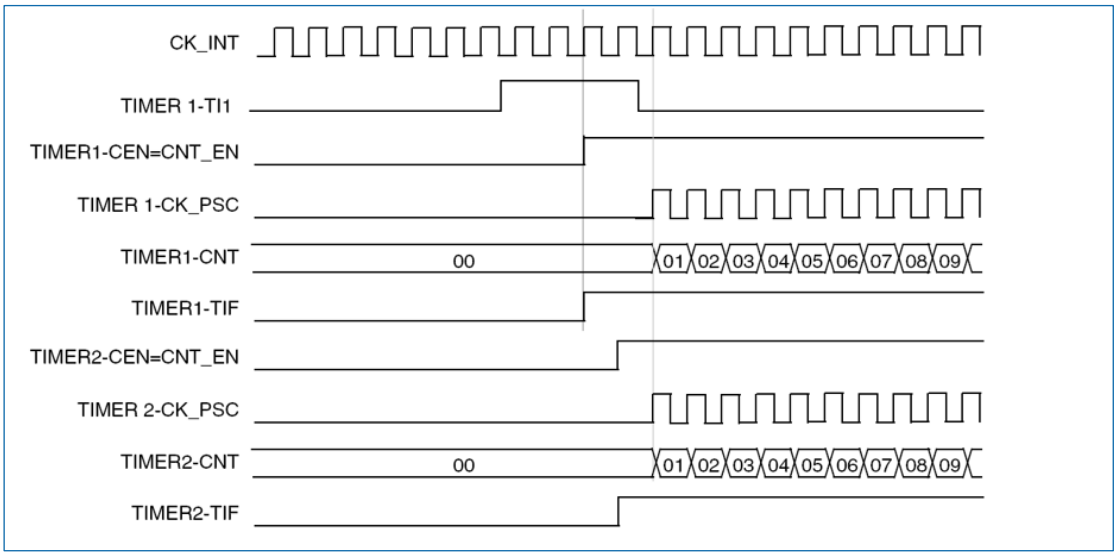

图 13-46 使用定时器 1（TIMx）的 TI1 输入触发定时器 1（TIMx）和定时器 2（TIMy）

13.2.16 调试模式

当 MCU 进入调试模式（Cortex-M0 内核停止），根据 DBG 模块中 DBG_TIM2_STOP 的设置，TIM2 计数器继续正常操作或者停止。参见“22.8.2 对定时器、看门狗和 I2C 的调试支持”。

13.3 TIM2 寄存器

基地址：0x4000 0000

空间大小：0x0400

13.3.1 TIM2 控制寄存器 1（TIM2_CR1）

偏移地址：0x00

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						rw		rw	rw		rw	rw	rw	rw	rw

位 15:10	Res：保留 必须保持复位值。
位 9:8	CKD[1:0]：时钟分频因子（Clock division） 这2 位定义在定时器时钟（CK_INT）频率与数字滤波器（ETR，TIx）使用的采样频率之间的分频比例。 • 00：$t_{DTS} = t_{CK\_INT}$ • 01：$t_{DTS} = 2 \times t_{CK\_INT}$ • 10：$t_{DTS} = 4 \times t_{CK\_INT}$ • 11：保留，未使用
位 7	ARPE：自动重装载预装载允许（Auto-reload preload enable） • 0：TIMx_ARR 寄存器没有缓冲 • 1：TIMx_ARR 寄存器有缓冲

位 6:5	CMS[1:0]：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、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、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、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。
位 4	DIR：方向（Direction） • 0：计数器向上计数 • 1：计数器向下计数
位 3	OPM：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。
位 2	URS：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请求，则下述任一事件将产生更新中断请求： o 计数器溢出/下溢 o 设置 UG 位 o 从模式控制器产生的更新 • 1：如果使能了更新中断请求，则只有计数器溢出/下溢才产生更新中断请求。
位 1	UDIS：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： o 计数器溢出/下溢 o 设置 UG 位 o 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 位或从模式控制器产出了一个硬件复位，则计数器和预分频器被重新初始化。
位 0	CEN：使能计数器（Counter enable）

	• 0：禁止计数器 • 1：使能计数器

13.3.2 TIM2 控制寄存器 2（TIM2_CR2）

偏移地址：0x04

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								TI1S	MMS[2:0]			Res			
								rw	rw						

位 15:8	Res：保留 必须保持复位值。
位 7	TI1S：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。
位 6:4	MMS[2:0]：主模式选择（Master mode selection） 该位域可选择在主模式下送到从定时器的同步信息（TRGO）。 • 000：复位 TIMx_EGR 寄存器的 UG 位被用于作为触发输出（TRGO）。如果是触发输入产生的复位（从模式控制器处于复位模式），则 TRGO 上的信号相对实际的复位会有一个延迟。 • 001：使能 计数器使能信号 CNT_EN 被作为触发输出（TRGO）。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式。 • 010：更新 更新事件被选为触发输入（TRGO）。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 • 011：比较脉冲 在发生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 信号被用作触发输出（TRGO）。 • 101：比较 OC2REF 信号被用作触发输出（TRGO）。 • 110：比较 OC3REF 信号被用作触发输出（TRGO）。 • 111：比较 OC4REF 信号被用作触发输出（TRGO）。

位 3:0	Res：保留 必须保持复位值。 读该位始终为 0。

13.3.3 TIM2 从模式控制寄存器（TIM2_SMCR）

偏移地址：0x08

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Res	SMS[2:0]		
rw	rw	rw		rw				rw	rw				rw		

位 15	ETP：外部触发极性（External trigger polarity） 该位选择 ETR 或 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。
位 14	ECE：外部时钟使能位（External clock enable） 该位启用外部时钟模式2。 • 0：禁止外部时钟模式2 • 1：使能外部时钟模式2 计数器由 ETRF 信号的任意有效边沿驱动。
位 13:12	ETPS[1:0]：外部触发预分频（External trigger prescaler） 外部触发信号 ETRP 的频率最多是 CK_INT 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/2 • 10：ETRP 频率/4 • 11：ETRP 频率/8
位 11:8	ETF[3:0]：外部触发滤波（External trigger filter） 该位域定义了采样 ETRP 信号的频率和对 ETRP 信号数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6

	• 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 7	MSM：主/从模式（Master/Slave mode） • 0：无作用 • 1：触发输入（TRGI）上的事件被延迟，以允许当前定时器（通过 TRGO）与它的从定时器之间的完美同步。这适用于多个定时器需要同步到一个单一的外部事件的情况。
位 6:4	TS[2:0]：触发选择（Trigger selection） 该位域选择用于同步计数器的触发输入。 • 000：内部触发0（ITR0） • 001：内部触发1（ITR1） • 010：内部触发2（ITR2） • 011：保留 • 100：TI1 的边沿检测器（TI1F_ED） • 101：滤波后的定时器输入1（TI1FP1） • 110：滤波后的定时器输入2（TI2FP2） • 111：外部触发输入（ETRF）
位 3	Res：保留 必须保持复位值。
位 2:0	SMS[2:0]：从模式选择（Slave mode selection） 当选择了外部信号，触发信号（TRGI）的有效边沿与选中的外部输入极性相关。 • 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 TI1FP1（和 TI2FP2）的电平，计数器在 TI2FP2（和 TI1FP1）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。

表 13-3 TIM2 内部触发连接

从定时器	ITR0（TS=000）	ITR1（TS=001）	ITR2（TS=010）
TIM2	TIM1	TIM6	ADC_AWD

13.3.4 TIM2 中断允许寄存器（TIM2_DIER）

偏移地址：0x0C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
									rw		rw	rw	rw	rw	rw

位 15:7	Res：保留 必须保持复位值。
位 6	TIE：触发中断使能（Trigger interrupt enable） • 0：触发中断禁用 • 1：触发中断允许
位 5	Res：保留 必须保持复位值。
位 4	CC4IE：捕捉/比较 4 中断使能（Capture/Compare 4 interrupt enable） • 0：CC4 中断禁用 • 1：CC4 中断允许
位 3	CC3IE：捕捉/比较 3 中断使能（Capture/Compare 3 interrupt enable） • 0：CC3 中断禁用 • 1：CC3 中断允许
位 2	CC2IE：捕捉/比较 2 中断使能（Capture/Compare 2 interrupt enable） • 0：CC2 中断禁用 • 1：CC2 中断允许
位 1	CC1IE：捕捉/比较 1 中断使能（Capture/Compare 1 interrupt enable）

	• 0：CC1 中断禁用 • 1：CC1 中断允许
位 0	UIE：更新中断使能（Update interrupt enable） • 0：更新中断禁用 • 1：更新中断允许

13.3.5 TIM2 状态寄存器（TIM2_SR）

偏移地址：0x10

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			CC4OF	CC3OF	CC2OF	CC1OF	Res		TIF	Res	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

位 15:13	Res：保留 必须保持复位值。
位 12	CC4OF：捕捉/比较4 重复捕捉标志（Capture/Compare 4 overcapture flag）
位 11	CC3OF：捕捉/比较3 重复捕捉标志（Capture/Compare 3 overcapture flag）
位 10	CC2OF：捕捉/比较2 重复捕捉标志（Capture/Compare 2 overcapture flag）
位 9	CC1OF：捕捉/比较1 重复捕捉标志（Capture/Compare 1 overcapture flag） 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。 该位由软件清 0。 • 0：无重复捕获产生。 • 1：当计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为 1。
位 8:7	Res：保留 必须保持复位值。
位 6	TIF：触发器中断标记（Trigger interrupt flag） 当发生触发事件（当从模式控制器处于除门控模式外的其他模式时，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。该位由软件清 0。 • 0：无触发器事件产生。 • 1：触发器中断等待响应。
位 5	Res：保留 必须保持复位值。
位 4	CC4IF：捕捉/比较 4 中断标志（Capture/Compare 4 interrupt flag）

位 3	CC3IF：捕捉/比较 3 中断标志（Capture/Compare 3 interrupt flag）
位 2	CC2IF：捕捉/比较 2 中断标志（Capture/Compare 2 interrupt flag）
位 1	CC1IF：捕捉/比较 1 中断标志（Capture/Compare 1 interrupt flag） • 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中央对齐模式下除外。该位由软件清 0。 ○ 0：无匹配发生 ○ 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配 当 TIMx_CCR1 > TIMx_APR 时，在计数器溢出（向上、向上/向下计数模式）或计数器下溢出时（向下计数模式）时 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。该位由软件清 0 或通过读 TIMx_CCR1 清 0。 ○ 0：无输入捕获产生； ○ 1：计数器值已被捕获（拷贝）至 TIMx_CCR1（在 IC1 上检测到与所选极性相同的边沿）。
位 0	UIF：更新中断标记（Update interrupt flag） • 0：无更新事件产生； • 1：更新中断等待响应。 当产生以下更新事件时，该位由硬件置 1。 ○ 若 TIMx_CR1 寄存器的 UDIS=0 且 URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件（软件对计数器 CNT 重新初始化）。 ○ 若 TIMx_CR1 寄存器的 UDIS=0 且 URS=0，当计数器 CNT 被触发事件重初始化时产生。 该位由软件清 0。

13.3.6 TIM2 事件产生寄存器（TIM2_EGR）

偏移地址：0x14

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TG	Res	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

位 15:7	Res：保留 必须保持复位值。
位 6	TG：产生触发事件（Trigger generation） 该位由软件置 1，用于产生一个触发事件。该位由硬件自动清 0。 • 0：无动作 • 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断，则产生相应的中断。

位 5	Res：保留 必须保持复位值。
位 4	CC4G：捕捉/比较 4 产生（Capture/Compare 4 generation）
位 3	CC3G：捕捉/比较 3 产生（Capture/Compare 3 generation）
位 2	CC2G：捕捉/比较 2 产生（Capture/Compare 2 generation）
位 1	CC1G：捕捉/比较 1 产生（Capture/Compare 1 generation） • 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断，则产生相应的中断。 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 配置为输入： 当前的计数器值捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断，则产生相应的中断。若 CC1IF 已经为 1，则设置 CC1OF=1。
位 0	UG：产生更新事件（Update generation） 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：若 UG=1，预分频器的计数器也被清 0（但是预分频系数不变）。若在中央对齐模式下或 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。*

13.3.7 TIM2 捕捉/比较模式寄存器 1（TIM2_CCMR1）

偏移地址：0x18

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]				IC1PSC[1:0]			
rw	rw			rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	

输出比较模式

位 15	OC2CE：使能输出比较 2 清除（Output compare 2 clear enable）
位 14:12	OC2M[2:0]：输出比较 2 模式（Output compare 2 mode）
位 11	OC2PE：输出比较 2 预装载使能（Output compare 2 preload enable）
位 10	OC2FE：输出比较 2 快速使能（Output compare 2 fast enable）

位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/Compare 2 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC2 通道配置为输出。 • 01：CC2 通道配置为输入，IC2 映射在 TI2 上。 • 10：CC2 通道配置为输入，IC2 映射在 TI1 上。 • 11：CC2 通道配置为输入，IC2 映射在 TRC 上。
位 7	OC1CE：输出比较 1 清除使能（Output compare 1 clear enable） • 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。
位 6:4	OC1M[2:0]：输出比较 1 模式（Output compare 1 mode） 该位域定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1 和 OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1PN 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 o 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 则通道 1 为有效电平，否则为无效电平。 o 在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 则通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。 • 111：PWM 模式 2 o 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。 o 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。
位 3	OC1PE：输出比较 1 预装载使能（Output compare 1 preload enable） • 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生效。

	• 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅针对预装载寄存器，TIMx_CCR1 的预装载值在更新事件到来时被传送至当前寄存器中。
位 2	OC1FE：输出比较 1 快速使能（Output compare 1 fast enable） 该位加快 CC 输出对触发器输入事件的响应。 • 0：CC1 的正常操作依赖于计数器与 CCR1 的值，即使触发器是打开的。当触发器的输入出现一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 • 1：输入到触发器的有效沿的作用就像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。该位只在通道被配置成 PWM1 或 PWM2 模式时生效。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare 1 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 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。

输入捕捉模式

位 15:12	IC2F[3:0]：输入捕捉 2 滤波器（Input capture 2 filter） 该位域定义了 TI2 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 11:10	IC2PSC[1:0]：输入捕捉 2 预分频（Input capture 2 prescaler）
位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/compare 2 selection）

	该位域定义通道的方向（输入/输出），及输入信号的选择： • 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。
位 7:4	IC1F[3:0]：输入捕获 1 滤波器（Input capture 1 filter） 该位域定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 3:2	IC1PSC[1:0]：输入捕捉 1 预分频（Input capture 1 prescaler） 该位域定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0（TIMx_CCER 寄存器中），则预分频器复位。 • 00：无预分频器，捕获输入口上检测到的每一个边沿都会触发一次捕获。 • 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。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/compare 1 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 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。

13.3.8 TIM2 捕捉/比较模式寄存器 2（TIM2_CCMR2）

偏移地址：0x1C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
rw	rw			rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	

输出比较模式

位 15	OC4CE：输出比较4 清除使能（Output compare 4 clear enable）
位 14:12	OC4M[2:0]：输出比较4 模式（Output compare 4 mode）
位 11	OC4PE：输出比较4 预装载使能（Output compare 4 preload enable）
位 10	OC4FE：输出比较4 快速使能（Output compare 4 fast enable）
位 9:8	CC4S[1:0]：捕捉/比较4 选择（Capture/Compare 4 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 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）。
位 7	OC3CE：输出比较3 清除使能（Output compare 3 clear enable）
位 6:4	OC3M[2:0]：输出比较3 模式（Output compare 3 mode）
位 3	OC3PE：输出比较3 预装载使能（Output compare 3 preload enable）
位 2	OC3FE：输出比较3 快速使能（Output compare 3 fast enable）
位 1:0	CC3S[1:0]：捕捉/比较3（Capture/Compare 3 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC3 通道被配置为输出。 • 01：CC3 通道被配置为输入，IC4 映射在 TI3 上。 • 10：CC3 通道被配置为输入，IC4 映射在 TI4 上。 • 11：CC3 通道被配置为输入，IC4 映射在 TRC 上。

输入捕捉模式

位 15:12	IC4F[3:0]：输入捕捉4 滤波器（Input capture 4 filter） 该位域定义了 TI4 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器

	组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 11:10	IC4PSC[1:0]：输入捕捉4 预分频器（Input capture 4 prescaler）
位 9:8	CC4S[1:0]：捕捉/比较4 选择（Capture/Compare 4 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 00：CC4 通道被配置为输出。 • 01：CC4 通道被配置为输入，IC3 映射在 TI4 上。 • 10：CC4 通道被配置为输入，IC3 映射在 TI3 上。 • 11：CC4 通道被配置为输入，IC3 映射在 TRC 上。
位 7:4	IC3F[3:0]：输入捕捉3 滤波器（Input capture 3 filter） 该位域定义了 TI3 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 • 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5

	• 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 3:2	IC3PSC[1:0]：输入比较3 预分频器（Input capture 3 prescaler）
位 1:0	CC3S[1:0]：捕捉/比较3 选择（Capture/Compare 3 selection） 该位域定义通道的方向（输入/输出），及输入信号的选择： • 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。

13.3.9 TIM2 捕捉/比较使能寄存器（TIM2_CCER）

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Res	CC4P	CC4E	CC3NP	Res	CC3P	CC3E	CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
rw		rw	rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

位 15	CC4NP：捕捉/比较 4 输出极性（Capture/Compare 4 output Polarity）
位 14	Res：保留 必须保持复位值。
位 13	CC4P：捕捉/比较 4 输出极性（Capture/Compare 4 output Polarity）
位 12	CC4E：捕捉/比较 4 输出使能（Capture/Compare 4 output enable）
位 11	CC3NP：捕捉/比较 3 输出极性（Capture/Compare 3 output Polarity）
位 10	Res：保留 始终保持复位值。
位 9	CC3P：捕捉/比较 3 输出极性（Capture/Compare 3 output Polarity）
位 8	CC3E：捕捉/比较 3 输出使能（Capture/Compare 3 output enable）
位 7	CC2NP：捕捉/比较 2 输出极性（Capture/Compare 2 output Polarity）
位 6	Res：保留

	必须保持复位值。
位 5	CC2P：捕捉/比较 2 输出极性（Capture/Compare 2 output Polarity）
位 4	CC2E：捕捉/比较 2 输出使能（Capture/Compare 2 output enable）
位 3	CC1NP：捕捉/比较 1 输出极性（Capture/Compare 1 output Polarity） • 通道 CC1 配置为输出时，CC1NP 必须清 0 并保持 CC1NP＝0。 • 通道 CC1 配置为输入时，CC1NP 与 CC1P 联合控制 TI1FP1/TI2FP1 的极性。
位 2	Res：保留 必须保持复位值。
位 1	CC1P：捕捉/比较 1 输出极性（Capture/Compare 1 output Polarity） • 通道 CC1 配置为输出: o 0：OC1 高有效 o 1：OC1 低有效 • 通道 CC1 配置为输入: CC1NP/CC1P 用于选择作为触发或捕获的信号 TI1FP1 和 TI2FP1 的极性是 IC1 还是 IC1 的反相信号。 o 00：不反相/上升沿 捕获发生在 TIxFP1 的上升沿（复位、外部时钟或触发模式的捕捉或触发），TIxFP1 不反相（在门控、编码器模式下的触发）。 o 01：反相/下降沿 捕获发生在 TIxFP1 的下降沿（复位、外部时钟或触发模式的捕捉或触发），TIxFP1 反相（在门控、编码器模式下的触发）。 o 10：保留，不使用此配置 o 11：不反相/上升和下降沿 捕获发生在 TIxFP1 的上升沿和下降沿（复位、外部时钟或触发模式的捕捉或触发），TIxFP1 不反相（门控模式的触发）。此配置不能用于编码器模式。
位 0	CC1E：捕捉/比较 1 输出使能（Capture/Compare 1 output enable） • CC1 通道配置为输出： o 0：关闭，OC1 禁止输出。 o 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。 o 0：捕获禁止 o 1：捕获使能

13.3.10 TIM2 计数器寄存器（TIM2_CNT）

偏移地址：0x24

复位值：0x0000

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

位 15:0	CNT[15:0]：低16 位值（Counter value）

13.3.11 TIM2 预分频寄存器（TIM2_PSC）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频值（Prescaler value） 计数器的时钟频率 CK_CNT 等于 f_{CK_PSC}/（PSC[15:0]+1）。PSC 包含了当更新事件产生时装入当前预分频器寄存器的值。

13.3.12 TIM2 自动重装寄存器（TIM2_ARR）

偏移地址：0x2C

复位值：0x0000

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

位 15:0	ARR[15:0]：自动重装低16 位值（Low auto-reload value） ARR 包含了将传送至实际的自动重装载寄存器的数值。当自动重装载的值为空时，计数器不工作。

13.3.13 TIM2 捕捉/比较寄存器 1（TIM2_CCR1）

偏移地址：0x34

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
rw															

位 15:0	CCR1[15:0]：捕捉/比较 1 值（Low Capture/Compare 1 value） • 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。如果在 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载特性，写入的数值会被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。

13.3.14 TIM2 捕捉/比较寄存器 2（TIM2_CCR2）

偏移地址：0x38

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
rw															

位 15:0	CCR2[15:0]：捕捉/比较 2 值（Low Capture/Compare 2 value） • 若CC2 通道配置为输出： CCR2 包含了装入 TIMx_CCR2 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器（OC2PE 位）中未选择预装载特性，写入的数值会被立即传输至 TIMx_CCR2 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 TIMx_CCR2 寄存器。 TIMx_CCR2 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。

13.3.15 TIM2 捕捉/比较寄存器 3（TIM2_CCR3）

偏移地址：0x3C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
rw															

位 15:0	CCR3[15:0]：捕捉/比较 3 值 （Low Capture/Compare 3 value） • 若 CC3 通道配置为输出： CCR3 包含了装入 TIMx_CCR3 寄存器的值（预装载值）。如果在 TIMx_CCMR3 寄存器（OC3PE 位）中未选择预装载特性，写入的数值会被立即传输至 TIMx_CCR3 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 TIMx_CCR3 寄存器。 TIMx_CCR3 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。

13.3.16 TIM2 捕捉/比较寄存器 4（TIM2_CCR4）

偏移地址：0x40

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															
rw															

位 15:0	CCR4[15:0]：捕捉/比较 4 值 （Low Capture/Compare 4 value） • 若 CC4 通道配置为输出： CCR4 包含了装入当前 TIMx_CCR4 寄存器的值（预装载值）。 如果在 TIMx_CCMR4 寄存器（OC4PE 位）中未选择预装载特性，写入的数值会被立即传输至当前寄存器。否则只有当更新事件发生时，此预装载值才传输至 TIMx_CCR4 寄存器。 TIMx_CCR4 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。

14 基本定时器（TIM6）

基本定时器 TIM6 包含一个 16 位自动重装载计数器，由它的编程预分频器驱动。

TIM6 可以作为通用定时器提供时间基准。

表 14-1 TIM6 特性

符号	参数	条件	最小值	典型值	最大值	单位
$t_{res(TIM)}$	定时器分辨时间	$f_{TIMxCLK}$=48MHz	-	20.8	-	ns
t_{MAX_COUNT}	当选择内部时钟时，16 位计数器的时钟周期	-	-	2^{16}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48MHz	-	1365	-	μs

14.1 TIM6 主要功能

- 16 位向上、向下自动重装载计数器
- 16 位可编程（可实时修改）预分频器，用于对输入的时钟按系数为 1~65536 之间的任意数值分频。
- 在更新事件（计数器溢出）时产生中断。

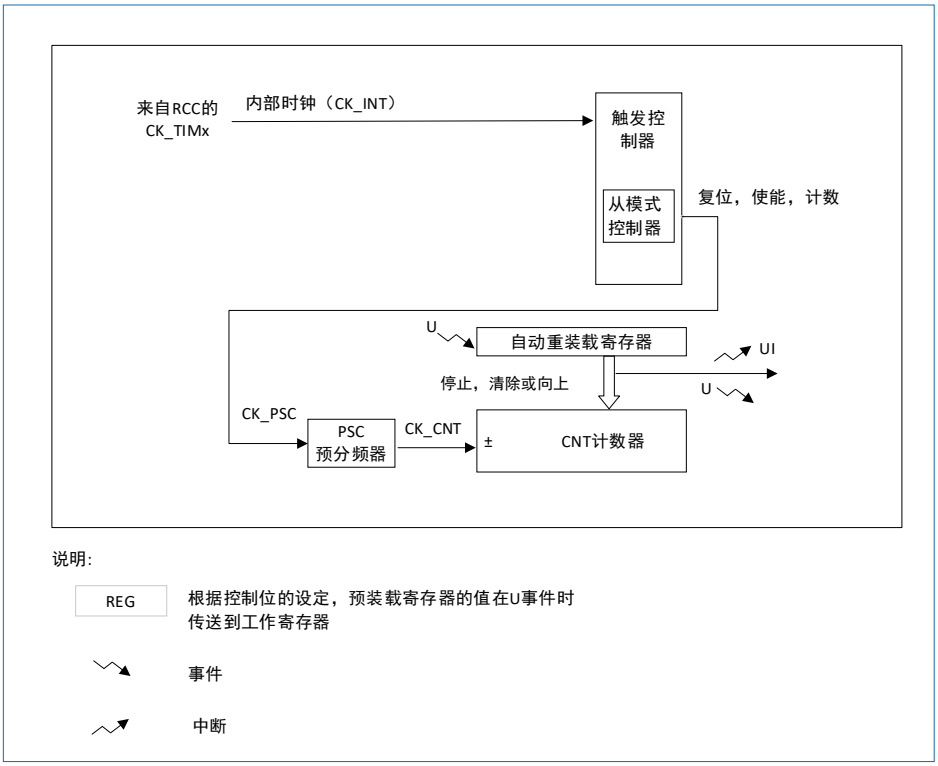

图 14-1 基本寄存器框图

14.2 TIM6 功能描述

14.2.1 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位累加计数器。计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包含：

- 计数器寄存器（TIM6_CNT）
- 预分频寄存器（TIM6_PSC）
- 自动重装载寄存器（TIM6_ARR）

自动重装载寄存器是预加载的。每次读写自动重装载寄存器时，实际上是通过读写预加载寄存器实现。根据 TIM6_CR1 寄存器中的自动重装载预加载使能位（ARPE），写入预加载寄存器的内容能够立即或在每次更新事件时，传送到它的影子寄存器。当 TIM6_CR1 寄存器的 UDIS 位为’0’，则每当计数器达到溢出值时，硬件产出更新事件；软件也可以产生更新事件；关于更新事件的产生，随后会有详细的介绍。

计数器由预分频输出 CK_CNT 驱动，设置 TIM6_CR1 寄存器中的计数器使能位（CEN）使能计数器计数。

说明：实际的设置计数器使能信号 CNT_EN 相对于 CEN 滞后一个时钟周期。

预分频器

预分频能以系数介于 1 至 65536 之间的任意数值对计数器时钟分频。它是通过一个 16 位寄存器（TIM6_PSC）的计数实现分频。因为 TIM6_PSC 控制寄存器具有缓冲，可以在运行过程中改变它的数值，新的预分频数值将在下一个更新事件时起作用。

以下两图是在运行过程中改变预分频系数的例子。

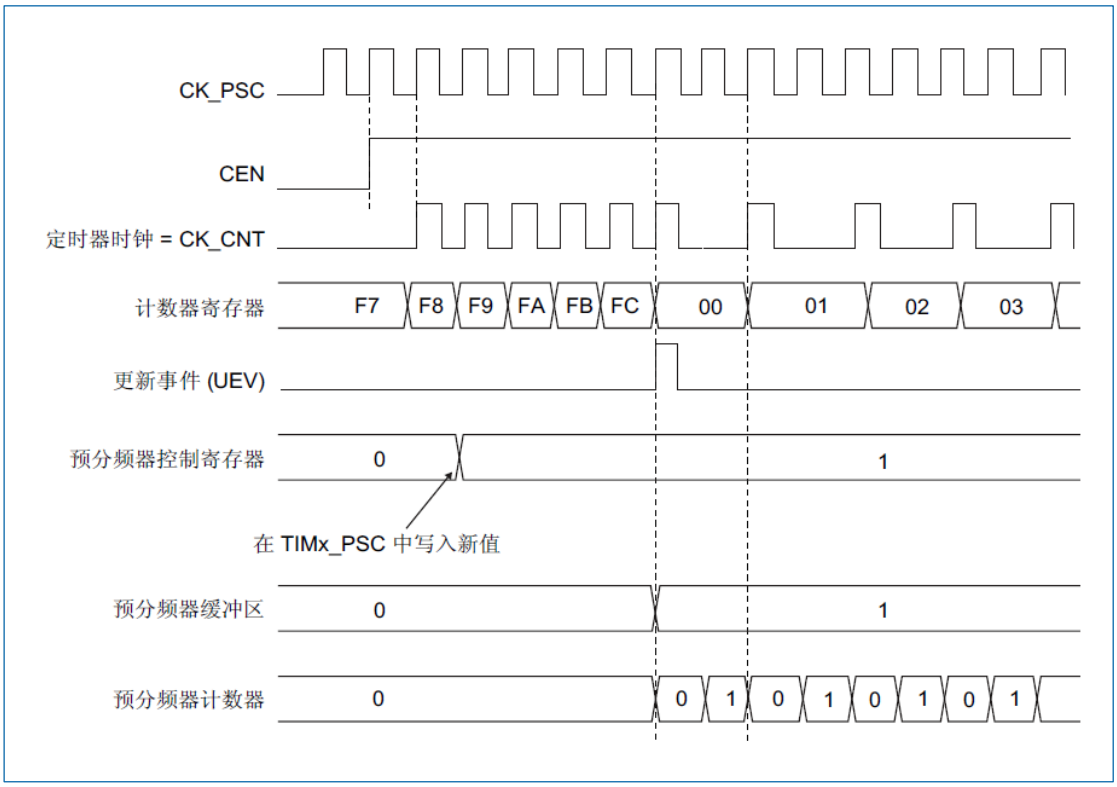

图 14-2 预分频系数从 1 变到 2 的计数器时序图

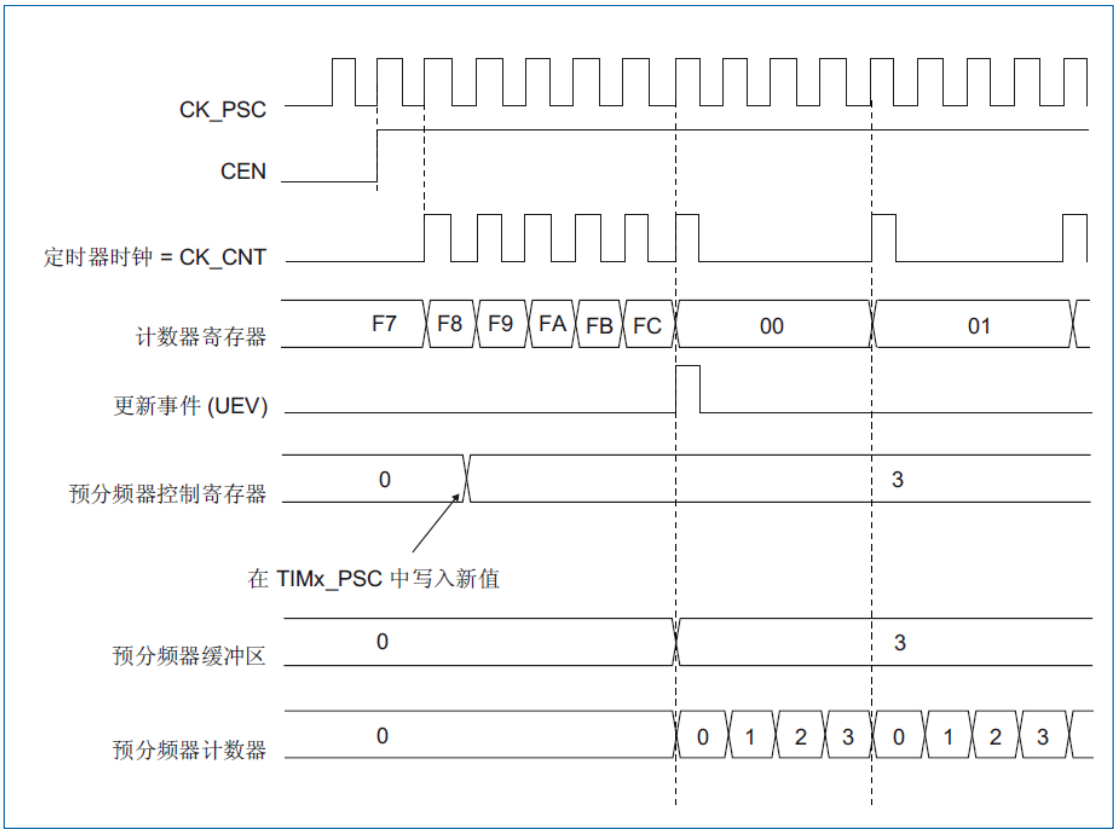

图 14-3 预分频系数从 1 变到 4 的计数器时序图

14.2.2 计数模式

14.2.2.1 向上计数模式

计数器从 0 累加计数到自动重装载数值（TIM6_ARR 寄存器），然后重新从 0 开始计数并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；（通过软件或使用从模式控制器）设置 TIM6_EGR 寄存器的 UG 位也可以产生更新事件。

设置 TIM6_CR1 中的 UDIS 位可以禁止产生 UEV 事件，这可以避免在写入预加载寄存器时更改影子寄存器。在清除 UDIS 位为’0’之前，将不再产生更新事件，但计数器和预分频器依然会在应产生更新事件时重新从 0 开始计数（但预分频系数不变）。另外，如果设置了 TIM6_CR1 寄存器中的 URS（选择更新请求），设置 UG 位可以产生一次更新事件 UEV，但不设置 UIF 标志（即没有中断）。

当发生一次更新事件时，所有寄存器会被更新并（根据 URS 位）设置更新标志（TIM6_SR 寄存器的 UIF 位）：

- 传送预装载值（TIM6_PSC 寄存器的内容）至预分频器的缓冲区。
- 自动重装载影子寄存器被更新为预装载值（TIM6_ARR）。

以下是一些在 TIM6_ARR=0x36 时不同时钟频率下计数器工作的图示例。

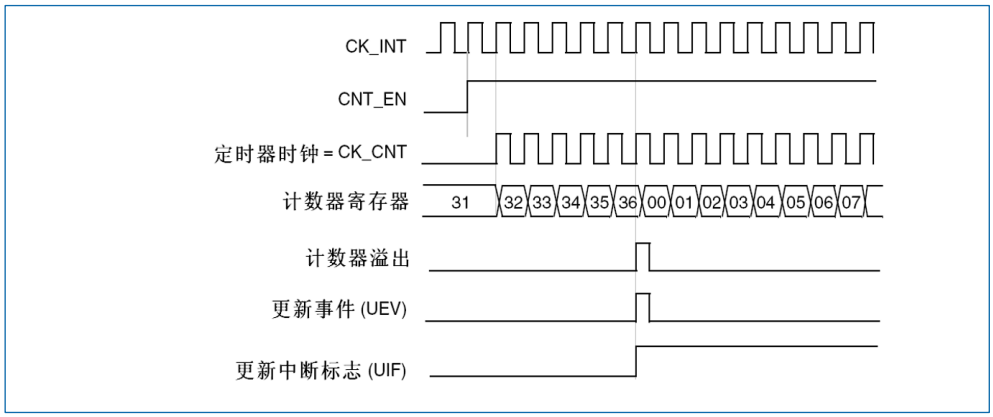

图 14-4 计数器时序图，内部时钟分频系数为 1

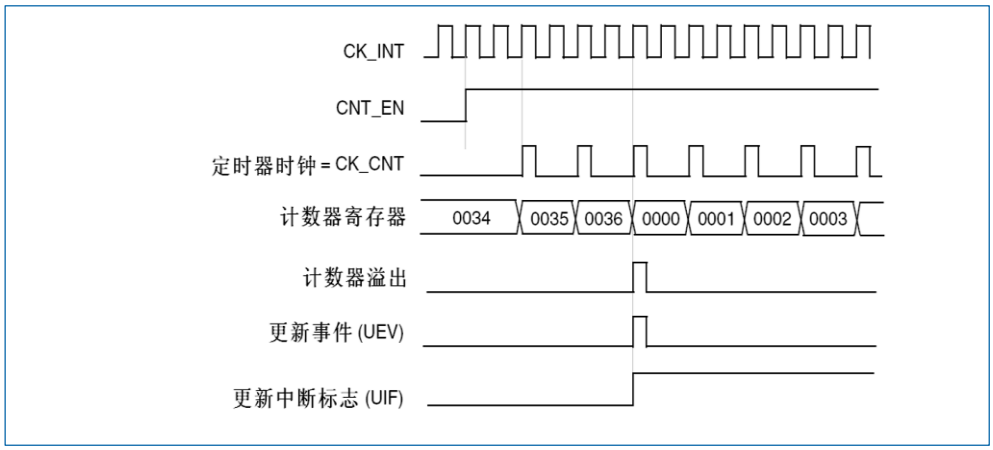

图 14-5 计数器时序图，内部时钟分频系数为 2

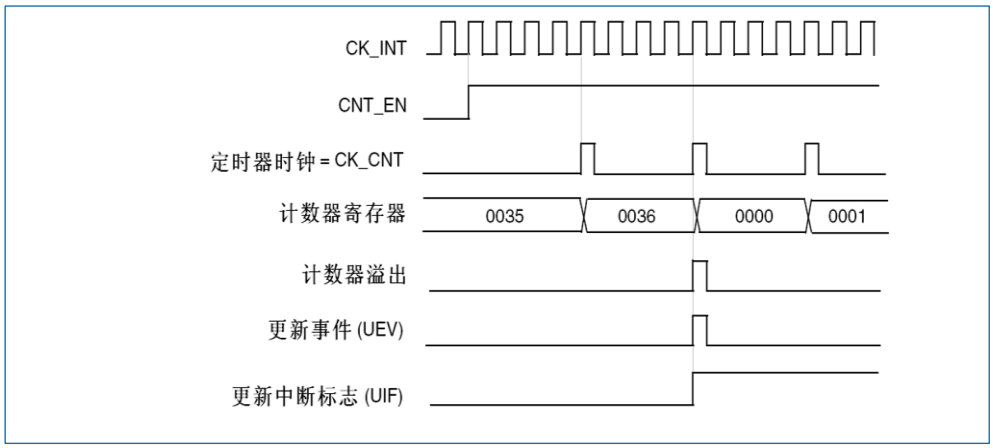

图 14-6 计数器时序图，内部时钟分频系数为 4

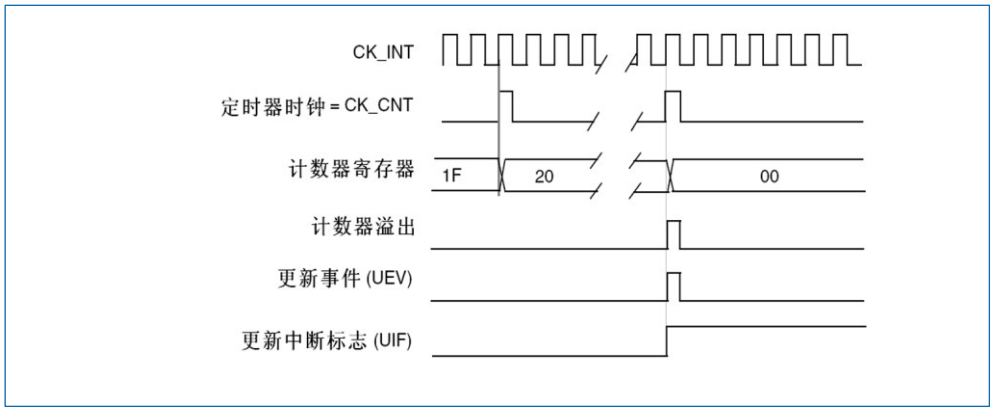

图 14-7 计数器时序图，内部时钟分频系数为 N

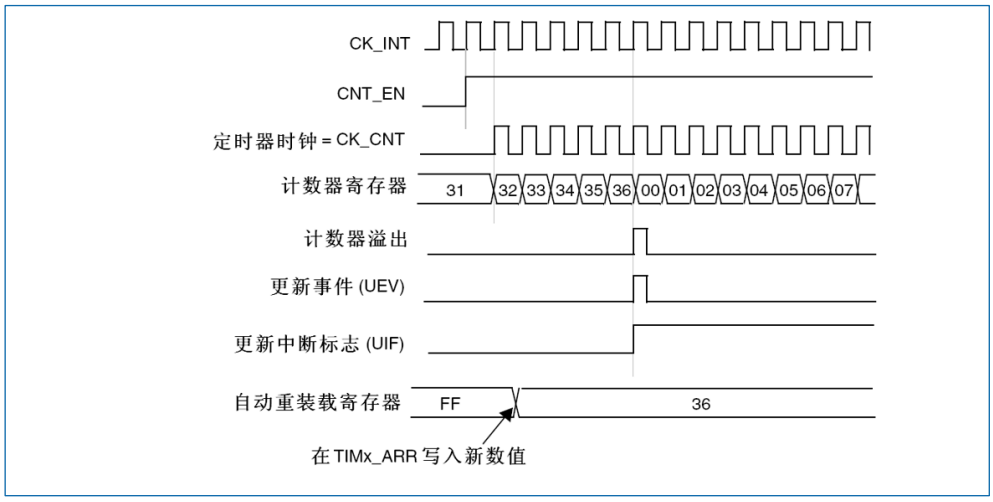

图 14-8 计数器时序图，当 ARPE=1 时的更新事件（无预装载 TIM6_ARR）

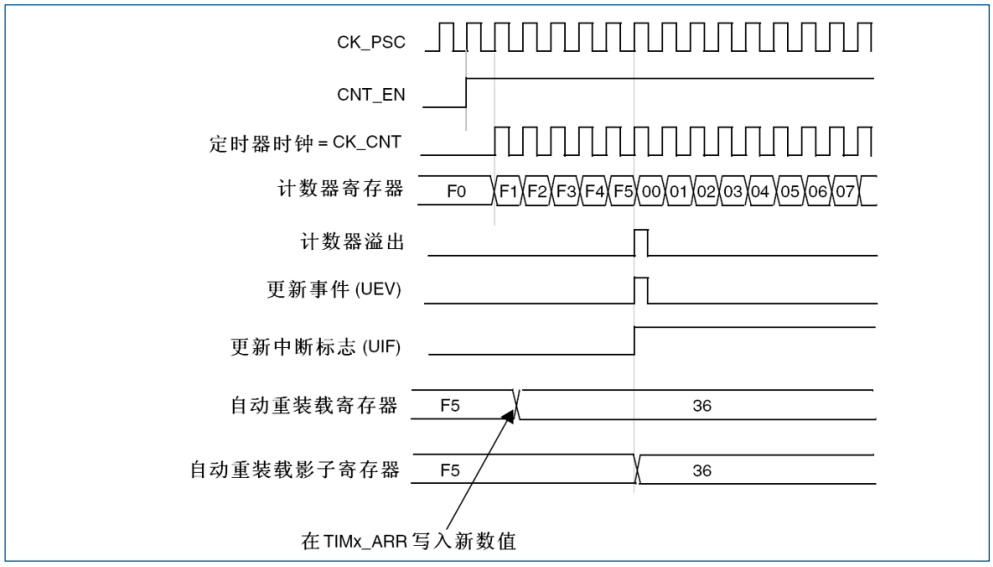

图 14-9 计数器时序图，当 ARPE=1 时的更新事件（预装载 TIM6_ARR）

14.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（TIM6_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 TIM6_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 TIM6_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，同时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 TIM6_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM6_SR 寄存器中的 UIF 位）也被设置。

- 预分频器的缓存器被置入预装载寄存器的值（TIM6_PSC 寄存器的值）。
- 当前的自动加载寄存器被更新为预装载值（TIM6_ARR 寄存器中的内容）。

说明：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 TIM6_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

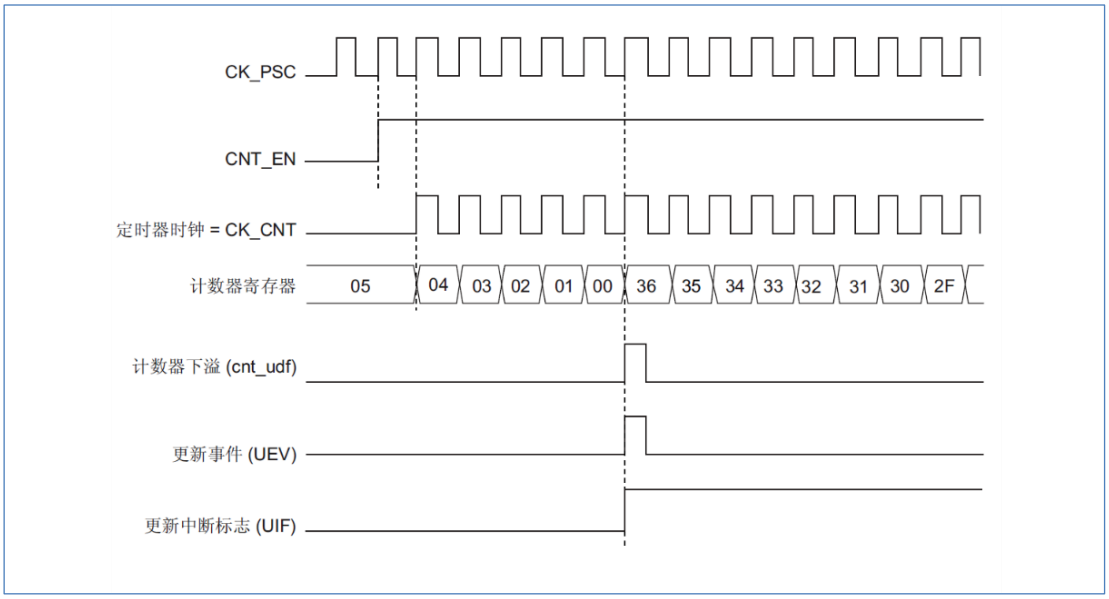

图 14-10 计数器时序图，内部时钟分频因子为 1

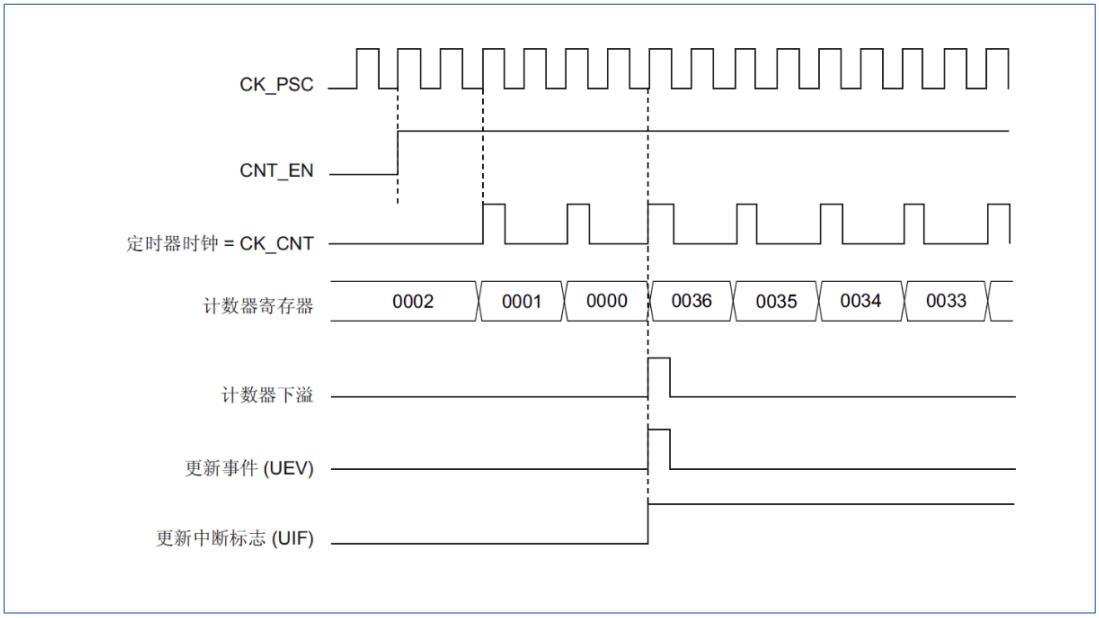

图 14-11 计数器时序图，内部时钟分频因子为 2

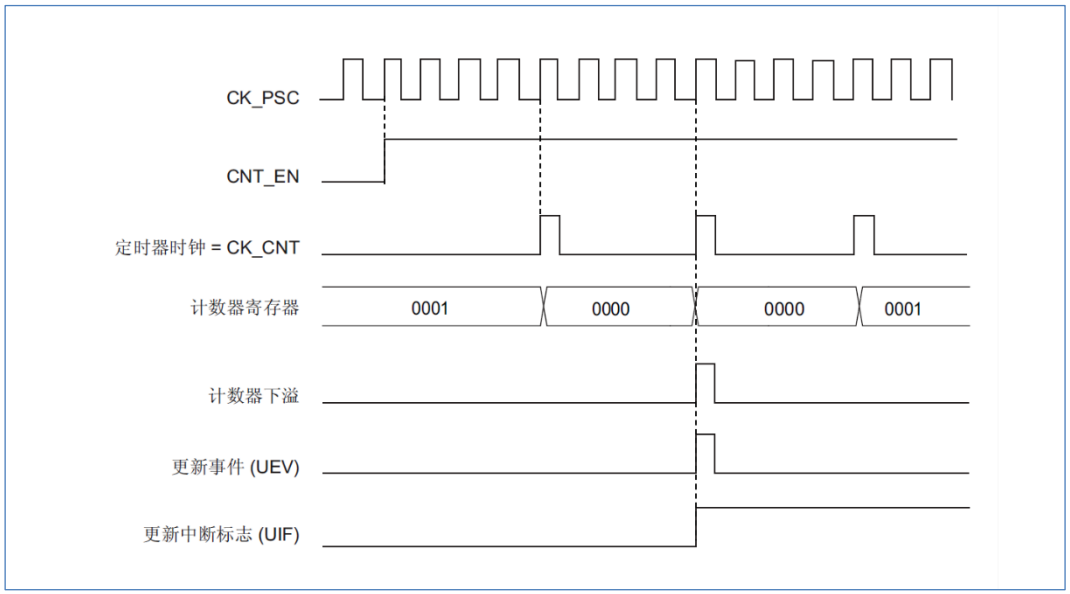

图 14-12 计数器时序图，内部时钟分频因子为 4

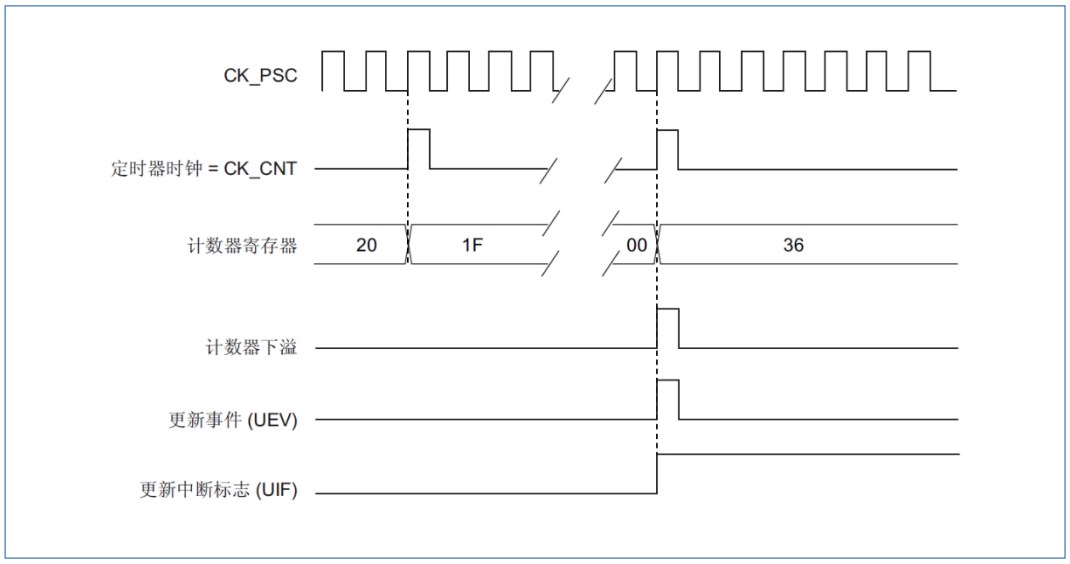

图 14-13 计数器时序图，内部时钟分频因子为 N

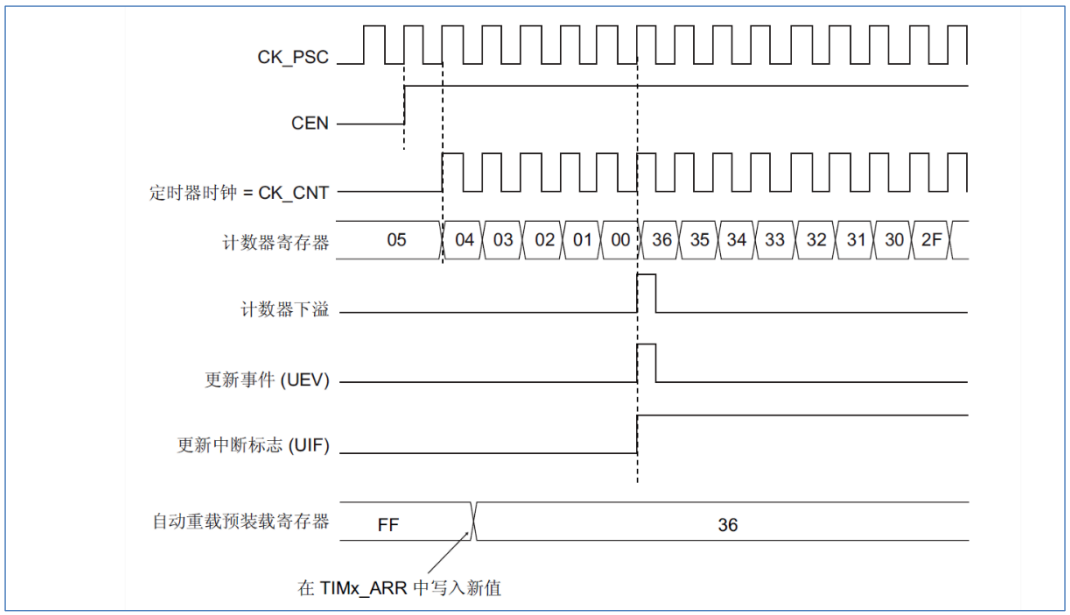

图 14-14 计数器时序图，当没有使用重复计数器时的更新事件

14.2.3 时钟源

计数器的时钟由内部时钟（CK_INT）提供。

TIM6_CR1 寄存器的 CEN 位和 TIM6_EGR 寄存器的 UG 位是实际的控制位，（除了 UG 位被自动清除外）只能通过软件改变它们。一旦置 CEN 位为’1’，内部时钟即向预分频器提供时钟。

下图示出控制电路和向上计数器在普通模式下，没有预分频器时的操作。

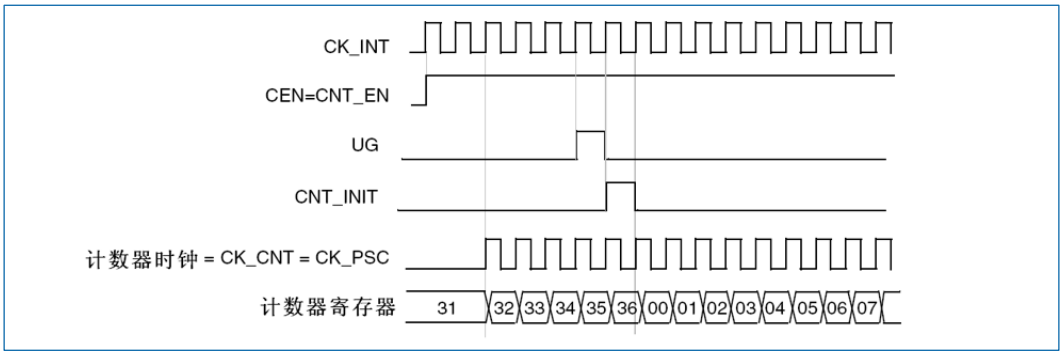

图 14-15 普通模式时序图，内部时钟分频系数为 1

14.2.4 调试模式

当 MCU 进入调试模式（Cortex-M0 内核停止）时，根据 DBG 模块中的配置位 DBG_TIM6_STOP 的设置，TIM6 计数器或者继续计数或者停止工作。

14.3 TIM6 寄存器

基地址：0x4000 1000

空间大小：0x400

14.3.1 TIM6 控制寄存器 1（TIM6_CR1）

偏移地址：0x00

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								ARPE	Res		DIR	OPM	URS	UDIS	CEN
								rw			rw	rw	rw	rw	rw

位 15:8	Res：保留 必须保持复位值。
位 7	ARPE：自动重装预装载使能（Auto-reload preload enable） • 0：TIMx_ARR 寄存器无缓冲器 • 1：TIMx_ARR 寄存器有缓冲器
位 6:5	Res：保留 必须保持复位值。
位 4	DIR：方向（Direction） • 0：计数器向上计数

	• 1：计数器向下计数
位 3	OPM：单脉冲模式（One pulse mode） • 0：在发生更新事件时，计数器不停止。 • 1：在发生下次更新事件时，计数器停止计数（清除 CEN 位）。

位 2	URS：更新请求源（Update request source）
	该位由软件置 1 和清 0，以选择 UEV 事件的请求源。
	• 0：如果使能了中断请求，以下任一事件可以产生一个更新中断请求：
	  ○ 计数器上溢或下溢
	  ○ 设置 UG 位
	  ○ 通过从模式控制器产生的更新
	• 1：如果使能了中断请求，只有计数器上溢或下溢才产生更新中断请求。

位 1	UDIS：禁止更新（Update disable）
	该位由软件置 1 和清 0，以使能或禁止 UEV 事件的产生。
	• 0：UEV 使能。更新事件（UEV）可以由下列事件产生：
	  ○ 计数器上溢或下溢
	  ○ 设置 UG 位
	  ○ 通过从模式控制器产生的更新生成更新事件后，带缓冲的寄存器被加载为预加载数值。
	• 1：禁止 UEV
	  不产生更新事件，影子寄存器保持它的内容（ARR、PSC）。但是如果设置了 UG 位或从模式控制器产生了一个硬件复位，则计数器和预分频器将被重新初始化。

位 0	CEN：计数器使能（Counter enable） • 0：计数器禁止 • 1：计数器使能 在单脉冲模式下，当产生更新事件时 CEN 被自动清除。

14.3.2 TIM6 控制寄存器 2（TIM6_CR2）

偏移地址：0x04

复位值：0x0000

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

位 15:7	Res：保留 必须保持复位值。
位 6:4	MMS[2:0]：主模式选择（Master mode selection） 这些位用于选择在主模式下向从定时器发送的同步信息（TRGO）。 • 000：复位

	使用 TIMx_EGR 寄存器的 UG 位作为触发输出（TRGO）。如果触发输入产生了复位（从模式控制器配置为复位模式），则相对于实际的复位信号，TRGO 上的信号有一定的延迟。 • 001：使能 计数器使能信号 CNT_EN 被用作为触发输出（TRGO）。它可用于在同一时刻启动多个定时器，或控制使能从定时器的时机。计数器使能信号是通过 CEN 控制位和配置为门控模式时的触发输入的'逻辑或'产生。当计数器使能信号是通过触发输入控制时，在 TRGO 输出上会有一些延迟，除非选择了主/从模式。 • 010：更新 更新事件被用作为触发输出（TRGO）。例如：一个主定时器可以作为从定时器的预分频器使用。
位3:0	Res：保留 必须保持复位值。

14.3.3 TIM6 中断使能寄存器（TIM6_DIER）

偏移地址：0x0C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															UIE
															rw

位 15:1	Res：保留 必须保持复位值。
位0	UIE：更新中断使能（Update interrupt enable） • 0：更新中断禁用 • 1：更新中断使能

14.3.4 TIM6 状态寄存器（TIM6_SR）

偏移地址：0x10

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															UIF
															rc_w0

位 15:1	Res：保留 必须保持复位值。
位0	UIF：更新中断标志（Update interrupt flag） 硬件在更新中断时设置该位，它由软件清除。 • 0：没有产生更新。 • 1：产生了更新中断。满足以下情况，该位由硬件置位：

	◦ 计数器产生上溢或下溢并且 TIMx_CR1 中的 UDIS=0。 ◦ 如果 TIMx_CR1 中的 URS=0 并且 UDIS=0，当使用 TIMx_EGR 寄存器的 UG 位重新初始化计数器 CNT 时。

14.3.5 TIM6 事件产生寄存器（TIM6_EGR）

偏移地址：0x14

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															UG
															w

位 15:1	Res：保留 必须保持复位值。
位 0	UG：产生更新事件（Update generation） 该位由软件置 1，由硬件自动清 0。 • 0：无作用 • 1：重新初始化定时器的计数器，并产生一个更新事件。 *注意：预分频器的计数器清 0，但预分频系数不变。若在中央对齐模式下或 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。*

14.3.6 TIM6 计数寄存器（TIM6_CNT）

偏移地址：0x24

复位值：0x0000

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

位 15:0	CNT[15:0]：计数器值（counter value）

14.3.7 TIM6 预分频寄存器（TIM6_PSC）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频器值（Prescaler value） 计数器的时钟频率 CK_CNT 等于 f_{CK_PSC}/（PSC[15:0]+1）。在每一次更新事件时，PSC 的数值被传送到实际的预分频寄存器。

14.3.8 TIM6 自动重装寄存器（TIM6_ARR）

偏移地址：0x2C

复位值：0xFFFF

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

位 15:0	ARR[15:0]：自动重装载数值（Auto-reload value） ARR 的数值将传送到实际的自动重装载寄存器。如果自动重装载数值为空，则计数器停止。

15 自动唤醒定时器（AWUT）

AWUT 用于在 MCU 停机模式（Stop）下计时并产生中断唤醒 MCU；AWUT 内置超低功耗 22 位定时器，工作时钟可配置为 GPIO 外部输入时钟或片内 LSI 慢速时钟；定时器使用向下计数的方式计数。

该模块输入输出示意图如下：

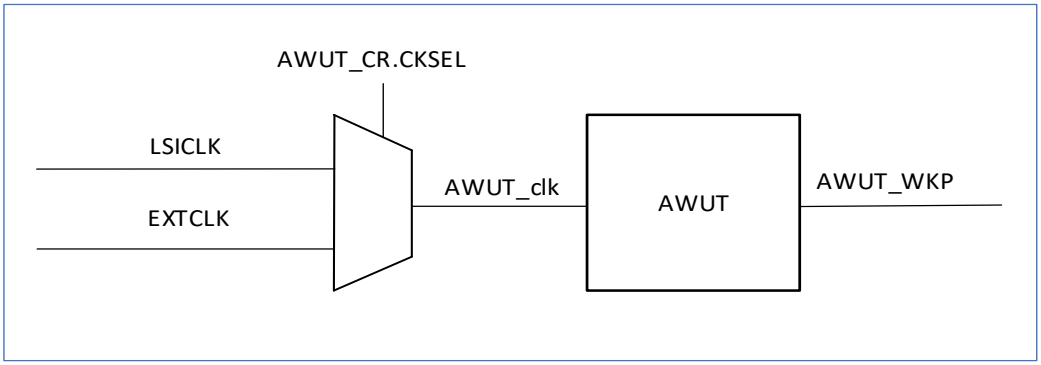

图 15-1 AWUT 输入输出示意图

15.1 AWUT 寄存器

基地址：0x4000 7800

空间大小：0x400

15.1.1 AWUT 控制寄存器（AWUT_CR）

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RLR_WBUSY	Res								AWUT_RLR[21:15]						
r									rw						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AWUT_RLR[14:0]															CKSEL
rw															rw

位 31	RLR_WBUSY：AWUT 重载寄存器状态位（Reload register write busy） • 0：AWUT_RLR 位域未被 APB 总线写操作 • 1：AWUT_RLR 位域正在被 APB 总线写操作 （当 RLR_WBUSY=1 时，软件禁止再次对 RLR[21:0]位域进行写操作）。
位 30:23	Res：保留 必须保持复位值。
位 22:1	AWUT_RLR[21:0]：AWUT 重载寄存器（AWUT reload register） AWUT 计数器装载值；此重载值将在 MCU 进入 Stop 模式时刻被自动加载到 AWUT 内部的 22 位计时器中，并开始计时。 *说明：* *当 AWUT_RLR[21:0]=22'd0 或 22'd1 时，重载行为将不会发生且 AWUT 将不会启动工*

	作。 *当 AWUT_WBUSY =1 时，对 AWUT_RLR[21:0]位域的写操作将无效。*
位 0	CKSEL：自动唤醒时钟选择位（Automatic wakeup timer clock selection） • 0：选择 LSI 作为 AWUT 定时器的计时时钟 • 1：选择 GPIO 外部输入时钟作为 AWUT 定时器的计时时钟

15.1.2 AWUT 状态寄存器（AWUT_SR）

偏移地址：0x04

复位值：0x0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															BUSY
															r

位 31:1	Res：保留 必须保持复位值。
位 0	BUSY：自动唤醒计时器状态位（Automatic wakeup timer busy） • 0：AWUT 计时器未工作 • 1：AWUT 计时器正在工作 除了 AWUT，在停机（Stop）模式下的 MCU 还可以被其他事件唤醒。当 MCU 被其他事件唤醒进入第一次 Stop 模式时，在还未退出 Stop 模式的一段延迟时间内，AWUT 还在继续计时；若在这段延迟时间内，MCU 再次进入第二次 Stop 模式，则它有可能被这段延迟时间内产生的 AWUT CNT_INT 事件意外地唤醒。 为了避免第二次 Stop 模式被 AWUT 意外唤醒的问题，推荐软件在执行第二次 WFI/WFE 指令前通过读取 BUSY 状态标志位判断 AWUT 是否正在工作（AWUT_BUSY=1）。若 AMUT 正在工作，需要等待 AWUT_BUSY=0 后，检测和清除 AWUT_WKP 中断，然后再执行第二次 WFI/WFE 指令。 AWUT_WKP 中断标志的检测和清除通过软件访问 EXTI 模块相应寄存器完成。

16 独立看门狗（IWDG）

器件内置两个看门狗（独立看门狗 IWDG 和窗口看门狗 WWDG），提供了更高的安全性、精确性和灵活性。两个看门狗设备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（如表 16-1）时，将产生系统复位或触发一个中断（中断仅适用于 WWDG）。

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。而 WWDG 由从 APB1 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 最适合那些需要看门狗独立工作于在主程序之外，且对时间精度要求较低的应用。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。

关于窗口看门狗的详情，请参见章节：“17 窗口看门狗（WWDG）”。

16.1 IWDG 主要性能

- 自由运行的递减计数器
- 时钟由独立的 RC 振荡器提供（可在停机模式下工作）
- 复位条件
  - 当递减计数器值等于 0x000 时复位（如果看门狗已激活）
  - 在窗口之外重载递减计数器时复位（如果看门狗已激活）
- IWDG 计数器复位初始值可由 Flash 选项字设置（请参见：“3.3 Flash 选项字节”）通过配置 Flash 选项字，可以保证当芯片复位后如果程序出现运行故障，IWDG 复位的时间间隔不会太长。

  可以通过 Flash 选项字 LSI_LP_CTL 控制芯片进入停机（Stop）模式后 LSI 的状态。通过配置 Flash 选项字，可以选择使芯片进入停机（Stop）模式后，如果 LSION 设置为 0，则关掉 LSI；在芯片唤醒后，LSI 恢复成进该模式之前的状态。如果不配置 Flash 选项字，则在使能 IWDG 后再进入停机（Stop）模式后，系统会被 IWDG 周期唤醒。

16.2 IWDG 功能描述

图 16-1 为独立看门狗模块的功能框图。

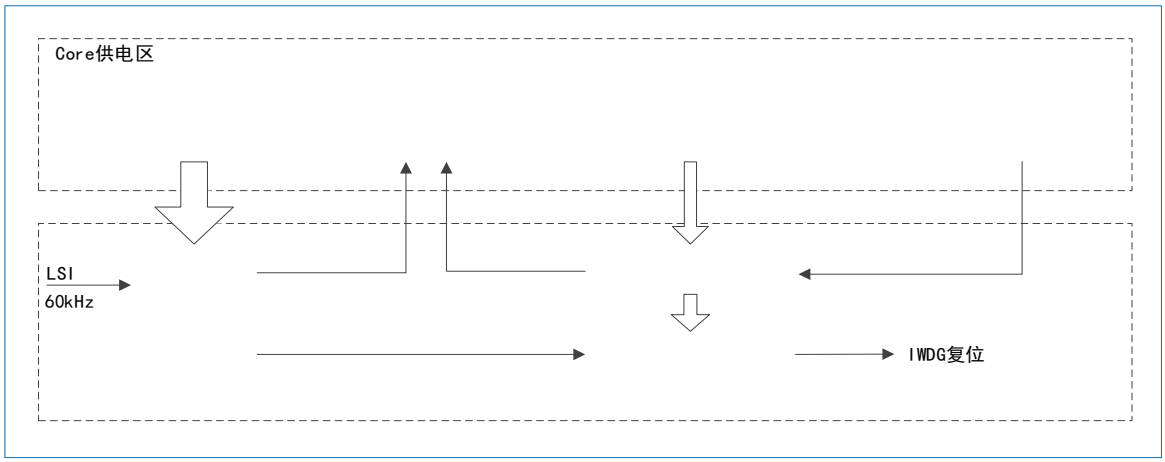

图 16-1 IWDG 框图

在键寄存器（IWDG_KR）中写入 0xCCCC，开始启用 IWDG；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。

支持看门狗窗口（window）模式，详细的描述请参见章节：“16.3.5 窗口寄存器（IWDG_WINR）”。

注意：看门狗功能处于 V_{DD} 供电区，即在停机模式时仍能正常工作。

计算超时的公式为：

T_{IWDG}=（（1/（f_{LSI}/PR[2:0]））*（RL[11:0]+1）

其中：

- T_{IWDG} 为 IWDG 超时时间；
- f_{LSI}：LSI 的频率。

表 16-1 IWDG 超时时间表（60kHz 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	0.067	273.07
/8	1	0.13	546.13
/16	2	0.27	1092.27
/32	3	0.53	2184.53
/64	4	1.07	4369.07
/128	5	2.13	8738.13
/256	6 或 7	4.27	17476.27

注意:

这些时间是按照 60 kHz 时钟给出。实际上，MCU 内部的 RC 频率会有 10%的误差。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。

通过对 LSI 进行校准可获得相对精确的看门狗超时时间。

16.2.1 窗口选项

通过在 IWDG_WINR 寄存器中设置合适的窗口，IWDG 也可以用作窗口看门狗。IWDG_WINR 的默认值为 0x0000 0FFF，因此如果不更新此默认值，窗口选项将一直处于禁用状态。窗口值一经更改，便会执行重载操作，以便将递减计数器的值复位为 IWDG_RLR 值，方便计算周期数以生成下一次重载。当计数器值大于窗口寄存器（IWDG_WINR）中存储的值时，如果执行重载操作，则会产生复位。

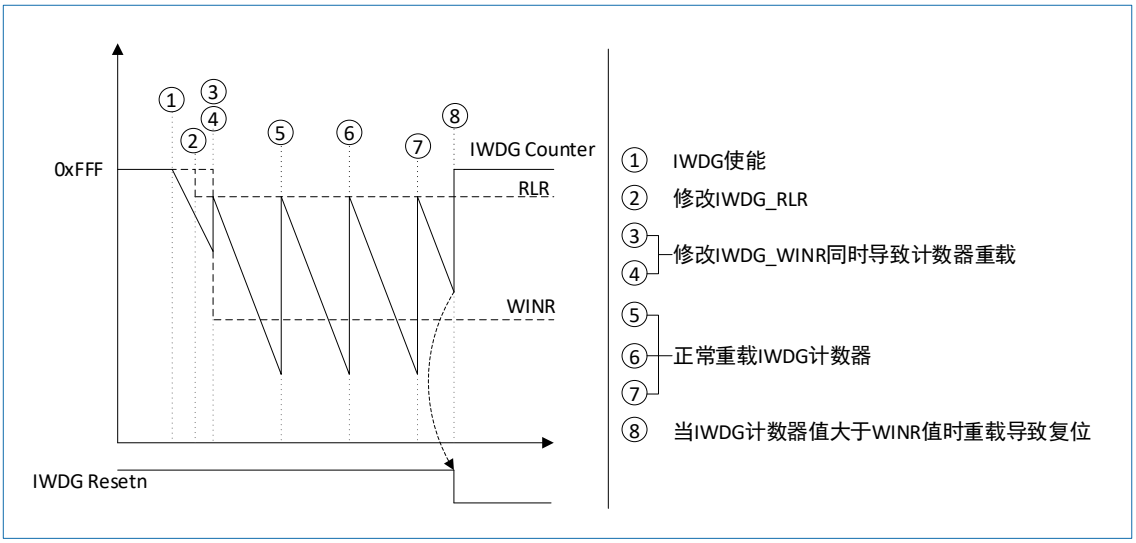

图 16-2 IWDG 使能窗口选项时的工作说明

使能窗口选项时配置 IWDG 的流程：

1. 向 IWDG_KR 写入 0x0000 CCCC 来使能 IWDG。
2. 向 IWDG_KR 写入 0x0000 5555 来使能寄存器访问。
3. 修改 IWDG_PR 的值为 0~7 中的值，配置 IWDG 的预分频器。
4. 写重装载寄存器 IWDG_RLR。
5. 等待 IWDG_RLR 寄存器值更新完成（IWDG_SR = 0x0000 0000）。
6. 写窗口寄存器 IWDG_WINR，这个操作会导致 IWDG 计数器自动更新为 IWDG_RLR 中的值。

注意：当 IWDG_SR 为 0x0000 0000 时，才能写 IWDG_WINR，否则 IWDG 计数器可能不会正确的重载为 IWDG_RLR 中的值。

禁能窗口选项时配置 IWDG 的流程：

1. 向 IWDG_KR 写入 0x0000 CCCC 来使能 IWDG。
2. 向 IWDG_KR 写入 0x0000 5555 来使能寄存器访问。
3. 修改 IWDG_PR 的值为 0~7 中的值，配置 IWDG 的预分频器。
4. 写重载寄存器 IWDG_RLR。
5. 等待 IWDG_RLR 寄存器值更新完成（IWDG_SR = 0x0000 0000）。
6. 刷新计数器值为 IWDG_RLR 值（IWDG_KR = 0x0000 AAAA）。

16.2.2 硬件看门狗

如果用户在选择字节中启用了硬件看门狗功能，在系统上电复位后，看门狗会自动开始运行；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。

16.2.3 寄存器访问保护

IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器具有写保护功能。要修改这三个寄存器的值，必须先向 IWDG_KR 寄存器写入 0x5555。以其他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（IWDG_SR）指示预分频值和递减计数器是否正在被更新。

16.2.4 调试模式

当微控制器进入调试模式时（Cortex-M0 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。详细信息，请参见：“22.8.2 对定时器、看门狗和 I2C 的调试支持”。

16.3 IWDG 寄存器

基地址：0x4000 3000

空间大小：0x400

16.3.1 关键字寄存器（IWDG_KR）

偏移地址：0x00

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	KEY[15:0]：键值（Key value） • 软件必须以一定的时间间隔写入 0xAAAA；否则，当计数器为 0 时，看门狗会产生复位。 • 写入 0x5555 表示允许访问 IWDG_PR，IWDG_RLR 和 IWDG_WINR 寄存器，参见“16.2.3 寄存器访问保护”。 • 写入 0xCCCC，启动看门狗工作（若选择了硬件看门狗，则不受此命令字限制）。

16.3.2 预分频寄存器（IWDG_PR）

偏移地址：0x04

复位值：0x0000 0000

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

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

位 31:3	Res：保留 必须保持复位值。
位 2:0	PR[2:0]：预分频因子（Prescaler divider） 该位域具有写保护设置（参见“16.2.3 寄存器访问保护”），通过设置该位域来选择计数器时钟的预分频因子。若需改变预分频因子，则 IWDG_SR 寄存器的 PVU 位必须为

	0。 具体的值与预分频因子对应关系如下： • 000：4 分频 • 001：8 分频 • 010：16 分频 • 011：32 分频 • 100：64 分频 • 101：128 分频 • 110：256 分频 • 111：256 分频 *说明：读取该寄存器会返回 V_{DD} 电压域的预分频器值。如果正在对该寄存器执行写操作，则读取的值可能不算最新的有效值。因此，只有在 IWDG_SR 寄存器中的 PVU 位为 0 时，从寄存器读取的值才有效。*

16.3.3 重装载寄存器（IWDG_RLR）

偏移地址：0x08

复位值：0x0000 0FFF

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

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

位 31:12	Res：保留 必须保持复位值。
位 11:0	RL[11:0]：看门狗计数器重载值（Watchdog counter reload value） 该位域具有写保护设置（参见“16.2.3 寄存器访问保护）。 RL 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。 看门狗的超时周期可以通过此重装载值和时钟预分频值来计算，可以参考表 16-1。 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。 *说明：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 寄存器中的 RVU 位为 0 时。读出的值才有效。*

16.3.4 状态寄存器（IWDG_SR）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res													WVU	RVU	PVU
													r	r	r

位 31:3	Res：保留 必须保持复位值。
位 2	WVU：看门狗计数器窗口值更新（Watchdog counter window value update） 该位由硬件置 1，用于表明正在更新窗口值。 当在 V_{DD} 电压域下完成重载值更新操作后,该位会被硬件清零（需要多达 5 个 60kHz 的 RC 震荡周期）。 窗口值只有在 WVU 位为 0 时才可更新。 此位只有在窗口功能打开时才有效。
位 1	RVU：看门狗计数器重装载值更新（Watchdog counter reload value update） 该位由硬件置 1，用于指示重载值正在更新。 当在 V_{DD} 电压域下完成重载值更新操作后，会通过硬件将该位清零（需要多达 5 个 60 kHz 的 RC 震荡周期）。 重装载值只有在 RVU 位为 0 时才可更新。
位 0	PVU：看门狗预分频值更新（Watchdog prescaler value update） 该位由硬件置 1，用于指示预分频值正在更新。 当在 V_{DD} 电压域下完成预分频器值更新操作后,会通过硬件将该位清零（需要多达 5 个 60kHz 的 RC 震荡周期）。 预分频器值只有在 PVU 位为 0 时才可更新。

说明：如果在应用程序中使用了多个重载值或预分频值，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。而且必须等到 WVU 位被复位后才能更改窗口值。但是，在更新预分频器和/或重载/窗口值之后，则无需等到 RVU、PVU 或 WVU 复位后再继续执行代码（进入低功耗模式时除外）。

16.3.5 窗口寄存器（IWDG_WINR）

偏移地址：0x10

复位值：0x00000FFF

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

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

位 31:12	Res：保留 必须保持复位值。
位 11:0	WIN[11:0]：看门狗计数器窗口值（Watchdog counter window value） 这些位受写访问保护（参考 25.3.5 寄存器访问保护）这些位包含用于与递减计数器进

	行比较的窗口值上限。 为防止发生复位，当递减计数器的值低于窗口寄存器值且大于 0x0 时必须重载 若要更改重载值，IWDG_SR 中的 WVU 位必须为 0。 *说明：读取该寄存器会返回 V_{DD} 电压域的重载值。如果正在对该寄存器执行写操作，则读取的值可能无效。因此，只有在 IWDG_SR 寄存器中的 WVU 位为 0 时，从寄存器读取的值才有效。*

17 窗口看门狗（WWDG）

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所造成的应用程序背离正常的运行顺序而产生的软件故障。除非递减计数器的值 T6 位在变成 0 前被刷新，否则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 位的递减计数器的数值（在控制寄存器中）被刷新，那么也将产生一个 MCU 复位。这表明递减计数器必须在一个有限的时间窗口中被刷新。

WWDG 时钟由 APB 时钟分频得到，它具有可配置的时间窗口。该窗口可编程，以检测异常过晚或过早的应用行为。

WWDG 适用于那些需要看门狗工作在准确的时间窗口下的应用。

17.1 WWDG 主要特性

- 可编程的自由运行递减计数器。
- 复位条件：
  - 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。
- 提前唤醒中断（EWI）：如果启动了看门狗并且允许中断，当递减计数器等于 0x40 时产生提前唤醒中断，它可以被用于重装载计数器以避免 WWDG 复位。

17.2 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’），且当 7 位（T[6:0]）递减计数器从 0x40 翻转到 0x3F（T6 位清零）时，产生一个复位。如果软件在计数器值大于配置寄存器中的 W[6:0]值时，重新装载计数器，也将产生一个复位。

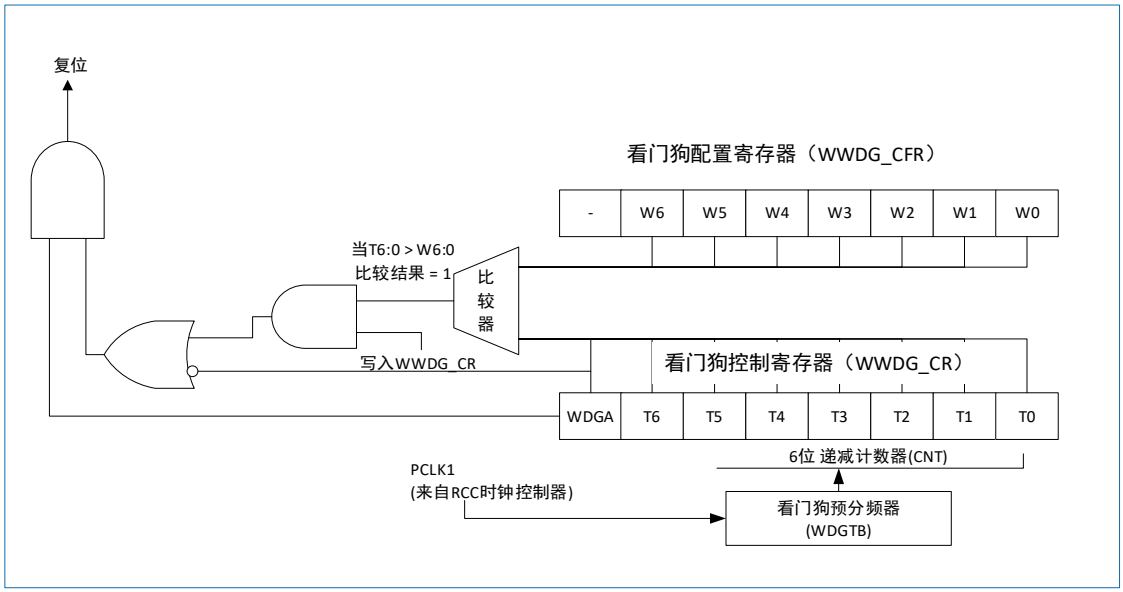

图 17-1 WWDG 框图

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配置寄存器的 W[6:0]值时，才能进行该操作。

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 17-2 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提前唤醒中断（EWI）。设置 WWDG_CFR 寄存器中的 EWI 位开启该中断。当递减计数器到达 0x40 时，则产生此中断，相应的中断服务程序（ISR）可以用来加载计数器以防止 WWDG 复位。在 WWDG_SR 寄存器中写'0'可以清除该中断。

注意：可以用 T6 位产生一个软件复位（设置 WDGA 位为'1'，T6 位为'0'）。

17.3 如何编写看门狗超时程序

可以使用图 17-2 提供的公式计算窗口看门狗的超时时间。

警告：当写入 WWDG_CR 寄存器时，确保 T6 位始终为'1'以避免立即产生一个复位。

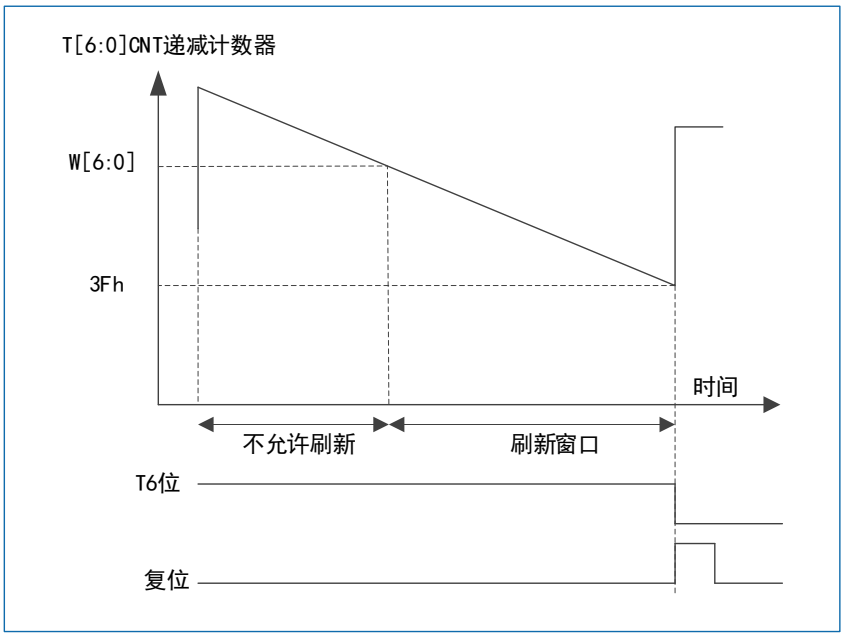

图 17-2 WWDG 时序图

计算超时的公式如下：

$T_{WWDG}=T_{PCLK1} * 4096 * 2^{WDGTB} *$（T[5:0] +1）

- T_{WWDG}：WWDG 超时时间
- T_{PCLK}：以 ms 为单位的 APB 时钟周期
- 4096：对应于内部分频器的值
- WDGTB：WWDG 分频系数

例如：

假设 APB 频率等于 48 MHz，将 WDGTB[1:0]设置为 3，将 T[6:0]设置为 63：

$T_{WWDG} = 1/ 48000 \times 4096 \times 2^3 \times (63 +1) = 43.69$ ms

表 17-1 WDGTB 超时值(PCLK=48M)

WDGTB	最小超时值 T[5:0]=0x00 (μs)	最大超时值 T[5:0]=0x3F (ms)
0	85	5.46
1	170	10.92
2	341	21.85
3	682	43.69

17.4 调试模式

当微控制器进入调试模式时（Cortex® -M0 核心停止），根据调试模块中的 DBG_WWDG_STOP 配置位的状态，WWDG 的计数器能够继续工作或停止。详见章节：“22.8.2 对定时器、看门狗和 I2C 的调试支持”。

17.5 WWDG 寄存器

基地址：0x4000 2C00

空间大小：0x400

17.5.1 控制寄存器（WWDG_CR）

偏移地址：0x00

复位值：0x0000007F

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								WDGA	T[6:0]						
								rs	rw						

位 31:8	Res：保留 必须保持复位值。
位 7	WDGA：激活位（Activation bit） 该位由软件置’1’，但仅在硬件复位后清’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
位 6:0	T[6:0]：7 位计数器（MSB 至 LSB）（7-bit counter, MSB to LSB） 该位域用来存储看门狗的计数器值。该值每（$4096x2^{WDGTB}$）个 PCLK 周期减 1。当计数器值从 0x40 翻转为 0x3F 时（T6 变成 0），产生看门狗复位。

17.5.2 配置寄存器（WWDG_CFR）

偏移地址：0x04

复位值：0x0000 007F

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						EWI	WDGTB[1:0]		W[6:0]						
						rs	rw		rw						

位 31:10	Res：保留 必须保持复位值。
位 9	EWI：提前唤醒中断（Early wakeup interrupt） 该位若置’1’，则当计数器值达到 0x40 就会产生中断。此中断只能由硬件在复位后清除。
位 8:7	WDGTB[1:0]：定时器时基（Timer base） 可按如下方式修改预分频器的时基： • 00：CK 计数器时钟（PCLK/4096）除以 1。 • 01：CK 计数器时钟（PCLK/4096）除以 2。 • 10：CK 计数器时钟（PCLK/4096）除以 4。 • 11：CK 计数器时钟（PCLK/4096）除以 8。
位 6:0	W[6:0]：7 位窗口值（7-bit window value） 该位域包含了用来与递减计数器进行比较的窗口值。

17.5.3 状态寄存器（WWDG_SR）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															EWIF
															rc_w0

位 31:1	Res：保留 必须保持复位值。
位 0	EWIF：提前唤醒中断标志（Early wakeup interrupt flag） 当计数器达到 0x40 时，该位由硬件置’1’，它必须通过软件写’0’来清除。对该位写’1’无效。若中断未被使能，该位也会被置’1’。

18 通用异步收发器（UART）

通用异步收发器（UART）支持标准的不归零码（Not return to Zero，NRZ）异步串行数据格式，并且能够进行全双工数据通信。UART 可以通过自带的波特率发生器产生不同的波特率，并且还支持单线半双工通信、多处理器通信等功能。

18.1 UART 主要特性

- 全双工异步通信
- NRZ 标准格式（标记/空格）
- 可配置为 16 倍过采样或 8 倍过采样（最高收发波特率= UART 时钟频率/8）
- 双时钟域允许：
  - 方便的波特率编程，独立于 PCLK 重新编程
- 数据字长可编程（7 位、8 位或 9 位）
- 可编程的数据顺序（MSB 或 LSB）
- 停止位可配置（支持 1 个或 2 个停止位）
- 单线半双工通信
- 发送器和接收器可单独使能
- 通信控制/错误检测标志
- 奇偶校验控制：
  - 发送奇偶校验位
  - 检查接收数据帧的奇偶性
- 具有多种中断状态标志位
- 多处理器通信
  - 如果地址不匹配，UART 将进入静默模式。
- 从静默模式唤醒（通过空闲线检测或地址标记检测）

18.2 UART 实现

表 18-1 UART 特性

UART 模式/特性	UART1/2
数据字长	7/8/9 位
多处理器通信	支持
单线半双工通信	支持
双时钟域	支持

18.3 UART 功能说明

UART 通信包括两个引脚，分别是接收数据输入引脚（RX）、发送数据输出引脚（TX）。它们的具体描述如下：

- RX：接收数据输入引脚。该引脚用于接收串行数据。

- TX：发送数据输出引脚。若禁用发送器，则该输出引脚的模式由其 I/O 端口配置决定。若使能发送器，则该引脚在空闲状态下处于高电平。在单线模式下，该引脚用于发送和接收数据。

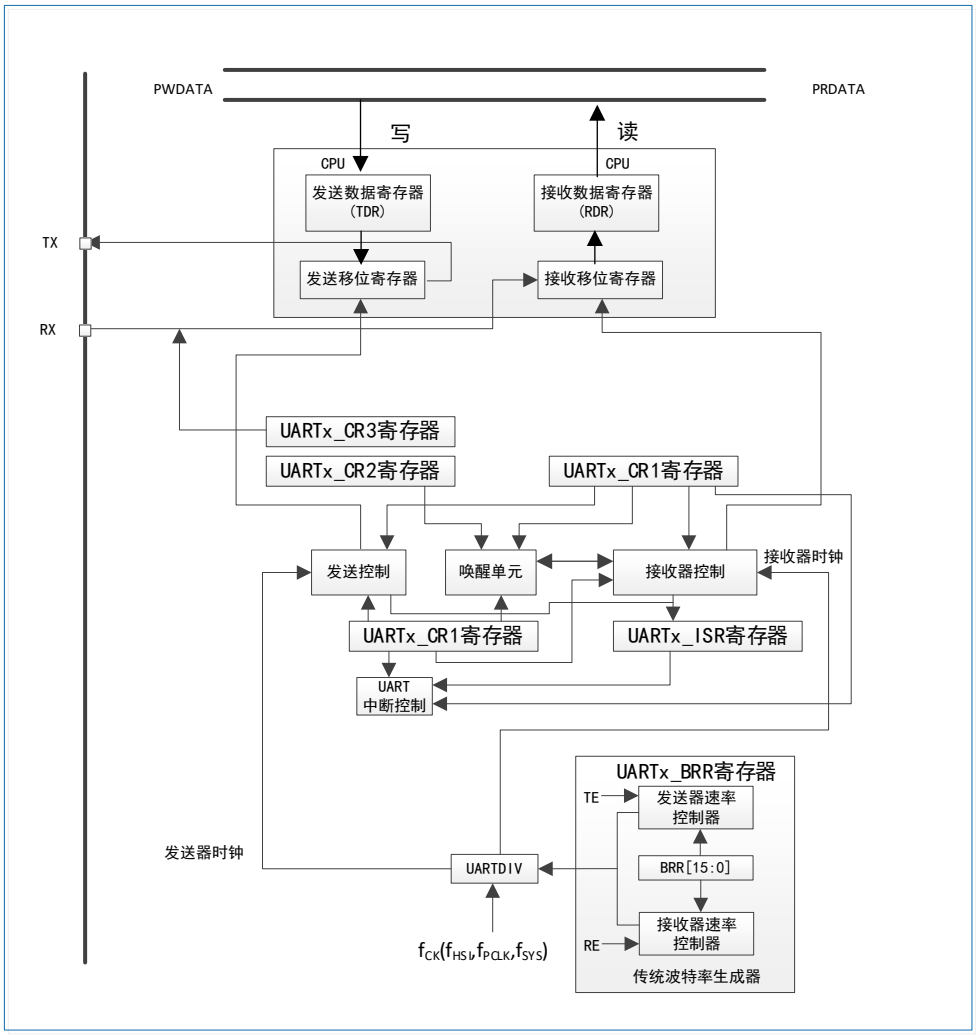

图 18-1 UART 框图

在图 18-1 中，关于在 UARTx_BRR 寄存器中编码 UARTDIV 的详细信息，请参见“18.3.4 UART 波特率生成”。f_{CK} 可以是 f_{HSI}、f_{PCLK} 或 f_{SYS}。

18.3.1 UART 字符说明

通过配置 UARTx_CR1 寄存器中的 M[1:0]位来选择 7 位、8 位或 9 位的字长。

- 7 位字符长度：M[1:0]=10
- 8 位字符长度：M[1:0]=00
- 9 位字符长度：M[1:0]=01

默认情况下，数据引脚（TX 或 RX）在起始位时处于低电平状态，在停止位时处于高电平状态。

通过极性配置（UARTx_CR2 寄存器中的 TXINV/RXINV 位）使 TX/RX 的有效电平反向。

空闲帧是指一帧数据的平值均为“1”。

中断帧是指一帧数据的电平值均为“0”。

下图给出了不同字长模式下的编程。

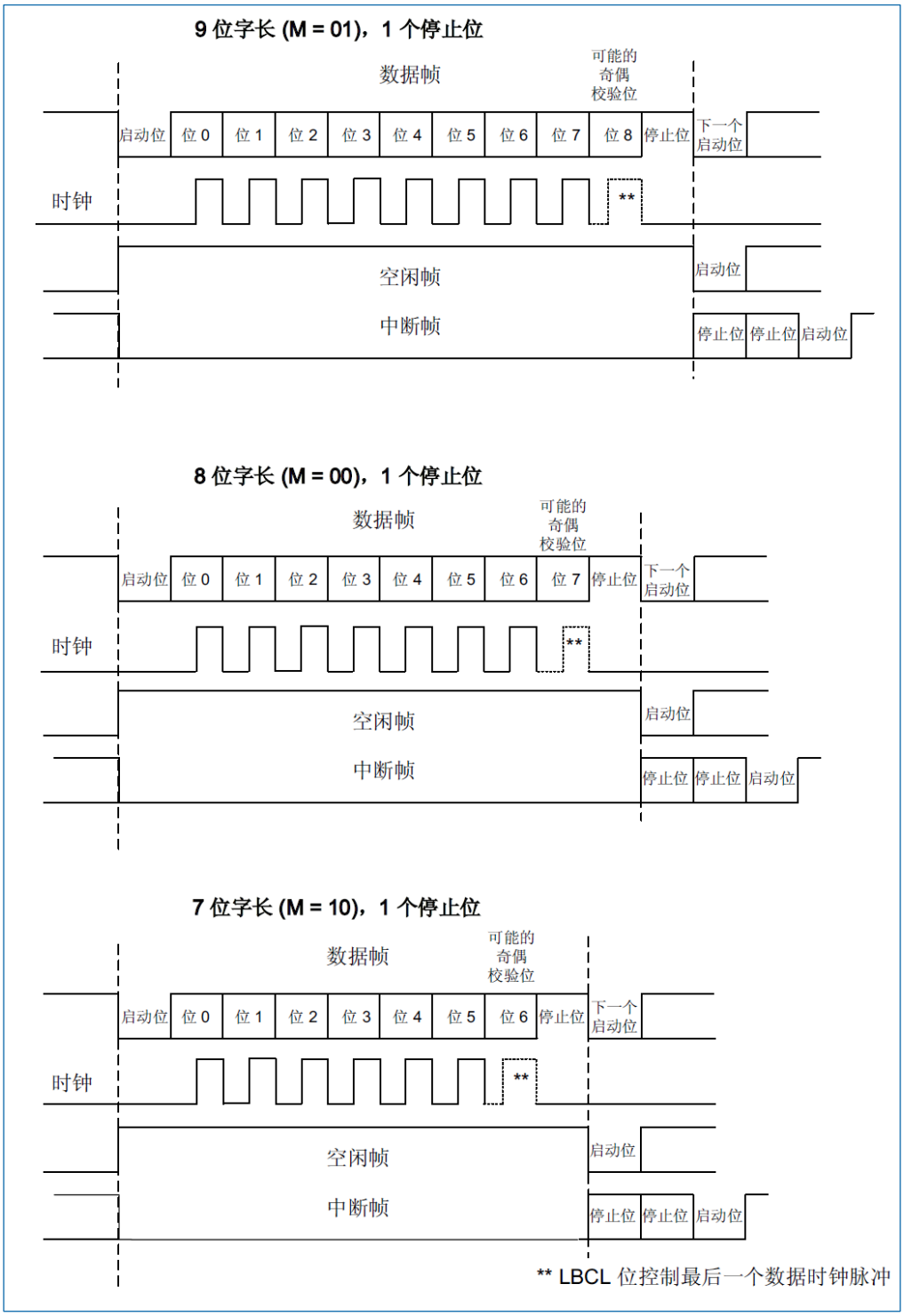

图 18-2 字长编程

18.3.2 UART 发送器

通过配置 UARTx_CR1 寄存器中的 M[1:0]位来选择发送器的数据字长（7 位、8 位或 9 位）。

通过配置 UARTx_CR1 寄存器中的发送使能位（TE），以使能发送器。发送移位寄存器中的数据通过 TX 引脚输出。

可配置的停止位

通过配置 UARTx_CR2 寄存器中的 STOP[1:0]位来选择停止位的个数。

- 1 个停止位：这是停止位数量的默认值。
- 2 个停止位：正常 UART 模式和单线模式支持该值。

空闲帧发送将包括停止位。

中断发送是 10 个低电平位（M[1:0]=00 时）、11 个低电平位（M[1:0]=01 时）或 9 个低电平位（M[1:0]=10 时），然后是 2 个停止位。无法传送长中断（中断长度大于 9/10/11 个低电平位）。

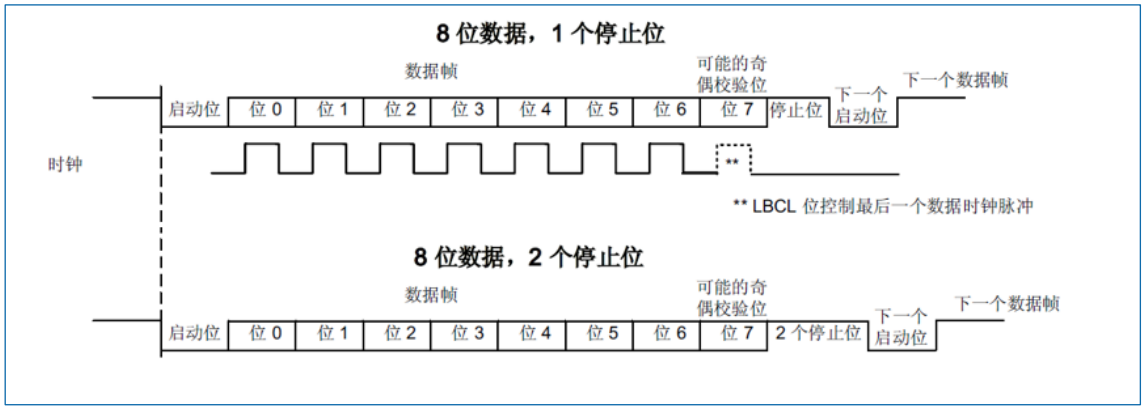

图 18-3 可配置的停止位

字符发送

UART 在发送字符的时候，默认配置（LSB）下 TX 引脚会先发送数据的最低有效位。每个字符都包含数据位、起始位以及停止位。

通过向数据发送寄存器（UARTx_TDR）写入数据可将 TXE 位清零。TXE 位由硬件置 1，它表示：

- 数据已从 UARTx_TDR 寄存器移到移位寄存器中并且数据开始发送。
- UARTx_TDR 寄存器为空。
- UARTx_TDR 寄存器中可写入下一个数据。

如果数据帧发送完成（停止位后）且 TXE 位置 1，这时 TC 位将变为高电平。若配置 UARTx_CR1 寄存器中的 TCIE 位，则会产生 TC 中断。向 UARTx_TDR 寄存器中写入最后一个数据后，必须等待至 TC=1，才可以保证数据准确无误地发送出去。

UART 字符发送时序如图 18-4 所示。

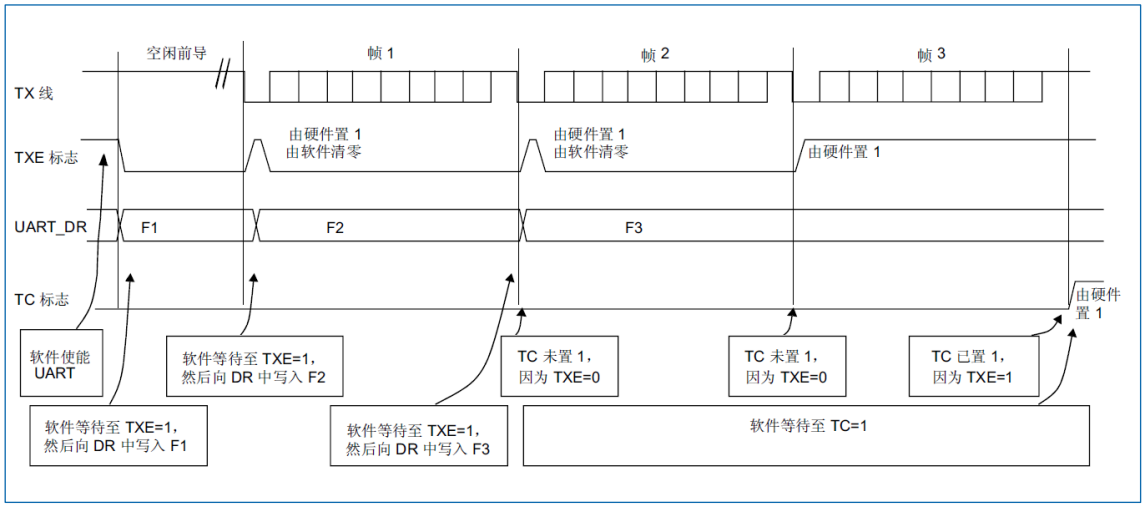

图 18-4 发送时的 TC/TXE 行为

字符发送配置步骤：

1. 配置 UARTx_CR1 中的 M 位来定义字长。
2. 配置 UARTx_BRR 寄存器来选择所需波特率。

3. 配置 UARTx_CR2 中的停止位（STOP[1:0]）个数。
4. 配置 UARTx_CR1 寄存器中的 UE 位来使能 UART 通信。
5. 配置 UARTx_CR1 中的 TE 位以便在首次发送时发送一个空闲帧。
6. 在 UARTx_TDR 寄存器中写入要发送的数据（该操作将清零 TXE 位）。重复这一步骤可满足连续发送数据要求。
7. UARTx_TDR 寄存器写入最后一个数据后，等待 TC=1，表明数据发送完成。

中断帧（BREAK）发送

若置位 UARTx_RQR 寄存器中的 SBKRQ 位，将发送一个中断帧。中断帧在整个字符周期内，电平始终为“0”。通过将 UARTx_ISR 中的 SBKF 位置 1 来发送中断字符，并且在中断字符发送完成后，SBKF 位清零。UART 在中断帧末尾插入 2 个停止位。

空闲帧发送

在初始情况下，将 UARTx_CR1 寄存器中的 TE 位置 1 会驱动 UART 在第一个数据帧之前发送一个空闲帧。空闲帧在整个字符周期内，电平始终为“1”。发送的空闲帧包含了停止位。

18.3.3 UART 接收器

通过配置 UART_CR1 寄存器中的 M 位来选择接收器接收的数据字长（7 位、8 位或 9 位），接收器支持 1 和 2 个停止位。

起始位检测

UART 支持 8/16 倍过采样，起始位检测序列相同。

在 UART 中，识别出特定序列的采样时会检测起始位。例如，此序列为：1 1 1 0 X 0 X 0X 0X 0 X 0X 0。

在过采样中，当进行第一次采样时，检测到第 3 位、第 5 位和第 7 位均为 0；同时进行第二次采样时，检测到第 8 位、第 9 位和第 10 位均为 0 时，则检测到起始位。当配置了 RXNE 和 RXNEIE 时，可以产生中断。

在检测到起始位（RXNE 标志位置 1）的情况下，以下描述均会导致 NF 噪声标志置位：3 个采样位中有 2 位为 0（第 3 位、第 5 位和第 7 位进行采样或第 8 位、第 9 位和第 10 位采样）。

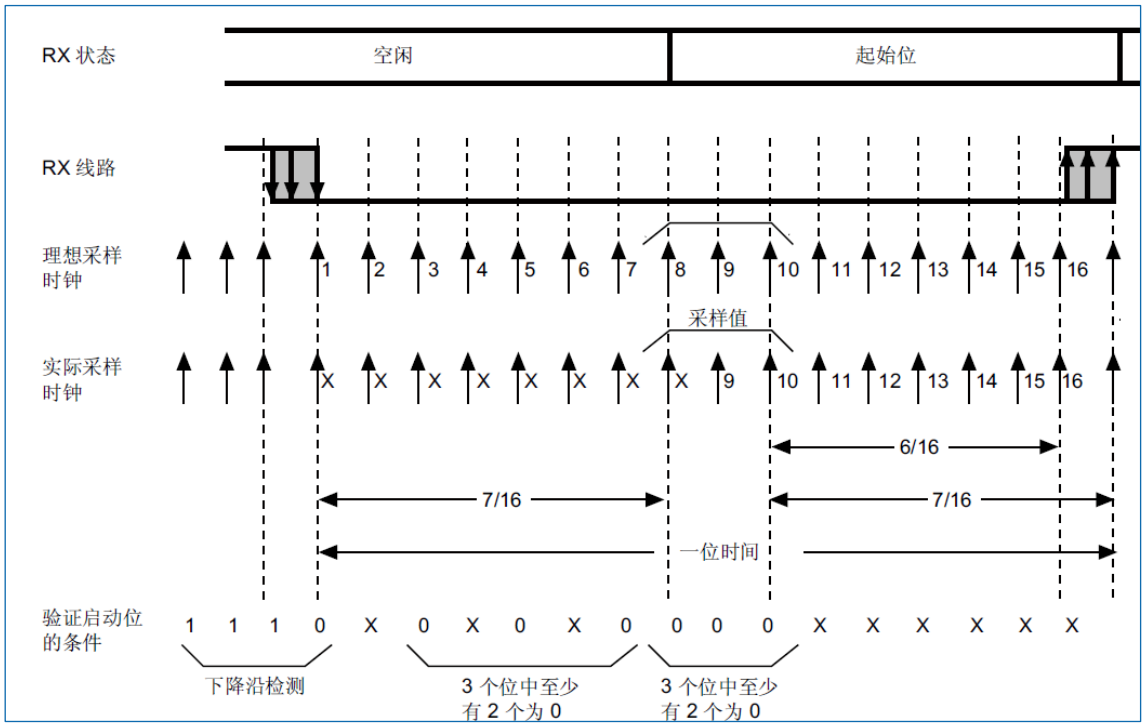

图 18-5 16 倍或 8 倍过采样时的起始位检测

说明：如果序列不完整，起始位检测将中止，接收器将返回空闲状态（无标志位置 1）等待下降沿。

字符接收

UART 接收期间，首先通过 RX 引脚移入数据的最低有效位（默认配置）。

配置字符接收的步骤如下：

1. 配置 UARTx_CR1 中的 M 位来定义字长。
2. 配置 UARTx_BRR 寄存器来选择所需波特率。
3. 配置 UARTx_CR2 中的停止位（STOP[1:0]）个数。
4. 配置 UARTx_CR1 寄存器中的 UE 位来使能 UART 通信。
5. 将 UARTx_CR1 寄存器中的 RE 位置 1，以便接收器搜索起始位。

接收到字符时：

- RXNE 位置 1。这表明移位寄存器的内容已传送到 RDR，且可进行读取操作。
- 如果 RXNEIE 位置 1，则会生成中断。
- 如果接收过程中检测到帧错误、噪声错误、上溢错误或校验位错误，错误标志位（FE、NF、ORE 或 PE）会被置 1。
- 在普通模式下，通过软件对 UARTx_RDR 寄存器执行读操作将 RXNE 位清零。也可以通过将 UARTx_RQR 寄存器中的 RXFRQ 位置 1 将 RXNE 标志位清零。RXNE 位必须在结束接收下一个字符前清零，否则会发生上溢错误。

中断帧接收

接收到中断字符时，UARTx_ISR 寄存器中的帧错误标志 FE 会置 1。

空闲帧接收

检测到空闲帧时，UARTx_ISR 寄存器中的 IDLE 位会被置 1；若 IDLEIE 位被置 1，则会产生中断。同时可以通过置位 UARTx_ICR 寄存器中的 IDLECF 位来对 IDLE 位进行清零。

上溢错误

当接收到一个字符且 RXNE 位未被复位时，将发生上溢错误。除非 RXNE 位被清零，否则移位寄存器中的数据不能传送到 RDR 寄存器。

每接收到一个字节后，RXNE 标志位都会被置 1。若在 RXNE 标志位未被清零时接收到字符，则会发生上溢错误。

发生上溢错误时：

- ORE 位被置 1。
- RDR 中的内容不会丢失。
- 移位寄存器中的数据将被覆盖，并且发生上溢错误时接收到的任何数据都将直接被丢弃。
- 如果 RXNEIE 位置 1 或 EIE 位置 1，则会生成中断。
- 通过设置 UARTx_ICR 寄存器中的 ORECF 位可将 ORE 位清零。

说明：ORE 位置 1 时表示至少 1 个数据丢失。存在两种可能：

- *如果 RXNE=1，则最后一个有效数据存储于接收寄存器 RDR 中并且可进行读取。*
- *如果 RXNE=0，则表示最后一个有效数据已被读取，因此 RDR 中没有要读取的数据。接收到新（和丢失）数据的同时已读取 RDR 中的最后一个有效数据时，会发生该情况。*

选择时钟源和合适的过采样方法

UART 在停机模式唤醒时支持双时钟域，时钟源可以是：PCLK（默认值）、HSI 或 SYSCLK。在低功耗模式下，UART 可以选择 HSI 作为时钟源来接收数据，从低功耗模式下唤醒的方式可以通过接收数据和唤醒模式的配置来实现。

接收器通过配置过采样方式，可从噪声中提取有效数据。这可在最大通信速度与抗噪声/时钟误差性能之间实现平衡。

可通过编程 UARTx_CR1 寄存器中的 OVER8 位来选择采样方式。采样时钟可以是波特率时钟的 16 倍或 8 倍。

- 选择 8 倍过采样（OVER8=1）以获得更高的传输速率（高达 $f_{CK}/8$），其中 f_{CK} 为时钟源频率。这种情况下，接收器对时钟偏差的最大容差将会降低（请参见“18.3.5 UART 接收器对时钟偏差的容差”）。
- 选择 16 倍过采样（OVER8=0）以增加接收器对时钟偏差的容差。这种情况下，最大传输速率限制为最高 $f_{CK}/16$。

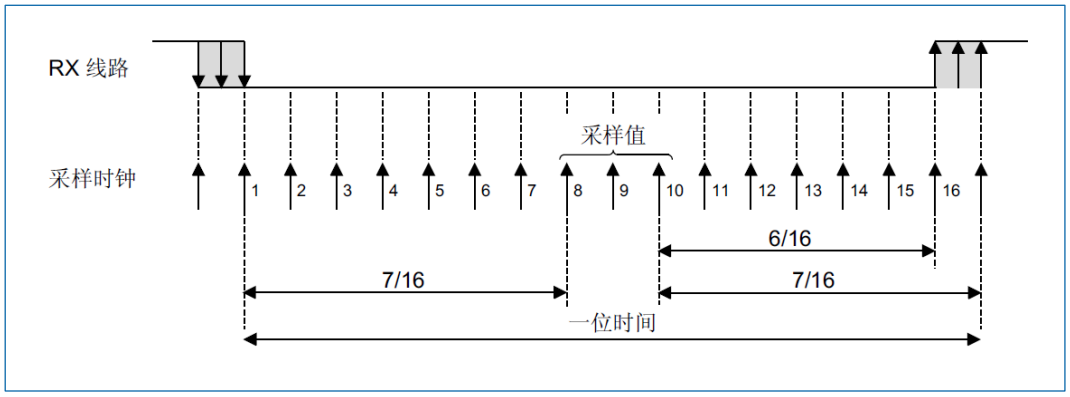

图 18-6 16 倍过采样时的数据采样

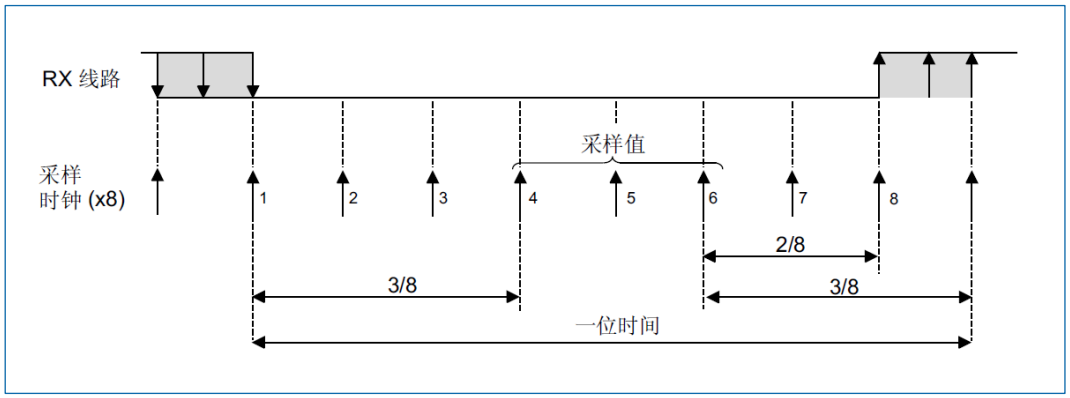

图 18-7 8 倍过采样时的数据采样

通过编程 UARTx_CR3 寄存器中的 ONEBIT 位选择逻辑电平的采样方式：

- 配置 ONEBIT=0 时，在已接收位的中心进行三次采样。在这种情况下，如果用于多数表决的 3 次采样结果不相等，则 NF 位被置 1。在噪声环境下工作时，选择三次采样的多数表决法。因为在检测到噪声时拒绝接收数据（表 18-2）可提高抗干扰能力。
- 配置 ONEBIT=1 时，在已接收位的中心进行单次采样。线路无噪声时请选择单次采样法（ONEBIT=1）以增加接收器对时钟偏差的容差（请参见“18.3.5 UART 接收器对时钟偏差的容差”）。

表 18-2 通过采样数据进行噪声检测

采样值	NF 状态	接收的位值
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

帧错误

如果接收数据时，未在预期时间内识别出停止位，帧错误标志位会被置 1。

检测到帧错误时：

- FE 位由硬件置 1。
- 无效数据从移位寄存器传送到 UARTx_RDR 寄存器。
- 单字节通信时无中断产生。然而，在 RXNE 位产生中断时，该位出现上升沿。多缓冲区通信时，UARTx_CR3 寄存器中的 EIE 位置 1 时将发出中断。通过将 1 写入 UARTx_ICR 寄存器中的 FECF 位来复位 FE 位。

18.3.4 UART 波特率生成

配置 UARTx_BRR 寄存器的 BRR[15:0]位（其中 BRR[3:0]：UARTDIV 的低 4 位，BRR[15:4]：UARTDIV 的高 12 位）来生成接收器和发送器（RX 和 TX）的波特率。

通过配置 UARTx_CR1 中的 OVER8 位，来选择 16 倍过采样 8 倍过采样。

16 倍过采样（OVER8=0）时的波特率生成公式为：

公式 18-1：

$$Tx（Rx）baud=\frac{f_{ck}}{UARTDIV}$$

8 倍过采样（OVER8=1）时的波特率生成公式为：

公式 18-2：

$$Tx（Rx）baud=\frac{2*f_{ck}}{UARTDIV}$$

其中 UARTDIV 的值由 BRR 位获得，支持 16 倍过采样和 8 倍过采样。

- 当 OVER8=0 时，BRR=UARTDIV。
- 当 OVER8=1 时：
  - BRR[2:0]=（UARTDIV[3:0]>>1）。
  - BRR[3]必须保持清零。
  - BRR[15:4]=UARTDIV[15:4]

说明：切勿在通信过程中修改波特率寄存器的值，16 倍或 8 倍过采样时，UARTDIV 必须大于或等于 0x10。

UARTx_BRR 寄存器中获取 UARTDIV 的示例如下：

- 当采用 16 倍过采样时，根据公式 18-1：
  - UARTDIV = fck/(Tx 或 Rx) baud= 8000 000/9600
  - BRR=UARTDIV=833D=0341h
- 当采用 8 倍过采样时，根据公式 18-2：
  - UARTDIV= 2 * fck/(Tx 或 Rx) baud = 2*8000 000/9600
  - UARTDIV=1666,66（1667D=683h）
  - BRR[3:0]=3h>>1=1h
  - BRR[15:0]=0x681

示例 2：通过 f_{CK}=32 MHz 获得 921.6K 波特率

- 当采用 16 倍过采样时，根据公式 18-1：
  - UARTDIV= fck/(Tx 或 Rx baud) =32000 000/921 600
  - BRR=UARTDIV=35D=23h
- 当采用 8 倍过采样时，根据公式 18-2：
  - UARTDIV=2 * fck/(Tx 或 Rx baud) =2*32000 000/921 600
  - UARTDIV=70D=46h
  - BRR[3:0]=UARTDIV[3:0]>>1=6h>>1=3h
  - BRR[15:0]=0x43

表 18-3 采用 16 倍或 8 倍过采样时，在 f_{ck}=48MHz 下编程波特率的误差计算 (1)

波特率		16 倍过采样（OVER8=0）			8 倍过采样（OVER8=1）		
序号	所需值	实际值	BRR	误差%= （计算值-所需值）/所需波特率	实际值	BRR	误差%

波特率		16 倍过采样（OVER8=0）			8 倍过采样（OVER8=1）		
1	2.4Kbps	2.4Kbps	0x4E20	0	2.4Kbps	0x9C40	0
2	9.6Kbps	9.6Kbps	0x1388	0	9.6Kbps	0x2710	0
3	19.2Kbps	19.2Kbps	0x9C4	0	19.2Kbps	0x1388	0
4	38.4Kbps	38.4Kbps	0x4E2	0	38.39Kbps	0x9C4	0
5	57.6Kbps	57.62Kbps	0x341	0.04	57.588Kbps	0x683	0.02
6	115.2Kbps	115.1Kbps	0x1A1	0.08	115.11Kbps	0x341	0.04
7	230.4Kbps	230.77Kbps	0xD0	0.16	230.21Kbps	0x1A1	0.8
8	460.8Kbps	461.5Kbps	0x68	0.16	461.5Kbps	0xD0	0.16
9	921.6Kbps	923.1Kbps	0x34	0.16	923.1Kbps	0x68	0.16
10	2Mbps	2Mbps	0x18	0	2Mbps	0x30	0
12	4Mbps	4Mbps	NA	NA	4Mbps	0x18	0

(1). CPU 时钟越低，特定波特率的精度就越低。这些数据可用于确定可达到的波特率上限。

18.3.5 UART 接收器对时钟偏差的容差

当总时钟系统偏差小于 UART 接收器的容差时，UART 异步接收器才能正常工作。影响总偏差的因素包括：

- DTRA：发送器误差引起的偏差（其中还包括发送器本地振荡器的偏差）。
- DQUANT：接收器的波特率量化引起的误差。
- DREC：接收器本地振荡器的偏差。
- DTCL：传输线路引起的偏差（通常是由于收发器所引起，它可能会在低电平到高电平转换时序与高电平到低电平转换时序之间引入不对称）。

DTRA＋DQUANT＋DREC＋DTCL＋DWU <UART 接收器的容差

其中，DWU 是从停机模式唤醒时因采样点偏差产生的误差。当 M 位取值不同时，对应的 DWU 值为：

- M[1:0]=00 时：$DWU=\frac{t_{WUUART}}{10*Tbit}$
- M[1:0]=01 时：$DWU=\frac{t_{WUUART}}{11*Tbit}$
- M[1:0]=10 时：$DWU=\frac{t_{WUUART}}{9*Tbit}$

t_{WUUART}是从检测到唤醒事件到时钟（由外设请求）与调压器均就绪的时间。

UART 接收器在表 18-4 中指定的最大容许偏差下可正确接收数据，取决于以下选择：

- 由 UARTx_CR1 寄存器中的 M 位定义的 9 位、10 位或 11 位字符长度。
- 由 UARTx_CR1 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样。
- UARTx_BRR 寄存器的 BRR[3:0]位等于或不等于 0000。
- 使用 1 位或 3 位对数据进行采样，取决于 UARTx_CR3 寄存器中 ONEBIT 位的值。

表 18-4 BRR[3:0]=0000 时的 UART 接收器容差

M[1:0]位	OVER8 位=0		OVER8 位=1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
00	3.75%	4.375%	2.50%	3.75%
01	3.41%	3.97%	2.27%	3.41%
10	4.16%	4.86%	2.77%	4.16%

表 18-5 BRR[3:0]！=0000 时的 UART 接收器容差

M[1:0]位	OVER8 位=0		OVER8 位=1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
00	3.33%	3.88%	2%	3%
01	3.03%	3.53%	1.82%	2.73%
10	3.7%	4.31%	2.22%	3.33%

说明：当接收的帧恰好包含 10 个（M[1:0]位=00）、11 个（M[1:0]位=01）或 9 个（M[1:0]位=10）位持续时间的空闲帧时，表 18-4 和表 18-5 中指定的数据可能与特例中的数据略微不同。

18.3.6 使用 UART 进行多处理器通信

在多处理器通信中，将其中的一个 UART 作为主 UART，其 TX 输出与其他 UART 的 RX 输入相连接。其他 UART 作为从 UART，其各自的 TX 在逻辑上通过与运算连接在一起，并且与主 UART 的 RX 输入相连接。

在多处理器配置中，一般情况下只有预期的消息接收方主动接收完整的消息内容，以减少由所有未被寻址的接收器造成的冗余 UART 服务开销。该功能可以通过将未被寻址的器件配置为静默模式来实现。将 UARTx_CR1 寄存器中的 MME 位置 1，并且，将 UARTx_RQR 寄存器的 MMRQ 位写 1，方可进入静默模式。

在静默模式下：

- 任何接收状态位都为 0。
- 任何接收中断都被禁止。
- UARTx_ISR 寄存器中的 RWU 位置 1。

配置 UARTx_CR1 寄存器中 WAKE 位，UART 可使用以下两种方法进入或退出静默模式：

- 若 WAKE 位=0，则进行空闲线路检测。
- 若 WAKE 位=1，则进行地址标记检测。

空闲线路检测（WAKE=0）

向 UARTx_RQR 寄存器的 MMRQ 位写入 1，UART 进入静默模式（RWU 位置 1）。

当检测到空闲帧时，UART 会被唤醒。此时 RWU 位会被硬件清零。空闲线路检测时静默模式如下图所示：

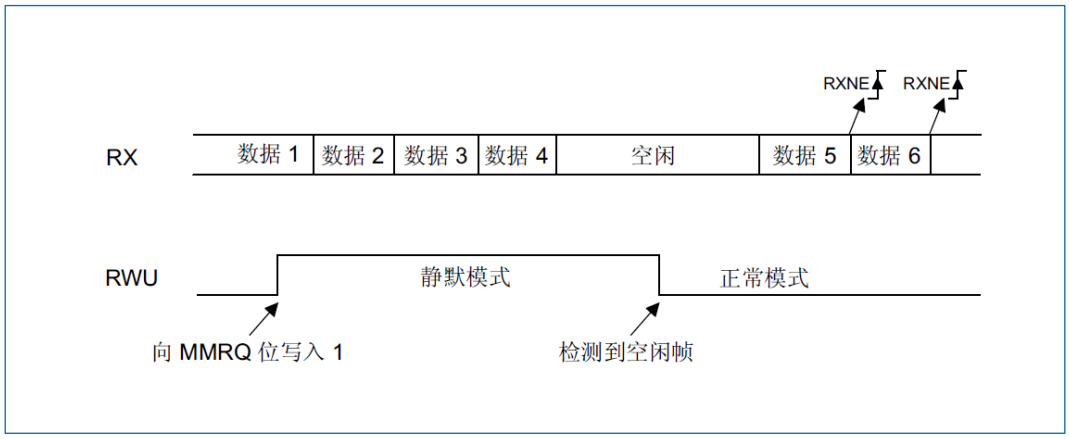

图 18-8 使用空闲线路检测时的静默模式

4 位/7 位地址标记检测（WAKE=1）

在此模式下，如果字节的 MSB 为 1，则将这些字节识别为地址，否则将其识别为数据。通过配置 UARTx_CR2 中的 ADDM7 位来选择 4 位/7 位的地址匹配方式，然后接收器会将此 4 位/7 位字与其地址进行比较。

当接收到与其编程地址不匹配的地址字符时，UART 会进入静默模式。此时，RWU 位将由硬件置 1。UART 进入静默模式后，接收的地址字符不会置位 RXNE 标志。

当接收到与编程地址匹配的地址字符时，UART 会退出静默模式。此时，RWU 位被清零，UART 可开始正常接收后续字节。由于 RWU 位已清零，RXNE 位会针对地址字符置 1。

下图中给出了使用地址标记检测时静默模式行为的示例。

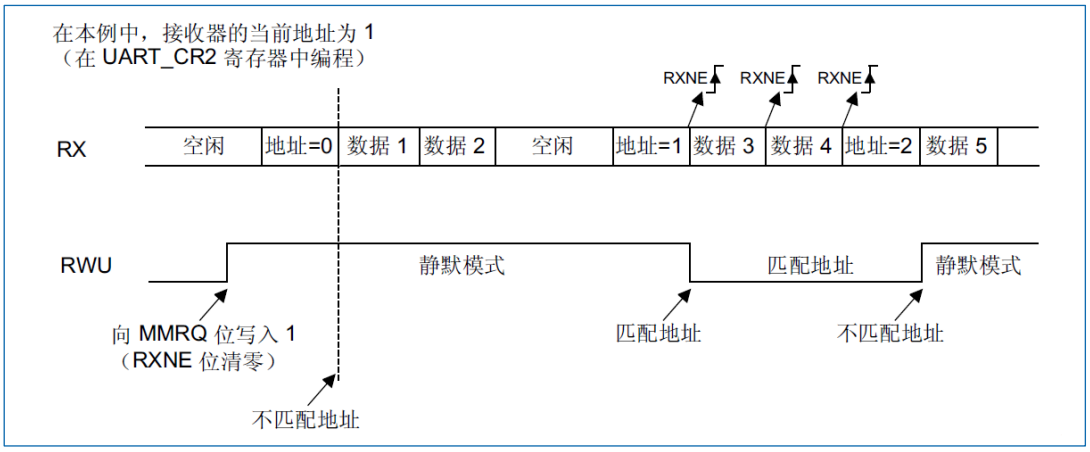

图 18-9 使用地址标记检测时的静默模式

说明：在 7 位和 9 位数据模式下，地址检测分别在 6 位和 8 位地址上完成（ADD[5:0]和 ADD[7:0]）。

18.3.7 UART 奇偶校验

将 UARTx_CR1 寄存器中的 PCE 位置 1，可以使能奇偶校验控制（发送时生成奇偶校验位，接收时进行奇偶校验检查）。根据 M[1:0]位定义的帧长度，下表列出了可能的 UART 帧格式。

表 18-6 帧格式

M[1:0]位	PCE 位	UART 帧 (1)
00	0	|SB|8 位数据|STB|
00	1	|SB|7 位数据|PB|STB|

M[1:0]位	PCE 位	UART 帧(1)
01	0	|SB|9 位数据|STB|
01	1	|SB|8 位数据|PB|STB|
10	0	|SB|7 位数据|STB|
10	1	|SB|6 位数据|PB|STB|

(1). SB：起始位，STB：停止位，P：奇偶校验位。在数据寄存器中，PB 始终位于 MSB 位置（第 7 位、第 8 位或第 9 位，具体取决于 M[1:0]位的值）。

偶校验

计算奇偶校验位，使数据帧和奇偶校验位中“1”的数量为偶数。例如，如果数据=00110101，其中 4 个数据位被置 1，在选择偶校验（UARTx_CR1 寄存器中的 PS 位=0）时，校验位则为 0。

奇校验

计算奇偶校验位，使数据帧和奇偶校验位中“1”的数量为奇数。例如，如果数据=00110101，其中 4 个数据位被置 1，在选择奇校验（UARTx_CR1 寄存器中的 PS 位=1）时，校验位则为 1。

接收时进行奇偶校验检查

如果奇偶校验检查失败，则 UARTx_ISR 寄存器中的 PE 标志被置 1；如果 UARTx_CR1 寄存器中 PEIE 位被置 1，则会生成中断。通过软件置位 UARTx_ICR 寄存器中的 PECF 位，可清零 PE 标志位。

发送时的奇偶校验生成

如果 UARTx_CR1 寄存器中的 PCE 位被置 1，则写入数据寄存器（UARTx_TDR）中的数据的 MSB 位会被奇偶校验位更改，如果选择偶校验（PS=0），则“1”的数量为偶数；如果选择奇校验（PS=1），则“1”的数量为奇数。

18.3.8 UART 单线半双工通信

通过将 UARTx_CR3 寄存器中的 HDSEL 位置 1 来选择单线半双工模式。

UART 支持单线半双工协议，其中 TX 和 RX 线路从内部相连接。一旦 UARTx_CR3 寄存器中的 HDSEL 位置 1：

- TX 和 RX 线路从内部相连接。
- RX 引脚被禁用，通信双方通过 TX 连接在一起。
- TX 在空闲状态或接收过程中用作标准 I/O。在单线半双工模式下，TX 需配置为复用功能开漏并外接上拉电阻。除此之外，通信协议与正常 UART 模式下的通信协议相似。此线路上的任何冲突必须由软件管理。

18.4 UART 中断

表 18-7 UART 中断请求

中断事件	事件标志位	使能控制位
发送数据寄存器为空	TXE	TXEIE
发送完成	TC	TCIE
接收数据寄存器非空	RXNE	RXNEIE

中断事件	事件标志位	使能控制位
上溢错误	ORE	RXNEIE
检测到空闲线路	IDLE	IDLEIE
奇偶校验错误	PE	PEIE
噪声错误、上溢错误和帧错误	NF 或 ORE 或 FE	EIE

UART 中断映射图如下：

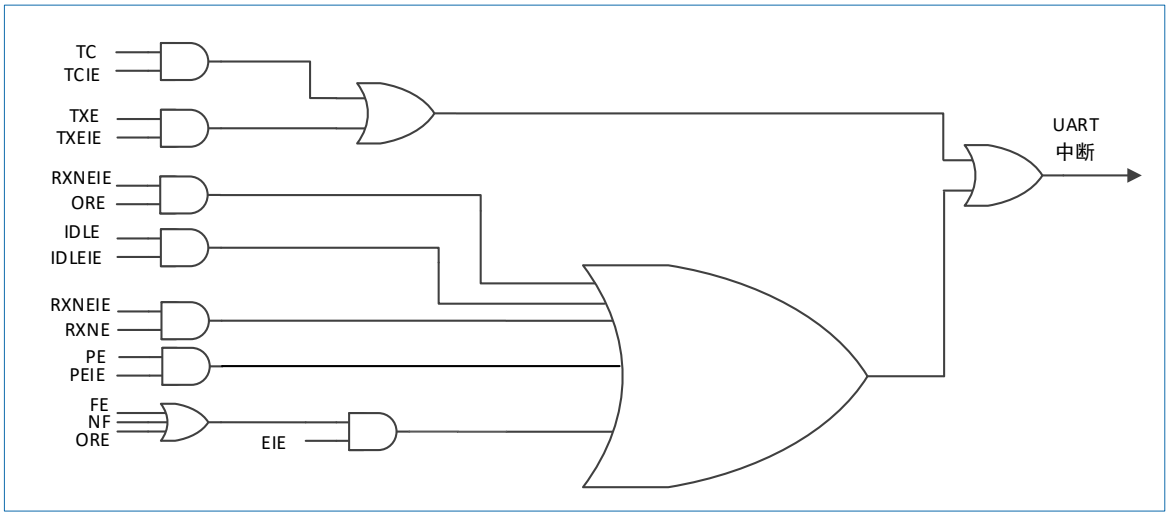

图 18-10 UART 中断映射图

18.5 UART 寄存器

基地址：（UART1, UART2）=（0x4001 3800, 0x4001 3C00）

空间大小：（UART1, UART2）=（0x400, 0x400）

18.5.1 控制寄存器 1（UARTx_CR1）（x=1..2）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			M1	Res											
			rw												

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	Res	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	Res	UE
rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

位 31:29	Res：保留 必须保持复位值。
位 28	M1：字长，M1=M[1]（Word length） 该位和 M0 位配合使用，以配置字长。 • M[1:0]=00：1 个起始位，8 个数据位，n 个停止位 • M[1:0]=01：1 个起始位，9 个数据位，n 个停止位

	• M[1:0]=10：1 个起始位，7 个数据位，n 个停止位
位 27:16	Res：保留 必须保持复位值。
位 15	OVER8：过采样模式（Oversampling mode） • 0：16 倍过采样 • 1：8 倍过采样 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 14	Res：保留 必须保持复位值。
位 13	MME：静默模式使能（Mute mode enable） 该位开启 UART 的静默模式功能。当置为 1 时，UART 可以在活动模式和静默模式之间切换，和 WAKE 位的定义一样，由软件置 1 和清零。 • 0：接收器长期处于活动模式。 • 1：接收器可以在活动模式和静默模式间切换。
位 12	M0：字长，M0=M[0]（Word length） 该位决定串口字长。由软件置 1 和清零。 • 0：1 个起始位，8 位数据位，n 个停止位 • 1：1 个起始位，9 个数据位，n 个停止位 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 11	WAKE：接收器唤醒方式（Receiver wakeup method） 该位决定 UART 从静默模式唤醒的方式，由软件置 1 和清零。 • 0：空闲线 • 1：地址标记 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 10	PCE：校验控制使能（Parity control enable） 该位选择硬件校验控制（产生和检测）功能。当校验控制被打开，计算好的校验位被插入到最高位（M=0b01 时，检验位是第九位，M=0b00 时，检验位是第八位），并检测接收数据的校验位。由软件置 1 和清零。一旦该位被置 1，在当前字节之后就激活了校验控制（在收发的时候都有）。 • 0：校验控制禁止 • 1：校验控制使能 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 9	PS：校验选择（Parity selection） 该位选择在校验生成和检测功能被打开的时候（PCE=1）使用奇校验还是使用偶校验。由软件置 1 和清零。校验方式会在当前字节结束后生效。 • 0：偶校验 • 1：奇校验

	该位域只能在 UART 未被使能的时候（UE=0）改写。
位 8	PEIE：校验错误中断使能（PE interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：在 UART_ISR 寄存器中的 PE 被置 1 的时候，产生 UART 中断。
位 7	TXEIE：发送寄存器空中断使能（TXE interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：在 UART_ISR 寄存器中的 TXE 被置 1 的时候，产生 UART 中断。
位 6	TCIE：发送完毕中断使能（Transmission complete interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：在 UART_ISR 寄存器中的 TC 位被置 1 的时候，产生 UART 中断。
位 5	RXNEIE：接收寄存器非空中断使能（RXNE interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：在 UART_ISR 寄存器中的 ORE 或者 RXNE 被置 1 的时候，会产生 UART 中断。
位 4	IDLEIE：空闲中断使能（IDLE interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：在 UART_ISR 寄存器中的 IDLE 位被置 1 的时候，产生 UART 中断
位 3	TE：发送器使能（Transmitter enable） 该位打开发送器。由软件置 1 和清零。 • 0：发送器关闭 • 1：发送器打开，当 TE 被置为 1 后，和发送开始之间有 1 个位的延迟时间。
位 2	RE：接收器使能（Receiver enable） 该位打开接收器。由软件置 1 和清零。 • 0：接收器被关闭 • 1：接收器被打开并开始等待起始位
位 1	Res：保留 必须保持复位值。
位 0	UE：UART 使能（UART enable） 当该位被清零，UART 的预分频器和输出都立即停止，并且当前的操作也被取消。对 UART 的设置都不会丢，但 UART_ISR 中所有的状态标志都会被复位。由软件置 1 和清零。 • 0：UART 预分频器和输出关闭，降低功耗。

	• 1：UART 开启。

18.5.2 控制寄存器 2（UARTx_CR2）（x=1..2）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								Res				MSBFIRST	DATAINV	TXINV	RXINV
rw												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		STOP[1:0]		Res							ADDM7	Res			
		rw									rw				

位 31:24	ADD[7:0]：UART 的节点地址（Address of the UART node） • 位域[7:4]给出 UART 节点的地址或等待确认的字符码。在多机通讯并且进入静默状态或者 Stop 模式的时候，该位域用于 7 位地址标记的检测。发送器发出的字符的最高位应该为 1。也用在正常接收过程中的字符检测中，这时不打开静默状态。 该位域只能在接收器被关闭（RE=0）或者在 UART 被关闭的时候（UE=0）才能改写。 • 位域[3:0]给出 UART 节点的地址或等待确认的字符码。在多机通讯并且进入静默状态或者 Stop 模式的时候，位[3:0]用于 7 位地址标记的检测。 该位域只能在接收器被关闭（RE=0）或者在 UART 被关闭的时候（UE=0）才能改写。
位 23:20	Res：保留 必须保持复位值。
位 19	MSBFIRST：高位在前（Most significant bit first） 该位由软件置 1 和清零。 • 0：数据在发送和接收的时候，采用起始位在前，后面跟着第 0 位的顺序。 • 1：数据在发送和接收的时候，采用起始位在前，后面跟着最高位（位 7、位 8 或位 9）的顺序。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 18	DATAINV：二进制数反向（Binary data inversion） 该位由软件置 1 和清零。 • 0：数据寄存器中的逻辑数据在发送和接收的时候，采用正/直接逻辑（1=H、0=L）。 • 1：数据寄存器中的逻辑数据在发送和接收的时候，采用负/反向逻辑（1=L、0=H）。 校验位也一样反向。该位域只能在 UART 未被使能的时候（UE=0）改写。
位 17	TXINV：TX 脚有效电平反向（TX pin active level inversion） 该位由软件置 1 和清零。 • 0：TX 脚信号工作于标准逻辑电平（V_{DD}=1/idle、Gnd=0/mark）。 • 1：TX 脚信号被反向。（V_{DD}=0/mark、Gnd=1/idle，这可以用于 TX 线上带有外部反相

	器的时候。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 16	RXINV：RX 脚有效电平反向（RX pin active level inversion） 该位由软件置 1 和清零。 • 0：RX 脚信号工作于标准逻辑电平（VDD=1/idle、Gnd=0/mark）。 • 1：RX 脚信号被反向（VDD=0/mark、Gnd=1/idle）。这可以用于 RX 线上带有外部反相器的时候。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 15:14	Res：保留 必须保持复位值。
位 13:12	STOP[1:0]：停止位（STOP bits） 该位域用来定制停止位的个数。 • 00：1 个停止位 • 10：2 个停止位 • 其他值：保留 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 11:5	Res：保留 必须保持复位值。
位 4	ADDM7：7 位地址检测或 4 位地址检测选择（With 7-bit Address Detection/4-bit Address Detection） 该位用来选择使用 4 位还是 7 位地址检测。 • 0：4 位地址检测 • 1：7 位地址检测（8 位数据模式下） 该位只能在 UART 未被使能的时候（UE=0）改写。
位 3:0	Res：保留 必须保持复位值。

18.5.3 控制寄存器 3（UARTx_CR3）（x=1..2）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			OVERDIS	ONEBIT	Res							HDSEL	Res		EIE
			rw	rw								rw			rw

位 31:13	Res：保留

	必须保持复位值。
位 12	OVERDIS：溢出检测禁止（Overrun Disable） 该位用于禁止对接收溢出现象的检测。 • 0：当之前接收到的数据没有在新接收到数据之前读走时，会引起溢出错误，ORE 被硬件置 1。 • 1：溢出检测功能关闭。如果在新的接收数据到来时，RXNE 标志仍然是 1，但 ORE 标志还不是 1 时，新的数据会将 UART_RDR 中以前的内容覆盖掉。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 11	ONEBIT：单次采样方式使能（One sample bit method enable） 该位允许用户选择采样方式。当选择单次采样方式的时候，噪声监测标志（NF）就被禁止了。 • 0：三次采样方式 • 1：单次采样方式 该位只能在 UART 未被使能的时候（UE=0）改写。
位 10:4	Res：保留 必须保持复位值。
位 3	HDSEL：半双工选择（Half-duplex selection） 该位选择单线半双工模式。 • 0：不选择半双工模式 • 1：选择半双工模式 该位只能在 UART 未被使能的时候（UE=0）改写。
位 2:1	Res：保留 必须保持复位值。
位 0	EIE：错误中断使能（Error interrupt enable） 在允许帧错误、溢出错误或噪声错误产生中断请求时，需要打开这个开关。 • 0：中断禁止 • 1：当 UART_ISR 寄存器中的 FE=1 或 ORE=1 或 NF=1 时，会产生中断。

18.5.4 波特率寄存器（UARTx_BRR）（x=1..2）

偏移地址：0x0C

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:4	BRR[15:4]：UARTDIV 的高 12 位（High 12 bits of UARTDIV） 这 12 位定义 UART 分频器除法因子的高 12 位。
位 3:0	BRR[3:0]：UARTDIV 的低 4 位（Low 4 bits of UARTDIV） 当 OVER8=0，BRR[3:0]=UARTDIV[3:0]。当 OVER8=1，BRR[3:0]=UARTDIV[3:0]右移 1 位。

18.5.5 请求寄存器（UARTx_RQR）（x=1..2）

偏移地址：0x18

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												RXFRQ	MMRQ	SBKRQ	Res
												w	w	w	

位 31:4	Res：保留 必须保持复位值。
位 3	RXFRQ：接收数据清空请求（Receive data flush request） 对该位写 1 会直接令 RXNE 标志被硬件清零。这里允许直接丢弃还没有读的接收数据，以免提示溢出错误。
位 2	MMRQ：静默模式请求（Mute mode request） 对该位写 1 将导致 UART 进入静默模式，同时清除 RWU 标志。
位 1	SBKRQ：请求发送断开字符（Send break request） 对该位写 1 会令 SBKF 标志置 1，并在发送状态机可用的时候向线路发出一个断开字符。
位 0	Res：保留 *必须保持复位值。*

18.5.6 中断和状态寄存器（UARTx_ISR）（x=1..2）

偏移地址：0x1C

复位值：0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									REACK	TEACK	Res	RWU	SBKF	Res	BUSY
									r	r		r	r		r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								TXE	TC	RXNE	IDLE	ORE	NF	FE	PE

	r	r	r	r	r	r	r	r

位 31:23	Res：保留 必须保持复位值。
位 22	REACK：接收使能通知标志（Receive enable acknowledge flag） 该位由硬件控制，当接收使能信号被 UART 读到的时候，会给出一个回执。这可以在进入 Stop 模式之前，用来确认 UART 是不是已经准备好接收了。
位 21	TEACK：发送使能通知标志（Transmit enable acknowledge flag） 该位由硬件控制，当发送使能信号被 UART 读到的时候，会给出一个回执。这可以在写 UART_CR1 寄存器的 TE=0 以产生一个空闲帧请求，然后又将 TE 写成 1 的时候，用于保证 TE=0 的最小周期的时候用。
位 20	Res：保留 必须保持复位值。
位 19	RWU：接收器从静默模式唤醒（Receiver wakeup from Mute mode） 该位表示 UART 处于静默模式时，当唤醒和静默模式切换时，由硬件清零和置 1。静默模式控制序列（地址还是空闲）用 UART_CR1 寄存器的 WAKE 位来选择。如果选择由空闲信号唤醒，那该位就只能由软件置 1 了，方法是向 UART_RQR 寄存器的 MMRQ 位置 1。 • 0：接收器处于活动模式 • 1：接收器处于静默模式
位 18	SBKF：断开信号发送标志（Send break flag） 该位表示当要求发送一个断开字符，由软件置 1。其方法是向 UART_RQR 寄存器的 SBKRQ 位写 1。当断开字符的停止位传出后，由硬件自动清零。 • 0：不发送断开字符 • 1：断开字符将会被发送
位 17	Res：保留 必须保持复位值。
位 16	BUSY：忙标志（Busy flag） 该位由硬件置 1 和清零。当 RX 线在通讯时（成功检测到起始位），该位被硬件置 1。接收结束后（不管成功与否）会由硬件清零。 • 0：UART 处于空闲（无接收） • 1：正在接收数据
位 15:8	Res：保留 必须保持复位值。
位 7	TXE：发送数据寄存器空（Transmit data register empty） 当 UART_TDR 寄存器中的值被取到移位寄存器的同时，该位被硬件置 1。再向 UART_TDR 寄存器写数据就会同时清掉该位。如果 UART_CR1 寄存器中的 TXEIE 位被置起时，则会产生中断。

	• 0：没有数据被传到移位寄存器。 • 1：有数据被传到移位寄存器，发送数据寄存器为空。
位 6	TC：发送完成（Transmission complete） 在 TXE 为 1 的条件下，当数据发送完成的时候，该位会被硬件置 1。如果 UART_CR1 寄存器中的 TCIE 位是 1，就会产生中断请求。向 UART_TDR 寄存器再次写入数据，或者向 UART_ICR 寄存器的 TCCF 位写 1，都可以清除这个标志。 • 0：发送未完成 • 1：发送已完成
位 5	RXNE：接收数据寄存器非空（Receive data register not empty） 当接收移位寄存器的内容被传递到 UART_RDR 寄存器中时，该位被硬件置 1。读取 UART_RDR 寄存器的数据就会同时清掉该位。RXNE 标志也可以通过对 UART_RQR 寄存器中的 RXFRQ 位写 1 来清除。如果 UART_CR1 寄存器中的 RXNEIE 位是 1，就会产生中断请求。 • 0：没收到数据 • 1：收到的数据已经可读
位 4	IDLE：空闲线检测到（Idle line detected） 当检测到线路空闲时由硬件置 1。如果 UART_CR1 寄存器中的 IDLEIE 位是 1，就会产生中断请求。由软件向 UART_ICR 寄存器的 IDLECF 位写 1，可以清除这个标志。 • 0：没有检测到线路空闲 • 1：检测到线路空闲
位 3	ORE：溢出错误（Overrun error） 在 RXNE=1 的条件下（也就是上次数据还没有读走），串口接收寄存器又接收好了一个字节的数据并准备往 RDR 寄存器去转移的时候，会由硬件将该位置 1。由软件向 UART_ICR 寄存器的 ORECF 位写 1，可以清除这个标志。如果 UART_CR1 寄存器中的 RXNEIE 位或 EIE 位是 1，就会产生中断请求。 • 0：没有溢出错误 • 1：检测到溢出错误
位 2	NF：噪声检测标志（START bit Noise detection flag） 当收帧的时候检测到噪声，该位由硬件置 1。由软件向 UART_ICR 寄存器的 NFCF 位写 1，可以清除这个标志。 • 0：没有检测到噪声 • 1：检测到噪声
位 1	FE：帧错误（Framing error） 当一个不同步现象、强噪声或一个断开符号被检测到的时候，该位由硬件置 1。由软件向 UART_ICR 寄存器的 FECF 位写 1，可以清除这个标志。如果 UART_CR1 寄存器中的 EIE 位是 1，会产生中断请求。 • 0：没有检测到帧错误 • 1：有检测到帧错误或者有收到断开字符
位 0	PE：校验错误标志（Parity error）

	当在接收数据的时候发现校验错误，该位会由硬件置 1。由软件向 UART_ICR 寄存器的 PECF 位写 1，可以清除这个标志。如果 UART_CR1 寄存器中的 PEIE 位是 1，会产生中断请求。 • 0：没有校验错误 • 1：有校验错误

18.5.7 中断标志清除寄存器（UARTx_ICR）（x=1..2）

偏移地址：0x20

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									TCCF	Res	IDLECF	ORECF	NCF	FECF	PECF
									w_r1		w_r1	w_r1	w_r1	w_r1	w_r1

位 31:7	Res：保留 必须保持复位值。
位 6	TCCF：发送完成标志的清除（Transmission complete clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 TC 标志位。
位 5	Res：保留 必须保持复位值。
位 4	IDLECF：线路空闲检测标志的清除（Idle line detected clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 IDLE 标志位。
位 3	ORECF：溢出错误标志的清除（Overrun error clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 ORE 标志位。
位 2	NCF：噪声检测标志的清除（Noise detected clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 NF 标志位。
位 1	FECF：帧错误标志的清除（Framing error clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 FE 标志位。
位 0	PECF：校验错误标志的清除（Parity error clear flag） 对该位写 1，会清除 UART_ISR 寄存器中的 PE 标志位。

18.5.8 数据接收寄存器（UARTx_RDR）（x=1..2）

偏移地址：0x24

复位值：0xXXXX XXXX

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

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

位 31:9	Res：保留 必须保持复位值。
位 8:0	RDR[8:0]：接收数据的值（Receive data value） 该位域用于写入已接收的数据字节。 RDR 寄存器提供输入移位寄存器和内部总线间的并行接口。当接收数据时打开了校验位，读这个寄存器得到的最高位是校验位。

18.5.9 数据发送寄存器（UARTx_TDR）（x=1..2）

偏移地址：0x28

复位值：0xXXXX XXXX

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

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

位 31:9	Res：保留 必须保持复位值。
位 8:0	TDR[8:0]：发送数据的值（Transmit data value） 该位域用于写入要发送的数据字节。 TDR 寄存器提供发送移位寄存器和内部总线间的并行接口。当发送的时候设置了校验功能（UART_CR1 中的 PCE=1），向最高位（位 7、位 8 还是位 9 取决于设置的字长）写入的信息是无效的，因为它总是要被校验位代替之后再发送。

19 内部集成电路接口（I2C）

内部集成电路（I2C）总线接口处理 MCU 与串行 I2C 总线间的通信。它遵循 I2C 规范，支持标准模式（Standard mode，Sm）和快速模式（Fast mode，Fm）。

它与系统管理总线（System management bus，SMBus）和电源管理总线（Power management bus，PMBus）兼容。

19.1 I2C 主要特性

- 兼容 I2C 总线规范第 03 版
  - 支持从模式和主模式
  - 支持多主设备
  - 支持标准速度模式（高达 100 kbit/s）
  - 支持快速模式（高达 400 kbit/s）
  - 支持 7 位和 10 位寻址模式
  - 支持多个 7 位从地址（2 个从设备地址寄存器，1 个具有可配置的掩码位段）
  - 所有 7 位地址应答模式
  - 支持广播呼叫
  - 支持总线上的数据建立和保持时间可软件配置
  - 支持事件管理
  - 支持时钟延展功能
  - 支持软件复位
- 可编程模拟和数字噪声滤波器
- 兼容 SMBus 规范第 2.0 版
  - 支持硬件数据包错误校验（Packet Error Checking，PEC）生成和验证
  - 支持命令和数据应答控制
  - 支持地址解析协议（Address Resolution Protocol，ARP）
  - 支持主机和从设备
  - 支持 SMBus 报警
  - 支持超时和空闲条件检测
- 兼容 PMBus 第 1.1 版标准
- 独立时钟：选择独立时钟源可使 I2C 通信速度不受 PCLK 时钟频率更改的影响
- 地址匹配时从停机模式唤醒

19.2 I2C 功能说明

该接口通过数据引脚（SDA）和时钟引脚（SCL）连接到 I2C 总线。它可以连接到标准速度（高达 100 kbit/s）或快速（高达 400 kbit/s）I2C 总线。

该接口也可通过数据引脚（SDA）和时钟引脚（SCL）连接到 SMBus。还可使用额外的 SMBus 报警引脚（SMBA）。

19.2.1 I2C 框图

I2C 接口的框图如图 19-1 所示。

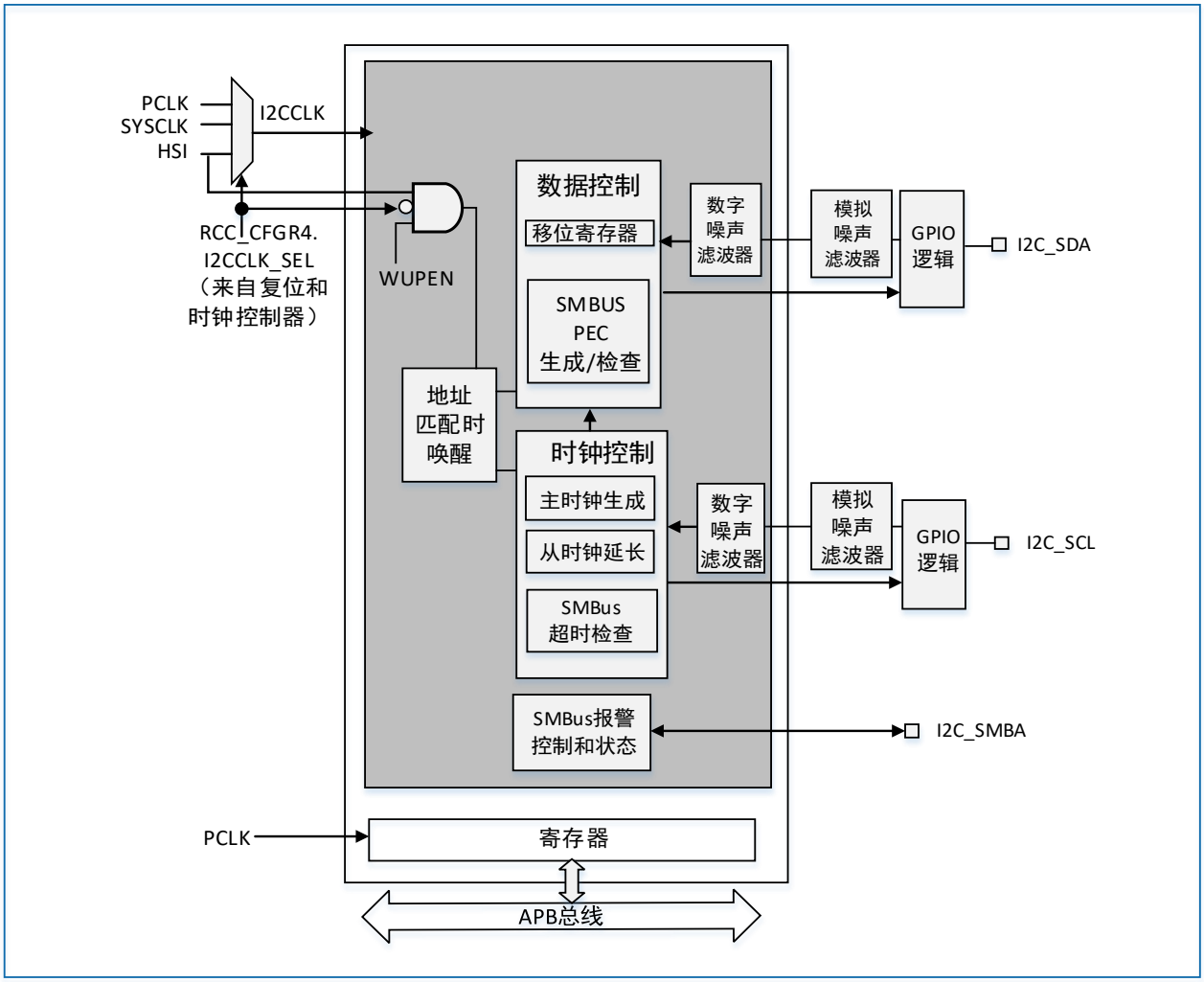

图 19-1 I2C 框图

I2C 的时钟由独立时钟源提供，这使得 I2C 能够独立于 PCLK 频率工作。

该时钟源可以是以下任一时钟源：

- PCLK：APB 时钟（默认值）
- HSI：内部 RC 振荡器
- SYSCLK：系统时钟

更多详细信息，请参见“6 复位和时钟控制（RCC）”。

19.2.2 I2C 时钟要求

I2C 内核的时钟由 I2CCLK 提供。

I2CCLK 周期 t_{I2CCLK} 必须遵循以下条件：

$$t_{I2CCLK} < (t_{LOW} - t_{filters}) / 4 \text{ 且 } t_{I2CCLK} < t_{HIGH}$$

其中：

- t_{LOW}：SCL 低电平时间
- t_{HIGH}：SCL 高电平时间

- $t_{filters}$：滤波器使能时，该值为模拟滤波器和数字滤波器引入的延时总和。模拟滤波器延时最大值为 260 ns。数字滤波器延时为 DNF x t_{I2CCLK}。

PCLK 时钟周期 t_{PCLK} 必须遵循以下条件：

$$t_{PCLK}<4/3t_{SCL}$$

其中，t_{SCL} 表示 SCL 周期。

说明：当 I2C 内核的时钟由 PCLK 提供时，PCLK 必须遵循 t_{I2CCLK} 的条件。

19.2.3 模式选择

该接口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

默认工作模式是从模式。

通信流程

在主模式下，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。在主模式下，起始条件 START 和地址由软件触发，停止位可由软件生成，也可由硬件自动生成。

在从模式下，该接口能够识别其自身地址（7 或 10 位）以及广播呼叫地址。广播呼叫地址检测可由软件使能或禁止。SMBus 总线中的主机地址、器件默认地址、报警地址也可由软件使能是否进行响应。

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）。地址始终由主设备发出。

在传输一个字节所需的 8 个时钟周期后是第 9 个时钟脉冲。在第 9 个时钟期间，接收器必须向发送器发送一个应答位（ACK），如下图所示。

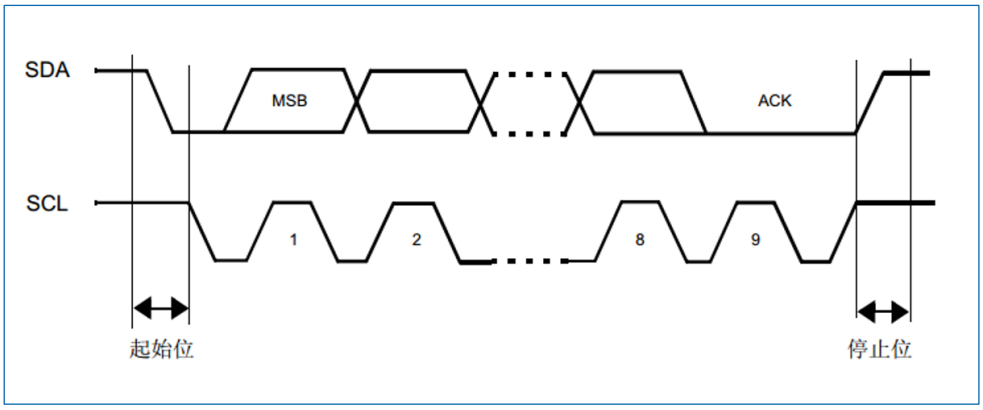

图 19-2 I2C 总线协议

ACK 信号可由软件使能或禁止。I2C 接口地址可通过软件进行选择。

19.2.4 I2C 初始化

使能和禁止外设

I2C 外设时钟必须在时钟控制器中进行配置和使能（请参见“6 复位和时钟控制（RCC）”）。通过将 I2C_CR1 寄存器中的 PE 位置 1 可使能 I2C。

当禁止 I2C（PE=0）时，I2C 将执行软件复位。更多详细信息，请参见“19.2.5 软件复位”。

噪声滤波器

SDA 和 SCL 输入上集成了模拟和数字噪声滤波器。如果用户要使用滤波器，必须在使能 I2C 模块之前完成滤波器的配置。

模拟滤波器符合 I2C 规范，此规范要求在快速模式下对脉宽在 50 ns 以下的脉冲都要抑制。模块默认使能模拟滤波器，用户可通过将 ANFOFF 位置 1 来禁止该模拟滤波器。

数字滤波器通过配置 I2C_CR1 寄存器中的 DNF[3:0]位可实现对尖峰脉宽 1 到 15 个 I2CCLK 的噪声抑制。使能数字滤波器时，SCL 或 SDA 线的电平只有在电平稳定时间超过 DNF x I2CCLK 个周期后才会发生内部变化。

表 19-1 模拟滤波器与数字滤波器对比

	模拟滤波器	数字滤波器
抑制尖峰的脉冲宽度	≥50ns	可编程长度为 1 到 15 个 I2C 外设时钟
优点	停机模式下仍可用	• 可编程长度：额外的滤波能力与标准要求 • 稳定的长度
缺点	随温度、电压和过程变化会发生变化	当使能数字滤波器后，无法在地址匹配时从停机模式唤醒。

说明：使能 I2C 时，不允许更改滤波器配置。

I2C 时序

必须配置时序，以保证主模式和从模式下使用正确的数据保持和建立时间。配置方法是编程 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLDEL[3:0]和 SDADEL[3:0]位。

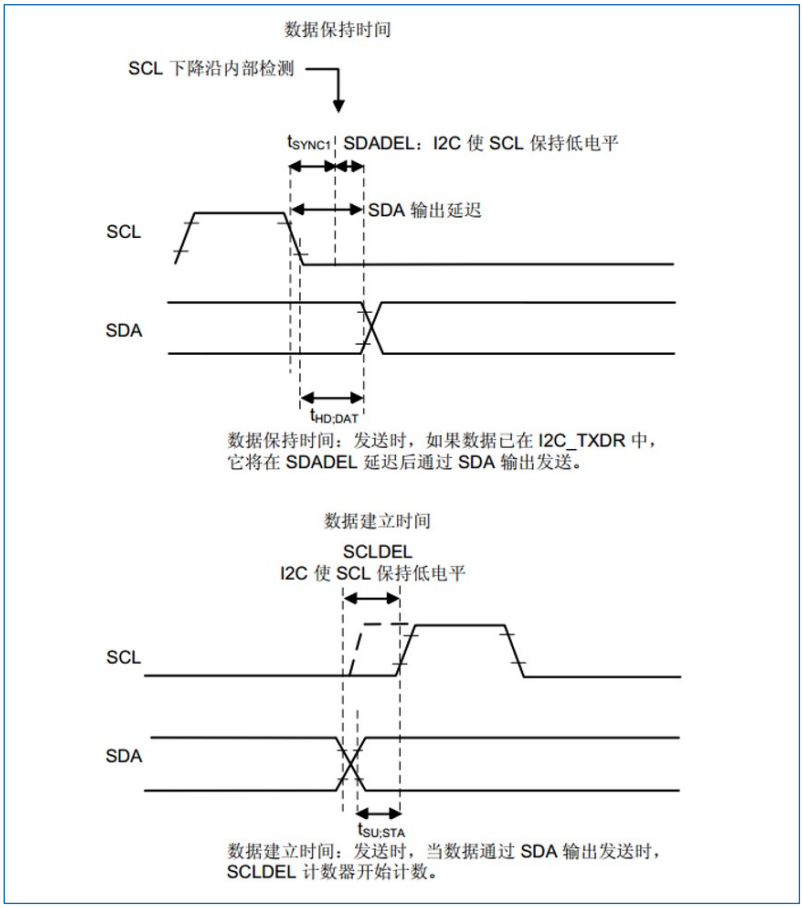

图 19-3 建立和保持时序

- 当内部检测到 SCL 下降沿时，会在发送 SDA 输出之前插入一段延时。该延时为 t_{SDADEL}= SDADEL x t_{PRESC} + t_{I2CCLK}，其中 t_{PRESC} =（PRESC+1）x t_{I2CCLK}

  t_{SDADEL} 会影响保持时间 $t_{HD;DAT}$。

- SDA 总输出延时为：

  t_{SYNC1}+{[SDADEL x (PRESC+1)+1] x t_{I2CCLK}}

  t_{SYNC1} 持续时间取决于以下参数：

  - SCL 下降斜率
  - 模拟滤波器（使能时）引入的输入延时：t_{AF}（min）<t_{AF}<t_{AF}（max）ns
  - 数字滤波器（使能时）引入的输入延时：t_{DNF}=DNF x t_{I2CCLK}
  - SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

为了桥接 SCL 下降沿的未定义区域，用户编程 SDADEL 时必须遵循以下条件：

$$\{t_{f(max)}+t_{HD;DAT(min)}-t_{AF(min)}-[(DNF+3)xt_{I2CCLK}]\}/\{(PRESC+1)xt_{I2CCLK}\} \leq SDADEL$$

$$SDADEL \leq \{t_{HD;DAT(max)}-t_{AF(max)}-[(DNF+4)xt_{I2CCLK}]\}/\{(PRESC+1)xt_{I2CCLK}\}$$

说明：只有使能模拟滤波器时，公式中才包含 t_{AF}（min）/ t_{AF}（max）。有关 t_{AF} 值的信息，请参见 HK32F0301MxxxxC 数据手册。

标准模式和快速模式下的 $t_{HD;DAT}$ 最大值分别可达 3.45 µs 和 0.9 µs，但必须小于 $t_{VD;DAT}$ 最大值（差值为跳变时间）。只有器件未延长 SCL 信号的低电平周期（t_{LOW}）时，才必须满足该最大值条件。如果时钟延长 SCL，数据必须在建立时间内保持有效，之后才能释放时钟。

SDA 上升沿通常为最坏情况，因此在这种情况下，上述公式变成如下形式：

$$SDADEL \leq \{t_{VD;DAT（max）}-t_{R（max）}-260ns-[（DNF+4）xt_{I2CCLK}]\}/\{（PRESC+1）x\ t_{I2CCLK}\}$$

说明：NOSTRETCH=0 时会违反该条件，这是因为器件会根据 SCLDEL 值来延长 SCL 低电平时间，以保证建立时间。

有关 t_f、tr、$t_{HD;DAt}$ 和 $t_{VD;DAT}$ 标准值的信息，请参见表 19-2。

- 在 t_{SDADEL} 延时后，或在因数据未写入 I2C_TXDR 寄存器而导致从器件必须延长时钟的情况下发送 SDA 输出后，SCL 线会在建立时间内保持低电平。该建立时间为 t_{SCLDEL}=（SCLDEL+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）x t_{I2CCLK}。

t_{SCLDEL} 会影响建立时间 $t_{SU;DAT}$。

为了桥接 SDA 跳变（上升沿通常为最坏情况）的未定义区域，编程 SCLDEL 时必须遵循以下条件：

$$\{[t_R（max）+ t_{SU;DAT}（min）]/[（PRESC+1）] x\ t_{I2CCLK}\}-1 <= SCLDEL$$

有关 tr 和 $t_{SU;DAT}$ 标准值的信息，请参见表 19-2。

将使用的 SDA 和 SCL 跳变时间值就是应用中的值。使用最大值而非标准值会增加 SDADEL 和 SCLDEL 计算的约束条件，但能够确保任意应用的特性。

说明：在发送和接收模式下，对于每个时钟脉冲，检测到 SCL 下降沿后，I2C 主器件或从器件会至少在[(SDADEL+SCLDEL+1) x (PRESC+1) + 1] x t_{I2CCLK} 期间内延长 SCL 低电平时间。在发送模式下，如果 SDADEL 计数器计数结束后数据还未写入 I2C_TXDR，则 I2C 会继续延长 SCL 低电平时间，直到写入下一个数据。随后，会将新数据 MSB 发送到 SDA 输出，SCLDEL 计数器将开始计数，同时会继续延长 SCL 低电平时间以确保提供充足的数据建立时间。

如果从模式下 NOSTRETCH=1，则 SCL 不会延长。因此，编程 SDADEL 时还必须确保提供充足的建立时间。

表 19-2 I2C-SMBus 规范数据建立和保持时间

符号	参数	标准模式（Sm）		快速模式（Fm）		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
$t_{HD;DAT}$	数据保持时间	0	-	0	-	0.3	-	μs
$t_{VD;DAT}$	数据有效时间	-	3.45	-	0.9	-	-	
$t_{SU;DAT}$	数据建立时间	250	-	100	-	250		ns
tr	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	1000	
tf	SDA 和 SCL 信号的下降时间	-	300	-	300	-	300	

此外，在主模式下，必须通过编程 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLH[7:0]和 SCLL[7:0]位来配置 SCL 时钟的高电平和低电平。

- 当内部检测到 SCL 下降沿时，会在释放 SCL 输出之前插入一段延时。该延时为 t_{SCLL}=（SCLL+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）x t_{I2CCLK}。t_{SCLL} 会影响 SCL 低电平时间 t_{LOW}。
- 当内部检测到 SCL 上升沿时，会在将 SCL 输出强制为低电平之前插入一段延时。该延时为 t_{SCLH}=（SCLH+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）xt_{I2CCLK}。t_{SCLH} 会影响 SCL 高电平时间 t_{HIGH}。

更多详细信息，请参见“19.2.8 主模式”中的“I2C 主模式初始化”。

说明：使能 I2C 后，不允许更改时序配置。

此外，还必须在使能从设备前，对 NOSTRETCH 进行配置。更多详细信息，请参见“19.2.7 从模式”中的“I2C 从模式初始化”。

说明：使能 I2C 后，不允许更改 NOSTRETCH 配置。

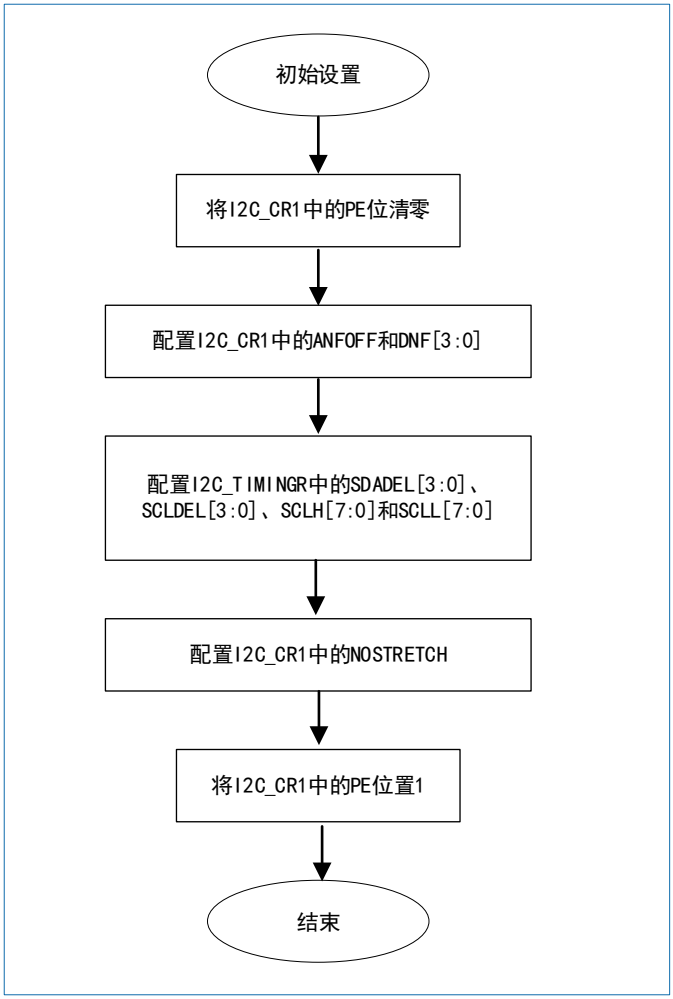

图 19-4 I2C 初始化流程图

19.2.5 软件复位

可通过将 I2C_CR1 寄存器中的 PE 位清零来执行软件复位。在这种情况下，I2C 线 SCL 和 SDA 被释放。内部状态机复位，通信控制位和状态位重置为复位值。配置寄存器不受影响。

下面列出了受影响的寄存器位：

- I2C_CR2 寄存器：START、STOP 和 NACK
- I2C_ISR 寄存器：BUSY、TXE、TXIS、RXNE、ADDR、NACKF、TCR、TC、STOPF、BERR、ARLO 和 OVR

支持 SMBus 功能时还会影响到以下寄存器位：

- I2C_CR2 寄存器：PECBYTE
- I2C_ISR 寄存器：PECERR、TIMEOUT 和 ALERT

由于寄存器存在跨时钟域设计，必须使 PE 保持低电平持续至少 3 个 APB 时钟周期，才能成功执行软件复位。写入以下软件序列可确保这一点：

1. 写入 PE=0；
2. 检查 PE 位，直到 PE=0；
3. 写入 PE=1。

19.2.6 数据传输

数据传输由发送和接收数据寄存器以及移位寄存器来管理。

接收

在 SDA 上接收的数据被填充到内部移位寄存器。在第 8 个 SCL 脉冲后，数据已全部接收到移位寄存器中。如果 I2C_RXDR 寄存器为空（RXNE=0），则移位寄存器的内容会复制到 I2C_RXDR 寄存器中。

如果 RXNE=1，意味着尚未读取上一次接收到的数据字节。

作为 I2C Master 时，硬件将延长第 8 和第 9 个 SCL 脉冲之间的低电平，直到软件读取 I2C_RXDR 为止。

作为 I2C Slave 时，如果 NOSTRETCH=0，将延长 SCL 线的低电平时间，直到读取了 I2C_RXDR 为止。如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

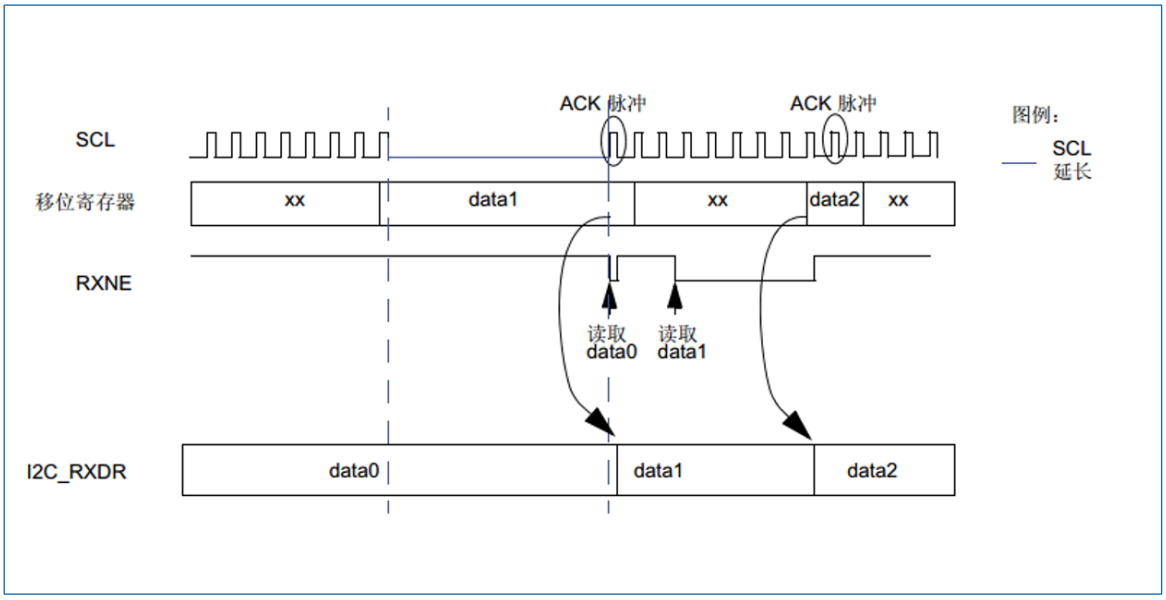

图 19-5 数据接收

发送

如果 I2C_TXDR 寄存器不为空（TXE=0），则其内容会在第 9 个 SCL 脉冲（ACK 脉冲）后复制到移位寄存器中。然后移位寄存器的内容会移出到 SDA 线上。

如果 TXE=1，意味着 I2C_TXDR 内尚未写入任何数据。

作为 I2C Master，硬件将延长第 9 个 SCL 脉冲后的低电平，直到写入了 I2C_TXDR 为止。

作为 I2C Slave，如果 NOSTRETCH=0，将延长时钟，直到软件写 I2C_TXDR；如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

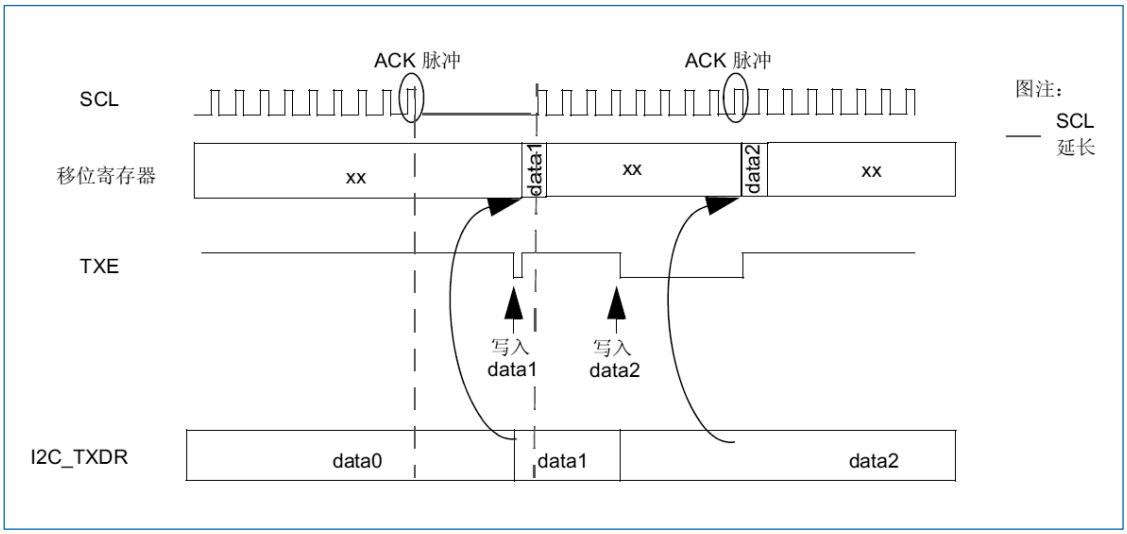

图 19-6 数据发送

硬件传输管理

I2C 在硬件中内置了字节计数器，以便在下列各种模式下管理字节传输和结束通信：

- 主模式下生成 NACK、STOP 和 RESTART
- 从接收器模式下控制回复 ACK/NACK
- SMBus 模式下生成 PEC/对接收的数据进行 PEC 校验

字节计数器在主模式下默认开启。在从模式下，字节计数器默认为关闭状态，但可以通过软件来使能，方法是将 I2C_CR1 寄存器中的 SBC（从字节控制）位置 1。

待传输的字节数在 I2C_CR2 寄存器的 NBYTES[7:0]位域中进行编程。如果待传输的字节数（NBYTES）大于 255，或者接收方希望控制是否对接收到的数据字节进行应答，则必须选择重载模式，方法是将 I2C_CR2 寄存器的 RELOAD 位置 1。在该模式下，完成 NBYTES 中所编程字节数的数据传输之后，TCR 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TCR 标志置 1，SCL 便会延长。当往 NBYTES 写入一个非零值时，TCR 由软件清零。

在向 NBYTES 中设置最后一次传输的字节数前，必须把 RELOAD 位清零。

在主模式下，RELOAD=0 时，字节计数器功能如下：

- 自动结束模式（I2C_CR2 寄存器中的 AUTOEND="1"）。在该模式下，一旦完成 NBYTES[7:0]位域中所编程字节数的数据传输，主器件便会自动发送停止位。
- 软件结束模式（I2C_CR2 寄存器中的 AUTOEND="0"）在该模式下，一旦完成 NBYTES[7:0]位域中所编程字节数的数据传输，TC 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TC 标志置 1，SCL 信号便会延长，需要软件介入操作。当软件把 I2C_CR2 寄存器中的起始位或停止位置 1 时，TC 标志将被清零。当主器件要发送重复起始位时，必须使用该模式。

说明：当 RELOAD 位置 1 时，AUTOEND 位将不起作用。

表 19-3 I2C 配置表

功能	SBC 位	RELOAD 位	AUTOEND 位
主 Tx/Rx+NBYTES+STOP	x	0	1
主 Tx/Rx+NBYTES+RESTART	x	0	0
从 Tx/Rx 接收的所有字节都要回复应答	0	x	x
具有 ACK 控制的从 Rx	1	1	x

19.2.7 从模式

I2C 从模式初始化

在从模式下工作，用户必须至少使能一个从地址。可使用 I2C_OAR1 和 I2C_OAR2 这两个寄存器来编程自身从地址 OA1 和 OA2。

- OA1 既可配置为 7 位寻址模式（默认），也可通过将 I2C_OAR1 寄存器 OA1MODE 位置 1 配置为 10 位寻址模式。

通过将 I2C_OAR1 寄存器中的 OA1EN 位置 1 来使能 OA1。

- 如果需要额外的从地址，可配置第 2 个从地址 OA2。将 I2C_OAR2 寄存器的 OA2MSK[2:0]位置 1 最多可屏蔽 7 个 OA2LSB。因此，当 OA2MSK 配置为 1 到 6 时，将分别只有 OA2[7:2]、OA2[7:3]、OA2[7:4]、OA2[7:5]、OA2[7:6]或 OA2[7]与接收到的地址作比较。只要 OA2MSK 不等于 0，OA2 的地址比较器便会排除 I2C 保留地址（0000XXX 和 1111XXX），这些地址将不会得到应答。如果 OA2MSK=7，接收到的所有 7 位地址（保留地址除外）均得到应答。OA2 始终为 7 位地址。

如果这些保留地址在 I2C_OAR1 或 I2C_OAR2 寄存器中进行了编程并且 OA2MSK=0，则它们可以在通过特定使能位使能后得到应答。

通过将 I2C_OAR2 寄存器中的 OA2EN 位置 1 来使能 OA2。

- 通过将 I2C_CR1 寄存器中的 GCEN 位置 1 来使能广播呼叫地址。

当通过 I2C 的其中一个使能地址来寻址到该 I2C 设备时，ADDR 中断状态标志将置 1，并且 ADDRIE 位置 1 时将生成中断。

默认情况下，从器件使用其时钟延长功能（即必要时延长 SCL 信号的低电平时间）来为软件操作的执行提供时机。如果主器件不支持时钟延长，则必须对 I2C 进行如下配置：将 I2C_CR1 寄存器中 NOSTRETCH 位置 1。

接收到 ADDR 中断后，如果使能多个地址，则用户必须读取 I2C_ISR 寄存器中的 ADDCODE[6:0]位，以确定是哪个地址匹配。还必须检查 DIR 标志，以获悉传输方向。

带时钟延长的从模式（NOSTRETCH=0）

在默认模式下，I2C 从器件会在以下情况下延长 SCL 时钟：

- ADDR 标志置 1 时：接收到的地址与其中一个使能的从地址匹配。通过软件将 ADDRCF 位置 1 以清零 ADRR 标志时，将释放该时钟延展。
- 发送时，前一次数据传输已完成但 I2C_TXDR 寄存器中未写入任何新数据，或者 ADDR 标志清零（TXE=1）时未写入第一个数据字节。往 I2C_TXDR 寄存器中写入数据时，将释放该时钟延展。
- 接收时，尚未读取 I2C_RXDR 寄存器但新的数据接收已完成。读取 I2C_RXDR 时，将释放该时钟延展。
- 当从器件字节控制模式和重载模式（SBC=1 且 RELOAD=1）下 TCR=1 时，这意味着最后一个数据字节已完成传输。通过向 NBYTES[7:0]字段写入一个非零值以将 TCR 清零时，将释放该时钟延展。
- 在 SCL 下降沿检测之后，I2C 会延长 SCL 的低电平时间（不超过[（SDADEL+SCLDEL+1）x（PRESC+1）+1] x t_{I2CCLK}）。

不带时钟延长的从模式（NOSTRETCH=1）

当 I2C_CR1 寄存器中的 NOSTRETCH=1 时，I2C 从器件不会延长 SCL 信号。

- ADDR 标志置 1 时，不会延长 SCL 时钟。
- 发送时，必须在与发送数据对应的第一个 SCL 脉冲出现之前，向 I2C_TXDR 寄存器写入数据。

否则，会发生下溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。当第一次数据发送开始而 STOPF 位仍置 1（尚未清零）时，OVR 标志也将置 1。因此，如果写入下一次传输要发送的第一个数据后才清零上一次传输的 STOPF 标志，则应提供 OVR 状态，甚至对于待发送的第一个数据也是如此。

- 接收时，必须在下一个数据字节的第 9 个 SCL 脉冲（ACK 脉冲）出现之前，从 I2C_RXDR 寄存器读取数据。否则，会发生上溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

从器件字节控制模式

要在从接收模式下实现字节 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从器件字节控制模式。这样符合 SMBus 标准。

要在从接收模式下实现字节 ACK 控制，必须选择重载模式（RELOAD=1）。要控制每个字节，必须在 ADDR 中断子程序中将 NBYTES 初始化为 0x1，并在每接收一个字节后将 NBYTE 重载为 0x1。接收到字节后，TCR 位将置 1，从而延长 SCL 信号的第 8 个和第 9 个脉冲之间的低电平时间。用户可以从 I2C_RXDR 寄存器中读取数据，然后通过配置 I2C_CR2 寄存器中的 NACK 位来决定是否应答。通过将 NBYTES 编程为非零值来释放 SCL 延长：发送应答或不应答信号，然后可继续接收下一个字节。

NBYTES 可加载大于 0x1 的值，在这种情况下，接收流在 NBYTES 个数据接收期间是连续的。

说明：

SBC 位只能在 I2C 被禁止时、从器件不被寻址时或 ADDR=1 时配置。

ADDR=1 或 TCR=1 时，可以更改 RELOAD 位的值。

从器件字节控制模式与 NOSTRETCH 模式不兼容。不允许在 NOSTRETCH=1 时将 SBC 位置 1。

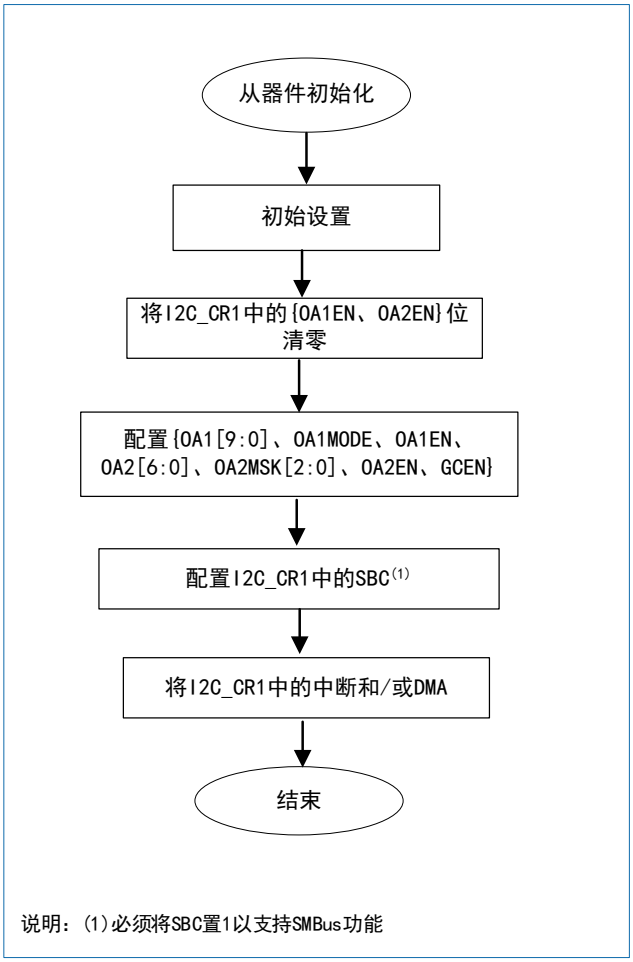

图 19-7 从器件初始化流程图

从发送器

当 I2C_TXDR 寄存器为空时，将生成发送中断状态（TXIS）。如果 I2C_CR1 寄存器中的 TXIE 位置 1，将生成中断。

I2C_TXDR 寄存器中写入待发送的下一个数据字节时，TXIS 位将被清零。

接收到 NACK 时，I2C_ISR 寄存器中的 NACKF 位将置 1，如果 I2C_CR1 寄存器中的 NACKIE 位置 1，还将生成中断。从器件自动释放 SCL 和 SDA 线，以使主器件执行停止或重复起始位的发送。收到 NACK 时，TXIS 位不会置 1。

当接收到停止位且 I2C_CR1 寄存器中的 STOPIE 位置 1 时，I2C_ISR 寄存器中的 STOPF 标志将置 1 并且会生成中断。在大多数应用中，SBC 位通常编程为“0”。在这种情况下，如果接收到从地址（ADDR=1）时 TXE=0，用户可以选择发送 I2C_TXDR 寄存器的内容作为第一个数据字节，也可以选择通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器以编程新的数据字节。

在从器件字节控制模式（SBC=1）下，必须在地址匹配中断子程序（ADDR=1）中向 NBYTE 写入待发送数据的个数。在这种情况下，传输期间 TXIS 事件的数量对应于 NBYTES 中编程的值。

说明：如果 NOSTRETCH=1，当 ADDR 标志置 1 时不会延长 SCL 时钟，因此用户无法在 ADDR 子程序中刷新 I2C_TXDR 寄存器的内容，从而编程第一个数据字节。必须在 I2C_TXDR 寄存器中预编程待发送的第一个数据字节：

- 该数据可以是前一个传输消息的最后一个 TXIS 事件中写入的数据。
- 如果该数据字节不是待发送的数据字节，可通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器，从而编程新的数据字节。必须仅在执行完这些操作后再清零 STOPF 位，以确保在地址应答之后，第一次数据传输开始之前执行这些操作。

如果第一次数据传输开始时 STOPF 仍置 1，则将生成下溢错误（OVR 标志置 1）。

如果需要 TXIS 事件（发送中断），用户必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

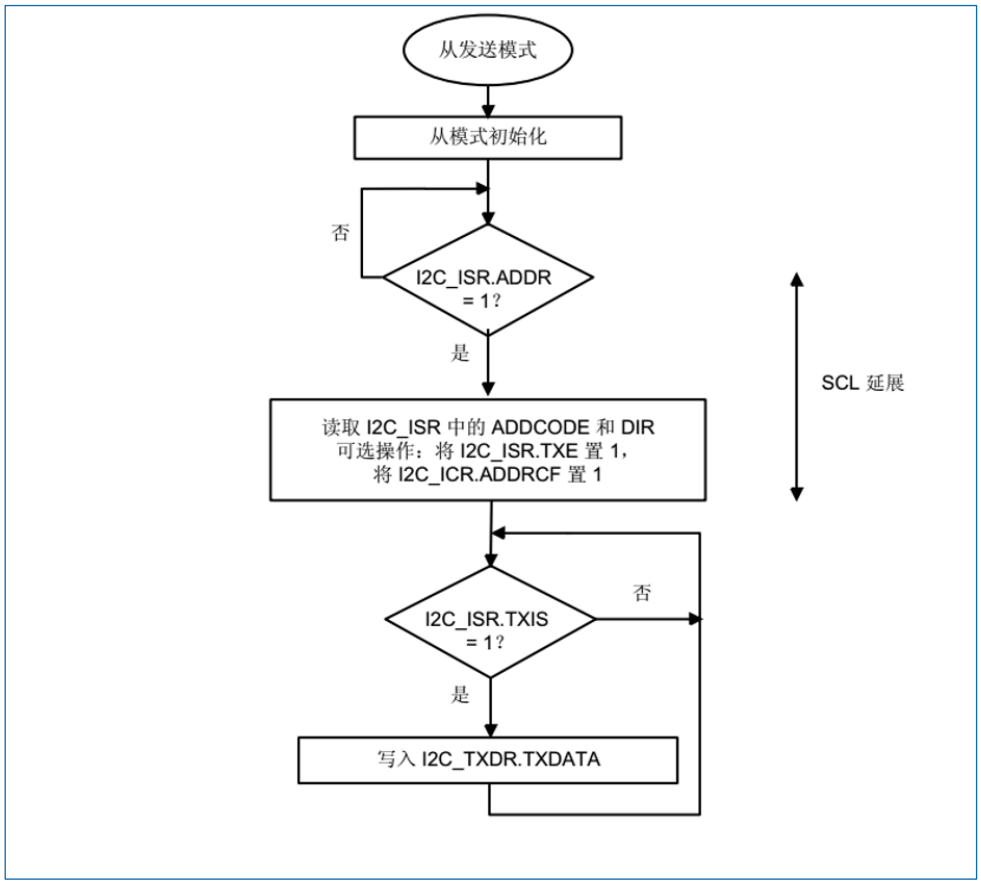

图 19-8 I2C 从发送器的传输序列流程图（NOSTRETCH=0）

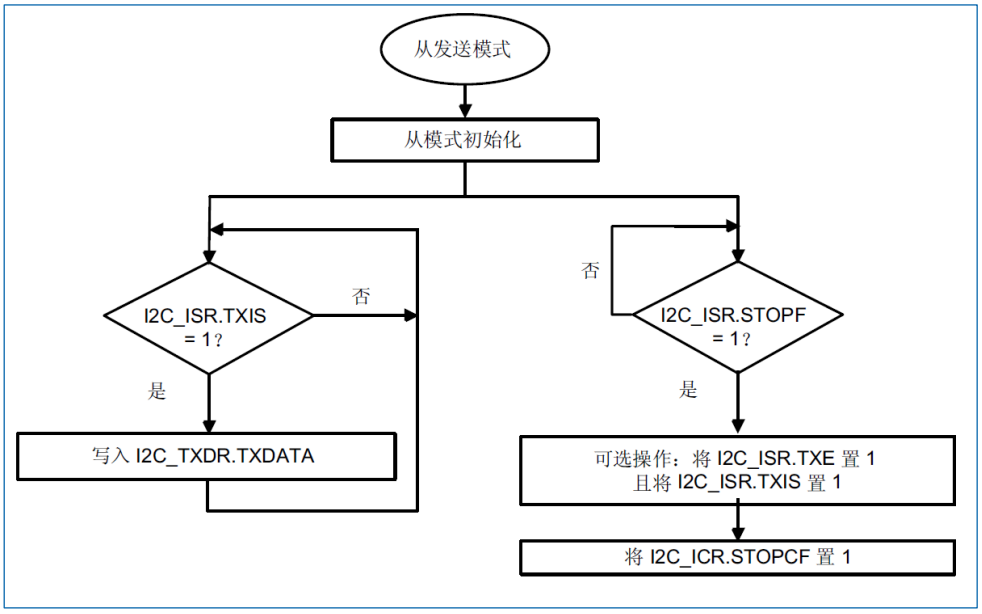

图 19-9 I2C 从发送器的传输序列流程图（NOSTRETCH=1）

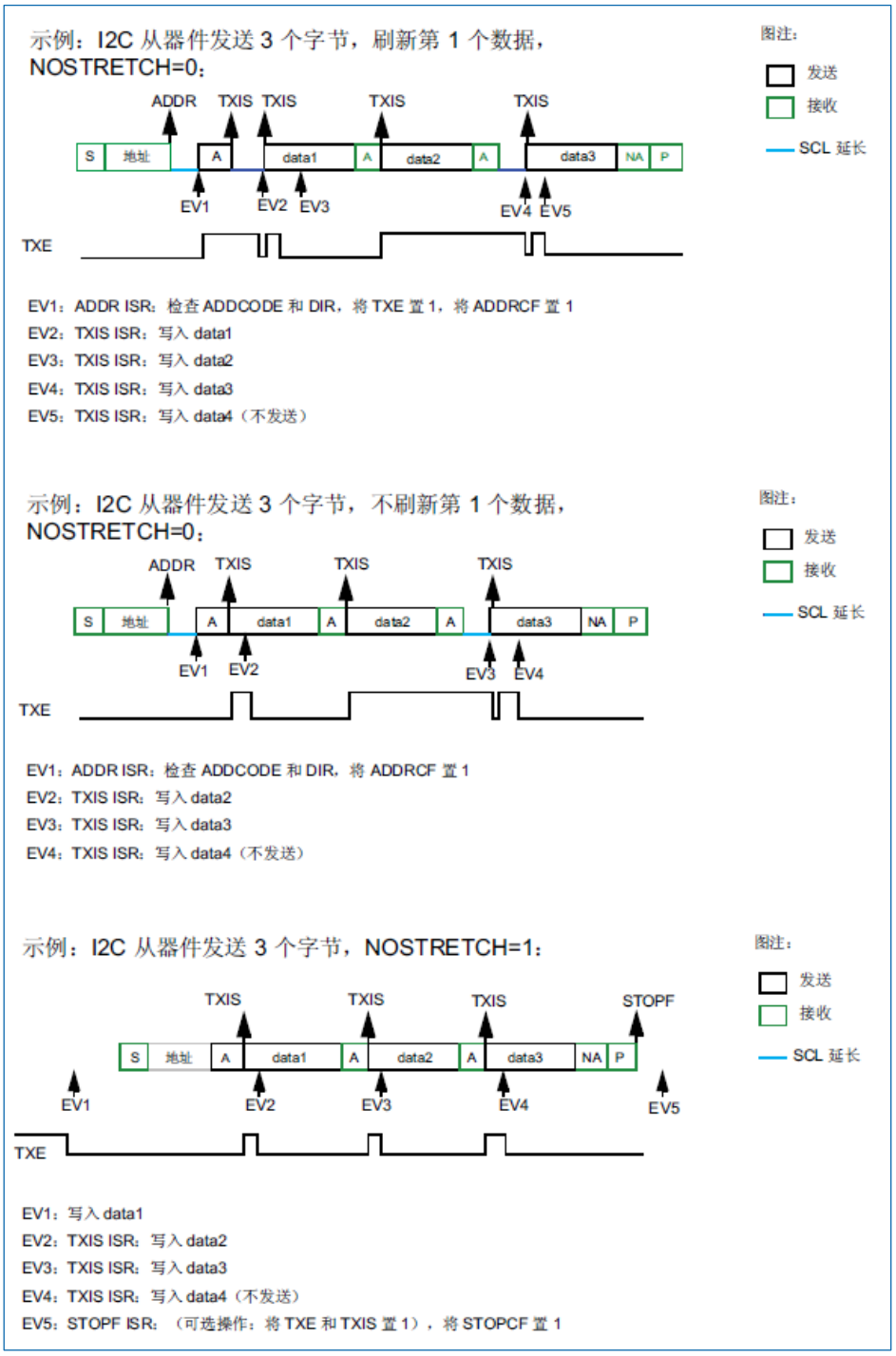

图 19-10 从发送器的传输总线图

从接收器

当 I2C_RXDR 满时，I2C_ISR 中的 RXNE 将置 1，如果 I2C_CR1 中的 RXIE 置 1，还将生成中断。读取 I2C_RXDR 时，将清零 RXNE。

接收到停止条件且 I2C_CR1 寄存器中的 STOPIE 置 1 时，I2C_ISR 中的 STOPF 将置 1 并且会生成中断。

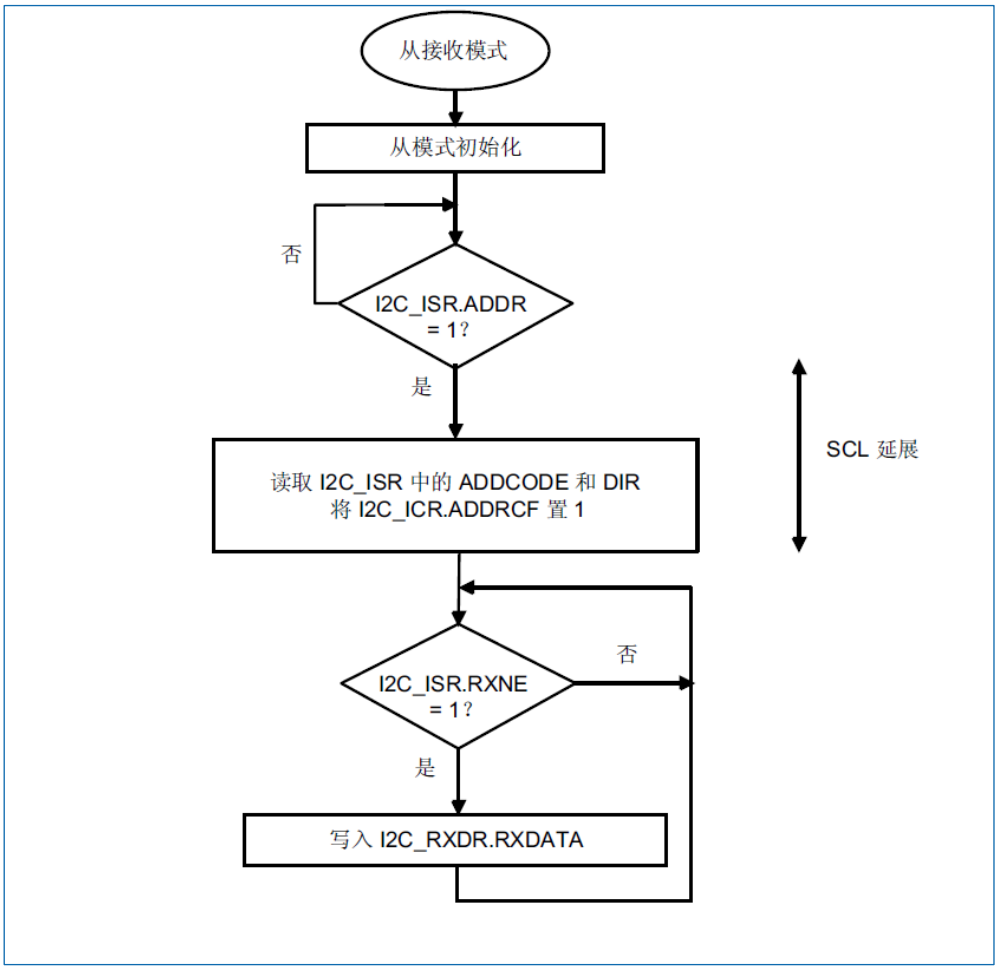

图 19-11 从接收器的传输序列流程图（NOSTRETCH=0）

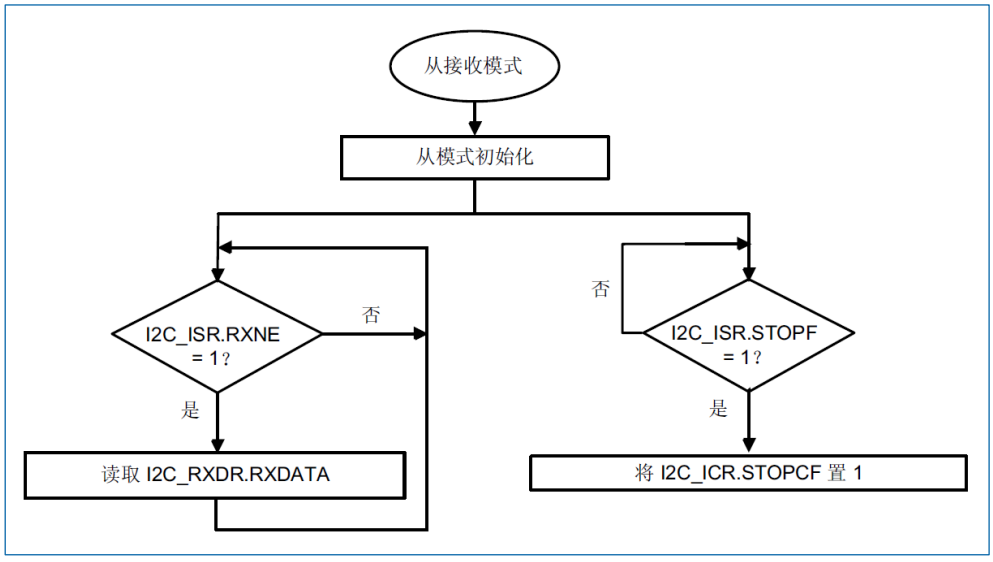

图 19-12 从接收器的传输序列流程图（NOSTRETCH=1）

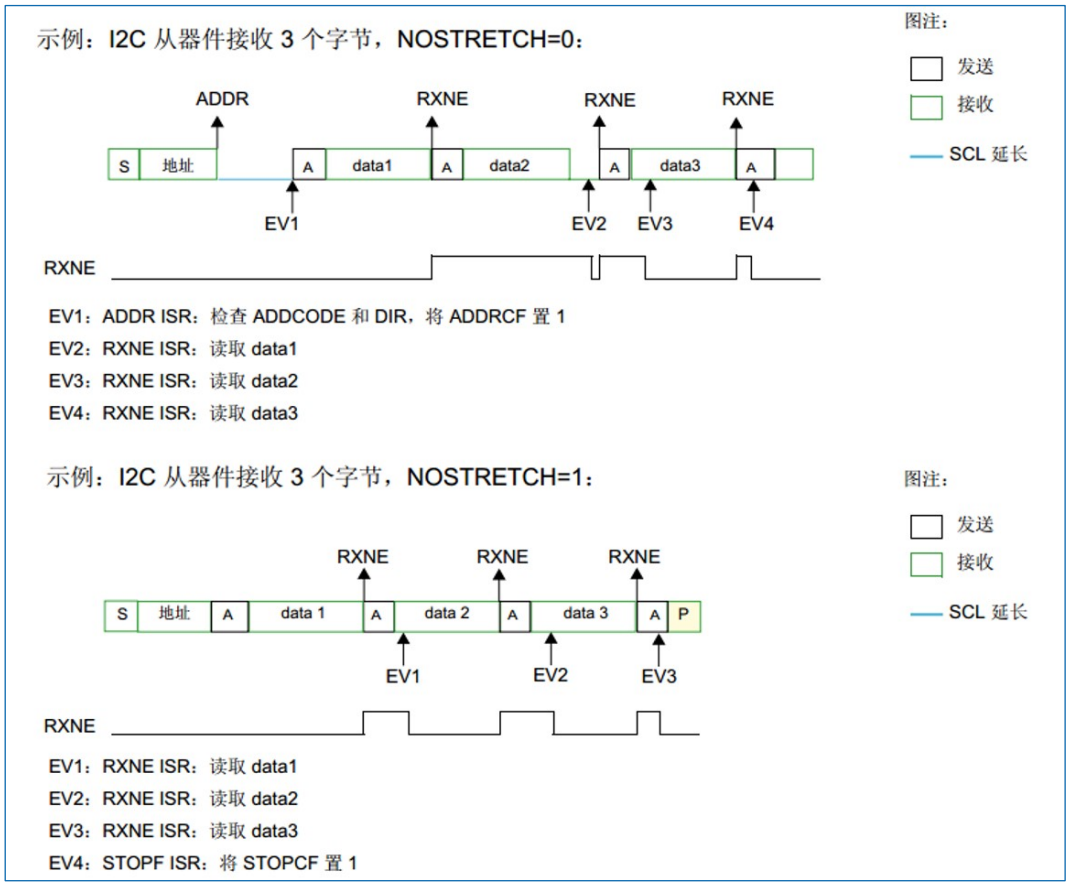

图 19-13 I2C 从接收器的传输总线图

19.2.8 主模式

I2C 主模式初始化

使能外设前，必须通过设置 I2C_TIMINGR 寄存器中的 SCLH 和 SCLL 位来配置 I2C 主时钟。

为了支持多主环境和从时钟延长，I2C 实现了时钟同步机制。

为了实现时钟同步，需执行以下操作：

- 使用 SCLL 计数器，从 SCL 低电平内部检测开始对时钟的低电平进行计数。
- 使用 SCLH 计数器，从 SCL 高电平内部检测开始对时钟的高电平进行计数。

I2C 经过 T_{SYNC1} 延时后，检测其自身的 SCL 低电平，该延时取决于 SCL 下降沿、SCL 输入噪声滤波器（模拟+数字）以及 SCL 与 I2CCLK 时钟的同步。一旦 SCLL 计数器达到 I2C_TIMINGR 寄存器的 SCLL[7:0]位中编程的值，I2C 便会将 SCL 释放为高电平。

I2C 经过 T_{SYNC2} 延时后检测其自身的 SCL 高电平，该延时取决于 SCL 上升沿、SCL 输入噪声滤波器（模拟+数字）以及 SCL 与 I2CCLK 时钟的同步。一旦 SCLH 计数器达到 I2C_TIMINGR 寄存器的 SCLH[7:0]位中编程的值，I2C 便会使 SCL 变为低电平。

因此，主时钟周期为：

$$t_{SCL}=t_{SYNC1}+t_{SYNC2}+\{[（SCLH+1）+（SCLL+1）]x（PRESC+1）x\ t_{I2CCLK}\}$$

t_{SYNC1} 的持续时间取决于以下参数：

- SCL 下降斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}

- SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

t_{SYNC2} 的持续时间取决于以下参数：

- SCL 上升斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}
- SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

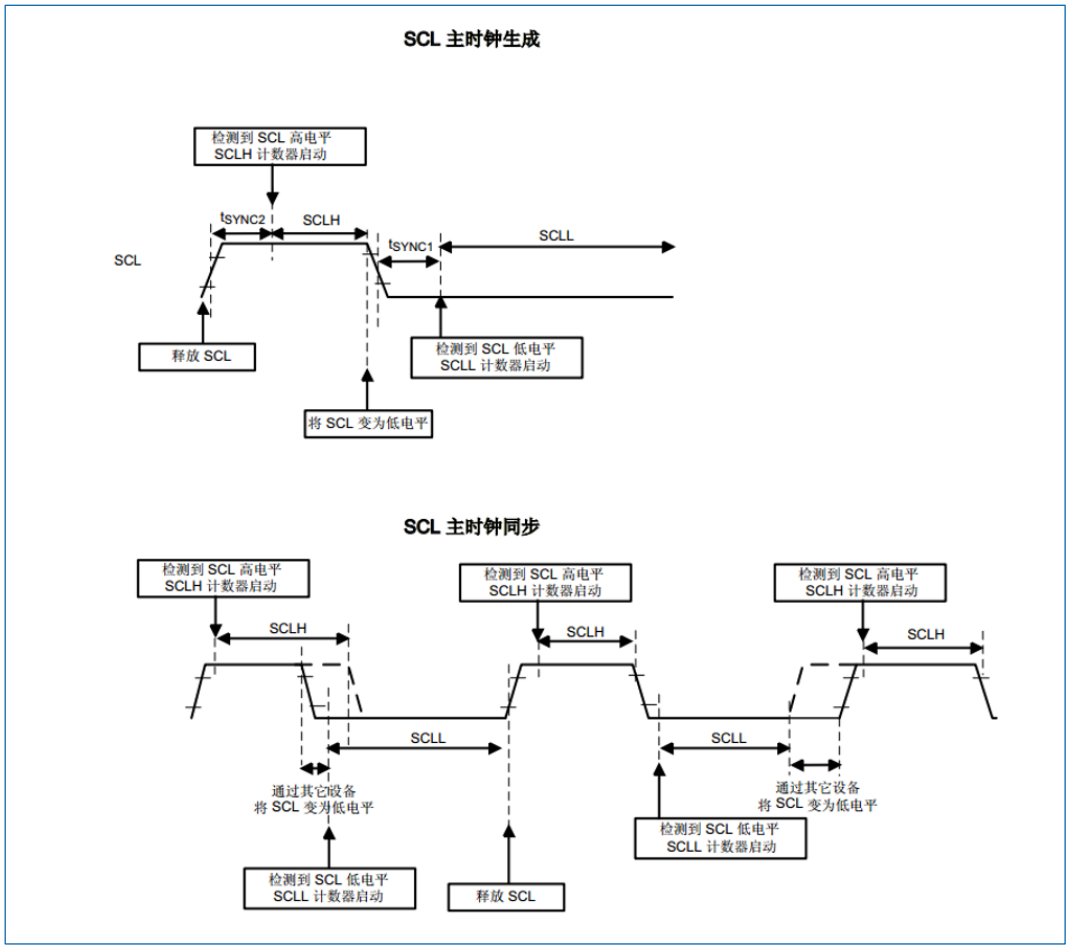

图 19-14 主时钟生成

说明：为了符合 I2C 或 SMBus 规范，主时钟必须遵循下表中给出的时序：

表 19-4 I2C-SMBus 规范时钟时序

符号	参数	标准模式（Sm）		快速模式（Fm）		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
f_{SCL}	SCL 时钟频率		100		400		100	kHz
$t_{HD:STA}$	（重复）起始条件的保持时间	4.0	-	0.6		4.0	-	μs
$t_{SU:STA}$	重复起始条件的建立时间	4.7	-	0.6		4.7	-	μs
$t_{SU:STO}$	停止条件的建立时间	4.0	-	0.6	-	4.0	-	μs
t_{BUF}	停止条件和起始条件	4.7	-	1.3	-	4.7	-	μs

符号	参数	标准模式（Sm）		快速模式（Fm）		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	
	之间的总线空闲时间							
t_{LOW}	SCL 时钟的低电平周期	4.7	-	1.3	-	4.7	-	μs
t_{HIGH}	SCL 时钟的高电平周期	4.0	-	0.6	-	4.0	50	μs
t_r	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	1000	μs
t_f	SDA 和 SCL 信号的下降时间	-	300	-	300	-	300	μs

说明：

- SCLL 还用于生成 t_{BUF} 和 $t_{SU;STA}$ 时序。
- SCLH 还用于生成 $t_{HD;STA}$ 和 $t_{SU;STO}$ 时序。

主模式通信初始化（地址阶段）

要发起通信，用户必须在 I2C_CR2 寄存器中为寻址的从器件编程以下参数：

- 寻址模式（7 位或 10 位）：ADD10
- 待发送的从地址：SADD[9:0]
- 传输方向：RD_WRN
- 读取 10 位地址时：HEAD10R 位。必须对 HEAD10R 进行相应配置，以指示传输方向变化时必须发送完整的地址序列，还是只发送地址头。
- 待传输的字节数：NBYTES[7:0]。如果字节数等于或大于 255，则初始化时必须将 NBYTES[7:0]填充为 0xFF。然后，用户必须将 I2C_CR2 寄存器中的 START 位置 1。START 位置 1 时，不允许更改上述所有位。

之后，当主器件检测到总线空闲（BUSY=0）时，它会在经过 T_{BUF} 的延时后自动发送起始位，随后发出从器件地址。

仲裁丢失时，主器件将自动切换回从模式，如果作为从器件被寻址，还可对其自身地址进行应答。

说明：无论接收到的应答值为何，只要已在总线上发送从地址，START 位便会由硬件复位。如果仲裁丢失，START 位也会由硬件复位。如果当 START 位置 1 时，I2C 作为从器件（ADDR=1）被寻址，则 I2C 将切换为从模式，START 位将在 ADDRCF 位置 1 时清零。

说明：该步骤同样适用于重复起始位。在这种情况下，BUSY=1。

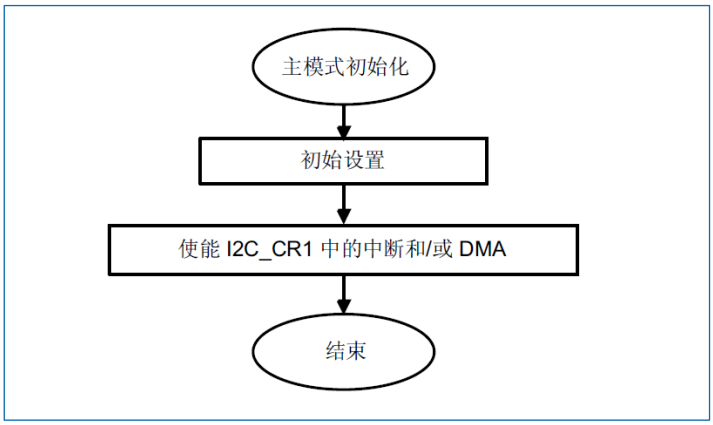

图 19-15 主模式初始化流程图

主接收器寻址 10 位地址从器件的初始化过程

- 如果从地址采用 10 位格式，用户可选择将 I2C_CR2 寄存器中的 HEAD10R 位清零来发送完整的读序列。在这种情况下，主器件会在 START 位置 1 后自动发送以下完整序列：（重复）起始位+带写方向的从器件 10 位地址头字节+从器件地址第 2 个字节+重复起始位+带读方向的从器件 10 位地址头字节

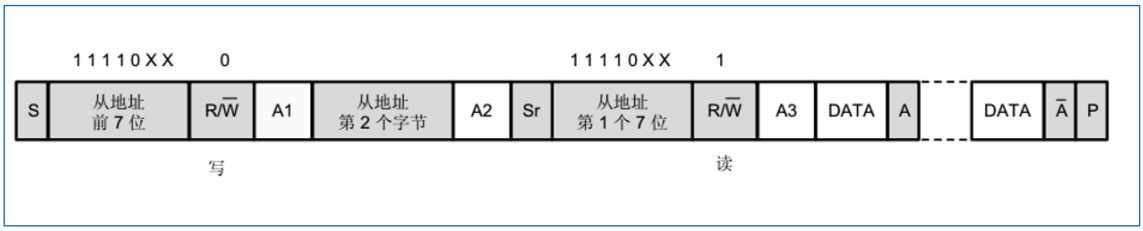

图 19-16 10 位地址读访问（HEAD10R=0）

- 如果主器件对 10 位地址从器件进行寻址、向该从器件发送数据、然后再从该从器件读取数据，则必须首先完成主器件发送过程。然后，重复起始位置 1，10 位从地址配置为 HEAD10R=1。在这种情况下，主器件发送以下序列：重复起始位+从地址 10 位头读取。

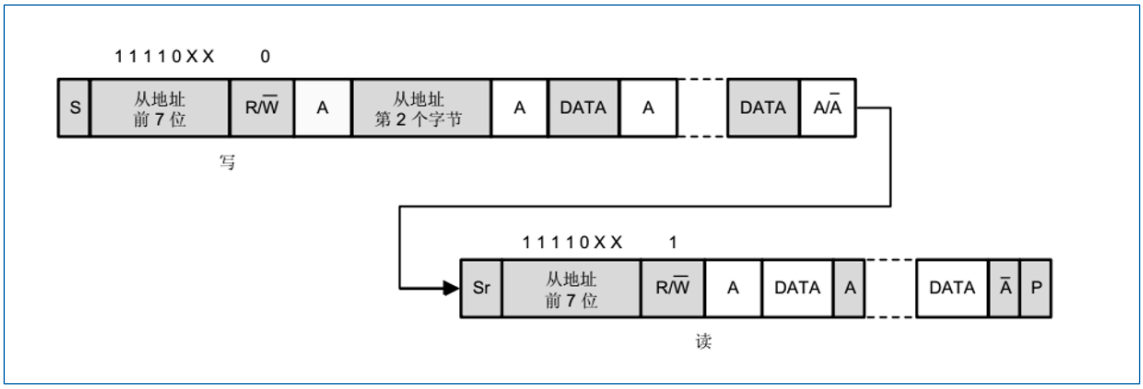

图 19-17 10 位地址读访问（HEAD10R=1）

主发送器

写传输时，在发送完每个字节（即第 9 个 SCL 脉冲（接收到 ACK 时））后，TXIS 标志将置 1。

如果 I2C_CR1 寄存器中的 TXIE 位置 1，TXIS 事件将生成中断。当 I2C_TXDR 寄存器中写入待发送的下一个数据字节时，该标志将被清零。

传输期间的 TXIS 事件的数量对应于 NBYTES[7:0]中编程的值。如果待发送的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES 数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延展，直到 NBYTES[7:0]被写入非零值。

收到 NACK 时，TXIS 标志不会置 1。

- 当 RELOAD=0 且 NBYTES 数据传输完成时：
  - 在自动结束模式（AUTOEND=1）下，将自动发送停止位。
  - 在软件结束模式（AUTOEND=0）下，TC 标志将置 1 且 SCL 线的低电平将被延展，以便执行以下软件操作：

  可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位。

  可通过将 I2C_CR2 寄存器中的 Stop 位置 1 来请求停止位。将 Stop 位置 1 会将 TC 标志清零，并在总线上发送停止位。
- 如果接收到 NACK：TXIS 标志不会置 1，并且接收到 NACK 后会自动发送停止位。I2C_ISR 寄存器中的 NACKF 标志置 1，如果 NACKIE 位置 1，还将生成中断。

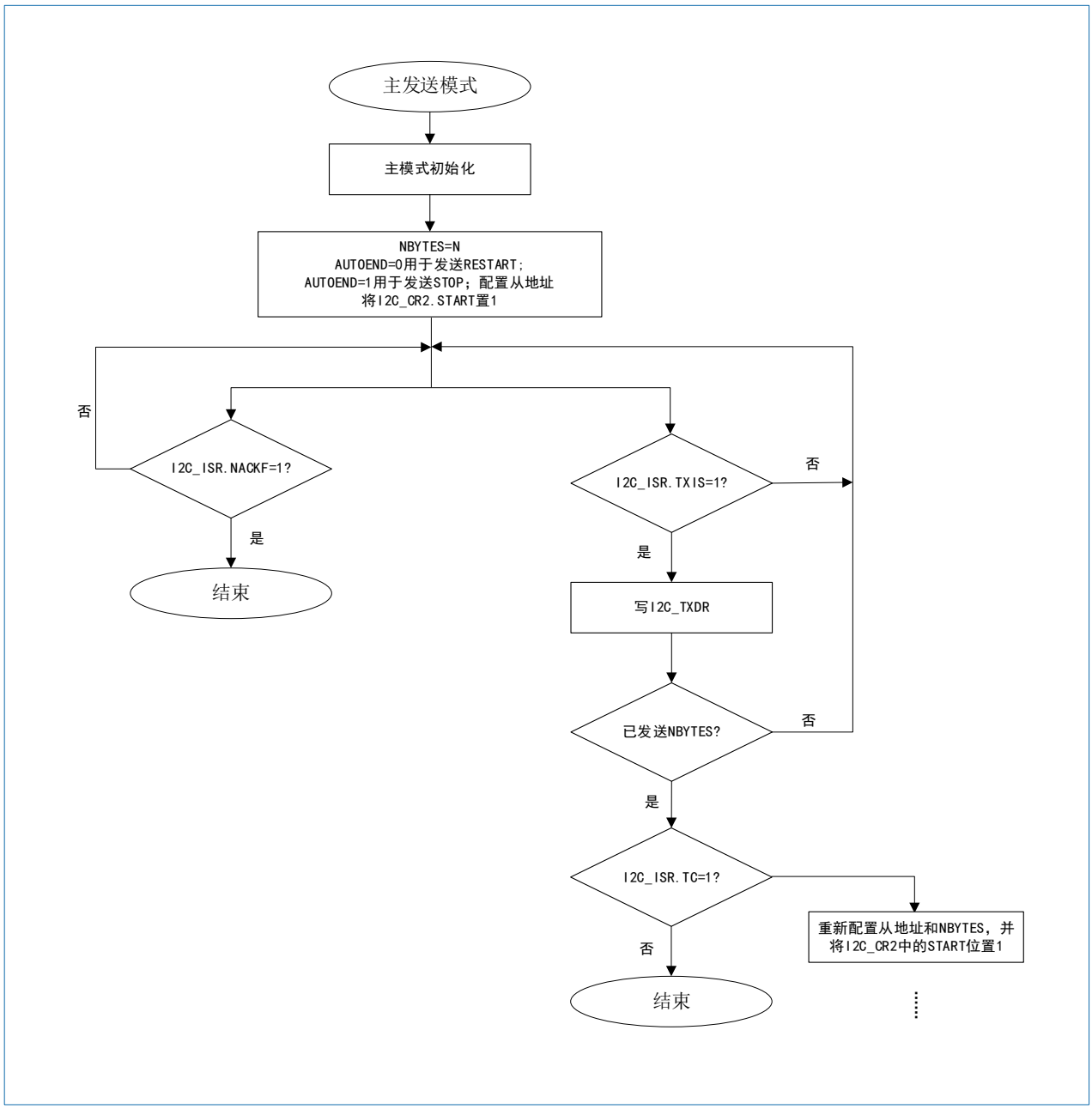

图 19-18 I2C 主发送器的传输序列流程图（N≤255 字节）

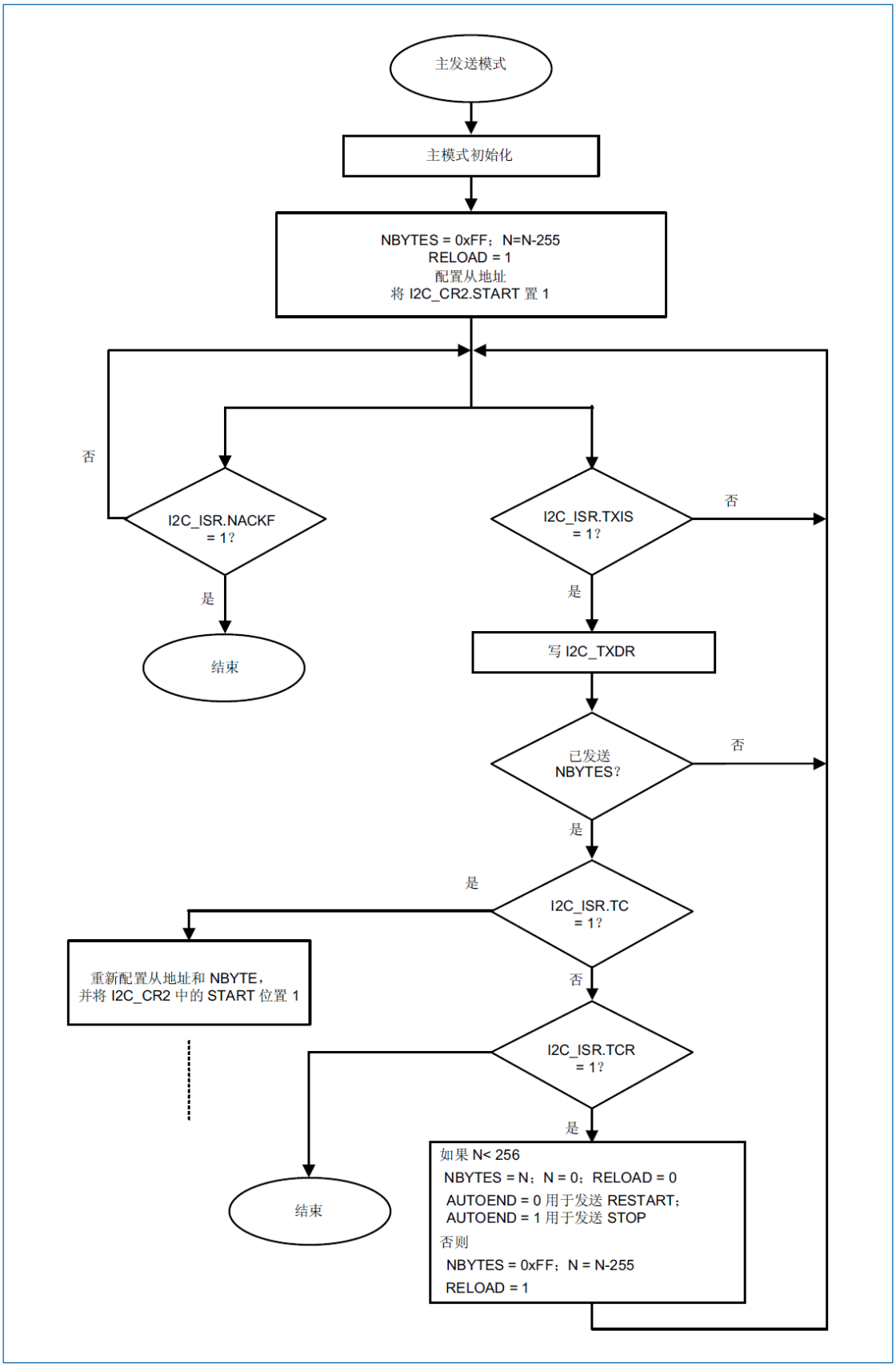

图 19-19 I2C 主发送器的传输序列流程图（N>255 字节）

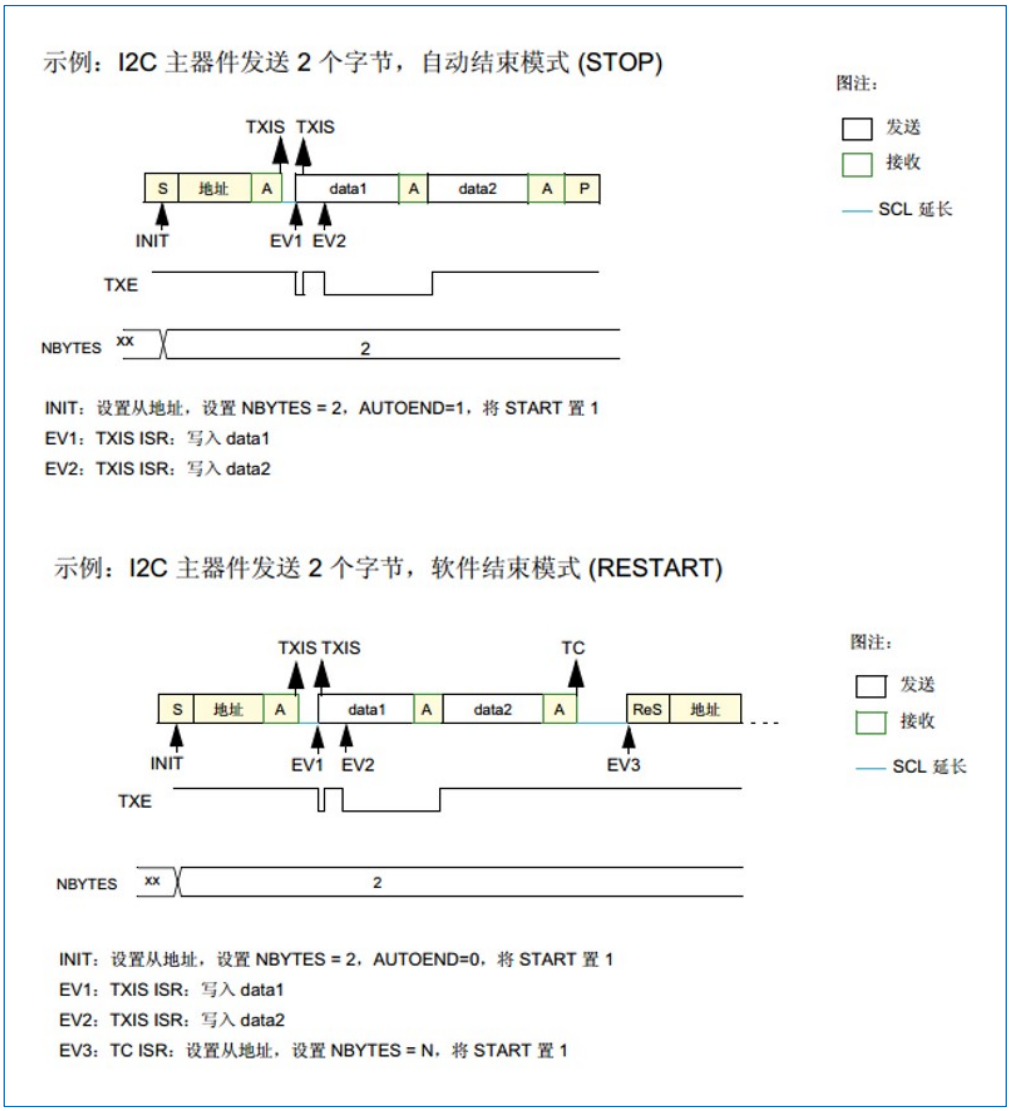

图 19-20 I2C 主发送器的传输总线图

主接收器

读传输时，在接收到每个字节（即第 8 个 SCL 脉冲）后，RXNE 标志将置 1。如果 I2C_CR1 寄存器中的 RXIE 位置 1，RXNE 事件将生成中断。读取 I2C_RXDR 时，将清零该标志。

如果待接收的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES[7:0]数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延展，直到 NBYTES[7:0]被写入非零值。

- 当 RELOAD=0 且 NBYTES[7:0]数据传输完成时：
  - 在自动结束模式（AUTOEND=1）下，接收到最后一个字节后，将自动发送 NACK 和停止位。
  - 在软件结束模式（AUTOEND=0）下，接收到最后一个字节后，将自动发送 NACK，TC 标志将置 1 且 SCL 线的低电平将被延展，以便执行以下软件操作：

  可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位，后跟从地址。

  可通过将 I2C_CR2 寄存器中的 STOP 位置 1 来请求停止位。将 STOP 位置 1 会将 TC 标志清零，并在总线上发送停止位。

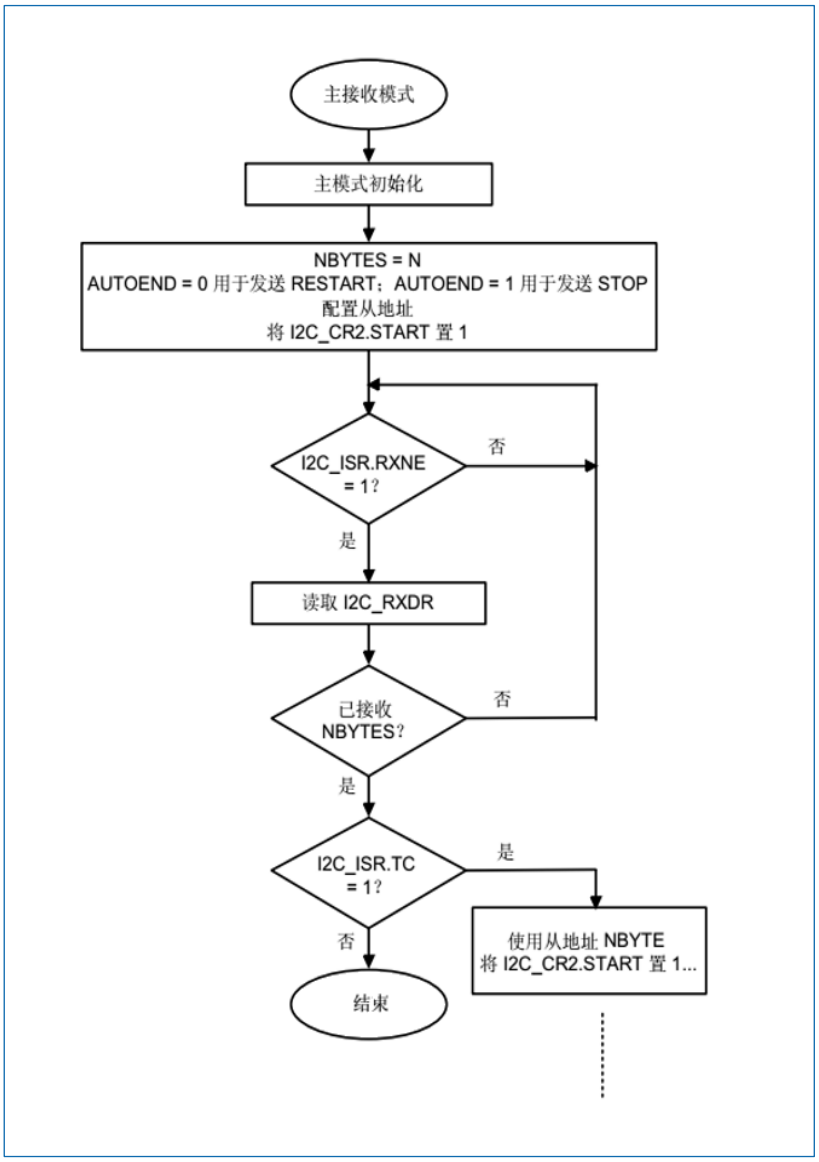

图 19-21 I2C 主接收器的传输序列流程图（N≤255 字节）

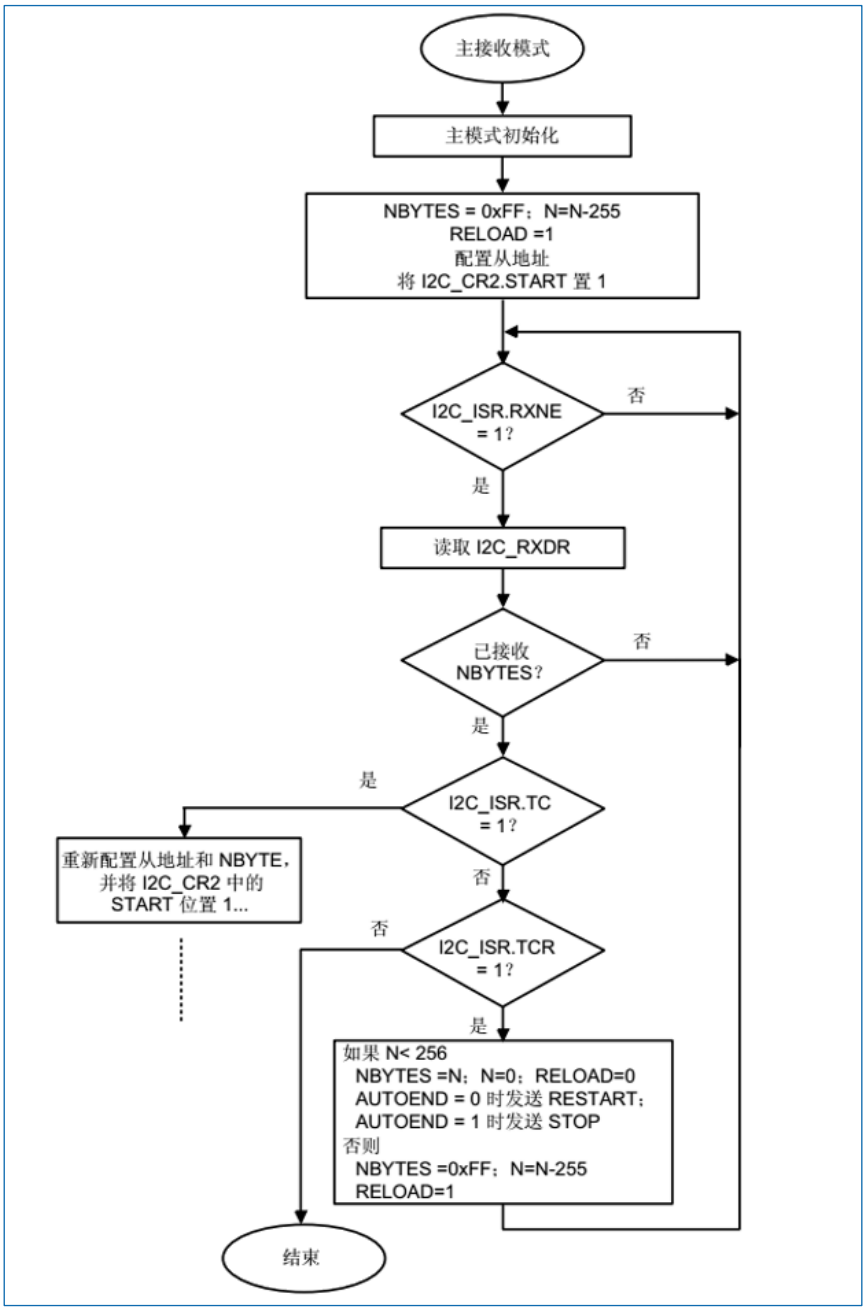

图 19-22 I2C 主接收器的传输序列流程图（N>255 字节）

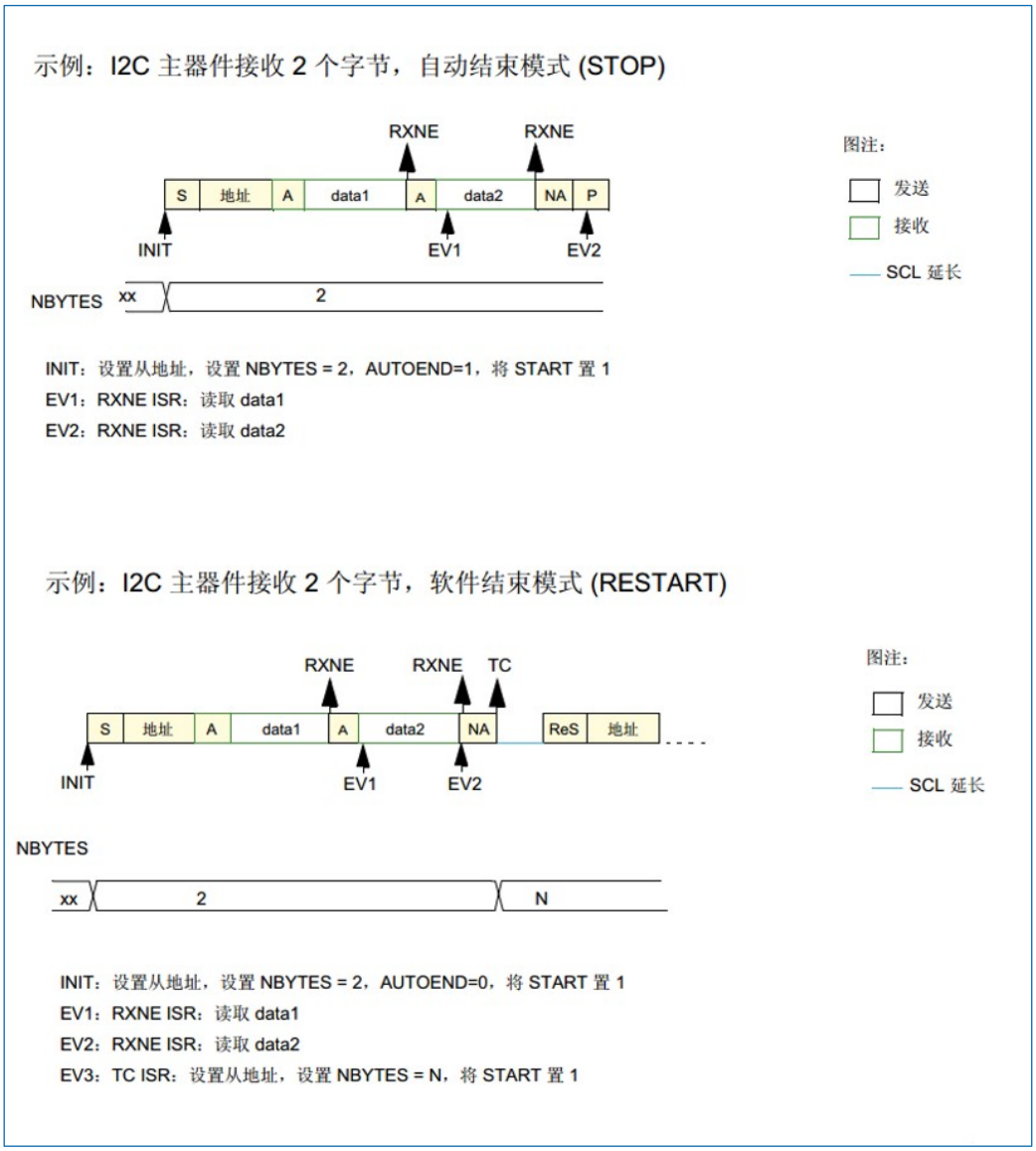

图 19-23 I2C 主接收器的传输总线图

19.2.9 I2C_TIMINGR 寄存器配置示例

下文各表提供了相应示例，以介绍如何编程 I2C_TIMINGR 才能获得符合 I2C 规范的时序。

表 19-5 f_{I2CCLK} = 8MHz 时的时序设置示例

参数	标准模式(Sm)		快速模式(Fm)
	10 kHz	100 kHz	400 kHz
PRESC	1	1	0
SCLL	0xC7	0x13	0x9
t_{SCLL}	200x250ns=50μs	20x250ns=5.0μs	10x125ns=1250ns
SCLH	0xC3	0xF	0x3
t_{SCLH}	196x250ns=49μs	16x250ns=4.0μs	4x125ns=500ns
t_{SCL}(1)	约 100μs(2)	约 10μs(2)	约 2500ns(3)
SDADEL	0x2	0x2	0x1

参数	标准模式(Sm)		快速模式(Fm)
	10 kHz	100 kHz	400 kHz
t_{SDADEL}	2x250ns=500ns	2x250ns=500ns	1x125ns=125ns
SCLDEL	0x4	0x4	0x3
t_{SCLDEL}	5x250ns=1250ns	5x250ns=1250ns	4x125ns=500ns

(1). 由于 SCL 内部检测存在延时，SCL 周期 t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 tSCL 提供的值仅用于举例说明。

(2). t_{SYNC1}+t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=1000 ns 时的示例。

(3). t_{SYNC1}+ t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=750 ns 时的示例。

(4). t_{SYNC1}+ t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=500 ns 时的示例。

表 19-6 f_{I2CCLK}=16MHz 时的时序设置示例

参数	标准模式(Sm)		快速模式(Fm)
	10kHz	100kHz	400kHz
PRESC	3	3	1
SCLL	0xC7	0x13	0x9
t_{SCLL}	200x250ns=50μs	20x250ns=5.0μs	10x125ns=1250ns
SCLH	0xC3	0xF	0x3
t_{SCLH}	196x250ns=49μs	16x250ns=4.0μs	4x125ns=500ns
t_{SCL}(1)	约 100μs(2)	约 10μs(2)	约 2500ns(3)
SDADEL	0x2	0x2	0x2
t_{SDADEL}	2x250ns=500ns	2x250ns=500ns	2x125ns=250ns
SCLDEL	0x4	0x4	0x3
t_{SCLDEL}	5x250ns=1250ns	5x250ns=1250ns	4x125ns=500ns

(1). 由于 SCL 内部检测存在延时，SCL 周期 TSCL 大于 t_{SCLL}+t_{SCLH}。为 TSCL 提供的值仅用于举例说明。

(2). t_{SYNC1}+t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=1000ns 时的示例。

(3). t_{SYNC1}+ t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=750ns 时的示例。

(4). t_{SYNC1}+ t_{SYNC2} 最小值为 4xt_{I2CCLK}=250ns。t_{SYNC1}+t_{SYNC2}=500ns 时的示例。

19.2.10 SMBus I2C 特性

系统管理总线（SMBus）是一个以 I2C 协议为基础的双线制接口，实现了总线上多设备之间的通信，主要用于系统和电源管理。SMBus 还可以使用 SMBA 信号实现设备之间的通信请求。

该外设与 SMBus 规范第 2.0 版兼容，该版本以 I2C 规范第 2.1 版为基础。SMBus 包括三类器件。

- 从器件，用于接收或响应命令。
- 主器件，用于发出命令、生成时钟和管理传输。
- 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。SMBus 系统中只允许存在一个主机。

SMBus 可配置为主器件或从器件，也可配置为主机。

总线协议

SMBus 协议包含 11 种可用命令协议。SMBus 器件可以使用任意一种协议进行通信。这 11 种协议分别为快速命令、发送字节、接收字节、写入字节、写入字、读取字节、读取字、过程调用、块读取、块写入以及块写入-块读取过程调用。这些协议由用户通过软件实现。

地址解析协议（ARP）

动态为新器件分配唯一地址可解决 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）。该 128 位数字由软件实现。

该外设支持地址解析协议（ARP）。通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默认地址（0b110 0001）。ARP 命令应通过用户软件实现。

在从模式下，通过仲裁来实现对 ARP 的支持。

有关 SMBus 地址解析协议的详细信息，请参见 SMBus 规范第 2.0 版。

接收的命令和数据应答控制 SMBus 接收器必须能够对接收到的每个命令或数据进行否定应答。要在从模式下实现 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。

主机通知协议

该外设通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来支持主机通知协议。在这种情况下，主机将应答 SMBus 主机地址（0b000 1000）。

使用主机通知协议时，连接到总线上的设备作为主器件，而主机作为从器件。

SMBus 报警

器件支持 SMBus ALERT 信号。只具备从功能的器件可通过 SMBALERT#引脚向主机发出信号，指示它想要通信。主机会处理该中断并通过报警响应地址（0b000 1100）同时访问所有 SMBALERT#器件。只有那些将 SMBALERT#拉到低电平的器件会应答报警响应地址。

如果配置为从器件（SMBHEN=0），通过将 I2C_CR1 寄存器中的 ALERTEN 位置 1 来将 SMBA 引脚拉为低电平。这同时还会使能报警响应地址。

如果配置为主机（SMBHEN=1），当 SMBA 引脚上检测到下降沿且 ALERTEN=1 时，I2C_ISR 寄存器中的 ALERT 标志置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。当 ALERTEN=0 时，即使外部 SMBA 引脚为低电平，也不会产生中断标志。

如果无需 SMBus ALERT 引脚，则当 ALERTEN=0 时，SMBA 引脚可用作标准 GPIO。

数据包错误校验

SMBus 规范中引入了数据包错误校验机制来实现可靠的数据传输。具体的实施方式是在每次数据传输结束时，附加数据包错误校验码（PEC）。PEC 的计算方式是对 START 到 STOP 之间的所有字节（包括地址和读/写位）使用 CRC-8 多项式 C（x）=x^8+x^2+x+1 进行计算。

内置的硬件 PEC 计算器：

在接收端：接收到的最后一个字节数据为 PEC 值，如果与硬件计算的 PEC 不匹配，将自动发送 NACK。

在发送端：发送的最后一个字节数据为 PEC 值。

超时

该外设内置了硬件定时器，以便符合 SMBus 规范第 2.0 版中定义的 3 个超时。

表 19-7 SMBus 超时规范

符号	参数	极限值		单位
		最小值	最大值	
$t_{TIMEOUT}$	检测时钟低电平超时	25	35	ms
$t_{LOW:SEXT}$(1)	累积时钟低电平延长时间（从器件）	-	25	ms
$t_{LOW:MEXT}$(2)	累积时钟低电平延长时间（主器件）	-	10	ms

(1). $t_{LOW:SEXT}$ 是一段累积时间，即给定从器件在一条消息的最初起始到停止期间时钟信号可延展的时间。其他从器件或主器件也可能延长时钟，进而导致时钟低电平总延长时间超过 $t_{LOW:SEXT}$。因此，测量该参数时该器件应该是全速主器件寻址的唯一器件。

(2). $t_{LOW:MEXT}$ 是一段累积时间，即主器件在消息的每个字节（定义为 START 到 ACK、ACK 到 ACK 或 ACK 到 STOP）内时钟信号可延展的时间。从器件或其他主器件也可能延长时钟，进而导致时钟低电平总时间超过 $t_{LOW:MEXT}$（针对给定字节）。因此，测量该参数时该全速主器件只寻址一个从器件。

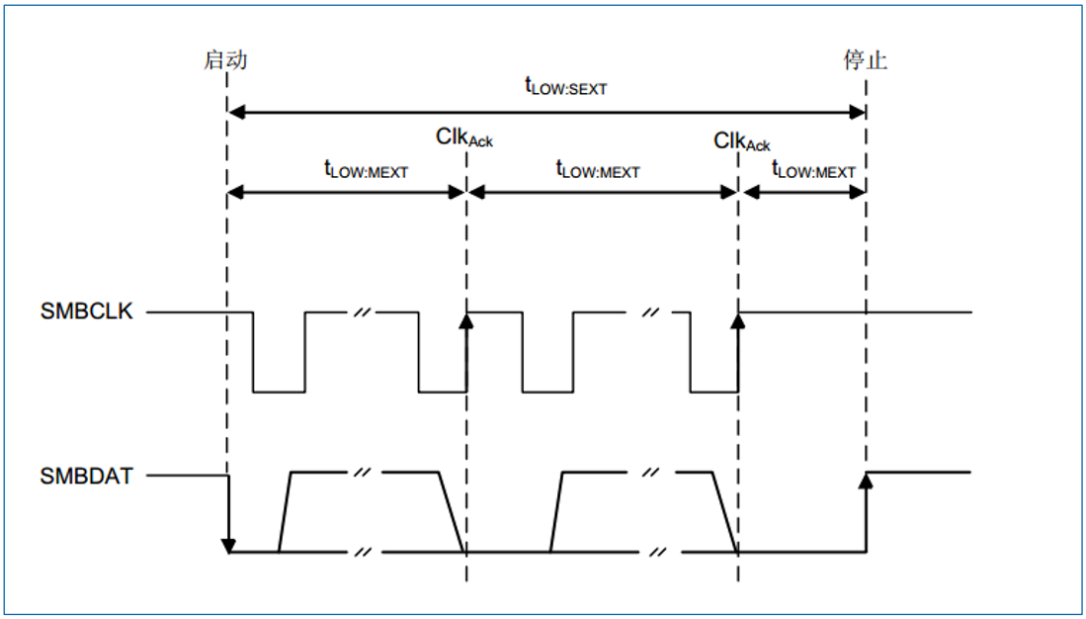

图 19-24 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔

总线空闲检测

如果主器件检测到时钟和数据信号的高电平时间已达 t_{IDLE}（超过 t_{HIGH}、MAX），则认为总线空闲。

该时序参数已考虑如下情况：主器件已动态添加至总线，但可能尚未检测到 SMBCLK 或 SMBDAT 线上的状态转换。在这种情况下，主器件必须等待足够长的时间，以确定当前未进行任何传输。外设支持硬件总线空闲检测。

19.2.11 SMBus 初始化

SMBus 的初始化包括 I2C 初始化之外，还必须进行一些其他的特定初始化，以便执行 SMBus 通信。

接收命令和数据应答控制（从模式）

SMBus 接收器必须能够对接收到的每个命令或数据进行否定应答。在从模式下，为了实现 ACK 控制，通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。

特定地址（从模式）

SMBus 作为从设备时，包含以下 3 个特殊地址。这 3 个地址的使能方法如下：

- 通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默认地址（0b110 0001）。
- 通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来使能 SMBus 主机地址（0b000 1000）。
- 通过将 I2C_CR1 寄存器中的 ALERTEN 位置 1 来使能报警响应地址（0b000 1100）。

数据包错误校验

通过将 I2C_CR1 寄存器中的 PECEN 位置 1 来使能 PEC 的计算。然后，借助硬件字节计数器（I2C_CR2 寄存器中的 NBYTES[7:0]）来管理 PEC 传输。使能 I2C 之前，必须配置 PECEN 位。

PEC 传输由硬件字节计数器来管理，因此在从模式下连接 SMBus 时必须将 SBC 位置 1。当 PECBYTE 位置 1 且 RELOAD 位清零时，传输完 NBYTES-1 字节的数据后会传输 PEC。如果 RELOAD 置 1，PECBYTE 将不起作用。

说明：使能 I2C 时，不允许更改 PECEN 配置。

表 19-8 带 PEC 的 SMBus 配置

模式	SBC 位	RELOAD 位	AUTOEND 位	PECBYTE 位
主 Tx/RxNBYTES+PEC+Stop	x	0	1	1
主 Tx/RxNBYTES+PEC+ReSTART	x	0	0	1
从 Tx/Rx+PEC	1	0	x	1

超时检测

将 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须按如下方式编程：即在 SMBus 规范第 2.0 版规定的时间最大值之前检测出超时情况。

- $t_{TIMEOUT}$ 检查

要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0]位编程为定时器重载值，以检查 $t_{TIMEOUT}$ 参数。必须将 TIDLE 位配置为“0”，以检测 SCL 低电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 位置 1 来使能定时器。

如果 SCL 的低电平持续时间超过（TIMEOUTA+1）x 2048 x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。

说明：TIMEOUTEN 位置 1 时，不允许更改 TIMEOUTA[11:0]位和 TIDLE 位的配置。

- $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 检查

必须根据外设配置为主器件还是从器件来配置 TIMEOUTB 定时器，以便为从器件校验 $t_{LOW:SEXT}$，为主器件校验 $t_{LOW:MEXT}$。由于标准只规定了最大值，用户可以为这两个参数选择相同的值。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。

如果 SMBus 外设延展 SCL 的累积时间超过（TIMEOUTB+1）x 2048 x t_{I2CCLK}，并且达到“19.2.10 SMBus I2C 特性”中的“总线空闲检测”一节给出的超时间隔，则 I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。请参见表 19-8。

说明：TEXTEN 位置 1 时，不允许更改 TIMEOUTB 配置。

总线空闲检测

要使能 t_{IDLE} 检查，必须将 12 位 TIMEOUTA[11:0]字段编程为定时器重载值，以获取 t_{IDLE} 参数。必须将 TIDLE 位配置为“1”，以检测 SCL 和 SDA 高电平超时。然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过（TIMEOUTA+1）x 4 x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT

标志将置 1。

请参见表 19-9。

说明：TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

19.2.12 SMBus：I2C_TIMEOUTR 寄存器配置示例

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms

表 19-9 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$=25 ms）

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	$t_{TIMEOUT}$
8 MHz	0x61	0	1	98x2048x125ns=25 ms
16 MHz	0xC3	0	1	196x2048x62.5ns=25 ms
32 MHz	0x186	0	1	391x2048x31.25ns=25 ms

- 将 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的最大持续时间配置为 8 ms

表 19-10 不同 I2CCLK 频率下的 TIMEOUTB 设置示例

f_{I2CCLK}	TIMEOUTB[11:0]位	TEXTEN 位	$t_{LOW:EXT}$
8 MHz	0x1F	1	32 x 2048 x 125ns=8 ms
16 MHz	0x3F	1	64 x 2048 x 62.5ns=8 ms
32 MHz	0x7C	1	125 x 2048 x 31.25ns=8 ms

- 将 t_{IDLE} 的最大持续时间配置为 50 µs

表 19-11 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE}=50 µs）

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	t_{IDLE}
8 MHz	0x63	1	1	100 x 4 x 125 ns=50 µs
16 MHz	0xC7	1	1	200 x 4 x 62.5 ns=50 µs
32 MHz	0x18F	1	1	400 x 4 x 31.25 ns=50 µs

19.2.13 SMBus 模式

除了 I2C 模式传输管理之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 从发送器

在 SMBus 模式下，必须将 SBC 编程为“1”，以便在完成已编程数据字节数的传输后进行 PEC 传输。当 PECBYTE 位置 1 时，NBYTES[7:0]中编程的字节数包含 PEC 传输。在这种情况下，总 TXIS 中断数为 NBYTES-1，如果主器件在完成 NBYTES-1 字节的数据传输后请求传输额外的字节，则将自动发送 I2C_PECR 寄存器的内容。

说明：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

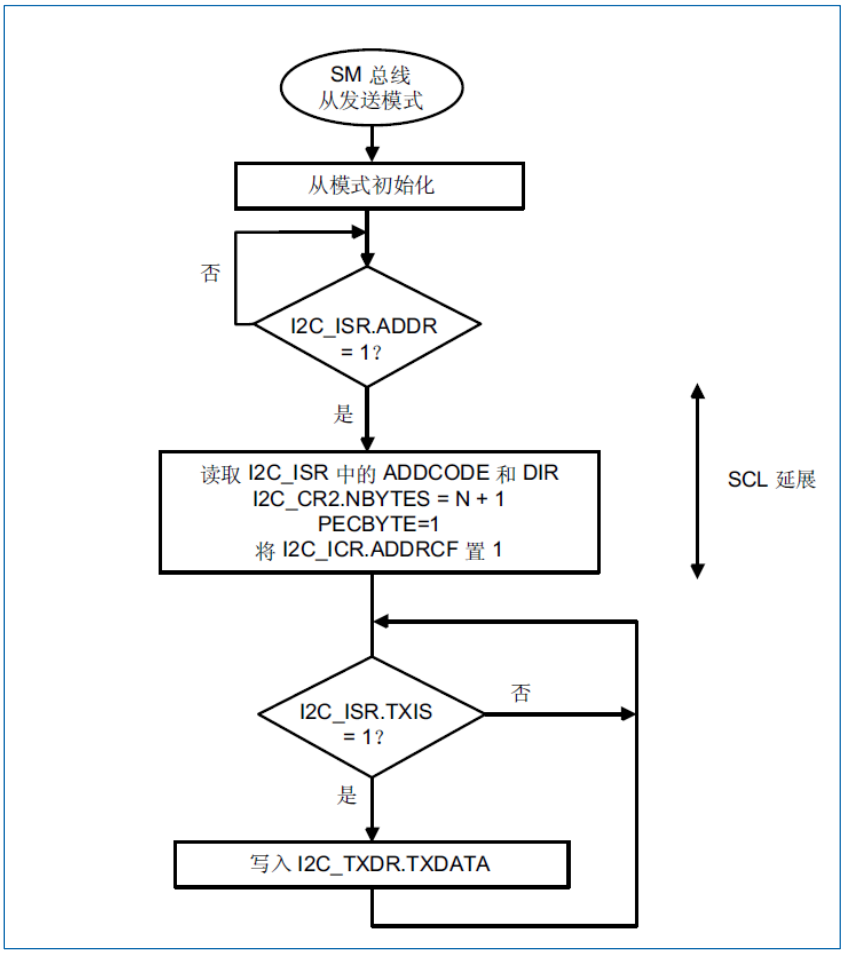

图 19-25 SMBus 从发送器的传输序列流程图（N 字节+PEC）

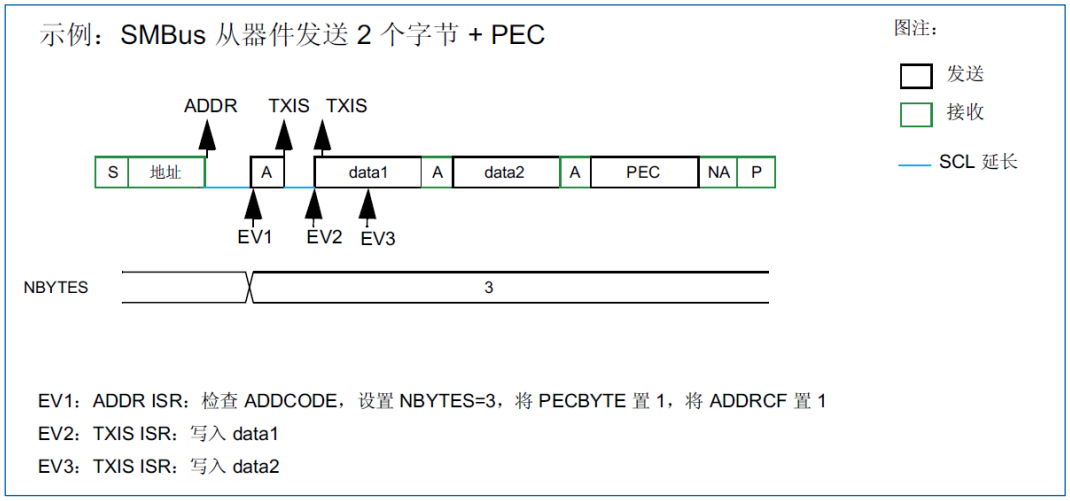

图 19-26 SMBus 从发送器的传输总线图（SBC=1）

SMBus 从接收器

在 SMBus 模式下使用 I2C 时，必须将 SBC 编程为“1”，以便在完成已编程数据字节数的传输后进行 PEC 校验。要对每个字节进行 ACK 控制，必须选择重载模式（RELOAD=1）。更多详细信息，请参见“19.2.7 从模式”中的“从器件字节控制模式”。

要校验 PEC 字节，必须将 RELOAD 位清零并将 PECBYTE 位置 1。在这种情况下，当接收到 NBYTES-1 字节的数据后，接收的下一个字节将与内部 I2C_PECR 寄存器的内容作比较。如果比较不匹配，则将自动生成 NACK 信号；如果比较匹配，则将自动生成 ACK 信号，而与 ACK 位的值无关。PEC 字节一经接收，便会像任何其他数据一样复制到 I2C_RXDR 寄存器中，并且 RXNE 标志将置 1。

当 PEC 不匹配时，PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

如果无需 ACK 软件控制，用户可编程 PECBYTE=1，在同一写操作下，将 NBYTES 编程为连续接收的字节数。接收到 NBYTES-1 字节的数据后，会将接收的下一个字节视为 PEC 进行校验。

说明：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

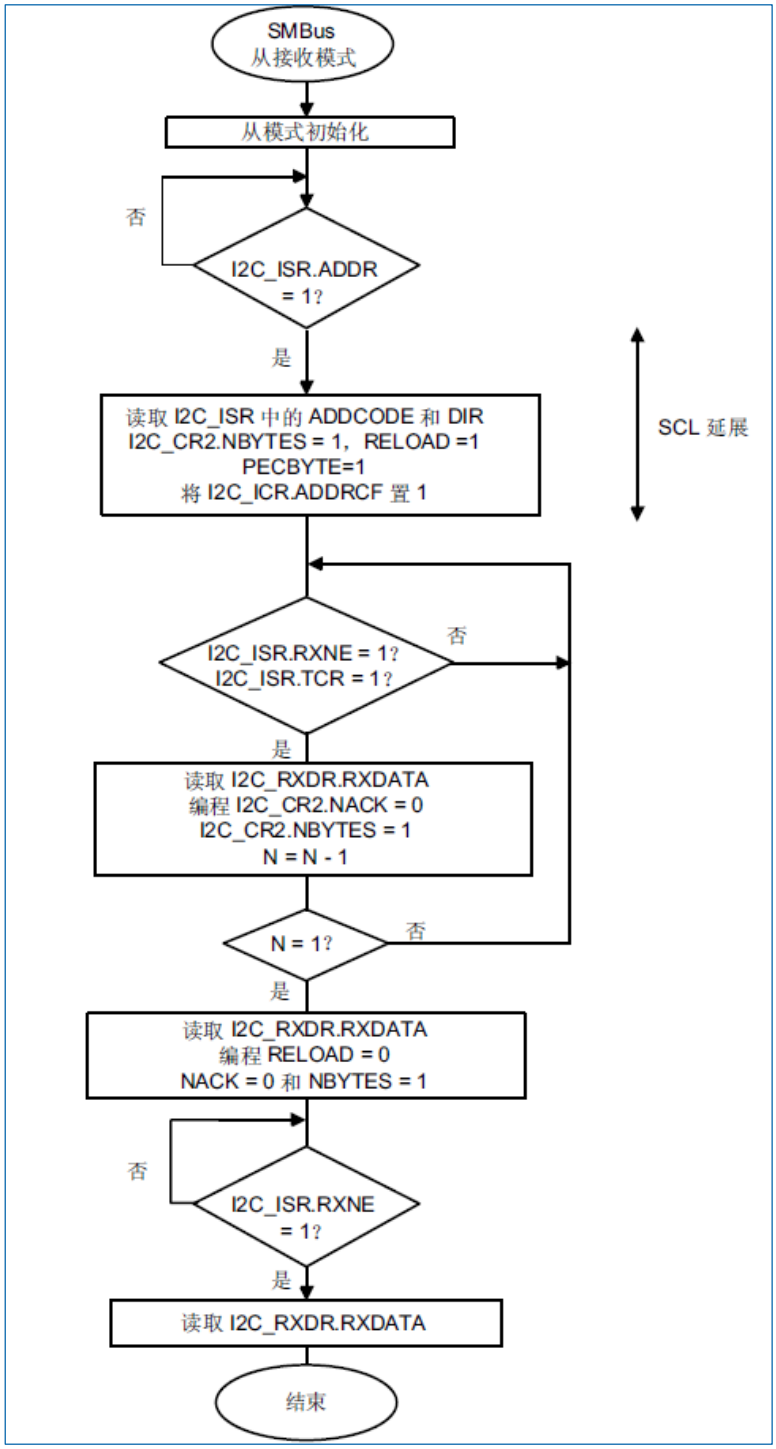

图 19-27 SMBus 从接收器的传输序列流程图（N 字节+PEC）

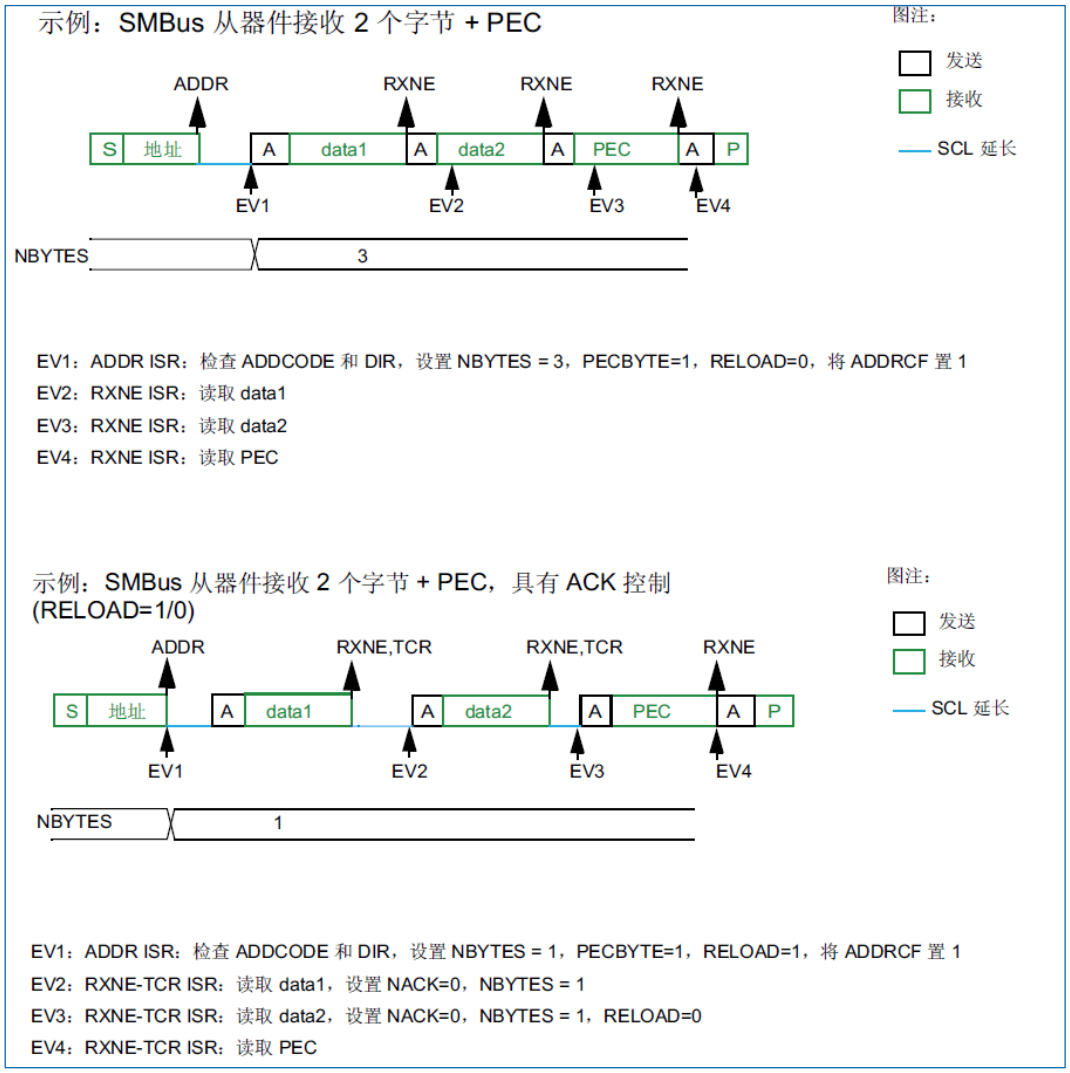

图 19-28 从接收器的总线传输图（SBC=1）

SMBus 主发送器

当 SMBus 主器件想要发送 PEC 时，必须在 START 位置 1 前，将 PECBYTE 位置 1 并在 NBYTES[7:0]字段中设置字节数。在这种情况下，总 TXIS 中断数为 NBYTES-1。因此，如果 PECBYTE 位在 NBYTES=0x1 时置 1，则将自动发送 I2C_PECR 寄存器的内容。

如果 SMBus 主器件想要在 PEC 后发送停止位，则应选择自动结束模式（AUTOEND=1）。在这种情况下，传输 PEC 后将自动发送停止位。

如果 SMBus 主器件想要在 PEC 后发送重复起始位，则必须选择软件模式（AUTOEND=0）。

在这种情况下，发送 NBYTES-1 字节的数据后，将发送 I2C_PECR 寄存器的内容，TC 标志将在传输完 PEC 之后置 1，SCL 线的低电平时间将延长。必须在 TC 中断子程序中设置重复起始位。

说明：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

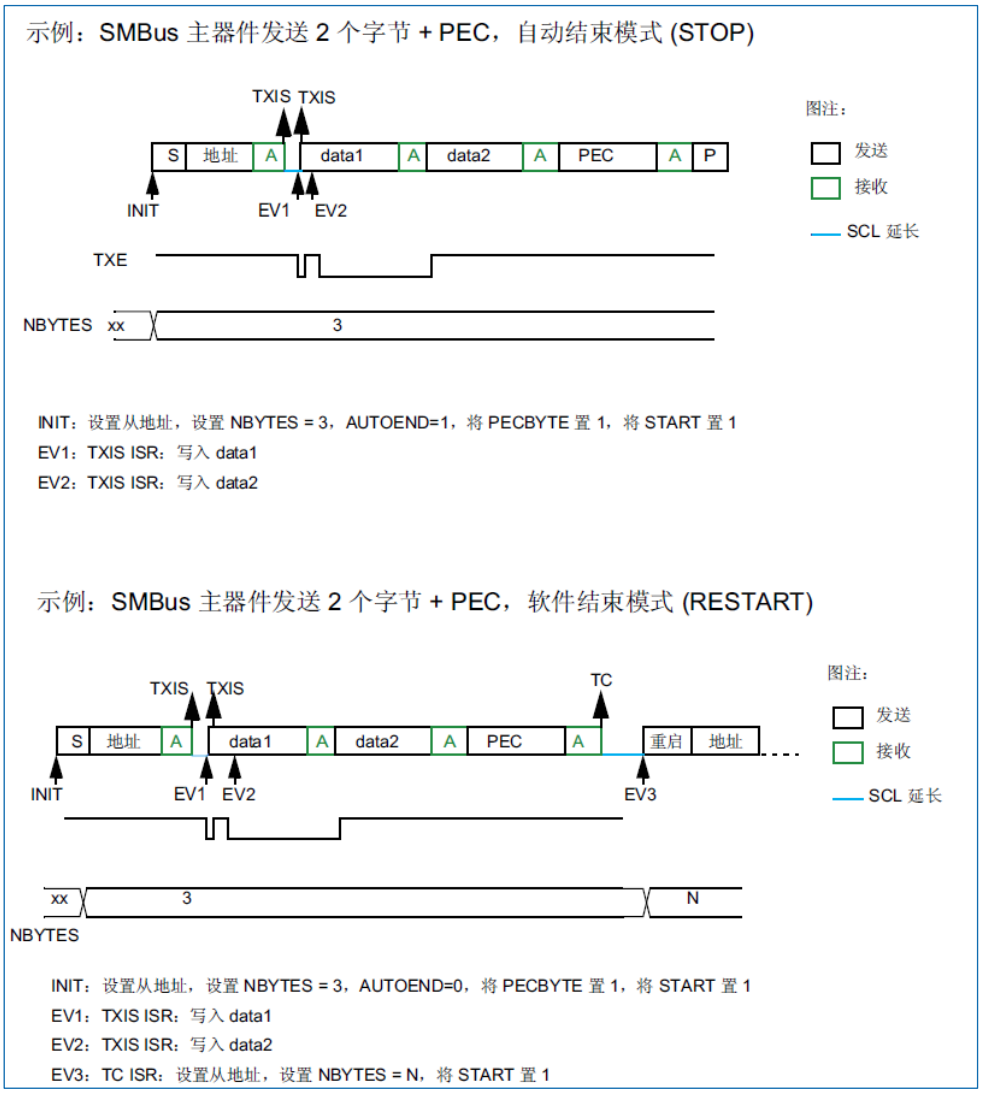

图 19-29 SMBus 主发送器的总线传输图

SMBus 主接收器

当 SMBus 主器件想要接收 PEC，并在传输结束后接收 STOP 时，可选择自动结束模式（AUTOEND=1）。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。在这种情况下，当接收到 NBYTES-1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。PEC 字节（其后跟有停止位）将得到 NACK 响应。

当 SMBus 主接收器想要接收 PEC 字节，并且在传输结束后接收重复起始位时，必须选择软件模式（AUTOEND=0）。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。

在这种情况下，当接收到 NBYTES-1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。接收到 PEC 字节后，TC 标志将置 1，SCL 线的低电平时间将延长。可以在 TC 中断子程序中设置重复起始位。

说明：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

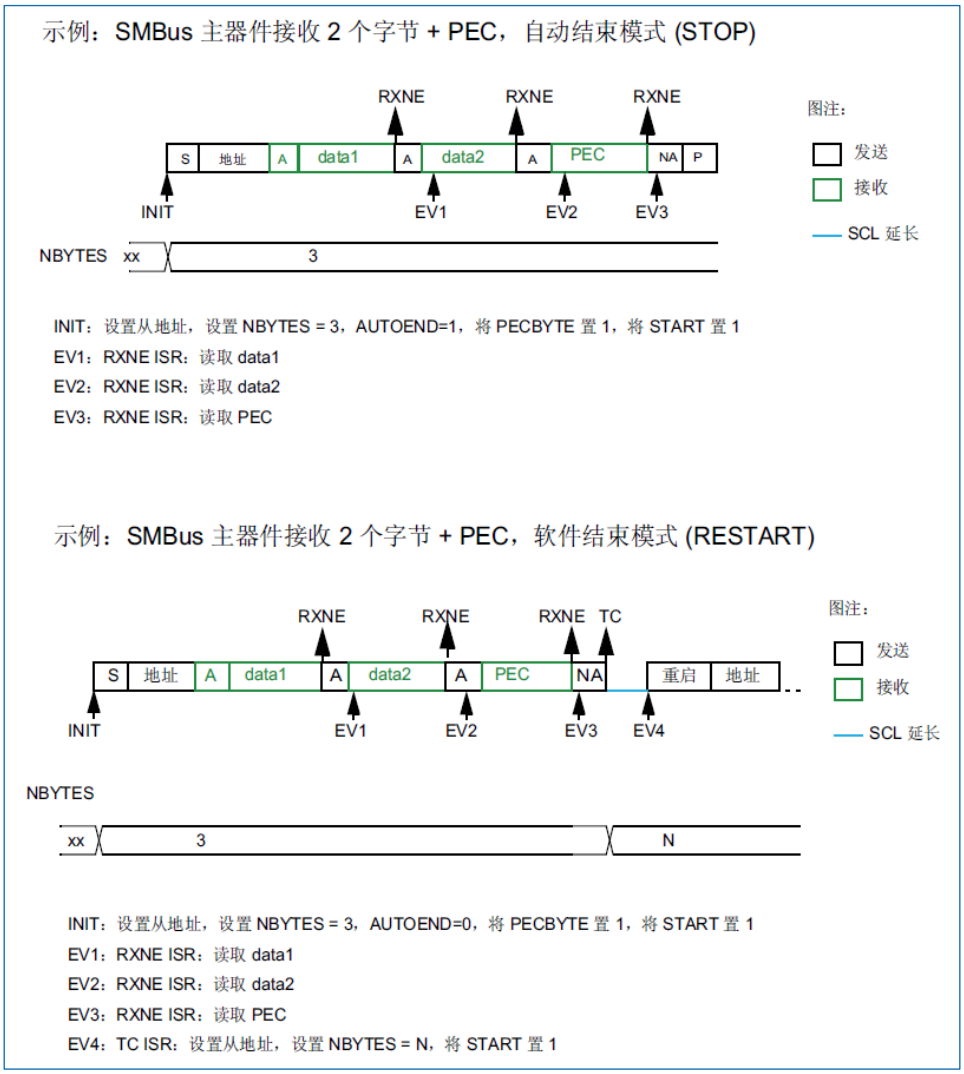

图 19-30 SMBus 主接收器的总线传输图

19.2.14 地址匹配时从停机模式唤醒

作为从设备被正确寻址时，I2C 能够将 MCU 从 Stop 模式唤醒（APB 时钟关断）。从 Stop 模式唤醒支持所有寻址模式。

要实现从 Stop 模式唤醒 MCU，需要注意以下几点：

- 如果 I2C 时钟是系统时钟，或者 WUPEN=0，则接收到 START 后，HSI 振荡器不会接通。只有当 I2C 时钟源为 HSI 振荡器时，该功能才可用。
- 数字滤波器与从停机模式唤醒功能不兼容。如果 DNF 位不等于 0，则将 WUPEN 位置 1 将不起任何作用。
- 必须使能时钟延长（NOSTRETCH=0）才能确保从停机模式唤醒功能正常工作。
- 如果禁止从停机模式唤醒（WUPEN=0），则在进入停机模式前必须禁止 I2C 外设（PE=0）。

将 I2C_CR1 寄存器中的 WUPEN 位置 1，可以使能从停机模式唤醒功能。对于 I2CCLK，必须选择 HSI 振荡器作为时钟源，以便从停机模式唤醒。

在停机模式中，HSI 关闭。当检测到 START 时，I2C 接口将 HSI 接通，并延长 SCL 低电平持续时间，直到 HSI 唤醒。HSI 随后用来接收地址。地址匹配的情况下，MCU 唤醒时间内，I2C 延长 SCL 的低电平持续时间。通过软件清除 ADDR 标志后，此延长操作被释放，传输正常进行。如果地址不匹配，HSI 再次关断，MCU 不被唤醒。

只有 ADDR 中断能够唤醒 MCU。因此，当 I2C 以主器件身份或在 ADDR 标志置 1 后以被寻址从器件

身份执行传输时，不要进入停机模式。将 ADDR 中断程序中的 SLEEPDEEP 位清零，然后仅在 STOPF 标志置 1 后将其置 1，可对此进行管理。

19.2.15 错误条件

以下错误条件可能导致通信失败。

总线错误（BERR）

当 SDA 边沿出现且 SCL 为高电平时，会检测到起始或停止位。当检测到起始位或停止位但不位于第 9N 个 SCL 时钟脉冲之后时，则认为出现了总线错误。

只有当 I2C 在传输过程中用作主器件或被寻址为从器件时（即未处于从模式下的地址阶段），才会将总线错误标志置 1。

在从模式下误检测到起始位或重复起始位时，I2C 会像接收到正确的起始位一样进入地址识别状态。

检测到总线错误时，I2C_ISR 寄存器中的 BERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

仲裁丢失（ARLO）

当 SDA 线上发送高电平但在 SCL 上升沿却采样到低电平时，会检测到仲裁丢失。

- 在主模式下，将在地址阶段、数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，SDA 线和 SCL 线被释放，起始控制位由硬件清零，主器件自动切换为从模式。
- 在从模式下，将在数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，传输停止，SCL 和 SDA 线被释放。
- 检测到仲裁丢失时，I2C_ISR 寄存器中的 ARLO 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

上溢/下溢错误（OVR）

当满足 NOSTRETCH=1 和以下条件时，将在从模式下检测到上溢或下溢错误：

- 在接收过程中接收到一个新的字节，但 RXDR 寄存器的值还未被读取。接收的新字节丢失，自动发送 NACK 来响应新字节。
- 在发送过程中：
  - 当 STOPF=1 且应发送第一个数据字节时。TXE=0 时发送 I2C_TXDR 寄存器的内容，否则发送 0xFF。
  - 应发送一个新字节但尚未向 I2C_TXDR 寄存器写入数据时，将发送 0xFF。
  - 检测到上溢或下溢错误时，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

数据包错误校验错误（PECERR）

当接收到的 PEC 字节与 I2C_PECR 寄存器的内容不匹配时，将检测到 PEC 错误。接收到错误的 PEC 后，将自动发送 NACK。

检测到 PEC 错误时，I2C_ISR 寄存器中的 PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

超时错误（TIMEOUT）

满足以下任何条件均会出现超时错误：

- TIDLE=0 且 SCL 的低电平持续时间达到 TIMEOUTA[11:0]位中定义的时间：这用于检测 SMBus 超时。
- TIDLE=1 且 SDA 和 SCL 的高电平持续时间达到 TIMEOUTA[11:0]位中定义的时间：这用于检测总

线空闲情况。

- 主器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0]位中定义的时间（SMBus $t_{LOW:MEXT}$ 参数）
- 从器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0]位中定义的时间（SMBus $t_{LOW:SEXT}$ 参数）

当在主模式下检测到超时时，将自动发送停止位。

当在从模式下检测到超时时，将自动释放 SDA 和 SCL 线。

检测到超时错误时，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

报警（ALERT）

当 I2C 接口配置为主机（SMBHEN=1）、使能了报警引脚检测（ALERTEN=1）并且在 SMBA 引脚上检测到下降沿时，ALERT 标志将置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。

19.2.16 调试模式

当 MCU 进入调试模式时（内核停止），SMBus 超时定时器会根据 DBG 模块中的 DBG_I2C_SMBus_TIMEOUT 配置位选择继续正常工作还是停止工作。

19.3 I2C 低功耗模式

表 19-12 低功耗模式

模式	说明
睡眠	对 I2C 通信无影响。 I2C 中断可使器件退出睡眠模式。
停机	I2C 模块的寄存器内容仍被保持。

19.4 I2C 中断

下表给出了 I2C 中断请求列表。

表 19-13 I2C 中断请求

中断事件	事件标志	事件标志/中断清除方法	中断使能控制位
接收缓冲区非空	RXNE	读取 I2C_RXDR 寄存器	RXIE
发送缓冲区中断状态	TXIS	写入 I2C_TXDR 寄存器	TXIE
停止位检测中断标志	STOPF	写入 STOPCF=1	STOPIE
传输完成等待重载	TCR	写入 I2C_CR2 （NBYTES[7:0]≠ 0）	TCIE
传输完成	TC	写入 START=1 或 Stop=1	
地址匹配	ADDR	写入 ADDRCF=1	ADDRIE
接收到 NACK 应答	NACKF	写入 NACKCF=1	NACKIE
总线错误（Bus error）	BERR	写入 BERRCF=1	ERRIE

中断事件	事件标志	事件标志/中断清除方法	中断使能控制位
仲裁丢失	ARLO	写入 ARLOCF=1	
上溢/下溢（Overrun/Underrun）	OVR	写入 OVRCF=1	
PEC 错误	PECERR	写入 PECERRCF=1	
超时/t_{LOW} 错误	TIMEOUT	写入 TIMEOUTCF=1	
SMBus 报警	ALERT	写入 ALERTCF=1	

根据产品实现的不同，上述所有中断既可以共享同一个中断向量（I2C 全局中断），也可以分配到 2 个不同的中断向量上（I2C 事件中断和 I2C 错误中断）。

要使能 I2C 中断，需按照以下顺序操作：

1. 配置 NVIC 中的 I2CIRQ 通道并将其使能。
2. 配置 I2C 以生成中断。

I2C 唤醒事件连接到 EXTI 控制器（参见“10.3 EXTI 寄存器”）。

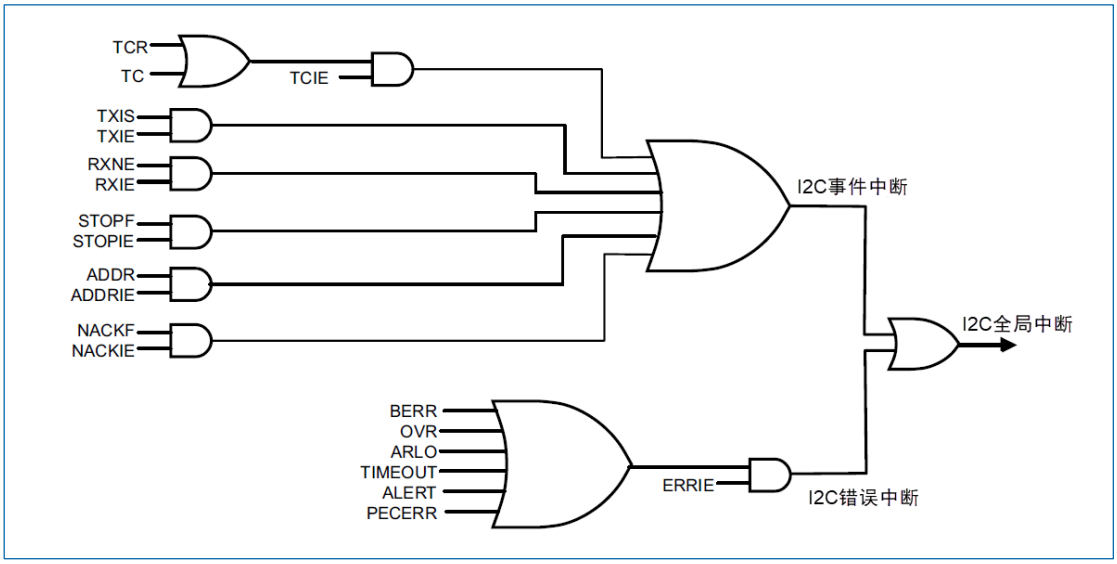

图 19-31 I2C 中断映射图

19.5 I2C 寄存器

基地址：0x4000 5400

空间大小：0x400

19.5.1 控制寄存器 1（I2C_CR1）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	WUPEN	NOSTRETCH	SBC
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			ANFOFF	DNF[3:0]				ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
			rw	rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31:24	Res：保留 必须保持复位值。
位 23	PECEN：启用 PEC（PEC enable） ● 0：禁用 PEC 计算 ● 1：启用 PEC 计算
位 22	ALERTEN：SMBus 通知使能（SMBus alert enable） ● 设备模式（SMBHEN=0）： ○ 0：释放 SMBA 引脚为高，并禁用 NACK 之后的通知响应地址头：0001100x。 ○ 1：拉低 SMBA 引脚并启用 ACK 之后的通知响应地址头：0001100x。 ● 主机（Host）模式（SMBHEN=1）： ○ 0：SMBus 通知引脚（SMBA）不支持。 ○ 1：支持 SMBus 通知引脚（SMBA）。
位 21	SMBDEN：SMBus 器件的默认地址启用（SMBus Device Default address enable） ● 0：禁用设备的默认地址。地址 0b1100001x 会被 NACK 应答。 ● 1：启用设备的默认地址。地址 0b1100001x 会被 ACK 应答。
位 20	SMBHEN：SMBus Host 地址启用（SMBus Host address enable） ● 0：禁用 Host 地址。地址 0b0001000x 会被 NACK 应答。 ● 1：启用 Host 地址。地址 0b0001000x 会被 ACK 应答。
位 19	GCEN：广播呼叫地址使能（General call enable） ● 0：禁止广播呼叫。地址 0b00000000 会被 NACK 应答。 ● 1：启用广播呼叫。地址 0b00000000 会被 ACK 应答。
位 18	WUPEN：从 Stop 模式唤醒（Wakeup from Stop mode enable） ● 0：禁止从 Stop 唤醒 ● 1：允许从 Stop 模式唤醒
位 17	NOSTRETCH：禁止时钟延长（Clock stretching disable） 该位用于在从机模式中禁止时钟延长。 ● 0：允许时钟延长 ● 1：禁止时钟延长
位 16	SBC：从机字节控制（Slave byte control） SBC 被置 1 时，I2C 的 SCL 和 SDA 线都被释放。 内部状态机和所有的状态为回到其复位值。控制位的内容被保留。 该位用于在从机模式使能硬件字节控制。 ● 0：从机字节控制模式关闭

	• 1：从机字节控制模式使能
位 15:13	Res：保留 必须保持复位值。
位 12	ANFOFF：模拟噪声滤波器关闭（Analog noise filter OFF） • 0：模拟噪声滤波器开启 • 1：模拟噪声滤波器关闭
位 11:8	DNF[3:0]：数字噪声滤波器（Digital noise filter） 该位域用来配置 SDA 和 SCL 输入上的数字噪声滤波器。数字滤波器会用 DNF[3:0]*t_{I2C_CLK} 的长度来工作。 • 0000：数字滤波器禁用 • 0001：数字滤波器启用，其滤波能力达到 1 个 t_{I2C_CLK}。 • … • 1111：数字滤波器启用，其滤波能力达到 15 个 t_{I2C_CLK}。
位 7	ERRIE：错误中断使能（Error interrupts enable） • 0：错误检测中断禁用 • 1：错误检测中断使能
位 6	TCIE：传输完成中断使能（Transfer Complete interrupt enable） • 0：传输完成中断禁用 • 1：传输完成中断使能
位 5	STOPIE：STOP 检测中断使能（STOP detection Interrupt enable） • 0：停止检测（STOPF）中断禁止 • 1：停止检测（STOPF）中断使能
位 4	NACKIE：收到 NACK 中断使能（Not acknowledge received Interrupt enable） • 0：NACKF 收到中断禁用 • 1：NACKF 收到中断允许
位 3	ADDRIE：地址匹配中断使能（仅从机）（Address match Interrupt enable, only for slaver） • 0：地址匹配（ADDR）中断禁用 • 1：启用地址匹配（ADDR）中断
位 2	RXIE：接收中断使能（RX Interrupt enable） • 0：接收（RXNE）中断禁止 • 1：启用接收（RXNE）中断
位 1	TXIE：发送中断使能（TX Interrupt enable） • 0：发送（TXIS）中断禁止 • 1：发送（TXIS）中断使能

位 0	PE：外设使能（Peripheral enable） • 0：外设禁用 • 1：外设使能

19.5.2 控制寄存器 2（I2C_CR2）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					PECBYTE	AUTOEND	RELOAD	NBYTES[7:0]							
					rs	rw	rw	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD10R	ADD10	RD_WRN	SADD[9:8]		SADD[7:1]							SADD[0]
rs	rs	rs	rw	rw	rw	rw		rw							rw

位 31:27	Res：保留 必须保持复位值。
位 26	PECBYTE：包错误检查字节（Packet error checking byte） 该位由软件置位。在 PEC 传输完后、在收到 STOP 条件后、在收到地址匹配事件后、以及在 PE=0 或者 SWRST 被写为 1 的时候都会被硬件清零。 • 0：没有 PEC 传送。 • 1：要求发送/接收 PEC。
位 25	AUTOEND：自动结束模式（主机模式）（Automatic end mode, for master mode） 该位由软件置 1 和清零。 • 0：软件结束模式：当 NBYTES 个数据传输完毕后，TC 标志被置 1，SCL 被拉低。 • 1：自动结束模式：NBYTES 个数据传输完后，会自动发送一个停止条件。
位 24	RELOAD：NBYTES 重装载模式（NBYTES reload mode） 由软件设置和清除。 • 0：在传输完 NBYTES 个字节后，传输结束。 • 1：传输 NBYTES 个字节后，传输并不结束（NBYTES 将被重装载）。当 NBYTES 个数据传输完毕后，TC 标志被置 1，SCL 被拉低。
位 23:16	NBYTES[7:0]：字节数（Number of bytes） 这里写入要发送/接收的字节数。从机模式并且 SBC=0 的时候，该位域的值不起作用。
位 15	NACK：产生 NACK（从机模式）（NACK generation, for slave mode） 该位由软件置位，在 NACK 发送后由硬件清零，或者在收到 STOP 条件后，在收到地址匹配事件后以及在 PE=0 或者 SWRST 被写为 1 的时候，都会被硬件清零。 • 0：在当前字节收到后发送 ACK。 • 1：当前字节接收到后发送一个 NACK。

位 14	STOP：产生停止条件（主机模式）（STOP generation, for master mode） 该位由软件置 1，在检测到 STOP 条件或者 PE=0 或者 SWRST 被置 1 时由硬件清零。 在主机模式下： • 0：不产生 STOP 条件。 • 1：当前字节传输完后产生停止条件。
位 13	START：产生起始条件（Start generation） 该位通过软件设置，在发送完一个起始条件和地址序列之后由硬件清零，或者由于仲裁丢失、超时错误、PE=0 以及 SWRST 被置 1 等事件由硬件清零。该位也可以由软件向 I2C_ICR 寄存器的 ADDRCF 位写 1 来清零。 • 0：没有起始条件产生 • 1：产生 START/RESTART 条件 o 如果 I2C 已经是在主机模式下并且 AUTOEND=0，RELOAD=0，并且 NBYTES 个字节发送完毕后，设置该位会产生重复起始条件。 o 否则只要总线空闲，设置该位将会立即产生一个起始条件。
位 12	HEAD10R：10 位地址头只读方向（主接收器模式）（10-bit address header only read direction, for master receiver mode） • 0：主机发送完整的 10 位从机地址读序列：START+2 字节 10 位地址（写方向）+RESTART+10 位地址中的前 7 位（读方向）。 • 1：主机只发送 10 位地址的前 7 位，跟着是读方向。
位 11	ADD10：10 位地址模式（主机模式）（10-bit addressing mode, for master mode） • 0：主机按 7 位地址模式操作 • 1：主机按 10 位地址模式操作
位 10	RD_WRN：传输方向（主机模式）（Transfer direction, for master mode） • 0：主机请求一个写传输 • 1：主机请求一个读传输
位 9:8	SADD[9:8]：从机地址位 9:8（主机模式）（Slave address bit 9:8, for master mode） • 在 7 位地址模式下（ADD10=0）：该位域的值不起作用。 • 在 10 位地址模式下（ADD10=1）：该位域应写入要发送的从机地址位的 9:8。
位 7:1	SADD[7:1]：从机地址位 7:1（主机模式）（Slave address bit 7:1, for master mode） • 在 7 位地址模式下（ADD10=0）：该位域应写入要发送的 7 位从机地址位。 • 在 10 位地址模式下（ADD10=1）：该位域应写入要发送的从机地址位的 7:1。
位 0	SADD[0]：从机地址的 0 位（主模式）（Slave address bit 0, for master mode） • 在 7 位地址模式下（ADD10=0）：该位的值不起作用。 • 在 10 位地址模式下（ADD10=1）：该位应写入要发送的从机地址位的位 0。

19.5.3 本机地址 1 寄存器（I2C_OAR1）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA1EN	Res				OA1MODE	OA1[9:8]		OA1[7:1]							OA1[0]
rw					rw	rw		rw							rw

位 31:16	Res：保留 必须保持复位值。
位 15	OA1EN：本机地址 1 启用（Own Address 1 enable） • 0：禁用本机地址 1，接收到从机地址 OA1 后会用 NACK 回应。 • 1：本机地址 1 启用，接收到从机地址 OA1 后会用 ACK 回应。
位 14:11	Res：保留 必须保持复位值。
位 10	OA1MODE：本机地址 1 的 10 位模式（Own Address 1 10-bit mode） • 0：本机地址 1 是 7 位地址。 • 1：本机地址 1 是一个 10 位的地址。
位 9:8	OA1[9:8]：接口地址的 9:8 位（Interface address 9:8 bit） • 7 位地址模式：该位域的值不起作用 • 10 位地址模式：地址的位 9:8
位 7:1	OA1[7:1]：接口地址的 7:1（Interface address 7:1 bit）
位 0	OA1[0]：接口地址的 0 位（Interface address 0 bit） • 7 位地址模式：该位域的值不起作用 • 10 位地址模式：地址的 0 位

19.5.4 本机地址 2 寄存器（I2C_OAR2）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA2EN	Res				OA2MSK[2:0]			OA2[7:1]							Res
rw					rw			rw							

位 31:16	Res：保留 必须保持复位值。
位 15	OA2EN：本机地址 2 启用（Own Address 2 enable） • 0：禁用本机地址 2，接收到从机地址 OA2 后会用 NACK 回应。

	• 1：本机地址 2 启用，接收到从机地址 OA2 后会用 ACK 回应。
位 14:11	Res：保留 必须保持复位值。
位 10:8	OA2MSK[2:0]：本机地址 2 屏蔽（Own Address 2 masks） • 000：没有屏蔽 • 001：OA2[1]被屏蔽掉，可忽略。只有 OA2[7:2]参与比较。 • 010：OA2[2:1]被屏蔽掉，可忽略。只有 OA2[7:3]参与比较。 • 011：OA2[3:1]被屏蔽掉，可忽略。只有 OA2[7:4]参与比较。 • 100：OA2[4:1]被屏蔽掉，可忽略。只有 OA2[7:5]参与比较。 • 101：OA2[5:1]被屏蔽掉，可忽略。只有 OA2[7:6]参与比较。 • 110：OA2[6:1]被屏蔽掉，可忽略。只有 OA2[7]参与比较。 • 111：OA2[7:1]被屏蔽掉，可忽略。没有比较，所有的（除保留地址）收到的 7 位地址都会用 ACK 回应。
位 7:1	OA2[7:1]：接口地址位 7:1（Interface address）
位 0	Res：保留 必须保持复位值。

19.5.5 时序寄存器（I2C_TIMINGR）

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRESC[3:0]				Res				SCLDEL[3:0]				SDADEL[3:0]			
rw								rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCLH[7:0]								SCLL[7:0]							
rw								rw							

位 31:28	PRESC[3:0]：时序预分频器（Timing prescaler）
位 27:24	Res：保留 必须保持复位值。
位 23:20	SCLDEL[3:0]：数据建立时间（Data setup time） 此字段用于生成发送模式中 SDA 的沿和 SCL 上升沿之间的延迟 t_{SCLDEL}。t_{SCLDEL}=（SCLDEL+1）*t_{PRESC}
位 19:16	SDADEL[3:0]：数据保持时间（Data hold time） 此字段用于生成发送模式中 SCL 的下降沿和 SDA 的沿之间的延迟 t_{SDADEL}。 t_{SDADEL} = SDADEL* t_{PRESC}
位 15:8	SCLH[7:0]：SCL 高电平时间（主机模式）（SCL high period, for master mode）

	此字段用于在主机模式下产生 SCL 的高电平时间。 t_{SCLH}=（SCLH+1）*t_{PRESC}
位 7:0	SCLL[7:0]：SCL 低电平时间（主机模式）（SCL low period, for master mode） 此字段用于在主机模式下产生 SCL 的低电平时间。 t_{SCLL}=（SCLL+1）*t_{PRESC}

19.5.6 超时寄存器（I2C_TIMEOUTR）

偏移地址：0x14

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMEOUTEN	Res		TIDLE	TIMEOUTA[11:0]											
rw			rw	rw											

位 31	TEXTEN：外部时钟超时启用（Extended clock timeout enable） • 0：外部时钟超时检测被禁用 • 1：外部时钟超时检测启用
位 30:28	Res：保留 必须保持复位值。
位 27:16	TIMEOUTB[11:0]：总线超时 B（Bus timeout B） 此字段用于配置累计时钟延长超时： • 在主机模式下，要检测的累计主机时钟低延长时间（$t_{LOW:MEXT}$）。 • 在从机模式下，要检测的累积从机时钟低延长时间（$t_{LOW:SEXT}$）。 $t_{LOW:EXT}$=（TIMEOUTB+1）*2048 * t_{I2C_CLK}
位 15	TIMEOUTEN：时钟超时检测启用（Clock timeout enable） • 0：SCL 超时检测被禁用 • 1：启用 SCL 超时检测：当 SCL 保持低的时间超过 TTIMEOUT（TIDLE=0）或保持高的时间超过 TIDLE（TIDLE=1）、会检测到一个超时错误（TIMEOUT=1）。
位 14:13	Res：保留 必须保持复位值。
位 12	TIDLE：空闲时钟超时检测（Idle clock timeout detection） • 0：TIMEOUTA 用于检测 SCL 低电平超时 • 1：TIMEOUTA 用于同时检测 SCL 和 SDA 高电平超时（总线空闲条件）
位 11:0	TIMEOUTA[11:0]：总线超时 A（Bus Timeout A） 此字段用于配置： • 在 TIDLE=0 的时候，SCL 低超时条件 $t_{TIMEOUT}$。

	$t_{TIMEOUT}$=（TIMEOUTA+1）* 2048 * t_{I2C_CLK} • TIDLE=1 的时候，总线空闲条件（SCL 和 SDA 同时为高）。 t_{IDLE}=（TIMEOUTA+1）* 4 * t_{I2C_CLK}

19.5.7 中断和状态寄存器（I2C_ISR）

偏移地址：0x18

复位值：0x00000001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								ADDCODE[6:0]							DIR
								r							r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	Res	ALERT	TIMEOUT	PECERR	OVR	ARLO	BERR	TCR	TC	STOPF	NACKF	ADDR	RXNE	TXIS	TXE
r		r	r	r	r	r	r	r	r	r	r	r	r	rs	rs

位 31:24	Res：保留 必须保持复位值。
位 23:17	ADDCODE[6:0]：匹配的地址码（从机模式）（Address match code, for slave mode） 该位域由地址匹配事件发生时所接收到的地址（ADDR=1）来更新。在 10 位地址的情况下，ADDCODE 提供 10 位地址的头 2 位以后的地址。
位 16	DIR：传输方向（从机模式）（Transfer direction, for slave mode） 该位在地址匹配事件发生时（ADDR=1）更新。 • 0：写传输，从机进入接收模式。 • 1：读传输，从机进入发送模式。
位 15	BUSY：总线忙（Bus busy）
位 14	Res：保留 必须保持复位值。
位 13	ALERT：SMBus 通知（SMBus alert） 在 SMBHEN=1（SMBusHost 配置）及 ALERTEN=1 的条件下，在 SMBA 脚上检测到 SMBALERT 事件（下降沿）时，该位由硬件置 1。通过设置 ALERTCF 位，该位由软件清零。
位 12	TIMEOUT：超时或 TLOW 检测标志（Timeout or t_{LOW} detection flag） 在超时或外部时钟超时发生时，该位被硬件置 1。通过设置 TIMEOUTCF 位，该位由软件清零。
位 11	PECERR：接收中的 PEC 错误（PEC Error in reception） 在收到的 PEC 值和 PEC 寄存器的内容不匹配时，该位由硬件置 1。收到错误的 PEC 后，会自动发送一个 NACK。通过将 PECCF 位置 1，该位由软件清零。
位 10	OVR：溢出/欠载（从机模式）（Overrun/Underrun, for slave mode）

	在从机模式下 NOSTRETCH=1 时，发生溢出/欠载错误的时候，该位由硬件置 1。通过设置 OVRCF 位，该位由软件清零。
位 9	ARLO：仲裁丢失（Arbitration lost） 在总线仲裁丢失的情况下，该位由硬件置 1。通过设置 ARLOCF 位，该位由软件清零。
位 8	BERR：总线错误（Bus error） 在检测到错位的起始或者停止条件的时候，该位由硬件置 1。通过设置 BERRCF 位，该位由软件清零。
位 7	TCR：传输完成重加载（Transfer Complete Reload） 在 RELOAD=1，并且 NBYTES 个数据发送完毕后，该位由硬件置 1。向 NBYTES 写入一个非零的值时，TCR 标志由软件清除。
位 6	TC：发送完毕（主机模式）（Transfer Complete, for master mode） 在 RELOAD=0、AUTOEND=0 并且 NBYTES 个数据发送完毕后，该位由硬件置 1。该位在软件将 START 或 STOP 位置 1 的时候清零。
位 5	STOPF：停止检测标志（STOP detection flag） 该位在外设参与传输的下列情况下，在总线上检测到一个停止条件时，由硬件置 1： • 作为主机，如果由外设生成一个停止条件的时候。 • 作为从机，如果外设在这次传输之前被正确的寻址到了。该位可以通过将 STOPCF 位置 1，由软件清零。
位 4	NACKF：收到 NACK 标志（Not Acknowledge received flag） 该位在一个字节传输后收到一个 NACK 的时候由硬件设置。该位可以通过将 NACKCF 位置 1，由软件清零。
位 3	ADDR：地址匹配（从机模式）（Address matched, for slave mode） 该位在收到的从机地址与其中一个有效的从机地址匹配的时候，由硬件置 1。通过设置 ADDRCF 位，该位由软件清零。
位 2	RXNE：接收数据寄存器非空（接收）（Receive data register not empty, for receivers） 该位在当接收到的数据被复制到 I2C_RXDR 寄存器，准备好被软件读取的时候由硬件置位。在读取 I2C_RXDR 时 RXNE 会被清除。
位 1	TXIS：发送中断状态（发送）（Transmit interrupt status, for transmitters） 在 I2C_TXDR 寄存器为空的时候由硬件置 1，这时必须把要发的数据写到 I2C_TXDR 寄存器。下一个要发送的数据被写到 I2C_TXDR 寄存器的时候它会被清除。该位只在 NOSTRETCH=1 的时候可以由软件写成 1，以生成一个 TXIS 事件（如果 TXIE=1 就有中断）。
位 0	TXE：发送数据寄存器空（发送）（Transmit data register empty, for transmitters） 在 I2C_TXDR 寄存器为空的时候由硬件置 1。下一个要发送的数据被写到 I2C_TXDR 寄存器的时候它会被清除。 该位可通过软件写 1，以清空发送数据寄存器 I2C_TXDR。

19.5.8 中断清除寄存器（I2C_ICR）

偏移地址：0x1C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		ALERTCF	TIMOUTCF	PECCF	OVRCF	ARLOCF	BERRCF	Res		STOPCF	NACKCF	ADDRCF	Res		
		w	w	w	w	w	w			w	w	w			

位 31:14	Res：保留 必须保持复位值。
位 13	ALERTCF：通知标志清零（Alert flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ALERT 标志位。
位 12	TIMOUTCF：超时检测标志清除（Timeout detection flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 TIMEOUT 标志位。
位 11	PECCF：PEC 错误标志清除（PEC Error flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 PECERR 标志位。
位 10	OVRCF：溢出/欠载标志清除（Overrun/Underrun flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 OVR 标志位。
位 9	ARLOCF：仲裁丢失标志清除（Arbitration Lost flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ARLO 标志位。
位 8	BERRCF：总线错误标志清除（Bus error flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 BERRF 标志位。
位 7:6	Res：保留 必须保持复位值。
位 5	STOPCF：停止检测标志清除（STOP detection flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 STOPF 标志位。
位 4	NACKCF：收到 NACK 标志清除（Not Acknowledge flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 NACKF 标志位。
位 3	ADDRCF：地址匹配标志清除（Address matched flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ADDR 标志位。对该位写 1，还会清除 I2C_CR2 寄存器中的 START 位。
位 2:0	Res：保留

	必须保持复位值。

19.5.9 PEC 寄存器（I2C_PECR）

偏移地址：0x20

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	PEC[7:0]：包错误检查寄存器（Packet error checking register） 当 PECEN=1 时，此字段包含内部 PEC 结果。PEC 在 PE=0 或 SWRST 被置 1 时，该位由硬件清零。

19.5.10 接收数据寄存器（I2C_RXDR）

偏移地址：0x24

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	RXDATA[7:0]：8 位接收数据（8-bit receive data） 该位域是从 I2C 总线接收的数据字节。

19.5.11 发送数据寄存器（I2C_TXDR）

偏移地址：0x28

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	TXDATA[7:0]：8 位发送数据（8-bit transmit data） 该位域是发送到 I2C 总线的数据字节。

20 串行外设接口（SPI）

SPI 接口使用 SPI 协议与外部器件进行通信。

串行外设接口（SPI）协议支持与外部器件进行半双工、全双工和单工同步串行通信。该接口可配置为主模式，在这种情况下，它可为外部从器件提供通信时钟（SCK）。该接口还能够在多主模式配置下工作。

20.1 SPI 主要特性

- 主模式或从模式操作
- 基于三条线的全双工同步传输
- 基于双线的半双工同步传输，其中一条可作为双向数据线
- 基于双线的单工同步传输，其中一条可作为单向数据线
- 4 位到 16 位传输帧格式选择
- 多主模式功能
- 8 个主模式波特率预分频系数，可达 $f_{PCLK}/2$
- 从模式频率可达 $f_{PCLK}/2$
- 对于主模式和从模式都可通过硬件或软件进行 NSS 管理：动态切换主/从操作
- 可编程的时钟极性和相位
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 支持 SPI Motorola 模式
- 确保可靠通信的硬件 CRC 功能
  - 在发送模式下可将 CRC 值作为最后一个字节发送。
  - 根据收到的最后一个字节自动进行 CRC 错误校验。
- 可触发中断的主模式故障和上溢标志
- CRC 错误标志
- 2 个 32 位嵌入式 TX 和 RX FIFO
- 支持 SPI TI 模式

20.2 SPI 实现

表 20-1 SPI 实现

SPI 特性	SPI
硬件 CRC 计算	支持
RX/TX FIFO	支持
NSS 脉冲模式	支持
TI 模式	支持

20.3 SPI 功能说明

SPI 支持在 MCU 与外部器件之间进行同步串行通信。应用软件可通过轮询状态标志或使用专用 SPI 中断对通信进行管理。SPI 的主要组件及其交互方式如下图所示。

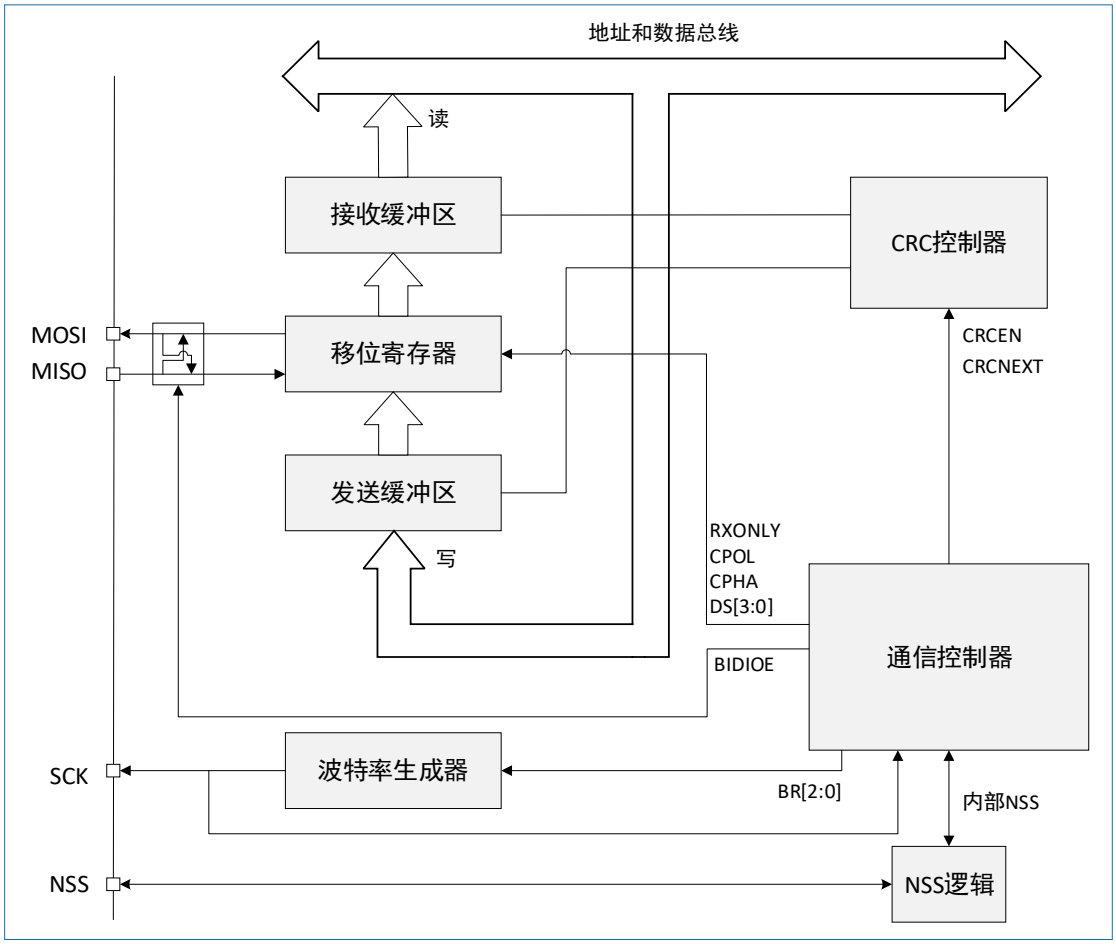

图 20-1 SPI 框图

四个 I/O 引脚专用于与外部器件进行 SPI 通信。

- MISO：主输入/从输出数据。通常情况下，此引脚用于在从模式下发送数据和在主模式下接收数据。
- MOSI：主输出/从输入数据。通常情况下，此引脚用于在主模式下发送数据和在从模式下接收数据。
- SCK：SPI 主器件的串行时钟输出引脚以及 SPI 从器件的串行时钟输入引脚。
- NSS：从器件选择引脚。根据 SPI 和 NSS 设置，该引脚可用于：
  - 选择单个从器件以进行通信
  - 同步数据帧
  - 检测多个主器件之间是否存在冲突。

更多详细信息，请参见“20.3.4 从器件选择（NSS）引脚管理”。

SPI 总线支持一个主器件与一个或多个从器件之间进行通信。该总线至少由两条线构成：一条用于时钟信号，另一条用于同步数据传输。其他信号可以根据 SPI 节点间的数据交换及其从器件选择信号管理进行添加。

20.3.1 一个主器件和一个从器件之间的通信

SPI 支持 MCU 基于目标器件和应用要求使用不同的配置进行通信。这些配置使用 2 条或 3 条线（通

过软件 NSS 管理），也可以使用 3 条或 4 条线（通过硬件 NSS 管理）。通信始终由主器件发起。

20.3.1.1 全双工通信

默认情况下，SPI 配置为全双工通信。在这种配置下，主器件和从器件的移位寄存器通过 MOSI 和 MISO 引脚之间的两条单向线连接。在 SPI 通信过程中，数据随主器件提供的 SCK 时钟边沿同步移位。主器件通过 MOSI 线将待发送的数据发送给从器件，通过 MISO 线从从器件接收数据。当数据帧传输完成时（所有位均移出），主器件和从器件之间即完成信息交换。

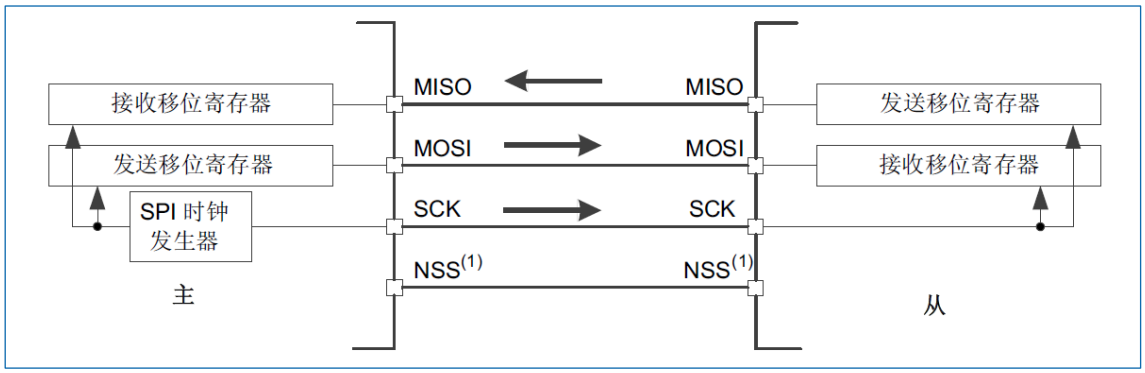

图 20-2 全双工单个主器件/单个从器件应用

(1). NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理硬件控制流。有关更多详细信息，请参见“20.3.4 从器件选择（NSS）引脚管理”。

20.3.1.2 半双工通信

通过将 SPI_CR1 寄存器的 BIDIMODE 位置 1，SPI 可采用半双工模式进行通信。在这种配置下，使用一条交叉连接线将主器件和从器件的移位寄存器连接起来。在此通信过程中，数据随 SCK 时钟边沿在移位寄存器之间进行移位，传输方向由主器件和从器件通过各自 SPI_CR1 寄存器中的 BIDIOE 位进行选择。

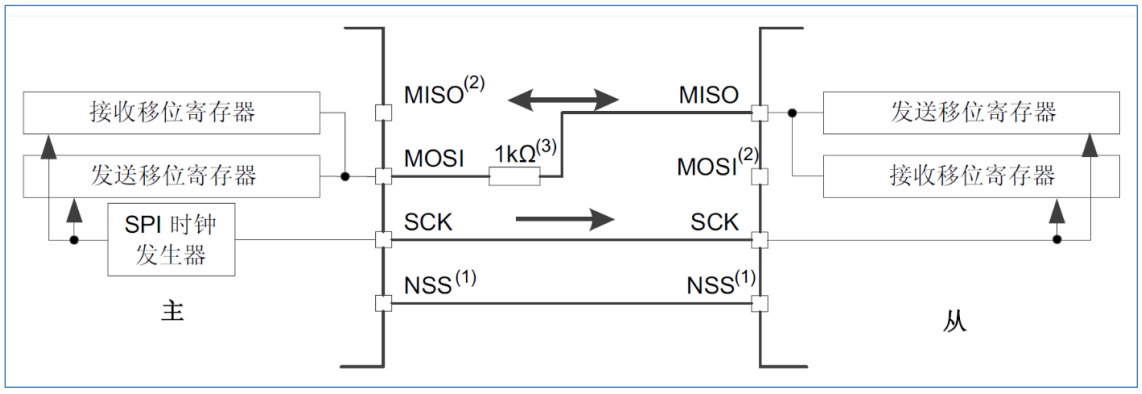

图 20-3 半双工单个主器件/单个从器件应用

(1). NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理硬件控制流。有关更多详细信息，请参见“20.3.4 从器件选择（NSS）引脚管理”。

(2). 在这种配置下，主器件的 MISO 引脚和从器件的 MOSI 引脚可用作 GPIO。

(3). 当以双向模式工作的两个节点间的通信方向不是同步变化时，会出现临界情况，新发送器访问共用数据线，而前一个发送器仍保持线路上的相反值（值取决于 SPI 配置和通信数据）。两个节点会出现冲突，在共用线上短暂提供相反的输出电平，直到下一个节点也相应地改变其方向设置。建议此模式下在 MISO 和 MOSI 引脚之间插入串行电阻以在这种情况下保护输出并限制电流在二者之间流过。

20.3.1.3 单工通信

通过 SPI_CR1 寄存器中的 RXONLY 位将 SPI 设置为只发送模式或只接收模式，可使 SPI 以单工模式进行通信。在这种配置下，仅使用一条线在主器件和从器件的移位寄存器之间进行传输。其余 MISO 和

MOSI 引脚对不用于通信，可用作标准 GPIO。

- 只发送模式（RXONLY=0）：其设置与全双工设置相同。应用必须忽略在未使用的输入引脚上捕获的信息。该引脚可以用作标准 GPIO。
- 只接收模式（RXONLY=1）：应用可通过将 RXONLY 位置 1 来禁止 SPI 输出功能。在从器件配置下，MISO 输出被禁止，该引脚可用作 GPIO。当从器件选择信号有效时，从器件继续从 MOSI 引脚接收数据（请参见“20.3.3 多主器件通信”）。基于数据缓冲区的配置产生接收数据事件。在主器件配置下，MOSI 输出被禁止，该引脚可用作 GPIO。只要 SPI 处于使能状态，则不断生成时钟信号。停止时钟的唯一方式是将 RXONLY 位或 SPE 位清零，直至来自 MISO 引脚的传入模式结束，然后基于相应配置填充数据缓冲区结构。

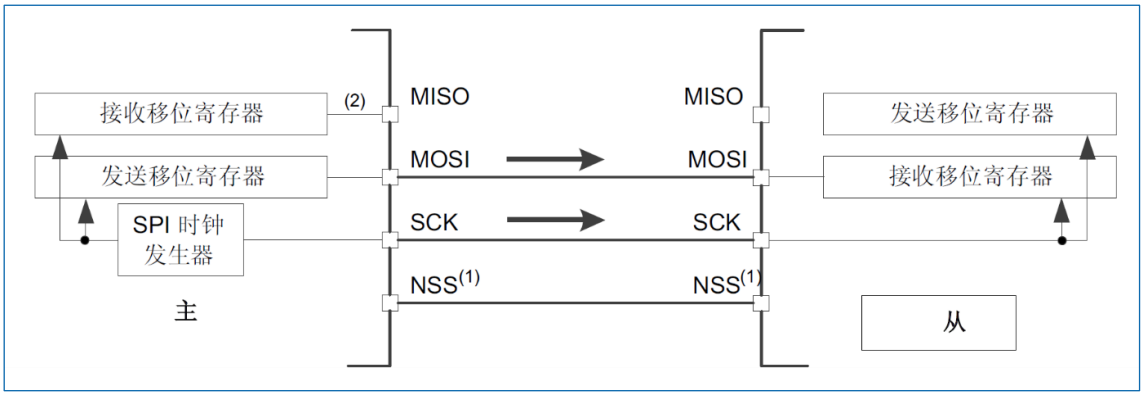

图 20-4 单工单个主器件/单个从器件应用（主器件为只发送模式/从器件为只接收模式）

(1). NSS 引脚可用于在主器件和从器件之间提供硬件控制流。外设也可选择不使用这些引脚。之后，必须在内部为主器件和从器件处理硬件控制流。有关更多详细信息，请参见“20.3.4 从器件选择（NSS）引脚管理”。

(2). 在发送器 RX 移位寄存器的输入上捕获意外输入信息。标准只发送模式下必须忽略与发送器接收流相关的所有事件（例如 OVF 标志）。

(3). 在这种配置下，两个 MISO 引脚均可用作 GPIO。

说明：任何单工通信均可以通过半双工通信的一种变型来替换，该变型中设置的数据传输方向不变（在 BIDIOE 位保持不变的同时，双向模式处于使能状态）。

20.3.2 标准多从器件通信

在具有两个或多个独立从器件的配置下，主器件使用 GPIO 引脚来管理每个从器件的片选线（请参见图 20-5）。主器件必须通过拉低与从器件 NSS 输入相连的 GPIO 的电平来单独选择一个从器件。执行该操作后，便建立了标准主器件与专用从器件之间的通信。

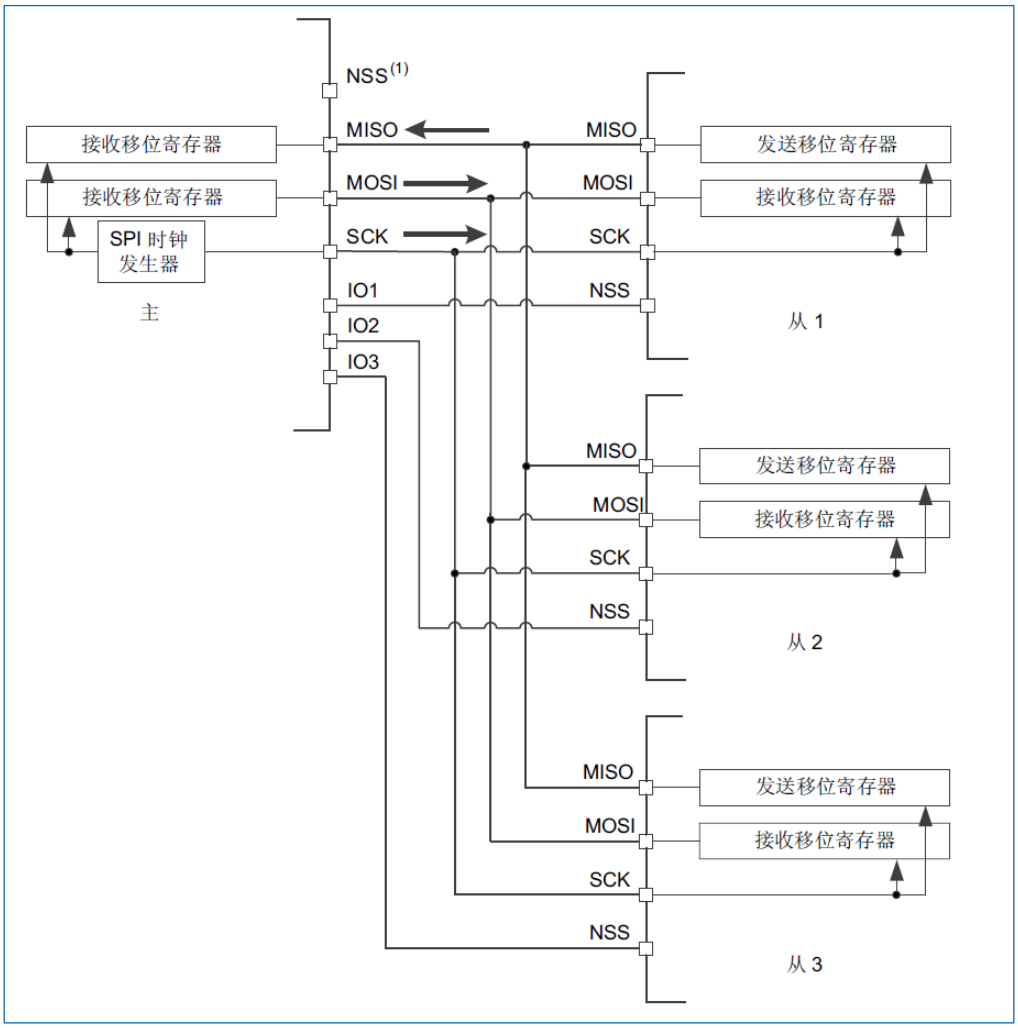

图 20-5 主器件和三个独立的从器件

(1). 此配置的主器件侧不使用 NSS 引脚。该引脚必须在内部管理（SSM=1，SSI=1）以避免任何 MODF 错误。

(2). 由于从器件的 MISO 引脚连在一起，所有从器件 MISO 引脚的 GPIO 配置必须设置为复用功能开漏。

20.3.3 多主器件通信

如果 SPI 总线未用于多主功能，用户可使用内置功能来检测两个尝试同时控制总线的节点间是否存在潜在冲突。对于该检测，NSS 引脚配置为硬件输入模式。

由于此时只有一个节点可将其输出施加到公用数据线上，因此无法连接超过两个以此模式工作的 SPI 节点。

当节点无效时，默认情况下均保持从模式。一旦一个节点要接管总线的控制，它会将自身切换到主模式，然后通过专用 GPIO 引脚向其他节点的从器件选择输入施加有效电平。会话完成后，有效的从器件选择信号将被释放，控制总线的节点会短暂切换回被动从模式，等待下一个会话开始。

如果两个节点同时发出各自的控制请求，则会出现总线冲突（请参见“20.5.3 SPI 状态寄存器（SPI_SR）”中 MODF 位所指示的模式故障）。随后，用户可应用某个简单的仲裁过程（例如，在两个节点上施加不同的预定义超时来推迟下一个尝试）。

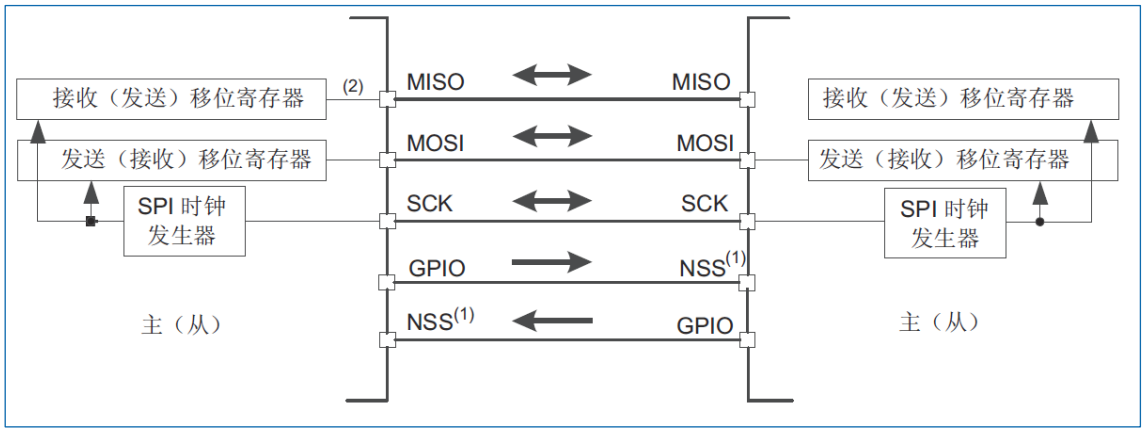

图 20-6 多主器件应用

(1). 在两个节点上，NSS 引脚配置为硬件输入模式。当无效节点配置为从器件时，其有效电平将使能 MISO 线输出控制。

20.3.4 从器件选择（NSS）引脚管理

在从模式下，NSS 用作标准的“片选”输入，使从器件与主器件进行通信。在主模式下，NSS 可用作输出或输入。NSS 用作输入时，可防止多主模式总线冲突；NSS 用作输出时，可驱动单个从器件的从器件选择信号。

可以使用 SPI_CR1 寄存器中的 SSM 位设置硬件或软件从器件选择管理：

- 软件 NSS 管理（SSM=1）：在这种配置下，由 SPI_CR1 寄存器中的 SSI 位的值决定内部驱动从器件选择信息。外部 NSS 引脚空闲，可供其他应用使用。
- 硬件 NSS 管理（SSM=0）：在这种配置下，所用配置取决于 NSS 输出配置（SPI_CR2 寄存器中的 SSOE 位）。可行的配置有两种：
  - NSS 输出使能（SSM=0 且 SSOE=1）：仅在 MCU 被设置为主器件时才使用该配置。NSS 引脚由硬件管理。只要在主模式下使能 SPI（SPE=1），NSS 信号便会被驱动为低电平，并且会一直保持低电平状态，直至禁止 SPI（SPE=0）。
  - NSS 输出禁止（SSM=0 且 SSOE=0）：如果 MCU 在总线上用作主器件，此配置可实现多主模式功能。如果在该模式下将 NSS 引脚拉至低电平，SPI 将进入主模式故障状态，器件将在从模式下自动进行重新配置。在从模式下，NSS 引脚用作标准的“片选”输入，当 NSS 线为低电平时将选择从器件。

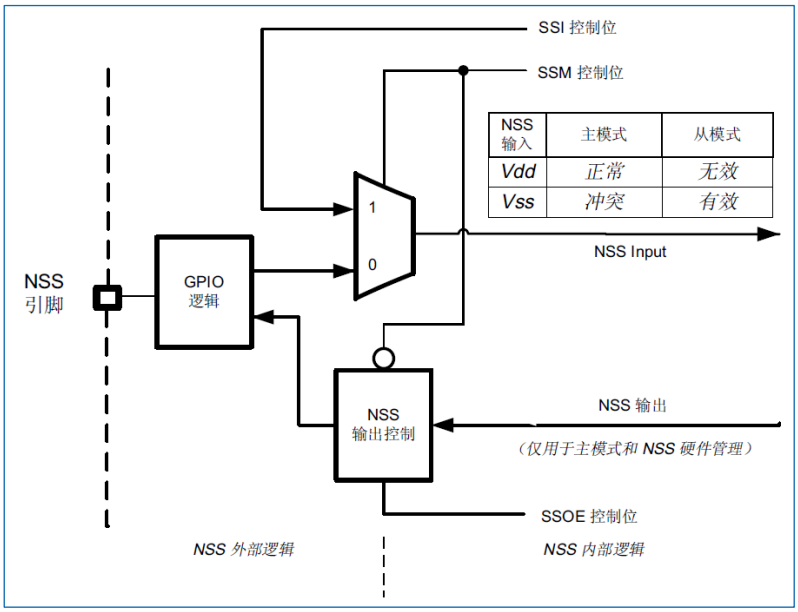

图 20-7 硬件/软件从器件选择管理

20.3.5 通信格式

SPI 通信过程中，将同时执行接收和发送操作。使用串行时钟（SCK）对数据线上的信息的移位和采样进行同步。通信格式取决于时钟相位、时钟极性和数据帧格式。为了能够在彼此间进行通信，主器件和从器件必须遵循相同的通信格式。

20.3.5.1 时钟相位和极性控制

通过 SPI_CR1 寄存器中的 CPOL 和 CPHA 位，可以用软件选择四种可能的时序关系。CPOL（时钟极性）位控制不传输任何数据时时钟的空闲状态值。该位对主器件和从器件都起作用。如果复位 CPOL，SCK 引脚则在空闲状态处于低电平。如果将 CPOL 置 1，SCK 引脚则在空闲状态处于高电平。

如果将 CPHA 位置 1，则会在 SCK 引脚的第二个边沿捕获传输的第一个数据位（如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。如果将 CPHA 位复位，则会在 SCK 引脚的第一个边沿捕获传输的第一个数据位（如果将 CPOL 位置 1，则为下降沿；如果将 CPOL 位复位，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。

CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

图 20-8 给出了在 CPHA 和 CPOL 位的四种组合下的 SPI 全双工传输。

说明：在切换 CPOL/CPHA 位之前，必须通过复位 SPE 位来禁止 SPI。

SCK 的空闲状态必须与 SPI_CR1 寄存器中选择的极性相对应（如果 CPOL=1，则上拉 SCK；如果 CPOL=0，则下拉 SCK）。

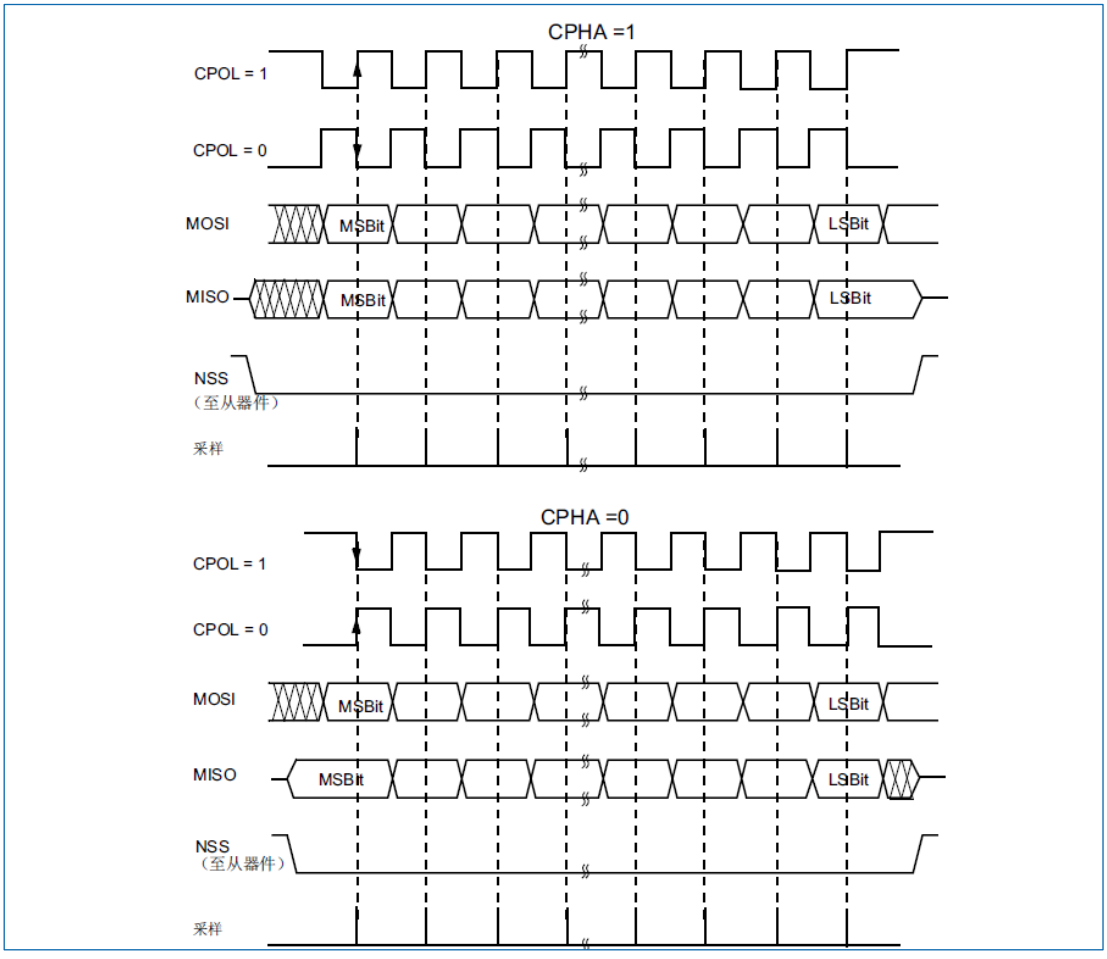

图 20-8 数据时钟时序图

说明：数据位的顺序取决于 LSBFIRST 位的设置。

20.3.5.2 数据帧格式

SPI 移位寄存器可设置为以 MSB 在前或 LSB 在前的方式移出数据，具体取决于 LSBFIRST 位的值。每个数据帧的长度均可配置为 4 位到 16 位，具体取决于 SPI_CR2 寄存器中的 DS 位编程的数据长度。所选的数据帧格式适用于发送和接收。无论选定的数据帧大小如何，对 FIFO 的读访问必须与 FRXTH 位的配置对齐。当访问 SPI_DR 寄存器时，数据帧总是右对齐为一个字节（如果数据为一个字节）或半个字。在通信期间，只有数据帧内的位被计时和传输。

20.3.6 SPI 配置

主器件和从器件的配置步骤几乎相同。对于具体的模式设置，请遵从相应章节的内容。若要对标准通信进行初始化，请执行以下步骤：

1. 对相应的 GPIO 寄存器执行写操作：将 MOSI、MISO 和 SCK 引脚配置为 GPIO。
2. 对 SPI_CR1 寄存器执行写操作：
   A. 通过 BR[2:0]位配置串行时钟波特率(1)。
   B. 配置 CPOL 位和 CPHA 位组合，从数据与时钟四组时序定义中选择一种定义。
   C. 通过配置 RXONLY 或 BIDIMODE 和 BIDIOE 来选择单工或半双工模式（RXONLY 和 BIDIMODE 不可同时置 1）。
   D. 配置 LSBFIRST 位以定义帧格式(2)。
   E. 如果需要 CRC，配置 CRCL 和 CRCEN 位（SCK 时钟信号处于空闲状态时）。
   F. 配置 SSM 和 SSI (2)。
   G. 配置 MSTR 位（在多主模式 NSS 配置下，为防止主器件发生 MODF 错误，应避免 NSS 引脚上出现状态冲突）。
3. 对 SPI_CR2 寄存器执行写操作：
   A. 配置 DS 位来设置数据帧格式（4 位到 16 位）
   B. 配置 SSOE(2) (3)(4)。
   C. 如果需要 TI 协议，请将 FRF 位置 1（TI 模式下需保持 NSSP 位为 0）。
   D. 如果两个数据单元之间需要 NSS 脉冲模式，则设置 NSSP 位（在 NSSP 模式下保持 CHPA 和清除 TI 位）。
   E. 配置 FRXTH 位。RXFIFO 阈值必须与 SPI_DR 寄存器的读取访问大小对齐。
4. 对 SPI_CRCPR 寄存器执行写操作：需要时配置 CRC 多项式。

注释：
(1). 从模式下无需此步骤，但从器件在 TI 模式下工作时除外。
(2). TI 模式下无需此步骤。
(3). 从模式下无需此步骤。
(4). NSSP 模式下无需此步骤。

20.3.7 使能 SPI 的步骤

建议在主器件发送时钟前使能 SPI 从器件。否则，数据传输可能会不正常。从器件的数据寄存器必须在通信时钟的第一个边沿之前写入待发送的数据，才能开始与主器件通信（如果时钟信号连续，则是在正在进行的通信结束前）。使能 SPI 从器件前，SCK 信号必须稳定为所选极性对应的空闲状态电平。

当 SPI 启用且 TXFIFO 不为空时，或在下一次写入 TXFIFO 时，处于全双工（或任何仅发送模式）的主机开始通信。

在任何主器件只接收模式（RXONLY=1 或 BIDIMODE=1，且 BIDIOE=0）下，使能 SPI 后，主器件立即

开始通信且时钟立即开始运行。

从器件接收到来自主器件的正确时钟信号时，它将开始通信。在 SPI 主器件启动传输前，从软件必须写入待发送的数据。

20.3.8 数据发送和接收过程

20.3.8.1 RXFIFO 和 TXFIFO

SPI 数据传输都使用 32 位嵌入式 FIFO。这使 SPI 能够连续的工作，并且在数据帧较小时防止溢出。每个方向都有自己的 FIFO，称为 TXFIFO 和 RXFIFO。除了启用 CRC 计算的仅接收器模式（从或主）外，这些 FIFO 用于所有 SPI 模式（参见“20.3.14 CRC 计算”）。

FIFO 的处理取决于数据交换模式（双工、单工）、数据帧格式（帧中的位数）、对 FIFO 数据寄存器执行的访问大小（8 位或 16 位），以及访问 FIFO 时是否使用数据组包（参见“20.3.13 TI 模式”）。

对 SPI_DR 寄存器的读取访问将返回 RXFIFO 中存储的尚未读取的第一个数值。对 SPI_DR 的写入访问将写入的数据存储在发送队列末尾的 TXFIFO 中。读取访问必须始终与 SPI_CR2 寄存器中的 FRXTH 位配置的 RXFIFO 阈值对齐。FTLVL[1:0]和 FRLVL[1:0]位表示两个 FIFO 的当前占用水平。

对 SPI_DR 寄存器的读取访问必须由 RXNE 事件管理。当数据存储在 RXFIFO 中且达到阈值（由 FRXTH 位定义）时，会触发此事件。清除 RXNE 时，RXFIFO 被视为空。以类似的方式，要传输的数据帧的写访问由 TXE 事件管理。TXFIFO 级别小于或等于其容量的一半时触发此事件。否则，TXE 被清除，TXFIFO 被视为已满。这样，RXFIFO 最多可以存储四个数据帧，而 TXFIFO 在数据帧格式不大于 8 位时最多只能存储三个数据帧。当软件试图以 16 位模式将更多数据写入 TXFIFO 时，此差异可防止 TXFIFO 中已存储的 3x8 位数据帧可能损坏。TXE 和 RXNE 事件都可以通过中断进行轮询或处理。

如果接收到下一个数据时 RXFIFO 的是满的，将导致发生溢出事件（参见“20.3.10 SPI 状态标志”）。溢出事件可以查询处理或中断处理。

当前数据帧传输正在进行时，BSY 位将置 1。当时钟信号连续运行时，BSY 标志在主器件侧的两个数据帧间保持置 1。不过，在从器件侧，它将在每个数据帧传输之间变为 0 并持续至少一个 SPI 时钟周期。

20.3.8.2 序列处理

多个数据字节可以顺序发送以组成一个消息。启用发送后，在主机 TXFIFO 中的数据会开始发送并连续发完。主机会连续输出时钟信号，直至 TXFIFO 变为空，然后停下来等待新增的数据。

在单接收模式，半双工（BIDIMODE=1，BIDIOE=0），或只发送（BIDIMODE=0，RXONLY=1）模式下，只要 SPI 和只接收模式被使能，主机会立即开始接收序列。主机连续提供时钟信号，直到主机关闭 SPI 或者关闭只接收模式之前都不会停下来。这时主机会连续接收数据。

虽然主机可以以连续模式提供所有交互（SCK 信号是连续的），但它必须考虑从机在任何时候处理数据流及其内容的能力。必要时，主机必须减慢通信速度，并提供较慢的时钟或具有足够延迟的单独帧或数据会话。请注意，在 SPI 模式下，主设备或从设备没有下溢错误信号，并且来自从设备的数据始终由主设备进行处理。

数据帧开始后，从机无法控制或延迟数据序列。出于这个原因，从机必须在开始传输之前准备好数据，并总是一直保持有数据待传（存在 TXFIFO 中）。主机必须在每个序列之间给从机保留足够的时间以准备数据。如果可能的话，序列中的字节的数量应得到限制，以便从机完成自动的数据处理（通过使用 FIFO）。主机必须提供额外的时间给从机处理数据内容。

在多从机并行的系统中，每个序列应该由 NSS 脉冲来分隔，以将每个序列对应到不同的从机。在单从机系统中通过 NSS 控制从机就显得没那么有必要了，但这里有个脉冲还是更好，这可以令从机和每个数据序列完成同步。NSS 既可以由软件管理也可以由硬件管理（见“20.3.4 从器件选择（NSS）引脚管理”）。

BSY 位被置 1 时，它标志着一个持续的传输。这一点，结合 FTLVL[1:0]位，可用于检查传输是否完成。在系统进入停机模式前这是很有必要的，过早的进入停机模式可能导致数据破坏。判断 BSY 位的另外一个目的是可以用来管理 NSS 信号。当 RXNE 标志被置 1，它意味着对当前的传输结束了。即最后一位刚刚采样完毕，以及完整的数据帧存储在 RXFIFO 中。

20.3.8.3 数据组包

当数据帧的大小适合一个字节（小于或等于 8 位），并且对 SPI_DR 寄存器执行任何 16 位的读写访问时，数据会自动组包在一起。在这种情况下，可以并行处理双数据。SPI 先操作低 8 位，然后操作高 8 位。图 20-9 提供了组包方式顺序处理数据的一个例子。在一次对 SPI_DR 的 16 位写访问后，就会有 2 个字节的数据被发送出去。如果 RXFIFO 的阈值设置为 16 位（FRXTH=0），该序列则只会生成一个接收 RXNE 事件，而不是两个。针对这种单个的 RXNE 事件的响应，接收器必须对 SPI_DR 寄存器进行一次 16 位的读访问才能够把数据全都取到。RXFIFO 的阈值和跟进的数据访问的位宽必须保持一致，否则就会丢数据。

字节数为奇数的数据帧传输时，会出现特定的问题。在发送端，可以用 8 位方式写 SPI_DR 将最后一个字节发出来。在接收端必须改变 RXFIFO 门限，以便在接收奇数字节的数据帧的最后字节时能够产生 RXNE 事件。

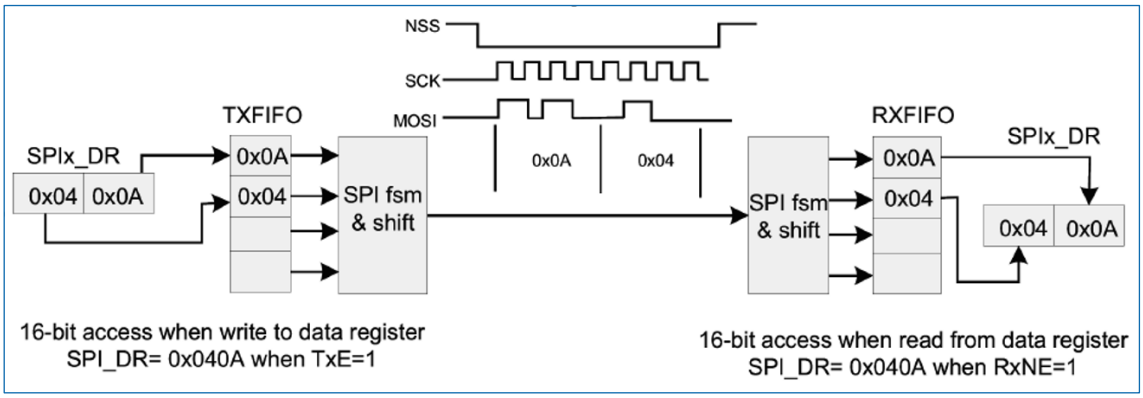

图 20-9 发送和接收 FIFO 中的数据组包

20.3.9 禁止 SPI 的步骤

当禁止 SPI 时，必须按照本节中介绍的禁止步骤进行操作。在外设时钟停止，系统进入低功耗模式前禁止 SPI 是十分重要的。否则会损坏正在进行的交互。在某些模式下，禁止步骤是停止所进行的连续通信的唯一方式。

处于全双工或只发送模式下的主器件一旦停止提供待发送的数据，即可结束任何交互。在这种情况下，时钟在完成最后一个数据传输后停止。

当处理奇数个数据帧时，必须特别注意组包模式，以防止一些虚假字节交换（参见数据组包部分），在这些模式下禁止 SPI 之前，用户必须遵循标准的禁止过程。当一个帧事务正在进行或下一个数据帧存储在 TXFIFO 中时，若 SPI 在主发送机被禁止，SPI 行为是无法保证的。

当主机处于任意接收模式时，停止连续时钟的唯一方法就是设置 SPE=0 以禁用外围设备。此操作必须发生在最后一个数据帧传输的特定的时间窗口内，即，从采样该数据帧第一个比特开始，到传输该数据帧的最后一个比特数据之前。这是为了确保如期接收完整数目的数据帧，且避免在读取最后一个有效数据帧后，读取额外的虚假数据。在该模式下禁止 SPI 必须遵循特定的步骤。

当 SPI 被禁止时，已接收但还未被读取的数据仍然存储在 RXFIFO 中，并且必须在下一次 SPI 被启用时进行处理，然后再开始一个新的序列。为了避免在禁止 SPI 时，还有未被读取的数据，需通过以下方式，确保 RXFIFO 在禁止 SPI 时为空：

- 通过使用正确的禁止 SPI 步骤，

或者，

- 通过特定的外设复位寄存器中的控制位产生软件复位来初始化 SPI 的所有寄存器（详见“6.3.4 APB 外设复位寄存器 2（RCC_APBRSTR2）”）。

SPI 的标准禁止步骤是通过轮询 BSY 状态以及 TXE 标志来检查发送会话是否完全结束。还可以在必须识别正在处理的传输是否结束的特定情况下完成这种检查，例如：

- 当 NSS 信号由任意 GPIO 切换管理且主器件必须为从器件提供 NSS 脉冲结束时。

正确的禁止步骤如下（使用只接收模式时除外）：

1. 等待 RXNE=1，以接收最后的数据。
2. 等待 TXE=1，然后等待至 BSY=0，再关闭 SPI。
3. 读取接收的数据。

说明：在不连续通信期间，在对 SPI_DR 寄存器执行写操作与 BSY 位置 1 之间有 2 个 APB 时钟周期的延迟。因此，写入最后的数据后，必须先等待 TXE 位置 1，然后等待 BSY 位清零。

某些只接收模式的正确禁止步骤如下：

1. 当最后一个数据帧正在处理时，通过在特定时间窗口内禁止 SPI（SPE=0）来中断接收流。
2. 等待至 BSY=0（最后一个数据帧已处理完）。
3. 读取接收的数据。

说明：要停止连续的接收序列，必须在接收最后一个数据帧时遵循特定的时间窗口。该时间窗口在第一个位采样时开始，在最后一个位的传输开始前结束。

20.3.10 SPI 状态标志

应用可通过三种状态标志监视 SPI 总线的状态。

发送缓冲区为空（TXE）

当传输 TXFIFO 有足够的空间存储要发送的数据时，设置 TXE 标志。

TXE 标志连接到 TXFIFO 深度。标志置 1 并保持直到 TXFIFO 深度低于或等于 FIFO 深度的 1/2。如果 SPI_CR2 寄存器中的 TXEIE 位置 1，当 TXE 置 1 时就会生成一个发送中断。当 TXFIFO 电平大于 1/2 时，TXE 位自动清除。

接收缓冲区非空（RXNE）

RXNE 标志的设置取决于 SPI_CR2 寄存器中的 FRXTH 位值：

- 如果 FRXTH 位被置 1，当 RXFIFO 深度达到 1/4，RXNE 位将会置 1 并一直保持。
- 如果 FRXTH 位置 0，当 RXFIFO 深度达到 1/2，RXNE 位将会置 1 并保持。

如果 SPI_CR2 寄存器中 RXNEIE 位置 1，当 RXNE 置 1 时将会生成接收中断。

忙标志（BSY）

BSY 标志由硬件置 1 和清零（软件写该位不生效）。

当 BSY 置 1 时，表示 SPI 上正在进行数据传输（SPI 总线繁忙）。在主模式下的双向通信接收模式（MSTR=1 且 BIDIMODE=1 且 BIDIOE=0）有一个例外情况，BSY 标志在接收过程中始终保持为 0。

在某些模式下，可使用 BSY 标志来检测传输是否结束，从而避免在进入低功耗模式前禁止 SPI 外设时钟时或通过软件管理 NSS 脉冲结束时破坏最后一次传输。

BSY 标志还可用于避免在多主模式系统中发生写冲突。

在以下任意一种条件下，BSY 标志将清零：

- 正确禁止 SPI 时。
- 在主模式下检测到故障时（MODF 位置 1）。
- 在主模式下，完成了数据发送并且不准备发送任何新数据时。
- 在从模式下，BSY 标志在各传输之间的至少一个 SPI 时钟周期内置为“0”时。

注意：建议始终使用 TXE 和 RXNE 标志（而非 BSY 标志）来处理数据发送或接收操作。

20.3.11 SPI 错误标志

如果以下其中一个错误标志置 1 且已通过将 ERRIE 位置 1 使能了中断，则将生成 SPI 中断。

上溢标志（OVR）

当主机或从机接收到数据时，如果 RXFIFO 没有足够空间存储接收到的数据时就会发生溢出。这种情况通常发生在软件没有足够时间去读出以前接收的数据（存储在 RXFIFO）或数据存储空间有限时，如当启用 CRC 并处于仅接收模式时 RXFIFO 不可用，这种情况下应将接收缓冲区限制到一个数据帧缓冲。

当一个上溢事件产生，新接收到的数据将会被丢弃，不会覆盖 RXFIFO 中以前的值，随后传输的所有数据都将会被丢弃。对 SPI_DR 寄存器的读访问和对 SPI_SR 寄存器的读访问将会清除 OVR 位。

模式故障（MODF）

当主器件的内部 NSS 信号（NSS 硬件模式下为 NSS 引脚，NSS 软件模式下为 SSI 位）被拉低时，将发生模式故障。这会自动将 MODF 位置 1。主模式故障会在以下几方面影响 SPI 接口：

- 如果 ERRIE 位置 1，MODF 位将置 1，并生成 SPI 中断。
- SPE 位清零。这将关闭器件的所有输出，并禁止 SPI 接口。
- MSTR 位清零，从而强制器件进入从模式。

使用以下软件序列将 MODF 位清零：

1. 在 MODF 位置 1 时，对 SPI_SR 寄存器执行读或写访问。
2. 对 SPI_CR1 寄存器执行写操作。

为避免包含多个 MCU 的系统中发生多从模式冲突，必须在 MODF 位清零序列期间将 NSS 引脚拉高。在该清零序列后，可以将 SPE 和 MSTR 位恢复到原始状态。安全起见，硬件不允许在 MODF 位置 1 时将 SPE 和 MSTR 位置 1。在从器件中，MODF 位不可置 1，但由前一次多主模式冲突引起时除外。

CRC 错误（CRCERR）

当 SPI_CR1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。当 CRCEN 位置 1，如果移位寄存器中接收的值与 SPI_RXCRC 的值不匹配，SPI_SR 寄存器中的 CRCERR 标志将置 1。该标志由软件清零。

TI 模式帧格式错误（FRE）

如果 SPI 在从模式下工作，并配置为符合 TI 模式协议，则在通信进行期间出现 NSS 脉冲时，将检测到 TI 模式帧格式错误。出现此错误时，SPI_SR 寄存器中的 FRE 标志将置 1。

发生错误时不会禁止 SPI，但会忽略 NSS 脉冲，并且 SPI 会等待下一个 NSS 脉冲，然后再开始新的传输。由于错误检测可能导致丢失两个数据字节，因此数据可能会损坏。

读取 SPI_SR 寄存器时，将清零 FRE 标志。如果 ERRIE 位置 1，则检测到 NSS 错误时将生成中断。在这种情况下，由于无法保证数据的一致性，应禁止 SPI，并在重新使能从 SPI 后，由主器件重新发起通信。

20.3.12 NSS 脉冲模式

该模式由 SPI_CR2 寄存器中的 NSSP 位激活，仅当 SPI 接口配置为 Motorola SPI master（FRF=0）并在第一个边沿捕获（SPI_CR1.CPHA =0，CPOL 设置被忽略）时才生效。当激活时，当 NSS 保持高电平至少持续一个时钟周期时，在两个连续的数据帧传输之间产生一个 NSS 脉冲。这种模式允许从机锁存数据。NSSP 脉冲模式是为单主从对应用而设计的。

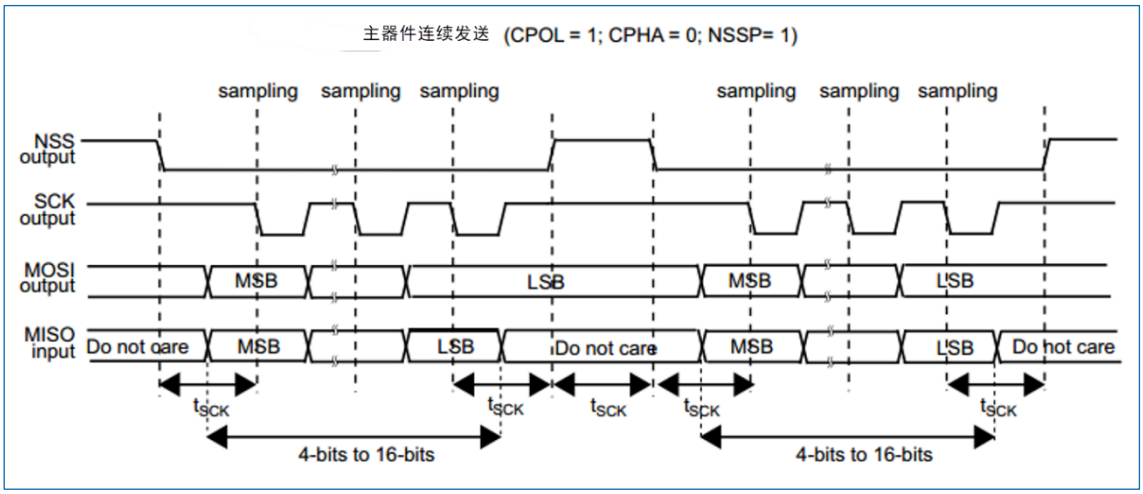

图 20-10 SPI Motorola 主模式 NSS 脉冲生成

注意：当 CPOL=0 时也会遇到类似的行为。在这种情况下，采样边是上升的，NSS 断言和去断言指的就是这个采样边。

20.3.13 TI 模式

主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPI_CR2 寄存器的 FRF 位来配置 SPI，以兼容此协议。

时钟极性和相位都强制遵循 TI 协议，和 SPI_CR1 中的设置无关。NSS 管理也特定于 TI 协议，在这种情况下，无法通过 SPI_CR1 和 SPI_CR2 寄存器（SSM、SSI 和 SSOE）来对 NSS 管理进行配置。

在从模式下，SPI 波特率预分频器用于控制在当前传输完成时 MISO 引脚切换为高阻态的时刻（请参见图 20-11）。可以使用任意波特率，因此可以非常灵活地确定此时刻。但是，波特率通常设置为外部主时钟波特率。MISO 信号变为高阻态的延时（$t_{release}$）取决于内部重新同步以及通过 SPI_CR1 寄存器的 BR[2:0]位设置的波特率值。具体公式如下：

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

如果从器件在数据帧传输期间检测到错位的 NSS 脉冲，TI FRE 标志将置 1。

如果数据大小为 4 位或 5 位，则在全双工模式或只发送模式下，主机使用的协议在 LSB 之后再增加一个虚拟数据位。TI NSS 脉冲是在这个虚拟位时钟周期以上产生的，而不是在每个周期的 LSB。

此特性不适用于 Motorola SPI 通信（FRF 位为 0）。

图 20-11 给出了选择 TI 模式时的 SPI 通信波形。

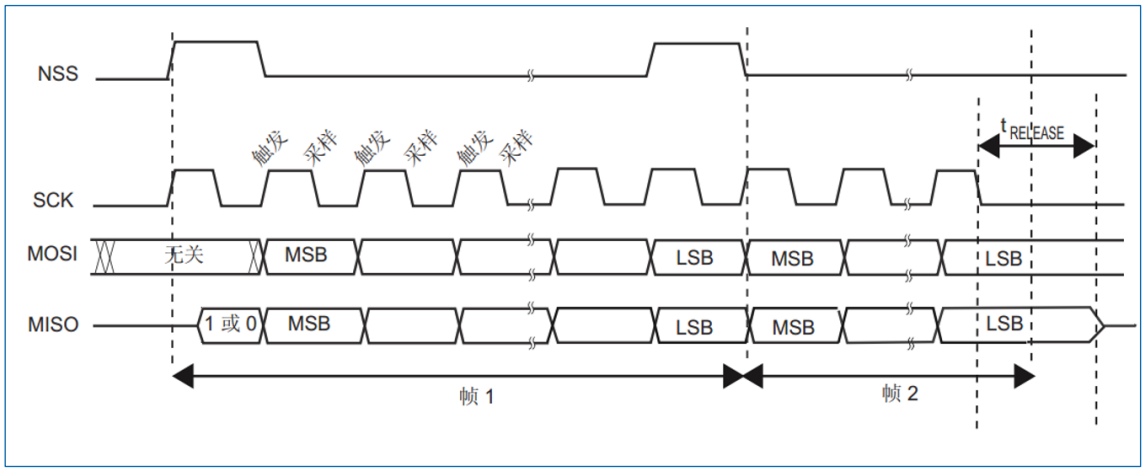

图 20-11 TI 模式传输

20.3.14 CRC 计算

为检查发送数据和接收数据的可靠性，使用两个独立的 CRC 计算器（作用于发送和接收数据流）。SPI 提供 CRC8 或 CRC16 计算，具体取决于通过 CRCL 位选择的数据格式。对于其他长度的数据帧，没有 CRC 可用。

20.3.14.1 CRC 原理

在使能 SPI（SPE=1）前，通过将 SPI_CR1 寄存器中的 CRCEN 位置 1 来使能 CRC 计算。使用值为奇数的可编程多项式对每个位计算 CRC 值。在由 SPI_CR1 寄存器中的 CPHA 位和 CPOL 位定义的采样时钟边沿进行计算。所计算的 CRC 值在数据块末尾自动进行校验，以及针对由 CPU 管理的传输进行校验。当检测到所接收数据内部计算的 CRC 与发送器发送的 CRC 不匹配时，CRCERR 标志将置 1 以指示数据损坏错误。CRC 计算的正确处理步骤取决于 SPI 配置和所选的传输管理。

说明：多项式值只应为奇数。不支持任何偶数值。

20.3.14.2 CPU 管理的 CRC 传输

通信开始后将一直持续到发送或接收 SPI_DR 寄存器中的最后一个数据帧时。之后，SPI_CR1 寄存器中的 CRCNEXT 位必须置 1，以指示当前处理的数据帧传输后将处理 CRC 帧传输。CRCNEXT 位必须在最后一个数据帧传输结束前置 1。在 CRC 传输期间，CRC 计算将冻结。

与任何其他数据帧一样，接收的 CRC 存储在 RXFIFO 中，这就是为什么在 CRC 模式下，接收缓冲区必须被认为是一个用于一次只接收一个数据帧的单独的 16 位缓冲区。

CRC 格式的事务在数据序列的末尾通常需要多一个数据帧通信，然而，当设置一个由 16 位 CRC 检查的 8 位数据帧时，需要另外两个帧来发送完整的 CRC 数据。

接收最后一个 CRC 数据后，将执行自动校验，将接收的值与 SPI_RXCRC 寄存器中的值进行比较。软件必须校验 SPI_SR 寄存器中的 CRCERR 标志，以确定数据传输是否损坏。软件通过向 CRCERR 标志写入“0”来将其清零。

接收 CRC 后，CRC 值存储到 RXFIFO 中，且必须在 SPI_DR 寄存器中进行读取，以将 RXNE 标志清零。

20.3.14.3 复位 SPI_TXCRC 和 SPI_RXCRC 值

当使能 CRC 计算时，SPI_TXCRC 和 SPI_RXCRC 值自动清零。

要将 CRC 清零，其操作步骤如下：

1. 禁止 SPI。

2. 将 CRCEN 位清零。
3. 使能 CRCEN 位。
4. 使能 SPI。

说明：

- *当 SPI 处于从模式时，CRC 计算单元对 SCK 从输入时钟是敏感的，只要 CRCEN 位被设置，这是无论 SPE 位的值是什么情况。为了避免错误的 CRC 计算，软件必须在时钟稳定(处于稳定状态)时才启用 CRC 计算。*
- *当 SPI 接口被配置为从模式时，一旦 CRCNEXT 信号被释放，在 CRC 阶段的事务期间，NSS 内部信号需要保持低电平。这就是为什么 CRC 计算不能在 NSS 脉冲模式下使用，而 NSS 硬件模式应该在从机正常应用。*

20.4 SPI 中断

在 SPI 通信过程中，中断可由以下事件产生：

- 发送缓冲区准备就绪，可以装载数据。
- 接收缓冲区中接收了数据。
- 主模式故障
- 上溢错误
- TI 帧格式错误
- CRC 校验错误

中断可分别进行使能和禁止。

表 20-2 SPI 中断请求

中断事件	事件标志	使能控制位
发送缓冲区准备就绪，可以装载数据	TXE	TXEIE
接收缓冲区中接收了数据	RXNE	RXNEIE
主模式故障	MODF	ERRIE
上溢错误	OVR	
CRC 错误	CRCERR	
TI 帧格式错误	FRE	

20.5 SPI 寄存器

基地址：0x4001 3000

空间大小：0x400

20.5.1 SPI 控制寄存器 1（SPI_CR1）

偏移地址：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMODE	BIDIOE	CRCEN	CRCNEXT	CRCL	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw	rw

位 31:16	Res：保留 必须保持复位值。
位 15	BIDIMODE：双向数据模式使能（Bidirectional data mode enable） • 0：选择双线的单向数据模式 • 1：选择单线双向数据模式
位 14	BIDIOE：双向模式输出使能（Output enable in bidirectional mode） 该位和 BIDIMODE 位一起决定在“单线双向”模式下数据的输出方向。 • 0：输出禁止（只收模式） • 1：输出使能（只发模式）
位 13	CRCEN：硬件 CRC 计算使能（Hardware CRC calculation enable） • 0：CRC 计算禁用 • 1：CRC 计算使能
位 12	CRCNEXT：下一个发送 CRC（Transmit CRC next） • 0：下一个发送的值来自发送缓冲区。 • 1：下一个发送的值来自发送 CRC 寄存器。
位 11	CRCL：CRC 长度（CRC length） 该位由软件设置和清零，用于选择 CRC 长度。 • 0：8 位 CRC 长度 • 1：16 位 CRC 长度
位 10	RXONLY：只接收（Receive only mode enabled） 该位和 BIDIMODE 位一起决定在“双线单向”模式下数据的输出方向。在多个从设备的配置中，在未被访问的从设备上该位被置 1，使得只有被访问的从设备有输出，从而不会造成数据线上数据冲突。 • 0：全双工（发送和接收） • 1：输出禁止（只收模式）
位 9	SSM：软件从机管理（Software slave management） 当 SSM 被置位时，NSS 引脚上的电平由 SSI 位的值决定。 • 0：禁止软件从设备管理 • 1：启用软件从设备管理
位 8	SSI：内部从设备选择（Internal slave select） 该位只有当 SSM 位为 1 的时候才有效。它决定了 NSS 上的电平，在 NSS 引脚上的 I/O

	操作无效。
位 7	LSBFIRST：帧格式（Frame format） • 0：先发送 MSB • 1：先发送 LSB
位 6	SPE：SPI 使能（SPI enable） • 0：禁止 SPI 设备 • 1：开启 SPI 设备
位 5:3	BR[2:0]：波特率控制（Baud rate control） • 000：$f_{PCLK}/2$ • 001：$f_{PCLK}/4$ • 010：$f_{PCLK}/8$ • 011：$f_{PCLK}/16$ • 100：$f_{PCLK}/32$ • 101：$f_{PCLK}/64$ • 110：$f_{PCLK}/128$ • 111：$f_{PCLK}/256$
位 2	MSTR：主设备选择（Master selection） • 0：配置为从设备 • 1：配置为主设备
位 1	CPOL：时钟极性（Clock polarity） • 0：空闲状态时，SCK 保持低电平。 • 1：空闲状态时，SCK 保持高电平。
位 0	CPHA：时钟相位（Clock phase） • 0：第一个时钟沿对准第一位数据。 • 1：第二和时钟沿对准第一位数据。

20.5.2 SPI 控制寄存器 2（SPI_CR2）

偏移地址：0x04

复位值：0x0000 0700

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			FRXTH	DS[3:0]				TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	Res	
			rw	rw				rw	rw	rw	rw	rw	rw		

位 31:13	Res：保留 必须保持复位值。

位 12	FRXTH：FIFO 接收门限（FIFO reception threshold） 该位用来设置触发 RXNE 事件时的 RXFIFO 的阈值。 • 0：如果 FIFO 的存储水平大于或等于 1/2（16 位），产生 RXNE 事件。 • 1：如果 FIFO 的存储水平大于或等于 1/4（8 位），产生 RXNE 事件。
位 11:8	DS[3:0]：数据位宽（Data size） 该位域配置 SPI 传输数据的位宽： • 0000：不使用 • 0001：不使用 • 0010：不使用 • 0011：4 位 • 0100：5 位 • 0101：6 位 • 0110：7 位 • 0111：8 位（默认） • 1000：9 位 • 1001：10 位 • 1010：11 位 • 1011：12 位 • 1100：13 位 • 1101：14 位 • 1110：15 位 • 1111：16 位 如果软件试图写一个“不使用”的值，它们会被强制赋值“0111”（8 位）。
位 7	TXEIE：TX 缓冲器空中断使能（TX buffer empty interrupt enable） • 0：TXE 中断屏蔽 • 1：TXE 中断没有被屏蔽。用于在 TXE 标志置 1 的时候产生一个中断请求。
位 6	RXNEIE：RX 缓冲区非空中断使能（RX buffer not empty interrupt enable） • 0：RXNE 中断屏蔽 • 1：RXNE 中断没有被屏蔽。用于在 RXNE 标志置 1 的时候产生一个中断请求。
位 5	ERRIE：错误中断使能（Error interrupt enable） 该位控制在出现错误事件（CRCERR，OVR，SPI 模式中的 MODF，TI 模式中的 FRE 和 UDR）时是否产生中断。 • 0：错误中断屏蔽 • 1：错误中断使能
位 4	FRF：帧格式（Frame format） • 0：SPI Motorola 模式 • 1：SPI TI 模式

位 3	NSSP：NSS 脉冲管理（NSS pulse management） 该位仅在主模式下使用。它允许 SPI 在连续传输时，两个数据传输之间产生一个 NSS 脉冲。在单个数据传输的情况下，它会在传输结束后将 NSS 引脚强制为高电平。在 CPHA=1 或 FRF=1 的时候，该位没有意义。 • 0：没有 NSS 脉冲 • 1：产生 NSS 脉冲
位 2	SSOE：SS 输出使能（SS output enable） • 0：在主模式下 SS 输出被禁用，SPI 接口可以工作在多主机的配置下。 • 1：SPI 接口启用的同时在主模式下启用 SS 输出。SPI 接口不能在多主环境下工作。
位 1:0	Res：保留 必须保持复位值。

20.5.3 SPI 状态寄存器（SPI_SR）

偏移地址：0x08

复位值：0x0000 0002

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			FTLVL[1:0]		FRLVL[1:0]		FRE	BSY	OVR	MODF	CRCERR	Res		TXE	RXNE
			r		r		r	r	r	r	rc_w0			r	r

位 31:13	Res：保留 必须保持复位值。
位 12:11	FTLVL[1:0]：FIFO 发送存储水平（FIFO Transmission Level） 该位由硬件设置或清零。 • 00：FIFO 空 • 01：1/4 FIFO • 10：1/2 FIFO • 11：FIFO 满（当 FIFO 门限大于 1/2 时认为是满）
位 10:9	FRLVL[1:0]：FIFO 接收存储水平（FIFO reception level） 该位由硬件设置或清零。 • 00：FIFO 空 • 01：1/4 FIFO • 10：1/2 FIFO • 11：FIFO 满
位 8	FRE：TI 帧格式错误（Frame format error） • 0：未发生帧格式错误

	• 1：发生了一个帧格式错误
位 7	BSY：忙标志（Busy flag） • 0：SPI 不忙 • 1：SPI 通信忙或发送缓冲区不为空 该位由硬件设置或清零。
位 6	OVR：溢出标志（Overrun flag） • 0：没有发生溢出 • 1：发生溢出 该位由硬件置位，由软件序列复位。
位 5	MODF：模式故障（Mode fault） • 0：无模式故障发生 • 1：模式故障发生 该位由硬件置位，由软件序列复位。
位 4	CRCERR：CRC 错误标志（CRC error flag） • 0：收到的 CRC 值和 SPI_RXCRCR 的值是匹配的。 • 1：收到的 CRC 值和 SPI_RXCRCR 值不匹配。 该位由硬件置位，由软件清零。
位 3:2	Res：保留 必须保持复位值。
位 1	TXE：发送缓冲区为空标志（Transmit buffer empty） • 0：TX 缓冲区非空 • 1：TX 缓冲器空
位 0	RXNE：接收缓冲区非空标志（Receive buffer not empty） • 0：RX 缓冲区空 • 1：RX 缓冲非空

20.5.4 SPI 数据寄存器（SPI_DR）

偏移地址：0x0C

复位值：0x0000 0000

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

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

位 31:16	Res：保留

	必须保持复位值。
位 15:0	DATA[15:0]：数据信息（Data message） 待发送或者已经收到的数据，数据寄存器作为 RX 和 TX FIFOs 的接口。当读取数据寄存器时，RXFIFO 会被访问，而写入数据寄存器则会访问 TXFIFO。

20.5.5 SPI 的 CRC 多项式寄存器（SPI_CRCPR）

偏移地址：0x10

复位值：0x0000 0007

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	CRCPOLY[15:0]：CRC 多项式寄存器（CRC polynomial register） 该寄存器包含 CRC 计算多项式。根据需要，可以配置成另一个多项式。

20.5.6 SPI 接收 CRC 寄存器（SPI_RXCRCR）

偏移地址：0x14

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	RXCRC[15:0]：RXCRC 寄存器（RX CRC register） 当启用 CRC 计算功能，RXCRC[15:0]位包含根据收到的字节计算出来的 CRC 值。当 SPI_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 • 当数据帧格式被设置为 8 位（SPI_CR1 的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 • 当数据帧格式为 16 位（SPI_CR1 的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

20.5.7 SPI 发送 CRC 寄存器（SPI_TXCRCR）

偏移地址：0x18

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	TXCRC[15:0]：TXCRC 寄存器（TX CRC register） 当启用 CRC 计算功能，TXCRC[7:0]位包含根据发送的字节计算出来的 CRC 值。当 SPI_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 • 当数据帧格式被设置为 8 位（SPI_CR1 中的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 • 当数据帧格式为 16 位（SPI_CR1 的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

21 设备电子签名（UID）

设备电子签名存储在 Flash 模块的系统存储区中，可以使用 JTAG/SWD 或 CPU 对其进行读取。

UID 包含出厂前编程的标识数据，这些标识数据允许用户固件或其他外部设备将其接口与 HK32F0301MxxxxC 的特性自动匹配。

21.1 唯一设备 ID 寄存器（96 位）

基地址：0x1FFF F128

空间大小：0x0C

唯一设备标识符适用于以下应用场景。

- 用作序列号。
- 在对内部 Flash 进行编程前将唯一 ID 与软件加密原语和协议结合使用时，用作安全密钥以提高 Flash 中代码的安全性。
- 激活安全自举过程等。

96 位的唯一设备标识符提供了一个对于任何设备和任何上下文都唯一的参考号码。用户永远不能改变该位域。

96 位的唯一设备标识符也可以以单字节/半字/字等不同方式读取，然后使用自定义算法组合为 96 位。

21.1.1 UID 寄存器 0（U_ID0）

偏移地址：0x00

复位值：0xXXXX XXXX

说明：此处 X 表示出厂前编程设置。

U_ID0 为 UID 的低 32 位。

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

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

位 31:0	U_ID[31:0]：芯片唯一 ID 位（Unique ID bits[31:0]）

21.1.2 UID 寄存器 1（U_ID1）

偏移地址：0x04

复位值：0xXXXX XXXX

说明：此处 X 表示出厂前编程设置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID[63:48]															
r															

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

UID[47:32]
r

位 31:0	U_ID[63:32]：芯片唯一 ID 位（Unique ID bits[63:32]）

21.1.3 UID 寄存器 2（U_ID2）

偏移地址：0x08

复位值：0xXXXX XXXX

说明：此处 X 表示出厂前编程设置。

UID 寄存器 2 的值为 UID 的 65 至 96 位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U_ID[95:80]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U_ID[79:64]															
r															

位 31:0	U_ID[95:64]：芯片唯一 ID 位（Unique ID bits[95:64]）

22 调试支持（DBG）

22.1 概述

HK32F0301MxxxxC 器件的内核是 Cortex®-M0，该内核包含用于高级调试功能的硬件扩展。调试扩展允许内核在取指（指令断点）或取访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统，并恢复程序执行。

当调试主机与 HK32F0301MxxxxC MCU 相连并进行调试时，将使用调试功能。

提供一个调试接口：

- 串行接口

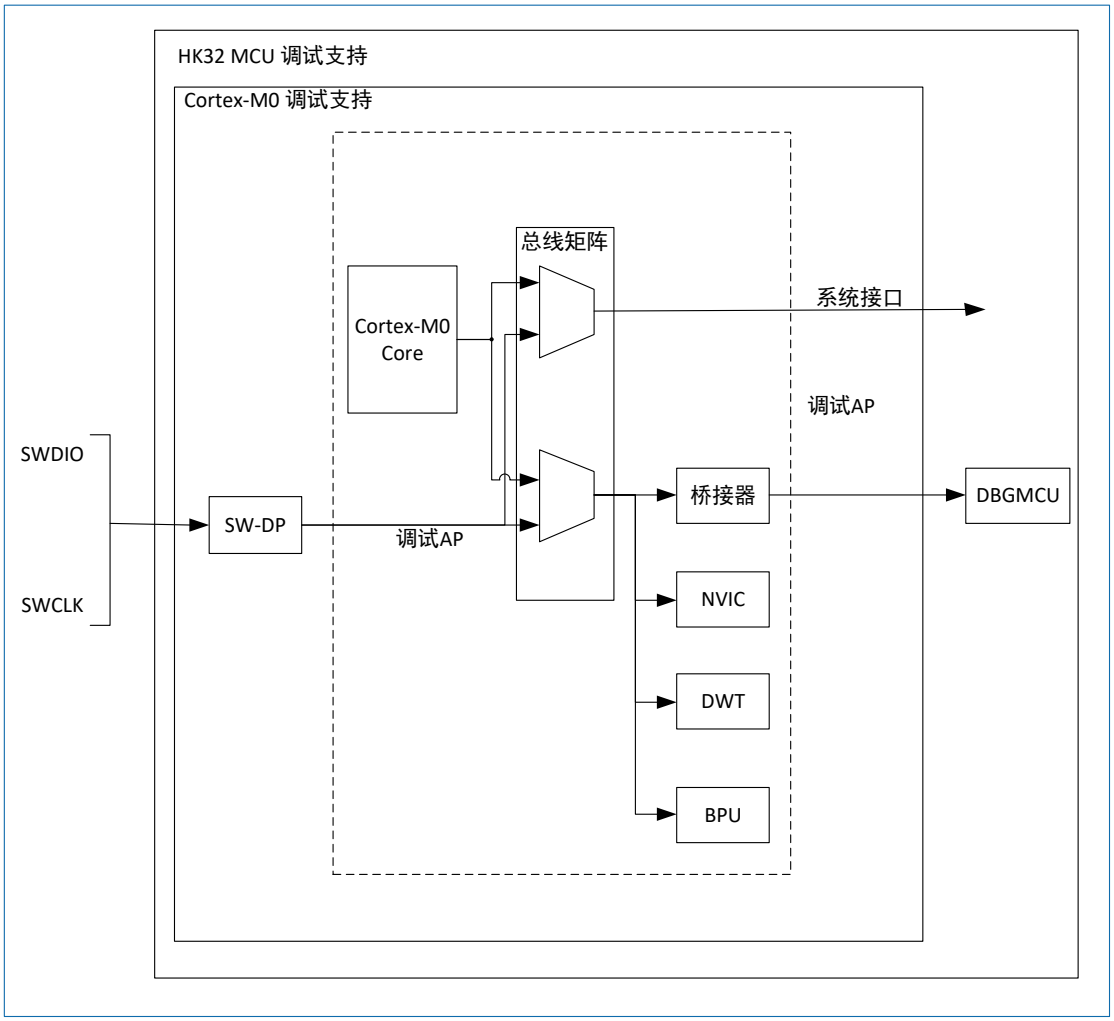

图 22-1 HK32F0301MxxxxC MCU 和 Cortex®-M0 级调试支持框图

Cortex®-M0 内核中内置的调试功能是 ARM® CoreSight 设计套件的一部分。ARM® Cortex®-M0 内核提供集成片上调试支持。它包括：

- SW-DP：串行线
- BPU：断点单元
- DWT：数据观察点触发

它还包括专用于 HK32F0301MxxxxC 的调试功能：

- 灵活调试引脚分配

- MCU 调试盒（支持低功耗模式和对外设时钟的控制等）

说明：有关 ARM® Cortex®-M0 内核支持的调试功能的详细信息，请参见 Cortex®-M0 技术参考手册。

22.2 ARM®参考文档

- Cortex®-M0 技术参考手册（TRM）
- ARM®调试接口 V5
- ARM® CoreSight 设计套件版本 r1p1 技术参考手册

22.3 引脚排列和调试端口引脚

HK32F0301MxxxxC MCU 的不同封装有不同的有效引脚数。

22.3.1 SWD 端口引脚

两个引脚被用作 SW-DP 的输出，作为通用 I/O 的复用功能。所有封装都提供这些引脚。

表 22-1 SW 调试端口引脚

SW-DP 引脚名称	SW 调试端口		引脚分配
	类型	调试分配	
SWDIO	IO	串行线数据输入/输出	PD5
SWCLK	I	串行线时钟	PB5

22.3.2 SW-DP 引脚分配

复位（SYSRESETn 或 PORESETn）后，用于 SW-DP 的引脚将分配为可由调试主机立即使用的专用引脚。

但是 MCU 可以禁止 SWD 端口，进而可释放相关引脚以用作通用 I/O（GPIO）。有关如何禁止 SW-DP 端口引脚的更多详细信息，请参见“8.2.2 I/O 引脚复用功能复用器和映射”。

22.3.3 SWD 引脚上的内部上拉和下拉

用户软件释放 SW I/O 后，GPIO 控制器便会控制这些引脚。GPIO 控制寄存器的复位状态会将 I/O 置于等效的状态：

- SWDIO：输入上拉
- SWCLK：输入下拉

由于内置上拉和下拉电阻，因此无需添加外部电阻。

22.4 SWD 端口

22.4.1 SWD 协议简介

此同步串行协议使用两个引脚：

- SWCLK：从主机到目标的时钟
- SWDIO：双向

利用该协议，可以同时读取和写入两组寄存器组（DPACC 寄存器组和 APACC 寄存器组）。传输数据时，LSB 在前。

对于 SWDIO 双向管理，必须在电路板上对线路进行上拉。这些上拉电阻可在内部配置。无需外部

上拉电阻。

每次在协议中更改 SWDIO 的方向时，都会插入转换时间，此时线路即不受主机驱动也不受目标驱动。默认情况下，此转换时间为一位时间，但也可以通过配置 SWCLK 频率来调整。

22.4.2 SWD 协议序列

每个序列包括三个阶段:

1. 主机发送的数据包请求（8 位）。
2. 目标发送的确认响应（3 位）。
3. 主机或目标发送的数据传输阶段（33 位）。

表 22-2 数据包请求（8 位）

位	名称	说明
0	启动	必须为 1
1	APnDP	● 0：DP 访问 ● 1：AP 访问
2	RnW	● 0：写请求 ● 1：读请求
4:3	A[3:2]	DP 或 AP 寄存器的地址字段
5	奇偶校验	该位表示前面几位（包括启动位）的奇偶校验值
6	停止	0
7	驻留	不受主机驱动。由于存在上拉，因此必须由目标读为 1。

有关 DPACC 和 APACC 寄存器的详细说明，请参见 Cortex®-M0 TRM。

数据包请求后面始终为转换时间（默认 1 位），此时主机和目标都不会驱动线路。

表 22-3 ACK 响应（3 位）

位	名称	说明
0..2	ACK	● 001：FAULT ● 010：WAIT ● 100：OK

仅当发生 READ 事务或者接收到 WAIT 或 FAULT 确认时，ACK 响应后才必须是转换时间。

表 22-4 DATA 传输（33 位）

位	名称	说明
0-31	WDATA 或 RDATA	写入或读取数据
32	奇偶校验	32 个数据位的单奇偶校验

仅当发生 READ 事务时，DATA 传输后才必须是转换时间。

22.4.3 SW-DP 状态机（复位、空闲状态、ID 代码）

SW-DP 的状态机有一个用于标识 SW-DP 的内部 ID 代码。该代码符合 JEP-106 标准。此 ID 代码是默认的 ARM®代码，设置为 0x0BB11477（相当于 Cortex®-M0）。

说明：在目标读取此 ID 代码前，SW-DP 状态机是不工作的。

- 在上电复位后或者线路处于高电平超过 50 个周期后，SW-DP 状态机处于复位状态。
- 如果在复位状态后线路处于低电平至少两个周期，SW-DP 状态机处于空闲状态。
- 复位状态后，该状态机必须首先进入空闲状态，然后对 DP-SW ID CODE 寄存器执行读访问。否则，目标将在另一个事务上发出 FAULT 确认响应。

有关 SW-DP 状态机的更多详细信息，请参见 Cortex®-M0+ TRM 和 CoreSight 设计套件 r1p0 TRM。

22.4.4 DP 和 AP 读/写访问

- 不延迟对 DP 的读访问：可以立即发送目标响应（如果 ACK=OK），也可以延迟发送目标响应（如果 ACK=WAIT）。
- 延迟对 AP 的读访问。这意味着会在下次传输时返回访问结果。如果要执行的下次访问不是 AP 访问，则必须读取 DP-RDBUFF 寄存器来获取结果。每次进行 AP 读访问或 RDBUFF 读请求时都会更新 DP-CTRL/STAT 寄存器的 READOK 标志，以便了解 AP 读访问是否成功。
- SW-DP 有写缓冲区（用于 DP 或 AP 写入），这样即使在其他操作仍未完成时，也可以接受写入操作。如果写缓冲区已满，则目标确认响应为 WAIT。但 IDCODE 读取、CTRL/STAT 读取或 ABORT 写入除外，这几项操作在写缓冲区已满时也会被接受。
- 由于存在异步时钟域 SWCLK 和 HCLK，因此写操作后（奇偶校验位后）还需要两个额外的 SWCLK 周期，以使写入操作在内部生效。应在将线路驱动为低电平时（空闲状态）应用这些周期。在写 CTRL/STAT 寄存器以提出一个上电请求时，这一点特别重要。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。

22.4.5 SW-DP 寄存器描述

当 APnDP=0 时能够访问这些寄存器。

表 22-5 SW-DP 寄存器

A[3:2]	R/W	SELECT 寄存器的 CTRLSEL 位	寄存器	说明
00	读取		IDCODE	制造商代码设置为 Cortex®-M0 的默认 ARM®代码。 0x0BB11477（标识 SW-DP）
00	写		ABORT	
01	读/写	0	DP-CTRL/STAT	目的： ● 请求系统或调试上电 ● 配置 AP 访问的传输操作 ● 控制比较和验证操作 ● 读取一些状态标志（上溢和上电确认）
01	读/写	1	WIRE CONTROL	用于配置物理串行端口协议（如转换时间的持续时间）。
10	读取		READ RESEND	允许从已损坏的调试软件传输中恢复读取数据，无需重复执行原始 AP 传输。
10	写		SELECT	用于选择当前访问端口和活动的 4 字寄存器窗口。

A[3:2]	R/W	SELECT 寄存器的 CTRLSEL 位	寄存器	说明
11	读/写		READ BUFFER	由于已发出 AP 访问，因此该读缓冲区非常有用（在执行下个 AP 事务时提供读取 AP 请求的结果）。 此读取缓冲区捕获 AP 中的数据，显示为前一次读取的结果，无需启动新操作。

22.4.6 SW-AP 寄存器描述

当 APnDP=1 时能够访问这些寄存器。

有多个 AP 寄存器，这些寄存器按以下组合进行寻址：

- 移位值 A[3:2]
- DP SELECT 寄存器的当前值

表 22-6 32 位调试端口寄存器，通过移位值 A[3:2]进行寻址

地址	A[3:2]值	说明
0x0	00	保留位，必须保持复位值。
0x4	01	DP CTRL/STAT 寄存器，用于： ● 请求系统或调试上电 ● 配置 AP 访问的传输操作 ● 控制比较和验证操作 读取一些状态标志（上溢和上电确认）
0x8	10	DP SELECT 寄存器：用于选择当前访问端口和活动的 4 字寄存器窗口。 ● 位 31:24：APSEL，用于选择当前 AP（select the current AP） ● 位 23:8：保留 ● 位 7:4：APBANKSEL，用于在当前 AP 上选择活动的 4 字寄存器窗口。 ● 位 3:0：保留
0xC	11	DP RDBUFF 寄存器：用于通过调试器在执行一系列操作后获取最后结果（无需请求新的 JTAG-DP 操作）

22.5 内核调试

通过内核调试寄存器调试内核。通过调试访问端口调试访问这些寄存器。它由四个寄存器组成：

表 22-7 内核调试寄存器

寄存器	说明
DHCSR	32 位调试停止控制和状态寄存器： 此寄存器提供有关处理器状态的信息，能够使内核进入调试停止状态并提供处理器步进功能。
DCRSR	17 位调试内核寄存器选择器寄存器： 此寄存器选择需要进行读写操作的处理器寄存器。
DCRDR	32 位调试内核寄存器数据寄存器： 此寄存器保存在寄存器与 DCRSR（选择器）寄存器选择的处理器之间读取和写入的数据。
DEMCR	32 位调试异常和监视控制寄存器： 此寄存器提供向量捕获和调试监视控制。

这些寄存器在系统复位时不复位。它们只能通过上电复位来复位。有关更多详细信息，请参见 Cortex®-M0 TRM。

为了在复位后立即使内核进入调试停止状态，必须：

- 使能调试和异常监视控制寄存器的位 0（VC_CORRESET）
- 使能调试停止控制和状态寄存器的位 0（C_DEBUGEN）

22.6 BPU（断点单元）

Cortex®-M0 BPU 实现提供四个断点寄存器。BPU 是 ARMv7-M（Cortex®-M3 和 Cortex®-M4）中提供的 Flash 补丁和断点（FPB）模块的一部分。

22.6.1 BPU 功能

处理器断点实现了基于 PC 的断点功能。

有关 BPU CoreSight 标识寄存器及其地址和访问类型的更多信息，请参见 ARMv6-M ARM®和 ARM® CoreSight 组件技术参考手册。

22.7 DWT（数据观察点）

Cortex®-M0 DWT 实现提供了两个观察点寄存器组。

22.7.1 DWT 功能

处理器观察点实现了数据地址和基于 PC 的观察点功能（即 PC 采样寄存器），并支持比较器地址掩码，如 ARMv6-M ARM®中所述。

22.7.2 DWT 程序计数器采样寄存器

实现数据观察点单元的处理器还实现了 ARMv6-M 可选 DWT 程序计数器采样寄存器（DWT_PCSR）。此寄存器允许调试程序定期采样 PC，无需停止处理器。这可提供粗略分析。有关更多信息，请参见 ARMv6-M ARM®。

Cortex®-M0 DWT_PCSR 记录通过条件代码的指令以及未通过条件代码的指令。

22.8 MCU 调试组件（DBG）

MCU 调试组件帮助调试器为以下各项提供支持：

- 低功耗模式
- 断点期间的定时器、看门狗和 I2C 的时钟控制

22.8.1 对低功耗模式的调试支持

要进入低功耗模式，必须执行指令 WFI 或 WFE。

MCU 支持多个低功耗模式，这些模式可以禁止 CPU 时钟或降低 CPU 功耗。

内核不允许在调试会话期间关闭 FCLK 或 HCLK。由于调试期间需要使用它们进行调试连接，因此其必须保持激活状态。MCU 集成了特殊方法，允许用户在低功耗模式下调试软件。

为此，调试主机首先必须设置一些调试配置寄存器，以更改低功耗模式行为：

- 在睡眠模式下，FCLK 和 HCLK 仍有效。因此，此模式对标准调试功能没有任何限制。
- 在停机（Stop）模式下，调试程序必须事先将 DBG_STOP 位置 1。这样便可使能内部 RC 振荡器时钟，以在停止模式下为 FCLK 和 HCLK 提供时钟。

22.8.2 对定时器、看门狗和 I2C 的调试支持

断点期间，必须选择定时器和看门狗的计数器的行为方式：

- 在产生断点时，计数器继续计数。例如，当 PWM 控制电机时，通常需要这种方式。
- 在产生断点时，计数器停止计数。用于看门狗时需要这种方式。

对于 I2C，用户可以选择在断点期间阻止 SMBUS 超时。

22.9 DBGMCU 寄存器

基地址：0x4001 5800

空间大小：0x400

22.9.1 MCU 器件 ID 代码（DBGMCU_IDCODE）

偏移地址：0x00

复位值：0x1000 6FFF

说明：该寄存器仅支持读和 32 位访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REV_ID[15:0]															
r															

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

位 31:16	REV_ID[15:0]：版本标识符（Revision identifier） 0x1000：版本编号 1.0
位 15:12	Res：保留 必须保持复位值。
位 11:0	DEV_ID[11:0]：器件标识符（Device identifier） 复位后从 Flash 加载。 该字段指定设备 ID 为 0xFFF。

22.9.2 调试 MCU 配置寄存器（DBGMCU_CR）

偏移地址：0x04

复位值：0x0000 0000

说明：该寄存器仅支持 32 位访问。该寄存器仅能被上电复位（POR），系统复位信号不能将其复位。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res														DBG_STOP	Res
														rw	

位 31:2	Res：保留 必须保持复位值。
位 1	DBG_STOP：调试停机模式（Debug Stop mode） • 0：（FCLK=关闭，HCLK=关闭）在停机模式下，时钟控制器禁止所有时钟（包括 HCLK 和 FCLK）。 当退出停机模式时，时钟配置与 RESET 后的时钟配置相同。因此，软件必须重新编程时钟控制器以使能 PLL 和晶振等。 • 1：（FCLK=开启，HCLK=开启）进入停机模式时，FCLK 和 HCLK 由在停机模式下仍保持激活状态的内部 RC 振荡器提供。 当退出停机模式时，软件必须重新编程时钟控制器以使能 PLL 和晶振等（操作步骤与在 DBG_STOP=0 时相同）。
位 0	Res：保留 必须保持复位值。

22.9.3 调试 MCUAPB 冻结寄存器（DBGMCU_APB_FZ）

偏移地址：0x08

复位值：0x0000 0000

说明：该寄存器仅支持 32 位访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										DBG_I2C_SMBUS_TIMEOUT	Res				
										rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			DBG_IWDG_STOP	DBG_WWDG_STOP	Res						DBG_TIM6_STOP	Res		DBG_TIM2_STOP	DBG_TIM1_STOP
			rw	rw							rw			rw	rw

位 31:22	Res：保留 必须保持复位值。
位 21	DBG_I2C_SMBUS_TIMEOUT：内核停止时，停止 I2C SMBus 超时模式。（SMBUS tiomout mode stopped when core is halted） • 0：行为方式与正常模式下相同。 • 1：冻结 I2C SMBus 超时。
位 20:13	Res：保留 必须保持复位值。
位 12	DBG_IWDG_STOP：内核停止时，停止调试独立看门狗。（Debug independent watchdog stopped when core is halted） • 0：即使内核停止，独立看门狗计数器时钟仍继续工作。 • 1：内核停止时，独立看门狗计数器时钟停止。
位 11	DBG_WWDG_STOP：内核停止时，停止调试窗口看门狗。（Debug window watchdog

	stopped when core is halted） • 0：即使内核停止，窗口看门狗计数器时钟仍继续工作。 • 1：内核停止时，窗口看门狗计数器时钟停止。
位 10:5	Res：保留 必须保持复位值。
位 4	DBG_TIM6_STOP：内核停止时，TIM6 停止。（TIM6 counter stopped when core is halted） • 0：即使内核停止，TIM6 计数器时钟仍继续工作。 • 1：内核停止时，TIM6 计数器时钟停止。
位 3:2	Res：保留 必须保持复位值。
位 1	DBG_TIM2_STOP：内核停止时，TIM2 停止。（TIM2 counter stopped when core is halted） • 0：即使内核停止，仍然馈送 TIM2 计数器时钟。 • 1：内核停止时，TIM2 计数器时钟停止。
位 0	DBG_TIM1_STOP：内核停止时 TIM1 停止。（TIM1 counter stopped when core is halted） • 0：即使内核停止，仍然馈送 TIM1 计数器时钟。 • 1：内核停止时，TIM1 计数器时钟停止。

22.9.4 MCU 型号 ID 代码（DBGMCU_ENGR_IDCODE）

偏移地址：0x30

复位值：0xXXXX 0515

说明：该寄存器仅支持读和 32 位访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEV_ID2[15:0]															
r															

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

位 31:16	DEV_ID2[15:0]：设备具体型号标识符（Device identifier） 复位后从 Flash 加载。参见表 22-8。
位 15:12	Res：保留 必须保持复位值。
位 11:0	ENG_ID[11:0]：芯片标识符（Chip identifier） 复位后从 Flash 加载。 该字段指定芯片 ID 为 0x521。

以下说明 DEV_ID2 和 ENG_ID。

表 22-8 HK32F0301MxxxxC 的 DEV_ID2 和 ENG_ID

器件	DEV_ID2	ENG_ID
HK32F0301MJ4M7C	0x007B	0x521
HK32F0301MD4P7C	0x007C	0x521
HK32F0301MF4P7C	0x007D	0x521
HK32F0301MF4N7C	0x007E	0x521
HK32F0301MF4U7C	0x007F	0x521

23 缩略语与术语

23.1 寄存器描述中的缩略语

缩写	全称	中文描述
r	read-only	只读
w	write-only	只写
rc_w0	read/clear this field by writing ‘0'	可读；可通过对该位域写0 清除。 对该位域写1，该位域值无变化。
rc_w1	read/clear this field by writing ‘1'	可读；可通过对该位域写1 清除。 对该位域写0，该位域值无变化。
rs	read/set	可读写，与rw 有区别，通常设置该位域为1 时启动某种硬件动作；当完成硬件动作后，该位域会被硬件自动清0。
rw	read/write	可读写该位或指定位。

23.2 术语

名称	中文描述
Byte	字节，8 位数据长度。
Half word	半字，16 位的数据或指令长度。
Option byte	选项字节，保存在Flash 中的MCU 配置字节。
Word	字，32 位的数据或指令长度。

23.3 缩略语

缩写	全称	中文描述
AHB	Advanced High-Performance Bus	高级高性能总线
APB	Advanced Peripheral Bus	外围总线
BGR	Bandgap reference	带隙参考
GPIO	General Purpose Input Output	通用输入输出
EXTI	Extended interrupts and events controller	外部中断事件控制器
NVIC	Nested vectored interrupt controller	嵌套中断向量列表控制器
HSI	High-Speed Internal（Clock Signal）	高速内部（时钟信号）
IAP	In-Application Programming	在线应用编程
ICP	In Circuit Programing	在电路编程
LSI	Low-Speed Internal（Clock Signal）	低速内部（时钟信号）

缩写	全称	中文描述
MCU	Microcontroller Unit	微控制单元
OBL	Option Byte Loader	选项字节装载器
SWD	Serial Wire Debug	内核集成的调试口，它是基于SWD 协议的 2 线调试接口。

24 重要提示

航顺芯片和其他航顺商标均为深圳市航顺芯片技术研发有限公司的商标。本文档提及的其他商标或注册商标，由各自的所有人持有。

在未经深圳市航顺芯片技术研发有限公司同意下，不得以任何形式或途径修改本公司产品规格和数据表中的任何部分以及子部份。深圳市航顺芯片技术研发有限公司在以下方面保留权利：修改数据单和/或产品、停产任一产品或者终止服务不做通知；建议顾客获取最新版本的相关信息，在下定订单前进行核实以确保信息的及时性和完整性。所有的产品都依据订单确认时所提供的销售合同条款出售，条款内容包括保修范围、知识产权和责任范围。

深圳市航顺芯片技术研发有限公司保证在销售期间，产品的性能按照本公司的标准保修。公司认为有必要维持此项保修，会使用测试和其他质量控制技术。除了政府强制规定外，其他仪器的测量表没有必要进行特殊测试。

顾客认可本公司的产品的设计、生产的目的不涉及与生命保障相关或者用于其他危险的活动或者环境的其他系统或产品中。出现故障的产品会导致人身伤亡、财产或环境的损伤（统称高危活动）。人为在高危活动中使用本公司产品，本公司据此不作保修，并且不对顾客或者第三方负有责任。

深圳市航顺芯片技术研发有限公司将会提供与现在一样的技术支持、帮助、建议和信息，（全部包括关于购买的电路板或其他应用程序的设计，开发或调试）。特此声明，对于所有的技术支持、可销性或针对特定用途，及在支持技术无误下，电路板和其他应用程序可以操作或运行的，本公司将不作任何有关此类支持技术的担保，并对您在使用这项支持服务不负任何法律责任。