HK32M06x 用户手册

版本：1.7

发布日期：2024-02-23

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 HK32M06x 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 HK32M06x 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
- 芯片选型工程师

版本说明

本文档对应的产品系列为 HK32M06x 系列芯片。

修订记录

版本	日期	修订说明
0.99	2022/09/28	Alpha 版本发布
1.0	2023/07/04	官网首次发布
1.1	2023/08/25	将原文档更名为“HK32M06x 用户手册”，以支持更多 HK32M06x 系列芯片（新增 HK32M063C 和 HK32M066B 系列），一并增加对不同系列芯片的支持的差异描述。主要修改如下： 1. “2.1 系统架构”下增加子章节分别介绍各系列的功能框图。 2. 原“4.1 电源”修改为现在的“4.1 HK32M060 电源”、“4.2 HK32M063C 电源”和“4.3 HK32M066B 电源”分别描述各个系列电源。 3. “15 高级控制定时器（TIM1）”章节增加了新增 HK32M063C 和 HK32M066B 系列中因为 TIM1 连接预驱，故而“CH1~CH3 不支持输入捕获功能”的说明。
1.2	2023/09/15	1. 更新了“3.3 选项字节”中“WDG_SW=1”为硬件看门狗的描述为“WDG_SW=0”。 2. 更新了“10 模拟数字转换（ADC）”章节的描述。 3. 更新了“12.2 OPAMP 功能说明”章节表中的 OPAMP3 所属的部分描述。 4. 更新了“15.3.18 TIM1 刹车和死区寄存器（TIM1_BDTR）”中“BKP”寄存器位的描述，原文 0 和 1 的描述写反了。
1.3	2023/10/13	1. 更新了章节“10.16 ADC 寄存器”中的英文描述格式。 2. 更新了章节“11.4 COMP 寄存器”中寄存器名称的描述。 3. 更新了章节“18.2 IWDG 功能描述”中的 IWDG 框图，预分频器为 3 位，而不是之前的 8 位。
1.4	2023/11/14	1. 更新了“3.2.3 读保护”中关于调试模式的描述。 2. 更新了“15.3.21 TIM1 选项寄存器（TIM1_OR）”中寄存器“VCOMPO1”的描述（原文 0 和 1 的值写反了）。 3. 更新了“图 18 1 IWDG 框图”中的“3 位预分频器”为“8 位预分频器”
1.5	2023/11/20	更新了“3.3.1.2 选项字节擦除”中的描述：不是按字节删除，而是按页删除。

版本	日期	修订说明
1.6	2023/12/20	更新了“15.3.18 TIM1 刹车和死区寄存器（TIM1_BDTR）”中“DTG[7:0]”寄存器位的描述。
1.7	2024/02/23	1. 更新了“3.1 Flash 特性”中的描述：删除了“内置数据缓冲区”。 2. 更新了“12.2.3 ADC 采样 OPAMP 输出”的描述。 3. 更新了“15.3.2 TIM1 控制寄存器 2（TIM1_CR2）”中“CMPR1_EN”的描述：删了“PWM1 或”。

目录

1 简介

本文档为 HK32M06x 系列芯片的用户手册。HK32M06x 系列芯片是由深圳市航顺芯片技术研发有限公司研发的电机驱动专用 MCU 芯片，包括以下子系列：

- HK32M060 系列
  - HK32M060K8U7（QFN32 封装）
  - HK32M060K8T7（LQFP32 封装）
  - HK32M060C8U7（QFN48 封装）
  - HK32M060C8T7（LQFP48 封装）
- HK32M063C 系列
  - HK32M063CK8U7（QFN32 封装）
- HK32M066B 系列
  - HK32M066BC8T7（LQFP48 封装）

用户可以查看《HK32M060 数据手册》、《HK32M063C 数据手册》及《HK32M066B 数据手册》，进一步了解相应系列 MCU 的功能特性，如外设接口、电气特性、引脚封装等。

2 系统和存储器概述

本章介绍了 HK32M06x 的系统架构、内部存储器。

2.1 系统架构

HK32M06x 主要包括以下几个模块：

- 主模块：
  - Cortex®-M0 内核
  - 高性能总线
- 从模块：
  - 内部 SRAM
  - 内部 Flash 存储器
  - AHB 到 APB 的桥，所有的外设都挂在 APB 总线上
  - 连接 GPIO 口、EMACC 等的 AHB 总线

HK32M06x 子系列芯片的功能结构框图有所不同，详见后续章节。

2.1.1 HK32M060 功能框图

以 HK32M060C8T7 为例，HK32M060 功能框图如下所示：

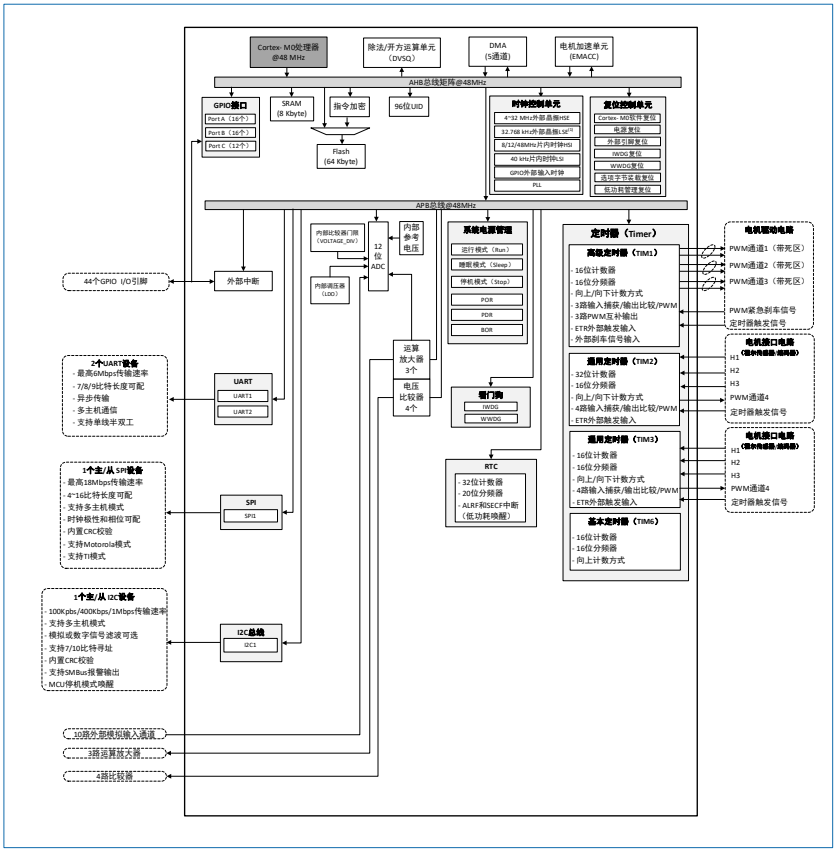

图 2-1 HK32M060 功能框图

图 2-1 的说明：

(1). 仅 48 引脚封装提供外部低速时钟 LSE。

2.1.2 HK32M063C 功能框图

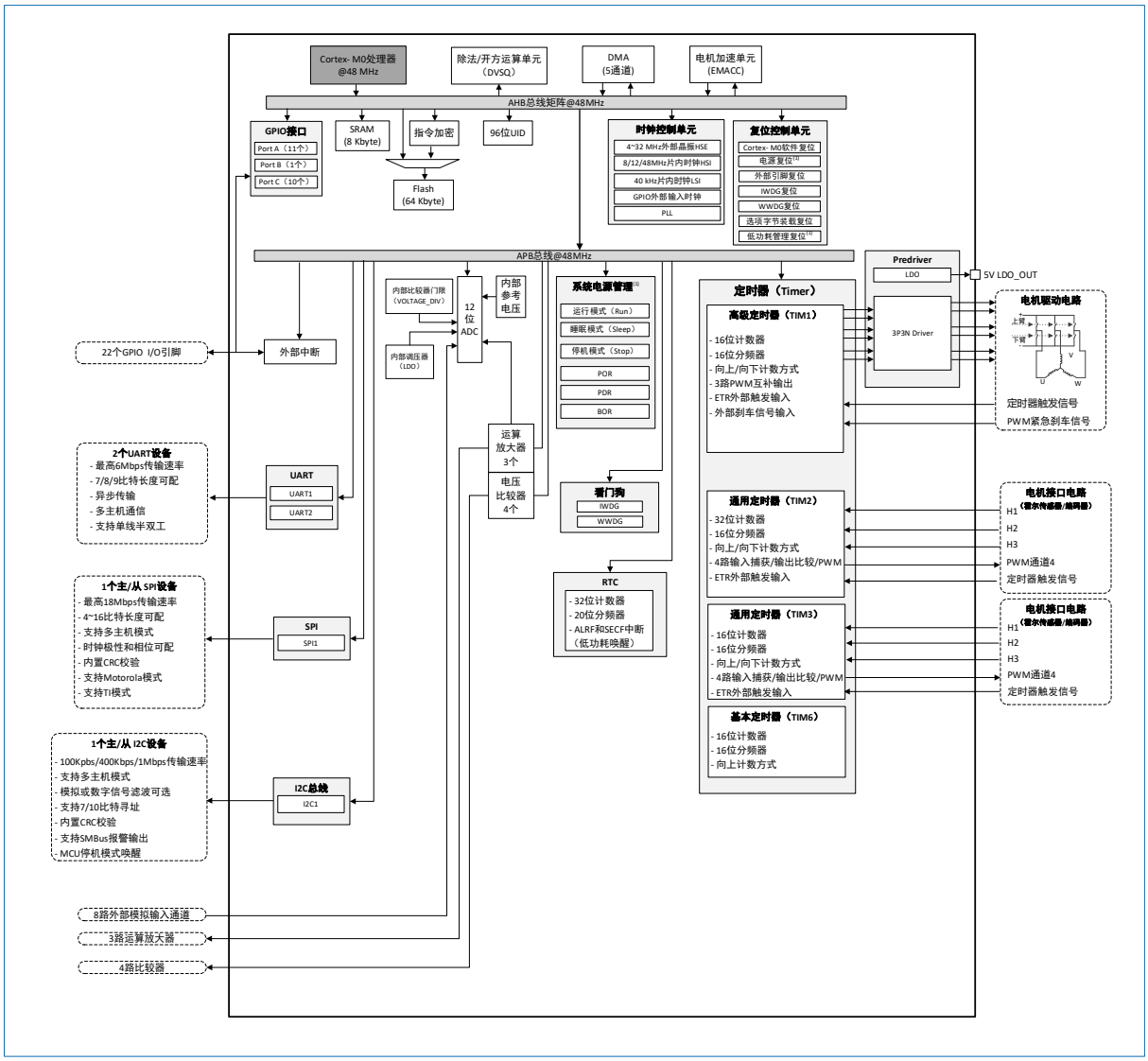

图 2-2 HK32M063C 功能框图

图 2-2 的说明：

(1). 由于 HK32M063C 系列芯片是高压供电，使得原本兼容低压供电设计的系统电源管理、电源复位和低功耗管理复位等功能意义不大，建议用户不使用这几个功能。

2.1.3 HK32M066B 功能框图

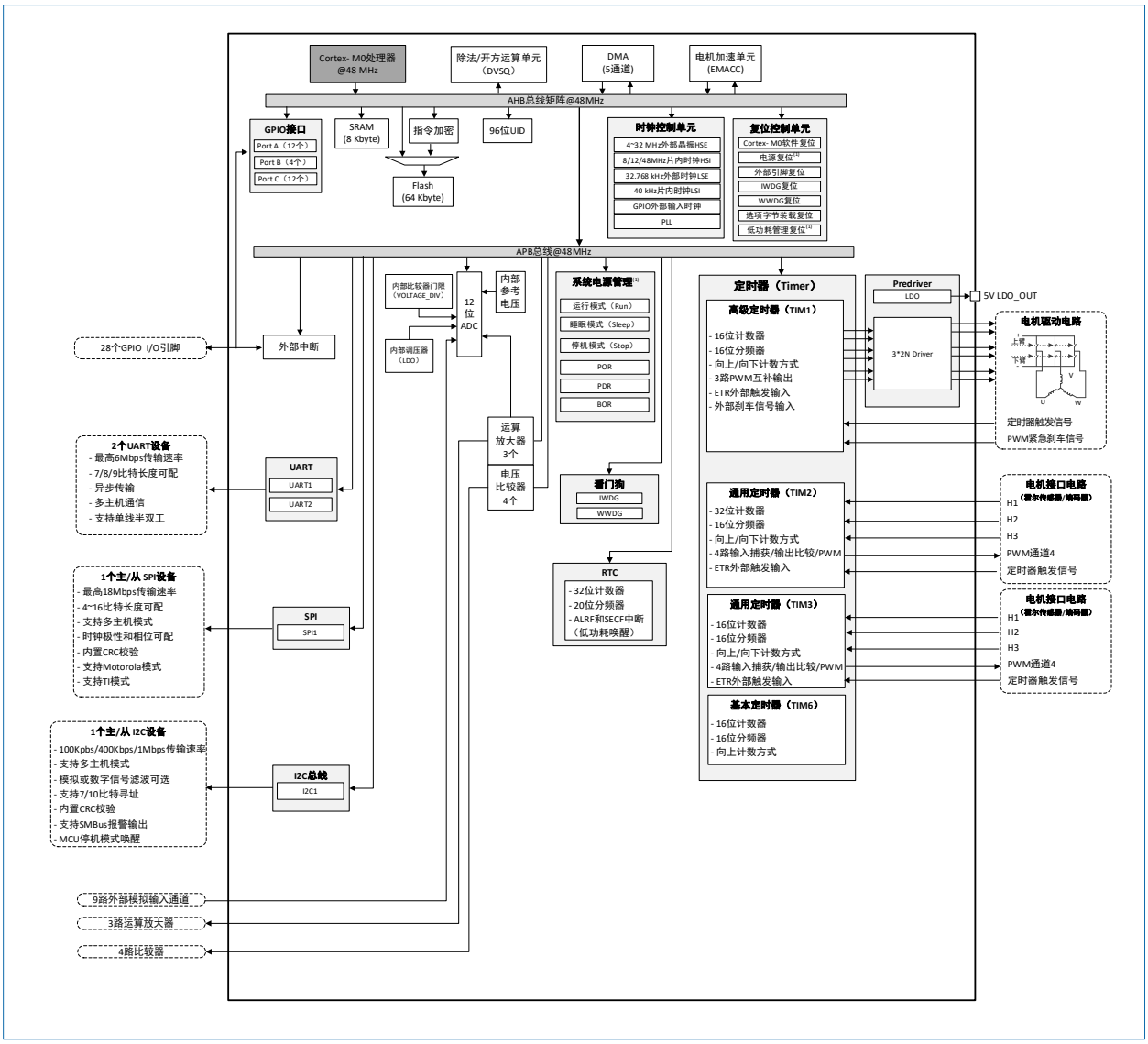

图 2-3 HK32M066B 功能框图

图 2-3 的说明：

(1). 由于 HK32M066B 系列芯片是高压供电，使得原本兼容低压供电设计的系统电源管理、电源复位和低功耗管理复位等功能意义不大，建议用户不使用这几个功能。

2.2 存储器映射

程序存储器、数据存储器、寄存器及I/O 口统一编址，其线性地址空间高达4G Byte。HK32M06x 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。

HK32M06x 的存储器映射如下图所示。

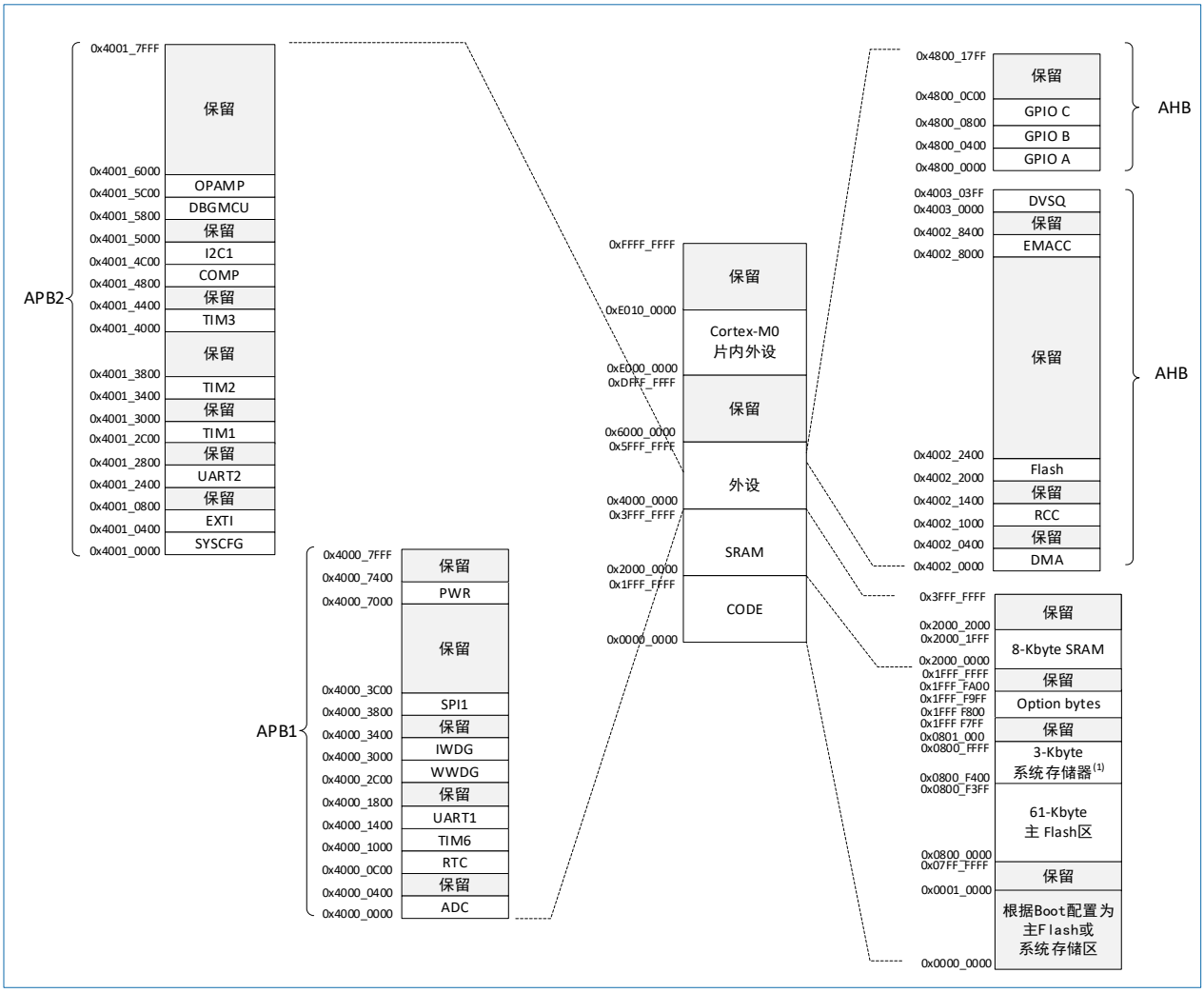

图 2-4 HK32M06x 存储器映射

(1). 64Kbyte Flash 包括 61Kbyte 主 Flash 区和 3Kbyte 系统存储区。3Kbyte 系统存储区也可用作用户代码区。若选择保留系统存储区，则用户仅有 61Kbyte 的主 Flash 区用于存放程序和数据。

2.3 SRAM

HK32M06x 内置 8 Kbyte 的 SRAM。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。

2.4 启动配置

可通过 BOOT0 引脚配置两种不同的启动模式，如下表所示。

表 2-1 启动模式说明

启动模式选择		启动模式	说明
BOOT_PIN_SEL[15:0]（Option byte 配置位）	BOOT0（引脚）		
非 0xADBC	x	主 Flash	主 Flash 存储器选为启动区
0xADBC	0	内置 SRAM	主 Flash 存储器选为启动区
	1	系统存储器	系统存储器选为启动区

当从复位模式唤醒时，CPU 将采样启动模式的配置值，以选择启动模式。在启动延迟之后，CPU 从

地址 0x0000 0000 获取堆栈顶的值，并从所选择的启动存储器的地址 0x0000 0004 开始执行代码。

根据选定的启动模式，可按照以下方式访问主 Flash 存储器或系统存储器：

- 从主 Flash 存储器启动：主 Flash 存储器被映射到启动存储空间（0x0000 0000），但仍能从它原有的地址空间（0x0800 0000）进行访问。即 Flash 存储器的内容可从两个地址开始访问：0x0000 0000 或 0x0800 0000。
- 从系统存储器启动：系统存储器被映射到启动空间（0x0000 0000），但仍能够从它原有的地址空间（0x0800 F400）进行访问。

2.5 物理重映射

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。

下图说明了中断向量表地址映射的关系：

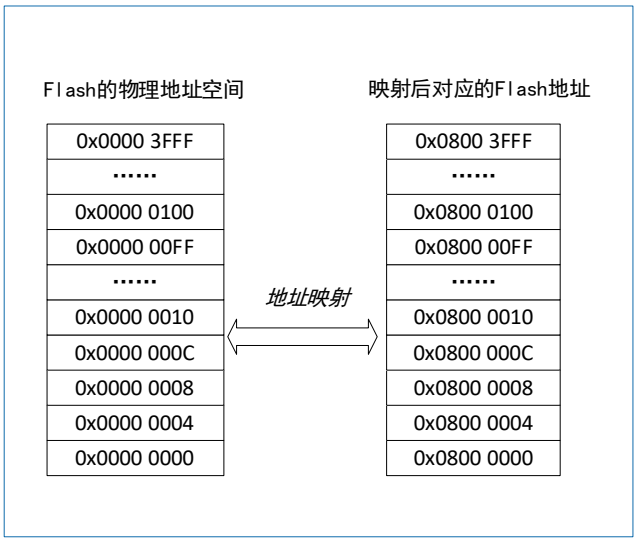

图 2-5 中断向量表地址映射

Flash 的物理地址空间：Flash 的物理地址空间从 0x0000 0000 开始向上增长，中断向量表也从低地址开始存放。

CPU 访问 Flash 的地址映射规则：规定了 CPU 访问 0x0800 0000 地址，实则为访问 Flash 物理地址 0x0000 0000。

2.6 内嵌的自举程序

内嵌的自举程序存放在系统存储器，在 MCU 生产时写入。该程序可以通过 UART1 PC8/PC9 引脚对 Flash 进行重新编程。

3 Flash

本章描述了 Flash 特性和 Flash 寄存器描述。

3.1 Flash 特性

- Flash 结构
  - 64 Kbyte 主 Flash 块（61Kbyte 用户代码区和 3Kbyte ISP Bootloader 区，其中 ISP 区也可以通过选项字节配置用作用户代码区），其包括：
    - 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。
    - 系统存储器（System memory）：包含 Bootloader 代码，共 3 Kbyte（可以通过选项字节配置用作用户代码区）。
- Flash 物理特性
  - 数据位宽：32 位
  - 页大小：256 byte
- Flash 访问位宽：支持字（32 位）编程（不支持半字编程）；32 位读
- 内含加解密模块：支持 Flash 指令自动加解密、保护片内软件知识产权（IP）
- 支持 Flash 读/写保护访问控制
- 内置预取指令缓冲区
- 支持页擦除和全片擦除

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，可存储程序代码和数据。主 Flash 块可划分为 256 页（每页 256 byte）。

表 3-1 Flash 结构

Flash 区	地址	大小	页号	描述
主 Flash 存储器	0x0800 0000-0x0800 00FF	256byte	Page0	用户代码区
	0x0800 0100-0x0800 01FF	256byte	Page1	
	……	……	……	
	0x0800 0700-0x0800 07FF	256byte	Page7	
	0x0800 0800-0x0800 08FF	256byte	Page8	
	0x0800 F200-0x0800 F3FF	256byte	Page243	
主 Flash 存储器/系统存储器	0x0800 F400-0x0800 FFFF	3 Kbyte	Page244~255	可通过选项字 ISP_AWRP 来选择这 3 Kbyte 区用作主 Flash 存储器或系统存储器。

Flash 区	地址	大小	页号	描述
选项字节	0x1FFF F800-0x1FFF F8FF	256 byte	-	选择字节区域

3.2.2 读操作

嵌入式 Flash 模块可以直接寻址访问。对 Flash 内容的读操作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）所指定的方式执行：

- 取指：预取值缓冲区使能后可提高 CPU 运行速度。
- 等待周期：正确读操作的周期个数，保证正确地读取。

取指

Cortex-M0 通过 AHB 总线取指。预取指模块能提高取指效率。

预取指缓冲区

预取指缓冲区分为 3 块，每块 4 个字节，能够完全替代一次同样大小的 Flash 读取访问。预取指缓冲区能够提高 CPU 执行效率，因为 CPU 取一个字指令的同时下一个字的指令内容已经在预取指缓冲区中。这意味着如果代码是按照 32 位对齐的前提下，可以达到 2 倍的取指加速。

预取指缓冲区只有在等待周期大于 0 的时候才有效。在无等待周期的时候，是否预取指令对性能无影响。预取的效率依赖于应用程序。

预取指控制器

预取指控制器会根据预取指缓冲区的可用空间来控制访问 Flash 的时机。当预取指缓冲区中存在至少一块可用空间时，预取指控制器会发起一次读取请求。复位后，预取指缓冲区的默认是关闭状态。

访问等待周期

为了确保对 Flash 的正确读取，必须在 FLASH_ACR 寄存器的 LATENCY[2:0]中指定预取指控制器的速度比。这个数值等于每次访问 Flash 结束到下次访问开始之间所需插入的等待周期的个数。复位后，等待周期默认为零，也就是没有插入等待周期。

3.2.3 读保护

将选项字节中的RDP 字节置位，然后重新复位，Flash 存储器的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 选项字节及其反码内容的对应关系，如下表：

表 3-2 读保护级别和 RDP 字节及其反码的对应关系

RDP 字节值	RDP 反码值	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 在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反码。

- 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 或从系统存储区启动模式：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。任何尝试改写的操作都会引起 FLASH_SR 中的 PGERR 标志置位。当 RDP 被重编程为 0xAA 值以回到 Level 0 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，但 Cortex-M0 的调试接口被禁止了，也不支持从 RAM 启动、系统存储区启动等功能。

  在用户模式下，允许对主 Flash 区进行读写和擦除操作；而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。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。

注意：设置为 Level 2 是不可恢复的操作。

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 或从系统存储区启动		
		读	写	擦除	读	写	擦除
主 Flash 区	1	允许	允许	允许	禁止	禁止	禁止 (3)
	2	允许	允许	允许	- (1)	- (1)	- (1)
系统存储区 (2)	1	允许	禁止	禁止	允许	禁止	禁止
	2	允许	禁止	禁止	- (1)	- (1)	- (1)
选项字节	1	允许	允许 (3)	允许	允许	允许 (3)	允许
	2	允许	允许 (4)	禁止	- (1)	- (1)	- (1)
备份寄存器	1	允许	允许	-	禁止	禁止	允许
	2	允许	允许	-	- (1)	- (1)	- (1)

(1). 当使能 Level 2 保护级别，调试口被禁止从 RAM 或从系统存储区启动。

(2). 无论保护级别（0、1 或 2）和执行模式如何，系统内存只能读取访问。

(3). 当 RDP 被改成不保护时，主 Flash 会被擦除。

(4). 除 RDP 以外的其他选项字节均能被再次编程。

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 成功并进入到 Level 0 之前，MCU 已经启动了整片擦除。

3.2.4 写和擦除操作

电路编程（In Circuit Programming，ICP）是指使用 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，在线应用编程（In Application Programming，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 MCU 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。

写和擦除操作在全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：

- 关键字寄存器（FLASH_KEYR）
- 选项字节关键字寄存器（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）
- 选项字节寄存器（FLASH_OBR）
- 写保护寄存器（FLASH_WRPR）

在写/擦除 Flash 时编程，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。

3.2.4.1 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。

错误的解锁操作顺序将会锁死 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 正确且 KEY2 错误时，会在 KEY2 错的时刻触发中断。

3.2.4.2 Flash 编程

主 Flash 一次可以 32 位（字）编程，由 FLASH_CR 控制。

解锁 Flash 后：

当 FLASH_CR 中的 WPG 位为 1 时，直接对相应的地址写一个字（32 位），是一次编程操作。

标准编程

Flash 存储器接口会预判待编程地址的内容是否已擦除（全 1）。如果未全擦除，则编程操作被自动取消，并且通过 FLASH_SR 寄存器的 PGERR 位提示编程错误告警。但是，如果待编程的内容为全零，则会将其正确编程为零，而不提示错误告警。

如果待编程地址所对应的 FLASH_WRPR 中的写保护位有效，不会有编程动作，并会产生编程错误告警。编程操作结束后，FLASH_SR 寄存器中的 EOP 位置位。

主 Flash 字编程的流程如下图所示：

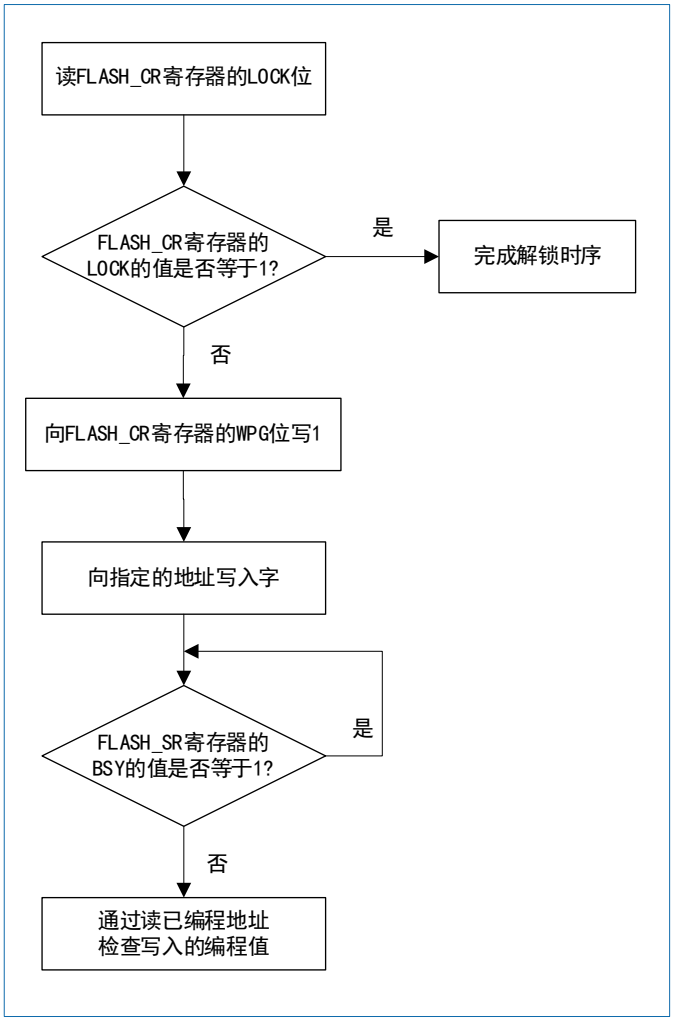

图 3-1 对 Flash 的字编程

主 Flash 存储器的标准字编程流程如下：

1. 检查 FLASH_SR 中的 BSY 位，以确认上次操作已经结束。
2. 置位 FLASH_CR 寄存器中的 WPG 位（字编程），以写入 Flash。
3. 根据配置，以字为单位向目标地址写入数据。
4. 等待 FLASH_SR 寄存器中的 BSY 变为 0。
5. 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。
6. 在完成 Flash 的数据写入后，将 FLASH_CR 寄存器中的 WPG 位置为 0。

3.2.4.3 Flash 擦除

Flash 存储器可以支持页擦除及整片擦除。

- 页擦除

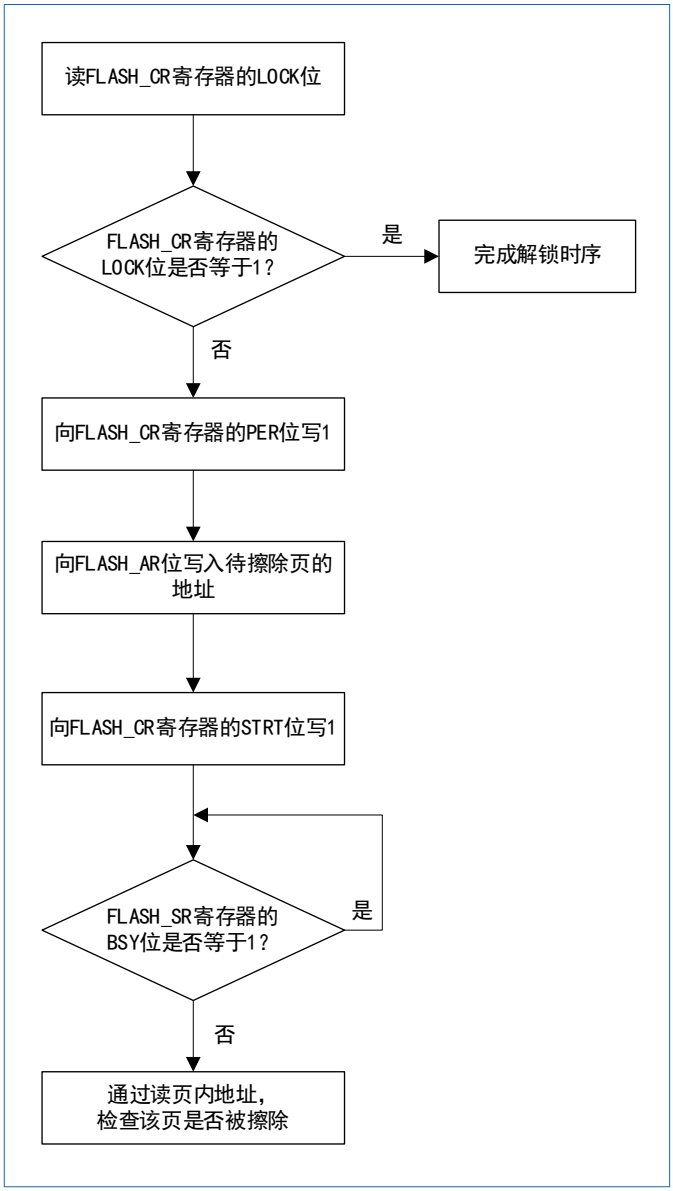

图 3-2 Flash 擦除页的流程

页擦除的操作步骤如下：

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 位清零，以恢复默认值。

- 整片擦除

整片擦除命令可以一次擦除整个主 Flash 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 位清零，以恢复默认值。

注意：整片擦除命令对信息块不起作用。

3.2.4.4 写保护

写保护操作是以 4 扇区（8 页）为单位。通过配置选项字节中的 WRP 位，然后重新复位系统（通过 FLASH_CR 寄存器的 OBL_LAUNCH 位置 1）以使能写保护功能。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。

说明：仅当 WRP 和 nWRP 具有按位互反关系时，写保护才能生效。

写保护的解除

解除写保护实例的操作步骤如下：

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。
3. 置 FLASH_CR 中的 OBL_LAUNCH 位为 1，引起选项字节（包括新 WRP 位）重新加载和写保护解除。

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。写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 Flash 数据加密

Flash 控制器内部有一个加解密模块，用于加解密用户存储在 Flash 中的程序和数据。用户可以选择以密文或者明文的方式存储。当用户使能 Flash 数据加解密后，Flash 控制器会自动进行加解密运算，加解密过程对应用程序透明。如果用户使能 Flash 数据加解密后，同样的程序在两颗不同芯片上实际存储的内容也不一样。加密使用的用户密钥（UKEY）为 64 位。

芯片默认禁能 Flash 加解密。

在 Flash 的选项字节区中存储有加密/解密使能位和 UKEY。在系统复位时，Flash 控制器会自动从 Flash 载入加密/解密使能标志和 UKEY。另外还可以配置一组影子寄存器来改写加解密标志和 UKEY，以便通过调试口调试程序的时候可以直接写入使能加密，写入密文到 Flash。如果已经把 KEY 保存到 Flash 选项字，然后再读该选项字地址，则读到的值始终为 0xAAAA AAAA，以保证 UKEY 不会泄露。

当使能加密后，只加密主 Flash 块，不会加密 Flash 系统存储器和选项字节区。

注意：

如果主 Flash 块的内容被加密后，但是没有使能解密，则 CPU 读到的是密文，会执行出错。如果主空间的内容没有被加密，但是使能解密了，则CPU 读到的也是乱码，会执行出错。因此在程序执行时加密和解密必须同时使能或者禁能。

在使能 Flash 数据加解密后，如果应用程序读 Flash 刚擦除完且还未写入数据的地址，则读到的数据不是 0xFFFF FFFF，而是乱码，但此时应用程序可以成功写数据到这些地址，并不会触发编程错误告警

（PGERR）。同理，在使能 Flash 数据加密后，CPU 读到的数据为 0xFFFF FFFF 也不代表该地址没有被编写过数据。

使能 Flash 数据加解密的步骤如下：

1. 向 Flash 地址 0x1FFF F820 写入 0x1357 ECA8（配置 ENCRY_EN 以使能 Flash 数据加密）。示例如下：

```
FLASH->KEYR = 0x45670123 ;
FLASH->KEYR = 0xCDEF89AB ;
FLASH->OPTKEYR = 0x45670123 ;
FLASH->OPTKEYR = 0xCDEF89AB ;
FLASH->CR |= 0x00000010 ;
*（(volatile u32 *)（0x1FFFF820)）=0x1357ECA8;
while（ （ FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xFFFFFFFE ;
```

2. 向 Flash 地址 0x1FFF F824 写入 0x2468 DB97（配置 DECRY_EN 以使能 Flash 数据解密）。示例如下：

```
FLASH->KEYR = 0x45670123 ;
FLASH->KEYR = 0xCDEF89AB ;
FLASH->OPTKEYR = 0x45670123 ;
FLASH->OPTKEYR = 0xCDEF89AB ;
FLASH->CR |= 0x00000010 ;
*（(volatile u32 *)（0x1FFFF824)） = 0x2468DB97;
while（ （ FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xFFFFFFFE ;
```

3. 向 Flash 地址 0x1FFF F828~0x1FFF F82C 写入 UKEY，UKEY 的高字节写到高地址。示例如下：

```
FLASH->KEYR = 0x45670123 ;
FLASH->KEYR = 0xCDEF89AB ;
FLASH->OPTKEYR = 0x45670123 ;
FLASH->OPTKEYR = 0xCDEF89AB ;
FLASH->CR |= 0x00000010 ;
*（(volatile u32 *)（0x1FFFF828)） = UKEY[31:0];
while（ （ FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
*（(volatile u32 *)（0X1FFFF82C)） = UKEY[63:32];
while（ （ FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xFFFFFFFE ;
```

4. 烧录程序至 Flash。

注意：

向 Flash 地址 0x1FFF F828~0x1FFF F82C 写入 UKEY 后，不能读其值进行校验，因为往 FLASH 地址 0x1FFF F828~0x1FFF F82C 写入 UKEY 后再读 0x1FFF F828~0x1FFF F82C 得到的值始终为 0xAAAA AAAA。

向寄存器 0x4002 2084 和 0x4002 2080 写入 UKEY 后，也不读其值进行校验。

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	ERRIE

3.3 选项字节

用户可根据应用需求配置 Flash 选项字节。配置示例：配置看门狗是由硬件还是软件模式启动。一个 32 位选项字可拆分成如下表所示的选项字节：

表 3-5 选项字拆分

位[31:24]	位[23:16]	位[15:8]	位[7:0]
选项字节 1 反码	选项字节 1	选项字节 0 反码	选项字节 0

选项字若有内容更新，需复位系统才能生效。置位 OBL_LAUNCH 或触发 NRST 引脚产生系统复位。前面 4 个选项字以反码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	位[31:24]	位[23:16]	位[15:8]	位[7:0]
0x1FFF F800	nUSER	USER	nRDP	RDP
0x1FFF F804	nDATA1	DATA1	nDATA0	DATA0
0x1FFF F808	nWRP1	WRP1	nWRP0	WRP0
0x1FFF F80C	nWRP3	WRP3	nWRP2	WRP2
0x1FFF F810	保留		nBOR_LEVEL	BOR_LEVEL
0x1FFF F814～0x1FFF F81F	保留			
0x1FFF F820	ENCRY_CTL[31:0]			
0x1FFF F824	DECRY_CTL[31:0]			
0x1FFF F828	UKEY[31:0]			
0x1FFF F82C	UKEY[63:32]			
0x1FFF F830	IWDG_INI_KEY[15:0]		保留	IWDG_RL_IV[11:0]
0x1FFF F834	ISP_AWRP[15:0]		IWDG_LP_CTL[15:0]	
0x1FFF F838	NRST_IOEN[15:0]		BOOT_SEL[15:0]	
0x1FFF F83C	保留			
0x1FFF F840	DBG_CTL_KEY[31:0]			

表 3-7 选项字节描述

地址	位域	选项字节描述
0x1FFF F800	31:24	nUSER：USER 按位取反
	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。 ● 当进入停机模式时，复位事件。 位[23:18]：保留 位[17]：nRST_STOP ● 0：当进入停机模式时，产生复位。 ● 1：当进入停机模式时，不产生复位。 位[16]：WDG_SW ● 0：硬件看门狗 ● 1：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 F804	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 F808	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 8（Page）进行控制。WRP0作用于0~63 页；WRP1作用于 64~127 页。
0x1FFF F80C	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 8 页（Page）进行控制。WRP2 作用于 128~191 页。WRP3 作用于 192~255 页（当 244~255 页用作主 Flash 区时）。
0x1FFF F810	31:16	保留

地址	位域	选项字节描述
	15:0	位[15:8]：nBOR_LEVEL 位[7:0]：BOR_LEVEL[7:0]位，控制 BOR 状态，BOR 默认处于关闭状态。 BOR_LEVEL[5:7]：保留 BOR_LEVEL[4]=0 时：关闭 BOR BOR_LEVEL[3]：保留 BOR_LEVEL[2:0]：控制 BOR 的检测电压 • 000：复位门限阈值 2.2V • 001：复位门限阈值 2.6V • … • 011：复位门限阈值 3.4V • … • 111：复位门限阈值 5V BOR 阈值步进是 0.4V，迟滞 150 mV~400mV。
0x1FFF F820	31:0	ENCRY_CTL[31:0]：Flash 数据加密使能控制 • 0x1357 ECA8：使能 Flash 数据加密 • 其它值：禁用 Flash 数据加密
0x1FFF F824	31:0	DECRY_CTL[31:0]：Flash 数据解密使能控制 • 0x2468 DB97：使能 Flash 数据解密 • 其它值：禁用 Flash 数据解密
0x1FFF F828	31:0	UKEY[31:0]：用户密钥低 32 位
0x1FFF F82C	31:0	UKEY[63:32]：用户密钥高 32 位
0x1FFF F830	31:16	IWDG_INI_KEY[15:0]：决定IWDG_RL_IV是否生效。 当 IWDG_INI_KEY[15:0]为 0x5B1E 时，IWDG_RL_IV配置有效，否则无效。
	15:12	保留
	11:0	IWDG_RL_IV[11:0]：存储 IWDG_RLR 寄存器的初始值，当 IWDG 配置为硬件看门狗（WDG_SW=0）时，可配置 IWDG_RL_IV[11:0]来设计 IWDG 的复位时间间隔。
0x1FFF F834	31:16	ISP_AWRP[15:0]：等于 0x34CB 时，允许编程和擦除 ISP bootloader 空间；为其它值时不允许编程和擦除 ISP bootloader 空间。
	15:0	IWDG_LP_CTL[15:0]：Stop 模式下 IWDG 计数器是否工作 • 0x369C：MCU 进入 Stop 模式后，强制关闭 IWDG 时钟，IWDG 计数器暂停计数。 • 其他值：MCU 进入 Stop 模式后，IWDG 计数器不暂停。
0x1FFF F838	31:16	NRST_IOEN[15:0]：PC7 引脚功能选择 • 0x3546：PC7 引脚为 NRST 功能。 • 其他值：PC7 引脚为 GPIO PC7 功能或其它 AF 功能。
	15:0	BOOT_SEL_KEY[15:0]：PC0 引脚功能选择 • 0xADBC：PC0 引脚为 BOOT0 功能。 • 其它值：PC0 引脚为 GPIO PC0 功能或其它 AF 功能。

地址	位域	选项字节描述
0x1FFF F840	31:0	DBG_CTL_KEY[31:0]：开关芯片内部调试组件的时钟 ● 0x1234 BCDE：禁能 debug 引脚功能，debugger 不能连接 MCU。 ● 其它值：使能 debug 引脚功能。

表 3-8 Boot 模式

BOOT_SEL_KEY	BOOT0 引脚	启动方式
其他值	x	从主 Flash 启动
0xADBC	0	从主 Flash 启动
0xADBC	1	从系统存储器启动

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 Flash 写保护寄存器（FLASH_WRPR）中。

编程 0x1FFF_F800~0x1FFF_F81F 地址空间，会自动把位[15:8]写入位[7:0]的反码，把位[31:24]写入位[23:16]的反码。

在芯片复位后，0x1FFF_F800~0x1FFF_F811 地址的值自动装载时，会校验位[15:8]是否为位[7:0]的反码，如果是，则位[7:0]有效，否则使用对应的默认值；同时也会校验位[31:24]是否为位[23:16]的反码，如果是，则位[23:16]有效，否则使用对应的默认值。

3.3.1.1 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 13 个字，包括：

- 1 个字包含读保护和用户选项
- 1 个字用户数据
- 2 个字的写保护
- 半字的 BOR 配置
- 1 个字的加密使能
- 1 个字的解密使能
- 2 个字的用户密钥
- 1 个字的 DEBUG 时钟配置
- 1 个字 IWDG 配置
- 半字的 Bootloader 空间当作用户代码区的使能配置，半字的低功耗模式下 IWDG 时钟配置
- 半字的 GPIO PC7 引脚替换为 NRST 功能的配置，半字的 GPIO PC0 引脚替换为 BOOT0 pin 的配置

选项字节编程前的准备：

编程操作开始前，需要对 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或编程操作。编程操作开始前，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. Flash 空间的解锁
   A. 向 FLASH_KEYR 寄存器写入关键字 KEY1=0x45670123；

B. 向 FLASH_KEYR 寄存器写入关键字 KEY2=0xCDEF89AB。

2. FLASH 选项字空间解锁

A. 向 FLASH_OPTKEYR 寄存器写入关键字 KEY1=0x45670123；

B. 向 FLASH_OPTKEYR 寄存器写入关键字 KEY2=0xCDEF89AB。

3. 检查 FLASH_SR 寄存器中的 BSY 位，以确保上次操作结束。

4. 将 FLASH_CR 寄存器中的 OPTPG 位置 1，以选择字方式写入。

5. 写数据（字）到目标地址。

6. 等待 FLASH_SR 中的 BSY 位为 0，表示编程操作结束。

7. 将 FLASH_CR 寄存器中的 OPTPG 位置为 0，以恢复默认值。

当 Flash 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选项字节。若用户仅想改写读保护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。

3.3.1.2 选项字节擦除

选项字节是按页擦除的，步骤如下：

1. Flash 空间的解锁

A. 向 FLASH_KEYR 寄存器写入关键字 KEY1=0x45670123；

B. 向 FLASH_KEYR 寄存器写入关键字 KEY2=0xCDEF89AB。

2. FLASH 选项字空间解锁

A. 向 FLASH_OPTKEYR 寄存器写入关键字 KEY1=0x45670123；

B. 向 FLASH_OPTKEYR 寄存器写入关键字 KEY2=0xCDEF89AB。

3. 检查 FLASH_SR 寄存器中的 BSY 位，以确保上次操作结束。

4. 将 FLASH_CR 寄存器中的 OPTER 位置为 1，以擦除整个选项字节区域。

5. 将待擦除的地址写入 FLASH_AR 寄存器（待擦除地址必须属于选项字节表里已有的地址，否则会出现不可预计的错误）。

6. 将 FLASH_CR 寄存器中的 STRT 位置为 1，以启动擦除操作。

7. 等待 FLASH_SR 中的 BSY 位变成 0，表明擦除操作结束。

8. 将 FLASH_CR 寄存器中的 OPTER 位置为 0，以恢复默认值。

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 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										PRFTBS	PRFTBE	Res	LATENCY[2:0]		
										r	rw		rw		

位 31:6	Res：保留 必须保持复位值。
位 5	PRFTBS：预取指缓冲区的状态（Prefetch buffer status） ● 0：禁用预取指缓冲区 ● 1：使能预取指缓冲区
位 4	PRFTBE：预取指缓冲区使能（Prefetch buffer enable） ● 0：禁止预取指 ● 1：使能预取指
位 3	Res：保留 必须保持复位值。
位 2:0	LATENCY[2:0]：等待周期（Latency） 该位预设 HCLK 周期和 Flash 访问时间的比率关系。 ● 000：零等待周期 ● 001：1 个等待周期 ● 010：2 个等待周期 ● 011：3 个等待周期 ● 100：4 个等待周期 ● 101：5 个等待周期 ● 110：6 个等待周期 ● 111：7 个等待周期 ● 推荐配置： ● 当芯片供电电压≥4.5V 时： ○ HCLK ≤ 24MHz，0 等待周期； ○ 24MHz < HCLK ≤ 48MHz，1 个等待周期； ● 当芯片供电电压在 2.7V~4.5V 范围时： ○ HCLK ≤ 19MHz，0 等待周期； ○ 19MHz < HCLK ≤ 38MHz，1 个等待周期； ○ 38MHz < HCLK ≤ 48MHz，2 个等待周期； ● 当芯片供电电压<2.7V 时： ○ HCLK ≤ 14MHz，0 等待周期； ○ 14MHz < HCLK ≤ 28MHz，1 个等待周期； ○ 28MHz < HCLK ≤ 42MHz，2 个等待周期； ○ 42MHz < HCLK ≤ 48MHz，3 个等待周期； 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
FKEY[31:16]															
w															

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

位 31:0	FKEY[31:0]：Flash 关键字（Flash key） 该位域用于存储可解锁 Flash 的关键字。 ● KEY1：0x45670123 ● KEY2：0xCDEF89AB

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。 ● KEY1：0x45670123 ● KEY2：0xCDEF89AB

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
										rw	rw		rw		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） 当对写保护区域进行写操作，该位将被硬件置位。该位由软件写 1 清零。
位 3	Res：保留 必须保持复位值。
位 2	PGERR：写入错误标志（Programming error） 字编程时，被编程区的值不为'0xFFFF FFFF'，对被编程区进行写入操作，则该位被硬件置位。 该位由软件写 1 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		OBL_LAUNCH	EOPIE	Res	ERRIE	OPTWRE	Res	LOCK	STRT	OPTER	OPTPG	Res	MER	PER	WPG
		rw	rw		rw	rc_w0		rw	rw	rw	rw		rw	rw	rw

位 31:14	Res：保留 必须保持复位值。
位 13	OBL_LAUNCH：选项字节强制更新（Force option byte loading） 当被写为 1 时，该位强制选项字节的重加载。该操作会引起系统复位。 ● 0：无效 ● 1：有效
位 12	EOPIE：操作结束中断使能（End of operation interrupt enable） 当 FLASH_SR 中的 EOP 位变为 1 时，该位产生中断请求。 ● 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） 当 FLASH_SR 中的 PGERR/WRPRTERR 位变为 1 时，该位产生中断请求。 ● 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。向 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） 选项字节（0x1fff_f800 ~ 0x1fff_f8ff 地址范围）擦除时选择。
位 4	OPTPG：选项字节写入（Option byte programming） 选项字（32 位）写入时选择。 ● 0：不起作用 ● 1：选项字（对 0x1FFF_F800 ~ 0x1FFF_F8FF 地址范围进行按字编程（32 位）。
位 3	Res：保留 必须保持复位值。
位 2	MER：整片擦除（Mass erase） 整片擦除时选择。 当 ISPA_WRP[15:0]！=0x34CB 时，MER 置 1 只会擦除主 Flash 空间的低 61Kbyte，ISP bootloader 不会被擦除，此时不会产生 WRPRTERR。 当 ISPA_WRP[15:0]=0x34CB 时，MER 置 1 会擦除整个 64Kbytes 主 Flash 空间。
位 1	PER：页擦除（Page erase） 页擦除时选择。 当 ISPA_WRP[15:0]！=0x34CB 时，如果按页擦除 0x0000_F400~0X0000_FFFF 地址或者 0x0800_F400 ~ 0X0800_FFFF，则会擦除失败并产生 WRPRTERR。 当 ISPA_WRP[15:0]=0x34CB 时，可以擦除 64KFlash 里的任意页。
位 0	WPG：字写入（Word programming） ● 1：字编程 Flash ● 0：无意义 ● 当 ISPA_WRP[15:0]！=0x34CB 时，若编程 0x0000_F400~0x0000_FFFF 地址或者 0x0800_F400 ~ 0x0800_FFFF，会编程失败并产生 WRPRTERR。 ● 当 ISPA_WRP[15:0]=0x34CB 时，以字为单位编程 64Kbyte 主 Flash 空间。

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 WPG 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

说明：X 表示不定值。

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
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：当进入停机模式时，是否产生复位（Whether reset when entering STOP mode） ● 0：当进入停机模式时，产生复位。 ● 1：不产生复位
位 8	WDG_SW：选择软件或硬件看门狗（Hardware or software watchdog selection） 该位由选项字节内容决定。 ● 0：硬件看门狗 ● 1：软件看门狗
位 7:3	Res：保留 必须保持复位值。
位 2:1	RDPRT[1:0]：读保护等级状态（Read protection level status） ● 00：读保护等级 0 ● 01：读保护等级 1 ● 11：读保护等级 2
位 0	OPTERR：选项字节错误（Option byte error） 置 1，表示加载的选项字节与其补码不匹配。

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
WRP[31:16]															
r															

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

位 31:0	WRP[31:0]：写保护（Write protection） 该位保持由 OBL 载入的写保护选项字（只读）。共 32 位，每个位标识 2 Kbyte 的 Flash 空间写保护。 ● 0：写保护激活 ● 1：写保护禁止

3.4.9 Flash 数据加密控制寄存器（FLASH_ENCRY_CTL）

偏移地址：0x78

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															ENCRY_EN
															rw

位 31:1	Res：保留 必须保持复位值。
位 0	ENCRY_EN：加密使能（Encryption enable） ● 0：禁用加密 flash ● 1：使能加密 flash 当 Flash 选项字地址 0x1FFF F820 中的值为 0x1357 ECA8 时，ENCRY_EN 的复位值为 1，否则复位值为 0。 读取 FLASH_ENCRY_CTL[31:1]时总是返回 0 。

3.4.10 Flash 数据解密控制寄存器（FLASH_DECRY_CTL）

偏移地址：0x7C

复位值：0x0000 000X

DECRY_EN 的复位值由选项字决定。

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															DECRY_EN
															rw

位 31:1	Res：保留 必须保持复位值。

位 0	DECRY_EN：解密使能（Decryption enable） ● 0：禁用解密 flash ● 1：使能解密 flash 当 Flash 选项字地址 0x1FFF F824 中的值为 0x2468 DB97 时，DECRY_EN 的复位值为 1，否则复位值为 0。 读取 FLASH_DECRY_CTL[31:1]时总是返回 0。

3.4.11 Flash 指令加解密钥低 32 位（FLASH_UKEY1）

偏移地址：0x80

复位值：0x0000 0000

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

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

位 31:0	UKEY[31:0]：指令加解密钥低 32 位（Low 32 bits for Flash data encryption） 用户可以通过写 UKEY1 和 UKEY2 寄存器配置指令加解密密钥，指令加解密密钥默认使用的是 Flash 选项字节保存的 UKEY 值。 读该寄存器时始终返回 0。

3.4.12 Flash 指令加解密钥高 32 位（FLASH_UKEY2）

偏移地址：0x84

复位值：0x0000 0000

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

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

位 31:0	UKEY[31:0]：指令加解密钥高 32bit（High 32 bits for Flash data encryption） 用户可以通过写 UKEY1 和 UKEY2 寄存器配置指令加解密密钥，指令加解密密钥默认使用的是 Flash 选项字节保存的 UKEY 值。 读该寄存器时始终返回 0。

4 电源控制（PWR）

HK32M06x 子系列芯片的供电有所不同，详见后续章节。

4.1 HK32M060 电源

芯片的工作电压（V_{DD}）为 2.2~5.5 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 的输出电压可通过软件根据应用场景进行调节，以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。

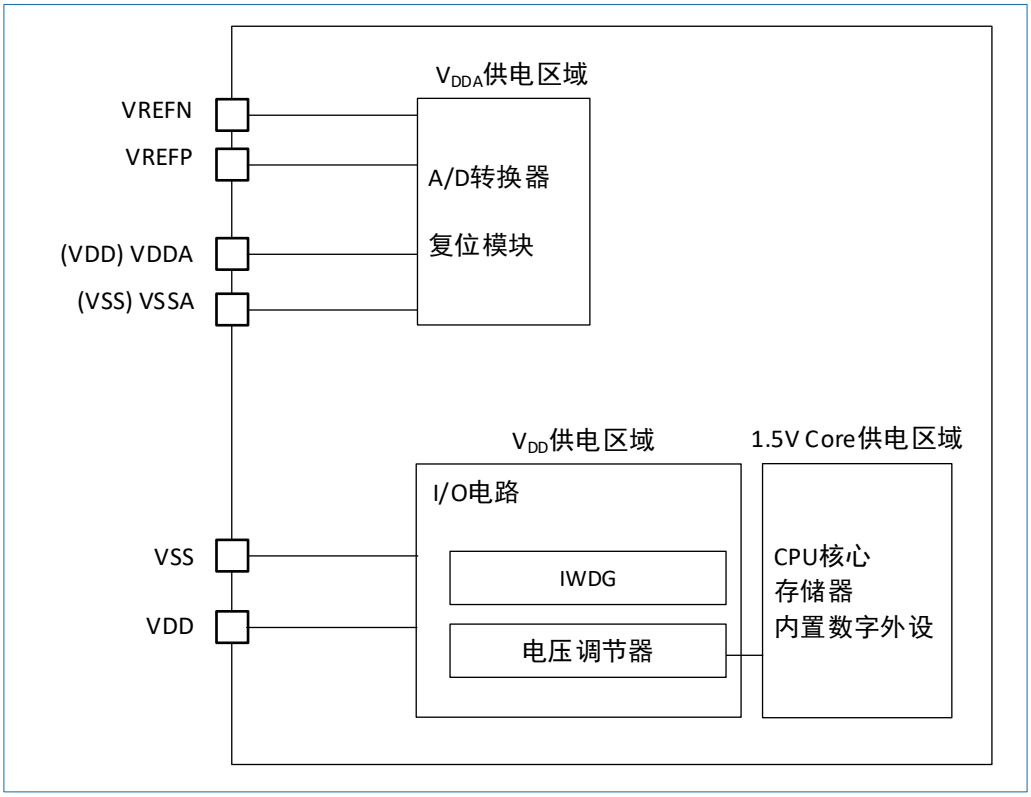

图 4-1 HK32M060 电源框图

注意：V_{DDA} 和 V_{SSA} 在芯片内部分别连到 V_{DD} 和 V_{SS}。

4.1.1 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 使用一个独立的参考电源供电，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 VDDA（VDDA 连到 VDD）
- 电源地 VSSA（VSSA 连到 VSS）
- 独立的参考电源供电引脚为 VREFP
- 独立的电源供电地引脚为 VREFN

4.1.2 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。

4.2 HK32M063C 电源

如图 4-2 所示：

- V_M = 7~28V：输入工作电压范围 7~28V，内置输入欠压功能，为内部 5V LDO 和三路栅极驱动器供电。
- $V_{LDO05}/V_{DD}/V_{DDA}$ =4.5 ~ 5.5V：内部 5V LDO 输出，同时也为芯片的低压域供电，内置过温关闭保护功能。通过内置的电压调节器提供所需的 1.5 V 电源。
- V_{REFP} = 2.2 ~ 5.5V：管脚为 ADC、电压比较器及运算放大器等模拟部分提供参考电压。

说明：

(1). V_{REFP} 为独立引脚。

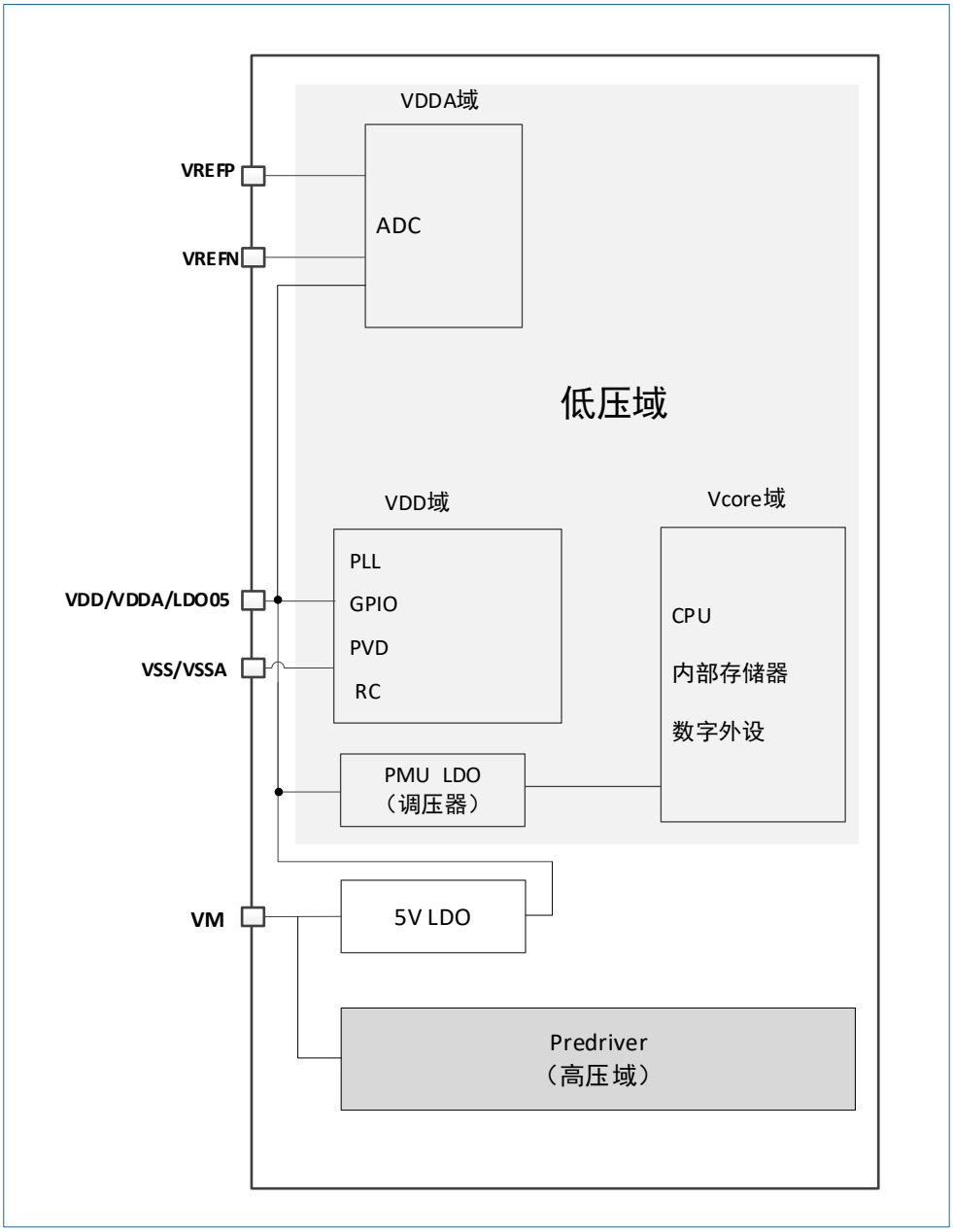

图 4-2 HK32M063C 电源框图

注意：V_{DDA} 和 V_{SSA} 在芯片内部分别连到 V_{DD} 和 V_{SS}。

4.2.1 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 使用一个独立的参考电源供电，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 VDDA（VDDA 连到 VDD）
- 电源地 VSSA（VSSA 连到 VSS）
- 独立的参考电源供电引脚为 VREFP

4.2.2 电压调节器

说明：HK32M063C 系列芯片为高压电源供电，通过电压调节器来降低的功耗相较于高压供电的消耗功耗可忽略，用户可跳过本章节内容。

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。

4.3 HK32M066B 电源

如图 4-3 所示：

- V_M = 10~36V：输入工作电压范围 10~36V，内置输入欠压功能，为内部 5V LDO 和三路 N&N MOS 半桥栅极驱动器供电。
- V_{LDO05} =5V：内部 5V LDO 输出，内置输出短路保护功能。
- V_{DD}/V_{DDA}=3.3 ~ 5 V：为芯片的低压域供电，推荐电压值为 5V。
- V_{REFP} = 2.2 ~ 5.5V：为 ADC、电压比较器及运算放大器等模拟部分提供参考电压。

说明：

(1). V_{DD} 和 V_{DDA} 在内部连接在一起。

(2). V_{REFP} 为独立引脚。

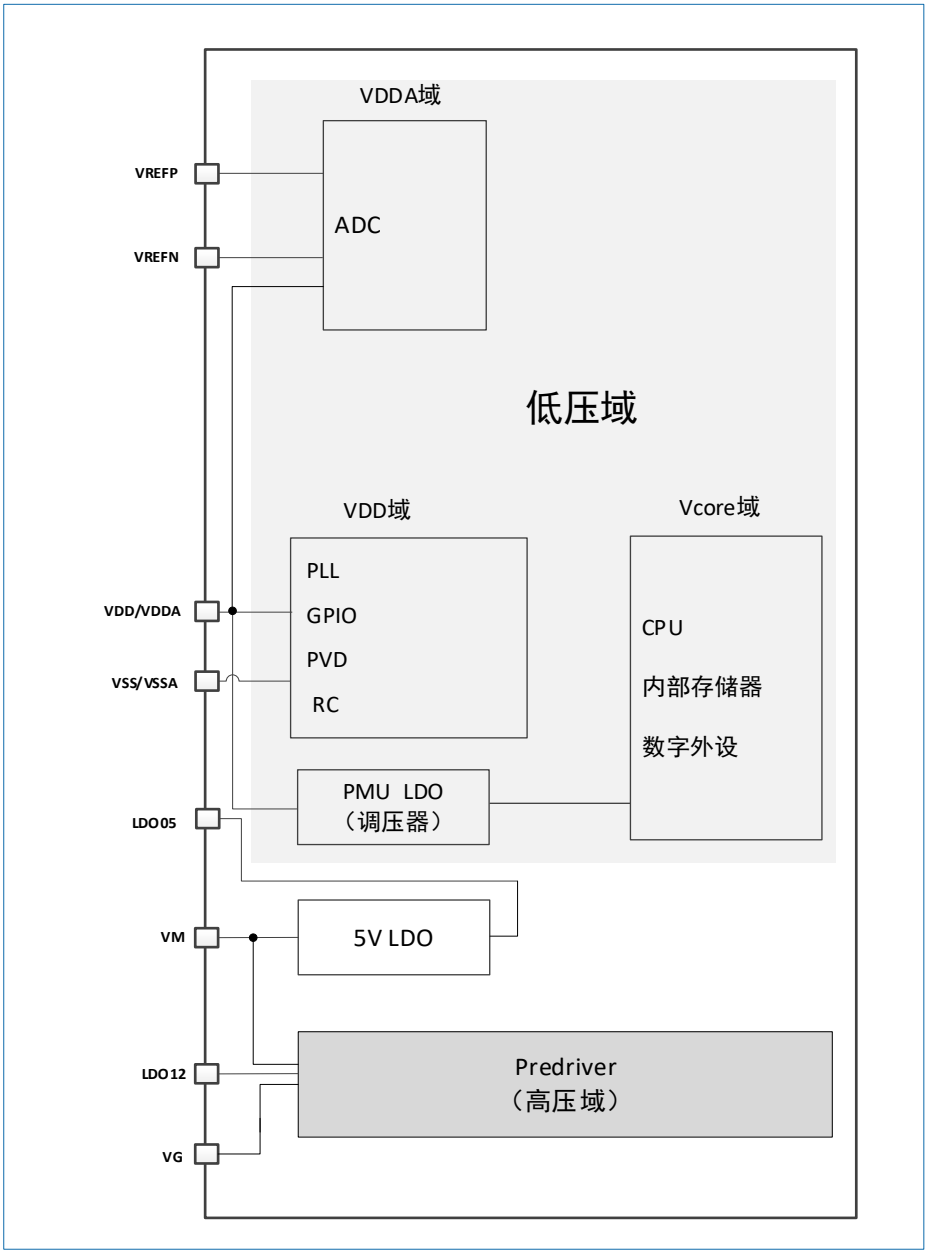

图 4-3 HK32M066B 电源框图

注意：V_{DDA} 和 V_{SSA} 在芯片内部分别连到 V_{DD} 和 V_{SS}。

4.3.1 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 使用一个独立的参考电源供电，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 VDDA（VDDA 连到 VDD）
- 电源地 VSSA（VSSA 连到 VSS）
- 独立的参考电源供电引脚为 VREFP
- 独立的电源供电地引脚为 VREFN

4.3.2 电压调节器

说明：HK32M066B 系列芯片为高压电源供电，通过电压调节器来降低的功耗相较于高压供电的消耗功耗可忽略，用户可跳过本章节内容。

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。

4.4 电源监控器

说明：HK32M063C 和 HK32M066B 系列芯片为高压电源供电，使得原本兼容低压供电设计的电源监控器功能意义不大，用户可忽略本章节内容。

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 2.2 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 2.2V 开始正常工作。器件达到 BOR 阈值（如2.2V）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 2.2 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 2.2 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 4-4 中说明了不同的电源监控器（POR、PDR、BOR）。

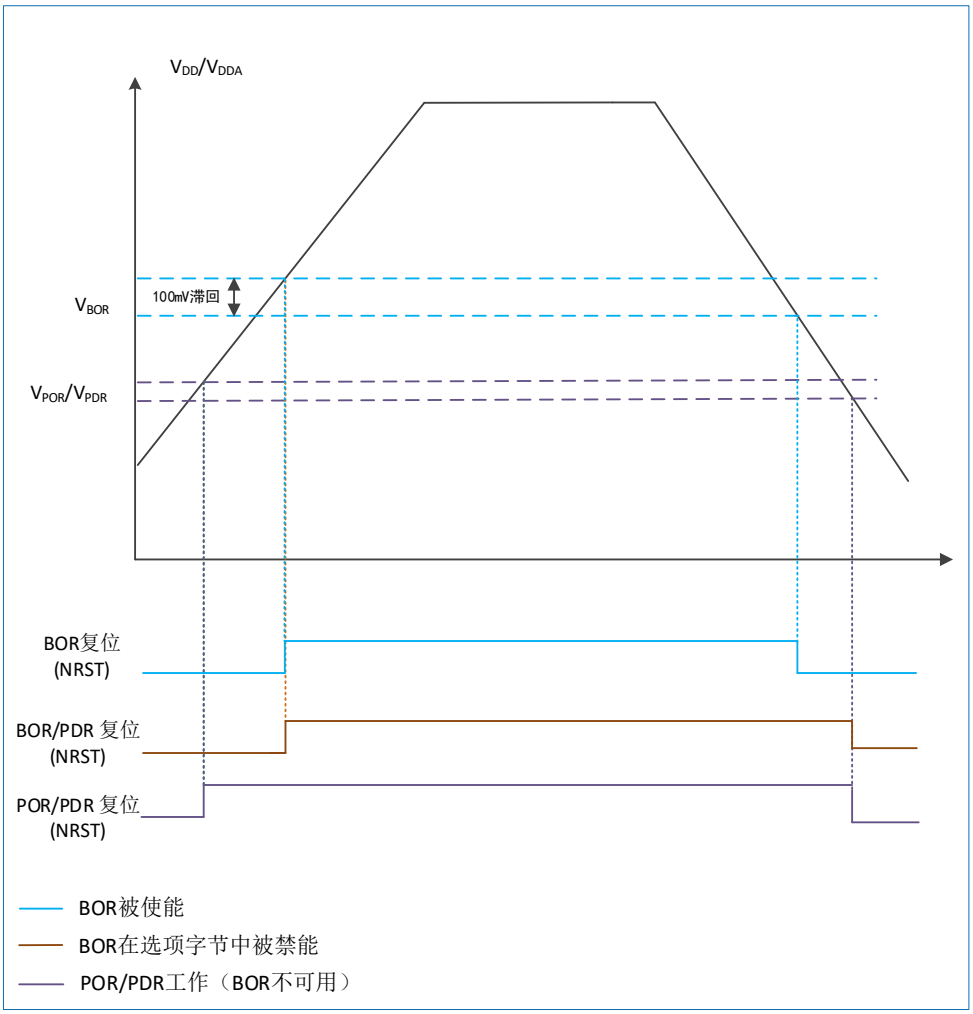

图 4-4 电源监控器

说明：

(1). BOR 功能在工作电压从2.2 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2.2 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。

4.4.1 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。

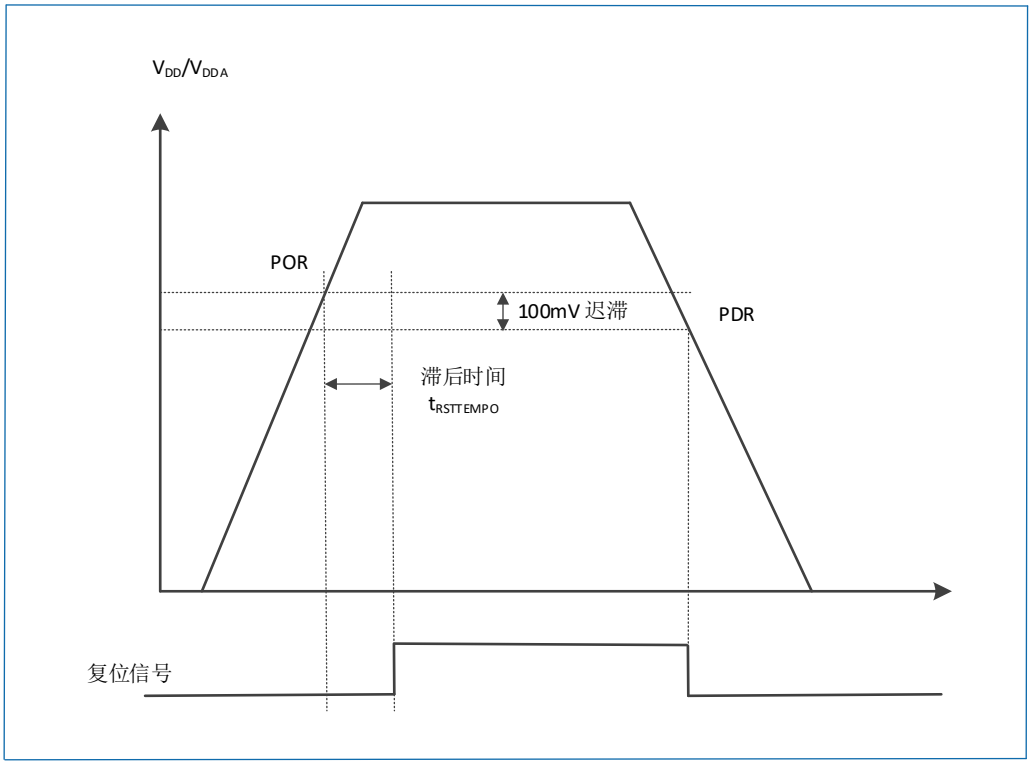

图 4-5 上电复位和掉电复位的波形图

4.4.2 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。BOR 可配置为禁能，电源供电则由POR/PDR 监控。由于 POR/PDR 阈值低于 2.2V，因此在V_{POR}/V_{PDR}阈值和最低产品工作电压 2.2V 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。

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

- BOR 级别 0（VBOR0）：约 2.2 V 的复位阈值级别，迟滞约 150mV。
- BOR 级别 1（VBOR1）：约 2.6 V 的复位阈值级别，迟滞约 200mV。
- BOR 级别 2（VBOR2）：约 3.0V 的复位阈值级别，迟滞约 250mV。
- BOR 级别 3（VBOR3）：约 3.4 V 的复位阈值级别，迟滞约 300mV。
- BOR 级别 4（VBOR4）：约 3.8 V 的复位阈值级别，迟滞约 300mV。
- BOR 级别 5（VBOR5）：约 4.2 V 的复位阈值级别，迟滞约 350mV。

- BOR 级别 6（VBOR6）：约 4.6V 的复位阈值级别，迟滞约 400mV。
- BOR 级别 7（VBOR7）：约 5.0V 的复位阈值级别，迟滞约 400mV。

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“4.4.1 上电/掉电复位（POR/PDR）”）。BOR 阈值滞回电压在150 mV~400㎷ 的范围（电源电压的上升沿与下降沿之间）。

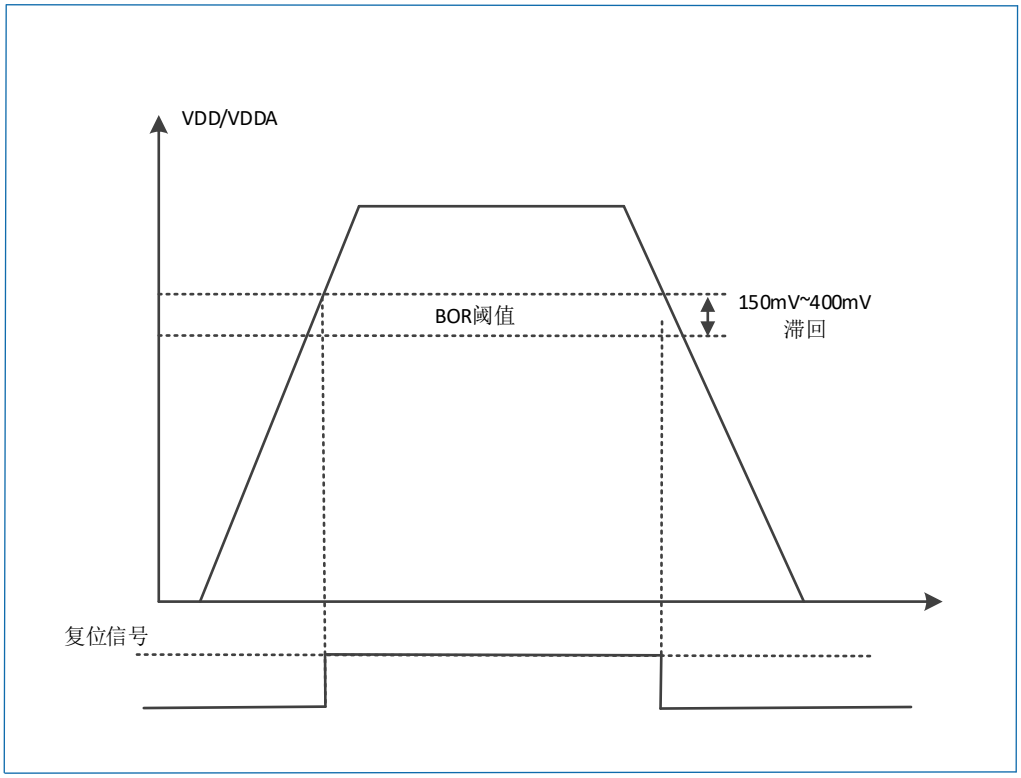

图 4-6 BOR 阈值

4.5 低功耗模式

说明：HK32M063C 和 HK32M066B 系列芯片为高压电源供电，使得原本兼容低压供电设计的低功耗功能意义不大，建议用户不使用该功能，可忽略本章节内容。

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，内核域所有的时钟被关闭，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 4-1 低功耗模式的进入/唤醒条件

工作模式	进入条件	唤醒条件	Vcore 域时钟状态	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（Stop）	1. 设置 PWR_CR:LPDS = 0。 2. 设置 CM0 系统控制寄存器的 SLEEPDEEP 位。 3. 软件执行 WFI/WFE 指令进入。	支持任何一个 EXTI 外部中断线唤醒。 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：“9.1.3 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：“9.2.4 唤醒事件管理”。	所有时钟停止	HSI 关闭	开启或者处于低功耗模式（在 PWR_CR 中设置）

4.5.1 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：“5.3.2 时钟配置寄存器（RCC_CFGR）”。

4.5.2 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“5.3.6 AHB 外部时钟使能寄存器（RCC_AHBENR）”）APB 外设时钟使能寄存器（参见“5.3.8 APB1 外设时钟使能寄存器（RCC_APB1ENR）”和“5.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR）”）来开关各个外设模块的时钟。

4.5.3 睡眠（Sleep）模式

4.5.3.1 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 4-2 和表 4-3。

4.5.3.2 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 4-2 和表 4-3。

表 4-2 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“9.1.3 中断和异常向量”。 ● 如果执行 WFE 进入睡眠模式： 唤醒事件：参考“9.2.4 唤醒事件管理”。
唤醒延时	无

表 4-3 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“9.1.3 中断和异常向量。
唤醒延时	无

4.5.4 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 引脚保持它们在运行模式时的状态。

4.5.4.1 进入停机模式

关于如何进入停机模式，参见表 4-1。

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，通过配置 ADC 寄存器可关闭该外设。

4.5.4.2 退出停机模式

关于如何退出停机模式，可参见表 4-1。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。

4.5.5 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：“24.8.1 对低功耗模式的调试支持”。

4.6 PWR 寄存器

基地址：0x4000 7000

空间大小：0x400

说明：HK32M063C 和 HK32M066B 系列芯片为高压电源供电，可忽略本章节内容。

4.6.1 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。

4.6.2 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。

5 复位和时钟控制（RCC）

5.1 复位

该系列芯片有两种复位方式：系统复位、电源复位。

5.1.1 系统复位

系统复位将复位除时钟控制/状态寄存器（RCC_CSR）中的复位标志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）
- 选项字节装载器复位
- 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

可通过查看 RCC_CSR 控制状态寄存器中的复位状态标志位识别复位事件来源。

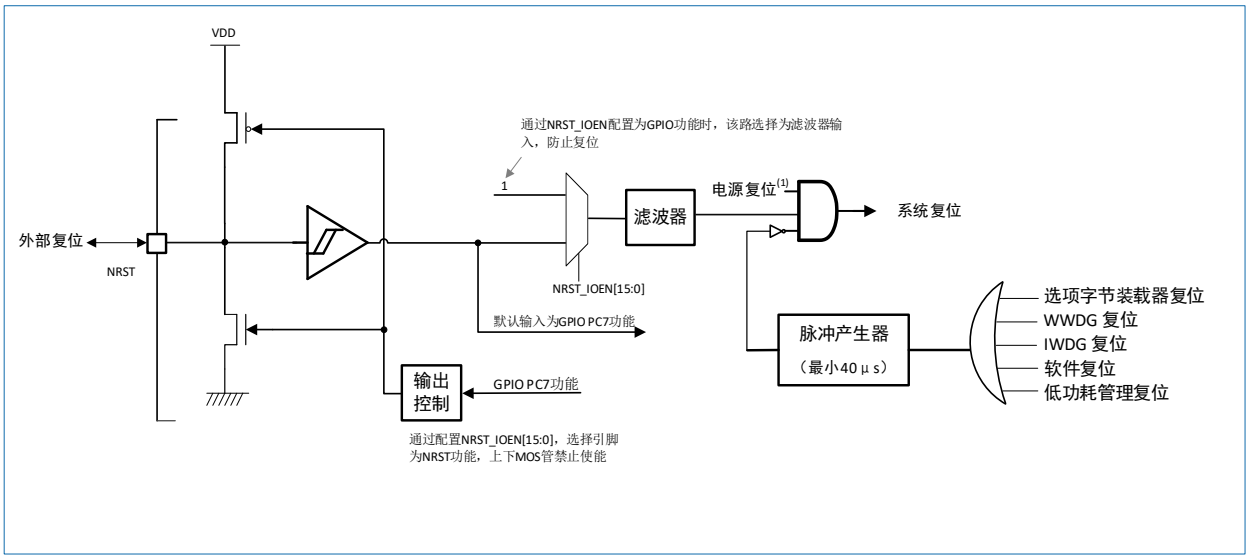

图 5-1 复位电路

图 5-1 的说明：

(1). HK32M063C 和 HK32M066B 系列芯片因为工作电压值的原因，电源复位不起作用。

上图中的复位源（除电源复位以外）将最终作用于 NRST 引脚，并在一定的延时时段内保持低电平。复位入口地址被固定在 0x0000 0004 处。

内部的复位信号（除电源复位以外）会在 NRST 引脚上输出，脉冲发生器保证这些（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。

选项字节装载器复位

在设置 OBL_LAUNCH（FLASH_CR 寄存器中）为 1 的情况下，当软件读取选项字时，发出选项字节装载器复位信号。

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 位清零将使能该复位。这时，即使正处于进入停机模式的进程，系统将被复位而不是进入停机模式。

NRST 与 GPIO 功能切换

当用户选择字节中的 NRST_IOEN 为 0x3546 时，PC7 引脚为 NRST 功能，其他值时为 GPIO PC7 功能或其它 AF 功能。PC7 引脚默认为 GPIO（PC7）引脚功能，但在上电复位后，PC7 引脚的功能从 GPIO 功能自动切换为 NRST 功能。

5.1.2 电源复位

当以下事件发生时，将产生电源复位：

- 上电/掉电复位（POR/PDR）
- 欠压复位（BOR）

电源复位将复位所有寄存器。

5.2 时钟

支持多种时钟源驱动系统时钟（SYSCLK）：

- 外部高速时钟（HSE）：4~32 MHz
- 外部低速时钟（LSE）：32.768 kHz （仅 LQFP48 和 QFN48 封装芯片支持 LSE）
- 片内高速时钟（HSI）：8MHz/12MHz/48MHz
- 片内低速时钟（LSI）：40kHz
- PLL 时钟：48MHz（最大值）
- GPIO 外部输入时钟：5~30 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 域的最大时钟频率为 48 MHz。Cortex 系统定时器由 AHB 时钟驱动，其可由 AHB/8 或 AHB 时钟频率直接驱动（通过 Cortex Systick 配置位来配置）。

所有的外设时钟由其所在的总线时钟（HCLK 或 PCLK）驱动，以下几个除外：

- ADC 时钟由以下任一时钟得到（由软件选择）：
  - HSI12 时钟。
  - APB1（PCLK）时钟的 1/2/4 分频
- UARTx 的时钟为以下的时钟源之一（由软件选择）：
  - HSI12
  - LSI 时钟
  - LSE 时钟
  - APB1 时钟（PCLK）
- I2C1 的时钟为以下的时钟源之一（由软件选择）：
  - HSI 时钟

- 系统时钟（SYSCLK）
- APB1 时钟（PCLK）

RCC 将 AHB 时钟（HCLK）8 分频后作为 Cortex 系统定时器（SysTick）的时钟。通过对 SysTick 控制与状态寄存器的设置，可选择 HCLK/8 时钟或 Cortex（HCLK）时钟作为 SysTick 时钟。

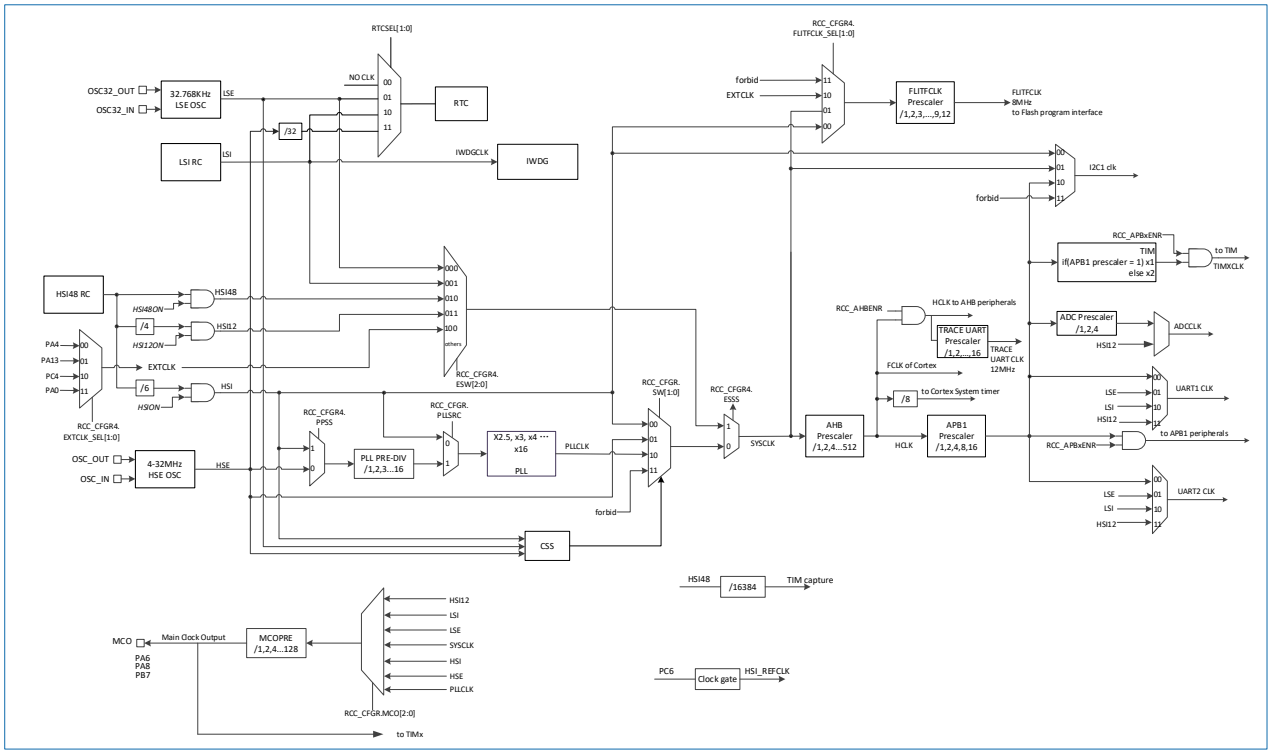

图 5-2 HK32M06x 时钟树

上图说明：HK32M06x 系列中仅 LQFP48 和 QFN48 封装支持 LSE（32.768kHz）。

5.2.1 HSE 时钟

高速外部时钟信号（HSE）由以下两种时钟源产生：

- HSE 外部晶振/陶瓷谐振器
- HSE 用户外部时钟

为了减少时钟输出的失真和缩短启动稳定时间，晶体/陶瓷谐振器和负载电容器必须尽可能地靠近振荡器引脚。负载电容值必须根据所选择的振荡器来调整。

外部源（HSE 旁路）

在这个模式里，必须提供外部时钟。它的频率最高可达 30MHz。用户可通过设置在时钟控制寄存器（RCC_CR）中的 HSEBYP 和 HSEON 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 OSC_IN 引脚，同时 OSC_OUT 可作为 GPIO 口使用。

外部晶振/陶瓷谐振器（HSE 晶振）

支持 4 ~32 MHz 外部振荡器（HSE），其优点是精度非常高。

在时钟控制寄存器（RCC_CR）中的 HSERDY 位用于指示高速外部振荡器是否稳定。在启动时，直到该位被硬件置 1，才能使用该时钟。如果在时钟中断寄存器（RCC_CIR）中允许产生中断，将会产生相应中断。

通过配置时钟控制寄存器（RCC_CR）中的 HSEON 位，可打开或关闭 HSE 晶振。

5.2.2 HSI 时钟

HSI 时钟信号由 48 MHz 内部 RC 振荡器进行分频生成的 8M 时钟，可直接作为系统时钟或作为 PLL 输入。

在从复位重启后，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。它的启动时间比 HSE 晶体振荡器短。然而，即使在校准之后，其精度也低于外部晶振或陶瓷谐振器。

5.2.3 HSI12 时钟

HSI12 时钟信号是从 48 MHz 内部 RC 振荡器进行 4 分频生成的。该时钟信号可直接用作系统时钟。

HSI12 RC 振荡器的优点是成本较低（无需使用外部元件）。此外，其启动速度也要比 HSE 晶振快，但即使校准后，其精度也低于外部晶振或陶瓷谐振器。

5.2.4 HSI48 时钟

HSI48 时钟信号由内部 48 MHz RC 振荡器产生，可直接使用。

HSI48 要求使能与 48 MHz RC 相关的 V_{REFINT} 及其缓冲区。

时钟控制寄存器 2（RCC_CR2）中的 HSI48RDY 位指示 HSI48 RC 是否稳定。在启动时，需等待硬件将该位置 1 后，HSI48 才可以使用。

HSI48 可通过时钟控制寄存器 2（RCC_CR2）中的 HSI48ON 位打开或关闭。

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 1%（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（RCC_CR）的 HSIFTRIM[5:0]。

如果用户的应用基于不同的电压或环境温度，这将会影响 RC 振荡器的精度。可以通过时钟控制寄存器（RCC_CR）里的 HSICTRIM[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。

如果 HSE 晶体振荡器失效，HSI 时钟会被作为备用时钟源。参见“5.2.10 HSE 时钟安全系统（CSSHSE）”。

5.2.5 PLL 时钟

内部 PLL 可以用来倍频 HSI 时钟或 HSE 晶体输出时钟，参见图 5-2 和“5.3.1 时钟控制寄存器（RCC_CR）”。

在使能 PLL 前，必须先配置 PLL（选择输入时钟，倍频因子等）。一旦使能 PLL，PLL 使用到的这些参数就不能被改变。

配置 PLL 参数的过程如下：

1. 设置 PLLON = 0，禁用 PLL。
2. 等待 PLLRDY 清零，PLLRDY 清零时，才表明 PLL 已经完全停止。
3. 改变 PLL 所需参数。

4. 设置 PLLON = 1，以重新使能 PLL。

5. 等待 PLLRDY 置 1。

当使能时钟中断寄存器（RCC_CIR）的相应位，当 PLL 时钟就绪时会产生一个中断。PLL 输出频率设置的范围是 1~48 MHz。

5.2.6 LSE 时钟

LSE 晶振是 32.768 kHz 低速外部晶振或陶瓷谐振器。LSE 具有功耗低且精度高的优点。

通过配置 LSE 控制寄存器（RCC_LSECTLR）中的 LSEON 位，可打开或关闭 LSE 晶振。

LSE 控制寄存器（RCC_LSECTLR）中的 LSERDY 标志指示 LSE 晶振是否稳定。在启动时，需等待硬件将该位置 1 后，才能使用 LSE 晶振输出时钟信号。如果在时钟中断寄存器（RCC_CIR）中使能中断，则可产生中断。

外部源（LSE 旁路）

在此模式下，必须提供外部时钟源。最高频率不超过 32 MHz。通过将 LSE 控制寄存器（RCC_LSECTLR）中的 LSEBYP 和 LSEON 位置 1 来选择此模式。必须使用占空比约为 50%的外部时钟信号（方波、正弦波或三角波）来驱动 OSC32_IN 引脚，同时 OSC32_OUT 引脚可作为 GPIO 口使用。

5.2.7 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 40 kHz 左右。

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。在启动时，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。

从 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（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准确的且精度水平可接受的 IWDG 超时（当 LSI 用作这些外设的时钟源时）。

5.2.8 EXTCLK 外部时钟

在这个模式里，必须提供外部时钟。用户可以通过外部 GPIO 输入最高 30MHz 时钟，并通过配置 RCC_CFGR4.ESW[2:0]作为 SYSCLK 使用。通过 RCC_CFGR4.EXTCLK_SEL[1:0]选择作为外部时钟输入的 IO。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。

5.2.9 系统时钟（SYSCLK）选择

可以使用 8 种不同的时钟源来驱动系统时钟（SYSCLK）：

- HSE 振荡器
- LSE 振荡器
- LSI 振荡器
- HSI48 振荡器
- HSI12 振荡器
- HSI 振荡器
- PLLCLK

- GPIO 外部输入时钟（EXTCLK）

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。

时钟的切换只有在目标时钟源可用的情况下才能进行。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）指示当前系统时钟采用哪个时钟源作为系统时钟。

5.2.10 HSE 时钟安全系统（CSSHSE）

时钟安全系统可以通过软件被激活。可通过写入时钟控制寄存器（RCC_CR）中的 CSSON 位来完成此操作。一旦其被激活，时钟监测器将在 HSE 振荡器启动延迟后被使能，并在 HSE 时钟关闭后禁止。

HSE 时钟使能（HSEON 置 1）并就绪（硬件将 HSERDY 置 1）后，如果写入 CSSON 位来使能 CSSHSE，当 HSE 时钟起振偏离阈值后，则判断 HSE 出现故障，RCC_CIR.CSSHSEF 置 1，HSE 振荡器自动被关闭，并且会产生 CSSHSE 中断信号（时钟安全系统中断）以通知软件 HSE 振荡器发生故障，从而使 MCU 执行救援操作。CSSHSE 信号与 Cortex®-M0+NMI（不可屏蔽中断）异常向量相连接，然后 NMI 中断就会被挂起。

注意：一旦 CSS 被激活，并且 HSE 时钟出现故障，CSSHSE 中断信号就会产生，并且 NMI 也自动产生。NMI 将被不断执行，直到 CSSHSE 中断挂起位被清除。因此，在 NMI 的处理程序中必须通过设置时钟中断寄存器（RCC_CIR）里的 CSSHSEC 位来清除 CSSHSE 中断。

如果直接或间接使用 HSE 振荡器作为系统时钟（间接是指它用作 PLL 输入时钟，PLL 时钟用作系统时钟），检测到故障时会导致系统时钟切换并禁止 HSE 振荡器。当故障发生时，如果 HSE 振荡器时钟正用作系统时钟的 PLL 时钟输入，该 PLL 也会被禁用。

支持 HSE 频率偏离故障检测功能，当开启 CSSHSE 功能后，如果 HSE 频率超过了 RCC_HSECNT 设定阈值时，则判断 HSE 出现异常，RCC_CIR.CSSHSEF 会被硬件置 1。详细请参考章节：“5.3.15 HSE 时钟偏离计数寄存器（RCC_HSECNT）”。

5.2.11 LSE 时钟安全系统（CSSLSE）

时钟安全系统可通过软件在 LSE 上激活。可通过写入时钟控制寄存器（RCC_CR）中的 CSSON 位来完成此操作。可通过硬件复位或在检测到 LSE 时钟故障后禁止该位。

LSE 时钟使能（LSEON 置 1）并就绪（硬件将 LSERDY 置 1）后，如果写入 CSSON 位来使能 CSSLSE，当 LSE 时钟起振偏离阈值后，则判断 LSE 出现问题，RCC_CIR.CSSLSEF 置 1；如果配置了中断寄存器（RCC_CIR）中的 LSEFAILIE 位则触发 LSE 故障中断，RCC 中断会被挂起。

CSSLSE 适用于所有工作模式：运行、睡眠、停机。

如果在外部 32.768kHz 振荡器上检测到故障，则 LSE 时钟会停止，但寄存器的内容保持不变。

支持 LSE 频率偏离故障检测功能，当开启 CSSLSE 功能后，如果 LSE 频率超过了 RCC_LSECNT 设定阈值时，则判断 LSE 出现异常，RCC_CIR.CSSLSEF 会被硬件置 1。详细请参考章节：“5.3.16 LSE 时钟偏离计数寄存器（RCC_LSECNT）”。

5.2.12 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。

5.2.13 时钟输出功能（MCO）

微控制器时钟输出（Microcontroller clock output，MCO）功能允许使用可配置的预分频值

（1/2/4/8/16/.../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 7 种时钟信号之一作为 MCO 时钟：

- HSI12
- LSI
- LSE
- SYSCLK
- HSI
- HSE
- PLLCLK

时钟信号选择由时钟配置寄存器（RCC_CFGR）中的 MCO[2:0]位控制。

5.3 RCC 寄存器

基地址：0x4002 1000

空间大小：0x400

5.3.1 时钟控制寄存器（RCC_CR）

偏移地址：0x00

复位值：0x3580 248B

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						PLLRDY	PLLON	Res				CSSON	HSEBYP	HSERDY	HSEON
						r	rw					rw	rw	r	rw

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

位 31:26	Res：保留 必须保持复位值。
位 25	PLLRDY：PLL 时钟就绪标志（PLL clock ready flag） 该位由硬件置 1，用于指示 PLL 已锁定。 ● 0：PLL 未锁定 ● 1：PLL 已锁定
位 24	PLLON：PLL 使能位（PLL enable） 此位由软件置位和清零，以使能 PLL。 当进入停机模式时，该位由硬件清零。如果 PLL 时钟用作系统时钟或被选作系统时钟，该位不能复位。 ● 0：PLL 关闭 ● 1：PLL 开启
位 23:20	Res：保留 必须保持复位值。
位 19	CSSON：时钟安全系统使能（Clock security system enable） 该位由软件设置或清零。 ● 0：时钟检测器关闭 ● 1：时钟检测器打开（假如 HSE/LSE 就绪，时钟检测器则打开。若未就绪，则关闭）。

位 18	HSEBYP：外部高速时钟旁路（HSE crystal oscillator bypass） 该位由软件设置或清零。外部时钟必须通过 HSEON=1 打开，HSEBYP 位只能在 HSE 振荡器关闭的情况下使用。 ● 0：HSE 晶体振荡器无旁路 ● 1：HSE 晶体振荡器旁路
位 17	HSERDY：HSE 时钟就绪标志（HSE clock ready flag） 该位由硬件设置，表明 HSE 振荡器是否稳定。当 HSEON 清零后，该位需要 1 个 HSE 振荡器周期才清零。 ● 0：HSE 振荡器未就绪 ● 1：HSE 振荡器就绪
位 16	HSEON：HSE 时钟使能（HSE clock enable） 该位由软件设置和清零。 当进入停机模式后，该位由硬件清除并停止 HSE 振荡器。当直接或间接使用 HSE 时钟时，该位不能被清零。 ● 0：HSE 振荡器关闭 ● 1：HSE 振荡器开启
位 15:14	Res：保留 必须保持复位值。
位 13:8	HSICTRIM[5:0]：HSI 时钟粗校准（HSI clock coarse frequency trimming） 在启动时，该位域被自动初始化为出厂值校准值。
位 7:2	HSIFTRIM[5:0]：HSI 时钟精校准（HSI clock fine frequency trimming） 在启动时，该位域被自动初始化为出厂值校准值。
位 1	HSIRDY：HSI 时钟就绪标志（HSI clock ready flag） 该位由硬件置 1 来指示内部 HSI 振荡器已经稳定。在 HSION 位清零后，HSIRDY 位需 1 个 HSI 振荡器周期清零。 ● 0：HSI 振荡器未就绪 ● 1：HSI 振荡器就绪
位 0	HSION：HSI 时钟使能（HSI clock enable） 该位由软件设置和清零。 当从停机模式返回或用作系统时钟的 HSE 振荡器发生故障时，该位由硬件置 1 来启动 HSI 振荡器。当 HSI 振荡器被直接或间接地用作或被选择将要作为系统时钟时，该位不能被清零。 ● 0：HSI 振荡器关闭 ● 1：HSI 振荡器开启

5.3.2 时钟配置寄存器（RCC_CFGR）

偏移地址：0x04

复位值：0x0080 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	MCOPRE[2:0]			MCO[2:0]			PLLWT[4:0]					PLLMUL[3:0]			
	rw			rw			rw					rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLLXTPRE	PLLSRC	Res			PPRE1[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
rw	rw				rw			rw				r		rw	

位 31	Res：保留 必须保持复位值。
位 30:28	MCOPRE[2:0]：微控制器时钟输出分频系数（Microcontroller Clock Output Prescaler） 该位由软件设置选择 MCO 分频因子。推荐在 MCO 输出关闭时进行设置。 ● 000：MCO/1 ● 001：MCO/2 ● 010：MCO/4 ● 011：MCO/8 ● 100：MCO/16 ● 101：MCO/32 ● 110：MCO/64 ● 111：MCO/128
位 27:25	MCO[2:0]：微控制器时钟输出选择（Microcontroller clock output） 该位由软件设置和清除。 ● 000：时钟输出禁止，MCO 引脚上没有时钟输出 ● 001：选择内部 HSI12 时钟输出 ● 010：选择内部 LSI 时钟输出 ● 011：选择外部 LSE 时钟输出 ● 100：系统时钟（SYSCLK） ● 101：选择内部 HSI 时钟输出 ● 110：选择外部 HSE 时钟输出 ● 111：选择 PLL 时钟输出 *注意：该时钟输出在启动或 MCO 时钟源切换期间可能有一些截短的周期。*
位 24:20	PLLWT[4:0]：设置 PLL 稳定等待时间（PLL stabilize wait time setting） PLL 稳定等待时间为 PLLWT x 1024 个 PLL 时钟脉冲。
位 19:16	PLLMUL[3:0]：PLL 倍频系数（PLL multiplication factor） 由软件设置来确定 PLL 倍频系数。该位域只能在 PLL 关闭的情况下才可被写入。 ● 0000：PLL 输入时钟的 2.5 倍频 ● 0001：PLL 输入时钟的 3 倍频 ● 0010：PLL 输入时钟的 4 倍频 ● 0011：PLL 输入时钟的 5 倍频 ● 0100：PLL 输入时钟的 6 倍频 ● 0101：PLL 输入时钟的 7 倍频 ● 0110：PLL 输入时钟的 8 倍频 ● 0111：PLL 输入时钟的 9 倍频 ● 1000：PLL 输入时钟的 10 倍频 ● 1001：PLL 输入时钟的 11 倍频 ● 1010：PLL 输入时钟的 12 倍频 ● 1011：PLL 输入时钟的 13 倍频 ● 1100：PLL 输入时钟的 14 倍频

	● 1101：PLL 输入时钟的 15 倍频 ● 1110：PLL 输入时钟的 16 倍频 ● 1111：PLL 输入时钟的 16 倍频
位 15	PLLXTPRE：PLL 输入时钟 HSE 分频器（HSE divider for PLL input clock） PLLXTPRE 与 RCC_CFGR2.PLLPREDIV[0]意义相同。
位 14	PLLSRC：PLL 输入时钟源（PLL input clock source） 由软件置'1'或清'0'来选择 PLL 输入时钟源。仅在关闭 PLL 时，才能写入此位。 ● 0：HSI 作为 PLL 输入时钟 ● 1：RCC_CFGR4.PPSS 选择的时钟作为 PLL 输入时钟
位 13:11	Res：保留 必须保持复位值。
位 10:8	PPRE1[2:0]：APB1 预分频器（APB1PCLK prescaler） 该位域由软件置位和清零，用于控制 APB1 时钟的分频系数。 ● 0xx：HCLK ● 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 factor） 由软件置位和清零来控制 AHB 时钟的分频系数。 ● 0xxx：SYSCLK ● 1000：SYSCLK/2 ● 1001：SYSCLK/4 ● 1010：SYSCLK/8 ● 1011：SYSCLK/16 ● 1100：SYSCLK/64 ● 1101：SYSCLK/128 ● 1110：SYSCLK/256 ● 1111：SYSCLK/512
位 3:2	SWS[1:0]：系统时钟切换状态（System clock switch status） 由硬件置位或清零，结合 RCC_CFGR4.ESWS 以指示哪一个时钟源被作为系统时钟。当 RCC_CFGR4.ESSS 为 0 时，系统时钟状态如下： ● 00：HSI 用作系统时钟 ● 01：HSE 用作系统时钟 ● 10：PLL 输出用作系统时钟 ● 11：保留
位 1:0	SW[1:0]：系统时钟切换（System clock switch） 由软件置位或清零，结合 RCC_CFGR4.ESW 以选择系统时钟源。在从停机模式中返回时或直接或间接作为系统时钟的 HSE 出现故障时，由硬件强制选择 HSI 作为系统时钟（如果时钟安全系统已经启动）。当 RCC_CFGR4.ESSS 为 0 时，系统时钟源如下： ● 00：HSI 用作系统时钟 ● 01：HSE 用作系统时钟

	● 10：PLL 输出用作系统时钟 ● 11：保留

5.3.3 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
CSSLSEC	CSSLSEF	Res						CSSHSEC	Res	HSI12RDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
w	r							w		w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSEFAILIE	Res	HSI12RDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSHSEF	Res	HSI12RDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
rw		rw	rw	rw	rw	rw	rw	r		r	r	r	r	r	r

位 31	CSSLSEC：时钟安全系统 LSE 中断清零（Clock security system LSE interrupt flag clear） 由软件置 1，用于将 CSSLSEF 标志位清零。该位由硬件清零。 ● 0：不起作用 ● 1：将 CSSLSEF 标志清零
位 30	CSSLSEF：CSS 模块检测 LSE 故障（Clock security system fail） 当 LSE 时钟出现故障且 LSEFAILIE 使能，由硬件置 1；软件在 CSSLSEC 写入 1 将该位清零。 ● 1：CSS 模块检测到 LSE 故障 ● 0：CSS 模块未检测到 LSE 故障
位 29:24	Res：保留 必须保持复位值。
位 23	CSSHSEC：时钟安全系统 HSE 中断清零（Clock security system HSE interrupt flag clear） 该位由软件置 1，用于将 CSSHSEF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 CSSHSEF 标志清零
位 22	Res：保留 必须保持复位值。
位 21	HSI12RDYC：HSI12 就绪中断清零（HSI12 ready Interrupt clear） 该位由软件置 1，用于将 HSI12RDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 HSI12RDYF 标志清零
位 20	PLLRDYC：PLL 就绪中断清零（PLL ready interrupt clear） 该位由软件置 1，用于将 PLLRDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 PLLRDYF 标志清零
位 19	HSERDYC：HSE 就绪中断清零（HSE ready interrupt clear）

	该位由软件置 1，用于将 HSERDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 HSERDYF 标志清零
位 18	HSIRDYC：HSI 就绪中断清零（HSI ready interrupt clear） 该位由软件置 1，用于将 HSIRDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 HSIRDYF 标志清零
位 17	LSERDYC：LSE 就绪中断清零（LSE ready interrupt clear） 该位由软件置 1，用于将 LSERDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 LSERDYF 标志清零
位 16	LSIRDYC：LSI 就绪中断清零（LSI ready interrupt clear） 该位由软件置 1，用于将 LSIRDYF 标志清零。该位由硬件复位。 ● 0：不起作用 ● 1：将 LSIRDYF 标志清零
位 15	LSEFAILIE：LSE 错误中断使能（LSE fail interrupt enable） ● 0：不起作用 ● 1：使能 LSE 故障中断。当 LSEFAILIE 为 1，并且 LSE 故障后，产生 RCC 中断。
位 14	Res：保留 必须保持复位值。
位 13	HSI12RDYIE：HSI12 就绪中断使能（HSI12 ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 HSI12 振荡器稳定所引起的中断。 ● 0：禁止 HSI12 就绪中断 ● 1：使能 HSI12 就绪中断
位 12	PLLRDYIE：PLL 就绪中断使能（PLL ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 PLL 锁定引起的中断。 ● 0：禁止 PLL 锁定中断 ● 1：使能 PLL 锁定中断
位 11	HSERDYIE：HSE 就绪中断使能（HSE ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 HSE 振荡器稳定所引起的中断。 ● 0：禁止 HSE 就绪中断 ● 1：使能 HSE 就绪中断
位 10	HSIRDYIE：HSI 就绪中断使能（HSI ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 HSI 振荡器稳定所引起的中断。 ● 0：禁止 HSI 就绪中断 ● 1：使能 HSI 就绪中断
位 9	LSERDYIE：LSE 就绪中断使能（LSE ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 LSE 振荡器稳定所引起的中断。 ● 0：禁止 LSE 就绪中断 ● 1：使能 LSE 就绪中断

位 8	LSIRDYIE：LSI 就绪中断使能（LSI ready interrupt enable） 该位由软件置 1 和复位，用于使能/禁止由 LSI 振荡器稳定所引起的中断。 ● 0：禁止 LSI 就绪中断 ● 1：使能 LSI 就绪中断
位 7	CSSHSEF：时钟安全系统 HSE 中断标志（Clock security system HSE interrupt flag） 当系统检测到 HSE 振荡器错误时，由硬件置位该位。软件对 CSSHSEC 置 1 时清除该位。 ● 0：当前未因 HSE 时钟故障而引起时钟安全中断 ● 1：因 HSE 时钟故障而引起时钟安全中断
位 6	Res：保留 必须保持复位值。
位 5	HSI12RDYF：HSI12 就绪中断标志（HSI12 ready interrupt flag） 当 HSI12 时钟趋于稳定、HSI12RDYIE=1 且 HSI12ON =1（RCC_CR2）时，由硬件对该位置 1。 当 HSI12ON =0，无论 HSI12 是否已稳定都不会置位 HSI12RDYF。软件置 HSI12RDYC=1 时，该位清除。 ● 0：当前未因 HSI12 时钟故障而引起时钟就绪中断 ● 1：因 HSI12 时钟故障而引起时钟就绪中断
位 4	PLLRDYF：PLL 就绪中断标志（PLL ready interrupt flag） 当 PLL 时钟就绪且 PLLRDYIE=1 时，由硬件对该位置 1。软件置 PLLRDYC=1 时，该位被清除。 ● 0：当前未因 PLL 时钟故障而引起时钟就绪中断 ● 1：因 PLL 时钟故障而引起时钟就绪中断
位 3	HSERDYF：HSE 就绪中断标志（HSE ready interrupt flag） 该位通过软件写入 HSERDYC 位复位。当 HSE 时钟稳定且 HSERDYIE 置 1 时，该位由硬件置 1。 ● 0：当前未因 HSE 时钟故障而引起时钟就绪中断 ● 1：因 HSE 时钟故障而引起时钟就绪中断
位 2	HSIRDYF：HSI 就绪中断标志（HSI ready interrupt flag） 当 HSI 时钟趋于稳定、HSIRDYIE=1 且 HSION=1（RCC_CR）时，硬件对该位置 1。当 HSION=0，无论 HSI 是否已稳定都不会置位 HSIRDYF。软件置 HSIRDYC=1 时，该位被清除。 ● 0：当前未因 HSI 时钟故障而引起时钟就绪中断 ● 1：因 HSI 时钟故障而引起时钟就绪中断
位 1	LSERDYF：LSE 就绪中断标志（LSE ready interrupt flag） 当 LSE 时钟就绪且 LSERDYIE=1 时，由硬件对该位置 1。软件置 LSERDYC=1 时，该位被清除。 ● 0：当前未因 LSE 时钟故障而引起时钟就绪中断 ● 1：因 LSE 时钟故障而引起时钟就绪中断
位 0	LSIRDYF：LSI 就绪中断标志（LSI ready interrupt flag） 当 LSI 时钟就绪且 LSIRDYDIE=1 时，硬件对该位置 1。软件置 LSIRDYC=1 时，该位被清除。 ● 0：当前未因 LSI 时钟故障而引起时钟就绪中断 ● 1：因 LSI 时钟故障而引起时钟就绪中断

5.3.4 APB2 外设复位寄存器（RCC_APB2RSTR）

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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		I2C1RST	COMPRST	OPAMPRST	Res	DBGRST	Res	TIM3RST	Res		TIM2RST	Res	TIM1RST	UART2RST	SYSCFGRST
		rw	rw	rw		rw		rw			rw		rw	rw	rw

位 31:14	Res：保留 必须保持复位值。
位 13	I2C1RST：I2C1 模块复位（Reset I2C1） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 I2C1 模块
位 12	COMPRST：将电压比较器控制模块复位（Reset comparator） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位电压比较器控制模块
位 11	OPAMPRST：OPAMP 模块复位（Reset OPAMP） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 OPAMP 模块
位 10	Res：保留 必须保持复位值。
位 9	DBGRST：将 DBG 模块复位（Reset DBG） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 DBG
位 8	Res：保留 必须保持复位值。
位 7	TIM3RST：将 TIM3 定时器复位（Reset TIM3） 由软件置位或清零。 ● 0：不起作用 ● 1：复位 TIM3 定时器
位 6:5	Res：保留 必须保持复位值。
位 4	TIM2RST：将 TIM2 定时器复位（Reset TIM2） 由软件置位或清零。 ● 0：不起作用 ● 1：复位 TIM2 定时器

位 3	Res：保留 必须保持复位值。
位 2	TIM1RST：将 TIM1RST 复位（Reset TIM1RST） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 TIM1RST
位 1	UART2RST：将 UART2 复位（Reset UART2） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 UART2
位 0	SYSCFGRST：将 SYSCFG 复位（Reset SYSCFG） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 SYSCFG

5.3.5 APB1 外设复位寄存器（RCC_APB1RSTR）

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					PWRRST	Res									
					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI1RST	Res		WWDGRST	Res						UART1RST	TIM6RST	RTCRST	Res		ADCRST
rw			rw							rw	rw	rw			rw

位 30:27	Res：保留 必须保持复位值。
位 26	PWRRST：电源接口复位（Reset power interface） ● 0：不起作用 ● 1：复位电源接口
位 25:16	Res：保留 必须保持复位值。
位 15	SPI1RST：SPI1 接口复位（Reset SPI1） 由软件置位或清零。 ● 0：不起作用 ● 1：复位 SPI1 接口
位 14:13	Res：保留 必须保持复位值。
位 12	WWDGRST：窗口看门狗复位（Reset WWDG） 由软件置位或清零

	● 0：不起作用 ● 1：复位窗口看门狗
位 11:6	Res：保留 必须保持复位值。
位 5	UART1RST：UART1 复位（Reset UART1） 由软件置位或清零。 ● 0：不起作用 ● 1：复位 UART1
位 4	TIM6RST：TIM6 复位（Reset TIM6） ● 0：不起作用 ● 1：复位 TIM6 定时器
位 3	RTCRST：RTC 复位（Reset RTC） 此位由软件置位和清零。 ● 0：不起作用 ● 1：复位 RTC
位 2:1	Res：保留 必须保持复位值。
位 0	ADCRST：将 ADC 复位（Reset ADC） 此位由软件置位和清零。 ● 0：不起作用 ● 1：复位 ADC

5.3.6 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												IOPCEN	IOPBEN	IOPAEN	Res
												rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		TRACE_UART_PREDIV[3:0]				EMACCEN	DVSQEN	Res			FLITFEN	Res	SRAMEN	Res	DMA1EN
		rw				rw	rw				rw		rw		rw

位 31:20	Res：保留 必须保持复位值。
位 19	IOPCEN：GPIOC 时钟使能（I/O port C clock enable） 该位由软件置位或清零。 ● 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） 该位由软件置位或清零。

	● 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） 该位由软件置位或清零。 ● 0：GPIOA 时钟关闭 ● 1：GPIOA 时钟开启
位 16:14	Res：保留 必须保持复位值。
位 13:10	TRACE_UART_PREDIV[3:0]: TRACE_UART 时钟分频系数（TRACE_UART clock PREDIV） ● 0000：HCLK ● 0001：HCLK/2 ● 0010：HCLK/3 ● 0011：HCLK/4 ● 0100：HCLK/5 ● 0101：HCLK/6 ● 0110：HCLK/7 ● 0111：HCLK/8 ● 1000：HCLK/9 ● 1001：HCLK/10 ● 1010：HCLK/11 ● 1011：HCLK/12 ● 1100：HCLK/13 ● 1101：HCLK/14 ● 1110：HCLK/15 ● 1111：HCLK/16
位 9	EMACCEN：EMACC 时钟使能（EMACC clock enable） 由软件置位或清零。 ● 0：EMACC 时钟关闭 ● 1：EMACC 时钟开启
位 8	DVSQEN：DVSQ 时钟使能（DVSQ clock enable） 由软件置位或清零。 ● 0：DVSQ 时钟关闭 ● 1：DVSQ 时钟开启
位 7:5	Res：保留 必须保持复位值。
位 4	FLITFEN：FLITF 时钟使能（FLASH interface clock enable） 该位由软件置位或清零来开启/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） ● 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	Res：保留 必须保持复位值。
位 0	DMA1EN：DMA1 时钟使能位（DMA1 clock enable） 该位由软件置 1 和复位。 ● 0：禁能 DMA1 时钟 ● 1：使能 DMA1 时钟

5.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR）

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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		I2C1EN	COMPEN	OPAMPEN	Res	DBGEN	Res	TIM3EN	Res		TIM2EN	Res	TIM1EN	UART2EN	SYSCFGEN
		rw	rw	rw		rw		rw			rw		rw	rw	rw

位 31:14	Res：保留 必须保持复位值。
位 13	I2C1EN：I2C1 时钟使能（I2C1 clock enable） 该位由软件置位和清零。 ● 0：I2C1 时钟关闭 ● 1：I2C1 时钟开启
位 12	COMPEN：COMP 时钟使能（COMP clock enable） 该位由软件置位和清零。 ● 0：COMP 时钟关闭 ● 1：COMP 时钟开启
位 11	OPAMPEN：OPAMP 时钟使能（OPAMP clock enable） 该位由软件置位和清零。 ● 0：OPAMP 时钟关闭 ● 1：OPAMP 时钟开启
位 10	Res：保留 必须保持复位值。
位 9	DBGEN：DBG 时钟使能（DBG clock enable） 该位由软件置位和清零。 ● 0：DBG 时钟关闭

	● 1：DBG 时钟开启
位 8	Res：保留 必须保持复位值。
位 7	TIM3EN：TIM3 定时器时钟使能（TIM3 clock enable） 该位由软件置位和清零。 ● 0：TIM3 定时器时钟关闭 ● 1：TIM3 定时器时钟开启
位 6:5	Res：保留 必须保持复位值。
位 4	TIM2EN：TIM2 定时器时钟使能（TIM2 clock enable） 该位由软件置位和清零。 ● 0：TIM2 定时器时钟关闭 ● 1：TIM2 定时器时钟开启
位 3	Res：保留 必须保持复位值。
位 2	TIM1EN：TIM1EN 时钟使能（TIM1EN clock enable） 该位由软件置位和清零。 ● 0：TIM1 定时器时钟关闭 ● 1：TIM1 定时器时钟开启
位 1	UART2EN：UART2 时钟使能（UART2 clock enable） 该位由软件置位和清零。 ● 0：UART2 时钟关闭 ● 1：UART2 时钟开启
位 0	SYSCFGEN：系统配置控制器时钟使能（System control configuration clock enable） 该位由软件置位和清零。 ● 0：SYSCFG 时钟关闭 ● 1：SYSCFG 时钟开启

5.3.8 APB1 外设时钟使能寄存器（RCC_APB1ENR）

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 APB1 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					PWREN	Res									
					rw										

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI1EN	Res		WWDGEN	Res						UART1EN	TIM6EN	RTCPCLKEN	Res		ADCEN
rw			rw							rw	rw	rw			rw

位 31:27	Res：保留 必须保持复位值。

位 26	PWREN：电源接口时钟使能位（Power interface clock enable） 此位由软件置位和清零。 ● 0：禁能电源接口时钟 ● 1：使能电源接口时钟
位 25:16	Res：保留 必须保持复位值。
位 15	SPI1EN：SPI1 时钟使能位（SPI1 clock enable） 此位由软件置位和清零。 ● 0：禁能 SPI1 时钟 ● 1：使能 SPI1 时钟
位 14:13	Res：保留 必须保持复位值。
位 12	WWDGEN：窗口看门狗时钟使能位（WWDG clock enable） 该位由软件置位和清零。 ● 0：禁止窗口看门狗时钟 ● 1：使能窗口看门狗时钟
位 11:6	Res：保留 必须保持复位值。
位 5	UART1EN：UART1 时钟使能（UART1 clock enable） 该位由软件置位和清零。 ● 0：禁止 UART1 时钟关闭 ● 1：使能 UART1 时钟开启
位 4	● TIM6EN：TIM6 时钟使能位（TIM6 clock enable） ● 0：禁止 TIM6 时钟关闭 ● 1：使能 TIM6 时钟开启
位 3	● RTCPCLKEN：RTC 时钟使能位（TIM6 clock enable） ● 0：禁止 RTC 时钟关闭 ● 1：使能 RTC 时钟开启
位 2:1	Res：保留 必须保持复位值。
位 0	ADCEN：ADC 时钟使能位（ADC clock enable） 此位由软件置位和清零。 ● 0：禁能 ADC 时钟 ● 1：使能 ADC 时钟

5.3.9 LSE 控制寄存器（RCC_LSECTLR）

偏移地址：0x20

复位值：0x0000 CE00

访问：支持字、半字和字节；0 到 3 个等待周期；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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									LSEWT[2:0]			LSERST	LSEBYP	LSERDY	LSEON
									rw			rw	rw	r	rw

位 31:7	Res：保留 必须保持复位值。
位 6:4	LSEWT[2:0]：LSE 时钟稳定等待时间（LSE stabilize wait time setting） ● 000：无等待时间 ● 001：256 时钟周期（大约 7.81ms） ● 010：1024 时钟周期（大约 32.25ms） ● 011：4096 时钟周期（大约 125ms） ● 100：8192 时钟周期（大约 250ms） ● 101：16384 时钟周期（大约 500ms） ● 110：32768 时钟周期（大约 1s） ● 111：65536 时钟周期（大约 2s）
位 3	LSERST：复位 LSE 控制位（Reset LSE） 该位由软件置位和清零。 ● 0：复位未激活 ● 1：复位 LSE 控制位
位 2	LSEBYP：外部低速振荡器旁路位（LSE oscillator bypass） 由软件置 1 和清零，用于在调试模式下旁路振荡器。 通过上电复位或 LSERST 置 1 来复位。 ● 0：不旁路 LSE 振荡器 ● 1：旁路 LSE 振荡器
位 1	LSERDY：外部低速振荡器就绪位（LSE oscillator ready） 由硬件置位或清零来指示外部 32.768 kHz 振荡器是否就绪。 ● 0：外部 LSE 振荡器未就绪 ● 1：外部 LSE 振荡器已就绪
位 0	LSEON：外部低速振荡器使能位（LSE oscillator enable） 由软件置 1 和清零。通过上电复位或 LSERST 置 1 来复位。 ● 0：LSE 振荡器关闭 ● 1：LSE 振荡器开启

5.3.10 控制/状态寄存器（RCC_CSR）

偏移地址：0x24

复位值：0x0800 0080

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：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
LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLRSTF	RMVF	Res							
r	r	r	r	r	r	r	rt_w								

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

位 31	LPWRRSTF：低功耗复位标志（Low-power reset flag） 在低功耗管理复位发生时，该位由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：无低功耗管理复位发生 ● 1：发生低功耗管理复位
位 30	WWDGRSTF：窗口看门狗复位标志（Window watchdog reset flag） 在窗口看门狗复位发生时，该位由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：无窗口看门狗复位发生 ● 1：发生窗口看门狗复位
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。 该位由软件写 RMVF 位清除。 ● 0：无看门狗复位发生 ● 1：发生看门狗复位
位 28	SFTRSTF：软件复位标志（Software reset flag） 在软件复位发生时，该位由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：无软件复位发生 ● 1：发生软件复位
位 27	PORRSTF：上电/掉电复位标志（POR/PDR reset flag） 在上电/掉电复位发生时，该位由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：无上电/掉电复位发生 ● 1：发生上电/掉电复位
位 26	PINRSTF：NRST 引脚复位标志（NRST pin reset flag） 在 NRST 引脚复位发生时，该位由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：无 NRST 引脚复位发生 ● 1：发生 NRST 引脚复位
位 25	OBLRSTF：选项字节装载复位标志（Option byte loader reset flag） 在选项字节装载器装载选项字节时，由硬件置 1。 该位由软件写 RMVF 位清除。 ● 0：未发生 OBL 复位 ● 1：发生 OBL 复位

位 24	RMVF：清除复位标志（Remove reset flag） 该位由软件置 1 来清除复位标志。 ● 0：不起作用 ● 1：清除复位标志
位 23:8	Res：保留 ● 必须保持复位值。
位 7:2	LSIFREQSEL[5:0]：LSI 输出频率选择（LSI output frequency selection） 该位由软件置位或清零。 ● 010000：输出 32KHz ● 100000：输出 40KHz ● 110000：输出 64KHz ● 111011：输出 128KHz
位 1	LSIRDY：LSI 振荡器就绪（LSI oscillator ready） 通过硬件置位或清零来指示内部 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） 该位由软件置位或清零。 ● 0：LSI 振荡器关闭 ● 1：LSI 振荡器开启

5.3.11 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												IOPCRST	IOPBRST	IOPARST	Res
												rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						EMACCRST	DVSQRST	Res							DMA1RST
						rw	rw								rw

位 31:20	Res：保留 必须保持复位值。
位 19	IOPCRST：将 GPIOC 口复位（Reset I/O port C） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 GPIOC 口
位 18	IOPBRST：将 GPIOB 口复位（Reset I/O port B） 该位由软件置位或清零。 ● 0：不起作用

	● 1：复位 GPIOB 口
位 17	IOPARST：将 GPIOA 口复位（Reset I/O port A） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 GPIOA 口
位 16:10	Res：保留 必须保持复位值。
位 9	EMACCRST：将 EMACC 模块复位（Reset EMACC） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 EMACC
位 8	DVSQRST：将 DVSQ 模块复位（Reset DVSQ） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 DVSQ
位 7:1	Res：保留 必须保持复位值。
位 0	DMA1RST：将 DMA1 复位（Reset DMA1） 该位由软件置位或清零。 ● 0：不起作用 ● 1：复位 DMA1

5.3.12 时钟配置寄存器 2（RCC_CFGR2）

偏移地址：0x2C

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															

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

位 31:4	Res：保留 必须保持复位值。
位 3:0	PLLPREDIV[3:0]：PLL PREDIV 分频因子（PLL PREDIV division factor） 该位域仅能在 PLL 关闭后改写，用于设置或清除 PREDIV 分频因子。 注意：位 0 与 RCC_CFGR 的位 15 相同，修改 RCC_CFGR 的位 15 同时改变这里的位 0。 ● 0000：PREDIV 输入时钟，不分频 ● 0001：PREDIV 输入时钟 2 分频 ● 0010：PREDIV 输入时钟 3 分频 ● 0011：PREDIV 输入时钟 4 分频 ● 0100：PREDIV 输入时钟 5 分频

	• 0101：PREDIV 输入时钟 6 分频 • 0110：PREDIV 输入时钟 7 分频 • 0111：PREDIV 输入时钟 8 分频 • 1000：PREDIV 输入时钟 9 分频 • 1001：PREDIV 输入时钟 10 分频 • 1010：PREDIV 输入时钟 11 分频 • 1011：PREDIV 输入时钟 12 分频 • 1100：PREDIV 输入时钟 13 分频 • 1101：PREDIV 输入时钟 14 分频 • 1110：PREDIV 输入时钟 15 分频 • 1111：PREDIV 输入时钟 16 分频

5.3.13 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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												UART2SW[1:0]		UART1SW[1:0]	
												rw		rw	

位 31:4	Res：保留 必须保持复位值。
位 3:2	UART2SW[1:0]：UART2 时钟源选择（UART2 clock source selection） • 00：PCLK1 被选为 UART2 的时钟源 • 01：LSE 被选为 UART2 的时钟源 • 10：LSI 时钟被选为 UART2 的时钟源 • 11：HSI12 时钟被选为 UART2 的时钟源
位 1:0	UART1SW[1:0]：UART1 时钟源选择（UART1 clock source selection） • 00：PCLK1 被选为 UART1 的时钟源 • 01：LSE 被选为 UART1 的时钟源 • 10：LSI 时钟被选为 UART1 的时钟源 • 11：HSI12 时钟被选为 UART1 的时钟源

5.3.14 时钟控制寄存器 2（RCC_CR2）

偏移地址：0x34

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															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res											HSI48RDY	HSI48ON	HSI48DIS	HSI12RDY	HSI12ON
											r	rw	rw	r	rw

位 31:5	Res：保留 必须保持复位值。
位 4	HSI48RDY：HSI48MHz RC 振荡器时钟就绪标志（HSI48 clock ready flag） ● 0：HSI48 时钟未就绪 ● 1：HSI48 时钟已就绪
位 3	HSI48ON：打开芯片内部 HSI48MHz RC 振荡器（HSI48 oscillator enable） ● 0：关闭 HSI48 时钟 ● 1：打开 HSI48 时钟
位 2	HSI48DIS：ADC HSI48 时钟请求禁止（HSI48 clock request from ADC disable） 该位由软件置位或清零。 当置 1 时，禁止 ADC 接口打开 HSI48 振荡器。 ● 0：ADC 控制器可以打开 HSI48 时钟（这种情况下，ADC 可使用 HSI12） ● 1：ADC 控制器不能打开 HSI48 时钟（这种情况下，ADC 不可使用 HSI12，除非软件使能 HSI48ON）
位 1	HSI12RDY：HSI12 时钟就绪标志（HSI12 clock ready flag） 由硬件置 1 来指示内部 HSI12 振荡器已经稳定。在 HSI12 位清零后，HSI12 位需要等待 6 个 HSI12 振荡器周期再清零。 ● 0：HSI12 时钟未就绪 ● 1：HSI12 时钟已就绪
位 0	HSI12ON：HSI12 时钟使能（HSI12 clock enable） 该位由软件置位或清零。 ● 0：关闭 HSI12 时钟 ● 1：打开 HSI12 时钟

5.3.15 HSE 时钟偏离计数寄存器（RCC_HSECNT）

偏移地址：0x3C

复位值：0x6400 0100

访问：无等待周期，字、半字和字节访问（按字节操作禁止）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSE_CNT_UPPER_LIMIT[15:0]															
rw															

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

位 31:16	HSE_CNT_UPPER_LIMIT[15:0]：HSE 频率最大计数（HSE maximum frequency count） 最大 HSE 频率计数值，如果大于设定值，CSS 将生成 HSEFAIL。 HSE 的最小频率约为 8M*1024/HSE_CNT_UPPER_LIMIT，因此使用 HSE_CNT_UPPER_LIMIT 的默认值，HSE 的最小频率约为 8M*1024/25600≈320 K。 *注意：采样时钟为 HSI，此处其频率默认为 8MHz，请参考系统设置。* 该位由软件置位或清零。在 CSS 无效的条件下才能写入。
位 15:0	HSE_CNT_LOWER_LIMIT[15:0]：HSE 频率最小计数（HSE minimum frequency count） 最小 HSE 频率计数值，如果小于设定值，CSS 会生成 HSEFAIL。 HSE 的最大频率约为 8M*1024/HSE_CNT_LOWER_LIMIT，因此使用 HSE_CNT_LOWER_LIMIT 的默认值，HSE

	的最大频率为 8M*1024/256≈32M *注意：采样时钟为 HSI，此处其频率默认为 8MHz，请参考系统设置。* 该位由软件置位或清零。在 CSS 无效的条件下才能写入。

5.3.16 LSE 时钟偏离计数寄存器（RCC_LSECNT）

偏移地址：0x40

复位值：0x03D0 003D

访问：无等待周期，字、半字和字节访问（按字节操作禁止）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LSE_CNT_UPPER_LIMIT[15:0]															
rw															

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

位 31:16	LSE_CNT_UPPER_LIMIT[15:0]：LSE 频率最大计数（LSE maximum frequency count） 最大 LSE 频率计数值，如果大于设定值，CSS 将生成 LSEFAIL。 LSE 的最小频率约为 8M/LSE_CNT_UPPER_LIMIT，因此使用 LSE_CNT_UPPER_LIMIT 的默认值，LSE 的最小频率为 8M/976≈8.2 kHz。 *注意：采样时钟为 HSI，此处其频率默认为 8MHz，请参考系统设置。* 该位由软件置位或清零。在 CSS 无效的条件下才能写入。
位 15:0	LSE_CNT_LOWER_LIMIT[15:0]：LSE 频率最小计数（LSE minimum frequency count） 最小 LSE 频率计数值，如果小于设定值，CSS 将生成 LSEFAIL。 LSE 的最大频率约为 8M/LSE_CNT_LOWER_LIMIT，因此使用 LSE_CNT_LOWER_LIMIT 的默认值，LSE 的最大频率为 8M/61≈131kHz。 *注意：采样时钟为 HSI，此处其频率默认为 8MHz，请参考系统设置。* 该位由软件置位或清零。在 CSS 无效的条件下才能写入。

5.3.17 RCC HSE 时钟控制寄存器 （RCC_HSECTL）

偏移地址：0xE0

复位值：0x014D 0100

访问：无等待周期，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															

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

位 31:12	Res：保留 必须保持复位值。
位 11:0	HSEWT[11:0]：设置 HSE 稳定等待时间（HSE stabilize wait time setting） HSI 稳定等待时间为 HSEWTx8 个 HSE 时钟周期。 默认等待 256x8=2048 个 HSE 时钟周期。

5.3.18 RCC PLL 时钟控制寄存器（RCC_PLLCTL）

偏移地址：0xE4

复位值：0x0021 0C21

访问：无等待周期，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															

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

位 31:7	Res：保留 必须保持复位值。
位 6:5	DIV_POST[1:0]：除法器除数（The divisor of a divider） 输出除法器的除数。
位 4:0	DIV_PRE[4:0]：预分频系数（Pre-divider of the reference clock） 若 RCC_CFGR.PLLSRC 为 0 时（HSI/2 作为 PLL 输入时钟），此寄存器值固定为 5’b00010；否则，值为 RCC_CFGR2.PLL_PREDIV 的值加 1。

5.3.19 时钟配置寄存器 4（RCC_CFGR4）

偏移地址：0xE8

复位值：0x0000 0438

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											ADCCLKSW[1:0]		PPSS	I2C1CLKSW[1:0]	
											rw		rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTCLK_SEL[1:0]		FLITFCLK_PRE[3:0]				FLITFCLK_SEL[1:0]		ESSS	Res	ESWS[2:0]			ESW[2:0]		
rw		rw				rw		rw		r			rw		

位 31:21	Res：保留 必须保持复位值。
位 20:19	ADCCLKSW[1:0]：ADC 时钟源选择（ADC clock source selection） ● 00：HSI12 选择为 ADC 时钟源，选择这一项，需要使能 HSI12MHz 时钟。 ● 01：PCLK/2 选择为 ADC 时钟源 ● 10：PCLK/4 选择为 ADC 时钟源 ● 11：PCLK 选择为 ADC 时钟源
位 18	PPSS：PLL 前置分频输入时钟源选择（PLL pre-scale clock source selection） ● 0：HSE 时钟选择为 PLL 前置分频时钟源 ● 1：HSI 时钟选择为 PLL 前置分频时钟源
位 17:16	I2C1CLKSW[1:0]：I2C1 时钟源选择（I2C1 clock source selection） ● 00：HSI 选择为 I2C1 时钟源 ● 01：SYSCLK 选择为 I2C1 时钟源 ● 10：PCLK1 选择为 I2C1 时钟源

	● 11：保留
位 15:14	EXTCLK_SEL[1:0]：外部时钟输入引脚选择（External clock pin selection） ● 00：外部时钟引脚 1（PA4）选择作为输入源 ● 01：外部时钟引脚 2（PA13）选择作为输入源 ● 10：外部时钟引脚 3（PC4）选择作为输入源 ● 11：外部时钟引脚 4（PA0）选择作为输入源
位 13:10	FLITFCLK_PRE[3:0]：Flash 擦写操作时钟分频因子（Flash memory programming interface clock prescaler factor） FLITFCLK_PRE 配合 FLITFCLK_SEL 设置 Flash 擦写操作时钟，当 Flash 擦写时必须保证其时钟频率为 8 MHz。 ● 0000：分频系数为 1 ● 0001：分频系数为 2 ● 0010：分频系数为 3 ● 0011：分频系数为 4 ● 0100：分频系数为 5 ● 0101：分频系数为 6 ● 0110：分频系数为 7 ● 0111：分频系数为 8 ● 1000：分频系数为 9 ● 1001：分频系数为 10 ● 1010：分频系数为 11 ● 1011：分频系数为 12 ● 其他：保留 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 9:8	FLITFCLK_SEL[1:0]：Flash 擦写操作时钟源选择（Flash memory programming interface clock source selection） ● 00：HSI 时钟选择为 Flash 擦写操作时钟源 ● 01：SYSCLK 时钟选择为 Flash 擦写操作时钟源 ● 10：外部时钟源选择为 Flash 擦写操作时钟源 ● 11：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 7	ESSS：选择由 RCC_CFGR.SW 还是由 RCC_CFGR4.ESW 配置 SYSCLK（SYSCLK setting bit selection） ● 0：选择 RCC_CFGR.SW 设置 SYSCLK ● 1：选择 RCC_CFGR4.ESW 设置 SYSCLK
位 6	Res：保留 必须保持复位值。
位 5:3	ESWS[2:0]：SYSCLK 时钟源状态（SYSCLK clock source status） 当 ESSS 为 1 时，指示 SYSCLK 时钟状态。 ● 000：LSE 作为 SYSCLK ● 001：LSI 作为 SYSCLK ● 010：内部 48MHz RC 振荡器输出作为 SYSCLK ● 011：HSI12 作为 SYSCLK ● 100：外部时钟引脚输入作为 SYSCLK

	● 其它：未定义
位 2:0	ESW[2:0]：SYSCLK 时钟源选择（SYSCLK clock source selection） 当 ESSS 为 1 时，选择不同时钟源作为 SYSCLK。 ● 000：选择 LSE 作为 SYSCLK ● 001：选择 LSI 作为 SYSCLK ● 010：选择内部 48MHz RC 振荡器输出作为 SYSCLK ● 011：选择 HSI12 作为 SYSCLK ● 100：选择外部时钟引脚输入作为 SYSCLK ● 其它：未定义

5.3.20 RTC 控制寄存器（RCC_BDCR）

偏移地址：0xEC

复位值：0x0000 0100

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															BDRST
															rw

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

位 31:17	Res：保留 必须保持复位值。
位 16	BDRST：RTC 控制寄存器软件复位（RTC control register software reset） 通过软件置位或复位。 ● 0：无效果 ● 1：复位整个 RTC 控制寄存器
位 15	RTCEN：RTC 时钟使能（RTC clock enable） 通过软件置位或复位。 ● 0：禁止 RTC 时钟 ● 1：使能 RTC 时钟
位 14:10	Res：保留 必须保持复位值。
位 9:8	RTCSEL[1:0]：RTC 时钟源选择（RTC clock source selection） 通过软件置位或复位。 ● 00：不选择时钟源 ● 01：LSE 作为时钟源 ● 10：LSI 作为时钟源 ● 11：HSE/32 作为时钟源
位 7:0	Res：保留 必须保持复位值。

6 通用 I/O（GPIO）

本章介绍该系列芯片的 GPIO 功能和寄存器描述。

6.1 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
- 复用功能选择寄存器
- 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。

6.2 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

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期间发生中断请求也不会有问题。

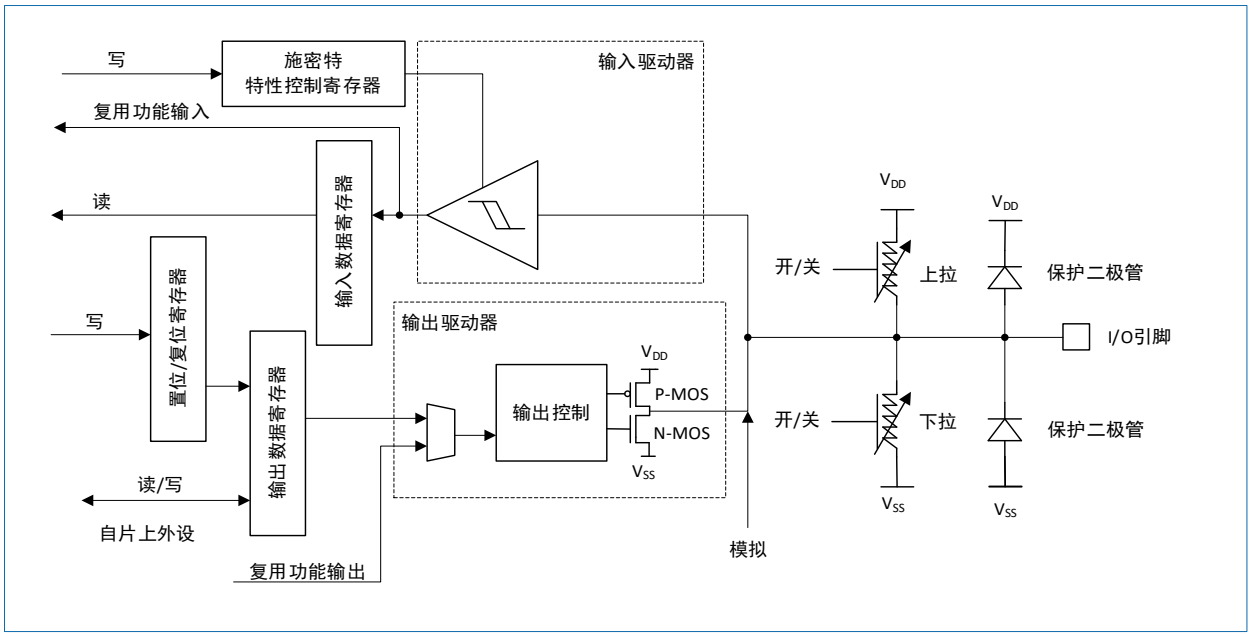

图 6-1 I/0 端口位的基本结构

表 6-1 端口位配置表

MODER（i）[1:0]	OTYPER（i）	OSPEEDR（i）[1:0]		PUPDR（i）[1:0]		I/O 配置	
01	0	OSPEEDR[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	OSPEEDR[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	X	0	0	输入	浮空
	X	X	X	0	1	输入	上拉

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

（1）“X”表示无影响，不起作用。

6.2.1 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 端口都默认为模拟模式，但调试引脚例外：

- PC7：NRST 处于上拉输入状态
- PC10：处于下拉输入状态

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。

6.2.2 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_AFRL（针对引脚 0 至 7）和 GPIOx_AFRH（针对引脚 8 至 15）寄存器对这些输入进行配置。

复位后，复用器选择为复用功能 0（AF0）。在复用模式下通过 GPIOx_MODER 寄存器配置 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，用户必须按照以下进行操作：

- 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 的模式：在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入、复用功能或模拟。
  - 外设复用功能：

    A. 在 GPIOx_AFRL 或 GPIOx_AFRH 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，分别选择输出类型、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。
  - 其它功能：

对于 ADC、OPAMP 和 COMP，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟或复用模式，并在 ADC、

OPAMP 和 COMP 寄存器中配置所需功能。

6.2.3 I/O 端口控制寄存器

每个通用 I/O 端口包括：

- 5 个 32 位的控制寄存器，用于配置 GPIO 口的特性：
  - 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
- 2 个 32 位复用功能选择寄存器 GPIOx_AFRH 和 GPIOx_AFRL
- 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，可以对其进行读/写访问。

6.2.4 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于 GPIOx_BSRR 中的两个控制位：BSy 和 BRy（y= 15..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。

6.2.5 GPIO 锁定机制

通过将特定的序列写到 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_AFRL、GPIOx_AFRH 和 GPIOx_IOSR。

对 GPIOx_LCKR 寄存器执行写操作必须写入特定的写/读序列。当正确的 LOCK 序列写到此寄存器的第 16 位后，会使用 LCKR[15:0]的值来锁定对应 I/O 的配置（在写序列期间，LCKR[15:0]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_AFRL、GPIOx_AFRH 和 GPIOx_IOSR）中的对应位。

6.2.6 I/O 复用功能输入输出

每个通用 I/O 端口包括 2 个 32 位复用功能选择寄存器 GPIOx_AFRH 和 GPIOx_AFRL。这两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 选择信号由复用功能输入和复用功能输出共用，所以只需要为指定的 I/O 的复用功能输入/输出选择一个通道即可。

6.2.7 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。

6.2.8 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。

6.2.9 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 1 个 AHB 时钟周期对 I/O 引脚上的数据采样一次。
- 对输入数据寄存器的读访问可获取 I/O 的状态。
- 对输出数据寄存器的读访问可获取最后的写入值。

6.2.10 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（复用功能输出）。
- 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 1 个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样。
- 对输入数据寄存器的读访问可获取 I/O 状态。

6.2.11 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’。

6.2.12 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 寄存器中的对应位决定。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抗干扰能力。

6.2.13 HSE 或 LSE 引脚用作 GPIO

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

当 HSE 或 LSE 打开的时候，振荡器会控制与其相关的引脚，此时这些引脚的 GPIO 配置不起作用。

将振荡器配置为用户外部输入时钟模式时，仅为时钟输入保留 OSC_IN 或 OSC32_IN 引脚，OSC_OUT 或 OSC32_OUT 引脚仍可以作为 GPIO 使用。

6.3 GPIO 寄存器

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

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

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

偏移地址：0x00

复位值：0xFFCA 3FFF（端口 C）/ 0xFFFF FFFF（其它端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 2y+1:2y （y=15..0）	MODERy[1:0]：端口 x 的 y 引脚工作模式配置位（Port x pin y mode configuration） 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）

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

偏移地址：0x04

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16	Res：保留 必须保持复位值。
位 y （y=15..0）	OTy：端口x 的 y 引脚输出类型配置位（Port x pin y output type configuration） 该位域由软件配置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

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

偏移地址：0x08

复位值：0x000C 0000（端口 C）/ 0x0000 0000（其它端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]		OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 2y+1:2y（y=15..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） 该位域由软件配置I/O 口的速度。 ● x0：低速 ● 01：中速 ● 11：高速

6.3.4 GPIO 口上拉/下拉寄存器（GPIOx_PUPDR）（x= A..C）

偏移地址：0x0C

复位值：0x0006 4000（端口 C）/ 0x0000 0000（其它端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]		PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
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 2y+1:2y（y=15..0）	PUPDRy[1:0]：端口x 的 y 引脚配置为上拉或下拉（Port x pin y pull -up/-down configuration） 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

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

偏移地址：0x10

复位值：0x0000 0280（端口 C）/ 0x0000 0000（其它端口）

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
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16	Res：保留

	必须保持复位值。
位 y （y=15..0）	IDRy：端口x 的 y 引脚的输入数据（Port x pin y input data） 该位包含相应 I/O 口的输入值，只能读。

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

偏移地址：0x14

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	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 y （y=15..0）	ODRy：端口 x 的 y 引脚输出值（Port x pin y output data） 该位用于配置端口的输出状态。 ● 0：端口 x 的第 y 个引脚输出低电平 ● 1：端口 x 的第 y 个引脚输出高电平

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

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 16+y （y=15..0）	BRy：端口x 的 y 引脚复位控制位（Port x pin y reset control） 该位只能写。 读该位时返回值为0。若 BSy 和 BRy 同时设置，BSy 有优先权。 ● 0：对端口 x 的 ODRy 位无影响 ● 1：复位端口 x 的 ODRy 位
位 y （y=15..0）	BSy：端口x 的 y 引脚的设置控制位（Port x pin y set bit） 该位只能写。 读该位时返回值为0。 ● 0：对相应的ODRy 位无影响 ● 1：置位端口 x 的ODRy 位

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

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
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

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 y （y=15..0）	LCKy：端口x 的 y 引脚的锁定位（Port x pin y lock bit） 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

6.3.9 GPIO 复用功能低位寄存器（GPIOx_AFRL）（x= A..C）

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFRL7[3:0]				AFRL6[3:0]				AFRL5[3:0]				AFRL4[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFRL3[3:0]				AFRL2[3:0]				AFRL1[3:0]				AFRL0[3:0]			
rw				rw				rw				rw			

位 4y+3:4y （y = 7..0）	AFRLy[3:0]：端口 x 引脚y 的复用功能选择（Alternate function selection low bits of port x pin y） 可由软件来配置 I/O 口的复用功能。 AFRLy 的选择： ● 0000：AF0 ● 0001：AF1 ● 0010：AF2 ● 0011：AF3 ● 0100：AF4

	● 0101：AF5 ● 0110：AF6 ● 0111：AF7 ● 其他值：保留

6.3.10 GPIO 复用功能高位寄存器（GPIOx_AFRH）（x= A..C）

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
AFRH15[3:0]				AFRH14[3:0]				AFRH13[3:0]				AFRH12[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFRH11[3:0]				AFRH10[3:0]				AFRH9[3:0]				AFRH8[3:0]			
rw				rw				rw				rw			

位 4y-29:4y-32 (y=15..8)	AFRHy[3:0]：端口x 引脚y 的复用功能选择（Alternate function selection high bits of port x pin y） 该位域可由软件配置复用功能I/O 口。 AFRHy 的选择： ● 0000：AF0 ● 0001：AF1 ● 0010：AF2 ● 0011：AF3 ● 0100：AF4 ● 0101：AF5 ● 0110：AF6 ● 0111：AF7 ● 其他值：保留

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

偏移地址：0x28

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:16	Res：保留 必须保持复位值。
位 y （y=15..0）	BRy：端口x 的 y 引脚的复位控制位（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 的ODy 位无影响 ● 1：复位端口 x 的ODy 位

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

偏移地址：0x30

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOSEN15	IOSEN14	IOSEN13	IOSEN12	IOSEN11	IOSEN10	IOSEN9	IOSEN8	IOSEN7	IOSEN6	IOSEN5	IOSEN4	IOSEN3	IOSEN2	IOSEN1	IOSEN0
rw	rw	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 y （y=15..0）	IOSENy：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

7 系统配置控制器（SYSCFG）

该系列芯片有一组系统配置寄存器。系统配置控制器的主要功能如下：

- 在部分 IO 口上启用或禁用 I2C 超快速模式（Fast Mode Plus）。
- 重映射存储器到代码起始区域。
- 管理连接到 GPIO 口的外部中断。
- 管理 TIM3_CH4 输入重映射到 LSI、HSI48/16384、LSE 等信号。
- 管理内部一些模拟信号输出到 IO 的开关。
- 配置内部分压网络（8 位 DAC）。

7.1 SYSCFG 配置寄存器 1 的功能说明

7.1.1 I2C 超快速模式驱动能力

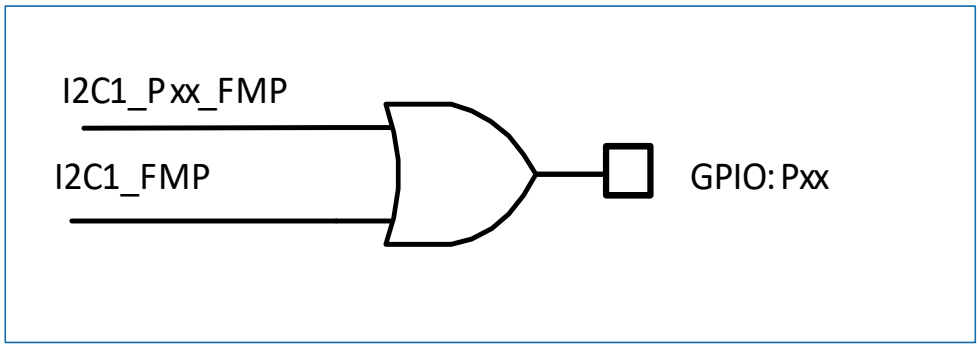

图 7-1 I2C 超快速模式驱动能力控制位

I2C 超快速模式若需要更强的驱动能力，可通过配置 SYSCFG_CFGR1 寄存器的超快速模式驱动能力位实现。如上图所示，超快速模式驱动能力由 I2C1_Pxx_FMP 和 I2C1_FMP 或运算控制。I2C1_Pxx_FMP 仅控制一个 GPIO（Pxx），I2C1_FMP 为 1 时则所有 I2C1_Pxx_FMP 引脚均启用超快速模式（FM+）。

7.1.2 重映射存储器到代码起始区域

通过配置 MEM_MODE 位，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：系统 Flash 映射到 0x0000 0000
- 11：嵌入式 SRAM 映射到 0x0000 0000

7.2 SYSCFG 配置寄存器 2 的功能说明

7.2.1 TIM3_CH4 输入功能重映射

通过配置 TIM3_CH4_REMAP 位，可选择 TIM3_CH4 的输入信号：

- 000：GPIO 输入（PC7）
- 001：LSI 时钟
- 010：HSI48 时钟/16384
- 011：LSE 时钟
- 其他值：（PC7）

7.2.2 Cortex-M0 LOCKUP 位使能

通过配置 LOCKUP_LOCK，使能并锁定连接 Cortex-M0 LOCKUP（硬件故障）输出到 TIM1 的刹车（Break）输入。

7.3 SYSCFG 配置寄存器 3 的功能说明

7.3.1 控制内部信号输出到 IO

- EN_SW_VREF12_PC4：1.2V VREFINT 参考电压输出到 PC4 的开关。
- EN_SW_VOLTAGE_DIV_PA5：VOLTAGE_DIV 输出到 PA5 的开关。
- EN_SW_VTEST_PC4：VTEST 输出到 PC4 的开关。
- EN_SW_VIN_LVD：VIN_LVD 连接到 PA8 的使能。

7.3.2 VOLTAGE_DIV（DAC_8 位）控制

DAC_CLK_DIVSEL：DAC8 输入时钟选择。

VOLTAGE_DIV_EN：VOLTAGE_DIV（DAC_8 位）使能。

VOLTAGE_DIV_DATA_IN[7:0]：VOLTAGE_DIV（DAC_8 位）数据输入配置位。

分压器（8 位 DAC）输出电压大小为：VOLTAGE_DIV_DATA_IN[7:0]/256*VDDA。内部分压器输出电压作为比较器的一路参考电压。

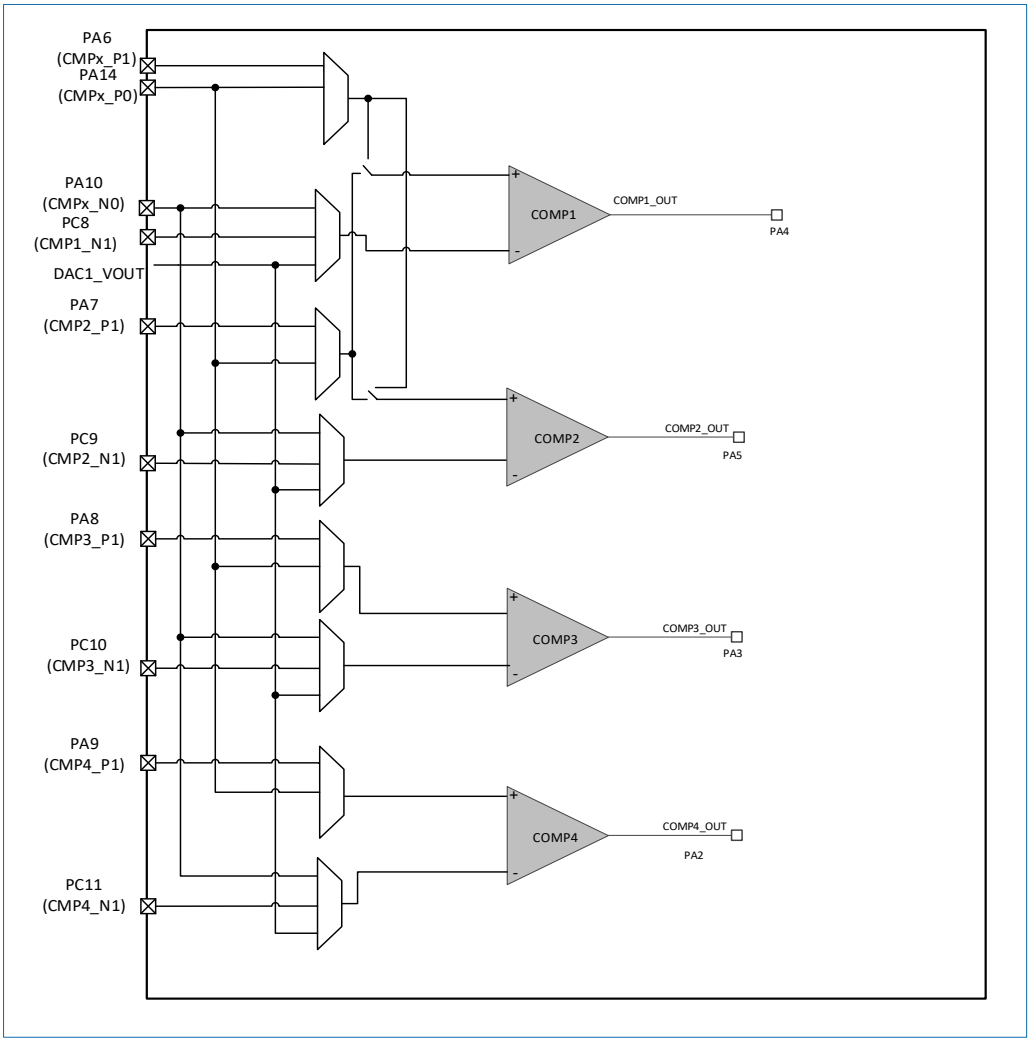

图 7-2 内部分压器（8 位 DAC）输出电压作为比较器的参考电压

7.4 SYSCFG 寄存器

基地址：0x4001 0000

空间大小：0x400

7.4.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）

偏移地址：0x00

复位值：0x0000 0000

该寄存器用于存储器和 DMA 请求重映射的特定配置，并控制特殊的 I/O 功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res											I2C1_FMP	Res			
											rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		I2C1_PC6_FMP	I2C1_PC5_FMP	I2C1_PB15_FMP	I2C1_PB12_FMP	Res								MEM_MODE[1:0]	
		rw	rw	rw	rw									rw	

位 31:21	Res：保留 必须保持复位值。
位 20	I2C1_FMP：I2C1 的超快速模式（FM+）驱动能力激活位（FM+ driving capability activation for I2C1） 该位与 I2C1_Pxx_FMP 位进行或运算，由软件设置和清零。 ● 0：超快速模式（FM+）仅由 I2C1_Pxx_FMP 位控制。 ● 1：PC6、PC5、PB15 和 PB12 引脚均启用超快速模式（FM+）。
位 19:14	Res：保留 必须保持复位值。
位 13	I2C1_PC6_FMP：PC6 超快速模式（FM+）驱动能力激活位（I2C1 Fast Mode Plus (FM+) driving capability activation bit for PC6） 该位为 PC6 引脚开启 I2C 超快速模式（FM+），由软件设置和清零。 ● 0：PC6 引脚设置为输出速度在 2MHz/10MHz/50MHz 下的标准模式。 ● 1：PC6 引脚配置为 I2C 超快速模式（FM+），且 I2C 速度控制被旁路。
位 12	I2C1_PC5_FMP：PC5 超快速模式（FM+）驱动能力激活位（I2C1 Fast Mode Plus (FM+) driving capability activation bit for PC5） 该位为 PC5 引脚开启 I2C 超快速模式（FM+），由软件设置和清零。 ● 0：PC5 引脚设置为输出速度在 2MHz/10MHz/50MHz 下的标准模式。 ● 1：PC5 引脚配置为 I2C 超快速模式（FM+），且 I2C 速度控制被旁路。
位 11	I2C1_PB15_FMP：PB15 超快速模式（FM+）驱动能力激活位（I2C1 Fast Mode Plus (FM+) driving capability activation bit for PB15） 该位为 PB15 引脚开启 I2C 超快速模式（FM+），由软件设置和清零。 ● 0：PB15 引脚设置为输出速度在 2MHz/10MHz/50MHz 下的标准模式。 ● 1：PB15 引脚配置为 I2C 超快速模式（FM+），且 I2C 速度控制被旁路。
位 10	I2C1_PB12_FMP：PB12 超快速模式（FM+）驱动能力激活位（I2C1 Fast Mode Plus (FM+) driving capability activation bit for PB12） 该位为 PB12 引脚开启 I2C 超快速模式（FM+），由软件设置和清零。 ● 0：PB12 引脚设置为输出速度在 2MHz/10MHz/50MHz 下的标准模式。

	● 1：PB12 引脚配置为 I2C 超快速模式（FM+），且 I2C 速度控制被旁路。
位 9:2	Res：保留 必须保持复位值。
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。当复位后，这些位值由实际引导模式配置选择的值决定。 ● x0：主 Flash 存储器映射到 0x0000 0000 ● 01：系统 Flash 映射到 0x0000 0000 ● 11：嵌入式 SRAM 映射到 0x0000 0000

7.4.2 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（EXTIy external interrupt source input selection） 该位域由软件设置。 ● x000：PA[y]引脚 ● x001：PB[y]引脚 ● x010：PC[y]引脚 ● x011：保留 ● x100：保留 ● x101：PF[y]引脚 ● 其它配置：保留

7.4.3 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)	EXTIy[3:0]：选择 EXTIy 的外部中断源（EXTIy external interrupt source input selection）

(y =7..4)	该位域由软件设置。 ● x000：PA[y]引脚 ● x001：PB[y]引脚 ● x010：PC[y]引脚 ● x011：保留 ● x100：保留 ● x101：PF[y]引脚 ● 其它配置：保留

7.4.4 SYSCFG 外部中断配置寄存器 3（SYSCFG_EXTICR3）

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
EXTI11[3:0]				EXTI10[3:0]				EXTI9[3:0]				EXTI8[3:0]			
rw				rw				rw				rw			

位 31:16	Res：保留 必须保持复位值。
位 4(y-8)+3:4(y-8) (y = 11..8)	EXTIy[3:0]：选择 EXTIy 的外部中断源（EXTIy external interrupt source input selection） 该位域由软件设置。 ● x000：PA[y]引脚 ● x001：PB[y]引脚 ● x010：PC[y]引脚 ● x011：保留 ● x100：保留 ● x101：PF[y]引脚 ● 其它配置：保留

7.4.5 SYSCFG 外部中断配置寄存器 4（SYSCFG_EXTICR4）

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
EXTI15[3:0]				EXTI14[3:0]				EXTI13[3:0]				EXTI12[3:0]			
rw				rw				rw				rw			

位 31:16	Res：保留 必须保持复位值。
位 4(y-12)+3:4(y-12) (y = 15..12)	EXTIy[3:0]：选择 EXTIy 的外部中断源（EXTIy external interrupt source input selection） 该位域由软件设置。 ● x000：PA[y]引脚

	• x001：PB[y]引脚 • x010：PC[y]引脚 • x011：保留 • x100：保留 • x101：PF[y]引脚 • 其它配置：保留

7.4.6 SYSCFG 配置寄存器 2（SYSCFG_CFGR2）

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													TIM3_CH4_REMAP[2:0]		
													rw		

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

位 31:19	Res：保留 必须保持复位值。
位 18:16	TIM3_CH4_REMAP[2:0]：TIM3 CH4 输入信号重映射控制位（TIM3 channel4 remapping） 这些位由软件设置或清零。 • 000：GPIO 输入（PC7） • 001：LSI 时钟 • 010：HSI48 时钟/16384 • 011：LSE 时钟 • 其他值：PC7
位 15:1	Res：保留 必须保持复位值。
位 0	LOCKUP_LOCK：Cortex-M0 LOCKUP 位使能位（Cortex-M0 LOCKUP bit enable bit） 该位由软件设置，由系统复位清零。它可用于使能并锁定连接 Cortex-M0 LOCKUP（硬件故障）输出到 TIM1 的刹车（Break）输入。 • 0：Cortex-M0 LOCKUP 输出断开与 TIM1 刹车（Break）输入的连接。 • 1：Cortex-M0 LOCKUP 输出连接到 TIM1 刹车（Break）输入。

7.4.7 SYSCFG 配置寄存器 3（SYSCFG_CFGR3）

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
REF_LOCK	Res													DAC_CLK_DIVSEL[1:0]	
rs														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			EN_SW_VREF12_PC4	EN_SW_VOLTAGE_DIV_PA5	EN_SW_VTEST_PC4	EN_SW_VIN_LVD	VOLTAGE_DIV_EN	VOLTAGE_DIV_DATA_IN[7:0]							
			rw	rw	rw	rw	rw	rw							

位 31	REF_LOCK：SYSCFG_CFGR3 锁定位（SYSCFG_CFGR3 Register lock ） 该位由软件设置，由硬件复位清除。它锁定了参考控制/状态寄存器 SYSCFG_CFGR3 的全部内容。 ● 0：SYSCFG_CFGR3[31:0]位是可读/写的 ● 1：SYSCFG_CFGR3[31:0]位是只读的
位 30:18	Res：保留 必须保持复位值。
位 17:16	DAC_CLK_DIVSEL[1:0]：DAC 时钟输入选择（DAC clock select） ● 00：PCLK 2 分频 ● 01：PCLK 4 分频 ● 10：PCLK 8 分频 ● 11：PCLK16 分频
位 15:13	Res：保留 必须保持复位值。
位 12	EN_SW_VREF12_PC4：1.2V VREFINT 参考电压输出到 PC4 开关（Enable switch of 1.2V VREFINT reference Output to PC4） 此位由软件设置和清除。（仅当未设置 REF_LOCK 时） ● 0：断开 1.2V VREFINT 参考电压输出到 PC4 开关 ● 1：闭合 1.2V VREFINT 参考电压输出到 PC4 开关
位 11	EN_SW_VOLTAGE_DIV_PA5：VOLTAGE_DIV 输出到 PA5 开关（Enable switch of VOLTAGE_DIV out to PA5） 此位由软件设置和清除。（仅当未设置 REF_LOCK 时） ● 0：断开 VOLTAGE_DIV 输出到 PA5 开关 ● 1：闭合 VOLTAGE_DIV 输出到 PA5 开关
位 10	EN_SW_VTEST_PC4：VTEST 输出到 PC4 开关（Enable switch of VTEST to PC4） 此位由软件设置和清除。（仅当未设置 REF_LOCK 时） ● 0：断开 VTEST 输出到 PC4 开关 ● 1：闭合 VTEST 输出到 PC4 开关
位 9	EN_SW_VIN_LVD：VIN_LVD 连接到 PA8 开关（Enable switch of VIN_LVD to PA8） 此位由软件设置和清除。（仅当未设置 REF_LOCK 时） ● 0：断开 VIN_LVD 连接到 PA8 开关 ● 1：闭合 VIN_LVD 连接到 PA8 开关
位 8	VOLTAGE_DIV_EN：VOLTAGE_DIV（DAC_8 位）使能（Enable the VOLTAGE_DIV） 此位由软件设置和清除。（仅当未设置 REF_LOCK 时） ● 0：禁用 VOLTAGE_DIV（DAC_8 位） ● 1：使能 VOLTAGE_DIV（DAC_8 位）
位 7:0	VOLTAGE_DIV_DATA_IN[7:0]：VOLTAGE_DIV（DAC_8 位）数据输入配置位（VOLTAGE_DIV DATA_IN bit） 这些位由软件设置和清除（仅当未设置 REF_LOCK 时）。VOLTAGE_DIV_DATA_IN[7:0]数据输入位用于配置 8 位 DAC 输出电压大小。 ● 00000000 8 位 DAC 输出电压大小为 0/256*VDDA ● 00000001 8 位 DAC 输出电压大小为 1/256*VDDA

	● … ● 11111111 8 位 DAC 输出电压大小为 256/256*VDDA

8 直接存储器访问控制器（DMA）

直接存储读取器（DMA）用于在外设与存储器之间或是存储器到存储器之间的高速数据传输。数据可以在不占用任何 CPU 资源的情况下快速地从指定的源地址搬移到目标地址，以便让 CPU 有更多的资源去处理其它应用。

芯片集成了 1 个 DMA 控制器，DMA 控制器拥有 5 个通道，DMA 控制器管理来自于一个或多个外设的访问请求。DMA 拥有一个仲裁器去处理不同的 DMA 请求的优先级。

8.1 DMA 的主要功能

- DMA 拥有 5 个通道请求。
- 每个通道能够通过选择矩阵开关与硬件 DMA 请求相连，同时每个通道都支持软件触发，这些配置都需要由软件完成。
- 不同通道的 DMA 请求优先级可以由软件进行配置（共 4 个可配的优先等级：最高，高，中，低）。
- 独立可配的源以及目标传输位宽（字节，半字，字），模拟封包和拆包。源/目标地址必须以传输数据的位宽作为最小单位来对齐。
- 支持循环缓冲管理。
- 每个通道支持 3 个事件标志（DMA 完成一半传输标志，DMA 传输完成标志，DMA 传输错误标志），以逻辑或的关系为每个通道请求对应的中断。
- 支持内存（内部存储器）到内存传输模式。
- 支持外设到内存，内存到外设，外设到外设传输模式。
- 支持将 Flash、SRAM、APB 或 AHB 总线上的外设配置为源或目标。
- 传输数据的个数（每个通道上单次 DMA 传输的次数）可配置：最多可配置到 65535。
- DMA 拥有 32 个请求源，每一个通道可以灵活配置这些请求源。

8.1.1 功能说明

DMA 的结构框图如下：

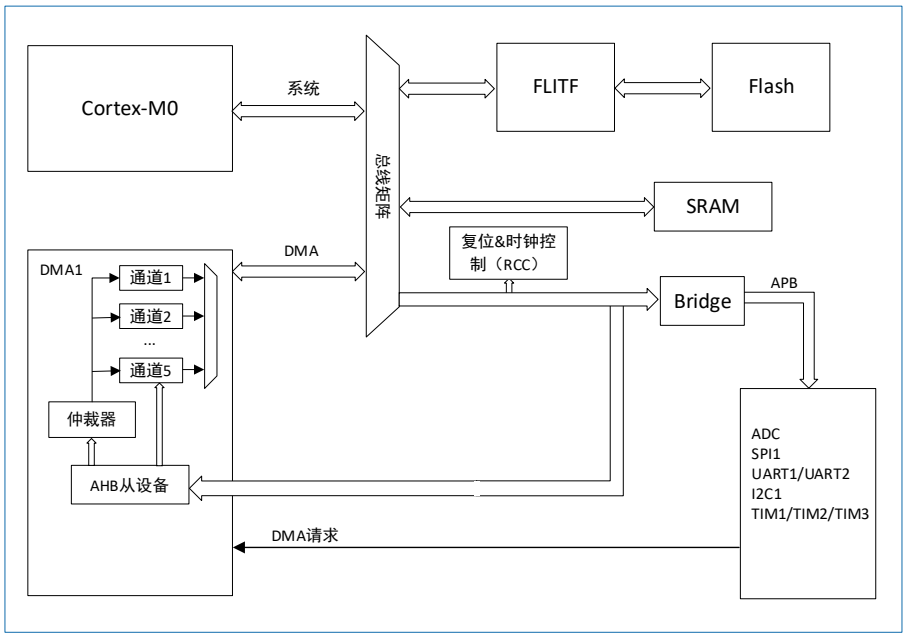

图 8-1 DMA 结构框图

DMA 控制器通过与 Cortex-M0 内核共享系统总线完成直接存储传输功能，当 CPU 和 DMA 访问同一个目标地址的时候（内存或外设），DMA 请求可能会让 CPU 访问系统总线的某些循环暂停。总线矩阵实现了循环机制，以保证至少有一半的系统总线带宽（对于内存和外设都有效）供给 CPU。

8.1.2 DMA 传输

发生一次事件之后，外设会发送一个请求信号给 DMA 控制器。DMA 控制器根据对应通道的配置来处理收到的这个请求。在 DMA 控制器访问外设的同时，DMA 控制器也会发送一个应答给外设。外设一旦收到 DMA 控制器的应答信号就会释放当前的 DMA 请求。一旦外设的 DMA 请求失效，DMA 控制器也会释放掉应答信号。如果外设仍有其他的请求，外设可以发起下一次传输。

归纳一下就是每个 DMA 传输由三个操作组成：

- 根据当前外设/内存地址寄存器中的值，从外设的数据寄存器或者指定地址的内存加载数据。用于第一次传输的起始地址通过对 DMA_CPAR 或 DMA_CMAR 寄存器编程确定。
- 根据当前外设/内存地址寄存器中的值，将上一步中加载得到的数据传输到指定的外设寄存器或者指定的地址中。用于第一次传输的起始地址是通过对 DMA_CPAR 或 DMA_CMAR 编程来确定的。
- 用于存剩余传输次数的 DMA_CNDTR 寄存器。

8.1.3 仲裁器

仲裁器根据通道优先级管理 DMA 的通道请求，并执行外设/内存访问序列。

DMA 通道间的优先级规则有两种：

软件可配的优先级：DMA 的每个通道可以通过对 DMA_CCR 寄存器编程配置成以下四种优先级之一：

- 最高优先级
- 高优先级
- 中优先级
- 低优先级

硬件默认优先级：如果两个通道配成了相同的软件优先级，则序号较小的通道具有更高的优先级，例如通道 2 的优先级高于通道 4 的优先级。

8.1.4 DMA 通道

每个 DMA 通道都能完成在固定的外设地址与内存地址之间的 DMA 传输。每次 DMA 传输数据的总量（最大 65535）可以编程控制。每完成一次传输，存储剩余传输次数的 DMA_CNDTR 寄存器就会进行一次减计数。

可编程的数据传输位宽

外设和内存传输的数据宽度可以通过对 DMA_CCR 寄存器中的 PSIZE 以及 MSIZE 进行编程确定。

指针自加

通过对 DMA_CCR 寄存器中的 PINC 和 MINC 进行编程，可以将外设和内存指针设置为每次传输后自动增加（如果使能了自增模式，每次增加的值与位宽有关，如果位宽设置为字节，则加 1；如果位宽设置为半字则加 2，字则加 4）。第一次传输所使用到的外设或内存地址都由软件对 DMA_CPAR 和 DMA_CMAR 寄存器进行编程确定。DMA 传输过程中，DMA_CPAR 和 DMA_CMAR 寄存器值保持为初次配置的值，因此通道的外设或内存地址自增功能使能时，软件是无法访问到当前传输对应的外设或者内存地址的。

如果通道被配置为非循环模式，那么在完成了最后一次传输之后（存储剩余传输次数的寄存器 DMA_CNDTR 值变为 0），就不会再处理该通道上的 DMA 请求。如果要加载新的值到 DMA_CNDTR 寄存器

中，对应的 DMA 通道必须先关闭。

如果一个 DMA 通道被关闭了，但是该 DMA 通道的寄存器不会被复位。DMA_CCR，DMA_CPAR 和 DMA_CMAR 保持之前配置的值。

如果通道被配置为循环工作模式，在最后一次传输完成后，DMA_CNDTR 寄存器会自动加载初始配置的值。当前内部地址寄存器也会重新加载 DMA_CPAR 和 DMA_CMAR 寄存器中的值。

DMA 通道配置步骤

请遵循以下顺序去配置一个 DMA 通道 x（x 代表通道编号）。

1. 配置 DMA_CPAR 寄存器设置首次传输的外设地址。传输时会根据这个寄存器中的值将数据从这个地址对应的外设寄存器中搬出或者是从内存中将数据搬入到这个地址对应的外设寄存器。
2. 配置 DMA_CMAR 寄存器，配置首次传输的内存地址。传输时会根据这个寄存器中的值将数据从内存中这个地址搬出或者从外设（或其他地址的内存）将数据搬入。
3. 配置 DMA_CNDTR 寄存器，配置总共要传输的数据个数，每完成一个数据的传输，DMA_CNDTR 的值会减 1。
4. 通过配置 DMA_CCR 寄存器中的 PL[1:0]配置该通道的优先级。
5. 配置 DMA_CCR 寄存器，配置数据传输方向，循环模式，外设和内存地址自增模式，外设和内存的数据传输位宽，以及是否在传输完成一半或传输完成时产生中断。
6. 通过置位 DMA_CCR 寄存器中的 EN 位激活该通道。

通道一旦被激活，就可以处理来自与该通道相连外设的 DMA 请求了。

当已完成的传输个数达到设定的总数一半时，半传输标志（HTIF）置位，此时如果半传输中断使能位（HTIE）为’1’，则会产生一个中断。当所有传输完成时，传输完成标志（TCIF）置位，此时如果传输完成中断使能位（TCIE）为’1’，则会产生一个传输完成中断。

循环模式

循环模式用于循环地向一个指定大小的数据缓存空间中搬移数据（数据流传输，如在 ADC 扫描模式下可能会用到）。循环模式可以通过配置 DMA_CCR 寄存器中的 CIRC 位为’1’为使能。循环模式激活的情况下，每当所有指定个数的数据传输完成后，DMA_CNDTR 的值会自动地加载为初始配置时的值，对应的通道也会一直处理 DMA 请求。

内存到内存传输模式

DMA 通道也可以在没有被外设请求触发的情况下工作，这种模式叫做内存到内存传输模式。

如果 DMA_CCR 寄存器中 MEM2MEM 位为’1’，则一旦 DMA_CCR 中的 EN 位置位，该通道上的 DMA 传输就会立即启动。当 DMA_CNDTR 寄存器值变为 0 时，该通道的数据传输停止。内存到内存传输模式不能与循环模式一起使用。

8.1.5 可编程的数据位宽，数据对齐及字节顺序

当 PSIZE 和 MSIZE 配置不一致时，DMA 会按下表描述进行数据对齐：可编程数据宽度、字节存储次序（位 PINC = MINC = 1 时）所述方式进行对齐。

表 8-1 可编程的数据宽度、字节存储次序（位 PINC = MINC = 1 时）

源端口宽度	目标端口宽度	待传输数据项数目 NDT	源内容: 地址/数据	传输操作	目标内容: 地址/数据
8	8	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1:读取 B0[7:0]@0x0，然后写入 B0[7:0]@0x0 2:读取 B1[7:0]@0x0，然后写入 B1[7:0]@0x1 3:读取 B2[7:0]@0x0，然后写入 B2[7:0]@0x2 4:读取 B3[7:0]@0x0，然后写入 B3[7:0]@0x3	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3
8	16	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1:读取 B0[7:0]@0x0，然后写入 00B0[15:0]@0x0 2:读取 B1[7:0]@0x1，然后写入 00B1[15:0]@0x2 3:读取 B2[7:0]@0x2，然后写入 00B2[15:0]@0x4 4:读取 B3[7:0]@0x3，然后写入 00B3[15:0]@0x6	@0x0/00B0 @0x2/00B1 @0x4/00B2 @0x6/00B3
8	32	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1:读取 B0[7:0]@0x0，然后写入 000000B0[31:0] @0x0 2:读取 B1[7:0]@0x1，然后写入 000000B1[31:0] @0x4 3:读取 B2[7:0]@0x2，然后写入 000000B2[31:0] @0x8 4:读取 B3[7:0]@0x3，然后写入 000000B3[31:0] @0xC	@0x0/000000B0 @0x4/000000B1 @0x8/000000B2 @0xC/000000B3
16	8	4	@0x0/B1B0 @0x2/B3B2 @0x4/B5B4 @0x6/B7B6	1:读取 B1B0[15:0]@0x0，然后写入 B0[7:0]@0x0 2:读取 B3B2[15:0]@0x2，然后写入 B2[7:0]@0x1 3:读取 B5B4[15:0]@0x4，然后写入 B4[7:0]@0x2 4:读取 B7B6[15:0]@0x6，然后写入 B6[7:0]@0x3	@0x0/B0 @0x1/B2 @0x2/B4 @0x3/B6
16	16	4	@0x0/B1B0 @0x2/B3B2 @0x4/B5B4 @0x6/B7B6	1:读取 B1B0[15:0]@0x0，然后写入 B1B0[15:0]@0x0 2:读取 B3B2[15:0]@0x2，然后写入 B3B2[15:0]@0x2 3:读取 B5B4[15:0]@0x4，然后写入 B5B4[15:0]@0x4 4:读取 B7B6[15:0]@0x6，然后写入 B7B6[15:0]@0x6	@0x0/B1B0 @0x2/B3B2 @0x4/B5B4 @0x6/B7B6
16	32	4	@0x0/B1B0 @0x2/B3B2 @0x4/B5B4 @0x6/B7B6	1:读取 B1B0[15:0]@0x0，然后写入 0000B1B0[31:0]@0x0 2:读取 B3B2[15:0]@0x2，然后写入 0000B3B2[31:0]@0x4 3:读取 B5B4[15:0]@0x4，然后写入 0000B5B4[31:0]@0x8 4: 读取 B7B6[15:0]@0x6 ， 然后写入 0000B7B6[31:0]@0xC	@0x0/0000B1B0 @0x4/0000B3B2 @0x8/0000B5B4 @0xC/0000B7B6
32	8	4	@0x0/B3B2B1B0 @0x4/B7B6B5B4 @0x8/BBBAB9B8 @0xC/BFBEBDBC	1:读取 B3B2B1B0[31:0]@0x0，然后写入 B0[7:0]@0x0 2:读取 B7B6B5B4[31:0]@0x4，然后写入 B4[7:0]@0x1 3:读取 BBBAB9B8[31:0]@0x8，然后写入 B8[7:0]@0x2 4:读取 BFBEBDBC[31:0]@0xC，然后写入 BC[7:0]@0x3	@0x0/B0 @0x1/B4 @0x2/B8 @0x3/BC
32	16	4	@0x0/B3B2B1B0 @0x4/B7B6B5B4 @0x8/BBBAB9B8 @0xC/BFBEBDBC	1: 读取 B3B2B1B0[31:0]@0x0 ， 然后写入 B1B0[15:0]@0x0 2: 读取 B7B6B5B4[31:0]@0x4 ， 然后写入 B5B4[15:0]@0x2 3: 读取 BBBAB9B8[31:0]@0x8 ， 然后写入 B9B8[15:0]@0x4 4: 读取 BFBEBDBC[31:0]@0xC ， 然后写入 BDBC[15:0]@0x6	@0x0/B1B0 @0x2/B5B4 @0x4/B9B8 @0x6/BDBC
32	32	4	@0x0/B3B2B1B0 @0x4/B7B6B5B4	1: 读取 B3B2B1B0[31:0]@0x0 ， 然后写入 B3B2B1B0[31:0]@0x0	@0x0/B3B2B1B0 @0x4/B7B6B5B4

源端口宽度	目标端口宽度	待传输数据项数目 NDT	源内容: 地址/数据	传输操作	目标内容: 地址/数据
			@0x8/BBBAB9B8 @0xC/BFBEBDBC	2: 读取 B7B6B5B4[31:0]@0x4，然后写入 B7B6B5B4[31:0]@0x4 3: 读取 BBBAB9B8[31:0]@0x8，然后写入 BBBAB9B8[31:0]@0x8 4: 读取 BFBEBDBC[31:0]@0xC，然后写入 BFBEBDBC[31:0]@0xC	@0x8/BBBAB9B8 @0xC/BFBEBDBC

解决 AHB 外设无法支持字节或半字写操作的问题

如果 DMA 发起一次 AHB 字节或半字写操作，数据则会被复制到 HWDATA[31:0]总线的未使用通道上。因此，若所用的 AHB 从外设不支持字节或半字写操作（即外设未使用 HSIZE）且不会产生任何错误，则 DMA 会对 HWDATA 的 32 个位执行写操作，如下两个示例所示：

- 要写入半字”0xABCD”，则 DMA 会将 HWDATA 总线设为”0xABCDABCD”，同时将 HSIZE 设为”HalfWord”
- 要写入字节”0xAB”，则 DMA 会将 HWDATA 总线设为”0xABABABAB”，同时将 HSIZE 设为”Byte”

假设 AHB/APB 桥为 AHB 32 位从外设，且该外设未设置数据的 HSIZE，则任何 AHB 字节或半字操作都将转换为 32 位 APB 操作，转换方式如下所示：

- 将 AHB 字节写操作转化为 APB 字写操作，如向 0x0（或 0x1、0x2、0x3）写入数据”0xB0”转化为向 0x0 写入数据”0xB0B0B0B0”
- 将 AHB 半字写操作转化为 APB 字写操作，如向 0x0（或 0x2）写入数据”0xB1B0”转化为向 0x0 写入数据”0xB1B0B1B0”

例如，若用户希望对 APB 备份寄存器（与 32 位地址边界对齐的 16 位寄存器）执行写操作，则必须将存储器源大小（MSIZE）配置为”16 位”，同时将外设目标大小（PSIZE）配置为”32 位”。

8.1.6 错误管理

当对保留的地址空间执行读写操作时，将产生 DMA 传输错误。若在 DMA 读或写访问过程中产生了 DMA 传输错误，则会将相应的通道的配置寄存器（DMA_CCR）中的 EN 位进行硬件清零，从而自动禁止出错的通道；同时，该通道的 DMA_ISR 寄存器中的传输错误中断标志（TEIF）将置 1，如果此前已将 DMA_CCR 寄存器中的传输错误中断使能位（TEIE）置 1，则将产生一个中断。

8.1.7 DMA 中断

对于每个 DMA 通道，在发生”半传输”、”传输完成”或”传输错误”时都可以产生中断。可以使用单独的中断使能位以提高灵活性。

表 8-2 DMA 中断请求

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

8.1.8 DMA 请求映射

来自外设的各硬件请求可通过 DMA 通道选择寄存器映射到 DMA 通道（1 到 5）。在一个通道上，一次只能响应一个请求（参见图 8-2）。外设 DMA 请求可对相应外设的寄存器的 DMA 控制位进行编程，从而单独进行激活或取消激活。

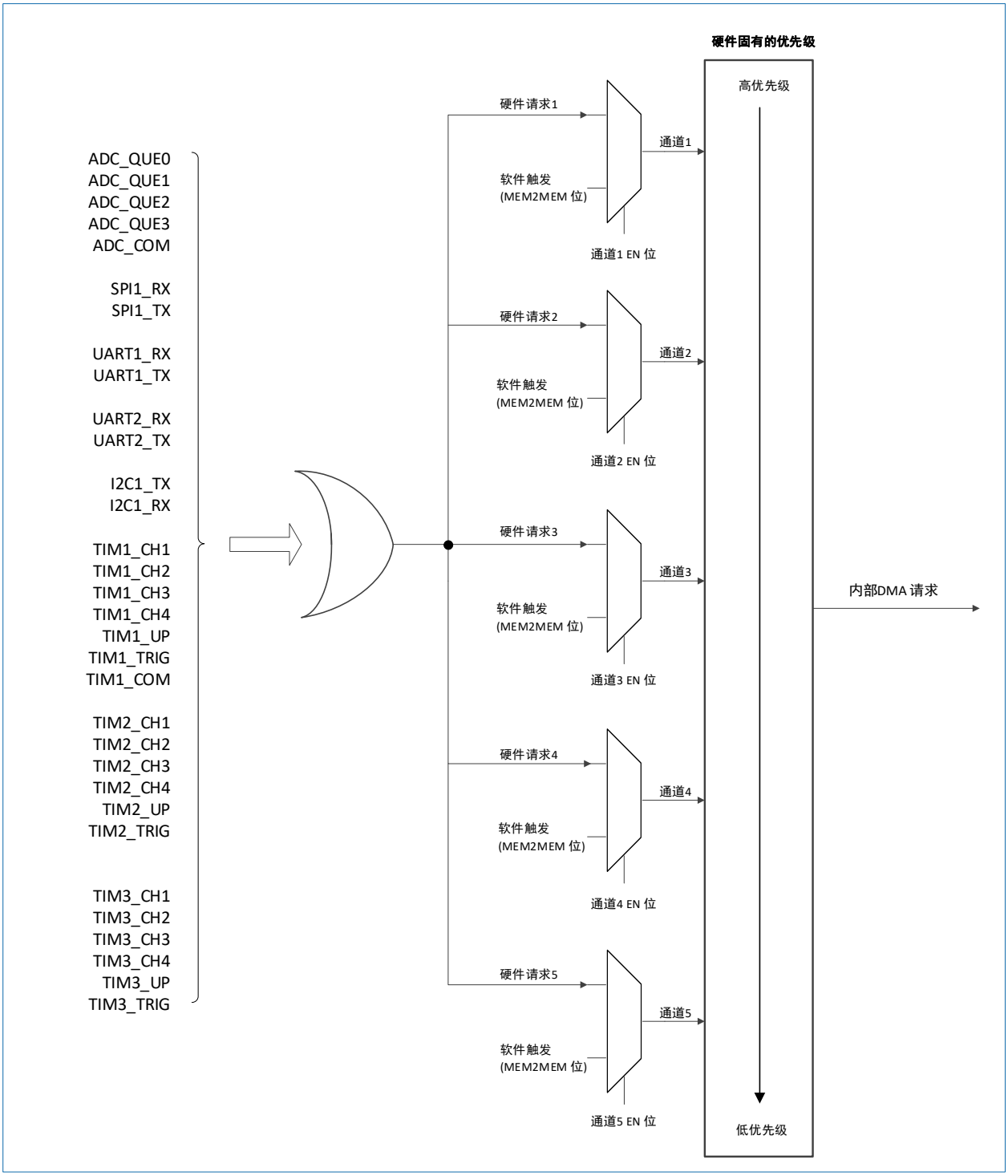

图 8-2 DMA 请求映射

DMA 的各个通道都支持以下 DMA 请求，见下表：

表 8-3 DMA 通道请求

DMA_CSELR.CS[4:0]	请求源
0	ADC_QUE0
1	ADC_QUE1
2	ADC_QUE2
3	ADC_QUE3
4	ADC_COM
5	SPI1_RX
6	SPI1_TX
7	UART1_RX
8	UART1_TX
9	UART2_RX
10	UART2_TX
11	I2C1_TX
12	I2C1_RX
13	TIM1_CH1
14	TIM1_CH2
15	TIM1_CH3
16	TIM1_CH4
17	TIM1_UP
18	TIM1_TRIG
19	TIM1_COM
20	TIM2_CH1
21	TIM2_CH2
22	TIM2_CH3
23	TIM2_CH4
24	TIM2_UP
25	TIM2_TRIG
26	TIM3_CH1
27	TIM3_CH2
28	TIM3_CH3

DMA_CSELR.CS[4:0]	请求源
29	TIM3_CH4
30	TIM3_UP
31	TIM3_TRIG

注意：DMA 的每一个通道，都可以选择表8-3 中的请求源，但是不能将 2 个通道都配置为同一个请求源（一个通道对应一个请求源）。

8.2 DMA 通道寄存器

基地址：（DMA_Channel1, DMA_Channel2, DMA_Channel3, DMA_Channel4, DMA_Channel5）=（0x40020000, 0x40020020, 0x40020040, 0x40020060, 0x40020080）

空间大小：（DMA_Channel1, DMA_Channel2, DMA_Channel3, DMA_Channel4, DMA_Channel5）=（0x20, 0x20, 0x20, 0x20, 0x20）

8.2.1 DMA 通道中断状态寄存器（DMA_ISR）

偏移地址：0x00

复位值：0x00000000

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												TEIF	HTIF	TCIF	GIF
												r	r	r	r

位 31:4	Res：保留 必须保持复位值。
位 3	TEIF：传输错误标志（Transfer error interrupt flag） 此位由硬件置 1，由软件清零（软件只需将 1 写入 DMA_IFCR 寄存器的相应位即可）。 ● 0：无传输错误 ● 1：出现传输错误
位 2	HTIF：半传输标志（Half transfer interrupt flag） 此位由硬件置 1，由软件清零（软件只需将 1 写入 DMA_IFCR 寄存器的相应位即可）。 ● 0：无半传输事件 ● 1：发生半传输事件
位 1	TCIF：传输完成标志（Transfer complete interrupt flag） 此位由硬件置 1，由软件清零（软件只需将 1 写入 DMA_IFCR 寄存器的相应位即可）。 ● 0：无传输完成事件 ● 1：发生传输完成事件
位 0	GIF：全局中断标志（Global interrupt flag） 此位由硬件置 1，由软件清零（软件只需将 1 写入 DMA_IFCR 寄存器的相应位即可）。 ● 0：无 TE、HT 或 TC 事件中的任何一个 ● 1：发生了 TE、HT、或 TC 事件

8.2.2 DMA 通道中断标志清零寄存器（DMA_IFCR）

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												CTEIF	CHTIF	CTCIF	CGIF
												w	w	w	w

位 31:4	Res：保留 必须保持复位值
位 3	CTEIF：通道传输错误标志清零（Clear transfer error interrupt flag） 此位由软件置 1。 ● 0：无操作 ● 1：将 DMA_ISR 寄存器中对应的 TEIF 标志清零
位 2	CHTIF：通道半传输标志清零（Clear half transfer interrupt flag） 此位由软件置 1。 ● 0：无操作 ● 1：将 DMA_ISR 寄存器中对应的 HTIF 标志清零
位 1	CTCIF：通道传输完成标志清零（Clear transfer complete interrupt flag） 此位由软件置 1。 ● 0：无操作 ● 1：将 DMA_ISR 寄存器中对应的 TCIF 标志清零
位 0	CGIF：通道全局中断标志清零（Clear global interrupt flag） 此位由软件置 1。 ● 0：无操作 ● 1：将 DMA_ISR 寄存器中对应的 GIF 标志位清零

8.2.3 DMA 通道配置寄存器（DMA_CCR）

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	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	rw	rw		rw		rw		rw	rw	rw	rw	rw	rw	rw	rw

位 31:15	Res：保留 必须保持复位值
位 14	MEM2MEM：存储器到存储器模式（Memory to memory） ● 0：不使能存储器到存储器模式

	● 1：使能存储器到存储器模式
位 13:12	PL[1:0]：通道优先级配置位（Priority level） ● 00：低优先级 ● 01：中优先级 ● 10：高优先级 ● 11：最高的优先级
位 11:10	MSIZE[1:0]：存储器端数据位宽（Memory data size） ● 00：8 位 ● 01：16 位 ● 10：32 位 ● 11：保留
位 9:8	PSIZE[1:0]：外设端数据位宽（Peripheral data size） ● 00：8 位 ● 01：16 位 ● 10：32 位 ● 11：保留
位 7	MINC：存储器地址递增模式控制（Memory address increase control） ● 0：不使能存储器地址递增模式 ● 1：使能存储器地址递增模式
位 6	PINC：外设地址递增模式控制（Peripheral address increase control） ● 0：不使能外设地址递增模式 ● 1：使能外设地址递增模式
位 5	CIRC：循环传输模式控制（Circle transfer control） ● 0：不使能循环传输模式 ● 1：使能循环传输模式
位 4	DIR：数据传输方向控制（Data transfer direction control） ● 0：从外设读取数据 ● 1：从存储器读取数据
位 3	TEIE：传输错误中断使能控制（Transfer error interrupt enable control） ● 0：不使能传输错误中断 ● 1：使能传输错误中断
位 2	HTIE：半传输中断使能控制（Half transfer interrupt enable control） ● 0：不使能会传输中断 ● 1：使能半传输中断
位 1	TCIE：传输完成中断使能控制（Transfer complete interrupt enable control） ● 0：不使能传输完成中断 ● 1：使能传输完成中断
位 0	EN：通道使能控制（Channel enable control） ● 0：不使能当前 DMA 通道（关闭通道）

	● 1：使能当前 DMA 通道（打开通道）

8.2.4 DMA 通道数据数寄存器（DMA_CNDTR）

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
NDT[15:0]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	NDT[15:0]：待传输的数据数目（Numbers of data to transfer ） 当前通道待传输的数据个数（0 至 65535）。只有在通道关闭（非使能状态）时才能对该寄存器执行写操作。通道使能后，该寄存器为只读，用于指示待传输的剩余传输次数。 DMA 每执行一次传输，该寄存器的值减 1。传输完成后，该寄存器的值可以保持为零，若已将通道配置为循环模式，则自动重新加载先前编程的值。若该寄存器的值为 0，则无论该通道是否使能，都不会处理任何事务。

8.2.5 DMA 通道外设地址寄存器（DMA_CPAR）

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
PA[31:16]															
rw															

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

位 31:0	PA[31:0]：外设地址（Peripheral address） 读/写数据的外设数据寄存器的基地址。 ● 若 PSIZE[1:0] = 01（16 位），则忽略 PA[0]，访问将自动对齐到半字地址。 ● 若 PSIZE[1:0] = 10（32 位），则忽略 PA[1:0]，访问将自动对齐到字地址。

8.2.6 DMA 通道存储器地址寄存器（DMA_CMAR）

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
MA[31:16]															
rw															

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

位 31:0	MA[31:0]：存储器地址（Memory address）

	读/写数据的存储器的基地址。 ● 若 MSIZE[1:0] = 01（16 位），则忽略 MA[0]，访问将自动对齐到半字地址。 ● 若 MSIZE[1:0] = 10（32 位），则忽略 MA[1:0]，访问将自动对齐到字地址。

8.2.7 DMA 通道选择寄存器（DMA_CSELR）

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											CS[4:0]				
											rw				

位 31:5	Res：保留 必须保持复位值。
位 4:0	CS[4:0]：通道请求选择（Channel request selection） 请参考 DMA 请求映射表。 DMA 的请求映射，请参考表 8-3。

9 中断和事件（NVIC & EXTI）

9.1 嵌套向量中断控制器（NVIC）

9.1.1 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内核异常等中断。

- 22 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（使用了2 位的中断优先级）
- 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

9.1.2 系统嘀嗒校准值寄存器

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

9.1.3 中断和异常向量

表 9-1 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 时钟安全系统连接到 NMI 向量 （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	RTC	RTC 中断（RTC global interrupt）	0x0000 0048
3	10	可配置	FLASH	Flash 全局中断（Flash global interrupt）	0x0000 004C
4	11	可配置	RCC	RCC 全局中断（RCC global interrupt）	0x0000 0050

位置	优先级		名称	描述	地址
5	12	可配置	EXTI0_1	EXTI 线[1:0]中断（EXTI Line[1:0] interrupt）	0x0000 0054
6	13	可配置	EXTI2_3	EXTI 线[3:2]中断（EXTI Line[3:2] interrupt）	0x0000 0058
7	14	可配置	EXTI4_15	EXTI 线[15:4]中断（EXTI Line[15:4] interrupt）	0x0000 005C
8	15	可配置	DMA1_CH1	DMA1 通道 1 中断（DMA1 Channel 1 interrupt）	0x0000 0060
9	16	可配置	DMA1_CH2_3	DMA1 通道 2/3 中断（DMA1 Channel 2/3 interrupt）	0x0000 0064
10	17	可配置	DMA1_CH4_5	DMA1 通道 4/5 中断（DMA1 Channel 4/5 interrupt）	0x0000 0068
11	18	可配置	ADC	ADC 中断（ADC global interrupt）	0x0000 006C
12	19	可配置	TIM1	TIM1 全局中断（TIM1 global interrupt）	0x0000 0070
13	20	可配置	UART1	UART1 全局中断（UART1 global interrupt）	0x0000 0074
14	21	可配置	TIM2	TIM2 全局中断（TIM2 global interrupt）	0x0000 0078
15	22	可配置	TIM3	TIM3 全局中断（TIM3 global interrupt）	0x0000 007C
16	23	可配置	TIM6	TIM6 全局中断（TIM6 global interrupt）	0x0000 0080
17	24	可配置	I2C1_SPI1	I2C1 和 SPI1 全局中断（I2C1 and SPI1 global interrupt）	0x0000 0084
18	25	可配置	UART2	UART2 全局中断（UART2 global interrupt）	0x0000 0088
19	26	可配置	EMACC	EMACC 全局中断（EMACC global interrupt）	0x0000 008C
20	27	可配置	DVSQ	DVSQ 全局中断（DVSQ global interrupt）	0x0000 0090
21	28	可配置	COMP1_COMP2_COMP3_COMP4	COMP1/COMP2/COMP3/COMP4 全局中断（COMP1/COMP2/COMP3/COMP4 global interrupt）	0x0000 0094

9.2 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，向电源管理模块输出唤醒请求。

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 可选择上升沿/下降沿触发，挂起状态寄存器中记录中断状态。除硬件触发中断外，还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。而固定 EXTI 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 22 个内外部中断/事件。每条输入线的中断或事件均可独立屏蔽。

9.2.1 主要特性

- 支持多达 22 个事件/中断请求

- 22 根可配置 EXTI 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
- 1 根固定 EXTI 线

- 每根中断/事件线都可单独被触发和屏蔽。
- 检测脉冲宽度低于 APB2 时钟宽度的外部信号。

9.2.2 框图

EXTI 结构框图如下所示：

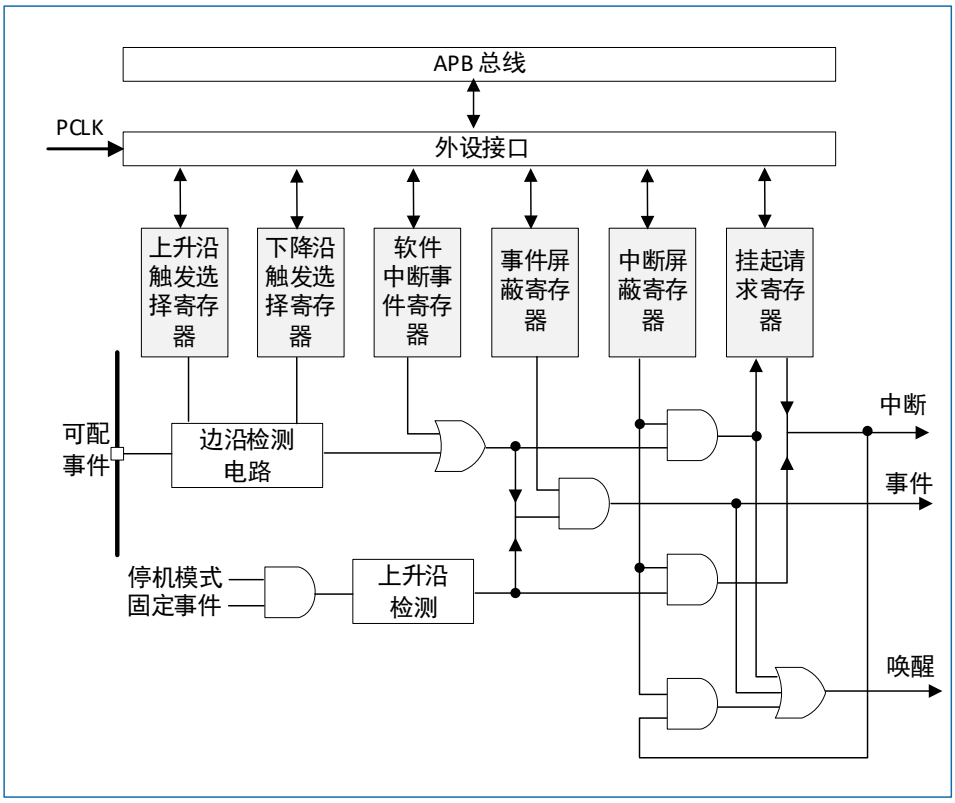

图 9-1 EXTI 框图

9.2.3 EXTI 与周边模块关系

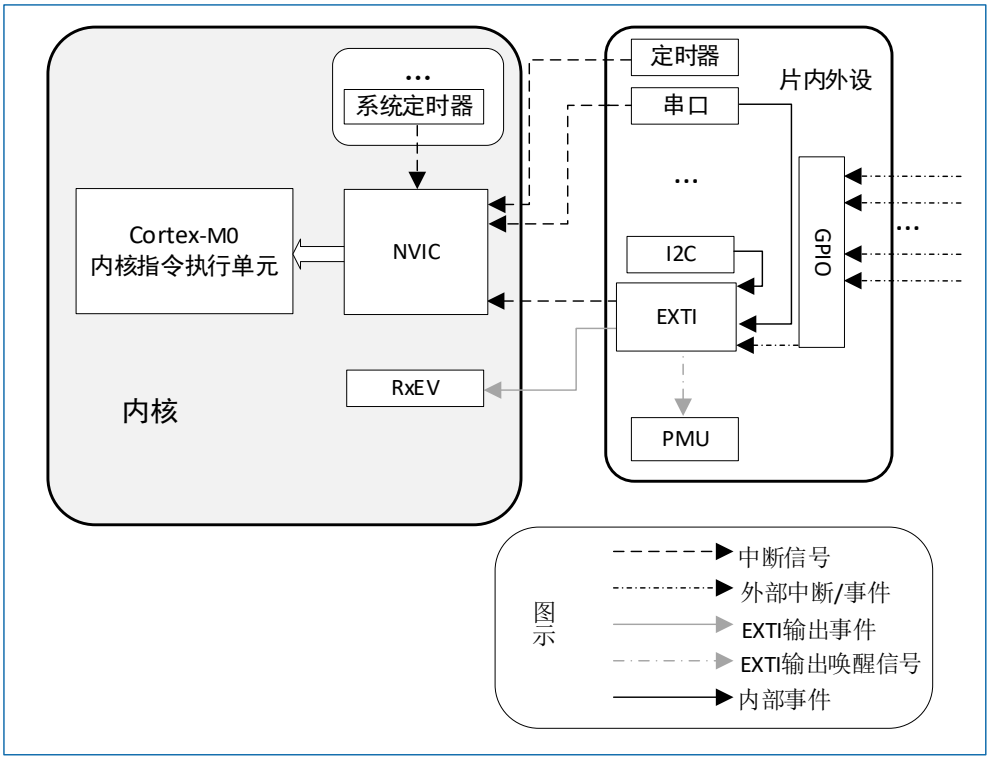

图 9-2 EXTI 与周边模块关系框图

如图 9-2 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 PMU 单元。

9.2.4 唤醒事件管理

HK32ASPIN06x 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配置产生：

- 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，请参见“9.2.5 功能说明”。

9.2.5 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 默认中断未屏蔽。但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。

9.2.5.1 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。

9.2.5.2 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）。

9.2.5.3 软件中断/事件的选择

可配 EXTI 线可以由软件模拟生成相应的中断/事件，操作步骤如下：

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）

9.2.6 外部中断/事件线映射

GPIO 口以下图的方式连接到 16 个外部 EXTI 口（EXTI 0 ~ EXTI 15）：

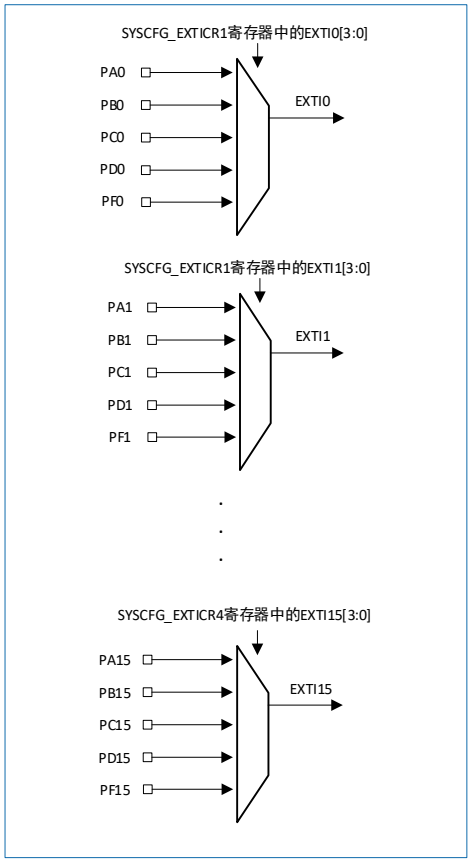

图 9-3 外部中断 GPIO 映射

说明：图 9-3 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 9-3 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_APB2ENR.SYSCFGEN 位（参见“5.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR）”）。

全部 EXTI 线的中断事件映射关系如表 9-2 所示：

表 9-2 EXTI 中断事件映射关系表

EXTI 序号	信号	EXTI 类型
0-15	GPIO	可配
16	保留	-
17	RTC	可配
18	COMP1	可配
19	COMP2	可配
20	COMP3	可配
21	COMP4	可配
22	I2C1 唤醒	固定

其中 EXTI22 作为固定事件没有 RTSR、FTSR、SWIER 和 PR 寄存器，只能在停机（Stop）模式下采事件的上升沿产生 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。

9.3 EXTI 寄存器

基地址：0x4001 0400

空间大小：0x400

9.3.1 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x00

复位值：0x0040 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									IM22	IM21	IM20	IM19	IM18	IM17	Res
									rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:23	Res：保留 必须保持复位值。
位 x （x = 22..17）	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 16	Res：保留

	必须保持复位值。
位 x （x = 15..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

9.3.2 事件屏蔽寄存器（EXTI_EMR）

偏移地址：0x04

复位值：0x0040 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									EM22	EM21	EM20	EM19	EM18	EM17	Res
									rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:23	Res：保留 必须保持复位值。
位 x （x = 22..17）	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 16	Res：保留 必须保持复位值。
位 x （x = 15..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

9.3.3 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										RT21	RT20	RT19	RT18	RT17	Res
										rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22	Res：保留 必须保持复位值。
位 x （x = 21..17）	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）
位 16	Res：保留 必须保持复位值。

位 x （x = 15..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）

9.3.4 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										FT21	FT20	FT19	FT18	FT17	Res
										rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22	Res：保留 必须保持复位值。
位 x （x = 21..17）	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）
位 16	Res：保留 必须保持复位值。
位 x （x = 15..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）

9.3.5 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										SWI21	SWI20	SWI19	SWI18	SWI17	Res
										rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWI15	SWI14	SWI13	SWI12	SWI11	SWI10	SWI9	SWI8	SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22	Res：保留 必须保持复位值。
位 x （x = 21..17）	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。
位 16	Res：保留 必须保持复位值。

位 x （x = 15..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。

9.3.6 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										PIF21	PIF20	PIF19	PIF18	PIF17	Res
										rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIF15	PIF14	PIF13	PIF12	PIF11	PIF10	PIF9	PIF8	PIF7	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:22	Res：保留 必须保持复位值。
位 x （x = 21..17）	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，可清除该位。
位 16	Res：保留 必须保持复位值。
位 x （x = 15..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，可清除该位。

10 模拟数字转换（ADC）

该系列芯片模数转换模块（ADC）包括一组 ADC 转换模块，两个采样保持（采保）单元。两个采保单元可以同时操作，也可以使用单个采保，每个队列都可以单独配置请求源。

10.1 ADC 的主要特性

- 高性能
  - 12 位分辨率
  - ADC 工作时钟可达到 24 MHz，12 位采样，ADC 采样转换速率高达 1.7MSPS。
  - ADC 转换时间：在 1MSPS 的转换速率下，12 位分辨率对应的转换时间为 1μs。
  - ADC 具有可编程采样时间：所有队列可配置不同 ADC 采样时间。
  - 支持 DMA，每个独立队列及公用通道均可独立发起 DMA 请求，队列转换中的每一个通道数据转换完成，均可发起 DMA 请求。
  - 转换数据统一的右对齐方式，结构清晰。
- 低功耗
  - PCLK 大于等于 ADCCLK 的 2 倍，应用可降低 PCLK 频率从而以低功耗运行，采保单元可配置为低功耗模式。
  - 采样保持单元和 ADC 模块可以分别禁能，以节省 ADC 功耗。
- 同步时钟模式
- 灵活的采保单元
  - 两个独立的采样保持单元，两个采保单元均可从 14 个模拟输入通道中选择。其中 10 个为外部输入通道，4 个为内部模拟信号（OPAMP1_OUT/OPAMP2_OUT/OPAMP3_OUT/VOLTAGE_DIV）。
  - 采样时间可编程，1~8 个 ADC 时钟周期可调。
- 模拟输入通道
  - 通道 0-9 为 10 路外部模拟输入
  - 在采保模式（双采保/单采保）模式下：
    - 通道 10 用于运放 1 的输出（OPAMP1_OUT）
    - 通道 11 用于运放 2 的输出（OPAMP2_OUT）
    - 通道 12 用于运放 3 的输出（OPAMP3_OUT）
    - 通道 13 用于 DAC 8 位的输出（VOLTAGE_DIV）
  - 在 BK（backup）模式下：
    - BK 通道 11 为 OPAMP1_OUT
    - BK 通道 13 为 OPAMP2_OUT
    - BK 通道 14 为 OPAMP3_OUT
    - BK 通道 15 为 DAC 8 位的输出（VOLTAGE_DIV）
    - BK 通道 16 为内部参考电压（VREFINT）的通道（仅支持通过测试队列采样）
    - BK 通道 17 为内部 Core 电压（LDO）的通道（仅支持通过测试队列采样）

说明：不同系列/封装的芯片支持的外部模拟输入通道数量有所不同，详见该系列数据手册中的说明。

- 灵活的队列模式

- 4 个独立的常规队列和 1 个测试队列。
- 每个常规队列至多可配置：
  - 单路/双路采保模式下 8 个转换通道。
  - 单路采保扫描模式 1/2 和 BK 模式下 14 个转换通道。
- 测试队列只能配置 1 个转换通道，可以从 14 个采保模拟输入通道和 16 个 BK 模式模拟输入通道（通道 10/12 不可使用）中选择。当测试队列工作在不同模式时，模拟输入通道有所不同；
- 每个队列采用单独的采样时间配置。

- 启动方式
  - 常规队列和测试队列都可通过软件启动。
  - 常规队列和测试队列，都可配置不同的硬件触发：TIM1_TRGO、TIM1_CC4、TIM1_CC5、TIM1_CC6、TIM2_TRGO、TIM3_TRGO 输入事件。
  - 通过外部 GPIO 触发（GPIO 的 AF 功能具有 EXT_TRIG 的 IO 均可作为触发源）。
  - 队列中通道转换的启动可以配置多个触发源同时使用。
- 转换模式
  - 扫描模式，可以按通道号从大到小的扫描通道，也可以从小到大的扫描通道。
  - 单次模式，一次触发即可对选定队列里的所有通道执行一次转换。
  - 循环模式，一次触发即可循环转换选定队列里的所有通道。
  - 间断模式，每一次触发转换队列中的一个通道元素。
- 灵活仲裁机制
  - 常规队列和测试队列，每个队列可配置 0-3 四个级别优先级，数字越高优先级越高。
  - 同优先级的队列发生抢占时，等当前队列转换完成后，再开始转换。
  - 高优先级队列，可在低优先级队列当前转换通道转换完成以后，抢占低优先级队列的使用权。高优先级转换完成后，恢复低优先级队列的转换。
- 数据平均功能
  - 每个队列里的元素可实现平均值功能，最大可平均 8 个通道，通过 ADC_SQUEx_AVERAGE 寄存器来配置。
- 在每次转换完成、队列转换结束，发生比较告警或溢出事件时产生中断。
- ADC 电源要求：
  - HK32M060 系列：2.2~5.5V
  - HK32M063C 系列：4.5~5.5V
  - HK32M066B 系列：3.3~5.5V
- ADC 输入范围：VSS≤VIN≤5.5V。
- 集成了 2 个模拟多路复用控制器 MUX0/MUX1，支持使用片内采样保持方案时，对三相电机信号进行三选二的选通。
- 灵活的比较模式
  - 单独配置下溢比较门限值和上溢比较门限值，并且产生对应的比较告警中断。
- 通道替换功能
  - 可以将通道的转换请求重映射到其它通道，可以利用这个特性测量同一路输入通道并将转换结果保存在其他通道元素的结果寄存器中。以节省软件负载。
  - 通道替换只能在常规队列中进行。

- 独立的结果寄存器
  - 15 个通道都有自己独立的结果寄存器，可保存当前转换值。
- 触发延时可配置功能
  - 内部或外部输入的触发信号产生以后，延时一段时间再启动 ADC 转换。
- 外部 GPIO 触发输入的数字滤波功能
  - IO 输入的触发信号可用数字滤波过滤高频干扰信号，脉宽小于 6~7 触发时钟周期的触发信号会被过滤掉。

10.2 ADC 功能描述

ADC 功能框图如下：

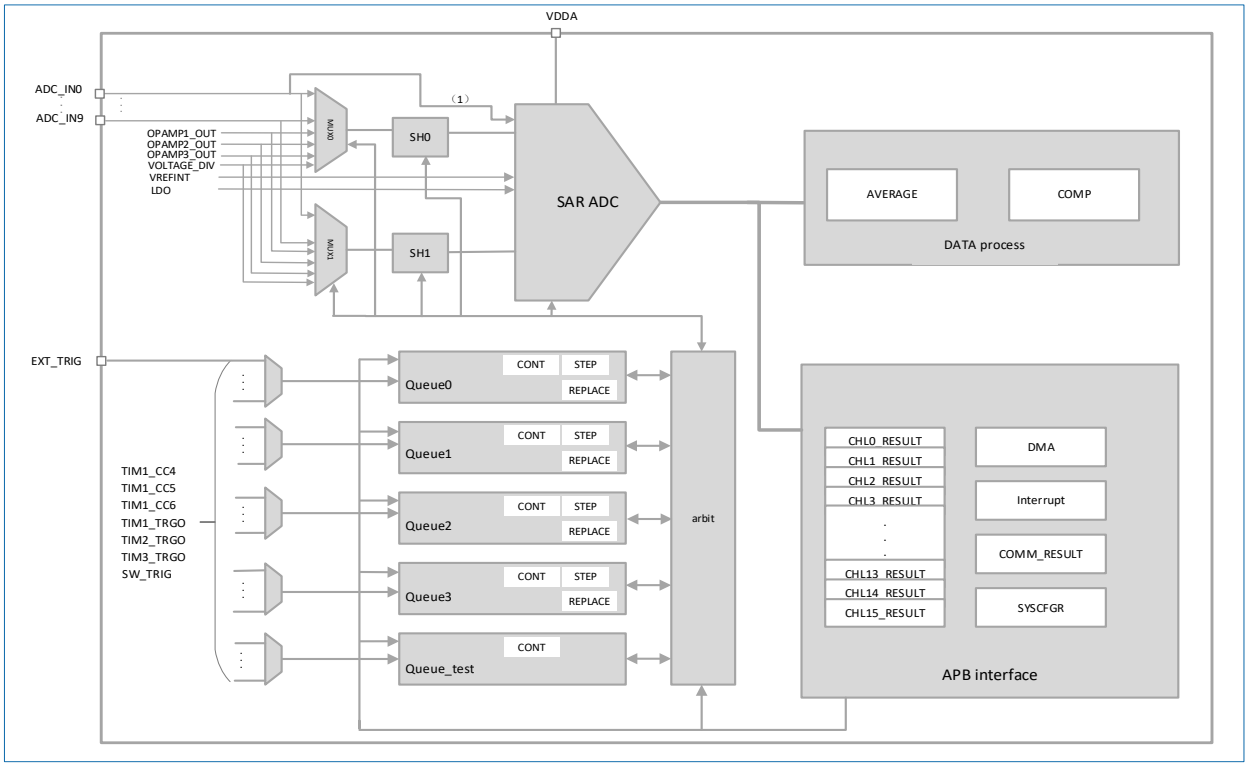

图 10-1 ADC 框架图

上图说明：

(1). 此线表示 BK 模式下，CH0~9 直连到 ADC，此外，OPAMP1/2/3_OUT, VOLTAGE_DIV 也是直连到 ADC，在图中没有逐一画出。

(2). 不同系列/封装的芯片支持的外部模拟输入通道数量有所不同，详见该系列数据手册中的说明。

10.2.1 ADC 引脚和内部信号

表 10-1 ADC 引脚

名称	信号类型	说明
VREFP	ADC 的正参考电压	ADC 的正参考电压等于 V_{DD}
VREFN	ADC 负参考电压	ADC 的负参考电压等于 V_{SS}
ADC_IN[9:0](1)	模拟输入信号	10 个模拟输入通道

说明：不同系列/封装的芯片支持的外部模拟输入通道数量有所不同，详见该系列数据手册中的说明。

表 10-2 ADC 内部信号

内部信号名称	信号类型	说明
VREFINT	输入	内部参考输出电压
LDO	输入	内部 core 电压
VOLTAGE_DIV	输入	内部比较器中集成的 8 位 DAC 电压分频器
OPAMP1_OUT	输入	运放 1 输出
OPAMP2_OUT	输入	运放 2 输出
OPAMP3_OUT	输入	运放 3 输出

10.2.2 ADC 开关控制（ADC_EN，ADC_RDY）

MCU 上电时，ADC 被禁用并进入掉电模式（ADC_EN=0）。

如图 10-2 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 ADC_EN 置 1 可使能 ADC。ADC 准备就绪后，ADC_RDY 标志会立即置 1。
- 将 ADC_EN 置 0 可禁用 ADC 并使 ADC 进入掉电模式。随后，ADC 被完全禁止。

可通过将 SW_TRIG 置 1 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 ADC_RDY 位写 0，将此位清零。
2. 将 ADC_SYSCFGR1 寄存器中的 ADC_EN 位置 1。
3. 等待，直至 ADC_ISR 寄存器中的 ADC_RDY=1（ADC_RDY 会在 ADC 启动时间 t_{STAB} 后置 1。）如果已通过将 ADC_IER 寄存器中的 ADC_RDY_IE 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_SQUEx_CFG1）（x=0~3）寄存器中的 SW_TRIG 为 0，以确保当前未执行任何转换。如有需要，可向（ADC_SQUEx_CFG1）（x=0~3）寄存器中的 SW_STOP 位写入 1 并等待此位读取值为 0，当前转换完成后将停止转换。
2. 将 ADC_EN 清零。
3. 将 ADC_ISR 寄存器中的 ADC_RDY 位编程为 0，将此位清零（可选步骤）。

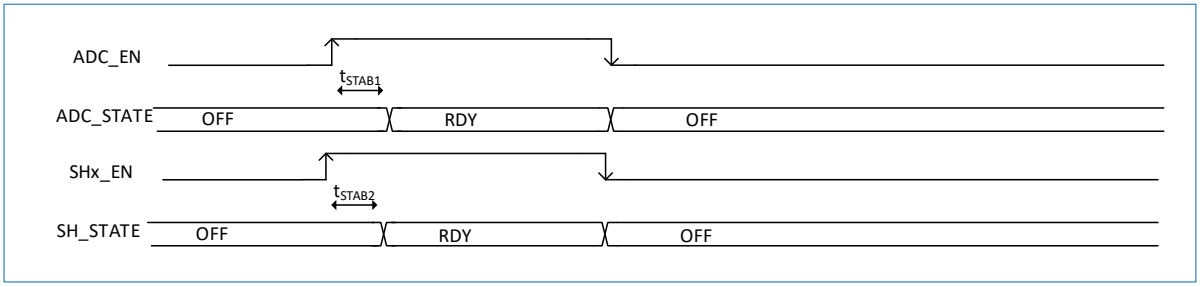

图 10-2 使能/禁用 ADC 或采保

10.2.3 ADC 时钟

通过 RCC_CFGR4. ADCCLKSW 寄存器位选择输入时钟源；随后可对输入时钟源配置数字分频，产生两个时钟：

- CLK_ADC 输入到 ADC 和控制逻辑；
- CLK_TRIG 用于 TRIG 延时计数，也可作为 IO 输入信号数字滤波器的工作时钟。

*注意：要满足 PCLK>=2*CLK_ADC。*

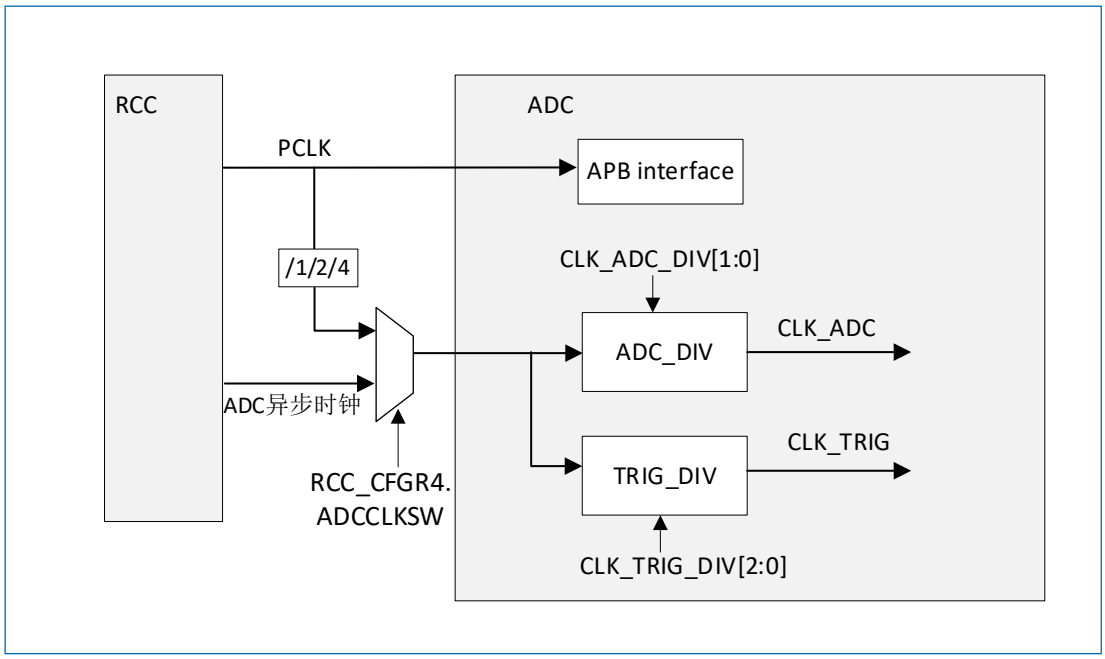

图 10-3 ADC 时钟

10.2.4 配置 ADC

如果 ADC 已禁用（ADC_EN=0），必须通过软件写 ADC_SYSCFGR 寄存器中的 ADC_EN 位为 1 来使能 ADC。

仅当 ADC 已使能，软件才能写 SW_TRIG 位。

如果 ADC 已使能（可能正在进行转换），软件写 ADC_SQUEx_CFG1 寄存器中的 SW_STOP 位才能停止 ADC 转换。

注意：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 ADC_EN 以及 ADC_SYSCFGR 寄存器中的所有位都清零）。

10.2.5 通道选择

复用通道多达 16 路：

说明：不同系列/封装的芯片支持的外部模拟输入通道数量有所不同，详见该系列数据手册中的说明。

14 路可选经过采保的通道：

- 10 路来自 GPIO 引脚的模拟输入（CH0~CH9）
- 1 路用于运算放大器 OPAMP1（OPAMP1_OUT）的采样通道（CH10）
- 1 路用于运算放大器 OPAMP2（OPAMP2_OUT）的采样通道（CH11）

- 1 路用于运算放大器 OPAMP3（OPAMP3_OUT）的采样通道（CH12）
- 1 路用于内部比较器中集成的 8 位 DAC（VOLTAGE_DIV）的采样通道（CH13）

16 路在 BK 模式下接 ADC 的通道：

- 10 路来自 GPIO 引脚的模拟输入（CH0~CH9）
- 1 路用于运算放大器 OPAMP1（OPAMP1_OUT）的采样通道（CH11）
- 1 路用于运算放大器 OPAMP2（OPAMP2_OUT）的采样通道（CH13）
- 1 路用于运算放大器 OPAMP3（OPAMP3_OUT）的采样通道（CH14）
- 1 路用于内部比较器中集成的 8 位 DAC（VOLTAGE_DIV）的采样通道（CH15）
- 1 路用于内部参考电压（VREFINT）的通道（CH16）（仅支持通过测试队列采样）
- 1 路用于内部 Core 电压（LDO）的通道（CH17）（仅支持通过测试队列采样）

常规队列的通道选择：

1. 在单路/双路采保模式下，需要根据转换顺序在元素配置寄存器（ADC_SH_SQUEx_ELEMENT）中，配置 NUM_x_CHL 选择该队列第 x 次转换的通道（可选择 0~13 通道）。最多可配置 8 个对应的通道：NUM_1_CHL~ NUM_8_CHL。单路采保模式中每个元素对应一个通道，双路采保模式中，每个元素对应两个通道。不同元素可重复选择同一通道。详见 10.16.6 常规队列 x 元素配置寄存器（ADC_SH_SQUEx_ELEMENT）（x=0..3）。

   注意：单路采保扫描模式 1/2，及 BK 模式下模式不需要进行此步骤配置。

2. 在常规队列配置寄存器 ADC_SQUEx_CFG1（x=0~3）中使能相应的队列元素：
   - 单路采保模式下，最多可使能 8 个转换元素 E0~E7（每个元素对应一个通道，共计 8 次顺序转换）。
   - 双路采保模式下，最多可使能 4 个转换元素 E0~E4（每个元素对应两个通道，共计 8 次顺序转换）。
   - 单路采保扫描模式 1/2 及 BK 模式下，最多可配置使能 14 个转换元素（BK 模式是：E0~E9，E11/13/14/15；单路采保扫描模式是 E0-13）。元素序号也是其对应的通道序号，比如使能 E0 则使能了本序列的第一次转换通道为通道 0，以此类推。

测试队列的通道选择：

在测试通道配置寄存器 ADC_TEST_CHL_CFG 的 CHL_NUM[4:0]位配置通道号，每次只能配置 1 个通道。

10.2.6 可编程采样时间

开始转换之前，需要配置采样电路的采样时间，以及 ADC 内部的采样间隔，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。采保会在数个时钟间隔后对输入电压进行采样，该时钟间隔可使用 CYCLE_DLY_CFG 寄存器中的 CHL_HOLD[2:0]位进行修改。采样时间可使用 CYCLE_DLY_CFG 寄存器中的 SH_SAMPLE_CYCLE[2:0]位进行修改。

ADC 内部的采样间隔使用 CYCLE_DLY_CFG 寄存器中的 EOC_SOC_DIS[1:0]位来选择。

当使用 BK 模式时，ADC 的采样时间需要单独配置，通过 CYCLE_DLY_CFG 寄存器中的 SAMPLE_CYCLE_x[2:0]（x=0，1，2，3）或 SAMPLE_CYCLE_TEST[2:0]位来修改。

在双采保或者单采保的模式下：

使用采样保持电路的采样时间（双采保或者单采保）：

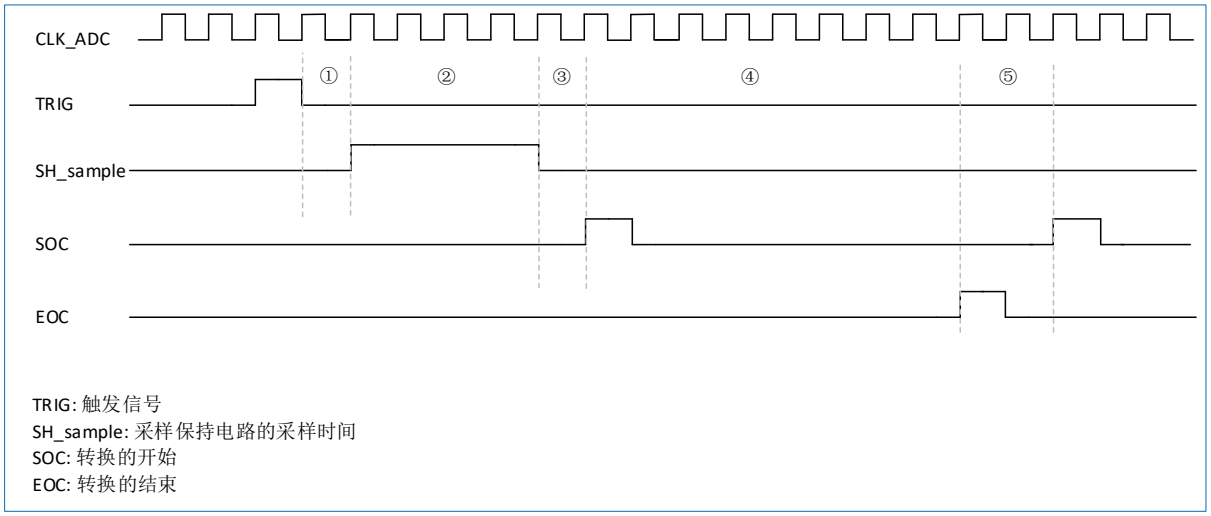

图 10-4 使用采样保持电路的采样时间

上图中的标号说明：

①：CHL_HOLD，ADC 收到 Trig 信号后，通道选择会马上选通寄存器中配置的通道，然后等待 CHL_HOLD 个时钟后，SH_CLK 拉高表示采样保持电路开始进行采样；

②：采样电路的采样阶段，采样时间由 CYCLE_DLY_CFG 寄存器中的 SH_SAMPLE_CYCLE 位来决定；

③：采样电路完成采样后的数据保持时间，时间由 CYCLE_DLY_CFG 寄存器中的 SH_SAMPLE_HOLD 位来决定，同时也是 ADC 的采样阶段；

④：ADC 转换时间，由不同的分辨率来决定；

⑤：ADC 的采样阶段，时间由 CYCLE_DLY_CFG 寄存器中的 EOC_SOC_DIS[1:0]位来决定。

总的转换时间计算公式如下：

$$t_{CONV}=（采样时间+12.5）x\ ADC\ 时钟周期$$

示例：ADC_CLK=28M，CH_HOLD 配置成 2 个 CLK_ADC；SH_SAMPLE_CYCLE 配置成 4 个 CLK_ADC；SH_SAMPLE_HOLD 配置成 2 个 CLK_ADC；EOC_SOC_DIS 配置成 1.5CLK_ADC；分辨率为 12 位（转换时间为 12.5 个 ADC 时钟周期）。

那么：

第一次转换完成：

$t_{CONV}=（CH\_HOLD+SH\_SAMPLE\_CYCLE+SH\_SAMPLE\_HOLD+EOC\_SOC\_DIS+12.5）x\ t_{ADC\_CLK}$

$t_{CONV}=（1.5+8+1+1.5+12.5）x\ t_{ADC\_CLK}=24.5\ x\ t_{ADC\_CLK}=0.875\mu s$

当流水线操作起来以后：

$t_{CONV}=（EOC\_SOC\_DIS+12.5）x\ t_{ADC\_CLK}$

$t_{CONV}=（1.5+12.5）x\ t_{ADC\_CLK}=14\ t_{ADC\_CLK}=0.5\mu s$

不使用采样保持电路的采样时间（BK 模式）：

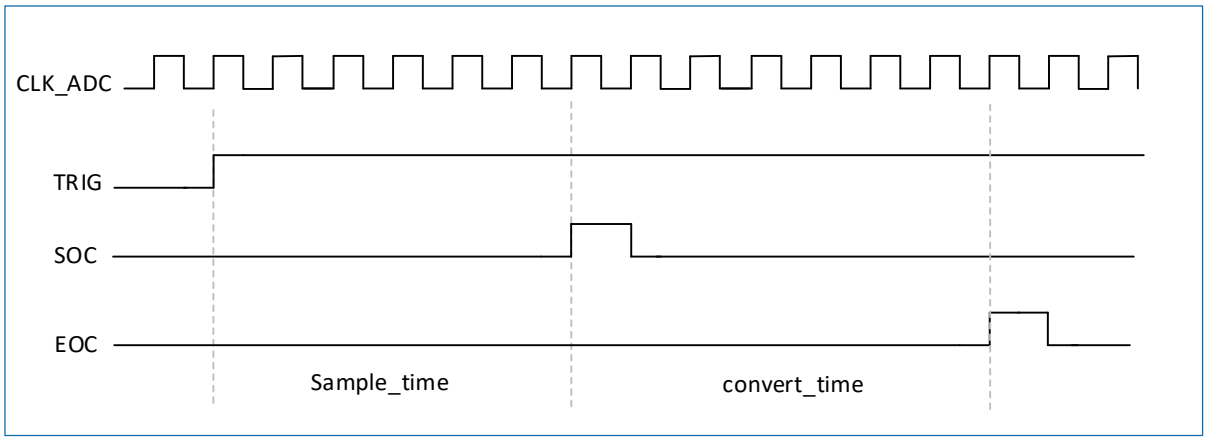

图 10-5 不使用采样保持电路的采样时间

这种情况下 ADC 的采样时间只包括 ADC 本身的采样时间；ADC 的采样时间可以使用 ADC_CYCLE_DLY_CFG 寄存器中的 SAMPLE_CYCLE_x（x=0，1，2，3）进行配置，可配置的范围是 1.5*ADC 时钟周期~239.5*ADC 时钟周期。

总的转换时间计算公式如下（分辨率为 12 位）：

t_{CONV}=采样时间+12.5x t_{ADC_CLK}

示例：ADC_CLK=28M，采样时间配置成 1.5 CLK_ADC，那么：

t_{CONV}=1.5+12.5= 14 x ADC 时钟周期=0.5μs。

10.2.7 队列工作模式

支持以下队列工作模式：

- 双路采保模式
- 单路采保模式
- 单路采保扫描模式 1
- 单路采保扫描模式 2
- BK 模式（非采保）

队列的工作模式取决于队列配置寄存器中的模式配置位，常规队列和测试队列的配置有所不同，具体配置信息参见下表。

表 10-3 队列工作模式配置

队列工作模式	常规队列中此模式的配置	测试队列中此模式的配置
双路采保模式	ADC_SQUEx_CFG1.QUE_MODE[2:0]=000	不支持
单路采保模式	ADC_SQUEx_CFG1.QUE_MODE[2:0]=001	ADC_TEST_CHL_CFG.MODE=0
单路采保扫描模式 1	ADC_SQUEx_CFG1.QUE_MODE[2:0]=011	不支持
单路采保扫描模式 2	ADC_SQUEx_CFG1.QUE_MODE[2:0]=100	不支持
BK 模式（非采保模式）	ADC_SYSCFGR2.BK_ENx=1	ADC_TEST_CHL_CFG. MODE=1

10.2.7.1 双路采保模式

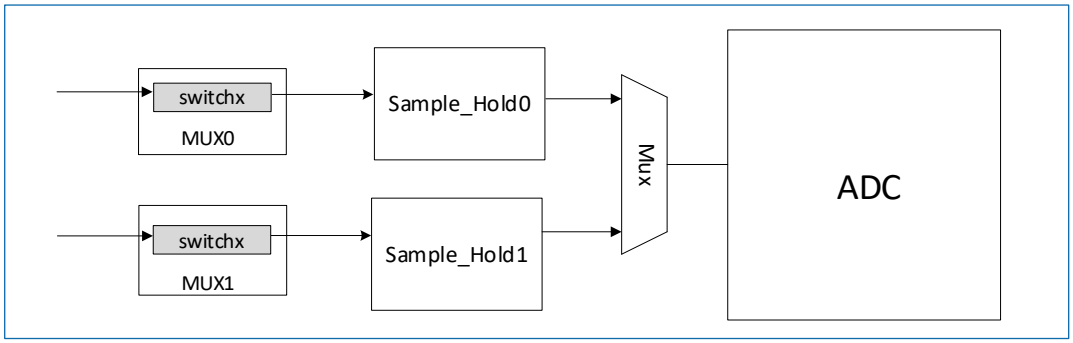

图 10-6 双路采保模式

在双路采保模式下，两个采保单元同时对两个通道进行采样，然后 ADC 依次对采保 0 和采保 1 的数据进行采样转换。

队列的序列被分为四对：NUM_1_CHL 和 NUM_2_CHL、NUM_3_CHL 与 NUM_4_CHL、NUM_5_CHL 与 NUM_6_CHL、NUM_7_CHL 与 NUM_8_CHL 分别配置一对双路采保模式下的两个 AD 转换的通道号。

需要使能元素 E0~E3，其中 E0 对应 NUM_1_CHL 和 NUM_2_CHL， E1 对应 NUM_3_CHL 和 NUM_4_CHL，E2 对应 NUM_5_CHL 和 NUM_6_CHL，E3 对应 NUM_7_CHL 和 NUM_8_CHL。

每个队列采样转换顺序从 NUM_1_CHL 到 NUM_8_CHL。最多采样 8 个通道。

10.2.7.2 单路采保模式

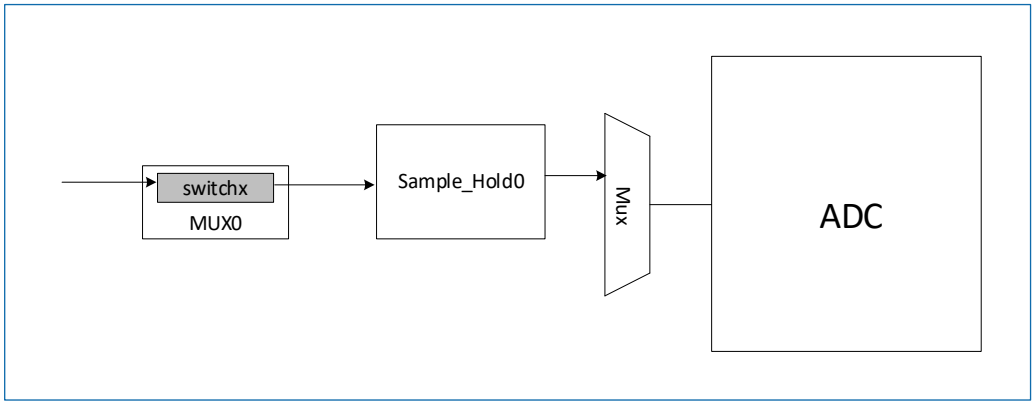

图 10-7 单路采保模式

在单路模式下，只有采保 0 工作，采保 1 关闭。在此模式下，根据队列配置的通道号按次序进行转换。

在 NUM_1_CHL~NUM_8_CHL 分别配置通道号，通过 E0 对应使能元素 NUM_1_CHL，E1 使能元素 NUM_2_CHL，以此类推 E7 使能 NUM_8_CHL。

每个队列采样转换顺序为 NUM_1_CHL 到 NUM_8_CHL。最多采样 8 个通道。

10.2.7.3 BK（Backup）模式

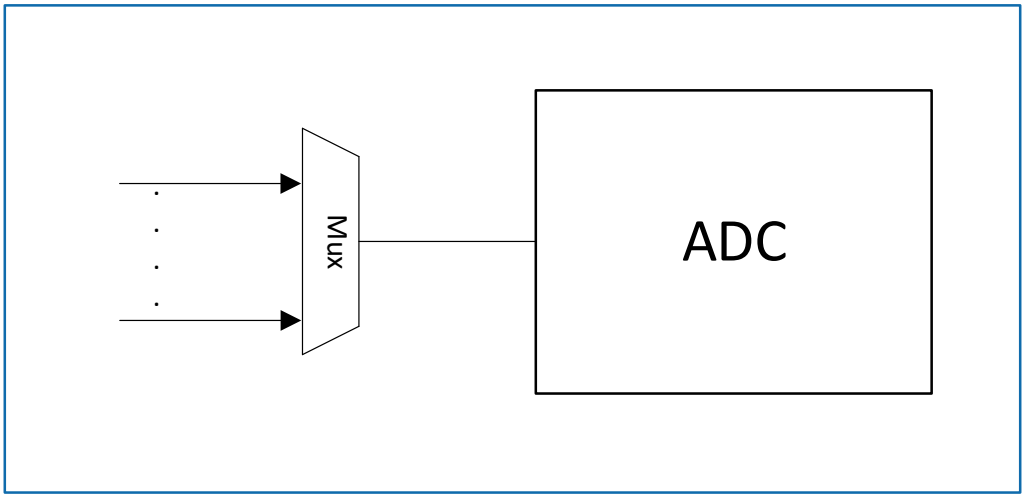

图 10-8 BK（Backup）模式

BK 模式，不使用 ADC 外部的 MUX0 和 MUX1 开关，采样使用 ADC 内部的 Mux 选择。

在 BK 模式下，ADC_SQUEx_CFG1 寄存器中的元素 E0-E15 分别对应 16 个模拟通道，当需要转换某个模拟通道时，将对应的元素设置成 1 即可（BK 模式下没有通道 10/12，所以 E10/12 无需配置）。

10.2.7.4 单路采保扫描模式 1

与单路采保模式相似，只是该模式是将队列使能的通道，按照通道的序号从小到大顺序扫描。

ADC_SQUEx_CFG1 寄存器中的元素 E0-E13 分别对应 14 个模拟通道，当需要转换某个模拟通道时，将对应的元素设置成 1 即可。

10.2.7.5 单路采保扫描模式 2

与单路采保模式相似，只是该模式是将队列使能的通道，按照通道的序号从大到小顺序扫描。

ADC_SQUEx_CFG1 寄存器中的元素 E0-E13 分别对应 14 个模拟通道，当需要转换某个模拟通道时，将对应的元素设置成 1 即可。

10.2.8 常规队列配置流程

配置流程：

1. 配置队列要转换的通道，配置队列工作模式。
2. 配置采样时间（BK 模式下，配置 ADC 的采样时间 SAMPLE_CYCLEx；采保模式下，通道选择到采保采样之间的时间 CHL_HOLD、配置采保的采样时间 SH_SAMPLE_CYCLE、采保的保持时间 SH_SAMPLE_HOLD 和结束转换与开始转换间的时间 EOC_SOC_DIS），选择触发源和触发极性（根据实际应用可选择触发延时。在使用 GPIO 触发下，可启用数字滤波功能）。
3. 配置队列的优先级，选择 ADC 的时钟源，配置 ADC 的时钟分频，使能队列。
4. 使能 ADC 和采保。
5. 等待触发。

示例

使用队列 0 的双采保模式，采样通道 0，和通道 1 的模拟信号，由软件触发转换。

1. NUM_1_CHL= 0x00，NUM_2_CHL= 0x01（配置队列通道）；QUE_MODE=0x00（双采保模式）；E0= 1（使能元素 1）；QUE_EN=1（使能本队列）。

2. 设置 ADC_EN=1、S_H0_EN=1、S_H1_EN=1。

3. 设置 SW_TRIG=1（软件开始转换）。

4. 从 ADC_CHL0_RESULT 寄存器的[11:0]位读出采样通道 0 的值；从 ADC_CHL1_RESULT 寄存器的[11:0]位读出采样通道 1 的值。

10.2.9 测试队列配置流程

测试队列只能工作单路采保模式或者 BK 模式，只能配置 1 个元素，可以配置的通道有 16 个，其他功能与常规队列功能基本一致。

配置流程：

1. 在 ADC_TEST_CHL_CFG 寄存器的 CHL_NUM[4:0]位配置想要转换的通道，MODE 位配置工作模式，在 TRIG_CFG[1:0]位选择触发极性和触发方式，ADC_TEST_CHL_CFG[23:16]位选择触发源，TEST_EN=1 位使能测试模式。

2. ADC_EN=1、S_H0_EN=1。

3. 配置 SW_TRIG=1，（使能软件触发），如果配置了外部触发则等待相应的触发事件触发 AD 转换。

4. 在 ADC_COMM_RESULT 寄存器的[11:0]位读取通道数据。

10.2.10 触发转换工作模式

按照触发信号和转换之间的不同关联来区分，可将 ADC 的工作模式划分为：

- 单次转换模式
- 循环转换模式
- 间断转换模式

10.2.10.1 单次转换模式

在单次转换模式下，ADC 会执行单次队列转换：对队列内的所有通道进行一次转换。当队列配置寄存器中的 CONT=0，（常规队列中还需要确保 STEP=0）时，会选择此模式。

可通过以下任一触发方式开始转换：

- 将队列配置寄存器中的 SW_TRIG 位置 1（软件触发）
- 硬件触发事件

在序列中，每次通道转换完成后：

- 转换后的数据会存储在通道结果寄存器和公用结果寄存器中（测试队列时只会存储在公用结果寄存器中）
- EOC_COMM（通用转换完成标志）置 1
- EOC_COMM_IE 位置 1 时将产生中断

常规队列中的所有通道转换结束后：

- EOC_QUEx（队列转换完成标志）置 1
- EOC_QUEx_IE 位置 1 时产生中断

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

10.2.10.2 循环转换模式（CONT）

在循环转换模式下，如果发生软件或硬件触发事件，ADC 会执行队列转换，对队列内的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队列转换。当队列配置寄存器的 CONT 位配置为 1 时会选择此模式。

可通过以下任一触发方式开始转换：

- 将队列配置寄存器中的 SW_TRIG 位置 1（软件触发）
- 硬件触发事件

在序列中，每次通道转换完成后：

- 转换后的数据会存储在通道结果寄存器或和公用结果寄存器中（测试队列时只会存储在公用结果寄存器中）
- EOC_COMM（通用转换完成标志）置 1
- EOC_COMM_IE 位置 1 时将产生中断

常规队列中的所有通道转换结束后：

- EOC_QUEx（队列转换完成标志）置 1
- EOC_QUEx_IE 位置 1 时产生中断

随后，会立即重启新序列，ADC 会继续重复执行转换序列。如需终止循环转换，常规队列中可配置 SW_STOP=1，ADC 将停止转换；测试队列中，需要将 CONT 改写为 0 以退出循环模式。

注意：CONT 模式和 STEP 模式功能冲突，不能同时使能（建议配置该功能时，队列配置为最低优先级）。

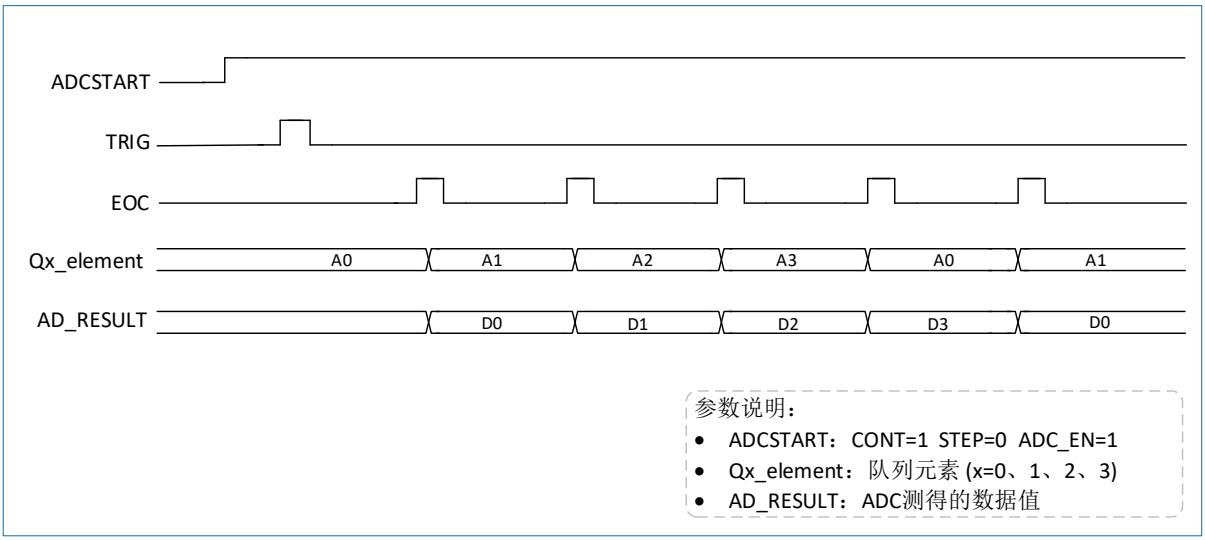

图 10-9 循环模式时序图

10.2.10.3 间断转换模式（STEP）

仅常规队列支持间断转换模式。在间断转换模式下，ADC 会执行单次通道转换，每次触发，ADC 会单次转换队列内的一个通道。当 ADC_SQUEx_CFG2（x=0~3）寄存器中的 STEP 配置为 1 时，会选择此模式。

可通过以下任一触发方式开始转换：

- 将队列配置寄存器中的 SW_TRIG 位置 1（软件触发）
- 硬件触发事件

转换完成后：

- 通道转换后的数据会存储在通道结果寄存器和公用结果寄存器中
- EOC_COMM（通用转换完成标志）置 1
- EOC_COMM_IE 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 SW_TRIG 位再次置 1，继续开始下一通道的转换。

注意：CONT 模式和 STEP 模式功能冲突，不能同时使能。

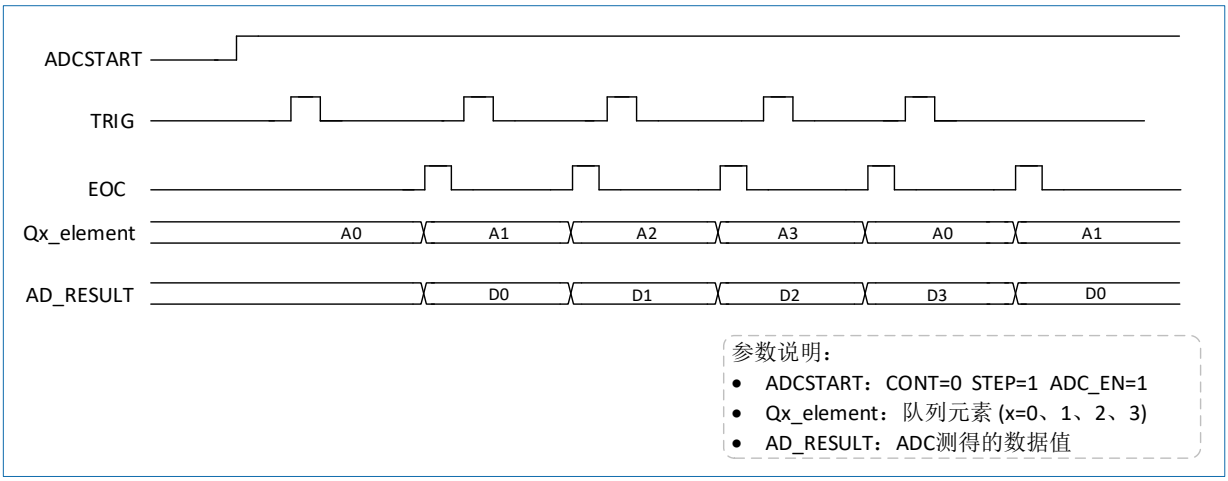

图 10-10 间断模式时序图

10.3 外部触发转换和触发极性

可通过软件（SW_TRIG）或者外部事件（定时器捕获事件或者外部 GPIO）触发队列转换，可支持选用多个不同的触发源，多个触发源可以共同作用或者独立作用。

10.3.1 常规队列的触发配置

- 常规队列可配置 TIM1_TRGO、TIM1_CC4、TIM1_CC5、TIM1_CC6、TIM2_TRGO、TIM3_TRGO、EXT_TRIG（GPIO 输入事件）等触发事件，可以在 ADC_SQUEx_CFG2（x=0~3）寄存器的[31:24]位选择一个或者多个不同的触发源，这些触发源会以逻辑“或”运行。
- 置 ADC_SQUEx_TRIG（x=0~3）寄存器的 XXX_CFG 位，可以选择外部事件的触发极性。
- 配置 ADC_SQUEx_CFG2（x=0~3）寄存器的 TRIG_DLY_CNT[2:0]位，可以配置外部事件的触发延时，使用 CLK_TRIG 时钟计数，0 ~7 个 CLK_TRIG 周期可配。

注意：如果配置 GPIO 输入事件作为触发源，可配置 EXT_TRIG_FILTER=1，开启数字滤波功能，避免噪声误触发。

表 10-4 常规队列配置触发极性

触发极性	ADC_SQUEx_TRIG.xxx_CFG [1:0]
禁止外部事件触发	00
上升沿触发	01
下降沿触发	10
双沿触发	11

表 10-5 常规队列配置触发事件

触发源	ADC_SQUEx_CFG2[31:24]（x=0~3）
TIM1_TRGO	TIM1_TRGO_SEL=1
TIM1_CC4	TIM1_CC4_SEL=1
TIM1_CC5	TIM1_CC5_SEL=1
TIM1_CC6	TIM1_CC6_SEL=1
TIM2_TRGO	TIM2_TRGO_SEL=1
TIM3_TRGO	TIM3_TRGO_SEL=1
EXT_TRIG（GPIO 输入事件）	EXT_TRIG_SEL=1（GPIO 输入事件）

10.3.2 测试队列的触发配置

测试队列也可通过软件或外部事件（定时器捕获事件）触发转换或转换队列。

- 测试队列可配置 TIM1_TRGO、TIM1_CC4、TIM1_CC5、TIM1_CC6、TIM2_TRGO、TIM3_TRGO、EXT_TRIG（GPIO 输入事件）等触发事件，可以在 ADC_TEST_CHL_CFG 寄存器的[23:16]位选择一个或者多个不同的触发源，这些触发源会以“或”的逻辑运行。
- 配置 ADC_TEST_CHL_CFG 寄存器的 TRIG_CFG[1:0]位，可以选择外部事件的触发极性。
- 配置 ADC_TEST_CHL_CFG 寄存器的 TRIG_DLY_CNT[2:0]位，可以配置触发事件的延时响应。

注意：如果配置 GPIO 输入事件作为触发源，可配置 EXT_TRIG_FILTER=1，开启数字滤波功能，避免噪声误触发。

10.4 灵活的仲裁配置

仲裁控制单元，4 个常规队列和 1 个测试队列都可以通过相应的 PRIORITY[1:0]位配置队列优先级（可配置 4 个优先级）。

通过 4 个优先级的配置，一个队列正在转换过程中，又一个新的队列收到触发信号，如果新的队列优先级高于此时正在转换的队列优先级，那么就会中断正在转换的队列，开始新的队列转换，等待队列转换完成后，继续恢复打断的队列；如果新的队列优先级低于或者等于此时正在转换的队列优先级，那么就要等正在转换的队列完全结束，才能开始新的队列转换；如果队列优先级相同，队列按队列序号执行转换，队列 0 优先级最高，测试队列优先级最低。

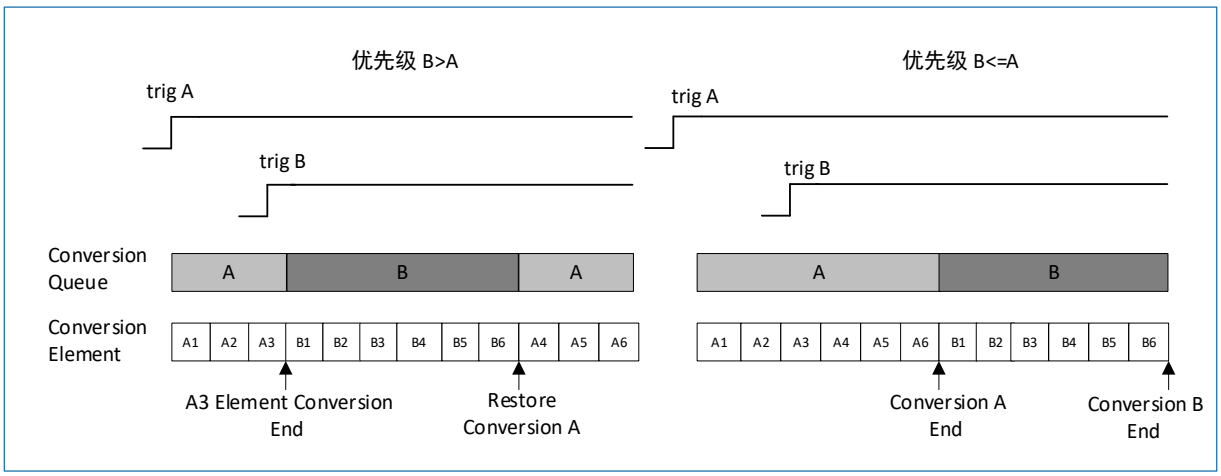

图 10-11 仲裁模式

10.5 数据管理

10.5.1 ADC 数据输出

ADC 通道 0~15 拥有独立的转换结果寄存器 ADC_CHLx_RESULT，CHANNEL_NUM[3:0]位指示数据对应的真实通道号，AD_RESULT[11:0]位为 AD 转换后的结果输出。

除此之外 ADC 还有公用的结果寄存器 ADC_COMM_RESULT[15:0]，每次 ADC 转换后的值都也会更新在该结果寄存器中。CHANNEL_NUM[3:0]位指示数据对应的真实通道号，AD_RESULT[11:0]位为 AD 转换后的结果输出。

注意：测试队列中，通道结果寄存器不可用，通道转换结果仅存储在公用结果寄存器中。

10.5.2 ADC 溢出

如果转换后的数据没有及时被 CPU 或者 DMA 读走，在新转换生成数据之前，可由以下溢出标志指示通道数据溢出事件：

- 通道结果寄存器中的溢出标志：ADC_CHLx_RESULT.OVER_ALM
- 公用结果寄存器中的溢出标志：ADC_CHLx_RESULT.OVER_ALM
- 状态寄存器中的各通道溢出标志：ADC_ISR.OVR0~OVR15
- 状态寄存器中的通用过载告警标志：ADC_ISR.OVR
- 队列过载告警 OVER_ALM
- 如使能了队列平均功能，则还会有队列平均功能寄存器中的过载告警标志：ADC_SQUEx_AVERAGE.OVER_ALM

如果配置了 OVRx_IE（x=0~15）=1 或 OVR_IE=1，可以产生相应的溢出中断。

当发生溢出时保留数据还是覆盖数据：

- OVER_OPT=0，新数据覆盖没被读走的 AD_RESULT。
- OVER_OPT=1，保持没被读走的 AD_RESULT，丢掉新的数据。

10.5.3 使用 DMA 的情况下管理转换数据

每个队列的元素通道的数据更新，和每次 ADC 数据转换都能产生 DMA 请求，使用 DMA 可以提高效率及时读出数据避免数据丢失。

在 ADC_SYSCFGR 寄存器的 DMA_QUE0_EN~DMA_QUE3_EN 和 DMA_COMM_EN 位使能相应的 DMA 请求。当队列或者 ADC 转换完成后，数据将通过 DMA 搬运到相应的存储单元中。

10.6 功耗特性

采样保持和 ADC 模块可以分别被禁能，以节省 ADC 功耗。

当不需要使用某一采样保持模块时，可将其关闭（如 BK 模式下可以关闭两个采保模块；单采保模式中，SH1 采保模块不工作，可将其关闭）。

另外配置 BIAS_CT 位可以配置采样保存电路的高低功耗模式，当 ADC 时钟高于 14M 的时候，BIAS_CT 必须配成 1，以满足采样保持电路的采样时间。

10.7 数据窗口比较功能

注意：仅常规队列支持数据窗口比较功能。

数据比较功能，可以将 ADC 转换后保存于通道结果寄存器中的转换数据与设定的数值进行比较。

在 ADC_COMP_GLBCFG 寄存器中，配置 COMP_UP_EN 和 COMP_DOWN_EN 位使能所有通道的上溢比较和下溢比较功能。

配置 COMP_UP_VALUE 和 COMP_DOWN_VALUE 位来设置相应上/下限比较门限值。

当转换后的数值（ADC_CHLx_RESULT. AD_RESULT[11:0]）超过上限值或者低于下限值时，会在相应通道的结果寄存器（ADC_CHLx_RESULT），以及 ADC 中断状态寄存器中的 COMP_DALM 和 COMP_UALM 发出告警，如果使能相应的中断使能，则会产生中断。COMP_DALM 和 COMP_UALM 告警状态写 0 清零，清零后清除中断请求。

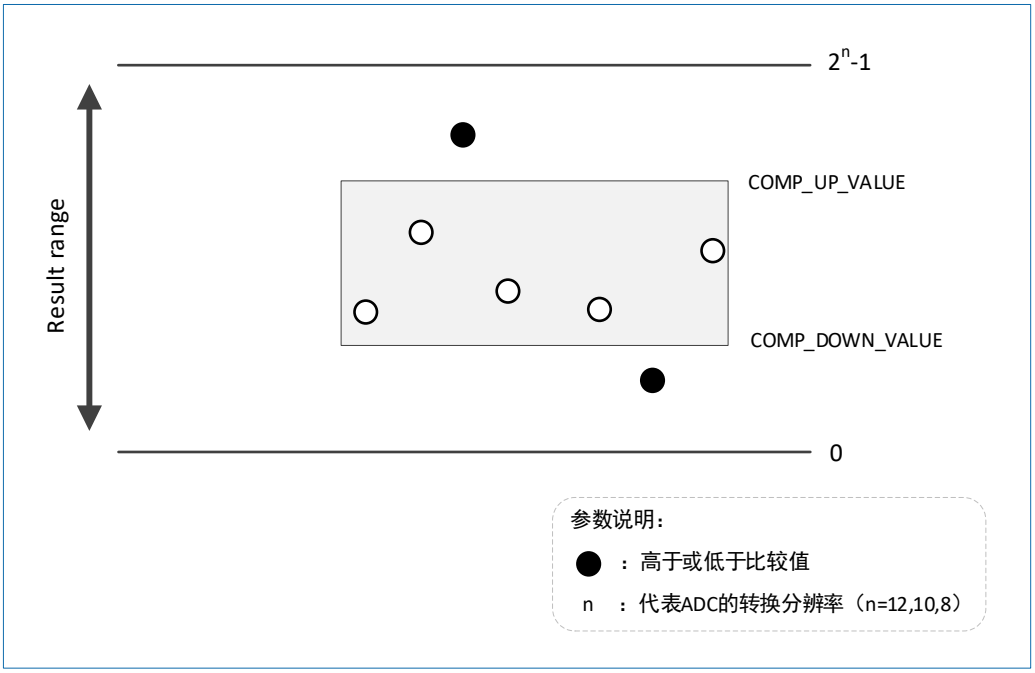

图 10-12 比较功能

10.8 数据平均功能

注意：仅常规队列支持数据平均功能。

内置的数据平均单元会进行数据预处理，以减轻 CPU 的负担。数据平均单元可处理多个转换（最多 8 个），并计算多次转换结果的平均值。通过配置 ADC_SQUEx_AVERAGE（x=0~3）寄存器中的 AVE_EN 位来使能本队列的数据平均功能，配置 D0~D7 来选择参与平均计算的通道元素（D0~D7 与 ADC_SH_SQUEx_ELEMENT 寄存器中的元素 E0-E7 相对应），参与平均计算的通道元素的个数必须是 2 个、4 个或 8 个，计算后的结果存在 AVERAGE_VALUE 位中。

队列平均功能也可以配置溢出模式，溢出处理与普通通道一致，参见章节：“10.5.2 ADC 溢出”。

10.9 通道替代功能

注意：仅单路/双路采保模式下的常规队列支持通道替代功能。

该功能支持将通道的转换请求重指向（一个或多个）其他通道，可以利用这个特性测量同一路输入通道，并将转换结果保存在多个不同的通道结果寄存器中。一个通道可以转换多次而无需读取转换结果，避免数据丢失。

通过配置 ADC_SQUEx_CFG2（x=0~3）中 REPLACE_EN 使能该队列的通道替代功能。所有在 ADC_SH_SQUEx_ELEMENT 寄存器中的做了配置的 NUM_y_CHL 都必须在 ADC_SQUEx_REPLACE 寄存器中重新配置其替换通道。不需要被替换通道的元素其通道配置和 ADC_SH_SQUEx_ELEMENT 中保持一致即可。

通道替代功能只是将目标通道（ADC_SQUEx_REPLACE 配置为的通道）的结果寄存器（ADC_CHLx_RESULT）用于保存原有通道（ADC_SH_SQUEx_ELEMENT 配置的通道）的转换结果，并不会改变该元素的真实采样通道。通道结果寄存器中的 CHANNEL_NUM[3:0]指明了实际的被转换通道号（也就是所存储的转换数据所属的实际通道）。

10.10 与电机加速单元的联动

在电机加速单元的 Clarke 变换中，CLARKE 值载入寄存器（EMACC_CLARKE_LDR）的值由 ADC 的采样值减去偏置来提供。ADC_CHSEL_CTRL_CLARK、ADC_PHAB_OFFSET、ADC_PHAB_OFFSET_R 这三个寄存器控制 ADC 的采样值减去偏置的结果传输到 EMACC_CLARKE_LDR 寄存器，具体配置流程如下：

1. 选择采样电机相电流 PHA（phase A）和 PHB（phase A）的队列和通道：配置 ADC_CHSEL_CTRL_CLARK 寄存器中的 QUE_SEL[1:0]位域选择队列，CHL_PHA[3:0]和 CHL_PHB[3:0]选择采样相电流的通道。
2. 配置相电流 PHA 和 PHB 的偏置：在 ADC_PHAB_OFFSET 寄存器中配置偏置。
3. 使能相电流偏置运算：配置 ADC_CHSEL_CTRL_CLARK 寄存器的 ADC_PHAB_OFFSET_EN 位。
4. 偏置计算的结果保存在 ADC_PHAB_OFFSET_R 寄存器中，并且将计算的结果传到电机加速单元的 CLARKE 值载入寄存器中。计算结果(ADC_PHAB_OFFSET_R) = PHAB 的 ADC 采样值 – 偏置(ADC_PHAB_OFFSET)。

10.11 ADC 增益补偿功能

可以通过使用增益修正参数来修正 ADC 增益误差问题，其配置值为带符号整数，支持-31LSB ~ +31 LSB 的修正范围。

例如，当实测数据斜率不为理想值，推测满幅输入时和理想值差 N 和 LSB，可以通过配置增益修正参数补到理想值。如下图所示：

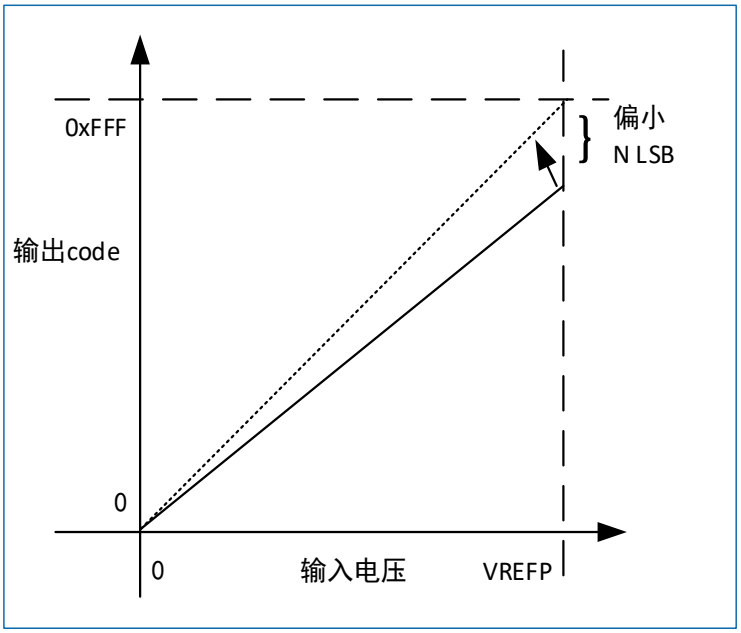

图 10-13 实际数据斜率小于理想斜率，增益修正参数为+N LSB

如图所示，当输入 VREFP 时，转换的 Code 比理想值偏小 N 个 LSB。那么配置为 N（为正数），即可一定程度修正 0~VREFP 范围内的增益误差。

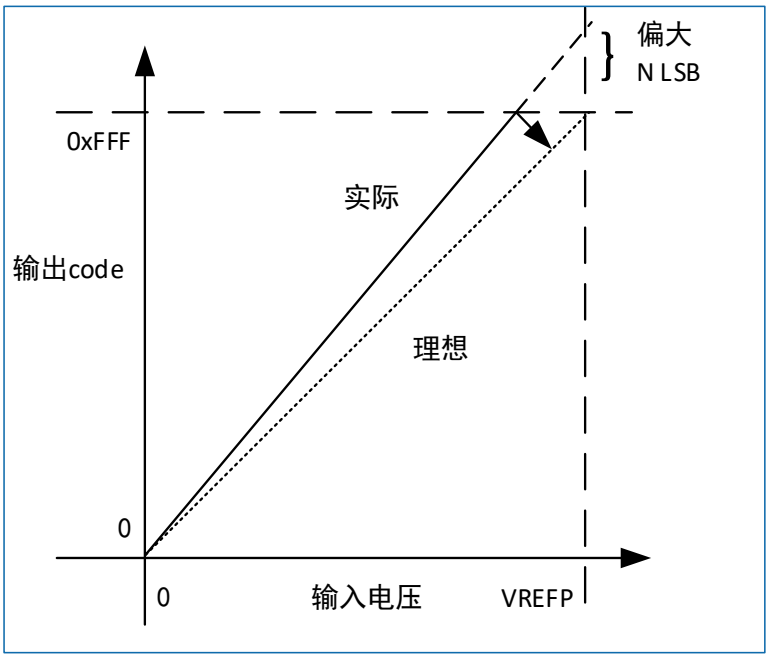

图 10-14 实际数据斜率大于理想斜率，增益修正参数为-N LSB

当推测偏大 N LSB 时，可以配置 –N（注意是带符号的补码），可以一定程度修正回来。

配置流程如下：将修正参数写入 ADC_SYSCFGR2 寄存器中的 GAIN_FIX[5:0]的位域中，然后配置 ADC_SYSCFGR2 寄存器的 GAIN_EN 位使能增益补偿功能。

注意：GAIN_FIX[5:0]的负值需要以补码的形式写入。

10.12 内部参考电压

内部参考电压（VREFINT）为芯片内部一个稳定的（带隙基准）电压输出。VREFINT 内部连接到 ADC_IN13 输入通道（BK 采样通道 16）。VREFINT 的精确电压由航顺在芯片生产测试期间对每颗芯片单独测量，并存储于系统存储区。访问模式为只读。

使用内部参考电压计算实际的 VDDA 电压

施加给器件的 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}=3.3V 的条件下获得的内部参考电压（VREFINT）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：

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

其中：

- VREFINT_CAL 是 VREFINT 校准值，见表 10-6。
- VREFINT_DATA 是由 ADC 转换得到的实际 VREFINT 输出值。

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，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：

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

从而得出以下公式：

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

其中：

- ADC_DATAx 是由 ADC 在通道 x 上测得的值（右对齐）。
- VREFINT_DATA 是由 ADC 转换得到的实际 VREFINT 输出值。
- FULL_SCALE 是 ADC 输出的最大数字值。这里分辨率为 12 位，该值为2^{12}-1=4095。
- VREFINT_CAL 是 VREFINT 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 10-6 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} = 3.3v	0x1FFF F94C~0x1FFF F94D

10.13 内部模拟信号采样

除了从芯片引脚上输入的内部电压信号，芯片内部还有 6 个内部信号如下：

- 1 路用于内部 OPAMP1 输出采样通道（采保模式的 CH10，BK 模式的 CH11）。
- 1 路用于内部 OPAMP2 输出采样通道（采保模式的 CH11，BK 模式的 CH13）。
- 1 路用于内部 OPAMP3 输出采样通道（采保模式的 CH12，BK 模式的 CH14）
- 1 路用于内部比较器中集成的 8 位 DAC（VOLTAGE_DIV）的采样通道（采保模式的 CH13，BK 模式的 CH15）。
- 1 路用于内部参考电压（VREFINT）的采样通道（BK 模式的 CH16，仅支持测试通道队列）。
- 1 路用于内部 Core 电压（LDO）的采样通道（BK 模式的 CH17，仅支持测试通道队列）。

10.14 ADC 中断

发生下列任一事件均可以生成中断：

- ADC 就绪（可以开始 AD 转换）（ADC_RDY 标志）
- 通用转换完成（EOC_COMM 标志）
- 队列 0 转换完成（EOC_QUE0 标志）
- 队列 1 转换完成（EOC_QUE1 标志）
- 队列 2 转换完成（EOC_QUE2 标志）
- 队列 3 转换完成（EOC_QUE3 标志）
- 测试队列转换完成（EOC_TEST 标志）
- 通用过载告警（OVR 标志）
- AD 转换数据低于配置比较值告警（COMP_DALM 标志）
- AD 转换数据高于配置比较值告警（COMP_UALM 标志）
- 通道过载告警（OVRx（x=0~15）标志）

可以使用单独的中断使能位以提高使用灵活性：

表 10-7 中断类型

中断事件	事件标志	使能控制位
ADC 就绪	ADC_RDY	ADC_RDY_IE
通用转换完成	EOC_COMM	EOC_COMM_IE
队列 0 转换完成	EOC_QUE0	EOC_QUE0_IE
队列 1 转换完成	EOC_QUE1	EOC_QUE1_IE
队列 2 转换完成	EOC_QUE2	EOC_QUE2_IE
队列 3 转换完成	EOC_QUE3	EOC_QUE3_IE
测试队列转换完成	EOC_TEST	EOC_TEST_IE
通用过载告警	OVR	OVR_IE
AD 转换数据低于配置比较值告警	COMP_DALM	COMP_DALM_IE
AD 转换数据高于配置比较值告警	COMP_UALM	COMP_UALM_IE
通道过载告警	OVRx（x=0~15）	OVRx_IE（x=0~15）

10.15 DMA 请求

ADC 总共可以发出 5 个 DMA 请求源：4 个队列请求和 1 个 COMM 请求。

每个队列的元素通道的数据更新，和每次 ADC 数据转换都能产生 DMA 请求，使用 DMA 可以提高效率及时读出数据避免数据丢失。

在 ADC_SYSCFGR 寄存器的 DMA_QUE0_EN~DMA_QUE3_EN 和 DMA_COMM_EN 位使能相应的 DMA 请求。当队列或者 ADC 转换完成后，数据将通过 DMA 搬运到相应的用户存储单元中。

10.16 ADC 寄存器

基地址：0x4000 0000

空间大小：0x400

10.16.1 ADC 系统配置寄存器 1（ADC_SYSCFGR1）

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res							BIAS_CT	Res			DMA_COMM_EN	DMA_QUE3_EN	DMA_QUE2_EN	DMA_QUE1_EN	DMA_QUE0_EN
							rw				rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res							CLK_TRIG_DIV[2:0]			CLK_ADC_DIV[1:0]		Res	S_H0_EN	S_H1_EN	ADC_EN
							rw			rw			rw	rw	rw

位 31:25	Res：保留 必须保持复位值。
位 24	BIAS_CT：采保模块功耗模式选择（SH power consumption select）

	此位由软件置 1 和清零，用于选择采保模块功耗模式。 ● 0：低功耗模式 ● 1：高功耗模式 *注意：当 ADC 时钟大于 14M 时，必须配置 BIAS_CT 为 1。*
位 23:21	Res：保留 必须保持复位值。
位 20	DMA_COMM_EN：DMA_ COMM 通道使能（DMA_COMM enable） 此位由软件置 1 和清零，用于使能 DMA_ COMM 通道,使能后每一个 AD 数据转换完成（保存于 ADC_COMM_RESULT 中），将发生一次 DMA 请求。该 DMA 通道用于每一次 AD 转换的数据（ADC_COMM_RESULT 中的数据）搬运。 ● 0：不使能 DMA_ COMM 通道 ● 1：使能 DMA_ COMM 通道
位 19	DMA_QUE3_EN：DMA_QUE3 通道使能（DMA_QUE3 enable） 此位由软件置 1 和清零，用于使能 DMA_QUE3 通道,使能后 QUE3 队列的每一个数据转换完成，将发生一次 DMA 请求。该 DMA 通道用于 que3 队列的（通道寄存器中的）数据搬运。 ● 0：不使能 DMA_QUE3 通道 ● 1：使能 DMA_QUE3 通道
位 18	DMA_QUE2_EN：DMA_QUE2 通道使能（DMA_QUE2 enable） 此位由软件置 1 和清零，用于使能 DMA_QUE2 通道，使能后 QUE2 队列的每一个数据转换完成，将发生一次 DMA 请求。该 DMA 通道用于 que2 队列的数据搬运。 ● 0：不使能 DMA_QUE2 通道 ● 1：使能 DMA_QUE2 通道
位 17	DMA_QUE1_EN：DMA_QUE1 通道使能（DMA_QUE1 enable） 此位由软件置 1 和清零，用于使能 DMA_QUE1 通道，使能后 QUE1 队列的每一个数据转换完成，将发生一次 DMA 请求。该 DMA 通道用于 que1 队列的（通道寄存器中的）数据搬运。 ● 0：不使能 DMA_QUE1 通道 ● 1：使能 DMA_QUE1 通道
位 16	DMA_QUE0_EN：DMA_QUE0 通道使能（DMA_QUE0 enable） 此位由软件置 1 和清零，用于使能 DMA_QUE0 通道，使能后 QUE0 队列的每一个数据转换完成，将发生一次 DMA 请求。该 DMA 通道用于 que0 队列的（通道寄存器中的）数据搬运。 ● 0：不使能 DMA_QUE0 通道 ● 1：使能 DMA_QUE0 通道
位 15:9	Res：保留 必须保持复位值
位 8:6	CLK_TRIG_DIV[2:0]：RCC 输入时钟的分频配置，分频后用于触发延时配置使用（Trigger clk division） 这些位由软件置 1 和清零，用于对 RCC 输入的时源进行分频。分频后的时钟用于触发延时配置使用，还可作为 IO 输入信号的数字滤波器工作时钟。 ● 000：不做分频 ● 001：2 分频 ● 010：4 分频 ● 011：8 分频

	● 100：16 分频 ● 101：32 分频 ● 110：64 分频 ● 111：128 分频
位 5:4	CLK_ADC_DIV[1:0]：RCC 输入时钟的分频配置，分频后输入 ADC 和控制逻辑使用（ADC clk division） 这些位由软件置 1 和清零，用于对 RCC 输入时钟的分频配置，分频后输入 ADC 和控制逻辑使用。 ● 00：不做分频 ● 01：2 分频 ● 10：4 分频 ● 11：8 分频 *注意：因为 PCLK 必须大于等于 2*CLK_ADC，所以当 RCC 配置的 ADC 输入时钟为未分频的 PCLK 时，本寄存器位不能配置为“00”。*
位 3	Res：保留 必须保持复位值。
位 2	S_H0_EN：sample_Hold1 模块使能（Sample_Hold0 enable） 此位由软件置 1 和清零，用于 sample_Hold0 模块。 ● 0：关闭 sample_Hold0 模块 ● 1：sample_Hold0 模块使能，当 ADC_EN 使能时才能正常工作。
位 1	S_H1_EN：sample_Hold1 模块使能（Sample_Hold0 enable） 此位由软件置 1 和清零，用于 sample_Hold1 模块。 ● 0：关闭 sample_Hold1 模块 ● 1：sample_Hold1 模块使能，当 ADC_EN 使能时才能正常工作。
位 0	ADC_EN：ADC 模块使能（ADC enable） 此位由软件置 1 和清零，用于使能 ADC 模块。 ● 0：关闭 ADC ● 1：使能 ADC

10.16.2 ADC 中断状态寄存器（ADC_ISR）

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
OVR15	OVR14	OVR13	OVR12	OVR11	OVR10	OVR9	OVR8	OVR7	OVR6	OVR5	OVR4	OVR3	OVR2	OVR1	OVR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP_UALM	COMP_DALM	CHL_NUM[3:0]				Res	OVR	EOC_TEST	EOC_QUE3	EOC_QUE2	EOC_QUE1	EOC_QUE0	EOC_COMM	Res	ADC_RDY
rc_w0	rc_w0	r					rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0		rc_w0

位 x+16 （x=15..0）	OVRx：通道 x 过载告警标志（Channel x OVR flag） 这些位是只读的，通道 x 对应的寄存器旧数据还没有及时取走，新的数据已转换完成，硬件将标志置 1。如果使能相应中断，同时会产生对应中断。对应通道的 ADC_CHLx_RESULT.OVER_ALM 写 0 会

	使得本寄存器位清零，清零后清除中断请求。 ● 0：过载告警状态没有发生 ● 1：过载告警状态发生
位 15	COMP_UALM：AD 转换数据高于配置比较值告警中断状态（COMP up alarm interrupt status ） 此位是只读的，用于 AD 转换数据高于配置比较值告警中断状态指示，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：AD 转换数据高于配置比较值告警中断状态没有发生 ● 1：AD 转换数据高于配置比较值告警中断状态发生
位 14	COMP_DALM：AD 转换数据低于配置比较值告警中断状态（COMP down alarm interrupt status） 此位是只读的，用于 AD 转换数据低于配置比较值告警中断状态指示，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：AD 转换数据低于配置比较值告警中断状态没有发生 ● 1：AD 转换数据低于配置比较值告警中断状态发生
位 13:10	CHL_NUM[3:0]：比较功能告警对应的真实通道号（COMP alarm channel） 该位域是只读的。指示发生比较告警时，用于比较的转换数据的真实通道。如果替代功能开启，则为替代前的通道（真实采样通道）的值。
位 9	Res：保留 必须保持复位值。
位 8	OVR：通用过载告警中断状态（Common OVR interrupt status） 此位是只读的，公用结果寄存器旧数据还没有及时取走，新的数据已转换完成，硬件将标志置 1。如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：过载告警状态没有发生 ● 1：过载告警状态发生
位 7	EOC_TEST：测试队列转换完成中断状态（Test queue convert done interrupt status） 此位是只读的，测试队列 AD 转换完成硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：测试队列 AD 转换未完成状态（或者测试队列没有进行转换状态） ● 1：测试队列 AD 转换完成状态
位 6	EOC_QUE3：队列 3 转换完成中断状态（Queue3 convert done interrupt status） 此位是只读的，队列 3 AD 转换完成硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：队列 3 AD 转换未完成状态（或者队列 3 没有进行转换状态） ● 1：队列 3 AD 转换完成状态
位 5	EOC_QUE2：队列 2 转换完成中断状态（Queue2 convert done interrupt status） 此位是只读的，队列 2 AD 转换完成硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：队列 2 AD 转换未完成状态（或者队列 2 没有进行转换状态） ● 1：队列 2 AD 转换完成状态
位 4	EOC_QUE1：队列 1 转换完成中断状态（Queue1 convert done interrupt status） 此位是只读的，队列 1 AD 转换完成硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此

	位写 0 清零，清零后清除中断请求。 ● 0：队列 1 AD 转换未完成状态（或者队列 1 没有进行转换状态） ● 1：队列 1 AD 转换完成状态
位 3	EOC_QUE0：队列 0 转换完成中断状态（Queue0 convert done interrupt status） 此位是只读的，队列 0 AD 转换完成硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：队列 0 AD 转换未完成状态（或者队列 0 没有进行转换状态） ● 1：队列 0 AD 转换完成状态
位 2	EOC_COMM：通用 AD 转换完成中断状态（Common convert done interrupt status） 此位是只读的，每一次 AD 转换完成硬件将标志置 1。如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：通用 AD 转换未完成状态（或者 ADC 没有进行转换状态） ● 1：通用 AD 转换完成状态
位 1	Res：保留 必须保持复位值。
位 0	ADC_RDY：ADC 就绪标志（ADC ready status） 此位是只读的，ADC 使能后（ADC_EN=1）并且 ADC 达到可以开始 AD 转换状态时硬件将标志置 1，如果使能相应中断，同时会产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：ADC 未准备好开始 AD 转换（或标志事件已通过软件确认并清零）。 ● 1：ADC 可以开始 AD 转换。

10.16.3 ADC 中断使能寄存器（ADC_IER）

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
OVR15_IE	OVR14_IE	OVR13_IE	OVR12_IE	OVR11_IE	OVR10_IE	OVR9_IE	OVR8_IE	OVR7_IE	OVR6_IE	OVR5_IE	OVR4_IE	OVR3_IE	OVR2_IE	OVR1_IE	OVR0_IE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP_UALM_IE	COMP_DALM_IE	Res					OVR_IE	EOC_TEST_IE	EOC_QUE3_IE	EOC_QUE2_IE	EOC_QUE1_IE	EOC_QUE0_IE	EOC_COMM_IE	Res	ADC_RDY_IE
rw	rw						rw	rw	rw	rw	rw	rw	rw		rw

位 x+16 （x=15..0）	OVRx_IE：通道 x 过载告警中断使能（Channel x OVR interrupt enable） 此位由软件置 1 和清零，用于通道 x 过载告警中断使能。 ● 0：不使能通道 x 过载告警中断 ● 1：使能通道 x 过载告警中断
位 15	COMP_UALM_IE：AD 转换数据高于配置比较值告警中断使能（COMP up alarm interrupt enable） 此位由软件置 1 和清零，用于 AD 转换数据高于配置比较值告警中断使能。 ● 0：不使能 AD 转换数据高于配置比较值告警中断 ● 1：使能 AD 转换数据高于配置比较值告警中断

位 14	COMP_DALM_IE：AD 转换数据低于配置比较值告警中断使能（COMP down alarm interrupt enable） 此位由软件置 1 和清零，用于 AD 转换数据低于配置比较值告警中断使能。 ● 0：不使能 AD 转换数据低于配置比较值告警中断 ● 1：使能 AD 转换数据低于配置比较值告警中断
位 13:9	Res：保留 必须保持复位值。
位 8	OVR_IE：通用过载告警中断使能（Common OVR interrupt enable） 此位由软件置 1 和清零，用于过载告警中断使能。 ● 0：不使能过载告警中断 ● 1：使能过载告警中断
位 7	EOC_TEST_IE：测试转换完成中断使能（Test queue convert done interrupt enable） 此位由软件置 1 和清零，用于测试转换完成中断使能。 ● 0：不使能测试转换完成中断 ● 1：使能测试转换完成中断
位 6	EOC_QUE3_IE：队列 3 转换完成中断使能（Queue3 convert done interrupt enable） 此位由软件置 1 和清零，用于队列 3 转换完成中断使能。 ● 0：不使能队列 3 转换完成中断 ● 1：使能队列 3 转换完成中断
位 5	EOC_QUE2_IE：队列 2 转换完成中断使能（Queue2 convert done interrupt enable） 此位由软件置 1 和清零，用于队列 2 转换完成中断使能。 ● 0：不使能队列 2 转换完成中断 ● 1：使能队列 2 转换完成中断
位 4	EOC_QUE1_IE：队列 1 转换完成中断使能（Queue1 convert done interrupt enable） 此位由软件置 1 和清零，用于队列 1 转换完成中断使能。 ● 0：不使能队列 1 转换完成中断 ● 1：使能队列 1 转换完成中断
位 3	EOC_QUE0_IE：队列 0 转换完成中断使能（Queue0 convert done interrupt enable） 此位由软件置 1 和清零，用于队列 0 转换完成中断使能。 ● 0：不使能队列 0 转换完成中断 ● 1：使能队列 0 转换完成中断
位 2	EOC_COMM_IE：通用 AD 转换完成中断使能（Common convert done interrupt enable） 此位由软件置 1 和清零，用于 ADC 通用转换完成中断使能。 ● 0：不使能通用 AD 转换完成中断 ● 1：使能通用 AD 转换完成中断
位 1	Res：保留 必须保持复位值。
位 0	ADC_RDY_IE：ADC 就绪中断使能（ADC ready enable） 此位由软件置 1 和清零，用于 ADC 就绪的中断使能。

	● 0：不使能 ADC 就绪中断 ● 1：使能 ADC 就绪中断

10.16.4 常规队列 x 配置寄存器 1（ADC_SQUEx_CFG1）（x=0..3）

偏移地址：0x0C + x*4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E15	E14	E13	E12	E11	E10	E9	E8	E7	E6	E5	E4	E3	E2	E1	E0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW_TRIG	SW_STOP	QUE_DONE	QUE_STATE[1:0]		PRIORITY[1:0]		Res					QUE_MODE[2:0]			QUE_EN
rs	rs	rc_w0	r		rw							rw			rw

位 y+16 （y=15..0）	Ey：队列元素 y 使能位（Element y enable） 这些位由软件置 1 和清零，用于使能队列内的元素。 ● 0：不使能队列元素 y ● 1：使能队列元素 y *注意：* ● *单路采保模式中，最多可使能 8 位元素：E0~E7,每位元素对应的通道在 ADC_SH_SQUEx_ELEMENT 寄存器中另行配置。* ● *双路采保模式中，最多可使能 4 位元素：E0~E4，每位元素对应两路通道。每位元素对应的通道在 ADC_SH_SQUEx_ELEMENT 寄存器中另行配置。* ● *BK 模式及单路扫描模式 1/2 中，最多可使能 14 位元素（BK 模式是：E0~E9，E11，E13/14/15；单路采保扫描模式是 E0-13）；使能元素 Ey 也就使能了该元素 Ey 对应的转换通道 Channel y。*
位 15	SW_TRIG：软件触发位（Software trigger） 此位由软件置 1，当队列处于空闲状态时，该位置 1 将触发队列开始转换。当转换结束，队列回到空闲状态时，该位由硬件清零（不可软件清零）。 ● 0：未执行队列转换命令（队列处于空闲状态） ● 1：执行队列转换命令
位 14	SW_STOP：软件队列停止命令位（Software stop） 此位由软件置 1，在工作或者保持状态，配置为 1，则队列停止转换；队列指针归 0，返回到空闲状态，一个 ADCCLK 时钟周期后自动归 0。 ● 0：空闲状态，或者队列工作中 ● 1：执行停止队列转换命令
位 13	QUE_DONE：队列转换完成标志（Queue conversion complete flag） 此位指示队列转换完成，由硬件置位。置位时，如 EOC_QUEx_IE =1，触发相应的中断。由软件清零，同时清除对应中断。 ● 0：队列转换未完成（或者 ADC 转换完成被清除状态） ● 1：队列转换完成
位 12:11	QUE_STATE[1:0]：当前的队列工作状态（Queue state） 这些位指示当前队列的工作状态，由硬件置位和清零。 ● 00：空闲状态，队列空闲，可触发队列，启动 AD 转换。

	● 01：保持状态，队列工作中，因高优先级队列抢占而跳转到保持状态。 ● 10：工作状态，队列工作中。 ● 11：未使用状态，队列不可用状态。
位 10:9	PRIORITY[1:0]：配置队列优先级（Queue priority） 这些位由软件置 1 和清零，用于配置队列优先级。 ● 00：最低优先级 ● .. ● 11：最高优先级 同优先级队列按顺序执行，高优先级队列可插入低优先级队列，处理完后，低优先级队列恢复。
位 8:4	Res：保留 必须保持复位值。
位 3:1	QUE_MODE[2:0]：配置自定义模式的工作子模式（Queue mode configure） 这些位由软件置 1 和清零，用于配置队列自定义的工作子模式。 ● 000：双路采保模式 ● 001：单路采保模式 ● 011：单路采保扫描模式 1，从小通道号扫描到大通道号（根据 Q0-Q15 使能位来扫描） ● 100：单路采保扫描模式 2，从大通道号扫描到小通道号（根据 Q0-Q15 使能位来扫描） ● 其他值：保留 *注意：常规队列的 BK 模式参见章节：“10.16.13 ADC 系统配置寄存器 2（ADC_SYSCFGR2）”。*
位 0	QUE_EN：队列使能（Queue enable） 此位由软件置 1 和清零，用于队列使能。 ● 0：不使能本队列 ● 1：使能本队列

10.16.5 常规队列 x 配置寄存器 2（ADC_SQUEx_CFG2）（x=0..3）

偏移地址：0x1C + x* 4

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIM1_TRGO_SEL	TIM1_CC4_SEL	TIM1_CC5_SEL	TIM1_CC6_SEL	TIM2_TRGO_SEL	TIM3_TRGO_SEL	Res	EXT_TRIG_SEL	Res							REPLACE_EN
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						EXT_TRIG_FILTER	TRIG_DLY_CNT[2:0]			Res				STEP	CONT
						rw	rw							rw	rw

位 31	TIM1_TRGO_SEL：选择 TIM1_TRGO 作为触发转换开始的外部事件（TIM1 TRGO select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_TRGO 为触发源 ● 1：选择 TIM1_TRGO 为触发源
位 30	TIM1_CC4_SEL：选择 TIM1_CC4 作为触发转换开始的外部事件（TIM1 CC4 select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC4 为触发源

	● 1：选择 TIM1_CC4 为触发源
位 29	TIM1_CC5_SEL：选择 TIM1_CC5 作为触发转换开始的外部事件（TIM1 CC5 select） 当配置为 1 时，触发源使能。可支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC5 为触发源 ● 1：选择 TIM1_CC5 为触发源
位 28	TIM1_CC6_SEL：选择 TIM1_CC6 作为触发转换开始的外部事件（TIM1 CC6 select） 当配置为 1 时，触发源使能。可支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC6 为触发源 ● 1：选择 TIM1_CC6 为触发源
位 27	TIM2_TRGO_SEL：选择 TIM2_TRGO 输入的信号（TIM2 TRGO select） 当配置为 1 时，触发源使能。可支持选用多个不同的触发源。 ● 0：不选择 TIM2_TRGO 为触发源 ● 1：选择 TIM2_TRGO 为触发源
位 26	TIM3_TRGO_SEL：选择 TIM3_TRGO 作为触发转换开始的外部事件（TIM3 TRGO select） 当配置为 1 时，触发源使能。可支持选用多个不同的触发源。 ● 0：不选择 TIM3_TRGO 为触发源 ● 1：选择 TIM3_TRGO 为触发源
位 25	Res：保留 必须保持复位值。
位 24	EXT_TRIG_SEL：选择 GPIO 输入事件作为触发转换开始的外部事件（External trigger select） 当配置为 1 时，触发源使能。可支持选用多个不同的触发源。 ● 0：不选择 GPIO 输入事件为触发源 ● 1：选择 GPIO 输入事件为触发源
位 23:17	Res：保留 必须保持复位值。
位 16	REPLACE_EN：通道替代功能使能（Replace function enable） 此位由软件置 1 和清零，用于队列通道替代功能的使能。 ● 0：不使能通道替代功能 ● 1：使能通道替代功能
位 15:10	Res：保留 必须保持复位值。
位 9	EXT_TRIG_FILTER：当触发源为 GPIO 输入的信号时数字滤波使能（External GPIO event trigger filter enable） 这些位由软件置 1 和清零。 ● 0：不开启数字滤波功能 ● 1：开启数字滤波功能
位 8:6	TRIG_DLY_CNT[2:0]：配置硬件触发信号的触发延时（Trigger delay configure）

	这些位由软件置 1 和清零，使用 CLK_TRIG 时钟计数，可配置 0 ~7 个 CLK_TRIG 周期。 ● 000：触发延时 0 CLK_TRIG 周期 ● 001：触发延时 1 CLK_TRIG 周期 ● 010：触发延时 2 CLK_TRIG 周期 ● 011：触发延时 3 CLK_TRIG 周期 ● 100：触发延时 4 CLK_TRIG 周期 ● 101：触发延时 5 CLK_TRIG 周期 ● 110：触发延时 6 CLK_TRIG 周期 ● 111：触发延时 7 CLK_TRIG 周期
位 5:2	Res：保留 必须保持复位值。
位 1	STEP：配置队列工作在间断模式（Step mode configure） 这些位由软件置 1 和清零，用于配置队列工作在间断模式，一次触发只转换队列里的一个元素，同时队列指针加 1。） ● 0：不使能间断模式 ● 1：使能间断模式 *注意：CONT 模式和 STEP 模式功能冲突，不能同时使能。*
位 0	CONT：配置队列工作在循环模式（Continual mode configure） 这些位由软件置 1 和清零，用于配置队列工作在循环模式，只要触发队列，队列转换完成会重新转换队列元素，如此往复。建议配置该功能时，队列配置为最低优先级。 ● 0：不使能循环模式 ● 1：使能循环模式 当 CONT=0，STEP=0 时，队列工作于单次转换模式。 *注意：CONT 模式和 STEP 模式功能冲突，不能同时使能。*

10.16.6 常规队列 x 元素配置寄存器（ADC_SH_SQUEx_ELEMENT）（x=0..3）

偏移地址：0x2C + x*4

复位值：0x00000000

队列 x 元素配置寄存器一个可以配置 8 个元素（元素 7-0），每个元素可以配置为 0~13，指明该元素所选择的采保模式的 ADC 模拟通道输入。

注意：仅单/双路采保模式下的常规序列需要配置本寄存器，其他模式无需配置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_8_CHL[3:0]				NUM_7_CHL[3:0]				NUM_6_CHL[3:0]				NUM_5_CHL[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_4_CHL[3:0]				NUM_3_CHL[3:0]				NUM_2_CHL[3:0]				NUM_1_CHL[3:0]			
rw				rw				rw				rw			

位 31:28	NUM_8_CHL[3:0]：单路采保模式下的队列元素 E7 对应的 AD 转换通道号，双路采保模式下的队列元素 E3 对应的第二路采保的 AD 转换通道号（The eighth conversion channel of the queue sequence） ● 0000：单路采保第八次转换通道是通道 0，双路采保第四次转换通道中的第二路采保通道是通道 0。 ● 0001：单路采保第八次转换通道是通道 1，双路采保第四次转换通道中的第二路采保通道是通道 1。

	● 0010：单路采保第八次转换通道是通道 2，双路采保第四次转换通道中的第二路采保通道是通道 2。 ● 0011：单路采保第八次转换通道是通道 3，双路采保第四次转换通道中的第二路采保通道是通道 3。 ● … ● 1101：单路采保第八次转换通道是通道 13，双路采保第四次转换通道中的第二路采保通道是通道 13。 ● 其他值：保留
位 27:24	NUM_7_CHL[3:0]：单路采保模式下的队列元素 E6 对应的 AD 转换通道号，双路采保模式下的队列元素 E3 对应的第一路采保对应的 AD 转换通道号（The seventh conversion channel of the queue sequence） ● 0000：单路采保第七次转换通道是通道 0，双路采保第四次转换通道中的第一路采保通道是通道 0。 ● 0001：单路采保第七次转换通道是通道 1，双路采保第四次转换通道中的第一路采保通道是通道 1。 ● 0010：单路采保第七次转换通道是通道 2，双路采保第四次转换通道中的第一路采保通道是通道 2。 ● 0011：单路采保第七次转换通道是通道 3，双路采保第四次转换通道中的第一路采保通道是通道 3。 ● … ● 1101：单路采保第七次转换通道是通道 13，双路采保第四次转换通道中的第一路采保通道是通道 13。 ● 其他值：保留
位 23:20	NUM_6_CHL[3:0]：单路采保模式下的队列元素 E5 对应的 AD 转换通道号，双路采保模式下的队列元素 E2 对应的第二路采保对应的 AD 转换通道号（The sixth conversion channel of the queue sequence） ● 0000：单路采保第六次转换通道是通道 0，双路采保第三次转换通道中的第二路采保通道是通道 0。 ● 0001：单路采保第六次转换通道是通道 1，双路采保第三次转换通道中的第二路采保通道是通道 1。 ● 0010：单路采保第六次转换通道是通道 2，双路采保第三次转换通道中的第二路采保通道是通道 2。 ● 0011：单路采保第六次转换通道是通道 3，双路采保第三次转换通道中的第二路采保通道是通道 3。 ● … ● 1101：单路采保第六次转换通道是通道 13，双路采保第三次转换通道中的第二路采保通道是通道 13。 ● 其他值：保留
位 19:16	NUM_5_CHL[3:0]：单路采保模式下的队列元素 E4 对应的 AD 转换通道号，双路采保模式下的队列元素 E2 对应的第一路采保对应的 AD 转换通道号（The fifth conversion channel of the queue sequence） ● 0000：单路采保第五次转换通道是通道 0，双路采保第三次转换通道中的第一路采保通道是通道 0。 ● 0001：单路采保第五次转换通道是通道 1，双路采保第三次转换通道中的第一路采保通道是通道 1。 ● 0010：单路采保第五次转换通道是通道 2，双路采保第三次转换通道中的第一路采保通道是通道 2。 ● 0011：单路采保第五次转换通道是通道 3，双路采保第三次转换通道中的第一路采保通道是通道 3。

	● … ● 1111：单路采保第五次转换通道是通道 13，双路采保第三次转换通道中的第一路采保通道是通道 13。 ● 其他值：保留
位 15:12	NUM_4_CHL[3:0]：单路采保模式下的队列元素 E3 对应的 AD 转换通道号，双路采保模式下的队列元素 E1 对应的第二路采保对应的 AD 转换通道号（The fourth conversion channel of the queue sequence） ● 0000：单路采保第四次转换通道是通道 0，双路采保第二次转换通道中的第二路采保通道是通道 0。 ● 0001：单路采保第四次转换通道是通道 1，双路采保第二次转换通道中的第二路采保通道是通道 1。 ● 0010：单路采保第四次转换通道是通道 2，双路采保第二次转换通道中的第二路采保通道是通道 2。 ● 0011：单路采保第四次转换通道是通道 3，双路采保第二次转换通道中的第二路采保通道是通道 3。 ● … ● 1101：单路采保第四次转换通道是通道 13，双路采保第二次转换通道中的第二路采保通道是通道 13。 ● 其他值：保留
位 11:8	NUM_3_CHL[3:0]：单路采保模式下的队列元素 E2 对应的 AD 转换通道号，双路采保模式下的队列元素 E1 对应的第一路采保对应的 AD 转换通道号（The third conversion channel of the queue sequence） ● 0000：单路采保第三次转换通道是通道 0，双路采保第二次转换通道中的第一路采保通道是通道 0。 ● 0001：单路采保第三次转换通道是通道 1，双路采保第二次转换通道中的第一路采保通道是通道 1。 ● 0010：单路采保第三次转换通道是通道 2，双路采保第二次转换通道中的第一路采保通道是通道 2。 ● 0011：单路采保第三次转换通道是通道 3，双路采保第二次转换通道中的第一路采保通道是通道 3。 ● … ● 1101：单路采保第三次转换通道是通道 13，双路采保第二次转换通道中的第一路采保通道是通道 13。
位 7:4	NUM_2_CHL[3:0]：单路采保模式下的队列元素 E1 对应的 AD 转换通道号，双路采保模式下的队列元素 E0 对应的第二路采保对应的 AD 转换通道号（The second conversion channel of the queue sequence） ● 0000：单路采保第二次转换通道是通道 0，双路采保第一次转换通道中的第二路采保通道是通道 0。 ● 0001：单路采保第二次转换通道是通道 1，双路采保第一次转换通道中的第二路采保通道是通道 1。 ● 0010：单路采保第二次转换通道是通道 2，双路采保第一次转换通道中的第二路采保通道是通道 2。 ● 0011：单路采保第二次转换通道是通道 3，双路采保第一次转换通道中的第二路采保通道是通道 3。 ● … ● 1101：单路采保第二次转换通道是通道 13，双路采保第一次转换通道中的第二路采保通道是通道 13。
位 3:0	NUM_1_CHL[3:0]：单路采保模式下的队列元素 E0 对应的 AD 转换通道号，双路采保模式下的队列元素 E0 对应的第一路采保对应的 AD 转换通道号（The first conversion channel of the queue sequence）

	• 0000：单路采保第一次转换通道是通道 0，双路采保第一次转换通道中的第一路采保通道是通道 0。 • 0001：单路采保第一次转换通道是通道 1，双路采保第一次转换通道中的第一路采保通道是通道 1。 • 0010：单路采保第一次转换通道是通道 2，双路采保第一次转换通道中的第一路采保通道是通道 2。 • 0011：单路采保第一次转换通道是通道 3，双路采保第一次转换通道中的第一路采保通道是通道 3。 • … • 1101：单路采保第一次转换通道是通道 13，双路采保第一次转换通道中的第一路采保通道是通道 13。 举例说明： • 在单路采保模式模式下： 需要使能队列元素 E0 对应使能元素 NUM_1_CHL，队列元素 E1 对应使能元素 NUM_2_CHL，以此类推队列元素 E7 对应使能 NUM_8_CHL。最多可以使能 E0~E7 共 8 个元素，共输出 8 个结果。 如果序列配置为单路采保扫描模式，则不需要配置 ADC_SQUEx_ELEMENT 寄存器。根据队列元素 E0~队列元素 E13 的使能，按配置，没使能的元素通道跳过。队列顺序从 CHL0~CHL13，或者从 CHL13~CHL0。最多输出 14 个结果。 • 在双路采保模式下： 双路采保模式下 NUM_1_CHL 与 NUM_2_CHL：一对双路采保模式下的两个 AD 转换的通道号，第一个采保采集 NUM_1_CHL，第二个采保采集 NUM_2_CHL。 举例说明：使能 E0，对应元素 NUM_1_CHL 和 NUM_2_CHL 均使能，以此类推 E3 对应使能 NUM_6_CHL 和 NUM_7_CHL。即当 Ex 使能，则 NUM_（2x+1）_CHL 和 NUM_（2x+2）_CHL 使能。队列顺序 1 到 8，采保 1 到采保 2。最多输出 8 个结果。

10.16.7 ADC 通道 x 结果寄存器（ADC_CHLx_RESULT）（x=0..15）

偏移地址：0x3C+ x*4

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVER_OPT	Res		REFRESH_IND	COMP_UALM	COMP_DALM	OVER_ALM	Res								
rw			r	rc_w0	rc_w0	rc_w0									

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

位 31	OVER_OPT：过载发生时的策略配置（Over run option configure） 此位由软件置 1 和清零，用于配置发生过载时的数据寄存器更新策略。 • 0：新数据覆盖没被读走的 AD_RESULT。 • 1：保持没被读走的 AD_RESULT，丢掉新的数据。
位 30:29	Res：保留 必须保持复位值。
位 28	REFRESH_IND：AD 转换结果刷新标志（Refresh indication） 此位当 AD 值有新的数据更新时由硬件置 1，数据被读走后硬件自动清零。 • 0：没有新的数据刷新或者数据已被读走后 • 1：表示有新的数据刷新。

位 27	COMP_UALM：比较功能上溢告警，需软件清零（Comp up alarm flag） 此位当 ADC 转换数据大于上溢阈值，由硬件置 1，软件清零，清零后清除对应中断。 ● 0：数据没有发生上溢 ● 1：数据发生上溢
位 26	COMP_DALM：比较功能下溢告警，需软件清零（Comp down alarm flag） 此位当 ADC 转换数据小于下溢阈值，由硬件置 1，软件清零，清零后清除对应中断。 ● 0：数据没有发生下溢 ● 1：数据发生下溢
位 25	OVER_ALM：过载告警标志，需软件清零（Over run alarm flag） 当 ADC 有新的数据更新，但 AD_RESULT 上一次的转换数据仍未被读走时由硬件置 1，由软件清零，清零后清除对应中断。 ● 0：数据没有发生过载 ● 1：数据发生过载
位 24:16	Res：保留 必须保持复位值。
位 15:12	CHANNEL_NUM[3:0]：指示数据对应的真实通道号（Channel num indication） 这些位是只读的，用于指示每次转换数据的真实通道号。如通道替代功能开启，则为 CHANNEL_REPLACE 的值。
位 11:0	AD_RESULT[11:0]：每次 AD 转换后的结果输出（ADC result output） 这些位是只读的，是每次 AD 转换的结果，结果为正数。

10.16.8 ADC 公用结果寄存器（ADC_COMM_RESULT）

偏移地址：0x7C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVER_OPT	Res		REFRESH_IND	Res		OVER_ALM	Res								TEST_IND
rw			r			rc_w0									r

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

位 31	OVER_OPT：过载发生时的策略配置（Over run option configure） 此位由软件置 1 和清零，用于配置发生过载时的数据寄存器更新策略。 ● 0：新数据覆盖没被读走的 AD_RESULT。 ● 1：保持没被读走的 AD_RESULT，丢掉新的数据。
位 30:29	Res：保留 必须保持复位值。
位 28	REFRESH_IND：AD 转换结果刷新标志（Refresh indication） 此位当 AD 值有新的数据更新时由硬件置 1，数据被读走后硬件自动清零。 ● 0：没有新的数据刷新或者数据被读走后，逻辑置位为 0。 ● 1：表示有新的数据刷新。

位 27:26	Res：保留 必须保持复位值。
位 25	OVER_ALM：过载告警标志，需软件清零（Over run alarm flag） 此位当 ADC 有新的数据更新，但 AD_RESULT 上一次的转换数据仍未被读走由硬件置 1，由软件清零，清零后清除对应中断。 ● 0：数据没有发生过载 ● 1：数据发生过载
位 24:17	Res：保留 必须保持复位值。
位 16	TEST_IND：指示数据为测试队列通道的数据（Test mode data indication） 此位是只读的，用于指示数据为测试队列通道的数据。
位 15:12	CHANNEL_NUM[3:0]：指示数据对应的真实通道号（Channel num indication） 这些位是只读的，用于指示每次转换数据所属的真实通道号。 *注意：在测试通道的转换中，本位域不起作用。*
位 11:0	AD_RESULT[11:0]：每次 AD 转换后的结果输出（ADC result output） 这些位是只读的，是每次 AD 转换的结果，结果为正数。所有通道的 AD 转换数据均会在此输出。

10.16.9 ADC 延时配置寄存器（ADC_CYCLE_DLY_CFG）

偏移地址：0x80

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	SAMPLE_CYCLE_TEST[2:0]			SAMPLE_CYCLE_3[2:0]			SAMPLE_CYCLE_2[2:0]			SAMPLE_CYCLE_1[2:0]			SAMPLE_CYCLE_0[2:0]		
	rw			rw			rw			rw			rw		

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

位 31	Res：保留 必须保持复位值。
位 30:28	SAMPLE_CYCLE_TEST[2:0]：测试队列采样时钟周期配置（Test queue sample cycle configure） ● 000：1.5 个 ADC 时钟周期 ● 001：7.5 个 ADC 时钟周期 ● 010：13.5 个 ADC 时钟周期 ● 011：28.5 个 ADC 时钟周期 ● 100：41.5 个 ADC 时钟周期 ● 101：55.5 个 ADC 时钟周期 ● 110：71.5 个 ADC 时钟周期 ● 111：239.5 个 ADC 时钟周期 此位域配置测试队列的 ADC 采样时间，作用于队列内所有 ADC 通道。当采样保持电路关闭时（BK 模式时），此时间有效。

位 27:25	SAMPLE_CYCLE_3[2:0]：常规队列 3 采样时钟周期配置（Queue3 sample cycle configure） ● 000：1.5 个 ADC 时钟周期 ● 001：7.5 个 ADC 时钟周期 ● 010：13.5 个 ADC 时钟周期 ● 011：28.5 个 ADC 时钟周期 ● 100：41.5 个 ADC 时钟周期 ● 101：55.5 个 ADC 时钟周期 ● 110：71.5 个 ADC 时钟周期 ● 111：239.5 个 ADC 时钟周期 此位域配置常规队列 3 ADC 的采样时间，作用于队列内所有 ADC 通道。当采样保持电路关闭时，此时间有效。
位 24:22	SAMPLE_CYCLE_2[2:0]：常规队列 2 采样时钟周期配置（Queue2 sample cycle configure） ● 000：1.5 个 ADC 时钟周期 ● 001：7.5 个 ADC 时钟周期 ● 010：13.5 个 ADC 时钟周期 ● 011：28.5 个 ADC 时钟周期 ● 100：41.5 个 ADC 时钟周期 ● 101：55.5 个 ADC 时钟周期 ● 110：71.5 个 ADC 时钟周期 ● 111：239.5 个 ADC 时钟周期 此位域配置常规队列 2 ADC 的采样时间，作用于队列内所有 ADC 通道。当采样保持电路关闭时，此时间有效。
位 21:19	SAMPLE_CYCLE_1[2:0]：常规队列 1 采样时钟周期配置（Queue1 sample cycle configure） ● 000：1.5 个 ADC 时钟周期 ● 001：7.5 个 ADC 时钟周期 ● 010：13.5 个 ADC 时钟周期 ● 011：28.5 个 ADC 时钟周期 ● 100：41.5 个 ADC 时钟周期 ● 101：55.5 个 ADC 时钟周期 ● 110：71.5 个 ADC 时钟周期 ● 111：239.5 个 ADC 时钟周期 此位域配置常规队列 1 ADC 的采样时间，作用于队列内所有 ADC 通道。当采样保持电路关闭时，此时间有效。
位 18:16	SAMPLE_CYCLE_0[2:0]：常规队列 0 采样时钟周期配置（Queue0 sample cycle configure） ● 000：1.5 个 ADC 时钟周期 ● 001：7.5 个 ADC 时钟周期 ● 010：13.5 个 ADC 时钟周期 ● 011：28.5 个 ADC 时钟周期 ● 100：41.5 个 ADC 时钟周期 ● 101：55.5 个 ADC 时钟周期 ● 110：71.5 个 ADC 时钟周期 ● 111：239.5 个 ADC 时钟周期 此位域配置常规队列 0 ADC 的采样时间，作用于队列内所有 ADC 通道。当采样保持电路关闭时，此时间有效。

位 15:9	Res：保留 必须保持复位值。
位 8:7	SH_SAMPLE_HOLD[1:0]：采保模块采样完成后的数据保持时间（SH sample hold cycle configure） 保持时间至少>80ns，ADC 在保持时间之后采样。 ● 00：1 个 CLK_ADC 周期 ● 01：5 个 CLK_ADC 周期 ● 10：9 个 CLK_ADC 周期 ● 11：13 个 CLK_ADC 周期
位 6:5	EOC_SOC_DIS[1:0]：配置 ADC 的 EOC 和 SOC 之间的间隔（Interval between EOC and SOC configure） 该位域通过软件写 1 或置 0，用于配置 ADC 的 EOC 和 SOC 之间的间隔。当无时间间隔时，则采样时间为最小的 1.5 cycle。 ● 00：1.5 个 CLK_ADC 周期 ● 01：4.5 个 CLK_ADC 周期 ● 10：10.5 个 CLK_ADC 周期 ● 11：14.5 个 CLK_ADC 周期
位 4:3	CHL_HOLD[1:0]：配置通道改变到采样操作的时间间隔（Channel hold configure） 这些位通过软件写 1 或清 0，用于配置 CHANNEL 改变到采样操作的时间间隔。 ● 00：无时间间隔 ● 01：0.5 个 CLK_ADC 周期 ● 10：1 个 CLK_ADC 周期 ● 11：1.5 个 CLK_ADC 周期 *注意：BK 模式中，无需配置此位。*
位 2:0	SH_SAMPLE_CYCLE[2:0]：配置采样保持电路的采样时间（SH sample cycle configure） 这些位通过软件写 1 或清 0，用于配置采样保持电路的采样时间。采样时间 = SAMPLE_CYCLE + 1（CHL_HOLD 为 00 或者 10） 或者 SAMPLE_CYCLE（CHL_HOLD 为 01 或者 11）个 CLK_ADC 周期（最少 1 个）（只在使用采样保持功能时生效） ● 000：1 个 CLK_ADC 周期 ● 001：2 个 CLK_ADC 周期 ● 010：3 个 CLK_ADC 周期 ● 011：4 个 CLK_ADC 周期 ● 100：5 个 CLK_ADC 周期 ● 101：6 个 CLK_ADC 周期 ● 110：7 个 CLK_ADC 周期 ● 111：8 个 CLK_ADC 周期 *注意：* • *当 BIAS_CT 为 0 时，采样保持电路的采样时间至少为 360ns；* • *当 BIAS_CT 为 1 时，采样保持电路的采样时间至少为 280ns。*

10.16.10 ADC 比较功能全局配置寄存器（ADC_COMP_GLBCFG）

偏移地址：0x90

复位值：0x07FF0000

注意：仅通道 0-15 支持数据窗口比较功能。

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

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

位 31:29	Res：保留 必须保持复位值。
位 28	COMP_UP_EN：上溢比较功能使能位（COMP up enable） 此位由软件置 1 和清零，用于使能上溢比较功能。 ● 0：不使能上溢比较功能 ● 1：使能上溢比较功能
位 27:16	COMP_UP_VALUE[11:0]：配置对应通道的上溢门限值（COMP up threshold value configure） 这些位由软件写入，用于配置对应通道的上溢门限值。
位 15:13	Res：保留 必须保持复位值。
位 12	COMP_DOWN_EN：下溢比较功能使能位（COMP down enable） 此位由软件置 1 和清零，用于使能下溢比较功能。 ● 0：不使能下溢比较功能 ● 1：使能下溢比较功能
位 11:0	COMP_DOWN_VALUE[11:0]：配置对应通道的下溢门限值（COMP down threshold value configure） 这些位由软件写入，用于配置对应通道的下溢门限值。 *注意：配置需要满足 COMP_UP_VALUE> COMP_DOWN_VALUE 的要求。*

10.16.11 常规队列平均功能寄存器（ADC_SQUEx_AVERAGE）（x=0..3）

偏移地址：0xD0+ x* 4

复位值：0x0000 0000

注意：仅常规队列支持平均功能时，需要配置本寄存器。测试队列不支持此功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVER_OPT	Res		REFRESH_IND	AVE_EN	Res	OVER_ALM	Res	D7	D6	D5	D4	D3	D2	D1	D0
rw			r	rw		rc_w0		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				AVERAGE_VALUE[11:0]											
				r											

位 31	OVER_OPT：过载发生时的策略配置（Over run option configure） 此位由软件置 1 和清零，用于配置发生过载时的数据寄存器更新策略。 ● 0：仍然刷新没被读走的 AVERAGE_VALUE ● 1：保持没被读走的 AVERAGE_VALUE，丢掉新的数据
位 30:29	Res：保留 必须保持复位值。

位 28	REFRESH_IND：平均值结果刷新标志（Refresh indication） 此位当平均值有新的数据更新时由硬件置 1，数据被读走后硬件自动清零。 ● 0：没有新的数据刷新或者数据已被读走后。 ● 1：表示有新的数据刷新
位 27	AVE_EN：求平均功能使能位（Average enable） 此位由软件置 1 和清零，用于使能求平均功能。 ● 0：不使能求平均功能 ● 1：使能求平均功能
位 26	Res：保留 必须保持复位值。
位 25	OVER_ALM：过载告警标志，需软件清零（Over run alarm flag） 此位当平均值有新的数据更新，但 AVERAGE_VALUE[的数据还未被及时读走由硬件置 1，需软件清零。 ● 0：数据没有发生过载 ● 1：数据发生过载
位 24	Res：保留 必须保持复位值。
位 16+y （y=7..0）	Dy：参与平均计算的通道元素使能位（Average element enable） Dy 由软件置 1 和清零，用于使能参与求平均计算的通道元素，D0~D7 与 ADC_SH_SQUEx_ELEMENT 寄存器中的元素 E0-E7 相对应。参与平均计算的通道元素的个数必须为 2 个、4 个或 8 个。 ● 0：不使能 Dy（对应 ADC_SH_SQUEx_ELEMENT 寄存器中的元素 Ey）参与平均计算 ● 1：使能 Dy（对应 ADC_SH_SQUEx_ELEMENT 寄存器中的元素 Ey）参与平均计算
位 15:12	Res：保留 必须保持复位值。
位 11:0	AVERAGE_VALUE[11:0]：平均后的结果输出（Average output） 这些位是只读的，是队列所使能的元素通道转换结果最终的平均值。

10.16.12 测试队列配置寄存器（ADC_TEST_CHL_CFG）

偏移地址：0xE0

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEST_EN	EXT_TRIG_FILTER	TRIG_DLY_CNT[2:0]			TRIG_CFG[1:0]		Res	TIM1_TRGO_SEL	TIM1_CC4_SEL	TIM1_CC5_SEL	TIM1_CC6_SEL	TIM2_TRGO_SEL	TIM3_TRGO_SEL	Res	EXT_TRIG_SEL
rw	rw	rw			rw			rw	rw	rw	rw	rw	rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW_TRIG	CONT	DONE	MODE	QUE_STATE[1:0]		Res			PRIORITY[1:0]		CHL_NUM[4:0]				
rs	rw	rc_w0	rw	rw					rw		rw				

位 31	TEST_EN：测试模式使能位（Test mode enable） 此位由软件置 1 和清零，用于测试模式使能。 ● 0：不使能测试模式 ● 1：使能测试模式 *注意：测试队列仅支持 BK 和单路采保模式。SW_TRIG 触发转换后，会根据其配置优先级，和常规队列 que0~que3 一起做优先级判断。* 该模式公用 COMM 的 DMA 通道，可以使用 DMA 搬运。
位 30	EXT_TRIG_FILTER：当触发源为 GPIO 输入的信号数字滤波使能（External trigger filter enable） 这些位由软件置 1 和清零。 ● 0：不开启数字滤波功能 ● 1：开启数字滤波功能
位 29:27	TRIG_DLY_CNT[2:0]：配置硬件触发信号的触发延时（Trigger delay configure） 这些位由软件置 1 和清零，使用 CLK_TRIG 时钟计数，可配置 0 ~7 个 CLK_TRIG 周期。 ● 000：触发延时 0 CLK_TRIG 周期 ● 001：触发延时 1 CLK_TRIG 周期 ● 010：触发延时 2 CLK_TRIG 周期 ● 011：触发延时 3 CLK_TRIG 周期 ● 100：触发延时 4 CLK_TRIG 周期 ● 101：触发延时 5 CLK_TRIG 周期 ● 110：触发延时 6 CLK_TRIG 周期 ● 111：触发延时 7 CLK_TRIG 周期
位 26:25	TRIG_CFG[1:0]：配置外部触发信号的使能和极性选择（Trigger enable and polarity select） 这些位由软件置 1 和清零，用于选择硬件触发源的极性及使能外部触发。 ● 00：禁止外部触发事件 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 24	Res：保留 必须保持复位值。
位 23	TIM1_TRGO_SEL：选择用于触发转换开始的外部事件，触发源为 TIM1_TRGO 输入的信号（TIM1 TRGO select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_TRGO 为触发源 ● 1：选择 TIM1_TRGO 为触发源
位 22	TIM1_CC4_SEL：选择用于触发转换开始的外部事件，触发源为 TIM1_CC4 输入的信号（TIM1 CC4 select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC4 为触发源 ● 1：选择 TIM1_CC4 为触发源
位 21	TIM1_CC5_SEL：选择用于触发转换开始的外部事件，触发源为 TIM1_CC5 输入的信号（TIM1 CC5

	select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC5 为触发源 ● 1：选择 TIM1_CC5 为触发源
位 20	TIM1_CC6_SEL：选择用于触发转换开始的外部事件，触发源为 TIM1_CC6 输入的信号（TIM1 CC6 select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM1_CC6 为触发源 ● 1：选择 TIM1_CC6 为触发源
位 19	TIM2_TRGO_SEL：选择用于触发转换开始的外部事件，触发源为 TIM2_TRGO 输入的信号（TIM2 TRGO select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM2_TRGO 为触发源 ● 1：选择 TIM2_TRGO 为触发源
位 18	TIM3_TRGO_SEL：选择用于触发转换开始的外部事件，触发源为 TIM3_TRGO 输入的信号（TIM3 TRGO select） 当配置为 1 时，触发源使能。支持选用多个不同的触发源。 ● 0：不选择 TIM3_TRGO 为触发源 ● 1：选择 TIM3_TRGO 为触发源
位 17	Res：保留 必须保持复位值。
位 16	EXT_TRIG_SEL：选择 GPIO 输入事件作为触发转换开始的外部事件（External trigger select） 当配置为 1 时，GPIO 触发源使能。支持选用多个不同的触发源。 ● 0：不选择 GPIO 输入事件为触发源 ● 1：选择 GPIO 输入事件为触发源
位 15	SW_TRIG：软件触发位（Software trigger） 此位由软件置 1，在空闲状态，软件触发队列工作。在队列回到空闲状态逻辑自动归 0。软件不能置 0。 ● 0：不启动软件触发 ● 1：启动软件触发
位 14	CONT：配置队列工作在循环模式（Continual mode configure） 这些位由软件置 1 和清零，用于配置队列工作在循环模式，只要触发队列，队列转换完成会重新转换队列元素，如此往复。建议配置该功能时，队列配置为最低优先级。 ● 0：不使能循环模式（队列工作于单次转换模式） ● 1：使能循环模式
位 13	DONE：ADC 转换完成标志（ADC convert done flag） ADC 转换完成标志，数据放在公用结果寄存器（ADC_COMM_RESULT）内，同时产生对应中断。此位写 0 清零，清零后清除中断请求。 ● 0：ADC 转换未完成状态（或者 ADC 转换完成被清除状态） ● 1：ADC 转换完成标志

位 12	MODE：ADC 模式选择（ADC mode select） 此位选择 ADC 的工作模式。 ● 0：单路采保 ● 1：BK 模式
位 11:10	QUE_STATE[1:0]：当前的队列工作状态（Queue state） 这些位通过硬件置位来表示当前队列的工作状态。 ● 00：空闲状态，队列空闲，可触发队列，启动 AD 转换。 ● 01：保持和状态，队列工作中，因高优先级队列抢占而跳转到保持状态。 ● 10：工作状态，队列工作中。 ● 11：未使用状态，队列不可用状态。
位 9:7	Res：保留 必须保持复位值。
位 6:5	PRIORITY[1:0]：配置队列优先级（Queue priority configure） 这些位通过软件置 1 和清零，用于配置队列的优先级，00 为最低 ，11 为最高。同优先级队列按顺序执行，高优先级队列可插入低优先级队列处理完后，低优先级队列恢复。 ● 00：最低优先级 ● … ● 11：最高优先级
位 4:0	CHL_NUM[4:0]：配置转换的通道（Channel configure） ● 在单路采保模式下： ○ 00000：外部输入通道 CHL0 为采样通道 ○ 00001：外部输入通道 CHL1 为采样通道 ○ … ○ 01001：外部输入通道 CHL9 为采样通道 ○ 01010：内部输入通道 10（OPAMP1 输出）为采样通道 ○ 01011：内部输入通道 11（OPAMP2 输出）为采样通道 ○ 01100：内部输入通道 12（OPAMP3 输出）为采样通道 ○ 01101：内部输入通道 13（8 位 DAC 输出 VOLTAGE_DIV）为采样通道 ○ 其他值：保留 ● 在 BK 模式下： ○ 00000：外部输入通道 CHL0 为采样通道 ○ 00001：外部输入通道 CHL1 为采样通道 ○ … ○ 01001：外部输入通道 CHL9 为采样通道 ○ 01010：保留 ○ 01011：内部输入通道 11（OPAMP1 输出）为采样通道 ○ 01100：保留 ○ 01101：内部输入通道 13（OPAMP2 输出）为采样通道 ○ 01110：内部输入通道 14（OPAMP3 输出）为采样通道 ○ 01111：内部输入通道 15（8 位 DAC 输出 VOLTAGE_DIV）为采样通道 ○ 10000：内部输入通道 16（内部参考电压 VREFINT）为采样通道 ○ 10001：内部输入通道 17（内部 Core 电压 LDO）为采样通道 ○ 其他值：保留

10.16.13 ADC 系统配置寄存器 2（ADC_SYSCFGR2）

偏移地址：0xE4

复位值：0x0080000

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
GAIN_EN	GAIN_FIX[5:0]						Res				BK_EN_TEST	BK_EN3	BK_EN2	BK_EN1	BK_EN0
rw	rw										rw	rw	rw	rw	rw

位 31:16	Res：保留 必须保持复位值。
位 15	GAIN_EN：增益补偿使能（Gain enable） ● 0：关闭 ● 1：开启
位 14:9	GAIN_FIX[5:0]：增益修正参数（符号数）（Gain fix parameter ）
位 8:5	Res：保留 必须保持复位值。
位 4	BK_EN_TEST：测试队列 BK 模式使能（Test queue BK mode enable） 此位通过软件置 1 和清零，用于测试。 ● 0：不使能 test 队列 BK 模式 ● 1：使能 test 队列 BK 模式
位 3	BK_EN3：队列 3 BK 模式使能（Queue3 BK mode enable） 这些位通过软件置 1 和清零，用于使能队列 3 的 BK 模式。 ● 0：不使能队列 3 的 BK 模式 ● 1：使能队列 3 的 BK 模式 *注意：仅当 ADC_SQUEx_CFG1.QUE_MODE=011/100 时，本寄存器位配置为 1 才起效。*
位 2	BK_EN2：队列 2 BK 模式使能（Queue2 BK mode enable） 这些位通过软件置 1 和清零，用于使能队列 2 的 BK 模式。 ● 0：不使能队列 2 的 BK 模式 ● 1：使能队列 2 的 BK 模式 *注意：仅当 ADC_SQUEx_CFG1.QUE_MODE=011/100 时，本寄存器位配置为 1 才起效。*
位 1	BK_EN1：队列 1 BK 模式使能（Queue1 BK mode enable） 这些位通过软件置 1 和清零，用于使能队列 1 的 BK 模式。 ● 0：不使能队列 1 的 BK 模式 ● 1：使能队列 1 的 BK 模式 *注意：仅当 ADC_SQUEx_CFG1.QUE_MODE=011/100 时，本寄存器位配置为 1 才起效。*
位 0	BK_EN0：队列 0 BK 模式使能（Queue0 bk mode enable） 这些位通过软件置 1 和清零，用于使能队列 0 的 BK 模式。 ● 0：不使能队列 0 的 BK 模式

	● 1：使能队列 0 的 BK 模式 *注意：仅当 ADC_SQUEx_CFG1.QUE_MODE=011/100 时，本寄存器位配置为 1 才起效。*

10.16.14 常规队列 x 元素替换配置寄存器（ADC_SQUEx_REPLACE）（x=0..3）

偏移地址：0xE8+ x* 4

复位值：0x00000000

注意：仅单/双路采保模式下的常规序列支持替换功能，需要配置本寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NUM_8_CHL_REPLACE[3:0]				NUM_7_CHL_REPLACE[3:0]				NUM_6_CHL_REPLACE[3:0]				NUM_5_CHL_REPLACE[3:0]			
rw				rw				rw				rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NUM_4_CHL_REPLACE[3:0]				NUM_3_CHL_REPLACE[3:0]				NUM_2_CHL_REPLACE[3:0]				NUM_1_CHL_REPLACE[3:0]			
rw				rw				rw				rw			

位 4y-1:4y-4（y=8..1）	NUM_y_CHL_REPLACE[3:0]：配置替代功能对应的通道号（Configure the replaced channel of y conversion） 该位域通过软件置 1 和清零。NUM_y_CHL_REPLACE 与 ADC_SQUEx_ELEMENT 寄存器中的 NUM_y_CHL 相对应（y=8..1），代表了其替换后的通道号。 ● 0000：替代后，对应的通道为通道 0 ● 0001：替代后，对应的通道为通道 1 ● 0010：替代后，对应的通道为通道 2 ● 0011：替代后，对应的通道为通道 3 ● … ● 1111：替代后，对应的通道为通道 15 *注意：所有在 ADC_SQUEx_ELEMENT 寄存器中的做了配置的 NUM_y_CHL 都必须在 ADC_SQUEx_REPLACE 寄存器中重新配置其替换通道。不需要被替换通道的元素其通道配置和 ADC_SH_SQUEx_ELEMENT 中保持一致即可。*

10.16.15 常规队列 x 触发配置寄存器（ADC_SQUEx_TRIG）（x=0..3）

偏移地址：0x100 + x* 4

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
TIM1_TRGO_CFG[1:0]		TIM1_CC4_CFG[1:0]		TIM1_CC5_CFG[1:0]		TIM1_CC6_CFG[1:0]		TIM2_TRGO_CFG[1:0]		TIM3_TRGO_CFG[1:0]		Res		EXT_TRGO_CFG[1:0]	
rw		rw		rw		rw		rw		rw				rw	

位 31:16	Res：保留 必须保持复位值。
位 15:14	TIM1_TRGO_CFG[1:0]：配置 TIM1_TRGO 的使能和极性选择（TIM1_TRGO trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发

	● 11：双沿触发
位 13:12	TIM1_CC4_CFG[1:0]：配置 TIM1_CC4 的使能和极性选择（TIM1_CC4 trigger enable and polarity selection） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 11:10	TIM1_CC5_CFG[1:0]：配置 TIM1_CC5 的使能和极性选择（TIM1_CC5 trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 9:8	TIM1_CC6_CFG[1:0]：配置 TIM1_CC6 的使能和极性选择（TIM1_CC6 trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 7:6	TIM2_TRGO_CFG[1:0]：配置 TIM2_TRIGO 的使能和极性选择（TIM2_TRGO trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 5:4	TIM3_TRGO_CFG[1:0]：配置 TIM3_TRIGO 的使能和极性选择（TIM3_TRGO trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。 ● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发
位 3:2	Res：保留 必须保持复位值。
位 1:0	EXT_TRGO_CFG[1:0]：配置 EXT_TRIG 的使能和极性选择（EXT_TRGO trigger enable and polarity selection ） 该位域由软件置 1 和清零，用于选择外部触发源的极性及使能外部触发。

	● 00：不使能该触发源 ● 01：上升沿触发 ● 10：下降沿触发 ● 11：双沿触发

10.16.16 CLARK 运算 ADC 通道选择控制寄存器（ADC_CHSEL_CTRL_CLARK）

偏移地址：0x110

复位值：0x00000000

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
ADC_PHAB_OFFSET_EN	Res					QUE_SEL[1:0]		CHL_PHB[3:0]				CHL_PHA[3:0]			
rw						rw		rw				rw			

位 31:16	Res：保留 必须保持复位值。
位 15	ADC_PHAB_OFFSET_EN： ADC phase AB OFFSET 操作使能（ADC phase AB OFFSET operation enable） 该位域由软件置 1 和清零，用于使能 ADC 通道结果寄存器与 PHA/B OFFSET 操作使能。 ● 0：不使能 ● 1：使能 ADC 通道结果寄存器与 PHA/B OFFSET 值运算操作
位 14:10	Res：保留 必须保持复位值。
位 9:8	QUE_SEL[1:0]： ADC 通道的队列选择（Queue selection for the ADC） 该位域由软件配置，指定 ADC 通道所在的队列（0,1,2,3），通道的结果（通道结果寄存器）将与 PHA/B OFFSET 进行操作。 ● 00：队列 0 ● 01：队列 1 ● 10：队列 2 ● 11：队列 3
位 7:4	CHL_PHB[3:0]：选择用于跟 PHB OFFSET 操作的 ADC 通道结果寄存器（The resulting register used for the and PHB operation is selected） 该位域由软件配置。 ● 0000：ADC Ch.0 结果寄存器 用于 PHB OFFSET 操作 ● 0001：ADC Ch.1 结果寄存器 用于 PHB OFFSET 操作 ● 0010：ADC Ch.2 结果寄存器 用于 PHB OFFSET 操作 ● 0011：ADC Ch.3 结果寄存器 用于 PHB OFFSET 操作 ● 0100：ADC Ch.4 结果寄存器 用于 PHB OFFSET 操作 ● 0101：ADC Ch.5 结果寄存器 用于 PHB OFFSET 操作 ● 0110：ADC Ch.6 结果寄存器 用于 PHB OFFSET 操作 ● 0111：ADC Ch.7 结果寄存器 用于 PHB OFFSET 操作 ● 1000：ADC Ch.8 结果寄存器 用于 PHB OFFSET 操作 ● 1001：ADC Ch.9 结果寄存器 用于 PHB OFFSET 操作

	● 1010：ADC Ch.10 结果寄存器 用于 PHB OFFSET 操作 ● 1011：ADC Ch.11 结果寄存器 用于 PHB OFFSET 操作 ● 1100：ADC Ch.12 结果寄存器 用于 PHB OFFSET 操作 ● 1101：ADC Ch.13 结果寄存器 用于 PHB OFFSET 操作 ● 1110：ADC Ch.14 结果寄存器 用于 PHB OFFSET 操作 ● 1111：保留
位 3:0	CHL_PHA[3:0]：选择用于跟 PHA OFFSET 操作的 ADC 通道结果寄存器（The resulting register used for the and PHA operation is selected） 该位域由软件置。 ● 0000：ADC Ch.0 结果寄存器 用于 PHA OFFSET 操作 ● 0001：ADC Ch.1 结果寄存器 用于 PHA OFFSET 操作 ● 0010：ADC Ch.2 结果寄存器 用于 PHA OFFSET 操作 ● 0011：ADC Ch.3 结果寄存器 用于 PHA OFFSET 操作 ● 0100：ADC Ch.4 结果寄存器 用于 PHA OFFSET 操作 ● 0101：ADC Ch.5 结果寄存器 用于 PHA OFFSET 操作 ● 0110：ADC Ch.6 结果寄存器 用于 PHA OFFSET 操作 ● 0111：ADC Ch.7 结果寄存器 用于 PHA OFFSET 操作 ● 1000：ADC Ch.8 结果寄存器 用于 PHA OFFSET 操作 ● 1001：ADC Ch.9 结果寄存器 用于 PHA OFFSET 操作 ● 1010：ADC Ch.10 结果寄存器 用于 PHA OFFSET 操作 ● 1011：ADC Ch.11 结果寄存器 用于 PHA OFFSET 操作 ● 1100：ADC Ch.12 结果寄存器 用于 PHA OFFSET 操作 ● 1101：ADC Ch.13 结果寄存器 用于 PHA OFFSET 操作 ● 1110：ADC Ch.14 结果寄存器 用于 PHA OFFSET 操作 ● 1111：保留

注意：CHL_PHA 和 PHL_PHB 做配置的通道必须为 ADC 队列 X 配置寄存器中使能的位（BK 模式），或者 ADC 队列 x 采保模式元素寄存器中配置的通道，同时 ADC 队列 X 配置寄存器中对应位使能。

10.16.17 PHAB OFFSET 寄存器（ADC_PHAB_OFFSET）

偏移地址：0x114

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PHA_OFFSET[15:0]															
rw															

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

位 31:16	PHA_OFFSET[15:0]：PHA_OFFSET 值（Value of the PHA OFFSET） 由软件配置。
位 15:0	PHB_OFFSET[15:0]：PHB_OFFSET 值（Value of the PHB OFFSET） 由软件配置。

10.16.18 PHAB OFFSET 结果寄存器（ADC_PHAB_OFFSET_R）

偏移地址：0x118

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PHA_OFFSET_R[15:0]															
r															

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

位 31:16	PHA_OFFSET_R[15:0]：PHA_OFFSE 与 CHL_PHA 选中 ADC 通道结果寄存器差值（PHA OFFSET and CHL PHA selected ADC channel result register difference）
位 15:0	PHB_OFFSET_R[15:0]：PHB_OFFSE 与 CHL_PHB 选中 ADC 通道结果寄存器差值（PHB OFFSET and CHL PHA selected ADC channel result register difference）

11 电压比较器（COMP）

本系列芯片内置四个低功耗比较器 COMP1、COMP2、COMP3 和 COMP4。这四个比较器可分别用作独立器件，也可与定时器结合使用。

这四个比较器可用于多种功能，包括：

- 在模拟信号的触发下从低功耗模式唤醒 MCU。
- 调理模拟信号。
- 与定时器的 PWM 输出结合使用时，构成逐周期电流控制环路。

11.1 COMP 主要功能

- 比较器功耗模式可配置
- 比较器拥有施密特迟滞功能（迟滞电压有三挡：0mV、10mV、20mV）
- 比较器输出极性可配置
- 每个比较器都具有正输入和可配置的负输入，用于灵活选择电压：
  - I/O 引脚
  - 通过调节器（缓冲分压器）提供的内部参考电压
- 输出可以重定向到用于触发以下事件的 I/O 或定时器输入：
  - 捕捉事件
  - GPIO
  - 定时器/低功耗定时器
- 从睡眠模式和停机模式唤醒（通过 EXTI 控制器）

11.2 COMP 功能说明

11.2.1 COMP 框图

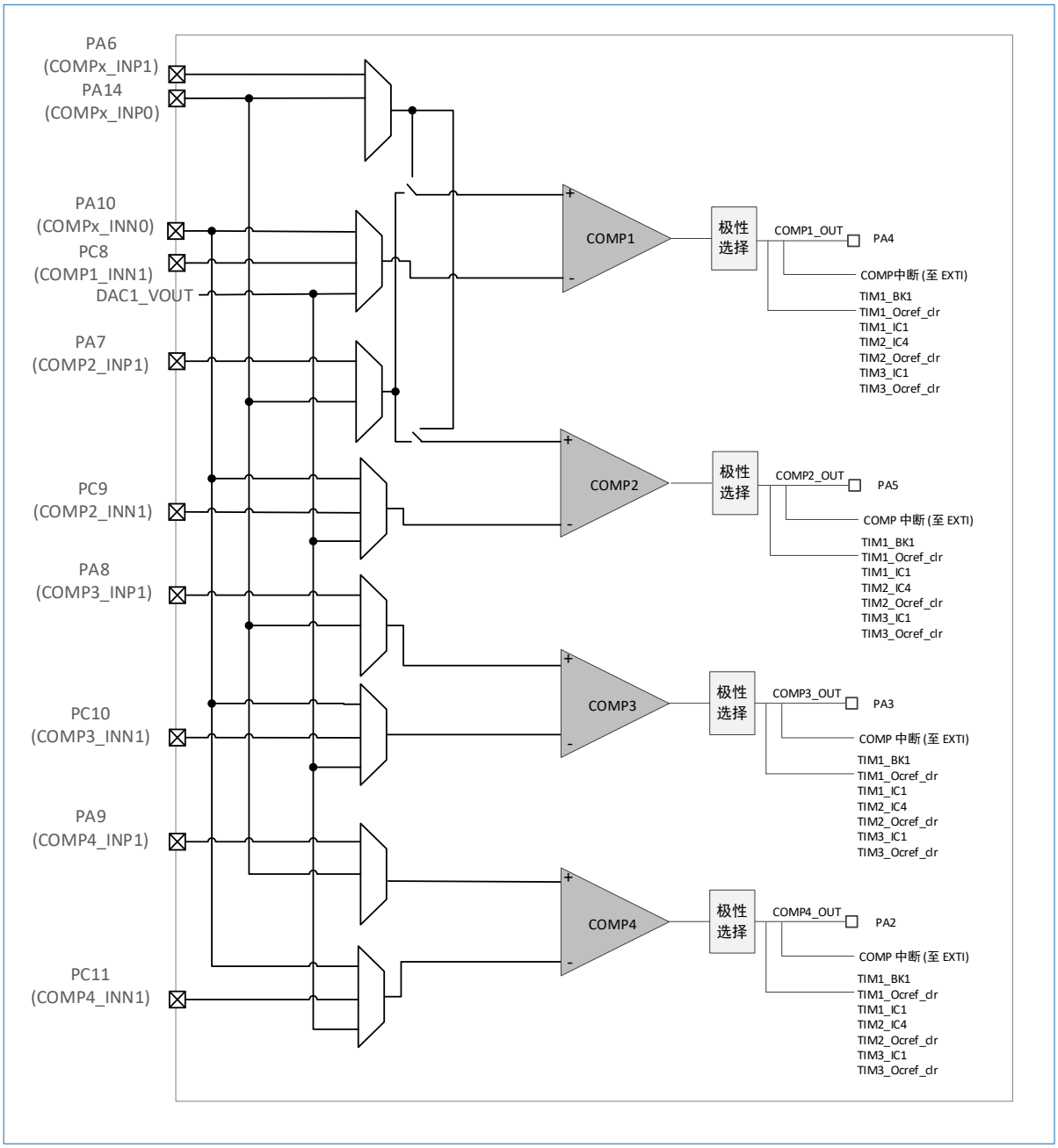

图 11-1 COMP 框图

11.2.2 COMP 引脚和内部信号

- 用作比较器输入的 I/O 必须在 GPIO 寄存器中配置为模拟模式。
- 比较器输出可以使用数据手册的“复用功能映射”表中给出的复用功能通道连接到 I/O。
- 输出也可以在内部重定向到用于以下用途的各种定时器输入：
  - 用于时序测量的输入捕捉。
- 可以在内部和外部同时对比较器输出进行重定向。

11.2.3 COMP 复位和时钟

时钟控制器提供的 COMP 时钟与 PCLK（APB 时钟）同步。

RCC 控制器中不单独提供其时钟使能控制位。COMP 和 SYSCFG 共用复位和时钟使能位。

11.2.4 COMP 锁定机制

这四个比较器可用于过流或热保护等安全用途。对于具有特定功能安全要求的应用，在发生意外的寄存器访问或程序计数器损坏时，必须保证不能更改比较器编程。

为此，可以对比较器控制和状态寄存器进行写保护（只读）。

一旦编程完成，COMPxLOCK 位便会设置为 1。这将导致整个 COMPx_CSR 寄存器变成只读，包括 COMPxLOCK 位。

只能通过 MCU 复位来复位写保护。

11.2.5 COMP 功耗模式

比较器有 2 种功耗模式供用户选择，以匹配不同的应用场景。通过 COMPx_CSR 寄存器的 COMPxMODE 位可配置如下功耗模式：

- 高速/高功耗
- 低速/低功耗

11.3 COMP 中断

比较器输出从内部连接到扩展中断和事件控制器。每个比较器都有其各自的 EXTI 线，能够产生中断或事件。该机制还可用于退出低功耗模式。

详细请参考中断和事件部分。

11.4 COMP 寄存器

基地址：0x4001 4800

空间大小：0x400

11.4.1 COMP1 和 COMP2 控制和状态寄存器（COMP_CSR1）

偏移地址：0x11C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP2LOCK	COMP2OUT	COMP2HYST[1:0]		COMP2POL	COMP2PINSEL[2:0]			WNDWEN2	COMP2NINSEL[2:0]			COMP2MODE[1:0]		Res	COMP2EN
rw	r	rw		rw	rw			rw	rw			rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP1LOCK	COMP1OUT	COMP1HYST[1:0]		COMP1POL	COMP1PINSEL[2:0]			WNDWEN1	COMP1NINSEL[2:0]			COMP1MODE[1:0]		Res	COMP1EN
rw	r	rw		rw	rw			rw	rw			rw			rw

位 31	COMP2LOCK：COMP2 有关寄存器锁定位（Comparator 2 lock） 该位由软件置 1。通过硬件系统复位清零，用于锁定比较器 2 控制寄存器（COMP_CSR1[31:16]）的全部内容。 ● 0：比较器 2 的 COMP_CSR1[31:16]可读/写 ● 1：比较器 2 的 COMP_CSR1[31:16]只读
位 30	COMP2OUT：比较器 2 输出状态位（Comparator 2 output） 该位为只读，用于反映当前比较器 2 输出的状态（受到 COMP1POLARITY 位的影响）。 ● 0：低输出（同相输入低于反相输入）

	● 1：高输出（同相输入高于反相输入）
位 29:28	COMP2HYST[1:0]：比较器 2 迟滞（Comparator 2 hysteresis） ● 00：保留 ● 01：无迟滞 ● 10：低迟滞 ● 11：高迟滞
位 27	COMP2POL：比较器 2 极性选择位（Comparator 2 output polarity） 该位由软件置位和清零（COMP2LOCK 置 1 时除外），用于使比较器 2 的极性反相。 ● 0：比较器 2 输出值不反相 ● 1：比较器 2 输出值反相
位 26:24	COMP2PINSEL[2:0]：比较器 2 正端 MUX 选择位（Comparator 2 non inverting input MUX selection） ● 000：选择 PA14 CMP2_P0（OPAMPx_PGA_N） ● 001：选择 PA7 CMP2_P1 ● 其它：保留
位 23	WNDWEN2：比较器 2 正端开关选择位（Comparator 2 non inverting input switch selection） ● 0：比较器 2 正端 MUX 输出连接到比较器 2 的正端 ● 1：比较器 1 正端 MUX 输出连接到比较器 2 的正端
位 22:20	COMP2NINSEL[2:0]：比较器 2 负端 MUX 选择位（Comparator 2 inverting input MUX selection） ● 000：选择 PA10 CMP2_N0 ● 001：选择 PC9 CMP2_N1 ● 010：选择内部 DAC1 输出 DAC1_VOUT ● 其它：保留
位 19:18	COMP2MODE[1:0]：比较器 2 模式选择位（Comparator 2 mode selection） 该位由软件置位和清零（COMP2LOCK 置 1 时除外）。 ● 00：高速/高功耗 ● 01：低速/低功耗 ● 其它：保留
位 17	Res：保留 必须保持复位值。
位 16	COMP2EN：比较器 2 使能位（Comparator 2 enable） 该位由软件置位和清零（COMP2LOCK 置 1 时除外）。 ● 0：关闭比较器 2 ● 1：开启比较器 2
位 15	COMP1LOCK：COMP1 有关寄存器锁定位（Comparator 1 lock） 该位由软件置 1。通过硬件系统复位清零，用于锁定比较器 1 控制寄存器（COMP_CSR1[15:0]）的全部内容。 ● 0：比较器 1 的 COMP_CSR1[15:0]可读/写 ● 1：比较器 1 的 COMP_CSR1[15:0]只读
位 14	COMP1OUT：比较器 1 输出状态位（Comparator 1 output） 该位为只读，用于反映当前比较器 1 输出的状态（受到 COMP1POL 位的影响）。

位 13:12	COMP1HYST[1:0]：比较器 1 迟滞（Comparator 1 hysteresis） ● 00：保留 ● 01：无迟滞 ● 10：低迟滞 ● 11：高迟滞
位 11	COMP1POL：比较器 1 极性选择位（Comparator 1 output polarity） 该位由软件置位和清零（COMP1LOCK 置 1 时除外），用于使比较器 1 的极性反相。 ● 0：比较器 1 输出值不反相 ● 1：比较器 1 输出值反相
位 10:8	COMP1PINSEL[2:0]：比较器 1 正端 MUX 选择位（Comparator 1 non inverting input MUX selection） ● 000：选择 PA14 CMP1_P0（OPAX_PGA_N） ● 001：选择 PA6 CMP1_P1 ● 其它：保留
位 7	WNDWEN1：比较器 1 正端开关选择位（Comparator 1 non inverting input switch selection） ● 0：比较器 1 正端 MUX 输出连接到比较器 1 的正端 ● 1：比较器 2 正端 MUX 输出连接到比较器 1 的正端
位 6:4	COMP1NINSEL[2:0]：比较器 1 负端 MUX 选择位（Comparator 1 inverting input MUX selection） ● 000：选择 PA10 CMP1_N0 ● 001：选择 PC8 CMP1_N1 ● 010：选择内部 DAC1 输出 DAC1_VOUT ● 其它：保留
位 3:2	COMP1MODE[1:0]：比较器 1 模式选择位（Comparator 1 mode selection） 该位由软件置位和清零（COMP1LOCK 置 1 时除外）。 ● 00：高速/高功耗 ● 01：低速/低功耗 ● 其它：保留
位 1	Res：保留 必须保持复位值。
位 0	COMP1EN：比较器 1 使能位（Comparator 1 enable） 该位由软件置位和清零（COMP1OCK 置 1 时除外）。 ● 0：关闭比较器 1 ● 1：开启比较器 1

11.4.2 COMP3 和 COMP4 控制和状态寄存器（COMP_CSR2）

偏移地址：0x120

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMP4LOCK	COMP4OUT	COMP4HYST[1:0]		COMP4POL	COMP4PINSEL[2:0]			Res	COMP4NINSEL[2:0]			COMP4MODE[1:0]		Res	COMP4EN
rw	r	rw		rw	rw				rw			rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMP3LOCK	COMP3OUT	COMP3HYST[1:0]		COMP3POL	COMP3PINSEL[2:0]			Res	COMP3NINSEL[2:0]			COMP3MODE[1:0]		Res	COMP3EN
rw	r	rw		rw	rw				rw			rw			rw

位 31	COMP4LOCK：COMP4 有关寄存器锁定位（Comparator 4 lock） 该位由软件置 1。通过硬件系统复位清零，用于锁定比较器 4 控制寄存器（COMP_CSR[31:16]）的全部内容。 ● 0：比较器 4 的 COMP_CSR2[31:16]可读/写 ● 1：比较器 4 的 COMP_CSR2[31:16]只读
位 30	COMP4OUT：比较器 4 输出状态位（Comparator 4 output） 该位为只读，用于反映当前比较器 4 输出的状态（受到 COMP4POL 位的影响）。 ● 0：低输出（同相输入低于反相输入） ● 1：高输出（同相输入高于反相输入）
位 29:28	COMP4HYST[1:0]：比较器 4 迟滞（Comparator 4 hysteresis） ● 00：保留 ● 01：无迟滞 ● 10：低迟滞 ● 11：高迟滞
位 27	COMP4POL：比较器 4 极性选择位（Comparator 4 output polarity） 该位由软件置位和清零（COMP4LOCK 置 1 时除外），用于使比较器 4 的极性反相。 ● 0：比较器 4 输出值不反相 ● 1：比较器 4 输出值反相
位 26:24	COMP4PINSEL[2:0]：比较器 4 正端 MUX 选择位（Comparator 4 non inverting input MUX selection） ● 000：选择 PA14 CMP4_P0（OPAx_PGA_N） ● 001：选择 PA9 CMP4_P1 ● 其它：保留
位 23	Res：保留 必须保持复位值。
位 22:20	COMP4NINSEL[2:0]：比较器 4 负端 MUX 选择位（Comparator 4 inverting input MUX selection） ● 000：选择 PA10 CMP4_N0 ● 001：选择 PC11 CMP2_N1 ● 010：选择内部 DAC1 输出 DAC1_VOUT ● 其它：保留
位 19:18	COMP4MODE[1:0]：比较器 4 模式选择位（Comparator 2 mode selection） 该位由软件置位和清零（COMP4LOCK 置 1 时除外）。 ● 00：高速/高功耗 ● 01：低速/低功耗 ● 其它：保留
位 17	Res：保留 必须保持复位值。
位 16	COMP4EN：比较器 4 使能位（Comparator 4 enable）

	该位由软件置位和清零（COMP4LOCK 置 1 时除外）。 ● 0：关闭比较器 4 ● 1：开启比较器 4
位 15	COMP3LOCK：COMP3 有关寄存器锁定位（Comparator 3 lock） 该位由软件置 1。通过硬件系统复位清零，用于锁定比较器 3 控制寄存器（COMP_CSR2[15:0]）的全部内容。 ● 0：比较器 3 的 COMP_CSR2[15:0]可读/写 ● 1：比较器 3 的 COMP_CSR2[15:0]只读
位 14	COMP3OUT：比较器 3 输出状态位（Comparator 3 output） 该位为只读，用于反映当前比较器 3 输出的状态（受到 COMP3POL 位的影响）。
位 13:12	COMP3HYST[1:0]：比较器 3 迟滞（Comparator 3 hysteresis） ● 00：保留 ● 01：无迟滞 ● 10：低迟滞 ● 11：高迟滞
位 11	COMP3POL：比较器 3 极性选择位（Comparator 3 output polarity） 该位由软件置位和清零（COMP3LOCK 置 1 时除外），用于使比较器 3 的极性反相。 ● 0：比较器 3 输出值不反相 ● 1：比较器 3 输出值反相
位 10:8	COMP3PINSEL[2:0]：比较器 3 正端 MUX 选择位（Comparator 3 non inverting input MUX selection） ● 000：选择 PA14 CMP3_P0（OPAx_PGA_N） ● 001：选择 PA8 CMP3_P1 ● 其它：保留
位 7	Res：保留 必须保持复位值。
位 6:4	COMP3NINSEL[2:0]：比较器 3 负端 MUX 选择位（Comparator 3 inverting input MUX selection） ● 000：选择 PA10 CMP3_N0 ● 001：选择 PC10 CMP3_N1 ● 010：选择内部 DAC1 输出 DAC1_VOUT ● 其它：保留
位 3:2	COMP3MODE[1:0]：比较器 3 模式选择位（Comparator 3 mode selection） 该位由软件置位和清零（COMP3LOCK 置 1 时除外）。 ● 00：高速/高功耗 ● 01：低速/低功耗 ● 其它：保留
位 1	Res：保留 必须保持复位值。
位 0	COMP3EN：比较器 3 使能位（Comparator 3 enable） 该位由软件置位和清零（COMP3OCK 置 1 时除外）。 ● 0：关闭比较器 1

	● 1：开启比较器 1

11.4.3 COMP 滤波控制寄存器（COMP_FLT_CTRL）

偏移地址：0x124

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												COMP4FEN	COMP3FEN	COMP2FEN	COMP1FEN
												rw	rw	rw	rw

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

位 31:20	Res：保留 必须保持复位值。
位 19	COMP4FEN：比较器 4 滤波使能位（Comparator 4 filter enable） 该位由软件置位和清零。 ● 0：关闭比较器 4 滤波功能 ● 1：开启比较器 4 滤波功能
位 18	COMP3FEN：比较器 3 滤波使能位（Comparator 3 filter enable） 该位由软件置位和清零。 ● 0：关闭比较器 3 滤波功能 ● 1：开启比较器 3 滤波功能
位 17	COMP2FEN：比较器 2 滤波使能位（Comparator 2 filter enable） 该位由软件置位和清零。 ● 0：关闭比较器 2 滤波功能 ● 1：开启比较器 2 滤波功能
位 16	COMP1FEN：比较器 1 滤波使能位（Comparator 1 filter enable） 该位由软件置位和零。 ● 0：关闭比较器 1 滤波功能 ● 1：开启比较器 1 滤波功能
位 15:0	COMPFCLK_CNT：比较器滤波采样时钟计数器配置位（Comparator filter sample clock counter configure） 该位有软件配置，用于设置滤波时钟的计数宽度，当计数器到达设置值时，滤波采样一次比较器输出。比较器固定连续 8 次滤波采样。需先配置该位，再使能比较器滤波功能。

12 模拟运算放大器（OPAMP）

器件集成了 3 个运算放大器（下文简称“运放”）。它们可以工作在 Standalone、Follower、PGA 三种模式。

运放的输出，可以输出到管脚，也可以内部反馈到反相输入端，还可以被选通输入到内部 ADC 进行采样。

12.1 OPAMP 主要特性

- 轨到轨输入输出
- 低失调电压
- 支持 Standalone、Follower、PGA 三种工作模式
- 多路输入复用

12.2 OPAMP 功能说明

每一个运放，正端支持从外部管脚 3 选 1 输入，负端支持从外部管脚 2 选 1 输入。输入选择通过寄存器 OPAMPx_CSR 的 VM_SEL[1:0]/VP_SEL[1:0]进行配置。

被用作运放输入输出的 GPIO 必须配置为模拟工作模式。

表 12-1 运放端口和 GPIO 的对应关系

OPAMP1 反相输入	OPAMP1 同相输入	OPAMP1 输出	OPAMP1 外部地输入
PA8(OPAMP1_N1)	PA9(OPAMP1_P2)	PA7(OPAMP1_OUT1)	PA14
PB0(OPAMP1_N0)	PA13(OPAMP1_P1)	PA15(OPAMP1_OUT0)	-
-	PB1(OPAMP1_P0)	-	-
OPAMP2 反相输入	OPAMP2 同相输入	OPAMP2 输出	OPAMP2 外部地输入
PA3 (OPAMP2_N1)	PA9(OPAMP2_P2)	PA2(OPAMP2_OUT1)	PA14
PA12 (OPAMP2_N0)	PA13(OPAMP2_P1)	PA11(OPAMP2_OUT0)	-
-	PB1(OPAMP2_P0)	-	-
OPAMP3 反相输入	OPAMP3 同相输入	OPAMP3 输出	OPAMP3 外部地输入
PC10 (OPAMP3_N1)	PC4(OPAMP3_P2)	PC11(OPAMP3_OUT2)	-
PB0 (OPAMP3_N0)	PA10(OPAMP3_P1)	PA11(OPAMP3_OUT1)	-
-	PA14(OPAMP3_P0)	PA15(OPAMP3_OUT0)	-

12.2.1 OPAMP 框图

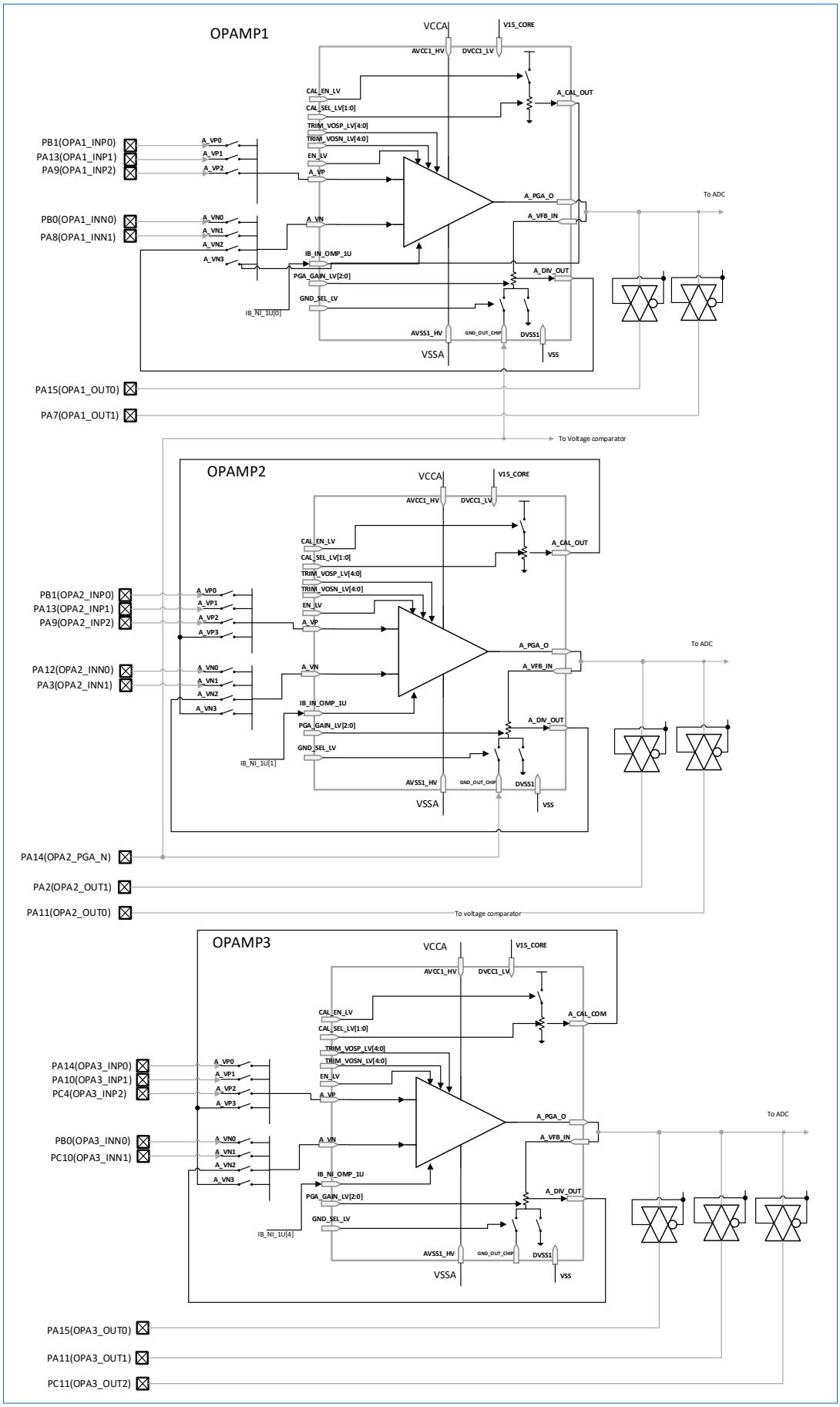

图 12-1 OPAMP 框图

12.2.2 时钟

运放的时钟同步于 APB，在 RCC 控制器中有单独的 OPAMP 时钟控制位。

12.2.3 ADC 采样 OPAMP 输出

表 12-2 ADC 采样 OPAMP 输出

OPAMP	ADC / ADC Sampe&Hold	ADC 通道	OPAMP 输出引脚
OPAMP1	ADC1	AIN11(2)	PA7(1)
	ADC1 Sampel&Hold3)	AIN10(3)	-
OPAMP2	ADC1	AIN13(2)	PA2(1)
	ADC1 Sampel&Hold(3)	AIN11(3)	-
OPAMP3	ADC1	AIN14(2)	-
	ADC1 Sampel&Hold(3)	AIN12(3)	-

注意：

(1). 在使能运放后，使能运放对应的输出管脚，将被一直占用。

(2). 在使能运放后，对应的 ADC 输入通道也被占用，是否选通采样由 ADC 通道选择器控制。

(3). 在使能运放后，使能 Sample&Hold 功能，选择 Sample＆Hold0 或者 Sample&Hold1 的对应通道。

12.2.4 校正

运放的轨到轨输入级由 2 组差分对 MOS 管组成：一组 NMOS 差分对管和一组 PMOS 差分对管。运放在使用之前，可以对 OPAMP 进行校正。通过配置硬件校准时钟选择寄存器，以及 CALON 和 CALSEL，使能硬件校准 HWT_EN，等待 HWT_EN 清除，完成 PMOS/NMOS 管校准。

推荐校准点选择：

- x1 增益时，选择 PMOS 与 NMOS 交替校准(CALSEL=11 or 01)；
- x2、x5、x8 增益时，选择只校准 PMOS(CALSEL = 10)；
- x10、x12、x16、x20 增益时，选择只校准 PMOS(CALSEL = 10)；
- 输入持续>50%*AVCC1_HV，选择只校准 NMOS(CALSEL = 00);

用户也可以通过设置运放控制器的 USER_TRIM 位，再次对运放进行 Trimming 操作。

TRIMVOSN 寄存器位负责 NMOS 差分对管校正，TRIMVOSP 寄存器位负责 PMOS 差分对管校正。

内部 0.9*VDDA 电压会接到运放的正负端。正负输入端的电压可以被测量。软件配置 TRIMVOSN 寄存器，从 0x0 逐渐增加，一直到 CALOUT 从 1 变为 0。如果 OUTCAL 位被复位，说明运放 Offset 被正确校正，并且对应的 Trimming 值将会被保存起来。

NMOS 差分对，软件校正流程如下：

1. 设置 OPAMPEN 位。
2. 设置 USER_TRIM 位。
3. 设置 CALON 位，芯片内部会连接运放 VM、VP 到内部参考电压。
4. 设置 CALSEL=01。

5. 逐渐增加 TRIMVOSN，直到 CALOUT 被清零，保存 TRIMVOSN。

对于 PMOS 软件配置流程完全相同，只是需要配置 CALSEL=01，TRIMVOSP 寄存器需要被软件配置。内部电压 0.05*VDDA 会接到运放的正负输入端。

12.2.5 OPAMP 工作模式

运放的输入和输出，都可连接到 GPIO 上。可以支持三种工作模式：

- Standalone 模式
- Follower 模式
- PGA 模式

Standalone

Standalone 模式的增益反馈网络在芯片外，运放的正负输入以及输出都连接到管脚，用户电路设计灵活。通过设置 OPAMPx_CSR.VP_SEL 对正端输入管脚进行 3 选 1。通过设置 OPAMPx_CSR.VM_SEL 对负端输入管脚进行 2 选 1。运放输出，通过芯片外围电路与反相输入端连接。

不管是正端，还是负端，没有选通使用的 IO，都可以当作 GPIO 使用。

表 12-3 运放反相输入端选择配置(x 为 1 或 2)

VM_SEL	管脚
00	OPAMPx_N0
01	OPAMPx_N1

表 12-4 运放正相输入端选择配置(x 为 0 或 1)

VP_SEL	管脚
00	OPAMPx_P0
01	OPAMPx_P1
10	OPAMPx_P2

结构如下图：

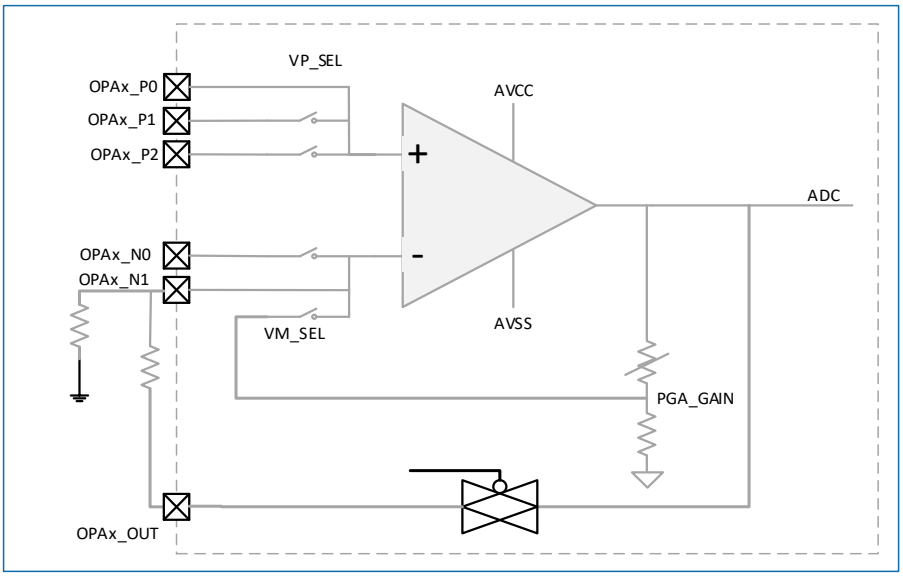

图 12-2 Standalone 结构图

Follower

配置 VM_SEL=10，配置 PGA GAIN 为 X1 时，运放工作为跟随器模式。该模式对信号起到驱动作用，并提供高的输出阻抗。

运放的输出在芯片内部与反相输入端连接。通过设置 OPAMPx_CSR 寄存器的 VP_SEL 对正端输入管脚进行 3 选 1。

正端没有选通使用的 IO 口，都可以当作 GPIO 使用。OPAMPx_N0/OPAMPx_N1 管脚都可以当作 GPIO 使用。

结构如下图：

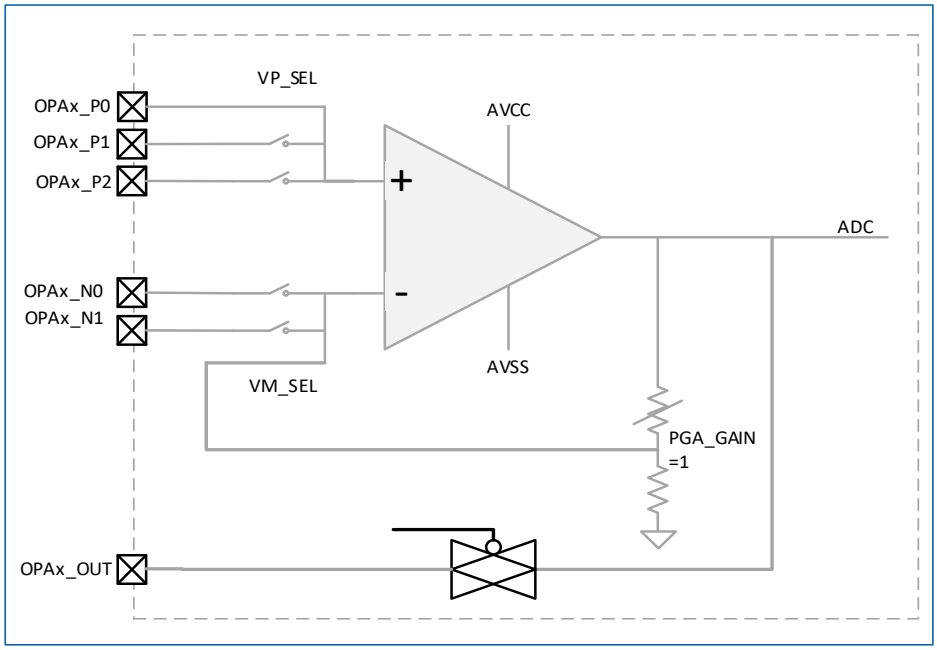

图 12-3 Follower 结构图

PGA

配置 VM_SEL=10，运放工作为内部增益可编程模式。负反馈增益网络通过 PGA_GAIN 寄存器设置，增益固定有 2/5/8/10/14/16/20 倍。通过设置 OPAMPx_CSR 寄存器的 VP_SEL 对正端输入管脚进行 3 选 1。

在这个模式下，支持两种应用电路。

电路一：OPAMPx_N0/OPAMPx_N1 管脚不需要，内部增益可编程，内部增益 x2/x5/x8/x10/14/x16/x20 可选

正端没有选通使用的 IO 口，都可以当作 GPIO 使用。OPAMPx_N0/OPAMPx_N1 管脚都当作 GPIO 使用。

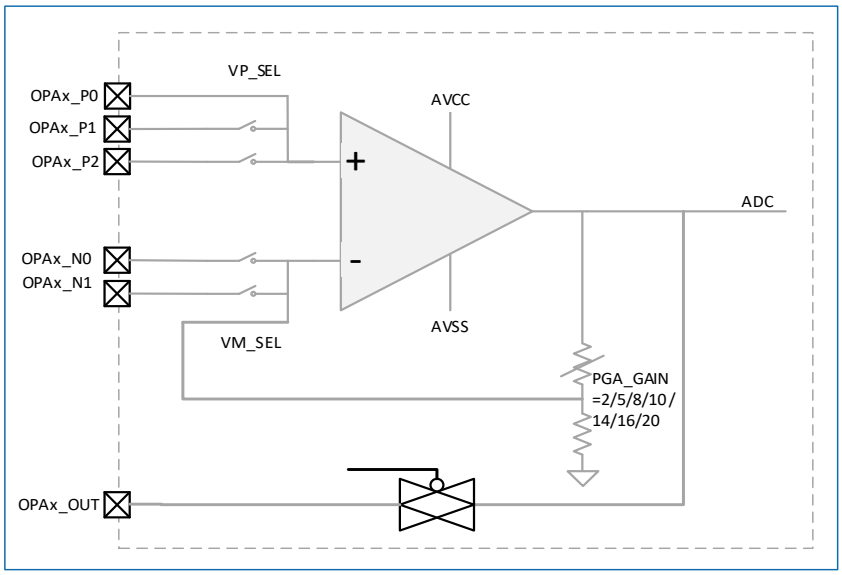

图 12-4 PGA 应用电路 1

电路二：有源低通滤波器，OPAMPx_N0/OPAMPx_N1 管脚需要选通，并且 OPAMPx_N0/OPAMPx_N1 管脚与运放输出管脚之间有外接电路，内部增益可编程，内部增益 x2/x5/x8/x10/14/x16/x20 可选。

正端没有选通使用的 IO，都可以当作 GPIO 使用。负端 OPAMPx_N0/OPAMPx_N1，没有选通的管脚可以当作 GPIO 使用。

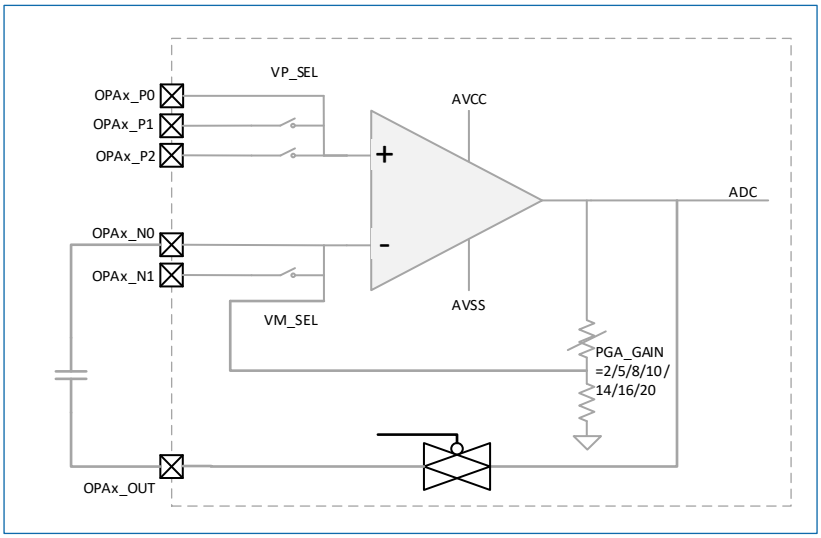

图 12-5 PGA 应用电路 2

PGA_GAIN 寄存器具体设置请参照寄存器有关描述。

12.3 OPAMP 寄存器

基地址：（OPAMP1;OPAMP2; OPAMP3） = （0x4001 5C00; 0x4001 5C04; 0x4001 5C08）

空间大小：（OPAMP1; OPAMP2; OPAMP3） = （0x14; 0x10; 0x0C）

12.3.1 运放控制寄存器（OPAMP_CSR）

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
LOCK	CALOUT	HWT_EN	TRIMVOSN[4:0]					TRIMVOSP[4:0]					PGA_GAIN[4:2]		
rw	r	rw	rw					rw					rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PGA_GAIN[1:0]		CALSEL[1:0]		CALON	OPAMPx_OUT_SEL2	OPAMPx_OUT_SEL1	OPAMPx_OUT_SEL0	GND_SEL	VM_SEL[1:0]		USER_TRIM	VP_SEL[1:0]		FORCE_VP	OPAMPEN
rw		rw		rw	rw	rw	rw	rw	rw		rw	rw		rw	rw

位 31	LOCK：运放 x 有关寄存器锁定位（OPAMPx_CSR register lock） 这一位由软件置位，可以被系统复位清零。置位将使 CSR 寄存器只读。 ● 0：运放 x 有关寄存器可读/写 ● 1：运放 x 有关寄存器只读
位 30	CALOUT：运放校准时输出结果（Operational amplifier calibration output）
位 29	HWT_EN：硬件校准使能位（Hardware trimming enable） 使能硬件校准，完成校准后自动清除。
位 28:24	TRIMVOSN[4:0]：Offset 校准值（Trim offset of NMOS） 该位只能当 USER_TRIM 位使能时，更改有效。
位 23:19	TRIMVOSP[4:0]：Offset 校准值（Trim offset of PMOS） 该位只能当 USER_TRIM 位使能时，更改有效。
位 18:14	PGA_GAIN[4:0]：PGA 增益配置位（Gain in PGA mode configuration） ● 00000：同相增益= 1，OPAMPx_N0/ OPAMPx_N1 无连接(Follower mode) ● 00001：同相增益= 2，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00010：同相增益= 5，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00011：同相增益= 8，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00100：同相增益= 10，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00101：同相增益= 14，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00110：同相增益= 16，OPAMPx_N0/ OPAMPx_N1 无连接 ● 00111：同相增益= 20，OPAMPx_N0/ OPAMPx_N1 无连接 ● 10001：同相增益= 2 - OPAMPx_N0 滤波 ● 10010：同相增益= 5 - OPAMPx_N0 滤波 ● 10011：同相增益= 8 - OPAMPx_N0 滤波 ● 10100：同相增益= 10 - OPAMPx_N0 滤波 ● 10101：同相增益= 14 - OPAMPx_N0 滤波 ● 10110：同相增益= 16 - OPAMPx_N0 滤波 ● 10111：同相增益= 20 - OPAMPx_N0 滤波
位 13:12	CALSEL[1:0]：校准基准电压选择（Calibration selection） 软件可以置位和清零，用于选择校准时的基准电压值，需 CALON = 1 或者 FORCE_VP= 1。 ● 00：只校准 NMOS ● 10：只校准 PMOS ● 其它：交替校准 PMOS/NMOS
位 11	CALON：校准模式使能位（Calibration mode enable） 软件可以置位和清零，用于使能校准模式，此时 VM 和 VP 连接到运放内部参考电压。 ● 0：校准模式禁能 ● 1：校准模式使能

位 10	● OPAMPx_OUT_SEL2：运放 x 输出选择位 2（OPAMPx output selection 2） ● 软件可以置位和清零，用于选择 OPAMPx_OUT 输出开关使能。 ● 0：运放 x 不输出到 IO OPAMPx_OUT2 ● 1：运放 x 输出到 IO OPAMPx_OUT2
位 9	● OPAMPx_OUT_SEL1：运放 x 输出选择位 1（OPAMPx output selection 1） ● 软件可以置位和清零，用于选择 OPAMPx_OUT 输出开关使能。 ● 0：运放 x 不输出到 IO OPAMPx_OUT1 ● 1：运放 x 输出到 IO OPAMPx_OUT1
位 8	● OPAMPx_OUT_SEL0：运放 x 输出选择位 0（OPAMPx output selection 0） ● 软件可以置位和清零，用于选择 OPAMPx_OUT 输出开关使能。 ● 0：运放 x 不输出到 IO OPAMPx_OUT0 ● 1：运放 x 输出到 IO OPAMPx_OUT0
位 7	● GND_SEL：运放 x 反馈回路地选择位（OPAMPx feedback loop GND selection） ● 软件可以置位和清零，用于选择运放 x 反馈环路地。 ● 0：内部地作为运放 x 反馈回路的地 ● 1：外部的地 PGA_N 作为反馈回路的地
位 6:5	VM_SEL[1:0]：运放 x 负端输入选择位（OPAMPx Minus input selection） 软件可以置位和清零。 ● 00：OPAMPx_N0 为反相 VM 输入 ● 01：OPAMPx_N1 为反相 VM 输入 ● 10：PGA 模式 mode / follower mode ● 11：保留
位 4	USER_TRIM：用户校准使能位（User trimming enable） ● 0：用户校准禁能 ● 1：用户校准使能
位 3:2	VP_SEL[1:0]：运放 x 正端输入选择位（OPAMPx positive input selection） 软件可以置位和清零。 ● 00：OPAMPx_P0 作为同相 VP 输入 ● 01：OPAMPx_P1 作为同相 VP 输入 ● 10：OPAMPx_P2 作为同相 VP 输入 ● 11：保留
位 1	FORCE_VP：强制校准选择位（Forces calibration） ● 0：正常操作模式：同相 VP 输入被连接到输入 ● 1：校准模式：同相 VP 输入被连接到校准参考电压 A_CAL_OUT
位 0	OPAMPEN：运放 x 使能位（OPAMPx enable） 软件可以置位和清零。 ● 0：OPAMPx 禁能 ● 1：OPAMPx 使能

12.3.2 运放硬件校准时钟选择寄存器（OPAMP_CKSEL）

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											DIV1024_SEL	Res			
											rw				

位 31:5	Res：保留 必须保持复位值。
位 4	DIV1024_SEL：APB clock 1024 分频选择作为硬件校准时钟（APB clock 1024-division as hardware trimming clock）
位 3:0	Res：保留 必须保持复位值。

13 电机加速单元（EMACC）

电机加速单元（EMACC）可用于通过磁场定向控制（Field Oriented Control，FOC）算法控制的直流无刷电机。EMACC 可以加速电机驱动的数学运算，运算速度较纯软件计算更快，并且减少 CPU 占用，对于相同的 CPU 工作频率，可以支持更高的电机转速和驱动频率。

器件支持对整个 FOC 算法中 CPU 耗费大量时间参与运算的 Cordic 运算、Clarke 变换、Park 变换、反 Park 变换、PID 算法单元、SVPWM 模块进行硬件化。用户输入 Ia、Ib 电流和 θ 电角度，通过 EMACC 电机加速单元运算之后，得到空间矢量脉宽调制（SVPWM）的输出：A\B\C 相的 PWM 占空比等；从而节约 FOC 算法时间。经过 EMACC 单元处理的计算，效率可大幅度提升。

器件包含一个数据追踪器（EMACC_TRACE），实际为一个硬件化的高速串口模块，用于在电机调试过程中或者高速运行时，实时输出 EMACC 模块的参数。还额外提供 4 个字节用于用户自定义数据输出，便于电机调试。

13.1 EMACC 功能

13.1.1 电机加速单元主要特性

- Cordic 运算单元：可用于计算正余弦值。
- Clarke 变换计算单元：可由用户输入的相电流 Ia、Ib，得到 I_α 和 I_β。
- Park 变换计算单元：通过 Clarke 变换计算结果 I_α 和 I_β 作为输入，输出 Iq 和 Id。
- PI 运算单元：可由 Park 变换结果 Iq 和 Id，得到 Vq 和 Vd，该单元还包含了循环限制算法。
- 反 Park 变换计算单元：通过 Park 变换计算结果 Iq 和 Id 作为输入，输出 U_α 和 U_β。
- SVPWM 运算单元：通过反 Park 变换计算结果 U_α 和 U_β 作为输入，最后输出 A\B\C 相的 PWM 占空比、扇区、采样点、ADC 采样通道等参数。

13.1.2 电机数据追踪模块（TRACE）主要特性

- 只能输出数据
- 实时输出 EMACC 模块的参数
- 额外提供 4 个字节用于自定义数据输出
- 可调的数据传输波特率 12Mbps（Max）/8Mbps/6Mbps/4Mbps/2Mbps
- 选择硬件或者软件触发数据传输
- 通过配置 IO 和 AF，即可完成 Trace 功能配置

13.1.3 电机加速单元框图

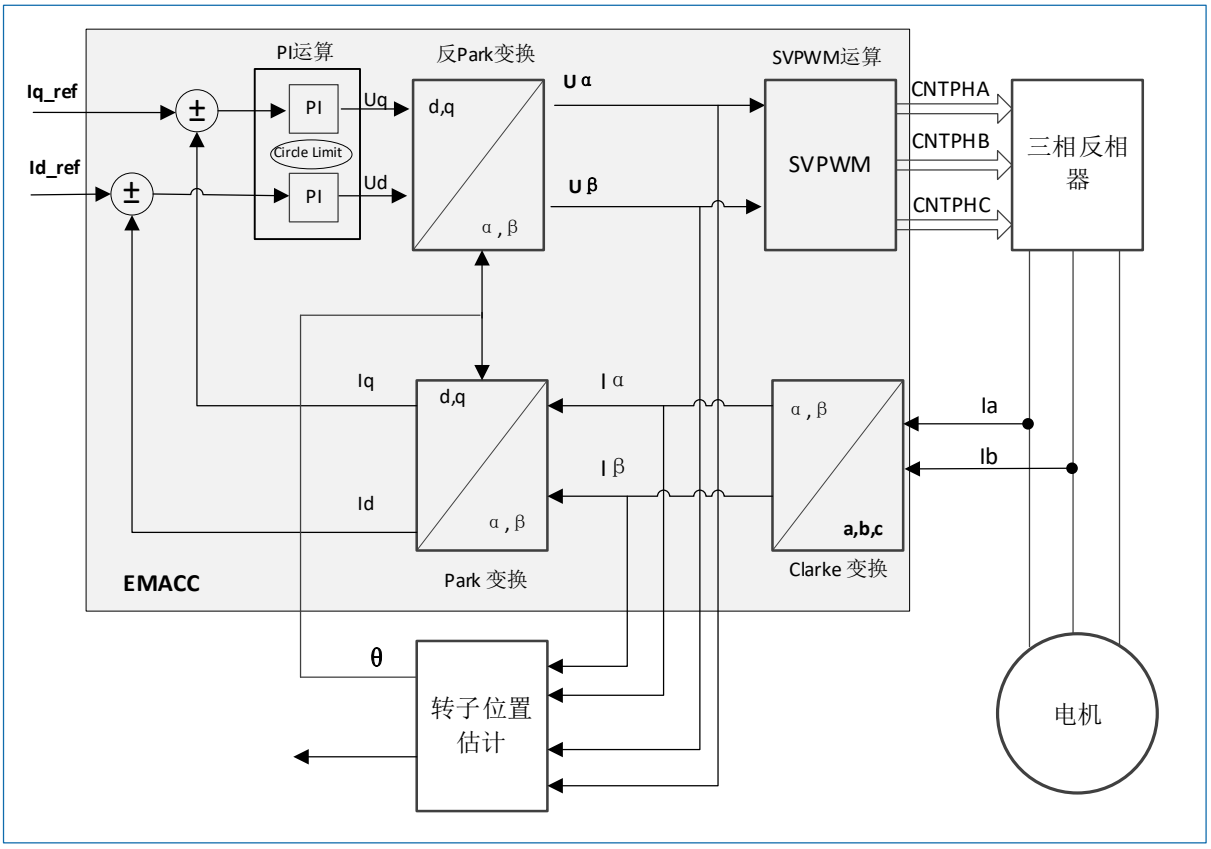

图 13-1 电机加速单元框图

如上图所示，电机硬件加速单元实现了正余弦计算，Clarke 变换，Park 变换，PI 运算（包含了 Circle Limit），反 Park 变换，SVPWM 运算。

- EMACC 的输入参数：相电流 Ia、Ib 和 Iq_ref、Id_ref 和电角度 θ 。

输入参数对应的寄存器位请参见：13.2.11 CLARKE 值载入寄存器（EMACC_CLARKE_LDR），13.2.19 D 轴 PID 参数参考输入配置寄存器（EMACC_ID_REF），13.2.10 电角度载入寄存器（EMACC_ANG_LDR）。

- EMACC 的输出参数：A\B\C 相的 PWM 占空比（图 13-1 中的 CNTPHA，CNTPHB 和 CNTPHC）、扇区、采样点、ADC 采样通道等。

输出参数对应的寄存器位请参见 13.2.32 三相驱动信号的 PWM 占空比 1 寄存器（EMACC_SVPWM_CNTPH1），13.2.33 三相驱动信号的 PWM 占空比 2 寄存器（EMACC_SVPWM_CNTPH2），13.2.35 SVPWM 的各项其它输出寄存器（EMACC_SVPWM_OUTR）。

13.1.4 电机加速单元工作流程

1. 用户将 ADC 采集到的相电流 Ia、Ib 输入到 Clarke 计算单元。
2. 使用软件位置估算算法得到前一个时刻的电角度 θ，或霍尔传感器得到电机转子当前的电角度 θ。
3. 将电角度 θ 赋给 Cordic 计算单元得到对应的正余弦值。
4. 将 FOC 算法计算过程中的 Iq_ref 和 Id_ref 分别传入 EMACC 的 EMACC_ID_REFF 和 EMACC_IQ_REFF 寄存器。
5. 等待 EMACC 电机硬件加速单元完成计算（通过状态寄存器查看）。

6. 计算完成后，获得并读取 CNTPHA、CNTPHB、CNTPHC，数据被读取后，清除状态位。

7. 最终输出结果在 EMACC_SVPWM_CNTPH1/EMACC_SVPWM_CNTPH2/EMACC_SVPWM_CNTSMP/EMACC_SVPWM_OUTR 寄存器中，可参考该寄存器获取数据。

13.1.5 电机数据追踪模块（TRACE）可选择的输出参数

通过配置 EMACC_TRACE_CFGR 寄存器相应的传输选择位，选择想要发送的数据（1 表示发送该数据，0 表示不发送该数据）。

表 13-1 电机数据追踪模块可选的输出参数列表

输出数据（按输出顺序排列）	位宽	对应软件程序变量名称	释义
UART_SYNCODE.DATA	16 位	同步头	-
CORDIC_ANG_LDR.CORDIC_ANG_IN	16 位	hElAngle	电角度
CLARKE_LDR.IA	16 位	Ia	Clarke 输入 A 相电流
CLARKE_LDR.IB	16 位	Ib	Clarke 输入 B 相电流
CLARKE_OUT.IALPHA	16 位	Ialpha	Clarke 输出 α 轴电流
CLARKE_OUT.IBETA	16 位	Ibeta	Clarke 输出 β 轴电流
PARK_OUT.IQ	16 位	Iq	Park 输出 Q 轴电流
PARK_OUT.ID	16 位	Id	Park 输出 D 轴电流
IDQ_REF.IQREF	16 位	Iq reference	PID 变换 Q 轴参考电流
IDQ_REF.IDREF	16 位	Id reference	PID 变换 D 轴参考电流
PID_OUT.VQ	16 位	Vq	PID 输出 Q 轴电压
PID_OUT.VD	16 位	Vd	PID 输出 D 轴电压
IPARK_OUT.VALPHA	16 位	Valpha	反 Park 输出 α 轴电压
IPARK_OUT.VBETA	16 位	Vbeta	反 Park 输出 β 轴电压
SVPWM_OUT1.CNTPHA	16 位	hCntPhA	SVPWM 输出 A 相占空比
SVPWM_OUT2.CNTPHB	16 位	hCntPhB	SVPWM 输出 B 相占空比
SVPWM_OUT3.CNTPHC	16 位	hCntPhC	SVPWM 输出 C 相占空比
EMACC_CORDIC_OUT.SINX	16 位	-	CORDIC 输出 sinx 值
EMACC_CORDIC_OUT.COSX	16 位	-	CORDIC 输出 cosx 值
UART_USRTXD1.DATA	16 位	用户自定义	-
UART_USRTXD2.DATA	16 位	用户自定义	-
UART_USRTXD3.DATA	16 位	用户自定义	-
UART_USRTXD4.DATA	16 位	用户自定义	-

13.1.6 电机数据追踪模块（TRACE）数据传输波特率

EMACC_TRACE 工作时钟由 RCC 的 AHB 提供，波特率和工作时钟一致，可通过 RCC_AHBENR.TRACE_UART_PREDIV[3:0]寄存器中相应的分频配置得到。

波特率支持 12Mbps（Max）/8Mbps/6Mbps/4Mbps/2Mbps（工作频率整数分频），不需要奇偶校验，停止位固定为 1 比特。

13.1.7 电机数据追踪模块（TRACE）传输数据同步头

配置 EMACC_TRACE_SYNCODE 寄存器，可配置发送数据前的数据同步帧头。

13.1.8 电机数据追踪模块（TRACE）数据传输触发选择

配置 EMACC_TRACE_CR.HWST_EN，选择软件/硬件触发方式发起数据传输。

- HWST_EN = 0，软件触发。
- HWST_EN = 1，硬件触发。

13.1.9 电机数据追踪模块（TRACE）配置步骤

使用 TRACE 功能的配置流程必须按以下顺序操作：

1. 配置 RCC_AHBENR.TRACE_UART_PREDIV，以设置 TRACE 数据发送的波特率；
2. 在 RCC_AHBENR 寄存器中，使能 EMACC 时钟；
3. 根据数据手册的引脚定义，配置 Trace 引脚的 IO；
4. 根据数据手册的引脚复用功能表，在 GPIOx_AFRL/AFRH 寄存器中配置 Trace IO 的复用功能为 Trace 功能；
5. 配置 EMACC_TRACE_CFGR 寄存器，选择要发送的数据；
6. 配置 EMACC_TRACE_CR.HWST_EN 位，选择硬件或软件触发方式；

   A. 如果选择硬件触发，则当对应 EMACC 模块计算完成之后，硬件自动发送 Trace 数据；

   B. 如果选择软件触发，则需要手动使能 EMACC_TRACE_CR.TX_START 位，则 Trace 模块会将选择的数据依次发送出去，发送完成之后，硬件清零该位。如要再次发送数据，需再次使能该位。

13.2 EMACC 寄存器

基地址：0x4002 8000

空间大小：0x400

13.2.1 CORDIC 输出寄存器（EMACC_CORDIC_OUT）

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
COSX[15:0]															
r															

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

位 31:16	COSX[15:0]：余弦值（16 位）（Cosine X value） 该位域仅支持读。
位 15:0	SINX[15:0]：正弦值（16 位）（Sinusoidal X value） 该位域仅支持读。

13.2.2 CLARKE 变换输出寄存器（EMACC_CLARKE_OUT）

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
I_ALPHA[15:0]															
r															

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

位 31:16	I_ALPHA[15:0]：Clarke 变换 Iα 输出结果（16 位）（Output result, Alpha axis of Clarke transformation） 该位域仅支持读。
位 15:0	I_BETA[15:0]：Clarke 变换 Iβ 输出结果（16 位）（Output result, Beta axis of Clarke transformation） 该位域仅支持读。

13.2.3 PARK 变换输出寄存器（EMACC_PARK_OUT）

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
ID[15:0]															
r															

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

位 31:16	ID[15:0]：Park 变换的 D 路输出结果（16 位）（Output result, D axis of Park transform） 该位域仅支持读。
位 15:0	IQ[15:0]：Park 变换的 Q 路输出结果（16 位）（Output result, Q axis of Park transform） 该位域仅支持读。

13.2.4 PID 模块输出寄存器（EMACC_PID_OUT）

偏移地址：0x0C

复位值：0x0000 0000

对于 FOC 算法结构里面的 Circle limit 已在该 PID 环节处理好。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VD[15:0]															
r															

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

位 31:16	VD[15:0]：PID 模块的 D 路输出结果（16 位）（Output result, D axis of PID module） 该位域仅支持读。
位 15:0	VQ[15:0]：PID 模块的 Q 路输出结果（16 位）（Output result, Q axis of PID module） 该位域仅支持读。

13.2.5 REV-PARK 变换输出寄存器（EMACC_REVPARK_OUT）

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
V_ALPHA[15:0]															
r															

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

位 31:16	V_ALPHA[15:0]：反 Park 变换模块的 α 路输出 Vα（Output Vα，Alpha axis of Reverse Park Transform Module） 16 位输出结果。 该位域仅支持读。
位 15:0	V_BETA[15:0]：反 Park 变换模块的 β 路输出 Vβ（Output Vβ，Beta axis of Reverse Park Transform Module） 16 位输出结果。 该位域仅支持读。

13.2.6 中断使能寄存器（EMACC_IER）

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										SVPWM_IE	RPARK_IE	PID_IE	PARK_IE	CLARK_IE	CORDIC_IE
										w	w	w	w	w	w

位 31:6	Res：保留 必须保持复位值。
位 5	SVPWM_IE：SVPWM 的中断使能位（SVPWM interrupt enable） ● 0：禁止 SVPWM 的中断 ● 1：允许 SVPWM 的中断

位 4	RPARK_IE：反 Park 变换的中断使能位（Reverse-PARK interrupt enable） ● 0：禁止反 Park 变换的中断 ● 1：允许反 Park 变换的中断
位 3	PID_IE：PID 变换的中断使能位（PID interrupt enable） ● 0：禁止 PID 变换的中断 ● 1：允许 PID 变换的中断
位 2	PARK_IE：Park 变换的中断使能位（PARK interrupt enable） ● 0：禁止 Park 变换的中断 ● 1：允许 Park 变换的中断
位 1	CLARK_IE：Clarke 变换的中断使能位（CLARKE interrupt enable） ● 0：禁止 Clarke 变换的中断 ● 1：允许 Clarke 变换的中断
位 0	CORDIC_IE：Cordic 变换的中断使能位（CORDIC interrupt enable） ● 0：禁止 Cordic 变换的中断 ● 1：允许 Cordic 变换的中断

13.2.7 状态清除寄存器（EMACC_CLRSR）

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										CLR_SVPWM	CLR_RPARK	CLR_PID	CLR_PARK	CLR_CLARKE	CLR_CORDIC
										w	w	w	w	w	w

位 31:6	Res：保留 必须保持复位值。
位 5	CLR_SVPWM：清除 SVPWM 变换数据准备标记（Reverse-SVPWM data ready flag clear）
位 4	CLR_RPARK：清除反 Park 变换数据准备标记（Reverse-PARK data ready flag clear） 该位仅支持读。 该位由软件写 1 清除，写 0 无效。
位 3	CLR_PID：清除 PID 变换数据准备标记（PID data ready flag clear） 该位仅支持读。 该位由软件写 1 清除，写 0 无效。
位 2	CLR_PARK：清除 Park 变换数据准备标记（PARK data ready flag clear） 该位仅支持读。 该位由软件写 1 清除，写 0 无效。
位 1	CLR_CLARKE：清除 Clarke 变换数据准备标记（CLARKE data ready flag clear）

	该位仅支持读。 该位由软件写 1 清除，写 0 无效。
位 0	CLR_CORDIC：清除 Cordic 变换数据准备标记（CORDIC data ready flag clear） 该位仅支持读。 该位由软件写 1 清除，写 0 无效。

13.2.8 控制寄存器（EMACC_CR）

偏移地址：0x20

复位值：0x0010 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			RAMWR_EN	RAMRD_EN	Res	PARASEL[1:0]		Res	SVPWM_EN	RPARK_EN	PID_EN	PARK_EN	CLARKE_EN	CORDIC_EN	EMACC_EN
			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	RAMWR_EN：允许 SRAM 从 AHB 接口写访问（SRAM write access from AHB interface）
位 11	RAMRD_EN：允许 SRAM 从 AHB 接口读访问（SRAM Read access from AHB interface）
位 10	Res：保留 必须保持复位值。
位 9:8	PARASEL[1:0]：选择读取 D 轴还是 Q 轴的 PID 参数（Choose to read the PID parameters of d-axis or q-axis） ● 01：Q 轴 ● 其他值：D 轴
位 7	Res：保留 必须保持复位值。
位 6	SVPWM_EN：SVPWM 模块的使能位（SVPWM module enable） ● 0：禁能 SVPWM 模块 ● 1：使能 SVPWM 模块 *注意：使用该模块，需要将该寄存器的位 0 和位 1、位 2、位 3、位 4、位 5、位 6 都置为 1。*
位 5	RPARK_EN：反 Park 变换模块使能（Reverse park transformation enable） ● 0：禁能反 Park 变换模块 ● 1：使能反 Park 变换模块 *注意：使用该模块，需要将该寄存器的位 0 和位 1、位 2、位 3、位 4、位 5、位 6 都置为 1。*
位 4	PID_EN：PID 模块使能（PID module enable） ● 0：禁能 PID 模块 ● 1：使能 PID 模块 *注意：使用该模块，需要将该寄存器的位 0 和位 1、位 2、位 3、位 4、位 5、位 6 都置为 1。*

位 3	PARK_EN：Park 变换模块使能（Park transformation enable） ● 0：禁能 Park 变换模块 ● 1：使能 Park 变换模块 *注意：使用该模块，需要将该寄存器的位 0 和位 1、位 2、位 3、位 4、位 5、位 6 都置为 1。*
位 2	CLARKE_EN：Clarke 变换模块使能（Clarke transformation enable） ● 0：禁能 Clarke 变换模块 ● 1：使能 Clarke 变换模块 *注意：使用该模块，需要将该寄存器的位 0 和位 1 都置为 1。*
位 1	CORDIC_EN：CORDIC 模块使能（Cordic module enable） ● 0：禁能 CORDIC 模块 ● 1：使能 CORDIC 模块 *注意：使用该模块，需要将该寄存器的位 0 置为 1。*
位 0	EMACC_EN：EMACC 模块使能（Electric Motor acceleration module enable） ● 0：禁能 EMACC 模块 ● 1：使能 EMACC 模块

13.2.9 状态寄存器（EMACC_SR）

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															TRACE_BUSY
															r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res										SVPWM_CF	RPARK_CF	PID_CF	PARK_CF	CLARKE_CF	CORDIC_CF
										r	r	r	r	r	r

位 31:17	Res：保留 必须保持复位值。
位 16	TRACE_BUSY：数据跟踪调试模块正在运行标记（Trace is busy now） ● 0：表示 TRACE 模块处于空闲状态 ● 1：表示 TRACE 模块处于忙碌或者运行状态
位 15:6	Res：保留 必须保持复位值。
位 5	SVPWM_CF：SVPWM 运算完成标记（SVPWM calculation completed flag） ● 0：表示 SVPWM 运算未完成 ● 1：表示 SVPWM 运算完成
位 4	RPARK_CF：反 Park 变换数据转换完成标记（Reverse-PARK transformation completed flag） 该位仅支持读。 ● 0：表示反 Park 变换数据未开始转换或转换未完成 ● 1：表示反 Park 变换数据转换完成

位 3	PID_CF：PID 变换数据转换完成标记（PID transformation completed flag） 该位仅支持读。 ● 0：表示 PID 变换数据未开始转换或转换未完成 ● 1：表示 PID 变换数据转换完成。
位 2	PARK_CF：Park 变换数据转换完成标记（PARK transformation completed flag） 该位仅支持读。 ● 0：表示 Park 变换数据未开始转换或转换未完成 ● 1：表示 Park 变换数据转换完成
位 1	CLARKE_CF：Clarke 变换数据转换完成标记（CLARKE transformation completed flag） 该位仅支持读。 ● 0：表示 Clarke 变换数据未开始转换或转换未完成 ● 1：表示 Clarke 变换数据转换完成
位 0	CORDIC_CF：Cordic 变换数据转换完成标记（CORDIC transformation completed flag） 该位仅支持读。 ● 0：表示 Cordic 变换数据未开始转换或转换未完成 ● 1：表示 Cordic 变换数据转换完成

13.2.10 电角度载入寄存器（EMACC_ANG_LDR）

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
ANGLE_IN[15:0]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	ANGLE_IN[15:0]：电角度输入（Angle input） 使用软件位置估算算法得到前一个时刻的电角度 θ，或霍尔传感器得到电机转子当前的电角度 θ，放在该寄存器的低 16 位。

13.2.11 CLARKE 值载入寄存器（EMACC_CLARKE_LDR）

偏移地址：0x2C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IA[15:0]															
rw															

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

位 31:16	IA[15:0]：由 ADC 采样得到的 A 相电流值 Ia（IA phase A current value sampled by ADC）
位 15:0	IB[15:0]：由 ADC 采样得到的 B 相电流值 Ib（IB phase B current value sampled by ADC）

13.2.12 CIRCLE_LMT 循环限制算法的参数配置寄存器（EMACC_CIRCLELMT_CFG）

偏移地址：0x30

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								START_INDEX[7:0]							
								w							

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

位 31:24	Res：保留 必须保持复位值。
位 23:16	START_INDEX[7:0]：循环限制算法的数组初始索引值（Initiate index value of limit control array）
位 15:0	MAX_MODULE[15:0]：循环限制算法的最大极限值（Limit Ring Control Maximum of Group）

13.2.13 读取 CIRCLE_LMT 循环限制算法配置参数寄存器（EMACC_READ_CIRCLELMT_CFG）

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								START_INDEX[7:0]							
								w							

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

位 31:24	Res：保留 必须保持复位值。
位 23:16	START_INDEX[7:0]：读取当前循环限制算法的数组初始索引值（Initiate index value of limit control array）
位 15:0	MAX_MODULE[15:0]：读取当前循环限制算法的最大极限值（Limit Ring Control Maximum of Group）

13.2.14 D 轴 PID 参数 KP/KI 配置寄存器（EMACC_D_KPKI）

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
KP[15:0]															
w															

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

位 31:16	KP[15:0]：设置 D 轴 PID 模块的比例计算增益参数（PID gain parameter for proportion computation of D axis）
位 15:0	KI[15:0]：设置 D 轴 PID 模块的积分计算增益参数（PID gain parameter for integration computation of D axis）

13.2.15 D 轴 PID 积分上极限值配置寄存器（EMACC_D_INTUPPER）

偏移地址：0x44

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：设置 D 轴 PID 模块积分上限值（PID integration part upper limit of D axis）

13.2.16 D 轴 PID 积分下极限值配置寄存器（EMACC_D_INTLOWER）

偏移地址：0x48

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：设置 D 轴 PID 模块积分下极限值（PID integration part lower limit of D axis）

13.2.17 D 轴 PID 积分输出极限值配置寄存器（EMACC_D_INTOUT）

偏移地址：0x4C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERLMT[15:0]															
w															

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

位 31:16	UPPERLMT[15:0]：设置 D 轴 PID 模块输出上极限值（PID Group x PI module output data upper limit value of D axis）
位 15:0	LOWERLMT[15:0]：设置 D 轴 PID 模块输出下极限值（PID Group x PI module output data lower limit value of D axis）

13.2.18 D 轴 PID 参数右移位数配置寄存器（EMACC_D_PI_DIV）

偏移地址：0x50

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KPDIV[15:0]															
w															

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

位 31:16	KPDIV[15:0]：设置 D 轴 PID 模块比例项除法参数（PID PI module production part divisor parameter, the number of bits shifted by right） 该位域表示除法参数右移的位数。
位 15:0	KIDIV[15:0]：设置 D 轴 PID 模块积分项除法参数（PID PI module integration part divisor parameter, the number of bits shifted by right） 该位域表示除法参数右移的位数。

13.2.19 D 轴 PID 参数参考输入配置寄存器（EMACC_ID_REF）

偏移地址：0x54

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDREF[15:0]															
w															

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

位 31:16	IDREF[15:0]：设置 D 轴 PID 模块参考输入（PID module D channel reference input）
位 15:0	Res：保留 必须保持复位值。

13.2.20 Q 轴 PID 参数 KP/KI 配置寄存器（EMACC_Q_KPKI）

偏移地址：0x60

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KP[15:0]															
w															

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

位 31:16	KP[15:0]：设置 Q 轴 PID 模块的比例计算增益参数（PID gain parameter for proportion computation of Q axis）
位 15:0	KI[15:0]：设置 Q 轴 PID 模块的积分计算增益参数（PID gain parameter for integration computation of Q axis）

13.2.21 Q 轴 PID 积分上极限值配置寄存器（EMACC_Q_INTUPPER）

偏移地址：0x64

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：设置 Q 轴 PID 模块积分上限值（PID integration part upper limit of Q axis）

13.2.22 Q 轴 PID 积分下极限值配置寄存器（EMACC_Q_INTLOWER）

偏移地址：0x68

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：设置 Q 轴 PID 模块积分下极限值（PID integration part lower limit of Q axis）

13.2.23 Q 轴 PID 积分输出极限值配置寄存器（EMACC_Q_INTOUT）

偏移地址：0x6C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERLMT[15:0]															
w															

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

位 31:16	UPPERLMT[15:0]：设置 Q 轴 PID 模块输出上极限值（PID Group x PI module output data upper limit value of Q axis）
位 15:0	LOWERLMT[15:0]：设置 Q 轴 PID 模块输出下极限值（PID Group x PI module output data lower limit value of Q axis）

13.2.24 Q 轴 PID 参数右移位数配置寄存器（EMACC_Q_PI_DIV）

偏移地址：0x70

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KPDIV[15:0]															
w															

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

位 31:16	KPDIV[15:0]：设置 Q 轴 PID 模块比例项除法参数（PID PI module production part divisor parameter, the number of bits shifted by right） 该位域表示除法参数右移的位数。
位 15:0	KIDIV[15:0]：设置 Q 轴 PID 模块积分项除法参数（PID PI module integration part divisor parameter, the number of bits shifted by right） 该位域表示除法参数右移的位数。

13.2.25 Q 轴 PID 参数参考输入配置寄存器（EMACC_IQ_REF）

偏移地址：0x74

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
IQREF[15:0]															
w															

位 31:16	Res：保留 必须保持复位值。
位 15:0	IQREF[15:0]：设置 Q 轴 PID 模块参考输入（PID module Q channel reference input）

13.2.26 读取当前选定 PID 的 KP/KI 参数配置寄存器（EMACC_READ_KPKI）

偏移地址：0xC0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KP[15:0]															
r															

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

位 31:16	KP[15:0]：读取当前选定的 KP 增益配置（Current KP gain configuration） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。
位 15:0	KI[15:0]：读取当前选定的 KI 增益配置（Current KI gain configuration） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.27 读取当前选定 PID 的积分上极限值寄存器（EMACC_READ_INTUPPER）

偏移地址：0xC4

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：读取当前选定的积分项上极限值（Current integral upper limit value） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.28 读取当前选定 PID 积分下极限值寄存器（EMACC_READ_INTLOWER）

偏移地址：0xC8

复位值：0x0000 0000

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

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

位 31:0	LMT[31:0]：读取当前选定的积分下极限值（Current integral lower limit value） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.29 读取当前选定 PID 的积分输出极限值配置寄存器（EMACC_READ_INTOUT）

偏移地址：0xCC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPPERLMT[15:0]															
r															

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

位 31:16	UPPERLMT[15:0]：读取当前选定的上极限值（Current data-out upper limit value） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。
位 15:0	LOWERLMT[15:0]：读取当前选定的下极限值（Current data-out lower limit value） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.30 读取当前选定 PID 的参数右移位数寄存器（EMACC_READ_PI_DIV）

偏移地址：0xD0

复位值：0x0000 0000

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

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

位 31:16	KPDIV[31:16]：读取当前选定的 KP 参数右移位数（Current KP: the number of bits shifted by right） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。
位 15:0	KIDIV[15:0]：读取当前选定的 KI 参数右移位数（Current KI: the number of bits shifted by right） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.31 读取当前选定 PID 的参数 DQ 轴参考输入寄存器（EMACC_READ_IDQ_REF）

偏移地址：0xD4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IDREF[15:0]															
r															

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

位 31:16	IDREF[15:0]：读取当前选定的 D 轴参考值（D channel reference input） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。
位 15:0	IQREF[15:0]：读取当前选定的 Q 轴参考值（Q channel reference input） 注意：由 EMACC_CR.PARASEL[1:0]位来选定的当前读出的是哪个轴的 PID 参数。

13.2.32 三相驱动信号的 PWM 占空比 1 寄存器（EMACC_SVPWM_CNTPH1）

偏移地址：0xD8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNTPHB[15:0]															
r															

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

位 31:16	CNTPHB[15:0]：B 相的 PWM 占空比（PWM Duty cycle for phase B）
位 15:0	CNTPHA[15:0]：A 相的 PWM 占空比（PWM Duty cycle for phase A）

13.2.33 三相驱动信号的 PWM 占空比 2 寄存器（EMACC_SVPWM_CNTPH2）

偏移地址：0xDC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[15:0]															
rw															

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

位 31:16	CCR4[15:0]：设置 ADC 在 PWM 周期的中间点采样（Set the ADC to sample at the middle point of the PWM cycle）
位 15:0	CNTPHC[15:0]：C 相的 PWM 占空比（PWM Duty cycle for phase C）

13.2.34 单电阻模式触发相电流的采样点寄存器（EMACC_SVPWM_CNTSMP）

偏移地址：0xE0

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNTSMP2[15:0]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTSMP1[15:0]															
r															

位 31:16	CNTSMP2[15:0]：单电阻模式的第二个采样点。（The Second sampling point in the single-resistance Mode.） 以计时器计数方式表示。 注意：该项配置只适用于单电阻模式。
位 15:0	CNTSMP1[15:0]：单电阻模式的第一个采样点。（The first sampling point in the single-resistance Mode.） 以计时器计数方式表示。 注意：该项配置只适用于单电阻模式。

13.2.35 SVPWM 的各项其它输出寄存器（EMACC_SVPWM_OUTR）

偏移地址：0xE4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATE	Res												ACHS[2:0]		
r													r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWMNEW[1:0]		SAMPCUR2[3:0]				SAMPCUR1[3:0]				SECTOR[2:0]			SECTOR_FINAL[2:0]		
r		r				r				r			r		

位 31	ATE：ADC 采样触发沿（ADC group regular conversion trigger polarity edge） ● 0：上升沿 ● 1：下降沿 *注意：该项配置只适用于三电阻模式。*
位 30:19	Res：保留 必须保持复位值。
位 18:16	ACHS[2:0]：ADC 调度（ADC channel schedule） ● 01：AC ● 10：BC ● 其它值：AB *注意：该项配置只适用于三电阻模式。*
位 15:14	PWMNEW[1:0]：当前 PWM 周期类型（This value indicates the type of the current PWM period） ● 00：采样窗口无失真 ● 01：采样窗口 A 相失真 ● 10：采样窗口 B 相失真 ● 11：采样窗口 C 相失真 该值表示：单电阻模式中，采样窗口很窄，导致采样不了的相。
位 13:10	SAMPCUR2[3:0]：第二个电流采样点（Current sampled in the second sampling point） 注意：该项配置只适用于单电阻模式。
位 9:6	SAMPCUR1[3:0]：第一采样点采样电流（Current sampled in the first sampling point） 注意：该项配置只适用于单电阻模式。
位 5:3	SECTOR[2:0]：扇区中间值，值 0-5 代表扇区 1-6（Temp of Sector）
位 2:0	SECTOR_FINAL[2:0]：扇区最终值，值 0-5 代表扇区 1-6（Final of Sector）

13.2.36 SVPWM 控制寄存器（EMACC_SVPWM_CR）

偏移地址：0xE8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DTCNT[15:0]															
rw															

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

位 31:16	DTCNT[15:0]：死区时间（Dead time in number of TIM clock cycles）
位 15:3	Res：保留 必须保持复位值。
位 2	DSTEN：表示单电阻方案中是否必须执行采样窗口失真补偿算法（Enabling or disabling distortion for single shunt） ● 0：执行 ● 1：不执行 *注意：该项配置只适用于单电阻模式。*
位 1	DTTEST：死区时间补偿（Dead time compensation disable/enable） ● 0：不补偿 ● 1：补偿
位 0	MODE：工作模式选择（Work mode） ● 0：三电阻模式 ● 1：单电阻模式

13.2.37 PWM 驱动信号周期参数寄存器（EMACC_SVPWM_PERIOD）

偏移地址：0xEC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SQRT3[15:0]															
rw															

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

位 31:16	SQRT3[15:0]：SVPWM 算法中用到的$\sqrt{3}$常数（A constant utilized by PWM algorithm） 注意：为了方便进行整形计算，将常数放大之后再运算，然后再等比缩小。 SQRT3FACTOR：16384 x $\sqrt{3}$ x 2 SQRT3 =（PWM_PERIOD_CYCLES x SQRT3FACTOR）/16384
位 15:0	PWM_PERIOD_CYCLE[15:0]：PWM 的周期计数值（PWM period expressed in timer clock cycles unit） 注意： PWM_PERIOD_CYCLE = TIMER_CLK/Fpwm TIMER_CLK：产生 PWM 的定时器时钟，比如 48M，72M，96M Fpwm：电机驱动信号的频率，比如 10K，20K。

13.2.38 单电阻模式下相电流采样时间寄存器（EMACC_SVPWM_ITS）

偏移地址：0xF4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSAMPLE[15:0]															
rw															

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

位 31:16	TSAMPLE[15:0]：相电流采样时间（It is the sampling time express in numbers of TIM clocks） 注意：该参数只在单电阻模式下使用。
位 15:0	Res：保留 必须保持复位值。

13.2.39 相电流噪声时间寄存器（EMACC_SVPWM_TIME）

偏移地址：0xF8

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TAFTER[15:0]															
rw															

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

位 31:16	TAFTER[15:0]：上升时间与噪声时间的最大值与死区时间之和（The sum of dead time plus rise time express in number of TIM clocks）
位 15:0	TBEFORE[15:0]：采样时间（The value of sampling time expressed in numbers of TIM clocks）

13.2.40 单电阻模式下的极限参数寄存器（EMACC_SVPWM_MAXMIN）

偏移地址：0xFC

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAXTRTS[15:0]															
rw															

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

位 31:16	MAXTRTS[15:0]：两倍上升时间和两倍采样时间中的最大值（The maximum between twice of rise time express in number of TIM clocks and twice of sampling time express in numbers of TIM clocks） 注意：该参数只在单电阻模式下使用。
位 15:0	TMIN[15:0]：死区时间与上升时间及采样时间的和（The sum of dead time plus rise time plus sampling time express in numbers of TIM clocks） 注意：该参数只在单电阻模式下使用。

13.2.41 循环限制算法的 MMITABLE 数组的 SRAM 空间寄存器（EMACC_PID_RAM_SPACE）

偏移地址：0x100

复位值：0x0000 01FF

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

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

位 31:0	RAMDATA[31:0]：将循环限制算法的 MMITABLE 数组加载到 EMACC 的 SRAM 中去（MMITABLE data load to EMACC） 注意：将用于循环限制算法的 MMITABLE 数组加载到 EMACC 模块的 SRAM 中去。

13.2.42 EMACC TRACE 控制寄存器（EMACC_TRACE_CR）

偏移地址：0x200

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														HWST_EN	TX_START
														rw	rw

位 31:2	Res:保留 必须保持复位值。
位 1	HWST_EN：选择软件/硬件触发方式发起数据传输（Select a software / hardware trigger mode to initiate a data transfer） ● 0：软件触发方式发起数据传输 ● 1：硬件触发方式发起数据传输
位 0	TX_START：发起传输数据（Start to transmit a data） ● 0：不发起传输数据 ● 1：发起传输数据

13.2.43 EMACC_TRACE 配置寄存器（EMACC_TRACE_CFGR）

偏移地址：0x204

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									UTD4TXE	UTD3TXE	UTD2TXE	UTD1TXE	COSXTXE	SINXTXE	CNTPHCTXE
									rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTPHBTXE	CNTPHATXE	VBETXE	VALTXE	VDTXE	VQTXE	IDREFTXE	IQREFTXE	IDTXE	IQTXE	IBETXE	IALTXE	IBTXE	IATXE	ELATXE	SYNCODETXE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:23	Res：保留 必须保持复位值。

位 22	UTD4TXE：用户自定义数据 4 传输选择位（User Custom data 4 transfer selection bit） ● 0：不发送用户自定义数据 4 ● 1：发送用户自定义数据 4
位 21	UTD3TXE：用户自定义数据 3 传输选择位（User Custom data 3 transfer selection bit） ● 0：不发送用户自定义数据 3 ● 1：发送用户自定义数据 3
位 20	UTD2TXE：用户自定义数据 2 传输选择位（User Custom data 2 transfer selection bit） ● 0：不发送用户自定义数据 2 ● 1：发送用户自定义数据 2
位 19	UTD1TXE：用户自定义数据 1 传输选择位（User Custom data 1 transfer selection bit） ● 0：不发送用户自定义数据 1 ● 1：发送用户自定义数据 1
位 18	COSXTXE：CORDIC_OUT.COSX（余弦函数）计算值传输选择位（The COSX calculation value transfer selection bit for CORDIC_OUT） ● 0：不发送 CORDIC_OUT. COSX 结果 ● 1：发送 CORDIC_OUT. COSX 结果
位 17	SINXTXE：CORDIC_OUT.SINX（余弦函数）计算值传输选择位（The SINX calculation value transfer selection bit for CORDIC_OUT） ● 0：不发送 CORDIC_OUT. SINX 结果 ● 1：发送 CORDIC_OUT. SINX 结果
位 16	CNTPHCTXE：CNTPHC（C 相占空比计数值）传输选择位（The CNTPHC calculation value transfer selection bit for EMACC_SVPWM_CNTPH2） ● 0：不发送 EMACC_SVPWM_CNTPH2.CNTPHC 值 ● 1：发送 EMACC_SVPWM_CNTPH2.CNTPHC 值
位 15	CNTPHBTXE：CNTPHB（B 相占空比计数值）传输选择位（The CNTPHB calculation value transfer selection bit for EMACC_SVPWM_CNTPH1） ● 0：不发送 EMACC_SVPWM_CNTPH1.CNTPHB 值 ● 1：发送 EMACC_SVPWM_CNTPH1.CNTPHB 值
位 14	CNTPHATXE：CNTPHA（A 相占空比计数值）传输选择位（The CNTPHA calculation value transfer selection bit for EMACC_SVPWM_CNTPH1） ● 0：不发送 EMACC_SVPWM_CNTPH1.CNTPHA 值 ● 1：发送 EMACC_SVPWM_CNTPH1.CNTPHA 值
位 13	VBETXE：REVPARK_OUT.VBETA（Vbeta）传输选择位（The VBETA calculation value transfer selection bit for REVPARK_OUT） ● 0：不发送 Vbeta ● 1：发送 Vbeta
位 12	VALTXE：REVPARK_OUT.VALPHA（Valpha）传输选择位（The VALPHA calculation value transfer selection bit for REVPARK_OUT） ● 0：不发送 Valpha ● 1：发送 Valpha

位 11	VDTXE：PID_OUT.VD（Vd）传输选择位（The VD calculation value transfer selection bit for PID_OUT） ● 0：不发送 Vd ● 1：发送 Vd
位 10	VQTXE：PID_OUT.VQ（Vq）传输选择位（The VQ calculation value transfer selection bit for PID_OUT） ● 0：不发送 Vq ● 1：发送 Vq
位 9	IDREFTXE：IDQ_REF.IDREF（Id_reference）传输选择位（The IDREF calculation value transfer selection bit for IDQ_REF） ● 0：不发送 Id_ref ● 1：发送 Id_ref
位 8	IQREFTXE：IDQ_REF.IQREF（Iq_reference）传输选择位（The IQREF calculation value transfer selection bit for IDQ_REF） ● 0：不发送 Iq_ref ● 1：发送 Iq_ref
位 7	IDTXE：PARK_OUT.ID （Id）传输选择位（The ID calculation value transfer selection bit for PARK_OUT） ● 0：不发送 Id ● 1：发送 Id
位 6	IQTXE：PARK_OUT.IQ （Iq）传输选择位（The IQ calculation value transfer selection bit for PARK_OUT） ● 0：不发送 Iq ● 1：发送 Iq
位 5	IBETXE：CLARK_OUT.IBET（Ibeta）传输选择位（The IBET calculation value transfer selection bit for CLARK_LDR） ● 0：不发送 Ibeta ● 1：发送 Ibeta
位 4	IALTXE：CLARK_OUT.IALPHA（alpha）传输选择位（The IALPHA calculation value transfer selection bit for CLARK_LDR） ● 0：不发送 Ialpha ● 1：发送 Ialpha
位 3	IBTXE：CLARK_LDR.IB（Ib）传输选择位（The IB calculation value transfer selection bit for CLARK_LDR） ● 0：不发送 Ib ● 1：发送 Ib
位 2	IATXE：CLARK_LDR.IA（Ia）传输选择位（The IA calculation value transfer selection bit for CLARK_LDR） ● 0：不发送 Ia ● 1：发送 Ia
位 1	ELATXE：CORDIC_ANG_LDR.CORDIC_ANG_IN （hElAngle）电角度传输选择位（The hElAngle calculation value transfer selection bit for CORDIC_ANG_LDR） ● 0：不发送 hElAngle ● 1：发送 hElAngle
位 0	SYNCODETXE：EMACC_SYNCODE.DATA（数据传输同步头）传输选择位（The DATA calculation value transfer selection bit for EMACC_SYNCODE）

	• 0：不发送同步头 • 1：发送同步头

13.2.44 用户自定义传输数据 1 寄存器（EMACC_TRACE_USRTXD1）

偏移地址：0x208

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRTXD1[15:0]															
rw															

位 15:0	USRTXD1[15:0]：用户自定义发送的数据 1。（User Custom data 1） 允许用户自定义发送 16 位数据 1。

13.2.45 用户自定义传输数据 2 寄存器（EMACC_TRACE_USRTXD2）

偏移地址：0x20C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRTXD2[15:0]															
rw															

位 15:0	USRTXD2[15:0]：用户自定义发送的数据 2。（User Custom data 2） 允许用户自定义发送 16 位数据 2。

13.2.46 用户自定义传输数据 3 寄存器（EMACC_TRACE_USRTXD3）

偏移地址：0x210

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRTXD3[15:0]															
rw															

位 15:0	USRTXD3[15:0]：用户自定义发送的数据 3。（User Custom data 3） 允许用户自定义发送 16 位数据 3。

13.2.47 用户自定义传输数据 4 寄存器（EMACC_TRACE_USRTXD4）

偏移地址：0x214

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRTXD4[15:0]															
rw															

位 15:0	USRTXD4[15:0]：用户自定义发送的数据 4。（User Custom data 4） 允许用户自定义发送 16 位数据 4。

13.2.48 传输数据同步头寄存器（EMACC_TRACE_SYNCODE）

偏移地址：0x218

复位值：0x0000

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

位 15:0	TRACE_SYNCODE[15:0]：传输数据同步头（Transfer the data synchronization header） 用于发送数据的同步帧头。

14 除法开方运算单元（DVSQ）

14.1 DVSQ 主要特性

除法和开方（Division and square root，DVSQ）计算单元支持以下特性：

- 支持 32 位带符号数（SDIV）和无符号数除法（UDIV），支持 32 位开方运算。
  - 在同一时刻，DVSQ 计算单元不能同时支持除法和开方运算，只能两者选其一执行。
  - 32 位有符号/无符号整数除法运算结束后，可同时获取商和余数并更新到相应的寄存器。
  - 除法运算支持 MOD 操作。
- 无符号开方运算，可以通过软件选择高精度开方运算。
- 流水线设计，每个时钟完成 2 位运算。
- 运算时间根据运算数据不同而改变。
- 支持除零中断和溢出中断。

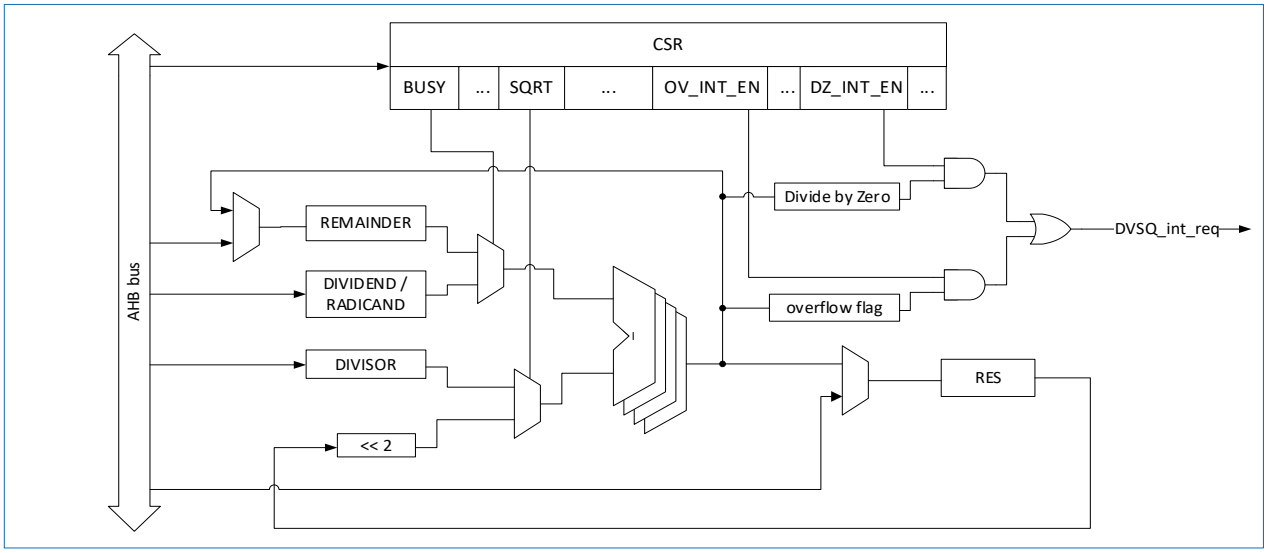

图 14-1 DVSQ 结构图

14.2 除法操作流程

除法操作

结果寄存器 DVSQ_RES 和余数寄存器 DVSQ_REMAINDER 中保持的值始终为补码格式。如果执行无符号除法运算，则 DVSQ_RES 寄存器和 DVSQ_REMAINDER 寄存器中的值都为正数。如果执行带符号数除法，则 DVSQ_RES 寄存器和 DVSQ_REMAINDER 寄存器的符号位由输入的操作数决定：

- 如果被除数和除数的符号位不同，则商为负数；否则商为正数。
- 余数的符号位始终和除数的符号位相同。

操作流程

可以通过快速启动和非快速启动两种方式启动除法运算。除法运算的操作流程示意图为：

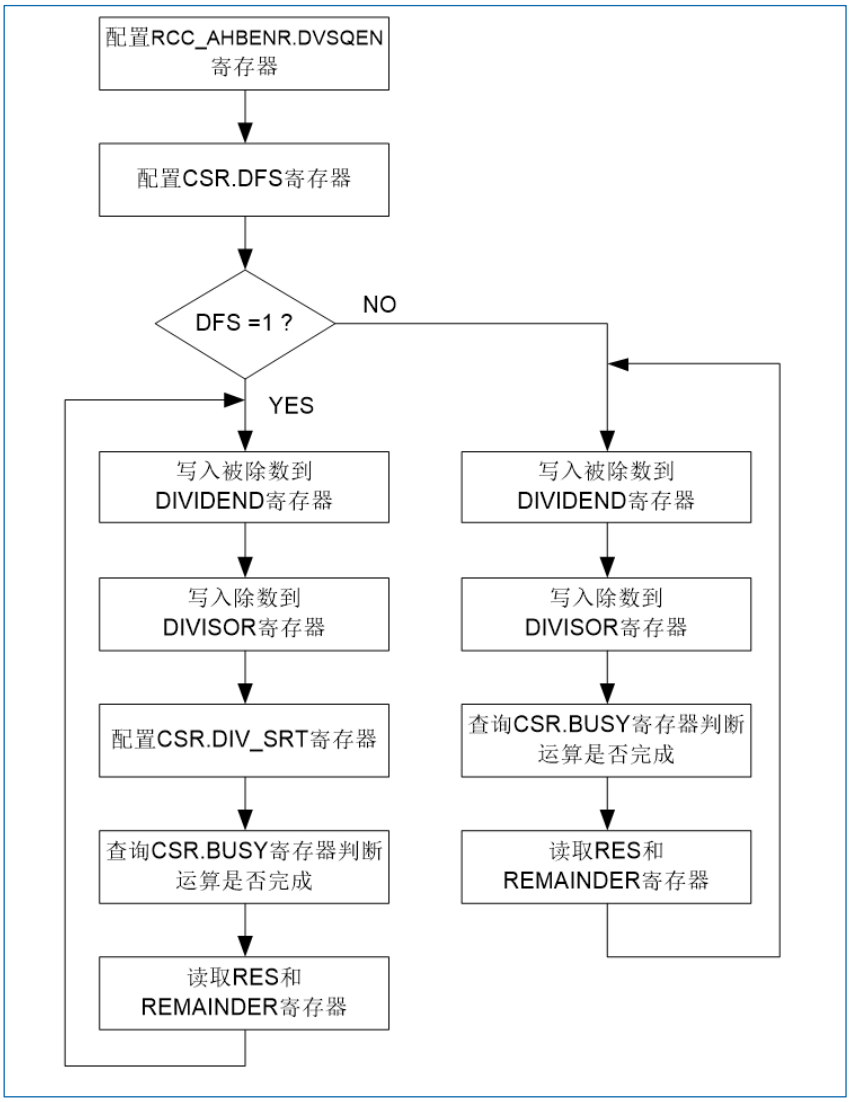

图 14-2 除法运算流程示意图

14.3 除法运行时间

DVSQ 加速器通过被除数的数据决定运算时间，以尽快得到运算结果。运算时间如下表所示，其中运算时间定义为从运算开始到得到运算结果的时钟周期数，也就是 DVSQ_CSR.BUSY 为高的持续时间。

表 14-1 除法运算时间对应表

被除数的绝对值（有效位的位置）	运算时间[周期数]
（01、1x）xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	17
00（01、1x）_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	16
0000_（01、1x）xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	15
0000_00（01、1x）_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	14
0000_0000_（01、1x）xx_xxxx__xxxx_xxxx_xxxx_xxxx	13
0000_0000_00（01、1x）_xxxx__xxxx_xxxx_xxxx_xxxx	12
0000_0000_0000_（01、1x）xx__xxxx_xxxx_xxxx_xxxx	11
0000_0000_0000_00（01、1x）__xxxx_xxxx_xxxx_xxxx	10

被除数的绝对值（有效位的位置）	运算时间[周期数]
0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	8
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	7
0000_0000_0000_0000_0000_00（01、1x）_xxxx_xxxx	6
0000_0000_0000_0000_0000_0000_（01、1x）xx_xxxx	5
0000_0000_0000_0000_0000_0000_00（01、1x）_xxxx	4
0000_0000_0000_0000_0000_0000_0000_（01、1x）xx	3
0000_0000_0000_0000_0000_0000_0000_00（01、1x）	2
0000_0000_0000_0000_0000_0000_0000_0000	1

14.4 开方操作描述

DVSQ 加速器只能进行无符号数开方，因此进行开方运算的时候 DVSQ_RADICAND 和 DVSQ_RES 中的值都是无符号数。

操作流程

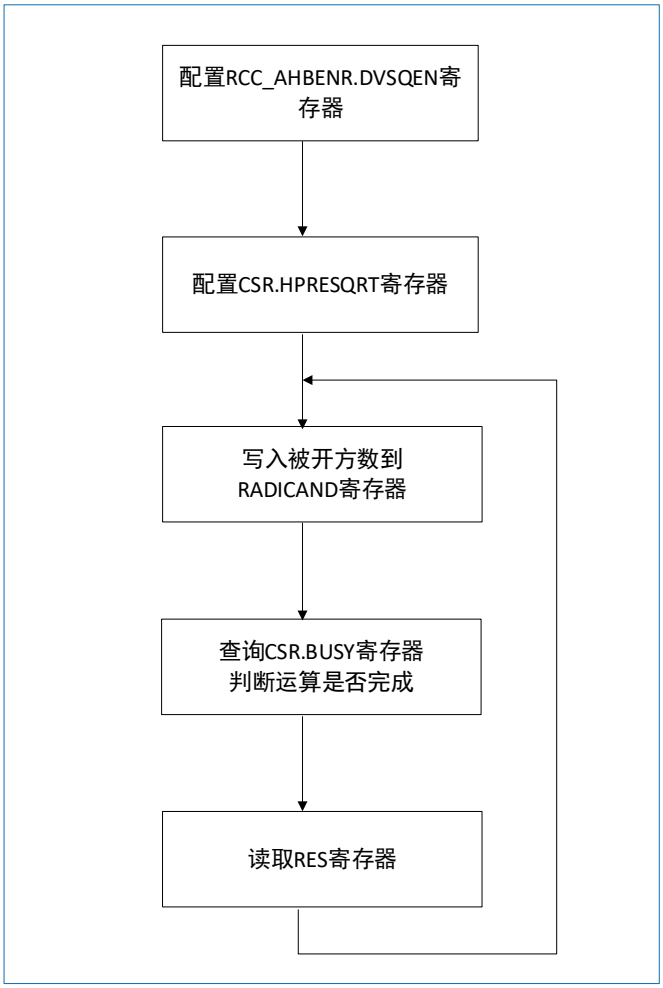

图 14-3 开方运算流程图

14.5 开方运行时间

DVSQ 加速器根据被开方数决定运算时间，同时 DVSQ_CSR.HPRESQRT 也会影响开方运算时间。具体时间如下表，其中运算时间定义为从运算开始到得到运算结果的时钟周期数，也就是 DVSQ_CSR.BUSY 为高的持续时间。

当 DVSQ_CSR.HPRESQRT 为 0 时：

表 14-2 开方运行时间（DVSQ_CSR. HPRESQRT=0）

被开方数（有效位的位置）	运算时间[周期数]
（01、1x）xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	17
00（01、1x）_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	16
0000_（01、1x）xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	15
0000_00（01、1x）_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	14
0000_0000_（01、1x）xx_xxxx__xxxx_xxxx_xxxx_xxxx	13
0000_0000_00（01、1x）_xxxx__xxxx_xxxx_xxxx_xxxx	12
0000_0000_0000_（01、1x）xx__xxxx_xxxx_xxxx_xxxx	11
0000_0000_0000_00（01、1x）__xxxx_xxxx_xxxx_xxxx	10
0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	8
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	7
0000_0000_0000_0000_0000_00（01、1x）_xxxx_xxxx	6
0000_0000_0000_0000_0000_0000_（01、1x）xx_xxxx	5
0000_0000_0000_0000_0000_0000_00（01、1x）_xxxx	4
0000_0000_0000_0000_0000_0000_0000_（01、1x）xx	3
0000_0000_0000_0000_0000_0000_0000_00（01、1x）	2
0000_0000_0000_0000_0000_0000_0000_0000	1

当 DVSQ_CSR.HPRESQRT 为 1 时：

表 14-3 开方运行时间（DVSQ_CSR. HPRESQRT=1）

被开方数（有效位的位置）	运算时间[周期数]
（01、1x）xx_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	33
00（01、1x）_xxxx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	32
0000_（01、1x）xx_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	31
0000_00（01、1x）_xxxx_xxxx__xxxx_xxxx_xxxx_xxxx	30
0000_0000_（01、1x）xx_xxxx__xxxx_xxxx_xxxx_xxxx	29

被开方数（有效位的位置）	运算时间[周期数]
0000_0000_00（01、1x）_xxxx__xxxx_xxxx_xxxx_xxxx	28
0000_0000_0000_（01、1x）xx__xxxx_xxxx_xxxx_xxxx	27
0000_0000_0000_00（01、1x）__xxxx_xxxx_xxxx_xxxx	26
0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx_xxxx	25
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	24
0000_0000_0000_0000_0000_（01、1x）xx_xxxx_xxxx	23
0000_0000_0000_0000_0000_00（01、1x）_xxxx_xxxx	22
0000_0000_0000_0000_0000_0000_（01、1x）xx_xxxx	21
0000_0000_0000_0000_0000_0000_00（01、1x）_xxxx	20
0000_0000_0000_0000_0000_0000_0000_（01、1x）xx	19
0000_0000_0000_0000_0000_0000_0000_00（01、1x）	18
0000_0000_0000_0000_0000_0000_0000_0000	1

14.6 中断

DVSQ 加速器内部的两个中断源共用芯片中的同一个中断号。当系统检测到中断请求后，需要通过读 DVSQ_CSR 寄存器来判断是除零中断还是溢出中断。在同一时刻，除零中断和溢出中断不能同时发生，只可能是其一。

- 带符号数除法溢出中断：
  - 可以通过配置 DVSQ_CSR.OV_INT_EN 使能或关闭。
  - 当带符号数除法的被除数为 0x8000 0000，除数为 0xFFFF FFFF 时，中断请求会被硬件置位。
  - 本中断情况可以通过软件或硬件清除：
    - 软件配置 DVSQ_CSR.OV_FLAG 为 0。
    - 开始下一次除法或开方运算。
- 除零中断：
  - 可以通过配置 DVSQ_CSR.DZ_INT_EN 使能或关闭。
  - 当除数为 0 时，中断请求会被硬件置位。
  - 本中断请求可以通过软件或硬件清除：
    - 软件配置 DVSQ_CSR.DZ_FLAG 为 0。
    - 开始下一次除法或开方运算。

14.7 注意事项

除法操作

因为 DVSQ_DIVIDEND 寄存器和 DVSQ_DIVISOR 寄存器的值在运算过程中不会被硬件改变，所以软件通过字节或半字写这两个寄存器的时候需要小心。比如软件字节写 DIVIDEND[7:0]后，DVSQ_DIVIDEND 寄存器中的高 24 位为上一次除法运算写入的值，低 8 位为新写入的值。

当除法配置为快速启动后，无论是以字节、半字还是字写 DVSQ_DIVISOR 寄存器都会使除法运算开

始。

开方运算

无论是字节、半字还是字的方式写 DVSQ_RADICAND 寄存器都会使开方运算开始。

结果读取

如果在 DVSQ 还没有完成运算的时候访问 DVSQ_RES 和 DVSQ_REMAINDER 寄存器，将会使总线处于等待状态，在等待状态中也不能响应中断。软件可以通过轮询 DVSQ_CSR.BUSY 的状态来判断 DVSQ_RES 和 DVSQ_REMAINDER 的值是否已经就绪。

如果在 DVSQ 还没有完成运算的时候，访问 DVSQ_DIVIDEND/DVSQ_DIVISOR/DVSQ_RADICAND 寄存器也会使总线处于等待状态。

14.8 DVSQ 寄存器

基地址：0x4003 0000

空间大小：0x400

14.8.1 被除数寄存器（DVSQ_DIVIDEND）

偏移地址：0x00

复位值：0x0000 0000

软件配置除法运算所需的被除数值。如果在 DVSQ 加速器处于工作态时，对 DVSQ_DIVIDEND 寄存器进行读写访问，则总线将处于等待状态直到 DVSQ 加速器的操作完成。DVSQ_DIVIDEND 寄存器只能被软件写访问更新，硬件运算不会更新 DVSQ_DIVIDEND 寄存器。

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

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

位 31:0	DIVIDEND[31:0]：被除数寄存器（Dividend） ● 读操作：读出寄存器里存储的值。 ● 写操作：更新除法运算的被除数。

14.8.2 除数寄存器（DVSQ_DIVISOR）

偏移地址：0x04

复位值：0x0000 0000

软件配置除法运算所需的除数值。当 DVSQ_CSR.DFS=0 时，如果软件写 DVSQ_DIVISOR 寄存器会立即使能除法运算。如果在 DVSQ 加速器处于工作态时，对 DVSQ_DIVISOR 寄存器进行读写访问，则总线将处于等待状态直到 DVSQ 加速器的操作完成。DVSQ_DIVISOR 寄存器只能被软件写访问更新，硬件运算不会更新 DVSQ_DIVISOR 寄存器。

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

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

位 31:0	DIVISOR[31:0]：被除数的值（Divisor）

14.8.3 控制和状态寄存器（DVSQ_CSR）

偏移地址：0x08

复位值：0x0000 0000

本寄存器控制除法和开方运算工作，并返回 DVSQ 加速器的工作状态。DVSQ_CSR 寄存器的值会被 DVSQ 加速器硬件更新。软件可以轮询 DVSQ_CSR 寄存器来判断除法和开方运算是否已经完成。如果在 DVSQ 加速器处于工作态的时候对 DVSQ_CSR 寄存器进行写操作，则总线将处于等待状态直到 DVSQ 加速器的操作完成。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								HPRESQRT	OV_FLAG	OV_INT_EN	DZ_FLAG	DZ_INT_EN	DFS	UNSIGN_DIV	DIV_SRT
								rw	rw	rw	rw	rw	rw	rw	rw

位 31	BUSY：DVSQ 加速器工作状态标志（Busy bit） ● 0：DVSQ 加速器处于空闲状态。 ● 1：DVSQ 加速器处于工作状态，表示有除法或开方运算正在工作。 位[31]BUSY 信号与位[30]DIV 和位[29]SQRT 结合起来可以标识 DVSQ 加速器的工作状态，具体当[BUSY|DIV|SQRT]为不同值时表示的含义为： ● 000：标识 DVSQ 加速器处于空闲态，并且还没有执行过除法或开方操作。 ● 001：标识 DVSQ 加速器处于空闲态，并且 DVSQ 加速器完成的上一个运算为开方运算。 ● 010：标识 DVSQ 加速器处于空闲态，并且 DVSQ 加速器完成的上一个运算为除法运算。 ● 101：标识 DVSQ 加速器处于工作态，并且正在进行的是开方运算。 ● 110：标识 DVSQ 加速器处于工作态，并且正在进行的是除法运算。 ● 其他值：没有定义。
位 30	DIV：除法运算标志（Divide operation flag） ● 0：表示 DVSQ 加速器进行的上一个运算或者当前运算不是除法运算。 ● 1：表示 DVSQ 加速器进行的上一个运算或者当前运算为除法运算。
位 29	SQRT：开方运算标志（Square-root operation flag） ● 0：表示 DVSQ 加速器进行的上一个运算或者当前运算不是开方运算。 ● 1：表示 DVSQ 加速器进行的上一个运算或者当前运算为开方运算。
位 28:8	Res：保留 必须保持复位值。
位 7	HPRESQRT：高精度开方运算选择寄存器（High precision square-root operation selection） ● 写操作： ○ 0：DVSQ 执行普通精度开方运算。 ○ 1：DVSQ 执行高精度开方运算。 ● 读操作：读出寄存器里存储的值。 *说明：见本表格说明部分。*
位 6	OV_FLAG：带符号数除法溢出标志（Sign divide overflow flag）

	● 写操作： ○ 0：清除 OV_FLAG 标志。 ○ 1：无效操作。 ● 读操作： ○ 0：上一次除法运算没有发生溢出。 ○ 1：上一次除法运算溢出。 在进行带符号数除法的时候，如果被除数为 0x8000 0000，同时除数为 0xFFFF FFFF，则结果超出了 32 位补码的表示范围。在这种情况下，DVSQ 加速器置位 OV_FLAG，并且返回的商为 0x8000 0000，余数为 0x0000 0000。 OV_FLAG 可以被软件清除，如果发生带符号数除法溢出，OV_FLAG 一直保持为高，直到被软件清除，或者 DVSQ 加速器开始下一次除法或开方运算。
位 5	OV_INT_EN：带符号数除法溢出中断使能（Sign divide overflow interrupt enable） ● 写操作： ○ 0：关闭带符号数除法溢出中断。 ○ 1：使能带符号数除法溢出中断。 ● 读操作：读出寄存器里存储的值。
位 4	DZ_FLAG：除数为 0 标志（Zero divisor flag） ● 写操作： ○ 0：清除 DZ_FLAG 标志。 ○ 1：无效操作 ● 读操作： ○ 0：上一次运算的除数不为 0。 ○ 1：上一次运算的除数为 0。 无论是带符号数除法还是不带符号数除法，当除数为‘0’时，DVSQ 加速器都会把 DZ_FLAG 置位为 1、并且返回的商和余数都为 0x0000 0000。 DZ_FLAG 可以被软件清除，如果发生除数为‘0’的情况，DZ_FLAG 保持为高，直到被软件清除，或者 DVSQ 加速器开始下一次除法或开方运算。
位 3	DZ_INT_EN：除数为 0 中断使能（Zero divisor interrupt enable） ● 写操作： ○ 0：关闭除数为 0 中断。 ○ 1：使能除数为 0 中断。 ● 读操作：读出寄存器里存储的值。
位 2	DFS：快速启动除法运算关闭（Divide operation fast start disable） ● 写操作： ○ 0：使能快速启动除法运算。 ○ 1：关闭快速启动除法运算。 ● 读操作：读出寄存器里存储的值。 DVSQ 加速器支持两种方式启动除法运算。默认方式为‘快速启动’，如果对 DIVISOR 进行写操作，就立即启动除法运算。另一种启动方式为：软件置位 DVSQ_CSR.DIV_SRT 寄存器后才开始除法运算。由 DFS 位选择使用哪一种启动方式。
位 1	UNSIGN_DIV：无符号数除法（Unsign divide operation enable） ● 写操作： ○ 0：执行带符号数除法。 ○ 1：执行无符号数除法。

	● 读操作：读出寄存器里存储的值。
位 0	DIV_SRT：开始除法运算（Divide operation start） ● 写操作： ○ 0：无效操作 ○ 1：如果 DVSQ_CSR.DFS=1，则开始除法运算；否则不进行任何下一步动作。 ● 读操作：一直返回为 0。

说明：

软件可以使用 Q notation 来增加开方运算的精度。

如果使用 Q notation 来进行开方运算，无符号 Qm.n 记号法需要 m+n 位作为被开方数（m+n=32），产生的结果为 Q（m/2）.（n/2）格式。这占用 n 位被开方数作为小数位（n<31），因此如果软件需要高精度开方结果的时候，将比较大的影响被开方数的表示范围。

DVSQ 加速器提供了一种同时满足高精度开方和被开方数表示范围的方法，当 HPRESQRT 被置位为 1 时，DVSQ 加速器内部附加 32 位 ‘0’ 到被开方数寄存器后面，即：DVSQ_RADICAND（[RADICAND.0x0000 0000]）。然后把这个新的数作为被开方数带入开方运算单元进行运算。这时被开方数为 Q32.32，开方结果为 Q16.16。这样被开方数的结果不会受到影响，并且同时达到高精度开方的要求。如果 HPRESQRT 被设置为 ‘0’，则被开方数的小数位由软件决定，并且满足 m+n=32。

14.8.4 被开方数寄存器（DVSQ_RADICAND）

偏移地址：0x0C

复位值：0x0000 0000

软件通过写 DVSQ_RADICAND 寄存器配置开方运算的被开方数。如果在 DVSQ 加速器处于工作态时，对 DVSQ_RADICAND 寄存器进行读写访问，则总线将处于等待状态直到 DVSQ 加速器的操作完成。DVSQ_RADICAND 寄存器只能被软件写访问更新，硬件运算不会更新 DVSQ_RADICAND 寄存器。

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

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

位 31:0	RADICAND[31:0]：被开方数（Radicand）

14.8.5 结果寄存器（DVSQ_RES）

偏移地址：0x10

复位值：0x0000 0000

DVSQ_RES 寄存器存储除法运算的商或开方运算的平方根结果。在 DVSQ 加速器完成除法或开方运算后会自动更新 DVSQ_RES 寄存器。如果在 DVSQ 加速器处于工作态时，对 DVSQ_RES 寄存器进行读写访问，则总线将处于等待状态直到 DVSQ 加速器的操作完成。在 DVSQ 寄存器进行运算的时候有可能被系统中断服务程序打断，软件需要保存 DVSQ 加速器的上下文。因此 DVSQ_RES 寄存器和 DVSQ_REMAINDER 寄存器可以被软件写操作更新。

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

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

位 31:0	RESULT[31:0]：结果寄存器（Result） ● 写操作：更新结果寄存器中的值。 ● 读操作：读出寄存器里存储的值。

14.8.6 余数寄存器（DVSQ_REMAINDER）

偏移地址：0x14

复位值：0x0000 0000

DVSQ_REMAINDER 寄存器保存除法运算的余数结果。DVSQ 加速器进行除法运算或开方运算的时候都会自动更新 DVSQ_REMAINDER 寄存器。如果在 DVSQ 加速器处于工作态时，对 DVSQ_REMAINDER 寄存器进行读写访问，则总线将处于等待状态直到 DVSQ 加速器的操作完成。

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

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

位 31:0	REMAINDER[31:0]：余数寄存器（Remainder） ● 写操作：更新结果寄存器中的值。 ● 读操作：读出寄存器里存储的值。

15 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、嵌入死区时间的互补 PWM 等）。

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

高级控制（TIM1）和其他定时器完全独立，不共享任何资源。TIM1 和内部其他定时器可以被同步连在一起工作，请参考章节“16.2.15 定时器同步”。

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

表 15-1 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	$f_{TIMxCLK}$=48MHz	-	20.8	-	ns
f_{EXT}	定时器的 CH1 至 CH3，外部输入的时钟频率	-	-	$f_{TIMxCLK}/2$	-	MHz
		$f_{TIMxCLK}$=48MHz	-	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}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48MHz	-	1365	-	μs

15.1 TIM1 主要特征

TIM1 定时器的功能包括：

- 16 位向上、向下、向上/向下自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为 1~65536 的任意数值
- 多达 3 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
- 3 路额外的内部输出比较通道
- PWM 前后死区时间可编程的互补输出
- 使用外部信号控制定时器 TIM1 同步
- TIM1 与其他内部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
- 更新事件的移位功能
- PWM 输出下的简易的数据搬移功能
- 如下事件发生时产生中断/DMA：

- 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
- 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
- 输入捕获
- 输出比较
- 刹车信号输入

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

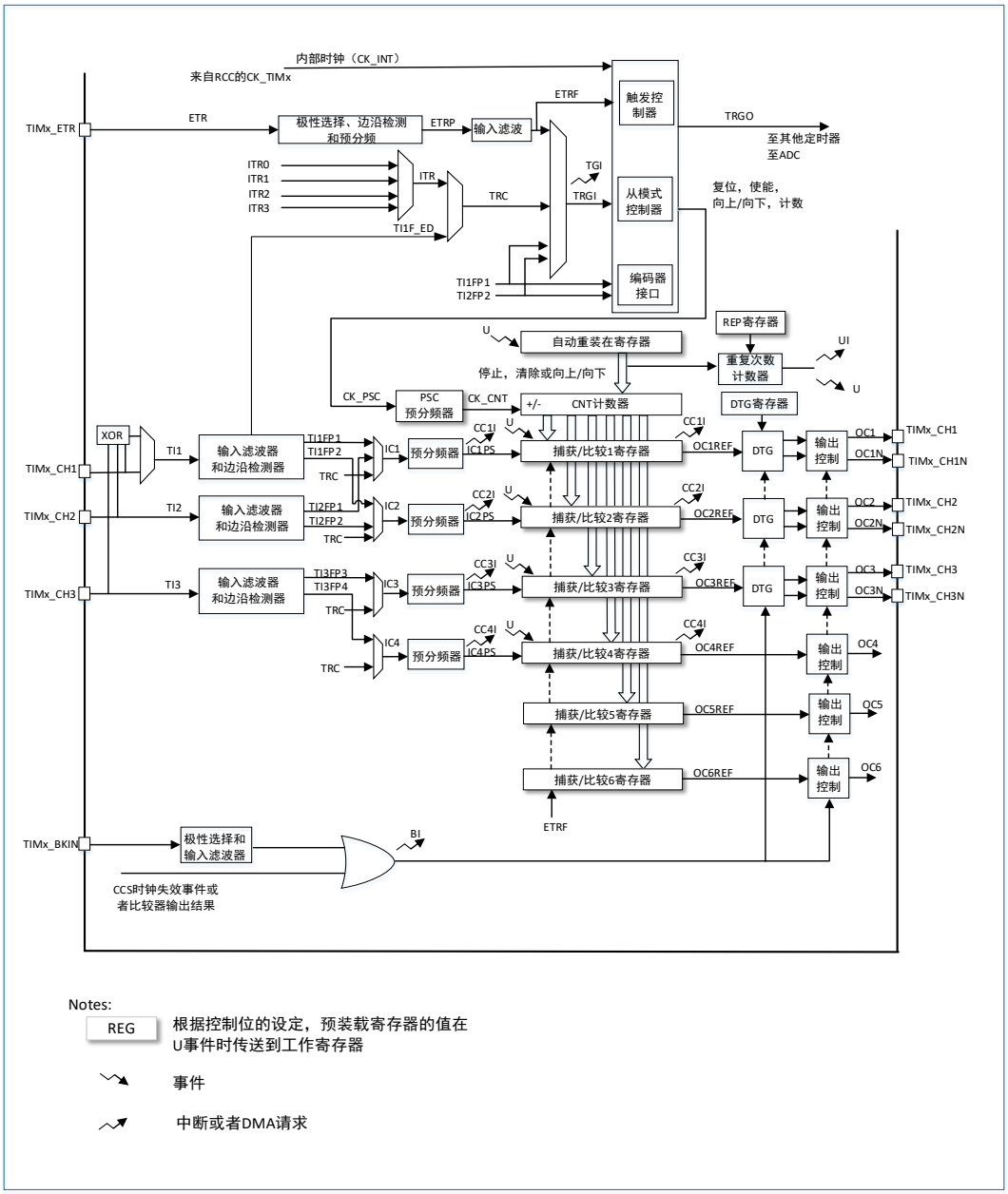

图 15-1 高级控制定时器框图

15.2 TIM1 功能描述

15.2.1 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“15.2.19 TIM1 定时器和外部触发的同步”控制器的从模式描述）。

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 1~65536 之间的任意值分频。它是基于一个 16 位寄存器（TIM1_PSC）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 15-2 和图 15-3 给出了在预分频器运行时，更改计数器参数的例子。

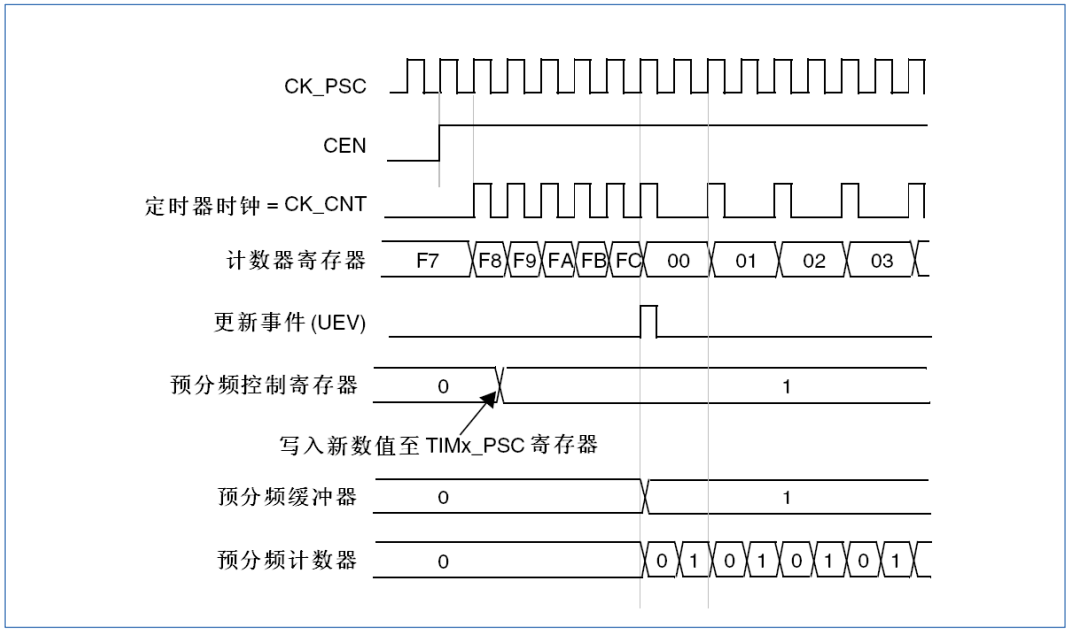

图 15-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

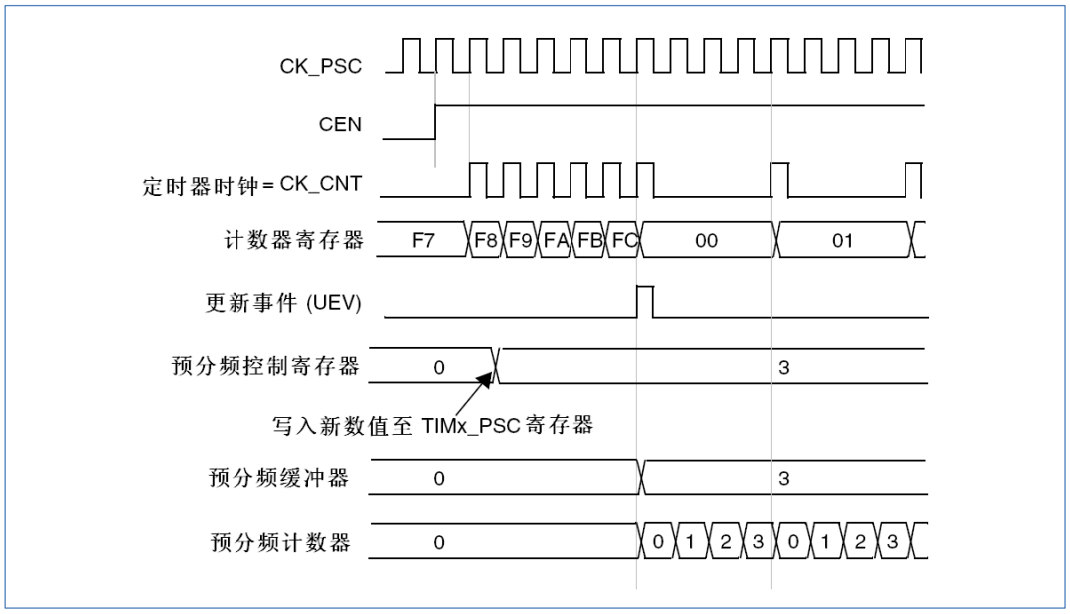

图 15-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

15.2.2 计数器模式

15.2.2.1 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都产生更新事件。

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 0（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（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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

图 15-4 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。

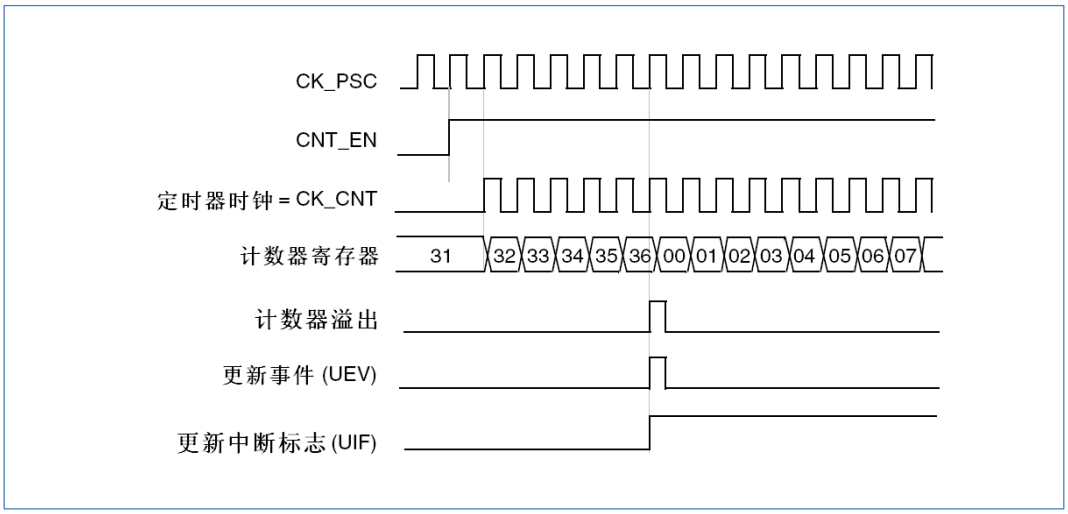

图 15-4 计数器时序图，内部时钟分频因子为 1

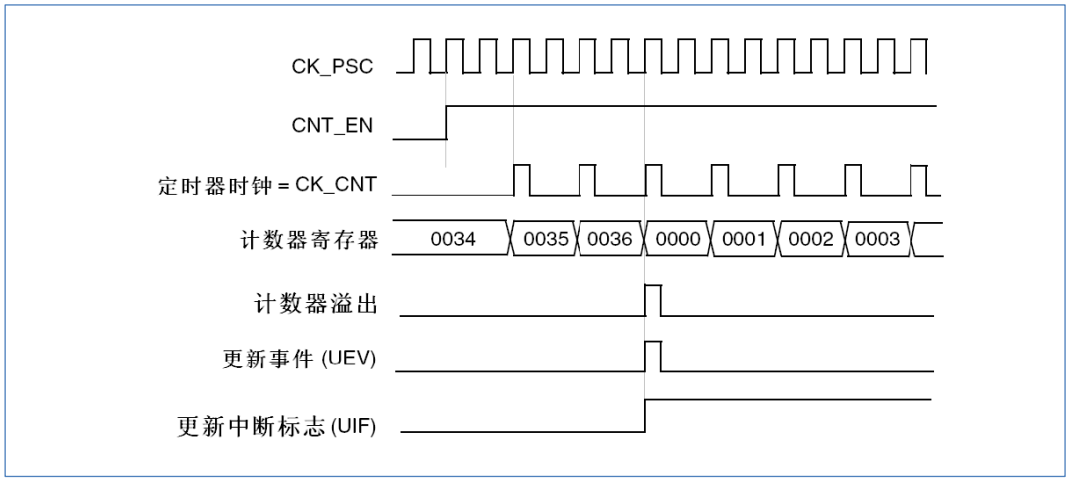

图 15-5 计数器时序图，内部时钟分频因子为 2

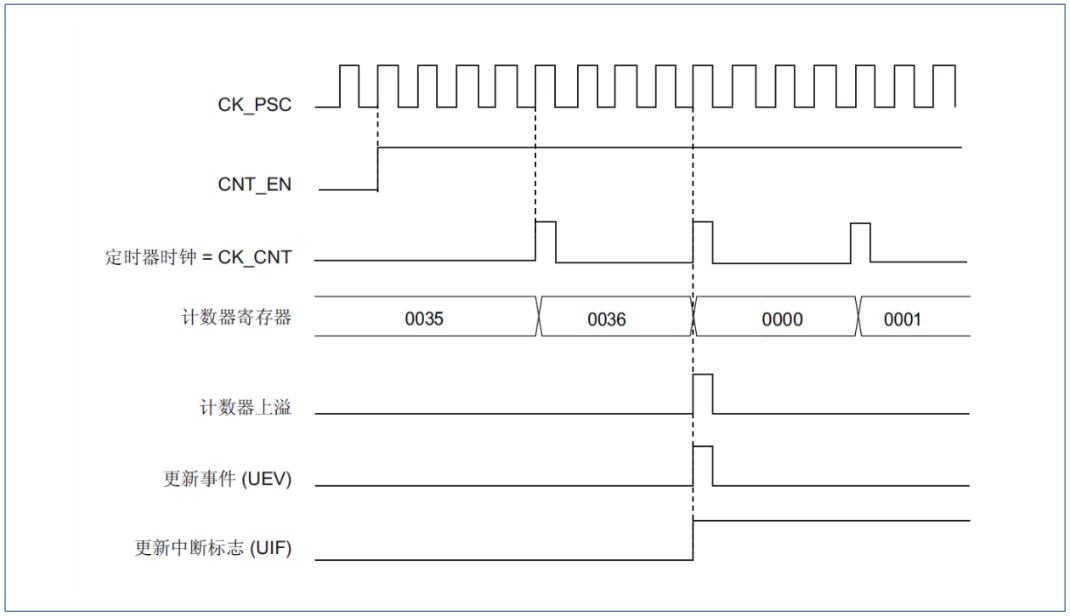

图 15-6 计数器时序图，内部时钟分频因子为 4

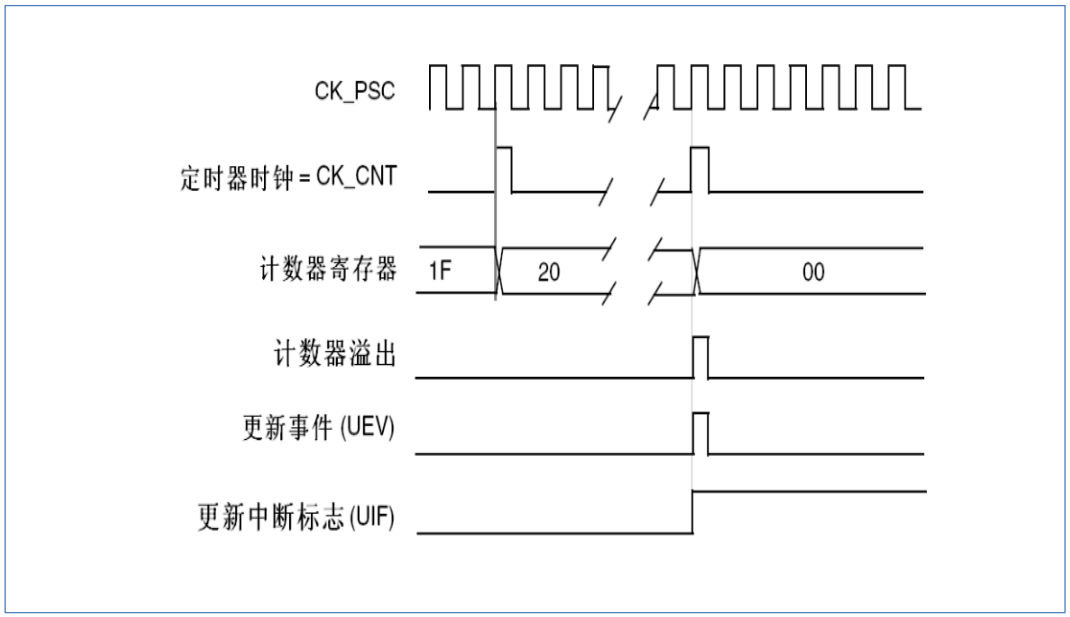

图 15-7 计数器时序图，内部时钟分频因子为 N

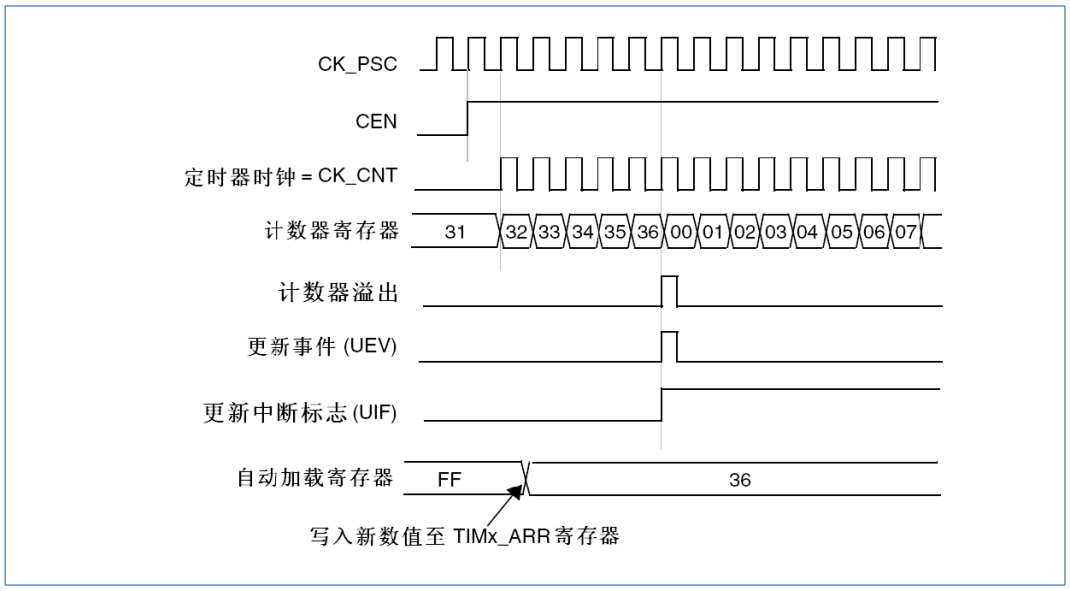

图 15-8 计数器时序图，当 ARPE=0 时的更新事件（TIM1_ARR 没有预装入）

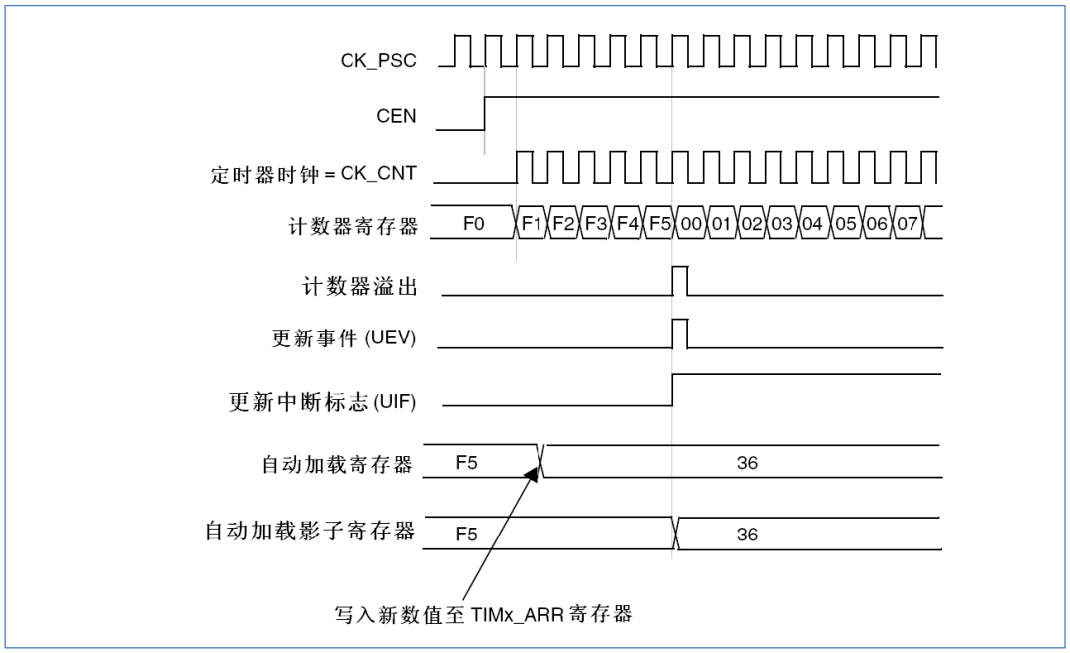

图 15-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM1_ARR）

15.2.2.2 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都产生更新事件。

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（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

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

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

以下是一些当 TIM1_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

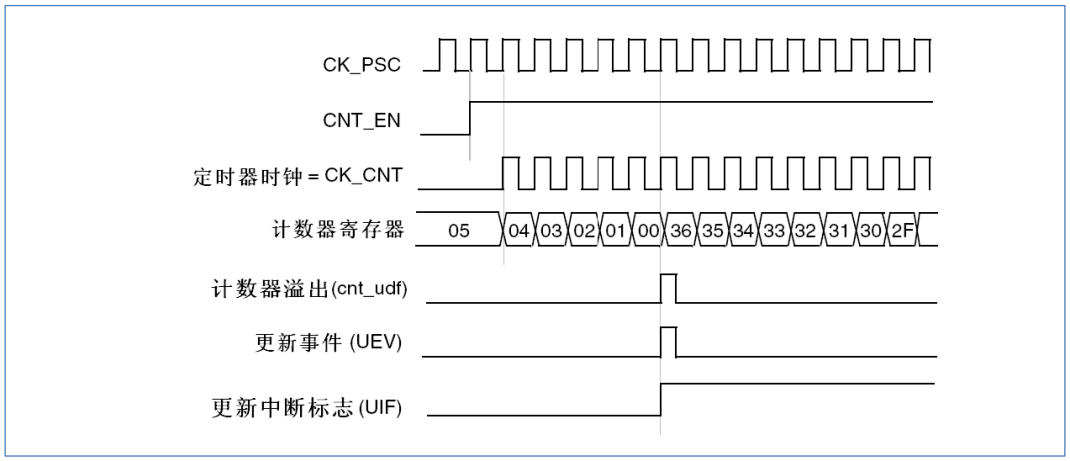

图 15-10 计数器时序图，内部时钟分频因子为 1

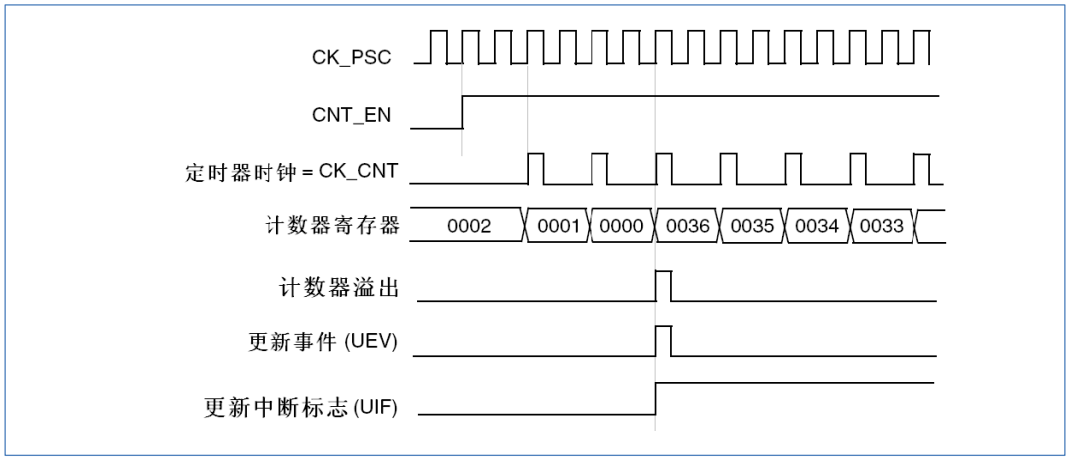

图 15-11 计数器时序图，内部时钟分频因子为 2

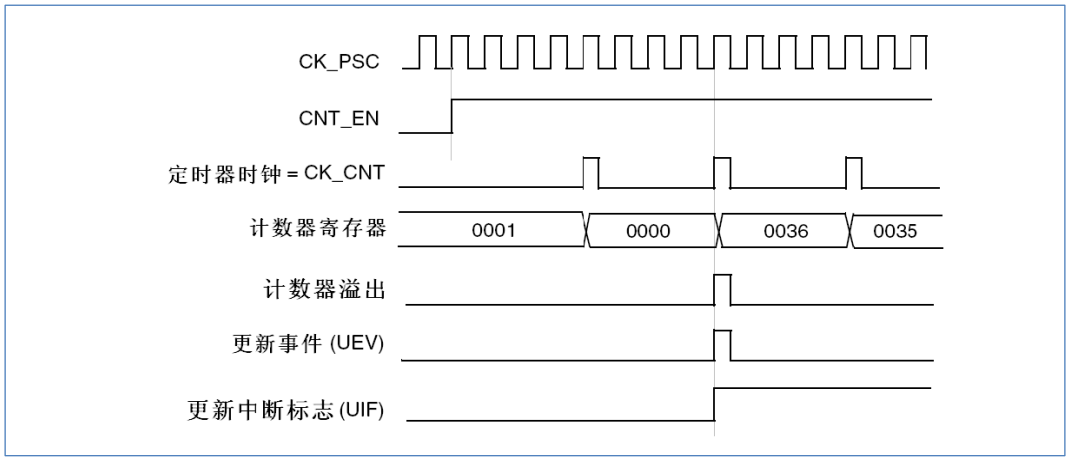

图 15-12 计数器时序图，内部时钟分频因子为 4

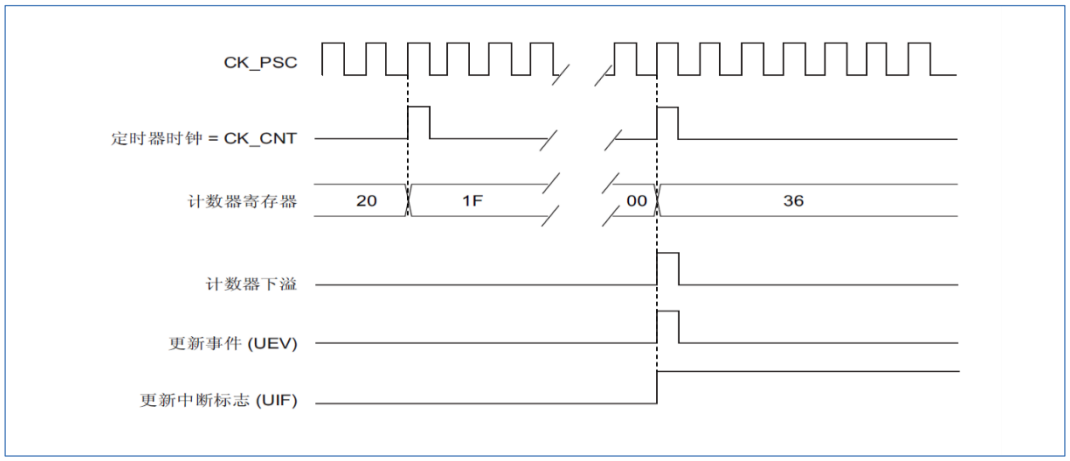

图 15-13 计数器时序图，内部时钟分频因子为 N

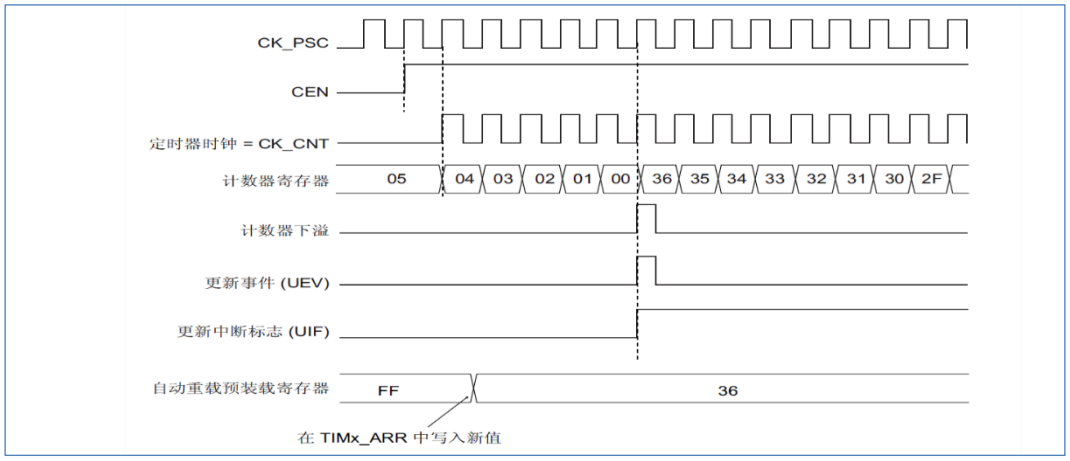

图 15-14 计数器时序图，当没有使用重复计数器时的更新事件

15.2.2.3 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（TIM1_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（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（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：

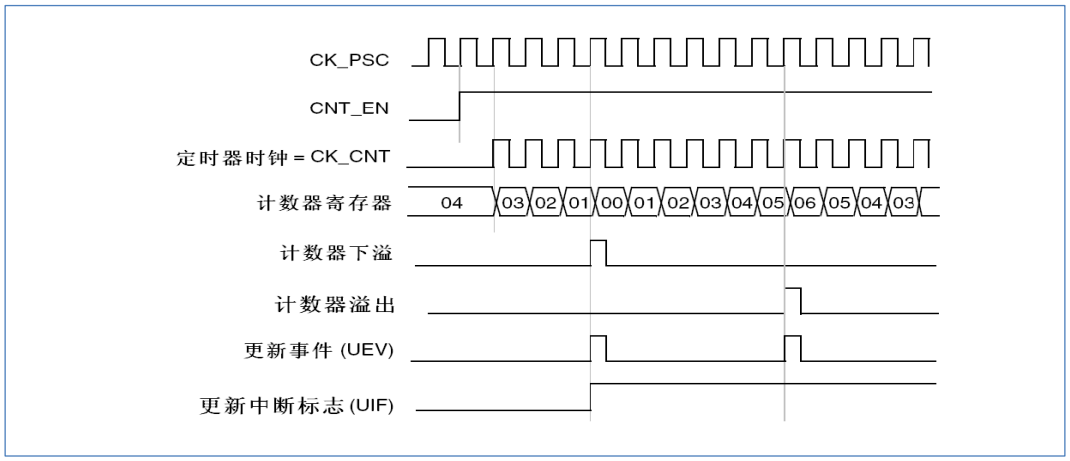

图 15-15 计数器时序图，内部时钟分频因子为 1，TIM1_ARR=0x6（这里使用了中央对齐模式 1）

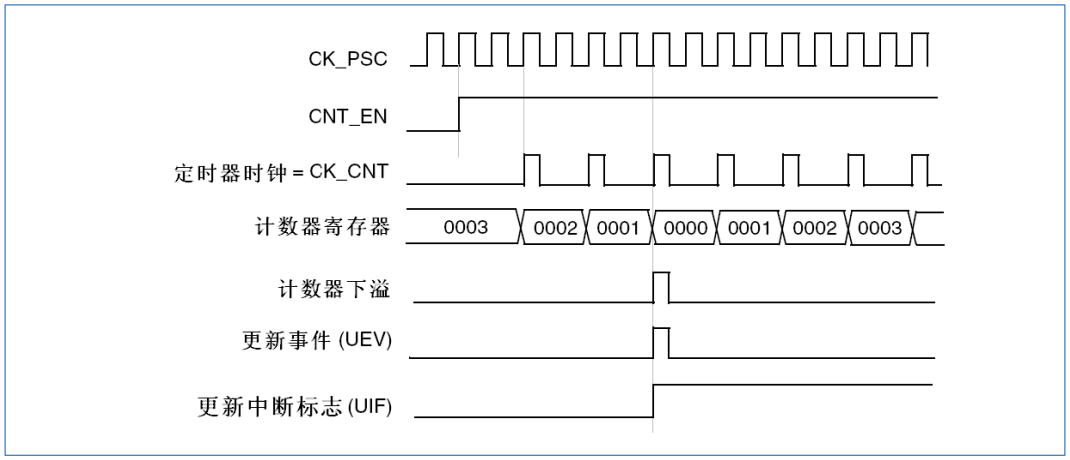

图 15-16 计数器时序图，内部时钟分频因子为 2（这里使用了中央对齐模式 1）

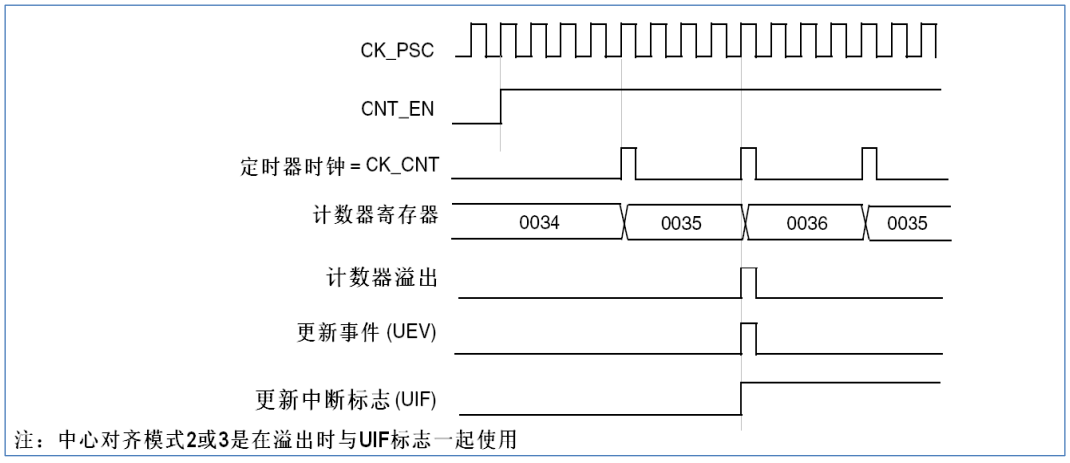

图 15-17 计数器时序图，内部时钟分频因子为 4，TIM1_ARR=0x36（这里使用了中央对齐模式 2 或 3）

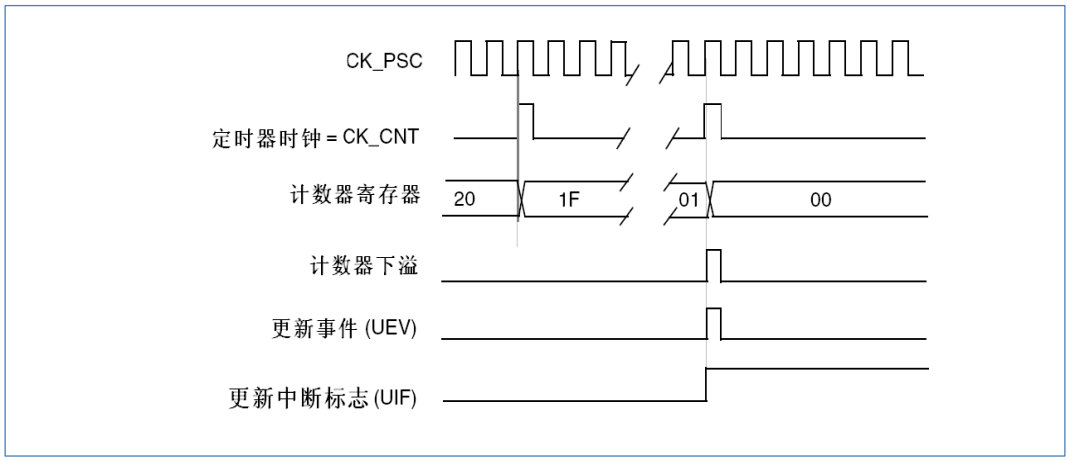

图 15-18 计数器时序图，内部时钟分频因子为 N

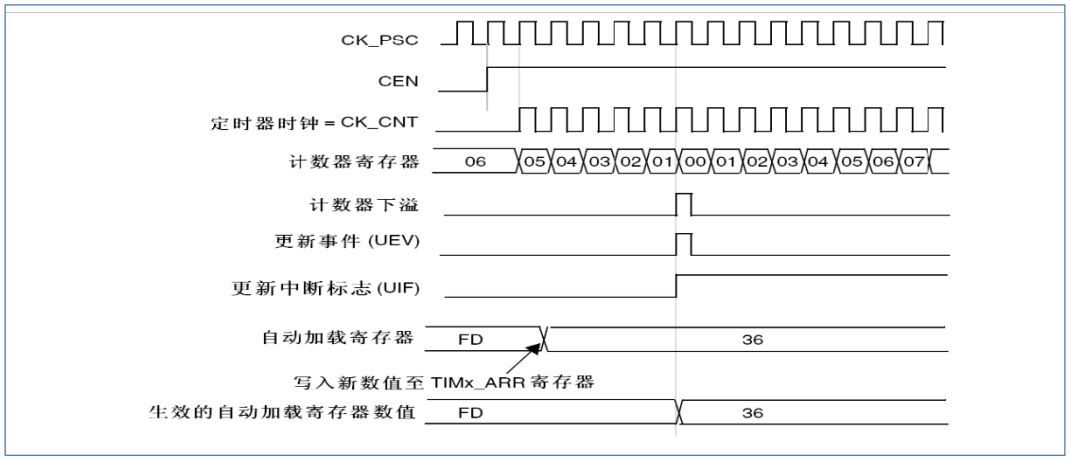

图 15-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

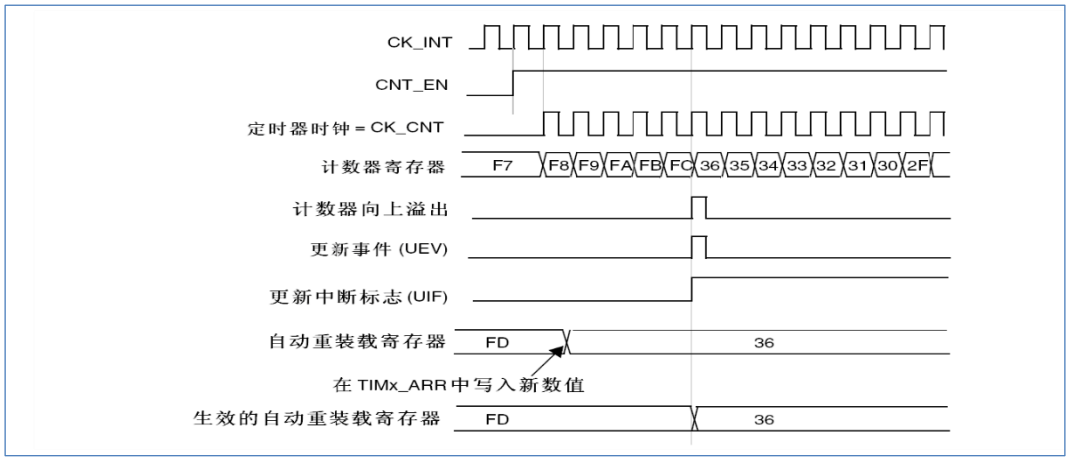

图 15-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

15.2.3 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 2xTck。

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 15-21）。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。

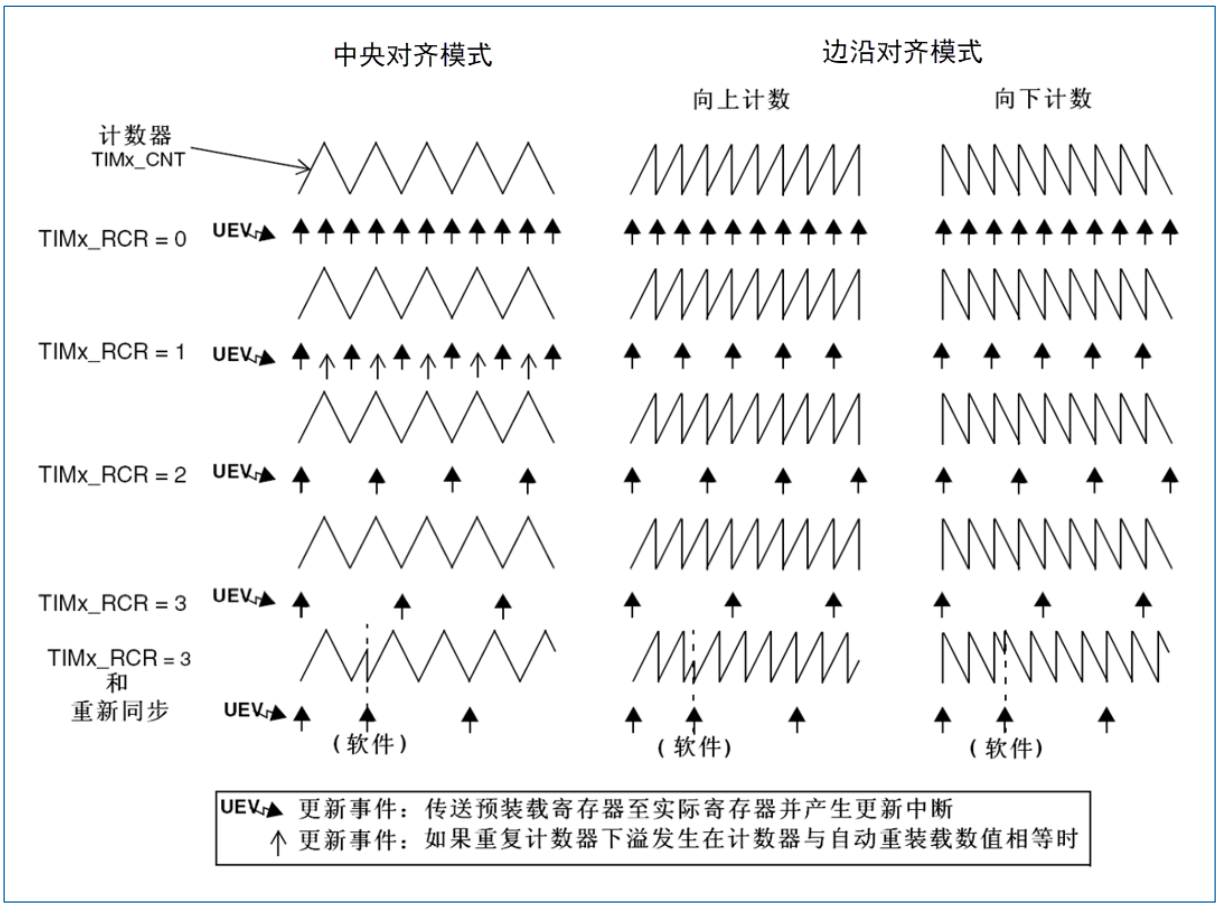

图 15-21 不同模式下更新速率的例子，及 TIM1_RCR 的寄存器设置

更新事件移位功能（UEV SHIFT）：在原有 TIM1_RCR 寄存器上新增一个比特位。配置 TIM1_RCR 寄存器的 UEV_SHIFT 位，可以控制第一次产生 UEV 事件的位置移位，移位的单位是一次 CNT 计数到最大值时间，如下当 UEV_SHIFT=0，UEV 事件在 CNT=0 起始时刻发生，当 UEV_SHIFT=1，UEV 事件在第一次 CNT 计数到最大值时刻产生。

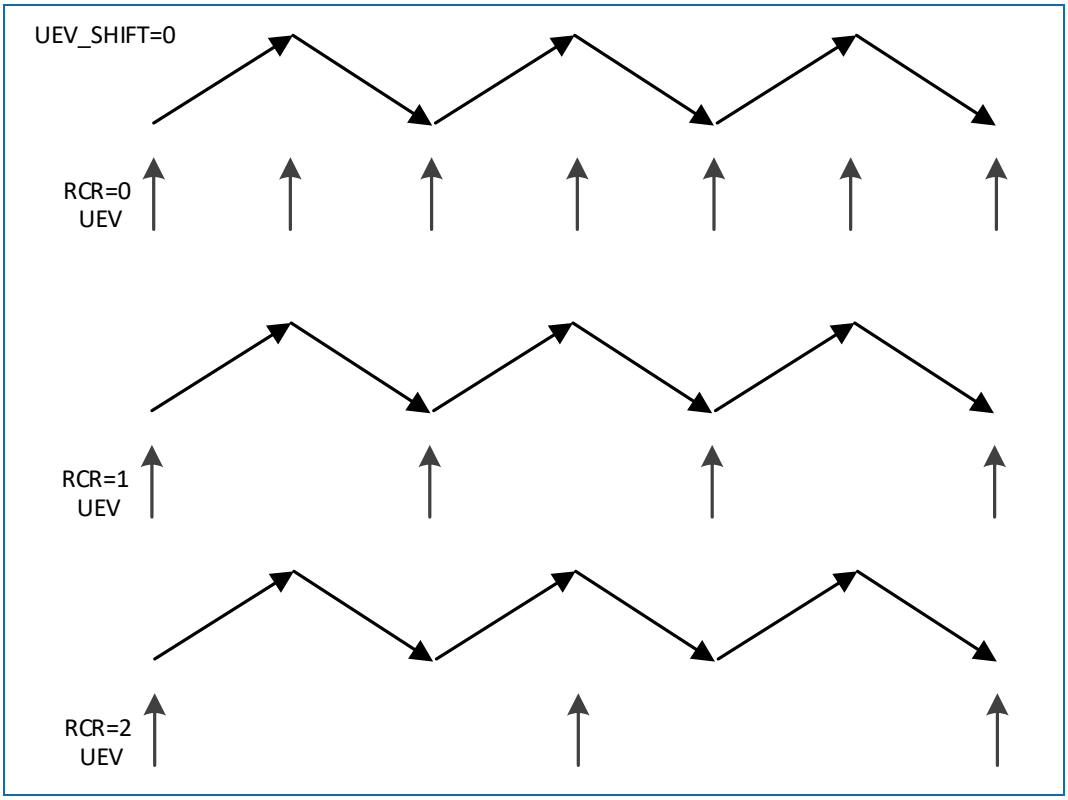

图 15-22 UEV_SHIFT 为 0 时，不同 RCR 下的 UEV 产生情况

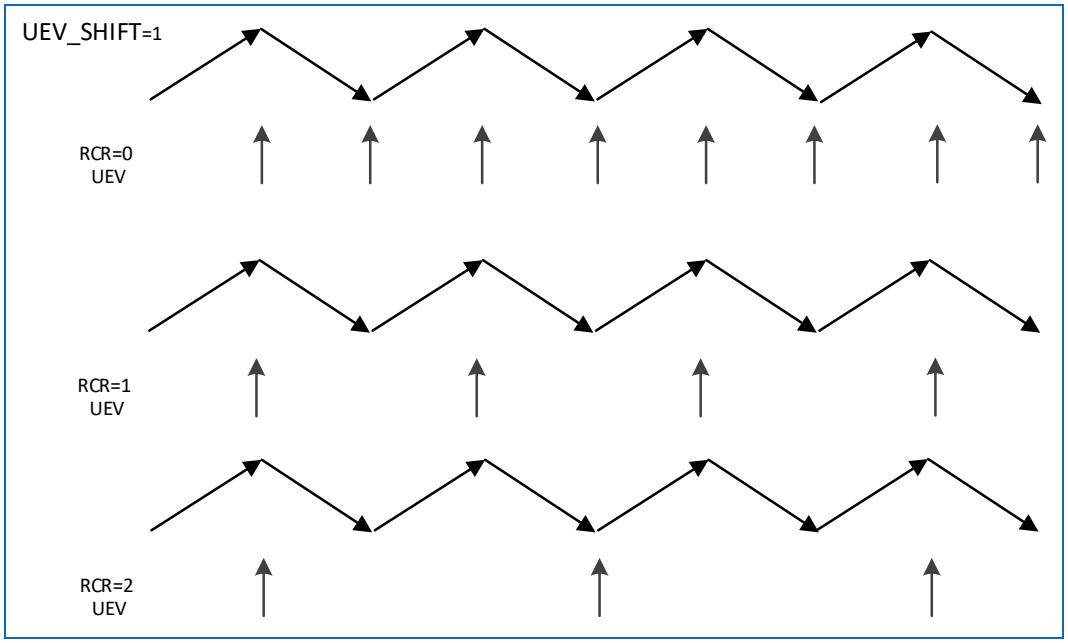

图 15-23 UEV_SHIFT 为 1 时，不同 RCR 下的 UEV 产生情况

15.2.4 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（TI1 或 TI2）
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器，参见“16.2.15 定时器同步”。

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）是事实上的控制位，并且只能被软件修改（UG 位仍被自动清除）。只要 CEN 位被写成'1'，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

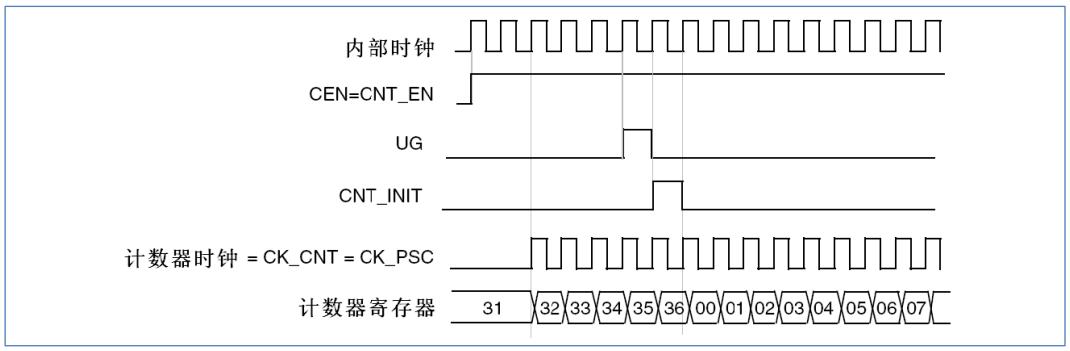

图 15-24 一般模式下的控制电路，内部时钟分频因子为 1

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（TI1 或 TI2）的每个上升沿或下降沿计数。

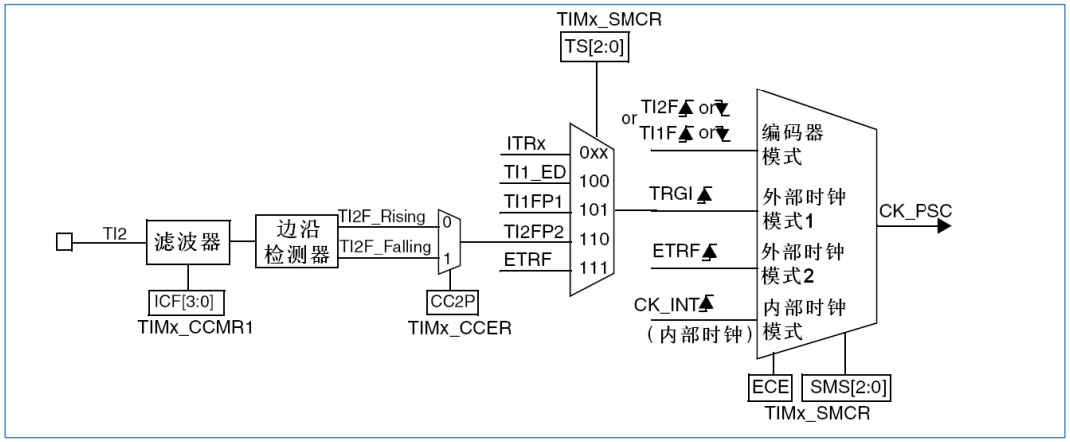

图 15-25 TI2 外部时钟连接例子

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。

注意：捕获预分频器不用作触发，所以不需要对它进行配置。

当上升沿出现在 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。

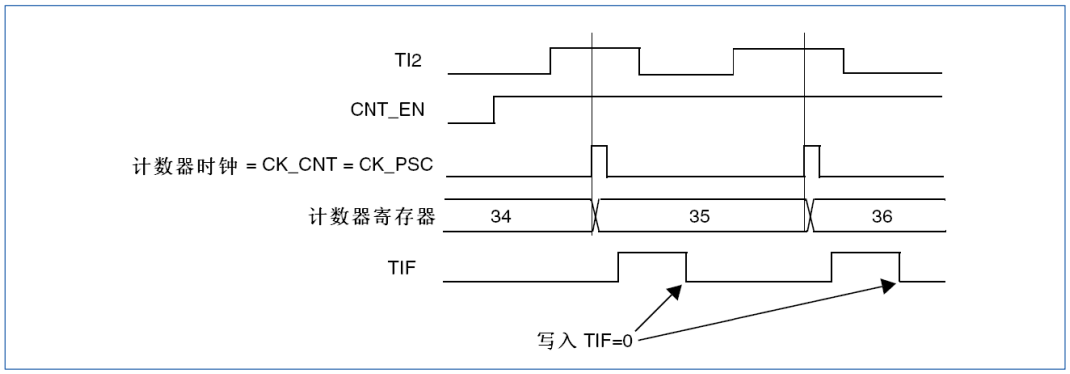

图 15-26 外部时钟模式 1 下的控制电路

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。

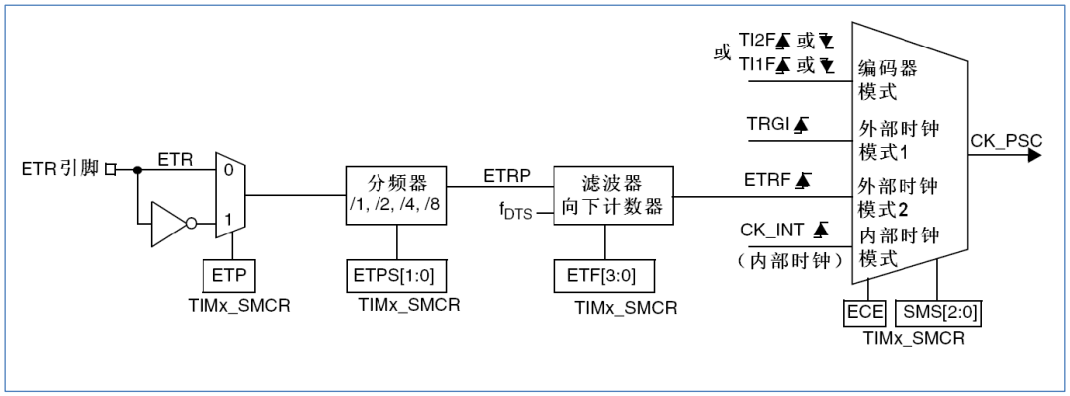

图 15-27 外部触发输入框图

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。

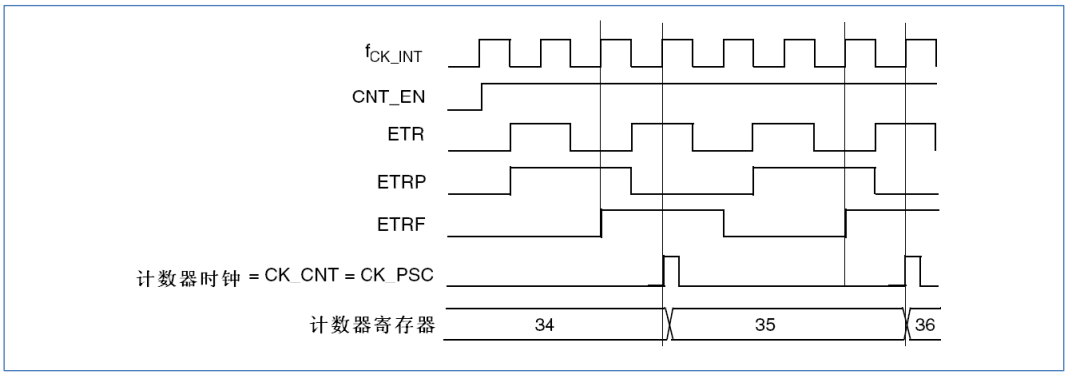

图 15-28 外部时钟模式 2 下的控制电路

15.2.5 捕获/比较通道

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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）。通道 5 和通道 6 只能被配置为输出模式。

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）。

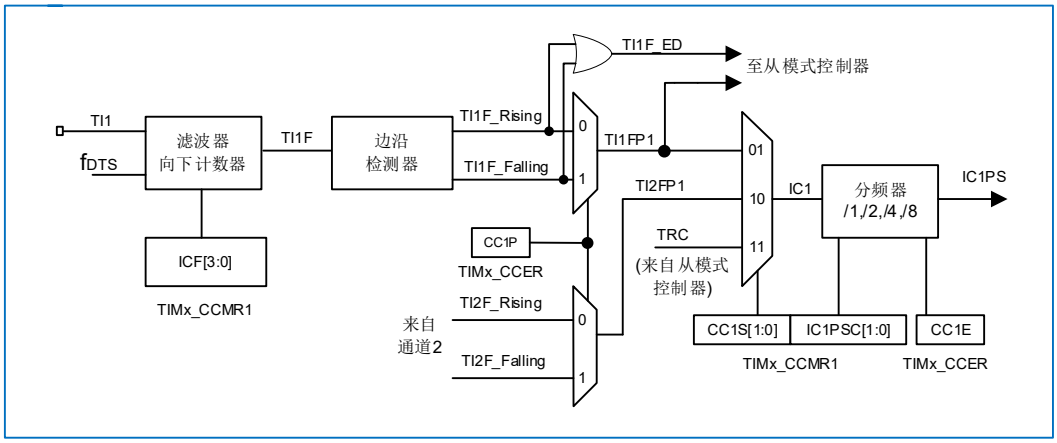

图 15-29 捕获/比较通道（如：通道 1 输入部分）

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。

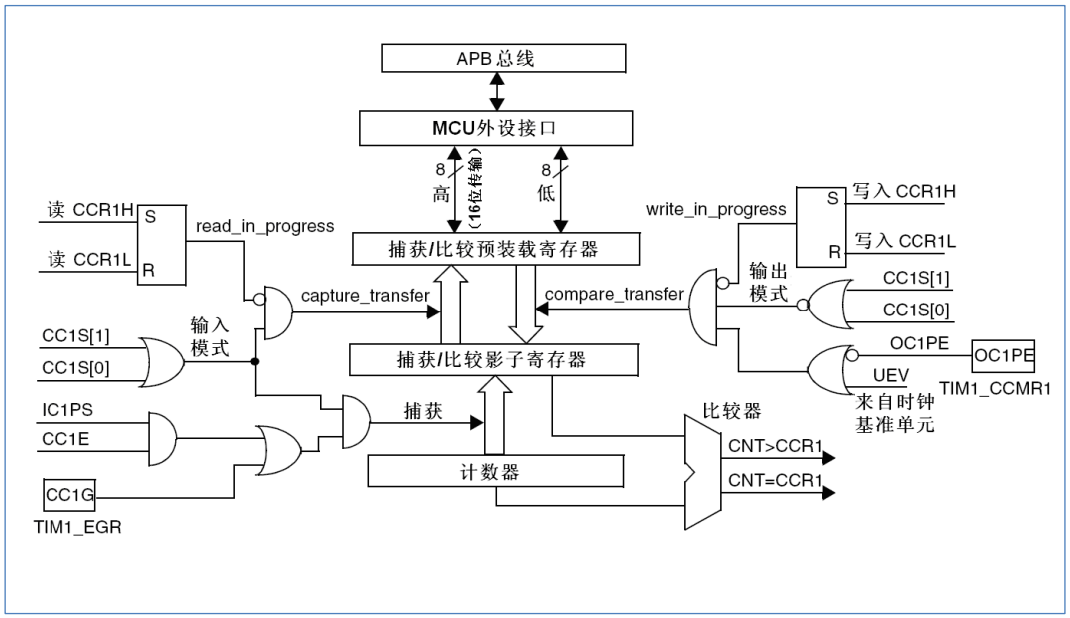

图 15-30 捕获/比较通道 1 的主电路

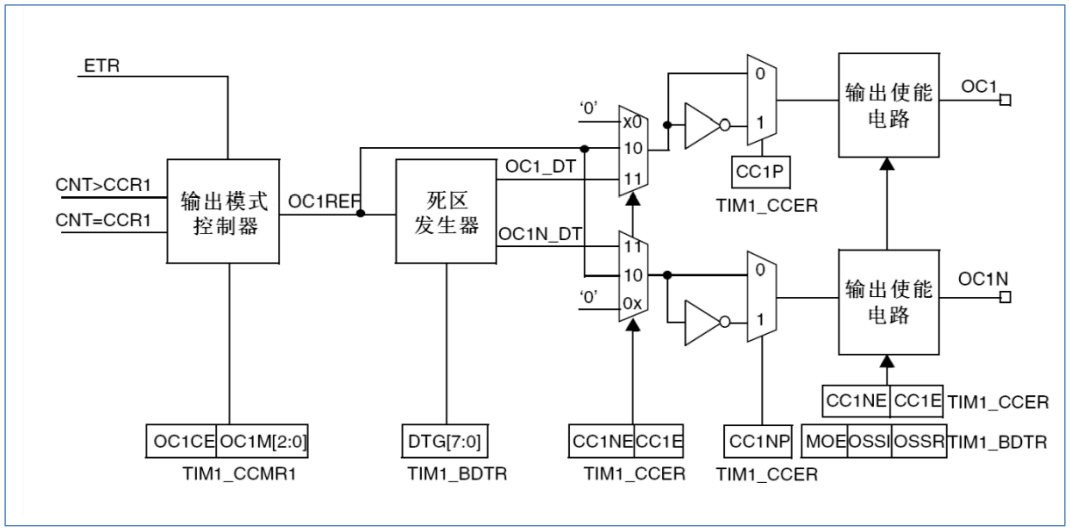

图 15-31 捕获/比较通道的输出部分（通道 1 至 3）

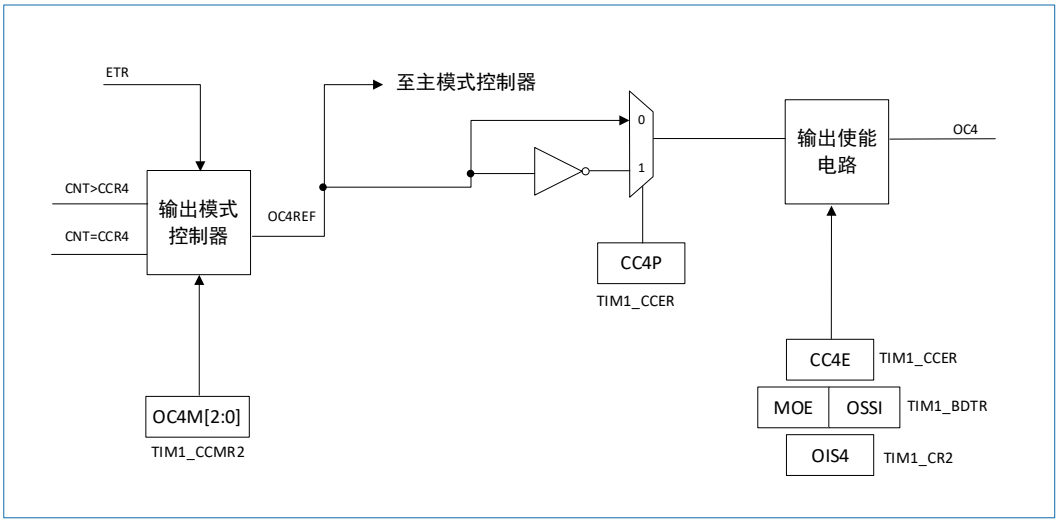

图 15-32 捕获/比较通道的输出部分（通道 4）

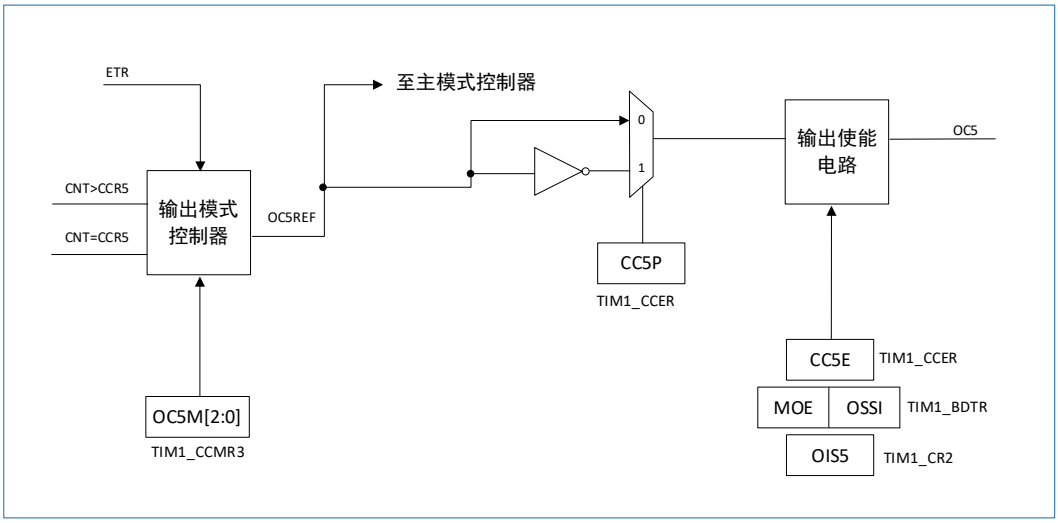

图 15-33 捕获/比较通道的输出部分（通道 5/6）

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。

15.2.6 输入捕获模式

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（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，通过设置 TIM1_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM1_CCR1 寄存器。

- CC1IF 标志被设置（中断标志）。当发生至少两个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

注意：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和/或 DMA 请求。

15.2.7 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。
- 这两个 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/CC1NP=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。

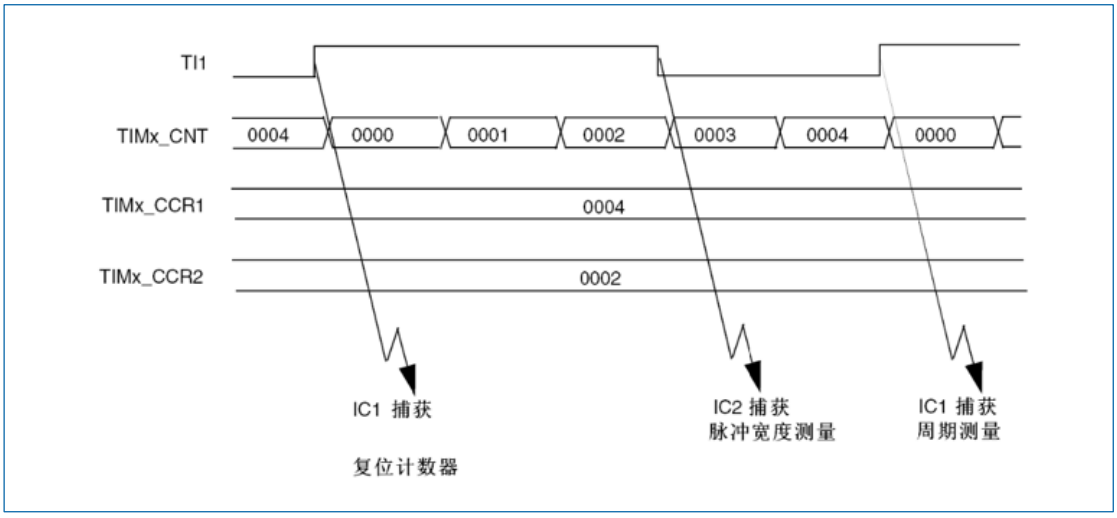

图 15-34 PWM 输入模式时序

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。

15.2.8 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和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

15.2.9 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屏蔽（TIM1_DIER 寄存器中的 CCxIE 位），则产生一个中断。
- 若设置了相应的使能位（TIM1_DIER 寄存器中的 CCxDE 位，TIM1_CR2 寄存器中的 CCDS 位选择 DMA 请求功能），则产生一个 DMA 请求。

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。

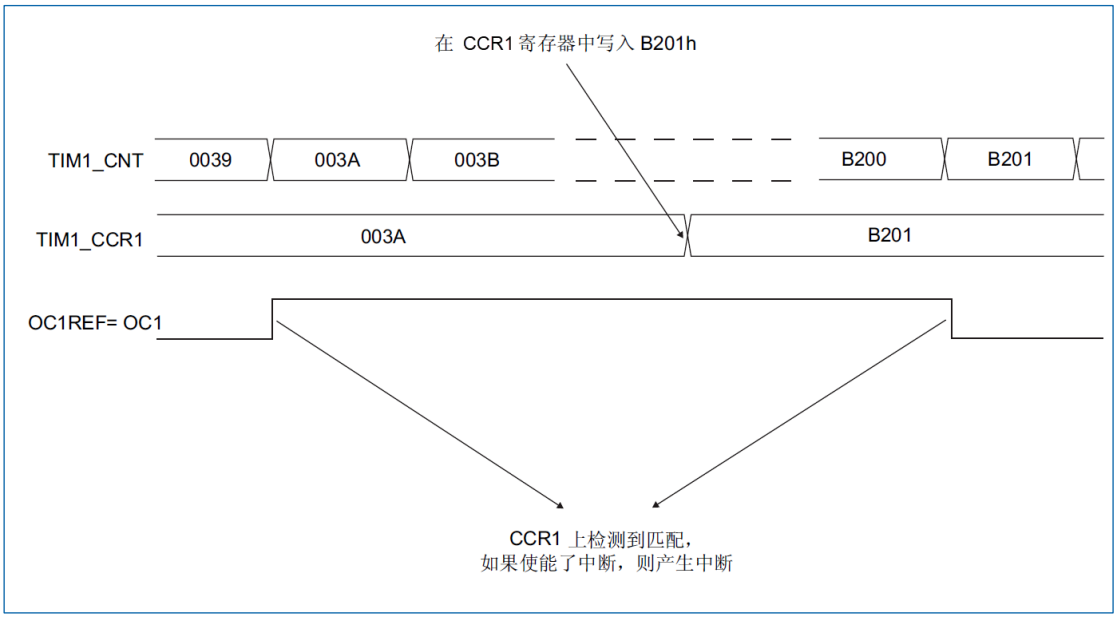

图 15-35 输出比较模式，翻转 OC1

15.2.10 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（TIM1_CCER 和 TIM1_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见章节：“15.3.9 TIM1 捕捉/比较使能寄存器（TIM1_CCER）”。

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’。图 15-36 为 TIM1_ARR=8 时边沿对齐的 PWM 波形实例。

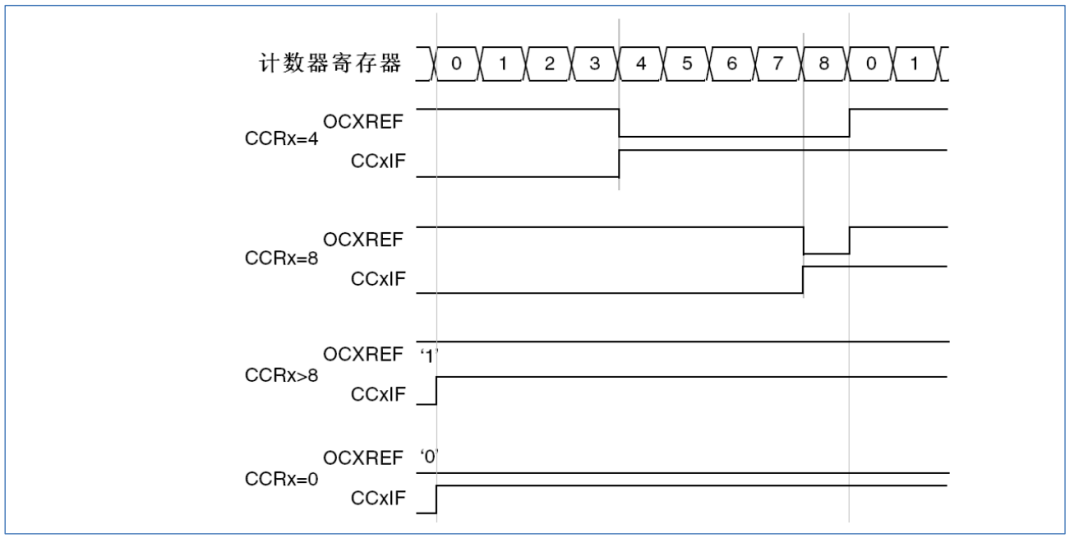

图 15-36 边沿对齐的 PWM 波形（ARR=8）

- 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。

在 PWM1 或 PWM2 模式下，可以设置 CMPR_EN 位和 TIM1_CMPR 寄存器，改变 PWM 的脉宽。通过设置 TIM1_CR2 的 CMPRy_EN 位，使得 TIM1_CMPRy 与 TIM1_ARR 比较，控制通道 y 的输出脉宽；反之，清空 TIM1_CR2 的 CMPRy_EN 位，则 TIM1_CCRy 与 TIM1_ARR 比较，控制通道 y 的输出脉宽。

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。

中央对齐模式时，在向下计数时，在 PWM1 或 PWM2 模式下，可以设置 CMPR_EN 位和 TIM1_CMPR 寄存器，改变 PWM 的脉宽，其他时候使用 CCR 与 ARR 比较改变 PWM 脉宽。

图 15-37 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。

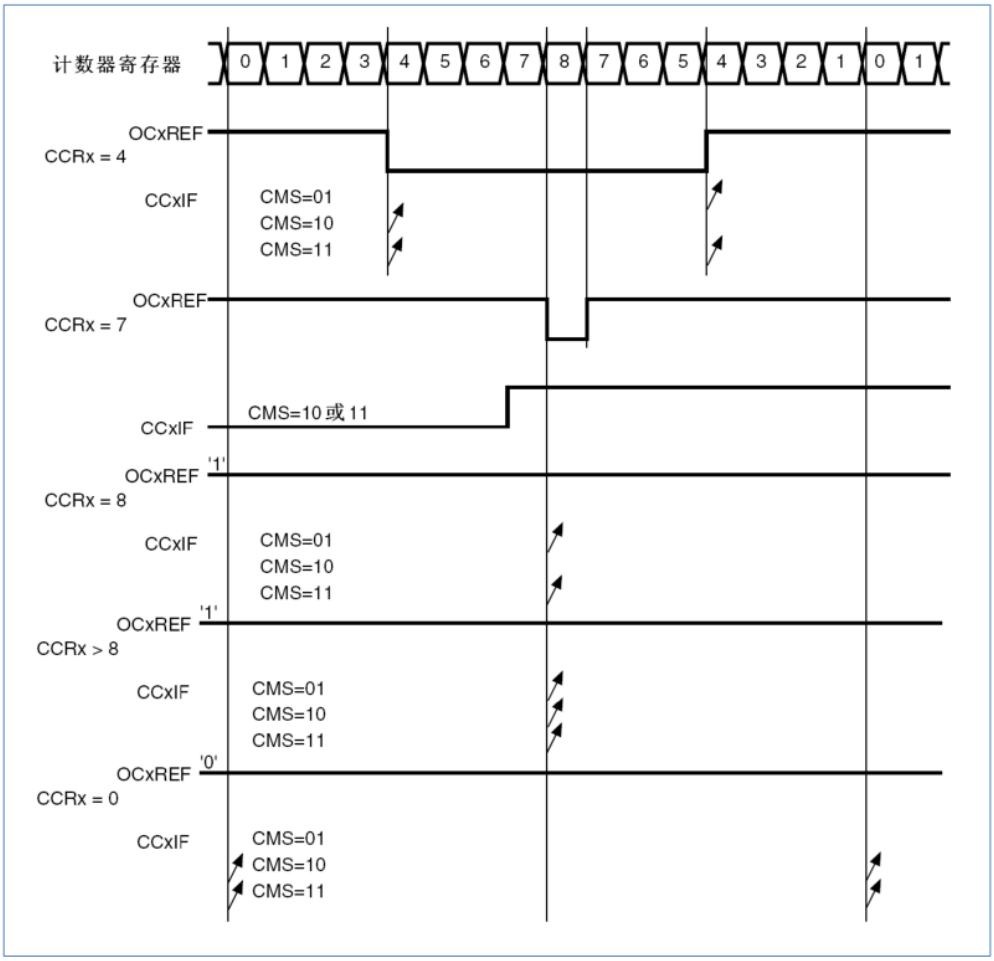

图 15-37 中央对齐的 PWM 波形（ARR=8）

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 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。需要注意：
  - 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。

简易的数据搬移功能：TIM1 实现简易的数据搬移功能，当 TIM1 工作在 PWM 输出模式时，第一次 CCR4 等于 CNT 时，将寄存器 CCR_PRE1[15:0]加载到目的地址，当第二次 CCR4 等于 TIMCNT 时，将 CCR_PRE1[15:0]的值加载到目的地址。

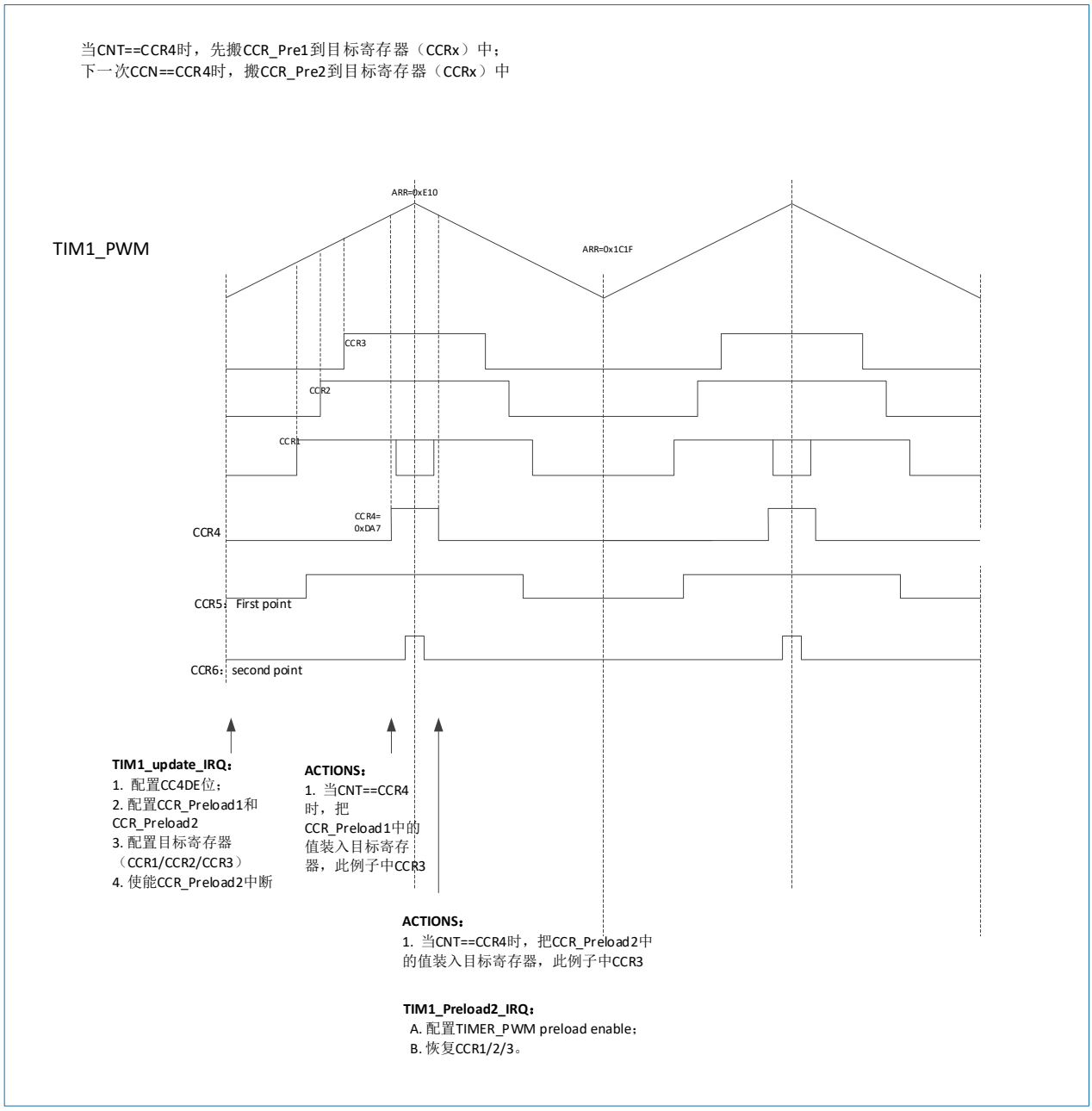

图 15-38 PWM 模式下，开启数据搬移的 PWM 波形

15.2.11 互补输出和死区插入

高级控制定时器（TIM）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 15-4。

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。死区时间的插入，根据需求可以分别设置前后死区时间。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，延时时间由 TIM1_BDTR 寄存器中 DTG[7:0]确定；如果设置 TIM1_DTGR 寄存器中的 DTG2_EN 位，则延时时间由 TIM1_DTGR 寄存器中 DTG2[7:0]确定。
- 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，延时时间由 TIM1_BDTR 寄存器中 DTG[7:0]确定。

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）

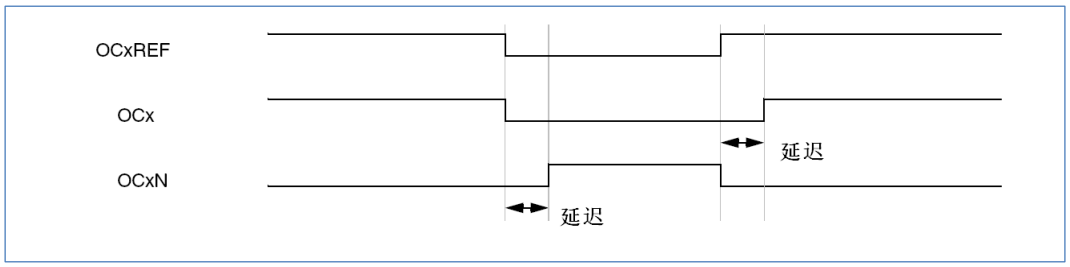

图 15-39 带死区插入的互补输出

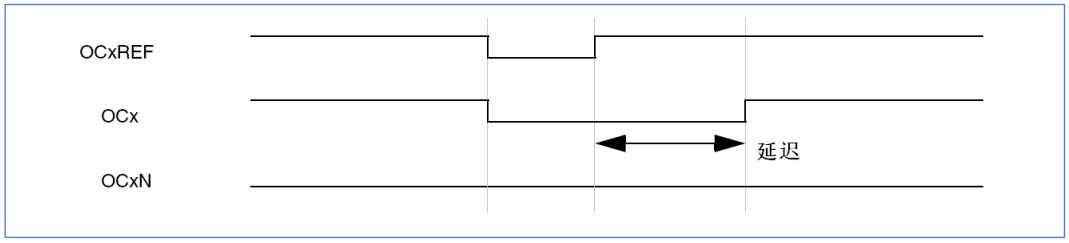

图 15-40 死区波形延迟大于负脉冲

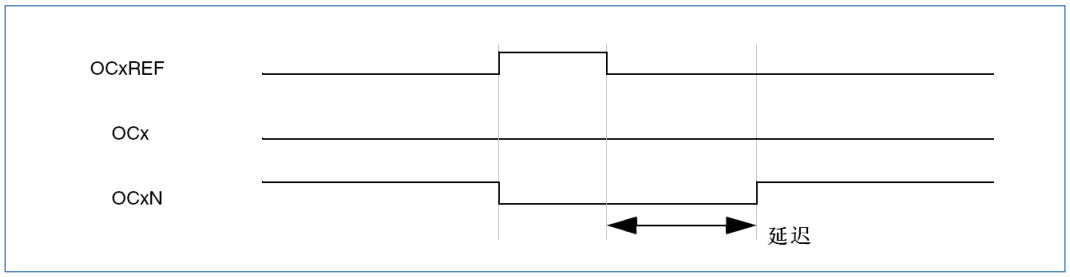

图 15-41 死区波形延迟大于正脉冲

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（和 TIM1_DTGR 寄存器中的 DTG2 位）编程配置。详见 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。

注意：当只使能 OCxN（CCxE=0, CCxNE=1）时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 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。

15.2.12 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 15-4。

刹车信号源既可以是刹车输入引脚也可以是一个时钟失败事件，或者是 TIM1_OR 寄存器的 VCOMPO1、VCOMPO2 和 VCOMPO3 指定比较器输出的结果。时钟失败事件由复位时钟控制器中的时钟安全系统产生，详见“5.2.10 HSE 时钟安全系统（CSSHSE）”和“5.2.11 LSE 时钟安全系统（CSSLSE）”。

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，配置刹车信号输入滤波器（滤波控制位在 TIM1_BDTR 的 BKF[3:0]），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。尤其是当它为低时写 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（大约两个 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。

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

刹车由 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，LOCK 为 level1 时锁定 BKF[3:0]）。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。

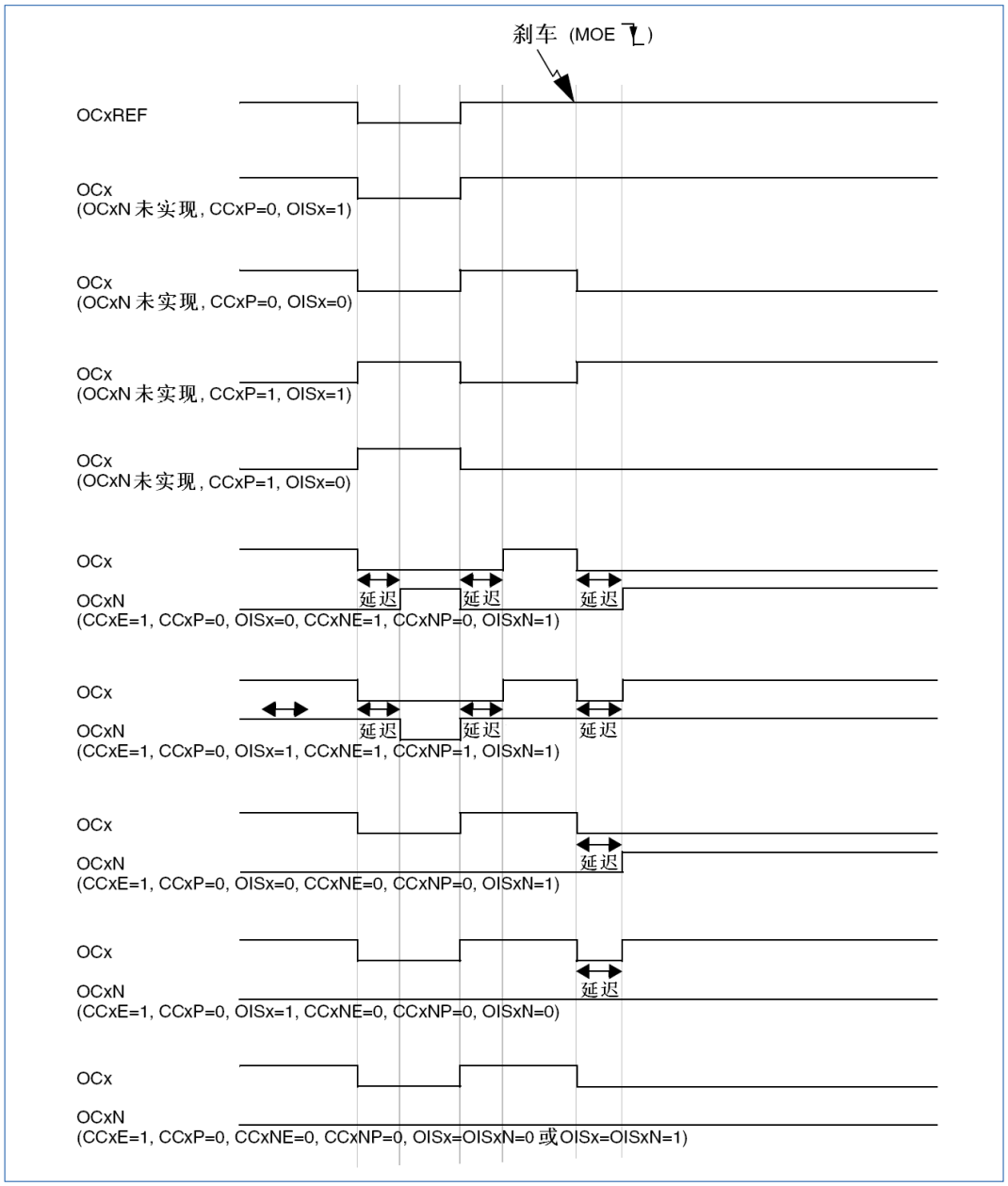

图 15-42 响应刹车的输出

15.2.13 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 TIMx 被置于 PWM 模式。

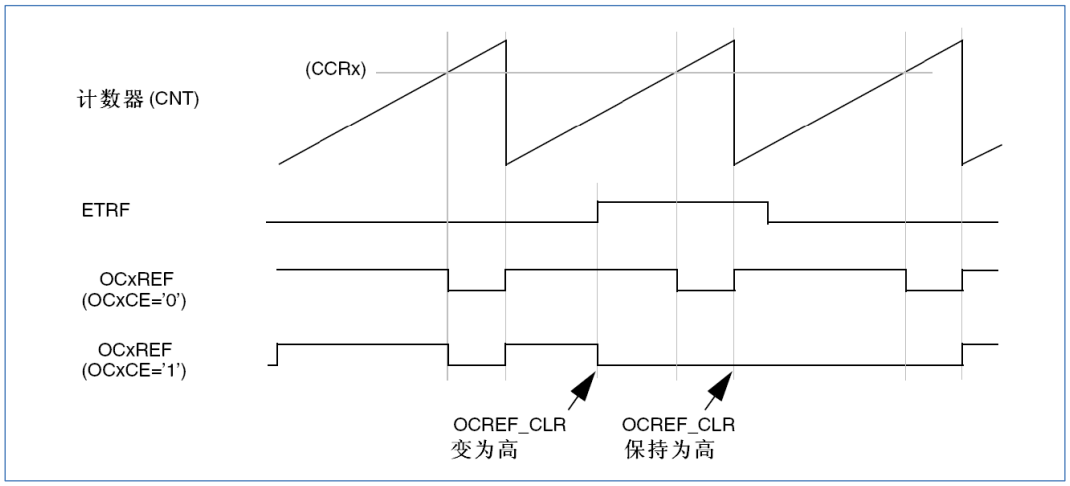

图 15-43 清除 TIMx 的 OCxREF

15.2.14 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。这样你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 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIM1_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

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

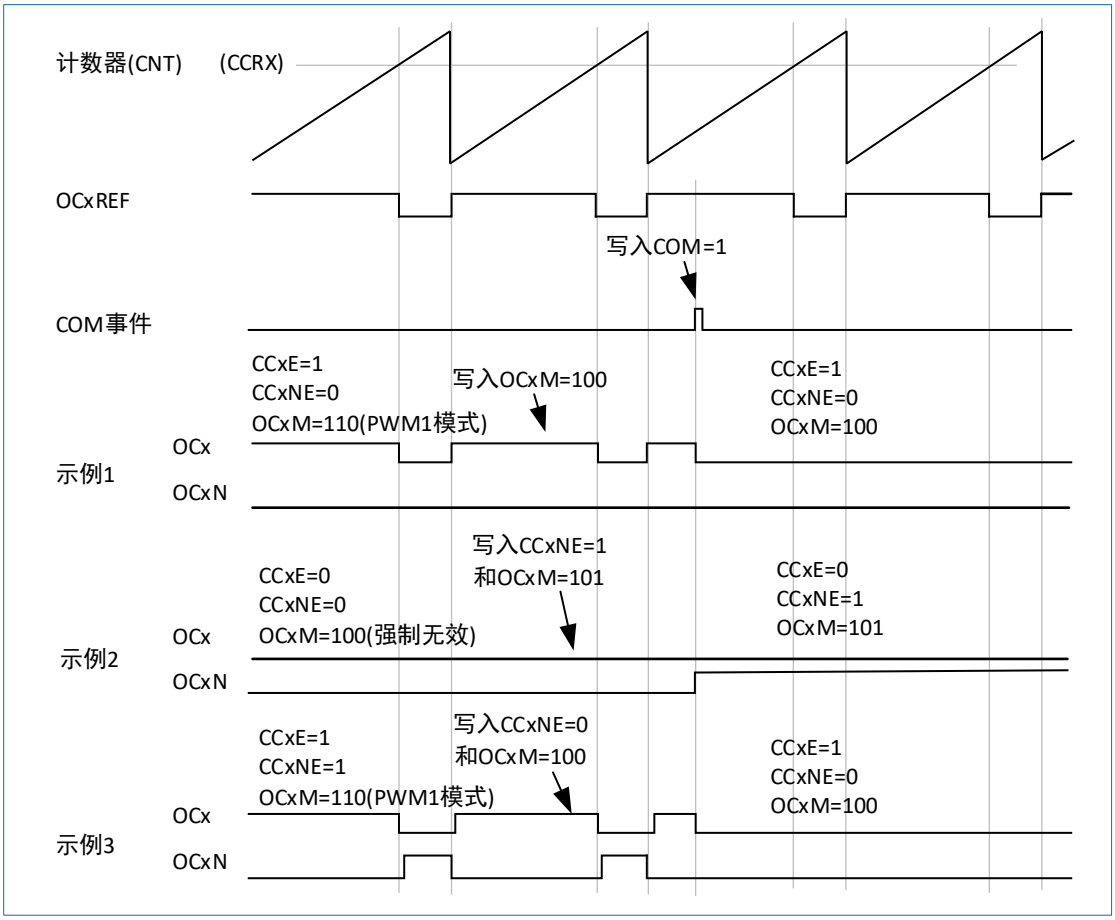

图 15-44 产生六步 PWM，使用 COM 的例子（OSSR=1）

15.2.15 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
- 向下计数方式：计数器 CNT>CCRx

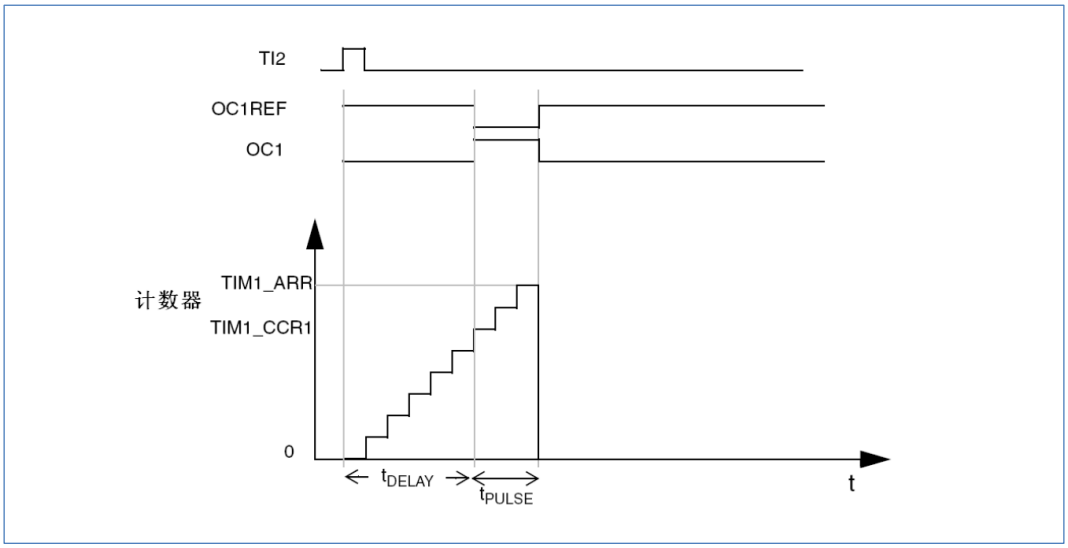

图 15-45 单脉冲模式的例子

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 映像到 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。

15.2.16 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。参看表 15-2，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 15-2 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）	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械

零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

图 15-46 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）。

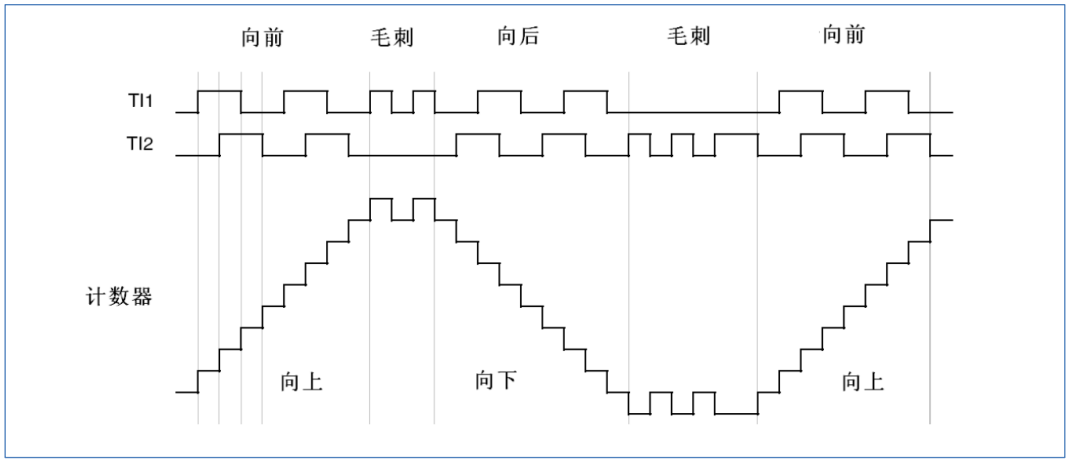

图 15-46 编码器模式下的计数器操作实例

图 15-47 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）。

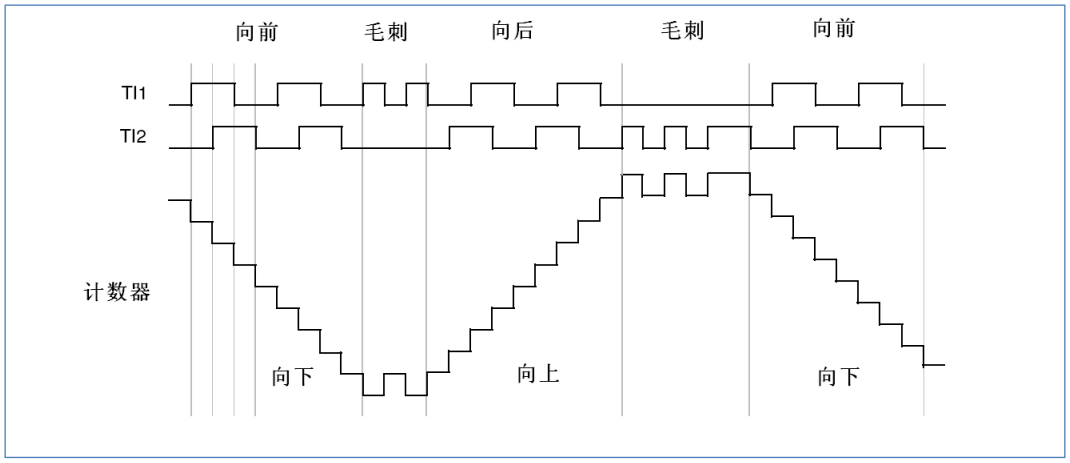

图 15-47 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度，加速度，减速度）。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，你可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

15.2.17 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 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下节给出了此特性用于连接霍尔传感器的例子。

15.2.18 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（如 TIM2 或 TIM3）定时器作为“接口定时器”来连接霍尔传感器，TIMx 的 3 个定时器输入脚（CC1、CC2、CC3）通过一个异或门连接到 TI1 输入通道（通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择），“接口定时器”捕获这个信号。

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

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

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以（通过触发一个 COM 事件）用于改变高级定时器 TIM1 各个通道的属性，而高级控制定时器产生 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（TIMx_CR2 寄存器中 CCPC=1），同时触发输入控制 COM 事件（TIMx_CR2 寄存器中 CCUS=1）。在一次 COM 事件后，写入下一步的 PWM 控制位（CCxE、OCxM），这可以在处理 OC2REF 上升沿的中断子程序里实现。

图 15-48 显示了这个实例。

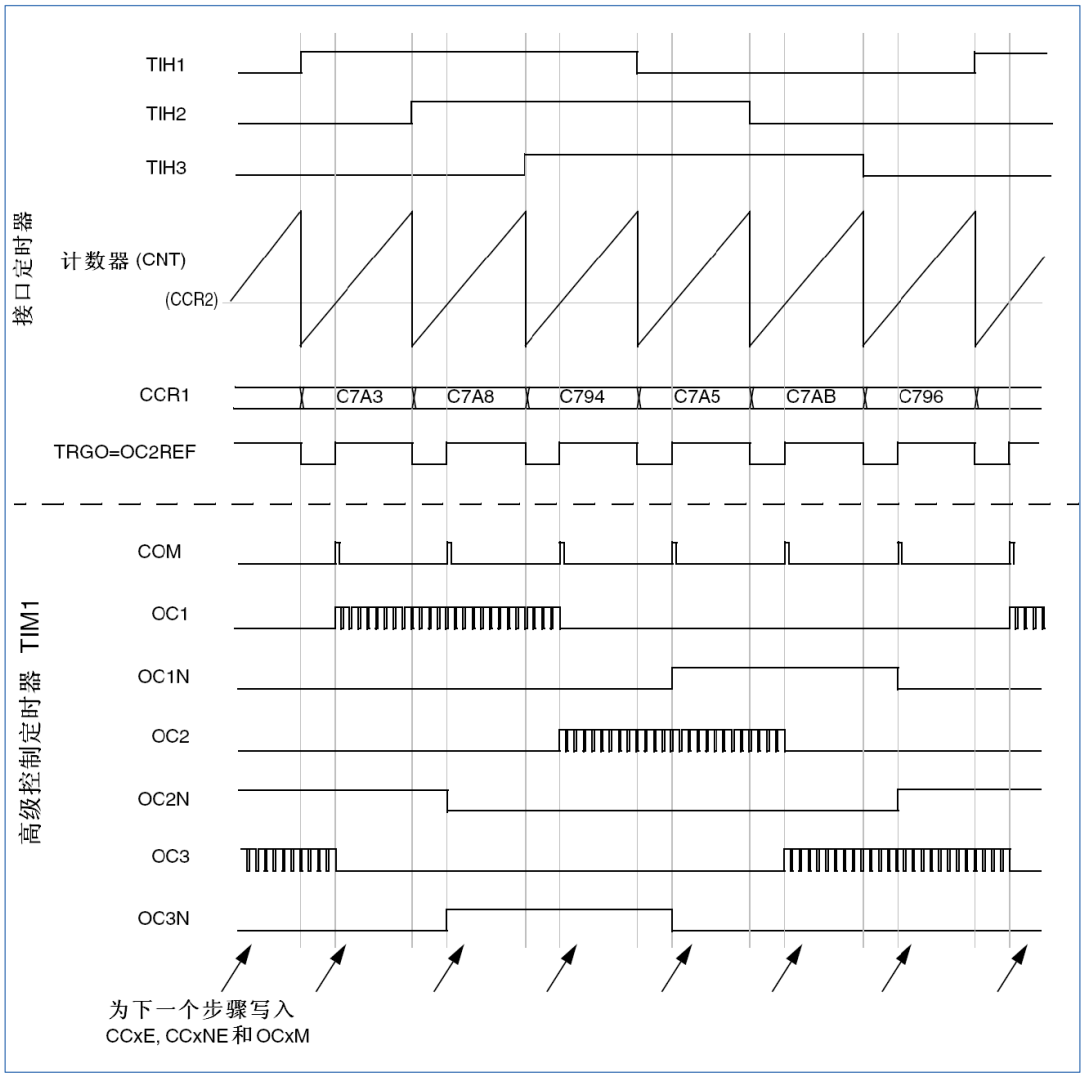

图 15-48 霍尔传感器接口的实例

15.2.19 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。

15.2.19.1 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。

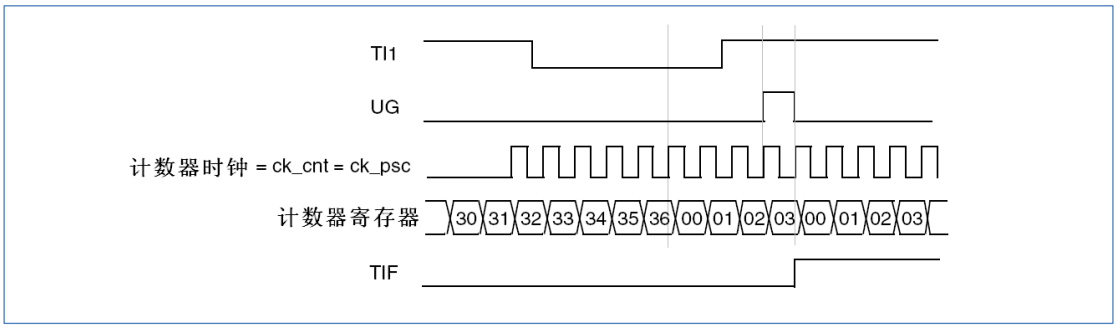

图 15-49 复位模式下的控制电路

15.2.19.2 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。

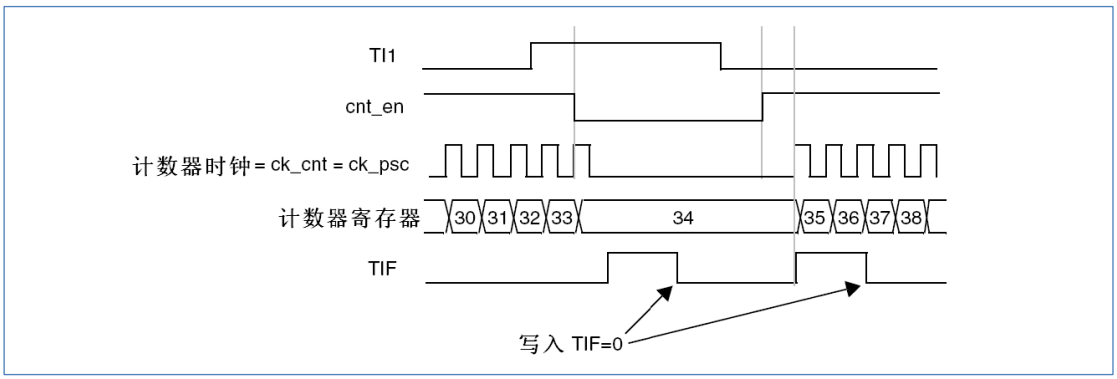

图 15-50 门控模式下的控制电路

15.2.19.3 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。

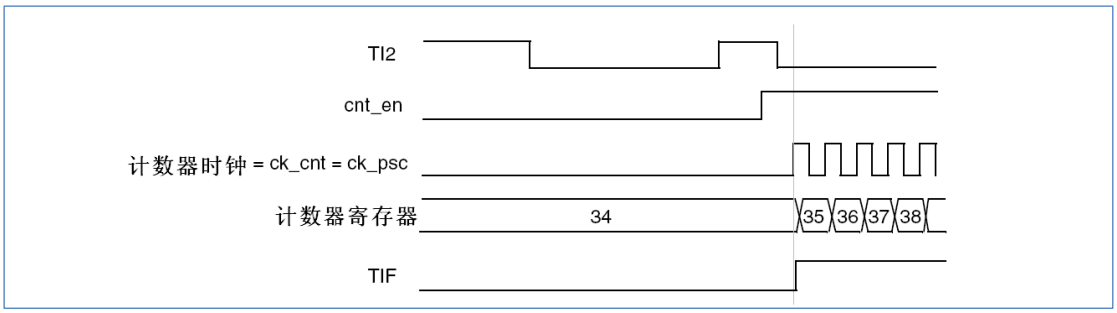

图 15-51 触发器模式下的控制电路

15.2.19.4 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。

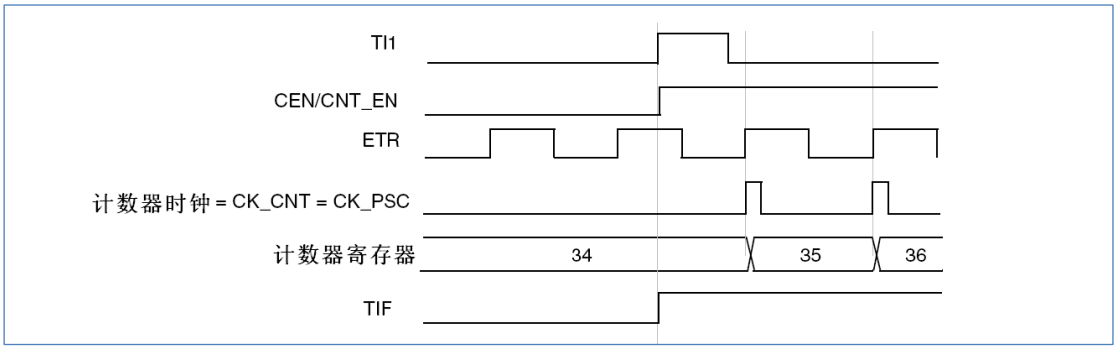

图 15-52 外部时钟模式 2＋触发模式下的控制电路

15.2.20 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，详见章节“16.2.15 定时器同步”。

15.2.21 调试模式

当微控制器进入调试模式时（Cortex-M0 核心停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIM1 计数器可以或者继续正常操作，或者停止。

15.3 TIM1 寄存器

基地址：0x4001 2C00

空间大小：0x400

15.3.1 TIM1 控制寄存器 1（TIM1_CR1）

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CC4_ADC_SEL	Res			CC6_ADC_SEL	CC5_ADC_SEL	Res									
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：选择 CC4 触发 ADC 的信号来源（ADC trigger CC4 generation source selection） 参见 CC5_ADC_SEL。
位 30:28	Res：保留 必须保持复位值。
位 27	CC6_ADC_SEL：选择 CC6 触发 ADC 的信号来源（ADC trigger CC6 generation source selection） 参见 CC5_ADC_SEL。
位 26	CC5_ADC_SEL：选择 CC5 触发 ADC 的信号来源（ADC trigger CC5 generation source selection），参考 ADC 章节的触发源选择 ● 0：CC5_TO_ADC 触发输出信号来自 OC5 ● 1：CC5_TO_ADC 触发输出信号来自 OC5REF
位 25:10	Res：保留 必须保持复位值。
位 9:8	CKD[1:0]：时钟分频因子（Clock division） 定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR, TIx）所用的采样时钟之间的分频比例。 ● 00：$t_{DTS}=t_{CK\_INT}$ ● 01：$t_{DTS}=2*t_{CK\_INT}$ ● 10：$t_{DTS}=4*t_{CK\_INT}$ ● 11：保留
位 7	ARPE：自动重装载预装载允许位（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。计数器交替地向上和向下计数。配置为输出的通道（TIM1_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向下计数时被设置。 ● 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道（TIM1_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向上计数时被设置。 ● 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道（TIM1_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，在计数器向上和向下计数时均被设置。
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或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： ○ 计数器溢出/下溢 ○ 设置 UG 位 ○ 从模式控制器产生的更新 ● 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
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。发生下述任一事件将产生更新（UEV）事件： ○ 计数器溢出/下溢 ○ 设置 UG 位 ○ 从模式控制器产生的更新，具有缓存的寄存器被装入它们的预装载值。 ● 1：禁止 UEV。不产生更新事件，影子寄存器（ARR、PSC、CCRx）保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
位 0	CEN：使能计数器（Counter enable） ● 0：禁止计数器 ● 1：使能计数器

15.3.2 TIM1 控制寄存器 2（TIM1_CR2）

偏移地址：0x04

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMPR4_EN	CMPR3_EN	CMPR2_EN	CMPR1_EN	Res									OIS6	Res	OIS5
rw	rw	rw	rw										rw		rw

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

位 31	CMPR4_EN：OC4 选择比较寄存器（CMPR4 enable）。 参见 CMPR1_EN 位。
位 30	CMPR3_EN：OC3 选择比较寄存器（CMPR3 enable）。 参见 CMPR1_EN 位。
位 29	CMPR2_EN：OC2 选择比较寄存器（CMPR2 enable）。 参见 CMPR1_EN 位。
位 28	CMPR1_EN：OC1 选择比较寄存器（CMPR1 enable） 在 PWM2 模式下，当计数器向下计数时，OC1 可选择两种比较寄存器与 ARR 比较，控制通道 1 的脉宽。 ● 0：TIM1_CCR1 与 ARR 比较 ● 1：TIM1_CMPR1 与 ARR 比较
位 27:19	Res：保留 必须保持复位值。
位 18	OIS6：输出空闲状态 6（OC6 Output Idle state）。 参见 OIS1 位。
位 17	Res：保留 必须保持复位值。
位 16	OIS5：输出空闲状态 5（OC5 Output Idle state）。 参见 OIS1 位。
位 15	Res：保留 必须保持复位值。
位 14	OIS4：输出空闲状态 4（OC4 Output Idle state）。 参见 OIS1 位。
位 13	OIS3N：输出空闲状态 3（OC3N Output Idle state） 参见 OIS1N 位。
位 12	OIS3：输出空闲状态 3（OC3 Output Idle state） 参见 OIS1 位。
位 11	OIS2N：输出空闲状态 2（OC2N Output Idle state） 参见 OIS1N 位。
位 10	OIS2：输出空闲状态 2（OC2 Output Idle state） 参见 OIS1 位。
位 9	OIS1N：输出空闲状态 1（OC1N Output Idle state） ● 0：当 MOE=0 时，死区后 OC1N=0。 ● 1：当 MOE=0 时，死区后 OC1N=1。
位 8	OIS1：输出空闲状态 1（OC1 Output Idle state） ● 0：当 MOE=0 时，如果完成了 OC1N，则死区后 OC1=0。 ● 1：当 MOE=0 时，如果完成了 OC1N，则死区后 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 引脚经异或后连到 TI1 输入。
位 6:4	MMS[2:0]：主模式选择（Master mode selection） 这 3 位用于选择在主模式下送到从定时器的同步信息（TRGO）。可能的组合如下： ● 000：复位 TIM1_EGR 寄存器的 UG 位被用于作为触发输出（TRGO）。如果是触发输入产生的复位（从模式控制器处于复位模式），则 TRGO 上的信号相对实际的复位会有一个延迟。 ● 001：使能 计数器使能信号 CNT_EN 被用于作为触发输出（TRGO）。可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 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时（即使它已经为高），触发输出送出一个正脉冲（TRGO）。 ● 100：比较 OC1REF 信号被用于作为触发输出（TRGO） ● 101：比较 OC2REF 信号被用于作为触发输出（TRGO） ● 110：比较 OC3REF 信号被用于作为触发输出（TRGO） ● 111：比较 OC4REF 信号被用于作为触发输出（TRGO）
位 3	CCDS：捕获/比较的 DMA 选择（Capture/Compare DMA selection） ● 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 ● 1：当发生更新事件时，送出 CCx 的 DMA 请求。
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 位更新它们。 ● 1：如果捕获/比较控制位是预装载的（CCPC=1），可以通过设置 COMG 位或 TRGI 上的一个上升沿更新它们。
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，只能在发生通信（COM）事件（COMG 设置或 TRGI 是检测到上升沿，取决于 CCUS 位时被更新。

15.3.3 TIM1 从模式控制寄存器（TIM1_SMCR）

偏移地址：0x08

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
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 31:16	Res：保留 必须保持复位值。
位 15	ETP：外部触发极性（External trigger polarity） 该位选择是用 ETR 还是 ETR 的反相来作为触发操作。 ● 0：ETR 不反相，高电平或上升沿有效。 ● 1：ETR 被反相，低电平或下降沿有效。
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 prescaler） 外部触发信号 ETRP 的频率最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 ● 00：关闭预分频 ● 01：ETRP 频率除以 2 ● 10：ETRP 频率除以 4 ● 11：ETRP 频率除以 8
位 11:8	ETF[3:0]：外部触发滤波（External trigger Filter） 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。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）上的事件被延迟了，以允许在当前定时器与它的从定时器间的完美同步（通过 TRGO）。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
位 6:4	TS[2:0]：触发选择（Trigger selection） 选择用于同步计数器的触发输入。 ● 000：内部触发 0（ITR0） ● 001：内部触发 1（ITR1） ● 010：内部触发 2（ITR2） ● 011：内部触发 3（ITR3） ● 100：TI1 的边沿检测器（TI1F_ED） ● 101：滤波后的定时器输入 1（TI1FP1） ● 110：滤波后的定时器输入 2（TI2FP2） ● 111：外部触发输入（ETRF） 参见表 15-3。 *注意：这些位只有在不使用时才必须更改（例如 SMS=000），以避免在过渡时错误的边沿检测。*
位 3	OCCS：OCREF Clear 选择（OCREF clear selection） 该位用来选择 OCREF 清除源。 ● 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/下计数。 ● 010：编码器模式 2 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 ● 011：编码器模式 3 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的 边沿向上/下计数。 ● 100：复位模式 选中的触发输入（TRGI）的上升沿重新初始化计数器，并且产生一次更新寄存器。 ● 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（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 15-3 TIM1 内部触发连接

从定时器	ITR0（TS=000）	ITR1（TS=001）	ITR2（TS=010）	ITR3（TS=011）
TIM1	未连接	TIM2	未连接	TIM3

15.3.4 TIM1 DMA/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	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	Res：保留 必须保持复位值。
位 14	TDE：允许触发 DMA 请求（Trigger DMA request enable） ● 0：触发 DMA 请求禁止 ● 1：触发 DMA 请求允许
位 13	COMDE：COM DMA 请求使能（COM DMA request enable） ● 0：COM DMA 请求禁止 ● 1：COM DMA 请求允许
位 12	CC4DE：捕捉/比较 4 DMA 请求使能（Capture/Compare4 DMA request enable） ● 0：CC4 DMA 请求禁止 ● 1：CC4 DMA 请求允许
位 11	CC3DE：捕捉/比较 3 DMA 请求使能（Capture/Compare3 DMA request enable） ● 0：CC3 DMA 请求禁止 ● 1：CC3 DMA 请求允许
位 10	CC2DE：捕捉/比较 2 DMA 请求使能（Capture/Compare2 DMA request enable） ● 0：CC2 DMA 请求禁止 ● 1：CC2 DMA 请求允许
位 9	CC1DE：捕捉/比较 1 DMA 请求使能（Capture/Compare1 DMA request enable） ● 0：CC1 DMA 请求禁止 ● 1：CC1 DMA 请求允许
位 8	UDE：更新 DMA 请求使能（Update DMA request enable） ● 0：更新 DMA 请求禁止 ● 1：更新 DMA 请求允许
位 7	BIE：刹车中断使能（Break interrupt enable） ● 0：刹车中断禁止 ● 1：刹车中断允许
位 6	TIE：触发中断使能（Trigger interrupt enable） ● 0：触发中断禁止 ● 1：触发中断允许
位 5	COMIE：COM 中断使能（COM interrupt enable） ● 0：COM 中断禁止 ● 1：COM 中断允许

位 4	CC4IE：捕捉/比较 4 中断使能（Capture/Compare4 interrupt enable） ● 0：CC4 中断禁止 ● 1：CC4 中断允许
位 3	CC3IE：捕捉/比较 3 中断使能（Capture/Compare3 interrupt enable） ● 0：CC3 中断禁止 ● 1：CC3 中断允许
位 2	CC2IE：捕捉/比较 2 中断使能（Capture/Compare2 interrupt enable） ● 0：CC2 中断禁止 ● 1：CC2 中断允许
位 1	CC1IE：捕捉/比较 1 中断使能（Capture/Compare1 interrupt enable） ● 0：CC1 中断禁止 ● 1：CC1 中断允许
位 0	UIE：更新中断使能（Update interrupt enable） ● 0：更新中断禁止 ● 1：更新中断允许

15.3.5 TIM1 状态寄存器（TIM1_SR）

偏移地址：0x10

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														CC6IF	CC5IF
														rc_w0	rc_w0

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 31:18	Res：保留 必须保持复位值。
位 17	CC6IF：捕捉/比较 6 中断标志（Capture/Compare6 interrupt flag） 参见 CC5IF。
位 16	CC5IF：捕捉/比较 5 中断标志（Capture/Compare5 interrupt flag） 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。它由软件清'0'。 ● 0：无匹配发生 ● 1：TIM1_CNT 的值与 TIM1_CCR5 的值匹配。 当 TIM1_CCR5 的内容大于 TIM1_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC5IF 位变高。
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/Compare4 over capture flag） 参见 CC1OF。

位 11	CC3OF：捕捉/比较 3 重复捕捉标志（Capture/Compare3 over capture flag） 参见 CC1OF。
位 10	CC2OF：捕捉/比较 2 重复捕捉标志（Capture/Compare2 over capture flag） 参见 CC1OF。
位 9	CC1OF：捕捉/比较 1 重复捕捉标志（Capture/Compare1 over capture 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。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，则该位可由软件清'0'。 ● 0：无刹车事件产生 ● 1：刹车输入上检测到有效电平
位 6	TIF：触发器中断标志（Trigger interrupt flag） 当发生触发事件（当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿）时由硬件对该位置'1'。它由软件清'0'。 ● 0：无触发器事件产生 ● 1：触发中断等待响应
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 已被更新） 该位由硬件置'1'。它由软件清'0'。 ● 0：无 COM 事件产生 ● 1：COM 中断等待响应
位 4	CC4IF：捕捉/比较 4 中断标志（Capture/Compare4 interrupt flag） 参见 CC1IF。
位 3	CC3IF：捕捉/比较 3 中断标志（Capture/Compare3 interrupt flag） 参见 CC1IF。
位 2	CC2IF：捕捉/比较 2 中断标志（Capture/Compare2 interrupt flag） 参见 CC1IF。
位 1	CC1IF：捕捉/比较 1 中断标志（Capture/Compare1 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。它由软件清'0'。 ○ 0：无匹配发生 ○ 1：TIM1_CNT 的值与 TIM1_CCR1 的值匹配。 当 TIM1_CCR1 的内容大于 TIM1_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高。 ● 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置'1'，它由软件清'0'或通过读 TIM1_CCR1 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’。它由软件清’0’。 ● 0：无更新事件产生； ● 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： ○ 若 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、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 重新初始化时。 ○ 若 TIM1_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

15.3.6 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
CC6G	CC5G	Res						BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
w	w							w	w	w	w	w	w	w	w

位 15	CC6G：捕捉/比较 6 发生（Capture/Compare6 generation） 参见 CC1G 位。
位 14	CC5G：捕捉/比较 5 发生（Capture/Compare5 generation） 参见 CC1G 位。
位 13:8	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，由硬件自动清’0’。 ● 0：无动作 ● 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
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，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
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/Compare4generation） 参见 CC1G 位。
位 3	CC3G：捕捉/比较 3 发生（Capture/Compare3generation） 参见 CC1G 位。

位 2	CC2G：捕捉/比较 2 发生（Capture/Compare2generation） 参见 CC1G 位。
位 1	CC1G：捕捉/比较 1 发生（Capture/Compare1generation） 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，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 ○ 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIM1_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
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 TIM1_ARR 的值。

15.3.7 TIM1 捕捉/比较模式寄存器 1（TIM1_CCMR1）

偏移地址：0x18

复位值：0x0000

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

注意：同一个位在输出模式和输入模式下的功能是不同的。

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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	rw	rw	rw

输出比较模式:

位 15	OC2CE：输出比较 2 清 0 允许（Output Compare2 clear enable） 参见 OC1CE 位。
位 14:12	OC2M[2:0]：输出比较模式 2（Output Compare2 mode） 参见 OC1M 位。
位 11	OC2PE：输出比较 2 预装允许（Output Compare2 preload enable） 参见 OC1PE 位。
位 10	OC2FE：输出比较 2 快速允许（Output Compare2 fast enable） 参见 OC1FE 位。
位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/Compare2 selection） 该位定义通道的方向（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）。 *注意：CC2S 仅在通道关闭时（TIM1_CCER 寄存器的 CC2E=0）才是可写的。*
位 7	OC1CE：输出比较 1 清 0 允许（Output Compare1 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]：输出比较模式 1（Output Compare1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 ● 000：冻结 输出比较寄存器 TIM1_CCR1 与计数器 TIM1_CNT 间的比较，对 OC1REF 不起作用。 ● 001：匹配时设置通道 1 为有效电平 当计数器 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 o 在向上计数时，一旦 TIM1_CNT<TIM1_CCR1 时通道 1 为有效电平，否则为无效电平。 o 在向下计数时，一旦 TIM1_CNT>TIM1_CCR1 时通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。 ● 111：PWM 模式 2 o 在向上计数时，一旦 TIM1_CNT<TIM1_CCR1 时通道 1 为无效电平，否则为有效电平。 o 在向下计数时，一旦 TIM1_CNT>TIM1_CCR1 时通道 1 为有效电平，否则为无效电平。
位 3	OC1PE：输出比较 1 预装允许（Output Compare1 preload enable） ● 0：禁止 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起作用。 ● 1：开启 TIM1_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM1_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。
位 2	OC1FE：输出比较 1 快速使能（Output Compare1 fast enable） 该位用于加快 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，即使工作于触发器状态。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 ● 1：输入到触发器的有效沿的作用就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 模式时起作用。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare1 selection）

	这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC1 通道被配置为输出。 ● 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 ● 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 ● 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。

输入捕捉模式：

位 15:12	IC2F[3:0]：输入捕捉 2 滤波器（Input capture2 filter）
位 11:10	IC2PSC[1:0]：输入捕捉 2 预分频器（Input capture2 prescaler）
位 9:8	CC2S[1:0]：捕捉/比较 2 选择（Capture/Compare2 selection） 这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC2 通道被配置为输出。 ● 01：CC2 通道被配置为输入，IC2 映射在 TI2 上。 ● 10：CC2 通道被配置为输入，IC2 映射在 TI1 上。 ● 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。
位 7:4	IC1F[3:0]：输入捕捉 1 滤波器（Input capture1 filter） 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： ● 0000：无滤波器，以 $f_{SAMPLING}=f_{DTS}$ 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 3:2	IC1PSC[1:0]：输入捕捉 1 预分频器（Input capture1 prescaler） 这 2 位定义了 CC1 输入（IC1）的预分频系数。一旦 CC1E=0（TIM1_CCER 寄存器中），则预分频器复位。 ● 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。 ● 01：每 2 个事件触发一次捕获。 ● 10：每 4 个事件触发一次捕获。 ● 11：每 8 个事件触发一次捕获。

位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare1 Selection） 这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC1 通道被配置为输出。 ● 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 ● 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 ● 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。

15.3.8 TIM1 捕捉/比较模式寄存器 2（TIM1_CCMR2）

偏移地址：0x1C

复位值：0x0000

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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	rw	rw	rw

输出比较模式：

位 15	OC4CE：输出比较 4 清 0 允许（Output Compare4 clear enable） 参见 TIM1_CCMR1.OC1CE 位。
位 14:12	OC4M[2:0]：输出比较模式 4（Output Compare4 mode） 参见 TIM1_CCMR1.OC1M 位。
位 11	OC4PE：输出比较 4 预装允许（Output Compare4 preload enable） 参见 TIM1_CCMR1.OC1PE 位。
位 10	OC4FE：输出比较 4 快速允许（Output Compare4 fast enable） 参见 TIM1_CCMR1.OC1FE 位。
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare4 selection） 该位定义通道的方向（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 TIM1_SMCR 寄存器的 TS 位选择）。 *注意：CC4S 仅在通道关闭时（TIM1_CCER 寄存器的 CC4E=0）才是可写的。*
位 7	OC3CE：输出比较 3 清 0 允许（Output Compare3 clear enable） 参见 TIM1_CCMR1.OC1CE 位。
位 6:4	OC3M[2:0]：输出比较 3 模式（Output Compare3 mode） 参见 TIM1_CCMR1.OC1M 位。
位 3	OC3PE：输出比较 3 预装允许（Output Compare3 preload enable） 参见 TIM1_CCMR1.OC1PE 位。
位 2	OC3FE：输出比较 3 快速使能（Output Compare3 fast enable）

	参见 TIM1_CCMR1.OC1FE 位。
位 1:0	CC3S[1:0]：捕捉/比较 3 选择（Capture/Compare3 selection） 这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC3 通道被配置为输出。 ● 01：CC3 通道被配置为输入，IC3 映射在 TI3 上。 ● 10：CC3 通道被配置为输入，IC3 映射在 TI4 上。 ● 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。

输入捕捉模式：

位 15:12	IC4F[3:0]：输入捕捉 4 滤波器（Input capture4 filter） 参见 IC3F 位。
位 11:10	IC4PSC[1:0]：输入捕捉 4 预分频器（Input capture4 prescaler） 参见 IC3PSC 位。
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare4 Selection） 这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC4 通道被配置为输出。 ● 01：CC4 通道被配置为输入，IC4 映射在 TI4 上。 ● 10：CC4 通道被配置为输入，IC4 映射在 TI3 上。 ● 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。
位 7:4	IC3F[3:0]：输入捕捉 3 滤波器（Input capture3 filter） 参见 TIM1_CCMR1.IC1F 位。
位 3:2	IC3PSC[1:0]：输入捕捉 3 预分频器（Input capture3 prescaler） 参见 TIM1_CCMR1.IC1PSC 位。
位 1:0	CC3S[1:0]：捕捉/比较 3 选择（Capture/Compare3 Selection） 这 2 位定义通道的方向（输入/输出），及输入脚的选择： ● 00：CC3 通道被配置为输出； ● 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； ● 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； ● 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）。

15.3.9 TIM1 捕捉/比较使能寄存器（TIM1_CCER）

偏移地址：0x20

复位值：0x0000 0000

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res										CC6P	CC6E	Res		CC5P	CC5E
										rw	rw			rw	rw

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 31:22	Res：保留 必须保持复位值。
位 21	CC6P：比较 6 输出极性（Capture/Compare6 output polarity） 参见 CC1P 位。
位 20	CC6E：比较 6 输出使能（Capture/Compare6 output enable） 参见 CC1E 位。
位 19:18	Res：保留 必须保持复位值。
位 17	CC5P：比较 5 输出极性（Capture/Compare5 output polarity） 参见 CC1P 位。
位 16	CC5E：比较 5 输出使能（Capture/Compare5 output enable） 参见 CC1E 位。
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/Compare4 output polarity） 参见 CC1P 位。
位 12	CC4E：捕捉/比较 4 输出使能（Capture/Compare4 output enable） 参见 CC1E 位。
位 11	CC3NP：捕捉/比较 3 互补输出极性（Capture/Compare3 complementary output polarity） 参见 CC1NP 位。
位 10	CC3NE：捕捉/比较 3 互补输出使能（Capture/Compare3 complementary output enable） 参见 CC1NE 位。
位 9	CC3P：捕捉/比较 3 输出极性（Capture/Compare3 output polarity） 参见 CC1P 位。
位 8	CC3E：捕捉/比较 3 输出使能（Capture/Compare3 output enable） 参见 CC1E 位。
位 7	CC2NP：捕捉/比较 2 互补输出极性（Capture/Compare2 complementary output polarity） 参见 CC1NP 位。
位 6	CC2NE：捕捉/比较 2 互补输出使能（Capture/Compare2 complementary output enable） 参见 CC1NE 位。
位 5	CC2P：捕捉/比较 2 输出极性（Capture/Compare2 output polarity） 参见 CC1P 位。

位 4	CC2E：捕捉/比较 2 输出使能（Capture/Compare2 output enable） 参见 CC1E 位。
位 3	CC1NP：捕捉/比较 1 互补输出极性（Capture/Compare1 complementary output polarity） ● 0：OC1N 高电平有效 ● 1：OC1N 低电平有效 *注意：一旦 LOCK 级别（TIM_BDTR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00（通道配置为输出）则该位不能被修改。*
位 2	CC1NE：捕捉/比较 1 互补输出使能（Capture/Compare1 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/Compare1 output polarity） ● CC1 通道配置为输出： ○ 0：OC1 高电平有效 ○ 1：OC1 低电平有效 ● CC1 通道配置为输入： CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 ● 00：非反相/上升沿 电路作用于 TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 非反相 ● 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 ● 00：保留不用 ● 11：非反相/上升或下降沿 电路作用于 TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作），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/Compare1 output enable） ● CC1 通道配置为输出： ○ 0：关闭 OC1 禁止输出， OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 ○ 1：开启 OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 ● CC1 通道配置为输入： 本位用于决定是否一个定时器值的捕捉要装载到捕捉/比较寄存器 1（TIM1_CCR1）。 ○ 0：捕捉禁止 ○ 1：捕捉允许

表 15-4 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

控制位					输出状态 (1)	
1	X	0	0	0	输出禁止（与定时器断开）OCx=0，OCx_EN=0	输出禁止（与定时器断开）OCxN=0，OCxN_EN=0
		0	0	1	输出禁止（与定时器断开）OCx=0，OCx_EN=0	OCxREF+极性，OCxN=OCxREFxorCCxNP，OCxN_EN=1
		0	1	0	OCxREF+极性，OCx=OCxREFxorCCxP，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=OCxREFxorCCxNP，OCxN_EN=1
		1	1	0	OCxREF+极性，OCx=OCxREFxorCCxP，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). 如果一个通道的两个输出都没有使用(CCxE = CCxNE = 0)，那么 OISx, OISxN, CCxP 和 CCxNP 都必须清零。

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

15.3.10 TIM1 计数器（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）

15.3.11 TIM1 预分频器（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]：预分频器的值（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’。

15.3.12 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 包含了将要装载入实际的自动重装载寄存器的值。

15.3.13 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							UEV_SHIFT	REP[7:0]							
							rw	rw							

位 15:9	Res：保留 必须保持复位值。
位 8	UEV_SHIFT：第一次更新事件移位控制（Update event shift control for the first time） ● 0：禁用移位功能； ● 1：启用移位功能； *注意：可以控制第一次产生 UEV 事件的位置移位，移位的单位是一次 CNT 计数的最大值时间。*
位 7:0	REP[7:0]：重复计数器的值（Repetition counter value） 预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率（即周期性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

	新值只在下次周期更新事件发生时才起作用。这意味着在 PWM 模式中，（REP+1）对应着： ● 在边沿对齐模式下，PWM 周期的数目。 ● 在中央对齐模式下，PWM 半周期的数目。

15.3.14 TIM1 捕捉/比较寄存器 1（TIM1_CCR1）

偏移地址：0x34

复位值：0x0000

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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/Compare1 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 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。

15.3.15 TIM1 捕捉/比较寄存器 2（TIM1_CCR2）

偏移地址：0x38

复位值：0x0000

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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/Compare2 value） ● 若 CC2 通道配置为输出： CCR2 决定了装入当前捕获/比较 2 寄存器的值（预装载值）。 如果在 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当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 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。

15.3.16 TIM1 捕捉/比较寄存器 3（TIM1_CCR3）

偏移地址：0x3C

复位值：0x0000

说明：HK32M063C 和 HK32M066B 系列中，TIM1 连接预驱模块（Predriver），CH1~CH3 不支持输入捕获功能。

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/Compare3 value） ● 若 CC3 通道配置为输出： CCR3 决定了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIM1_CCMR3 寄存器（OC3PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 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。

15.3.17 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/Compare4 value） ● 若 CC4 通道配置为输出： CCR4 决定了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 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当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 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。

15.3.18 TIM1 刹车和死区寄存器（TIM1_BDTR）

偏移地址：0x44

复位值：0x0000

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

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 31:20	Res：保留 必须保持复位值。
位 19:16	BKF[3:0]：刹车输入信号滤波器（Break Filter） 此位域定义了刹车输入信号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，BRK 信号异步输入 ● 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 *注意：当 BDTR 寄存器 LOCK[1:0]位域被编程为锁定级别 1 时，这个位域不能被修改。*
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 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 ● 0：禁止 OC 和 OCN 输出或强制为空闲状态。 ● 1：如果设置了相应的使能位（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：刹车输入高电平有效 ● 1：刹车输入低电平有效
位 12	BKE：刹车使能（Break enable） ● 0：刹车输入禁止（BRK 和 CCS 时钟失效事件） ● 1：刹车输入允许（BRK 和 CCS 时钟失效事件）
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。 ● 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：运行模式下“空闲状态”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，置 OC/OCN 使能输出信号=1。
位 9:8	LOCK[1:0]：锁定设置（Lock configuration） 该位为防止软件错误而提供写保护。 ● 00：锁定关闭，寄存器无写保护。 ● 01：锁定级别 1，不能写入 TIM1_BDTR 寄存器的 BKF[3:0]、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） 这些位定义了插入互补输出之间的死区持续时间。 ● DTG[7:5]=0xx=>DT=DTG[7:0]x t_{dtg}，其中 $t_{dtg}=t_{DTS}$ ● DTG[7:5]=10x=>DT=（64+DTG[5:0]）x t_{dtg}，其中 $t_{dtg}=2x\ t_{DTS}$ ● DTG[7:5]=110=>DT=（32+DTG[4:0]）x t_{dtg}，其中 $t_{dtg}=8x\ t_{DTS}$ ● DTG[7:5]=111=>DT=（32+DTG[4:0]）x t_{dtg}，其中 $t_{dtg}=16x\ t_{DTS}$ 例如：如果 t_{DTS}=125ns（8MHz），死区时间可能的值为： ● 0 到 15875ns，步长为 125ns。 ● 16ns 到 31750ns，步长为 250ns。 ● 32µs 到 63µs，步长为 1µs。 ● 64µs 到 126µs，步长为 2µs。 *注意：当 LOCK 锁定级别设置为等级 1、2、3 时，此位域不能再被修改。*

15.3.19 TIM1 DMA 控制寄存器（TIM1_DCR）

偏移地址：0x48

复位值：0x0000

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

位 15:13	Res：保留 必须保持复位值。
位 12:8	DBL[4:0]：DMA 连续传送长度（DMA burst length） 这 5 位定义了 DMA 在连续模式下的传送长度（当对 TIM1_DMAR 寄存器进行读或写时，定时器则进行一次连续传送）。 ● 00000：1 次传输 ● 00001：2 次传输 ● 00010：3 次传输 ● …… ● 10001：18 次传输
位 7:5	Res：保留 必须保持复位值。
位 4:0	DBA[4:0]：DMA 基地址（DMA base address） 该位域定义 DMA 传输的基地址（通过 TIM1_DMAR 地址进行读/写访问时）。DBA 为从 TIM1_CR1 寄存器地址开始计算的偏移量。 ● 00000：TIM1_CR1 ● 00001：TIM1_CR2 ● 00010：TIM1_SMCR ● …… 示例：考虑以下传输：DBL=7 次传输，DBA=TIM1_CR1。这种情况下将向/从自 TIM1_CR1 地址开始的 7 个寄存器传输数据。

15.3.20 TIM1 全部传输时 DMA 地址（TIM1_DMAR）

偏移地址：0x4C

复位值：0x0000

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

位 15:0	DMAB[15:0]：DMA 并发（连续）传送寄存器（DMA register for burst accesses） 对 TIM1_DMAR 寄存器的读或写会导致对以下地址所在寄存器的访问： （TIM1_CR1 地址）+（DBA+DMA 索引）x4，其中： ● “TIM1_CR1 地址”是控制寄存器 1（TIM1_CR1）所在的地址。 ● “DBA”是 TIM1_DCR 寄存器中定义的基地址。 ● “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIM1_DCR 寄存器中定义的 DBL。 ● DMA 连续传送功能使用方法示例 ● 使用定时器 DMA 连续传送功能，将 TIM1_CCRx 寄存器（x = 2、3、4）的内容更新为 DMA 传输到 TIM1_CCRx 寄存器的内容。 ● 具体操作步骤如下： ● 将相应的 DMA 通道配置如下： o DMA 通道外设地址为 TIM1_DMAR 寄存器地址。 o DMA 通道存储器地址为包含要通过 DMA 传输到 TIM1_CCRx 寄存器的数据的 RAM 缓冲区地址。 o 要传输的数据量为 3（参见下文“注意”）。 o 禁止循环模式。 ● 配置 TIM1_DCR 寄存器的 DBA 和 DBL 位如下： o DBL = 3 次传输，DBA = 0xE。 ● 使能 TIM1 更新 DMA 请求（将 TIM1_DIER.UDE 置 1）。 ● 使能 TIM1（将 TIM1_CR1.CEN 置 1）。 ● 使能 DMA 通道。 *注意：本例适用于每个 TIM1_CCRx 寄存器只更新一次的情况。如果每个 CCRx 寄存器要更新两次，则要传输的数据量应为 6。* 下面以包含 data1、data2、data3、data4、data5 和 data6 的 RAM 缓冲区为例说明。 数据将按照如下方式传输到 TIM1_CCRx 寄存器： 1. 在第一个更新 DMA 请求期间，data1 传输到 CCR2，data2 传输到 CCR3，data3 传输到 CCR4； 2. 在第二个更新 DMA 请求期间，data4 传输到 CCR2，data5 传输到 CCR3，data6 传输到 CCR4。

15.3.21 TIM1 选项寄存器（TIM1_OR）

偏移地址：0x50

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								VCOMPO4	Res		VCOMPO3	VCOMPO2	VCOMPO1	Res	
								rw			rw	rw	rw		

位 15:8	Res：保留 必须保持复位值。
位 7	VCOMPO4：刹车信号重映射到 COMP4_OUT（Break from Comparer4 output enable） 参见 VCOMPO1 位。

位 6:5	Res：保留 必须保持复位值。
位 4	VCOMPO3：刹车信号重映射到 COMP3_OUT（Break from Comparer3 output enable） 参见 VCOMPO1 位。
位 3	VCOMPO2：刹车信号重映射到 COMP2_OUT（Break from Comparer2 output enable） 参见 VCOMPO1 位。
位 2	VCOMPO1：刹车信号重映射到 COMP1_OUT（Break from Comparer1 output enable） ● 0：刹车信号不重映射到 COMP1_OUT ● 1：刹车信号重映射到 COMP1_OUT
位 1:0	Res：保留 必须保持复位值。

15.3.22 TIM1 捕获/比较模式寄存器 3（TIM1_CCMR3）

偏移地址：0x54

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC6CE	OC6M[2:0]			OC6PE	OC6FE	Res		OC5CE	OC5M[2:0]			OC5PE	OC5FE	Res	
rw	rw			rw	rw			rw	rw			rw	rw		

位 15	OC6CE：输出比较 6 清 0 允许（Output Compare6 clear enable） 参见 OC1CE 位。
位 14:12	OC6M[2:0]：输出比较模式 6（Output Compare6 mode） 参见 OC1M[2:0]位。
位 11	OC6PE：输出比较 6 预装允许（Output Compare6 preload enable） 参见 OC1M[2:0]位。
位 10	OC6FE：输出比较 6 快速允许（Output Compare6 fast enable） 参见 OC1FE 位。
位 9:8	Res：保留 必须保持复位值。
位 7	OC5CE：输出比较 5 清 0 允许（Output Compare5 clear enable） 参见 OC1CE 位。
位 6:4	OC5M[2:0]：输出比较模式 5（Output Compare5 mode） 参见 OC1M[2:0]位。
位 3	OC5PE：输出比较 5 预装允许（Output Compare6 preload enable） 参见 OC1PE 位。
位 2	OC5FE：输出比较 5 快速允许（Output Compare6 fast enable） 参见 OC1FE 位。
位 1:0	Res：保留

	必须保持复位值。

15.3.23 TIM1 捕获/比较寄存器 5（TIM1_CCR5）

偏移地址：0x58

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR5[15:0]															
rw															

位 15:0	CCR5[15:0]：捕捉/比较通道 5 的值（Capture/Compare5 value） CCR5 决定了装入当前捕获/比较 5 寄存器的值（预装载值）。 如果在 TIM1_CCMR3 寄存器（OC5PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 5 寄存器中。 当前捕获/比较寄存器参与同计数器 TIM1_CNT 的比较，并在 OC5 上产生输出信号。

15.3.24 TIM1 捕获/比较寄存器 6 （TIM1_CCR6）

偏移地址：0x5C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR6[15:0]															
rw															

位 15:0	CCR6[15:0]：捕捉/比较通道 6 的值（Capture/Compare6 value） CCR6 决定了装入当前捕获/比较 6 寄存器的值（预装载值）。 如果在 TIM1_CCMR3 寄存器（OC6PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 6 寄存器中。 当前捕获/比较寄存器参与同计数器 TIM1_CNT 的比较，并在 OC6 上产生输出信号。

15.3.25 TIM1 比较寄存器 1（TIM1_CMPR1）

偏移地址：0x70

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPR1[15:0]															
rw															

位 15:0	CMPR1[15:0]：比较通道 1 的值（Compare1 value） 通道 1 在 PWM2 模式下，当计数器向下计数时，OC1 可选择 CMPR1 与 ARR 比较，控制通道 1 的脉宽。 CMPR1 决定了装入当前比较 1 寄存器的值（预装载值）。

15.3.26 TIM1 比较寄存器 2（TIM1_CMPR2）

偏移地址：0x74

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPR2[15:0]															
rw															

位 15:0	CMPR2[15:0]：比较通道 2 的值（Compare2 value） 通道 2 在 PWM2 模式下，当计数器向下计数时，OC2 可选择 CMPR2 与 ARR 比较，控制通道 2 的脉宽。 CMPR2 决定了装入当前比较 2 寄存器的值（预装载值）。

15.3.27 TIM1 比较寄存器 3（TIM1_CMPR3）

偏移地址：0x78

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPR3[15:0]															
rw															

位 15:0	CMPR3[15:0]：比较通道 3 的值（Compare3 value） 通道 3 在 PWM2 模式下，当计数器向下计数时，OC3 可选择 CMPR3 与 ARR 比较，控制通道 3 的脉宽。 CMPR3 决定了装入当前比较 3 寄存器的值（预装载值）。

15.3.28 TIM1 比较寄存器 4 （TIM1_CMPR4）

偏移地址：0x7C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPR4[15:0]															
rw															

位 15:0	CMPR4[15:0]：比较通道 4 的值（Compare4 value） 通道 4 在 PWM2 模式下，当计数器向下计数时，OC4 可选择 CMPR4 与 ARR 比较，控制通道 4 的脉宽。 CMPR4 决定了装入当前比较 4 寄存器的值（预装载值）。

15.3.29 TIM1 死区发生寄存器（TIM1_DTGR）

偏移地址：0x80

复位值：0x8000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTG2_EN	Res							DTG2[7:0]							
rw								rw							

位 15	DTG2_EN：PWM 后死区时间使能（Dead-time2 generator enable） ● 1：PWM 后死区时间使能。 ● 0：PWM 后死区时间无效，死区时间由 DTG[7:0]配置。
位 14:8	Res：保留 必须保持复位值。
位 7:0	DTG2[7:0]：后死区发生器设置（Dead-time2 generator setup） 当设置 DTG2_EN 后，DTG2[7:0]可以设置 OCx 上升沿相较于 OCxREF 的延时。

15.3.30 TIM1 数据搬移控制寄存器（TIM1_DATA_TRANSFER_CR）

偏移地址：0x84

复位值：0x0001

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res											PLIF	PLIE	PLE	DES_ADDR[1:0]	
											rc_w0	rw	rw	rw	

位 15:5	Res：保留 必须保持复位值。
位 4	PLIF：内部数据预装载完成中断标志（Pre-load interrupt flag） ● 0：数据预装载未完成 ● 1：数据预装载已完成
位 3	PLIE：内部数据预装载完成中断使能（Pre-load interrupt enable） ● 0：禁用 ● 1：启用
位 2	PLE：内部数据预装载使能（Pre-load enable） ● 0：禁用 ● 1：启用
位 1:0	DES_ADDR[1:0]：传输目标地址（Destination address） ● 00：保留 ● 01：目的地址为 CCR1 ● 10：目的地址为 CCR2 ● 11：目的地址为 CCR3

15.3.31 TIM1 搬移数据寄存器 1（TIM1_TRANSFER_DATA1）

偏移地址：0x88

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR_PRE1[15:0]															
rw															

位 15:0	CCR_PRE1[15:0]：第一次装载的值（Value for the first load）

15.3.32 TIM1 搬移数据寄存器 2（TIM1_TRANSFER_DATA2）

偏移地址：0x8C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR_PRE2[15:0]															
rw															

位 15:0	CCR_PRE2[15:0]：第二次装载的值（Value for the second load）

15.3.33 TIM1 的外部触发源选择寄存器（TIM1_ETR_SEL）

偏移地址：0x90

复位值：0x0000

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

位 15:3	Res：保留 必须保持复位值。
位 2:0	ETR_SEL[2:0]:外部触发源选择（External trigger source selection） ● 0：外部 pin 输入 ● 1：COMP1 作为 ETR 输入 ● 2：COMP2 作为 ETR 输入 ● 3：COMP3 作为 ETR 输入 ● 4：COMP4 作为 ETR 输入

16 通用定时器（TIM2 和 TIM3）

通用定时器是一个由可编程预分频器驱动的 16/32 位（TIM2 是 32 位，TIM3 是 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，参见“16.2.15 定时器同步”。

表 16-1 TIM2/3 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
		$f_{TIMxCLK}$=48MHz	-	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}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48MHz	-	1365.3	-	μs
	当选择内部时钟时，32 位计数器的时钟周期		-	2^{32}	-	$t_{TIMxCLK}$
		$f_{TIMxCLK}$=48MHz	-	178.957	-	s

16.1 TIM2 和 TIM3 主要功能

通用 TIMx（TIM2、TIM3）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
- 16/32 位（TIM2 是 32 位，TIM3 是 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/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
- 支持针对定位的增量（正交）编码器和霍尔传感器电路

- 触发输入作为外部时钟或者按周期的电流管理信号

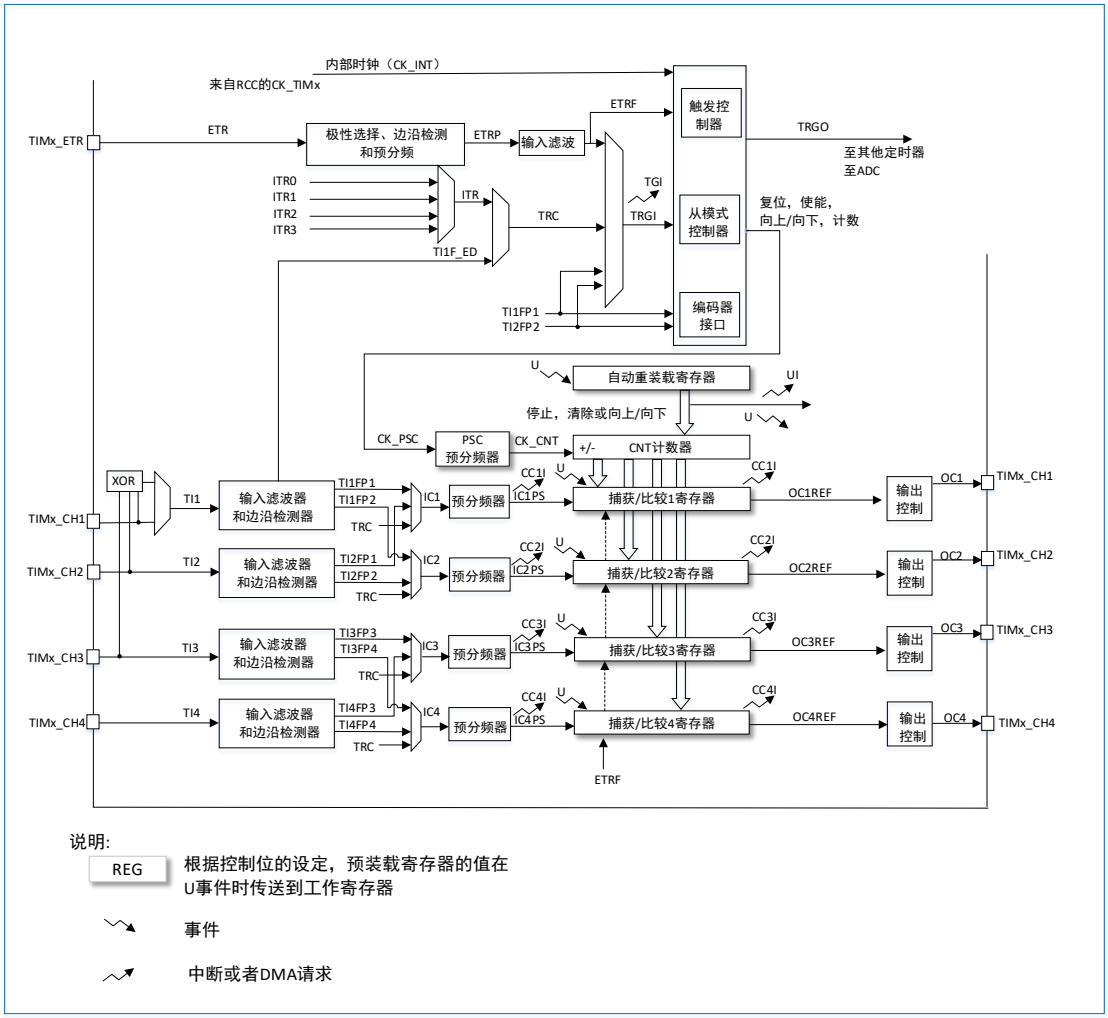

图 16-1 通用定时器框图

16.2 TIM2 和 TIM3 功能描述

16.2.1 时基单元

可编程通用定时器的主要部分是一个 16/32 位计数器（TIM3 是 16 位，TIM2 是 32 位）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（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 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 TIMx_CR1 寄存器中的 UDIS 位等于’0’时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能

位（CEN）时，CK_CNT 才有效。

注意：计数器使能信号 CNT_EN 是在 CEN 的一个时钟周期才后被设置。

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 16 位寄存器（TIMx_PSC）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。

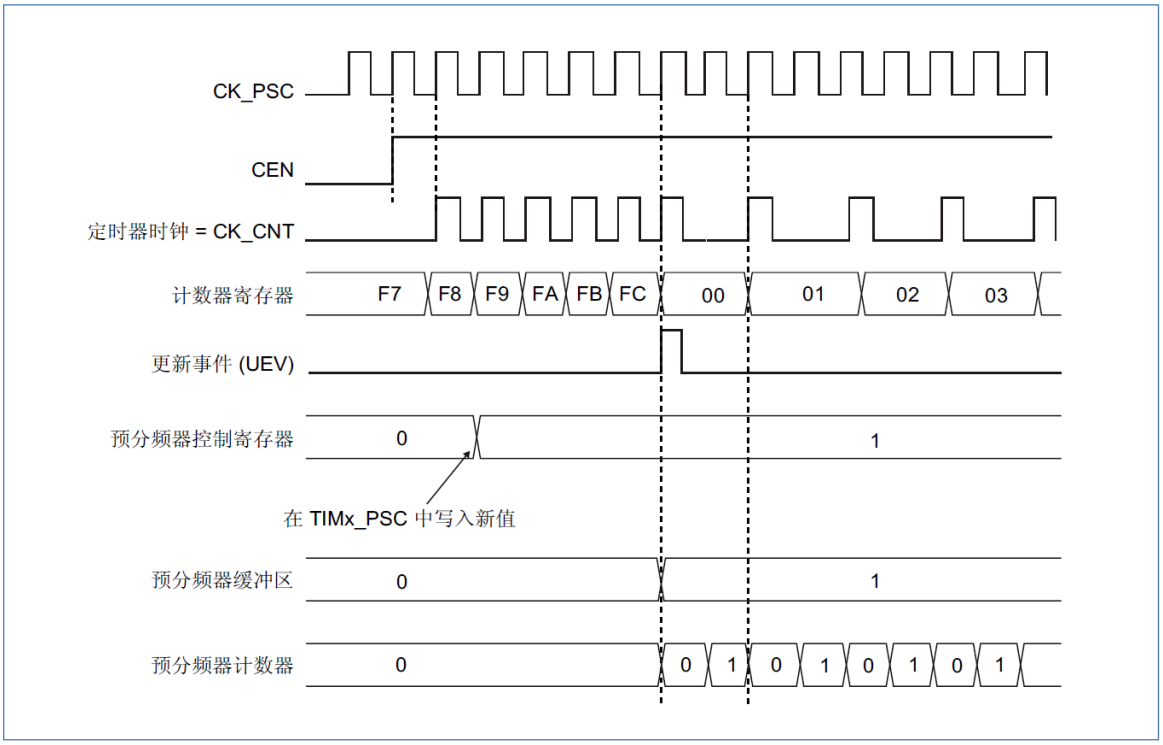

图 16-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

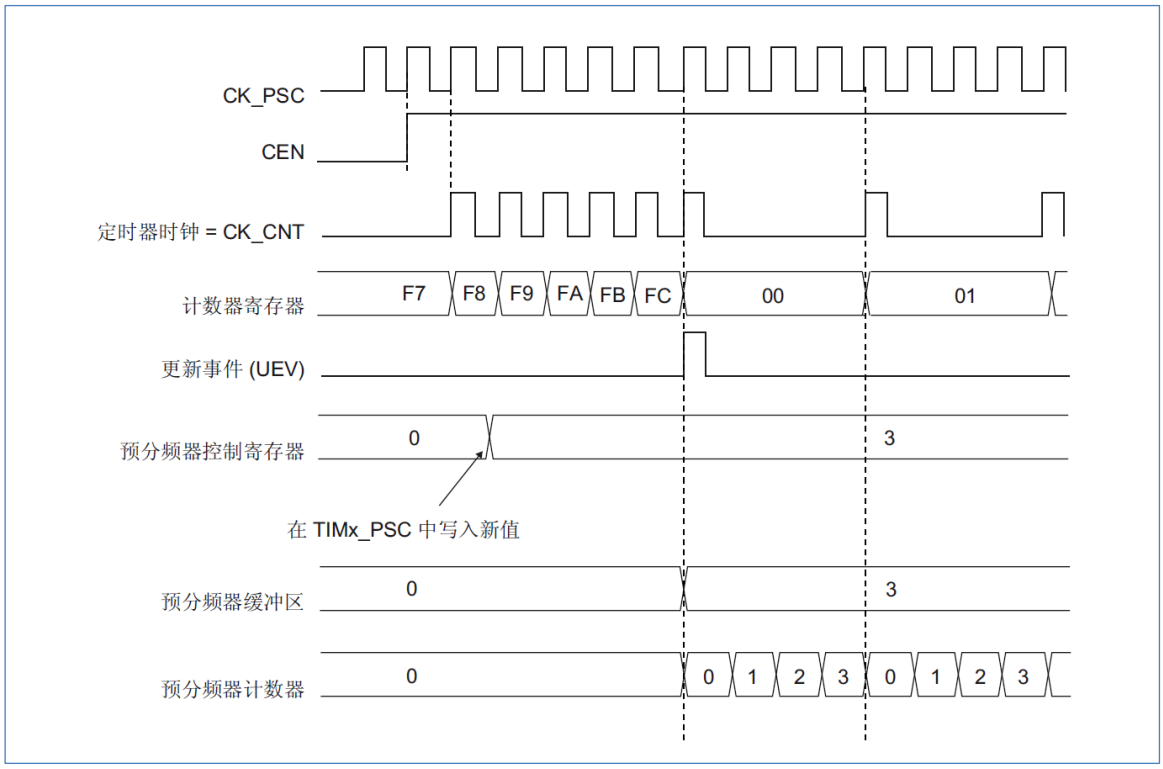

图 16-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

16.2.2 计数器模式

16.2.2.1 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（TIMx_ARR 计数器的内容），然后重新从 0 开始计数并且产生一个计数器溢出事件。

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

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也清零（但预分频系数不变）。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断或 DMA 请求）；这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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

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

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

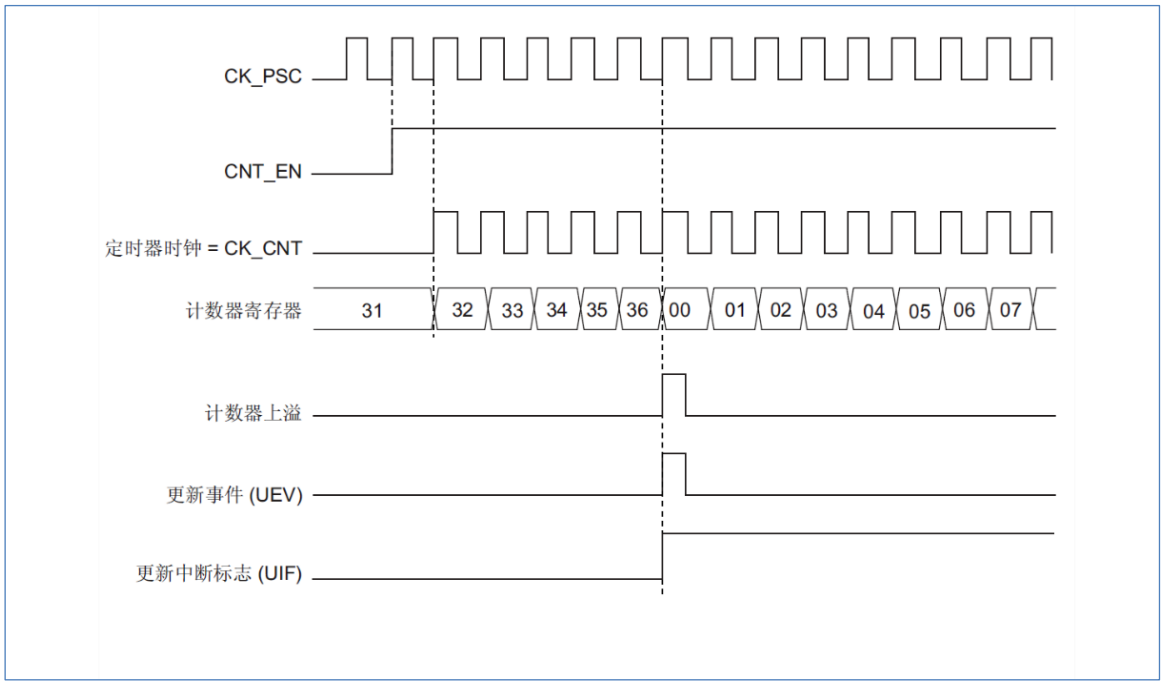

图 16-4 计数器时序图，内部时钟分频因子为 1

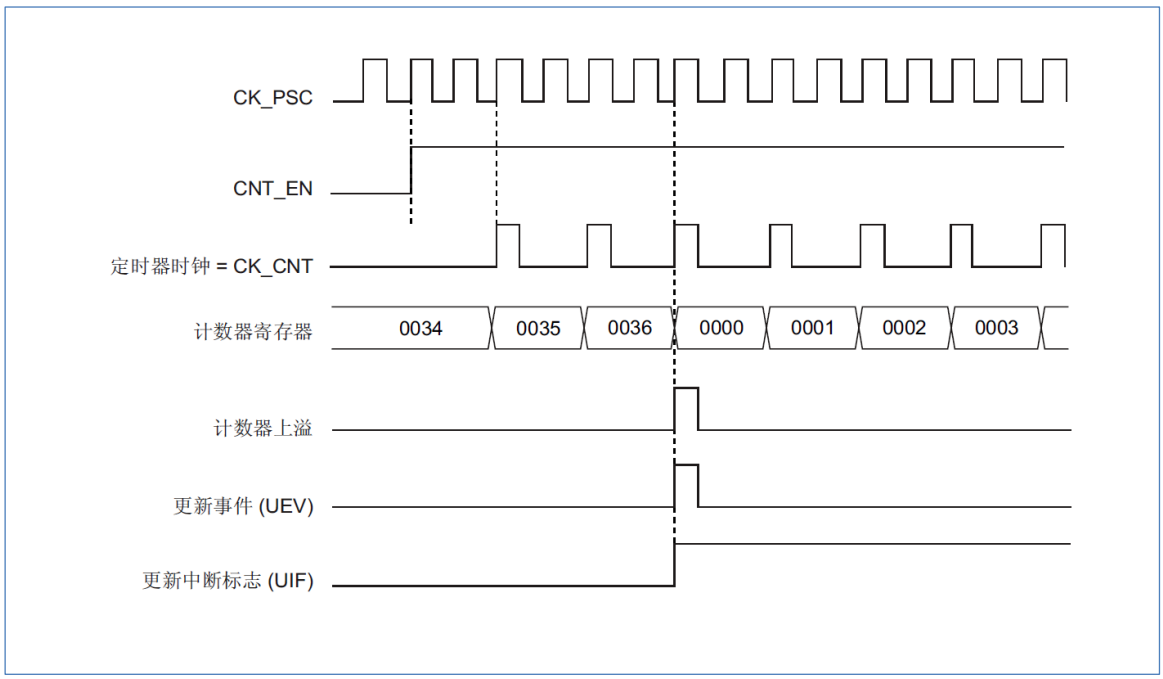

图 16-5 计数器时序图，内部时钟分频因子为 2

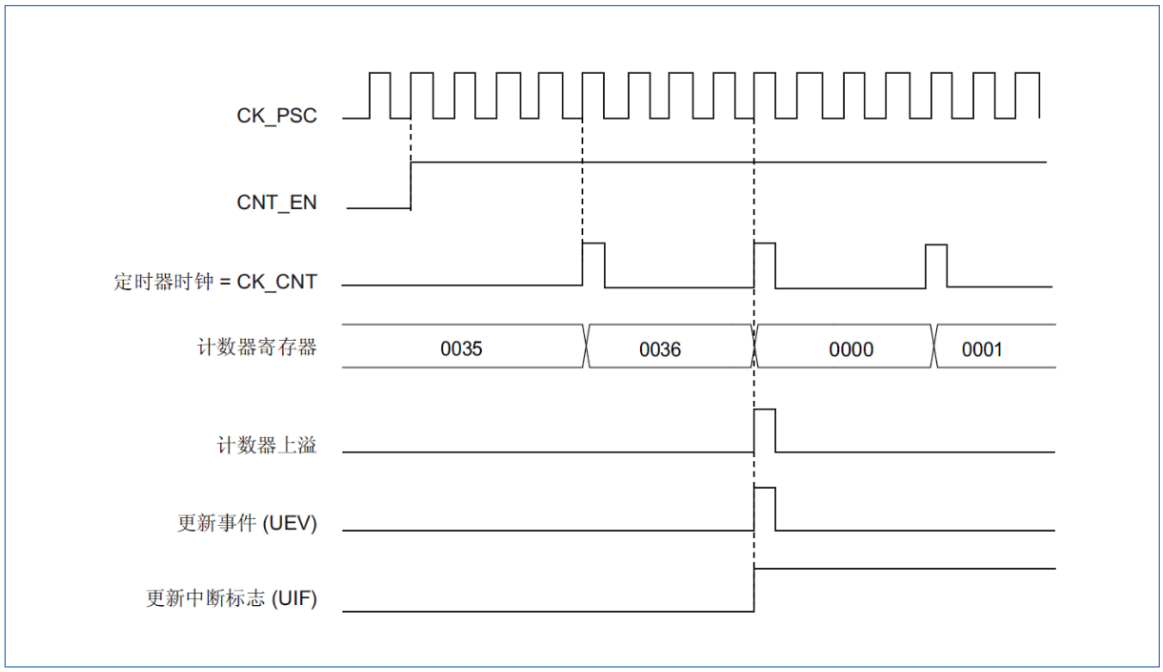

图 16-6 计数器时序图，内部时钟分频因子为 4

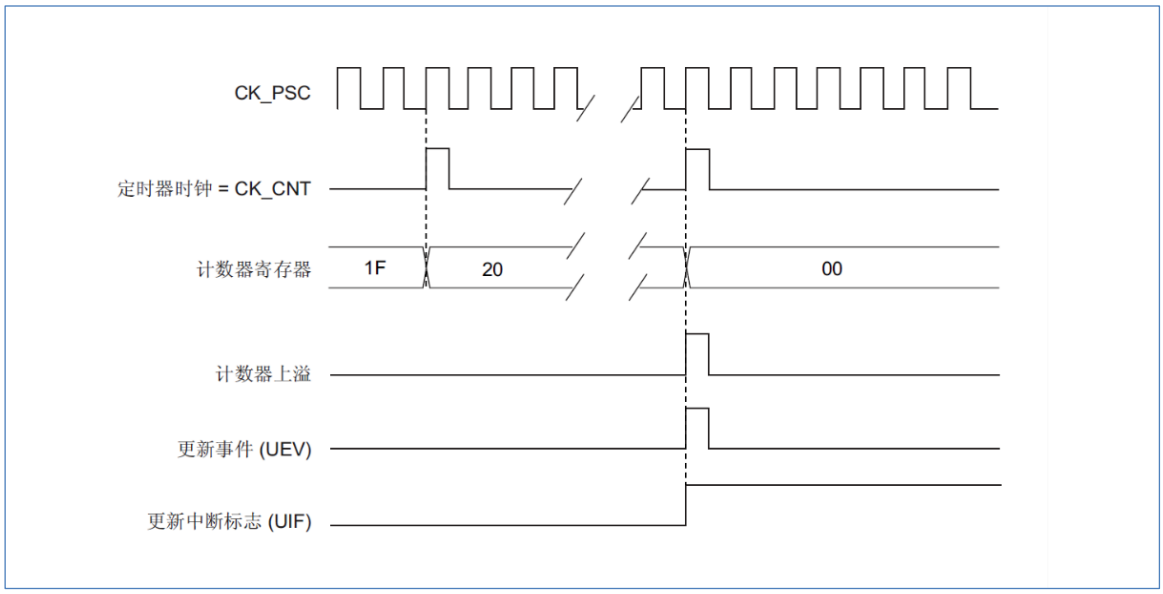

图 16-7 计数器时序图，内部时钟分频因子为 N

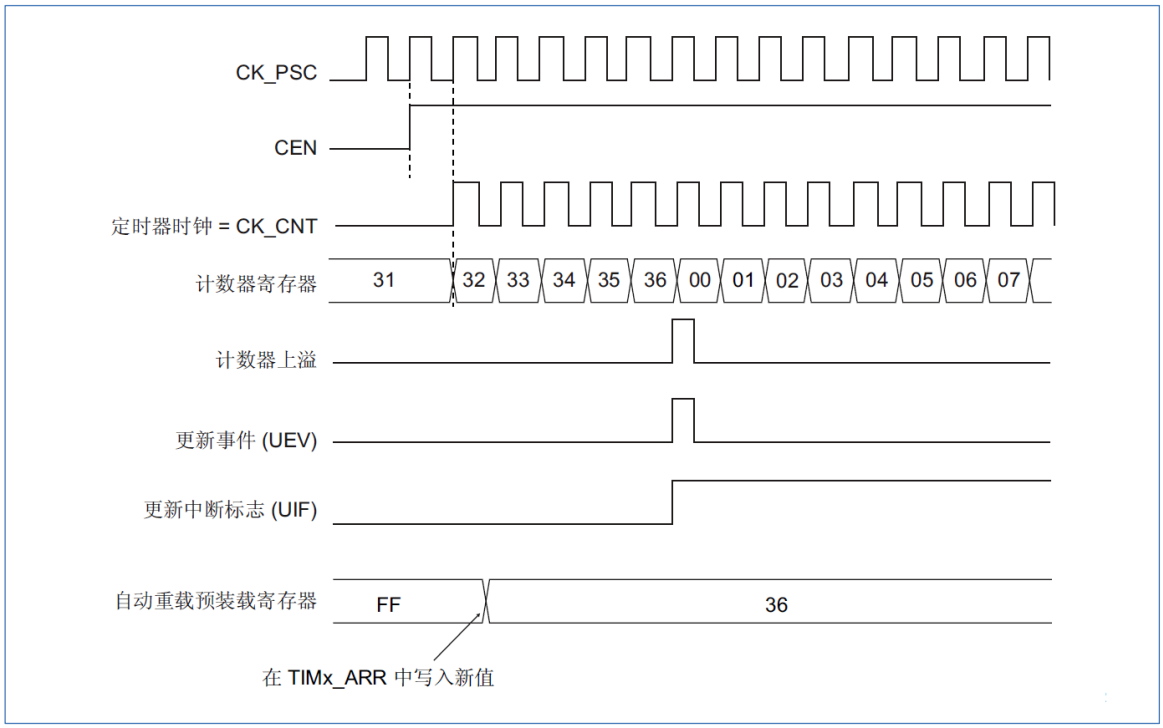

图 16-8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

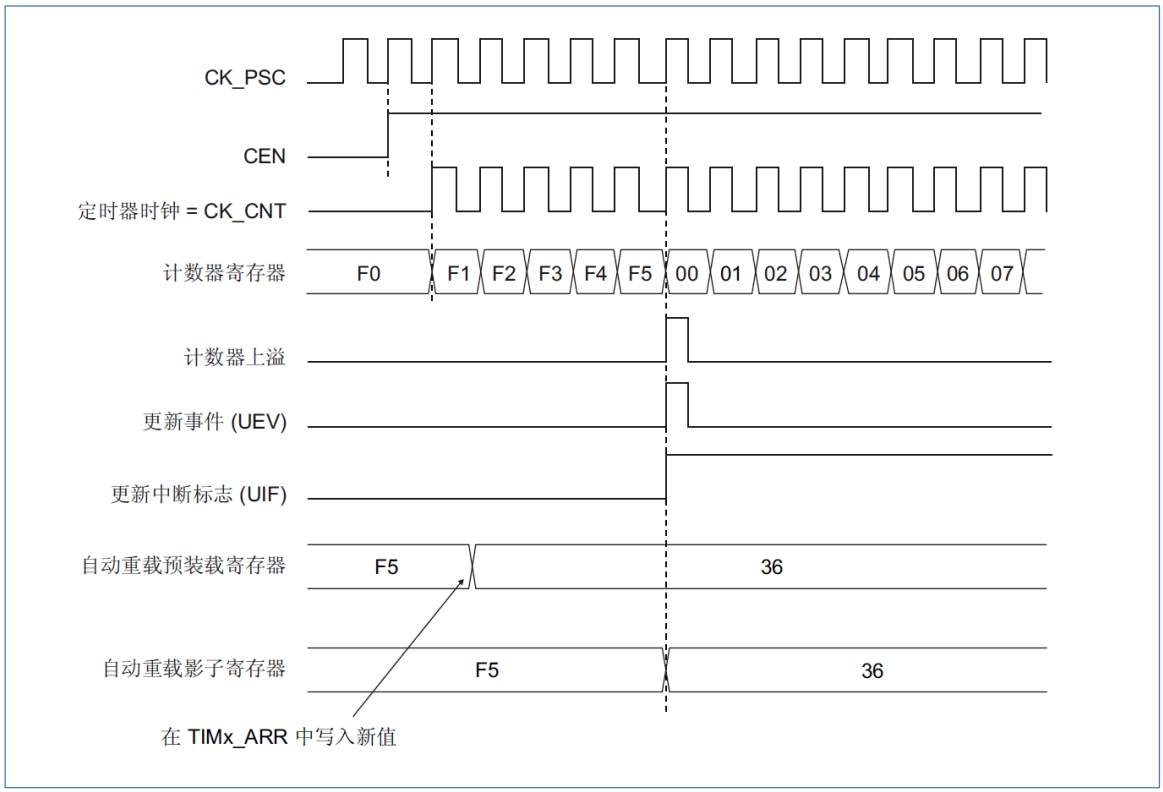

图 16-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

16.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（TIMx_ARR 计数器的值）开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

每次计数器溢出时可以产生更新事件，在 TIMx_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位，也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更

新影子寄存器。因此 UDIS 位被清为'0'之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，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 TIMx_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

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

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

注意：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

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

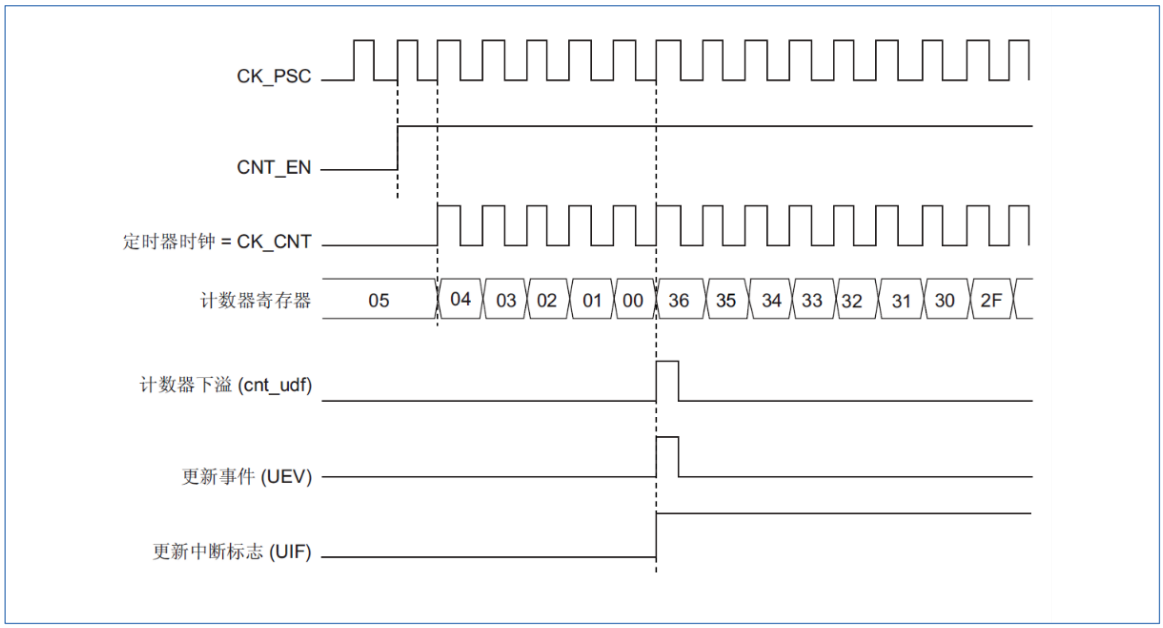

图 16-10 计数器时序图，内部时钟分频因子为 1

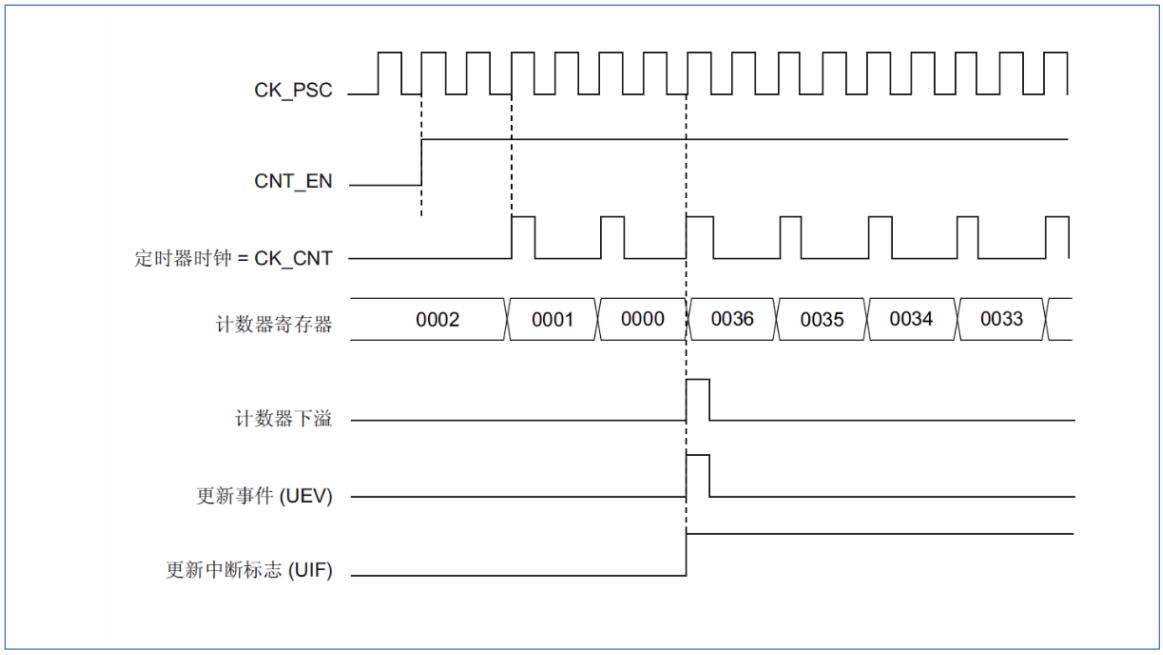

图 16-11 计数器时序图，内部时钟分频因子为 2

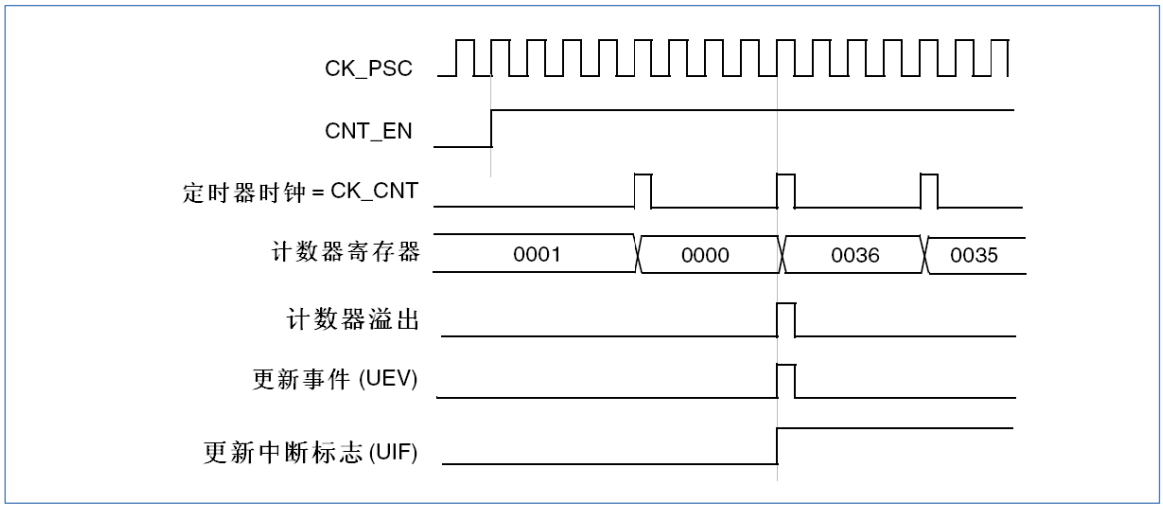

图 16-12 计数器时序图，内部时钟分频因子为 4

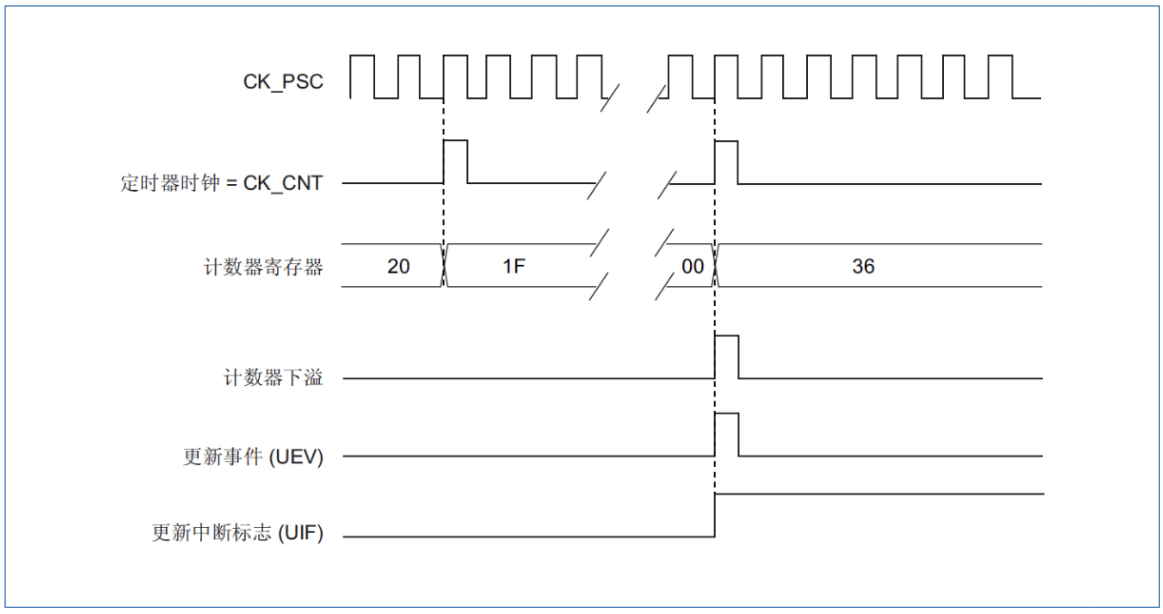

图 16-13 计数器时序图，内部时钟分频因子为 N

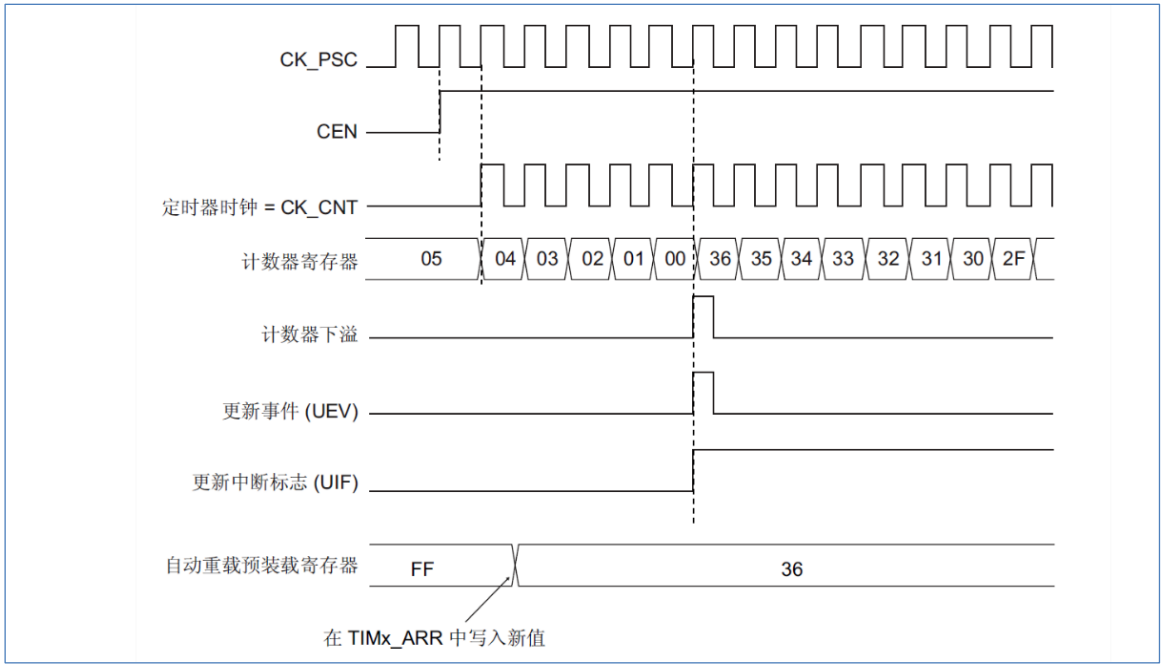

图 16-14 计数器时序图，当没有使用重复计数器时的更新事件

16.2.2.3 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（TIMx_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 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

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

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

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

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

注意：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：

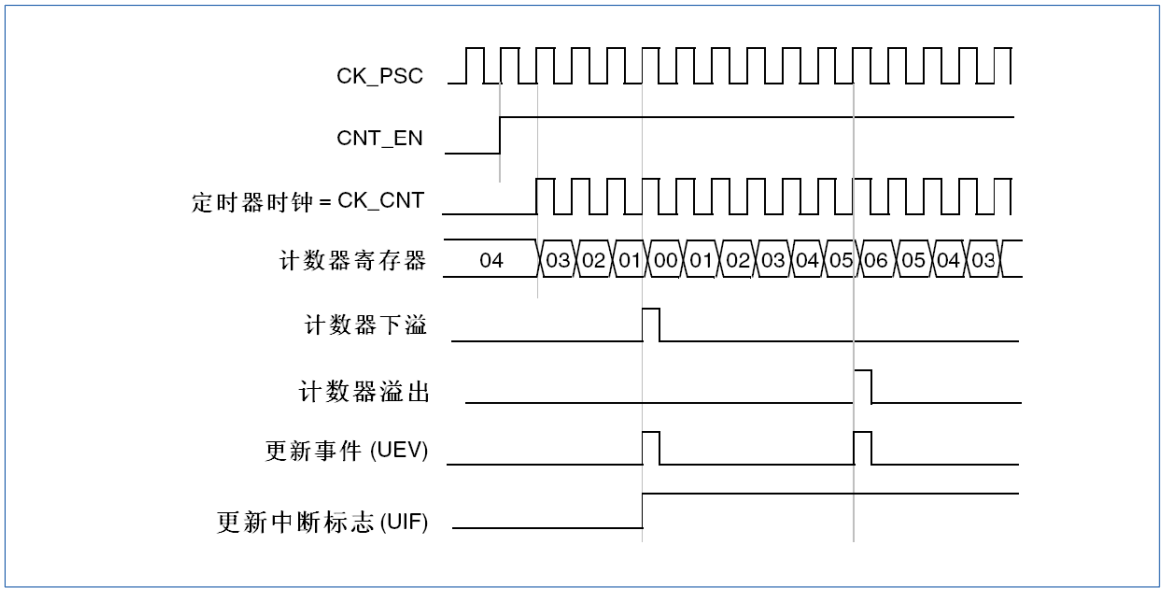

图 16-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6（这里使用了中央对齐模式 1）

图 16-15 使用了中央对齐模式 1（参见“16.3.1 TIMx 控制寄存器 1（TIMx_CR1）（x=2..3）”）。

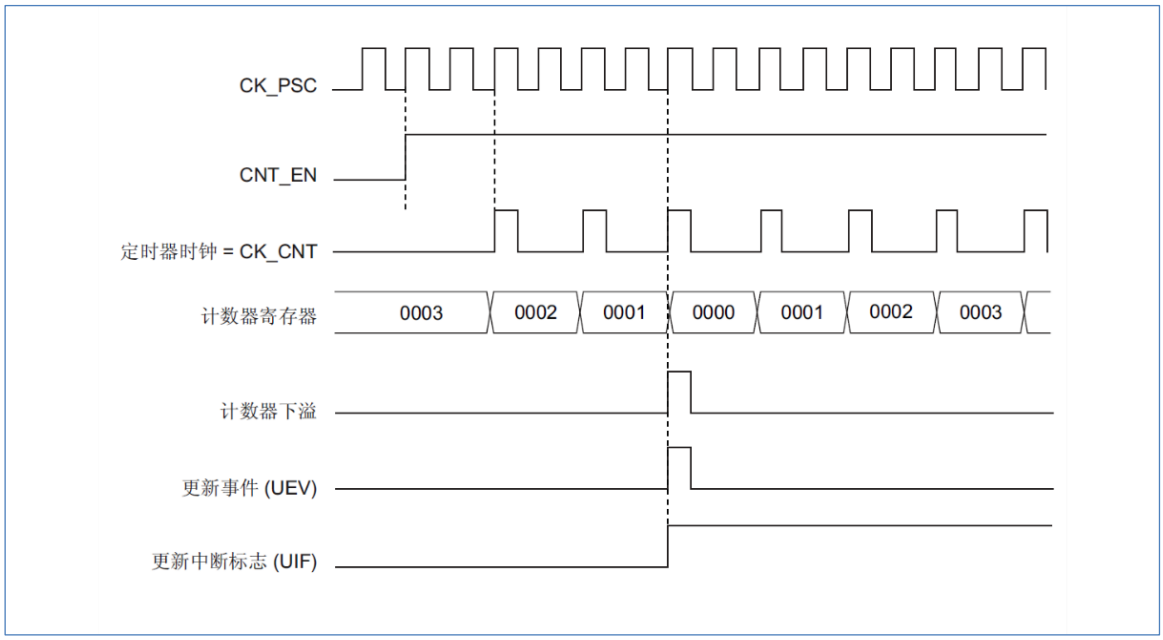

图 16-16 计数器时序图，内部时钟分频因子为 2（这里使用了中央对齐模式 1）

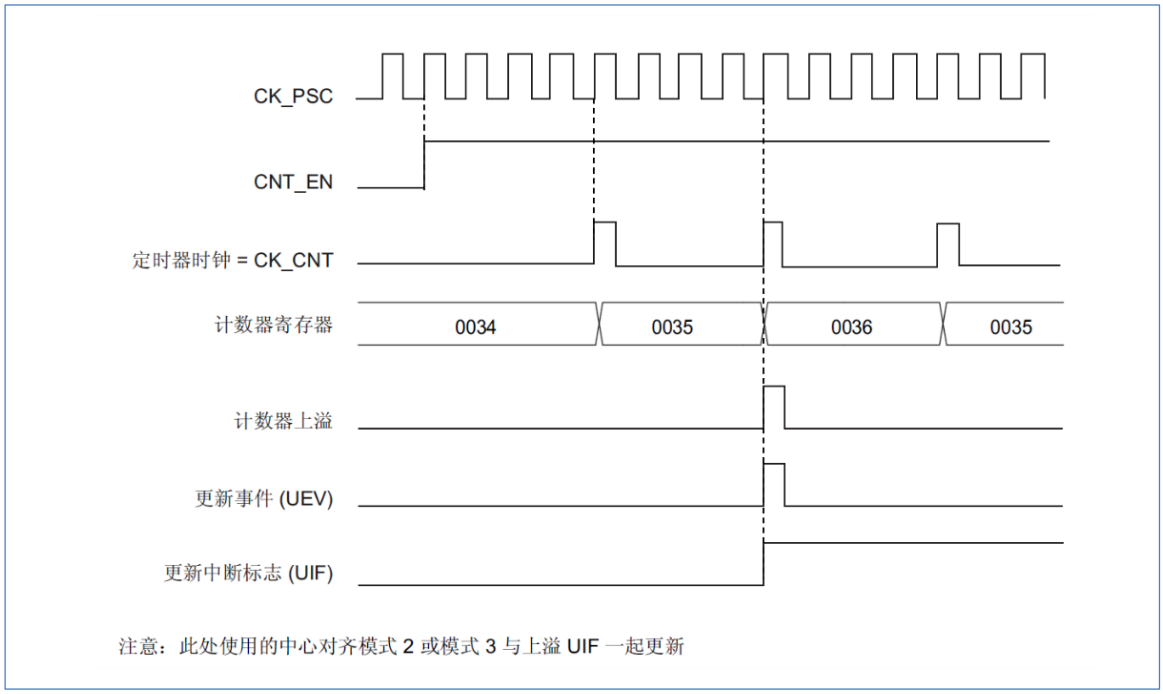

图 16-17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36（这里使用了中央对齐模式 2 或 3）

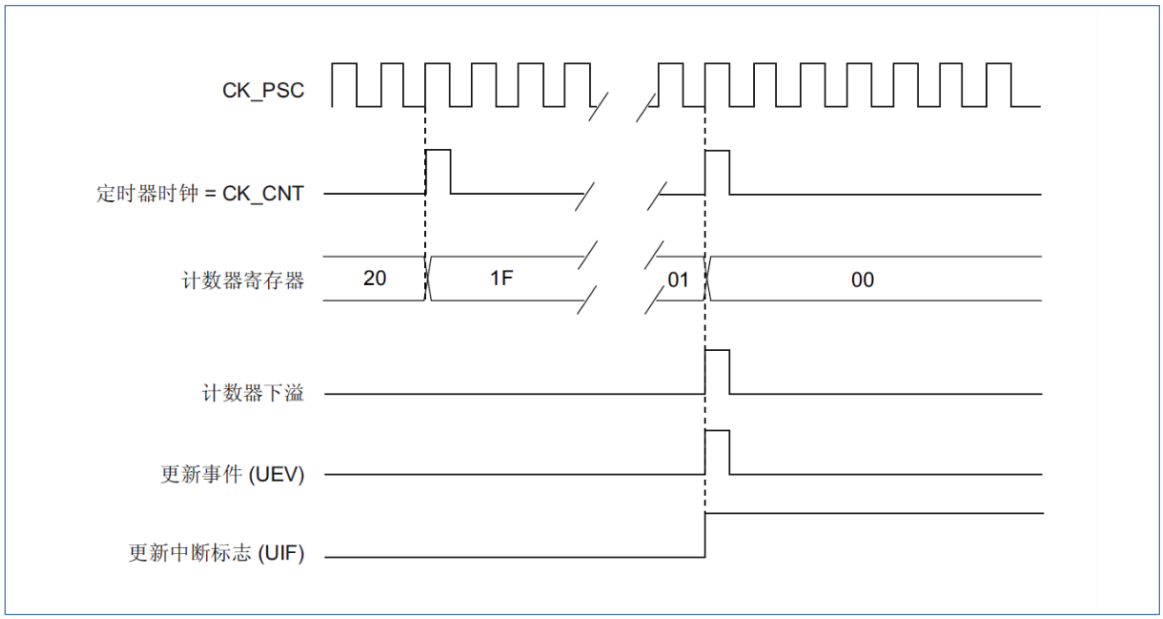

图 16-18 计数器时序图，内部时钟分频因子为 N（这里使用了中央对齐模式 1）

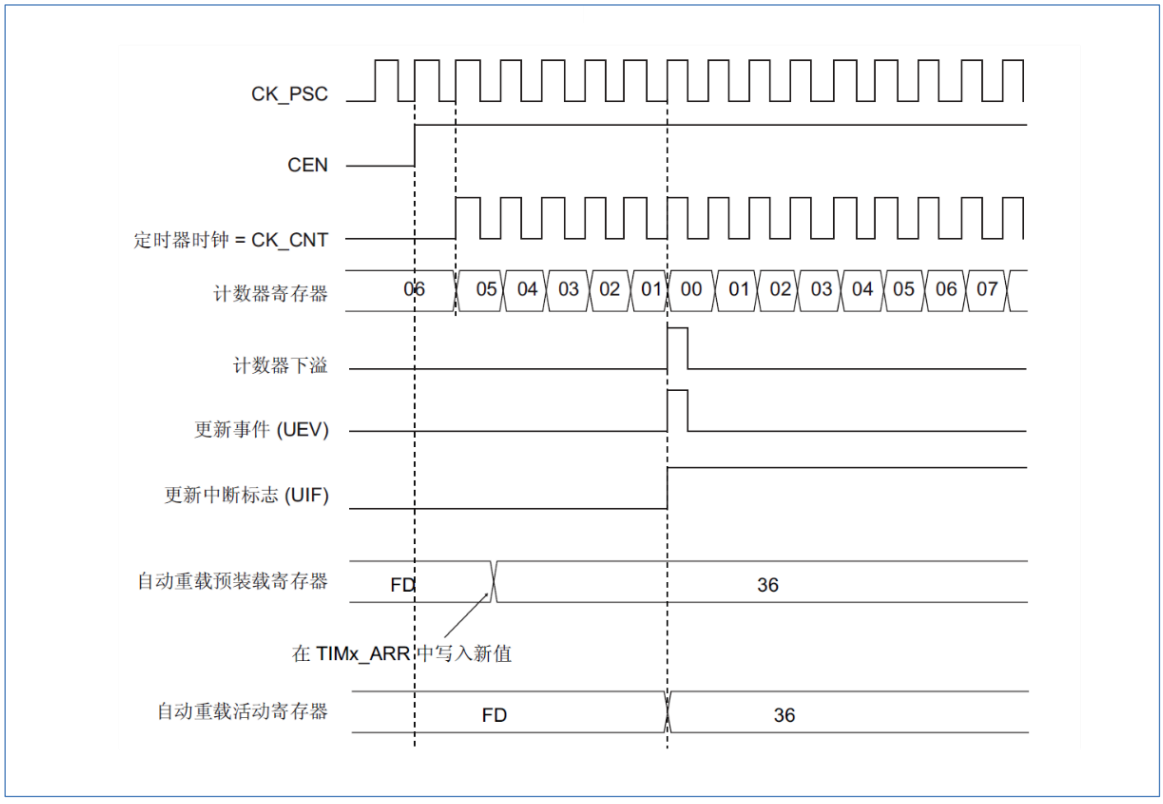

图 16-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

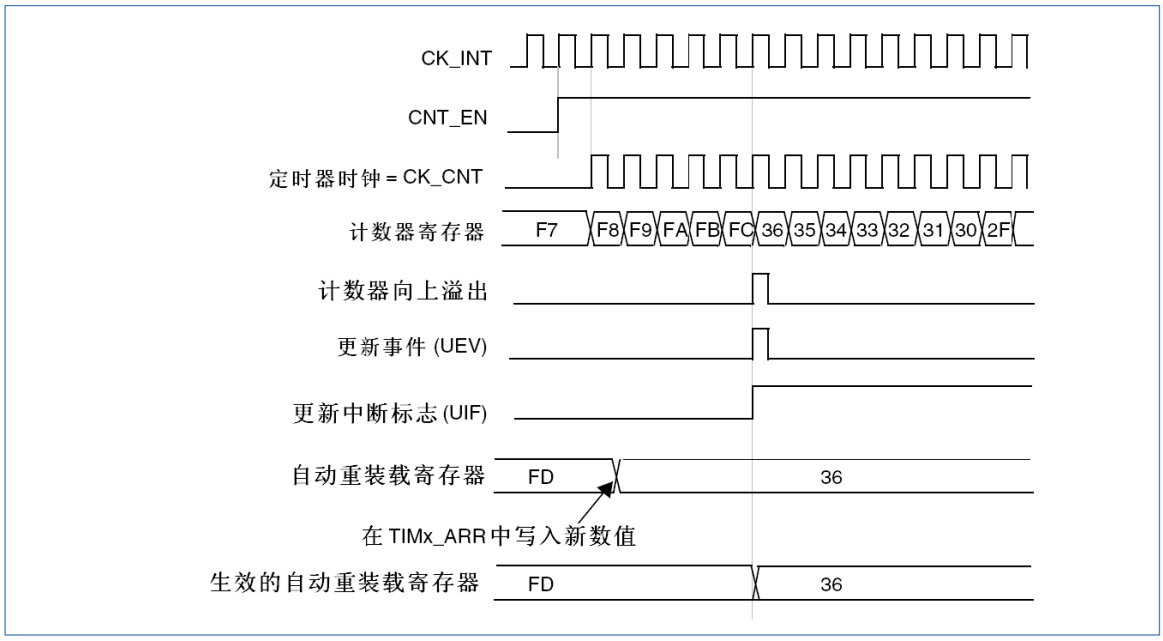

图 16-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

16.2.3 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。参见“16.2.15 定时器同步”。

16.2.3.1 内部时钟源（CK_INT）

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

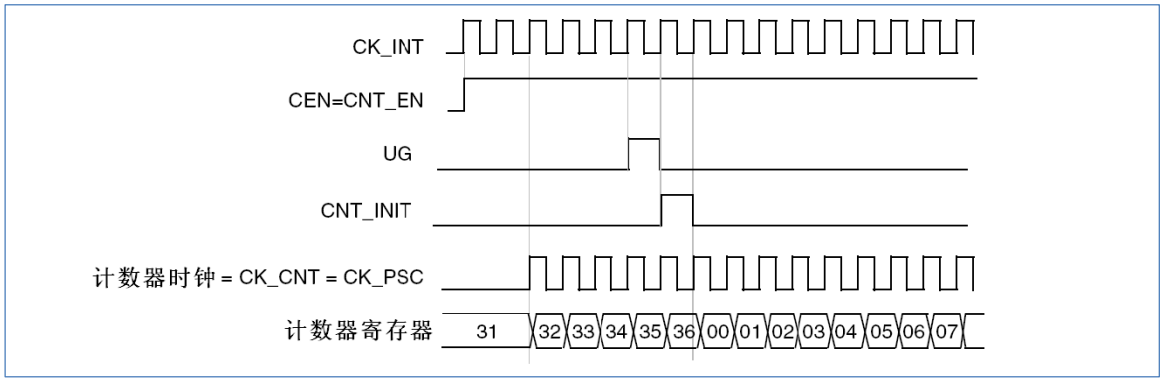

图 16-21 一般模式下的控制电路，内部时钟分频因子为 1

16.2.3.2 外部时钟源模式 1

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

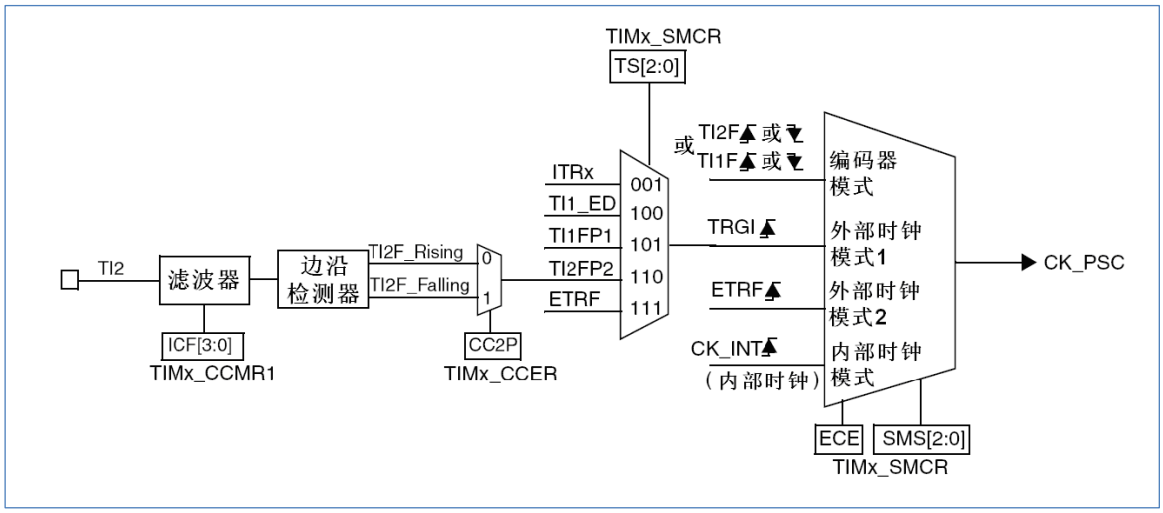

图 16-22 TI2 外部时钟连接例子

例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

1. 配置 TIMx_CCMR1 寄存器 CC2S='01'，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽（如果不需要滤波器，保持 IC2F=0000）。

注意：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 TIMx_CCER 寄存器的 CC2P='0'，选定上升沿极性。
4. 配置 TIMx_SMCR 寄存器的 SMS='111'，选择定时器外部时钟模式 1。
5. 配置 TIMx_SMCR 寄存器中的 TS='110'，选定 TI2 作为触发输入源。
6. 设置 TIMx_CR1 寄存器的 CEN='1'，启动计数器。

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

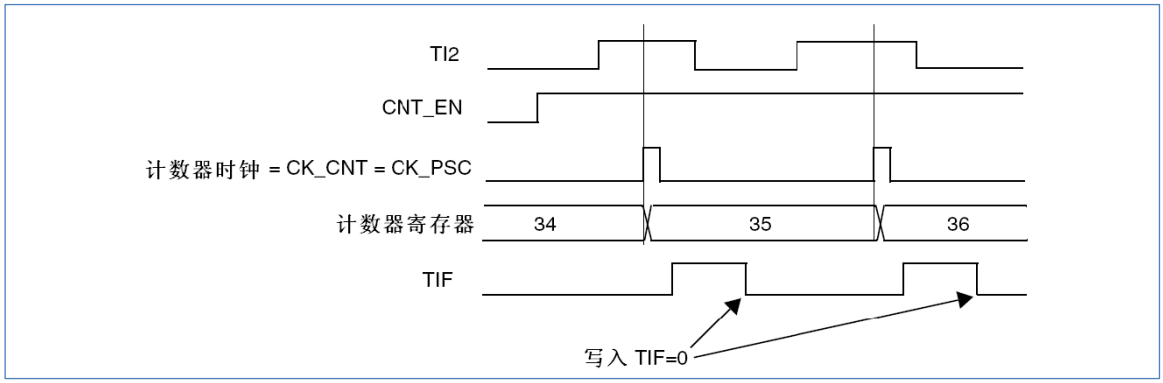

图 16-23 外部时钟模式 1 下的控制电路

16.2.3.3 外部时钟源模式 2

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

下图是外部触发输入的框图。

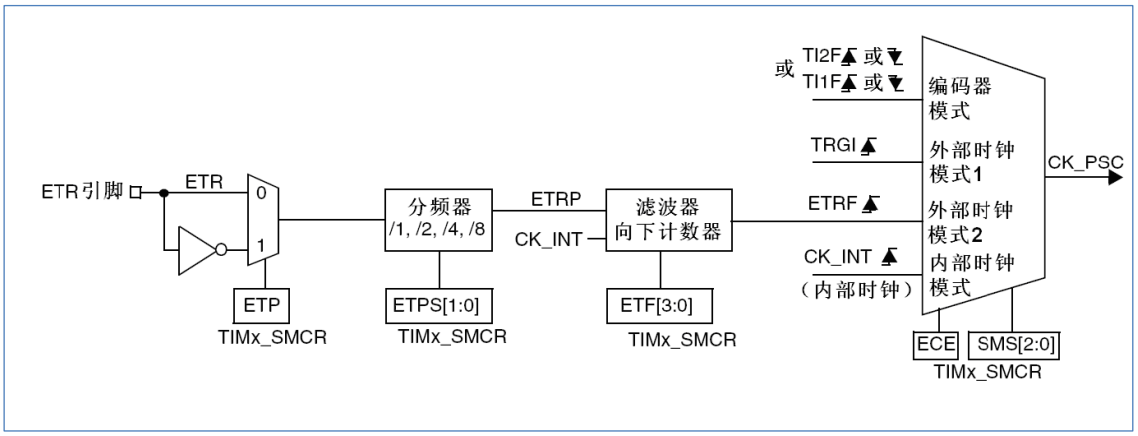

图 16-24 外部触发输入框图

例如，要配置在 ETR 下每两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 TIMx_SMCR 寄存器中的 ETF[3:0]=0000。
2. 设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0]=01。
3. 设置在 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0。
4. 开启外部时钟模式 2，置 TIMx_SMCR 寄存器中的 ECE=1。
5. 启动计数器，置 TIMx_CR1 寄存器中的 CEN=1。

计数器在每两个 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。

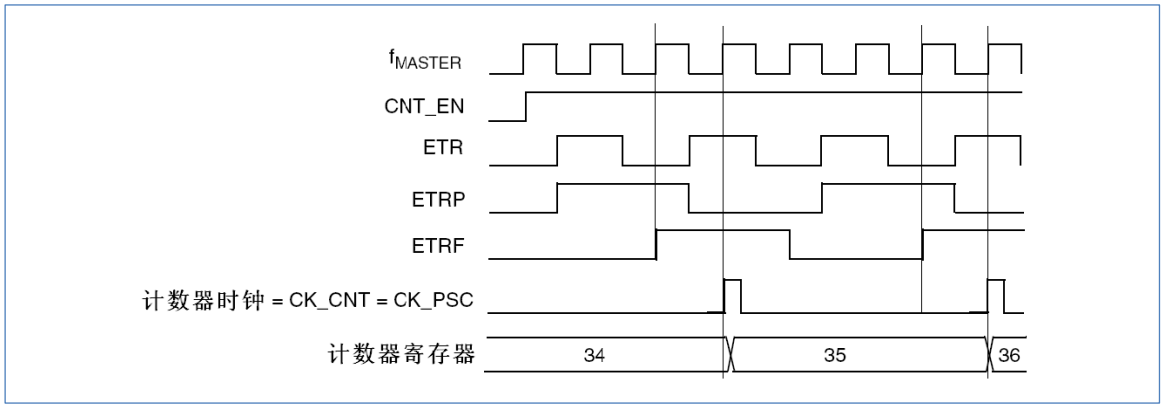

图 16-25 外部时钟模式 2 下的控制电路

16.2.4 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）。

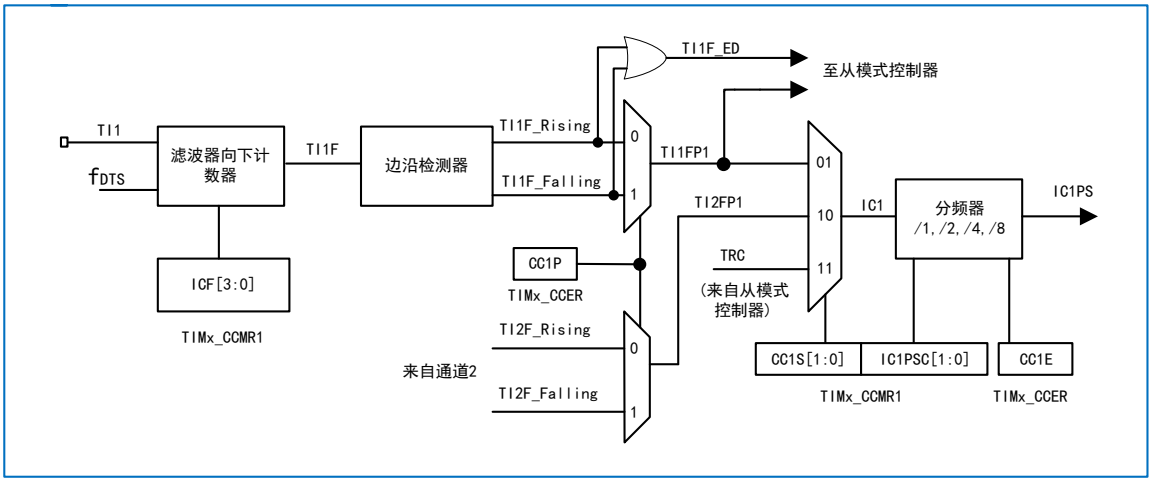

图 16-26 捕获/比较通道（如：通道 1 输入部分）

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。

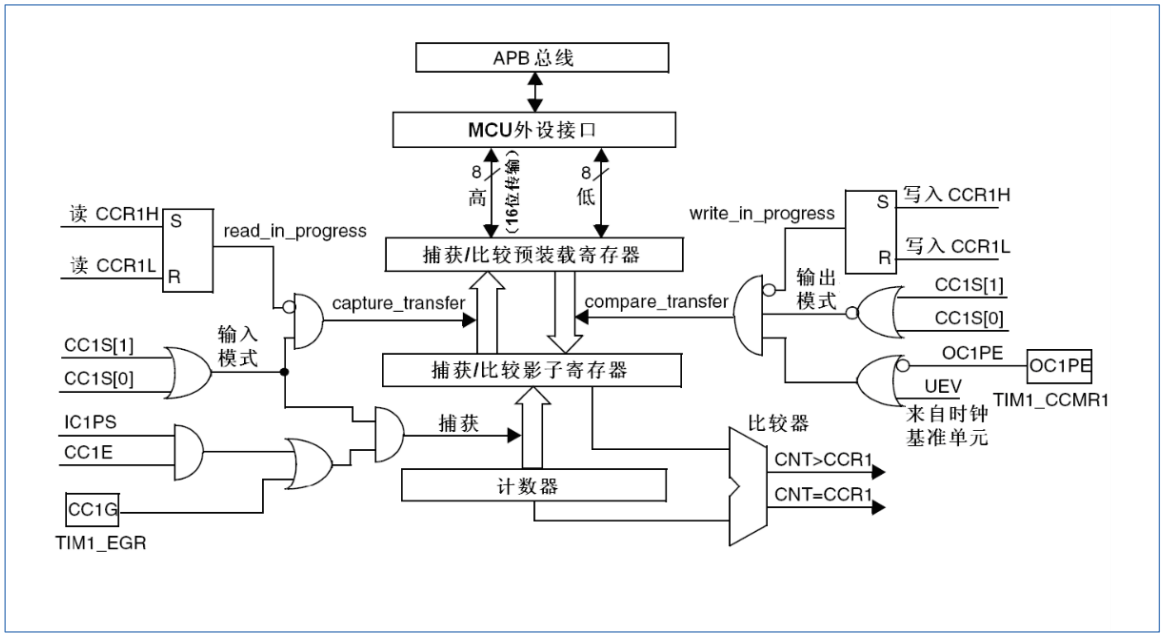

图 16-27 捕获/比较通道 1 的主电路

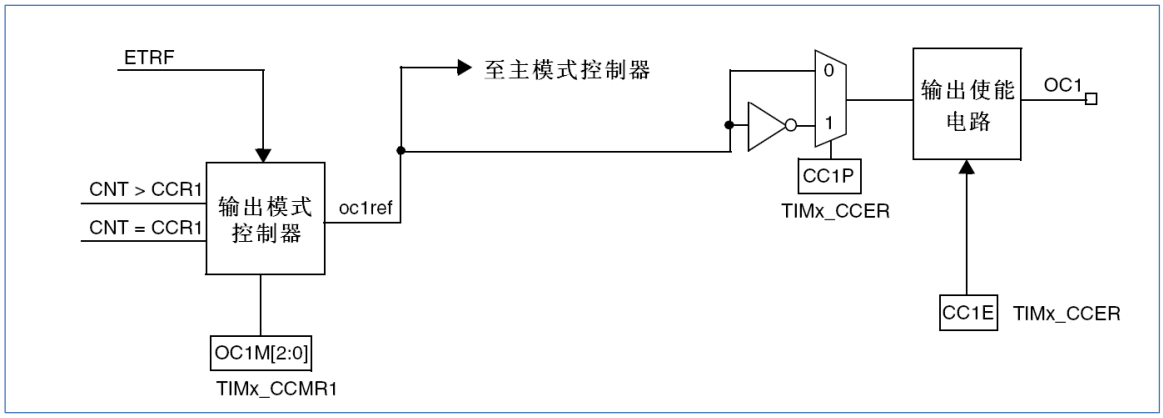

图 16-28 捕获/比较通道的输出部分（通道 1）

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。

16.2.5 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（TIMx_CCRx）中。当捕获事件发生时，相应的 CCxIF 标志（TIMx_SR 寄存器）被置'1'，如果使能了中断或者 DMA 操作，则将产生中断或者 DMA 操作。如果捕获事件发生时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMx_SR 寄存器）被置'1'。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

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

1. 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01。只要 CC1S 不为'00'，通道将被配置为输入并且 TIMx_CCR1 寄存器变为只读。
2. 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽大于 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 TIMx_CCMR1 寄存器中写入 IC1F=0011。
3. 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0（上升沿）。
4. 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PSC=00）。
5. 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
6. 如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少两个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置'1'。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

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

16.2.6 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。
- 这两个 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（TIMx_CCR1 寄存器）和占空比（TIMx_CCR2 寄存器），具体步骤如下（取决于 CK_INT 的频率和预分频器的值）：

1. 选择 TIMx_CCR1 的有效输入：置 TIMx_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 TIMx_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。

3. 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10（选择 TI1）。
4. 选择 TI1FP2 的有效极性（捕获数据到 TIMx_CCR2）：置 TIMx_CCER 寄存器的 CC2P=1（下降沿有效）。
5. 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
6. 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7. 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

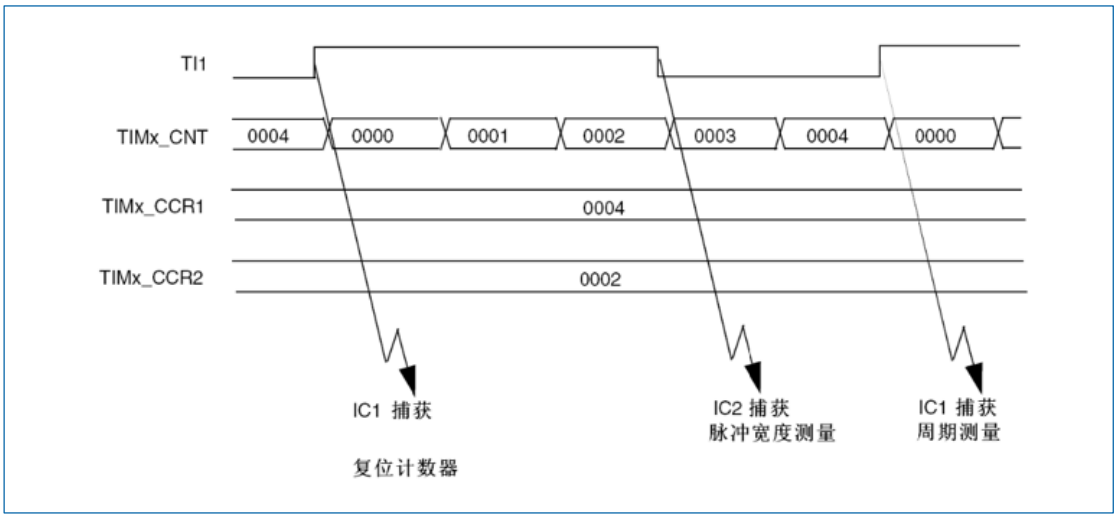

图 16-29 PWM 输入模式时序

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 TIMx_CH1/TIMx_CH2 信号。

16.2.7 强置输出模式

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

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性位相反的值。

例如：CCxP=0（OCx 高电平有效），则 OCx 被强置为高电平。

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

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

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

- 设置中断状态寄存器中的标志位（TIMx_SR 寄存器中的 CCxIF 位）。
- 若设置了相应的中断屏蔽（TIMx_DIER 寄存器中的 CCxIE 位），则产生一个中断。
- 若设置了相应的使能位（TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择

DMA 请求功能），则产生一个 DMA 请求。

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

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

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

输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求和/或一个 DMA 请求，设置 CCxIE 位和/或 CCxDE 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 TIMx_CR1 寄存器的 CEN 位启动计数器。

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

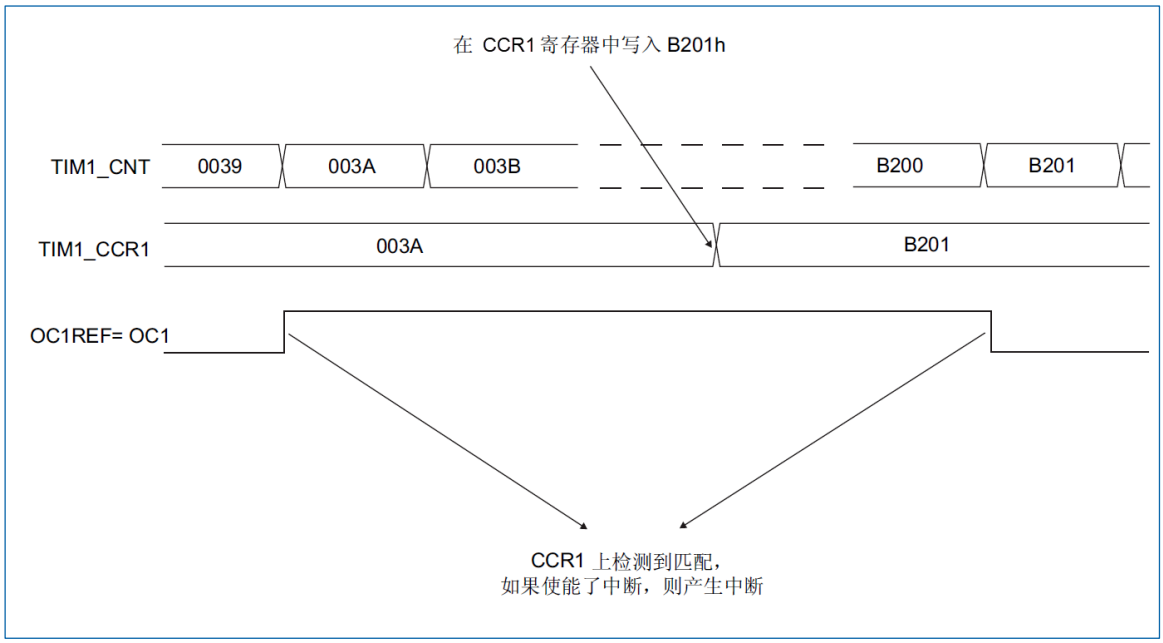

图 16-30 输出比较模式，翻转 OC1

16.2.9 PWM 模式

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

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’（PWM 模式 1）或’111’（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须设置 TIMx_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 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 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。TIMx_CCER 寄存器中的

CCxE 位控制 OCx 输出使能。参见“16.3.9 TIMx 捕捉/比较使能寄存器（TIMx_CCER）（x=2..3）”。

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

16.2.9.1 PWM 边沿对齐模式

向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参看“16.2.2 计数器模式”。

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

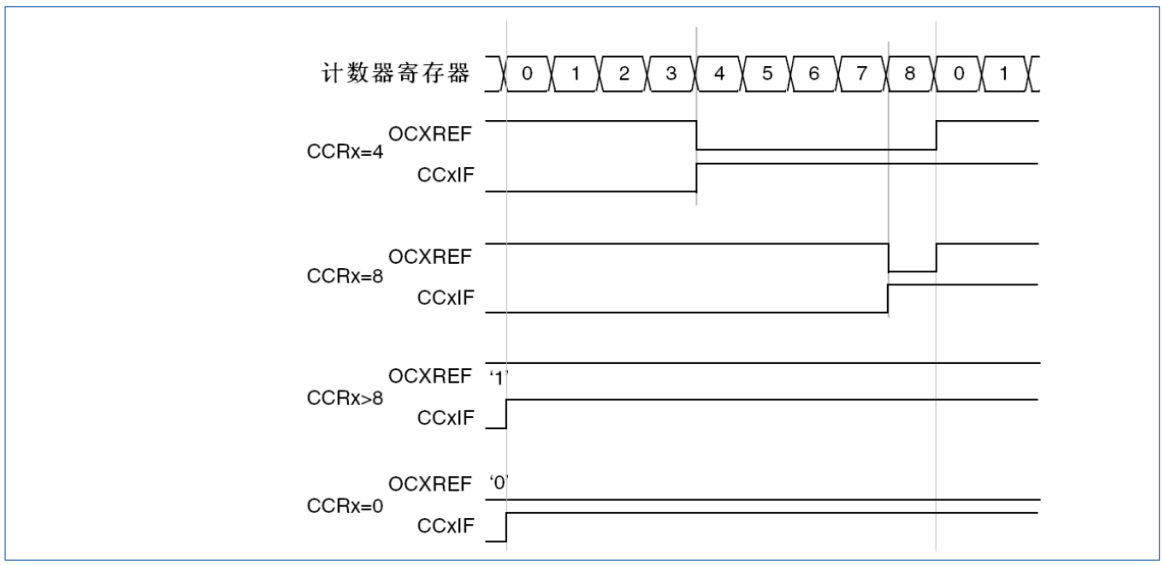

图 16-31 边沿对齐的 PWM 波形（ARR=8）

向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。参看“16.2.2 计数器模式”。

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

16.2.9.2 PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时，为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置’1’、在计数器向下计数时被置’1’、或在计数器向上和向下计数时被置’1’。TIMx_CR1 寄存器中的计数方向位（DIR）由硬件更新，不用软件修改。参见“16.2.2.3 中央对齐模式（向上/向下计数）”。

下图给出了一些中央对齐的 PWM 波形的例子。

- TIMx_ARR=8
- PWM 模式 1
- TIMx_CR1 寄存器中的 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。

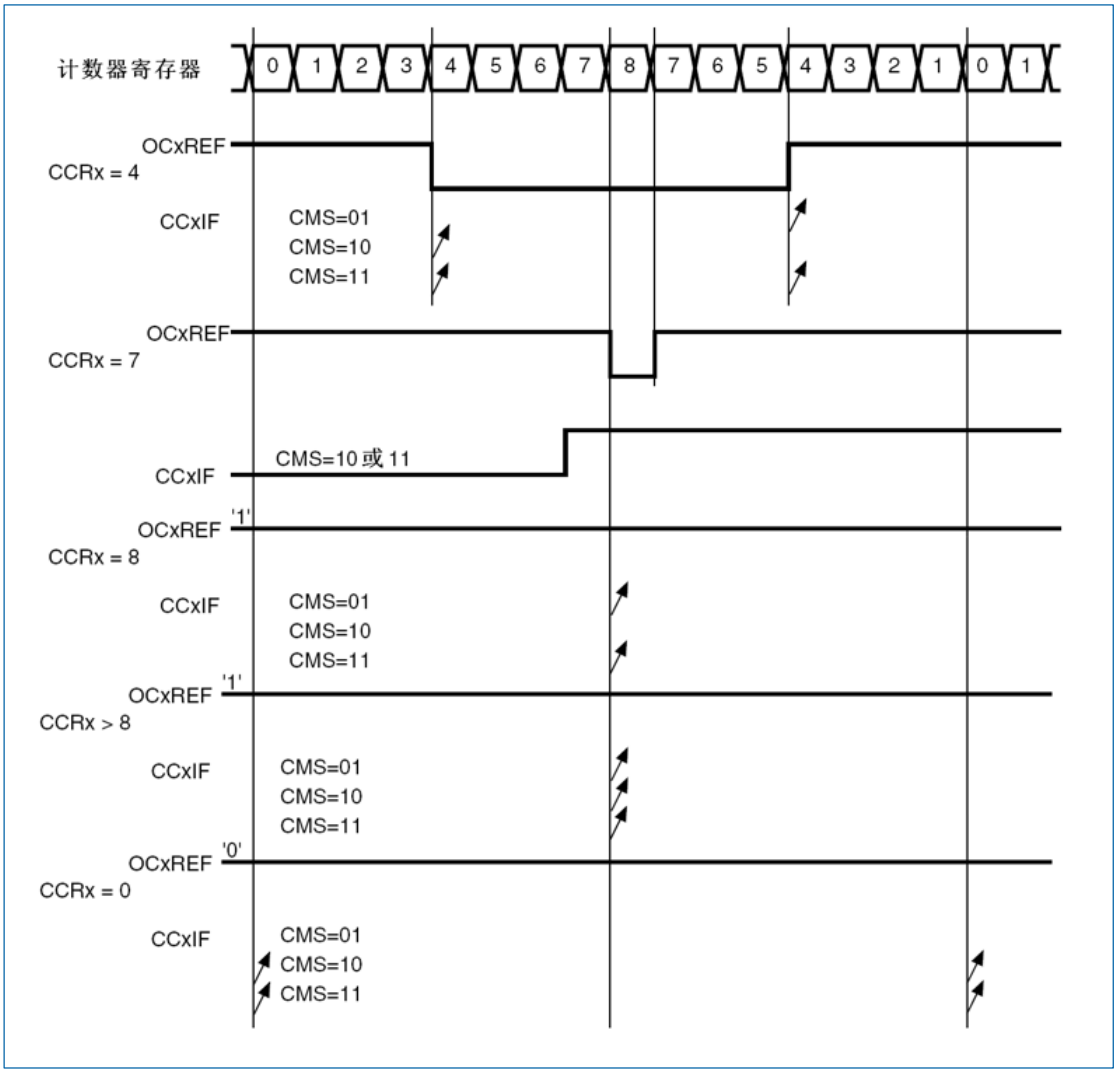

图 16-32 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

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

16.2.10 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 TIMx_CR1 寄

存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

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

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

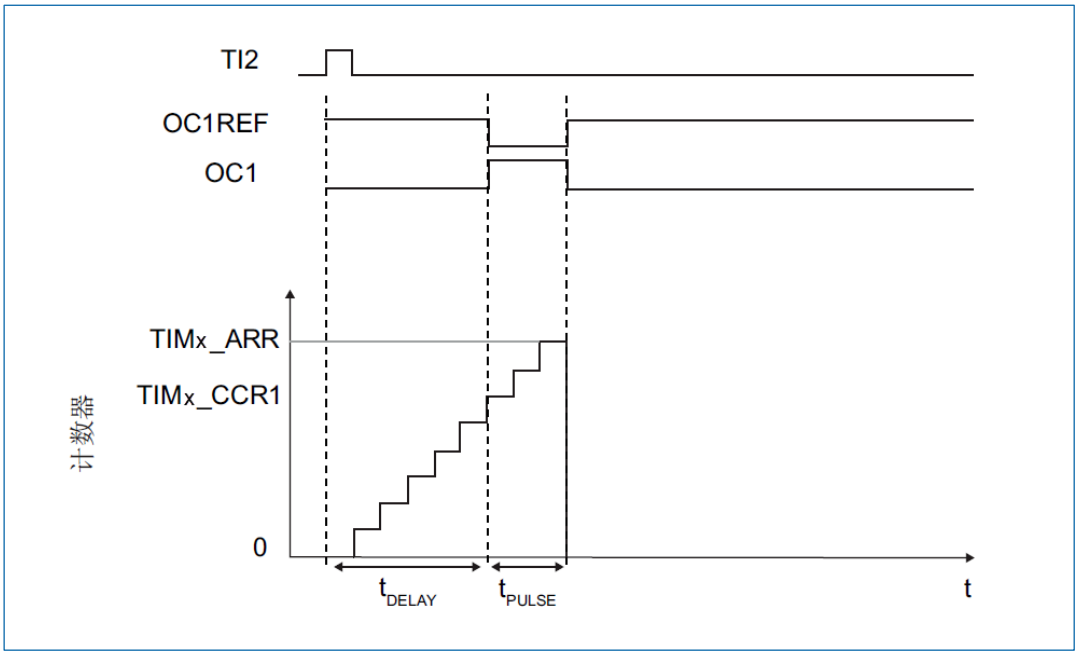

图 16-33 单脉冲模式的例子

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：

- 置 TIMx_CCMR1 寄存器中的 CC2S='01'，把 TI2FP2 映射到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P='0'，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS='110'，TI2FP2 作为从模式控制器的触发（TRGI）。
- 置 TIMx_SMCR 寄存器中的 SMS='110'（触发模式），TI2FP2 被用来启动计数器。

OPM 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 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义（TIMx_ARR-TIMx_CCR1）。
- 假定当发生比较匹配时要产生从'0'到'1'的波形，当计数器到达预装载值时要产生一个从'1'到'0'的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M='111'，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE='1'和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，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，TIMx_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 TIMx_CR1 寄存器中的 OPM='1'，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

特殊情况：OCx 快速使能：

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

t_{DELAY}。

如果要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF（和 OCx）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。

16.2.11 在外部事件时清除 OCxREF 信号

对于一个给定的通道，通过 ETRF 输入端或 OCREF_CLR 输出的高电平把 OCxREF 信号拉低（通过配置 TIMx_SMCR 寄存器中的 OCCS 位来选择 OCxREF 信号的清除源，OCREF_CLR 的信号源可以通过配置 TIMx_OR 寄存器中的 OCREF_CLR_RMP 位选择，ETR 的信号源可以通过配置 TIMx_OR 寄存器中的 ETR_RMP 位选择），OCxREF 信号将保持为低，直到发生下一次的更新事件 UEV。

注意：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：TIMx_SMCR 寄存器中的 ETPS[1:0]=’00’。
2. 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=‘0’。
3. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF 输入变为高时，在不同的 OCxCE 条件下，OCxREF 信号的值。在这个例子中，定时器 TIMx 被置于 PWM 模式。

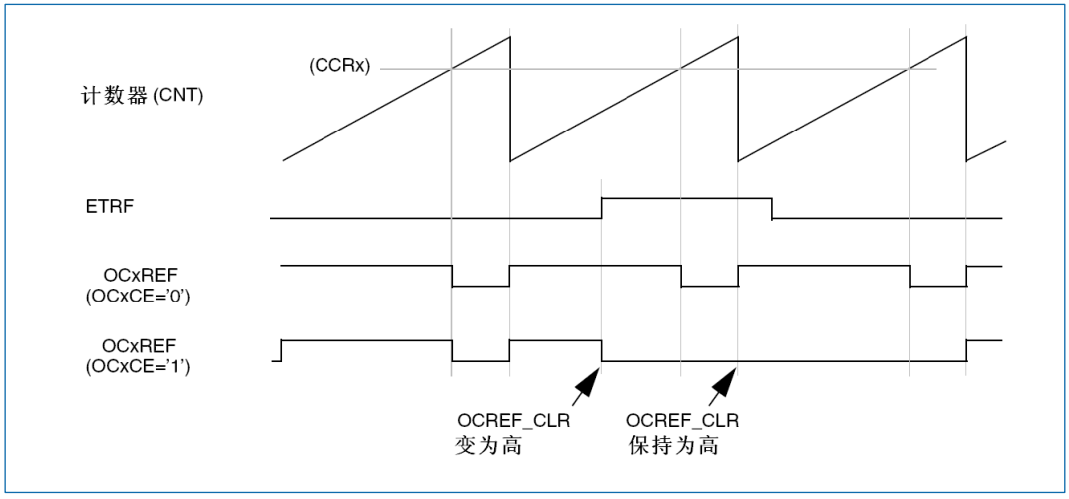

图 16-34 清除 TIMx 的 OCxREF

16.2.12 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 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

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

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参见表 16-2，假定计数器已经启动（TIMx_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 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2

计数或者同时依靠 TI1 和 TI2 计数。在任一输入端（TI1 或者 TI2）的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数（根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数）。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、触发输出特性等仍工作如常。

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 16-2 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）	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差分输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

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

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

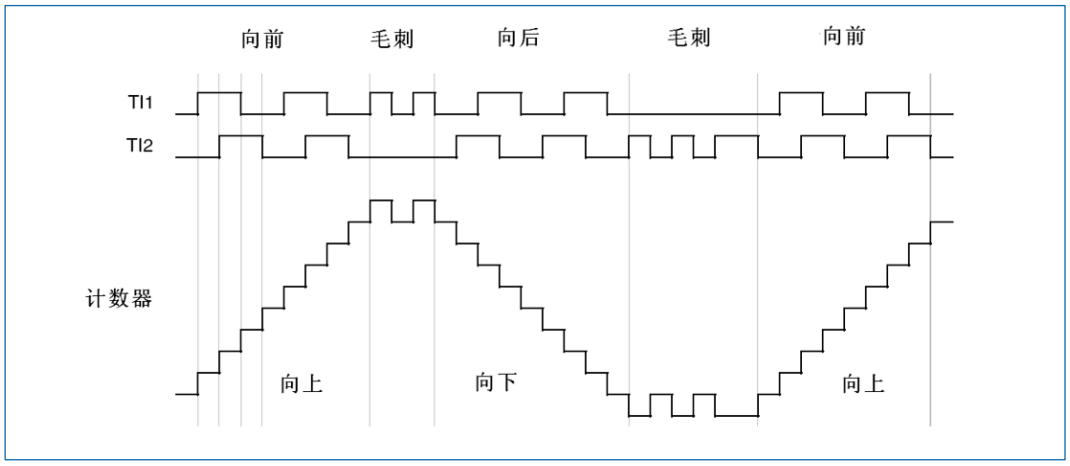

图 16-35 编码器模式下的计数器操作实例

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）。

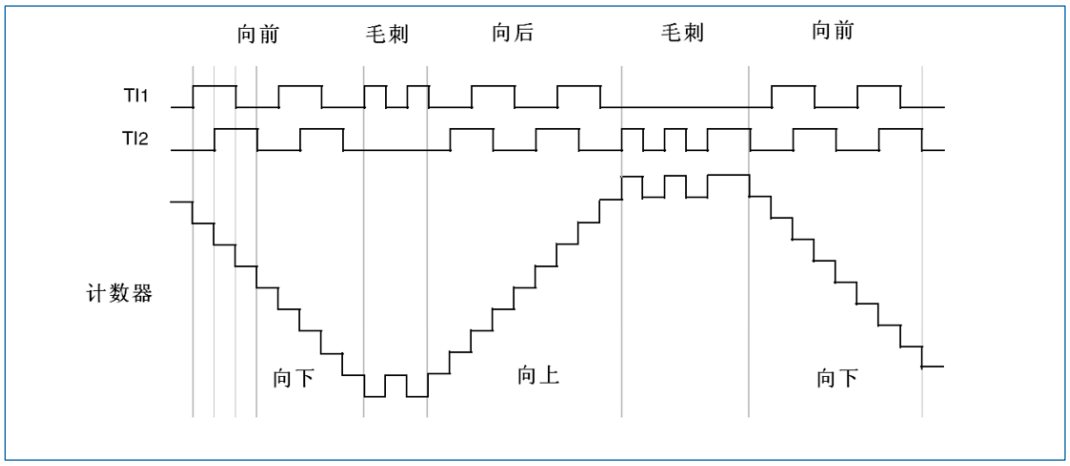

图 16-36 IC1FP1 反相的编码器接口模式实例

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）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

16.2.13 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。章节“15.2.18 与霍尔传感器的接口”给出了此特性用于连接霍尔传感器的例子。

16.2.14 定时器和外部触发的同步

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

16.2.14.1 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 TIMx_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（TIMx_ARR，TIMx_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 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

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

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

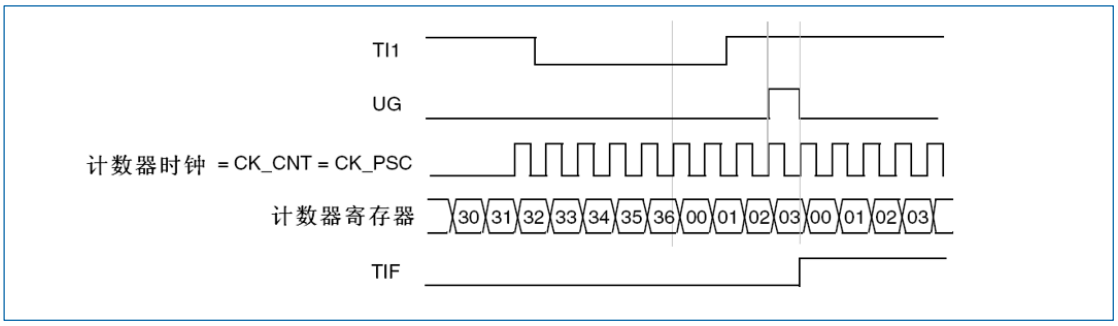

图 16-37 复位模式下的控制电路

16.2.14.2 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 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，在 TI1 变高时停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时，取决于 TI1 输入端的重同步电路。

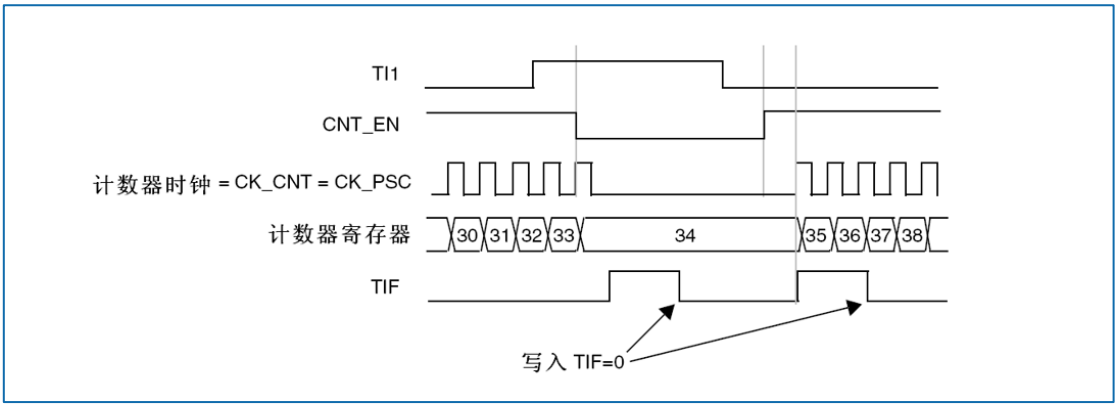

图 16-38 门控模式下的控制电路

16.2.14.3 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 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。
- 当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

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

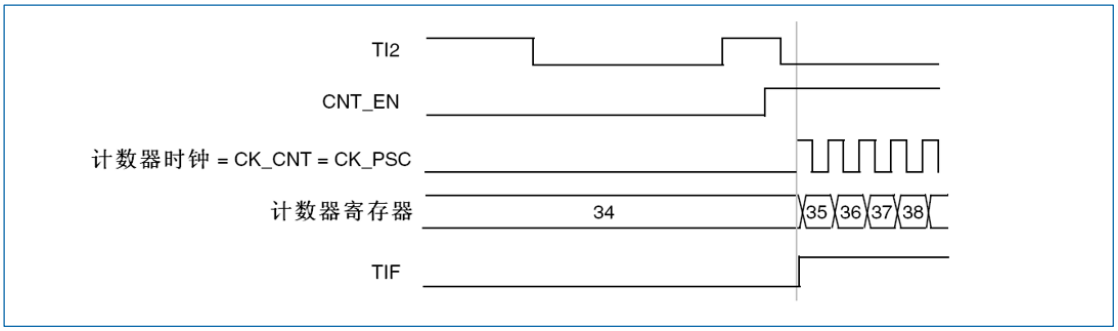

图 16-39 触发器模式下的控制电路

16.2.14.4 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 TIMx_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 TIMx_SMCR 寄存器配置外部触发输入电路：
  - ETF=0000：没有滤波
  - ETPS=00：不用预分频器
  - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式。
- 按如下配置通道 1，检测 TI 的上升沿：
  - IC1F=0000：没有滤波

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

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

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

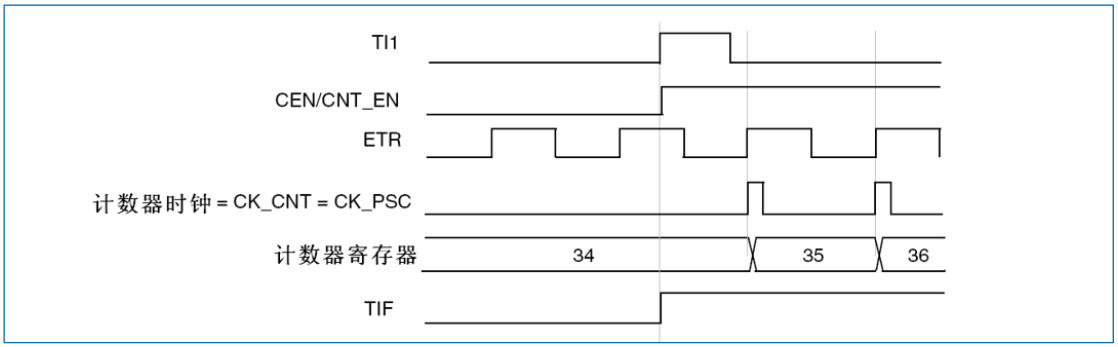

图 16-40 外部时钟模式 2＋触发模式下的控制电路

16.2.15 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。

16.2.15.1 使用一个定时器作为另一个定时器的预分频器

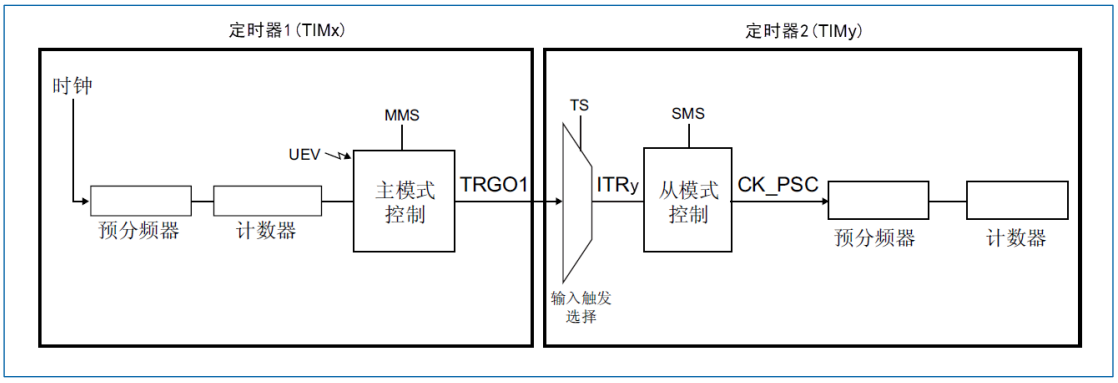

图 16-41 主/从定时器的例子

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 1TIMx 的触发输出（MMS=1xx），它的上升沿用于驱动定时器 2TIMy 的计数器。

16.2.15.2 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 16-41 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。

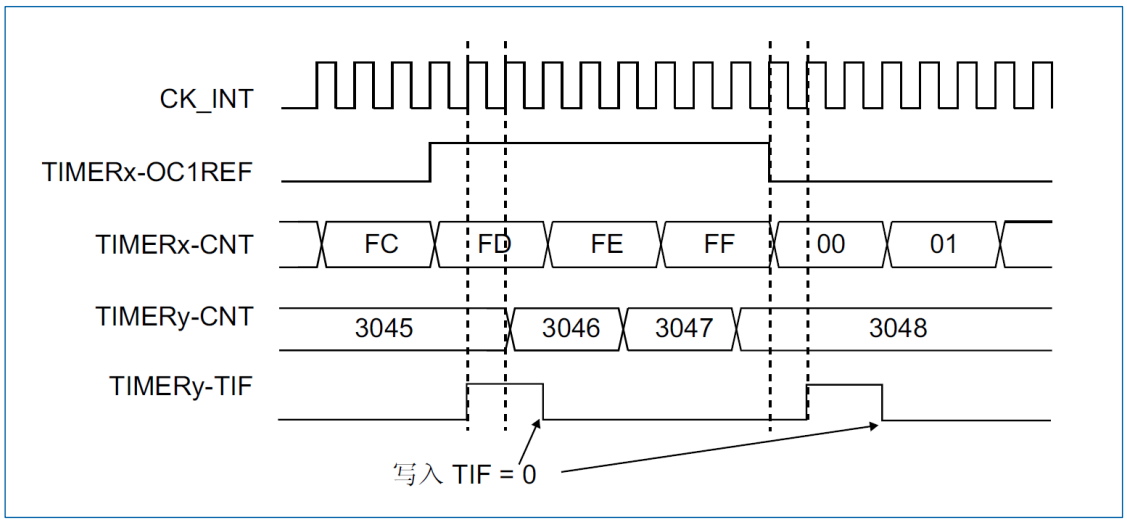

图 16-42 定时器 1（TIMx）的 OC1REF 控制定时器 2（TIMy）

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）之前复位两个定时器，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；两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。

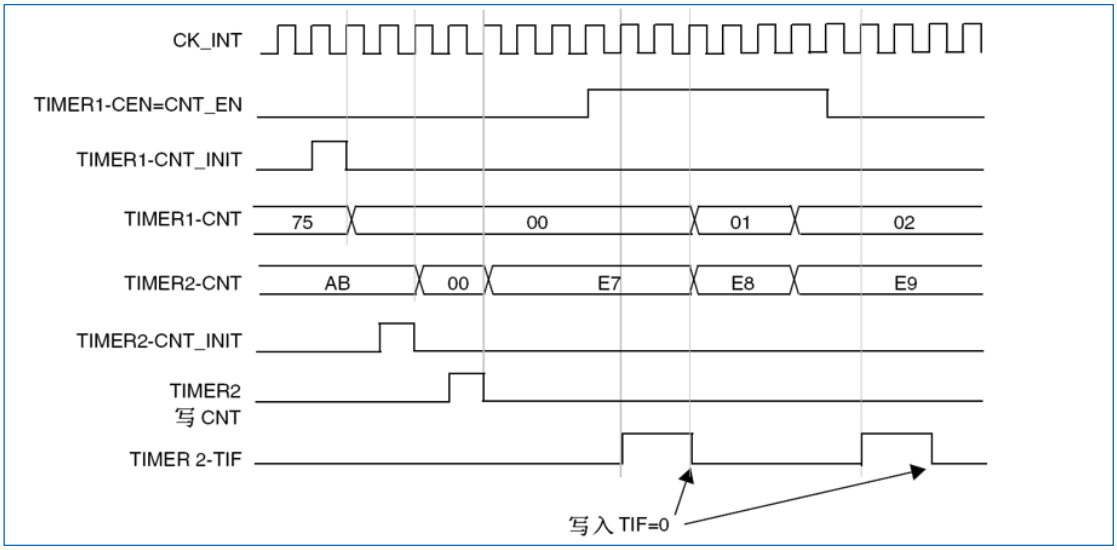

图 16-43 通过使能定时器 1（TIMx）可以控制定时器 2（TIMy）

16.2.15.3 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 16-41 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）做为触发输出（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。

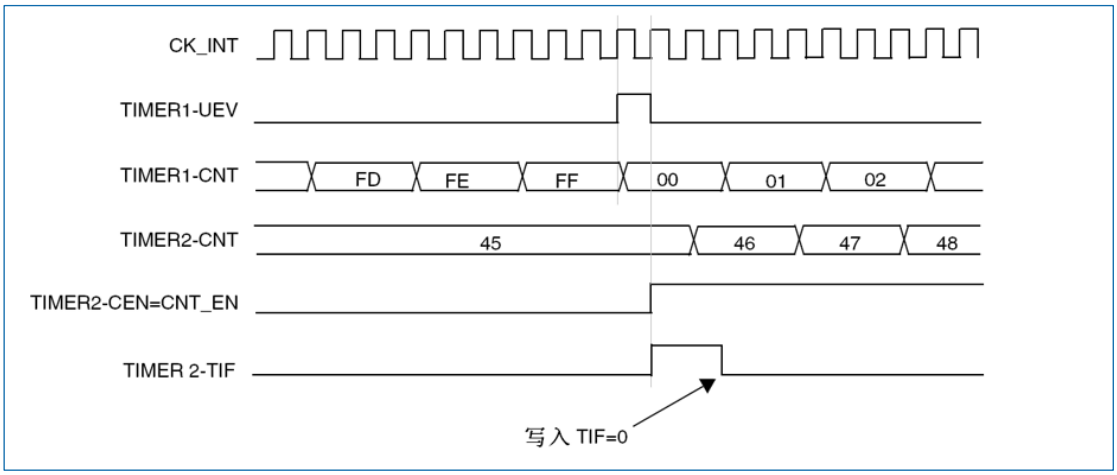

图 16-44 使用定时器 1（TIMx）的更新触发定时器 2（TIMy）

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。

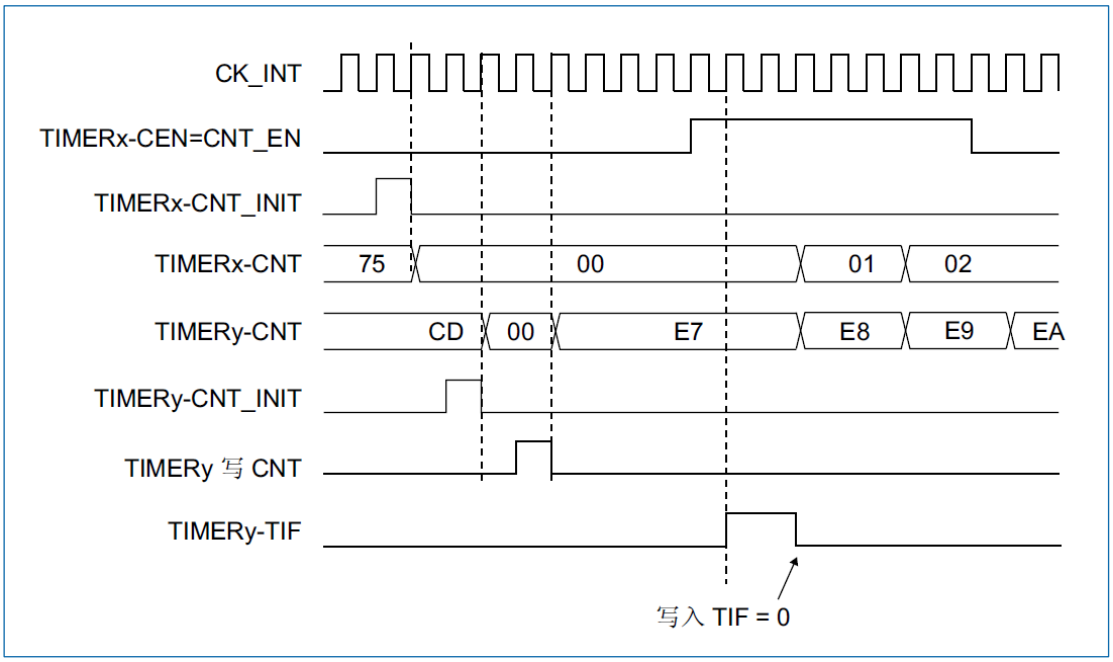

图 16-45 利用定时器 1（TIMx）的使能触发定时器 2（TIMy）

16.2.15.4 使用一个外部触发同步地启动两个定时器

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 16-41。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。

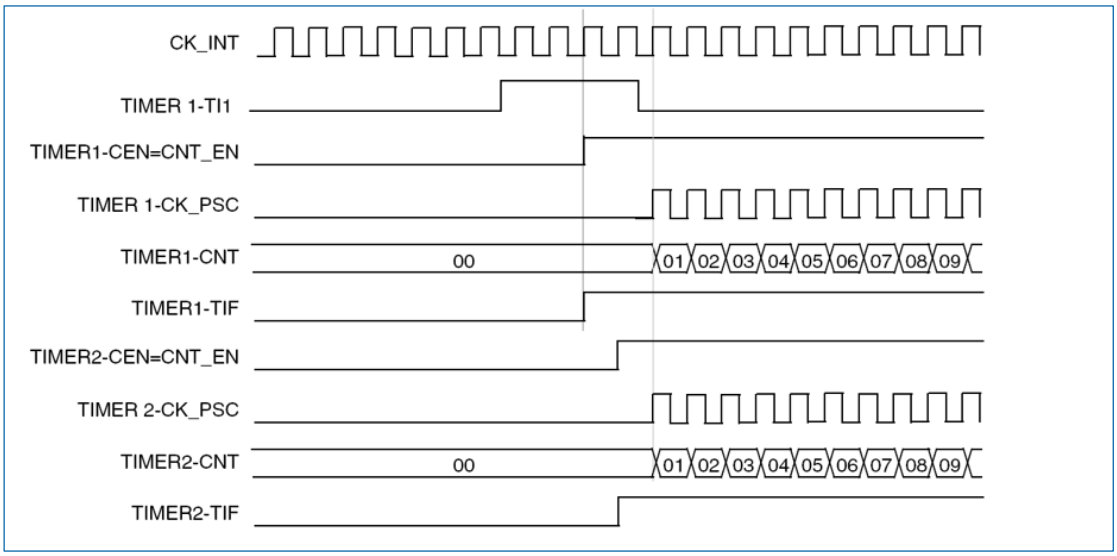

图 16-46 使用定时器 1（TIMx）的 TI1 输入触发定时器 1（TIMx）和定时器 2（TIMy）

16.2.16 调试模式

当微控制器进入调试模式（Cortex-M0 内核停止），根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器继续正常操作或者停止。参见“24.8.2 对定时器、看门狗和 I2C 的调试支持”。

16.3 TIM2/3 寄存器

基地址：（TIM2, TIM3）=（0x4001 3400, 0x4001 4000）

空间大小：（TIM2, TIM3）=（0x400, 0x400）

16.3.1 TIMx 控制寄存器 1（TIMx_CR1）（x=2..3）

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） 该位域定义在定时器时钟（CK_INT）频率、死区时间和由死区发生器与数字滤波器（ETR、TIx）所用的采样时钟之间的分频比例。 ● 00：$t_{DTS}=t_{CK\_INT}$ ● 01：$t_{DTS}=2*t_{CK\_INT}$ ● 10：$t_{DTS}=4*t_{CK\_INT}$ ● 11：保留（不使用该配置）

位 7	ARPE：自动重装载预装载允许位（Auto-reload preload enable） ● 0：TIMx_ARR 寄存器没有缓冲 ● 1：TIMx_ARR 寄存器有缓冲
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 计数器交替地向上和向下计数。配置为输出的通道（TIMx_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向下计数时被设置。 ● 10：中央对齐模式 2 计数器交替地向上和向下计数。配置为输出的通道（TIMx_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向上计数时被设置。 ● 11：中央对齐模式 3 计数器交替地向上和向下计数。配置为输出的通道（TIMx_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，在计数器向上和向下计数时均被设置。
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时，计数器停止（清除 CEN 位）。
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或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： ○ 计数器溢出/下溢 ○ 设置 UG 位 ○ 从模式控制器产生的更新 ● 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
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： ○ 计数器溢出/下溢 ○ 设置 UG 位 ○ 从模式控制器产生的更新，具有缓存的寄存器被装入它们的预装载值。 ● 1：禁止 UEV。不产生更新事件，影子寄存器（ARR、PSC、CCRx）保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
位 0	CEN：使能计数器（Counter enable） ● 0：禁止计数器 ● 1：使能计数器

16.3.2 TIMx 控制寄存器 2（TIMx_CR2）（x=2..3）

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]			CCDS	Res		
								rw	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） 该位域用于选择在主模式下送到从定时器的同步信息（TRGO）。可能的组合如下： ● 000：复位 TIMx_EGR 寄存器的 UG 位被用于作为触发输出（TRGO）。如果是触发输入产生的复位（从模式控制器处于复位模式），则 TRGO 上的信号相对实际的复位会有一个延迟。 ● 001：使能计数器使能信号 CNT_EN 被用于作为触发输出（TRGO）。可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 标志时（即使它已经为高），触发输出送出一个正脉冲（TRGO）。 ● 100：比较 OC1REF 信号被用于作为触发输出（TRGO）。 ● 101：比较 OC2REF 信号被用于作为触发输出（TRGO）。 ● 110：比较 OC3REF 信号被用于作为触发输出（TRGO）。 ● 111：比较 OC4REF 信号被用于作为触发输出（TRGO）。
位 3	CCDS：捕获/比较的 DMA 选择（Capture/Compare DMA selection） ● 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求。 ● 1：当发生更新事件时，送出 CCx 的 DMA 请求。
位 2:0	Res：保留 必须保持复位值。

16.3.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x=2..3）

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） 该位选择是用 ETR 还是 ETR 的反相来作为触发操作。 ● 0：ETR 不反相，高电平或上升沿有效。 ● 1：ETR 被反相，低电平或下降沿有效。
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 prescaler） 外部触发信号 ETRP 的频率最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 ● 00：关闭预分频 ● 01：ETRP 频率除以 2 ● 10：ETRP 频率除以 4 ● 11：ETRP 频率除以 8
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：不起作用 ● 1：触发输入（TRGI）上的事件被延迟了，以允许在当前定时器与它的从定时器间的完美同步（通过 TRGO）。这对要求把几个定时器同步到一个单一的外部事件是非常有用的。
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：内部触发 3（ITR3）。 ● 100：TI1 的边沿检测器（TI1F_ED）。 ● 101：滤波后的定时器输入 1（TI1FP1）。 ● 110：滤波后的定时器输入 2（TI2FP2）。 ● 111：外部触发输入（ETRF）。

位 3	OCCS：清除 OCxREF 选择（OCxREF clear selection） 该位用于选择 OCxREF 信号的清除源。 ● 0：OCxREF_CLR_INT 连接到 OCREF_CLR 输入。 ● 1：OCxREF_CLR_INT 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/下计数。 ● 010：编码器模式 2 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 ● 011：编码器模式 3 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 ● 100：复位模式选中的触发输入（TRGI）的上升沿重新初始化计数器，并且产生一次更新寄存器。 ● 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（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 16-3 TIM2 和 TIM3 内部触发连接

从定时器	ITR0（TS=000）	ITR1（TS=001）	ITR2（TS=010）	ITR3（TS=011）
TIM2	TIM1_TRGO	未连接	TIM3_TRGO	未连接
TIM3	TIM1_TRGO	TIM2_TRGO	未连接	未连接

16.3.4 TIMx DMA/中断允许寄存器（TIMx_DIER）（x=2..3）

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	TDE	Res	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

位 15	Res：保留 必须保持复位值。
位 14	TDE：允许触发 DMA 请求（Trigger DMA request enable） ● 0：触发 DMA 请求禁止 ● 1：触发 DMA 请求允许
位 13	Res：保留 必须保持复位值。
位 12	CC4DE：捕捉/比较 4 DMA 请求使能（Capture/Compare 4 DMA request enable） ● 0：CC4 DMA 请求禁止 ● 1：CC4 DMA 请求允许
位 11	CC3DE：捕捉/比较 3 DMA 请求使能（Capture/Compare 3 DMA request enable） ● 0：CC3 DMA 请求禁止

	● 1：CC3 DMA 请求允许
位 10	CC2DE：捕捉/比较 2 DMA 请求使能（Capture/Compare 2 DMA request enable） ● 0：CC2 DMA 请求禁止 ● 1：CC2 DMA 请求允许
位 9	CC1DE：捕捉/比较 1 DMA 请求使能（Capture/Compare 1 DMA request enable） ● 0：CC1 DMA 请求禁止 ● 1：CC1 DMA 请求允许
位 8	UDE：更新 DMA 请求使能（Update DMA request enable） ● 0：更新 DMA 请求禁止 ● 1：更新 DMA 请求允许
位 7	Res：保留 必须保持复位值。
位 6	TIE：触发中断使能（Trigger interrupt enable） ● 0：触发中断禁止 ● 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 中断禁止 ● 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 中断禁止 ● 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 中断禁止 ● 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 中断禁止 ● 1：CC1 中断允许
位 0	UIE：更新中断使能（Update interrupt enable） ● 0：更新中断禁止 ● 1：更新中断允许

16.3.5 TIMx 状态寄存器（TIMx_SR）（x=2..3）

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） 参见 CC1OF 位。
位 11	CC3OF：捕捉/比较 3 重复捕捉标志（Capture/Compare 3 overcapture flag） 参见 CC1OF 位。
位 10	CC2OF：捕捉/比较 2 重复捕捉标志（Capture/Compare 2 overcapture flag） 参见 CC1OF 位。
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。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 TIMx_CCR1 寄存器。
位 8:7	Res：保留 必须保持复位值。
位 6	TIF：触发器中断标志（Trigger interrupt flag） 当发生触发事件（当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿）时由硬件对该位置'1'。它由软件清'0'。 ● 0：无触发器事件产生 ● 1：触发中断等待响应
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） 参见 CC1IF 位。
位 3	CC3IF：捕捉/比较 3 中断标志（Capture/Compare 3 interrupt flag） 参见 CC1IF 位。
位 2	CC2IF：捕捉/比较 2 中断标志（Capture/Compare 2 interrupt flag） 参见 CC1IF 位。
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。它由软件清'0'。 ○ 0：无匹配发生 ○ 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_ARR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高。 ● 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置'1'，它由软件清'0'或通过读 TIMx_CCR1 清'0'。 ○ 0：无输入捕获产生； ○ 1：计数器值被捕获至 TIMx_CCR1（在 IC1 上检测到与所选极性相同的边沿）。
位 0	UIF：更新中断标志（Update interrupt flag）

	当产生更新事件时该位由硬件置’1’。它由软件清’0’。 ● 0：无更新事件产生； ● 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： ○ 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时（重复计数器=0 时产生更新事件）。 ○ 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 ○ 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

16.3.6 TIMx 事件产生寄存器（TIMx_EGR）（x=2..3）

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：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：不起作用 ● 1：TIMx_SR 中 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
位 5	Res：保留 必须保持复位值。
位 4	CC4G：捕捉/比较 4 发生（Capture/Compare 4 generation） 参见 CC1G 位。
位 3	CC3G：捕捉/比较 3 发生（Capture/Compare 3 generation） 参见 CC1G 位。
位 2	CC2G：捕捉/比较 2 发生（Capture/Compare 2 generation） 参见 CC1G 位。
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：不起作用 ● 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，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
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：不起作用；

	● 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。

16.3.7 TIMx 捕捉/比较模式寄存器 1（TIMx_CCMR1）（x=2..3）

偏移地址：0x18

复位值：0x0000

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

注意：同一个位在输出模式和输入模式下的功能是不同的。

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	

输出比较模式：

位 15	OC2CE：输出比较 2 清零允许（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：输出比较 2 预装允许（Output compare 2 preload enable） 参见 OC1PE 位。
位 10	OC2FE：输出比较 2 快速允许（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 TIMx_SMCR 寄存器的 TS 位选择）。 *注意：CC2S 仅在通道关闭时（TIMx_CCER 寄存器的 CC2E=0）才是可写的。*
位 7	OC1CE：输出比较 1 清零允许（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、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 ● 000：冻结 输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用。 ● 001：匹配时设置通道 1 为有效电平 当计数器 TIMx_CNT 的值与捕获/比较寄存器 1（TIMx_CCR1）相同时，强制 OC1REF 为高。

	● 010：匹配时设置通道 1 为无效电平 当计数器 TIMx_CNT 的值与捕获/比较寄存器 1（TIMx_CCR1）相同时，强制 OC1REF 为低。 ● 011：翻转 当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 ● 100：强制为无效电平 强制 OC1REF 为低。 ● 101：强制为有效电平 强制 OC1REF 为高。 ● 110：PWM 模式 1 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。 ● 111：PWM 模式 2 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。
位 3	OC1PE：输出比较 1 预装允许（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起作用。 ● 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操。TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。
位 2	OC1FE：输出比较 1 快速使能（Output compare 1 fast enable） 该位用于加快 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，即使工作于触发器状态。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 ● 1：输入到触发器的有效沿的作用就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 模式时起作用。
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），及输入脚的选择： ● 00：CC1 通道被配置为输出。 ● 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 ● 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 ● 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。

输入捕捉模式：

位 15:12	IC2F[3:0]：输入捕捉 2 滤波器（Input capture 2 filter） 参见 IC1F 位。
位 11:10	IC2PSC[1:0]：输入捕捉 2 预分频器（Input capture 2 prescaler） 参见 IC1PSC 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），及输入脚的选择： ● 00：CC2 通道被配置为输出； ● 01：CC2 通道被配置为输入，IC2 映射在 TI2 上；

	● 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； ● 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。
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（TIMx_CCER 寄存器中），则预分频器复位。 ● 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。 ● 01：每 2 个事件触发一次捕获。 ● 10：每 4 个事件触发一次捕获。 ● 11：每 8 个事件触发一次捕获。
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），及输入脚的选择： ● 00：CC1 通道被配置为输出。 ● 01：CC1 通道被配置为输入，IC1 映射在 TI1 上。 ● 10：CC1 通道被配置为输入，IC1 映射在 TI2 上。 ● 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。

16.3.8 TIMx 捕捉/比较模式寄存器 2（TIMx_CCMR2）（x=2..3）

偏移地址：0x1C

复位值：0x0000

参见章节“16.3.7 TIMx 捕捉/比较模式寄存器 1（TIMx_CCMR1）”的描述。

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	rw	

输出比较模式:

位 15	OC4CE：输出比较 4 清零允许（Output compare 4 clear enable） 参见 TIMx_CCMR1.OC1CE 位。
位 14:12	OC4M[2:0]：输出比较模式 4（Output compare 4 mode） 参见 TIMx_CCMR1.OC1M 位。
位 11	OC4PE：输出比较 4 预装允许（Output compare 4 preload enable） 参见 TIMx_CCMR1.OC1PE 位。
位 10	OC4FE：输出比较 4 快速允许（Output compare 4 fast enable） 参见 TIMx_CCMR1.OC1FE 位。
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare 4 selection） 该位定义通道的方向（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）。 *注意：CC4S 仅在通道关闭时（TIMx_CCER 寄存器的 CC4E=0）才是可写的。*
位 7	OC3CE：输出比较 3 清零允许（Output compare 3 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	OC3M[2:0]：输出比较 3 模式（Output compare 3 mode） 参见 TIMx_CCMR1.OC3M 位。
位 3	OC3PE：输出比较 3 预装允许（Output compare 3 preload enable） 参见 TIMx_CCMR1.OC3PE 位。
位 2	OC3FE：输出比较 3 快速使能（Output compare 3 fast enable） 参见 TIMx_CCMR1.OC3FE 位。
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），及输入脚的选择： ● 00：CC3 通道被配置为输出。 ● 01：CC3 通道被配置为输入，IC3 映射在 TI3 上。 ● 10：CC3 通道被配置为输入，IC3 映射在 TI4 上。 ● 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。

输入捕捉模式：

位 15:12	IC4F[3:0]：输入捕捉 4 滤波器（Input capture 4 filter）

	参见 IC3F 位。
位 11:10	IC4PSC[1:0]：输入捕捉 4 预分频器（Input capture 4 prescaler） 参见 IC3PSC 位。
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），及输入脚的选择： ● 00：CC4 通道被配置为输出。 ● 01：CC4 通道被配置为输入，IC4 映射在 TI4 上。 ● 10：CC4 通道被配置为输入，IC4 映射在 TI3 上。 ● 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。
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]：输入捕捉 3 预分频器（Input capture 3 prescaler） 该位域定义了 CC3 输入（IC3）的预分频系数。一旦 CC3E=0（TIMx_CCER 寄存器中），则预分频器复位。 ● 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。 ● 01：每 2 个事件触发一次捕获。 ● 10：每 4 个事件触发一次捕获。 ● 11：每 8 个事件触发一次捕获。
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），及输入脚的选择： ● 00：CC3 通道被配置为输出。 ● 01：CC3 通道被配置为输入，IC3 映射在 TI3 上。 ● 10：CC3 通道被配置为输入，IC3 映射在 TI4 上。 ● 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMx_SMCR 寄存器的 TS 位选择）。

16.3.9 TIMx 捕捉/比较使能寄存器（TIMx_CCER）（x=2..3）

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） 参见 CC1P 位。
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） 参见 CC1P 位。
位 12	CC4E：捕捉/比较 4 输出使能（Capture/Compare 4 output enable） 参见 CC1E 位。
位 11	CC3NP：捕捉/比较 3 输出极性（Capture/Compare 3 output polarity） 参见 CC1P 位。
位 10	Res：保留 必须保持复位值。
位 9	CC3P：捕捉/比较 3 输出极性（Capture/Compare 3 output polarity） 参见 CC1P 位。
位 8	CC3E：捕捉/比较 3 输出使能（Capture/Compare 3 output enable） 参见 CC1E 位。
位 7	CC2NP：捕捉/比较 2 输出极性（Capture/Compare 2 output polarity） 参见 CC1P 位。
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） 参见 CC1P 位。
位 4	CC2E：捕捉/比较 2 输出使能（Capture/Compare 2 output enable） 参见 CC1E 位。
位 3	CC1NP：捕捉/比较 1 输出极性（Capture/Compare 1 output polarity） 参见 CC1P 位。
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） ● CC1 通道配置为输出： ○ 0：OC1 高电平有效。

	○ 1：OC1 低电平有效。 ● CC1 通道配置为输入： CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 ○ 00：非反相/上升沿 电路作用于 TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 非反相。 ○ 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。 ○ 10：保留不用 ○ 11：非反相/上升或下降沿 电路作用于 TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作）、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 3 output enable） ● CC1 通道配置为输出： ○ 0：关闭 OC1 禁止输出。 ○ 1：开启 OC1 信号输出到对应的输出引脚。 ● CC1 通道配置为输入： 本位用于决定是否一个定时器值的捕捉要装载到捕捉/比较寄存器 1（TIMx_CCR1）。 ○ 0：捕捉禁止 ○ 1：捕捉允许

表 16-4 标准 OCx 通道的输出控制位

CCxE 位	OCx 输出状态
0	禁止输出（OCx=0，OCx_EN=0）
1	OCx=OCxREF +极性，OCx_EN=1

16.3.10 TIMx 计数寄存器（TIMx_CNT）（x=2..3）

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
CNT_H[15:0]															
rw															

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

位 31:16	CNT_H[15:0]：计数器值（仅 TIM2 支持）（Counter value high bits of TIM2）
位 15:0	CNT_L[15:0]：计数器值（Counter value）

16.3.11 TIMx 预分频寄存器（TIMx_PSC）（x=2..3）

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]：预分频器的值（Prescaler value） 计数器的时钟频率：CK_CNT = 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 TIMx_EGR 的 UG 位清'0'或被工作在复位模式的从控制器清'0'。

16.3.12 TIMx 自动重装寄存器（TIMx_ARR）（x=2..3）

偏移地址：0x2C

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR_H[15:0]															
rw															

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

位 31:16	ARR_H[15:0]：自动重装载的值（High auto-reload value high bits of TIM2） ARR_H 包含了将要装载入实际的自动重装载寄存器的值。（仅 TIM2）
位 15:0	ARR_L[15:0]：自动重装载的值（Low auto-reload value） ARR_L 包含了将要装载入实际的自动重装载寄存器的值。

16.3.13 TIMx 捕捉/比较寄存器 1（TIMx_CCR1）（x=2..3）

偏移地址：0x34

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1_L[15:0]															
rw															

位 31:16	CCR1_H[15:0]：捕捉/比较通道 1 高 16 位值（仅 TIM2）（High Capture/Compare 1 value of TIM2）
位 15:0	CCR1_L[15:0]：捕捉/比较通道 1 的值（Low 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 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 ● 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

16.3.14 TIMx 捕捉/比较寄存器 2（TIMx_CCR2）（x=2..3）

偏移地址：0x38

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2_L[15:0]															
rw															

位 31:16	CCR2_H[15:0]：捕捉/比较通道 2 高 16 位值（仅 TIM2）（High Capture/Compare 2 value of TIM2）
位 15:0	CCR2_L[15:0]：捕捉/比较通道 2 的值（Low Capture/Compare 2 value） ● 若 CC2 通道配置为输出： CCR2 决定了装入当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器（OC2PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 ● 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件（IC2）传输的计数器值。

16.3.15 TIMx 捕捉/比较寄存器 3（TIMx_CCR3）（x=2..3）

偏移地址：0x3C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3_L[15:0]															
rw															

位 31:16	CCR3_H[15:0]：捕捉/比较通道 3 高 16 位值（仅 TIM2）（High Capture/Compare 3 value of TIM2）
位 15:0	CCR3_L[15:0]：捕捉/比较通道 3 的值（Low Capture/Compare 3 value） ● 若 CC3 通道配置为输出： CCR3 决定了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIMx_CCMR3 寄存器（OC3PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 ● 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件（IC3）传输的计数器值。

16.3.16 TIMx 捕捉/比较寄存器 4（TIMx_CCR4）（x=2..3）

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
CCR4_H[15:0]															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4_L[15:0]															
rw															

位 31:16	CCR4_H[15:0]：捕捉/比较通道 4 高 16 位值（仅 TIM2）（High Capture/Compare 4 value of TIM2）
位 15:0	CCR4_L[15:0]：捕捉/比较通道 4 的值（Low Capture/Compare 4 value） ● 若 CC4 通道配置为输出： CCR4 决定了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIMx_CCMR4 寄存器（OC4PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 ● 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件（IC4）传输的计数器值。

16.3.17 TIMx DMA 控制寄存器（TIMx_DCR）（x=2..3）

偏移地址：0x48

复位值：0x0000

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

位 15:13	Res：保留 必须保持复位值。
位 12:8	DBL[4:0]：DMA 连续传送长度（DMA burst length） 该位域定义了 DMA 在连续模式下的传送长度（当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送）。 ● 00000：1 次传输 ● 00001：2 次传输 ● 00010：3 次传输 ● ... ● 10001：18 次传输
位 7:5	Res：保留 必须保持复位值。
位 4:0	DBA[4:0]：DMA 基地址（DMA base address） 该位域定义 DMA 传输的基地址（通过 TIMx_DMAR 地址进行读/写访问时）。DBA 为从 TIMx_CR1 寄存器地址开始计算的偏移量。 ● 00000：TIMx_CR1 ● 00001：TIMx_CR2 ● 00010：TIMx_SMCR ... 示例：考虑以下传输：DBL=7 次传输，DBA=TIMx_CR1。这种情况下将向/从自 TIMx_CR1 地址开始的 7 个寄存器传输数据。

16.3.18 TIMx DMA 完全传送地址寄存器（TIMx_DMAR）（x=2..3）

偏移地址：0x4C

复位值：0x0000

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

位 15:0	DMAB[15:0]：DMA 并发（连续）传送寄存器 （DMA register for burst accesses） 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1 地址）+（DBA+DMA 索引）x4，其中： ● “TIMx_CR1 地址”是控制寄存器 1（TIMx_CR1）所在的地址。 ● “DBA”是 TIMx_DCR 寄存器中定义的基地址。 ● “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。 **DMA 连续传送功能使用方法示例：** 使用定时器 DMA 连续传送功能，将 TIMx_CCRx 寄存器（x = 2、3、4）的内容更新为 DMA 传输到 TIMx_CCRx 寄存器的内容。 具体操作步骤如下： ● 将相应的 DMA 通道配置如下： o DMA 通道外设地址为 TIMx_DMAR 寄存器地址。 o DMA 通道存储器地址为包含要通过 DMA 传输到 TIMx_CCRx 寄存器的数据的 RAM 缓冲区地址。 o 要传输的数据量为 3（参见下文“注意”）。 o 禁止循环模式。 ● 配置 DCR 寄存器的 DBA 和 DBL 位如下： o DBL = 3 次传输，DBA = 0xE。 ● 使能 TIMx 更新 DMA 请求（将 TIMx_DIER.UDE 置 1）。 ● 使能 TIMx（将 TIMx_CR1.CEN 置 1）。 ● 使能 DMA 通道。 *注意：本例适用于每个 TIMx_CCRx 寄存器只更新一次的情况。如果每个 CCRx 寄存器要更新两次，则要传输的数据量应为 6。* 下面以包含 data1、data2、data3、data4、data5 和 data6 的 RAM 缓冲区为例说明。 数据将按照如下方式传输到 TIMx_CCRx 寄存器： 1. 在第一个更新 DMA 请求期间，data1 传输到 CCR2，data2 传输到 CCR3，data3 传输到 CCR4； 2. 在第二个更新 DMA 请求期间，data4 传输到 CCR2，data5 传输到 CCR3，data6 传输到 CCR4。

16.3.19 TIM2 比较器控制寄存器（TIM2_OR）

偏移地址：0x50

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									OCREF_CLR_RMP[1:0]		TI4_RMP[1:0]		ETR_RMP[2:0]		
									rw		rw		rw		

位 15:7	Res：保留 必须保持复位值。
位 6:5	OCREF_CLR_RMP[1:0]：清除 OCREF 信号的重定向（Clear OCREF remap） 选择比较器 1 或比较器 2 的输出作为 OCREF_CLR 的来源。 ● 00：OCREF_CLR 连接 COMP1_OUT ● 01：OCREF_CLR 连接 COMP2_OUT ● 1x：OCREF_CLR 连接 COMP1_OUT 或 COMP2_OUT
位 4:3	TI4_RMP[1:0]：TI4 内部重定向（Internal trigger remap）

	● 01：TI4 连接 COMP2_OUT ● 10：TI4 连接 COMP1_OUT ● 其他值：TI4 连接 CH4_IN
位 2:0	ETR_RMP[2:0]：外部触发重定向（External trigger remap） ● 011：ETR 连接 HSI14 ● 101：ETR 连接 LSE_CLK ● 110：ETR 连接 COMP2_OUT ● 111：ETR 连接 COMP1_OUT ● 其他值：ETR 连接外部的输入引脚

16.3.20 TIM3 比较器控制寄存器（TIM3_OR）

偏移地址：0x50

复位值：0x0000

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

位 15:7	Res：保留 必须保持复位值。
位 6:5	OCREF_CLR_RMP[1:0]：清除 OCREF 信号的重定向（Clear OCREF remap） 选择比较器 1 或比较器 2 的输出作为 OCREF_CLR 的来源。 ● 00：OCREF_CLR 连接 COMP1_OUT ● 01：OCREF_CLR 连接 COMP2_OUT ● 1x：OCREF_CLR 连接 COMP1_OUT 或 COMP2_OUT
位 4:2	Res：保留 必须保持复位值。
位 1:0	ETR_RMP[1:0]：外部触发重定向（External trigger remap） ● 10：ETR 连接 HSI8 ● 其他：ETR 连接外部的输入引脚

17 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，由它的可编程预分频器驱动。

TIM6 可以作为通用定时器提供时间基准。

17.1 TIM6 主要功能

- 16 位自动重装载累加计数器
- 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）时产生中断请求

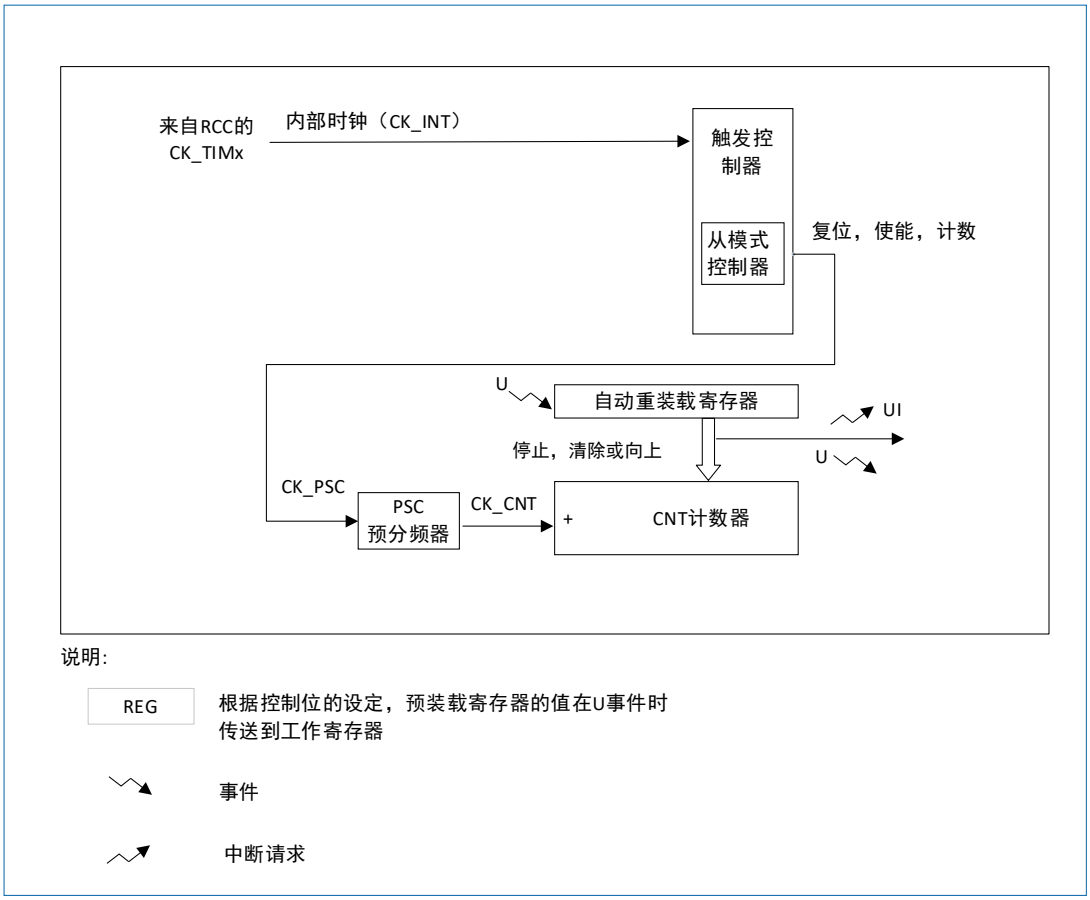

图 17-1 基本定时器框图

17.2 TIM6 主要功能

17.2.1 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，硬件发出更新事件；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。

注意：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。

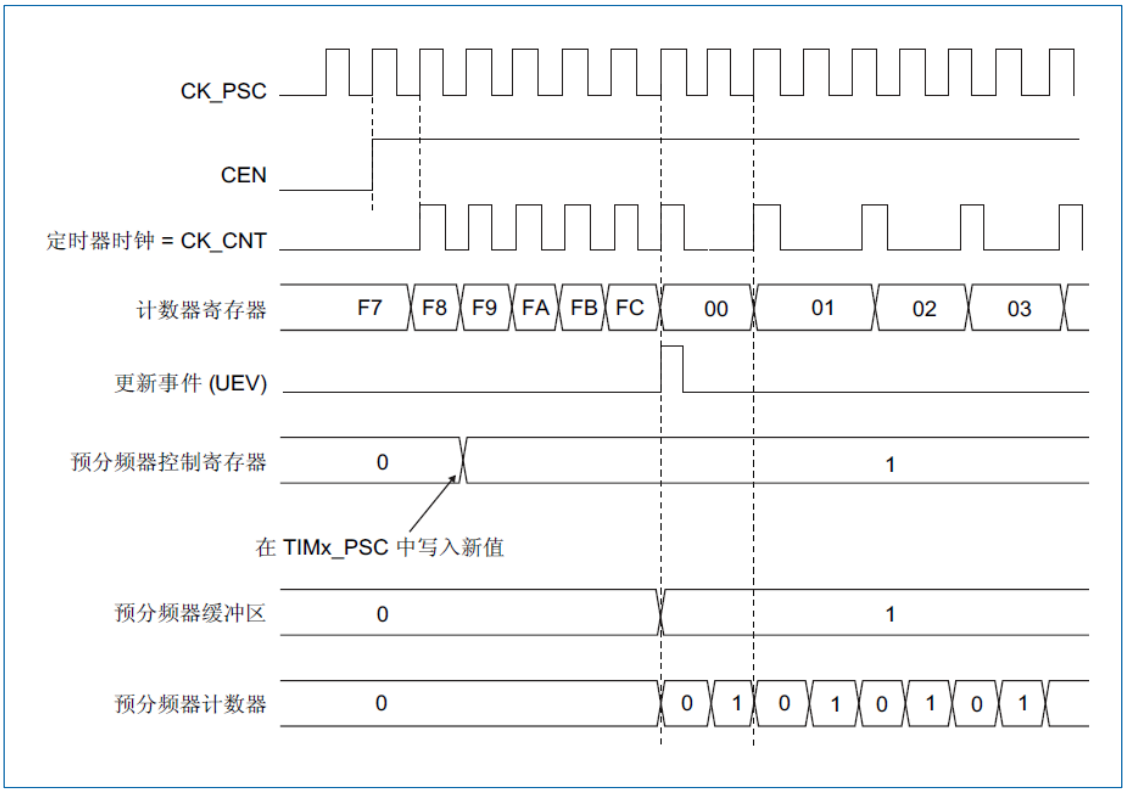

图 17-2 预分频系数从 1 变到 2 的计数器时序图

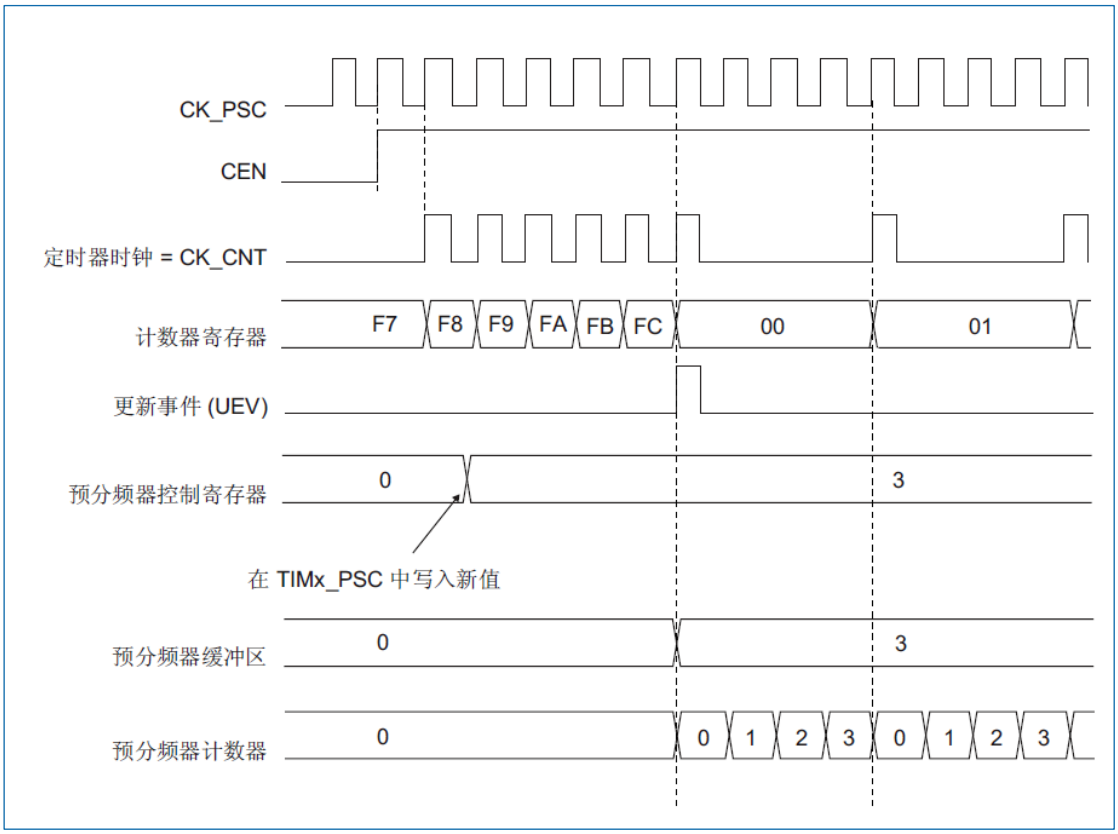

图 17-3 预分频系数从 1 变到 4 的计数器时序图

17.2.2 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（即没有中断或 DMA 请求）。

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。

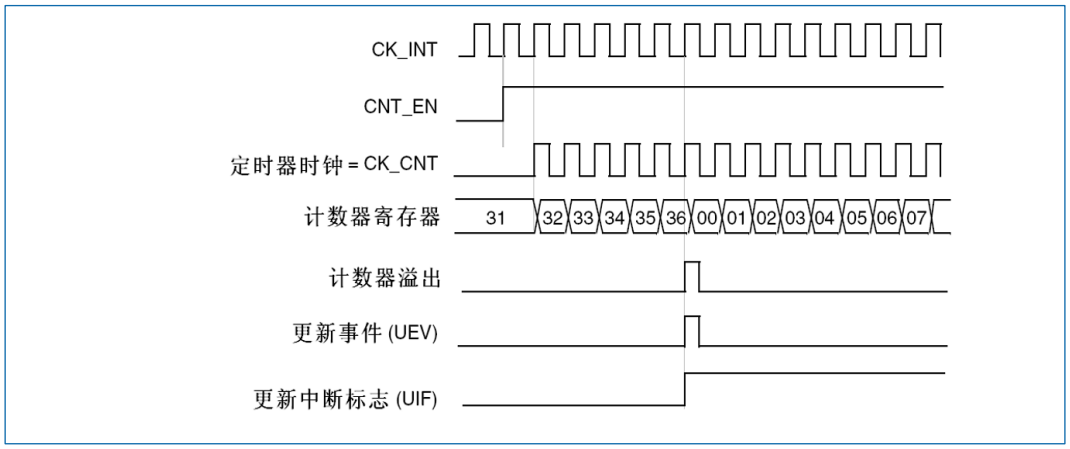

图 17-4 计数器时序图，内部时钟分频系数为 1

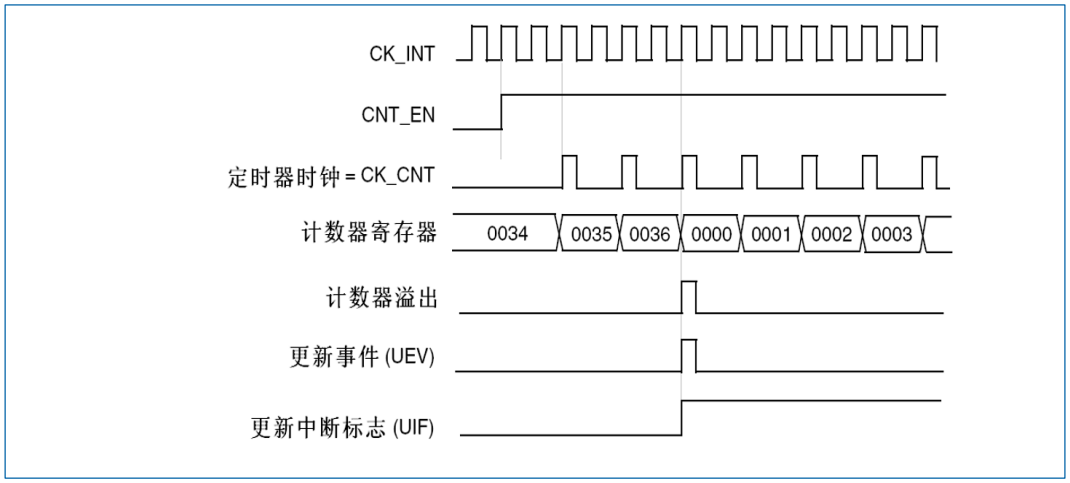

图 17-5 计数器时序图，内部时钟分频系数为 2

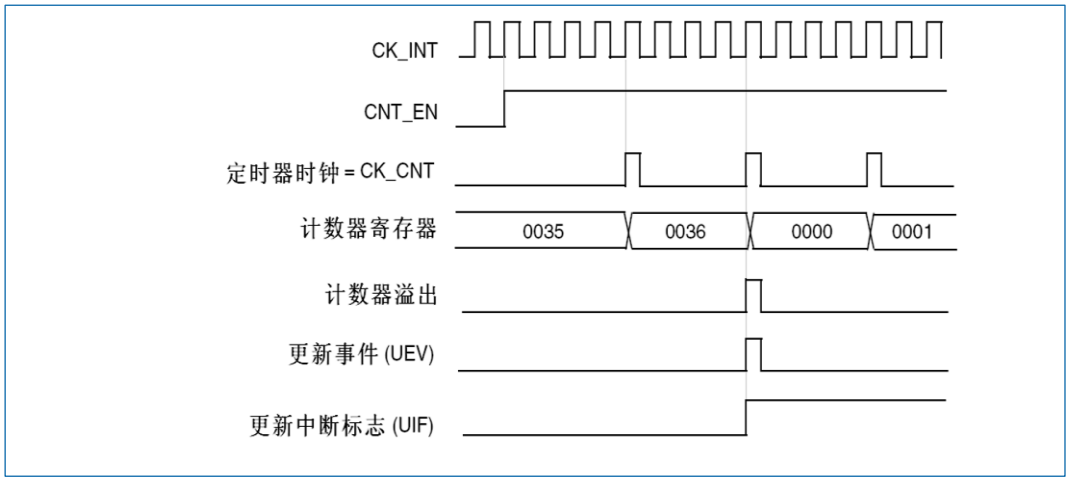

图 17-6 计数器时序图，内部时钟分频系数为 4

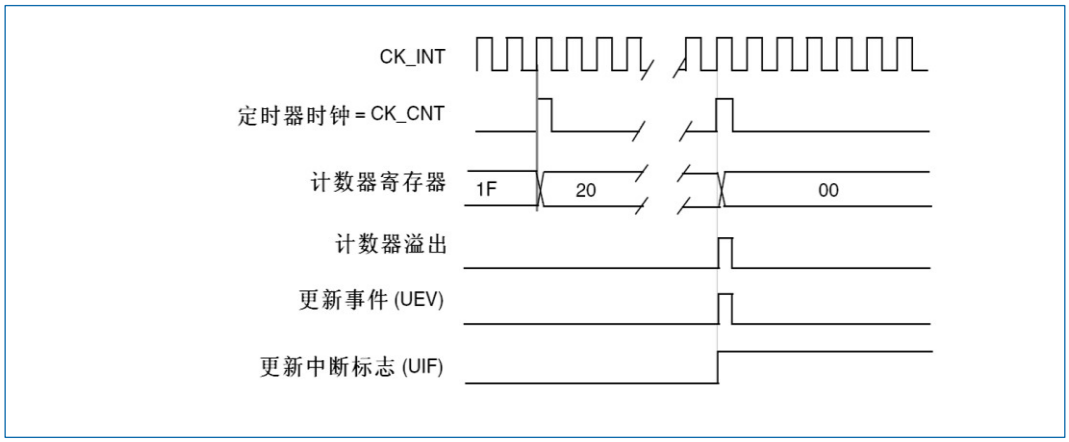

图 17-7 计数器时序图，内部时钟分频系数为 N

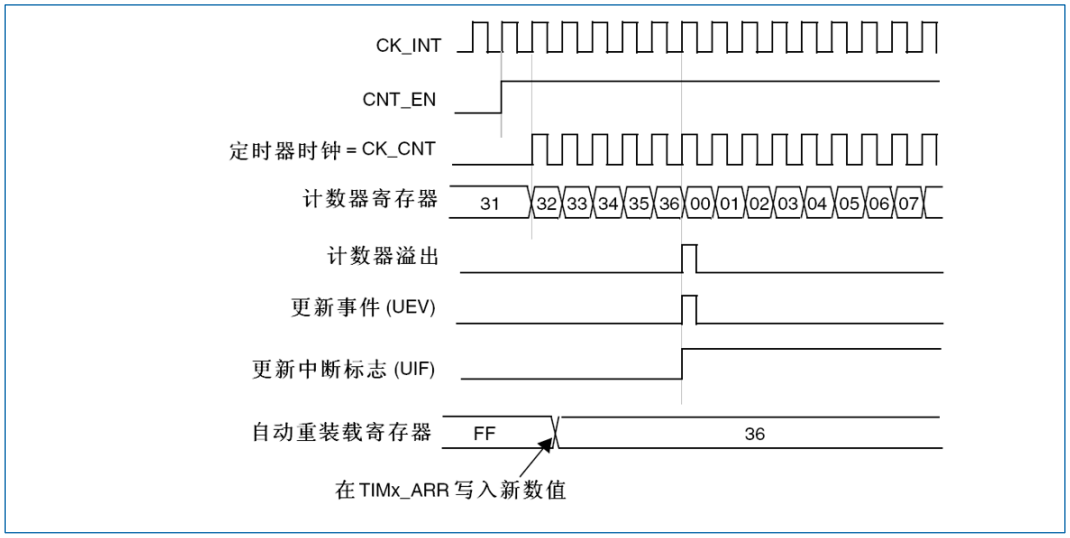

图 17-8 计数器时序图，当 ARPE=1 时的更新事件（无预装载 TIM6_ARR）

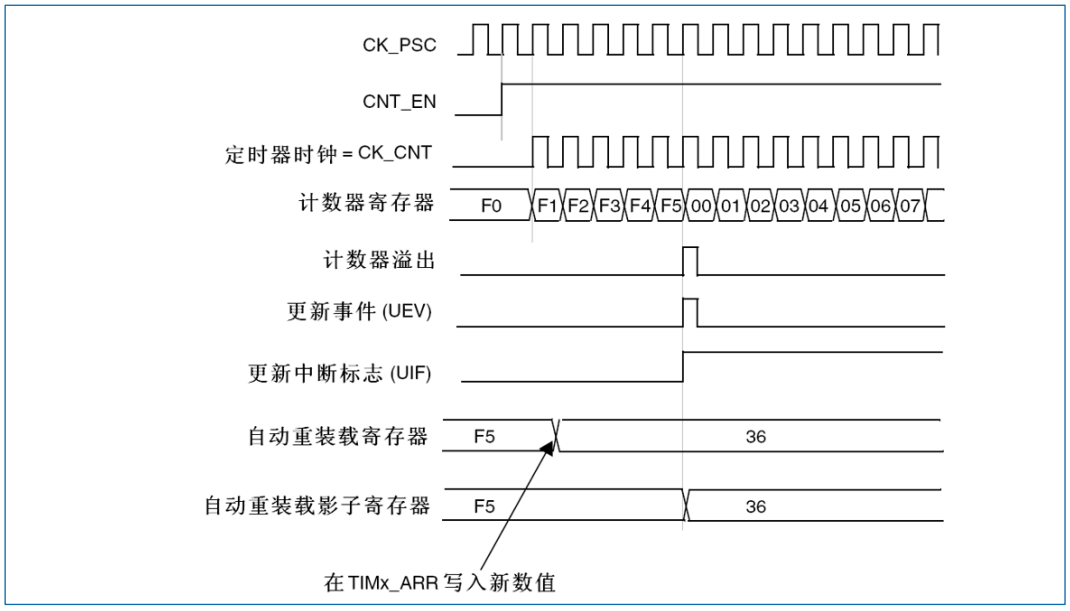

图 17-9 计数器时序图，当 ARPE=1 时的更新事件（预装载 TIM6_ARR）

17.2.3 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，只能通过软件改变它们

（除了 UG 位被自动清除外）。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。

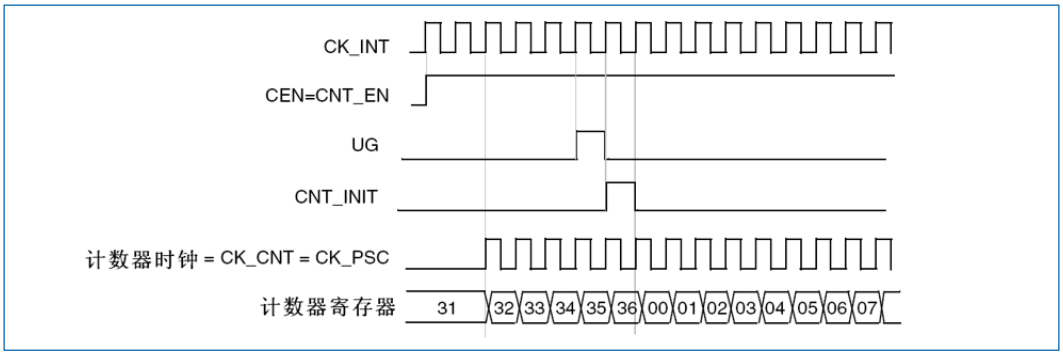

图 17-10 普通模式时序图，内部时钟分频系数为 1

17.2.4 调试模式

当微控制器进入调试模式（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。参见“24.8.2 对定时器、看门狗和 I2C 的调试支持”。

17.3 TIM6 寄存器

基地址：0x4000 1000

空间大小：0x400

17.3.1 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			OPM	URS	UDIS	CEN
								rw				rw	rw	rw	rw

位 15:8	Res：保留 必须保持复位值。
位 7	ARPE：自动重装载预装载允许位（Auto-reload preload enable） ● 0：TIM6_ARR 寄存器没有缓冲 ● 1：TIM6_ARR 寄存器有缓冲
位 6:4	Res：保留 必须保持复位值。
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或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： o 计数器溢出/下溢 o 设置 UG 位

	○ 从模式控制器产生的更新 ● 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
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： ○ 计数器溢出/下溢 ○ 设置 UG 位 ○ 从模式控制器产生的更新，具有缓存的寄存器被装入它们的预装载值。 ● 1：禁止 UEV ● 不产生更新事件，影子寄存器（ARR、PSC、CCRx）保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
位 0	CEN：使能计数器（Counter enable） ● 0：禁止计数器 ● 1：使能计数器

17.3.2 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） 该位域用于选择在主模式下向从定时器发送的同步信息（TRGO），有以下几种组合： ● 000：复位：使用 TIM6_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 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（见 TIM6_SMCR 寄存器的 MSM 位）。 ● 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。

17.3.3 TIM6 中断允许寄存器（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：更新中断禁止 ● 1：更新中断允许

17.3.4 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） 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 ● 0：无更新事件产生。 ● 1：更新中断等待响应。当以下寄存器的控制位被更新时，该位由硬件置’1’： ○ 若 TIM6_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 TIM6_CR1 寄存器的 URS=0 且 UDIS=0，当设置 TIM6_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 ○ 若 TIM6_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。

17.3.5 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：不起作用 ● 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也被清’0’（但是预分频系数不变）。

17.3.6 TIM6 计数器寄存器（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）

17.3.7 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]：预分频值（Prescaler value） 预分频器的值，计数器的时钟频率：CK_CNT = 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 TIM6_EGR 的 UG 位清零或被工作在复位模式的从控制器清”0”。

17.3.8 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]：自动重装载的值（Auto-reload value） ARR 包含了将要装载入实际的自动重装载寄存器的值。

18 独立看门狗（IWDG）

本系列芯片内置两个看门狗（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（表 18-1，表 19-1）时，将产生系统复位或触发一个中断。

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。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，请参看章节：“19 窗口看门狗（WWDG）”。

18.1 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）
- 看门狗被激活后，在计数器计数至 0x000 时产生复位
- IWDG 计数器复位初始值可由 Flash 选项字设置（请参见：“3.3 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 IWDG_LP_CTL 控制芯片进入停机（Stop）模式后 LSI 的状态。

存储的值为 0x369C 时，MCU 进入 Stop 模式后 IWDG 时钟被关闭，但 LSI 的状态受 LSION 位（RCC_CSR 寄存器）控制的状态保持不变；在 MCU 唤醒后，LSI 恢复成进 Stop 模式之前的状态。

如果不配置该位域，则在使能 IWDG 后再进入 Stop 模式，系统会被 IWDG 周期唤醒。用户可以通过配置 IWDG_LP_CTL 来决定在使能 IWDG 后再进入 Stop 模式时，是否需要被 IWDG 周期唤醒。

18.2 IWDG 功能描述

图 18-1 为独立看门狗模块的功能框图。

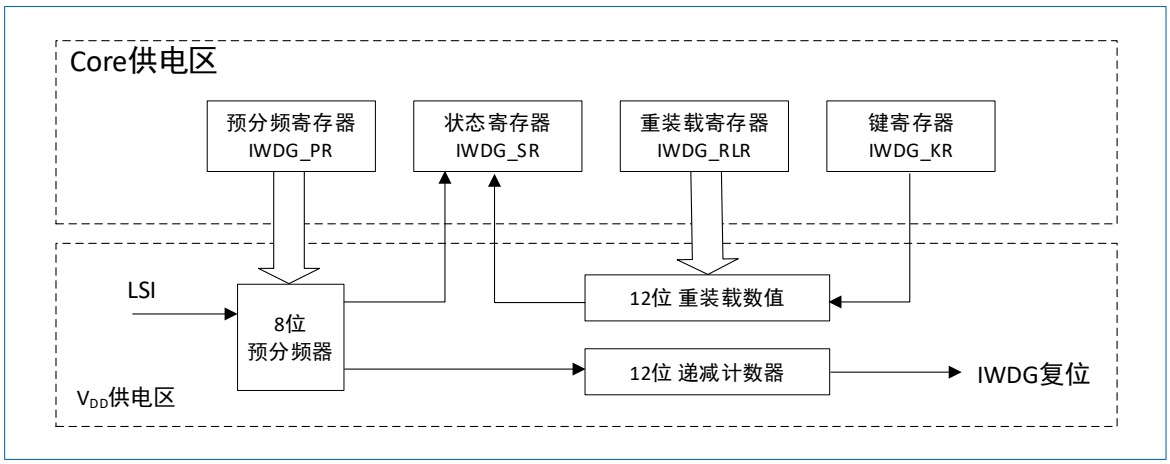

图 18-1 IWDG 框图

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，详细的描述请参考章节：“18.2.1 窗口选项”。

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 18-1 IWDG 超时时间表（40kHz 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	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	6 或 7	6.4	26214.4

注意：

这些时间是按照 40 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。

18.2.1 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。

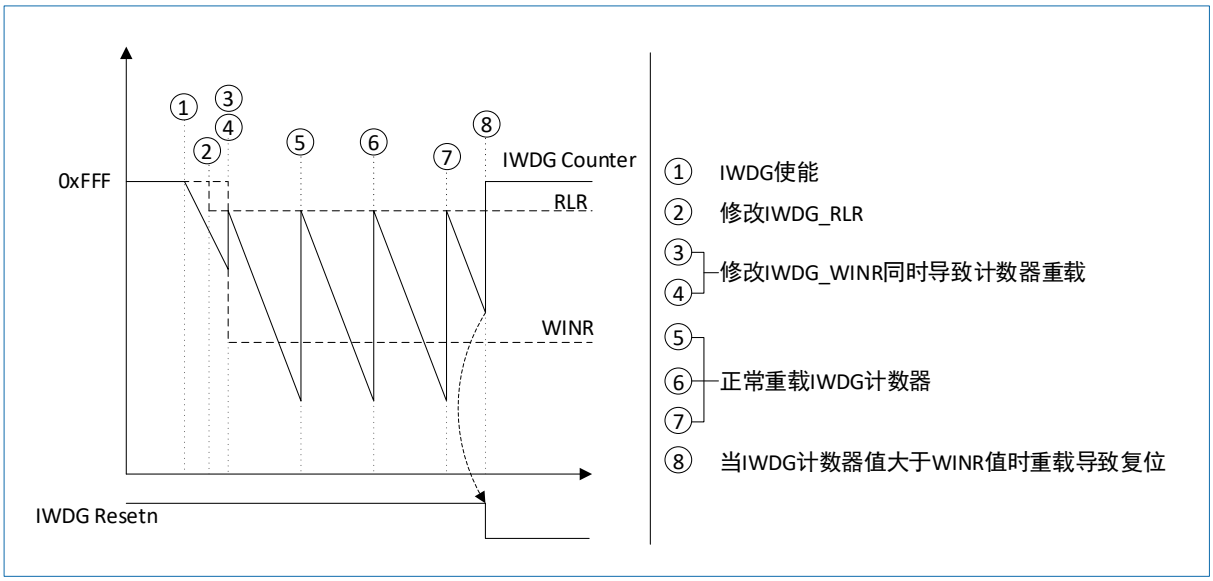

图 18-2 IWDG 使能窗口选项时的工作说明

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. 刷新计数器值为 IWDG_RLR 值（IWDG_KR = 0x0000 AAAA）。

18.2.2 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 0x0000 AAAA 以重载 IWDG 计数器，则系统会产生复位。

18.2.3 寄存器访问保护

IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器具有写保护功能。要修改这两个寄存器的值，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。

状态寄存器指示预分频值和递减计数器是否正在被更新。

18.2.4 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：“24.8.2 对定时器、看门狗和 I2C 的调试支持”。

18.3 IWDG 寄存器

基地址：0x4000 3000

空间大小：0x400

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

18.3.1 键寄存器（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（只写寄存器，读出值为 0x0000）（Key value （write only, read 0000h）） ● 软件必须以一定的间隔写入 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“18.2.3 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）。

18.3.2 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“18.2.3 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。要改变预分频因子，IWDG_SR 寄存器的 PVU 位必须为 0。

	• 000：4 分频 • 001：8 分频 • 010：16 分频 • 011：32 分频 • 100：64 分频 • 101：128 分频 • 110：256 分频 • 111：256 分频 *注意：对此寄存器进行读操作，将从 V_{DD} 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或是过期的。因此，只有当 IWDG_SR 寄存器的 PVU 位为 0 时，读出的值才有效。*

18.3.3 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]：看门狗计数器重装载值（Watchdog counter reload value） 该位域具有写保护功能，参看“18.2.3 寄存器访问保护”。用于定义看门狗计数器的重装载值，每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此重装载值和时钟预分频值来计算，参见表 18-1。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。 *注意：对此寄存器进行读操作，将从 V_{DD} 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或是过期的。因此，只有当 IWDG_SR 寄存器的 RVU 位为 0 时，读出的值才有效。*

18.3.4 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 31:2	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） 该位由硬件置位，用于表明正在更新窗口值。 当 V_{DD} 供电区域中完成了看门狗定时器的重加载时，该位会被硬件清零（需要最多 5 个 32kHz 的 RC 周

	期）。只有当 WVU 的值为 0 时，才能再次改写窗口值设置。该位只在窗口功能打开的时候有效。
位 1	RVU：看门狗计数器重装载值更新（Watchdog counter reload value update） 此位由硬件置'1'用来指示重装载值的更新正在进行中。当在 V_{DD} 域中的重装载更新结束后，此位由硬件清零（最多需 5 个 f_{LSI} 的 RC 周期）。 重装载值只有在 RVU 位被清零后才可更新。
位 0	PVU：看门狗预分频值更新（Watchdog prescaler value update） 此位由硬件置'1'用来指示预分频值的更新正在进行中。当在 V_{DD} 域中的预分频值更新结束后，此位由硬件清零（最多需 5 个 f_{LSI} 的 RC 周期）。 预分频值只有在 PVU 位被清零后才可更新。

注意：如果在应用程序中使用了多个重装载值或预分频值，则必须在RVU 位被清零后才能重新改变预装载值，在 PVU 位被清零后才能重新改变预分频值。然而，在更新预分频和/或重装载值后，不必等待 RVU 或 PVU 位清零，可继续执行下面的代码。（即使是在低功耗模式下，此写操作仍会被继续执行完成。）

18.3.5 窗口寄存器（IWDG_WINR）

偏移地址：0x10

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				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）

19 窗口看门狗（WWDG）

窗口看门狗通常用于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。

19.1 WWDG 主要特性

- 可编程的自由运行递减计数器。
- 复位条件：
  - 当看门狗被启动后，递减计数器的值小于 0x40。
  - 当看门狗被启动后，递减计数器在窗口外被重新装载。具体信息请参见：图 19-2。
- 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，它可以用于重装载计数器以避免 WWDG 复位。

19.2 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。

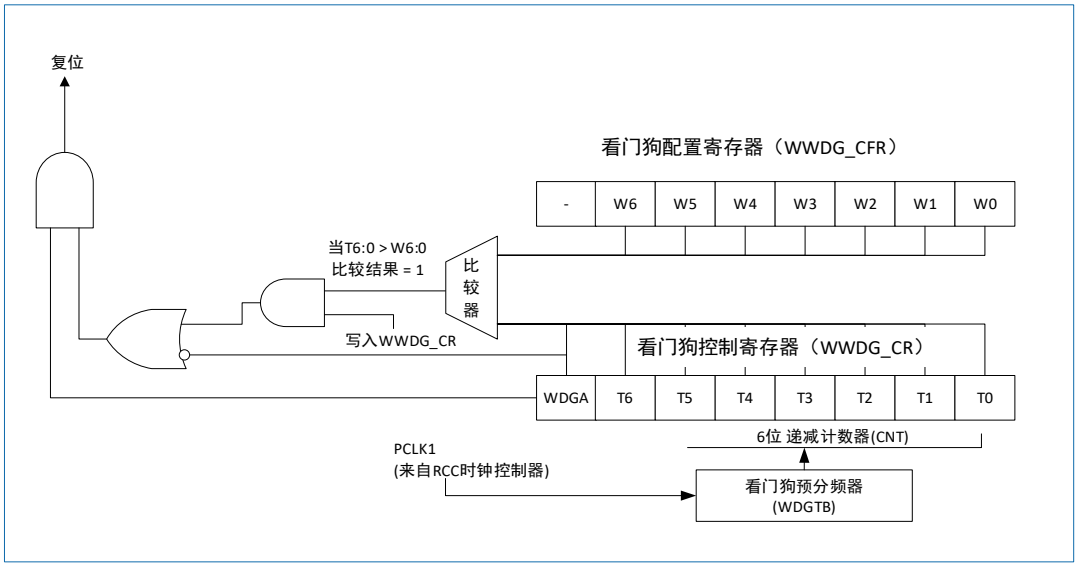

图 19-1 看门狗框图

应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。只有当计数器值小于配置寄存器的 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 WWDG_CFR.W[6:0]的数值并且大于 0x3F 时被重新装载，图 19-2 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'）。

19.3 如何编写看门狗超时程序

可以使用图 19-2 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。

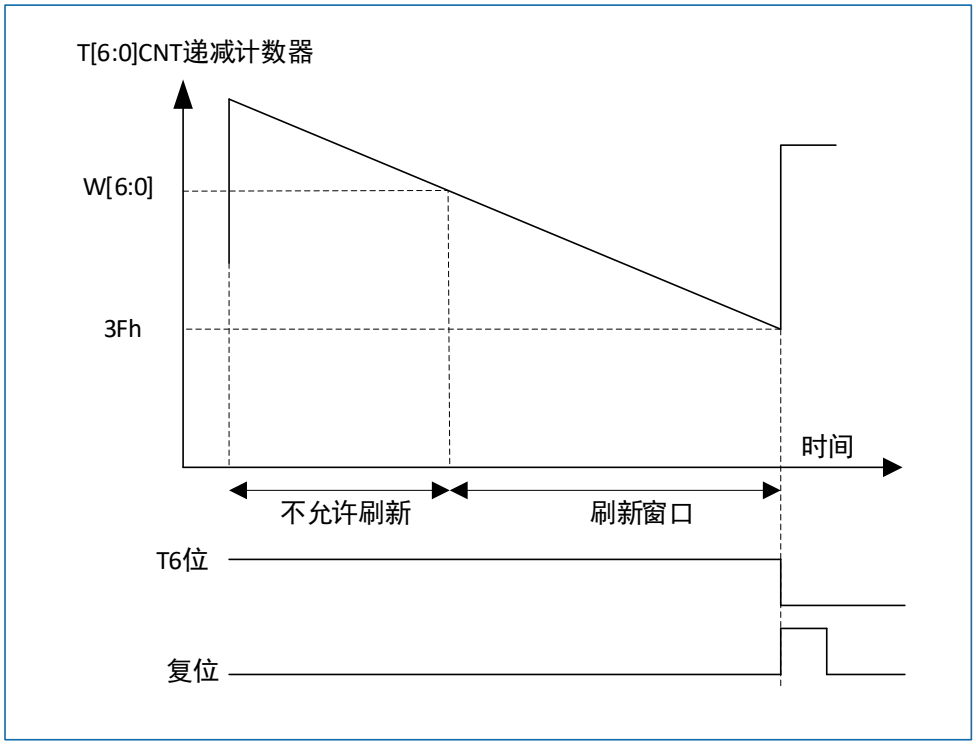

图 19-2 WWDG 时序图

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_{PCLK1}：APB1 以 ms 为单位的时钟周期，在 PCLK1=36MHz 时的最小/最大超时值（具体值如下表）。

表 19-1 WDGTB 超时值

WDGTB	最小超时值（T[5:0]=00 0000）	最大超时值（T[5:0]=11 1111）
0	113μs	7.28ms
1	227μs	14.56ms

WDGTB	最小超时值（T[5:0]=00 0000）	最大超时值（T[5:0]=11 1111）
2	455µs	29.12ms
3	910µs	58.25ms

19.4 调试模式

当微控制器进入调试模式时（Cortex®-M0 内核停止），根据调试模块中的 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：24.8.2 对定时器、看门狗和 I2C 的调试支持。

19.5 WWDG 寄存器

基地址：0x4000 2C00

空间大小：0x400

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

19.5.1 控制寄存器（WWDG_CR）

偏移地址：0x00

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								WDGA	T[6:0]						
								rw	rw						

位 31:8	Res：保留 必须保持复位值。
位 7	WDGA：激活位（Activation bit） 此位由软件置’1’，但仅能由硬件在复位后清’0’。当 WDGA=1 时，看门狗可以产生复位。 ● 0：禁止看门狗 ● 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 PCLK1 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。

19.5.2 配置寄存器（WWDG_CFR）

偏移地址：0x04

复位值：0x7F

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]						
						rw	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） 此位若置’1’，则当计数器值达到 0x40，即产生中断。此中断只能由硬件在复位后清除。
位 8:7	WDGTB[1:0]：时基（Timer base） 预分频器的时基可以设置如下： ● 00：CK 计时器时钟（PCLK1 除以 4096）除以 1 ● 01：CK 计时器时钟（PCLK1 除以 4096）除以 2 ● 10：CK 计时器时钟（PCLK1 除以 4096）除以 4 ● 11：CK 计时器时钟（PCLK1 除以 4096）除以 8
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用的窗口值。

19.5.3 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） 当计数器值达到 0x40 时，此位由硬件置’1’。它必须通过软件写’0’来清除。对此位写’1’无效。若中断未被使能，此位也会被置’1’。

20 实时时钟（RTC）

20.1 主要特性

- 可编程的预分频系数：分频系数最高为 2^{20}。
- 32 位的可编程计数器，可用于较长时间段的测量。
- 两个独立的时钟：用于 APB1 接口的 PCLK1 和 RTC 时钟（RTC 时钟的频率必须小于 PCLK1 时钟频率的四分之一以上）。
- 可以选择以下三种 RTC 的时钟源：
  - HSE 时钟除以 32
  - LSE 振荡器时钟
  - LSI 振荡器时钟
- 1 个独立的复位类型：
  - APB1 接口由系统复位
- 两个专门的可屏蔽中断（均可做停机模式下的唤醒源）：
  - 闹钟中断，用来产生一个软件可编程的闹钟中断。
  - 秒中断，用来产生一个可编程的周期性中断信号（最长可达 1 秒）。

20.2 RTC 功能描述

20.2.1 概述

RTC 由两个主要部分组成（参见下图）。

- 第一部分为 APB1 接口，用来和 APB1 总线相连。此单元还包含一组 16 位寄存器，可通过 APB1 总线对其进行读写操作（参见章节：“20.3 RTC 寄存器”）。APB1 接口由 APB1 总线时钟驱动，用来和 APB1 总线通信。
- 第二部分为 RTC 核心，由一组可编程计数器组成。RTC 核心包括两个主要模块：
  - 第一个模块是 RTC 的预分频模块，它可编程产生周期最长为 1 秒的 RTC 时间基准 TR_CLK。RTC 的预分频模块包含了一个 20 位的可编程分频器（RTC 预分频器）。如果在RTC_CR寄存器中设置了相应的允许位，则在每个TR_CLK 周期中，RTC 产生一个中断（秒中断）。
  - 第二个模块是一个 32 位的可编程计数器，可被初始化为当前的系统时间。系统时间按 TR_CLK 周期累加并与存储在 RTC_ALR 寄存器中的可编程时间相比较，如果 RTC_CR 控制寄存器中设置了相应允许位，比较匹配时将产生一个闹钟中断。

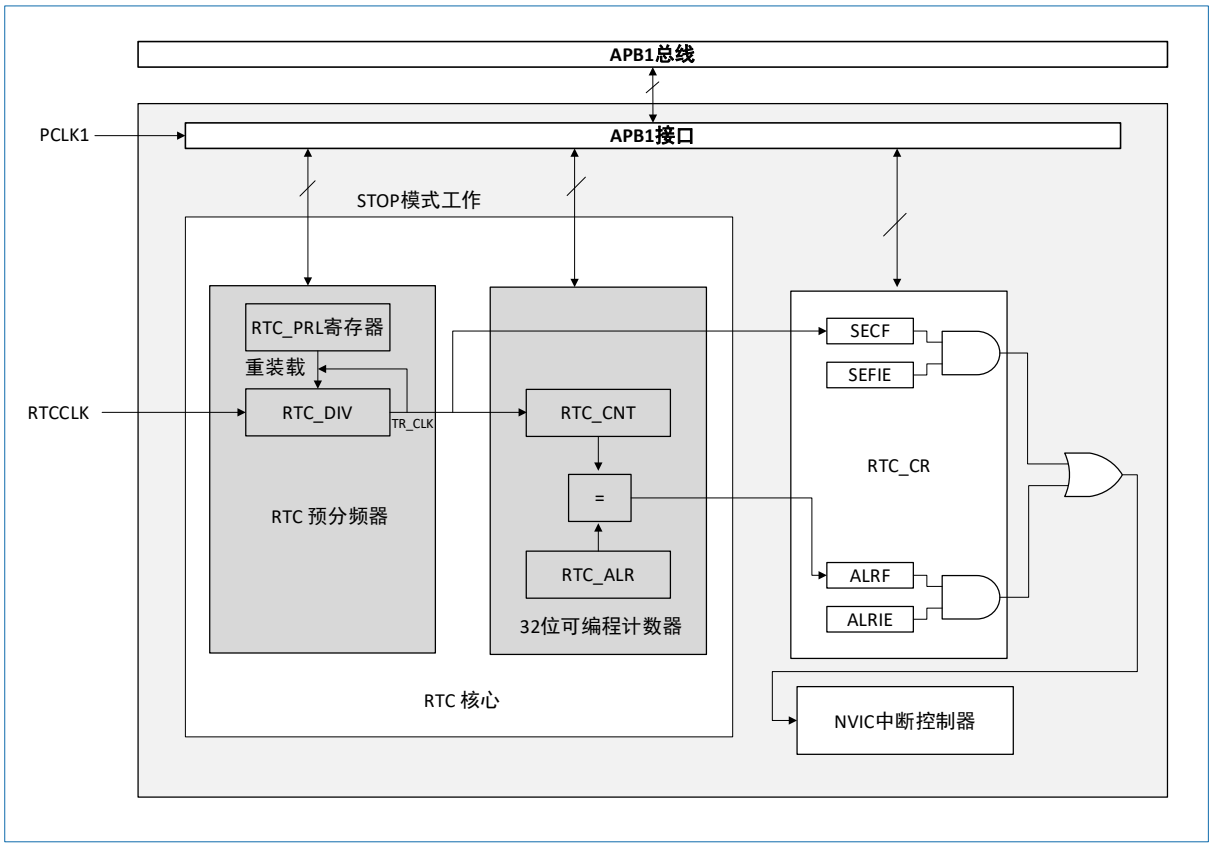

图 20-1 简化的 RTC 框图

20.2.2 RTC 复位

当前 RTC 所有的系统寄存器都由系统复位或电源复位进行异步复位。

20.2.3 读 RTC 寄存器

RTC 核完全独立于 RTC APB1 接口。软件通过 APB1 接口访问 RTC 的预分频值、计数器值和闹钟值。但是，相关的可读寄存器只在与 RTC APB1 时钟进行重新同步的 RTC 时钟的上升沿被更新。RTC 标志也是如此的。

这意味着，如果 APB1 接口曾经被关闭，而读操作又是在刚刚重新开启 APB1 接口之后，在第一次的内部寄存器更新之前，那么从 APB1 上读出的 RTC 寄存器数值可能被破坏了（通常读到 0）。下述这种情况下能够发生这种情形：

- 系统刚从停机模式唤醒（参见：“4.5 低功耗模式”）。

这种情况中，当 APB1 接口被禁止时（无时钟），RTC 核仍保持运行状态。

因此，在读取 RTC 寄存器时，如果 RTC 的 APB1 接口曾被禁止，那么软件首先必须等待 RTC_CRL 寄存器中的 RSF 位（寄存器同步标志）被硬件置’1’。

注意：RTC 的 APB1 接口不受 WFI 和 WFE 等低功耗模式的影响。

20.2.4 配置 RTC 寄存器

必须先设置RTC_CRL 寄存器中的CNF 位，RTC 进入配置模式后，才能写入RTC_PRL、RTC_CNT、RTC_ALR 寄存器。

另外，对RTC任何寄存器的写操作，都必须在前一次写操作结束后再进行。可以通过查询 RTC_CR 寄存器中的 RTOFF 状态位，判断 RTC 寄存器是否处于更新中。只有当 RTOFF 状态位是’1’时，才可以写入 RTC 寄存器。

配置过程：

1. 查询 RTOFF 位，直到 RTOFF 的值变为’1’。
2. 置 CNF 值为 1，进入配置模式。
3. 对一个或多个 RTC 寄存器进行写操作。
4. 清除 CNF 标志位，退出配置模式。
5. 查询 RTOFF，直至 RTOFF 位变为’1’以确认写操作已经完成。

只有当 CNF 标志位被清除时，写操作才能进行，这个过程至少需要 3 个 RTCCLK 周期。

20.2.5 RTC 标志的设置

在每一个 RTC 核的时钟周期中，在更改 RTC 计数器之前，RTC 秒标志（SECF）被设置。

在计数器的值到达闹钟寄存器的值加 1（RTC_ALR+1）之前的最后一个 RTC 时钟周期中，设置 RTC_Alarm 和 RTC 闹钟标志（ALRF）。对 RTC 闹钟的写操作必须使用下述过程之一与 RTC 秒标志（SECF）同步：

- 使用 RTC 闹钟中断，并在中断处理程序中修改 RTC 闹钟和/或 RTC 计数器。
- 等待 RTC 控制寄存器中的 SECF 位被设置，再更改 RTC 闹钟和/或 RTC 计数器。

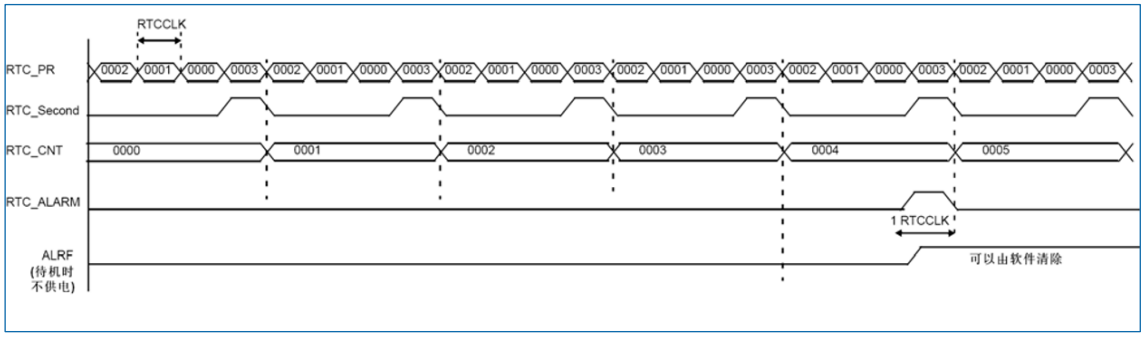

图 20-2 RTC 秒和闹钟波形图示例，PR=0003，ALARM=00004

20.3 RTC 寄存器

基地址：0x4000 0C00

空间大小：0x400

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

20.3.1 RTC 控制寄存器

20.3.1.1 RTC 控制寄存器高位（RTC_CRH）

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														ALRIE	SECIE
														rw	rw

位 31:2	Res：保留 必须保持复位值。
位 1	ALRIE：允许闹钟中断（Alarm interrupt enable） ● 0：屏蔽（不允许）闹钟中断 ● 1：允许闹钟中断
位 0	SECIE：允许秒中断（Second interrupt enable） ● 0：屏蔽（不允许）秒中断 ● 1：允许秒中断

这些位用来屏蔽中断请求。

注意：系统复位后所有的中断被屏蔽，因此可通过写 RTC 寄存器来确保在初始化后没有挂起的中断请求。当外设正在完成前一次写操作时（标志位 RTOFF=0），不能对 RTC_CRH 寄存器进行写操作。RTC 功能由这个控制寄存器控制。一些位的写操作必须经过一个特殊的配置过程来完成（详见"20.2.4 配置 RTC 寄存器"）。

20.3.1.2 RTC 控制寄存器低位（RTC_CRL）

偏移地址：0x04

复位值：0x0000 0020

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										RTOFF	CNF	RSF	Res	ALRF	SECF
										r	rw	rc_w0		rc_w0	rc_w0

位 31:6	Res：保留 必须保持复位值。
位 5	RTOFF：RTC 操作关闭（RTC operation OFF） RTC 模块利用这位来指示对其寄存器进行的最后一次操作的状态，指示操作是否完成。若此位为'0'，则表示无法对任何的 RTC 寄存器进行写操作。此位为只读位。 ● 0：上一次对 RTC 寄存器的写操作仍在进行 ● 1：上一次对 RTC 寄存器的写操作已经完成
位 4	CNF：配置标志（Configuration flag） 此位必须由软件置'1'以进入配置模式，从而允许向 RTC_CNT、RTC_ALR 或 RTC_PRL 寄存器写入数据。只有当此位在被置'1'并重新由软件清零后，才会执行写操作。 ● 0：退出配置模式（开始更新 RTC 寄存器） ● 1：进入配置模式
位 3	RSF：寄存器同步标志（Registers synchronized flag） 每当 RTC_CNT 寄存器和 RTC_DIV 寄存器由软件更新或清零时，此位由硬件置'1'。在 APB1 复位后，或 APB1 时钟停止后，此位必须由软件清零。要进行任何的读操作之前，用户程序必须等待这位被硬件置'1'，以确保 RTC_CNT、RTC_ALR 或 RTC_PRL 已经被同步。 ● 0：寄存器尚未被同步 ● 1：寄存器已经被同步

位 2	Res：保留 必须保持复位值。
位 1	ALRF：闹钟标志（Alarm flag） 当 32 位可编程计数器达到 RTC_ALR 寄存器所设置的预定值，此位由硬件置‘1’。如果 RTC_CRH 寄存器中 ALRIE=1，则产生中断。此位只能由软件清零。对此位写‘1’是无效的。 ● 0：无闹钟 ● 1：有闹钟
位 0	SECF：秒标志（Second flag） 当 32 位可编程预分频器溢出时，此位由硬件置‘1’同时 RTC 计数器加 1。因此，此标志为分辨率可编程的 RTC 计数器提供一个周期性的信号（通常为 1 秒）。如果 RTC_CRH 寄存器中 SECIE=1，则产生中断。此位只能由软件清除。对此位写‘1’是无效的。 ● 0：秒标志条件不成立 ● 1：秒标志条件成立

RTC 的功能由这个控制寄存器控制。当前一个写操作还未完成时（RTOFF=0 时，详见：“20.2.4 配置 RTC 寄存器”），不能写 RTC_CR 寄存器。

注意：

- *任何标志位都将保持挂起状态，直到适当的 RTC_CR 请求位被软件复位，表示所请求的中断已经被接受。*
- *在复位时禁止所有中断，无挂起的中断请求，可以对 RTC 寄存器进行写操作。*
- *当 APB1 时钟不运行时，ALRF、SECF 和 RSF 位不被更新。*
- *ALRF、SECF 和 RSF 位只能由硬件置位，由软件来清零。*
- *若 ALRF=1 且 ALRIE=1，则允许产生 RTC 全局中断。如果在 EXTI 控制器中允许产生 EXTI 线 17 中断，则允许产生 RTC 全局中断和 RTC 闹钟中断。*
- *若 ALRF=1，如果在 EXTI 控制器中设置了 EXTI 线 17 的中断模式，则允许产生 RTC 闹钟中断；如果在 EXTI 控制器中设置了 EXTI 线17 的事件模式，则这条线上会产生一个脉冲（不会产生 RTC 闹钟中断）。*

20.3.2 RTC 预分频装载寄存器

预分频装载寄存器用来保存 RTC 预分频器的周期计数值。它们受 RTC_CR 寄存器的 RTOFF 位写保护，仅当 RTOFF 值为‘1’时允许进行写操作。

20.3.2.1 RTC 预分频装载寄存器高位（RTC_PRLH）

偏移地址：0x08

复位值：0x0000 0000

只写（参见“20.2.4 配置 RTC 寄存器”）

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												PRL[19:16]			
												w			

位 31:4	Res：保留 必须保持复位值。
位 3:0	PRL[19:16]：RTC 预分频装载值高位（RTC prescaler reload value high） 根据以下公式，该位域结合 RTC_PRLL 用来定义计数器的时钟频率： $f_{TR\_CLK}=f_{RTCCLK}/$（PRL[19:0]+1） *注意：不推荐使用 0 值，否则无法正确的产生 RTC 中断和标志位。*

20.3.2.2 RTC 预分频装载寄存器低位（RTC_PRLL）

偏移地址：0x0C

复位值：0x0000 8000

只写（参见:"20.2.4 配置 RTC 寄存器"）

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
PRL[15:0]															
w															

位 31:16	Res：保留 必须保持复位值。
位 15:0	PRL[15:0]：RTC 预分频装载值低位（RTC prescaler reload value low） 根据以下公式，该位域结合 RTC_PRLH.PRL[19:16]用来定义计数器的时钟频率：$f_{TR\_CLK}= f_{RTCCLK}/$（PRL[19:0]+1）

注意：如果输入时钟频率是 32.768kHz（f_{RTCCLK}），这个寄存器中写入 7FFFH 可获得周期为 1 秒钟的信号。

20.3.3 RTC 预分频器余数寄存器

在 TR_CLK 的每个周期里，RTC 预分频器中计数器的值都会被重新设置为 RTC_PRL 寄存器的值。用户可通过读取 RTC_DIV 寄存器，获得预分频计数器的当前值，而不停止分频计数器的工作，从而获得精确的时间测量。此寄存器是只读寄存器，其值在 RTC_PRL 或 RTC_CNT 寄存器中的值发生改变后，由硬件重新装载。

20.3.3.1 RTC 预分频器余数寄存器高位（RTC_DIVH）

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												RTC_DIV[19:16]			
												r			

位 31:4	Res：保留

	必须保持复位值。
位 3:0	RTC_DIV[19:16]：RTC 时钟分频器余数高位（RTC clock divider high）

20.3.3.2 RTC 预分频器余数寄存器低位（RTC_DIVL）

偏移地址：0x14

复位值：0x0000 8000

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
RTC_DIV[15:0]															
r															

位 31:16	Res：保留 必须保持复位值。
位 15:0	RTC_DIV[15:0]：RTC 时钟器余数低位（RTC clock divider low）

20.3.4 RTC 计数器寄存器

RTC 核有一个 32 位可编程的计数器，可通过两个 16 位的寄存器访问。计数器以预分频器产生的 TR_CLK 时间基准为参考进行计数。RTC_CNT 寄存器用来存放计数器的计数值。它们受 RTC_CR 的位 RTOFF 写保护，仅当 RTOFF 值为'1'时，允许写操作。在高或低寄存器（RTC_CNTH 或 RTC_CNTL）上的写操作，能够直接装载到相应的可编程计数器，并且重新装载 RTC 预分频器。当进行读操作时，直接返回计数器内的计数值（系统时间）。

20.3.4.1 RTC 计数器寄存器高位（RTC_CNTH）

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
RTC_CNT[31:16]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	RTC_CNT[31:16]：RTC 计数器高位（RTC counter high） 可通过读 RTC_CNTH 寄存器来获得 RTC 计数器当前值的高位部分。要对此寄存器进行写操作前，必须先进入配置模式（参见"20.2.4 配置 RTC 寄存器"）。

20.3.4.2 RTC 计数器寄存器低位（RTC_CNTL）

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
RTC_CNT[15:0]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	RTC_CNT[15:0]：RTC 计数器低位（RTC counter low） 可通过读 RTC_CNTL 寄存器来获得 RTC 计数器当前值的低位部分。要对此寄存器进行写操作，必须先进入配置模式（参见“20.2.4 配置 RTC 寄存器”）。

20.3.5 RTC 闹钟寄存器

当可编程计数器的值与 RTC_ALR 中的 32 位值相等时，即触发一个闹钟事件，并且产生 RTC 闹钟中断。此寄存器受 RTC_CR 寄存器里的 RTOFF 位写保护，仅当 RTOFF 值为’1’时，允许写操作。

20.3.5.1 RTC 闹钟寄存器高位（RTC_ALRH）

偏移地址：0x20

复位值：0x0000 FFFF

只写（参见“20.2.4 配置 RTC 寄存器”）

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
RTC_ALR[31:16]															
w															

位 31:16	Res：保留 必须保持复位值。
位 15:0	RTC_ALR[31:16]：RTC 闹钟值高位（RTC alarm high） 此寄存器用来保存由软件写入的闹钟时间的高位部分。要对此寄存器进行写操作，必须先进入配置模式（参见“20.2.4 配置 RTC 寄存器”）。

20.3.5.2 RTC 闹钟寄存器低位（RTC_ALRL）

偏移地址：0x24

复位值：0x0000 FFFF

只写（参见“20.2.4 配置 RTC 寄存器”）

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
RTC_ALR[15:0]															
w															

位 31:16	Res：保留

	必须保持复位值。
位 15:0	RTC_ALR[15:0]：RTC 闹钟值低位（RTC alarm low） 此寄存器用来保存由软件写入的闹钟时间的低位部分。要对此寄存器进行写操作，必须先进入 配置模式（参见“20.2.4 配置 RTC 寄存器”）。

21 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）、快速模式（Fast mode，Fm）和超快速模式（Fast mode plus，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。

该接口还支持 DMA 数据传输方式，减轻 CPU 的工作量。

21.1 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 kHz）
  - 支持快速模式（高达 400 kHz）
  - 支持超快速模式（高达 1 MHz）
  - 支持 7 位和 10 位寻址模式
  - 支持多个 7 位从地址（两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
  - 可软件配置总线上的数据建立和保持时间
  - 支持事件管理
  - 支持时钟延展功能
  - 支持软件复位
- 带 DMA 功能的 1 字节缓冲
- 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

21.2 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 ）、快速（高达 400 kbit/s）或超快速（高达 1 Mbit/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）。

21.2.1 I2C 框图

I2C 接口的框图如图 21-1 所示。

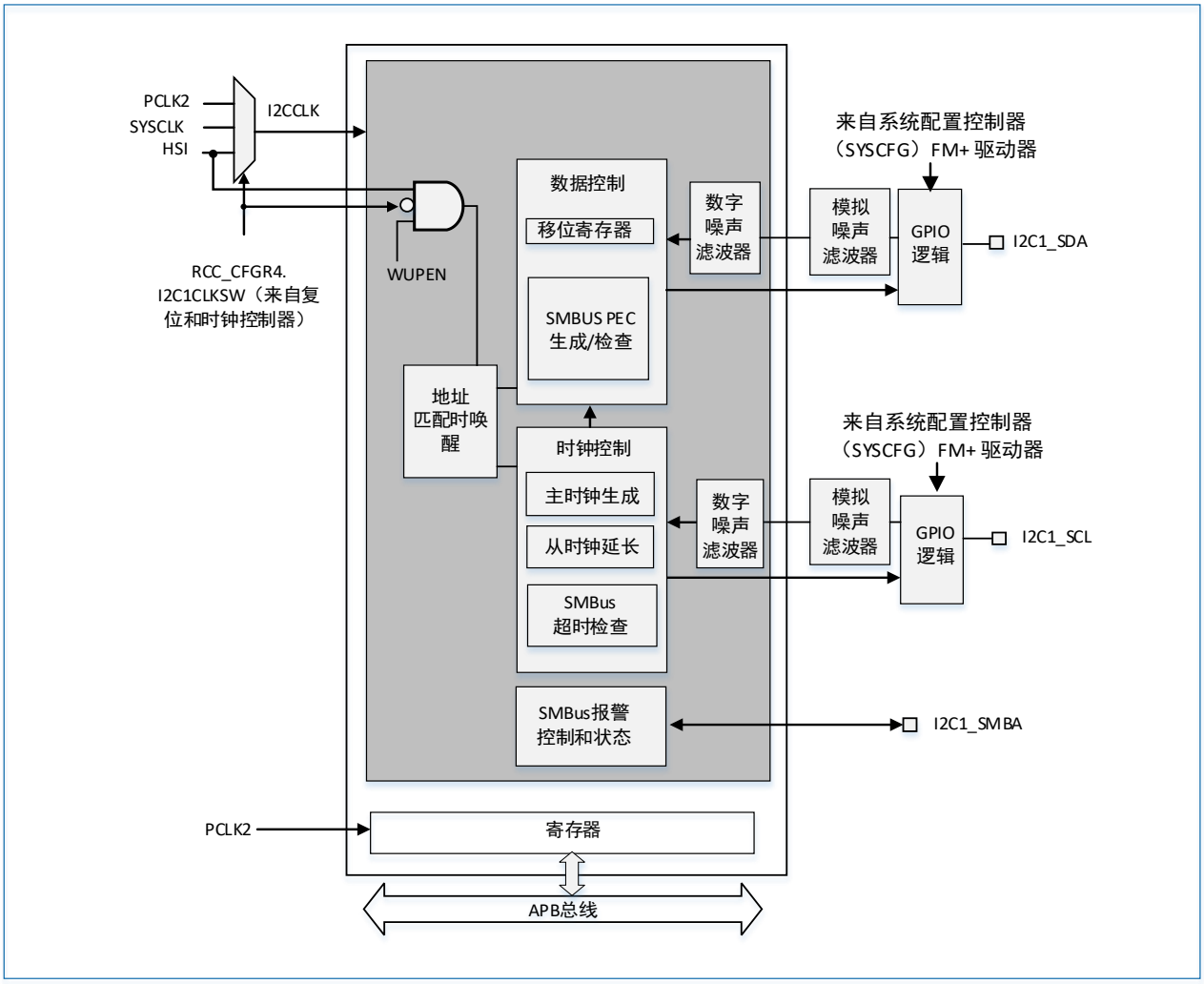

图 21-1 I2C 框图

I2C 的时钟可由独立时钟源提供，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：

- PCLK2：APB2 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“5.2 时钟”。

I2C I/O 可用于驱动超快速操作。将 SCL 和 SDA 的驱动能力控制位置 1 可使能该功能，该位域位于“7.4.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）”。

21.2.2 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}，DNF 的值可通过查询 I2C1_CR1 寄存器得到。

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。

注意：当 I2C 内核的时钟由 PCLK 提供时，PCLK 必须遵循 t_{I2CCLK} 的条件。

21.2.3 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。

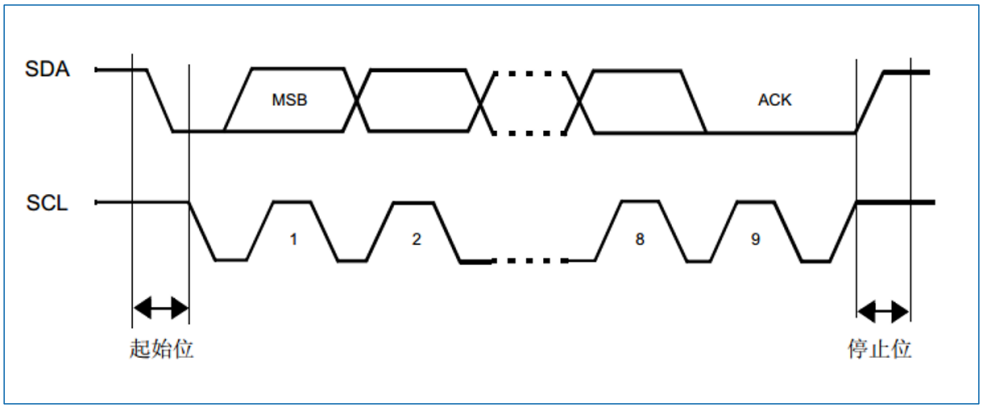

图 21-2 I2C 总线协议

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。

21.2.4 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“5.3.19 时钟配置寄存器 4（RCC_CFGR4）”）。通过将 I2C1_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“21.2.5 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和超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 I2C1_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 DNFxI2CCLK 个周期后才会发生内部变化。

表 21-1 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。

注意：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 I2C1_TIMINGR 寄存器中的 PRESC[3:0]、SCLDEL[3:0]和 SDADEL[3:0]位。

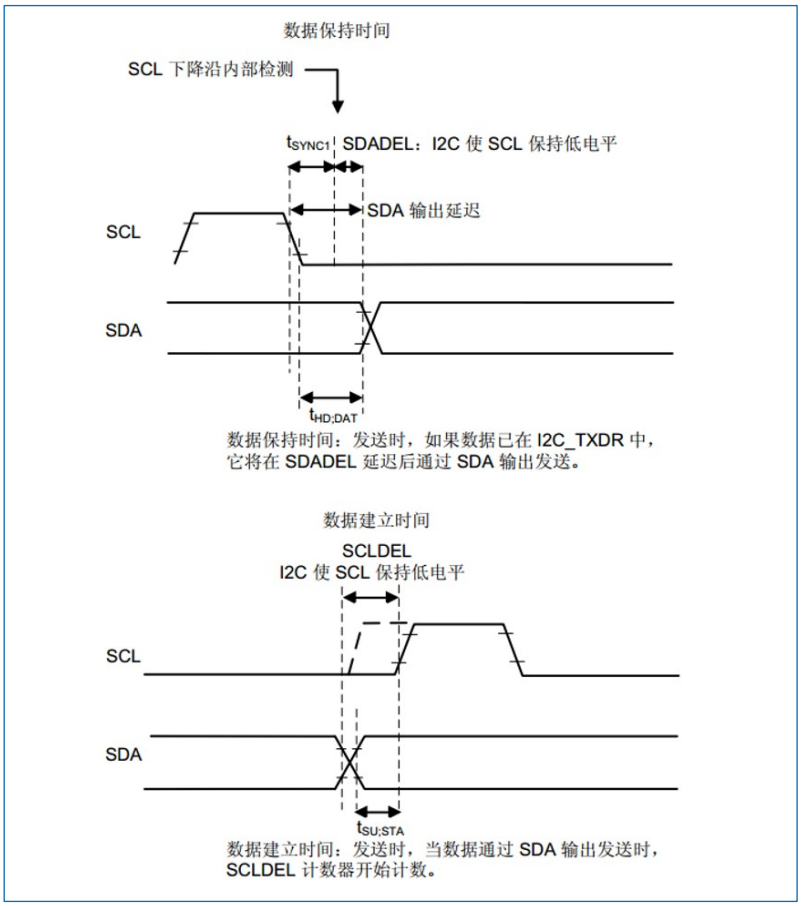

图 21-3 建立和保持时序

- 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}}≤SDADEL
- SDADEL≤{$t_{HD;DAT（max）}$-$t_{AF（max）}$-[（DNF+4）xt_{I2CCLK}]}/{（PRESC+1）xt_{I2CCLK}}

注意：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，请参见数据手册。

标准模式、快速模式和超快速模式下的 $T_{HD;DAT}$ 最大值分别可达 3.45 µs、0.9 µs 和 0.45 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 ≤{$t_{VD;DAT}$（max）-t_R（max）-260ns-[（DNF+4）xt_{I2CCLK}]}/{（PRESC+1）xt_{I2CCLK}}

注意：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 21-2。

- 在 t_{SDADEL} 延时后，或在因数据未写入 I2C1_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）xt_{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）]xt_{I2CCLK}]}-1<=SCLDEL

有关 tr 和 $t_{SU;DAT}$ 标准值的信息，请参见表 21-2。

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。

注意：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 I2C1_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 21-2 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)		超快速模式(Fm+)		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值	
$t_{HD;DAT}$	数据保持时间	0	-	0	-	0	-	0.3	-	μs
$t_{VD;DAT}$	数据有效时间	-	3.45	-	0.9	-	0.45	-	-	
$t_{SU;DAT}$	数据建立时间	250	-	100	-	50	-	250		ns
t_r	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	120	-	1000	
t_f	SDA 和 SCL 信号的下降时间	-	300	-	300	-	120	-	300	

此外，在主模式下，必须通过编程 I2C1_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“21.2.8 主模式”中的“I2C 主模式初始化”。

注意：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“21.2.7 从模式”中的“I2C 从模式初始化”。

注意：使能 I2C 后，不允许更改 NOSTRETCH 配置。

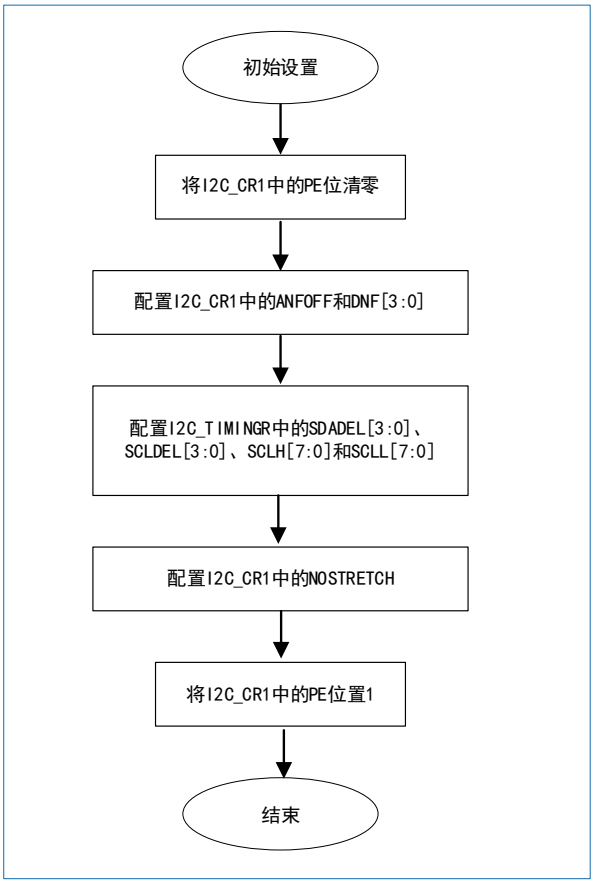

图 21-4 I2C 初始化流程图

21.2.5 软件复位

可通过将 I2C1_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：

- I2C1_CR2 寄存器：START、STOP 和 NACK
- I2C1_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：

- I2C1_CR2 寄存器：PECBYTE
- I2C1_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。

21.2.6 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 I2C1_RXDR 寄存器为空（RXNE=0），则移位寄存器的内容会复制到 I2C1_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 主器件时，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 I2C1_RXDR 为止。

作为I2C从器件时，如果NOSTRETCH=0，将延长SCL线的低电平时间，直到读取了I2C1_RXDR为止。如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

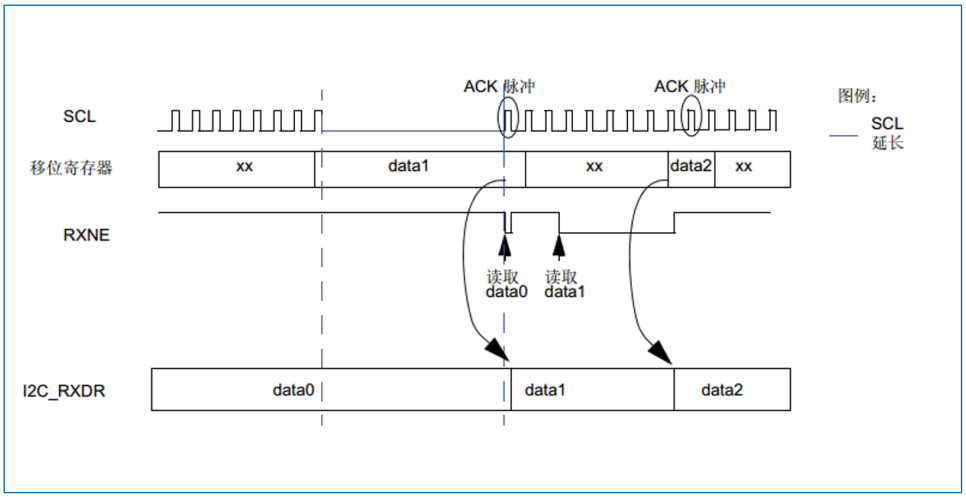

图 21-5 数据接收

发送

如果 I2C1_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 I2C1_TXDR 内尚未写入任何数据。

作为 I2C 主器件，硬件将延长第 9 个 SCL 脉冲后的低电平，直到写入了 I2C1_TXDR 为止。

作为 I2C 从器件，如果 NOSTRETCH=0，将延长时钟，直到软件写 I2C1_TXDR；如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

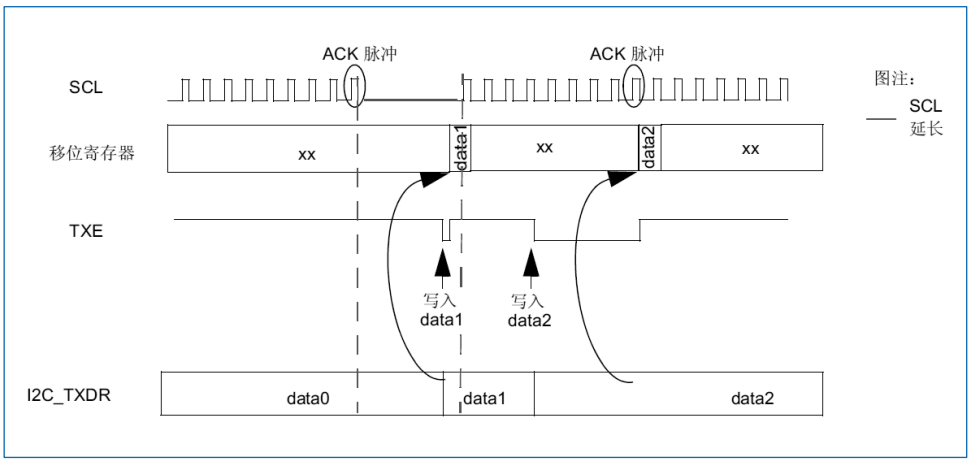

图 21-6 数据发送

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 I2C1_CR1 寄存器中的 SBC（从字节控制）位置 1。

待传输的字节数在 I2C1_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 I2C1_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（I2C1_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（I2C1_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 I2C1_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。

注意：当 RELOAD 位置 1 时，AUTOEND 位将不起作用。

表 21-3 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

21.2.7 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 I2C1_OAR1 和 I2C1_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 I2C1_OAR1 寄存器 OA1MODE 位置 1 配置为 10 位寻址模式。

  通过将 I2C1_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 I2C1_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 I2C1_OAR1 或 I2C1_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 I2C1_OAR2 寄存器中的 OA2EN 位置 1 来使能 OA2。

- 通过将 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_RXDR 寄存器但新的数据接收已完成。读取 I2C1_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 I2C1_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 I2C1_TXDR 寄存器写入数据。否则，会发生下溢，I2C1_ISR 寄存器中的 OVR 标志将置 1，如果 I2C1_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 I2C1_RXDR 寄存器读取数据。否则，会发生上溢，I2C1_ISR 寄存器中的 OVR 标志将置 1，如果 I2C1_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 I2C1_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 NBYTES 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 I2C1_RXDR 寄存器中读取数据，然后通过配置 I2C1_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。

注意：

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。

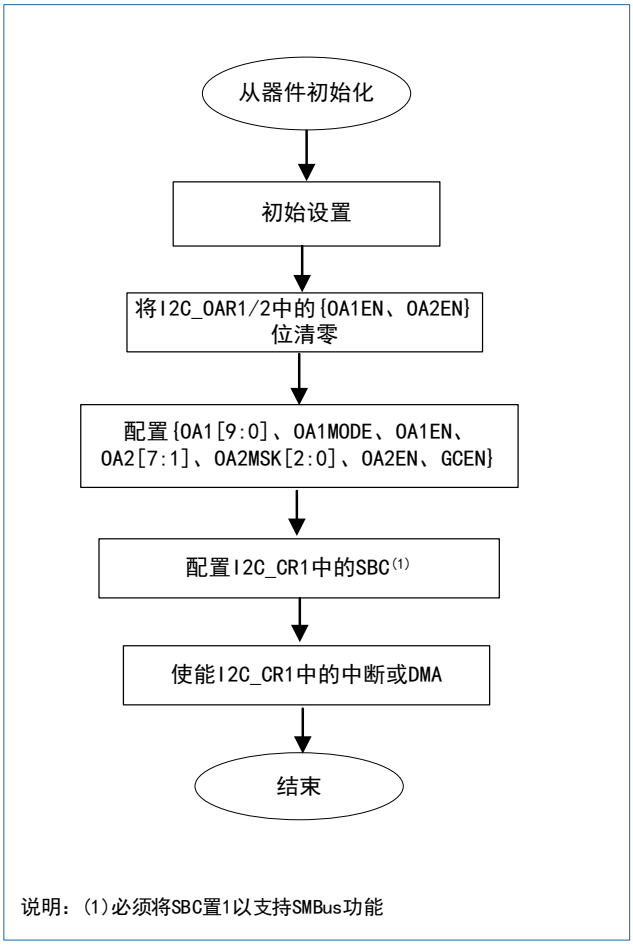

图 21-7 从器件初始化流程图

从发送器

当 I2C1_TXDR 寄存器为空时，将生成发送中断状态（TXIS）。如果 I2C1_CR1 寄存器中的 TXIE 位置 1，将生成中断。

I2C1_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，I2C1_ISR 寄存器中的 NACKF 位将置 1，如果 I2C1_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 I2C1_CR1 寄存器中的 STOPIE 位置 1 时，I2C1_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 I2C1_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 I2C1_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 NBYTES 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。

注意：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 I2C1_TXDR 寄存器的内容，从而编程第一个数据字节。必须在 I2C1_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 I2C1_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（发送中断或发送 DMA 请求），用户必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

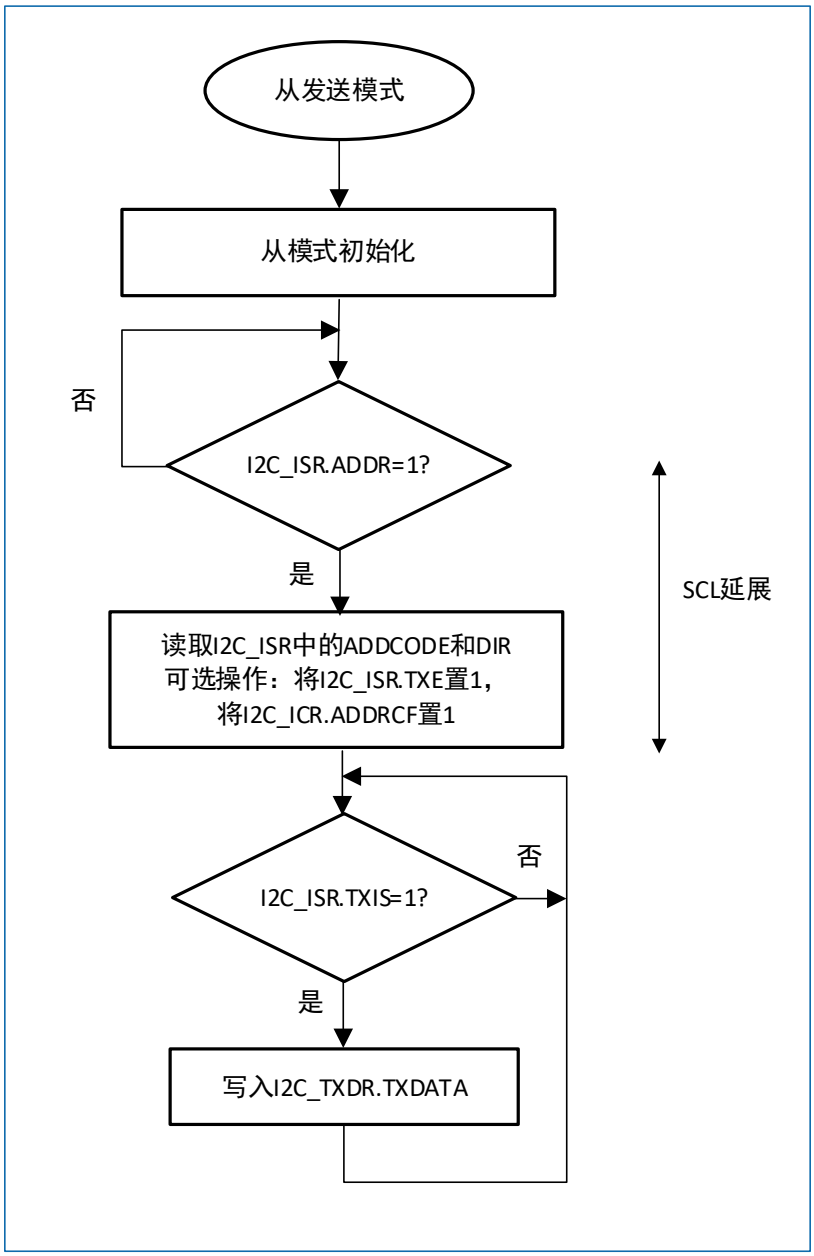

图 21-8 I2C 从发送器的传输序列流程图（NOSTRETCH=0）

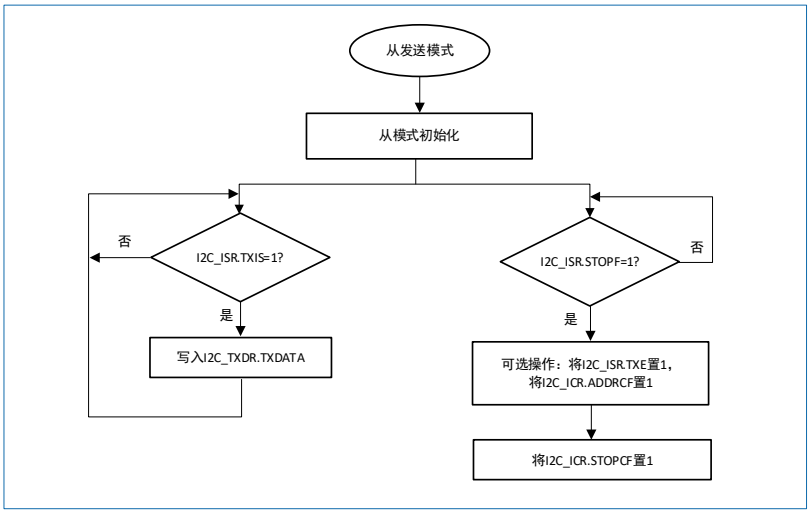

图 21-9 I2C 从发送器的传输序列流程图（NOSTRETCH=1）

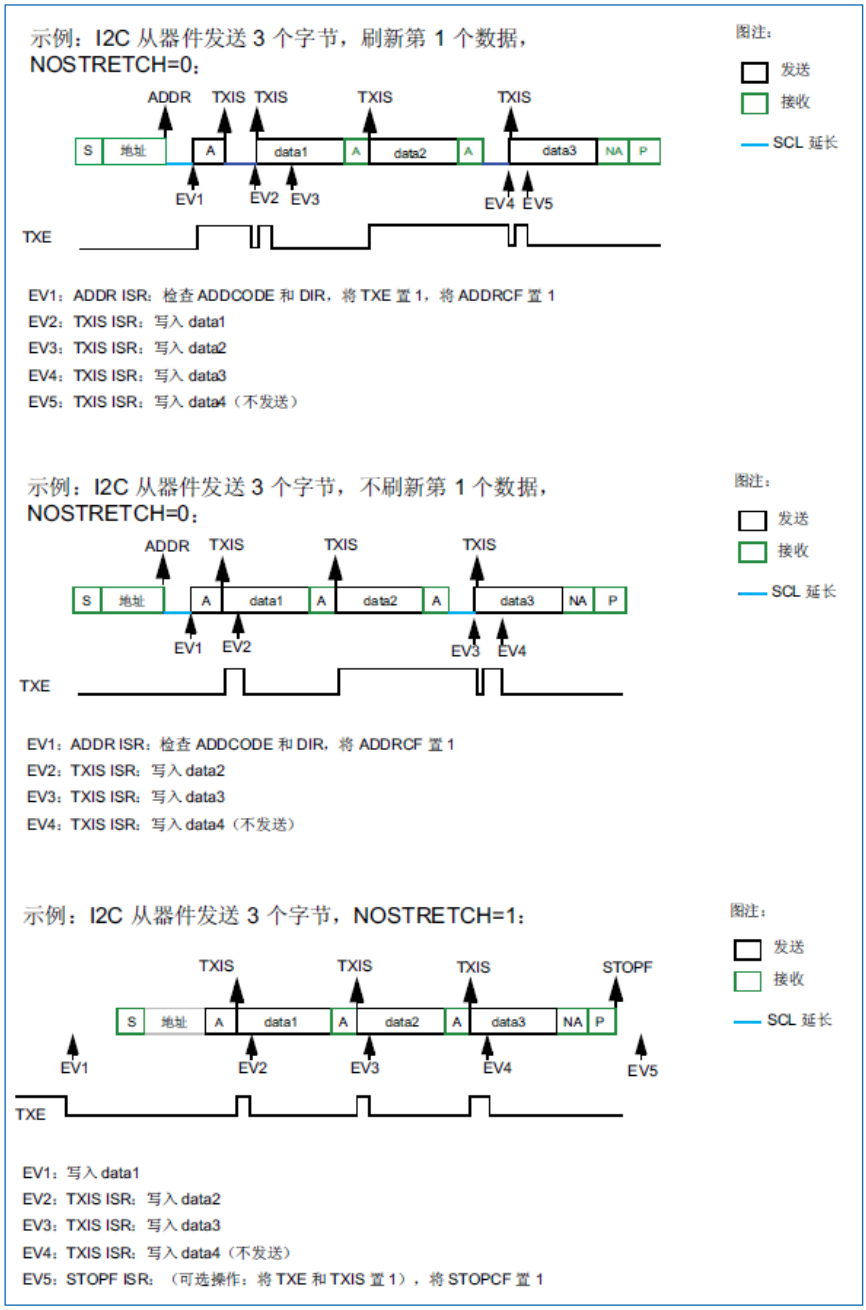

图 21-10 从发送器的传输总线图

从接收器

当 I2C1_RXDR 满时，I2C1_ISR 中的 RXNE 将置 1，如果 I2C1_CR1 中的 RXIE 置 1，还将生成中断。读取 I2C1_RXDR 时，将清零 RXNE。

接收到停止条件且 I2C1_CR1 寄存器中的 STOPIE 置 1 时，I2C1_ISR 中的 STOPF 将置 1 并且会生成中断。

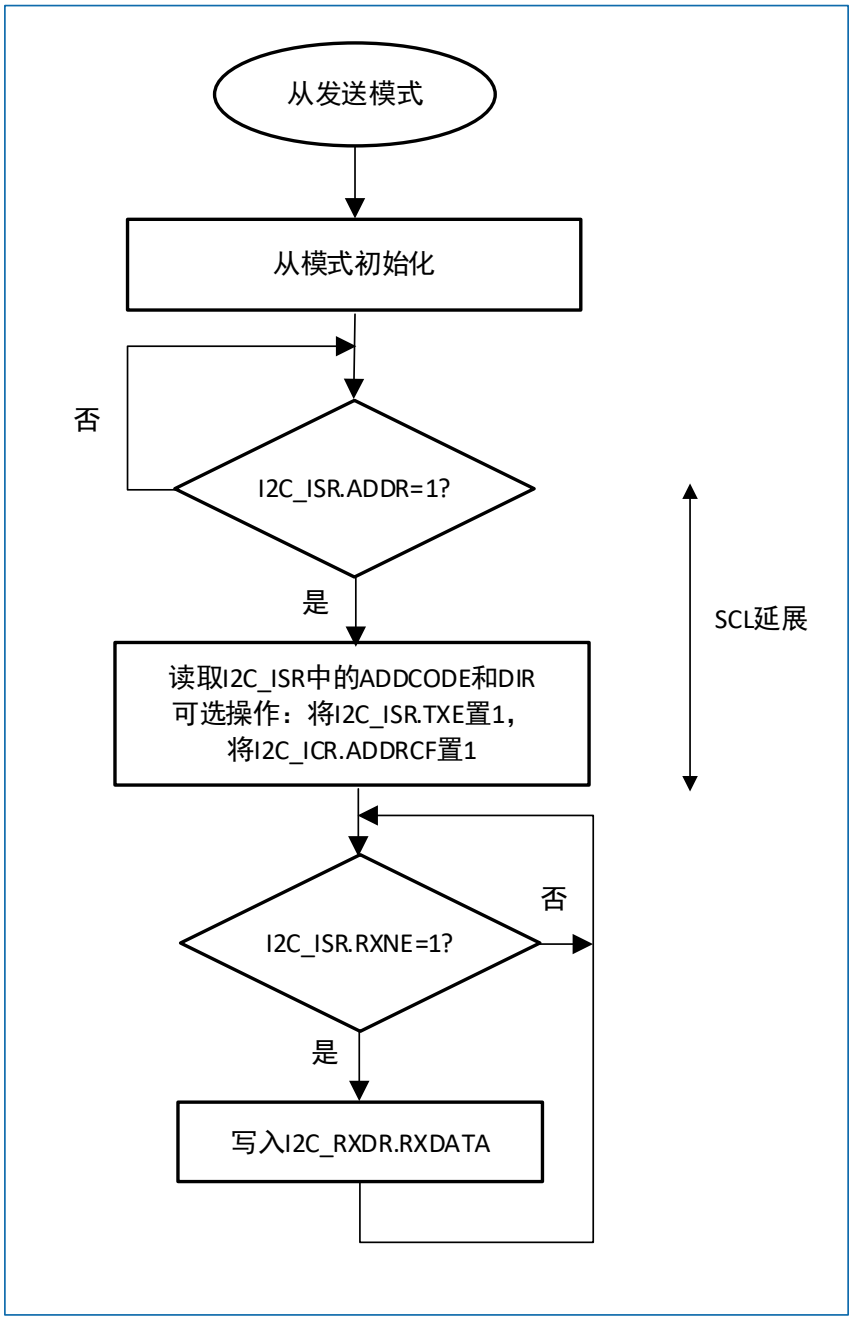

图 21-11 从接收器的传输序列流程图（NOSTRETCH=0）

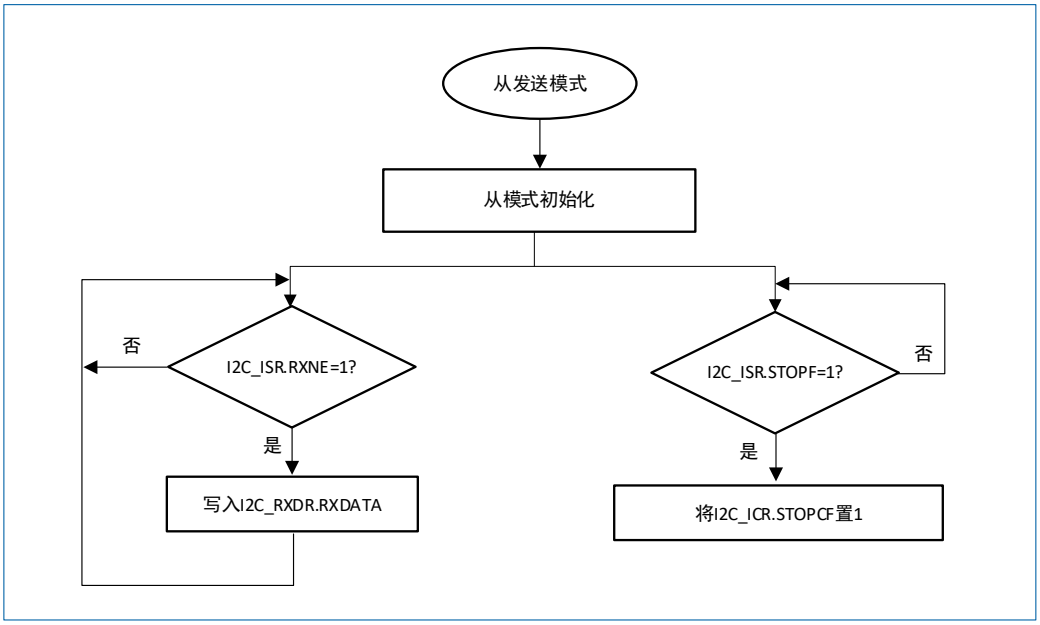

图 21-12 从接收器的传输序列流程图（NOSTRETCH=1）

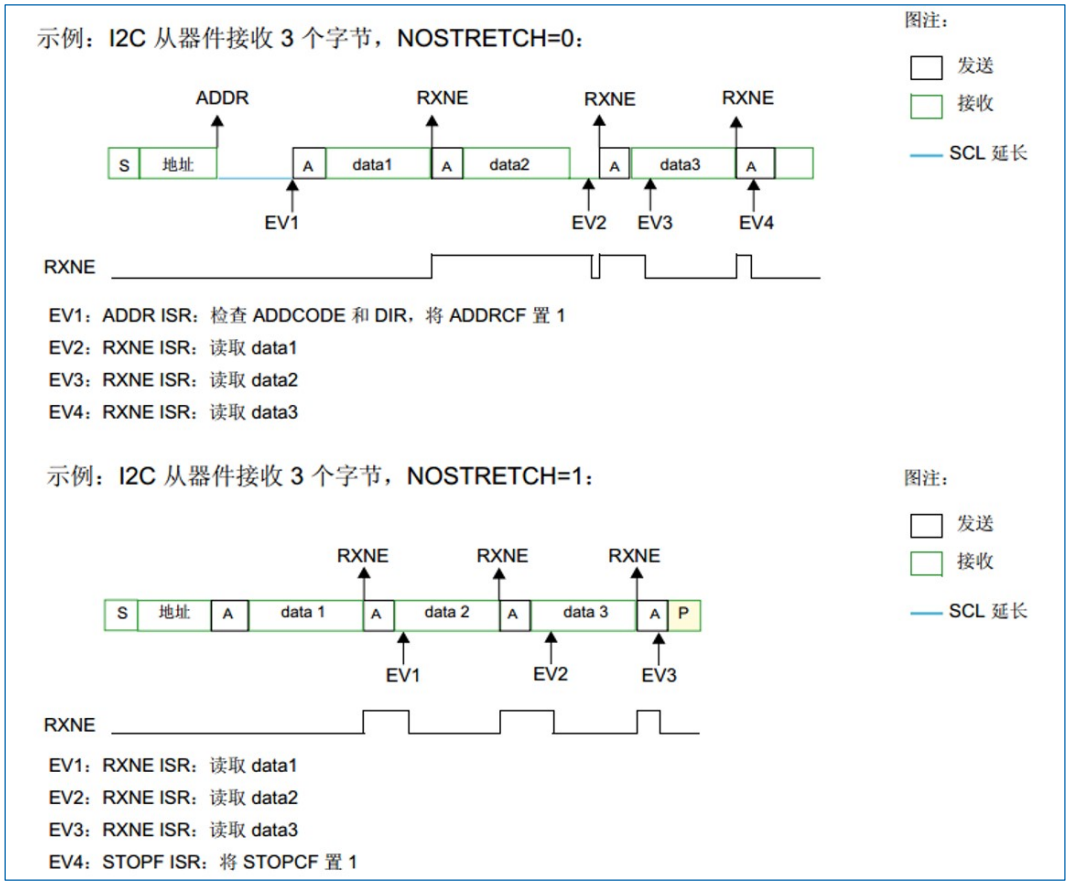

图 21-13 I2C 从接收器的传输总线图

21.2.8 主模式

I2C 主模式初始化

使能外设前，必须通过设置 I2C1_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 I2C1CLK 时钟的同步。一旦 SCLL 计数器达到 I2C1_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 I2C1CLK 时钟的同步。一旦 SCLH 计数器达到 I2C1_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：

$$t_{SCL}=T_{SYNC1}+T_{SYNC2}+\{[（SCLH+1）+（SCLL+1）]x（PRESC+1）xt_{I2CCLK}\}$$

- 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）

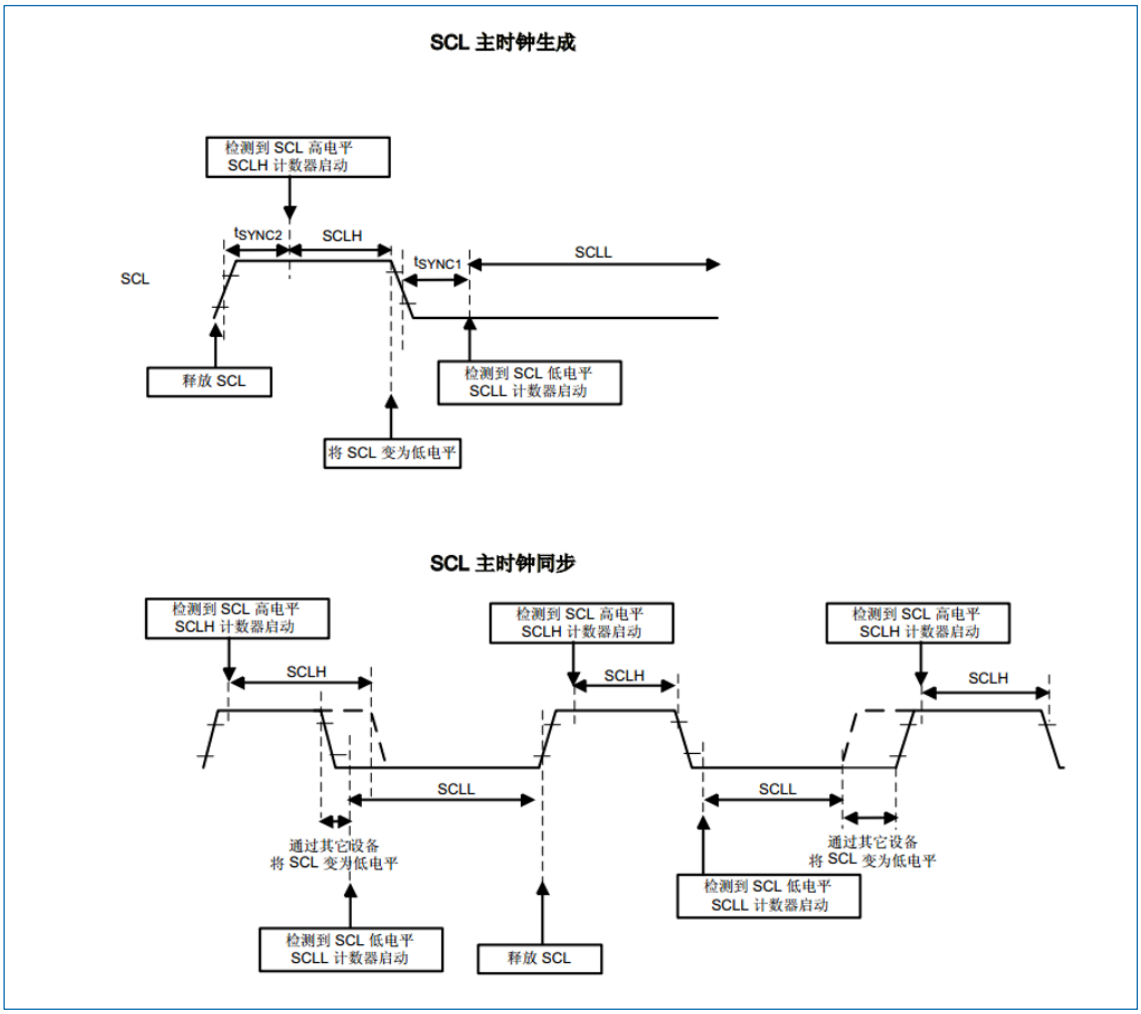

图 21-14 主时钟生成

注意：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 21-4 I2C-SMBus 规范时钟时序

符号	参数	标准模式(Sm)		快速模式(Fm)		超快速模式(Fm+)		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值	
f_{SCL}	SCL 时钟频率	-	100	-	400	-	1000	-	100	kHz
$t_{HD:STA}$	（重复）起始条件的保持时间	4.0	-	0.6		0.26	-	4.0	-	μs
$t_{SU:STA}$	重复起始条件的建立时间	4.7	-	0.6		0.26	-	4.7	-	μs
$t_{SU:STO}$	停止条件的建立时间	4.0	-	0.6		0.26	-	4.0	-	μs
t_{BUF}	停止条件和起始条件之间的总线空闲时间	4.7	-	1.3		0.5	-	4.7	-	μs
t_{LOW}	SCL 时钟的低电平周期	4.7	-	1.3		0.5	-	4.7	-	μs
t_{HIGH}	SCL 时钟的高电平周期	4.0	-	0.6		0.26	-	4.0	50	μs

符号	参数	标准模式(Sm)		快速模式(Fm)		超快速模式(Fm+)		SMBus		单位
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值	
t_r	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	120	-	1000	μs
t_f	SDA 和 SCL 信号的下降时间	-	300	-	300	-	120	-	300	μs

注意：

- *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 I2C1_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 I2C1_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。

注意：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。

注意：该步骤同样适用于重复起始位。在这种情况下，BUSY=1。

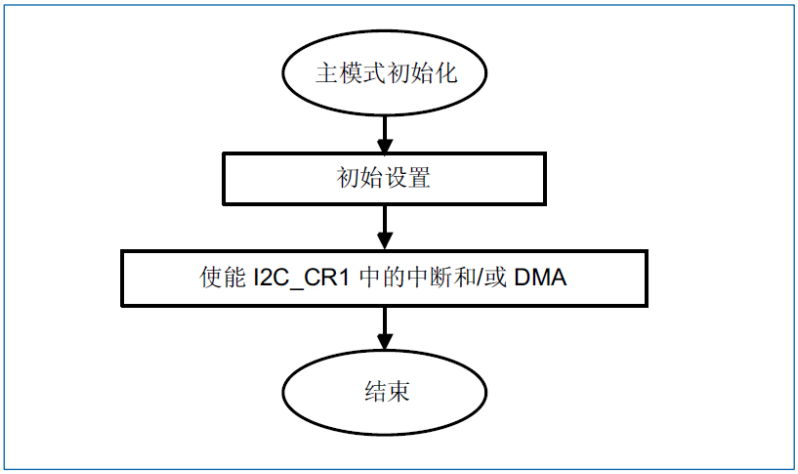

图 21-15 初始化流程图

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 I2C1_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

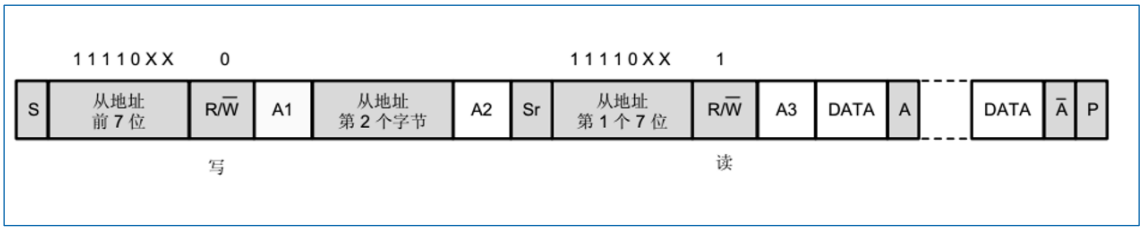

图 21-16 10 位地址读访问（HEAD10R=0）

- 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。

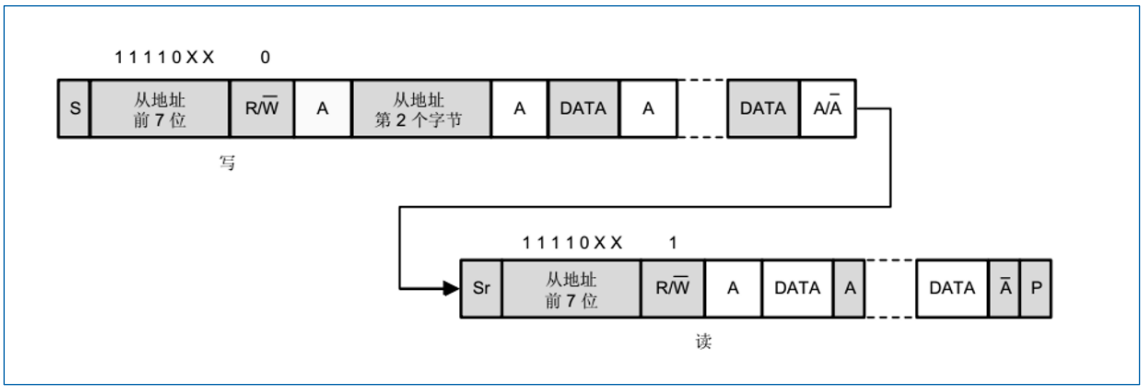

图 21-17 10 位地址读访问（HEAD10R=1）

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 I2C1_CR1 寄存器中的 TXIE 位置 1，TXIS 事件将生成中断。当 I2C1_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 I2C1_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 I2C1_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 I2C1_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。I2C1_ISR 寄存器中的 NACKF 标志置 1，如果 NACKIE 位置 1，还将生成中断。

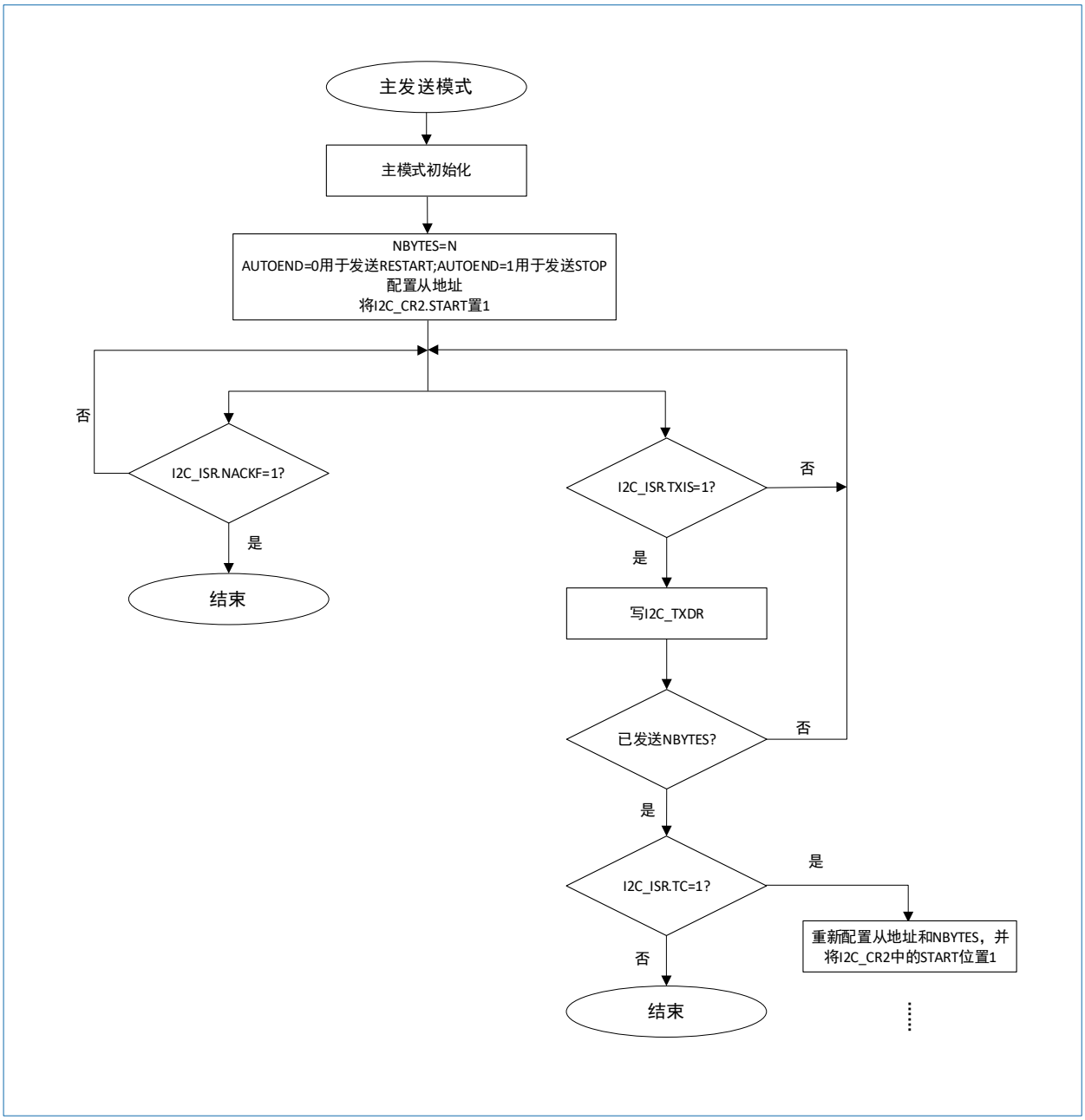

图 21-18 I2C 主发送器的传输序列流程图（N≤255 字节）

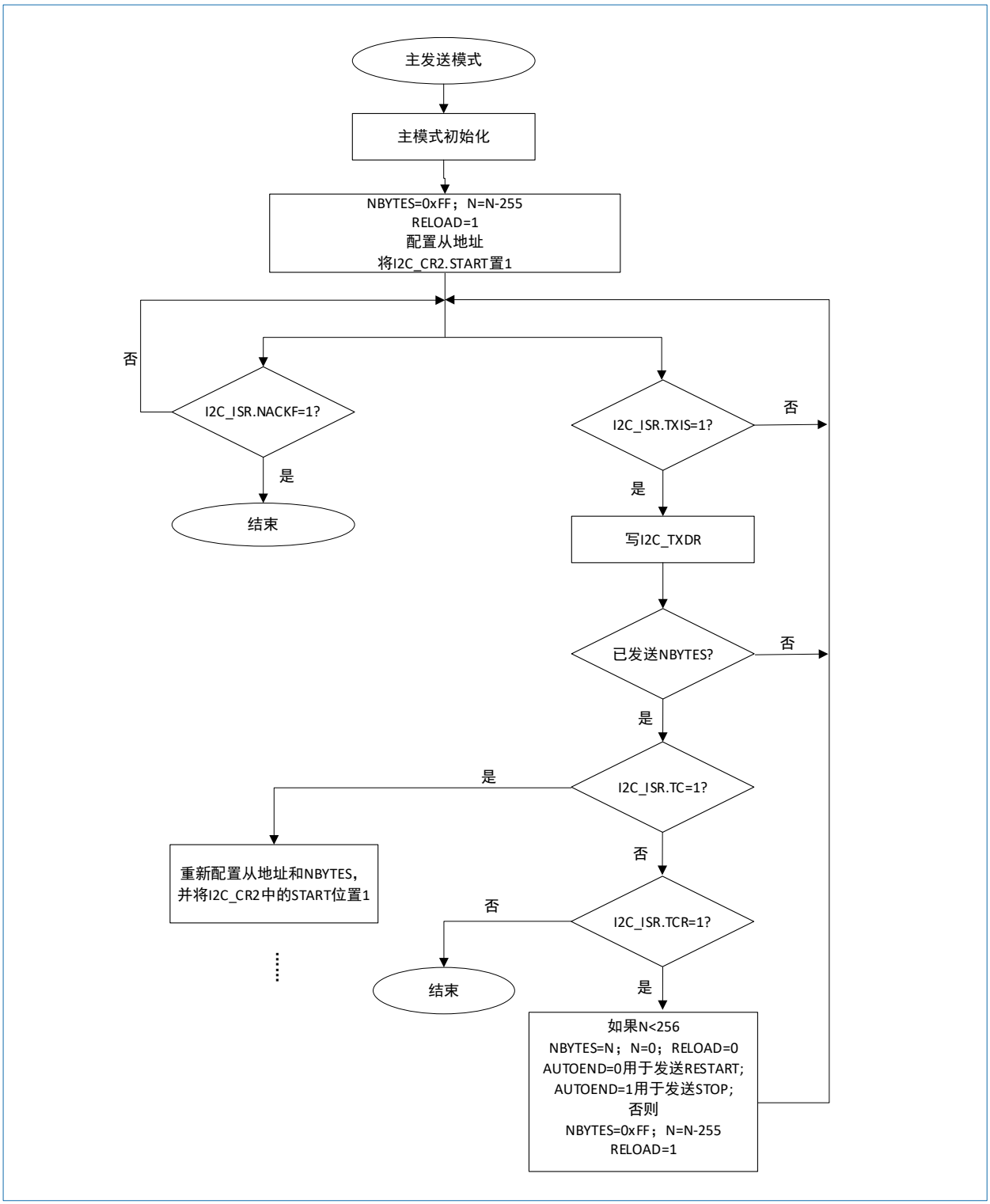

图 21-19 I2C 主发送器的传输序列流程图（N>255 字节）

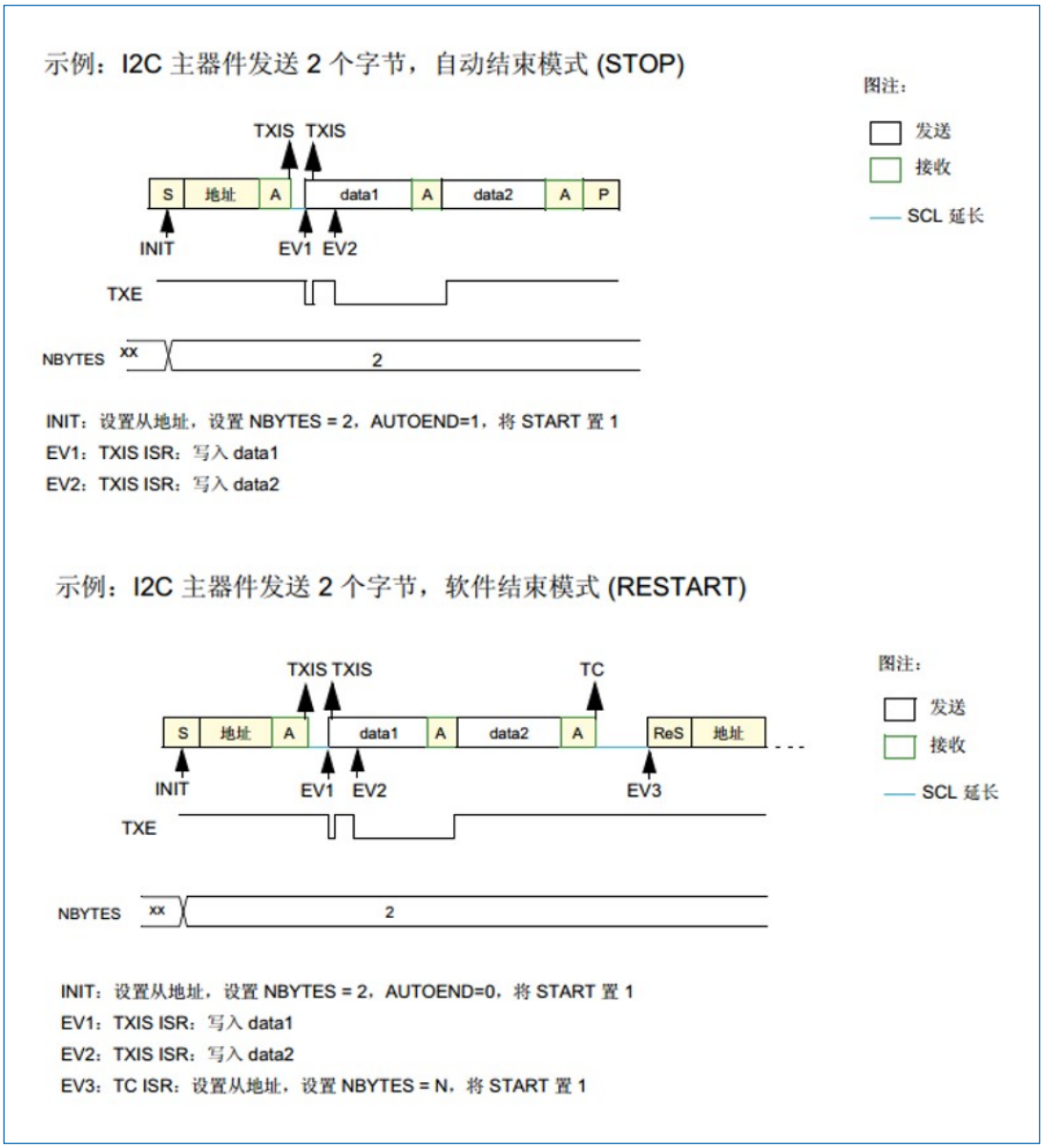

图 21-20 I2C 主发送器的传输总线图

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 I2C1_CR1 寄存器中的 RXIE 位置 1，RXNE 事件将生成中断。读取 I2C1_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 I2C1_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 I2C1_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 I2C1_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。

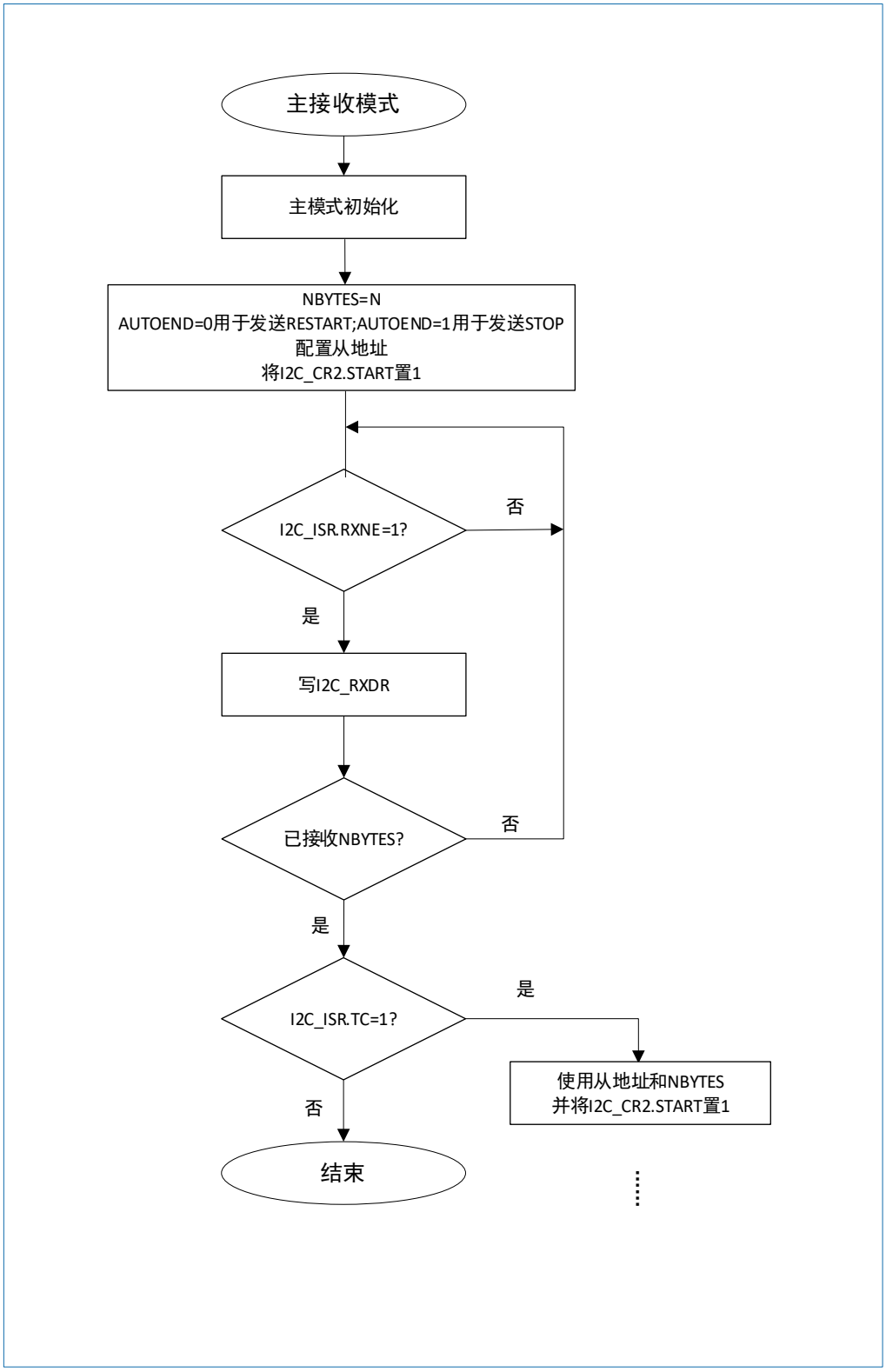

图 21-21 I2C 主接收器的传输序列流程图（N≤255 字节）

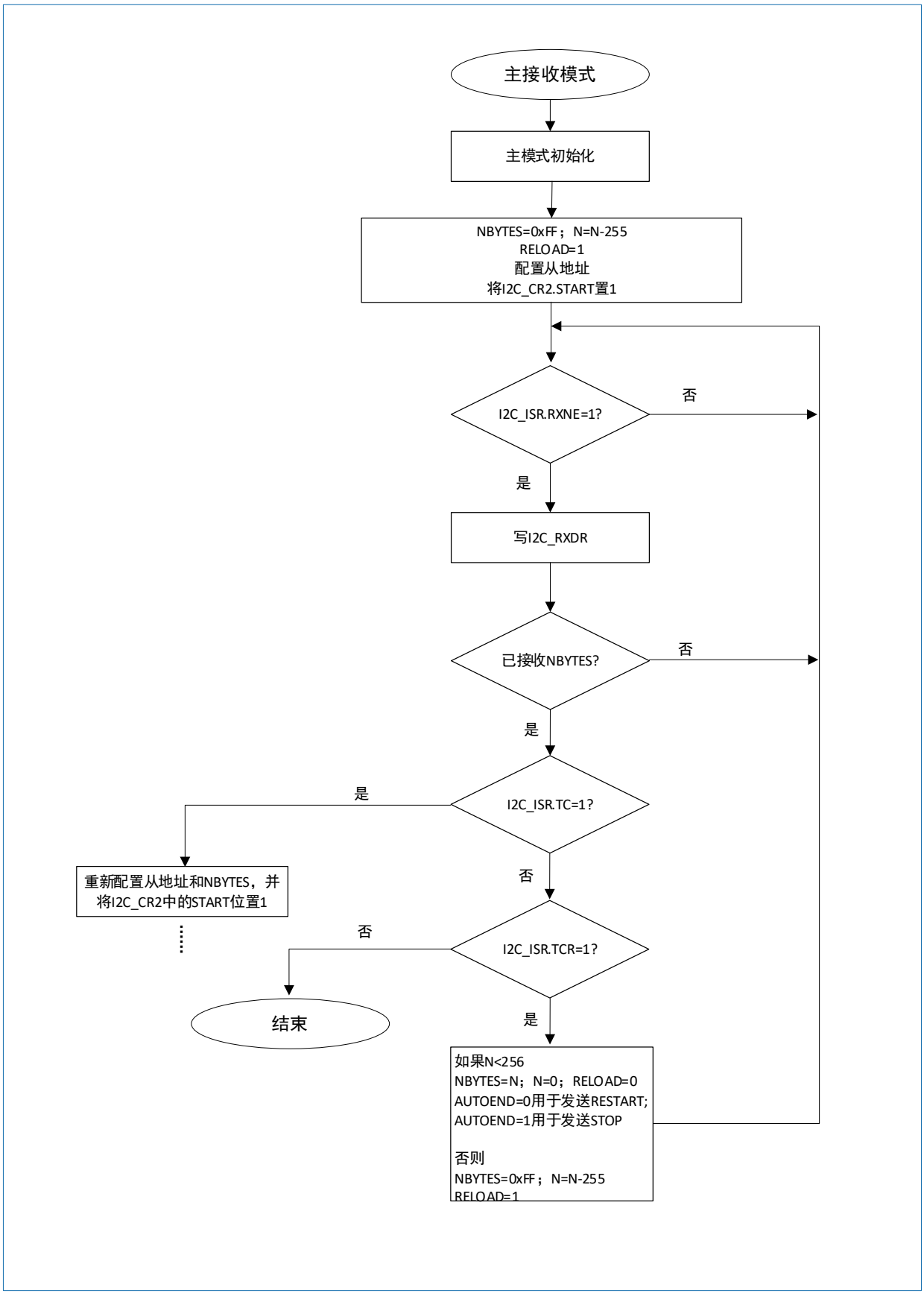

图 21-22 I2C 主接收器的传输序列流程图（N>255 字节）

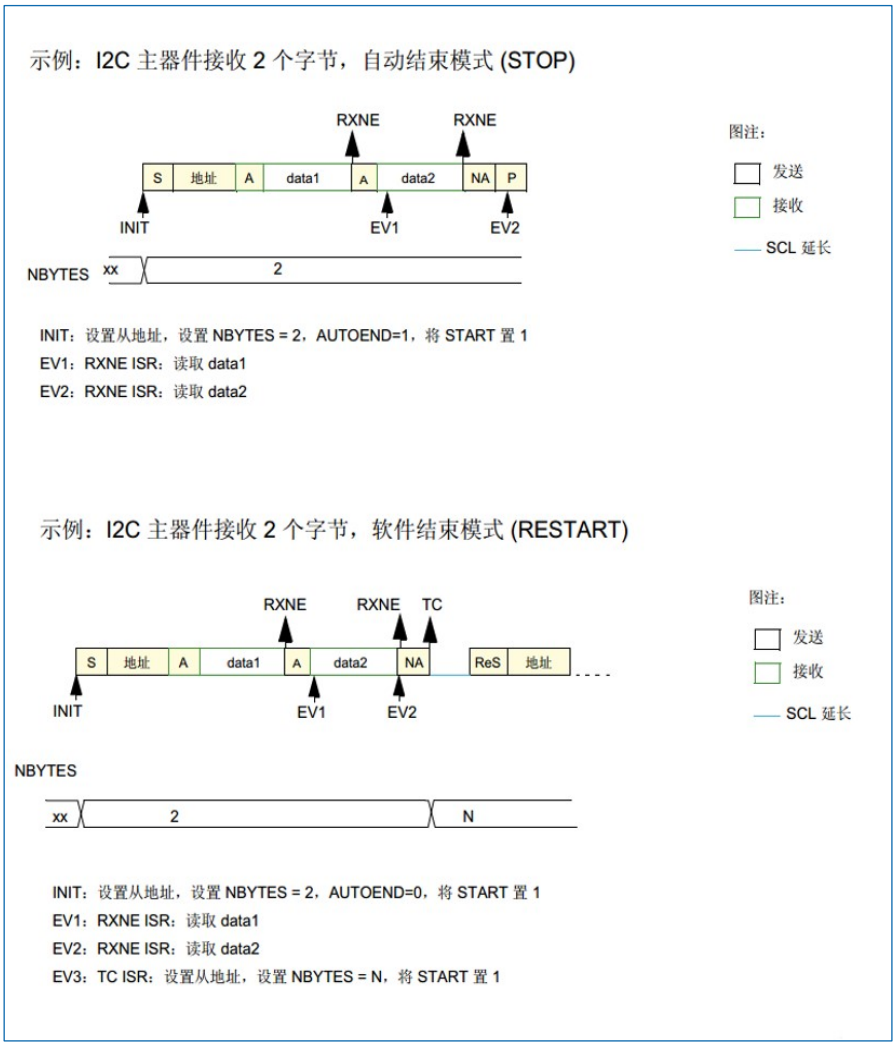

图 21-23 I2C 主接收器的传输总线图

21.2.9 I2C1_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 I2C1_TIMINGR 才能获得符合 I2C 规范的时序。

表 21-5 f_{I2CCLK} =8MHz 时的时序设置示例

参数	标准模式（Sm）		快速模式（Fm）	超快速模式（Fm+）
	10 kHz	100 kHz	400 kHz	500 kHz
PRESC	1	1	0	0
SCLL	0xC7	0x13	0x9	0x6
t_{SCLL}	200x250 ns=50 µs	20x250 ns=5.0 µs	10x125 ns=1250 ns	7x125 ns=875 ns
SCLH	0xC3	0xF	0x3	0x3
t_{SCLH}	196x250 ns=49 µs	16x250 ns=4.0 µs	4x125 ns=500 ns	4x125 ns=500 ns
t_{SCL} (1)	约 100 µs (2)	约 10 µs (2)	约 2500 ns (3)	约 2000 ns (4)
SDADEL	0x2	0x2	0x1	0x0
t_{SDADEL}	2x250 ns=500 ns	2x250 ns=500 ns	1x125ns=125ns	0 ns

参数	标准模式（Sm）		快速模式（Fm）	超快速模式（Fm+）
	10 kHz	100 kHz	400 kHz	500 kHz
SCLDEL	0x4	0x4	0x3	0x1
t_{SCLDEL}	5x250ns=1250ns	5x250ns=1250ns	4x125ns=500ns	2x125ns=250ns

(1). 由于 SCL 内部检测存在延时，SCL 周期 t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
(2). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=1000 ns 时的示例。
(3). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=750 ns 时的示例。
(4). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=655 ns 时的示例。

表 21-6 f_{I2CCLK}=16MHz 时的时序设置示例

参数	标准模式（Sm）		快速模式（Fm）	超快速模式（Fm+）
	10 kHz	100 kHz	400 kHz	1 MHz
PRESC	3	3	1	0
SCLL	0xC7	0x13	0x9	0x4
t_{SCLL}	200x250 ns=50 µs	20x250 ns=5.0 µs	10x125 ns=1250 ns	5x62.5ns=312.5 ns
SCLH	0xC3	0xF	0x3	0x2
t_{SCLH}	196x250 ns=49 µs	16x250 ns=4.0 µs	4x125 ns=500 ns	3x62.5ns=187.5 ns
t_{SCL} (1)	约 100 µs (2)	约 10 µs (2)	约 2500 ns (3)	约 1000 ns (4)
SDADEL	0x2	0x2	0x2	0x0
t_{SDADEL}	2x250 ns=500 ns	2x250 ns=500 ns	2x125 ns=250 ns	0ns
SCLDEL	0x4	0x4	0x3	0x2
t_{SCLDEL}	5x250 ns=1250 ns	5x250 ns=1250 ns	4x125 ns=500 ns	3x62.5 ns=187.5 ns

(1). 由于 SCL 内部检测存在延时，SCL 周期 t_{SCL} 大于 t_{SCLL} + t_{SCLH}。为 t_{SCL} 提供的值仅用于举例说明。
(2). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=1000 ns 时的示例。
(3). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=750 ns 时的示例。
(4). t_{SYNC1}+t_{SYNC2} 最小值为 4 x t_{I2CCLK} =500 ns。t_{SYNC1}+t_{SYNC2}=655 ns 时的示例。

21.2.10 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 I2C1_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 I2C1_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 I2C1_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（0b0001000）。

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（0b0001100）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 I2C1_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，I2C1_ISR 寄存器中的 ALERT 标志置 1。如果 I2C1_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 21-7 SMBus 超时规范

符号	参数	限值		单位
		最小值	最大值	
$t_{TIMEOUT}$	检测时钟低电平超时	25	35	ms

符号	参数	限值		单位
		最小值	最大值	
$t_{LOW:SEXT}$（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。其它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。从器件或其它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。

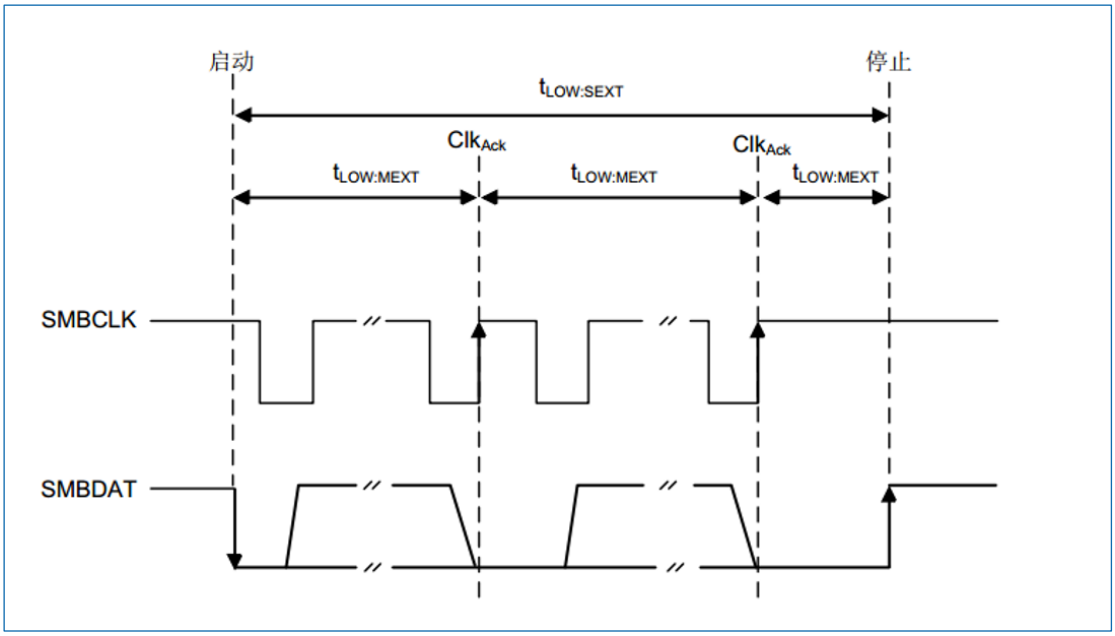

图 21-24 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔

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 THIGH、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。

21.2.11 SMBus 初始化

SMBus 的初始化包括 I2C 初始化之外，还必须进行一些其它的特定初始化，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 I2C1_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 I2C1_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 I2C1_CR1 寄存器中的 SMBHEN 位置 1 来使能 SMBus 主机地址（0b000 1000）。
- 通过将 I2C1_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 I2C1_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（I2C1_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。

注意：使能 I2C 时，不允许更改 PECEN 配置。

表 21-8 带 PEC 的 SMBus 配置

模式	SBC 位	RELOAD 位	AUTOEND 位	PECBYTE 位
主 Tx/Rx+NBYTES+PEC+STOP	x	0	1	1
主 Tx/Rx+NBYTES+PEC+RESTART	x	0	0	1
从 Tx/Rx+PEC	1	0	x	1

超时检测

将 I2C1_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 I2C1_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}，I2C1_ISR 寄存器中的 TIMEOUT 标志将置 1。

注意：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 I2C1_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“21.2.10 SMBus I2C 特性”中的“总线空闲检测”一节给出的超时间隔，则 I2C1_ISR 寄存器中的 TIMEOUT 标志将置 1。请参见表 21-8。

注意：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 I2C1_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}，I2C1_ISR 寄存器中的 TIMEOUT 标志将置 1。请参见表 21-9。

注意：TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

21.2.12 SMBus：I2C1_TIMEOUTR 寄存器配置示例

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms

表 21-9 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 21-10 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 21-11 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE}=50µs）

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	t_{TIDLE}
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

21.2.13 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 减 1，如果主器件在完成 NBYTES 减 1 字节的数据传输后请求传输额外的字节，则将自动发送 I2C1_PECR 寄存器的内容。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

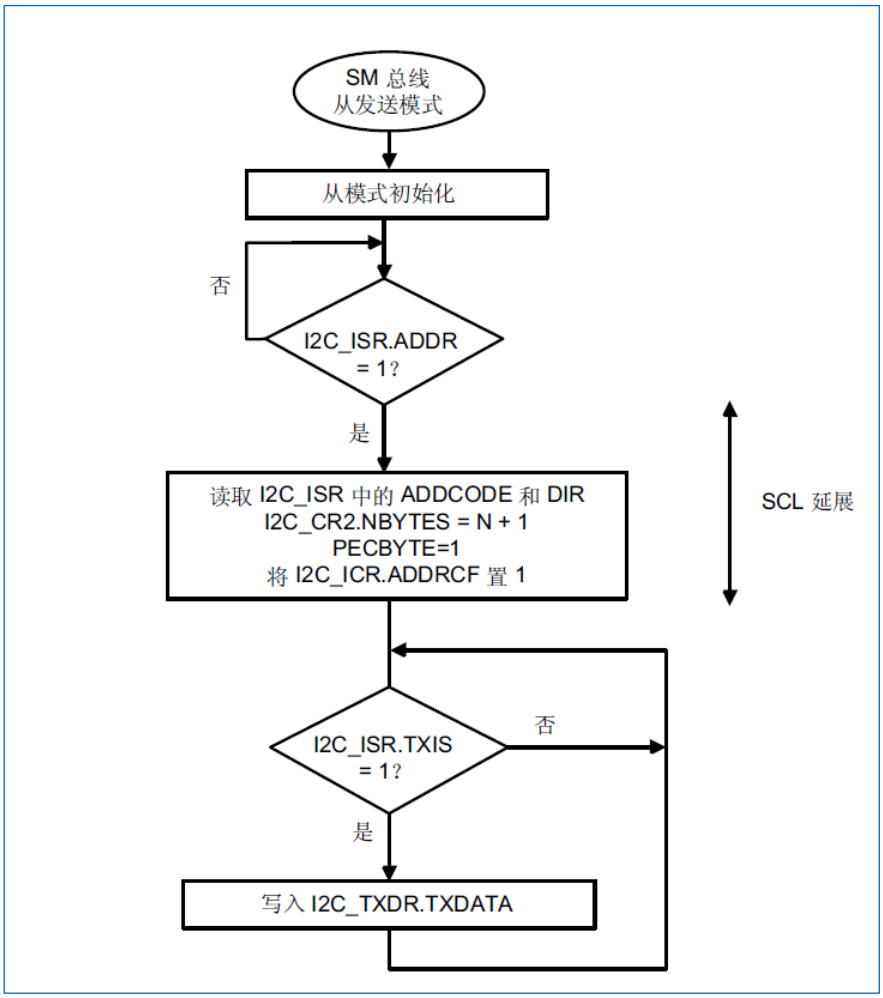

图 21-25 SMBus 从发送器的传输序列流程图（N 字节+PEC）

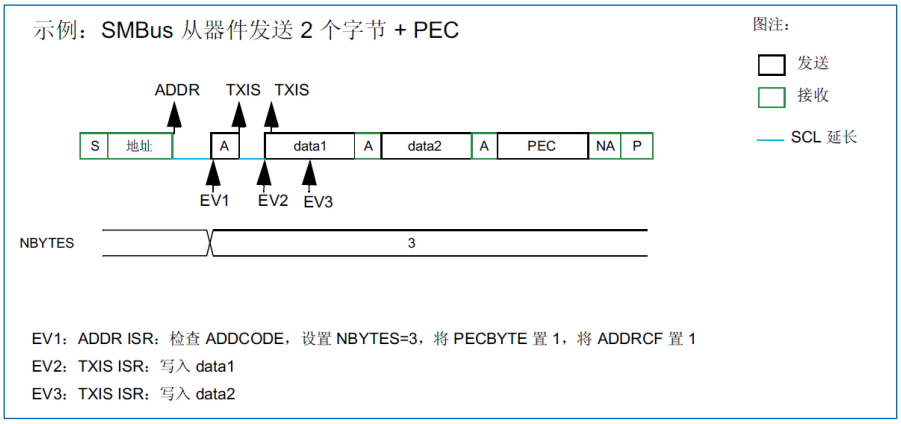

图 21-26 SMBus 从发送器的传输总线图（SBC=1）

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“21.2.7 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 I2C1_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 NACK 位的值无关。PEC 字节一经接收，便会像任何其它数据一样复制到 I2C1_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 I2C1_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。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

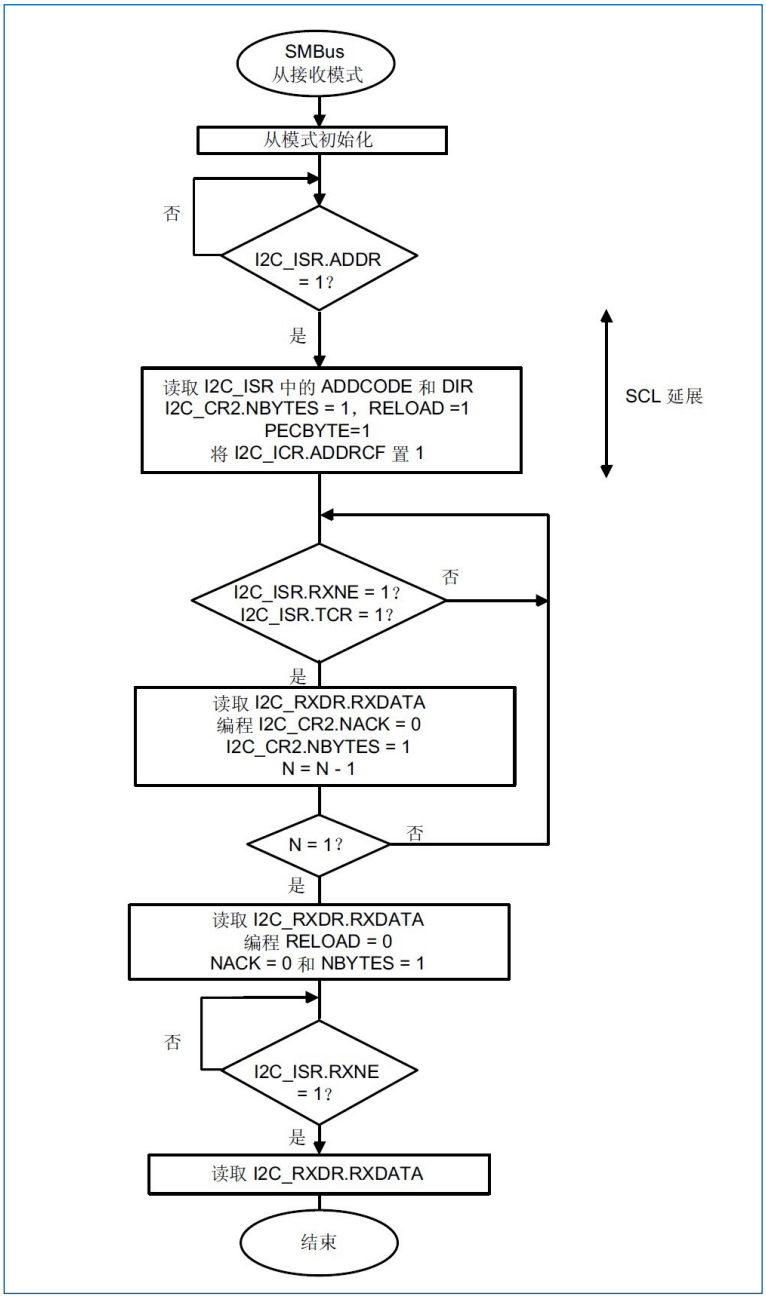

图 21-27 SMBus 从接收器的传输序列流程图（N 字节+PEC）

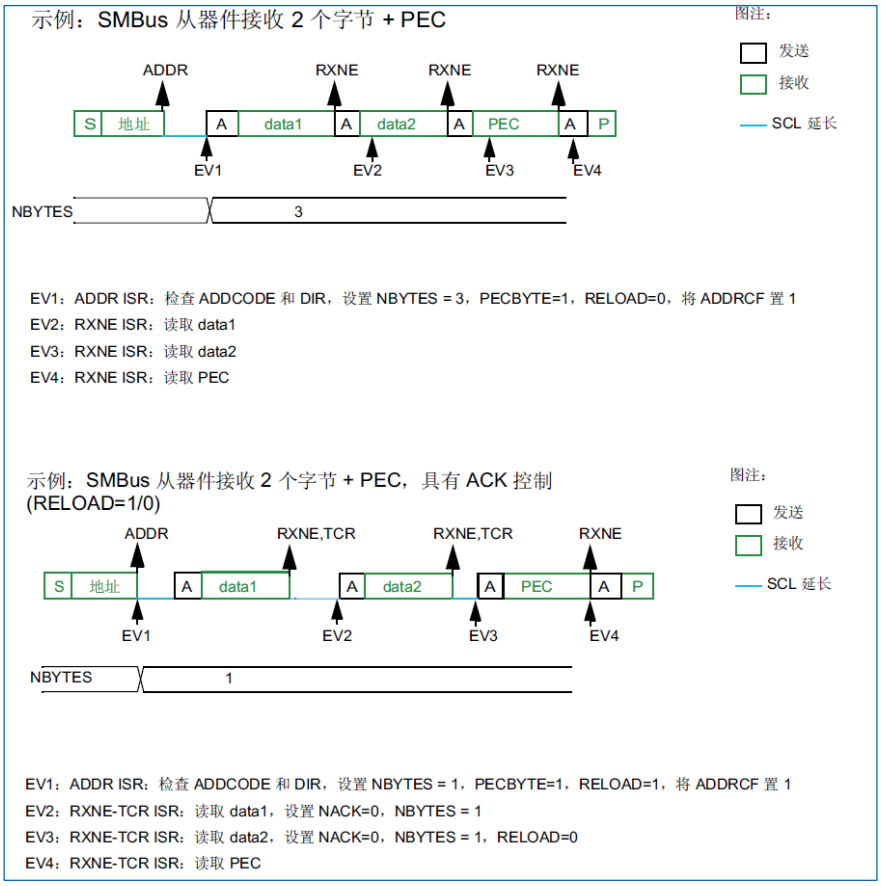

图 21-28 从接收器的总线传输图（SBC=1）

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 减 1。因此，如果 PECBYTE 位在 NBYTES=0x1 时置 1，则将自动发送 I2C1_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 减 1 字节的数据后，将发送 I2C1_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。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

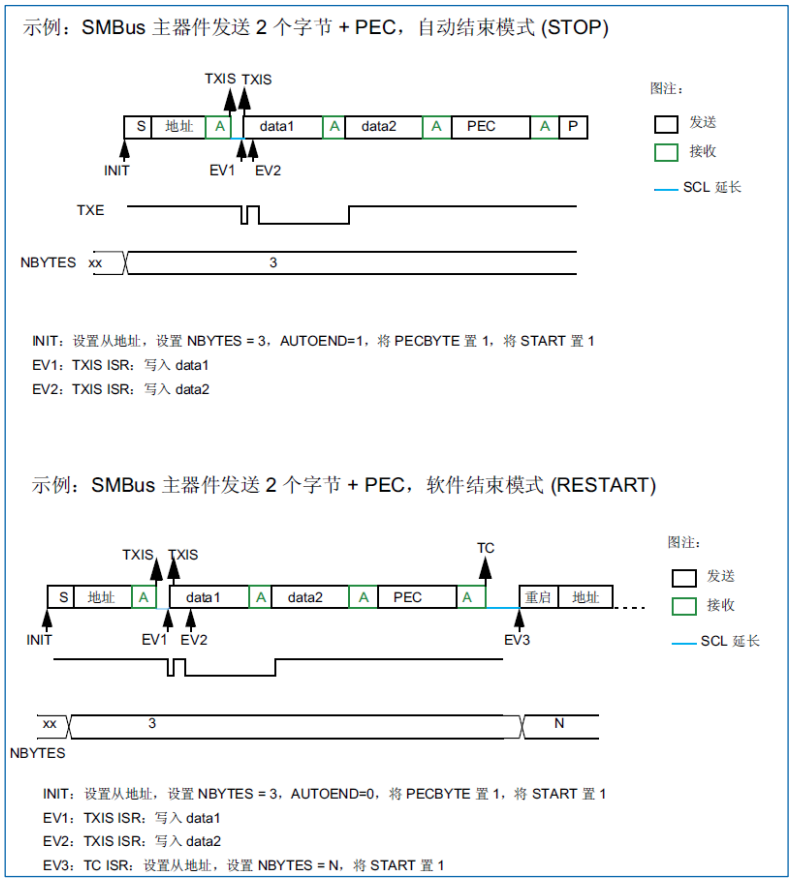

图 21-29 SMBus 主发送器的总线传输图

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 减 1 字节的数据后，将自动使用 I2C1_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 I2C1_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。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

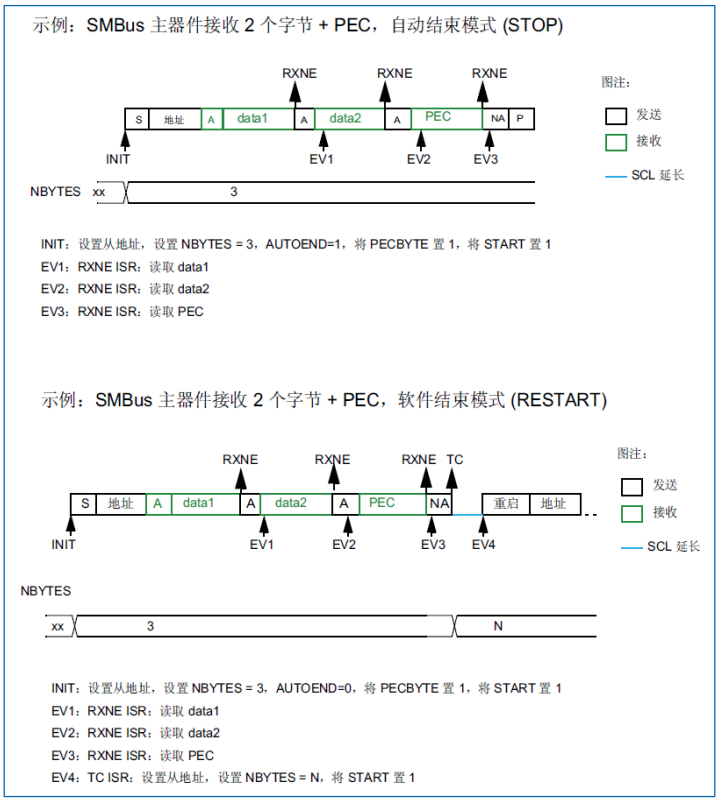

图 21-30 SMBus 主接收器的总线传输图

21.2.14 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。
- 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。
- 只有当 I2C 时钟源为 HSI 振荡器时，该功能才可用。
- 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 I2C1_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。

21.2.15 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，I2C1_ISR 寄存器中的 BERR 标志将置 1，如果 I2C1_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，I2C1_ISR 寄存器中的 ARLO 标志将置 1，如果 I2C1_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 I2C1_TXDR 寄存器的内容，否则发送 0xFF。
  - 应发送一个新字节但尚未向 I2C1_TXDR 寄存器写入数据时，将发送 0xFF。
  - 检测到上溢或下溢错误时，I2C1_ISR 寄存器中的 OVR 标志将置 1，如果 I2C1_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 I2C1_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，I2C1_ISR 寄存器中的 PECERR 标志将置 1，如果 I2C1_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，I2C1_ISR 寄存器中的 TIMEOUT 标志将置 1，如果 I2C1_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 I2C1_CR1 寄存器中的 ERRIE 位置 1，将生成中断。

21.2.16 DMA 请求

使用 DMA 进行发送

将 I2C1_CR1 寄存器中的 TXDMAEN 位置 1 可以使能 DMA（直接存储器访问）进行发送。当 TXIS 位置 1 时，数据将从由 DMA 外设配置的 SRAM 区（请参见“8 直接存储器访问控制器（DMA）”）装载进 I2C1_TXDR 寄存器。

只有数据字节采用 DMA 进行传输。

- 在主模式下：

初始化、从地址、方向、字节数和起始位均由软件编程（发送的从地址无法通过 DMA 传输）。当所有数据均通过 DMA 传输时，必须在起始位置 1 之前初始化 DMA。传输结束由 NBYTES 计数器来管理。请参见“21.2.8 主模式”中的“主发送器”。

- 在从模式下：
  - 当 NOSTRETCH=0 时，如果所有数据均通过 DMA 传输，则必须在地址匹配事件之前（或清零 ADDR 之前在 ADDR 中断子程序中）初始化 DMA。
  - 当 NOSTRETCH=1 时，必须在地址匹配事件之前初始化 DMA。

支持 SMBus 时：PEC 传输由 NBYTES 计数器管理。请参见章节“21.2.13 SMBus 模式”中的“SMBus 从发送器”和“SMBus 主发送器”。

注意：如果使用 DMA 进行发送，则无需使能 TXIE 位。

使用 DMA 进行接收

将 I2C1_CR1 寄存器中的 RXDMAEN 位置 1 可以使能 DMA（直接存储器访问）进行接收。

当 RXNE 位置 1 时，数据将从 I2C1_RXDR 寄存器装载进由 DMA 外设配置的 SRAM 区（请参见“8 直接存储器访问控制器（DMA）”）。只有数据字节（包括 PEC）采用 DMA 进行传输。

- 在主模式下：初始化、从地址、方向、字节数和起始位均由软件编程。当所有数据均通过 DMA 传输时，必须在起始位置 1 之前初始化 DMA。传输结束由 NBYTES 计数器来管理。
- 在从模式下，当 NOSTRETCH=0 时，如果所有数据均通过 DMA 传输，则必须在地址匹配事件之前（或清零 ADDR 标志之前在 ADDR 中断子程序中）初始化 DMA。

如果支持 SMBus：PEC 传输由 NBYTES 计数器管理。请参见章节“21.2.13 SMBus 模式”中的“SMBus 从接收器”和“SMBus 主接收器”。

注意：如果使用 DMA 进行接收，则无需使能 RXIE 位。

21.2.17 调试模式

当微控制器进入调试模式时（内核停止），SMBus 超时定时器会根据 DBG 模块中的 DBG_I2C1_SMBus_TIMEOUT 配置位选择继续正常工作还是停止工作。

21.3 I2C 低功耗模式

表 21-12 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。

21.4 I2C 中断

下表给出了 I2C 中断请求列表。

表 21-13 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 I2C1_RXDR 寄存器	RXIE
发送缓冲区中断状态	TXIS	写入 I2C1_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 I2C1_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
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），也可以分配到两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，请参见“9.3 EXTI 寄存器”。

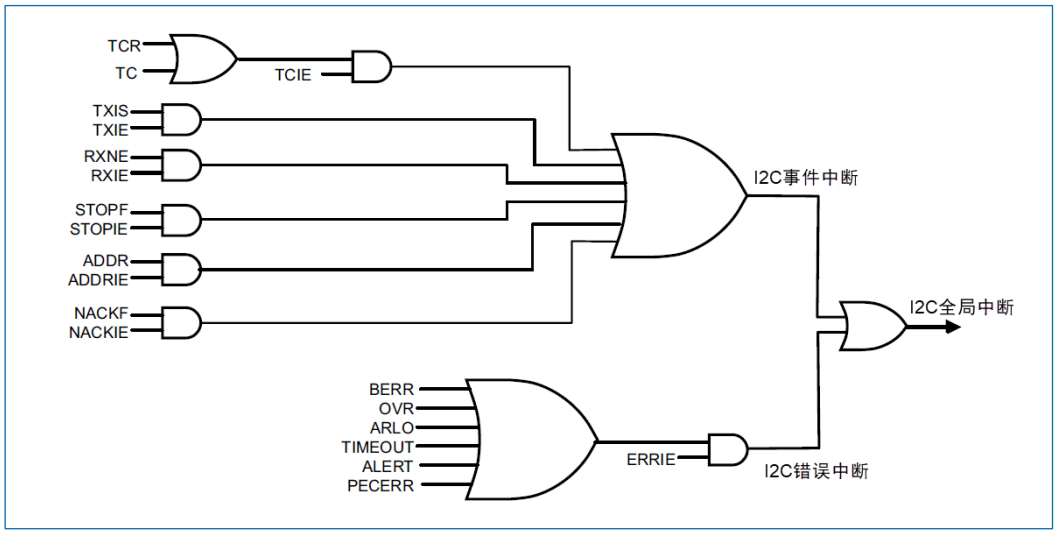

图 21-31 I2C 中断映射图

21.5 I2C 寄存器

基地址：0x4001 4C00

空间大小：0x400

21.5.1 控制寄存器 1（I2C1_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
SWAP	Res							PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	WUPEN	NOSTRETCH	SBC
rw								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDMAEN	TXDMAEN	Res	ANFOFF	DNF[3:0]				ERRIE	TCIE	STOPIE	NACKIE	ADDRIE	RXIE	TXIE	PE
rw	rw		rw	rw				rw	rw	rw	rw	rw	rw	rw	rw

位 31	SWAP：交换 SDA/SCL 引脚（Exchange SDA/SCL pin） 该位由软件置 1 或清 0。 ● 0：SDA/SCL 引脚为定义的标准输出引脚。 ● 1：SDA/SCL 引脚功能交换。
位 30:24	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。

	● 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	RXDMAEN：DMA 接收请求使能（DMA reception requests enable） ● 0：关闭 DMA 接收请求 ● 1：开启 DMA 接收请求
位 14	TXDMAEN：DMA 发送请求使能（DMA transmission requests enable） ● 0：关闭 DMA 发送功能 ● 1：开启 DMA 发送功能
位 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_{I2C1_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_{I2C1_CLK}。 ● …… ● 1111：数字滤波器启用，其滤波能力达到 15 个 T_{I2C1_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 (slave only)） ● 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

21.5.2 控制寄存器（I2C1_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[9:0]									
rs	rs	rs	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 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 (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 (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 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 (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 I2C1_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 ○ 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。 ○ 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 (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 (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 (master mode)） ● 0：主机请求一个写传输 ● 1：主机请求一个读传输
位 9:0	SADD[9:0]：从机地址位 9:0（主机模式）（Slave address bit 9:0 (master mode)） SADD[9:8]：从机地址位 9:8（Slave address bit 9:8） 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。 SADD[7:1]：从机地址位 7:1（Slave address bit 7:1） ● 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。 SADD[0]：从机地址的 0 位（Slave address bit 0） ● 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。

21.5.3 本机地址 1 寄存器（I2C1_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[9:0]									
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 9:0	OA1[9:0]：接口地址的 9:0 位（Interface address 9:0 bit） OA1[9:8]：接口地址的 9:8 位（Interface address 9:8 bit） ● 7 位地址模式：该值不起作用 ● 10 位地址模式：地址的位 9:8 OA1[7:1]：接口地址的 7:1（Interface address 7:1 bit） 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

21.5.4 本机地址 2 寄存器（I2C1_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。

21.5.5 时序寄存器（I2C1_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 (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 (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}

21.5.6 超时寄存器（I2C1_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 30:29	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 $t_{TIMEOUT}$（t_{IDLE}=0）或保持高的时间超过 t_{IDLE}（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_{I2C1_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_{I2C1_CLK}

21.5.7 中断和状态寄存器（I2C1_ISR）

偏移地址：0x18

复位值：0x0000 0001

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 (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 (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 (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 (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 (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 (receivers)） 该位在当接收到的数据被复制到 I2C1_RXDR 寄存器，准备好被软件读取的时候由硬件置位。在读取 I2C1_RXDR 时 RXNE 会被清除。
位 1	TXIS：发送中断状态（发送）（Transmit interrupt status (transmitters)） 在 I2C1_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 I2C1_TXDR 寄存器。下一个要发送的数据被写到 I2C1_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，如果 TXDMAEN=1 就有 DMA 请求）。
位 0	TXE：发送数据寄存器空（发送）（Transmit data register empty (transmitters)）

	在 I2C1_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 I2C1_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 I2C1_TXDR。

21.5.8 中断清除寄存器（I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_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 I2C1_ISR 寄存器中的 ADDR 标志位。对该位写 1，还会清除 I2C1_CR2 寄存器中的 START 位。
位 2:0	Res：保留

	必须保持复位值。

21.5.9 PEC 寄存器（I2C1_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 时，该位由硬件清零。

21.5.10 接收数据寄存器（I2C1_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。

21.5.11 发送数据寄存器（I2C1_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。

22 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；同时还可以通过使用直接存储器访问（Direct memory access，DMA）实现高速通信。

22.1 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
- 最大可编程收发波特率为：f_{UART} /8
- 双时钟域允许：
  - 方便的波特率编程，可独立于 PCLK 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
- 支持 DMA 传输数据
- 发送器和接收器可单独使能
- 发送和接收的信号极性可单独控制
- 发送（TX）和接收（RX）引脚交换可配置
- 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）

22.2 UART 实现

表 22-1 UART 特性

UART 模式/特性	UART1/UART2
数据字长	7 位、8 位和 9 位
DMA 传输	支持
多处理器通信	支持
单线半双工通信	支持

UART 模式/特性	UART1/UART2
RS232 硬件流控制	不支持
RS485 驱动器使能	不支持

22.3 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。

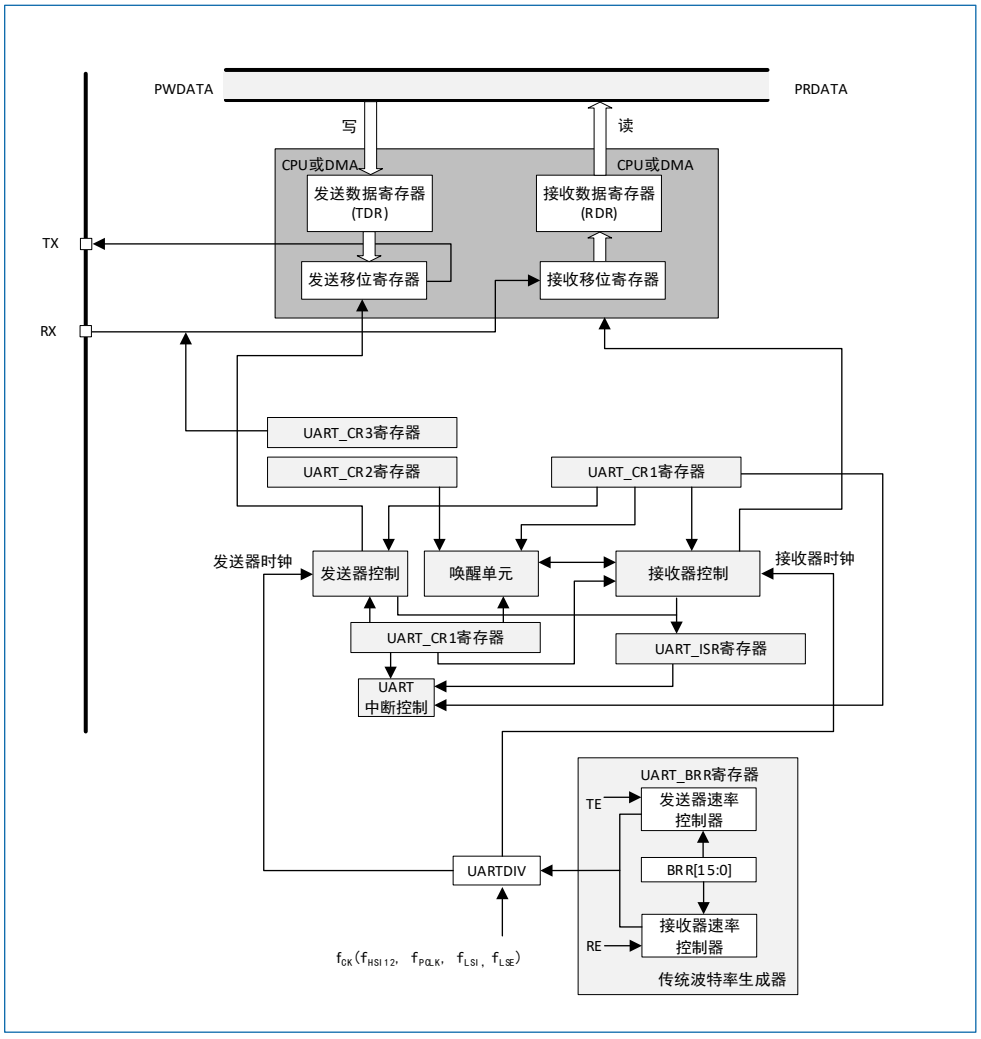

图 22-1 UART 框图

在图 22-1 中，关于在 UARTx_BRR 寄存器中编码 UARTDIV 的详细信息，请参见“22.3.4 UART 波特率生成”。f_{CK} 可以是 f_{HSI12}、f_{PCLK}、f_{LSI} 或 f_{LSE}。

22.3.1 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（请参见图 22-2）。

- 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电平值均为“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。

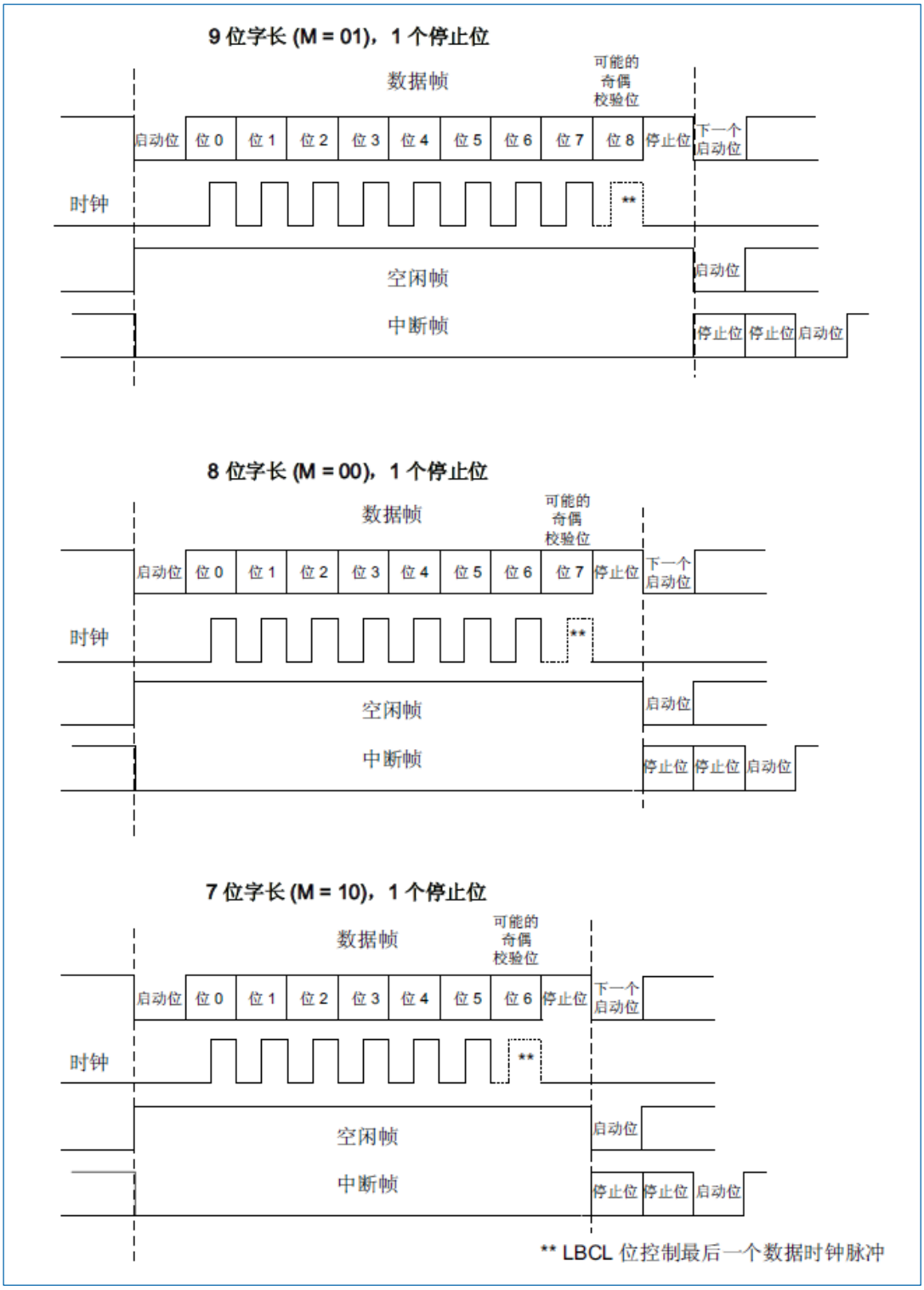

图 22-2 字长编程

22.3.2 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]位来选择停止位的数量。

- 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、单线模式模式支持该值。

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）。

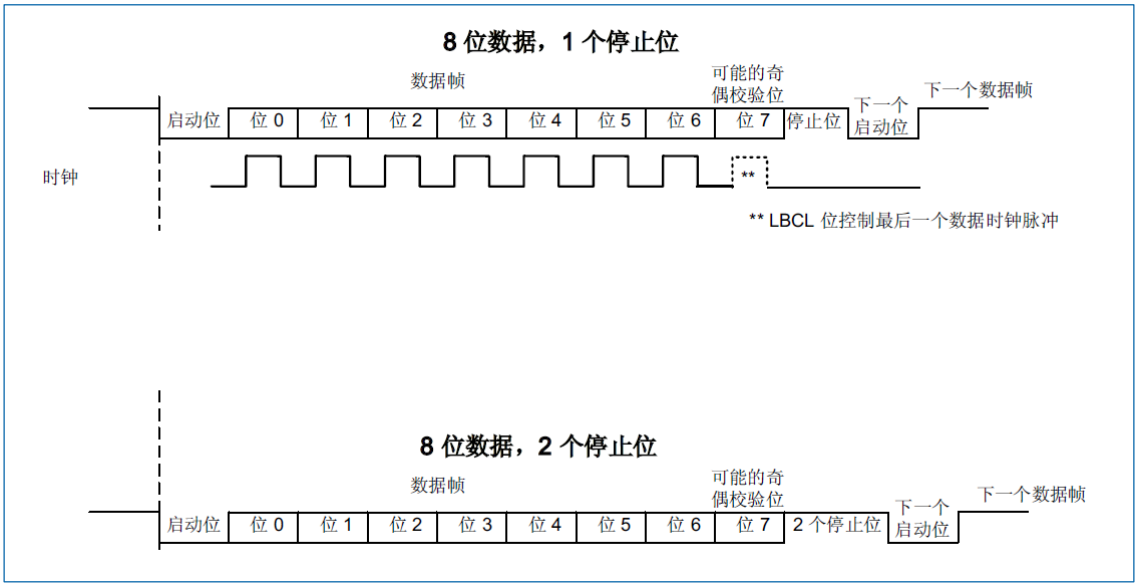

图 22-3 可配置的停止位

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。

通过向发送数据寄存器（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 22-4 所示。

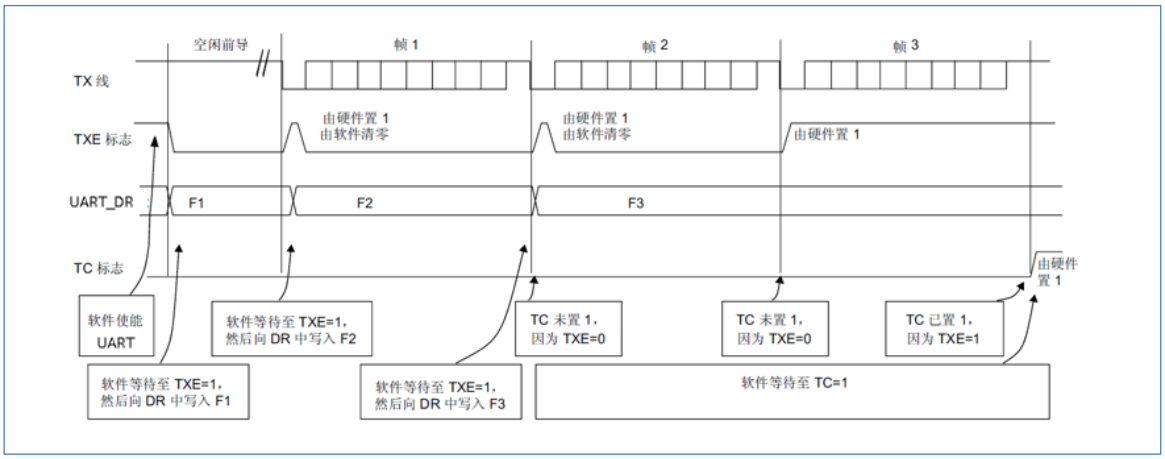

图 22-4 发送时的 TC/TXE 行为

字符发送配置步骤：

1. 配置 UARTx_CR1 中的 M 位来定义字长。
3. 配置 UARTx_BRR 寄存器来选择所需波特率。
4. 配置 UARTx_CR2 中的停止位（STOP[1:0]）个数。
5. 配置 UARTx_CR1 寄存器中的 UE 位来使能 UART 通信。
6. 若需要配合 DMA 使用，则配置 UARTx_CR3 中的 DMAT 位以使能 DMA 使能。按照 DMA 模式下的 UART 通信的要求配置 DMA 寄存器。
7. 配置 UARTx_CR1 中的 TE 位以便在首次发送时发送一个空闲帧。
8. 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。
9. UARTx_TDR 寄存器写入最后一个数据后，等待 TC=1，表明最后一个帧的发送完成，禁止 UART 发送器或进入停机模式时需要此步骤，避免损坏最后一次发送。

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。

22.3.3 UART 接收器

通过配置 UARTx_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）。

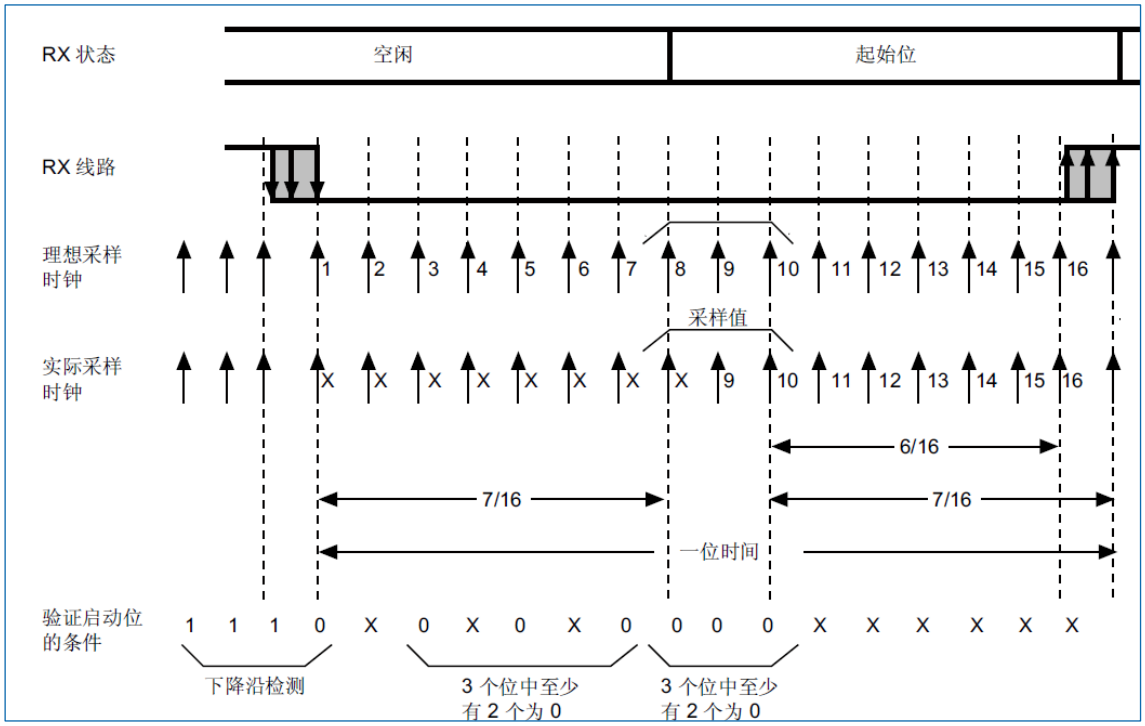

图 22-5 16 倍或 8 倍过采样时的起始位检测

注意：如果上述 NF 标志置位的条件不满足或序列不完整，将中止起始位检测，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_BRR 寄存器来选择所需波特率。
2. 配置 UARTx_CR2 中的停止位（STOP[1:0]）个数。
3. 配置 UARTx_CR1 寄存器中的 UE 位来使能 UART 通信。
4. 若需要配合 DMA 使用，则配置 UARTx_CR3 中的 DMAR 位以使能 DMA。按照 DMA 模式下的 UART 通信要求配置所需的 DMA 寄存器。
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 DMA 模式下，每接收到一个字节后 RXNE 标志位均会被置 1，然后通过 DMA 对接收数据寄存器执行读操作可将 RXNE 标志位清零。
- 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 位为复位，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）、LSE、HSI（12M）或 SYSCLK。

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（请参见图 22-6 和图 22-7）。

- 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“22.3.5 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 $f_{CK}/16$。

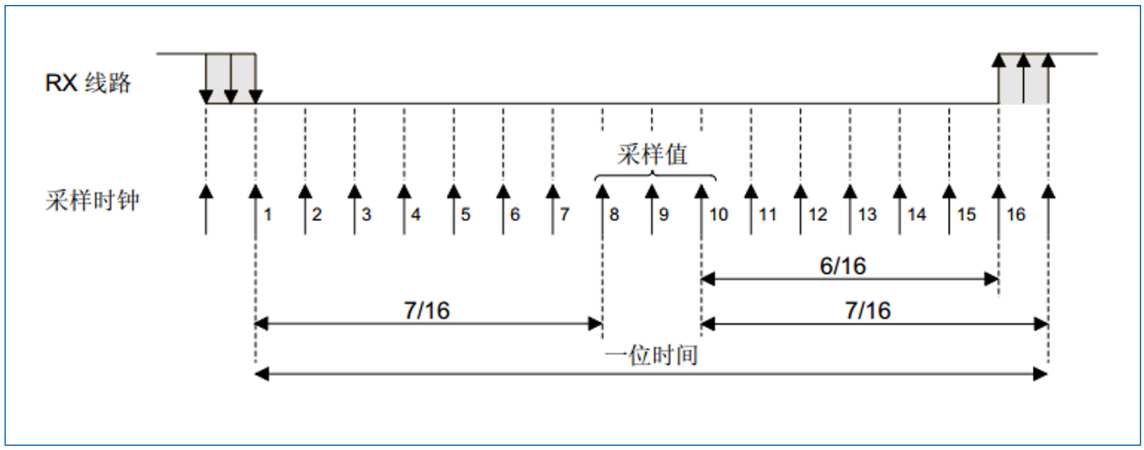

图 22-6 16 倍过采样时的数据采样

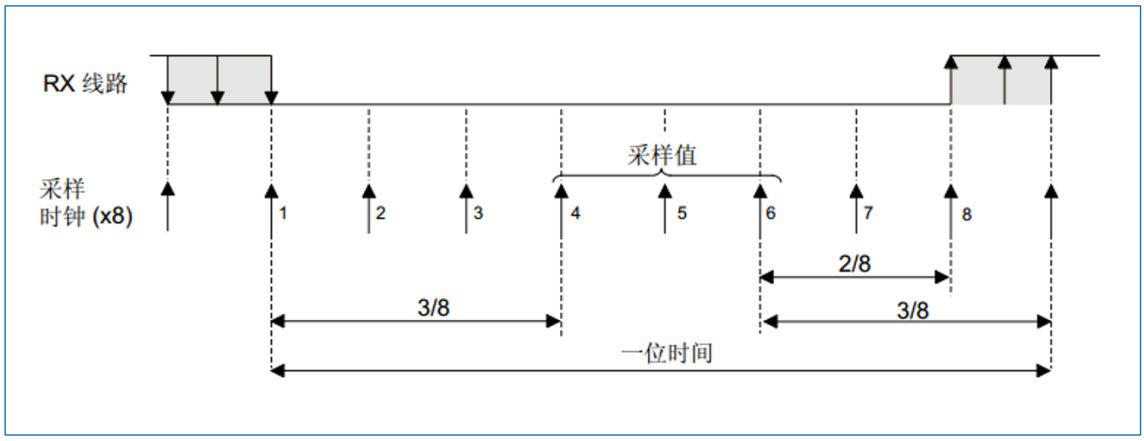

图 22-7 8 倍过采样时的数据采样

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 22-2）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“22.3.5 UART 接收器对时钟偏差的容差”）。

表 22-2 通过采样数据进行噪声检测

采样值	NE 状态（NF）	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。

接收期间可配置的停止位

可通过控制寄存器 2 中的控制位配置要接收的停止位的数量：1 或 2 个。

- 1 个停止位：将在第 8、第 9 和第 10 次采样时对 1 个停止位进行采样。
- 2 个停止位：采样 2 个停止位时，在第 8、第 9 和第 10 次采样时对第一个停止位进行采样。如果在第一个停止位期间检测到帧错误，则帧错误标志位将会置 1。发生帧错误时不检测第 2 个停止位。RXNE 标志将在第一个停止位末尾时置 1。

22.3.4 UART 波特率生成

配置 UARTx_BRR 寄存器的 BRR 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 倍过采样或 8 倍过采样。

16 倍过采样（OVER8=0）时的波特率生成公式如下。

公式 22-1：

$$Tx（Rx）baud=\frac{f_{ck}}{UARTDIV}$$

8 倍过采样（OVER8=1）时的波特率生成公式如下。

公式 22-2：

$$Tx（Rx）baud=\frac{2*f_{ck}}{UARTDIV}$$

其中 UARTDIV 的值由 BRR 位获得。UART 的普通模式支持 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]

注意：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。

通过 UARTx_BRR 寄存器获取 UARTDIV 的示例如下：

示例 1：通过 f_{CK}=8 MHz 获得 9600 波特率

- 当采用 16 倍过采样时，根据公式 22-1
  - UARTDIV = f_{ck}/（Tx/Rx) baud = 8000 000/9600
  - BRR=UARTDIV=833D=0341h
- 当采用 8 倍过采样时，根据公式 22-2：

- UARTDIV= 2 * f_{ck}/（Tx/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}=32MHz 获得 921.6K 波特率

- 当采用 16 倍过采样时，根据公式 22-1
  - UARTDIV= f_{ck}/（Tx/Rx) baud =32000 000/921 600
  - BRR=UARTDIV=35D=23h
- 当采用 8 倍过采样时，根据公式 22-2：
  - UARTDIV=2 * f_{ck}/（Tx/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 22-3 采用 16 倍或 8 倍过采样时，在 f_{ck}=32MHz 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%
1	2.4Kbps	2.4Kbps	0x3415	0	2.4Kbps	0x6825	0
2	9.6Kbps	9.6Kbps	0xD05	0	9.6Kbps	0x1A05	0
3	19.2Kbps	19.19Kbps	0x683	0.02	19.2Kbps	0xD02	0
4	38.4Kbps	38.41Kbps	0x341	0.04	38.39Kbps	0x681	0.02
5	57.6Kbps	57.55Kbps	0x22C	0.08	57.6Kbps	0x453	0
6	115.2Kbps	115.1Kbps	0x116	0.08	115.11Kbps	0x226	0.08
7	230.4Kbps	230.21Kbps	0x8B	0.08	230.21Kbps	0x113	0.08
8	460.8Kbps	463.76Kbps	0x045	0.64	460.06Kbps	0x85	0.08
9	921.6Kbps	914.28Kbps	0x23	0.79	927.5Kbps	0x42	0.79
10	2Mbps	2Mbps	0x10	0	2Mbps	0x20	0
12	4Mbps	4Mbps	-	-	4Mbps	0x10	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。

22.3.5 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 22-4 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 22-4 BRR[3:0]=0000 时的 UART 接收器容差

M 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 22-5 BRR[3:0]!=0000 时的 UART 接收器容差

M 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%

注意：当接收的帧恰好包含 10 个（M 位=00）、11 个（M 位=01）或 9 个（M 位=10）位持续时间的空闲帧时，表 22-4 和表 22-5 中指定的数据可能与特例中的数据略微不同。

22.3.6 使用 UART 进行多处理器通信

在多处理器通信中，以下位需要清零：

- UARTx_CR3 寄存器中的 HDSEL 位

在多处理器通信中，将其中的一个 UART 作为主 UART，它的 TX 输出与其它 UART 的 RX 输入相连接。其它的 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，可以进入静默模式。

在静默模式下：

- 接收状态位为 0。
- 禁止任何接收中断。
- 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:

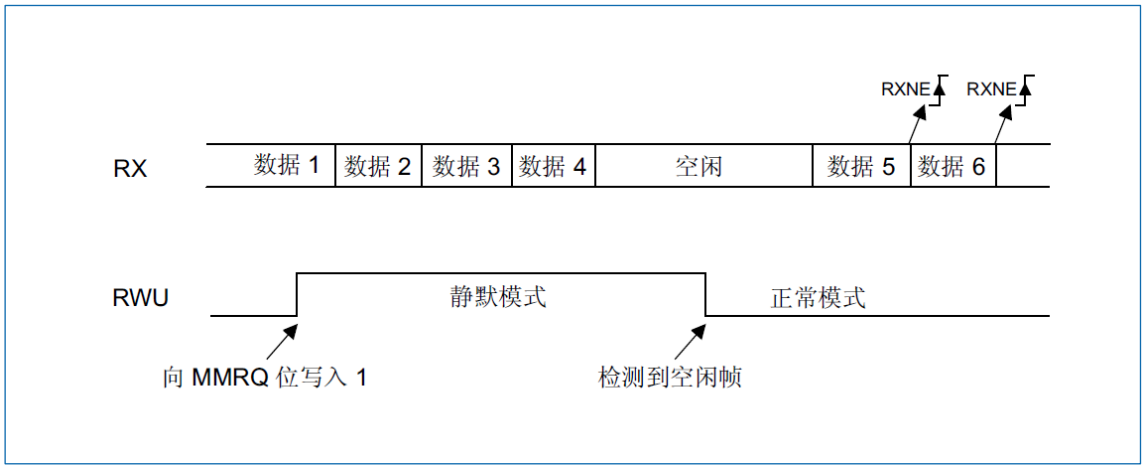

图 22-8 使用空闲线路检测时的静默模式

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，接收的地址字符既不会置位 RXNE 标志，也不会发出中断或 DMA 请求。

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。

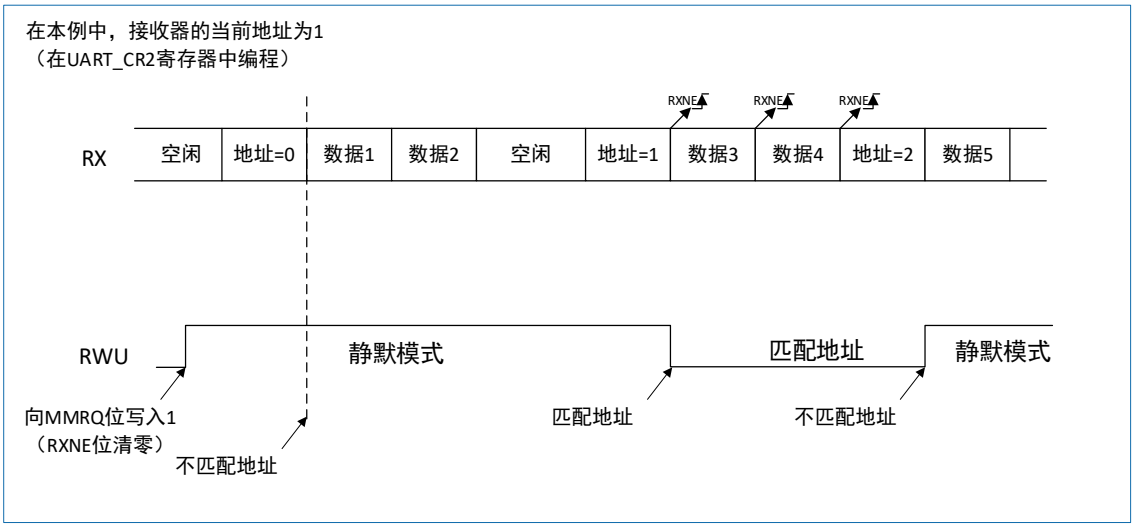

图 22-9 使用地址标记检测时的静默模式

注意：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]）。

22.3.7 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 位定义的帧长度，下表列出了可能的 UART 帧格式。

表 22-6 帧格式

M 位	PCE 位	UART 帧 (1)
00	0	|SB|8 位数据|STB|
00	1	|SB|7 位数据|PB|STB|
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，PB：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 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，校验则位为 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。

22.3.8 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。

22.3.9 DMA 模式下的 UART 连续通信

UART 能够使用 DMA 进行连续通信。接收缓冲区和发送缓冲区的 DMA 请求是独立生成的。

使用 DMA 进行发送

将 UARTx_CR3 寄存器中的 DMAT 位置 1，可使能 DMA 模式进行发送。当 TXE 位置 1 时，可将数据从 SRAM 区（通过 DMA 外设，请参见“8 直接存储器访问控制器（DMA）”）加载到 UARTx_TDR 寄存器。

要配置一个 DMA 通道以进行 UART 发送，需要按以下步骤操作：

1. 在 DMA 控制寄存器中写入 UARTx_TDR 寄存器地址，将其配置为传输的目标地址。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的源地址。
3. 在 DMA 控制寄存器中配置要传输的字节数。
4. 在 DMA 寄存器中配置通道优先级。
5. 根据应用的需求，在完成一半或全部传输后产生 DMA 中断。
6. 通过将 UARTx_ICR 寄存器中的 TCCF 位置 1，将 UARTx_ISR 寄存器中的 TC 标志清零。
7. 在 DMA 寄存器中激活该通道。

UART 使用 DMA 发送的时序图如下：

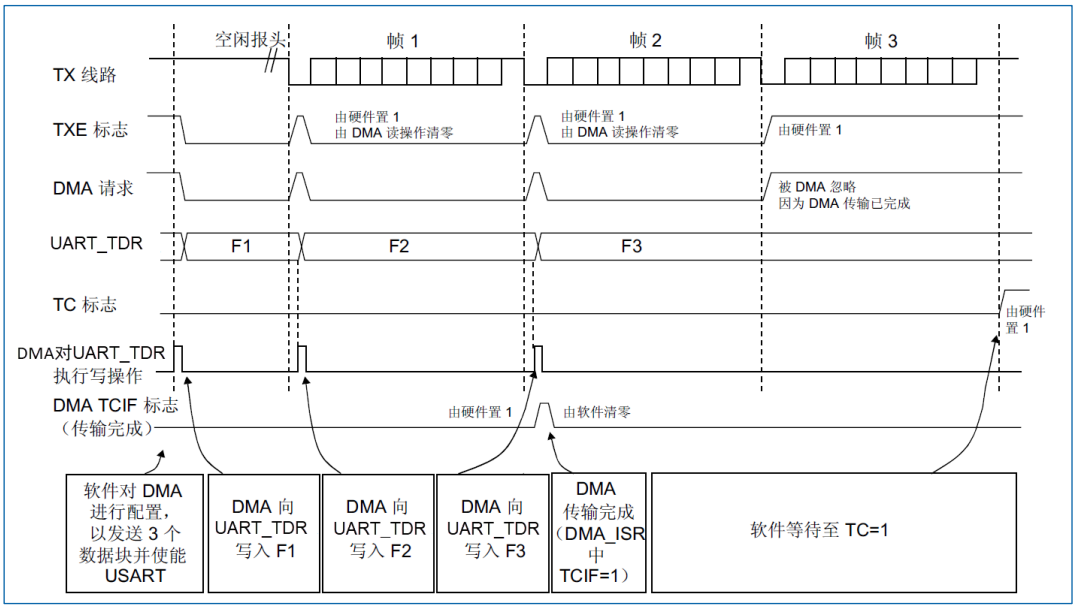

图 22-10 使用 DMA 进行发送

使用 DMA 进行接收

将 UARTx_CR3 寄存器中的 DMAR 位置 1，可使能 DMA 模式进行接收。接收数据字节时，数据会从 UARTx_RDR 寄存器加载到 SRAM 区域中。

配置一个 DMA 通道以进行 UART 接收，操作步骤如下：

1. 在 DMA 控制寄存器中写入 UARTx_RDR 寄存器地址，将其配置为传输的源地址。
2. 在 DMA 控制寄存器中写入存储器地址，将其配置为传输的目标地址。
3. 在 DMA 控制寄存器中配置要传输的总字节数。
4. 在 DMA 控制寄存器中配置通道优先级。
5. 根据应用的需求，在完成一半或全部传输后产生中断。
6. 在 DMA 控制寄存器中激活该通道。

UART 使用 DMA 接收的时序图如下：

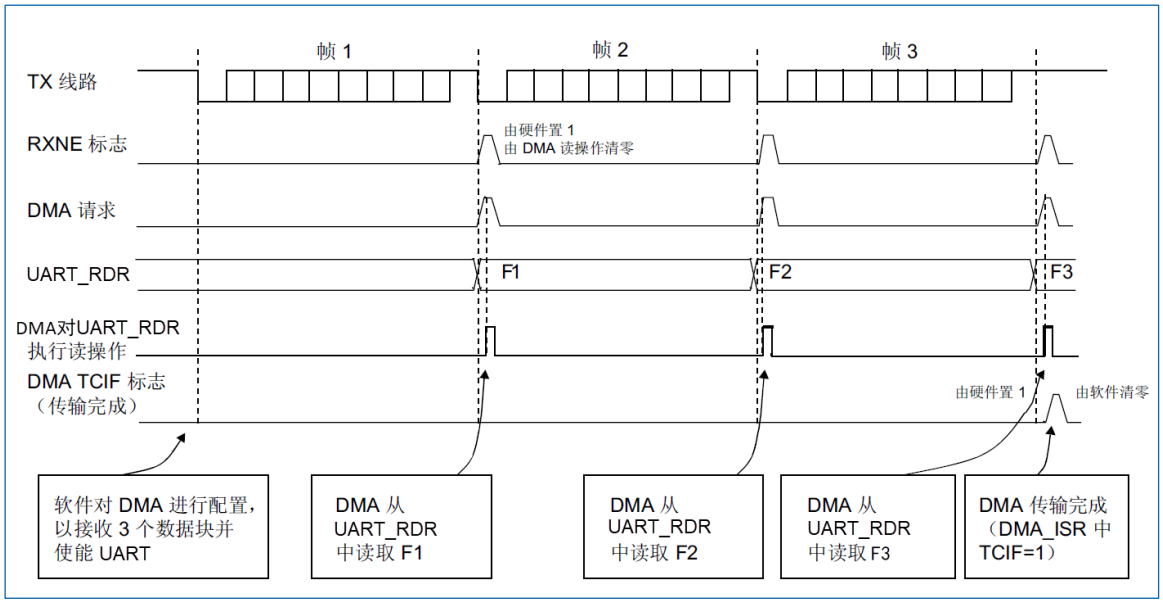

图 22-11 使用 DMA 进行接收

22.4 UART 低功耗模式

表 22-7 低功耗模式对 UART 的影响

模式	说明
睡眠模式	UART 不能将 MCU 从睡眠模式唤醒。
停机模式	UART 不能将 MCU 从停机模式唤醒。

22.5 UART 中断

表 22-8 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
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
字符匹配	CMF	CMIE

UART 中断映射图（请参见图 22-12）。

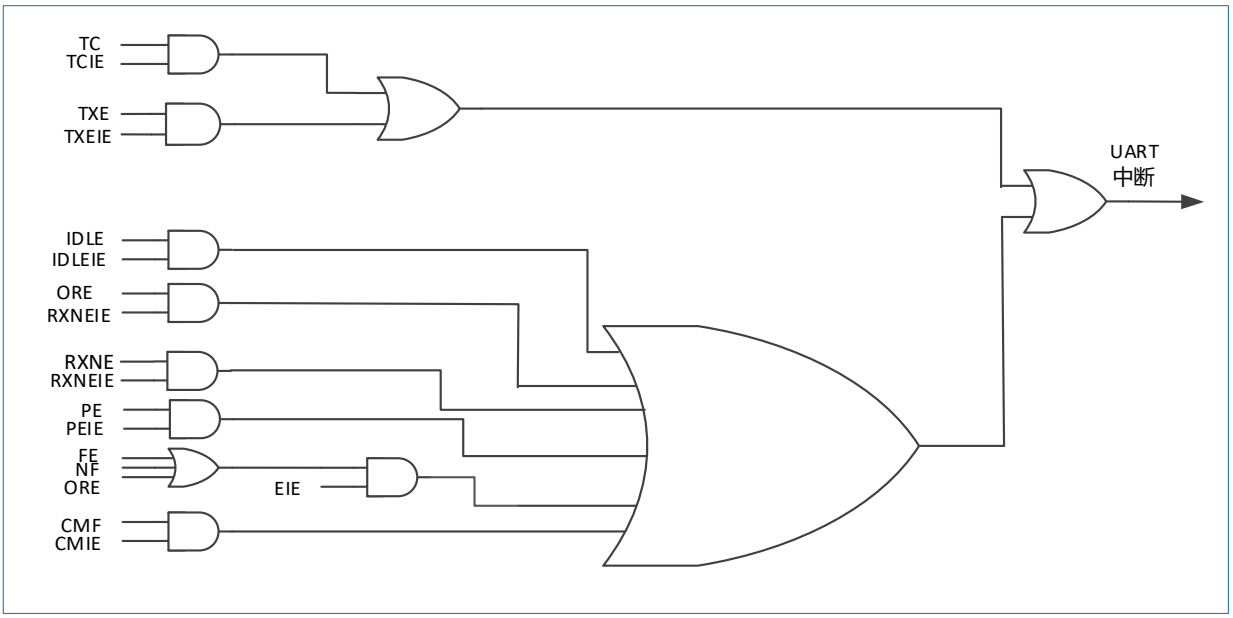

图 22-12 UART 中断映射图

22.6 UART1/2 寄存器

基地址：（UART1, UART2）=（0x4000 1400, 0x4001 2400）

空间大小：（UART1, UART2）=（0x400, 0x400）

22.6.1 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		DEAT[4:0]					DEDT[4:0]				
			rw			rw					rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	CMIE	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		rw

位 31:29	Res：保留 必须保持复位值。
位 28	M1：字长（Word length） M1 和 M0 位组成 M[1:0]，M[1:0]选择字长： ● 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 27:26	Res：保留 必须保持复位值。
位 25:21	DEAT[4:0]：驱动使能提前时间（Driver Enable assertion time） 该位域定义了 DE（驱动器使能）信号激活和第一个发送字节的起始位的时间间隔。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定）。 该位域只能在 UART 未被使能的时候（UE=0）改写。
位 20:16	DEDT[4:0]：驱动使能滞后时间（Driver Enable de-assertion time） 该位域定义一个发送消息的最后一个字节的停止位和释放 DE 信号之间的时间间隔。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定）。 如果 UARTx_TDR 寄存器在 DEDT 时间内被改写，新的数据只会在 DEDT 和 DEAT 时间都过去之后才会被发送。 该位域只能在 UART 未被使能的时候（UE=0）改写。
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	CMIE：字符匹配中断使能（Character match interrupt enable） 该位由软件置 1 和清零。 ● 0：中断禁止 ● 1：当 UARTx_ISR 寄存器中的 CMF 标志被置 1 时引发 UART 中断。
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（Word length） M1 和 M0 位组成 M[1:0]来决定串口字长，取值可参见 M1 位。 由软件置 1 和清零。 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=1 时是第九位，M=0 时是第八位），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 UARTx_ISR 寄存器中的 PE 被置 1 的时候会产生 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 UARTx_ISR 寄存器中的 TXE 被置 1 的时候会产生 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 UARTx_ISR 寄存器中的 TC 位被置 1 的时候会产生 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 UARTx_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 UARTx_ISR 寄存器中的 IDLE 位被置 1 的时候会产生 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 UARTx_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

22.6.2 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
SWAP	Res	STOP[1:0]		Res							ADDM7	Res			
rw		rw									rw				

位 31:24	ADD[7:0]：UART 的节点地址（Address of the UART node） 该位域给出 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。这个时候，整个收到的 8 位字节与 ADD[7:0]进行全面比较，如果匹配，将会引起 CMF 标志被硬件置起。 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 或者 8）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（VDD=1/idle、Gnd=0/mark）。 ● 1：TX 脚信号被反向。（VDD=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	SWAP：交换 TX/RX 引脚（Swap TX/RX pins） 该位由软件置 1 和清零。 ● 0：TX/RX 引脚按照标准引脚分配来使用。 ● 1：TX 和 RX 的引脚功能交换使用。这用于和其它 UART 口进行交叉互连的时候。 该位只能在 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 13:12	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 ● 01：保留 ● 10：2 个停止位 ● 11：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（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。

22.6.3 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
DEP	DEM	DDRE	OVRDIS	ONEBIT	Res			DMAT	DMAR	Res		HDSEL	Res		EIE
rw	rw	rw	rw	rw				rw	rw			rw			rw

位 31:16	Res：保留 必须保持复位值。
位 15	DEP：驱动使能输出脚的极性选择（Driver enable polarity selection） ● 0：DE 信号高有效。 ● 1：DE 信号低有效。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 14	DEM：驱动器使能模式（Driver enable mode） 它允许用户通过 DE（驱动使能）信号来激活外部收发器的控制端。 ● 0：DE 功能被禁止。 ● 1：DE 功能被打开。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 13	DDRE：在接收错误的时候禁止 DMA（DMA Disable on Reception Error） ● 0：在发生接收错误的时候不禁止 DMA。 相应的错误标志被置 1，但 RXNE 仍保持零以阻止数据溢出覆盖。作为结果，不会发起 DMA 请求，所以错误的数据不会被传输（因为没有 DMA 请求），但下一个正确的数据会被传送。 ● 1：在发生接收错误之后，DMA 被关闭。 相应的错误标志会随着 RXNE 的置 1 而被置 1。DMA 请求会被屏蔽，直到相关的错误标志被清零。这意味着软件必须先禁止 DMA 请求（DMAR=0）或者在清零错误标志前先清零 RXNE 标志。 该位只能在 UART 未被使能的时候（UE=0）改写。
位 12	OVRDIS：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 10:8	Res：保留 必须保持复位值。
位 7	DMAT：DMA 发送使能（DMA enable transmitter） 该位由软件置 1 和清零。 ● 0：关闭发送 DMA 模式

	● 1：使能 DMA 模式以发送数据
位 6	DMAR：DMA 接收使能（DMA enable receiver） 该位由软件置 1 和清零。 ● 0：关闭接收 DMA 模式 ● 1：使能 DMA 模式以接收数据
位 5:4	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要打开这个开关。 ● 0：中断禁止 ● 1：当 UARTx_ISR 寄存器中的 FE=1 或 ORE=1 或 NF=1 时，会产生中断。

22.6.4 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 15:0	BRR[15:0]：串口波特率的 UARTDIV 系数（The UARTDIV coefficient of the serial port wave rate） ● BRR[15:4]：UARTDIV 的整数部分（Mantissa of UARTDIV） 这 12 位定义 UART 分频器除法因子的整数部分。 ● BRR[3:0]：UARTDIV 的小数部分（Fraction of UARTDIV） 这 4 位定义 UART 分频器除法因子的小数部分。 当 OVER8=0，BRR[3:0]=UARTDIV[3:0]。当 OVER8=1，BRR[3:0]=UARTDIV[3:0]右移 1 位。

22.6.5 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。

22.6.6 中断和状态寄存器（UARTx_ISR）（x=1..2）

偏移地址：0x1C

复位值：0x00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									REACK	TEACK	Res	RWU	SBKF	CMF	BUSY
									r	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 UARTx_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。静默模式控制顺序

	（地址还是空闲）用 UARTx_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 UARTx_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 UARTx_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：未发送断开字符 ● 1：断开字符将会被发送
位 17	CMF：字符匹配标志（Character match flag） 当收到一个字符和 ADD[7:0]中设置的内容相同时，由硬件置 1。由软件向 UARTx_ICR 寄存器的 CMCF 位写 1，可以清除这个标志。 如果 UARTx_CR1 寄存器中的 CMIE 位是 1，就会产生中断请求。 ● 0：未发现字符匹配 ● 1：有发现字符匹配
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 UARTx_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 UARTx_TDR 寄存器写数据就会同时清掉该位。如果 UARTx_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 UARTx_CR1 寄存器中的 TCIE 位是 1，就会产生中断请求。向 UARTx_TDR 寄存器再次写入数据，或者向 UARTx_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 UARTx_RDR 寄存器中时，该位被硬件置 1。读取 UARTx_RDR 寄存器的数据就会同时清掉该位。RXNE 标志也可以通过对 UARTx_RQR 寄存器中的 RXFRQ 位写 1 来清除。如果 UARTx_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 UARTx_CR1 寄存器中的 IDLEIE 位是 1，就会产生中断请求。由软件向 UARTx_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 UARTx_ICR 寄存器的 ORECF 位写 1，可以清除这个标志。如果 UARTx_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（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 UARTx_ICR 寄存器的 NCF 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 UARTx_ICR 寄存器的 FECF 位写 1，可以清除这个标志。在智能卡模式中发送数据时，当重发尝试的次数达到上限，由没有收到成功的回应（卡一直响应 NACK）的时候，该位也会被硬件置 1。如果 UARTx_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 UARTx_ICR 寄存器的 PECF 位写 1，可以清除这个标志。如果 UARTx_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

22.6.7 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														CMCF	Res
														w_r1	

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 31:18	Res：保留 必须保持复位值。
位 17	CMCF：字符匹配标志的清除（Character match clear flag） 对该位写 1，会清除 UARTx_ISR 寄存器中的 CMF 标志位。
位 16:7	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 UARTx_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 UARTx_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 UARTx_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 UARTx_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 UARTx_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 UARTx_ISR 寄存器中的 PE 标志位。

22.6.8 数据接收寄存器（UARTx_RDR）（x=1..2）

偏移地址：0x24

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							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。

22.6.9 数据发送寄存器（UARTx_TDR）（x=1..2）

偏移地址：0x28

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							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（UARTx_CR1 中的 PCE=1），向最高位（位 7 还是位 8 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。

23 串行外设接口（SPI）

SPI 接口可用于使用 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。

23.1 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
- 具有一条双向数据线的双线半双工同步传输
- 具有一条单向数据线的双线单工同步传输
- 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，最高可达 $f_{PCLK}/2$
- 从模式频率最高可达 $f_{PCLK}/2$
- 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
- TX 和 RX FIFO 带 DMA 功能
- 支持 SPI TI 模式

23.2 SPI 实现

表 23-1 SPI 实现

SPI 特性	SPI1
硬件 CRC 计算	支持
RX/TX FIFO	支持
TI 模式	支持
I2S 模式	不支持

23.3 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。

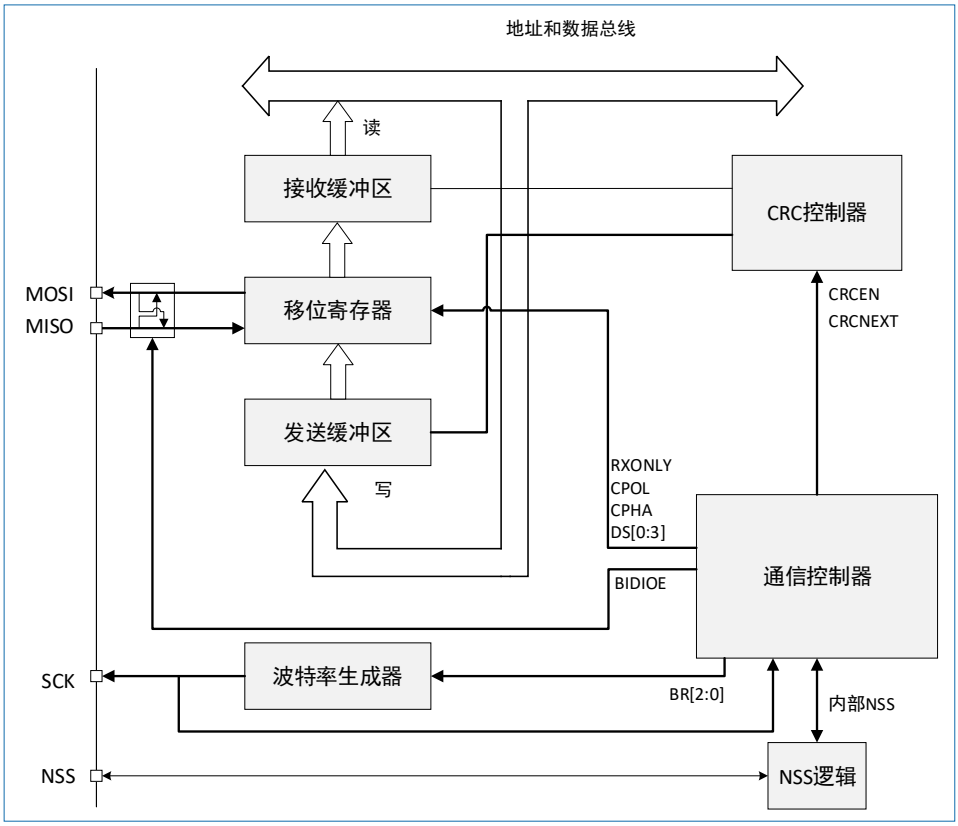

图 23-1 SPI 框图

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“23.3.4 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。其它信号可以根据 SPI 节点间的数据交换及其从器件选择信号管理进行添加。

23.3.1 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。

23.3.1.1 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。

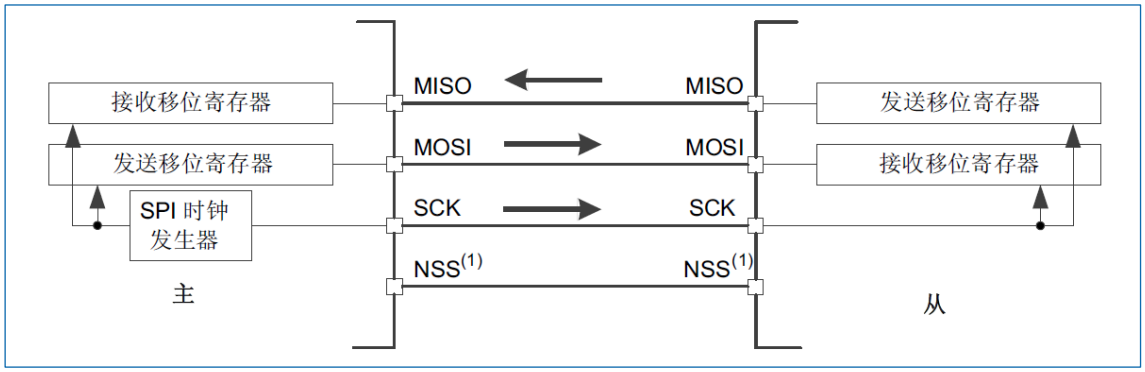

图 23-2 全双工单个主器件/单个从器件应用

(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“23.3.4 从器件选择（NSS）引脚管理”。

23.3.1.2 半双工通信

通过将 SPI1_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的 CR1 寄存器中的 BDIOE 位进行选择。

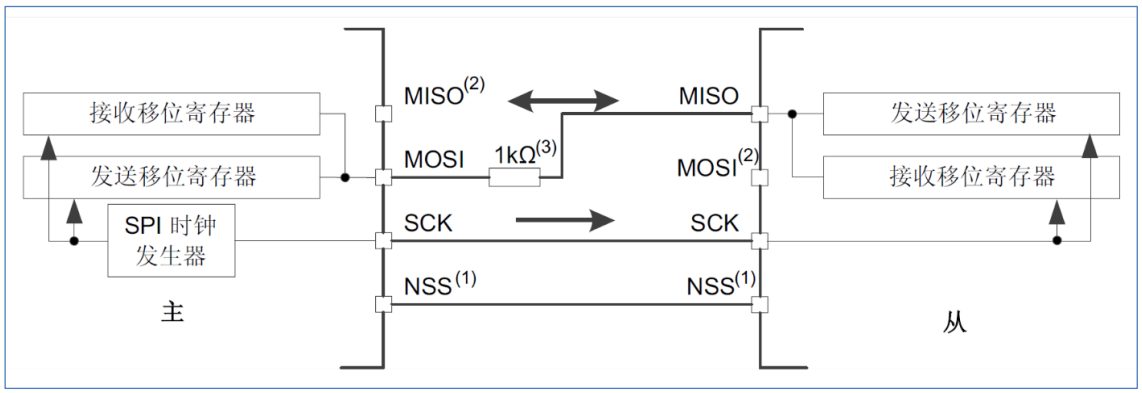

图 23-3 半双工单个主器件/单个从器件应用

(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，请参见章节“23.3.4 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。

23.3.1.3 单工通信

通过 SPI1_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“23.3.3 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 MOSI，禁止 Tx，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。

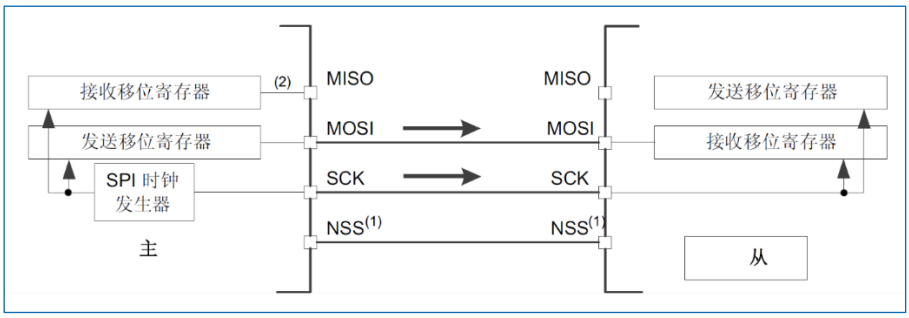

图 23-4 单工单个主器件/单个从器件应用（主器件为只发送模式/从器件为只接收模式）

(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“23.3.4 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）。

23.3.2 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 23-5）。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。

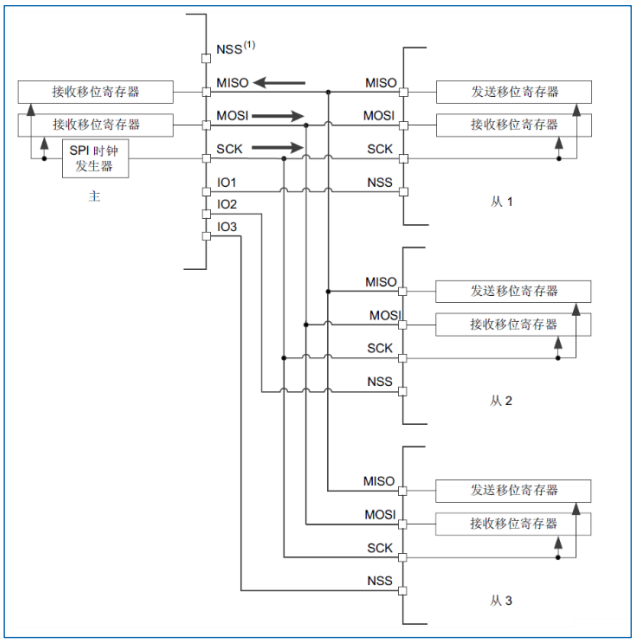

图 23-5 主器件和三个独立的从器件

(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（请参见“6.2.6 I/O 复用功能输入输出”）。

23.3.3 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 引脚向其它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“23.5.3 SPI1 状态寄存器（SPI1_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）。

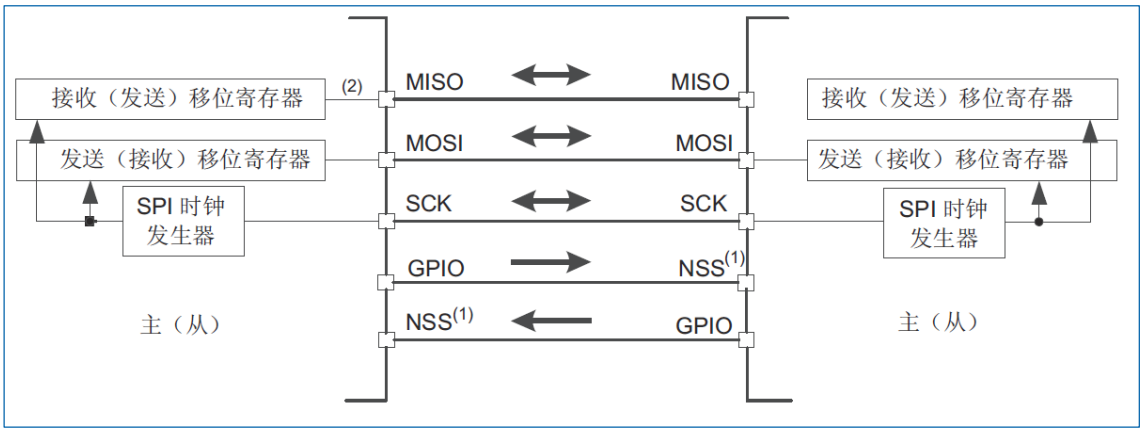

图 23-6 多主器件应用

(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。

23.3.4 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 SPI1_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 SPI1_CR1 寄存器中的 SSI 位的值决定内部驱动从器件选择信息。此时，外部 NSS 引脚空闲，可供其它应用使用。
- 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（SPI1_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。

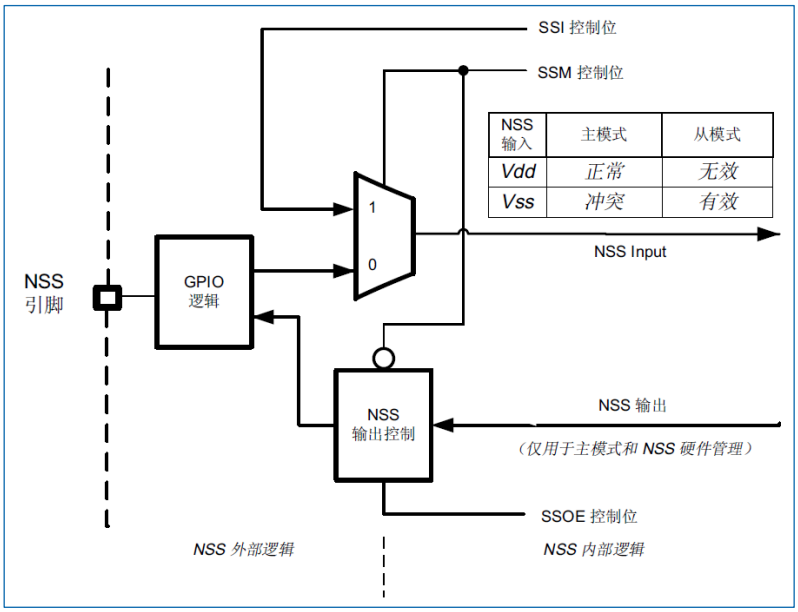

图 23-7 硬件/软件从器件选择管理

23.3.5 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。

23.3.5.1 时钟相位和极性控制

通过 SPI1_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 23-8 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 SPI1_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）。

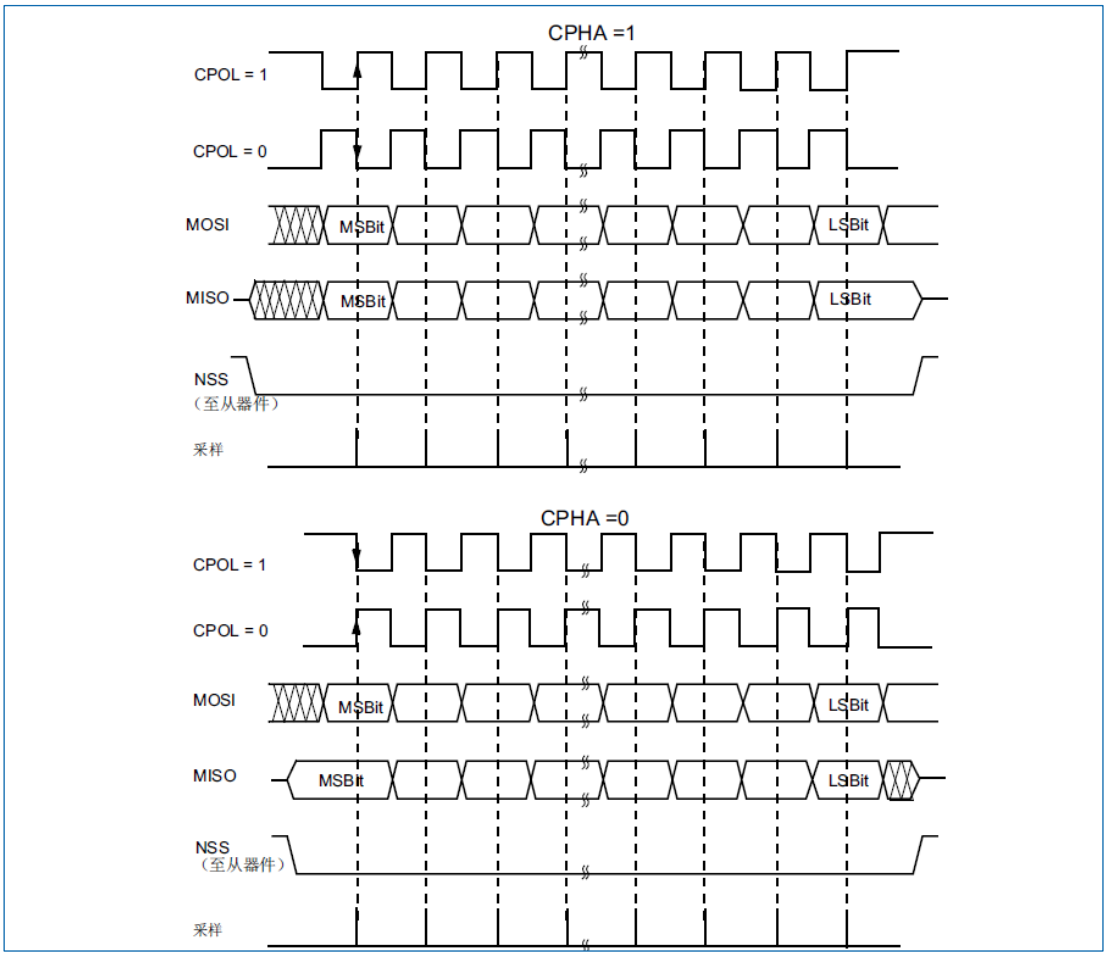

图 23-8 数据时钟时序图

说明：数据位的顺序取决于 LSBFIRST 位的设置。

23.3.5.2 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 SPI1_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 SPI1_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。

23.3.6 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 SPI1_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 SPI1_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 SPI1_DR 寄存器的读取访问大小对齐。

F. 如果使能 DMA 模式，需要置位 LDMA_TX 和 LMDA_RX 位。

4. 对 SPI1_CRCPR 寄存器执行写操作：需要时配置 CRC 多项式。

5. 对相应的 DMA 寄存器执行写操作：如果使用 DMA 数据流，请在 DMA 寄存器中配置 SPI TX 和 RX 专用的 DMA 数据流。

上文注释说明：
(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。

23.3.7 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。

有关如何处理 DMA 的详细信息，请参见“23.3.10 使用 DMA（直接存储器寻址）进行通信”。

23.3.8 数据发送和接收过程

23.3.8.1 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。每个传输方向都有自己的 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“23.3.15 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“23.3.14 TI 模式”）。

对 SPI1_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 SPI1_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 SPI1_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 SPI1_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“23.3.11 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。

23.3.8.2 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），或单工（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，也就是说从设备无法在 SPI 模式下及时准确地准备好数据，从机最好使用 DMA，尤其是当数据帧较短且总线速率较高时。

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“23.3.4 从器件选择（NSS）引脚管理”）。

当 BSY 位被设置时，它表示正在进行数据帧的传输。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检查 BSY 和 FTLVL[1:0]位是很有必要的，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。

23.3.8.3 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 SPI1_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 23-9 提供了组包方式顺序处理数据的一个例子。在一次对 SPI1_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 SPI1_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 的阈值和后续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 SPI1_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。

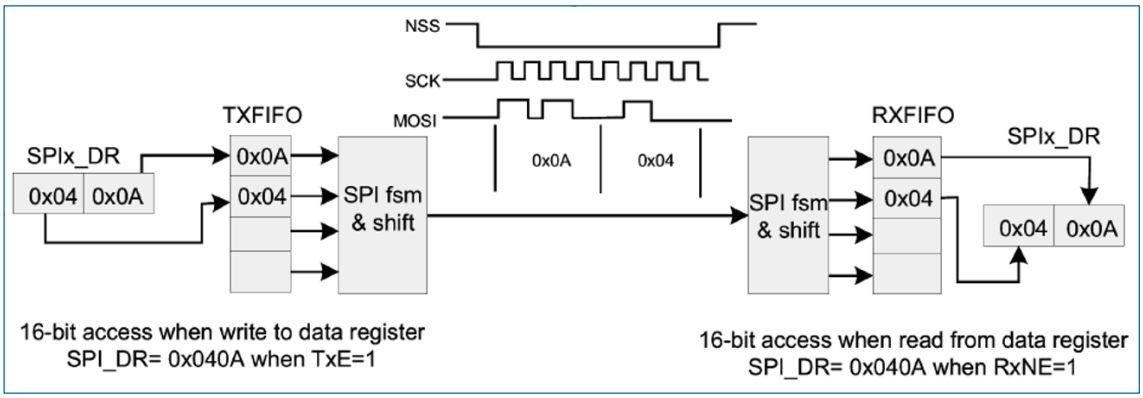

图 23-9 发送和接收 FIFO 中的数据组包

23.3.9 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。在外设时钟停止及在系统进入低功耗模式前完成此操作是十分重要的。否则会破坏正在进行的交互。在某些模式下，该禁止步骤是停止所进行的连续通信的唯一方式。

当处于全双工或只发送模式下的主器件停止提供待发送的数据时，可结束任何交互。在这种情况下，时钟在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，以防止一些虚拟字节交换（参见数据组包部分），在这些模式下禁用 SPI 之前，用户必须遵循标准的禁用过程。当一个帧传输正在进行或下一个数据帧存储在 TXFIFO 中，SPI 在主发送机被禁用时，SPI 行为是不保证的。

当主机处于任意接收模式时，停止连续时钟的唯一方法是通过配置 SPE=0 禁用该外设。这个必须发生在特定的时间窗口内，即，最后一个数据帧传输期间：从第一个比特的采样时间开始，到最后一个比特开始传输之前，（为了得到一个完整的预期数量数据帧，并防止在读取最后一个有效的数据帧后读取任何额外的“假”数据）。在该模式下禁用 SPI 必须遵循特定的步骤。

当 SPI 被禁用时，已被接收但没有被读取的数据将继续存储在 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。为了防止有未读的数据，可通过以下操作确保在禁用 SPI 时 RXFIFO 是空的：

- 通过使用正确的禁用过程，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（详见章节“5.3.5 APB1 外设复位寄存器（RCC_APB1RSTR）”）。

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。还可以在需要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。
- 最后一个数据帧或 CRC 帧传输仍在外设总线中处理时，来自 DMA 的传输数据流已完成。

正确的 SPI 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 SPI1_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。

23.3.10 使用 DMA（直接存储器寻址）进行通信

为了以最大速度工作并且在数据寄存器读/写过程中避免上溢，SPI 提供了 DMA 功能，该功能采用了简单的请求/应答协议。

将 SPI1_SR 寄存器中的使能位 TXE 或 RXNE 置 1 时，将请求 DMA 访问。必须向发送缓冲区和接收缓冲区发出单独的请求。

- 在发送过程中，每次 TXE 位置 1 都会发出 DMA 请求。然后，DMA 将对 SPI1_DR 寄存器执行写操作。
- 在接收过程中，每次 RXNE 位置 1 都会发出 DMA 请求。然后，DMA 将对 SPI1_DR 寄存器执行读操作。

有关 DMA 发送和接收波形的说明，请参见图 23-10 和图 23-11。

当 SPI 仅用于发送数据时，可以只使能 SPI TX DMA 通道。在这种情况下，OVR 标志会置 1，因为未读取接收的数据。当 SPI 仅用于接收数据时，可以只使能 SPI RX DMA 通道。

在发送模式下，DMA 写入所有要发送的数据（DMA_ISR 寄存器中的 TCIF 标志置 1）后，可以对 BSY 标志进行监视，以确保 SPI 通信已完成。在禁止 SPI 或进入停机模式前必须执行此步骤，以避免损坏最后一次发送数据。软件必须首先等待 FTLVL[1:0]=00，再等待 BSY=0。

通过 DMA 开始通信时，为防止 DMA 通道管理引发错误事件，必须按顺序执行以下步骤：

1. 如果使用 DMA RX，通过 SPI1_CR2 寄存器中的 RXDMAEN 位来使能 DMA 接收缓冲区；
2. 如果使用数据流，通过 DMA 寄存器来使能 TX 和 RX 的 DMA 数据流；
3. 如果使用 DMA TX，通过 SPI1_CR2 寄存器中的 TXDMAEN 位来使能 DMA 发送缓冲区。
4. 将 SPIx_CR1 寄存器中的 SPE 位置 1，使能 SPI。

要关闭通信，必须按顺序执行以下步骤：

1. 如果使能了 DMA 数据流，通过 DMA 寄存器来禁止 TX 和 RX 的 DMA 数据流。
2. 将 SPIx_CR1 寄存器中的 SPE 置 0 来禁止 SPI。
3. 如果使用 DMA TX 和/或 DMA RX，通过将 SPI1_CR2 寄存器中的 TXDMAEN 和 RXDMAEN 位清零来禁止 DMA 发送缓冲区和接收缓冲区。

23.3.10.1 DMA 与传输组包

如果传输由 DMA 来管理（TXDMAEN 和 RXDMAEN 在 SPI1_CR2 寄存器中设置），会根据 SPI TX 和 RX DMA 通道的 SPI 配置状态来自动将组包模式启用或禁用。如果 DMA 通道设置为按 16 位访问，而 SPI 数据的大小是小于或等于 8 位，那么组包模式会被启用。DMA 会自动管理对 SPI1_DR 寄存器的写操作。

如果启用了组包模式，而传输的数据个数又不一定是偶数，则 LDMA_TX/LDMA_RX 位必须置 1。而 SPI 则会认为在最后的 DMA 传输中只有一个有效的传输或接收字节（更多详情，请参见“23.3.8.3 数据组包”。）

23.3.10.2 通信图

本节将介绍一些典型的时序方案。无论 SPI 事件是通过轮询、中断还是 DMA 处理，这些模式都是有效的。为简单起见，这里使用 LSBFIRST=0、CPOL=0 和 CPHA=1 设置作为常见的假设。没有提供完整的 DMA 流配置。

以下注释为本小节所有图的共同注释：

1. 当 NSS 激活和 SPI 启用时，从机开始控制 MISO 线，当其中一个被释放时，从机断开连接。必须为从机提供足够的时间，以便其在事务开始之前提前准备专用于主机的数据。

2. 在主机上，仅在使能 SPI 后，SPI 外设控制 MOSI 和 SCK 信号（偶尔也控制 NSS 信号）。如果 SPI 被禁用，SPI 外设将与 GPIO 逻辑断开连接，因此这些线上的电平完全依赖于 GPIO 设置。

3. 如果通信是连续时，主机上的 BSY 信号在两帧之间保持为高；但是在从机上，在两帧数据间 BSY 信号会拉低一个时钟周期。

4. 只有 TXFIFO 为满时，TXE 位才会被置 0。

5. DMA 仲裁过程在设置 TXDMAEN 位之后开始。设置完 TXIE 后，会产生 TXE 中断。当 TXE=1 时，开始向 TXFIFO 传输数据，直到 TXFIFO 满了或 DMA 传输完成。

6. 如果所有要发送的数据都可以装入 TXFIFO，DMA TX TCIF 标志甚至可以在 SPI 总线上开始通信之前置位。在 SPI 事务完成之前，这个标志始终置起。

7. 数据包的 CRC 值是在 SPI1_TXCRCR 和 SPI1_RXCRCR 寄存器中逐帧连续计算的。CRC 信息在整个数据包完成后，自动通过 DMA 进行处理（必须将 TX 通道设置为要处理的数据帧数）或通过软件进行处理（用户在最后一次处理数据帧时必须处理 CRCNEXT 位）。

8. 虽然在 SPI1_TXCRCR 中计算的 CRC 值简单地由发送器发送出去，但接收到的 CRC 信息被加载到 RXFIFO 中，然后与 SPI1_RXCRCR 寄存器内容进行比较；如果有任何差异，可以在这里引发 CRC 错误标志。这就是为什么用户必须注意从 FIFO 刷新这些信息，通过软件阅读所有 RXFIFO 的存储内容，或者通过 DMA 适当数量的数据帧时预设 RX 频道（数据帧数+ CRC 的帧数）（参见示例假设）的设置。

9. 在数据组包模式下，TXE 和 RXNE 事件是配对的，每个 FIFO 的读/写访问是 16 位宽，直到数据帧的数量是偶数。如果 TXFIFO 是 3/4 满，FTLVL 状态保持在 FIFO 满水平。这就是为什么在 TXFIFO 变为 1/2 满之前，最后的奇数数据帧不能被存储的原因。当设置 LDMA_TX 控制时，此帧被存储到 TXFIFO 中，并通过软件或 DMA 自动访问 8 位。

10. 为了以组包方式接收最后的奇数数据帧，当最后的数据帧被处理时，RX 阈值必须更改为 8 位，要么通过软件设置 FRXTH=1，要么在设置 LDMA_RX 时通过 DMA 内部信号自动更改。

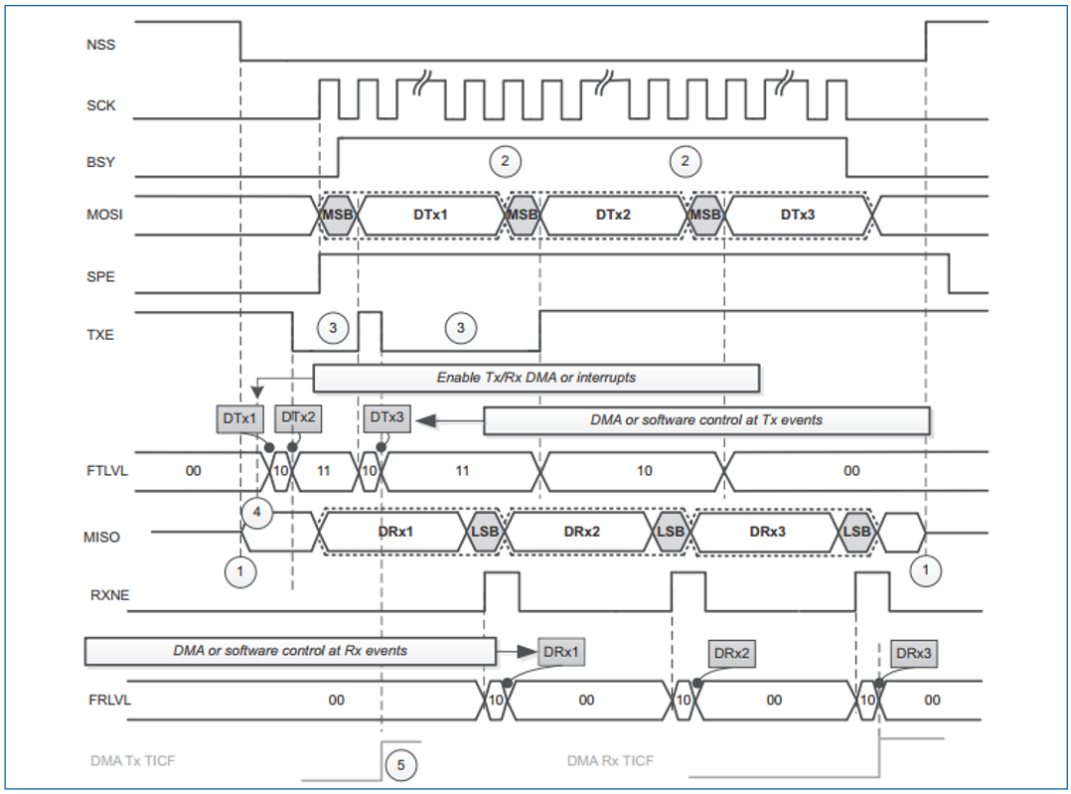

图 23-10 主模式全双工通信

假设为主机全双工通信的示例：

- 数据大小 > 8 位

如果使用 DMA：

- DMA 处理的发送帧数量为 3
- DMA 处理的接收帧数量为 3

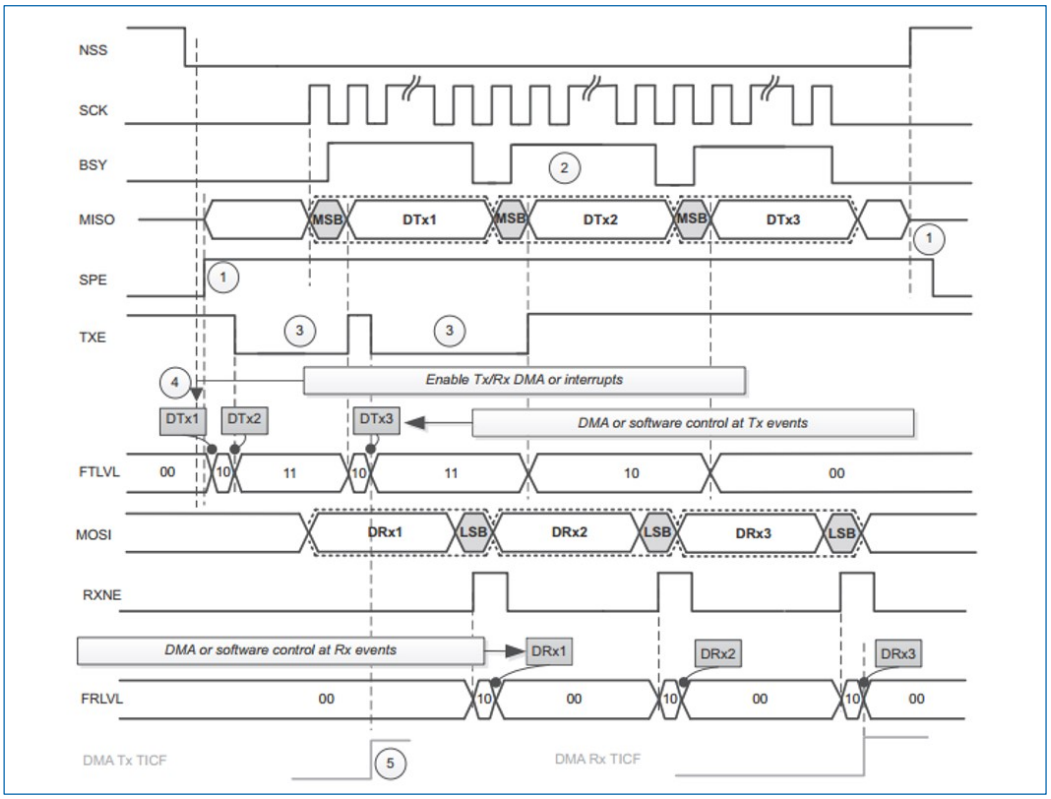

图 23-11 从模式全双工通信

假设为从机全双工通信的示例：

- 数据大小 > 8 位

如果使用 DMA：

- DMA 处理的发送帧数量为 3
- DMA 处理的接收帧数量为 3

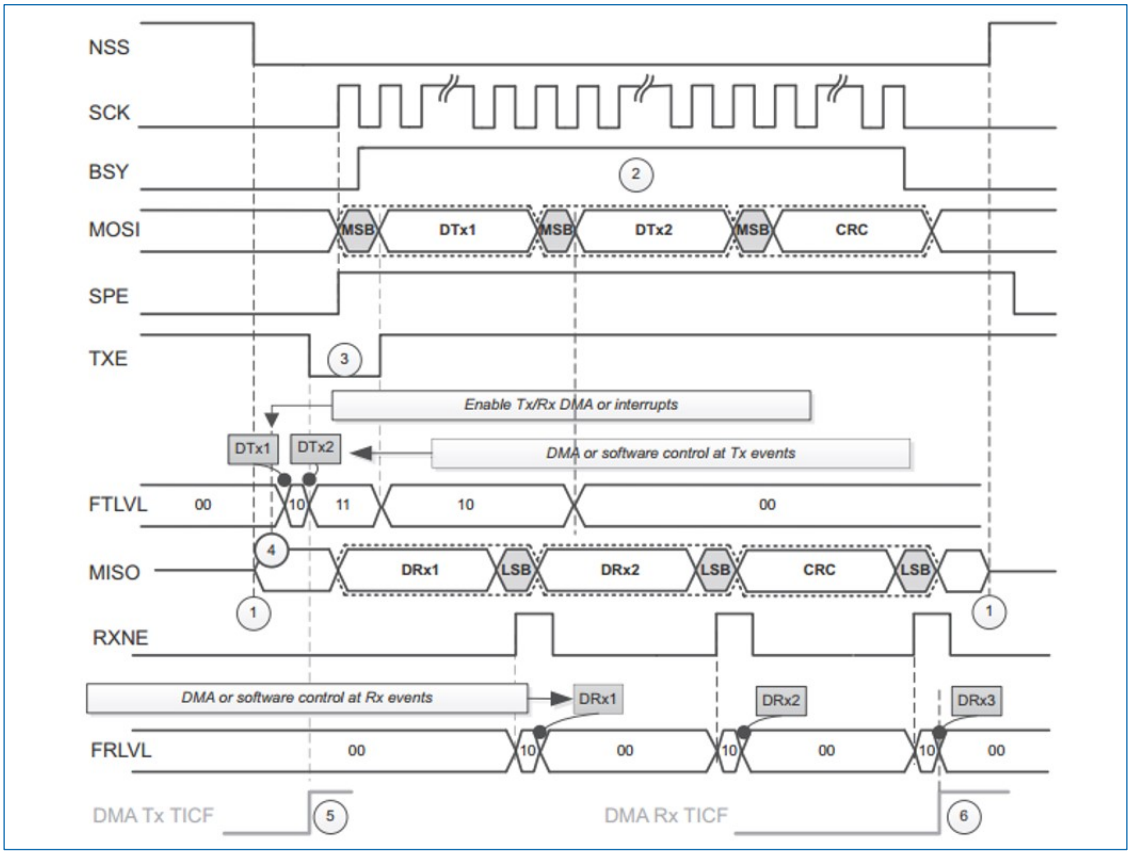

图 23-12 主机全双工通信带 CRC 校验

假设为主机全双工通信带 CRC 校验的示例：

- 数据大小=16 位
- 启用 CRC

如果使用 DMA：

- DMA 处理的发送帧数量为 2
- DMA 处理的接收帧数量为 3

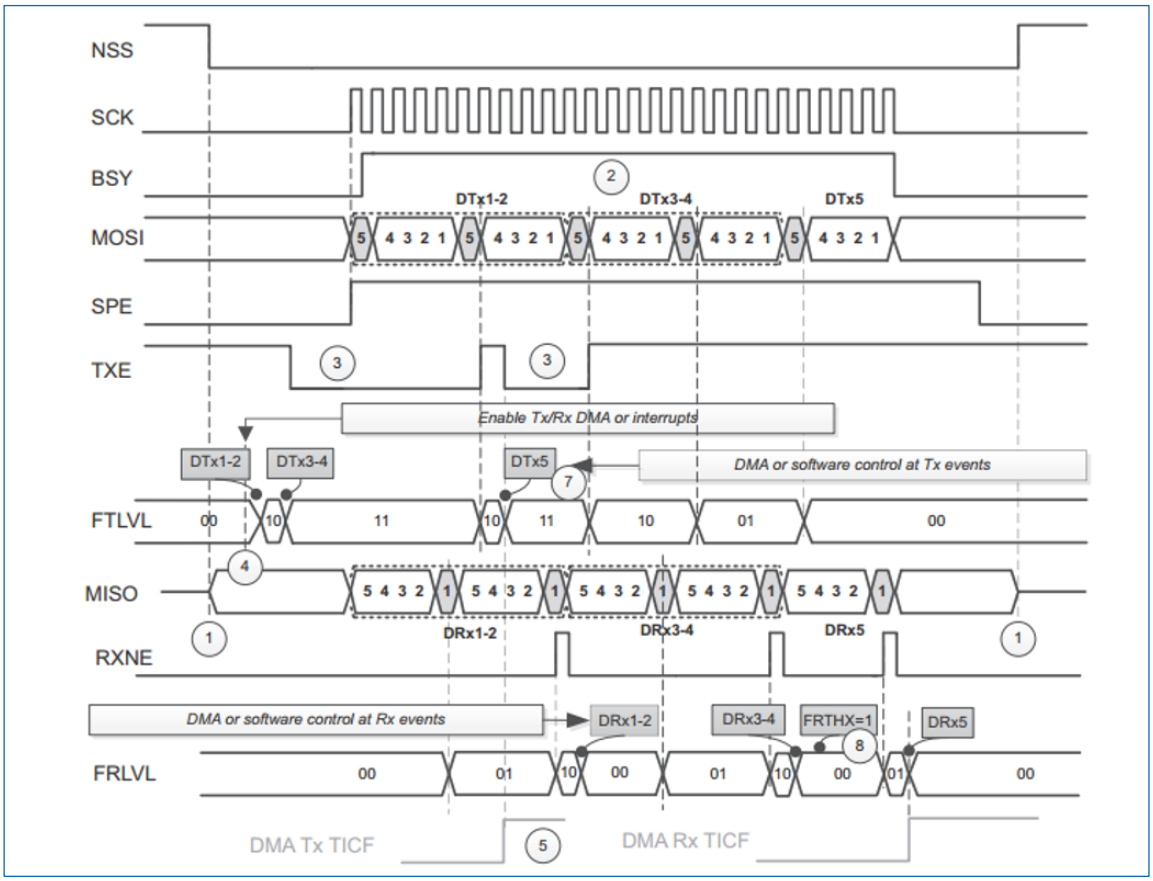

图 23-13 主模式全双工模式使用数据包传输

假设为主机全双工通信带 CRC 校验的示例：

- 数据大小=5 位
- 读写 FIFO 通过 16 位访问来完成
- FRXTH=0

如果使用 DMA：

- DMA 处理的发送帧数量为 2
- DMA 处理的接收帧数量为 3
- DMA TX 通道和 DMA RX 通道都设置为 16 位
- LDMA_TX=1 和 LDMA_RX=1

23.3.11 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。TXE 标志置 1 并保持，直到 TXFIFO 深度低于或等于 FIFO 深度的 1/2。当 TXFIFO 深度大于 1/2 时，TXE 位自动清除。如果 SPI1_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。

接收缓冲区非空（RXNE）

RXNE 标志的设置取决于 SPI1_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 SPI1_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。

23.3.12 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或 DMA 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 SPI1_DR 寄存器的读访问和对 SPI1_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 SPI1_SR 寄存器执行读或写访问。
2. 对 SPI1_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 位恢复到初始状态。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 SPI1_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。如果移位寄存器中接收的值与 SPI1_RXCRC 的值不匹配，SPI1_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，SPI1_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 SPI1_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。

23.3.13 NSS 脉冲模式

该模式由 SPI1_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（SPI1_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。

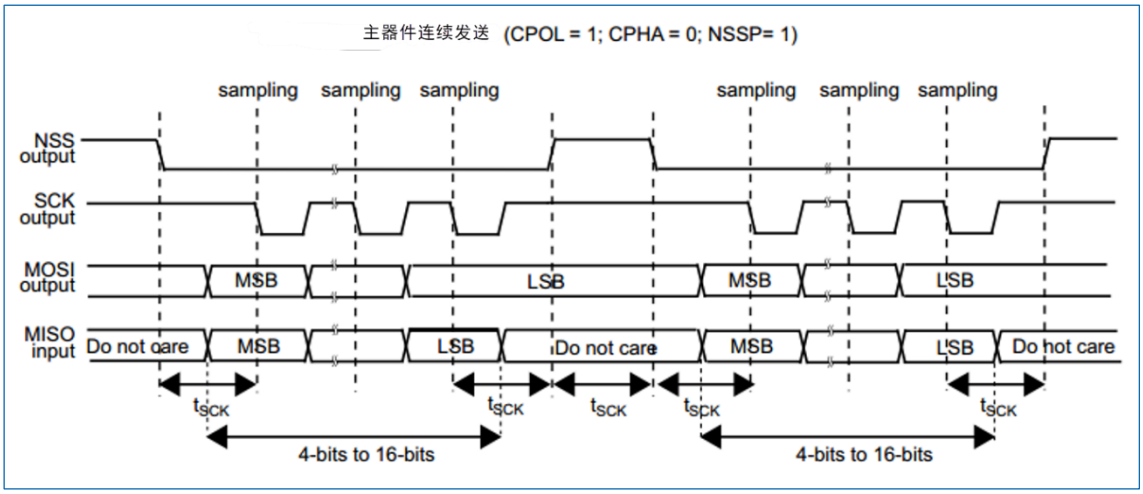

图 23-14 SPI Motorola 主模式 NSS 脉冲生成

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。

23.3.14 TI 模式

主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPI1_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 SPI1_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 SPI1_CR1 和 SPI1_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 23-15）。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 SPI1_CR1 寄存器的 BR[2:0]位设置的波特率值。具体公式如下：

$$\frac{t_{baudrate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baudrate}}{2} + 6 \times t_{pclk}$$

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 23-15 给出了选择 TI 模式时的 SPI 通信波形。

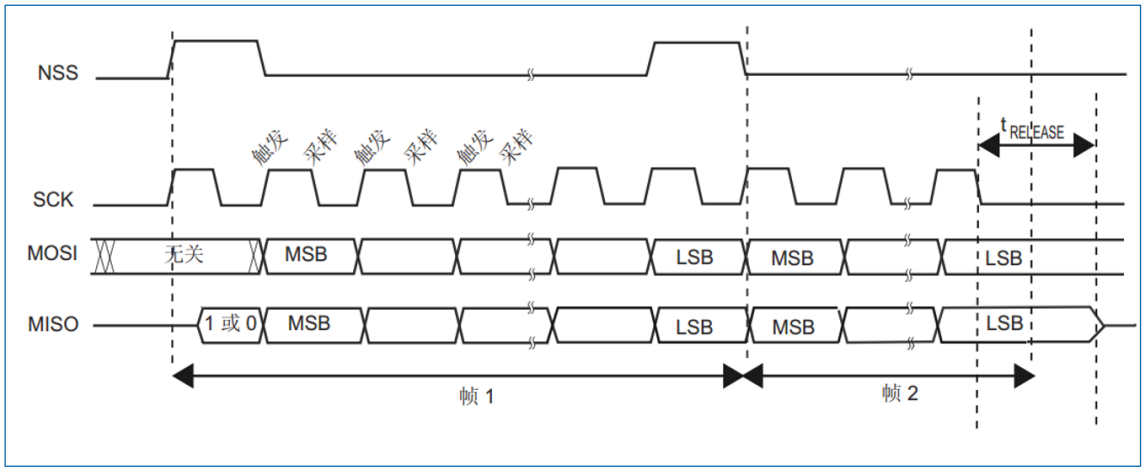

图 23-15 TI 模式传输

23.3.15 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。

23.3.15.1 CRC 原理

在使能 SPI（SPE=1）前，通过将 SPI1_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 SPI1_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 或 DMA 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。

23.3.15.2 CPU 管理的 CRC 传输

通信开始后将一直持续到发送或接收 SPI1_DR 寄存器中的最后一个数据帧。之后，SPI1_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。

与任何其它数据帧一样，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 SPI1_RXCRC 寄存器中的值进行比较。软件必须校验 SPI1_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 SPI1_DR 寄存器中进行读取，以将 RXNE 标志清零。

23.3.15.3 DMA 管理的 CRC 传输

当使能的 SPI 通信支持 CRC 通信和 DMA 模式时，在通信结束时会自动发送和接收 CRC（在只接收模式下读取 CRC 数据时除外）。CRCNEXT 位并非一定要通过软件来处理。SPI 发送 DMA 通道计数器必须设置为要发送的数据帧数，其中不包括 CRC 帧。在接收器侧，接收的 CRC 值在传输结束时通过 DMA 自动处理，但用户必须注意刷新 SPI1_DR 中接收的 CRC 信息。在全双工模式下，接收 DMA 通道的计数器可以设置为接收的数据帧数，包括 CRC 校验，例如 8 位数据帧经过 16 位 CRC 校验的具体情况为：

DMA_RX = Numb_of_data + 2

在仅接收模式中，DMA 接收信道计数器应该仅包含被传输数据的数量，不包括 CRC 计算。然后基于从 DMA 的完全传输，所有的 CRC 值必须由软件从 FIFO 读回，因为它在这种模式下作为单个的缓冲区工作。

在数据和 CRC 传输结束时，如果在传输期间发生损坏，则设置 SPI1_SR 寄存器中的 CRCERR 标志。

如果采用数据组包方式，如果数据数量为奇数，则 LDMA_RX 位需要管理。

23.3.15.4 复位 SPI1_TXCRC 和 SPI1_RXCRC 值

当使能 CRC 计算时，SPI1_TXCRC 和 SPI1_RXCRC 值自动清零。这就允许使用 DMA 循环模式（只接收模式中不可用）来实现没有任何间断的传输数据（中间有几个数据块涉及 CRC 检查阶段）。

要将 CRC 清零，其操作步骤如下：

1. 禁止 SPI。
3. 将 CRCEN 位清零。
4. 使能 CRCEN 位。
5. 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。

23.4 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 23-2 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	

23.5 SPI 寄存器

基地址：0x4000 3800

空间大小：0x400

23.5.1 SPI1 控制寄存器 1（SPI1_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
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 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：第二个时钟沿对准第一位数据。

23.5.2 SPI1 控制寄存器 2（SPI1_CR2）

偏移地址：0x04

复位值：0x0700

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	LDMA_TX	LDMA_RX	FRXTH	DS[3:0]				TXEIE	RXNEIE	ERRIE	FRF	NSSP	SSOE	TXDMAEN	RXDMAEN
	rw	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	LDMA_TX：最后一次 DMA 发送（Last DMA transfer for transmission） 该位是在数据组包模式中，用来定义 DMA 数据传输的总数是奇数还是偶数。它只有在 SPI1_CR2 寄存器的 TXDMAEN 位被设置，并且组包模式已开启（数据长度小于等于 8 位和写入访问 SPI1_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPI1_CR1 寄存器的 SPE=0）时写入。 ● 0：传输的数据数量为偶数 ● 1：传输的数据数量为奇数
位 13	LDMA_RX：最后一次 DMA 接收（Last DMA transfer for reception） 该位是在数据组包模式中，用来定义 DMA 数据接收的总数是奇数还是偶数。它只有在 SPI1_CR2 寄存器的 RXDMAEN 位被设置，并且组包模式已开启（数据长度小于等于 8 位和写入访问 SPI1_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPI1_CR1 寄存器的 SPE=0）时写入。 ● 0：传输的数据数量为偶数 ● 1：传输的数据数量为奇数
位 12	FRXTH：FIFO 接收门限（FIFO reception threshold） 该位用来设置触发 RXNE 事件的 RXFIFO 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	TXDMAEN：TX 缓冲 DMA 使能（TX buffer DMA enable） 当该位被设置，TXE 标志被设置时，产生一个 DMA 请求。 ● 0：TX 缓冲 DMA 禁止 ● 1：TX 缓冲 DMA 使能
位 0	RXDMAEN：RX 缓冲 DMA 使能（RX buffer DMA enable） 当该位被设置，RXNE 标志被设置时，产生一个 DMA 请求。 ● 0：RX 缓冲 DMA 禁止 ● 1：RX 缓冲 DMA 使能

23.5.3 SPI1 状态寄存器（SPI1_SR）

偏移地址：0x08

复位值：0x0002

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 15:13	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 SPI1_RXCRCR 的值是匹配的。 ● 1：收到的 CRC 值和 SPI1_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

23.5.4 SPI1 数据寄存器（SPI1_DR）

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
DR[15:0]															
rw															

位 15:0	DR[15:0]：数据寄存器（Data register） 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。

23.5.5 SPI1 的 CRC 多项式寄存器（SPI1_CRCPR）

偏移地址：0x10

复位值：0x0007

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

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。

23.5.6 SPI1 接收 CRC 寄存器（SPI1_RXCRCR）

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
RXCRC[15:0]															
r															

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 SPI1_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 SPI1_CRCPR 中的多项式。 ● 当数据帧格式被设置为 8 位（SPI1_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（SPI1_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。

23.5.7 SPI1 发送 CRC 寄存器（SPI1_TXCRCR）

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
TXCRC[15:0]															
r															

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 SPI1_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 SPI1_CRCPR 中的多项式。 ● 当数据帧格式被设置为 8 位（SPI1_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（SPI1_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。

24 调试支持（DBG）

24.1 概述

HK32M06x 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。调试扩展允许内核可以在取指（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 HK32M06x MCU 相连并进行调试时，将使用调试功能。

器件提供一个调试接口：串行 SW-DP 接口

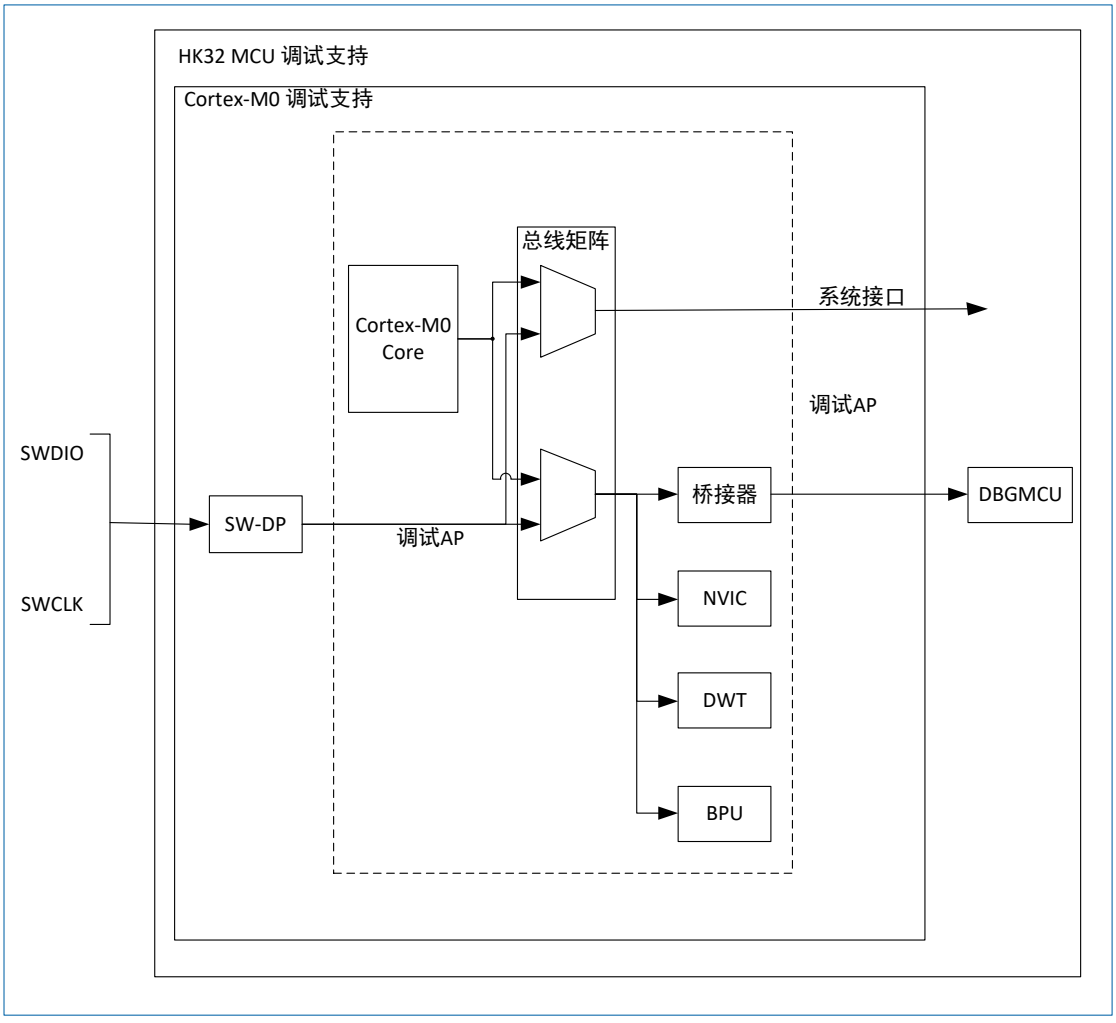

图 24-1 HK32M06x MCU 和 Cortex®-M0 级调试支持框图

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 HK32M06x 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。

24.2 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

24.3 引脚排列和调试端口引脚

HK32M06x MCU 的不同封装有不同的有效引脚数。

24.3.1 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 24-1 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	PC9
SWCLK	I	串行线时钟	PC8

24.3.2 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“6.2.2 I/O 引脚复用功能复用器和映射”。

24.3.3 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。

24.4 SWD 端口

24.4.1 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，但可以通过配置 SWCLK 频率来调整。

24.4.2 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 24-2 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 24-3 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 24-4 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。

24.4.3 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 0x0BB1 1477（相当于 Cortex®-M0）。

注意：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。

24.4.4 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），这样即使在其它操作仍未完成时，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。

24.4.5 SW-DP 寄存器描述

当 APnDP=0 时能够访问这些寄存器。

表 24-5 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。
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）。

A[3:2]	R/W	SELECT 寄存器的 CTRLSEL 位	寄存器	说明
				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。

24.4.6 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 24-6 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）

24.5 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 24-7 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）

24.6 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。

24.6.1 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。

24.7 DWT（数据观察点）

Cortex®-M0 DWT 实现提供了两个观察点寄存器组。

24.7.1 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。

24.7.2 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 记录通过条件代码和指令以及未通过条件代码的指令。

24.8 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

24.8.1 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，以在停机模式下为 FCLK 和 HCLK 提供时钟。

24.8.2 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。

24.9 DBGMCU 寄存器

基地址：0x4001 5800

空间大小：0x400

24.9.1 MCU 器件 ID 代码寄存器（DBGMCU_IDCODE）

偏移地址：0x00

复位值：0x1000 0FF0

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 ● 0x2000：版本编号 2.0
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。

24.9.2 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。

24.9.3 调试 MCUAPB1 冻结寄存器（DBGMCU_APB1_FZ）

偏移地址：0x08

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。

当 MCU 处于调试状态下时，DBG_APB1_FZ 寄存器用于配置以下 APB 外设：

- 定时器时钟计数器冻结
- 支持系统窗口看门狗和独立看门狗计数器冻结
- I2C SMBus 超时冻结

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_RTC_STOP	Res				DBG_TIM6_STOP	Res		DBG_TIM3_STOP	DBG_TIM2_STOP
			rw	rw		rw					rw			rw	rw

位 24:22	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（I2C SMBUS timeout 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	Res：保留 必须保持复位值。
位 9	DBG_RTC_STOP：内核停止时，停止调试 RTC（Debug RTC stopped when core is halted）

	● 0：即使内核停止，RTC 计数器时钟仍继续工作。 ● 1：内核停止时，RTC 计数器时钟停止。
位 8:5	Res：保留 必须保持复位值。
位 4	DBG_TIM6_STOP：内核停止时，TIM6 计数器停止（Debug TIM6 stopped when core is halted） ● 0：即使内核停止，仍然馈送 TIM6 计数器时钟。 ● 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_TIM3_STOP：内核停止时，TIM3 计数器停止（Debug TIM3 stopped when core is halted） ● 0：即使内核停止，仍然馈送 TIM3 计数器时钟。 ● 1：内核停止时，TIM3 计数器时钟停止。
位 0	DBG_TIM2_STOP：内核停止时 TIM2 计数器停止（Debug TIM2 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。

24.9.4 调试 MCUAPB2 冻结寄存器（DBGMCU_APB2_FZ）

偏移地址：0x0C

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_TIM1_STOP	Res										
				rw											

位 31:12	Res：保留 必须保持复位值。
位 11	DBG_TIM1_STOP：内核停止时，TIM1 计数器停止（Debug TIM1 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。
位 10:0	Res：保留 必须保持复位值。

25 设备电子签名（UID）

电子签名存储在 Flash 模块的系统存储区中，可以使用 SWD 或 CPU 对其进行读取。

它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它外部设备将其接口与微控制器的特性自动匹配。

25.1 唯一设备 ID 寄存器（96 位）

基地址：0x1FFF F924

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 与软件加密明文和协议结合使用时，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而言都唯一的参考号码。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连接起来。

25.1.1 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。

UID 寄存器 0 的值为 UID 的低 32 位。

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

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

位 31:0	U_ID[31:0]：芯片唯一 ID 位（Unique ID bits[31:0]）

25.1.2 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。

UID 寄存器 1 的值为 UID 的 33 至 64 位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U_ID[63:48]															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U_ID[47:32]															
r															

位 31:0	U_ID[63:32]：芯片唯一 ID 位（Unique ID bits[63:32]）

25.1.3 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]）

26 缩略语与术语

26.1 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。
rt_w	Read-only write trigger	软件可以读此位；对该位写 0 或 1 触发一个事件，但对此位没有数值没有影响。
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。

26.2 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
GPIO	General Purpose Input Output	通用输入输出
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）
MCU	Microcontroller Unit	微控制单元
OBL	Option Byte Loader	选项字节装载器
PGA	Programmable Gain Amplifier	可编程的增益放大器
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。

26.3 术语

名称	中文描述
Byte	字节，8 位数据长度。

名称	中文描述
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。

27 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。

深圳市航顺芯片技术研发有限公司

联系电话：0755-83247667

网址：www.hsxp-hk.com