HK32F031 用户手册

版本：1.4

发布日期：2023-11-20

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 HK32F031 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 HK32F031 系列芯片的应用及产品。

读者对象

本文适用于以下读者：

- 开发工程师
- 芯片测试工程师

版本说明

本文档对应的产品系列为 HK32F031 系列芯片。

修订记录

版本	日期	修订内容
1.0	2020/07/13	首次发布
1.1	2020/08/18	增加 5.23 器件电子签名章节
1.2	2020/08/21	更新公司 Logo 图标
1.3	2020/09/03	更新 SYSCFG_CFGR1 寄存器
1.4	2023/11/20	增加功能描述信息。基于 HK32F030 用户手册 V1.8 梳理本 HK32F031 用户手册：删除不支持的 SPI2，USART2，I2C2，TIM15，增加支持的 TIM2，VBAT，修改 Flash 大小为最高 32 Kbyte，并更新由以上改动影响的系统和存储器概述，RCC，PWR，DMA，NVIC，EXTI 等章节。

目 录

1 简介

本文档为 HK32F031 系列芯片的用户手册。HK32F031 系列芯片是由深圳市航顺芯片技术研发有限公司研发的主流型 MCU 芯片，包括以下子系列：

- HK32F031CxT6 (LQFP48)
  - HK32F031C6T6
  - HK32F031C4T6
- HK32F031KxT6 (LQFP32)
  - HK32F031K6T6
  - HK32F031K4T6
- HK32F031KxU6 (QFN32)
  - HK32F031K6U6
  - HK32F031K4U6
- HK32F031GxU6 (QFN28)
  - HK32F031G6U6
  - HK32F031G4U6
- HK32F031FxP6 (TSSOP20)
  - HK32F031F4P6
  - HK32F031F6P6

用户可以查看《HK32F031 数据手册》，进一步了解 HK32F031 MCU 的功能特性，如外设接口、电气特性、管脚封装等。

2 系统和存储器概述

本章介绍了 HK32F031 MCU 的系统架构、内部存储器。

2.1 系统架构

HK32F031 MCU 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
  - 专用于连接 GPIO 口的 AHB 总线

以 HK32F031C6T6 为例，HK32F031 MCU 功能框图如下所示：

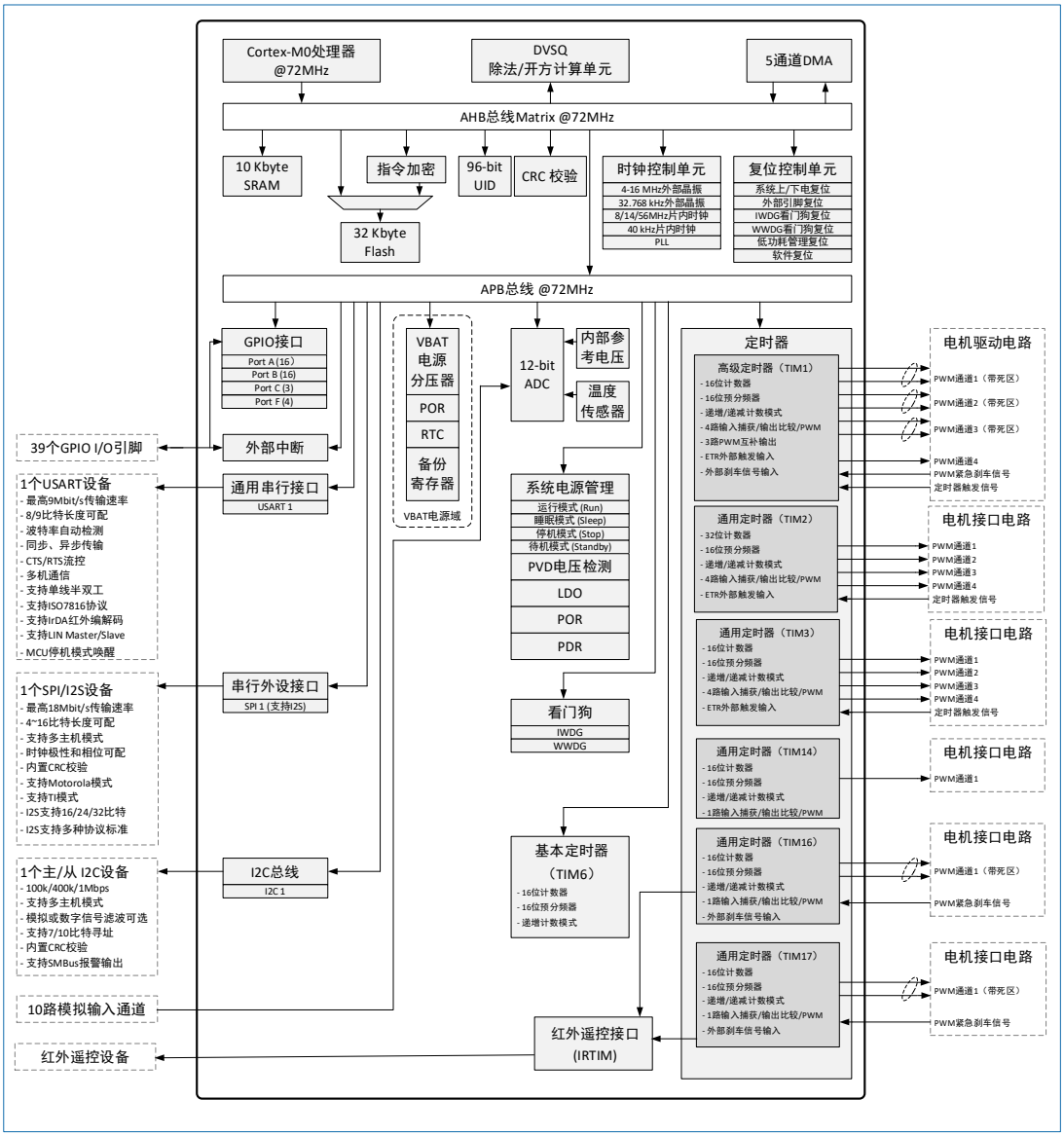


图 2-1 HK32F031C6T6 功能框图

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4Gbyte。HK32F031 MCU 支持小端模式的数据存储，即数据的低字节存放于低地址，数据的高字节存放于高地址。

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

以 HK32F031C6T6 为例，HK32F031 MCU 的存储器映射如下图所示。

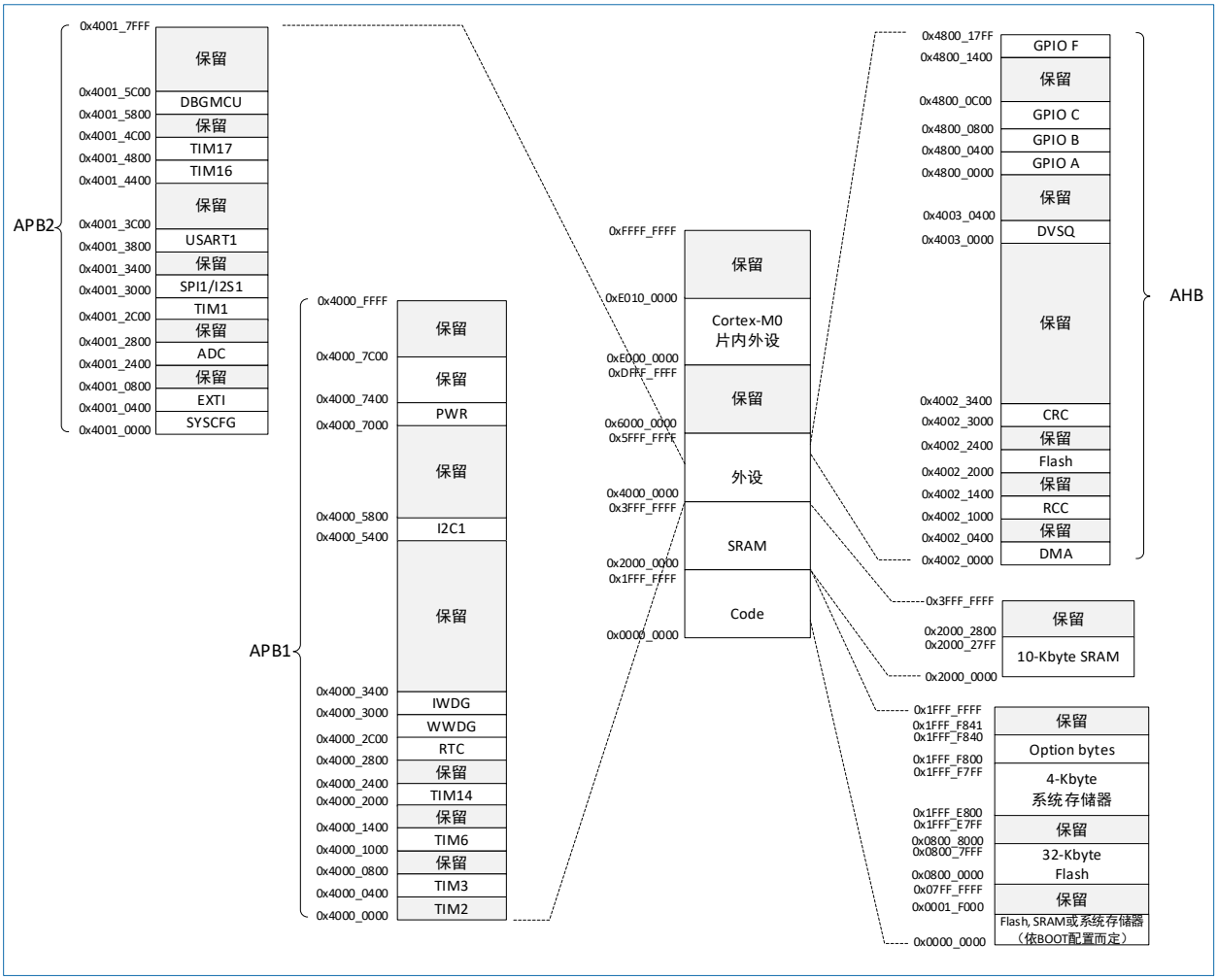


图 2-2 HK32F031R8T6 MCU 存储器映射

2.3 SRAM

HK32F031 MCU 内置 10Kbyte 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 引脚及用户选项字节中的 boot 配置位 nBOOT1 配置三种不同的启动模式，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
nBOOT1（配置位）	BOOT0（管脚）		
x	0	主 Flash	主 Flash 存储器选为启动区域
0	1	内置 SRAM	内置 SRAM 选为启动区域
1	1	系统存储器	系统存储器选为启动区域

当从复位或待机模式唤醒时，CPU 将采样启动模式的配置值，因此在待机模式下需要保持启动模式的配置值。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、SRAM 或系统存储器：

- 从主 Flash 存储器启动：主 Fl ash 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（0x800 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 0x800 0000。
- 从内置的 SRAM 启动：SRAM 映射到启动空间（0x0000 0000），但仍能够从它原有的地址空间（0x2000 0000）进行访问。
- 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（0x1FFF EC00）进行访问。

2.5 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 USART1 PA9/PA10 或 PA14/PA15 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
  - 32Kbyte 主 Flash 块，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 4K Byte。
- Flash 物理特性
  - 数据位宽：32 位
  - 页大小：1Kbyte
  - 半页大小：512byte（每页由两个半页组成）
  - 扇区大小：4Kbyte
  - 线（Line）大小：256byte（每半页由两个 Line 组成）
- Flash 访问位宽：支持半字（16 位）、字（32 位）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
- 内置数据缓冲区
- 支持半页擦除、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 块可分划分为 32 页（每页 1Kbyte）或 8 扇区（每扇区 4Kbyte）。Flash 写保护以扇区位单位。

表 3-1 Flash 结构

Flash 区	地址	大小	页号	描述
主 Flash 存储器	0x0800 0000-0x0800 03FF	1Kbyte	Page0	扇区 0
	0x0800 0400-0x0800 07FF	1Kbyte	Page1	
	0x0800 0800-0x0800 0BFF	1Kbyte	Page2	
	0x0800 0C00-0x0800 0FFF	1Kbyte	Page3	
	……	……	……	……
	0x0800 7000-0x0800 73FF	1Kbyte	Page28	

Flash 区	地址	大小	页号	描述
	0x0800 7400-0x0800 77FF	1Kbyte	Page29	扇区 7
	0x0800 7800-0x0800 7BFF	1Kbyte	Page30	
	0x0800 7C00-0x0800 7FFF	1Kbyte	Page31	
系统存储器	0x1FFF E800-0x1FFF F7FF	4Kbyte	-	-
选项字节	0x1FFF F800-0x1FFF F840	64byte	-	-

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

为了保护对 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即被激活。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）

RDP 字节值	RDP 反码值	读保护级别
0xCC	0x33	Level 2

读保护状态包括三个级别：

- Level 0：无保护

  允许对主 Flash 区域和选项字节进行读写和擦除操作。

- Level 1：读保护

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

  - 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。
  - 调试模式，从 RAM 启动或从系统存储区启动：在调试模式下或运行在 boot RAM 和 boot loader 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。设置为 Level 2 是不可恢复的操作。当试图改写 RDP 字节时，FLASH_SR 寄存器中的保护错误标志 WRPRTERR 会被置位并引发一个中断。

表 3-3 不同工作模式下保护级别和保护状态的对应关系

区域	保护级别	用户代码执行			调试/从 RAM 或从系统存储区启动		
		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). 系统存储区是唯一在任何情况下可读的区域。

(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 8 个寄存器的配置：

- 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）
- Flash 控制寄存器 2（FLASH_ECR）

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

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

3.2.4.1 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=0x45670123；
2. 向 FLASH_KEYR 寄存器写入关键字 KEY2=0xCDEF89AB。

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 一次可以编程 16 位（半字）或 32 位（字），由 FLASH_CR 和 FLASH_ECR 控制。

解锁 Flash 后：

- 当 FLASH_CR 中的 PG 位为 1 时，直接对相应的地址写一个半字（16 位），是一次编程操作。

- 当 FLASH_ECR 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。

注意：

若 Flash 处于解锁状态且 FLASH_CR 寄存器中的 PG 位为 1 时，写入的数据长度不是半字，会引起总线错误中断。

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 半字编程的流程如下图所示：

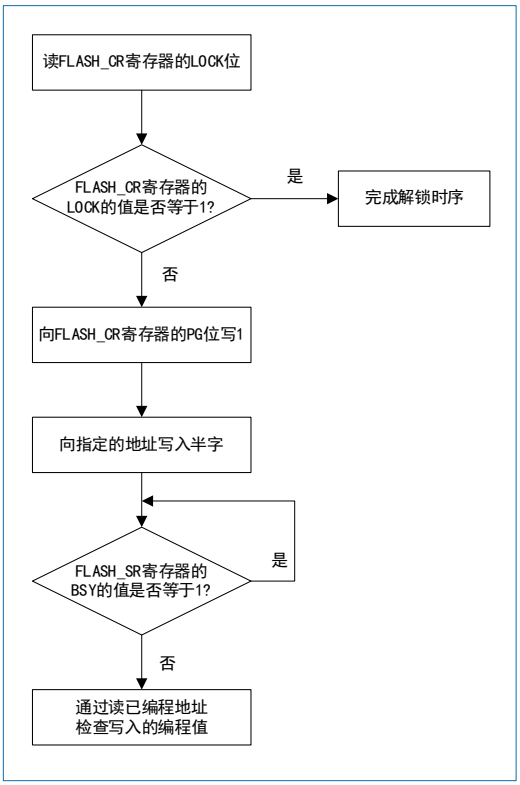


图 3-1 对 Flash 的半字编程

字编程的流程图与半字编程流程相似，区别在于将 FLASH_ECR 中的 WPG 位置'1'，和向指定的地址中写入字。

主 Flash 存储器的标准半字和字编程流程如下：

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 PG 位（半字编程）或 FLASH_ECR 的 WPG 位（字编程），以写入 Flash。
3. 根据配置，以半字或字为单位向目标地址写入数据。
4. 在完成 Flash 的数据写入后，将 FLASH_CR 寄存器中的 PG 位（半字编程时）或 FLASH_ECR 的 WPG 位（字编程时）置为 0。
5. 等待 FLASH_SR 寄存器中的 BSY 变为 0。

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

3.2.4.3 Flash 擦除

Flash 存储器可以支持页擦除、半页擦除及整片擦除。

- **页擦除**

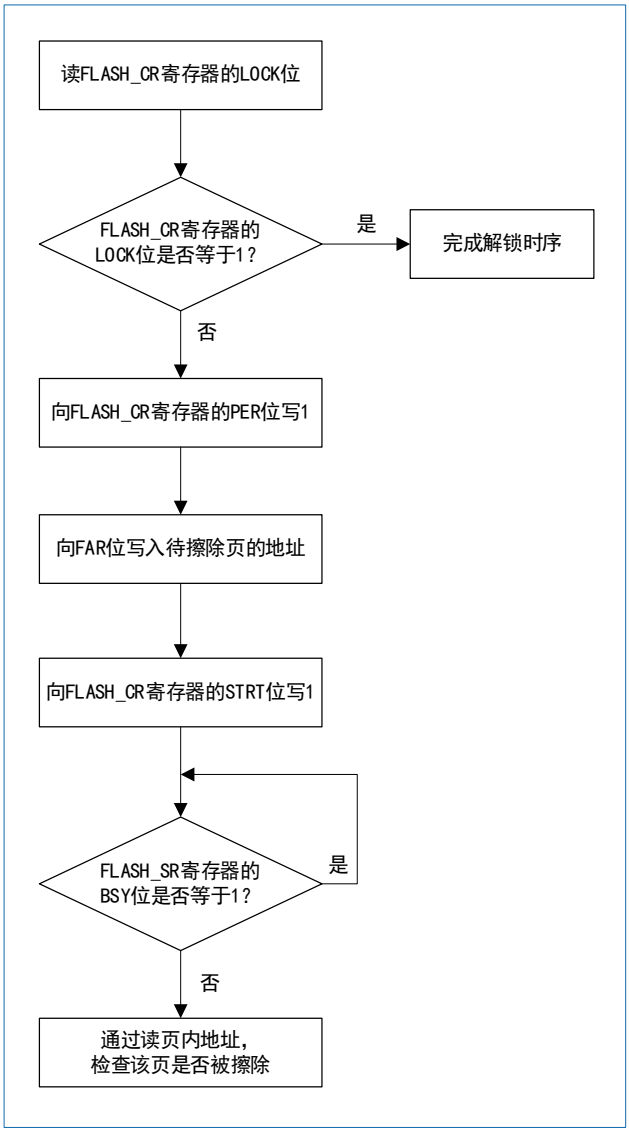


图 3-2 Flash 擦除页的流程

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 LASH_SR 寄存器的 EOP 标志（若 Flash 擦除成功会置位 EOP），然后软件清除该标志位。
7. 将 FLASH_CR 寄存器中的 PER 位置 0，以恢复默认值。

- **半页擦除**

半页擦除和页擦除的流程类似，区别在于将 FLASH_ECR 中的 HPER 位置 1，而不是把 FLASH_CR 中的 PER 置 1。

- **整片擦除**

整片擦除命令可以一次擦除整个 Flash 扇区。整片擦除的具体步骤如下：

1. 检查 FLASH_SR 寄存器的 BSY 位，以确认上次操作已经结束。
2. 将 FLASH_CR 寄存器中的 MER 位置 1，以选择整片擦除。
3. 将 FLASH_CR 寄存器中的 STRT 位置为 1，以启动擦除操作。
4. 等待 FLASH_SR 中的 BSY 位置 0，表明整片擦除操作结束。
5. 检查 FLASH_SR 寄存器的 EOP 标志位（如果 Flash 擦除成功会置位 EOP），然后软件清除该标志位。

注意：整片擦除命令对信息块不起作用。

3.2.4.4 写保护

写保护操作是以扇区（4 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。

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[1:0]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。

芯片默认不使能 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不使能。

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 也不代表改地址没有被编写过数据。

使能 Flash 数据加解密的步骤如下：

4. 向 Flash 地址 0x1FFFF820 写入 0x1357ECA8（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 u16 *)（0x1FFFF820））=0xeca8;
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
*（(volatile u16 *)（0x1FFFF822））=0x1357;
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xfffffffe ;
```

5. 向 Flash 地址 0x1FFFF824 写入 0x2468DB97（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 u16 *)（0x1FFFF824）） = 0xdb97;
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
*（(volatile u16 *)（0x1FFFF826）） = 0x2468;
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xfffffffe ;
```

6. 向 Flash 地址 0x1FFF F828~0x1FFF F82E 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 u16 *)（0x1FFFF828）） =UKEY[15:0];
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
```

```
*（(volatile u16 *)（0x1FFFF82a)）  =UKEY[31:16];
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
*（(volatile u16 *)（0x1FFFF82c)）  =UKEY[47:32];
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
*（(volatile u16 *)（0x1FFFF82e)）  =UKEY[63:48];
while（ （FLASH->SR & 0x00000001 ） == 0x00000001 ） ;
FLASH->SR = 0x20 ;
FLASH->CR &= 0xfffffffe ;
```

7. 烧录程序至 Flash。

注意：

- *向 Flash 地址 0x1FFF F828~0x1FFF F82F 写入 UKEY 后，不能读其值进行校验，因为往 FLASH 地址 0x1FFF F828~0x1FFF F82F 写入 UKEY 后再读 0x1FFF F828~0x1FFF F82F 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

地址	位[31:24]	位[23:16]	位[15:8]	位[7:0]
0x1FFF F810～0x1FFF 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	保留			
0x1FFF F834	保留			
0x1FFF F838～0x1FFF F83C	保留			
0x1FFF F840	DBG_CLK_CTL[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。 ● 当进入待机模式时，复位事件。 位[23]：保留 位[22]：RAM_PARITY_CHECK ● 0：RAM 奇偶校验使能。 ● 1：RAM 奇偶校验禁能。 位[21]：VDDA_MONITOR ● 0：VDDA 电源电压监控禁能。 ● 1：VDDA 电源电压监控使能。 位[20]：nBOOT1，连同 BOOT0 引脚，选择从主 Flash 存储器、SRAM 或系统存储器启动。 位[19]：保留 位[18]：nRST_STDBY ● 0：当进入待机模式时，产生复位。 ● 1：不产生复位。 位[17]：nRST_STOP ● 0：当进入停机模式时，产生复位。 ● 1：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。

地址	位域	选项字节描述
		● 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 4 页（Page）进行控制。WRP0作用于0 ~ 31 页；本系列芯片 Flash 最多仅 32 页，WRP1 不起作用。
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 本系列芯片 Flash 最多仅 32 页，WRP2/3 不起作用。
0x1FFF F820	31:0	ENCRY_CTL[31:0]：Flash 数据加密使能控制 ● 0x1357ECA8：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 ● 0x2468DB97：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	31:0	保留
0x1FFF F834	31:0	保留
0x1FFF F840	31:0	DBG_CLK_CTL[31:0]：开关芯片内部调试组件的时钟 ● 0x1234BCDE：关闭芯片内部调试组件的时钟。 ● 其它值：使能芯片内部调试组件的时钟。

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 保护寄存器（FLASH_WRPR）中。

每个选项字节都有其值的反码数据存放在信息块中，其目的用于校验选项字节的正确性。当选项字

节装载后，CPU 会检查选项字节的正确性。若选项字节与其反码比较不一致时，会产生选项字节校验错（OPTERR）信息。当产生 OPTERR 后，CPU 会强制将相应的选项字节值变为 0xFF。当选项字节与其反码都为 0xFF（擦除状态）时，CPU 不会比较其与反码的差异。

3.3.1 选项字节编程

选项字节区按照半字为单位进行编程。该区大小总共 18 个半字，包括：

- 1 个半字读保护
- 2 个半字用户数据
- 1 个半字硬件配置
- 4 个半字的写保护
- 2 个半字的加密使能
- 2 个半字的解密使能
- 4 个半字的用户密钥
- 2 个半字的 DEBUG 时钟配置

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. 解锁 Flash 空间
   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. 解锁 FLASH 选项字空间
   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，以选择半字方式写入。
5. 写数据（半字）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。

3.3.2 选项字节擦除过程

选项字节是按页擦除的，步骤如下：

1. 解锁 Flash 空间
   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. 解锁 FLASH 选项字空间
   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，以选择按页擦除。
5. 将待擦除的地址写入 FLASH_AR 寄存器（待擦除地址必须属于 OPT 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 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										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（适用于 0 <HCLK ≤ 24MHz）

	• 001：1 个等待周期（适用于 24MHz <HCLK ≤ 48MHz） • 010：2 个等待周期（适用于 48MHz <HCLK ≤72MHz） • 011：3 个等待周期 • 100：7 个等待周期 • 101：9 个等待周期 • 110：19 个等待周期 • 111：39 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	FKEYR[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	OPTKEY[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 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	PG
		rw	rw		rw	rw		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） 选项字节擦除时选择。
位 4	OPTPG：选项字节写入（Option byte programming） 选项字（16 位）写入时选择。
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。
位 1	PER：页擦除（Page erase） 页擦除时选择。
位 0	PG：半字写入（Half-word programming） • 0：无意义 • 1：半字编程 Flash

3.4.5 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） 当出现对写保护区域的写操作时，该位被硬件置位。该位由软件写 1 清零。
位 3	Res：保留 必须保持复位值。
位 2	PGERR：写入错误标志（Programming error） 当出现以下情况时，对被编程区进行写入操作，则该位被硬件置位。 半字编程时，被编程区的值不为’0xFFFF’。

	字编程时，被编程区的值不为’0xFFFF FFFF’。 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。

3.4.6 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 FLASH_CR.PG 位或 FLASH_ECR.WPG 位使能时，该位域为写入 Flash 的地址；或当 FLASH_CR 的 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		VDDA_MONITOR	nBOOT1	Res	nRST_STDBY	nRST_STOP	WDG_SW	Res					RDPRT[1:0]		OPTERR
		r	r	r	r	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 15:14	Res：保留 必须保持复位值。
位 13	VDDA_MONITOR：V_{DDA} 监控（V_{DDA} monitor） • 0：禁能 V_{DDA} 电源监控 • 1：使能 V_{DDA} 电源监控
位 12	nBOOT1：BOOT1 按位取反（BOOT1reverse bit） 该位与 BOOT0 输入引脚搭配使用，用于选择自举源： • 如果 BOOT0 = 0 且 nBOOT1 =X，则在主 Flash 存储器中自举 • 如果 BOOT0 = 1 且 nBOOT1 = 0，则在 SRAM 存储器中自举 • 如果 BOOT0 = 1 且 nBOOT1 = 1，则在系统存储器中自举 若要更改自举源，需要执行选项字节重载操作。如果该配置在选项字节加载期间发生不匹配，则为其加载 1。 如果器件的保护级别为级别 2，BOOT0 和 nBOOT1 将不起作用：始终强制在 Flash 程序存储器中自举。
位 11	Res：保留 必须保持复位值。
位 10	nRST_STDBY：进入待机模式是否产生复位（Enter Standby mode when generate reset） 该位属于用户选项字节。 • 0：当进入待机模式产生复位 • 1：不产生复位
位 9	nRST_STOP：进入停机模式是否产生复位（Enter Stop mode when generate reset） 该位属于用户选项字节。 • 0：当进入停机模式产生复位 • 1：不产生复位
位 8	WDG_SW：选择软件或硬件看门狗（Hardware or software watchdog selection） 该位属于用户选项字节。 • 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）
位 0	OPTERR：选项字节错误（Option byte error）

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）。 • 0：写保护激活 • 1：写保护禁止

3.4.9 Flash 控制寄存器 2（FLASH_ECR）

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
Res															

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

位 31:4	Res：保留 必须保持复位值。
位 3	WPG：字编程（Word programming）
位 2:1	Res：保留 必须保持复位值。
位 0	HPER：半页擦除（512 字节/次）（Half-page erase）

3.4.10 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） 当 Flash 选项字地址 0x1FFF F820 中的值为 0x1357ECA8 时，ENCRY_EN 的复位值为 1，否则其复位值为 0。 如果软件向 FLASH_ENCRY_CTL 寄存器写入值 0x1357ECA8 也会置位 ENCRY_EN；如果软件写入的值不是 0x1357ECA8，则把 ENCRY_EN 清零。

3.4.11 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） 当 Flash 选项字地址 0x1FFF F824 中的值为 0x2468DB97 时，DECRY_EN 的复位值为 1，否则其复位值为 0。 如果软件向 FLASH_DECRY_CTL 寄存器写入值 0x2468DB97 也会置位 DECRY_EN；如果软件写入的值不是 0x2468DB97，则把 DECRY_EN 清零。

3.4.12 Flash 密钥寄存器 1（FLASH_UKEY1）

偏移地址：0x80

复位值：0xXXXX XXXX

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]：Flash 数据加密的用户密钥低 32 位（Low 32 bits for Flash data encryption） 系统复位时，Flash 控制器自动将选项字中的密钥低 32 位载入到 FLASH_UKEY1 寄存器。 软件不可读该寄存器，读的返回值始终为 0。

3.4.13 Flash 密钥寄存器 2（FLASH_UKEY2）

偏移地址：0x84

复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UKEY[63:48]															
w															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UKEY[47:32]															
w															

位 31:0	UKEY[63:32]：Flash 数据加密的用户密钥高 32 位（High 32 bits for Flash data encryption） 系统复位时，Flash 控制器自动将选项字中的密钥高 32 位载入到 FLASH_UKEY2 寄存器。 软件不可读该寄存器，读的返回值始终为 0。 软件可以向 FLASH_UKEY2 和 FLASH_UKEY1 寄存器写入不同值来修改密钥。这适用于芯片初始化时，还未配置 Flash 选项字中的密钥，但需要把数据加密后再下载到 Flash 的应用场景。在该场景中，在芯片复位后且用户下载数据之前，先向 FLASH_UKEY2 和 FLASH_UKEY1 寄存器写入密钥，再下载数据到 Flash，最后将该密钥写入到选项字对应的地址。在芯片下一次复位时就能从选项字中载入正确的密钥，并解密存储在芯片内 Flash 中的内容。

4 CRC 计算单元

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

4.1 CRC 主要特性

- 采用的 CRC-32（与以太网标准相同）多项式 0x4C11DB7

  $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$
- 能处理 8 位、16 位和 32 位数据宽度。
- 可编程 CRC 初始值。
- 单输入/输出 32 位数据寄存器
- 输入缓冲器可避免计算期间发生总线阻塞。
- 对于 32 位数据大小，CRC 计算在 4 个 AHB 时钟周期（HCLK）内完成。
- 8 位通用寄存器（可用于临时存储）
- I/O 数据的可反转性选项 CRC 功能说明

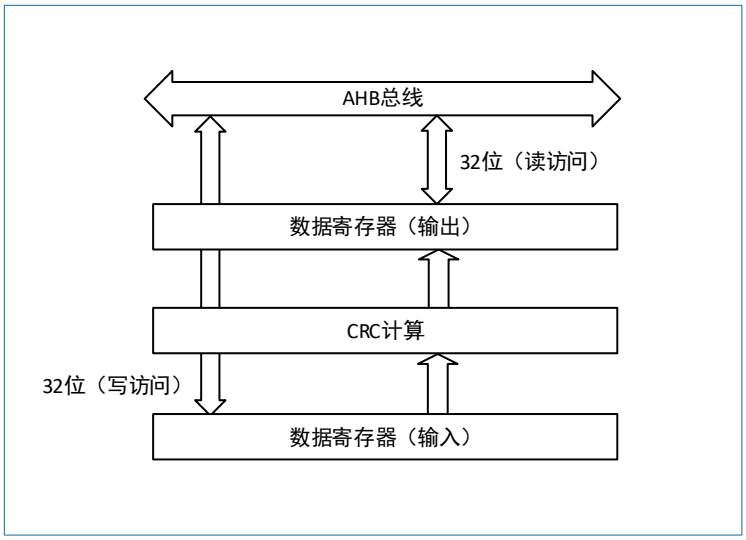


图 4-1 CRC 计算单元框图

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

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

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

计算时间取决于数据宽度：

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

输入缓冲器中可立即写入第二个数据，无需因之前的 CRC 计算而等待。

CRC 计算单元可动态调整数据大小，从而能最大程度地减少给定字节数的写访问次数。例如，对 5 个字节进行 CRC 计算时，可先写入一个字，然后写入一个字节。

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

例如，输入数据 0x1A2B 3C4D 在 CRC 计算中用作：

- 按字节执行位反转后的 0x58D4 3CB2
- 按半字执行位反转后的 0xD458B23C
- 按全字执行位反转后的 0xB23C D458

通过将 CRC_CR 寄存器中 REV_OUT 位置 1，也可以将输出数据反转。该操作按位进行：例如，输出数据 0x1122 3344 将转换为 0x22CC 4488。

配置 CRC_CR 寄存器中的 RESET 控制位可将 CRC 计算器初始化为可编程值（默认值为 0xFFFF FFFF）。

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

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

4.2 CRC 寄存器

基地址：0x4002 3000

空间大小：0x400

4.2.1 数据寄存器（CRC_DR）

偏移地址：0x00

复位值：0xFFFF FFFF

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

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

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

4.2.2 独立数据寄存器（CRC_IDR）

偏移地址：0x04

复位值：0x0000 0000

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

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

位 31:8	Res：保留

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

4.2.3 控制寄存器（CRC_CR）

偏移地址：0x08

复位值：0x0000 0000

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

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

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

4.2.4 CRC 初值寄存器（CRC_INIT）

偏移地址：0x10

复位值：0xFFFF FFFF

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

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

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

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

5 电源控制（PWR）

5.1 电源

该系列芯片的工作电压（V_{DD}）为 2.0~5.5V。通过内置的电压调节器提供所需的 1.5V 电源。当主电源 V_{DD} 掉电后，通过 VBAT 脚为实时时钟（RTC）和备份寄存器提供电源。

该系列芯片片内数字逻辑的电源由片内 LDO 提供。

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

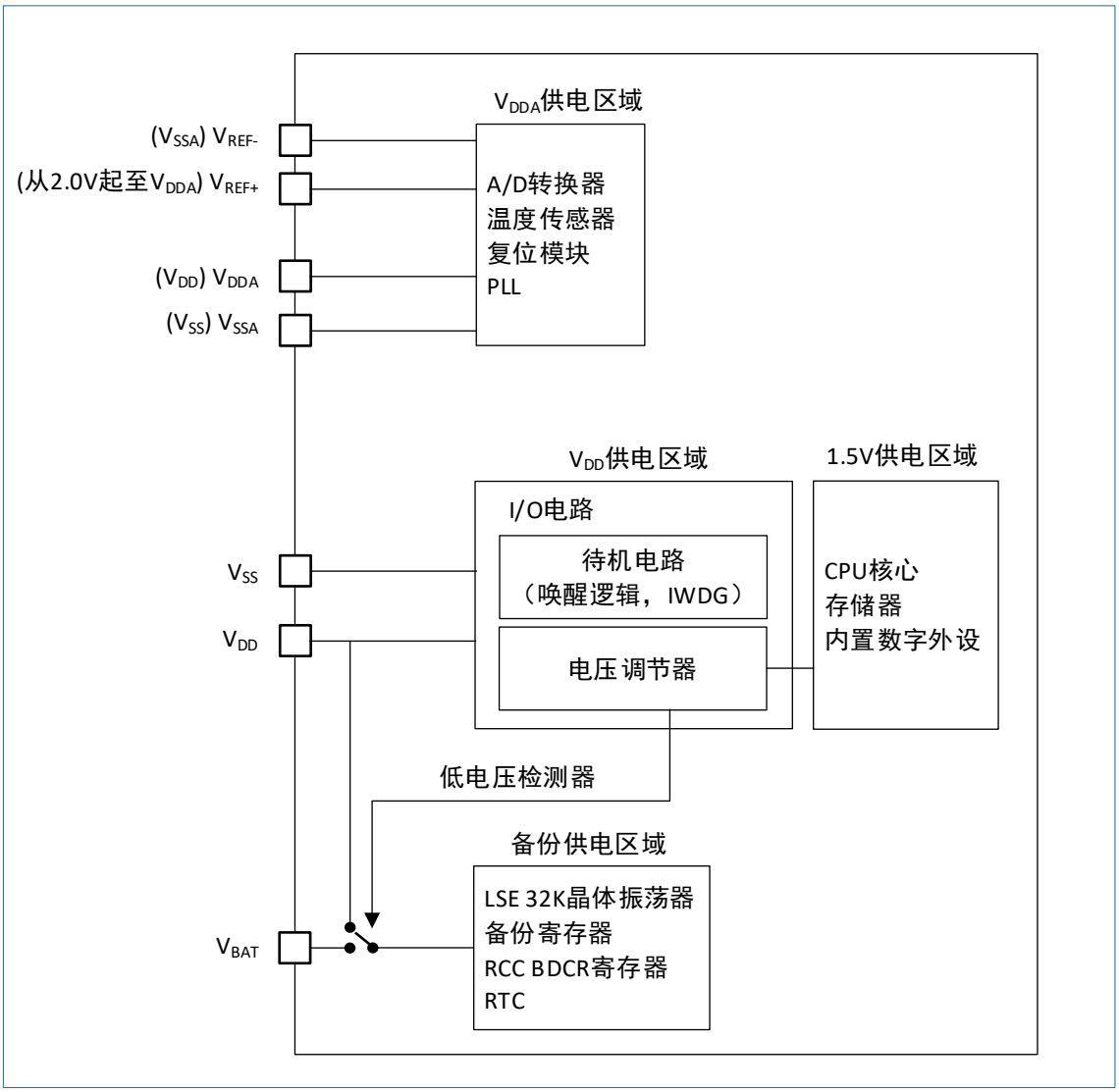


图 5-1 电源框图

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

5.1.1 独立的 A/D 转换器供电和参考电压

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

- ADC 的电源引脚为 V_{DDA}
- 独立的电源地 V_{SSA}

没有 V_{REF+}和 V_{REF-}引脚，它们在芯片内部与 ADC 的电源（V_{DDA}）和地（V_{SSA}）相连。

5.1.2 电池备份区域

使用电池或其他电源连接到 V_{BAT} 脚上，当 V_{DD} 断电时，可以保存备份寄存器的内容和保持 RTC 的功能。

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

警告：

- *在 V_{DD} 上升阶段（$t_{RSTTEMPO}$）或者探测到掉电复位（PDR）之后，V_{BAT} 和 V_{DD} 之间的电源开关仍会保持 连接在 V_{BAT}。*
- *在 V_{DD} 上升阶段，如果 V_{DD} 在小于 $t_{RSTTEMPO}$ 的时间内达到稳定状态（关于 $t_{RSTTEMPO}$ 数值可参考数据手册中的相关部分），且 $V_{DD} > V_{BAT} + 0.6$ V 时，电流可能通过 V_{DD} 和 V_{BAT} 之间的内部二极管注入到 V_{BAT}。*
- *如果与 V_{BAT} 连接的电源或者电池不能承受这样的注入电流，强烈建议在外部 V_{BAT} 和电源之间连接一个低压降二极管。*

如果在应用中没有外部电池，建议 V_{BAT} 在外部连接到 V_{DD} 并连接一个 100 nF 的陶瓷滤波电容。

当备份区域由 V_{DD}（内部模拟开关连到 V_{DD}）供电时，下述功能可用：

- PC14 和 PC15 可以用于 GPIO 或 LSE 引脚。
- PC13 可以作为通用 I/O 口、TAMPER 引脚、RTC 校准时钟、RTC 闹钟或秒输出（参见“21.3.17 备份寄存器（RTC_BKPxR）”）。

注意：

因为模拟开关只能通过少量的电流（3 mA），在输出模式下使用 PC13 至 PC15 的 I/O 口功能是有限制的：速度必须限制在 2 MHz 以下，最大负载为 30 pF，而且这些 I/O 口绝对不能当作电流源（如驱动 LED）。

当备份区域由 V_{BAT} 供电时（V_{DD} 消失后，模拟开关连到 V_{BAT}），可以使用下述功能：

- PC14 和 PC15 只能用于 LSE 引脚。
- PC13 可以作为 TAMPER 引脚、RTC 闹钟或秒输出（参见“21.2.12 RTC 精密数字校准”）。

5.1.3 电压调节器

复位后，电压调节器总是使能的。根据应用方式，调节器以如下几种不同的模式工作：

- 运行（Run）模式：调节器以正常功耗模式提供 1.5V 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 1.5V 电源，以保存寄存器和 SRAM 的内容。
- 待机（Standby）模式：调节器停止供电。除了备用电路和备份域外，寄存器和 SRAM 的内容全部丢失。

5.2 电源管理器

5.2.1 上电/掉电复位（POR/PDR）

该系列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 2.0V 时，系统即能正常工作。

当 V_{DD}/V_{DDA} 低于指定的限位电压 V_{POR}/V_{PDR} 时，系统保持为复位状态，而无需外部复位电路。关于上电复位和掉电复位的更多细节，请参考数据手册的电气特性部分。

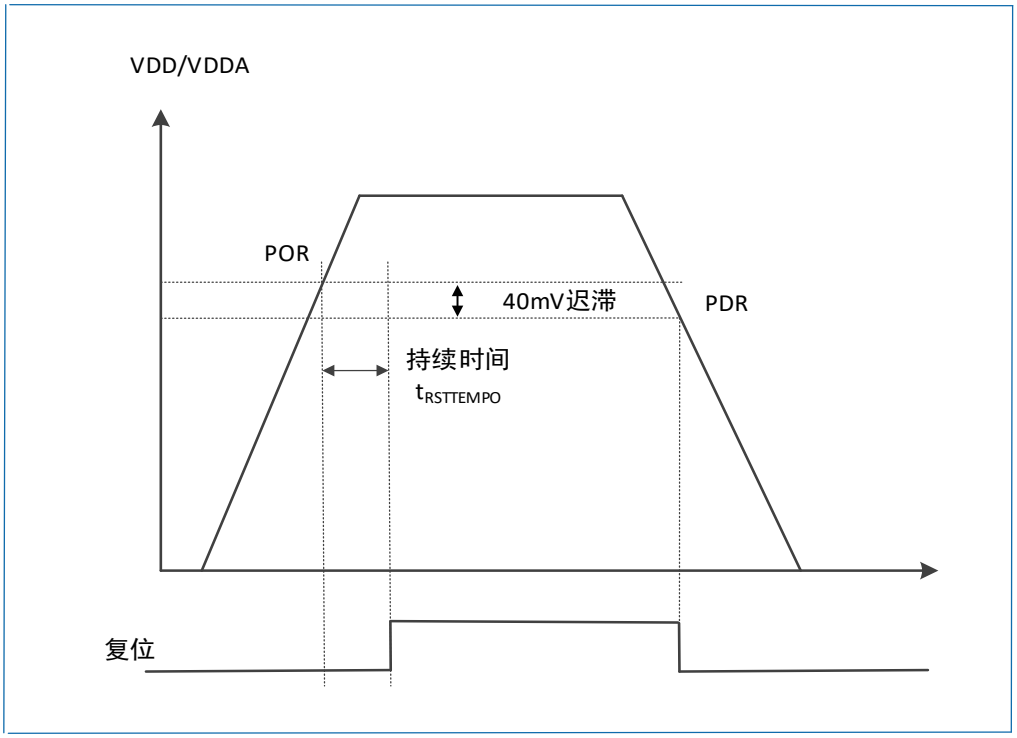


图 5-2 上电复位和掉电复位的波形图

5.2.2 可编程电压监测器（PVD）

可以使用可编程电压监测器（Programmable Voltage Detector，PVD）监测 V_{DD}/V_{DDA} 的供电。PVD 功能是通过将 V_{DD} 电压与电源控制寄存器（PWR_CR）中的 PLS[2:0]位选择的监控电压阈值进行比较来监控电源。

通过设置（PWR_CR）的 PVDE 位以使能 PVD。

电源控制/状态寄存器（PWR_CSR）中的 PVDO 标志用于表示 V_{DD}/V_{DDA} 高于或低于 PVD 阈值。该事件在内部连接到外部中断的第 16 线（EXTI 线 16），如果在外部中断寄存器使能了该中断，则会产生中断。当 V_{DD}/V_{DDA} 下降到 PVD 阈值以下和（或）当 V_{DD}/V_{DDA} 上升到 PVD 阈值以上时，根据 EXTI 线 16 上升沿/下降沿触发配置，则会产生 PVD 输出中断。例如，这一特性可用于服务程序执行紧急关闭任务。

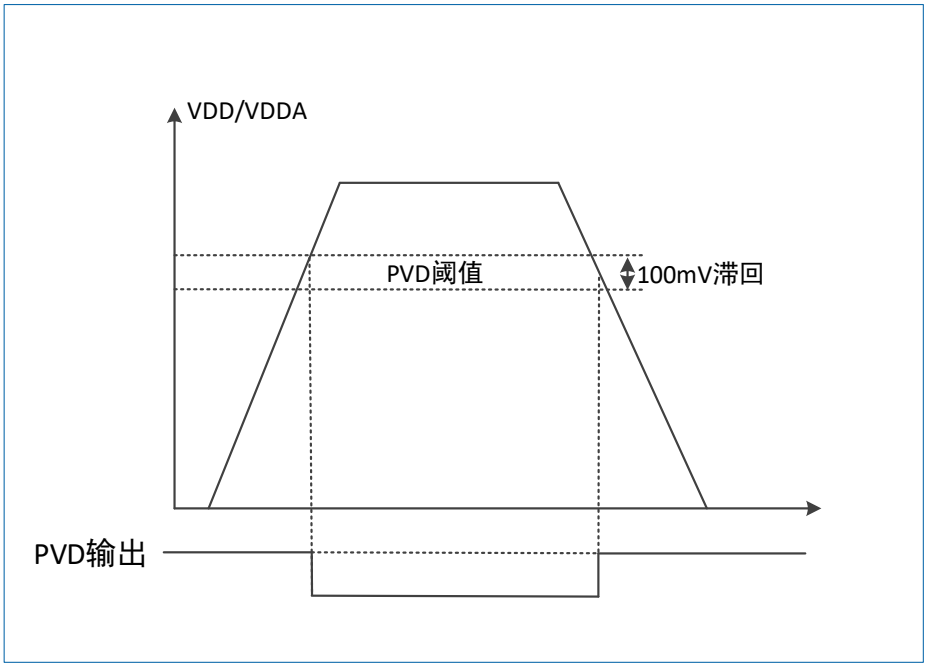


图 5-3 PVD 的门限

5.3 低功耗模式

在系统或电源复位以后，微控制器处于运行状态。当 CPU 不需继续运行时，可以利用多种低功耗模式来节省功耗，例如等待某个外部事件。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。

该系列芯片有如下几种低功耗模式：

- 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：所有的时钟都已停止。
- 待机（Standby）模式：1.5V 电源关闭。

在运行模式下，可以通过以下任一方式降低功耗：

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

表 5-1 低功耗模式的进入/唤醒条件

工作模式	进入条件	唤醒条件	内部核电源时钟状态	V_{DD} 主区域时钟状态	电压调节器状态
睡眠模式(Sleep)	1. 设置 PWR_CR:LPDS = 0； 2. 软件执行 WFI/WFE 指令进入。	由任何一个普通 IRQ 中断事件唤醒，包括 SystemTicker。	CPU 时钟关闭，对其他时钟和 ADC 时钟无影响	开启	开启
停机模式(Stop)	1. 设置 PWR_CR:LPDS = 0； 2. 设置 CM0 系统控制寄存器的 SLEEPDEEP 位，软件执行 WFI/WFE 指令进入。	支持任何一个 EXTI 外部中断线唤醒。	所有时钟停止	HSI 和 HSE 关闭	开启或者处于低功耗模式（在 PWR_CR 中设置）
待机模式(Standby)	1. 设置 PWR_CR:LPDS =0； 2. 设置 PWR_CR:PDDS =1； 3. 设置 Cortex-M0 系统控制寄存器的 SLEEPDEEP 位，软件执行 WFI/WFE 指令进入。	3 个可配置极性的外部唤醒引脚以及 RTC 报警唤醒。	所有时钟停止	HSI 和 HSE 关闭	关闭

5.3.1 降低系统时钟

在运行模式下，通过对预分频寄存器进行编程，可以降低任意一个系统时钟（SYSCLK、HCLK、PCLK）的速度。进入睡眠模式前，也可以利用预分频器来降低外设的时钟频率。参见“6.3.2 时钟配置寄存器（RCC_CFGR）”。

5.3.2 外部时钟的控制

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

通过设置 AHB 外设时钟使能寄存器（6.3.6 AHB 外部时钟使能寄存器（RCC_AHBENR））、APB2 外设时钟使能寄存器（6.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR））和 APB1 外设时钟使能寄存器（6.3.8 APB1 外设时钟使能寄存器（RCC_APB1ENR））来开关各个外设模块的时钟。

5.3.3 睡眠（Sleep）模式

5.3.3.1 进入睡眠模式

通过执行 WFI 或 WFE 指令进入睡眠状态。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，更多的细节参见表 5-1 和表 5-3。

5.3.3.2 退出睡眠模式

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

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

- 在外设控制寄存器中使能一个中断，而不是在 NVIC 中使能，并且在 Cortex-M0 系统控制寄存器中使能 SEVONPEND 位。当 MCU 从 WFE 中唤醒后，外设的中断挂起位和外设的 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）必须被清除。
- 配置一个外部或内部的 EXTI 线为事件模式。当 MCU 从 WFE 中唤醒后，因为与事件线对应的挂起位未被设置，不必清除外设的中断挂起位或 NVIC 中断请求（IRQ）通道挂起位。

该模式唤醒所需的时间最短，因为中断的进入或退出没有消耗时间。关于如何退出睡眠模式，更多的细节参见表 5-2 和表 5-3。

表 5-2 SLEEP-NOW 模式

SLEEP-NOW 模式	说明
进入条件	在以下条件下，执行 WFI（等待中断）或 WFE（等待事件）指令： ● SLEEPDEEP = 0 ● SLEEPONEXIT = 0 参考 Cortex-M0 系统控制寄存器。
退出条件	● 如果执行 WFI 进入睡眠模式： 中断：参见“10.3 中断和异常向量”。 ● 如果执行 WFE 进入睡眠模式： 唤醒事件：参见“11.3 唤醒事件管理”。
唤醒延时	无

表 5-3 SLEEP-ON-EXIT 模式

SLEEP-ON-EXIT 模式	说明
进入条件	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。

SLEEP-ON-EXIT 模式	说明
退出条件	中断：参见“10.3 中断和异常向量”。
唤醒延时	无

5.3.4 停机（Stop）模式

停机模式是在 Cortex®-M0 的深睡眠模式基础上结合了外设的时钟控制机制。在停机模式下，电压调节器可运行在正常或低功耗模式。此时，在 1.5V 供电区域的所有时钟都停止，PLL、HSI 和 HSE RC 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，所有的 I/O 引脚都保持它们在运行模式时的状态。

5.3.4.1 进入停机模式

关于如何进入停机模式，参见表 5-4。

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。可以通过设置独立的控制位，可选择以下功能：

- 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“19.2 IWDG 功能描述”。
- 实时时钟（RTC）：通过备份域控制寄存器（RCC_BDCR）的 RTCEN 位来设置。
- 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。
- 外部 32.768kHz 振荡器（LSE）：通过备份域控制寄存器（RCC_BDCR）的 LSEON 位设置。

在停机模式下，ADC 也会产生功耗，除非在进入停机模式前配置 ADC 使能位将其禁止。

5.3.4.2 退出停机模式

关于如何退出停机模式，可参见表 5-4。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。

表 5-4 停机模式

停机模式	说明
进入	在以下条件下执行 WFI（等待中断）或 WFE（等待事件）指令： 1. 设置 Cortex®-M0 系统控制寄存器中的 SLEEPDEEP 位 2. 清除电源控制寄存器（PWR_CR）中的 PDDS 位 3. 通过设置 PWR_CR 中 LPDS 位选择电压调节器的模式 说明：为了进入停机模式，所有的外部中断的请求位（挂起寄存器（EXTI_PR）和 RTC 的闹钟标志都必须被清除，否则停机模式的进入流程将会被跳过，程序继续运行。
退出	● 如果执行 WFI 进入停机模式： 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）。参见“10.3 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。参见“11.3 唤醒事件管理。”

停机模式	说明
唤醒延时	HSI RC 唤醒时间+电压调节器从低功耗唤醒的时间。

5.3.5 待机（Standby）模式

该模式是在 Cortex-M0 深睡眠模式时关闭电压调节器。整个 1.5V 供电区域被断电。PLL、HSI 和 HSE 振荡器也被断电。SRAM 和寄存器内容丢失。只有备份的寄存器和待机电路维持供电（参见图 5-1 电源框图

）。

5.3.5.1 进入待机模式

关于如何进入待机模式，参见表 5-5 待机模式。可以通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗（IWDG）：可通过写入看门狗的关键字寄存器或硬件选择来启动 IWDG。独立看门狗一旦启动，除了系统复位，它不能被停止。参见“19.2 IWDG 功能描述”。
- 实时时钟（RTC）：通过备份域控制寄存器（RCC_BDCR）的 RTCEN 位来设置。
- 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。
- 外部 32.768kHz 振荡器（LSE）：通过备用区域控制寄存器（RCC_BDCR）的 LSEON 位设置。

5.3.5.2 退出待机模式

当一个外部复位（NRST 引脚）、IWDG 复位、WKUP 引脚上的上升沿或 RTC 闹钟事件的上升沿发生时（参见图 21-1），MCU 从待机模式退出。从待机唤醒后，除了电源控制/状态寄存器（PWR_CSR）外，所有寄存器被复位。

从待机模式唤醒后，程序以复位后重启一样的方式开始执行（采样启动模式引脚、读取复位向量等）。电源控制/状态寄存器（PWR_CSR）将会指示内核由待机状态退出。

关于如何退出待机模式，参见下表。

表 5-5 待机模式

待机模式	说明
进入	在以下条件下，执行 WFI（等待中断）或 WFE（等待事件）指令： 1. 设置 Cortex®-M0 系统控制寄存器中的 SLEEPDEEP 位。 2. 设置电源控制寄存器（PWR_CR）中的 PDDS 位。 3. 清除电源控制/状态寄存器（PWR_CSR）中的 WUF 位。
退出	WKUP 引脚的上升沿、RTC 闹钟事件的上升沿、NRST 引脚上外部复位或 IWDG 复位。
唤醒延时	复位阶段时电压调节器的启动。

5.3.5.3 待机模式下的输入/输出端口状态

在待机模式下，所有的 I/O 引脚处于高阻态，除了以下的引脚：

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

5.3.6 调试模式

默认情况下，如果调试 MCU 时，使 MCU 进入停机、或待机模式，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。更多的细节参见“26.8.1 对低功耗模式的调试支持”。

5.3.7 低功耗模式下的自动唤醒（AWU）

RTC 可以在不依赖外部中断的情况下唤醒低功耗模式下的 MCU（自动唤醒模式）。RTC 提供一个可编程的时间基准，用于周期性从停机或待机模式下唤醒 MCU。通过对备份区域控制寄存器（RCC_BDCR）的 RTC_SEL[1:0]位的编程，三个 RTC 时钟源中的二个时钟源可以选中以实现此功能。

- 低功耗 32.768kHz 外部晶振（LSE）
  该时钟源提供了一个低功耗且精确的时间基准。（在典型情形下消耗小于 1μA）
- 低功耗内部 RC 振荡器（LSI RC）
  使用该时钟源，节省了一个 32.768kHz 晶振的成本。但是 RC 振荡器将增加少量电源消耗。

为了用 RTC 闹钟事件将系统从停机模式下唤醒，必须进行如下操作：

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

如果要从待机模式中唤醒，不必配置外部中断线 17。

5.4 PWR 寄存器

基地址：0x4000 7000

空间大小：0x400

可以按半字（16 位）或字（32 位）的方式操作 PWR 寄存器。

5.4.1 电源控制寄存器（PWR_CR）

偏移地址：0x00

复位值：0x0000 0000

从待机模式唤醒时，该寄存器被复位。

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							DBP	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
							rw	rw			rw	rc_w1	rc_w1	rw	rw

位 31:9	Res：保留 必须保持复位值。
位 8	DBP：备份区域的写保护（Disable RTC domain write protection） 在复位后，RTC 和备份寄存器处于被保护状态以防意外写入。设置该位，则允许写入这些寄存器。 • 0：禁止写入 RTC 和备份寄存器

	• 1：允许写入 RTC 和备份寄存器 *注意：如果 RTC 的时钟是 HSE/128，该位必须保持为'1'。*
位 7:5	PLS[2:0]：PVD 电平选择（PVD level selection） 该位域用于选择电源电压监测器的电压阈值。 • 000：2.2V • 001：2.3V • 010：2.4V • 011：2.5V • 100：2.6V • 101：2.7V • 110：2.8V • 111：2.9V *注意：详细说明参见数据手册中的电气特性部分。*
位 4	PVDE：电源电压监测器（PVD）使能（Power voltage detector enable） • 0：禁止 PVD • 1：开启 PVD
位 3	CSBF：清除待机位（Clear standby flag） 该位始终读为 0。 • 0：不起作用 • 1：清除 SBF 待机位（写）
位 2	CWUF：清除唤醒位（Clear wakeup flag） 该位始终读为 0。 • 0：不起作用 • 1：2 个系统时钟周期后，清除 WUF 唤醒位（写）。
位 1	PDDS：掉电深睡眠（Power down deepsleep） 该位与 LPDS 位协同操作。 • 0：当 CPU 进入深睡眠时进入停机模式，调压器的状态由 LPDS 位控制。 • 1：CPU 进入深睡眠时进入待机模式。
位 0	LPDS：电压调节器状态（Low-power deepsleep） 该位用软件设置或清除。 • 0：在停机模式下，电压调节器开启（处于正常功耗模式）。 • 1：在停机模式下，电压调节器处于低功耗模式。

5.4.2 电源控制/状态寄存器（PWR_CSR）

偏移地址：0x04

复位值：0x0000 0000 （从待机模式唤醒时不被清除）

从待机模式唤醒时，该寄存器不会被复位。与标准的 APB 读相比，读此寄存器需要额外的 APB 周期。

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						EWUP2	EWUP1	Res					PVDO	SBF	WUF
						rw	rw						r	r	r

位 31:10	Res：保留 必须保持复位值。
位 9	EWUP2：使能 WKUP2 引脚（Enable WKUP2 pin） 外部唤醒 pin2 使能，可唤醒待机模式。
位 8	EWUP1：使能 WKUP1 引脚（Enable WKUP1 pin） • 0：WKUP 引脚为通用 I/O WKUP 引脚上的事件不能将 CPU 从待机模式唤醒 • 1：WKUP 引脚用于将 CPU 从待机模式唤醒 WKUP 引脚被强制置为输入下拉的配置（WKUP 引脚的上升沿将系统从待机模式唤醒）。也可唤醒待机模式。
位 7:3	Res：保留 必须保持复位值。
位 2	PVDO：PVD 输出（PVD output） 当 PVD 被 PVDE 位使能后，该位才有效。 • 0：V_{DD}/V_{DDA} 高于由 PLS[2:0]选定的 PVD 阈值 • 1：V_{DD}/V_{DDA} 低于由 PLS[2:0]选定的 PVD 阈值 *注意：在待机模式下 PVD 被停止。因此，待机模式后或复位后，直到设置 PVDE 位之前，该位为 0。*
位 1	SBF：待机标志（Standby flag） 该位由硬件设置，并且只能由上电/掉电复位（POR/PDR）或设置电源控制寄存器（PWR_CR）的 CSBF 位清除。 • 0：系统不在待机模式 • 1：系统进入待机模式
位 0	WUF：唤醒标志（Wakeup flag） 该位由硬件设置，并只能由上电/掉电复位（POR/PDR）或设置电源控制寄存器（PWR_CR）的 CWUF 位清除。 • 0：没有发生唤醒事件 • 1：在 WKUP 引脚上发生唤醒事件或出现 RTC 闹钟事件 *注意：当 WKUP 引脚已经是高电平时，在（通过设置 EWUP 位）使能 WKUP 引脚时，会检测到一个额外的事件。*

5.4.3 唤醒引脚极性选择寄存器（PWR_WUP_POL）

偏移地址：0x34

复位值：0x0000 0000

复位时，该寄存器恢复默认值。

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														WUPOL2	WUPOL1
														rw	rw

位 31:2	Res：保留 必须保持复位值。
位 1	WUPOL2：Wakeup 引脚 2 极性选择（Wakeup pin 2 polarity selection） • WUPOL2 = 0，WKUP2 为内部下拉，上升沿唤醒。 • WUPOL2 = 1，WKUP2 为内部上拉，下降沿唤醒。
位 0	WUPOL1：Wakeup 引脚 1 极性选择（Wakeup pin 1 polarity selection） • WUPOL1 = 0，WKUP1 为内部下拉，上升沿唤醒。 • WUPOL1 = 1，WKUP1 为内部上拉，下降沿唤醒。

5.4.4 PDR 控制寄存器（PWR_PORPDR_CFG）

偏移地址：0x3C

复位值：0x0000 0000

复位时，该寄存器恢复默认值。

通过设置此寄存器， 将在待机模式下关闭掉电检测 PDR，以进一步降低待机功耗。

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[7:0]								Res							PDR_PD
w															rw

位 31:16	Res：保留 必须保持复位值。
位 15:8	KEY[7:0]：用于解锁 PDR_PD 位（Key to unlock PDR_PD） 按照以下序列写入以解锁 PDR_PD 位。 • 写入：0x0000 • 写入：0x5500 • 写入：0xAA00 • 写入：0x5A00 • 写入：0xA500

	• 写入：0xC800 • 写入：0x8C00 • 写入：0x6900 • 写入：0x9601
位 7:1	Res：保留 必须保持复位值。
位 0	PDR_PD：在待机（Standby）模式下关闭 PDR（以降低功耗）（PDR power-down in Standby mode） • 0：打开 PDR 对该位写一个 0x0，就会把 PDR 打开。 • 1：关闭 PDR 关闭 PDR 后，约节省 200nA 电流。

5.4.5 运行模式下内部 LDO 控制寄存器（PWR_LDO）

偏移地址：0x40

复位值：0x0000 2538

复位时，该寄存器恢复默认值。通过设置此寄存器，将在运行模式下控制内核电压。

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												SW_V15[3:0]			
												rw			

位 31:4	Res：保留 必须保持复位值。
位 3:0	SW_V15：LDO 电压设置 • 0000：1.212v（不推荐设置） • 0001：1.251v • 0010：1.291v • 0011：1.330v • 0100：1.369v • 0101：1.409v • 0110：1.449v • 0111：1.488v（默认） • 1000：1.530v • 1001：1.568v • 1010：1.606v • 1011：1.643v • 1100：1.682v

	• 1101：1.718v • 1110：1.754v • 1111：1.787v（不推荐设置） *注意：* *适当降低电压可以降低动态功耗。但可能导致程序跑飞。* *适当升高电压可以提高最高速度（超频）但会导致发热和寿命减少。*

5.4.6 停机模式下内部 LDO 控制寄存器（PWR_LDO_LOW）

偏移地址：0x44

复位值：0x0000 20B8

复位时，该寄存器恢复默认值。通过设置此寄存器，将在停机模式下控制内核电压。

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												SW_V15_LOW[3:0]			
												rw			

位 31:4	Res：保留 必须保持复位值。
位 3:0	SW_V15_LOW：进入停机模式后 LDO 电压设置 • 0000：1.212v（不推荐设置） • 0001：1.251v • 0010：1.291v • 0011：1.330v • 0100：1.369v • 0101：1.409v • 0110：1.449v • 0111：1.488v（默认） • 1000：1.530v • 1001：1.568v • 1010：1.606v • 1011：1.643v • 1100：1.682v • 1101：1.718v • 1110：1.754v • 1111：1.787v（不推荐设置） *注意：* *适当降低电压可以降低停机模式下的待机功耗，但电压过低可能无法唤醒。*

5.4.7 待机模式下 I2C 唤醒地址寄存器（PWR_I2CWUP_OA_CFG）

偏移地址：0x60

复位值：0x0000 0000

复位时，该寄存器恢复默认值。此寄存器用于设置 I2C 作为从机时的地址，在待机模式下可以通过响应总线地址实现唤醒。停机模式下请勿使能此功能。

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
OAEN	Res				MSK[2:0]			OA[7:1]							Res
w					w			w							

位 31:16	Res：保留 必须保持复位值。
位 15	OAEN：启用 I2C 帧头检测，当处于待机模式下时，将检测 I2C START 条件，匹配地址是否和 OA[7:1]一致。
位 14:11	Res：保留 必须保持复位值。
位 10:8	MSK：屏蔽 OA[7:1] • 000：不屏蔽，收到地址与 OA[7:1]匹配时即唤醒。 • 001：屏蔽 OA[1]，收到地址与 OA[7:2]匹配时即唤醒。 • 010：屏蔽 OA[2:1]，收到地址与 OA[7:3]匹配时即唤醒。 • 011：屏蔽 OA[3:1]，收到地址与 OA[7:4]匹配时即唤醒。 • 100：屏蔽 OA[4:1]，收到地址与 OA[7:5]匹配时即唤醒。 • 101：屏蔽 OA[5:1]，收到地址与 OA[7:6]匹配时即唤醒。 • 110：屏蔽 OA[6:1]，收到地址与 OA[7]匹配时即唤醒。 • 111：屏蔽 OA[7:1]，收到任何地址都会唤醒，除非 I2C 协议规定的一些保留地址。
位 7:1	OA[7:1]：响应地址设置
位 0	Res：保留 必须保持复位值。

5.4.8 待机模式下 I2C 地址匹配寄存器（PWR_I2CWUP_ADDCODE）

偏移地址：0x64

复位值：0x0000 20B8

复位时，该寄存器恢复默认值。通过此寄存器，可以在唤醒后，查看是否在待机模式下有 I2C 地址匹配事件。以及具体匹配的地址。

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
OAEN_R	ADDRF	Res						ADDCODE[7:1]							DIR
r	r							r							r

位 31:16	Res：保留 必须保持复位值。
位 15	OAEN_R：检查 PWR_I2CWUP_OA_CFG.OAEN 的值。
位 14	ADDRF：若地址匹配，则此位置 1 清除 PWR_I2CWUP_OA_CFG.OAEN 可清除 ADDRF。
位 13:8	Res：保留 必须保持复位值。
位 7:1	ADDCODE[7:1]：记录地址匹配事件时，匹配到的地址。
位 0	DIR：地址匹配时的方向 • 0：接收数据，SLAVE 接收模式。 • 1：发送数据，SLAVE 发送模式。

6 复位和时钟控制（RCC）

6.1 复位

该系列芯片有三种复位方式：系统复位、电源复位和备份域复位。

6.1.1 系统复位

系统复位将复位除时钟控制/状态寄存器（RCC_CSR）中的复位标志和备份域中的寄存器以外的所有寄存器为它们的复位值。

当以下任一事件发生时，将产生系统复位：

- NRST 引脚上的低电平（外部复位）
- 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）
- 软件复位（SW 复位）
- 电源复位
- 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。

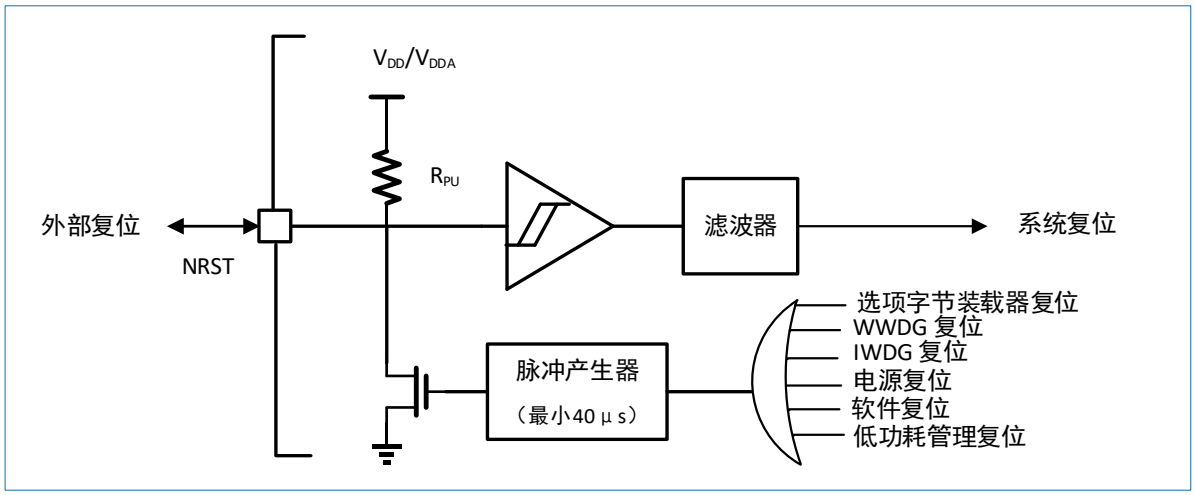


图 6-1 复位电路

上图中的复位源将最终作用于 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。

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

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

在以下两种情况下，可产生低功耗管理复位：

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

- 在进入停止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。

6.1.2 电源复位

当以下任一事件发生时，将产生电源复位：

- 上电/掉电复位（POR/PDR）
- 从待机模式中退出

电源复位将复位除了备份域外的所有寄存器。

6.1.3 备份域复位

备份域拥有两个专门的复位，它们只影响备份域。

当以下任一事件发生时，将产生备份域复位：

- 软件复位，由备份域控制寄存器（RCC_BDCR）的 BDRST 位触发。
- 在 V_{DD} 和 V_{BAT} 两者掉电后，V_{DD} 和（或）V_{BAT} 再上电时将触发备份域复位。

6.2 时钟

支持多种时钟源驱动系统时钟：

- 内部高速 56MHz RC 振荡器时钟（HSI）
- 内部高速 14MHz RC 振荡器（HSI14）
- 内部高速 8MHz RC 振荡器（HSI）
- 内部低速 40kHz RC 振荡器（LSI），用于驱动独立的看门狗和可选的 RTC 从停止/待机模式自动唤醒
- 外部低速 32.768kHz 晶体（LSE 晶体），可选驱动实时时钟（RTCCLK）
- 外部高速 4-16 MHz 振荡器时钟（HSE）
- PLL 时钟

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 72MHz。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：

- 选项字节装载器时钟由 HSI 时钟驱动。
- ADC 时钟由以下任一时钟得到（由软件选择）：
  - 专门的 HSI14 时钟，运行在最大的采样率。
  - APB（PCLK）时钟的 2/4 分频。
- USART1 的时钟为以下的时钟源之一（由软件选择）：

- 系统时钟
- HSI 时钟
- LSE 时钟
- APB 时钟（PCLK）

- I2C1 的时钟为以下的时钟源之一（由软件选择）：
  - 系统时钟
  - HSI 时钟
  - PCLK

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

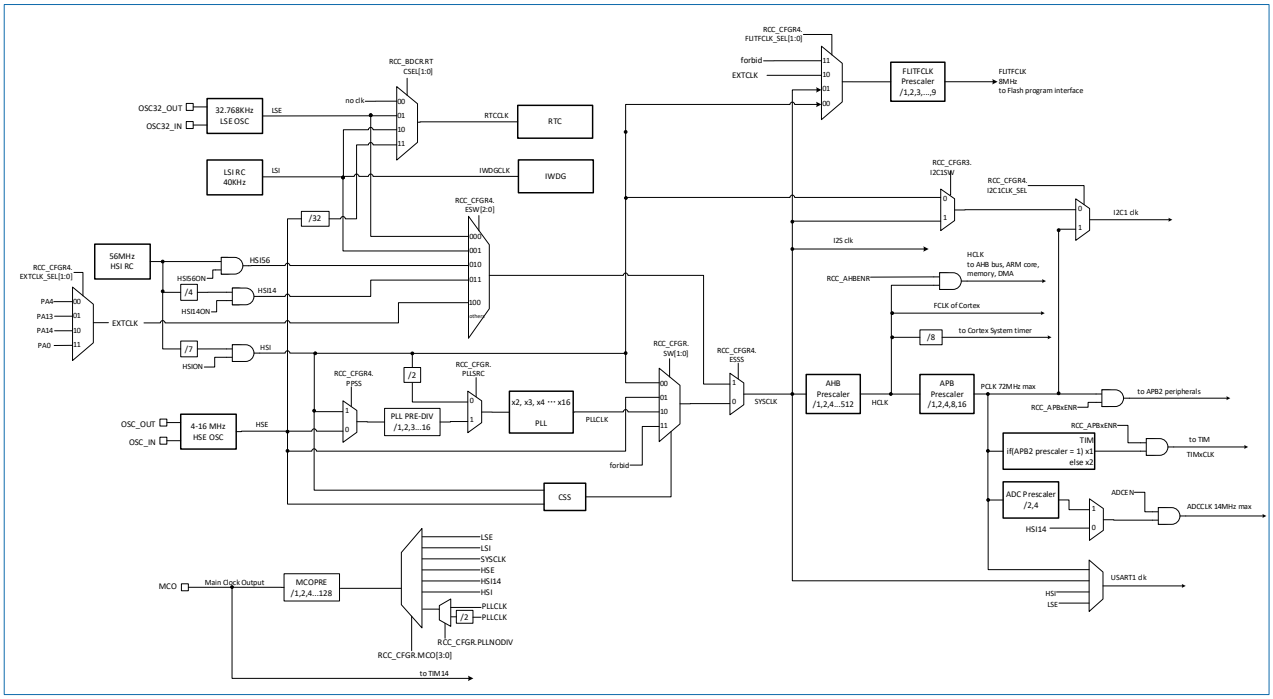


图 6-2 HK32F031 MCU 时钟树

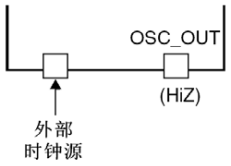
6.2.1 HSE 时钟

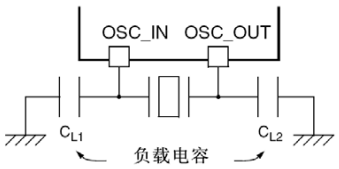
高速外部时钟信号（HSE）由以下两种时钟源产生：

- 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。

表 6-1 HSE/LSE 时钟源

时钟源	硬件配置
外部时钟	

时钟源	硬件配置
晶体/陶瓷谐振器	

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 32MHz。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 40%~60%的外部时钟信号（方波、正弦波或三角波）必须连到 OSC_IN 引脚，同时保证 OSC_OUT 不作为 IO 口使用。

外部晶振/陶瓷谐振器（HSE 晶振）

支持 4 ~16MHz 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。

6.2.2 HSI 时钟

HSI 时钟信号由 56MHz 内部 RC 振荡器进行 7 分频生成的，可直接作为系统时钟或在 2 分频后作为 PLL 输入。

在从复位重启或从待机唤醒后，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。

6.2.3 HSI14 时钟

HSI14 时钟信号是从 56MHz 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。

HSI14RC 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。

6.2.4 HSI56 时钟

HSI56 时钟信号由内部 56MHz RC 振荡器产生，可直接使用。

HSI56 要求使能与 56MHz RC 相关的 V_{REFINT} 及其缓冲区。

时钟配置寄存器 4（RCC_CFGR4）中的 HSI56RDY 位指示 HSI56RC 是否稳定。在启动时，需等待硬件将该位置 1 后，HSI56 才可以使用。

HSI56 可通过时钟配置寄存器 4（RCC_CFGR4）中的 HSI56ON 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 2%（25° C）的原因。系统复位时，工厂校准值被装载到时钟控制寄存器的 HSICAL[7:0]位（HSICAL 在时钟控制寄存器 RCC_CR 中）。

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

器（RCC_CR）里的 HSITRIM[4:0]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“6.2.9 时钟安全系统（CSS）”。

6.2.5 PLL

内部 PLL 可以用来倍频 56MHz HSI RC 的输出时钟或 HSE 晶体输出时钟，参见图 6-2 和“6.3.1 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。

修改 PLL 配置的过程如下：

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 30~72MHz。

6.2.6 LSE 时钟

LSE 晶振是 32.768kHz 低速外部晶振或陶瓷谐振器。LSE 可作为实时时钟（RTC）的时钟源来提供时钟/日历或其它定时功能，具有功耗低且精度高的优点。

通过配置备份域控制寄存器（RCC_BDCR）中的 LSEON 位，可打开或关闭 LSE 晶振。

通过配置备份域控制寄存器（RCC_BDCR）中的 LSEDRV[1:0]位，可在运行时更改晶振驱动强度，以实现稳健性、短启动时间和低功耗之间的最佳平衡。可在不同驱动水平之间动态更改驱动能力，但已运行在低驱动模式的情况除外。此时，只能通过上电复位或 RTC 复位切换到另一模式。

备份域控制寄存器（RCC_BDCR）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 1MHz。通过将备份域控制寄存器（RCC_BDCR）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 引脚应保持为高阻态（Hi-Z）。

6.2.7 LSI 时钟

LSI RC 可作为低功耗时钟源在停机和待机模式下保持运行，供独立看门狗（IWDG）使用。时钟频率在 40kHz 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 和/或 RTC 正常工作。

可通过测量 LSI 振荡器的频散来获得准确的 RTC 时基和/或精度水平可接受的 IWDG 超时（当 LSI 用作这些外设的时钟源时）。

6.2.8 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）：

- HSI56 振荡器
- HSI14 振荡器
- HSI 振荡器
- LSI 振荡器
- LSE 振荡器
- HSE 振荡器
- EXTCLK 外部时钟
- PLL

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。

6.2.9 时钟安全系统（CSS）

时钟安全系统可以通过软件被激活。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 时钟关闭后禁能。

如果检测到 HSE 时钟故障，HSE 振荡器自动被关闭，并且会产生 CSS 中断（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。CSS 与 Cortex®-M0+NMI（不可屏蔽中断）异常向量相连接。

注意：一旦 CSS 被激活（RCC_CR.CSSON=1），并且 HSE 时钟出现故障，CSS 中断就会产生（RCC_CIR.CSSF=1），并且 NMI 也自动产生。NMI 将被不断执行，直到 CSS 中断挂起位被清除。因此，在 NMI 的处理程序中必须通过设置时钟中断寄存器（RCC_CIR）里的 CSSC 位来清除 CSSF 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。

6.2.10 RTC 时钟

RTC 可以是 LSE、LSI 或 HSE/32 时钟（HSE 经过 32 分频获得）。通过配置 RCC_BDCR 寄存器中的 RTC_SEL[1:0]位，选择 RTC 的时钟源。

一旦选定 RTC 时钟源后，只能通过将 RCC_BDCR 寄存器中的 BDRST 位置 1 的方式来修改 RTC 时钟源。

如果使用 LSE 或 LSI 作为 RTC 时钟源，RTC 会继续在停机和待机两种低功耗模式下工作，并可用作唤醒源。但是，如果使用 HSE 作为 RTC 时钟源，则 RTC 不能在停机和待机两种低功耗模式下使用。

如果 RTC 的时钟由 LSE 提供，则 RTC 在系统复位后仍可获得时钟并保持正常工作。

*注意：为了能够在 APB1 时钟频率低于 RTC 时钟频率的七倍（7*RTCLCK）时读取 RTC 日历寄存器，软件必须分两次读取日历时间和日期寄存器。这样，当两次读取的 RTC_TR 结果相同时，才能确保数据正确。否则必须执行第三次读访问。*

6.2.11 看门狗时钟

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

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

6.2.12 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 或 16，最高可达 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：

- HSI14
- 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[3:0]位控制。

6.3 RCC 寄存器

基地址：0x4002 1000

空间大小：0x400

6.3.1 时钟控制寄存器（RCC_CR）

偏移地址：0x00

复位值：0x0000XX83

访问：无等待状态，字、半字和字节访问。

说明：X 表示不定值。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						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
HSICAL[7:0]								HSITRIM[4:0]					Res	HSIRDY	HSION
r								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。 当进入停机或待机模式时，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 就绪，时钟检测器打开。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 6 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。 当进入停机或待机模式后，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 15:8	HSICAL[7:0]：HSI 时钟校准（HSI clock calibration） 在启动时，该位域被自动初始化为出厂值校准值。
位 7:3	HSITRIM[4:0]：HSI 时钟调整（HSI clock trimming） 在 HSICAL[7:0]的基础上，用户可以在该位域输入一个调整数值，根据电压和温度的变化调整内部 HSI RC 振荡器的频率。默认值为 16。 芯片内部有一个默认输出频率为 56MHz 的 RC 振荡器，通过 7 分频产生一个 8MHz 的时钟和 4 分频产生一个 14MHz 的时钟。14MHz 时钟除了可以作为 SYSCLK，还输出给

	ADC 使用。HSICAL 调整的是 RC 振荡器的输出频率，因此调整 HSICAL 的值会同时影响 56M 时钟，以及 14M 时钟 HSI 8M 时钟频率。HSICAL 的调整步进约 40kHz。
位 2	Res：保留 必须保持复位值。
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 6 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。 当从待机和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

6.3.2 时钟配置寄存器（RCC_CFGR）

偏移地址：0x04

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
PLLNODIV	MCOPRE[2:0]			MCO[3:0]				Res		PLLMUL[3:0]				PLLXTPRE	PLLSRC
rw	rw			rw						rw				rw	rw

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

位 31	PLLNODIV：控制 PLL 分频输出到 MCO（PLL clock not divided for MCO） 该位由软件置位或清零。 • 0：PLL 2 分频后通过 MCO 输出 • 1：PLL 不分频通过 MCO 输出
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 • ... • 111：MCO/128
位 27:24	MCO[3:0]：微控制器时钟输出选择（Microcontroller clock output）

	该位由软件设置和清除。 • 0000：时钟输出禁止，MCO 引脚上没有时钟输出 • 0001：选择内部 HSI14 时钟输出 • 0010：选择内部 LSI 时钟输出 • 0011：选择外部 LSE 时钟输出 • 0100：系统时钟（SYSCLK） • 0101：选择内部 HSI 时钟输出 • 0110：选择外部 HSE 时钟输出 • 0111：选择 PLL 时钟输出（PLL 或 PLL 2 分频，由 PLLNODIV 决定） • 1xxx：保留 *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 23:22	Res：保留 必须保持复位值。
位 21:18	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 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 17	PLLXTPRE：PLL 输入时钟 HSE 分频器（HSE divider for PLL input clock） PLLXTPRE 与 RCC_CFGR2.PREDIV[0]意义相同。
位 16	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/2 作为 PLL 输入时钟 • 1：RCC_CFGR4.PPSS 选择的时钟作为 PLL 输入时钟
位 15:11	Res：保留

	必须保持复位值。
位 10:8	PPRE[2:0]：APB 低速预分频器（APB1）（PCLK prescaler） 该位域由软件置位和清零，用于控制 APB 低速时钟（PCLK）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。在从停机或待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

6.3.3 时钟中断寄存器（RCC_CIR）

偏移地址：0x08

复位值：0x0000 0000

访问：无等待状态，字、半字和字节访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								CSSC	Res	HSI14RDYC	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
								w		w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		HSI14RDYIE	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	Res	HSI14RDYF	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
		rw	rw	rw	rw	rw	rw	rw		r		r	r	r	r

位 31:24	Res：保留 必须保持复位值。
位 23	CSSC：时钟安全系统中断清零(Clock security system interrupt 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 CSSF 标志清零
位 22	Res：保留 必须保持复位值。
位 21	HSI14RDYC：HSI14 就绪中断清零(HSI14 ready interrupt clear) 该位由软件置 1，用于将 HSI14RDYF 标志清零。该位由硬件复位。 • 0：不起作用 • 1：将 HSI14RDYF 标志清零
位 20	PLLRDYC：PLL 就绪中断清零(PLL ready interrupt clear) 由软件置 1 来清除 PLLRDYF 标志 • 0：无作用 • 1：清 PLLRDYF 标志
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) 由软件置 1 来清除 LSERDYF 标志 • 0：无作用 • 1：清 LSERDYF 标志

位 16	LSIRDYC：LSI 就绪中断清零(LSI ready interrupt clear) 由软件置 1 来清除 LSIRDYF 标志 • 0：无作用 • 1：清除 LSIRDYF 标志
位 15:14	Res：保留 必须保持复位值。
位 13	HSI14RDYIE：HSI14 就绪中断使能(HSI14 ready interrupt enable) 该位由软件置 1 和复位，用于使能/禁止由 HSI14 振荡器稳定所引起的中断。 • 0：禁止 HSI14 就绪中断 • 1：使能 HSI14 就绪中断
位 12	PLLRDYIE：PLL 就绪中断使能(PLL ready interrupt enable) 由软件设置和清除来使能/关闭由 PLL 锁定引发的中断 • 0：PLL 锁定中断关闭 • 1：PLL 锁定中断使能
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) 由软件设置和清除来使能/关闭由 LSE 振荡器引发的就绪中断 • 0：LSE 就绪中断关闭 • 1：LSE 就绪中断使能
位 8	LSIRDYIE：LSI 就绪中断使能(LSI ready interrupt enable) 由软件设置和清除来使能/关闭由 LSI 振荡器引发的就绪中断 • 0：LSI 就绪中断关闭 • 1：LSI 就绪中断使能
位 7	CSSF：时钟安全系统中断标志(Clock security system interrupt flag) 该位通过软件写入 CSSC 位复位。如果 HSE 时钟出现故障，则由硬件置 1。 • 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	HSI14RDYF：HSI14 就绪中断标志(HSI14 ready interrupt flag) 该位通过软件写入 HSI14RDYC 位复位。当 HSE 时钟稳定且 HSI14RDYIE 置 1 时，该位由硬件置 1。 • 0：当前未因 HSI14 时钟故障而引起时钟就绪中断 • 1：因 HSI14 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 PLLRDYDIE=1 时由硬件对该位置 1。软件置 PLLRDYC=1 时该位清除。 • 0：无由 PLL 时钟引发的时钟就绪中断 • 1：有由 PLL 时钟引发的时钟就绪中断
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) 该位通过软件写入 HSIRDYC 位复位。当 CSS 稳定且 HSIRDYIE 置 1 时，该位由硬件置 1。 • 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 LSERDYDIE=1 时由硬件对该位置 1。软件置 LSERDYC=1 时该位清除。 • 0：无由 LSE 振荡器引发的时钟就绪中断 • 1：有由 LSE 振荡器引发的时钟就绪中断
位 0	LSIRDYF：LSI 就绪中断标志(LSI ready interrupt flag) 当 LSI 时钟就绪且 LSIRDYDIE=1 时由硬件对该位置 1。软件置 LSIRDYC=1 时该位清除。 • 0：无由 LSI 振荡器引发的时钟就绪中断 • 1：有由 LSI 振荡器引发的时钟就绪中断

6.3.4 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									DBGRST	Res			TIM17RST	TIM16RST	Res
									rw				rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	USART1RST	Res	SPI1RST	TIM1RST	Res	ADC1RST	Res								SYSCFGRST
	rw		rw	rw		rw									rw

位 31:23	Res：保留 必须保持复位值。
位 22	DBGRST：DBG 复位（DBG reset） 此位由软件置位和清零。 • 0：不起作用 • 1：复位 DBG
位 21:19	Res：保留 必须保持复位值。
位 18	TIM17RST：将 TIM17 定时器复位（Reset TIM17） 由软件置位或清零。 • 0：不起作用 • 1：复位 TIM17 定时器
位 17	TIM16RST：将 TIM16 定时器复位（Reset TIM16） 由软件置位或清零。 • 0：不起作用 • 1：复位 TIM16 定时器
位 16:15	Res：保留 必须保持复位值。
位 14	USART1RST：将 USART1 复位（Reset USART1） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 USART1
位 13	Res：保留 必须保持复位值。
位 12	SPI1RST：将 SPI1 复位（Reset SPI1） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 SPI1
位 11	TIM1RST：将 TIM1 定时器复位（Reset TIM1） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 TIM1 定时器
位 10	Res：保留

	必须保持复位值。
位 9	ADCRST：将 ADC 接口复位（Reset ADC interface） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 ADC 接口
位 8:1	Res：保留 必须保持复位值。
位 0	SYSCFGRST：将 SYSCFG 复位（Reset SYSCFG） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 SYSCFG

6.3.5 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						I2C1RST	Res				
			rw							rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				WWDGRST	Res		TIM14RST	Res			TIM6RST	Res		TIM3RST	TIM2RST
				rw			rw				rw			rw	rw

位 31:29	Res：保留 必须保持复位值。
位 28	PWRRST：将电源接口复位（Reset Power interface） 此位由软件置位和清零。 • 0：不起作用 • 1：复位电源接口
位 27:22	Res：保留 必须保持复位值。
位 21	I2C1RST：将 I2C1 复位（Reset I2C1） 此位由软件置位和清零。 • 0：不起作用 • 1：复位 I2C1
位 20:12	Res：保留 必须保持复位值。

位 11	WWDGRST：将窗口看门狗复位（Reset WWDG） 此位由软件置位和清零。 • 0：不起作用 • 1：复位窗口看门狗
位 10:9	Res：保留 必须保持复位值。
位 8	TIM14RST：将 TIM14 定时器复位（Reset TIM14） 此位由软件置位或清零。 • 0：不起作用 • 1：复位 TIM14
位 7:5	Res：保留 必须保持复位值。
位 4	TIM6RST：将 TIM6 定时器复位（Reset TIM6） 此位由软件置位或清零。 • 0：不起作用 • 1：复位 TIM6
位 3:2	Res：保留 必须保持复位值。
位 1	TIM3RST：将 TIM3 定时器复位（Reset TIM3） 此位由软件置位和清零。 • 0：不起作用 • 1：复位 TIM3
位 0	TIM2RST：将 TIM2 定时器复位（Reset TIM2） 此位由软件置位和清零。 • 0：不起作用 • 1：复位 TIM2

6.3.6 AHB 外部时钟使能寄存器（RCC_AHBENR）

偏移地址：0x14

复位值：0x0000 0014

访问：无等待周期，支持字、半字和字节访问。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res									CRC EN	Res	FLITF EN	Res	SRA MEN	Res	DMA EN
									rw		rw		rw		rw

位 31:23	Res：保留 必须保持复位值。
位 22	IOPFEN：GPIOF 时钟使能（I/O port F clock enable） 该位由软件置位或清零。 • 0：GPIOF 时钟关闭 • 1：GPIOF 时钟开启
位 21:20	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 16:7	Res：保留 必须保持复位值。
位 6	CRCEN：CRC 时钟使能（CRC clock enable） 该位由软件置位或清零。 • 0：CRC 时钟关闭 • 1：CRC 时钟开启
位 5	Res：保留 必须保持复位值。
位 4	FLITFEN：FLITF 时钟使能（FLITF 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） 该位由软件置位或清零来开启/关闭在睡眠模式下的 SRAM 时钟。 • 0：在睡眠模式下 SRAM 接口时钟关闭 • 1：在睡眠模式下 SRAM 接口时钟开启
位 1	Res：保留 必须保持复位值。
位 0	DMAEN：DMA 时钟使能位（DMA clock enable） 该位由软件置 1 和复位。 • 0：禁能 DMA 时钟 • 1：使能 DMA 时钟

6.3.7 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									DBGEN	Res			TIM17EN	TIM16EN	Res
									rw				rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	USART1EN	Res	SPI1EN	TIM1EN	Res	ADCEN	Res								SYSCFGEN
	rw		rw	rw		rw									rw

位31:23	Res：保留 必须保持复位值。
位22	DBGEN：调试模块时钟使能（DBG clock enable） 该位由软件置位和清零。 • 0：调试模块时钟关闭 • 1：调试模块时钟开启
位21:19	Res：保留 必须保持复位值。
位 18	TIM17EN：TIM17 定时器时钟使能（TIM17clock enable） 该位由软件置位和清零。 • 0：TIM17 定时器时钟关闭 • 1：TIM17 定时器时钟开启
位 17	TIM16EN：TIM16 定时器时钟使能（TIM16clock enable） 该位由软件置位和清零。 • 0：TIM16 定时器时钟关闭 • 1：TIM16 定时器时钟开启

位 16:15	Res：保留 必须保持复位值。
位 14	USART1EN：USART1 时钟使能（USART1clock enable） 该位由软件置位和清零。 • 0：USART1 时钟关闭 • 1：USART1 时钟开启
位13	Res：保留 必须保持复位值。
位 12	SPI1EN：SPI1 时钟使能（SPI1clock enable） 该位由软件置位和清零。 • 0：SPI1 时钟关闭 • 1：SPI1 时钟开启
位11	TIM1EN：TIM1 定时器时钟使能（TIM1clock 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10	Res：保留 必须保持复位值。
位9	ADCEN：ADC 接口时钟使能（ADC clock enable） 该位由软件置位和清零。 • 0：ADC 接口时钟关闭 • 1：ADC 接口时钟开启
位8:1	Res：保留 必须保持复位值。
位0	SYSCFGEN：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

6.3.8 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						I2C1EN	Res				
			rw							rw					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				WWDGEN	Res		TIM14EN	Res			TIM6EN	Res		TIM3EN	TIM2EN
				rw			rw				rw			rw	rw

位 31:29	Res：保留 必须保持复位值。
位 28	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 27:22	Res：保留 必须保持复位值。
位 21	I2C1EN：I2C1 时钟使能位（I2C1 clock enable） 此位由软件置位和清零。 • 0：禁能 I2C1 时钟 • 1：使能 I2C1 时钟
位 20:12	Res：保留 必须保持复位值。
位 11	WWDGEN：窗口看门狗时钟使能位（WWDG clock enable） 此位由软件置位和清零。 • 0：禁能窗口看门狗时钟 • 1：使能窗口看门狗时钟
位 10:9	Res：保留 必须保持复位值。
位 8	TIM14EN：定时器 14 时钟使能位（TIM14 clock enable） 此位由软件置位和清零。 • 0：禁能定时器 14 时钟 • 1：使能定时器 14 时钟
位 7:5	Res：保留 必须保持复位值。
位 4	TIM6EN：定时器 6 时钟使能位（TIM6 clock enable） 此位由软件置位和清零。 • 0：禁能定时器 6 时钟 • 1：使能定时器 6 时钟
位 3:2	Res：保留 必须保持复位值。

位 1	TIM3EN：定时器 3 时钟使能位（TIM3 clock enable） 此位由软件置位和清零。 • 0：禁能定时器 3 时钟 • 1：使能定时器 3 时钟
位 0	TIM2EN：定时器 2 时钟使能位（TIM2 clock enable） 此位由软件置位和清零。 • 0：禁能定时器 2 时钟 • 1：使能定时器 2 时钟

6.3.9 备份域控制寄存器（RCC_BDCR）

偏移地址：0x20

复位值：0x0000 0000（由备份域复位电路复位）

访问：支持字、半字和字节访问；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。

注意：LSEON、LSEBYP、RTC_SEL 和 RTCEN 位在 RTC 域，因此，复位后这些位将被写保护；如果要写这些位，需要先把 PWR_CR.DBP 置 1。同时，只有 RTC 域的复位会复位这些位。

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					RTC_SEL[1:0]		Res			LSEDRV[1:0]		LSEBYP	LSERDY	LSEON
rw						rw					rw		rw	r	rw

位 31:17	Res：保留 必须保持复位值。
位 16	BDRST：备份域软件复位（Backup domain software reset） 该位由软件置位或清零。 • 0：复位未激活 • 1：复位整个 RTC 备份域
位 15	RTCEN：RTC 时钟使能位（RTC clock enable） 该位由软件置位和清零。 • 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	RTC_SEL[1:0]：RTC 时钟源选择（RTC clock source selection） 由软件设置来选择 RTC 时钟源。一旦 RTC 时钟源被选定，这些位的值不能被改变，除非 RTC 域被复位。可通过设置 BDRST 位来复位 RTC 域。 • 00：无时钟 • 01：LSE 振荡器作为 RTC 时钟

	• 10：LSI 振荡器作为 RTC 时钟 • 11：HSE/32 作为 RTC 时钟
位 7:5	Res：保留 必须保持复位值。
位 4:3	LSEDRV[1:0]：LSE 振荡器驱动能力位（LSE oscillator drive capability） 由软件设置或清除 LSE 振荡器驱动能力的配置。当复位备份域时，会重装载缺省值。 • 00：最低驱动 • 01：中低驱动 • 10：中高驱动 • 11：最高驱动
位 2	LSEBYP：外部低速振荡器旁路位（LSE oscillator bypass） 由软件设置和清除，该位仅在外部 32kHz 振荡器关闭时写值。 通过将 RTCRST 位置 1 或者通过 POR 将该位复位。 • 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 32kHz 振荡器是否就绪。在 LSEON 被清零后，该位需要 6 个外部低速振荡器的周期才被清零。 通过将 RTCRST 位置 1 或者通过 POR 将该位复位。 • 0：外部 32kHz 振荡器未就绪 • 1：外部 32kHz 振荡器已就绪
位 0	LSEON：外部低速振荡器使能位（LSE oscillator enable） 该位由软件置位和清零。 通过将 RTCRST 位置 1 或者通过 POR 将该位复位。 • 0：LSE 振荡器关闭 • 1：LSE 振荡器开启

6.3.10 控制/状态寄存器（RCC_CSR）

偏移地址：0x24

复位值：0x0C00 0000

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

访问：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	V15PWRRSTF	Res						
r	r	r	r	r	r	r	rt_w	r							

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														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：未发生选项字节装载器复位 • 1：发生选项字节装载器复位

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	V15PWRRSTF：1.5V COREVCC 复位标志（Reset flag of the 1.5V domain） • 0：1.5V 电源域未发生复位 • 1：1.5V 电源域有发生复位
位 22:2	Res：保留 必须保持复位值。
位 1	LSIRDY：LSI 振荡器就绪（LSI oscillator ready） 通过硬件置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

6.3.11 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									IOPFRST	Res		IOPCRST	IOPBRST	IOPARST	Res
									rw			rw	rw	rw	

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

位 31:23	Res：保留 必须保持复位值。
位 22	IOPFRST：将 GPIOF 口复位（Reset I/O port F） • 0：不起作用 • 1：复位 GPIOF 口
位 21:20	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 16:7	Res：保留 必须保持复位值。
位 6	CRCRST：将 CRC 模块复位（Reset CRC） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 CRC
位 5:1	Res：保留 必须保持复位值。
位 0	DMARST：将 DMA 复位（Reset DMA） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 DMA

6.3.12 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												PREDIV[3:0]			
												rw			

位 31:4	Res：保留 必须保持复位值。
位 3:0	PREDIV[3:0]：PREDIV 分频因子（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 17 相同，修改 RCC_CFGR 的位 17 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

6.3.13 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							ADCSW	Res			I2C1SW	Res		USART1SW[1:0]	
							rw				rw			rw	

位 31:9	Res：保留 必须保持复位值。
位 8	ADCSW：ADC 时钟源选择（ADC clock source selection） 由软件设置和清 0 来选择 ADC 的时钟源。 • 0：HSI14 时钟被选为 ADC 时钟 • 1：PLCK2 分频或 4 分频被选为 ADC 时钟 *注：当 HSI14 被选为 ADC 的时钟源，则 HSI14 振荡器不能关闭(RCC_CR2 中的* HSI14DIS=0)。
位 7:5	Res：保留 必须保持复位值。

位 4	I2C1SW：选择 I2C1 时钟源（I2C1 clock source selection） 该位和 RCC_CFGR4.I2C1CLK_SEL 共同决定 I2C1 时钟源，由软件置位和清零。 当 RCC_CFGR4.I2C1CLK_SEL 为 0 时，I2C1 的时钟源为： • 0：HSI 时钟被选为 I2C1 时钟源 • 1：系统时钟（SYSCLK）被选为 I2C1 的时钟源
位 3:2	Res：保留 必须保持复位值。
位 1:0	USART1SW[1:0]：USART1 时钟源选择（USART1 clock source selection） 由软件置位和清零来选择 USART1 的时钟源。 • 00：PCLK 被选为 USART1 的时钟源（缺省） • 01：系统时钟（SYSCLK）被选为 USART1 的时钟源 • 10：LSE 时钟被选为 USART1 的时钟源 • 11：HSI 时钟被选为 USART1 的时钟源

6.3.14 时钟控制寄存器 2（RCC_CR2）

偏移地址：0x34

复位值：0x1E00 0000

访问：无等待周期，支持字、半字和字节访问

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HSI14CAL[7:0]								HSI14TRIM[4:0]					HSI14DIS	HSI14RDY	HSI14ON
r								rw					rw	r	rw

位 31:16	Res：保留 必须保持复位值。
位 15:8	HSI14CAL[7:0]：HSI14 时钟校准（HSI14clock calibration） 参见 RCC_CR.HSICAL[7:0]。
位 7:3	HSI14TRIM[4:0]：HSI14 时钟调整（HSI14clock trimming） 参见 RCC_CR.HSITRIM[4:0]。
位 2	HSI14DIS：ADC HSI14 时钟请求禁止（HSI14clock 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 HSI14 振荡器。 • 0：ADC 控制器可以打开 HSI14 时钟 • 1：ADC 控制器不能打开 HSI14 时钟
位 1	HSI14RDY：HSI14 时钟就绪标志（HSI14clock ready flag） 由硬件置 1 来指示内部 HSI14 振荡器已经稳定。在 HSI14ON 位清零后，HSI14RDY 位需要等待 6 个 HSI14 振荡器周期再清零。

	• 0：HSI14 时钟未就绪 • 1：HSI14 时钟已就绪
位 0	HSI14ON：HSI14 时钟使能（HSI14clock enable） 该位由软件置位或清零。 • 0：关闭 HSI14 时钟 • 1：打开 HSI14 时钟

6.3.15 RCC HSE 时钟控制寄存器（RCC_HSECTL）

偏移地址：0xE0

复位值：0x1E2E 0020

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
CSS_THRESHOLD[6:0]							Res						HSEDRV[2:0]		
rw													rw		

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

位 31:25	CSS_THRESHOLD[6:0]：控制 CSS 计数器的阈值（CSS counter threshold control） 当 CSS 功能开启后，RCC 使用 HSI 时钟来采样 HSE 分频后的波形。如果 HSE 输入频率非常低，则即使 HSE 还在正常工作也可能触发 CSS 中断。可以调整通过 CSS_THRESHOLD[6:0]的值来避免这种情况。当配置不同的阈值时，CSS 判断的最低 HSE 频率大约为 4M/CSS_THRESHOLD[6:0]。因此在复位值情况下，当 HSE 输入小于 266kHz 时就会产生 CSS 中断。
位 24:19	Res：保留 必须保持复位值。
位 18:16	HSEDRV[2:0]：调整 HSE 的驱动能力 HSEDRV 的值越大芯片 HSE 的驱动能力越大。
位 15:12	Res：保留 必须保持复位值。
位 11:0	HSERDYWT[11:0]：设置 HSE 就绪等待时间（HSE stabilize wait time setting） 在 HSEON=1 后，HSE 输入 HSERDYWT[11:0]x8 个时钟后置位 HSERDY。默认等待时钟数为 64x8=512。

6.3.16 时钟配置寄存器 4（RCC_CFGR4）

偏移地址：0xE8

复位值：0x0000 0038

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						EXTCLK_SEL[1:0]		Res						HSI56RDY	HSI56ON
						rw								rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2C1CLK_SEL	FLITFCLK_PRE[3:0]				FLITFCLK_SEL[1:0]		PPSS	ESSS	Res	ESWS[2:0]			ESW[2:0]		
rw	rw				rw		rw	rw		r			rw		

位 31:26	Res：保留 必须保持复位值。
位25:24	EXTCLK_SEL[1:0]：外部时钟输入管脚选择（External clock pin selection） • 00：外部时钟管脚 1（PA4）选择作为输入源 • 01：外部时钟管脚 2（PA13）选择作为输入源 • 10：外部时钟管脚 3（PA14）选择作为输入源 • 11：外部时钟管脚 4（PA0）选择作为输入源
位 23:18	Res：保留 必须保持复位值。
位 17	HSI56RDY：芯片内部 56MHz RC 振荡器就绪（HSI56oscillator ready） • 0：HSI 56MHz 时钟未就绪 • 1：HSI 56MHz 时钟就绪
位 16	HSI56ON：打开芯片内部 56MHz RC 振荡器（HSI 56oscillator enable） • 0：关闭 HSI56 时钟 • 1：打开 HSI56 时钟
位 15	I2C1CLK_SEL：配合 RCC_CFGR3.I2C1SW 选择 I2C1 时钟源（I2C1clock source selection） • 0：I2C1 时钟源由 RCC_CFGR3.I2C1SW 决定 • 1：PCLK 用作 I2C1 时钟源
位 14:11	FLITFCLK_PRE[3:0]：Flash 擦写操作时钟分频因子（Flash memory programming interface clock prescaler factor） 分频数为 FLITFCLK_PRE + 1，比如 FLITFCLK_PRE 为 2 时，分频数为 3。 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 8MHz。 *注意：不能在执行 Flash 编程和擦除的时候，更改 FLITFCLK_PRE 寄存器的值。*
位 10:9	FLITFCLK_SEL[1:0]：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：不可用 *注意：不能在执行 Flash 编程和擦除的时候更改 FLITFCLK_SEL 寄存器的值。*
位 8	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 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 56MHz RC 振荡器输出作为 SYSCLK • 011：HSI14 作为 SYSCLK • 100：外部时钟管脚输入作为 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 56MHz RC 振荡器输出作为 SYSCLK • 011：选择 HSI14 作为 SYSCLK • 100：选择外部时钟管脚输入作为 SYSCLK • 其它：未定义

6.3.17 AHB 外设时钟使能寄存器 2（RCC_AHBENR2）

偏移地址：0xEC

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															DVSQEN
															rw

位 31:1	Res：保留 必须保持复位值。
位 0	DVSQEN：DVSQ 时钟使能（DVSQ clock enable） 该位由软件置位或清零。 • 0：DVSQ 时钟关闭

	• 1：DVSQ 时钟开启

6.3.18 AHB 外设复位寄存器 2（RCC_AHBRSTR2）

偏移地址：0x110

复位值：0x0000 0000

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															DVSQRST
															rw

位 31:1	Res：保留 必须保持复位值。
位 0	DVSQRST：将 DVSQ 复位（Reset DVSQ） 该位由软件置位或清零。 • 0：不起作用 • 1：复位 DVSQ 模块

7 通用 I/O（GPIO）

本章介绍该系列芯片的 GPIO 功能和寄存器描述。

7.1 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之间发生中断请求也不会有问题。

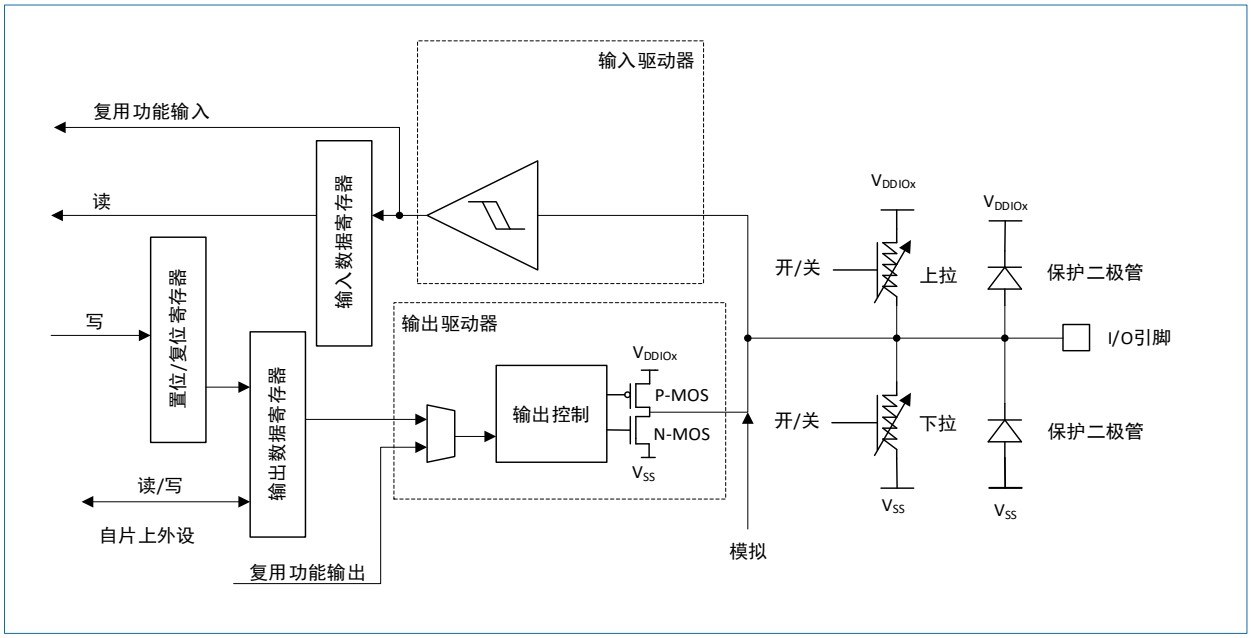


图 7-1 I/O 端口位的基本结构

表 7-1 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	

MODER（i）[1:0]	OTYPER（i）	OSPEEDR（i）[1:0]		PUPDR（i）[1:0]		I/O 配置	
	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
	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。

7.1.1 通用 I/O（GPIO）

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

- PA14：SWCLK 处于下拉状态
- PA13：SWDIO 处于上拉状态

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

输入数据寄存器（GPIOx_IDR）每隔 1 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。

7.1.2 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 16 路复用功能输入（AF0 到 AF15），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：在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟。
- 外设复用功能：
  C. 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需的 AFx。
  D. 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器，分别选择类型、上拉/下拉以及输出速度。
  E. 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，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC 寄存器中配置所需功能。

注意：对于 RTC、WKUPx 和振荡器的其它功能，在相关的 RTC、PWR 和 RCC 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。

7.1.3 I/O 端口控制寄存器

每个通用 I/O 端口包括：

- 6 个 32 位的控制寄存器，可配置多达 16 个 IO：
  - 端口模式寄存器 GPIOx_MODER，用于选择 I/O 模式（输入、输出、AF 或模拟）。
  - 端口输出类型寄存器 GPIOx_OTYPER，用于选择输出类型（推挽或开漏）。
  - 端口输出速度寄存器 GPIOx_OSPEEDR，用于选择速度。
  - 端口上下拉寄存器 GPIOx_PUPDR，用于选择上拉/下拉。
- 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，可以其进行读/写访问。

7.1.4 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。

7.1.5 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 和 GPIOx_AFRH。

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 和 GPIOx_AFRH）中的对应位。

7.1.6 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 选择信号由复位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。

7.1.7 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。

7.1.8 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。
- 每隔 1 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。

7.1.9 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。
- 输入数据寄存器每隔 1 个 AHB 时钟周期对 I/O 引脚上的数据采样一次。
- 在开漏模式时，对输入数据寄存器的读访问可获取 I/O 的状态。
- 在推挽模式时，对输出数据寄存器的读访问可获取最后的写入值。

7.1.10 复用功能配置

对 I/O 端口进行编程作为复用功能时：

- 可将输出缓冲器配置为开漏或推挽模式。
- 输出缓冲器由来自外设的信号驱动（发送使能和数据）。
- 根据 GPIOx_PUPDR 寄存器的配置决定是否打开上拉电阻和下拉电阻。
- 输入数据寄存器每隔 1 个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样。
- 对输入数据寄存器的读访问可获取 I/O 状态。

注意：当所选复用功能为 LCD 功能时，不应用上述复用功能配置。这在种情况下，编程为复用功能输出的 I/O 按模拟配置中所述进行配置。

7.1.11 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。
- 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'。

7.1.12 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。

7.1.13 在 RTC 电源域中使用 GPIO 引脚

当内核电源域掉电时（器件进入待机模式时），PC13/PC14/PC15GPIO 功能会丢失。在这种情况下，如果不被配置成 RTC 相关复用功能，则会被自动设置成模拟输入模式。

7.2 GPIO 寄存器

基地址：（GPIOA; GPIOB, GPIOC, GPIOF）=（0x4800 0000; 0x4800 0400, 0x4800 0800, 0x4800 1400）

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

7.2.1 GPIO 端口模式寄存器（GPIOx_MODER）

偏移地址：0x00

复位值：0x2800 0000（端口 A）/ 0x0000 0000（其他端口）

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	MODERy[1:0]：端口 x 的 y 引脚工作模式配置位（Port x pin y mode bits）

（y=15..0）	该位域可由软件配置 I/O 口的工作模式。 • 00：输入模式（复位状态） • 01：通用输出模式 • 10：复用功能模式 • 11：模拟模式

7.2.2 GPIO 端口输出类型寄存器（GPIOx_OTYPER）

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 bits） 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

7.2.3 GPIO 口输出速度寄存器（GPIOx_OSPEEDR）

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
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 bits） 该位域由软件配置 I/O 口的速度。 • 00/10：低速 • 01：中速 • 11：高速

7.2.4 GPIO 口上拉/下拉寄存器（GPIOx_PUPDR）

偏移地址：0x0C

复位值：0x2400 0000（端口 A）/ 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 配置的 y 引脚配置为上拉或下拉（Port x pin y pull -up/-down bits） 该位域由软件配置 I/O 口为上拉或下拉。 • 00：无上拉和下拉 • 01：上拉 • 10：下拉 • 11：保留

7.2.5 GPIO 端口输入数据寄存器（GPIOx_IDR）

偏移地址：0x10

复位值：0x0000XXXX

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
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 的输入数据（Port input data bit） 该位包含相应 I/O 口的输入值，只能读。

7.2.6 GPIO 端口输出数据寄存器（GPIOx_ODR）

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 output data bit） 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

7.2.7 GPIO 端口置位/复位寄存器（GPIOx_BSRR）

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 复位控制位（Port x reset bit y） 该位只能写。 读该位时返回值为0。若 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 设置控制位（Port x set bit y） 该位只能写。 读该位时返回值为0。 • 0：对相应的 ODRy 位无影响 • 1：置位端口 x 的 ODRy 位

7.2.8 GPIO 端口配置锁定寄存器（GPIOx_LCKR）

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 锁定位（Port x lock bit y） 该位可读/写，但仅 LCKK 为 0 时写。 • 0：端口配置未锁定 • 1：端口配置锁定

7.2.9 GPIO 复用功能低位寄存器（GPIOx_AFRL）

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 • 其他值：保留 • 1111：AF15

7.2.10 GPIO 复用功能高位寄存器（GPIOx_AFRH）

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+3:4y（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 • 其他值：保留 • 1111：AF15

7.2.11 GPIO 端口位复位寄存器（GPIOx_BRR）

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

8 系统配置控制器（SYSCFG）

该系列芯片有一组配置寄存器，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 超快模式（Fast Mode Plus）。
- 重映射部分 DMA 触发源到其它不同的 DMA 通道上。
- 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。
- 管理系统的可靠性特性。

8.1 SYSCFG 寄存器

基地址：0x4001 0000

空间大小：0x400

8.1.1 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								I2C_PA10_FMP	I2C_PA9_FMP	Res	I2C1_FMP	I2C_PB9_FMP	I2C_PB8_FMP	I2C_PB7_FMP	I2C_PB6_FMP
								rw	rw		rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			TIM17_DMA_RMP	TIM16_DMA_RMP	USART1_RX_DMA_RMP	USART1_TX_DMA_RMP	ADC_DMA_RMP	Res						MEM_MODE[1:0]	
			rw	rw	rw	rw	rw							rw	

位 31:24	Res：保留 必须保持复位值。
位 23	I2C_PA10_FMP：超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PA10） 该位为 PA10 口开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PA10 引脚设置为标准模式或快速模式。 • 1：PA10 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 22	I2C_PA9_FMP：超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PA9） 该位为 PA9 口线开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PA9 引脚设置为标准模式或快速模式。 • 1：PA9 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 21	Res：保留 必须保持复位值。

位 20	I2C1_FMP：I2C1 的超快模式（FM+）驱动能力激活位（FM+ driving capability activation for I2C1） 该位与 I2C_Pxx_FMP 位进行或运算，由软件设置和清零。 • 0：超快模式（FM+）只是由 I2C_Pxx_FMP 位控制。 • 1：通过 GPIOx_AFR 寄存器中的选择位选择的所有 I2C1 引脚均启用超快模式（FM+），这是为没有专用 I2C_Pxx_FMP 控制位的焊盘启用超快模式（FM+）的唯一方法。
位 19	I2C_PB9_FMP：PB9 超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PB9） 该位为 PB9 口线开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PB9 引脚设置为标准模式或快速模式。 • 1：PB9 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 18	I2C_PB8_FMP：PB8 超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PB8） 该位为 PB8 口线开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PB8 引脚设置为标准模式或快速模式。 • 1：PB8 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 17	I2C_PB7_FMP：PB7 超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PB7） 该位为 PB7 口线开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PB7 引脚设置为标准模式或快速模式。 • 1：PB7 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 16	I2C_PB6_FMP：PB6 超快模式（FM+）驱动能力激活位（I2C Fast Mode Plus (FM+)driving capability activation bit for PB6） 该位为 PB6 口线开启 I2C 超快模式（FM+），由软件设置和清零。 • 0：PB6 引脚设置为标准模式或快速模式。 • 1：PB6 引脚配置为 I2C 超快模式（FM+），且 I2C 速度控制被旁路。
位 15:13	Res：保留 必须保持复位值。
位 12	TIM17_DMA_RMP：TIM17DMA 请求重映射位（TIM17 DMA request remapping bit） 该位控制 TIM17 DMA 通道请求的重映射，由软件设置和清零。 • 0：无重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射到 DMA 通道 1 上） • 1：重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射到 DMA 通道 2 上）
位 11	TIM16_DMA_RMP：TIM16DMA 请求重映射位（TIM16 DMA request remapping bit） 该位控制 TIM16 DMA 通道请求的重映射，由软件设置和清零。 • 0：无重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射到 DMA 通道 3 上） • 1：重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射到 DMA 通道 4 上）
位 10	USART1_RX_DMA_RMP：USART1_RX DMA 请求重映射位（USART1RX DMA request remapping

	bit） 该位控制 USART1_RX DMA 通道请求的重映射，由软件设置和清零。 • 0：无重映射（USART1_RX 请求映射在 DMA 通道 3 上） • 1：重新映射（USART1_RX DMA 请求映射到 DMA 通道 5 上）
位 9	USART1_TX_DMA_RMP：USART1_TX DMA 请求重映射位（USART1TX DMA request remapping bit） 该位控制 USART1_TX DMA 通道请求的重映射，由软件设置和清零。 • 0：无重映射（USART1_TX 请求映射在 DMA 通道 2 上） • 1：重映射（USART1_TX 请求映射在 DMA 通道 4 上）
位 8	ADC_DMA_RMP：ADC DMA 请求重映射位（ADC DMA request remapping bit） 该位控制 ADC DMA 通道请求的重映射，由软件设置和清零。 • 0：无重映射（ADC DMA 请求映射在 DMA 通道 1 上） • 1：重映射（ADC DMA 请求映射在 DMA 通道 2 上）
位 7:2	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：SRAM 映射到 0x0000 0000

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

偏移地址：0x08

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 4x+3:4x （x = 3..0）	EXTIx[3:0]：选择 EXTIx 的外部中断源（EXTI x external interrupt source input selection） 该位域由软件设置。 • x000：PA[x]引脚 • x001：PB[x]引脚 • x010：PC[x]引脚 • x011：PD[x]引脚

	• x100：保留 • x101：PF[x]引脚 • 其它配置：保留

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

偏移地址：0x0C

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 4(x-4)+3:4(x-4) (x =7..4)	EXTIx[3:0]：选择 EXTIx 的外部中断源（EXTI x external interrupt source input selection） 该位域由软件设置。 • x000：PA[x]引脚 • x001：PB[x]引脚 • x010：PC[x]引脚 • x011：PD[x]引脚 • x100：保留 • x101：PF[x]引脚 • 其它配置：保留

8.1.4 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(x-8)+3:4(x-8) (x = 11..8)	EXTIx[3:0]：选择 EXTIx 的外部中断源（EXTI x external interrupt source input selection） 该位域由软件设置。 • x000：PA[x]引脚

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

8.1.5 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(x-12)+3:4(x-12) (x = 15..12)	EXTIx[3:0]：选择 EXTIx 的外部中断源（EXTI x external interrupt source input selection） 该位域由软件设置。 • x000：PA[x]引脚 • x001：PB[x]引脚 • x010：PC[x]引脚 • x011：保留 • x100：保留 • x101：PF[x]引脚 • 其它配置：保留

8.1.6 SYSCFG 配置寄存器 2（SYSCFG_CFGR2）

偏移地址：0x18

复位值：0x1000 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							SRAM_PEF	Res					PVD_LOCK	SRAM_PARITY_LOCK	LOCKUP_LOCK
													rw	rw	rw

位 31:9	Res：保留 必须保持复位值。

位 8	SRAM_PEF：SRAM 校验标志（SRAM parity error flag） 当 SRAM 校验错被检测到时硬件自动设置该位。对该位写 1 时清该位。 ● 0：无 SRAM 校验错 ● 1：检测到 SRAM 校验错
位 7:3	Res：保留 必须保持复位值。
位 2	PVD_LOCK：PVD 锁定使能位（PVD lock enable bit） 该位由软件设置，由系统复位清零。可用于使能并锁定 PVD 连接到 TIM1 的刹车（Break）输入，锁定 PVDE 和 PLS[2:0]（PWR_CR 寄存器）。 ● 0：PVD 中断信号断开与 TIM1 刹车（Break）输入的连接。PVDE 和 PLS[2:0]位可被应用编程。 ● 1：PVD 中断信号连接到 TIM1 刹车（Break）输入，PVDE 和 PLS[2:0]位为只读。
位 1	SRAM_PARITY_LOCK：SRAM 校验锁定位（SRAM parity lock bit） 该位由软件设置，由系统复位清除。可用于锁定 SRAM 校验错信号连接到 TIM1/15/16/17 的刹车 (Break) 输入。 ● 0：SRAM 校验错信号断开与 TIM1/15/16/17 刹车 (Break) 输入的连接 ● 1：SRAM 校验错信号连接到 TIM1/15/16/17 刹车 (Break) 输入
位 0	LOCKUP_LOCK：Cortex-M0 LOCKUP 位使能位（Cortex-M0LOCKUP 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-M0LOCKUP（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

9 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内存到内存的高速数据传输。在不占用任何 CPU 资源的情况下，能将数据从指定的源地址快速地搬移到目标地址，以使得 CPU 有更多的资源去处理其它应用。

DMA 控制器拥有 5 个通道，每个通道都专用于管理来自于一个或多个外设的存储访问请求。同时它拥有一个仲裁器去处理不同的 DMA 请求的优先级。

9.1 DMA 的主要功能

- 多达 5 个独立的可配置通道（请求）。
- 每个通道都与专用的硬件 DMA 请求相连，同时也支持软件触发。这些配置由软件完成。
- 不同通道的 DMA 请求优先级可以由软件配置（共 4 个可配的优先等级：较高、高、中、低）；如某些通道的 DMA 请求优先级配置相同时，则由硬件来决定最终优先级（例如：同等优先级配置的情况下，通道 1 的 DMA 请求高于通道 2 的 DMA 请求）。
- 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。
- 支持内存到内存传输模式。
- 支持外设到内存、内存到外设传输模式。
- 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（或者说每个通道上 DMA 传输的次数）可配置：最多可配置到 65535。

9.1.1 DMA 结构

DMA 的结构框图如下：

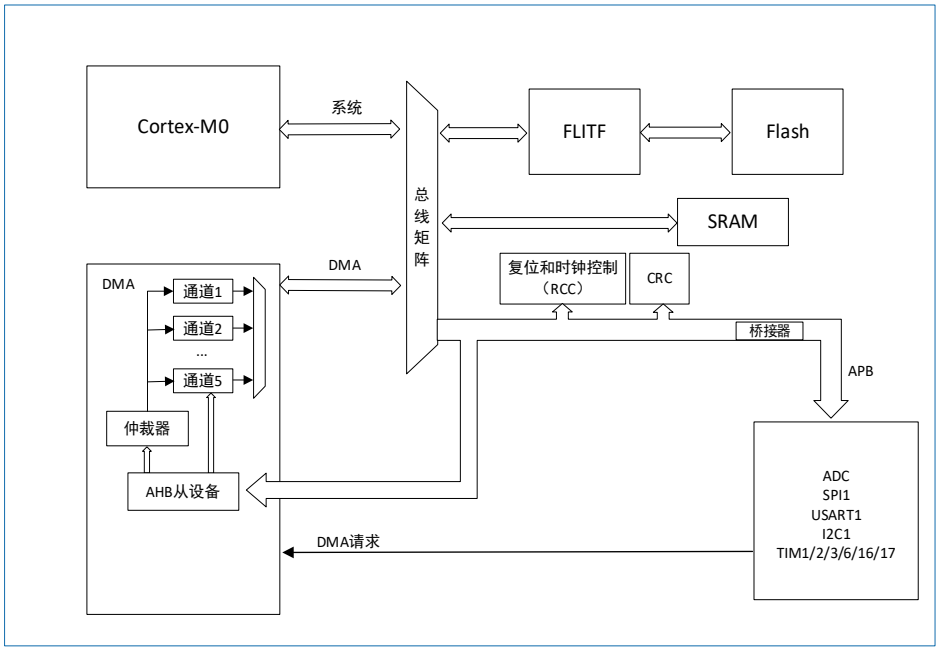


图 9-1 DMA 结构图

DMA 控制器通过与 Cortex-M0 内核共享系统总线完成直接存储数据传输功能。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 对系统总线的访问暂停几个总线周期。总线矩阵执行轮询调度，以保证 CPU 占用至少一半的系统总线带宽（包括内存和外设在内）。

9.1.2 DMA 传输

产生一次事件之后，外设会向 DMA 控制器发送一个请求信号。DMA 控制器根据对应通道的优先级处理该请求。当 DMA 控制器访问外设，DMA 控制器就会向外设发送一个应答信号。外设一旦收到来自 DMA 控制器的应答信号，就会释放当前的 DMA 请求。一旦外设释放了 DMA 请求，DMA 控制器也会释放应答信号。如果有很多的请求，外设可以发起下一次传输。

总之，每次 DMA 传输包括三个操作：

- 根据当前外设/内存地址寄存器的值，从外设的数据寄存器或者内存的指定地址加载数据。用于第一次传输的起始地址，通过对 DMA_CHPAR 或 DMA_CHMAR 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加载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_CHPAR 或 DMA_CHMAR 编程来确定的。
- 用于存剩余待传输次数的 DMA_CHDTR 寄存器递减计数。

9.1.3 仲裁器

仲裁器根据通道优先级管理通道请求并按顺序执行外设/内存访问。

DMA 通道优先级有两种:

- **软件可配的优先级**：每个 DMA 通道可通过配置 DMA_CHCR 寄存器选择以下任一种优先级：
  - 较高优先级
  - 高优先级
  - 中优先级
  - 低优先级
- **硬件默认优先级**：如果两个通道具有相同的软件优先级，则序号较小的通道具有更高的优先级。例如通道 2 比通道 4 的优先级更高。

9.1.4 DMA 通道

每个 DMA 通道都能处理固定地址的外设寄存器和内存地址之间的 DMA 传输。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寄存器就会减 1 计数。

可编程的数据传输位宽

外设和内存传输的数据宽度完全由 DMA_CHCR 寄存器中的 PSIZE 以及 MSIZE 配置决定。

指针自加

通过设置 DMA_CHCR 寄存器中的 PINC 和 MINC 标志位，外设和存储器的指针在每次传输后可以选成自动增加模式。当设置为增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决由所选的数据宽度决定，可以为 1（位宽为字节）、2（位宽为半字）或 3（位宽为字）。第一个传输的地址是存放在 DMA_CHPAR/DMA_CHMAR 寄存器中。在传输过程中，这些寄存器保持它们初始的数值，软件不能改变和读出当前正在传输的地址（它在当前的内部外设/存储器地址寄存器中）。

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_CHDTR 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_CHDTR 寄存器中，对应的 DMA 通道必须先关闭。

注意：如果一个 DMA 通道关闭了，DMA 寄存器不会被复位。DMA_CHCR，DMA_CHPAR 和 DMA_CHMAR 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_CHDTR 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将 DMA_CHPAR 和 DMA_CHMAR 寄存器中的值加载。

DMA 通道配置步骤

按照以下步骤配置一个 DMA 通道 x（x 代表通道编号）：

1. 配置 DMA_CHPAR 寄存器，以设置第一次传输的外设地址。传输时会根据该寄存器的值，将数据从该地址对应的外设寄存器中搬出或者是从内存中将数据搬到该地址对应的外设寄存器。
2. 配置 DMA_CHMAR 寄存器，以设置第一次传输的内存地址。传输时会根据该寄存器的值，将数据从内存中的该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_CHDTR 寄存器，以设置待传输的数据个数。每完成一个数据的传输，DMA_CHDTR 的值会减 1。
4. 通过配置 DMA_CHCR 寄存器中的 PL[1:0]位，以设置该通道的优先级。
5. 配置 DMA_CHCR 寄存器，以设置数据传输方向、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_CHCR 寄存器中的 EN 位激活该通道。

通道一旦被激活，就可以处理与该通道相连外设的 DMA 请求了。

当已完成的传输数量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（Circular mode）

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，如应用在 ADC 扫描模式下）。使能循环模式可以通过配置 DMA_CHCR 寄存器中的 CIRC 位为'1'来完成。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_CHDTR 的值会自动地加载初始配置值，对应的通道也会一直处理 DMA 请求。

内存到内存传输模式（Memory-to-memory mode）

DMA 通道也可在没有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_CHCR 寄存器中 MEM2MEM 位为'1'，则一旦 DMA_CHCR 中的 EN 置位，该通道上的 DMA 传输就会立即启动。当 DMA_CHDTR 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。

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 会按照下表进行数据对齐。

表 9-1 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	要传输的数据项的数目（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	1：读取 B0[7:0]@0x0，然后写入 00B0[15:0]@0x0	@0x0/00B0

源端口宽度	目标端口宽度	要传输的数据项的数目（NDT）	源内容：地址/数据	传输操作	目标内容：地址/数据
			@0x1/B1 @0x2/B2 @0x3/B3	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	@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

源端口宽度	目标端口宽度	要传输的数据项的数目（NDT）	源内容：地址/数据	传输操作	目标内容：地址/数据
32	32	4	@0x0/B3B2B1B0 @0x4/B7B6B5B4 @0x8/BBBAB9B8 @0xC/BFBEBDBC	1：读取 B3B2B1B0[31:0]@0x0，然后写入 B3B2B1B0[31:0]@0x0 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	@0x0/B3B2B1B0 @0x4/B7B6B5B4 @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 开始了一个 AHB 字节或半字写操作，则会将数据复制到 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：

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

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”。

9.1.5 错误管理

当对保留的地址空间执行读取操作时，将生成 DMA 传输错误。若在 DMA 读或写访问过程中产生了 DMA 传输错误，则会将相应的通道配置寄存器（DMA_CHCR）的 EN 位进行硬件清零，从而自动禁止出错的通道。如果 DMA_CHCR 寄存器中的传输错误中断使能位（TEIE）置 1，则 DMA_IFR 寄存器中该通道的传输错误中断标志（TEIF）将置 1 并生成中断。

9.1.6 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 9-2 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

9.1.7 DMA 请求映射

来自外设（ADC、SPI1、USART1、I2C1 和 TIM1/2/3/6/16/17）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。通过配置相应的外设寄存器中的 DMA 控制位，可单独激活或取消激活外设 DMA 请求。

下表列出了各通道的 DMA 请求：

表 9-3 每个通道的 DMA 请求一览

外设	通道 1	通道 2	通道 3	通道 4	通道 5
ADC	ADC (1)	ADC (2)	-	-	-
SPI	-	SPI1_RX	SPI1_TX	-	-
USART	-	USART1_TX (1)	USART1_RX (1)	USART1_TX (2)	USART1_RX (2)
I2C	-	I2C1_TX (1)	I2C1_RX (1)	-	-
TIM1	-	TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_CH3 TIM1_UP
TIM2	TIM2_CH3	TIM2_UP	TIM2_CH2	TIM2_CH4	TIM2_CH1
TIM3	-	TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	-
TIM6	-	-	TIM6_UP	-	-
TIM16	-	-	TIM16_CH1 (1) TIM16_UP (1)	TIM16_CH1 (2) TIM16_UP (2)	-
TIM17	TIM17_CH1 (1) TIM17_UP (1)	TIM17_CH1 (2) TIM17_UP (2)	-	-	-

(1). 只有在 SYSCFG_CFGR1 寄存器相应的重映射位清零时，该 DMA 请求映射到这个 DAM 通道。详情请参见 “8.1.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）”。

(2). 只有在 SYSCFG_CFGR1 寄存器相应的重映射位置位时，该 DMA 请求映射到这个 DMA 通道。详情请参见 “8.1.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）”。

9.2 DMA 中断寄存器

基地址：0x4002 0000

空间大小：0x400

9.2.1 DMA 中断状态寄存器（DMA_ISR）

偏移地址：0x00

复位值：0x0000 0000

DMA_ISR 寄存器仅支持读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												TEIF5	HTIF5	TCIF5	GIF5
												r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:20	Res：保留 必须保持复位值。
位 4*x-1 (x=5..1)	TEIFx：通道 x 传输错误标志（Channel x transfer error 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：通道 x 上无传输错误 • 1：通道 x 上出现传输错误
位 4*x-2 (x=5..1)	HTIFx：通道 x 半传输标志（Channel x half transfer 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：通道 x 上无半传输事件 • 1：通道 x 上发生半传输事件
位 4*x-3 (x=5..1）	TCIFx：通道 x 传输完成标志（Channel x transfer complete 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：通道 x 上无传输完成事件 • 1：通道 x 上发生传输完成事件
位 4*(x-1) (x=5..1)	GIFx：通道 x 全局中断标志（Channel x 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：通道 x 上无 TE、HT 或 TC 事件中的任何一个 • 1：通道 x 上发生了 TE、HT、或 TC 事件

9.2.2 DMA 中断标志清零寄存器（DMA_IFCR）

偏移地址：0x04

复位值：0x0000 0000

DMA_IFCR 寄存器仅支持写。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res												CTEIF5	CHTIF5	CTCIF5	CGIF5
												w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:20	Res：保留 必须保持复位值。
位 4x-1 (x=5..1)	CTEIFx：通道 x 传输错误标志清零（Channel x transfer error clear） 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 4x-2 (x=5..1)	CHTIFx：通道 x 半传输标志清零（Channel x half transfer clear） 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 4x-3 (x=5..1)	CTCIFx：通道 x 传输完成标志清零（Channel x transfer complete clear） 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 4(x-1) (x=5..1)	CGIFx：通道 x 全局中断标志清零（Channel x global interrupt clear） 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

9.3 DMA 通道寄存器

基地址：（DMA_Channel1, DMA_Channel2, DMA_Channel3, DMA_Channel4, DMA_Channel5）=（0x4002 0008, 0x4002 001C, 0x4002 0030, 0x4002 0044, 0x4002 0058）

空间大小：（DMA_Channel1, DMA_Channel2, DMA_Channel3, DMA_Channel4, DMA_Channel5）=（0x14, 0x14, 0x14, 0x14, 0x14）

9.3.1 DMA 通道配置寄存器（DMA_CHCR）

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	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 mode） • 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（Channel priority level） • 00：低优先级 • 01：中优先级 • 10：高优先级 • 11：非常高的优先级

位 11:10	MSIZE[1:0]：存储器端数据位宽（Memory 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 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（Memory increment mode） • 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（Peripheral increment mode） • 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（Circular mode） • 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（Data transfer direction） • 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（Transfer error interrupt enable） • 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（Half transfer interrupt enable） • 0：不使能半传输中断 • 1：使能半传输中断
位 1	TCIE：传输完成中断使能（Transfer complete interrupt enable） • 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） • 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）

9.3.2 DMA 通道数据寄存器（DMA_CHDTR）

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
NDT[15:0]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	NDT[15:0]：当前通道待传输数据的个数（0 至 65535）（Number of data to transfer） 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。

9.3.3 DMA 通道外设地址寄存器（DMA_CHPAR）

偏移地址：0x08

复位值：0x0000 0000

注意：在通道使能时，不能对该寄存器进行写操作。

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/写数据操作中外设数据寄存器的基地址。 • 若 PSIZE[1:0] = 01（16 位），则忽略 PA[0]位，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]位，访问将自动对齐到字地址。

9.3.4 DMA 通道 x 存储器地址寄存器（DMA_CHMAR）

偏移地址：0x0C

复位值：0x0000 0000

注意：在通道使能时，不能对该寄存器进行写操作。

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]位，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]位，访问将自动对齐到字地址。

10 嵌套向量中断控制器（NVIC）

10.1 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。

- 24 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

10.2 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 1ms 时间基准。

10.3 中断和异常向量

表 10-1 NVIC 表

位置	优先级		名称	描述	地址
-	-	-	-	保留	0X0000_0000
-	-3	固定	Reset	复位	0X0000_0004
-	-2	固定	NMI	非屏蔽中断	0X0000_0008
-	-1	固定	HardFault	所有类型错误	0X0000_000C
-	3	可配置	SVCall	通过 SWI 指令的系统服务调用	0X0000_002C
-	5	可配置	PendSV	可挂起的系统服务请求	0X0000_0038
-	6	可配置	SysTick	系统嘀嗒定时器	0X0000_003C
0	7	可配置	WWDG	窗口看门狗中断	0X0000_0040
1	8	可配置	PVD	连到 EXTI 的电源电压检测 PVD 中断（与 EXTI 线 16 共用）	0X0000_0044
2	9	可配置	RTC	RTC 全局中断（与 EXTI 线 17，19 和 20 共用）	0X0000_0048
3	10	可配置	FLASH	Flash 全局中断	0X0000_004C
4	11	可配置	RCC	RCC 全局中断	0X0000_0050
5	12	可配置	EXTI0_1	EXTI 线[1:0]中断	0X0000_0054
6	13	可配置	EXTI2_3	EXTI 线[3:2]中断	0X0000_0058
7	14	可配置	EXTI4_15	EXTI 线[15:4]中断	0X0000_005C
8	15	-	-	保留	0X0000_0060

位置	优先级		名称	描述	地址
9	16	可配置	DMA_CH1	DMA1 通道 1 全局中断	0X0000_0064
10	17	可配置	DMA_CH2_3	DMA1 通道 2 和 3 全局中断	0X0000_0068
11	18	可配置	DMA_CH4_5	DMA1 通道 4 和 5 全局中断	0X0000_006C
12	19	可配置	ADC	ADC 中断	0X0000_0070
13	20	可配置	TIM1_BRK_UP_TRG_COM	TIM1 刹车、更新、触发和 COM 中断	0X0000_0074
14	21	可配置	TIM1_CC	TIM1 捕获比较中断	0X0000_0078
15	22	可配置	TIM2	TIM2 全局中断	0X0000_007C
16	23	可配置	TIM3	TIM3 全局中断	0X0000_0080
17	24	可配置	TIM6	TIM6 全局中断	0X0000_0084
18	25	-	-	保留	0X0000_0088
19	26	可配置	TIM14	TIM14 全局中断	0X0000_008C
20	27	-	-	保留	0X0000_0090
21	28	可配置	TIM16	TIM16 全局中断	0X0000_0094
22	29	可配置	TIM17	TIM17 全局中断	0X0000_0098
23	30	可配置	I2C1	I2C1 全局中断（和 EXTI 线 23 共用）	0X0000_009C
24	31	-	-	保留	0X0000_00A0
25	32	可配置	SPI1	SPI1 全局中断	0X0000_00A4
26	33	-	-	保留	0X0000_00A8
27	34	可配置	USART1	USART1 全局中断（和 EXTI 线 25 共用）	0X0000_00AC
28	35	-	-	保留	0X0000_00B0
29	36	-	-	保留	0X0000_00B4
30	37	-	-	保留	0X0000_00B8
31	38	可配置	DVSQ	DVSQ 全局中断	0X0000_00BC

11 扩展中断和事件控制器（EXTI）

外部中断/事件控制器由 22 个能产生事件/中断请求的边沿检测器组成。每根输入线可以独立地配置类型（事件或中断）和对应的触发事件（上升沿、下降沿或双边沿触发）。每根输入线都可以单独被屏蔽。挂起寄存器保持着中断请求的状态线。

11.1 主要特性

EXTI 控制器的主要特性如下：

- 每根中断/事件线都可单独被触发和屏蔽
- 每根中断线都有专用的状态位
- 支持多达 22（16 个外部和 6 个内部）个中断/事件请求
- 检测脉冲宽度低于 APB2 时钟宽度的外部信号

11.2 框图

EXTI 结构框图如下所示：

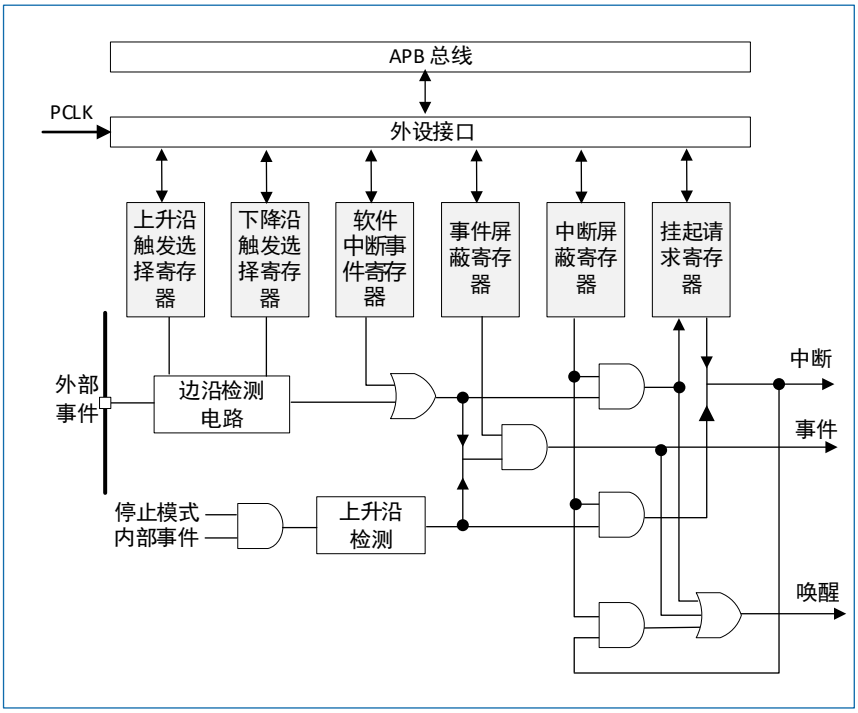


图 11-1 EXTI 框图

11.3 唤醒事件管理

HK32F031 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“11.4 功能说明”。

11.4 功能说明

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 2 个触发寄存器，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'。在挂起寄存器的对应位写'1'，将清除该中断请求。

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 2 个触发寄存器，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'。

通过软件向软件中断事件寄存器写'1'，以产生中断/事件请求。

11.4.1 硬件中断选择

配置 EXTI 线作为中断源的步骤如下：

1. 配置其中断屏蔽位（EXTI_IMR）。
2. 配置其触发选择位（EXTI_RTSR 和 EXTI_FTSR）。
3. 配置其对应到外部中断控制器（EXTI）的 NVIC 中断通道的使能和屏蔽位，以使得 EXTI 线上来的中断能正确地被响应。

11.4.2 硬件事件选择

配置 EXTI 线作为事件源的步骤如下：

1. 配置其事件屏蔽位（EXTI_EMR）。
2. 配置其触发选择位（EXTI_RTSR 和 EXTI_FTSR）。

11.4.3 软件中断/事件的选择

EXTI 线可以被配置成软件中断/事件线。下面是产生软件中断的步骤：

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

11.4.4 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）：

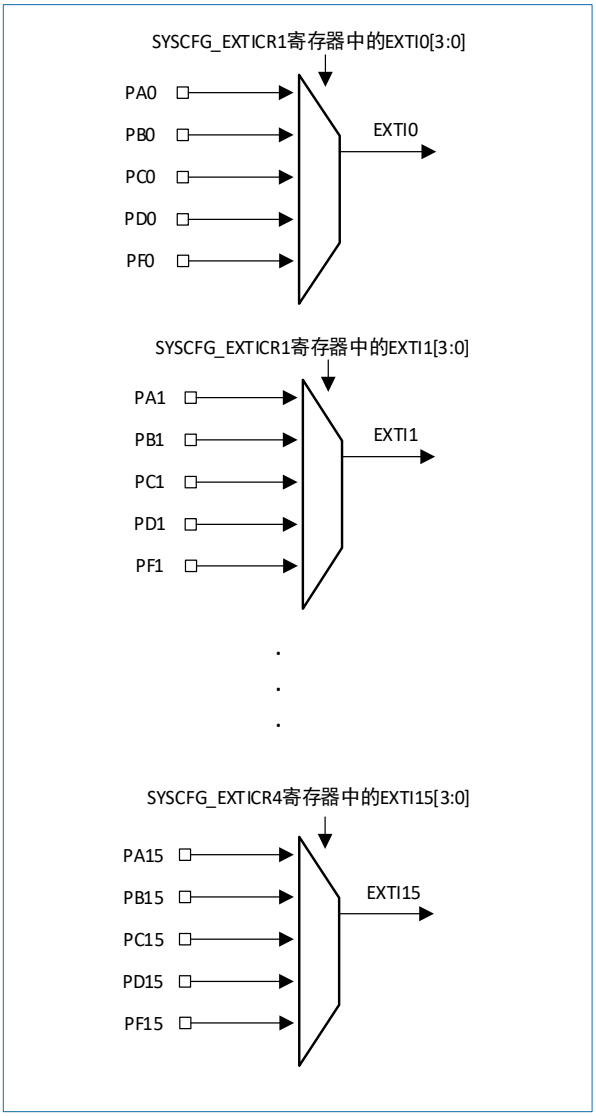


图 11-2 外部中断 GPIO 映射

上图说明：更多 HK32F031 系列 GPIO 的详细信息，请查阅数据手册中管脚定义章节。

如图 11-2 所示，通过 SYSCFG_EXTICRx 配置 GPIO 线上的外部中断/事件，除需使能 GPIO 时钟外，也需配置 RCC_APBENR2.SYSCFGEN 来使能 AFIO 时钟。参见“6.3.7 APB2 外设时钟使能寄存器（RCC_APB2ENR）”。

其余的 EXTI 口连接以下事件：

- EXTI 16 连接 PVD 输出。
- EXTI 17 连接 RTC 的报警事件。
- EXTI 19 连接 RTC 的侵入检测和时间戳事件。
- EXTI 20 连接 RTC 的唤醒事件。
- EXTI 23 连接 I2C1 的唤醒事件。
- EXTI 25 连接 USART1 的唤醒事件。

EXTI 23、25 不带 RTSR、FTSR、SWIER 和 PR 寄存器，仅能在 Stop 模式下采集事件的上升沿以产生 ERQ 和 IRQ 信号唤醒系统。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。

11.5 EXTI 寄存器

基地址：0x4001 0400

空间大小：0x400

11.5.1 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						IM25	Res	IM23	Res		IM20	IM19	Res	IM17	IM16
						rw		rw			rw	rw		rw	rw

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

位 31:26	Res：保留 必须保持复位值。
位 25	IM25：外部/内部线 25 的中断屏蔽位（interrupt mask on line 25） • 0：屏蔽来自线 25 上的中断请求 • 1：允许来自线 25 上的中断请求
位 24	Res：保留 必须保持复位值。
位 23	IM23：外部/内部线 23 的中断屏蔽位（interrupt mask on line 23） • 0：屏蔽来自线 23 上的中断请求 • 1：允许来自线 23 上的中断请求
位 22:21	Res：保留 必须保持复位值。
位 x (x = 20..19)	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 18	Res：保留 必须保持复位值。
位 x (x = 17..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

11.5.2 事件屏蔽寄存器（EXTI_EMR）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res						EM25	Res	EM23	Res		EM20	EM19	Res	EM17	EM16
						rw		rw			rw	rw		rw	rw

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

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

位 31:26	Res：保留 必须保持复位值。
位 25	EM25：外部/内部线 25 的事件屏蔽位（Event mask on line 25） • 0：屏蔽来自线 25 上的事件请求 • 1：允许来自线 25 上的事件请求
位 24	Res：保留 必须保持复位值。
位 23	EM23：外部/内部线 23 的事件屏蔽位（Event mask on line 23） • 0：屏蔽来自线 23 上的事件请求 • 1：允许来自线 23 上的事件请求
位 22:21	Res：保留 必须保持复位值。
位 x (x = 20..19)	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 18	Res：保留 必须保持复位值。
位 x (x = 17..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

11.5.3 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											RT20	RT19	Res	RT17	RT16
											rw	rw		rw	rw

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

位 31:21	Res：保留 必须保持复位值。
位 x (x = 20..19)	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 18	Res：保留 必须保持复位值。
位 x (x = 17..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）

11.5.4 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											FT20	FT19	Res	FT17	FT16
											rw	rw		rw	rw

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

位 31:21	Res：保留 必须保持复位值。
位 x (x = 20..19)	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 18	Res：保留 必须保持复位值。
位 x (x = 17..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）

11.5.5 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											SWI 20	SWI 19	Res	SWI 17	SWI 16
											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 31:21	Res：保留 必须保持复位值。
位 x	SWIx：线 x 的软件中断（Software interrupt on line x）

(x = 20..19)	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 18	Res：保留 必须保持复位值。
位 x (x = 17..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。

11.5.6 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											PIF20	PIF19	Res	PIF17	PIF16
											rw	rw		rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIF 15	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 31:21	Res：保留 必须保持复位值。
位 x (x = 20..19)	PIFx：线 x 挂起位（Pending interrupt flag on line x） • 0：没有发生触发请求 • 1：发生了选择事件的触发请求。当外部中断线上发生了所选择的边沿事件，该位被置 1。 向该位写入 1 或改变边沿检测的极性进行清除。
位 18	Res：保留 必须保持复位值。
位 x (x = 17..0)	PIFx：线 x 挂起位（Pending interrupt flag on line x） • 0：没有发生触发请求 • 1：发生了选择事件的触发请求。当外部中断线上发生了所选择的边沿事件，该位被置 1。 向该位写入 1 或改变边沿检测的极性进行清除。

12 模拟数字转换器（ADC）

该系列芯片内置的 12 位 ADC 是逐次趋近型模数转换器。它具有多达 18 个复用通道，可测量来自 16 个外部和 2 个内部源的信号。各个通道的 A/D 转换支持单次、连续、扫描或不连续采样模式。ADC 的结果可以左对齐或右对齐的存储在一个 16 位数据寄存器中。

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

12.1 ADC 主要特性

- 高性能
  - 可配置 12 位、10 位、8 位或 6 位分辨率。
  - ADC 转换时间：12 位分辨率对应的转换时间为 1 μs（1MHz）；10 位分辨率对应的转换时间为 0.93 μs；若降低分辨率，可进一步缩短转换时间。
  - 自校准
  - 可编程采样时间
  - 内置数据一致性，可确保数据对齐。
  - 支持 DMA
- 低功耗
  - 应用可降低 PCLK 频率从而以低功耗运行，同时仍可保持最优的 ADC 性能。例如，无论 PCLK 的频率如何，都保持 1.0 μs 的转换时间。
  - 等待模式：防止 ADC 在低频 PCLK 应用中溢出。
  - 自动关闭模式：ADC 会自动断电，但正在进行转换时除外。这可大幅降低 ADC 的功耗。
- 模拟输入通道
  - 16 条外部模拟输入
  - 1 条用于内部温度传感器（VSENSE）的通道
  - 1 条用于内部参考电压（VREFINT）的通道
- 可通过以下方式启动 A/D 转换：
  - 软件
  - 极性可配置的硬件触发器（来自 TIM1、TIM2、TIM3 的内部定时器事件，GPIO 输入事件）
- 转换模式
  - 可转换单条通道，也可扫描一系列通道。
  - 单次模式：会在每次触发时对选定的输入执行一次转换
  - 连续模式：可连续转换选定的输入
  - 不连续转换模式
- 在采样结束、转换结束、序列转换结束、发生模拟看门狗事件或溢出事件时产生中断。
- 带模拟看门狗
- ADC 电源要求：2.0V～5.5V
- ADC 输入范围：$V_{SSA} \le V_{IN} \le V_{DDA}$

12.2 ADC 功能描述

ADC 功能框图如下：

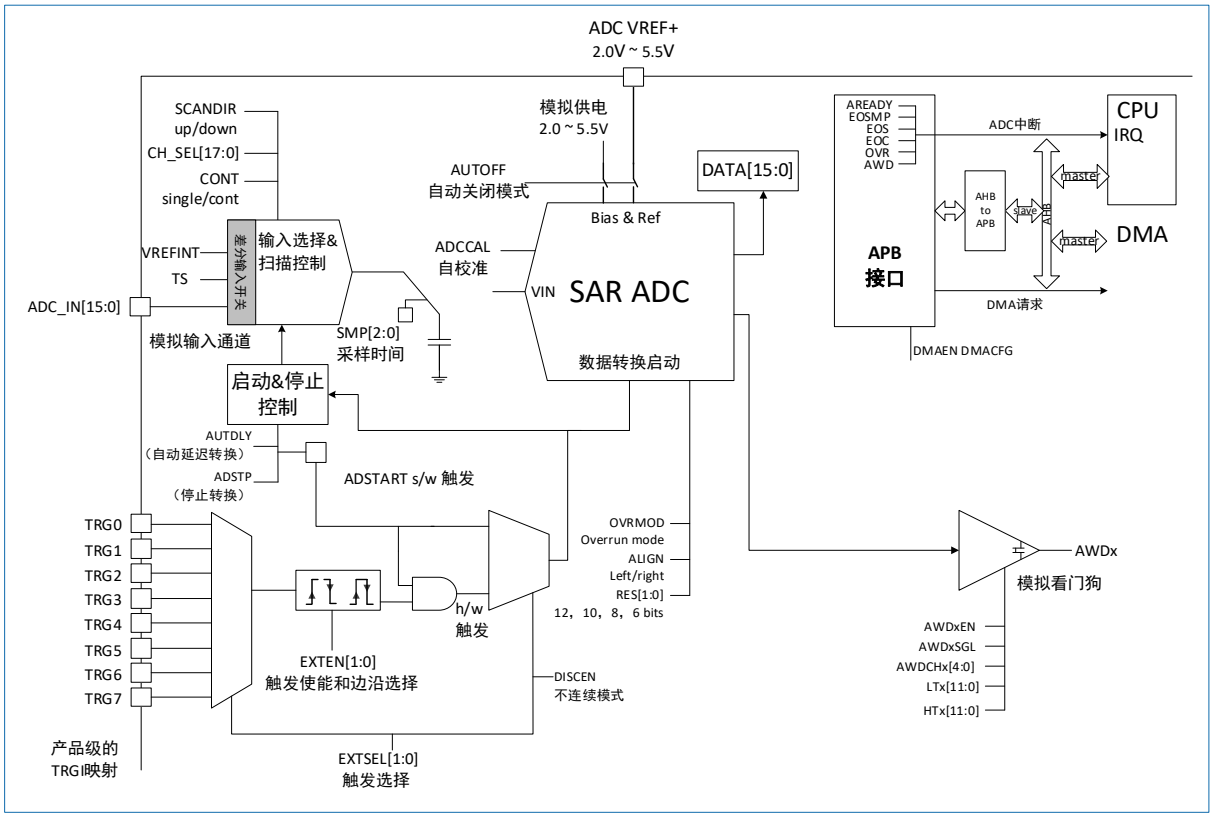


图 12-1 ADC 功能框图

12.2.1 ADC 引脚和内部信号

表 12-1 ADC 引脚

名称	信号类型	备注
V_{DDA}	模拟电源输入	ADC 的模拟电源和正参考电压（$V_{DDA} \geqslant V_{DD}$）
V_{SSA}	模拟电源地输入	模拟电源接地引脚。电压必须等于 V_{SS}
ADC_IN[15:0]	模拟输入信号	16 个模拟输入通道

表 12-2 ADC 内部信号

内部信号名称	信号类型	说明
TRGx	输入	ADC 转换触发信号
VSENSE	输入	内部温度传感器输出电压
VREFINT	输入	内部参考输出电压

12.2.2 校准（ADCAL）

ADC 有一个内置自校准模式。校准可显著减小因内部电容器组的变化而造成的精度误差由于制造工艺的不同，各芯片的偏移误差也有所不同。

校准过程中，ADC 会计算校准系数。ADC 下一次掉电之前，会在内部应用此校准系数。校准过程中，不得使用 ADC，必须等待校准完成才行。

校准应在启动 A/D 转换之前进行。仅当 ADC 禁用（ADEN=0）后，才能发起校准。通过软件将 ADCAL 位置 1，以启动校准。ADCAL 位在所有校准序列过程中保持为 1。校准完成后，此位会立即由硬件清零。

随后，可从 ADC_DR 寄存器（位 5 到位 0）中读取校准系数。

若禁止了 ADC（ADEN=0），则会保留内部模拟校准。如果 ADC 工作条件发生变化（VDDA 变化是造成 ADC 偏移变化的主要原因，并会导致温度发生小幅变化），建议重新运行校准。

在下列情况下，校准系数会丢失：

- MCU 处于待机模式（ADC 掉电）。
- ADC 外设复位。

在睡眠和停机这两种低功耗模式下，会保留校准系数。仍可通过软件保存并恢复校准系数，以节省 ADC 重启时间（前提是 ADC 掉电期间的温度和电压保持稳定）。

如果 ADC 已使能但未进行转换（ADEN=1 且 ADSTART=0），可写入校准系数。随后，下次转换启动时，校准系数会自动添加到模拟 ADC 中。这一载入过程是透明的，不会对转换的启动造成延迟。

校准软件程序

1. 确保 ADEN=0。
2. 将 ADCAL 置 1。
3. 等待直到 ADCAL=0。
4. 校准系数可从 ADC_DR 寄存器，或校准值地址 0x400127f4 的位[6:0]读取。

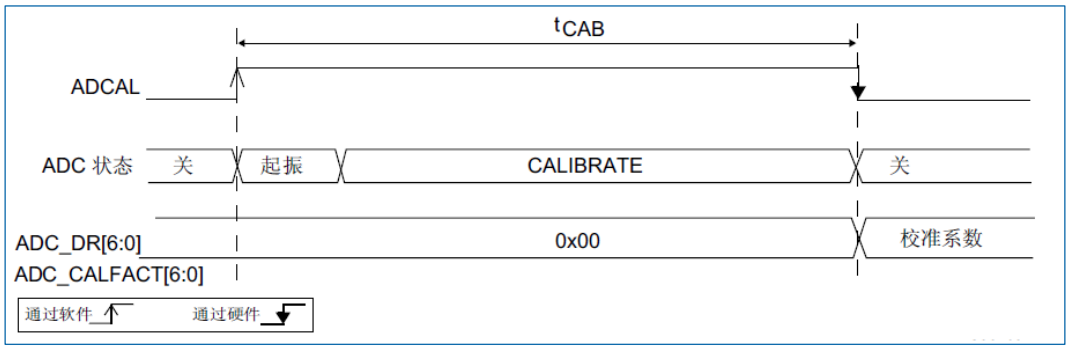


图 12-2 ADC 校准

校准系数强制软件程序

1. 确保 ADEN= 1 且 ADSTART =0（ADC 启动时没有进行任何转换）。
2. 将已保存的校准系数写入校准值地址 0x4001 27F4 的[6:0]位。
3. 启动新的转换后，将立即使用该校准系数。

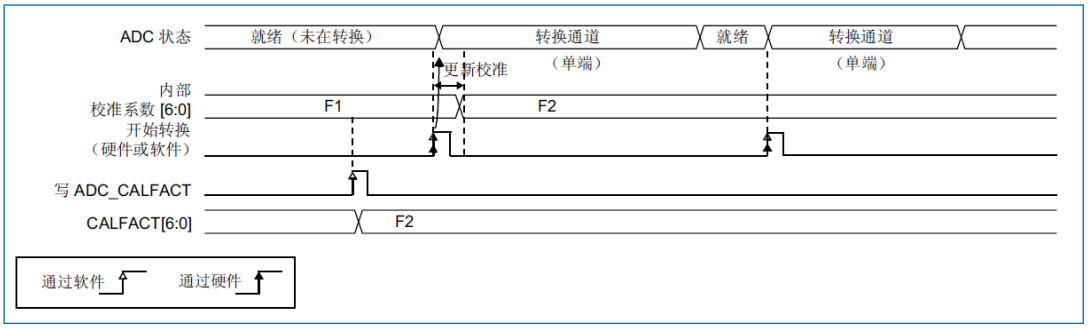


图 12-3 校准系数强制

12.2.3 ADC 开关控制（ADEN, ADDIS, ADRDY）

MCU 上电时，ADC 被禁用并进入掉电模式（ADEN=0）。

如图 12-4 所示，ADC 在开始精确转换之前需要一段稳定时间 t_{STAB}。

以下两个控制位可用于使能或禁止 ADC：

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

可通过将 ADSTART 置 1（请参见“12.3 外部触发转换和触发极性（EXTSEL, EXTEN）”）开始进行转换；如果触发器已使能，也可在发生外部触发事件时开始进行转换。

使能 ADC 的流程如下：

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

禁用 ADC 的流程如下：

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

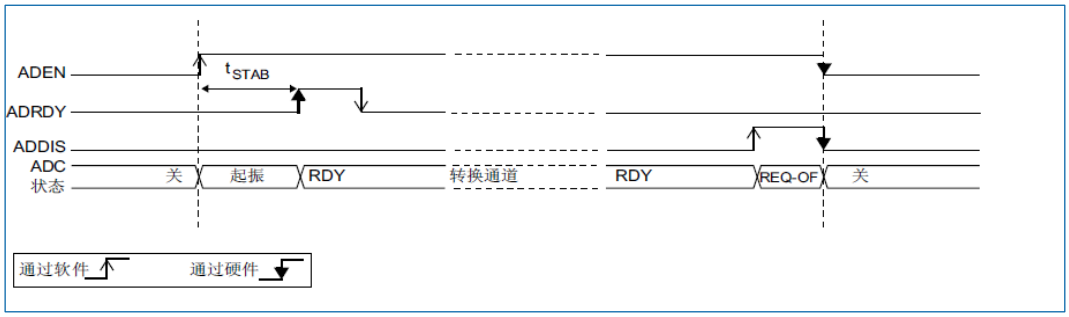


图 12-4 使能/禁用 ADC

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

12.2.4 ADC 时钟（CKMODE）

ADC 采用双时钟域架构，因此，ADC 可由独立于 APB 时钟（PCLK）的时钟提供时钟（ADC 异步时钟）。

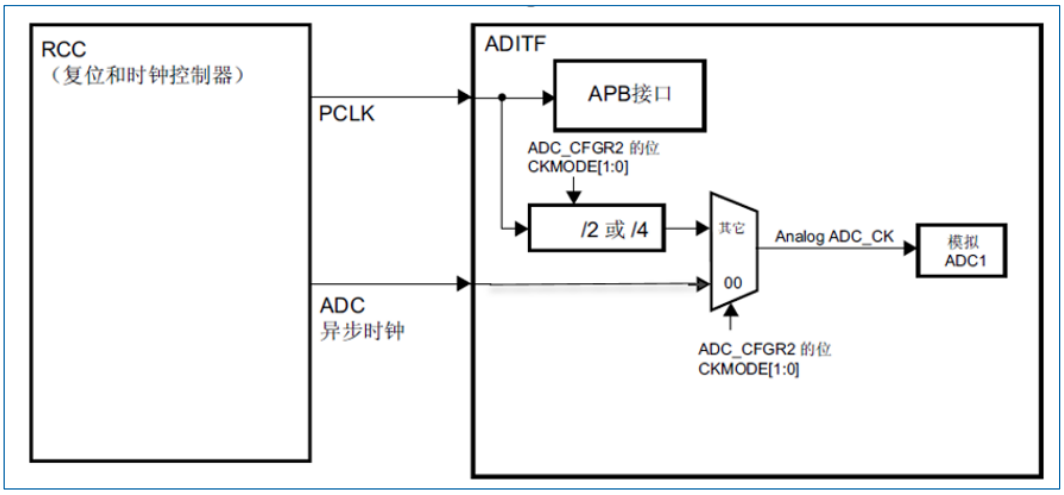


图 12-5 ADC 时钟图

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

如图 12-5，ADC 的输入时钟可在两个不同的时钟源之间进行选择：

- ADC 时钟可选择名为“ADC 异步时钟”的特定时钟源，该时钟源独立于 APB 时钟，并与 APB 时钟异步。

  更多有关生成该时钟源的信息，参见“6 复位和时钟控制（RCC）”。

  要选择该时钟，必须将 ADC_CFGR2 寄存器的 CKMODE[1:0]位复位。

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

- ADC 时钟可由 ADC 总线接口的 APB 时钟除以一个可编程因子（2 或 4，由 CKMODE[1:0]位而定）来提供。

  要选择该时钟，ADC_CFGR2 寄存器的 CKMODE[1:0]位不得为“00”。

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

表 12-3 触发与转换开始之间的延迟

ADC 时钟源	CKMODE[1:0]	触发事件与转换开始之间的延迟
HSI 14MHz 时钟	00	延迟是不确定的（存在抖动）
PCLK 2 分频	01	延迟是确定的（无抖动），等于 4.25 个 ADC 时钟周期
PCLK 4 分频	10	延迟是确定的（无抖动），等于 4.125 个 ADC 时钟周期
保留	11	保留

12.2.5 配置 ADC

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

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

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

如果 ADC 已使能（可能正在进行转换）、并且没有待处理的禁用 ADC 的请求时（ADSTART=1 且 ADDIS=0），

软件才能写 ADC_CR 寄存器中的 ADSTP 位才能停止 ADC。

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 ADEN 以及 ADC_CR 寄存器中的所有位都清零）。

12.2.6 通道选择

复用通道多达 18 条：

- 16 路来自 GPIO 引脚（ADC_IN0~ADC_IN15）的模拟输入。
- 2 路内部模拟输入（温度传感器、内部参考电压、电池电压）。

可转换单条通道，也可以自动扫描一系列通道。

待转换通道的顺序必须在 ADC_CHSELR 通道选择寄存器中进行编程，每条模拟输入通道都有专用的选择位（CHSEL0~CHSEL17）。

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

- SCANDIR=0：正向扫描通道 0 到通道 17
- SCANDIR=1：反向扫描通道 17 到通道 0

温度传感器和 V_{REFINT} 内部通道

温度传感器连接至通道 ADC_IN16。内部参考电压 V_{REFINT} 连接至通道 ADC_IN17。

12.2.7 可编程采样时间（SMP）

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

使用可编程采样时间后，可根据输入电压源的输入电阻调整转换速度。

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

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

总转换时间的计算公式如下：

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

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

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

ADC 通过将 EOSMP 标志置 1 来指示采样阶段结束。

12.2.8 单次转换模式（CONT=0）

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

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

在序列中，每次转换完成后：

- 转换后的数据会存储在 16 位 ADC_DR 寄存器中
- EOC（转换结束）标志置 1

- EOCIE 位置 1 时将产生中断

转换序列完成后：

- EOS（序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

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

注意：要转换单个通道，可将序列长度编程为 1。

12.2.9 连续转换模式（CONT=1）

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

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

在序列中，每次转换完成后：

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

转换序列完成后：

- EOS（序列结束）标志置 1
- EOSIE 位置 1 时将产生中断

随后，会立即重启新序列，ADC 会继续重复执行转换序列。

注意：要转换单个通道，可将序列长度编程为 1。

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

12.2.10 开始转换（ADSTART）

软件通过将 ADSTART 置 1 的方式开始进行 ADC 转换。

ADSTART 置 1 后：

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

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

ADSTART 位由硬件清零：

- 在使用软件触发的单次模式下（CONT=0 且 EXTEN=00）
  - 只要转换序列结束（EOS=1）就清零
- 在使用软件触发的不连续模式下（CONT=0、DISCEN=1 且 EXTEN=00）
  - 转换结束时（EOC=1）清零
- 在所有情况下（CONT=X、EXTEN=XX）
  - 执行由软件调用的 ADSTP 程序之后（参见“12.2.12 停止正在进行的转换（ADSTP）”）清零。

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

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

12.2.11 时序

从转换开始到转换结束所经过的时间是配置的采样时间与逐次趋近时间（具体视数据分辨率而定）的总和：

$t_{ADC}=t_{SMPL}+t_{SAR}=[\ 1.5_{|min} + 12.5_{|12bit}\ ]x\ t_{ADC\_CLK}$

$t_{ADC}=t_{SMPL}+t_{SAR}=93.8ns_{|min} + 781.3ns_{|12bit} = 0.875\mu s_{|min}$（对于 f_{ADC_CLK} = 16MHz）

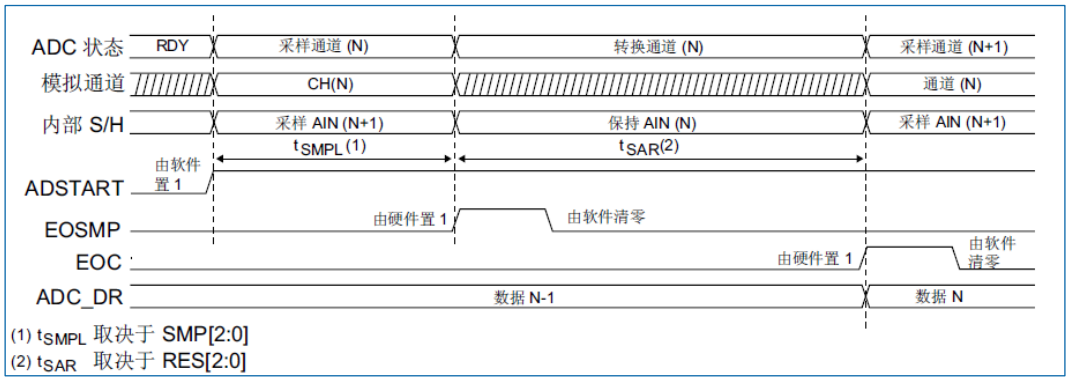


图 12-6 模数转换时间

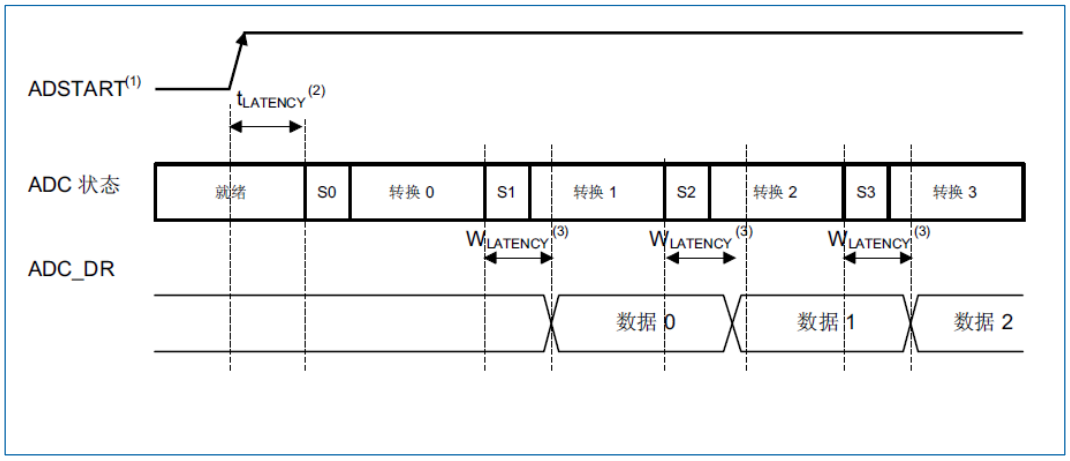


图 12-7 模数转换时序

(1). EXTEN = 00 或 EXTEN ≠ 00

(2). 触发延迟（更多详细信息，请参见数据手册）

(3). ADC_DR 寄存器写入延迟（更多详细信息，请参见数据手册）

12.2.12 停止正在进行的转换（ADSTP）

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

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

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

图 12-8 停止正在进行的转换

12.3 外部触发转换和触发极性（EXTSEL, EXTEN）

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

在转换进行时发生的硬件触发会被忽略。

如果位 ADSTART=0，则会忽略发生的任何硬件触发。

表 12-4 提供 EXTEN[1:0]值与触发极性之间的对应关系。

表 12-4 配置触发极性

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

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

表 12-5 给出了可用于常规转换的外部触发。EXTSEL[2:0]控制位用于选择 8 个可能的事件中哪一事件可触发转换。

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

表 12-5 外部触发器

名称	源	EXTSEL[2:0]
TRG0	TIM1_TRGO	000
TRG1	TIM1_CC4	001
TRG2	TIM2_TRGO	010
TRG3	TIM2_CH4	011
TRG4	TIM1_CC2	100
TRG5	TIM2_CH3	101
TRG6	TIM3_TRGO	110
TRG7	保留	111

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

12.3.1 不连续模式（DISCEN）

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

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

示例：

- DISCEN = 1，待转换通道=0、3、7、10
  - 第一次触发：转换通道 0 并生成 EOC 事件
  - 第二次触发：转换通道 3 并生成 EOC 事件
  - 第三次触发：转换通道 7 并生成 EOC 事件
  - 第四次触发：转换通道 10 并同时生成 EOC 和 EOS 事件
  - 第五次触发：转换通道 0 并生成 EOC 事件
  - 第六次触发：转换通道 3 并生成 EOC 事件
- DISCEN = 0，待转换通道=0、3、7、10
  - 第一次触发：转换整个序列：通道 0，然后是通道 3、7 和 10。每次转换都会生成 EOC 事件，最后一次转换还会生成 EOS 事件。
  - 任何后续触发事件都将重启整个序列。

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

12.3.2 可编程分辨率（RES）快速转换模式

可通过降低 ADC 分辨率缩短转换时间（t_{SAR}）。

通过对 ADC_CFGR1 寄存器中的 RES[1:0]位进行编程，可将分辨率配置为 12 位、10 位、8 位或 6 位。对于不要求使用高数据精度的应用，分辨率越低，转换时间越短。

注意：RES[1:0]位必须在 ADEN 位复位后才能进行更改。

转换结果的宽度始终为 12 位，任何未使用的 LSB 位都会读为零。

降低分辨率可缩短逐次趋近步骤所需的转换时间，如表 12-6 所示。

表 12-6 t_{SAR} 与转换分辨率有关的转换时间

RES[1:0]位	t_{SAR}（ADC 时钟周期）	t_{SAR}（ns）f_{ADC}=16MHz	t_{SMPL}（min）（ADC 时钟周期）	t_{CONV}（ADC 时钟周期）（使用最短 t_{SMPL}）	t_{CONV}（ns），f_{ADC}=16MHz
12	12.5	781ns	1.5	14	875ns
10	11.5	719ns	1.5	13	812ns
8	9.5	594ns	1.5	11	688ns
6	7.5	469ns	1.5	9	562ns

12.3.3 转换结束、采样阶段结束（EOC, EOSMP 标志）

ADC 指示每个转换结束（EOC）事件。

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

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

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

12.3.4 转换序列结束（EOS 标志）

每次序列（EOS）事件结束时，ADC 都会通知应用。

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

12.3.5 时序图示例（单次/连续模式硬件/软件触发）

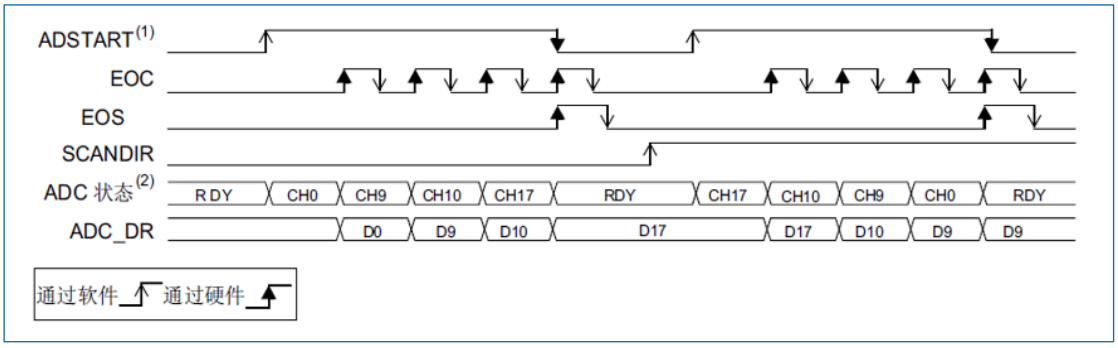


图 12-9 单次序列转换，软件触发

上图的说明：

（1）EXTEN=00，CONT=0

（2）CHSEL=0x20601，AUTDLY=0，AUTOFF=0

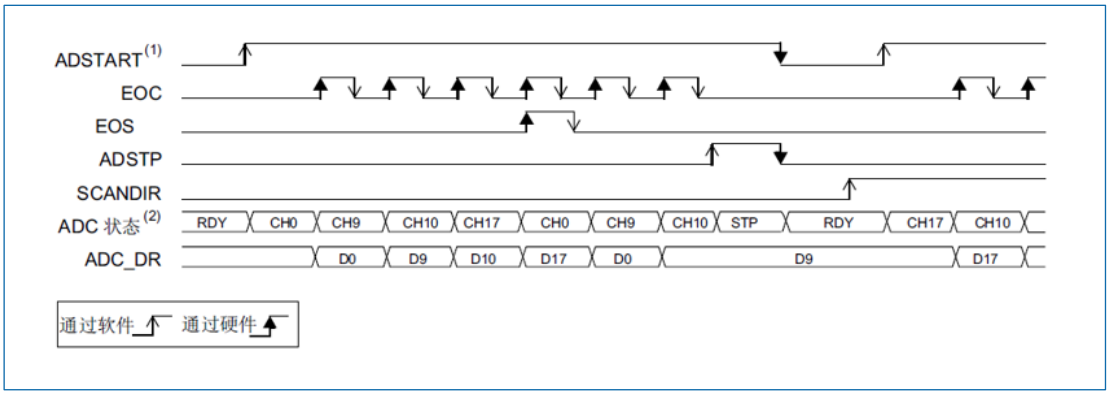


图 12-10 连续序列转换，软件触发

上图的说明：

（1）EXTEN=00，CONT=1

（2）CHSEL=0x20601，AUTDLY=0，AUTOFF=0

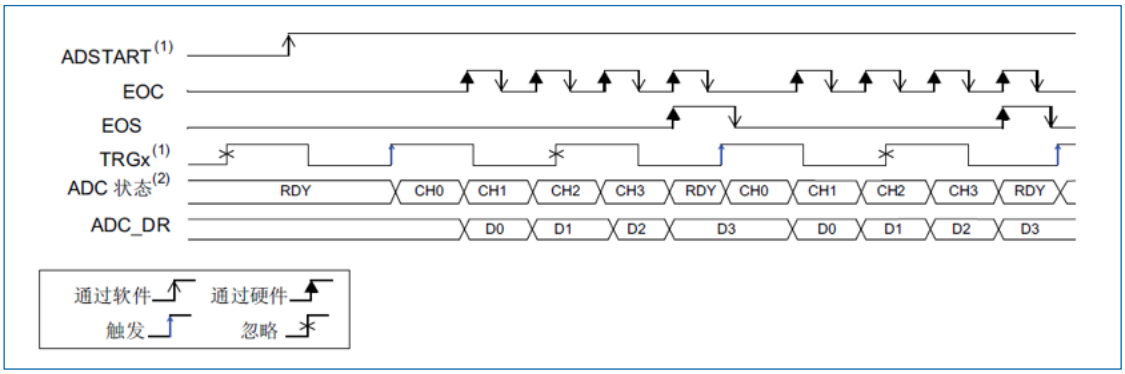


图 12-11 单次序列转换，硬件触发

上图的说明：

（1）EXTSEL=TRGx（过频），EXTEN=01（上升沿），CONT=0

（2）CHSEL=0xF，SCANDIR=0，AUTDLY=0，AUTOFF=0

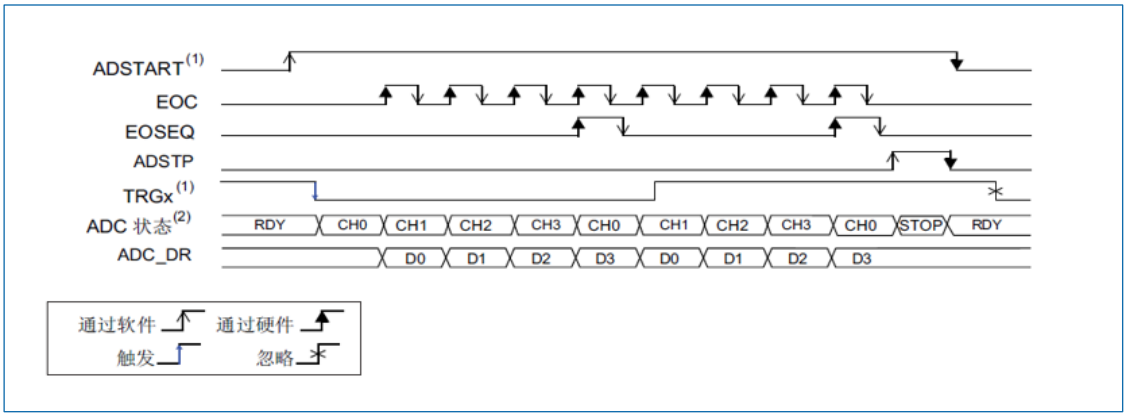


图 12-12 连续序列转换，硬件触发

上图的说明：

（1）EXTSEL=TRGx，EXTEN=10（下降沿），CONT=1

（2）CHSEL=0xF，SCANDIR=0，AUTDLY=0，AUTOFF=0

12.4 数据管理

12.4.1 数据管理和数据对齐（ADC_DR, ALIGN）

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

ADC_DR 的格式取决于配置的数据对齐方式和分辨率。

ADC_CFGR1 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。数据可右对齐（ALIGN=0）或左对齐（ALIGN=1），如图 12-13 所示。

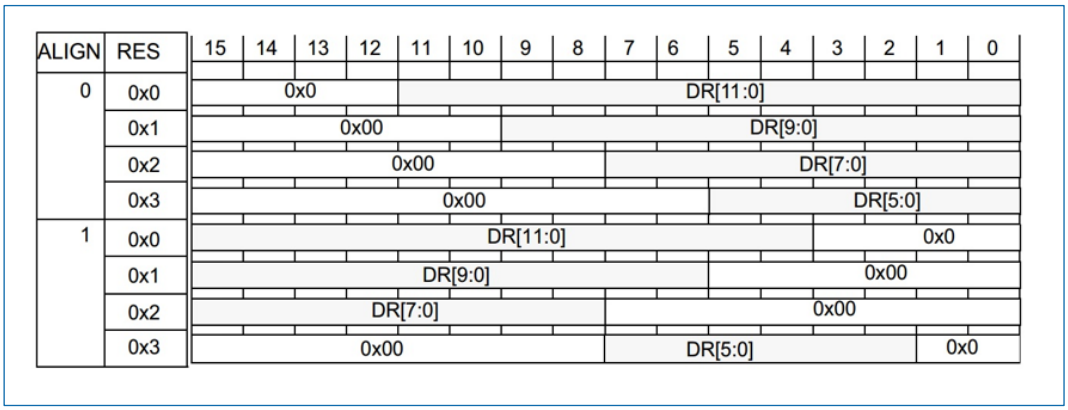


图 12-13 数据对齐方式和分辨率

12.4.2 ADC 溢出（OVR, OVRMOD）

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

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

如果 ADC_IER 寄存器中的 OVRIE 位置 1，可产生中断。

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

OVR 标志可通过软件向其写入 1 的方式来清零。

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

- OVRMOD=0
  - 溢出事件会保留数据寄存器的数据，防止其被覆盖：会保留原数据，并会丢弃新的转换结果。如果 OVR 保持为 1，可继续进行转换，但会丢弃所得的数据。
- OVRMOD=1
  - 数据寄存器会将上一次转换结果覆盖，之前未读取的数据会丢失。如果 OVR 保持为 1，可继续进行转换，ADC_DR 寄存器始终包含最新转换得出的数据。

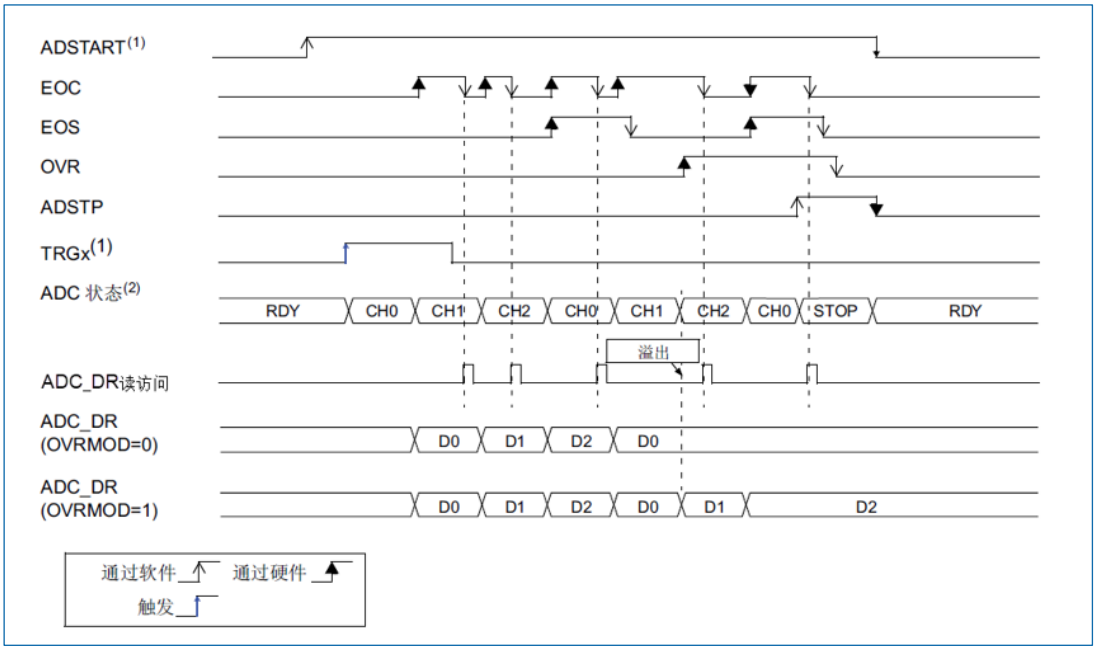


图 12-14 溢出示例（OVR）

12.4.3 在不使用 DMA 的情况下管理转换的数据序列

如果转换过程足够慢，则可使用软件来处理转换序列。在这种情况下，软件必须使用 EOC 标志及其相关中断来处理各个数据结果。每次转换完成时，都会将 ADC_ISR 寄存器中的 EOC 位置 1，并可读取 ADC_DR 寄存器。ADC_CFGR1 寄存器中的 OVRMOD 位应配置为 0，以便将溢出事件作为错误进行管理。

12.4.4 在不使用 DMA 且不发生溢出的情况下管理转换的数据

ADC 在转换一条或多条通道时，无需在每次转换后都读取数据是很有用的。在这种情况下，OVRMOD 位必须配置为 1，OVR 标志应被软件忽略。如果 OVRMOD=1，溢出事件不会阻止 ADC 继续进行转换，且 ADC_DR 寄存器始终包含最新的转换数据。

12.4.5 使用 DMA 管理转换的数据

由于转换得出的所有通道值都存储在单个数据寄存器中，因此，在转换多条通道时使用 DMA 可提高效率。这样可以避免存储在 ADC_DR 寄存器中的转换数据结果丢失。

若 DMA 模式已使能（ADC_CFGR1 寄存器中的 DMAEN 位置 1），则会在每个通道转换后生成 DMA 请求。这样便可将转换后的数据从 ADC_DR 寄存器传输到由软件选择的目标位置。

注意：ADC 校准阶段完成后，必须将 ADC_CFGR1 寄存器中的 DMAEN 位置 1。

尽管如此，如果因 DMA 无法及时处理 DMA 传输请求而发生溢出（OVR=1），ADC 会停止生成 DMA 请求，新转换的数据不会通过 DMA 进行传输。这意味着可将传输到 RAM 的所有数据都视为有效数据。

根据 OVRMOD 位的配置，可保留或覆盖数据（参见“12.4.2 ADC 溢出（OVR, OVRMOD）”）。

DMA 传输请求会被禁止，直至软件将 OVR 位清零。

根据应用用途的不同，推荐使用两种不同的 DMA 模式，并使用 ADC_CFGR1 寄存器中的 DMACFG 位配置相应的模式：

- DMA 单次模式（DMACFG=0）。

如果通过编程将 DMA 设置为传输固定数目的数据字，应选择此模式。

- DMA 循环模式（DMACFG=1）

如果通过编程将 DMA 设置为循环模式或双缓冲区模式，应选择此模式。

DMA 单次模式（DMACFG=0）

在该模式下，每次出现新的转换数据字时，ADC 都会生成 DMA 传输请求，DMA 到达最后一个 DMA 传输操作时（发生 DMA_EOT 中断，请参见“9 直接存储器访问控制器（DMA）”），即使转换已再次开始，ADC 也会停止生成 DMA 请求。

DMA 传输完成后（在 DMA 控制器中配置的所有传输操作均已完成后）：

- ADC 数据寄存器的内容会冻结。
- 任何正在进行的转换都会中止，其部分结果会被丢弃。
- 不会将任何新的 DMA 请求发送到 DMA 控制器。如果仍存在已开始的转换，这样可避免生成溢出错误。
- 扫描序列会停止并复位。
- DMA 会停止。

DMA 循环模式（DMACFG=1）

在该模式下，每次数据寄存器中出现新的转换数据字时，ADC 都会生成 DMA 传输请求，即使 DMA 已到达最后一次 DMA 传输操作也不例外。这样可将 DMA 配置为循环模式，从而处理连续的模拟输入数

据流。

12.5 功耗特性

12.5.1 自动延迟转换模式

自动延迟转换模式可用于简化软件，并可优化采用低频时钟的应用（此类应用可能存在 ADC 溢出的风险）的性能。

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

通过这种方式，可自动调整 ADC 的速度，使其适应系统读取数据的速度。

注意：转换进行时发生的或在读访问之前的等待时间内发生的硬件触发会被忽略。

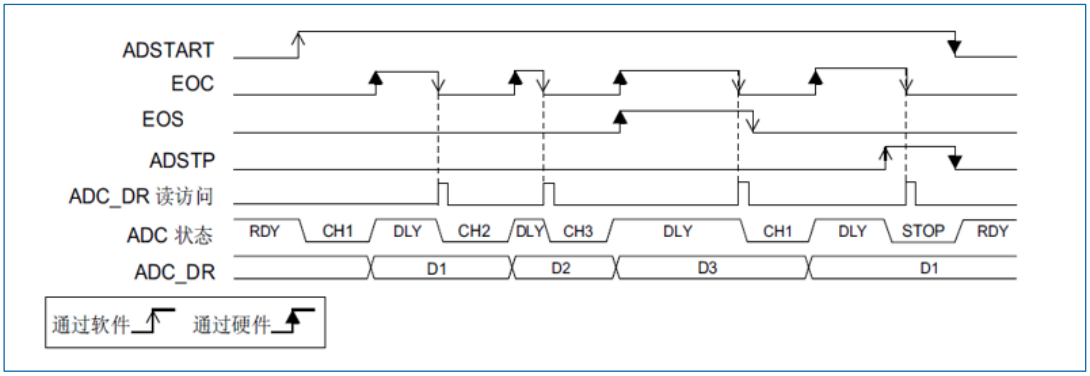


图 12-15 等待模式转换（连续模式，软件触发）

上图说明：

(1) EXTEN=00，CONT=1

(2) CHSEL=0x3，SCANDIR=0，AUTDLY=1，AUTOFF=0

12.5.2 自动关闭模式（AUTOFF）

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

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

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

对于采用低频时钟的应用，可将自动关闭模式与等待模式转换（AUTDLY=1）结合使用，如果 ADC 在等待过程中自动掉电，将会在应用读取 ADC_DR 寄存器后立即重启。这种组合可显著降低功耗（请参见图 12-17）。

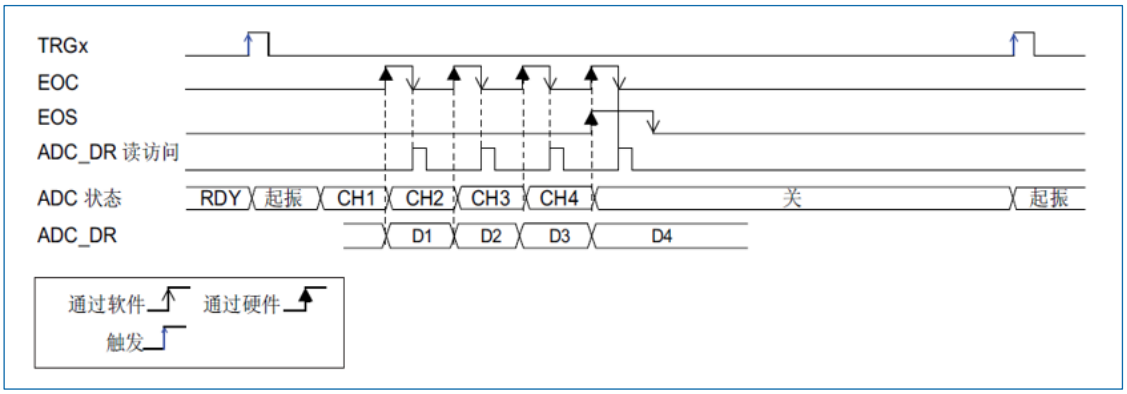


图 12-16 AUTDLY=0、AUTOFF=1 时的行为

上图说明：EXTSEL=TRGx，EXTEN=01（上升沿），CONT=x，ADSTART=1，CHSEL=0xF，SCANDIR=0，AUTDLY=1，AUTOFF=1

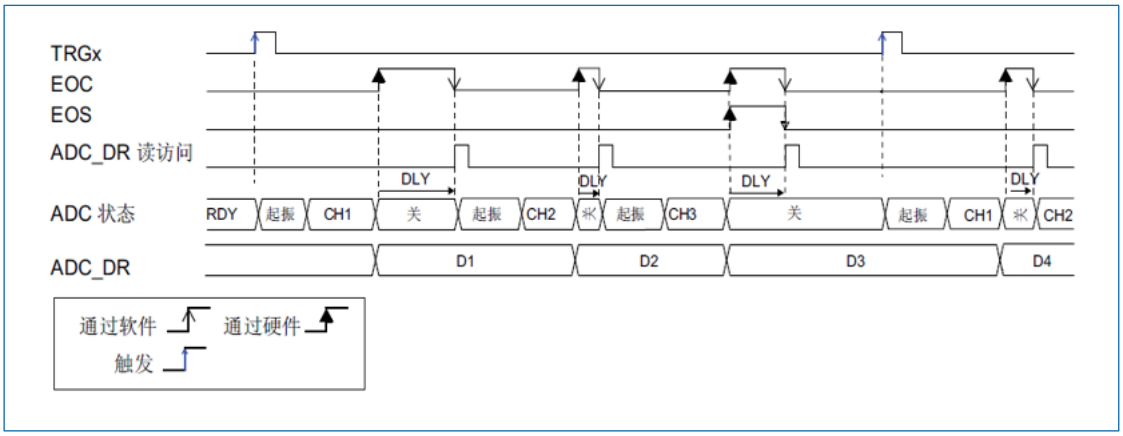


图 12-17 AUTDLY=1、AUTOFF=1 时的行为

上图说明：EXTSEL=TRGx，EXTEN=01（上升沿），CONT=x，ADSTART=1，CHSEL=0xF，SCANDIR=0，AUTDLY=1，AUTOFF=1

12.6 模拟窗口看门狗（AWDEN, AWDSGL, AWDCH, AWD_HTR/LTR, AWD）

将 ADC_CFGR1 寄存器中的 AWDEN 位置 1，可使能 AWD 模拟看门狗功能。此功能用于监控一条选定的通道或所有已使能的通道（参见表 12-8）是否仍处于所配置的电压范围（窗口）内，如图 12-18 所示。

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

AWD 标志可通过软件向其写入 1 的方式来清零。

如果转换的数据分辨率小于 12 位（取决于 RES[1:0]位），由于内部比较通常会基于 12 位原始转换数据执行（左对齐），因此被编程阈值的 LSB 必须保持清零状态。

表 12-7 介绍了如何对所有可能的分辨率进行比较。

表 12-7 模拟看门狗比较

分辨率位 RES[1:0]	模拟看门狗比较对象		备注
	原始转换数据，左对齐 (1)	阈值	
00：12 位	DATA[11:0]	LT[11:0]和 HT[11:0]	-
01：10 位	DATA[11:2]，00	LT[11:0]和 HT[11:0]	用户必须将 LT[1:0]和 HT[1:0]配置为“00”
10：8 位	DATA[11:4]，0000	LT[11:0]和 HT[11:0]	用户必须将 LT[3:0]和 HT[3:0]配置为“0000”

分辨率位 RES[1:0]	模拟看门狗比较对象		备注
	原始转换数据，左对齐 (1)	阈值	
11：6 位	DATA[11:6]，000000	LT[11:0]和 HT[11:0]	用户必须将 LT[5:0]和 HT[5:0]配置为“000000”

(1). 进行任何对齐计算之前，会对原始转换数据进行看门狗比较。

表 12-8 介绍了如何配置 ADC_CFGR1 寄存器中的 AWDSGL 位和 AWDEN 位，以使能一条或多条通道上的模拟看门狗。

表 12-8 模拟看门狗通道选择

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

(1). 通过 AWDCH[4:0]位选择。

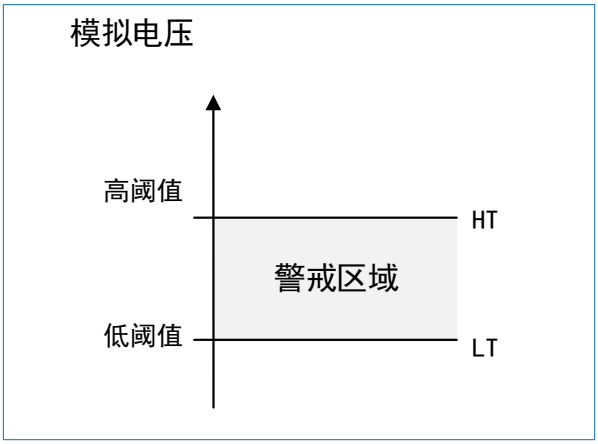


图 12-18 模拟看门狗的保护区域

12.7 温度传感器

温度传感器可用于测量器件的结温（TJ）。温度传感器在内部连接到 ADC_IN16 输入通道，该通道用于将传感器输出电压转换为数字值。温度传感器模拟引脚的采样时间必须大于数据手册中指定的最小 TS_temp 值。不使用时可将传感器置于掉电模式。

温度传感器的输出电压随温度线性变化。由于工艺不同，该线的偏移量取决于各个芯片（芯片之间的温度变化可达 45°C）。

未校准的内部温度传感器更适用于对温度变量而非绝对温度进行测量的应用。为提高温度传感器测量的准确性，航顺在生产过程中将校准值存储在每个器件的系统存储器中，见下表。访问模式为只读。用户应用可读取并使用这些数据提高温度传感器的准确性。

表 12-9 温度传感器校准值

校准值名称	描述	存储器地址
TS_CAL1	在 25°C 温度下获得的 ADC 原始数据，V_{DDA} = 3.3V	0x1FFF F7B8-0x1FFF F7B9
TS_CAL2	在 85°C 温度下获得的 ADC 原始数据，V_{DDA} = 3.3V	0x1FFF F7C2-0x1FFF F7C3

图 12-19 显示的是温度传感器与 ADC 之间连接的方框图。

必须将 TSEN 位置 1 才能使能 ADC_IN16（温度传感器）的转换，必须将 VREFEN 位置 1 才能使能 ADC_IN17（V_{REFINT}）的转换。

主要特性

- 支持的温度范围：-40°C 到 125°C
- 线性度：最高±2°C，精度取决于校准情况。

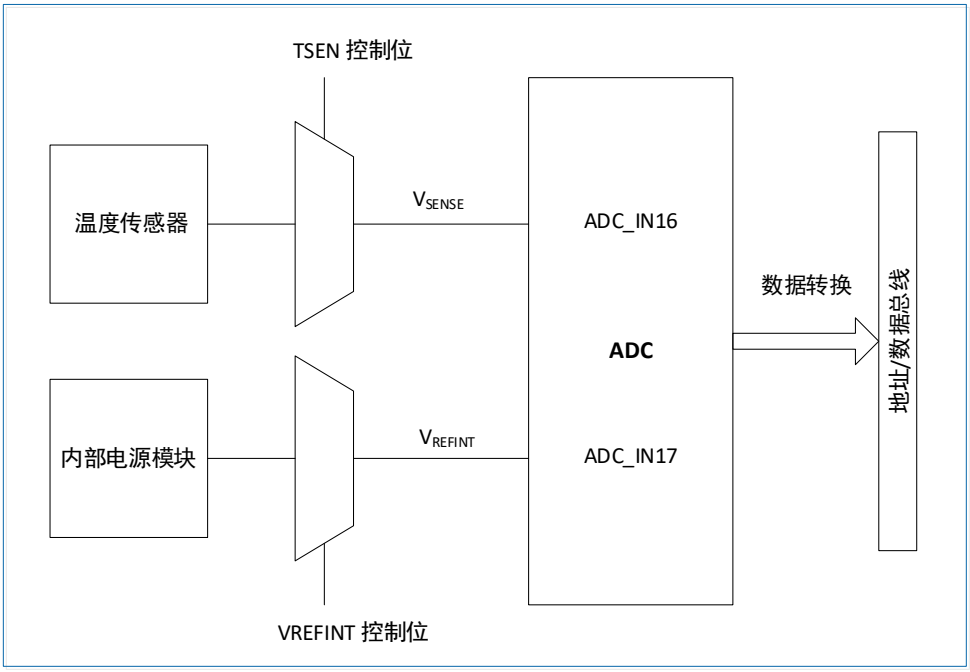


图 12-19 温度传感器和 VREFINT 通道框图

读取温度

1. 选择 ADC_IN16 输入通道。
2. 选择器件数据手册中规定的合适的采样时间（T_{s_temp}）。
3. 将 ADC_CCR 寄存器中的 TSEN 位置 1，将温度传感器从掉电模式中唤醒，并等待其稳定时间（t_{START}）。
4. 将 ADC_CR 寄存器中的 ADSTART 位置 1（或通过外部触发），开始 ADC 转换。
5. 读取 ADC_DR 寄存器中生成的 VSENSE 数据。
6. 使用以下公式计算温度：

$$温度（°C）=\frac{85°C-25°C}{TS\_CAL2-TS\_CAL1}\times（TS\_DATA-TS\_CAL1）+25$$

其中：

- TS_CAL2 是在 85°C 下获得的温度传感器校准值，见表 12-9。
- TS_CAL1 是在 25°C 下获得的温度传感器校准值，见表 12-9。
- TS_DATA 是由 ADC 转换得到的实际温度传感器输出值（上述步骤 5 中获取的值）。

更多关于 TS_CAL1 和 TS_CAL2 校准值的信息，请参见相应的器件数据手册。

注意：传感器从掉电模式中唤醒需要一个启动时间，启动时间过后其才能输出正确的 V_{SENSE}。ADC 在上电后同样需要一个启动时间，因此，为尽可能缩短延迟时间，应同时将 ADEN 和 TSEN 位置 1。

12.8 内部参考电压

内部参考电压（VREFINT）为 ADC 和比较器提供了一个稳定的（带隙基准）电压输出。VREFINT 内部连接到 ADC_IN17 输入通道。图 12-19 显示的是内部参考电压与 ADC 之间连接的方框图。

它可以精确地监视 V_{DDA} 值(当 ADC 没有外部电压 VREF+时)。VREFINT 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。访问模式为只读。用户应用可读取并使用这些数据提高内部参考电压的准确性。

表 12-10 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 F7BA-0x1FFF F7BB

使用内部参考电压计算实际的 V_{DDA} 电压

施加给微控制器的 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 的条件下获得的内置内部参考电压（V_{REFINT}）及其校准数据可用于评估实际的 V_{DDA} 电压水平。

以下公式可求得为器件供电的实际的 V_{DDA} 电压：

$$\frac{V_{REFINT\_CAL}}{4096} * 3.3 = \frac{V_{REFINT\_S}}{4096} * V_{DDA}$$

由上述公式可得：

$$V_{DDA} = \frac{3.3 \times V_{REFINT\_CAL}}{V_{REFINT\_S}}$$

其中：

- V_{REFINT_CAL} 是 VREFINT 校准值，见表 12-10。
- V_{REFINT_S} 表示内部参考电压的实际采样值。

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。对于大部分应用用例，需要将该比值转换成与 V_{DDA} 无关的电压。对于 V_{DDA} 已知、ADC 转换值进行了右对齐的应用，可使用以下公式得到该绝对值：

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

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

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

其中：

- V_{REFINT_CAL} 是 VREFINT 校准值，见表 12-10。
- ADC_DATA_x 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。例如，如果分辨率为 12 位，该值为2^{12} - 1 = 4095，如果分辨率为 8 位，该值为2^8-1 = 255。

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

12.9 ADC 中断

发生下列任一事件均可生成中断：

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

可以使用单独的中断使能位以提高灵活性。

表 12-11 ADC 中断

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

12.10 ADC 寄存器

基地址：0x4001 2400

空间大小：0x400

12.10.1 ADC 中断和状态寄存器（ADC_ISR）

偏移地址：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res								AWD	Res		OVR	EOS	EOC	EOSMP	ADRDY
								rc_w1			rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

位 31:8	Res：保留 必须保持复位值。
位 7	AWD：模拟看门狗标志（Analog watchdog flag） 当转换电压超过在 ADC_TR 寄存器中的编程值时，硬件会将该位置 1。通过软件写入 1 可将该位清零。

	• 0：未发生模拟看门狗事件（或标志事件已通过软件确认并清零） • 1：发生模拟看门狗事件
位 6:5	Res：保留 必须保持复位值。
位 4	OVR：ADC 溢出（ADC overrun flag） 该位在发生溢出事件时由硬件置 1，这意味着在 EOC 标志已置 1 时，新转换已完成。通过软件写入 1 可将该位清零。 • 0：未发生溢出事件（或标志事件已通过软件确认并清零） • 1：发生溢出
位 3	EOS：序列结束标志（End of sequence flag） 在由 CHSEL 位选择的一系列通道转换结束时，会通过硬件将该位置 1。通过软件写入 1 可将该位清零。 • 0：转换序列未完成（或标志事件已通过软件确认并清零） • 1：转换序列已完成
位 2	EOC：转换结束标志（End of conversion flag） 当通道的每次转换结束，新数据结果出现在 ADC_DR 寄存器时，会通过硬件将该位置 1。通过软件向该位写入 1，或读取 ADC_DR 寄存器都可将该位清零。 • 0：通道转换未完成（或标志事件已通过软件确认并清零） • 1：通道转换已完成
位 1	EOSMP：采样结束标志（End of sampling flag） 在转换过程中，当采样阶段结束时该位由硬件置 1。通过软件将该位写入 1，可将该位清零。 • 0：采样阶段未结束（或标志事件已通过软件确认并清零） • 1：采样阶段已结束
位 0	ADRDY：ADC 就绪（ADC ready） ADC 使能后（位 ADEN=1）以及 ADC 达到准备好接收转换请求的状态时，会通过硬件将该位置 1。 通过软件写入 1 可将该位清零。 • 0：ADC 未准备好开始转换（或标志事件已通过软件确认并清零） • 1：ADC 已准备好开始转换

12.10.2 ADC 中断使能寄存器（ADC_IER）

偏移地址：0x04

复位值：0x0000 0000

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

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

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

位 31:8	Res：保留 必须保持复位值。
位 7	AWDIE：模拟看门狗中断使能（Analog watchdog interrupt enable） 此位由软件置位和清零，用于使能/禁止模拟看门狗中断。 • 0：禁止模拟看门狗中断 • 1：使能模拟看门狗中断 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 6:5	Res：保留 必须保持复位值。
位 4	OVRIE：溢出中断使能（Overrun interrupt enable） 此位由软件置位和清零，用于使能/禁止溢出中断。 • 0：禁止溢出中断 • 1：使能溢出中断 OVR 位置 1 时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 3	EOSIE：转换序列结束中断使能（End of conversion sequence interrupt enable） 此位由软件置位和清零，用于使能/禁止转换序列结束中断。 • 0：禁止 EOS 中断 • 1：使能 EOS 中断 EOS 位置 1 时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 2	EOCIE：转换结束中断使能（End of conversion interrupt enable） 此位由软件置位和清零，用于使能/禁止转换结束中断。 • 0：禁止 EOC 中断 • 1：使能 EOC 中断 EOC 位置 1 时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 1	EOSMPIE：采样结束中断使能（End of sampling interrupt enable） 此位由软件置位和清零，用于使能/禁止采样阶段结束中断。 • 0：禁止 EOSMP 中断。 • 1：使能 EOSMP 中断。EOSMP 位置 1 时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*
位 0	ADRDYIE：ADC 就绪中断使能（ADC ready interrupt enable）

	此位由软件置位和清零，用于使能/禁止 ADC 就绪中断。 • 0：禁止 ADRDY 中断。 • 1：使能 ADRDY 中断。ADRDY 位置 1 时产生中断。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许软件对此位执行写操作。*

12.10.3 ADC 控制寄存器（ADC_CR）

偏移地址：0x08

复位值：0x0000 0000

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

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

位 31	ADCAL：ADC 校准（ADC calibration） 该位由软件置 1,来启动 ADC 校准。当校准完成后，由硬件清零。 • 0：校准完成 • 1：写 1 时，校准 ADC；读为 1 时，表明校准正在进行。 *注意：仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、ADSTP=0、ADDIS=0 且 ADEN=0），才允许通过软件将 ADCAL 位置 1。* *仅当 ADEN=1 且 ADSTART=0（ADC 已使能，当前未进行任何转换）时，才允许通过软件对 0x4001 27F4 地址执行写操作来更新校准系数。*
位 30:5	Res：保留 必须保持复位值。
位 4	ADSTP：ADC 停止转换命令（ADC stop conversion command） 该位由软件置 1，用于停止和丢弃正在进行的转换（ADSTP 命令）。 当转换结果已丢弃、并且 ADC 已准备好接收新的开始转换命令时，会通过硬件将该位清零。 • 0：当前未执行 ADC 停止转换命令。 • 1：写 1 用于停止 ADC；读为 1 时，表明 ADSTP 命令正在执行中。 *注意：仅当 ADSTART=1 且 ADDIS=0 时（ADC 已使能，可能正在进行转换，并且没有任何待处理的禁止 ADC 的请求），才允许通过软件将 ADSTP 置 1。*
位 3	Res：保留 必须保持复位值。
位 2	ADSTART：ADC 开始转换命令（ADC start conversion command） 此位由软件置 1，用于开始 ADC 转换。根据 EXTEN[1:0]配置位的值，可以立即开始转换（软件触发配置），也可以在发生硬件触发事件后开始转换（硬件触发配置）。

	• 0：当前未进行 ADC 转换。 • 1：写入 1 可开始 ADC。读取值为 1 表示 ADC 正在工作，可能正在进行转换。 该位通过硬件清零： • 在单次转换模式下（CONT=0、DISCEN=0），如果选择了软件触发（EXTEN=00）：出现转换序列结束（EOS）标志时清零。 • 在不连续转换模式下（CONT=0、DISCEN=1），如果选择了软件触发（EXTEN=00）：出现转换结束（EOC）标志时清零。 • 在所有其他情况下：执行完 ADSTP 命令后，由硬件将 ADSTP 位清零的同时清零该位。 *注意：仅当 ADEN=1 且 ADDIS=0 时（ADC 已使能，并且没有任何待处理的禁止 ADC 的请求），才允许软件将 ADSTART 置 1。*
位 1	ADDIS：ADC 禁止命令（ADC disable command） 该位通过软件置 1，用于禁止 ADC（ADDIS 命令）并使其进入掉电状态（OFF 状态）。 ADC 已有效禁止后，会立即通过硬件将该位清零（此时也会通过硬件将 ADEN 清零）。 • 0：当前未执行 ADDIS 命令 • 1：写入 1 可禁止 ADC。读取值为 1，表示正在执行 ADDIS 命令。 *注意：仅当 ADEN=1 且 ADSTART=0 时（确保当前未进行任何转换），才允许软件将 ADDIS 置 1。*
位 0	ADEN：ADC 使能命令（ADC enable command） 该位通过软件置 1，用于使能 ADC。ADRDY 标志置 1 后，ADC 将立即准备好运行。 如果 ADC 已禁止，则执行 ADDIS 命令后，将通过硬件对该位清零。 • 0：禁止 ADC（OFF 状态） • 1：写入 1 来使能 ADC *注意：仅当 ADC_CR 寄存器的所有位均为 0 时（ADCAL=0、ADSTP=0、ADSTART=0、ADDIS=0 且 ADEN=0），才允许软件将 ADEN 位置 1。*

12.10.4 ADC 配置寄存器 1（ADC_CFGR1）

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	AWDCH[4:0]					Res		AWDEN	AWDSGL	Res					DISCEN
	rw							rw	rw						rw

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

位 31	Res：保留 必须保持复位值。
位 30:26	AWDCH[4:0]：模拟看门狗通道选择（Analog watchdog channel selection）

	该位域由软件置位和清零，用于选择由模拟看门狗监控的输入通道。 • 00000：通过 AWD 监控 ADC 模拟输入通道 0。 • 00001：通过 AWD 监控 ADC 模拟输入通道 1。 • … • 10001：通过 AWD 监控 ADC 模拟输入通道 17。 • 其他值：保留（不能使用） *注意：由 AWDCH[4:0]位选择的通道也必须设置到 CHSELR 寄存器中。仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 25:24	Res：保留 必须保持复位值。
位 23	AWDEN：模拟看门狗使能（Analog watchdog enable） 该位由软件置位和清零。 • 0：禁止模拟看门狗 • 1：使能模拟看门狗 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 22	AWDSGL：在单一通道或所有通道上使能看门狗（Enable the watchdog on a single channel or on all channels） 该位由软件置位和清零，用于使能由 AWDCH[4:0]位确定的通道或所有通道上的模拟看门狗。 • 0：在所有通道上使能模拟看门狗 • 1：在单一通道上使能模拟看门狗 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 21:17	Res：保留 必须保持复位值。
位 16	DISCEN：不连续模式（Discontinuous mode） 该位由软件置位和清零，用于使能/禁止不连续模式。 • 0：禁止不连续模式 • 1：使能不连续模式 *注意：不能同时使能不连续模式和连续模式：禁止将 DISCEN 和 CONT 位同时置 1。* *仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 15	AUTOFF：自动关闭模式（Auto-off mode） 该位由软件置位和清零，用于使能/禁止自动关闭模式。 • 0：禁止自动关闭模式 • 1：使能自动关闭模式 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行*

	写操作。
位 14	AUTDLY：自动延迟转换模式（Auto delay conversion mode） 该位由软件置位和清零，用于使能/禁止自动延迟转换模式。. • 0：自动延迟转换模式关闭 • 1：自动延迟转换模式开启 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 13	CONT：单次/连续转换模式（Single /continuous conversion mode） 该位由软件置位和清零。该位置 1 时，转换将持续进行，直到该位清零。 • 0：单次转换模式 • 1：连续转换模式 *注意：不能同时使能不连续模式和连续模式：禁止同时将 DISCEN 和 CONT 位置 1。* *仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 12	OVRMOD：溢出管理模式（Overrun management mode） 该位通过软件进行置位和清零，并用于配置数据溢出的管理方式。 • 0：如果检测到溢出，ADC_DR 寄存器会保留原有数据。 • 1：如果检测到溢出，ADC_DR 寄存器会被上一次转换结果覆盖。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 11:10	EXTEN[1:0]：外部触发使能和极性选择（External trigger enable and polarity selection） 该位域由软件置位和清零，用于选择外部触发极性并使能触发。 • 00：禁止硬件触发检测（可通过软件开始转换）。 • 01：在上升沿执行硬件触发检测。 • 10：在下降沿执行硬件触发检测。 • 11：在上升沿和下降沿都执行硬件触发检测。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 9	Res：保留 必须保持复位值。
位 8:6	EXTSEL[2:0]：外部触发选择（External trigger selection） 该位域可选择用于触发转换开始的外部事件（详情请参见“表 12-5 外部触发器”）： • 000：TRG0 • 001：TRG1 • 010：TRG2 • 011：TRG3 • 100：TRG4

	• 101：TRG5 • 110：TRG6 • 111：TRG7 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 5	ALIGN：数据对齐（Data alignment） 该位由软件置位和清零，用于选择右对齐或左对齐。请参见“图 12-13 数据对齐方式和分辨率”。 • 0：右对齐 • 1：左对齐 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 4:3	RES[1:0]：数据分辨率（Data resolution） 通过软件写入该位域，以选择转换的分辨率。 • 00：12 位 • 01：10 位 • 10：8 位 • 11：6 位 *注意：仅当 ADEN=0 时，才允许通过软件对该位域执行写操作。*
位 2	SCANDIR：扫描序列方向（Scan sequence direction） 该位由软件置位和清零，用于选择扫描序列中各通道的方向。 • 0：向前扫描（CHSEL0 到 CHSEL17） • 1：向后扫描（CHSEL17 到 CHSEL0） *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 1	DMACFG：直接存储器访问配置（Direct memory access configuration） 该位由软件置位和清零，用于在 DMA 两种工作模式之间进行选择。仅当 DMAEN=1 时，该位才有效。 • 0：选择 DMA 单次模式 • 1：选择 DMA 循环模式 更多详细信息，请参见“12.4.5 使用 DMA 管理转换的数据”。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 0	DMAEN：直接存储器访问使能（Direct memory access enable） 该位由软件置位和清零，用于使能 DMA 请求的生成。这样便可使用 DMA 控制器自动管理转换的数据。有关详细信息，请参见“12.4.5 使用 DMA 管理转换的数据”。 • 0：禁止 DMA • 1：使能 DMA

	注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。

12.10.5 ADC 配置寄存器 2（ADC_CFGR2）

偏移地址：0x10

复位值：0x0000 0000

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

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

位 31:30	CKMODE[1:0]：ADC 时钟模式（ADC clock mode） 该位由软件置位和清零，用于定义为模拟 ADC 提供时钟的方式： • 00：ADCCLK（异步时钟模式）（请参见“6 复位和时钟控制（RCC）”）。 • 01：PCLK/2（同步时钟模式） • 10：PCLK/4（同步时钟模式） • 11：保留
位 29:0	Res：保留 必须保持复位值。

12.10.6 ADC 采样时间寄存器（ADC_SMPR）

偏移地址：0x14

复位值：0x0000 0000

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

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

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

12.10.7 ADC 看门狗阈值寄存器（ADC_TR）

偏移地址：0x20

复位值：0x00000FFF

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

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

位 31:28	Res：保留 必须保持复位值。
位 27:16	HT[11:0]：模拟看门狗阈值上限（Analog watchdog higher threshold） 该位域由软件写入，用于定义模拟看门狗的阈值上限。请参见“12.6 模拟窗口看门狗（AWDEN, AWDSGL, AWDCH, AWD_HTR/LTR, AWD）”。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*
位 15:12	Res：保留 必须保持复位值。
位 11:0	LT[11:0]：模拟看门狗阈值下限（Analog watchdog lower threshold） 该位域由软件写入，用于定义模拟看门狗的阈值下限。 请参见“12.6 模拟窗口看门狗（AWDEN, AWDSGL, AWDCH, AWD_HTR/LTR, AWD）”。 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*

12.10.8 ADC 通道选择寄存器（ADC_CHSELR）

偏移地址：0x28

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res														CHSEL17	CHSEL16
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHSEL15	CHSEL14	CHSEL13	CHSEL12	CHSEL11	CHSEL10	CHSEL9	CHSEL8	CHSEL7	CHSEL6	CHSEL5	CHSEL4	CHSEL3	CHSEL2	CHSEL1	CHSEL0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:18	Res：保留 必须保持复位值。
位 x (x=17..0)	CHSELx：通道 x 选择（Channel-x selection） 该位域由软件写入，用于定义将哪些通道纳入要转换的通道序列。 • 0：未选择输入通道 x 进行转换 • 1：已选择输入通道 x 进行转换 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位域执行写操作。*

12.10.9 ADC 数据寄存器（ADC_DR）

偏移地址：0x40

复位值：0x0000 0000

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

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

位 31:16	Res：保留 必须保持复位值。
位 15:0	DATA[15:0]：转换后的数据（Converted data） 该位域为只读，包含上一转换通道的转换结果。数据可以采用左对齐和右对齐，如图 12-13 所示。校准完成后，DATA[5:0]会立即包含校准系数。

12.10.10 ADC 通用配置寄存器（ADC_CCR）

偏移地址：0x308

复位值：0x0000 0000

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

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

位 31:24	Res：保留 必须保持复位值。

位 23	TSEN：温度传感器使能（Temperature sensor enable） 该位由软件置位和清零，用于使能/禁止温度传感器。 • 0：禁止温度传感器 • 1：使能温度传感器 *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 22	VREFEN：V_{REFINT} 使能（V_{REFINT} enable） 该位由软件置位和清零，用于使能/禁止 V_{REFINT} 通道。 • 0：禁止 V_{REFINT} • 1：使能 V_{REFINT} *注意：仅当 ADSTART=0 时（确保当前未进行任何转换），才允许通过软件对该位执行写操作。*
位 21:0	Res：保留 必须保持复位值。

13 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、带死区时间的互补 PWM 等）。

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

高级控制定时器（TIM1）和其他定时器完全独立，不共享任何资源。

13.1 TIM1 主要特征

TIM1 定时器的功能包括：

- 16 位向上、向下、向上/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。
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中央对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 以下事件发生时产生中断/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

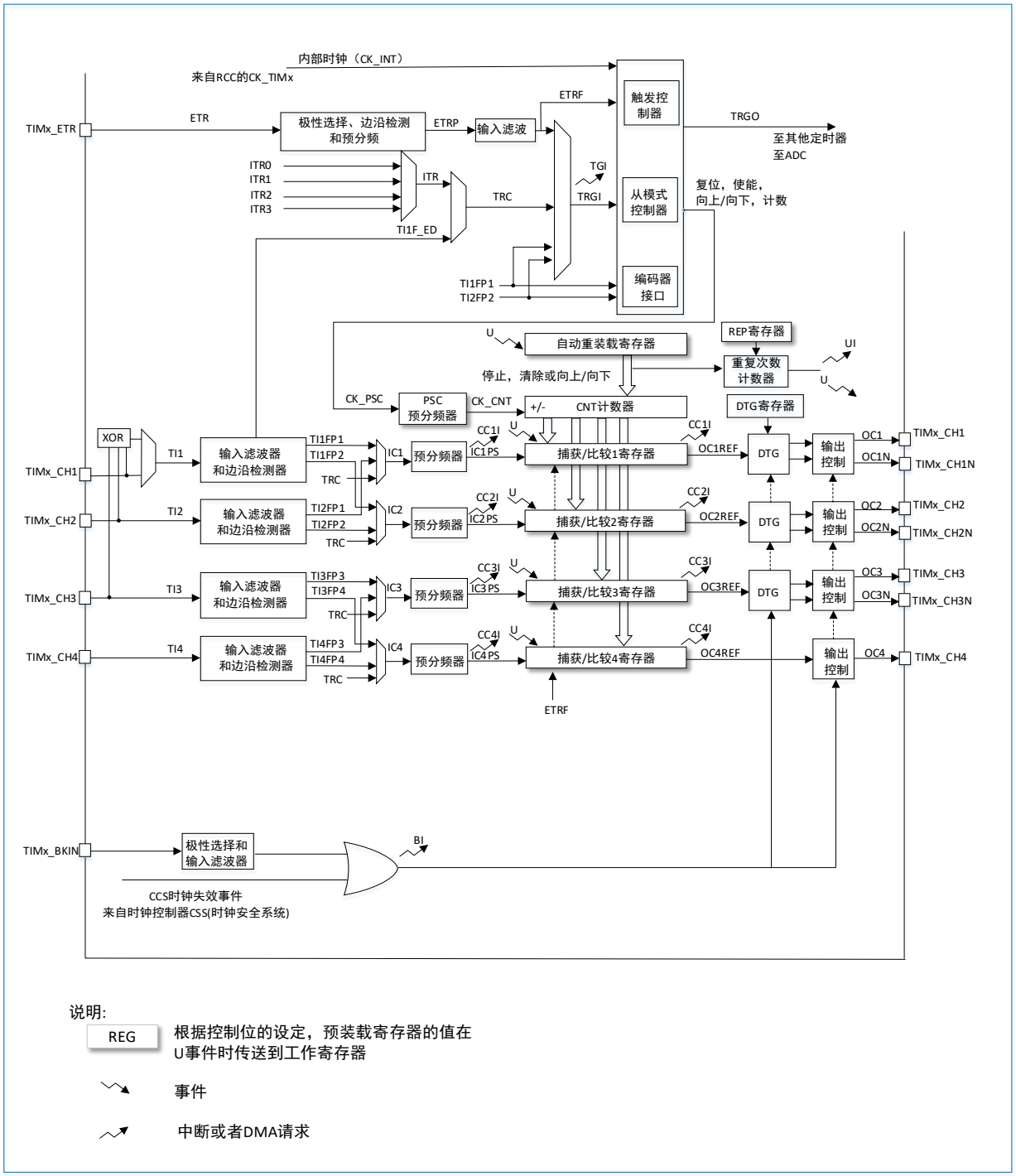


图 13-1 高级控制定时器框图

13.2 TIM1 功能描述

13.2.1 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，参见“13.2.19 TIM1 定时器和外部触发的同步”）。

注意：在设置了 TIM1_CR 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

预分频器描述

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

图 13-2 和图 13-3 给出了在预分频器运行时，更改计数器参数的例子。

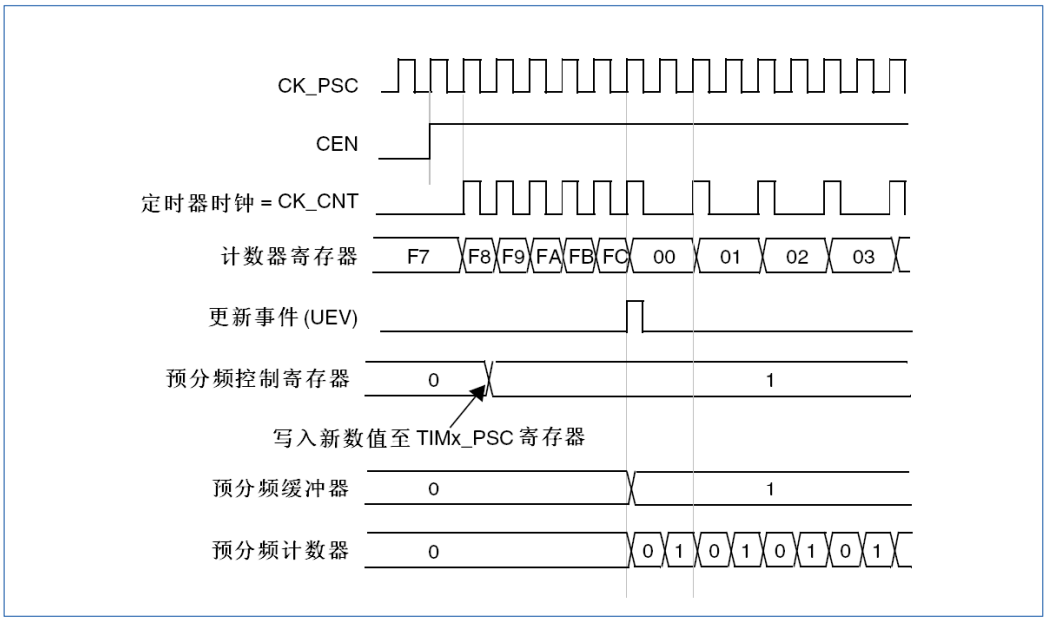


图 13-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

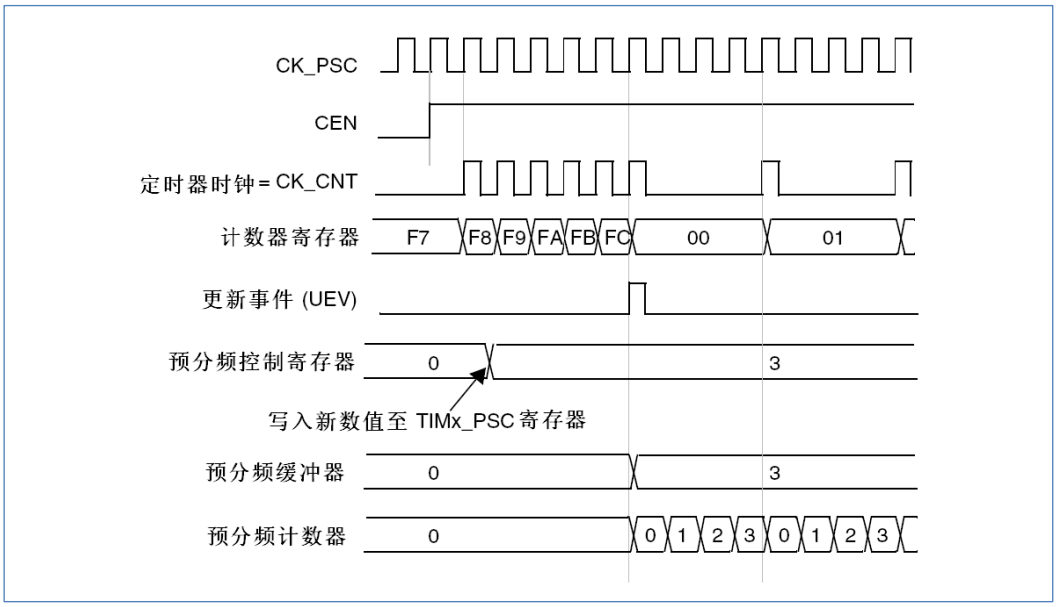


图 13-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

13.2.2 计数器模式

13.2.2.1 向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值（TIM1_ARR 计数器的内容），然后重新从 0 开始计数并且产生一个计数器溢出事件。

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数（TIM1_RCR）时，产生更新事件（UEV）；否则每次计数器溢出时才产生更新事件。

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

设置 TIM1_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清零（但预分频器的数值不变）。此外，如果设置了 TIM1_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断或 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）。

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

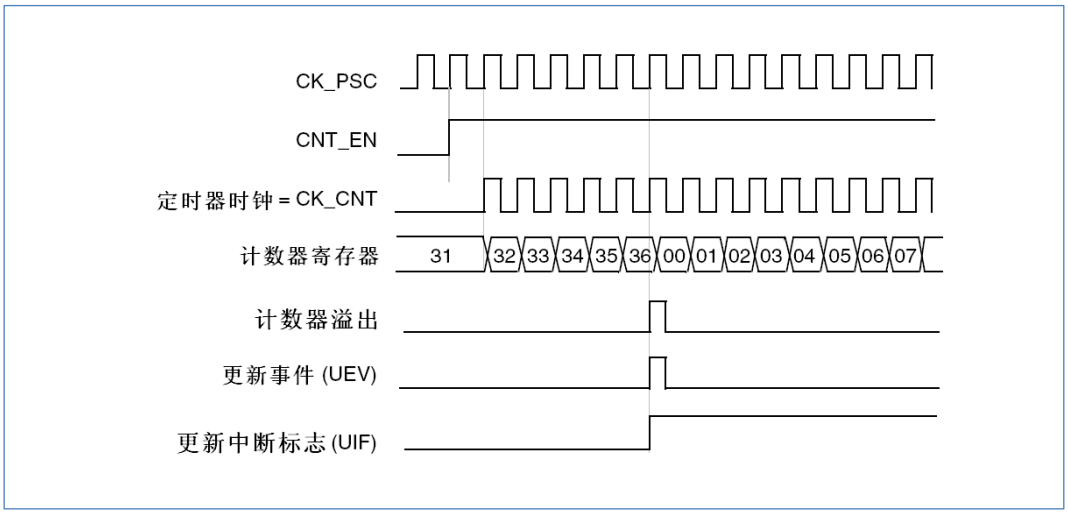


图 13-4 计数器时序图，内部时钟分频因子为 1

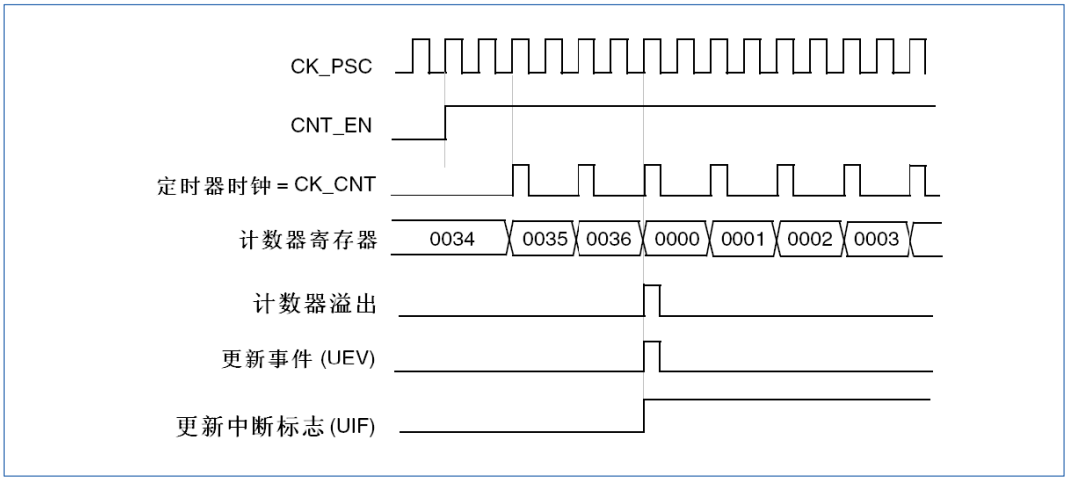


图 13-5 计数器时序图，内部时钟分频因子为 2

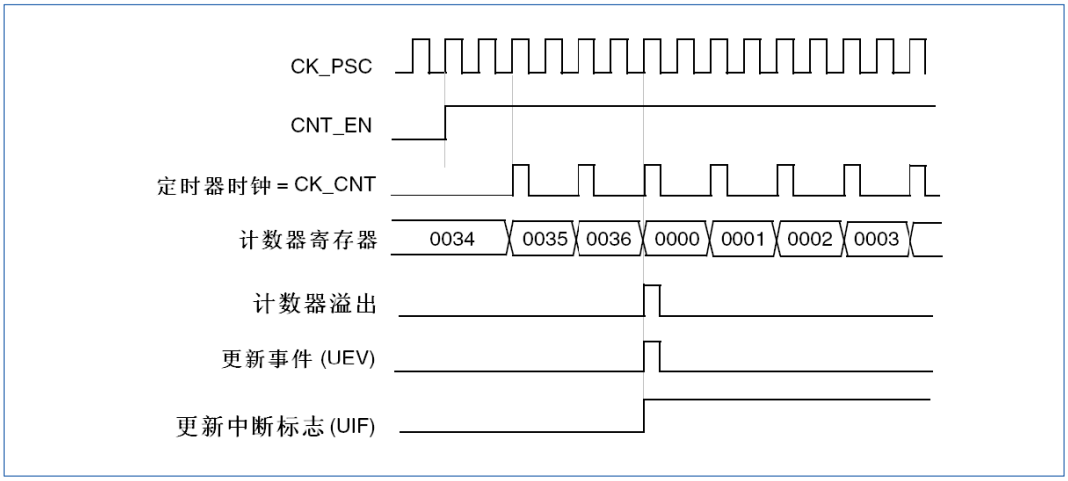


图 13-6 计数器时序图，内部时钟分频因子为 4

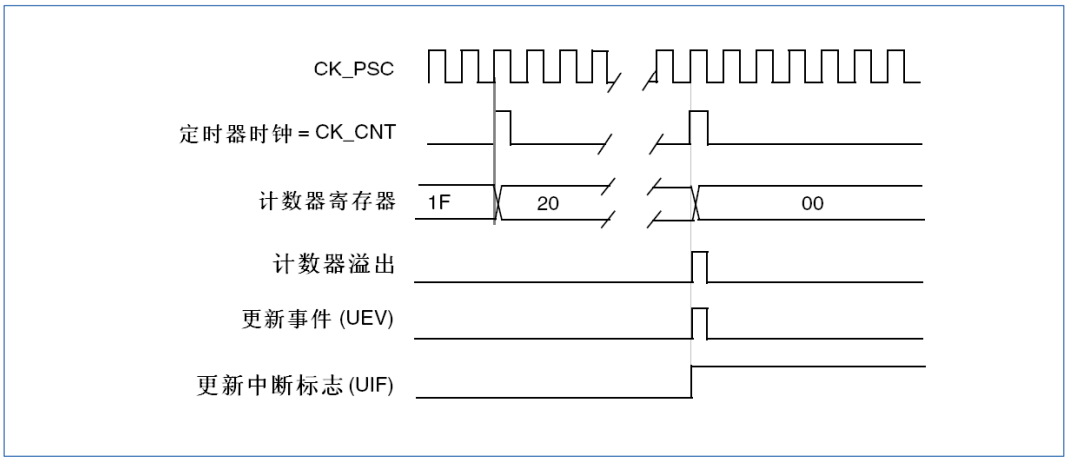


图 13-7 计数器时序图，内部时钟分频因子为 N

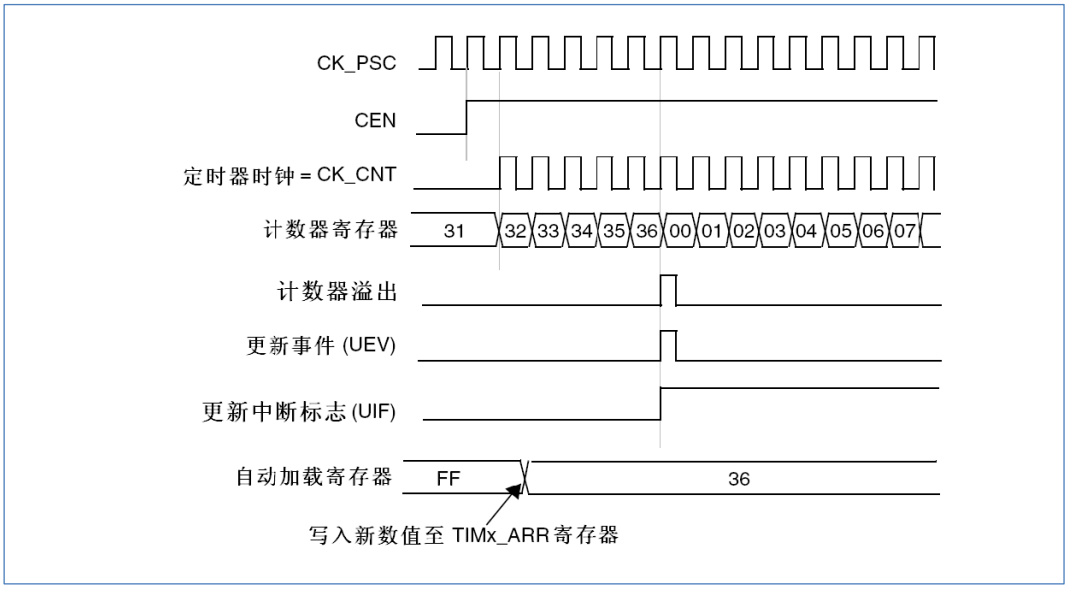


图 13-8 计数器时序图，当 ARPE=0 时的更新事件（TIM1_ARR 没有预装入）

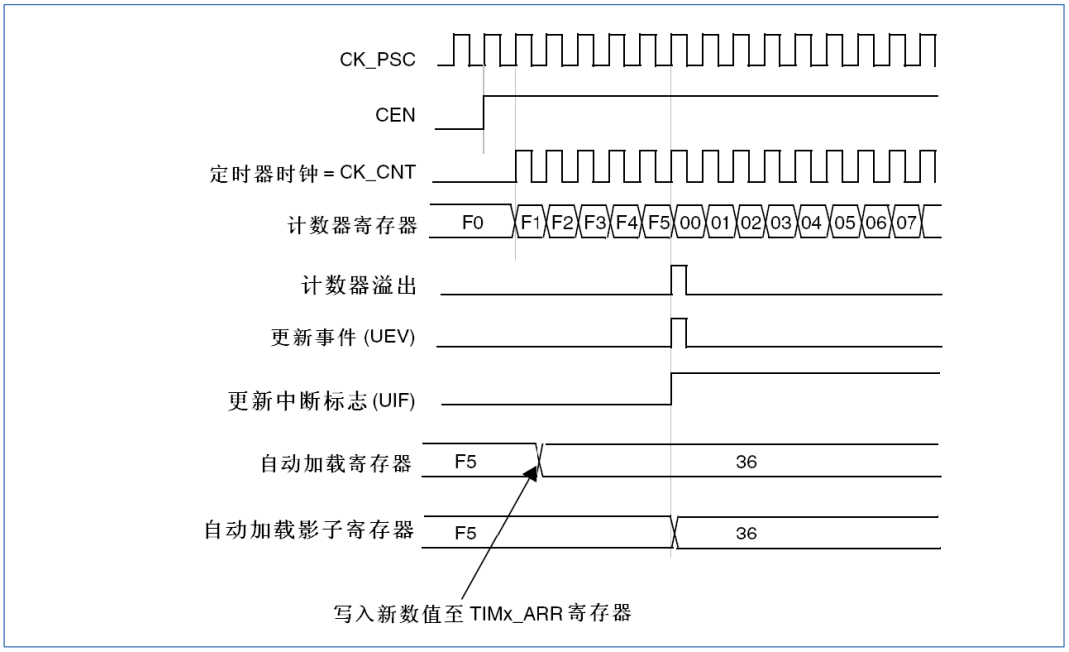


图 13-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM1_ARR）

13.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（TIM1_ARR 计数器的值）开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

如果使用了重复计数器，当向下计数重复了重复计数寄存器（TIM1_RCR）中设定的次数后，将产生更新事件（UEV），否则每次计数器下溢时才产生更新事件。

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

设置 TIM1_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始（但预分频系数不变）。

此外，如果设置了 TIM1_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断和 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 TIM1_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

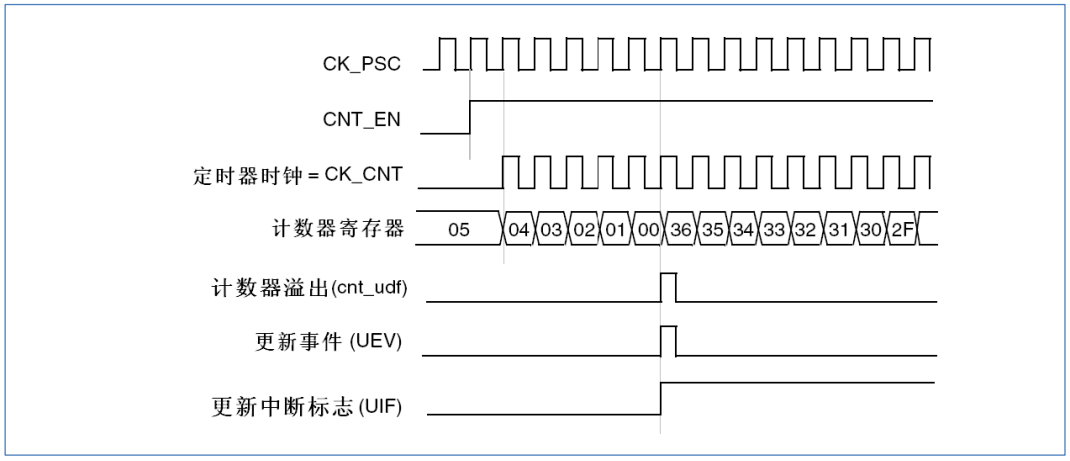


图 13-10 计数器时序图，内部时钟分频因子为 1

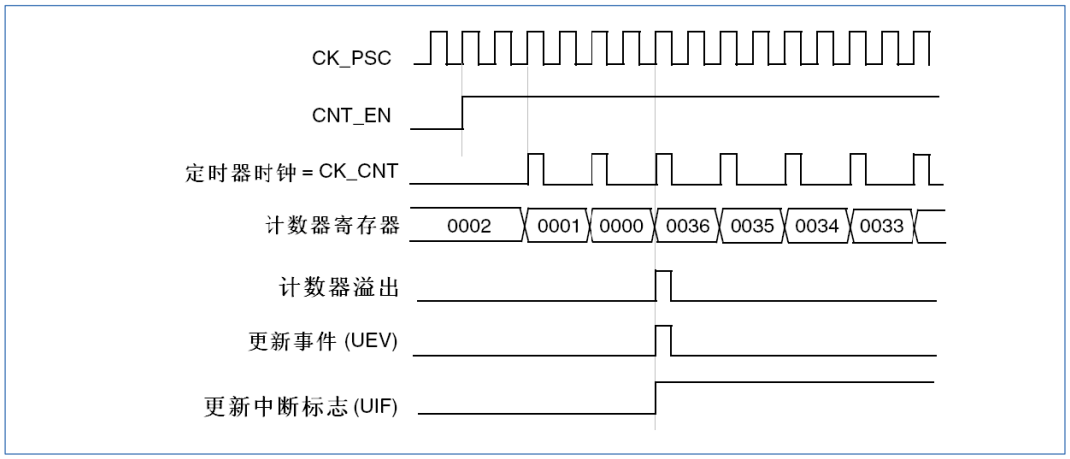


图 13-11 计数器时序图，内部时钟分频因子为 2

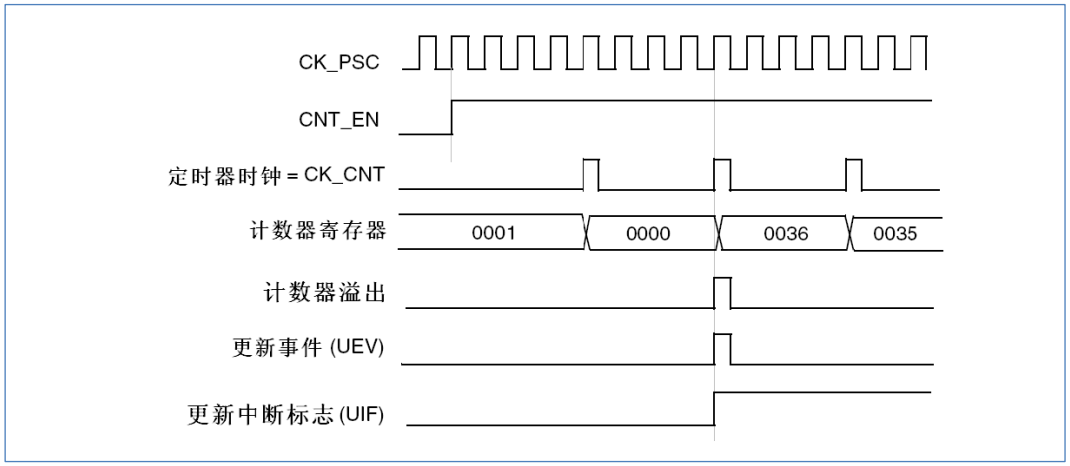


图 13-12 计数器时序图，内部时钟分频因子为 4

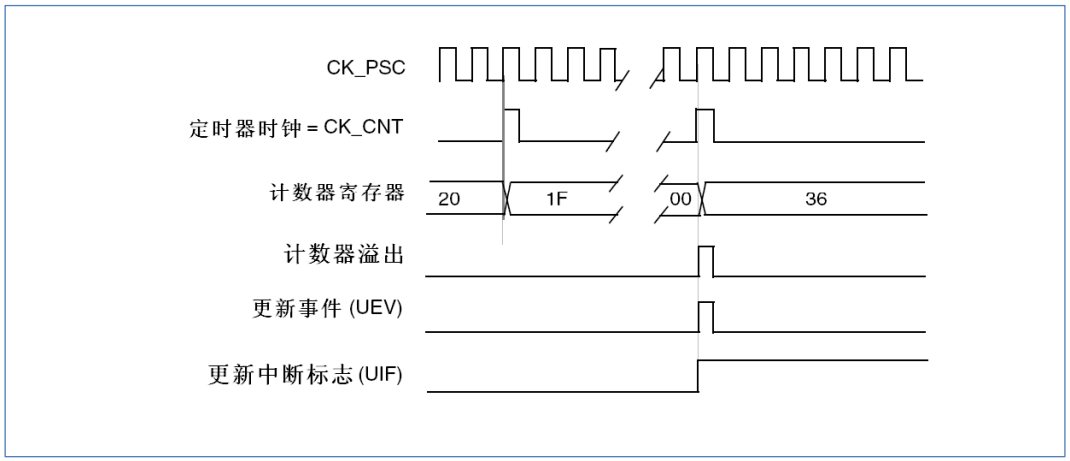


图 13-13 计数器时序图，内部时钟分频因子为 N

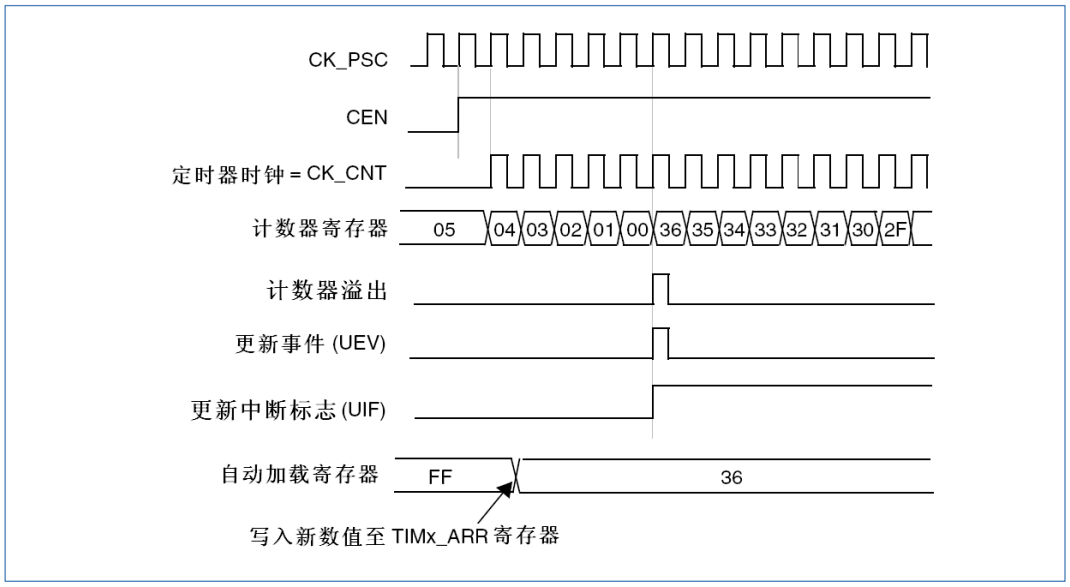


图 13-14 计数器时序图，当没有使用重复计数器时的更新事件

13.2.2.3 中央对齐模式（向上/向下计数）

在中央对齐模式，计数器从 0 开始计数到自动加载的值（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）。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：

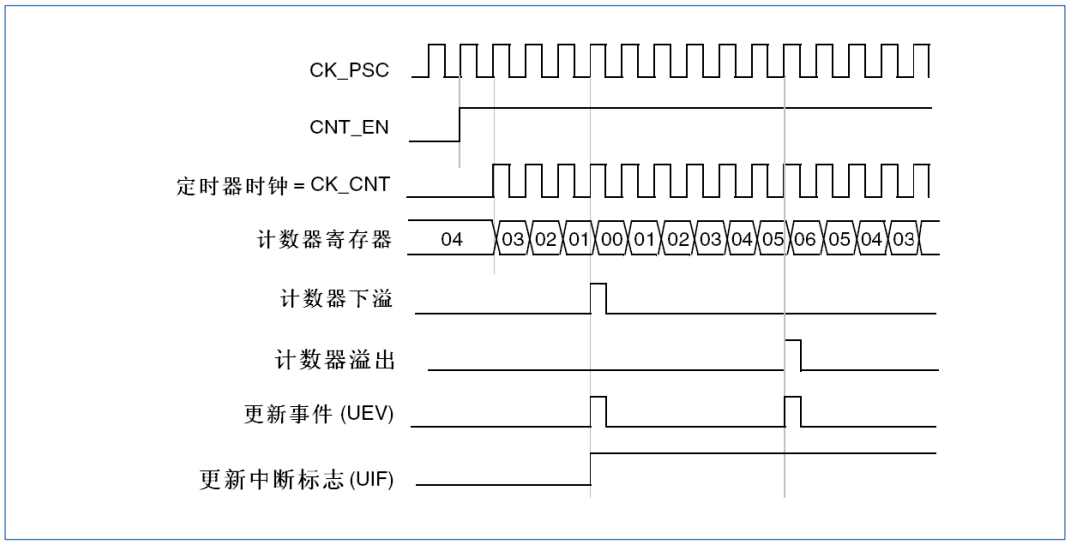


图 13-15 计数器时序图，内部时钟分频因子为 1，TIM1_ARR=0x6（这里使用了中央对齐模式 1）

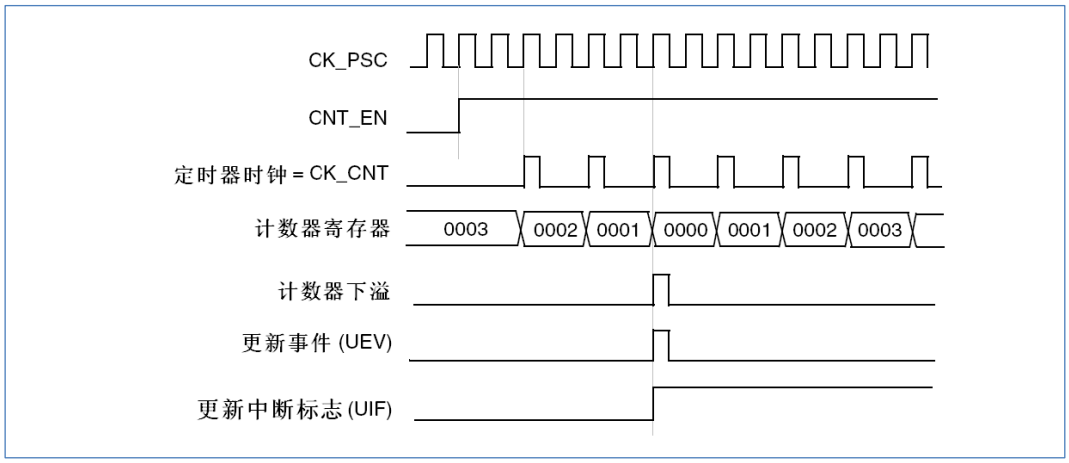


图 13-16 计数器时序图，内部时钟分频因子为 2（中央对齐模式 1）

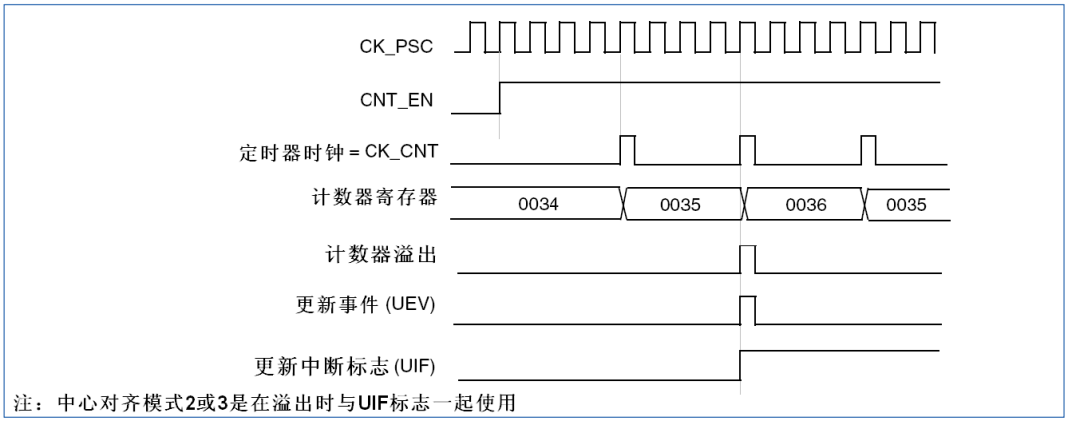


图 13-17 计数器时序图，内部时钟分频因子为 4，TIM1_ARR=0x36（中央对齐模式 2 或 3）

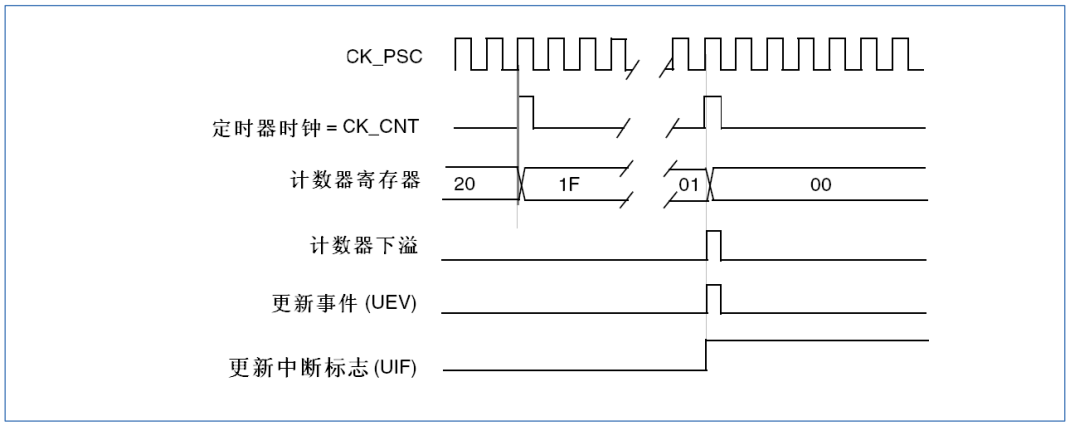


图 13-18 计数器时序图，内部时钟分频因子为 N

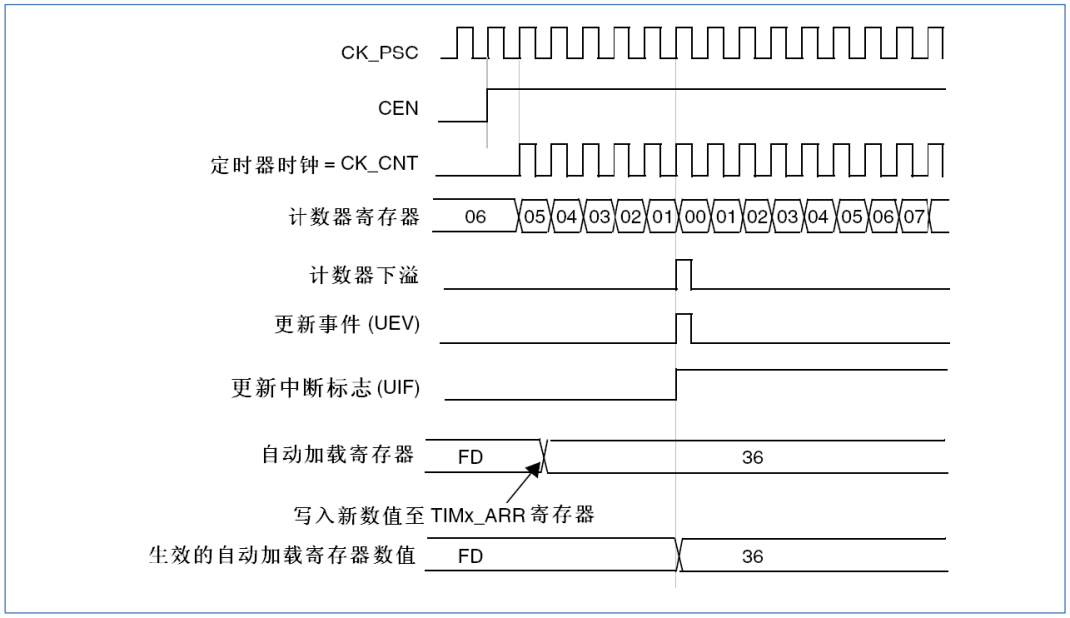


图 13-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

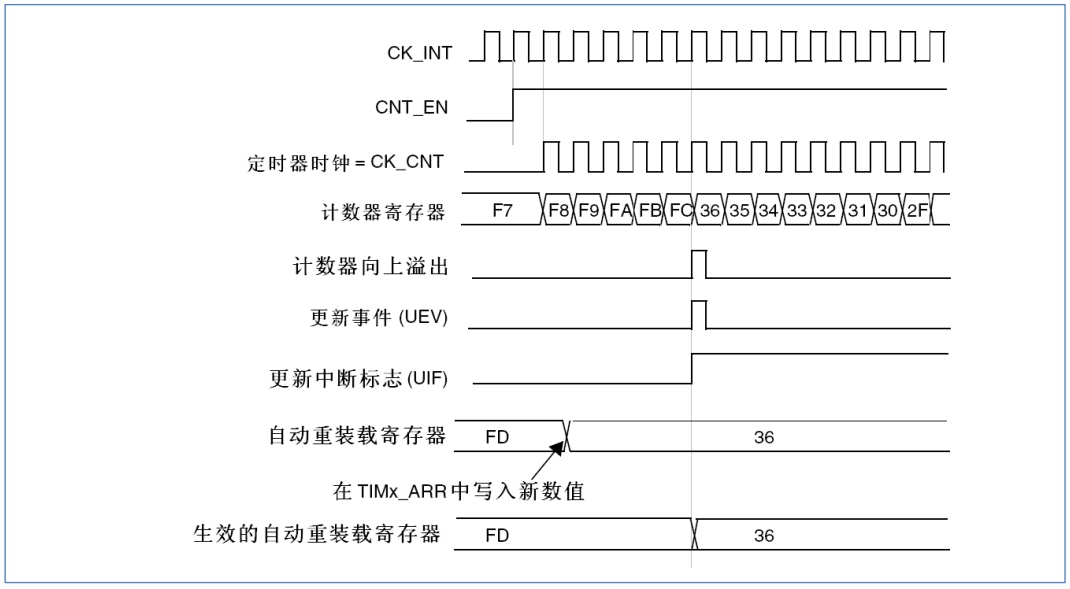


图 13-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

13.2.3 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 2x Tck。

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（参看图 13-21）。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。

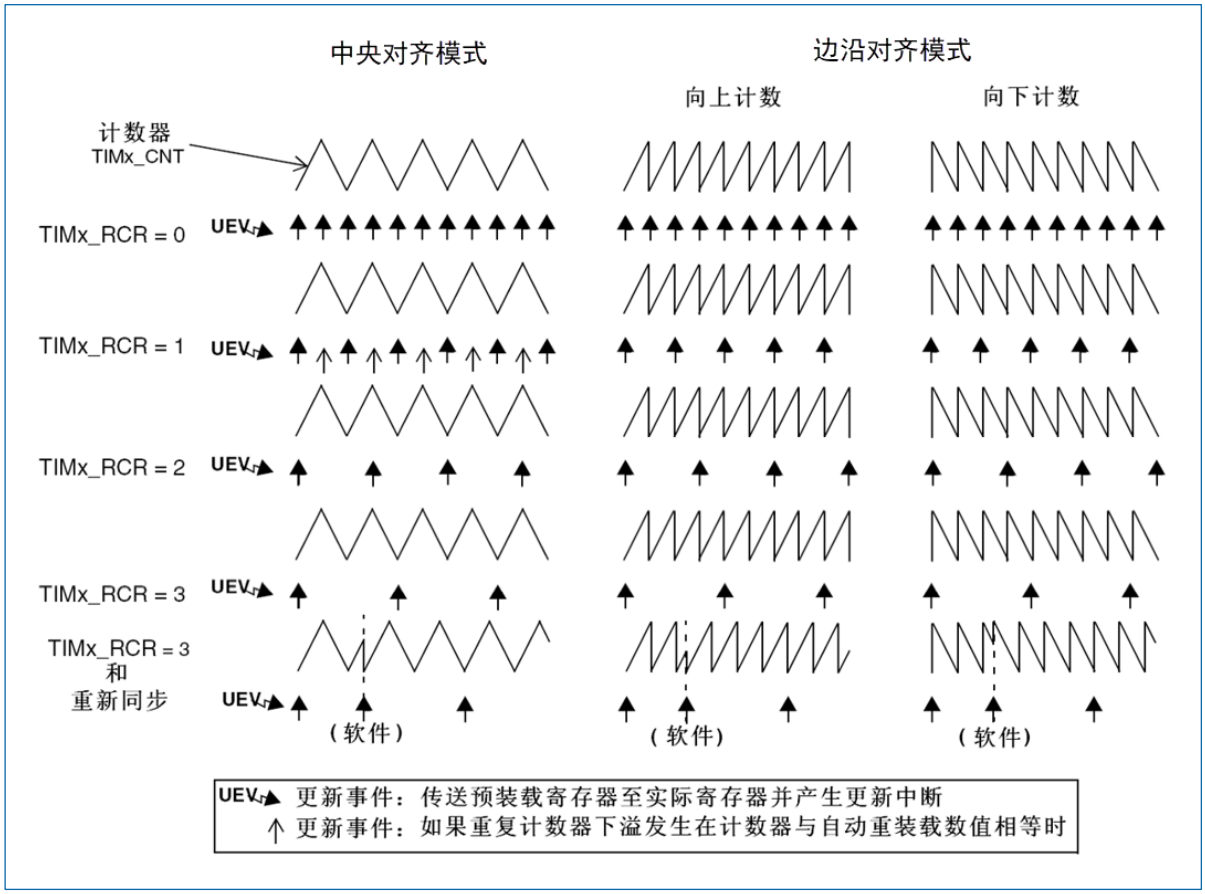


图 13-21 不同模式下更新速率的例子，及 TIM1_RCR 的寄存器设置

13.2.4 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
- 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“14.2.15.1 使用一个定时器作为另一个定时器的预分频器”。

13.2.4.1 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。

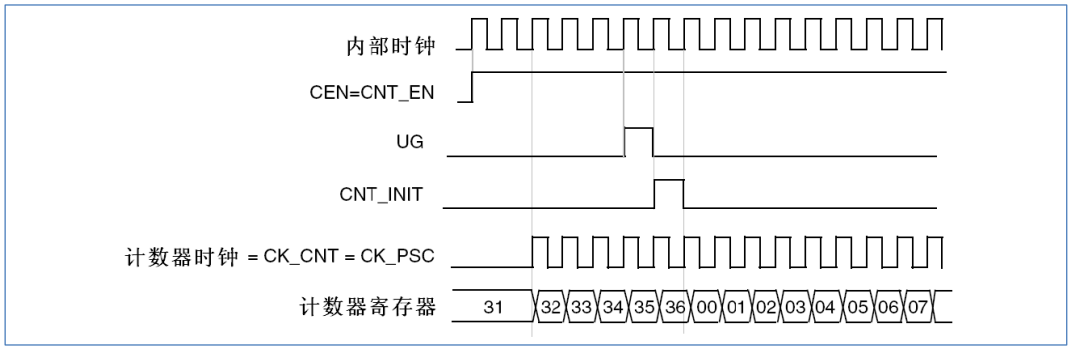


图 13-22 一般模式下的控制电路，内部时钟分频因子为 1

13.2.4.2 外部时钟源模式 1

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

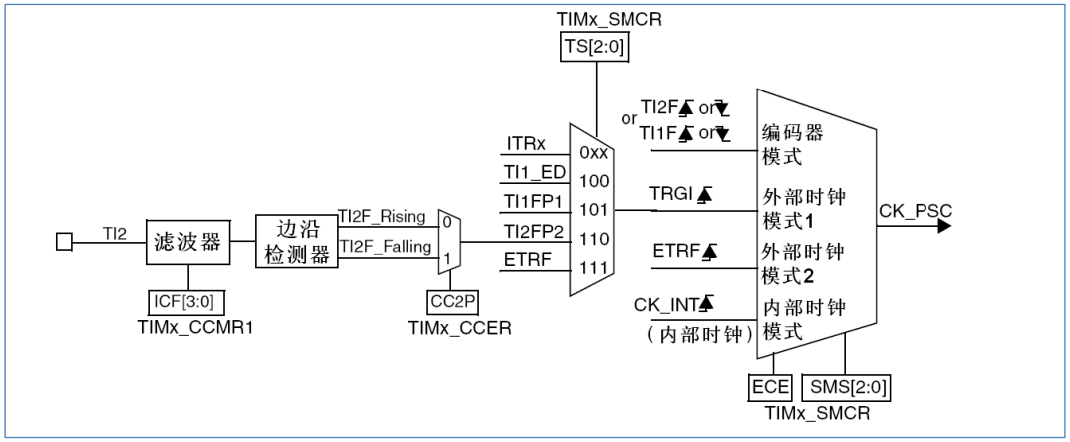


图 13-23 TI2 外部时钟连接例子

例如，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。

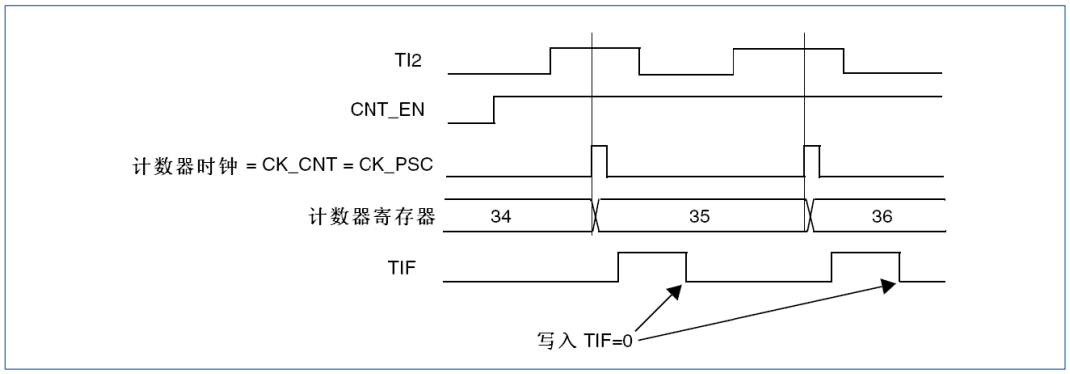


图 13-24 外部时钟模式 1 下的控制电路

13.2.4.3 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。

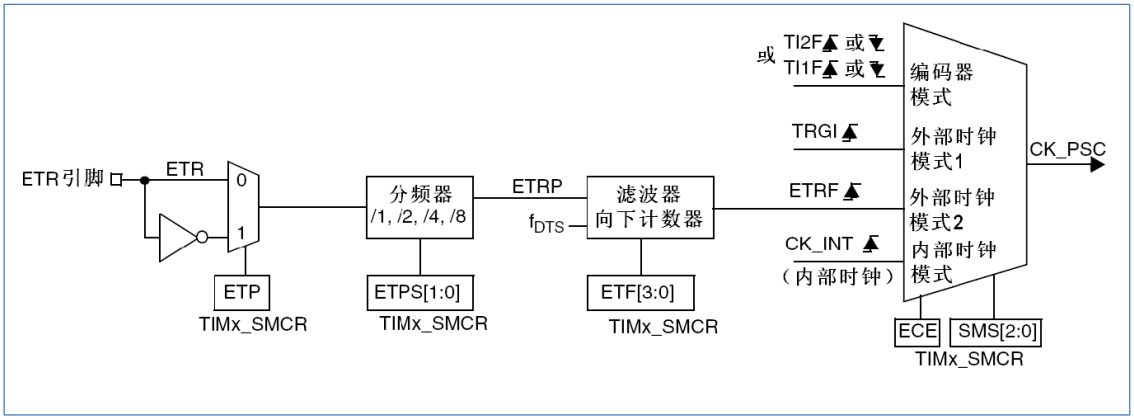


图 13-25 外部触发输入框图

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

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

计数器在每 2 个 ETR 上升沿计数一次。在 ETR 的上升沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。

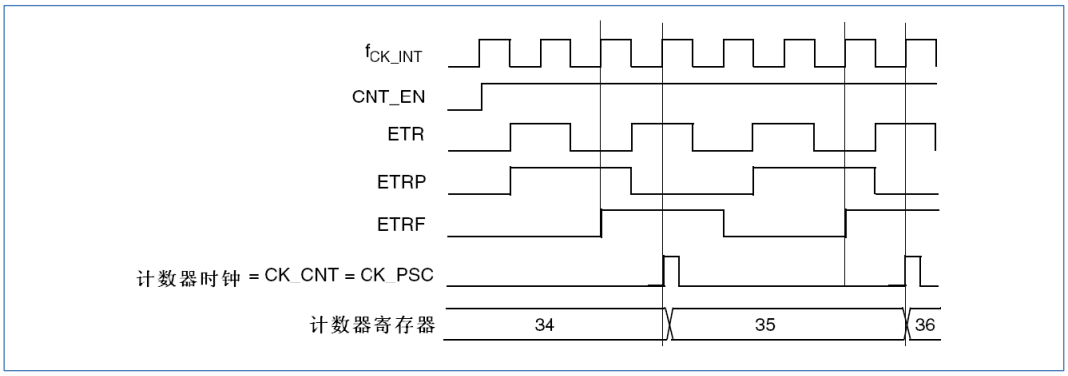


图 13-26 外部时钟模式 2 下的控制电路

13.2.5 捕获/比较通道

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

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

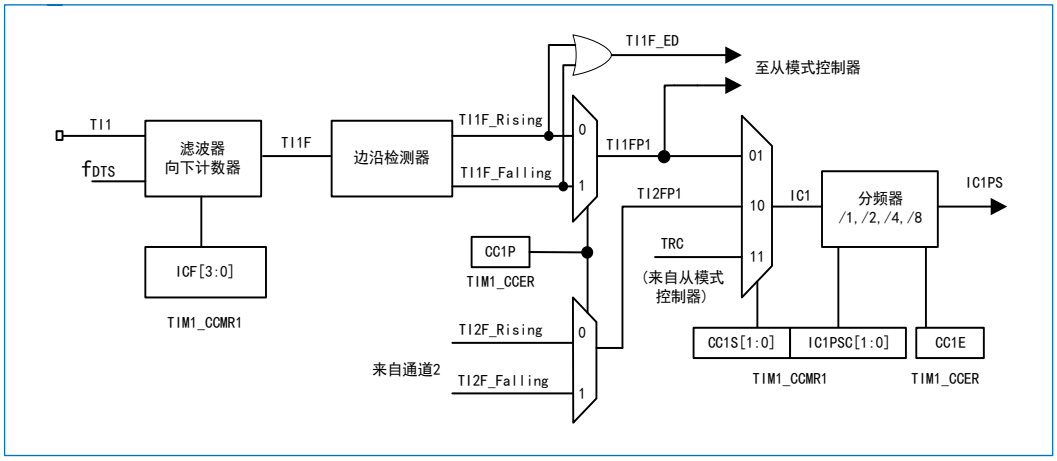


图 13-27 捕获/比较通道（如：通道 1 输入部分）

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

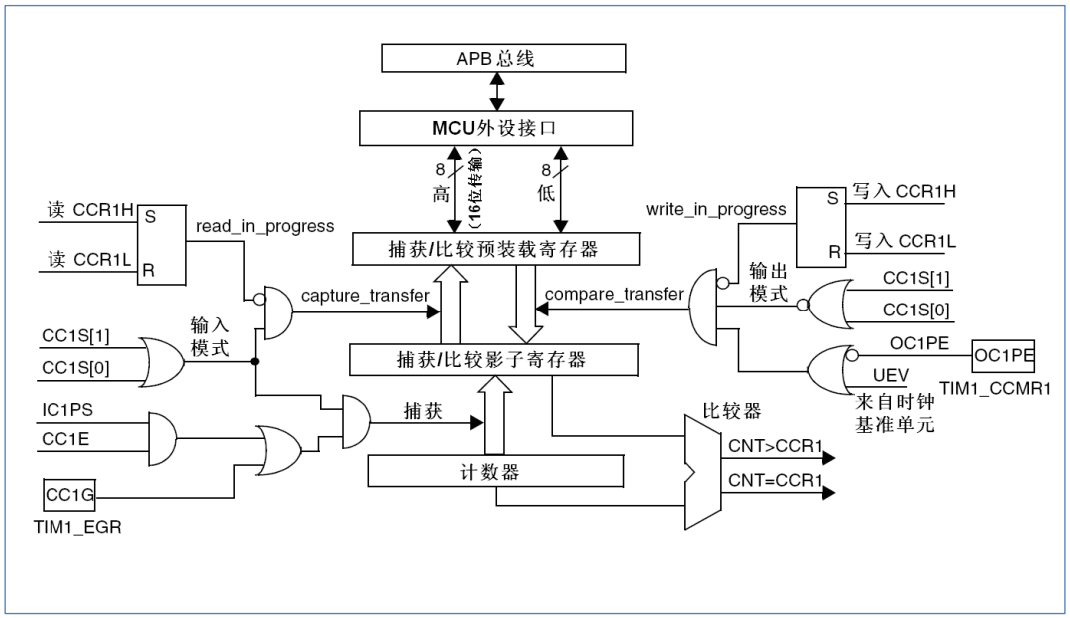


图 13-28 捕获/比较通道 1 的主电路

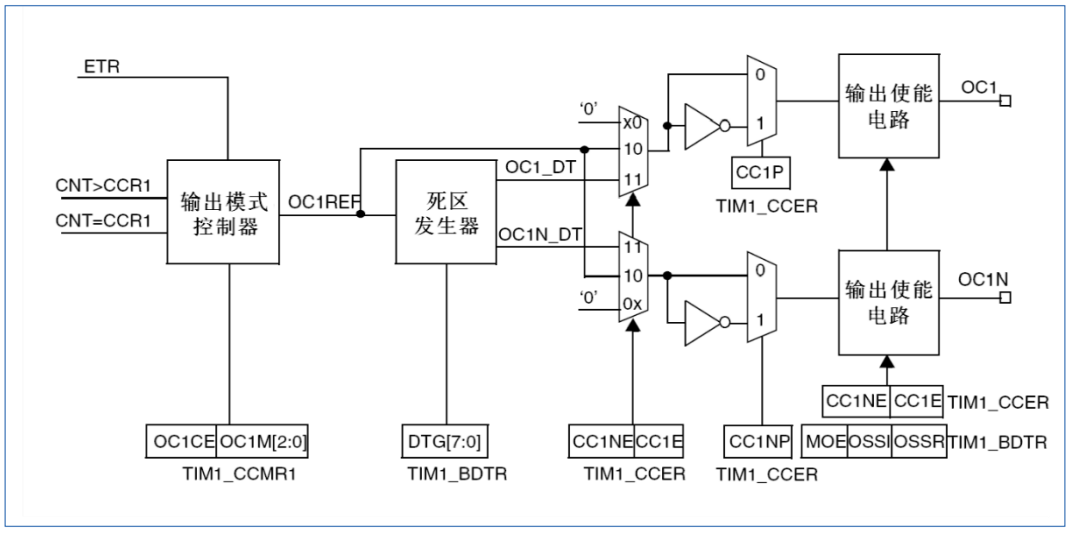


图 13-29 捕获/比较通道的输出部分（通道 1 至 3）

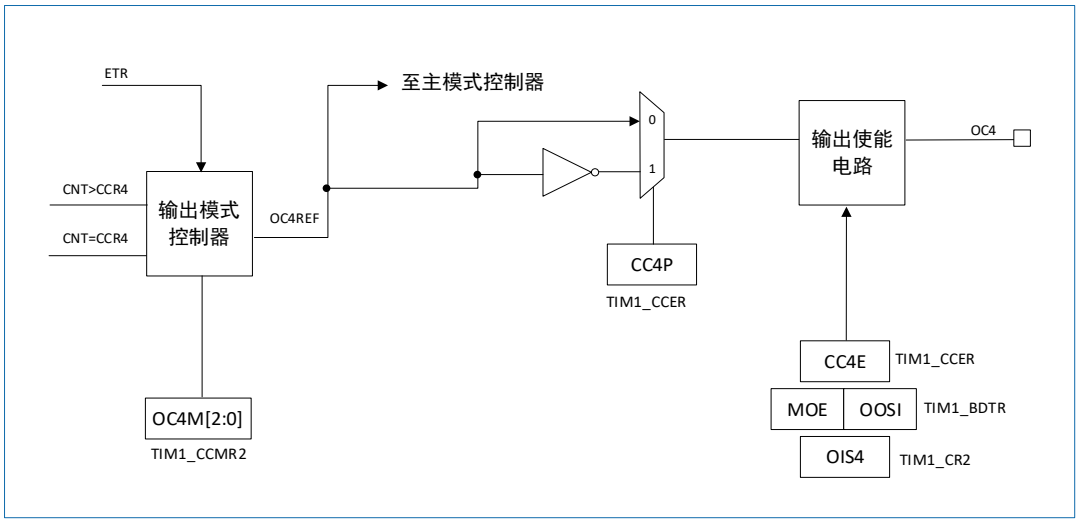


图 13-30 捕获/比较通道的输出部分（通道 4）

捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

13.2.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIM1_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIM1_SR 寄存器）被置 1，如果开放了中断或者 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 fDTS 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 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 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

注意：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。

13.2.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

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

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

1. 选择 TIM1_CCR1 的有效输入：置 TIM1_CCMR1 寄存器的 CC1S=01（选中 TI1）。

2. 选择 TI1FP1 的有效极性（用来捕获数据到 TIM1_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。

3. 选择 TIM1_CCR2 的有效输入：置 TIM1_CCMR1 寄存器的 CC2S=10（选中 TI1）。

4. 选择 TI1FP2 的有效极性（捕获数据到 TIM1_CCR2）：置 CC2P=1（下降沿有效）。

5. 选择有效的触发输入信号：置 TIM1_SMCR 寄存器中的 TS=101（选择 TI1FP1）。

6. 配置从模式控制器为复位模式：置 TIM1_SMCR 中的 SMS=100。

7. 使能捕获：置 TIM1_CCER 寄存器中 CC1E=1 且 CC2E=1。

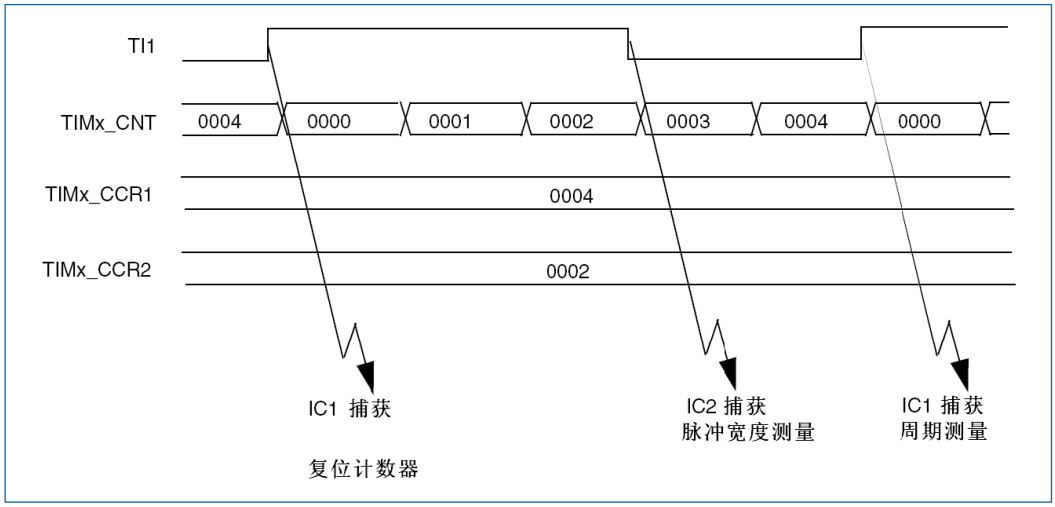


图 13-31 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIM1_CH1/TIM1_CH2 信号。

13.2.8 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。

13.2.9 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，输出比较功能按如下操作：

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

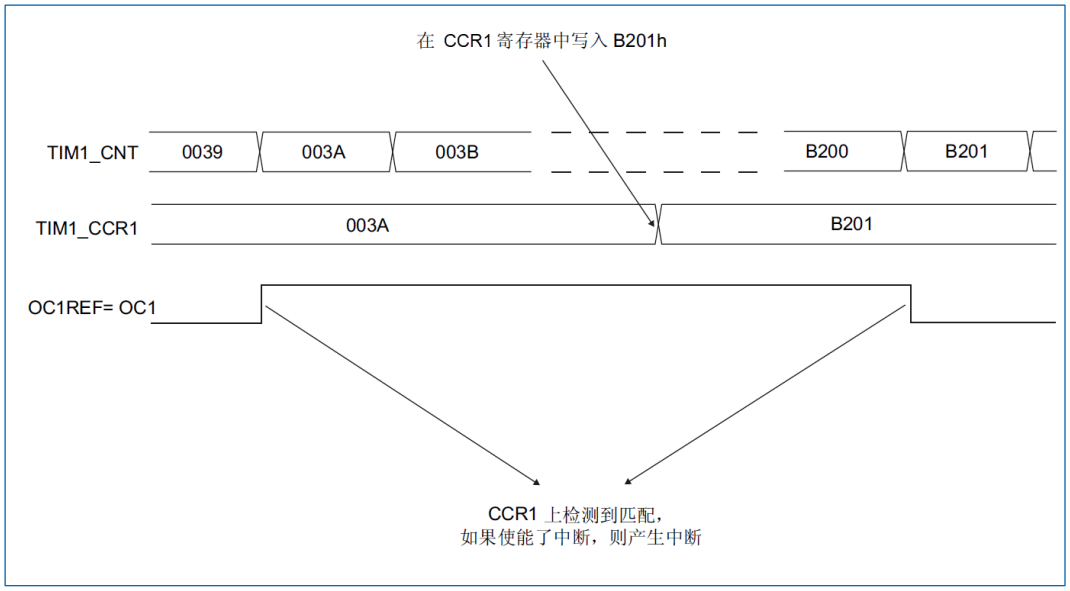


图 13-32 输出比较模式，翻转 OC1

13.2.10 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，（在向上计数或中心对称模式中）使能自动重装

载的预装载寄存器。

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

OCx 的极性可以通过软件在 TIM1_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过 CCxE、CCxNE、MOE、OSSI 和 OSSR 位（TIM1_CCER 和 TIM1_BDTR 寄存器中）的组合控制。参见“13.3.9 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。

13.2.10.1 PWM 边沿对齐模式

向上计数配置

当 TIM1_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

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

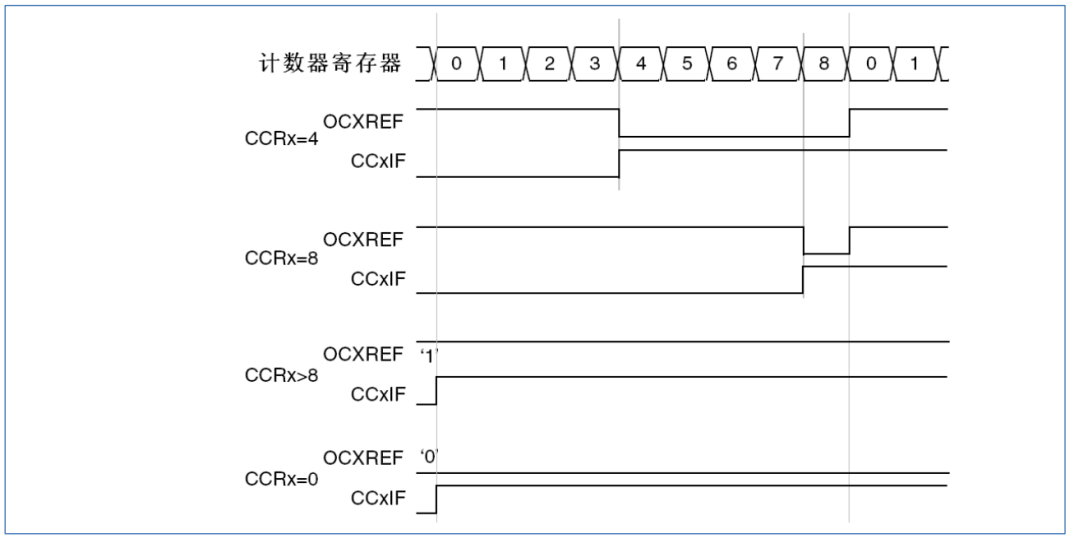


图 13-33 边沿对齐的 PWM 波形（ARR=8）

向下计数的配置

当 TIM1_CR1 寄存器的 DIR 位为高时执行向下计数。

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

13.2.10.2 PWM 中央对齐模式

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

下图给出了一些中央对齐的 PWM 波形的例子。

- TIM1_ARR=8

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

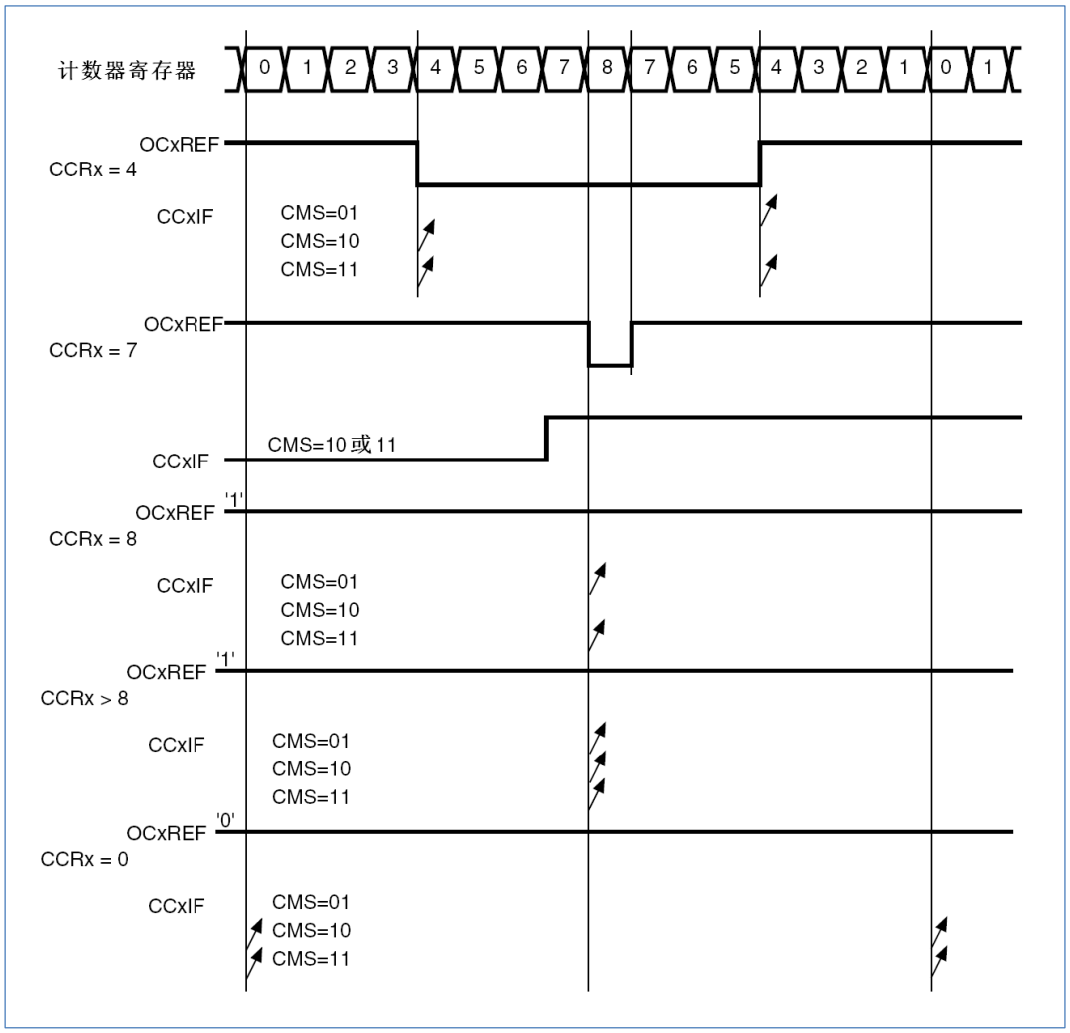


图 13-34 中央对齐的 PWM 波形（ARR=8）

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。需要注意的是：
  - 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。

13.2.11 互补输出和死区插入

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

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

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

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIM1_CCER 寄存器的 CCxE 和 CCxNE 位，

TIM1_BDTR 和 TIM1_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，参见表 13-3。需要注意，在转换到 IDLE 状态时（MOE 下降到 0）死区被激活。

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

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

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

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

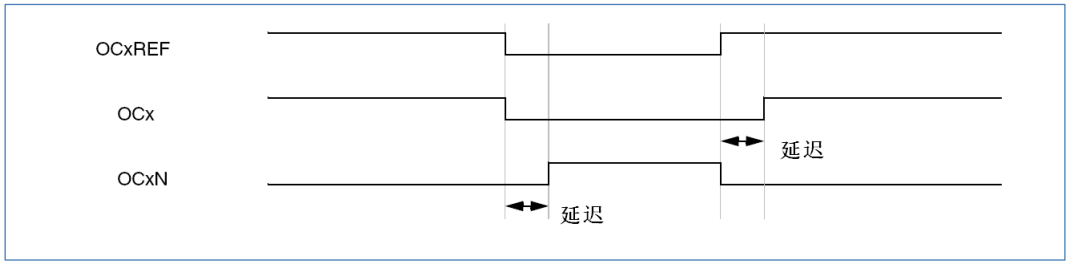


图 13-35 带死区插入的互补输出

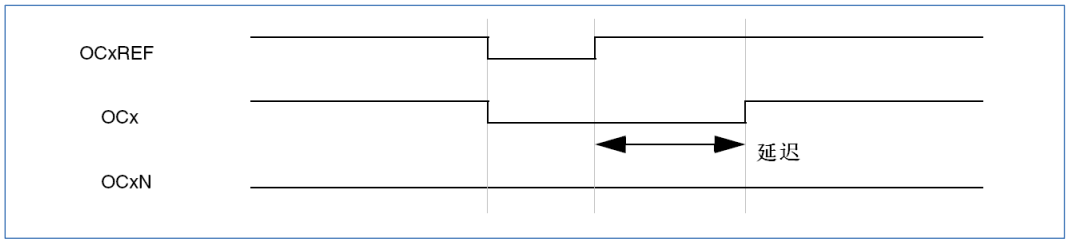


图 13-36 死区波形延迟大于负脉冲

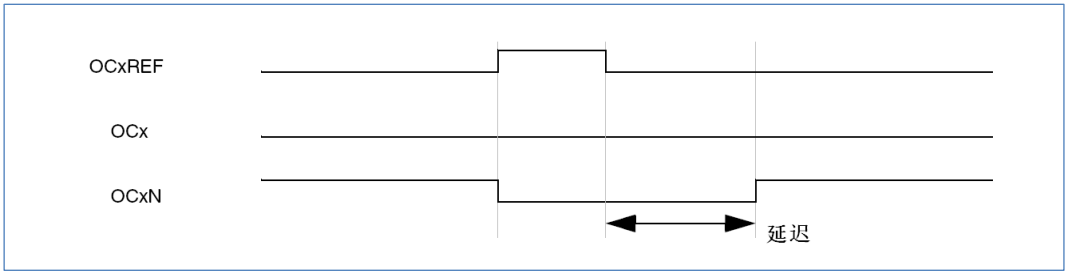


图 13-37 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIM1_BDTR 寄存器中的 DTG 位编程配置。参见“13.3.18 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。

13.2.12 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 13-3。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生，参见“6.2.9 时钟安全系统（CSS）”。

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

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

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIM1_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注意：因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIM1_DIER 寄存器中的 BIE 位，当刹车状态标志（TIM1_SR 寄存器中的 BIF 位）为’1’时，则产生一个中断。
- 如果设置了 TIM1_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

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 BRK 输入产生，它的有效极性是可编程的，且由 TIM1_BDTR 寄存器中的 BKE 位开启。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度、OCx/OCxN 极性和被禁止的状态、OCxM 配置、刹车使能和极性）。用户可以通过 TIM1_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参见“13.3.18 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。

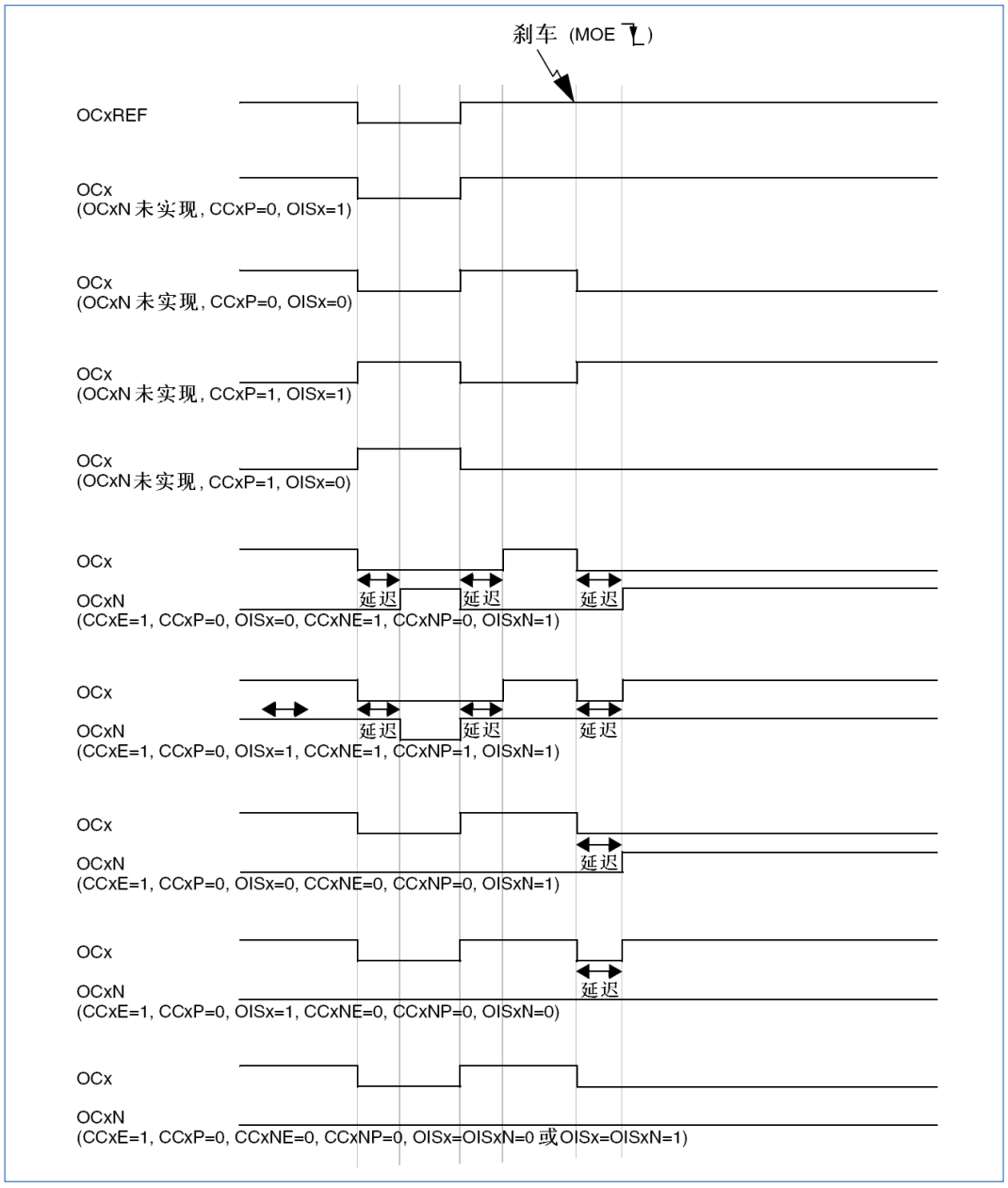


图 13-38 响应刹车的输出

13.2.13 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIM1_CCMRx 寄存器中对应的 OCxCE 位为'1'，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。

该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

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

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

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

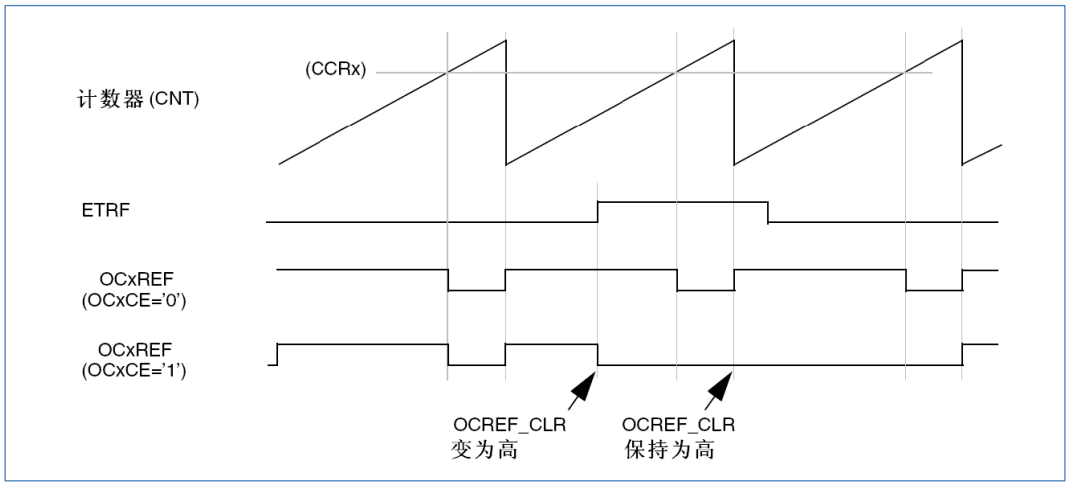


图 13-39 清除 TIM1 的 OCxREF

13.2.14 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样用户就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。COM 可以通过设置 TIM1_EGR 寄存器的 COMG 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。

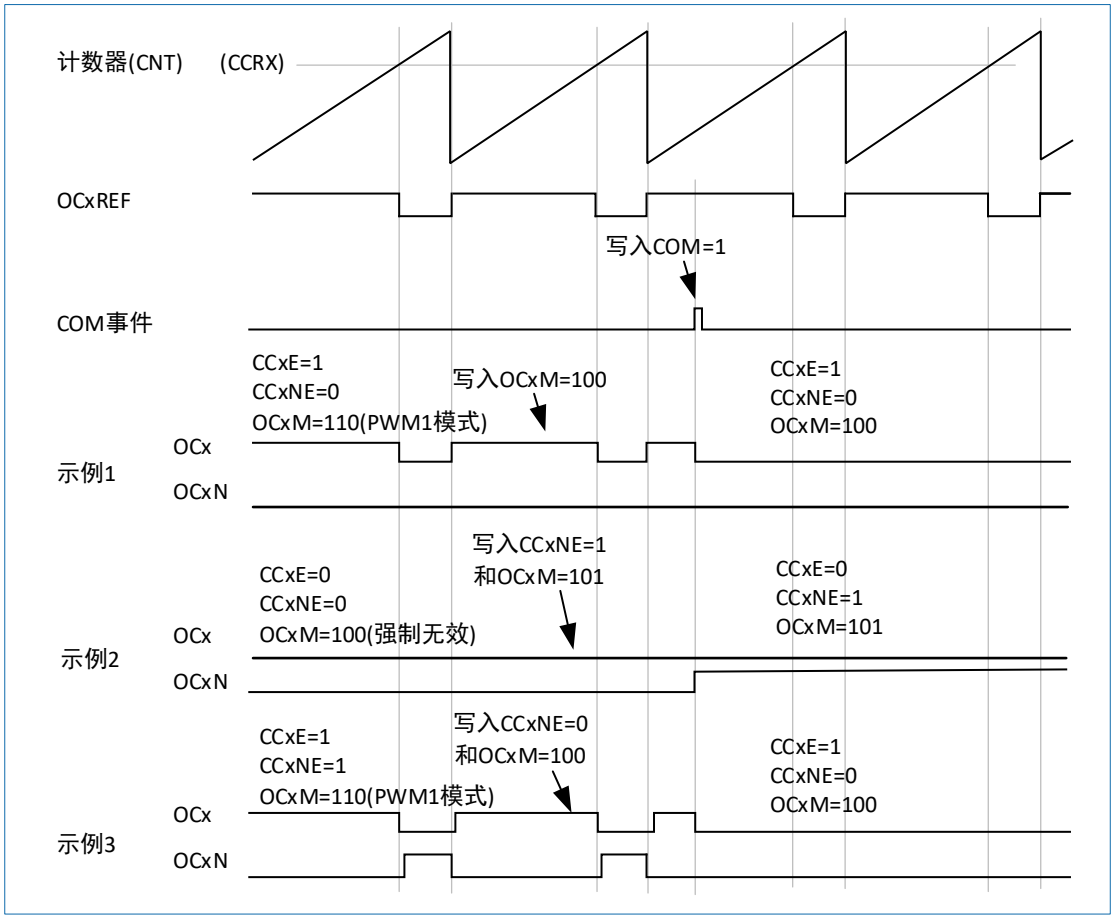


图 13-40 产生六步 PWM，使用 COM 的例子（OSSR=1）

13.2.15 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。

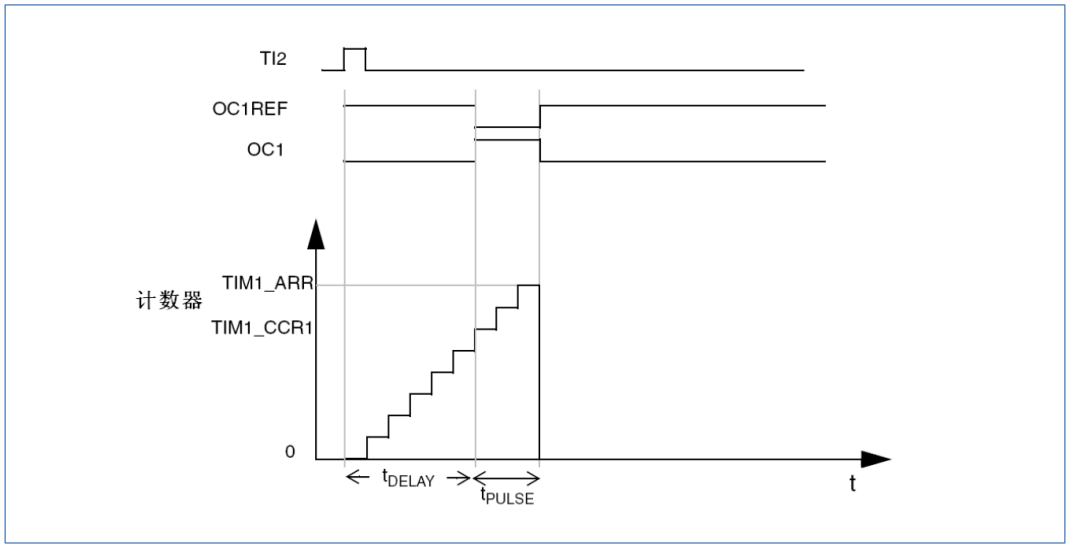


图 13-41 单脉冲模式的例子

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

假定 TI2FP2 作为触发 1：

- 置 TIM1_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
- 置 TIM1_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIM1_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
- 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。

13.2.16 编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI2 的边沿计数，则置 TIM1_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

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

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参见表 13-1，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 13-1 计数方向与编码器信号的关系

有效边沿	相对信号的电平（TI1FP1 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'（TIM1_CCMR1 寄存器，IC1FP1 映射到 TI1）
- CC2S='01'（TIM1_CCMR1 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）

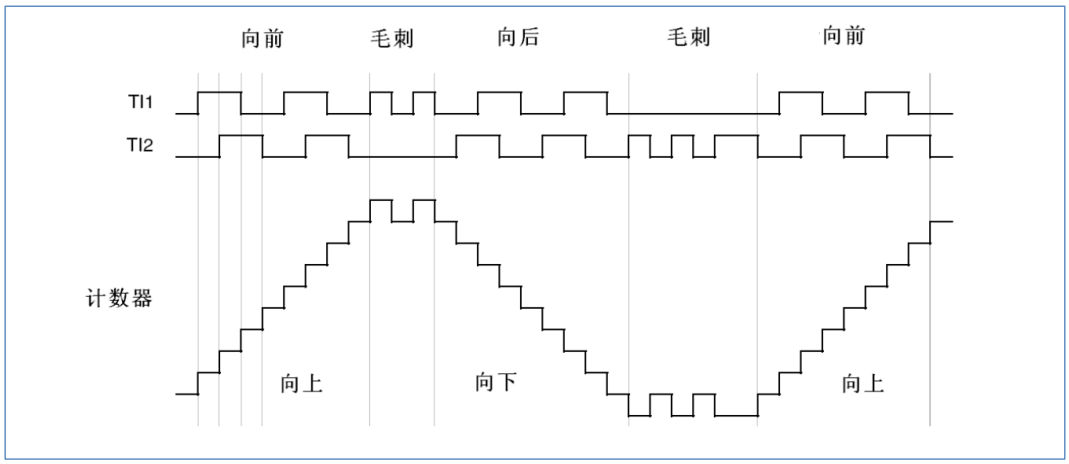


图 13-42 编码器模式下的计数器操作实例

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

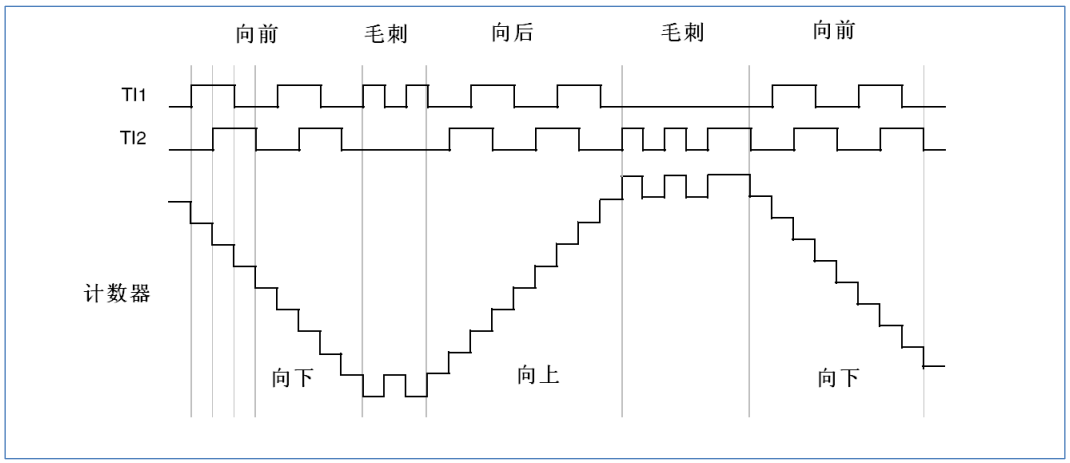


图 13-43 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度，加速度，减速度）。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，用户可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

13.2.17 定时器输入异或功能

TIM1_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIM1_CH1、TIM1_CH2 和 TIM1_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下一章节给出了此特性用于连接

霍尔传感器的例子。

13.2.18 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，可以用另一个通用定时器（TIM3）作为“接口定时器”来连接霍尔传感器，参见图 13-44。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 TIM1_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（参见图 13-27）。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

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

举例：霍尔输入连接到 TIM1 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 TIM1_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 TIM1_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 TIM1_CCMR1 寄存器中 CC1S=11，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 TIM1_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIM1_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的（TIM1_CR2 寄存器中 CCPC=1），同时触发输入控制 COM 事件（TIM1_CR2 寄存器中 CCUS=1）。在一次 COM 事件后，写入下一步的 PWM 控制位（CCxE、OCxM），这可以在处理 OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例。

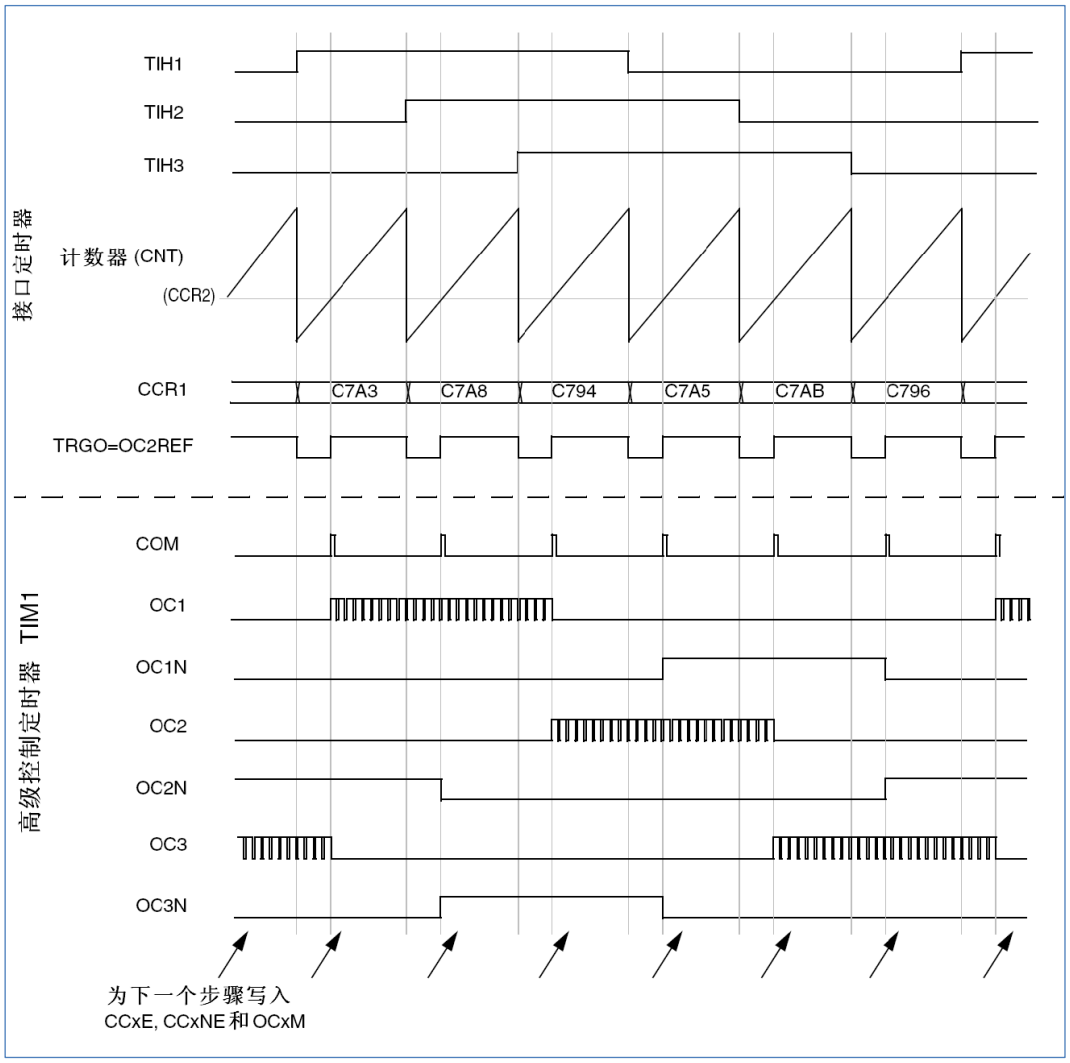


图 13-44 霍尔传感器接口的实例

13.2.19 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。

13.2.19.1 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 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。

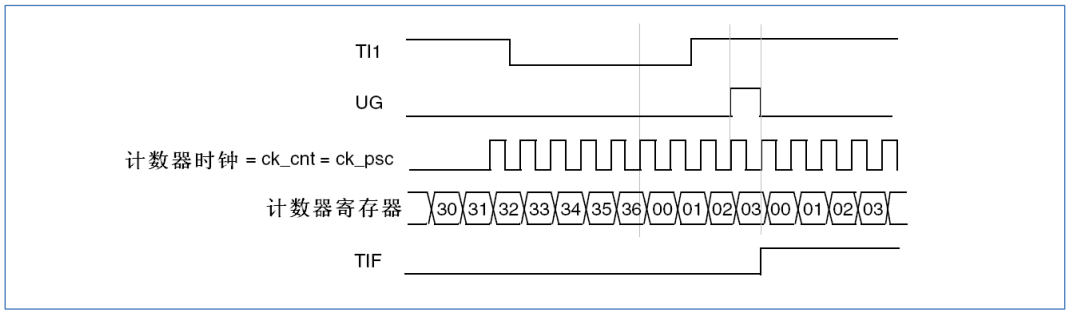


图 13-45 复位模式下的控制电路

13.2.19.2 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。

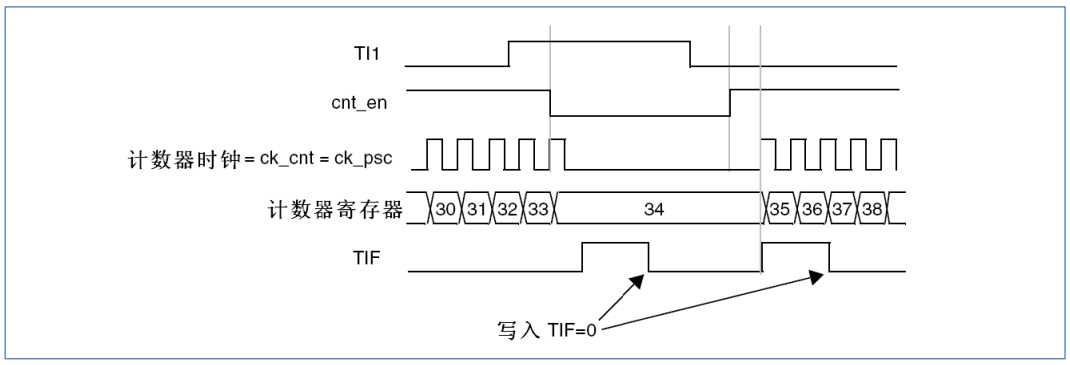


图 13-46 门控模式下的控制电路

13.2.19.3 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。

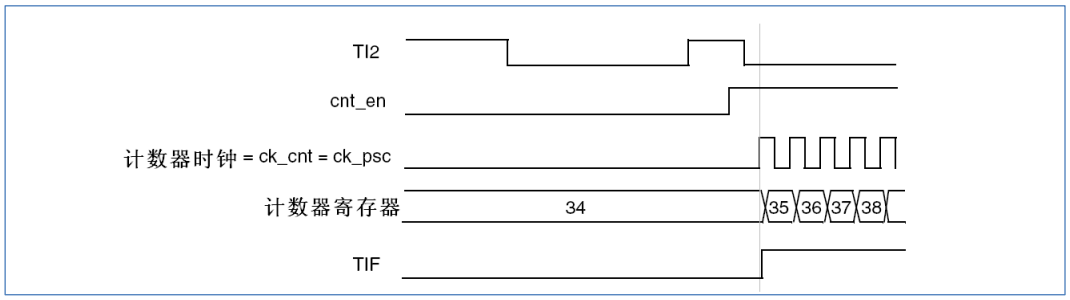


图 13-47 触发器模式下的控制电路

13.2.19.4 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。

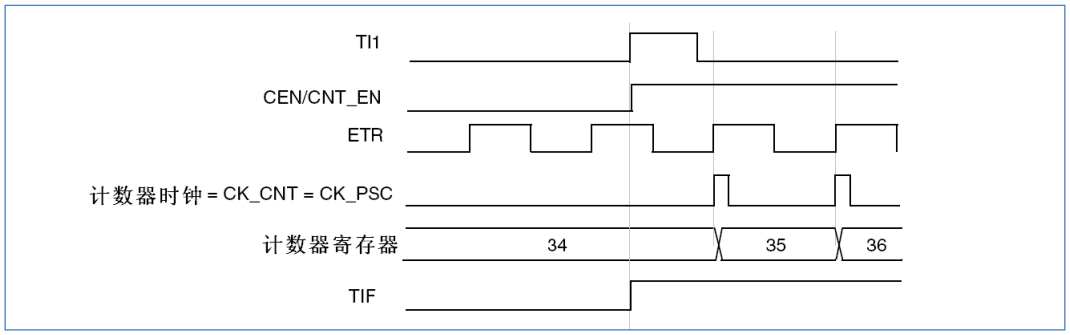


图 13-48 外部时钟模式 2＋触发模式下的控制电路

13.2.20 定时器同步

所有 TIM 定时器在内部相连，用于定时器同步或链接。参见章节“14.2.15 定时器同步”。

13.2.21 调试模式

当微控制器进入调试模式时（Cortex-M0 内核停止），根据 DBG 模块中 DBG_TIM1_Stop 的设置，TIM1 计数器可以继续正常操作或者停止。参见“26.8.2 对定时器、看门狗和 I2C 的调试支持”。

13.3 TIM1 寄存器

基地址：0x4001 2C00

空间大小：0x400

13.3.1 TIM1 控制寄存器 1（TIM1_CR1）

偏移地址：0x00

复位值：0x0000

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

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

位 31	CC4_ADC_SEL：ADC 的 CC4 触发信号的产生选择（ADC triger CC4 generation source selection） • 0：默认值；ADC 的 CC4 触发信号由端口 CH4 信号产生 • 1：ADC 的 CC4 触发信号改由 Timer 内部 OC4REF 信号产生
位 30:10	Res：保留 必须保持复位值。
位 9:8	CKD[1:0]：时钟分频因子（Clock division factor） 该位域定义了定时器时钟频率（CK_INT）与死区发生器和数字滤波器所使用的采样时钟（t_{DTS}）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：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，则下述任一事件将产生更新中断或 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

13.3.2 TIM1 控制寄存器 2（TIM1_CR2）

偏移地址：0x04

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res	CCPC
	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	OIS4：输出空闲状态 4（OC4 输出）（Output Idle state 4）（OC4output） 参见 OIS1 位。
位 13	OIS3N：输出空闲状态 3（OC3N 输出）（Output Idle state 3）（OC3N output） 参见 OIS1N 位。
位 12	OIS3：输出空闲状态 3（OC3 输出）（Output Idle state 3）（OC3output） 参见 OIS1 位。
位 11	OIS2N：输出空闲状态 2（OC2N 输出）（Output Idle state 2）（OC2N output） 参见 OIS1N 位。
位 10	OIS2：输出空闲状态 2（OC2 输出）（Output Idle state 2）（OC2output） 参见 OIS1 位。
位 9	OIS1N：输出空闲状态 1（OC1N 输出）（Output Idle state 1）（OC1N output） • 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 输出）（Output Idle state 1）（OC1output） • 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（TI1selection） • 0：TIM1_CH1 引脚连到 TI1 输入。 • 1：TIM1_CH1、TIM1_CH2 和 TIM1_CH 引脚经过异或后连到 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 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。

13.3.3 TIM1 从模式控制寄存器（TIM1_SMCR）

偏移地址：0x08

复位值：0x0000

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

位 18:16	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 TIM x CLK 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：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/向下计数。 • 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 13-2 TIM1 内部触发连接

从定时器	ITR0	ITR1	ITR2	ITR3
TIM1	-	TIM2	TIM3	TIM17

13.3.4 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/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 1DMA 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/Compare 4interrupt 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 3interrupt 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 2interrupt 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 1interrupt 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

13.3.5 TIM1 状态寄存器（TIM1_SR）

偏移地址：0x10

复位值：0x0000

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

位 15:13	Res：保留 必须保持复位值。
位 12	CC4OF：捕捉/比较 4 重复捕捉标志（Capture/Compare 4overcapture flag） 参见 CC1OF 位。
位 11	CC3OF：捕捉/比较 3 重复捕捉标志（Capture/Compare 3overcapture flag） 参见 CC1OF 位。
位 10	CC2OF：捕捉/比较 2 重复捕捉标志（Capture/Compare 2overcapture flag） 参见 CC1OF 位。

位 9	CC1OF：捕捉/比较 1 重复捕捉标志（Capture/Compare 1overcapture flag） 仅当相应的通道被配置为输入捕获时，该位可由硬件置 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/Compare 4interrupt flag） 参见 CC1IF 位。
位 3	CC3IF：捕捉/比较 3 中断标志（Capture/Compare 3interrupt flag） 参见 CC1IF 位。
位 2	CC2IF：捕捉/比较 2 中断标志（Capture/Compare 2interrupt flag） 参见 CC1IF 位。
位 1	CC1IF：捕捉/比较 1 中断标志（Capture/Compare 1interrupt 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，但在中心对称模式下除外。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_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 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'，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。

13.3.6 TIM1 事件产生寄存器（TIM1_EGR）

偏移地址：0x14

复位值：0x0000

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

位 15:8	Res：保留 必须保持复位值。
位 7	BG：产生刹车事件（Break generation） 该位由软件置'1'，用于产生一个刹车事件，由硬件自动清'0'。 ● 0：不起作用 ● 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） 该位用于产生一个事件，由软件置'1'，并由硬件自动清'0'。 ● 0：不起作用 ● 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：不起作用 ● 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。
位 4	CC4G：捕捉/比较 4 发生（Capture/Compare 4generation） 参见 CC1G 位。
位 3	CC3G：捕捉/比较 3 发生（Capture/Compare 3generation） 参见 CC1G 位。

位 2	CC2G：捕捉/比较 2 发生（Capture/Compare 2generation） 参见 CC1G 位。
位 1	CC1G：捕捉/比较 1 发生（Capture/Compare 1generation） 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 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：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）。若在中心对称模式下或 DIR=0（向上计数）则计数器被清’0’；若 DIR=1（向下计数）则计数器取 TIM1_ARR 的值。

13.3.7 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。

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	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 1mode） 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 TIM1_CCR1 与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用。 • 001：匹配时，设置通道 1 为有效电平。 当计数器 TIM1_CNT 的值与捕获/比较寄存器 1（TIM1_CCR1）相同时，强制 OC1REF 为高。 • 010：匹配时，设置通道 1 为无效电平。 当计数器 TIM1_CNT 的值与捕获/比较寄存器 1（TIM1_CCR1）相同时，强制 OC1REF 为低。 • 011：翻转 当 TIM1_CCR1=TIM1_CNT 时，翻转 OC1REF 的电平。 • 100：强制为无效电平 强制 OC1REF 为低。 • 101：强制为有效电平 强制 OC1REF 为高。 • 110：PWM 模式 1 ○ 在向上计数时，一旦 TIM1_CNT<TIM1_CCR1 时通道 1 为有效电平，否则为无效电平； ○ 在向下计数时，一旦 TIM1_CNT>TIM1_CCR1 时通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。 • 111：PWM 模式 2 ○ 在向上计数时，一旦 TIM1_CNT<TIM1_CCR1 时通道 1 为无效电平，否则为有效电平。 ○ 在向下计数时，一旦 TIM1_CNT>TIM1_CCR1 时通道 1 为有效电平，否则为无效电平。
位 3	OC1PE：输出比较 1 预装允许 （Output Compare 1 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 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象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
位 1:0	CC1S[1:0]：捕捉/比较 1 选择（Capture/Compare 1 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 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[2:0]：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 TIM1_SMCR 寄存器的 TS 位选择）。
位 7:4	IC1F[3:0]：输入捕捉 1 滤波器（Input capture 1 filter） 该位域定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： • 0000：无滤波器，以 f_{DTS} 采样 • 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 • 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 • 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 • 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 • 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8

	• 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 • 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 • 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 • 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 • 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 • 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 • 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 • 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 • 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 • 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
位 3:2	IC1PSC：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（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：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 TIM1_SMCR 寄存器的 TS 位选择）。

13.3.8 TIM1 捕捉/比较模式寄存器 2（TIM1_CCMR2）

偏移地址：0x1C

复位值：0x0000

参见 13.3.7 TIM1 捕捉/比较模式寄存器 1（TIM1_CCMR1）中的描述。

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	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 TIM1_CCMR1.OC1CE 位。
位 14:12	OC4M[2:0]：输出比较模式 4（Output compare 4 mode） 参见 TIM1_CCMR1.OC1M 位。

位 11	OC4PE：输出比较 4 预装允许（Output compare 4preload enable） 参见 TIM1_CCMR1.OC1PE 位。
位 10	OC4FE：输出比较 4 快速允许（Output compare 4fast enable） 参见 TIM1_CCMR1.OC1FE 位。
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare 4selection） 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 清零允许（Output compare 3clear 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[3:0]：输出比较 3 模式（Output compare 3mode） 参见 TIM1_CCMR1.OC1M 位。
位 3	OC3PE：输出比较 3 预装允许（Output compare 3preload enable） 参见 TIM1_CCMR1.OC1PE 位。
位 2	OC3FE：输出比较 3 快速使能 （Output compare 3fast enable） 参见 TIM1_CCMR1.OC1FE 位。
位 1:0	CC3S[1:0]：捕捉/比较 3 选择（Capture/Compare 3selection） 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 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 capture 4filter） 参见 IC3F 位。
位 11:10	IC4PSC[1:0]：输入捕捉 4 预分频器（Input capture 4prescaler） 参见 IC3PSC 位。
位 9:8	CC4S[1:0]：捕捉/比较 4 选择（Capture/Compare 4selection） 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 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 capture 3filter） 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 3prescaler） 该位域定义了 CC3 输入（IC3）的预分频系数。一旦 CC3E=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	CC3S[1:0]：捕捉/比较 3 选择（Capture/Compare 3selection） 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 TIM1_SMCR 寄存器的 TS 位选择）。

13.3.9 TIM1 捕捉/比较使能寄存器（TIM1_CCER）

偏移地址：0x20

复位值：0x0000

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

位 15:14	Res：保留 必须保持复位值。
位 13	CC4P：捕捉/比较 4 输出极性（Capture/Compare 4output polarity） 参见 CC1P 位。
位 12	CC4E：捕捉/比较 4 输出使能（Capture/Compare 4output enable） 参见 CC1E 位。
位 11	CC3NP：捕捉/比较 3 互补输出极性（Capture/Compare 3complementary output polarity） 参见 CC1NP 位。
位 10	CC3NE：捕捉/比较 3 互补输出使能（Capture/Compare 3complementary output enable） 参见 CC1NE 位。
位 9	CC3P：捕捉/比较 3 输出极性（Capture/Compare 3output polarity） 参见 CC1P 位。
位 8	CC3E：捕捉/比较 3 输出使能（Capture/Compare 3output enable） 参见 CC1E 位。
位 7	CC2NP：捕捉/比较 2 互补输出极性（Capture/Compare 1complementary output polarity） 参见 CC1NP 位。
位 6	CC2NE：捕捉/比较 2 互补输出使能（Capture/Compare 2complementary output enable） 参见 CC1NE 位。
位 5	CC2P：捕捉/比较 2 输出极性（Capture/Compare 2output polarity） 参见 CC1P 位。
位 4	CC2E：捕捉/比较 2 输出使能（Capture/Compare 2output enable） 参见 CC1E 位。
位 3	CC1NP：捕捉/比较 1 互补输出极性（Capture/Compare 1complementary 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/Compare 1complementary output enable） • 0：关闭 OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 • 1：开启 OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
位 1	CC1P：捕捉/比较 1 输出极性（Capture/Compare 1output 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/Compare 1output enable） • CC1 通道配置为输出： ○ 0：关闭 OC1 禁止输出，因此 OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 ○ 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 13-3 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	OSSI	OSSR	CCxE	CCxNE	OCx 输出状态	OCxN 输出状态

控制位					输出状态 (1)	
位	位	位	位	位		
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		

13.3.10 TIM1 计数器寄存器（TIM1_CNT）

偏移地址：0x24

复位值：0x0000

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

位 15:0	CNT[15:0]：计数器值（Counter value）

13.3.11 TIM1 预分频器寄存器（TIM1_PSC）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频器的值（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'。

13.3.12 TIM1 自动重装载寄存器（TIM1_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。

13.3.13 TIM1 重复计数寄存器（TIM1_RCR）

偏移地址：0x30

复位值：0x0000

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

位 15:8	Res：保留 必须保持复位值。
位 7:0	REP[7:0]：重复计数器的值（Repetition counter value） 预装载寄存器被使能后，该位域允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIM1_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。这意味着在 PWM 模式中，（REP+1）对应着： • 在边沿对齐模式下，PWM 周期的数目。 • 在中心对称模式下，PWM 半周期的数目。

13.3.14 TIM1 捕捉/比较寄存器 1（TIM1_CCR1）

偏移地址：0x34

复位值：0x0000

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

位 15:0	CCR1[15:0]：捕捉/比较通道 1 的值（Capture/Compare 1 value） • 若 CC1 通道配置为输出： CCR1 决定了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIM1_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。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。

13.3.15 TIM1 捕捉/比较寄存器 2（TIM1_CCR2）

偏移地址：0x38

复位值：0x0000

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

位 15:0	CCR2[15:0]：捕捉/比较通道 2 的值（Capture/Compare 2 value） • 若 CC2 通道配置为输出： CCR2 决定了装入当前捕获/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。

13.3.16 TIM1 捕捉/比较寄存器 3（TIM1_CCR3）

偏移地址：0x3C

复位值：0x0000

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

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

	• 若 CC3 通道配置为输出： CCR3 决定了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIM1_CCMR2 寄存器（OC3PE 位）中未选择预装载功能，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。

13.3.17 TIM1 捕捉/比较寄存器 4（TIM1_CCR4）

偏移地址：0x40

复位值：0x0000

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

位 15:0	CCR4[15:0]：捕捉/比较通道 4 的值（Capture/Compare 4 value） • 若 CC4 通道配置为输出： CCR4 决定了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIM1_CCMR2 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。

13.3.18 TIM1 刹车和死区寄存器（TIM1_BDTR）

偏移地址：0x44

复位值：0x0000

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

位 15	MOE：主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，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：刹车输入低电平有效 • 1：刹车输入高电平有效
位 12	BKE：刹车使能（Break enable） • 0：刹车输入禁止（BRK 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：空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOE=0 时通道为输出。 • 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）。 • 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 被强制输出空闲电平。然后置 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 DTG、BKE、BKP、AOE 位和 TIM1_CR2 寄存器的 OISx/OISxN 位。 • 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIM1_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位。 • 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIM1_CCMRx 寄存器的 OCxM/OCxPE 位）。
位 7:0	DTG[7:0]：死区发生器设置（Dead-time generator setup） 该位域定义了插入互补输出之间的死区持续时间。

13.3.19 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） 该位域定义了 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。

13.3.20 TIM1 全部传输时 DMA 地址寄存器（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： 1. 将相应的 DMA 通道配置如下：

	○ DMA 通道外设地址为 TIM1_DMAR 寄存器地址。 ○ 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。 ○ 要传输的数据量为 3（参见下文“注意”）。 ○ 禁止循环模式。 2. 配置 TIM1_DCR 寄存器的 DBA 和 DBL 位如下： ○ DBL = 3 次传输，DBA = 0xE。 3. 使能 TIM1 更新 DMA 请求（将 TIM1_DIER.UDE 置 1）。 4. 使能 TIM1（将 TIM1_CR1.CEN 置 1）。 5. 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。

14 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（TIM3 是 16 位，TIM2 是 32 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“14.2.15 定时器同步”。

14.1 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下降沿触发和双沿触发功能
- 16/32 位（TIM3 是 16 位，TIM2 是 32 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

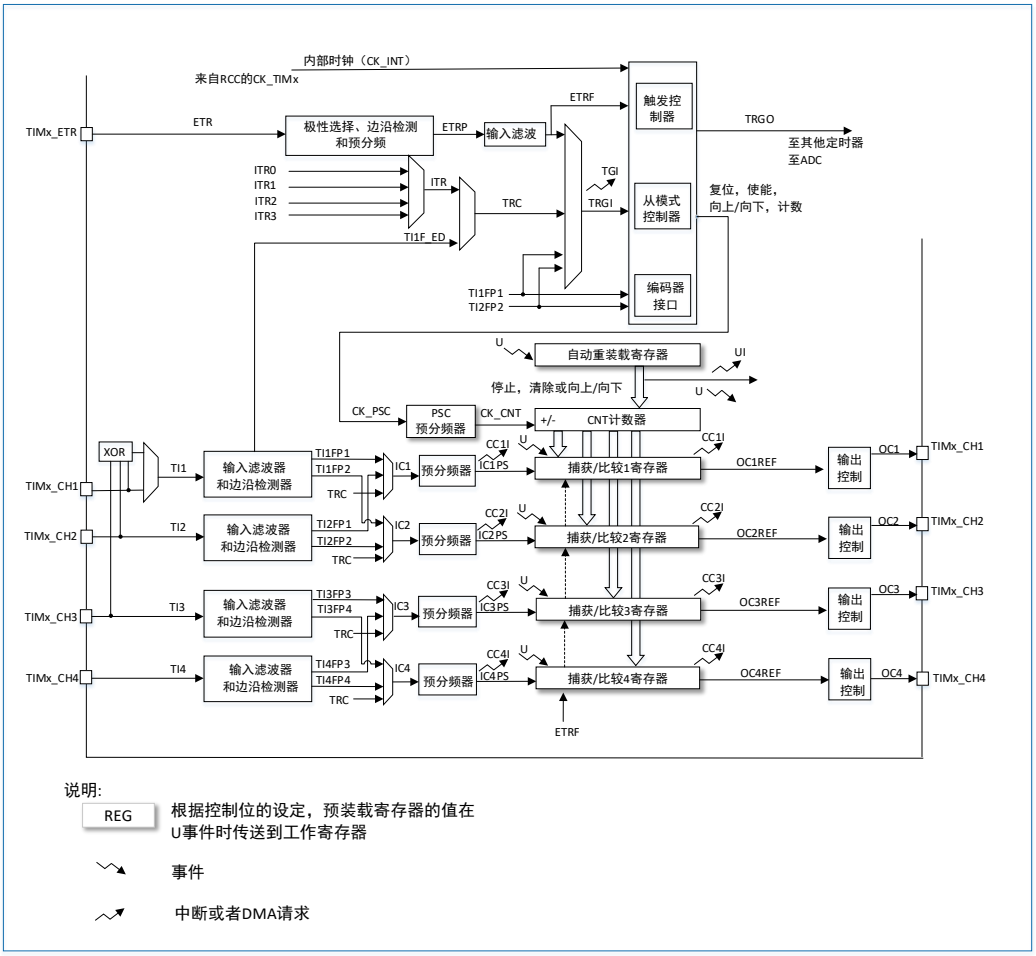


图 14–1 通用定时器框图

14.2 TIM2 和 TIM3 功能描述

14.2.1 时基单元

可编程通用定时器的主要部分是一个 16/32 位（TIM3 是 16 位，TIM2 是 32 位）计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，且在计数器运行时仍可以读写。

时基单元包含：

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

预分频器描述

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

下面两个图给出了在预分频器运行时，更改计数器参数的例子。

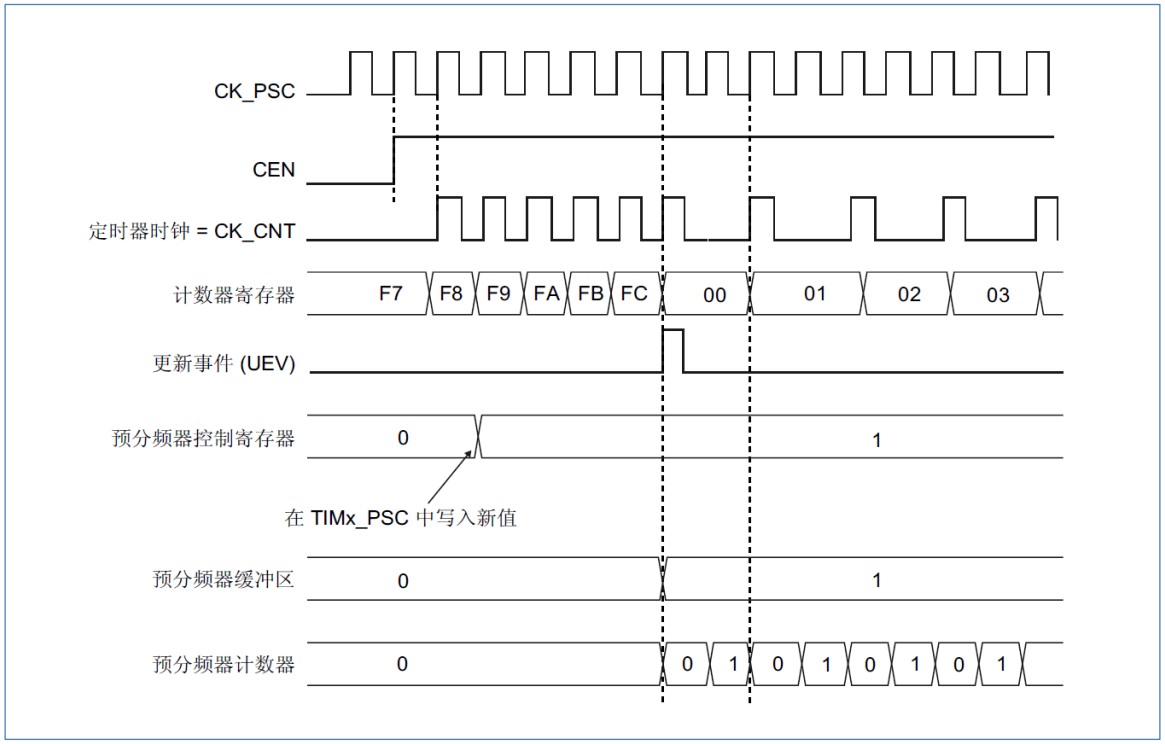


图 14-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

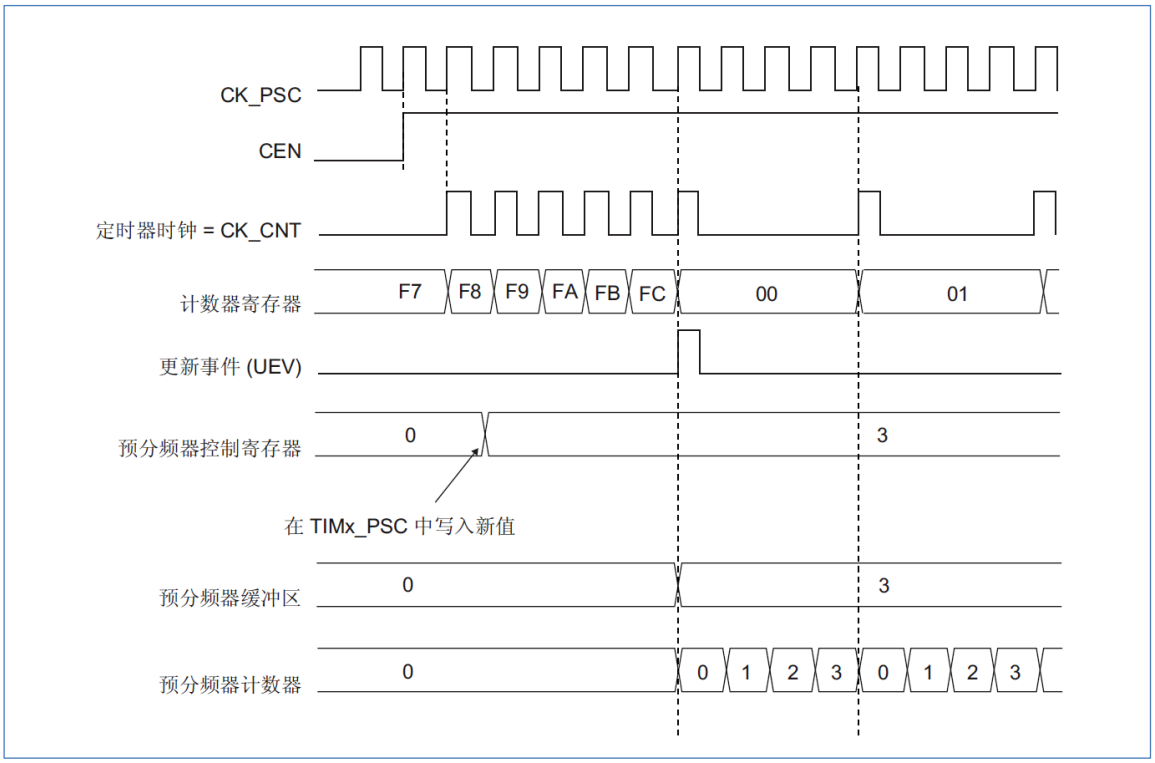


图 14-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

14.2.2 计数器模式

14.2.2.1 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。

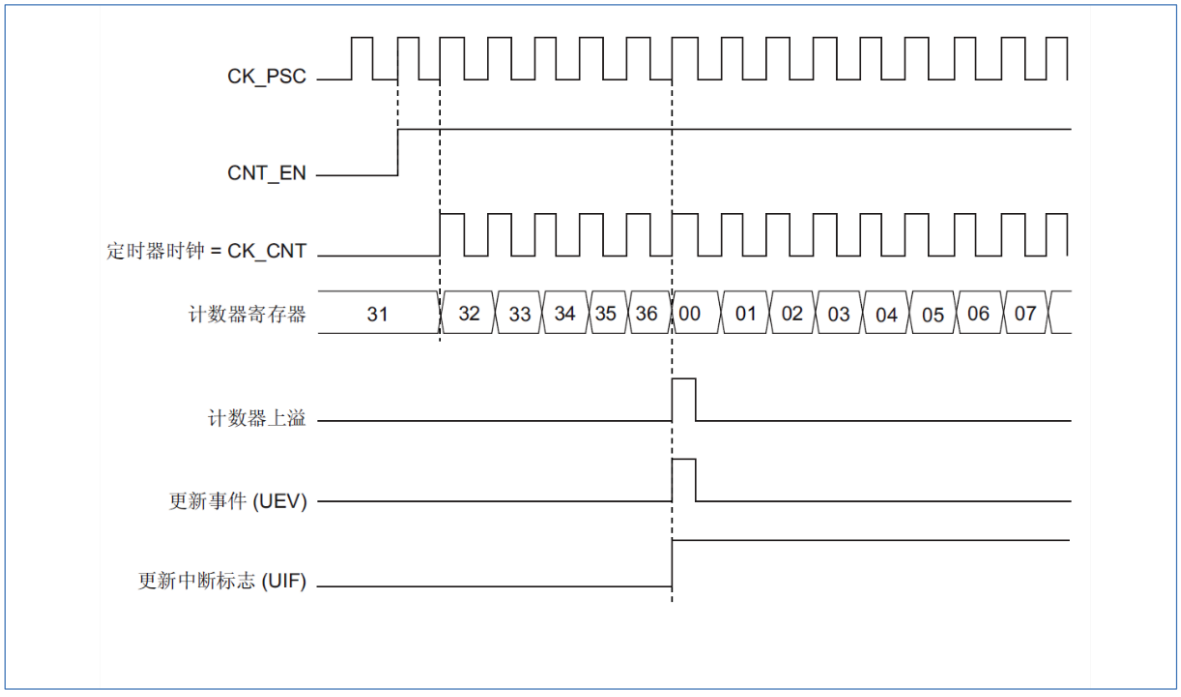


图 14-4 计数器时序图，内部时钟分频因子为 1

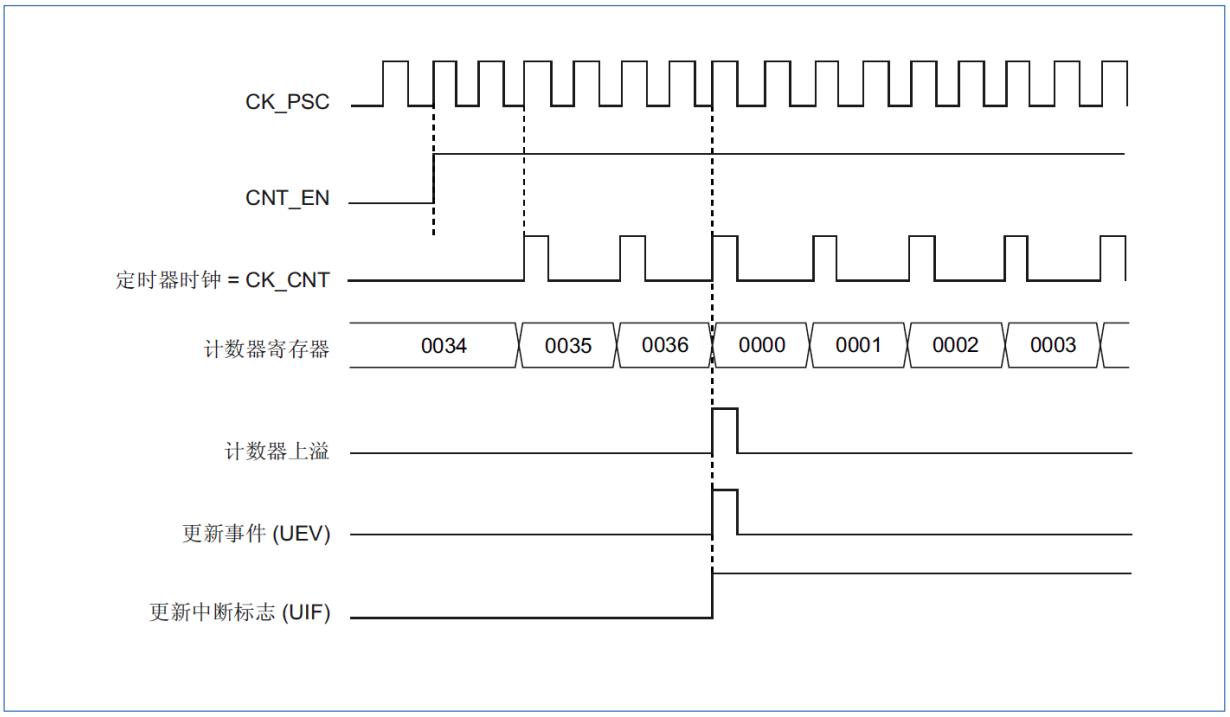


图 14-5 计数器时序图，内部时钟分频因子为 2

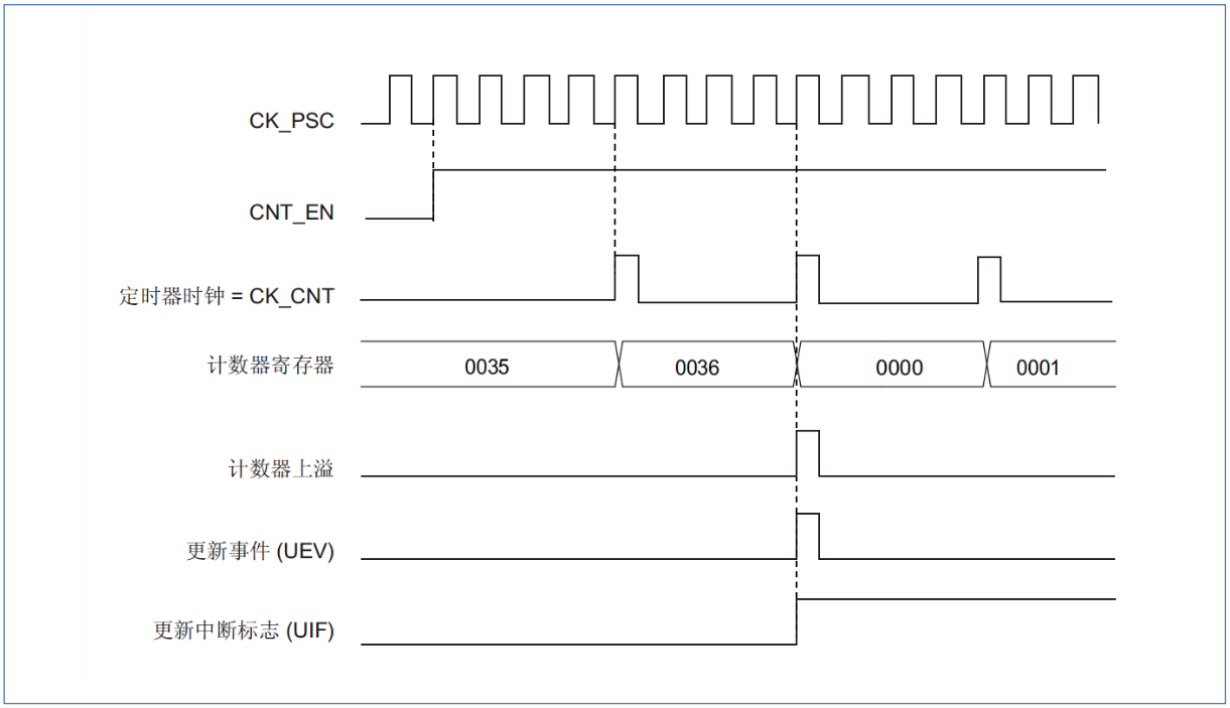


图 14-6 计数器时序图，内部时钟分频因子为 4

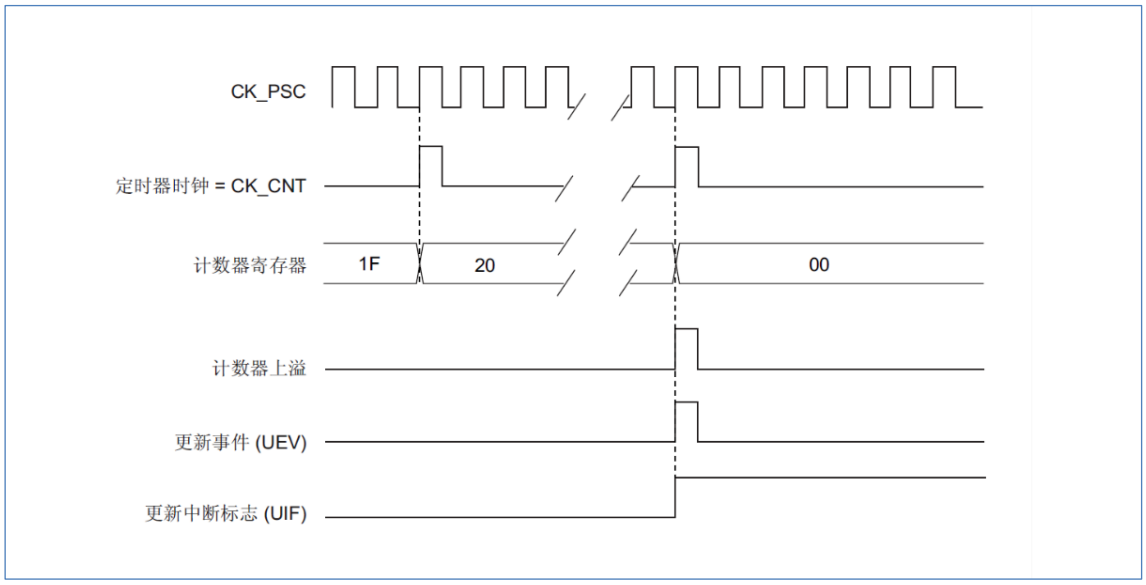


图 14-7 计数器时序图，内部时钟分频因子为 N

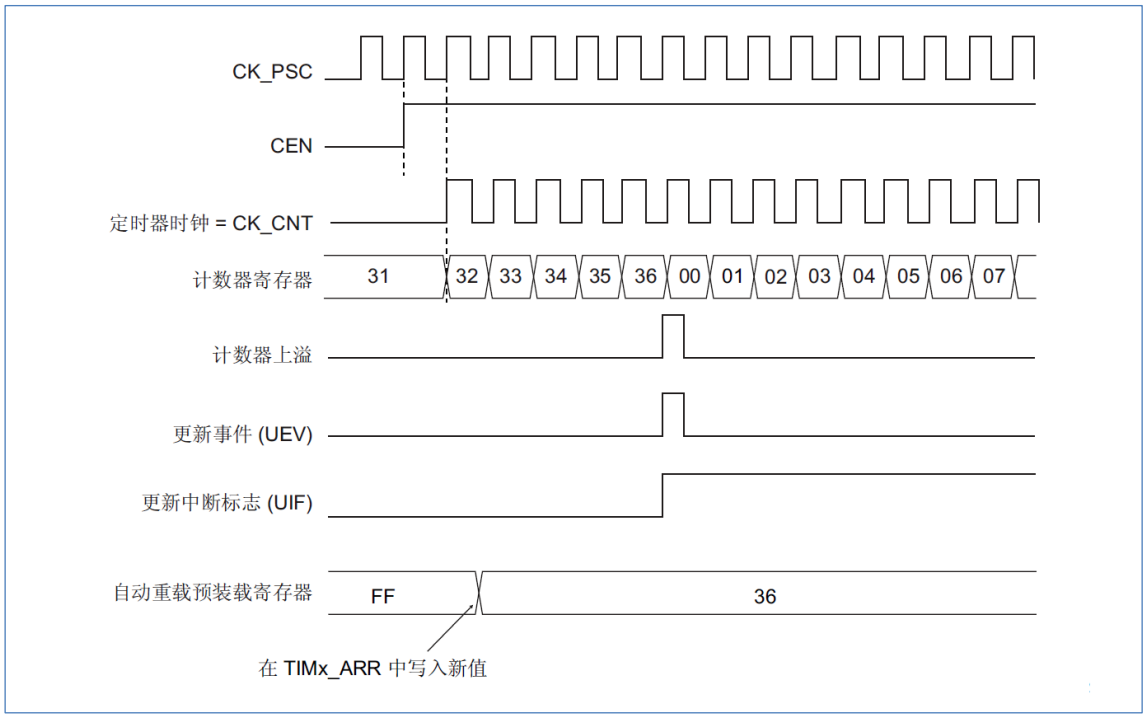


图 14-8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

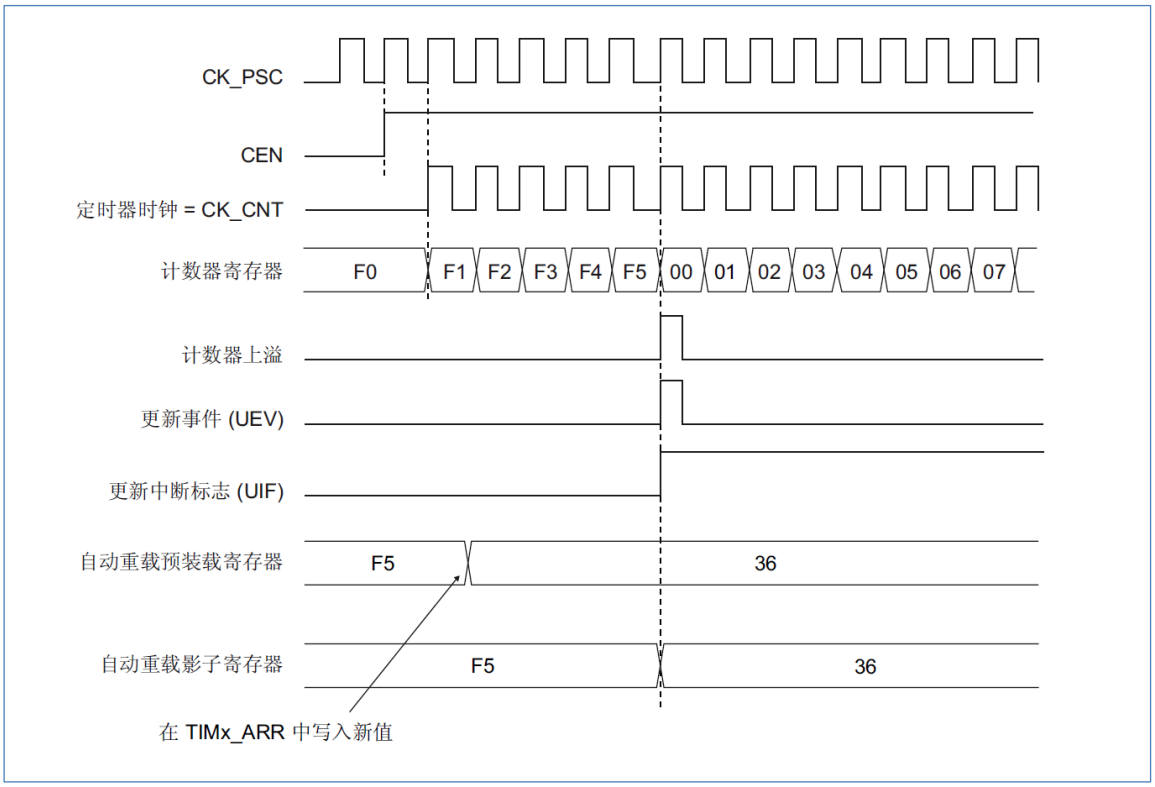


图 14-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

14.2.2.2 向下计数模式

在向下模式中，计数器从自动装入的值（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。

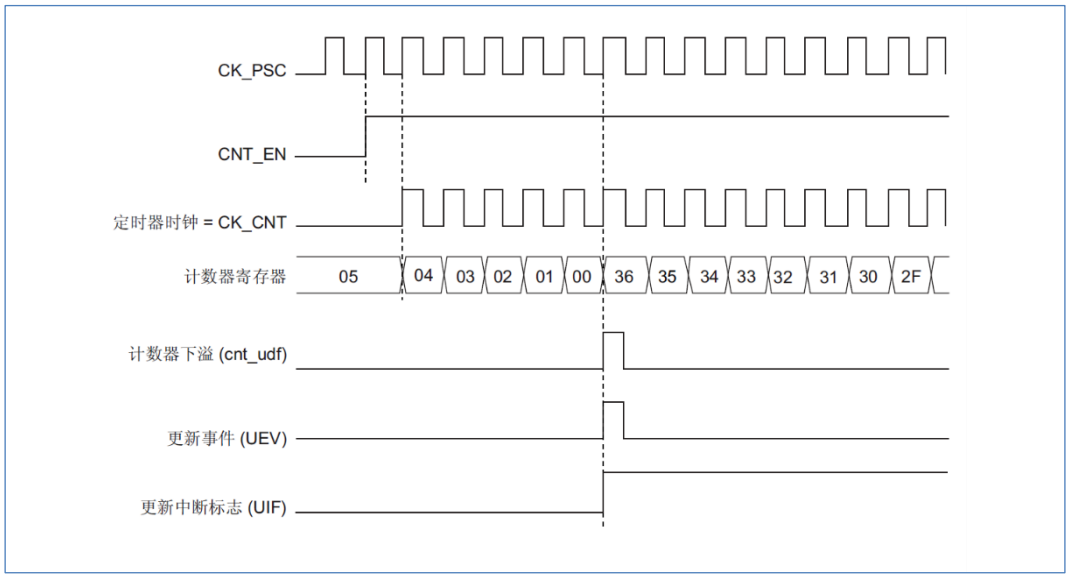


图 14-10 计数器时序图，内部时钟分频因子为 1

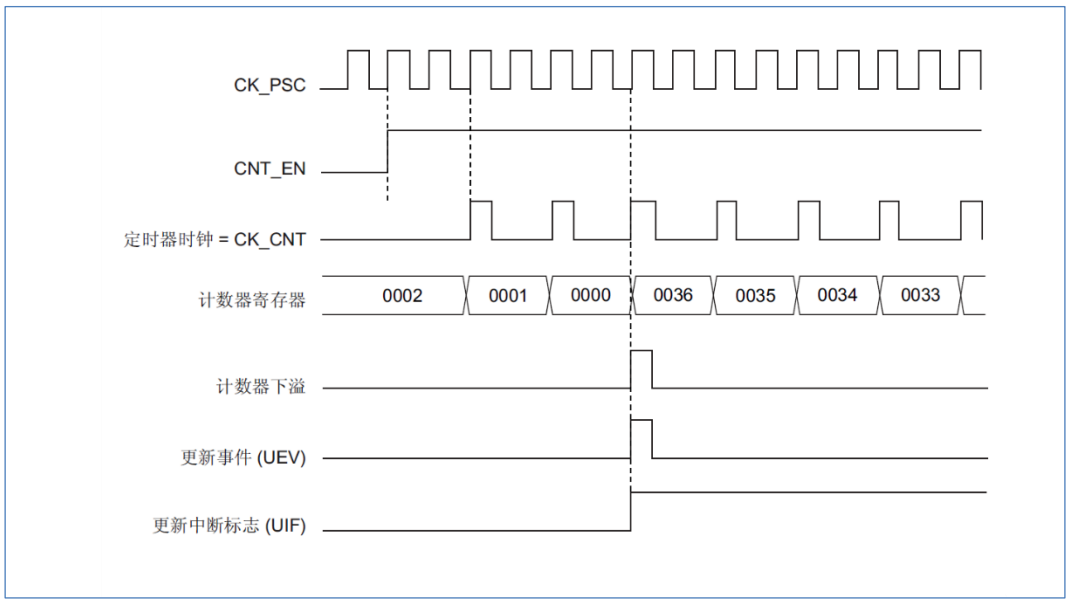


图 14-11 计数器时序图，内部时钟分频因子为 2

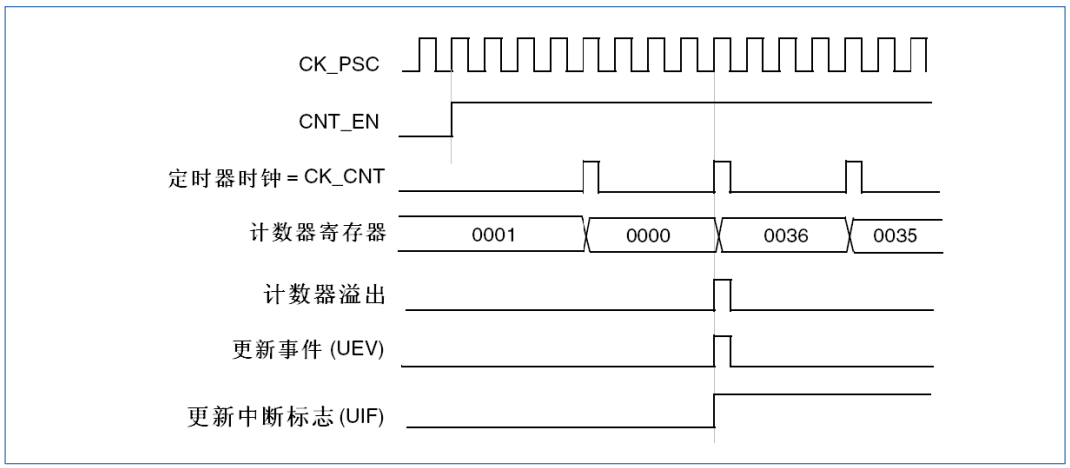


图 14-12 计数器时序图，内部时钟分频因子为 4

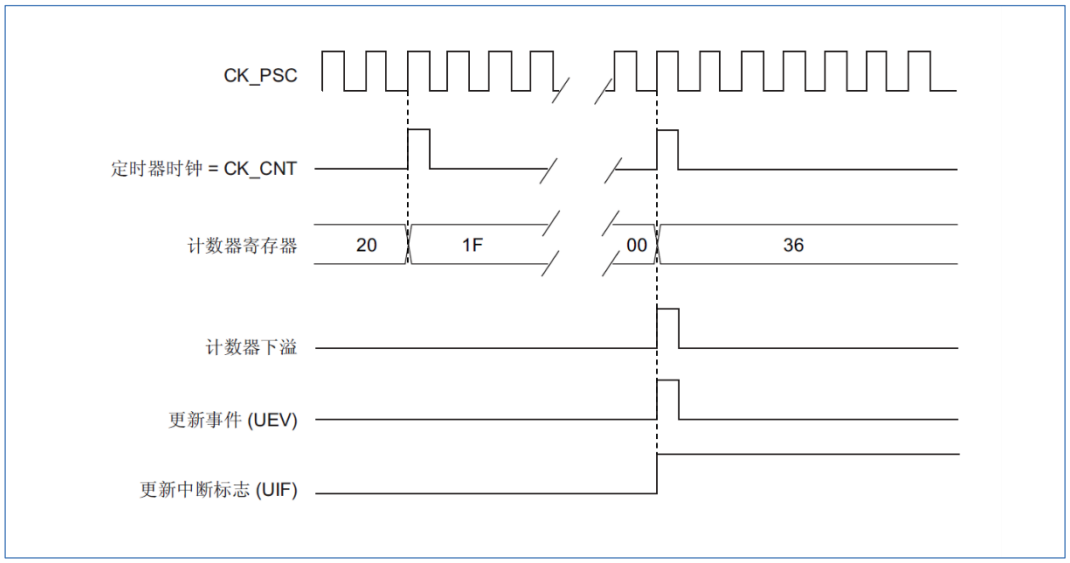


图 14-13 计数器时序图，内部时钟分频因子为 N

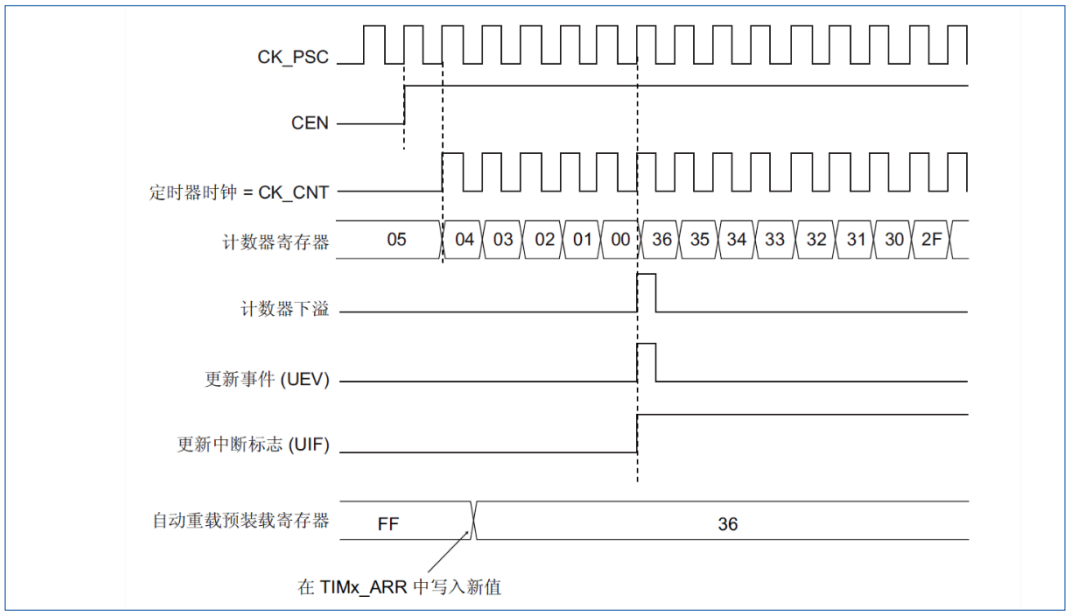


图 14-14 计数器时序图，当没有使用重复计数器时的更新事件

14.2.2.3 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：

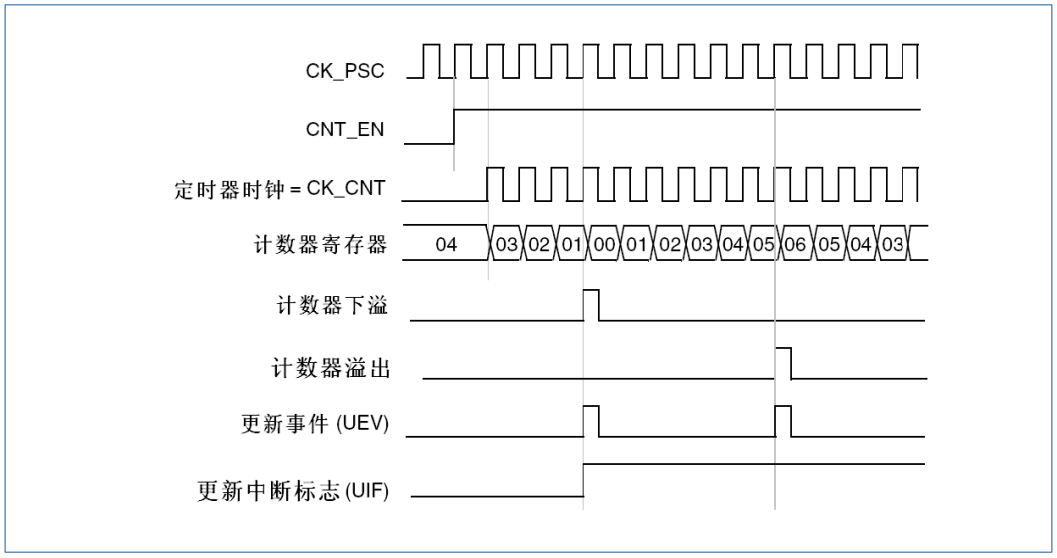


图 14-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6（这里使用了中央对齐模式 1）

图 14-15 使用了中央对齐模式 1（参见“14.3.1 TIM2/3 控制寄存器 1（TIMx_CR1）（x=2..3）”）。

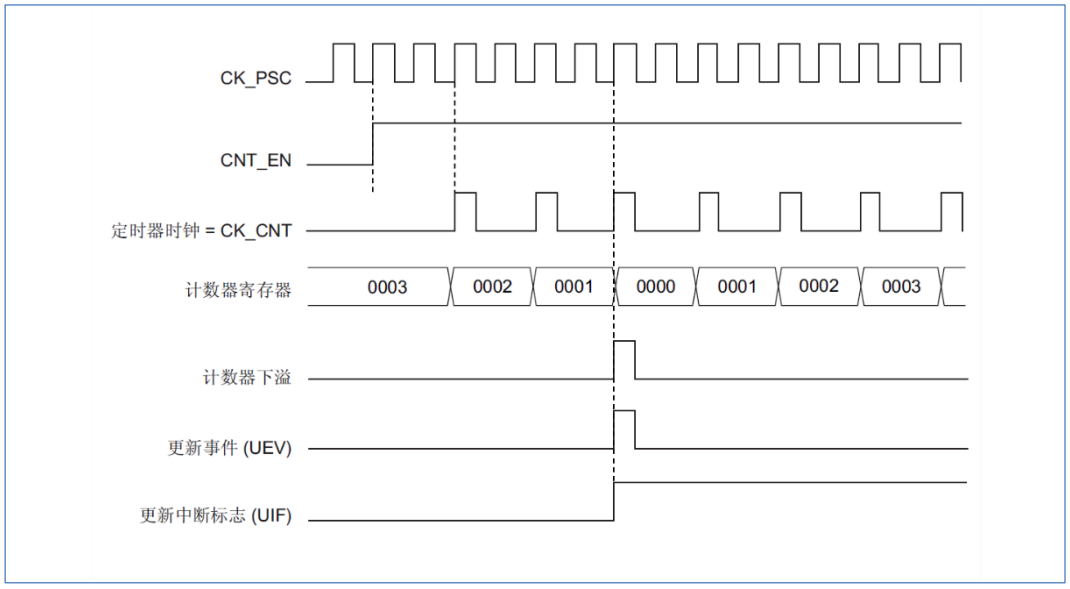


图 14-16 计数器时序图，内部时钟分频因子为 2（这里使用了中央对齐模式 1）

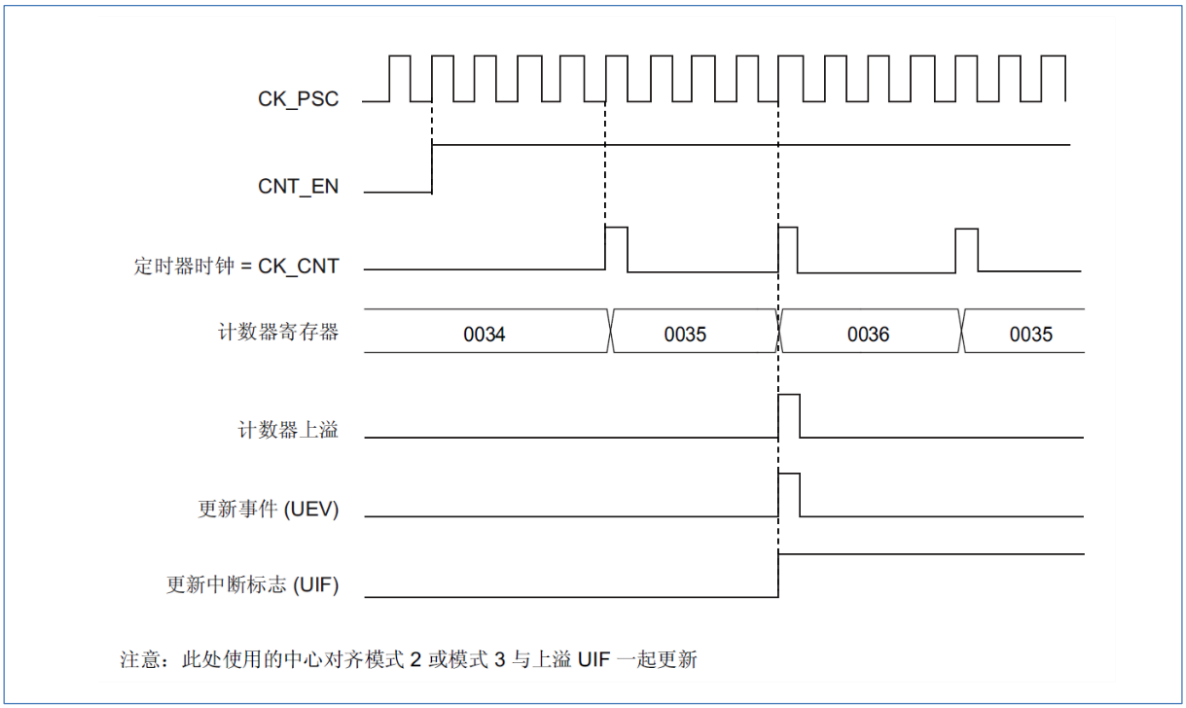


图 14-17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36（这里使用了中央对齐模式 2 或 3）

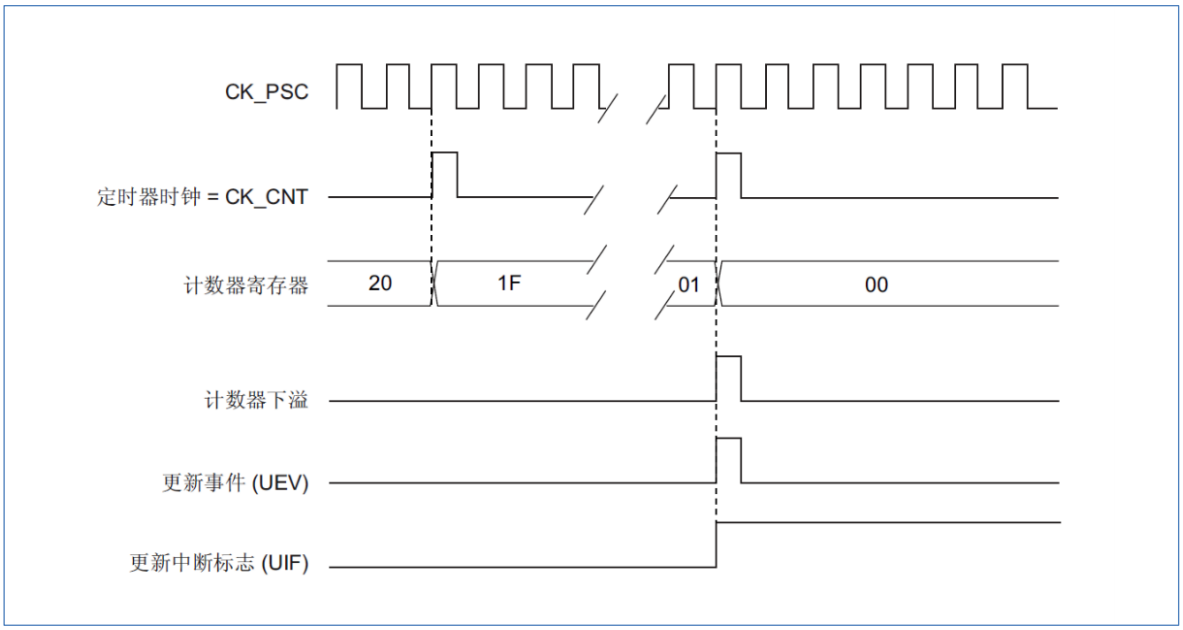


图 14-18 计数器时序图，内部时钟分频因子为 N（这里使用了中央对齐模式 1）

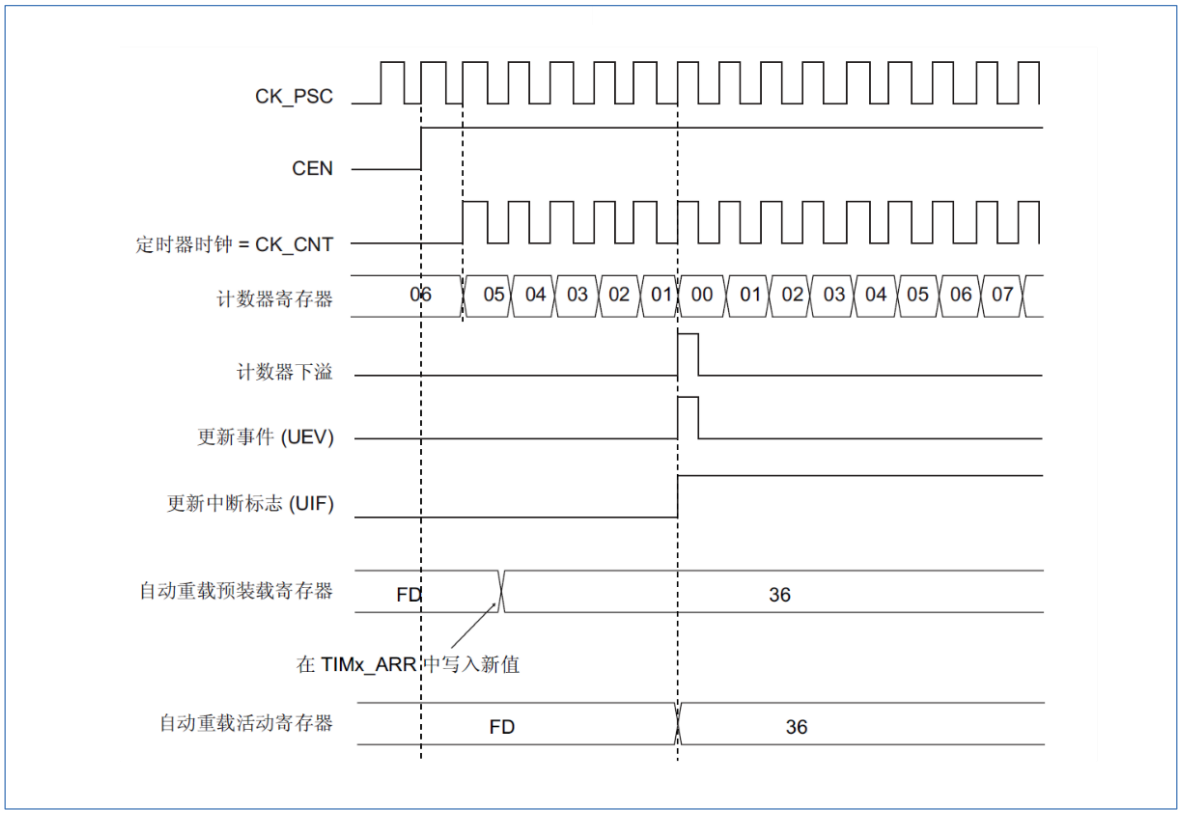


图 14-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

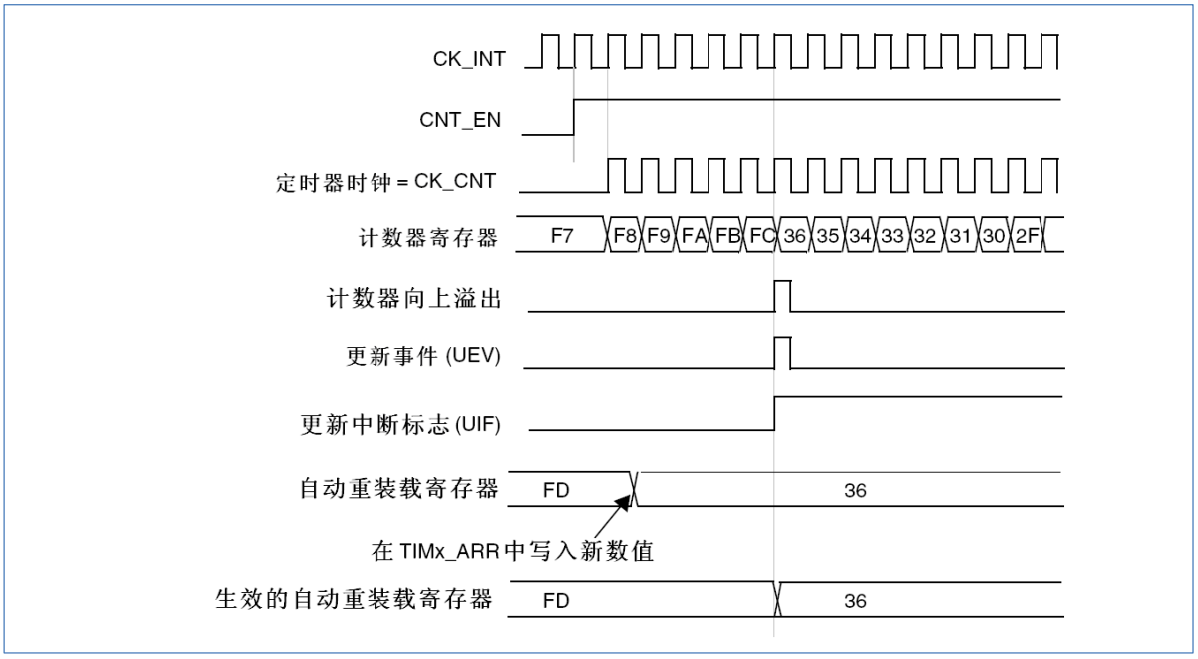


图 14-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

14.2.3 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“14.2.15.1 使用一个定时器作为另一个定时器的预分频器”。

14.2.3.1 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。

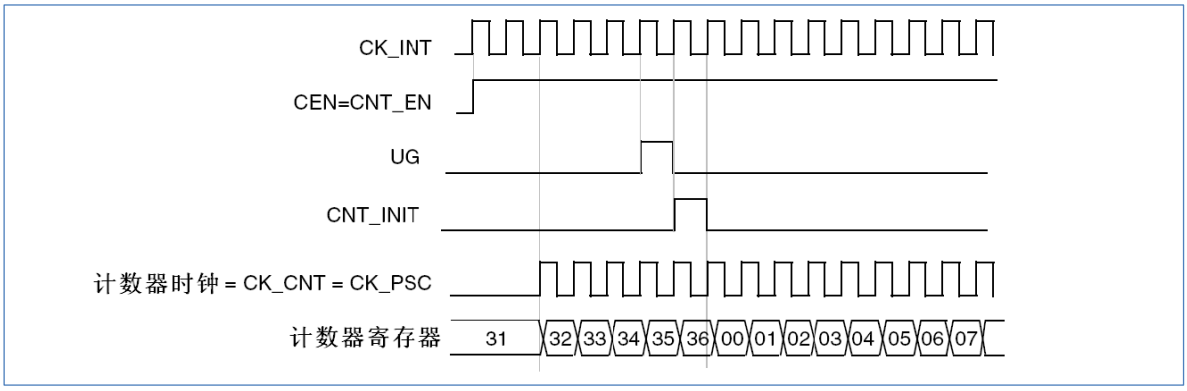


图 14-21 一般模式下的控制电路，内部时钟分频因子为 1

14.2.3.2 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。

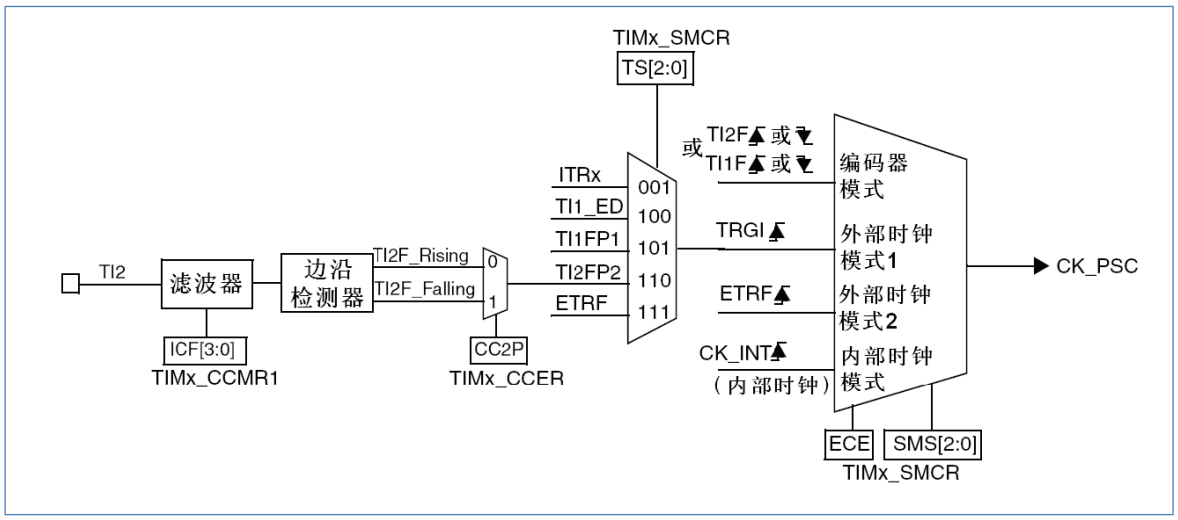


图 14-22 TI2 外部时钟连接例子

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 配置 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。

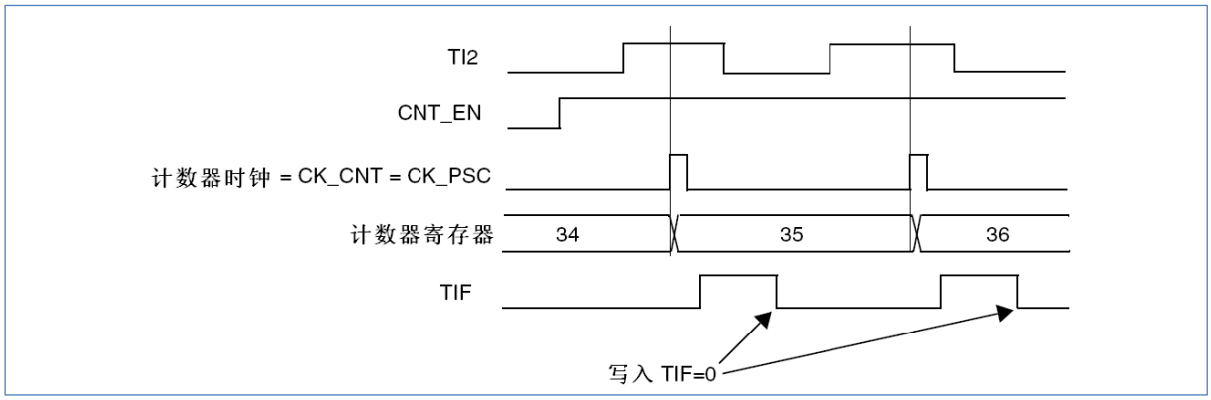


图 14-23 外部时钟模式 1 下的控制电路

14.2.3.3 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。

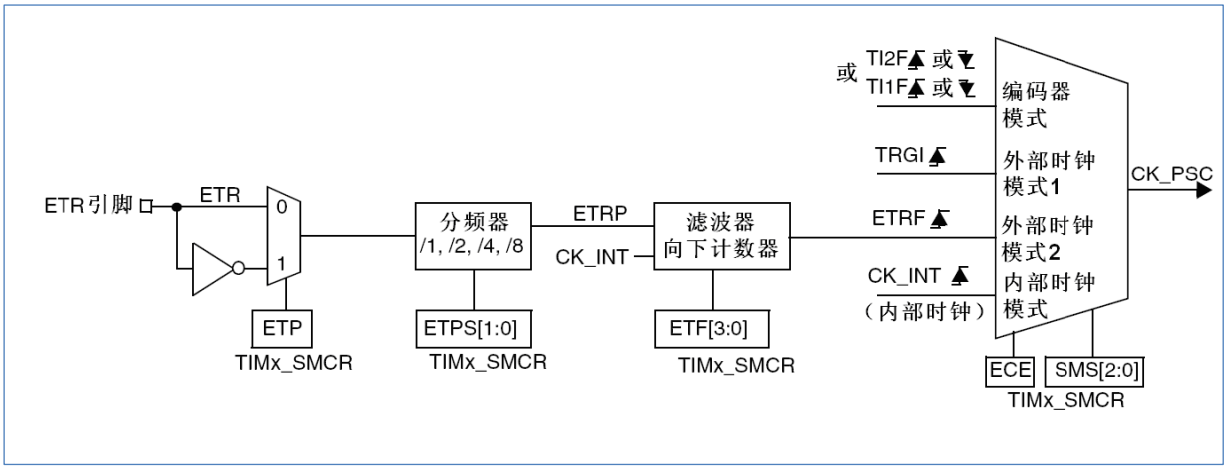


图 14-24 外部触发输入框图

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

1. 本例中不需要滤波器，置 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 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。

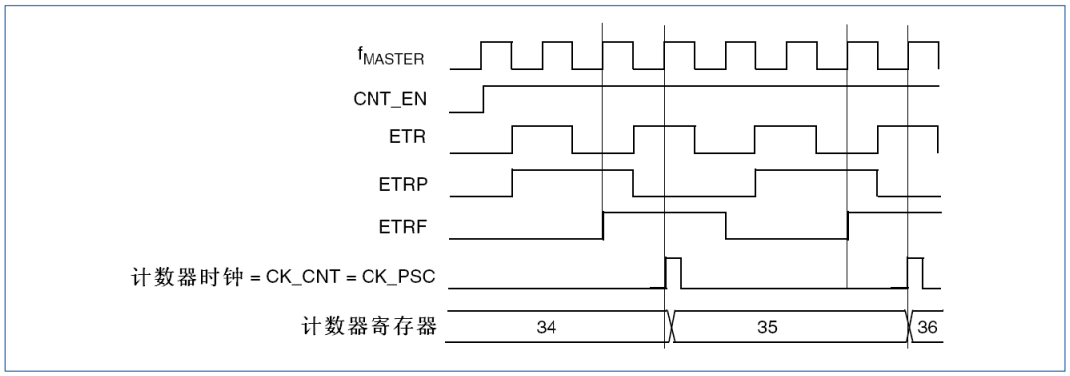


图 14-25 外部时钟模式 2 下的控制电路

14.2.4 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）。

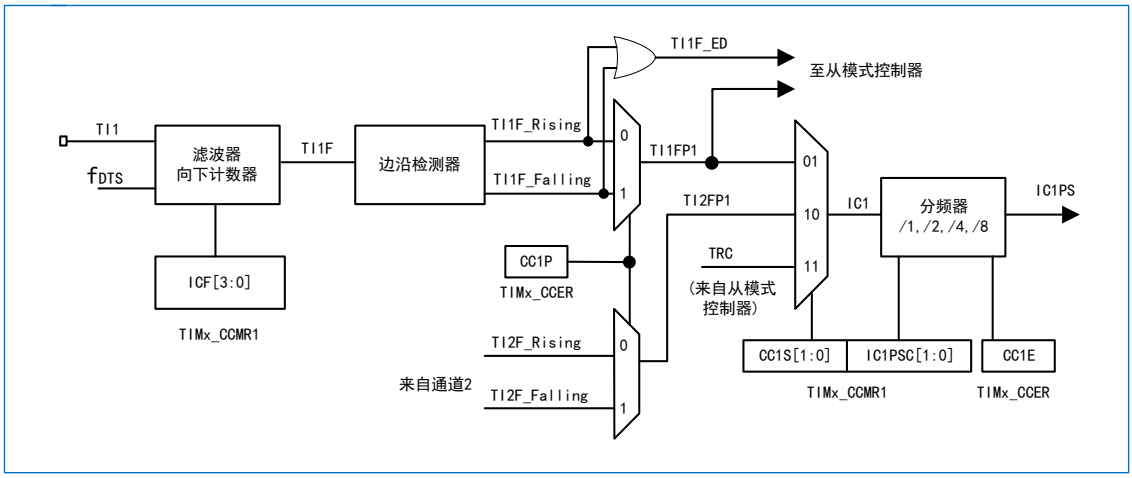


图 14-26 捕获/比较通道（如：通道 1 输入部分）

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。

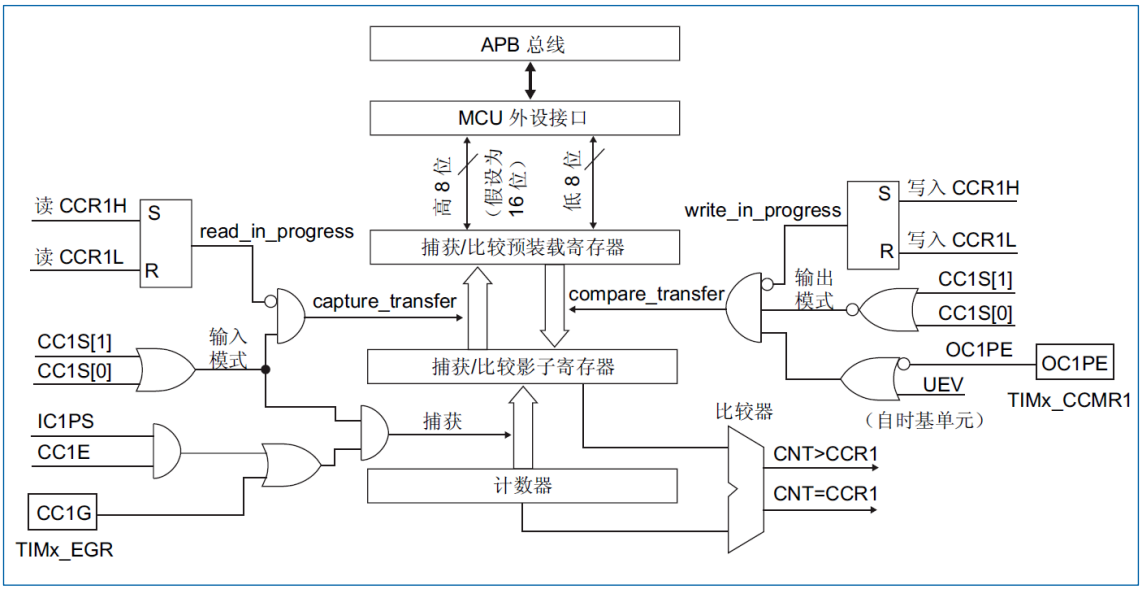


图 14-27 捕获/比较通道 1 的主电路

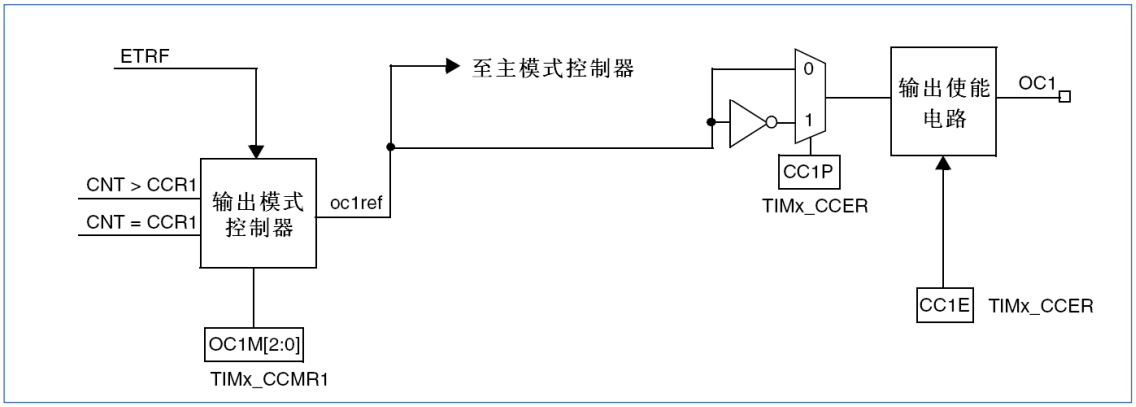


图 14-28 捕获/比较通道的输出部分（通道 1）

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。

14.2.5 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 fDTS 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

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 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置'1'。
- 如设置了 CC1IE 位，则会产生一个中断。
- 如设置了 CC1DE 位，则还会产生一个 DMA 请求。

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

注意：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。

14.2.6 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 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。

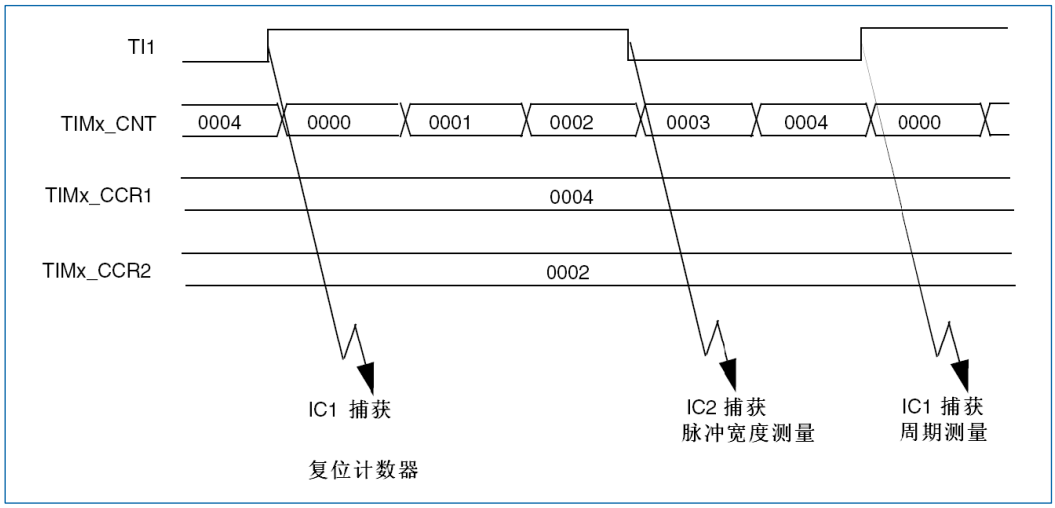


图 14-29 PWM 输入模式时序

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。

14.2.7 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。

14.2.8 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。

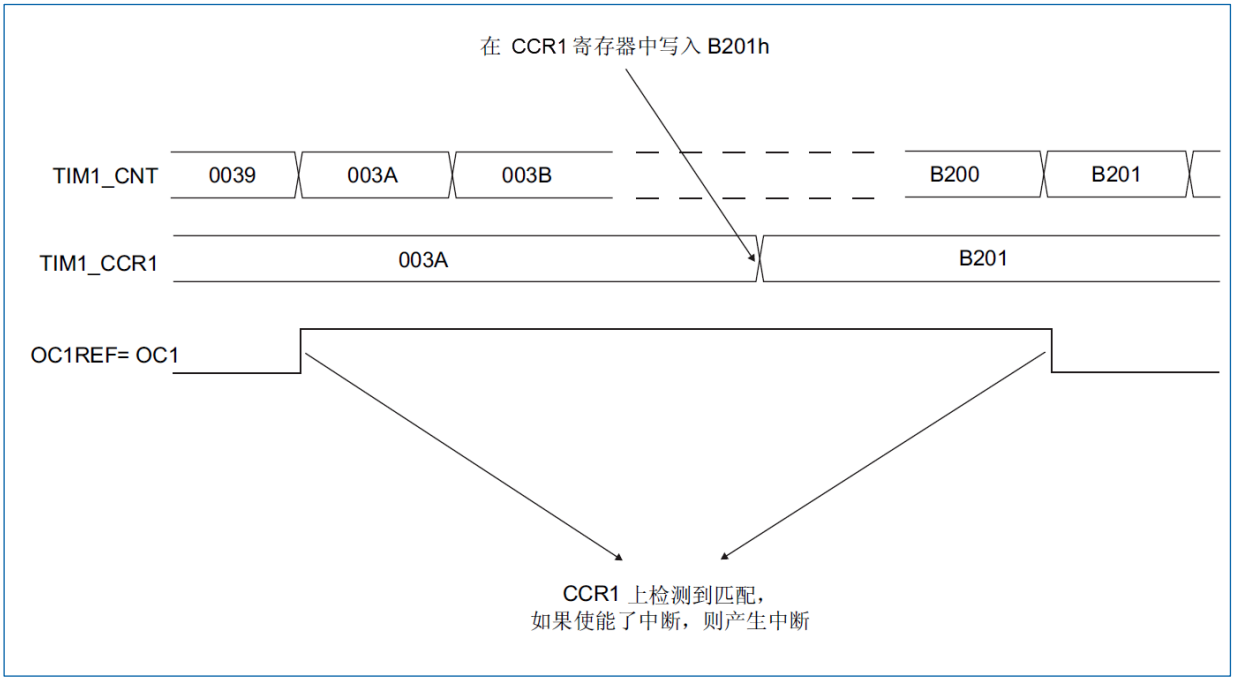


图 14-30 输出比较模式，翻转 0C1

14.2.9 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“14.3.9 TIM2/3 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。

14.2.9.1 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。参见“14.2.2 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。

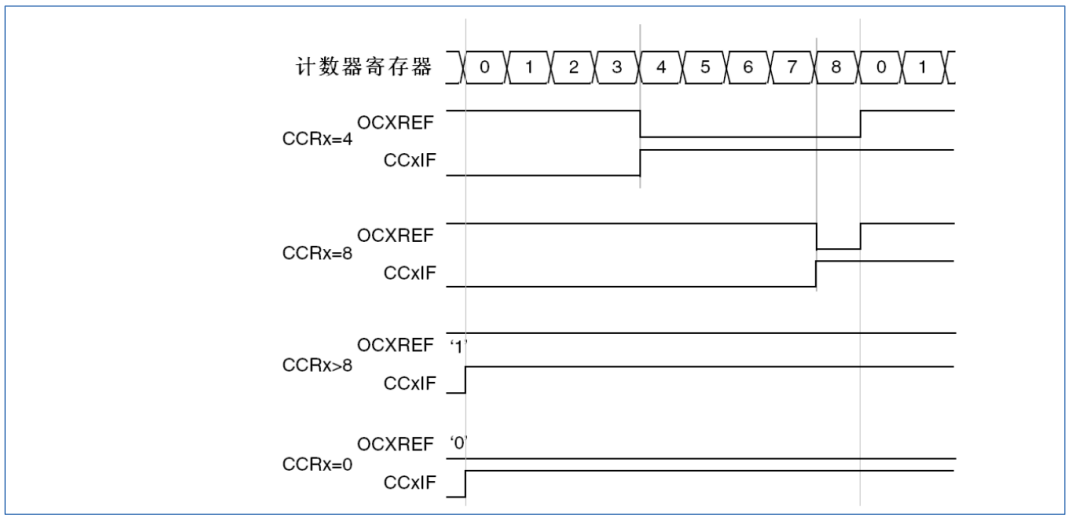


图 14-31 边沿对齐的 PWM 波形（ARR=8）

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。参见“14.2.2 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。

14.2.9.2 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“14.2.2.3 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。

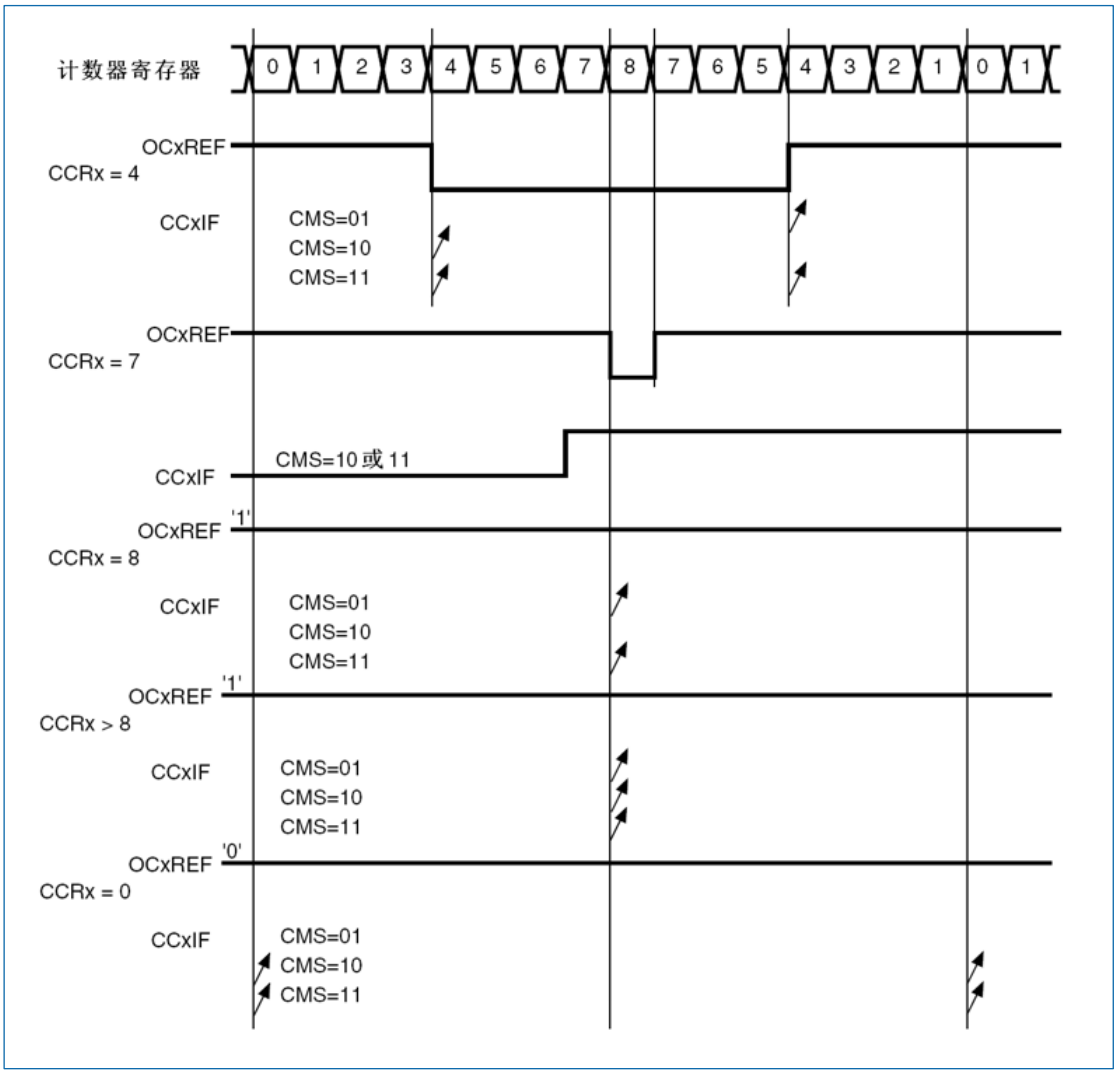


图 14-32 中央对齐的 PWM 波形（APR=8）

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。

14.2.10 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

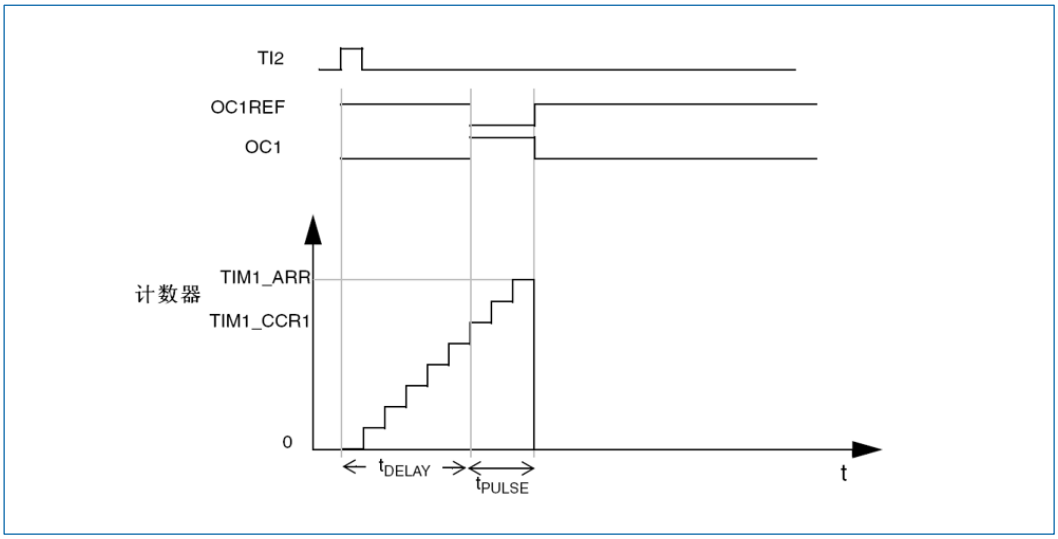


图 14-33 单脉冲模式的例子

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。

14.2.11 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为'1'，能够用 ETRF 输入端的高电

平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。

该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

例如，OCxREF 信号可以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，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。

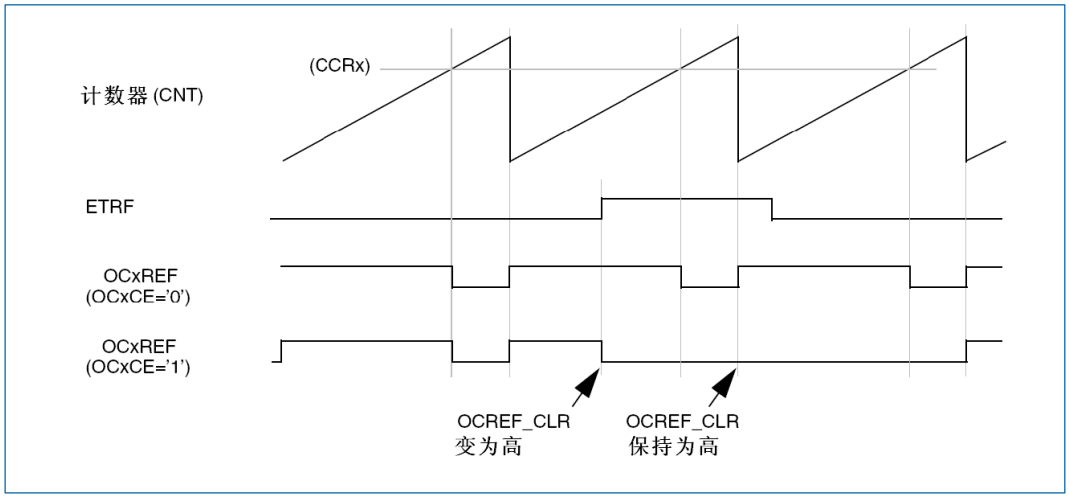


图 14-34 清除 TIMx 的 OCxREF

14.2.12 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 14-1，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 14-1 计数方向与编码器信号的关系

有效边沿	相对信号的电平（TI1FP1 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）

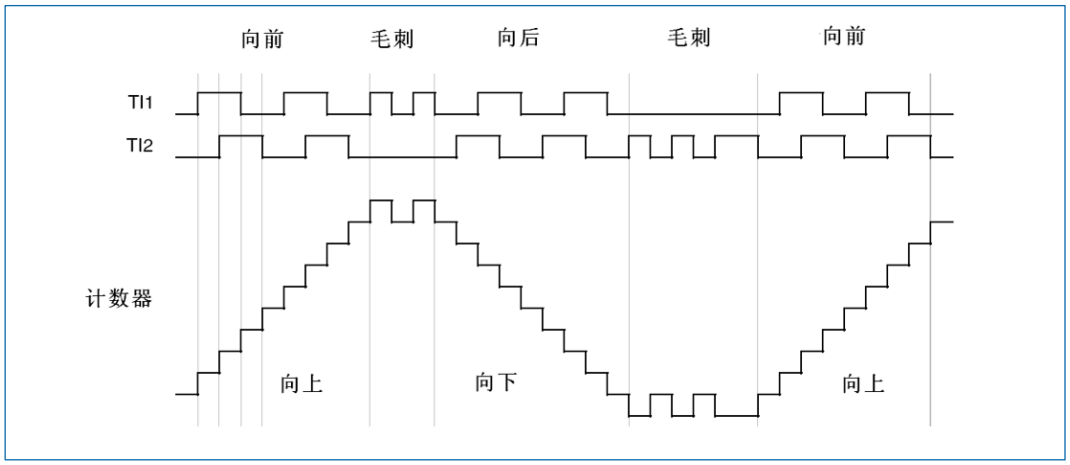


图 14-35 编码器模式下的计数器操作实例

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）。

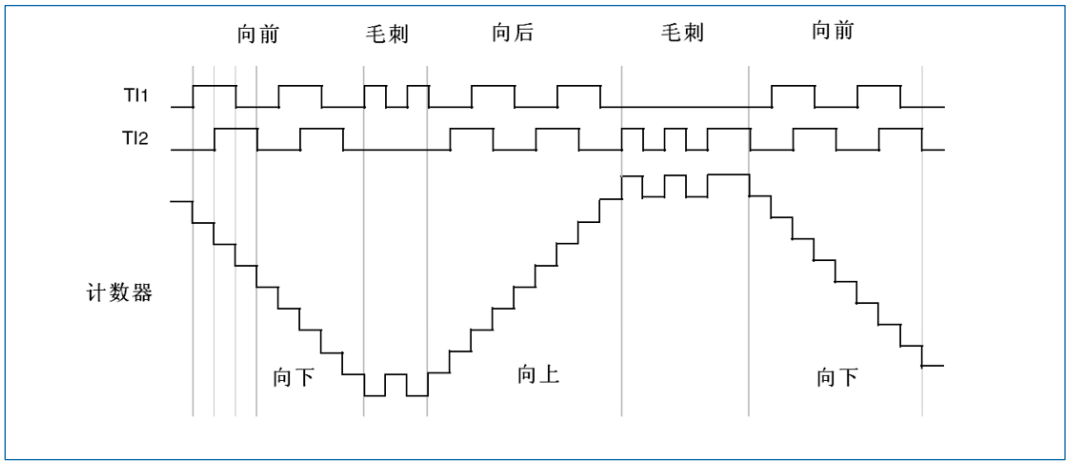


图 14-36 IC1FP1 反相的编码器接口模式实例

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。

14.2.13 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。“13.2.18 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。

14.2.14 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。

14.2.14.1 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。

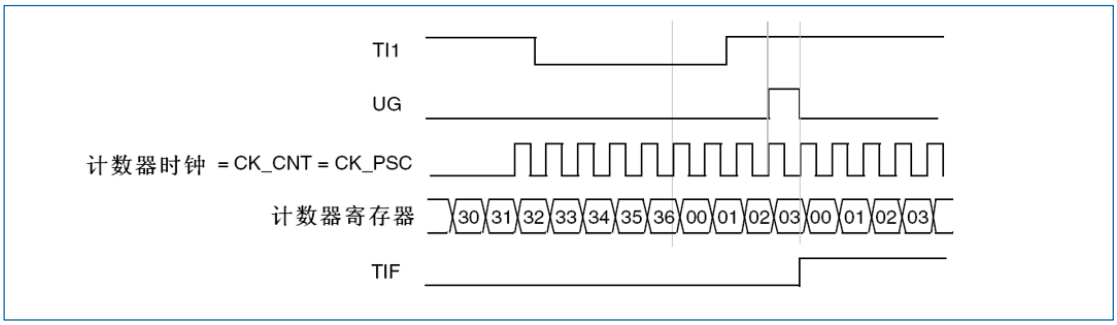


图 14-37 复位模式下的控制电路

14.2.14.2 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。

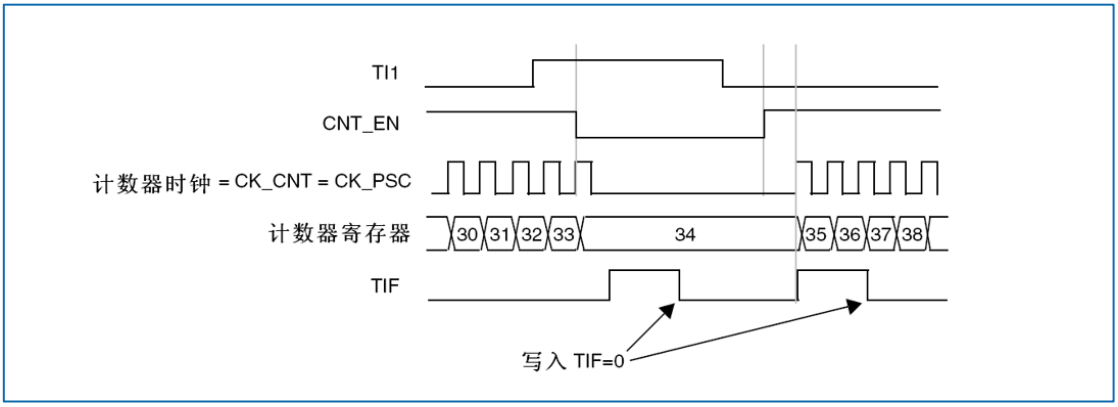


图 14-38 门控模式下的控制电路

14.2.14.3 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。

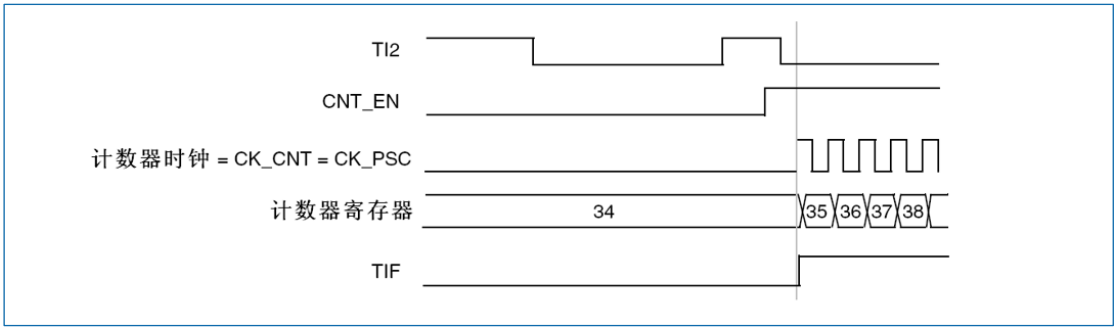


图 14-39 触发器模式下的控制电路

14.2.14.4 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。

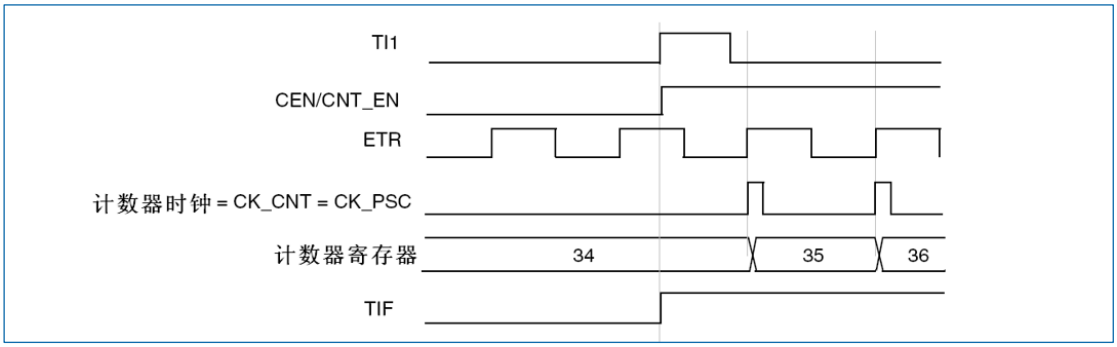


图 14-40 外部时钟模式 2＋触发模式下的控制电路

14.2.15 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。

14.2.15.1 使用一个定时器作为另一个定时器的预分频器

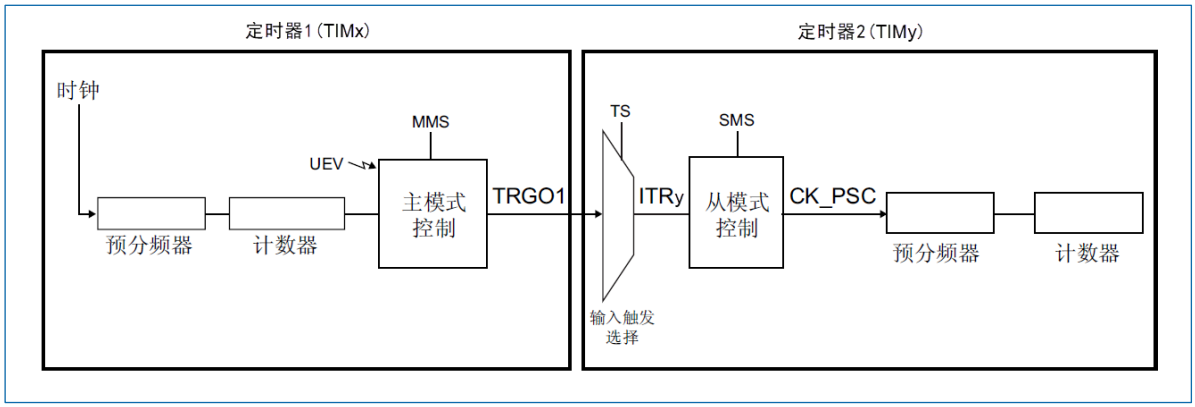


图 14-41 主/从定时器的例子

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。

14.2.15.2 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 14-41。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 2TIMy 计数器的使能信号。

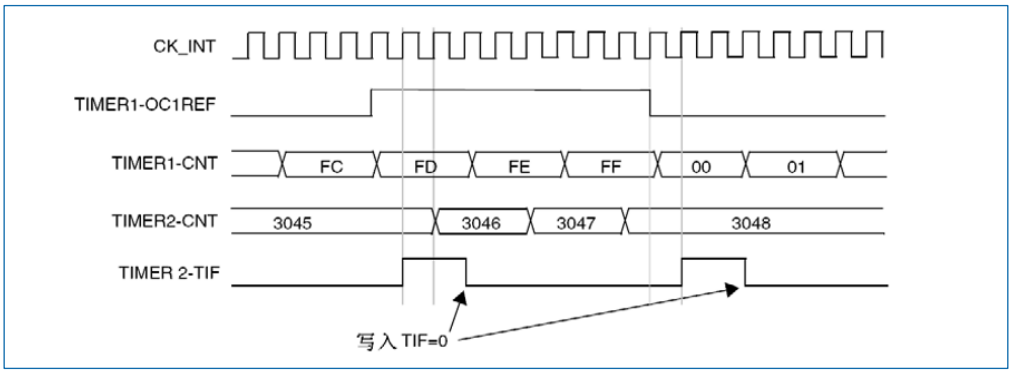


图 14-42 定时器 1 的 OC1REF 控制定时器 2

在上图的例子中，在定时器 2（TIMy）启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 1（TIMx）之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入需要的任意数值。写 TIMy_EGR 寄存器的 UG 位即可复位定时器。

在下一个例子中，需要同步定时器 1（TIMx）和定时器 2（TIMy）。TIMx 是主模式并从 0 开始，TIMy 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写'0'到 TIMx 的 TIMx_CR1 的 CEN 位将禁止 TIMx，TIMy 随即停止。

1. 配置 TIMx 为主模式，送出输出比较 1 参考信号（OC1REF）做为触发输出（TIMx_CR2 寄存器的 MMS=100）。
2. 配置 TIMx 的 OC1REF 波形（TIMx 的 TIMx_CCMR1 寄存器）。
3. 配置 TIMy 从 TIMx 获得输入触发（TIMy 的 TIMy_SMCR 寄存器的 TS=000）。
4. 配置 TIMy 为门控模式（TIMy_SMCR 寄存器的 SMS=101）。
5. 置 TIMx 的 TIMx_EGR 寄存器的 UG='1'，复位 TIMx。
6. 置 TIMy 的 TIMy_EGR 寄存器的 UG='1'，复位 TIMy。
7. 写'0xE7'至 TIMy 的计数器（TIMy_CNTL），初始化它为 0xE7。
8. 置 TIMy_CR1 寄存器的 CEN='1'以使能 TIMy。
9. 置 TIMx_CR1 寄存器的 CEN='1'以启动 TIMx。
10. 置 TIMx 的 TIMx _CR1 寄存器的 CEN='0'以停止 TIMx。

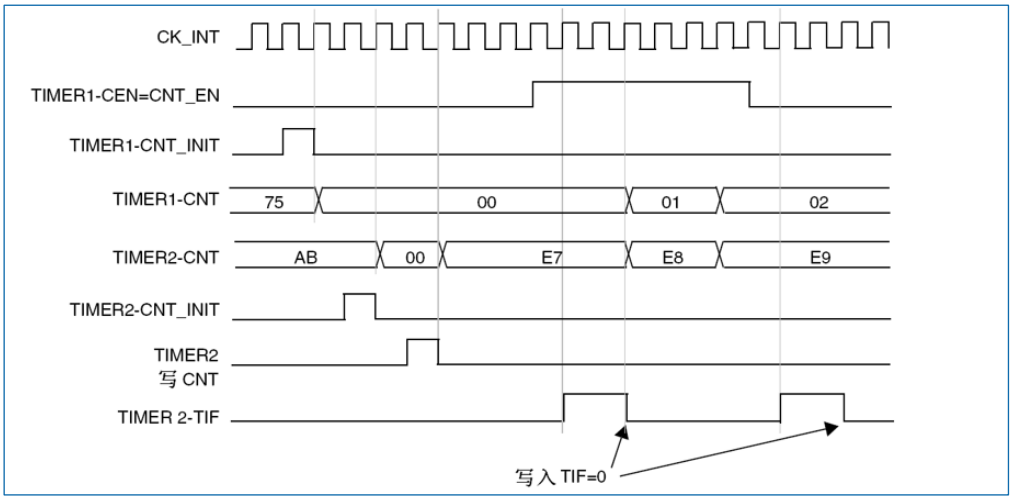


图 14-43 通过使能定时器 1 可以控制定时器 2

14.2.15.3 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），参见图 14-41。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'到定时器 2 的 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_ARR 寄存器）。
3. 配置 TIMy 从 TIMx 获得输入触发（TIMy_SMCR 寄存器的 TS 位）。
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。

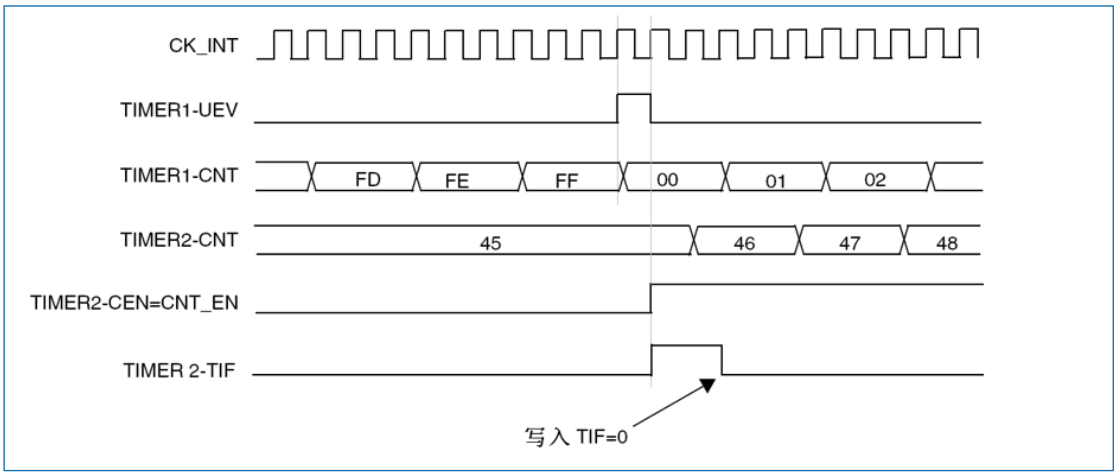


图 14-44 使用定时器 1 的更新触发定时器 2

在上一个例子中，可以在启动计数之前初始化两个计数器。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_SMCR 寄存器的 SMS=110）的动作。

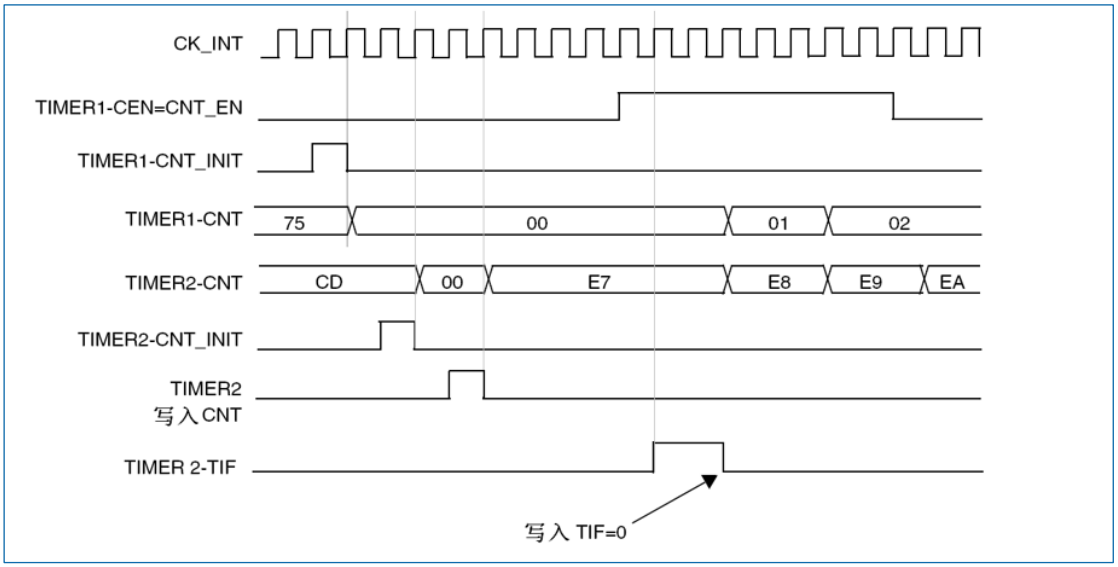


图 14-45 利用定时器 1 的使能触发定时器 2

14.2.15.4 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 输入上升时使能 TIMx，使能 TIMx 的同时使能定时器 2（TIMy），

参见图 14-41。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，送出它的使能做为触发输出（TIMx_CR2 寄存器的 MMS=’001’）。
2. 配置 TIMx 为从模式，从 TI1 获得输入触发（TIMx_SMCR 寄存器的 TS=’100’）。
3. 配置 TIMx 为触发模式（TIMx_SMCR 寄存器的 SMS=’110’）。
4. 配置 TIMx 为主/从模式， TIMx_SMCR 寄存器的 MSM=’1’。
5. 配置 TIMy 从 TIMx 获得输入触发（TIMx_SMCR 寄存器的 TS 位）。
6. 配置 TIMy 为触发模式（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。

注意：在这个例子中，在启动之前两个定时器都被初始化（设置相应的 UG 位），两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器（TIMx_CNT）在定时器间插入一个偏移。下图中能看到主/从模式下在 TIMx 的 CNT_EN 和 CK_PSC 之间有个延迟。

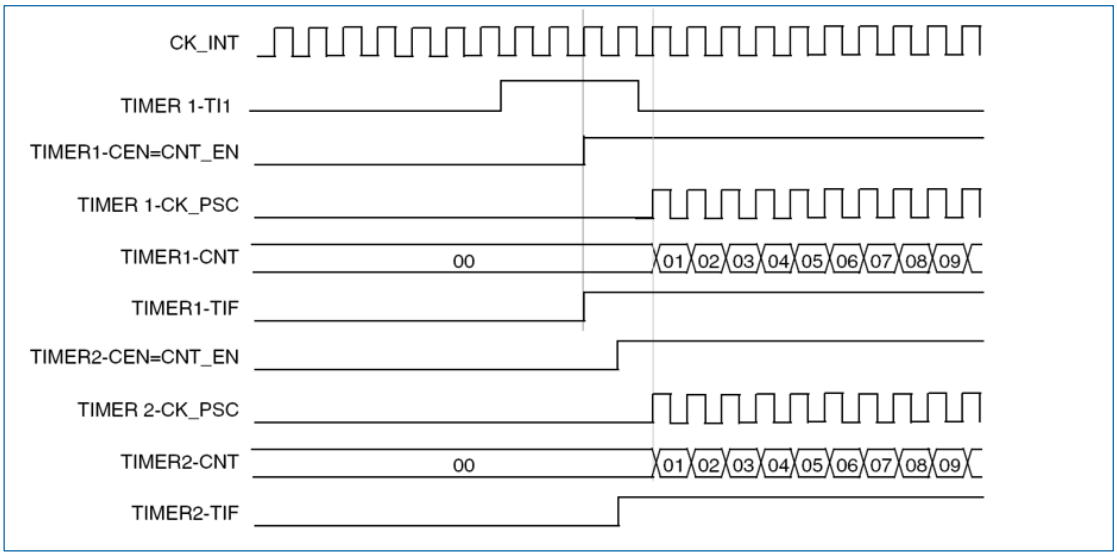


图 14-46 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2

14.2.16 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“26.8.2 对定时器、看门狗和 I2C 的调试支持”。

14.3 TIM2/3 寄存器

基地址：（TIM2, TIM3）=（0x4000 0000, 0x4000 0400）

空间大小：（TIM2, TIM3）=（0x400, 0x400）

14.3.1 TIM2/3 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（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 o 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

14.3.2 TIM2/3 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。

14.3.3 TIM2/3 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]			Res	SMS[2:0]		
rw	rw	rw		rw				rw	rw				rw		

位 15	ETP：外部触发极性（External trigger polarity） 该位选择是用 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）100：TI1 的边沿检测器（TI1F_ED）。 • 001：内部触发 1（ITR1）101：滤波后的定时器输入 1（TI1FP1）。 • 010：内部触发 2（ITR2）110：滤波后的定时器输入 2（TI2FP2）。 • 011：内部触发 3（ITR3）111：外部触发输入（ETRF）。
位 3	Res：保留 必须保持复位值。
位 2:0	SMS[2:0]：从模式选择（Slave mode selection） 当选择了外部信号，触发信号（TRGI）的有效边沿与选中的外部输入极性相关。 • 000：关闭从模式如果 CEN=1，则预分频器直接由内部时钟驱动。 • 001：编码器模式 1 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 • 010：编码器模式 2 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 • 011：编码器模式 3 根据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 14-2 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	-	TIM3	TIM14
TIM3	TIM1	TIM2	-	TIM14

14.3.4 TIM2/3 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	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

14.3.5 TIM2/3 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。

14.3.6 TIM2/3 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。

14.3.7 TIM2/3 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象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
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）。

14.3.8 TIM2/3 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 14.3.7 TIM2/3 捕捉/比较模式寄存器 1（TIMx_CCMR1）（x=2..3）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）。

14.3.9 TIM2/3 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：捕获/比较 4 互补输出极性（Capture/Compare 4 complementary output polarity） 参见 CC1NP 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：捕获/比较 3 互补输出极性（Capture/Compare 3 complementary output polarity）

	参见 CC1NP 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：捕获/比较 2 互补输出极性（Capture/Compare 2 complementary output polarity） 参见 CC1NP 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：捕获/比较 1 互补输出极性（Capture/Compare 1 complementary output polarity） • CC1 通道配置为输出时，CC1NP 必须保持零，即 CC1NP=0 • CC1 通道配置为输入时，CC1NP 与 CC1P 共同控制 TI1FP1/TI2FP1 的极性。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： o 0：OC1 高电平有效。 o 1：OC1 低电平有效。 • CC1 通道配置为输入： CC1NP 和 CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 o 00：非反相/上升沿 电路作用于 TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 非反相。 o 01：反相/下降沿 电路作用于 TIxFP1 的下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP1 反相。 o 10：保留不用 o 11：非反相/上升或下降沿 电路作用于 TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕捉或

	触发操作）、TIxFP1 非反相（在门控模式）。在编码模式下不能使用此配置。
位 0	CC1E：捕捉/比较 1 输出使能（Capture/Compare 3 output enable） • CC1 通道配置为输出： ○ 0：关闭 OC1 禁止输出，因此 OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 ○ 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（TIMx_CCR1）。 ○ 0：捕捉禁止 ○ 1：捕捉允许

表 14-3 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

14.3.10 TIM2/3 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]（仅 TIM2）															
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（只支持 TIM2）（Counter value high bits of TIM2）
位 15:0	CNT_L[15:0]：计数器值（Counter value）

14.3.11 TIM2/3 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 = fCK_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'。

14.3.12 TIM2/3 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]（仅 TIM2）															
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（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（Auto-reload value low bits） ARR_L 包含了将要装载入实际的自动重装载寄存器的值。

14.3.13 TIM2/3 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]（仅 TIM2）															
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 (Only 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。

14.3.14 TIM2/3 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]（仅 TIM2）															
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 (Only 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。

14.3.15 TIM2/3 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]（仅 TIM2）															
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 (Only 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。

14.3.16 TIM2/3 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]（仅 TIM2）															
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 (Only 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。

14.3.17 TIM2/3 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。

14.3.18 TIM2/3 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，参见“13.3.20 TIM1 全部传输时 DMA 地址寄存器（TIM1_DMAR）”。

15 通用定时器（TIM14）

通用定时器 TIM14 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。TIM14 适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较和 PWM）。

使用定时器预分频器和 RCC 时钟控制预分频器，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。

通用定时器 TIM14 是完全独立的，不共享任何资源。

15.1 TIM14 主要功能

- 16 位自动装载计数器
- 16 位向上、向下、向上/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。
- 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）
  - 输入捕获
  - 输出比较

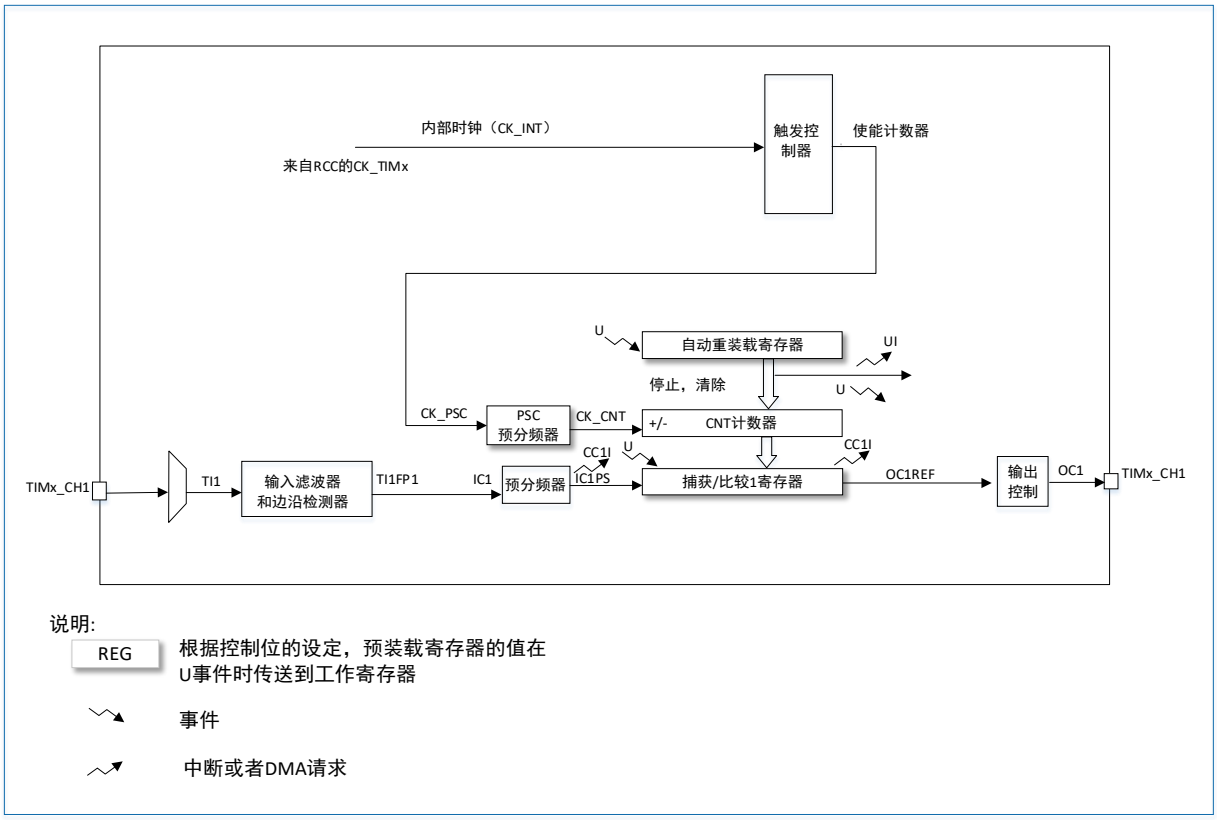


图 15-1 通用定时器框图（TIM14）

15.2 TIM14 功能描述

15.2.1 时基单元

可编程通用定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数。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，在计数器运行时仍可以读写。

时基单元包含：

- 计数器寄存器（TIM14_CNT）
- 预分频器寄存器（TIM14_PSC）
- 自动装载寄存器（TIM14_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 TIM14_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容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并当 TIM14_CR1 寄存器中的 UDIS 位等于'0'时，产生更新事件。更新事件也可以由软件产生。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 TIM14_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效。

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

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TIM14_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器参数在下一次更新事件到来时被采用。

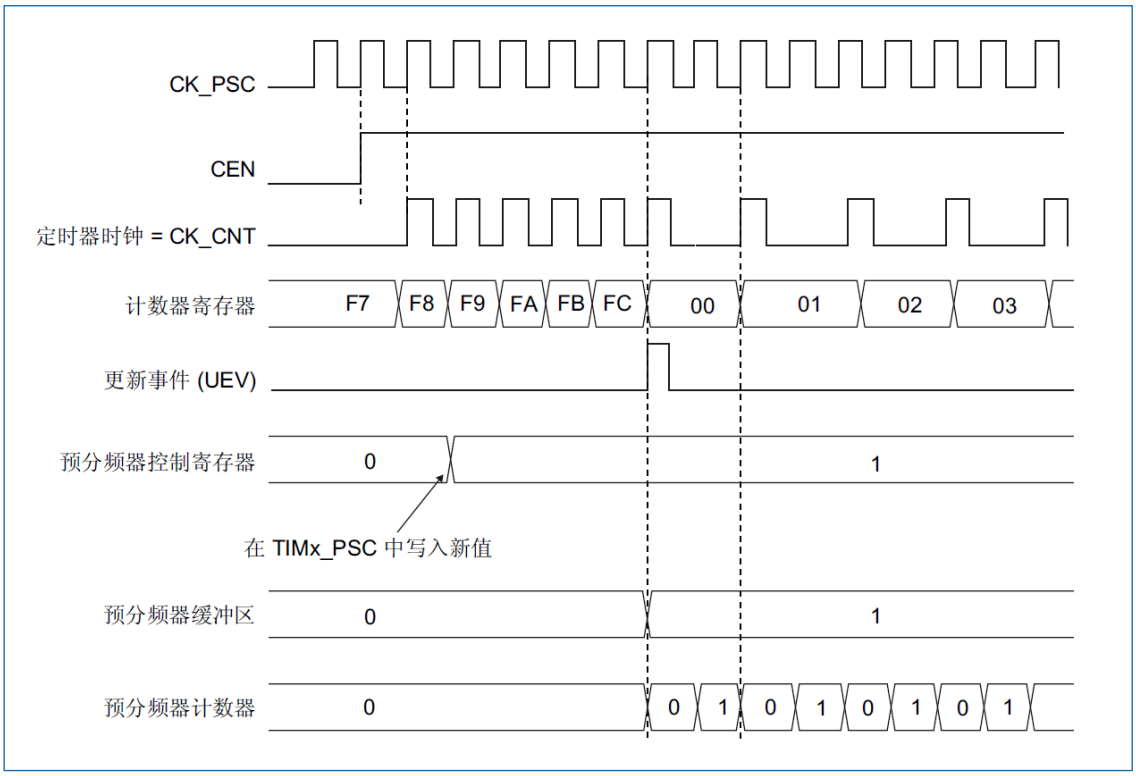


图 15-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

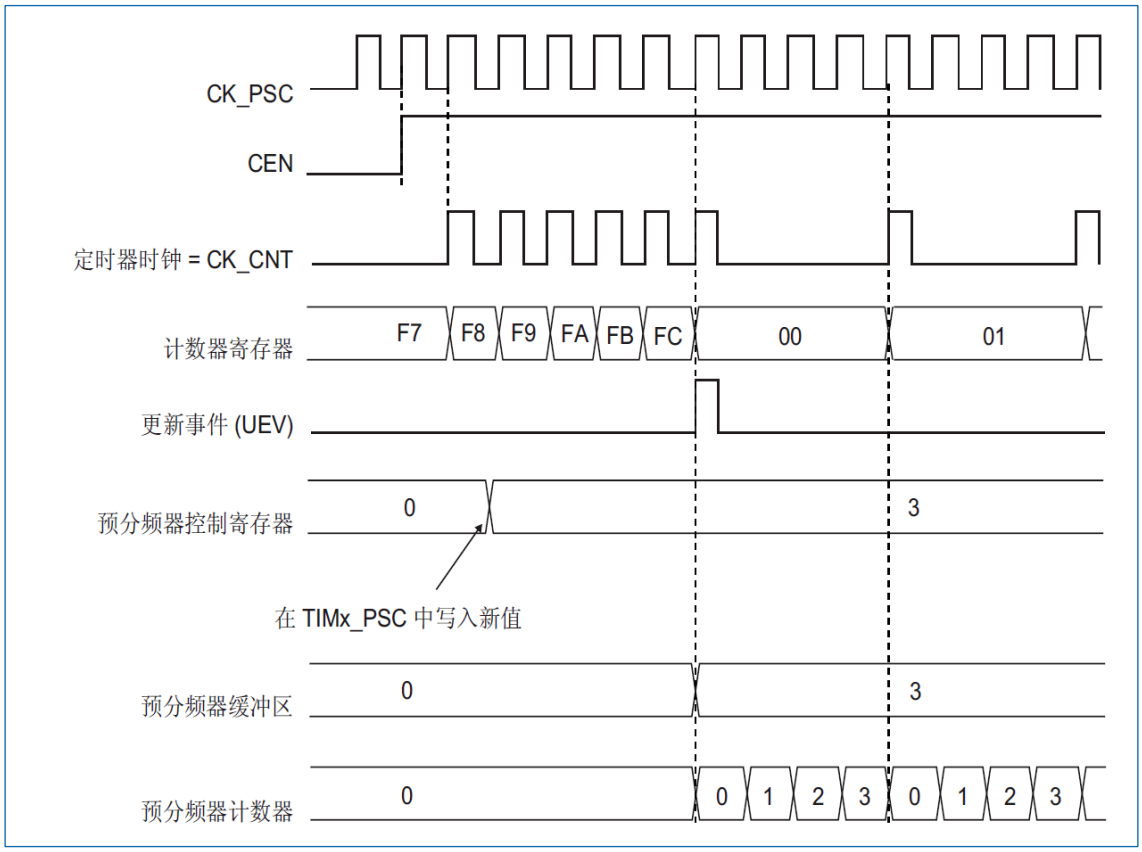


图 15-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

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（TIM14_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。

在 TIM14_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位可以产生一个更新事件。

设置 TIM14_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清零（但预分频器的数值不变）。此外，如果设置了 TIM14_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 URS 位）设置更新标志位（TIM14_SR 寄存器中的 UIF 位）。

- 自动装载影子寄存器被重新置入预装载寄存器的值（TIM14_ARR）。
- 预分频器的缓冲区被置入预装载寄存器的值（TIM14_PSC 寄存器的内容）。

下图给出一些例子，当 TIM14_ARR=0x36 时计数器在不同时钟频率下的动作。

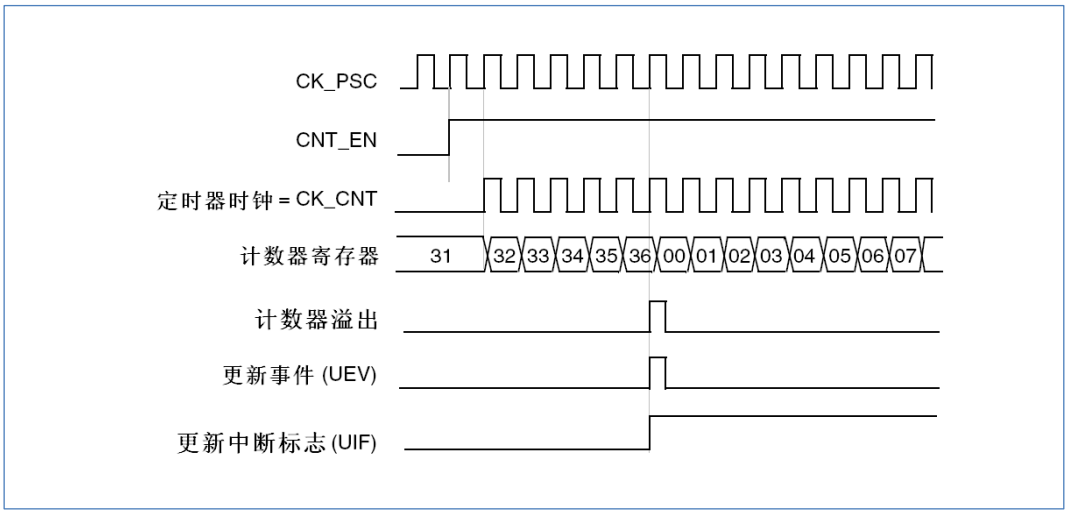


图 15-4 计数器时序图，内部时钟分频因子为 1

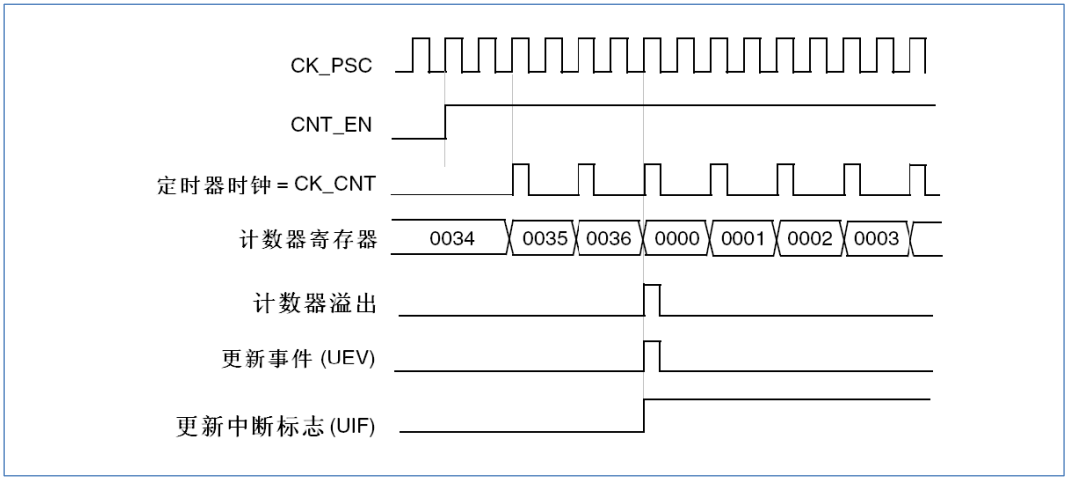


图 15-5 计数器时序图，内部时钟分频因子为 2

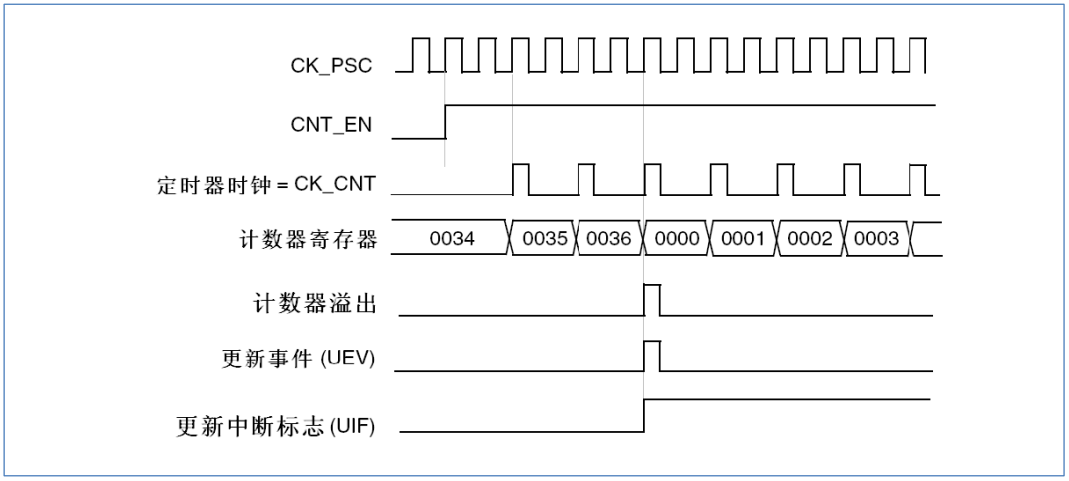


图 15-6 计数器时序图，内部时钟分频因子为 4

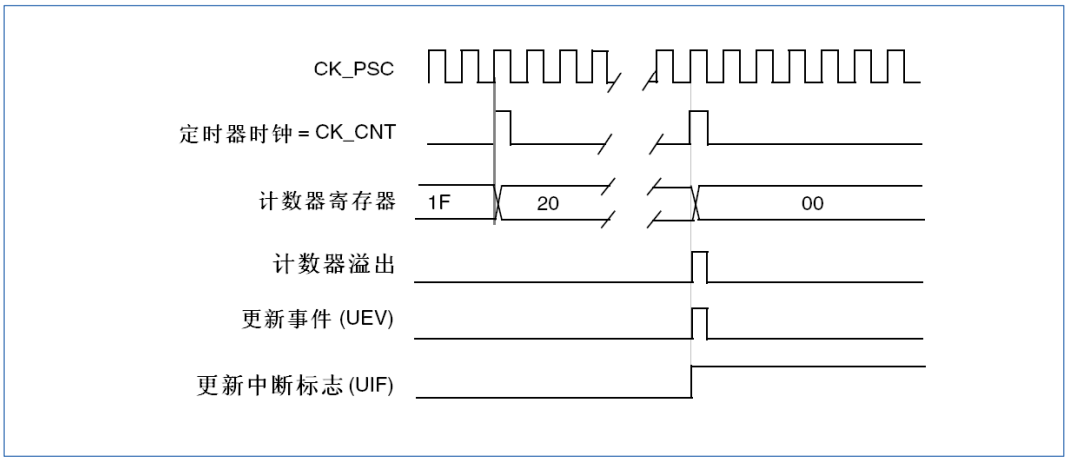


图 15-7 计数器时序图，内部时钟分频因子为 N

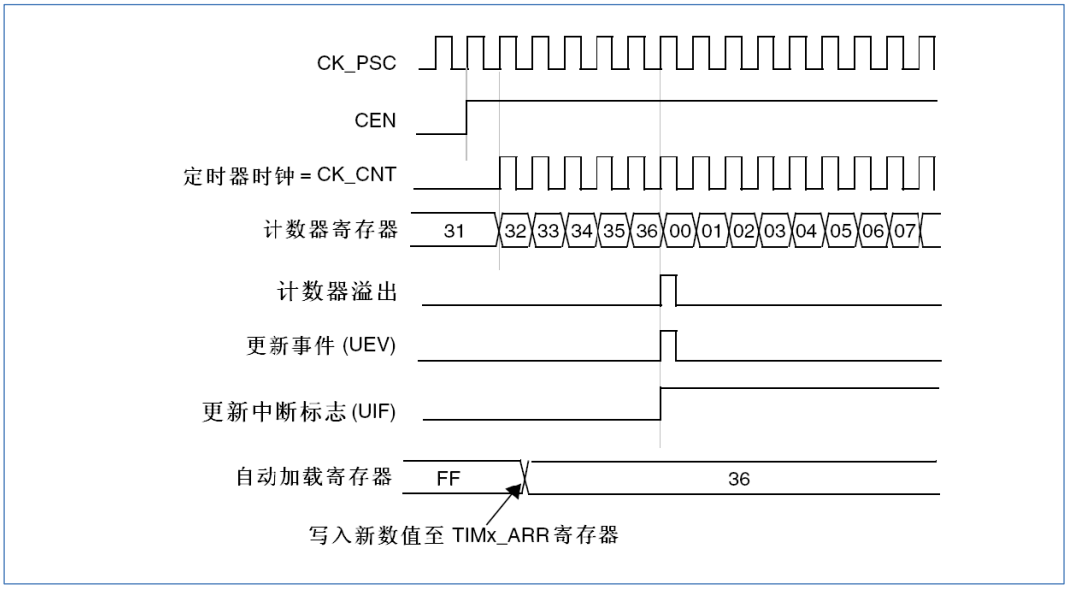


图 15-8 计数器时序图，当 ARPE=0 时的更新事件（TIM14_ARR 没有预装入）

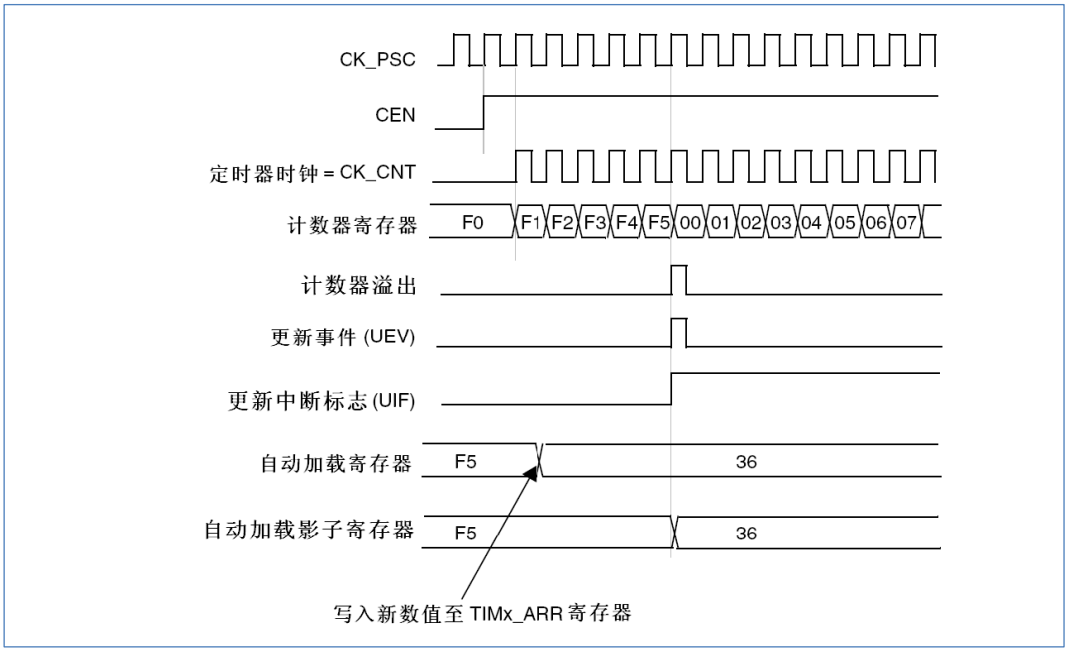


图 15-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM14_ARR）

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（TIM14_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。

在 TIM14_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，可以产生一个更新事件。

设置 TIM14_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 TIM14_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM14_SR 寄存器中的 UIF 位）也被设置。

- 预分频器的缓存器被加载为预装载的值（TIM14_PSC 寄存器的值）。
- 当前的自动加载寄存器被更新为预装载值（TIM14_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 TIM14_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

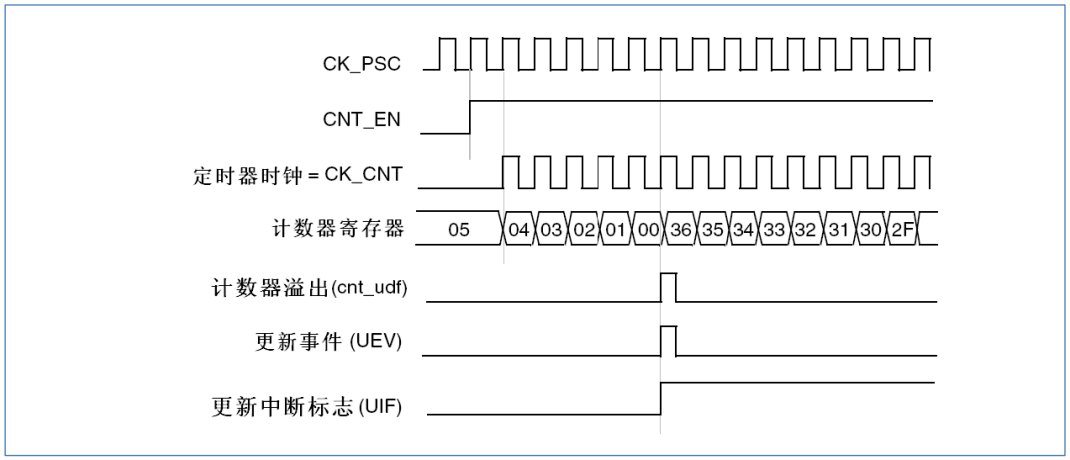


图 15-10 计数器时序图，内部时钟分频因子为 1

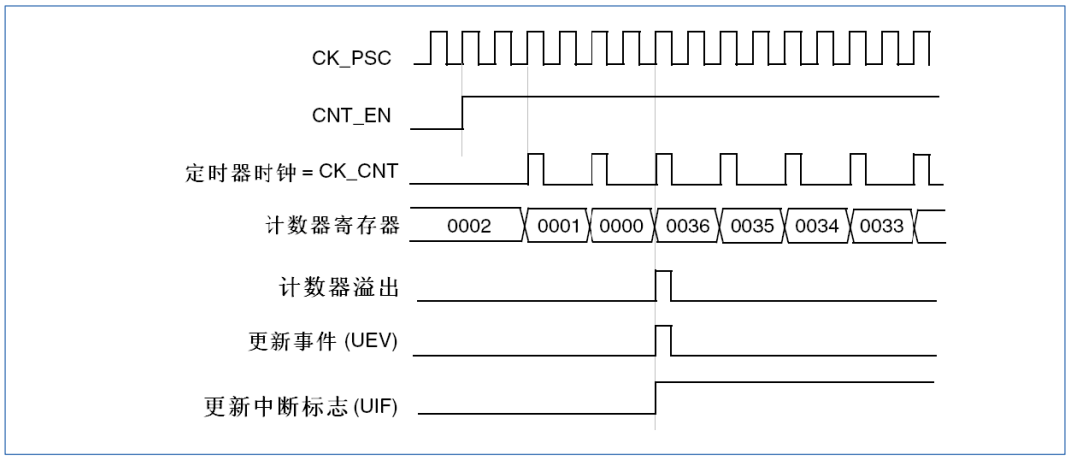


图 15-11 计数器时序图，内部时钟分频因子为 2

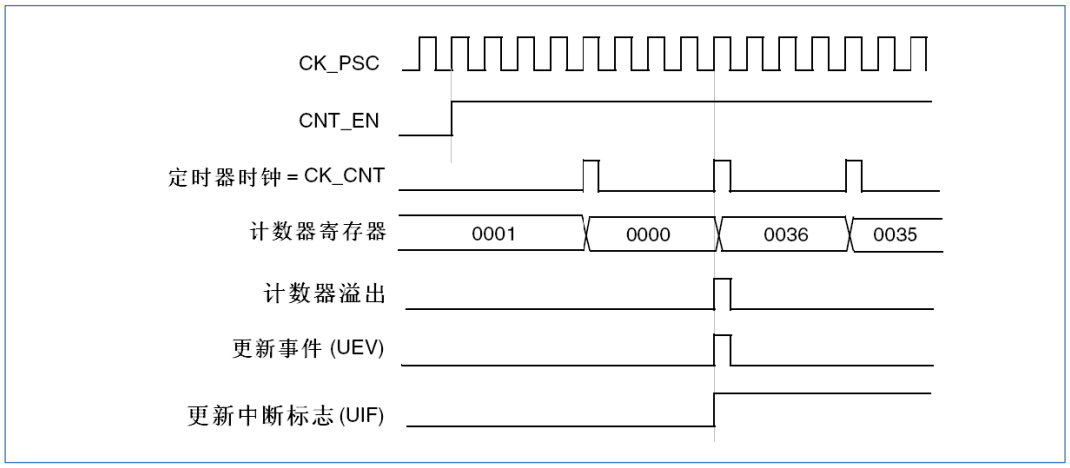


图 15-12 计数器时序图，内部时钟分频因子为 4

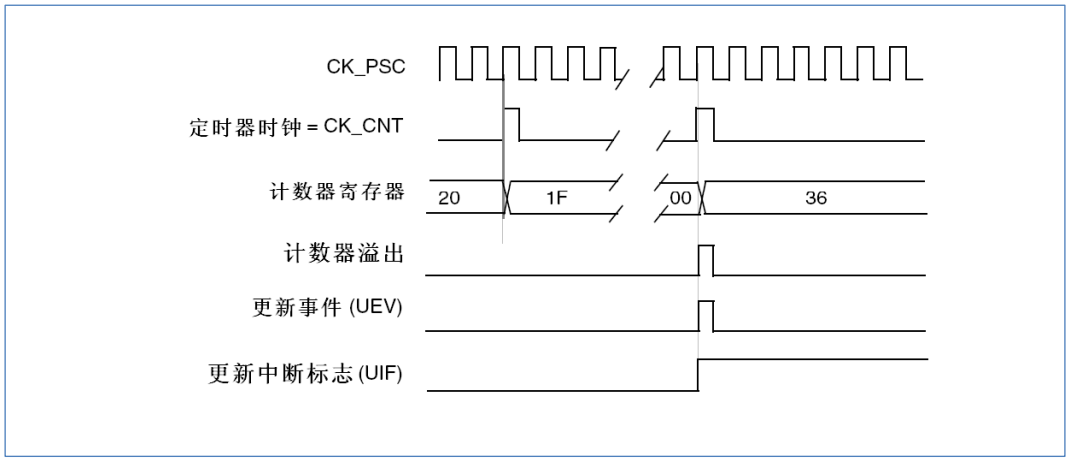


图 15-13 计数器时序图，内部时钟分频因子为 N

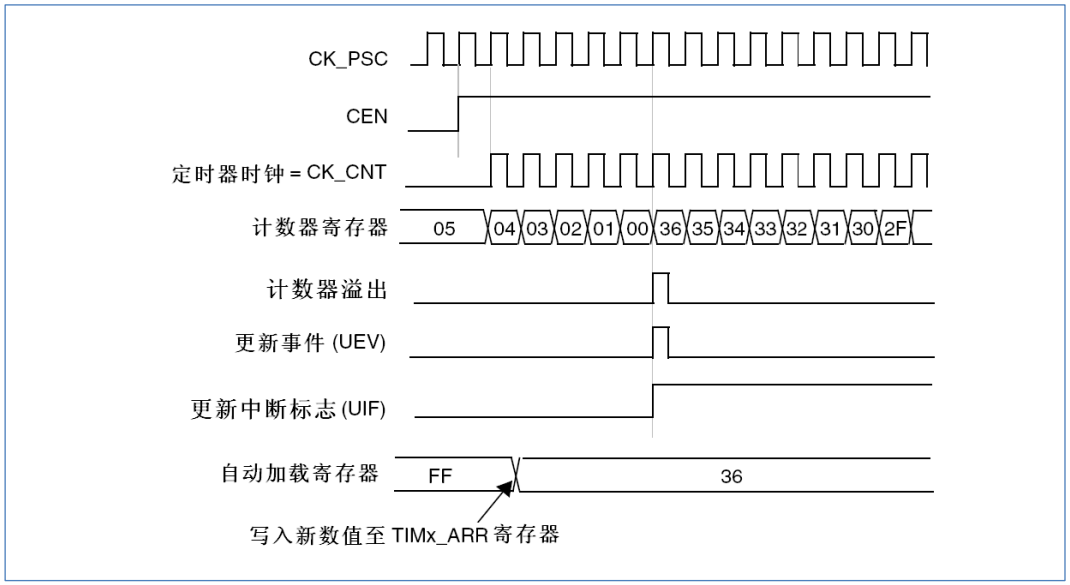


图 15-14 计数器时序图，当没有使用重复计数器时的更新事件

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（TIM14_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 TIM14_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 TIM14_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 TIM14_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 TIM14_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM14_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（TIM14_PSC 寄存器）的值。
- 当前的自动加载寄存器被更新为预装载值（TIM14_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：

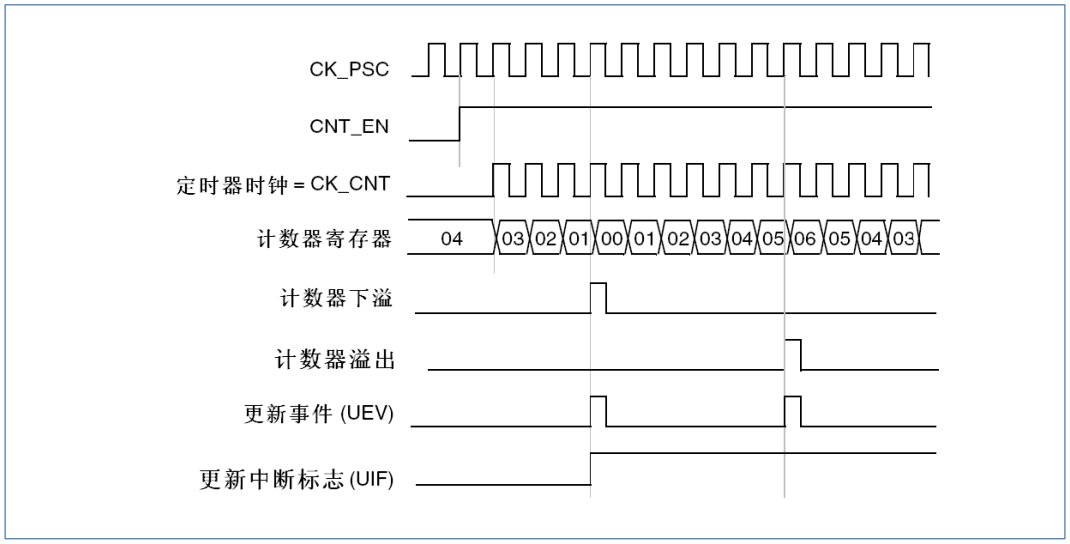


图 15-15 计数器时序图，内部时钟分频因子为 1，TIM14_ARR=0x6（这里使用了中央对齐模式 1）

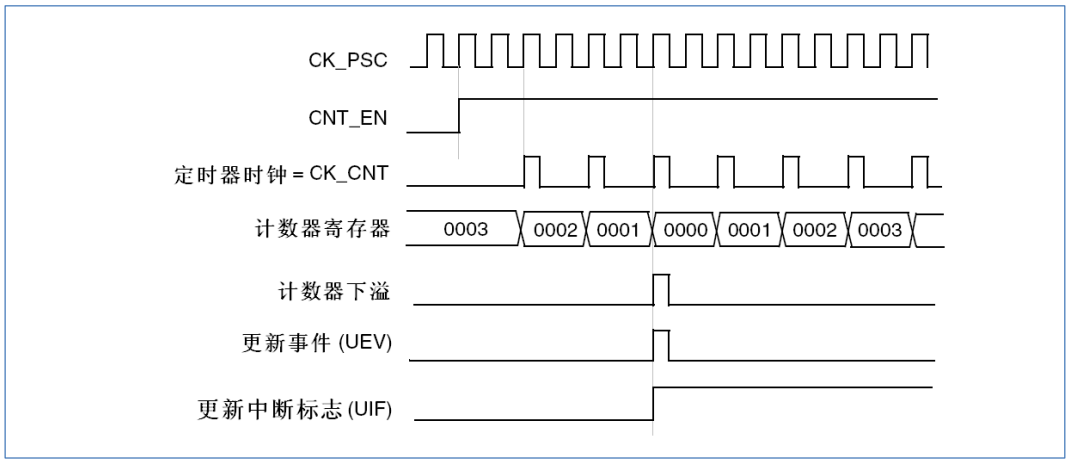


图 15-16 计数器时序图，内部时钟分频因子为 2（中央对齐模式 1）

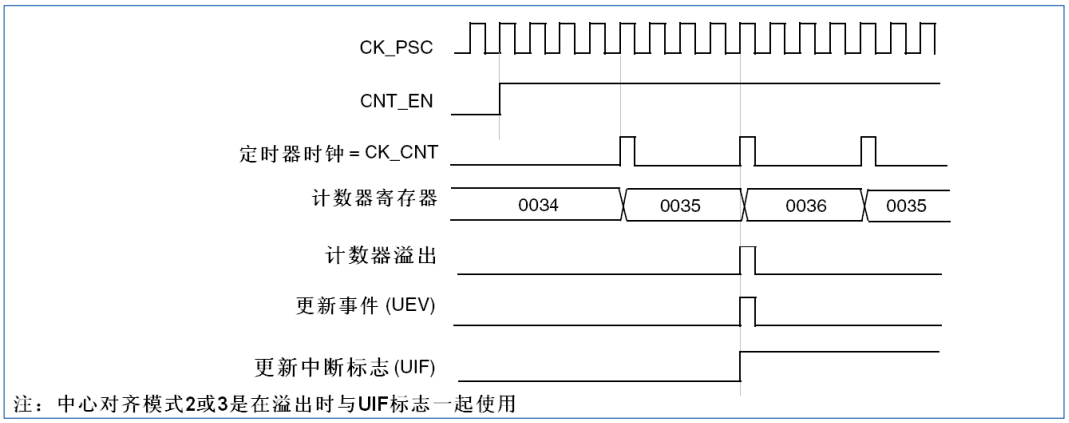


图 15-17 计数器时序图，内部时钟分频因子为 4，TIM14_ARR=0x36（中央对齐模式 2 或 3）

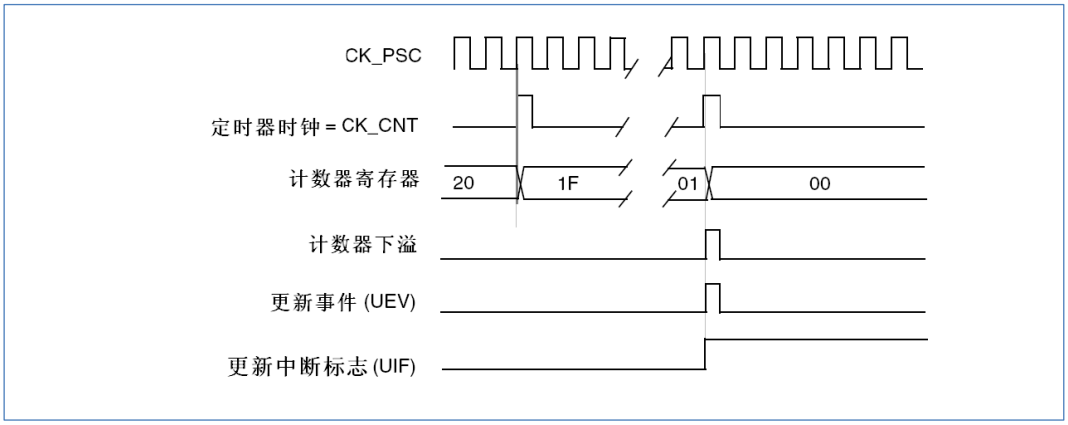


图 15-18 计数器时序图，内部时钟分频因子为 N

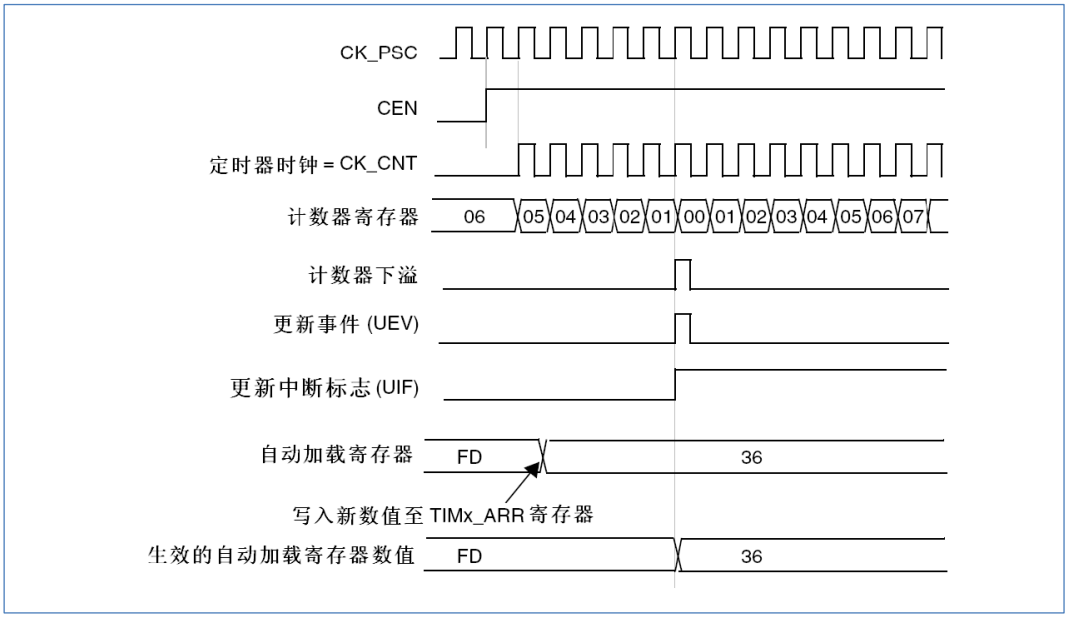


图 15-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

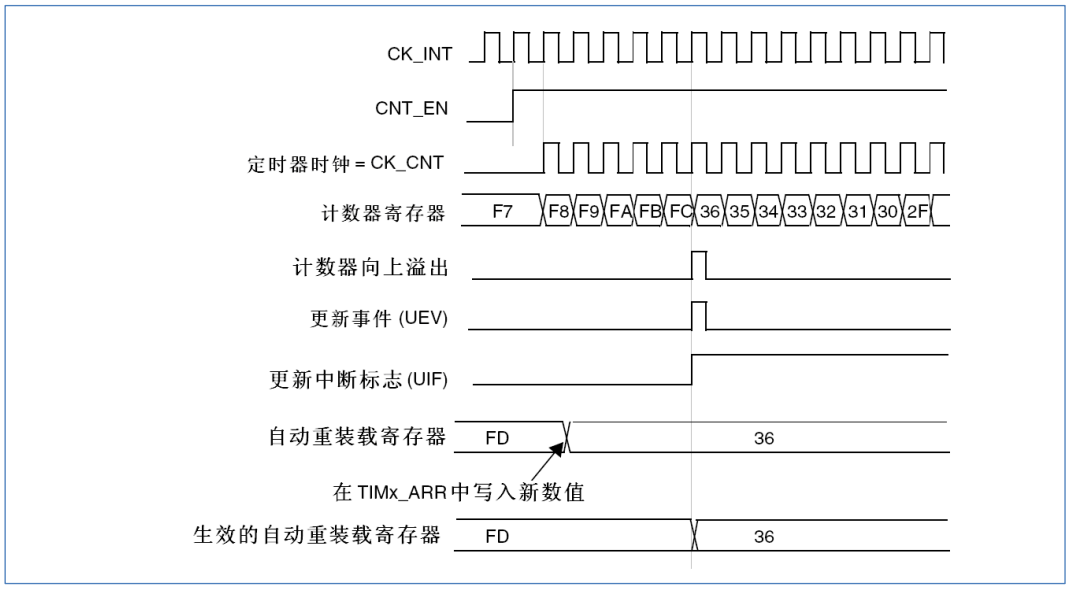


图 15-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

15.2.3 时钟选择

计数器时钟由内部时钟（CK_INT）提供。

CEN（TIM14_CR1 寄存器）和 UG 位（TIM14_EGR 寄存器）是实际的控制位，并且只能被软件修改（除了 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。

图 15-21 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。

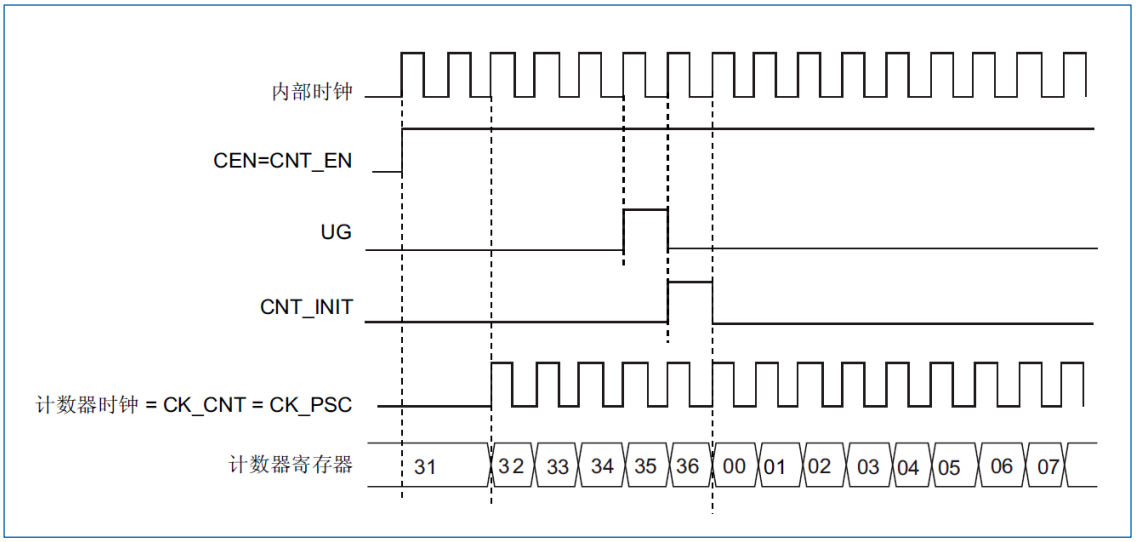


图 15-21 一般模式下的控制电路，内部时钟分频因子为 1

15.2.4 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）。

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）。

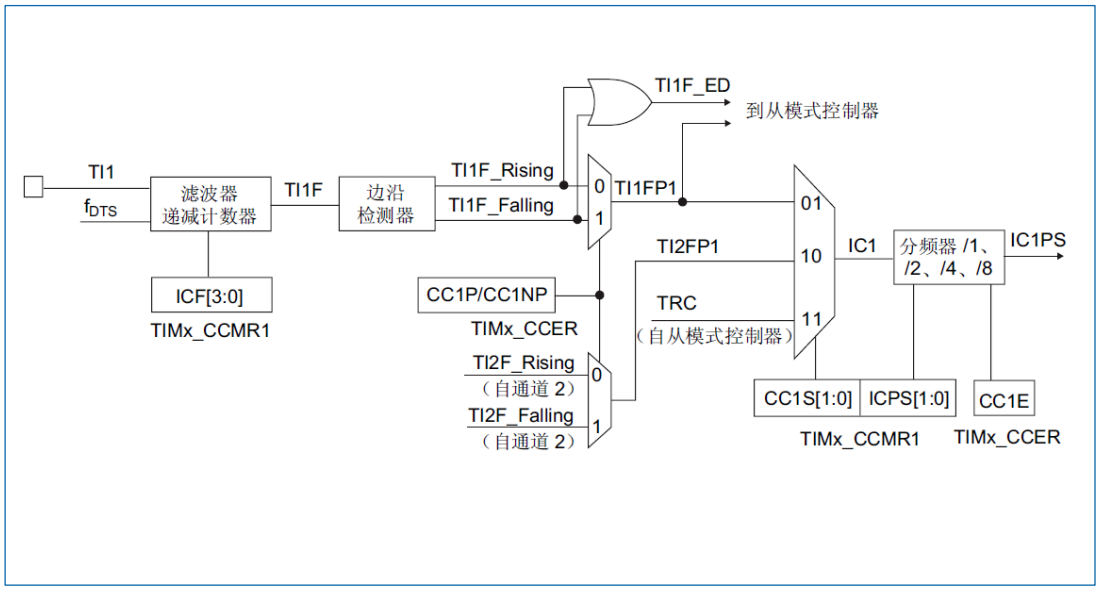


图 15-22 捕获/比较通道（如：通道 1 输入部分）

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。

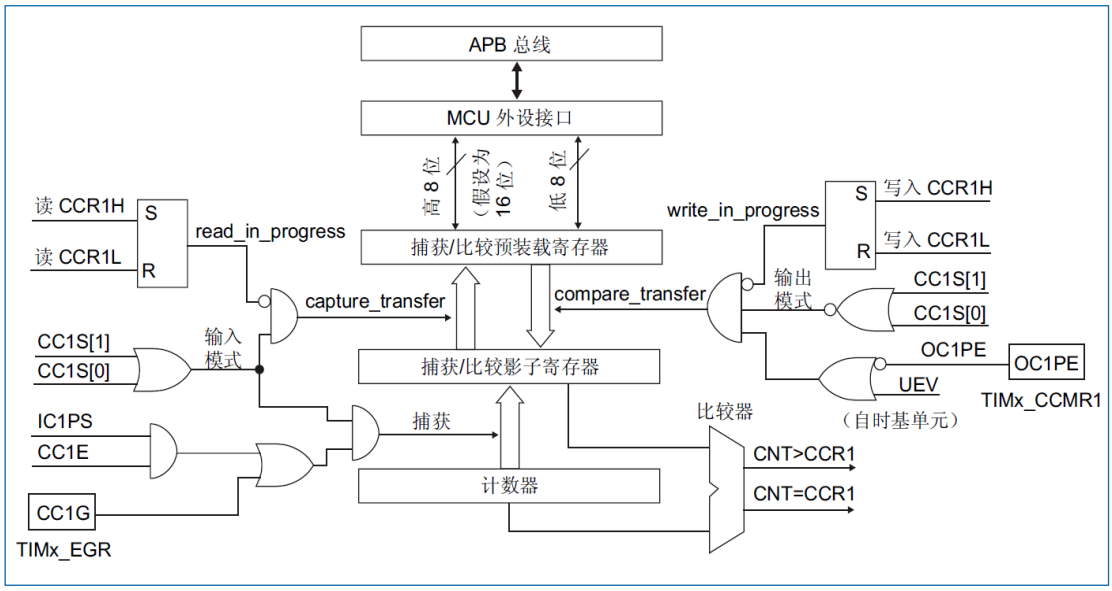


图 15-23 捕获/比较通道 1 的主电路

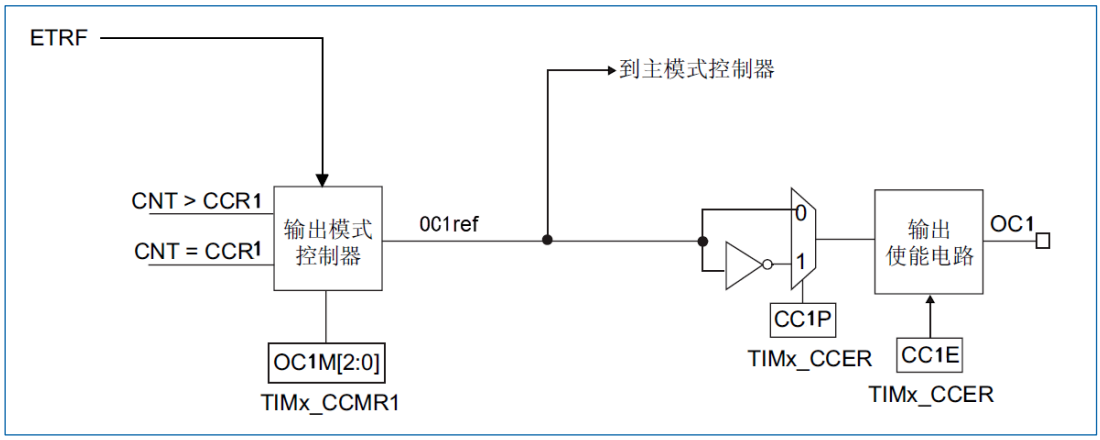


图 15-24 捕获/比较通道的输出部分（通道 1）

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。

15.2.5 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（TIM14_CCRx）中。当捕获事件发生时，相应的 CCxIF 标志（TIM14_SR 寄存器）被置'1'，如果使能了中断或者 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（TIM14_SR 寄存器）被置'1'。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIM14_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 TIM14_CCR1 寄存器中，步骤如下：

1. 选择有效输入端：TIM14_CCR1 必须连接到 TI1 输入，所以写入 TIM14_CCR1 寄存器中的 CC1S=01。只要 CC1S 不为'00'，通道则被配置为输入并且 TM1_CCR1 寄存器变为只读。
2. 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIM14_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽大于 5 个时钟周期。因此我们可以（以 fDTS 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIM14_CCMR1 寄存器中写入 IC1F=0011。
3. 选择 TI1 通道的有效转换边沿，在 TIM14_CCER 寄存器中写入 CC1P=0 和 CC1NP=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 TIM14_CCMR1 寄存器的 IC1PSC=00）。
5. 设置 TIM14_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 TIM14_DIER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIM14_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置'1'。
- 如设置了 CC1IE 位，则会产生一个中断。

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

注意：设置 TIM14_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断请求。

15.2.6 强制输出模式

在输出模式（TIM14_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 TIM14_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 TIM14_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIM14_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断请求。这将会在下面的输出比较模式一节中介绍。

15.2.7 输出比较模式

此功能是用来控制一个输出波形，或者指示一段给定的间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能的配置步骤如下：

1. 将输出比较模式（TIM14_CCMRx 寄存器中的 OCxM 位）和输出极性（TIM14_CCER 寄存器中的 CCxP 位）定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平（OCxM=000）、被设置成有效电平（OCxM=001）、被设置成无效电平（OCxM=010）或进行翻转（OCxM=011）。
2. 设置中断状态寄存器中的标志位（TIM14_SR 寄存器中的 CCxIF 位）。
3. 若设置了相应的中断屏蔽（TIM14_DIER 寄存器中的 CCxIE 位），则产生一个中断。

TIM14_CCMRx 中的 OCxPE 位选择 TIM14_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

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 TIM14_ARR 和 TIM14_CCRx 寄存器中。
3. 如果要产生一个中断请求请求，设置 CCxIE 位。
4. 选择输出模式：
   - 写 OCxM=‘011’，当计数器 CNT 与 CCRx 匹配时翻转 OCx 的输出引脚。
   - 写 OCxPE=‘0’，禁止预装载。
   - 写 CCxP=‘0’，选择高电平有效。
   - 写 CCxE=‘1’，允许输出。
5. 设置 TIM14_CR1 寄存器的 CEN 位启动计数器。

TIM14_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 TIM14_CCRx 影子寄存器只能在发生下一次更新事件时被更新）。

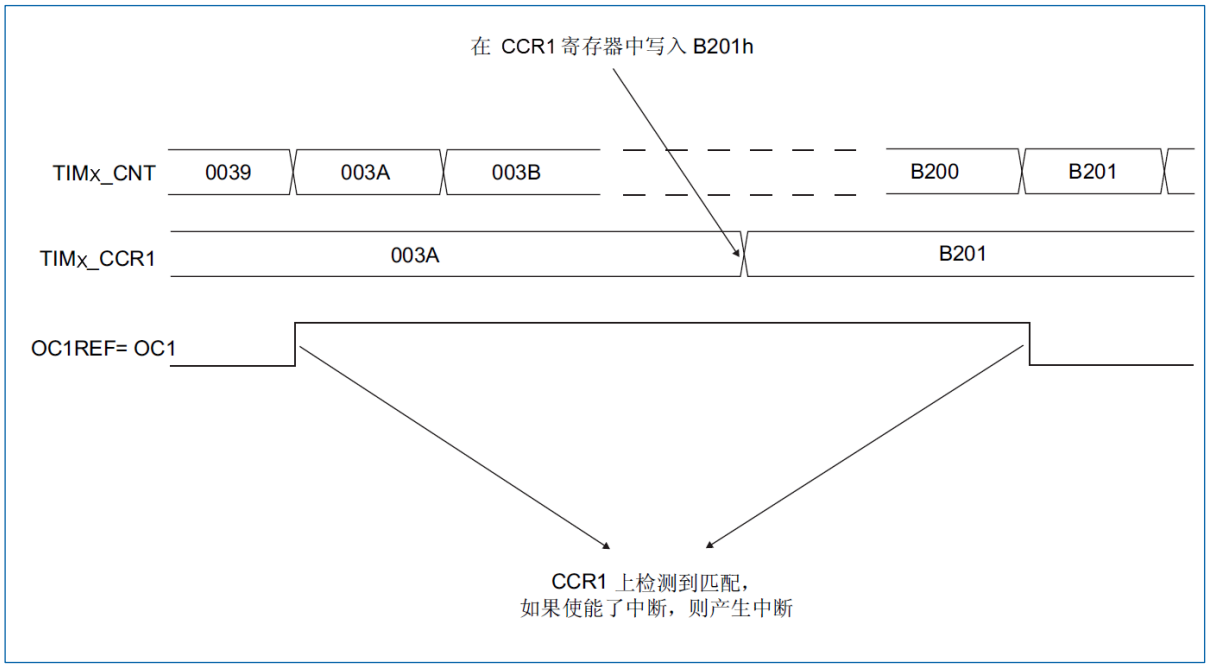


图 15-25 输出比较模式，翻转 0C1

15.2.8 PWM 模式

脉冲宽度调制模式可以产生一个由 TIM14_ARR 寄存器确定频率、由 TIM14_CCRx 寄存器确定占空比的信号。

在 TIM14_CCMRx 寄存器中的 OCxM 位写入’110’（PWM 模式 1）或’111’（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须设置 TIM14_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 TIM14_CR1 寄存器的 ARPE 位，（在向上计数或中心对称模式中）使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIM14_EGR 寄存器中的 UG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TIM14_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。TIM14_CCER 寄存器中的 CCxE 位控制 OCx 输出使能。参见“15.3.6 TIM14 捕捉/比较使能寄存器（TIM14_CCER）”。

在 PWM 模式（模式 1 或模式 2）下，TIM14_CNT 和 TIM14_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 TIM14_CCRx≤TIM14_CNT 或者 TIM14_CNT≤TIM14_CCRx。

15.2.8.1 PWM 边沿对齐模式

向上计数配置

当 TIM14_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参见“15.2.2.1 向上计数模式”。

下面是一个 PWM 模式 1 的例子。当 TIM14_CNT<TIM14_CCRx 时 PWM 信号参考 OCxREF 为高，否则为低。如果 TIM14_CCRx 中的比较值大于自动重装载值（TIM14_ARR），则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。下图为 TIM14_ARR=8 时边沿对齐的 PWM 波形实例。

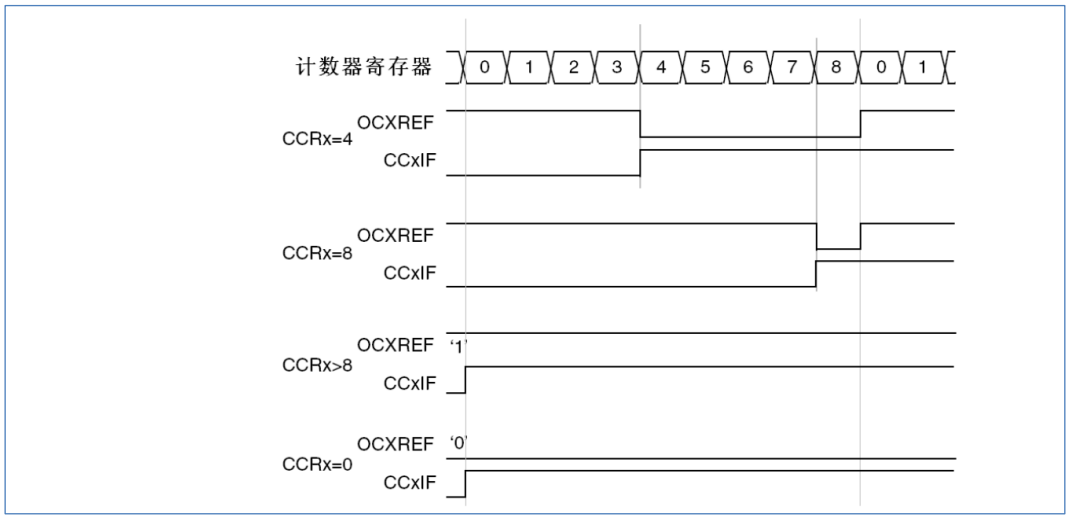


图 15-26 边沿对齐的 PWM 波形（ARR=8）

向下计数的配置

当 TIM14_CR1 寄存器的 DIR 位为高时执行向下计数。参见“15.2.2.2 向下计数模式”。

在 PWM 模式 1，当 TIM14_CNT>TIM14_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIM14_CCRx 中的比较值大于 TIM14_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。

15.2.8.2 PWM 中央对齐模式

当 TIM14_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’。TIM14_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“15.2.2.3 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。

- TIM14_ARR=8
- PWM 模式 1
- TIM14_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。

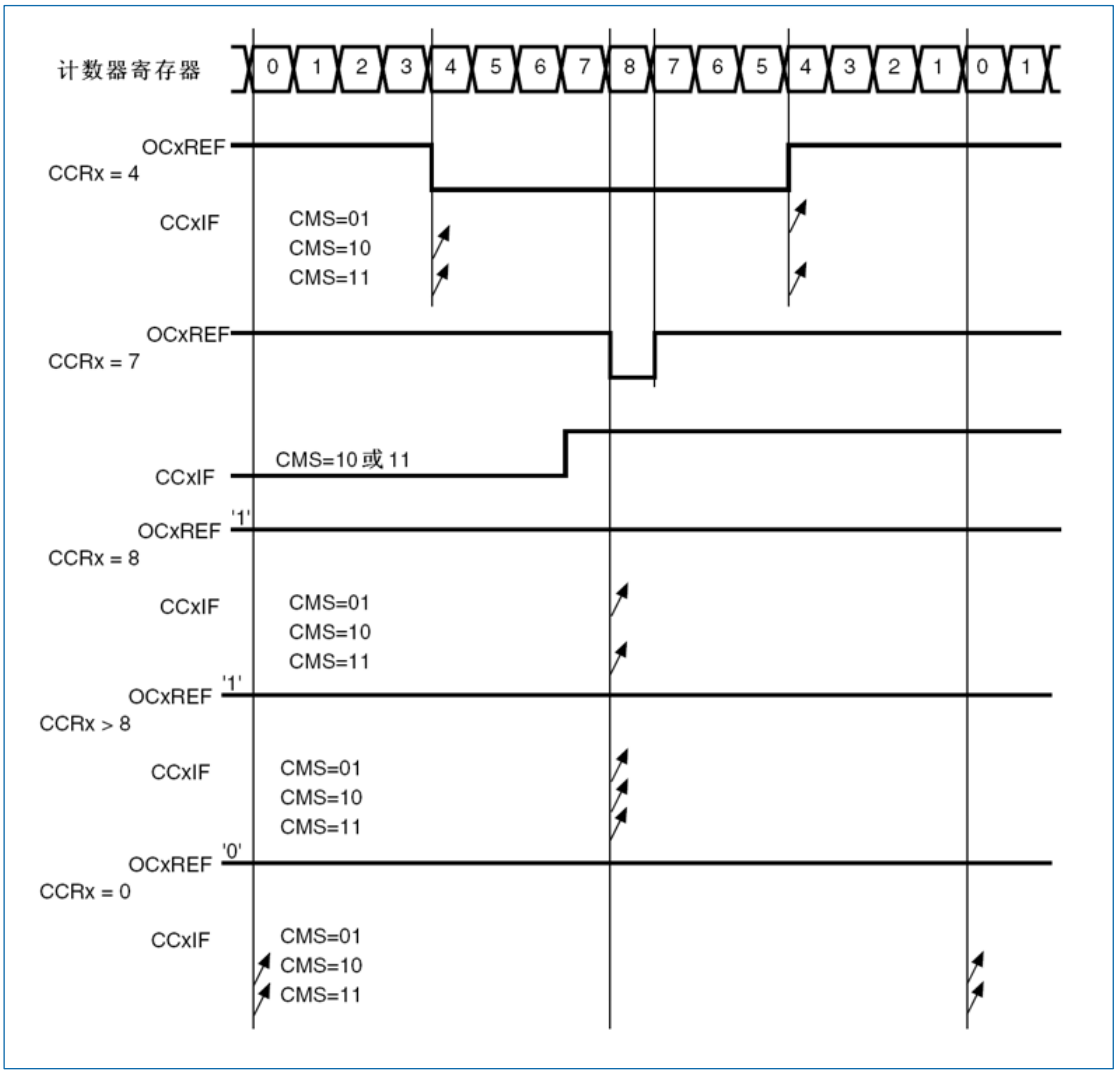


图 15-27 中央对齐的 PWM 波形（APR=8）

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 TIM14_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（TIM14_CNT>TIM14_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 TIM14_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 TIM14_EGR 位中的 UG 位），不要在计数进行时修改计数器的值。

15.2.9 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 TIM14_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

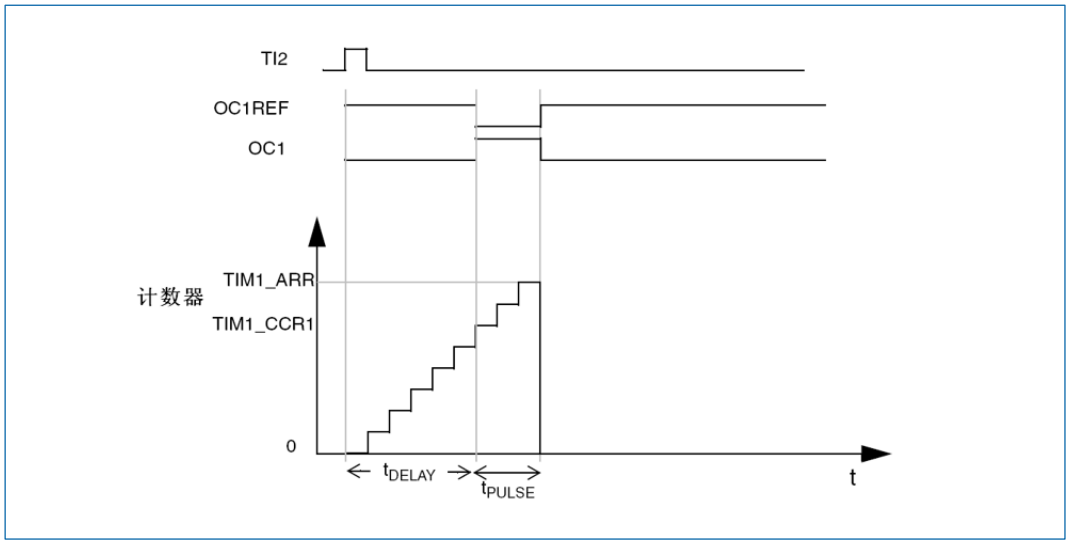


图 15-28 单脉冲模式的例子

例如，你需要在从 TI1 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 TI1FP1 为触发 1：

- 置 TIM14_CCMR1 寄存器中的 CC1S='01'，把 TI1FP1 映射到 TI1。
- 置 TIM14_CCER 寄存器中的 CC1P='0'，使 TI1FP1 能够检测上升沿。

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 TIM14_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（TIM14_ARR-TIM14_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 TIM14_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 TIM14_CCMR1 中的 OC1PE='1'和 TIM14_CR1 寄存器中的 ARPE；然后在 TIM14_CCR1 寄存器中填写比较值，在 TIM14_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，TIM14_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 TIM14_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 TIM14_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。

15.2.10 调试模式

当微控制器进入调试模式（Cortex-M0 内核停止），根据 DBGMCU 模块中 DBG_TIM14_Stop 的设置，TIM14 计数器继续正常操作或者停止。参见"26.8.2 对定时器、看门狗和 I2C 的调试支持"。

15.3 TIM14 寄存器

基地址：0x4000 2000

空间大小：0x400

15.3.1 TIM14 控制寄存器 1（TIM14_CR1）

偏移地址：0x00

复位值：0x0000

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

位 15:10	Res：保留 必须保持复位值。
位 9:8	CKD[1:0]：时钟分频因子（Clock division） 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：TIM14_ARR 寄存器没有缓冲 • 1：TIM14_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（TIM14_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（TIM14_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（TIM14_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。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

15.3.2 TIM14DMA/中断允许寄存器（TIM14_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														CC1IE	UIE
														rw	rw

位 15:2	Res：保留 必须保持复位值。
位 1	CC1IE：捕捉/比较 1 中断使能（Capture/Compare 1 interrupt enable） • 0：CC1 通道中断禁止 • 1：CC1 通道中断允许
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

15.3.3 TIM14 状态寄存器（TIM14_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						CC1OF	Res							CC1IF	UIF
						rc_w0								rc_w0	rc_w0

位 15:10	Res：保留 必须保持复位值。
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 TIM14_CCR1 寄存器。
位 8:2	Res：保留 必须保持复位值。
位 1	CC1IF：捕捉/比较 1 中断标志（Capture/Compare 1 interrupt flag） • 如果通道 CC1 通道配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外。它由软件清'0'。 o 0：无匹配发生。 o 1：TIM14_CNT 的值与 TIM14_CCR1 的值匹配。 当 TIM14_CCR1 的内容大于 TIM14_ARR 的内容时，在向上模式时计数器溢出，CC1IF 位变高。 • 如果通道 CC1 通道配置为输入模式： 当捕获事件发生时该位由硬件置'1'，它由软件清'0'或通过读 TIM14_CCR1 清'0'。 o 0：无输入捕获产生。 o 1：计数器值被捕获至 TIM14_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。当寄存器发生以下更新事件时，该位由硬件置'1'： o 若 TIM14_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。 o 若 TIM14_CR1 寄存器的 URS=0、UDIS=0，当设置 TIM14_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。 o 若 TIM14_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。

15.3.4 TIM14 事件产生寄存器（TIM14_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														CC1G	UG
														w	w

位 15:2	Res：保留 必须保持复位值。
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，由硬件自动清零。 • 0：不起作用 • 1：在通道 1 上产生一个捕获/比较事件。 o 若通道 CC1 通道配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 o 若通道 CC1 通道配置为输入： 当前的计数器值被捕获至 TIM14_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’，由硬件自动清零。 • 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）。若在中央对称模式下或 DIR=0（向上计数）则计数器被清’0’；

15.3.5 TIM14 捕捉/比较模式寄存器 1（TIM14_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。

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
Res									OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
Res								IC1F[3:0]				IC1PSC[1:0]			
								rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式：

位 15:7	Res：保留 必须保持复位值。

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 TIM14_CCR1 与计数器 TIM14_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 TIM14_CNT 的值与捕获/比较寄存器 1（TIM14_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 TIM14_CNT 的值与捕获/比较寄存器 1（TIM14_CCR1）相同时，强制 OC1REF 为低。 • 011：翻转 当 TIM14_CCR1=TIM14_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 TIM14_CNT<TIM14_CCR1 时通道 1 为有效电平，否则为无效电平。 • 111：PWM 模式 2 在向上计数时，一旦 TIM14_CNT<TIM14_CCR1 时通道 1 为无效电平，否则为有效电平。
位 3	OC1PE：输出比较 1 预装允许（Output compare 1 preload enable） • 0：禁止 TIM14_CCR1 寄存器的预装载功能，可随时写入 TIM14_CCR1 寄存器，并且新写入的数值立即起作用。 • 1：开启 TIM14_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM14_CCR1 的预装载值在更新事件到来时被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 通道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 通道输出的最小延时为 5 个时钟周期。 • 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 通道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 TIM14_SMCR 寄存器的 TS 位选择）。

输入捕捉模式：

位 15:8	Res：保留 必须保持复位值。
位 7:4	IC1F[3:0]：输入捕捉 1 滤波器（Input capture 1 fliter） 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 通道输入（IC1）的预分频系数。一旦 CC1E=0（TIM14_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 TIM14_SMCR 寄存器的 TS 位选择）。

15.3.6 TIM14 捕捉/比较使能寄存器（TIM14_CCER）

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												CC1NP	Res	CC1P	CC1E
												rw		rw	rw

位 15:4	Res：保留 必须保持复位值。
位 3	CC1NP：捕捉/比较 1 互补输出极性（Capture/Compare 1 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	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。
	  ○ 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/Compare 1 output enable）
	• CC1 通道配置为输出：
	  ○ 0：关闭
	OC1 禁止输出，因此 OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
	  ○ 1：开启
	OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、

	OIS1N 和 CC1E 位的值。 • CC1 通道配置为输入 该位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（TIM14_CCR1）。 ○ 0：捕捉禁止 ○ 1：捕捉允许

15.3.7 TIM14 计数寄存器（TIM14_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）

15.3.8 TIM14 预分频寄存器（TIM14_PSC）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频值（Prescaler value） 计数器的时钟频率：CK_CNT =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'。

15.3.9 TIM14 自动重装寄存器（TIM14_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。

15.3.10 TIM14 捕捉/比较寄存器 1（TIM14_CCR1）

偏移地址：0x34

复位值：0x0000

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

位 15:0	CCR1[15:0]：捕捉/比较通道 1 的值（Capture/Compare 1 value） • 若 CC1 通道配置为输出： CCR1 决定了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIM14_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 TIM14_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。

15.3.11 TIM14 选项寄存器（TIM14_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														RMP	
														rw	

位 15:2	Res：保留 必须保持复位值。
位 1:0	RMP：定时器输入 1 重映射（Timer Input 1 remap） 该位由软件清除或设置。 • 00：TIM14 通道 1 连接到 GPIO：参考器件数据手册中关于可变功能映射表。 • 01：RTC_CLK 连接到 TIM14_CH1 输入端。 • 10：HSE/32 连接到 TIM14_CH1 输入端。 • 11：MCO 连接到 TIM14_CH1 输入端。

16 通用定时器（TIM16/17）

通用定时器 16/17 由一个 16 位自动装载计数器组成，由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较和 PWM）。

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

通用定时器 16/17 是完全独立的，它们不共享任何资源。

16.1 TIM16/17 主要功能

通用 TIM16/TIM17 定时器功能包括：

- 16 位向上、向下、向上/下自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值。
- 2 个独立通道支持：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿对齐模式）
  - 单脉冲模式输出
- 可编程死区时间的互补输出
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 以下事件发生时产生中断/DMA：
  - 更新：计数器溢出
  - 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
  - 刹车输入（中断请求）

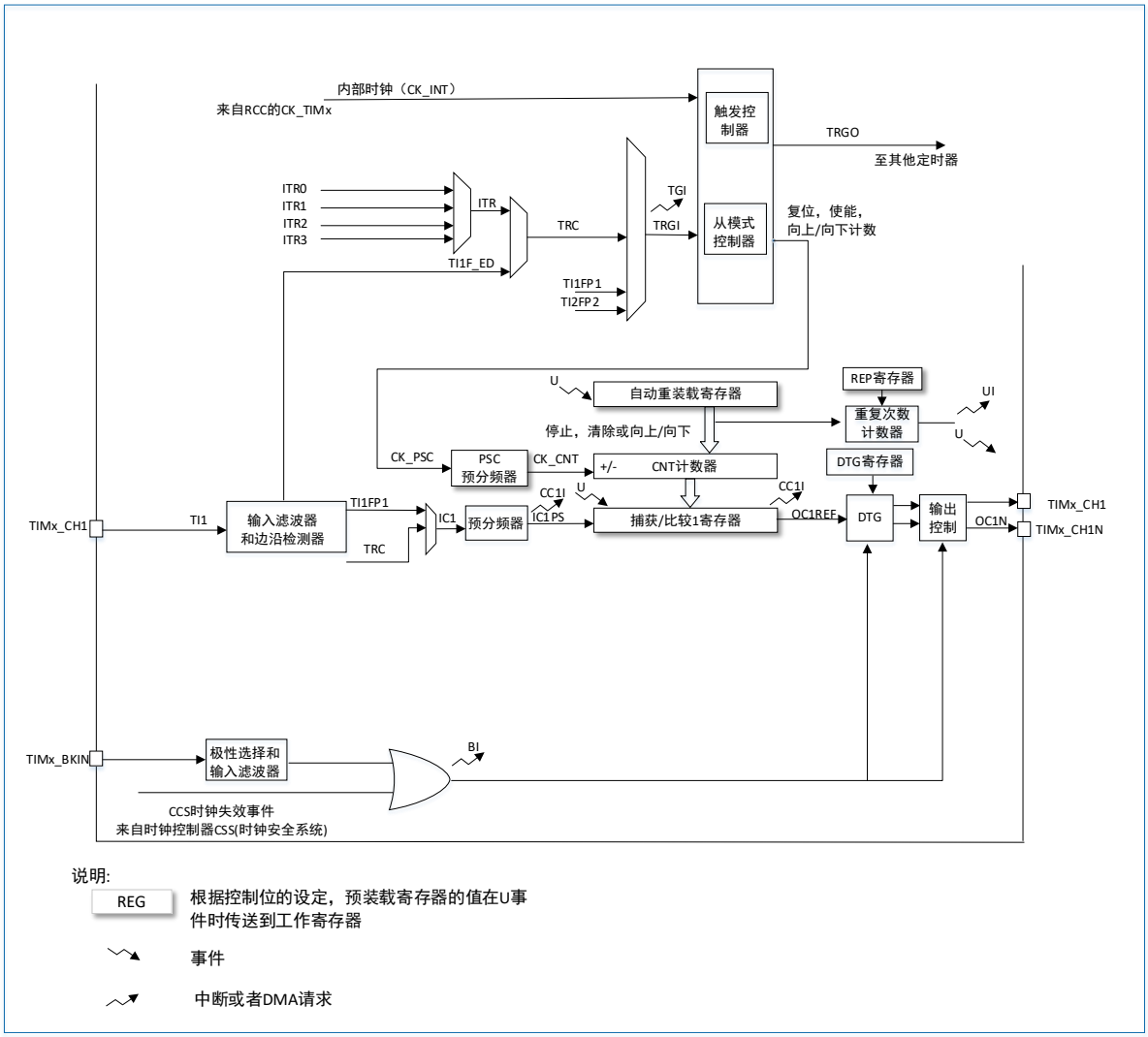


图 16-1 TIM16 和 TIM17 框图

16.2 16/17 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 位计数器和与其相关的自动装载寄存器。这个计数器仅支持向上计数。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，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）
- 重复寄存器（TIMx_RCR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 到来时传送到影子寄存器。当计数器达到溢出条件并当 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：真正的计数器开始计数在 CEN 的一个时钟周期后被设置。

预分频器描述

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

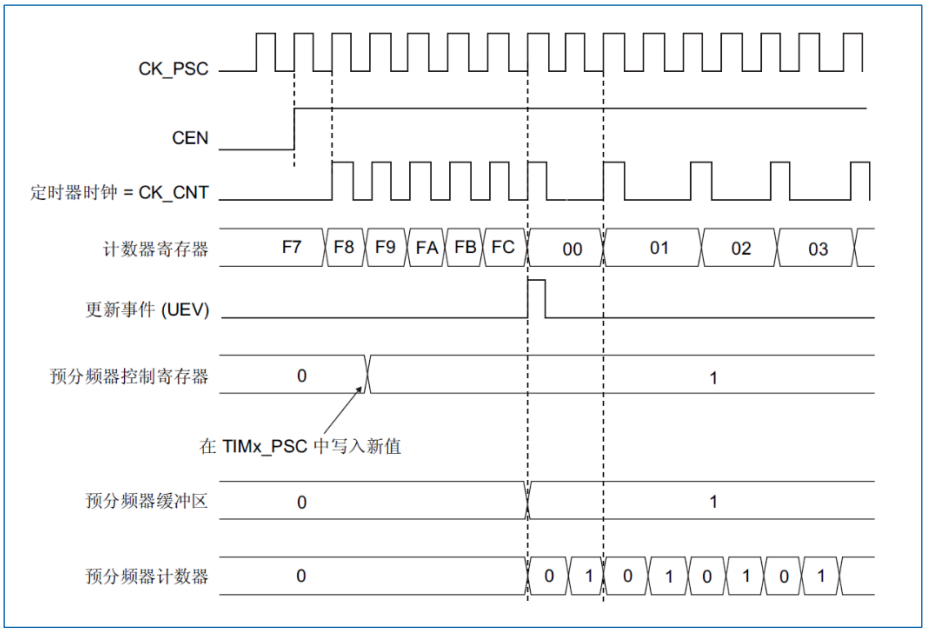


图 16-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

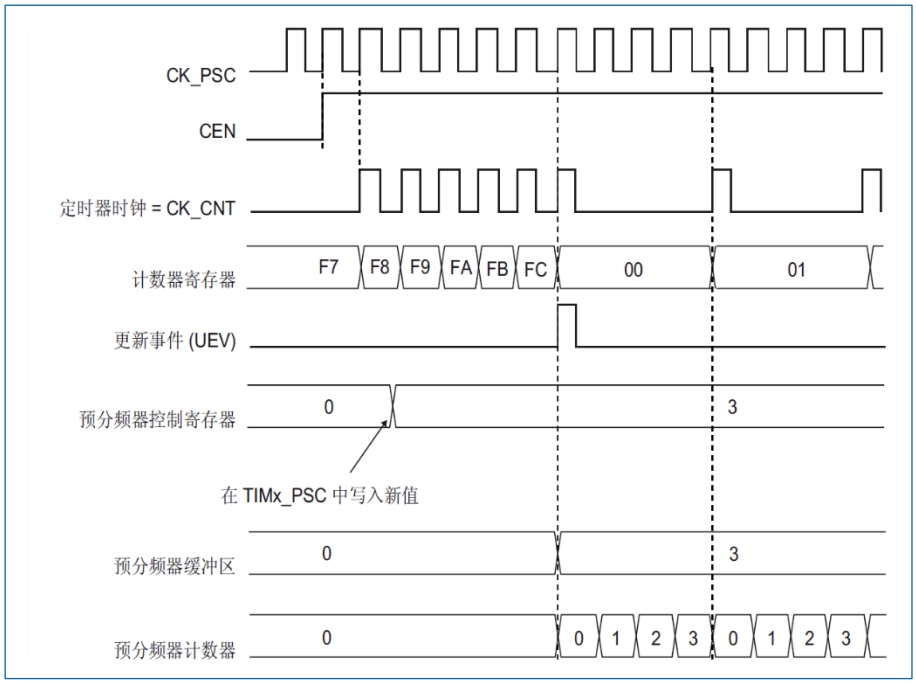


图 16-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

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。

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数（TIMx_RCR）时，产生更新事件

（UEV）；否则每次计数器溢出时才产生更新事件。

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

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清零（但预分频器的数值不变）。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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

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

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

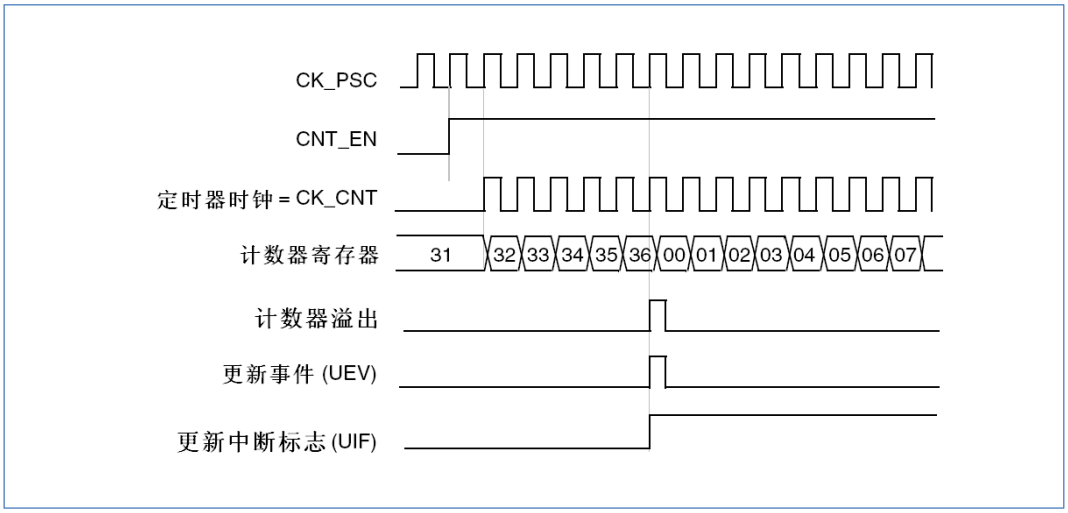


图 16-4 计数器时序图，内部时钟分频因子为 1

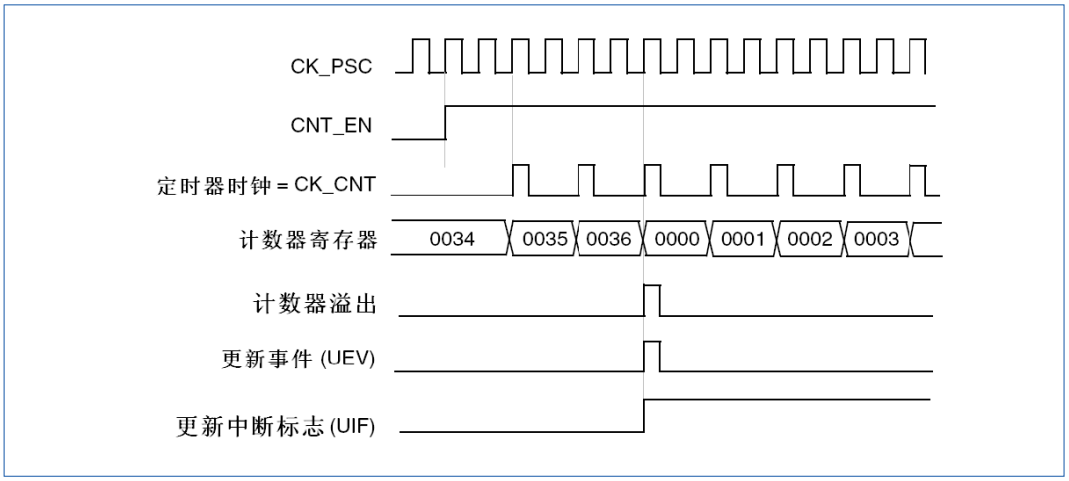


图 16-5 计数器时序图，内部时钟分频因子为 2

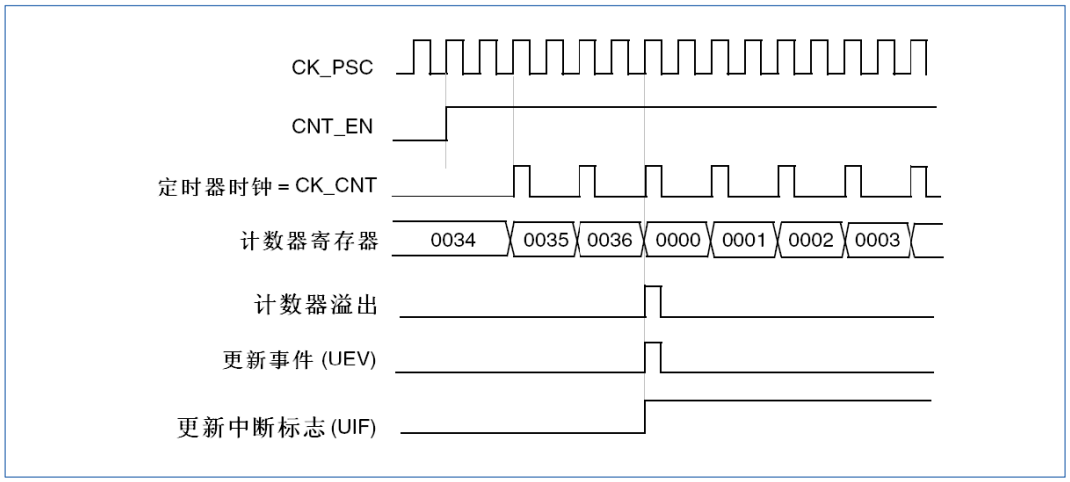


图 16-6 计数器时序图，内部时钟分频因子为 4

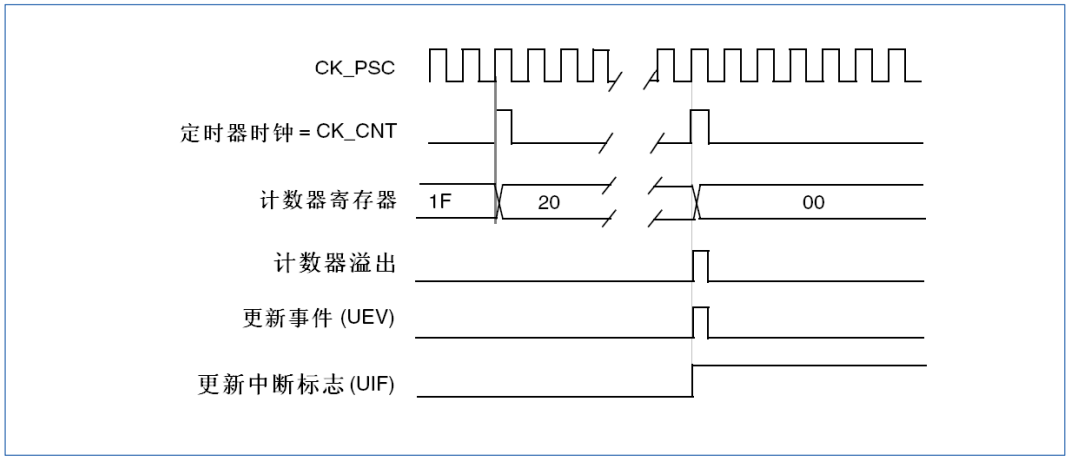


图 16-7 计数器时序图，内部时钟分频因子为 N

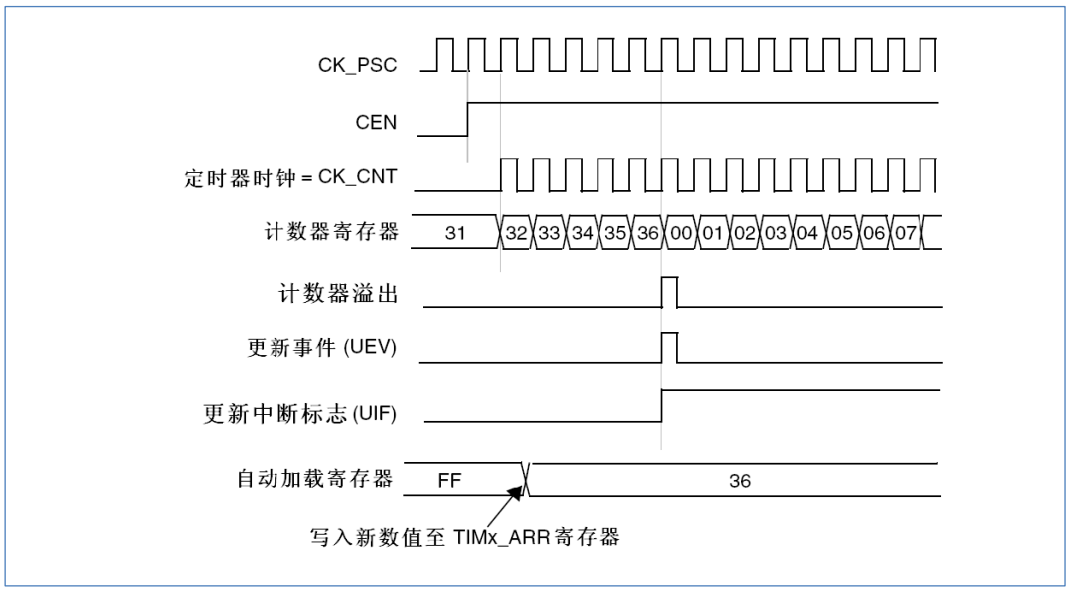


图 16-8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

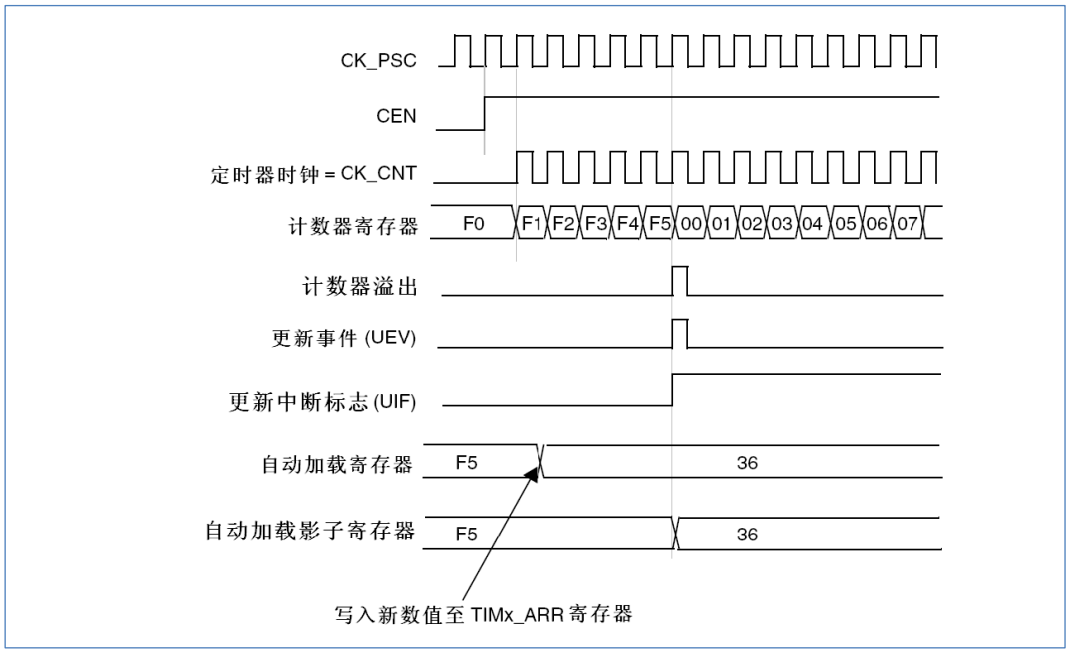


图 16-9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

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。

如果使用了重复计数器，当向下计数重复了重复计数寄存器（TIMx_RCR）中设定的次数后，将产生更新事件（UEV），否则每次计数器下溢时才产生更新事件。

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

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

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

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

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

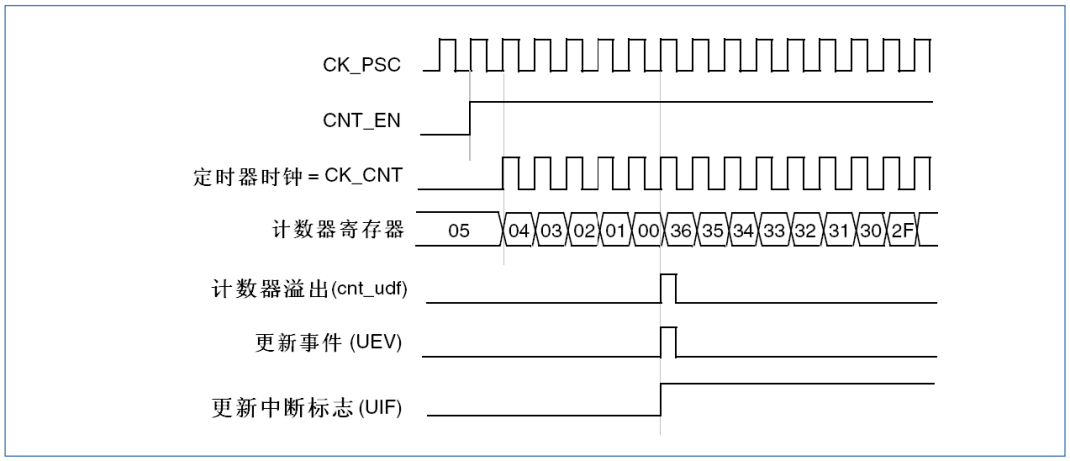


图 16-10 计数器时序图，内部时钟分频因子为 1

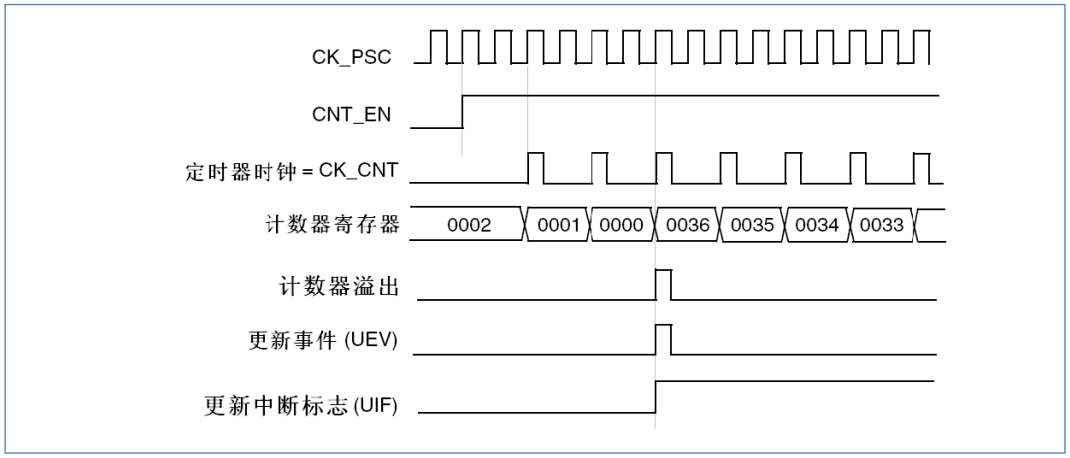


图 16-11 计数器时序图，内部时钟分频因子为 2

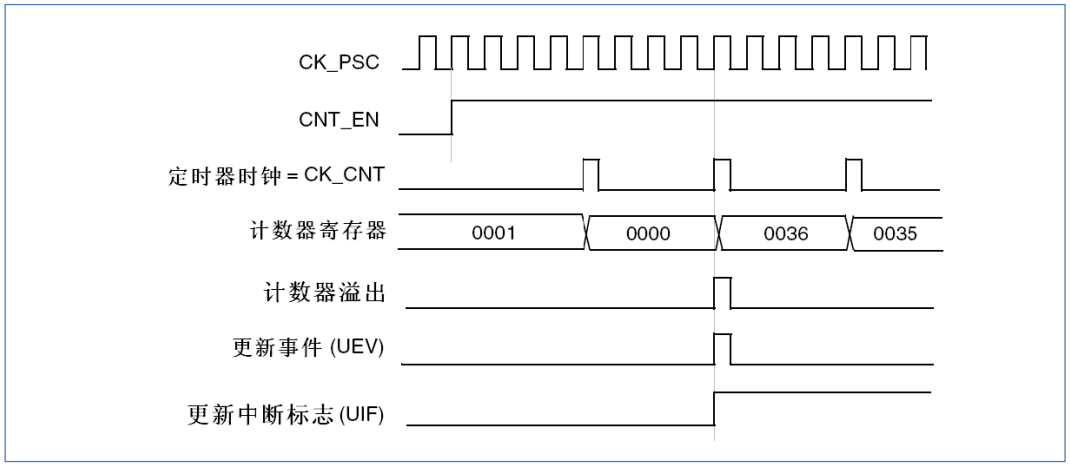


图 16-12 计数器时序图，内部时钟分频因子为 4

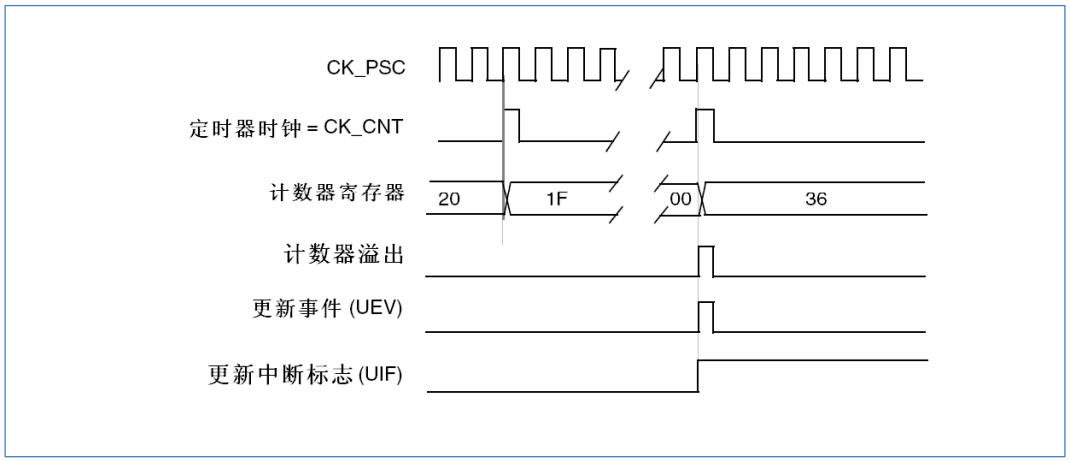


图 16-13 计数器时序图，内部时钟分频因子为 N

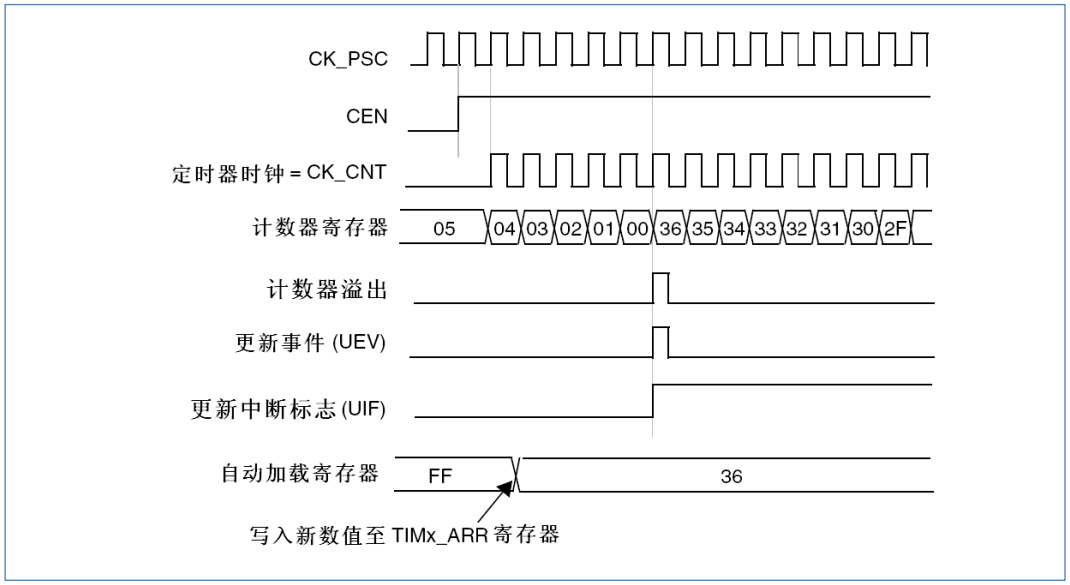


图 16-14 计数器时序图，当没有使用重复计数器时的更新事件

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。

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

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

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

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

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容。

- 预分频器的缓存器被加载为预装载（TIMx_PSC 寄存器）的值。
- 当前的自动加载寄存器被更新为预装载值（TIMx_ARR 寄存器中的内容）。注意：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：

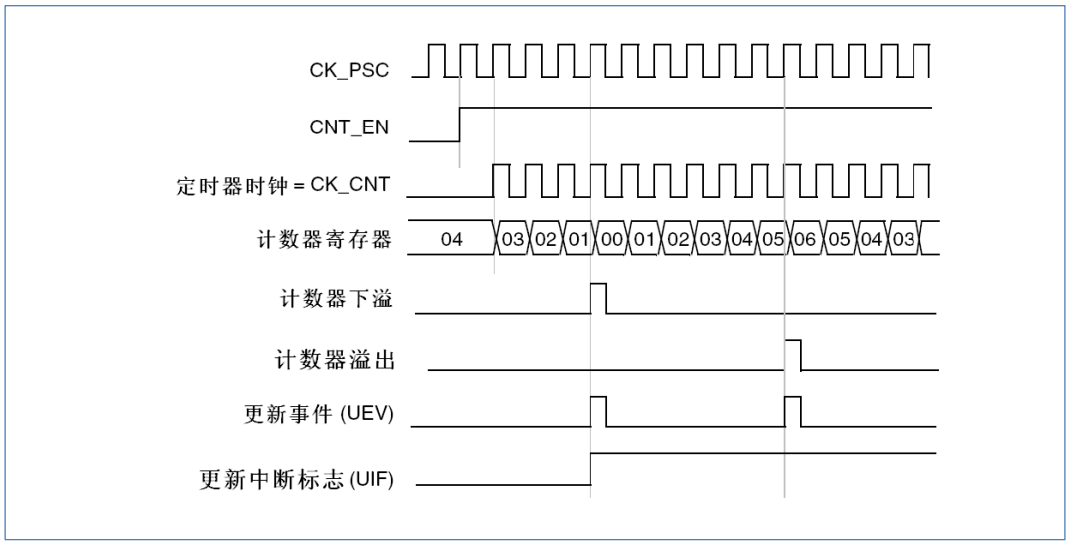


图 16-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6（这里使用了中央对齐模式 1）

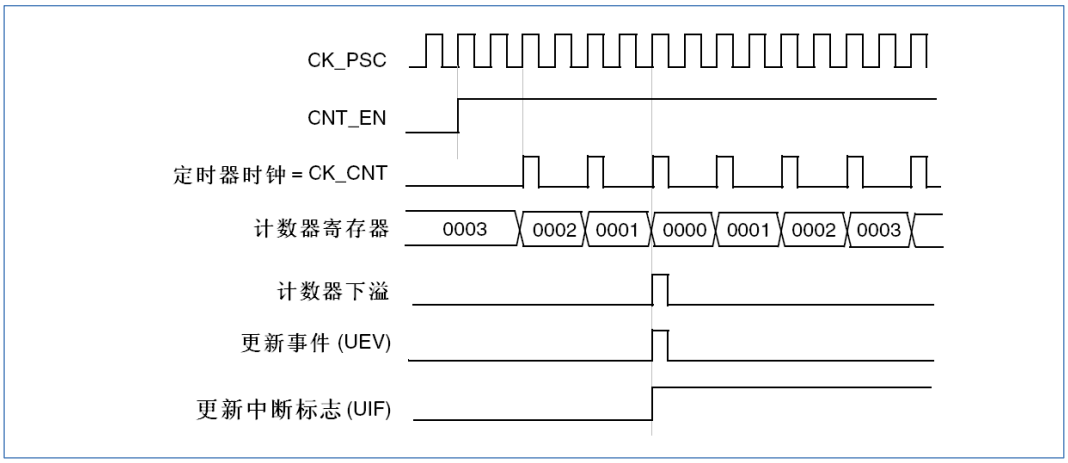


图 16-16 计数器时序图，内部时钟分频因子为 2（中央对齐模式 1）

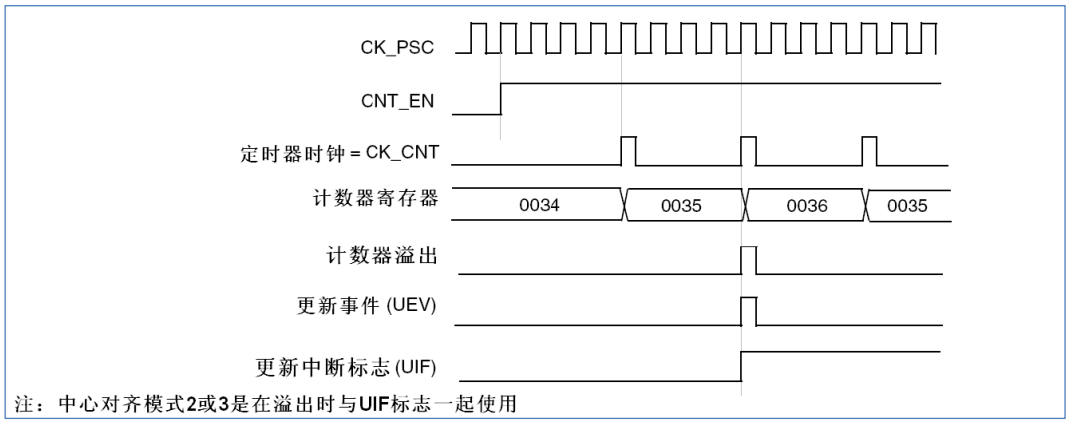


图 16-17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36（中央对齐模式 2 或 3）

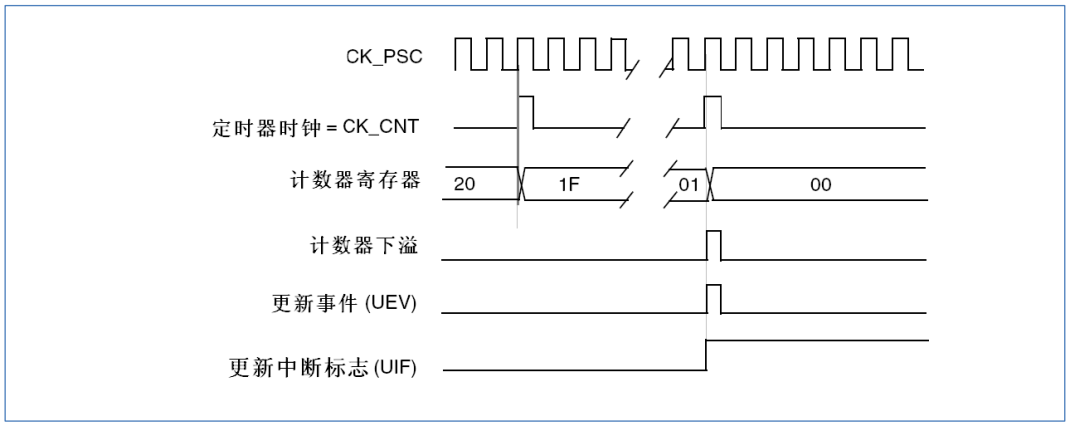


图 16-18 计数器时序图，内部时钟分频因子为 N

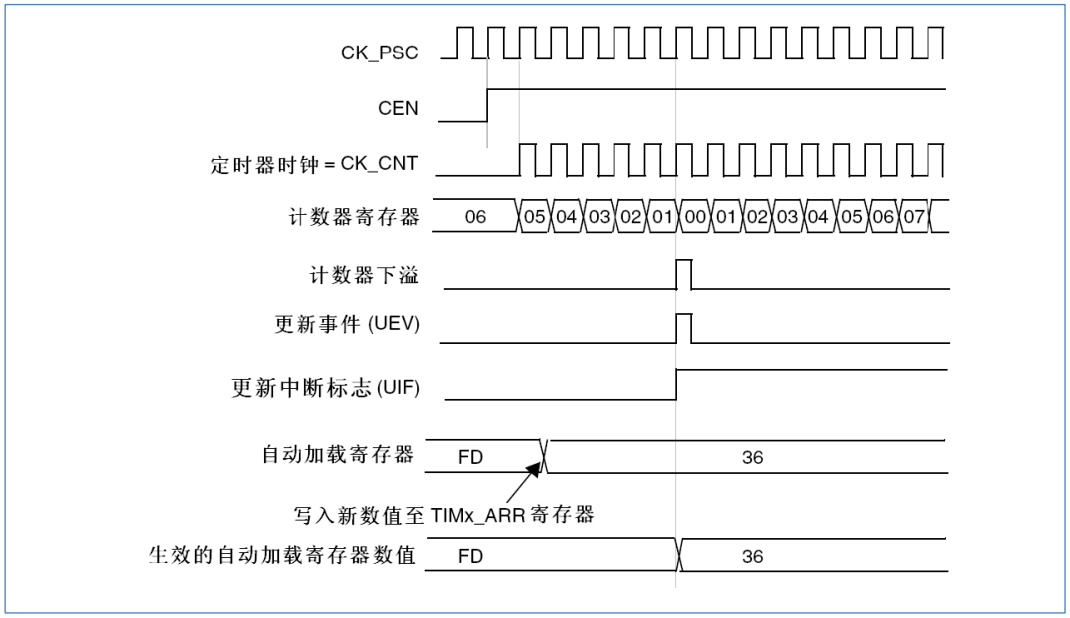


图 16-19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

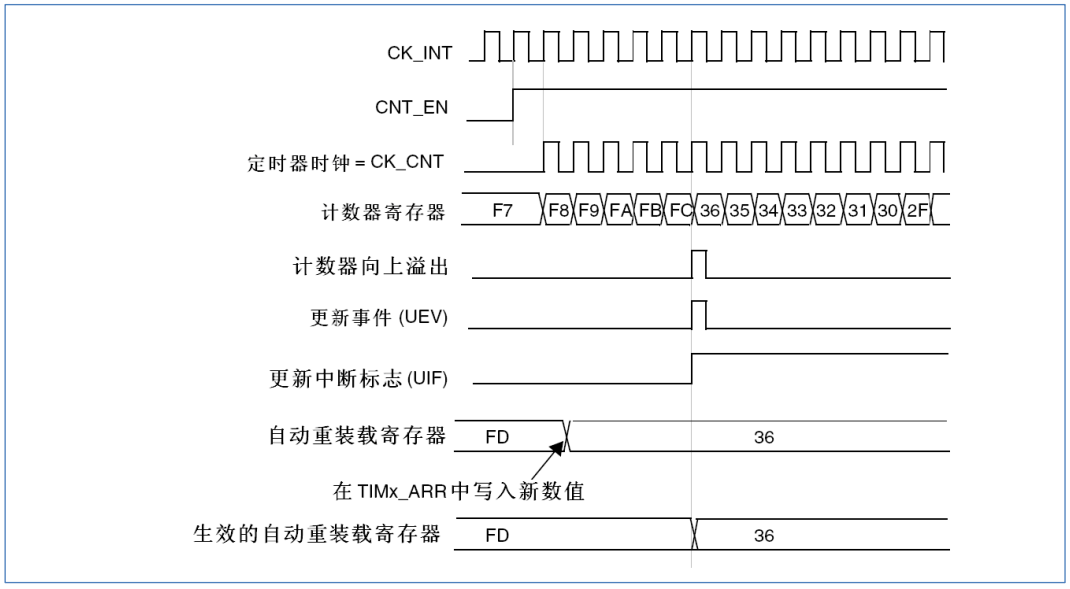


图 16-20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

16.2.3 重复计数器

“时基单元”解释了计数器上溢/下溢时是如何产生更新事件（UEV）的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器（TIMx_ARR 自动重载入寄存器、TIMx_PSC 预装载寄存器、还有在比较模式下的捕获/比较寄存器 TIMx_CCRx），N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

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

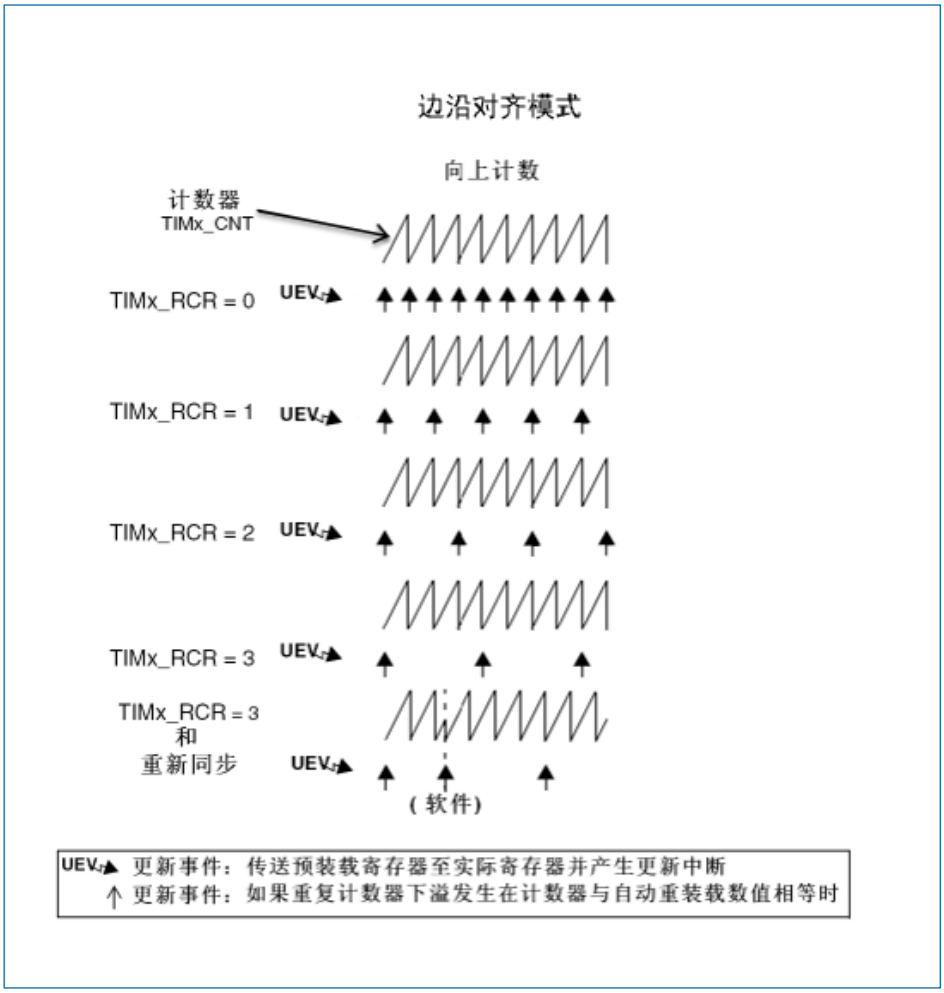


图 16-21 不同模式及不同 TIMx_RCR 寄存器设置下更新速率的例子

16.2.4 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：外部输入脚（TIx）

16.2.4.1 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。

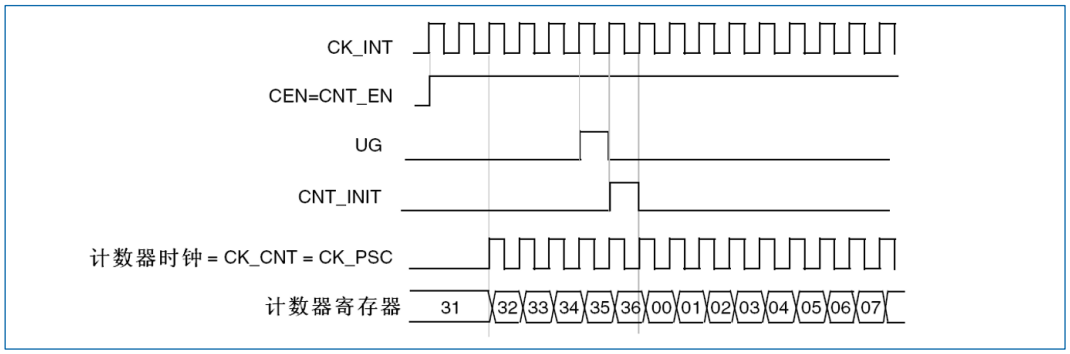


图 16-22 一般模式下的控制电路，内部时钟分频因子为 1

16.2.4.2 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。

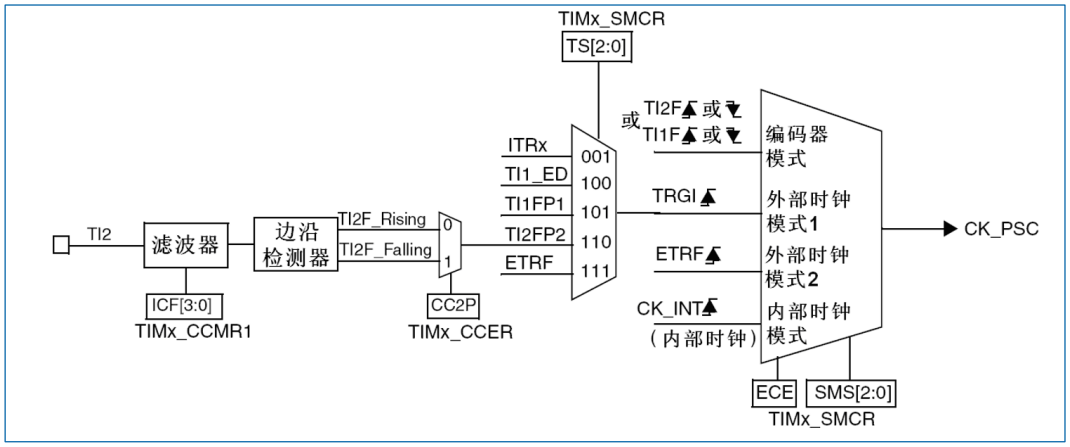


图 16-23 TI2 外部时钟连接示例

例如，配置向上计数器在 TI2 输入端的上升沿计数，其步骤如下：

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

注意：捕获预分频器不用作触发，所以不需要对其进行配置当上升沿出现再 TI2，计数器计数一次，且 TIF 标志被设置。在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。

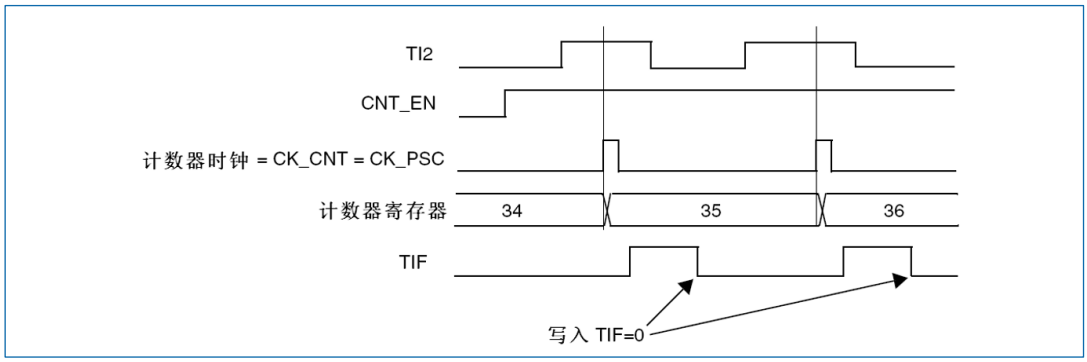


图 16-24 外部时钟模式 1 下的控制电路

16.2.5 捕获/比较通道

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

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）。

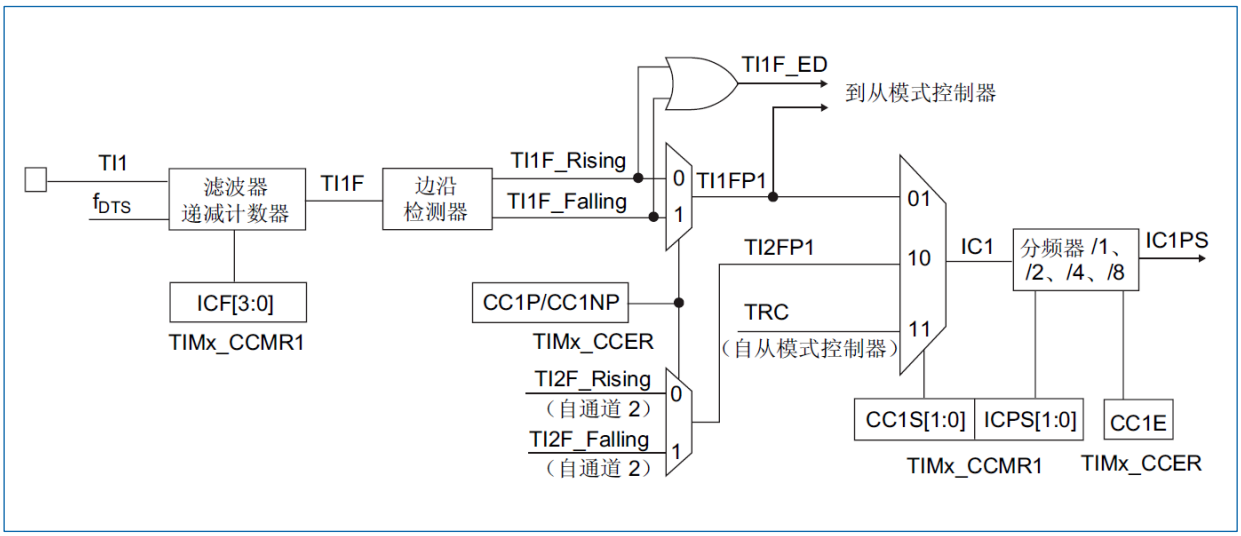


图 16-25 捕获/比较通道（如：通道 1 输入部分）

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。

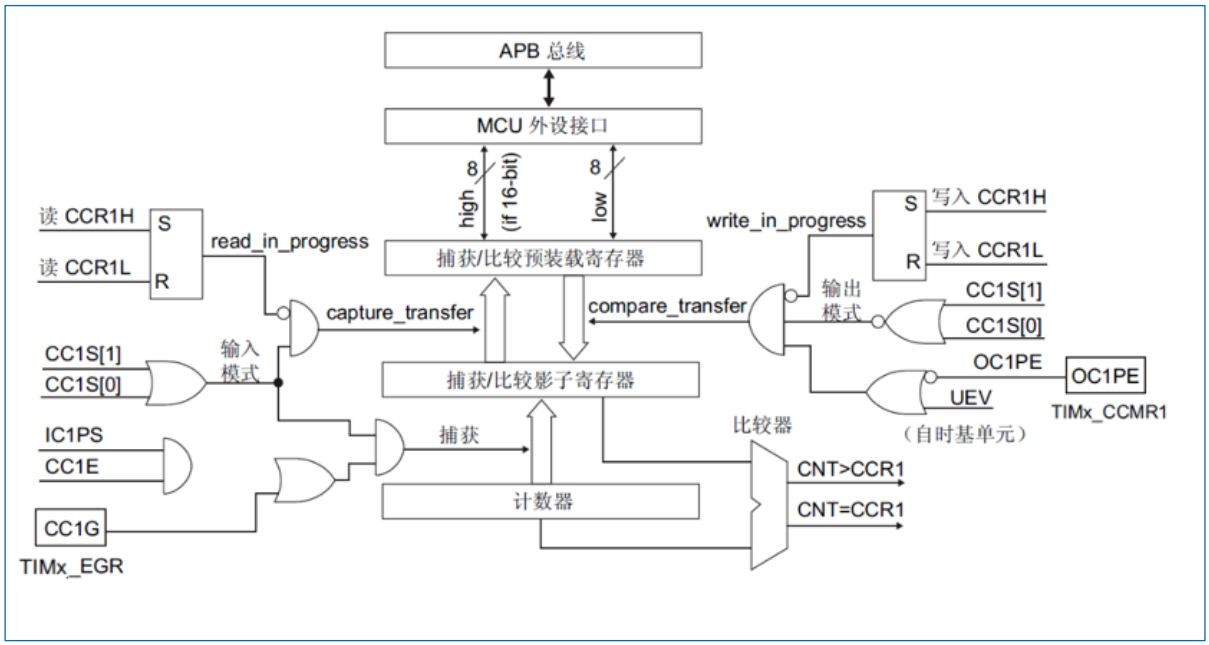


图 16-26 捕获/比较通道 1 的主电路

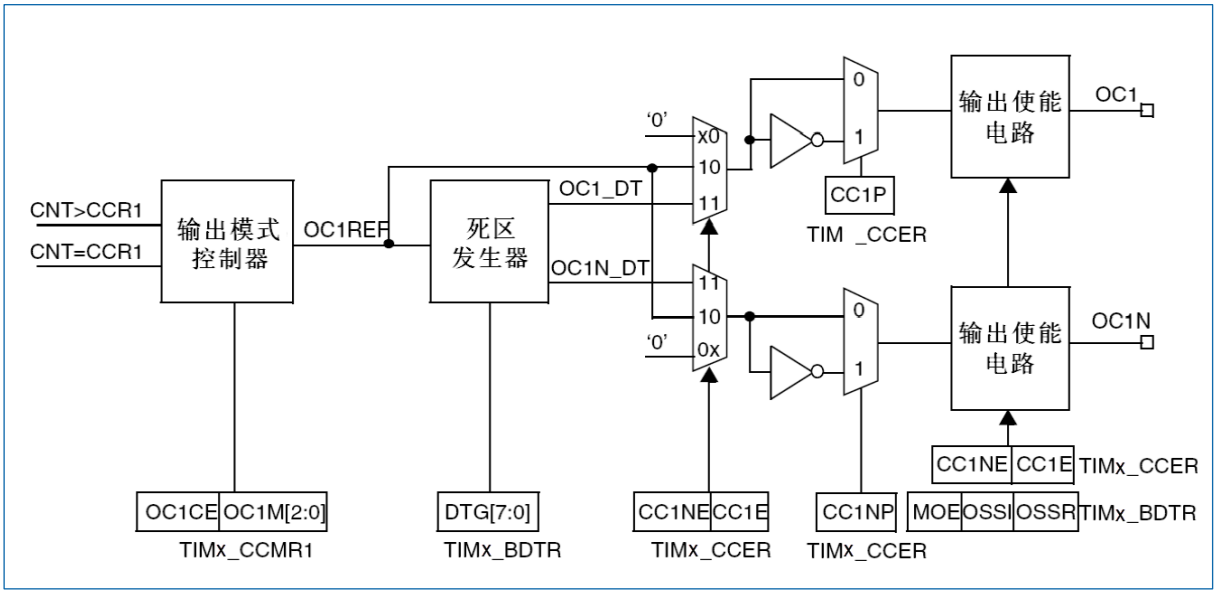


图 16-27 捕获/比较通道的输出部分（通道 1）

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。

16.2.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIMx_CCRx）中。当捕获事件发生时，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：

- 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’，通道则被配置为输入并且 TM1_CCR1 寄存器变为只读。

- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽大于 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。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0（本例中是上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PSC=00）。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

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

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获

溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注意：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。

16.2.7 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN 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

此功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

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

TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部，外部，预分频器）。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式：
   - 置 OCxM=‘011’，CNT 与 CCRx 匹配时翻转 OCx 的输出引脚。
   - 写 OCxPE=‘0’，预装载未用。
   - 写 CCxP=‘0’，高电平有效。
   - 写 CCxE=‘1’，开启 OCx 输出。
5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器。

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

下图给出了一个例子。

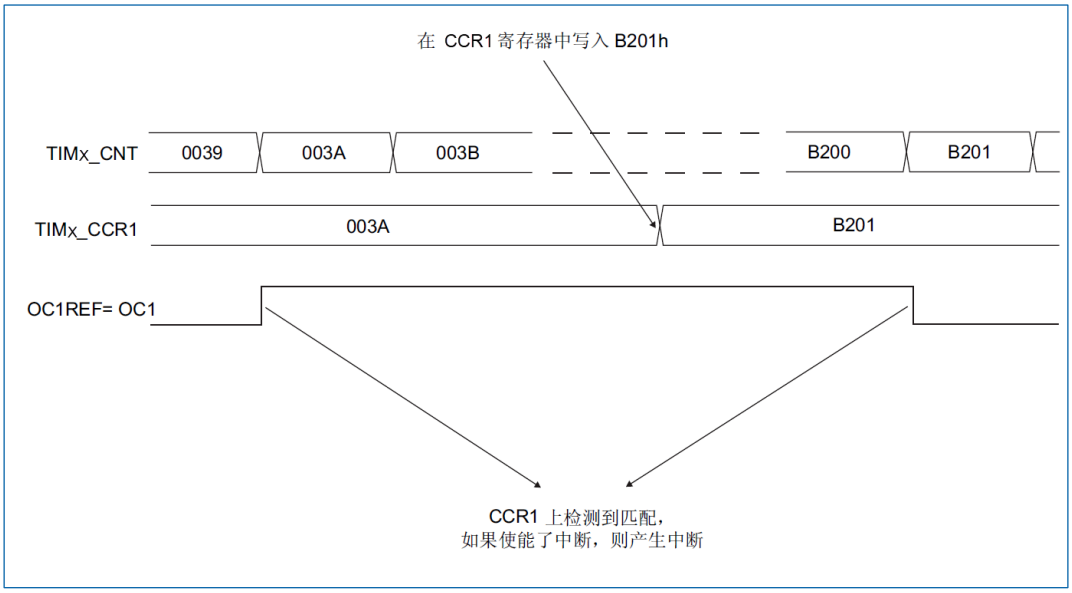


图 16-28 输出比较模式，翻转 OC1

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，（在向上计数或中心对称模式中）使能自动重装载的预装载寄存器。

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

TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。TIMx_CCER 寄存器中的 CCxE 位控制 OCx 输出使能。参见“16.3.7 TIM16/17 捕捉/比较使能寄存器（TIMx_CCER）（x=16..17）”。

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。

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。参见“16.2.2.1 向上计数模式”。

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。

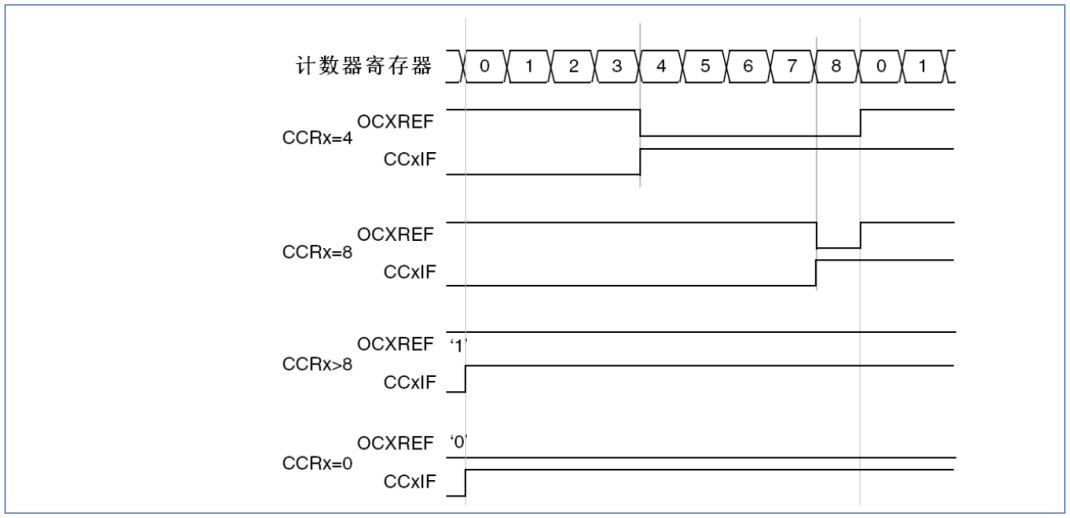


图 16-29 边沿对齐的 PWM 波形（ARR=8）

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。参见“16.2.2.2 向下计数模式”。

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。

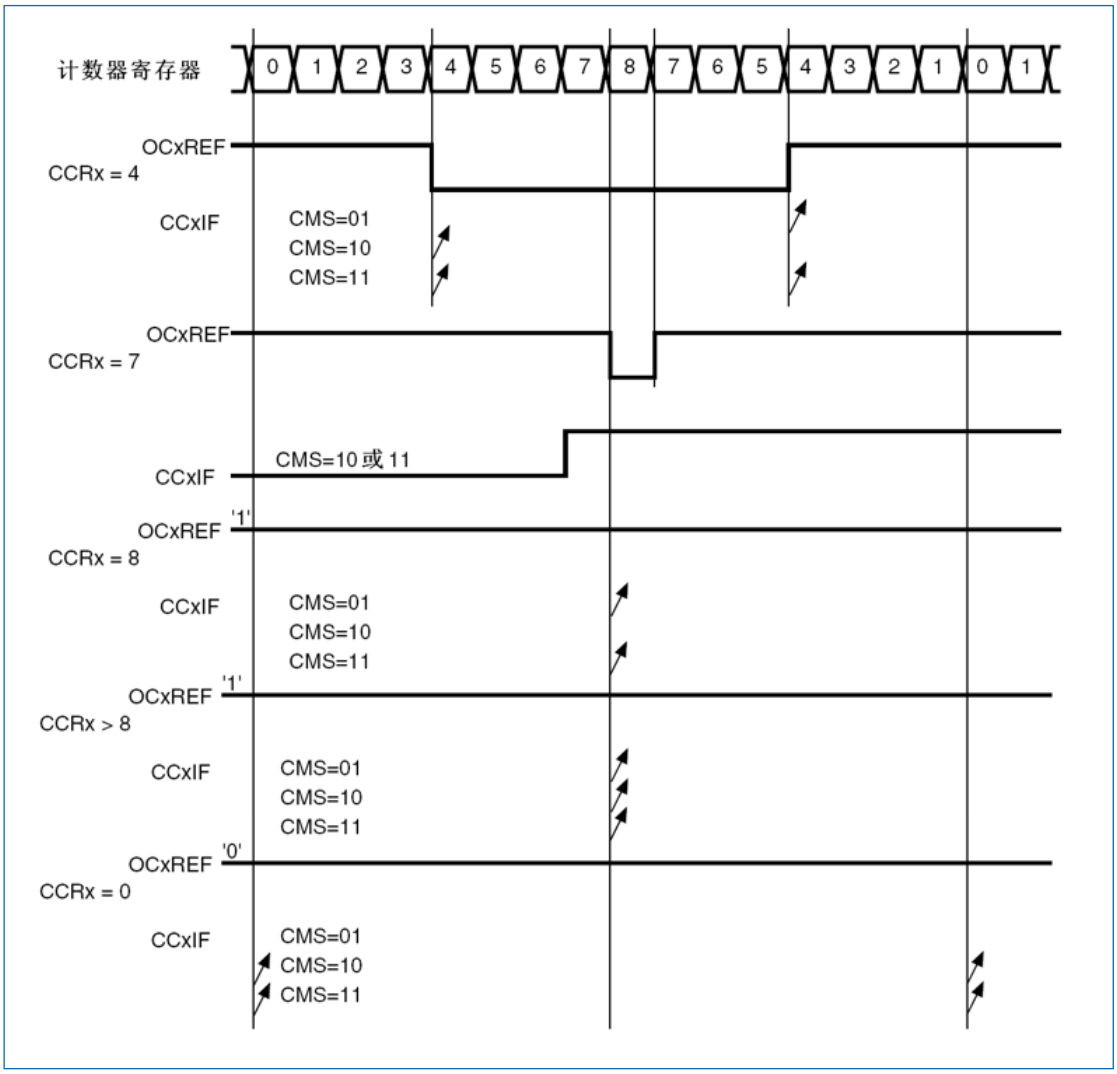


图 16-30 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别是：
  - 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 互补输出和死区插入

通用定时器 TIM16/17 能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。

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

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

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位，参见表 13-3。需要注意：

在转换到 IDLE 状态时（MOE 下降到 0）死区被激活。

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

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

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

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

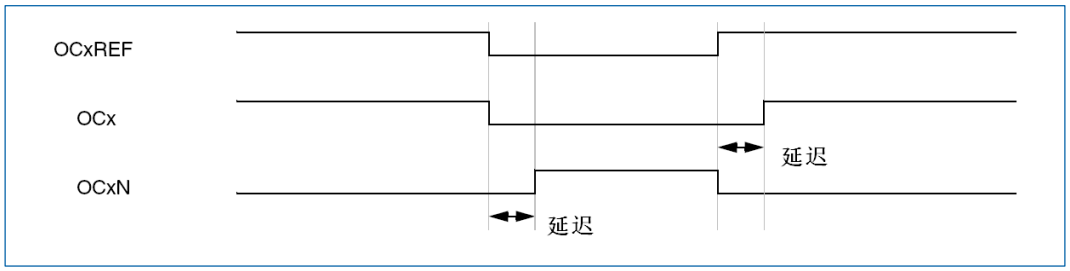


图 16-31 带死区插入的互补输出

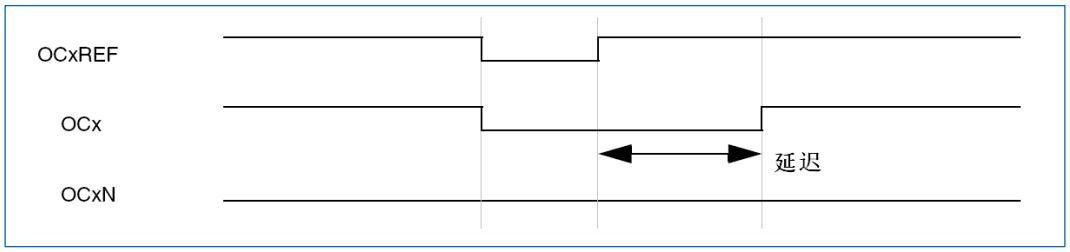


图 16-32 死区波形延迟大于负脉冲

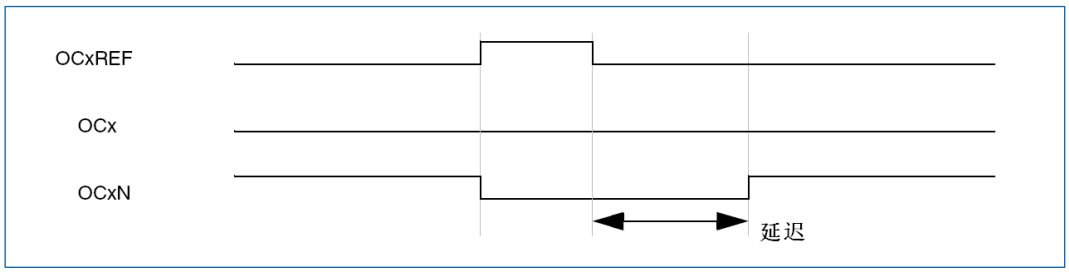


图 16-33 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。参见“16.3.13 TIM16/17 刹车和死区寄存器（TIMx_BDTR）（x=16..17）”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 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注意：当使能 OCxN（CCxE=0，CCxNE=1）时，它不会反相，当 OCxREF 变高时立即有效。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。

16.2.11 使用刹车功能

当使用刹车功能时，依据相应的控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中的 OISx 和 OISxN 位），输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上，参见表 13-3。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统（CSS）产生，参见“6.2.9 时钟安全系统（CSS）”。

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

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

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注：因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志（TIMx_SR 寄存器中的 BIF 位）为’1’时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注意：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。

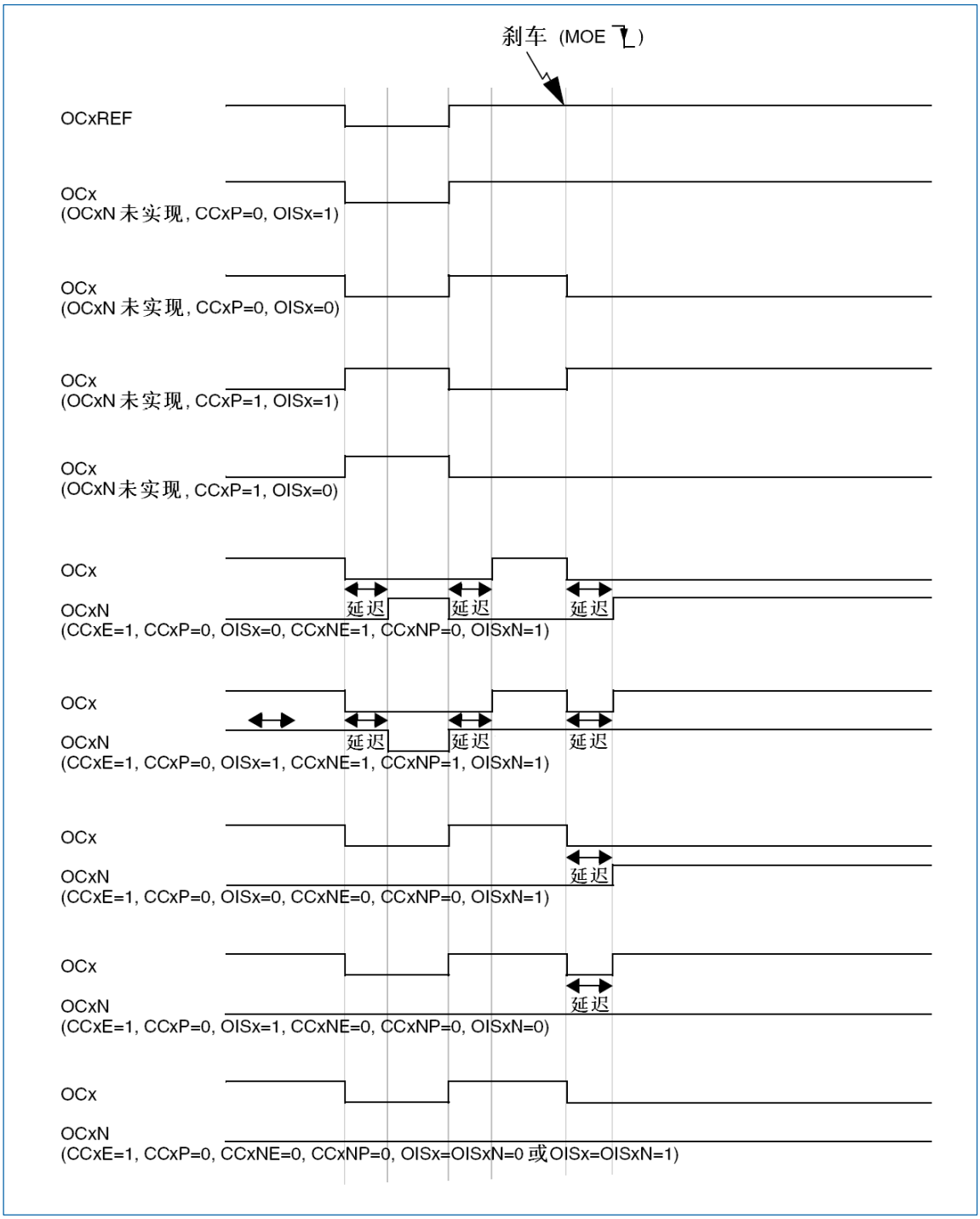


图 16-34 响应刹车的输出

16.2.12 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，产生一个脉宽可编程控制的脉冲。

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：CNT<CCRx≤ARR（特别地：0<CCRx）（仅支持向上计数方式）。

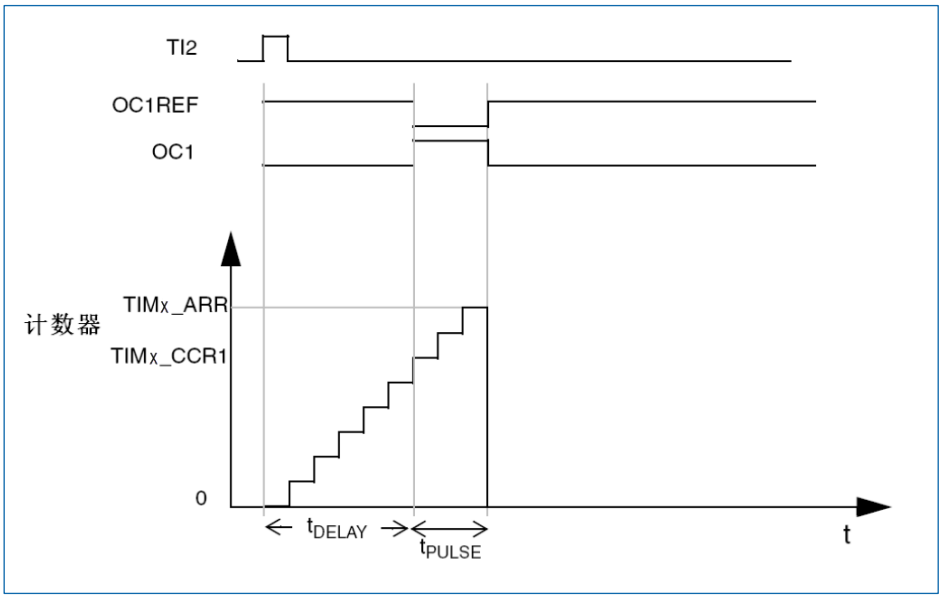


图 16-35 单脉冲模式的例子

例如，用户需要在从 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。设置 TIMx_CR1 寄存器中的 OPM='0'，就会选择重复模式。

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。

16.2.13 调试模式

当微控制器进入调试模式时（Cortex-M0 内核停止），根据 DBG 模块中 DBG_TIMx_Stop 的设置，TIMx

计数器可以或者继续正常操作，或者停止。参见“26.8.2 对定时器、看门狗和 I2C 的调试支持”。

16.3 TIM16/17 寄存器

基地址：（TIM16, TIM17）=（0x4001 4400, 0x4001 4800）

空间大小：（TIM16, TIM17）=（0x400, 0x400）

16.3.1 TIM16/17 控制寄存器 1（TIMx_CR1）（x=16..17）

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

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（该配置未使用）
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（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。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 TIM16/17 控制寄存器 2（TIMx_CR2）（x=16..17）

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						OIS1N	OIS1	Res				CCDS	CCUS	Res	CCPC
						rw	rw					rw	rw		rw

位 15:10	Res：保留 必须保持复位值。
位 9	OIS1N：输出空闲状态 1（OC1N 输出）（Output Idle state 1）（OC1N output） • 0：OC1N=0，当 MOE=0 后一个死区时间。 • 1：OC1N=1，当 MOE=0 后一个死区时间。
位 8	OIS1：输出空闲状态 1（OC1 输出）（Output Idle state 1）（OC1output） • 0：OC1=0，当 MOE=0（如果 OC1N 被使用还要等一个死区时间）。 • 1：OC1=1，当 MOE=0（如果 OC1N 被使用还要等一个死区时间）。

位 7:4	Res：保留 必须保持复位值。
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：捕捉/比较控制更新选择（Capture/Compare control update selection） • 0：当捕捉/比较控制位被预装载时（CCPC=1），只有通过设置 COMG 位才会被更新。 • 1：当捕捉/比较控制位被预装载时（CCPC=1），在设置 COMG 位或在 TRGI 上产生上升沿时都会被更新。
位 1	Res：保留 必须保持复位值。
位 0	CCPC：捕捉/比较预装载控制（Capture/Compare preloaded control） • 0：CCxE、CCxNE 和 OCxM 位不进行预装载。 • 1：CCxE、CCxNE 和 OCxM 位在被写入后被预装载，只有在 COM 位被设置为‘1’时才更新。

16.3.3 TIM16/17 DMA 中断允许寄存器（TIMx_DIER）（x=16..17）

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						CC1DE	UDE	BIE	Res	COMIE	Res			CC1IE	UIE
						rw	rw	rw		rw				rw	rw

位 15:10	Res：保留 必须保持复位值。
位 9	CC1DE：捕捉/比较 1DMA 请求使能（Capture/Compare 1 DMA request enable） • 0：CC1DMA 请求禁止 • 1：CC1DMA 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：刹车中断允许（Break interrupt enable） • 0：刹车中断禁止 • 1：刹车中断允许
位 6	Res：保留 必须保持复位值。

位 5	COMIE：COM 中断允许（COM interrupt enable） • 0：COM 中断禁止 • 1：COM 中断允许
位 4:2	Res：保留 必须保持复位值。
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

16.3.4 TIM16/17 状态寄存器（TIMx_SR）（x=16..17）

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						CC1OF	Res	BIF	Res	COMIF	Res			CC1IF	UIF
						rc_w0		rc_w0		rc_w0				rc_w0	rc_w0

位 15:10	Res：保留 必须保持复位值。
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	Res：保留 必须保持复位值。
位 7	BIF：刹车中断标记（Break interrupt flag） 当刹车输入信号有效时由硬件对该位置’1’。刹车信号无效时可由软件清’0’。 • 0：无刹车事件产生。 • 1：刹车输入端检测到有效电平。
位 6	Res：保留 必须保持复位值。
位 5	COMIF：COM 中断标记（COM interrupt flag） 当发生 COM 事件（一旦捕捉/比较控制位：CCxE、CCxNE、OCxM，被更新）时，由硬件对该位置’1’。它由软件清’0’。 • 0：无 COM 事件产生

	• 1：COM 中断等待响应
位 4:2	Res：保留 必须保持复位值。
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，但在中心对称模式下除外。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，在向上计数模式时计数器溢出，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。当存在以下寄存器更新时，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。

16.3.5 TIM16/17 事件产生寄存器（TIMx_EGR）（x=16..17）

偏移地址：0x14

复位值：0x0000

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

位 15:8	Res：保留 必须保持复位值。
位 7	BG：刹车产生（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：不起作用 • 1：刹车事件产生，清除 MOE，设置 TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。

位 6	Res：保留 必须保持复位值。
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：不起作用 • 1：当 CCPC 位被设置时，才可能更新 CCxE、CCxNE 和 OCxM 位
位 4:2	Res：保留 必须保持复位值。
位 1	CC1G：捕捉/比较 1 发生（Capture/Compare 1generation） 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）。若在中心对称模式下或 DIR=0（向上计数）则计数器被清’0’；

16.3.6 TIM16/17 捕捉/比较模式寄存器 1（TIMx_CCMR1）（x=16..17）

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
Res									OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
Res								IC1F[3:0]				IC1PSC[1:0]			
								rw	rw	rw	rw	rw	rw	rw	

输出比较模式：

位 15:7	Res：保留 必须保持复位值。
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； • 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；
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作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。
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 通道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 通道输出的最小延时为 5 个时钟周期。 • 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 通道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 15:8	Res：保留 必须保持复位值。
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）。

16.3.7 TIM16/17 捕捉/比较使能寄存器（TIMx_CCER）（x=16..17）

偏移地址：0x20

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res												CC1NP	CC1NE	CC1P	CC1E
												rw	rw	rw	rw

位 15:4	Res：保留

	必须保持复位值。
位 3	CC1NP：捕捉/比较 1 输出极性（Capture/Compare 1 complementary output polarity） • 0：OC1N 高有效 • 1：OC1N 低有效
位 2	CC1NE：捕捉/比较 1 互补输出使能（Capture/Compare 1 complementary output enable） • 0：关闭 OC1N 无效。 • 1：开启 OC1N 输出到相关输出引脚的信号取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位。
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。 ○ 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）。
位 0	CC1E：捕捉/比较 1 输出使能（Capture/Compare 1 output enable） • CC1 通道配置为输出： ○ 0：关闭 OC1 禁止输出，因此 OC1 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 ○ 1：开启 OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 • CC1 通道配置为输入： 该位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

16.3.8 TIM16/17 计数器寄存器（TIMx_CNT）（x=16..17）

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）

16.3.9 TIM16/17 预分频寄存器（TIMx_PSC）（x=16..17）

偏移地址：0x28

复位值：0x0000

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

位 15:0	PSC[15:0]：预分频值（Prescaler value） 计数器的时钟频率：CK_CNT =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'。

16.3.10 TIM16/17 自动重装寄存器（TIMx_ARR）（x=16..17）

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。

16.3.11 TIM16/17 重复计数寄存器（TIMx_RCR）（x=16..17）

偏移地址：0x30

复位值：0x0000

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

位 15:8	Res：保留 必须保持复位值。
位 7:0	REP[7:0]：重复计数器的值（Repetition counter value） 预装载寄存器被使能后，该位域允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中

	断的速率。

16.3.12 TIM16/17 捕捉/比较寄存器 1（TIMx_CCR1）（x=16..17）

偏移地址：0x34

复位值：0x0000

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

位 15:0	CCR1[15:0]：捕捉/比较通道 1 的值（Capture/Compare 1 value） • 若 CC1 通道配置为输出： CCR1 决定了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 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。

16.3.13 TIM16/17 刹车和死区寄存器（TIMx_BDTR）（x=16..17）

偏移地址：0x44

复位值：0x0000

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

位 15	MOE：主输出使能（Main output enable） 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 • 0：禁止 OC 和 OCN 输出或强制为空闲状态。 • 1：如果设置了相应的使能位（TIMx_CCER 寄存器的 CCxE、CCxNE 位），则开启 OC 和 OCN 输出。
位 14	AOE：自动输出使能（Automatic output enable） • 0：MOE 只能被软件置’1’。 • 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’（如果刹车输入无效）。
位 13	BKP：刹车输入极性（Break polarity） • 0：刹车输入低电平有效 • 1：刹车输入高电平有效
位 12	BKE：刹车使能（Break enable） • 0：刹车输入禁止（BRK 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 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位。 • 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位。 • 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位）。
位 7:0	DTG[7:0]：死区发生器设置（Dead-time generator setup） 该位域定义了插入互补输出之间的死区持续时间。

16.3.14 TIM16/17 DMA 控制寄存器（TIMx_DCR）（x=16..17）

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 定义为从 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。

16.3.15 TIM16/17 DMA 完全传送地址寄存器（TIMx_DMAR）（x=16..17）

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 位。 DMA 连续传送功能使用方法示例，参见“13.3.20 TIM1 全部传输时 DMA 地址寄存器（TIM1_DMAR）”。

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/DMA 请求

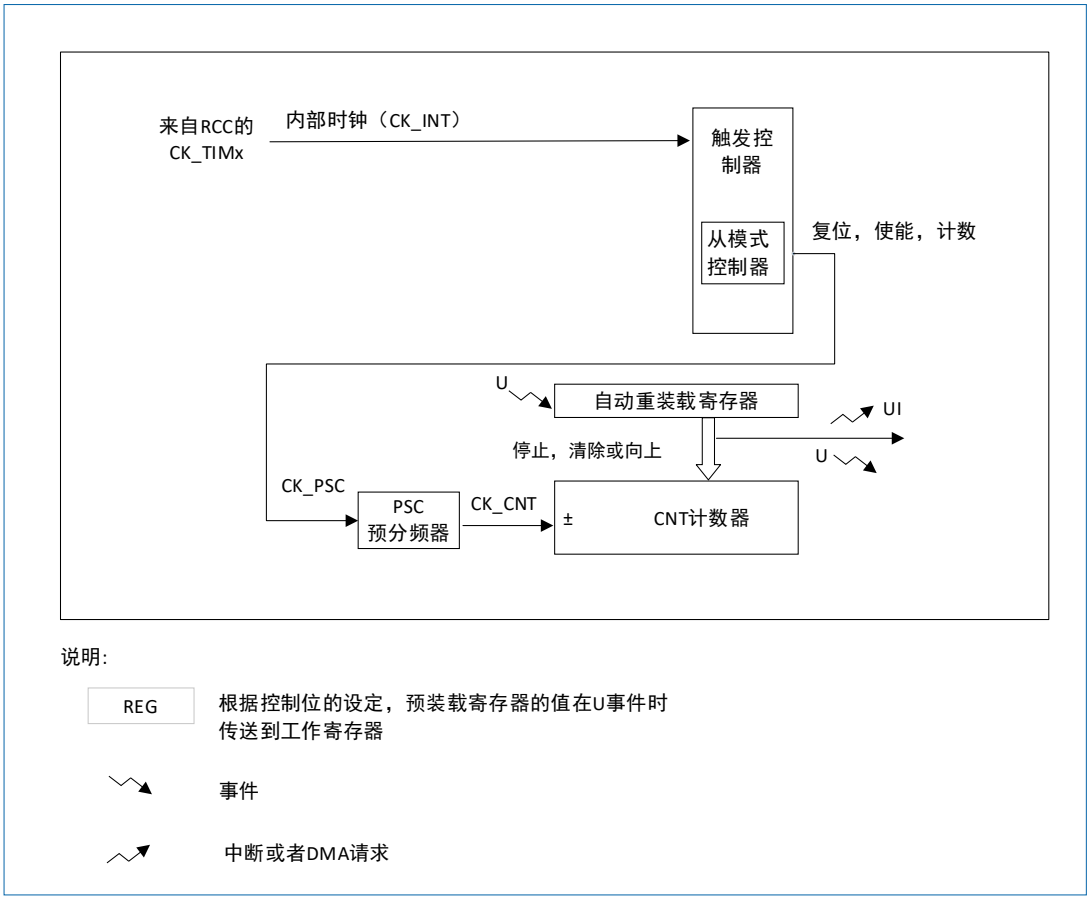


图 17-1 基本定时器框图

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（TIMx_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。

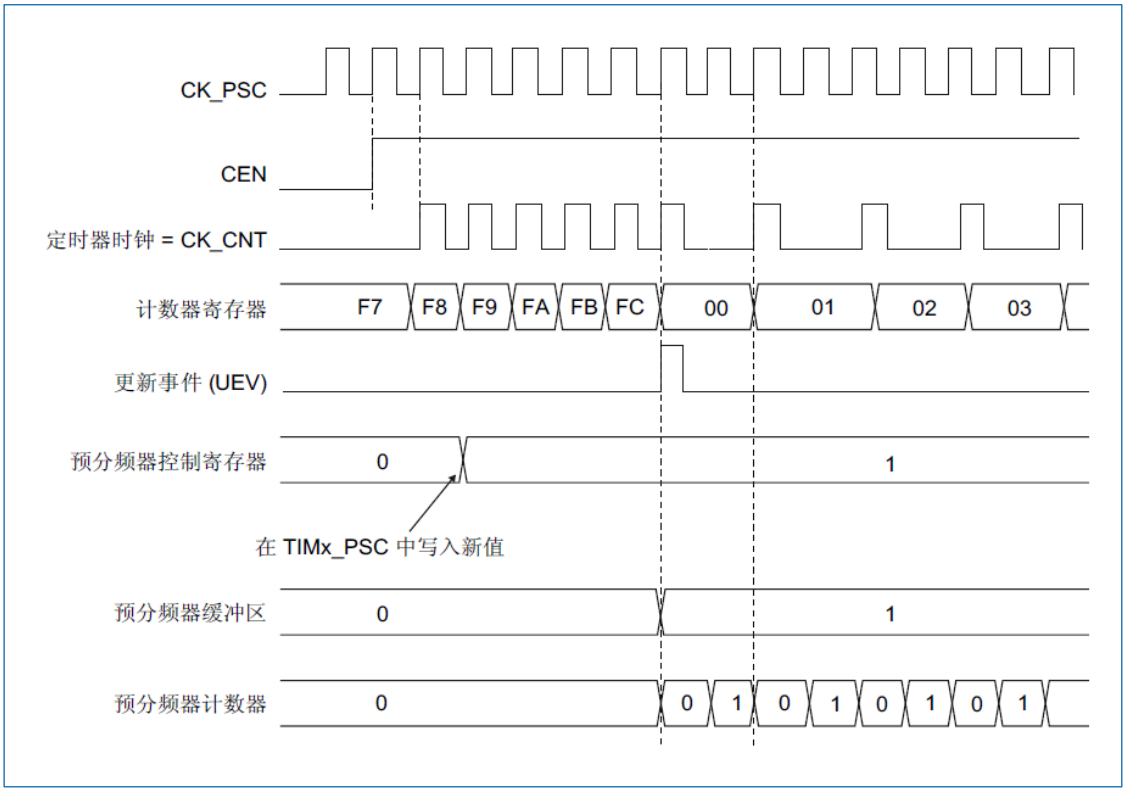


图 17-2 预分频系数从 1 变到 2 的计数器时序图

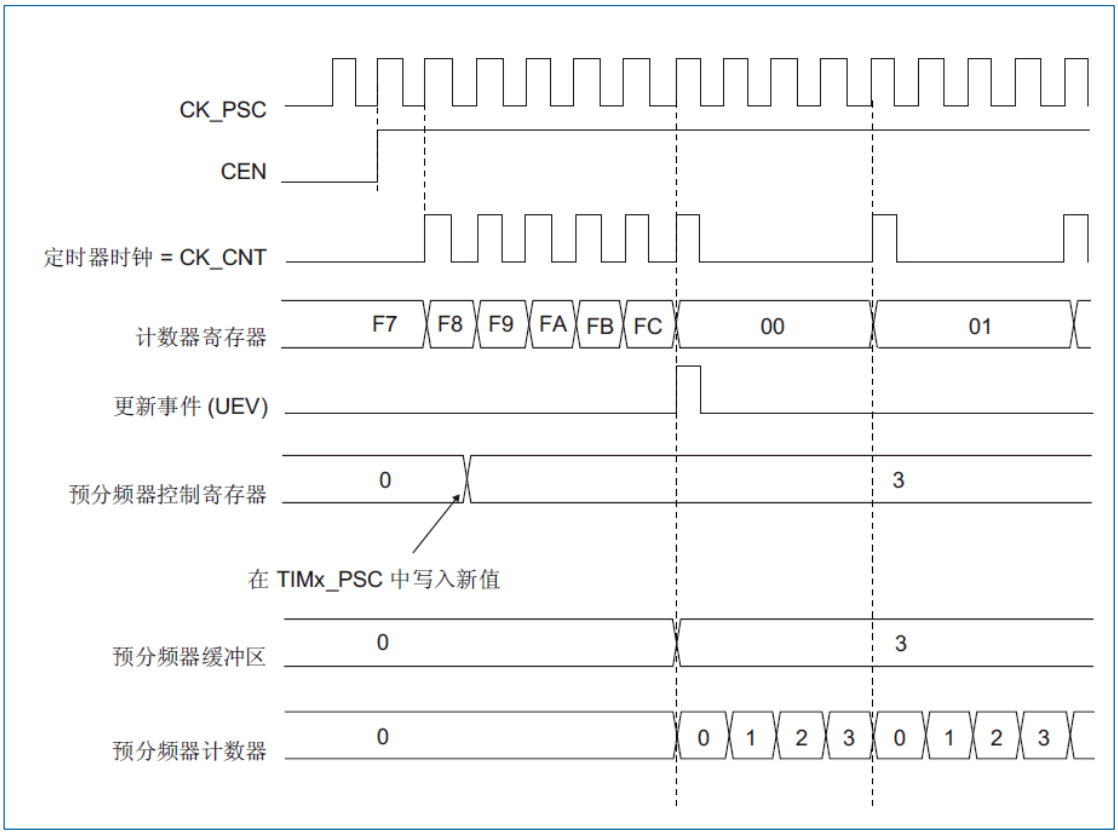


图 17-3 预分频系数从 1 变到 4 的计数器时序图

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。

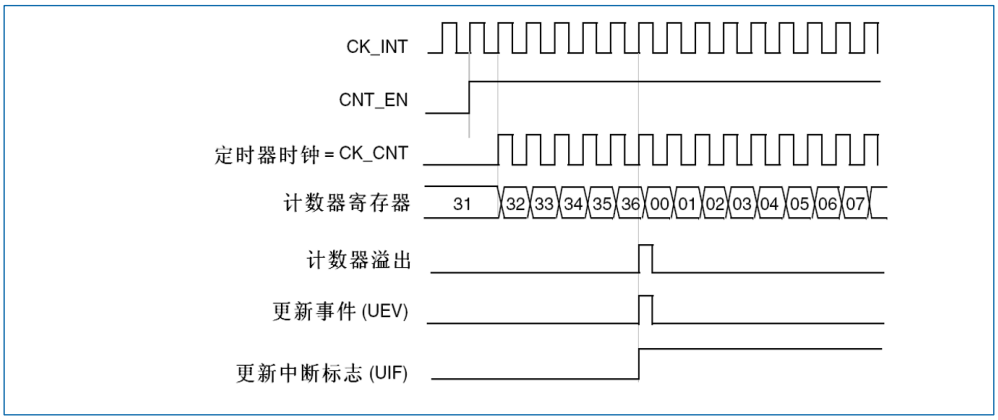


图 17-4 计数器时序图，内部时钟分频系数为 1

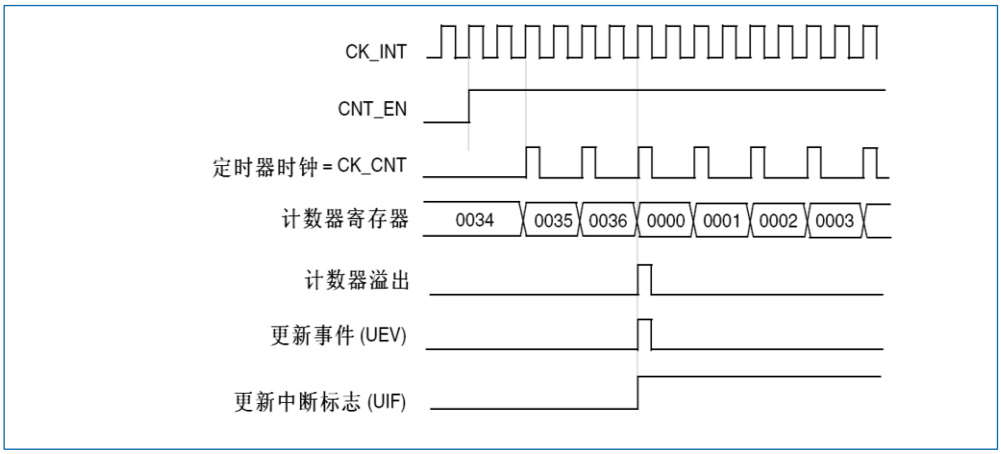


图 17-5 计数器时序图，内部时钟分频系数为 2

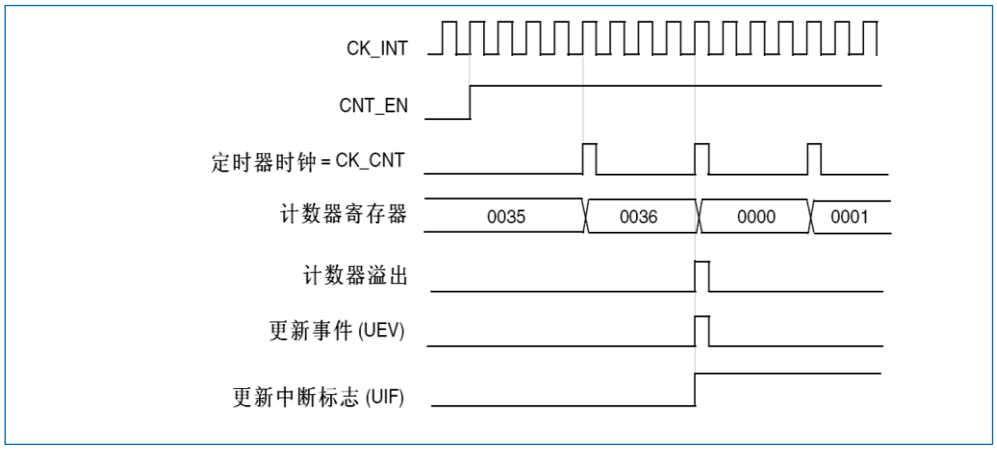


图 17-6 计数器时序图，内部时钟分频系数为 4

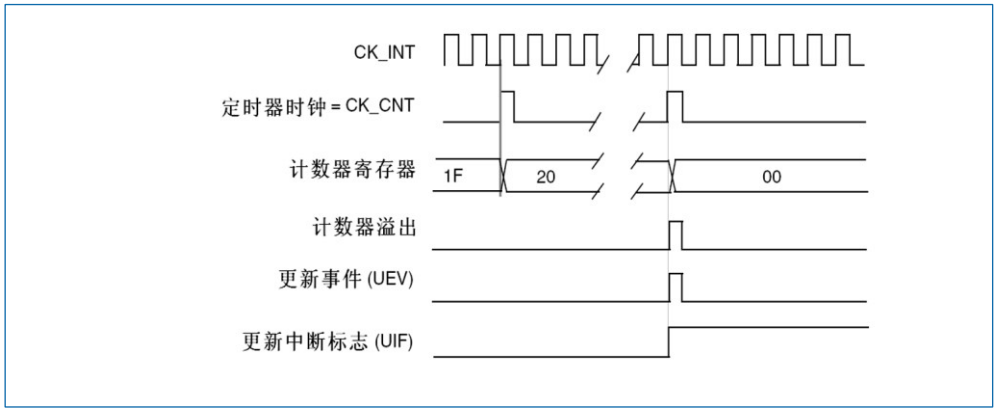


图 17-7 计数器时序图，内部时钟分频系数为 N

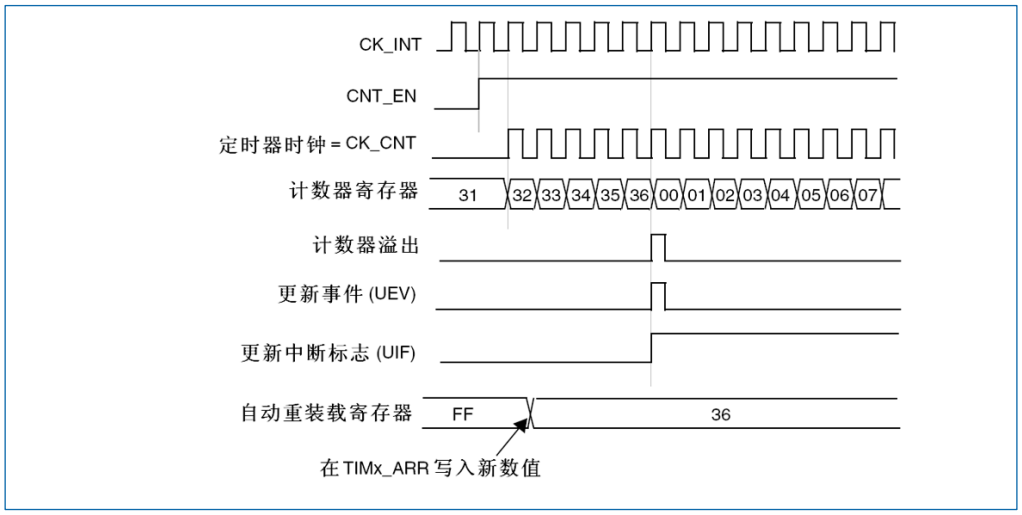


图 17-8 计数器时序图，当 ARPE=1 时的更新事件（无预装载 TIM6_ARR）

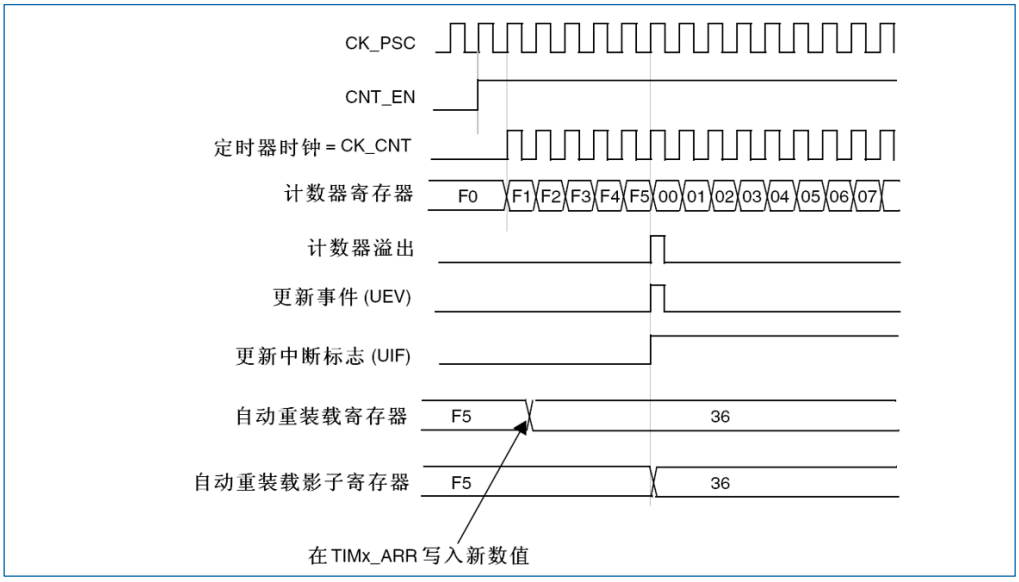


图 17-9 计数器时序图，当 ARPE=1 时的更新事件（预装载 TIM6_ARR）

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。

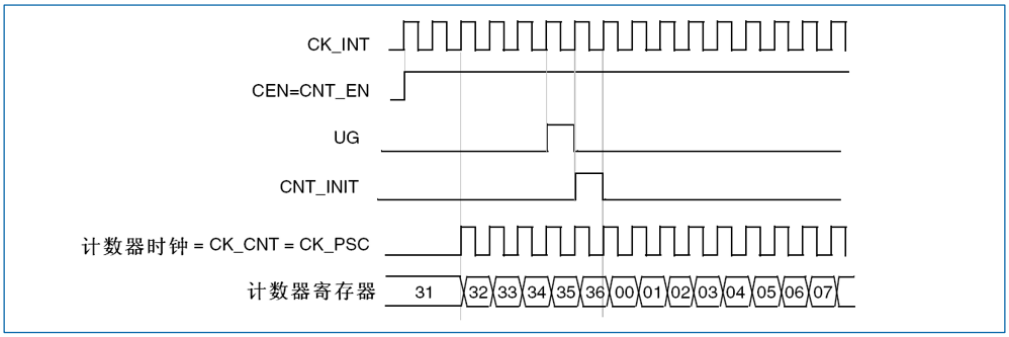


图 17-10 普通模式时序图，内部时钟分频系数为 1

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“26.8.2 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：
	  ○ 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

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/DMA 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							UDE	Res							UIE
							rw								rw

位 15:9	Res：保留

	必须保持复位值。
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 7:1	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’： o 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）。 o 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。 o 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）。若在中心对称模式下或 DIR=0（向上计数），则计数器被清’0’。

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 TIM_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 红外遥控接口（IRTIM）

设备上带红外遥控接口（IRTIM）。通过配合红外 LED 使用，可实现远程控制功能。它与 TIM16 进行内部连接，如图 18-1 所示。要产生红外遥控信号，必须使能红外接口和正确配置 TIM16 通道 1（TIM16_OC1）及 TIM17 通道 1（TIM17_OC1），以产生正确的波形。

通过配置 TIM16 或 TIM17 为基本的输入捕获模式，可以实现红外接收器的功能。

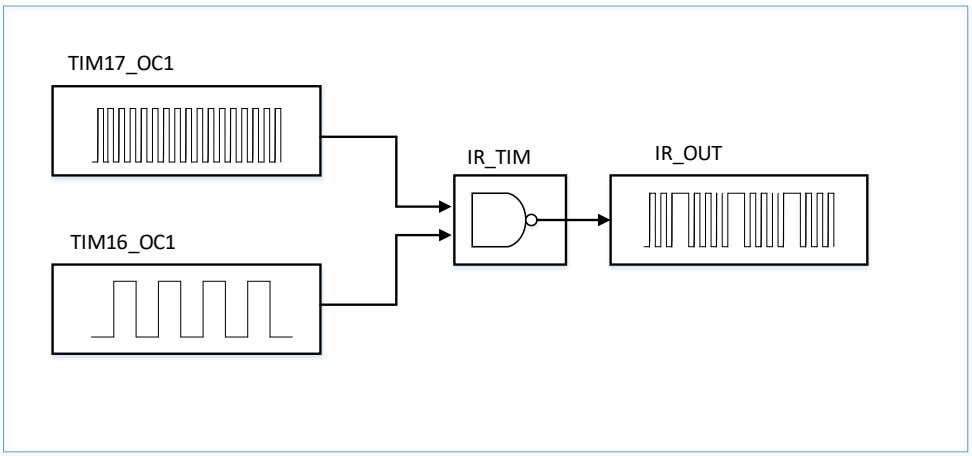


图 18-1 IR 内部硬件与 TIM16/17 连接

通过编程 TIM16 和 TIM17 两个定时器输出比较通道，可以设置红外脉冲调制模式。TIM17 用于产生高频载波信号，TIM16 产生基带信号。

红外功能在 IR_OUT 引脚上输出。通过 GPIOx_AFRx 寄存器启用相关的替代功能位，以激活这个功能。

LED 的驱动能力（仅在 PB9 引脚上可用）可以通过 SYSCFG_CFGR1 寄存器中的 I2C_PB9_FMP 位来激活。当 PB9 引脚控制一个红外 LED 时，该引脚用于吸收大电流。

19 独立看门狗（IWDG）

独立看门狗用于检测和解决由软件失效引起的故障，并在计数器达到指定的超时值时触发系统复位。独立看门狗由专用的低速时钟（LSI）驱动，即使主时钟发生故障它也仍然有效。独立看门狗由 APB1 时钟分频的时钟驱动，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 适用于看门狗能完全独立在主程序之外工作，并且对时间精度要求较低的场合。

19.1 IWDG 主要功能

- 自由运行的递减计数器
- 时钟由独立的 RC 振荡器提供（可在停机和待机模式下工作）
- 看门狗被激活后，则在计数器计数至 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，以确保在芯片复位后若程序跑飞，IWDG 复位的时间间隔不会太长。

19.2 IWDG 功能描述

在键寄存器（IWDG_KR）中写入 0xCCCC，开始启用独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号（IWDG_RESET）。

无论何时，只要在键寄存器 IWDG_KR 中写入 0xAAAA，IWDG_RLR 中的值就会被重新加载到计数器，从而避免产生看门狗复位。

支持看门狗窗口模式，详细的描述请参考“19.3.5 窗口寄存器（IWDG_WINR）”。

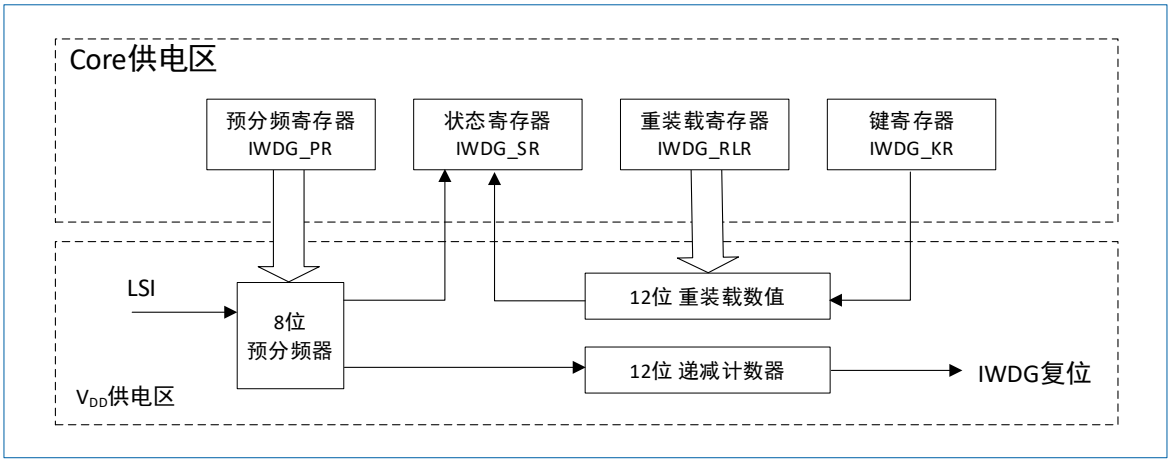


图 19-1 IWDG 框图

注意：看门狗功能处于 V_{DD} 供电区，即在停机和待机模式时仍能正常工作。

表 19-1 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

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
/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 40kHz 时钟给出。实际上，MCU 内部的 RC 频率会在 30kHz 到 60kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。*
- *通过对 LSI 进行校准可获得相对精确的看门狗超时时间。*

19.2.1 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。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。IWDG_WINR 的默认值为 0x0000FFFF，如果不更新默认值，将会禁用窗口选项。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。

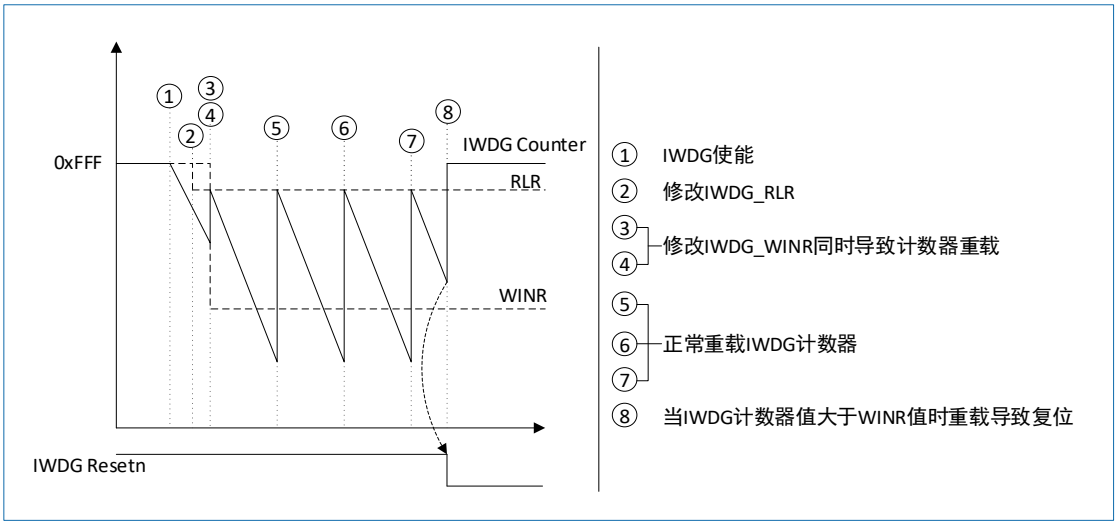


图 19-2 IWDG 使能窗口选项时的工作说明

使能窗口选项时，配置 IWDG 的流程如下：

1. 向 IWDG_KR 寄存器写入 0x0000CCCC 来使能 IWDG。
2. 向 IWDG_KR 寄存器写入 0x0000 5555 来使能寄存器访问。
3. 修改 IWDG_PR 寄存器的值为 0~7 中的值，以配置 IWDG 的预分频器。
4. 写重载寄存器 IWDG_RLR。
5. 等待 IWDG_RLR 寄存器值更新完成（IWDG_SR=0x0000 0000）。
6. 写窗口寄存器 IWDG_WINR。该操作会导致 IWDG 计数器自动更新为 IWDG_RLR 中的值。

注意：当 IWDG_SR 设置为 0x0000 0000 时，才能写 IWDT_WINR 寄存器，否则 IWDG 计数器可能不会正确的重载为 IWDG_RLR 中的值。

禁用窗口选项时，配置 IWDG 的流程如下：

1. 向 IWDG_KR 寄存器写入 0x0000CCCC 来使能 IWDG。
2. 向 IWDG_KR 寄存器写入 0x0000 5555 来使能寄存器访问。
3. 修改 IWDG_PR 寄存器的值为 0~7 中的值，以配置 IWDG 的预分频器。
4. 写重载寄存器 IWDG_RLR。
5. 等待 IWDG_RLR 寄存器值更新完成（IWDG_SR=0x0000 0000）。
6. 刷新计数器值为 IWDG_RLR 值（IWDG_KR=0x0000AAAA）。

19.2.2 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 IWDG_KR 寄存器写入 0x0000AAAA 以重载 IWDG 计数器，则系统会产生复位。

19.2.3 寄存器访问保护

IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器具有写保护功能。若需修改这三个寄存器的值，必须先向 IWDG_KR 寄存器中写入 0x0000 5555。以不同的值写入这个寄存器将会打乱操作顺序，寄存器将重新被保护。重装载操作（即写入 0x0000AAAA）也会启动写保护功能。

状态寄存器（IWDG_SR）指示预分频值和递减计数器是否正在被更新。

19.2.4 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_IWDG_STOP 配置位的状态，IWDG 的计数器能够继续或停止工作，参见“26 调试支持（DBG）”。

19.3 IWDG 寄存器

基地址：0x4000 3000

空间大小：0x400

19.3.1 关键字寄存器（IWDG_KR）

偏移地址：0x00

复位值：0x0000 0000

该寄存器在待机模式复位。

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
KEY[15:0]															
w															

位 31:16	Res：保留 必须保持复位值。
位 15:0	KEY[15:0]：键值（Key value） • 软件必须以一定的时间间隔写入 0xAAAA；否则，当计数器为 0 时，看门狗会产生复位。

	• 写入 0x5555 表示允许访问 IWDG_PR、IWDG_RLR 和 IWDG_WINR 寄存器，参见“19.2.3 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）。

19.3.2 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“19.2.3 寄存器访问保护”），通过设置该位域来选择计数器时钟的预分频因子。若需改变预分频因子，则 IWDG_SR 寄存器的 PVU 位必须为 0。 具体的值与预分频因子对应关系如下： • 000：预分频因子=4 • 001：预分频因子=8 • 010：预分频因子=16 • 011：预分频因子=32 • 100：预分频因子=64 • 101：预分频因子=128 • 110：预分频因子=256 • 111：预分频因子=256 *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的。因此，只有当 IWDG_SR 寄存器的 PVU 位为 0 时，读出的值才有效。*

19.3.3 重装载寄存器（IWDG_RLR）

偏移地址：0x08

复位值：0x00000FFF

该寄存器在待机模式复位。

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				RL[11:0]											
				rw											

位 31:12	Res：保留 必须保持复位值。
位 11:0	RL[11:0]：看门狗计数器重载值（Watchdog counter reload value） 该位域具有写保护设置（参见“19.2.3 寄存器访问保护”）。 RL 用于定义看门狗计数器的重装载值。每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。 看门狗的超时周期可以通过此重装载值和时钟预分频值来计算，可以参考表 19-1。 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的。因此，只有当 IWDG_SR 寄存器中的 RVU 位为 0 时。读出的值才有效。*

19.3.4 状态寄存器（IWDG_SR）

偏移地址：0x0C

复位值：0x0000 0000

该寄存器在待机模式复位。

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													WVU	RVU	PVU
													r	r	r

位 31:3	Res：保留 必须保持复位值。
位 2	WVU：看门狗计数器窗口值更新（Watchdog counter window value update） 该位由硬件置位，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 40kHz 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） 该位由硬件置 1，用于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，该位由硬件清’0’（最多需 5 个 40kHz 的 RC 周期）。重装载值只有在 RVU 位被清零后才可更新。
位 0	PVU：看门狗预分频值更新（Watchdog prescaler value update） 该位由硬件置 1，用于指示预分频值的更新正在进行中。 当在 V_{DD} 域中的预分频值更新结束后，该位由硬件清’0’（最多需 5 个 40kHz 的 RC 周期）。预分频值只有在 PVU 位被清零后才可更新。

说明：如果在应用程序中使用了多个重装载值或预分频值，则必须在 RVU 位被清除后才能重新改变预装载值，在 PVU 位被清除后才能重新改变预分频值。然而，在预分频和/或重装值更新后，不必等待 RVU 或 PVU 复位，可继续执行下面的代码。（即是在低功耗模式下，此写操作仍会被继续执行完成。）

19.3.5 窗口寄存器（IWDG_WINR）

偏移地址：0x10

复位值：0x0000 0FFF

该寄存器在待机模式复位。

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				WIN[11:0]											
				rw											

位 31:12	Res：保留 必须保持复位值。
位 1	WIN[11:0]：看门狗计数器窗口值（Watchdog counter window value）

20 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造成的应用程序背离正常的运行次序而产生的软件故障。除非递减计数器的值在 T6 位变成 0 前被刷新，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 位的递减计数器数值（在控制寄存器中）被刷新，那么也将产生一个 MCU 复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。

窗口看门狗时钟由 APB 时钟分频得到，它具有可配置的时间窗口。该窗口可编程，以检测异常过晚或过早的应用行为。

窗口看门狗适用于那些需要看门狗工作在准确的时间窗口下的应用。

20.1 WWDG 主要特性

- 可编程的自由运行递减计数器
- 条件复位
  - 当递减计数器的值小于 0x40，（若看门狗被启动）则产生复位。
  - 当递减计数器在窗口外被重新装载，（若看门狗被启动）则产生复位。
- 如果启动了看门狗并且允许中断，当递减计数器等于 0x40 时产生早期唤醒中断（Early wakeup interrupt，EWI），它可以用于重装载计数器以避免 WWDG 复位。

20.2 WWDG 功能描述

如果启动了看门狗（WWDG_CR 寄存器中的 WDGA 位被置'1'），并且当 7 位（T[6:0]）递减计数器从 0x40 翻转到 0x3F（T6 位清零）时，则产生一个复位信号。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。

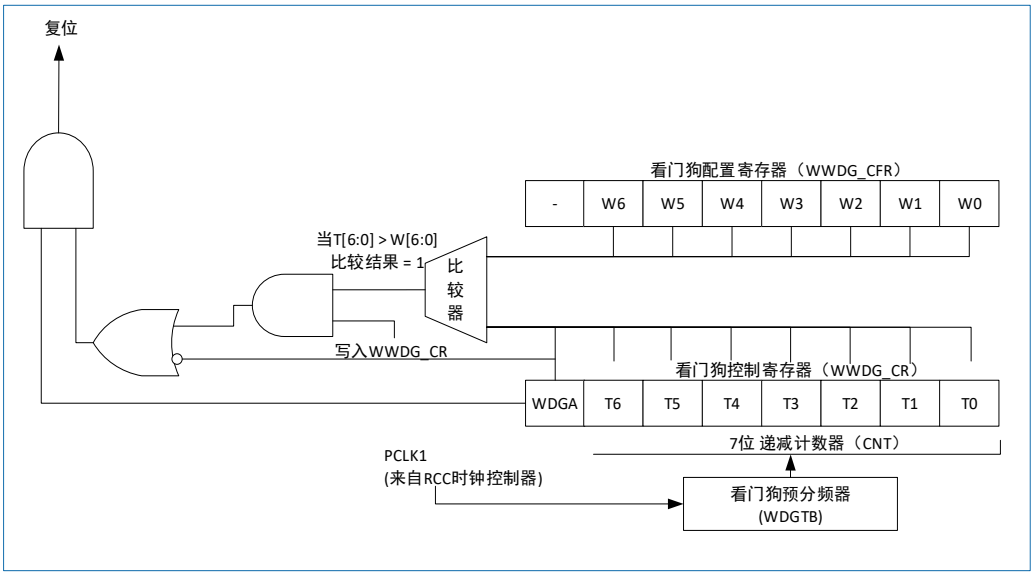


图 20-1 WWDG 框图

应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在 WWDG_CR 寄存器中的数值必须在 0xFF 和 0xC0 之间。

20.2.1 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。

20.2.2 控制递减计数器

递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时，T6 位必须被设置，以防止立即产生一个复位。

T[5:0]位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频值是未知的。

配置寄存器（WWDG_CFR）中包含窗口的上限值：要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载，图 20-2 描述了窗口寄存器的工作过程。另一个重装载计数器的方法是利用早期唤醒中断（EWI）。设置 WWDG_CFR 寄存器中的 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 产生一个软件复位（设置 WDGA 位为'1'，T 位为'0'）。

20.3 如何编写看门狗超时程序

可以使用下图中提供的公式计算 WWDG 的超时时间。

警告：当写入 WWDG_CR 寄存器时，请始终置 T6 位为'1'以避免立即产生一个复位。

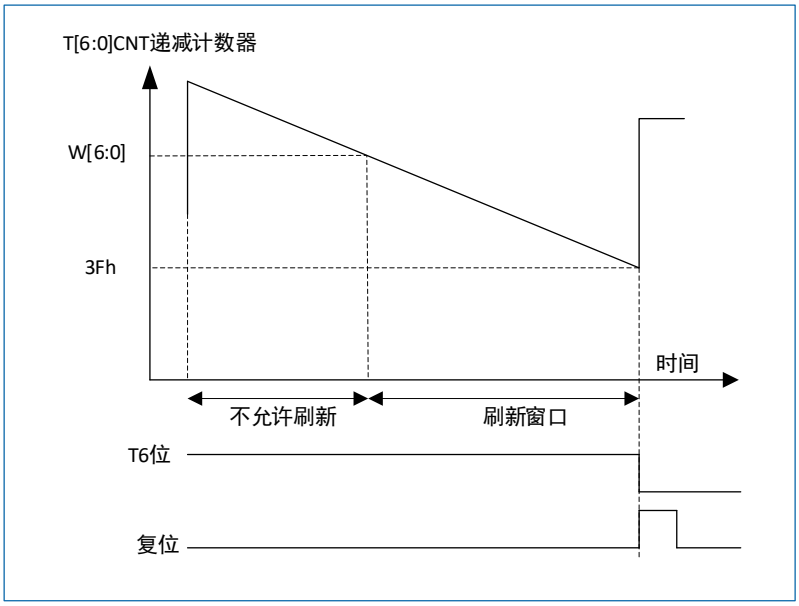


图 20-2 WWDG 时序图

如图 20-2 所示，计算超时的公式为：

$$T_{WWDG}=T_{PCLK1}x\ 4096x2^{WDGTB}\ x\ (T[5:0]+1)$$

其中：

- T_{WWDG} 为 WWDG 超时时间；
- T_{PCLK1}：以 ms 为单位的 APB1 时钟周期。

表 20-1 在 PCLK1=36MHz 时的最小/最大超时值

WDGTB	最小超时值	最大超时值
0	113 µs	7.28ms
1	227 µs	14.56ms

WDGTB	最小超时值	最大超时值
2	455 µs	29.12ms
3	910 µs	58.25ms

20.4 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。

20.5 WWDG 寄存器

基地址：0x4000 2C00

空间大小：0x400

20.5.1 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]						
								rs	rw						

位 31:8	Res：保留 必须保持复位值。
位 7	WDGA：激活位（Activation bit） 该位由软件置’1’，但仅在硬件复位后清’0’。当 WDGA=1 时，看门狗可以产生复位。 • 0：禁止看门狗 • 1：启用看门狗
位 6:0	T[6:0]：7 位计数器值（7-bit counter value） 该位域用来存储看门狗的计数器值。 每（4096 * 2WDGTB）个 PCLK1 周期减 1。当计数器值从 0x40 变为 0x3F 时（T6 变成 0），产生看门狗复位。

20.5.2 配置寄存器（WWDG_CFR）

偏移地址：0x04

复位值：0x0000 007F

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

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

位 31:10	Res：保留 必须保持复位值。
位 9	EWI：提前唤醒中断（Early wakeup interrupt） 该位若置’1’，则当计数器值达到 0x40 时即会产生中断。此中断只能由硬件在复位后清除。
位 8:7	WDGTB[1:0]：时基（Timer base） 预分频器的时基分频设置位 • 00：CK 计数器时钟（PCLK1 除以 4096）除以 1。 • 01：CK 计数器时钟（PCLK1 除以 4096）除以 2。 • 10：CK 计数器时钟（PCLK1 除以 4096）除以 4。 • 11：CK 计数器时钟（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。

20.5.3 状态寄存器（WWDG_SR）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res															EWIF
															rc_w0

位 31:1	Res：保留 必须保持复位值。
位 0	EWIF：提前唤醒中断标志（Early wakeup interrupt flag） 当计数器达到 0x40 时，该位由硬件置’1’，它必须通过软件写’0’来清除。对该位写’1’无效。若中断未被使能，该位也会被置’1’。

21 实时时钟（RTC）

实时时钟（RTC）提供用于管理所有低功耗模式的自动唤醒单元。

实时时钟（RTC）是一个独立的 BCD 定时器/计数器。RTC 提供具有可编程闹钟中断功能的日历时钟/日历。

RTC 还包含具有中断功能的周期性可编程唤醒标志。

两个 32 位寄存器包含二进码十进数格式（BCD）的秒、分钟、小时（12 或 24 小时制）、星期几、日期、月份和年份。此外，还可提供二进制格式的亚秒值。

系统可以自动将月份的天数补偿为 28、29（闰年）、30 和 31 天，并且还可以进行夏令时补偿。

其它 32 位寄存器还包含可编程的闹钟亚秒、秒、分钟、小时、星期几和日期。此外，还可以使用数字校准功能对晶振精度的偏差进行补偿。

RTC 域复位后，所有 RTC 寄存器都会受到保护，以防止可能的非正常写访问。

无论器件状态如何（运行模式、低功耗模式或处于复位状态），只要电源电压保持在工作范围内，RTC 便不会停止工作。

21.1 RTC 主要特性

RTC 单元的主要特性如下：

- 包含亚秒、秒、分钟、小时（12/24 小时制）、星期几、日期、月份和年份的日历。
- 软件可编程的夏令时补偿。
- 具有中断功能的可编程闹钟。可通过任意日历字段的组合触发闹钟。
- 自动唤醒单元，可周期性地生成标志以触发自动唤醒中断。
- 参考时钟检测：可使用更加精确的第二时钟源（50Hz 或 60Hz）来提高日历的精确度。
- 利用亚秒级移位特性与外部时钟实现精确同步。
- 数字校准电路（周期性计数器调整）：精度为 0.95ppm，在数秒钟的校准窗口中获得。
- 用于事件保存的时间戳功能
- 带可配置过滤器和内部上拉的侵入检测事件
- 可屏蔽中断/事件：
    - 闹钟 A
    - 唤醒中断
    - 时间戳
    - 侵入检测
- 5 个备份寄存器

21.2 RTC 功能说明

21.2.1 RTC 框图

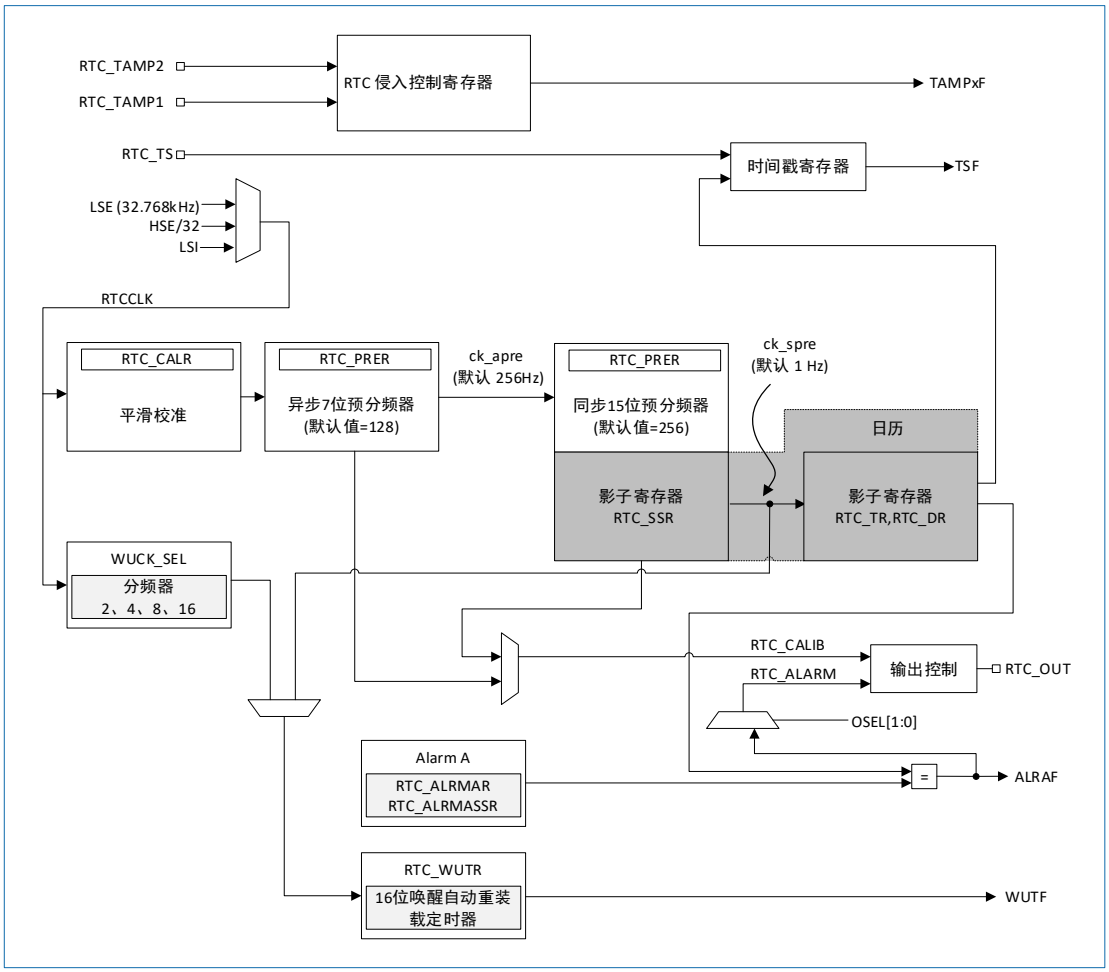


图 21-1 RTC 框图

21.2.2 RTC 控制的 GPIO

RTC_OUT、RTC_TS 和 RTC_TAMP1 映射到同一个引脚（PC13）。

通过 RTC_TAFCR 寄存器完成 RTC_ALARM 输出的选择，其中 PC13VALUE 位用于选择 RTC_ALARM 输出是推挽模式或开漏模式。

当 PC13 不用作 RTC 复用功能时，可通过设置 RTC_TAFCR 中 PC13MODE 位将 PC13 强制为推挽输出模式，输出值由 PC13VALUE 位给出。此时，PC13 的推挽输出状态和值在待机模式下是可以保持的。

输出机制遵循固定的优先顺序（见表 21-1）。

当 PC14 和 PC15 不用作 LSE 振荡器时，可通过分别设置 RTC_TAFCR 寄存器中 PC14MODE 位和 PC15MODE 位将 PC14 和 PC15 强制为推挽输出模式，输出值由 PC14VALUE 和 PC15VALUE 给出。此时，PC14 和 PC15 的推挽输出状态和值在待机模式下是可保持的。

输出机制遵循固定的优先顺序（见表 21-2 和表 21-3）。

表 21-1 RTC 引脚 PC13 配置

引脚配置和功能	RTC_ALARM 输出使能	RTC_CALIB 输出使能	RTC_TAMP1 输入使能	RTC_TS 输入使能	PC13MODE 位	PC13VALUE 位
RTC_ALARM 开漏输出	1	无影响	无影响	无影响	无影响	0
RTC_ALARM 推挽输出	1	无影响	无影响	无影响	无影响	1
RTC_CALIB 推挽输出	0	1	无影响	无影响	无影响	无影响
RTC_TAMP1 浮空输入	0	0	1	0	无影响	无影响
RTC_TS 和 RTC_TAMP1 浮空输入	0	0	1	1	无影响	无影响
RTC_TS 浮空输入	0	0	0	1	无影响	无影响
强制推挽输出	0	0	0	0	1	PC13 输出数据值

除此之外，若使能 RTC_OR 寄存器 ALRM_TYPE 位，PC13 引脚输出模式将由 RTC_OR 寄存器中某些配置位决定。

表 21-2 LSE 引脚 PC14 配置

引脚配置和功能	RCC_BDCR 寄存器 LSEON 位	RCC_BDCR 寄存器 LSEBYP 位	PC14MODE 位	PC14VALUE 位
LSE 晶振	1	0	无影响	无影响
LSE 旁路	1	1	无影响	无影响
强制推挽输出	0	无影响	1	PC14 输出数据值
标准 GPIO	0	无影响	0	无影响

表 21-3 LSE 引脚 PC15 配置

引脚配置和功能	RCC_BDCR 寄存器 LSEON 位	RCC_BDCR 寄存器 LSEBYP 位	PC15MODE 位	PC15VALUE 位
LSE 晶振	1	0	无影响	无影响
强制推挽输出	0	无影响	1	PC15 输出数据值
标准 GPIO	0	无影响	0	无影响

21.2.3 时钟和预分频器

由时钟控制器从以下 3 种时钟中选择 RTC 时钟源（RTCCLK）：

- LSE 振荡器作为 RTC 时钟；
- LSI 振荡器作为 RTC 时钟；
- HSE 振荡器 32 分频作为 RTC 时钟。

一个可编程预分频器产生一个 1Hz 的时钟，用于更新日历。为最大限度地减少功耗，预分频器可分割成 2 个可编程预分频器：

- 通过控制 RTC_PRER 寄存器中 PREDIV_A 位来配置 7 位异步预分频器。
- 通过控制 RTC_PRER 寄存器中 PREDIV_S 位来配置 15 位同步预分频器。

注：当所有分频器被启用时，建议将异步预分频器的值调高以最大限度地减少功率消耗。

异步预分频器的分频系数设为 128，同步预分频器的分频系数设为 256，从而得到一个基于 32.768kHz LSE 频率的 1Hz（ck_spre）内部时钟频率。

最小分频系数为 1，最大分频系数为 2^{22}，相当于最大输入频率大约 4MHz。

f_{ck_apre} 满足：

$$f_{ck\_apre}=\frac{f_{RTCCLK}}{PREDIV\_A+1}$$

ck_apre 时钟为二进制 RTC_SSR 亚秒递减计数器提供时钟。当值递减为 0 时，RTC_SSR 将被重置为 PREDIV_S 里的值。

f_{ck_spre} 满足：

$$f_{ck\_spre}=\frac{f_{RTCCLK}}{（PREDIV\_S+1）\times（PREDIV\_A+1）}$$

21.2.4 实时时钟和日历

RTC 的日历时间寄存器和日期寄存器是通过影子寄存器访问的。该影子寄存器与 PCLK（APB 时钟）同步。为避免同步等待时间，也可直接访问。

- RTC_SSR：亚秒
- RTC_TR：时间
- RTC_DR：日期

硬件会每两个 RTCCLK 时钟周期更新一次影子寄存器的日历值，并将 RTC_ISR 寄存器的 RSF 位置 1。停机或待机模式下则不做这个更新。当退出上述两种模式后，影子寄存器将在最多两个 RTCCLK 周期后执行更新操作。

默认情况下，应用程序通过访问影子寄存器获取日历寄存器的内容。如需直接访问日历寄存器，可通过设置 RTC_CR 寄存器的 BYPSHAD 控制位（默认为零）来实现。

在 BYPSHAD=0 模式下，如需读取 RTC_SSR、RTC_TR 或 RTC_DR 寄存器的内容，APB 时钟频率（f_{APB}）至少必须是 RTC 时钟频率（f_{RTCCLK}）的 7 倍。

系统复位后，影子寄存器也将自动复位。

21.2.5 可编程闹钟

RTC 单元提供一个可编程闹钟，即闹钟 A。以下说明针对闹钟 A。

可通过 RTC_CR 寄存器中的 ALRAE 位来使能可编程闹钟功能。如果日历亚秒、秒、分钟、小时、日期或日与闹钟寄存器 RTC_ALRMASSR 和 RTC_ALRMAR 中编程的值相匹配，则 ALRAF 标志会被置为 1。可通过 RTC_ALRMAR 寄存器的 MSKx 位以及 RTC_ALRMASSR 寄存器的 MASKSSx 位单独选择各日历字段。可通过 RTC_CR 寄存器中的 ALRAIE 位来使能闹钟中断。

注意：如果选择秒字段（RTC_ALRMAR 中的 MSK1 位复位），则 RTC_PRER 寄存器中设置的同步预分频器分频系数必须至少为 3，才能确保闹钟正确地运行。

闹钟 A（如果已通过 RTC_CR 寄存器中的位 OSEL[0:1]使能）可连接到 RTC_ALARM 输出。可通过 RTC_CR 寄存器的 POL 位配置 RTC_ALARM 输出极性。

21.2.6 周期性自动唤醒

周期性唤醒标志由 16 位可编程自动重载递减计数器生成。唤醒定时器范围可扩展至 17 位。

可通过 RTC_CR 寄存器中的 WUTE 位来使能此唤醒功能。

唤醒定时器的时钟输入可以是：

- 2、4、8 或 16 分频的 RTC 时钟（RTCCLK）。

  当 RTCCLK 为 LSE（32.768kHz）时，可配置的唤醒中断周期介于 122 µs 和 32s 之间，且分辨率低至 61 µs。

- ck_spre 通常为 1Hz 内部时钟。

  当 ck_spre 频率为 1Hz 时，可得到的唤醒时间为 1s 到 36h 左右，分辨率为 1s。这一较大的可编程时间范围分为两部分：

  - WUCKSEL[2:1]=10 时为 1s 到 18h。
  - WUCKSEL[2:1]=11 时约为 18h 到 36h。在后一种情况下，会将 2^{16} 添加到 16 位计数器当前值。完成初始化序列后，定时器开始递减计数。在低功耗模式下使能唤醒功能时，递减计数保持有效。此外，当计数器计数到 0 时，RTC_ISR 寄存器的 WUTF 标志会置 1，并且唤醒寄存器会使用其重载值（RTC_WUTR 寄存器值）动重载。

之后必须用软件清零 WUTF 标志。

通过将 RTC_CR2 寄存器中的 WUTIE 位置 1 来使能周期性唤醒中断时，它会使器件退出低功耗模式。

如果已通过 RTC_CR 寄存器的位 OSEL[1:0]使能周期性唤醒标志，则该标志可连接到 RTC_ALARM 输出。可通过 RTC_CR 寄存器的 POL 位配置 RTC_ALARM 输出极性。系统复位以及低功耗模式（睡眠、停机和待机）对唤醒定时器没有任何影响。

通过对 RTC_OR 寄存器中的 WUTO[3:0]的配置可使能唤醒定时器唤醒 ADC 功能。

21.2.7 RTC 初始化和配置

RTC 寄存器访问

RTC 寄存器为 32 位寄存器。除了当 BYPSHAD=0 时对日历影子寄存器执行的读访问之外，APB 接口会在访问 RTC 寄存器时引入 2 个等待周期。

RTC 寄存器写保护

系统复位后，可通过对 PWR_CR 寄存器中的 DBP 位清零来保护 RTC 寄存器以防止非正常的写访问。必须将 DBP 位置 1 才能使能 RTC 寄存器的写访问。

RTC 域复位后，所有 RTC 寄存器均受到写保护。通过向写保护寄存器（RTC_WPR）写入一个密钥来使能对 RTC 寄存器的写操作。

要解锁所有 RTC 寄存器（RTC_TAMPCR、RTC_BKPxR、RTC_OR 和 RTC_ISR[13:8]除外）的写保护，需要执行以下步骤：

1. 将“0xCA”写入 RTC_WPR 寄存器。
2. 将“0x53”写入 RTC_WPR 寄存器。

写入一个错误的关键字会再次激活写保护。

保护机制不受系统复位影响。

日历初始化和配置

若编程包括时间格式和预分频器配置在内的初始时间和日期日历值，操作步骤如下：

1. 将 RTC_ISR 寄存器中的 INIT 位置 1 以进入初始化模式。在此模式下，日历计数器将停止工作并且其值可更新。

2. 轮询 RTC_ISR 寄存器中的 INITF 位。当 INITF 置 1 时，进入初始化阶段模式。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。

3. 要为日历计数器生成 1Hz 时钟，应编程 RTC_PRER 寄存器中的两个预分频系数。

4. 在影子寄存器（RTC_TR 和 RTC_DR）中加载初始时间和日期值，然后通过 RTC_CR 寄存器中的 FMT 位配置时间格式（12 或 24 小时制）。

5. 通过清零 INIT 位退出初始化模式。随后，自动加载实际日历计数器值，在 4 个 RTCCLK 时钟周期后重新开始计数。

   当初始化序列完成之后，日历开始计数。

说明：

1. *系统复位后，应用可读取 RTC_ISR 寄存器中的 INITS 标志，以检查日历是否已初始化。如果该标志为 0，表明日历尚未初始化，因为年份字段设置为其 RTC 域复位时的默认值（0x00）。*

2. *要在初始化之后读取日历，必须首先用软件检查 RTC_ISR 寄存器的 RSF 标志是否置 1。*

夏令时

可通过 RTC_CR 寄存器的 SUB1H、ADD1H 和 BKP 位管理夏令时。

利用 SUB1H 或 ADD1H，软件只需单次操作便可在日历中减去或增加一个小时，无需执行整个初始化步骤。

此外，软件还可以使用 BKP 位来记录是否曾经执行过此操作。

编程闹钟

要对可编程的闹钟进行编程或更新，必须执行类似的步骤。以下步骤针对闹钟 A。

1. 将 RTC_CR 中的 ALRAE 位清零以禁用闹钟 A。

2. 编程闹钟 A 寄存器（RTC_ALRMASSR/RTC_ALRMAR）。

3. 将 RTC_CR 寄存器中的 ALRAE 位置 1 以再次使能闹钟 A。

说明：约 2 个 RTCCLK 时钟周期（由于时钟同步）后，将执行对 RTC_CR 寄存器的更改。

编程唤醒定时器

要配置或更改唤醒定时器的自动重载值（RTC_WUTR 中的 WUT[15:0]），操作步骤如下：

1. 清零 RTC_CR 中的 WUTE 以禁用唤醒定时器。

2. 轮询 RTC_ISR 中的 WUTWF，直到该位置 1，以确保可以访问唤醒自动重载计数器和 WUCKSEL[2:0]位。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。

3. 编程唤醒自动重载值 WUT[15:0]，并选择唤醒时钟（RTC_CR 中的 WUCKSEL[2:0]位）。将 RTC_CR 寄存器中的 WUTE 位置 1 以再次使能定时器。唤醒定时器重新开始递减计数。由于时钟同步，WUTWF 位会在 WUTE 清零达 2 个 RTCCLK 时钟周期后清零。

21.2.8 读取日历

当 RTC_CR 寄存器中的 BYPSHAD 控制位清零时

要正确读取 RTC 日历寄存器（RTC_SSR、RTC_TR 和 RTC_DR），APB1 时钟频率（f_{PCLK}）必须等于或大于 RTC 时钟频率（f_{RTCCLK}）的七倍。这可以确保同步机制行为的安全性。

如果 APB1 时钟频率低于 RTC 时钟频率的七倍，则软件必须读取日历时间寄存器和日期寄存器两次。这样，当两次读取的 RTC_TR 结果相同时，才能确保数据正确。否则必须执行第三次读访问。任何情况下，APB1 的时钟频率都不能低于 RTC 的时钟频率。

每次将日历寄存器中的值复制到 RTC_SSR、RTC_TR 和 RTC_DR 影子寄存器时，RTC_ISR 寄存器中的 RSF 位都会置 1。每两个 RTCCLK 周期执行一次复制。为确保这 3 个值一致，读取 RTC_SSR 或 RTC_TR 时会锁定高阶日历影子寄存器中的值，直到读取 RTC_DR。为避免软件对日历执行读访问的时间间隔小于 2 个 RTCCLK 周期：第一次读取日历之后必须通过软件将 RSF 清零，并且软件必须等待到 RSF 置 1 之后才可再次读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

从低功耗模式（停机模式或待机模式）唤醒之后，必须通过软件将 RSF 清零。之后，软件必须等待至 RSF 再次置 1 后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

RSF 位必须在唤醒之后，而不是进入低功耗模式之前进行清零。

系统复位之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。实际上，系统复位会将影子寄存器复位为其默认值。

初始化之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

同步之后，软件必须等待至 RSF 置 1 之后才可以读取 RTC_SSR、RTC_TR 和 RTC_DR 寄存器。

当 RTC_CR 寄存器中的 BYPSHAD 控制位置 1 时（旁路影子寄存器）

读取日历寄存器时会直接从日历计数器获取值，这样便无需等待至 RSF 位置 1。这对于从低功耗模式（停机模式或待机模式）退出后的情况特别有用，因为影子寄存器在这些模式下不更新。

当 BYPSHAD 位置 1 时，如果在对寄存器的两次读访问之间出现 RTCCLK 沿，则不同寄存器的结果彼此可能不一致。此外，如果在读操作期间出现 RTCCLK 沿，则可能导致其中一个寄存器的值不正确。软件必须分两次读取所有寄存器，然后将两次结果加以比较来确认数据是否一致和正确。此外，软件也可以只比较两次读取日历寄存器得到的结果的最低位。

说明：当 BYPSHAD=1 时，读取日历寄存器的指令需要一个额外的 APB 周期才能完成。

21.2.9 复位 RTC

日历影子寄存器（RTC_SSR、RTC_TR 和 RTC_DR）以及 RTC 状态寄存器（RTC_ISR）的某些位通过所有可用的系统复位源复位为各自的默认值。

相反，以下寄存器则通过 RTC 域复位来复位为各自的默认值并且不受系统复位的影响：RTC 当前日历寄存器、RTC 控制寄存器（RTC_CR）、预分频器寄存器（RTC_PRER）、RTC 校准寄存器（RTC_CALR）、RTC 移位寄存器（RTC_SHIFTR）、RTC 时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）、RTC 侵入和复用功能配置寄存器（RTC_TAFCR）、RTC 备份寄存器（RTC_BKPxR）、唤醒定时器寄存器（RTC_WUTR）、闹钟 A 寄存器（RTC_ALRMASSR/RTC_ALRMAR）以及选项寄存器（RTC_OR）。

此外，当由 LSE 提供时钟时，如果复位源并非 RTC 域复位源（有关不受系统复位影响的 RTC 时钟源列表的详细信息，请参见“6.2.10 RTC 时钟”），则 RTC 将在系统复位时保持运行状态。发生 RTC 域复位时，RTC 会停止工作，并且所有 RTC 寄存器都会设置为各自的复位值。

21.2.10 RTC 同步

RTC 可与高精度的远程时钟同步。在读取亚秒字段后（RTC_SSR 或 RTC_TSSSR），即可计算远程时钟的时间与 RTC 之间的精准偏差。之后，可使用 RTC_SHIFTR 对 RTC 的时钟进行零点几秒的“平移”，经过调整后可消除此偏差。

RTC_SSR 包含同步预分频器计数器的值。这样，便可计算分辨率低至 1/（PREDIV_S+1）秒的 RTC 的准确时间。因此，可通过增大同步预分频器的值（PREDIV_S[14:0]）来提高分辨率。将 PREDIV_S 设置为 0

x7FFF 时，可得到允许的最大分辨率（30.52 μs，时钟频率为 32768Hz）。

但是，提高 PREDIV_S 意味着必须降低 PREDIV_A 才能将同步预分频器的输出维持在 1Hz。这样，异步预分频器的输出频率会增大，RTC 的动态功耗也会相应增加。

可以使用 RTC 平移控制寄存器（RTC_SHIFTR）对 RTC 进行微调。可以用大小为 1/（PREDIV_S+1）秒的分辨率对 RTC_SHIFTR 进行写操作，将时钟平移（延迟或提前）最长 1 秒。在这种平移操作中，会将 SUBFS[14:0]值加到同步预分频器计数器 SS[15:0]中：这将使时钟产生延迟。如果同时将 ADD1S 位置 1，则会增加一秒，与此同时减去的时间为零点几秒，因此将使时钟提前。

说明：初始化平移操作前，用户必须检查确认 SS[15]=0，以确保不会发生上溢。

对 RTC_SHIFTR 寄存器执行写操作以启动平移操作时，硬件会将 SHPF 标志置 1 以指示平移操作挂起。完成平移操作时，硬件会将将该位清零。

说明：该同步功能与参考时钟检测功能不兼容：当 REFCKON=1 时，固件不能对 RTC_SHIFTR 执行写操作。

21.2.11 RTC 参考时钟检测

RTC 日历更新可与参考时钟 RTC_REFIN（通常为市电频率，50Hz 或 60Hz）同步。

RTC_REFIN 参考时钟的精度应高于 32.768kHz LSE 时钟。使能 RTC_REFIN 检测时（将 RTC_CR 的 REFCKON 位置 1），日历仍由 LSE 提供时钟，而 RTC_REFIN 用于补偿不准确的日历更新频率（1Hz）。

每个 1Hz 时钟边沿都与最近的 RTC_REFIN 时钟边沿进行比较（如果在给定的时间窗口内发现一个边沿）。在大多数情况下，两个时钟边沿恰好对齐。当 1Hz 时钟由于 LSE 时钟不精确而发生偏离时，RTC 会稍微偏移 1Hz 时钟，以便后续的 1Hz 时钟边沿能够对齐。利用这种机制，可使日历像参考时钟一样精确。

RTC 使用 32.768kHz 石英产生的 256Hz 时钟（ck_apre）来检测是否存在参考时钟源。大约在日历每次更新时（每 1 秒钟），便会在时间窗口期间执行一次检测。检测到第一个参考时钟边沿时，该窗口等于 7 个 ck_apre 周期。随后的日历更新使用长度为 3 个 ck_apre 周期的较小窗口。

每次在窗口中检测到参考时钟时，都会行强制输出 ck_apre 时钟的异步预分频器进行重载。

当参考时钟与 1Hz 时钟对齐时，此操作不起作用，因为预分频器会在同一时刻重载。当时钟不对齐时，重载操作会微调后续的 1Hz 时钟边沿，使其与参考时钟对齐。

如果参考时钟停止（在 3 个 ck_apre 窗口内未出现参考时钟边沿），日历将仅根据 LSE 时钟进行连续更新。RTC 随后使用 ck_spre 边沿上居中的大检测窗口（7 个 ck_apre 周期）等待参考时钟。

使能 RTC_REFIN 检测后，必须将 PREDIV_A 和 PREDIV_S 设置为各自的默认值：

- PREDIV_A=0x007F
- PREVID_S=0x00FF

说明：RTC_REFIN 时钟检测在待机模式下不可用。

21.2.12 RTC 精密数字校准

RTC 频率可采用约 0.954ppm 的分辨率进行数字校准，校准范围为-487.1ppm 到+488.5ppm。使用一系列微调（增加和/或减少单独的 RTCCLK 脉冲）进行频率校正。这些微调的分布非常均匀，因此 RTC 的校准效果相当好，即使在短时间内持续观察也是如此。

当输入频率为 32768Hz 时，精密数字校准的周期约为 2^{20} 个 RTCCLK 脉冲或 32 秒。此周期由一个通过 RTCCLK 提供时钟信号的 20 位计数器 cal_cnt[19:0]维持。

精密数字校准寄存器（RTC_CALR）可指定 32 秒周期内要减少的 RTCCLK 时钟周期数：

- 将 CALM[0]置 1 时，32 秒周期内将只减少 1 个脉冲。

- 将 CALM[1]置 1 时，将减少 2 个周期。
- 将 CALM[2]置 1 时，将减少 4 个周期。
- 依此类推，将 CALM[8]置 1 时，将减少 256 个时钟。

说明：CALM[8:0]（RTC_CALR）可指定 32 秒周期内要减少的 RTCCLK 脉冲数。将位 CALM[0]置 1 时，32 秒周期内将只减少 1 个脉冲（当 cal_cnt[19:0]=0x80000 时）；将 CALM[1]置 1 时，将减少 2 个周期（当 cal_cnt=0x40000 和 0xC0000 时）；将 CALM[2] 置 1 时，将减少 4 个周期（当 cal_cnt=0x20000/0x60000/0xA0000/0xE0000 时）；依此类推，将 CALM[8]置 1 时，将减少 256 个时钟（当 cal_cnt=0xXX800 时）。

使用适当分辨率时，CALM 可使 RTC 频率减少最多 487.1ppm，而 CALP 可用于使频率增加 488.5ppm。将 CALP 置“1”，可每隔 2^{11} 个 RTCCLK 周期有效插入一个额外的 RTCCLK 脉冲，这意味着每 32 秒周期可增加 512 个时钟。

与 CALM 和 CALP 配合使用时，可在 32 秒周期内增加一个范围为-511 到+512RTCCLK 周期的偏差，对应的校准范围为-487.1ppm 到+488.5ppm，分辨率约为 0.954ppm。

若输入频率（FRTCCLK）已知，可通过以下公式计算有效校准频率（FCAL）：

FCAL =FRTCCLK * [1 +（CALP*512-CALM）/（220+CALM-CALP*512）]

PREDIV_A<3 条件下的校准

当异步预分频器值（RTC_PRER 寄存器中的 PREDIV_A 位）小于 3 时，不能将 CALP 位置 1。如果 CALP 已置 1 并且 PREDIV_A 位的值小于 3，则会忽略 CALP，即假定 CALP 等于 0 而执行校准。

要在 PREDIV_A 小于 3 的条件下执行校准，应降低同步预分频器值（PREDIV_S）以便每秒内可加速 8 个 RTCCLK 时钟周期，这意味着每 32 秒可增加 256 个时钟周期。因此，仅使用 CALM 位，可在每 32 秒内有效增加 255 到 256 个时钟脉冲（对应的校准范围为 243.3ppm 到 244.1ppm）。

在标称 RTCCLK 频率 32768Hz 下，当 PREDIV_A 等于 1 时（分频系数为 2），应将 PREDIV_S 设置为 16379 而不是 16383（少 4）。唯一相关的其它情况是，当 PREDIV_A 等于 0 时，应将 PREDIV_S 设置为 32759 而不是 32767（少 8）。

如果以这种方式减少 PREDIV_S，则采用以下公式计算校准输入时钟的有效频率：

FCAL=FRTCCLK x [1+（256-CALM）/（2^{20}+CALM-256）]

在这种情况下，如果 RTCCLK 恰好为 32768.00Hz，则当 CALM[7:0]等于 0x100 时（CALM 范围的中值），说明设置正确。

验证 RTC 校准

通过测量 RTCCLK 的精确频率，计算正确的 CALM 和 CALP 值以确保 RTC 精度。此外，还为应用提供了一个可选的 1Hz 输出，用来测量和验证 RTC 精度。

如果在有限的间隔内测量 RTC 的精确频率，则会导致测量期间产生最多 2 个 RTCCLK 时钟周期的测量误差，具体取决于数字校准周期与测量周期的对齐方式。

但是，如果测量周期与校准周期的长度相同，则可以消除此测量误差。在这种情况下，观测到的唯一误差是由数字校准的分辨率导致的误差。

- 默认情况下，校准周期为 32 秒。

  在此模式下，测量整个 32 秒内 1Hz 输出的精度，可确保测量误差在 0.477ppm 内（32 秒内为 0.5 个 RTCCLK 周期，受校准分辨率限制）。

- 可将 RTC_CALR 寄存器的 CALW16 位置 1，以强制 16 秒的校准周期。

  此时，可在 16 秒内测量 RTC 精度，产生的最大误差为 0.954ppm（16 秒内为 0.5 个 RTCCLK 周

期）。但是，由于校准分辨率降低，长期的 RTC 精度也会降到 0.954ppm：将 CALW16 置 1 时，CALM[0]位将始终保持为 0。

- 可将 RTC_CALR 寄存器的 CALW8 位置 1，以强制 8 秒的校准周期。

  此时，可在 8 秒内测量 RTC 精度，产生的最大误差为 1.907ppm（8 秒内为 0.5 个 RTCCLK 周期）。长期的 RTC 精度也会降到 1.907ppm：将 CALW8 置 1 时，CALM[1:0]位将始终保持为 00。

动态重校准

当 RTC_ISR/INITF=0 时，可动态更新校准寄存器（RTC_CALR），具体步骤如下：

1. 轮询 RTC_ISR/RECALPF（重新校准挂起标志）。
2. 如果该标志为 0，则可以根据需要向 RTC_CALR 写入新值。随后 RECALPF 位会被自动置为 1。
3. 新校准设置将在对 RTC_CALR 执行写操作之后的三个 ck_apre 周期内生效。

21.2.13 时间戳功能

将 RTC_CR 寄存器的 TSE 位置 1 可使能时间戳。

当在 RTC_TS 引脚上检测到时间戳事件时，日历会保存到时间戳寄存器（RTC_TSSSR、RTC_TSTR 和 RTC_TSDR）中。发生时间戳事件时，RTC_ISR 寄存器中的时间戳标志位（TSF）将置 1。

通过将 RTC_CR 寄存器中的 TSIE 位置 1，可在发生侵入检测事件时生成中断。

如果在时间戳标志（TSF）已置 1 的条件下检测到新的时间戳事件，则时间戳上溢标志（TSOVF）将置 1，而时间戳寄存器（RTC_TSTR 和 RTC_TSDR）将保持上一事件的结果。

说明：在发生由同步过程引发的时间戳事件后，TSF 在 2 个 ck_apre 周期置为 1。

TSOVF 的产生中不存在延迟。也就是说如果两个时间戳事件接连发生，TSOVF 可能为“1”而 TSF 为“0”。因此，建议只在检测到 TSF 为“1”后再轮询 TSOVF。

注意：如果在 TSF 位清零后紧接着发生时间戳事件，则 TSF 和 TSOVF 位都将置 1。为防止在时间戳事件发生的同时屏蔽该事件，除非已将 TSF 位读为“1”，否则应用程序不得将“0”写入 TSF 位。

21.2.14 侵入检测

RTC_TAMPx 输入事件可设置为边沿检测或带过滤功能的电平检测。

侵入检测可配置于以下用途：

- 擦除 RTC 备份寄存器
- 产生中断，从停机（Stop）和待机（Standby）模式唤醒系统

RTC 备份寄存器

备份寄存器（RTC_BKPxR）处在 V_{DD} 备份域中，当 V_{DD} 电源被切断，它们仍由 V_{BAT} 维持供电（本系列芯片无 V_{BAT}，故当 V_{DD} 电源被切断时，备份域也掉电）。当系统复位或设备在待机模式下被唤醒时，它们不会被复位。上电复位时，备份寄存器将被复位。

当产生一个侵入检测事件时，备份寄存器将被复位。

侵入检测初始化

所有 RTC_TAMPx 侵入检测输入均与 RTC_ISR2 寄存器的 TAMPxF 标志相关。所有输入可通过设置 RTC_TAFCR 寄存器中相应的 TAMPxE 位为“1”来使能。

侵入检测事件可将所有备份寄存器（RTC_BKPxR）内容清除。

通过设置 RTC_TAFCR 寄存器的 TAMPIE 位，当检测到一个侵入检测事件时便产生一个中断。

侵入事件时间戳

设置 TAMPTS 为“1”，所有侵入事件将产生一个时间戳。这种情况下，RTC_ISR 中无论是 TSF 位还是 TSOVF 位被置 1，和正常的时间戳事件发生的效果相同。设置 TSF 或 TSOVF，与其关联的侵入标志寄存器 TAMPxF 将同时被设置。

侵入输入的边沿检测

如果 TAMPFLT 位为“00”，根据相关 TAMPxTRG 位，当检测到上升沿或下降沿时，RTC_TAMPx 引脚将生成侵入检测事件。当边沿检测被选中后，RTC_TAMPx 输入的内部上拉电阻将被停用。

警告：为避免侵入检测事件丢失并及时地发现侵入检测事件，用于边沿检测的信号与相应 TAMPxE 位执行逻辑与操作，以便在 RTC_TAMPx 引脚上检测到侵入检测事件。

- 当 TAMPxTRG=0：如果在启用侵入检测（通过设置 TAMPxE 位为 1）前 RTC_TAMPx 已经为高电平，一旦启用 RTC_TAMPx 输入，则会产生一个侵入事件（尽管在 TAMPxE 位置“1”后 RTC_TAMPx 输入中并没有出现上升沿）。
- 当 TAMPxTRG=1：如果在启用侵入检测前，RTC_TAMPx 已经为低电平，一旦启用 RTC_TAMPx，则会产生一个侵入事件（尽管在设置 TAMPxE 位后 RTC_TAMPx 输入中并没有出现下降沿）。

在一个侵入事件被检测到并清除后，RTC_TAMPx 复用功能应该被禁用，然后在再次写入备份寄存器前重新启用 RTC_TAMPx 复用功能（通过设置 TAMPxE 位为 1）。这样可以阻止软件在 RTC_TAMPx 检测出仍有侵入事件发生时对备份寄存器进行写操作。这相当于对 RTC_TAMPx 复用功能输入进行电平检测。

注：当电源断开时，侵入检测功能仍然有效。为避免对备份寄存器产生不必要的复位，RTC_TAMPx 复用功能对应的引脚应该在片外连接到正确的电平。

RTC_TAMPx 输入的带滤波功能电平检测

将 TAMPFLT 设置为非“0”值，会启用带滤波功能的电平检测。当检测到 2 个、4 个或 8 个（根据 TAMPFLT 设置）连续样本的指定电平（TAMPxTRG 位决定）时，将产生一个侵入检测事件。

RTC_TAMPx 输入在其状态被采样前，通过 I/O 内部上拉电阻实现预充电，直至 TAMPPUDIS 设置为“1”后，停止该功能。预充电时间由 TAMPPRCH 位决定，允许 RTC_TAMPx 输入拥有更大的电容。

可通过调整 TAMPFREQ 来改变电平检测的采样频率，从而在侵入检测延迟与通过上拉电阻产生的电源消耗间进行取舍。

21.2.15 校准时钟输出

将 RTC_CR 寄存器中的 COE 位置 1 时，会在 RTC_CALIB 器件输出上提供一个参考时钟。如果 RTC_CR 寄存器中的 COSEL 位复位且 PREDIV_A=0x7F，则 RTC_CALIB 频率为 $f_{RTCCLK}/64$。这相当于 RTCCLK 频率为 32.768kHz 时，512Hz 的校准输出。RTC_CALIB 占空比是不规则的：下降沿上存在轻微抖动。因此推荐使用上升沿。

如果 COSEL 置 1 且“PREDIV_S+1”为 256 的非零整数倍（比如：PREDIV_S[7:0]=0xFF），则 RTC_CALIB 频率为 f_{RTCCLK}/（256*（PREDIV_A+1）。这相当于 RTCCLK 频率为 32.768kHz 时，1Hz 的校准输出，其中预分频器为默认值（PREDIV_A=Ox7F、PREDIV_S=0xFF）。

说明：选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚会自动配置为输出复用功能。

21.2.16 闹钟输出

RTC_CR 寄存器中的 OSEL[1:0]控制位用于激活闹钟复用功能输出 RTC_ALARM，以及选择输出的功能。这些功能可反映 RTC_ISR 寄存器中相应标志的内容。

输出的极性由 RTC_CR 中的 POL 控制位确定，这样当 POL 置 1 时会输出选定标志位的相反值。

闹钟复用功能输出

使用 RTC_OR 寄存器中的控制位 RTC_ALARM_TYPE 可将 RTC_ALARM 引脚配置为输出开漏或输出上拉。

说明：

1.使能 RTC_ALARM 输出之后，其优先级高于 RTC_CALIB（与 COE 位无关，该为必须保持清零）。

2.选择 RTC_CALIB 或 RTC_ALARM 输出时，RTC_OUT 引脚会自动配置为输出复用功能。

21.2.17 RTC 低功耗模式

表 21-4 低功耗模式对 RTC 的作用

模式	说明
睡眠（Sleep）	无任何影响，RTC 保持工作状态。 RTC 中断可使器件退出睡眠模式。
停机（Stop）	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持工作状态。RTC 闹钟、RTC 侵入事件、RTC 时间戳事件和 RTC 唤醒会使器件退出停机模式。
待机（Standby）	当 RTC 时钟源为 LSE 或 LSI 时，RTC 保持工作状态。RTC 闹钟、RTC 侵入事件、RTC 时间戳事件和 RTC 唤醒会使器件退出待机模式。

21.2.18 RTC 中断

所有 RTC 中断均与 EXTI 控制器相连。

要使能 RTC 中断，需执行以下序列：

1. 将对应于 RTC 事件的 EXTI 线配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 RTC IRQ 通道并将其使能。
3. 将 RTC 配置为生成 RTC 中断。

表 21-5 中断控制位

中断事件	事件标志	使能控制位	退出睡眠模式	退出停机模式	退出待机模式
闹钟 A	ALRAF	ALRAIE	是	是(1)	是(1)
RTC_TS 输入（时间戳）	TSF	TSIE	是	是(1)	是(1)
RTC_TAMP1 输入检测	TAMP1F	TAMPIE	是	是(1)	是(1)
RTC_TAMP2 输入检测	TAMP2F	TAMPIE	是	是(1)	是(1)
唤醒定时器中断	WUTF	WUTIE	是	是(1)	是(1)

(1). 仅当 RTC 时钟源为 LSE 或 LSI 时，才能从停机和待机模式唤醒。

21.3 RTC 寄存器

基地址：0x4000 2800

空间大小：0x400

21.3.1 RTC 时间寄存器（RTC_TR）

偏移地址：0x00

复位值：0x0000 0000

系统重置：BYPSHAD=0 时为 0x0000 0000。BYPSHAD=1 时不受影响。

说明：RTC_TR 是日历时间影子寄存器，该寄存器只能在初始化模式下写入。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									PM	HT[1:0]		HU[3:0]			
									rw	rw		rw			

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

位 31:23	Res：保留 必须保持复位值。
位 22	PM：AM/PM 标记（AM/PM notation） • 0：AM 或 24 小时制 • 1：PM
位 21:20	HT[1:0]：小时的十位（Hour tens in BCD format）
位 19:16	HU[3:0]：小时的个位（Hour units in BCD format）
位 15	Res：保留 必须保持复位值。
位 14:12	MNT[2:0]：分钟的十位（Minute tens in BCD format）
位 11:8	MNU[3:0]：分钟的个位（Minute units in BCD format）
位 7	Res：保留 必须保持复位值。
位 6:4	ST[2:0]：秒的十位（Second tens in BCD format）
位 3:0	SU[3:0]：秒的个位（Second units in BCD format）

21.3.2 RTC 日期寄存器（RTC_DR）

偏移地址：0x04

复位值：0x0000 2101

系统重置：当 BYPSHAD=0 时，RTC_DR 为 0x00002101;当 BYPSHAD=1 时,RTC_DR 不受影响。

说明：RTC_DR 是日历日期影子寄存器。该寄存器只能在初始化模式下写入。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDU[2:0]			MT	MU[3:0]				Res		DT[1:0]		DU[3:0]			
rw			rw	rw						rw		rw			

位 31:24	Res：保留 必须保持复位值。
位 23:20	YT[3:0]：年的十位（Year tens in BCD format）
位 19:16	YU[3:0]：年的个位（Year units in BCD format）
位 15:13	WDU[2:0]：星期（Week day units） • 000：禁用 • 001：星期一 • … • 111：星期日
位 12	MT：月的十位（Month tens in BCD format）
位 11:8	MU[3:0]：月的个位（Month units in BCD format）
位 7:6	Res：保留 必须保持复位值。
位 5:4	DT[1:0]：日的十位（Date tens in BCD format）
位 3:0	DU[3:0]：日的个位（Date units in BCD format）

21.3.3 RTC 控制寄存器（RTC_CR）

偏移地址：0x08

复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
								rw	rw		rw	rw	rw	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WUTIE	Res	ALRAIE	TSE	WUTE	Res	ALRAE	Res	FMT	BYPSHAD	REFCKON	TSEDGE	WUCKSEL[2:0]		
rw	rw		rw	rw	rw		rw		rw	rw	rw	rw	rw		

位 31:24	Res：保留 必须保持复位值。
位 23	COE：校准输出使能（Calibration output enable） 该位使能 RTC_CALIB 输出。 • 0：校准输出禁用 • 1：校准输出启用

位 22:21	OSEL[1:0]：输出选择（Output selection） 该位用于选择 RTC_ALARM 输出关联的标志位。 • 00：输出禁用 • 01：Alarm A 输出启用 • 10：保留 • 11：保留
位 20	POL：输出极性（Output polarity） 该位用于配置 RTC_ALARM 输出的极性。 • 0：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），该引脚为高电平。 • 1：当 ALRAF/ALRBF/WUTF 置 1 时（取决于 OSEL[1:0]），该引脚为低电平。
位 19	COSEL：校准输出选择（Calibration output selection） 当 COE=1 时，该位可选择 RTC_CALIB 上输出的信号。 • 0：校准输出为 512Hz • 1：校准输出为 1Hz 在 RTCCLK 为 32.768kHz 且预分频器为其默认值（PREDIV_A=127 且 PREDIV_S=255）的条件下，这些频率有效。请参见“21.2.15 校准时钟输出”。
位 18	BKP：备份（Backup） 用户可对该位执行写操作以记录是否已对夏令时进行更改。
位 17	SUB1H：减少 1 小时（冬季时间更改）（Subtract 1hour for winter time change） 当该位在初始化模式以外的模式下置 1 时，如果当前小时不是 0，则日历时间将减少 1 小时。该位始终读为 0。 当前小时为 0 时，将该位置 1 没有任何作用。 • 0：无任何作用 • 1：将当前时间减少 1 小时。这可用于冬季时间更改。
位 16	ADD1H：增加 1 小时（夏季时间更改）（Add 1hour for summer time change） 当该位在初始化模式以外的模式下置 1 时，日历时间将增加 1 小时。该位始终读为 0。 • 0：无任何作用 • 1：将当前时间增加 1 小时。这可用于夏季时间更改。
位 15	TSIE：时间戳中断使能（Time-stamp interrupt enable） • 0：禁止时间戳中断 • 1：使能时间戳中断
位 14	WUTIE：唤醒定时器中断使能（Wakeup timer interrupt enable） • 0：禁止唤醒定时器中断 • 1：使能唤醒定时器中断
位 13	Res：保留 • 必须保持复位值。

位 12	ALRAIE：闹钟 A 中断使能（Alarm A interrupt enable） • 0：禁止闹钟 A 中断 • 1：使能闹钟 A 中断
位 11	TSE：时间戳使能（Timestamp enable） • 0：禁止时间戳 • 1：使能时间戳
位 10	WUTE：唤醒定时器使能（Wakeup timer enable） • 0：禁止唤醒定时器 • 1：使能唤醒定时器
位 9	Res：保留 • 必须保持复位值。
位 8	ALRAE：闹钟 A 使能（Alarm A enable） • 0：禁止闹钟 A • 1：使能闹钟 A
位 7	Res：保留 必须保持复位值。
位 6	FMT：小时格式（Hour format） • 0：24 小时/天格式 • 1：AM/PM 小时格式
位 5	BYPSHAD：旁路影子寄存器（Bypass the shadow registers） • 0：日历值（从 RTC_SSR、RTC_TR 和 RTC_DR 读取时）取自影子寄存器，该影子寄存器每两个 RTCCLK 周期更新一次。 • 1：日历值（从 RTC_SSR、RTC_TR 和 RTC_DR 读取时）直接取自日历计数器。 *注意：如果 APB1 时钟的频率低于 7 倍的 RTCCLK 频率，则必须将 BYPSHAD 置“1”。*
位 4	REFCKON：RTC_REFIN 参考时钟检测使能（50Hz 或 60Hz）（RTC_REFIN reference clock detection enable (50or 60Hz)） • 0：禁止 RTC_REFIN 检测 • 1：使能 RTC_REFIN 检测 *注意：PREDIV_S 必须为 0x00FF。*
位 3	TSEDGE：时间戳事件有效边沿（Time-stamp event active edge） • 0：RTC_TS 输入上升沿生成时间戳事件 • 1：RTC_TS 输入下降沿生成时间戳事件 TSEDGE 发生更改时，必须复位 TSE 以避免将 TSF 意外置 1。
位 2:0	WUCKSEL[2:0]：唤醒时钟选择（Wakeup clock selection） • 000：选择 RTC/16 时钟

	• 001：选择 RTC/8 时钟 • 010：选择 RTC/4 时钟 • 011：选择 RTC/2 时钟 • 10x：选择 Ck_spre 时钟（通常为 1Hz） • 11x：选择 Ck_spre 时钟（通常为 1Hz）并将 WUT 计数器值增加 2^{16}。

21.3.4 RTC 初始化和状态寄存器（RTC_ISR）

偏移地址：0x08

复位值：0x0000 0007

系统复位：不受影响（INIT、INITF 和 RSF 位除外，它们在复位时被清零）

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	TAMP2F	TAMP1F	TSOVF	TSF	WUTF	Res	ALRAF	INIT	INITF	RSF	INITS	SHPF	WUTWF	Res	ALRAWF
	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0		rc_w0	rw	r	rc_w0	r	r	rw		r

位 31:17	Res：保留 必须保持复位值。
位 16	RECALPF：重新校准挂起标志（Recalibration pending Flag） 当软件对 RTC_CALR 寄存器执行写操作时，RECALPF 状态标志将自动置“1”，指示 RTC_CALR 寄存器已屏蔽。当采用新的校准设置时，该位恢复为“0”。
位 15	Res：保留 必须保持复位值。
位 14	TAMP2F：RTC_TAMP2 检测标志（RTC_TAMP2detection flag） 在 RTC_TAMP2 输入上检测到侵入检测事件时，由硬件将此标志置 1。 该标志由软件写零清除。
位 13	TAMP1F：RTC_TAMP1 检测标志（RTC_TAMP1detection flag） 在 RTC_TAMP1 输入上检测到侵入检测事件时，由硬件将此标志置 1。 该标志由软件写零清除。
位 12	TSOVF：时间戳溢出标志（Time-stamp overflow flag） 当在 TSF 已置 1 的情况下发生时间戳事件时，由硬件将此标志置 1。 该标志由软件写零清除。 建议仅在 TSF 位清零之后再检查并清零 TSOVF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则溢出事件可能会被漏掉。
位 11	TSF：时间戳标志（Time-stamp flag） 发生时间戳事件时，由硬件将此标志置 1。该标志由软件写零清除。

位 10	WUTF：唤醒定时器标志（Wakeup timer flag） 当唤醒自动重载计数器计数到 0 时，由硬件将此标志置 1。 该标志由软件写零清除。软件必须在 WUTF 再次置 1 的 1.5 个 RTCCLK 周期之前将该标志清零。
位 9	Res：保留 必须保持复位值。
位 8	ALRAF：闹钟 A 标志（Alarm A flag） 当时间/日期寄存器（RTC_TR 和 RTC_DR）与闹钟 A 寄存器（RTC_ALRMAR）匹配时，由硬件将该标志置 1。 该标志由软件写零清除。
位 7	INIT：初始化模式（Initialization mode） • 0：自由运行模式 • 1：初始化模式，用于编程时间和日期寄存器（RTC_TR 和 RTC_DR）以及预分频器寄存器（RTC_PRER）。计数器停止计数，当 INIT 被复位后，计数器从新值开始计数。
位 6	INITF：初始化标志（Initialization flag） 当该位置 1 时，RTC 处于初始化状态，此时可更新事件、日期和预分频器寄存器。 • 0：不允许更新日历寄存器 • 1：允许更新日历寄存器
位 5	RSF：寄存器同步标志（Registers synchronization flag） 每次将日历寄存器的值复制到影子寄存器（RTC_SSRx、RTC_TRx 和 RTC_DRx）时，都会由硬件将该位置 1。在初始化模式下、平移操作挂起时（SHPF=1）或在旁路影子寄存器模式（BYPSHAD=1）下，该位由硬件清零。该位还可由软件清零。 在初始化模式下，该位可由软件或硬件清零。 • 0：日历影子寄存器尚未同步 • 1：日历影子寄存器已同步
位 4	INITS：初始化状态标志（Initialization status flag） 当日历年份字段不为 0 时（RTC 域复位状态），由硬件将该位置 1。 • 0：日历尚未初始化 • 1：日历已经初始化
位 3	SHPF：平移操作挂起（Shift operation pending） • 0：没有平移操作挂起 • 1：某个平移操作挂起 只要通过对 RTC_SHIFTR 寄存器执行写操作来启动平移操作，此标志便由硬件置 1。执行完相应的平移操作后，此标志由硬件清零。对 SHPF 位执行写入操作不起作用。
位 2	WUTWF：唤醒定时器写标志（Wakeup timer write flag） 该位在 RTC_CR 中的 WUTE 位清零 2 个 RTCCLK 周期后由硬件置 1，在 WUTE 位置 1 后

	2 个 RTCCLK 周期后清零。当 WUTE 位清零且 WUTWF 位置 1 时，可更改唤醒定时器的值。 • 0：不允许更新唤醒定时器配置 • 1：允许更新唤醒定时器配置
位 1	Res：保留 • 必须保持复位值。
位 0	ALRAWF：闹钟 A 写标志（Alarm A write flag） 在 RTC_CR 寄存器中的 ALRAE 位置 0 后，当闹钟 A 的值可更改时，由硬件将该位置 1。该位在初始化模式下由硬件清零。 • 0：不允许更新闹钟 A • 1：允许更新闹钟 A

21.3.5 RTC 预分频器寄存器（RTC_PRER）

偏移地址：0x10

复位值：0x007F00FF

系统复位：不受影响

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

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

位 31:23	Res：保留 必须保持复位值。
位 22:16	PREDIV_A[6:0]：异步预分频系数（Asynchronous prescaler factor） 下面是异步分频系数的公式： $f_{ck\_apre}=f_{RTCCLK}$/（PREDIV_A+1）
位 15	Res：保留 必须保持复位值。
位 14:0	PREDIV_S[14:0]：同步预分频系数（Synchronous prescaler factor） 下面是同步分频系数的公式： $f_{ck\_spre}=f_{RTCCLK}$ /（PREDIV_S+1）

21.3.6 RTC 唤醒定时器寄存器（RTC_WUTR）

偏移地址：0x14

复位值：0x0000FFFF

系统复位：不受影响

仅当 RTC_ISR 中的 WUTWF 置 1 时才可对该寄存器执行写操作。

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
WUT[15:0]															
rw															

位 31:16	Res：保留 必须保持复位值。
位 15:0	WUT[15:0]：唤醒自动重载值（Wakeup auto-reload value bits） 当使能唤醒定时器时（WUTE 置 1），每（WUT[15:0]+1）个 ck_wut 周期将 WUTF 标志置 1 一次。ck_wut 周期通过 RTC_CR 寄存器的 WUCKSEL[2:0]位进行选择。 当 WUCKSEL[2]=1 时，唤醒定时器变为 17 位，WUCKSEL[1]等效为 WUT[16]，即要重载到定时器的最高有效位。 WUTF 第一次置 1 发生在 WUTE 置 1 之后（WUT+1）个 ck_wut 周期。禁止在 WUCKSEL[2:0]=011（RTCCLK/2）时将 WUT[15:0]设置为 0x0000。

21.3.7 RTC 闹钟 A 寄存器（RTC_ALRMAR）

偏移地址：0x1C

复位值：0x0000 0000

系统复位：不受影响

仅当 RTC_ISR 中的 ALRAWF 置 1 时或在初始化模式下，才可以对该寄存器执行写操作。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSK4	WDSEL	DT[1:0]		DU[3:0]				MSK3	PM	HT[1:0]		HU[3:0]			
rw	rw	rw		rw				rw	rw	rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK2	MNT[2:0]			MNU[3:0]				MSK1	ST[2:0]			SU[3:0]			
rw	rw			rw				rw	rw			rw			

位 31	MSK4：闹钟 A 日期掩码（Alarm A date mask） • 0：如果日期/日匹配，则闹钟 A 置 1。 • 1：在闹钟 A 比较中，与日期/日的值无关。
位 30	WDSEL：星期几选择（Week day selection） • 0：DU[3:0]代表日期的个位。 • 1：DU[3:0]代表星期几 DT[1:0]为无关位。
位 29:28	DT[1:0]：日期的十位（BCD 格式）（Date tens in BCD format）
位 27:24	DU[3:0]：日期的个位或日（BCD 格式）（Date units or day in BCD format）
位 23	MSK3：闹钟 A 小时掩码（Alarm A hours mask） • 0：如果小时匹配，则闹钟 A 置 1。 • 1：在闹钟 A 比较中，与小时的值无关。

位 22	PM：AM/PM 符号（AM/PM notation） • 0：AM 或 24 小时制 • 1：PM
位 21:20	HT[1:0]：小时的十位（BCD 格式）（Hour tens in BCD format）
位 19:16	HU[3:0]：小时的个位（BCD 格式）（Hour units in BCD format）
位 15	MSK2：闹钟 A 分钟掩码（Alarm A minutes mask） • 0：如果分钟匹配，则闹钟 A 置 1。 • 1：在闹钟 A 比较中，与分钟的值无关。
位 14:12	MNT[2:0]：分钟的十位（BCD 格式）（Minute tens in BCD format）
位 11:8	MNU[3:0]：分钟的个位（BCD 格式）（Minute units in BCD format）
位 7	MSK1：闹钟 A 秒掩码（Alarm A seconds mask） • 0：如果秒匹配，则闹钟 A 置 1。 • 1：在闹钟 A 比较中，与秒的值无关。
位 6:4	ST[2:0]：秒的十位（BCD 格式）（Second tens in BCD format）
位 3:0	SU[3:0]：秒的个位（BCD 格式）（Second units in BCD format）

21.3.8 RTC 写保护寄存器（RTC_WPR）

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								KEY[7:0]							
								w							

位 31:8	Res：保留 必须保持复位值。
位 7:0	KEY[7:0]：写保护关键字（Write protection key） 可通过软件对该字节执行写操作。读取该字节时，始终返回 0x00。 有关如何解锁 RTC 寄存器写保护的介绍，请参见“21.2.7 RTC 初始化和配置”中的“RTC 寄存器写保护”。

21.3.9 RTC 亚秒寄存器（RTC_SSR）

偏移地址：0x28

复位值：0x0000 0000

系统复位：当 BYPSHAD=0 时为 0x0000 0000。当 BYPSHAD=1 时不受影响。

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
SS[15:0]															
r															

位 31:16	Res：保留 必须保持复位值。
位 15:0	SS[15:0]：亚秒值（Sub second value） SS[15:0]是同步预分频器计数器的值。此亚秒值可根据以下公式得出： 亚秒值=（PREDIV_S -SS）/（PREDIV_S + 1）。 *说明：仅当执行平移操作之后，SS 才能大于 PREDIV_S。在这种情况下，正确的时间/日期比 RTC_TR/RTC_DR 所指示的时间/日期慢一秒钟。*

21.3.10 RTC 平移控制寄存器（RTC_SHIFTR）

偏移地址：0x2C

复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD1S	Res														
w															

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

位 31	ADD1S：增加一秒钟（Add one second） • 0：无任何作用 • 1：对时钟/日历增加一秒钟 该位为只写位且始终读为 0。当平移操作挂起（RTC_ISR 中的 SHPF=1）时，对该位执行写操作无作用。 此函数应与 SUBFS 配合使用（请参见 SUBFS 字段描述），以便有效地向原子操作机制的时钟添加亚秒值。
位 30:15	Res：保留 必须保持复位值。
位 14:0	SUBFS[14:0]：减少亚秒值（Subtract a fraction of a second） 该位为只写位且始终读为 0。当平移操作挂起（RTC_ISR 中的 SHPF=1）时，对该位执行写操作无作用。 写入 SUBFS 的值将加到同步预分频器计数器中。由于该计数器递减计数，此操作可有效地从时钟减去（延迟）以下时间： *延迟（秒）=SUBFS / (PREDIV_S+1)*

	当 ADD1S 函数与 SUBFS 结合使用时，可有效地将亚秒值增加到时钟（提前时钟），使时钟提前以下时间： *提前（秒）=1 - (SUBFS / (PREDIV_S + 1))* *说明：对 SUBFS 执行写操作将使 RSF 清零。软件随后会等待至 RSF=1 以确定影子寄存器已更新为平移后的时间。*

21.3.11 RTC 时间戳时间寄存器（RTC_TSTR）

偏移地址：0x30

复位值：0x0000 0000

系统复位：不受影响

仅当 RTC_ISR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res									PM	HT[1:0]		HU[3:0]			
									r	r		r			

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

位 31:23	Res：保留 必须保持复位值。
位 22	PM：AM/PM 符号（AM/PM notation） • 0：AM 或 24 小时制 • 1：PM
位 21:20	HT[1:0]：小时的十位（BCD 格式）（Hour tens in BCD format）
位 19:16	HU[3:0]：小时的个位（BCD 格式）（Hour units in BCD format）
位 15	Res：保留 必须保持复位值。
位 14:12	MNT[2:0]：分钟的十位（BCD 格式）（Minute tens in BCD format）
位 11:8	MNU[3:0]：分钟的个位（BCD 格式）（Minute units in BCD format）
位 7	Res：保留 必须保持复位值。
位 6:4	ST[2:0]：秒的十位（BCD 格式）（Second tens in BCD format）
位 3:0	SU[3:0]：秒的个位（BCD 格式）（Second units in BCD format）

21.3.12 RTC 时间戳日期寄存器（RTC_TSDR）

偏移地址：0x34

复位值：0x0000 0000

系统复位：不受影响

仅当 RTC_ISR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

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
WDU[2:0]			MT	MU[3:0]				Res		DT[1:0]		DU[3:0]			
r			r	r						r		r			

位 31:16	Res：保留 必须保持复位值。
位 15:13	WDU[2:0]：星期几的个位（Week day units）
位 12	MT：月份的十位（BCD 格式）（Month tens in BCD format）
位 11:8	MU[3:0]：月份的个位（BCD 格式）（Month units in BCD format）
位 7:6	Res：保留 必须保持复位值。
位 5:4	DT[1:0]：日期的十位（BCD 格式）（Date tens in BCD format）
位 3:0	DU[3:0]：日期的个位（BCD 格式）（Date units in BCD format）

21.3.13 RTC 时间戳亚秒寄存器（RTC_TSSSR）

偏移地址：0x38

复位值：0x0000 0000

系统复位：不受影响

仅当 RTC_ISR 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

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
SS[15:0]															
r															

位 31:16	Res：保留 必须保持复位值。
位 15:0	SS[15:0]：亚秒值（Sub second value） 当发生时间戳事件时，SS[15:0]是同步预分频器计数器的值。

21.3.14 RTC 校准寄存器（RTC_CALR）

偏移地址：0x3C

复位值：0x0000 0000

系统复位：不受影响

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
CALP	CALW8	CALW16	Res				CALM[8:0]								
rw	rw	rw					rw								

位 31:16	Res：保留 必须保持复位值。
位 15	CALP：将 RTC 的频率增加 488.5ppm（Increase frequency of RTC by 488.5ppm） • 0：不增加 RTCCLK 脉冲。 • 1：每 211 个脉冲有效插入一个 RTCCLK 脉冲（将频率增加 488.5ppm）。 此功能应与 CALM 结合使用，后者在高分辨率下会降低日历的频率。如果输入频率为 32768Hz，则在 32 秒窗口中增加的 RTCCLK 脉冲数按如下公式计算：（512*CALP）-CALM。
位 14	CALW8：使用 8 秒校准周期（Use an 8-second calibration cycle period） 当 CALW8 置“1”时，选择 8 秒校准周期。 *说明：CALW8=“1”时，CALM[1:0]将始终保持为“00”。*
位 13	CALW16：使用 16 秒校准周期（Use a 16-second calibration cycle period） 当 CALW16 置“1”时，选择 16 秒校准周期。如果 CALW8=1，则不能将该位置“1”。 *说明：当 CALW16=“1”时，CALM[0]将始终保持为“0”。*
位 12:9	Res：保留 必须保持复位值。
位 8:0	CALM[8:0]：负校准（Calibration minus） 在 220 个 RTCCLK 脉冲内屏蔽 CALM 个脉冲（如果输入频率为 32768Hz，则为 32 秒）来降低日历的频率。其分辨率为 0.953 7ppm。 要提高日历的频率，则应将此功能与 CALP 结合使用。

21.3.15 RTC 侵入和复用功能配置寄存器（RTC_TAFCR）

偏移地址：0x40

复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								PC15MODE	PC15VALUE	PC14MODE	PC14VALUE	PC13MODE	PC13VALUE	Res	
								rw	rw	rw	rw	rw	rw		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TAMPPUDIS	TAMPPRCH[1:0]		TAMPFLT[1:0]		TAMPFREQ[2:0]			TAMPTS	Res		TAMP2TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E

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

位 31:24	Res：保留 必须保持复位值。
位 23	PC15MODE：PC15 模式（PC15mode） • 0：PC15 由 GPIO 配置寄存器控制，因此 PC15 在待机模式下浮空。 • 1：LSE 禁用时，PC15 强制为推挽输出模式。
位 22	PC15VALUE：PC15 值（PC15value） 如果 LSE 禁用且 PC15MODE=1，PC15VALUE 设置 PC15 的输出值。
位 21	PC14MODE：PC14 模式（PC14mode） • 0：PC14 由 GPIO 配置寄存器控制，因此 PC14 在待机模式下浮空。 • 1：LSE 禁用时，PC14 强制为推挽输出模式。
位 20	PC14VALUE：PC14 值（PC14value） 如果 LSE 禁用且 PC14MODE=1，PC14VALUE 设置 PC14 的输出值。
位 19	PC13MODE：PC13 模式（PC13mode） • 0：PC13 受 GPIO 配置寄存器约束，因此 PC13 在待机模式下浮空。 • 1：RTC 复用功能禁用时，PC13 强制为推挽输出模式。
位 18	PC13VALUE：RTC_ALARM 输出形式/PC13 值（RTC_ALARM output type/PC13value） 如果 PC13 用于输出 RTC_ALARM，PC13VALUE 设置输出： • 0：RTC_ALARM 为开漏输出。 • 1：RTC_ALARM 为推挽输出。 如果所有 RTC 复用功能禁用且 PC13MODE=1，PC13VALUE 设置 PC13 输出值。
位 17:16	Res：保留 必须保持复位值。
位 15	TAMPPUDIS：RTC_TAMPx 上拉禁止（RTC_TAMPx pull-up disable） 该位决定在每次采样之前是否对每个 RTC_TAMPx 引脚都进行预充电。 • 0：采样之前对 RTC_TAMPx 引脚进行预充电（使能内部上拉）。 • 1：禁止对 RTC_TAMPx 引脚进行预充电。
位 14:13	TAMPPRCH[1:0]：RTC_TAMPx 预充电持续时间（RTC_TAMPx precharge duration） 该位域决定了在每次采样之前激活上拉的持续时间。TAMPPRCH 对每个 RTC_TAMPx 输入都有效。 • 0x0：1 个 RTCCLK 周期 • 0x1：2 个 RTCCLK 周期 • 0x2：4 个 RTCCLK 周期 • 0x3：8 个 RTCCLK 周期

位 12:11	TAMPFLT[1:0]：RTC_TAMPx 过滤器计数（RTC_TAMPx filter count） 该位域决定了为激活侵入事件所需的指定电平上的连续采样次数。TAMPFLT 对每个 RTC_TAMPx 输入都有效。 • 0x0：在 RTC_TAMPx 输入转变为有效电平的边沿激活侵入事件（RTC_TAMPx 输入上无内部上拉）。 • 0x1：在有效电平上连续执行 2 次采样后激活侵入事件。 • 0x2：在有效电平上连续执行 4 次采样后激活侵入事件。 • 0x3：在有效电平上连续执行 8 次采样后激活侵入事件。
位 10:8	TAMPFREQ[2:0]：侵入采样频率（Tamper sampling frequency） 决定对每个 RTC_TAMPx 输入进行采样时的频率。 • 0x0：RTCCLK/32768（RTCCLK=32768Hz 时为 1Hz） • 0x1：RTCCLK/16384（RTCCLK=32768Hz 时为 2Hz） • 0x2：RTCCLK/8192（RTCCLK=32768Hz 时为 4Hz） • 0x3：RTCCLK/4096（RTCCLK=32768Hz 时为 8Hz） • 0x4：RTCCLK/2048（RTCCLK=32768Hz 时为 16Hz） • 0x5：RTCCLK/1024（RTCCLK=32768Hz 时为 32Hz） • 0x6：RTCCLK/512（RTCCLK=32768Hz 时为 64Hz） • 0x7：RTCCLK/256（RTCCLK=32768Hz 时为 128Hz）
位 7	TAMPTS：发生侵入检测事件时激活时间戳（Activate timestamp on tamper detection event） • 0：发生侵入检测事件时不保存时间戳 • 1：发生侵入检测事件时保存时间戳 即便 RTC_CR 寄存器中的 TSE=0，TAMPTS 仍有效。
位 6:5	Res：保留 必须保持复位值。

位 4	TAMP2TRG：RTC_TAMP2 输入的有效电平（Active level for RTC_TAMP2input）
	• 如果 TAMPFLT!=00：
	  ○ 0：RTC_TAMP2 输入保持低电平会触发侵入检测事件。
	  ○ 1：RTC_TAMP2 输入保持高电平会触发侵入检测事件。
	• 如果 TAMPFLT=00：
	  ○ 0：RTC_TAMP2 输入上升沿会触发侵入检测事件。
	  ○ 1：RTC_TAMP2 输入下降沿会触发侵入检测事件。

位 3	TAMP2E：RTC_TAMP2 输入检测使能（RTC_TAMP2input detection enable） • 0：禁止 RTC_TAMP2 检测 • 1：使能 RTC_TAMP2 检测
位 2	TAMPIE：侵入中断使能（Tamper interrupt enable） • 0：禁止侵入中断

	• 1：使能侵入中断 *说明：该位使能所有侵入引脚事件的中断，无论 TAMPxIE 电平如何。如果将该位清零，可以通过将 TAMPxIE 置 1 分别使能每个侵入事件中断。*

位 1	TAMP1TRG：RTC_TAMP1 输入的有效电平（Active level for RTC_TAMP1input）
	• 如果 TAMPFLT!=00
	  ○ 0：RTC_TAMP1 输入保持低电平会触发侵入检测事件。
	  ○ 1：RTC_TAMP1 输入保持高电平会触发侵入检测事件。
	• 如果 TAMPFLT=00：
	  ○ 0：RTC_TAMP1 输入上升沿会触发侵入检测事件。
	  ○ 1：RTC_TAMP1 输入下降沿会触发侵入检测事件。

位 0	TAMP1E：RTC_TAMP1 输入检测使能（RTC_TAMP1input detection enable） • 0：禁止 RTC_TAMP1 检测 • 1：使能 RTC_TAMP1 检测

21.3.16 RTC 闹钟 A 亚秒寄存器（RTC_ALRMASSR）

偏移地址：0x44

复位值：0x0000 0000

系统复位：不受影响

仅当 RTC_CR 中的 ALRAE 复位时或在初始化模式下，才可以对该寄存器执行写操作。

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

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

位 31:28	Res：保留 必须保持复位值。
位 27:24	MASKSS[3:0]：屏蔽从该位开始的最高有效位（Mask the most-significant bits starting at this bit） • 0：不对闹钟 A 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余字段均匹配）。 • 1：在闹钟 A 比较中，SS[14:1]为无关位。仅比较 SS[0]。 • 2：在闹钟 A 比较中，SS[14:2]为无关位。仅比较 SS[1:0]。 • 3：在闹钟 A 比较中，SS[14:3]为无关位。仅比较 SS[2:0]。 • ..... • 12：在闹钟 A 比较中，SS[14:12]为无关位。仅比较 SS[11:0]。 • 13：在闹钟 A 比较中，SS[14:13]为无关位。仅比较 SS[12:0]。 • 14：在闹钟 A 比较中，SS[14]为无关位。仅比较 SS[13:0]。 • 15：所有 15 个 SS 位均进行比较，并且必须全部匹配才能激活闹钟。

	同步计数器的溢出位（位 15）从不进行比较。仅当执行平移操作之后，该位才不为 0。
位 23:15	Res：保留 必须保持复位值。
位 14:0	SS[14:0]：亚秒值（Sub seconds value） 该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 A。仅比较位 0 到 MASKSS-1。

21.3.17 备份寄存器（RTC_BKPxR）（x=0..4）

偏移地址：0x50+4*x

复位值：0x0000 0000

系统复位：不受影响

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

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

位 31:0	BKP[31:0]：应用可向/从这些寄存器写入/读取数据。（The application can write or read data to and from these registers） 当器件以低功耗模式工作时，这些寄存器的内容保持有效。 发生侵入检测事件时该寄存器会被复位，并且只要 TAMPxF=1，该寄存器就一直保持复位。在禁止 Flash 读出保护时复位。

22 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 pluse，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。

22.1 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 100kHz）
  - 支持快速模式（高达 400kHz）
  - 支持超快速模式（高达 1MHz）
  - 支持 7 位和 10 位寻址模式
  - 支持多个 7 位从地址（2 个从设备地址寄存器，1 个具有可配置的掩码位段）
  - 所有 7 位地址应答模式
  - 支持广播呼叫
  - 支持总线上的数据建立和保持时间可软件配置
  - 支持事件管理
  - 支持时钟延展功能
  - 支持软件复位
- 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
- 超快速模式下有 20mA 驱动能力

22.2 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 100kHz)、快速（高达 400kHz）或超快速（高达 1MHz）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）。

22.2.1 I2C 框图

I2C 接口的框图如图 22-1 所示。

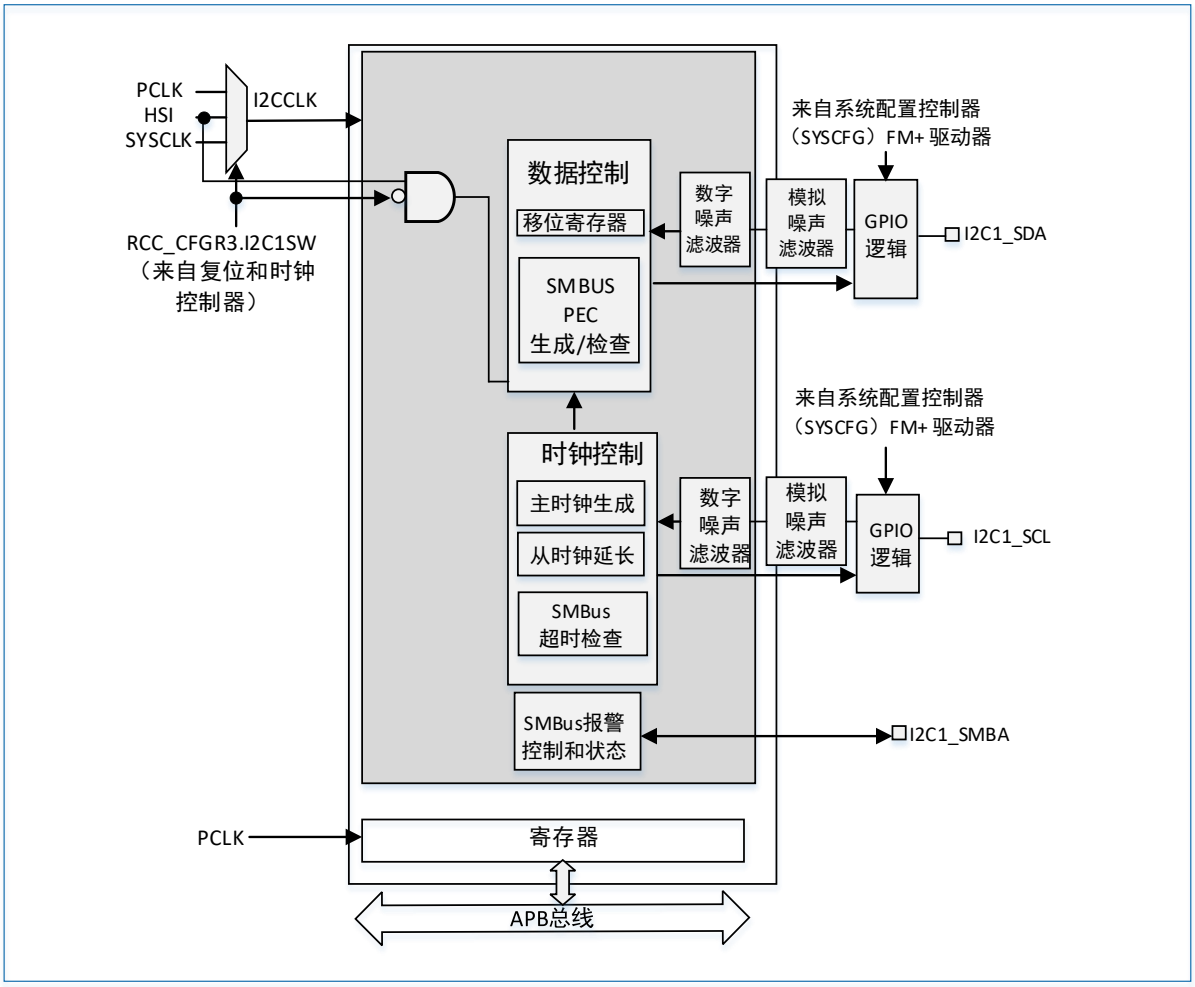


图 22-1 I2C 框图

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：

- PCLK：APB1 时钟（默认值）
- HSI：内部 RC 振荡器
- SYSCLK：系统时钟

更多详细信息，请参见“6 复位和时钟控制（RCC）”。

I2C I/O 支持 20mA 输出电流，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“8.1.6 SYSCFG 配置寄存器 2（SYSCFG_CFGR2）”。

22.2.2 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 260ns。数字滤波器延时为 DNF x t_{I2CCLK}，DNF 的值可通过查询 I2C_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。

22.2.3 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。

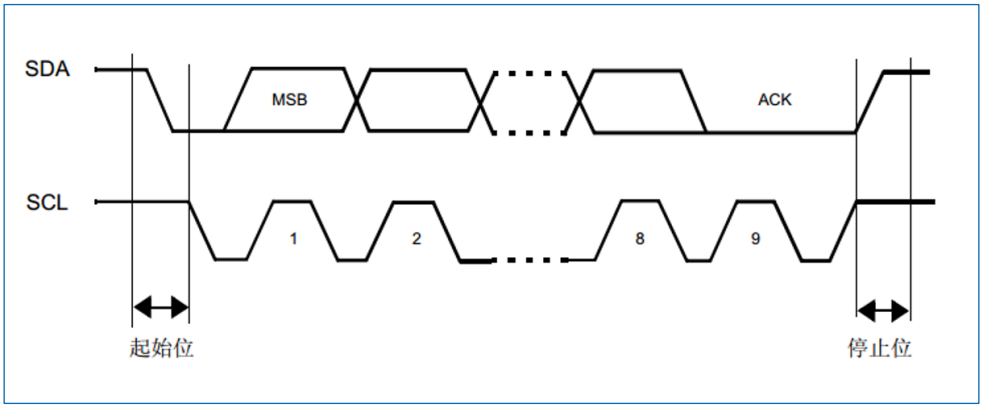


图 22-2 I2C 总线协议

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。

22.2.4 I2C 初始化

使能和禁止外设

I2C 外设时钟必须在时钟控制器中进行配置和使能（请参见“6 复位和时钟控制（RCC）”）。通过将

I2C_CR1 寄存器中的 PE 位置 1 可使能 I2C。

当禁止 I2C（PE=0）时，I2C 将执行软件复位。更多详细信息，请参见“22.2.5 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 50ns 以下的脉冲都要抑制。模块默认使能模拟滤波器，用户可通过将 ANFOFF 位置 1 来禁止该模拟滤波器。

数字滤波器通过配置 I2C_CR1 寄存器中的 DNF[3:0]位可实现对尖峰脉宽 1 到 15 个 I2CCLK 的噪声抑制。使能数字滤波器时，SCL 或 SDA 线的电平只有在电平稳定时间超过 DNFxI2CCLK 个周期后才会发生内部变化。

表 22-1 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止模式唤醒。

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 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLDEL[3:0]和 SDADEL[3:0]位。

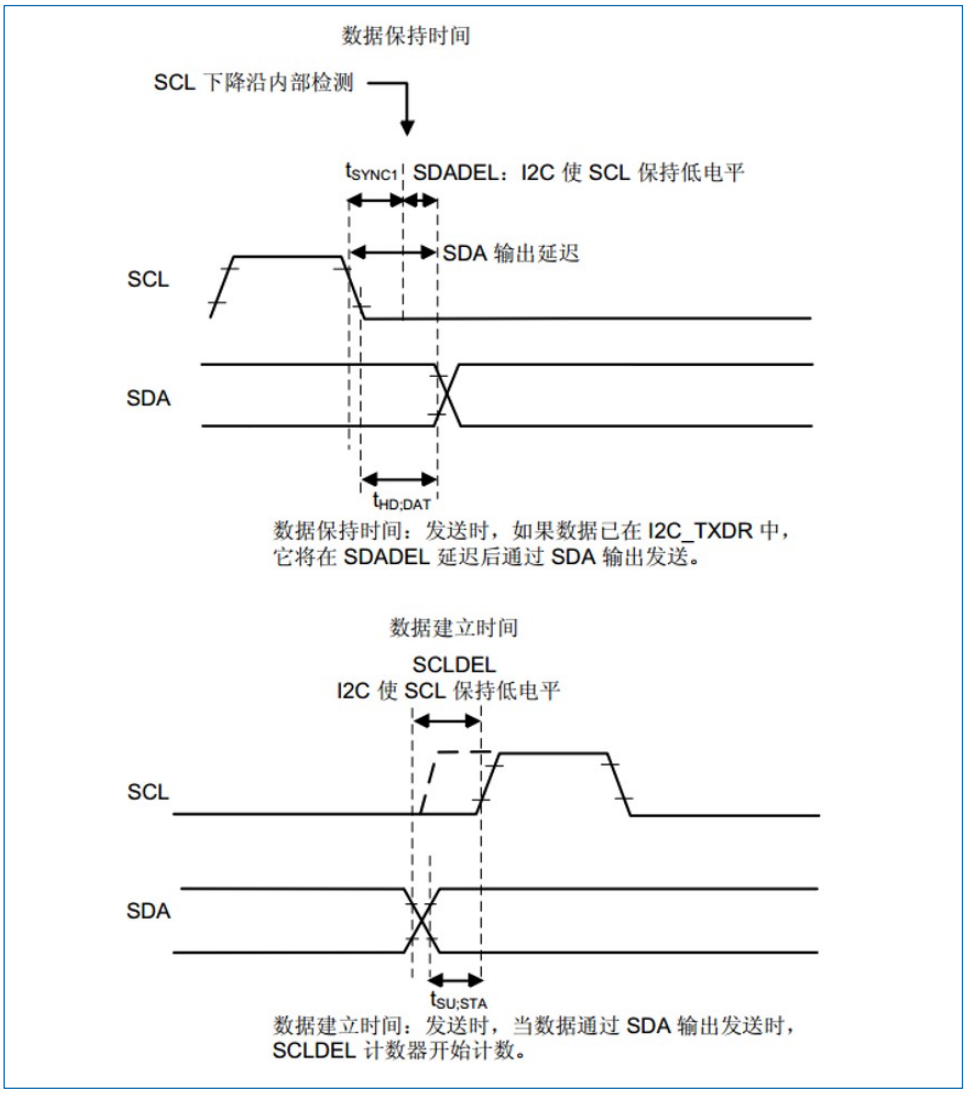


图 22-3 建立和保持时序

- 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}

  TSDADEL 会影响保持时间 $t_{HD;DAT}$。
- SDA 总输出延时为：

  t_{SYNC1}+{[SDADEL x (PRESC+1)+1]x t_{I2CCLK}}

  t_{SYNC1} 持续时间取决于以下参数：
  - SCL 下降斜率
  - 模拟滤波器（使能时）引入的输入延时：t_{AF}（min）<T_{AF}<T_{AF}（max）ns
  - 数字滤波器（使能时）引入的输入延时：t_{DNF}=DNF x t_{I2CCLK}
  - SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

为了桥接 SCL 下降沿的未定义区域，用户编程 SDADEL 时必须遵循以下条件：

$$\{t_{f(max)}+t_{HD;DAT(min)}-t_{AF(min)}-[(DNF+3)xt_{I2CCLK}]\}/\{(PRESC+1)xt_{I2CCLK}\}\leq SDADEL$$

$$SDADEL \leq \{t_{HD;DAT(max)}-t_{AF(max)}-[(DNF+4)xt_{I2CCLK}]\}/\{(PRESC+1)xt_{I2CCLK}\}$$

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《HK32F031 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：

$$SDADEL \leq \{t_{VD;DAT(max)} - t_{R(max)} - 260ns - [(DNF+4) \times t_{I2CCLK}]\} / \{(PRESC+1) \times t_{I2CCLK}\}$$

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 22-2。

- 在 t_{SDADEL} 延时后，或在因数据未写入 I2C_TXDR 寄存器而导致从器件必须延长时钟的情况下发送 SDA 输出后，SCL 线会在建立时间内保持低电平。该建立时间为 t_{SCLDEL}=（SCLDEL+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）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：

$$\{[t_R(max) + t_{SU;DAt}(min)]/[(PRESC+1) \times t_{I2CCLK}]\} - 1 <= SCLDEL$$

有关 tr 和 $t_{SU;DAt}$ 标准值的信息，请参见表 22-2。

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 I2C_TXDR，则 I2C 会继续延长 SCL 低电平时间，直到写入下一个数据。随后，会将新数据 MSB 发送到 SDA 输出，SCLDEL 计数器将开始计数，同时会继续延长 SCL 低电平时间以确保提供充足的数据建立时间。

如果从模式下 NOSTRETCH=1，则 SCL 不会延长。因此，编程 SDADEL 时还必须确保提供充足的建立时间。

表 22-2 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
tr	SDA 和 SCL 信号的上升时间	-	1000	-	300	-	120	-	1000	
tf	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 I2C_TIMINGR 寄存器中的 PRESC[3:0]、SCLH[7:0]和 SCLL[7:0]位来配置 SCL 时钟的高电平和低电平。

- 当内部检测到 SCL 下降沿时，会在释放 SCL 输出之前插入一段延时。该延时为 t_{SCLL}=（SCLL+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）x t_{I2CCLK}。t_{SCLL} 会影响 SCL 低电平时间 t_{LOW}。

- 当内部检测到 SCL 上升沿时，会在将 SCL 输出强制为低电平之前插入一段延时。该延时为 t_{SCLH}=（SCLH+1）x t_{PRESC}，其中 t_{PRESC}=（PRESC+1）xt_{I2CCLK}。t_{SCLH} 会影响 SCL 高电平时间 t_{HIGH}。

更多详细信息，请参见“22.2.8 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“22.2.7 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。

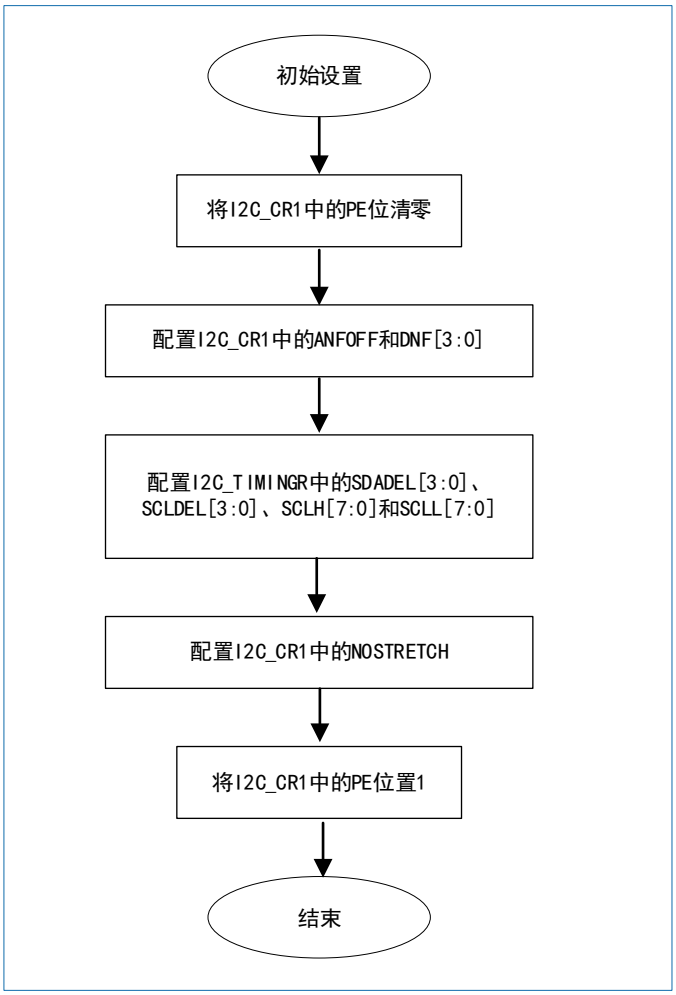


图 22-4 I2C 初始化流程图

22.2.5 软件复位

可通过将 I2C_CR1 寄存器中的 PE 位清零来执行软件复位。在这种情况下，I2C 线 SCL 和 SDA 被释放。内部状态机复位，通信控制位和状态位重置为复位值。配置寄存器不受影响。

下面列出了受影响的寄存器位：

- I2C_CR2 寄存器：START、STOP 和 NACK
- I2C_ISR 寄存器：BUSY、TXE、TXIS、RXNE、ADDR、NACKF、TCR、TC、STOPF、BERR、ARLO 和 OVR

支持 SMBus 功能时还会影响到以下寄存器位：

- I2C_CR2 寄存器：PECBYTE
- I2C_ISR 寄存器：PECERR、TIMEOUT 和 ALERT

由于寄存器存在跨时钟域设计，必须使 PE 保持低电平持续至少 3 个 APB 时钟周期，才能成功执行

软件复位。写入以下软件序列可确保这一点：

1. 写入 PE=0；
2. 检查 PE 位，直到 PE=0；
3. 写入 PE=1。

22.2.6 数据传输

数据传输由发送和接收数据寄存器以及移位寄存器来管理。

接收

在 SDA 上接收的数据被填充到内部移位寄存器。在第 8 个 SCL 脉冲后，数据已全部接收到移位寄存器中。如果 I2C_RXDR 寄存器为空（RXNE=0），则移位寄存器的内容会复制到 I2C_RXDR 寄存器中。

如果 RXNE=1，意味着尚未读取上一次接收到的数据字节。

作为 I2C 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 I2C_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 I2C_RXDR 为止。如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

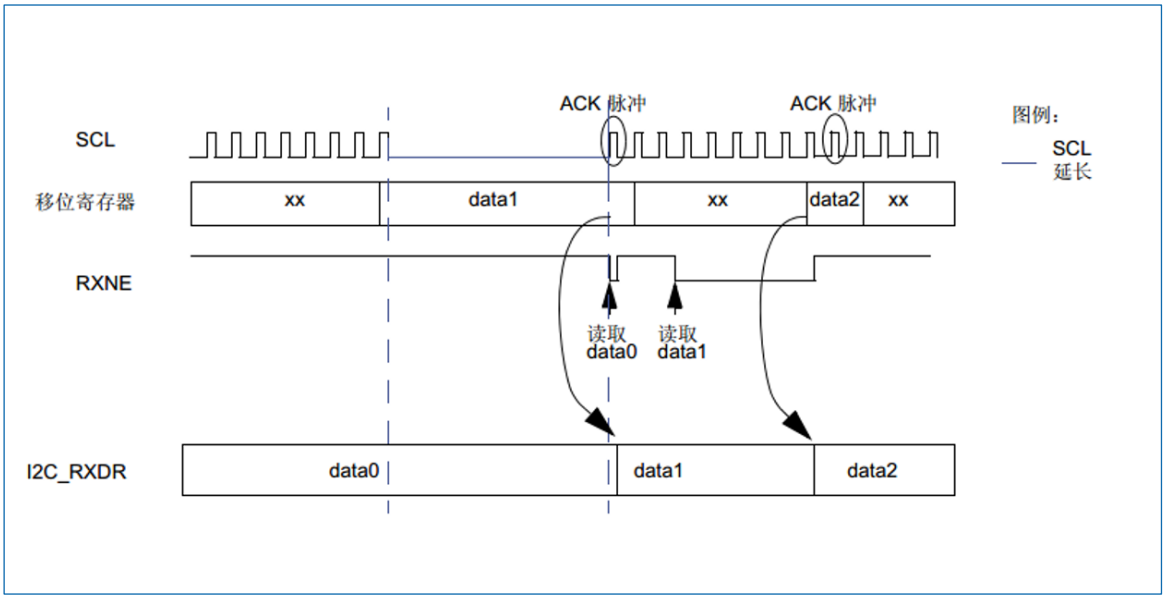


图 22-5 数据接收

发送

如果 I2C_TXDR 寄存器不为空（TXE=0），则其内容会在第 9 个 SCL 脉冲（ACK 脉冲）后复制到移位寄存器中。然后移位寄存器的内容会移出到 SDA 线上。

如果 TXE=1，意味着 I2C_TXDR 内尚未写入任何数据。

作为 I2C Master，硬件将延长第 9 个 SCL 脉冲后的低电平，直到写入了 I2C_TXDR 为止。

作为 I2C Slave，如果 NOSTRETCH=0，将延长时钟，直到软件写 I2C_TXDR；如果 NOSTRETCH=1，则不会延长时钟，但会产生溢出错误。

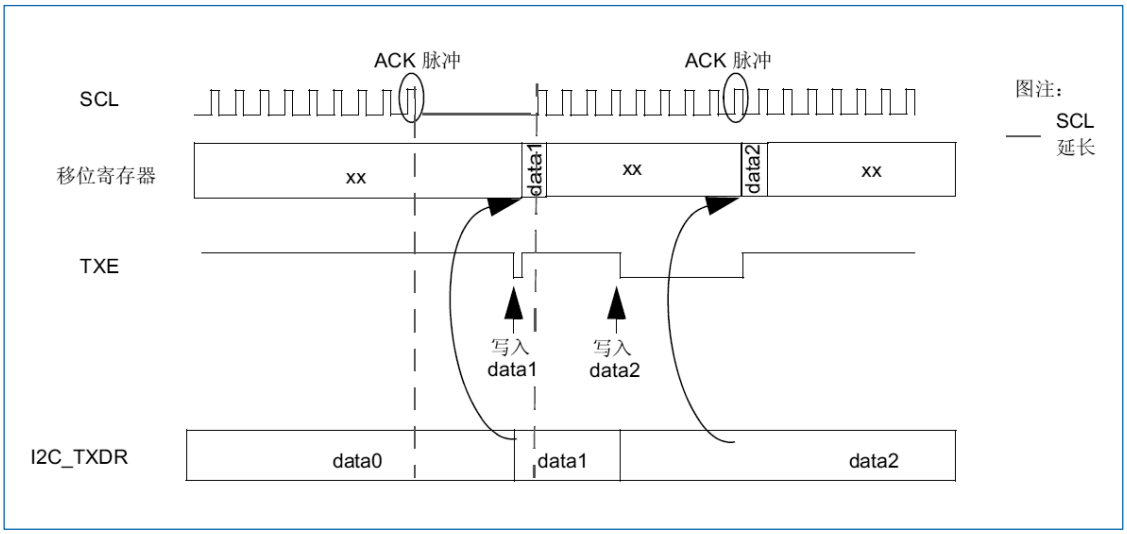


图 22-6 数据发送

硬件传输管理

I2C 在硬件中内置了字节计数器，以便在下列各种模式下管理字节传输和结束通信：

- 主模式下生成 NACK、STOP 和 ReSTART
- 从接收器模式下控制回复 ACK/NACK
- SMBus 模式下生成 PEC/对接收的数据进行 PEC 校验

字节计数器在主模式下默认开启。在从模式下，字节计数器默认为关闭状态，但可以通过软件来使能，方法是将 I2C_CR2 寄存器中的 SBC（从字节控制）位置 1。

待传输的字节数在 I2C_CR2 寄存器的 NBYTES[7:0]位域中进行编程。如果待传输的字节数（NBYTES）大于 255，或者接收方希望控制是否对接收到的数据字节进行应答，则必须选择重载模式，方法是将 I2C_CR2 寄存器的 RELOAD 位置 1。在该模式下，完成 NBYTES 中所编程字节数的数据传输之后，TCR 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TCR 标志置 1，SCL 便会延长。当往 NBYTES 写入一个非零值时，TCR 由软件清零。

在向 NBYTE 中设置最后一次传输的字节数前，必须把 RELOAD 位清零。

在主模式下，RELOAD=0 时，字节计数器功能如下：

- 自动结束模式（I2C_CR2 寄存器中的 AUTOEND="1"）。在该模式下，一旦完成 NBYTES[7:0]位域中所编程字节数的数据传输，主器件便会自动发送停止位。
- 软件结束模式（I2C_CR2 寄存器中的 AUTOEND="0"）在该模式下，一旦完成 NBYTES[7:0]位域中所编程字节数的数据传输，TC 标志将置 1，并且 TCIE 置 1 时将生成中断。只要 TC 标志置 1，SCL 信号便会延长，需要软件介入操作。当软件把 I2C_CR2 寄存器中的起始位或停止位置 1 时，TC 标志将被清零。当主器件要发送重复起始位时，必须使用该模式。

注意：当 RELOAD 位置 1 时，AUTOEND 位将不起作用。

表 22-3 I2C 配置表

功能	SBC 位	RELOAD 位	AUTOEND 位
主 Tx/RxNBYTES+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

22.2.7 从模式

I2C 从模式初始化

在从模式下工作，用户必须至少使能一个从地址。可使用 I2C_OAR1 和 I2C_OAR2 这两个寄存器来编程自身从地址 OA1 和 OA2。

- OA1 既可配置为 7 位寻址模式（默认），也可通过将 I2C_OAR1 寄存器 OA1MODE 位置 1 配置为 10 位寻址模式。

  通过将 I2C_OAR1 寄存器中的 OA1EN 位置 1 来使能 OA1。

- 如果需要额外的从地址，可配置第 2 个从地址 OA2。将 I2C_OAR2 寄存器的 OA2MSK[2:0]位置 1 最多可屏蔽 7 个 OA2LSB。因此，当 OA2MSK 配置为 1 到 6 时，将分别只有 OA2[7:2]、OA2[7:3]、OA2[7:4]、OA2[7:5]、OA2[7:6]或 OA2[7]与接收到的地址作比较。只要 OA2MSK 不等于 0，OA2 的地址比较器便会排除 I2C 保留地址（0000XXX 和 1111XXX），这些地址将不会得到应答。如果 OA2MSK=7，接收到的所有 7 位地址（保留地址除外）均得到应答。OA2 始终为 7 位地址。

  如果这些保留地址在 I2C_OAR1 或 I2C_OAR2 寄存器中进行了编程并且 OA2MSK=0，则它们可以在通过特定使能位使能后得到应答。

  通过将 I2C_OAR2 寄存器中的 OA2EN 位置 1 来使能 OA2。

- 通过将 I2C_CR1 寄存器中的 GCEN 位置 1 来使能广播呼叫地址。

当通过 I2C 的其中一个使能地址来寻址到该 I2C 设备时，ADDR 中断状态标志将置 1，并且 ADDRIE 位置 1 时将生成中断。

默认情况下，从器件使用其时钟延长功能（即必要时延长 SCL 信号的低电平时间）来为软件操作的执行提供时机。如果主器件不支持时钟延长，则必须对 I2C 进行如下配置：将 I2C_CR1 寄存器中 NOSTRETCH 位置 1。

接收到 ADDR 中断后，如果使能多个地址，则用户必须读取 I2C_ISR 寄存器中的 ADDCODE[6:0]位，以确定是哪个地址匹配。还必须检查 DIR 标志，以获悉传输方向。

带时钟延长的从模式（NOSTRETCH=0）

在默认模式下，I2C 从器件会在以下情况下延长 SCL 时钟：

- ADDR 标志置 1 时：接收到的地址与其中一个使能的从地址匹配。通过软件将 ADDRCF 位置 1 以清零 ADRR 标志时，将释放该时钟延展。
- 发送时，前一次数据传输已完成但 I2C_TXDR 寄存器中未写入任何新数据，或者 ADDR 标志清零（TXE=1）时未写入第一个数据字节。往 I2C_TXDR 寄存器中写入数据时，将释放该时钟延展。
- 接收时，尚未读取 I2C_RXDR 寄存器但新的数据接收已完成。读取 I2C_RXDR 时，将释放该时钟延展。
- 当从器件字节控制模式和重载模式（SBC=1 且 RELOAD=1）下 TCR=1 时，这意味着最后一个数据字节已完成传输。通过向 NBYTES[7:0]字段写入一个非零值以将 TCR 清零时，将释放该时钟延展。
- 在 SCL 下降沿检测之后，I2C 会延长 SCL 的低电平时间（不超过[（SDADEL+SCLDEL+1）x（PRESC+1）+1]x t_{I2CCLK}）。

不带时钟延长的从模式（NOSTRETCH=1）

当 I2C_CR1 寄存器中的 NOSTRETCH=1 时，I2C 从器件不会延长 SCL 信号。

- ADDR 标志置 1 时，不会延长 SCL 时钟。
- 发送时，必须在与发送数据对应的第一个 SCL 脉冲出现之前，向 I2C_TXDR 寄存器写入数据。

否则，会发生下溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。当第一次数据发送开始而 STOPF 位仍置 1（尚未清零）时，OVR 标志也将置 1。因此，如果写入下一次传输要发送的第一个数据后才清零上一次传输的 STOPF 标志，则应提供 OVR 状态，甚至对于待发送的第一个数据也是如此。

- 接收时，必须在下一个数据字节的第 9 个 SCL 脉冲（ACK 脉冲）出现之前，从 I2C_RXDR 寄存器读取数据。否则，会发生上溢，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

从器件字节控制模式

要在从接收模式下实现字节 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从器件字节控制模式。这样符合 SMBus 标准。

要在从接收模式下实现字节 ACK 控制，必须选择重载模式（RELOAD=1）。要控制每个字节，必须在 ADDR 中断子程序中将 NBYTES 初始化为 0x1，并在每接收一个字节后将 NBYTE 重载为 0x1。接收到字节后，TCR 位将置 1，从而延长 SCL 信号的第 8 个和第 9 个脉冲之间的低电平时间。用户可以从 I2C_RXDR 寄存器中读取数据，然后通过配置 I2C_CR2 寄存器中的 NACK 位来决定是否应答。通过将 NBYTES 编程为非零值来释放 SCL 延长：发送应答或不应答信号，然后可继续接收下一个字节。

NBYTES 可加载大于 0x1 的值，在这种情况下，接收流在 NBYTES 个数据接收期间是连续的。

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。*

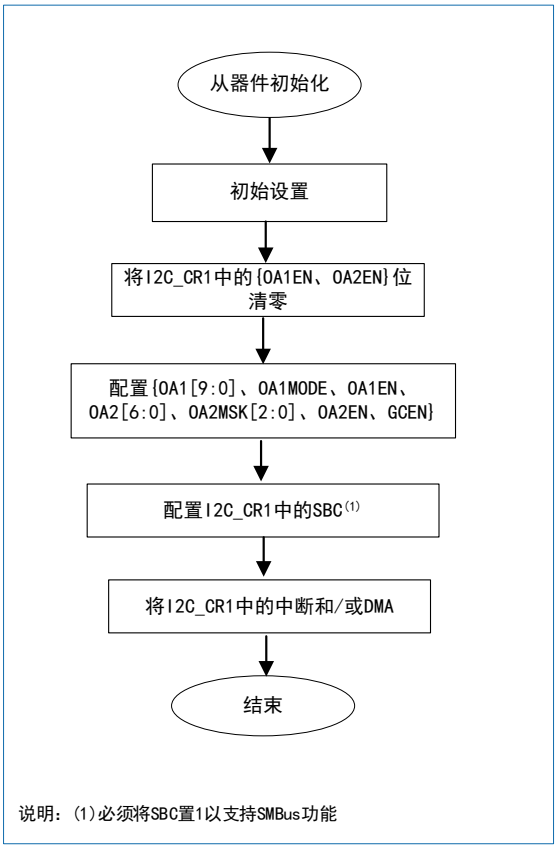


图 22-7 从器件初始化流程图

从发送器

当 I2C_TXDR 寄存器为空时，将生成发送中断状态（TXIS）。如果 I2C_CR1 寄存器中的 TXIE 位置 1，将

生成中断。

I2C_TXDR 寄存器中写入待发送的下一个数据字节时，TXIS 位将被清零。

接收到 NACK 时，I2C_ISR 寄存器中的 NACKF 位将置 1，如果 I2C_CR1 寄存器中的 NACKIE 位置 1，还将生成中断。从器件自动释放 SCL 和 SDA 线，以使主器件执行停止或重复起始位的发送。收到 NACK 时，TXIS 位不会置 1。

当接收到停止位且 I2C_CR1 寄存器中的 STOPIE 位置 1 时，I2C_ISR 寄存器中的 STOPF 标志将置 1 并且会生成中断。在大多数应用中，SBC 位通常编程为“0”。在这种情况下，如果接收到从地址（ADDR=1）时 TXE=0，用户可以选择发送 I2C_TXDR 寄存器的内容作为第一个数据字节，也可以选择通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器以编程新的数据字节。

在从器件字节控制模式（SBC=1）下，必须在地址匹配中断子程序（ADDR=1）中向 NBYTE 写入待发送数据的个数。在这种情况下，传输期间 TXIS 事件的数量对应于 NBYTES 中编程的值。

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 I2C_TXDR 寄存器的内容，从而编程第一个数据字节。必须在 I2C_TXDR 寄存器中预编程待发送的第一个数据字节：

- 该数据可以是前一个传输消息的最后一个 TXIS 事件中写入的数据。
- 如果该数据字节不是待发送的数据字节，可通过将 TXE 位置 1 来刷新 I2C_TXDR 寄存器，从而编程新的数据字节。必须仅在执行完这些操作后再清零 STOPF 位，以确保在地址应答之后，第一次数据传输开始之前执行这些操作。

如果第一次数据传输开始时 STOPF 仍置 1，则将生成下溢错误（OVR 标志置 1）。

如果需要 TXIS 事件（发送中断或发送 DMA 请求），用户必须将 TXE 位和 TXIS 位均置 1，以便生成 TXIS 事件。

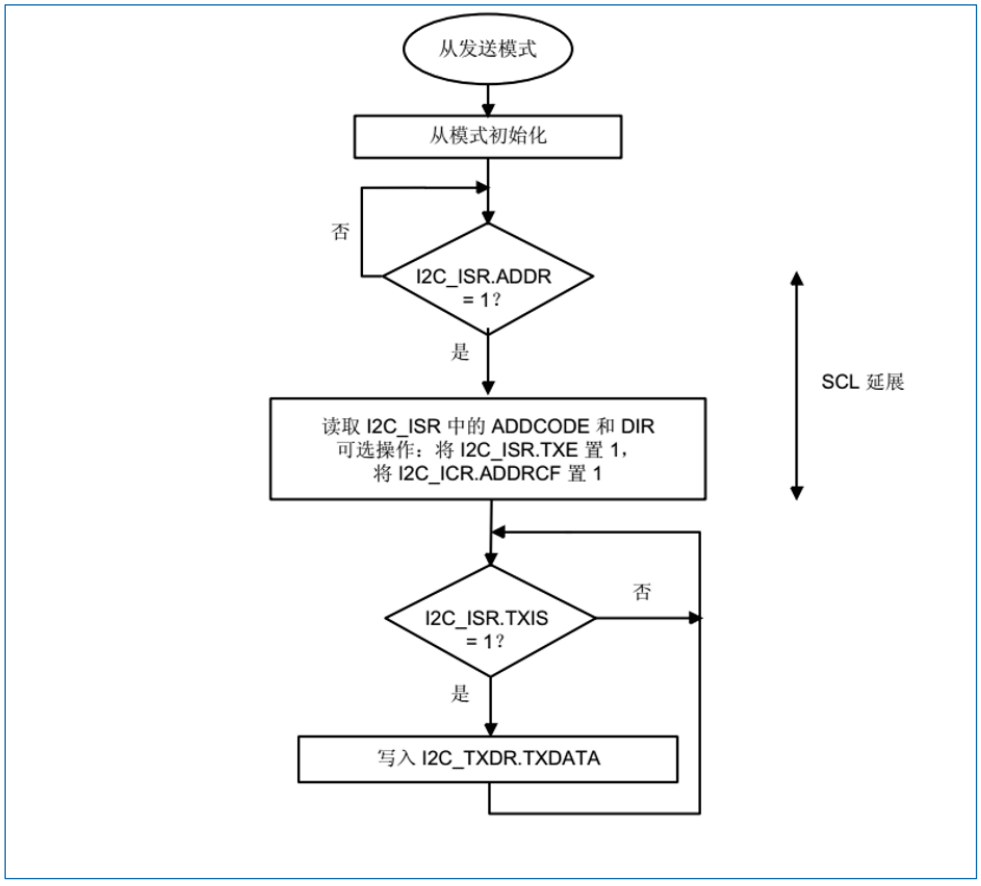


图 22-8 I2C 从发送器的传输序列流程图（NOSTRETCH=0）

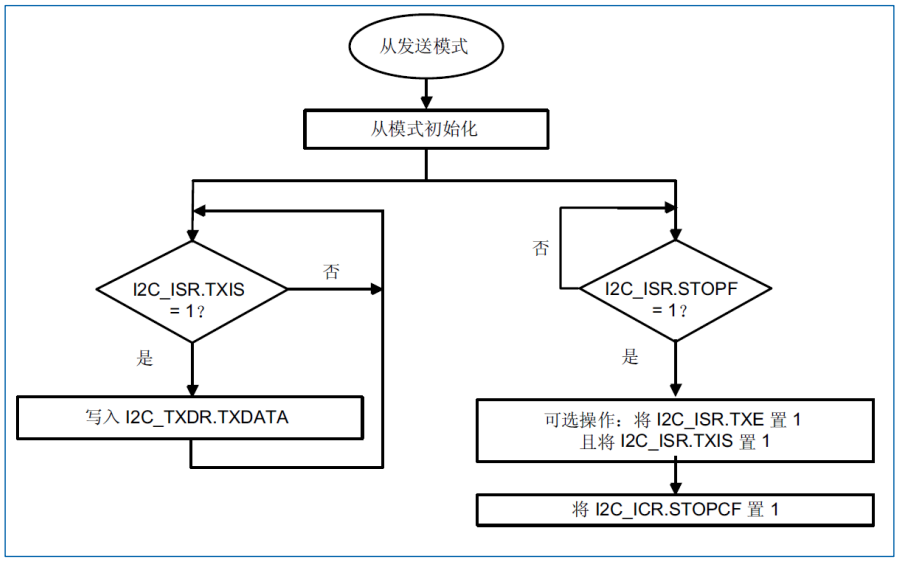


图 22-9 I2C 从发送器的传输序列流程图（NOSTRETCH=1）

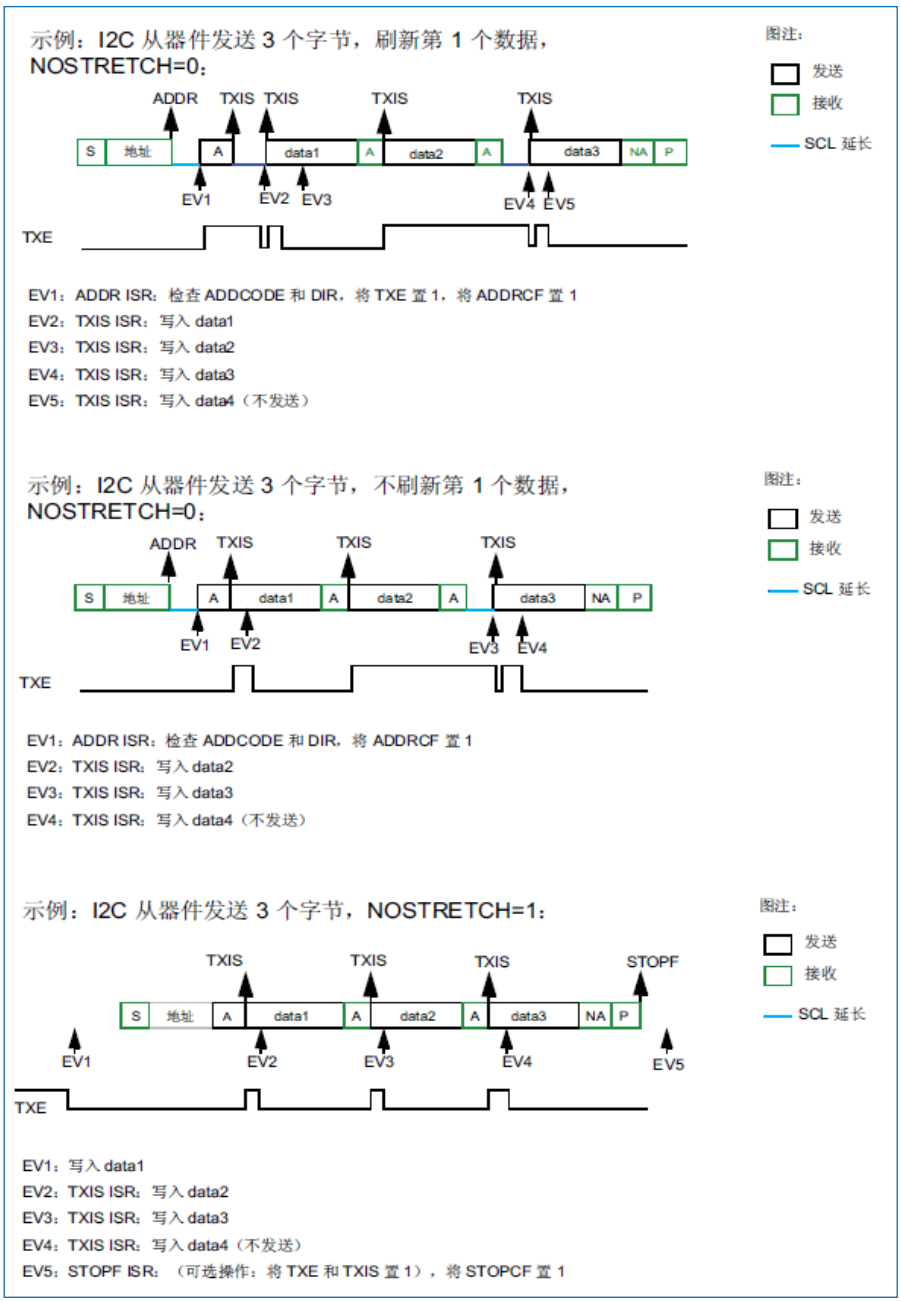


图 22-10 从发送器的传输总线图

从接收器

当 I2C_RXDR 满时，I2C_ISR 中的 RXNE 将置 1，如果 I2C_CR1 中的 RXIE 置 1，还将生成中断。读取 I2C_RXDR 时，将清零 RXNE。

接收到停止条件且 I2C_CR1 寄存器中的 STOPIE 置 1 时，I2C_ISR 中的 STOPF 将置 1 并且会生成中断。

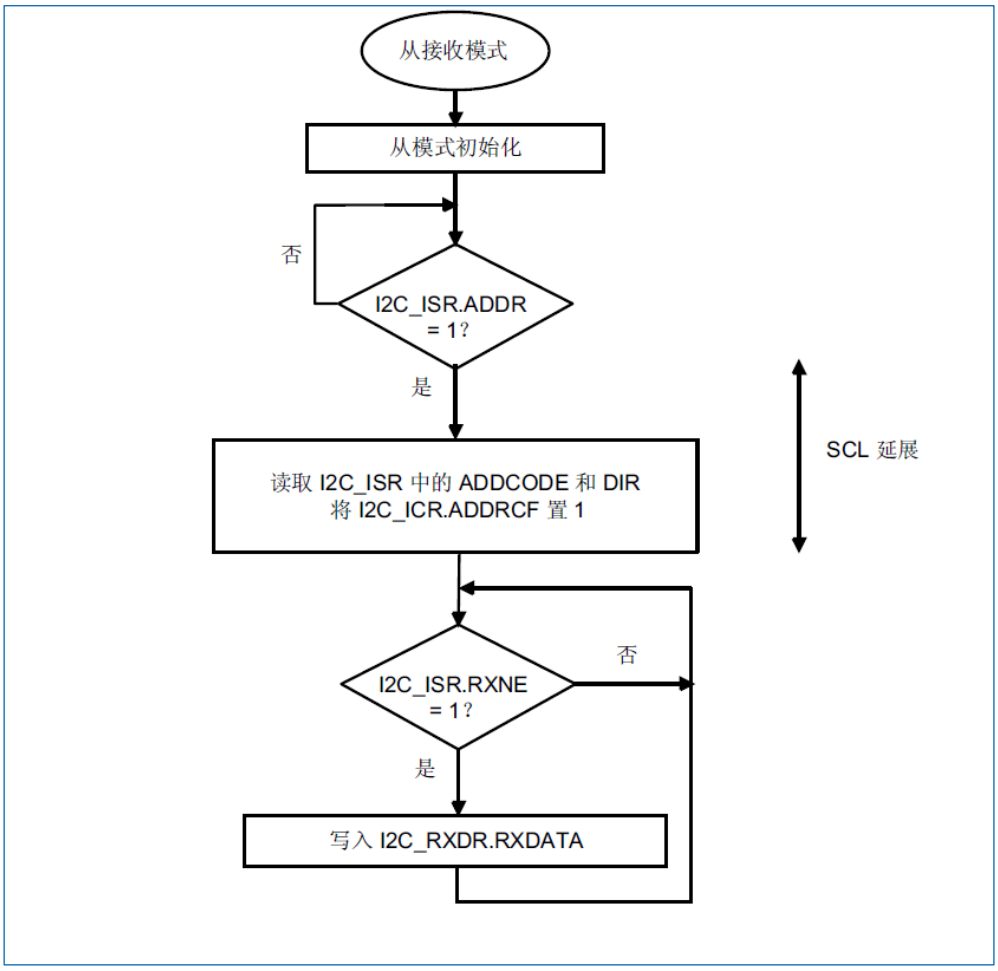


图 22-11 从接收器的传输序列流程图（NOSTRETCH=0）

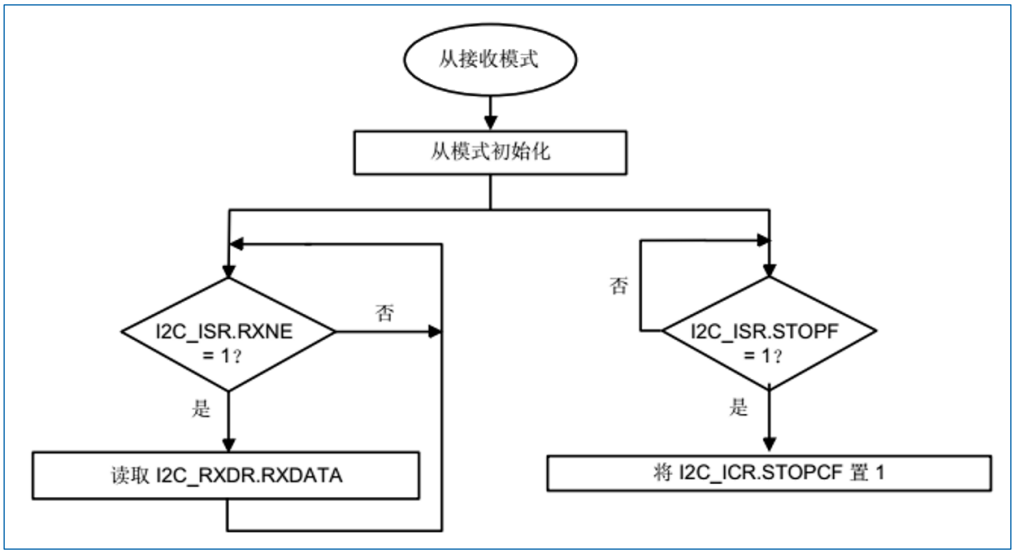


图 22-12 从接收器的传输序列流程图（NOSTRETCH=1）

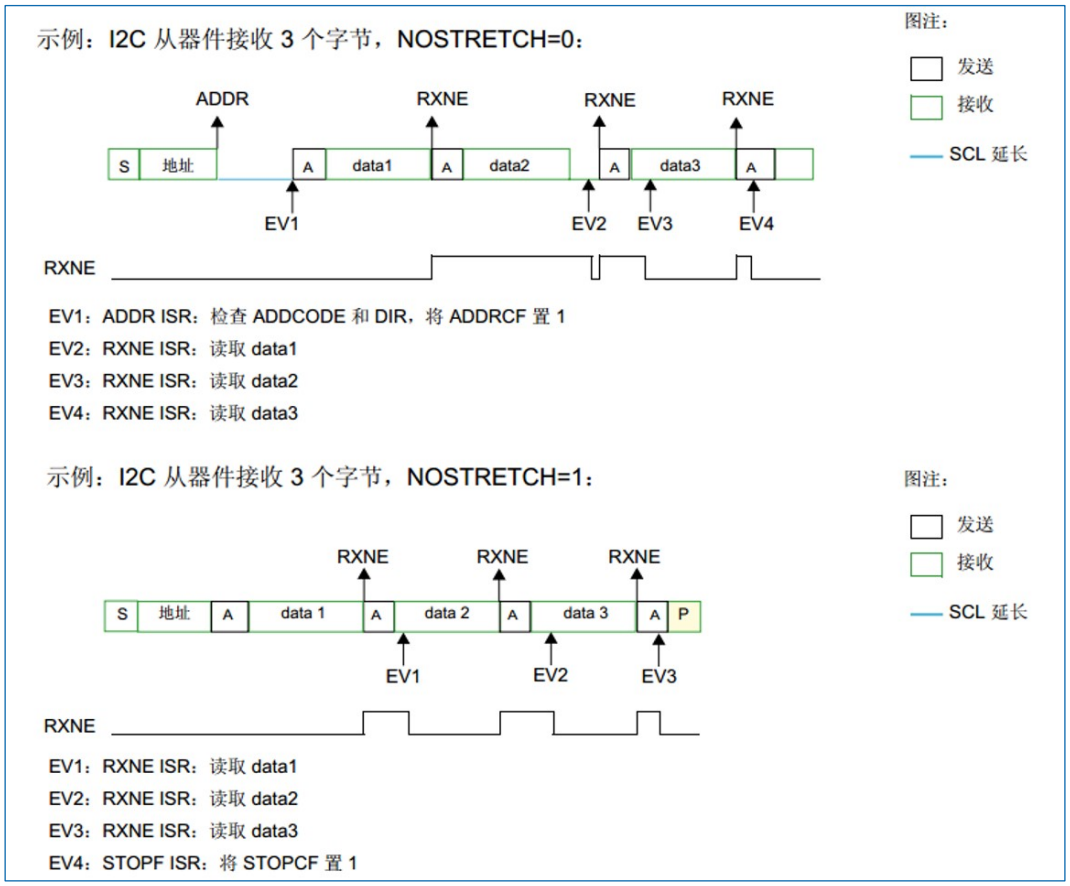


图 22-13 I2C 从接收器的传输总线图

22.2.8 主模式

I2C 主模式初始化

使能外设前，必须通过设置 I2C_TIMINGR 寄存器中的 SCLH 和 SCLL 位来配置 I2C 主时钟。

为了支持多主环境和从时钟延长，I2C 实现了时钟同步机制。

为了实现时钟同步，需执行以下操作：

- 使用 SCLL 计数器，从 SCL 低电平内部检测开始对时钟的低电平进行计数。
- 使用 SCLH 计数器，从 SCL 高电平内部检测开始对时钟的高电平进行计数。

I2C 经过 t_{SYNC1} 延时后，检测其自身的 SCL 低电平，该延时取决于 SCL 下降沿、SCL 输入噪声滤波器（模拟+数字）以及 SCL 与 I2CxCLK 时钟的同步。一旦 SCLL 计数器达到 I2C_TIMINGR 寄存器的 SCLL[7:0]位中编程的值，I2C 便会将 SCL 释放为高电平。

I2C 经过 t_{SYNC2} 延时后检测其自身的 SCL 高电平，该延时取决于 SCL 上升沿、SCL 输入噪声滤波器（模拟+数字）以及 SCL 与 I2CxCLK 时钟的同步。一旦 SCLH 计数器达到 I2C_TIMINGR 寄存器的 SCLH[7:0]位中编程的值，I2C 便会使 SCL 变为低电平。

因此，主时钟周期为：

$$t_{SCL}=t_{SYNC1}+t_{SYNC2}+\{[（SCLH+1）+（SCLL+1）]x（PRESC+1）x\ t_{I2CCLK}\}$$

t_{SYNC1} 的持续时间取决于以下参数：

- SCL 下降斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}

- SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

t_{SYNC2} 的持续时间取决于以下参数：

- SCL 上升斜率
- 模拟滤波器（使能时）引入的输入延时
- 数字滤波器（使能时）引入的输入延时：DNF x t_{I2CCLK}
- SCL 与 I2CCLK 时钟建立同步而产生的延时（2 到 3 个 I2CCLK 周期）

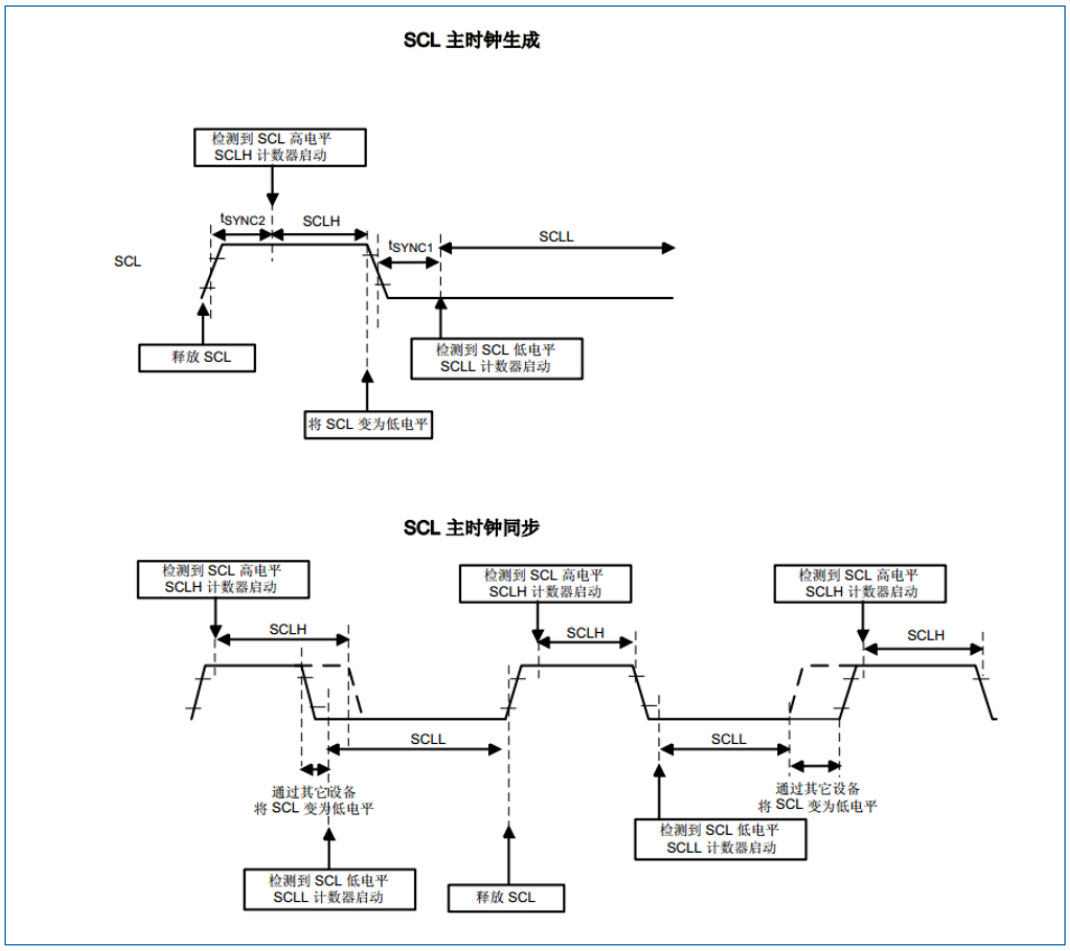


图 22-14 主时钟生成

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 22-4 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

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_{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
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：

(1). SCLL 还用于生成 t_{BUF} 和 $t_{SU;STA}$ 时序。

(2). SCLH 还用于生成 $t_{HD;STA}$ 和 $t_{SU;STO}$ 时序。

主模式通信初始化（地址阶段）

要发起通信，用户必须在 I2C_CR2 寄存器中为寻址的从器件编程以下参数：

- 寻址模式（7 位或 10 位）：ADD10
- 待发送的从地址：SADD[9:0]
- 传输方向：RD_WRN
- 读取 10 位地址时：HEAD10R 位。必须对 HEAD10R 进行相应配置，以指示传输方向变化时必须发送完整的地址序列，还是只发送地址头。
- 待传输的字节数：NBYTES[7:0]。如果字节数等于或大于 255，则初始化时必须将 NBYTES[7:0]填充为 0xFF。然后，用户必须将 I2C_CR2 寄存器中的 START 位置 1。START 位置 1 时，不允许更改上述所有位。

之后，当主器件检测到总线空闲（BUSY=0）时，它会在经过 T_{BUF} 的延时后自动发送起始位，随后发出从器件地址。

仲裁丢失时，主器件将自动切换回从模式，如果作为从器件被寻址，还可对其自身地址进行应答。

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。

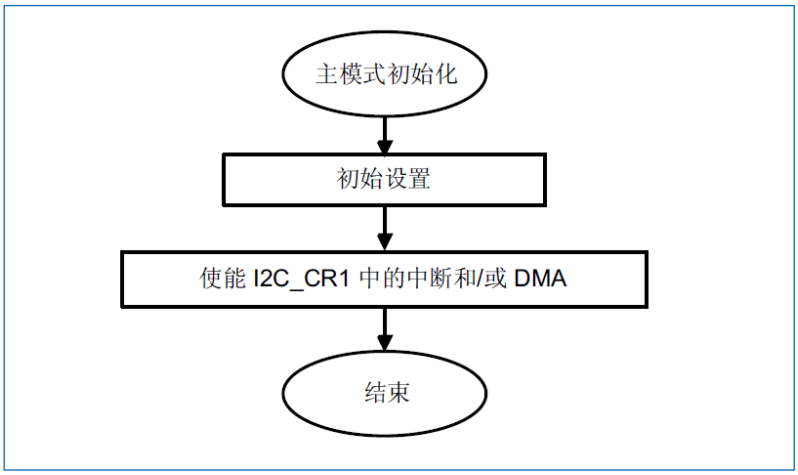


图 22-15 初始化流程图

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 I2C_CR2 寄存器中的 HEAD10R 位清零来发送完整的读序列。在这种情况下，主器件会在 START 位置 1 后自动发送以下完整序列：（重复）起始位+带写方向的从器件 10 位地址头字节+从器件地址第 2 个字节+重复起始位+带读方向的从器件 10 位地址头字节

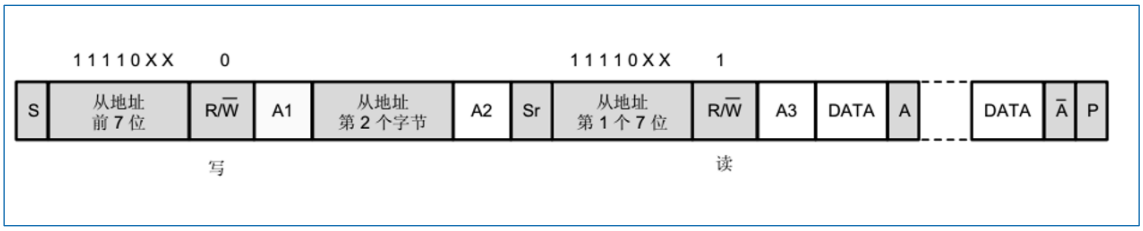


图 22-16 10 位地址读访问（HEAD10R=0）

- 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。

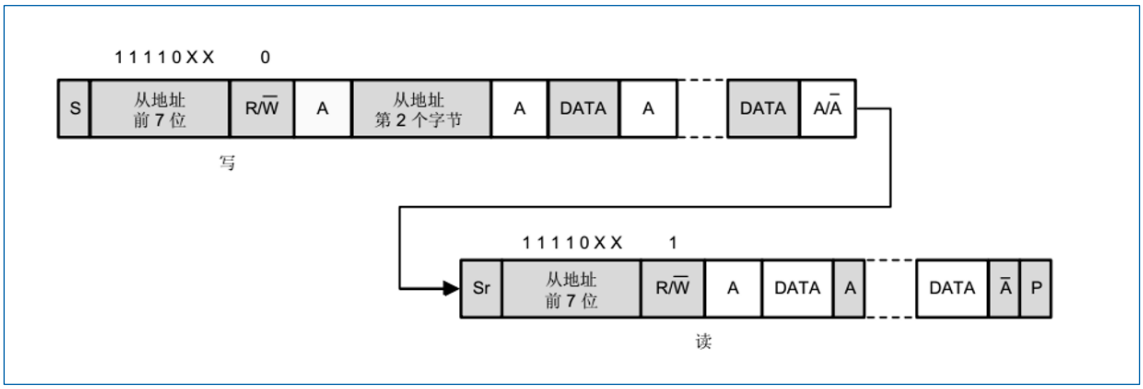


图 22-17 10 位地址读访问（HEAD10R=1）

主发送器

写传输时，在发送完每个字节（即第 9 个 SCL 脉冲（接收到 ACK 时））后，TXIS 标志将置 1。

如果 I2C_CR1 寄存器中的 TXIE 位置 1，TXIS 事件将生成中断。当 I2C_TXDR 寄存器中写入待发送的下一个数据字节时，该标志将被清零。

传输期间的 TXIS 事件的数量对应于 NBYTES[7:0]中编程的值。如果待发送的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES 数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延展，直到 NBYTES[7:0]被写入非零值。

收到 NACK 时，TXIS 标志不会置 1。

- 当 RELOAD=0 且 NBYTES 数据传输完成时：
  - 在自动结束模式（AUTOEND=1）下，将自动发送停止位。
  - 在软件结束模式（AUTOEND=0）下，TC 标志将置 1 且 SCL 线的低电平将被延展，以便执行以下软件操作：

  可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位。

  可通过将 I2C_CR2 寄存器中的 STOP 位置 1 来请求停止位。将 STOP 位置 1 会将 TC 标志清零，并在总线上发送停止位。
- 如果接收到 NACK：TXIS 标志不会置 1，并且接收到 NACK 后会自动发送停止位。I2C_ISR 寄存器中的 NACKF 标志置 1，如果 NACKIE 位置 1，还将生成中断。

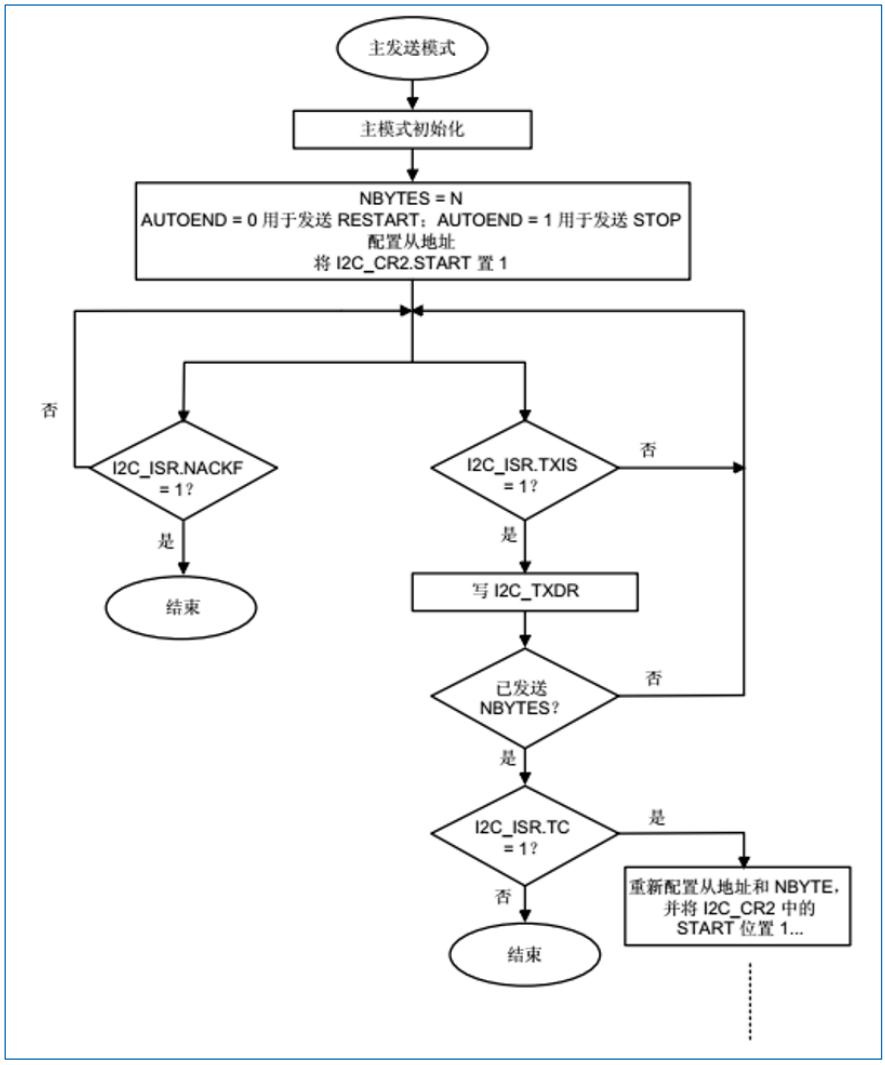


图 22-18 I2C 主发送器的传输序列流程图（N≤255 字节）

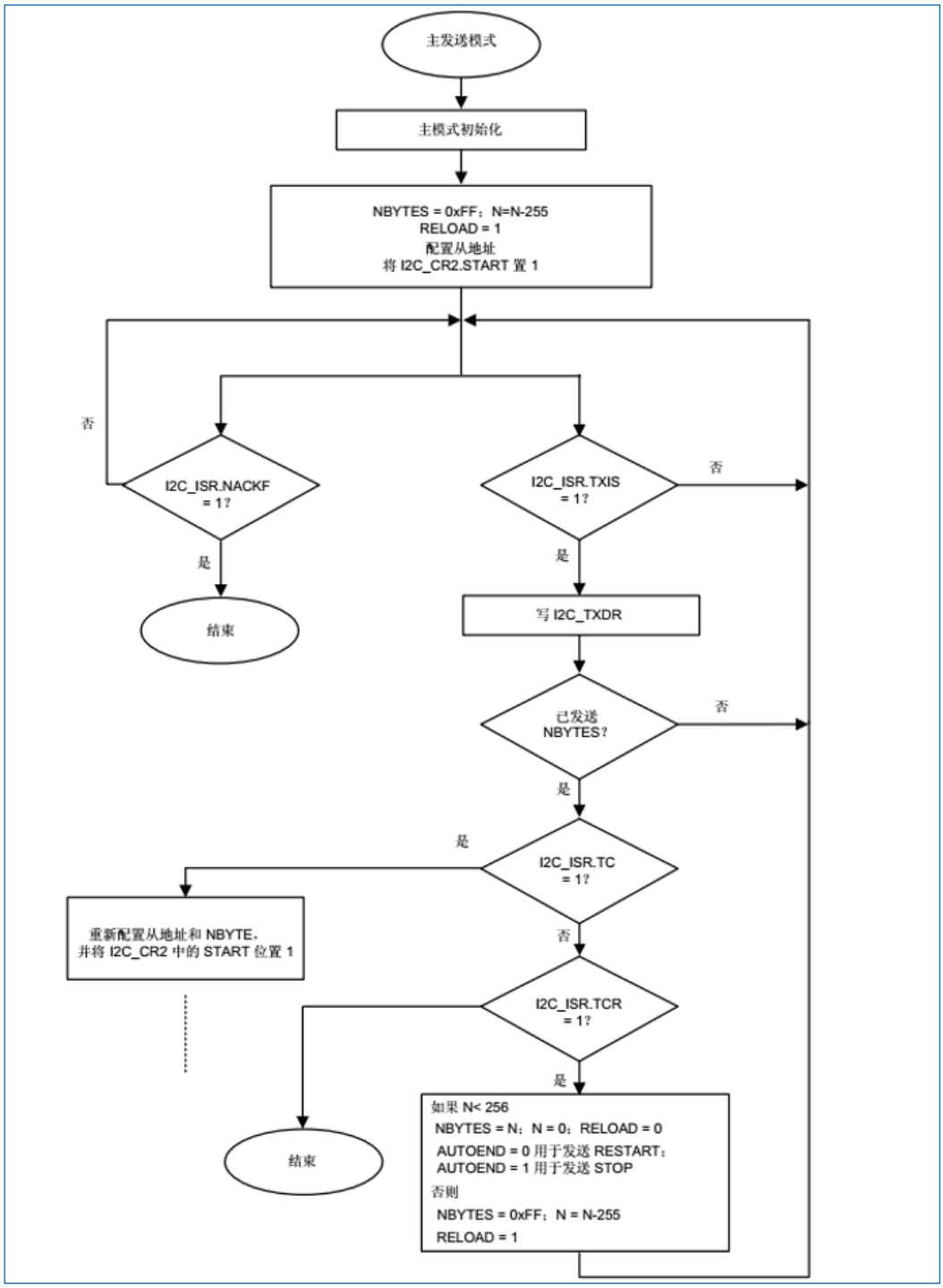


图 22-19 I2C 主发送器的传输序列流程图（N>255 字节）

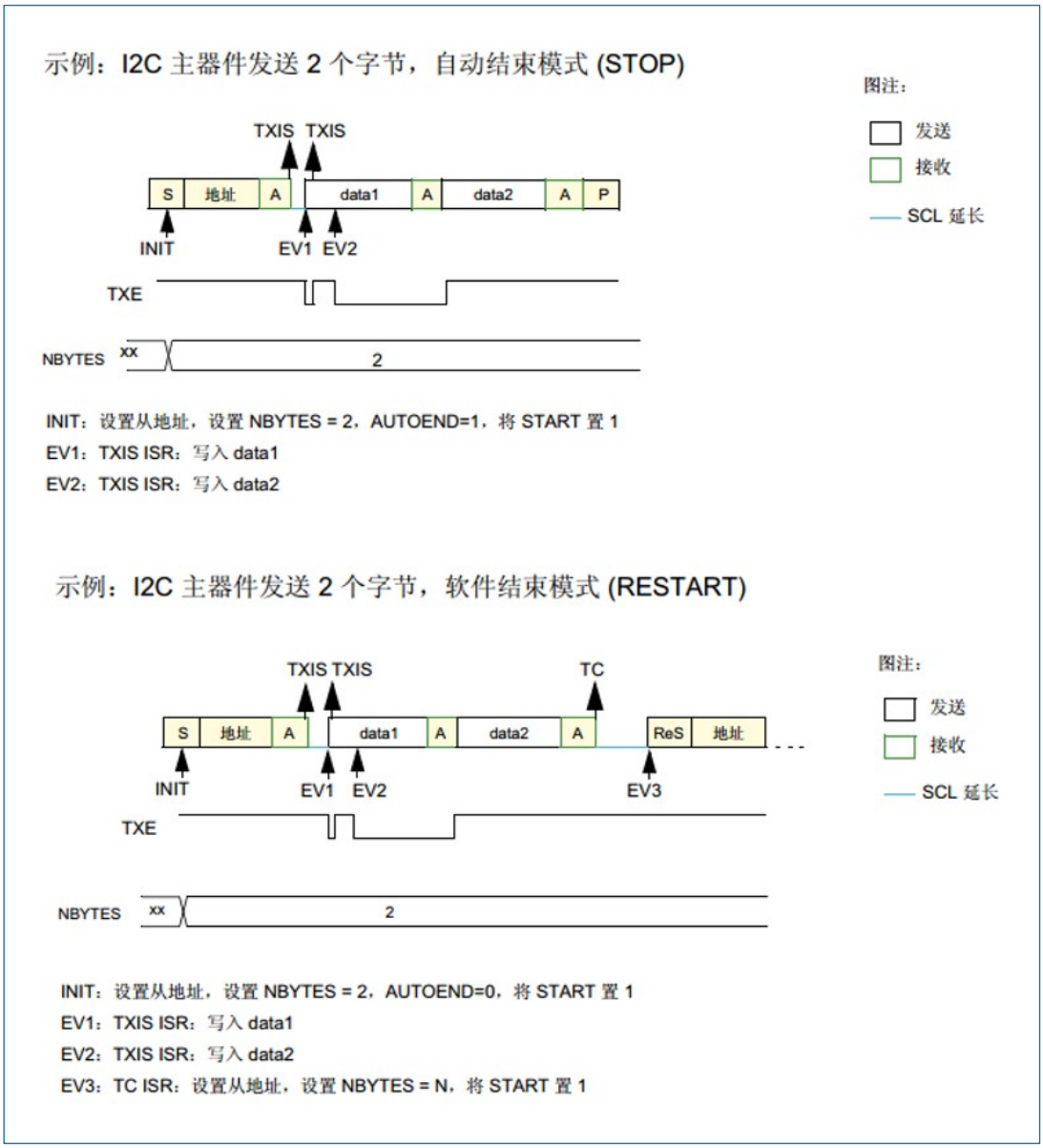


图 22-20 I2C 主发送器的传输总线图

主接收器

读传输时，在接收到每个字节（即第 8 个 SCL 脉冲）后，RXNE 标志将置 1。如果 I2C_CR1 寄存器中的 RXIE 位置 1，RXNE 事件将生成中断。读取 I2C_RXDR 时，将清零该标志。

如果待接收的数据字节总数大于 255，则必须通过将 I2C_CR2 寄存器中的 RELOAD 位置 1 来选择重载模式。在这种情况下，当 NBYTES[7:0]数据传输完成时，TCR 标志将置 1，并且 SCL 线的低电平将被延展，直到 NBYTES[7:0]被写入非零值。

- 当 RELOAD=0 且 NBYTES[7:0]数据传输完成时：
  - 在自动结束模式（AUTOEND=1）下，接收到最后一个字节后，将自动发送 NACK 和停止位。
  - 在软件结束模式（AUTOEND=0）下，接收到最后一个字节后，将自动发送 NACK，TC 标志将置 1 且 SCL 线的低电平将被延展，以便执行以下软件操作：

  可通过将 I2C_CR2 寄存器中的 START 位置 1 并配置适当的从地址和待传输字节数来请求发送重复起始位。将 START 位置 1 会将 TC 标志清零，并在总线上发送起始位，后跟从地址。

  可通过将 I2C_CR2 寄存器中的 STOP 位置 1 来请求停止位。将 STOP 位置 1 会将 TC 标志清零，并在总线上发送停止位。

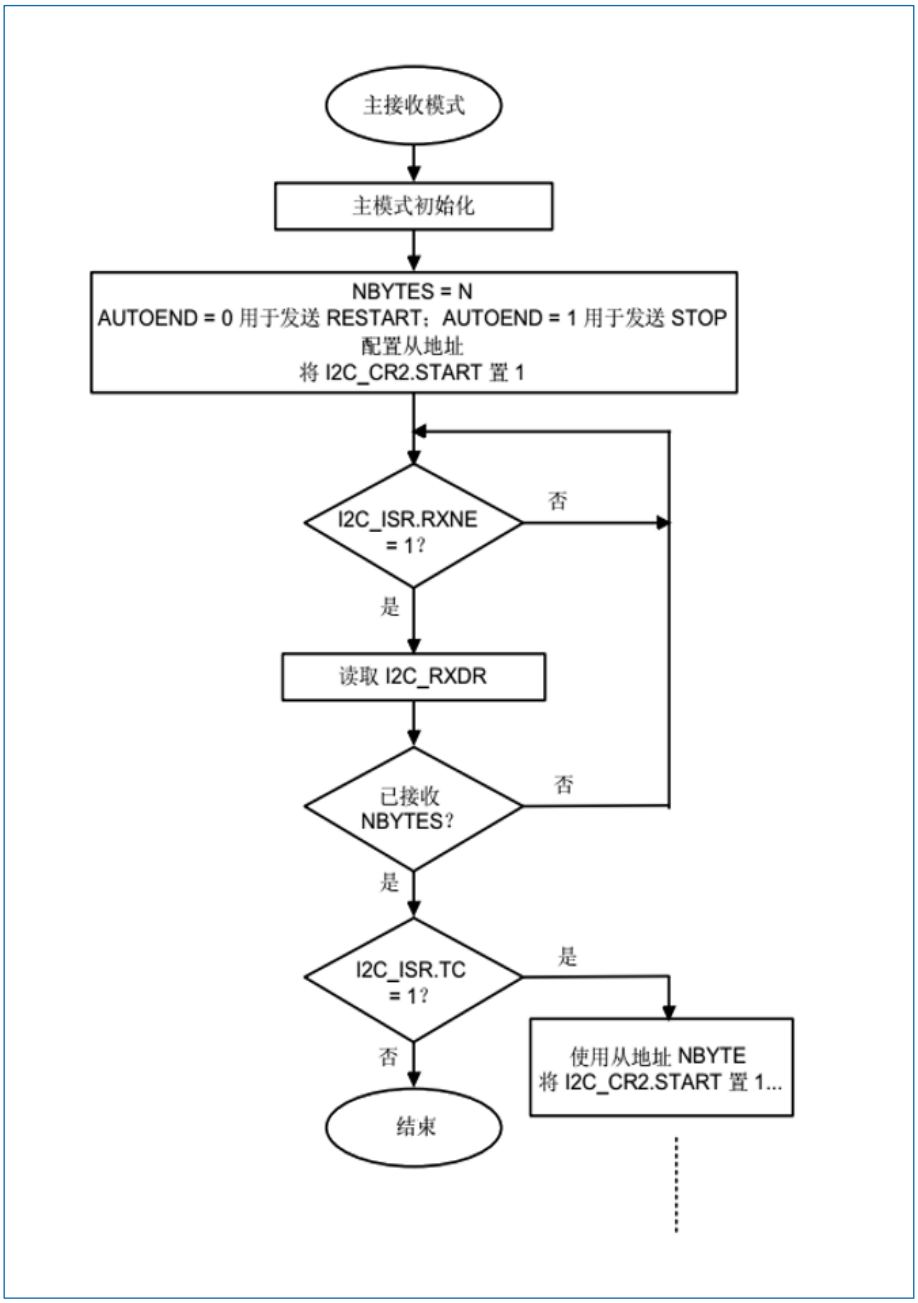


图 22-21 I2C 主接收器的传输序列流程图（N≤255 字节）

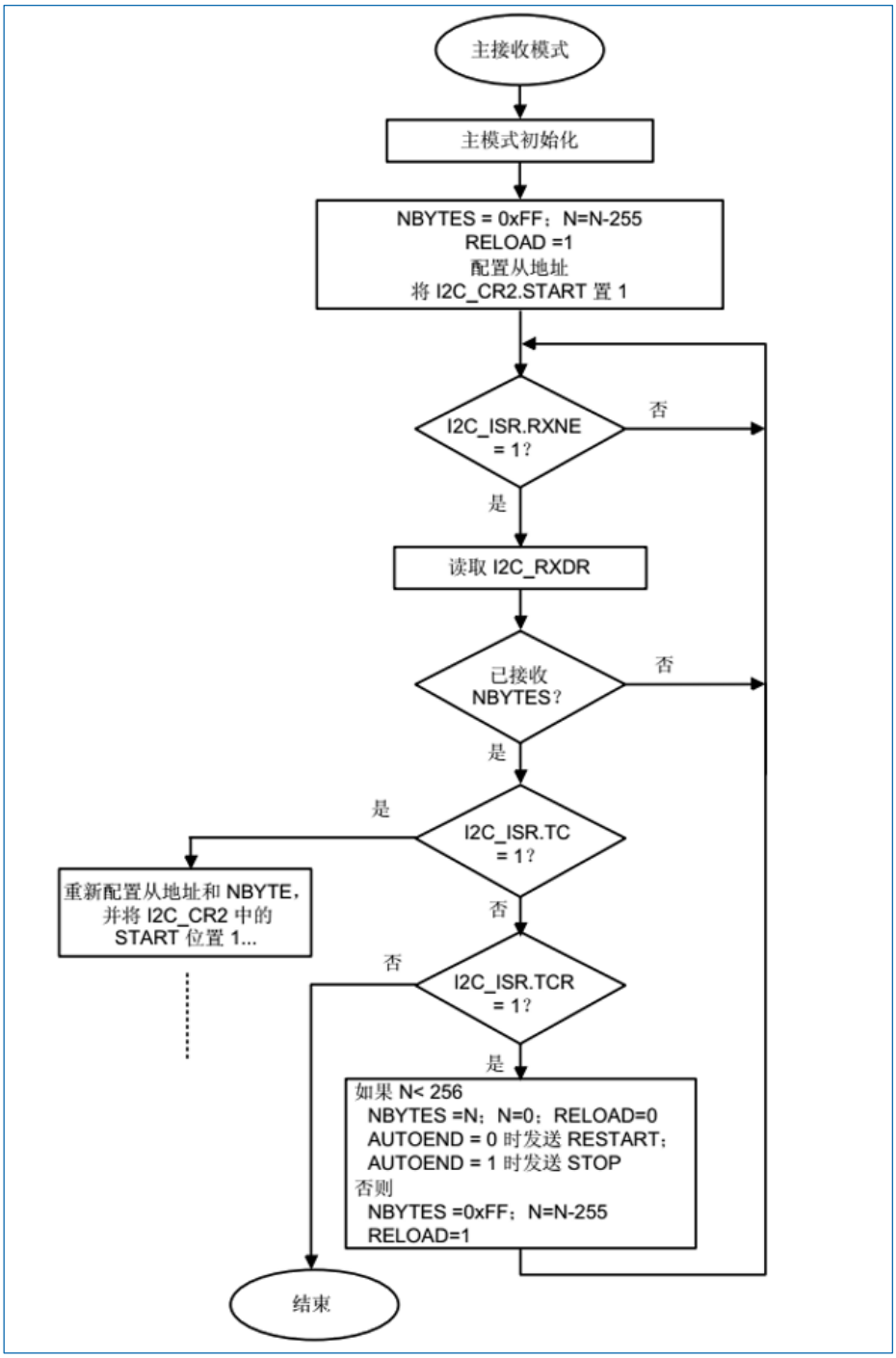


图 22-22 I2C 主接收器的传输序列流程图（N>255 字节）

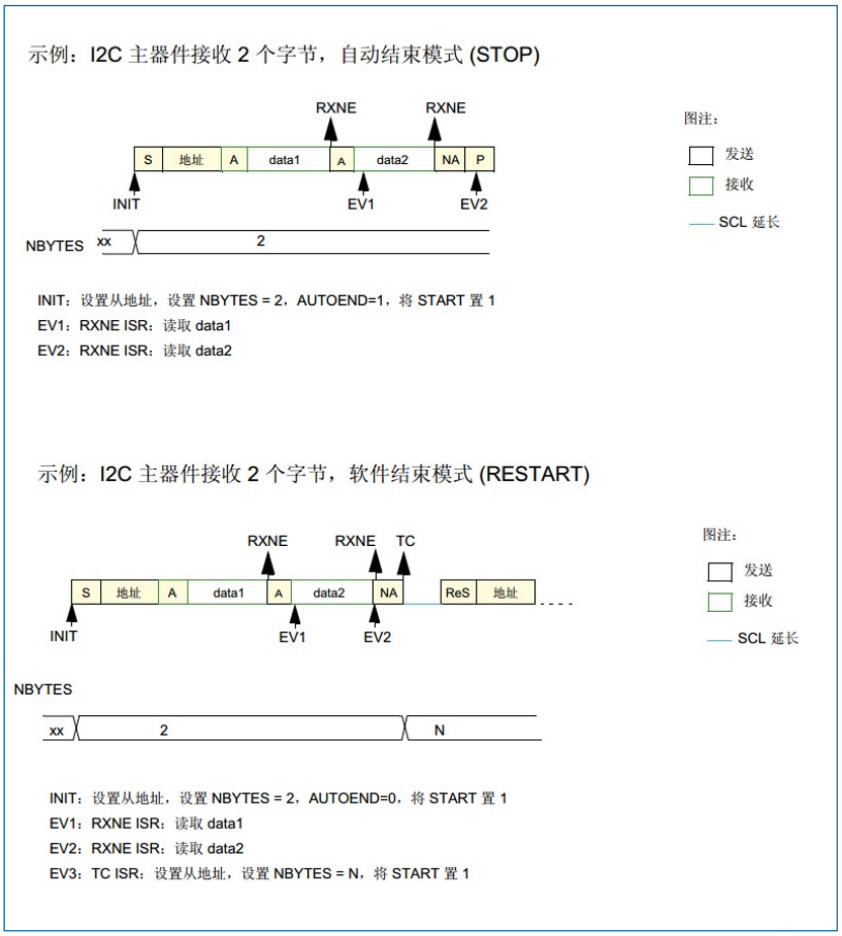


图 22-23 I2C 主接收器的传输总线图

22.2.9 I2C_TIMINGR 寄存器配置示例

下文各表提供了相应示例，以介绍如何编程 I2C_TIMINGR 才能获得符合 I2C 规范的时序。

表 22-5 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+)
	10kHz	100kHz	400kHz	500kHz
PRESC	1	1	0	0
SCLL	0xC7	0x13	0x9	0x6
t_{SCLL}	200x250ns=50 µs	20x250ns=5.0 µs	10x125ns=1250ns	7x125ns=875ns
SCLH	0xC3	0xF	0x3	0x3
t_{SCLH}	196x250ns=49 µs	16x250ns=4.0 µs	4x125ns=500ns	4x125ns=500ns
t_{SCL}(1)	约 100 µs(2)	约 10 µs(2)	约 2500ns(3)	约 2000ns(4)
SDADEL	0x2	0x2	0x1	0x0
t_{SDADEL}	2x250ns=500ns	2x250ns=500ns	1x125ns=125ns	0ns
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 tSCLL +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 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=1000ns 时的示例。
(3). t_{SYNC1}+t_{SYNC2} 最小值为 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=750ns 时的示例。
(4). t_{SYNC1}+t_{SYNC2} 最小值为 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=655ns 时的示例。

表 22-6 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+）
	10kHz	100kHz	400kHz	1000kHz
PRESC	3	3	1	0
SCLL	0xC7	0x13	0x9	0x4
t_{SCLL}	200x250ns=50 µs	20x250ns=5.0 µs	10x125ns=1250ns	5x62.5ns=312.5ns
SCLH	0xC3	0xF	0x3	0x2
t_{SCLH}	196x250ns=49 µs	16x250ns=4.0µs	4x125ns=500ns	3x62.5ns=187.5ns
t_{SCL}（1）	约 100 µs(2)	约 10 µs(2)	约 2500ns(3)	约 1000ns(4)
SDADEL	0x2	0x2	0x2	0x0
t_{SDADEL}	2x250ns=500ns	2x250ns=500ns	2x125ns=250ns	0ns
SCLDEL	0x4	0x4	0x3	0x2
t_{SCLDEL}	5x250ns=1250ns	5x250ns=1250ns	4x125ns=500ns	3x62.5ns=187.5ns

(1). 由于 SCL 内部检测存在延时，SCL 周期 t_{SCL} 大于 tSCLL +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 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=1000ns 时的示例。
(3). t_{SYNC1}+t_{SYNC2} 最小值为 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=750ns 时的示例。
(4). t_{SYNC1}+t_{SYNC2} 最小值为 4x t_{I2CCLK} =500ns。t_{SYNC1}+t_{SYNC2}=655ns 时的示例。

22.2.10 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 SMBuss 规范第 2.0 版兼容，该版本以 I2C 规范第 2.1 版为基础。SMBus 包括三类器件。

- 从器件，用于接收或响应命令。
- 主器件，用于发出命令、生成时钟和管理传输。
- 主机，专用的主器件，可提供连接系统 CPU 的主接口。主机必须具有主-从器件功能，并且必须支持 SMBus 主机通知协议。SMBus 系统中只允许存在一个主机。

SMBus 可配置为主器件或从器件，也可配置为主机。

总线协议

SMBus 协议包含 11 种可用命令协议。SMBus 器件可以使用任意一种协议进行通信。这 11 种协议分别为快速命令、发送字节、接收字节、写入字节、写入字、读取字节、读取字、过程调用、块读取、块写入以及块写入-块读取过程调用。这些协议由用户通过软件实现。

地址解析协议（ARP）

动态为新器件分配唯一地址可解决 SMBus 从地址冲突的问题。为了提供一种机制来针对地址分配隔离各个器件，各器件必须具有唯一的器件标识符（UDID）。该 128 位数字由软件实现。

该外设支持地址解析协议（ARP）。通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默

认地址（0b110 0001）。ARP 命令应通过用户软件实现。

在从模式下，通过仲裁来实现对 ARP 的支持。

有关 SMBus 地址解析协议的详细信息，请参见 SMBus 规范第 2.0 版。

接收的命令和数据应答控制

SMBus 接收器必须能够对接收到的每个命令或数据进行否定应答。要在从模式下实现 ACK 控制，必须通过将 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。

主机通知协议

该外设通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来支持主机通知协议。在这种情况下，主机将应答 SMBus 主机地址（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 I2C_CR1 寄存器中的 ALERTEN 位置 1 来将 SMBA 引脚拉为低电平。这同时还会使能报警响应地址。

如果配置为主机（SMBHEN=1），当 SMBA 引脚上检测到下降沿且 ALERTEN=1 时，I2C_ISR 寄存器中的 ALERT 标志置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。当 ALERTEN=0 时，即使外部 SMBA 引脚为低电平，也不会产生中断标志。

如果无需 SMBus ALERT 引脚，则当 ALERTEN=0 时，SMBA 引脚可用作标准 GPIO。

数据包错误校验

SMBus 规范中引入了数据包错误校验机制来实现可靠的数据传输。具体的实施方式是在每次数据传输结束时，附加数据包错误校验码（PEC）。PEC 的计算方式是对 START 到 STOP 之间的所有字节（包括地址和读/写位）使用 CRC-8 多项式 C（x）=x^8+x^2+x+1 进行计算。

内置的硬件 PEC 计算器：

在接收端：接收到的最后一个字节数据为 PEC 值，如果与硬件计算的 PEC 不匹配，将自动发送 NACK。

在发送端：发送的最后一个字节数据为 PEC 值。

超时

该外设内置了硬件定时器，以便符合 SMBus 规范第 2.0 版中定义的 3 个超时。

表 22-7 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
$t_{LOW:SEXT}$(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。

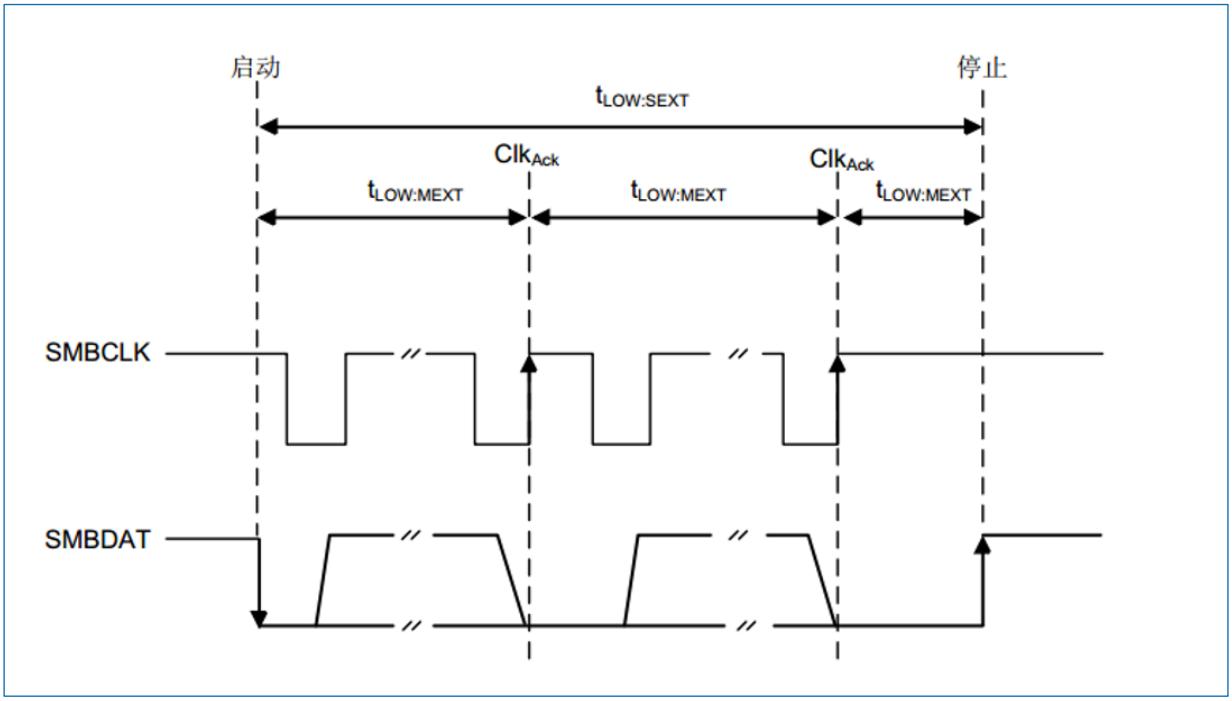


图 22-24 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的超时间隔

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。

22.2.11 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 I2C_CR1 寄存器中的 SBC 位置 1 来使能从字节控制模式。

特定地址（从模式）

SMBus 作为从设备时，包含以下 3 个特殊地址。这 3 个地址的使能方法如下：

- 通过将 I2C_CR1 寄存器中的 SMBDEN 位置 1 来使能 SMBus 器件默认地址（0b110 0001）。
- 通过将 I2C_CR1 寄存器中的 SMBHEN 位置 1 来使能 SMBus 主机地址（0b000 1000）。
- 通过将 I2C_CR1 寄存器中的 ALERTEN 位置 1 来使能报警响应地址（0b000 1100）。

数据包错误校验

通过将 I2C_CR1 寄存器中的 PECEN 位置 1 来使能 PEC 的计算。然后，借助硬件字节计数器（I2C_CR2 寄存器中的 NBYTES[7:0]）来管理 PEC 传输。使能 I2C 之前，必须配置 PECEN 位。

PEC 传输由硬件字节计数器来管理，因此在从模式下连接 SMBus 时必须将 SBC 位置 1。当 PECBYTE 位置 1 且 RELOAD 位清零时，传输完 NBYTES-1 字节的数据后会传输 PEC。如果 RELOAD 置 1，PECBYTE 将不起作用。

注意：使能 I2C 时，不允许更改 PECEN 配置。

表 22-8 带 PEC 的 SMBus 配置

模式	SBC 位	RELOAD 位	AUTOEND 位	PECBYTE 位
主 Tx/RxNBYTES+PEC+STOP	x	0	1	1
主 Tx/RxNBYTES+PEC+ReSTART	x	0	0	1
从 Tx/Rx+PEC	1	0	x	1

超时检测

将 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须按如下方式编程：即在 SMBus 规范第 2.0 版规定的时间最大值之前检测出超时情况。

- $t_{TIMEOUT}$ 检查

  要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0]位编程为定时器重载值，以检查 $t_{TIMEOUT}$ 参数。必须将 TIDLE 位配置为“0”，以检测 SCL 低电平超时。

  然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 位置 1 来使能定时器。

  如果 SCL 的低电平持续时间超过（TIMEOUTA+1）x 2048x t_{I2CCLK}，I2C_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 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。

如果 SMBus 外设延展 SCL 的累积时间超过（TIMEOUTB+1）x 2048x t_{I2CCLK}，并且达到“22.2.10SMBus I2C 特性”中的“总线空闲检测”一节给出的超时间隔，则 I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。请参见表 22-8。

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 I2C_TIMEOUTR 寄存器中的 TIMEOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过（TIMEOUTA+1）x 4x t_{I2CCLK}，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1。请参见表 22-9。

注意：TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

22.2.12 SMBus：I2C_TIMEOUTR 寄存器配置示例

- 将 $t_{TIMEOUT}$ 的最大持续时间配置为 25ms

表 22-9 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$=25ms）

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	$t_{TIMEOUT}$
8MHz	0x61	0	1	98x2048x125ns=25ms

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	$t_{TIMEOUT}$
16MHz	0xC3	0	1	196x2048x62.5ns=25ms
32MHz	0x186	0	1	391x2048x31.25ns=25ms

- 将 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的最大持续时间配置为 8ms

表 22-10 不同 I2CCLK 频率下的 TIMEOUTB 设置示例

f_{I2CCLK}	TIMEOUTB[11:0]位	TEXTEN 位	$t_{LOW:EXT}$
8MHz	0x1F	1	32x 2048x 125ns=8ms
16MHz	0x3F	1	64x 2048x 62.5ns=8ms
32MHz	0x7C	1	125x 2048x 31.25ns=8ms

- 将 T_{IDLE} 的最大持续时间配置为 50 µs

表 22-11 不同 I2CCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE}=50 µs）

f_{I2CCLK}	TIMEOUTA[11:0]位	TIDLE 位	TIMEOUTEN 位	t_{TIDLE}
8MHz	0x63	1	1	100x 4x 125ns=50 µs
16MHz	0xC7	1	1	200x 4x 62.5ns=50 µs
32MHz	0x18F	1	1	400x 4x 31.25ns=50 µs

22.2.13 SMBus 模式

除了 I2C 模式传输管理之外，还提供了一些额外的软件流程图来支持 SMBus。

SMBus 从发送器

在 SMBus 模式下，必须将 SBC 编程为“1”，以便在完成已编程数据字节数的传输后进行 PEC 传输。当 PECBYTE 位置 1 时，NBYTES[7:0]中编程的字节数包含 PEC 传输。在这种情况下，总 TXIS 中断数为 NBYTES-1，如果主器件在完成 NBYTES-1 字节的数据传输后请求传输额外的字节，则将自动发送 I2C_PECR 寄存器的内容。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

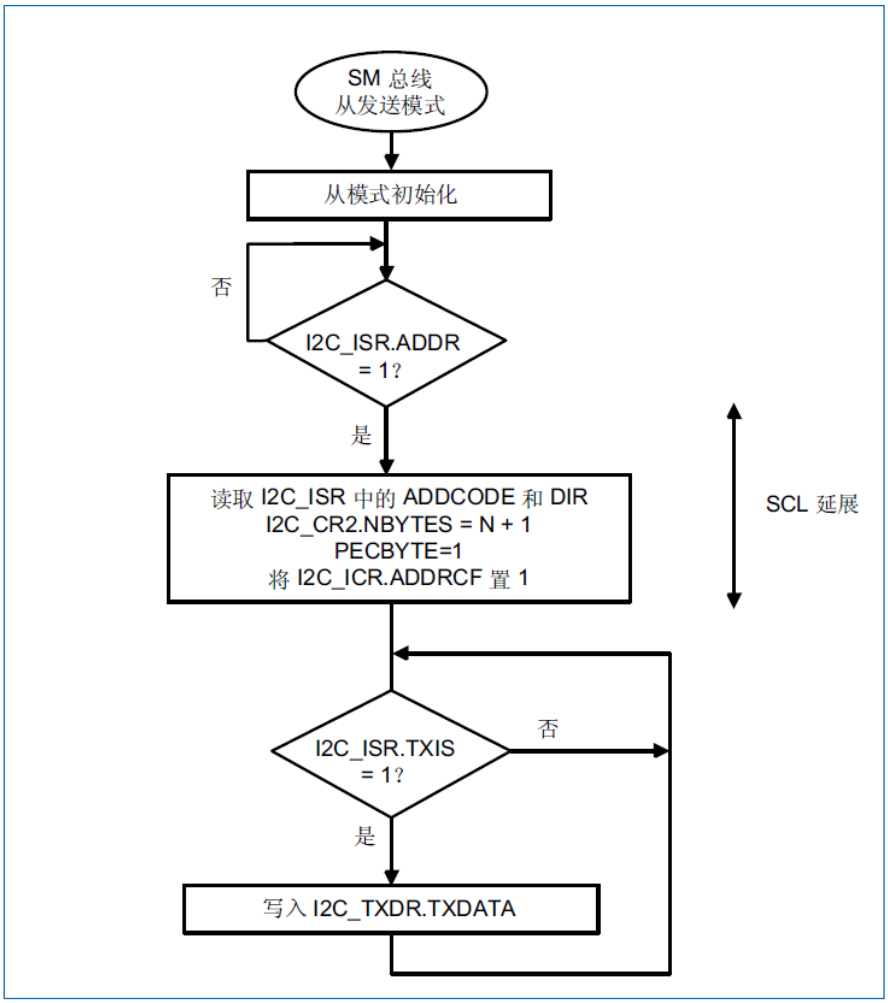


图 22-25 SMBus 从发送器的传输序列流程图（N 字节+PEC）

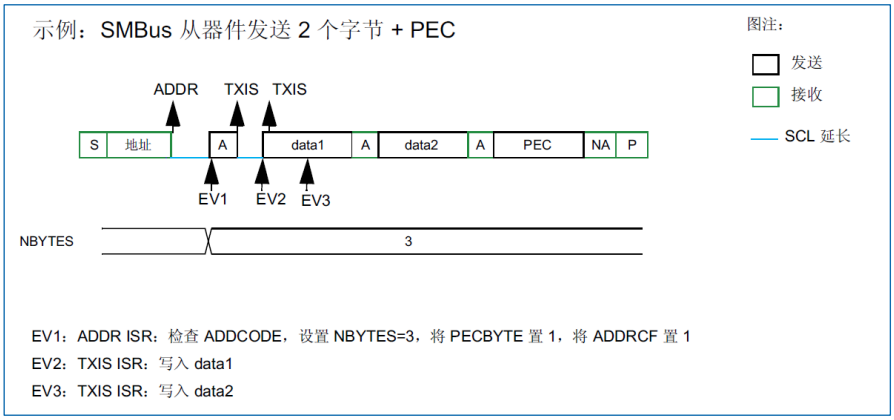


图 22-26 SMBus 从发送器的传输总线图（SBC=1）

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“22.2.7 从模式”中的“从器件字节控制模式”。

要校验 PEC 字节，必须将 RELOAD 位清零并将 PECBYTE 位置 1。在这种情况下，当接收到 NBYTES-1 字节的数据后，接收的下一个字节将与内部 I2C_PECR 寄存器的内容作比较。如果比较不匹配，则将自动生成 NACK 信号；如果比较匹配，则将自动生成 ACK 信号，而与 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 I2C_RXDR 寄存器中，并且 RXNE 标志将置 1。

当 PEC 不匹配时，PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

如果无需 ACK 软件控制，用户可编程 PECBYTE=1，在同一写操作下，将 NBYTES 编程为连续接收的字节数。接收到 NBYTES-1 字节的数据后，会将接收的下一个字节视为 PEC 进行校验。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

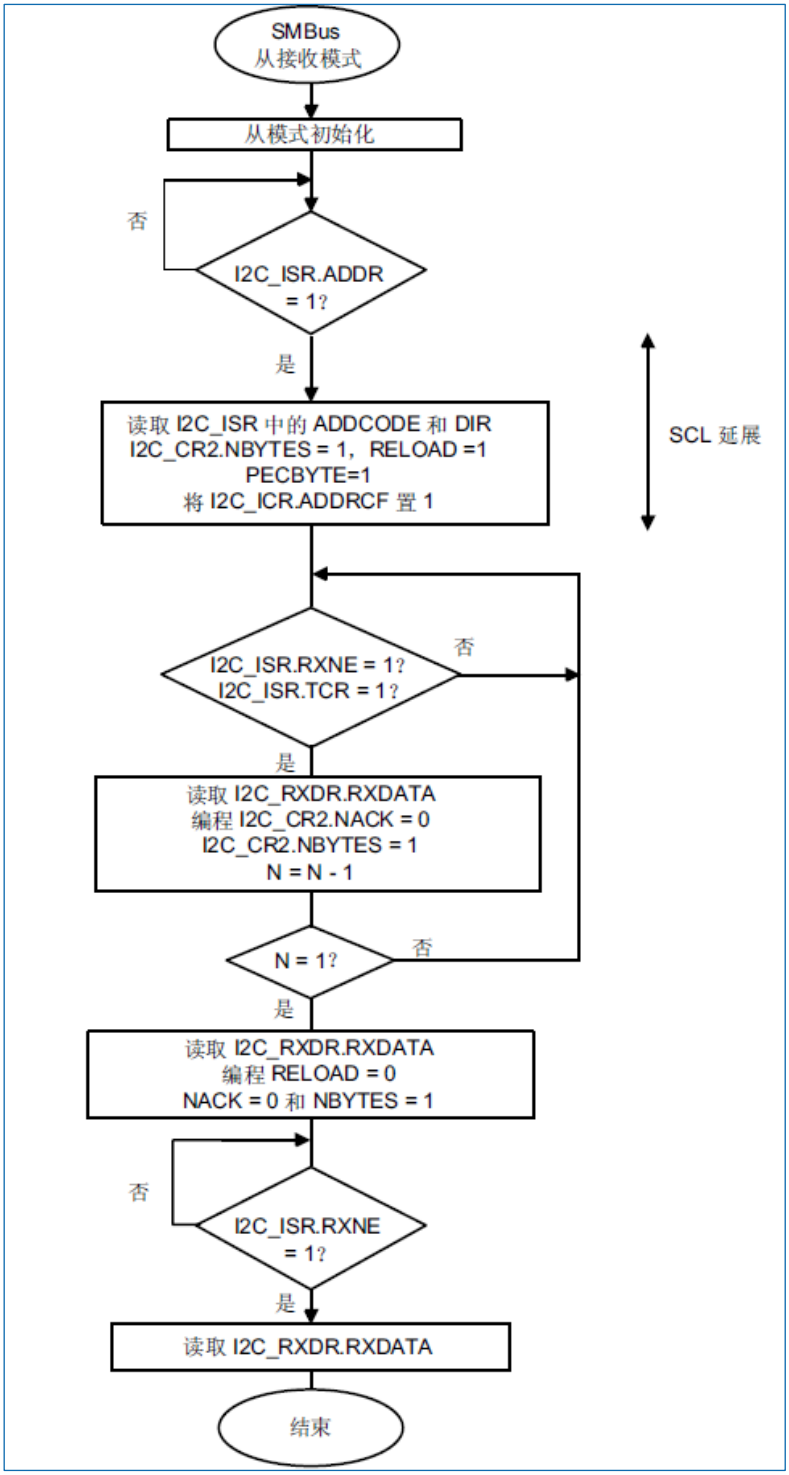


图 22-27 SMBus 从接收器的传输序列流程图（N 字节+PEC）

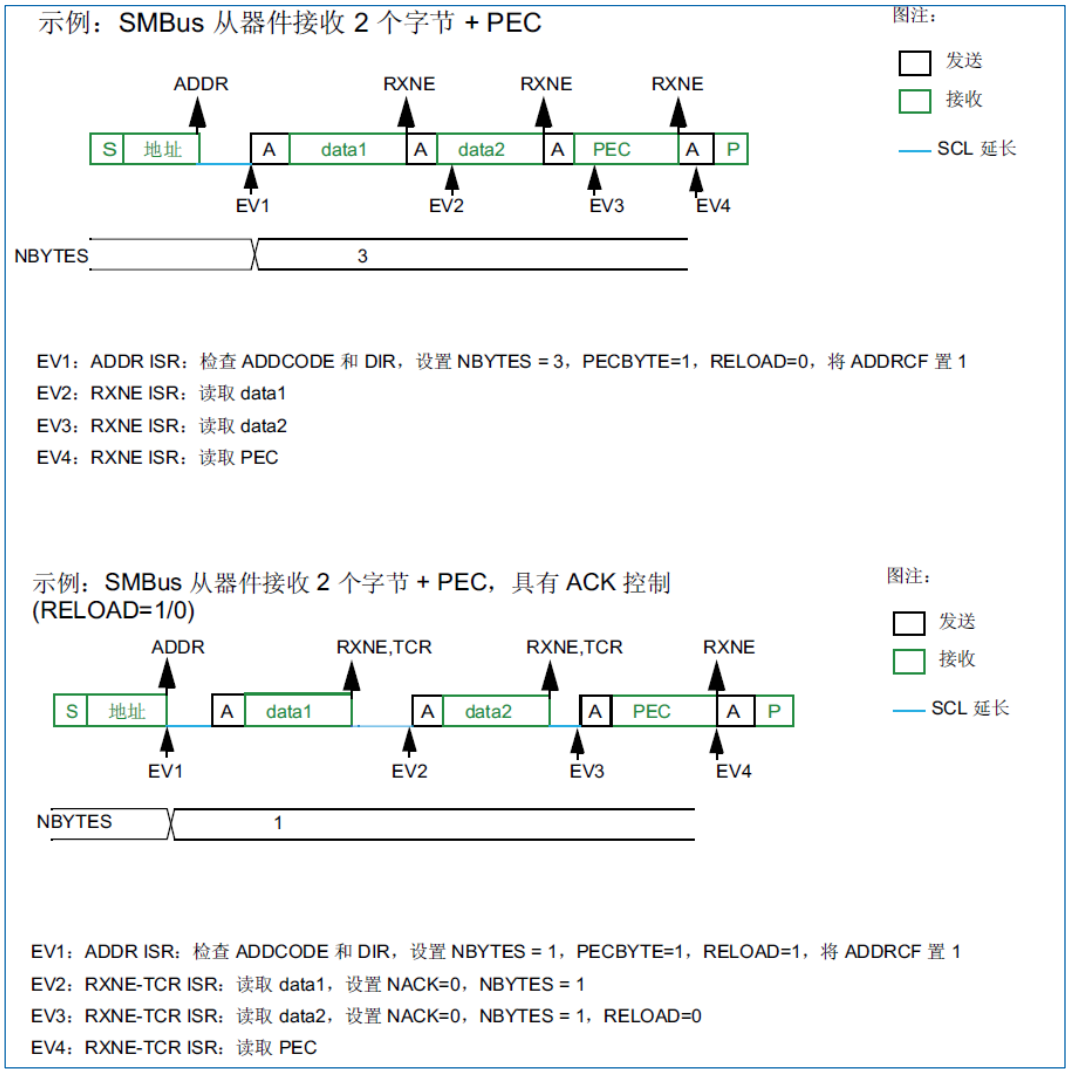


图 22-28 从接收器的总线传输图（SBC=1）

SMBus 主发送器

当 SMBus 主器件想要发送 PEC 时，必须在 START 位置 1 前，将 PECBYTE 位置 1 并在 NBYTES[7:0]字段中设置字节数。在这种情况下，总 TXIS 中断数为 NBYTES-1。因此，如果 PECBYTE 位在 NBYTES=0x1 时置 1，则将自动发送 I2C_PECR 寄存器的内容。

如果 SMBus 主器件想要在 PEC 后发送停止位，则应选择自动结束模式（AUTOEND=1）。在这种情况下，传输 PEC 后将自动发送停止位。

如果 SMBus 主器件想要在 PEC 后发送重复起始位，则必须选择软件模式（AUTOEND=0）。

在这种情况下，发送 NBYTES-1 字节的数据后，将发送 I2C_PECR 寄存器的内容，TC 标志将在传输完 PEC 之后置 1，SCL 线的低电平时间将延长。必须在 TC 中断子程序中设置重复起始位。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

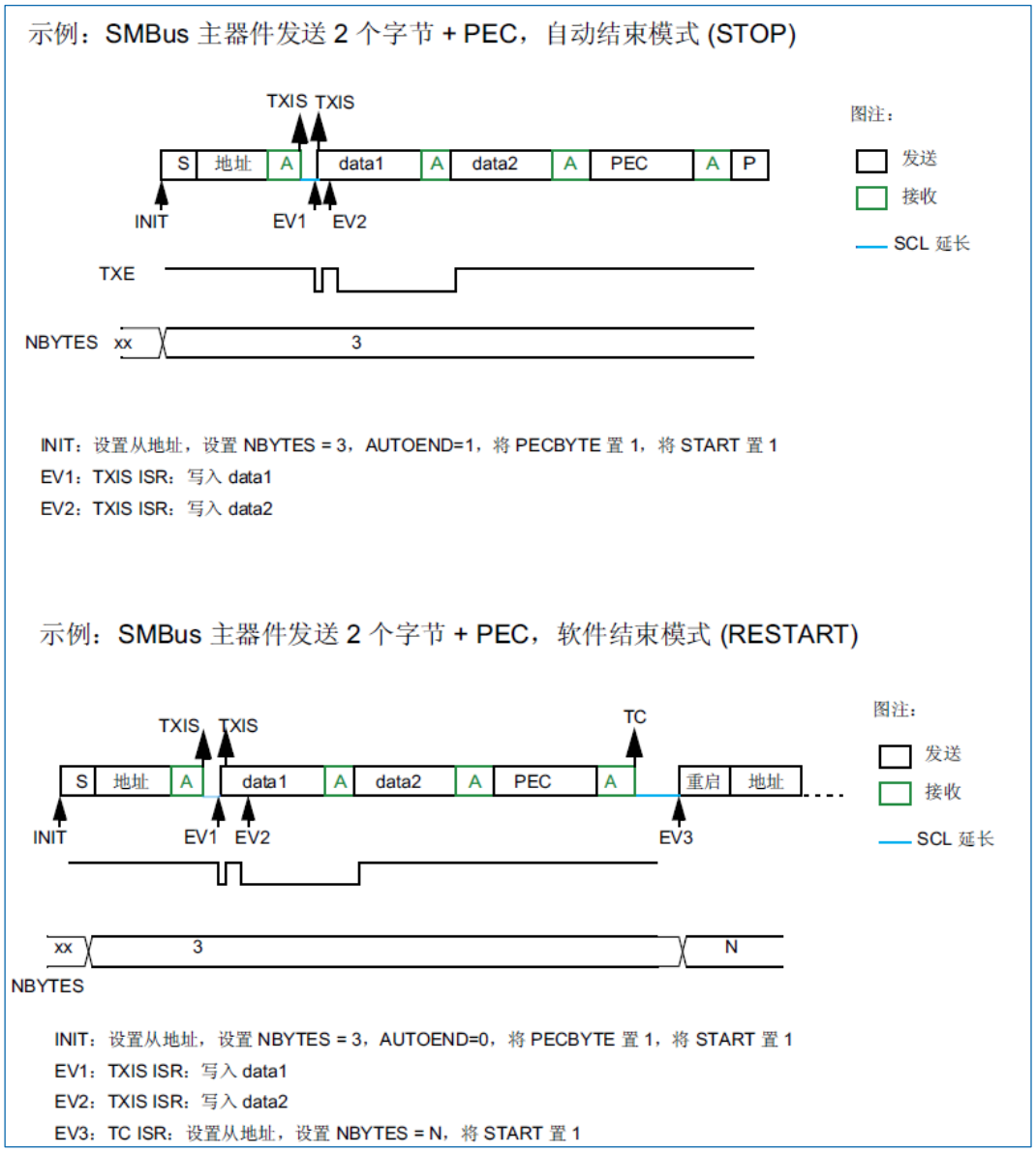


图 22-29 SMBus 主发送器的总线传输图

SMBus 主接收器

当 SMBus 主器件想要接收 PEC，并在传输结束后接收 STOP 时，可选择自动结束模式（AUTOEND=1）。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。在这种情况下，当接收到 NBYTES-1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。PEC 字节（其后跟有停止位）将得到 NACK 响应。

当 SMBus 主接收器想要接收 PEC 字节，并且在传输结束后接收重复起始位时，必须选择软件模式（AUTOEND=0）。将 START 位置 1 之前，必须将 PECBYTE 位置 1 并设置从地址。

在这种情况下，当接收到 NBYTES-1 字节的数据后，将自动使用 I2C_PECR 寄存器的内容对接收的下一个字节进行校验。接收到 PEC 字节后，TC 标志将置 1，SCL 线的低电平时间将延长。可以在 TC 中断子程序中设置重复起始位。

注意：当 RELOAD 位置 1 时，PECBYTE 位将不起作用。

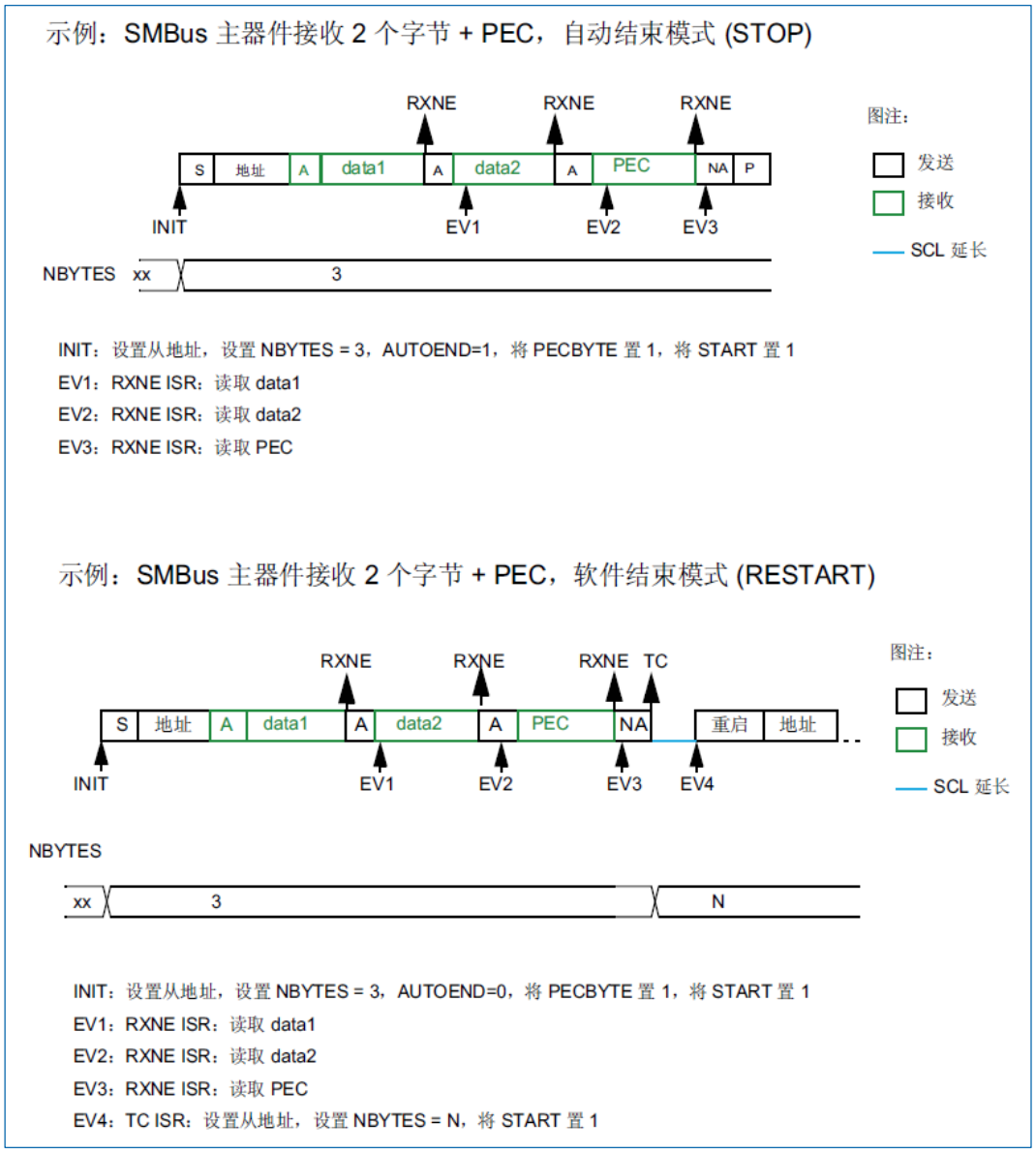


图 22-30 SMBus 主接收器的总线传输图

22.2.14 错误条件

以下错误条件可能导致通信失败。

总线错误（BERR）

当 SDA 边沿出现且 SCL 为高电平时，会检测到起始或停止位。当检测到起始位或停止位但不位于第 9N 个 SCL 时钟脉冲之后时，则认为出现了总线错误。

只有当 I2C 在传输过程中用作主器件或被寻址为从器件时（即未处于从模式下的地址阶段），才会将总线错误标志置 1。

在从模式下误检测到起始位或重复起始位时，I2C 会像接收到正确的起始位一样进入地址识别状态。

检测到总线错误时，I2C_ISR 寄存器中的 BERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

仲裁丢失（ARLO）

当 SDA 线上发送高电平但在 SCL 上升沿却采样到低电平时，会检测到仲裁丢失。

- 在主模式下，将在地址阶段、数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，SDA 线和 SCL 线被释放，起始控制位由硬件清零，主器件自动切换为从模式。

- 在从模式下，将在数据阶段和数据应答阶段检测到仲裁丢失。在这种情况下，传输停止，SCL 和 SDA 线被释放。
- 检测到仲裁丢失时，I2C_ISR 寄存器中的 ARLO 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

上溢/下溢错误（OVR）

当满足 NOSTRETCH=1 和以下条件时，将在从模式下检测到上溢或下溢错误：

- 在接收过程中接收到一个新的字节，但 RXDR 寄存器的值还未被读取。接收的新字节丢失，自动发送 NACK 来响应新字节。
- 在发送过程中：
  - 当 STOPF=1 且应发送第一个数据字节时。TXE=0 时发送 I2C_TXDR 寄存器的内容，否则发送 0xFF。
  - 应发送一个新字节但尚未向 I2C_TXDR 寄存器写入数据时，将发送 0xFF。
  - 检测到上溢或下溢错误时，I2C_ISR 寄存器中的 OVR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

数据包错误校验错误（PECERR）

当接收到的 PEC 字节与 I2C_PECR 寄存器的内容不匹配时，将检测到 PEC 错误。接收到错误的 PEC 后，将自动发送 NACK。

检测到 PEC 错误时，I2C_ISR 寄存器中的 PECERR 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

超时错误（TIMEOUT）

满足以下任何条件均会出现超时错误：

- TIDLE=0 且 SCL 的低电平持续时间达到 TIMEOUTA[11:0]位中定义的时间：这用于检测 SMBus 超时。
- TIDLE=1 且 SDA 和 SCL 的高电平持续时间达到 TIMEOUTA[11:0]位中定义的时间：这用于检测总线空闲情况。
- 主器件的累积时钟低电平延长时间达到了 TIMEOUTB[11:0]位中定义的时间（SMBus tLOW：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 tLOW：SEXT 参数）

当在主模式下检测到超时时，将自动发送停止位。

当在从模式下检测到超时时，将自动释放 SDA 和 SCL 线。

检测到超时错误时，I2C_ISR 寄存器中的 TIMEOUT 标志将置 1，如果 I2C_CR1 寄存器中的 ERRIE 位置 1，还将生成中断。

报警（ALERT）

当 I2C 接口配置为主机（SMBHEN=1）、使能了报警引脚检测（ALERTEN=1）并且在 SMBA 引脚上检测到下降沿时，ALERT 标志将置 1。如果 I2C_CR1 寄存器中的 ERRIE 位置 1，将生成中断。

22.2.15 DMA 请求

使用 DMA 进行发送

将 I2C_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“9 直接存储器访问控制器（DMA）”）装载进 I2C_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“22.2.8 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“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 I2C_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 I2C_RXDR 寄存器装载进由 DMA 外设配置的 SRAM 区（请参见“9 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“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。

22.2.16 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。

22.3 I2C 低功耗模式

表 22-12 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。
待机	I2C 外设掉电，退出待机模式后必须重新初始化。

22.4 I2C 中断

下表给出了 I2C 中断请求列表。

表 22-13 I2C 中断请求

中断事件	事件标志	事件标志/中断清除方法	中断使能控制位
接收缓冲区非空	RXNE	读取 I2C_RXDR 寄存器	RXIE
发送缓冲区中断状态	TXIS	写入 I2C_TXDR 寄存器	TXIE
停止位检测中断标志	STOPF	写入 STOPCF=1	STOPIE
传输完成等待重载	TCR	写入 I2C_CR2（NBYTES[7:0]≠ 0）	TCIE
传输完成	TC	写入 START=1 或 STOP=1	
地址匹配	ADDR	写入 ADDRCF=1	ADDRIE
接收到 NACK 应答	NACKF	写入 NACKCF=1	NACKIE
总线错误（Bus error）	BERR	写入 BERRCF=1	ERRIE
仲裁丢失	ARLO	写入 ARLOCF=1	
上溢/下溢（Overrun/Underrun）	OVR	写入 OVRCF=1	
PEC 错误	PECERR	写入 PECERRCF=1	
超时/t_{LOW} 错误	TIMEOUT	写入 TIMEOUTCF=1	
SMBus 报警	ALERT	写入 ALERTCF=1	

根据产品实现的不同，上述所有中断既可以共享同一个中断向量（I2C 全局中断），也可以分配到 2 个不同的中断向量上（I2C 事件中断和 I2C 错误中断）。

要使能 I2C 中断，需按照以下顺序操作：

1. 配置 NVIC 中的 I2CIRQ 通道并将其使能。
2. 配置 I2C 以生成中断。

I2C 唤醒事件连接到 EXTI 控制器，请参见“11.5 EXTI 寄存器”。

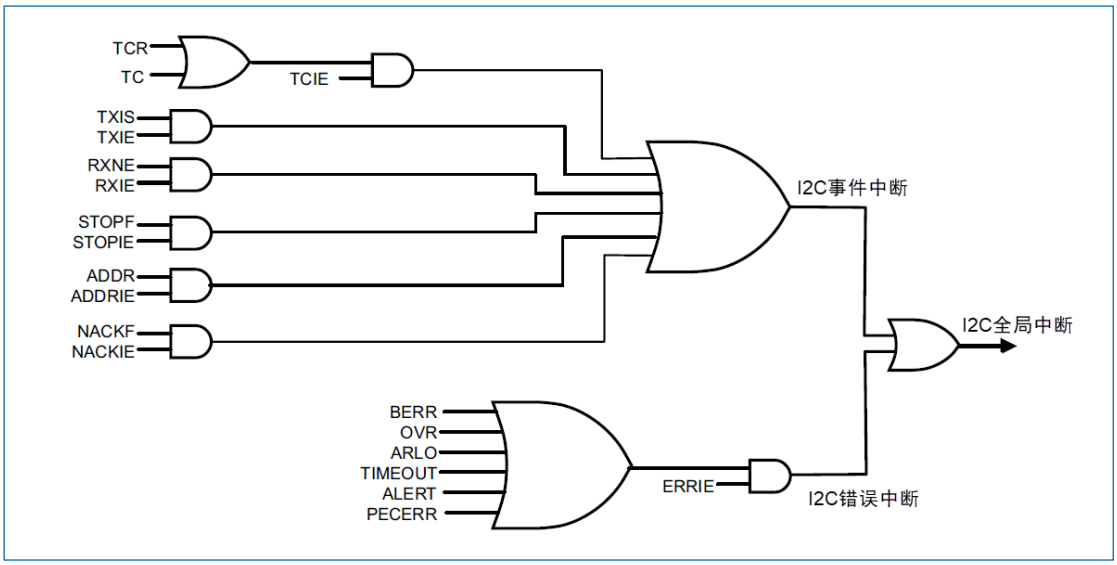


图 22-31 I2C 中断映射图

22.5 I2C 寄存器

基地址：0x4000 5400

空间大小： 0x400

22.5.1 控制寄存器 1（I2C_CR1）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	Res	NOSTRETCH	SBC
								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 31:24	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	Res：保留 必须保持复位值。

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]*TI2C_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 TI2C_CLK。 • …… • 1111：数字滤波器启用，其滤波能力达到 15 个 TI2C_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

22.5.2 控制寄存器 2（I2C_CR2）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					PECBYTE	AUTOEND	RELOAD	NBYTES[7:0]							
					rs	rw	rw	rw							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD10R	ADD10	RD_WRN	SADD[9:0]									
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 I2C_ICR 寄存器的 ADDRCF 位写 1 来清零。
	• 0：没有起始条件产生
	• 1：产生 START/RESTART 条件
	  ◦ 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 9:8	SADD[9:8]：从机地址位 9:8（主机模式）（Slave address bit 9:8 (master mode)） 在 7 位地址模式下（ADD10=0）：该位域的值不起作用。 在 10 位地址模式下（ADD10=1）：该位域应写入要发送的从机地址位的 9:8
位 7:1	SADD[7:1]：从机地址位 7:1（主机模式）（Slave address bit 7:1 (master mode)） • 在 7 位地址模式下（ADD10=0）：该位域应写入要发送的 7 位从机地址位。 • 在 10 位地址模式下（ADD10=1）：该位域应写入要发送的从机地址位的 7:1。
位 0	SADD[0]：从机地址的 0 位（主模式）（Slave address bit 0 (master mode)） • 在 7 位地址模式下（ADD10=0）：该位的值不起作用。 • 在 10 位地址模式下（ADD10=1）：该位应写入要发送的从机地址位的位 0。

22.5.3 本机地址 1 寄存器（I2C_OAR1）

偏移地址：0x08

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA1EN	Res				OA1MODE	OA1[9:8]		OA1[7:1]							OA1[0]
rw					rw	rw		rw							rw

位 31:16	Res：保留 必须保持复位值。
位 15	OA1EN：本机地址 1 启用（Own Address 1enable） • 0：禁用本机地址 1，接收到从机地址 OA1 后会用 NACK 回应。 • 1：本机地址 1 启用，接收到从机地址 OA1 后会用 ACK 回应。
位 14:11	Res：保留 必须保持复位值。
位 10	OA1MODE：本机地址 1 的 10 位模式（Own Address 1 10-bit mode） • 0：本机地址 1 是 7 位地址 • 1：本机地址 1 是一个 10 位的地址
位 9:8	OA1[9:8]：接口地址的 9:8 位（Interface address 9:8bit） • 7 位地址模式：该值不起作用。 • 10 位地址模式：地址的位 9:8。
位 7:1	OA1[7:1]：接口地址的 7:1（Interface address 7:1bit）
位 0	OA1[0]：接口地址的 0 位（Interface address 0bit） • 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。

22.5.4 本机地址 2 寄存器（I2C_OAR2）

偏移地址：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA2EN	Res				OA2MSK[2:0]			OA2[7:1]							Res
rw					rw			rw							

位 31:16	Res：保留 必须保持复位值。
位 15	OA2EN：本机地址 2 启用（Own Address 2enable） • 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 2masks） • 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。

22.5.5 时序寄存器（I2C_TIMINGR）

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRESC[3:0]				Res				SCLDEL[3:0]				SDADEL[3:0]			
rw								rw				rw			

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	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 tSCLDEL。 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 TSDADEL。 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}

22.5.6 超时寄存器（I2C_TIMEOUTR）

偏移地址：0x14

复位值：0x0000 0000

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

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

位 31	TEXTEN：外部时钟超时启用（Extended clock timeout enable） • 0：外部时钟超时检测被禁用 • 1：外部时钟超时检测启用
位 30:29	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	TIMOUTEN：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_{I2C_CLK} • TIDLE=1 的时候，总线空闲条件（SCL 和 SDA 同时为高）。 t_{IDLE}=（TIMEOUTA+1）* 4 *t_{I2C_CLK}

22.5.7 中断和状态寄存器（I2C_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 tLOW 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 I2C_RXDR 寄存器，准备好被软件读取的时候由硬件置位。在读取 I2C_RXDR 时 RXNE 会被清除。
位 1	TXIS：发送中断状态（发送）（Transmit interrupt status (transmitters)） 在 I2C_TXDR 寄存器为空的时候由硬件置 1，这时必须把要发的数据写到 I2C_TXDR 寄存器。下一个要发送的数据被写到 I2C_TXDR 寄存器的时候它会被清除。该位只在 NOSTRETCH=1 的时候可以由软件写成 1，以生成一个 TXIS 事件（如果 TXIE=1 就有中断，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 I2C_TXDR 寄存器为空的时候由硬件置 1。下一个要发送的数据被写到 I2C_TXDR 寄存器的时候它会被清除。 该位可通过软件写 1，以清空发送数据寄存器 I2C_TXDR。

22.5.8 中断清除寄存器（I2C_ICR）

偏移地址：0x1C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res		ALERTCF	TIMOUTCF	PECCF	OVERCF	ARLOCF	BERRCF	Res		STOPCF	NACKCF	ADDRCF	Res		
		w	w	w	w	w	w			w	w	w			

位 31:14	Res：保留 必须保持复位值。
位 13	ALERTCF：通知标志清零（Alert flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ALERT 标志位。
位 12	TIMOUTCF：超时检测标志清除（Timeout detection flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 TIMEOUT 标志位。
位 11	PECCF：PEC 错误标志清除（PEC Error flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 PECERR 标志位。
位 10	OVRCF：溢出/欠载标志清除（Overrun/Underrun flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 OVR 标志位。
位 9	ARLOCF：仲裁丢失标志清除（Arbitration Lost flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ARLO 标志位。
位 8	BERRCF：总线错误标志清除（Bus error flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 BERRF 标志位。

位 7:6	Res：保留 必须保持复位值。
位 5	STOPCF：停止检测标志清除（STOP detection flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 STOPF 标志位。
位 4	NACKCF：收到 NACK 标志清除（Not Acknowledge flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 NACKF 标志位。
位 3	ADDRCF：地址匹配标志清除（Address matched flag clear） 对该位写 1，会清除 I2C_ISR 寄存器中的 ADDR 标志位。对该位写 1，还会清除 I2C_CR2 寄存器中的 START 位。
位 2:0	Res：保留 必须保持复位值。

22.5.9 PEC 寄存器（I2C_PECR）

偏移地址：0x20

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	PEC[7:0]：包错误检查寄存器（Packet error checking register） 当 PECEN=1 时，此字段包含内部 PEC 结果。PEC 在 PE=0 时，该位由硬件清零。

22.5.10 接收数据寄存器（I2C_RXDR）

偏移地址：0x24

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。

位 7:0	RXDATA[7:0]：8 位接收数据（8-bit receive data） 该位域是从 I2C 总线接收的数据字节。

22.5.11 发送数据寄存器（I2C_TXDR）

偏移地址：0x28

复位值：0x0000 0000

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

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

位 31:8	Res：保留 必须保持复位值。
位 7:0	TXDATA[7:0]：8 位发送数据（8-bit transmit data） 该位域是发送到 I2C 总线的数据字节。

23 通用同步异步收发器（USART）

通用同步异步收发器（USART）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。USART 可以通过自带的波特率发生器产生不同的波特率，并且还支持同步通信、单线半双工通信、多处理器通信、硬件流控制、智能卡协议等功能；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。

23.1 USART 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
- 当采用 32MHz 时钟频率和 8 倍过采样时，通用可编程收发波特率最高为 4 Mbit/s。
- 双时钟域允许：
  - USART 功能和从停机模式唤醒
  - 方便的波特率编程，独立于 PCLK 重新编程
- 自动波特率检测
- 数据字长可编程（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）
- 用于同步通信的同步模式和时钟输出
- 单线半双工通信
- 支持 DMA 传输数据
- 发射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
- 可配置发送（TX）和接收（RX）引脚交换
- RS232 硬件流控制和 RS485 驱动器使能
- 通信控制/错误检测标志
- 奇偶校验控制：
  - 发送奇偶校验位
  - 检查接收数据帧的奇偶性
- 具有多种中断状态标志位
- 多处理器通信
  - 如果地址不匹配，USART 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）

23.2 USART 扩展特性

- LIN 主模式断路发送功能和 LIN 从模式断路检测功能
- 支持红外数据协议（IrDA）
- 支持智能卡协议
- 支持 ModBus 协议通信

23.3 USART 实现

表 23-1 USART1 特性

USART 模式/特性	USART1
数据字长	8 位和 9 位
DMA 传输	支持
多处理器通信	支持
同步模式	支持
单线半双工通信	支持
双时钟域和从停机模式唤醒	支持
自动波特率检测	支持
Modbus 通信	支持
RS232 硬件流控制	支持
RS485 驱动器使能	支持
IrDA SIR ENDEC 模块	支持
LIN 模式	支持
智能卡模式	支持

23.4 USART 功能说明

USART 通信包括六个引脚，分别是接收数据输入引脚（RX）、发送数据输出引脚（TX）、时钟输出引脚（CLK）、RS232 硬件流控模式引脚（CTS、RTS）以及 RS485 驱动器使能引脚（DE）。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。在单线和智能卡模式下，该引脚用于发送和接收数据。
- CK：时钟输出引脚。该引脚用于输出发送器的数据时钟，以便按照 SPI 主模式进行同步发送（起始位和结束位上无时钟脉冲，可通过软件向最后一个数据位发送时钟脉冲）。RX 上可同步接收数据。时钟相位和极性可编程。在智能卡模式下，CK 输出可向智能卡提供时钟。
- CTS：该引脚在 RS232 硬件流控模式下使用。输入高电平时，在当前传输结束时阻止 USART 发送新的数据。
- RTS：该引脚在 RS232 硬件流控模式下使用。输出低电平时，用于指示 USART 已准备好接收数据。
- DE：该引脚在 RS485 驱动器使能下使用。它用于使能外部收发的发送模式。

注意：DE 和 RTS 共用一个引脚。

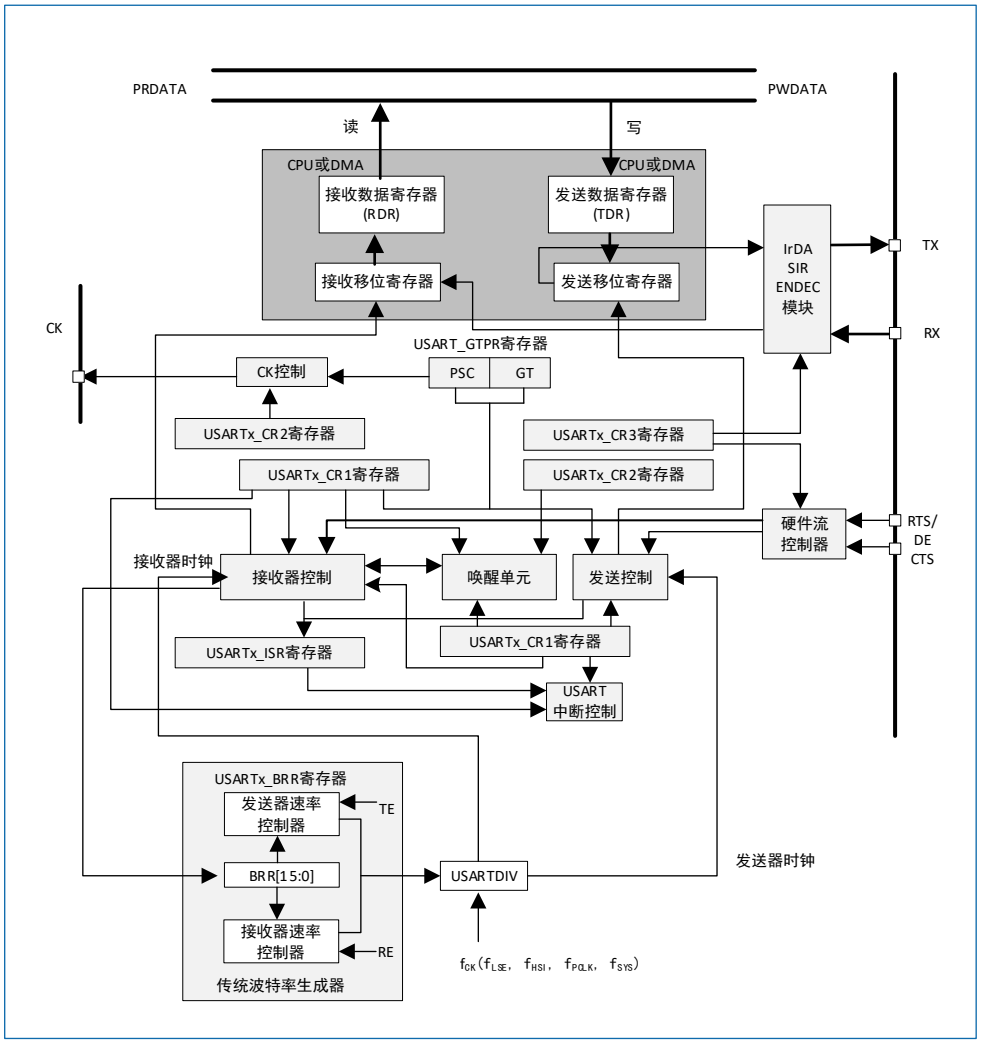


图 23-1 USART 框图

在图 23-1 中，关于在 USART_BRR 寄存器中编码 USARTDIV 的详细信息，请参见“23.4.4 USART 波特率生成”。f_{CK} 可以是 f_{LSE}、f_{HSI}、f_{PCLK} 或 f_{SYS}。

23.4.1 USART 字符说明

通过配置 USART_CR1 寄存器中的 M[1:0]位来选择 8 位或 9 位的字长（请参见图 23-2）。

- 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（USART_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。

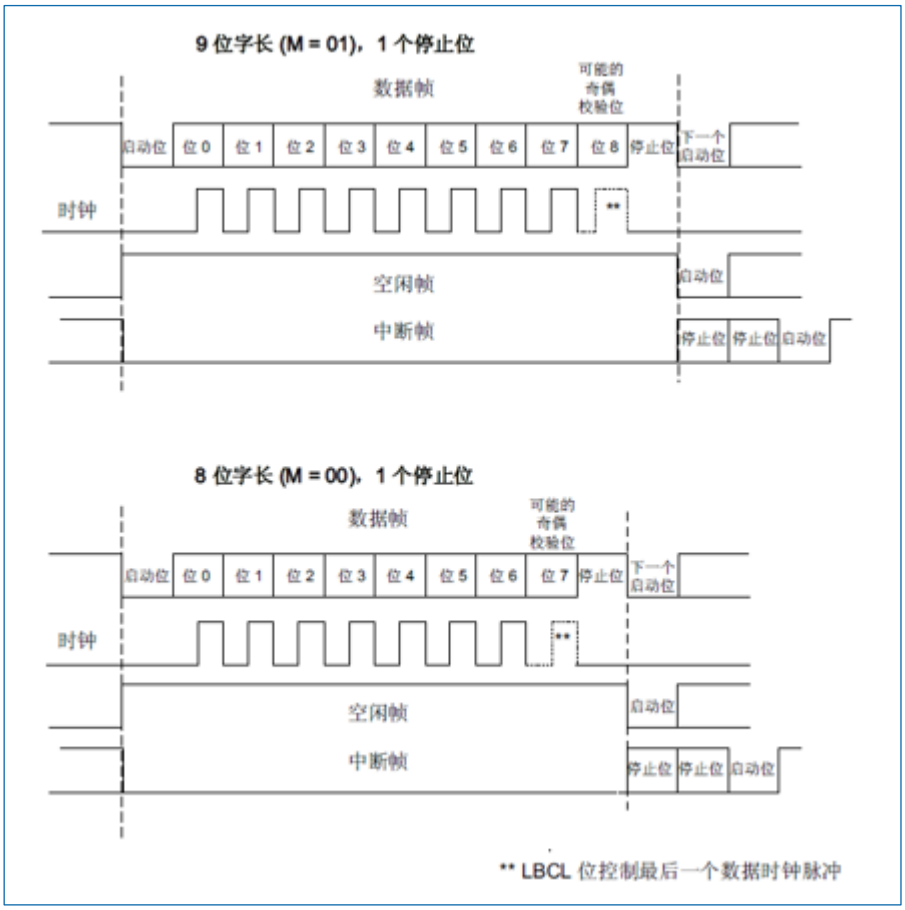


图 23-2 字长编程

23.4.2 USART 发送器

通过配置 USART_CR1 寄存器中的 M[1:0]位来选择发送器的数据字长（8 位或 9 位）。

通过配置 USART_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，相应的时钟脉冲从 CK 引脚输出。

可配置的停止位

通过配置 USART_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 USART 模式、单线模式和调制解调器模式支持该值。
- 1.5 个停止位：用于智能卡模式。

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）），然后是 2 个停止位。无法传送长中断（中断长度大于 10/11 个低电平位）。

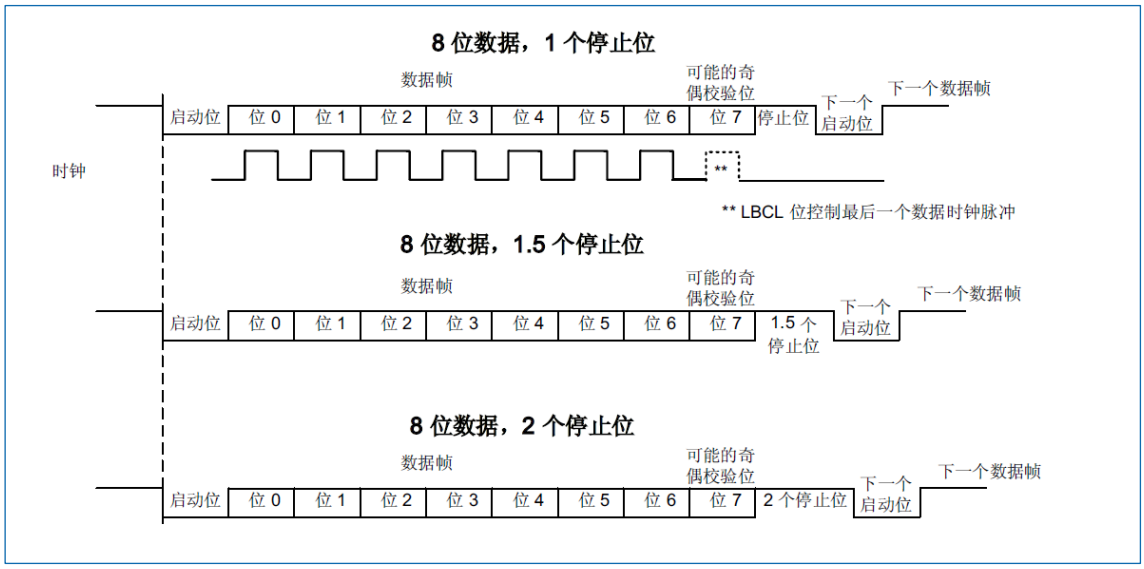


图 23-3 可配置的停止位

字符发送

USART 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（USART_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 USART_TDR 寄存器移到移位寄存器中并且数据开始发送。
- USART_TDR 寄存器为空。
- USART_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 USART_CR1 寄存器中的 TCIE 位，则会产生 TC 中断。向 USART_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。

USART 字符发送时序如图 26-4 所示。

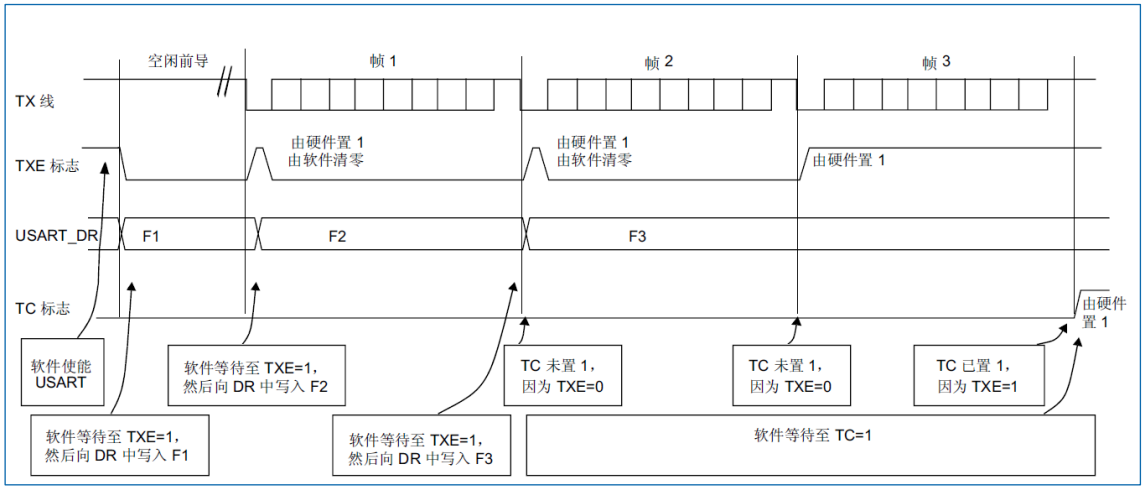


图 23-4 发送时的 TC/TXE 行为

字符发送配置步骤：

1. 配置 USART_CR1 中的 M 位来定义字长。
2. 配置 USART_BRR 寄存器来选择所需波特率。
3. 配置 USART_CR2 中的停止位（STOP[1:0]）个数。

4. 配置 USART_CR1 寄存器中的 UE 位来使能 USART 通信。

5. 若需要配合 DMA 使用，则配置 USART_CR3 中的 DMA 使能（DMAT 位）。按照 DMA 模式下的 USART 通信的要求配置 DMA 寄存器。

6. 配置 USART_CR1 中的 TE 位以便在首次发送时发送一个空闲帧。

7. 在 USART_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。

8. USART_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 USART 发送器或进入停止模式时需要此步骤，避免损坏最后一次发送。

中断帧（BREAK）发送

若置位 USART_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 USART_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。USART 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 USART_CR1 寄存器中的 TE 位置 1 会驱动 USART 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。

23.4.3 USART 接收器

通过配置 USART_CR1 寄存器中的 M 位来选择接收器接收的数据字长（8 位或 9 位），接收器支持 1、1.5 和 2 个停止位。

起始位检测

USART 支持 8/16 倍过采样，起始位检测序列相同。

在 USART 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 0X 0X 0X 0X 0X 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）。

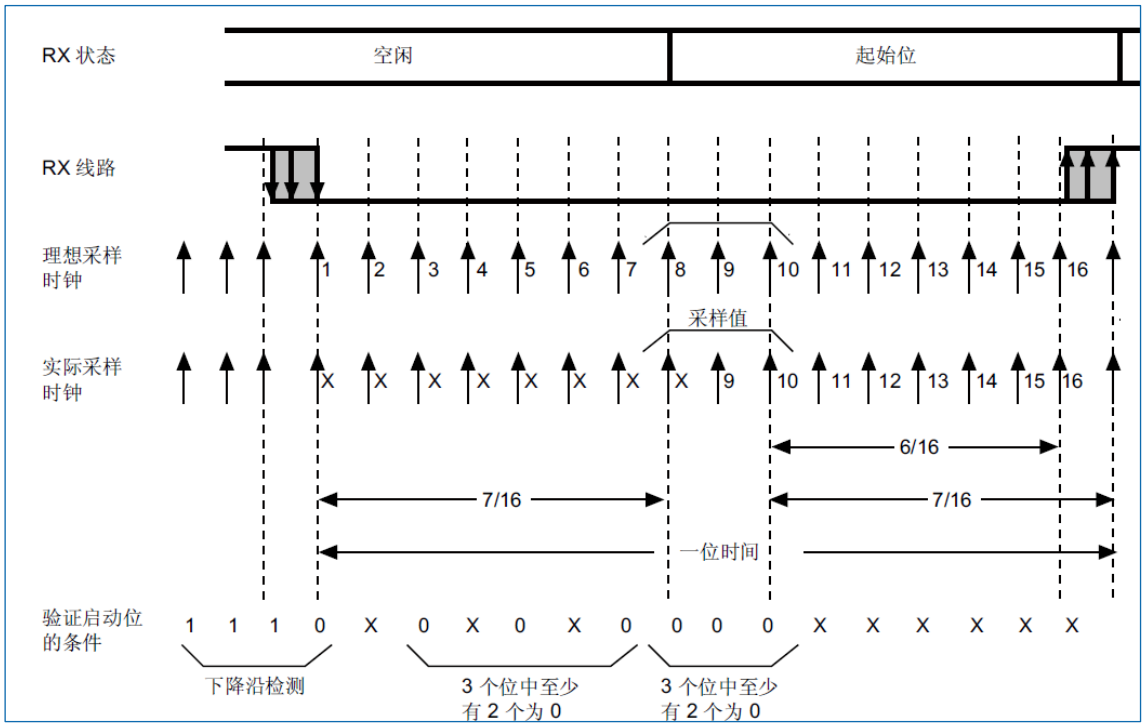


图 23-5 16 倍或 8 倍过采样时的起始位检测

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

USART 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 USART_CR1 中的 M 位来定义字长。
2. 配置 USART_BRR 寄存器来选择所需波特率。
3. 配置 USART_CR2 中的停止位（STOP[1:0]）个数。
4. 配置 USART_CR1 寄存器中的 UE 位来使能 USART 通信。
5. 若需要配合 DMA 使用，则配置 USART_CR3 中的 DMA 使能（DMAR）。按照 DMA 模式下的 USART 通信要求配置所需的 DMA 寄存器。
6. 将 USART_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 USART_RDR 寄存器执行读操作将 RXNE 位清零。也可以通过将 USART_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，USART_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，USART_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 USART_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 USART_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

USART 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（16M）或 SYSCLK。在低功耗模式下，USART 可以选择 LSE 或 HSI 作为时钟源来接收数据，从低功耗模式下唤醒的方式可以通过接收数据和唤醒模式的配置来实现。

接收器通过配置过采样方式，可从噪声中提取有效数据。这可在最大通信速度与抗噪声/时钟误差性能之间实现平衡。

可通过编程 USART_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 23-6 和图 23-7）。

- 选择 8 倍过采样（OVER8=1）以获得更高的传输速率（高达 fCK/8），其中 fCK 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“23.4.5 USART 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 fCK/16。

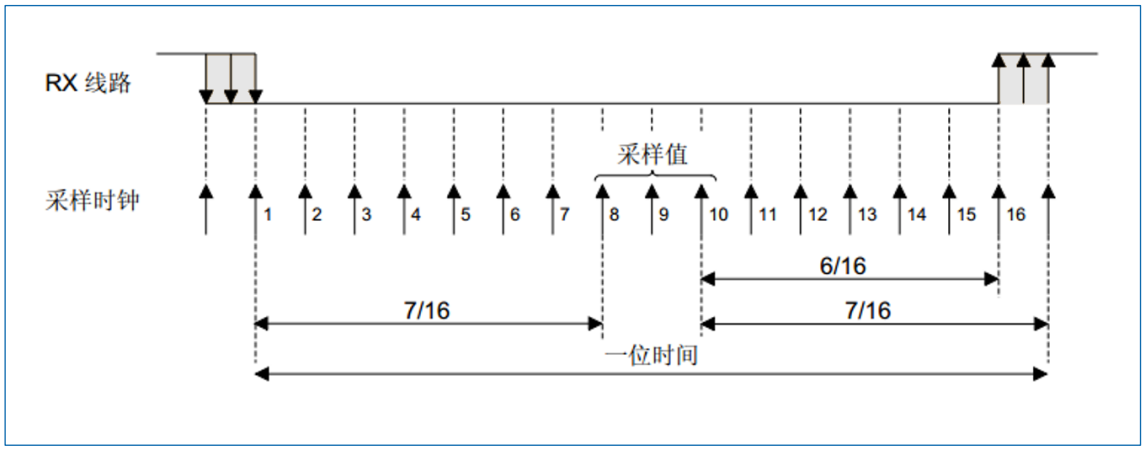


图 23-6 16 倍过采样时的数据采样

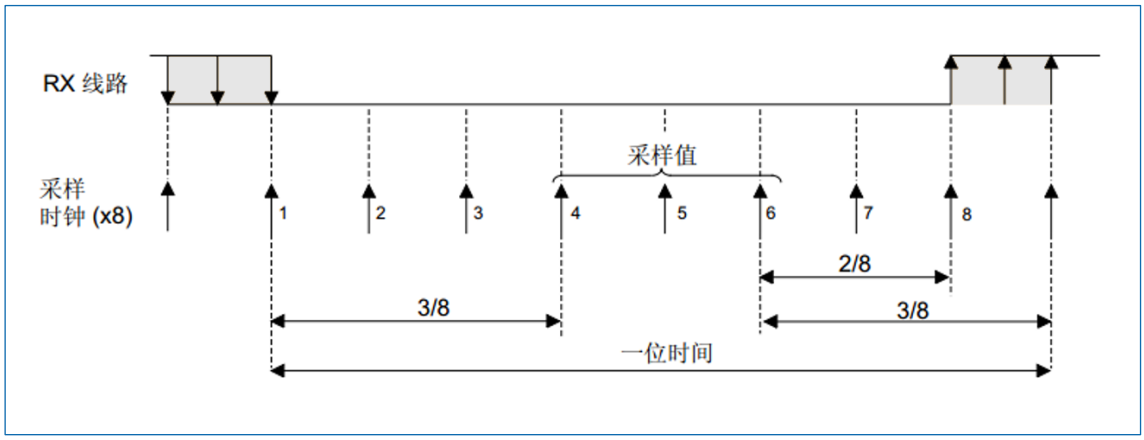


图 23-7 8 倍过采样时的数据采样

通过编程 USART_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 23-2）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“23.4.5 USART 接收器对时钟偏差的容差”）。

表 23-2 通过采样数据进行噪声检测

采样值	NE 状态	接收的位值
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

帧错误

如果接收数据时未在预期时间内识别出停止位，帧错误标志位会被置 1。

检测到帧错误时：

- FE 位由硬件置 1。
- 无效数据从移位寄存器传送到 USART_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，USART_CR3 寄存器中的 EIE 位置 1 时将发出中断。通过将 1 写入 USART_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：可以是 1 或 2 个（正常模式下），也可以是 1.5 个（智能卡模式下）。

- 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。
- 1.5 个停止位（智能卡模式）：在智能卡模式下发送时，设备必须检查数据是否正确发送。因此必须使能接收器块（USART_CR1 寄存器中的 RE =1）并检查停止位，以测试智能卡是否已检测到奇偶校验错误。发生奇偶校验错误时，智能卡会在采样时将数据信号强制为低电平，即 NACK 信号，该信号被标记为帧错误。之后，FE 标志在 1.5 个停止位的末尾由 RXNE 置 1。在第 16、第 17 和第 18 次采样时对 1.5 个停止位进行采样（停止位采样开始后维持 1 个波特时钟周期）。1.5 个停止位可分为 2 个部分：0.5 个波特时钟周期（未发生任何动作），然后是 1 个正常的停止位周期（一半时间处进行采样）。更多详细信息，请参见“23.4.13 USART 智能卡模式”。
- 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。

23.4.4 USART 波特率生成

配置 USART_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 USART_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为：

$$\text{Tx/Rx baud}=\frac{f_{ck}}{\text{USARTDIV}} \quad \text{（公式 23-1）}$$

8 倍过采样（OVER8=1）时的波特率生成公式为：

$$\text{Tx/Rx baud}=\frac{2*f_{ck}}{\text{USARTDIV}} \quad \text{（公式 23-2）}$$

其中 USARTDIV 的值由 BRR 位获得。USART 的普通模式和同步模式均支持 16 倍过采样和 8 倍过采样，但 LIN 模式、IrDA 模式仅支持 16 倍过采样。

- 当 OVER8=0 时，BRR=USARTDIV。
- 当 OVER8=1 时：
  - BRR[2:0]=（USARTDIV[3:0]>>1）。
  - BRR[3]必须保持清零。
  - BRR[15:4]=USARTDIV[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，USARTDIV 必须大于或等于 0x10。

通过 USART_BRR 寄存器获取 USARTDIV 的示例如下：

示例 1：通过 f_{CK}=8MHz 获得 9600 波特率

- 根据公式 23-1，当采用 16 倍过采样时：
  - USARTDIV =f_{ck}/(Tx/Rx baud) = 8000 000/9600
  - BRR=USARTDIV=833D=0341h
- 根据公式 23-2，当采用 8 倍过采样时：
  - USARTDIV= 2 * f_{ck}/(Tx/Rx baud) = 2*8000 000/9600
  - USARTDIV=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

- 根据公式 23-1，当采用 16 倍过采样时：
  - USARTDIV= f_{ck}/(Tx/Rx baud) =32000 000/921 600
  - BRR=USARTDIV=35D=23h
- 根据公式 23-2，当采用 8 倍过采样时：
  - USARTDIV=2 * f_{ck}/(Tx/Rx baud) =2*32000 000/921 600
  - USARTDIV=70D=46h
  - BRR[3:0]=USARTDIV[3:0]>>1=6h>>1=3h
  - BRR[15:0]=0x43

表 23-3 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	NA	NA	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。

23.4.5 USART 接收器对时钟偏差的容差

当总时钟系统偏差小于 USART 接收器的容差时，USART 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 <USART 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：$DWU=\frac{t_{WUUSART}}{10*Tbit}$
- M[1:0]=01 时：$DWU=\frac{t_{WUUSART}}{11*Tbit}$

$t_{WUUSART}$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。

USART 接收器在表 23-4 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 USART_CR1 寄存器中的 M 位定义的 10 位或 11 位字符长度。
- 由 USART_CR1 寄存器中的 OVER8 位定义的 8 倍或 16 倍过采样。
- USART_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 USART_CR3 寄存器中 ONEBIT 位的值。

表 23-4 BRR[3:0]=0000 时的 USART 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%

表 23-5 BRR[3:0]!=0000 时的 USART 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%

注意：当接收的帧恰好包含 10 个（M 位=00）、11 个（M 位=01）位持续时间的空闲帧时，表 23-4 和表 23-5 中指定的数据可能与特例中的数据略微不同。

23.4.6 USART 自动波特率检测

USART 可根据接收的字符检测对端设备的波特率并自动设置 USART_BRR 寄存器的值。

自动波特率检测适用于以下场景：

- 系统的通信速率未知。

- 系统使用精确度相对较低的时钟源。

时钟源频率必须与预期通信速度匹配（16 倍过采样时，波特率介于 $f_{CK}/65535$ 与 $f_{CK}/16$ 之间；8 倍过采样时，波特率介于 $f_{CK}/65535$ 与 $f_{CK}/8$ 之间）。

可通过配置 USART_CR2 寄存器中的 ABRMOD[1:0]字段选择自动波特率检测模式。

这些模式包括：

- 模式 0：以 1 位开头的任意字符。这种情况下，USART 会测量起始位的持续时间（下降沿到上升沿）。
- 模式 1：以 10xx 位模式开头的任意字符。这种情况下，USART 会测量起始位和第一个数据位的持续时间（下降沿到下降沿）。

激活自动波特率检测前，必须先通过向 USART_BRR 寄存器写入非零的波特率值来初始化该寄存器。

配置 USART_CR2 寄存器中的 ABREN 位来使能自动波特率检测功能。在使能该功能后，USART 将等待 RX 线路上的第一个字符。

自动波特率检测操作完成后，USART_ISR 寄存器中的 ABRF 标志位被置 1。

以下两种情况无法保证正确的波特率检测，可能导致 BRR 值损坏，即发生自动波特率错误（ABRE 错误标志位将置 1）：

- 线路受到噪声干扰。
- 通信速率不在自动波特率检测范围内（即 16 倍过采样时持续时间不在 16 个和 65536 个时钟周期之间，8 倍过采样时持续时间不在 8 个和 65536 个时钟周期之间）。

通过配置 ABRF 标志位（通过写入 0）重新使能自动波特率检测。

说明：如果在自动波特率操作期间禁止 USART（UE=0），则可能损坏 BRR 值。

23.4.7 使用 USART 进行多处理器通信

在多处理器通信中，以下位需要清零：

- USART_CR2 寄存器中的 LINEN 位。
- USART_CR3 寄存器中的 HDSEL、IREN 和 SCEN 位。

在多处理器通信中，将其中的一个 USART 作为主 USART，它的 TX 输出与其它 USART 的 RX 输入相连接。其它的 USART 作为从 USART，其各自的 TX 在逻辑上通过与运算连接在一起，并且与主 USART 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 USART 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 USART_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。
- USART_ISR 寄存器中的 RWU 位置 1。

配置 USART_CR1 寄存器中 WAKE 位，USART 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 USART_RQR 寄存器的 MMRQ 位写入 1，USART 进入静默模式（RWU 位置 1）。

当检测到空闲帧时，USART 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:

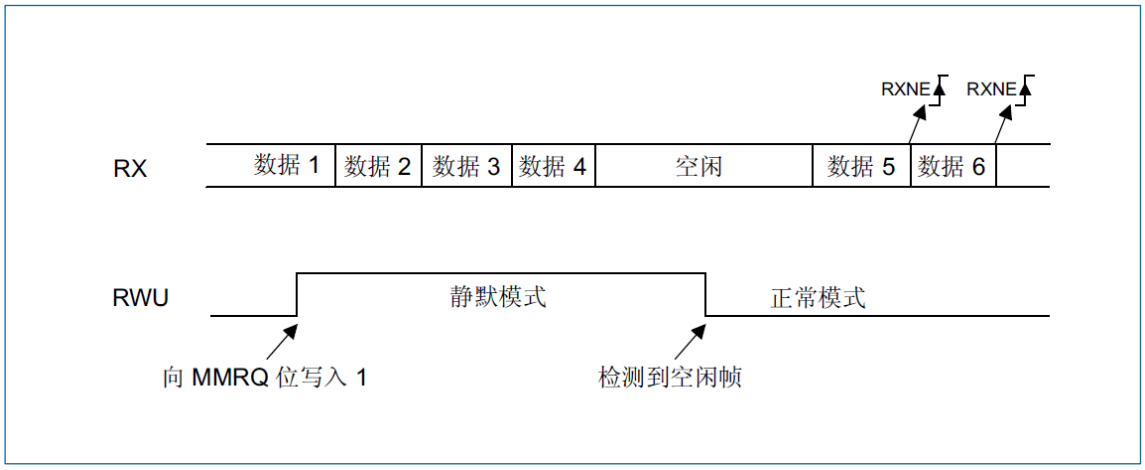


图 23-8 使用空闲线路检测时的静默模式

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。

在该模式下，MSB 为 1 的字节被识别为地址，否则被识别为数据。通过配置 USART_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，USART 会进入静默模式。此时，RWU 位将由硬件置 1。USART 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，USART 会退出静默模式。此时，RWU 位被清零，USART 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。

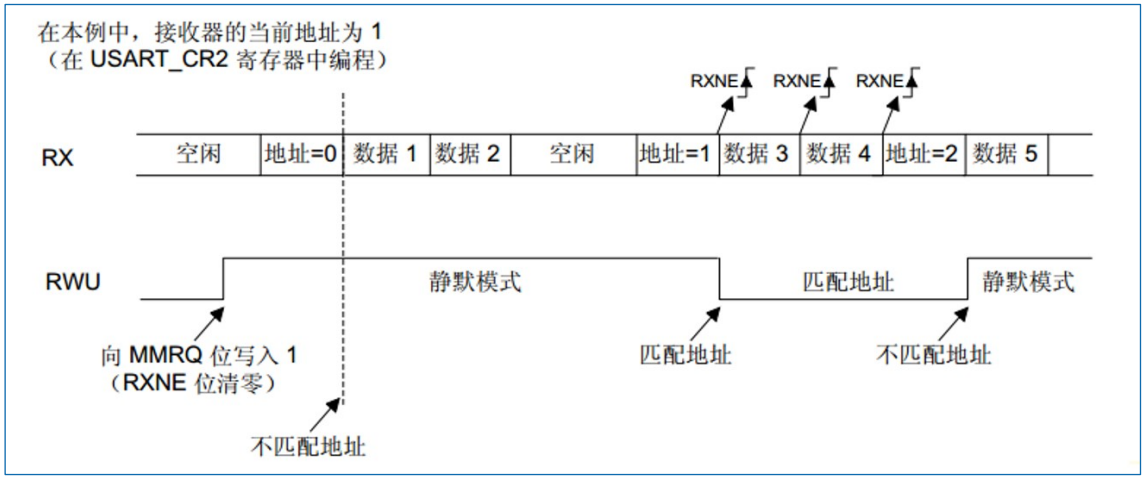


图 23-9 使用地址标记检测时的静默模式

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]）。

23.4.8 使用 USART 进行 Modbus 通信

USART 为 Modbus/RTU 和 Modbus/ASCII 协议的实现提供基本支持。

Modbus/RTU

在此模式下，块结束标志通过超过 2 个字符时间的空闲线路来识别。此功能通过 USART 接收超时功能实现。

通过配置 USART_CR2 寄存器中的 RTOEN 位和 USART_CR1 寄存器中的 RTOIE 位来使能超时功能。在接收线路空闲过程中，完成最后一个停止位接收后，生成中断，并通知软件当前块接收已完成。

Modbus/ASCII

在此模式下，块结束通过特定字符（CR/LF）序列识别。USART 通过字符匹配功能管理此机制。

通过配置 USART_CR2 中的 ADD[7:0]字段中编程特定字符 LF 对应的 ASCII 码以及使能字符匹配中断（CMIE=1），当接收到 LF 字符时，将产生中断，并且通知并检查接收缓存区中的 CR/LF。

23.4.9 USART 奇偶校验

将 USART_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 USART 帧格式。

表 23-6 帧格式

M 位	PCE 位	USART 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|

(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 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（USART_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（USART_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 USART_ISR 寄存器中的 PE 标志被置 1；如果 USART_CR1 寄存器中 PEIE 位被置 1，则会生成中断。通过软件置位 USART_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 USART_CR1 寄存器中的 PCE 位被置 1，则写入数据寄存器（USART_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。

23.4.10 USART LIN（局域互连网络）模式

通过配置 USART_CR2 寄存器中的 LINEN 位来选择 LIN 模式。在 LIN 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 STOP[1:0]和 CLKEN 位。
- USART_CR3 寄存器中的 SCEN、HDSEL 和 IREN 位。

LIN 发送

与正常的 USART 发送相比，在 LIN 主器件中发送时必须采用“23.4.2 USART 发送器”中介绍的步骤，同时还具有以下区别：

- 数据字长度必须为 8。
- LINEN 位置 1 以进入 LIN 模式。此时，将 SBKRQ 位置 1，USART 会发送 13 个 bit“0”作为中断帧。再发送 2 个 bit“1”以进行下一次启动检测。

LIN 接收

使能 LIN 模式后，将激活中断帧检测电路。该检测完全独立于正常的 USART 接收器。

使能接收器（USART_CR1 寄存器中 RE=1）后，电路便开始监测启动信号的 RX 输入。检测到起始位后，电路会对接下来的位进行采样，方法与数据采样相同（第 8、第 9 和第 10 次采样）。如果连续 10 个（USART_CR2 中 LBDL=0 时）或 11 个（USART_CR2 中 LBDL=1 时）连续位均检测为“0”，且其后跟随分隔符，则 USART_ISR 寄存器中的 LBDF 标志将置 1。如果 LBDIE 位=1，则会生成中断。在验证中断帧前，会对分隔符进行检查，分隔符表示 RX 线路已恢复到高电平。如果在第 10 或第 11 次采样前已对“1”采样，则中断帧检测电路会取消当前检测，并重新搜索起始位。

在使能 LIN 模式（LINEN=1）后，只要发生帧错误，接收器即会立即停止，直到中断帧检测电路接收到“1”（中断帧字不完整时）或接收到分隔符（检测到断路时）。

图 23-10 中显示了中断帧检测器状态机和断路标志的行为。

图 23-11 中列出了中断帧的示例。

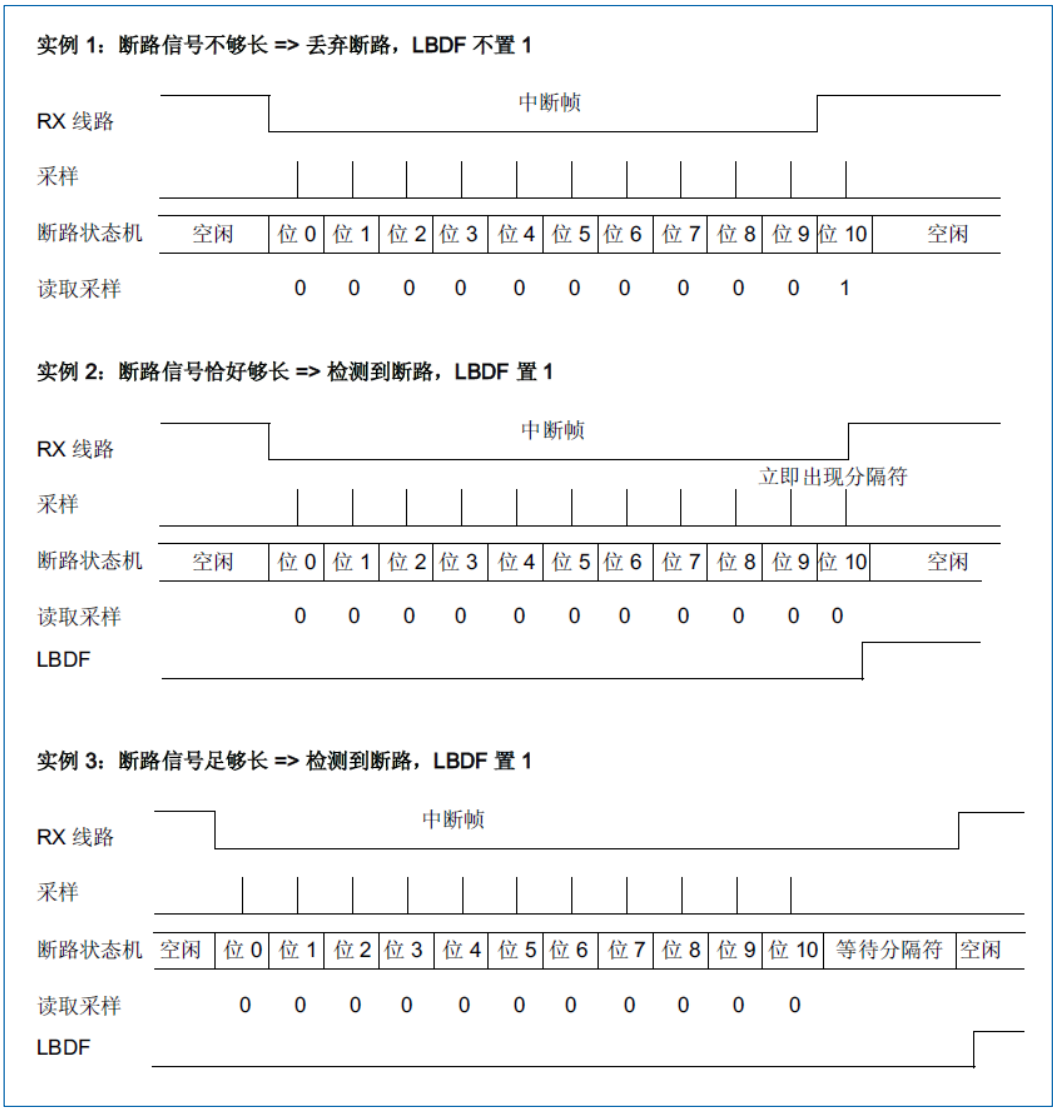


图 23-10 LIN 模式下的断路检测（11 位断路长度——LBDL 位置 1）

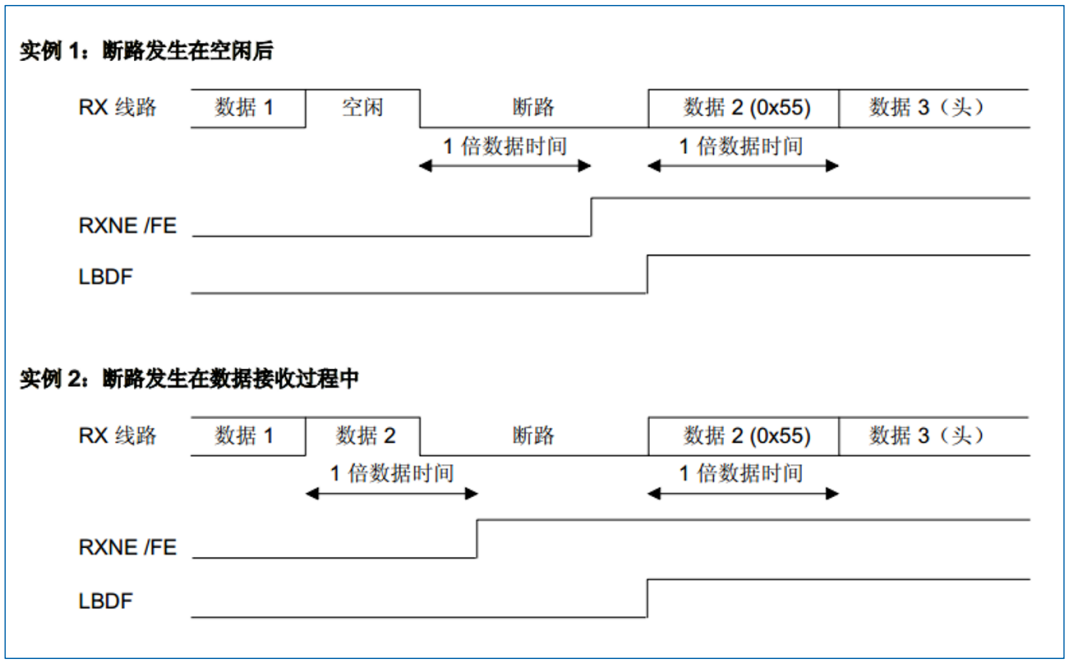


图 23-11 LIN 模式下的断路检测与帧错误检测

23.4.11 USART 同步模式

配置 USART_CR2 寄存器中的 CLKEN 位来选择同步模式。

在同步模式下，需要将以下位清零：

- USART_CR2 寄存器中的 LINEN 位。
- USART_CR3 寄存器中的 SCEN、HDSEL 和 IREN 位。

同步模式包括了 TX 引脚、RX 引脚以及 CK 引脚，可以在主模式下完成双向同步串行通信。CK 引脚是 USART 的时钟输出引脚，只有在数据帧期间发送时钟脉冲。通过配置 USART_CR2 寄存器中 LBCL 位来控制最后一个数据位是否生成时钟脉冲。同时也可以通过配置 USART_CR2 寄存器中的 CPOL 位和 CPHA 位来选择时钟极性和时钟的相位（请参见图 23-12、图 23-13 和图 23-14）。

USART 发送器在同步模式下的工作方式与异步模式下完全相同。但是由于 CK 与 TX 同步（根据 CPOL 和 CPHA），因此 TX 上的数据是同步的。

USART 接收器在同步模式下的工作方式与异步模式下不同。如果 RE=1，则数据在 CK 上采样（上升沿或下降沿，取决于 CPOL 和 CPHA），而不会进行任何过采样。此时需要确保建立时间和保持时间（取决于波特率：1/16 位持续时间）。

注意：CK 引脚需与 TX 引脚结合使用。只有在发送数据的时候才会产生 CK 时钟脉冲。

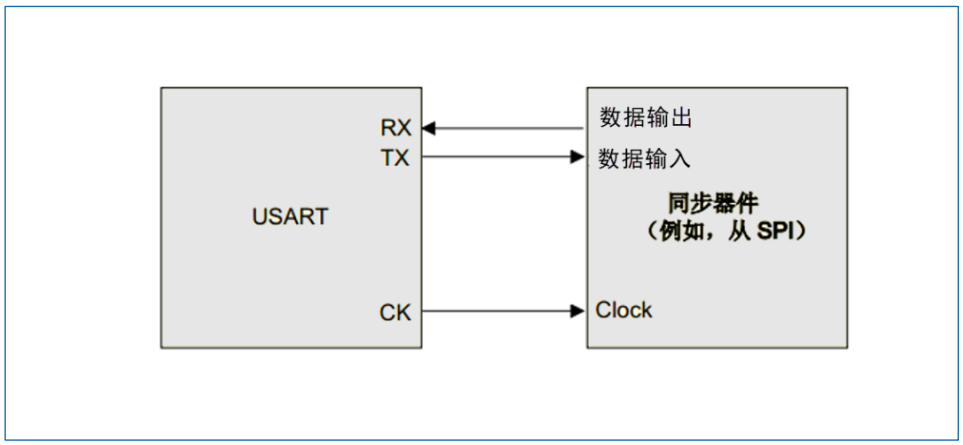


图 23-12 USART 同步发送示例

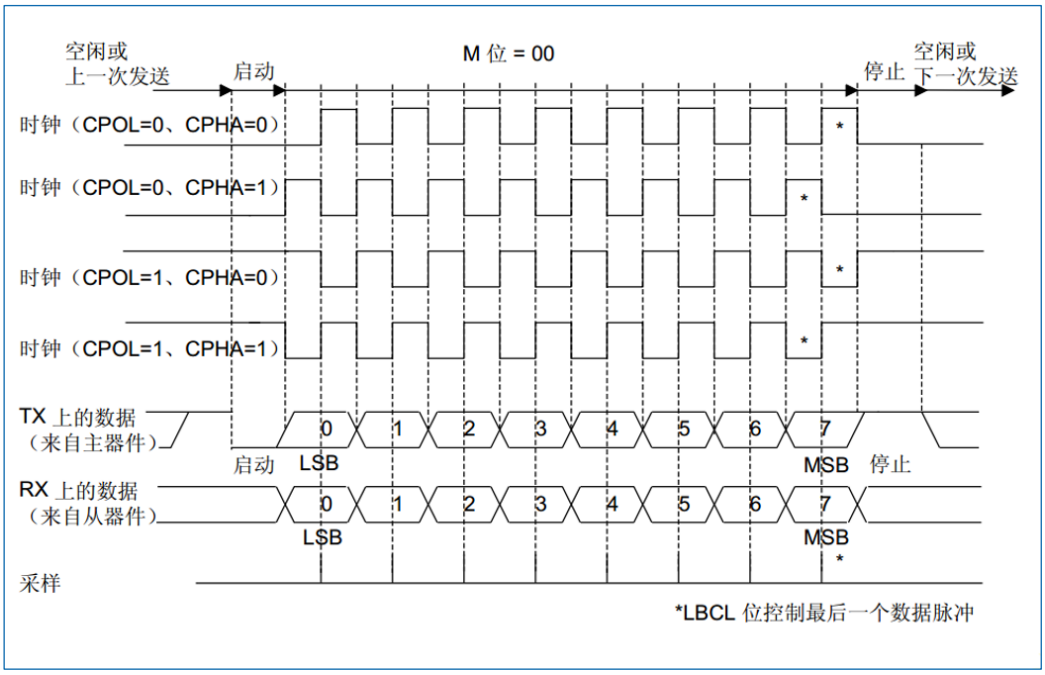


图 23-13 USART 数据时钟时序图（M 位=00）

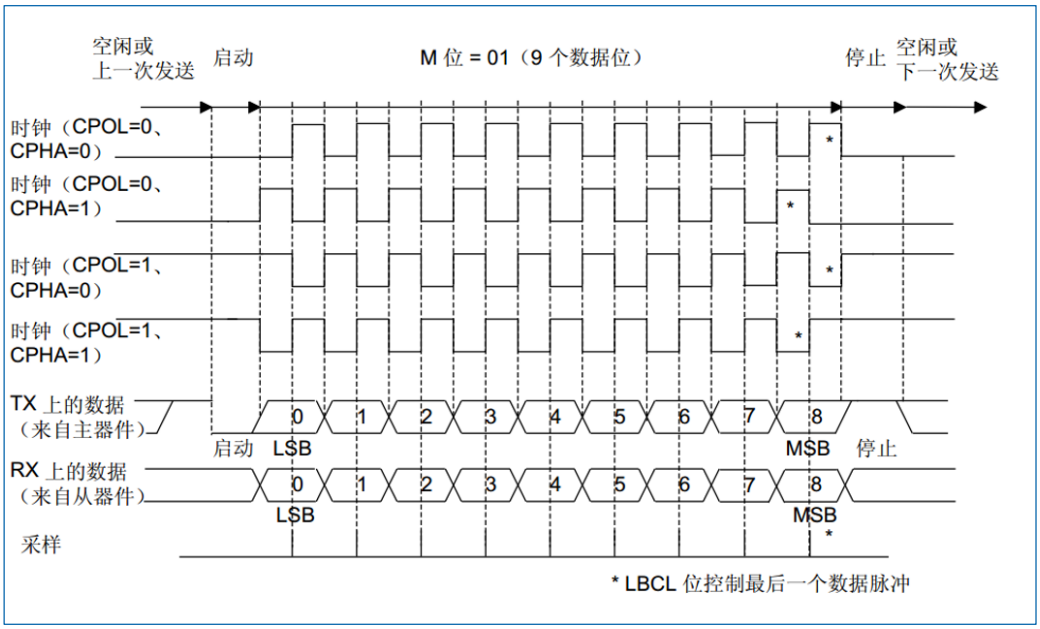


图 23-14 USART 数据时钟时序图（M 位=01）

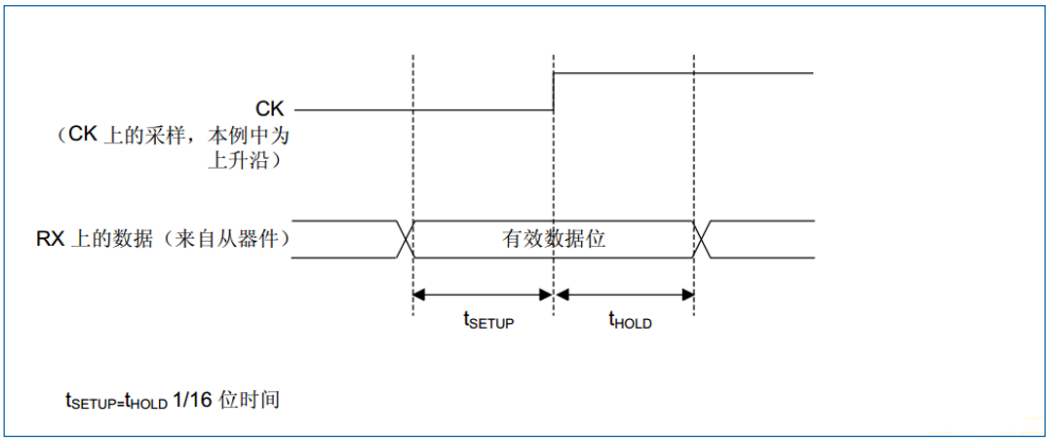


图 23-15 RX 数据建立/保持时间

注意：在智能卡模式下，CK 的功能跟上述描述有所不同。更多详细信息，请参见"23.4.13 USART 智能卡模式"。

23.4.12 USART 单线半双工通信

通过将 USART_CR3 寄存器中的 HDSEL 位置 1 来选择单线半双工模式。在此模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 位和 CLKEN 位。
- USART_CR3 寄存器中的 SCEN 和 IREN 位。

USART 支持单线半双工协议，其中 TX 和 RX 线路从内部相连接。一旦 USART_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志 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 USART 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。

23.4.13 USART 智能卡模式

通过将 USART_CR3 寄存器中的 SCEN 位置 1 选择智能卡模式。

在智能卡模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN 位。
- USART_CR3 寄存器中的 HDSEL 和 IREN 位。

此外，需配置 USART_CR2 寄存器中的 CLKEN 位来为智能卡提供时钟。

USART 的智能卡模式支持 ISO 7816-3 标准的异步协议。同时支持 T=0（字符模式）和 T=1（块模式）。

通过配置以下寄存器来实现智能卡模式：

- 8 个位加奇偶校验：USART_CR1 寄存器的 PCE 位=1。
- 1.5 个停止位：USART_CR2 寄存器中 STOP[0:1]=11。

在 T=0（字符模式）下，奇偶校验错误会在每个字符结束时指示。

下图为有无奇偶校验错误时，数据线上情况的示例。

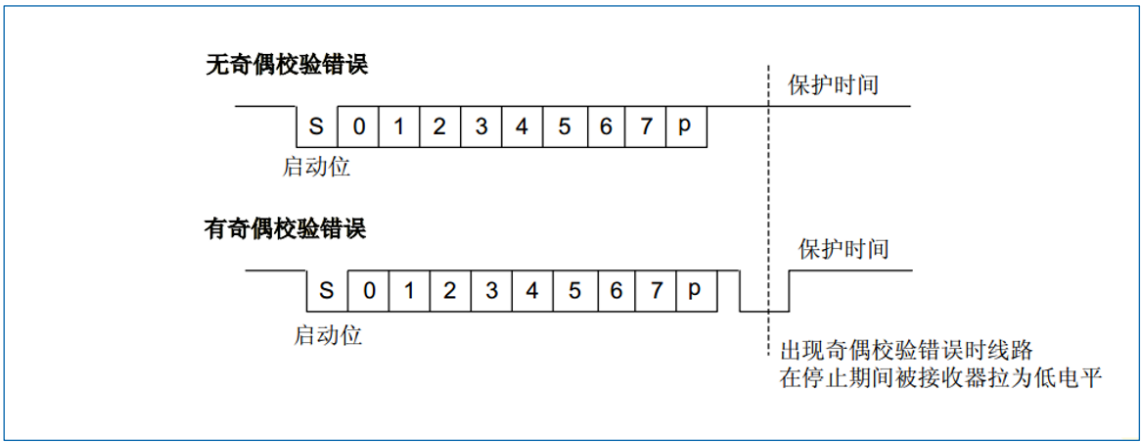


图 23-16 ISO7816-3 异步协议

在智能卡模式下，TX 引脚需要配置为开漏模式。

智能卡模式采用单线半双工通信协议。

- 从发送移位寄存器传输的数据被延迟至少 1/2 波特率时钟。正常工作时，装有数据的发送移位寄存器会在下一个波特时钟边沿开始移位。在智能卡模式下，这种传输方式更加有效的保证了 1/2 波特率时钟延迟。
- 在智能卡模式下，若发送时检测到奇偶校验错误，则会通过将线路驱动为低电平（NACK）。该信号会导致发送器端出现帧错误，与此同时，USART 会根据协议重新发送数据。若多次发送依然收到 NACK，则 USART 会停止发送。
- 发送时智能卡自动重试：USART 需要检测 NACK 与重复字符的起始位之间插入 2.5 个波特率周期的延迟，若需要重复此操作，需要确保最短 2 个波特率周期。在接收到最后一个重复字符时，TC 位会被置 1。
- 在接收一个 1.5 个停止位的数据帧期间检测到奇偶校验错误，发送线则会在接收数据帧后拉低一个时钟周期（这是为了通知智能卡：发送给 USART 的数据尚未被正确接收）。若配置寄存器 USART_CR3 中 NACK 位，则接收器会在奇偶校验错误的情况下发送 NACK 信号（T=1 模式下会使用）。根据智能卡协议规范，若接收到的字符错误，智能卡需重新发送相同的字符，通过配置 SCARCNT 位来指定的最大重试次数后接收到的字符仍然错误，则 USART 停止发送 NACK 信号，并以奇偶校验错误的形式发送出去。
- 接收时智能卡自动重试：若 USART 向智能卡发送 NACK 信号，则智能卡不能重复字符，同时 BUSY 标志位将保持为”1”。
- 在发送时，USART 会在两个连续字符之间插入保护时间（通过配置 USART_GTPR 寄存器中的 GT[7:0]）。通过对保护时间寄存器进行编程，可以延迟 TC 标志的置位。在智能卡模式下，空的发送移位寄存器会触发保护时间计数器，使其递增计数至保护时间寄存器中的值。在此期间，TC 标志被强制为低电平。当保护时间计数器达到设置值时，TC 会被置位。
- TC 标志的释放不受智能卡模式的影响。
- 在接收端，如果检测到奇偶校验错误并发送了 NACK 信号，则接收端不会将 NACK 作为起始位进行检测。

注意：中断字符在智能卡模式下无效。当翻转 TE 位时，不会发送空闲帧。空闲帧（在其它配置中进行了定义）在 ISO 协议中未进行定义。

下图详细介绍了 USART 如何对 NACK 信号采样：

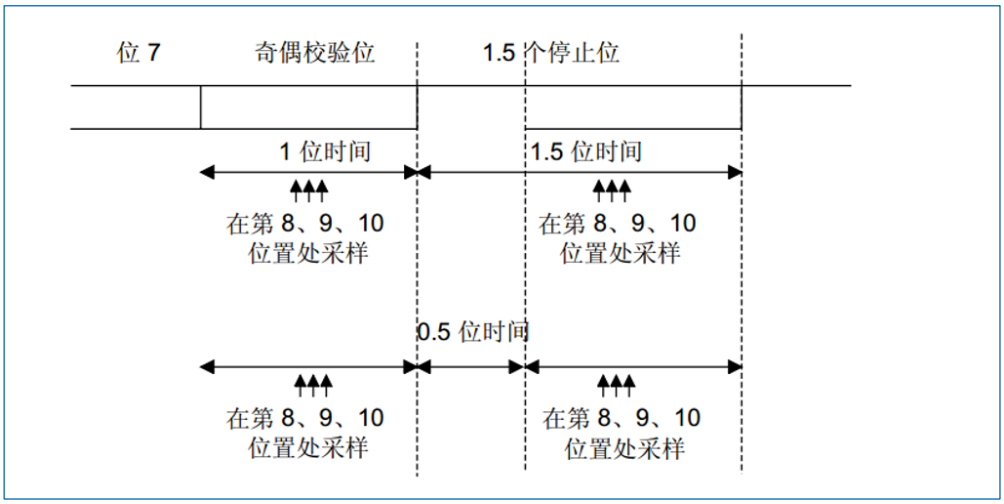


图 23-17 使用 1.5 个停止位检测奇偶校验错误

在智能卡模式下，需要 USART 提供 CK 时钟信号，CK 可以通过一个 5 位预分频器由内部外设输入时钟提供，并且可通过 USART_GTPR 寄存器来配置预分频。CK 频率可在 $f_{CK}/2$ 到 $f_{CK}/62$ 之间进行编程，其中 f_{CK} 为外设输入时钟。

在 T=1（块）模式下，通过将 USART_CR3 寄存器中的 NACK 位清零可停止奇偶校验错误发送。

在块模式下，从智能卡请求读操作时，需要配置 USART_CR2 寄存器中的 RTOEN 位来使能接收器超时，并且将 USART_RTOR 寄存器中的 RTO 设置为块等待时间（BWT）。若未收到智能卡的应答，RTOF 标

志位被置 1，若配置了 USART_CR1 寄存器中的 RTOIE 位，则会产生中断。若收到第一个字符，则通过 RXNE 中断发出信号指示。

注意：在使用 DMA 模式也需要能 RXNE 中断。

注意：RTO 计数器遵循以下规则开始计数：

- STOP=00 时，从停止位结束时开始计数。
- STOP=10 时，从第二个停止位结束时开始计数。
- STOP=11 时，从 STOP 位结束后开始计数 1 个位的持续时间。
- STOP=01 时，从 STOP 位开始时开始计数。

正向约定和反向约定

智能卡协议定义了两种约定：正向约定和反向约定。

正向约定：LSB 在前，逻辑位的值 1 对应于信号线路的 H 状态，奇偶校验为偶校验。若使用此约定，必须配置以下控制位：MSBFIRST=0，DATAINV=0（默认值）。

反向约定：MSB 在前，逻辑位的值 1 对应于信号线路的 L 状态，奇偶校验为偶校验。若使用此约定，必须配置以下控制位：MSBFIRST=1，DATAINV=1。

注意：将逻辑数据值取反（0=H，1=L）时，奇偶校验位将同样取反。

为识别智能卡约定，智能卡会将初始字符 TS 作为 ATR（复位应答）的第一个字符发送。TS 支持两种格式：LH HLLL LLLH 和 LH HLHH HLLH。

- （H）LH HLLL LLLH 建立反向约定：状态 L 编码为值 1，时间分量 2 传送最高有效位（MSB 在前）。按反向约定解码时，传送的字节等于“3F”。
- （H）LH HLHH HLLH 建立正向约定：状态 H 编码为值 1，时间分量 2 传送最低有效位（LSB 在前）。按正向约定解码时，传送的字节等于“3B”。

在 2 到 10 的九个时间分量中，如果有偶数个位设置为 1，则字符的奇偶校验正确。

USART 拥有一种可识别任意一种模式的相应操作，同时模式识别通过软件序列完成。

因此，有以下两种方法可用于识别 TS 模式：

方法 1

将 USART 配置为标准智能卡模式/正向约定。这种情况下，TS 模式接收会产生一个奇偶校验错误中断和错误信号到智能卡。

- 奇偶校验错误中断告知软件，智能卡未以正向约定正确应答。
- 为了响应错误信号，智能卡会重试同一 TS 字符。此时重新编程后的 USART 将正确接收该字符。为了应答奇偶校验错误中断，可重新编程 USART，同时向智能卡发送新的复位命令，并且等待 TS。

方法 2

将 USART 编程为 9 位/无奇偶校验模式，无位反向。在此模式下，按如下方式接收两种 TS 模式中的任一种：

- （H）LH HLLL LLLH=0x103->选择反向约定
- （H）LH HLHH HLLH=0x13B->选择正向约定

根据这两种模式检查接收到的字符，如果其中任意一种匹配，则编程相应 USART 以接收下一个字符。

如果两种都未被识别，则可能复位智能卡以重新开始协商。

23.4.14 USART IrDA SIR ENDEC 模块

配置 USART_CR3 寄存器中的 IREN 位来选择 IrDA 模式。

在 IrDA 模式下，必须将以下位清零：

- USART_CR2 寄存器中的 LINEN、STOP 和 CLKEN 位。
- USART_CR3 寄存器中的 SCEN 和 HDSEL 位。

在 IrDA 模式下，USART_CR2 寄存器中的 STOP 位必须配置为 1 个停止位。

IrDA SIR 物理层使用反相归零（RZI）调制方案，它以红外光脉冲表示逻辑 0（参见图 23-18）。

串行红外发送编码器将标准 USART 模块输出的非归零码（NRZ）的串口数据调制为 RZI 的红外数据，然后通过 TX 引脚输出到外部。USART 串口红外最高可支持 115.2Kbps 波特率。串行红外接收解码器将从 RX 引脚接收的 RZI 红外数据解调为 NRZ 串口数据，然后输出给标准 USART 模块。

在正常模式下，USART 所发送的脉冲宽度为一个位周期的 3/16。IrDA 规范要求脉冲宽度需要大于 1.41us。接收器端的干扰检测逻辑会滤除宽度小于 2 个 PSC 周期的脉冲（PSC 是在 USART_GTPR 中编程的预分频器值 PSC[7:0]）。宽度小于 1 个 PSC 周期的脉冲都将被拒绝，但宽度大于 1 个而小于 2 个周期的脉冲可能被接受也可能被拒绝，而宽度大于 2 个周期的脉冲将被接受作为有效脉冲。当 PSC=0 时，IrDA 编码器/解码器不工作。

在低功耗模式下，脉冲宽度不再保持为位周期的 3/16。此时的脉冲宽度为低功耗波特率的 3 倍，最小可为 1.42MHz。通常此值是 1.8432MHz（1.42MHz<PSC<2.12MHz）。接收时与正常模式下的工作方式一致。

红外是一个半双工通信协议。当发送数据时，即标准 USART 模块发送数据给编码器，解码器会忽略 RX 引脚上的数据。当接收数据时，即标准 USART 模块接收来自解码器的数据，编码器不会编码标准 USART 模块发送的数据。

注意：宽度小于两个但大于一个 PSC 周期的脉冲可能被接受，也可能被拒绝。

接收器的建立时间应由软件进行管理。IrDA 物理层规范规定发送和接收之间至少要经过 10ms 的延迟（IrDA 是一个半双工协议）。

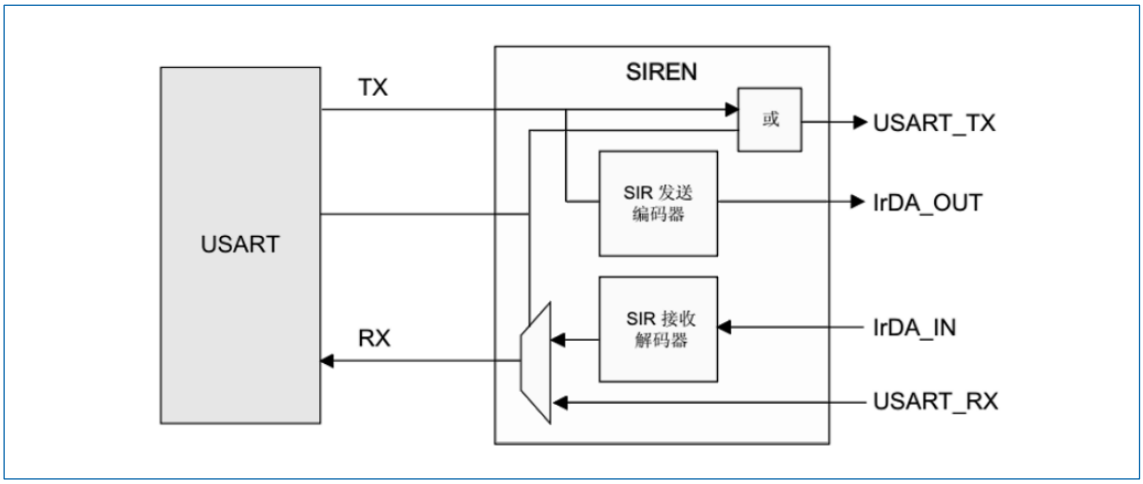


图 23-18 IrDA SIR ENDEC 框图

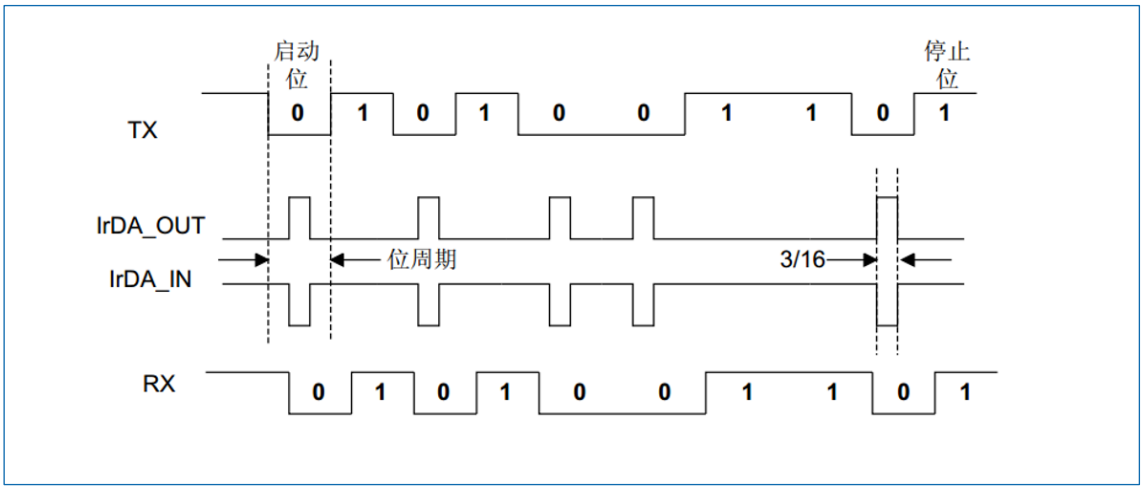


图 23-19 IrDA 数据调制（3/16）正常模式

23.4.15 DMA 模式下的 USART 连续通信

USART 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 USART_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“9 直接存储器访问控制器（DMA）”）加载到 USART_TDR 寄存器。

要配置一个 DMA 通道以进行 USART 发送，需要以下步骤操作：

1. 在 DMA 控制寄存器中写入 USART_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 USART_ICR 寄存器中的 TCCF 位置 1，将 USART_ISR 寄存器中的 TC 标志清零。
7. 在 DMA 寄存器中激活该通道。

USART 使用 DMA 发送的时序图如下：

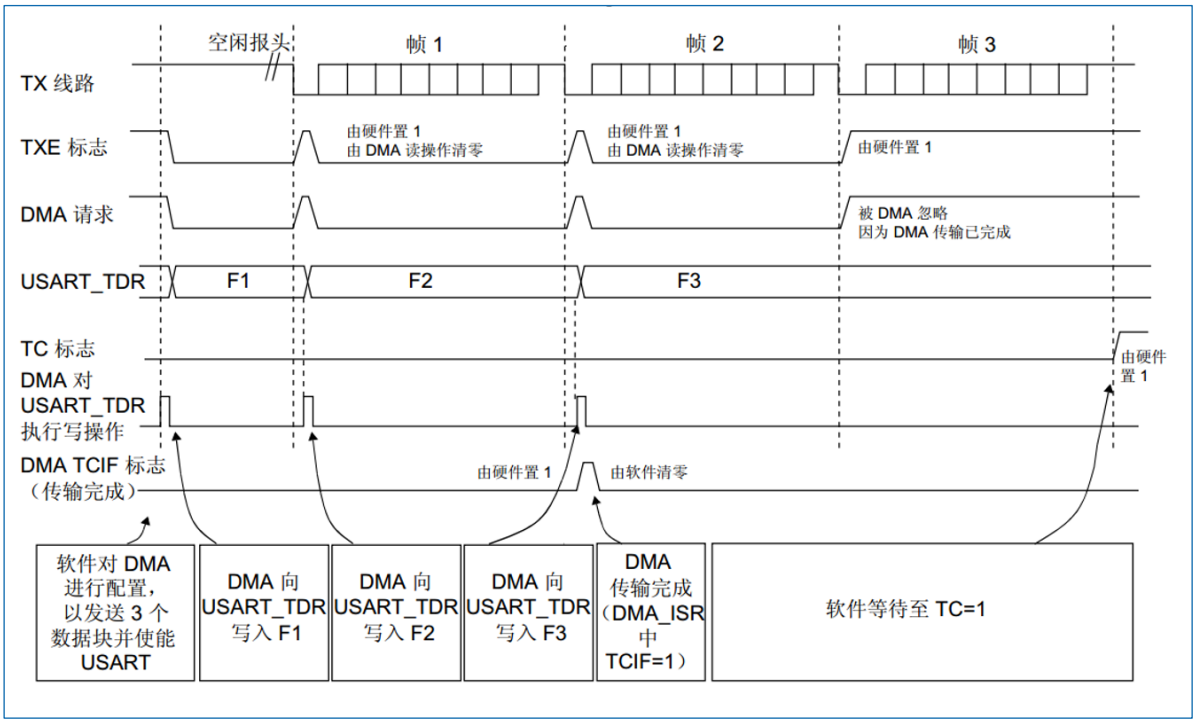


图 23-20 使用 DMA 进行发送

使用 DMA 进行接收

将 USART_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 USART_RDR 寄存器加载到 SRAM 区域中。

配置一个 DMA 通道以进行 USART 接收，操作步骤如下：

1. 在 DMA 控制寄存器中写入 USART_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。

USART 使用 DMA 接收的时序图如下：

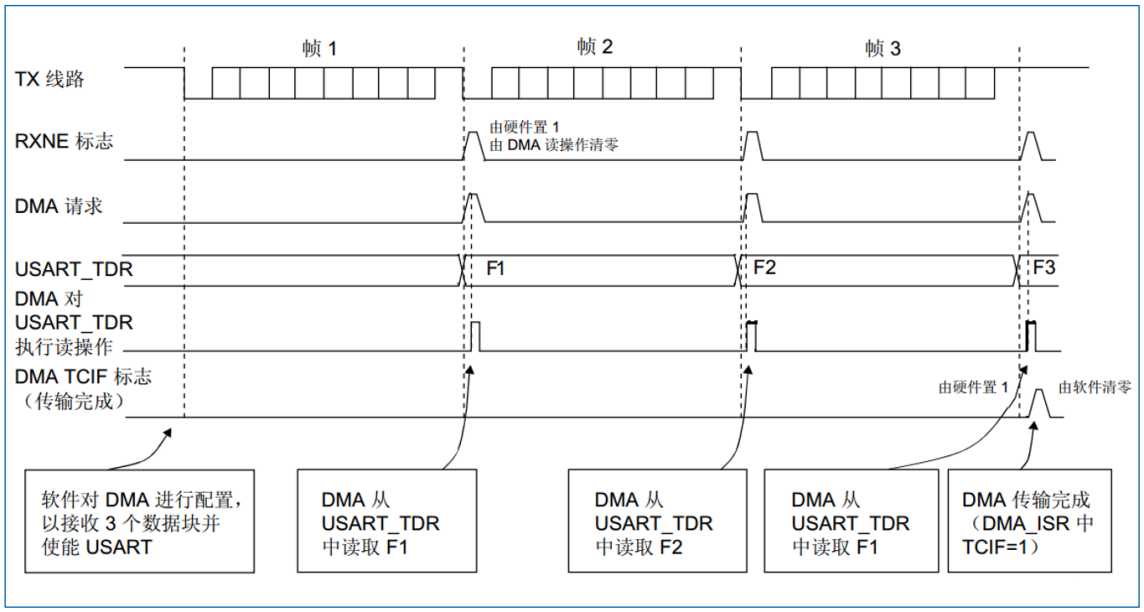


图 23-21 使用 DMA 进行接收

23.4.16 RS232 硬件流控制和 RS485 驱动器使能（使用 USART）

配置 USART_CR3 寄存器中的 RTSE 位和 CTSE 位，来分别使能 RS232RTS 和 CTS 流控制。CTS 输入和 RTS 输出可以控制 2 个器件间的串行数据流。

两个 USART 之间的硬件流控制连接图如下所示：

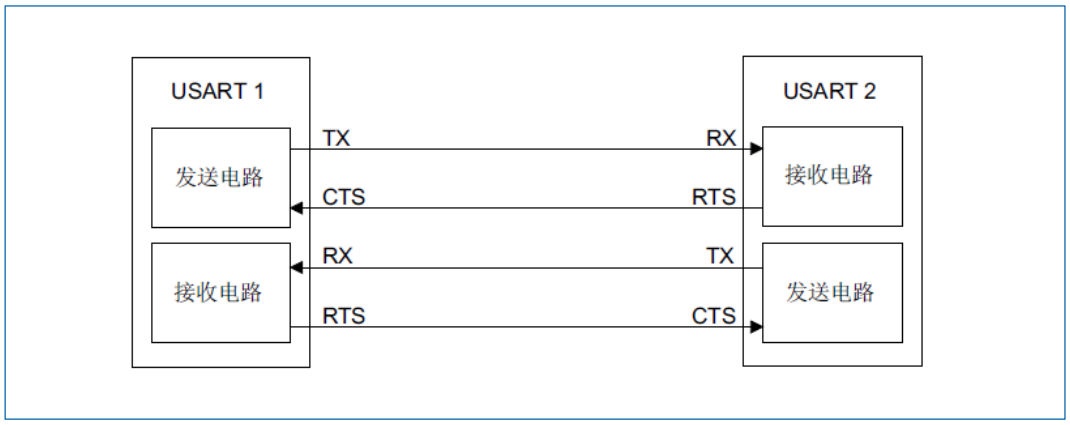


图 23-22 2 个 USART 间的硬件流控制

RS232RTS 流控制

在使能 RTS 流控制（RTSE=1）的情况下，只要 USART 接收器准备好接收新数据，便会驱动 RTS 输出低电平。当接收寄存器非空时，会驱动 RTS 输出高电平，以通知对端在完成当前数据帧发送之后停止发送。

RST 硬件流控制的时序图如下所示：

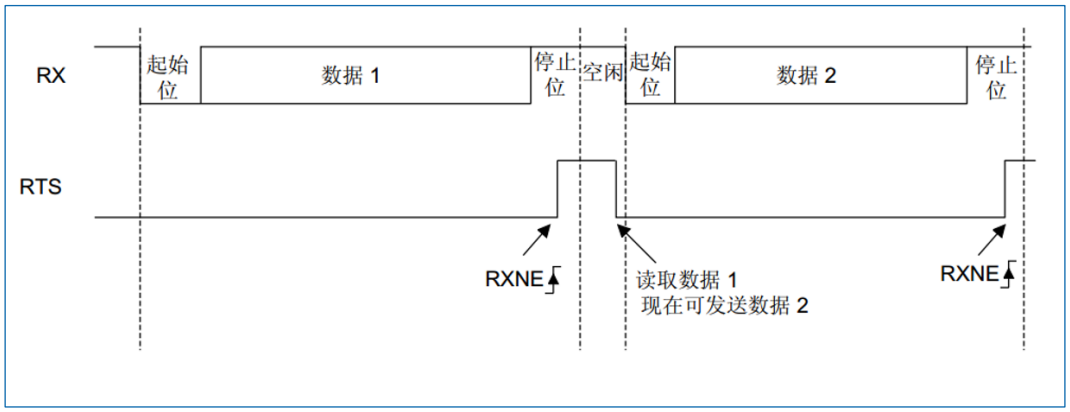


图 23-23 RS232RTS 流控制

RS232CTS 流控制

在使能 CTS 流控制（CTSE=1）的情况下，发送器会在发送下一帧数据前检查 CTS。

若 CTS 有效（输入低电平），则会发送下一数据；若 CTS 无效（输入高电平）则停止发送。若发送过程中 CTS 变为无效（高电平），则当前发送完成之后，发送器停止。

当 CTSE=1 时，只要 CTS 发生信号翻转，CTSIF 状态位便会由硬件自动置 1。此时若 USART_CR3 寄存器中的 CTSIE 位置 1，则会产生中断。在中断函数中检测 CTS 的输入信号，可知道对端是否已准备好进行通信。

下图是 CTS 流控制的时序：

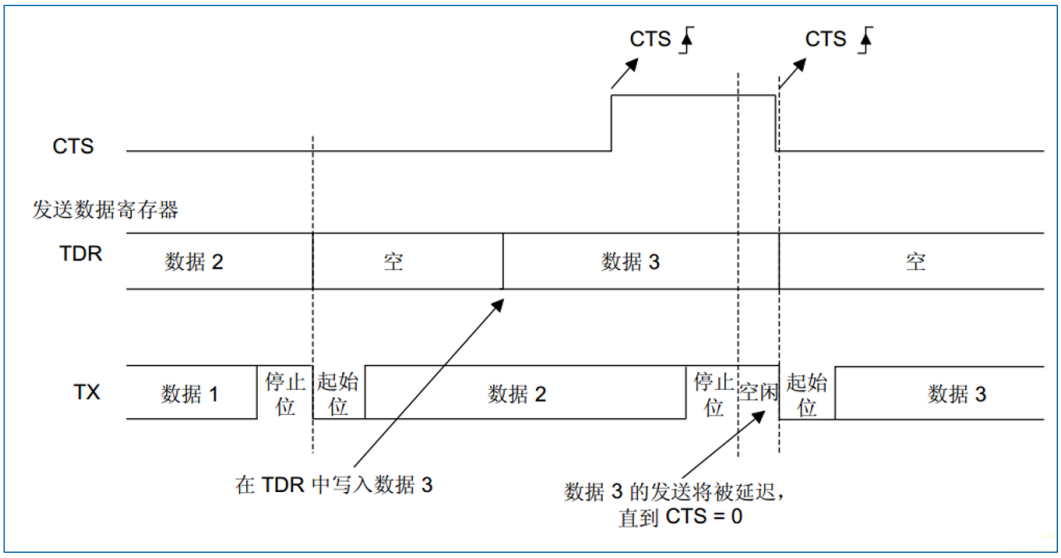


图 23-24 RS232CTS 流控制

RS485 驱动器使能

通过将 USART_CR3 控制寄存器中的 DEM 位置 1，可使能 DE 驱动器。可通过 DE（驱动器使能）信号激活外部收发器控制。

有效时间是指从激活 DE 信号与 START 位开始间的时间，可以通过 USART_CR1 控制寄存器中的 DEAT[4:0]位域来编程。

禁止时间是从完成停止位发送至取消激活 DE 信号之间的时间，可以通过 USART_CR1 控制寄存器中的 DEDT[4:0]位域来编程。

DE 信号的极性可通过 USART_CR3 控制寄存器中的 DEP 位配置。

DEAT 和 DEDT 以采样时间单位表示（1/8 持续时间对应 8 倍过采样，1/16 位持续时间对应 16 倍过采样）。

23.4.17 从停机模式唤醒

当 USART_CR1 寄存器中 UESM 位置 1 且 USART 时钟设置为 HSI 或 LSE 时，USART 能够将 MCU 从停机模式唤醒。

可使用标准 RXNE 中断将 MCU 从停机模式唤醒（必须在进入停机模式前将 RXNEIE 位置 1）。

也可以通过配置 USART_CR3 寄存器中的 WUS[1:0]域来选择唤醒方式，并且在检测到唤醒事件后，WUF 标志会被置 1（与 MCU 处于停机模式还是工作模式无关），若配置了 WUFIE 位，则产生中断。

若在初始化和使能接收器后立即进入停机模式，则必须校验 USART_ISR 寄存器的 REACK 位是否被置位，用于确认 USART 是否已经准备好接收数据。

在进入停机模式前，用户必须确保 USART 未在执行传输。

当 DMA 用于接收时，它必须在进入停机模式前禁止，并在退出停机模式后重新使能。

若 USART 在进入停机模式前处于静默模式：只能使用地址匹配唤醒。

从停机模式唤醒功能并非在所有模式下均可用。例如，该功能在同步模式下不起作用。

允许从停机模式正确唤醒的最大波特率为 1/23.31μs=42K 波特率。

23.5 USART 低功耗模式

表 23-7 低功耗模式对 USART 的影响

模式	说明
睡眠模式	无影响。USART 中断可使 MCU 退出睡眠模式。
停机模式	当 UESM 位置 1 且 USART 时钟设置为 HSI 或 LSE 时，USART 能够将 MCU 从停机模式唤醒。
待机模式	USART 掉电，退出待机模式，必须重新初始化 USART。

23.6 USART 中断

表 23-8 USART 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
CTS 中断	CTSIF	CTSIE
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
LIN 断路	LBDF	LBDIE
噪声错误、上溢错误和帧错误	NF 或 ORE 或 FE	EIE
字符匹配	CMF	CMIE
接收器超时	RTOF	RTOIE

中断事件	事件标志位	使能控制位
块结束	EOBF	EOBIE
从停机模式唤醒	WUF (1)	WUFIE

(1). WUF 中断仅在停机模式下有效。

USART 中断映射图（请参见图 23-25）。

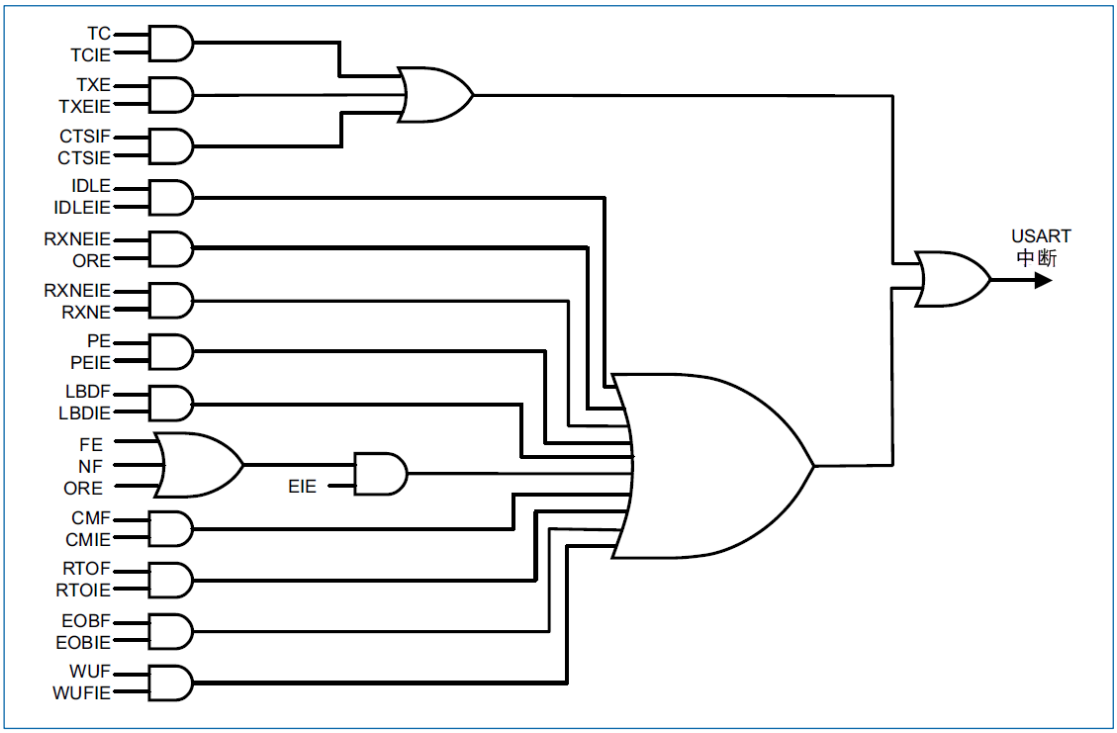


图 23-25 USART 中断映射图

23.7 USART 寄存器

基地址： 0x4001 3800

空间大小：0x400

23.7.1 控制寄存器 1（USART_CR1）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			M1	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]				
			rw	rw	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	UESM	UE
rw	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，M1=M[1]（Word length） • 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]=1x：保留
位 27	EOBIE：块尾中断使能（End of Block interrupt enable） 该位由软件置 1 和清零。 • 0：中断禁止 • 1：当 USART_ISR 寄存器中的 EOBF 标志被置 1 时，引发 USART 中断。
位 26	RTOIE：接收超时中断使能（Receiver timeout interrupt enable） 由软件置 1 和清零。 • 0：中断禁止 • 1：当 USART_ISR 寄存器中的 RTOF 标志被置 1 时，引发 USART 中断。
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 USART 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 USART_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 USART 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 USART 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 USART_ISR 寄存器中的 CMF 标志被置 1 时引发 USART 中断。
位 13	MME：静默模式使能（Mute mode enable） 该位开启 USART 的静默模式功能。当置为 1 时，USART 可以在活动模式和静默模式之间切换，和 WAKE 位的定义一样，由软件置 1 和清零。 • 0：接收器长期处于活动模式。 • 1：接收器可以在活动模式和静默模式间切换。
位 12	M0：字长，M0=M[0]（Word length） 该位决定串口字长。由软件置 1 和清零。 • 0：1 个起始位，8 位数据位，n 个停止位。 • 1：1 个起始位，9 个数据位，n 个停止位。

	该位域只能在 USART 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 USART 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 USART 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 USART 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 USART 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 USART_ISR 寄存器中的 PE 被置 1 的时候会产生 USART 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 USART_ISR 寄存器中的 TXE 被置 1 的时候会产生 USART 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 USART_ISR 寄存器中的 TC 位被置 1 的时候会产生 USART 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 USART_ISR 寄存器中的 ORE 或者 RXNE 被置 1 的时候会产生 USART 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 USART_ISR 寄存器中的 IDLE 位被置 1 的时候会产生 USART 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	UESM：USART 在 Stop 模式下使能（USART enable in Stop mode） 当该位为零，USART 不能够将 MCU 从 Stop 模式下唤醒。当该位为 1 时，USART 可以将 MCU 从 Stop 模式唤醒，条件是 USART 的时钟选择为 HSI 或 LSE。该位由软件置 1 和清零。 • 0：USART 不能从 Stop 模式中唤醒 MCU。 • 1：USART 可以从 Stop 模式中唤醒 MCU。当这个功能被打开，USART 的时钟源必须为 HSI 或者是 LSE。
位 0	UE：USART 使能（USART enable） 当该位被清零，USART 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 USART 的设置都不会丢，但 USART_ISR 中所有的状态标志都会被复位。由软件置 1 和清零。 • 0：USART 预分频器和输出关闭，低功耗模式。 • 1：USART 开启

23.7.2 控制寄存器 2（USART_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
ADD[7:4]				ADD[3:0]				RTOEN	ABRMOD [1:0]		ABREN	MSB FIRST	DATAINV	TXINV	RXINV
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
SWAP	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	ADDM7	Res			
rw	rw	rw		rw	rw	rw	rw		rw	rw	rw				

位 31:28	ADD[7:4]：USART 的节点地址（Address of the USART node） 该位域给出 USART 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（例如，在 ModBus 协议下对块尾的检测）。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 USART 被关闭的时候（UE=0）才能改写。
位 27:24	ADD[3:0]：USART 的节点地址（Address of the USART node） 该位域给出 USART 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。 该位域只能在接收器被关闭（RE=0）或者在 USART 被关闭的时候（UE=0）才能改写。
位 23	RTOEN：接收器超时检测功能使能（Receiver timeout enable） 该位由软件置 1 和清零。 • 0：接收器超时检测功能关闭 • 1：接收器超时检测功能开启 当这个功能开启，只要 RX 线发现空闲（不是接收）达到 RTOR 寄存器（接收超时寄存器）中设置的时间长度后，该位会被硬件置 1。
位 22:21	ABRMOD[1:0]：自动波特率检测模式（Auto baud rate mode） 该位域由软件设置或清零。 • 00：对起始位的测量被用来检测波特率。 • 01：使用下降沿对下降沿的测量。（接收数据必须以单个的 1 作为开头，帧格式为 Start10xxxxxx） • 10：保留 • 11：保留 该位域只能在 ABREN=0 或者 USART 未被使能的时候（UE=0）改写。
位 20	ABREN：自动波特率检测使能（Auto baud rate enable） 该位由软件置 1 和清零。 • 0：自动波特率检测被禁止。 • 1：自动波特率检测被打开。
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 USART 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 USART 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 USART 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 USART 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 USART 口进行交叉互联的时候。 该位只能在 USART 未被使能的时候（UE=0）改写。
位 14	LINEN：LIN 模式使能（LIN mode enable） 该位由软件置 1 和清零。 • 0：LIN 模式禁止 • 1：LIN 模式使能 LIN 模式打开后，可以具备发送 LIN 同步断开（13 个低位）的功能，用 USART_CR1 寄存器的 SBKRQ 位实现，同时还具备 LIN 同步断开信号的检测功能。 该位只能在 USART 未被使能的时候（UE=0）改写。
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 • 11：1.5 个停止位 该位域只能在 USART 未被使能的时候（UE=0）改写。
位 11	CLKEN：时钟使能（Clock enable） 该位用来开启 SCLK 引脚的功能。 • 0：SCLK 引脚被禁止 • 1：SCLK 引脚被使能 该位域只能在 USART 未被使能的时候（UE=0）改写。
位 10	CPOL：时钟极性（Clock polarity） 该位允许用户选择同步模式下 SCLK 引脚上的时钟输出的极性。它连同 CPHA 位一起决定所需要的时钟/数据时序关系。

	• 0：在没有数据传输的时候保持低电平。 • 1：在没有数据传输的时候保持高电平。 该位只能在 USART 未被使能的时候（UE=0）改写。
位 9	CPHA：时钟相位（Clock phase） 该位允许用户选择同步模式下 SCLK 引脚上的时钟输出的相位。它连同 CPOL 位一起决定所需要的时钟/数据时序关系。（参见图 23-13 和图 23-14） • 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 该位只能在 USART 未被使能的时候（UE=0）改写。
位 8	LBCL：末位时钟脉冲（Last bit clock pulse） 该位用来选择在同步模式下，SCLK 脚上传输最后一个位（MSB）的时候是否必须输出时钟脉冲。 • 0：SCLK 脚上在传输末位数据的时候不输出时钟脉冲。 • 1：SCLK 脚上在传输末位数据的时候输出时钟脉冲。 *警告：末位是第 8 位还是第 9 位，取决于 USART_CR1 寄存器中的 M 位的设置。该位域只能在 USART 未被使能的时候（UE=0）改写。*
位 7	Res：保留 必须保持复位值。
位 6	LBDIE：LIN 断开信号检测中断使能（LIN break detection interrupt enable） 断开中断屏蔽（利用断开分隔符来检测）。 • 0：中断禁止 • 1：在 USART_ISR 寄存器中的 LBDF 被置 1 的时候会产生 USART 中断。
位 5	LBDL：LIN 断开检测长度（LIN break detection length） 该位用来选择使用 11 位还是 10 位断开检测。 • 0：10 位断开检测 • 1：11 位断开检测 该位只能在 USART 未被使能的时候（UE=0）改写。
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 USART 未被使能的时候（UE=0）改写。
位 3:0	Res：保留 必须保持复位值。

23.7.3 控制寄存器 3（USART_CR3）

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									WUFIE	WUS[1:0]		SCARCNT[2:0]			Res
									rw	rw		rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEP	DEM	DDRE	OVRDIS	ONEBIT	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
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	WUFIE：从 Stop 模式唤醒中断使能（Wakeup from Stop mode interrupt enable） 由软件置 1 和清零。 • 0：中断禁止 • 1：在 USART_ISR 寄存器中的 WUF 被置 1 的时候会产生 USART 中断。
位 21:20	WUS[1:0]：从 Stop 模式唤醒中断标志（Wakeup from Stop mode interrupt flag selection） 选择该位域指定激活 WUF 标志的事件。 • 00：在发生地址匹配事件的时候激活 WUF • 01：保留 • 10：WUF 在检测到起始位的时候激活 • 11：WUF 在得到接受数据寄存器非空事件时激活 该位域只能在 USART 未被使能的时候（UE=0）改写。
位 19:17	SCARCNT[2:0]：智能卡模式重试计数器（Smartcard auto-retry count） 该位域指定智能卡模式中接收和发送的重试次数。在发送模式下，它指定的是在产生发送错误（FE=1）之前自动重试发送的次数。在接收模式下，它指定的是在产生接收错误事件前（RXNE 和 PE=1）所作的错误接收的尝试的次数。 该位域只能在 USART 未被使能的时候（UE=0）改写。 当 USART 被使能，该位域只允许写成 0x0，这是为了避免盲目的自动重发数据。 • 0x0：重发功能关闭-在发送模式下不进行自动重发操作。 • 0x1-0x7：自动重传的尝试次数（在提示错误之前）。
位 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 USART 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。DE 信号在 RTS 脚输出。 该位只能在 USART 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。（用于智能卡模式） • 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 USART 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 USART_RDR 中以前的内容覆盖掉。 该位只能在 USART 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 USART 未被使能的时候（UE=0）改写。
位 10	CTSIE：CTS 中断使能（CTS interrupt enable） • 0：中断禁止 • 1：在 USART_ISR 寄存器中的 CTSIF 被置 1 的时候会产生 USART 中断。
位 9	CTSE：CTS 功能使能（CTS enable） • 0：关闭 CTS 硬件流控 • 1：打开 CTS 硬件流控，只有在 nCTS 输入上收到有效信号（被拉低）时才会发送数据。如果在数据传送时 nCTS 输入变为无效，会在数据发送完成后再停。如果写数据到发送寄存器的时候，nCTS 处于无效状态，那么这个数据的发送会被推迟直到 nCTS 信号变成有效。 该位只能在 USART 未被使能的时候（UE=0）改写。
位 8	RTSE：RTS 使能（RTS enable） • 0：关闭 RTS 硬件流控 • 1：RTS 输出使能，只会在有接收空间的时候才请求下一个数据。当前数据发送完成后，发送操作就需要暂停下来。如果可以接收数据了，将 nRTS 输出置为有效（拉

	低）。该位域只能在 USART 未被使能的时候（UE=0）改写。
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	SCEN：智能卡模式使能（Smartcard mode enable） 该位用于打开智能卡模式。 • 0：关闭智能卡模式 • 1：打开智能卡模式 该位只能在 USART 未被使能的时候（UE=0）改写。
位 4	NACK：智能卡 NACK 发送使能（Smartcard NACK enable） • 0：出现校验错误的时候，不发送 NACK。 • 1：出现校验错误的时候，发送 NACK。 该位只能在 USART 未被使能的时候（UE=0）改写。
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 USART 未被使能的时候（UE=0）改写。
位 2	IRLP：IrDA 低功耗模式选择（IrDA low-power） 该位用来选择 IrDA 的普通模式还是低功耗模式。 • 0：正常模式 • 1：低功耗模式 该位只能在 USART 未被使能的时候（UE=0）改写。
位 1	IREN：红外模式使能（IrDA mode enable） 该位由软件置 1 和清零。 • 0：不使能红外模式 • 1：使能红外模式 该位只能在 USART 未被使能的时候（UE=0）改写。
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 USART_ISR 寄存器中的 FE=1 或 ORE=1 或 NF=1 时，会产生中断。

23.7.4 波特率寄存器（USART_BRR）

偏移地址：0x0C

复位值：0x0000 0000

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

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

位 31：16	Res：保留 必须保持复位值。
位 15:4	BRR[15:4]：USARTDIV 的高 12 位（High 12 bits of USARTDIV） 这 12 位定义 USART 分频器除法因子的整数部分 USARTDIV[15:4]。
位 3:0	BRR[3:0]：USARTDIV 的低 4 位（Low 4 bits of USARTDIV） 当 OVER8=0，BRR[3:0]=USARTDIV[3:0]。当 OVER8=1，BRR[3:0]=USARTDIV[3:0]右移 1 位。

23.7.5 保护时间和预分频器寄存器（USART_GTPR）

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
GT[7:0]								PSC[7:0]							
rw								rw							

位 31:16	Res：保留 必须保持复位值。
位 15:8	GT[7:0]：保护时间值（Guard time value） 该位域用来设置保护时间长度，以波特时钟为单位。在智能卡模式下会用到。在保护时间过去之后才会设置发送完成标志。该位域只能在 USART 未被使能的时候（UE=0）改写。
位 7:0	PSC[7:0]：预分频器值（Prescaler value） • 在红外低功耗和正常模式下：PSC[7:0]=IrDA 正常和低功耗模式波特率对 USART 时钟源进行分频以获得低功耗模式下的频率：时钟源按寄存器给定的值来分频（8 位有效）。 ◦ 00000000：保留

	○ 00000001：1 分频 ○ 00000010：2 分频 • … • 智能卡模式：PSC[4:0]：预分频器值 该位域用于设定对 USART 时钟源的分频系数，得到智能卡时钟。按寄存器中的值（低 5 位有效）乘以 2 得到的值作为分频系数来分频： ○ 00000：保留 ○ 00001：2 分频 ○ 00010：4 分频 ○ 00011：6 分频 • … 该位域只能在 USART 未被使能的时候（UE=0）改写。

23.7.6 接收超时寄存器（USART_RTOR）

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
BLEN[7:0]								RTO[23:16]							
rw								rw							

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

位 31:24	BLEN[7:0]：块长度（Block Length） 该位域给出了智能卡模式 T=1 的接收时的块长度。这个值等于信息块字符数+结束部分（1-LEC/2-CRC）-1。例如： • BLEN=0 -> 0 个信息字符+LEC • BLEN=1 -> 0 个信息字符+CRC • BLEN=255 -> 254 个信息字符+CRC（总共 256 个字符） 智能卡模式中，当 TXE=0，会导致块长度计数器清零。 该位域也可以在其它模式中使用。这时，块长度计数器在 RE=0 的时候清零和/或在 EOBCF 位被写 1 的时候清零。
位 23:0	RTO[23:0]：接收超时值（Receiver timeout value） 该位域给出了接收超时的设置，单位是波特时钟的时长。 标准模式下，最后一个字节接收后，在整个 RTO 值规定的时长内，再也没有检测到新的起始位的时候，RTOF 标志会被硬件置 1。 在智能卡模式中，这个值被用来实现 CWT 和 BWT。详细信息参见“23.4.13 USART 智能卡模式”。这里，超时测量时从最后一个字节的其实位开始算的。

23.7.7 请求寄存器（USART_RQR）

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											TXFRQ	RXFRQ	MMRQ	SBKRQ	ABRRQ
											w	w	w	w	w

位 31:5	Res：保留 必须保持复位值。
位 4	TXFRQ：发送数据清空请求（Transmit data flush request） 对该位写 1 会直接令 TX 标志被硬件置 1。 这里允许取消数据发送。该位只能在智能卡模式下使用，当数据由于所发生的错误导致还没发出去，USART_ISR 寄存器的 FE 标志是 1 的时候用。 如果 USART 不支持智能卡模式，该位为保留位并由硬件强制为零。
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 USART 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	ABRRQ：自动波特率检测请求（Auto baud rate request） 向该位写 1 会清除 USART_ISR 中的 ABRF 标志，并在下一次数据接收的时候开始一次自动波特率测量。

23.7.8 中断和状态寄存器（USART_ISR）

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	WUF	RWU	SBKF	CMF	BUSY
									r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABRF	ABRE	Res	EOBF	RTOF	CTS	CTSIF	LBDF	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
r	r		r	r	r	r	r	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 USART 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 USART 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 USART 读到的时候，会给出一个回执。这可以在写 USART_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	WUF：从 Stop 模式唤醒标志（Wakeup from Stop mode flag） 当检测到唤醒事件时由硬件置 1。具体时间由 WUS 位域来定义。由软件向 USART_ICR 寄存器的 PECF 位写 1，可以清除这个标志。如果 USART_CR3 寄存器的 WUFIE=1，会产生中断请求。
位 19	RWU：接收器从静默模式唤醒（Receiver wakeup from Mute mode） 该位表示 USART 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 USART_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 USART_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 USART_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 USART_ICR 寄存器的 CMCF 位写 1，可以清除这个标志。 如果 USART_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：USART 处于空闲（无接收） • 1：正在接收数据
位 15	ABRF：自动波特率检测标志（Auto baud rate flag） 当自动波特率功能打开时（RXNE 也被置 1，如果 RXNEIE=1 产生中断）或者当自动波特率操作不成功时（ABRE，RXNE 和 FE 也都被置 1 的时候），该位被硬件置 1。 开始一轮新的波特率检测（向 USART_RQR 寄存器的 ABRRQ 写 1）的时候会被清除。
位 14	ABRE：自动波特率检测错误（Auto baud rate error） 在波特率测量失败的时候（超量程或者字符比较失败）由硬件置 1。由软件清零，方法

	是向 USART_CR3 寄存器的 ABRRQ 位写 1。
位 13	Res：保留 必须保持复位值。
位 12	EOBF：块结束标志（End of block flag） 当一个完整的块被接收时由硬件置 1（例如 T=1 智能卡模式中）。当收到的字节数（从块开始，包括序言部分）大于等于 BLEN+4 的时候完成检测。如果 USART_CR2 寄存器的 EOBIE=1，会产生中断请求。由软件向 USART_ICR 寄存器的 EOBCF 位写 1，可以清除这个标志。 • 0：未到块结束 • 1：块结束（足够字节数）已到
位 11	RTOF：接收超时标志（Receiver timeout） 如果没有通讯的时间长度达到了 RTOR 寄存器中设定的超时值，该位会被硬件置 1。由软件向 USART_ICR 寄存器的 RTOCF 位写 1，可以清除这个标志。如果 USART_CR2 寄存器的 RTOIE=1，会产生中断请求。在智能卡模式中，这个超时相当于 CWT 或 BWT 计时。 • 0：尚未超时 • 1：已经达到超时
位 10	CTS：CTS 标志（CTS flag） 该位由硬件置 1 和清零。这是 nCTS 输入脚状态的反向拷贝。 • 0：nCTS 线是高 • 1：nCTS 线是低
位 9	CTSIF：CTS 中断标志（CTS interrupt flag） 如果 CTSE 位为 1，那么当 nCTS 输入状态变化的时候，由硬件置 1。由软件向 USART_ICR 寄存器的 CTSCF 位写 1，可以清除这个标志。如果 USART_CR3 寄存器的 CTSIE=1，会产生中断请求。 • 0：nCTS 线的状态无变化 • 1：nCTS 线的状态有变化
位 8	LBDF：LIN 断开检测（LIN break detection flag） 当检测到 LIN 断开字符时，该位由硬件置 1。由软件向 USART_ICR 寄存器的 LBDCF 位写 1，可以清除这个标志。如果 USART_CR2 寄存器的 LBDIE=1，会产生中断请求。 • 0：没检测到 LIN 断开字符 • 1：检测到 LIN 断开字符
位 7	TXE：发送数据寄存器空（Transmit data register empty） 当 USART_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 USART_TDR 寄存器写数据就会同时清掉该位。TXE 标志还可以用其它方式清除，例如向 USART_RQR 寄存器的 TXFRQ 位写 1，为了丢弃数据（仅于智能卡模式 T=0，传送失败的情形）。如果 USART_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 USART_CR1 寄存器中的 TCIE 位是 1，就会产生中断请求。向 USART_TDR 寄存器再次写入数据，或者向 USART_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 USART_RDR 寄存器中时，该位被硬件置 1。读取 USART_RDR 寄存器的数据就会同时清掉该位。RXNE 标志也可以通过对 USART_RQR 寄存器中的 RXFRQ 位写 1 来清除。如果 USART_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 USART_CR1 寄存器中的 IDLEIE 位是 1，就会产生中断请求。由软件向 USART_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 USART_ICR 寄存器的 ORECF 位写 1，可以清除这个标志。如果 USART_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 USART_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 USART_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 USART_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 USART_ICR

	寄存器的 PECF 位写 1，可以清除这个标志。如果 USART_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

23.7.9 中断标志清除寄存器（USART_ICR）

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											WUCF	Res		CMCF	Res
											w_r1			w_r1	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			EOBCF	RTOCF	Res	CTSCF	LBDCF	Res	TCCF	Res	IDLECF	ORECF	NCF	FECF	PECF
			w_r1	w_r1		w_r1	w_r1		w_r1		w_r1	w_r1	w_r1	w_r1	w_r1

位 31:21	Res：保留 必须保持复位值。
位 20	WUCF：从 Stop 模式唤醒标志的清除（Wakeup from Stop mode clear flag） 对该位写 1，会清除 USART_ISR 寄存器中的 WUF 标志位。
位 19:18	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 USART_ISR 寄存器中的 CMF 标志位。
位 16:13	Res：保留 必须保持复位值。
位 12	EOBCF：块结束标志的清除（End of block clear flag） 对该位写 1，会清除 USART_ISR 寄存器中的 EOBF 标志位。
位 11	RTOCF：接收超时标志的清除（Receiver timeout clear flag） 对该位写 1，会清除 USART_ISR 寄存器中的 RTOF 标志位。
位 10	Res：保留 必须保持复位值。
位 9	CTSCF：CTS 标志的清除（CTS clear flag） 对该位写 1，会清除 USART_ISR 寄存器中的 CTSIF 标志位。
位 8	LBDCF：LIN 断开检测标志的清除（LIN break detection clear flag） 对该位写 1，会清除 USART_ISR 寄存器中的 LBDF 标志位。
位 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 USART_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 USART_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 USART_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 USART_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 USART_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 USART_ISR 寄存器中的 PE 标志位。

23.7.10 数据接收寄存器（USART_RDR）

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。

23.7.11 数据发送寄存器（USART_TDR）

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（USART_CR1 中的 PCE=1），向最高位（位 8 还是位 9 取决于设置的字长）写入的信息是无效的，因为它总是要被校验位代替之后再发送。

24 串行外设接口（SPI/I2S）

SPI/I2S 接口可用于使用 SPI 协议或 I2S 音频协议与外部器件进行通信。SPI 或 I2S 模式可通过软件进行选择。器件复位后默认选择 SPI 模式。

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。

集成电路内置音频总线（I2S）也是同步串行通信接口。它能够在从模式或主模式下作为接收器或发送器工作。它可满足四种不同音频标准的要求，包括 Philips I2S 标准、MSB 和 LSB 对齐标准以及 PCM 标准。

24.1 SPI 主要特性

- 主模式或从模式操作
- 基于三条线的全双工同步传输
- 基于双线的半双工同步传输，其中一条可作为双向数据线
- 基于双线的单工同步传输，其中一条可作为单向数据线
- 4 位到 16 位传输帧格式选择
- 多主模式功能
- 8 个主模式波特率预分频器，可达 $f_{PCLK}/2$
- 从模式频率可达 $f_{PCLK}/2$
- 对于主模式和从模式都可通过硬件或软件进行 NSS 管理：动态切换主/从操作
- 可编程的时钟极性和相位
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 支持 SPI Motorola 模式
- 确保可靠通信的硬件 CRC 功能
  - 在发送模式下可将 CRC 值作为最后一个字节发送。
  - 根据收到的最后一个字节自动进行 CRC 错误校验。
- 可触发中断的主模式故障和上溢标志
- CRC 错误标志
- 2 个 32 位嵌入式 TX 和 RX FIFO 带 DMA 功能
- 支持 SPI TI 模式

24.2 I2S 功能

- 半双工通信（仅作为发送器或接收器）
- 主模式或从模式操作
- 8 位可编程线性预分频器，可实现精确的音频采样频率（从 8kHz 到 192kHz）。
- 数据格式可以是 16 位、24 位或 32 位
- 数据包帧由音频通道固定为 16 位（可容纳 16 位数据帧）或 32 位（可容纳 16 位、24 位、32 位数据帧）

- 可编程的时钟极性（就绪时的电平状态）
- 发送模式下的下溢标志（仅从模式）、接收模式下的上溢标志（主模式和从模式），以及接收和发送模式下的帧错误标志（仅从模式）
- 发送和接收使用同一个 16 位数据寄存器
- 支持的 I2S 协议
  - I2S Philips 标准
  - MSB 对齐标准（左对齐）
  - LSB 对齐标准（右对齐）
  - PCM 标准（在 16 位通道帧或扩展为 32 位通道帧的 16 位数据帧上进行短帧和长帧同步）
- 数据方向始终为 MSB 在前
- 用于发送和接收的 DMA 功能（16 位宽）
- 可输出主时钟以驱动外部音频元件。比率固定为 256×F_S（其中 F_S 为音频采样频率）

24.3 SPI/I2S 实现

表 24-1 SPI 实现

SPI 特性	SPI1
硬件 CRC 计算	支持
RX/TX FIFO	支持
I2S 模式	-
TI 模式	支持

24.4 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。

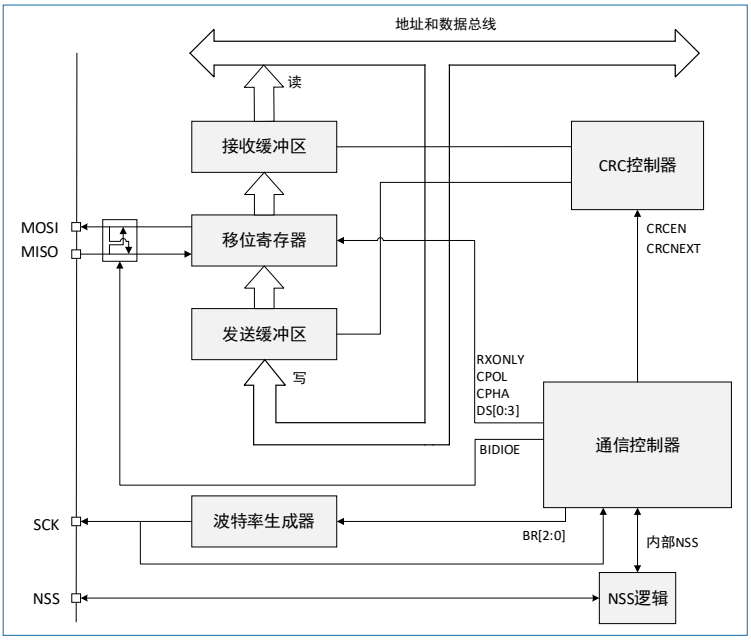


图 24-1 SPI 框图

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“24.4.4 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。

24.4.1 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。

24.4.1.1 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。

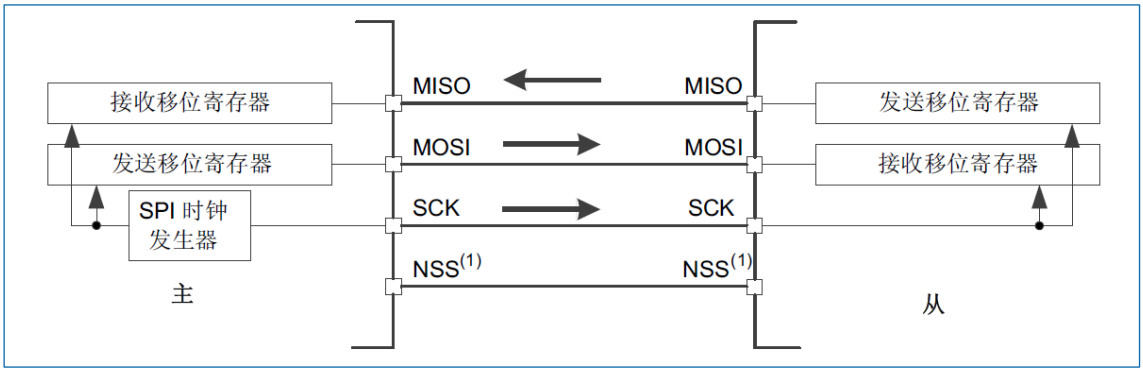


图 24-2 全双工单个主器件/单个从器件应用

(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“24.4.4 从器件选择（NSS）引脚管理”。

24.4.1.2 半双工通信

通过将 SPIx_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 SPIx_CR1 寄存器中的 BDIOE 位进行选择。

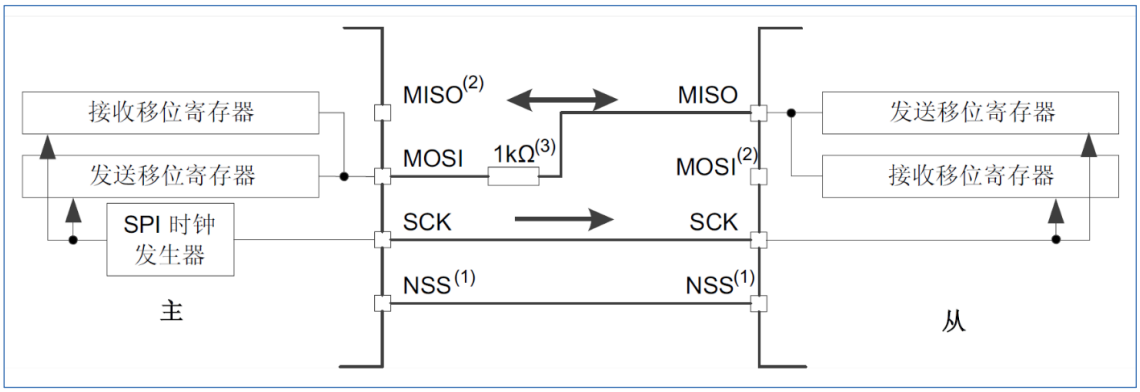


图 24-3 半双工单个主器件/单个从器件应用

(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“24.4.4 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。

24.4.1.3 单工通信

通过 SPIx_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“24.4.3 多主器件通信”）。基于数据缓冲区的配置产生接收数据事件。在主器件配置下，MOSI 输出被禁止，该引脚可用作 GPIO。只要 SPI 处于使能状态，则不断生成时钟信号。停止时钟的唯一方式是将 RXONLY 位或 SPE 位清零，直至来自 MISO 引脚的传入模式结束，然后基于相应配置填充数据缓冲区结构。

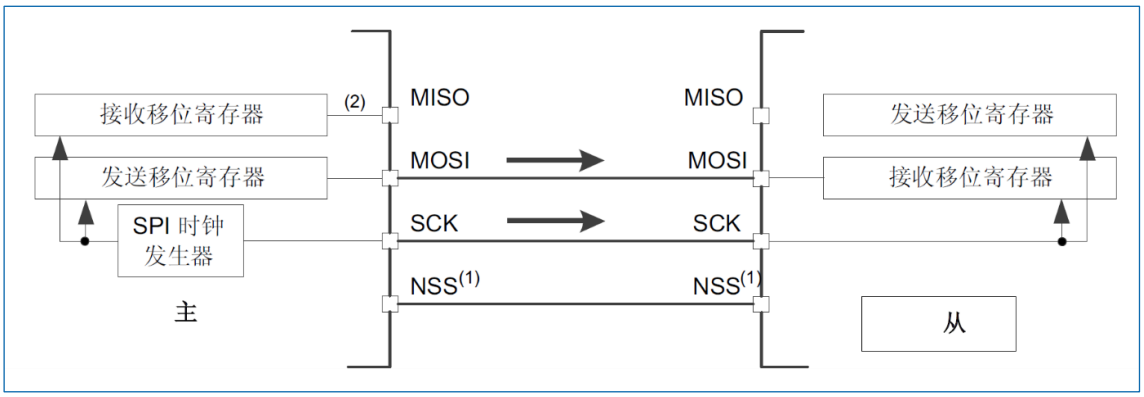


图 24-4 单工单个主器件/单个从器件应用（主器件为只发送模式/从器件为只接收模式）

(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“24.4.4 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）。

24.4.2 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 24-5）。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。

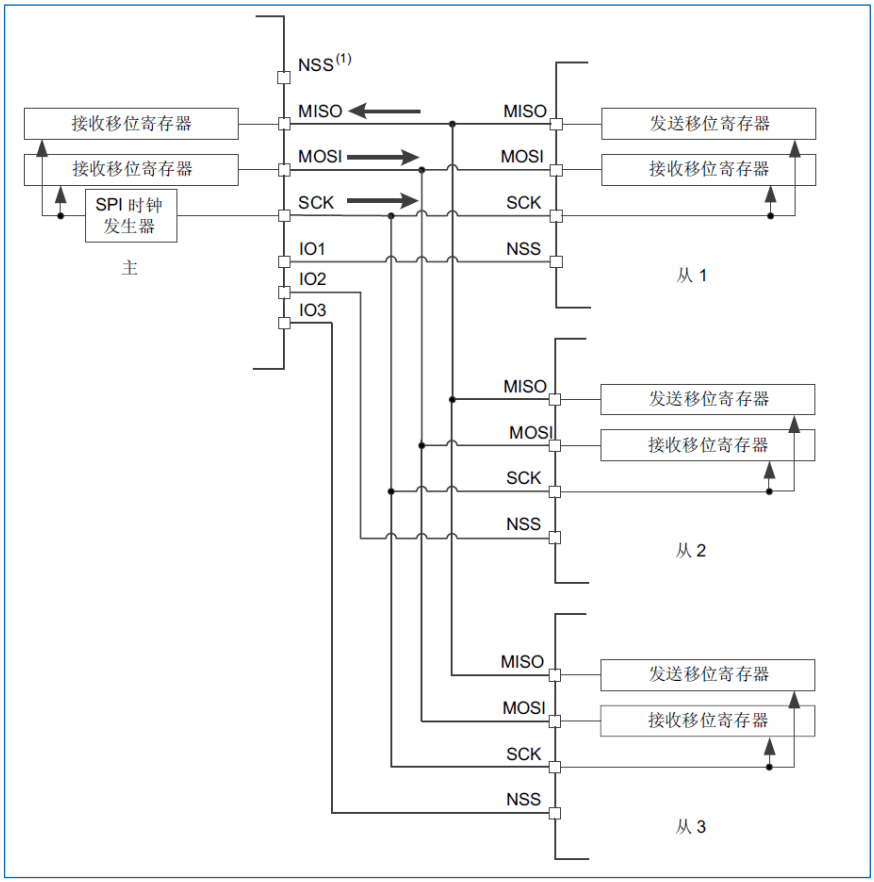


图 24-5 主器件和三个独立的从器件

(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“7.1.6 I/O 复用功能输入输出”）。

24.4.3 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。

由于此时只有一个结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“24.8.3 SPI 状态寄存器（SPI_SR）”中 MODF 位所指示的模式故障）。随后，用户可应用某个简单的仲裁过程（例如，在两个节点上施加不同的预定义超时来推迟下一个尝试）。

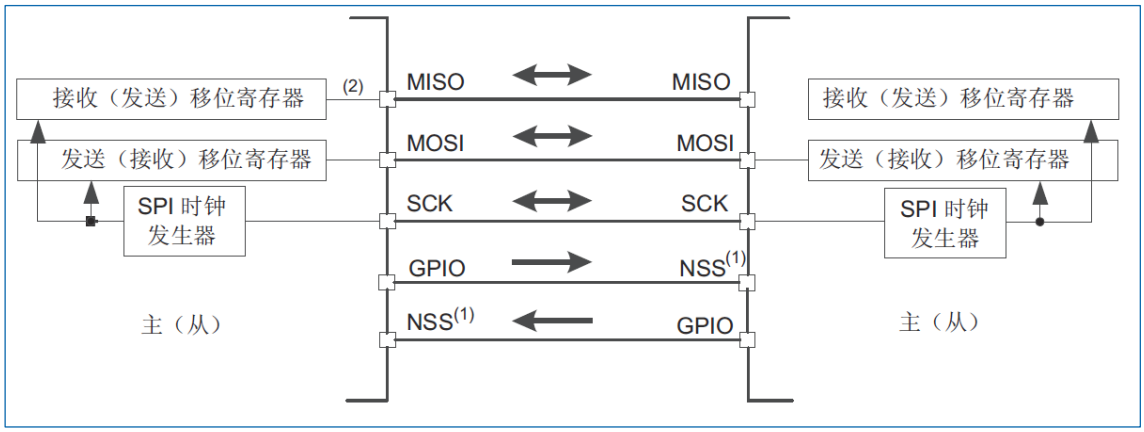


图 24-6 多主器件应用

(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。

24.4.4 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 SPIx_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 SPIx_CR1 寄存器中的 SSI 位的值决定内部驱动从器件选择信息。外部 NSS 引脚空闲，可供其它应用使用。
- 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（SPIx_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。

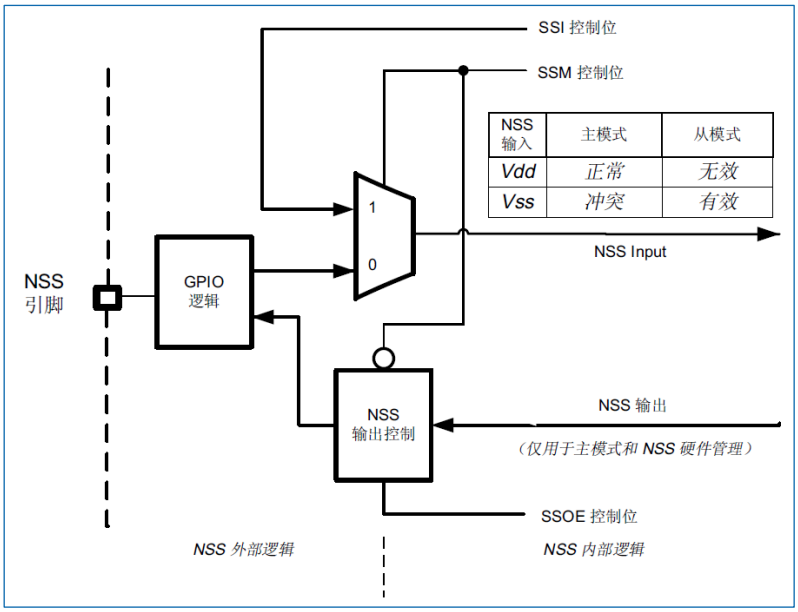


图 24-7 硬件/软件从器件选择管理

24.4.5 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。

24.4.5.1 时钟相位和极性控制

通过 SPIx_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 24-8 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 SPIx_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）。

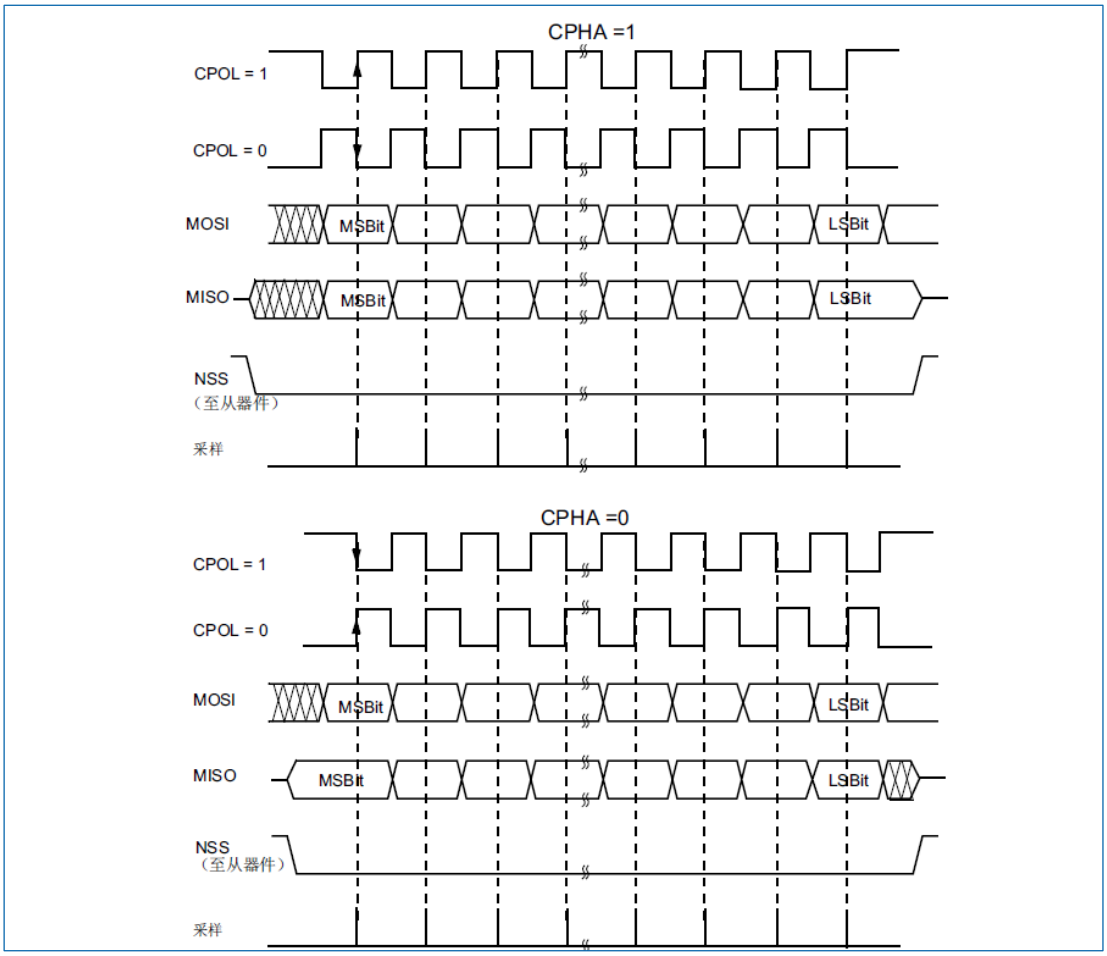


图 24-8 数据时钟时序图

说明：数据位的顺序取决于 LSBFIRST 位的设置。

24.4.5.2 数据帧格式

SPI 移位寄存器可设置为以 MSB 在前或 LSB 在前的方式移出数据，具体取决于 LSBFIRST 位的值。每个数据帧的长度均可配置为 4 位到 16 位，具体取决于 SPI_CR2 寄存器中的 DS 位编程的数据长度。所选的数据帧格式适用于发送和接收。无论选定的数据帧大小如何，对 FIFO 的读访问必须与 FRXTH 位的配置对齐。当访问 SPIx_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。

24.4.6 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 SPIx_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 SPIx_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 SPIx_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 SPIx_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 SPITx 和 Rx 专用的 DMA 数据流。

注释：
(1). 从模式下无需此步骤，但从器件在 TI 模式下工作时除外。
(2). TI 模式下无需此步骤。
(3). 从模式下无需此步骤。
(4). NSSP 模式下无需此步骤。

24.4.7 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“24.4.10 使用 DMA（直接存储器寻址）进行通信”。

24.4.8 数据发送和接收过程

24.4.8.1 RXFIFO 和 TXFIFO

SPI 数据传输都使用 32 位嵌入式 FIFO。这使 SPI 能够连续的工作，并且在数据帧较小时防止溢出。每个方向都有自己的 FIFO，称为 TXFIFO 和 RXFIFO。除了启用 CRC 计算的仅接收器模式（从或主）外，这些 FIFO 用于所有 SPI 模式（参见“24.4.15 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“24.4.14 TI 模式”）。

对 SPIx_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 SPIx_DR 的写入访问将写入的数据存储在发送队列末尾的 TXFIFO 中。读取访问必须始终与 SPI_CR2 寄存器中的 FRXTH 位配置的 RXFIFO 阈值对齐。FTLVL[1:0]和 FRLVL[1:0]位表示两个 FIFO 的当前占用水平。

对 SPI_DR 寄存器的读取访问必须由 RXNE 事件管理。当数据存储在 RXFIFO 中且达到阈值（由 FRXTH 位定义）时，会触发此事件。清除 RXNE 时，RXFIFO 被视为空。以类似的方式，要传输的数据帧的写访问由 TXE 事件管理。TXFIFO 级别小于或等于其容量的一半时触发此事件。否则，TXE 被清除，TXFIFO 被视为已满。这样，RXFIFO 最多可以存储四个数据帧，而 TXFIFO 在数据帧格式不大于 8 位时最多只能存储三个数据帧。当软件试图以 16 位模式将更多数据写入 TXFIFO 时，此差异可防止 TXFIFO 中已存储的 3x8 位数据帧可能损坏。TXE 和 RXNE 事件都可以通过中断进行轮询或处理。

如果接收到下一个数据时 RXFIFO 的是满的，将导致发生溢出事件（参见“24.4.11 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。

24.4.8.2 序列处理

多个数据字节可以顺序发送以组成一个消息。启用发送后，在主机 TXFIFO 中的数据会开始发送并连续发完。主机会连续输出时钟信号，直至 TXFIFO 变为空，然后停下来等待新增的数据。

在单接收模式，半双工（BIDIMODE=1，BIDIOE=0），或只发送（BIDIMODE=0，RXONLY=1）模式下，只要 SPI 和只接收模式被使能，主机会立即开始接收序列。主机连续提供时钟信号，直到主机关闭 SPI 或者关闭只接收模式之前都不会停下来。这时主机会连续接收数据。

虽然主机可以以连续模式提供所有交互（SCK 信号是连续的），但它必须考虑从机在任何时候处理数据流及其内容的能力。必要时，主机必须减慢通信速度，并提供较慢的时钟或具有足够延迟的单独帧或数据会话。请注意，在 SPI 模式下，主设备或从设备没有下溢错误信号，并且来自从设备的数据始终由主设备进行处理，也就是说从设备无法在 SPI 模式下正确准备数据时间从机最好使用 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“24.4.4 从器件选择（NSS）引脚管理”）。

BSY 位被置 1 时，它标志着一个持续的传输。这一点，结合 FTLVL[1:0]位，可用于检查传输是否完成。在系统进入停机模式前这是很有必要的，过早的进入停机模式可能导致数据破坏。判断 BSY 位的另外一个目的是可以用来管理 NSS 信号。当 RXNE 标志被置 1，它意味着对当前的传输结束了。即最后一位刚刚采样完毕，以及完整的数据帧存储在 RXFIFO 中。

24.4.8.3 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 SPIx_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 24-9 提供了组包方式顺序处理数据的一个例子。在一次对 SPIx_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，接受器必须对 SPI_DR 寄存器作一次 16 位的读访问才能够把数据全都取到。RxFIFO 的阈值和跟进的数据访问的位宽必须保持一致，否则就会丢数据。

字节数为奇数的数据帧传输时，会出现特定的问题。在发送端，可以用 8 位方式写 SPIx_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。

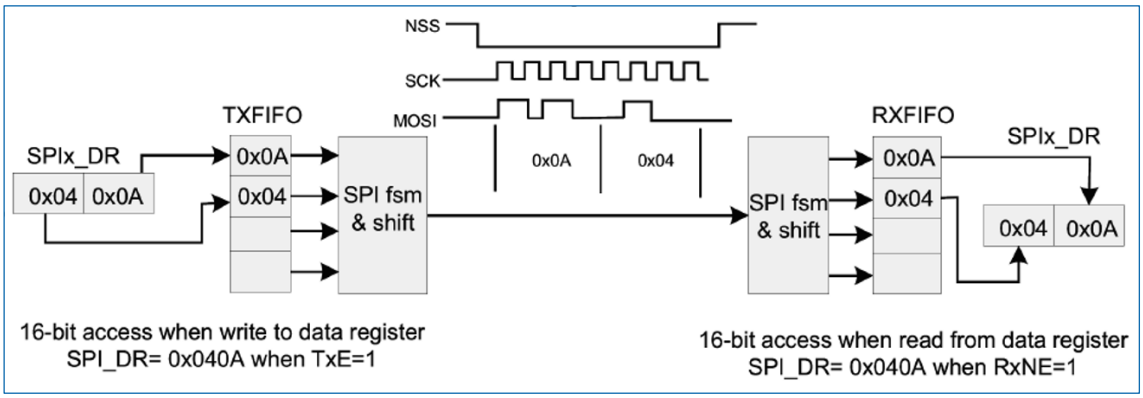


图 24-9 发送和接收 FIFO 中的数据组包

24.4.9 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。当外设时钟停止时，在系统进入低功耗模式前做到这一点是十分重要的。否则会损坏正在进行的交互。在某些模式下，禁止步骤是停止所进行的连续通信的唯一方式。

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。当一个帧事务正在进行或下一个数据帧存储在 TXFIFO 中，SPI 在主发射机被禁用时，SPI 行为是不保证的。

当主机处于任何接收模式时，停止连续时钟的唯一方法是通过 SPE=0 禁用外围设备。这个必须发生在特定的时间窗口内最后一个数据帧采样时间之间的交易，只是第一点和最后一点前转移开始（为了得到一个完整的预期数量数据帧，并防止任何额外的“假”数据后阅读最后一个有效的数据帧）。在该模式下禁用 SPI 必须遵循特定的步骤。

当 SPI 被禁用时，接收但没有读取的数据仍然存储在 RXFIFO 中，并且必须在下一次 SPI 被启用时进行处理，然后开始一个新的序列。为了防止有未读的数据，确保在禁用 SPI 时 RXFIFO 是空的，通过使用正确的禁用过程，或通过软件复位初始化所有 SPI 寄存器，通过控制一个专门用于外设复位的特定寄存器。

标准禁止步骤通过轮询 BSY 状态以及 TXE 标志来检查发送会话是否完全结束。还可以在必须识别正在处理的传输是否结束的特定情况下完成这种检查，例如：

- 当 NSS 信号由任意 GPIO 切换管理且主器件必须为从器件提供 NSS 脉冲结束时。
- 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 的传输数据流完成。

正确的禁止步骤如下（使用只接收模式时除外）：

1. 等待 RXNE=1，以接收最后的数据。
2. 等待 TXE=1，然后等待至 BSY=0，再关闭 SPI。
3. 读取接收的数据。

说明：在不连续通信期间，在对 SPIx_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。

24.4.10 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 SPIx_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 SPIx_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 SPIx_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 24-10 和图 24-11。

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 SPIx_CR2 寄存器中的 RXDMAEN 位来使能 DMA 接收缓冲区；如果使用数据流，通过 DMA 寄存器来使能 TX 和 RX 的 DMA 数据流；如果使用 DMA TX，通过

SPIx_CR2 寄存器中的 TXDMAEN 位来使能 DMA 发送缓冲区。

2. 通过将 SPE 位置 1 使能 SPI。
3. 要关闭通信，必须按顺序执行以下步骤：
4. 如果使能了 DMA，通过 DMA 寄存器来禁止 TX 和 RX 的 DMA 数据流。
5. 通过 SPE 置 0 来禁止 SPI。
6. 如果使用 DMA TX 和/或 DMA RX，通过将 SPIx_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。

24.4.10.1 DMA 与传输组包

如果传输由 DMA 来管理（TXDMAEN 和 RXDMAEN 在 SPIx_CR2 寄存器中设置），会根据 SPITX 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 SPIx_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“24.4.8.3 数据组包”。）

24.4.10.2 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 处理数据，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，以便在事务开始之前提前准备专用于主机的数据。

   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。

2. 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。
3. 只有 TXFIFO 为满时，TXE 位才会被置 0。
4. 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。
5. 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。这个标志总是在 SPI 事务完成之前置起。
6. 数据包的 CRC 值是在 SPIx_TXCRCR 和 SPIx_RXCRCR 寄存器中逐帧连续计算的。CRC 信息在整个数据包完成后进行处理，自动通过 DMA（必须将 TX 通道设置为要处理的数据帧数）或 SW（用户在最后一次处理数据帧时必须处理 CRCNEXT 位）。

   虽然在 SPIx_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 SPIx_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。

7. 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。

8. 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。

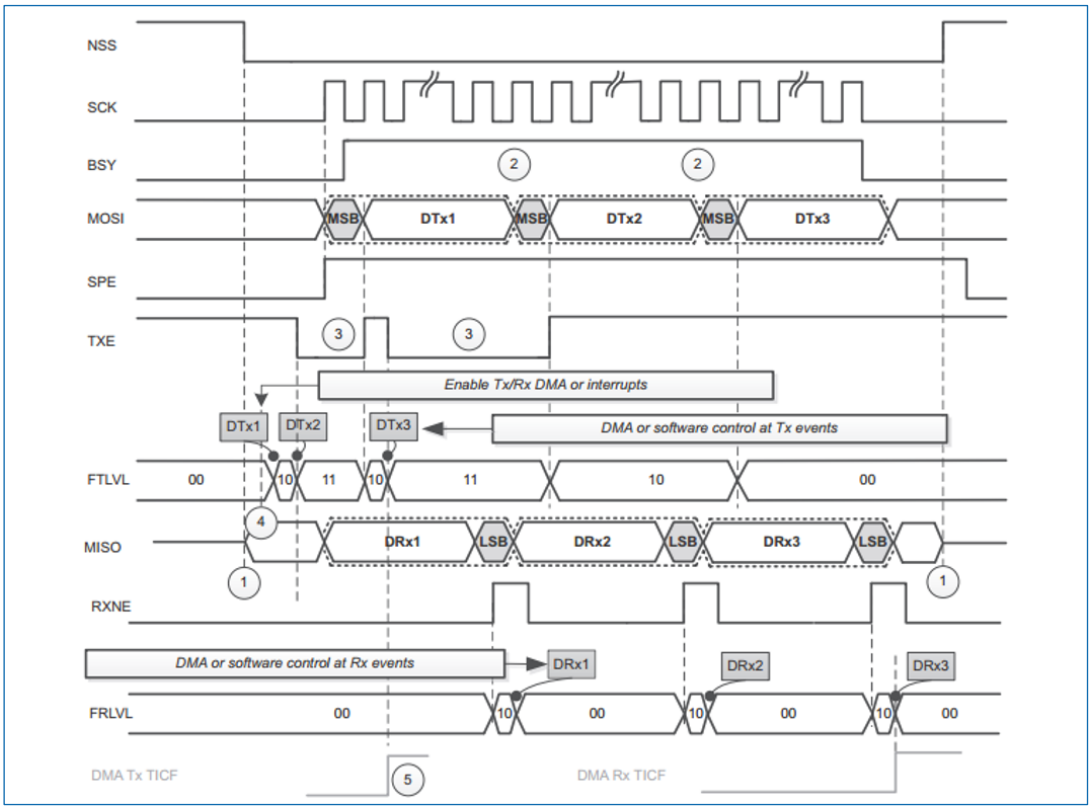


图 24-10 主模式全双工通信

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

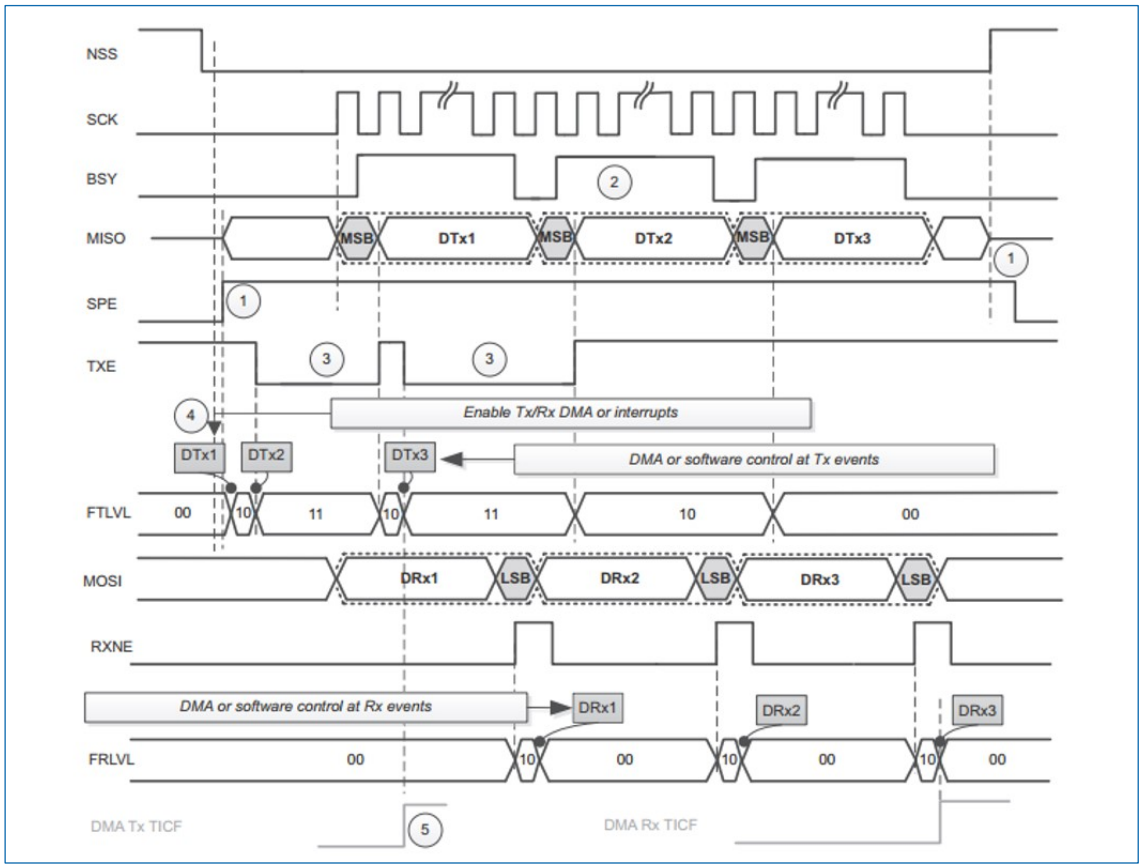


图 24-11 从模式全双工通信

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

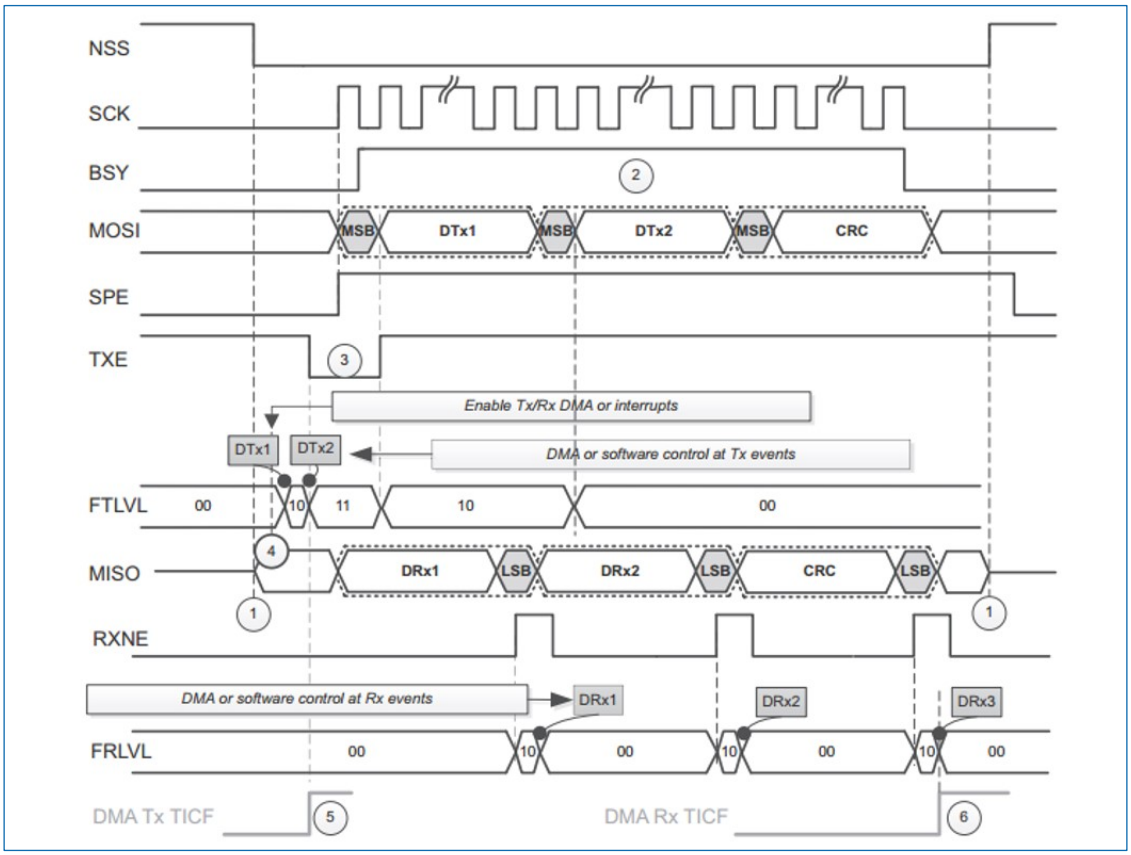


图 24-12 主机全双工通信带 CRC 校验

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

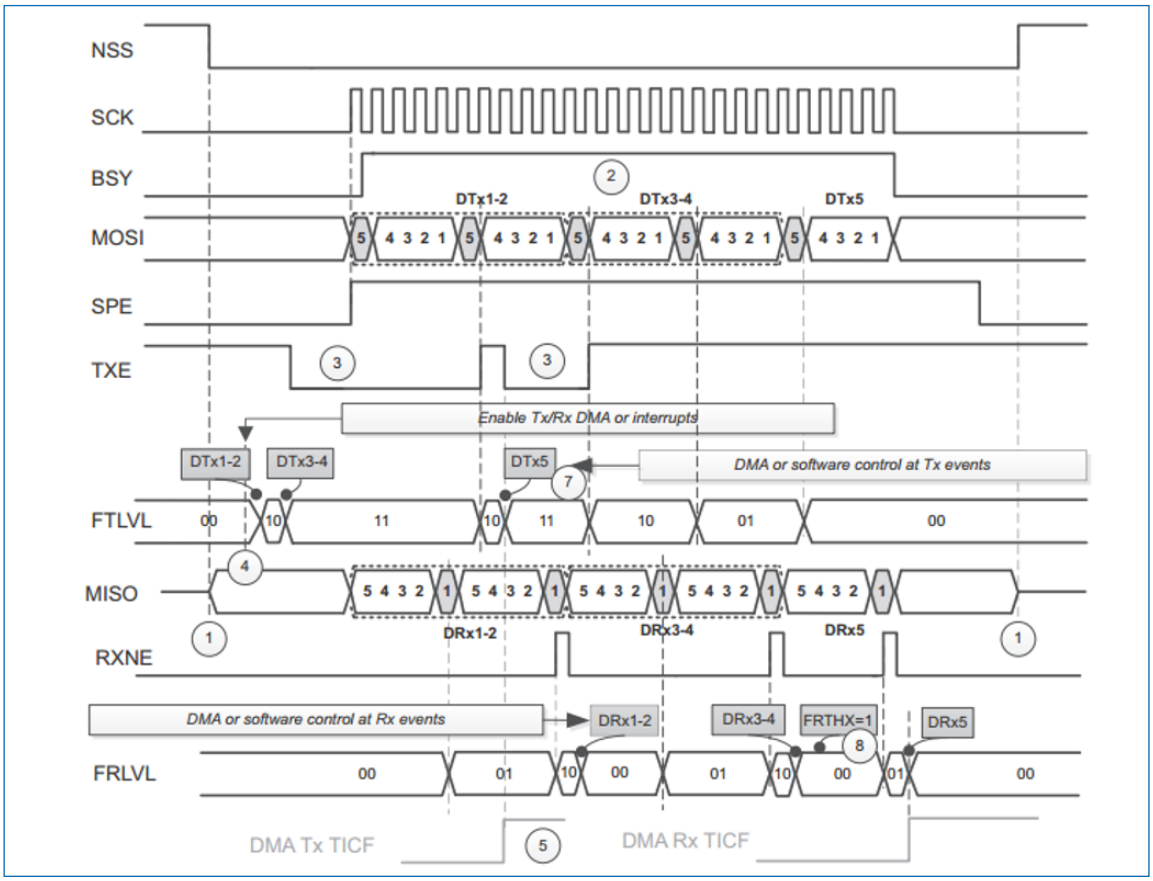


图 24-13 主模式全双工模式使用数据包传输

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

24.4.11 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。标志置 1 并保持直到 TXFIFO 深度是更低或等于 FIFO 深度的 1/2。如果 SPIx_CR2 寄存器中的 TXEIE 位置 1，当 TXE 置 1 时就会生成一个发送中断。当 TXFIFO 电平大于 1/2 时，位自动清除。

接收缓冲区非空（RXNE）

RXNE 标志的设置取决于 SPIx_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 SPIx_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。

24.4.12 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 SPIx_DR 寄存器的读访问和对 SPIx_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 SPIx_SR 寄存器执行读或写访问。
2. 对 SPIx_CR1 寄存器执行写操作。

为避免包含多个 MCU 的系统中发生多从模式冲突，必须在 MODF 位清零序列期间将 NSS 引脚拉高。在该清零序列后，可以将 SPE 和 MSTR 位恢复到原始状态。安全起见，硬件不允许在 MODF 位置 1 时将 SPE 和 MSTR 位置 1。在从器件中，MODF 位不可置 1，但由前一次多主模式冲突引起时除外。

CRC 错误（CRCERR）

当 SPIx_CR1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。当 CRCEN 位置 1，如果移位寄存器中接收的值与 SPIx_RXCRC 的值不匹配，SPIx_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，SPIx_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 SPIx_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。

24.4.13 NSS 脉冲模式

该模式由 SPIx_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（SPIx_CR1CPHA =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。

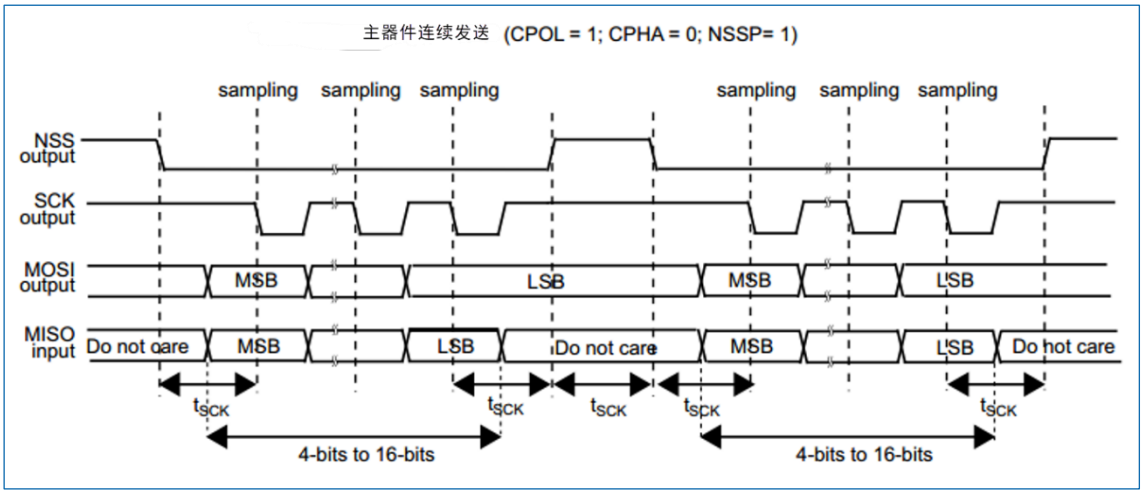


图 24-14 SPI Motorola 主模式 NSS 脉冲生成

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。

24.4.14 TI 模式

主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPIx_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 SPIx_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 SPIx_CR1 和 SPIx_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 24-15）。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 SPIx_CR1 寄存器的 BR[2:0]位设置的波特率值。具体公式如下：

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

如果从器件在数据帧传输期间检测到错位的 NSS 脉冲，TI FRE 标志将置 1。

如果数据大小为 4 位或 5 位，则在全双工模式或只发送模式下，主机使用的协议在 LSB 之后再增加

一个虚拟数据位。TI NSS 脉冲是在这个虚拟位时钟周期以上产生的，而不是在每个周期的 LSB。

此特性不适用于 Motorola SPI 通信（FRF 位为 0）。

图 24-15 给出了选择 TI 模式时的 SPI 通信波形。

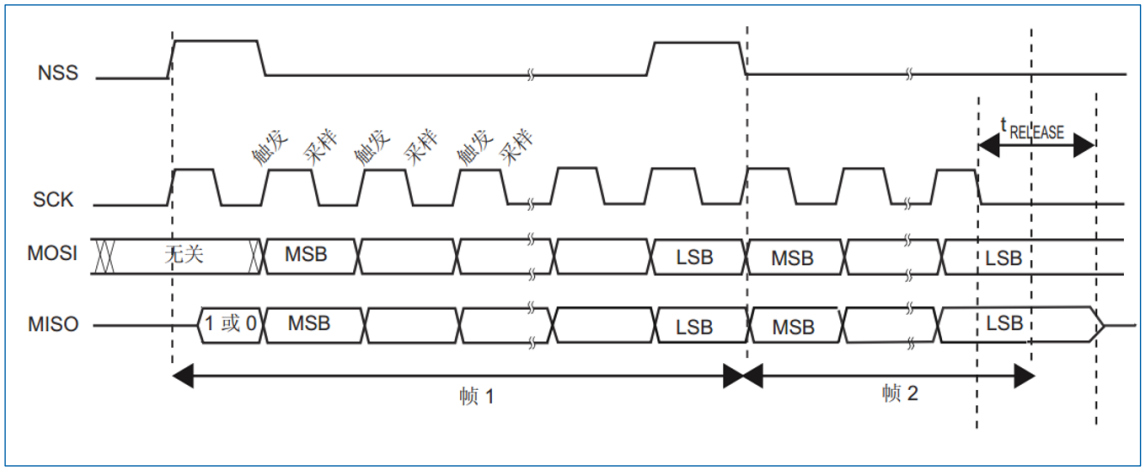


图 24-15 TI 模式传输

24.4.15 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。

24.4.15.1 CRC 原理

在使能 SPI（SPE=1）前，通过将 SPIx_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 SPIx_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。

24.4.15.2 CPU 管理的 CRC 传输

通信开始后将一直持续到发送或接收 SPIx_DR 寄存器中的最后一个数据帧时。之后，SPIx_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 SPIx_RXCRC 寄存器中的值进行比较。软件必须校验 SPIx_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 SPIx_DR 寄存器中进行读取，以将 RXNE 标志清零。

24.4.15.3 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 SPIx_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 SPIx_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。

24.4.15.4 复位 SPIx_TXCRC 和 SPIx_RXCRC 值

当使能 CRC 计算时，SPIx_TXCRC 和 SPIx_RXCRC 值自动清零。这就允许使用 DMA 循环模式（只接收模式除外）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。
2. 将 CRCEN 位清零。
3. 使能 CRCEN 位。
4. 使能 SPI。

说明：

- *当 SPI 处于从模式时，CRC 计算单元对 SCK 从输入时钟是敏感的，只要 CRCEN 位被设置，这是无论 SPE 位的值是什么情况。为了避免错误的 CRC 计算，软件必须在时钟稳定(处于稳定状态)时才启用 CRC 计算。*
- *当 SPI 接口被配置为从模式时，一旦 CRCNEXT 信号被释放，在 CRC 阶段的事务期间，NSS 内部信号需要保持低电平。这就是为什么 CRC 计算不能在 NSS 脉冲模式下使用，而 NSS 硬件模式应该在从机正常应用。*

24.5 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 24-2 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	

24.6 I2S 功能说明

24.6.1 I2S 一般说明

I2S 的框图如下图所示。

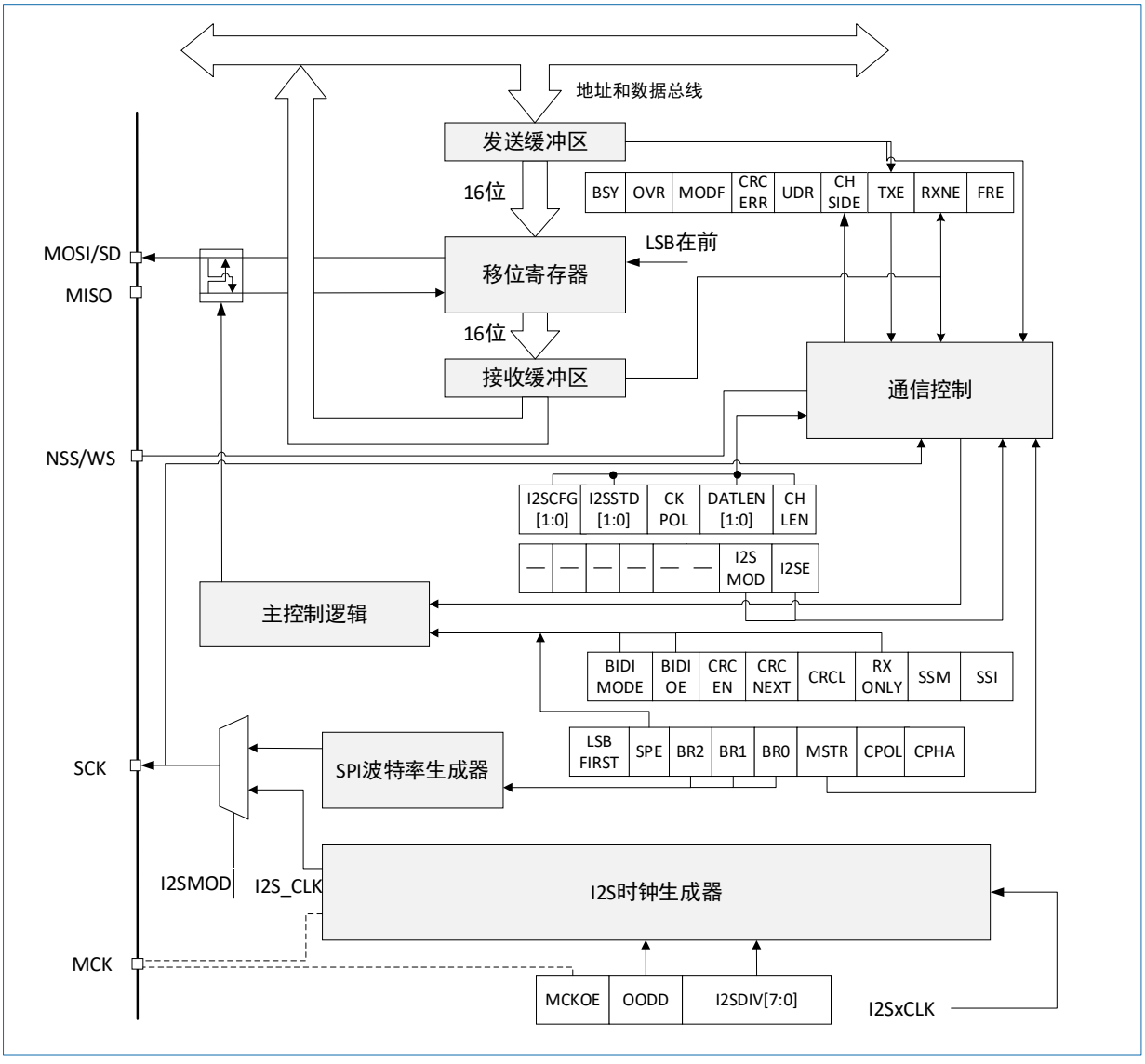


图 24-16 I2S 框图

(1). MCK 映射到 MISO 引脚上。

当使能 I2S 功能（将 SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1）后，SPI 可用作音频 I2S 接口。此接口使用几乎与 SPI 相同的引脚、标志和中断。

I2S 与 SPI 共用以下三个引脚：

- SD：串行数据（映射到 MOSI 引脚），用于发送或接收两个时分复用的数据通道上的数据（仅半双工模式）。
- WS：字选择（映射到 NSS 引脚），是主模式下的数据控制信号输出以及从模式下的数据控制信号输入。
- CK：串行时钟（映射到 SCK 引脚），是主模式下的串行时钟输出以及从模式下的串行时钟输入。

当某些外部音频设备需要使用主时钟输出时，可以使用其它引脚：

- MCK：当 I2S 配置为主模式（并且 SPIx_I2SPR 寄存器中的 MCKOE 位置 1）时，使用主时钟（单独映射）输出此附加时钟，该时钟以 256 ×fs 的预配置频率生成，其中 fs 为音频信号采样频率。

I2S 在主模式下使用自身的时钟发生器生成通信时钟。此时钟发生器也是主时钟输出的源。

在 I2S 模式下可以使用两个额外的寄存器。一个是时钟发生器配置寄存器 SPIx_I2SPR，另一个是通用 I2S 配置寄存器 SPIx_I2SCFGR（音频标准、从/主模式、数据格式、数据包帧、时钟极性等）。

在 I2S 模式下不使用 SPIx_CR1 寄存器和所有 CRC 寄存器。同样，也不使用 SPIx_CR2 寄存器中的 SSOE 位以及 SPIx_SR 中的 MODF 和 CRCERR 位。

I2S 使用相同的 SPI 数据寄存器（SPIx_DR）进行 16 位数据传输。

24.6.2 I2S 全双工

图 24-17 显示了如何使用两个 SPI/I2S 接口进行全双工通信。此时，两个 SPI/I2S 的 WS 和 CK 引脚必须连接在一起。

对于主机全双工模式，一个 SPI/I2S 块必须在主模式（I2SCFG='10'或'11'）中编程，另一个 SPI/I2S 块必须在从（I2SCFG='00'或'01'）中编程。根据应用程序的需要，可以决定是否生成 MCK。

对于从机全双工模式，两个 SPI/I2S 块必须在从模式编程。其中一个在从接收模式（I2SCFG='01'），另一个在从发送模式（I2SCFG='00'）。由外部主设备提供位时钟（CK）和帧同步（WS）。

注意：全双工模式可用于所有支持的标准，包括 I2S Philips、mbs-justified、LSB-justified 和 PCM。

对于全双工模式，两个 SPI/I2S 接口必须使用相同的标准，具有相同的参数：I2SMOD、I2SSTD、CKPOL、PCMSYNC、DATLEN 和 CHLEN 必须在两个实例上包含相同的值。

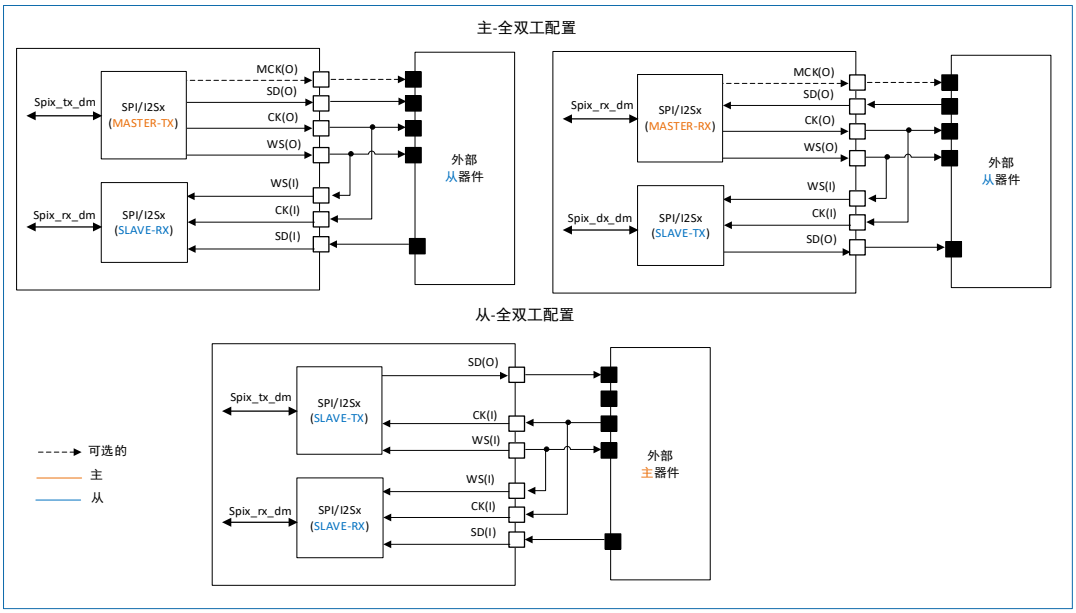


图 24-17 全双工通信

24.6.3 支持的音频协议

三线总线仅需要处理在左右两个通道上时分复用的音频数据。但是，只有一个 16 位寄存器进行发送和接收。所以，需由软件将与每个通道对应的值写入数据寄存器，以及从数据寄存器中读取数据，并通过检查 SPIx_SR 寄存器中的 CHSIDE 位来识别对应的通道。始终先发送左通道数据，而后再发送右通道数据（对于 PCM 协议来说，CHSIDE 没有意义）。

数据和帧格式组合有四种，可采用下列格式发送数据：

- 将 16 位数据封装在 16 位帧中。
- 将 16 位数据封装在 32 位帧中。
- 将 24 位数据封装在 32 位帧中。
- 将 32 位数据封装在 32 位帧中。

当使用 32 位数据包中的 16 位数据时，前 16 位（MSB）为有效位，后 16 位 LSB 被强制清零，无需任何软件操作或 DMA 请求（只需一个读/写操作）。

如果应用程序首选 DMA，则 24 位和 32 位数据帧需要对 SPIx_DR 寄存器执行两次 CPU 读取或写入操作，或者需要两次 DMA 操作。对于 24 位数据帧，硬件会在低 8 位填充 8 个 0 扩展到 32 位。

对于所有数据格式和通信标准而言，始终会先发送最高有效位（MSB 在前）。

I2S 接口支持四种音频标准，可使用 SPIx_I2SCFGR 寄存器中的 I2SSTD[1:0]和 PCMSYNC 位对其进行配置。

24.6.3.1 I2S Philips 标准

使用 WS 信号来指示当前正在发送的数据所属的通道。该信号从当前通道数据的第一个位（MSB）之前的一个时钟开始有效。

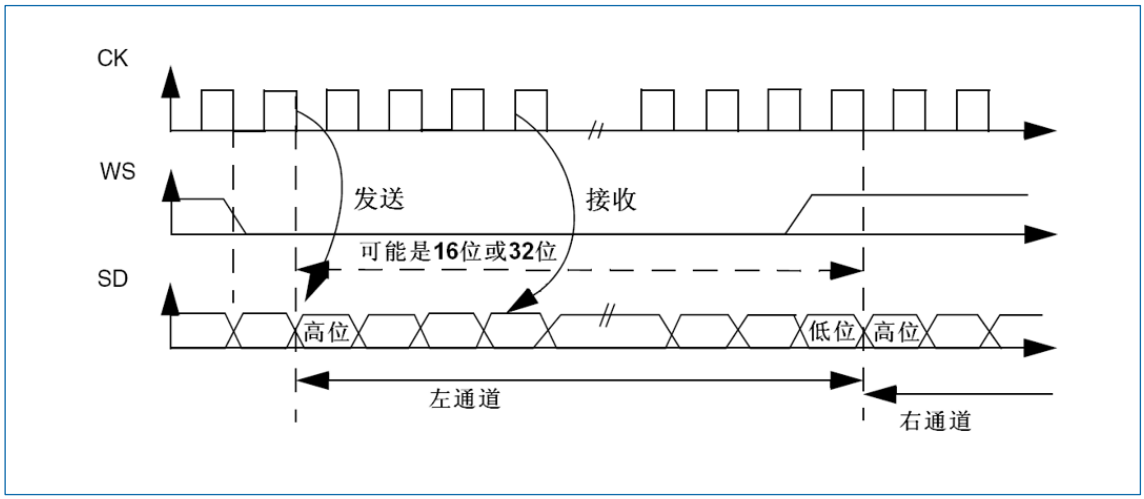


图 24-18 I2S Philips 协议波形（16/32 位全精度，CPOL=0）

发送方在时钟信号（CK）的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

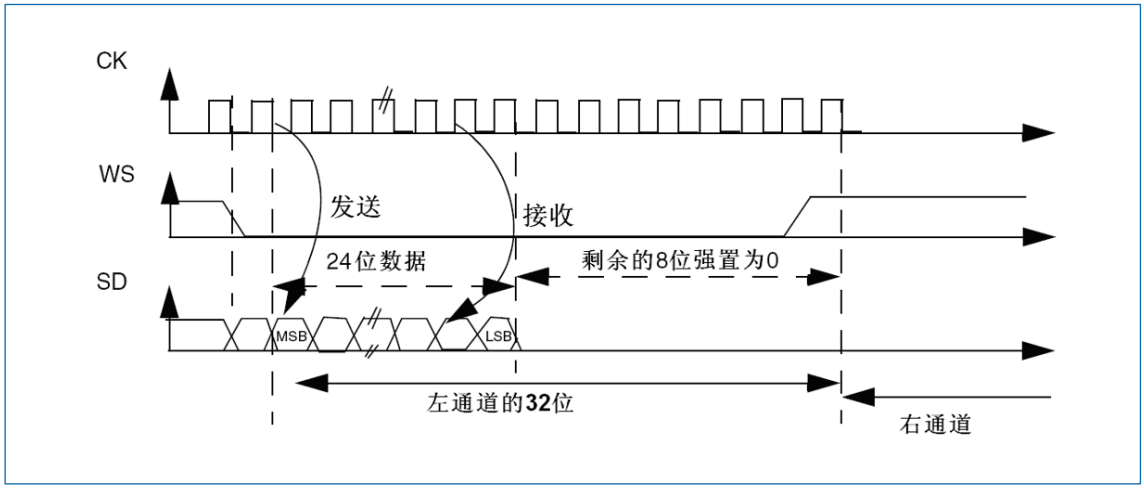


图 24-19 I2S Philips 标准波形（24 位帧，CPOL=0）

该模式需要对 SPIx_DR 寄存器执行两次写入或读取操作。

- 在发送模式下

如果需要发送 0x8EAA33（24 位）：

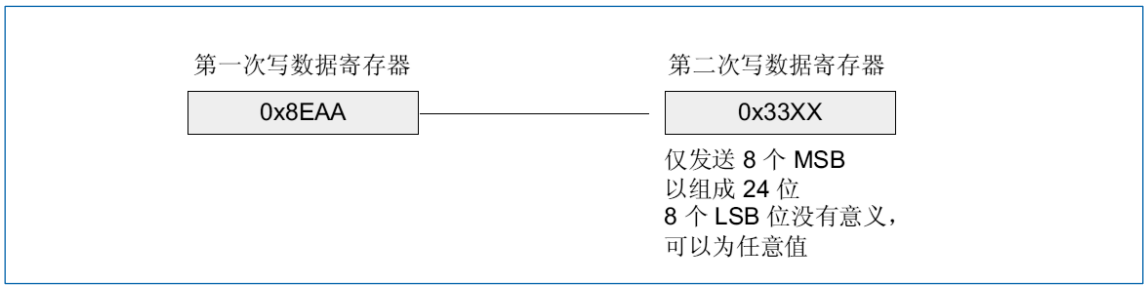


图 24-20 发送 0x8EAA33

- 在接收模式下

如果接收数据 0x8EAA33：

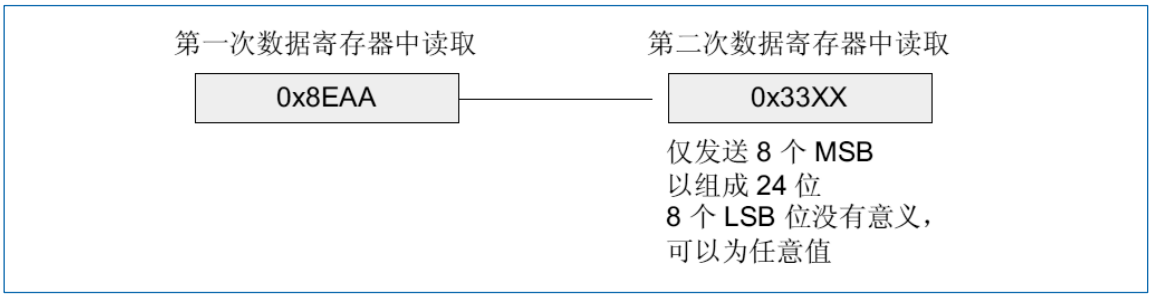


图 24-21 接收 0x8EAA33

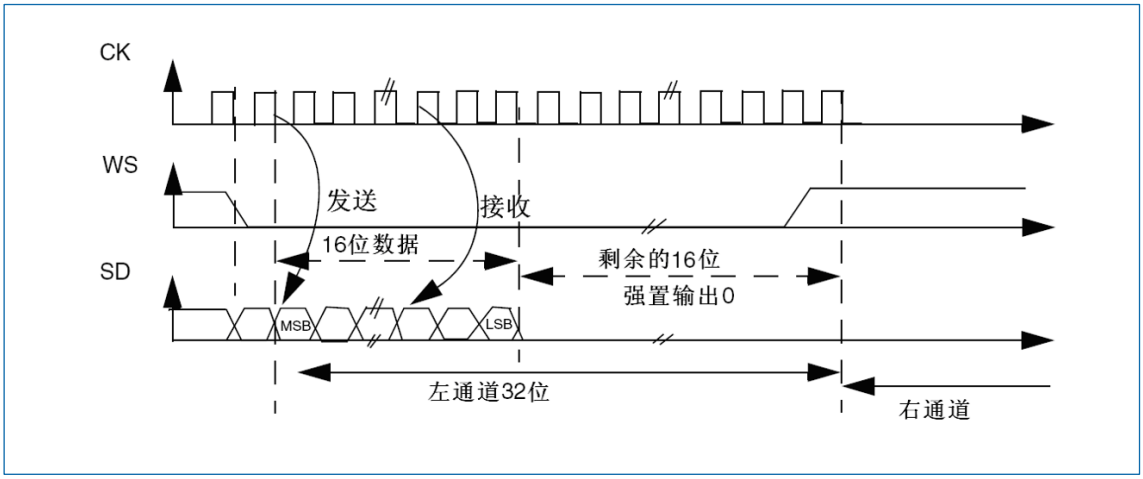


图 24-22 I2S Philips 标准（16 位扩展为 32 位数据包帧，CPOL=0）

如果在 I2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 SPIx_DR 寄存器。扩展到 32 位中高半字（16 位 MSB）被硬件置为 0x0000。

如果要发送的数据或已接收的数据为 0x76A3（0x76A30000 扩展为 32 位），则需要执行图 24-23 中显示的操作。

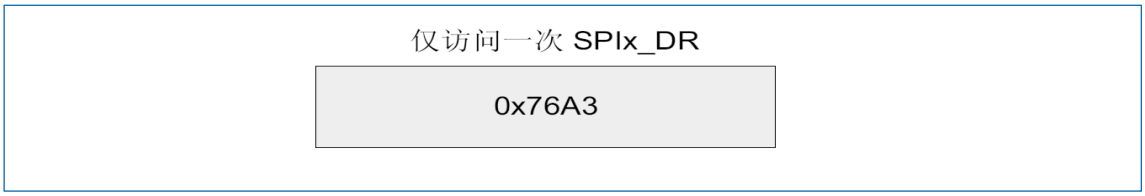


图 24-23 16 位数据帧扩展到 32 位通道帧的示例

发送时，每次将 MSB 写入 SPIx_DR，TXE 标志就会置 1，并在中断使能的情况下触发中断，以将要发送的新数据加载到 SPIx_DR 寄存器。即使硬件填充的低 16 位 0x0000 还未发送，也会如此，因为低 16 位是由硬件发送。

接收时，接收到第一个半字（高 16 位），则硬件将 RXNE 标志置 1，并在中断使能的情况下触发中断。

这样，就延长了两个写入或读取操作之间的时间间隔，从而可防止出现下溢或上溢情况（具体取决于数据传输方向）。

24.6.3.2 MSB 对齐标准

此标准同时生成 WS 信号和第一个数据位（即 MSB）。

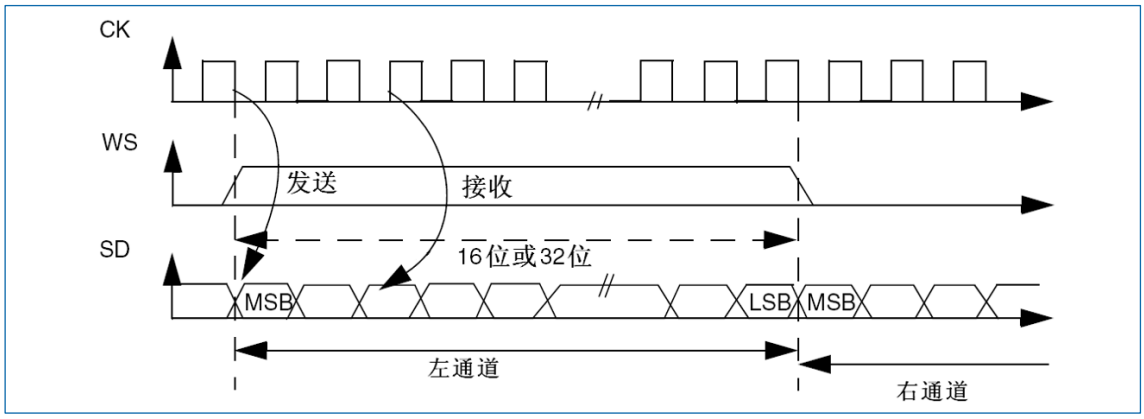


图 24-24 MSB 对齐的 16 位或 32 位全精度长度，CPOL=0

发送方在时钟信号的下降沿改变数据；接收方在上升沿读取数据。

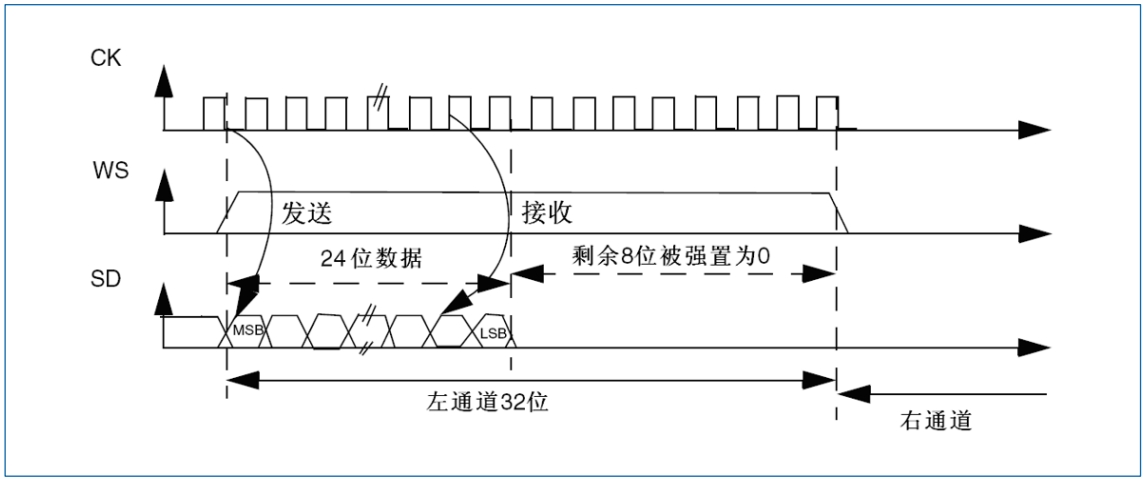


图 24-25 MSB 对齐的 24 位帧长度，CPOL=0

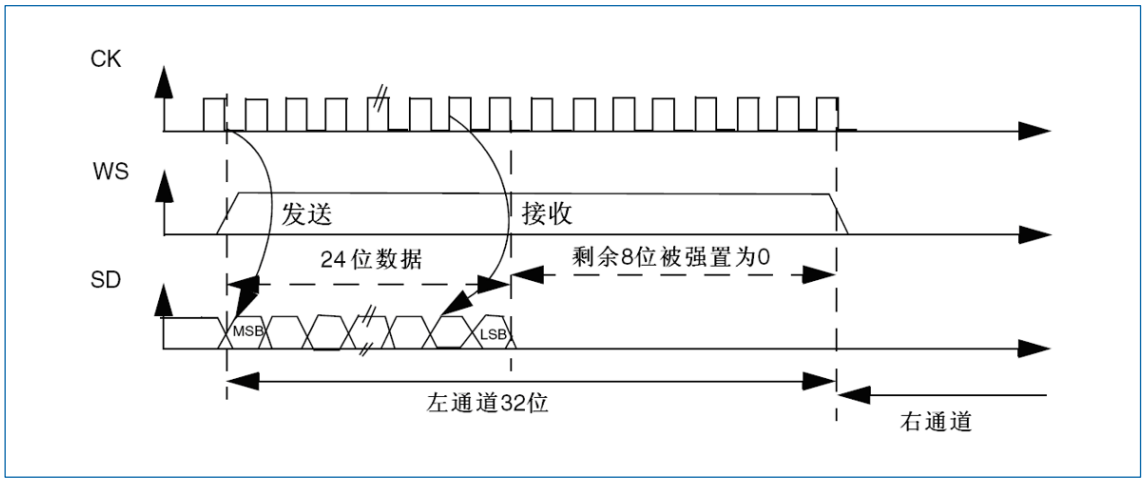


图 24-26 MSB 对齐 16 位数据扩展到 32 位数据包帧，CPOL=0

24.6.3.3 LSB 对齐标准

该标准与 MSB 对齐标准类似（对于 16 位和 32 位全精度帧格式，没有任何不同）。

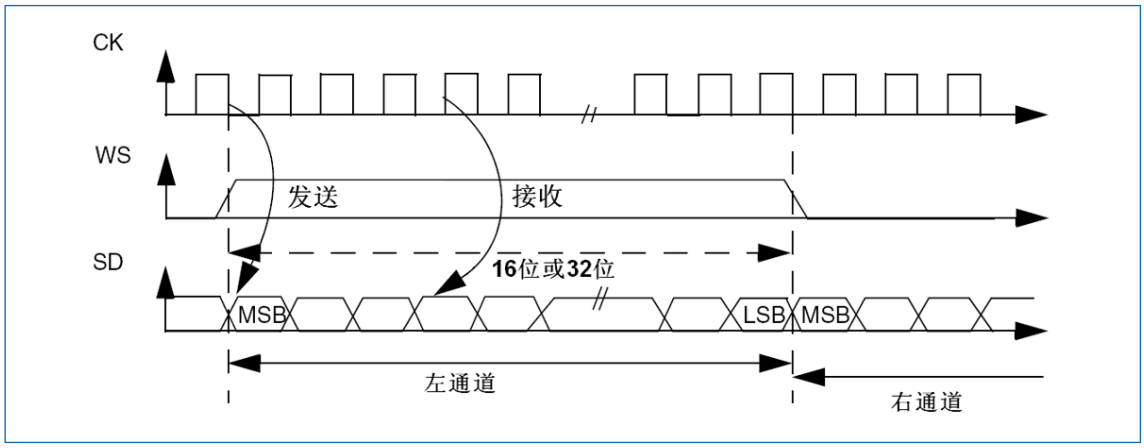


图 24-27 LSB 对齐的 16 位或 32 位全精度，CPOL=0

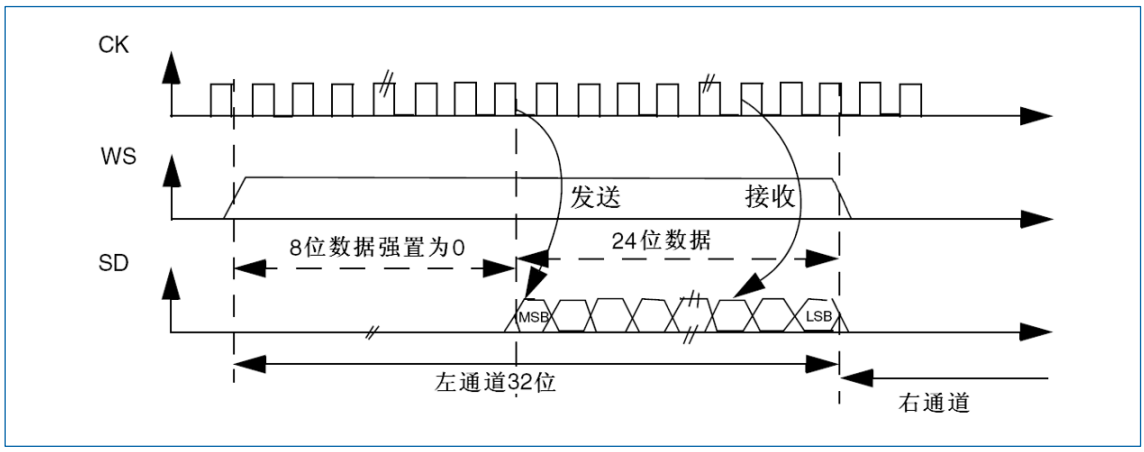


图 24-28 LSB 对齐的 24 位帧长度，CPOL=0

- 在发送模式下

如果需要发送数据 0x3478AE，则需要通过软件或 DMA 对 SPIx_DR 寄存器执行两次写入操作。下面给出了这些操作。

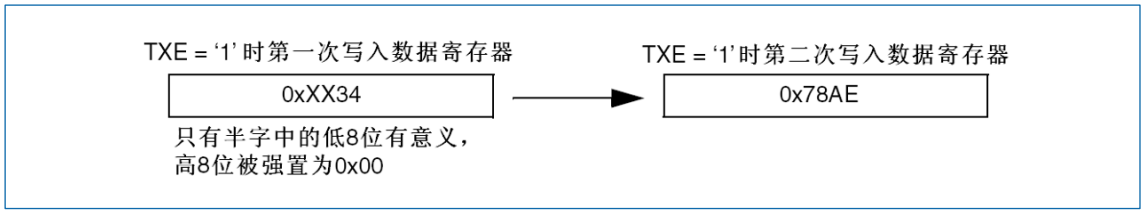


图 24-29 发送 0x3478AE 所需的操作

- 在接收模式下

如果接收到数据 0x3478AE，则在每个 RXNE 事件时需要对 SPIx_DR 寄存器执行两次连续的读取操作。

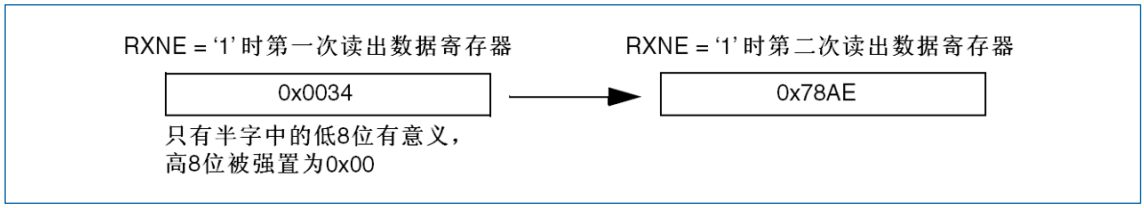


图 24-30 接收 0x3478AE 时所需的操作

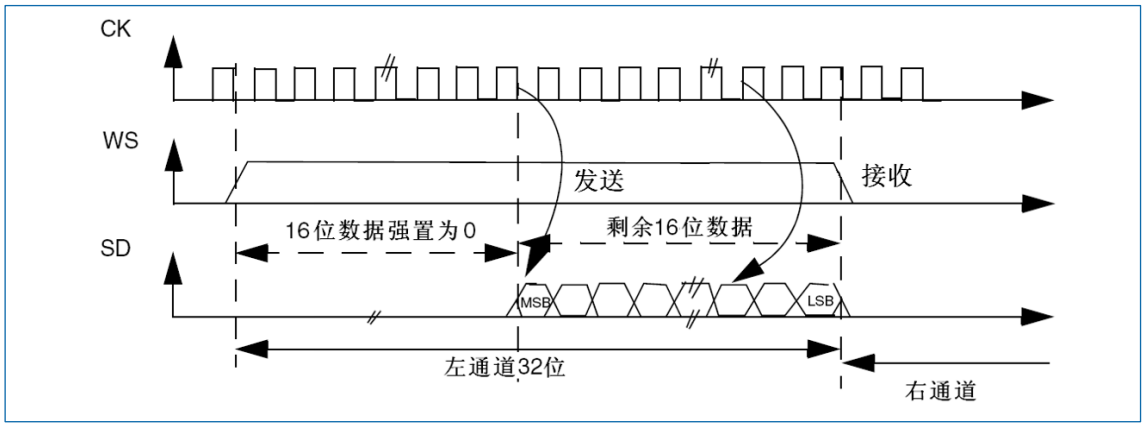


图 24-31 扩展为 32 位数据包帧的 LSB 对齐的 16 位，CPOL=0

如果在 I2S 配置阶段选择将 16 位数据帧扩展到 32 位通道帧，则只需要访问一次 SPIx_DR 寄存器。扩展到 32 位中高半字（16 位 MSB）被硬件置为 0x0000。在这种情况下，其对应于半字 MSB。

如果要发送的数据或已接收的数据为 0x76A3（0x000076A3 扩展为 32 位），则需要执行图 24-32 中显示的操作。

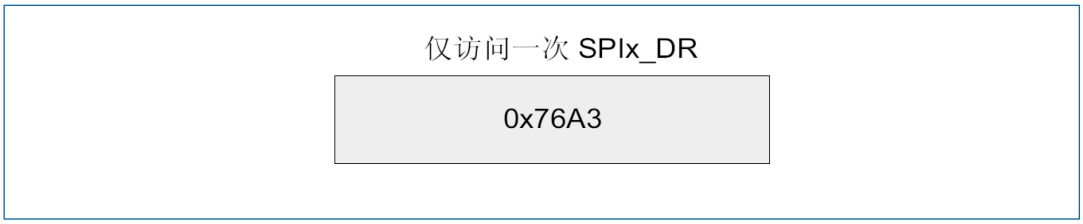


图 24-32 16 位数据帧扩展到 32 位通道帧的示例

发送时，每次将 MSB 写入 SPIx_DR，TXE 标志就会置 1，并在中断使能的情况下触发中断，以将要发送的新数据加载到 SPIx_DR 寄存器。即使硬件填充的低 16 位 0x0000 还未发送，也会如此，因为低 16 位是由硬件发送。

接收时，接收到第一个半字（高 16 位），则硬件将 RXNE 标志置 1，并在中断使能的情况下触发中断。

这样，就延长了两个写入或读取操作之间的时间间隔，以防止出现下溢或上溢情况（具体取决于数据传输方向）。

24.6.3.4 PCM 标准

对于 PCM 标准，无需使用通道信息。可使用两种 PCM 模式（短帧和长帧），并且可使用 SPIx_I2SCFGR 寄存器中的 PCMSYNC 位来配置。

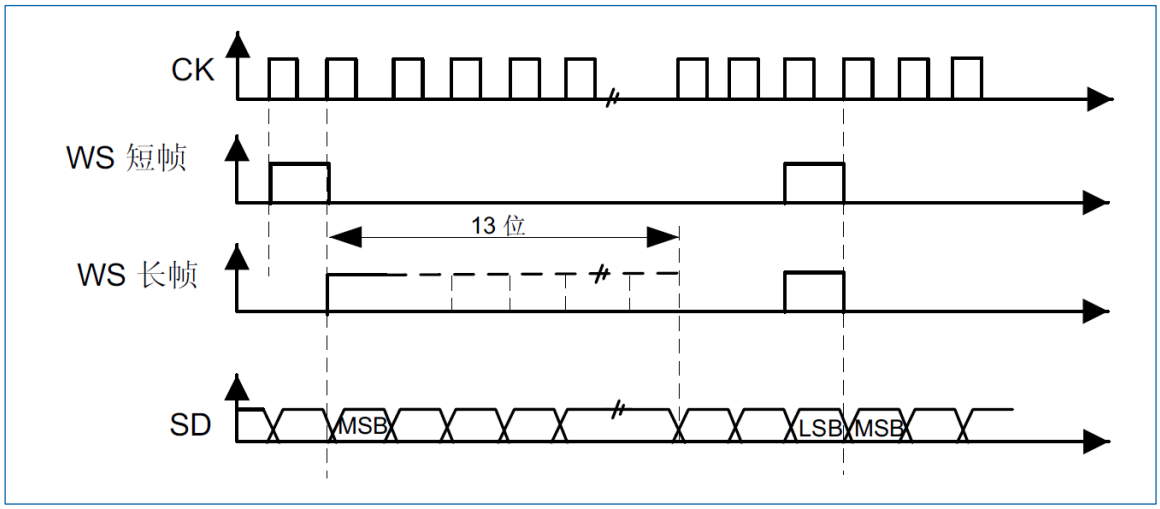


图 24-33 PCM 标准波形（16 位）

对于长帧同步，在主模式下会将 WS 信号持续 13 个周期。

对于短帧同步，WS 同步信号的持续时间仅为一个周期。

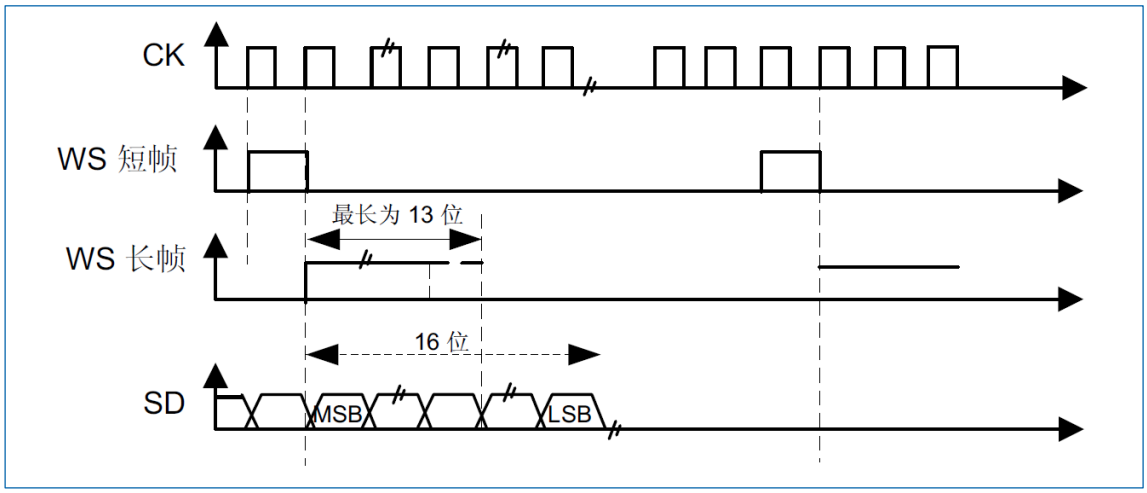


图 24-34 PCM 标准波形（16 位扩展到 32 位数据包帧）

说明：对于两种模式（主/从模式）和两种同步（短/长同步），即使在从模式下，也需要指定两组连续数据（以及两个同步信号）之间位的个数（SPIx_I2SCFGR 寄存器中的 DATLEN 位和 CHLEN 位）。

24.6.4 启动描述

图 24-35 显示了在 MASTER 模式下如何处理串行接口，当 I2S 被启用时(I2SE=1)。同时也显示了 CKPOL 对生成信号的影响。

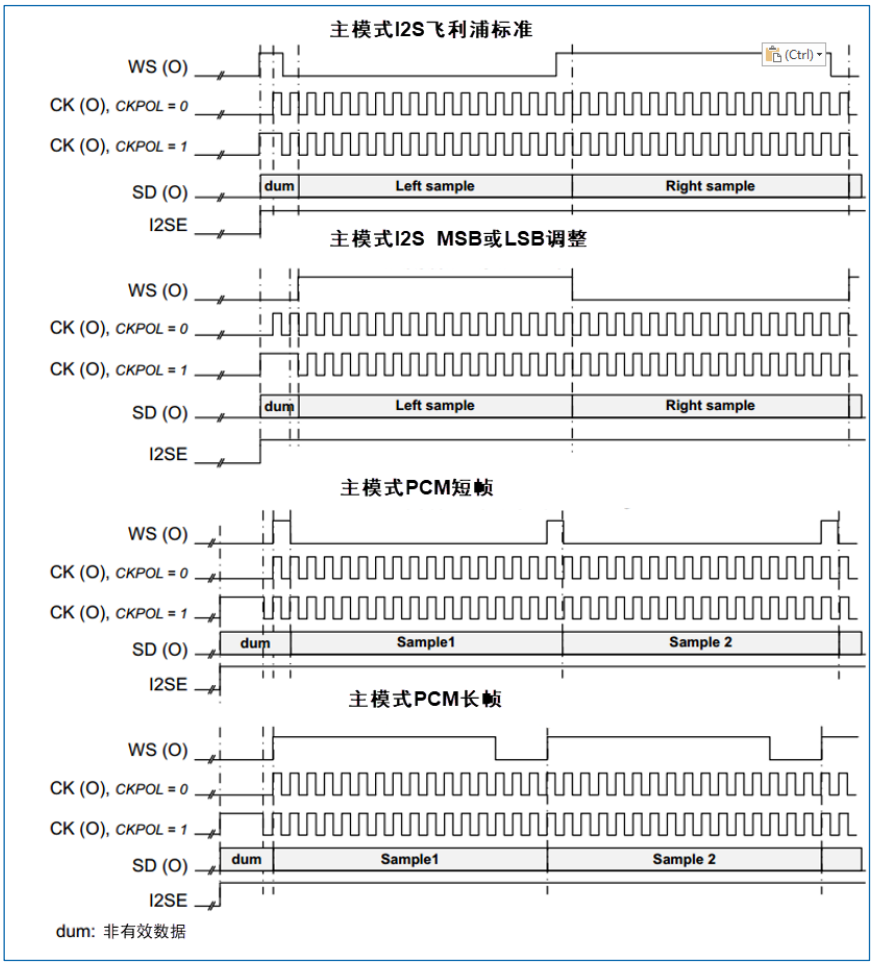


图 24-35 主模式下启动序列

在从模式下，用户必须在 WS 变得活跃之前启用音频接口。这意味着当飞利浦标准 I2S 的 WS = 1 时，I2SE 位必须设置为 1，或者当其他标准的 WS = 0。

24.6.5 时钟发生器

I2S 比特率用来确定 I2S 数据线上的数据流和 I2S 时钟信号频率。

I2S 比特率=每个通道的位数×通道数×音频采样频率

对于 16 位双通道音频，I2S 比特率的计算公式如下：

I2S 比特率=16 × 2 ×fs

如果数据包为 32 位宽，则 I2S 比特率=32x 2x fs。

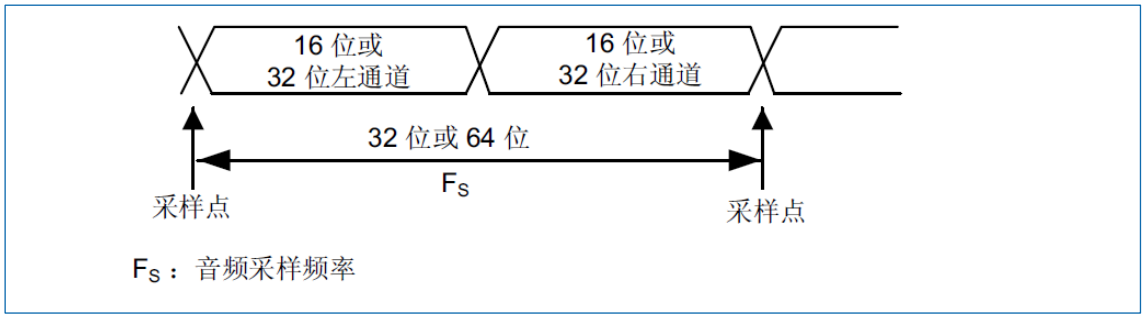


图 24-36 音频采样频率定义

配置主模式时，需要正确地对线性分频器进行设置，以便采用所需的音频频率进行通信。

图 24-37 显示了通信时钟架构。I2Sx 时钟始终为系统时钟。

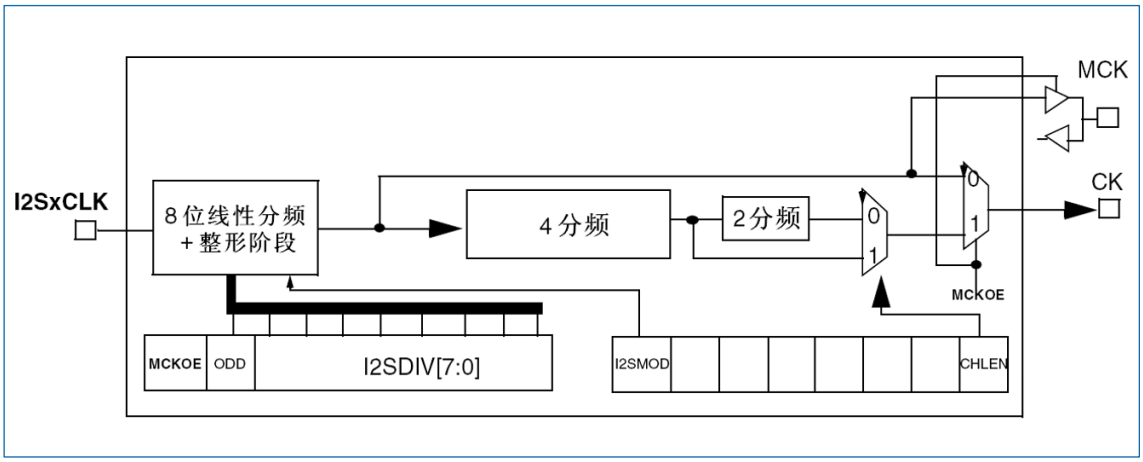


图 24-37 I2S 时钟发生器架构

图中说明：图中 x=2。

音频采样频率可以是 192KHz、96kHz、48kHz、44.1kHz、32kHz、22.05kHz、16kHz、11.025kHz 或 8kHz（或此范围内的任何其它值）。为达到所需频率，需要根据以下公式对线性分频器进行编程：

- 输出主时钟（SPIx_I2SPR 寄存器中的 MCKOE 置 1）时：

fs=I2SxCLK/[（16*2）*（（2*I2SDIV）+ODD）*8）]（通道帧宽度为 16 位时）

fs=I2SxCLK/[（32*2）*（（2*I2SDIV）+ODD）*4）]（通道帧宽度为 32 位时）

- 禁止主时钟输出（MCKOE 位清零）时：

fs=I2SxCLK/[（16*2）*（（2*I2SDIV）+ODD）]（通道帧宽度为 16 位时）

fs=I2SxCLK/[（32*2）*（（2*I2SDIV）+ODD）]（通道帧宽度为 32 位时）

表 24-3 提供了针对不同时钟配置的示例精度值。

说明：还可以采用其它配置以达到更好的时钟精度。

表 24-3 使用标准 8MHzHSE 时的音频频率精度

SYSCLK(MHz)	数据长度	I2SDIV	I2SODD	MCLK	目标 fs(Hz)	实际 fs(Hz)	误差
32	16	5	0	无	96000	100	4.1667%
32	32	2	0	无	96000	100	4.1667%
32	16	10	1	无	48000	47.619	0.7937%
32	32	5	0	无	48000	50	4.1667%
32	16	11	1	无	44100	43.478	1.4098%
32	32	5	1	无	44100	45.454	3.0715%
32	16	15	1	无	32000	32.258	0.8065%
32	32	8	0	无	32000	31.25	2.3430%
32	16	22	1	无	22050	22.222	0.7811%
32	32	11	1	无	22050	21.739	1.4098%
32	16	31	1	无	16000	15.873	0.7937%
32	32	15	1	无	16000	16.129	0.8065%

SYSCLK(MHz)	数据长度	I2SDIV	I2SODD	MCLK	目标 fs(Hz)	实际 fs(Hz)	误差
32	16	45	1	无	11025	10.989	0.3264%
32	32	22	1	无	11025	11.111	0.7811%
32	16	62	1	无	8000	8	0.0000%
32	32	31	1	无	8000	7.936	0.7937%
32	16	2	0	有	32000	31.25	2.3430%
32	32	2	0	有	32000	31.25	2.3430%
32	16	3	0	有	22050	20.833	5.5170%
32	32	3	0	有	22050	20.833	5.5170%
32	16	4	0	有	16000	15.625	2.3428%
32	32	4	0	有	16000	15.625	2.3428%
32	16	5	1	有	11025	11.363	3.0715%
32	32	5	1	有	11025	11.363	3.0715%
32	16	8	0	有	8000	7.812	2.3428%
32	32	8	0	有	8000	7.812	2.3428%

24.6.6 I2S 主模式

I2S 可配置为主模式。这意味着将在 CK 引脚输出串行时钟，在 WS 引脚生成字选信号。主时钟（MCK）可以输出，也可以不输出，具体由 SPIx_I2SPR 寄存器中的 MCKOE 位控制。

主模式配置步骤

1. 设置 SPIx_I2SPR 寄存器的 I2SDIV[7:0]位，以定义串行时钟波特率，从而达到相应的音频采样频率。SPIx_I2SPR 寄存器的 ODD 位也需要设置。

2. 设置 CKPOL 位，定义时钟在空闲时的电平状态。如果需要为外部 ADC 音频组件提供主时钟 MCK，则将 SPIx_I2SPR 寄存器的 MCKOE 位置 1（I2SDIV 和 ODD 值应根据 MCK 输出的状态进行计算。有关详细信息，请参见“24.6.5 时钟发生器”。

3. 将 SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1 以激活 I2S 功能，通过 I2SSTD[1:0]和 PCMSYNC 位选择 I2S 标准，通过 DATLEN[1:0]位选择数据长度并通过配置 CHLEN 位选择每个通道的位数。此外，通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0]位选择 I2S 主模式和方向（发送器或接收器）。

4. 如果需要，通过对 SPIx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。

5. SPIx_I2SCFGR 寄存器的 I2SE 位必须置 1。

WS 和 CK 配置为输出模式。如果 SPIx_I2SPR 的 MCKOE 位置 1，则 MCK 也是输出。

发送序列

将半字写入发送缓冲区后，发送序列随即开始。

假设写入发送缓冲区的第一个数据对应于左通道数据。数据从发送缓冲区传输到移位寄存器时，TXE 置 1，并且必须将对应于右通道的数据写入发送缓冲区。CHSIDE 标志指示将发送的数据对应的通道。TXE

标志置 1 时，CHSIDE 标志有意义，因为该标志在 TXE 变为高电平时进行更新。

一个完整的帧将先进行左通道数据发送，再进行右通道数据发送。不存在仅发送左通道的部分帧。

首位发送期间，数据按半字并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚（MSB 在前）。每次数据从发送缓冲区传输到移位寄存器后，TXE 标志都将置 1，如果 SPIx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

有关各种 I2S 标准模式的写操作的更多详细信息，请参见“24.6.3 支持的音频协议”。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送的数据写入 SPIx_DR 寄存器。

要通过将 I2SE 清零来关闭 I2S，必须等待 TXE=1 且 BSY=0。

接收序列

此工作模式与发送模式相同，只有第 3 点存在不同（请参见“24.6.6 I2S 主模式”所述的步骤），即通过 I2SCFG[1:0]位设置主器件接收模式。

无论数据或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收缓冲区满时，RXNE 标志即置 1，并且如果 SPIx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可通过一次或两次接收操作进入接收缓冲区，具体取决于数据和通道长度配置。

读取 SPIx_DR 寄存器即会使 RXNE 位清零。

CHSIDE 在每次接收后进行更新。它由 I2S 单元所产生的 WS 信号触发。

有关各种 I2S 标准模式中读操作的更多详细信息，请参见“24.6.3 支持的音频协议”。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误并将 OVR 标志置 1。

如果 SPIx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要关闭 I2S，需要执行特定操作来确保 I2S 正确完成传输周期而不启动新的数据传输。该序列取决于数据和通道长度的配置，以及所选的音频协议模式。在以下情况下：

- 32 位通道长度上扩展的 16 位数据长度（DATLEN=00 且 CHLEN=1），使用 LSB 对齐模式（I2SSTD=10）。
  A. 等待倒数第二个 RXNE=1（n-1）
  B. 然后等待 17 个 I2S 时钟周期（使用软件循环）
  C. 禁止 I2S（I2SE=0）
- 32 位通道长度上扩展的 16 位数据长度（DATLEN=00 且 CHLEN=1），使用 MSB 对齐、I2S 或 PCM 模式（分别为 I2SSTD=00、I2SSTD=01 或 I2SSTD=11）。
  A. 等待最后一个 RXNE
  B. 然后等待 1 个 I2S 时钟周期（使用软件循环）
  C. 禁止 I2S（I2SE=0）
- 对于 DATLEN 和 CHLEN 的所有其它组合，无论通过 I2SSTD 位选择何种音频模式，都将执行以下序列来关闭 I2S：
  A. 等待倒数第二个 RXNE=1（n-1）
  B. 然后等待一个 I2S 时钟周期（使用软件循环）
  C. 禁止 I2S（I2SE=0）

说明：传输期间，BSY 标志保持低电平。

24.6.7 I2S 从模式

对于从模式而言，I2S 可配置为发送模式或接收模式。

此工作模式所遵循的规则与 I2S 主模式配置基本相同。在从模式下，I2S 接口不产生时钟。

时钟和 WS 信号从 I2S 接口所连接的外部主器件输入。这样，用户便不需要配置时钟。

应遵循如下配置步骤：

1. 将 SPIx_I2SCFGR 寄存器的 I2SMOD 位置 1 以选择 I2S 模式，通过 I2SSTD[1:0]位选择 I2S 标准，通过 DATLEN[1:0]位选择数据长度并通过配置 CHLEN 位选择帧中每个通道的位数。此外，通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0]位选择从器件的模式（发送或接收）。
2. 如果需要，通过对 SPIx_CR2 寄存器执行写操作来选择所有可能的中断源和 DMA 功能。
3. SPIx_I2SCFGR 寄存器的 I2SE 位必须置 1。

发送序列

当外部主器件发送时钟并且通过 NSS_WS 信号请求传输数据时，发送序列开始。必须首先使能从器件，然后外部主器件才能开始通信。主器件开始通信前，还必须加载 I2S 数据寄存器。

对于 I2S、MSB 对齐和 LSB 对齐模式，要写入数据寄存器的第一个数据项对应于左通道的数据。通信开始时，数据从发送缓冲区传输到移位寄存器。TXE 标志随即置 1，以请求将右通道的数据写入 I2S 数据寄存器。

CHSIDE 标志指示将发送的数据对应的通道。与主发送模式相比，在从模式下，CHSIDE 由来自外部主器件的 WS 信号触发。这意味着，从器件需要首先为发送第一个数据做好准备，然后主器件才能产生时钟。WS 置位意味着首先发送左通道数据。

说明：必须要在主器件发出的第一个时钟出现在 CK 线上至少 2 个 PCLK 周期之前置位 I2SE。

首位发送期间，数据按半字从内部总线并行加载到 16 位移位寄存器中，然后以串行方式移位并输出到 MOSI/SD 引脚（MSB 在前）。每次数据从发送缓冲区传输到移位寄存器后，TXE 标志都将置 1，如果 SPIx_CR2 寄存器的 TXEIE 位置 1，将产生中断。

注意：仅当 TXE 标志为 1 时，才可以尝试向发送缓冲区写入数据。

有关各种 I2S 标准模式中读操作的更多详细信息，请参见“24.6.3 支持的音频协议”。

为确保连续进行音频数据发送，必须在当前数据发送结束前将下一个要发送数据写入 SPIx_DR 寄存器。如果在数据尚未写入 SPIx_DR 寄存器时下一个数据通信的首个时钟边沿到来，下溢标志将置 1 并可能产生中断。通过这种方式，软件可以获知所传输的数据不正确。如果 SPIx_CR2 寄存器的 ERRIE 位置 1，则当 SPIx_SR 寄存器中的 UDR 标志变为 1 时，将产生中断。这种情况下，必须关闭 I2S 并从左通道开始重新启动数据传输。

要通过将 I2SE 位清零来关闭 I2S，必须等待 TXE=1 且 BSY=0。

接收序列

此工作模式与发送模式相同，只有第 1 点存在不同（请参见“24.6.7 I2S 从模式”所述的步骤），即通过 SPIx_I2SCFGR 寄存器的 I2SCFG[1:0]位设置从器件接收模式。

无论数据长度或通道长度如何，音频数据始终按 16 位数据包进行接收。这意味着，每当接收缓冲区填满时，SPIx_SR 寄存器中的 RXNE 标志即置 1，并且如果 SPIx_CR2 寄存器的 RXNEIE 位置 1，还将产生中断。所接收的右通道或左通道的音频值可能通过一次或两次接收操作进入接收缓冲区，具体取决于数据长度和通道长度配置。

每次接收要从 SPIx_DR 寄存器读取数据时，CHSIDE 标志都将更新。该标志由外部主器件所管理的外

部 WS 线路触发。

读取 SPIx_DR 寄存器即会使 RXNE 位清零。有关各种 I2S 标准模式中读操作的更多详细信息，请参见“24.6.3 支持的音频协议”。

如果在先前收到的数据尚未读取时又接收到新数据，将产生上溢错误，OVR 标志将置 1。如果 SPIx_CR2 寄存器的 ERRIE 位置 1，将产生中断以指示该错误。

要在接收模式下关闭 I2S，必须在接收到最后一个 RXNE=1 后立即将 I2SE 清零。

说明：外部主器件应能够通过音频通道以 16 位或 32 位数据包发送/接收数据。

24.6.8 I2S 状态标志

应用程序可通过三个状态标志来全面监视 I2S 总线的状态。

忙标志（BSY）

BSY 标志由硬件置 1 和清零（软件写入该位不生效）。该标志表示 I2S 通信层的状态。

BSY 置 1 时，表示 I2S 正在忙于通信。在主接收模式（I2SCFG=11）中，BSY 标志的情况例外，该标志在接收期间仍保持低电平。

如果软件需要禁止 I2S，可使用 BSY 标志检测传输是否结束。这可以避免损坏最后传输的数据。为此，必须严格遵循下述步骤。

在传输开始时（I2S 处于主接收器模式时除外），将 BSY 标志置 1。

出现以下情况时，BSY 标志被硬件清零：

- 传输完成时（主发送模式除外，在该模式下通信是连续的）
- 禁止 I2S 时

当通信连续时：

- 在主发送模式下，BSY 标志在所有传输期间均保持高电平。
- 在从模式下，BSY 标志在每次传输之间变为低电平并持续一个 I2S 时钟周期。

说明：请勿使用 BSY 标志处理每次数据发送或接收，最好改用 TXE 标志和 RXNE 标志。

发送缓冲区为空（TXE）

如果此标志置 1，表示发送缓冲区为空，可将要发送的下一个数据加载到其中。发送缓冲区已包含要发送的数据时，TXE 标志复位。禁止 I2S（I2SE 位复位）时，该标志也会复位。

接收缓冲区非空（RXNE）

此标志置 1 时，表示接收缓冲区中存在有效的已接收数据。读取 SPIx_DR 寄存器时，该标志复位。

通道方向（CHSIDE）

在发送模式下，此标志将在 TXE 变为高电平时进行刷新。此标志指示 SD 上要传输的数据所属的通道。如果在从发送模式下发生下溢错误事件，此标志将不可靠，在恢复通信前需要关闭并重新开启 I2S。

在接收模式下，该标志将在 SPIx_DR 中接收数据时进行刷新。它表示接收的数据所属的通道。请注意，如果发生错误（例如 OVR），此标志将失去意义，应通过禁止并重新使能 I2S（根据需要更改配置）来复位。

此标志在 PCM 标准中没有意义（短帧和长帧模式）。

当 SPIx_SR 中的 OVR 或 UDR 标志置 1，并且 SPIx_CR2 中的 ERRIE 位也置 1 时，将产生中断。中断源被清除后，可通过读取 SPIx_SR 状态寄存器来将此中断清除。

24.6.9 I2S 错误标志

I2S 单元共有三个错误标志。

下溢标志（UDR）

在从发送模式下，如果在软件尚未将任何值加载到 SPIx_DR 之前出现第一个数据发送时钟，此标志将置 1。SPIx_I2SCFGR 寄存器中的 I2SMOD 位置 1 时，可以使用此标志。如果 SPIx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。

UDR 位通过 SPIx_SR 寄存器上的读操作进行清零。

上溢标志（OVR）

如果在尚未从 SPIx_DR 寄存器读取上一个数据时又接收到新数据，此标志将置 1。因此，传入的数据将丢失。如果 SPIx_CR2 寄存器中的 ERRIE 位置 1，可产生中断。

这种情况下，将不会用接收到的新数据更新接收缓冲区的内容。对 SPIx_DR 寄存器执行的读操作将返回先前正确接收的数据。主器件后续发送的所有其它半字都将丢失。

要将 OVR 位清零，应首先对 SPIx_DR 寄存器执行读操作，然后再对 SPIx_SR 寄存器进行读访问。

帧错误标志（FRE）

仅当 I2S 配置为从模式时，此标志才可由硬件置 1。如果外部主器件没有按照从器件期望的那样改变 WS 线，则此标志将置 1。如果失去同步，要从此状态中恢复并将外部主器件与 I2S 从器件重新同步，需执行下列步骤：

1. 禁止 I2S。
2. 在 WS 线上检测到正确的电平时将其重新使能（WS 线在 I2S 模式下为高电平，在 MSB 对齐、LSB 对齐或 PCM 模式下为低电平）。

主器件与从器件之间的同步失效可能是由于 SCK 通信时钟或 WS 帧同步信号线上存在噪音干扰。如果 ERRIE 位置 1，可产生错误中断。读取状态寄存器时，同步失效标志（FRE）由软件清零。

24.6.10 DMA 特性

在 I2S 模式下，DMA 的工作方式与在 SPI 模式下完全相同。除了由于不存在数据传输保护机制，I2S 模式下没有 CRC 功能外，没有其它差别。

24.7 I2S 中断

表 24-4 为 I2S 中断的列表。

表 24-4 I2S 中断请求

中断事件	事件标志	使能控制位
发送缓冲区为空	TXE	TXEIE
接收缓冲区非空	RXNE	RXNEIE
上溢错误	OVR	ERRIE
下溢错误	UDR	
帧错误	FRE	

24.8 SPI 寄存器

基地址：0x4001 3000

空间大小：0x400

24.8.1 SPI 控制寄存器 1（SPI_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“2 线单向”模式下数据的输出方向。在多个从设备的配置中，在未被访问的从设备上该位被置 1，使得只有被访问的从设备有输出，从而不会造成数据线上数据冲突。 • 0：全双工（发送和接收） • 1：输出禁止（只收模式）
位 9	SSM：软件从机管理（Software slave management） 当 SSM 被置位时，NSS 引脚上的电平由 SSI 位的值决定。 • 0：禁止软件从设备管理 • 1：启用软件从设备管理

位 8	SSI：内部从设备选择（Internal slave select） 该位只有当 SSM 位为 1 的时候才有效。它决定了 NSS 上的电平，在 NSS 引脚上的 I/O 操作无效。
位 7	LSBFIRST：帧格式（Frame format） • 0：先发送 MSB • 1：先发送 LSB
位 6	SPE：SPI 使能（SPI enable） • 0：禁止 SPI 设备 • 1：开启 SPI 设备
位 5:3	BR[2:0]：波特率控制（Baud rate control） • 000：$f_{PCLK}/2$ • 001：$f_{PCLK}/4$ • 010：$f_{PCLK}/8$ • 011：$f_{PCLK}/16$ • 100：$f_{PCLK}/32$ • 101：$f_{PCLK}/64$ • 110：$f_{PCLK}/128$ • 111：$f_{PCLK}/256$
位 2	MSTR：主设备选择（Master selection） • 0：配置为从设备 • 1：配置为主设备
位 1	CPOL：时钟极性（Clock polarity） • 0：空闲状态时，SCK 保持低电平。 • 1：空闲状态时，SCK 保持高电平。
位 0	CPHA：时钟相位（Clock phase） • 0：第一个时钟沿对准第一位数据。 • 1：第二和时钟沿对准第一位数据。

24.8.2 SPI 控制寄存器 2（SPI_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	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 SPI_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 SPI_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPI_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 SPI_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 SPI_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPI_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 事件时的 RXFIFO 的阈值。 • 0：如果 FIFO 的存储水平大于或等于 1/2（16 位），产生 RXNE 事件。 • 1：如果 FIFO 的存储水平大于或等于 1/4（8 位），产生 RXNE 事件。
位 11:8	DS[3:0]：数据位宽（Data size） 该位域配置 SPI 传输数据的位宽： • 0000：不使用 • 0001：不使用 • 0010：不使用 • 0011：4 位 • 0100：5 位 • 0101：6 位 • 0110：7 位 • 0111：8 位 • 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，它们会被迫赋值“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，I2S 模式中的 OVR）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

24.8.3 SPI 状态寄存器（SPI_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	UDR	CHSIDE	TXE	RXNE
			r		r		r	r	r	r	rc_w0	r	r	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（或 I2S）不忙 • 1：SPI（或 I2S）通信忙或发送缓冲区不为空 该位由硬件设置或清零。
位 6	OVR：溢出标志（Overrun flag） • 0：没有发生溢出 • 1：发生溢出 该位由硬件置位，由软件序列复位。
位 5	MODF：模式故障（Mode fault） • 0：无模式故障发生 • 1：模式故障发生 该位由硬件置位，由软件序列复位。
位 4	CRCERR：CRC 错误标志（CRC error flag） • 0：收到的 CRC 值和 SPI_RXCRCR 的值是匹配的。 • 1：收到的 CRC 值和 SPI_RXCRCR 值不匹配。 该位由硬件置位，由软件清零。
位 3	UDR：欠载标志（Underrun flag） • 0：没有欠载发生 • 1：欠载发生 该位由硬件置位，由软件序列复位。

位 2	CHSIDE：通道标志（Channel side） • 0：需要传输或者接收左声道 • 1：需要传输或者接收右声道
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

24.8.4 SPI 数据寄存器（SPI_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。

24.8.5 SPI 的 CRC 多项式寄存器（SPI_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。

24.8.6 SPI 接收 CRC 寄存器（SPI_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 SPI_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。

	• 当数据帧格式被设置为 8 位（SPI_CR1 的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 • 当数据帧格式为 16 位（SPI_CR1 的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

24.8.7 SPI 发送 CRC 寄存器（SPI_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 SPI_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 • 当数据帧格式被设置为 8 位（SPI_CR1 中的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行。 • 当数据帧格式为 16 位（SPI_CR1 的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行。

24.8.8 SPI_I2S 配置寄存器（SPI_I2SCFGR）

偏移地址：0x1C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				I2SMOD	I2SE	I2SCFG[1:0]		PCMSYNC	Res	I2SSTD[1:0]		CKPOL	DATLEN[1:0]		CHLEN
				rw	rw	rw		rw		rw		rw	rw		rw

位 15:12	Res：保留 必须保持复位值。
位 11	I2SMOD：I2S 模式选择（I2S mode selection） • 0：选中 SPI 模式 • 1：选中 I2S 模式
位 10	I2SE：I2S 使能（I2S enable） • 0：I2S 的外设被禁用 • 1：I2S 外设被使能
位 9:8	I2SCFG[1:0]：I2S 配置模式（I2S configuration mode） • 00：从机发送 • 01：从机接收 • 10：主机发送 • 11：主机接收

位 7	PCMSYNC：PCM 帧同步（PCM frame synchronization） • 0：短帧同步 • 1：长帧同步
位 6	Res：保留 必须保持复位值。
位 5:4	I2SSTD[1:0]：I2S 标准选择（I2S standard selection） • 00：I2S 飞利浦标准 • 01：高字节对齐标准（左对齐） • 10：低字节对齐标准（右对齐） • 11：PCM 标准
位 3	CKPOL：静止态时钟极性（Inactive state clock polarity） • 0：I2S 时钟静止态为低电平。 • 1：I2S 时钟静止态为高电平。
位 2:1	DATLEN[1:0]：待传输数据长度（Data length to be transferred） • 00：16 位数据长度 • 01：24 位数据长度 • 10：32 位数据长度 • 11：不允许
位 0	CHLEN：声道长度（每个音频通道的数据位数）（Channel length (number of bits per audio channel)） • 0：16 位宽 • 1：32 位宽 只有在 DATLEN=00 时，该位的写操作才有意义；否则声道长度都由硬件固定为 32 位。

24.8.9 SPI_I2S 预分频寄存器（SPI_I2SPR）

偏移地址：0x20

复位值：0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res						MCKOE	ODD	I2SDIV[7:0]							
						rw	rw	rw							

位 15:10	Res：保留 必须保持复位值。
位 9	MCKOE：主时钟输出使能（Master clock output enable） • 0：主时钟输出被禁用 • 1：主时钟输出启用
位 8	ODD：奇系数预分频（Odd factor for the prescaler）

	• 0：实际分频系数=I2SDIV*2 • 1：实际分频系数=（I2SDIV*2）+1
位 7:0	I2SDIV[7:0]：I2S 线性预分频器（I2S linear prescaler） 不允许设置 I2SDIV[7:0]=0 或者 I2SDIV[7:0]=1 的值。

25 除法开方运算单元（DVSQ）

25.1 DVSQ 主要特性

DVSQ 运算单元支持 32 位的无符号除法（UDIV）和带符号除法（SDIV），和 32 位无符号开方运算。除法运算同时支持 MOD 操作，运算完成后，商和余数同时更新到相应的寄存器供软件读取。

- 支持 32 位带符号数和无符号数除法，支持无符号 32 位开方运算。在同一时刻，DVSQ 加速器不能同时支持除法和开方运算，只能两者选其一来执行。32 位/32 位有符号或无符号整数除法（同时获取商和余数）。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（bit）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。

结构示图如下：

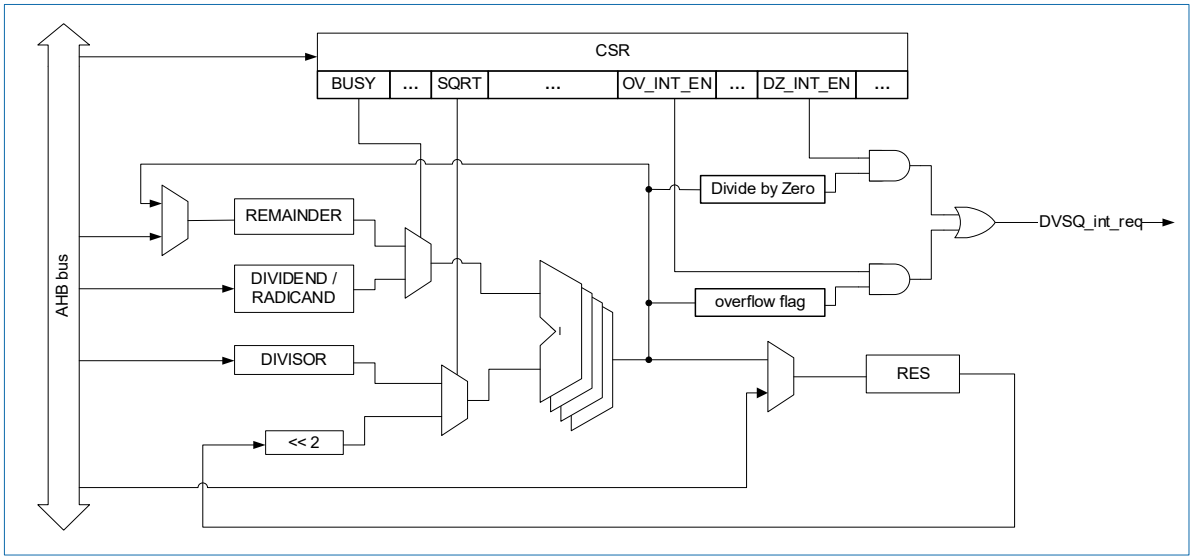


图 25-1 DVSQ 结构图

25.2 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：

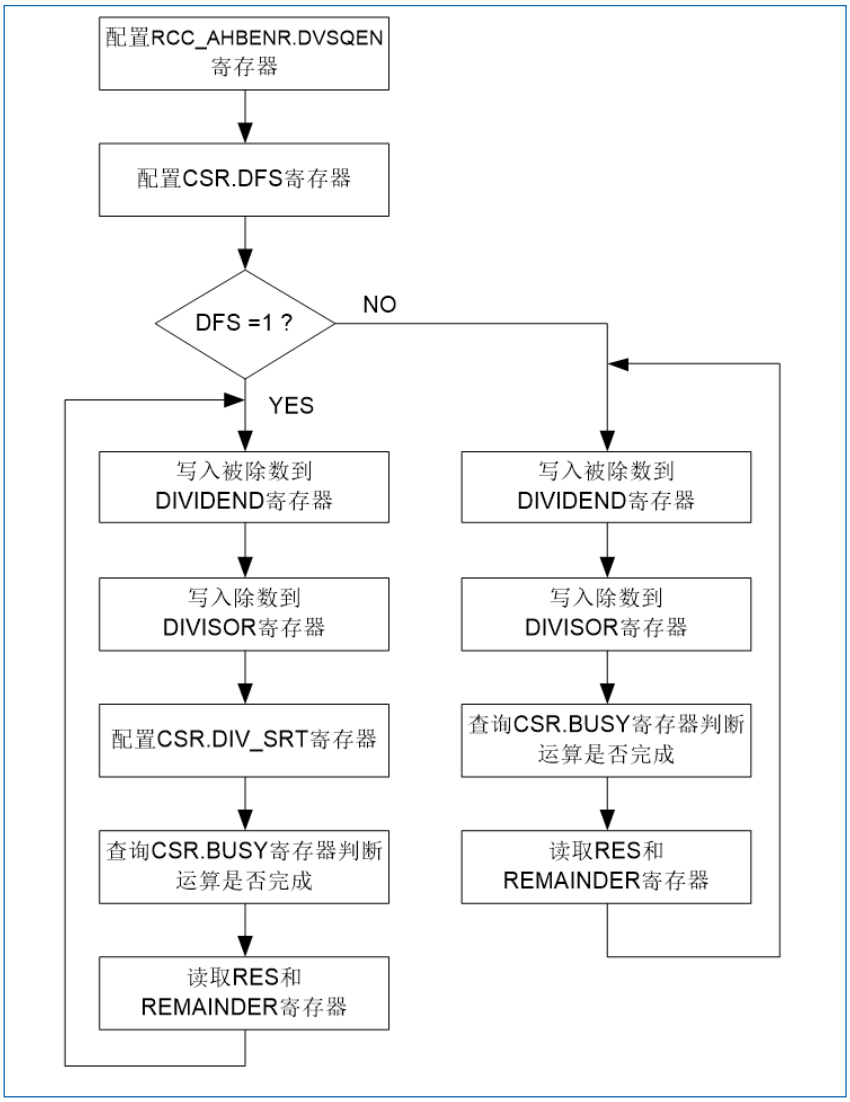


图 25-2 除法运算流程示意图

25.3 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 25-1 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

25.4 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

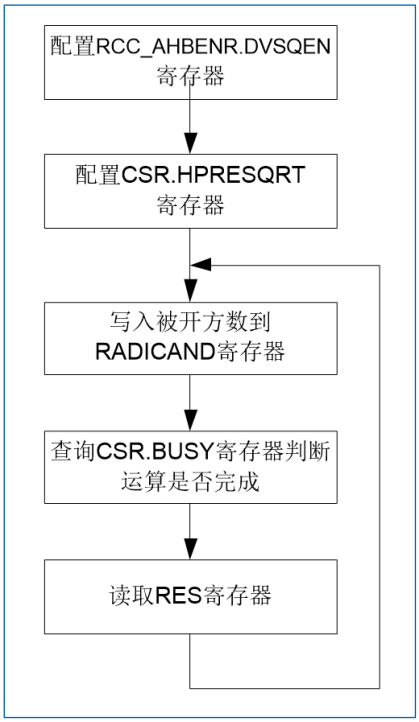


图 25-3 开方运算流程图

25.5 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 25-2 开放运行时间（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 25-3 开放运行时间（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
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

被开方数（有效位的位置）	运算时间[周期数]
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

25.6 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。

25.7 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

无论是字节写、半字写还是字写 DVSQ_DIVISOR 寄存器都会使除法运算开始。

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相应中断。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。

25.8 DVSQ 寄存器

基地址：0x4003 0000

空间大小：0x400

25.8.1 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。

25.8.2 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）

25.8.3 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：执行无符号是除法。 ● 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。

25.8.4 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）

25.8.5 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。

25.8.6 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

26 调试支持（DBG）

26.1 概述

HK32F031 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 HK32F031 MCU 相连并进行调试时，将使用调试功能。

提供一个调试接口：

- 串行接口

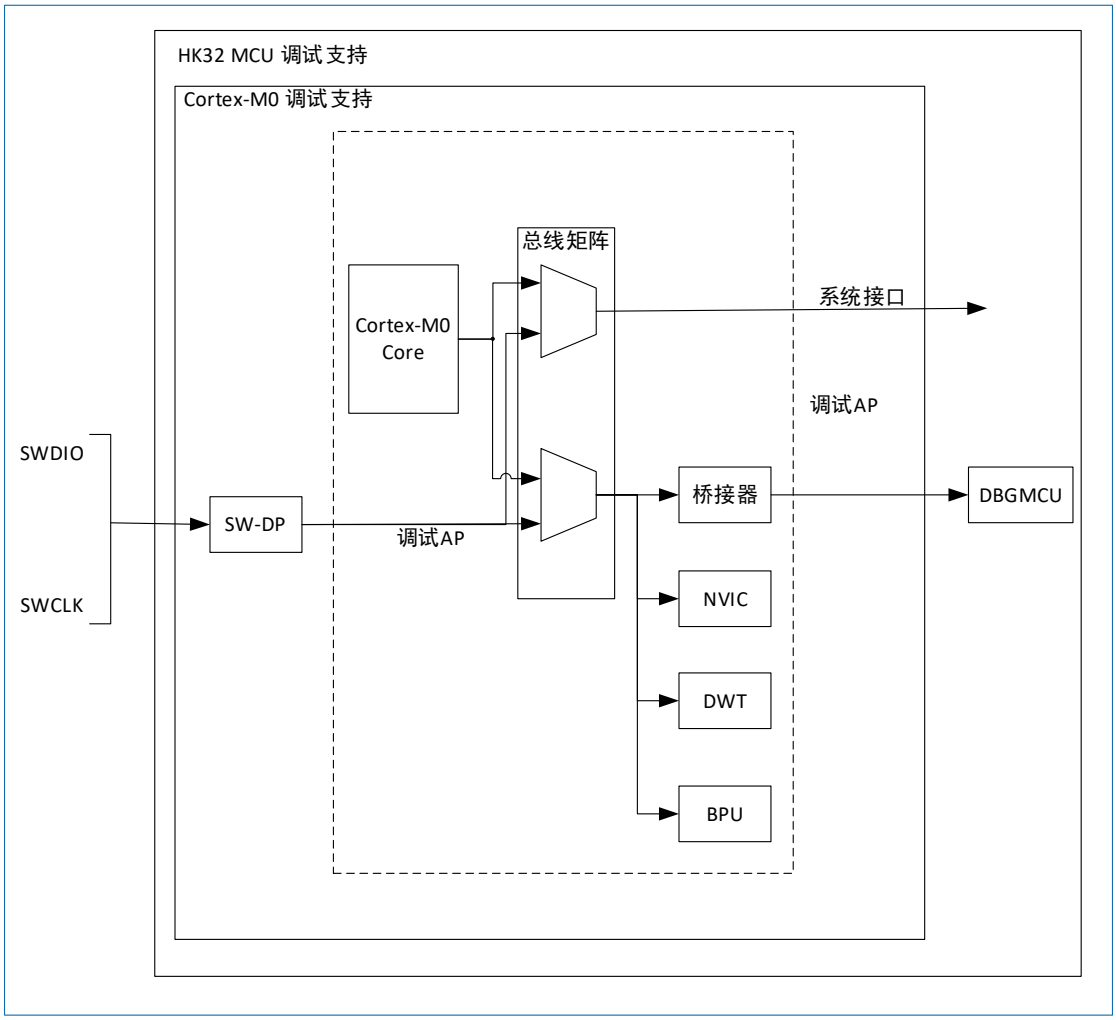


图 26-1 HK32F031 MCU 和 Cortex®-M0 级调试支持框图

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 HK32F031 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。

26.2 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

26.3 引脚排列和调试端口引脚

HK32F031 MCU 的不同封装有不同的有效引脚数。

26.3.1 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 26-1 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	PA13
SWCLK	I	串行线时钟	PA14

26.3.2 SW-DP 引脚分配

复位（SYSRESETn 或 PORESEaTn）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“7.1.2 I/O 引脚复用功能复用器和映射”。

26.3.3 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。

26.4 SWD 端口

26.4.1 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。

26.4.2 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 26-2 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®-M0TRM。

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 26-3 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 26-4 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。

26.4.3 SW-DP 状态机（复位、空闲状态、ID 代码）

SW-DP 的状态机有一个用于标识 SW-DP 的内部 ID 代码。该代码符合 JEP-106 标准。此 ID 代码是默

认的 ARM®代码，设置为 0x0BB11477（相当于 Cortex®-M0）。

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 r1p0TRM。

26.4.4 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。

26.4.5 SW-DP 寄存器描述

当 APnDP=0 时能够访问这些寄存器。

表 26-5 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。

26.4.6 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 26-6 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）

26.5 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 26-7 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®-M0TRM。

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）

26.6 BPU（断点单元）

Cortex®-M0BPU 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。

26.6.1 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。

26.7 DWT（数据观察点）

Cortex®-M0 DWT 实现提供了两个观察点寄存器组。

26.7.1 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。

26.7.2 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。

26.8 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

26.8.1 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：

- 在休眠模式下，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/待机模式下，调试程序必须事先将 DBG_STOP/DBG_STANDBY 位置 1。这样便可使能内部 RC 振荡器时钟，以在停机/待机模式下为 FCLK 和 HCLK 提供时钟。

26.8.2 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。

26.9 DBGMCU 寄存器

基地址：0x4001 5800

空间大小：0x400

26.9.1 MCU 器件 ID 代码寄存器（DBGMCU_IDCODE）

偏移地址：0x00

复位值：0x1000 6440

说明：该寄存器仅支持读和 32 位访问。

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

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

位 31:16	REV_ID[15:0]：版本标识符（Revision identifier） 0x1000：版本编号 1.0
位 15:12	Res：保留 必须保持复位值。
位 11:0	DEV_ID[11:0]：器件标识符（Device identifer） 复位后从 Flash 加载。 该字段指定设备 ID 为：0x444

26.9.2 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_STANDBY	DBG_STOP	Res
													rw	rw	

位 31:3	Res：保留 必须保持复位值。
位 2	DBG_STANDBY：调试待机模式（Debug Standby mode）

	• 0：将整个数字部分断电（FCLK=关闭，HCLK=关闭）。 从软件角度来看，退出待机模式相当于取复位向量（指示 MCU 从待机状态恢复的几个状态位除外）。 • 1：在这种情况下，数字部分不断电，FCLK 和 HCLK 由保持激活状态的内部 RC 振荡器提供（FCLK=开启，HCLK=开启）。 MCU 会在待机模式期间产生系统复位，这样退出待机模式相当于取复位向量。
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。

26.9.3 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_I2C1_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	DBG_RTC_STOP	Res	DBG_TIM14_STOP	Res			DBG_TIM6_STOP	Res		DBG_TIM3_STOP	DBG_TIM2_STOP
			rw	rw	rw		rw				rw			rw	rw

位 31:22	Res：保留 必须保持复位值。
位 21	DBG_I2C1_SMBus_TIMEOUT：内核停止时，停止 I2C1SMBus 超时模式（I2C1 SMBUS timeout mode stopped when core is halted） • 0：行为方式与正常模式下相同。 • 1：冻结 I2C1SMBus 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	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 9	Res：保留 必须保持复位值。
位 8	DBG_TIM14_STOP：内核停止时，停止调试 TIM14（Debug TIM14 stopped when core is halted） • 0：即使内核停止，TIM14 计数器时钟仍继续工作。 • 1：内核停止时，TIM14 计数器时钟停止。
位 7:5	Res：保留 必须保持复位值。
位 4	DBG_TIM6_STOP：内核停止时，停止调试 TIM6（Debug TIM6 stopped when core is halted） • 0：即使内核停止，TIM6 计数器时钟仍继续工作。 • 1：内核停止时，TIM6 计数器时钟停止。
位 3:2	Res：保留 必须保持复位值。
位 1	DBG_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。

26.9.4 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													DBG_TIM17_STOP	DBG_TIM16_STOP	Res
													rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res				DBG_TIM1_STOP	Res										
				rw											

位 31:19	Res：保留 必须保持复位值。
位 18	DBG_TIM17_STOP：内核停止时，停止调试 TIM17（Debug TIM17 stopped when core is halted） • 0：即使内核停止，TIM17 计数器时钟仍继续工作。 • 1：内核停止时，TIM17 计数器时钟停止。
位 17	DBG_TIM16_STOP：内核停止时，停止调试 TIM16（Debug TIM16 stopped when core is halted） • 0：即使内核停止，TIM16 计数器时钟仍继续工作。 • 1：内核停止时，TIM16 计数器时钟停止。
位 16:12	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。

27 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 JTAG/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。

27.1 存储器大小寄存器

27.1.1 Flash 大小寄存器

基地址：0x1FFF F7CC

复位值：0xXXXX

说明：此处 X 表示出厂前编程设置。

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

位 15:0	F_SIZE[15:0]：Flash 大小（Flash size） 此字段中存储的值指示器件的 Flash 大小（用 Kbyte 表示）。 示例：0x0080=128Kbyte

27.2 唯一设备 ID 寄存器（96 位）

基地址：0x1FFF F7AC

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。

27.2.1 UID 寄存器 0

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]															

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

位 31:0	U_ID[31:0]：晶圆编号和芯片批号（Unique ID bits） 其中： • U_ID[31:24]：晶圆编号（WAF_NUM[7:0]），为 8 位无符号数。 • U_ID[23:0]：芯片批号（LOT_NUM[55:32]），用 ASCⅡ代码表示。

27.2.2 UID 寄存器 1

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]：芯片批号（LOT_NUM[31:0]），用 ASCⅡ代码表示。（Unique ID bits）

27.2.3 UID 寄存器 2

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）

28 缩略语与术语

28.1 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。

28.2 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
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。

28.3 术语

名称	中文描述
Byte	字节，8 位数据长度。
Half word	半字，16 位的数据或指令长度。
Option byte	选项字节，保存在 Flash 中的 MCU 配置字节。

名称	中文描述
Word	字，32 位的数据或指令长度。

29 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