武汉芯源半导体有限公司
WUHAN XINYUAN SEMICONDUCTOR CO., LTD

CW32L011 用户手册

ARM® Cortex®-M0+ 32 位低功耗微控制器

版本号：Rev 1.1

2026 年 06 月

声明

重要声明

本参考手册针对应用程序开发人员，它提供了关于如何使用 CW32L011 微控制器内存和外设的完整信息。仅适用于 CW32L011 设备。

免责声明

我们保留随时对文档内容进行更新的权利，文档内容如有改动，恕不另行通知。请在购买产品前到我们的官网上下载最新版本手册。

版权声明

技术支持

若您在使用过程中有任何意见或建议，请随时与我们联系。

网址：www.whxy.com

通信地址：湖北省武汉市东湖新技术开发区光谷大道武大园三路 5 号

邮编：430070

目录

1 文档约定

1.1 排版约定

本文中的排版惯例是：

斜体 表示特殊术语或引文，或者提示注意项。

粗体 表示组件或信号名称，如用于无章节序号的标题说明、图表的标题等。

大写英文 用于具有特定技术含义的术语，并包含在词汇表中，如寄存器或位域名称。

彩色文本 表示链接。包括：

- URL 外部链接，如 www.whxy.com。
- 交叉引用，本文的内部章节互相引用。
- 内部链接，图、表或附录、词汇条目。

1.2 寄存器协议

符号示例	描述
0x53CA	带‘0x’前缀的数字字符串表示十六进制数，数值为数字 0 ~ 9 或 A ~ F 大写英文
REGNAME[n]	REGNAME 的特定位，REGNAME 可以是寄存器或寄存器字段，例如，CR1[2] 指的是 CR1 寄存器 (字段) 的第 2 位
REGNAME[m:n]	REGNAME 的特定位，REGNAME 可以是寄存器或寄存器字段，例如，CR1[7:5] 指的是 CR1 寄存器 (字段) 的第 7 位 ~ 第 5 位
RW	可读可写，软件可以读写这些位域
RO	只读，软件只能读取这些位域
WO	只写，软件只能写入该位域，读取该位域时将返回无效数据
RW0	软件可以读写该位域，只能对该位域写入 0，写入 1 对该位域无影响
RW1	软件可以读写该位域，只能对该位域写入 1，写入 0 对该位域无影响
R0W1	软件读取该位域为 0，只能对该位域写入 1，写入 0 对该位域无影响
R1W0	软件读取该位域为 1，只能对该位域写入 0，写入 1 对该位域无影响
RFU	保留位域，请保持默认值
Word	一个字的数据长度为 32 bit
Half Word	一个半字的数据长度为 16 bit
Byte	一个字节的数据长度为 8 bit

2 系统和存储器概述

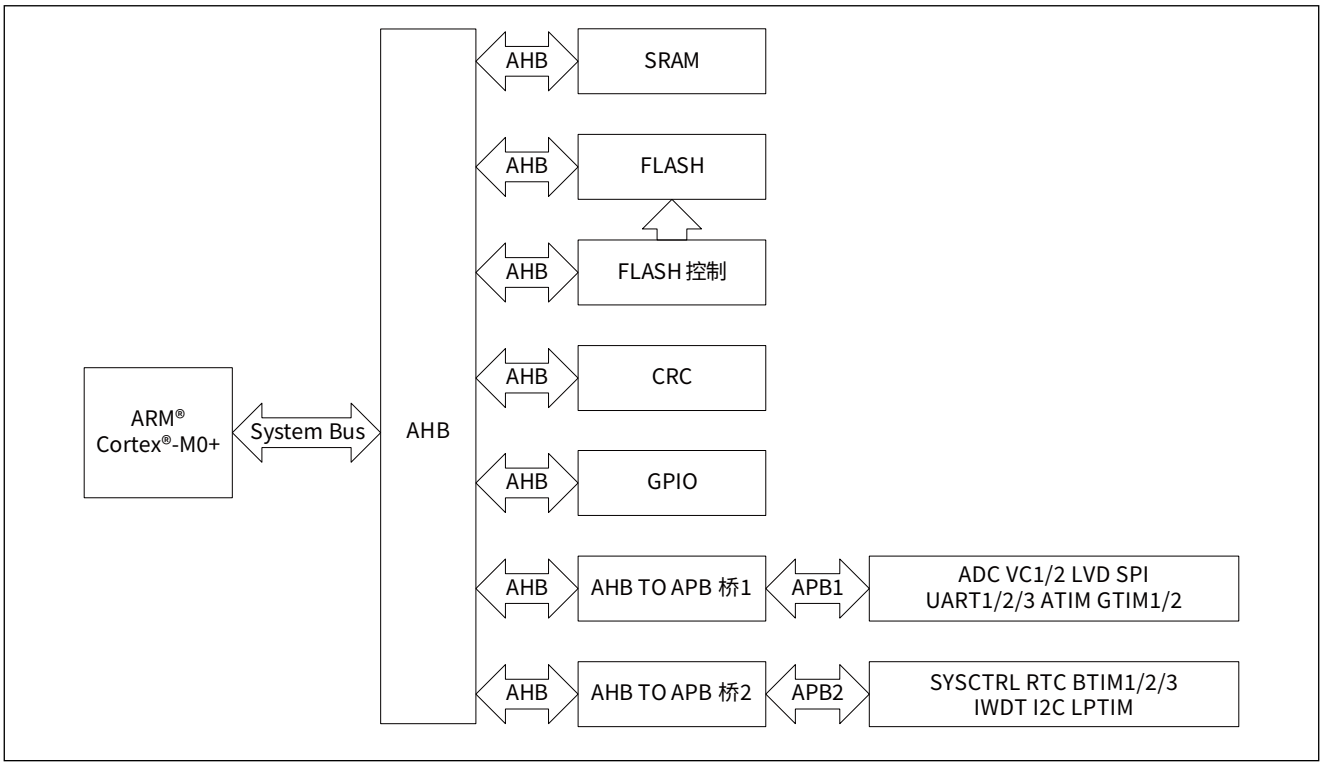
2.1 系统架构

CW32L011 微控制器系统包含：

- 1 个主设备：
  - ARM® Cortex®-M0+ 内核
- 多个从设备：
  - 片上 SRAM
  - 片上 FLASH
  - FLASH 控制器
  - CRC 冗余计算单元
  - GPIO 端口
  - AHB 到 APB1 转换桥及 APB1 总线上所有设备
  - AHB 到 APB2 转换桥及 APB2 总线上所有设备

主从设备通过 AHB 总线矩阵进行连接，系统架构如下图所示：

图 2-1 系统架构



- 系统总线

  实现 M0+ 微处理器的外设总线和 AHB 总线的连接。

- AHB TO APB 桥 1/2

  提供 AHB 总线到 APB1/APB2 总线的完全同步的连接，即 AHB 和 APB 总线的桥接。

2.2 存储器组织

2.2.1 概述

CW32L011 内核为 32 位的 ARM® Cortex®-M0+ 微处理器，最大寻址空间为 4GB。芯片内置的程序存储器、数据存储器、各外设及端口寄存器被统一编址在同一个 4GB 的线性地址空间内。

存储器中字节组织为小端格式。一个字存储空间的最低字节数据为字的最低有效位，最高字节数据为最高有效位。

例：

将 0x1122 3344 存放在地址为 0x2000 0000 的存储器空间中，实际存放结果是：

0x20000000 字节存放 0x44，

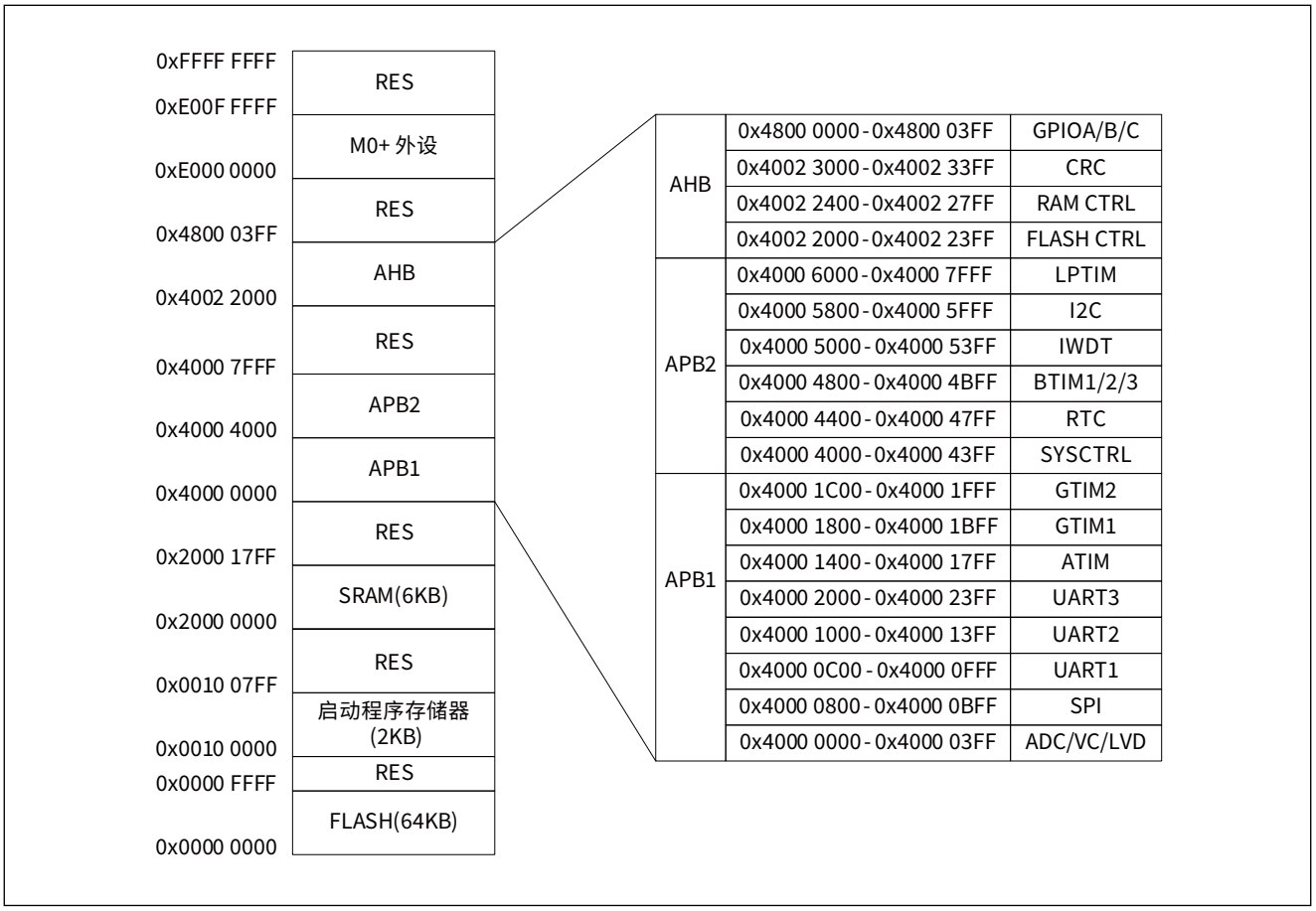
0x20000001 字节存放 0x33，

0x20000002 字节存放 0x22，

0x20000003 字节存放 0x11。

系统地址分配如下图所示，RES 为保留区域。

图 2-2 系统地址分配



2.2.2 存储器映射和寄存器边界地址

片上存储器及各外设的详细起始地址空间分配，如下表所示：

表 2-1 存储器和外设地址分配

设备或总线	边界地址	大小	对应外设
主 FLASH 存储器	0x0000 0000 - 0x0000 FFFF	64KB	主 FLASH
启动程序存储器	0x0010 0000 - 0x0010 07FF	2KB	BootLoader
OTP 存储器	0x0010 0760 - 0x0010 0775	22B	OTP
SRAM 存储器	0x2000 0000 - 0x2000 17FF	6KB	SRAM
APB1 外设	0x4000 0000 - 0x4000 03FF	1KB	ADC/VC/LVD
	0x4000 0800 - 0x4000 0BFF	1KB	SPI
	0x4000 0C00 - 0x4000 0FFF	1KB	UART1
	0x4000 1000 - 0x4000 13FF	1KB	UART2
	0x4000 2000 - 0x4000 23FF	1KB	UART3
	0x4000 1400 - 0x4000 17FF	1KB	ATIM
	0x4000 1800 - 0x4000 1BFF	1KB	GTIM1
	0x4000 1C00 - 0x4000 1FFF	1KB	GTIM2
APB2 外设	0x4000 4000 - 0x4000 43FF	1KB	SYSCTRL
	0x4000 4400 - 0x4000 47FF	1KB	RTC
	0x4000 4800 - 0x4000 4BFF	1KB	BTIM123
	0x4000 5000 - 0x4000 53FF	1KB	IWDT
	0x4000 5800 - 0x4000 5FFF	1KB	I2C
	0x4000 6000 - 0x4000 7FFF	1KB	LPTIM
AHB 外设	0x4002 2000 - 0x4002 23FF	1KB	FLASH CTRL
	0x4002 2400 - 0x4002 27FF	1KB	RAM CTRL
	0x4002 3000 - 0x4002 33FF	1KB	CRC
	0x4800 0000 - 0x4800 03FF	1KB	GPIOA/B/C
M0+ 外设	0xE000 0000 - 0xE00F FFFF	1MB	M0+ 内核外设

2.3 片上 SRAM 存储器

CW32L011 内部集成 6KB 的片上 SRAM，起始地址为 0x2000 0000。SRAM 支持以字节 (8bit)、半字 (16bit) 或全字 (32bit)3 种位宽进行访问，能够被 CPU 以最大的系统时钟频率进行访问，零等待延迟。

- 奇偶校验

  SRAM 支持奇偶校验功能，芯片上电后默认打开，用户不可配置。

  SRAM 的数据总线为 36bit，包括 32bit 数据位以及 4bit 奇偶校验位（每 8bit 数据位配有 1bit 的奇偶校验位），用来增加存储器的健壮性。

  奇偶校验位在 CPU 对 SRAM 进行写入时计算和存储，在 CPU 对 SRAM 进行读取时自动检查。系统检测到奇偶校验失败后，会产生对应的中断标志。

关于 SRAM 的详细介绍，请参见 6 RAM 存储器章节。

2.4 片上 FLASH 闪存存储器

片上 FLASH 闪存由两部分物理区域组成：主 FLASH 存储器和启动程序存储器。

- 主 FLASH 存储器，共 64KB，地址空间为 0x0000 0000 - 0x0000 FFFF。该区域主要用于存放应用程序代码和用户数据，用户可编程。
- 启动程序存储器，共 2KB，地址空间为 0x0010 0000 - 0x0010 07FF。该区域主要用于存储 BootLoader 启动程序，在芯片出厂时已编程，用户不可更改。

FLASH 控制器实现对 FLASH 的各种操作（擦除、写、读取）。

FLASH 支持以字节 (8bit)、半字 (16bit) 或全字 (32bit)3 种位宽进行访问，访问的最高频率为 24MHz，如果系统配置的 HCLK 时钟频率高于 24MHz，则必须通过 FLASH 控制寄存器 FLASH_CR2 的 WAIT 位域配置合理的响应等待时间，才能保证 FLASH 被正确访问。

关于 FLASH 的详细介绍，请参见 7 FLASH 存储器章节。

2.5 一次性可编程 OTP 存储器

芯片内部有 22B 的存储器空间为 OTP 区域，地址空间为 0x0010 0760 – 0x0010 0775。该存储区域读写方法同 FLASH 区，但仅能写入 1 次，之后就只能被读取，不能擦除和写入。该区域主要用于存储不可更改的信息，如用户自定义的产品 UID、算法密钥等，在设备出厂时写入。

2.6 系统启动配置

设备支持以下 2 种不同的启动模式：

- 从主 FLASH 存储器启动，运行用户程序。
- 从启动程序存储器启动，固定运行芯片的 BootLoader 程序，此时用户可通过 UART1 接口（PA13/PA14）利用 ISP 通信协议进行 FLASH 编程。

CW32L011 进入 ISP 模式的步骤如下：

步骤 1：将芯片置于 RESET 状态；

步骤 2：向芯片的 UART1_RXD（SWDIO）提供 50kHz 的方波；

步骤 3：释放芯片的 RESET 状态并延时 5ms；

步骤 4：芯片进入 ISP 模式。

在 ISP 模式下可写入用户 CODE 代码，完成下载后需重新复位 MCU，以执行用户应用程序。

系统启动完成之后，CPU 从存储器的 0x0000 0000 地址获取堆栈顶的地址，并从存储器的 0x0000 0004 指示的地址开始执行代码。

BootLoader 程序位于启动程序存储器区域，由设备提供商在生产时进行编程。用户可以通过 UART1（引脚为 PA13/PA14）利用 ISP 通信协议进行 FLASH 编程。

2.7 注意事项

CW32L011 在使用中需要注意如下事项：

- FLASH、SRAM 以及 GPIOx_ODR、CRC_DR 等少数寄存器支持 8bit/16bit/32bit 访问方式，其它外设只支持 32bit 访问方式。对不支持 8bit/16bit 访问方式的地址进行 8bit/16bit 访问会产生 HardFault 硬件错误异常。
- 图 2-2 系统地址分配中的 RES 区域均为保留的地址空间，无对应的物理设备，如果对这些保留地址空间进行访问，会产生 HardFault 硬件错误异常。
- 芯片复位后，除了 SYSTICK 和 SRAM 外的所有外设时钟都处于关闭状态，用户在使用这些外设前必须通过外设时钟使能控制寄存器（AHB 外设时钟使能控制寄存器 SYSCTRL_AHBEN、APB 外设时钟使能控制寄存器 SYSCTRL_APBEN1/2）打开对应外设的配置时钟和工作时钟。没有打开外设配置时钟会导致访问失败：写入数据失败，且不会产生 HardFault 硬件错误异常，读数据的结果不可用。没有打开外设的工作时钟，则外设主要功能失效。

3 电源控制（PWR）与功耗

3.1 概述

CW32L011 工作时需要两组电源供电：工作电源（VDD、VSS）和模拟电源（VDDA、VSSA）。用户可以选择工作电源和模拟电源使用同一电源，也可以使用两组不同电源，但是二者相差不能超过 0.3V，详细电源内容请参阅数据手册相关章节。

CW32L011 内嵌一路 1.5V 低压差 LDO 稳压器，为芯片内部数字电源域供电。

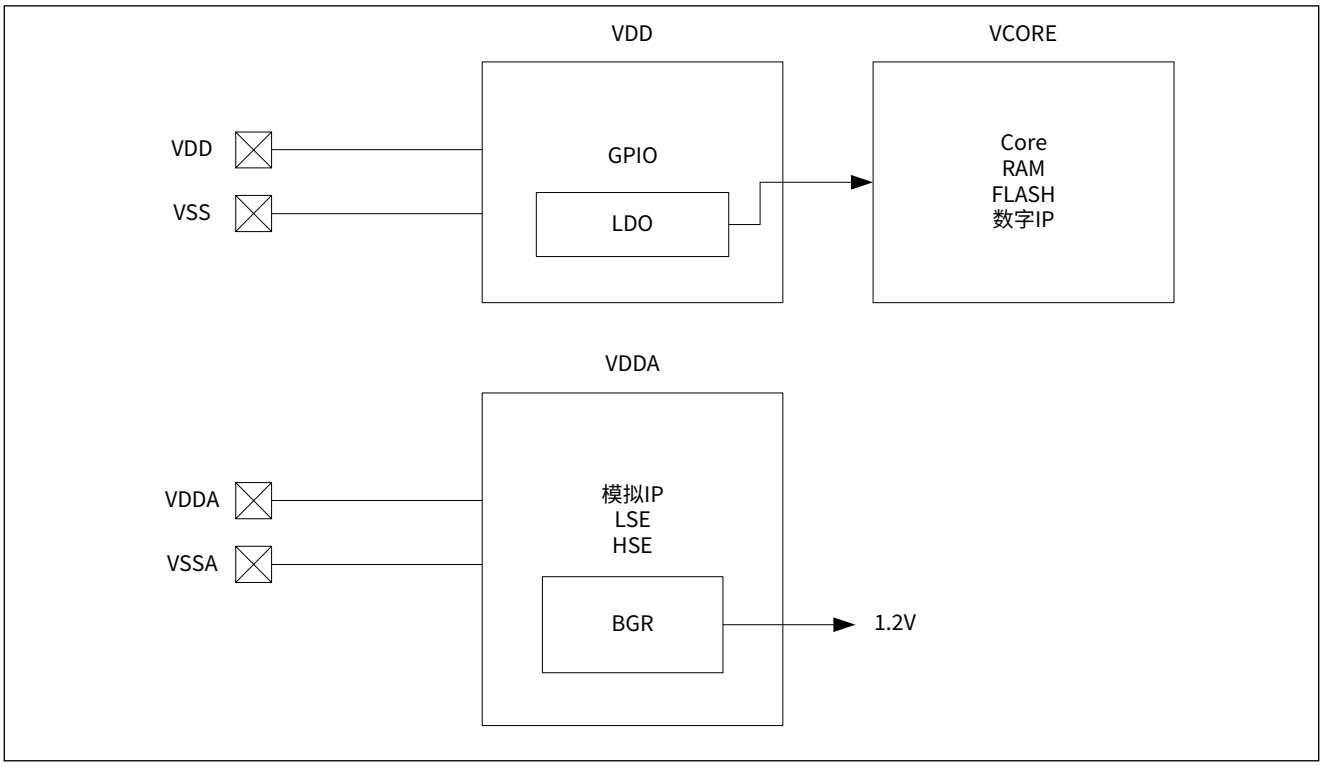
芯片内置参考电压生成器（BGR）电路，可为其他模拟模块提供参考电压。

注意：

模拟电源电压与参考电压的压差须高于 0.3V。关于电压参考的详细参数请参阅数据手册相关章节。

芯片内部电源域的划分与分配，如下图所示：

图 3-1 系统电源分配



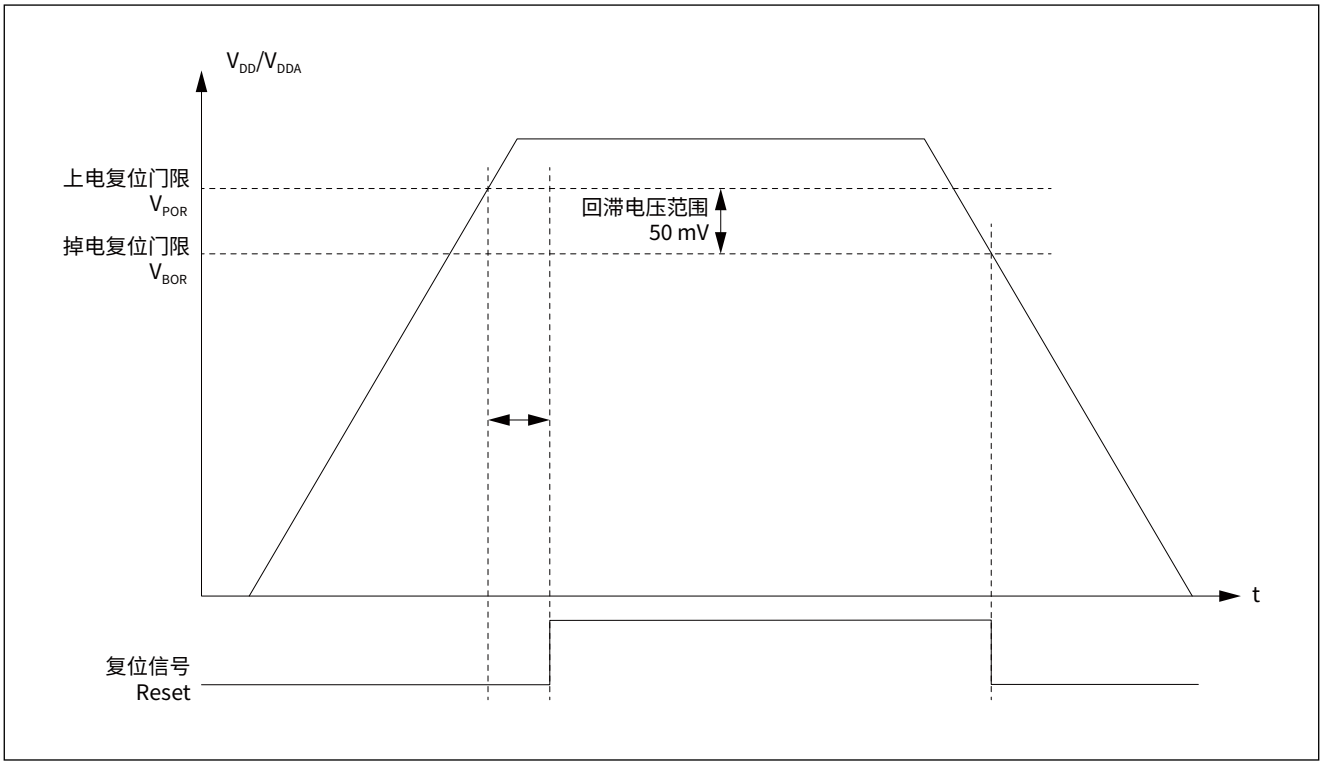
3.2 电源监控

上电复位（POR）/ 掉电复位（BOR）

CW32L011 集成了上电复位（POR）和掉电复位（BOR）电源监控电路，电源上电后始终处于工作状态。POR/BOR 同时监控 VDD 和 VDDA 电源电压，当监测到电源电压低于复位阈值（V_{BOR}）时，系统会进入复位状态。用户无需额外增加外部硬件复位电路。

电源上电启动以及电源跌落阶段的复位信号波形，如下图所示：

图 3-2 上电 / 掉电复位时序图



关于上电 / 掉电复位阈值的更多详细信息，请参阅本芯片的数据手册中的电气特征章节。

3.3 工作模式

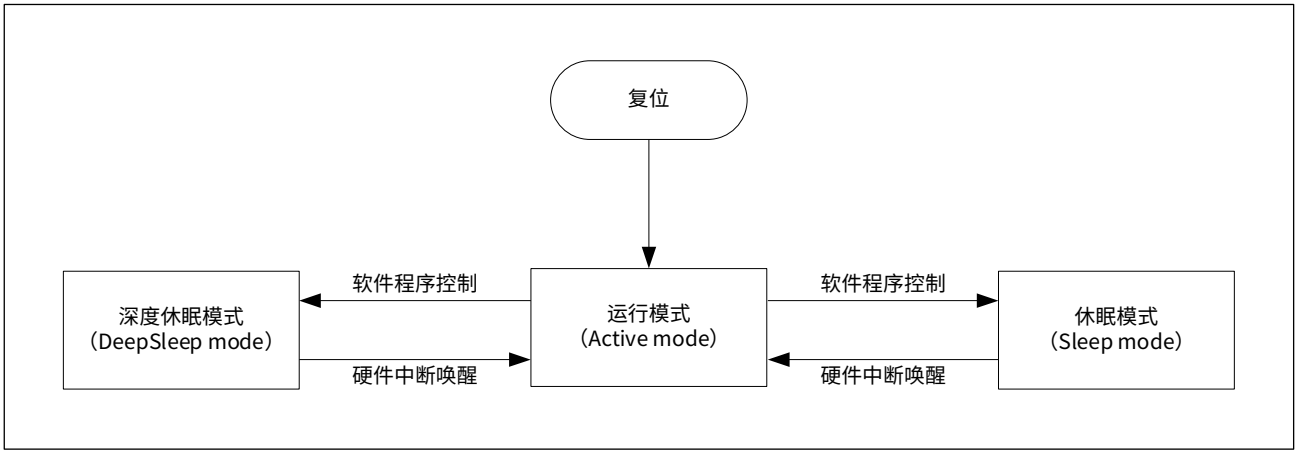
CW32L011 支持三种工作模式，由内嵌的电源管理模块自动完成电源的统一管理。三种工作模式是：

- 运行模式（Active mode）
- 休眠模式（Sleep mode）
- 深度休眠模式（DeepSleep mode）

电源上电后，系统自动进入运行模式。用户可通过软件程序，进入休眠或深度休眠两种低功耗运行状态；在低功耗运行状态时，可通过硬件中断触发唤醒机制，使系统返回到运行模式。

三种工作模式的转换机制，如下图所示：

图 3-3 工作模式转换机制



运行模式下 CPU 正常运行，所有模块用户均可正常使用。休眠模式下，CPU 停止运行，所有外设不受影响，所有 I/O 引脚保持状态不变。深度休眠模式下，CPU 停止运行，高速时钟（HSE、HSIOSC）自动关闭，低速时钟（LSE、LSI）保持原状态不变。

不同工作模式下 CPU 与时钟状态，如下表所示：

表 3-1 工作模式与 CPU 及时钟状态

工作模式	CPU 状态	高速时钟	低速时钟
运行模式（Active mode）	运行	ON/OFF	ON/OFF
休眠模式（Sleep mode）	停止	ON/OFF	ON/OFF
深度休眠模式（DeepSleep mode）	停止	OFF	ON

3.3.1 进入休眠模式或深度休眠模式

使用 M0+ 内核的 ARM 等待中断专用指令，WFI（Wait for Interrupt），配合 M0+ 内核的系统控制寄存器（SCR，System Control Register）的 SLEEPONEXIT 和 SLEEPDEEP 位域，可实现立即进入或退出（中断服务程序）时进入休眠模式或深度休眠模式。

- 立即进入

  执行 WFI 指令，MCU 将立即进入休眠模式（SLEEPDEEP 为 0 时）或深度休眠模式（SLEEPDEEP 为 1 时）。

- 退出时进入

  将 SLEEPONEXIT 位置 1，当退出最低优先级的中断服务程序后，MCU 会进入休眠模式（SLEEPDEEP 为 0 时）或深度休眠模式（SLEEPDEEP 为 1 时），而不需执行 WFI 指令 。

表 3-2 进入休眠模式或深度休眠模式

进入方式	SLEEPONEXIT 位	进入条件	SLEEPDEEP 位	进入模式
立即进入	-	执行 WFI 指令	0	休眠模式
			1	深度休眠模式
退出时进入	1	退出最低优先级的中断服务程序后	0	休眠模式
			1	深度休眠模式

注意：

在进入深度休眠模式之前，若 FLASH 正在进行擦写操作，则必须等待 FLASH_ISR.BUSY 标志位清 0，同时须确保 FLASH_CR1.MODE 为 0。

在深度休眠模式下，系统将自动关闭高速时钟。如用户需要在深度休眠模式下使部分外设仍保持运行，则须在进入深度休眠模式前，启动相应的低速时钟并将该外设时钟设置为此低速时钟。

3.3.2 退出休眠模式或深度休眠模式

在休眠模式或深度休眠模式下，均可通过中断来唤醒 CPU，返回到运行模式。但是，值得注意的是，如果用户在中断服务程序中执行 WFI 命令进入休眠（包括深度休眠），则需要比此中断更高优先级的中断才能唤醒 CPU，因此，我们强烈建议用户在准备进入休眠前，应先处理完所有中断服务程序，并且清除所有中断请求和中断标志。

不同工作模式下，CPU 可响应的中断类型，如下表所示：

表 3-3 工作模式与中断源

中断号	中断源	运行模式	休眠模式	深度休眠模式
0	IWDT	Y	Y	Y
1	LVD	Y	Y	Y
2	RTC	Y	Y	Y
3	FLASH	Y	Y	N
	RAM	Y	Y	N
4	RCC	Y	Y	Y
5	GPIOA	Y	Y	Y
6	GPIOB	Y	Y	Y
7	GPIOC	Y	Y	Y
12	ADC	Y	Y	N
13	ATIM	Y	Y	N
14	VC1	Y	Y	Y
15	VC2	Y	Y	Y
16	GTIM1	Y	Y	N
17	GTIM2	Y	Y	N
19	LPTIM	Y	Y	Y
20	BTIM1	Y	Y	N
21	BTIM2	Y	Y	N
22	BTIM3	Y	Y	N
23	I2C	Y	Y	N
25	SPI	Y	Y	N
27	UART1	Y	Y	Y
28	UART2	Y	Y	Y
29	UART3	Y	Y	Y
31	FAULT	Y	Y	Y

使用中断退出休眠模式，用户必须在进入休眠（包括深度休眠）前使能此中断的允许位。

中断唤醒退出休眠模式后，CPU 将立即进入此中断的中断服务程序。如果用户未设置此中断服务程序，且设置为立即进入休眠时：CPU 将继续执行进入休眠的 WFI 指令的下一条语句；而设置为退出时进入休眠时：继续执行最后进入的中断服务程序的下一条语句。一般情况下，基于系统可靠性考虑，强烈建议用户设置此中断的服务程序，并在中断服务程序中清除中断请求和中断标志。

中断唤醒退出深度休眠模式时，CPU 运行状态与退出休眠模式相同。

深度休眠模式下系统将自动关闭高速时钟，在退出深度休眠时，CW32L011 为用户额外增加了一种系统时钟选择，用户既可以选择继续使用进入深度休眠时使用的时钟，也可选择 HSI 4MHz 作为系统时钟。配置系统控制寄存器 SYSCTRL_CR2 的 WAKEUPCLK 位域为 1，则在中断唤醒退出深度休眠模式后自动使用内部高速时钟 HSI 4MHz 作为系统时钟，加速系统唤醒。

3.3.3 工作模式与复位源

即使在休眠模式或深度休眠模式，CPU 亦可响应部分复位源。不同工作模式下 CPU 可响应的硬件复位或软件复位，如下表所示：

表 3-4 工作模式与复位源

复位源	运行模式	休眠模式	深度休眠模式
上电复位 / 掉电复位（POR/BOR）	Y	Y	Y
引脚输入复位 (NRST)	Y	Y	Y
LVD 复位	Y	Y	Y
IWDT 复位	Y	Y	Y
内核 LOCKUP 故障复位	Y	N	N
内核 SYSRESETREQ 复位	Y	N	N

3.4 低功耗应用

休眠模式下，CPU 停止运行，所有外设保持运行，包括 ARM® Cortex®-M0+ 内核外设，比如 NVIC、SysTick 等外设。休眠模式的功耗低于运行模式。

深度休眠模式下，CPU 停止运行，高速时钟关闭，低速时钟保持状态不变，部分外设可以配置为继续运行，NVIC 中断处理仍然工作。深度休眠模式的功耗远小于休眠模式。

用户可以通过以下方式降低系统运行功耗：

1. 降低系统时钟频率

- 使用低频率的高速时钟 HSI、HSE 或低速时钟 LSI、LSE。
- 通过编程预分频寄存器降低 SYSCLK、HCLK、PCLK 的频率：
  - 设置 SYSCTRL_CR0 寄存器的 SYSCLK 位域，选择适当的时钟源。
  - 设置 SYSCTRL_CR0 寄存器的 HCLKPRS 位域，降低 HCLK 频率。
  - 设置 SYSCTRL_CR0 寄存器的 PCLKPRS 位域，降低 PCLK 频率。

2. 关闭休眠期间不使用的时钟和外设

- 可以根据需要关闭 AHB 总线时钟 HCLK 和 APB 总线时钟 PCLK。
- 关闭与唤醒无关的外设的时钟：
  - AHB 外设时钟使能控制寄存器，SYSCTRL_AHBEN。
  - APB 外设时钟使能控制寄存器 1，SYSCTRL_APBEN1。
  - APB 外设时钟使能控制寄存器 2，SYSCTRL_APBEN2。

3.5 Cortex®-M0+ 内核系统控制寄存器 (SCB->SCR)

Address: 0xE000 ED10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:5	RFU	-	保留位，请保持默认值
4	SEVONPEND	RW	设置为 1 时，中断的每次新的挂起都会产生一个事件，如果使用了 WFE 休眠，它可用于唤醒处理器。
3	RFU	-	保留位，请保持默认值
2	SLEEPDEEP	RW	设置为 1 时，执行 WFI 或 SLEEPONEXIT 为 1 且退出所有中断服务程序时进入深度休眠模式（DeepSleep mode）； 设置为 0 时，执行 WFI 或 SLEEPONEXIT 为 1 且退出所有中断服务程序时进入休眠模式（Sleep mode）
1	SLEEPONEXIT	RW	设置为 1 时，当退出所有中断服务程序时，处理器自动进入休眠模式 (或深度休眠模式)； 设置为 0 时，退出时进入休眠功能被禁止
0	RFU	-	保留位，请保持默认值

4 复位和时钟（RCC）

4.1 系统复位

CW32L011 支持以下 6 种系统复位：

- 上电复位 / 掉电复位（POR/BOR）
- 引脚输入复位（NRST）
- IWDT 复位
- LVD 复位
- 内核 SYSRESETREQ 复位
- 内核 LOCKUP 故障复位

每种系统复位的复位范围如下表所示：

表 4-1 复位方式及范围

复位方式	复位范围
上电复位 / 掉电复位（POR/BOR）	整个 MCU
引脚输入复位（NRST）	整个 MCU（除 RTC 外）
IWDT 复位	M0+ 内核 / 外设（除 RAM 控制器 /RTC 外）
LVD 复位	M0+ 内核 / 外设（除 LVD 控制部分 /RTC 外）
内核 SYSRESETREQ 复位	M0+ 内核（除 SWD 调试逻辑外）/ 外设（除 RAM 控制器 /LVD/RTC 外）
内核 LOCKUP 故障复位	M0+ 内核 / 外设（除 RAM 控制器 /LVD/RTC 外）

发生系统复位后，CPU 重新运行，大部分寄存器都被复位到默认值，程序从中断向量表的复位中断入口地址开始执行。

用户可通过系统复位标志寄存器 SYSCTRL_RESETFLAG 来查询本次系统复位的复位源。复位标志由硬件置位，软件清零。建议用户在读取标志后清除该寄存器相关标志位使标志位为 0，以避免在下次复位后发生混淆。

4.1.1 上电复位（POR）/ 掉电复位（BOR）

CW32L011 集成了专门的 POR 和 BOR 电路对电源电压进行监控，在电源电压低于安全范围时将芯片保持在复位状态，防止芯片在上电 / 掉电过程中误动作。为保证系统工作稳定，用户须保持电源电压在安全范围内。

POR 和 BOR 的具体功能请参见 3 电源控制（PWR）与功耗章节。

4.1.2 引脚输入复位（NRST）

CW32L011 具有专门的复位输入引脚，输入一定宽度的低电平信号会引起系统复位。芯片内部设计有专用防抖电路，短于 20μs 的低电平脉冲信号会被屏蔽。

复位输入引脚内置有上拉电阻，用户如需外接 RC 电路，须考虑内部上拉电阻的影响。内置上拉电阻的电路特性请参阅数据手册相关章节。

4.1.3 IWDT 复位

CW32L011 集成独立看门狗 (IWDT)，当 IWDT 满足复位条件时，会产生复位信号引起系统复位。

IWDT 复位详细功能请参见 15 独立看门狗定时器（IWDT）章节。

4.1.4 LVD 低电压检测复位

CW32L011 内部集成低电压检测器 (LVD)，可将选定的监测电压和设定的 LVD 门限电压持续比较，当比较结果满足触发条件时，将产生 LVD 复位信号引起系统复位。

与 POR/BOR 复位方式相比，LVD 低电压检测复位的功能更强大，复位门限电压、监测电压源、脉冲滤波宽度和迟滞时间都可通过相关寄存器进行设置。

LVD 详细信息请参见 22 低电压检测器（LVD）章节。

4.1.5 内核 SYSRESETREQ 复位

内核 SYSRESETREQ 复位是软件复位，通过设置 ARM® Cortex®-M0+ 的应用中断和控制状态寄存器（AIRCR，Application Interrupt and Reset Control Register）的 SYSRESETREQ 位域来实现。应用程序设置该位为 1 则会产生内核 SYSRESETREQ 复位，从而实现软件复位。

4.1.6 内核 LOCKUP 故障复位

当 CPU 遇到严重异常（如读取到的指令无效、访问 FLASH 时位宽和目标地址不匹配），会将 PC 指针停在当前地址处锁定，并产生内核 LOCKUP 故障复位信号。

芯片上电后，LOCKUP 复位功能默认处于不使能状态，用户需要手动使能，通过设置系统控制寄存器 SYSCTRL_CR2 的 LOCKUP 位域为 1，即可使能 LOCKUP 复位功能。

4.2 外设复位

用户可使用软件将 CW32L011 内部的各种外设单独复位。外设复位可以让各外设的寄存器、状态机以及各种控制逻辑等，恢复到上电复位后的默认状态。用户通过设置 SYSCTRL_AHBRST、SYSCTRL_APBRST1、SYSCTRL_APBRST2 这 3 个寄存器来进行各外设模块的独立复位操作。

对于上电复位 / 掉电复位（POR/BOR），内部各外设模块已处于复位后的默认状态，可直接进行模块初始化配置，不需要再单独对各模块进行独立复位。

对于其它类型系统复位情形，部分外设可能保留复位前的工作状态。用户如需要将外设恢复到上电后的默认状态，应通过对应的复位寄存器执行外设复位操作后再使用。RTC 模块是典型的可能需要在复位后保留当前状态的外设，具体判断逻辑示例请参阅相关例程。

4.3 时钟及控制

4.3.1 概述

CW32L011 内置多路时钟产生电路，通过分频器和多路选择器产生各种不同频率的时钟给 CPU 及各外设使用。系统时钟树结构示意图如图 4-1 系统内部时钟树所示。

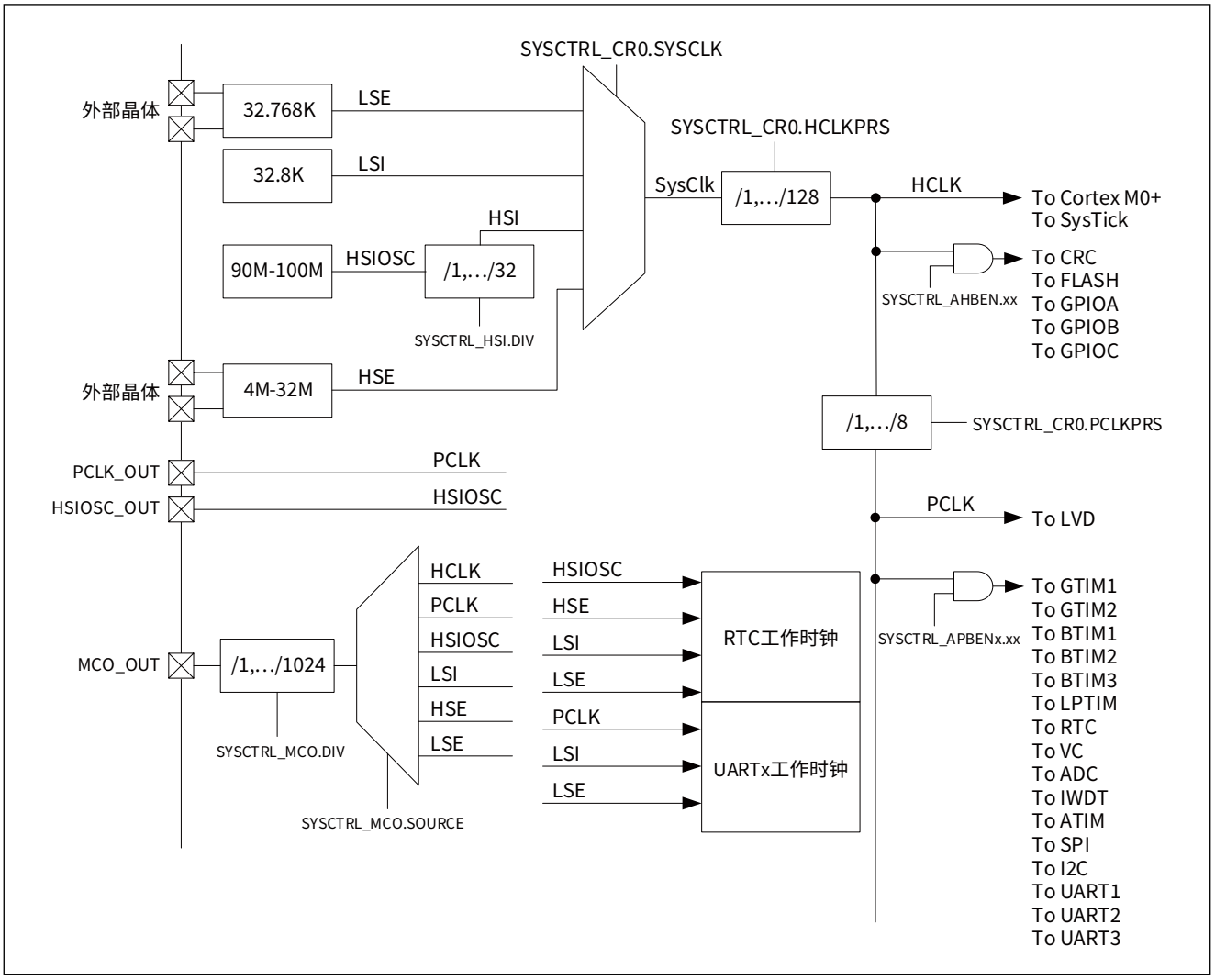
系统内部时钟 SysClk 经过分频为 CPU 内核提供高级高性能总线时钟 HCLK，HCLK 时钟经过分频为数字及模拟外设提供高级外设时钟 PCLK。

系统时钟 SYSCLK 有 4 个时钟源：

- 外部高速振荡器时钟（HSE）
- 外部低速振荡器时钟（LSE）
- HSI 时钟，由内部高速 RC 振荡器时钟（HSIOSC）经过分频产生
- 内部低速 RC 振荡器时钟（LSI）

系统内部时钟树如下图所示：

图 4-1 系统内部时钟树



内部高速 RC 振荡器时钟 HSIOSC 经过分频器分频后产生 HSI 时钟，分频系数通过内置高频时钟控制寄存器 SYSCTRL_HSI 的 DIV 位域进行设置，有效分频系数为 1、2、3、4、5、6、7、8、9、10、12、16、20、24、28、32。

系统时钟 SysClk 可选 4 个时钟源：HSE、LSE、HSI、LSI，通过系统控制寄存器 SYSCTRL_CR0 的 SYSCLK 位域进行选择。

高级高性能总线时钟 HCLK，由系统时钟 SysClk 经过分频产生，作为 M0+ 内核、SysTick、FLASH、CRC、GPIO 等模块的配置时钟及工作时钟。分频系数通过系统控制寄存器 SYSCTRL_CR0 的 HCLKPRS 位域设置，有效分频系数为 2^n (n = 0~7)。

高级外设时钟 PCLK，由 HCLK 经过分频产生，作为定时器、SPI、I2C 等外设的配置时钟及工作时钟。分频系数通过系统控制寄存器 SYSCTRL_CR0 的 PCLKPRS 位域设置，有效分频系数为 2^n (n = 0~3)。

4.3.2 系统时钟与工作模式

CW32L011 支持三种工作模式：运行模式（Active mode），休眠模式（Sleep mode）和深度休眠模式（DeepSleep mode）。

运行模式下 CPU 正常运行，所有模块用户均可正常使用。

休眠模式下，CPU 停止运行，各时钟振荡器及外设保持原状态不变。

深度休眠模式下，CPU 停止运行，高速时钟 HSE、HSIOSC 的振荡器被自动关闭以节省功耗；低速时钟 LSE、LSI 的振荡器保持原状态不变；系统时钟 SysClk 及 HCLK、PCLK 时钟是否有效，取决于 SysClk 系统时钟的时钟源状态。

从深度休眠模式唤醒后，如果系统控制寄存器 SYSCTRL_CR2 的 WAKEUPCLK 位域配置为 1，则系统会自动使用 HSI 4MHz 作为系统时钟的时钟源，原系统时钟源保持使能。如果 SYSCTRL_CR2.WAKEUPCLK 为 0，则系统会等待系统进入深度休眠前所使用的系统时钟源稳定后才开始运行。由于 HSI 启动速度很快，可快速响应用户需求，因此建议在进入深度休眠模式前切换系统时钟的时钟源为 HSI 时钟或者配置 SYSCTRL_CR2.WAKEUPCLK 为 1。

4.3.3 HSE 时钟

外部高速时钟（HSE）支持两种工作模式：

- 石英晶体 / 陶瓷谐振器，设置外置高频晶体控制寄存器 SYSCTRL_HSE 的 MODE 位域为 0，同时须配置 OSC_IN、OSC_OUT 引脚为模拟功能。
- 外部时钟输入，设置外置高频晶体控制寄存器 SYSCTRL_HSE 的 MODE 位域为 1，同时须配置 OSC_IN 引脚为数字输入功能。

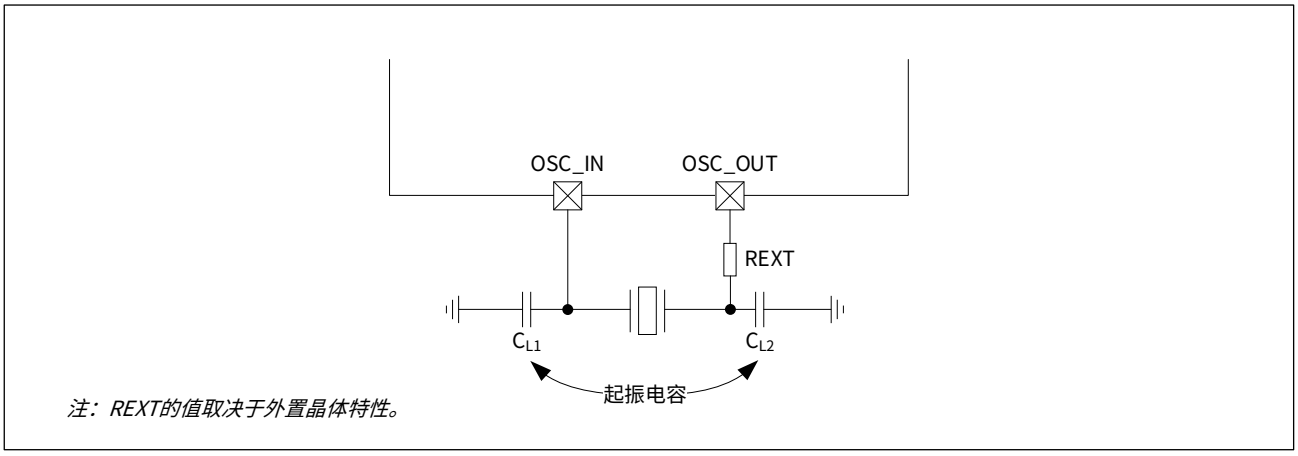
表 4-2 HSE 工作模式配置

工作模式	SYSCTRL_HSE.MODE	OSC_IN	OSC_OUT
石英晶体 / 陶瓷谐振器	0	模拟	模拟
外部时钟输入	1	数字输入	可作为通用 GPIO

石英晶体 / 陶瓷谐振器时钟模式

外置石英晶体 / 陶瓷谐振器接在 OSC_IN 和 OSC_OUT 两个引脚之间，配合起振电容，产生稳定的 4 ~ 32MHz 时钟信号，电路连接如下图所示：

图 4-2 HSE 外接石英晶体 / 陶瓷谐振器连接方式



石英晶体 / 陶瓷谐振器和起振电容须尽可能靠近芯片振荡器引脚放置，使得振荡器输出失真和启动时间达到最小。

起振电容取值大小须根据所选择的石英晶体 / 陶瓷谐振器进行调整，计算方法为：

$$C_L = (C_{L1} \times C_{L2}) / (C_{L1} + C_{L2}) + C_0$$

其中，

C_L 为晶体标称负载电容值；

C_0 为杂散电容值，跟电路板设计相关，一般可取值 4pF ~ 6pF；

C_{L1} 和 C_{L2} 为外接的 2 个起振电容，通常 C_{L1} 和 C_{L2} 相同。

如用户采用标称负载电容值为 12.5pF 的晶体谐振器，则晶体外接的 2 个起振电容常见取值为 15pF。

注：

由于晶体厂家众多，各厂家晶体特性各不相同，如果 HSE 振荡器输出时钟信号不理想，可联系晶体供货厂家协调处理。

在 HSE 时钟稳定前，用户可以通过设置外置高频晶体控制寄存器 SYSCTRL_HSE 的 PDRIVER 位域进行驱动能力调节，有 8 档驱动能力可选择，驱动能力越强功耗越大；在 HSE 时钟稳定后，用户可以通过设置外置高频晶体控制寄存器 SYSCTRL_HSE 的 DRIVER 位域进行驱动能力调节，有 8 档驱动能力可选择，驱动能力越强功耗越大。用户可根据需要在时钟可靠性、启动时间以及低功率消耗三者之间取得平衡。

外部时钟输入模式

外部时钟输入模式下，外部时钟从 OSC_IN 引脚输入，OSC_OUT 引脚可以作为通用 GPIO 使用。输入的时钟信号可以是方波、正弦波或者三角波，占空比必须在 40% ~ 60% 之间，频率在 4 ~ 32MHz 之间。

当使用外部有源晶振时，PA01 或 PA11 引脚（需配置为 HEXEN 复用输出功能）可输出使能信号，在芯片运行模式和深度休眠模式下分别输出不同的电平，输出极性通过 SYSCTRL_HSE 寄存器的 HEXENPOL 位域进行配置。

HSE 两种工作模式下，HSE 时钟振荡器上电后均默认处于关闭状态，通过设置系统控制寄存器 SYSCTRL_CR1 的 HSEEN 位域为 1 启动。

HSE 振荡器启动后，当芯片内部时钟监控模块检测到一定数量的 HSE 时钟信号，则认为 HSE 时钟已稳定。检测时钟数量可通过外置高频晶体控制寄存器 SYSCTRL_HSE 的 WAITCYCLE 位域进行设置，如下表所示：

表 4-3 HSE 稳定检测时钟信号数量

SYSCTRL_HSE.WAITCYCLE	HSE 检测时钟数量
00	8192
01	32768
10	131072（默认值）
11	262144

通过外置高频晶体控制寄存器 SYSCTRL_HSE 的 STABLE 标志位，可确定 HSE 时钟的起振状态，STABLE 标志为 1 则 HSE 时钟已稳定，为 0 则表示 HSE 时钟还未稳定。

注意：

HSE 振荡器应在启动前设置好所有参数，启动后禁止修改相关参数。

4.3.4 HSIOSC 时钟

HSIOSC 时钟由内部 RC 振荡器产生，不需要外部电路，启动速度快，频率固定为 96MHz。

RC 振荡器输出时钟的频率受芯片加工过程、工作电压、环境温度等因素影响，CW32L011 提供了 HSIOSC 时钟频率校准功能，用户可通过设置内置高频时钟控制寄存器 SYSCTRL_HSI 的 TRIM 位域值来校准 HSIOSC 时钟频率，详情请参见 4.4.2 时钟校准。

HSIOSC 内部高速 RC 振荡器在芯片上电后，默认处于开启状态，用户可通过设置系统控制寄存器的 SYSCTRL_CR1 的 HSIEN 位域为 0 来关闭。如用户停止并重新启动了 HSIOSC 振荡器，可通过内置高频时钟控制寄存器 SYSCTRL_HSI 的 STABLE 标志位来确定 HSI 时钟是否稳定，STABLE 标志为 1 表示 HSIOSC 时钟已稳定，为 0 则表示 HSIOSC 时钟还未稳定。

注意：

HSIOSC 振荡器应在启动前设置好所有参数，启动后禁止修改相关参数。

HSIOSC 时钟经过分频后输出 HSI 时钟，分频系数通过内置高频时钟控制寄存器 SYSCTRL_HSI 的 DIV 位域设置，有效分频系数为 1、2、3、4、5、6、7、8、9、10、12、16、20、24、28、32，上电后默认值为 24，所以 HSI 时钟默认频率为 4MHz。

4.3.5 LSE 时钟

外部低速时钟（LSE）支持两种工作模式：

- 石英晶体 / 陶瓷谐振器，设置外置低频晶体控制寄存器 SYSCTRL_LSE 的 MODE 位域为 0，同时配置 OSC32_IN、OSC32_OUT 引脚为模拟功能。
- 外部时钟输入，设置外置低频晶体控制寄存器 SYSCTRL_LSE 的 MODE 位域为 1，同时配置 OSC32_IN 引脚为数字输入功能。

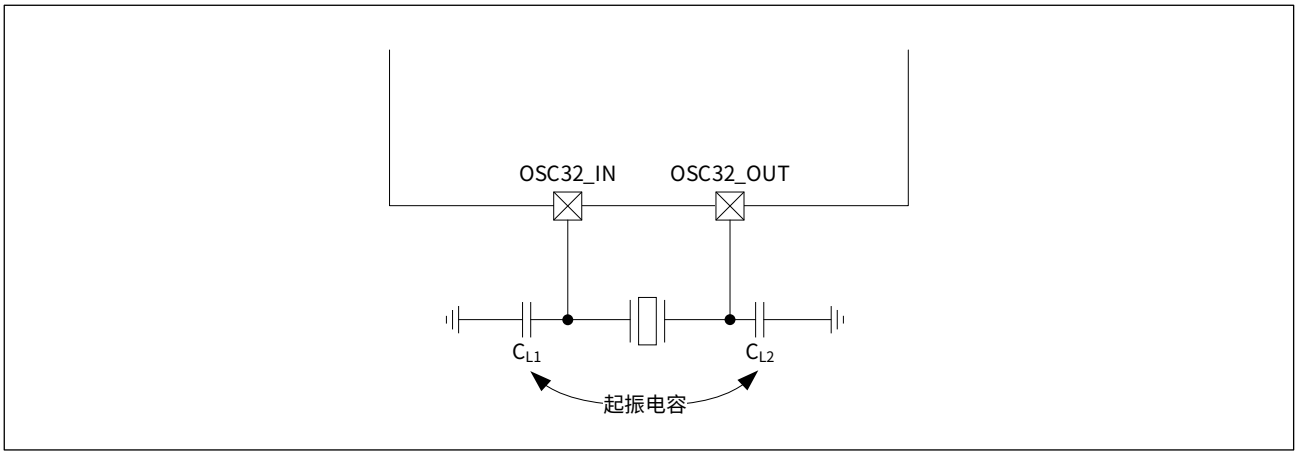
表 4-4 LSE 工作模式配置

工作模式	SYSCTRL_LSE.MODE	OSC32_IN	OSC32_OUT
石英晶体 / 陶瓷谐振器	0	模拟	模拟
外部时钟输入	1	数字输入	可作为通用 GPIO

石英晶体 / 陶瓷谐振器时钟模式

外部低频石英晶体 / 陶瓷谐振器接在 OSC32_IN 和 OSC32_OUT 两个引脚之间，配合起振电容，可产生最高 1MHz 的稳定时钟信号。LSE 通常作为系统实时时钟 RTC 的工作时钟，此时应选用频率为 32.768kHz 的石英晶体 / 陶瓷谐振器。外部电路连接如下图所示：

图 4-3 LSE 外接石英晶体 / 陶瓷谐振器连接方式



石英晶体 / 陶瓷谐振器和起振电容须尽可能靠近芯片振荡器引脚放置，使得振荡器输出失真和启动时间达到最小。

起振电容取值大小须根据所选择的石英晶体 / 陶瓷谐振器进行调整，计算方法为：

$$C_L = (C_{L1} \times C_{L2}) / (C_{L1} + C_{L2}) + C_0$$

其中，

C_L 为晶体标称负载电容值；

C_0 为杂散电容值，跟电路板设计相关，一般可取值 4pF ~ 6pF；

C_{L1} 和 C_{L2} 为外接的 2 个起振电容，通常 C_{L1} 和 C_{L2} 相同。

如用户采用标称负载电容值为 12.5pF 的晶体谐振器，则晶体外接的 2 个起振电容常见取值为 15pF。

注：

由于晶体厂家众多，各厂家晶体特性各不相同，如果 LSE 振荡器输出时钟信号不理想，可联系晶体供货厂家协调处理。

在 LSE 时钟稳定前，用户可以通过设置外置低频晶体控制寄存器 SYSCTRL_LSE 的 PDRIVER 位域进行驱动能力调节，有 16 档驱动能力可选择，驱动能力越强功耗越大。在 LSE 时钟稳定后，用户可以通过设置外置低频晶体控制寄存器 SYSCTRL_LSE 的 DRIVER 位域进行驱动能力调节，有 16 档驱动能力可选择，驱动能力越强功耗越大。用户可根据需要在时钟可靠性、启动时间以及低功率消耗三者之间取得平衡。

外部时钟输入模式

外部时钟输入模式下，外部时钟从 OSC32_IN 引脚输入，OSC32_OUT 引脚可以作为通用 GPIO 使用。输入的时钟信号可以是方波、正弦波或者三角波，占空比必须在 45% ~ 55% 之间，频率最高为 1MHz。

LSE 两种工作模式下，LSE 时钟振荡器上电后均默认处于关闭状态，通过设置系统控制寄存器 SYSCTRL_CR1 的 LSEEN 位域为 1 来启动。

LSE 振荡器启动后，当芯片内部时钟监控模块检测到一定数量的 LSE 时钟信号，则认为 LSE 时钟已稳定。检测时钟数量可通过外置低频晶体控制寄存器 SYSCTRL_LSE 的 WAITCYCLE 位域进行设置，如下表所示：

表 4-5　LSE 稳定检测时钟信号数量

SYSCTRL_LSE.WAITCYCLE	LSE 检测时钟数量
00	256
01	1024
10	4096（默认值）
11	16384

通过外置低频晶体控制寄存器 SYSCTRL_LSE 的 STABLE 标志位，可确定 LSE 时钟的起振状态，STABLE 标志为 1 表示 LSE 时钟已稳定，为 0 则表示 LSE 时钟还未稳定。

注意：

LSE 振荡器应在启动前设置好所有参数，启动后禁止修改相关参数。

LSE 振荡器具有锁定功能，锁定功能开启后将禁止软件对 LSE 时钟振荡器的使能位（SYSCTRL_CR1.LSEEN）进行清零操作。即，LSE 一旦开启，不能由软件关闭。锁定功能主要为满足部分特殊应用场景需求而设计，例如，对 RTC 时间丢失敏感的应用场景，可有效防止误操作导致的 RTC 时间丢失。该锁定功能默认关闭，通过设置系统控制寄存器 SYSCTRL_CR1 的 LSELOCK 位域为 1 启动。

4.3.6 LSI 时钟

LSI 时钟由内部低速 RC 振荡器产生，默认频率为 32.8kHz。内部低速 RC 振荡器不需要外部电路，比 LSE 时钟的成本低，但精度低于 LSE 时钟。

RC 振荡器输出时钟的频率受芯片加工过程、工作电压、环境温度等因素影响，CW32L011 提供了 LSI 时钟频率校准功能。用户可通过设置内置低频时钟控制寄存器 SYSCTRL_LSI 的 TRIM 位域值来校准 LSI 时钟频率，详情请参见 4.4.2 时钟校准。

LSI 内部低速 RC 振荡器默认处于关闭状态，通过设置系统控制寄存器 SYSCTRL_CR1 的 LSIEN 位域为 1 启动。LSI 振荡器启动后，芯片内部时钟监控模块检测到一定数量的 LSI 时钟信号，则认为 LSI 时钟已稳定。检测时钟数量可通过内置低频振荡器控制寄存器 SYSCTRL_LSI 的 WAITCYCLE 位域进行设置，如下表所示：

表 4-6 LSI 稳定检测时钟信号数量

SYSCTRL_LSI.WAITCYCLE	LSI 检测时钟数量
00	6（默认值）
01	18
10	66
11	258

通过内置低频时钟控制寄存器 SYSCTRL_LSI 的 STABLE 标志位，可确定 LSI 时钟是否稳定，STABLE 标志为 1 表示 LSI 时钟已稳定，为 0 则表示 LSI 时钟还未稳定。

注意：

LSI 振荡器在启动前必须设置好所有参数，启动后禁止修改相关参数。

4.3.7 SysClk 系统时钟

系统时钟 SysClk 可选 4 种时钟源，包括 HSE、LSE、HSI、LSI。

系统上电复位完成后默认选择 HSI 作为 SysClk 的时钟源，时钟频率默认值是 4MHz。

用户可通过系统控制寄存器 SYSCTRL_CR0 的 SYSCLK 位域选择系统时钟源，当选择某一时钟作为系统时钟源后，用户无法通过设置该时钟电路的使能位 SYSCTRL_CR1.xxxEN 为 0 来停止该时钟。

4.3.8 片内外设时钟控制

片内外设一般都需要有配置时钟和工作时钟，配置时钟用来响应 CPU 对外设寄存器的读写操作，工作时钟用于各外设的功能实现（如 UART 的传输时钟，定时器的计数时钟等）。

在使用外设前都必须打开外设的配置时钟和工作时钟，否则外设无法工作。通过设置 AHB 外设时钟使能控制寄存器 SYSCTRL_AHBEN、APB 外设时钟使能控制寄存器 SYSCTRL_APBEN1 和 SYSCTRL_APBEN2 的对应位为 1，打开对应外设的配置时钟和工作时钟。

RTC、UART、FLASH 等外设只打开配置时钟，工作时钟通过各模块的时钟源选择寄存器进行配置。

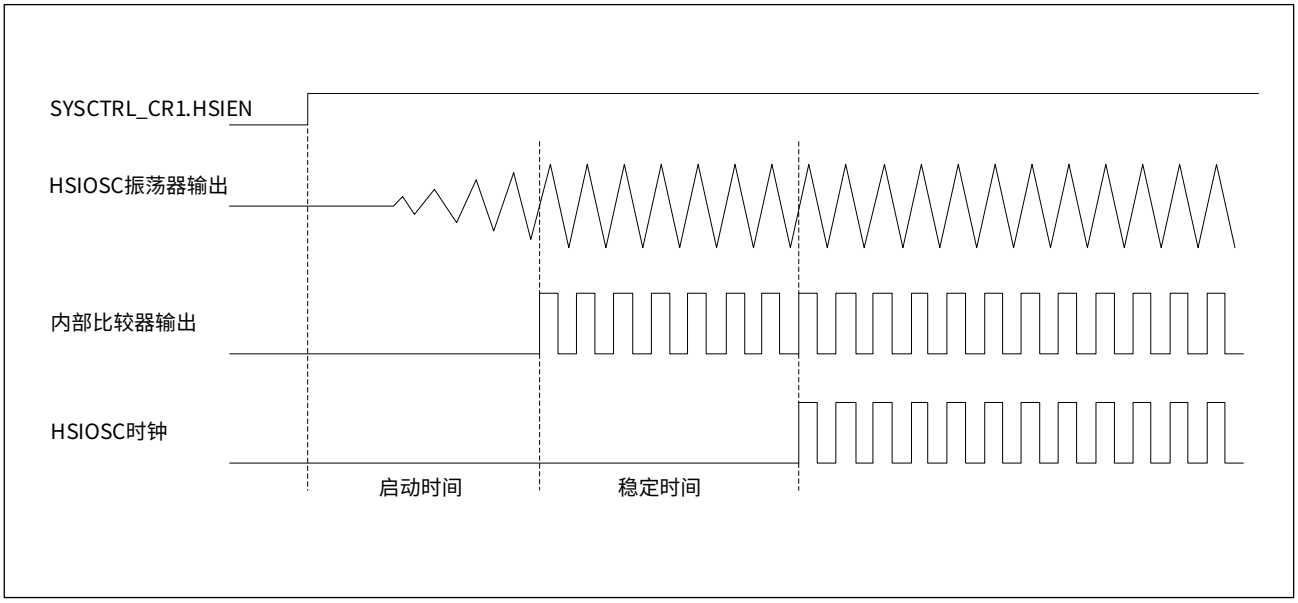
当外设不需要使用时，通过关闭外设的配置时钟和工作时钟禁止外设，能有效降低芯片功耗。

4.4 时钟启动、校准与状态检测

4.4.1 时钟启动

CW32L011 的时钟源启动过程类似，以 HSIOSC 为例说明时钟启动后的稳定过程，如下图所示：

图 4-4 HSIOSC 时钟启动过程



当设置 SYSCTRL_CR1.HSIEN 为 1 后，HSIOSC 时钟振荡电路开始工作，但此时输出的时钟信号振幅很小。经过启动时间阶段后，输出时钟信号的振幅、占空比等可满足内部采样电路需求，进入稳定时间阶段。在稳定时间阶段，芯片内部时钟监控电路对 HSIOSC 输出的时钟信号进行计数，当计数值达到设定的个数后，认为 HSIOSC 时钟信号已稳定，HSIOSC 时钟稳定标志位 SYSCTRL_HSI.STABLE 被置 1。

其它时钟振荡器的时钟启动过程类似，但注意各时钟振荡器在稳定阶段检测时钟数量以及起振失败检测的检测时间和检测时钟数量等都不相同，具体参见各时钟振荡器相关描述。

4.4.2 时钟校准

时钟校准主要针对 HSIOSC 时钟和 LSI 时钟。通过调整振荡器的 TRIM 值来实现时钟频率校准。

HSIOSC 时钟校准

HSIOSC 安全工作范围为 90 ~ 100MHz，超过安全范围，芯片可能出现异常。芯片出厂时已预调好 96MHz 频点的校准参数，并存放在 FLASH 中。应用程序只需要将 FLASH 内的校准值读出并写入 SYSCTRL_HSI.TRIM 即可获得精准的 96MHz 时钟。96MHz 频率校准值存放地址为：0x0010 07C0 – 0x0010 07C1。如需其它频率的时钟则需要用户自行调整 SYSCTRL_HSI.TRIM 的值。

LSI 时钟校准

LSI 输出时钟频率范围为 32.8kHz±10%，如果将 LSI 的输出时钟频率调节到此范围外，则该时钟有可能异常。芯片出厂时已预调好 32.8kHz 频点的校准参数，并存放在 FLASH 中，应用程序只需要将 FLASH 内的校准值读出并写入 SYSCTRL_LSI.TRIM 即可获得精准的 32.8kHz 时钟。32.8kHz 频率校准值存放地址：0x0010 07C2 - 0x0010 07C3。如需其它频率的时钟则需要用户自行调整 SYSCTRL_LSI.TRIM 的值。

4.4.3 时钟状态检测

芯片具备时钟启动过程中的时钟稳定检测、起振失败检测，以及时钟运行中失效检测功能，并支持在当前选定的系统时钟源故障后自动进行时钟源切换。

4.4.3.1 时钟稳定检测

HSE、LSE、HSIOSC、LSI 这 4 种时钟源都支持时钟稳定检测功能，用户可通过对应时钟源的稳定标志位来确定时钟状态。时钟稳定标志在关闭时钟源时由硬件清 0，在时钟源启动并稳定后由硬件置 1。

注意时钟稳定标志只针对时钟启动过程而言，在时钟稳定运行过程中，检测到时钟运行失效不会影响该时钟稳定标志。

以 HSIOSC 时钟源为例，针对 HSIOSC 时钟稳定标志和时钟稳定中断标志，说明如下：

- SYSCTRL_HSI.STABLE

  硬件在关闭 HSIOSC 时清零，硬件在检测到 HSIOSC 时钟稳定时置位。

- SYSCTRL_ISR.HSISTABLE

  和 SYSCTRL_HSI.STABLE 一样，属于同一个信号。该信号虽然位于中断标志寄存器 SYSCTRL_ISR 中，但被硬件置 1 后并不会引起中断请求。

- SYSCTRL_ISR.HSIRDY

  当硬件检测到时钟由不稳定状态变为稳定状态时（即 SYSCTRL_HSI.STABLE 标志位由 0 变为 1）置 1，为时钟稳定中断标志，用户可通过设置 SYSCTRL_ICR.HSIRDY 为 0 清除该标志位。

4.4.3.2 时钟起振失败检测

CW32L011 支持外部时钟（HSE 和 LSE）起振失败检测功能。启动外部时钟后，时钟检测逻辑对所检测的时钟信号进行计数，在检测时间内检测到设定个数的时钟信号则时钟起振成功，否则起振失败。

HSE 起振失败检测的时钟数量通过 SYSCTRL_HSE.WAITCYCLE 设置，检测时间无需用户设置，由系统自动设定，如下表所示：

表 4-7 HSE 起振失败检测时间及检测时钟数量

SYSCTRL_HSE.WAITCYCLE	检测时间	检测时钟数量
00	65ms	8192
01	65ms	32768
10	65ms	131072
11	130ms	262144

LSE 起振失败检测的时钟数量通过 SYSCTRL_LSE.WAITCYCLE 设置，检测时间无需用户设置，由系统自动设定，如下表所示：

表 4-8 LSE 起振失败检测时间及检测时钟数量

SYSCTRL_LSE.WAITCYCLE	检测时间	检测时钟数量
00	1s	256
01	1s	1024
10	1s	4096
11	2s	16384

外部时钟（HSE 和 LSE）故障检测功能默认处于关闭状态，此时 HSE/LSE 起振失败不会产生相应的中断标志。通过设置 SYSCTRL_CR1.HSECCS 为 1 和 SYSCTRL_CR1.LSECCS 为 1 分别使能外部时钟 HSE 和 LSE 的故障检测功能。

使能外部时钟的故障检测功能后，如果 HSE 或者 LSE 起振失败，会产生起振失败中断标志（对应 SYSCTRL_ISR.HSEFAIL 或者 SYSCTRL_ISR.LSEFAIL 被置为 1）；如果中断使能（对应 SYSCTRL_IER.HSEFAIL 或者 SYSCTRL_IER.LSEFAIL 设置为 1），则 CPU 会执行中断服务程序进行时钟起振失败处理。

产生起振失败中断标志 LSEFAIL 之后可以软件关闭 LSEEN，然后软件清除 LSEFAIL 标志。如果没有关闭 LSEEN，则每隔一段时间就会产生一次 LSEFAIL，间隔时间参见表 4-8 LSE 起振失败检测时间及检测时钟数量。

4.4.3.3 时钟运行中失效检测

CW32L011 支持外部时钟（HSE 和 LSE）运行中失效检测功能。在外部时钟稳定运行过程中，时钟检测逻辑持续以一定的检测周期对 HSE 和 LSE 时钟信号进行计数：在检测周期内检测到设定个数的时钟信号则运行正常，否则运行失效。

HSE 时钟的运行中失效检测周期通过 SYSCTRL_HSE.DETCNT 配置，实际时间为 SYSCTRL_HSE.DETCNT/f_{LSI}，检测时钟个数为 0x20000。考虑到时钟都有一定的偏差，为保证检测功能可靠，HSE 检测周期参数 HSE.DETCNT 的配置须留有一定的裕量，一般根据 HSE 的运行频率，配置为 8000/f_{HSE}（其中 f_{HSE} 为 HSE 时钟的频率，单位为 MHz）。

例：

如果 HSE 时钟频率为 4MHz，则 SYSCTRL_HSE.DETCNT 应配置为 2000。

LSE 时钟的运行中失效检测周期不可配置，固定为 256 个 LSI 时钟周期，检测时钟个数为 128。

外部时钟故障检测功能默认处于关闭状态，此时 HSE/LSE 运行中失效不会产生相应的中断标志。通过设置 SYSCTRL_CR1.HSECCS 为 1 和 SYSCTRL_CR1.LSECCS 为 1，分别使能 HSE 和 LSE 时钟故障检测功能。

使能外部时钟的故障检测功能后，如果运行过程中检测到 HSE 或者 LSE 失效，则会产生运行中失效标志（对应 SYSCTRL_ISR.HSEFAULT 或者 SYSCTRL_ISR.LSEFAULT 被置 1），如果中断使能（对应 SYSCTRL_IER.HSEFAULT 或者 SYSCTRL_IER.LSEFAULT 设置为 1），则 CPU 会执行中断服务程序进行时钟运行中失效处理。

产生运行中失效标志 LSEFAULT 之后建议软件关闭 LSEEN 以清除 LSESTABLE 状态，然后软件清除 LSEFAULT 标志。如果没有关闭 LSEEN，则每 256 个 LSI 时钟持续时间（约 7.8ms）都会产生一次 LSEFAULT，直到 LSE 恢复正常。

如果当前系统时钟来源为 HSE，设置 SYSCTRL_CR1.CLKCCS 为 0，HSE 运行中失效时无动作；设置 SYSCTRL_CR1.CLKCCS 为 1，HSE 运行中失效时系统时钟源会自动切换到 HSI4MHz 时钟。

如果当前系统时钟来源为 LSE，设置 SYSCTRL_CR1.CLKCCS 为 0，LSE 运行中失效时无动作；设置 SYSCTRL_CR1.CLKCCS 为 1，LSE 运行中失效时系统时钟源会自动切换到 HSI4MHz 时钟。

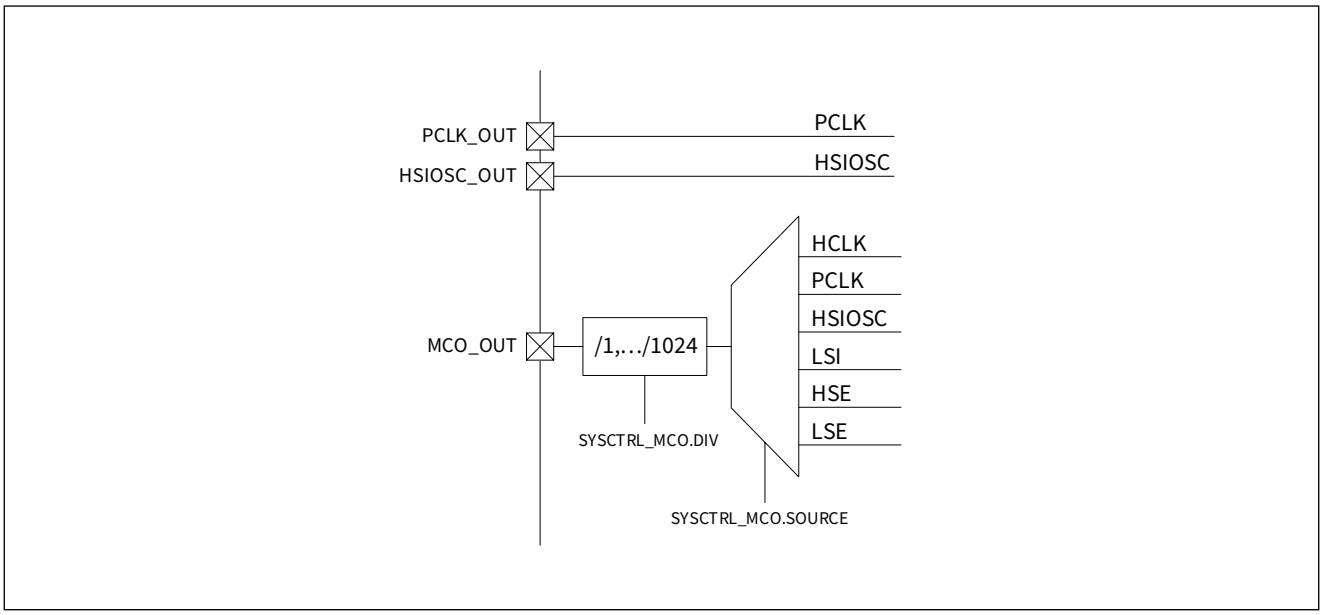
4.4.4 时钟验证与输出

CW32L011 支持将内部各种时钟信号输出到外部引脚。用户可利用该功能对当前系统的 CPU 运行频率、系统总线频率、外设工作频率等进行测量。

- PCLK_OUT 引脚：输出 PCLK 时钟信号。
- HSIOSC_OUT：输出 HSIOSC 时钟信号。
- MCO_OUT 引脚：输出 HCLK/PCLK/HSIOSC/LSI/HSE/LSE 时钟信号，时钟输出到 MCO_OUT 引脚前可通过预分频器进行分频（有效分频系数为 1、2、8、64、128、256、512、1024），以便低带宽仪表能准确测量信号。

时钟输出框图如下图所示：

图 4-5 时钟输出



4.5 SysClk 系统时钟切换

系统时钟 SysClk 可选择 4 种时钟源，包括 HSE、LSE、HSI、LSI，通过对系统控制寄存器 SYSCTRL_CR0 的 SYSCLK 位域进行设置，可在不同时钟源之间进行切换。

主要应用场景包括：

- 时钟故障自动切换

  当检测到当前系统时钟源失效时，可快速安全切换到其它可用时钟源上，提高产品可靠性。

- 功耗管理

  在待机状态切换 SysClk 的时钟源为低速时钟降低系统功耗，在正常使用状态切换 SysClk 的时钟源为高速时钟来快速响应用户使用需求。

时钟源安全切换规则如下：

- HSE、LSE、HSI、LSI 四个时钟源中的任意两个之间可以相互切换。

系统时钟切换操作必须按照下文所描述的时钟切换流程进行，否则可能出现异常。

注意：

系统时钟切换时需要同步配置 FLASH 控制寄存器 FLASH_CR2.WAIT 读等待周期参数：系统时钟频率不大于 24MHz 则应设置 FLASH 控制寄存器 FLASH_CR2.WAIT 为 0；系统时钟频率大于 24MHz 则应设置 FLASH 控制寄存器 FLASH_CR2.WAIT 为 1；系统时钟频率大于 48MHz 则应设置 FLASH 控制寄存器 FLASH_CR2.WAIT 为 2；系统时钟频率大于 72MHz 则应设置 FLASH 控制寄存器 FLASH_CR2.WAIT 为 3。

4.5.1 标准的时钟切换流程

标准时钟切换操作流程如下：

步骤 1：如果新时钟源为 HSE 或者 LSE，则需要配置对应的外部引脚为适当的模式：接外部晶体时需要配置为模拟功能，接外部时钟输入时需要将 GPIO 设置为数字输入功能并使能端口的输入功能；

步骤 2：配置新时钟源的振荡参数，如驱动能力、振荡幅度、稳定周期等；

步骤 3：设置新时钟源振荡器使能位为 1；

步骤 4：根据当前时钟源和新时钟源两者中较高的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 5：等待新时钟源输出稳定的频率（新时钟源的 STABLE 信号变为 1）；

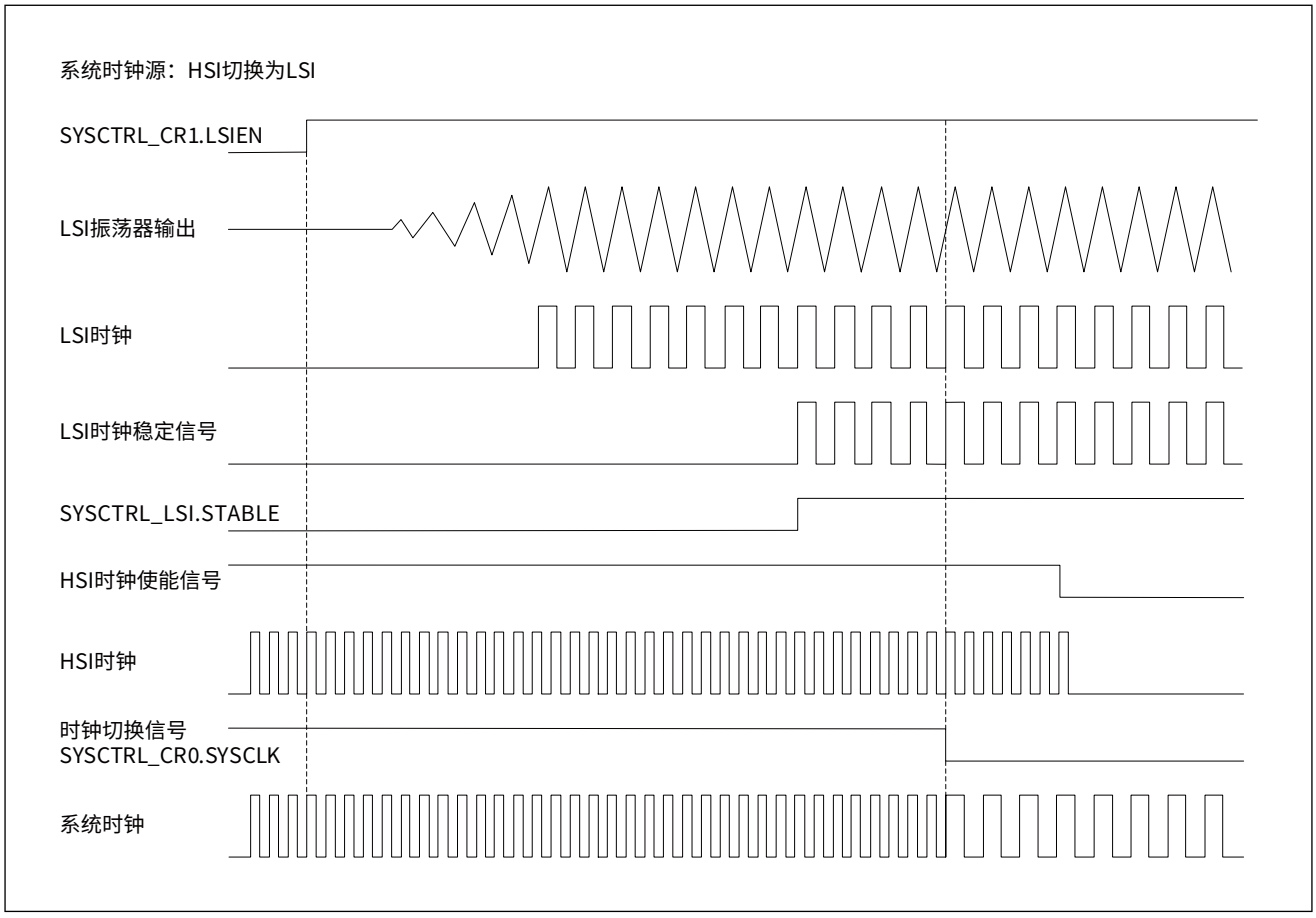
步骤 6：配置系统控制寄存器 SYSCTRL_CR0.SYSCLK，选择系统时钟 SysClk 的时钟源为新时钟源；

步骤 7：根据新时钟源的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 8：关闭不再使用的时钟源。

以系统时钟从 HSI 切换为 LSI 为例，新旧时钟的切换过程如下图所示：

图 4-6 HSI 时钟切换为 LSI 时钟



4.5.2 HSI 时钟不同频率间切换流程

HSI 时钟不同频率切换不需要启动新时钟源，只需要改变 HSI 时钟预分频器的分频比即可，切换速度快，切换前后无需等待时钟稳定时间。切换操作流程如下：

步骤 1：根据当前时钟源和新时钟源两者中较高的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 2：读取内置高频时钟控制寄存器 SYSCTRL_HSI 寄存器的值为 Value；

步骤 3：根据需要的目标频率，设置内置高频时钟控制寄存器 SYSCTRL_HSI.DIV 为对应的分频系数（注意 SYSCTRL_HSI.TRIM 域要保持不变）；

步骤 4：根据新时钟源的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数。

4.5.3 从其它时钟切换到 LSE 示例

切换操作流程如下：

步骤 1：设置 PC14/PC15 引脚为模拟功能端口；

步骤 2：根据 LSE 振荡器外接晶体特性，配置新时钟源 LSE 的振荡参数，如驱动能力（通过外置低频晶体控制寄存器 SYSCTRL_LSE.DRIVER 设置）、稳定周期 (通过外置低频晶体控制寄存器 SYSCTRL_LSE.WAITCYCLE 设置，建议设置为 3) 等；

步骤 3：设置系统控制寄存器 SYSCTRL_CR1.LSEEN 为 1，使能 LSE 晶体振荡电路；

注意：

SYSCTRL_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 4：循环查询并等待外置低频晶体控制寄存器 SYSCTRL_LSE.STABLE 稳定标志变为 1，即等待 LSE 晶体振荡电路输出稳定时钟；

步骤 5：设置系统控制寄存器 SYSCTRL_CR0.SYSCLK 为 4，将 SysClk 时钟来源切换为 LSE；

注意：

SYSCTRL_CR0 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 6：按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 7：设置系统控制寄存器 SYSCTRL_CR1.xxxEN 为 0，关闭原时钟。

4.5.4 从其它时钟切换到 HSE 示例

切换操作流程如下：

步骤 1：设置 PB07/PC13 引脚为模拟功能端口；

步骤 2：根据 HSE 振荡器外接晶体特性，配置新时钟源 HSE 的振荡参数，如驱动能力（通过外置高频晶体控制寄存器 SYSCTRL_HSE.DRIVER 配置）、稳定周期（通过外置高频晶体控制寄存器 SYSCTRL_HSE.WAITCYCLE 配置，建议设置为 3）等；

步骤 3：设置系统控制寄存器 SYSCTRL_CR1.HSEEN 为 1，使能 HSE 晶体振荡电路；

注意：

SYSCTRL_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 4：根据当前时钟和 HSE 两者中较高的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 5：循环查询并等待外置高频晶体控制寄存器 SYSCTRL_HSE.STABLE 稳定标志变为 1 后，确保晶体振荡电路输出稳定时钟；

步骤 6：设置系统控制寄存器 SYSCTRL_CR0.SYSCLK 为 1，将 SysClk 时钟来源切换为 HSE；

注意：

SYSCTRL_CR0 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 7：根据 HSE 的时钟频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 8：设置系统控制寄存器 SYSCTRL_CR1.xxxEN 为 0，关闭原时钟。

4.5.5 从其它时钟切换到 LSI 示例

切换操作流程如下：

步骤 1：配置内置低频时钟控制寄存器 SYSCTRL_LSI.TRIM 及内置低频时钟控制寄存器 SYSCTRL_LSI.WAITCYCLE 为合适的值；

步骤 2：设置系统控制寄存器 SYSCTRL_CR1.LSIEN 为 1，使能 LSIRC 振荡电路；

注意：

SYSCTRL_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 3：循环查询并等待内置低频时钟控制寄存器 SYSCTRL_LSI.STABLE 稳定标志变为 1，即等待 LSI 输出稳定时钟；

步骤 4：设置系统控制寄存器 SYSCTRL_CR0.SYSCLK 为 3，将 SysClk 时钟来源切换为 LSI；

注意：

SYSCTRL_CR0 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 5：配置 FLASH_CR2.WAIT 读等待周期参数为 0；

步骤 6：设置系统控制寄存器 SYSCTRL_CR1.xxxEN 为 0，关闭原时钟。

4.5.6 从其它时钟切换到 HSI 示例

切换操作流程如下：

步骤 1：配置内置高频时钟控制寄存器 SYSCTRL_HSI.TRIM、SYSCTRL_HSI.WAITCYCLE 以及 SYSCTRL_HSI.DIV 为合适的值；

步骤 2：设置系统控制寄存器 SYSCTRL_CR1.HSIEN 为 1，使能 HSIOSC 时钟振荡电路；

注意：

SYSCTRL_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 3：循环查询并等待内置高频时钟控制寄存器 SYSCTRL_HSI.STABLE 稳定标志变为 1，即等待 HSIOSC 输出稳定时钟；

步骤 4：根据当前时钟和 HSI 两者中较高的频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 5：设置系统控制寄存器 SYSCTRL_CR0.SYSCLK 为 0，将 SysClk 时钟来源切换为 HSI；

注意：

SYSCTRL_CR0 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 6：根据 HSI 的时钟频率，按 7 FLASH 存储器章节要求配置 FLASH_CR2.WAIT 读等待周期参数；

步骤 7：设置系统控制寄存器 SYSCTRL_CR1.xxxEN 为 0，关闭原时钟。

4.6 寄存器列表

SYSCTRL 基地址：SYSCTRL_BASE = 0x4000 4000

表 4-9 SYSCTRL 寄存器列表

寄存器名称	寄存器地址	寄存器描述
SYSCTRL_CR0	SYSCTRL_BASE + 0x00	系统控制寄存器 0
SYSCTRL_CR1	SYSCTRL_BASE + 0x04	系统控制寄存器 1
SYSCTRL_CR2	SYSCTRL_BASE + 0x08	系统控制寄存器 2
SYSCTRL_IER	SYSCTRL_BASE + 0x0C	系统中断使能控制寄存器
SYSCTRL_ISR	SYSCTRL_BASE + 0x10	系统中断标志寄存器
SYSCTRL_ICR	SYSCTRL_BASE + 0x14	系统中断标志清除寄存器
SYSCTRL_HSI	SYSCTRL_BASE + 0x18	内置高频时钟控制寄存器
SYSCTRL_HSE	SYSCTRL_BASE + 0x1C	外置高频晶体控制寄存器
SYSCTRL_LSI	SYSCTRL_BASE + 0x20	内置低频时钟控制寄存器
SYSCTRL_LSE	SYSCTRL_BASE + 0x24	外置低频晶体控制寄存器
SYSCTRL_DEBUG	SYSCTRL_BASE + 0x2C	调试状态定时器控制寄存器
SYSCTRL_AHBEN	SYSCTRL_BASE + 0x30	AHB 外设时钟使能控制寄存器
SYSCTRL_APBEN2	SYSCTRL_BASE + 0x34	APB 外设时钟使能控制寄存器 2
SYSCTRL_APBEN1	SYSCTRL_BASE + 0x38	APB 外设时钟使能控制寄存器 1
SYSCTRL_AHBRST	SYSCTRL_BASE + 0x40	AHB 外设复位控制寄存器
SYSCTRL_APBRST2	SYSCTRL_BASE + 0x44	APB 外设复位控制寄存器 2
SYSCTRL_APBRST1	SYSCTRL_BASE + 0x48	APB 外设复位控制寄存器 1
SYSCTRL_RESETFLAG	SYSCTRL_BASE + 0x4C	系统复位标志寄存器
SYSCTRL_MCO	SYSCTRL_BASE + 0x70	系统时钟输出控制寄存器

4.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

4.7.1 SYSCTRL_CR0 系统控制寄存器 0

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:8	RFU	-	保留位，请保持默认值
7:5	HCLKPRS	RW	配置 HCLK 的时钟来源 000：设置 HCLK 的时钟源为 SysClk 001：设置 HCLK 的时钟源为 SysClk / 2 010：设置 HCLK 的时钟源为 SysClk / 4 011：设置 HCLK 的时钟源为 SysClk / 8 100：设置 HCLK 的时钟源为 SysClk / 16 101：设置 HCLK 的时钟源为 SysClk / 32 110：设置 HCLK 的时钟源为 SysClk / 64 111：设置 HCLK 的时钟源为 SysClk / 128
4:3	PCLKPRS	RW	配置 PCLK 的时钟来源 00：设置 PCLK 的时钟源为 HCLK 01：设置 PCLK 的时钟源为 HCLK / 2 10：设置 PCLK 的时钟源为 HCLK / 4 11：设置 PCLK 的时钟源为 HCLK / 8
2:0	SYSCLK	RW	配置 SysClk 的时钟来源 000：设置 SysClk 的时钟源为 HSI 001：设置 SysClk 的时钟源为 HSE 011：设置 SysClk 的时钟源为 LSI 100：设置 SysClk 的时钟源为 LSE

4.7.2 SYSCTRL_CR1 系统控制寄存器 1

Address offset: 0x04 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:9	RFU	-	保留位，请保持默认值
8	CLKCCS	RW	配置晶体振荡器运行中失效时的安全操作 0：无动作 1：自动将 SysClk 的时钟源切换为 HSI 4MHz
7	HSECCS	RW	高频晶体振荡器时钟监测器使能配置 0：禁止 HSE 时钟监测器 1：使能 HSE 时钟监测器 *注 1：运行中，每 HSE.DETCNT 个 LSI 时钟持续时间内 HSE 输出的时钟个数小于 0x20000 个，则判定 HSE 运行中失效。* *注 2：启动时，若不满足 HSE.WAITCYCLE 指定的条件，则判定 HSE 起振失败。* *注 3：使能该功能，DeepSleep 模式的功耗将增加约 1μA。*
6	LSECCS	RW	低频晶体振荡器时钟监测器使能配置 0：禁止 LSE 时钟监测器 1：使能 LSE 时钟监测器 *注 1：运行中，每 256 个 LSI 时钟持续时间（约 7.8ms）内 LSE 输出的时钟个数小于 128 个则判定 LSE 运行中失效。* *注 2：启动时，若不满足 LSE.WAITCYCLE 指定的条件，则判定 LSE 起振失败。* *注 3：使能该功能，DeepSleep 模式的功耗将增加约 1μA。*
5	LSELOCK	RW	外部低速时钟 LSE 锁定控制 0：LSEEN 可置位可清零 1：LSEEN 只可置位 *注 1：使能后，仅上电复位可以复位该位域。* *注 2：当产生 LSEFAIL 或 LSEFAULT 标志后该功能不起作用，即此时 LSEEN 可置位可清零，直到用户软件清除上述标志。*
4	LSEEN	RW	外部低速时钟 LSE 使能控制 0：关闭 1：使能 *注 1：当系统进入 DeepSleep，此低速时钟不会自动关闭。* *注 2：当 LSELOCK=0 时，可通过软件关闭 LSEEN，除此之外只能通过重新上电复位 LSEEN；当 LSELOCK=1 时，仅上电复位可以复位 LSEEN。*
3	LSIEN	RW	内部低速时钟 LSI 使能控制 0：关闭 1：使能 *注：当系统进入 DeepSleep，此低速时钟不会自动关闭。*

位域	名称	权限	功能描述
2	RFU	-	保留位，请保持默认值
1	HSEEN	RW	外部高速时钟 HSE 使能控制 0：关闭 1：使能 *注：当系统进入 DeepSleep，此高速时钟会自动关闭。*
0	HSIEN	RW	内部高速时钟 HSIOSC 使能控制 0：关闭 1：使能 *注：当系统进入 DeepSleep，此高速时钟会自动关闭。*

注：

以下方式能够硬件自动开启 LSI 时钟，但是 SYSCTRL_CR1.LSIEN 位域不会被置位，只有软件写 SYSCTRL_CR1.LSIEN 为 1 才能置位该位域：

（a）SYSCTRL_CR1.HSECCS = 1；

（b）SYSCTRL_CR1.LSECCS = 1；

（c）GPIOx_FILTER.FLTCLK = 0x5 (LSI)；

（d）VCx_CR1.FLTCLK = 0x0 (LSI) 且 VCx_CR0.EN = 1；

（e）LVD_CR0.FLTCLK = 0x0 (LSI) 且 LVD_CR0.EN = 1；

（f）IWDT_KR = 0xCCCC (启动 IWDT)。

4.7.3 SYSCTRL_CR2 系统控制寄存器 2

Address offset: 0x08 Reset value: 0x0000 0C00

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15	RAMBRKEN	RW	RAM 奇偶校验错误触发 ATIM 刹车使能控制 0：关闭 1：使能
14	LVDBRKEN	RW	LVD 输出触发 ATIM 刹车使能控制 0：关闭 1：使能
13	DSBRKEN	RW	DeepSleep 触发 ATIM 刹车使能控制 0：关闭 1：使能
12	CLLBRKEN	RW	产生 HardFault 或 Cortex-M0+ LockUp 标志触发 ATIM 刹车使能控制 0：关闭 1：使能
11	LSEBRKEN	RW	LSE 运行中失效触发 ATIM 刹车使能控制 0：关闭 1：使能
10	HSEBRKEN	RW	HSE 运行中失效触发 ATIM 刹车使能控制 0：关闭 1：使能
9:7	RFU	-	保留位，请保持默认值
6:4	FLASHWAIT	RW	FLASH 取指周期配置 000：1 个 HCLK 周期，适用于 HCLK <= 24MHz 001：2 个 HCLK 周期，适用于 24MHz < HCLK <= 48MHz 010：3 个 HCLK 周期，适用于 48MHz < HCLK <= 72MHz 011：4 个 HCLK 周期，适用于 72MHz < HCLK <= 96MHz *注：该控制位与 FLASH_CR2[2:0] 功能相同。*
3	WAKEUPCLK	RW	DeepSleep 唤醒时，系统时钟的来源配置 0：保持原系统时钟来源 1：切换系统时钟来源为 HSI 4MHz，原系统时钟来源保持使能
2	LOCKUP	RW	Cortex-M0+ LockUp 功能配置 0：关闭 1：使能 *注：使能该功能，则 CPU 读到无效指令时会复位 MCU。*
1	SWDIO	RW	SWD 端口功能配置 0：PA14、PA13 作为 SWD 端口，GPIO 功能不可用 1：PA14、PA13 作为 GPIO 端口，SWD 功能不可用
0	RFU	-	保留位，请保持默认值

4.7.4 SYSCTRL_HSI 内置高频时钟控制寄存器

Address offset: 0x18 Reset value: 0x---- ----

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	STABLE	RO	HSIOSC 时钟稳定状态位 0：HSIOSC 时钟尚未稳定 1：HSIOSC 时钟已经稳定
14:11	DIV	RW	HSI 时钟与 HSIOSC 时钟分频系数配置 0000：HSI = HSIOSC / 32 0001：HSI = HSIOSC / 1 0010：HSI = HSIOSC / 2 0011：HSI = HSIOSC / 3 0100：HSI = HSIOSC / 4 0101：HSI = HSIOSC / 5 0110：HSI = HSIOSC / 6 0111：HSI = HSIOSC / 7 1000：HSI = HSIOSC / 8 1001：HSI = HSIOSC / 9 1010：HSI = HSIOSC / 10 1011：HSI = HSIOSC / 12 1100：HSI = HSIOSC / 16 1101：HSI = HSIOSC / 20 1110：HSI = HSIOSC / 24 (默认值) 1111：HSI = HSIOSC / 28
10:0	TRIM	RW	时钟频率调整，更改该寄存器位的数值即可调整 HSIOSC 的振荡频率。TRIM 值每增加 1 则 HSIOSC 的振荡频率增加约 0.2%，总调整范围为 90 ~ 100MHz。FLASH 中已保存了 96MHz 的校准值，将 FLASH 内的校准值读出并写入 SYSCTRL_HSI.TRIM 即可获得精准的频率。 96.00MHz 校准值地址：0x0010 07C0 – 0x0010 07C1

4.7.5 SYSCTRL_LSI 内置低频时钟控制寄存器

Address offset: 0x20 Reset value: 0x---- ----

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	STABLE	RO	LSI 时钟稳定状态位 0：LSI 时钟尚未稳定 1：LSI 时钟已经稳定
14:12	RFU	-	保留位，请保持默认值
11:10	WAITCYCLE	RW	内部低速时钟 LSI 稳定时间选择 00：6 个周期 01：18 个周期 10：66 个周期 11：258 个周期
9:0	TRIM	RW	时钟频率调整，更改该寄存器位的数值即可调整 LSI 的振荡频率。TRIM 值每增加 1 则 LSI 的振荡频率增加约 0.4%，总调整范围为 30kHz ~ 36kHz。FLASH 中已保存了 1 组频率的校准值，从 FLASH 中读出预设的校准值，写入到本位域即可获得精准的 LSI 频率。 32.8kHz 校准值地址：0x0010 07C2 - 0x0010 07C3

4.7.6 SYSCTRL_HSE 外置高频晶体控制寄存器

Address offset: 0x1C Reset value: 0x0027 FF22

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	DIGFLT	RW	外部高速时钟 HSE 数字滤波控制 0：关闭 HSE 数字滤波功能 1：使能 HSE 数字滤波功能，HSE 频率应不大于 8MHz *注 1：干扰信号极强时才建议使能滤波功能。* *注 2：当 HCLK 来源为 HSE 时，禁止修改本控制位。*
23:20	PDRIVER	RW	HSE 时钟稳定前，HSE 晶体驱动能力配置；参见 DRIVER 位域说明。
19	STABLE	RO	HSE 时钟稳定状态位 0：HSE 时钟尚未稳定 1：HSE 时钟已经稳定
18:8	DETCNT	RW	HSE 运行失效检测周期配置 考虑时钟误差，一般应设置 DETCNT 的值为 8000 / f_{HSE} 例：HSE 为 4MHz，则 DETCNT 应配置为 2000。 *注：DETCNT 必须大于 0。*
7	HEXENPOL	RW	输出到外部有源晶振使能信号极性配置 0：Active 模式时输出高电平，DeepSleep 模式时输出低电平 1：Active 模式时输出低电平，DeepSleep 模式时输出高电平 *注：需要配置相应引脚为 HEXEN 复用输出。*
6	MODE	RW	外部高速时钟 HSE 工作模式配置 0：晶体模式，其输出时钟由晶体与内部电路谐振产生 1：输入模式，其输出时钟来自 OSC_IN 引脚
5:4	WAITCYCLE	RW	HSE 稳定检测及起振失败检测时钟周期数量选择 00：等待 8192 个 HSE 时钟信号 01：等待 32768 个 HSE 时钟信号 10：等待 131072 个 HSE 时钟信号 11：等待 262144 个 HSE 时钟信号 *注：当使能 HSECCS 后，起振失败检测电路会在限定时间内检测 HSE 输出时钟信号的数量，四档限定时间分别为 65ms、65ms、65ms、130ms。*
3:0	DRIVER	RW	HSE 晶体驱动能力配置 0000：最小驱动能力 … 0111：最强驱动能力 *注：需要根据晶体特性、负载电容以及电路板的寄生参数选择适当的驱动能力。驱动能力越强则功耗越大，驱动能力越弱则功耗越小。*

4.7.7 SYSCTRL_LSE 外置低频晶体控制寄存器

Address offset: 0x24 Reset value: 0x0000 0A22

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18	STABLE	RO	LSE 时钟稳定状态位 0：LSE 时钟尚未稳定 1：LSE 时钟已经稳定
17	PINLOCK	RW	外部低速时钟 LSE 引脚锁定为模拟状态 0：LSE 引脚可改写 1：LSE 引脚锁定为模拟状态 *注 1：使能后，仅上电复位可以复位该位域。* *注 2：仅当 SYSCTRL_CR1.LSELOCK 为 1 时该功能才生效。* *注 3：当产生 LSEFAIL 或 LSEFAULT 标志后该功能不起作用，即此时 LSEEN 可置位可清零，直到用户软件清除上述标志。*
16:12	RFU	-	保留位，请保持默认值
11:8	PDRIVER	RW	LSE 时钟稳定前，LSE 晶体驱动能力配置；参见 DRIVER 位域说明。
7	RFU	-	保留位，请保持默认值
6	MODE	RW	外部低速时钟 LSE 工作模式配置 0：晶体模式，其输出时钟由晶体与内部电路谐振产生 1：输入模式，其输出时钟来自 OSC32_IN 引脚
5:4	WAITCYCLE	RW	LSE 稳定检测及起振失败检测时钟周期数量选择 00：等待 256 个 LSE 时钟信号 01：等待 1024 个 LSE 时钟信号 10：等待 4096 个 LSE 时钟信号 11：等待 16384 个 LSE 时钟信号 *注：当使能 LSECCS 后，起振失败检测电路会在限定时间内检测 LSE 时钟信号数量，四档限定时间分别为 1s、1s、1s、2s。*
3:0	DRIVER	RW	LSE 晶体驱动能力配置 0000：最小驱动能力 … 1111：最强驱动能力 *注：需要根据晶体特性、负载电容以及电路板的寄生参数选择适当的驱动能力。驱动能力越强则功耗越大，驱动能力越弱则功耗越小。*

注意：

仅上电复位可以复位该寄存器。

4.7.8 SYSCTRL_IER 系统中断使能控制寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:9	RFU	-	保留位，请保持默认值
8	HSEFAULT	RW	HSE 运行失效中断使能控制 0：禁止 HSE 运行失效中断 1：使能 HSE 运行失效中断
7	LSEFAULT	RW	LSE 运行失效中断使能控制 0：禁止 LSE 运行失效中断 1：使能 LSE 运行失效中断
6	HSEFAIL	RW	HSE 起振失败中断使能控制 0：禁止 HSE 起振失败中断 1：使能 HSE 起振失败中断
5	LSEFAIL	RW	LSE 起振失败中断使能控制 0：禁止 LSE 起振失败中断 1：使能 LSE 起振失败中断
4	LSERDY	RW	LSE 稳定中断使能控制 0：禁止 LSE 稳定中断 1：使能 LSE 稳定中断
3	LSIRDY	RW	LSI 稳定中断使能控制 0：禁止 LSI 稳定中断 1：使能 LSI 稳定中断
2	RFU	-	保留位，请保持默认值
1	HSERDY	RW	HSE 稳定中断使能控制 0：禁止 HSE 稳定中断 1：使能 HSE 稳定中断
0	HSIRDY	RW	HSIOSC 稳定中断使能控制 0：禁止 HSIOSC 稳定中断 1：使能 HSIOSC 稳定中断

注：

HSEFAULT 中断和 LSEFAULT 中断产生时执行 FAULT 中断（向量编号 47），其它中断产生时执行 RCC 全局中断（向量编号 20）。

4.7.9 SYSCTRL_ISR 系统中断标志寄存器

Address offset: 0x10 Reset value: 0x0000 0801

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	LSESTABLE	RO	LSE 时钟稳定状态位；硬件置位清零 0：LSE 时钟尚未稳定 1：LSE 时钟已经稳定
14	LSISTABLE	RO	LSI 时钟稳定状态位；硬件置位清零 0：LSI 时钟尚未稳定 1：LSI 时钟已经稳定
13	RFU	-	保留位，请保持默认值
12	HSESTABLE	RO	HSE 时钟稳定状态位；硬件置位清零 0：HSE 时钟尚未稳定 1：HSE 时钟已经稳定
11	HSISTABLE	RO	HSIOSC 时钟稳定状态位；硬件置位清零 0：HSIOSC 时钟尚未稳定 1：HSIOSC 时钟已经稳定
10:9	RFU	-	保留位，请保持默认值
8	HSEFAULT	RO	HSE 时钟运行失效中断标志位 0：HSE 时钟运行正常 1：HSE 时钟运行中停振
7	LSEFAULT	RO	LSE 时钟运行失效中断标志位 0：LSE 时钟运行正常 1：LSE 时钟运行中停振 *注：产生 LSEFAULT 之后建议软件关闭 LSEEN 以清除 LSESTABLE 状态，然后软件清除 LSEFAULT 标志。如果没有关闭 LSEEN，则每 256 个 LSI 时钟持续时间（约 7.8ms）都会产生一次 LSEFAULT，直到 LSE 恢复正常。*
6	HSEFAIL	RO	HSE 时钟起振失败中断标志位 0：HSE 时钟起振成功 1：HSE 时钟起振失败
5	LSEFAIL	RO	LSE 时钟起振失败中断标志位 0：LSE 时钟起振成功 1：LSE 时钟起振失败 *注：产生 LSEFAIL 之后可以软件关闭 LSEEN，然后软件清除 LSEFAIL 标志。如果没有关闭 LSEEN，则每隔一段时间就会产生一次 LSEFAIL，间隔时间参见 SYSCTRL_LSE 寄存器的 WAITCYCLE 位域描述。*

位域	名称	权限	功能描述
4	LSERDY	RO	LSE 时钟稳定中断标志位 0：LSE 时钟尚未稳定 1：LSE 时钟已经稳定 *注：清零 ICR.LSEFAULT、ICR.LSEFAIL、ICR.LSERDY 都能够清除 LSERDY 标志。*
3	LSIRDY	RO	LSI 时钟稳定中断标志位 0：LSI 时钟尚未稳定 1：LSI 时钟已经稳定
2	RFU	-	保留位，请保持默认值
1	HSERDY	RO	HSE 时钟稳定中断标志位 0：HSE 时钟尚未稳定 1：HSE 时钟已经稳定 *注：清零 ICR.HSEFAULT、ICR.HSEFAIL、ICR.HSERDY 都能够清除 HSERDY 标志。*
0	HSIRDY	RO	HSIOSC 时钟稳定中断标志位 0：HSIOSC 时钟尚未稳定 1：HSIOSC 时钟已经稳定

注 1：

HSEFAULT 中断和 LSEFAULT 中断产生时执行 FAULT 中断（向量编号 47），其它中断产生时执行 RCC 全局中断（向量编号 20）。

注 2：

时钟的 STABLE 信号由硬件置位由硬件清零，RDY 信号由硬件置位由软件写相应的 ICR 寄存器位域清零。

4.7.10 SYSCTRL_ICR 系统中断标志清除寄存器

Address offset: 0x14 Reset value: 0x0000 D9FB

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8	HSEFAULT	R1W0	HSE 时钟运行失效中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能 *注：清除 HSEFAULT 标志会同步清除 HSERDY 标志。*
7	LSEFAULT	R1W0	LSE 时钟运行失效中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能 *注：清除 LSEFAULT 标志会同步清除 LSERDY 标志。*
6	HSEFAIL	R1W0	HSE 时钟起振失败中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能 *注：清除 HSEFAIL 标志会同步清除 HSERDY 标志。*
5	LSEFAIL	R1W0	LSE 时钟起振失败中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能 *注：清除 LSEFAIL 标志会同步清除 LSERDY 标志*
4	LSERDY	R1W0	LSE 时钟稳定中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
3	LSIRDY	R1W0	LSI 时钟稳定中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
2	RFU	-	保留位，请保持默认值
1	HSERDY	R1W0	HSE 时钟稳定标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
0	HSIRDY	R1W0	HSIOSC 时钟稳定中断标志位清零控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能

4.7.11 SYSCTRL_AHBEN AHB 外设时钟使能控制寄存器

Address offset: 0x30 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:7	RFU	-	保留位，请保持默认值
6	GPIOC	RW	GPIOC 端口配置时钟及工作时钟使能控制 0：关闭 1：使能
5	GPIOB	RW	GPIOB 端口配置时钟及工作时钟使能控制 0：关闭 1：使能
4	GPIOA	RW	GPIOA 端口配置时钟及工作时钟使能控制 0：关闭 1：使能
3	RFU	-	保留位，请保持默认值
2	CRC	RW	CRC 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
1	FLASH	RW	FLASH 模块配置时钟使能控制 0：关闭 1：使能
0	RFU	-	保留位，请保持默认值

4.7.12 SYSCTRL_APBEN1 APB 外设时钟使能控制寄存器 1

Address offset: 0x38 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:9	RFU	-	保留位，请保持默认值
8	UART3	RW	UART3 模块配置时钟使能控制 0：关闭 1：使能
7	GTIM2	RW	GTIM2 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
6	GTIM1	RW	GTIM1 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
5	ATIM	RW	ATIM 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
4	UART2	RW	UART2 模块配置时钟使能控制 0：关闭 1：使能
3	UART1	RW	UART1 模块配置时钟使能控制 0：关闭 1：使能
2	SPI	RW	SPI 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
1	VC	RW	VC 模块配置时钟使能控制 0：关闭 1：使能
0	ADC	RW	ADC 模块配置时钟及工作时钟使能控制 0：关闭 1：使能

4.7.13 SYSCTRL_APBEN2 APB 外设时钟使能控制寄存器 2

Address offset: 0x34 Reset value: 0x0000 0100

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:8	RFU	-	保留位，请保持默认值
7	LPTIM	RW	LPTIM 模块配置时钟使能控制 0：关闭 1：使能
6	I2C	RW	I2C 模块配置时钟使能控制 0：关闭 1：使能
5	RFU	-	保留位，请保持默认值
4	IWDT	RW	IWDT 模块配置时钟使能控制 0：关闭 1：使能
3	RFU	-	保留位，请保持默认值
2	BTIM123	RW	BTIM1-3 模块配置时钟及工作时钟使能控制 0：关闭 1：使能
1	RTC	RW	RTC 模块配置时钟使能控制 0：关闭 1：使能
0	RFU	-	保留位，请保持默认值

4.7.14 SYSCTRL_AHBRST AHB 外设复位控制寄存器

Address offset: 0x40 Reset value: 0x0000 007F

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	GPIOC	RW	GPIOC 模块复位控制 0：模块处于复位状态 1：模块正常工作
5	GPIOB	RW	GPIOB 模块复位控制 0：模块处于复位状态 1：模块正常工作
4	GPIOA	RW	GPIOA 模块复位控制 0：模块处于复位状态 1：模块正常工作
3	RFU	-	保留位，请保持默认值
2	CRC	RW	CRC 模块复位控制 0：模块处于复位状态 1：模块正常工作
1	FLASH	RW	FLASH 模块复位控制 0：模块处于复位状态 1：模块正常工作
0	RFU	-	保留位，请保持默认值

4.7.15 SYSCTRL_APBRST1 APB 外设复位控制寄存器 1

Address offset: 0x48 Reset value: 0x0000 01FF

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8	UART3	RW	UART3 模块复位控制 0：模块处于复位状态 1：模块正常工作
7	GTIM2	RW	GTIM2 模块复位控制 0：模块处于复位状态 1：模块正常工作
6	GTIM1	RW	GTIM1 模块复位控制 0：模块处于复位状态 1：模块正常工作
5	ATIM	RW	ATIM 模块复位控制 0：模块处于复位状态 1：模块正常工作
4	UART2	RW	UART2 模块复位控制 0：模块处于复位状态 1：模块正常工作
3	UART1	RW	UART1 模块复位控制 0：模块处于复位状态 1：模块正常工作
2	SPI	RW	SPI 模块复位控制 0：模块处于复位状态 1：模块正常工作
1	VC	RW	VC 模块复位控制 0：模块处于复位状态 1：模块正常工作
0	ADC	RW	ADC 模块复位控制 0：模块处于复位状态 1：模块正常工作

4.7.16 SYSCTRL_APBRST2 APB 外设复位控制寄存器 2

Address offset: 0x44 Reset value: 0x0000 00FF

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	LPTIM	RW	LPTIM 模块复位控制 0：模块处于复位状态 1：模块正常工作
6	I2C	RW	I2C 模块复位控制 0：模块处于复位状态 1：模块正常工作
5	RFU	-	保留位，请保持默认值
4	IWDT	RW	IWDT 模块复位控制 0：模块处于复位状态 1：模块正常工作
3	RFU	-	保留位，请保持默认值
2	BTIM123	RW	BTIM1-3 模块复位控制 0：模块处于复位状态 1：模块正常工作
1	RTC	RW	RTC 模块复位控制 0：模块处于复位状态 1：模块正常工作
0	RFU	-	保留位，请保持默认值

4.7.17 SYSCTRL_RESETFLAG 系统复位标志寄存器

Address offset: 0x4C Reset value: 0x---- ----

位域	名称	权限	功能描述
31:10	RFU	-	保留位，请保持默认值
9	SYSRESETREQ	RW0	Cortex-M0+ CPU SYSRESETREQ 软复位标志 0：未发生 CPU SYSRESETREQ 软复位 1：已发生 CPU SYSRESETREQ 软复位 写 0 清除，写 1 无效
8	LOCKUP	RW0	Cortex-M0+ CPU Lockup 复位标志 0：未发生 Lockup 复位 1：已发生 Lockup 复位 写 0 清除，写 1 无效
7	RFU	-	保留位，请保持默认值
6	RSTB	RW0	NRST 引脚复位标志 0：未发生引脚复位 1：已发生引脚复位 写 0 清除，写 1 无效
5	RFU	-	保留位，请保持默认值
4	IWDT	RW0	IWDT 复位标志 0：未发生 IWDT 复位 1：已发生 IWDT 复位 写 0 清除，写 1 无效
3	LVD	RW0	LVD 复位标志 0：未发生 LVD 复位 1：已发生 LVD 复位 写 0 清除，写 1 无效
2:1	RFU	-	保留位，请保持默认值
0	POR	RW0	POR/BOR 复位标志 0：未发生 POR/BOR 复位 1：已发生 POR/BOR 复位 写 0 清除，写 1 无效

4.7.18 SYSCTRL_DEBUG 调试状态定时器控制寄存器

Address offset: 0x2C Reset value: 0x0000 07E7

位域	名称	权限	功能描述
31:10	RFU	-	保留位，请保持默认值
9	IWDT	RW	调试状态下，IWDT 计数功能配置 0：正常计数 1：暂停计数
8	RTC	RW	调试状态下，RTC 计数功能配置 0：正常计数 1：暂停计数
7	RFU	-	保留位，请保持默认值
6	LPTIM	RW	调试状态下，LPTIM 计数功能配置 0：正常计数 1：暂停计数
5	BTIM123	RW	调试状态下，BTIM1 / BTIM2 / BTIM3 计数功能配置 0：正常计数 1：暂停计数
4:3	RFU	-	保留位，请保持默认值
2	GTIM2	RW	调试状态下，GTIM2 计数功能配置 0：正常计数 1：暂停计数
1	GTIM1	RW	调试状态下，GTIM1 计数功能配置 0：正常计数 1：暂停计数
0	ATIM	RW	调试状态下，ATIM 计数功能配置 0：正常计数 1：暂停计数

4.7.19 SYSCTRL_MCO 系统时钟输出控制寄存器

Address offset: 0x70 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6:4	DIV	RW	MCO 输出分频控制 000：1 分频 001：2 分频 010：8 分频 011：64 分频 100：128 分频 101：256 分频 110：512 分频 111：1024 分频
3:0	SOURCE	RW	MCO 输出信号来源配置 0000：无输出 0001：HCLK 0010：PCLK 0011：HSIOSC 0100：LSI 0101：HSE 0110：LSE

5 中断

5.1 概述

ARM® Cortex®-M0+ 内核的嵌套向量中断控制器 (NVIC)，用于管理中断和异常。NVIC 和处理器内核紧密相连，可以实现低延迟的异常和中断处理。

处理器支持最多 32 个中断请求 (IRQ) 输入，支持多个内部异常。

本章节只介绍了处理器的 32 个外部中断请求（IRQ0 ~ IRQ31），处理器内部异常的具体情况请参考“ARM® Cortex®-M0+Technical Reference Manual”与“ARM® v6-M Architecture Reference Manual”。

5.2 主要特性

- 16 个内部异常
- 32 个可屏蔽外部中断
- 4 个可编程的优先级
- 低延时的异常和中断处理
- 支持中断嵌套
- 中断向量表重映射

5.3 中断优先级

外部中断可设置 4 级优先级，最高优先级为“0”，最低优先级为“3”，默认值为“0”。

当处理器正在执行一个中断处理程序时，如果出现一个更高优先级的中断，那么这个中断就被抢占。如果出现的中断的优先级和正在处理的中断的优先级相同或更低，这个中断就不会被抢占，但是新中断的状态就变为挂起。如果多个挂起的中断具有相同的优先级，中断编号越小的挂起中断优先处理。例如，如果 IRQ[0] 和 IRQ[1] 均挂起时，并且两者的优先级相同，那么先处理 IRQ[0]。

5.4 中断向量表

ARM® Cortex®-M0+ 响应中断时，处理器自动从存储器的中断向量表中取出中断服务程序 (ISR) 的起始地址。中断向量表包括主栈指针（MSP）的初始值，内部异常和外部中断的服务程序入口地址。每个中断向量占用 1 个字（4 字节），中断向量的存储地址为向量编号乘以 4，如下表所示：

表 5-1 中断向量表

向量编号	外部中断号（IRQ#）	优先级	中断源	简介	地址
0	-	-	-	MSP 初始值	0x0000 0000
1	-	-3	Reset	复位向量	0x0000 0004
2	-	-2	NMI	不可屏蔽中断 [2]	0x0000 0008
3	-	-1	HardFault	硬件错误异常（fault）	0x0000 000C
4-10	-	-	-	保留	0x0000 0010~002B
11	-	可配置	SVCall	通过 SWI 指令调用的管理程序	0x0000 002C
12-13	-	-	-	保留	0x0000 0030~0037
14	-	可配置	PendSV	系统服务的可挂起请求	0x0000 0038
15	-	可配置	SysTick	系统滴答定时器	0x0000 003C
16	0	可配置	WDT	独立看门狗中断	0x0000 0040
17	1	可配置	LVD	LVD 全局中断	0x0000 0044
18	2	可配置	RTC	RTC 全局中断	0x0000 0048
19	3	可配置	FLASHRAM[1]	FLASH/RAM 全局中断	0x0000 004C
20	4	可配置	RCC	RCC 全局中断 [3]	0x0000 0050
21	5	可配置	GPIOA	GPIOA 全局中断	0x0000 0054
22	6	可配置	GPIOB	GPIOB 全局中断	0x0000 0058
23	7	可配置	GPIOC	GPIOC 全局中断	0x0000 005C
24	8	-	-	保留	0x0000 0060
25	9	-	-	保留	0x0000 0064
26	10	-	-	保留	0x0000 0068
27	11	-	-	保留	0x0000 006C
28	12	可配置	ADC	ADC 全局中断	0x0000 0070
29	13	可配置	ATIM	ATIM 全局中断	0x0000 0074
30	14	可配置	VC1	VC1 全局中断	0x0000 0078
31	15	可配置	VC2	VC2 全局中断	0x0000 007C
32	16	可配置	GTIM1	GTIM1 全局中断	0x0000 0080

向量编号	外部中断号（IRQ#）	优先级	中断源	简介	地址
33	17	可配置	GTIM2	GTIM2 全局中断	0x0000 0084
34	18	-	-	保留	0x0000 0088
35	19	可配置	LPTIM	LPTIM 全局中断	0x0000 008C
36	20	可配置	BTIM1	BTIM1 全局中断	0x0000 0090
37	21	可配置	BTIM2	BTIM2 全局中断	0x0000 0094
38	22	可配置	BTIM3	BTIM3 全局中断	0x0000 0098
39	23	可配置	I2C	I2C 全局中断	0x0000 009C
40	24	-	-	保留	0x0000 00A0
41	25	可配置	SPI	SPI 全局中断	0x0000 00A4
42	26	-	-	保留	0x0000 00A8
43	27	可配置	UART1	UART1 全局中断	0x0000 00AC
44	28	可配置	UART2	UART2 全局中断	0x0000 00B0
45	29	可配置	UART3	UART3 全局中断	0x0000 00B4
46	30	-	-	保留	0x0000 00B8
47	31	可配置	FAULT	HSE/LSE 运行中失效中断 [4]	0x0000 00BC

注 1:

由于部分外设的中断复用一个 IRQ 中断源，用户在中断服务程序中应先检查中断标志位，以确定产生中断的外设。

注 2:

NMI 在 CW32L011 中未使用。

注 3:

HSE、LSE 时钟信号起振失败和 LSI、LSE、HSIOSC、HSE 时钟信号稳定对应 RCC 全局中断。

注 4:

HSE 或 LSE 时钟信号在运行中失效对应 FAULT 中断。

5.5 中断相关寄存器

5.5.1 NVIC 中断使能和禁止使能

ARM® Cortex®-M0+ 处理器支持最多 32 个外部中断源，分别对应中断使能设置寄存器 NIVC_ISER 的 32 个使能位，和中断使能清除寄存器 NVIC_ICER 的 32 个禁止位。将使能位置 1，允许中断；将禁止位置 1，禁止中断。

上文中 NVIC 中断使能仅针对处理器 NVIC 而言，外设的中断是否使能，还受相应外设的中断控制寄存器控制。

5.5.2 NVIC 中断挂起和清除挂起

在中断发生时，如果系统正在处理与之相同优先级或更高优先级的中断，系统将不会立即处理此中断，而是将中断的状态设置为挂起，保存在中断挂起状态寄存器中; 在处理器未进入此中断处理之前，如没有手动清除挂起状态，该状态将会一直保持有效。当处理器开始进入中断处理时，硬件会自动清除相应的中断挂起状态。

用户可通过设置中断挂起设置寄存器 NVIC_ISPR 的对应位，将此中断的状态设置为挂起状态，如果系统没有正在处理与之相同优先级或更高优先级的中断，此中断将被立即响应并处理。

用户可以通过设置中断挂起清除寄存器 NVIC_ICPR 的对应位，将此中断的状态设置为挂起清除状态。

5.5.3 NVIC 中断优先级

中断优先级控制寄存器 NVIC_IPR0 ~ NVIC_IPR7，用于设置 IRQ0~IRQ31 的中断优先级，每个中断源使用 8 位，在 CW32L011 中仅使用了高两位，最多可设置 4 个中断优先级。

注意：

ARM® Cortex®-M0+ 的中断优先级寄存器的设置应在中断使能之前，用户不可在中断使能之后改变中断优先级，这将导致不可预知的结果。

5.5.4 NVIC 中断屏蔽

在某些特殊场合，需要禁止所有中断，可以使用中断屏蔽寄存器 PRIMASK 实现。PRIMASK 只有最低 1 位有效，将此位置 1，除了 NMI 和硬件错误异常之外的所有外部中断和异常都被禁止；清 0 后，允许响应中断和异常。该位复位后默认为 0。

ARM® Cortex-M0+ 有专用的 ARM 指令用于修改 PRIMASK 寄存器，CPSIE i 和 CPSID i，详细请参考《ARM® v6-M Architecture Reference Manual》。

汇编指令示例参考：

```
CPSIE i    ; 清除 PRIMASK （使能中断）
CPSID i    ; 设置 PRIMASK （禁止中断）
```

C 语言（调用 CMSIS 设备驱动库）示例参考：

```
void __enable_irq(void);        // 清除 PRIMASK
void __disable_irq(void);       // 设置 PRIMASK
```

5.5.5 外设中断使能

外设模块一般都有各自的中断使能寄存器，在使用中断时，必须首先打开外设中断使能，同时参见表 5-1 中断向量表打开该中断源的 NVIC 中断使能。具体外设的中断使能，请参阅相关外设模块章节描述。

5.6 寄存器列表

NVIC 基地址：NVIC_BASE = 0xE000 E000

表 5-2 NVIC 寄存器列表

寄存器名称	寄存器地址	寄存器描述
NVIC_ISER	NVIC_BASE + 0x100	IRQ0~IRQ31 中断使能设置寄存器
NVIC_ICER	NVIC_BASE + 0x180	IRQ0~IRQ31 中断使能清除寄存器
NVIC_ISPR	NVIC_BASE + 0x200	IRQ0~IRQ31 中断挂起设置寄存器
NVIC_ICPR	NVIC_BASE + 0x280	IRQ0~IRQ31 中断挂起清除寄存器
NVIC_IPR0	NVIC_BASE + 0x400	IRQ0~IRQ3 中断优先级控制寄存器 0
NVIC_IPR1	NVIC_BASE + 0x404	IRQ4~IRQ7 中断优先级控制寄存器 1
NVIC_IPR2	NVIC_BASE + 0x408	IRQ8~IRQ11 中断优先级控制寄存器 2
NVIC_IPR3	NVIC_BASE + 0x40C	IRQ12~IRQ15 中断优先级控制寄存器 3
NVIC_IPR4	NVIC_BASE + 0x410	IRQ16~IRQ19 中断优先级控制寄存器 4
NVIC_IPR5	NVIC_BASE + 0x414	IRQ20~IRQ23 中断优先级控制寄存器 5
NVIC_IPR6	NVIC_BASE + 0x418	IRQ24~IRQ27 中断优先级控制寄存器 6
NVIC_IPR7	NVIC_BASE + 0x41C	IRQ28~IRQ31 中断优先级控制寄存器 7

5.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

5.7.1 NVIC_ISER 中断使能设置寄存器

Address offset: 0x100 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:0	SETIRQ	RW	设置使能外部中断 IRQ0 ~ IRQ31；写“1”置位，写“0”无效。 [0]：IRQ0 [1]：IRQ1 [2]：IRQ2 …… [31]：IRQ31 读出值表示当前中断使能状态

5.7.2 NVIC_ICER 中断使能清除寄存器

Address offset: 0x180 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:0	CLRIRQ	RW	设置禁止外部中断 IRQ0 ~ IRQ31；写“1”置位，写“0”无效。 [0]：IRQ0 [1]：IRQ1 [2]：IRQ2 …… [31]：IRQ31 读出值表示当前中断使能状态

5.7.3 NVIC_ISPR 中断挂起设置寄存器

Address offset: 0x200 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:0	SETPEND	RW	设置外部中断 IRQ0 ~ IRQ31 的挂起状态；写“1”置位，写“0”无效。 [0]：IRQ0 [1]：IRQ1 [2]：IRQ2 …… [31]：IRQ31 读出值表示当前中断挂起状态

5.7.4 NVIC_ICPR 中断挂起清除寄存器

Address offset: 0x280 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:0	CLRPEND	RW	清除外部中断 IRQ0 ~ IRQ31 的挂起状态; 写 “1” 清除，写 “0” 无效。 [0]：IRQ0 [1]：IRQ1 [2]：IRQ2 …… [31]：IRQ31 读出值表示当前中断挂起状态

5.7.5 NVIC_IPR0 中断优先级控制寄存器 0

Address offset: 0x400 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ3	RW	中断 IRQ3 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ2	RW	中断 IRQ2 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ1	RW	中断 IRQ1 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ0	RW	中断 IRQ0 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.6 NVIC_IPR1 中断优先级控制寄存器 1

Address offset: 0x404 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ7	RW	中断 IRQ7 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ6	RW	中断 IRQ6 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ5	RW	中断 IRQ5 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ4	RW	中断 IRQ4 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.7 NVIC_IPR2 中断优先级控制寄存器 2

Address offset: 0x408 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ11	RW	中断 IRQ11 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ10	RW	中断 IRQ10 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ9	RW	中断 IRQ9 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ8	RW	中断 IRQ8 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.8 NVIC_IPR3 中断优先级控制寄存器 3

Address offset: 0x40C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ15	RW	中断 IRQ15 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ14	RW	中断 IRQ14 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ13	RW	中断 IRQ13 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ12	RW	中断 IRQ12 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.9 NVIC_IPR4 中断优先级控制寄存器 4

Address offset: 0x410 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ19	RW	中断 IRQ19 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ18	RW	中断 IRQ18 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ17	RW	中断 IRQ17 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ16	RW	中断 IRQ16 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.10 NVIC_IPR5 中断优先级控制寄存器 5

Address offset: 0x414 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ23	RW	中断 IRQ23 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ22	RW	中断 IRQ22 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ21	RW	中断 IRQ21 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ20	RW	中断 IRQ20 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.11 NVIC_IPR6 中断优先级控制寄存器 6

Address offset: 0x418 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ27	RW	中断 IRQ27 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ26	RW	中断 IRQ26 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ25	RW	中断 IRQ25 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ24	RW	中断 IRQ24 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

5.7.12 NVIC_IPR7 中断优先级控制寄存器 7

Address offset: 0x41C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ31	RW	中断 IRQ31 的优先级，00 优先级最高，11 优先级最低
29:24	RFU	-	保留位，请保持默认值
23:22	IPRIRQ30	RW	中断 IRQ30 的优先级，00 优先级最高，11 优先级最低
21:16	RFU	-	保留位，请保持默认值
15:14	IPRIRQ29	RW	中断 IRQ29 的优先级，00 优先级最高，11 优先级最低
13:8	RFU	-	保留位，请保持默认值
7:6	IPRIRQ28	RW	中断 IRQ28 的优先级，00 优先级最高，11 优先级最低
5:0	RFU	-	保留位，请保持默认值

6 RAM 存储器

6.1 概述

CW32L011 内部集成 6KB 嵌入式 RAM 供用户使用，用来存放程序执行过程中的各种数据。RAM 的起始地址为 0x2000 0000，数据在 RAM 中以小端模式存储，即最低字节地址空间存放数据的最低有效字节数据。

6.2 主要特性

- 支持以字节（8bit）、半字（16bit）或全字（32bit）3 种位宽进行访问
- 零等待延迟，能够被 CPU 以最大的系统时钟频率进行访问
- 支持奇偶校验功能

6.3 RAM 存储器操作

用户可执行的 RAM 存储器操作包括：读操作、写操作。

对 RAM 的读写操作支持 8bit、16bit 和 32bit 三种位宽，用户程序可以通过直接访问绝对地址的方式完成读写，但要注意读写的数据位宽必须和对应地址边界对齐，否则读写操作无效，并会导致 HardFault 硬件错误异常。

6.3.1 读操作

读操作支持3种不同位宽，可采用直接访问绝对地址方式读取，但要注意读取的数据位宽必须和对应地址边界对齐。

代码示例：

8bit 读：

```
tempdata = * ( ( uint8_t * ) 0x2000 0001 ) ;
```

16bit 读：

```
tempdata = * ( ( uint16_t * ) 0x2000 0002 ) ;
```

32bit 读：

```
tempdata = * ( ( uint32_t * ) 0x2000 0004 ) .
```

6.3.2 写操作

写操作支持 3 种不同位宽，可采用直接访问绝对地址的方式写入数据，但要注意写入的数据位宽必须和对应地址边界对齐。

代码示例：

8bit 写：

```
* ( ( uint8_t * ) 0x2000 0001 ) = 0x12;
```

16bit 写：

```
* ( ( uint16_t * ) 0x2000 0002 ) = 0x1234;
```

32bit 写：

```
* ( ( uint32_t * ) 0x2000 0004 ) = 0x1234 5678;
```

6.4 奇偶校验功能

CW32L011 支持 RAM 的奇偶校验功能，上电后奇偶校验功能默认打开，用户不可配置。

每字节 RAM 数据实际存放在 9bit 的物理空间中，包括 8bit 数据位和 1bit 奇偶校验位。

CPU 在对 RAM 进行写入时，RAM 的奇偶校验单元会计算校验位并写入对应的校验位空间。在读取 RAM 数据时，数据连同校验位一起被读取，CPU 对数据进行每字节的奇偶校验，并将计算的校验位和读取的校验位进行比较，如果一致则说明 RAM 中数据正确，如果不一致则奇偶校验错误标志 RAM_ISR.PARITY 被置位，如果设置 RAM 奇偶校验出错中断使能控制位 RAM_IER.PARITY 为 1，CPU 会响应中断服务。用户程序可设置 RAM_ICR.PARITY 为 0 来清除奇偶校验错误标志。

用户可通过读取奇偶校验出错地址寄存器 RAM_ADDR，以获取发生奇偶校验错误的 RAM 地址。

6.5 寄存器列表

RAM 基地址：RAM_BASE = 0x4002 2400

表 6-1 RAM 寄存器列表

寄存器名称	寄存器地址	寄存器描述
RAM_IER	RAM_BASE + 0x00	中断使能控制寄存器
RAM_ADDR	RAM_BASE + 0x04	奇偶校验出错地址寄存器
RAM_ISR	RAM_BASE + 0x08	中断标志寄存器
RAM_ICR	RAM_BASE + 0x0C	中断标志清除寄存器

6.6 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

6.6.1 RAM_ADDR 奇偶校验出错地址寄存器

Address offset: 0x04 Reset value: 0x2000 0000

位域	名称	权限	功能描述
31:0	ADDR	RO	奇偶校验出错的 RAM 所在的地址

6.6.2 RAM_IER 中断使能控制寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	PARITY	RW	RAM 奇偶校验出错中断使能控制 0：禁止 1：使能

6.6.3 RAM_ISR 中断标志寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	PARITY	RO	奇偶校验错误标志 0：未发生奇偶校验错误 1：已发生奇偶校验错误

6.6.4 RAM_ICR 中断标志清除寄存器

Address offset: 0x0C Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	PARITY	R1W0	奇偶校验错误标志清除控制 W0：清除奇偶校验错误标志 W1：无功能

7 FLASH 存储器

7.1 概述

CW32L011 内部集成 64KB 嵌入式 FLASH 供用户使用，可用来存储应用程序和用户数据。芯片支持对 FLASH 存储器的读、擦除和写操作，支持擦写保护和读保护；芯片内置 FLASH 编程所需的高压 BOOST 电路，无须额外提供编程电压。

7.2 主要特性

- 支持以字节（8bit）、半字（16bit）或全字（32 bit）3 种位宽进行访问
- 支持读、擦除和写操作
- 支持擦写保护、读保护
- 支持安全运行库保护

7.3 FLASH 存储器组织

CW32L011 内部集成 64KB 用户可访问的 FLASH 存储器，按每页 512 字节进行分页管理，共 128 页，用户可以对 FLASH 进行整页擦除和逐字节编程操作。

除用户可访问的 64KB 的 FLASH 存储器外，CW32L011 还集成 2KB 的启动程序存储器，该存储器出厂时固化了 BootLoader 等代码，用户不可访问该存储器。

7.4 FLASH 存储器读等待周期配置

CW32L011 内部的 FLASH 存储器支持最快 24MHz 频率的操作时钟，当配置的 HCLK 频率大于 24MHz 时，需通过 FLASH 控制寄存器 FLASH_CR2 的 WAIT 位域来配置插入的等待 HCLK 周期个数，配置规则如下表所示：

表 7-1 FLASH 读等待周期配置

FLASH_CR2.WAIT	取指周期等待 HCLK 周期个数	HCLK 频率
000	0 个	HCLK ⩽ 24MHz
001	1 个	HCLK ⩽ 48MHz
010	2 个	HCLK ⩽ 72MHz
011	3 个	HCLK ⩽ 96MHz

注意：

配置 FLASH 读等待周期时，必须先使能 FLASH 配置时钟。

7.5 FLASH 存储器操作

用户可执行的 FLASH 存储器操作包括：读操作、擦除、写（编程）操作。

对 FLASH 的读写操作支持 8bit、16bit 和 32bit 三种位宽，用户程序可以通过直接访问绝对地址的方式完成读写，但要注意读写的数据位宽必须和对应地址边界对齐，否则会导致 HardFault 硬件错误异常。

7.5.1 页擦除

FLASH 的页擦除操作的最小单位为 1 页，即 512 字节。页擦除操作完成后，该页所有地址空间的数据内容均为 0xFF。

如果对未解锁的 FLASH 页面进行页擦除操作，会操作失败，同时 FLASH_ISR.PAGELOCK 标志位会被硬件置位，如果设置 FLASH_IER.PAGELOCK 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PAGELOCK 为 0 来清除 FLASH_ISR.PAGELOCK 中断标志。

如果程序在 FLASH 中运行，且对 PC（程序指针）所在页面的存储空间进行页擦除操作，会操作失败，同时 FLASH_ISR.PC 标志位会被硬件置位，如果设置 FLASH_IER.PC 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PC 为 0 来清除 FLASH_ISR.PC 中断标志。

FLASH 页擦除操作步骤如下：

步骤 1：设置 FLASH 操作模式为页擦除模式，向 FLASH 控制寄存器 FLASH_CR1 的工作模式位域 MODE 写入 0x02；

注意：

FLASH_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 2：解锁待擦除页的擦除保护，参见表 7-2 FLASH_PAGELOCK 擦写锁定保护，向擦写锁定寄存器 FLASH_PAGELOCK 对应的锁定位域 LOCKx 写入 1；

注意：

FLASH_PAGELOCK 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 3：对待擦除页内的任意地址写入任意数据，触发对该页的擦除操作。

注意：

写入数据位宽需要和对应地址边界对齐。

代码示例：

```
*( ( uint8_t * ) 0x0000 0001 ) = 0x55;
```

步骤 4：等待擦除完成，查询 FLASH 忙标志位 FLASH_ISR.BUSY 变成 0；

步骤 5：如需擦除其它页，则重复步骤 2 至步骤 4；

步骤 6：擦除操作完成后，如需恢复对 FLASH 的锁定保护，需向擦写锁定寄存器 FLASH_PAGELOCK 对应的锁定位域 LOCKx 写入 0，启动锁定保护功能，保护 FLASH 内容不被误擦写；

步骤 7：设置 FLASH 操作模式为读模式，向 FLASH 控制寄存器 FLASH_CR1 的工作模式位域 MODE 写入 0x00。

7.5.2 写操作

基于嵌入式 FLASH 的特性，写操作只能将 FLASH 存储器中位数据由‘1’改写为‘0’，不能由‘0’改写为‘1’，因此在写数据之前先要对对应地址所在页进行擦除操作。

对 FLASH 写操作必须遵循以下三个原则：

- 不可对数据位内容为‘0’的地址写入
- 不可对锁定区域内的地址写入
- 不可对 PC（程序指针）所在的页的地址写入

CW32L011 在执行写操作时，会检查存储空间内容是否全为‘1’。FLASH 的 8bit、16bit、32bit 三种写操作位宽，需要检查的字节数分别为 1Byte、2Byte、4Byte。

如果对内部不全是‘1’的存储空间进行写入操作，会写入失败，同时 FLASH_ISR.PROG 标志位会被硬件置位，如果设置 FLASH_IER.PROG 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PROG 为 0 来清除 FLASH_ISR.PROG 中断标志。

如果对未解锁的 FLASH 页面的存储空间进行写操作，会操作失败，同时 FLASH_ISR.PAGELOCK 标志位会被硬件置位，如果设置 FLASH_IER.PAGELOCK 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PAGELOCK 为 0 来清除 FLASH_ISR.PAGELOCK 中断标志。

如果程序在 FLASH 中运行，且对 PC（程序指针）所在页面的存储空间进行写操作，会操作失败，同时 FLASH_ISR.PC 标志位会被硬件置位，如果设置 FLASH_IER.PC 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PC 为 0 来清除 FLASH_ISR.PC 中断标志。

FLASH 写操作步骤如下：

步骤 1：设置 FLASH 操作模式为写（编程）模式，向 FLASH 控制寄存器 FLASH_CR1 的工作模式位域 MODE 写入 0x01；

注意：

FLASH_CR1 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 2：解锁待写页的擦除保护，参见表 7-2 FLASH_PAGELOCK 擦写锁定保护，向擦写锁定寄存器 FLASH_PAGELOCK 对应的锁定位域 LOCKx 写入 1；

注意：

FLASH_PAGELOCK 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

步骤 3：将待写入的数据写入目标地址，触发 FLASH 的写入操作，注意写入数据位宽需要和对应地址边界对齐；

8bit 位宽代码示例：

```
*( ( uint8_t * ) 0x0000 0001 ) = 0x55;
```

16bit 位宽代码示例：

```
*( ( uint16_t * ) 0x0000 0002 ) = 0x1234;
```

32bit 位宽代码示例：

```
*( ( uint32_t * ) 0x0000 0004 ) = 0x1234 5678;
```

步骤 4：等待编程完成，查询等待 FLASH 忙标志位 FLASH_ISR.BUSY 变成 0；

步骤 5：如需写更多数据到当前已解除锁定区域的地址范围内，重复步骤 3 至步骤 4；如写入地址超出当前已解除锁定区域的地址范围，重复步骤 2 至步骤 4；

步骤 6：写操作完成后，如需恢复对 FLASH 的锁定保护，向擦写锁定寄存器 FLASH_PAGELOCK 对应的锁定位域 LOCKx 写入 0，启动锁定保护功能，保护 FLASH 内容不被误擦写；

步骤 7：设置 FLASH 操作模式为读模式，向 FLASH 控制寄存器 FLASH_CR1 的工作模式位域 MODE 写入 0x00。

7.5.3 读操作

CW32L011 对 FLASH 的读操作支持 3 种不同位宽，可采用直接访问绝对地址方式读取，注意读取的数据位宽必须和对应地址边界对齐。

8bit 位宽代码示例：

```
tempdata = *( ( uint8_t * ) 0x0000 0001 ) ;
```

16bit 位宽代码示例：

```
tempdata = *( ( uint16_t * ) 0x0000 0002 ) ;
```

32bit 位宽代码示例：

```
tempdata = *( ( uint32_t * ) 0x0000 0004 ) ;
```

7.6 FLASH 存储器保护

FLASH 存储器具有擦写保护和读保护功能。

擦写保护包括锁定页擦写保护和 PC 地址页擦写保护，处于保护状态的页面不能被擦写，可避免 FLASH 内容被意外改写。

读保护以整片 FLASH 为保护对象，不支持单页保护，可避免用户代码被非法读取。

7.6.1 擦写保护

CW32L011 通过设置擦写锁定寄存器 FLASH_PAGELOCK 来实现 FLASH 页面的擦写锁定，处于锁定状态的页面不能进行页擦除或写操作。

CW32L011 内部 FLASH 存储器被划分为 128 页，每 8 页对应擦写锁定寄存器 FLASH_PAGELOCK 的 1 个 LOCKx 锁定位。LOCKx 域共 16 位，可实现全部 128 页 FLASH 存储器的锁定保护。擦写锁定寄存器 FLASH_PAGELOCK 的各位域与 FLASH 锁定页面的对应关系如下表所示：

表 7-2 FLASH_PAGELOCK 擦写锁定保护

FLASH_PAGELOCK 位域	位域名称	锁定页面
15	LOCK15	Page120 – Page127
14	LOCK14	Page112 – Page119
13	LOCK13	Page104- Page111
12	LOCK12	Page96 – Page103
11	LOCK11	Page88 – Page95
10	LOCK10	Page80 – Page87
9	LOCK9	Page72 – Page79
8	LOCK8	Page64- Page71
7	LOCK7	Page56 - Page63
6	LOCK6	Page48 - Page55
5	LOCK5	Page40 - Page47
4	LOCK4	Page32 - Page39
3	LOCK3	Page24 - Page31
2	LOCK2	Page16 - Page23
1	LOCK1	Page8 - Page15
0	LOCK0	Page0 - Page7

对 FLASH 进行页擦除和写操作时，必须先通过设置对应的 FLASH_PAGELOCK.LOCKx 位域为 1 来解锁对应页面。解锁操作示例如下：

锁定 Page0-Page7 代码示例：

```
CW_FLASH->PAGELOCK = 0x5A5A 0000 | ( CW_FLASH->PAGELOCK & 0x0000 FFFE );
```

解锁 Page32-Page39 代码示例：

```
CW_FLASH->PAGELOCK = 0x5A5A 0000 | ( CW_FLASH->PAGELOCK | 0x0000 0010 );
```

注意：

FLASH_PAGELOCK 寄存器具有 KEY 保护特性，写入的数据高 16bit 数据必须是 0x5A5A，否则无法写入。

对未解锁的 FLASH 页直接执行页擦除或者写操作，会操作失败并产生中断标志，请参见 7.5.1 页擦除和 7.5.2 写操作。

7.6.2 擦写 PC 页保护

CW32L011 的 FLASH 存储器支持擦写 PC 页保护功能。

当用户程序运行 FLASH 时，如果当前程序指针 PC 正好位于待擦写的 FLASH 地址页范围内，则该擦写操作失败，同时 FLASH_ISR.PC 标志位会被硬件置位，如果设置 FLASH_IER.PC 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 FLASH_ICR.PC 为 0 来清除 FLASH_ISR.PC 中断标志。

7.6.3 读保护

CW32L011 支持 FLASH 读保护功能，设置读保护后，无法通过 ISP 或 SWD 方式对 FLASH 进行读取操作。读保护只支持整片 FLASH 保护，不支持按页保护。

读保护分为 4 个保护等级，当前保护等级可通过读取 FLASH 控制寄存器 FLASH_CR1 的安全位域 SECURITY 来获取，安全位域是只读属性，不能修改。读保护等级可通过 ISP 指令进行设定，请参阅 ISP 编程文档。

CPU 从 FLASH 中取指操作和程序对 FLASH 的读取操作，不受 FLASH 读保护功能影响。

FLASH 的读保护功能如下表所示：

表 7-3 FLASH 的读保护

保护等级	FLASH_CR1.SECURITY	功能描述
Level0	00	未设置读保护。 可以通过 SWD 或 ISP 方式对 FLASH 进行读取操作。
Level1	01	FLASH 内容不可通过 SWD 或 ISP 方式读取。 可通过 ISP 或 SWD 方式将保护等级降低到 Level0，降级之后 FLASH 中内容为全 0xFF，即处于整片擦除的空片状态。
Level2	10	FLASH 内容不可通过 SWD 或 ISP 方式读取。 仅可通过 ISP 方式将保护等级降低，但降级之后 FLASH 中内容为全 0xFF，即处于整片擦除的空片状态。
Level3	11	FLASH 内容不可通过 SWD 或 ISP 方式读取。 ISP 和 SWD 降级功能都被禁止。 在此保护等级下，芯片只能进行一次编程。

7.6.4 安全运行库保护

CW32L011 的 FLASH 存储器支持安全运行库功能，方案商可将核心算法存储于安全运行库区域以供客户二次开发时调用。

使能安全运行库保护功能后，用户代码可正常调用运行安全运行库区域内的函数，但无法通过任何方式 (CPU、SWD、ISP) 读出安全运行库区域的内容，只有提供正确的密码时才能通过 ISP 协议禁止安全运行库保护功能，同时清空安全运行库存储空间的数据。

使能安全运行库保护功能的方法为：在地址 0xFFF0~ 0xFFFF 写入特定控制字，如下表所示：

表 7-4 使能安全运行库保护功能控制字

地址	0xFFF0	0xFFF1	0xFFF2	0xFFF3	0xFFF4	0xFFF5	0xFFF6	0xFFF7
控制字	SlibKey0	SlibKey1	SlibKey2	SlibKey3	SlibKey4	SlibKey5	SlibKey6	SlibKey7
地址	0xFFF8	0xFFF9	0xFFFA	0xFFFB	0xFFFC	0xFFFD	0xFFFE	0xFFFF
控制字	StartIdx	EndIdx	0x5A	0x5A	0xA5	0xA5	CRCL	CRCH

各控制字的含义如下：

- SlibKey0 ~ SlibKey7 为安全运行库密码，用于禁止安全运行库保护功能，用户应按实际需求进行设定并妥善保管，切勿将密码设置为全 FF、全 00。
- StartIdx 为安全运行库区域的起始页面序号，用户应按实际情况进行设置，写入后用户可通过 FLASH_SDKCFR 寄存器的 START 位域读出。
- EndIdx 为安全运行库区域的结束页面序号，固定为 0x7F，写入后用户可通过 FLASH_SDKCFR 寄存器的 END 位域读出。
- CRCL 为采用 CRC16_X25 算法对 0xFFF0 ~ 0xFFFD 数据计算所得结果的低字节。
- CRCH 为采用 CRC16_X25 算法对 0xFFF0 ~ 0xFFFD 数据计算所得结果的高字节。

禁止安全运行库保护功能有以下方法：

1. 通过 ISP 协议执行片擦操作：芯片接收到片擦指令及正确的安全运行库密码时，将自动擦除本芯片的所有数据并禁止安全运行库保护功能；芯片接收到片擦指令及错误的安全运行库密码时，不执行任何操作，所有区域的数据均不被擦除，安全运行库保护功能仍处于使能。指令请参阅 ISP 编程文档。

7.7 FLASH 存储器编程

CW32L011 内部集成 FLASH 存储器支持通过 SWD 或 ISP 方式进行 FLASH 的擦除和编程。SWD 方式是利用调试工具，通过 SWD 接口进行 FLASH 的擦除和编程。ISP 方式是通过芯片出厂前预装的 BootLoader 代码进行 FLASH 的擦除和编程。ISP 提供了快速和高效的在线擦除和编程方法，请参阅 ISP 编程文档。

7.8 注意事项

为正确操作 FLASH 和提高 FLASH 的访问效率及使用寿命，用户在编程应用时需要注意以下事项：

- 地址对齐要求

  地址边界对齐，即使用 16bit 位宽访问 FLASH 时的地址必须是偶地址，使用 32bit 位宽时的地址必须是 4 的倍数地址。

  正确地址对齐的代码示例：

  8bit 读取：

  ```
  tempdata = *( ( uint8_t * ) 0x0000 0001 ) ;
  ```

  16bit 读取：

  ```
  tempdata = *( ( uint16_t * ) 0x0000 0002 ) ;
  ```

  32bit 读取：

  ```
  tempdata = *( ( uint32_t * ) 0x0000 0004 ) ;
  ```

  错误地址对齐的代码示例：

  16bit 读取：

  ```
  tempdata = *( ( uint16_t * ) 0x0000 0001 ) ;
  ```

  32bit 读取：

  ```
  tempdata = *( ( uint32_t * ) 0x0000 0003 ) ;
  ```

- 操作完成标志查询

  当 CPU 从 FLASH 中取指并运行时，如果执行对 FLASH 的页擦除 / 写操作，CPU 会自动停止下一条指令存取，硬件自动等待擦写操作完成（FLASH_ISR.BUSY 状态位变成 0），故用户程序不必循环查询操作完成标志来判断操作是否完成。

  当 CPU 从 RAM 中取指并运行时，如果执行对 FLASH 的页擦除 / 写操作，在执行该操作的同时，CPU 会进行下一条指令存取，为保证程序下一条指令的执行正确性，用户程序必须在对 FLASH 擦写操作后循环查询 FLASH_ISR.BUSY 标志位，直到 FLASH_ISR.BUSY 标志位变成 0 后方可执行后续的任务。

  在进入深度休眠模式之前，若 FLASH 正在进行擦写操作，则必须等待 FLASH_ISR.BUSY 标志位清 0，同时须确保 FLASH_CR1.MODE 为 0。

- 使用寿命

  基于嵌入式 FLASH 的特性，FLASH 的操作次数和存储时间是有限的，用户在应用程序中应尽量避免频繁对某一页或某一地址的 FLASH 存储器进行擦写操作，以保证数据的可靠存储。具体寿命数据请参阅数据手册。

- 数据存储模式

  CW32L011 应用规定，数据在 FLASH 中以小端模式存储，即最低字节地址空间存放数据的最低有效字节数据。

7.9 寄存器列表

FLASH 基地址：FLASH_BASE = 0x4002 2000

表 7-5 FLASH 寄存器列表

寄存器名称	寄存器地址	寄存器描述
FLASH_CR1	FLASH_BASE + 0x00	控制寄存器 1
FLASH_CR2	FLASH_BASE + 0x04	控制寄存器 2
FLASH_PAGELOCK	FLASH_BASE + 0x08	擦写锁定寄存器
FLASH_IER	FLASH_BASE + 0x20	中断使能寄存器
FLASH_ISR	FLASH_BASE + 0x24	中断标志寄存器
FLASH_ICR	FLASH_BASE + 0x28	中断标志清除寄存器
FLASH_SDKCFR	FLASH_BASE + 0x70	安全运行库区域寄存器

7.10 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

7.10.1 FLASH_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:7	RFU	-	保留位，请保持默认值
6:5	SECURITY	RO	当前保护等级 00：Level0，ISP 可读写，SWD 可读写 01：Level1，ISP 可降级，SWD 可降级；数据不可读出 10：Level2，ISP 可降级，SWD 无功能；数据不可读出 11：Level3，ISP 无功能，SWD 无功能；数据不可读出
4:2	RFU	-	保留位，请保持默认值
1:0	MODE	RW	操作模式配置 00：读模式，Read 01：写模式，Program 10：页擦模式，PageErase 11：片擦模式，ChipErase

7.10.2 FLASH_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15:3	RFU	-	保留位，请保持默认值
2:0	WAIT	RW	FLASH 取指周期配置 000：1 个 HCLK 周期，适用于 HCLK <= 24MHz 001：2 个 HCLK 周期，适用于 24MHz < HCLK <= 48MHz 010：3 个 HCLK 周期，适用于 48MHz < HCLK <= 72MHz 011：4 个 HCLK 周期，适用于 72MHz < HCLK <= 96MHz *注：该控制位与 SYSCTRL_CR2[6:4] 功能相同。*

7.10.3 FLASH_PAGELOCK 擦写锁定寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	KEY	WO	仅当 KEY 为 0x5A5A 时，对该寄存器的写操作有效
15	LOCK15	RW	Page120 – Page127 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
14	LOCK14	RW	Page112 – Page119 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
13	LOCK13	RW	Page104 – Page111 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
12	LOCK12	RW	Page96 – Page103 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
11	LOCK11	RW	Page88 – Page95 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
10	LOCK10	RW	Page80 – Page87 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
9	LOCK9	RW	Page72 – Page79 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
8	LOCK8	RW	Page64 – Page71 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
7	LOCK7	RW	Page56 – Page63 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
6	LOCK6	RW	Page48 – Page55 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
5	LOCK5	RW	Page40 – Page47 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
4	LOCK4	RW	Page32 – Page39 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写

位域	名称	权限	功能描述
3	LOCK3	RW	Page24 – Page31 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
2	LOCK2	RW	Page16 – Page23 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
1	LOCK1	RW	Page8 – Page15 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
0	LOCK0	RW	Page0 – Page7 擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写

7.10.4 FLASH_IER 中断使能寄存器

Address offset: 0x20 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:5	RFU	-	保留位，请保持默认值
4	PROG	RW	编程错误中断使能控制 0：禁止 1：使能
3	RFU	-	保留位，请保持默认值
2	SDKERR	RW	擦写安全运行库区域错误中断使能控制 0：禁止 1：使能
1	PAGELOCK	RW	擦写 PAGELOCK 锁定的页面中断使能控制 0：禁止 1：使能
0	PC	RW	擦写 PC 所在页面中断使能控制 0：禁止 1：使能

7.10.5 FLASH_ISR 中断标志寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:6	RFU	-	保留位，请保持默认值
5	BUSY	RO	擦写状态标志 0：擦写操作已完成 1：擦写操作未完成
4	PROG	RO	编程错误标志 0：当前待写入地址的每个字节的内容全是 0xFF 1：当前待写入地址的每个字节的内容不全是 0xFF
3	RFU	-	保留位，请保持默认值
2	SDKERR	RO	擦写安全运行库区域错误中断标志 0：当前擦写区域不包括 SDK 区域 1：当前擦写区域包含 SDK 区域
1	PAGELOCK	RO	擦写 PAGELOCK 锁定的页面中断标志 0：当前擦写地址位于 PAGELOCK 锁定的页面之外 1：当前擦写地址位于 PAGELOCK 锁定的页面之内
0	PC	RO	擦写 PC 指针所在页面中断标志 0：当前擦写地址位于 PC 指针所在的页面之外 1：当前擦写地址位于 PC 指针所在的页面之内

7.10.6 FLASH_ICR 中断标志清除寄存器

Address offset: 0x28 Reset value: 0x0000 001F

位域	名称	权限	功能描述
31:5	RFU	-	保留位，请保持默认值
4	PROG	R1W0	编程错误标志清除 W0：清除编程错误标志 W1：无功能
3	RFU	-	保留位，请保持默认值
2	SDKERR	R1W0	擦写安全运行库区域错误中断标志清除 W0：清除擦写安全运行库区域错误中断标志 W1：无功能
1	PAGELOCK	R1W0	擦写 PAGELOCK 锁定的页面中断标志清除 W0：清除擦写 PAGELOCK 锁定的页面中断标志 W1：无功能
0	PC	R1W0	擦写 PC 指针所在页面中断标志清除 W0：清除擦写 PC 指针所在页面中断标志 W1：无功能

7.10.7 FLASH_SDKCFR 安全运行库区域寄存器

Address offset: 0x70 Reset value: 0x0000 007F

位域	名称	权限	功能描述
31:15	RFU	-	保留位，请保持默认值
14:8	END	RO	安全运行库区域结束地址所在的 Flash Page 的索引 0~127
7	RFU	-	保留位，请保持默认值
6:0	START	RO	安全运行库区域起始地址所在的 Flash Page 的索引 0~127

8 通用输入输出端口（GPIO）

8.1 概述

GPIO 控制器实现芯片内部各类数字和模拟电路与物理引脚之间的联系。

GPIO 可配置为数字输入输出和模拟功能，支持外设功能复用，支持上升沿和下降沿 2 种中断源，可在深度休眠模式下通过外部中断唤醒 MCU 回到运行模式。

8.2 主要特性

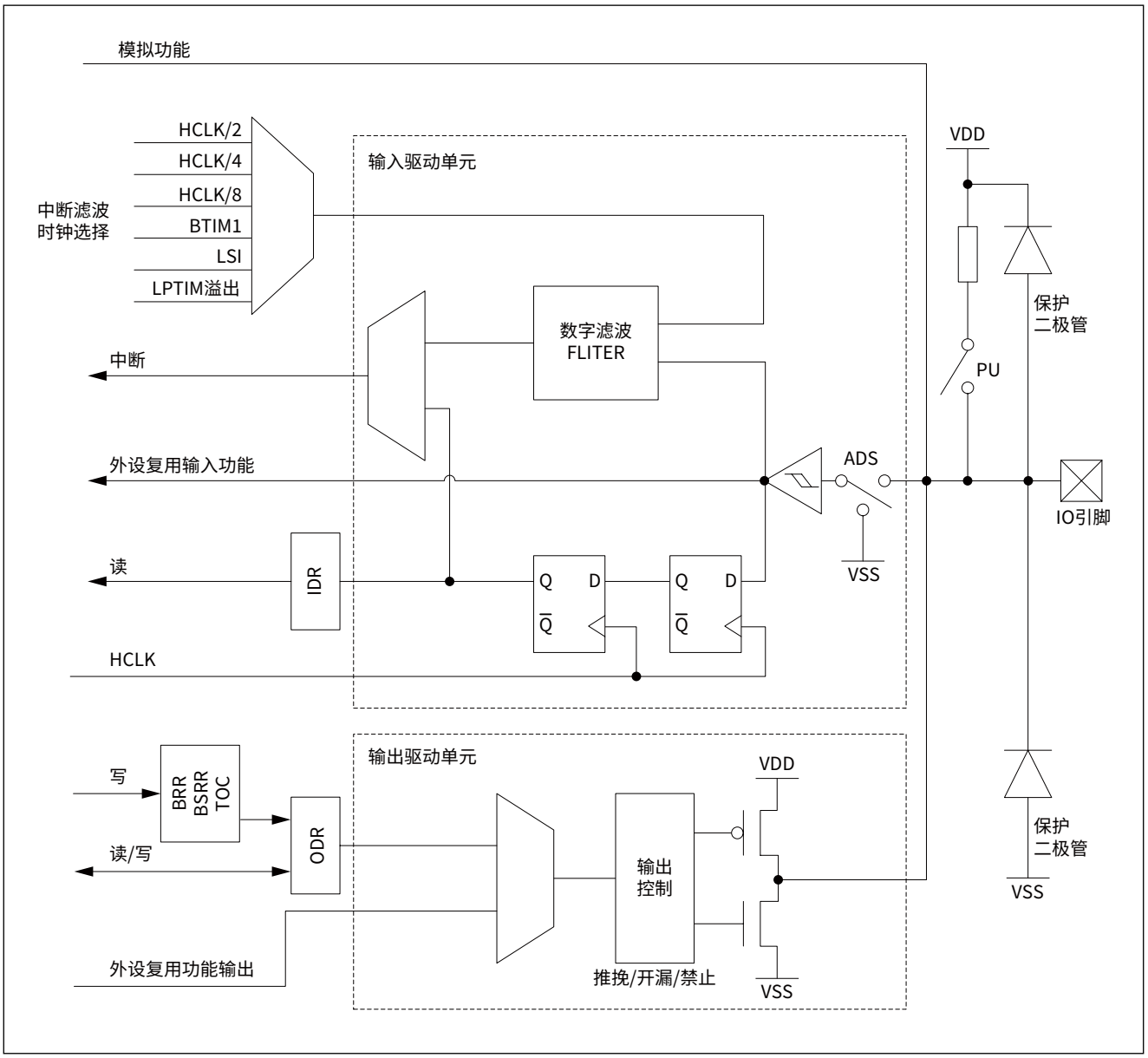
- 所有寄存器通过 AHB 总线接口读写
- 具有数字输入输出和模拟功能
- 数字输入输出支持普通 GPIO 和功能复用
- 模拟功能可作为 ADC、VC、LVD 的输入信号
- 支持内部多种时钟信号输出
- 数字输入支持内部上拉和高阻两种模式
- 数字输出支持推挽和开漏模式
- 数字输出支持位置位，位清零，位翻转的原子位操作
- 中断功能支持上升沿、下降沿触发方式
- 中断具有数字滤波功能，可选择 6 种时钟源
- 支持在深度休眠模式下通过外部中断唤醒 MCU

8.3 功能描述

8.3.1 功能框图

GPIO 控制器的功能框图如下图所示：

图 8-1 GPIO 功能框图



8.3.2 数字输出

将模拟数字配置寄存器 GPIOx_ANALOG[y]（x 是 GPIO 端口号，x=A、B、C；y 是引脚号，y = 0 ~ 15；下同）清零，配置相应的 GPIO 端口为数字功能；将输入输出方向寄存器 GPIOx_DIR[y] 清零，配置 GPIO 端口为输出模式。数字输出信号来源可以是：

- 输出数据寄存器 GPIOx_ODR
- 片内数字外设

通过输出模式寄存器 GPIOx_OPENDRAIN 配置输出模式，可选择推挽输出或开漏输出。

8.3.3 数字输入

将模拟数字配置寄存器 GPIOx_ANALOG[y] 清零，配置 GPIO 端口为数字功能；将输入输出方向寄存器 GPIOx_DIR[y] 置位，配置相应的 GPIO 端口为输入模式。数字输入信号可配置：

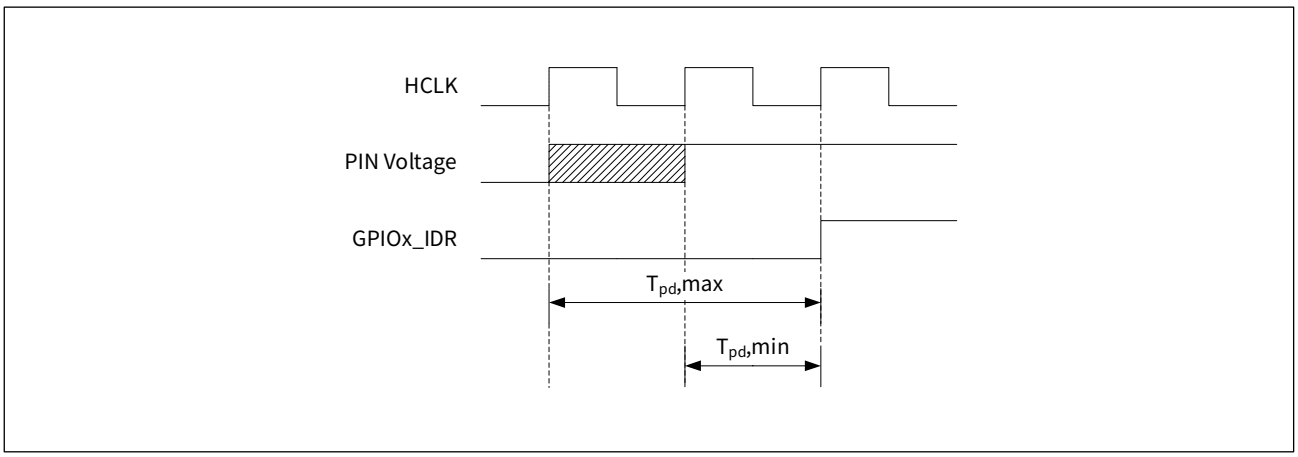
- 到达输入数据寄存器 GPIOx_IDR
- 到达片内数字外设
- 触发中断

在该模式下，数字输入信号通过 ADS 开关导入内部数字输入电路。

经施密特触发器确认电平状态后，可以直接被送往片内复用功能所指向的数字外设的输入，或者通过一个基于 HCLK 的同步器后，在输入数据寄存器 GPIOx_IDR[y] 上呈现。

GPIOx_IDR 寄存器的各位与其前面的锁存器组成了一个同步器，可以避免系统时钟变化的时间内引脚电平跳化而造成的信号不稳定，但是会产生一定的读取延迟。读端口引脚的同步时序如下图所示：

图 8-2 读端口引脚同步时序



在系统时钟上升沿之后的时钟周期，引脚电平信号会锁存在内部寄存器，如图中阴影部分所示，在下一次系统时钟上升沿之后，稳定的引脚电平信号被读取，再一个系统时钟上升沿时，数据被锁存到 GPIOx_IDR 寄存器中。信号延迟 T_{pd} 为 1~2 个系统时钟。

如果考虑将该输入信号用于触发中断，还可以启用内置的硬件滤波器电路。该滤波器电路是基于双 D 触发器同步器实现的，该同步器的时钟来源有 6 种，其中部分时钟源是低功耗模式特有的。例如，可以轻易的实现无软件干预的按键消抖操作。具体的时钟源选项及边沿触发选项请参见 8.3.6 中断功能。

在该模式下，通过上拉电阻寄存器 GPIOx_PUR 可以单独选择打开或者关闭内部上拉功能。

8.3.4 模拟功能

对于配置有模拟功能的 GPIO 端口，可通过设置模拟数字配置寄存器 GPIOx_ANALOG[y] 为 1，打开 GPIO 模拟信号通道。在打开 GPIO 模拟信号通道时，端口的数字功能关闭，内部上拉被断开，内部数字输入信号通过 ADS 开关被短接到 VSS，内部数字输出功能被禁止。

8.3.5 复用功能

通过复用功能寄存器 (GPIOx_AFRH 和 GPIOx_AFRL)，可以实现输入输出端口的复用功能。复用功能寄存器中每 4bit 的位域对应一个 GPIO 端口的复用功能选择，逻辑上可以选择多达 16 个输入输出信号目标。GPIO 复用功能具体定义如下表所示：

表 8-1 GPIO 复用功能设置

GPIOx_AFRL[4×y]	复用功能	GPIOx_AFRH[4×(z-8)]	复用功能
0000	GPIO	0000	GPIO
0001	AF1	0001	AF1
0010	AF2	0010	AF2
0011	AF3	0011	AF3
0100	AF4	0100	AF4
0101	AF5	0101	AF5
0110	AF6	0110	AF6
0111	AF7	0111	AF7

注：y、z 为引脚号：y = 0 ~ 7，对应 GPIOx_AFRL；z = 8 ~ 15，对应 GPIOx_AFRH。

每一个 AF 对应的具体外设功能如下表所示：

表 8-2 GPIO 复用功能分配表

引脚名称	复用功能							
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA00	GPIO	UART1_CTS	UART3_RXD	RTC_TAMP	VC1_OUT	BTIM3_TOGN	GTIM2_CH1	ATIM_BK
PA01	GPIO	UART1_RTS	UART3_TXD	RTC_OUT	HEXEN	BTIM3_TOGP	GTIM2_CH2	ATIM_ETR
PA02	GPIO	UART1_TXD	UART2_TXD	BTIM2_ETR	VC2_OUT	LPTIM_CH1	GTIM2_CH3	ATIM_CH4N
PA03	GPIO	UART1_RXD	UART2_RXD	RTC_OUT	PCLK_OUT	LPTIM_CH2	GTIM2_CH4	ATIM_CH4
PA04	GPIO	UART3_TXD	I2C_SCL	SPI_CS	MCO_OUT	LPTIM_OUT	ADC_SAM	ATIM_CH2N
PA05	GPIO	UART3_RXD	I2C_SDA	SPI_SCK	BTIM2_TOGN	LPTIM_ETR	GTIM1_CH3	ATIM_CH1
PA06	GPIO	UART2_RXD	BTIM1_ETR	SPI_MISO	BTIM2_TOGP	LPTIM_CH1	GTIM1_CH1	ATIM_BK
PA07	GPIO	UART2_TXD	GTIM2_ETR	SPI_MOSI	BTIM1_TOGN	LPTIM_CH2	GTIM1_CH2	ATIM_CH1N
PA08	GPIO	UART3_TXD		IR_OUT	MCO_OUT	BTIM2_TOGP	LVD_OUT	ATIM_CH1
PA09	GPIO	UART1_TXD	UART3_RXD	SPI_CS	I2C_SCL	LPTIM_ETR	BTIM1_TOGN	ATIM_CH2
PA10	GPIO	UART1_RXD	BTIM3_ETR	SPI_SCK	I2C_SDA	LPTIM_OUT	BTIM1_TOGP	ATIM_CH3
PA11	GPIO	UART1_CTS	VC1_OUT	SPI_MISO	HEXEN	ATIM_BK2	GTIM2_CH2	ATIM_CH4
PA12	GPIO	UART1_RTS	VC2_OUT	SPI_MOSI	BTIM3_ETR	ATIM_BK	GTIM2_CH3	ATIM_ETR
PA13	GPIO	UART1_RXD	UART3_TXD	UART2_RTS	I2C_SDA	ATIM_BK	IR_OUT	ATIM_CH6
PA14	GPIO	UART1_TXD	UART3_RXD	UART2_CTS	I2C_SCL	ATIM_BK2	GTIM1_ETR	ATIM_CH6N
PA15	GPIO	UART1_RXD	UART2_RXD	SPI_CS	ATIM_BKOUT	GTIM2_CH1	GTIM2_ETR	ATIM_CH1N

引脚名称	复用功能							
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PB00	GPIO	UART1_RXD	UART2_TXD	HSIOSC_OUT	BTIM1_TOGP	ATIM_ETR	GTIM1_CH3	ATIM_CH2N
PB01	GPIO	UART1_TXD	UART2_RXD	RTC_TAMP	IR_OUT	BTIM3_TOGN	GTIM1_CH4	ATIM_CH3N
PB03	GPIO	UART1_TXD	UART2_TXD	SPI_SCK	UART3_CTS	GTIM2_CH2	GTIM1_ETR	ATIM_CH2N
PB04	GPIO	UART3_TXD	UART2_CTS	SPI_MISO	UART3_RTS	ATIM_ETR	GTIM1_CH1	ATIM_CH3N
PB05	GPIO	UART3_RXD	UART2_RTS	SPI_MOSI	ADC_SAM	ATIM_BK	GTIM1_CH2	ATIM_CH1
PB06	GPIO	UART1_TXD	UART2_RXD	I2C_SCL	BTIM3_TOGN	GTIM2_CH3	GTIM1_CH3	ATIM_CH2
PB07	GPIO	UART1_RXD	UART2_TXD	I2C_SDA	BTIM3_TOGP	GTIM2_CH4	GTIM1_CH4	ATIM_CH3
PC13	GPIO	MCO_OUT	RTC_TAMP	LVD_OUT	BTIM3_ETR		ATIM_BK	ATIM_ETR
PC14	GPIO	UART1_RXD	UART2_RXD	UART3_CTS	I2C_SDA	BTIM2_TOGN	ATIM_CH6	ATIM_CH5N
PC15	GPIO	UART1_TXD	UART2_TXD	UART3_RTS	I2C_SCL	BTIM2_TOGP	ATIM_CH6N	ATIM_CH5

8.3.6 中断功能

每个 GPIO 在设置为数字输入模式时，可作为外部中断信号源，产生中断的信号源可以设置为上升沿、下降沿 2 种。中断触发方式可组合使用，但共用同一个中断标志位。

中断触发后，中断标志寄存器 GPIOx_ISR 的对应位会被硬件置位，程序可通过查询 GPIOx_ISR 来确认产生中断的端口。通过中断标志清除寄存器 GPIOx_ICR[y]，可以清除对应的中断标志位。

内部的中断数字滤波器可对引脚上的输入信号进行数字滤波，提供了 6 种滤波时钟选择，如下表所示：

表 8-3 数字滤波时钟选择

GPIOx_FILTER.FLTCLK	数字滤波时钟频率
000	HCLK / 2
001	HCLK / 4
010	HCLK / 8
011	BTIM1 溢出
101	LSI（约 32.8kHz）
111	LPTIM 溢出

由于选择的滤波时钟周期范围宽广，用户可以轻易实现灵活的输入中断防抖功能。输入电平的变化如果未保持超过一个完整的滤波时钟周期，将不会通过硬件滤波器传达到内部中断触发电路。输入电平的变化如果保持超过两个完整的滤波时钟周期，则一定会通过硬件滤波器。

对于边沿触发类型，考虑到对触发沿的时间的敏感性，建议在中断数字滤波器配置寄存器 GPIOx_FILTER[y] 中关闭硬件滤波器功能，因为硬件滤波器在提升信号稳定性的同时，也会插入一定延迟。

当 CW32L011 工作于休眠模式 (Sleep mode) 或深度休眠模式（DeepSleep mode）时，仍可使用 GPIO 的外部中断功能，当产生外部中断后，可将芯片从休眠模式或深度休眠模式唤醒回到运行模式。

注意：

同组 GPIOx.PINy 共用一个硬件滤波器时钟源选择寄存器，因此同组 GPIO 只能以相同的滤波时钟来过滤输入信号抖动。

8.3.7 其他功能

原子位操作

GPIO 控制器支持位置位、位清零和位翻转功能。

向 GPIO 位置位清零寄存器 GPIOx_BSRR[y] 或位清零寄存器 GPIOx_BRR[y] 写入 1，将直接改变输出数据寄存器 GPIOx_ODR 的对应位的状态，从而间接影响最终的输出电平，但不会影响该寄存器其它位的状态。

向 GPIO 位翻转寄存器 GPIOx_TOG[y] 写入 1，将使输出端口的电平状态发生翻转。

端口复位状态

上电或复位后，SWCLK（PA14）和 SWDIO（PA13）默认为数字上拉。其他端口默认为模拟高阻输入（high resistance input），上拉默认不打开。

8.4 编程示例

在配置 GPIO 端口时，必须先设置 SYSCTRL_AHBEN.GPIOx 为 1，使能对应的 GPIO 配置时钟及工作时钟。

8.4.1 数字输出编程示例

步骤 1：设置 GPIOx_ANALOG.PINy 为 0，将端口配置为数字功能；
步骤 2：设置 GPIOx_DIR.PINy 为 0，将端口配置成输出；
步骤 3：配置 GPIOx_OPENDRAIN 寄存器，设置端口输出模式；
步骤 4：配置 GPIOx_ODR 寄存器，设置端口输出电平。

8.4.2 数字输入编程示例

步骤 1：设置 GPIOx_ANALOG.PINy 为 0，将端口配置为数字功能；
步骤 2：设置 GPIOx_DIR.PINy 为 1，将端口配置成输入；
步骤 3：配置 GPIOx_PUR 寄存器，选择是否使能内部上拉电阻；
步骤 4：读取 GPIOx_IDR 寄存器，读出端口输入电平。

8.4.3 模拟功能编程示例

步骤 1：设置 GPIOx_ANALOG.PINy 为 1，将端口配置为模拟功能。

8.4.4 复用功能编程示例

步骤 1：根据应用需求将端口配置成数字输出或数字输入，具体寄存器配置步骤请参见 8.4.1 数字输出编程示例、8.4.2 数字输入编程示例；
步骤 2：配置 GPIOx_AFRH 或 GPIOx_AFRL 寄存器，设置端口复用功能，请参见表 8-2 GPIO 复用功能分配表。

8.4.5 中断功能编程示例

步骤 1：将端口配置成数字输入，具体寄存器配置步骤请参见 8.4.2 数字输入编程示例；
步骤 2：配置 GPIOx_FILTER.FLTCLK，选择端口中断滤波时钟；
步骤 3：设置 GPIOx_FILTER.PINy 为 1，使能相应端口滤波时钟；
步骤 4：配置 NVIC 控制器，请参见 5 中断章节；
步骤 5：根据应用需求，配置 GPIOx_RISEIE、GPIOx_FALLIE 寄存器，选择 GPIO 中断触发方式；
步骤 6：端口中断输入信号触发 GPIO 中断，执行中断服务函数。

8.5 寄存器列表

GPIOA 基地址：GPIOA_BASE = 0x4800 0000

GPIOB 基地址：GPIOB_BASE = 0x4800 0100

GPIOC 基地址：GPIOC_BASE = 0x4800 0200

表 8-4 GPIO 寄存器列表

寄存器名称	寄存器地址	寄存器描述
GPIOx_DIR	GPIOx_BASE + 0x00	GPIOx 输入输出方向寄存器
GPIOx_OPENDRAIN	GPIOx_BASE + 0x04	GPIOx 输出模式寄存器
GPIOx_PUR	GPIOx_BASE + 0x10	GPIOx 上拉电阻寄存器
GPIOx_AFRH	GPIOx_BASE + 0x14	GPIOx 复用功能寄存器高段
GPIOx_AFRL	GPIOx_BASE + 0x18	GPIOx 复用功能寄存器低段
GPIOx_ANALOG	GPIOx_BASE + 0x1C	GPIOx 模拟数字配置寄存器
GPIOx_RISEIE	GPIOx_BASE + 0x24	GPIOx 上升沿中断使能寄存器
GPIOx_FALLIE	GPIOx_BASE + 0x28	GPIOx 下降沿中断使能寄存器
GPIOx_ISR	GPIOx_BASE + 0x34	GPIOx 中断标志寄存器
GPIOx_ICR	GPIOx_BASE + 0x38	GPIOx 中断标志清除寄存器
GPIOx_FILTER	GPIOx_BASE + 0x40	GPIOx 中断数字滤波器配置寄存器
GPIOx_IDR	GPIOx_BASE + 0x50	GPIOx 输入数据寄存器
GPIOx_ODR	GPIOx_BASE + 0x54	GPIOx 输出数据寄存器
GPIOx_BRR	GPIOx_BASE + 0x58	GPIOx 位清零寄存器
GPIOx_BSRR	GPIOx_BASE + 0x5C	GPIOx 位置位清零寄存器
GPIOx_TOG	GPIOx_BASE + 0x60	GPIOx 位翻转寄存器

8.6 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

8.6.1 GPIOx_DIR GPIO 输入输出方向寄存器 (x =A, B , C)

Address offset: 0x00 Reset value: 0x0000 01FF(GPIOA)

0x0000 00FF(GPIOB)

0x0000 E000(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口输入输出方向控制 0：将端口配置成输出 1：将端口配置成输入

8.6.2 GPIOx_OPENDRAIN GPIO 输出模式寄存器 (x =A, B , C)

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口输出方式控制位 0：推挽输出 1：开漏输出

8.6.3 GPIOx_PUR GPIO 上拉电阻寄存器 (x =A, B , C)

Address offset: 0x10 Reset value: 0x0000 0000(GPIOA)

0x0000 0080(GPIOB)

0x0000 0000(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口上拉电阻使能控制 0：禁止上拉电阻 1：使能上拉电阻

8.6.4 GPIOx_AFRH GPIO 复用功能配置寄存器高段 (x =A, B , C)

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28 27:24 23:20 19:16 15:12 11:8 7:4 3:0	AFRy y = 8 ~ 15	RW	端口数字复用功能控制 0000：GPIO 0001：AF1 0010：AF2 0011：AF3 0100：AF4 0101：AF5 0110：AF6 0111：AF7

8.6.5 GPIOx_AFRL GPIO 复用功能配置寄存器低段 (x =A, B , C)

Address offset: 0x18 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28 27:24 23:20 19:16 15:12 11:8 7:4 3:0	AFRy y = 0 ~ 7	RW	端口数字复用功能控制 0000：GPIO 0001：AF1 0010：AF2 0011：AF3 0100：AF4 0101：AF5 0110：AF6 0111：AF7

8.6.6 GPIOx_ANALOG GPIO 模拟数字配置寄存器 (x =A, B , C)

Address offset: 0x1C Reset value: 0x0000 01FF(GPIOA)

0x0000 007F(GPIOB)

0x0000 E000(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口模拟 / 数字功能配置 0：将端口配置为数字功能 1：将端口配置为模拟功能

8.6.7 GPIOx_RISEIE GPIO 上升沿中断使能寄存器 (x =A, B , C)

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口上升沿中断使能控制 0：禁止相应端口的上升沿中断 1：使能相应端口的上升沿中断

8.6.8 GPIOx_FALLIE GPIO 下降沿中断使能寄存器 (x =A, B , C)

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	端口下降沿中断使能控制 0：禁止相应端口的下降沿中断 1：使能相应端口的下降沿中断

8.6.9 GPIOx_ISR GPIO 中断标志寄存器 (x =A, B , C)

Address offset: 0x34 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RO	端口中断状态标志 0：未检测到已使能的中断 1：已检测到已使能的中断

8.6.10 GPIOx_ICR GPIO 中断标志清除寄存器 (x =A, B , C)

Address offset: 0x38 Reset value: 0x0000 01FF(GPIOA)

0x0000 00FF(GPIOB)

0x0000 E000(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	R1W0	端口中断状态标志清除 W0：清除相应的中断标志位 W1：无功能

8.6.11 GPIOx_FILTER GPIO 中断数字滤波器配置寄存器 (x =A, B , C)

Address offset: 0x40 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18:16	FLTCLK	RW	端口中断滤波时钟选择 000：HCLK / 2 001：HCLK / 4 010：HCLK / 8 011：BTIM1 溢出 101：LSI（约 32.8kHz） 111：LPTIM 溢出
15:0	PINy y=0 ~ 15	RW	端口滤波时钟使能控制 0：禁止相应端口滤波时钟 1：使能相应端口滤波时钟 *注：滤除宽度小于 FLTCLK 时钟宽度的脉冲。*

注：

若需硬件滤波功能在 DeepSleep 模式下正常工作，请设置滤波时钟为 LSI 或 LPTIM 的溢出信号。

8.6.12 GPIOx_IDR GPIO 输入数据寄存器 (x =A, B , C)

Address offset: 0x50 Reset value: 0x0000 0180(GPIOA)

0x0000 0080(GPIOB)

0x0000 0000(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RO	读出端口输入电平状态 0：端口为低电平 1：端口为高电平

注：支持 8bit 读。

8.6.13 GPIOx_ODR GPIO 输出数据寄存器 (x =A, B , C)

Address offset: 0x54 Reset value: 0x---- ----

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	RW	设置端口输出电平 0：设置端口输出低电平 1：设置端口输出高电平

注：支持 8bit 读写。

8.6.14 GPIOx_BRR GPIO 端口位清零寄存器 (x =A, B , C)

Address offset: 0x58 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	BRRy y=0 ~ 15	R0W1	端口位清零控制 0：不影响 GPIOx_ODR 寄存器相应的比特 1：设置 GPIOx_ODR 寄存器相应的比特为 0

8.6.15 GPIOx_BSRR GPIO 端口位置位清零寄存器 (x =A, B , C)

Address offset: 0x5C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	BRRy y=0 ~ 15	R0W1	端口位清零控制 0：不影响 GPIOx_ODR 寄存器相应的比特 1：设置 GPIOx_ODR 寄存器相应的比特为 0
15:0	BSSy y=0 ~ 15	R0W1	端口位置位控制 0：不影响 GPIOx_ODR 寄存器相应的比特 1：设置 GPIOx_ODR 寄存器相应的比特为 1 *注：当 BRRy 与 BSSy 同时置 1 时，BSSy 具有更高优先级。*

8.6.16 GPIOx_TOG GPIO 端口位翻转寄存器 (x =A, B , C)

Address offset: 0x60 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PINy y=0 ~ 15	R0W1	端口位翻转控制 0：不影响 GPIOx_ODR 寄存器相应的比特 1：对 GPIOx_ODR 寄存器相应的比特取反

9 循环冗余校验（CRC）

9.1 概述

循环冗余校验 (CRC) 主要应用于核实数据传输或数据存储的正确性和完整性。CW32L011 内部集成 CRC 计算单元，支持采用多种 CRC 算法对输入数据进行 CRC 计算。

9.2 主要特性

- 8bit 输入数据位宽
- 2 种多项式
  - CRC-16 多项式 1：$x^{16} + x^{15} + x^2 + 1$
  - CRC-16 多项式 2：$x^{16} + x^{12} + x^5 + 1$
- 8 种常用的算法

  基于多项式，初始值，结果异或值，输入 / 输出反转的组合。

9.3 功能描述

CRC 单元通过对输入数据 (或输入数据的反转) 和选定的多项式值进行‘除’运算，得到的余数再进行反转或者不反转，以及异或处理，得到 CRC 计算结果。

CRC 单元在使用之前，需要设置 SYSCTRL_AHBEN.CRC 为 1，打开 CRC 单元的配置时钟及工作时钟，一般在系统初始化时进行设置。

9.3.1 算法模式

CW32L011 的 CRC 单元支持多种算法模式。不同的 CRC 算法，对应的多项式、初始值、输入数据反转、输出数据反转、结果异或值等参数不同，如下表所示：

表 9-1 CRC 算法模式

算法名称	多项式值	初始值	输入反转	输出反转	结果异或值
CRC16_IBM	0x8005	0x0000	True	True	0x0000
CRC16_MAXIM	0x8005	0x0000	True	True	0xFFFF
CRC16_USB	0x8005	0xFFFF	True	True	0xFFFF
CRC16_MODBUS	0x8005	0xFFFF	True	True	0x0000
CRC16_CCITT	0x1021	0x0000	True	True	0x0000
CRC16_CCITT_False	0x1021	0xFFFF	False	False	0x0000
CRC16_X25	0x1021	0xFFFF	True	True	0xFFFF
CRC16_XMODEM	0x1021	0x0000	False	False	0x0000

各参数含义如下：

- 多项式值

  多项式是码组的描述，如 CRC-16 多项式 2：$x^{16} + x^{12} + x^5 + 1$，对应的码组是 1 0001 0000 0010 0001。因为多项式码组的最高位固定为 1，且最高位的位置已知，因此一般将最高位 1 去掉后的码组称为多项式值，如 CRC-16 多项式 2 的值为 0001 0000 0010 0001，即 0x1021。

- 初始值

  在计算 CRC 校验值之前，CRC 寄存器的初始值。

- 输入数据反转

  即在计算开始前，将需要计算 CRC 校验值的数据进行高低序位反转，如数据位 1011，反转后为 1101。

- 输出数据反转

  在 CRC 计算结束后，与结果异或值进行异或之前，将计算值进行高低序位反转，如计算结果为 1011，反转后为 1101。

- 结果异或值

  在 CRC 计算结束后，得到的 CRC 计算值与结果异或值进行异或操作，就得到了最终的 CRC 校验值。

9.3.2 输入数据位宽

CRC 计算单元支持 8bit 输入数据位宽，输入数据时，先输入低字节再输入高字节。

例如，用户需要计算 0x00、0x11、0x22、0x33、0x44、0x55、0x66、0x77 这一组数据的 CRC 校验值，写入顺序为：0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77。

代码示例：

```
CW_CRC -> DR = 0x00 ;
CW_CRC -> DR = 0x11 ;
CW_CRC -> DR = 0x22 ;
CW_CRC -> DR = 0x33 ;
CW_CRC -> DR = 0x44 ;
CW_CRC -> DR = 0x55 ;
CW_CRC -> DR = 0x66 ;
CW_CRC -> DR = 0x77 ;
```

9.4 编程示例

9.4.1 CRC16_CCITT 算法模式

步骤 1：设置 CRC_CR.MODE 为 0x04，选择 CRC16_CCITT 算法模式，硬件自动配置 CRC 寄存器初始值为 0x0000；

步骤 2：将待编码的原始数据依次写入数据寄存器 CRC_DR ，写入方式请参见 9.3.2 输入数据位宽中的代码示例；

步骤 3：读取 CRC_RESULT 获取 CRC 校验值。

代码示例：

```
tempdata = CW_CRC -> RESULT ;
```

9.5 寄存器列表

CRC 基地址：CRC_BASE = 0x4002 3000

表 9-2 CRC 寄存器列表

寄存器名称	寄存器地址	寄存器描述
CRC_CR	CRC_BASE + 0x00	控制寄存器
CRC_DR	CRC_BASE + 0x08	数据寄存器
CRC_RESULT	CRC_BASE + 0x0C	结果寄存器

9.6 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

9.6.1 CRC_CR 控制寄存器

Address offset: 0x00 Reset value: 0x0000 0004

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3:0	MODE	RW	CRC 算法模式配置 0000：CRC16_IBM 0001：CRC16_MAXIM 0010：CRC16_USB 0011：CRC16_MODBUS 0100：CRC16_CCITT 0101：CRC16_CCITT_False 0110：CRC16_X25 0111：CRC16_XMODEM

9.6.2 CRC_DR 数据寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	DR	RW	数据写入寄存器，位宽 8bit

9.6.3 CRC_RESULT 结果寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	CRC 计算结果

10 实时时钟（RTC）

10.1 概述

实时时钟（RTC）是一个专用的计数器 / 定时器，可提供日历信息，包括小时、分钟、秒、亚秒、日、月份、年份以及星期。

RTC 具有两个独立闹钟，时间、日期可组合设定，可产生闹钟中断，并通过引脚输出；支持时间戳功能，可通过引脚触发，记录当前的日期和时间，同时产生时间戳中断；支持周期中断；支持自动唤醒功能，可产生中断并通过引脚输出；支持 1Hz 方波和 RTCOUT 输出功能；支持内部时钟校准补偿。

CW32L011 内置经独立校准的 32kHz 频率的 RC 时钟源，为 RTC 提供驱动时钟，RTC 可在深度休眠模式下运行，适用于要求低功耗的应用场合。

10.2 主要特性

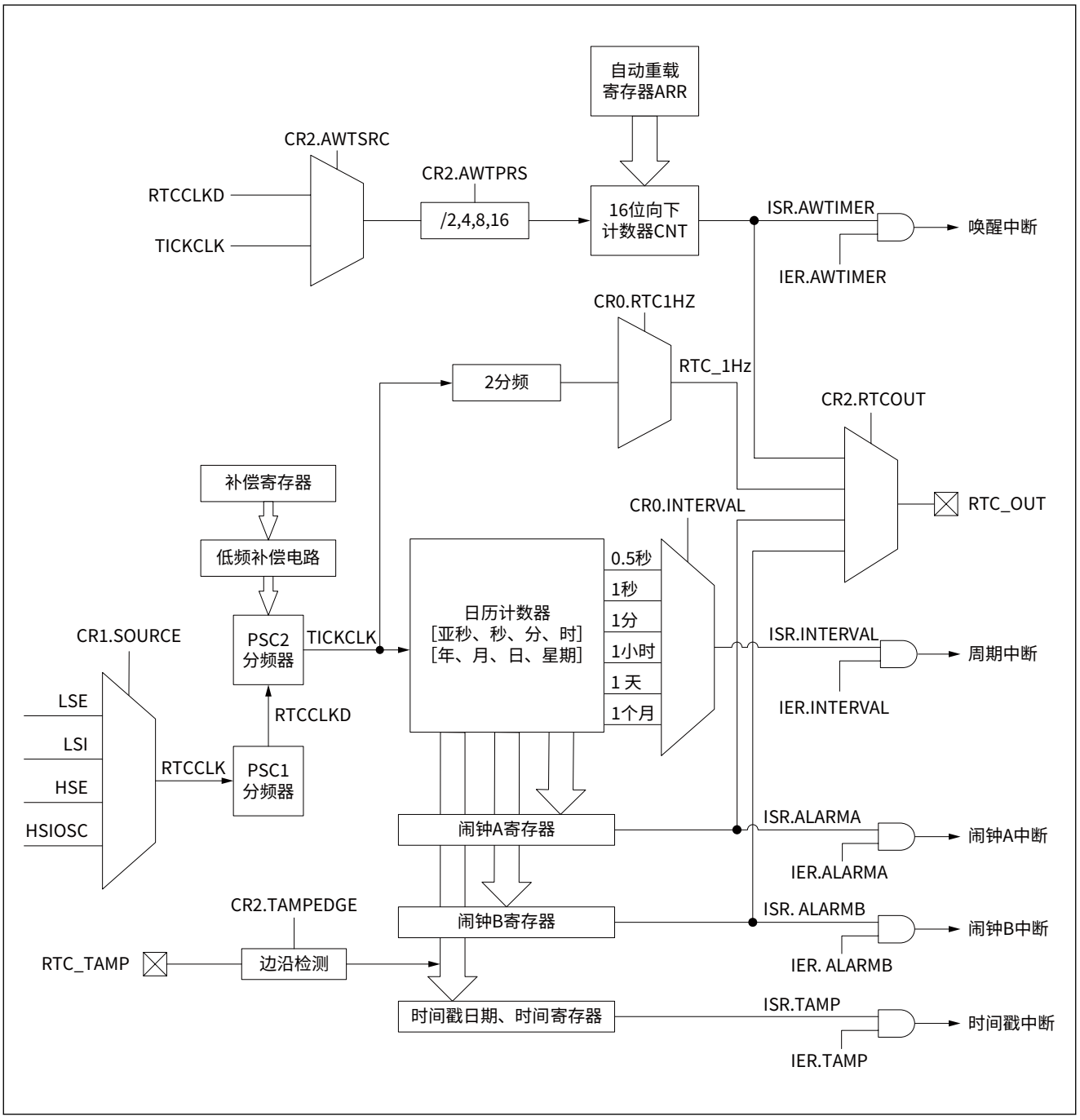
- 日历功能（BCD 码格式）：
  - 亚秒、秒、分、时（支持 12 / 24 小时制）。
  - 星期、日、月份和年份，支持自动闰年修正。
- 中断功能：
  - 闹钟 A、B 中断，可通过引脚输出。
  - 周期中断，宽范围：0.5s ~ 1 个月。
  - 自动唤醒中断，范围：61μs ~ 145h，可通过引脚输出。
  - 时间戳中断。
- 校准补偿功能：
  - 最小时钟误差补偿步长，0.119ppm。
  - 支持 1Hz 方波输出。
- 寄存器锁定保护功能，防止意外修改。
- 可工作于运行模式、休眠模式和深度休眠模式。

10.3 功能描述

10.3.1 功能框图

实时时钟 (RTC) 的功能框图如下图所示：

图 10-1 RTC 功能框图



10.3.2 时钟和预分频

实时时钟 (RTC) 主要由专用的高精度 RTC 定时器组成，时钟源 RTCCLK 可选择外部低速时钟 LSE、内部低速时钟 LSI、外部高速时钟 HSE 和内部高速时钟 HSIOSC，通过控制寄存器 RTC_CR1 的 SOURCE 位域进行选择。

预分频器分为 2 个可编程的预分频器：PSC1 和 PSC2。

经 PSC1 分频后的时钟为 RTCCLKD，可作为 RTC 自动唤醒定时器的时钟源，PSC1 最大分频后的时钟频率应小于等于 1MHz。计算公式如下：

$$f_{RTCCLKD} = f_{RTCCLK} / (PSC1 + 1)$$

经 PSC2 分频后的时钟为 TICKCLK，用于更新日历，也可作为 RTC 自动唤醒定时器的时钟源。计算公式如下：

$$f_{TICKCLK} = f_{RTCCLKD} / (PSC2 + 1)$$

用户需合理配置 PSC1 和 PSC2，以使 $f_{TICKCLK} = 2Hz$。

PSC1 默认值为 0，PSC2 默认值为 0x3FFF，故当 RTCCLK 时钟源为 LSE，即时钟频率为 32768Hz 时，计时时钟为 1s。

10.3.3 实时时钟和日历

时间寄存器 RTC_TIME 和日期寄存器 RTC_DATE，以 BCD 码格式分别记录当前的时间和日期值，在对其写入时会自动进行合法性检查，任何非法的时间或日期值将不能被写入，如 32 日、2A 时、61 秒、13 月等。

日期寄存器 RTC_DATE 中，YEAR 位域表示年，有效值 0 ~ 99；MONTH 位域表示月，有效值 1 ~ 12；DAY 位域表示日，有效值 1 ~ 31；WEEK 位域表示星期，有效值 0 ~ 6，其中 0 表示星期日，1 ~ 6 表示星期一至星期六。

时间寄存器 RTC_TIME 中，SECOND 位域表示秒，有效值 0 ~ 59；MINUTE 位域表示分，有效值 0 ~ 59；HOUR 位域代表小时，有效值为 1 ~ 12 或 0 ~ 23，可选择 12 小时制或 24 小时制。控制寄存器 RTC_CR0 的 H24 位域选择 12 或 24 小时制：

- H24 为 ‘1’ 时，选择 24 时制。
- H24 为 ‘0’ 时，选择 12 时制，HOUR 位域的最高位代表 AM/PM（上午 / 下午）：
  - ‘0’ 表示 AM。
  - ‘1’ 表示 PM。

HOUR 位域值详细见下表：

表 10-1 12/24 小时制时 HOUR 位域值

12 时制（RTC_CR0.H24=0）				24 时制（RTC_CR0.H24=1）			
时间	HOUR 位域	时间	HOUR 位域	时间	HOUR 位域	时间	HOUR 位域
AM12 时	0x12	PM12 时	0x32	00 时	0x00	12 时	0x12
AM1 时	0x01	PM1 时	0x21	01 时	0x01	13 时	0x13
AM2 时	0x02	PM2 时	0x22	02 时	0x02	14 时	0x14
AM3 时	0x03	PM3 时	0x23	03 时	0x03	15 时	0x15
AM4 时	0x04	PM4 时	0x24	04 时	0x04	16 时	0x16
AM5 时	0x05	PM5 时	0x25	05 时	0x05	17 时	0x17
AM6 时	0x06	PM6 时	0x26	06 时	0x06	18 时	0x18
AM7 时	0x07	PM7 时	0x27	07 时	0x07	19 时	0x19
AM8 时	0x08	PM8 时	0x28	08 时	0x08	20 时	0x20
AM9 时	0x09	PM9 时	0x29	09 时	0x09	21 时	0x21
AM10 时	0x10	PM10 时	0x30	10 时	0x10	22 时	0x22
AM11 时	0x11	PM11 时	0x31	11 时	0x11	23 时	0x23

10.3.4 RTC 初始化设置

标准的 RTC 模块初始化过程，应包括以下步骤：

步骤 1：向 RTC_KEY 寄存器顺序写入 0xCA、0x53，解除 RTC 寄存器锁定；

步骤 2：配置 RTC_CR0.H24 位域，选择 12/24 小时制；

步骤 3：配置 RTC_CR1.SOURCE 位域，选择 RTC 时钟源；

注：

如选择 LSE 或 LSI，需要先使能和启动 LSE 或 LSI，并等待时钟稳定。

步骤 4：设置正确的时间和日期值，写入时间和日期寄存器；

步骤 5：配置需要的周期中断单元、自动唤醒单元、闹钟 A、闹钟 B；

步骤 6：配置 RTC_IER 寄存器，设置周期中断、自动唤醒中断、闹钟 A、闹钟 B 中断；

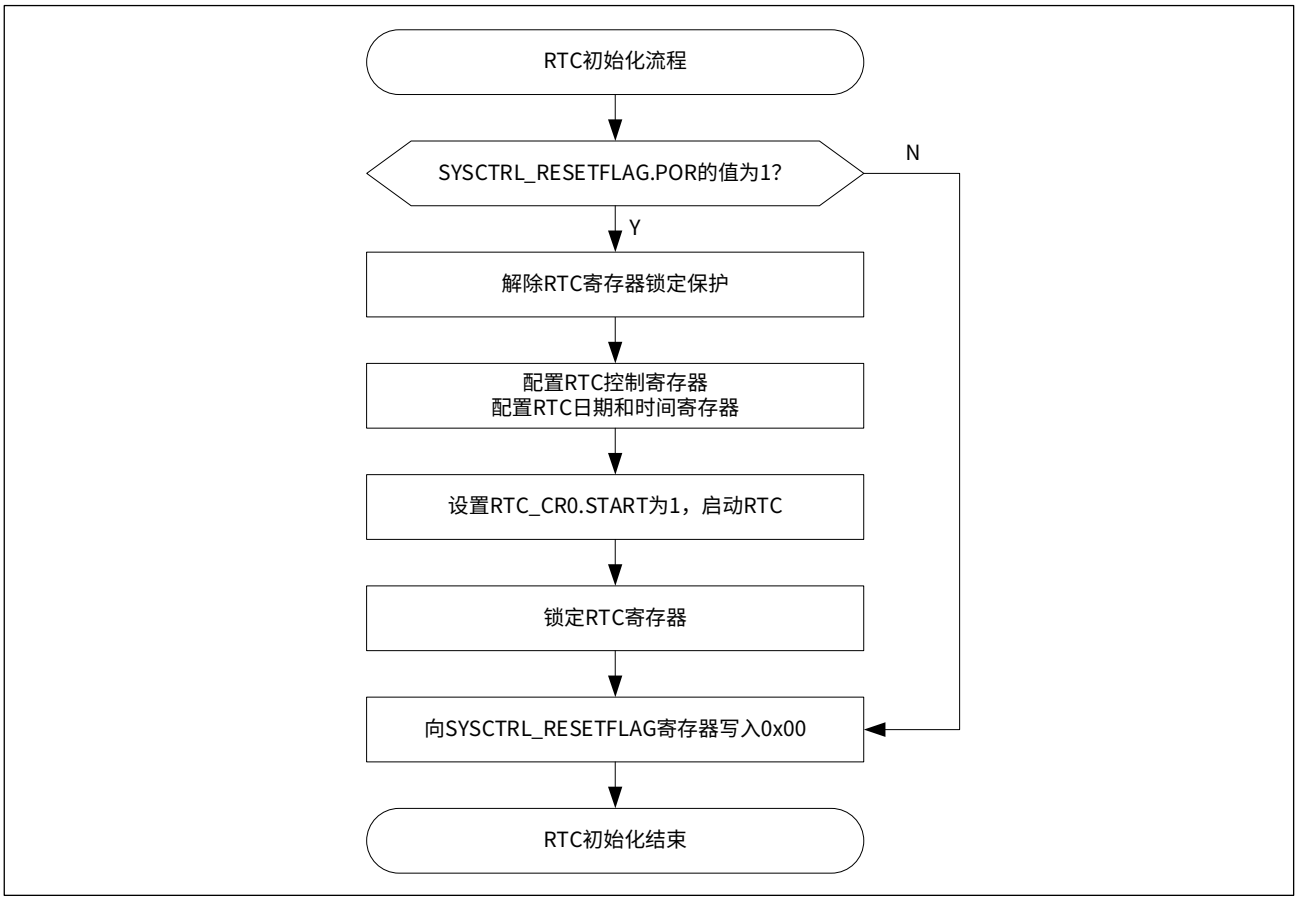
步骤 7：配置时间戳功能，配置时间戳中断；

步骤 8：RTC_CR0.START 位域置 1，启动 RTC 定时器；

步骤 9：向 RTC_KEY 寄存器顺序写入 0xCA、0x--，启动 RTC 寄存器锁定保护。

用户应用中，在对 RTC 进行初始化时，应先检查 MCU 的复位状态（SYSCTRL_RESETFLAG）。如果是上电复位（SYSCTRL_RESETFLAG.POR=1），RTC 的寄存器会被复位到默认值，需要重新配置 RTC 控制寄存器、RTC 日期和时间寄存器，并重新启动 RTC；其它复位条件不会复位 RTC 的寄存器，RTC 持续正常计时。

图 10-2 RTC 初始化流程



10.3.5 寄存器锁定功能

CW32L011 上电复位后，除 RTC_KEY 和 RTC_ICR 之外的 RTC 寄存器都处于锁定保护状态，不可修改。向键值寄存器 RTC_KEY 依次写入 0xCA、0x53 关键字，解除 RTC 寄存器锁定保护；向 RTC_KEY 寄存器依次写入 0xCA、0x--，RTC 寄存器将被锁定。

10.3.6 寄存器访问操作

RTC 没有运行时，解除寄存器锁定后，所有寄存器可直接读写。

RTC 运行时，解除寄存器锁定后，除计时相关寄存器（RTC_DATE、RTC_TIME、RTC_AWTARR）以外的寄存器，均可直接读写。

RTC 运行时，解除寄存器锁定后，由于 RTC 计数器的运行时钟可能与 RTC 控制寄存器的时钟不同步，为保证能正确访问 RTC_DATE、RTC_TIME 和 RTC_AWTARR 寄存器内容，同时不影响 RTC 的运行，需遵守以下操作规范：

步骤 1：查询等待 RTC_CR1.WAIT 为 0，表示装载已完成，可以读写；

步骤 2：向 RTC_KEY 寄存器顺序写入 0xCA、0x53，解除 RTC 寄存器锁定；

步骤 3：执行正常 RTC 寄存器访问操作；

步骤 4：查询等待 RTC_CR1.WAIT 为 0，表示装载已完成；

步骤 5：向 RTC_KEY 寄存器顺序写入 0xCA、0x--，启动 RTC 寄存器写保护。

注：

如用户需要更快速的读取 RTC 时间与日期信息，可选择连续读两次时间或日期寄存器，只要两次读出内容相同，即可认为读出的数据有效。

RTC 运行时，由于 RTC 模块的计数时钟与寄存器的读写时钟不同步，读取亚秒计数值寄存器 RTC_SSCNT 需要按照特定的操作步骤才能确保读出正确的亚秒值。读取亚秒时刻示例如下所示：

步骤 1：从 RTC_SSCNT 寄存器读出亚秒计数值，存入变量 SS1；

步骤 2：再次从 RTC_SSCNT 寄存器读出亚秒计数值，存入变量 SS2；

步骤 3：如果 SS1 与 SS2 不相等，则跳转到步骤 1；

步骤 4：如果 SS1[19:0] 为 0，则跳转到步骤 1。

10.3.7 RTCOUT 输出

通过设置控制寄存器 RTC_CR2 的 RTCOUT 位域，可选择 RTC_OUT 引脚的输出源，如下表所示：

表 10-2 RTC_OUT 输出配置

RTC_CR2.RTCOUT	RTC_OUT 输出
00	RTC_1Hz 信号
01	闹钟 A 匹配标志
10	闹钟 B 匹配标志
11	唤醒定时器溢出标志

10.3.8 1Hz 信号输出

通过控制寄存器 RTC_CR0 的 RTC1HZ 位域配置 RTC_1Hz 输出信号，如下表所示：

表 10-3 RTC_1Hz 输出配置

	RTC_CR0.RTC1HZ	RTC_1Hz 输出
关闭时钟误差补偿（RTC_COMPCFR1.EN=0）	01	未经时钟补偿的 TICKCLK 二分频所得到的 1Hz 信号
开启时钟误差补偿（RTC_COMPCFR1.EN=1）	01	经过时钟补偿的 TICKCLK 二分频所得到的 1Hz 信号

当设置 RTC_COMPCFR1.EN 为 0 且 RTC_CR0.RTC1HZ 为 0x1 时，RTC_1Hz 输出的信号为未经时钟补偿的 TICKCLK 的 2 分频，该信号与 RTC 的秒计数时钟完全相同。该时钟信号每一个周期的持续时间均完全相同。

当设置 RTC_COMPCFR1.EN 为 1 且 RTC_CR0.RTC1HZ 为 0x1 时，RTC_1Hz 输出的信号为经时钟补偿的 TICKCLK 的 2 分频，该信号与 RTC 的秒计数时钟完全相同。该时钟信号每一个周期包含的 RTCCLKD 时钟的数量为表达式 (PSC2+1)×2±COMP/N 的值邻近的两个整数，参见 10.3.9 时钟误差补偿。

10.3.9 时钟误差补偿

RTC 内置时钟补偿电路，当 RTCCLK 精度不能满足需求时，可对 RTC 时钟进行补偿，以实现更高精度的 RTC 定时。

RTC 时钟误差补偿配置寄存器 RTC_COMPCFR1 有三个位域，用于实现误差补偿：

- SIGN，补偿方向：
  - TICKCLK 时钟频率大于 2Hz 时，SIGN 设为 0，增加计数值，向下补偿。
  - TICKCLK 时钟频率小于 2Hz 时，SIGN 设为 1，减小计数值，向上补偿。
- PERIOD，补偿周期，可选择 32、128、256 秒为一个补偿周期：
  - 设置 PERIOD 为 00，选择 32 秒为一个补偿周期，可实现快速校准但是精度较低，补偿步长为 0.950ppm，补偿精度为 0.475ppm。
  - 设置 PERIOD 为 01，选择 128 秒为一个补偿周期，补偿步长是 0.238ppm，补偿精度是 0.119ppm。
  - 设置 PERIOD 为 10，选择 256 秒为一个补偿周期，调整过程最平滑，校准精度最高，补偿步长是 0.119ppm，补偿精度是 0.060ppm。
- COMP，补偿数值，时钟可补偿范围为 ±488.0ppm：
  - 当补偿周期选择 32 秒时，COMP 有效范围是 1 ~ 511。
  - 当补偿周期选择 128 秒时，COMP 有效范围是 1 ~ 2047。
  - 当补偿周期选择 256 秒时，COMP 有效范围是 1 ~ 4095。

时钟校准操作步骤如下：

步骤 1：关闭误差补偿功能，设置 RTC_COMPCFR1.EN 为 0；

步骤 2：根据表 8-2 GPIO 复用功能分配表配置某个 GPIO 端口为 RTC_OUT 输出，并设置 RTC_CR2.RTCOUT 为 0，输出未补偿的 1Hz 信号；

步骤 3：依 10.3.4 RTC 初始化设置，配置 LSE 为 RTC 时钟源；

步骤 4：RTC_CR0.START 位域置 1，启动 RTC；

步骤 5：使用高精度频率计对 RTC_OUT 引脚输出的 1Hz 信号进行测量，记录实际测量频率 f_0；

补偿值 COMP 的计算方法：

假设测量值 f_0= 0.9999888Hz，PSC2=0x3FFF，则 RTCCLKD 时钟实际频率 f_z：

$$f_z = (PSC2+1) \times 2 \times 0.9999888 \approx 32767.63Hz$$

频率误差 $\Delta f = (PSC2+1) \times 2 - f_z = 32768 - 32767.63 = 0.37$

选择补偿周期为 256 秒，校准精度最高，计算出补偿数值：

$$COMP = \Delta f \times PERIOD = 0.37 \times 256 = 94.72 \approx 95 = 0x5F$$

步骤 6：按 10.3.6 寄存器访问操作，配置 RTC_COMPCFR1 寄存器，此例中 SIGN 位域写 1，PERIOD 位域写 10，COMP 位域为 0x5F；

步骤 7：写 RTC_COMPCFR1.EN 为 1，启动时钟误差补偿。

注：

步骤 6 和 7 也可合并，直接按半字操作，向 RTC_COMPCFR1 写入 0xE05F。

10.3.10 闹钟 A 和闹钟 B

RTC 支持 2 个独立闹钟（闹钟 A 和闹钟 B），可在一周内任意时刻产生闹钟事件，并产生闹钟中断，同时将闹钟匹配事件通过外部 RTC_OUT 引脚输出。设置控制寄存器 RTC_CR2 的 ALARMAEN 和 ALARMBEN 位域为 1，可分别单独使能闹钟 A 和闹钟 B。

通过设置闹钟 A、B 控制寄存器（RTC_ALARMA 和 RTC_ALARMB）的时、分、秒匹配控制位 HOUREN、MINUTEEN、SECONDEN 和时、分、秒计数值 HOUR、MINUTE、SECOND，可设定闹钟在‘xx 时 xx 分 xx 秒’，或‘xx 分 xx 秒’或‘xx 时 xx 分’或‘xx 时’等多种组合产生闹钟事件；闹钟星期使能控制位 WEEKMASK，可选择一周中的任意一天产生闹钟事件，bit0 代表星期日，bit1 ~ 6 代表星期一至星期六。

采用 12 或 24 小时制，闹钟控制寄存器 RTC_ALARMx（x=A、B）的设置值可能不同，示例如下表：

表 10-4 定时闹钟举例

定时要求	时制	RTC_ALARMx 值	闹钟时刻
每天早上 6：30 闹钟	12/24 时制	0x7F06 3000	06：30：00
每工作日 7：00 闹钟	12/24 时制	0x3E07 0000	07：00：00
每小时的 50 分发生一次	12/24 时制	0x7F80 5000	xx：50：00
每分钟的 10 秒发生一次	12/24 时制	0x7F80 8010	xx：xx：10
每周一，中午 12 点每分钟的 5 秒发生一次	12 时制	0x0232 8005	12PM：xx：05
每周一，中午 12 点每分钟的 5 秒发生一次	24 时制	0x0212 8005	12：xx：05
每周日，晚上 12 点 30 分每秒钟发生一次	12 时制	0x0112 3080	12AM：30：xx
每周日，晚上 0 点 30 分每秒钟发生一次	24 时制	0x0100 3080	00：30：xx

当发生闹钟事件时，对应的闹钟匹配标志位 RTC_ISR.ALARMx 会被硬件置 1，如果设置了闹钟中断使能位 RTC_IER.ALARMx 为 1，将产生中断请求。

闹钟 A 或闹钟 B 的匹配标志可通过 RTC_OUT 引脚输出，请参见 10.3.7 RTCOUT 输出。

10.3.11 周期中断功能

RTC 内置周期中断模块，可产生固定周期的中断信号。定时周期通过控制寄存器 RTC_CR0 的 INTERVAL 位域来配置，如下表所示：

表 10-5 定时周期配置

RTC_CR0. INTERVAL	周期中断的周期
000	不产生周期中断
001	0.5 秒，每 1 秒和 0.5 秒时刻产生
010	1 秒，秒计数器发生变化时产生
011	1 分钟，分计数器发生变化时产生
100	1 小时，时计数器发生变化时产生
101	1 天，日计数器发生变化时产生
11x	1 个月，月计数器发生变化时产生

通过 RTC 中断使能寄存器 RTC_IER 的 INTERVAL 位域，可设置周期中断是否产生中断请求。

10.3.12 自动唤醒功能

自动唤醒定时器是一个 16 位可编程自动重载减法计数器，可选择时钟源，定时范围为：61μs ~ 145h。当计数器溢出时，可产生自动唤醒中断，并将溢出标志通过 RTC_OUT 引脚输出。

设置控制寄存器 RTC_CR2 的 AWTEN 位域为 1 使能自动唤醒功能，该功能专为低功耗应用场合而设计，可工作于 MCU 的全部工作模式。

RTC 控制寄存器 RTC_CR2 的 AWTSRC 和 AWTPRS 位域，用于选择自动唤醒定时器的计数时钟源。当设置 AWTSRC 为 0 时，选择 RTCCLKD 作为计数时钟来源；当设置 AWTSRC 为 1 时，选择 TICKCLK 作为计数时钟来源，此时必须使能 RTC。通过 AWTPRS 位域可对计数时钟来源进行分频，可选分频系数为 2、4、8、16。

自动唤醒定时器计数周期由计数时钟源和重载寄存器 RTC_AWTARR 决定，定时时长计算公式为：

自动唤醒定时器定时周期 =（RTC_AWTARR+1）/ 唤醒定时器时钟频率

最短定时：(0+1) / 16384Hz = 61μs

最长定时：(65535+1) / 0.125Hz = 524288s = 8738min ≈ 145.63h

通过 RTC 中断使能寄存器 RTC_IER 的 AWTIMER 位域，可选择自动唤醒定时器溢出时是否产生中断请求。自动唤醒定时器的溢出标志可通过 RTC_OUT 引脚输出，请参见 10.3.7 RTCOUT 输出。

用户可随时读取唤醒定时器当前计数值，读取 RTC_AWTCNT 寄存器时应读取多遍，直到读到两次相同的值。

10.3.13 时间戳功能

RTC 支持时间戳功能，即通过 RTC_TAMP 引脚触发，将当前时间和日期分别保存到时间戳日期寄存器 RTC_TAMPDATE 和时间戳时间寄存器 RTC_TAMPTIM，同时可产生时间戳中断。

控制寄存器 RTC_CR2 的 TAMPEDGE 位域用来选择触发时间戳的信号是上升沿还是下降沿有效，RTC_CR2 寄存器的 TAMPEN 位域用于使能时间戳功能。

用户可灵活选择触发引脚 RTC_TAMP，并需配置该引脚为数字输入和复用功能，具体 RTC_TAMP 引脚请参考数据手册引脚定义。

当发生时间戳事件时，时间戳事件标志位 RTC_ISR.TAMP 会被置 1，如果设置了时间戳中断使能位 RTC_IER.TAMP 为 1，将产生中断请求。

如果发生第一次时间戳事件后，未软件清除 RTC_ISR.TAMP 标志位，又产生了第二次时间戳事件，时间戳溢出标志位 RTC_ISR.TAMPOV 会被置 1，如果设置了时间戳溢出中断使能位 RTC_IER.TAMPOV 为 1，将产生中断请求。

10.3.14 RTC 中断

RTC 支持 6 个中断源：唤醒中断、闹钟 A 中断、闹钟 B 中断、周期中断、时间戳中断和时间戳溢出中断。

中断的开启由中断使能寄存器 RTC_IER 控制，对应位域写入‘1’允许中断，写‘0’禁止中断；中断发生时会硬件自动置位中断标志寄存器 RTC_ISR 的对应位；通过向中断标志清除寄存器 RTC_ICR 的对应位写入‘0’，可清除相应的中断标志。

用户应用程序中，启用 RTC 中断功能后，所有 RTC 中断事件发生时都会从 RTC 中断入口（RTC_IRQHandler）调用同一个中断服务程序，用户应当在中断服务程序中查询 RTC 中断标志寄存器 RTC_ISR，以确定引发中断的具体 RTC 中断源。

在中断事件处理完成后，退出中断服务程序之前，必须向 RTC 中断标志清除寄存器 RTC_ICR 的对应位写入‘0’，以清除中断标志，避免重复进入中断服务程序。

10.4 寄存器列表

RTC 基地址：RTC_BASE = 0x4000 4400

表 10-6 RTC 寄存器列表

寄存器名称	寄存器地址	寄存器描述
RTC_KEY	RTC_BASE + 0x00	键值寄存器
RTC_CR0	RTC_BASE + 0x04	控制寄存器 0
RTC_CR1	RTC_BASE + 0x08	控制寄存器 1
RTC_CR2	RTC_BASE + 0x0C	控制寄存器 2
RTC_DATE	RTC_BASE + 0x14	日期寄存器
RTC_TIME	RTC_BASE + 0x18	时间寄存器
RTC_ALARMA	RTC_BASE + 0x1C	闹钟 A 控制寄存器
RTC_ALARMB	RTC_BASE + 0x20	闹钟 B 控制寄存器
RTC_TAMPDATE	RTC_BASE + 0x24	时间戳日期寄存器
RTC_TAMPTIME	RTC_BASE + 0x28	时间戳时间寄存器
RTC_IER	RTC_BASE + 0x30	中断使能寄存器
RTC_ISR	RTC_BASE + 0x34	中断标志寄存器
RTC_ICR	RTC_BASE + 0x38	中断标志清除寄存器
RTC_AWTARR	RTC_BASE + 0x2C	唤醒定时器重载值寄存器
RTC_AWTCNT	RTC_BASE + 0x3C	唤醒定时器计数值寄存器
RTC_PSC	RTC_BASE + 0x40	时钟预分频寄存器
RTC_SSCNT	RTC_BASE + 0x44	亚秒计数值寄存器
RTC_COMPCFR1	RTC_BASE + 0x10	时钟误差补偿配置寄存器 1

10.5 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

10.5.1 RTC_KEY 键值寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	KEY	WO	键值寄存器 向该寄存器依次写入“0x00CA”，“0x0053”，方可修改 RTC 模块的寄存器。 向该寄存器依次写入“0x00CA”，“0x----”，RTC 模块除 KEY、ICR 以外寄存器不可修改。 *注：“0x----”为除“0x0053”之外的任意值。*

10.5.2 RTC_CR0 控制寄存器 0

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	START	RW	RTC 计时器运行控制 0：停止 RTC 计时器 1：启动 RTC 计时器
6:5	RTC1HZ	RW	RTC_1Hz 输出信号配置 01：TICKCLK 二分频所得到的 1Hz 信号 其他：保留
4	RFU	-	保留位，请保持默认值
3	H24	RW	计时时制设置 0：12 小时制 1：24 小时制
2:0	INTERVAL	RW	定时周期设置 000：不产生定时周期 001：0.5 秒，在秒计时器发生变化和 0.5 秒时刻时产生 010：1 秒钟，在秒计时器发生变化时产生 011：1 分钟，在分计时器发生变化时产生 100：1 小时，在时计时器发生变化时产生 101：1 天， 在天计时器发生变化时产生 11x：1 月， 在月计时器发生变化时产生

10.5.3 RTC_CR1 控制寄存器 1

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:11	RFU	-	保留位，请保持默认值
10:8	SOURCE	RW	计时时钟 RTCCLK 来源配置 000：外部低速时钟 LSE 001：外部高速时钟 HSE 010：内部低速时钟 LSI 011：内部高速时钟 HSIOSC
7:3	RFU	-	保留位，请保持默认值
2	WAIT	RO	RTC 计时相关寄存器[1] 装载状态 0：装载已完成，可以读写 1：装载进行中，不可读写 *注：包括 SECOND、MINUTE、HOUR、WEEK、DAY、MONTH、YEAR、AWTARR。*
1:0	RFU	-	保留位，请保持默认值

注 1：

RTC 计时相关寄存器是指 RTC_DATE、RTC_TIME 和 RTC_AWTARR 寄存器。

10.5.4 RTC_CR2 控制寄存器 2

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:11	RFU	-	保留位，请保持默认值
10	ALARMBEN	RW	闹钟 B 使能控制 0：禁止 1：使能
9	ALARMAEN	RW	闹钟 A 使能控制 0：禁止 1：使能
8	RFU	-	保留位，请保持默认值
7	AWTEN	RW	唤醒定时器使能控制 0：禁止 1：使能
6	TAMPEN	RW	时间戳使能控制 0：禁止 1：使能
5:4	RTCOUT	RW	RTC_OUT 引脚输出信号配置 00：RTC_1Hz 信号，详见 RTC_CR0.RTC1HZ 01：闹钟 A 匹配标志 10：闹钟 B 匹配标志 11：唤醒定时器溢出标志
3	TAMPEDGE	RW	时间戳信号边沿配置 0：在 RTC_TAMP 引脚的上升沿记录当前时间 1：在 RTC_TAMP 引脚的下降沿记录当前时间
2	AWTSRC	RW	唤醒定时器计数时钟来源配置 0：RTCCLKD 1：TICKCLK（需使能 RTC）
1:0	AWTPRS	RW	唤醒定时器计数时钟预分频配置 00：DIV2 01：DIV4 10：DIV8 11：DIV16

10.5.5 RTC_PSC 时钟预分频寄存器

Address offset: 0x40 Reset value: 0x0000 3FFF

位域	名称	权限	功能描述
31:28	RFU	-	保留位，请保持默认值
27:20	PSC1	RW	计时时钟预分频 1 $f_{RTCCLKD} = f_{RTCCLK} / (PSC1 + 1)$ *注：PSC1 最大分频后的时钟频率应小于等于 1MHz。*
19:0	PSC2	RW	计时时钟预分频 2 $f_{TICKCLK} = f_{RTCCLKD} / (PSC2 + 1)$

注：

用户应合理配置 PSC1 和 PSC2，以使 $f_{TICKCLK}$ = 2Hz。

10.5.6 RTC_DATE 日期寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:27	RFU	-	保留位，请保持默认值
26:24	WEEK	RW	周计数值 BCD 码格式，有效值范围为 0-6 0 代表周日，1 ~ 6 代表周一到周六
23:16	YEAR	RW	年计数值 BCD 码格式，有效值范围为 00-99
15:13	RFU	-	保留位，请保持默认值
12:8	MONTH	RW	月计数值 BCD 码格式，有效值范围为 1 ~ 12
7:6	RFU	-	保留位，请保持默认值
5:0	DAY	RW	日计数值 BCD 码格式，有效值范围为 1 ~ 28、1 ~ 29、1 ~ 30、1 ~ 31

10.5.7 RTC_TIME 时间寄存器

Address offset: 0x18 Reset value: 0x0001 0000

位域	名称	权限	功能描述
31:22	RFU	-	保留位，请保持默认值
21:16	HOUR	RW	时计数值 BCD 码格式 24 小时制，有效值范围为 0-23 12 小时制，有效值范围为 1 ~ 12(AM)、21 ~ 32(PM)，bit21 为 0 代表 AM、为 1 代表 PM
15	RFU	-	保留位，请保持默认值
14:8	MINUTE	RW	分计数值 BCD 码格式，有效值范围为 0 ~ 59
7	RFU	-	保留位，请保持默认值
6:0	SECOND	RW	秒计数值 BCD 码格式，有效值范围为 0 ~ 59

注：

写 RTC_TIME 寄存器会清零 RTC_SSCNT 寄存器当前值。

10.5.8 RTC_SSCNT 亚秒计数值寄存器

Address offset: 0x44 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:21	RFU	-	保留位，请保持默认值
20	SSCNT1	RO	亚秒计数值 0.5 秒标志 0：当前计数值小于 0.5 秒 1：当前计数值大于 0.5 秒
19:0	SSCNT0	RO	亚秒计数值不足 0.5 秒部分 亚秒计数值 = (PSC2+1)×SSCNT1 + SSCNT0

10.5.9 RTC_ALARMA 闹钟 A 控制寄存器

Address offset: 0x1C Reset value: 0x0212 1000

位域	名称	权限	功能描述
31	RFU	-	保留位，请保持默认值
30:24	WEEKMASK	RW	闹钟星期使能控制 bit0 代表周日，bit1 ~ bit6 分别代表周一到周六 相应的 bit 为 1，则使能该闹钟
23	HOUREN	RW	闹钟小时位屏蔽控制 0：该闹钟需检查小时位计数值 1：该闹钟与小时位计数值无关
22	RFU	-	保留位，请保持默认值
21:16	HOUR	RW	闹钟小时计数值，BCD 格式
15	MINUTEEN	RW	闹钟分钟位屏蔽控制 0：该闹钟需检查分钟位计数值 1：该闹钟与分钟位计数值无关
14:8	MINUTE	RW	闹钟分钟计数值，BCD 格式
7	SECONDEN	RW	闹钟秒钟位屏蔽控制 0：该闹钟需检查秒钟位计数值 1：该闹钟与秒钟位计数值无关
6:0	SECOND	RW	闹钟秒钟计数值，BCD 格式

10.5.10 RTC_ALARMB 闹钟 B 控制寄存器

Address offset: 0x20 Reset value: 0x0412 0000

位域	名称	权限	功能描述
31	RFU	-	保留位，请保持默认值
30:24	WEEKMASK	RW	闹钟星期使能控制 bit0 代表周日，bit1 ~ bit6 分别代表周一到周六 相应的 bit 为 1，则使能该闹钟
23	HOUREN	RW	闹钟小时位屏蔽控制 0：该闹钟与小时位计数值无关 1：该闹钟需检查小时位计数值
22	RFU	-	保留位，请保持默认值
21:16	HOUR	RW	闹钟小时计数值，BCD 格式
15	MINUTEEN	RW	闹钟分钟位屏蔽控制 0：该闹钟与分钟位计数值无关 1：该闹钟需检查分钟位计数值
14:8	MINUTE	RW	闹钟分钟计数值，BCD 格式
7	SECONDEN	RW	闹钟秒钟位屏蔽控制 0：该闹钟与秒钟位计数值无关 1：该闹钟需检查秒钟位计数值
6:0	SECOND	RW	闹钟秒钟计数值，BCD 格式

10.5.11 RTC_TAMPDATE 时间戳日期寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:13	WEEK	RO	发生时间戳信号时的周计数值，BCD 格式
12:8	MONTH	RO	发生时间戳信号时的月计数值，BCD 格式
7:6	RFU	-	保留位，请保持默认值
5:0	DAY	RO	发生时间戳信号时的日计数值，BCD 格式

10.5.12 RTC_TAMPTIME 时间戳时间寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:22	RFU	-	保留位，请保持默认值
21:16	HOUR	RO	发生时间戳信号时的时计数值，BCD 格式
15	RFU	-	保留位，请保持默认值
14:8	MINUTE	RO	发生时间戳信号时的分计数值，BCD 格式
7	RFU	-	保留位，请保持默认值
6:0	SECOND	RO	发生时间戳信号时的秒计数值，BCD 格式

10.5.13 RTC_IER 中断使能寄存器

Address offset: 0x30 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	INTERVAL	RW	定时周期中断使能控制 0：禁止 1：使能
5	RFU	-	保留位，请保持默认值
4	TAMPOV	RW	时间戳溢出中断使能控制 0：禁止 1：使能
3	TAMP	RW	时间戳中断使能控制 0：禁止 1：使能
2	AWTIMER	RW	唤醒定时器溢出中断使能控制 0：禁止 1：使能
1	ALARMB	RW	闹钟 B 中断使能控制 0：禁止 1：使能
0	ALARMA	RW	闹钟 A 中断使能控控制 0：禁止 1：使能

10.5.14 RTC_ISR 中断标志寄存器

Address offset: 0x34 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	INTERVAL	RO	周期性定时完成标志 0：未发生周期定时完成事件 1：已发生周期定时完成事件
5	RFU	-	保留位，请保持默认值
4	TAMPOV	RO	时间戳溢出标志 0：未发生时间戳溢出事件 1：已发生时间戳溢出事件
3	TAMP	RO	时间戳事件标志 0：未发生时间戳事件 1：已发生时间戳事件
2	AWTIMER	RO	唤醒定时器定时完成标志 0：未发生唤醒定时器定时完成事件 1：已发生唤醒定时器定时完成事件
1	ALARMB	RO	闹钟 B 匹配标志 0：未发生闹钟 B 匹配事件 1：已发生闹钟 B 匹配事件
0	ALARMA	RO	闹钟 A 匹配标志 0：未发生闹钟 A 匹配事件 1：已发生闹钟 A 匹配事件

10.5.15 RTC_ICR 中断标志清除寄存器

Address offset: 0x38 Reset value: 0x0000 007F

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	INTERVAL	R1W0	周期性定时完成标志清除 W0：清除周期性定时完成标志 W1：无功能
5	RFU	-	保留位，请保持默认值
4	TAMPOV	R1W0	时间戳溢出标志清除 W0：清除时间戳溢出标志 W1：无功能
3	TAMP	R1W0	时间戳事件标志清除 W0：清除时间戳事件标志 W1：无功能
2	AWTIMER	R1W0	唤醒定时器定时完成标志清除 W0：清除唤醒定时器定时完成标志 W1：无功能
1	ALARMB	R1W0	闹钟 B 匹配标志清除 W0：清除闹钟 B 匹配标志 W1：无功能
0	ALARMA	R1W0	闹钟 A 匹配标志清除 W0：清除闹钟 A 匹配标志 W1：无功能

10.5.16 RTC_AWTARR 唤醒定时器重载值寄存器

Address offset: 0x2C Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	ARR	RW	唤醒定时器重载值

10.5.17 RTC_AWTCNT 唤醒定时器计数值寄存器

Address offset: 0x3C Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CNT	RO	唤醒定时器计数值

注：

从本寄存器读取数据时应读取多遍，直到读到两次相同的值。

10.5.18 RTC_COMPCFR1 时钟误差补偿配置寄存器 1

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	EN	RW	时钟误差补偿使能控制 0：禁止时钟误差补偿 1：使能时钟误差补偿
14	SIGN	RW	时钟补偿方向 0：增加计数值（TICKCLK 时钟频率大于 2Hz） 1：减小计数值（TICKCLK 时钟频率小于 2Hz）
13:12	PERIOD	RW	补偿周期选择 00：32 秒，补偿值为 COMP[8:0]，补偿步长 0.950ppm 01：128 秒，补偿值为 COMP[10:0]，补偿步长为 0.238ppm 10：256 秒，补偿值为 COMP[11:0]，补偿步长为 0.119ppm *注：补偿精度为补偿步长的一半。*
11:0	COMP	RW	补偿值 可补偿范围为 ±488.0ppm

11 基本定时器（BTIM）

11.1 概述

CW32L011 内部集成 3 个基本定时器 (BTIM)，每个 BTIM 完全独立且功能完全相同，各包含一个 16bit 自动重装载计数器并由一个可编程预分频器驱动。BTIM 支持内部计数模式、外部计数模式、触发启动模式和门控计数模式 4 种工作模式，支持更新事件和触发事件发生时产生中断。不同工作模式下均可由复位输入信号控制计数器复位。

11.2 主要特性

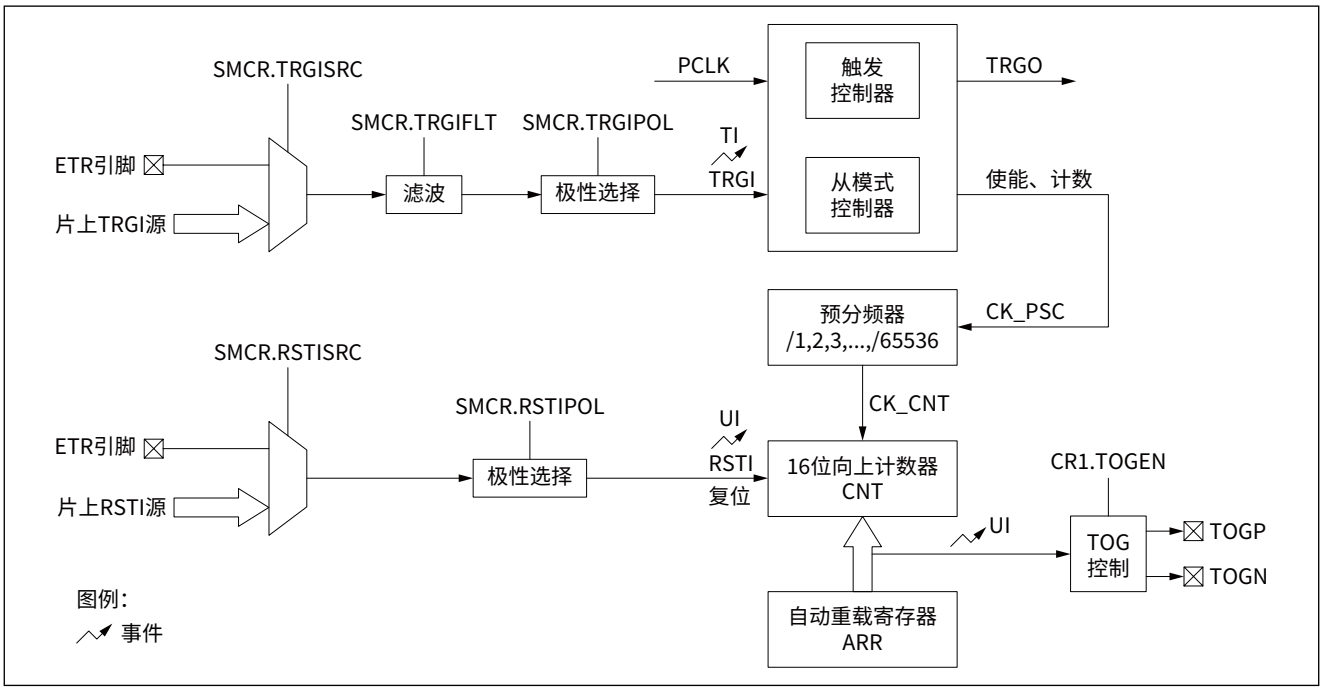
- 16bit 自动重载递增计数器
- 可编程预分频器支持 1、2、3、4、…、65536 分频
- 支持单次计数模式和连续计数模式
- 触发输入信号（TRGI）控制定时器实现多种从模式
- 用于外设间同步的触发输出信号（TRGO）
- 复位信号（RSTI）控制计数器复位
- 更新事件和触发事件发生时产生中断

11.3 功能描述

11.3.1 功能框图

BTIM 的功能框图如下图所示：

图 11-1 BTIM 功能框图



11.3.1.1 计数单元

计数单元的核心组件是一个 16bit 递增计数器 CNT 和一个 16bit 自动重载寄存器 ARR，计数器的时钟可通过预分频器进行分频。计数寄存器、自动重载寄存器和预分频寄存器可通过软件进行读写，即使在计数器运行时也可执行读写操作。

预分频器对 CK_PSC 时钟进行分频，得到计数时钟 CK_CNT，以驱动计数器计数。分频系数通过 BTIMx_PSC 寄存器进行设置，支持 1、2、3、4、…、65536 分频。

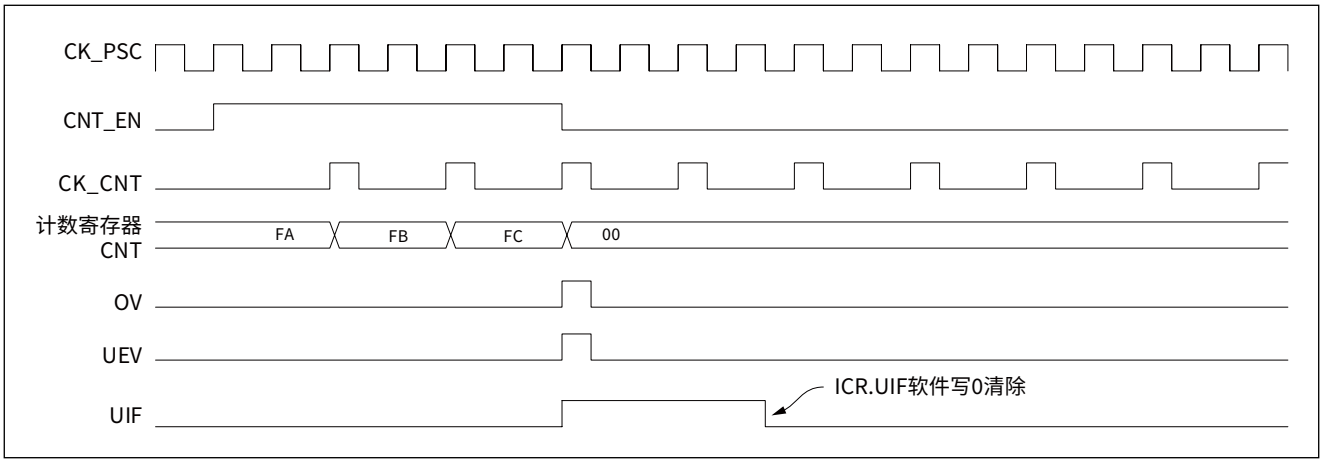
计数器可工作在单次计数或连续计数模式下，通过控制寄存器 BTIMx_CR1 的 ONESHOT 位域来选择。当设置 BTIMx_CR1 寄存器的 EN 位域为 1 时，计数器开始递增计数，注意实际的计数器使能信号 CNT_EN 在 EN 置 1 的一个时钟周期后被置 1。

单次计数模式

设置 BTIMx_CR1.ONESHOT 为 1，使定时器工作在单次计数模式下。设置 BTIMx_CR1.EN 为 1 使能 BTIMx，计数器 CNT 在计数时钟 CK_CNT 的驱动下累加计数。当计数值到达重载值 ARR 后产生溢出信号 OV 和更新事件 UEV（OV 信号和 UEV 信号会自动清除），计数器更新中断标志位 BTIMx_ISR.UIF 被硬件置位，同时计数器停止计数，BTIMx_CR1.EN 被硬件自动清零。

下图是单次计数模式示例，其中 BTIMx_PSC=0x01、BTIMx_ARR =0xFC。

图 11-2　单次计数模式示例

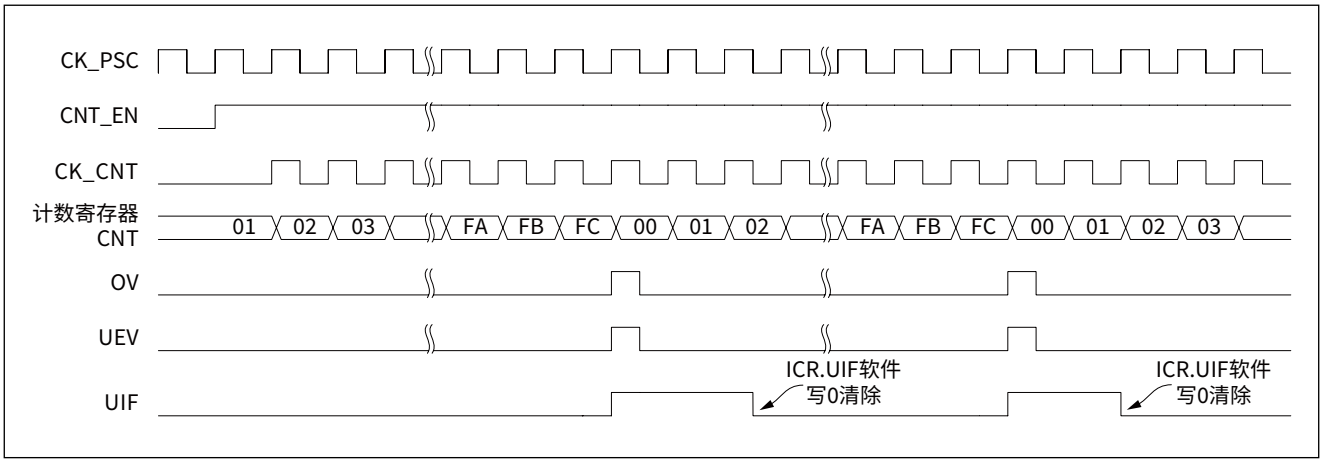


连续计数模式

设置 BTIMx_CR1.ONESHOT 为 0，使定时器工作在连续计数模式下。设置 BTIMx_CR1.EN 为 1 使能 BTIMx，计数器 CNT 在计数时钟 CK_CNT 的驱动下累加计数。当计数值到达重载值 ARR 后，将重新从 0 开始递增计数，同时产生溢出信号 OV 和更新事件 UEV（OV 信号和 UEV 信号会自动清除），计数器更新中断标志位 BTIMx_ISR.UIF 被硬件置位。

下图是连续计数模式示例，其中 BTIMx_PSC=0x00、BTIMx_ARR =0xFC。

图 11-3　连续计数模式示例



11.3.1.2 更新事件 UEV

允许通过控制寄存器 BTIMx_CR1 的 UDIS 位域来禁止或使能更新。

使能 UEV

设置 UDIS 位域为 0 使能 UEV，此时根据 URS 位域可选择更新请求源，如下表所示：

表 11-1 更新源设置

BTIMx_CR1.URS	更新源
0	计数器上溢出； UG 置位； 通过从模式控制器生成的更新事件
1	计数器上溢出

当发生更新事件时，将进行以下动作：

1. 重新初始化计数器，即计数器清零；
2. 预分频器的计数器被清零，但预分频比不受影响；
3. 事件更新中断标志位 BTIMx_ISR.UIF 被硬件置位。

当发生一个更新事件 UEV 时，事件更新中断标志位 BTIMx_ISR.UIF 会被硬件置位，如果允许中断（设置 BTIMx_IER.UIE 为 1），将产生一个更新中断请求，设置 BTIMx_ICR.UIF 为 0 可清除该标志位。

禁止 UEV

设置 UDIS 位域为 1 禁止 UEV，不再生成任何更新事件。

如果 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器的计数器。

11.3.1.3 触发输入通道

触发输入信号（TRGI）可用作计数器的计数时钟，也可作为计数器的触发启动信号和门控信号，具体请参见 11.3.2 工作模式。TRGI 信号的具体来源请参见 11.9.3 BTIMx_SMCR 从模式控制寄存器的 TRGISRC 位域说明。可对 TRGI 信号进行滤波控制，并设置 TRGI 信号的有效极性。

滤波器

滤波器采用数字滤波方式，以一定频率对触发输入信号进行采样，当连续采样到 N 个相同电平时信号有效，否则信号无效，以此滤除高频杂波信号。

滤波单元的采样时钟为 PCLK 或 PCLK 的分频，通过从模式控制寄存器 BTIMx_SMCR 的 TRGIFLT 位域可以选择 PCLK 的分频比及采样点个数 N。

极性选择

从模式控制寄存器 BTIMx_SMCR 的 TRGIPOL 位域用于选择触发输入信号的有效极性。当设置 TRGIPOL 为 0 时，TRGI 信号不反相，高电平或上升沿有效；当 TRGIPOL 设置为 1 时，TRGI 信号反相，低电平或下降沿有效。

11.3.1.4 触发输出通道

通过控制寄存器 BTIMx_CR2 的 MMS 位域，可以选择 BTIM 主模式下将要发送到其它外设以实现同步的触发输出信号（TRGO），如下表所示：

表 11-2 BTIM 触发输出（TRGO）

BTIMx_CR2.MMS	TRGO 信号
000	复位信号
001	计数器使能信号
010	更新事件
011	溢出信号

11.3.1.5 复位输入通道

BTIM 支持在不同工作模式下通过复位输入信号（RSTI）的有效边沿复位计数器，同时生成更新事件。

RSTI 信号的具体来源请参考从模式控制寄存器 BTIMx_SMCR 的 RSTISRC 位域说明。可通过 BTIMx_SMCR 寄存器的 RSTIPOL 位域设置 RSTI 信号的有效极性，当设置 RSTIPOL 为 0 时，RSTI 信号不反相，上升沿有效；当 RSTIPOL 设置为 1 时，RSTI 信号反相，下降沿有效。

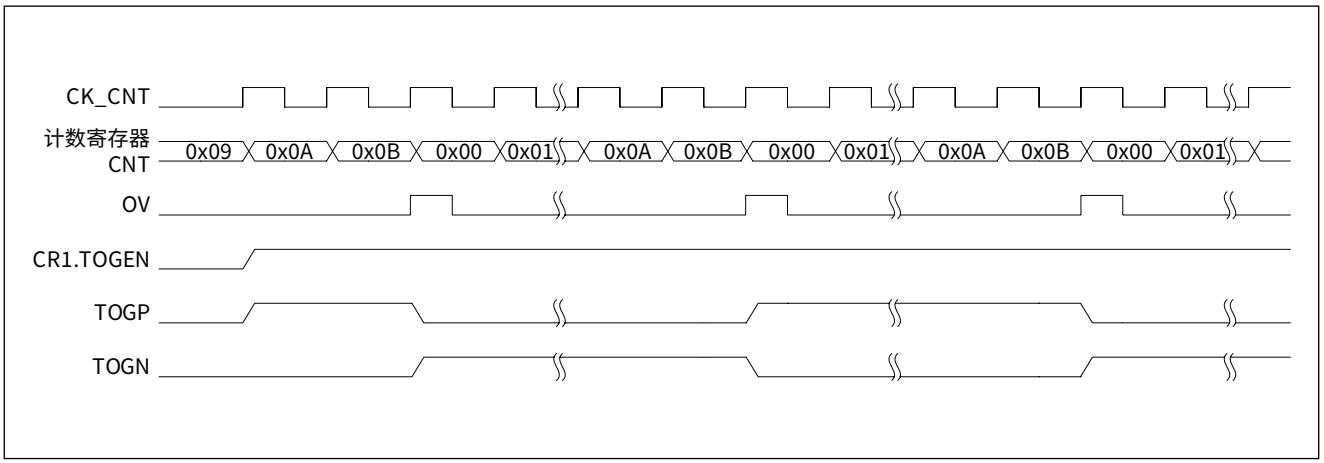
11.3.1.6 翻转输出单元

翻转输出单元可通过 ARR 溢出信号 OV 控制外部 BTIMx_TOGP 和 BTIMx_TOGN 引脚输出翻转信号。

- 当设置 BTIMx_CR1.TOGEN 为 0 时，BTIMx_TOGP 和 BTIMx_TOGN 引脚均输出低电平。
- 当设置 BTIMx_CR1.TOGEN 为 1 时，BTIMx_TOGP 和 BTIMx_TOGN 引脚输出电平相反的信号（BTIMx_TOGP 默认电平为高电平）；当计数器 ARR 溢出时，BTIMx_TOGP 和 BTIMx_TOGN 引脚输出电平将翻转。

下图所示为连续计数模式下，BTIMx_TOGP 和 BTIMx_TOGN 引脚电平翻转输出示意图：

图 11-4 电平翻转输出示意图（ARR=0x0B）



11.3.2 工作模式

BTIM 支持 4 种工作模式：内部计数模式、外部计数模式、触发启动模式和门控计数模式。通过从模式控制寄存器 BTIMx_SMCR 的 SMS 位域来配置，如下表所示：

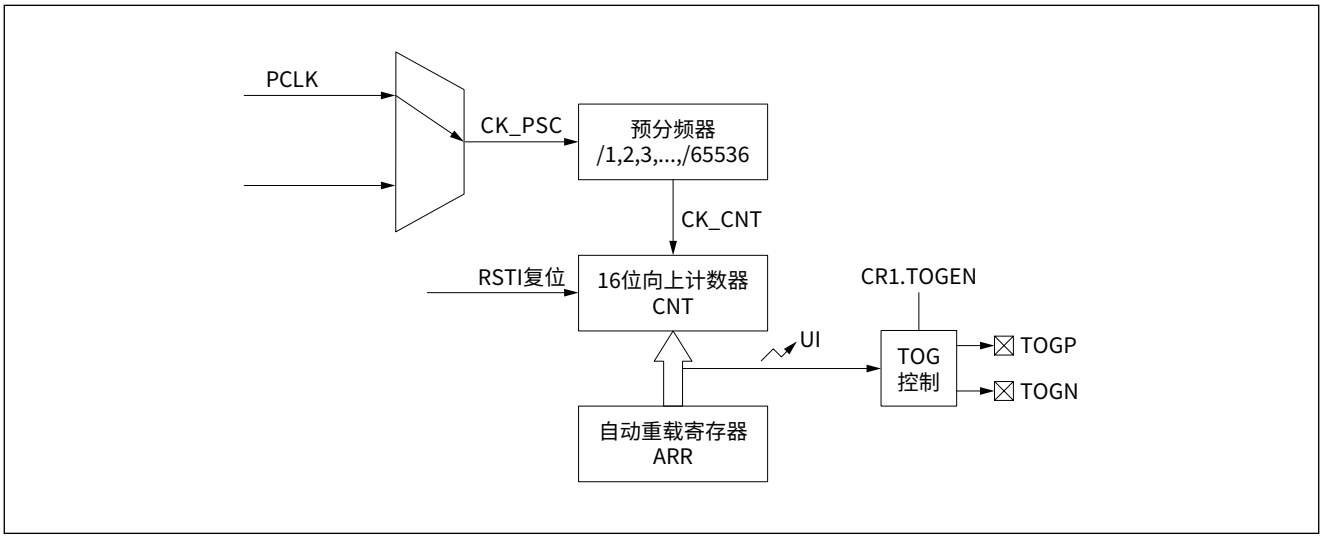
表 11-3 BTIM 工作模式

BTIMx_SMCR.SMS	工作模式	描述
000	禁止从模式，使用内部时钟	时钟源为 PCLK
001	触发启动模式	时钟源为 PCLK，TRGI 信号触发计数器启动
010	门控计数模式	时钟源为 PCLK，TRGI 信号作为门控信号
011	外部计数模式	时钟源为 TRGI 信号

11.3.2.1 内部计数模式

当从模式控制寄存器 BTIMx_SMCR 的 SMS 位域为 0x0 时，禁止定时器从模式，预分频器时钟直接由内部时钟 PCLK 提供。设置 BTIMx_CR1.EN 为 1，将使能计数器开始计数。内部计数模式框图如下图所示：

图 11-5 内部计数模式框图



在实际应用中，系统时钟 PCLK 的频率是已知的，通过合理设置预分频器系数 PSC 可以对固定时长的时钟进行计数，配合对重载值 ARR 的设置及计数器事件更新中断标志位的使用，可以精确获得某一特定时长，从而达到定时的目地。

定时时间 T 计算公式：

$$T = ((PSC+1)/PCLK) \times (ARR+1)$$

其中，PCLK 为计数器时钟源，PSC 为预分频系数，ARR 为重载值。

例：

当计数器时钟源 PCLK 的频率为 24MHz 时，要求定时 100ms。

如果设置预分频系数 PSC 为 0xFF，计算：

$$T = 100ms = ((255+1)/24MHz) \times (ARR+1)$$

则

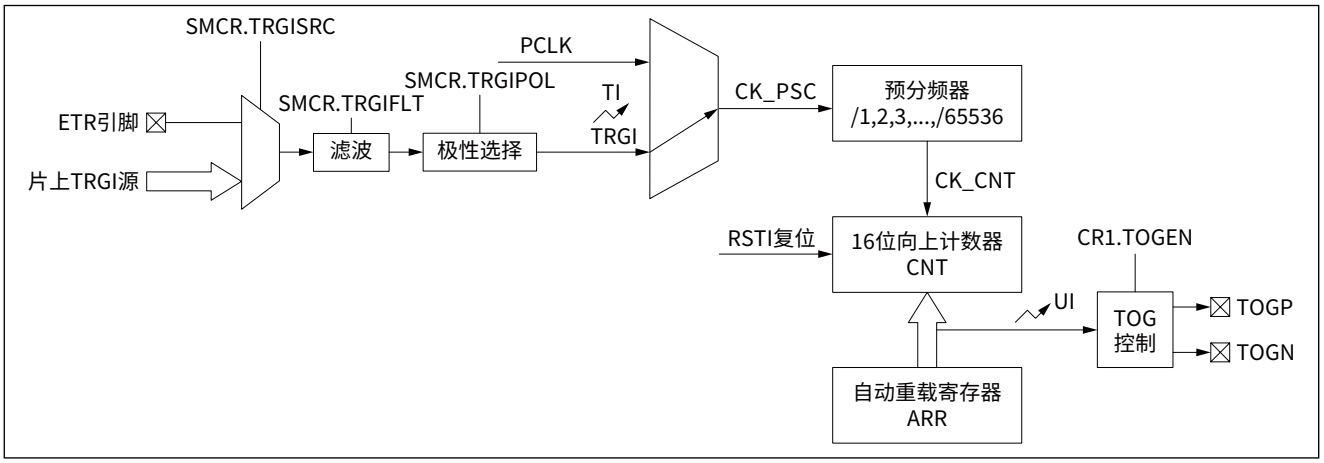
ARR = 9374 (0x249E)

即需要设置重载值 ARR 为 0x249E。

11.3.2.2 外部计数模式

当从模式控制寄存器 BTIMx_SMCR 的 SMS 位域为 0x3 时，由所选触发信号 (TRGI) 的上升沿提供计数器时钟。TRGI 信号有多种来源，具体通过 BTIMx_SMCR 寄存器的 TRGISRC 位域进行选择，并可进行滤波控制和极性选择。外部计数模式框图如下图所示：

图 11-6 外部计数模式框图



设置 BTIMx_CR1.EN 为 1 使能计数器，计数器将在 CK_CNT 时钟信号的驱动下累加计数。当 TRGI 出现有效边沿时，BTIMx_ISR.TIF 标志将置 1，向 BTIMx_ICR.TIF 写 0 可清除该标志。

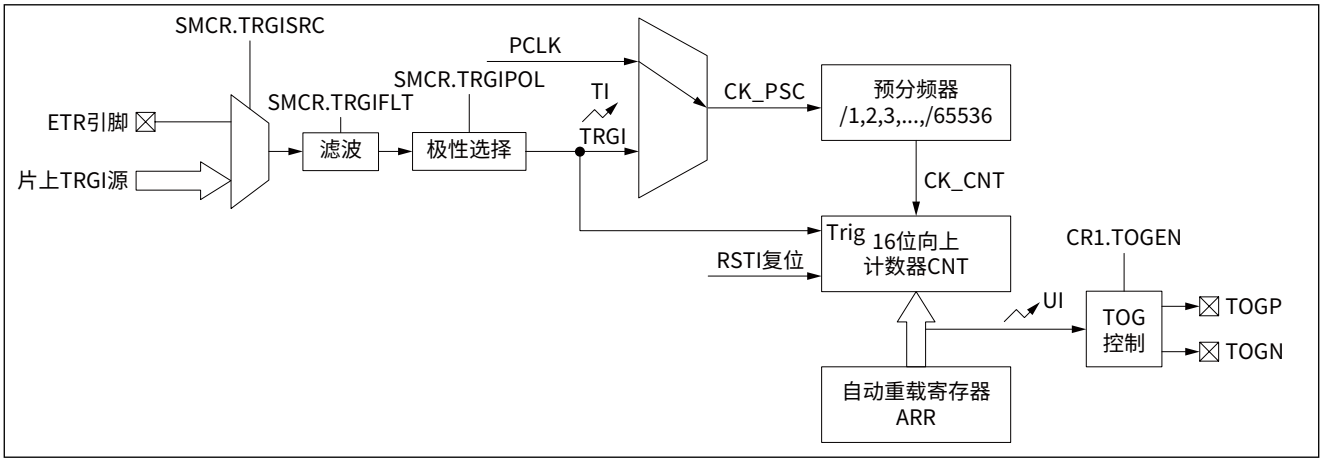
11.3.2.3 触发启动模式

当从模式控制寄存器 BTIMx_SMCR 的 SMS 位域为 0x1 时，BTIM 配置为触发启动模式。在该模式下，设置 BTIMx_CR1.EN 为 1 或触发信号 TRGI 出现上升沿时，将启动计数器 CNT 对内部时钟 PCLK 经预分频器分频后的 CK_CNT 信号进行计数。

TRGI 信号的来源由从模式控制寄存器 BTIMx_SMCR 的 TRGISRC 位域控制，并可通过 TRGIFLT 位域设置 TRGI 信号滤波，通过 TRGIPOL 位域设置 TRGI 信号的有效极性。

触发启动模式框图如下图所示：

图 11-7 触发启动模式框图

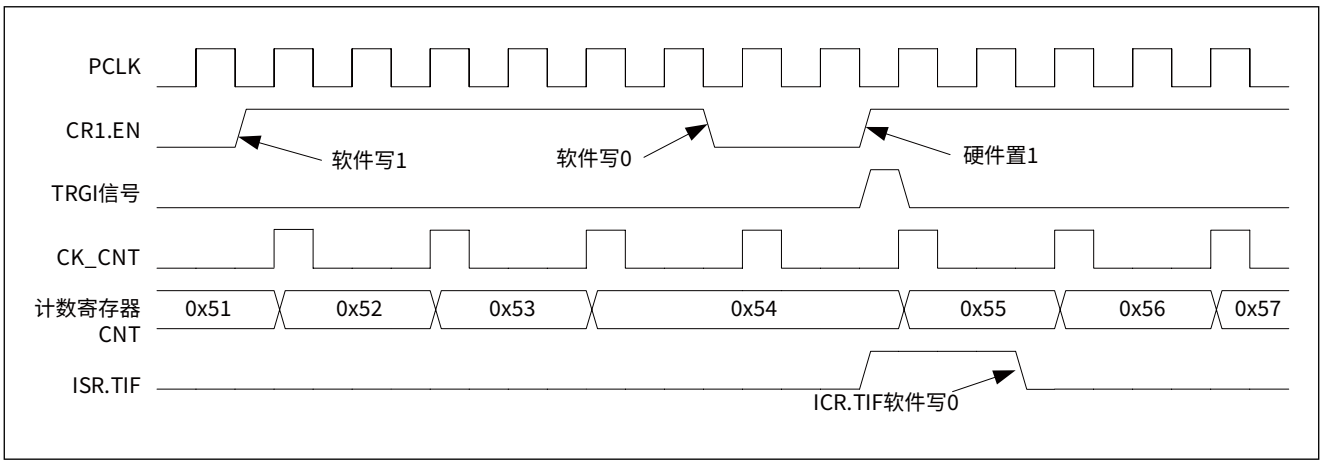


当检测到有效的触发信号时，将产生以下影响：

1. BTIMx_CR1.EN 被硬件置位；
2. 触发中断标志位 BTIMx_ISR.TIF 置 1，可产生中断；
3. 计数器启动，开始计数。

计数器启动后，计数器从初始值开始向上计数，当计数值到达重载值 ARR 后产生溢出。在任意时候设置运行控制位 BTIMx_CR1.EN 为 0 后，计数器立即暂停计数；再次设置运行控制位 BTIMx_CR1.EN 为 1 或者触发信号有效时，计数器按前一次的设置继续计数。触发启动模式时序图如下图所示，其中 BTIMx_PSC=0x01：

图 11-8 触发启动模式时序图



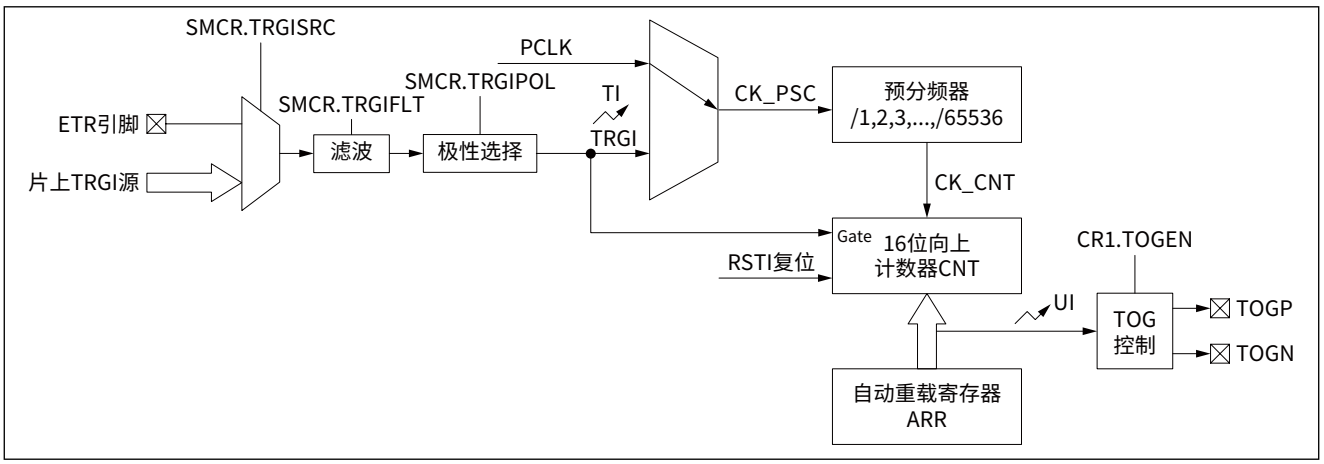
11.3.2.4 门控计数模式

当从模式控制寄存器BTIMx_SMCR的SMS位域为0x2时，BTIM配置为门控计数模式。在该模式下，触发输入(TRGI)为高电平且BTIMx_CR1.EN为1时，将启动计数器对内部时钟PCLK经预分频器分频后的CK_CNT信号进行计数；触发输入(TRGI)为低电平或BTIMx_CR1.EN为0时，计数器立即停止计数。

TRGI信号的来源由从模式控制寄存器BTIMx_SMCR的TRGISRC位域控制，并可通过TRGIFLT位域设置TRGI信号滤波，通过TRGIPOL位域设置TRGI信号的有效极性。

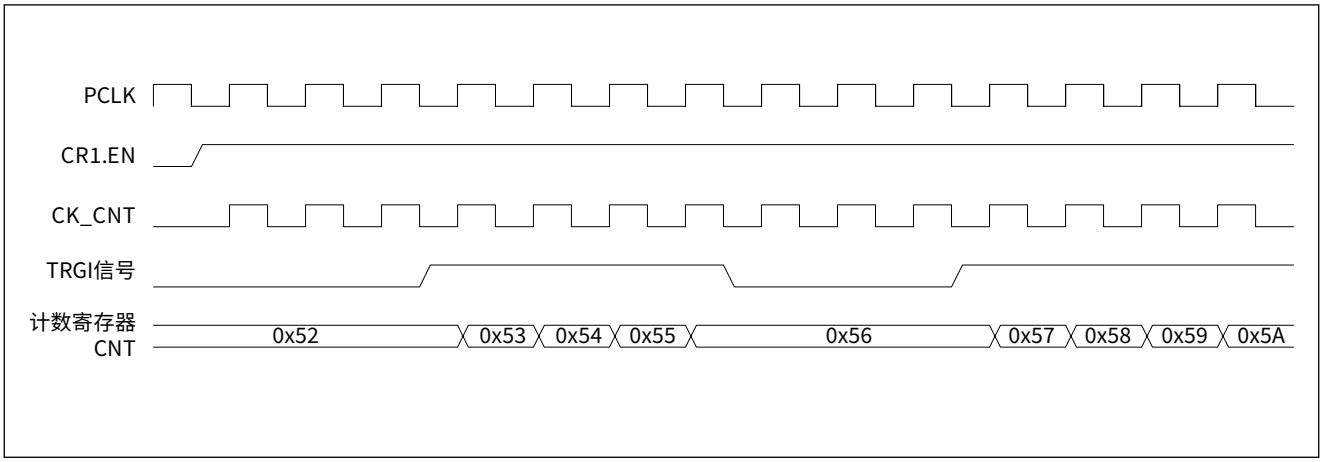
门控计数模式框图如下图所示：

图 11-9 门控计数模式框图



计数器启动后，计数器从初始值开始向上计数，当计数值到达重载值ARR后产生溢出。在任意时候设置运行控制位BTIMx_CR1.EN为0或门控信号无效时，计数器立即暂停计数；再次设置运行控制位BTIMx_CR1.EN为1且门控信号有效时，计数器按前一次的设置继续计数。门控计数模式时序图如下图所示，其中BTIMx_PSC=0x00：

图 11-10 门控计数模式时序图



11.3.3 UIF 位重映射

设置 BTIMx_CR1 寄存器中的 UIFREMAP 位为 1 可使能 UIF 状态位重映射功能，可强制将更新中断标志 BTIMx_ISR.UIF 连续复制到 BTIMx_CNT 寄存器的 UIFCPY 位域中。这样便可自动读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。在特定情况下，这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化计算。

UIF 和 UIFCPY 标志使能之间没有延迟。

当 BTIMx_CR1.UIFREMAP 为 0 时，UIFCPY 位域保留，读为 0。

11.4 BTIM 中断

BTIM 支持 2 个中断源，当 BTIM 中断事件发生时，中断标志位会被硬件置位，如果设置了对应的中断使能控制位，将产生中断请求。

在用户 BTIM 中断服务程序中，应查询相关 BTIM 中断标志位，以进行相应的处理，在退出中断服务程序之前，要清除该中断标志位，以避免重复进入中断服务程序。

各 BTIM 中断源的标志位、中断使能位、中断标志清除位或清除方法，如下表所示：

表 11-4 BTIM 中断控制

中断事件	中断标志位	中断使能位	中断标志清除
更新中断	ISR.UIF	IER.UIE	写 0 到 ICR.UIF
触发中断	ISR.TIF	IER.TIE	写 0 到 ICR.TIF

11.5 触发 ADC

BTIM 的触发输出信号 TRGO 可用于触发 ADC 启动，触发输出 TRGO 有多种可能的事件，具体由控制寄存器 BTIMx_CR2 的 MMS 位域进行选择。同时，ADC 外设需配置其外部触发启动寄存器，以选择对应触发源。

应注意，必须先使能 ADC 时钟，才能从主定时器接收事件，且从定时器接收触发信号时，不得实时更改 ADC 时钟。

11.6 调试支持

BTIM 支持在调试模式下停止或继续计数，通过调试状态定时器控制寄存器 SYSCTRL_DEBUG 的 BTIM123 位域来设置。

- 设置 SYSCTRL_DEBUG.BTIM123 为 1，则在调试状态时暂停 BTIM1/2/3 的计数器计数。
- 设置 SYSCTRL_DEBUG.BTIM123 为 0，则在调试状态时 BTIM1/2/3 的计数器继续计数。

11.7 编程示例

11.7.1 内部计数模式编程示例

步骤 1：设置 SYSCTRL_APBEN2.BTIM123 为 1，打开 BTIM1、BTIM2 和 BTIM3 的配置时钟及工作时钟；

注：

BTIM1、BTIM2 和 BTIM3 共用 SYSCTRL_APBEN2.BTIM123 位。

步骤 2：设置 BTIMx_SMCR.SMS 为 0x00，使 BTIMx 工作于内部计数模式；

步骤 3：配置 BTIMx_CR1.ONESHOT，选择单次或连续计数模式。配置为 0，则为连续计数模式；配置为 1，则为单次计数模式；

步骤 4：配置 BTIMx_PSC，设置预分频器的分频比；

步骤 5：配置 BTIMx_ARR，设置 BTIMx 计数溢出时间；

步骤 6：设置 BTIMx_CNT 为 0x0000，以便计数；

步骤 7：设置 BTIMx_IER.UIE 为 1 并配置对应 NVIC，使能更新中断；

步骤 8：设置 BTIMx_CR1.EN 为 1，启动 BTIMx；

步骤 9：当计数器溢出时，BTIMx_ISR.UIF 标志位置 1，进入中断服务程序，设置 BTIMx_ICR.UIF 为 0 清除该中断标志。

11.7.2 外部计数模式编程示例

以下示例中，配置计数器在 BTIMx_ETR 引脚的上升沿进行计数，步骤如下：

步骤 1：设置 SYSCTRL_APBEN2.BTIM123 为 1，打开 BTIM1、BTIM2 和 BTIM3 的配置时钟及工作时钟；

注：

BTIM1、BTIM2 和 BTIM3 共用 SYSCTRL_APBEN2.BTIM123 位。

步骤 2：将 BTIMx_ETR 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节；

步骤 3：设置 BTIMx_SMCR.TRGISRC 为 0xF，选择 TRGI 信号来源为 BTIMx_ETR 引脚；

步骤 4：设置 BTIMx_SMCR.TRGIFLT，配置 TRGI 输入信号滤波带宽；

步骤 5：设置 BTIMx_SMCR.TRGIPOL 为 0，选择上升沿有效；

步骤 6：设置 BTIMx_SMCR.SMS 为 0x3，使 BTIMx 工作于外部计数模式；

步骤 7：配置 BTIMx_CR1.ONESHOT，选择单次或连续计数模式。配置为 0，则为连续计数模式，配置为 1，则为单次计数模式；

步骤 8：配置 BTIMx_PSC，设置预分频器的分频比；

步骤 9：配置 BTIMx_ARR，设置 BTIMx 计数溢出时间；

步骤 10：设置 BTIMx_IER.UIE 为 1 并配置对应 NVIC，使能更新中断；

步骤 11：设置 BTIMx_CR1.EN 为 1，启动 BTIMx；

步骤 12：当计数器溢出时，BTIMx_ISR.UIF 标志位置 1，进入中断服务程序，设置 BTIMx_ICR.UIF 为 0 清除该中断标志。

11.7.3 触发启动模式编程示例

以下示例中，BTIMx_ETR 输入出现上升沿时触发启动计数器，步骤如下：

步骤 1：设置 SYSCTRL_APBEN2.BTIM123 为 1，打开 BTIM1、BTIM2 和 BTIM3 的配置时钟及工作时钟；

注：

BTIM1、BTIM2 和 BTIM3 共用 SYSCTRL_APBEN2.BTIM123 位。

步骤 2：将 BTIMx_ETR 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节；

步骤 3：设置 BTIMx_SMCR.TRGISRC 为 0xF，选择 TRGI 信号来源为 BTIMx_ETR 引脚；

步骤 4：设置 BTIMx_SMCR.TRGIFLT，配置 TRGI 输入信号滤波带宽；

步骤 5：设置 BTIMx_SMCR.TRGIPOL 为 0，选择上升沿有效；

步骤 6：设置 BTIMx_SMCR.SMS 为 0x1，使 BTIMx 工作于触发启动模式；

步骤 7：配置 BTIMx_CR1.ONESHOT，选择单次或连续计数模式。配置为 0，则为连续计数模式；配置为 1，则为单次计数模式；

步骤 8：配置 BTIMx_PSC，设置预分频器的分频比；

步骤 9：配置 BTIMx_ARR，设置 BTIMx 计数溢出时间；

步骤 10：当 BTIMx_ETR 输入出现上升沿时，计数器启动计数，同时 TIF 标志置 1。

11.7.4 门控计数模式编程示例

以下示例中，BTIMx_ETR 引脚输入的信号作为门控信号，控制计数器计数，步骤如下：

步骤 1：设置 SYSCTRL_APBEN2.BTIM123 为 1，打开 BTIM1、BTIM2 和 BTIM3 的配置时钟及工作时钟；

注：

BTIM1、BTIM2 和 BTIM3 共用 SYSCTRL_APBEN2.BTIM123 位。

步骤 2：将 BTIMx_ETR 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节；

步骤 3：设置 BTIMx_SMCR.TRGISRC 为 0xF，选择 TRGI 信号来源为 BTIMx_ETR 引脚；

步骤 4：设置 BTIMx_SMCR.TRGIFLT，配置 TRGI 输入信号滤波带宽；

步骤 5：设置 BTIMx_SMCR.TRGIPOL 为 0，选择高电平有效；

步骤 6：设置 BTIMx_SMCR.SMS 为 0x2，使 BTIMx 工作于门控计数模式；

步骤 7：配置 BTIMx_CR1.ONESHOT，选择单次或连续计数模式。配置为 0，则为连续计数模式；配置为 1，则为单次计数模式；

步骤 8：配置 BTIMx_PSC，设置预分频器的分频比；

步骤 9：配置 BTIMx_ARR，设置 BTIMx 计数溢出时间；

步骤 10：设置 BTIMx_CR1.EN 为 1，使能计数器；

步骤 11：当 BTIMx_ETR 输入为高电平时，计数器开始计数；当 BTIMx_ETR 输入为低电平时，计数器停止计数。

11.7.5 计数器复位编程示例

以下示例中，RSTI 输入出现上升沿时复位计数器，步骤如下：

步骤 1：根据编程示例，配置 BTIM 的工作模式；

步骤 2：配置 BTIMx_SMCR.RSTISRC，选择计数值复位信号 RSTI 来源；

步骤 3：设置 BTIMx_SMCR.RSTIPOL 为 0，选择上升沿有效；

步骤 4：当 RSTI 输入出现上升沿时，计数器清零，重新从 0 开始计数，同时 UIF 标志置 1。

11.8 寄存器列表

BTIM1 基地址：BTIM1_BASE = 0x4000 4800

BTIM2 基地址：BTIM2_BASE = 0x4000 4840

BTIM3 基地址：BTIM3_BASE = 0x4000 4880

表 11-5 BTIM 寄存器列表

寄存器名称	寄存器地址	寄存器描述
BTIMx_CR1	BTIMx_BASE + 0x00	控制寄存器 1
BTIMx_CR2	BTIMx_BASE + 0x04	控制寄存器 2
BTIMx_SMCR	BTIMx_BASE + 0x08	从模式控制寄存器
BTIMx_IER	BTIMx_BASE + 0x0C	中断使能寄存器
BTIMx_ISR	BTIMx_BASE + 0x10	中断标志寄存器
BTIMx_ICR	BTIMx_BASE + 0x18	中断标志清除寄存器
BTIMx_EGR	BTIMx_BASE + 0x14	事件生成寄存器
BTIMx_CNT	BTIMx_BASE + 0x24	计数寄存器
BTIMx_PSC	BTIMx_BASE + 0x28	预分频寄存器
BTIMx_ARR	BTIMx_BASE + 0x2C	自动重载寄存器

11.9 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

11.9.1 BTIMx_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	TOGEN	RW	TOG 引脚输出波形使能控制 0：TOGP、TOGN 均输出低电平 1：TOGP、TOGN 输出相反电平
14:12	RFU	-	保留位，请保持默认值
11	UIFREMAP	RW	UIF 状态位重映射使能控制 0：禁止重映射，BTIMx_CNT[31] 保持为 0 1：使能重映射，BTIMx_CNT[31] 等效于 BTIMx_ISR.UIF
10:4	RFU	-	保留位，请保持默认值
3	ONESHOT	RW	单次 / 连续计数模式控制 0：连续计数模式 1：单次计数模式
2	URS	RW	更新请求源配置 0：使能 UEV 时，所有以下事件都会产生更新中断 – 计数器上溢出 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：使能 UEV 时，只有计数器上溢出会生成更新中断
1	UDIS	RW	更新禁止控制 0：使能 UEV 更新事件可通过以下事件生成： – 计数器上溢出 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：禁止 UEV 不会生成更新事件。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。
0	EN	RW	定时器运行控制 0：定时器停止 1：定时器运行

11.9.2 BTIMx_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6:4	MMS	RW	主模式输出信号（TRGO）配置 000：复位信号，EGR.UG 用作触发输出 (TRGO)。如果复位由触发输入生成，则 TRGO 上的信号相比实际复位会有延迟。 001：使能信号，CR1.EN 用作触发输出 (TRGO)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 EN 控制位与门控模式下的触发输入的逻辑与运算组合而成。当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主 / 从模式时除外（请参见 SMCR.MSM 的说明）。 010：更新信号，选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。 011：溢出信号，每当 BTIM 溢出时输出一个脉冲。
3:0	RFU	-	保留位，请保持默认值

11.9.3 BTIMx_SMCR 从模式控制寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17	TRGIPOL	RW	触发输入信号（TRGI）极性选择 0：正相，高电平或上升沿有效 1：反相，低电平或下降沿有效
16	RSTIPOL	RW	计数值复位信号（RSTI）极性选择 0：正相，上升沿有效 1：反相，下降沿有效
15	RFU	-	保留位，请保持默认值
14:12	TRGIFLT	RW	TRGI 信号数字滤波电路采样时钟及采样点数配置 000: 无滤波 001：F_{sample} = PCLK， N=2 010：F_{sample} = PCLK， N=4 011：F_{sample} = PCLK， N=6 100：F_{sample} = PCLK/4， N=4 101：F_{sample} = PCLK/4， N=6 110：F_{sample} = PCLK/8， N=4 111：F_{sample} = PCLK/8， N=6
11	MSM	RW	主 / 从模式配置 0：不执行任何操作 1：当前定时器的触发输入事件 (TRGI) 的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。
10:7	TRGISRC	RW	触发输入信号（TRGI）来源配置 0000：保留 0111：ATIM_Trgo 0001：LVD_OUT 信号 1000：BTIM1_Trgo 0010：VCx_OUT 信号 1001：BTIM2_Trgo 0011：SPI_NCS 信号 1010：BTIM3_Trgo 0100：I2C_SCL 信号 1011：GTIM1_Trgo 0101：UARTx_RXD 信号 1100：GTIM2_Trgo 0110：RTC_OUT 信号 1111：BTIMx_ETR 信号 *注 1：切勿选择 BTIMx 自己输出的 Trgo。* *注 2：当设置为 0010 时，BTIM1 触发输入信号来源为 VC1_OUT；BTIM2 触发输入信号来源为 VC2_OUT；此时 BTIM3 触发输入信号来源无效。*

位域	名称	权限	功能描述
6:3	RSTISRC	RW	计数值复位信号（RSTI）来源配置 0000：无复位信号 0111：ATIM_Trgo 0001：LVD_OUT 信号 1000：BTIM1_Trgo 0010：VCx_OUT 信号 1001：BTIM2_Trgo 0011：SPI_NCS 信号 1010：BTIM3_Trgo 0100：I2C_SCL 信号 1011：GTIM1_Trgo 0101：UARTx_RXD 信号 1100：GTIM2_Trgo 0110：RTC_OUT 信号 1111：BTIMx_ETR 信号 *注 1：在 RSTI 信号的有效边沿复位计数器并生成更新事件。* *注 2：切勿选择 BTIMx 自己输出的 Trgo。* *注 3：当设置为 0010 时，BTIM1 复位信号来源为 VC1_OUT；BTIM2 复位信号来源为 VC2_OUT；此时 BTIM3 复位信号来源无效。*
2:0	SMS	RW	从模式功能配置 000：禁止从模式 001：触发启动模式：在 TRGI 上升沿使能计数器 010：门控计数模式：在 TRGI 高电平对 PCLK 计数 011：外部计数模式：在 TRGI 上升沿计数器自增 1

11.9.4 BTIMx_IER 中断使能寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	TIE	RW	触发中断使能控制 0：禁止 1：使能
5:1	RFU	-	保留位，请保持默认值
0	UIE	RW	更新中断使能控制 0：禁止 1：使能

11.9.5 BTIMx_ISR 中断标志寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	TIF	RO	触发中断标志 0：未发生触发事件 1：已发生触发事件
5:1	RFU	-	保留位，请保持默认值
0	UIF	RO	更新中断标志 0：未发生更新事件 1：已发生更新事件

11.9.6 BTIMx_ICR 中断标志清除寄存器

Address offset: 0x18 Reset value: 0x0000 0041

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	TIF	R1W0	触发标志清除 W0：清除触发标志 W1：无功能
5:1	RFU	-	保留位，请保持默认值
0	UIF	R1W0	更新标志清除 W0：清除计数器更新标志 W1：无功能

11.9.7 BTIMx_EGR 事件生成寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	TG	WO	此位由软件置 1 以生成事件，并由硬件自动清零 0：无功能 1：软件生成触发事件
5:1	RFU	-	保留位，请保持默认值
0	UG	WO	该位可通过软件置 1 以生成事件，并由硬件自动清零 0：无功能 1：重新初始化计数器并生成寄存器更新事件。请注意，预分频器的计数器也将清零，但预分频比不受影响。

11.9.8 BTIMx_CNT 计数寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	UIFCPY	RO	根据 BTIMx_CR1 寄存器中 UIFREMAP 位域的值，本位域表示不同的含义： UIFREMAP = 0，本位域保留，读为 0 UIFREMAP = 1，本位域表示 BTIMx_ISR 寄存器的 UIF 位的只读副本
30:16	RFU	-	保留位，请保持默认值
15:0	CNT	RW	定时器计数值

11.9.9 BTIMx_PSC 预分频寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PSC	RW	预分频器的值 计数器时钟频率 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)

11.9.10 BTIMx_ARR 自动重载寄存器

Address offset: 0x2C Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	ARR	RW	定时器重载值

12 低功耗定时器（LPTIM）

12.1 概述

CW32L011 内部集成 1 个 16 位低功耗定时器（LPTIM），可以以很低的功耗实现定时或对外部脉冲计数的功能。通过选择合适的时钟源和触发信号，可以实现系统低功耗休眠时将其唤醒的功能。LPTIM 内部具有一个比较寄存器，可实现比较输出和 PWM 输出，并可以控制输出波形的极性。此外，LPTIM 还可以与正交编码器连接，自动实现递增计数和递减计数。

12.2 主要特性

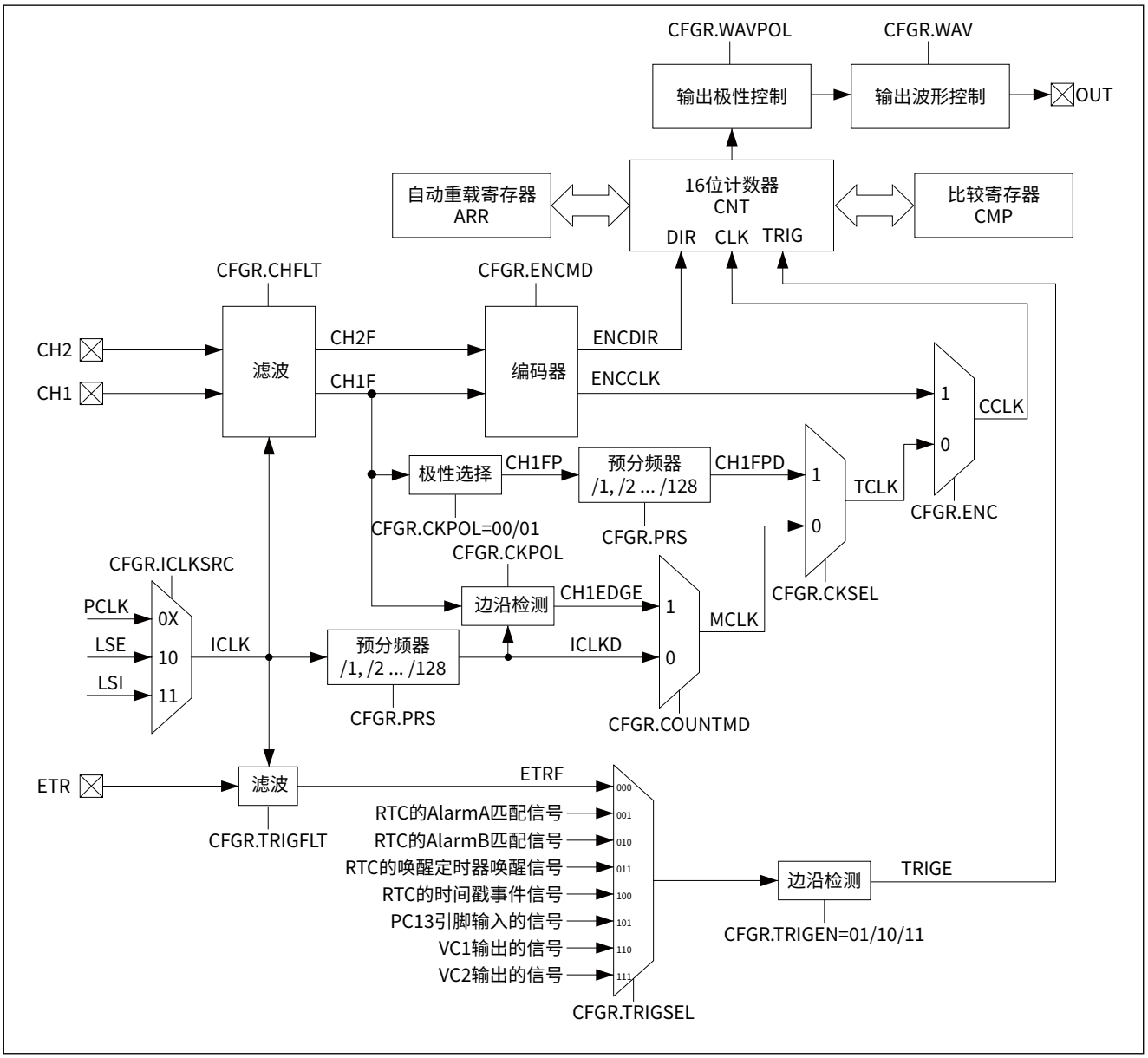
- 16bit 自动重载计数器
- 可编程预分频器支持 1、2、4、8、16、32、64、128 分频
- 工作时钟可选：
  - 内部时钟：PCLK、LSI、LSE
  - 外部时钟：LPTIM_CH1
- 支持单次模式和连续模式
- 丰富的软、硬件触发源
- 无时钟外部脉冲计数
- 输入、输出极性选择
- 可配置的输入端口滤波器
- 输出可配置为单脉冲、单次置位或 PWM 波
- 正交编码计数功能

12.3 功能描述

12.3.1 功能框图

LPTIM 功能框图如下图所示：

图 12-1 LPTIM 功能框图



12.3.1.1 时钟源

LPTIM 的计数时钟源可选 PCLK、LSE、LSI 或 LPTIM_CH1 引脚上输入的外部时钟，通过配置寄存器 LPTIM_CFGR 的 ICLKSRC、COUNTMD 和 CKSEL 位域进行配置。

当选择使用 LPTIM_CH1 引脚上输入的外部时钟作为计数时钟源时，LPTIM 可按如下两种配置模式工作：

- 第一种配置模式（CKSEL=0，COUNTMD=1）：CH1 输入的外部时钟作为 LPTIM 的计数时钟，同时内部时钟模块也需提供时钟给 LPTIM 做滤波和边沿检测使用。
- 第二种配置模式（CKSEL=1）：CH1 输入的外部时钟作为 LPTIM 的唯一时钟，该配置可用于在低功耗模式下所有内部振荡器都被关闭时，实现超时功能或脉冲计数器功能。

当配置为使用外部时钟源时，CKPOL 位域用于设置外部时钟信号的有效边缘。如果配置为上下两个边沿均有效，则还应提供内部时钟信号（第一配置）。这种情况下，内部时钟信号频率应至少是外部时钟信号频率的四倍。

12.3.1.2 预分频器

预分频器以 2 的倍数为系数对计数器时钟源进行分频，最大分频系数为 128。预分频系数由配置寄存器 LPTIM_CFGR 的 PRS 位域配置，分频系数配置如下表所示：

表 12-1 分频系数配置表

PRS	分频系数
000	1
001	2
010	4
011	8
100	16
101	32
110	64
111	128

12.3.1.3 滤波单元

LPTIM 输入，无论是外部（映射到 GPIO）还是内部（在芯片内映射到其他片上外设），都可以通过数字滤波器滤波，以防止毛刺和噪声扰动造成 LPTIM 虚假的计数或触发。

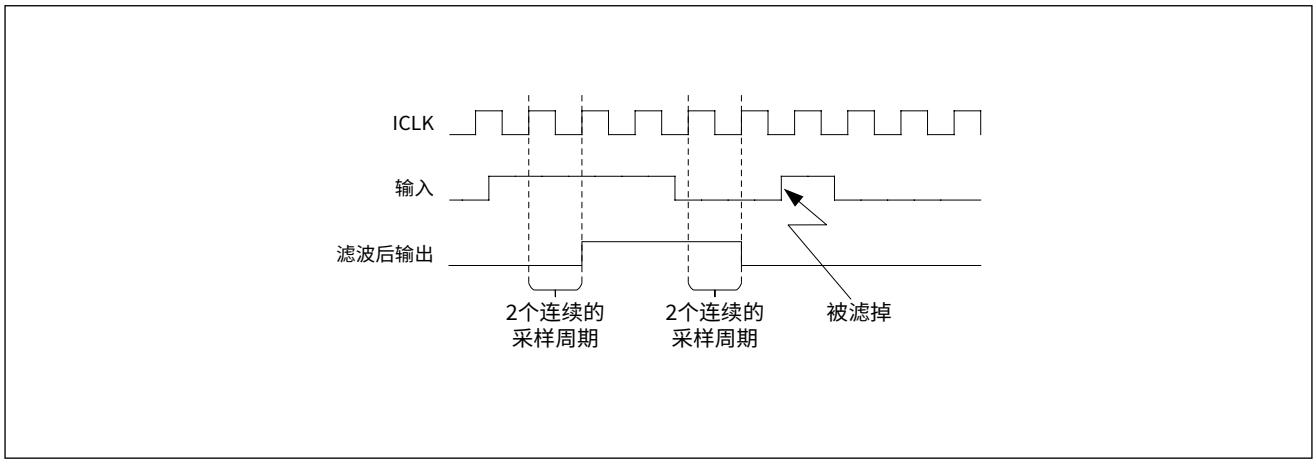
为了使滤波器能正常工作，应首先向 LPTIM 提供一个内部时钟源。这是保证滤波器的正常运行的必要条件。

数字滤波器可分为两组：

- 第一组数字滤波器用于 LPTIM 的外部输入（CH1、CH2），数字滤波器的灵敏度由 CHFLT 位控制。
- 第二组数字滤波器用于 LPTIM 触发器输入（ETR），数字滤波器的灵敏度由 TRGFLT 位控制。

改变滤波器对输入信号连续采样的有效电平保持数量可改变滤波器的灵敏度。例如，当滤波器设置为有效电平持续 2 个采样周期时，经滤波后的输出波形如下图所示：

图 12-2 滤波器时序图



注意：

如果没有提供内部时钟信号，则必须将 CHFLT 和 TRGFLT 位域设置为“00”来停用数字滤波器。在这种情况下，可以使用外部模拟滤波器来使 LPTIM 外部输入免受干扰的影响。

12.3.2 启动方式

控制寄存器 LPTIM_CR 的 EN 位域用于启用 / 禁用 LPTIM 内核。在设置 EN 位为 1 后，需要两个计数时钟周期的延时，LPTIM 才实际开始计数。

LPTIM 计数器可以通过软件启动或触发事件启动，通过配置寄存器 LPTIM_CFGR 的 TRIGEN 位域进行设置。

当设置 LPTIM_CFGR.TRIGEN 为 00 时，配置为软件启动。启动 LPTIM 后，设置 CNTSTART 或 SNGSTART 位域为 1，将启动 LPTIM 计数器进行计数。

当设置 LPTIM_CFGR.TRIGEN 为 01/10/11 时，配置为触发事件启动。启动 LPTIM 并配置操作模式（单次或连续模式）后，一旦检测到触发输入的有效边沿，将启动 LPTIM 计数器进行计数。具体触发输入信号请参见 12.3.3 触发源小节。

注意：

1. 只有在 EN 位域为 0，禁用 LPTIM 时，才可修改 LPTIM_CFGR 和 LPTIM_IER 寄存器。

2. 在设置 CNTSTART 或 SNGSTART 位域之前，必须先启用 LPTIM（EN=1）。当禁用 LPTIM 时，对这些位的任何写入都将被硬件丢弃。

3. 当配置为软件启动时，设置 CNTSTART 或 SNGSTART 位域后，计数器在延迟 3 个内核时钟周期后，才真正启动。

4. 由于存在触发信号和 LPTIM 的计时时钟不同步的情况，因此在触发器检测到触发信号后，计时器需要延迟两个计时时钟周期才开始运行。

12.3.3 触发源

LPTIM 计数器可以通过软件启动，也可以在检测到触发输入的有效边沿后启动。LPTIM 支持 8 个触发源，通过配置寄存器 LPTIM_CFGR 的 TRIGSEL 位域来设置，如下表所示：

表 12-2 LPTIM 触发输入来源表

TRIGSEL 位域	触发源
000	LPTIM_ETR 引脚的信号
001	RTC 的 ALARMA 匹配信号
010	RTC 的 ALARMB 匹配信号
011	RTC 的唤醒定时器唤醒信号
100	RTC 的时间戳事件信号
101	PC13 引脚输入的信号
110	VC1 输出的信号
111	VC2 输出的信号

TRIGEN 位域用于设置触发使能和极性，如下表所示：

表 12-3 LPTIM 触发使能和极性

TRIGEN	触发方式和极性
00	软件启动，软件写 CNTSTART 或 SNGSTART 启动计数器
01	硬件触发，在触发信号上升沿启动计数器
10	硬件触发，在触发信号下降沿启动计数器
11	在触发信号上升沿或下降沿启动计数器

注：

如果在计时器已启动时发生新的触发器事件，它将被忽略（除非启用了超时功能）。

12.3.4 操作模式

LPTIM 支持两种操作模式：

- 单次模式：计数器由软件启动或触发事件启动，在计数值到达重载值 ARR 时停止。
- 连续模式：计数器由软件启动或触发事件启动，计数器连续计数，直到计时器被禁用才停止。

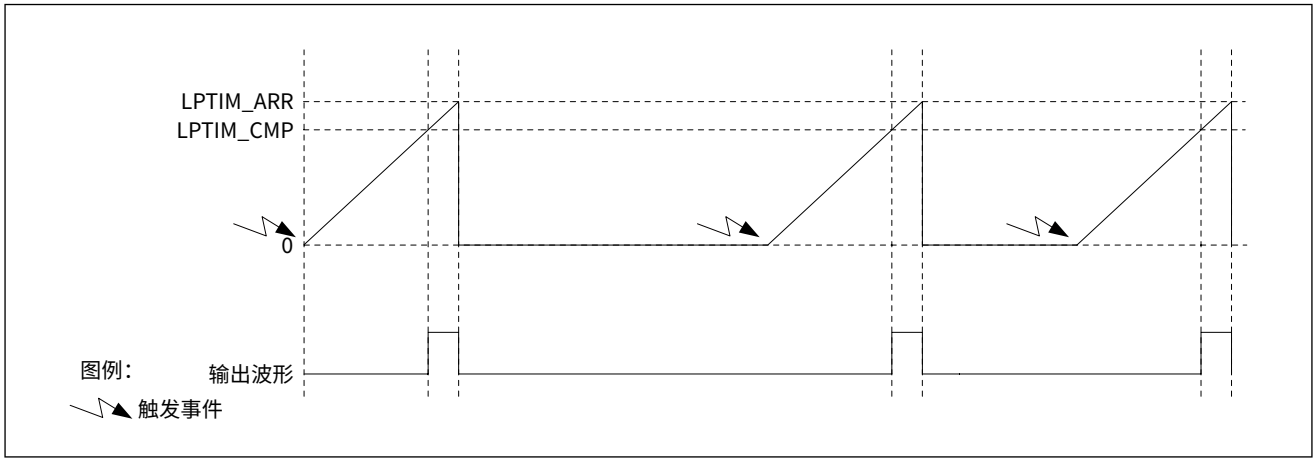
单次模式

单次计数模式下，计数器可由软件启动或触发事件启动，计数器 CNT 在 TCLK 时钟的驱动下累加计数。当计数值与重载值 ARR 相等时，自动重载匹配标志位 LPTIM_ISR.ARRM 被硬件置位，同时计数器停止计数。

如果设置为软件启动，则设置 LPTIM_CR.EN 为 1 后，设置 LPTIM_CR. SNGSTART 为 1，即可启动计数器进行计数。

如果设置为触发事件启动，则设置 LPTIM_CR.EN 为 1，设置 LPTIM_CR. SNGSTART 为 1，使 LPTIM 运行在单次模式。其后，一个触发事件将启动计数器进行计数。在计数器启动后和计数器到达 ARR 之前的任何触发事件将被丢弃。计数器停止后，新的触发事件将再次启动计数器开始计数。如下图所示：

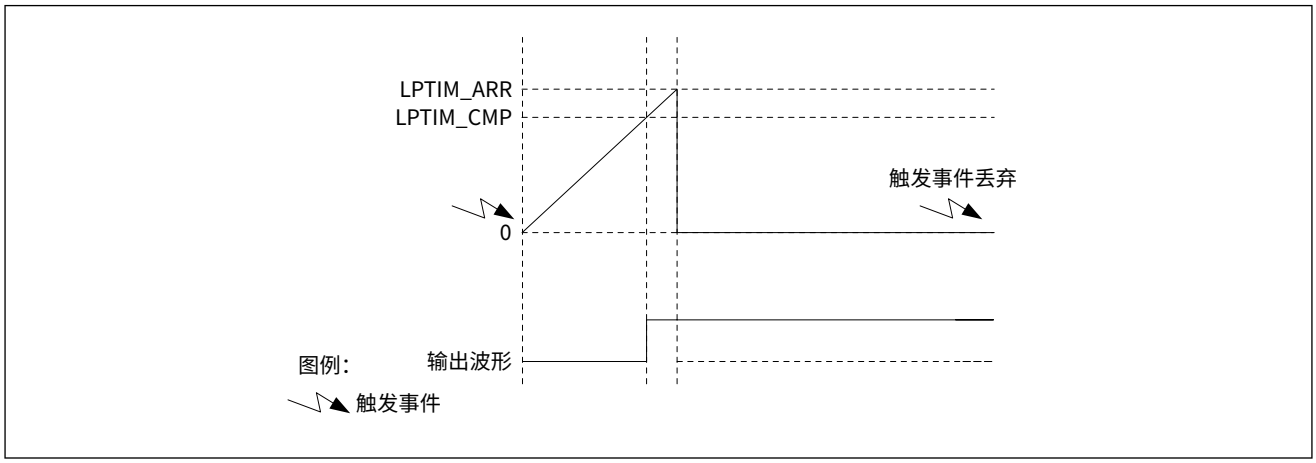
图 12-3 LPTIM 单次模式



单次置位的输出

需要注意的是，当设置了配置寄存器 LPTIM_CFGR 的 WAVE 位域为 1 时，单次置位模式将被激活。单次置位模式下，计数器只在第一个触发事件时启动一次，任何后续的触发事件将被丢弃，如下图所示：

图 12-4 LPTIM 单次置位波形



如果设置为软件启动（TRIGEN=‘00’），SNGSTART 设置为 1 后，将启动计数器进行一次计数。

连续模式

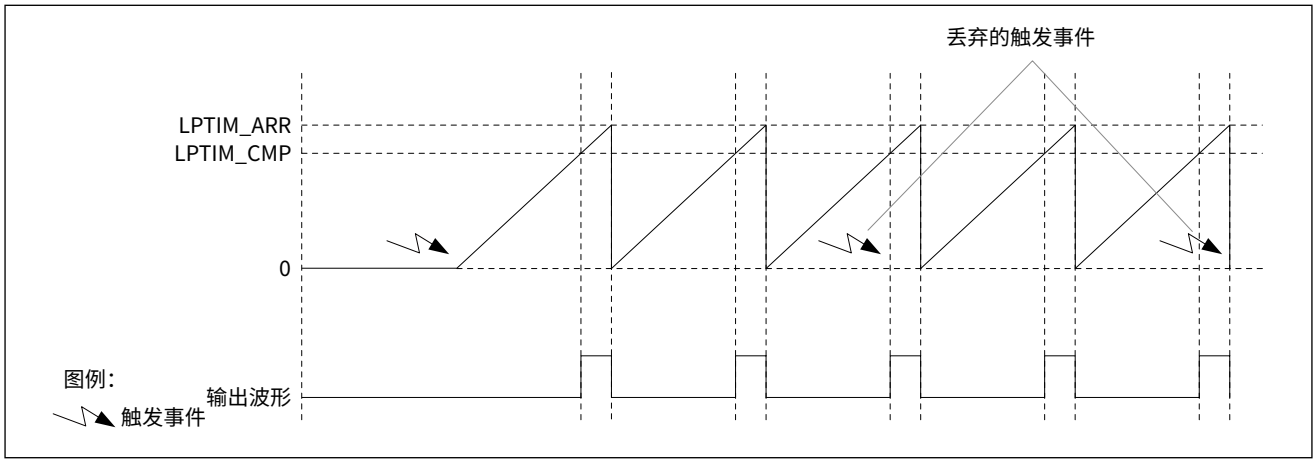
连续计数模式下，计数器可由软件启动或触发事件启动，计数器 CNT 在 TCLK 时钟的驱动下累加计数。当计数值与重载值 ARR 相等时，自动重载匹配标志位 LPTIM_ISR.ARRM 被硬件置位，计数器溢出并开始下一个周期的累加计数，直到设置 LPTIM_CR.EN 为 0 后才停止计数。

如果设置为软件启动，则设置 LPTIM_CR.EN 为 1 后，设置 LPTIM_CR. CNTSTART 为 1，即可启动计数器进行连续计数。

如果设置为触发事件启动，则设置 LPTIM_CR.EN 为 1，设置 LPTIM_CR. CNTSTART 为 1，使 LPTIM 运行在连续模式。其后，一个触发事件将启动计数器进行连续计数，任何后续的触发事件将被丢弃。

下图展示了连续模式下的计数和输出示例：

图 12-5 LPTIM 连续模式



单次模式和连续模式可以在运行时，相互切换。

- 如果之前配置为连续模式，那么设置 SNGSTART 位域将使 LPTIM 切换到单次模式。计数器（如果激活）将在到达重载值 ARR 后立即停止。
- 如果之前配置为单次模式，那么设置 CNTSTART 位域将使 LPTIM 切换到连续模式。计数器（如果激活）将在到达重载值 ARR 后立即重新启动。

12.3.5 超时功能

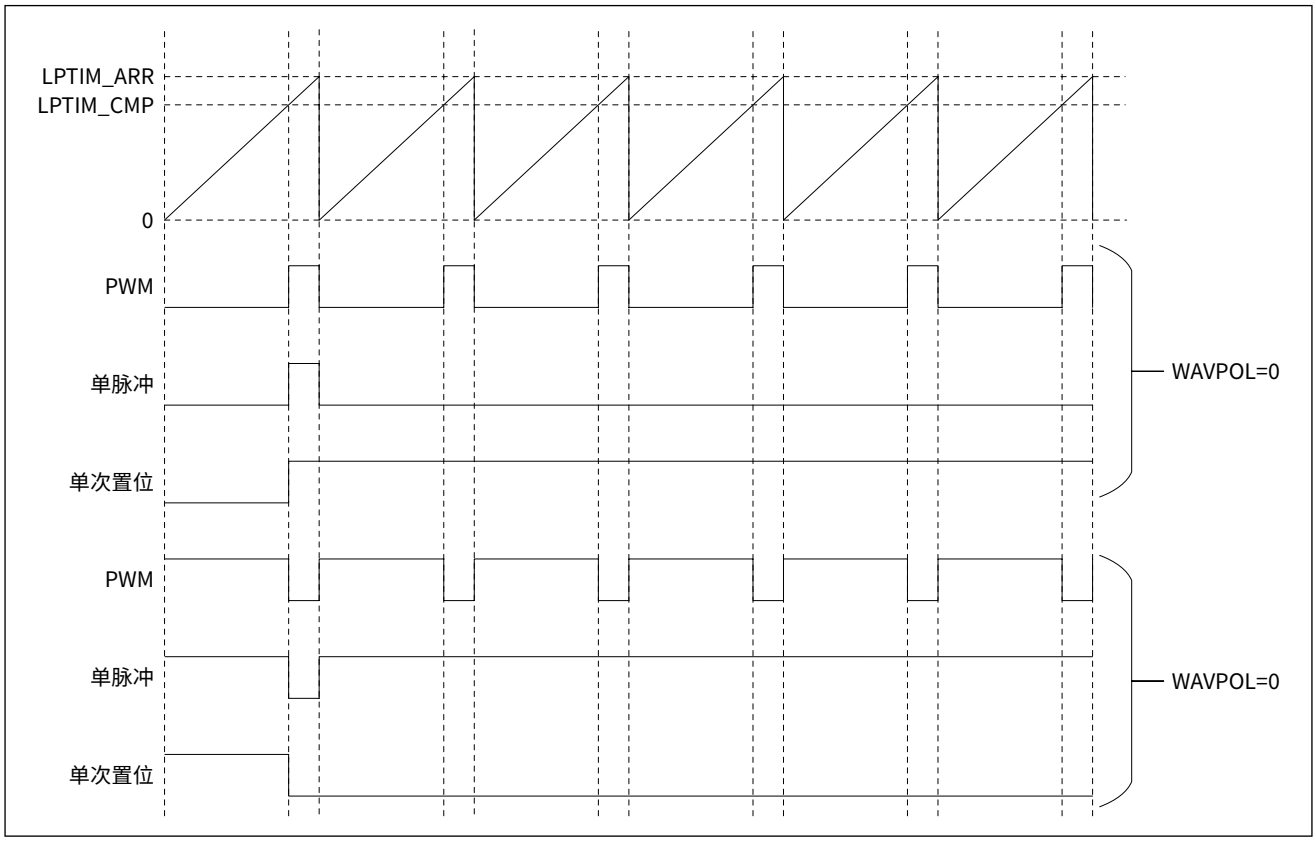
如果设置 TIMOUT 位域为 1，则 LPTIM 在计数时，可以通过触发信号使计数器重置，计数器将从 0 开始重新计数。

此特性可以实现一个低功耗的超时唤醒功能，超时门限通过比较寄存器 LPTIM_CMP 确定。当触发信号没有在预期时间范围内产生，即触发信号超时后，计数器计数到比较值 CMP，LPTIM 将产生一个比较匹配中断从而唤醒 MCU。

12.3.6 波形输出控制

LPTIM 在连续模式下可以输出 PWM 波，在单次模式下可以输出单脉冲波或单次置位波形，通过 WAVPOL 位可以控制输出波形的极性。当 WAVPOL 位为 0 时，为正向输出；当 WAVPOL 位为 1 时，为反向输出。更改 WAVPOL 位域，输出波形的极性更改将立即生效。LPTIM 没有使能时，修改 WAVPOL 的值也会更改输出的默认电平极性。下图显示了在 LPTIM 输出上可以产生的三种波形，此外，它还显示了 WAVPOL 位对极性变化的影响。

图 12-6 LPTIM 输出的三种波形



12.3.7 寄存器更新

配置寄存器 LPTIM_CFGR 的 PRELOAD 位域用于设置自动重载寄存器 LPTIM_ARR 和比较寄存器 LPTIM_CMP 的数据加载时刻。当设置 PRELOAD 为 0 时，LPTIM_ARR 和 LPTIM_CMP 寄存器将在写入操作后立即更新；当设置 PRELOAD 为 1 时，写操作的值在计数值到达前次 ARR 值时加载（如果定时器已经启动）。

由于 LPTIM 的 APB 接口和 LPTIM 的内核使用的是不同时钟，因此在对 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器进行写操作时，将值写入到寄存器和寄存器的值更新到 LPTIM 的 ARR 和 CMP 之间是存在一些延迟的。在此延迟期内，必须避免对这些寄存器的任何额外写入。通过 LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK 标志可以判断 ARR 和 CMP 是否更新完成。

LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK 标志为 0，表示对应寄存器的值正在更新，此时对 LPTIM_ARR 寄存器和 LPTIM_CMP 寄存器的任何写入，都会导致不可预测的结果。必须等待 LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK 标志为 1，即，寄存器已更新完成时，才能对同一寄存器执行新的写入操作。

12.3.8 计数功能

LPTIM 可对外部 LPTIM_CH1 引脚上输入的信号进行计数，也可对内部时钟信号进行计数。通过配置寄存器 LPTIM_CFGR 的 CKSEL 和 COUNTMD 位域来配置计数时钟源。

- 当 CKSEL 为 0 时，LPTIM 运行需要一个内部时钟 ICLK。

  如果 COUNTMD 为 0，则 LPTIM 对内部时钟源的脉冲进行计数；

  如果 COUNTMD 为 1，则 LPTIM 通过内部时钟对 LPTIM_CH1 引脚上的输入信号进行采样。因此，为了不错过任何事件，CH1 信号的变化频率不应超过提供给 LPTIM 的内部时钟的频率。此外，提供给 LPTIM 的内部时钟不能预分频（PRS = 000）。设置 CKPOL 位域的值，可选择在 CH1 信号的上升沿、下降沿或上下边沿进行计数。
- 当 CKSEL 为 1 时，LPTIM 运行由外部时钟源提供。

  在此模式下，COUNTMD 位域的值对 LPTIM 没有影响，LPTIM 不需要内部时钟源（除非启用了滤波器）。LPTIM 的 CH1 上输入的外部信号被用作 LPTIM 的计数。对于这种配置，LPTIM 计数器可以在 CH1 上的时钟信号的上升沿或下降沿时计数，但不能在上升沿和下降沿同时更新。由于输入到 LPTIM 的 CH1 上的信号也用于时钟 LPTIM 内核逻辑，所以在启动 LPTIM 之后，前 3 个计数时钟沿用于内核的配置，计数器将在第 4 个计数时钟沿才开始计数（参见 12.5.1 外部脉冲计数例程）。

12.3.9 编码模式

LPTIM 可以处理来自正交编码器的信号。正交编码器通过两个相位相差 90 度的脉冲信号输出来指示旋转元件的位置和旋转方向。LPTIM 在编码器模式下相当于连接了带有方向选择的外部时钟，这意味着，计数器仅在 0 到自动重载值 ARR 之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置自动重载寄存器 LPTIM_ARR。

编码器的两路输出信号通过 LPTIM 的两个外部输入端口 CH1 和 CH2 生成时钟信号作为 LPTIM 计数器时钟。这两个信号间的相位确定计数方向。编码模式下，LPTIM 必须由内部时钟源提供时钟，并且 CH1 和 CH2 上输入的信号频率不得超过 LPTIM 内部时钟频率的 4 分频，否则 LPTIM 不能正常工作。

方向变化由 LPTIM_ISR 寄存器中的 UP 和 DOWM 标志位指示。此外，如果设置 LPTIM_IER 寄存器中对应的使能位，则当两种方向变化事件产生时，会触发中断。

使用编码器模式前，LPTIM 必须首先配置为连续模式，然后设置 ENC 位为 1（注意此时 COUNTMD 位必须设置为 0），使能编码器模式。使能后，LPTIM 计数器会按照编码器的速度和方向自动修改 LPTIM_CNT 的计数值。因此，LPTIM_CNT 的计数值代表编码器的位置。计数方向由 UP 和 DOWN 标志指示，对应于所连传感器的旋转方向。

LPTIM 支持 3 种编码计数模式：设置 LPTIM_CFGR.ENCMD 为 0x00，计数器只在 CH1 的边沿计数；设置 LPTIM_CFGR.ENCMD 为 0x01，计数器只在 CH2 的边沿计数；设置 LPTIM_CFGR.ENCMD 为 0x02，计数器同时在 CH1 和 CH2 的边沿计数。

计数方向和编码器信号的关系如下表所示：

表 12-4 正交编码模式计数与方向

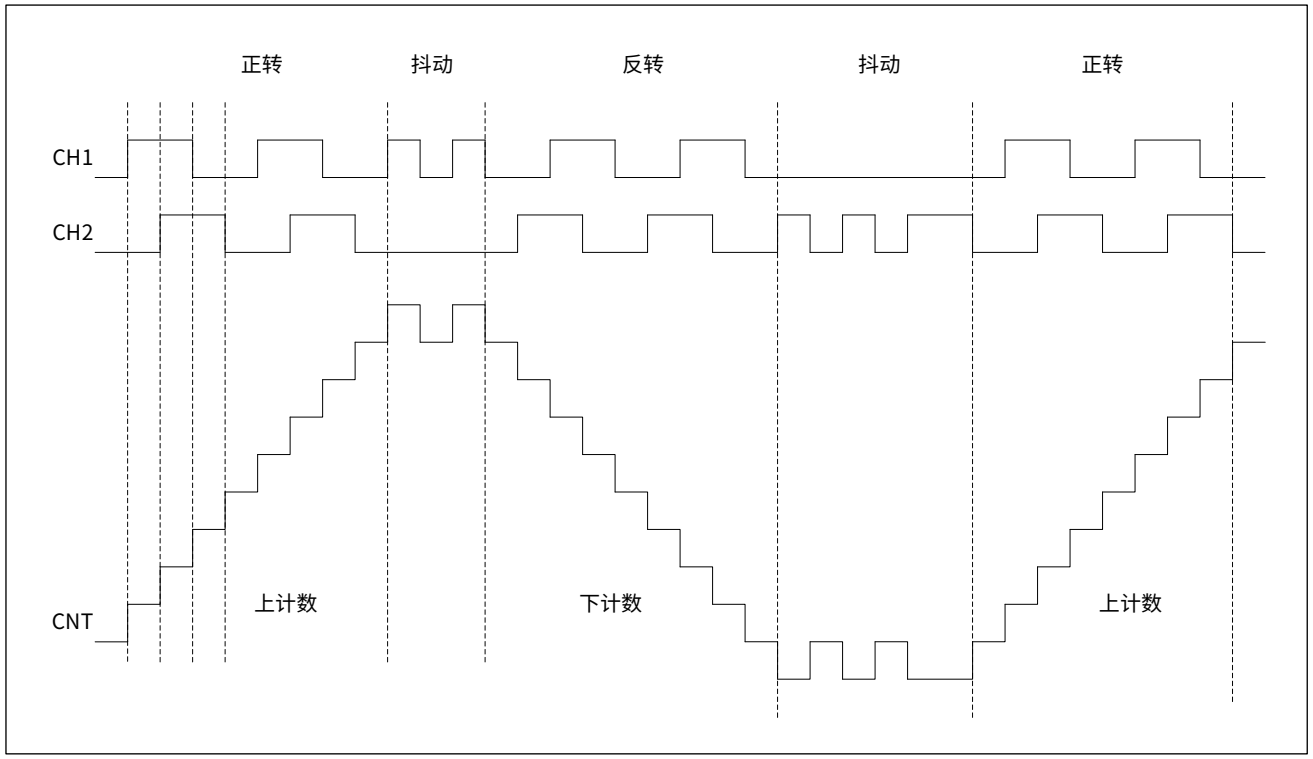
模式	信号的电平		CH1		CH2	
	CH1	CH2	上升	下降	上升	下降
模式 1	-	高	向下计数	向上计数	不计数	不计数
	-	低	向上计数	向下计数	不计数	不计数
模式 2	高	-	不计数	不计数	向上计数	向下计数
	低	-	不计数	不计数	向下计数	向上计数
模式 3	高	高	向下计数	向上计数	向上计数	向下计数
	低	低	向上计数	向下计数	向下计数	向上计数

注意：

在此模式下，LPTIM 必须由内部时钟源提供时钟，因此 CKSEL 位必须保持其复位值 0。另外，预分频器分频比必须等于其复位值 1（PRS 位域必须为“000”）。

下图是编码器模式计数示例，显示了计数信号的产生和方向控制：

图 12-7 编码器模式计数示例



12.4 调试支持

LPTIM 支持在调试模式下停止或继续计数，通过调试状态定时器控制寄存器 SYSCTRL_DEBUG 的 LPTIM 位域来设置。

- 设置 SYSCTRL_DEBUG.LPTIM 为 1，则在调试状态时暂停 LPTIM 的计数器计数。
- 设置 SYSCTRL_DEBUG.LPTIM 为 0，则在调试状态时 LPTIM 的计数器继续计数。

12.5 编程示例

12.5.1 外部脉冲计数例程

步骤 1：设置 SYSCTRL_APBEN2.LPTIM 为 1，使能 LPTIM 的配置时钟；

步骤 2：将 LPTIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节；

步骤 3：设置 LPTIM_CFGR.COUNTMD、LPTIM_CFGR.CKSEL 为 1，配置 LPTIM 计数模式，计数时钟由 CH1 提供；

步骤 4：设置 LPTIM_IER.ARRM 为 1，使能 ARR 自动重载匹配中断；

步骤 5：设置 LPTIM_ARR 为 6，目的是为了获得 10 个外部计数脉冲后，触发 ARR 自动重载匹配中断，因为前 5 个脉冲，CNT 并不计数；

步骤 6：设置 LPTIM_CR.EN 为 1，启动 LPTIM。

12.6 寄存器列表

LPTIM 基地址：LPTIM_BASE = 0x4000 6000

表 12-5 LPTIM 寄存器列表

寄存器名称	寄存器地址	寄存器描述
LPTIM_ISR	LPTIM_BASE + 0x00	中断和状态寄存器
LPTIM_ICR	LPTIM_BASE + 0x04	中断清除寄存器
LPTIM_IER	LPTIM_BASE + 0x08	中断使能寄存器
LPTIM_CFGR	LPTIM_BASE + 0x0C	配置寄存器
LPTIM_CR	LPTIM_BASE + 0x10	控制寄存器
LPTIM_CMP	LPTIM_BASE + 0x14	比较寄存器
LPTIM_ARR	LPTIM_BASE + 0x18	自动重载寄存器
LPTIM_CNT	LPTIM_BASE + 0x1C	计数器寄存器

12.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

12.7.1 LPTIM_CFGR 配置寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:27	RFU	-	保留位，请保持默认值
26:25	ICLKSRC	RW	ICLK 来源配置 00：PCLK 01：PCLK 10：LSE 11：LSI
24	ENC	RW	编码器模式使能 0：禁止 1：使能 *注：编码电路需要 ICLK 才能正常工作。*
23	COUNTMD	RW	计数模式配置 0：定时模式，ICLK 的分频时钟驱动计数器计数 1：计数模式，CH1 引脚的边沿驱动计数器计数 *注：编码器模式下，只可配置为 0。*
22	PRELOAD	RW	ARR 和 CMP 寄存器更新模式 0：立即对寄存器进行更新 1：当 LPTIM 周期结束时对寄存器进行更新
21	WAVPOL	RW	输出波形的极性配置 0：PWM 正向输出，CNT ≥ CMP 输出高电平 1：PWM 反向输出，CNT < CMP 输出高电平
20	WAVE	RW	波形控制 0：单次启动 SNGSTRT 对应单脉冲波形，连续启动 CNTSTRT 对应 PWM 波形 1：单次置位波形
19	TIMOUT	RW	触发超时使能 0：正在计数时发生的触发事件被忽略 1：正在计数时发生的触发事件会将计数值清零并重新开始计数
18:17	TRIGEN	RW	触发使能和极性 00：软件触发，由软件写寄存器启动计数器 01：硬件触发，在触发信号上升沿启动计数器 10：硬件触发，在触发信号下降沿启动计数器 11：硬件触发，在触发信号上升沿或下降沿启动计数器
16	RFU	-	保留位，请保持默认值

位域	名称	权限	功能描述
15:13	TRIGSEL	RW	触发源选择 000：LPTIM_ETR 引脚的信号 001：RTC 的 ALARMA 匹配信号 010：RTC 的 ALARMB 匹配信号 011：RTC 的唤醒定时器唤醒信号 100：RTC 的时间戳事件信号 101：PC13 引脚输入的信号 110：VC1 输出的信号 111：VC2 输出的信号
12	RFU	-	保留位，请保持默认值
11:9	PRS	RW	ICLK/CH1FP 信号分频配置 000：DIV1 001：DIV2 010：DIV4 011：DIV8 100：DIV16 101：DIV32 110：DIV64 111：DIV128 *注 1：当 COUNTMD 为 1 且 CKSEL 为 0 时，只可配置为 000。* *注 2：当工作于编码器模式时，只可配置为 000。*
8	RFU	-	保留位，请保持默认值
7:6	TRIGFLT	RW	触发信号数字滤波配置 00：不滤波 01：2 个 ICLK 时钟周期 10：4 个 ICLK 时钟周期 11：8 个 ICLK 时钟周期 *注 1：当 ICLK 时钟停止时，只可配置为无滤波功能。* *注 2：当触发源选择 RTC 相关的信号时请勿设置滤波时间。*
5	RFU	-	保留位，请保持默认值
4:3	CHFLT	RW	CH1 / CH2 通道数字滤波配置 00：无滤波功能 01：2 个 ICLK 时钟周期 10：4 个 ICLK 时钟周期 11：8 个 ICLK 时钟周期 *注：当 ICLK 时钟停止时，只可配置为无滤波功能。*
2:1	ENCMD	RW	编码器工作模式配置 00：编码计数模式 1，CH1 信号变化沿计数 01：编码计数模式 2，CH2 信号变化沿计数 10：编码计数模式 3，CH1/CH2 信号变化沿计数 11：保留

位域	名称	权限	功能描述
2:1	CKPOL	RW	CH1 信号计数边沿配置 00：CH1 信号上升沿计数器递增 01：CH1 信号下降沿计数器递增 10：CH1 信号上升沿或下降沿时计数器递增 11：保留 *注：当 CKSEL 为 1 时，只可配置为 00/01。*
0	CKSEL	RW	计数器计数时钟 TCLK 来源配置 0：计数时钟来自 MCLK，对 PCLK / LSE / LSI / CH1 边沿进行计数 1：计数时钟来自 CH1FPD，对 CH1 输入的时钟进行计数 *注：编码器模式下，只可配置为 0。*

注：

LPTIM 禁止时，才可设置此寄存器。

12.7.2 LPTIM_CR 控制寄存器

Address offset: 0x10 Reset value: 0x0000 0018

位域	名称	权限	功能描述
31:5	RFU	-	保留位，请保持默认值
4	ARST	RW	计数器计数值异步复位 W0：复位计数器计数值 W1：无功能 R0：计数器计数值正在进行复位 R1：计数器计数值复位完成 *注：仅读到该控制位为 1 时才可以写入 0 以进行复位。*
3	SRST	RW	计数器计数值同步复位 W0：复位计数器计数值 W1：无功能 R0：计数器计数值正在进行复位 R1：计数器计数值复位完成 *注：仅读到该控制位为 1 时才可以写入 0 以进行复位。*
2	CNTSTART	RW	连续模式运行控制 W0：无功能 W1：启动连续模式
1	SNGSTART	RW	单脉冲模式运行控制 W0：无功能 W1：启动单脉冲模式
0	EN	RW	LPTIM 使能控制 0：禁止 LPTIM 1：使能 LPTIM

12.7.3 LPTIM_ARR 自动重载寄存器

Address offset: 0x18 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	ARR	RW	自动重载值

12.7.4 LPTIM_CNT 计数器寄存器

Address offset: 0x1C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CNT	RO	计数器值 *注：连续两次读到相同的值才认为读出数据。*

12.7.5 LPTIM_CMP 比较寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CMP	RW	比较值

12.7.6 LPTIM_IER 中断使能寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	DOWN	RW	计数方向变为反向中断使能控制 0：禁止 1：使能
5	UP	RW	计数方向变为正向中断使能控制 0：禁止 1：使能
4	ARROK	RW	自动重载寄存器更新完成中断使能控制 0：禁止 1：使能
3	CMPOK	RW	比较寄存器更新完成中断使能控制 0：禁止 1：使能
2	EXTTRIG	RW	触发完成中断使能控制 0：禁止 1：使能
1	ARRM	RW	自动重载匹配中断使能控制 0：禁止 1：使能
0	CMPM	RW	比较匹配中断使能控制 0：禁止 1：使能

注意：

LPTIM 禁止时，才可设置此寄存器。

12.7.7 LPTIM_ISR 中断和状态寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8:7	DIR	RO	编码器计数方向 00：编码器停止计数，当前计数方向为递减 01：编码器停止计数，当前计数方向为递增 10：编码器正在计数，当前计数方向为递减 11：编码器正在计数，当前计数方向为递增
6	DOWN	RO	编码器计数方向变为反向标志 0：计数方向未发生变化 1：计数方向从递增变为递减
5	UP	RO	编码器计数方向变为正向标志 0：计数方向未发生变化 1：计数方向从递减变为递增
4	ARROK	RO	自动重载寄存器更新完成标志 0：自动重载寄存器正在更新 1：自动重载寄存器已更新完成
3	CMPOK	RO	比较寄存器更新完成标志 0：比较寄存器正在更新 1：比较寄存器已更新完成
2	EXTTRIG	RO	触发完成标志 0：未发生触发事件 1：已发生触发事件
1	ARRM	RO	自动重载匹配标志 0：计数器计数值与 ARR 寄存器数值不相等 1：计数器计数值与 ARR 寄存器数值相等
0	CMPM	RO	比较匹配标志 0：计数器计数值与 CMP 寄存器数值不相等 1：计数器计数值与 CMP 寄存器数值相等

12.7.8 LPTIM_ICR 中断清除寄存器

Address offset: 0x04 Reset value: 0x0000 007F

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	DOWN	R1W0	计数方向变为反向标志清除 W0：清除计数方向变为反向标志 W1：无功能
5	UP	R1W0	计数方向变为正向标志清除 W0：清除计数方向变为正向标志 W1：无功能
4	ARROK	R1W0	自动重载寄存器更新完成标志清除 W0：清除自动重载寄存器更新完成标志 W1：无功能
3	CMPOK	R1W0	比较寄存器更新完成标志清除 W0：清除比较寄存器更新完成标志 W1：无功能
2	EXTTRIG	R1W0	触发完成标志清除 W0：清除触发完成标志 W1：无功能
1	ARRM	R1W0	自动重载匹配标志清除 W0：清除自动重载匹配标志 W1：无功能
0	CMPM	R1W0	比较匹配标志清除 W0：清除比较匹配标志 W1：无功能

13 通用定时器（GTIM）

13.1 概述

CW32L011 内部集成 2 个通用定时器（GTIM），每个 GTIM 完全独立且功能完全相同，各包含一个 16bit 自动重装载计数器，并由一个可编程预分频器驱动。GTIM 支持定时、计数、复位、门控、触发和编码器等多种工作模式，每组 GTIM 带 4 路独立的捕获 / 比较通道，可以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）。

13.2 主要特性

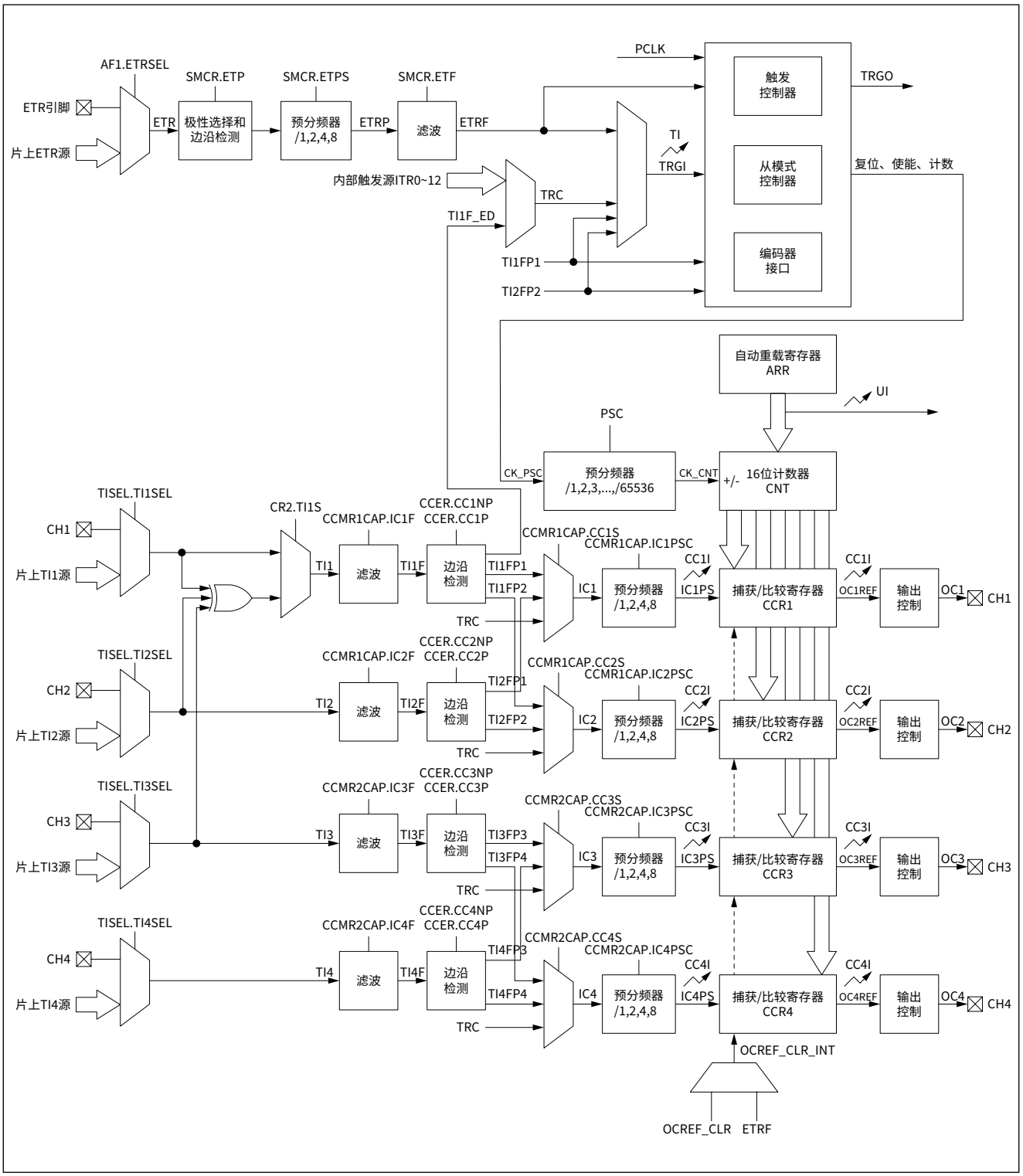
- 16bit 递增、递减和递增 / 递减自动重载计数器
- 可编程预分频器支持 1、2、3、4、…、65536 分频
- 支持单次计数模式和连续计数模式
- 4 路独立输入捕获和输出比较通道
- 触发输入信号（TRGI）控制定时器实现多种从模式
- 定时器级联 ITR 和片内外设互联 ETR
- 支持针对定位的增量 (正交) 编码器和霍尔传感器电路
- 多种事件发生时产生中断请求：
  - 更新事件
  - 触发事件
  - 输入捕获
  - 输出比较

13.3 功能描述

13.3.1 功能框图

GTIM 的功能框图如下图所示：

图 13-1 GTIM 功能框图



13.3.1.1 时钟源

计数器的计数时钟源可由内部时钟 PCLK、触发信号 TRGI 或外部触发输入信号 ETRF 提供，经预分频器 GTIMx_PSC 分频后驱动计数器进行计数。不同工作模式下具有不同时钟源，具体请参见 13.3.2 工作模式。

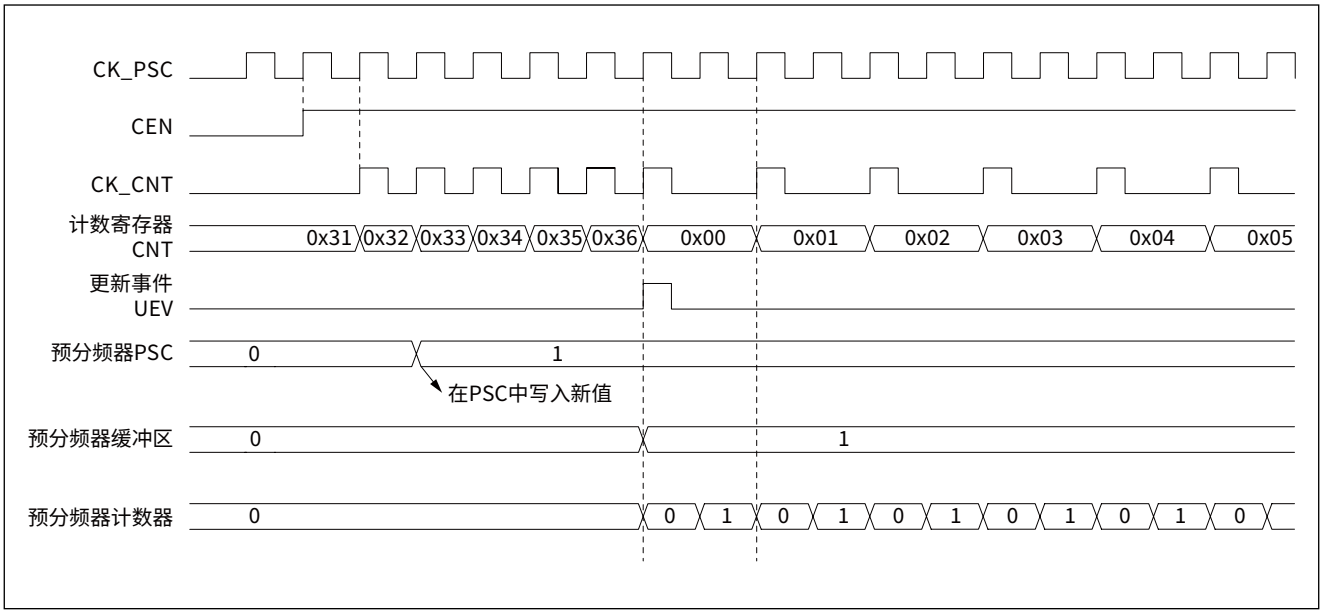
13.3.1.2 预分频器

预分频器对 CK_PSC 时钟进行分频，得到计数时钟 CK_CNT，以驱动计数器计数。分频系数通过 GTIMx_PSC 寄存器进行设置，支持 1、2、3、4、…、65536 分频。

GTIMx_PSC 寄存器具有缓冲功能，可在运行中修改，新的预分频值将在下一个更新事件发生时生效。

下图给出了运行过程中预分频器的分频由 1 变为 2 时的时序图：

图 13-2 预分频器分频由 1 变为 2 时的时序图



13.3.1.3 计数器与计数模式

计数器可设置为递增计数（边沿对齐模式）、递减计数（边沿对齐模式）或递增 / 递减双向计数（中心对齐模式）。具体通过控制寄存器 GTIMx_CR1 的 CMS 和 DIR 位域进行配置，如下表所示：

表 13-1 计数模式

GTIMx_CR1.CMS	GTIMx_CR1.DIR	计数模式
00	0	递增计数（边沿对齐模式）
	1	递减计数（边沿对齐模式）
01	-	中心对齐模式 1
10	-	中心对齐模式 2
11	-	中心对齐模式 3

当设置 GTIMx_CR1 寄存器的 CEN 位域为 1 时，计数器开始按设定模式计数，注意实际的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

递增计数模式

在递增计数模式下，计数器从 0 开始递增计数到重载值 ARR，然后重新从 0 开始递增计数，同时生成计数器上溢出事件。

当计数器上溢出时，产生上溢出信号 OV，同时产生更新事件 UEV（OV 信号和 UEV 信号会自动清除），计数器更新中断标志位 GTIMx_ISR.UIF 被硬件置位，如果允许中断（设置 GTIMx_IER.UIE 为 1），将产生中断请求，设置 GTIMx_ICR.UIF 为 0 清除该标志位。

以下是计数器在不同时钟频率下的操作示例，其中 GTIMx_ARR =0x36：

图 13-3 递增计数，内部时钟分频因子为 1

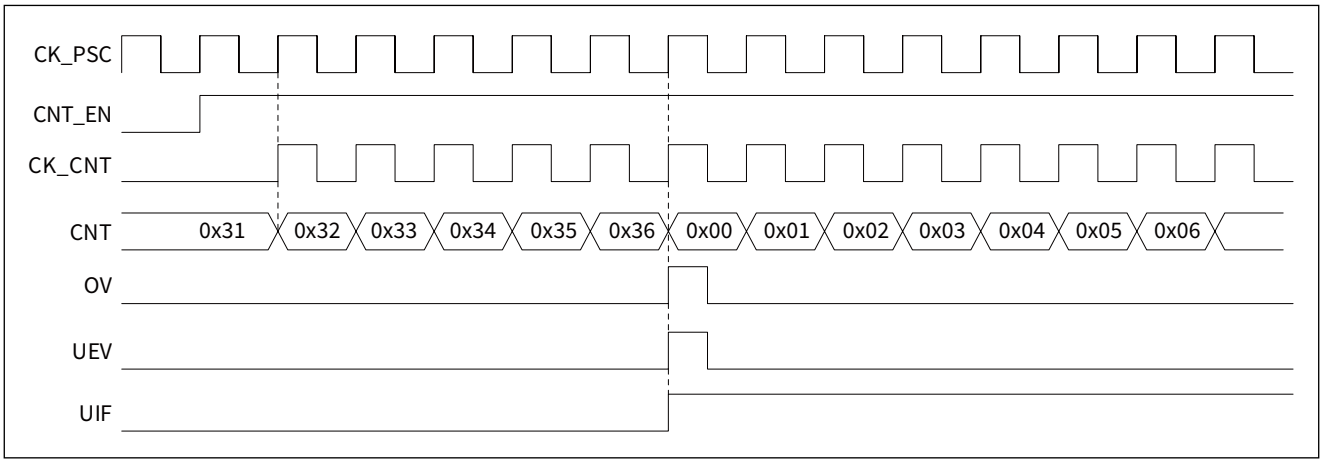


图 13-4　递增计数，内部时钟分频因子为 2

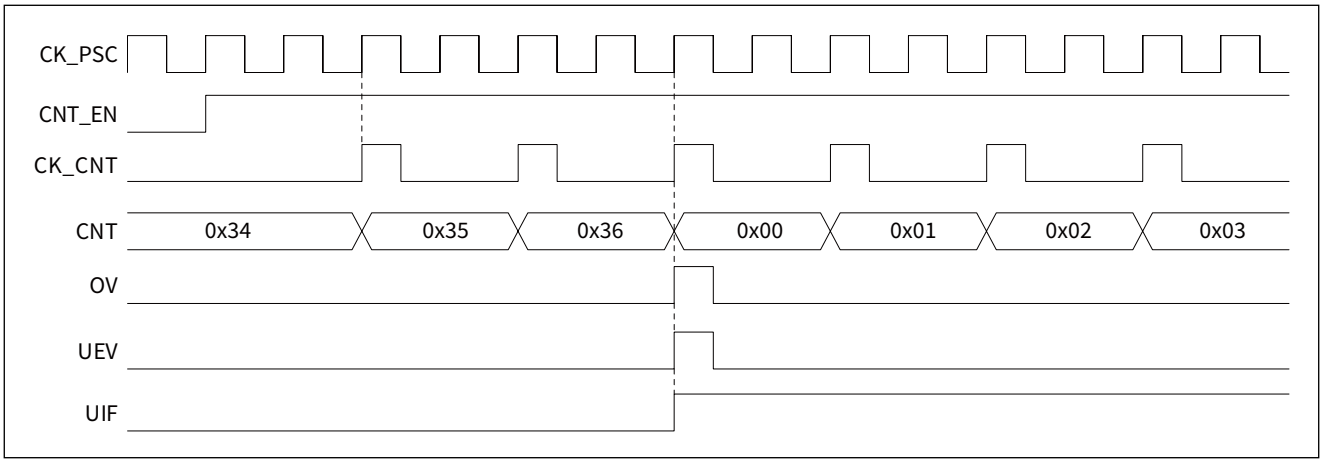


图 13-5　递增计数，内部时钟分频因子为 4

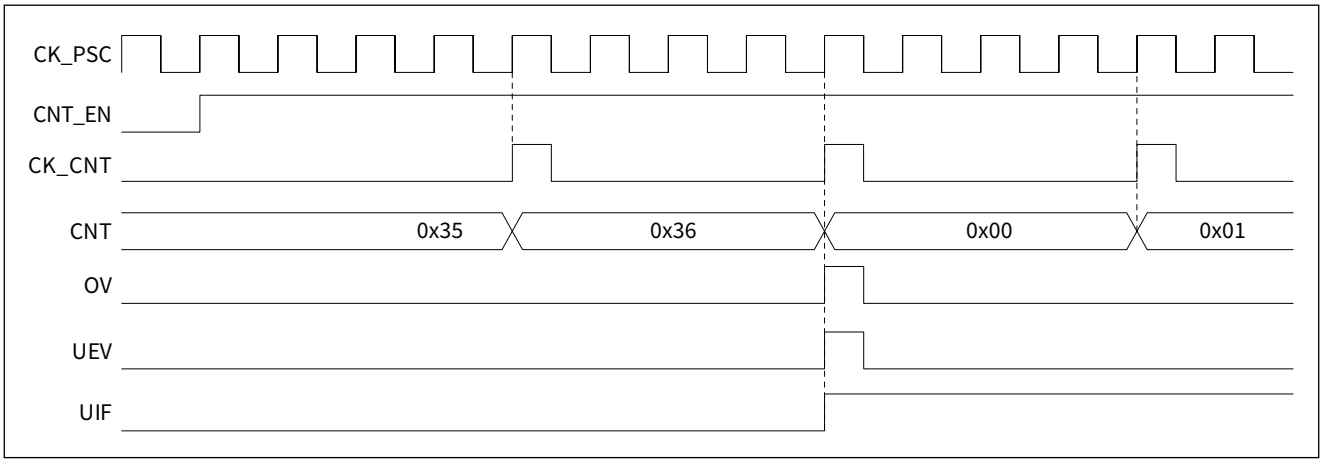


图 13-6　递增计数，内部时钟分频因子为 N

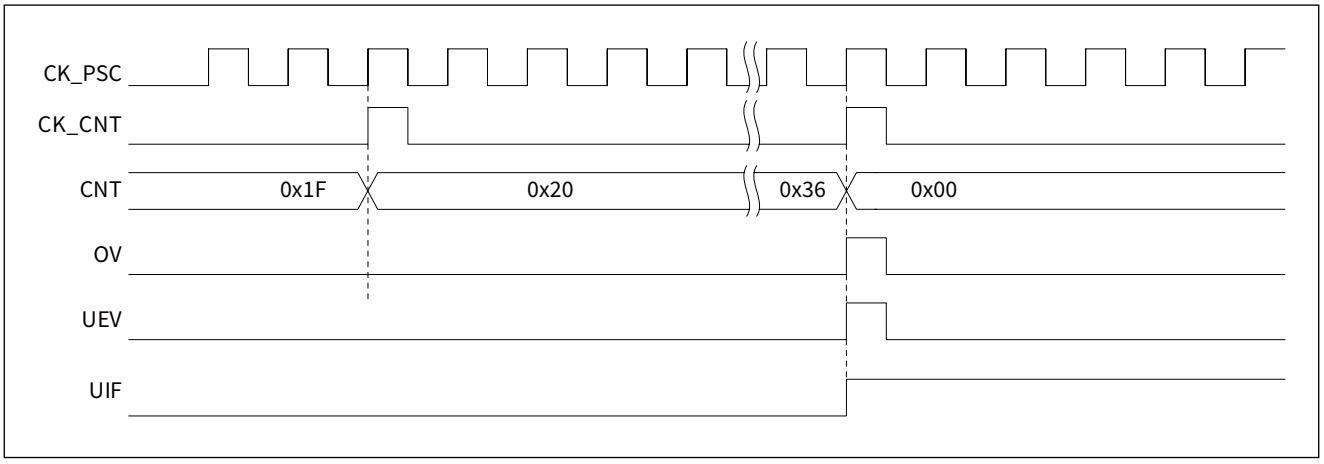


图 13-7　递增计数，当重载缓存禁止时的更新事件（GTIMx_CR1.ARPE=0)

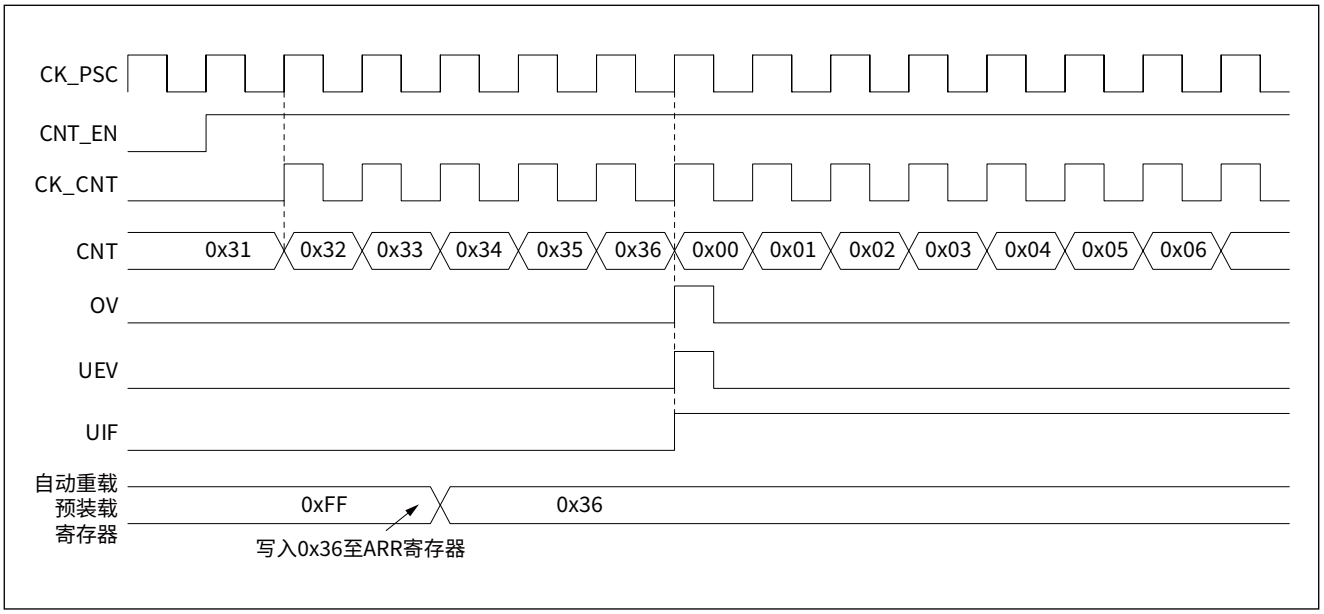
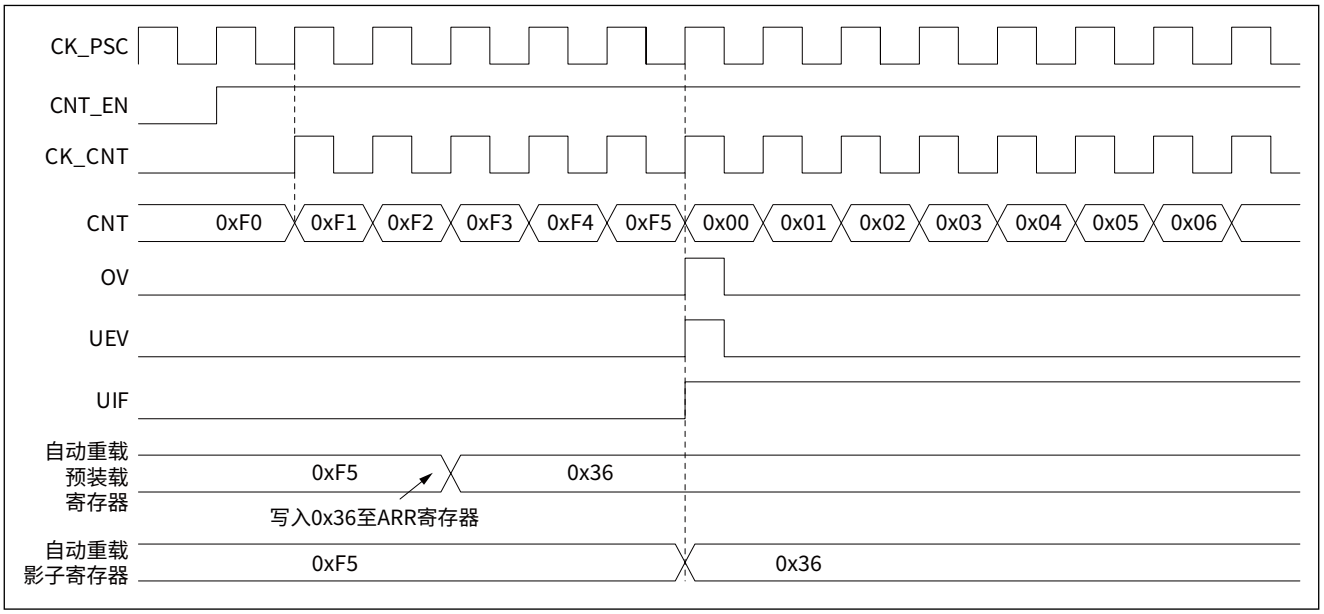


图 13-8　递增计数，当重载缓存使能时的更新事件（GTIMx_CR1.ARPE=1)



递减计数模式

在边沿对齐模式下设置控制寄存器 GTIMx_CR1 的 DIR 位为 1 时，计数器工作在递减计数模式。

在递减计数模式下，硬件自动加载 ARR 值到计数器 CNT，计数器开始递减计数到 0，然后重新装载 ARR 值递减计数，同时生成计数器下溢出事件。

当计数器下溢出时，产生下溢出信号 UND，同时产生更新事件 UEV（UND 信号和 UEV 信号会自动清除），计数器更新中断标志位 GTIMx_ISR.UIF 被硬件置位，如果允许中断（设置 GTIMx_IER.UIE 为 1），将产生中断请求，设置 GTIMx_ICR.UIF 为 0 清除该标志位。

以下是计数器在不同时钟频率下的操作示例，其中 GTIMx_ARR = 0x36：

图 13-9　递减计数，内部时钟分频因子为 1

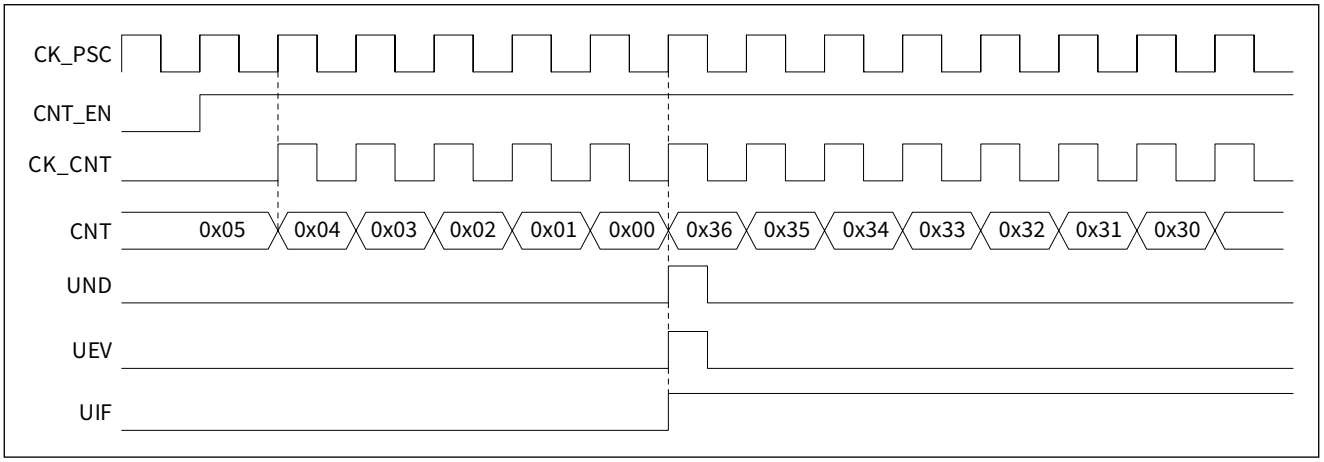


图 13-10　递减计数，内部时钟分频因子为 2

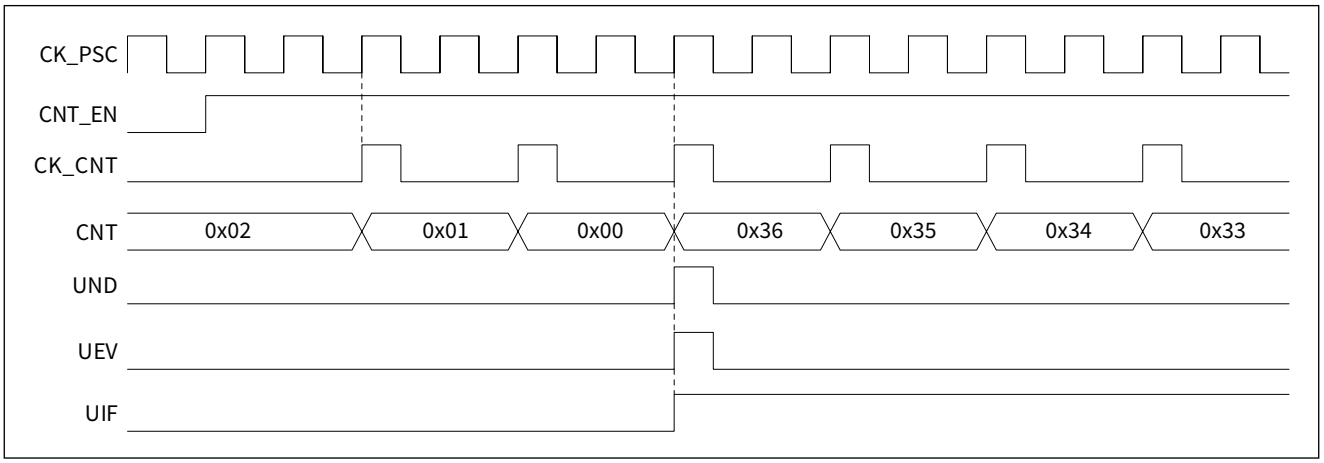


图 13-11　递减计数，内部时钟分频因子为 4

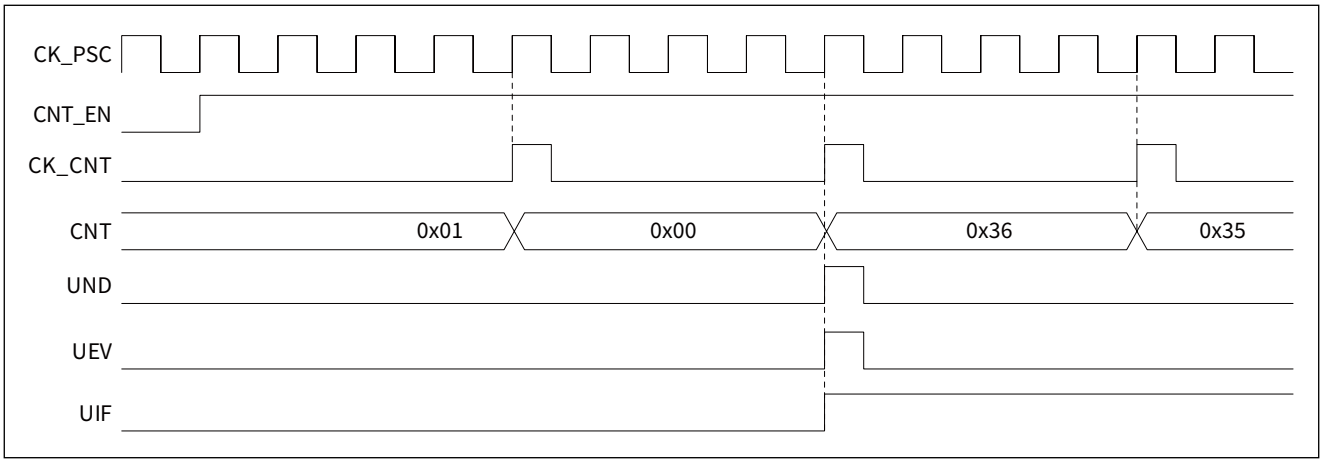
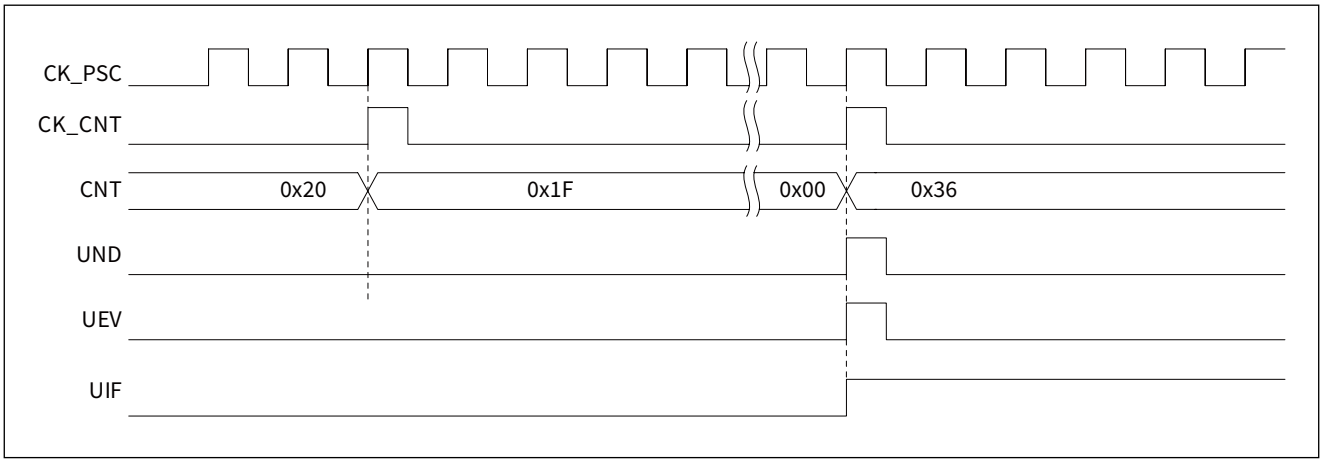


图 13-12　递减计数，内部时钟分频因子为 N



中心对齐模式（递增 / 递减计数）

在中心对齐模式下，计数器从 0 开始递增计数到 ARR-1，产生一个计数器上溢出事件，然后从 ARR 值开始递减计数到 1 并产生一个计数器下溢出事件，之后重新从 0 开始递增计数。

中心对齐模式包括中心对齐模式 1、中心对齐模式 2 和中心对齐模式 3，三种模式在计数方式上完全相同，只是输出比较中断标志的置位时机不同，具体请参见 13.9.1 GTIMx_CR1 控制寄存器 1 的 CMS 位域说明。

在中心对齐模式下，控制寄存器 GTIMx_CR1 的 DIR 位不能由软件写入，但可以读出，DIR 由硬件更新并指示当前的计数方向。启动中心对齐模式时，计数器将根据当前的 DIR 位域值进行递增或递减计数。不能同时通过软件修改 DIR 和 CMS 位域。

不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：

- 如果在计数器 CNT 中写入大于自动重载值 ARR 的值 (CNT>ARR)，则不会更新方向。即，如果计数器之前是递增计数，则继续递增计数。
- 如果向计数器 CNT 写入 0 或 ARR 值，计数方向会更新，但不生成更新事件 UEV。

使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新事件 UEV（设置 GTIMx_EGR.UG 为 1），并且不要在计数器运行过程中对其执行写操作。

以下是计数器在不同时钟频率下的操作示例：

图 13-13　中心对齐模式，内部时钟分频因子为 1，ARR=0x06

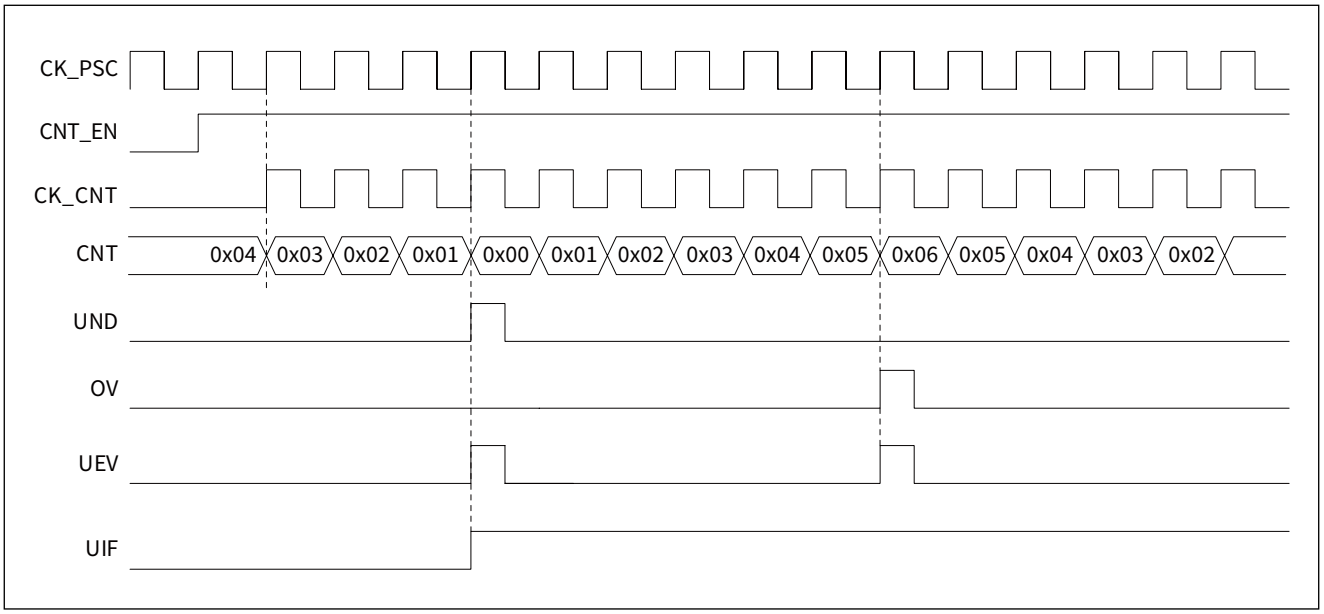


图 13-14　中心对齐模式，内部时钟分频因子为 2

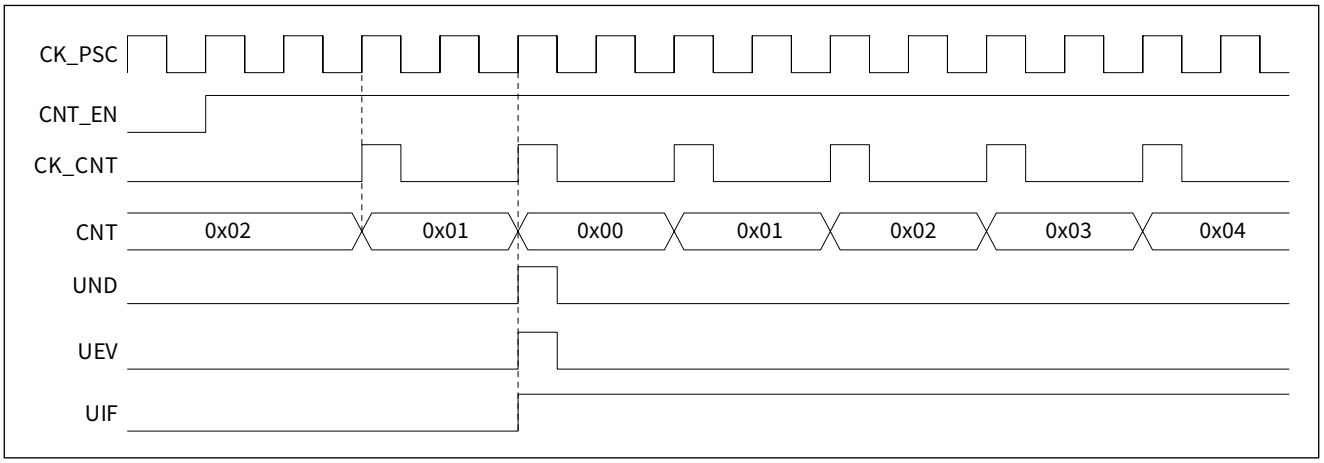


图 13-15　中心对齐模式，内部时钟分频因子为 4，ARR=0x36

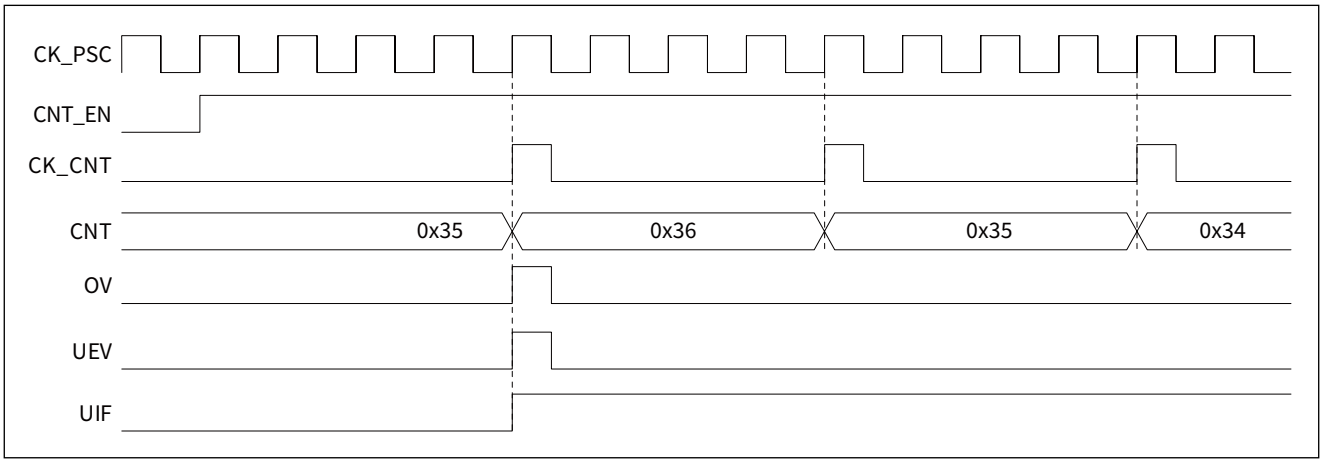


图 13-16　中心对齐模式，内部时钟分频因子为 N

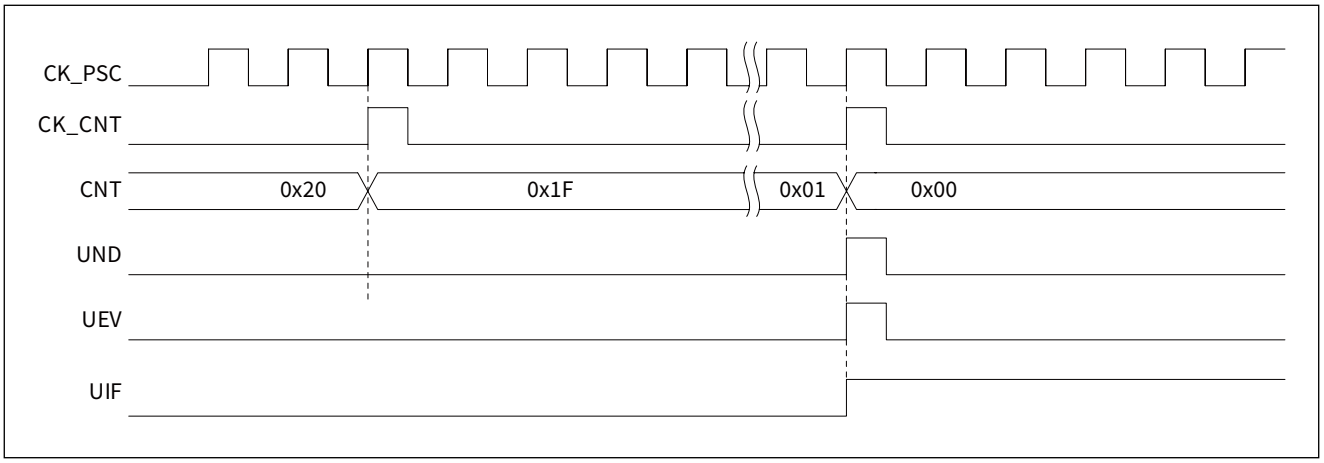


图 13-17　中心对齐模式，当重载缓存禁止时的更新事件（GTIMx_CR1.ARPE=0）

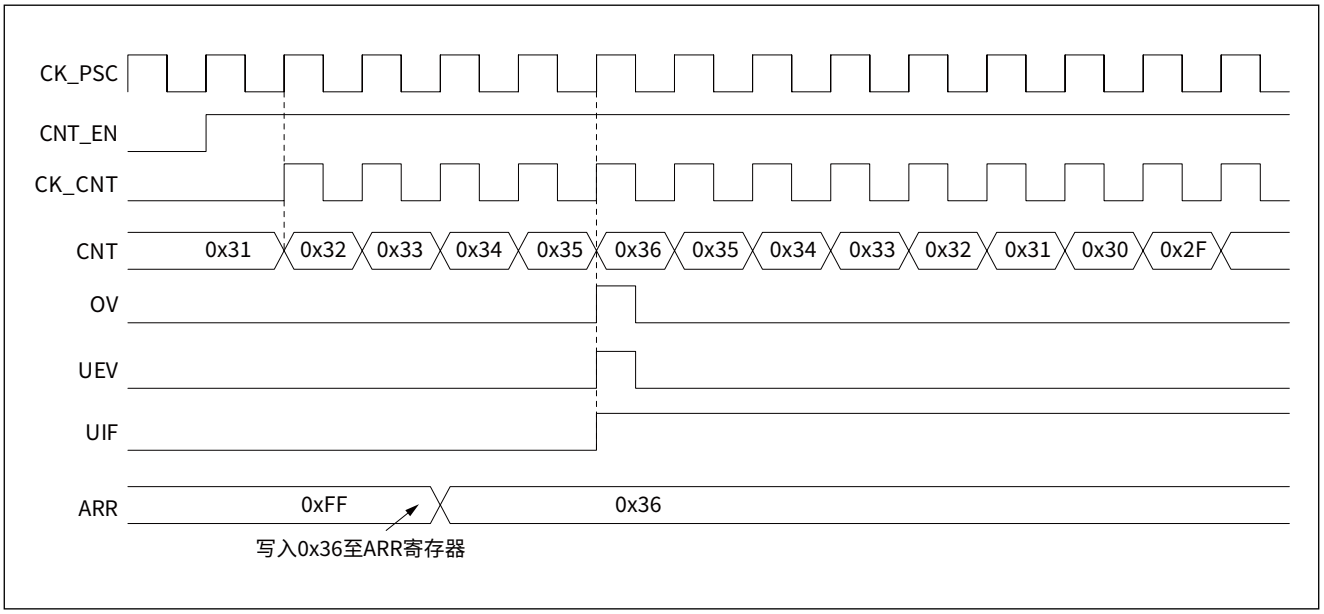
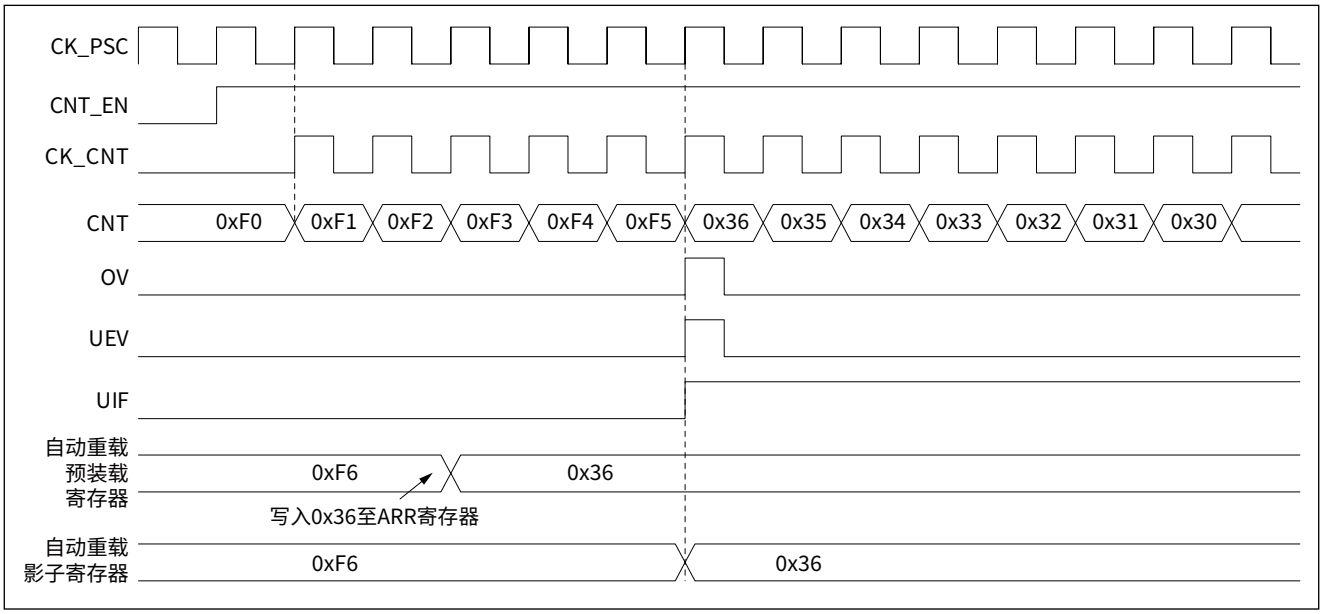


图 13-18　中心对齐模式，当重载缓存使能时的更新事件（GTIMx_CR1.ARPE=1）



13.3.1.4 重载寄存器

自动重载寄存器 GTIMx_ARR 具有缓存功能，通过控制寄存器 GTIMx_CR1 的 ARPE 位域开启或关闭。

当计数器处于停止状态或是缓存功能关闭时，更新重载寄存器 ARR 将会立即更新其影子寄存器。当定时器处于运行状态且缓存功能有效时，修改重载寄存器 ARR 将不会立即更新影子寄存器，仅当生成更新事件 UEV 时才会将重载寄存器 ARR 的值更新到影子寄存器中。

13.3.1.5 更新事件 UEV

允许通过控制寄存器 GTIMx_CR1 的 UDIS 位域来禁止或使能更新。

使能 UEV

设置 UDIS 位域为 0 使能 UEV，此时根据 URS 位域可选择更新请求源，如下表所示：

表 13-2 更新源设置

GTIMx_CR1.URS	更新源
0	计数器上溢出、下溢出； UG 置位； 通过从模式控制器生成的更新事件
1	计数器上溢出、下溢出

当发生更新事件时，将进行以下动作：

1. 重新初始化计数器：

   递减计数（边沿对齐模式）：重新加载自动重载值 GTIMx_ARR；

   中心对齐模式或递增计数（边沿对齐模式）：计数器清零。

2. 如果使用了缓存寄存器功能，将更新对应的预装载寄存器到其影子寄存器。
3. 预分频器的计数器被清零，GTIMx_PSC 中新的预分频值生效。
4. 事件更新中断标志位 GTIMx_ISR.UIF 被硬件置位。

当发生一个更新事件 UEV 时，事件更新中断标志位 GTIMx_ISR.UIF 会被硬件置位，如果允许中断（设置 GTIMx_IER.UIE 为 1），将产生一个更新中断请求，设置 GTIMx_ICR.UIF 为 0 可清除该标志位。

禁止 UEV

设置 UDIS 位域为 1 禁止 UEV，不再生成任何更新事件。

如果使用了缓存寄存器功能，对应的各影子寄存器的值保持不变。但如果 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器的计数器。

13.3.1.6 单脉冲模式

计数器可工作在单次计数或连续计数模式下，通过控制寄存器 GTIMx_CR1 的 OPM 位域来选择。

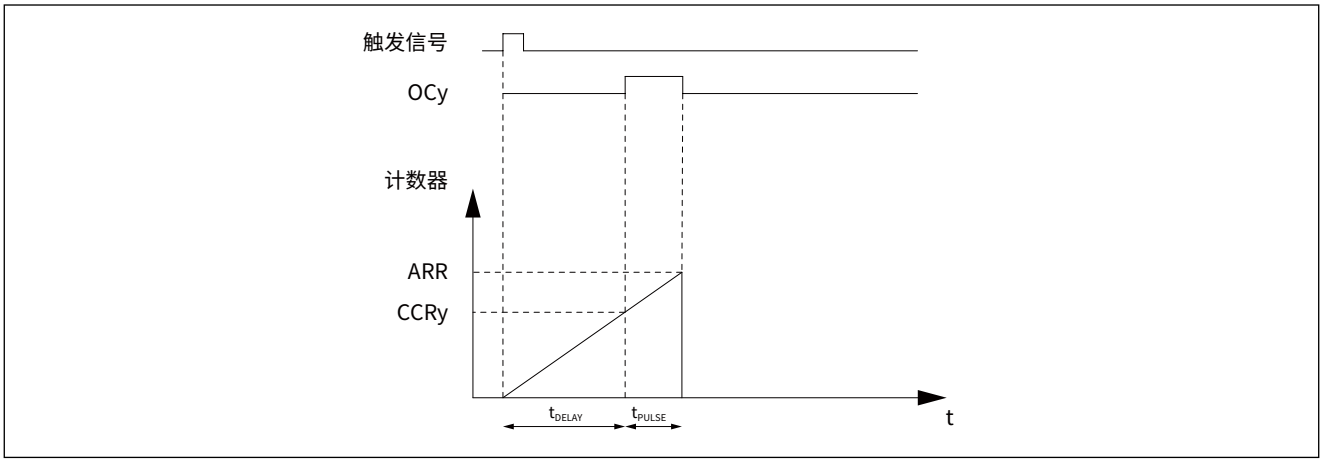
单次计数模式

设置 GTIMx_CR1.OPM 为 1，使定时器工作在单次计数模式下。

启动计数器后，计数器 CNT 在 CK_CNT 时钟的驱动下计数，在发生下一更新事件时，计数器停止计数，同时 GTIMx_CR1.CEN 被硬件自动清零。

可以通过从模式控制器启动计数器，配合定时器的输出比较模式，使得计数器在一个触发信号的触发下启动，并在一段可编程的延时后产生一个脉宽可编程的单脉冲。如下图所示，定时器在检测到触发信号的有效边沿时开始计数，延迟 t_{DELAY} 之后，在比较输出端口上产生一个宽度为 t_{PULSE} 的正脉冲。其中 t_{DELAY} 由计数寄存器 GTIMx_CNT 的初值和捕获 / 比较寄存器 GTIMx_CCRy 的差值确定，t_{PULSE} 由自动重载寄存器 GTIMx_ARR 和捕获 / 比较寄存器 GTIMx_CCRy 的差值来确定。

图 13-19 单脉冲输出示例（递增计数模式）



采样触发输入到激活 OC 输出需要多个时钟周期，可设置比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyFE 位域为 1，以缩短 OC 输出延迟时间。

连续计数模式

设置 GTIMx_CR1.OPM 为 0，使定时器工作在连续计数模式下，计数器在发生更新事件时不会停止计数。

13.3.1.7 外部触发输入通道

外部触发输入信号 ETR 可用作从模式控制器的触发输入 (TRGI)，也可用作计数器的计数时钟，具体请参见 13.3.2 工作模式。可对 ETR 信号进行输入控制，包括极性选择和边沿检测、预分频和滤波。

极性选择和边沿检测

从模式控制寄存器 GTIMx_SMCR 的 ETP 位域用于选择 ETR 输入信号的触发极性。当设置 ETP 为 0 时，ETR 不反相，高电平或上升沿有效；当 ETP 设置为 1 时，ETR 反相，低电平或下降沿有效。

预分频器

外部触发信号 ETRP 频率不得超过 PCLK 频率的 1/4。当 ETRP 的频率较高时，用户应通过适当的 ETPS 预分频器设置对外部信号进行分频，以降低 ETRP 频率，可设置分频系数为 1、2、4、8。

滤波器

滤波器采用数字滤波方式，以一定频率对输入信号进行采样，当连续采样到 N 个相同电平时信号有效，否则信号无效，以此滤除高频杂波信号。

ETR 输入信号的滤波器由从模式控制寄存器 GTIMx_SMCR 的 ETF 位域进行配置，可设置采样频率和采样点的个数。采样频率 $f_{SAMPLING}$ 由 f_{DTS} 分频后的时钟提供，f_{DTS} 是 f_{PCLK} 分频后得到的频率，分频因子由 GTIMx_CR1 寄存器的 CKD 位域配置，可设置 1、2、4 分频。

13.3.1.8 输入捕获通道

GTIM 支持 4 个输入捕获通道 TI1/2/3/4，可用作捕获命令。其中 TI1 和 TI2 还可用作从模式控制器的触发输入 (TRGI) 和编码器接口输入，具体请参见 13.3.2 工作模式。

支持对 TIy 信号进行输入控制，包括滤波、边沿检测和预分频。

滤波器

滤波阶段以一定频率对相应的 TIy 输入信号进行采样，生成滤波后信号 TIyF。

TIy 输入信号的滤波器由捕获模式寄存器 GTIMx_CCMR1CAP 和 GTIMx_CCMR2CAP 的 ICyF 位域进行配置，可设置采样频率和采样点的个数。采样频率 $f_{SAMPLING}$ 由 f_{PCLK} 或 f_{DTS} 分频后的时钟提供，f_{DTS} 是 f_{PCLK} 分频后得到的频率，分频因子由 GTIMx_CR1 寄存器的 CKD 位域配置，可设置 1、2、4 分频。

边沿检测

带有极性选择功能的边沿检测器可生成一个 TIxFPx 信号，具体有效极性请参见 13.9.12 GTIMx_CCER 捕获 / 比较使能寄存器的 CCyNP 和 CCyP 位域。

预分频器

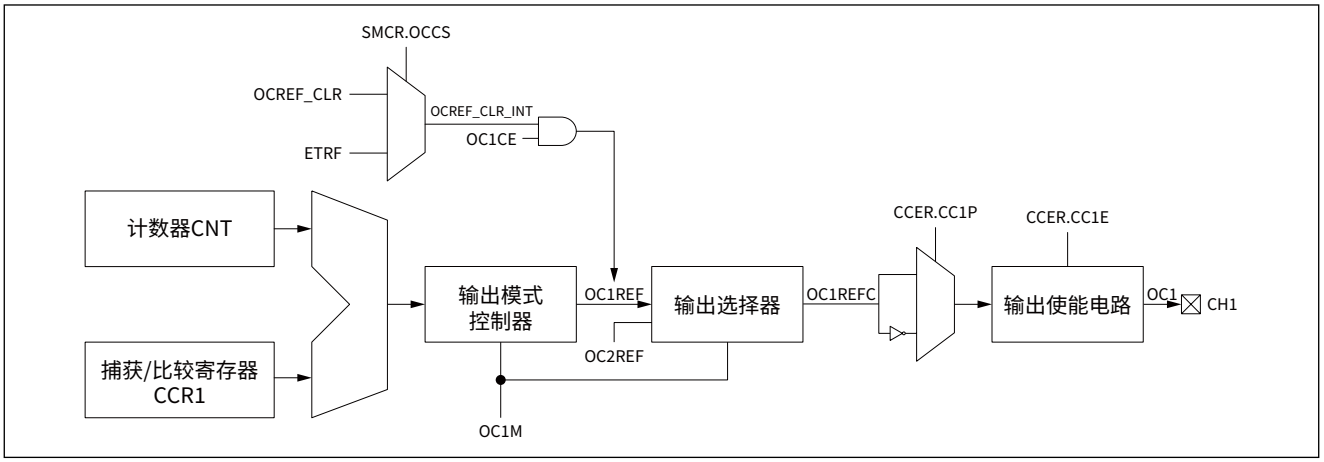
TIy 信号用于输入捕获时，可对捕获通道信号 ICy 进行分频，通过捕获模式寄存器 GTIMx_CCMR1CAP 和 GTIMx_CCMR2CAP 的 ICyPSC 位域进行控制，支持 1、2、4、8 分频。

13.3.1.9 输出比较通道

GTIM 支持 4 个独立输出比较通道 CH1/2/3/4，每个通道均由一个捕获 / 比较寄存器（包括一个影子寄存器）和一个输出阶段（比较器和输出控制）组成。

输出控制单元用于比较匹配时，控制输出端口的波形，可配置输出比较模式、输出极性选择和输出使能等。以比较通道 1 为例，其框图如下图所示：

图 13-20 输出比较 1 通道



13.3.2 工作模式

GTIM 支持多种工作模式，具体由从模式控制寄存器 GTIMx_SMCR 的 SMS 位域来配置，如下表所示：

表 13-3 GTIM 工作模式

GTIMx_SMCR.SMS 值	从模式选择
0000	禁止从模式，使用内部时钟
0111	外部时钟模式 1
0100	复位模式
0101	门控模式
0110	触发模式
1000	组合复位 + 触发模式
1001	组合门控 + 复位模式
1110	正交编码器模式，x1 模式
1111	正交编码器模式，x1 模式
0001	正交编码器模式，x2 模式
0010	正交编码器模式，x2 模式
0011	正交编码器模式，x4 模式
1010	编码模式（时钟 + 方向），x2 模式
1011	编码模式（时钟 + 方向），x1 模式
1100	编码模式（带方向时钟），x2 模式
1101	编码模式（带方向时钟），x1 模式

注：

1. 从模式选择位包括 GTIMx_SMCR[16] 和 GTIMx_SMCR[2:0]，需组合配置。

2. 从模式选择位 SMS[3:0] 支持预装载功能，请参见 13.9.3 GTIMx_SMCR 从模式控制寄存器的 SMSPE 和 SMSPS 位域说明。

13.3.2.1 内部时钟模式

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x0 时，禁止定时器从模式，预分频器时钟直接由内部时钟 PCLK 提供。设置 GTIMx_CR1.CEN 为 1，将使能计数器开始计数。

13.3.2.2 外部时钟模式

外部时钟模式 1

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x7 时，由所选触发信号 (TRGI) 的上升沿提供计数器时钟。TRGI 信号有多种触发选择，具体通过 GTIMx_SMCR 寄存器的 TS 位域进行选择，如下表所示：

表 13-4 TRGI 信号来源

GTIMx_SMCR.TS 位域值	TRGI 信号来源
00111	外部触发输入（ETRF）
000xx/01xxx/10000	内部触发（ITR0~12）
00100	TI1 边沿检测器（TI1F_ED）
00101	滤波后的定时器输入 1（TI1FP1）
00110	滤波后的定时器输入 2（TI2FP2）

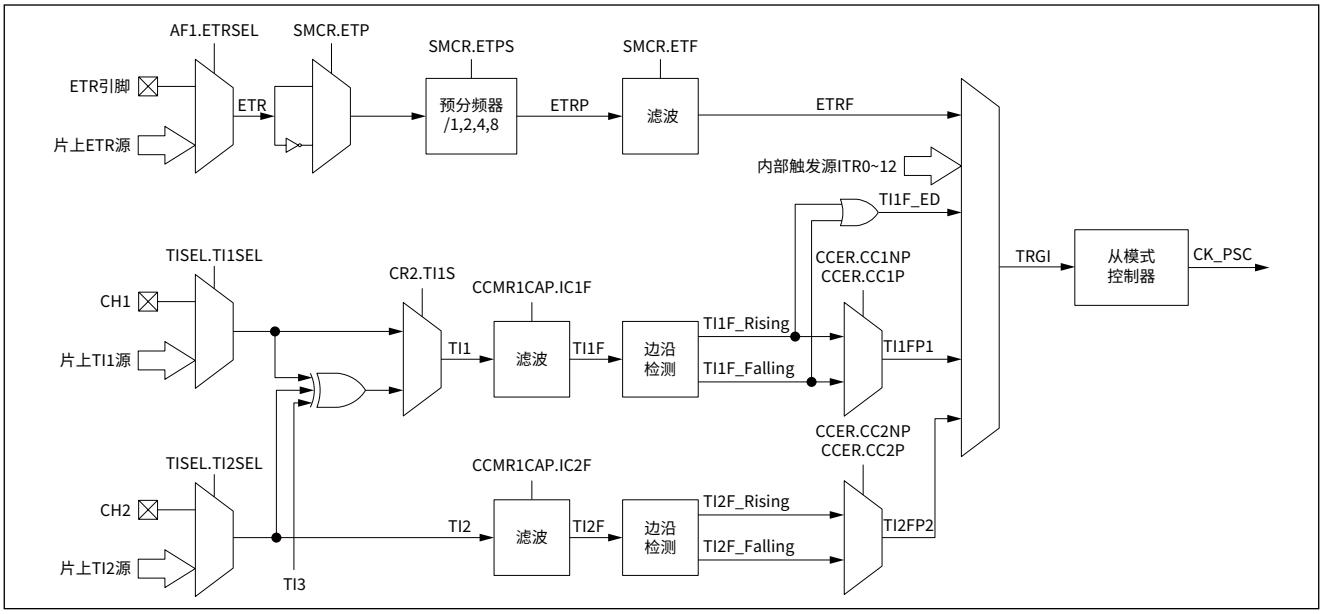
ETR 输入信号来源可以是外部 GTIMx_ETR 引脚，也可以是片内其它外设，请参见 13.3.7 片内外设互联 ETR。选择 ETR 为 TRGI 信号源时，可通过 GTIMx_SMCR.ETP 选择外部触发极性，通过 GTIMx_SMCR.ETPS 设置预分频，通过 GTIMx_SMCR.ETF 进行滤波控制。

内部触发 ITR 来源为 BTIM 和 ATIM 的触发输出信号 TRGO 以及 UART 的 TXD/RXD 信号，请参见 13.3.6 定时器级联 ITR。

TI1 和 TI2 都具有滤波和边沿检测功能，TI1 和 TI2 分别通过捕获模式寄存器 GTIMx_CCMR1CAP 的 IC1F 和 IC2F 位域进行滤波控制，分别通过捕获 / 比较使能寄存器 GTIMx_CCER 的 CC1P/CC1NP 和 CC2P/CC2NP 位域进行边沿检测。

外部时钟模式 1 连接示意图如下图所示：

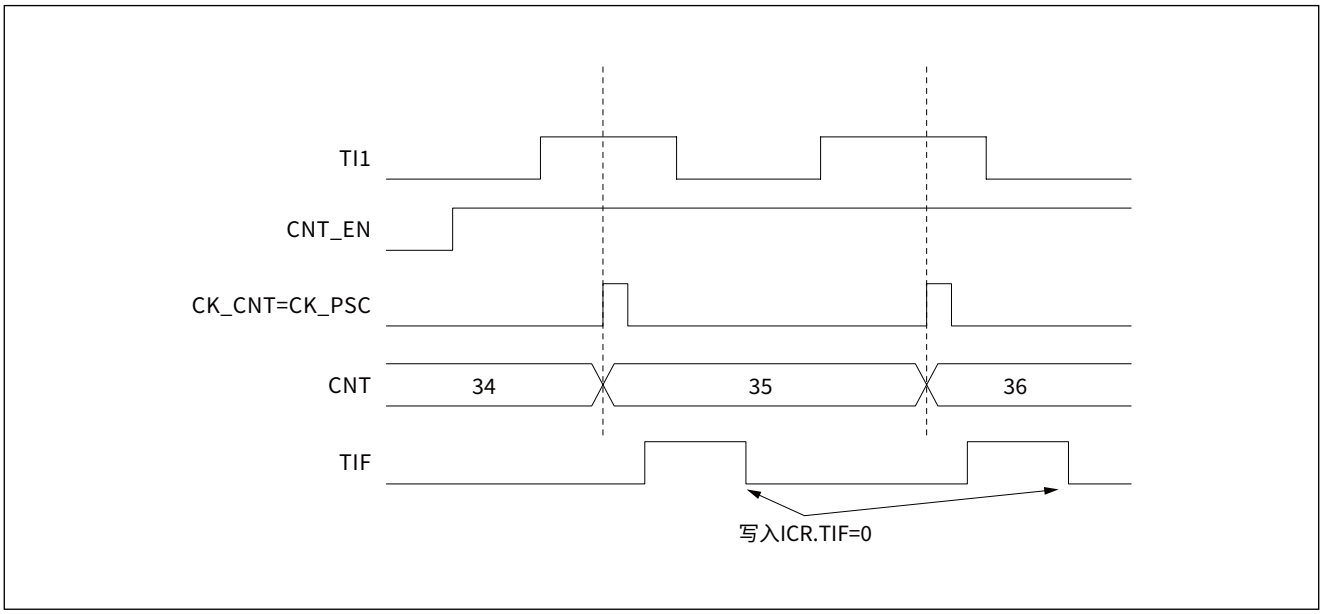
图 13-21 外部时钟模式 1 连接图



当 TRGI 出现有效边沿时，GTIMx_ISR.TIF 标志将置 1，向 GTIMx_ICR.TIF 写 0 可清除该标志。

下图所示为配置 TI1 上升沿的外部时钟模式 1 时序图，TI1 的上升沿与实际计数器时钟之间的延迟是由于 TI1 输入的重新同步电路引起的。

图 13-22　外部时钟模式 1 时序示例



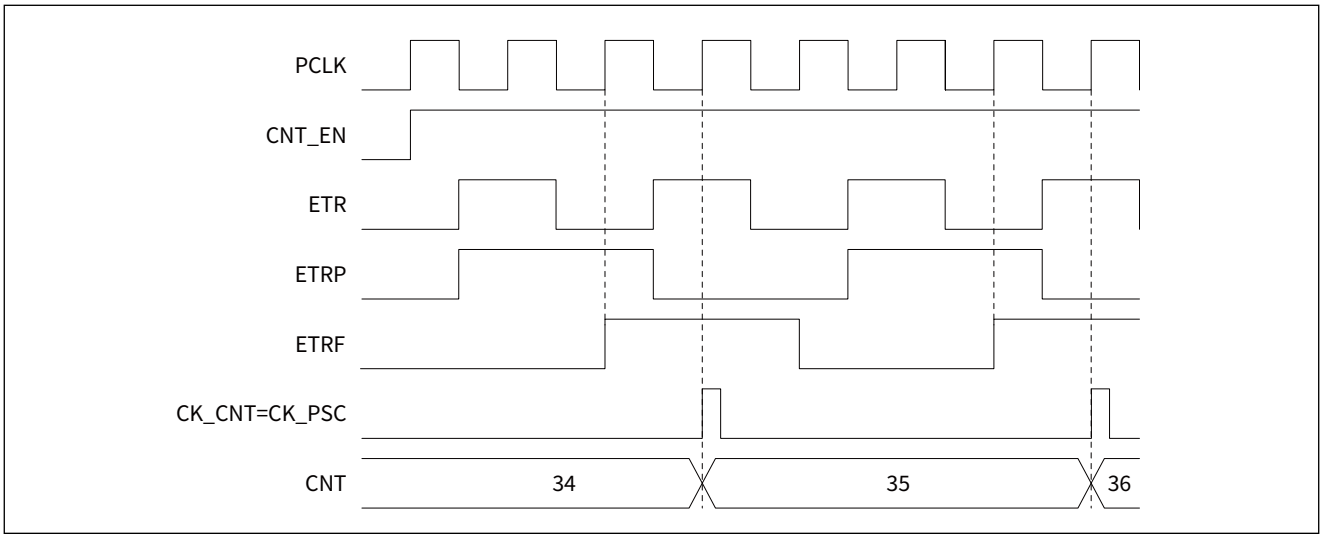
外部时钟模式 2

设置从模式控制寄存器 GTIMx_SMCR 的 ECE 位域为 1，使能外部时钟模式 2。该模式下，计数器时钟由 ETRF 信号的任意有效边沿提供，与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=0111 且 TS=00111）具有相同效果，连接框图参见图 13-21 外部时钟模式 1 连接图。

外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式，此时从模式下的 TRGI 不得连接 ETRF（即 TS 位域不能设置为 0x7）。如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。外部时钟模式 2 不能与编码器模式同时使用。

下图所示为 ETR 设置为 2 分频时的外部时钟模式 2 的时序示例，ETR 的上升沿与实际计数器时钟之间的延迟是由 ETRP 信号的重新同步电路引起的，因此计数器可以正确捕获的最大频率最高为内部时钟 PCLK 频率的 1/4，当 ETRP 信号变快时，用户应对外部信号进行适当的分频。

图 13-23　外部时钟模式 2 时序示例



13.3.2.3 复位模式

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x4 时，GTIM 配置为复位模式，计数器的复位由 TRGI 信号控制。

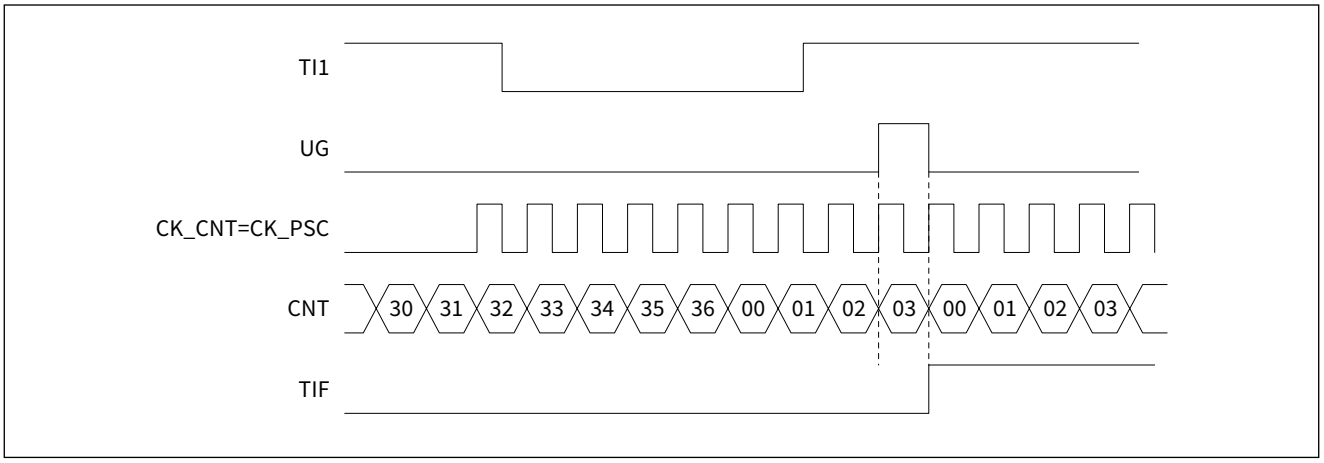
TRGI 信号的来源由从模式控制寄存器 GTIMx_SMCR 的 TS 位域控制，可参见表 13-4 TRGI 信号来源。

当检测到有效的触发输入（TRGI）时，将产生以下影响：

- 重新初始化计数器和预分频器的计数器。
- 如果控制寄存器 GTIMx_CR1 的 URS 位域为 0，则会产生更新事件 UEV，事件更新中断标志 GTIMx_ISR.UIF 置 1，可产生中断请求。
- 触发中断标志 GTIMx_ISR.TIF 置 1，可产生中断请求。

下图所示为复位模式时序图示例。设置 GTIMx_CR1.CEN 为 1 使能计数器，计数器根据计数时钟 CK_CNT 正常计数，当 TI1 出现上升沿时，计数器清零并重新从 0 开始计数，同时 GTIMx_ISR.TIF 标志位置 1。TI1 的上升沿与实际计数器复位之间的延迟是由 TI1 输入的重新同步电路引起的。

图 13-24 复位模式时序



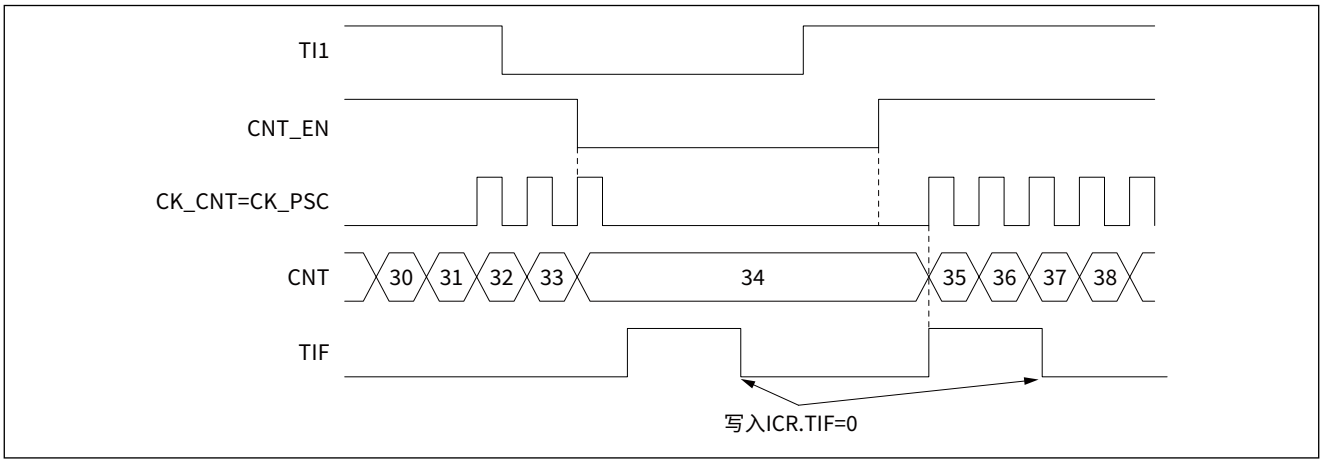
13.3.2.4 门控模式

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x5 时，GTIM 配置为门控模式。在该模式下，触发输入 (TRGI) 为高电平且 GTIMx_CR1.CEN 为 1 时，启动计数器计数；触发输入 (TRGI) 为低电平或 GTIMx_CR1.CEN 为 0 时，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。

TRGI 信号的来源由从模式控制寄存器 GTIMx_SMCR 的 TS 位域控制，可参见表 13-4 TRGI 信号来源，但需注意门控模式下不能选择 TI1F_ED 作为触发输入。

下图所示为门控模式时序图示例。当 TI1 为高电平时，计数器启动计数；当 TI1 为低电平时，计数器暂停计数。计数器启动和停止时，GTIMx_ISR.TIF 标志位都会置 1。TI1 的边沿与实际计数器使能信号 CNT_EN 之间的延迟是由 TI1 输入的重新同步电路引起的。

图 13-25 门控模式时序



13.3.2.5 触发模式

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x6 时，GTIM 配置为触发模式。在该模式下，设置 GTIMx_CR1.CEN 为 1 或触发信号 TRGI 出现上升沿时，触发启动计数器计数（但不复位）。

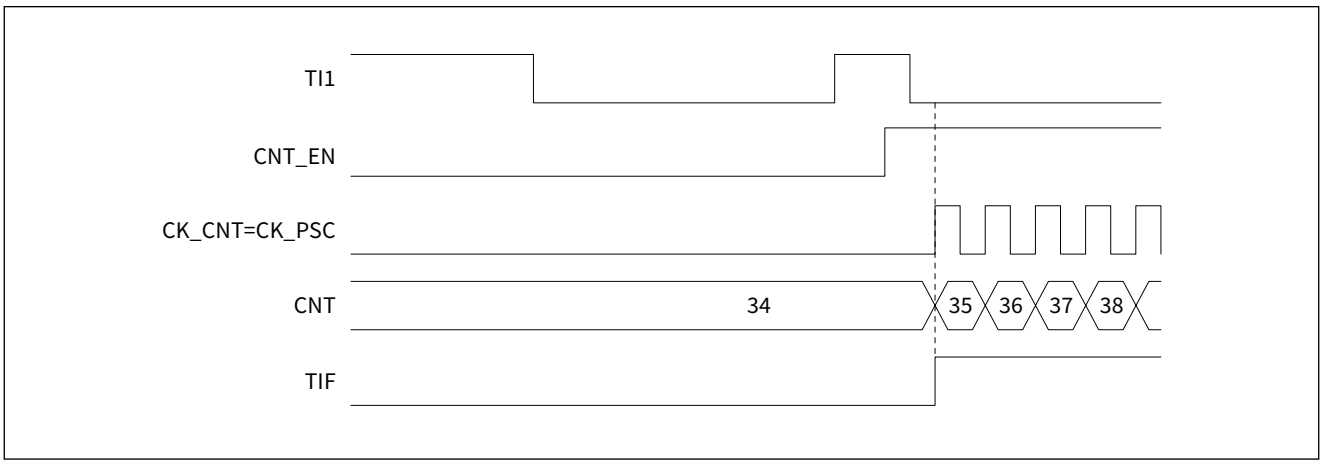
TRGI 信号的来源由从模式控制寄存器 GTIMx_SMCR 的 TS 位域控制，可参见表 13-4 TRGI 信号来源。

当检测到有效的触发信号时，将产生以下影响：

- GTIMx_CR1.CEN 被硬件置位。
- 触发中断标志位 GTIMx_ISR.TIF 置 1，可产生中断请求。
- 计数器启动，开始计数。

下图所示为触发模式时序图示例。当 TI1 出现上升沿时，计数器启动计数，同时 GTIMx_ISR.TIF 标志位置 1。TI1 的上升沿与实际计数器启动之间的延迟是由 TI1 输入的重新同步电路引起的。

图 13-26 触发模式时序



13.3.2.6 组合复位 + 触发模式

当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x8 时，GTIM 配置为组合复位 + 触发模式。在该模式下，出现所选触发输入 (TRGI) 上升沿时，将重新初始化计数器并启动计数器，同时触发中断标志 GTIMx_ISR.TIF 置 1。如果控制寄存器 GTIMx_CR1 的 URS 位域为 0，则会产生更新事件 UEV，事件更新中断标志 GTIMx_ISR.UIF 会被硬件置位。

TRGI 信号的来源由从模式控制寄存器 GTIMx_SMCR 的 TS 位域控制，可参见表 13-4 TRGI 信号来源。

13.3.2.7 组合门控 + 复位模式

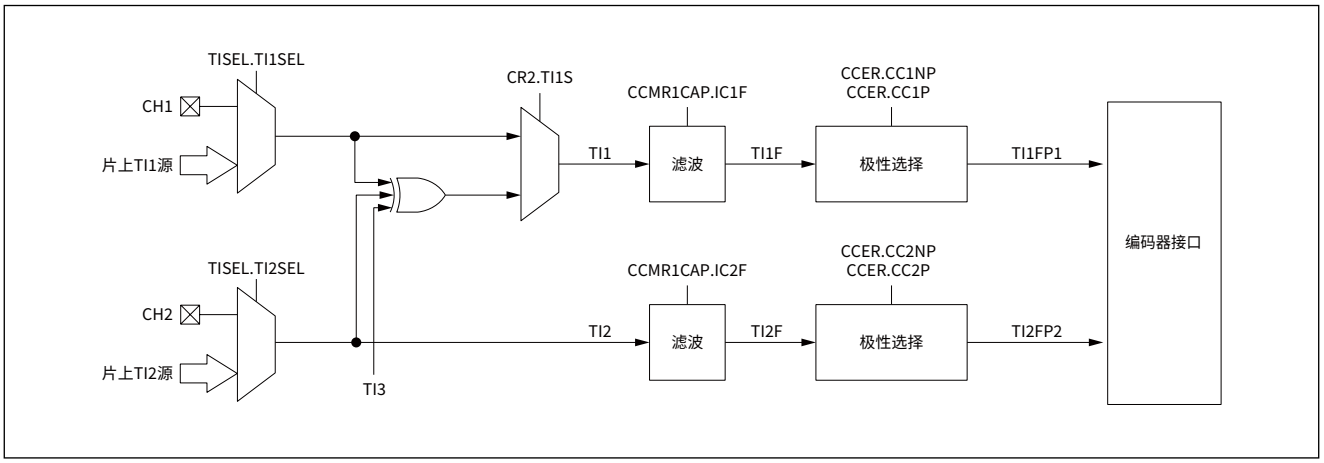
当从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0x9 时，GTIM 配置为组合门控 + 复位模式。在该模式下，计数器的启动和停止都被控制，触发输入 (TRGI) 为高电平且 GTIMx_CR1.CEN 为 1 时，启动计数器计数；触发输入 (TRGI) 为低电平时，计数器立即停止计数，并被复位。计数器启动和停止时，GTIMx_ISR.TIF 标志位都会置 1；计数器复位时，GTIMx_ISR.UIF 标志位置 1。

TRGI 信号的来源由从模式控制寄存器 GTIMx_SMCR 的 TS 位域控制，可参见表 13-4 TRGI 信号来源。

13.3.2.8 正交编码器模式

GTIM 支持正交编码器模式，用于接收并解码正交编码器的信号。在该模式下，允许通过 CH1、CH2 引脚与外部的正交编码器直接连接，根据输入信号的跳变顺序，实现计数器自动递增或递减计数，计数器值始终表示编码器的位置。其功能框图如下图所示：

图 13-27 编码器模式框图



CH1 和 CH2 输入信号具有滤波和极性选择功能，分别通过 GTIMx_CCMR1CAP 寄存器的 IC1F 和 IC2F 位域进行滤波控制，通过 GTIMx_CCER 寄存器的 CC1P 和 CC2P 位域选择输入极性，CC1NP 和 CC2NP 位域必须保持清零。

设置 GTIMx_CR1.CEN 为 1 使能计数器，计数器将由通道 CH1 和 CH2 引脚输入信号经滤波和极性选择后的信号 TI1FP1 和 TI2FP2 的每次有效跳变驱动，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，参见表 13-5 计数方向和编码器信号的关系。编码器当前计数方向标志 GTIMx_CR1.DIR 由硬件自动设置和清除，且在任何输入（CH1 或 CH2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 TI1FP1 或 TI2FP2 边沿处计数，还是同时在 TI1FP1 和 TI2FP2 处计数。

GTIM 支持多种正交编码计数模式，通过从模式控制寄存器 GTIMx_SMCR 的 SMS 位域进行设置，不同模式下计数方向和编码器信号的关系如下表所示：

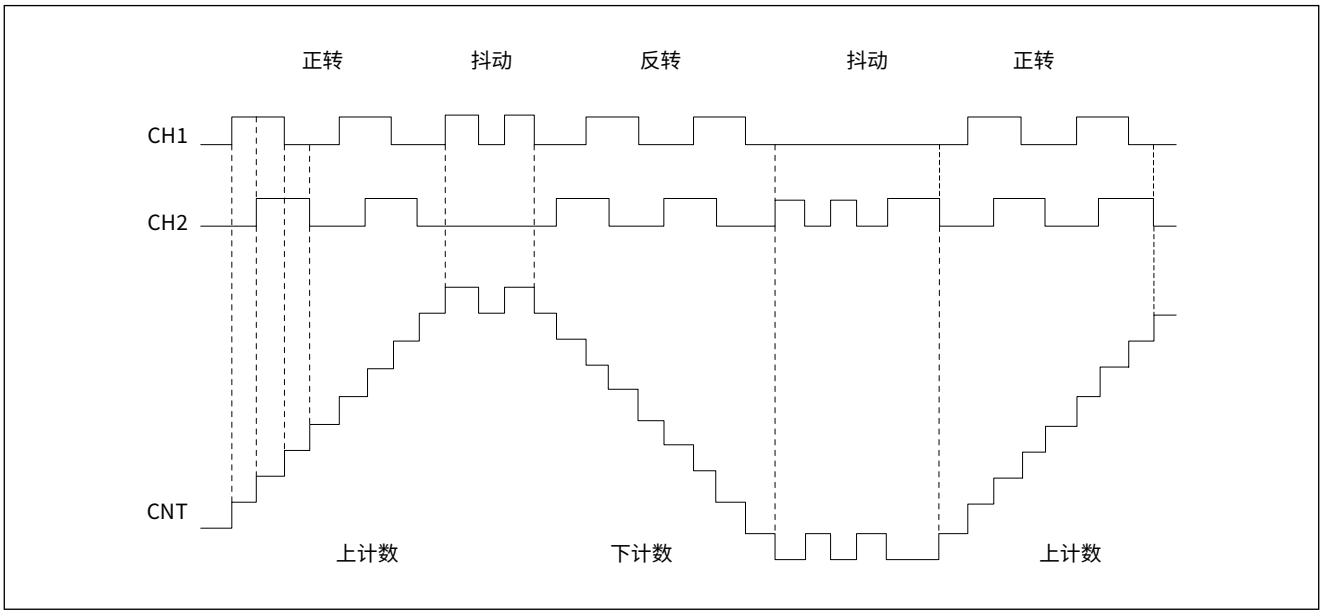
表 13-5 计数方向和编码器信号的关系

模式	SMS	信号的电平		TI1FP1		TI2FP2	
		TI2FP2	TI1FP1	上升	下降	上升	下降
在 TI1FP1 边沿计数（x1 模式）	1110	高	-	向下计数	向上计数	不计数	不计数
		低	-	不计数	不计数	不计数	不计数
在 TI2FP2 边沿计数（x1 模式）	1111	-	高	不计数	不计数	向上计数	向下计数
		-	低	不计数	不计数	不计数	不计数
在 TI1FP1 边沿计数（x2 模式）	0001	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
在 TI2FP2 边沿计数（x2 模式）	0010	-	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 TI1FP1 和 TI2FP2 边沿计数（x4 模式）	0011	高	高	向下计数	向上计数	向上计数	向下计数
		低	低	向上计数	向下计数	向下计数	向上计数

编码器输出的第三个信号表示机械零点，可以把它连接到外部触发输入引脚上，用以触发计数器复位。

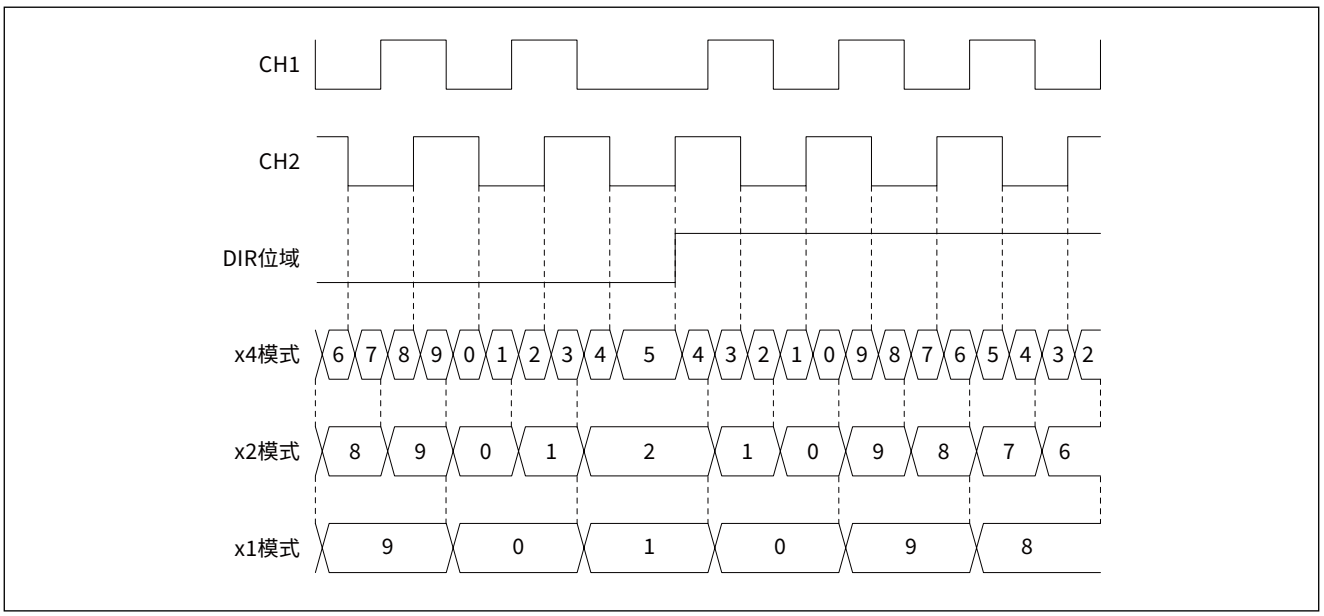
下图是一个正交编码计数模式 -x4 模式的操作实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。

图 13-28　正交编码器模式 -x4 模式操作示例



下图显示了各种正交编码计数模式下编码器反转期间的定时器计数值，示例中 CH1 和 CH2 输入均不反相。

图 13-29　各正交编码器模式计数示例



编码器模式可以提供传感器当前位置的相关信息。计数器根据编码器输出的脉冲自动进行递增或递减计数，当前计数器值即表示编码器当前位置。使用另一个配置为捕获模式的定时器，可以捕获编码器信号的上升沿或下降沿，并记录其时间戳，通过测量两个事件之间的时间间隔，可以计算出相应的动态信息，如速度、加速度和减速度；可以使用指示机械零位的编码器输出来实现此目的，通过测量从机械零位到其他位置的时间间隔，可以计算出相对于零位的位置和速度变化。

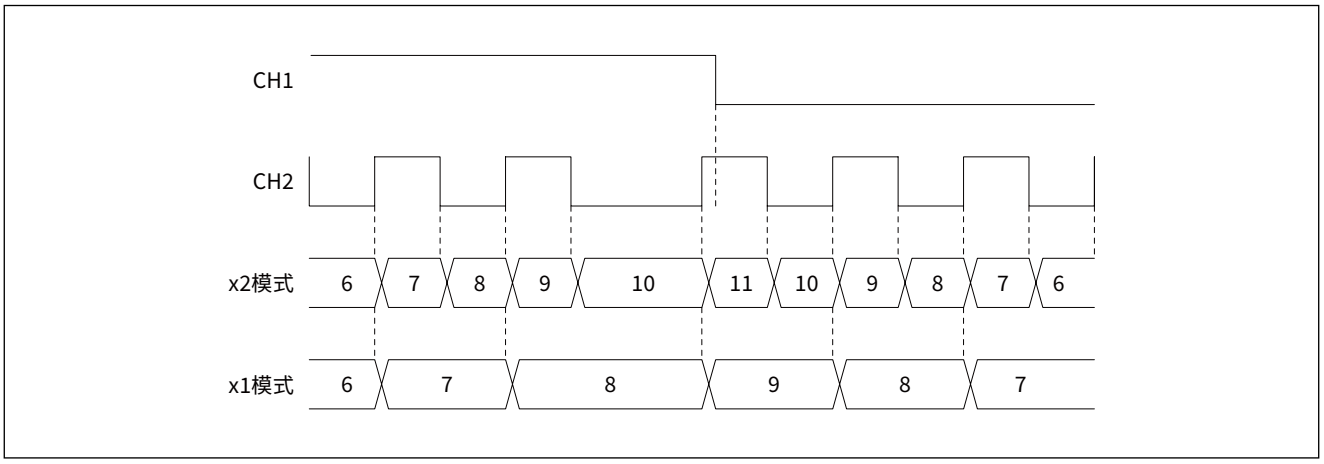
13.3.2.9 时钟加方向编码器模式

定时器还支持时钟加方向的编码器模式，需配置从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0xA 或 0xB，分别对应 x2 模式和 x1 模式。其功能框图可参见图 13-27 编码器模式框图，可对输入信号进行滤波和极性控制。

带方向的编码器通常具有两个输出信号：时钟信号和方向信号，使用时钟信号可以计算出旋转编码器的旋转速度和位置变化，而方向信号可以用来确定旋转的方向。在 x2 模式，计数器在时钟信号的上升沿和下降沿计数；在 x1 模式，计数器根据 CC2P 位域的值在单个时钟边沿计数，CC2P 为 0 对应上升沿敏感，CC2P 为 1 对应下降沿敏感。CH1 通道上方向信号的极性由 CC1P 位设置：CC1P 为 0 对应正极性（当 CH1 为高电平时递增计数，当 CH1 为低电平时递减计数），CC1P 为 1 对应负极性（当 CH1 为低电平时递增计数，当 CH1 为高电平时递减计数）。

下图显示了时钟加方向编码器模式下的计数器计数实例，示例中 CH1 和 CH2 输入均不反相：

图 13-30 时钟加方向编码器模式计数示例



13.3.2.10 定向时钟编码器模式

定时器还支持定向时钟编码器模式，需配置从模式控制寄存器 GTIMx_SMCR 的 SMS 位域为 0xC 或 0xD，分别对应 x2 模式和 x1 模式。其功能框图可参见图 13-27 编码器模式框图，可对输入信号进行滤波和极性控制。

定向时钟编码器会根据旋转方向分别提供一条向上计数时钟线和向下计数时钟线。在 x2 模式，计数器在两个时钟线中的任意一个的上升沿和下降沿计数；CC1P 和 CC2P 位是时钟空闲状态的编码，CCyP 为 0 对应高电平空闲，CCyP 为 1 对应低电平空闲。在 x1 模式，计数器根据 CC1P 和 CC2P 位域的值在单个时钟边沿计数，CCyP 为 0 对应下降沿敏感和高电平空闲，CCyP 为 1 对应上升沿敏感和低电平空闲。

不同模式下计数方向和编码器信号的关系如下表所示：

表 13-6　计数方向和编码器信号的关系

模式	SMS	信号的电平		TI1FP1		TI2FP2	
		TI2FP2	TI1FP1	上升	下降	上升	下降
x2 模式 CCyP=0	1100	高	高	向下计数	向下计数	向上计数	向上计数
		低	低	不计数	不计数	不计数	不计数
x2 模式 CCyP=1		高	高	不计数	不计数	不计数	不计数
		低	低	向下计数	向下计数	向上计数	向上计数
x1 模式 CCyP=0	1101	高	高	不计数	向下计数	不计数	向上计数
		低	低	不计数	不计数	不计数	不计数
x1 模式 CCyP=1		高	高	不计数	不计数	不计数	不计数
		低	低	向下计数	不计数	向上计数	不计数

下图是定向时钟编码器模式的计数示例：

图 13-31　定向时钟编码器模式计数示例 (CC1P = CC2P = 0)

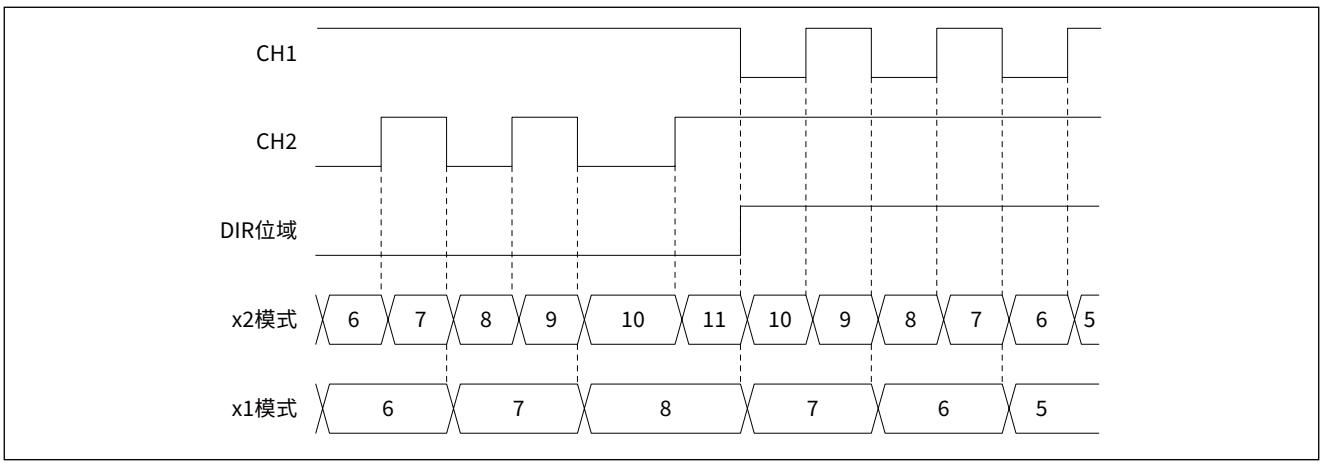
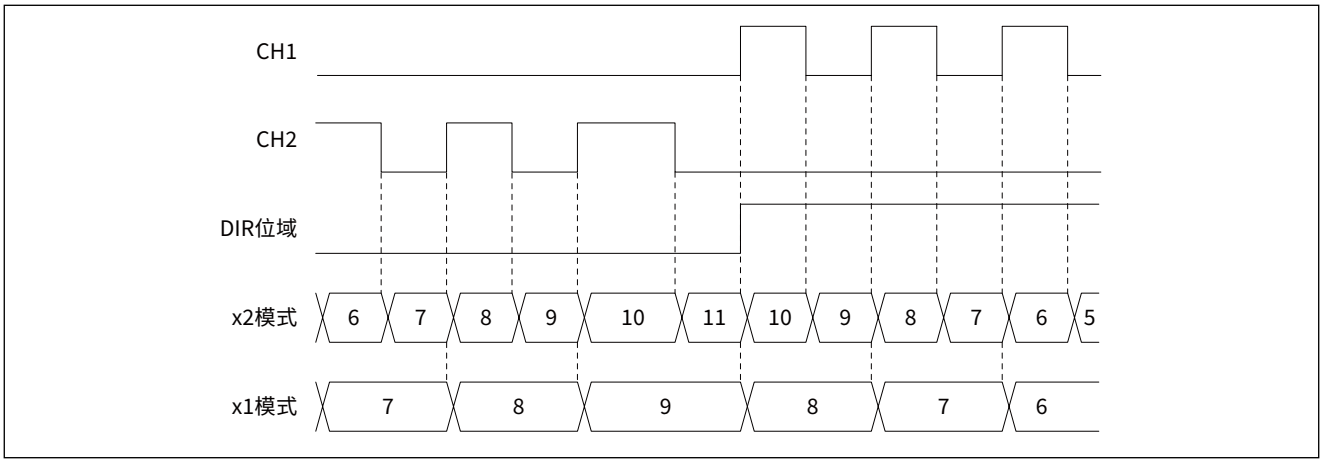


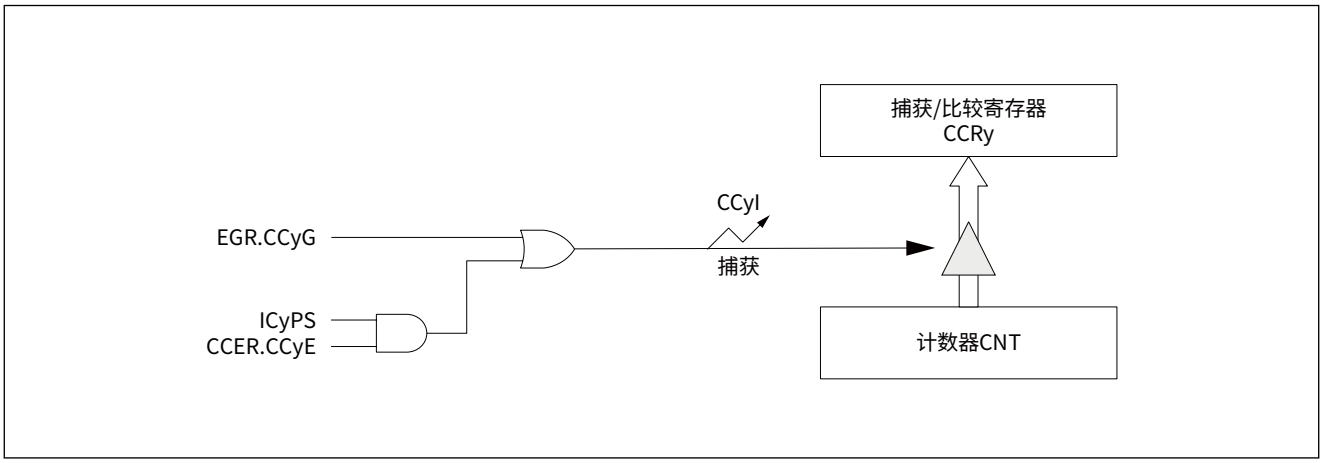
图 13-32　定向时钟编码器模式计数示例 (CC1P = CC2P = 1)



13.3.3 输入捕获功能

GTIM 支持输入捕获功能，设置捕获模式寄存器（GTIMx_CCMR1CAP 和 GTIMx_CCMR2CAP）的 CCyS 位域为非零值时，CCy 通道配置为输入，同时指定对应捕获通道 ICy 的输入映射。支持通过软件或硬件触发输入捕获，其功能框图如下图所示：

图 13-33 输入捕获模式框图



软件触发：设置 GTIMx_EGR.CCyG 为 1 时，立即软件触发一次捕获，当前计数器 CNT 的值被锁存到对应通道的捕获 / 比较寄存器 CCRy 中，完成一次捕获，捕获完成后 CCyG 位被硬件自动清零。

硬件触发：当检测到输入通道 TIy 上的有效边沿后，当前计数器 CNT 的值被锁存到对应通道的捕获 / 比较寄存器 CCRy 中，完成一次捕获。触发捕获的有效边沿通过捕获 / 比较使能寄存器 GTIMx_CCER 的 CCyNP 和 CCyP 位域来配置，如下表所示：

表 13-7 输入捕获模式配置

GTIMx_CCER 寄存器		捕获触发条件
CCyNP 位域	CCyP 位域	
0	0	上升沿捕获
0	1	下降沿捕获
1	1	上升沿和下降沿均捕获

当发生一次捕获时，对应通道的捕获 / 比较中断标志 GTIMx_ISR.CCyIF 被硬件置 1，同时：

- 如果使能了中断（设置 GTIMx_IER.CCyIE 为 1），将产生中断请求。
- 如果发生捕获事件时，GTIMx_ISR.CCyIF 标志位已经为高，那么重复捕获标志位 GTIMx_ISR.CCyOF 将被硬件置 1。
- 设置 GTIMx_ICR.CCyIF 为 0 或读取捕获寄存器 CCRy 可清除 GTIMx_ISR.CCyIF 标志位，设置 GTIMx_ICR.CCyOF 为 0 可清除 GTIMx_ISR.CCyOF 标志位。

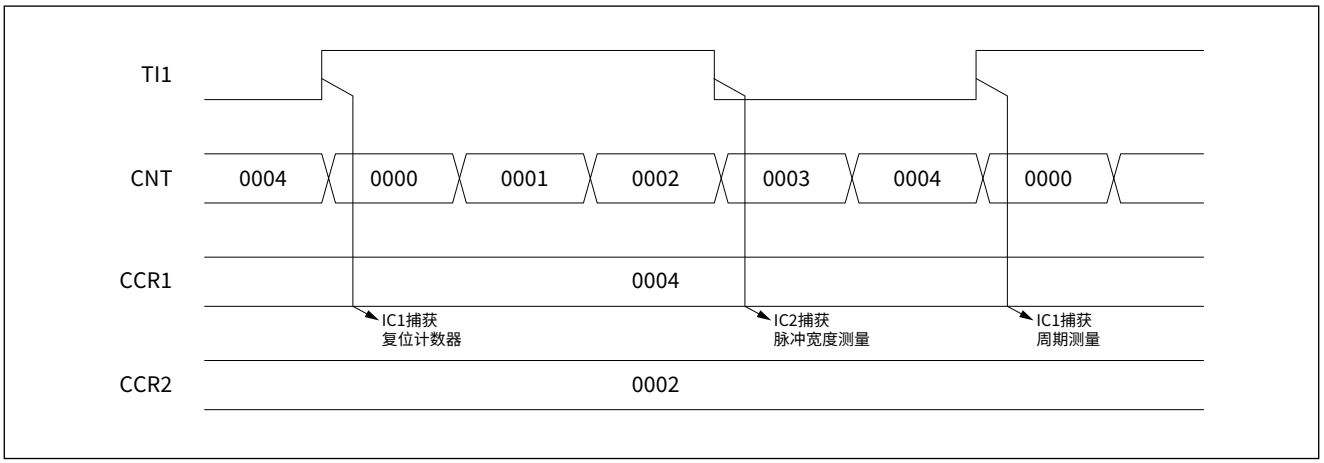
13.3.3.1 PWM 输入模式

PWM 输入模式是输入捕获模式的一个应用。输入捕获功能支持同一通道输入信号被两路捕获寄存器进行捕获，结合从模式复位功能，可方便测量 PWM 输入信号的周期和脉宽。

PWM 输入模式只能与 TI1 和 TI2 信号配合使用，因为只有 TI1FP1 和 TI2FP2 与从模式控制器相连。

下图是 PWM 信号从 TI1 通道输入时的时序示例：

图 13-34 PWM 输入模式时序



13.3.3.2 输入捕获来源

GTIM 的输入捕获来源可以是外部 GTIMx_CHy 引脚，也可以是片内其它外设，通过 TI 输入选择寄存器 GTIMx_TISEL 的 TIySEL 位域进行配置。

当 GTIMx_TISEL.TIySEL 为 0x0 时，TIy 通道输入捕获信号来源为外部 GTIMx_CHy 引脚，此时需通过 GPIO 复用功能寄存器 (GPIOx_AFRH 和 GPIOx_AFRL) 将对应引脚配置为复用功能。

当 GTIMx_TISEL.TIySEL 为 0x1~0xF 时，TIy 通道输入捕获信号来自片内其它外设，如下表所示：

表 13-8 通用定时器输入捕获来源

TIySEL 位域值	各通道输入捕获信号来源			
	TI1 通道	TI2 通道	TI3 通道	TI4 通道
0000	GTIMx_CH1 引脚	GTIMx_CH2 引脚	GTIMx_CH3 引脚	GTIMx_CH4 引脚
0001	VC1_OUT	VC1_OUT	VC1_OUT	VC1_OUT
0010	VC2_OUT	VC2_OUT	VC2_OUT	VC2_OUT
0011	UART1_RXD	UART2_RXD	UART3_RXD	-
0100	UART1_TXD	UART2_TXD	UART3_TXD	-
0101	MCO_OUT	MCO_OUT	MCO_OUT	MCO_OUT
0110	HSE_FAULT	HSE_FAULT	HSE_FAULT	HSE_FAULT
0111	LSE_FAULT	LSE_FAULT	LSE_FAULT	LSE_FAULT
1000	RTC_OUT	RTC_OUT	RTC_OUT	RTC_OUT
1001	LSI_OUT	LSI_OUT	LSI_OUT	LSI_OUT
1010	BTIM1_Trgo	BTIM1_Trgo	BTIM1_Trgo	BTIM1_Trgo
1011	BTIM2_Trgo	BTIM2_Trgo	BTIM2_Trgo	BTIM2_Trgo
1100	BTIM3_Trgo	BTIM3_Trgo	BTIM3_Trgo	BTIM3_Trgo
1101	GTIM1_Trgo	GTIM1_Trgo	GTIM1_Trgo	GTIM1_Trgo
1110	GTIM2_Trgo	GTIM2_Trgo	GTIM2_Trgo	GTIM2_Trgo
1111	ATIM_Trgo	ATIM_Trgo	ATIM_Trgo	ATIM_Trgo

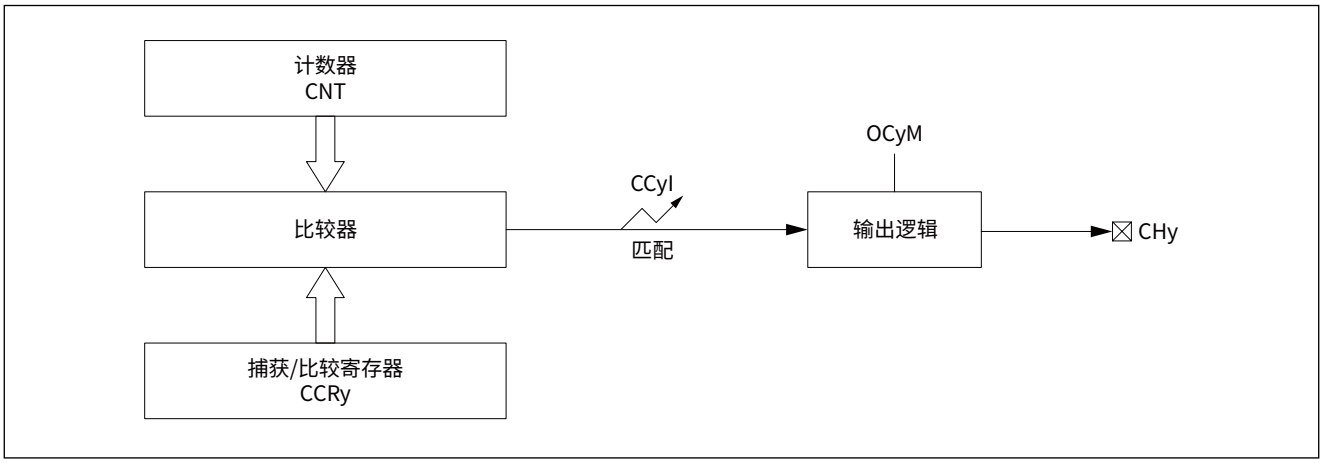
注意：

禁止选择 GTIMx 自己输出的 Trgo 信号。

13.3.4 输出比较功能

GTIM 支持输出比较功能，设置比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 CCyS 位域为 0 时，CCy 通道配置为输出。在输出比较模式下，当前计数器 CNT 的值与对应通道 CHy 的捕获 / 比较寄存器 CCRy 的值相比较，当两者匹配时，比较通道 CHy 输出为可设定的电平状态，同时产生比较中断。其功能框图如下图所示:

图 13-35 输出比较模式框图



CHy 引脚的输出动作由比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyM 位域设置，如下表所示：

表 13-9 输出比较模式配置

OCyM 位域值	比较模式配置
0000	比较匹配时 OCyREF 保持原电平
0001	比较匹配时 OCyREF 置 1
0010	比较匹配时 OCyREF 置 0
0011	比较匹配时 OCyREF 翻转
0100	强制 OCyREF 为低电平
0101	强制 OCyREF 为高电平
0110	PWM 模式 1
0111	PWM 模式 2
1000	可再触发 OPM 模式 1
1001	可再触发 OPM 模式 2
1011	计数方向输出
1100	组合 PWM 模式 1
1101	组合 PWM 模式 2
1110	不对称 PWM 模式 1
1111	不对称 PWM 模式 2

当发生比较匹配时，对应通道的捕获 / 比较中断标志 GTIMx_ISR.CCyIF 被硬件置 1，同时：

- 如果使能了中断（设置 GTIMx_IER.CCyIE 为 1），将产生中断请求。
- 设置 GTIMx_ICR.CCyIF 为 0 可清除 GTIMx_ISR.CCyIF 标志位。

捕获 / 比较寄存器 GTIMx_CCRy 具有缓存功能，通过 OCyPE 位域选择是否使用缓存功能。当 OCyPE 为 0 时，禁止通道 CHy 的比较缓存功能，可在任意时候通过软件更新 GTIMx_CCRy 寄存器，更新值立即生效并影响输出波形；当 OCyPE 为 1 时，使能通道 CHy 的比较缓存功能，更新 GTIMx_CCRy 寄存器不会立即生效，仅当发生更新事件 UEV 时才会将 GTIMx_CCRy 寄存器的值更新到有效寄存器。

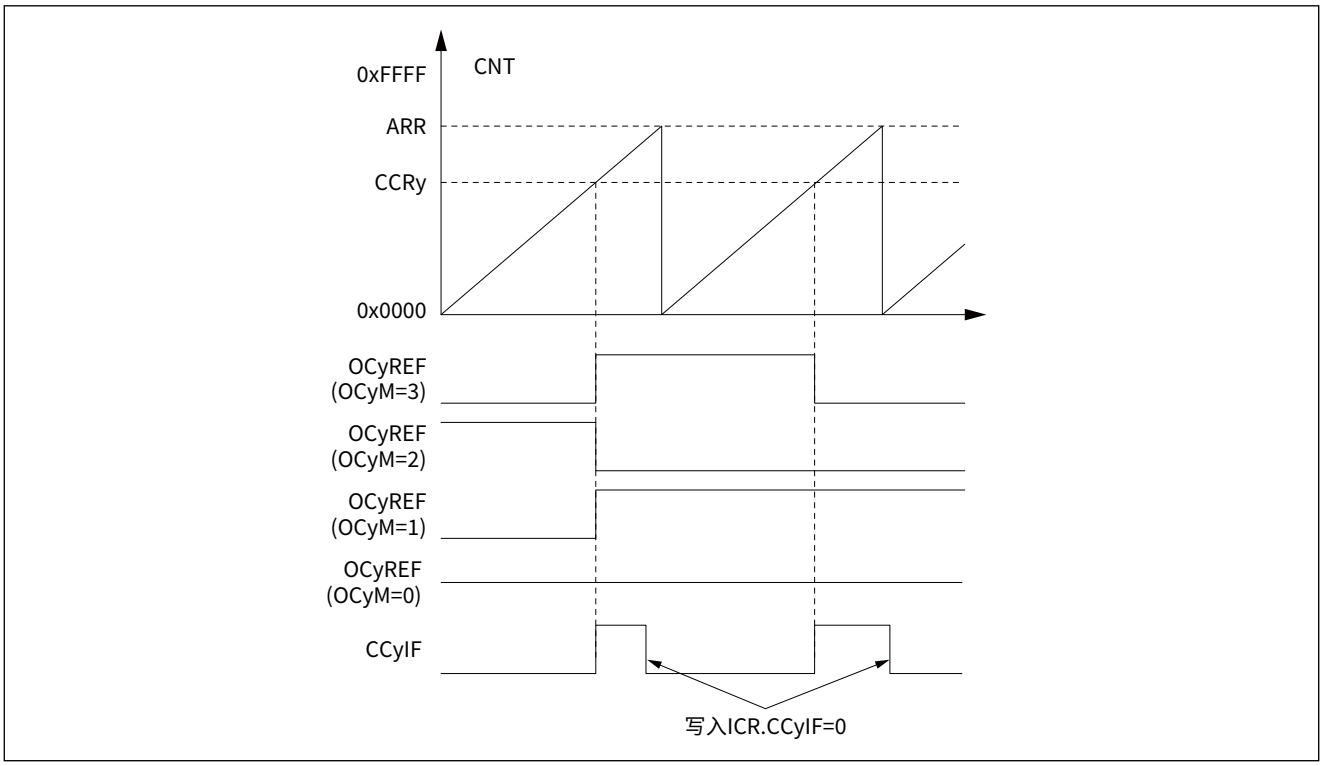
13.3.4.1 匹配输出

匹配输出模式用于控制输出波形，或指示已经过某一时间段：

- 设置 OCyM 位域为 0x0，比较匹配时，参考信号 OCyREF 保持其电平，同时 GTIMx_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x1，比较匹配时，参考信号 OCyREF 设置为高电平，同时 GTIMx_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x2，比较匹配时，参考信号 OCyREF 设置为低电平，同时 GTIMx_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x3，比较匹配时，参考信号 OCyREF 发生翻转，同时 GTIMx_ISR.CCyIF 标志位置 1。

下图是各比较匹配输出模式时序示例，示例为边沿对齐递增模式：

图 13-36 比较匹配输出模式



13.3.4.2 强制输出

在强制输出模式下，输出比较信号能够直接由软件强置为高或低状态，而不依赖于捕获 / 比较寄存器 GTIMx_CCRy 和计数寄存器 GTIMx_CNT 的比较结果。

设置比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyM 位域为 0x4，即可将参考信号 OCyREF 强制变为低电平；设置 OCyM 位域为 0x5，即可将参考信号 OCyREF 强制变为高电平。

强制输出模式下，GTIMx_CCRy 寄存器和计数器 GTIMx_CNT 之间的比较仍然在进行，相应的标志也会置 1，也会产生相应的中断请求。

13.3.4.3 PWM 输出

脉冲宽度调制（PWM）模式可以产生一个由重载寄存器 GTIMx_ARR 确定频率、由捕获 / 比较寄存器 GTIMx_CCRy 确定占空比的信号。

向比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyM 位域写入 0x6（PWM 模式 1）或 0x7（PWM 模式 2），能够独立地设置每个 CHy 输出通道产生一路 PWM，输出状态如下表所示：

表 13-10 PWM 模式 1/2 输出状态

工作模式	计数方向	PWM 模式 1	PWM 模式 2
边沿对齐模式 中央对齐模式	向上	CNT < CCRy 时，OCyREF 为高	CNT < CCRy 时，OCyREF 为低
	向下	CNT > CCRy 时，OCyREF 为低	CNT > CCRy 时，OCyREF 为高

以下是不同计数模式下各 PWM 模式的波形实例，其中 GTIMx_ARR 为 0x08：

图 13-37 PWM 模式 1，边沿对齐模式，递增计数

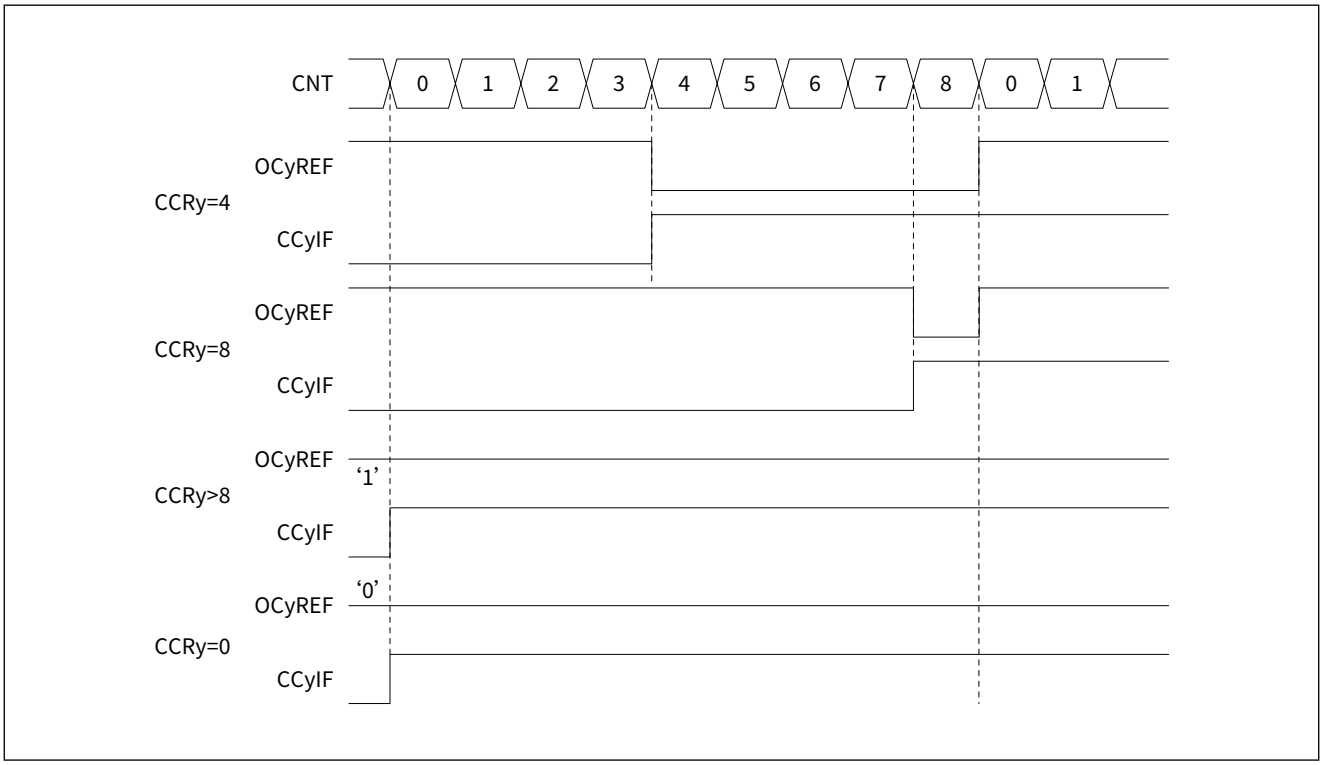


图 13-38　PWM 模式 1，边沿对齐模式，递减计数

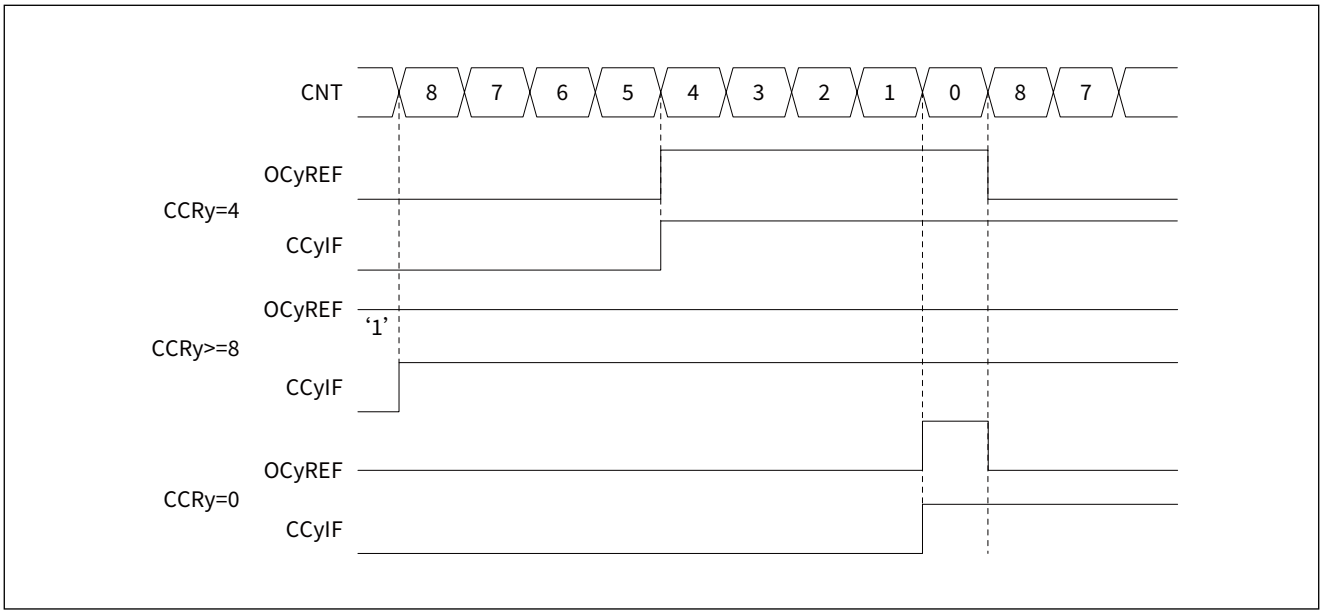
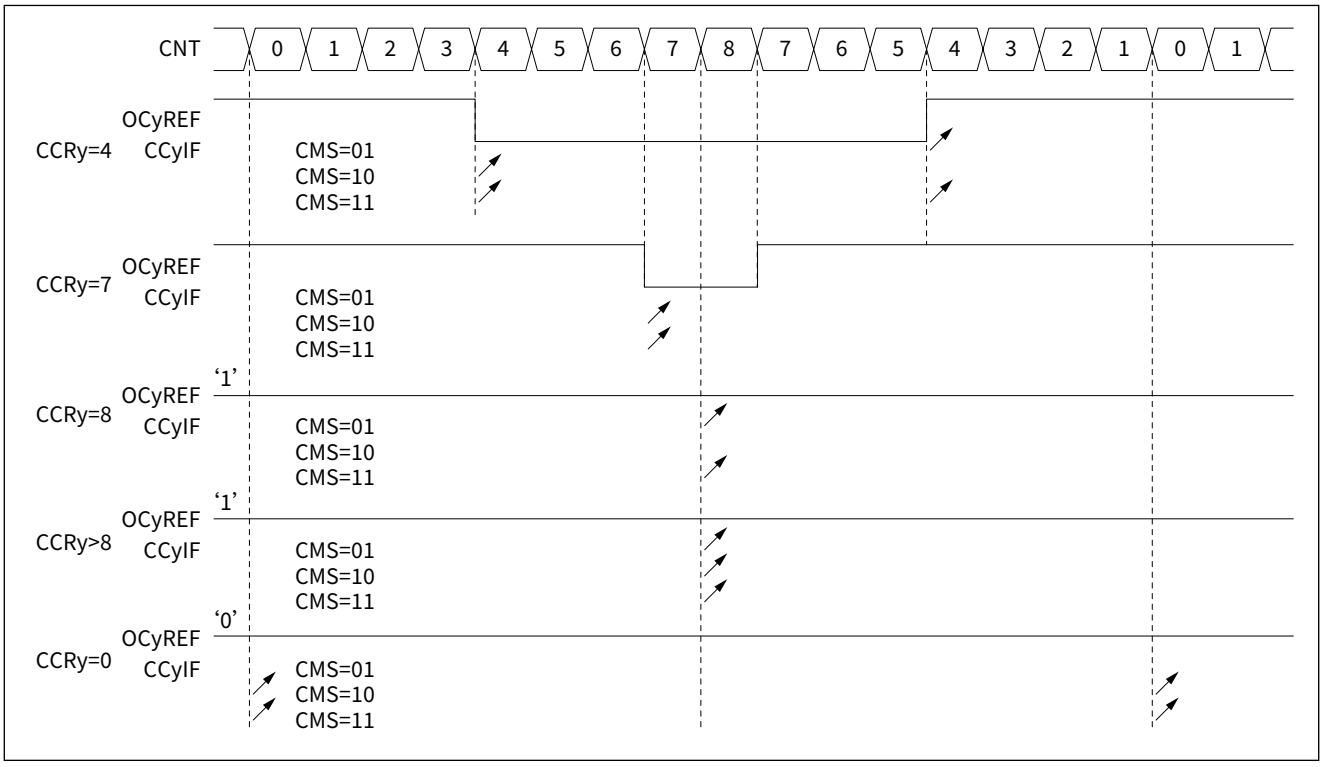


图 13-39　PWM 模式 1，中心对齐模式



13.3.4.4 不对称 PWM 输出

不对称 PWM 输出是相对于中心对齐模式计数时的对称 PWM 输出而言的。在不对称 PWM 输出模式下，允许在中心对齐模式下生成两个具有可编程相移的 PWM 信号，频率由重载寄存器 GTIMx_ARR 确定，占空比和相移由两个捕获 / 比较寄存器 GTIMx_CCRy 确定，其中一个 CCRy 寄存器控制递增计数时的 PWM，另一个 CCRy 寄存器控制递减计数时的 PWM，具体如下所示：

- OC1REFC 由 CCR1(递增计数) 与 CCR2(递减计数) 控制
- OC2REFC 由 CCR2(递增计数) 与 CCR1(递减计数) 控制
- OC3REFC 由 CCR3(递增计数) 与 CCR4(递减计数) 控制
- OC4REFC 由 CCR4(递增计数) 与 CCR3(递减计数) 控制

向比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyM 位域写入 0xE（不对称 PWM 模式 1）或 0xF（不对称 PWM 模式 2），能够独立地设置每个 CHy 输出通道产生不对称 PWM。

当设定通道用作不对称 PWM 输出通道时，另一辅助通道仍可使用。例如，通道 CH1 配置为不对称 PWM 模式 1 产生 OC1REFC 信号时，通道 CH2 仍可以配置为 PWM 模式 2 输出 OC2REF 信号，或者配置为不对称 PWM 模式 2 输出 OC2REFC 信号。

以下是不对称 PWM 模式 1/2 的波形实例，其中 ARR=8、CCR1=0、CCR2=8、CCR3=3、CCR4=5：

图 13-40　不对称 PWM 模式 1 示例

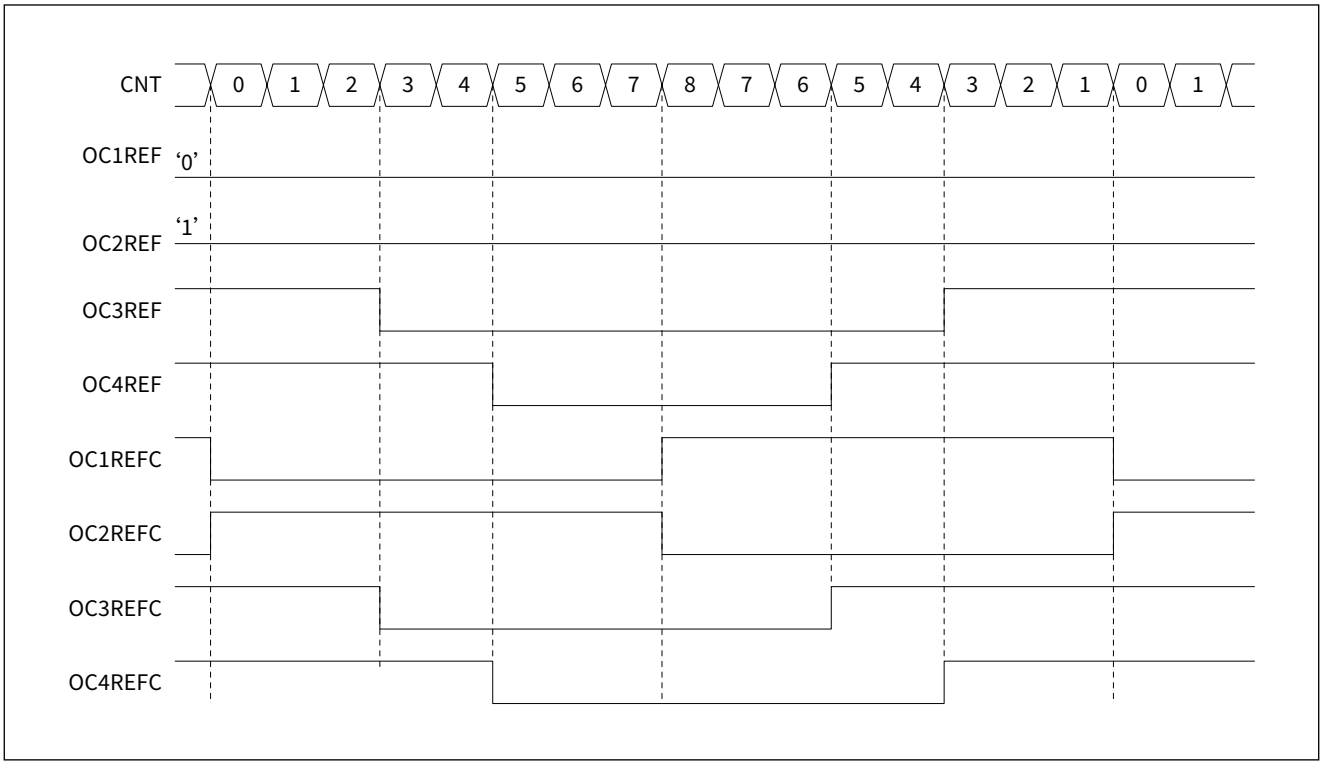
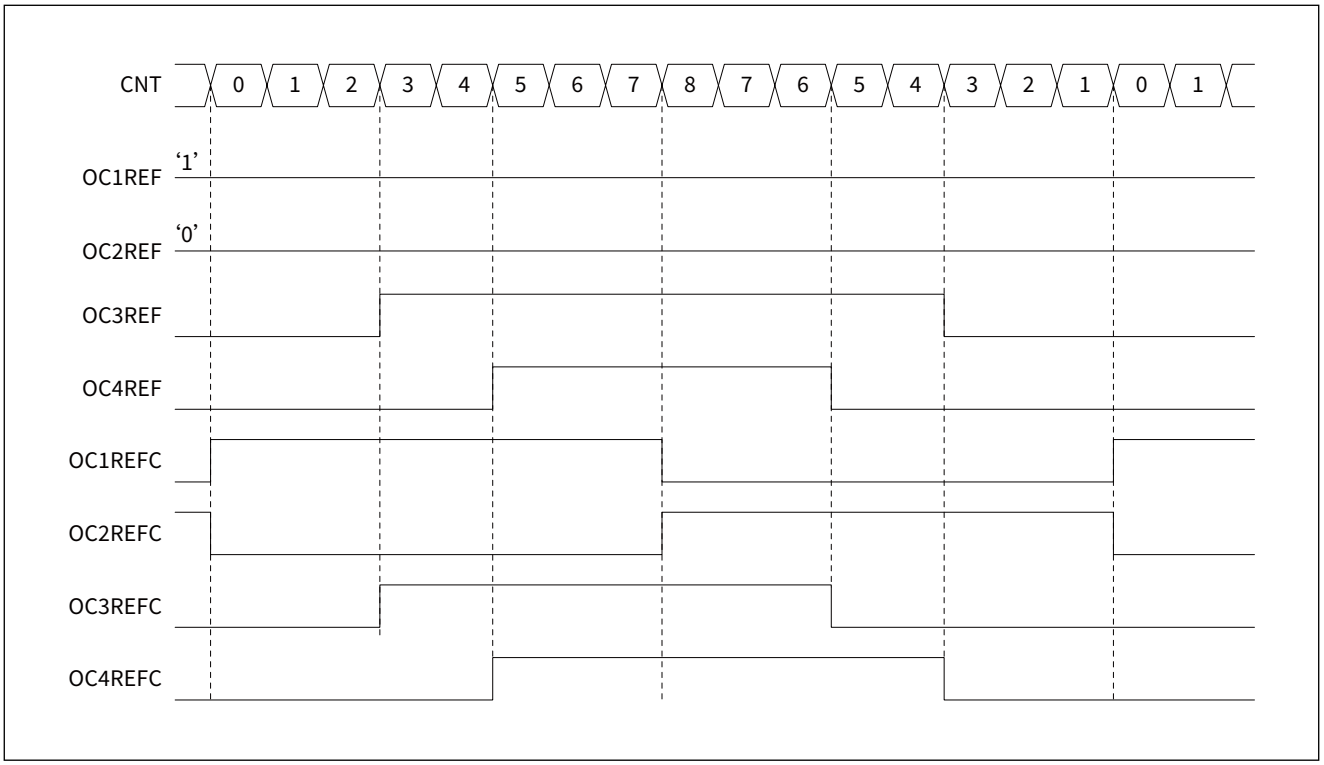


图 13-41　不对称 PWM 模式 2 示例



13.3.4.5 组合 PWM 输出

在组合 PWM 输出模式下，允许在边沿对齐或中心对齐模式下生成两个具有可编程延时和相移的 PWM 脉冲，频率由重载寄存器 GTIMx_ARR 确定，占空比和延时由两个捕获 / 比较寄存器 GTIMx_CCRy 确定。

该模式下，OCyREFC 信号由两个参考信号 OCyREF 的逻辑或运算或者逻辑与运算组合生成，通过比较模式寄存器（GTIMx_CCMR1CMP 和 GTIMx_CCMR2CMP）的 OCyM 位域可独立选择组合 PWM 模式，具体如下表所示：

表 13-11 组合 PWM 模式配置

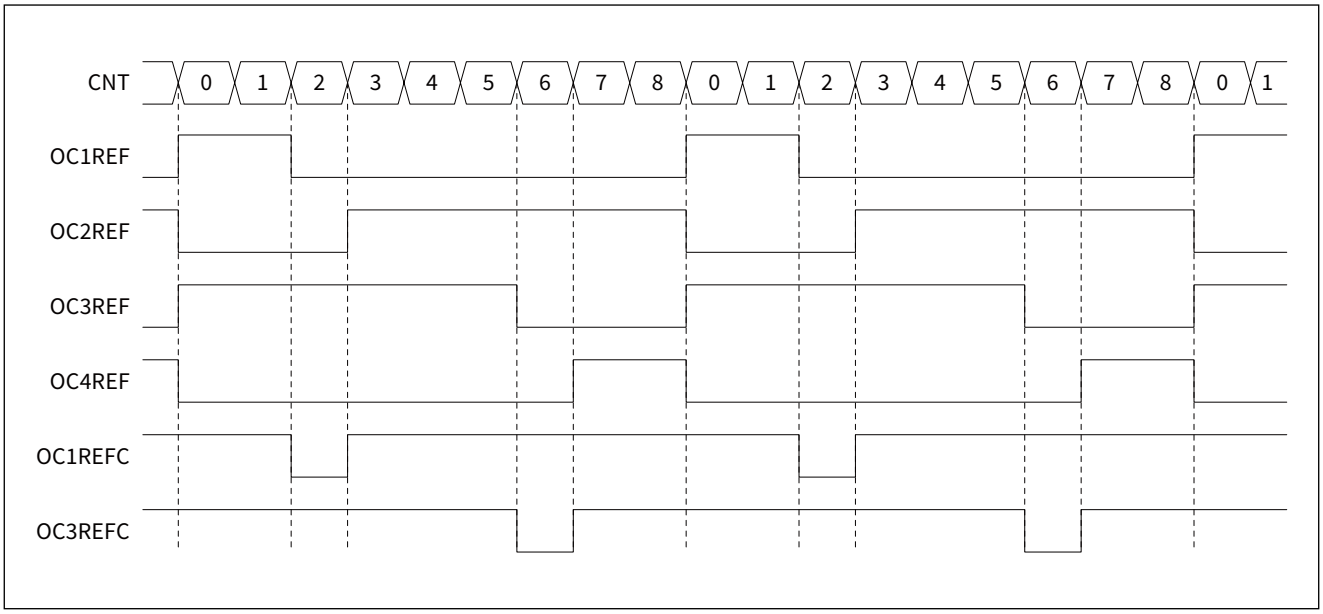
OCyM 位域值	输出状态
0xC （组合 PWM 模式 1）	OC1REFC（或 OC2REFC）由 OC1REF 和 OC2REF 逻辑或运算生成 OC3REFC（或 OC4REFC）由 OC3REF 和 OC4REF 逻辑或运算生成
0xD （组合 PWM 模式 2）	OC1REFC（或 OC2REFC）由 OC1REF 和 OC2REF 逻辑与运算生成 OC3REFC（或 OC4REFC）由 OC3REF 和 OC4REF 逻辑与运算生成

当设定通道用作组合 PWM 输出通道时，另一辅助通道仍可使用，但必须设置为相反的 PWM 模式。例如，通道 CH1 配置为组合 PWM 模式 1 产生 OC1REFC 信号时，通道 CH2 仍可使用，CH2 可以配置为 PWM 模式 2 输出 OC2REF 信号。

下图显示了边沿对齐递增模式下，组合 PWM 模式 1 的信号示例，具体配置如下：

- 重载值 ARR 为 8。
- 通道 1 配置为组合 PWM 模式 1，CCR1=2。
- 通道 2 配置为 PWM 模式 2，CCR2=3。
- 通道 3 配置为组合 PWM 模式 1，CCR3=6。
- 通道 4 配置为 PWM 模式 2，CCR4=7。

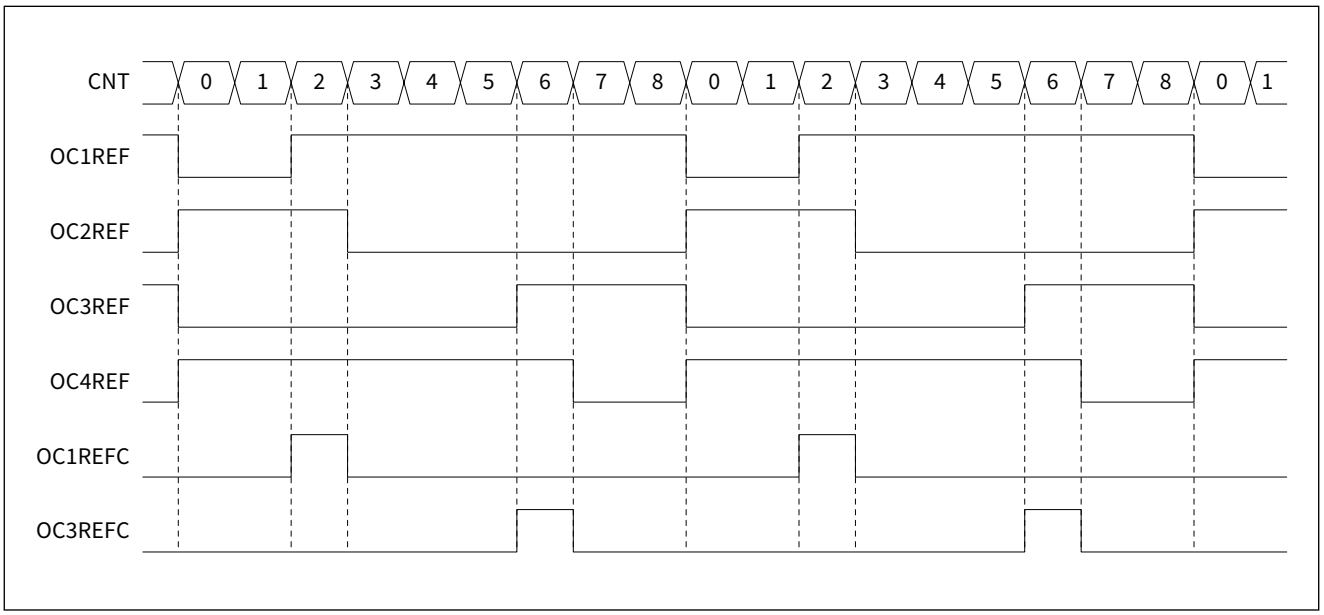
图 13-42 组合 PWM 模式 1 示例



下图显示了边沿对齐递增模式下，组合 PWM 模式 2 的信号示例，具体配置如下：

- 重载值 ARR 为 8。
- 通道 1 配置为组合 PWM 模式 2，CCR1=2。
- 通道 2 配置为 PWM 模式 1，CCR2=3。
- 通道 3 配置为组合 PWM 模式 2，CCR3=6。
- 通道 4 配置为 PWM 模式 1，CCR4=7。

图 13-43　组合 PWM 模式 2 示例



13.3.4.6 可再触发单脉冲模式

可再触发单脉冲模式配合组合复位 + 触发模式（SMS=8），允许在触发信号（TRGI）的触发下复位并启动计数器，并产生长度可编程的脉冲，脉冲宽度由 ARR 确定。

与单脉冲模式相比，具有如下区别：

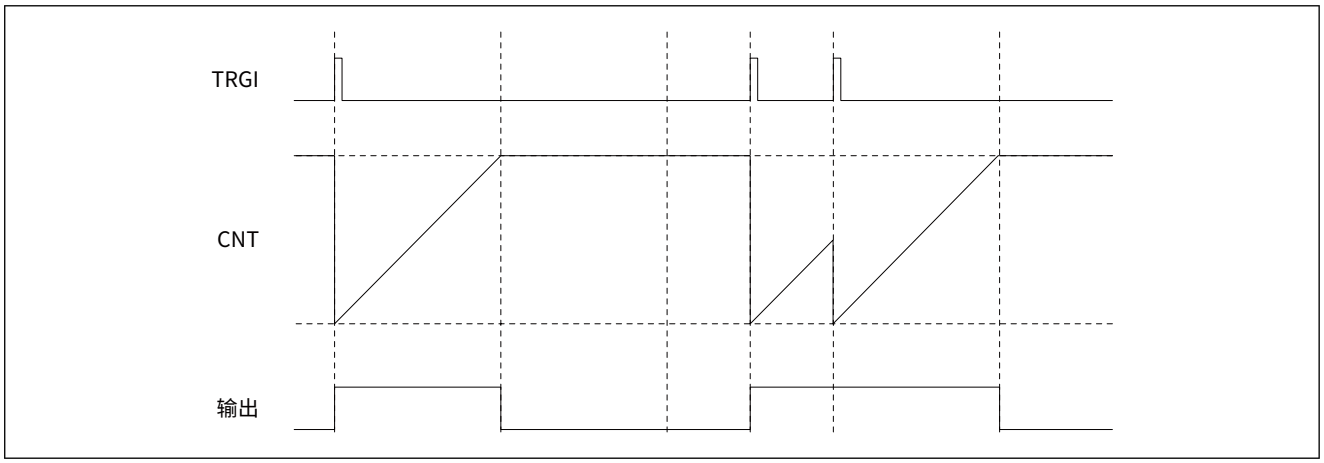
表 13-12 可再触发单脉冲模式和单脉冲模式

单脉冲模式	可再触发单脉冲模式
发生触发时，经一段可编程的延时后产生一个脉宽可编程的单脉冲	发生触发时，脉冲立即产生，无可编程延时
新的触发无效	如果上一个触发产生的脉冲未完成，又发生新的触发，脉冲将延长

此模式下，计数器只能设置为边沿对齐模式，不能设置为中心对齐模式。当配置为递增计数模式时，对应 CCRy 必须设置为 0；配置为递减计数模式时，对应 CCRy 必须大于或等于 ARR。

下图所示为可再触发单脉冲模式 2 的示例：

图 13-44 可再触发单脉冲模式 2 示例



13.3.4.7 计数方向输出

将 GTIMx_CCMRxCMP 寄存器的 OCyM 位域设置为 0xB，可在对应的 CHy 通道上输出计数方向信号（GTIMx_CR1 寄存器中 DIR 位的拷贝）。

此功能可用于在编码器模式下监控计数方向（或旋转方向），或在中心对齐 PWM 模式下指示向上 / 向下相位的信号。

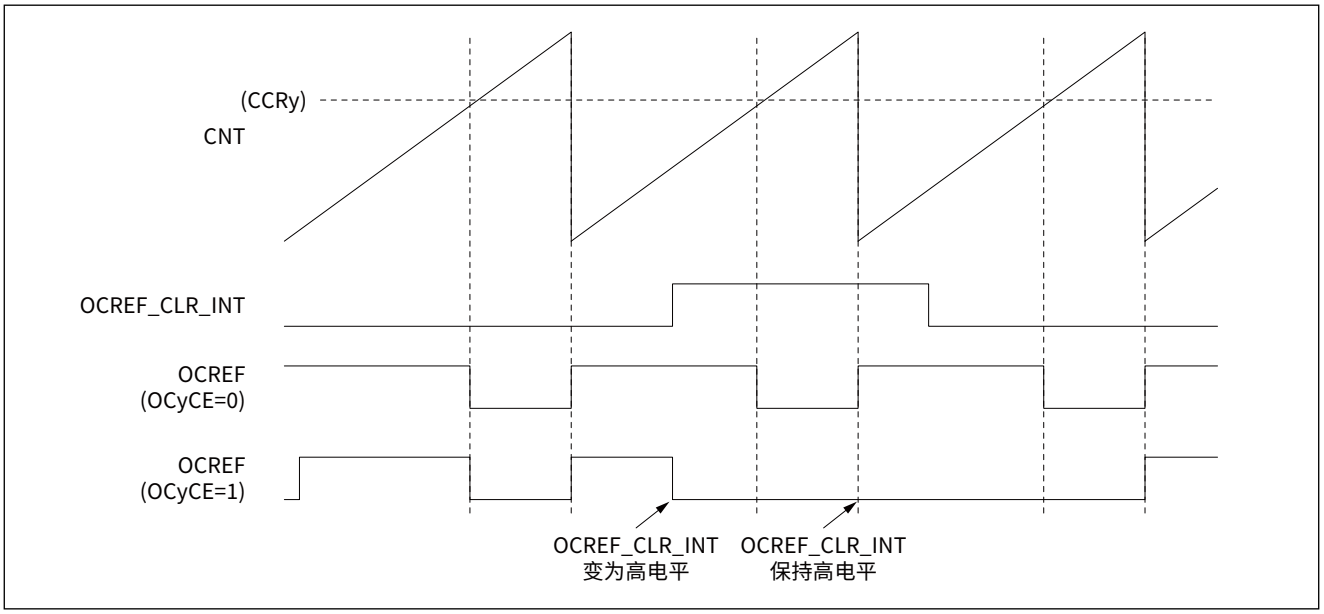
13.3.4.8 参考信号清零功能

设置 GTIMx_CCMRxCMP 寄存器的 OCyCE 位域为 1，可打开对应通道的 OCyREF 信号清零功能，即当 OCREF_CLR_INT 输入上检测到高电平时，可将 OCyREF 信号清零，OCyREF 信号将保持低电平，直到发生下一更新事件 UEV。该功能只在输出比较和 PWM 模式下可用，强制输出模式下不可用。

清零源 OCREF_CLR_INT 可选 OCREF_CLR 或 ETRF，通过从模式控制寄存器 GTIMx_SMCR 的 OCCS 位域进行选择，OCREF_CLR 也有多种输入源，参见 13.9.23 GTIMx_AF2 复用功能选项寄存器 2 的 OCRSEL 位域说明。

下图所示为 OCREF_CLR_INT 输入变为高电平时 OCyREF 信号状态：

图 13-45 OCyREF 清零示例



13.3.5 UIF 位重映射

设置 GTIMx_CR1 寄存器中的 UIFREMAP 位为 1 可使能 UIF 状态位重映射功能，可强制将更新中断标志 GTIMx_ISR.UIF 连续复制到 GTIMx_CNT 寄存器的 UIFCPY 位域中。这样便可自动读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化角速度的计算。

UIF 和 UIFCPY 标志使能之间没有延迟。

当 GTIMx_CR1.UIFREMAP 为 0 时，UIFCPY 位域保留，读为 0。

13.3.6 定时器级联 ITR

通过 ITR 信号可以实现定时器级联，即将主定时器的触发输出（TRGO）连接到从定时器的 ITR 输入，进而连接到从定时器的触发输入（TRGI）。ITR 级联可实现多种功能，如将一个定时器用作另一个定时器的预分频器、主定时器对从定时器的计数器执行复位、使能或提供时钟等功能。

GTIM 的级联输入 ITR 的来源为 BTIM、GTIM 和 ATIM 的触发输出信号（TRGO）以及 UART 的 TXD/RXD 信号，可通过从模式控制寄存器 GTIMx_SMCR 的 TS 位域来选择，具体配置如下表所示：

表 13-13 ITR 信号来源

GTIMx_SMCR.TS 位域值	ITR 信号名称	ITR 信号来源
00000	ITR0	ATIM_Trgo
00001	ITR1	-
00010	ITR2	GTIM1_Trgo
00011	ITR3	GTIM2_Trgo
01000	ITR4	UART1_TXD
01001	ITR5	UART2_TXD
01010	ITR6	UART3_TXD
01011	ITR7	UART1_RXD
01100	ITR8	UART2_RXD
01101	ITR9	UART3_RXD
01110	ITR10	BTIM1_Trgo
01111	ITR11	BTIM2_Trgo
10000	ITR12	BTIM3_Trgo

注意：

禁止选择 GTIMx 自己输出的 Trgo 信号。

通过控制寄存器 GTIMx_CR2 的 MMS 位域，可以选择 GTIM 主模式下将要发送到从定时器以实现同步的信息（TRGO），连接到其他定时器的 ITR 输入。如下表所示：

表 13-14　GTIM 触发输出（TRGO）

GTIMx_CR2.MMS	TRGO 信号
0000	复位
0001	计数器使能信号 CNT_EN
0010	更新事件
0011	通道 1 比较捕获脉冲
0100	通道 2 比较捕获脉冲
0101	通道 3 比较捕获脉冲
0110	通道 4 比较捕获脉冲
0111	OC1REF 信号
1000	OC2REF 信号
1001	OC3REF 信号
1010	OC4REF 信号
1011	OC1REFC 信号
1100	OC2REFC 信号
1101	OC3REFC 信号
1110	OC4REFC 信号
1111	编码器时钟输出

13.3.7 片内外设互联 ETR

GTIM 的 ETR 信号来源可以是外部 GTIMx_ETR 引脚，也可以是片内其它外设，通过复用功能选项寄存器 GTIMx_AF1 的 ETRSEL 位域进行选择。

当 ETR 信号来自 GTIMx_ETR 引脚输入时，具体外部输入端口可参见表 8-2 GPIO 复用功能分配表，并需通过 GPIO 复用功能寄存器 (GPIOx_AFRH 和 GPIOx_AFRL) 进行复用配置；当 ETR 信号来自片内其它外设时，可实现片内外设互联。

ETR 信号来源如下表所示：

表 13-15 ETR 信号来源

AF1.ETRSEL 位域值	ETR 信号来源	
	GTIM1	GTIM2
0000	GTIM1_ETR 引脚	GTIM2_ETR 引脚
0001	VC1_OUT	VC1_OUT
0010	VC2_OUT	VC2_OUT
0101	ADC_AWD	ADC_AWD
1000	LVD_OUT	LVD_OUT
1001	-	GTIM1_ETR 引脚
1010	GTIM2_ETR 引脚	-
1011	UART1_TXD	UART1_TXD
1100	UART2_TXD	UART2_TXD
1101	UART3_TXD	UART3_TXD
1110	HSE_FAULT	HSE_FAULT
1111	LSE_FAULT	LSE_FAULT

13.4 GTIM 中断

GTIM 支持 10 个中断源，当 GTIM 中断事件发生时，中断标志位会被硬件置位，如果设置了对应的中断使能控制位，将产生中断请求。

在用户 GTIM 中断服务程序中，应查询相关 GTIM 中断标志位，以进行相应的处理，在退出中断服务程序之前，要清除该中断标志位，以避免重复进入中断服务程序。

各 GTIM 中断源的标志位、中断使能位、中断标志清除位或清除方法，如下表所示：

表 13-16 GTIM 中断控制

中断事件	中断标志位	中断使能位	中断标志清除
更新中断	ISR.UIF	IER.UIE	写 0 到 ICR.UIF
捕获 / 比较 1 中断	ISR.CC1IF	IER.CC1IE	写 0 到 ICR.CC1IF 或读 CCR1 寄存器
捕获 / 比较 2 中断	ISR.CC2IF	IER.CC2IE	写 0 到 ICR.CC2IF 或读 CCR2 寄存器
捕获 / 比较 3 中断	ISR.CC3IF	IER.CC3IE	写 0 到 ICR.CC3IF 或读 CCR3 寄存器
捕获 / 比较 4 中断	ISR.CC4IF	IER.CC4IE	写 0 到 ICR.CC4IF 或读 CCR4 寄存器
触发中断	ISR.TIF	IER.TIE	写 0 到 ICR.TIF
编码器索引中断	ISR.IDXF	IER.IDXIE	写 0 到 ICR.IDXF
编码器方向改变中断	ISR.DIRF	IER.DIRIE	写 0 到 ICR.DIRF
编码器索引错误中断	ISR.IERRF	IER.IERRIE	写 0 到 ICR.IERRF
编码器转换错误中断	ISR.TERRF	IER.TERRIE	写 0 到 ICR.TERRF

13.5 触发 ADC

定时器支持多种内部信号触发启动 ADC，如捕获比较信号和触发输出信号 TRGO，触发输出 TRGO 有多种可能的事件，具体由控制寄存器 GTIMx_CR2 的 MMS 位域进行选择。同时，ADC 外设需配置其外部触发启动寄存器，以选择对应触发源。

应注意，必须先使能 ADC 时钟，才能从主定时器接收事件，且从定时器接收触发信号时，不得实时更改 ADC 时钟。

13.6 调试支持

GTIM 支持在调试模式下停止或继续计数，通过调试状态定时器控制寄存器 SYSCTRL_DEBUG 的 GTIMx 位域来设置:

- 设置 SYSCTRL_DEBUG.GTIMx 为 1，则在调试状态时暂停 GTIMx 的计数器计数。
- 设置 SYSCTRL_DEBUG.GTIMx 为 0，则在调试状态时 GTIMx 的计数器继续计数。

13.7 编程示例

13.7.1 外部时钟模式 1 编程示例

以下示例中，配置递增计数器在 GTIMx_CH1 通道的上升沿进行计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO) 章节；

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择上升沿计数有效；

步骤 6：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 GTIMx_SMCR.SMS 为 0x7，使 GTIM 工作于外部时钟模式 1；

步骤 8：设置 GTIMx_IER.UIE 为 1 并配置对应 NVIC，使能更新中断；

步骤 9：配置计数器预分频器 GTIMx_PSC；

步骤 10：设置期望的重载值 GTIMx_ARR；

步骤 11：设置 GTIMx_CR1.CEN 为 1，使能计数器；

步骤 12：当计数器溢出时，GTIMx_ISR.UIF 标志位置 1，进入中断服务程序，设置 GTIMx_ICR.UIF 为 0 清除该中断标志。

13.7.2 外部时钟模式 2 编程示例

以下是外部时钟模式 2+ 触发模式的编程示例，GTIMx_CH1 输入出现上升沿时触发启动计数器，计数器在 GTIMx_ETR 输入信号的每个上升沿计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_ETR、GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_ETR、GTIMx_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口 (GPIO) 章节；

步骤 3：设置 GTIMx_AF1.ETRSEL 为 0，选择 ETR 来源为 GTIMx_ETR 引脚；

步骤 4：设置 GTIMx_SMCR.ETP 为 0，选择 ETR 上升沿有效；

步骤 5：设置 GTIMx_SMCR.ETPS 为 0，关闭 ETR 的预分频器；

步骤 6：设置 GTIMx_SMCR.ETF 为 0，关闭 ETR 输入滤波；

步骤 7：设置 GTIMx_SMCR.ECE 为 1，使能外部时钟模式 2；

步骤 8：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 9：设置 GTIMx_CCMR1CAP.IC1F 为 0，关闭 TI1 输入滤波；

步骤 10：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测 TI1 上升沿；

步骤 11：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 12：设置 GTIMx_SMCR.SMS 为 0x6，使 GTIM 工作于触发模式；

步骤 13：当 GTIMx_CH1 输入出现上升沿时使能，计数器在 GTIMx_ETR 输入信号的每个上升沿计数。

13.7.3 复位模式编程示例

以下示例中，GTIMx_CH1 输入出现上升沿时重新初始化计数器，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO) 章节；

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测上升沿；

步骤 6：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 GTIMx_SMCR.SMS 为 0x4，使 GTIM 工作于复位模式；

步骤 8：设置期望的重载值 GTIMx_ARR；

步骤 9：设置 GTIMx_CR1.CEN 为 1，使能计数器，开始正常计数；

步骤 10：当 GTIMx_CH1 输入出现上升沿时，计数器清零，重新从 0 开始计数。

13.7.4 门控模式编程示例

以下示例中，GTIMx_CH1 引脚输入的信号作为门控信号，控制计数器计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO) 章节；

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测高电平；

步骤 6：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 GTIMx_SMCR.SMS 为 0x5，使 GTIM 工作于门控模式；

步骤 8：设置期望的重载值 GTIMx_ARR；

步骤 9：设置 GTIMx_CR1.CEN 为 1，使能计数器；

步骤 10：当 GTIMx_CH1 输入为高电平时，计数器开始计数；当 GTIMx_CH1 输入为低电平时，计数器停止计数。

13.7.5 触发模式编程示例

以下示例中，GTIMx_CH1 输入出现上升沿时触发启动计数器，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO) 章节；

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测上升沿；

步骤 6：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 GTIMx_SMCR.SMS 为 0x6，使 GTIM 工作于触发模式；

步骤 8：设置期望的重载值 GTIMx_ARR；

步骤 9：当 GTIMx_CH1 输入出现上升沿时，计数器启动计数，同时 TIF 标志置 1。

13.7.6 正交编码器模式编程示例

以下示例中，配置 GTIM 工作在正交编码器模式 -x4 模式，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1、GTIMx_CH2 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 和 GTIMx_CH2 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口 (GPIO) 章节；

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_TISEL.TI2SEL 为 0，选择 TI2 来源为 GTIMx_CH2 通道；

步骤 5：设置 GTIMx_CCMR1CAP.IC1F 和 GTIMx_CCMR1CAP.IC2F，配置输入滤波带宽；

步骤 6：设置 GTIMx_CCER.CC1NP、GTIMx_CCER.CC1P 为 0，CH1 输入不反相；

步骤 7：设置 GTIMx_CCER.CC2NP、GTIMx_CCER.CC2P 为 0，CH2 输入不反相；

步骤 8：设置 GTIMx_SMCR.SMS 为 0x3，使 GTIM 工作于正交编码器模式 -x4 模式；

步骤 9：设置合适的重载值 GTIMx_ARR；

步骤 10：设置 GTIMx_CR1.CEN 为 1，使能计数器。

13.7.7 输入捕获编程示例

13.7.7.1 基本输入捕获模式

以下示例中，GTIMx_CH1 输入出现上升沿时执行一次输入捕获，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1引脚对应的 GPIO 配置成复用输入模式,具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节;

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置 TI1 输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测 IC1 上升沿；

步骤 6：设置 GTIMx_CCMR1CAP.CC1S 为 1，CC1 通道配置为输入，IC1 映射到 TI1 上；

步骤 7：设置 GTIMx_CCMR1CAP.IC1PSC 为 0，关闭 IC1 预分频器；

步骤 8：设置 GTIMx_CCER.CC1E 为 1，使能输入捕获 1 模式；

步骤 9：设置 GTIMx_IER.CC1IE 为 1 并配置对应 NVIC，使能输入捕获 1 中断；

步骤 10：配置计数器预分频器 GTIMx_PSC；

步骤 11：设置期望的重载值 GTIMx_ARR；

步骤 12：设置 GTIMx_CR1.CEN 为 1，使能计数器，开始正常计数；

步骤 13：当发生捕获时，当前计数器 CNT 的值被锁存到捕获 / 比较寄存器 CCR1 中，完成一次捕获，同时 GTIMx_ISR.CC1IF 标志位置 1，进入中断服务程序，设置 GTIMx_ICR.CC1IF 为 0 清除该中断标志。

13.7.7.2 PWM 输入模式

以下示例测量从 GTIMx_CH1 引脚输入的 PWM 信号的周期和占空比，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1引脚对应的 GPIO 配置成复用输入模式,具体寄存器配置请参见 8 通用输入输出端口(GPIO)章节;

步骤 3：设置 GTIMx_TISEL.TI1SEL 为 0，选择 TI1 来源为 GTIMx_CH1 通道；

步骤 4：设置 GTIMx_CCMR1CAP.IC1F，配置 TI1 输入滤波带宽；

步骤 5：设置 GTIMx_CCER.CC1NP 为 0、GTIMx_CCER.CC1P 为 0，选择检测 IC1 上升沿；

步骤 6：设置 GTIMx_CCMR1CAP.CC1S 为 1，CC1 通道配置为输入，IC1 映射到 TI1 上；

步骤 7：设置 GTIMx_CCMR1CAP.IC1PSC 为 0，关闭 IC1 预分频器；

步骤 8：设置 GTIMx_CCER.CC1E 为 1，使能输入捕获 1 模式；

步骤 9：设置 GTIMx_IER.CC1IE 为 1 并配置对应 NVIC，使能输入捕获 1 中断；

步骤 10：设置 GTIMx_CCER.CC2NP 为 0、GTIMx_CCER.CC2P 为 1，选择检测 IC2 下降沿；

步骤 11：设置 GTIMx_CCMR1CAP.CC2S 为 2，CC2 通道配置为输入，IC2 映射到 TI1 上；

步骤 12：设置 GTIMx_CCMR1CAP.IC2PSC 为 0，关闭 IC2 预分频器；

步骤 13：设置 GTIMx_CCER.CC2E 为 1，使能输入捕获 2 模式；

步骤 14：设置 GTIMx_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 15：设置 GTIMx_SMCR.SMS 为 0x4，使 GTIM 工作于复位模式；

步骤 16：配置计数器预分频器 GTIMx_PSC；

步骤 17：设置期望的重载值 GTIMx_ARR；

步骤 18：设置 GTIMx_CR1.CEN 为 1，使能计数器，开始正常计数。

13.7.8 输出比较编程示例

以下示例中，通道 CH1 对外输出设定的波形，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.GTIMx 为 1，打开 GTIMx_CH1 引脚对应的 GPIO 时钟和 GTIM 配置时钟及工作时钟；

步骤 2：将 GTIMx_CH1 引脚对应的 GPIO 配置成复用输出模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO) 章节；

步骤 3：设置 GTIMx_CCER.CC1P 为 0，选择输出极性；

步骤 4：设置 GTIMx_CCMR1CMP.CC1S 为 0，CC1 通道配置为输出；

步骤 5：设置 GTIMx_CCMR1CMP.OC1M，选择输出比较 1 模式；

步骤 6：设置 GTIMx_CCER.CC1E 为 1，在相应输出引脚上输出 OC1 信号；

步骤 7：配置计数器预分频器 GTIMx_PSC；

步骤 8：设置期望的重载值 GTIMx_ARR；

步骤 9：设置期望的比较值 GTIMx_CCR1；

步骤 10：设置 GTIMx_IER.CC1IE 为 1 并配置对应 NVIC，使能比较 1 中断；

步骤 11：设置 GTIMx_CR1.CEN 为 1，使能计数器；

步骤 12：当发生比较匹配时，GTIMx_ISR.CC1IF 标志位置 1，进入中断服务程序，设置 GTIMx_ICR.CC1IF 为 0 清除该中断标志。

13.8 寄存器列表

GTIM1 基地址：GTIM1_BASE = 0x4000 1800

GTIM2 基地址：GTIM2_BASE = 0x4000 1C00

表 13-17 GTIM 寄存器列表

寄存器名称	寄存器地址	寄存器描述
GTIMx_CR1	GTIMx_BASE + 0x00	控制寄存器 1
GTIMx_CR2	GTIMx_BASE + 0x04	控制寄存器 2
GTIMx_SMCR	GTIMx_BASE + 0x08	从模式控制寄存器
GTIMx_IER	GTIMx_BASE + 0x0C	中断使能寄存器
GTIMx_ISR	GTIMx_BASE + 0x10	中断标志寄存器
GTIMx_ICR	GTIMx_BASE + 0x70	中断标志清除寄存器
GTIMx_EGR	GTIMx_BASE + 0x14	事件生成寄存器
GTIMx_CCMR1CAP	GTIMx_BASE + 0x18	捕获模式寄存器 1
GTIMx_CCMR1CMP	GTIMx_BASE + 0x18	比较模式寄存器 1
GTIMx_CCMR2CAP	GTIMx_BASE + 0x1C	捕获模式寄存器 2
GTIMx_CCMR2CMP	GTIMx_BASE + 0x1C	比较模式寄存器 2
GTIMx_CCER	GTIMx_BASE + 0x20	捕获 / 比较使能寄存器
GTIMx_CNT	GTIMx_BASE + 0x24	计数寄存器
GTIMx_PSC	GTIMx_BASE + 0x28	预分频寄存器
GTIMx_ARR	GTIMx_BASE + 0x2C	自动重载寄存器
GTIMx_CCR1	GTIMx_BASE + 0x34	捕获 / 比较寄存器 1
GTIMx_CCR2	GTIMx_BASE + 0x38	捕获 / 比较寄存器 2
GTIMx_CCR3	GTIMx_BASE + 0x3C	捕获 / 比较寄存器 3
GTIMx_CCR4	GTIMx_BASE + 0x40	捕获 / 比较寄存器 4
GTIMx_ECR	GTIMx_BASE + 0x58	编码控制寄存器
GTIMx_TISEL	GTIMx_BASE + 0x5C	TI 输入选择寄存器
GTIMx_AF1	GTIMx_BASE + 0x60	复用功能选项寄存器 1
GTIMx_AF2	GTIMx_BASE + 0x64	复用功能选项寄存器 2

13.9 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

13.9.1 GTIMx_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11	UIFREMAP	RW	UIF 状态位重映射 0：禁止重映射，GTIMx_CNT[31] 保持为 0 1：使能重映射，GTIMx_CNT[31] 等效于 GTIMx_ISR.UIF
10	RFU	-	保留位，请保持默认值
9:8	CKD	RW	配置 PCLK 时钟与滤波时钟分频比 00：$f_{DTS} = f_{PCLK}/1$ 01：$f_{DTS} = f_{PCLK}/2$ 10：$f_{DTS} = f_{PCLK}/4$ *注：滤波时钟用于 ETR 滤波和 TI 滤波。*
7	ARPE	RW	自动重载预装载使能 0：禁止 ARR 寄存器缓冲 1：使能 ARR 寄存器缓冲
6:5	CMS	RW	中心对齐模式选择 00：边沿对齐模式。计数器根据方向位 (DIR) 递增计数或递减计数。 01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（GTIMx_CCMRx 寄存器中的 CCyS=00）的输出比较中断标志才置 1。 10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（GTIMx_CCMRx 寄存器中的 CCyS=00）的输出比较中断标志才置 1。 11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（GTIMx_CCMRx 寄存器中的 CCyS=00）的输出比较中断标志都会置 1。 *注 1：只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。* *注 2：中心对齐模式的初始计数方向由 DIR 位域决定。*
4	DIR	RW/RO	计数方向 0：计数器递增计数 1：计数器递减计数 *注：当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。*

位域	名称	权限	功能描述
3	OPM	RW	单脉冲模式 0：计数器在发生更新事件时不会停止计数 1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）
2	URS	RW	更新请求源 此位由软件置 1 和清零，用以选择 UEV 事件源。 0：使能 UEV 时，以下所有事件都会产生更新中断。包括： – 计数器上溢 / 下溢 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：使能 UEV 时，只有计数器上溢 / 下溢会生成更新中断。
1	UDIS	RW	更新禁止 此位由软件置 1 和清零，用以使能 / 禁止 UEV 事件生成。 0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成，然后更新影子寄存器的值： – 计数器上溢 / 下溢 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRy）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。
0	CEN	RW	计数器使能 0：禁止计数器 1：使能计数器 *注 1：只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。* *注 2：在单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。*

13.9.2 GTIMx_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:27	RFU	-	保留位，请保持默认值
26:25	MMSH	RW	主模式选择 为 MMS 的最高位，需配合 MMS 使用
24:8	RFU	-	保留位，请保持默认值
7	TI1S	RW	TI1 输入选择 0：GTIMx_CH1 引脚连接到 TI1 输入 1：GTIMx_CH1、CH2 和 CH3 引脚的异或组合连接到 TI1 输入
6:4	MMS	RW	主模式选择 这些位可选择主模式下将要发送到从定时器以实现同步的信息 (TRGO)。配合 MMSH 使用，这些位的组合如下： 0000：复位——GTIMx_EGR 寄存器中的 UG 位用作触发输出 (TRGO)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。 0001：使能——计数器使能信号 CNT_EN 用作触发输出 (TRGO)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑与运算组合而成。当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主 / 从模式时除外（请参见 13.9.3 GTIMx_SMCR 从模式控制寄存器中 MSM 位的说明）。 0010：更新——选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。 0011：比较捕获脉冲——通道 1 一旦发生输入捕获或比较匹配事件，触发输出会发送一个正脉冲 (TRGO)，即使 CC1IF 标志已经被置 1（或即使已经为高电平）。 0100：比较捕获脉冲——通道 2 一旦发生输入捕获或比较匹配事件，触发输出会发送一个正脉冲 (TRGO)，即使 CC2IF 标志已经被置 1（或即使已经为高电平）。 0101：比较捕获脉冲——通道 3 一旦发生输入捕获或比较匹配事件，触发输出会发送一个正脉冲 (TRGO)，即使 CC3IF 标志已经被置 1（或即使已经为高电平）。 0110：比较捕获脉冲——通道 4 一旦发生输入捕获或比较匹配事件，触发输出会发送一个正脉冲 (TRGO)，即使 CC4IF 标志已经被置 1（或即使已经为高电平）。 0111：OC1_REF——OC1REF 信号用作触发输出 (TRGO)。 1000：OC2_REF——OC2REF 信号用作触发输出 (TRGO)。 1001：OC3_REF——OC3REF 信号用作触发输出 (TRGO)。 1010：OC4_REF——OC4REF 信号用作触发输出 (TRGO)。 1011：OC1_REFC——OC1REFC 信号用作触发输出 (TRGO)。 1100：OC2_REFC——OC2REFC 信号用作触发输出 (TRGO)。 1101：OC3_REFC——OC3REFC 信号用作触发输出 (TRGO)。 1110：OC4_REFC——OC4REFC 信号用作触发输出 (TRGO)。

位域	名称	权限	功能描述
6:4（续）	MMS	RW	1111：编码器时钟输出——编码器时钟信号被用作触发输出（TRGO）。此代码适用于以下 SMS[3:0] 值：0001、0010、0011、1010、1011、1100、1101、1110、1111。不允许任何其他 SMS[3:0] 值，可能导致意外行为。 *注：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。*
3:0	RFU	-	保留位，请保持默认值

13.9.3 GTIMx_SMCR 从模式控制寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:26	RFU	-	保留位，请保持默认值
25	SMSPS	RW	预加载源 此位选择 SMS[3:0] 位字段从预装载寄存器转移到有效寄存器的触发源 0：传输由计时器的更新事件触发 1：传输由索引事件触发
24	SMSPE	RW	预加载使能 此位选择 SMS[3:0] 位是否预装载 0：禁止预装载 1：使能预装载
23:22	RFU	-	保留位，请保持默认值
21:20	TSH	RW	触发选择 为 TS 的最高位，需配合 TS 使用
19:17	RFU	-	保留位，请保持默认值
16	SMSH	RW	从模式选择 为 SMS 的最高位，需配合 SMS 使用
15	ETP	RW	外部触发极性 此位可选择将 ETR 还是 ETR 的反相用于触发操作 0：ETR 未反相，高电平或上升沿有效 1：ETR 反相，低电平或下降沿有效
14	ECE	RW	外部时钟使能 此位可使能外部时钟模式 2 0：禁止外部时钟模式 2 1：使能外部时钟模式 2，计数器时钟由 ETRF 信号的任意有效边沿提供。 *注 1：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=00111）具有相同效果。* *注 2：外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 TRGI 不得连接 ETRF（TS 位不得为 00111）。* *注 3：如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。*
13:12	ETPS	RW	外部触发预分频器 外部触发信号 ETRP 频率不得超过 PCLK 频率的 1/4。可通过使能预分频器来降低 ETRP 频率。这种方法在输入快速外部时钟时非常有用。 00：预分频器关闭 01：2 分频 ETRP 频率 10：4 分频 ETRP 频率 11：8 分频 ETRP 频率

位域	名称	权限	功能描述
11:8	ETF	RW	外部触发滤波器 此位域可定义 ETRP 信号的采样频率和适用于 ETRP 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿。 0000：无滤波器，按 f_{DTS} 频率进行采样 0001：$f_{SAMPLING}=f_{DTS}/1$，N=2 0010：$f_{SAMPLING}=f_{DTS}/1$，N=4 0011：$f_{SAMPLING}=f_{DTS}/1$，N=8 0100：$f_{SAMPLING}=f_{DTS}/2$，N=6 0101：$f_{SAMPLING}=f_{DTS}/2$，N=8 0110：$f_{SAMPLING}=f_{DTS}/4$，N=6 0111：$f_{SAMPLING}=f_{DTS}/4$，N=8 1000：$f_{SAMPLING}=f_{DTS}/8$，N=6 1001：$f_{SAMPLING}=f_{DTS}/8$，N=8 1010：$f_{SAMPLING}=f_{DTS}/16$，N=5 1011：$f_{SAMPLING}=f_{DTS}/16$，N=6 1100：$f_{SAMPLING}=f_{DTS}/16$，N=8 1101：$f_{SAMPLING}=f_{DTS}/32$，N=5 1110：$f_{SAMPLING}=f_{DTS}/32$，N=6 1111：$f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	主 / 从模式 0：不执行任何操作 1：当前定时器的触发输入事件 (TRGI) 的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。此设置适用于由单个外部事件对多个定时器进行同步的情况。
6:4	TS	RW	触发输入 TRGI 选择 此位域可选择将要用于同步计数器的触发输入。 00000：内部触发 0 (ITR0) 00001：内部触发 1 (ITR1) 00010：内部触发 2 (ITR2) 00011：内部触发 3 (ITR3) 00100：TI1 边沿检测器 (TI1F_ED) 00101：滤波后的定时器输入 1 (TI1FP1) 00110：滤波后的定时器输入 2 (TI2FP2) 00111：外部触发输入 (ETRF) 01000：内部触发 4 (ITR4) 01001：内部触发 5 (ITR5) 01010：内部触发 6 (ITR6) 01011：内部触发 7 (ITR7) 01100：内部触发 8 (ITR8) 01101：内部触发 9 (ITR9) 01110：内部触发 10 (ITR10) 01111：内部触发 11 (ITR11)

位域	名称	权限	功能描述
6:4（续）	TS	RW	10000：内部触发 12 (ITR12) 其他值：保留 *注 1：这些位只能在未使用的情况下（例如，SMS=0000 时）进行更改，以避免转换时出现错误的边沿检测。* *注 2：其他位位于同一寄存器的位 21、20，需配合 TS 使用。* *注 3：具体 ITRx 来源请参见表 13-13 ITR 信号来源。* *注 4：切勿选择 GTIMx 自己输出的 Trgo。*
3	OCCS	RW	OCREF 清零选择 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 OCREF_CLR 输入 1：OCREF_CLR_INT 连接到 ETRF *注：OCREF_CLR 输入源请参见 13.9.23 GTIMx_AF2 复用功能选项寄存器 2 的 OCRSEL 位域说明。*
2:0	SMS	RW	从模式选择 选择外部信号时，触发信号 (TRGI) 的有效边沿与外部输入上所选的极性相关（请参见输入相关控制寄存器）。 0000：禁止从模式——如果 CEN=“1”，预分频器时钟直接由内部时钟提供。 0001：正交编码器模式——x2 模式，计数器根据 TI2FP2 电平在 TI1FP1 边沿递增 / 递减计数。 0010：正交编码器模式——x2 模式，计数器根据 TI1FP1 电平在 TI2FP2 边沿递增 / 递减计数。 0011：正交编码器模式——x4 模式，计数器在 TI1FP1 和 TI2FP2 的边沿计数，计数的方向取决于另外一个输入的电平。 0100：复位模式——在出现所选触发输入 (TRGI) 上升沿时，重新初始化计数器并生成一个寄存器更新事件。 0101：门控模式——触发输入 (TRGI) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。 0110：触发模式——触发信号 (TRGI) 出现上升沿时启动计数器（但不复位）。只控制计数器的启动。 0111：外部时钟模式 1——由所选触发信号 (TRGI) 的上升沿提供计数器时钟。 1000：组合复位 + 触发模式——在出现所选触发输入 (TRGI) 上升沿时，重新初始化计数器，生成一个寄存器更新事件并启动计数器。 1001：组合门控 + 复位模式——当触发输入 (TRGI) 为高电平时计数器被使能并开始计数，当触发输入变为低电平时计数器停止计数并被复位，在此模式计数器的启动和停止都被控制。 1010：编码模式——时钟 + 方向，x2 模式。 1011：编码模式——时钟 + 方向，x1 模式，TI2FP2 的边沿极性由 CC2P 设置。 1100：编码模式——带方向时钟，x2 模式。 1101：编码模式——带方向时钟，x1 模式，TI1FP1 和 TI2FP2 的边沿极性由 CC1P 和 CC2P 设置。

位域	名称	权限	功能描述
2:0（续）	SMS	RW	1110：正交编码器模式——x1 模式，仅计数 TI1FP1 边沿，其边沿极性由 CC1P 设置。 1111：正交编码器模式——x1 模式，仅计数 TI2FP2 边沿，其边沿极性由 CC2P 设置。 *注 1：如果将 TI1F_ED 选作触发输入 (TS=00100)，则不得使用门控模式。实际上，TI1F 每次转换时，TI1F_ED 都输出 1 个脉冲，而门控模式检查的则是触发信号的电平。* *注 2：必须先使能接收 TRGO 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。* *注 3：从模式只有配置成 0100、0101、0110、1000、1001 时，有 TIE 中断标志，其余从模式没有 TIE 中断标志。*

13.9.4 GTIMx_IER 中断使能寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRIE	RW	编码器转换错误中断使能控制 0：禁止转换错误中断 1：使能转换错误中断
22	IERRIE	RW	编码器索引错误中断使能控制 0：禁止索引错误中断 1：使能索引错误中断
21	DIRIE	RW	编码器方向改变中断使能控制 0：禁止方向改变中断 1：使能方向改变中断
20	IDXIE	RW	编码器索引中断使能控制 0：禁止索引中断 1：使能索引中断
19:7	RFU	-	保留位，请保持默认值
6	TIE	RW	触发中断使能 0：禁止触发中断 1：使能触发中断 *注：工作在从模式时，该位只在从模式配置为 0100、0101、0110、1000、1001 时有效。*
5	RFU	-	保留位，请保持默认值

位域	名称	权限	功能描述
4	CC4IE	RW	捕获 / 比较 4 中断使能 0：禁止捕获 / 比较 4 中断 1：使能捕获 / 比较 4 中断
3	CC3IE	RW	捕获 / 比较 3 中断使能 0：禁止捕获 / 比较 3 中断 1：使能捕获 / 比较 3 中断
2	CC2IE	RW	捕获 / 比较 2 中断使能 0：禁止捕获 / 比较 2 中断 1：使能捕获 / 比较 2 中断
1	CC1IE	RW	捕获 / 比较 1 中断使能 0：禁止捕获 / 比较 1 中断 1：使能捕获 / 比较 1 中断
0	UIE	RW	更新中断使能 0：禁止更新中断 1：使能更新中断

13.9.5 GTIMx_ISR 中断标志寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRF	RW	编码器转换错误中断标志 0：未发生转换错误 1：已发生转换错误
22	IERRF	RW	编码器索引错误中断标志 0：未发生索引错误 1：已发生索引错误
21	DIRF	RW	编码器方向改变中断标志 0：未发生编码方向改变 1：已发生编码方向改变
20	IDXF	RW	编码器索引中断标志 0：未发生索引标志 1：已发生索引标志
19:13	RFU	-	保留位，请保持默认值
12	CC4OF	RW	捕获 / 比较 4 重复捕获标志，请参见 CC1OF 说明
11	CC3OF	RW	捕获 / 比较 3 重复捕获标志，请参见 CC1OF 说明
10	CC2OF	RW	捕获 / 比较 2 重复捕获标志，请参见 CC1OF 说明
9	CC1OF	RW	捕获 / 比较 1 重复捕获标志 仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。 0：未检测到重复捕获 1：GTIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1
8:7	RFU	-	保留位，请保持默认值
6	TIF	RW	触发中断标志 在除门控模式以外的所有模式下，当使能从模式控制器后出现 TRGI 触发事件（在 TRGI 输入上检测到有效边沿）时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动和停止时置 1。该标志需要通过软件清零。 0：未发生触发事件 1：触发中断挂起
5	RFU	-	保留位，请保持默认值
4	CC4IF	RW	捕获 / 比较 4 中断标志，请参见 CC1IF 说明
3	CC3IF	RW	捕获 / 比较 3 中断标志，请参见 CC1IF 说明
2	CC2IF	RW	捕获 / 比较 2 中断标志，请参见 CC1IF 说明

位域	名称	权限	功能描述
1	CC1IF	RW	捕获 / 比较 1 中断标志 该标志由硬件置 1。它由软件清零（输入捕获或输出比较模式），或通过读取 GTIMx_CCR1 寄存器清零（仅限输入捕获模式）。 0：未发生比较匹配 / 输入捕获事件 1：发生比较匹配 / 输入捕获事件 **如果通道 CC1 配置为输出：** 当计数器 GTIMx_CNT 的值与 GTIMx_CCR1 寄存器的值匹配时，此标志置 1。当 GTIMx_CCR1 的值大于 GTIMx_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和中心对齐模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下有 3 种可能的标志设置选项，有关完整说明，请参见 GTIMx_CR1 寄存器中的 CMS 位。 **如果通道 CC1 配置为输入：** 该位在以下情况下置 1：在 GTIMx_CCR1 寄存器中捕获了计数器值（根据 GTIMx_CCER 中的 CC1P 和 CC1NP 位设置定义的边沿灵敏度，且在 IC1 上检测到一个边沿）。
0	UIF	RW	更新中断标志 该位在发生更新事件时通过硬件置 1。但需要通过软件清零。 0：未发生更新 1：更新中断挂起。 该位在以下情况下更新寄存器时由硬件置 1： 上溢或下溢并且当 GTIMx_CR1 寄存器中 UDIS=0 时。 GTIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 GTIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。 GTIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参见从模式控制寄存器说明）。

13.9.6 GTIMx_ICR 中断标志清除寄存器

Address offset: 0x70 Reset value: 0x00F0 1E5F

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRF	R1W0	编码器转换错误中断标志清除 0：清除编码器转换错误中断标志 1：无功能
22	IERRF	R1W0	编码器索引错误中断标志清除 0：清除编码器索引错误中断标志 1：无功能
21	DIRF	R1W0	编码器方向改变中断标志清除 0：清除编码器方向改变中断标志 1：无功能
20	IDXF	R1W0	编码器索引中断标志清除 0：清除编码器索引中断标志 1：无功能
19:13	RFU	-	保留位，请保持默认值
12	CC4OF	R1W0	捕获 / 比较 4 重复捕获标志清除，请参见 CC1OF 说明
11	CC3OF	R1W0	捕获 / 比较 3 重复捕获标志清除，请参见 CC1OF 说明
10	CC2OF	R1W0	捕获 / 比较 2 重复捕获标志清除，请参见 CC1OF 说明
9	CC1OF	R1W0	捕获 / 比较 1 重复捕获标志清除 0：清除捕获 / 比较 1 的重复捕获标志 1：无功能
8:7	RFU	-	保留位，请保持默认值
6	TIF	R1W0	触发中断标志清除 0：清除触发中断标志 1：无功能
5	RFU	-	保留位，请保持默认值
4	CC4IF	R1W0	捕获 / 比较 4 中断标志清除，请参见 CC1IF 说明
3	CC3IF	R1W0	捕获 / 比较 3 中断标志清除，请参见 CC1IF 说明
2	CC2IF	R1W0	捕获 / 比较 2 中断标志清除，请参见 CC1IF 说明
1	CC1IF	R1W0	捕获 / 比较 1 中断标志清除 0：清除捕获 / 比较 1 中断标志 1：无功能
0	UIF	R1W0	更新中断标志清除 0：清除更新中断标志 1：无功能

13.9.7 GTIMx_EGR 事件生成寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	TG	RW	触发生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：GTIMx_ISR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件。
5	RFU	-	保留位，请保持默认值
4	CC4G	RW	捕获 / 比较 4 生成，请参见 CC1G 说明
3	CC3G	RW	捕获 / 比较 3 生成，请参见 CC1G 说明
2	CC2G	RW	捕获 / 比较 2 生成，请参见 CC1G 说明
1	CC1G	RW	捕获 / 比较 1 生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：通道 1 上生成捕获 / 比较事件，根据 CC1 的配置分为以下情况： 如果通道 CC1 配置为输出： 使能时，CC1IF 标志置 1 并发送相应的中断请求。 如果通道 CC1 配置为输入： GTIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断请求。如果 CC1IF 标志已为高，CC1OF 标志将置 1。
0	UG	RW	更新生成 该位可通过软件置 1，并由硬件自动清零。 0：不执行任何操作 1：重新初始化计数器并生成寄存器更新事件。请注意，预分频器的计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或递增计数模式，计数器将清零；如果选择递减计数模式，计数器将使用自动重载值 GTIMx_ARR。

13.9.8 GTIMx_CCMR1CAP 捕获模式寄存器 1

Address offset: 0x18 Reset value: 0x0000 0000

此寄存器和 GTIMx_CCMR1CMP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	IC2F	RW	输入捕获 2 滤波器，请参见 IC1F 说明
11:10	IC2PSC	RW	输入捕获 2 预分频器，请参见 IC1PSC 说明
9:8	CC2S	RW	捕获 / 比较 2 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 上 10：CC2 通道配置为输入，IC2 映射到 TI1 上 11：CC2 通道配置为输入，IC2 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC2E=0），才可向 CC2S 位写入数据。*
7:4	IC1F	RW	输入捕获 1 滤波器 此位域可定义 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿。 0000：无滤波器，按 f_{DTS} 频率进行采样 0001：$f_{SAMPLING}=f_{PCLK}$，N=2 0010：$f_{SAMPLING}=f_{PCLK}$，N=4 0011：$f_{SAMPLING}=f_{PCLK}$，N=8 0100：$f_{SAMPLING}=f_{DTS}/2$，N=6 0101：$f_{SAMPLING}=f_{DTS}/2$，N=8 0110：$f_{SAMPLING}=f_{DTS}/4$，N=6 0111：$f_{SAMPLING}=f_{DTS}/4$，N=8 1000：$f_{SAMPLING}=f_{DTS}/8$，N=6 1001：$f_{SAMPLING}=f_{DTS}/8$，N=8 1010：$f_{SAMPLING}=f_{DTS}/16$，N=5 1011：$f_{SAMPLING}=f_{DTS}/16$，N=6 1100：$f_{SAMPLING}=f_{DTS}/16$，N=8 1101：$f_{SAMPLING}=f_{DTS}/32$，N=5 1110：$f_{SAMPLING}=f_{DTS}/32$，N=6 1111：$f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC	RW	输入捕获 1 预分频器 此位域定义 CC1 输入 (IC1) 的预分频比。只要 CC1E=0（GTIMx_CCER 寄存器），预分频器便立即复位。 00：无预分频器，捕获输入上每检测到一个边沿便执行捕获 01：每发生 2 个事件便执行一次捕获 10：每发生 4 个事件便执行一次捕获 11：每发生 8 个事件便执行一次捕获

位域	名称	权限	功能描述
1:0	CC1S	RW	捕获 / 比较 1 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 上 10：CC1 通道配置为输入，IC1 映射到 TI2 上 11：CC1 通道配置为输入，IC1 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC1E=0），才可向 CC1S 位写入数据。*

13.9.9 GTIMx_CCMR1CMP 比较模式寄存器 1

Address offset: 0x18 Reset value: 0x0000 0000

此寄存器和 GTIMx_CCMR1CAP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	OC2MH	RW	配合 OC2M 使用，为 OC2M 的最高位
23:17	RFU	-	保留位，请保持默认值
16	OC1MH	RW	配合 OC1M 使用，为 OC1M 的最高位
15	OC2CE	RW	输出比较 2 清零使能，请参见 OC1CE 说明
14:12	OC2M	RW	输出比较 2 模式，请参见 OC1M 说明
11	OC2PE	RW	输出比较 2 预装载使能，请参见 OC1PE 说明
10	OC2FE	RW	输出比较 2 快速使能，请参见 OC1FE 说明
9:8	CC2S	RW	捕获 / 比较 2 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 上 10：CC2 通道配置为输入，IC2 映射到 TI1 上 11：CC2 通道配置为输入，IC2 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC2E=0），才可向 CC2S 位写入数据。*
7	OC1CE	RW	输出比较 1 清零使能 0：OC1REF 不受 OCREF_CLR_INT 输入影响 1：OCREF_CLR_INT 输入上检测到高电平时，OC1REF 立即清零

位域	名称	权限	功能描述
6:4	OC1M	RW	输出比较 1 模式 这些位定义提供 OC1 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 的有效电平则取决于 CC1P 位。 0000：冻结——输出比较寄存器 GTIMx_CCR1 与计数器 GTIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。 0001：将通道 1 设置为匹配时输出有效电平。当计数器 GTIMx_CNT 与捕获 / 比较寄存器 1(GTIMx_CCR1) 匹配时，OC1REF 信号强制变为高电平。 0010：将通道 1 设置为匹配时输出无效电平。当计数器 GTIMx_CNT 与捕获 / 比较寄存器 1(GTIMx_CCR1) 匹配时，OC1REF 信号强制变为低电平。 0011：翻转——GTIMx_CNT=GTIMx_CCR1 时，OC1REF 发生翻转。 0100：强制变为无效电平——OC1REF 强制变为低电平。 0101：强制变为有效电平——OC1REF 强制变为高电平。 0110：PWM 模式 1——在递增计数模式下，只要 GTIMx_CNT< GTIMx_CCR1，通道 1 便为有效状态 (OC1REF=1)，否则为无效状态 (OC1REF=0)。在递减计数模式下，只要 GTIMx_CNT> GTIMx_CCR1，通道 1 便为无效状态 (OC1REF=0)，否则为有效状态 (OC1REF=1)。 0111：PWM 模式 2——在递增计数模式下，只要 GTIMx_CNT< GTIMx_CCR1，通道 1 便为无效状态 (OC1REF=0)，否则为有效状态 (OC1REF=1)。在递减计数模式下，只要 GTIMx_CNT> GTIMx_CCR1，通道 1 便为有效状态 (OC1REF=1)，否则为无效状态 (OC1REF=0)。 1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。在递减计数模式下，通道为无效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。 1001：可再触发 OPM 模式 2——在递增计数模式下，通道为无效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为有效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为有效状态。 1010：保留 1011: 计数方向输出，OC1REF 信号为 DIR 位的计数方向信号。 1100：组合 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑或运算结果。 1101：组合 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑与运算结果。

位域	名称	权限	功能描述
6:4（续）	OC1M	RW	1110：不对称 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。 1111：不对称 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。 *注：在 PWM 模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCyREF 电平才会发生更改。*
3	OC1PE	RW	输出比较 1 预装载使能 0：禁止与 GTIMx_CCR1 相关的预装载寄存器。可随时向 GTIMx_CCR1 写入数据，写入后将立即使用新值。 1：使能与 GTIMx_CCR1 相关的预装载寄存器。可读 / 写访问预装载寄存器。GTIMx_CCR1 预装载值在每次生成更新事件时都会装载到有效寄存器中。 *注：只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（GTIMx_CR1 寄存器中的 OPM 位置 1）。其他情况下则无法保证该行为。*
2	OC1FE	RW	输出比较 1 快速使能 此位用于加快触发输入事件对 CC 输出的影响。 0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。 1: 触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCyFE 才会起作用。
1:0	CC1S	RW	捕获 / 比较 1 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 上 10：CC1 通道配置为输入，IC1 映射到 TI2 上 11：CC1 通道配置为输入，IC1 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC1E=0），才可向 CC1S 位写入数据。*

13.9.10 GTIMx_CCMR2CAP 捕获模式寄存器 2

Address offset: 0x1C Reset value: 0x0000 0000

此寄存器和 GTIMx_CCMR2CMP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	IC4F	RW	输入捕获 4 滤波器，请参见 IC1F 说明
11:10	IC4PSC	RW	输入捕获 4 预分频器，请参见 IC1PSC 说明
9:8	CC4S	RW	捕获 / 比较 4 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 上 10：CC4 通道配置为输入，IC4 映射到 TI3 上 11：CC4 通道配置为输入，IC4 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC4E=0），才可向 CC4S 位写入数据。*
7:4	IC3F	RW	输入捕获 3 滤波器，请参见 IC1F 说明
3:2	IC3PSC	RW	输入捕获 3 预分频器，请参见 IC1PSC
1:0	CC3S	RW	捕获 / 比较 3 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 上 10：CC3 通道配置为输入，IC3 映射到 TI4 上 11：CC3 通道配置为输入，IC3 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC3E=0），才可向 CC3S 位写入数据。*

13.9.11 GTIMx_CCMR2CMP 比较模式寄存器 2

Address offset: 0x1C Reset value: 0x0000 0000

此寄存器和 GTIMx_CCMR2CAP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	OC4MH	RW	配合 OC4M 使用，为 OC4M 的最高位
23:17	RFU	-	保留位，请保持默认值
16	OC3MH	RW	配合 OC3M 使用，为 OC3M 的最高位
15	OC4CE	RW	输出比较 4 清零使能，请参见 OC1CE 说明
14:12	OC4M	RW	输出比较 4 模式，请参见 OC1M 说明
11	OC4PE	RW	输出比较 4 预装载使能，请参见 OC1PE 说明
10	OC4FE	RW	输出比较 4 快速使能，请参见 OC1FE 说明
9:8	CC4S	RW	捕获 / 比较 4 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 上 10：CC4 通道配置为输入，IC4 映射到 TI3 上 11：CC4 通道配置为输入，IC4 映射到 TRC 上，此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC4E=0），才可向 CC4S 位写入数据。*
7	OC3CE	RW	输出比较 3 清零使能，请参见 OC1CE 说明
6:4	OC3M	RW	输出比较 3 模式，请参见 OC1M 说明
3	OC3PE	RW	输出比较 3 预装载使能，请参见 OC1PE 说明
2	OC3FE	RW	输出比较 3 快速使能，请参见 OC1FE 说明
1:0	CC3S	RW	捕获 / 比较 3 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 上 10：CC3 通道配置为输入，IC3 映射到 TI4 上 11：CC3 通道配置为输入，IC3 映射到 TRC 上。此模式仅在通过 TS 位（GTIMx_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（GTIMx_CCER 中的 CC3E=0），才可向 CC3S 位写入数据。*

13.9.12 GTIMx_CCER 捕获 / 比较使能寄存器

Address offset: 0x20 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	CC4NP	RW	捕获 / 比较 4 互补输出极性，请参见 CC1NP
14	RFU	-	保留位，请保持默认值
13	CC4P	RW	捕获 / 比较 4 输出极性，请参见 CC1P 说明
12	CC4E	RW	捕获 / 比较 4 输出使能，请参见 CC1E 说明
11	CC3NP	RW	捕获 / 比较 3 互补输出极性，请参见 CC1NP
10	RFU	-	保留位，请保持默认值
9	CC3P	RW	捕获 / 比较 3 输出极性，请参见 CC1P 说明
8	CC3E	RW	捕获 / 比较 3 输出使能，请参见 CC1E 说明
7	CC2NP	RW	捕获 / 比较 2 互补输出极性，请参见 CC1NP
6	RFU	-	保留位，请保持默认值
5	CC2P	RW	捕获 / 比较 2 输出极性，请参见 CC1P 说明
4	CC2E	RW	捕获 / 比较 2 输出使能，请参见 CC1E 说明
3	CC1NP	RW	捕获 / 比较 1 互补输出极性 CC1 通道配置为输出：在这种情况下，CC1NP 必须保持清零。 CC1 通道配置为输入：该位与 CC1P 配合使用可定义 TI1FP1/TI2FP1 极性。请参见 CC1P 说明。
2	RFU	-	保留位，请保持默认值
1	CC1P	RW	捕获 / 比较 1 输出极性 0：OC1 高电平有效（输出模式）/ 边沿灵敏度选择（输入模式） 1：OC1 低电平有效（输出模式）/ 边沿灵敏度选择（输入模式） CC1 通道配置为输入时，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的有效极性。 – CC1NP=0，CC1P=0：非反相 / 上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。 – CC1NP=0，CC1P=1：反相 / 下降沿触发。电路对 TIxFP1 下升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。 – CC1NP=1，CC1P=1：非反相 / 双边沿触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。 – CC1NP=1，CC1P=0：该配置被保留，不得使用。

位域	名称	权限	功能描述
0	CC1E	RW	捕获 / 比较 1 输出使能 0：禁止捕获模式 /OC1 未激活 1：使能捕获模式 / 在相应输出引脚上输出 OC1 信号 *注：使用输入捕获功能时，该位必须配置为 1。*

13.9.13 GTIMx_CNT 计数寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	UIFCPY	RO	根据 GTIMx_CR1 寄存器中 UIFREMAP 位域的值，本位域表示不同的含义： UIFREMAP = 0，本位域保留，读为 0 UIFREMAP = 1，本位域表示 GTIMx_ISR 寄存器的 UIF 位的只读副本
30:16	RFU	-	保留位，请保持默认值
15:0	CNT	RW	计数值

13.9.14 GTIMx_PSC 预分频寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PSC	RW	预分频器值 计数器时钟频率 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)。 PSC 包含每次发生更新事件（包括计数器通过 GTIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到有效预分频器寄存器的值。

13.9.15 GTIMx_ARR 自动重载寄存器

Address offset: 0x2C Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	ARR	RW	自动重载值 ARR 为要装载到实际自动重载寄存器的值，当自动重载值为空时，计数器不工作。

13.9.16 GTIMx_CCR1 捕获 / 比较寄存器 1

Address offset: 0x34 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR1	RW	捕获 / 比较 1 值 **如果通道 CC1 配置为输出：** CCR1 是捕获 / 比较寄存器 1 的预装载值。 如果没有通过 GTIMx_CCMR1CMP 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获 / 比较寄存器 1）。有效捕获 / 比较寄存器中包含要与计数器 GTIMx_CNT 进行比较并在 OC1 输出上发出信号的值。 **如果通道 CC1 配置为输入：** CCR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。只能读取 GTIMx_CCR1 寄存器，无法对其进行编程。

13.9.17 GTIMx_CCR2 捕获 / 比较寄存器 2

Address offset: 0x38 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR2	RW	捕获 / 比较 2 值 **如果通道 CC2 配置为输出：** CCR2 是捕获 / 比较寄存器 2 的预装载值。 如果没有通过 GTIMx_CCMR1CMP 寄存器中的 OC2PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获 / 比较寄存器 2）。有效捕获 / 比较寄存器中包含要与计数器 GTIMx_CNT 进行比较并在 OC2 输出上发出信号的值。 **如果通道 CC2 配置为输入：** CCR2 为上一个输入捕获 2 事件 (IC2) 发生时的计数器值。只能读取 GTIMx_CCR2 寄存器，无法对其进行编程。

13.9.18 GTIMx_CCR3 捕获 / 比较寄存器 3

Address offset: 0x3C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR3	RW	捕获 / 比较 3 值 **如果通道 CC3 配置为输出：** CCR3 是捕获 / 比较寄存器 3 的预装载值。 如果没有通过 GTIMx_CCMR2CMP 寄存器中的 OC3PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获 / 比较寄存器 3）。有效捕获 / 比较寄存器中包含要与计数器 GTIMx_CNT 进行比较并在 OC3 输出上发出信号的值。 **如果通道 CC3 配置为输入：** CCR3 为上一个输入捕获 3 事件 (IC3) 发生时的计数器值。只能读取 GTIMx_CCR3 寄存器，无法对其进行编程。

13.9.19 GTIMx_CCR4 捕获 / 比较寄存器 4

Address offset: 0x40 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR4	RW	捕获 / 比较 4 值 **如果通道 CC4 配置为输出：** CCR4 是捕获 / 比较寄存器 4 的预装载值。 如果没有通过 GTIMx_CCMR2CMP 寄存器中的 OC4PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到有效的捕获 / 比较寄存器 4）。有效捕获 / 比较寄存器中包含要与计数器 GTIMx_CNT 进行比较并在 OC4 输出上发出信号的值。 **如果通道 CC4 配置为输入：** CCR4 为上一个输入捕获 4 事件 (IC4) 发生时的计数器值。只能读取 GTIMx_CCR4 寄存器，无法对其进行编程。

13.9.20 GTIMx_ECR 编码控制寄存器

Address offset: 0x58 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:6	IPOS	RW	编码索引定位 在正交编码器模式（SMS[3:0] = 0001、0010、0011、1110、1111）中，此位表示索引事件在哪个 AB 输入配置中重置计数器。 00：当 AB = 00 时，索引会重置计数器 01：当 AB = 01 时，索引会重置计数器 10：当 AB = 10 时，索引会重置计数器 11：当 AB = 11 时，索引会重置计数器 在定向时钟编码器模式或时钟加方向编码器模式（SMS[3:0] = 1010、1011、1100、1101）中，这些位指示索引事件在哪个电平上重置计数器。在定向时钟编码器模式下，这一点都适用于两个时钟输入。 x0：当时钟信号为低电平时，索引将重置计数器 x1：当时钟信号为高电平时，索引将重置计数器
5	FIDX	RW	该位表示是否只考虑第一个索引 0：索引始终处于活动状态 1：仅第一个索引重置计数器
4:3	RFU	-	保留位，请保持默认值
2:1	IDIR	RW	索引方向 此位域表示索引在哪种计数方向上重置计数器 00：在任何计数方向上重置计数器 01：仅在向上计数时重置计数器 10：仅在向下计数时重置计数器 11：保留
0	IE	RW	索引使能 0：禁止索引 1：使能索引

13.9.21 GTIMx_TISEL TI 输入选择寄存器

Address offset: 0x5C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28	RFU	-	保留位，请保持默认值
27:24	TI4SEL	RW	TI4 通道输入捕获信号来源选择 0000：GTIMx_CH1 0001：VC1_OUT 0010：VC2_OUT 0011：保留 0100：保留 0101：MCO_OUT 0110：HSE_FAULT 0111：LSE_FAULT 1000：RTC_OUT 1001：LSI_OUT 1010：BTIM1_Trgo 1011：BTIM2_Trgo 1100：BTIM3_Trgo 1101：GTIM1_Trgo 1110：GTIM2_Trgo 1111：ATIM_Trgo *注：切勿选择 GTIMx 自己输出的 Trgo。*
23:20	RFU	-	保留位，请保持默认值
19:16	TI3SEL	RW	TI3 通道输入捕获信号来源选择 0000：GTIMx_CH1 0001：VC1_OUT 0010：VC2_OUT 0011：UART3_RXD 0100：UART3_TXD 0101：MCO_OUT 0110：HSE_FAULT 0111：LSE_FAULT 1000：RTC_OUT 1001：LSI_OUT 1010：BTIM1_Trgo 1011：BTIM2_Trgo 1100：BTIM3_Trgo 1101：GTIM1_Trgo 1110：GTIM2_Trgo 1111：ATIM_Trgo *注：切勿选择 GTIMx 自己输出的 Trgo。*
15:12	RFU	-	保留位，请保持默认值

位域	名称	权限	功能描述
11:8	TI2SEL	RW	TI2 通道输入捕获信号来源选择 0000：GTIMx_CH1 0001：VC1_OUT 0010：VC2_OUT 0011：UART2_RXD 0100：UART2_TXD 0101：MCO_OUT 0110：HSE_FAULT 0111：LSE_FAULT 1000：RTC_OUT 1001：LSI_OUT 1010：BTIM1_Trgo 1011：BTIM2_Trgo 1100：BTIM3_Trgo 1101：GTIM1_Trgo 1110：GTIM2_Trgo 1111：ATIM_Trgo *注：切勿选择 GTIMx 自己输出的 Trgo。*
7:4	RFU	-	保留位，请保持默认值
3:0	TI1SEL	RW	TI1 通道输入捕获信号来源选择 0000：GTIMx_CH1 0001：VC1_OUT 0010：VC2_OUT 0011：UART1_RXD 0100：UART1_TXD 0101：MCO_OUT 0110：HSE_FAULT 0111：LSE_FAULT 1000：RTC_OUT 1001：LSI_OUT 1010：BTIM1_Trgo 1011：BTIM2_Trgo 1100：BTIM3_Trgo 1101：GTIM1_Trgo 1110：GTIM2_Trgo 1111：ATIM_Trgo *注：切勿选择 GTIMx 自己输出的 Trgo。*

13.9.22 GTIMx_AF1 复用功能选项寄存器 1

Address offset: 0x60 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17:14	ETRSEL	RW	GTIM1 的 ETR 来源选择 0000：GTIM1_ETR 引脚 0001：VC1_OUT 0010：VC2_OUT 0101：ADC_AWD 1000：LVD_OUT 1010：GTIM2_ETR 引脚 1011：UART1_TXD 1100：UART2_TXD 1101：UART3_TXD 1110：HSE_FAULT 1111：LSE_FAULT GTIM2 的 ETR 来源选择 0000：GTIM2_ETR 引脚 0001：VC1_OUT 0010：VC2_OUT 0101：ADC_AWD 1000：LVD_OUT 1001：GTIM1_ETR 引脚 1011：UART1_TXD 1100：UART2_TXD 1101：UART3_TXD 1110：HSE_FAULT 1111：LSE_FAULT
13:0	RFU	-	保留位，请保持默认值

13.9.23 GTIMx_AF2 复用功能选项寄存器 2

Address offset: 0x64 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18:16	OCRSEL	RW	OCREF_CLR 源选择 000：VC1_OUT 001：VC2_OUT 100：ADC_AWD 111：ETRF
15:0	RFU	-	保留位，请保持默认值

14 高级定时器（ATIM）

14.1 概述

高级定时器 (ATIM) 包含一个 16bit 自动重装载计数器，并由一个可编程的预分频器驱动。ATIM 支持定时、计数、复位、门控、触发和编码器等多种工作模式，带 6 路独立的捕获 / 比较通道，可实现 6 路独立 PWM 输出或 6 对带死区互补 PWM 输出或对 6 路输入进行捕获。可用于基本的定时 / 计数、测量输入信号的脉冲宽度和周期、产生输出波形（PWM、单脉冲、插入死区时间的互补 PWM 等）。

14.2 主要特性

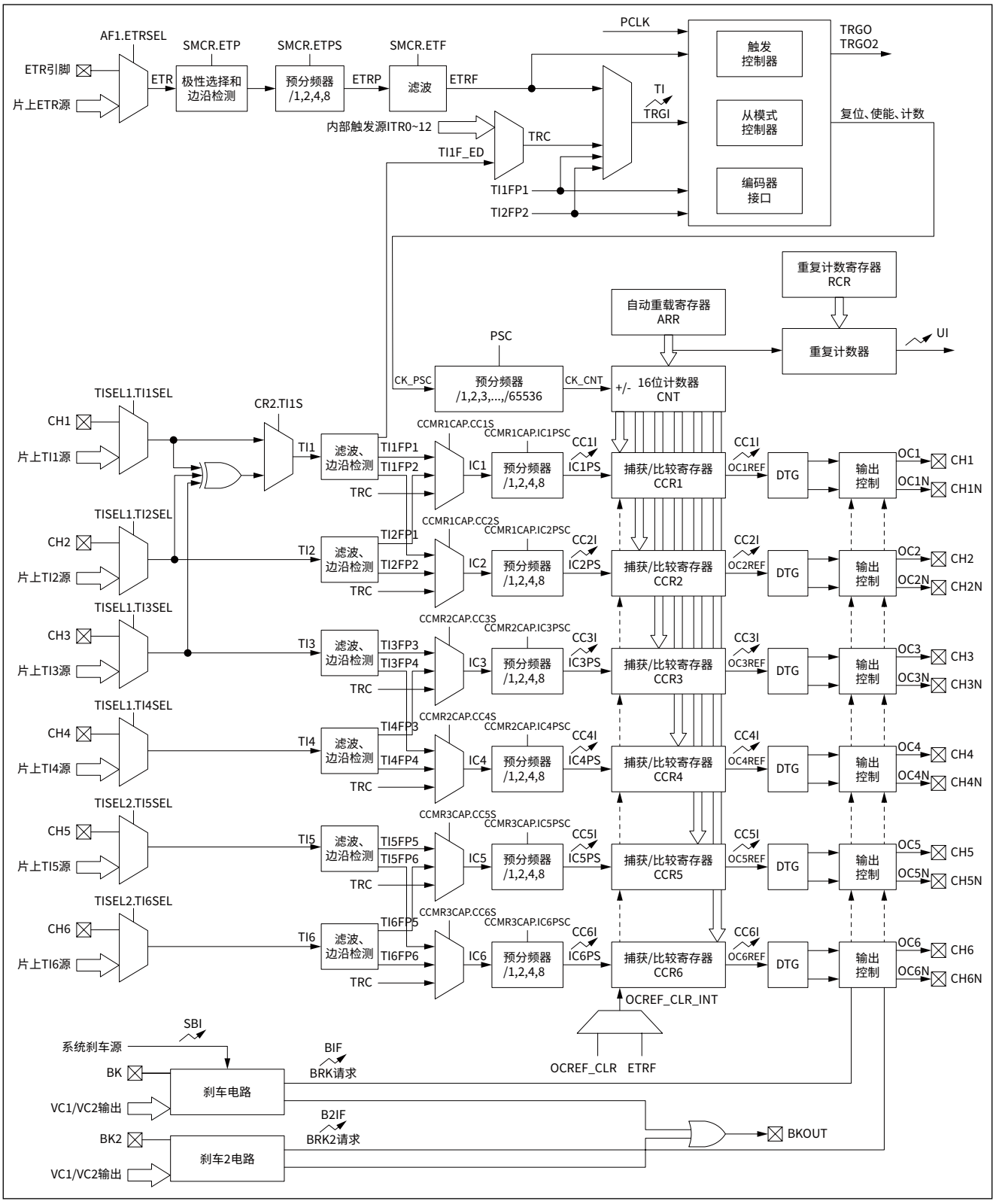
- 16bit 递增、递减和递增 / 递减自动重载计数器
- 可编程预分频器支持 1、2、3、4、…、65536 分频
- 支持单次计数模式和连续计数模式
- 6 路独立输入捕获和输出比较通道
- 死区时间可编程的互补 PWM 输出
- 支持双点比较移相，支持逻辑与移相
- 支持 2 个刹车输入
- 支持 1 个刹车输出
- 触发输入信号（TRGI）控制定时器实现多种从模式
- 定时器级联 ITR 和片内外设互联 ETR
- 支持针对定位的增量 (正交) 编码器和霍尔传感器电路
- 多种事件发生时产生中断请求：
  - 更新事件
  - 触发事件
  - 输入捕获
  - 输出比较
  - 换向事件
  - 刹车事件
  - 编码器事件

14.3 功能描述

14.3.1 功能框图

ATIM 的功能框图如下图所示：

图 14-1 ATIM 功能框图



14.3.1.1 时钟源

计数器的计数时钟源可由内部时钟 PCLK、触发信号 TRGI 或外部触发输入信号 ETRF 提供，经预分频器 ATIM_PSC 分频后驱动计数器进行计数。不同工作模式下具有不同时钟源，具体请参见 14.3.2 工作模式。

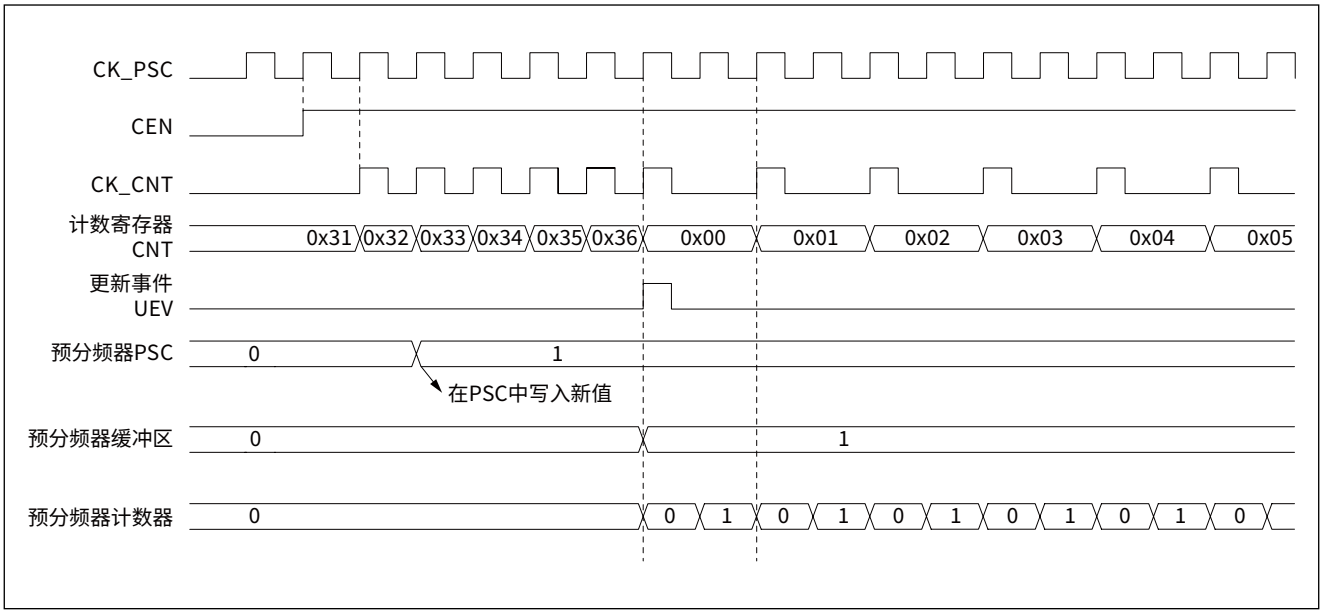
14.3.1.2 预分频器

预分频器对 CK_PSC 时钟进行分频，得到计数时钟 CK_CNT，以驱动计数器计数。分频系数通过 ATIM_PSC 寄存器进行设置，支持 1、2、3、4、…、65536 分频。

ATIM_PSC 寄存器具有缓冲功能，可在运行中修改，新的预分频值将在下一个更新事件发生时生效。

下图给出了运行过程中预分频器的分频由 1 变为 2 时的时序图：

图 14-2 预分频器分频由 1 变为 2 时的时序图



14.3.1.3 计数器与计数模式

计数器可设置为递增计数（边沿对齐模式）、递减计数（边沿对齐模式）或递增 / 递减双向计数（中心对齐模式）。具体通过控制寄存器 ATIM_CR1 的 CMS 和 DIR 位域进行配置，如下表所示：

表 14-1 计数模式

ATIM_CR1.CMS	ATIM_CR1.DIR	计数模式
00	0	递增计数（边沿对齐模式）
	1	递减计数（边沿对齐模式）
01	-	中心对齐模式 1
10	-	中心对齐模式 2
11	-	中心对齐模式 3

当设置 ATIM_CR1 寄存器的 CEN 位域为 1 时，计数器开始按设定模式计数，注意计数器将在 CEN 位置 1 时刻的一个时钟周期后开始计数。

递增计数模式

在递增计数模式下，计数器从 0 开始递增计数到重载值 ARR，然后重新从 0 开始递增计数，同时生成计数器上溢出事件并产生上溢出信号 OV（OV 信号会自动清除）。

如果使用了重复计数器，则当溢出次数达到重复值加 1（REP+1）时，将产生更新事件 UEV（UEV 信号会自动清除），否则，将在每次计数器上溢时产生更新事件 UEV，计数器更新中断标志位 ATIM_ISR.UIF 被硬件置位，如果允许中断（设置 ATIM_IER.UIE 为 1），将产生中断请求，设置 ATIM_ICR.UIF 为 0 清除该标志位。

以下是计数器在不同时钟频率下的操作示例，其中 ATIM_ARR =0x36、ATIM_RCR=0x00。

图 14-3 递增计数，内部时钟分频因子为 1

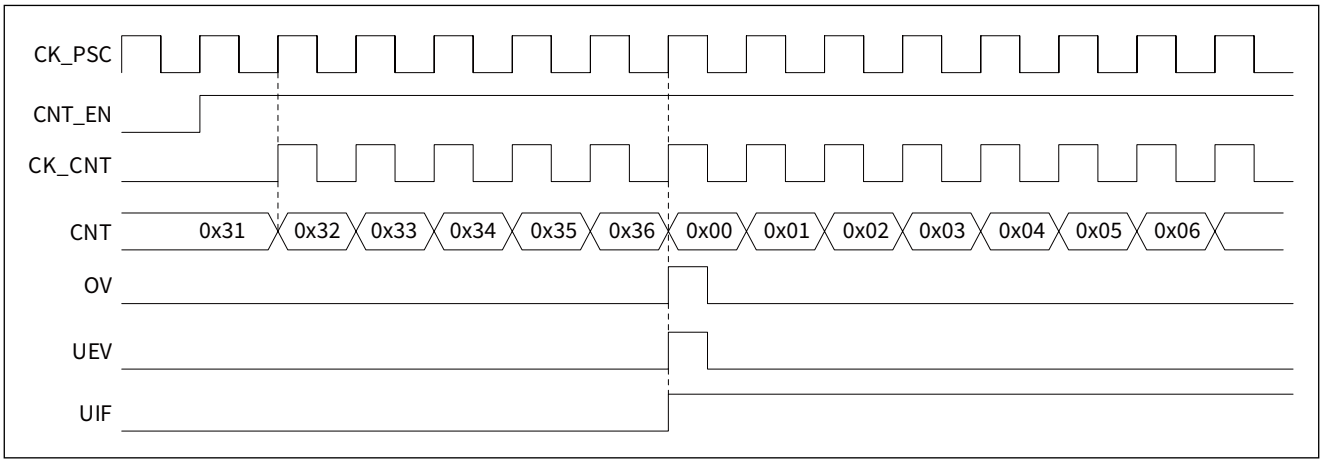


图 14-4　递增计数，内部时钟分频因子为 2

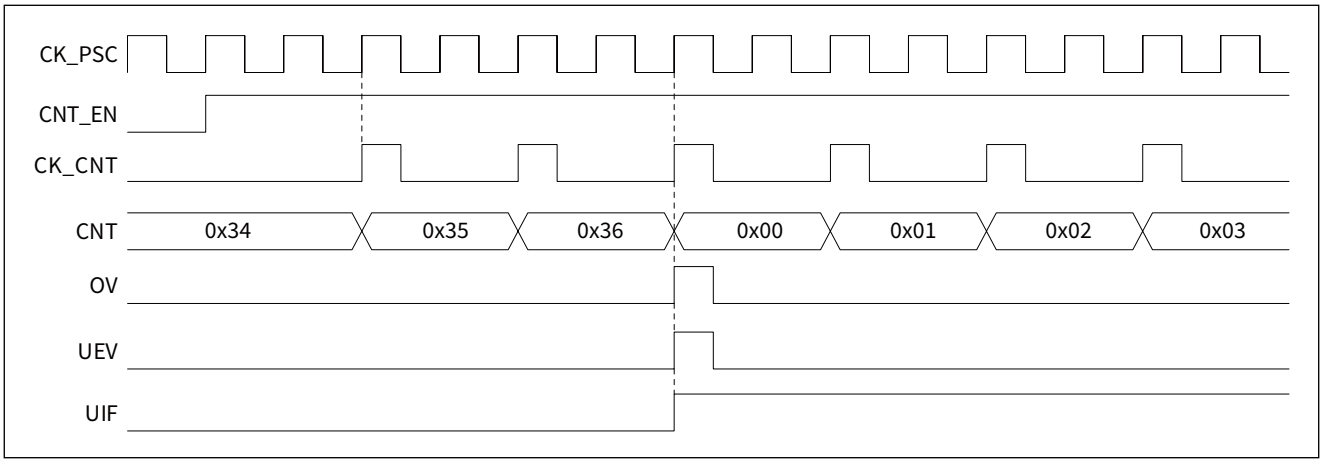


图 14-5　递增计数，内部时钟分频因子为 4

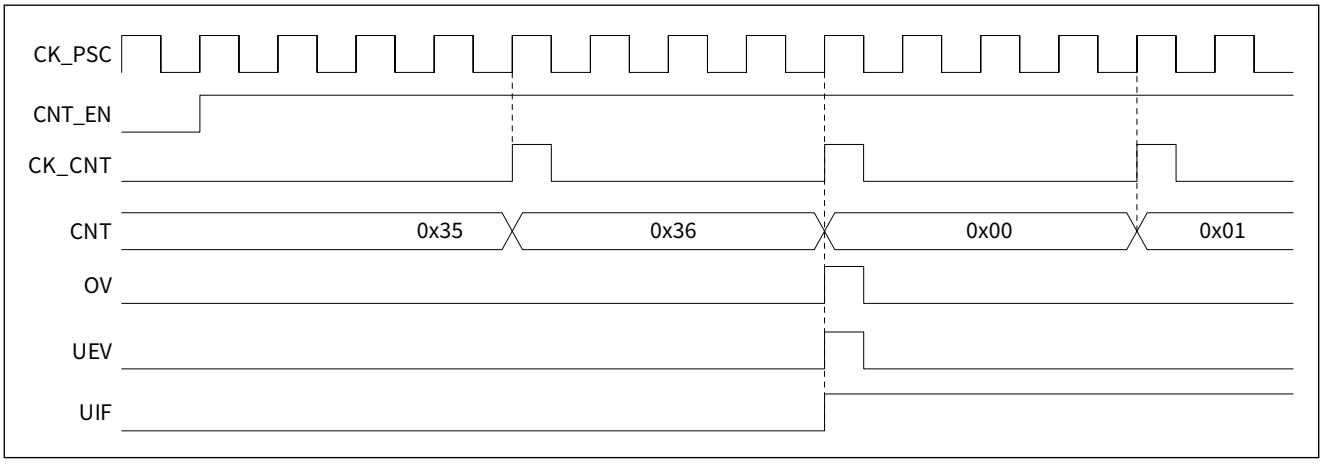


图 14-6　递增计数，内部时钟分频因子为 N

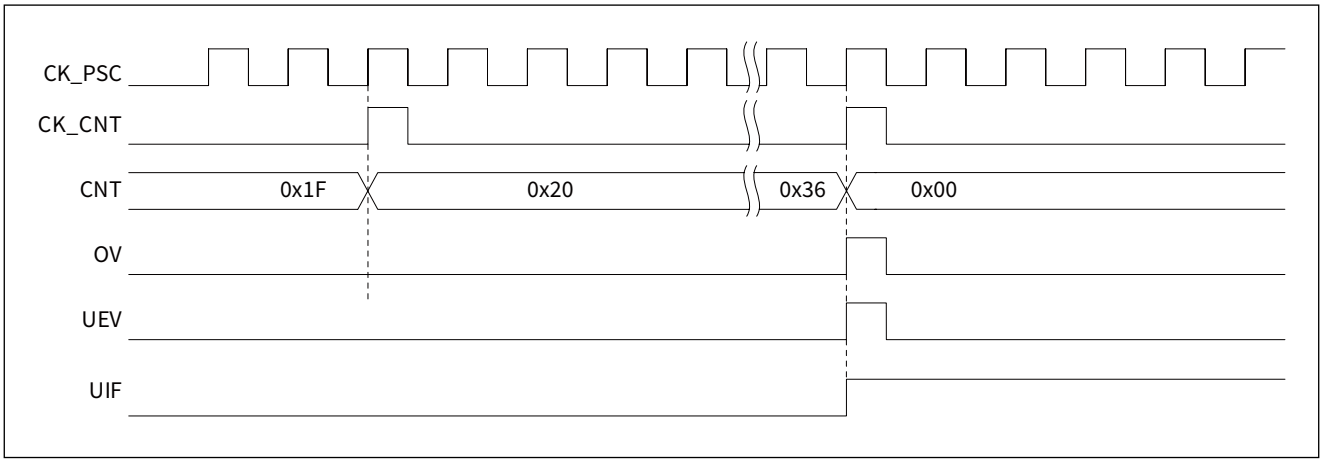


图 14-7　递增计数，当重载缓存禁止时的更新事件（ATIM_CR1.ARPE=0）

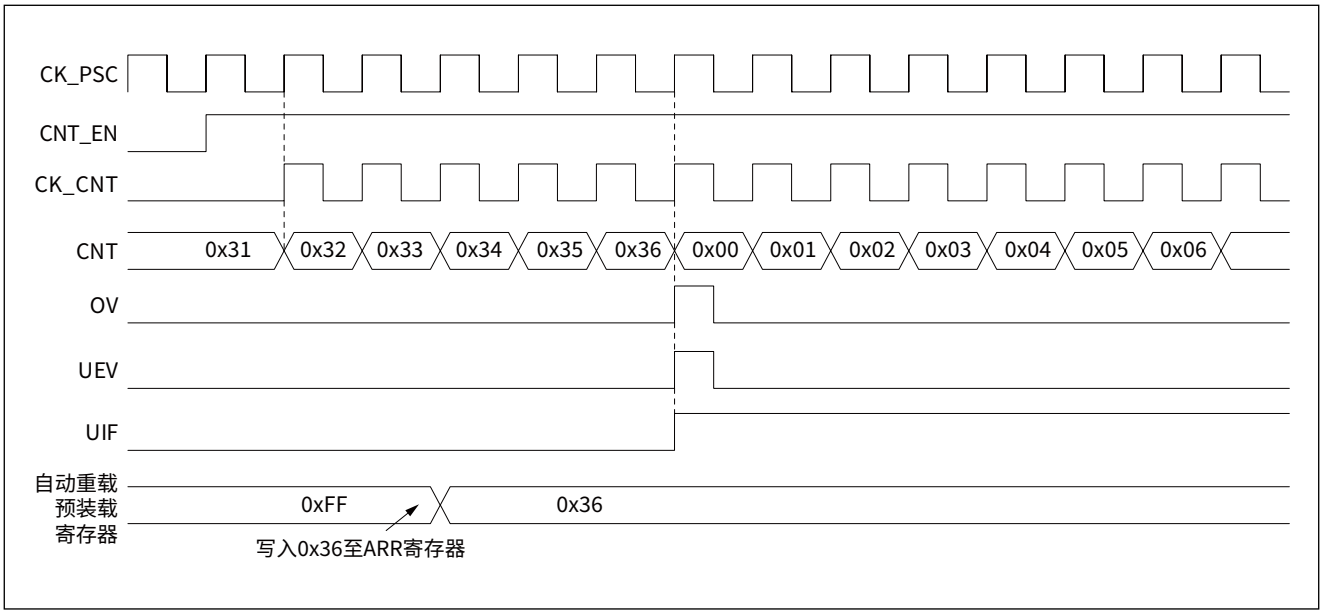
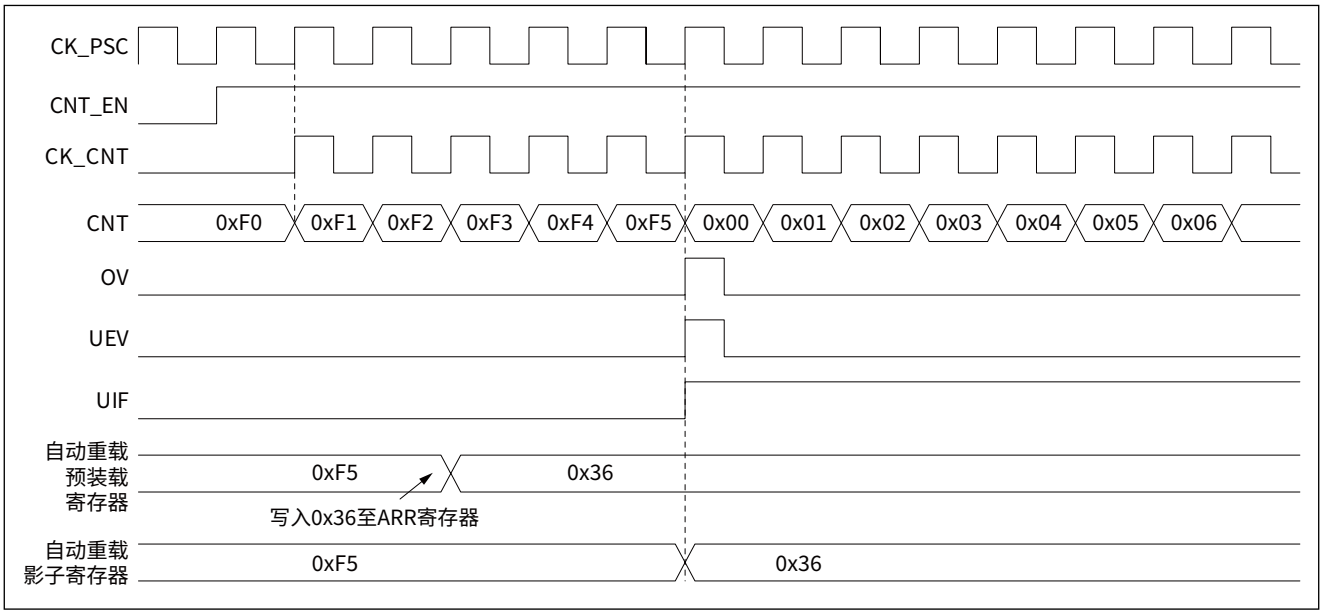


图 14-8　递增计数，当重载缓存使能时的更新事件（ATIM_CR1.ARPE=1）



递减计数模式

在边沿对齐模式下设置控制寄存器 ATIM_CR1 的 DIR 位为 1 时，计数器工作在递减计数模式。

在递减计数模式下，硬件自动加载 ARR 值到计数器 CNT，计数器开始递减计数到 0，然后重新装载 ARR 值递减计数，同时生成计数器下溢出事件并产生下溢出信号 UND（UND 信号会自动清除）。

如果使用了重复计数器，则当溢出次数达到重复值加 1（REP+1）时，将产生更新事件 UEV（UEV 信号会自动清除），否则，将在每次计数器下溢时产生更新事件 UEV，计数器更新中断标志位 ATIM_ISR.UIF 被硬件置位，如果允许中断（设置 ATIM_IER.UIE 为 1），将产生中断请求，设置 ATIM_ICR.UIF 为 0 清除该标志位。

以下是计数器在不同时钟频率下的操作示例，其中 ATIM_ARR =0x36、ATIM_RCR =0x00。

图 14-9　递减计数，内部时钟分频因子为 1

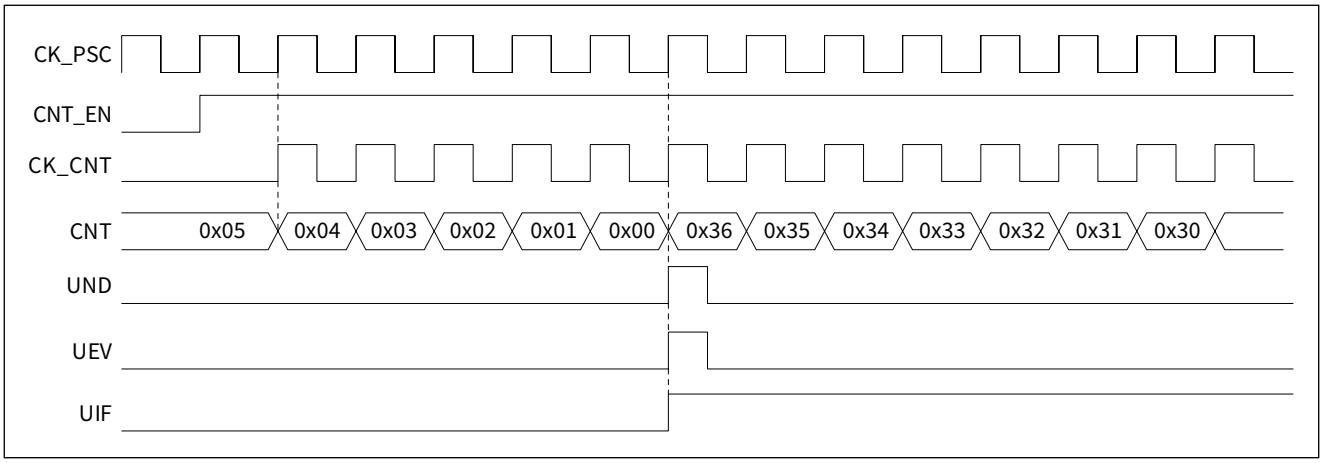


图 14-10　递减计数，内部时钟分频因子为 2

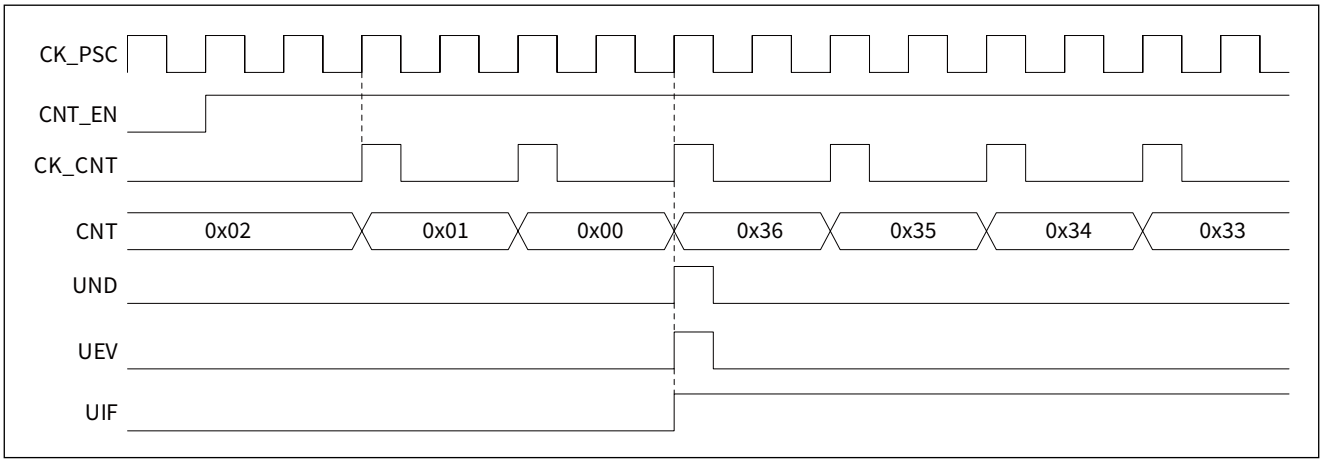


图 14-11　递减计数，内部时钟分频因子为 4

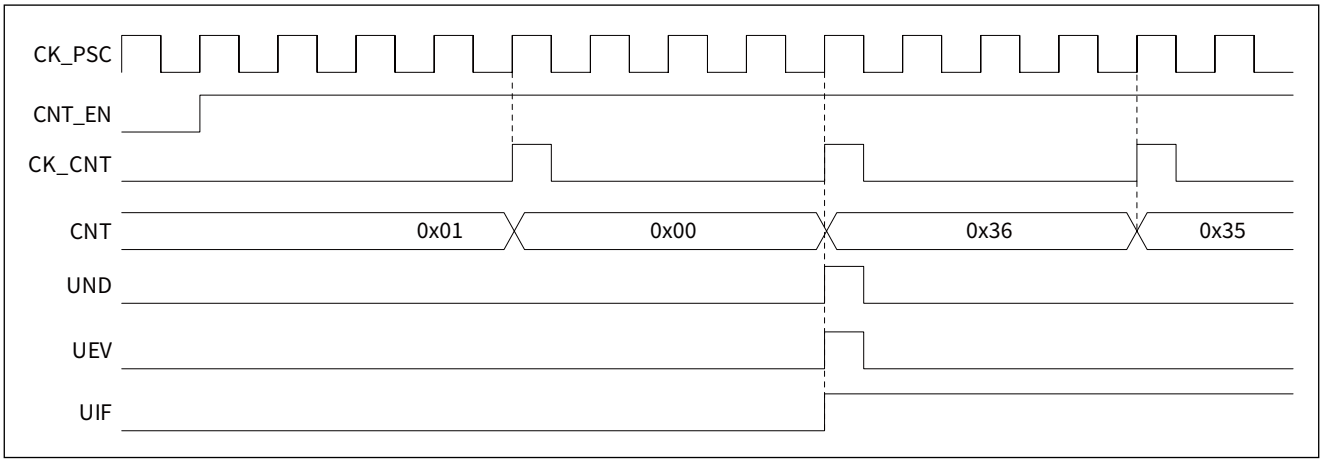
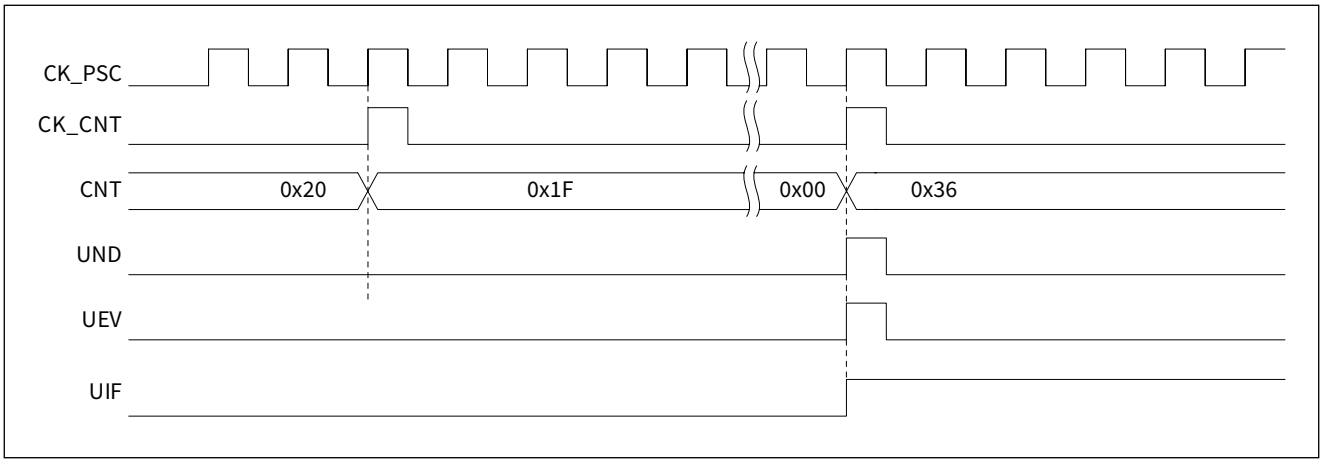


图 14-12　递减计数，内部时钟分频因子为 N



中心对齐模式（递增 / 递减计数）

在中心对齐模式下，计数器从 0 开始递增计数到 ARR-1，产生一个计数器上溢出事件，然后从 ARR 值开始递减计数到 1 并产生一个计数器下溢出事件，之后重新从 0 开始递增计数。

中心对齐模式包括中心对齐模式 1、中心对齐模式 2 和中心对齐模式 3，三种模式在计数方式上完全相同，只是输出比较中断标志的置位时机不同，具体请参见 14.9.1 ATIM_CR1 控制寄存器 1 的 CMS 位域说明。

在中心对齐模式下，控制寄存器 ATIM_CR1 的 DIR 位不能由软件写入，但可以读出，DIR 由硬件更新并指示当前的计数方向。启动中心对齐模式时，计数器将根据当前的 DIR 位域值进行递增或递减计数。不能同时通过软件修改 DIR 和 CMS 位域。

不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：

- 如果在计数器 CNT 中写入大于自动重载值 ARR 的值（CNT>ARR），则不会更新方向。例如，如果计数器之前是递增计数，则继续递增计数。
- 如果向计数器 CNT 写入 0 或 ARR 值，计数方向会更新，但不生成更新事件 UEV。

使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新事件 UEV（设置 ATIM_EGR.UG 为 1），并且不要在计数器运行过程中对其执行写操作。

以下是计数器在不同时钟频率下的操作示例，未使用重复计数器：

图 14-13　中心对齐模式，内部时钟分频因子为 1，ARR=0x06

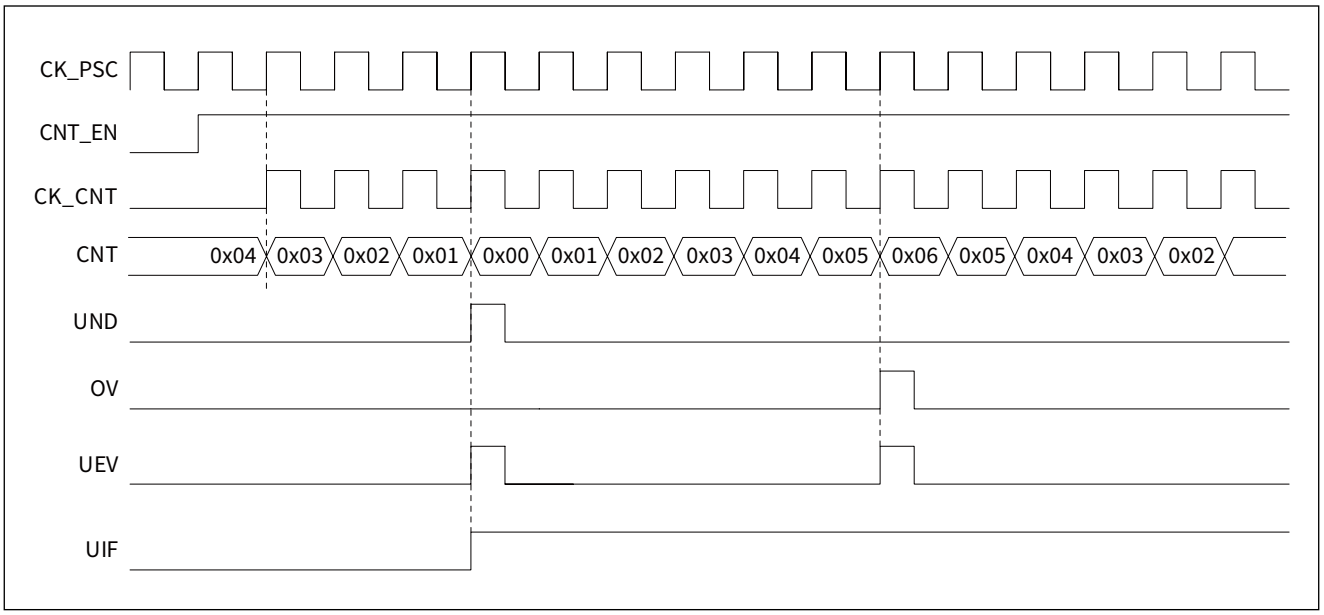


图 14-14　中心对齐模式，内部时钟分频因子为 2

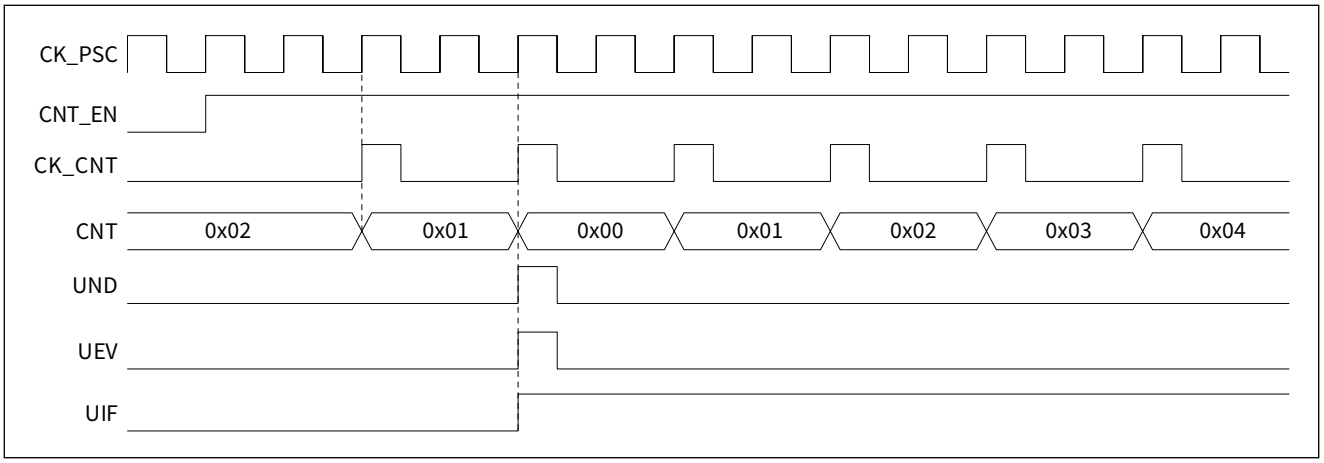


图 14-15　中心对齐模式，内部时钟分频因子为 4，ARR=0x36

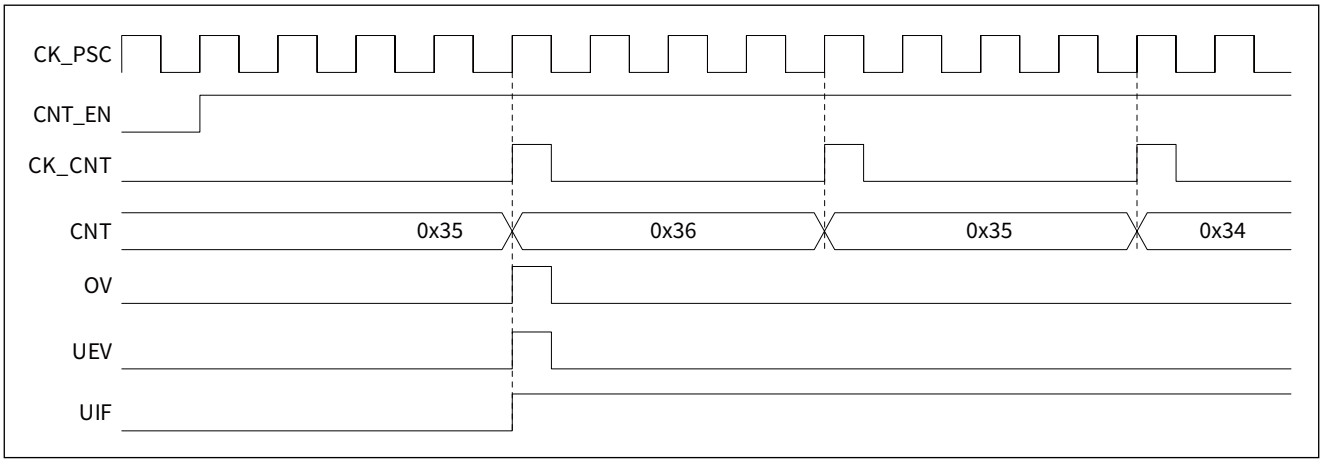


图 14-16　中心对齐模式，内部时钟分频因子为 N

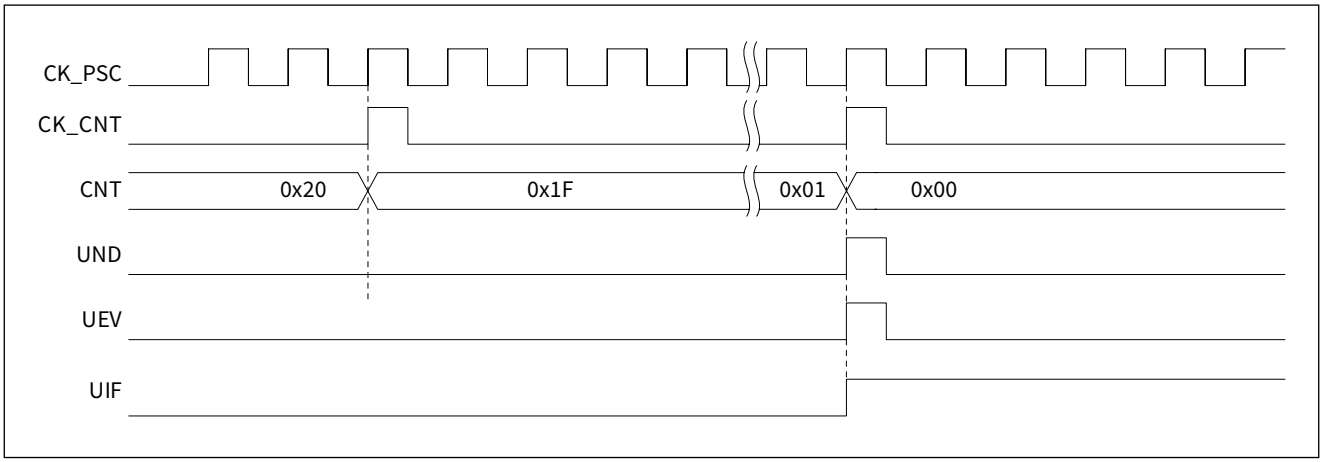


图 14-17　中心对齐模式，当重载缓存禁止时的更新事件（ATIM_CR1.ARPE=0）

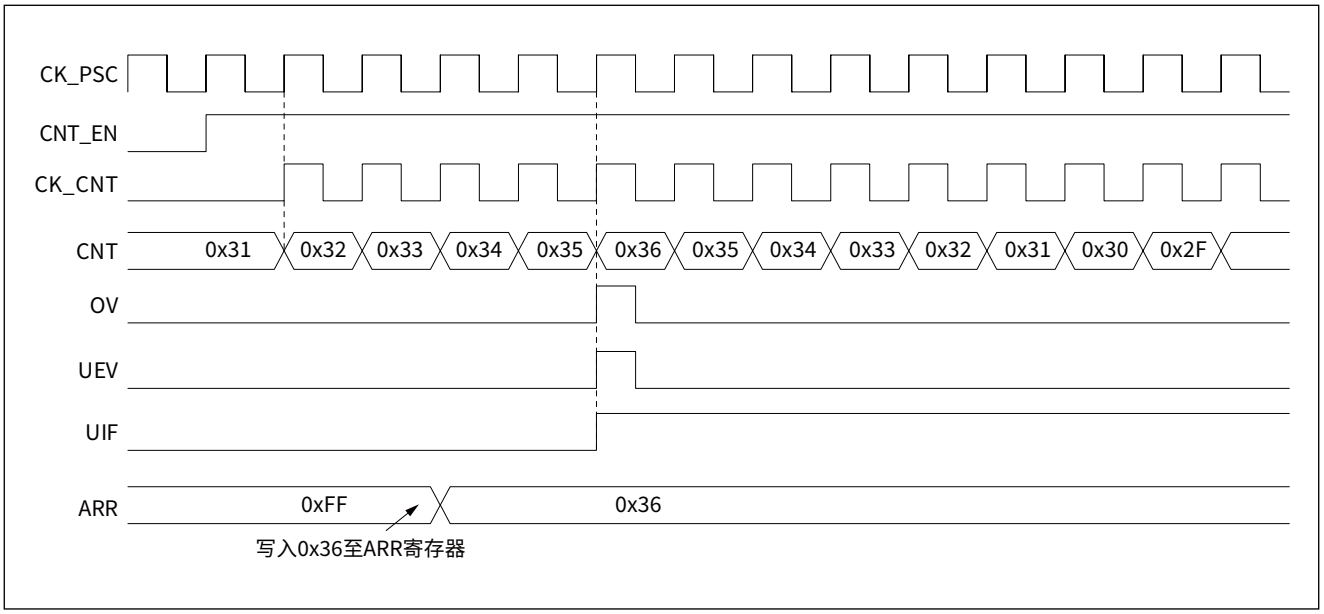
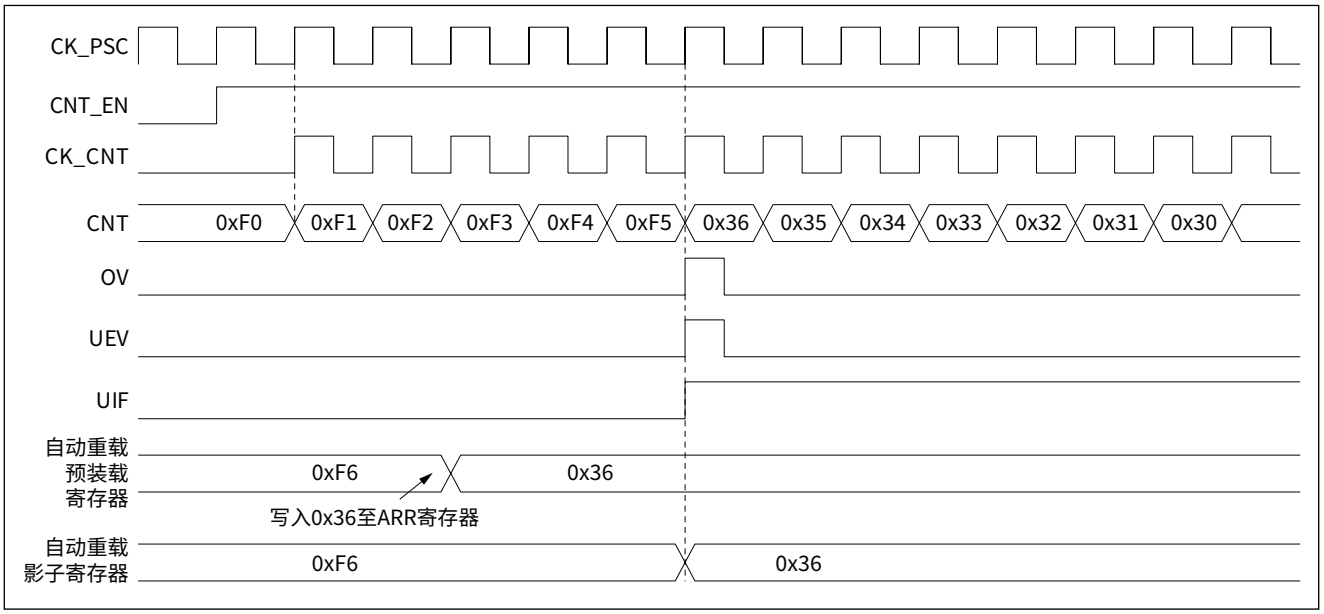


图 14-18　中心对齐模式，当重载缓存使能时的更新事件（ATIM_CR1.ARPE=1）



14.3.1.4　重载寄存器

自动重载寄存器 ATIM_ARR 具有缓存功能，通过控制寄存器 ATIM_CR1 的 ARPE 位域开启或关闭。

当计数器处于停止状态或是缓存功能关闭时，更新重载寄存器 ARR 将会立即更新其影子寄存器。当定时器处于运行状态且缓存功能有效时，修改重载寄存器 ARR 将不会立即更新影子寄存器，仅当生成更新事件 UEV 时才会将重载寄存器 ARR 的值更新到影子寄存器中。

14.3.1.5 重复计数器

启动 ATIM 会自动加载重复计数寄存器 ATIM_RCR 的 REP 值到重复计数器，在主计数器产生溢出时自动减 1。

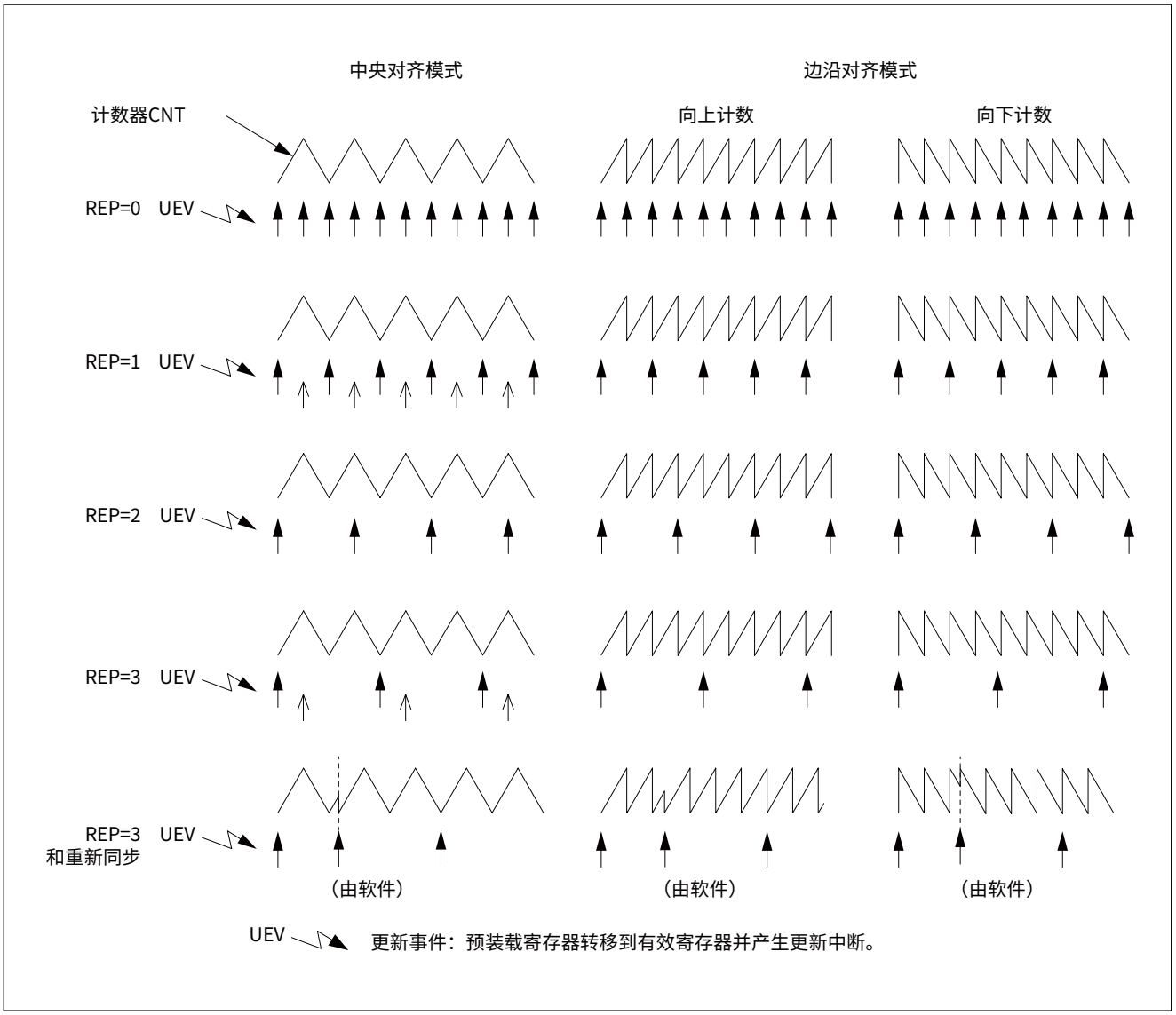
使用重复计数器功能（ATIM_RCR 的 REP 设置不为 0）时，只有当重复计数器减为 0 时，主计数器的溢出才会触发更新事件 UEV。

通过软件更新 UG 或从模式复位时，会立即产生更新事件 UEV，不关心当前重复计数器的值，并且，ATIM_RCR 寄存器中 REP 的内容将立即加载到重复计数器。

在中心对齐模式下，如果重复值 REP 为奇数，根据写入 ATIM_RCR 寄存器以及启动计数器的时机，更新事件将在上溢或下溢时发生：如果在启动计数器前写入 RCR 寄存器，则更新事件 UEV 在上溢时发生；如果在启动计数器后写入 RCR 寄存器，则更新事件 UEV 在下溢时发生。

下图是不同重复计数值，在不同计数模式下产生更新事件 UEV 的示例：

图 14-19 重复计数示例



14.3.1.6 更新事件 UEV

允许通过控制寄存器 ATIM_CR1 的 UDIS 位域来禁止或使能更新。

使能 UEV

设置 UDIS 位域为 0 使能 UEV，此时根据 URS 位域可选择更新请求源，如下表所示：

表 14-2 更新源设置

ATIM_CR1.URS	更新源
0	计数器上溢出、下溢出； UG 置位； 通过从模式控制器生成的更新事件
1	计数器上溢出、下溢出

当发生更新事件时，将进行以下动作：

1. 重新初始化计数器：

   递减计数（边沿对齐模式）：重新加载自动重载值 ATIM_ARR；

   中心对齐模式或递增计数（边沿对齐模式）：计数器清零。

2. 如果使能了重复计数器，重复计数器中将重新装载 ATIM_RCR 寄存器的内容。
3. 如果使用了缓存寄存器功能，将更新对应的预装载寄存器到其影子寄存器。
4. 预分频器的计数器被清零，ATIM_PSC 中新的预分频值生效。
5. 事件更新中断标志位 ATIM_ISR.UIF 被硬件置位。

当发生一个更新事件 UEV 时，事件更新中断标志位 ATIM_ISR.UIF 会被硬件置位，如果允许中断（设置 ATIM_IER.UIE 为 1），将产生一个更新中断请求，设置 ATIM_ICR.UIF 为 0 可清除该标志位。

注意：

如果使用了重复计数器（ATIM_RCR.REP 不为 0），只有在计数次数达到重复计数次数（ATIM_RCR.REP 达到 0）时，才会产生更新事件 UEV。

禁止 UEV

设置 UDIS 位域为 1 禁止 UEV，不再生成任何更新事件。

如果使用了缓存寄存器功能，对应的各影子寄存器的值保持不变。但如果 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器的计数器。

14.3.1.7 单脉冲模式

计数器可工作在单次计数或连续计数模式下，通过控制寄存器 ATIM_CR1 的 OPM 位域来选择。

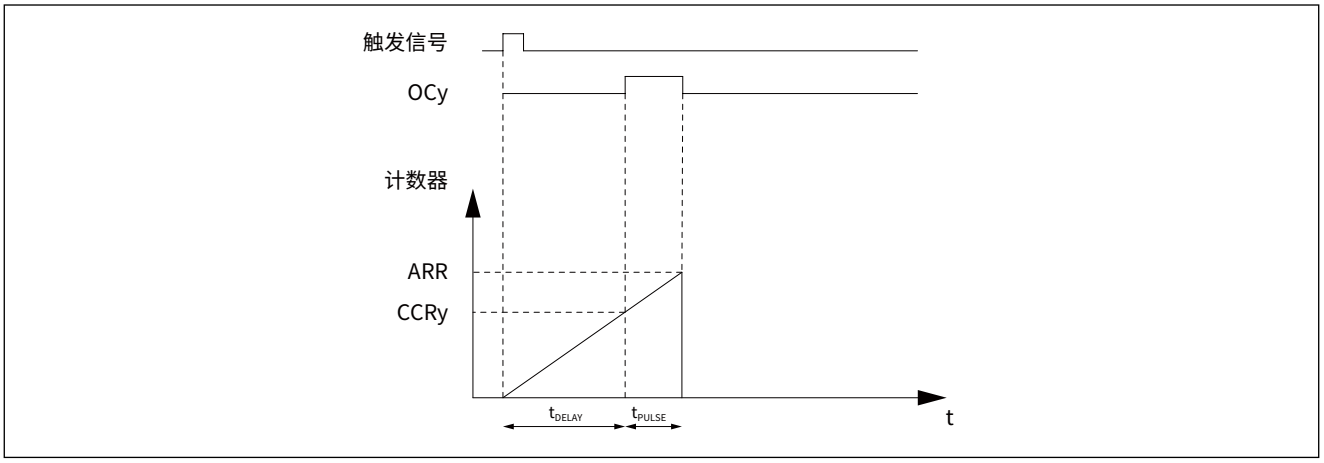
单次计数模式

设置 ATIM_CR1.OPM 为 1，使定时器工作在单次计数模式下。

启动计数器后，计数器 CNT 在 CK_CNT 时钟的驱动下计数，在发生下一更新事件时，计数器停止计数，同时 ATIM_CR1.CEN 被硬件自动清零。

可以通过从模式控制器启动计数器，配合定时器的输出比较模式，使得计数器在一个触发信号的触发下启动，并在一段可编程的延时后产生一个脉宽可编程的单脉冲。如下图所示，定时器在检测到触发信号的有效边沿时开始计数，延迟 t_{DELAY} 之后，在比较输出端口上产生一个宽度为 t_{PULSE} 的正脉冲。其中 t_{DELAY} 由计数寄存器 ATIM_CNT 的初值和捕获 / 比较寄存器 ATIM_CCRy 的差值确定，t_{PULSE} 由自动重载寄存器 ATIM_ARR 和捕获 / 比较寄存器 ATIM_CCRy 的差值来确定。

图 14-20 单脉冲输出示例（递增计数模式）



采样触发输入到激活 OC 输出需要多个时钟周期，可设置比较模式寄存器 ATIM_CCMRxCMP 的 OCyFE 位域为 1，以缩短 OC 输出延迟时间。

连续计数模式

设置 ATIM_CR1.OPM 为 0，使定时器工作在连续计数模式下，计数器在发生更新事件时不会停止计数。

14.3.1.8 外部触发输入通道

外部触发输入信号 ETR 可用作从模式控制器的触发输入 (TRGI)，也可用作计数器的计数时钟，具体请参见 14.3.2 工作模式。可对 ETR 信号进行输入控制，包括极性选择和边沿检测、预分频和滤波。

极性选择和边沿检测

从模式控制寄存器 ATIM_SMCR 的 ETP 位域用于选择 ETR 输入信号的触发极性。当设置 ETP 为 0 时，ETR 不反相，高电平或上升沿有效；当 ETP 设置为 1 时，ETR 反相，低电平或下降沿有效。

预分频器

外部触发信号 ETRP 频率不得超过 PCLK 频率的 1/4。当 ETRP 的频率较高时，用户应通过适当的 ETPS 预分频器设置对外部信号进行分频，以降低 ETRP 频率，可设置分频系数为 1、2、4、8。

滤波器

滤波器采用数字滤波方式，以一定频率对输入信号进行采样，当连续采样到 N 个相同电平时信号有效，否则信号无效，以此滤除高频杂波信号。

ETR 输入信号的滤波器由从模式控制寄存器 ATIM_SMCR 的 ETF 位域进行配置，可设置采样频率和采样点的个数。采样频率 $f_{SAMPLING}$ 由 f_{DTS} 分频后的时钟提供，f_{DTS} 是 f_{PCLK} 分频后得到的频率，分频因子由 ATIM_CR1 寄存器的 CKD 位域配置，可设置 1、2、4 分频。

14.3.1.9 输入捕获通道

ATIM 支持 6 个输入捕获通道 TI1/2/3/4/5/6，可用作捕获命令。其中 TI1 和 TI2 还可用作从模式控制器的触发输入 (TRGI) 和编码器接口输入，具体请参见 14.3.2 工作模式。

支持对 TIy 信号进行输入控制，包括滤波、边沿检测和预分频。

滤波器

滤波阶段以一定频率对相应的 TIy 输入信号进行采样，生成滤波后信号 TIyF。

TIy 输入信号的滤波器由捕获模式寄存器 ATIM_CCMRxCAP 的 ICyF 位域进行配置，可设置采样频率和采样点的个数。采样频率 $f_{SAMPLING}$ 由 f_{PCLK} 或 f_{DTS} 分频后的时钟提供，f_{DTS} 是 f_{PCLK} 分频后得到的频率，分频因子由 ATIM_CR1 寄存器的 CKD 位域配置，可设置 1、2、4 分频。

边沿检测

带有极性选择功能的边沿检测器可生成一个 TIxFPx 信号，具体有效极性请参见 14.9.14 ATIM_CCER 捕获 / 比较使能寄存器的 CCyNP 和 CCyP 位域。

预分频器

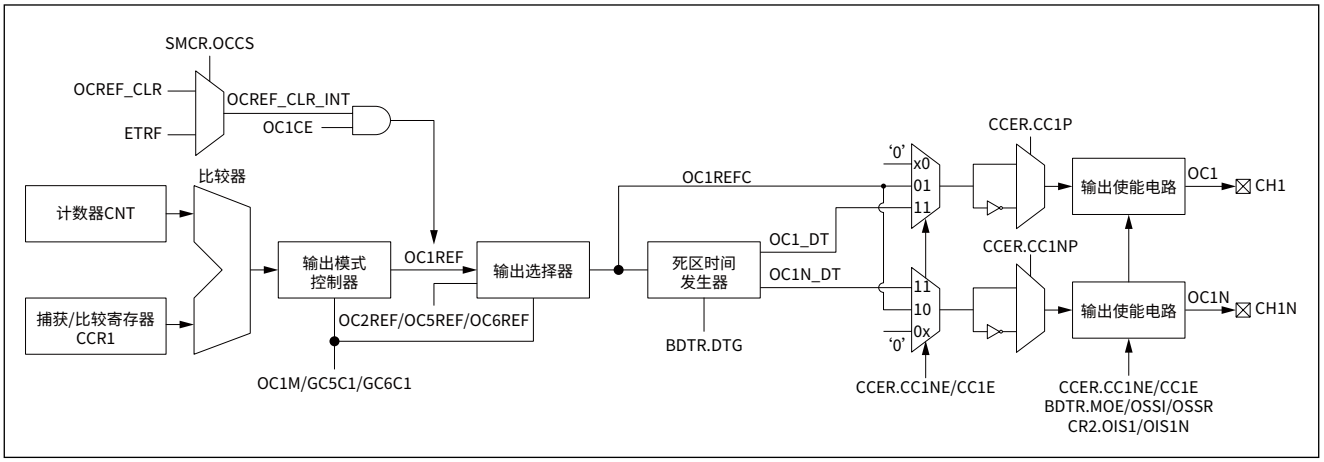
TIy 信号用于输入捕获时，可对捕获通道信号 ICy 进行分频，通过捕获模式寄存器 ATIM_CCMRxCAP 的 ICyPSC 位域进行控制，支持 1、2、4、8 分频。

14.3.1.10 输出比较通道

ATIM 支持 6 个独立输出比较通道 CH1/2/3/4/5/6（所有通道均支持互补输出 CHyN），每个通道均由一个捕获 / 比较寄存器（包括一个影子寄存器）和一个输出阶段（比较器和输出控制）组成。

输出控制单元用于比较匹配时，控制输出端口的波形，可配置输出比较模式、输出极性选择和输出使能等。以比较通道 1 为例，其框图如下图所示：

图 14-21　输出比较 1 通道



14.3.2 工作模式

ATIM 支持多种工作模式，具体由从模式控制寄存器 ATIM_SMCR 的 SMS 位域来配置，如下表所示：

表 14-3 ATIM 工作模式

ATIM_SMCR.SMS	从模式选择
0000	禁止从模式，使用内部时钟
0111	外部时钟模式 1
0100	复位模式
0101	门控模式
0110	触发模式
1000	组合复位 + 触发模式
1001	组合门控 + 复位模式
1110	正交编码器模式，x1 模式
1111	正交编码器模式，x1 模式
0001	正交编码器模式，x2 模式
0010	正交编码器模式，x2 模式
0011	正交编码器模式，x4 模式
1010	编码模式（时钟 + 方向），x2 模式
1011	编码模式（时钟 + 方向），x1 模式
1100	编码模式（带方向时钟），x2 模式
1101	编码模式（带方向时钟），x1 模式

注：

1. 从模式选择位包括 ATIM_SMCR[16] 和 ATIM_SMCR[2:0]，需组合配置。

2. 从模式选择位 SMS[3:0] 支持预装载功能，请参见 SMSPE 和 SMSPS 位域说明。

14.3.2.1 内部时钟模式

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x0 时，禁止定时器从模式，预分频器时钟直接由内部时钟 PCLK 提供。设置 ATIM_CR1.CEN 为 1，将使能计数器开始计数。

14.3.2.2 外部时钟模式

外部时钟模式 1

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x7 时，由所选触发信号（TRGI）的上升沿提供计数器时钟。TRGI 信号有多种触发选择，具体通过 ATIM_SMCR 寄存器的 TS 位域进行选择，如下表所示：

表 14-4 TRGI 信号来源

ATIM_SMCR.TS 位域值	TRGI 信号来源
00111	外部触发输入（ETRF）
000xx/01xxx/10000	内部触发（ITR0~12）
00100	TI1 边沿检测器 (TI1F_ED)
00101	滤波后的定时器输入 1（TI1FP1）
00110	滤波后的定时器输入 2（TI2FP2）

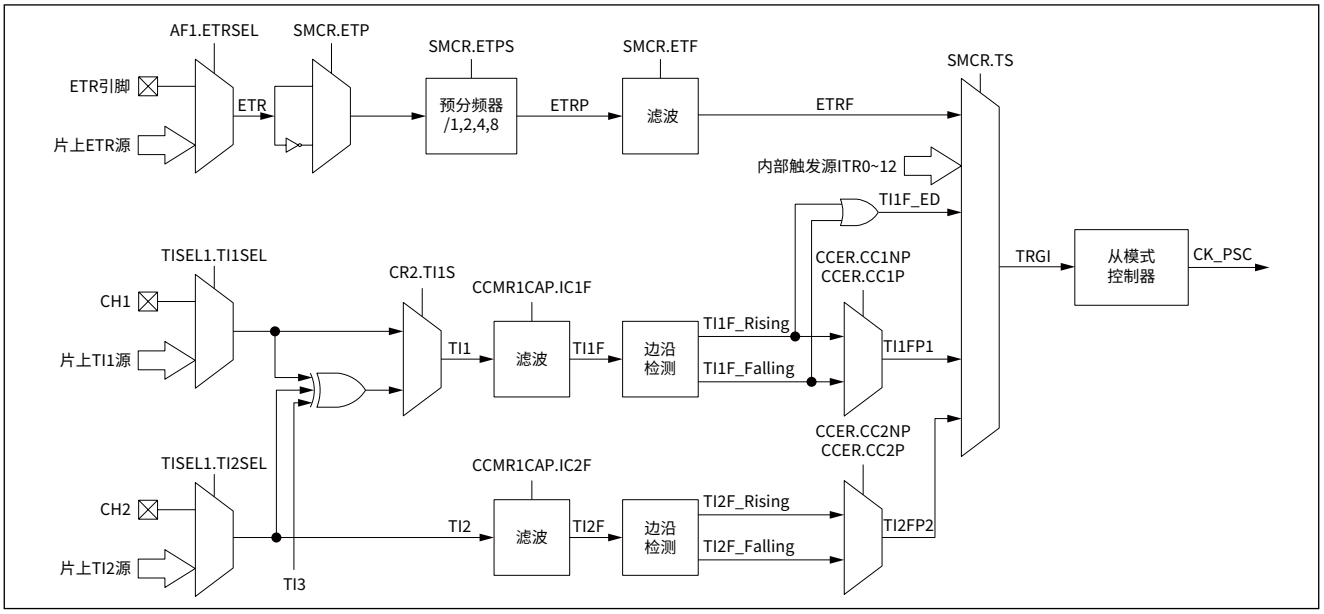
ETR 输入信号来源可以是外部 ATIM_ETR/GTIM_ETR 引脚，也可以是片内其它外设，请参见 14.3.7 片内外设互联 ETR。选择 ATIM_ETR 为 TRGI 信号源时，可通过 ATIM_SMCR.ETP 选择外部触发极性，通过 ATIM_SMCR.ETPS 设置预分频，通过 ATIM_SMCR.ETF 进行滤波控制。

内部触发 ITR 来源为 BTIM 和 GTIM 的触发输出信号 TRGO 以及 UART 的 TXD/RXD 信号，请参见 14.3.6 定时器级联 ITR。

TI1 和 TI2 都具有滤波和边沿检测功能，TI1 和 TI2 分别通过捕获模式寄存器 ATIM_CCMR1CAP 的 IC1F 和 IC2F 位域进行滤波控制，分别通过捕获 / 比较使能寄存器 ATIM_CCER 的 CC1P/CC1NP 和 CC2P/CC2NP 位域进行边沿检测。

外部时钟模式 1 连接示意图如下图所示：

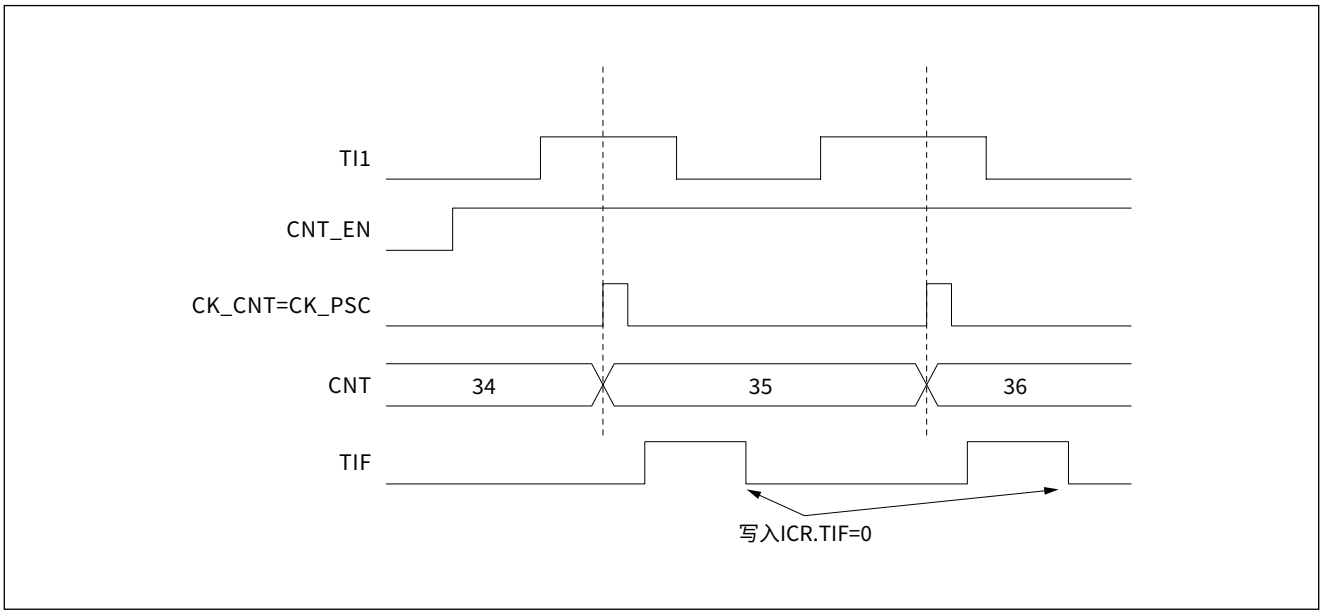
图 14-22 外部时钟模式 1 连接图



当 TRGI 出现有效边沿时，ATIM_ISR.TIF 标志将置 1，向 ATIM_ICR.TIF 写 0 可清除该标志。

下图所示为配置 TI1 上升沿的外部时钟模式 1 时序图，TI1 的上升沿与实际计数器时钟之间的延迟是由于 TI1 输入的重新同步电路引起的。

图 14-23　外部时钟模式 1 时序示例



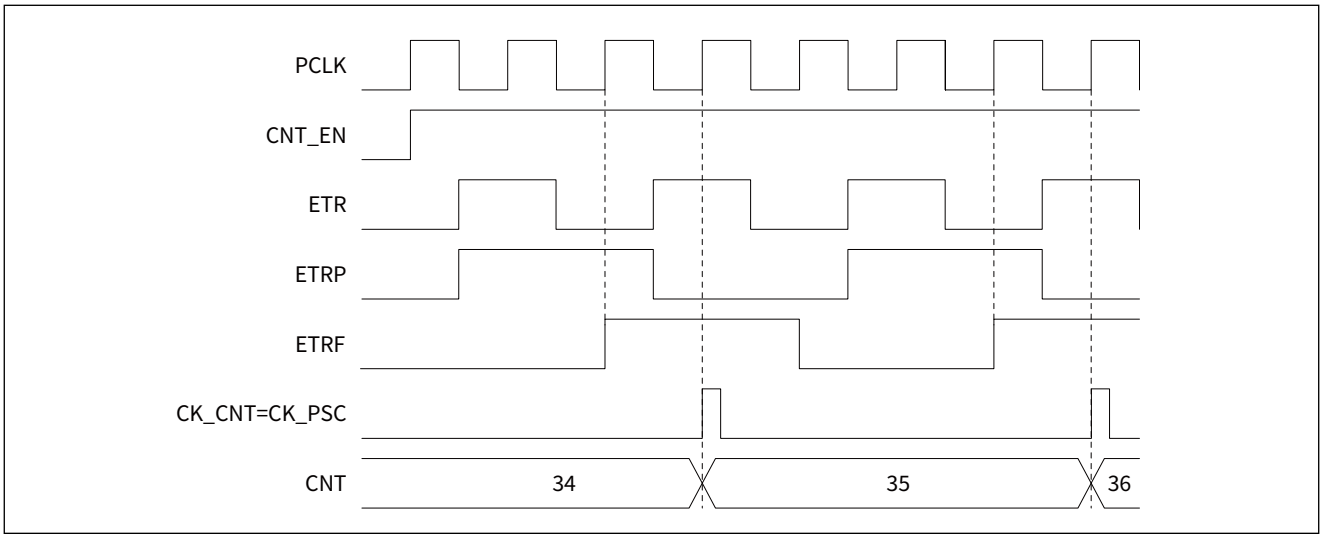
外部时钟模式 2

设置从模式控制寄存器 ATIM_SMCR 的 ECE 位域为 1，使能外部时钟模式 2。该模式下，计数器时钟由 ETRF 信号的任意有效边沿提供，与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=0111 且 TS=00111）具有相同效果，连接框图可参见图 14-22 外部时钟模式 1 连接图。

外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式，此时从模式下的 TRGI 不得连接 ETRF（即 TS 位域不能设置为 0x7）。如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。外部时钟模式 2 不能与编码器模式同时使用。

下图所示为 ETR 设置为 2 分频时的外部时钟模式 2 的时序示例，ETR 的上升沿与实际计数器时钟之间的延迟是由 ETRP 信号的重新同步电路引起的，因此计数器可以正确捕获的最大频率最高为内部时钟 PCLK 频率的 1/4，当 ETRP 信号变快时，用户应对外部信号进行适当的分频。

图 14-24　外部时钟模式 2 时序示例



14.3.2.3 复位模式

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x4 时，ATIM 配置为复位模式，计数器的复位由 TRGI 信号控制。

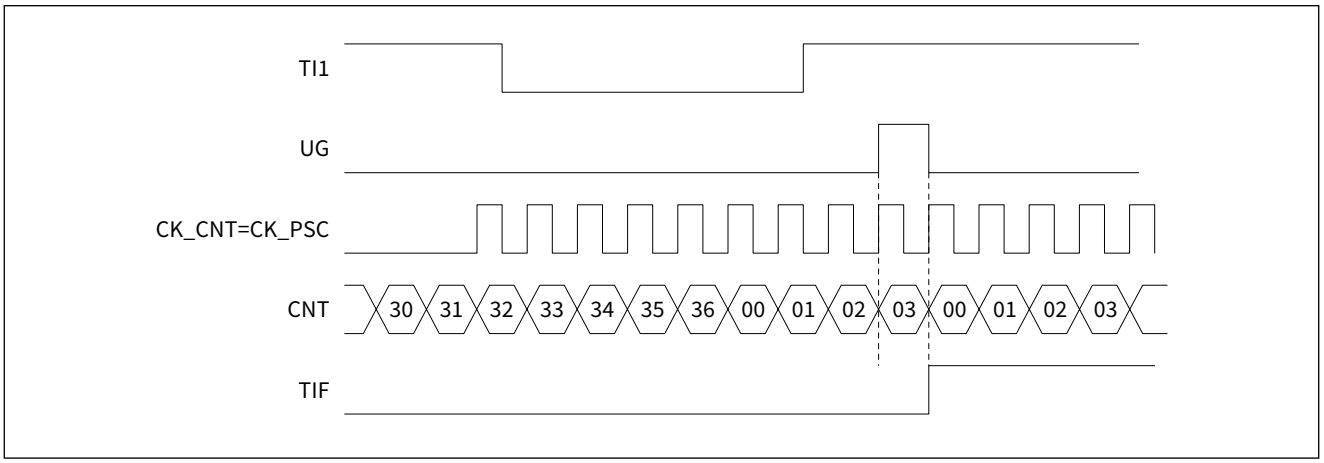
TRGI 信号的来源由从模式控制寄存器 ATIM_SMCR 的 TS 位域控制，可参见表 14-4 TRGI 信号来源。

当检测到有效的触发输入（TRGI）时，将产生以下影响：

1. 重新初始化计数器和预分频器的计数器；
2. 如果控制寄存器 ATIM_CR1 的 URS 位域为 0，则会产生更新事件 UEV，事件更新中断标志 ATIM_ISR.UIF 置 1，可产生中断请求；
3. 触发中断标志 ATIM_ISR.TIF 置 1，可产生中断请求。

下图所示为复位模式时序图示例。设置 ATIM_CR1.CEN 为 1 使能计数器，计数器根据计数时钟 CK_CNT 正常计数，当 TI1 出现上升沿时，计数器清零并重新从 0 开始计数，同时 ATIM_ISR.TIF 标志位置 1。TI1 的上升沿与实际计数器复位之间的延迟是由 TI1 输入的重新同步电路引起的。

图 14-25 复位模式时序



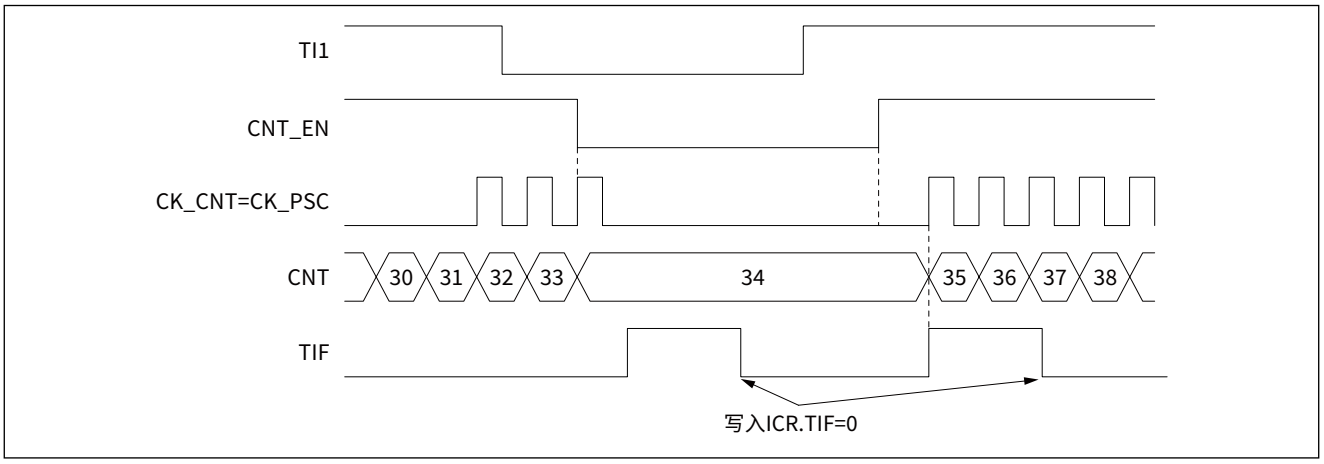
14.3.2.4 门控模式

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x5 时，ATIM 配置为门控模式。在该模式下，触发输入 (TRGI) 为高电平且 ATIM_CR1.CEN 为 1 时，启动计数器计数；触发输入 (TRGI) 为低电平或 ATIM_CR1.CEN 为 0 时，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。

TRGI 信号的来源由从模式控制寄存器 ATIM_SMCR 的 TS 位域控制，可参见表 14-4 TRGI 信号来源，但需注意门控模式下不能选择 TI1F_ED 作为触发输入。

下图所示为门控模式时序图示例。当 TI1 为高电平时，计数器启动计数；当 TI1 为低电平时，计数器暂停计数。计数器启动和停止时，ATIM_ISR.TIF 标志位都会置 1。TI1 的边沿与实际计数器使能信号 CNT_EN 之间的延迟是由 TI1 输入的重新同步电路引起的。

图 14-26 门控模式时序



14.3.2.5 触发模式

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x6 时，ATIM 配置为触发模式。在该模式下，设置 ATIM_CR1.CEN 为 1 或触发信号 TRGI 出现上升沿时，触发启动计数器计数（但不复位）。

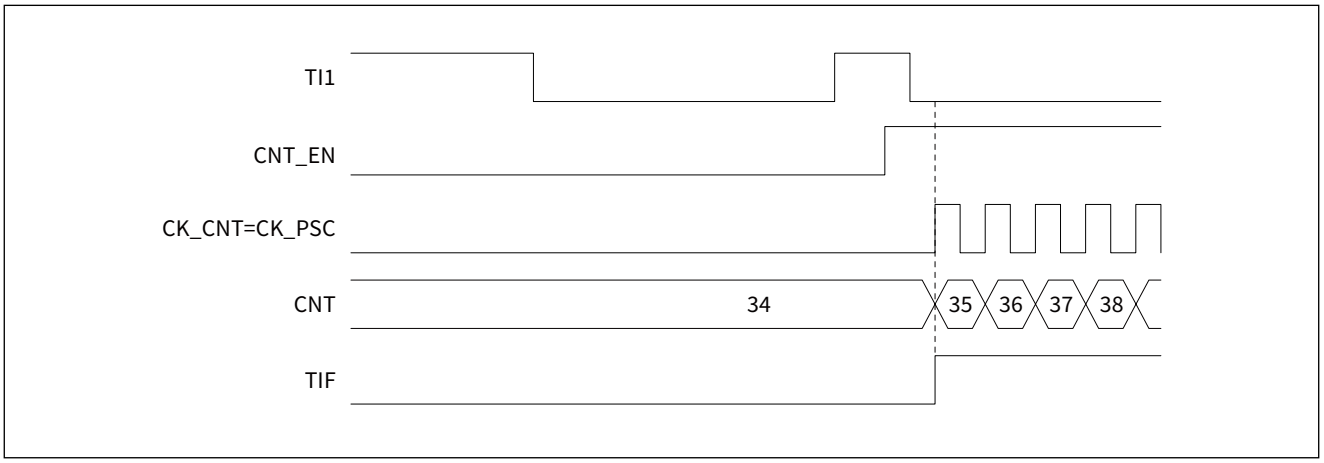
TRGI 信号的来源由从模式控制寄存器 ATIM_SMCR 的 TS 位域控制，可参见表 14-4 TRGI 信号来源。

当检测到有效的触发信号时，将产生以下影响：

1. ATIM_CR1.CEN 被硬件置位；
2. 触发中断标志位 ATIM_ISR.TIF 置 1，可产生中断请求；
3. 计数器启动，开始计数。

下图所示为触发模式时序图示例。当 TI1 出现上升沿时，计数器启动计数，同时 ATIM_ISR.TIF 标志位置 1。TI1 的上升沿与实际计数器启动之间的延迟是由 TI1 输入的重新同步电路引起的。

图 14-27 触发模式时序



14.3.2.6 组合复位 + 触发模式

当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x8 时，ATIM 配置为组合复位 + 触发模式。在该模式下，出现所选触发输入 (TRGI) 上升沿时，将重新初始化计数器并启动计数器，同时触发中断标志 ATIM_ISR.TIF 置 1。如果控制寄存器 ATIM_CR1 的 URS 位域为 0，则会产生更新事件 UEV，事件更新中断标志 ATIM_ISR.UIF 会被硬件置位。

TRGI 信号的来源由从模式控制寄存器 ATIM_SMCR 的 TS 位域控制，可参见表 14-4 TRGI 信号来源。

14.3.2.7 组合门控 + 复位模式

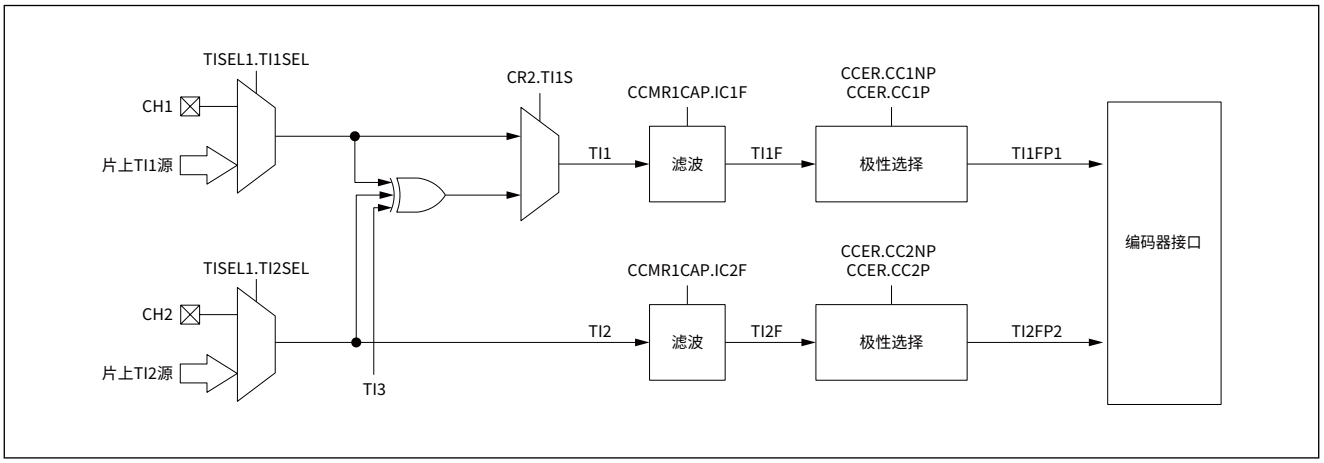
当从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0x9 时，ATIM 配置为组合门控 + 复位模式。在该模式下，计数器的启动和停止都被控制，触发输入 (TRGI) 为高电平且 ATIM_CR1.CEN 为 1 时，启动计数器计数；触发输入 (TRGI) 为低电平时，计数器立即停止计数，并被复位。计数器启动和停止时，ATIM_ISR.TIF 标志位都会置 1；计数器复位时，ATIM_ISR.UIF 标志位置 1。

TRGI 信号的来源由从模式控制寄存器 ATIM_SMCR 的 TS 位域控制，可参见表 14-4 TRGI 信号来源。

14.3.2.8 正交编码器模式

ATIM 支持正交编码器模式，用于接收并解码正交编码器的信号。在该模式下，允许通过 CH1、CH2 引脚与外部的正交编码器直接连接，根据输入信号的跳变顺序，实现计数器自动递增或递减计数，计数器值始终表示编码器的位置。其功能框图如下图所示：

图 14-28 编码器模式框图



CH1 和 CH2 输入信号具有滤波和极性选择功能，分别通过 ATIM_CCMR1CAP 寄存器的 IC1F 和 IC2F 位域进行滤波控制，通过 ATIM_CCER 寄存器的 CC1P 和 CC2P 位域选择输入极性，CC1NP 和 CC2NP 位域必须保持清零。

设置 ATIM_CR1.CEN 为 1 使能计数器，计数器将由通道 CH1 和 CH2 引脚输入信号经滤波和极性选择后的信号 TI1FP1 和 TI2FP2 的每次有效跳变驱动，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，参见表 14-5 计数方向和编码器信号的关系。编码器当前计数方向标志 ATIM_CR1.DIR 由硬件自动设置和清除，且在任何输入（CH1 或 CH2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 TI1FP1 或 TI2FP2 边沿处计数，还是同时在 TI1FP1 和 TI2FP2 处计数。

ATIM 支持多种正交编码计数模式，通过从模式控制寄存器 ATIM_SMCR 的 SMS 位域进行设置，不同模式下计数方向和编码器信号的关系如下表所示：

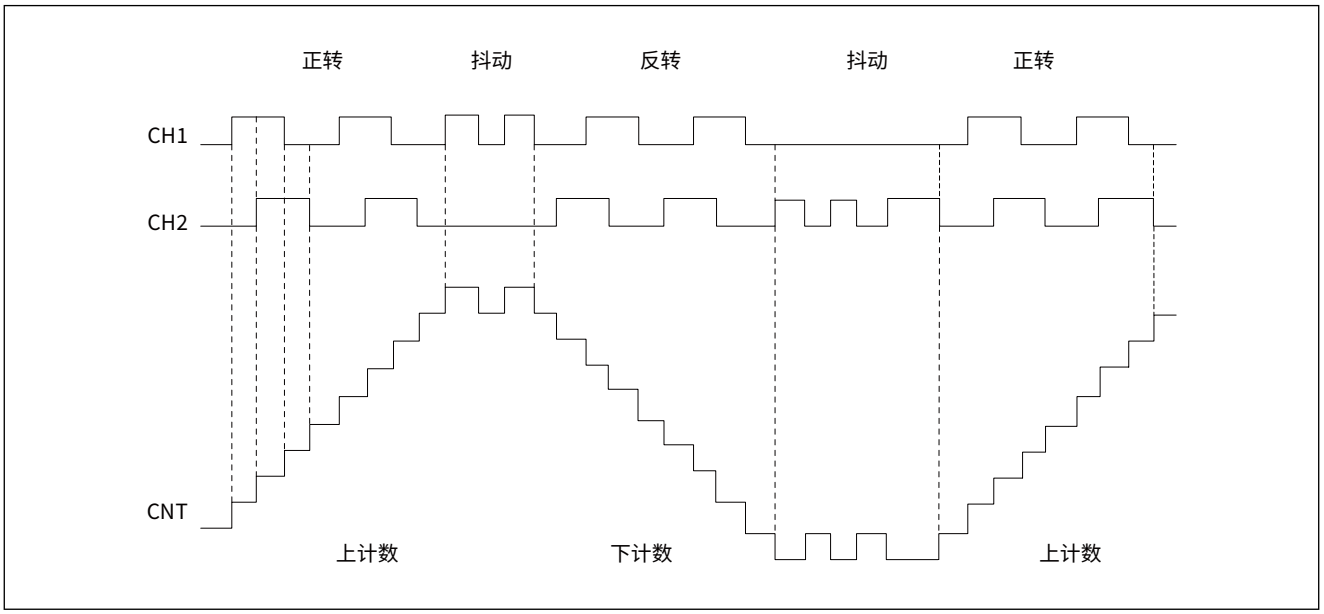
表 14-5 计数方向和编码器信号的关系

模式	SMS	信号的电平		TI1FP1		TI2FP2	
		TI2FP2	TI1FP1	上升	下降	上升	下降
在 TI1FP1 边沿计数（x1 模式）	1110	高	-	向下计数	向上计数	不计数	不计数
		低	-	不计数	不计数	不计数	不计数
在 TI2FP2 边沿计数（x1 模式）	1111	-	高	不计数	不计数	向上计数	向下计数
		-	低	不计数	不计数	不计数	不计数
在 TI1FP1 边沿计数（x2 模式）	0001	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
在 TI2FP2 边沿计数（x2 模式）	0010	-	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 TI1FP1 和 TI2FP2 边沿计数（x4 模式）	0011	高	高	向下计数	向上计数	向上计数	向下计数
		低	低	向上计数	向下计数	向下计数	向上计数

编码器输出的第三个信号表示机械零点，可以把它连接到外部触发输入引脚上，用以触发计数器复位。

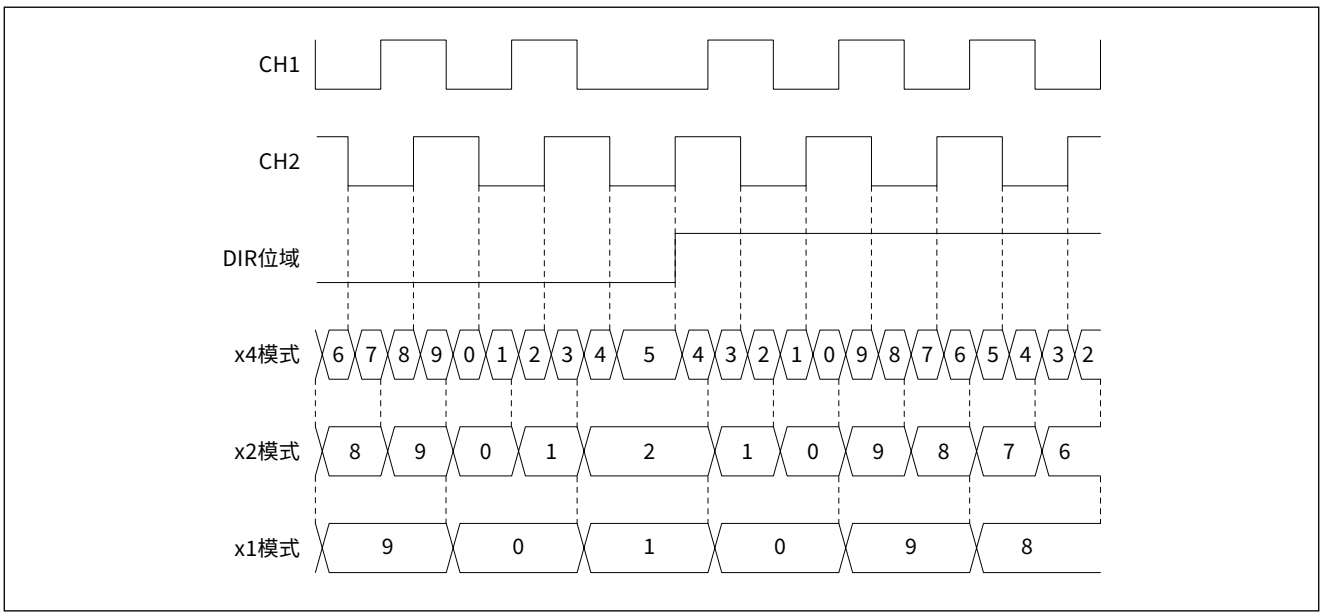
下图是一个正交编码计数模式 -x4 模式的操作实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。

图 14-29　正交编码器模式 -x4 模式操作示例



下图显示了各种正交编码计数模式下编码器反转期间的定时器计数值，示例中 CH1 和 CH2 输入均不反相。

图 14-30　各正交编码器模式计数示例



编码器模式可以提供传感器当前位置的相关信息。计数器根据编码器输出的脉冲自动进行递增或递减计数，当前计数器值即表示编码器当前位置。使用另一个配置为捕获模式的定时器，可以捕获编码器信号的上升沿或下降沿，并记录其时间戳，通过测量两个事件之间的时间间隔，可以计算出相应的动态信息，如速度、加速度和减速度；可以使用指示机械零位的编码器输出来实现此目的，通过测量从机械零位到其他位置的时间间隔，可以计算出相对于零位的位置和速度变化。

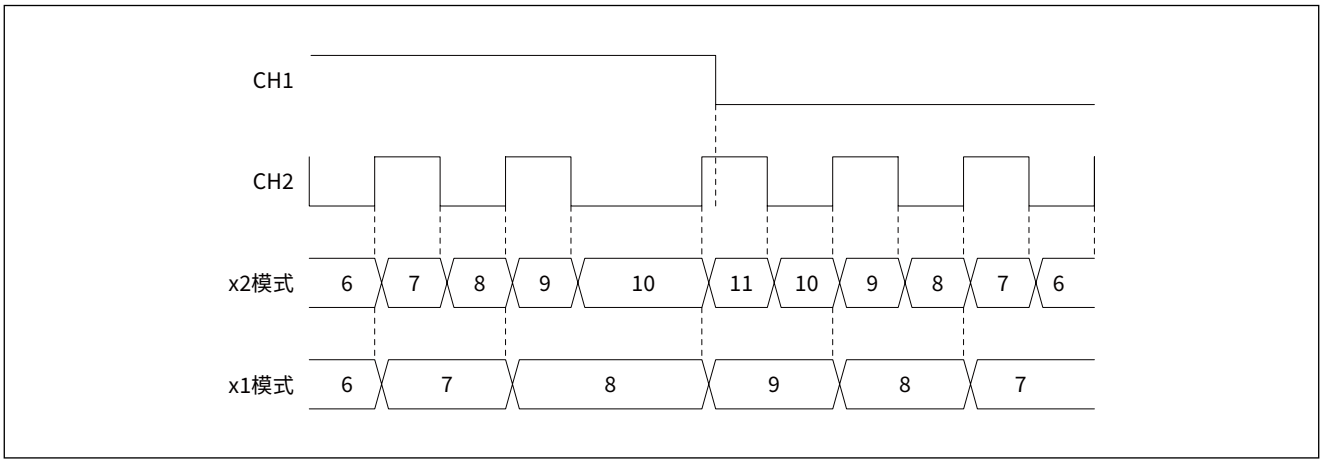
14.3.2.9 时钟加方向编码器模式

定时器还支持时钟加方向的编码器模式，需配置从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0xA 或 0xB，分别对应 x2 模式和 x1 模式。其功能框图可参见图 14-28 编码器模式框图，可对输入信号进行滤波和极性控制。

带方向的编码器通常具有两个输出信号：时钟信号和方向信号，使用时钟信号可以计算出旋转编码器的旋转速度和位置变化，而方向信号可以用来确定旋转的方向。在 x2 模式，计数器在时钟信号的上升沿和下降沿计数；在 x1 模式，计数器根据 CC2P 位域的值在单个时钟边沿计数，CC2P 为 0 对应上升沿敏感，CC2P 为 1 对应下降沿敏感。CH1 通道上方向信号的极性由 CC1P 位设置：CC1P 为 0 对应正极性（当 CH1 为高电平时递增计数，当 CH1 为低电平时递减计数），CC1P 为 1 对应负极性（当 CH1 为低电平时递增计数，当 CH1 为高电平时递减计数）。

下图显示了时钟加方向编码器模式下的计数器计数实例，示例中 CH1 和 CH2 输入均不反相。

图 14-31 时钟加方向编码器模式计数示例



14.3.2.10 定向时钟编码器模式

定时器还支持定向时钟编码器模式，需配置从模式控制寄存器 ATIM_SMCR 的 SMS 位域为 0xC 或 0xD，分别对应 x2 模式和 x1 模式。其功能框图可参见图 14-28 编码器模式框图，可对输入信号进行滤波和极性控制。

定向时钟编码器会根据旋转方向分别提供一条向上计数时钟线和向下计数时钟线。在 x2 模式，计数器在两个时钟线中的任意一个的上升沿和下降沿计数；CC1P 和 CC2P 位是时钟空闲状态的编码，CCyP 为 0 对应高电平空闲，CCyP 为 1 对应低电平空闲。在 x1 模式，计数器根据 CC1P 和 CC2P 位域的值在单个时钟边沿计数，CCyP 为 0 对应下降沿敏感和高电平空闲，CCyP 为 1 对应上升沿敏感和低电平空闲。

不同模式下计数方向和编码器信号的关系如下表所示：

表 14-6　计数方向和编码器信号的关系

模式	SMS	信号的电平		TI1FP1		TI2FP2	
		TI2FP2	TI1FP1	上升	下降	上升	下降
x2 模式 CCyP=0	1100	高	高	向下计数	向下计数	向上计数	向上计数
		低	低	不计数	不计数	不计数	不计数
x2 模式 CCyP=1		高	高	不计数	不计数	不计数	不计数
		低	低	向下计数	向下计数	向上计数	向上计数
x1 模式 CCyP=0	1101	高	高	不计数	向下计数	不计数	向上计数
		低	低	不计数	不计数	不计数	不计数
x1 模式 CCyP=1		高	高	不计数	不计数	不计数	不计数
		低	低	向下计数	不计数	向上计数	不计数

下图是定向时钟编码器模式的计数示例：

图 14-32　定向时钟编码器模式计数示例（CC1P = CC2P = 0）

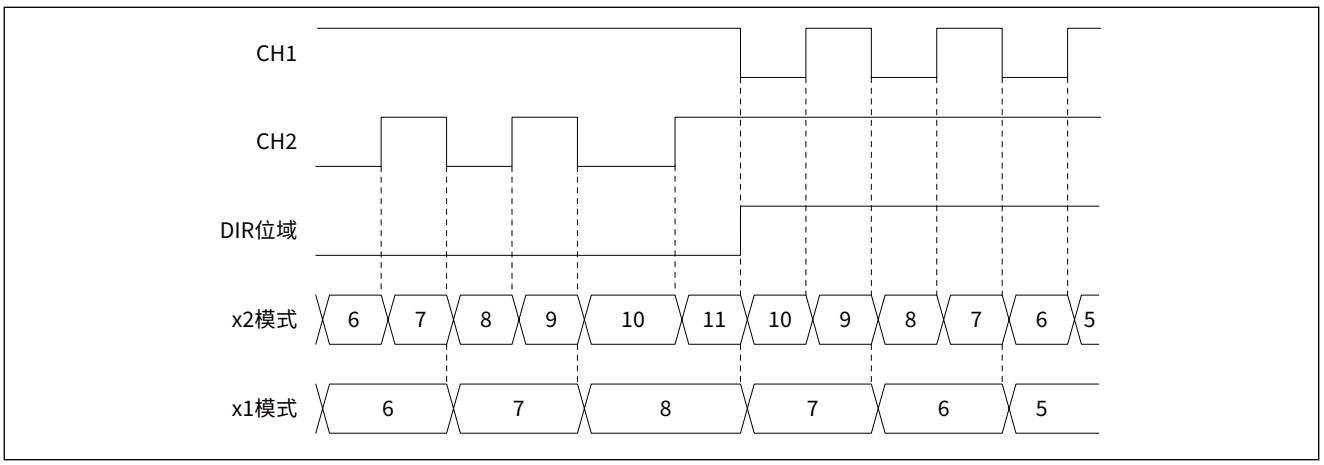
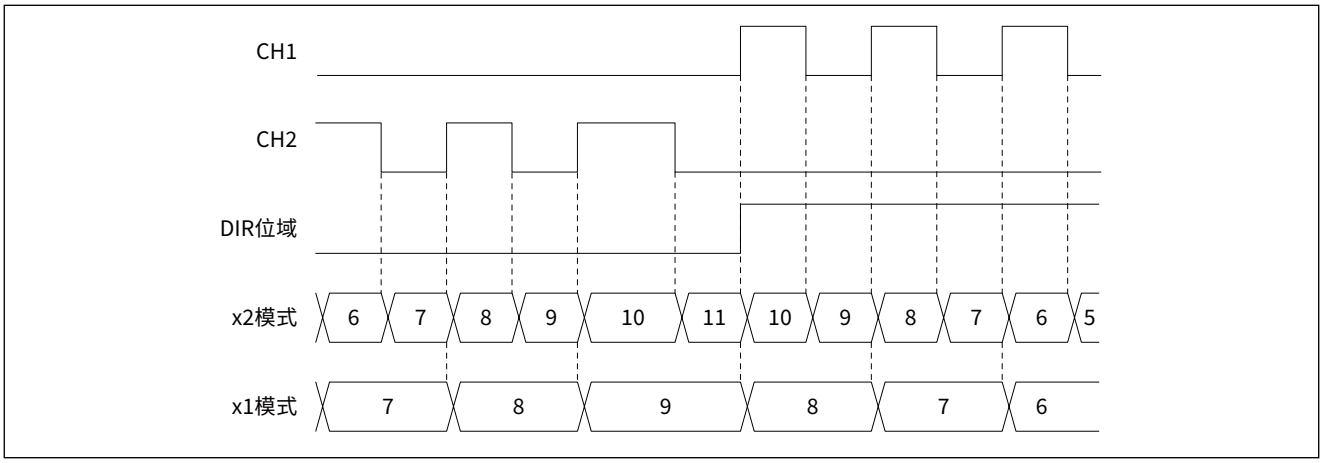


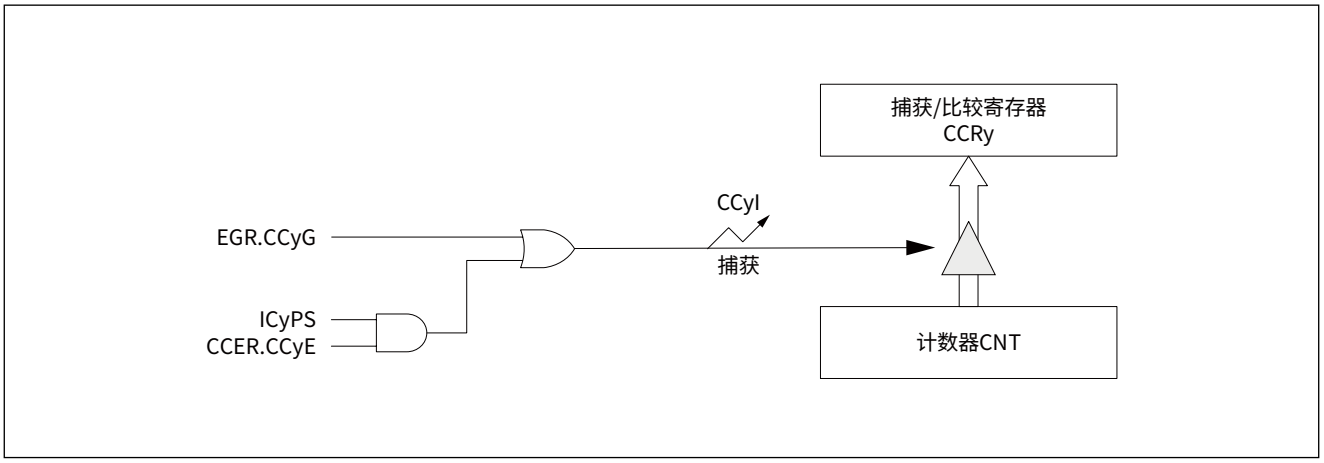
图 14-33　定向时钟编码器模式计数示例（CC1P = CC2P = 1）



14.3.3 输入捕获功能

ATIM 支持输入捕获功能，设置捕获模式寄存器 ATIM_CCMRxCAP 的 CCyS 位域为非零值时，CCy 通道配置为输入，同时指定对应捕获通道 ICy 的输入映射。支持通过软件或硬件触发输入捕获，其功能框图如下图所示：

图 14-34 输入捕获模式框图



软件触发：设置 ATIM_EGR.CCyG 为 1 时，立即软件触发一次捕获，当前计数器 CNT 的值被锁存到对应通道的捕获 / 比较寄存器 CCRy 中，完成一次捕获，捕获完成后 CCyG 位被硬件自动清零。

硬件触发：当检测到输入通道 TIy 上的有效边沿后，当前计数器 CNT 的值被锁存到对应通道的捕获 / 比较寄存器 CCRy 中，完成一次捕获。触发捕获的有效边沿通过捕获 / 比较使能寄存器 ATIM_CCER 的 CCyNP 和 CCyP 位域来配置，如下表所示：

表 14-7 输入捕获模式配置

ATIM_CCER 寄存器		捕获触发条件
CCyNP 位域	CCyP 位域	
0	0	上升沿捕获
0	1	下降沿捕获
1	1	上升沿和下降沿均捕获

当发生一次捕获时，对应通道的捕获 / 比较中断标志 ATIM_ISR.CCyIF 被硬件置 1，同时：

- 如果使能了中断（设置 ATIM_IER.CCyIE 为 1），将产生中断请求。
- 如果发生捕获事件时，ATIM_ISR.CCyIF 标志位已经为高，那么重复捕获标志位 ATIM_ISR.CCyOF 将被硬件置 1。
- 设置 ATIM_ICR.CCyIF 为 0 或读取捕获寄存器 CCRy 可清除 ATIM_ISR.CCyIF 标志位，设置 ATIM_ICR.CCyOF 为 0 可清除 ATIM_ISR. CCyOF 标志位。

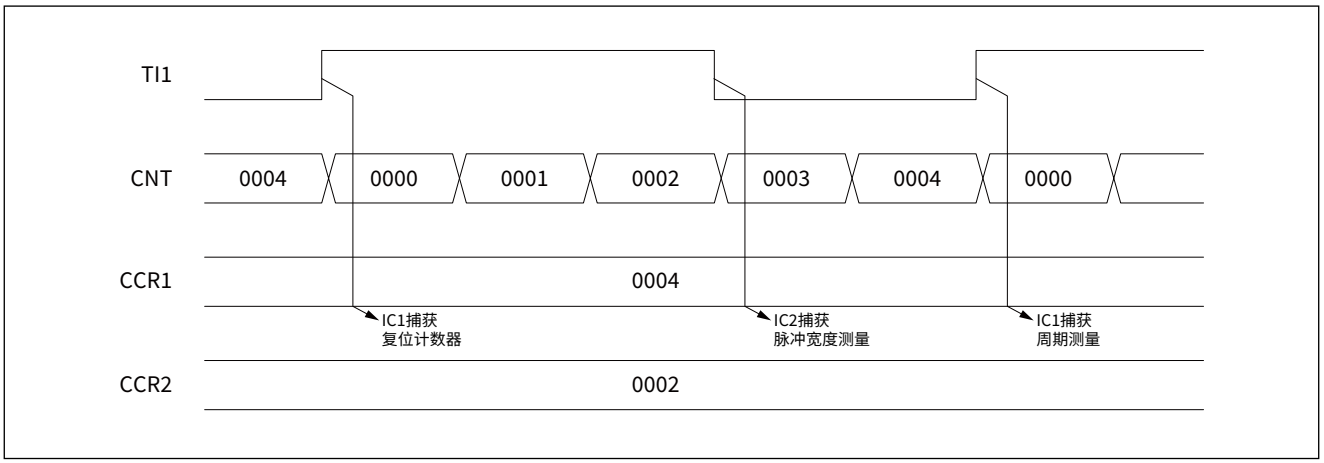
14.3.3.1 PWM 输入模式

PWM 输入模式是输入捕获模式的一个应用。输入捕获功能支持同一通道输入信号被两路捕获寄存器进行捕获，结合从模式复位功能，可方便测量 PWM 输入信号的周期和脉宽。

PWM 输入模式只能与 TI1 和 TI2 信号配合使用，因为只有 TI1FP1 和 TI2FP2 与从模式控制器相连。

下图是 PWM 信号从 TI1 通道输入时的时序示例：

图 14-35 PWM 输入模式时序



14.3.3.2 输入捕获来源

ATIM 的输入捕获来源可以是外部 ATIM_CHy 引脚，也可以是片内其它外设，通过 TI 输入选择寄存器（ATIM_TISEL1 和 ATIM_TISEL2）的 TIySEL 位域进行配置。

当 ATIM_TISELx.TIySEL 为 0x0 时，TIy 通道输入捕获信号来源为外部 ATIM_CHy 引脚，此时需通过 GPIO 复用功能寄存器 (GPIOx_AFRH 和 GPIOx_AFRL) 将对应引脚配置为复用功能。

当 ATIM_TISELx.TIySEL 为 0x1~0xD 时，TIy 通道输入捕获信号来自片内其它外设，如下表所示：

表 14-8 高级定时器输入捕获来源

TIySEL 位域值	TIy 通道的输入捕获来源
0000	ATIM_CHy 引脚
0001	VC1_OUT
0010	VC2_OUT
0011	UART1_RXD
0100	UART2_RXD
0101	UART3_RXD
0110	HSE_FAULT
0111	LSE_FAULT
1000	RTC_OUT
1001	LSI_OUT
1010	BTIM1_Trgo
1011	BTIM2_Trgo
1100	BTIM3_Trgo
1101	GTIM1_Trgo
1110	GTIM2_Trgo
1111	MCO_OUT

14.3.4 输出比较功能

ATIM 支持输出比较功能，设置比较模式寄存器 ATIM_CCMRxCMP 的 CCyS 位域为 0 时，CCy 通道配置为输出。在输出比较模式下，当前计数器 CNT 的值与对应通道 CCy 的捕获 / 比较寄存器 CCRy 的值相比较，当两者匹配时，参考信号 OCyREF 输出为可设定的电平状态，同时产生比较中断。参考信号经输出控制单元后由 CHy 和 CHyN 引脚输出，其输出逻辑可参见 14.3.1.10 输出比较通道。

参考信号 OCyREF 的输出动作由比较模式寄存器 ATIM_CCMRxCMP 的 OCyM 位域设置，如下表所示：

表 14-9 输出比较模式配置

OCyM 位域值	比较模式配置
0000	比较匹配时 OCyREF 保持原电平
0001	比较匹配时 OCyREF 置 1
0010	比较匹配时 OCyREF 置 0
0011	比较匹配时 OCyREF 翻转
0100	强制 OCyREF 为低电平
0101	强制 OCyREF 为高电平
0110	PWM 模式 1
0111	PWM 模式 2
1000	可再触发 OPM 模式 1
1001	可再触发 OPM 模式 2
1011	计数方向输出
1100	组合 PWM 模式 1
1101	组合 PWM 模式 2
1110	不对称 PWM 模式 1
1111	不对称 PWM 模式 2

当发生比较匹配时，对应通道的捕获 / 比较中断标志 ATIM_ISR.CCyIF 被硬件置 1，同时：

- 如果使能了中断（设置 ATIM_IER.CCyIE 为 1），将产生中断请求。
- 设置 ATIM_ICR.CCyIF 为 0 可清除 ATIM_ISR.CCyIF 标志位。

捕获 / 比较寄存器 ATIM_CCRy 具有缓存功能，通过 OCyPE 位域选择是否使用缓存功能。当 OCyPE 为 0 时，禁止通道 CCy 的比较缓存功能，可在任意时候通过软件更新 ATIM_CCRy 寄存器，更新值立即生效并影响输出波形；当 OCyPE 为 1 时，使能通道 CCy 的比较缓存功能，更新 ATIM_CCRy 寄存器不会立即生效，仅当发生更新事件 UEV 时才会将 ATIM_CCRy 寄存器的值更新到有效寄存器。

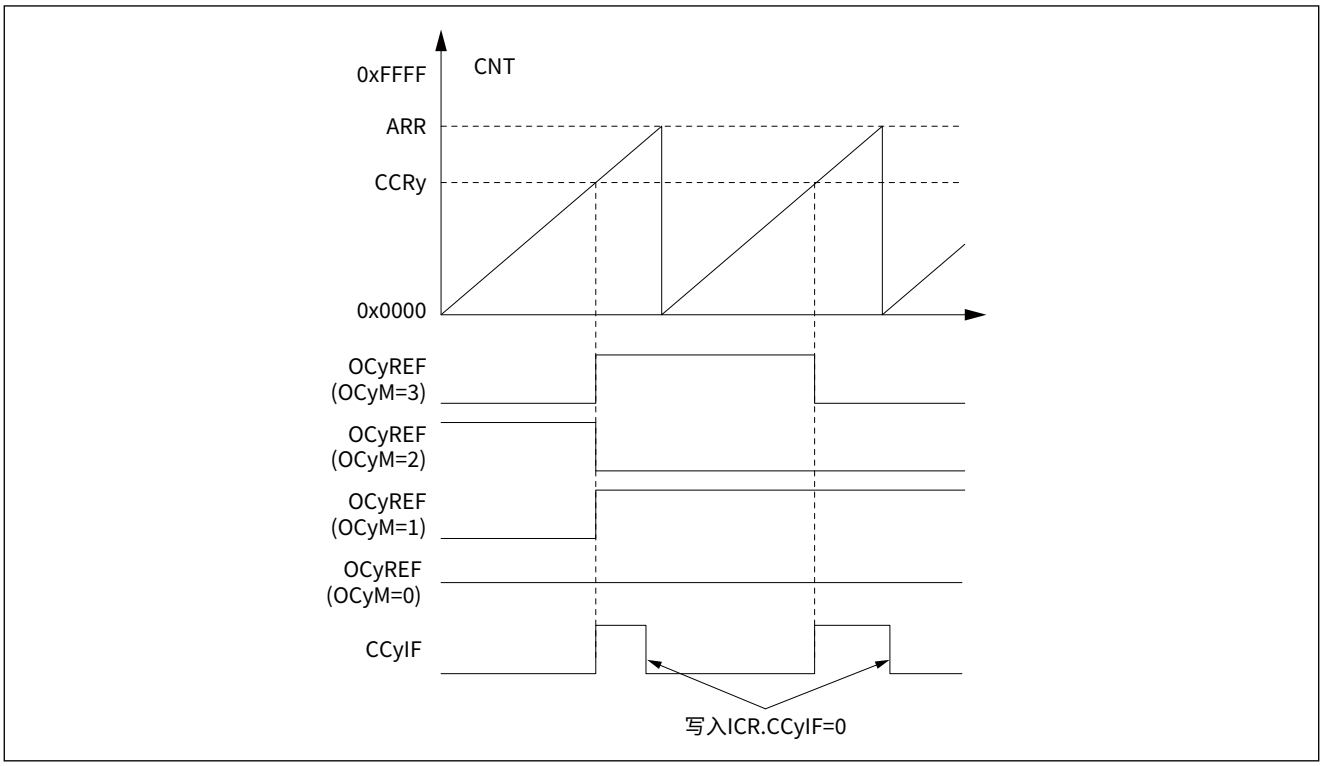
14.3.4.1 匹配输出模式

匹配输出模式用于控制输出波形，或指示已经过某一时间段。

- 设置 OCyM 位域为 0x0，比较匹配时，参考信号 OCyREF 保持其电平，同时 ATIM_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x1，比较匹配时，参考信号 OCyREF 设置为高电平，同时 ATIM_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x2，比较匹配时，参考信号 OCyREF 设置为低电平，同时 ATIM_ISR.CCyIF 标志位置 1。
- 设置 OCyM 位域为 0x3，比较匹配时，参考信号 OCyREF 发生翻转，同时 ATIM_ISR.CCyIF 标志位置 1。

下图是各比较匹配输出模式时序示例，示例为边沿对齐递增模式。

图 14-36 比较匹配输出模式



14.3.4.2 强制输出模式

在强制输出模式下，输出比较信号能够直接由软件强置为高或低状态，而不依赖于捕获 / 比较寄存器 ATIM_CCRy 和计数寄存器 ATIM_CNT 的比较结果。

设置比较模式寄存器 ATIM_CCMRxCMP 的 OCyM 位域为 0x4，即可将参考信号 OCyREF 强制变为低电平；设置 OCyM 位域为 0x5，即可将参考信号 OCyREF 强制变为高电平。

强制输出模式下，ATIM_CCRy 寄存器和计数器 ATIM_CNT 之间的比较仍然在进行，相应的标志也会置 1，也会产生相应的中断请求。

14.3.4.3 PWM 模式

脉冲宽度调制（PWM）模式可以产生一个由重载寄存器 ATIM_ARR 确定频率、由捕获 / 比较寄存器 ATIM_CCRy 确定占空比的信号。

为了防止修改 ARR 寄存器时输出的 PWM 信号出现错误，需设置 ATIM_CR1.ARPE 为 1 以使能 ARR 寄存器的缓存功能；为了防止修改 CCRy 寄存器时输出的 PWM 信号出现错误，需设置 ATIM_CCMRxCMP.OCyPE 为 1 以使能 CCRy 寄存器的缓存功能。只有发生更新事件 UEV 时预装载寄存器的值才会传送到影子寄存器；配置完成各种参数启动定时器之前，必须向 ATIM_EGR.UG 位写入 1 以初始化所有寄存器。

各通道可以独立选择 PWM 模式，只需向比较模式寄存器 ATIM_CCMRxCMP 的 OCyM 位域写入 0x6（PWM 模式 1）或 0x7（PWM 模式 2），输出状态如下表所示：

表 14-10　PWM 模式 1/2 输出状态

工作模式	计数方向	PWM 模式 1	PWM 模式 2
边沿对齐模式 中央对齐模式	向上	CNT < CCRy 时，OCyREF 为高	CNT < CCRy 时，OCyREF 为低
	向下	CNT > CCRy 时，OCyREF 为低	CNT > CCRy 时，OCyREF 为高

OCy 的极性可以通过 ATIM_CCER 寄存器的 CCyP 位域设置，可将其设置为高电平有效或低电平有效。通过 CCyE、CCyNE、MOE、OSSI 和 OSSR 位（ATIM_CCER 和 ATIM_BDTR 寄存器）的组合使能 OCy 输出。

以下是不同计数模式下各 PWM 模式的波形实例，其中 ATIM_ARR 为 0x08：

图 14-37　PWM 模式 1，边沿对齐模式，递增计数

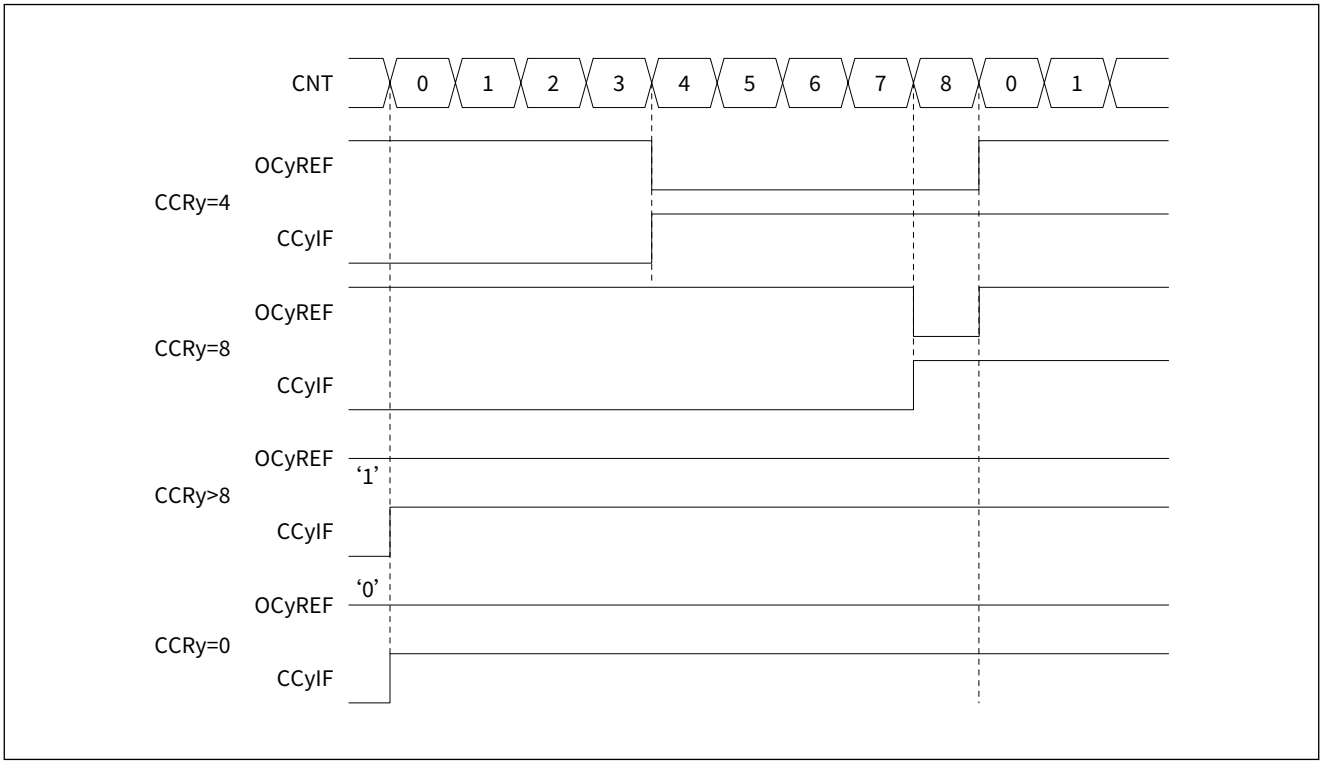


图 14-38　PWM 模式 1，边沿对齐模式，递减计数

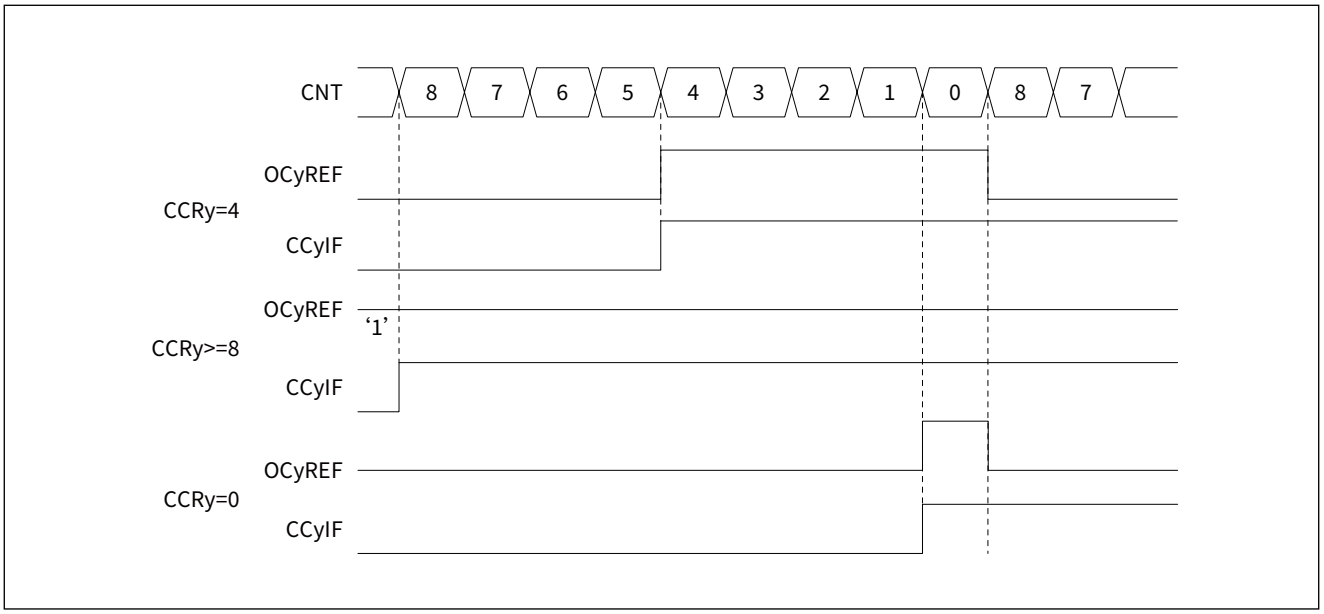
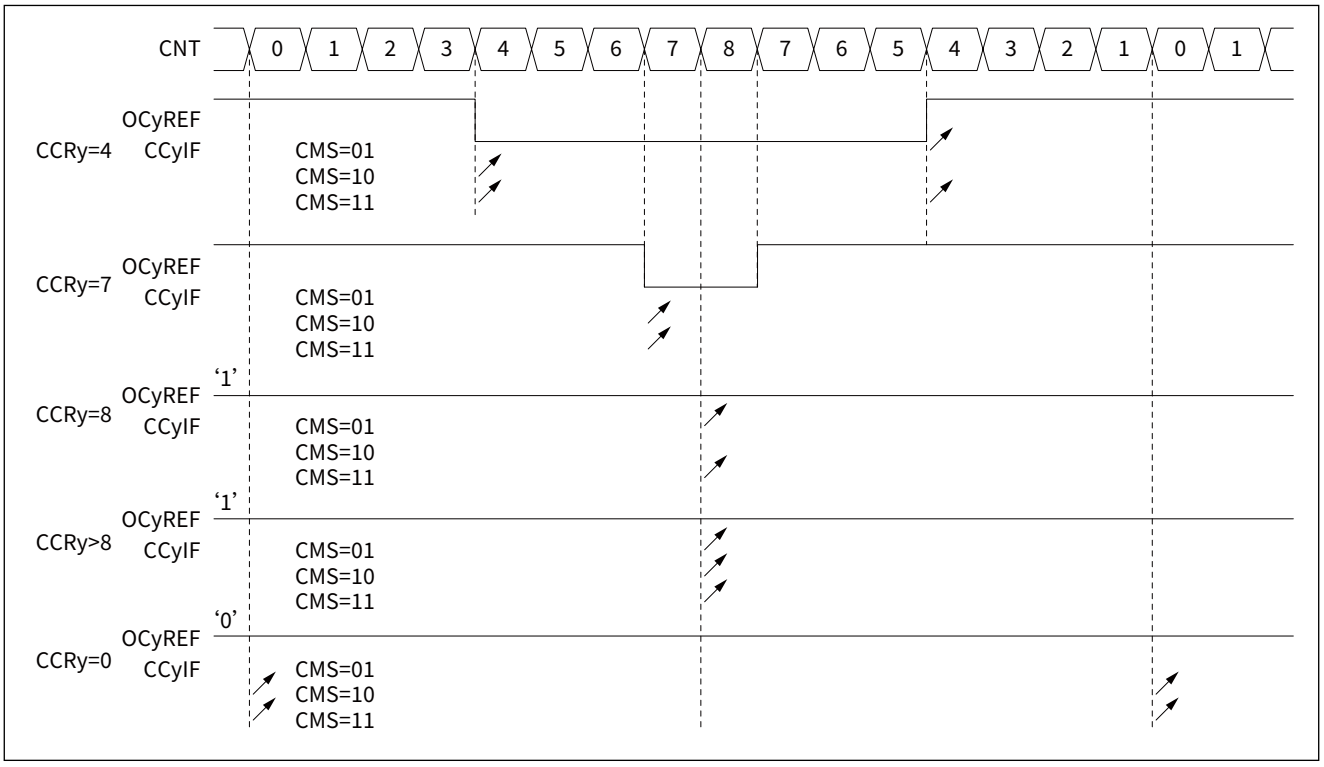


图 14-39　PWM 模式 1，中心对齐模式



14.3.4.4 不对称 PWM 模式

不对称 PWM 输出是相对于中心对齐模式计数时的对称 PWM 输出而言的。在不对称 PWM 输出模式下，允许在中心对齐模式下生成两个具有可编程相移的 PWM 信号，频率由重载寄存器 ATIM_ARR 确定，占空比和相移由两个捕获 / 比较寄存器 ATIM_CCRy 确定，其中一个 CCRy 寄存器控制递增计数时的 PWM，另一个 CCRy 寄存器控制递减计数时的 PWM，具体如下所示：

- OC1REFC 由 CCR1(递增计数) 与 CCR2(递减计数) 控制
- OC2REFC 由 CCR2(递增计数) 与 CCR1(递减计数) 控制
- OC3REFC 由 CCR3(递增计数) 与 CCR4(递减计数) 控制
- OC4REFC 由 CCR4(递增计数) 与 CCR3(递减计数) 控制
- OC5REFC 由 CCR5(递增计数) 与 CCR6(递减计数) 控制
- OC6REFC 由 CCR6(递增计数) 与 CCR5(递减计数) 控制

各通道可以独立选择不对称 PWM 模式，只需向比较模式寄存器 ATIM_CCMRxCMP 的 OCyM 位域写入 0xE（不对称 PWM 模式 1）或 0xF（不对称 PWM 模式 2）。

当设定通道用作不对称 PWM 输出通道时，另一辅助通道仍可使用。例如，通道 1 配置为不对称 PWM 模式 1 产生 OC1REFC 信号时，通道 2 仍可以配置为 PWM 模式 2 输出 OC2REF 信号，或者配置为不对称 PWM 模式 2 输出 OC2REFC 信号。

以下是不对称 PWM 模式 1/2 的波形实例，其中 ARR=8、CCR1=0、CCR2=8、CCR3=3、CCR4=5。

图 14-40　不对称 PWM 模式 1 示例

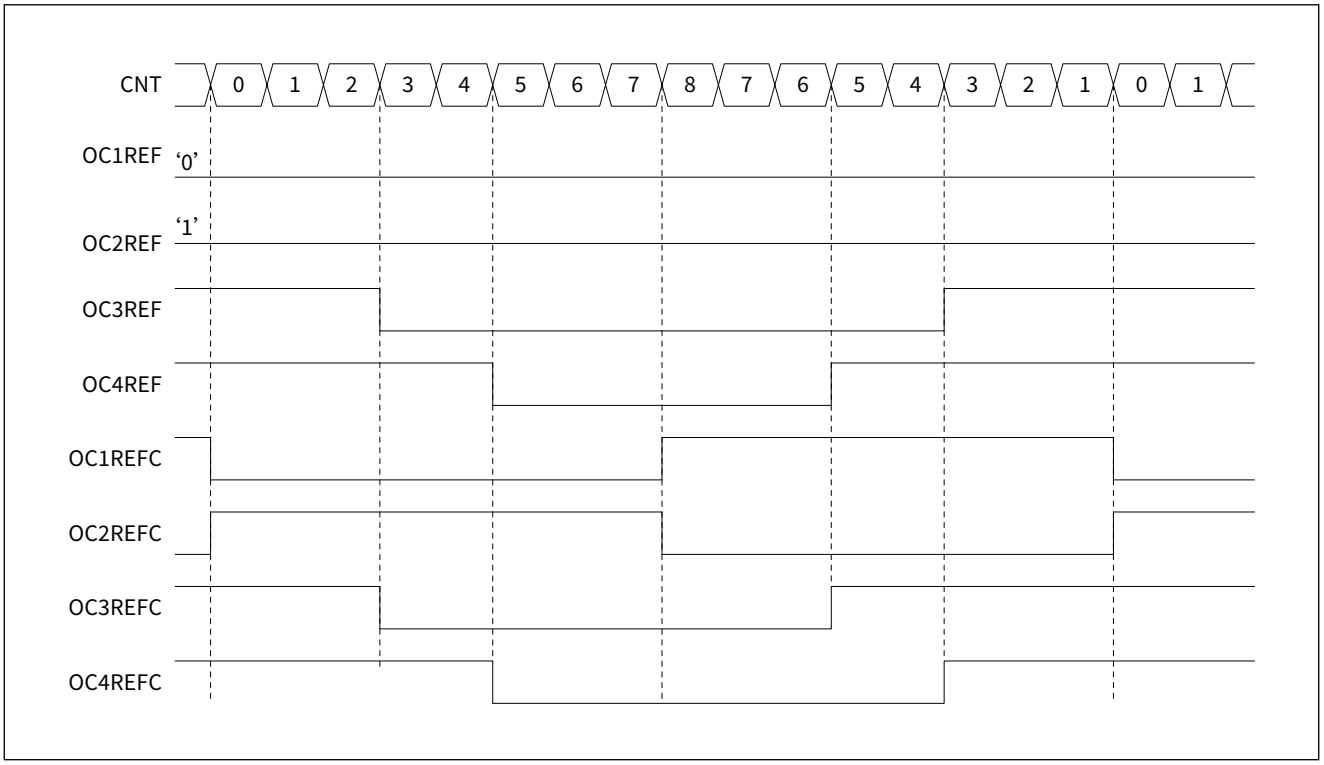
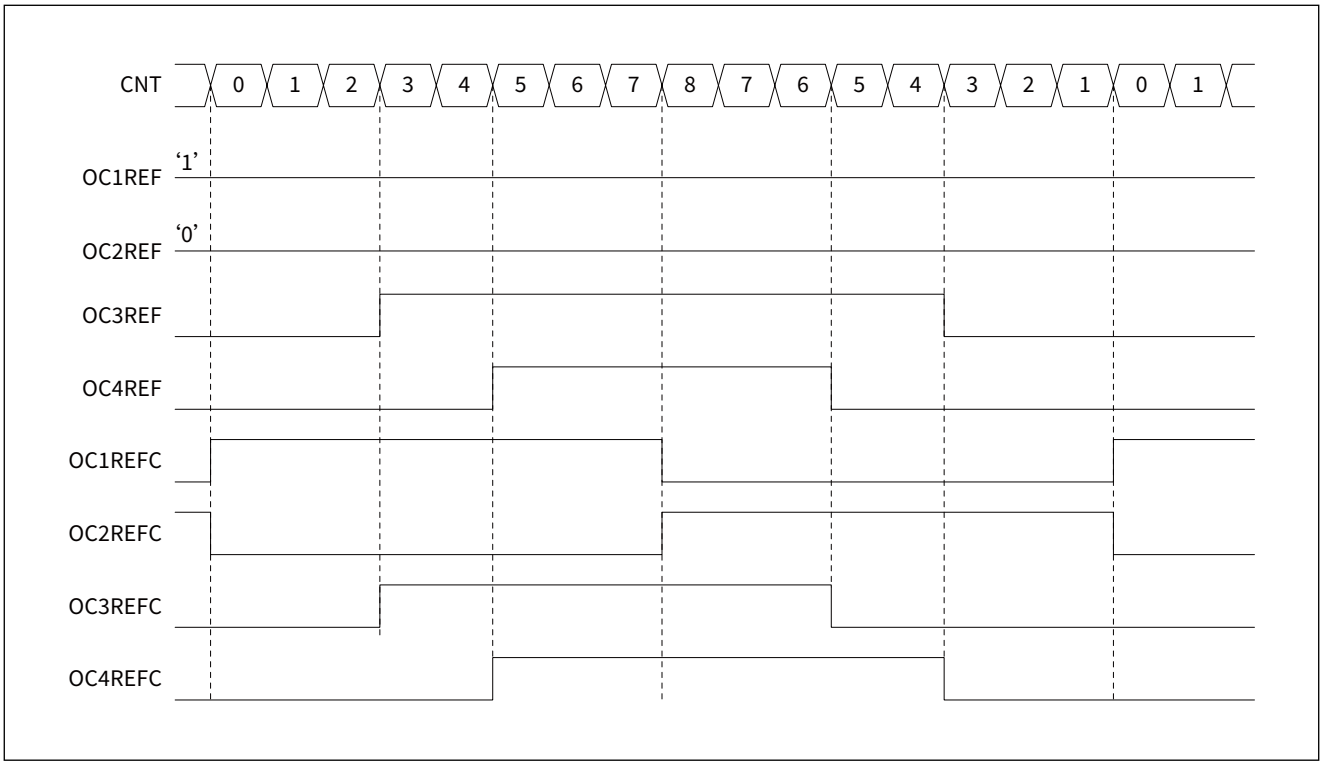


图 14-41　不对称 PWM 模式 2 示例



14.3.4.5 组合 PWM 模式

在组合 PWM 输出模式下，允许在边沿对齐或中心对齐模式下生成两个具有可编程延时和相移的 PWM 信号，频率由重载寄存器 ATIM_ARR 确定，占空比和延时由两个捕获 / 比较寄存器 ATIM_CCRy 确定。

该模式下，OCyREFC 信号由两个参考信号 OCyREF 的逻辑或运算或者逻辑与运算组合生成，通过比较模式寄存器 ATIM_CCMRxCMP 的 OCyM 位域可独立选择组合 PWM 模式，具体如下表所示：

表 14-11 组合 PWM 模式配置

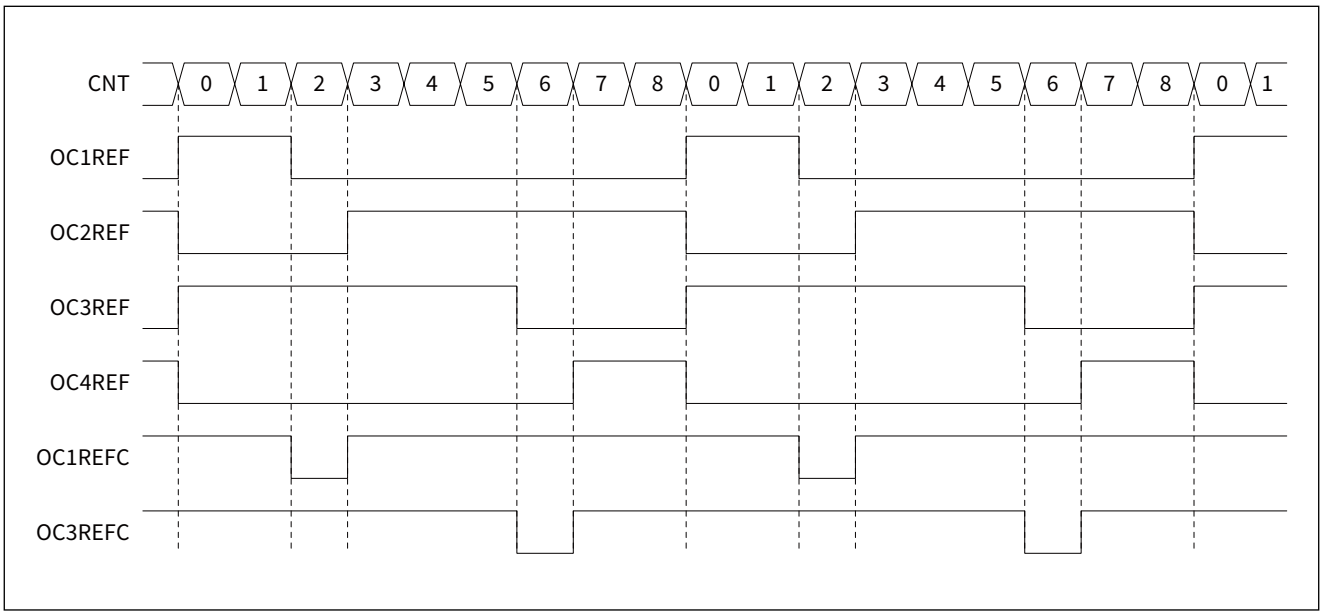
OCyM 位域值	输出状态
0xC （组合 PWM 模式 1）	OC1REFC（或 OC2REFC）由 OC1REF 和 OC2REF 逻辑或运算生成 OC3REFC（或 OC4REFC）由 OC3REF 和 OC4REF 逻辑或运算生成 OC5REFC（或 OC6REFC）由 OC5REF 和 OC6REF 逻辑或运算生成
0xD （组合 PWM 模式 2）	OC1REFC（或 OC2REFC）由 OC1REF 和 OC2REF 逻辑与运算生成 OC3REFC（或 OC4REFC）由 OC3REF 和 OC4REF 逻辑与运算生成 OC5REFC（或 OC6REFC）由 OC5REF 和 OC6REF 逻辑与运算生成

当设定通道用作组合 PWM 输出通道时，另一辅助通道仍可使用，但必须设置为相反的 PWM 模式。例如，通道 1 配置为组合 PWM 模式 1 产生 OC1REFC 信号时，通道 2 仍可使用，通道 2 可以配置为 PWM 模式 2 输出 OC2REF 信号。

下图显示了边沿对齐递增模式下，组合 PWM 模式 1 的信号示例，具体配置如下：

- 重载值 ARR 为 8
- 通道 1 配置为组合 PWM 模式 1，CCR1=2
- 通道 2 配置为 PWM 模式 2，CCR2=3
- 通道 3 配置为组合 PWM 模式 1，CCR3=6
- 通道 4 配置为 PWM 模式 2，CCR4=7

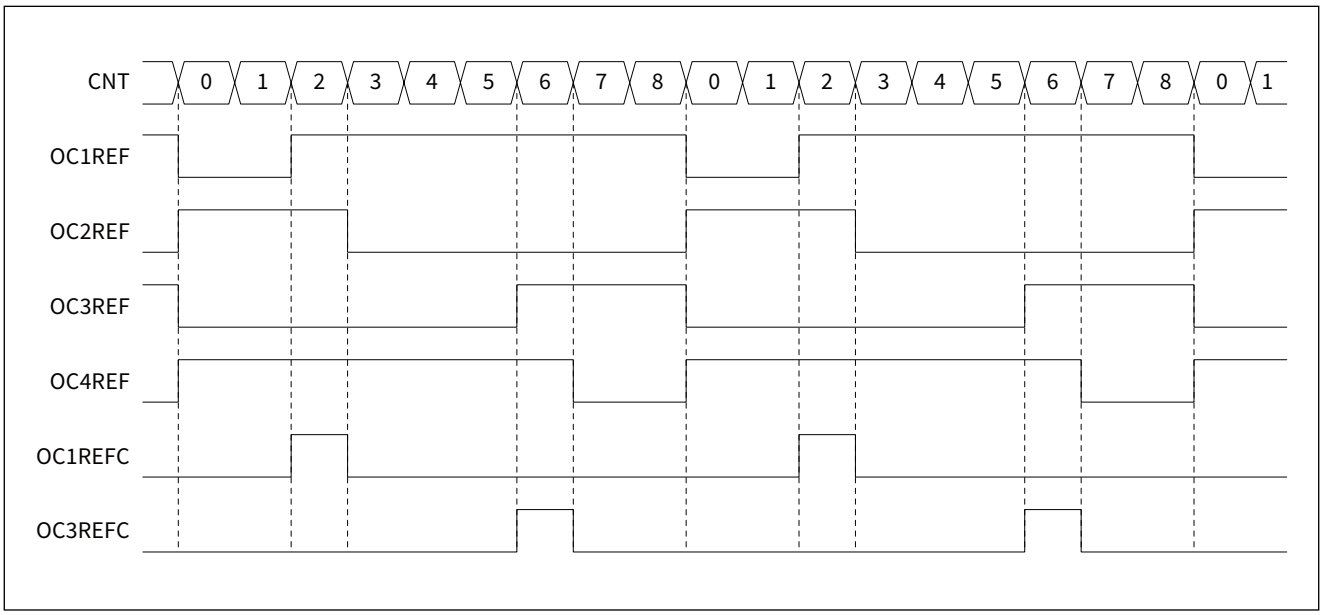
图 14-42 组合 PWM 模式 1 示例



下图显示了边沿对齐递增模式下，组合 PWM 模式 2 的信号示例，具体配置如下：

- 重载值 ARR 为 8
- 通道 1 配置为组合 PWM 模式 2，CCR1=2
- 通道 2 配置为 PWM 模式 1，CCR2=3
- 通道 3 配置为组合 PWM 模式 2，CCR3=6
- 通道 4 配置为 PWM 模式 1，CCR4=7

图 14-43 组合 PWM 模式 2 示例



14.3.4.6 移相 PWM 模式

在电机应用中，需要在两路 PWM 波形上升沿之间的特定时刻通过 ADC 对电流进行采样。当两路 PWM 波形上升沿之间的间隔太短而不能完成电流采样时，可通过移相 PWM 模式调整其中一路 PWM 波形的边沿，在保持占空比不变的条件下增大两路 PWM 波形上升沿之间的间隔。

设置 ATIM_CCR5 寄存器的 GC5Cy 位域为 1，使能移相 PWM 模式 5，OCyREFC 输出状态如下：

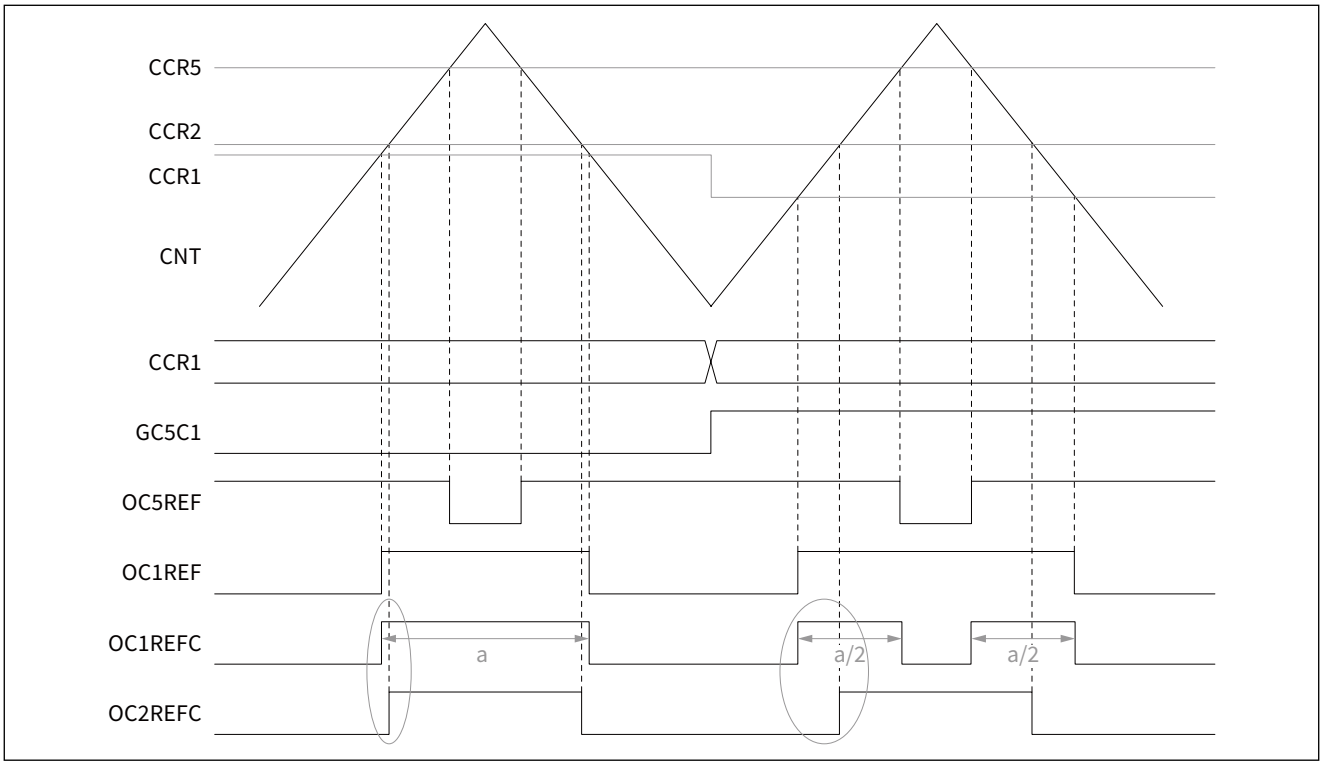
- 设置 GC5C1 为 1，则 OC1REFC 输出 OC1REF 和 OC5REF 的逻辑与
- 设置 GC5C2 为 1，则 OC2REFC 输出 OC2REF 和 OC5REF 的逻辑与
- 设置 GC5C3 为 1，则 OC3REFC 输出 OC3REF 和 OC5REF 的逻辑与
- 设置 GC5C4 为 1，则 OC4REFC 输出 OC4REF 和 OC5REF 的逻辑与
- 设置 GC5C6 为 1，则 OC6REFC 输出 OC6REF 和 OC5REF 的逻辑与

设置 ATIM_CCR6 寄存器的 GC6Cy 位域为 1，使能移相 PWM 模式 6，OCyREFC 输出状态如下：

- 设置 GC6C1 为 1，则 OC1REFC 输出 OC1REF（递增计数）和 OC6REF（递减计数）
- 设置 GC6C2 为 1，则 OC2REFC 输出 OC2REF（递增计数）和 OC6REF（递减计数）
- 设置 GC6C3 为 1，则 OC3REFC 输出 OC3REF（递增计数）和 OC6REF（递减计数）
- 设置 GC6C4 为 1，则 OC4REFC 输出 OC4REF（递增计数）和 OC6REF（递减计数）
- 设置 GC6C5 为 1，则 OC5REFC 输出 OC5REF（递增计数）和 OC6REF（递减计数）

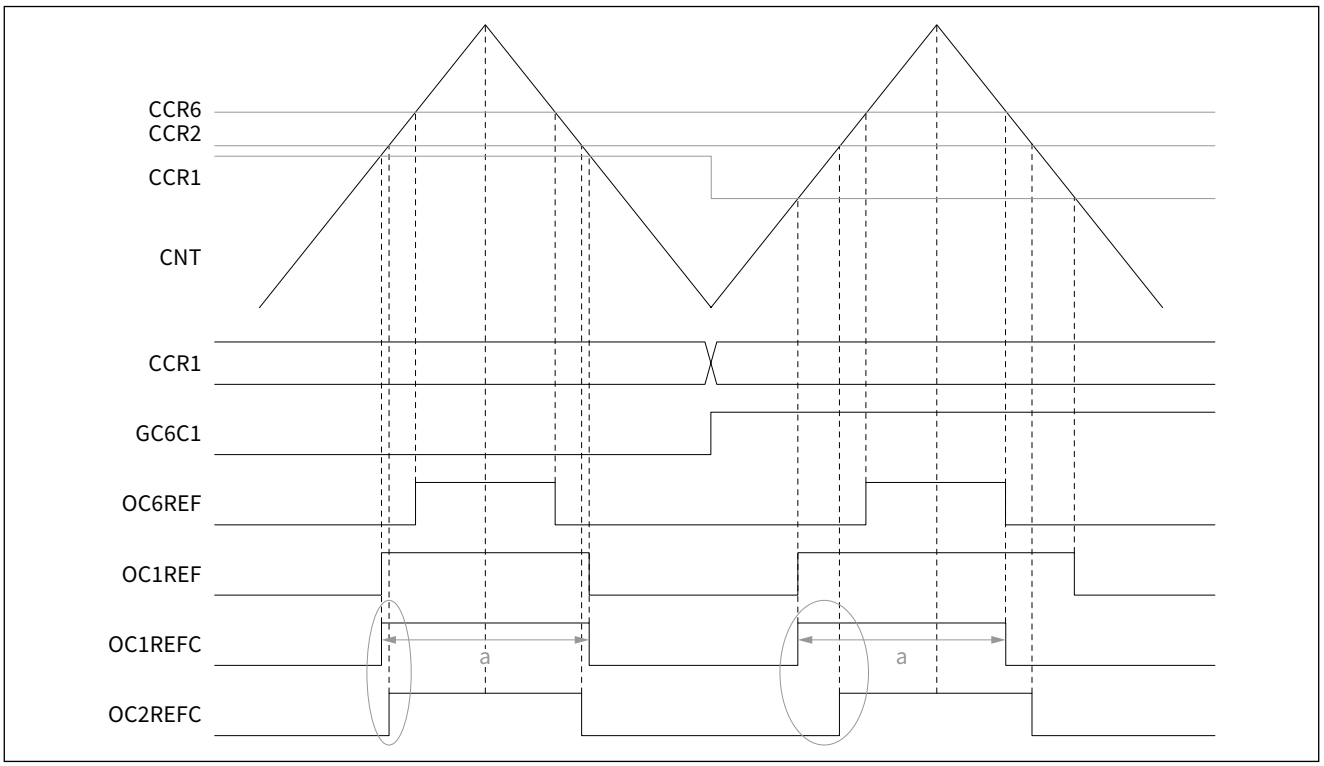
下图展示了移相 PWM 模式 5 下，OC1REFC 上升沿与 OC2REFC 上升沿之间间隔增大的时序图，通过调整 CCR1 与 GC5C1 实现。

图 14-44　移相 PWM 模式 5 示例



下图展示了移相 PWM 模式 6 下，OC1REFC 上升沿与 OC2REFC 上升沿之间间隔增大的时序图，通过调整 CCR1 与 GC6C1 实现。

图 14-45　移相 PWM 模式 6 示例



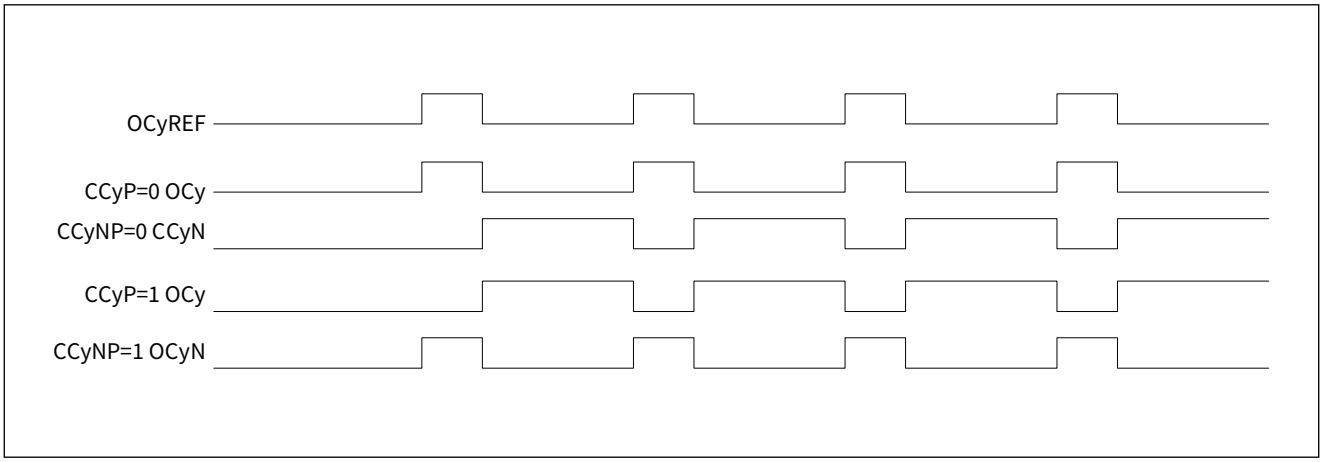
14.3.4.7 互补输出和死区插入

ATIM 可以输出 6 对互补 PWM 信号，且支持对称或不对称的死区插入时间。

主输出 OCy 和互补输出 OCyN 可独立设置输出极性，分别通过 ATIM_CCER 寄存器的 CCyP 和 CCyNP 位域进行控制。

主输出 OCy 和互补输出 OCyN 通过以下多个控制位的组合进行激活：ATIM_CCER 寄存器的 CCyE 和 CCyNE 位，ATIM_BDTR 寄存器的 MOE、OSSI、OSSR 位，以及 ATIM_CR2 寄存器的 OISy、OISyN 位，更多详细信息，请参见表 14-13 有刹车功能的互补通道 OCy 和 OCyN 的输出控制位。应当注意，切换至空闲状态（MOE 变为 0）的时刻，死区仍然有效。

图 14-46 互补 PWM 输出波形



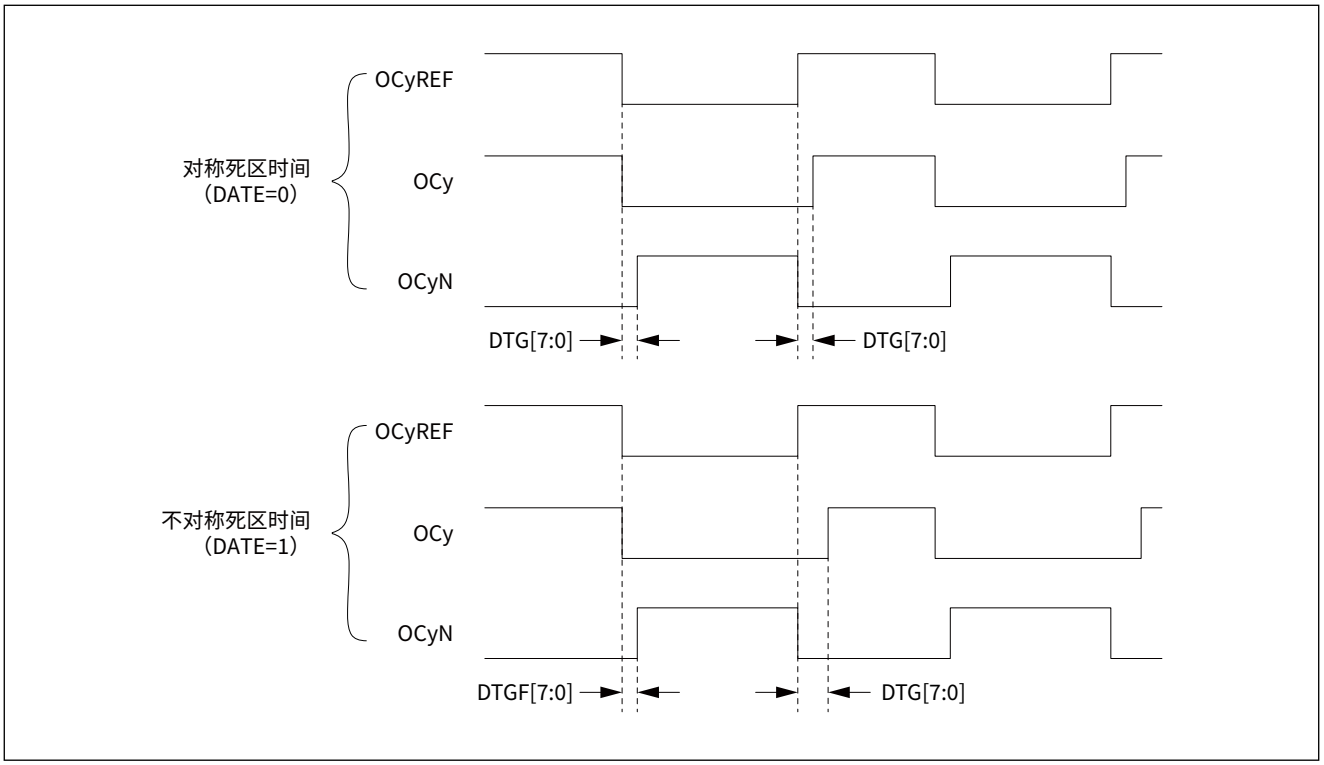
ATIM 支持对称或不对称的死区插入时间， 死区时间对所有通道均相同。当 ATIM_DTR2.DTAE 为 0 时，上升沿和下降沿的死区时间是相同的，由 ATIM_BDTR 寄存器的 DTG[7:0] 定义；当 ATIM_DTR2.DTAE 为 1 时，上升沿的死区时间由 DTG[7:0] 定义，下降沿的死区时间由 DTGF[7:0] 定义。DTAE 位必须在使能计数器之前写入，且 CEN 为 1 时不得修改。DTG[7:0] 定义如下表所示，DTGF[7:0] 定义与 DTG[7:0] 相同。

表 14-12 死区时间设置

ATIM_BDTR.DTG	步长	死区时间	当 t_{DTS}=125ns 时，死区时间范围
DTG[7:5]=0xx	$1t_{DTS}$	DTG[7:0]×t_{DTS}	0~15.875μs
DTG[7:5]=10x	$2t_{DTS}$	(64+DTG[5:0])×2×t_{DTS}	16~31.750μs
DTG[7:5]=110	$8t_{DTS}$	(32+DTG[4:0])×8×t_{DTS}	32~63μs
DTG[7:5]=111	$16t_{DTS}$	(32+DTG[4:0])×16×t_{DTS}	64-126μs

下图所示为对称和不对称死区插入时间时，互补 PWM 输出信号与参考信号 OCyREF 之间的关系，示例中 CCyP=0、CCyNP=0。

图 14-47　带死区插入的互补 PWM 输出波形



可以使用预加载机制在 PWM 操作期间即时更新死区时间值。当 DTPE 位置 1 时，使能死区时间值预加载，预加载值在下一个更新事件时加载到有效寄存器中。如果在计数器使能时 DTPE 位使能，则自上次更新以来写入的任何新值都将被丢弃，并使用先前的值。

14.3.4.8 刹车功能

ATIM 支持两个刹车输入 BRK 和 BRK2，当刹车信号有效时，刹车电路会关闭 PWM 输出，并将其强制为预定义的安全状态。

刹车状态支持从 GPIO 口输出，需要用户将对应 GPIO 端口配置为数字输出，同时选择功能复用。当 ATIM 处于刹车状态时输出高电平，当 ATIM 处于非刹车状态时输出低电平。用户可以利用该特性实现更多的板级保护。

刹车通道源包括系统级故障（时钟失效和奇偶校验错误等）和应用故障（来自输入引脚和内置比较器），可以在死区持续时间后将输出强制为预定义的电平（有效或无效）。刹车 2 通道源包括应用故障，只能将输出强制为无效状态。BRK 输入的优先级高于 BRK2 输入。

刹车（BRK）触发源：

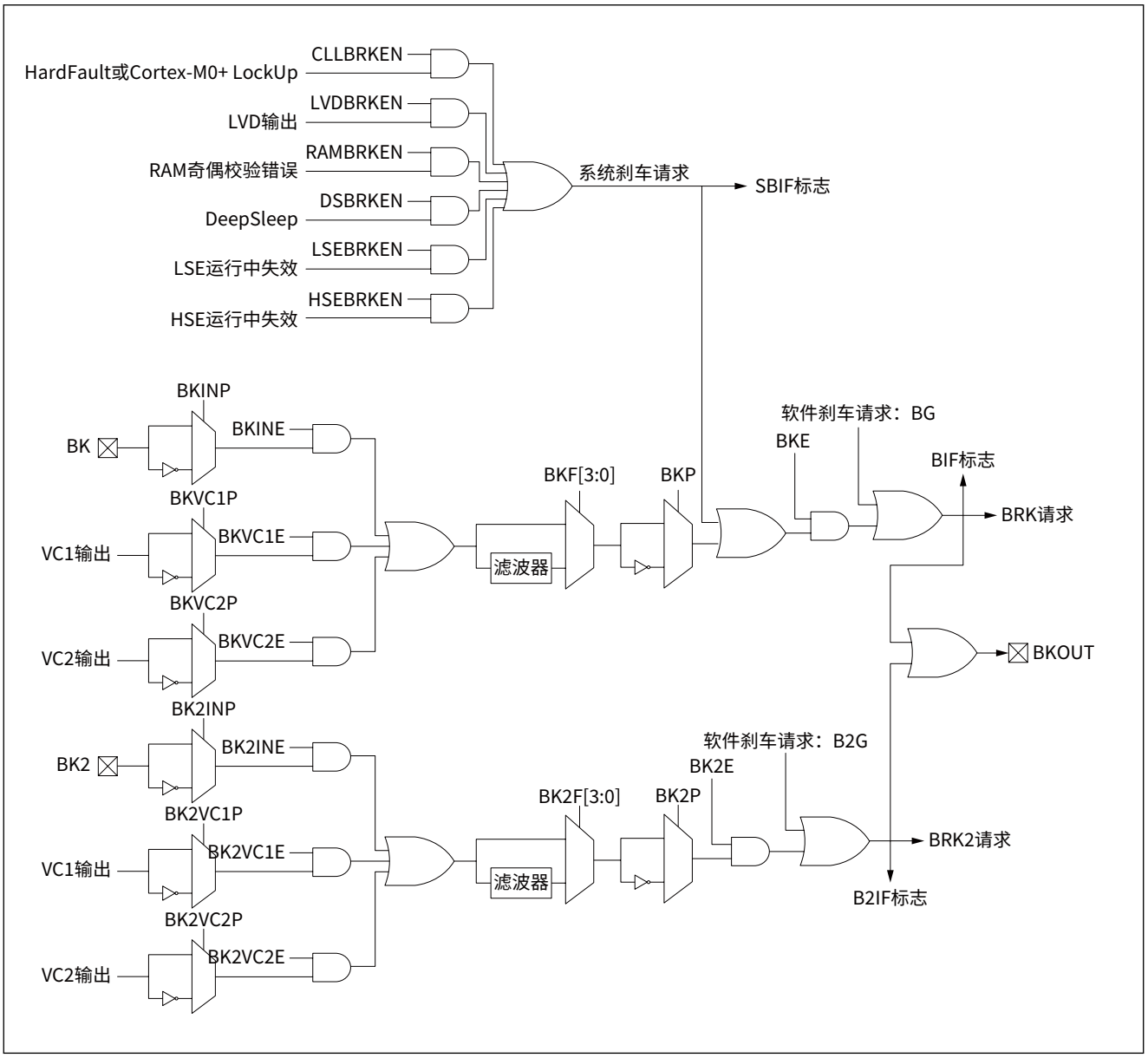
- 软件刹车，设置 ATIM_EGR 寄存器的 BG 位域为 1 生成刹车事件，BG 位由硬件自动清零。
- 电压比较器 VC1/2 的比较输出，ATIM_AF1 寄存器相关位域可设置有效极性和输入使能。
- 系统刹车请求，通过 SYSCTRL_CR2 寄存器相应位域使能：
  - 产生 HardFault 或 Cortex-M0+ LockUp 标志
  - LVD 输出
  - RAM 奇偶校验错误
  - DeepSleep
  - LSE 运行中失效
  - HSE 运行中失效
- 外部 ATIM_BK 引脚输入（GPIO 功能复用设置），ATIM_AF1 寄存器相关位域可设置有效极性和输入使能。

刹车 2（BRK2）触发源：

- 软件刹车，设置 ATIM_EGR 寄存器的 B2G 位域为 1 生成刹车事件，B2G 位由硬件自动清零。
- 电压比较器 VC1/2 的比较输出，ATIM_AF2 寄存器相关位域可设置有效极性和输入使能。
- 外部 ATIM_BK2 引脚输入（GPIO 功能复用设置），ATIM_AF2 寄存器相关位域可设置有效极性和输入使能。

在所有源进入定时器 BRK 或 BRK2 输入之前，对其进行逻辑或运算，如下图所示：

图 14-48　刹车和刹车 2 电路



注意：

只有禁止可编程滤波器时才能保证异步（无时钟）操作。如果使能可编程滤波器，必须使用故障安全时钟模式来保证能够处理刹车事件。

发生刹车事件时，MOE 位异步清零，使输出处于无效状态、空闲状态甚至释放控制权给 GPIO 控制器（通过 OSSI 位进行选择，具体请参见表 14-13 有刹车功能的互补通道 OCy 和 OCyN 的输出控制位）。即使 MCU 振荡器关闭，该功能仍然使能。如果 AOE 位置 1，则 MOE 位会在发生下一更新事件（UEV）时自动再次置 1，否则 MOE 将始终保持低电平，直到应用将其再次置 1。因为刹车输入为电平有效，因此，当刹车输入为有效电平时，MOE 位不能被置 1（自动或通过软件），状态标志 BIF 和 B2IF 也不能被清零。

下图所示为发生刹车事件 BRK 时 OCy 和 OCyN 的输出行为示例：

图 14-49　响应刹车事件 BRK 时的输出行为 (OSSI=1)

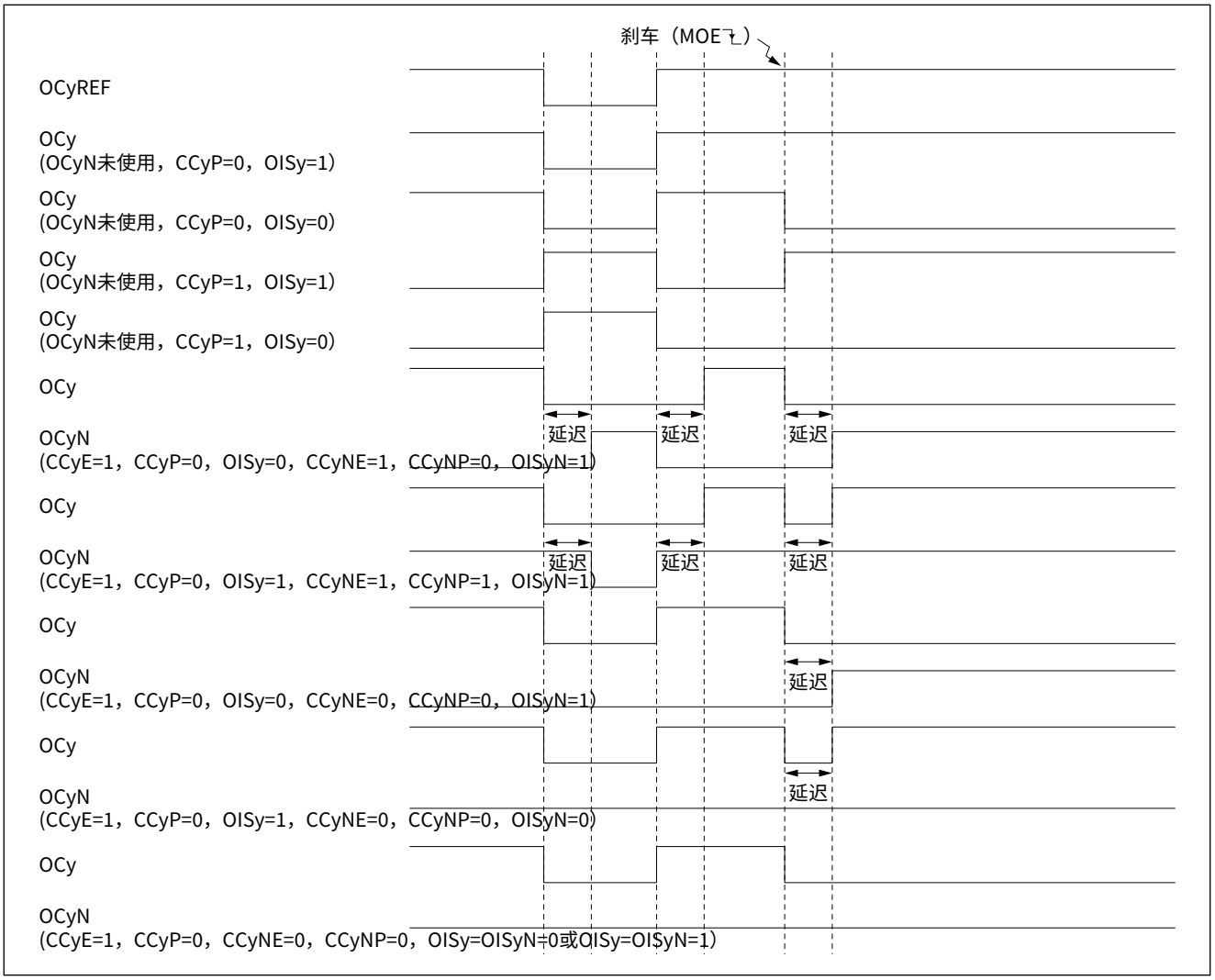
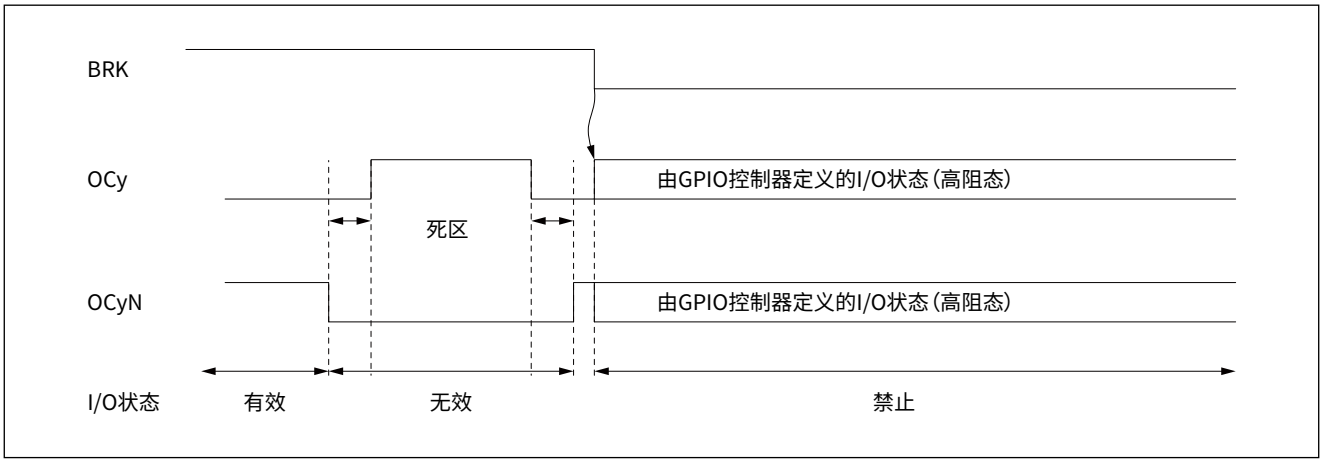


图 14-50　响应刹车事件 BRK 时的输出行为（OSSI=0）



下图所示为发生刹车事件 BRK 和 BRK2 时 OCy 和 OCyN 的输出行为示例，其中 CCyP=CCyNP=0。

图 14-51 响应刹车 BRK 和 BRK2 时的输出行为（OSSI=1）

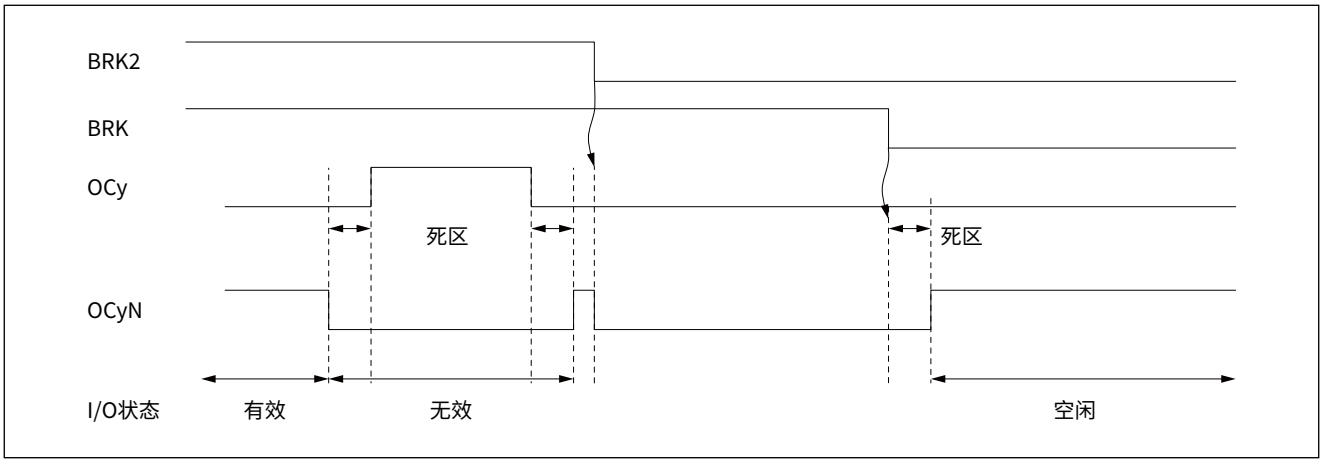


表 14-13 有刹车功能的互补通道 OCy 和 OCyN 的输出控制位

输出控制位					输出状态	
MOE	OSSI	OSSR	CCyE	CCyNE	OCy 输出状态	OCyN 输出状态
1	X	X	0	0	禁止输出（不再由定时器驱动）。 输出状态由 GPIO 逻辑强制为高阻态。 OCy=0、OCyN=0	
		0	0	1	禁止输出（不再由定时器驱动）。 输出状态由 GPIO 逻辑强制为高阻态。 OCy=0	OCyREF+ 极性 OCyN=OCyREF 异或 CCyNP
		0	1	0	OCyREF+ 极性 OCy=OCyREF 异或 CCyP	禁止输出（不再由定时器驱动）。 输出状态由 GPIO 逻辑强制为高阻态。 OCyN=0
		X	1	1	OCyREF+ 极性 + 死区	OCyREF 互补 + 极性 + 死区
		1	0	1	关闭状态（输出使能为无效状态） OCy=CCyP	OCyREF+ 极性 OCyN=OCyREF 异或 CCyNP
		1	1	0	OCyREF+ 极性 OCy=OCyREF 异或 CCyP	关闭状态（输出使能为无效状态） OCyN=CCyNP
0	0	X	X	X	禁止输出（不再由定时器驱动）。 输出状态由 GPIO 逻辑强制为高阻态。	
	1		0	0		
			0	1	关闭状态（输出使能为无效状态） 异步：OCy=CCyP、OCyN=CCyNP（如果触发 BRK 或 BRK2）。 随后（仅当触发 BRK 时才有效），如果存在时钟，则在死区时间后 OCy=OISy 且 OCyN=OISyN，假定 OISy 和 OISyN 并没有都设置成 OCy 及 OCyN 的有效电平（否则在半桥配置下驱动开关时可能导致短路）。 *注：BRK2 只能在 OSSI=OSSR=1 时使用。*	
			1	0		
			1	1		

注：

如果一个通道的两个输出均未使用（由 GPIO 接管控制），则 OISy、OISyN、CCyP 和 CCyNP 位必须保持清零状态。

14.3.4.9 可再触发单脉冲模式

可再触发单脉冲模式配合组合复位 + 触发模式（SMS=8），允许在触发信号（TRGI）的触发下复位并启动计数器，并产生长度可编程的脉冲，脉冲宽度由 ARR 确定。

与单脉冲模式相比，具有如下区别：

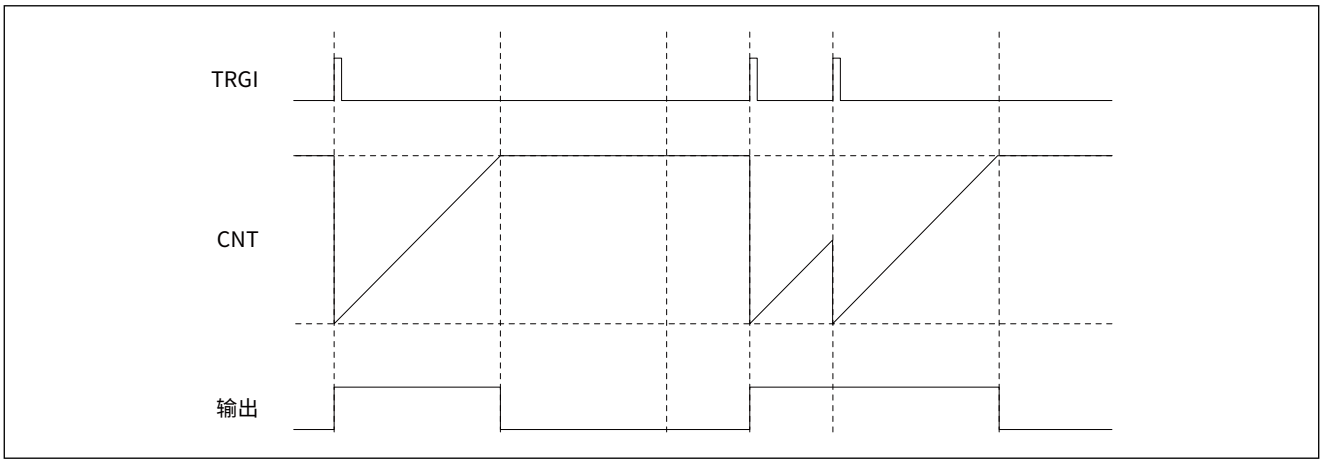
表 14-14 可再触发单脉冲模式和单脉冲模式

单脉冲模式	可再触发单脉冲模式
发生触发时，经一段可编程的延时后产生一个脉宽可编程的单脉冲	发生触发时，脉冲立即产生，无可编程延时
新的触发无效	如果上一个触发产生的脉冲未完成，又发生新的触发，脉冲将延长

此模式下，计数器只能设置为边沿对齐模式，不能设置为中心对齐模式。当配置为递增计数模式时，对应 CCRy 必须设置为 0；配置为递减计数模式时，对应 CCRy 必须大于或等于 ARR。

下图所示为可再触发单脉冲模式 2 的示例：

图 14-52 可再触发单脉冲模式 2 示例



14.3.4.10 计数方向输出

将 ATIM_CCMRxCMP 寄存器的 OCyM 位域设置为 0xB，可在对应的参考信号 OCyREF 上输出计数方向信号（ATIM_CR1 寄存器中 DIR 位的拷贝）。

此功能可用于在编码器模式下监控计数方向（或旋转方向），或在中心对齐 PWM 模式下指示向上 / 向下相位的信号。

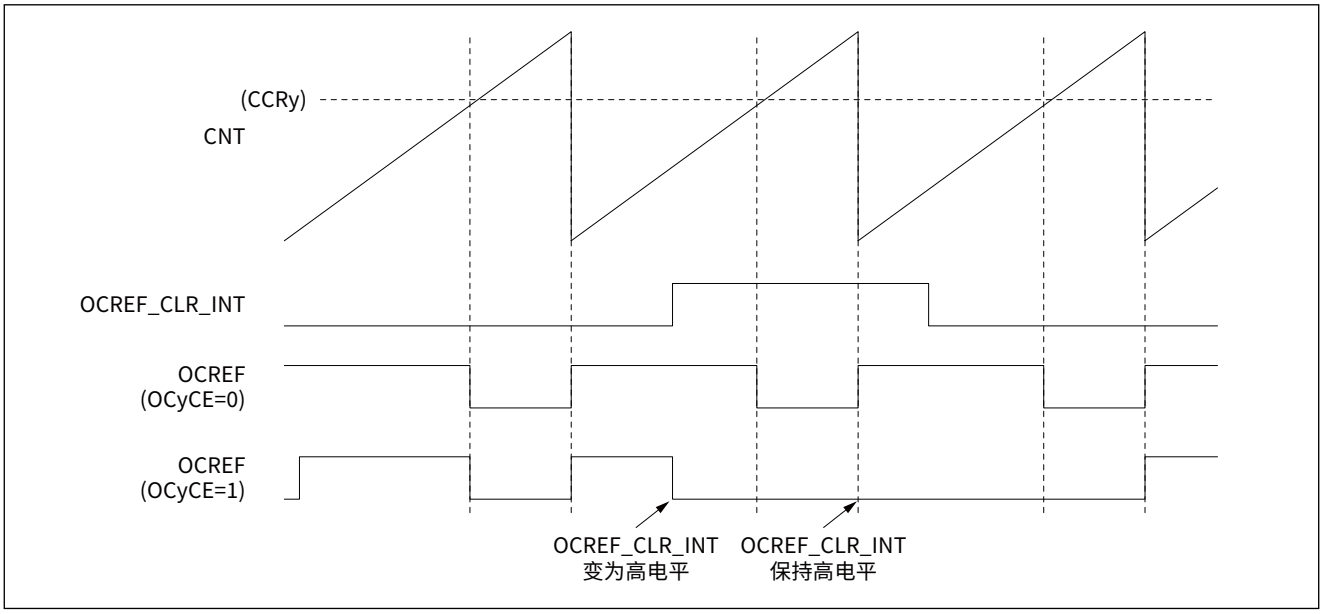
14.3.4.11 参考信号清零功能

设置 ATIM_CCMRxCMP 寄存器的 OCyCE 位域为 1，可打开对应通道的 OCyREF 信号清零功能，即当 OCREF_CLR_INT 输入上检测到高电平时，可将 OCyREF 信号清零，OCyREF 信号将保持低电平，直到发生下一更新事件 UEV。该功能只在输出比较和 PWM 模式下可用，强制输出模式下不可用。

清零源 OCREF_CLR_INT 可选 OCREF_CLR 或 ETRF，通过从模式控制寄存器 ATIM_SMCR 的 OCCS 位域进行选择，OCREF_CLR 也有多种输入源，参见 14.9.31 ATIM_AF2 复用功能选项寄存器 2 的 OCRSEL 位域说明。

下图所示为 OCREF_CLR_INT 输入变为高电平时 OCyREF 信号状态：

图 14-53　OCyREF 清零示例



14.3.5 UIF 位重映射

设置 ATIM_CR1 寄存器中的 UIFREMAP 位为 1 可使能 UIF 状态位重映射功能，可强制将更新中断标志 ATIM_ISR.UIF 连续复制到 ATIM_CNT 寄存器的 UIFCPY 位域中。这样便可自动读取计数器值以及由 UIFCPY 标志发出的电位翻转条件。这可避免在后台任务（计数器读）和中断（更新中断）之间共享处理时产生竞争条件，从而简化计算。

UIF 和 UIFCPY 标志使能之间没有延迟。

当 ATIM_CR1.UIFREMAP 为 0 时，UIFCPY 位域保留，读为 0。

14.3.6 定时器级联 ITR

通过 ITR 信号可以实现定时器级联，即将主定时器的触发输出（TRGO）连接到从定时器的 ITR 输入，进而连接到从定时器的触发输入（TRGI）。ITR 级联可实现多种功能，如将一个定时器用作另一个定时器的预分频器、主定时器对从定时器的计数器执行复位、使能或提供时钟等功能。

ATIM 的级联输入 ITR 的来源为 BTIM 和 GTIM 的触发输出信号（TRGO）以及 UART 的 TXD/RXD 信号，可通过从模式控制寄存器 ATIM_SMCR 的 TS 位域来选择，具体配置如下表所示：

表 14-15 ITR 信号来源

ATIM_SMCR.TS 位域值	ITR 信号名称	ITR 信号来源
00000	ITR0	-
00001	ITR1	-
00010	ITR2	GTIM1_Trgo
00011	ITR3	GTIM2_Trgo
01000	ITR4	UART1_TXD
01001	ITR5	UART2_TXD
01010	ITR6	UART3_TXD
01011	ITR7	UART1_RXD
01100	ITR8	UART2_RXD
01101	ITR9	UART3_RXD
01110	ITR10	BTIM1_Trgo
01111	ITR11	BTIM2_Trgo
10000	ITR12	BTIM3_Trgo

14.3.7 片内外设互联 ETR

ATIM 的 ETR 信号来源可以是外部 ATIM_ETR/GTIMx_ETR 引脚，也可以是片内其它外设，通过复用功能选项寄存器 ATIM_AF1 的 ETRSEL 位域进行选择。

当 ETR 信号来自 ETR 引脚输入时，具体外部输入端口可参见表 8-2 GPIO 复用功能分配表，并需通过 GPIO 复用功能寄存器（GPIOx_AFRH 和 GPIOx_AFRL）进行复用配置；当 ETR 信号来自片内其它外设时，可实现片内外设互联。

ETR 信号来源如下表所示：

表 14-16 ETR 信号来源

AF1.ETRSEL 位域值	ETR 信号来源
0000	ATIM_ETR 引脚
0001	VC1_OUT
0010	VC2_OUT
0101	ADC_AWD
1000	LVD_OUT
1001	GTIM1_ETR 引脚
1010	GTIM2_ETR 引脚
1011	UART1_TXD
1100	UART2_TXD
1101	UART3_TXD
1110	HSE_FAULT
1111	LSE_FAULT

14.4 ATIM 中断

ATIM 支持 16 个中断源，当 ATIM 中断事件发生时，中断标志位会被硬件置位，如果设置了对应的中断使能控制位，将产生中断请求。

在用户 ATIM 中断服务程序中，应查询相关 ATIM 中断标志位，以进行相应的处理，在退出中断服务程序之前，要清除该中断标志位，以避免重复进入中断服务程序。

各 ATIM 中断源的标志位、中断使能位、中断标志清除位或清除方法，如下表所示：

表 14-17 ATIM 中断控制

中断事件	中断标志位	中断使能位	中断标志清除
更新中断	ISR.UIF	IER.UIE	写 0 到 ICR.UIF
捕获 / 比较 1 中断	ISR.CC1IF	IER.CC1IE	写 0 到 ICR.CC1IF 或读 CCR1 寄存器
捕获 / 比较 2 中断	ISR.CC2IF	IER.CC2IE	写 0 到 ICR.CC2IF 或读 CCR2 寄存器
捕获 / 比较 3 中断	ISR.CC3IF	IER.CC3IE	写 0 到 ICR.CC3IF 或读 CCR3 寄存器
捕获 / 比较 4 中断	ISR.CC4IF	IER.CC4IE	写 0 到 ICR.CC4IF 或读 CCR4 寄存器
捕获 / 比较 5 中断	ISR.CC5IF	IER.CC5IE	写 0 到 ICR.CC5IF 或读 CCR5 寄存器
捕获 / 比较 6 中断	ISR.CC6IF	IER.CC6IE	写 0 到 ICR.CC6IF 或读 CCR6 寄存器
COM 中断	ISR.COMIF	IER.COMIE	写 0 到 ICR.COMIF
触发中断	ISR.TIF	IER.TIE	写 0 到 ICR.TIF
刹车中断	ISR.BIF	IER.BIE	写 0 到 ICR.BIF
刹车 2 中断	ISR.B2IF	IER.BIE	写 0 到 ICR.B2IF
系统刹车中断	ISR.SBIF	IER.BIE	写 0 到 ICR.SBIF
编码器索引中断	ISR.IDXF	IER.IDXIE	写 0 到 ICR.IDXF
编码器方向改变中断	ISR.DIRF	IER.DIRIE	写 0 到 ICR.DIRF
编码器索引错误中断	ISR.IERRF	IER.IERRIE	写 0 到 ICR.IERRF
编码器转换错误中断	ISR.TERRF	IER.TERRIE	写 0 到 ICR.TERRF

14.5 触发 ADC

定时器支持多种内部信号触发启动 ADC，如捕获比较信号、触发输出信号 TRGO 和 TRGO2，触发输出 TRGO 和 TRGO2 有多种可能的事件，分别由控制寄存器 ATIM_CR2 的 MMS 和 MMS2 位域进行选择。同时，ADC 外设需配置其外部触发启动寄存器，以选择对应触发源。

应注意，必须先使能 ADC 时钟，才能从主定时器接收事件，且从定时器接收触发信号时，不得实时更改 ADC 时钟。

14.6 调试支持

ATIM 支持在调试模式下停止或继续计数，通过调试状态定时器控制寄存器 SYSCTRL_DEBUG 的 ATIM 位域来设置。

- 设置 SYSCTRL_DEBUG.ATIM 为 1，则在调试状态时暂停 ATIM 的计数器计数。
- 设置 SYSCTRL_DEBUG.ATIM 为 0，则在调试状态时 ATIM 的计数器继续计数。

14.7 编程示例

14.7.1 外部时钟模式 1 编程示例

以下示例中，配置递增计数器在 ATIM_CH1 通道的上升沿进行计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择上升沿计数有效；

步骤 6：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 ATIM_SMCR.SMS 为 0x7，使 ATIM 工作于外部时钟模式 1；

步骤 8：设置 ATIM_IER.UIE 为 1 并配置对应 NVIC，使能更新中断；

步骤 9：配置计数器预分频器 ATIM_PSC；

步骤 10：设置期望的重载值 ATIM_ARR；

步骤 11：设置 ATIM_CR1.CEN 为 1，使能计数器；

步骤 12：当计数器溢出时，ATIM_ISR.UIF 标志位置 1，进入中断服务程序，设置 ATIM_ICR.UIF 为 0 清除该中断标志。

14.7.2 外部时钟模式 2 编程示例

以下是外部时钟模式 2+ 触发模式的编程示例，ATIM_CH1 输入出现上升沿时触发启动计数器，计数器在 ATIM_ETR 输入信号的每个上升沿计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_ETR、ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_ETR、ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参考见 8 通用输入输出端口 (GPIO)；

步骤 3：设置 ATIM_AF1.ETRSEL 为 0，选择 ETR 来源为 ATIM_ETR 引脚；

步骤 4：设置 ATIM_SMCR.ETP 为 0，选择 ETR 上升沿有效；

步骤 5：设置 ATIM_SMCR.ETPS 为 0，关闭 ETR 的预分频器；

步骤 6：设置 ATIM_SMCR.ETF 为 0，关闭 ETR 输入滤波；

步骤 7：设置 ATIM_SMCR.ECE 为 1，使能外部时钟模式 2；

步骤 8：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 9：设置 ATIM_CCMR1CAP.IC1F 为 0，关闭 TI1 输入滤波；

步骤 10：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测 TI1 上升沿；

步骤 11：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 12：设置 ATIM_SMCR.SMS 为 0x6，使 ATIM 工作于触发模式；

步骤 13：当 ATIM_CH1 输入出现上升沿时使能，计数器在 ATIM_ETR 输入信号的每个上升沿计数。

14.7.3 复位模式编程示例

以下示例中，ATIM_CH1 输入出现上升沿时重新初始化计数器，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测上升沿；

步骤 6：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 ATIM_SMCR.SMS 为 0x4，使 ATIM 工作于复位模式；

步骤 8：设置期望的重载值 ATIM_ARR；

步骤 9：设置 ATIM_CR1.CEN 为 1，使能计数器，开始正常计数；

步骤 10：当 ATIM_CH1 输入出现上升沿时，计数器清零，重新从 0 开始计数。

14.7.4 门控模式编程示例

以下示例中，ATIM_CH1 引脚输入的信号作为门控信号，控制计数器计数，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测高电平；

步骤 6：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 ATIM_SMCR.SMS 为 0x5，使 ATIM 工作于门控模式；

步骤 8：设置期望的重载值 ATIM_ARR；

步骤 9：设置 ATIM_CR1.CEN 为 1，使能计数器；

步骤 10：当 ATIM_CH1 输入为高电平时，计数器开始计数；当 ATIM_CH1 输入为低电平时，计数器停止计数。

14.7.5 触发模式编程示例

以下示例中，ATIM_CH1 输入出现上升沿时触发启动计数器，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测上升沿；

步骤 6：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 7：设置 ATIM_SMCR.SMS 为 0x6，使 ATIM 工作于触发模式；

步骤 8：设置期望的重载值 ATIM_ARR；

步骤 9：当 ATIM_CH1 输入出现上升沿时，计数器启动计数，同时 TIF 标志置 1。

14.7.6 正交编码器模式编程示例

以下示例中，配置 ATIM 工作在正交编码器模式 -x4 模式，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1、ATIM_CH2 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 和 ATIM_CH2 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口（GPIO）；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_TISEL1.TI2SEL 为 0，选择 TI2 来源为 ATIM_CH2 通道；

步骤 5：设置 ATIM_CCMR1CAP.IC1F 和 ATIM_CCMR1CAP.IC2F，配置输入滤波带宽；

步骤 6：设置 ATIM_CCER.CC1NP、ATIM_CCER.CC1P 为 0，CH1 输入不反相；

步骤 7：设置 ATIM_CCER.CC2NP、ATIM_CCER.CC2P 为 0，CH2 输入不反相；

步骤 8：设置 ATIM_SMCR.SMS 为 0x3，使 ATIM 工作于正交编码器模式 -x4 模式；

步骤 9：设置合适的重载值 ATIM_ARR；

步骤 10：设置 ATIM_CR1.CEN 为 1，使能计数器。

14.7.7 输入捕获编程示例

14.7.7.1 基本输入捕获模式

以下示例中，ATIM_CH1 输入出现上升沿时执行一次输入捕获，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置 TI1 输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测 IC1 上升沿；

步骤 6：设置 ATIM_CCMR1CAP.CC1S 为 1，CC1 通道配置为输入，IC1 映射到 TI1 上；

步骤 7：设置 ATIM_CCMR1CAP.IC1PSC 为 0，关闭 IC1 预分频器；

步骤 8：设置 ATIM_CCER.CC1E 为 1，使能输入捕获 1 模式；

步骤 9：设置 ATIM_IER.CC1IE 为 1 并配置对应 NVIC，使能输入捕获 1 中断；

步骤 10：配置计数器预分频器 ATIM_PSC；

步骤 11：设置期望的重载值 ATIM_ARR；

步骤 12：设置 ATIM_CR1.CEN 为 1，使能计数器，开始正常计数；

步骤 13：当发生捕获时，当前计数器 CNT 的值被锁存到捕获 / 比较寄存器 CCR1 中，完成一次捕获，同时 ATIM_ISR.CC1IF 标志位置 1，进入中断服务程序，设置 ATIM_ICR.CC1IF 为 0 清除该中断标志。

14.7.7.2 PWM 输入模式

以下示例测量从 ATIM_CH1 引脚输入的 PWM 信号的周期和占空比，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输入模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_TISEL1.TI1SEL 为 0，选择 TI1 来源为 ATIM_CH1 通道；

步骤 4：设置 ATIM_CCMR1CAP.IC1F，配置 TI1 输入滤波带宽；

步骤 5：设置 ATIM_CCER.CC1NP 为 0、ATIM_CCER.CC1P 为 0，选择检测 IC1 上升沿；

步骤 6：设置 ATIM_CCMR1CAP.CC1S 为 1，CC1 通道配置为输入，IC1 映射到 TI1 上；

步骤 7：设置 ATIM_CCMR1CAP.IC1PSC 为 0，关闭 IC1 预分频器；

步骤 8：设置 ATIM_CCER.CC1E 为 1，使能输入捕获 1 模式；

步骤 9：设置 ATIM_IER.CC1IE 为 1 并配置对应 NVIC，使能输入捕获 1 中断；

步骤 10：设置 ATIM_CCER.CC2NP 为 0、ATIM_CCER.CC2P 为 1，选择检测 IC2 下降沿；

步骤 11：设置 ATIM_CCMR1CAP.CC2S 为 2，CC2 通道配置为输入，IC2 映射到 TI1 上；

步骤 12：设置 ATIM_CCMR1CAP.IC2PSC 为 0，关闭 IC2 预分频器；

步骤 13：设置 ATIM_CCER.CC2E 为 1，使能输入捕获 2 模式；

步骤 14：设置 ATIM_SMCR.TS 为 0x5，选择 TRGI 输入为 TI1FP1；

步骤 15：设置 ATIM_SMCR.SMS 为 0x4，使 ATIM 工作于复位模式；

步骤 16：配置计数器预分频器 ATIM_PSC；

步骤 17：设置期望的重载值 ATIM_ARR；

步骤 18：设置 ATIM_CR1.CEN 为 1，使能计数器，开始正常计数。

14.7.8 输出比较编程示例

以下示例中，通道 CH1 对外输出设定的波形，步骤如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ATIM 为 1，打开 ATIM_CH1 引脚对应的 GPIO 时钟和 ATIM 配置时钟及工作时钟；

步骤 2：将 ATIM_CH1 引脚对应的 GPIO 配置成复用输出模式，具体寄存器配置请参见 8 通用输入输出端口(GPIO)；

步骤 3：设置 ATIM_CCER.CC1P 为 0，选择 OC1 输出有效极性；

步骤 4：设置 ATIM_CCMR1CMP.CC1S 为 0，CC1 通道配置为输出；

步骤 5：设置 ATIM_CCMR1CMP.OC1M，选择输出比较 1 模式；

步骤 6：设置 ATIM_CCER.CC1E 为 1，在相应输出引脚上输出 OC1 信号；

步骤 7：设置 ATIM_BDTR.MOE 为 1，使能主输出；

步骤 8：配置计数器预分频器 ATIM_PSC；

步骤 9：设置期望的重载值 ATIM_ARR；

步骤 10：设置期望的比较值 ATIM_CCR1；

步骤 11：设置 ATIM_IER.CC1IE 为 1 并配置对应 NVIC，使能比较 1 中断；

步骤 12：设置 ATIM_CR1.CEN 为 1，使能计数器；

步骤 13：当发生比较匹配时，ATIM_ISR.CC1IF 标志位置 1，进入中断服务程序，设置 ATIM_ICR.CC1IF 为 0 清除该中断标志。

14.8 寄存器列表

ATIM 基地址：ATIM_BASE = 0x4000 1400

表 14-18 ATIM 寄存器列表

寄存器名称	寄存器地址	寄存器描述
ATIM_CR1	ATIM_BASE + 0x00	控制寄存器 1
ATIM_CR2	ATIM_BASE + 0x04	控制寄存器 2
ATIM_SMCR	ATIM_BASE + 0x08	从模式控制寄存器
ATIM_IER	ATIM_BASE + 0x0C	中断使能寄存器
ATIM_ISR	ATIM_BASE + 0x10	中断标志寄存器
ATIM_ICR	ATIM_BASE + 0x70	中断标志清除寄存器
ATIM_EGR	ATIM_BASE + 0x14	事件生成寄存器
ATIM_CCMR1CAP	ATIM_BASE + 0x18	捕获模式寄存器 1
ATIM_CCMR1CMP	ATIM_BASE + 0x18	比较模式寄存器 1
ATIM_CCMR2CAP	ATIM_BASE + 0x1C	捕获模式寄存器 2
ATIM_CCMR2CMP	ATIM_BASE + 0x1C	比较模式寄存器 2
ATIM_CCMR3CAP	ATIM_BASE + 0x50	捕获模式寄存器 3
ATIM_CCMR3CMP	ATIM_BASE + 0x50	比较模式寄存器 3
ATIM_CCER	ATIM_BASE + 0x20	捕获 / 比较使能寄存器
ATIM_CNT	ATIM_BASE + 0x24	计数寄存器
ATIM_PSC	ATIM_BASE + 0x28	预分频寄存器
ATIM_ARR	ATIM_BASE + 0x2C	自动重载寄存器
ATIM_RCR	ATIM_BASE + 0x30	重复计数寄存器
ATIM_CCR1	ATIM_BASE + 0x34	捕获 / 比较寄存器 1
ATIM_CCR2	ATIM_BASE + 0x38	捕获 / 比较寄存器 2
ATIM_CCR3	ATIM_BASE + 0x3C	捕获 / 比较寄存器 3
ATIM_CCR4	ATIM_BASE + 0x40	捕获 / 比较寄存器 4
ATIM_CCR5	ATIM_BASE + 0x48	捕获 / 比较寄存器 5
ATIM_CCR6	ATIM_BASE + 0x4C	捕获 / 比较寄存器 6
ATIM_BDTR	ATIM_BASE + 0x44	刹车和死区寄存器
ATIM_DTR2	ATIM_BASE + 0x54	死区时间寄存器 2
ATIM_ECR	ATIM_BASE + 0x58	编码控制寄存器
ATIM_TISEL1	ATIM_BASE + 0x5C	TI 输入选择寄存器 1
ATIM_TISEL2	ATIM_BASE + 0x6C	TI 输入选择寄存器 2

寄存器名称	寄存器地址	寄存器描述
ATIM_AF1	ATIM_BASE + 0x60	复用功能选项寄存器 1
ATIM_AF2	ATIM_BASE + 0x64	复用功能选项寄存器 2

14.9 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

14.9.1 ATIM_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11	UIFREMAP	RW	UIF 状态位重映射配置 0：禁止重映射，ATIM_CNT[31] 保持为 0 1：使能重映射，ATIM_CNT[31] 等效于 ATIM_ISR.UIF
10	RFU	-	保留位，请保持默认值
9:8	CKD	RW	配置 PCLK 时钟与死区发生器以及数字滤波器（ETR、TIy）所使用的死区及采样时钟（f_{DTS}）之间的分频比 00：$f_{DTS} = f_{PCLK}/1$ 01：$f_{DTS} = f_{PCLK}/2$ 10：$f_{DTS} = f_{PCLK}/4$
7	ARPE	RW	自动重载预装载使能 0：禁止 ARR 寄存器缓冲 1：使能 ARR 寄存器缓冲
6:5	CMS	RW	计数器对齐模式配置 00：边沿对齐模式：由 CR1.DIR 位控制计数器递增或递减计数。 01：中心对齐模式 1：计数器交替进行递增计数和递减计数。仅当计数器递减计数时，输出比较中断标志才置 1。 10：中心对齐模式 2：计数器交替进行递增计数和递减计数。仅当计数器递增计数时，输出比较中断标志才置 1。 11：中心对齐模式 3：计数器交替进行递增计数和递减计数。计数器递增或递减计数时，输出比较中断标志均会置 1。 *注 1：只要计数器处于使能状态 (CEN=1)，就不得从边沿对齐模式切换为中心对齐模式。* *注 2：中心对齐模式的初始计数方向由 DIR 位域决定。*
4	DIR	RW/RO	计数器计数方向配置 0：递增计数 1：递减计数 *注：当定时器工作于中心对齐模式或编码器模式时，该控制位为只读。*
3	OPM	RW	单脉冲模式使能控制 0：禁止单脉冲模式 1：使能单脉冲模式，计数器运行中接收到更新事件则自动停止（设置 CEN 为 0）

位域	名称	权限	功能描述
2	URS	RW	更新请求源 此位由软件置 1 和清零，用以选择 UEV 事件源。 0：使能 UEV 时，以下所有事件都会产生更新中断。包括： – 计数器上溢 / 下溢 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：使能 UEV 时，只有计数器上溢 / 下溢会生成更新中断。
1	UDIS	RW	更新禁止 此位由软件置 1 和清零，用以使能 / 禁止 UEV 事件生成。 0：使能 UEV。更新 (UEV) 事件可通过以下事件之一生成，然后更新影子寄存器的值： – 计数器上溢 / 下溢 – 将 UG 位置 1 – 通过从模式控制器生成的更新事件 1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRy）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。
0	CEN	RW	计数器使能 0：禁止计数器 1：使能计数器 *注 1：只有事先通过软件将 CEN 位置 1，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将 CEN 位置 1。* *注 2：在单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。*

14.9.2 ATIM_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:27	RFU	-	保留位，请保持默认值
26:25	MMSH	RW	主模式选择 为 MMS 的最高 2 位，需配合 MMS 使用
24:20	MMS2	RW	主模式选择 2 这些位可选择将发送到 ADC 以实现同步的信息 (TRGO2)。 这些位的组合请参照 MMS 位域描述。
19	OIS6N	RW	输出空闲状态 6（OC6N 输出），请参见 OIS1N 说明
18	OIS6	RW	输出空闲状态 6（OC6 输出），请参见 OIS1 说明
17	OIS5N	RW	输出空闲状态 5（OC5N 输出），请参见 OIS1N 说明
16	OIS5	RW	输出空闲状态 5（OC5 输出），请参见 OIS1 说明
15	OIS4N	RW	输出空闲状态 4（OC4N 输出），请参见 OIS1N 说明
14	OIS4	RW	输出空闲状态 4（OC4 输出），请参见 OIS1 说明
13	OIS3N	RW	输出空闲状态 3（OC3N 输出），请参见 OIS1N 说明
12	OIS3	RW	输出空闲状态 3（OC3 输出），请参见 OIS1 说明
11	OIS2N	RW	输出空闲状态 2（OC2N 输出），请参见 OIS1N 说明
10	OIS2	RW	输出空闲状态 2（OC2 输出），请参见 OIS1 说明
9	OIS1N	RW	输出空闲状态 1（OC1N 输出） 0：当 MOE=0 时，经过死区时间后 OC1N=0 1：当 MOE=0 时，经过死区时间后 OC1N=1 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位即无法修改。*
8	OIS1	RW	输出空闲状态 1（OC1 输出） 0：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）OC1=0 1：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）OC1=1 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位即无法修改。*
7	TI1S	RW	TI1 输入选择 0：ATIM_CH1 引脚连接到 TI1 输入 1：ATIM_CH1、CH2 和 CH3 引脚的异或组合连接到 TI1 输入

位域	名称	权限	功能描述
6:4	MMS	RW	主模式选择 这些位可选择主模式下将要发送到从定时器以实现同步的信息 (TRGO)。配合 MMSH 使用，这些位的组合如下： 00000：复位——ATIM_EGR 寄存器中的 UG 位用作触发输出 (TRGO)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。 00001：使能——计数器使能信号 CNT_EN 用作触发输出 (TRGO)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑与运算组合而成。当计数器使能信号由触发输入控制时，TRGO 上会存在延迟，选择主 / 从模式时除外（请参见 14.9.3 ATIM_SMCR 从模式控制寄存器的 MSM 位的说明）。 00010: 更新——选择更新事件作为触发输出 (TRGO)。例如，主定时器可用作从定时器的预分频器。 00011：编码器计数时钟输出 00100：CH1 输出比较或输入捕获脉冲 00101：CH2 输出比较或输入捕获脉冲 00110：CH3 输出比较或输入捕获脉冲 00111：CH4 输出比较或输入捕获脉冲 01000：CH5 输出比较或输入捕获脉冲 01001：CH6 输出比较或输入捕获脉冲 01010：OC1REF 信号 01011：OC2REF 信号 01100：OC3REF 信号 01101：OC4REF 信号 01110：OC5REF 信号 01111：OC6REF 信号 10000：OC1REFC 信号 10001：OC2REFC 信号 10010：OC3REFC 信号 10011：OC4REFC 信号 10100：OC5REFC 信号 10101：OC6REFC 信号 10110：OC3REFC 上升沿或下降沿 10111：OC4REFC 上升沿或下降沿 11000：OC5REFC 上升沿或下降沿 11001：OC6REFC 上升沿或下降沿 11010：OC4REFC 上升沿或 OC5REFC 上升沿 11011：OC4REFC 上升沿或 OC5REFC 下降沿 11100：OC4REFC 上升沿或 OC6REFC 上升沿 11101：OC4REFC 上升沿或 OC6REFC 下降沿 11110：OC5REFC 上升沿或 OC6REFC 上升沿 11111：OC5REFC 上升沿或 OC6REFC 下降沿 *注：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。*

位域	名称	权限	功能描述
3	RFU	-	保留位，请保持默认值
2	CCUS	RW	捕获 / 比较控制更新选择 0：如果捕获 / 比较控制位进行预装载 (CCPC=1)，仅通过将 COMG 位置 1 来对这些位进行更新 1：如果捕获 / 比较控制位进行预装载 (CCPC=1)，可通过将 COMG 位置 1 或 TRGI 的上升沿对这些位进行更新。 *注：此位仅对具有互补输出的通道有效。*
1	RFU	-	保留位，请保持默认值
0	CCPC	RW	捕获 / 比较预装载控制 0：CCyE、CCyNE 和 OCyM 位未进行预装载 1：CCyE、CCyNE 和 OCyM 位进行了预装载，写入这些位后，仅当发生换向事件 (COM)（COMG 位置 1 或在 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。 *注：此位仅对具有互补输出的通道有效。*

14.9.3 ATIM_SMCR 从模式控制寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:26	RFU	-	保留位，请保持默认值
25	SMSPS	RW	预加载源 此位选择 SMS[3:0] 位字段从预装载寄存器转移到有效寄存器的触发源 0：传输由计时器的更新事件触发 1：传输由索引事件触发
24	SMSPE	RW	预加载使能 此位选择 SMS[3:0] 位是否预装载 0：禁止预装载 1：使能预装载
23:22	RFU	-	保留位，请保持默认值
21:20	TSH	RW	触发选择 为 TS 的最高位，需配合 TS 使用
19:17	RFU	-	保留位，请保持默认值
16	SMSH	RW	从模式选择 为 SMS 的最高位，需配合 SMS 使用
15	ETP	RW	外部触发极性 此位可选择将 ETR 还是 ETR 的反相用于触发操作 0：ETR 未反相，高电平或上升沿有效 1：ETR 反相，低电平或下降沿有效

位域	名称	权限	功能描述
14	ECE	RW	外部时钟使能 此位可使能外部时钟模式 2 0：禁止外部时钟模式 2 1：使能外部时钟模式 2，计数器时钟由 ETRF 信号的任意有效边沿提供。 *注 1：将 ECE 位置 1 与选择外部时钟模式 1 并将 TRGI 连接到 ETRF（SMS=111 且 TS=00111）具有相同效果。* *注 2：外部时钟模式 2 可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下 TRGI 不得连接 ETRF（TS 位不得为 00111）。* *注 3：如果同时使能外部时钟模式 1 和外部时钟模式 2，则外部时钟输入为 ETRF。*
13:12	ETPS	RW	外部触发预分频器 外部触发信号 ETRP 频率不得超过 PCLK 频率的 1/4。可通过使能预分频器来降低 ETRP 频率。这种方法在输入快速外部时钟时非常有用。 00：预分频器关闭 01：2 分频 ETRP 频率 10：4 分频 ETRP 频率 11：8 分频 ETRP 频率
11:8	ETF	RW	外部触发滤波器 此位域可定义 ETRP 信号的采样频率和适用于 ETRP 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿： 0000：无滤波器，按 f_{DTS} 频率进行采样 0001：$f_{SAMPLING}=f_{DTS}/1$，N=2 0010：$f_{SAMPLING}=f_{DTS}/1$，N=4 0011：$f_{SAMPLING}=f_{DTS}/1$，N=8 0100：$f_{SAMPLING}=f_{DTS}/2$，N=6 0101：$f_{SAMPLING}=f_{DTS}/2$，N=8 0110：$f_{SAMPLING}=f_{DTS}/4$，N=6 0111：$f_{SAMPLING}=f_{DTS}/4$，N=8 1000：$f_{SAMPLING}=f_{DTS}/8$，N=6 1001：$f_{SAMPLING}=f_{DTS}/8$，N=8 1010：$f_{SAMPLING}=f_{DTS}/16$，N=5 1011：$f_{SAMPLING}=f_{DTS}/16$，N=6 1100：$f_{SAMPLING}=f_{DTS}/16$，N=8 1101：$f_{SAMPLING}=f_{DTS}/32$，N=5 1110：$f_{SAMPLING}=f_{DTS}/32$，N=6 1111：$f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	主 / 从模式 0：不执行任何操作 1：当前定时器的触发输入事件 (TRGI) 的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 TRGO）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

位域	名称	权限	功能描述
6:4	TS	RW	触发输入 TRGI 选择 此位域可选择将要用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测器 (TI1F_ED) 00101：滤波后的定时器输入 1(TI1FP1) 00110：滤波后的定时器输入 2(TI2FP2) 00111：外部触发输入 (ETRF) 01000：内部触发 4 (ITR4) 01001：内部触发 5 (ITR5) 01010：内部触发 6 (ITR6) 01011：内部触发 7 (ITR7) 01100：内部触发 8 (ITR8) 01101：内部触发 9 (ITR9) 01110：内部触发 10 (ITR10) 01111：内部触发 11 (ITR11) 10000：内部触发 12 (ITR12) 其他值：保留 *注 1：这些位只能在未使用的情况下（例如，SMS=0000 时）进行更改，以避免转换时出现错误的边沿检测。* *注 2：其他位位于同一寄存器的位 21、20，需配合 TS 使用。* *注 3：具体 ITRx 来源请参见表 14-15 ITR 信号来源。*
3	OCCS	RW	OCREF 清零选择 该位用于选择 OCREF 清零源。 0：OCREF_CLR_INT 连接到 OCREF_CLR 输入 1：OCREF_CLR_INT 连接到 ETRF *注：OCREF_CLR 输入源请参见 14.9.31 ATIM_AF2 复用功能选项寄存器 2 的 OCRSEL 位域说明。*

位域	名称	权限	功能描述
2:0	SMS	RW	从模式选择 选择外部信号时，触发信号 (TRGI) 的有效边沿与外部输入上所选的极性相关（请参见输入相关控制寄存器说明）。 0000：禁止从模式——如果 CEN=“1”，预分频器时钟直接由内部时钟提供。 0001：正交编码器模式——×2 模式，计数器根据 TI2FP2 电平在 TI1FP1 边沿递增 / 递减计数。 0010：正交编码器模式——×2 模式，计数器根据 TI1FP1 电平在 TI2FP2 边沿递增 / 递减计数。 0011：正交编码器模式——×4 模式，计数器在 TI1FP1 和 TI2FP2 的边沿计数，计数的方向取决于另外一个输入的电平。 0100: 复位模式——在出现所选触发输入 (TRGI) 上升沿时，重新初始化计数器并生成一个寄存器更新事件。 0101：门控模式——触发输入 (TRGI) 为高电平时使能计数器时钟。只要触发输入变为低电平，计数器立即停止计数（但不复位）。计数器的启动和停止都被控制。 0110：触发模式——触发信号 (TRGI) 出现上升沿时启动计数器（但不复位）。只控制计数器的启动。 0111：外部时钟模式 1——由所选触发信号 (TRGI) 的上升沿提供计数器时钟。 1000：组合复位 + 触发模式——在出现所选触发输入 (TRGI) 上升沿时，重新初始化计数器，生成一个寄存器更新事件并启动计数器。 1001：组合门控 + 复位模式——当触发输入 (TRGI) 为高电平时计数器被使能并开始计数，当触发输入变为低电平时计数器停止计数并被复位，在此模式计数器的启动和停止都被控制。 1010：编码模式——时钟 + 方向，×2 模式。 1011：编码模式——时钟 + 方向，×1 模式，TI2FP2 的边沿极性由 CC2P 设置。 1100：编码模式——带方向时钟，×2 模式。 1101：编码模式——带方向时钟，×1 模式，TI1FP1 和 TI2FP2 的边沿极性由 CC1P 和 CC2P 设置。 1110：正交编码模式——×1 模式，仅计数 TI1FP1 边沿，其边沿极性由 CC1P 设置。 1111：正交编码模式——×1 模式，仅计数 TI2FP2 边沿，其边沿极性由 CC2P 设置。 *注 1：如果将 TI1F_ED 选作触发输入 (TS=00100)，则不得使用门控模式。实际上，TI1F 每次转换时，TI1F_ED 都输出 1 个脉冲，而门控模式检查的则是触发信号的电平。* *注 2：必须先使能接收 TRGO 或 TRGO2 信号的从外设（定时器、ADC 等）的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改时钟频率（预分频器）。* *注 3：从模式只有配置成 0100、0101、0110、1000、1001 时，有 TIE 中断标志，其余从模式没有 TIE 中断标志。*

14.9.4 ATIM_IER 中断使能寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRIE	RW	编码器转换错误中断使能控制 0：禁止转换错误中断 1：使能转换错误中断
22	IERRIE	RW	编码器索引错误中断使能控制 0：禁止索引错误中断 1：使能索引错误中断
21	DIRIE	RW	编码器方向改变中断使能控制 0：禁止方向改变中断 1：使能方向改变中断
20	IDXIE	RW	编码器索引中断使能控制 0：禁止索引中断 1：使能索引中断
19:18	RFU	-	保留位，请保持默认值
17	CC6IE	RW	捕获 / 比较 6 中断使能 0：禁止捕获 / 比较 6 中断 1：使能捕获 / 比较 6 中断
16	CC5IE	RW	捕获 / 比较 5 中断使能 0：禁止捕获 / 比较 5 中断 1：使能捕获 / 比较 5 中断
15:8	RFU	-	保留位，请保持默认值
7	BIE	RW	刹车中断使能 0：禁止刹车中断 1：使能刹车中断
6	TIE	RW	触发中断使能 0：禁止触发中断 1：使能触发中断 *注：工作在从模式时，该位只在从模式配置为 0100、0101、0110、1000、1001 时有效。*
5	COMIE	RW	COM 中断使能 0：禁止 COM 中断 1：使能 COM 中断
4	CC4IE	RW	捕获 / 比较 4 中断使能 0：禁止捕获 / 比较 4 中断 1：使能捕获 / 比较 4 中断
3	CC3IE	RW	捕获 / 比较 3 中断使能 0：禁止捕获 / 比较 3 中断 1：使能捕获 / 比较 3 中断

位域	名称	权限	功能描述
2	CC2IE	RW	捕获 / 比较 2 中断使能 0：禁止捕获 / 比较 2 中断 1：使能捕获 / 比较 2 中断
1	CC1IE	RW	捕获 / 比较 1 中断使能 0：禁止捕获 / 比较 1 中断 1：使能捕获 / 比较 1 中断
0	UIE	RW	更新中断使能 0：禁止更新中断 1：使能更新中断

14.9.5 ATIM_ISR 中断标志寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRF	RO	编码器转换错误中断标志 0：未发生转换错误 1：已发生转换错误
22	IERRF	RO	编码器索引错误中断标志 0：未发生索引错误 1：已发生索引错误
21	DIRF	RO	编码器方向改变中断标志 0：未发生编码方向改变 1：已发生编码方向改变
20	IDXF	RO	编码器索引中断标志 0：未发生索引标志 1：已发生索引标志
19	CC6OF	RO	捕获 / 比较 6 重复捕获标志，请参见 CC1OF 说明
18	CC5OF	RO	捕获 / 比较 5 重复捕获标志，请参见 CC1OF 说明
17	CC6IF	RO	捕获 / 比较 6 中断标志，请参见 CC1IF 说明
16	CC5IF	RO	捕获 / 比较 5 中断标志，请参见 CC1IF 说明
15:14	RFU	-	保留位，请保持默认值
13	SBIF	RO	系统刹车中断标志 只要系统刹车输入变为有效状态，此标志便由硬件置 1。系统刹车输入无效后可通过软件对其清零。 此标志必须复位以使 PWM 重新开始工作。 0：未发生系统刹车事件。 1：在系统刹车输入上检测到有效电平。如果 ATIM_IER 寄存器中 BIE=1，则会生成中断。
12	CC4OF	RO	捕获 / 比较 4 重复捕获标志，请参见 CC1OF 说明
11	CC3OF	RO	捕获 / 比较 3 重复捕获标志，请参见 CC1OF 说明
10	CC2OF	RO	捕获 / 比较 2 重复捕获标志，请参见 CC1OF 说明
9	CC1OF	RO	捕获 / 比较 1 重复捕获标志 仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。 0：未检测到重复捕获 1：ATIM_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1

位域	名称	权限	功能描述
8	B2IF	RO	刹车 2 中断标志 只要刹车 2 输入变为有效状态，此标志便由硬件置 1。刹车 2 输入无效后可通过软件对其清零。 0：未发生刹车 2 事件。 1：在刹车 2 输入上检测到有效电平。如果 ATIM_IER 寄存器中 BIE=1，则会生成中断。
7	BIF	RO	刹车中断标志 只要刹车输入变为有效状态，此标志便由硬件置 1。刹车输入无效后可通过软件对其清零。 0：未发生刹车事件。 1：在刹车输入上检测到有效电平。如果 ATIM_IER 寄存器中 BIE=1，则会生成中断。
6	TIF	RO	触发中断标志 在除门控模式以外的所有模式下，当使能从模式控制器后在 TRGI 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动和停止时置 1。该标志需要通过软件清零。 0：未发生触发事件 1：触发中断挂起
5	COMIF	RO	COM 中断标志 此标志在发生 COM 事件时（捕获 / 比较控制位 CCyE、CCyNE 和 OCyM 已更新时）由硬件置 1。但需要通过软件清零。 0：未发生 COM 事件 1：COM 中断挂起
4	CC4IF	RO	捕获 / 比较 4 中断标志，请参见 CC1IF 说明
3	CC3IF	RO	捕获 / 比较 3 中断标志，请参见 CC1IF 说明
2	CC2IF	RO	捕获 / 比较 2 中断标志，请参见 CC1IF 说明
1	CC1IF	RO	捕获 / 比较 1 中断标志 **如果通道 CC1 配置为输出：** 当计数器与比较值匹配时，此标志由硬件置 1，中心对齐模式下除外（请参见 14.9.1 ATIM_CR1 控制寄存器 1 的 CMS 位说明）。但需要通过软件清零。 0：不匹配。 1：ATIM_CNT 计数器的值与 ATIM_CCR1 寄存器的值匹配。当 ATIM_CCR1 的值大于 ATIM_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和中心对齐模式下）或下溢（递减计数模式下）时变为高。 **如果通道 CC1 配置为输入：** 此位将在发生捕获事件时由硬件置 1。通过软件或读取 ATIM_CCR1 寄存器将该位清零。 0：未发生输入捕获事件 1：ATIM_CCR1 寄存器中已捕获到计数器值（IC1 上已检测到与所选极性匹配的边沿）

位域	名称	权限	功能描述
0	UIF	RO	更新中断标志 该位在发生更新事件时通过硬件置 1。但需要通过软件清零。 0：未发生更新 1：更新中断挂起 该位在以下情况下更新寄存器时由硬件置 1： – ATIM_CR1 寄存器中的 UDIS=0，并且重复计数器值下溢时（重复计数器 =0 时更新）。 – ATIM_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 ATIM_EGR 寄存器中的 UG 位重新初始化 CNT 时。 – ATIM_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时。

14.9.6 ATIM_ICR 中断标志清除寄存器

Address offset: 0x70 Reset value: 0x00FF 3FFF

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	TERRF	R1W0	编码器转换错误中断标志清除 0：清除编码器转换错误中断标志 1：无功能
22	IERRF	R1W0	编码器索引错误中断标志清除 0：清除编码器索引错误中断标志 1：无功能
21	DIRF	R1W0	编码器方向改变中断标志清除 0：清除编码器方向改变中断标志 1：无功能
20	IDXF	R1W0	编码器索引中断标志清除 0：清除编码器索引中断标志 1：无功能
19	CC6OF	R1W0	捕获 / 比较 6 重复捕获标志清除，请参见 CC1OF 说明
18	CC5OF	R1W0	捕获 / 比较 5 重复捕获标志清除，请参见 CC1OF 说明
17	CC6IF	R1W0	捕获 / 比较 6 中断标志清除，请参见 CC1IF 说明
16	CC5IF	R1W0	捕获 / 比较 5 中断标志清除，请参见 CC1IF 说明
15:14	RFU	-	保留位，请保持默认值
13	SBIF	R1W0	系统刹车中断标志清除 0：清除系统刹车中断标志 1：无功能
12	CC4OF	R1W0	捕获 / 比较 4 重复捕获标志清除，请参见 CC1OF 说明
11	CC3OF	R1W0	捕获 / 比较 3 重复捕获标志清除，请参见 CC1OF 说明

位域	名称	权限	功能描述
10	CC2OF	R1W0	捕获 / 比较 2 重复捕获标志清除，请参见 CC1OF 说明
9	CC1OF	R1W0	捕获 / 比较 1 重复捕获标志清除 0：清除捕获 / 比较 1 的重复捕获标志 1：无功能
8	B2IF	R1W0	刹车 2 中断标志清除 0：清除刹车 2 中断标志 1：无功能
7	BIF	R1W0	刹车中断标志清除 0：清除刹车中断标志 1：无功能
6	TIF	R1W0	触发中断标志清除 0：清除触发中断标志 1：无功能
5	COMIF	R1W0	COM 中断标志清除 0：清除 COM 中断标志 1：无功能
4	CC4IF	R1W0	捕获 / 比较 4 中断标志清除，请参见 CC1IF 说明
3	CC3IF	R1W0	捕获 / 比较 3 中断标志清除，请参见 CC1IF 说明
2	CC2IF	R1W0	捕获 / 比较 2 中断标志清除，请参见 CC1IF 说明
1	CC1IF	R1W0	捕获 / 比较 1 中断标志清除 0：清除捕获 / 比较 1 中断标志 1：无功能
0	UIF	R1W0	更新中断标志清除 0：清除更新中断标志 1：无功能

14.9.7 ATIM_EGR 事件生成寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17	CC6G	RW	捕获 / 比较 6 生成，请参见 CC1G 说明
16	CC5G	RW	捕获 / 比较 5 生成，请参见 CC1G 说明
15:9	RFU	-	保留位，请保持默认值
8	B2G	RW	刹车 2 生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：生成刹车 2 事件。MOE 位清零且 B2IF 标志置 1。使能后可发生相关中断。
7	BG	RW	刹车生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：生成刹车事件。MOE 位清零且 BIF 标志置 1。使能后可发生相关中断。
6	TG	RW	触发生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：ATIM_ISR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件。
5	COMG	RW	捕获 / 比较控制更新生成 该位可通过软件置 1，并由硬件自动清零 0：不执行任何操作 1：CCPC 位置 1 时，可更新 CCyE、CCyNE 和 OCyM 位 *注：此位仅对具有互补输出的通道有效。*
4	CC4G	RW	捕获 / 比较 4 生成，请参见 CC1G 说明
3	CC3G	RW	捕获 / 比较 3 生成，请参见 CC1G 说明
2	CC2G	RW	捕获 / 比较 2 生成，请参见 CC1G 说明
1	CC1G	RW	捕获 / 比较 1 生成 此位由软件置 1 以生成事件，并由硬件自动清零。 0：不执行任何操作 1：通道 1 上生成捕获 / 比较事件 **如果通道 CC1 配置为输出：** 使能时，CC1IF 标志置 1 并发送相应的中断请求。 **如果通道 CC1 配置为输入：** ATIM_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断请求。如果 CC1IF 标志已为高，CC1OF 标志将置 1。

位域	名称	权限	功能描述
0	UG	RW	更新生成 该位可通过软件置 1，并由硬件自动清零。 0：不执行任何操作 1：重新初始化计数器并生成寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或递增计数模式，计数器将清零；如果选择递减计数模式，计数器将使用自动重载值 ATIM_ARR。

14.9.8 ATIM_CCMR1CAP 捕获模式寄存器 1

Address offset: 0x18 Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR1CMP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	IC2F	RW	输入捕获 2 滤波器，请参见 IC1F 说明
11:10	IC2PSC	RW	输入捕获 2 预分频器，请参见 IC1PSC 说明
9:8	CC2S	RW	捕获 / 比较 2 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 上 10：CC2 通道配置为输入，IC2 映射到 TI1 上 11：CC2 通道配置为输入，IC2 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC2E=0），才可向 CC2S 位写入数据。*
7:4	IC1F	RW	输入捕获 1 滤波器 此位域可定义 TI1 输入的采样频率和适用于 TI1 的数字滤波器的采样长度。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿。 0000：无滤波器，按 f_{DTS} 频率进行采样 0001：$f_{SAMPLING}=f_{PCLK}$，N=2 0010：$f_{SAMPLING}=f_{PCLK}$，N=4 0011：$f_{SAMPLING}=f_{PCLK}$，N=8 0100：$f_{SAMPLING}=f_{DTS}/2$，N=6 0101：$f_{SAMPLING}=f_{DTS}/2$，N=8 0110：$f_{SAMPLING}=f_{DTS}/4$，N=6 0111：$f_{SAMPLING}=f_{DTS}/4$，N=8 1000：$f_{SAMPLING}=f_{DTS}/8$，N=6 1001：$f_{SAMPLING}=f_{DTS}/8$，N=8

位域	名称	权限	功能描述
7:4（续）	IC1F	RW	1010：$f_{SAMPLING}=f_{DTS}/16$，N=5 1011：$f_{SAMPLING}=f_{DTS}/16$，N=6 1100：$f_{SAMPLING}=f_{DTS}/16$，N=8 1101：$f_{SAMPLING}=f_{DTS}/32$，N=5 1110：$f_{SAMPLING}=f_{DTS}/32$，N=6 1111：$f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC	RW	输入捕获 1 预分频器 此位域定义 CC1 输入 (IC1) 的预分频比。只要 CC1E=0（ATIM_CCER 寄存器），预分频器便立即复位。 00：无预分频器，捕获输入上每检测到一个边沿便执行捕获 01：每发生 2 个事件便执行一次捕获 10：每发生 4 个事件便执行一次捕获 11：每发生 8 个事件便执行一次捕获
1:0	CC1S	RW	捕获 / 比较 1 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 上 10：CC1 通道配置为输入，IC1 映射到 TI2 上 11：CC1 通道配置为输入，IC1 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC1E=0），才可向 CC1S 位写入数据。*

14.9.9 ATIM_CCMR1CMP 比较模式寄存器 1

Address offset: 0x18 Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR1CAP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	OC2MH	RW	配合 OC2M 使用，为 OC2M 的最高位
23:17	RFU	-	保留位，请保持默认值
16	OC1MH	RW	配合 OC1M 使用，为 OC1M 的最高位
15	OC2CE	RW	输出比较 2 清零使能，请参见 OC1CE 说明
14:12	OC2M	RW	输出比较 2 模式，请参见 OC1M 说明
11	OC2PE	RW	输出比较 2 预装载使能，请参见 OC1PE 说明
10	OC2FE	RW	输出比较 2 快速使能，请参见 OC1FE 说明
9:8	CC2S	RW	捕获 / 比较 2 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 上 10：CC2 通道配置为输入，IC2 映射到 TI1 上 11：CC2 通道配置为输入，IC2 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC2E=0），才可向 CC2S 位写入数据。*
7	OC1CE	RW	输出比较 1 清零使能 0：OC1REF 不受 OCREF_CLR_INT 输入影响 1：OCREF_CLR_INT 输入上检测到高电平时，OC1REF 立即清零
6:4	OC1M	RW	输出比较 1 模式 这些位定义提供 OC1 和 OC1N 的输出参考信号 OC1REF 的行为。OC1REF 为高电平有效，而 OC1 和 OC1N 的有效电平则分别取决于 CC1P 位和 CC1NP 位。 0000：冻结 —— 输出比较寄存器 ATIM_CCR1 与计数器 ATIM_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。 0001：将通道 1 设置为匹配时输出有效电平。当计数器 ATIM_CNT 与捕获 / 比较寄存器 1(ATIM_CCR1) 匹配时，OC1REF 信号强制变为高电平。 0010：将通道 1 设置为匹配时输出无效电平。当计数器 ATIM_CNT 与捕获 / 比较寄存器 1(ATIM_CCR1) 匹配时，OC1REF 信号强制变为低电平。 0011：翻转——ATIM_CNT=ATIM_CCR1 时，OC1REF 发生翻转。 0100：强制变为无效电平——OC1REF 强制变为低电平。 0101：强制变为有效电平——OC1REF 强制变为高电平。

6:4（续）	OC1M	RW	0110：PWM 模式 1——在递增计数模式下，只要 ATIM_CNT<ATIM_CCR1，通道 1 便为有效状态 (OC1REF=1)，否则为无效状态 (OC1REF=0)。在递减计数模式下，只要 ATIM_CNT>ATIM_CCR1，通道 1 便为无效状态 (OC1REF=0)，否则为有效状态 (OC1REF=1)。 0111：PWM 模式 2——在递增计数模式下，只要 ATIM_CNT<ATIM_CCR1，通道 1 便为无效状态 (OC1REF=0)，否则为有效状态 (OC1REF=1)。在递减计数模式下，只要 ATIM_CNT>ATIM_CCR1，通道 1 便为有效状态 (OC1REF=1)，否则为无效状态 (OC1REF=0)。 1000：可再触发 OPM 模式 1——在递增计数模式下，通道为有效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为有效状态。在递减计数模式下，通道为无效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 1 下进行比较，通道会在下一次更新时再次变为无效状态。 1001：可再触发 OPM 模式 2——在递增计数模式下，通道为无效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模式下，通道为有效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在 PWM 模式 2 下进行比较，通道会在下一次更新时再次变为有效状态。 1010：保留 1011：计数方向输出，OC1REF 信号为 DIR 位的计数方向信号。 1100：组合 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑或运算结果。 1101：组合 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。OC1REFC 是 OC1REF 和 OC2REF 的逻辑与运算结果。 1110：不对称 PWM 模式 1——OC1REF 与在 PWM 模式 1 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。 1111：不对称 PWM 模式 2——OC1REF 与在 PWM 模式 2 下的行为相同。计数器递增计数时，OC1REFC 输出 OC1REF；计数器递减计数时，OC1REFC 输出 OC2REF。 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S=“00”（通道配置为输出），这些位即无法修改。* *注 2：在 PWM 模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCyREF 电平才会发生更改。* *注 3：此位域将在具有互补输出的通道上进行预装载。如果 ATIM_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC1M 有效位才会从预装载位获取新值。*

3	OC1PE	RW	输出比较 1 预装载使能 0：禁止与 ATIM_CCR1 相关的预装载寄存器。可随时向 ATIM_CCR1 写入数据，写入后将立即使用新值。 1：使能与 ATIM_CCR1 相关的预装载寄存器。可读 / 写访问预装载寄存器。ATIM_CCR1 预装载值在每次生成更新事件时都会装载到有效寄存器中。 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S=“00”（通道配置为输出），这些位即无法修改。* *注 2：只有单脉冲模式下才可在未验证预装载寄存器的情况下使用 PWM 模式（ATIM_CR1 寄存器中的 OPM 位置 1）。其他情况下则无法保证该行为。*
2	OC1FE	RW	输出比较 1 快速使能 此位用于加快触发输入事件对 CC 输出的影响。 0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。 1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCyFE 才会起作用。
1:0	CC1S	RW	捕获 / 比较 1 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 上 10：CC1 通道配置为输入，IC1 映射到 TI2 上 11：CC1 通道配置为输入，IC1 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC1E=0），才可向 CC1S 位写入数据。*

14.9.10 ATIM_CCMR2CAP 捕获模式寄存器 2

Address offset: 0x1C Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR2CMP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	IC4F	RW	输入捕获 4 滤波器，请参见 IC1F 说明
11:10	IC4PSC	RW	输入捕获 4 预分频器，请参见 IC1PSC 说明
9:8	CC4S	RW	捕获 / 比较 4 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 上 10：CC4 通道配置为输入，IC4 映射到 TI3 上 11：CC4 通道配置为输入，IC4 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC4E=0），才可向 CC4S 位写入数据。*
7:4	IC3F	RW	输入捕获 3 滤波器，请参见 IC1F 说明
3:2	IC3PSC	RW	输入捕获 3 预分频器，请参见 IC1PSC
1:0	CC3S	RW	捕获 / 比较 3 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 上 10：CC3 通道配置为输入，IC3 映射到 TI4 上 11：CC3 通道配置为输入，IC3 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC3E=0），才可向 CC3S 位写入数据。*

14.9.11 ATIM_CCMR2CMP 比较模式寄存器 2

Address offset: 0x1C Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR2CAP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	OC4MH	RW	配合 OC4M 使用，为 OC4M 的最高位
23:17	RFU	-	保留位，请保持默认值
16	OC3MH	RW	配合 OC3M 使用，为 OC3M 的最高位
15	OC4CE	RW	输出比较 4 清零使能，请参见 OC1CE 说明
14:12	OC4M	RW	输出比较 4 模式，请参见 OC1M 说明
11	OC4PE	RW	输出比较 4 预装载使能，请参见 OC1PE 说明
10	OC4FE	RW	输出比较 4 快速使能，请参见 OC1FE 说明
9:8	CC4S	RW	捕获 / 比较 4 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 上 10：CC4 通道配置为输入，IC4 映射到 TI3 上 11：CC4 通道配置为输入，IC4 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC4E=0），才可向 CC4S 位写入数据。*
7	OC3CE	RW	输出比较 3 清零使能，请参见 OC1CE 说明
6:4	OC3M	RW	输出比较 3 模式，请参见 OC1M 说明
3	OC3PE	RW	输出比较 3 预装载使能，请参见 OC1PE 说明
2	OC3FE	RW	输出比较 3 快速使能，请参见 OC1FE 说明
1:0	CC3S	RW	捕获 / 比较 3 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 上 10：CC3 通道配置为输入，IC3 映射到 TI4 上 11：CC3 通道配置为输入，IC3 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC3E=0），才可向 CC3S 位写入数据。*

14.9.12 ATIM_CCMR3CAP 捕获模式寄存器 3

Address offset: 0x50 Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR3CMP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	IC6F	RW	输入捕获 6 滤波器，请参见 IC1F 说明
11:10	IC6PSC	RW	输入捕获 6 预分频器，请参见 IC1PSC 说明
9:8	CC6S	RW	捕获 / 比较 6 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC6 通道配置为输出 01：CC6 通道配置为输入，IC6 映射到 TI6 上 10：CC6 通道配置为输入，IC6 映射到 TI5 上 11：CC6 通道配置为输入，IC6 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC6E=0），才可向 CC6S 位写入数据。*
7:4	IC5F	RW	输入捕获 5 滤波器，请参见 IC1F 说明
3:2	IC5PSC	RW	输入捕获 5 预分频器，请参见 IC1PSC
1:0	CC5S	RW	捕获 / 比较 5 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC5 通道配置为输出 01：CC5 通道配置为输入，IC5 映射到 TI5 上 10：CC5 通道配置为输入，IC5 映射到 TI6 上 11：CC5 通道配置为输入，IC5 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC5E=0），才可向 CC5S 位写入数据。*

14.9.13 ATIM_CCMR3CMP 比较模式寄存器 3

Address offset: 0x50 Reset value: 0x0000 0000

此寄存器和 ATIM_CCMR3CAP 为同一寄存器，可用于输入捕获模式或输出比较模式。通道方向通过配置相应的 CCyS 位进行定义。此寄存器的所有其他位在输入捕获模式和输出比较模式下的功能均不同。

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	OC6MH	RW	配合 OC6M 使用，为 OC6M 的最高位
23:17	RFU	-	保留位，请保持默认值
16	OC5MH	RW	配合 OC5M 使用，为 OC5M 的最高位
15	OC6CE	RW	输出比较 6 清零使能，请参见 OC1CE 说明
14:12	OC6M	RW	输出比较 6 模式，请参见 OC1M 说明
11	OC6PE	RW	输出比较 6 预装载使能，请参见 OC1PE 说明
10	OC6FE	RW	输出比较 6 快速使能，请参见 OC1FE 说明
9:8	CC6S	RW	捕获 / 比较 6 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入映射。 00：CC6 通道配置为输出 01：CC6 通道配置为输入，IC6 映射到 TI6 上 10：CC6 通道配置为输入，IC6 映射到 TI5 上 11：CC6 通道配置为输入，IC6 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC6E=0），才可向 CC6S 位写入数据。*
7	OC5CE	RW	输出比较 5 清零使能，请参见 OC1CE 说明
6:4	OC5M	RW	输出比较 5 模式，请参见 OC1M 说明
3	OC5PE	RW	输出比较 5 预装载使能，请参见 OC1PE 说明
2	OC5FE	RW	输出比较 5 快速使能，请参见 OC1FE 说明
1:0	CC5S	RW	捕获 / 比较 5 选择 此位域定义通道方向（输入 / 输出）以及所使用的输入。 00：CC5 通道配置为输出 01：CC5 通道配置为输入，IC5 映射到 TI5 上 10：CC5 通道配置为输入，IC5 映射到 TI6 上 11：CC5 通道配置为输入，IC5 映射到 TRC 上，此模式仅在通过 TS 位（ATIM_SMCR 寄存器）选择内部触发输入时有效 *注：仅当通道关闭时（ATIM_CCER 中的 CC5E=0），才可向 CC5S 位写入数据。*

14.9.14 ATIM_CCER 捕获 / 比较使能寄存器

Address offset: 0x20 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23	CC6NP	RW	捕获 / 比较 6 互补输出极性，请参见 CC1NP 说明
22	CC6NE	RW	捕获 / 比较 6 互补输出使能，请参见 CC1NE 说明
21	CC6P	RW	捕获 / 比较 6 输出极性，请参见 CC1P 说明
20	CC6E	RW	捕获 / 比较 6 输出使能，请参见 CC1E 说明
19	CC5NP	RW	捕获 / 比较 5 互补输出极性，请参见 CC1NP 说明
18	CC5NE	RW	捕获 / 比较 5 互补输出使能，请参见 CC1NE 说明
17	CC5P	RW	捕获 / 比较 5 输出极性，请参见 CC1P 说明
16	CC5E	RW	捕获 / 比较 5 输出使能，请参见 CC1E 说明
15	CC4NP	RW	捕获 / 比较 4 互补输出极性，请参见 CC1NP 说明
14	CC4NE	RW	捕获 / 比较 4 互补输出使能，请参见 CC1NE 说明
13	CC4P	RW	捕获 / 比较 4 输出极性，请参见 CC1P 说明
12	CC4E	RW	捕获 / 比较 4 输出使能，请参见 CC1E 说明
11	CC3NP	RW	捕获 / 比较 3 互补输出极性，请参见 CC1NP 说明
10	CC3NE	EW	捕获 / 比较 3 互补输出使能，请参见 CC1NE 说明
9	CC3P	RW	捕获 / 比较 3 输出极性，请参见 CC1P 说明
8	CC3E	RW	捕获 / 比较 3 输出使能，请参见 CC1E 说明
7	CC2NP	RW	捕获 / 比较 2 互补输出极性，请参见 CC1NP 说明
6	CC2NE	RW	捕获 / 比较 2 互补输出使能，请参见 CC1NE 说明
5	CC2P	RW	捕获 / 比较 2 输出极性，请参见 CC1P 说明
4	CC2E	RW	捕获 / 比较 2 输出使能，请参见 CC1E 说明

位域	名称	权限	功能描述
3	CC1NP	RW	捕获 / 比较 1 互补输出极性 **CC1 通道配置为输出：** 0：OC1N 高电平有效 1：OC1N 低电平有效 **CC1 通道配置为输入：** 该位与 CC1P 配合使用可定义 TI1FP1/TI2FP1 极性。请参见 CC1P 说明。 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 2 或 3 且 CC1S=“00”（通道配置为输出），此位立即变为不可写状态。* *注 2：此位将在具有互补输出的通道上进行预装载。如果 ATIM_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1NP 有效位才会从预装载位获取新值。*
2	CC1NE	RW	捕获 / 比较 1 互补输出使能 0：关闭——OC1N 未激活。OC1N 电平取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位。 1：开启——在相应输出引脚上输出 OC1N 信号，具体取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位。 *注：此位将在具有互补输出的通道上进行预装载。如果 ATIM_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1NE 有效位才会从预装载位获取新值。*
1	CC1P	RW	捕获 / 比较 1 输出极性 **CC1 通道配置为输出：** 0：OC1 高电平有效 1：OC1 低电平有效 **CC1 通道配置为输入：** CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的有效极性。 00：非反相 / 上升沿触发。电路作用于 TIxFP1 的上升沿（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。 01：反相 / 下降沿触发。电路作用于 TIxFP1 的下降沿（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。 10：保留，不使用此配置。 11：非反相 / 上升沿和下降沿均触发。电路作用于 TIxFP1 的上升沿和下降沿（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 2 或 3，此位立即变为不可写状态。* *注 2：此位将在具有互补输出的通道上进行预装载。如果 ATIM_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1P 有效位才会从预装载位获取新值。*

位域	名称	权限	功能描述
0	CC1E	RW	捕获 / 比较 1 输出使能 **CC1 通道配置为输出：** 0：关闭——OC1 未激活。OC1 电平取决于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位。 1：开启——OC1 信号输出到相应的输出引脚上，具体取决于 MOE、OSSI、OSSR、OIS1、OIS1 和 CC1NE 位。 **CC1 通道配置为输入：** 此位决定了是否可以实际将计数器值捕获到输入捕获 / 比较寄存器 1(ATIM_CCR1) 中。 0：禁止捕获 1：使能捕获 *注 1：此位将在具有互补输出的通道上进行预装载。如果 ATIM_CR2 寄存器中的 CCPC 位置 1，则仅当生成换向事件时，CC1E 有效位才会从预装载位获取新值。* *注 2：使用输入捕获功能时，该位必须配置为 1。*

14.9.15 ATIM_CNT 计数寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	UIFCPY	RO	根据 ATIM_CR1 寄存器中 UIFREMAP 位域的值，本位域表示不同的含义： UIFREMAP = 0，本位域保留，读为 0 UIFREMAP = 1，本位域表示 ATIM_ISR 寄存器的 UIF 位的只读副本
30:16	RFU	-	保留位，请保持默认值
15:0	CNT	RW	计数值

14.9.16 ATIM_PSC 预分频寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	PSC	RW	预分频器值 计数器时钟频率 CK_CNT 等于 f_{CK_PSC} / (PSC[15:0] + 1)。 PSC 包含每次发生更新事件（包括计数器通过 ATIM_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到有效预分频器寄存器的值

14.9.17 ATIM_ARR 自动重载寄存器

Address offset: 0x2C Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	ARR	RW	自动重载值 ARR 为要装载到实际自动重载寄存器的值，当自动重载值为空时，计数器不工作。

14.9.18 ATIM_RCR 重复计数寄存器

Address offset: 0x30 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	REP	RW	重复计数器值 使能预装载寄存器时，用户可通过这些位设置比较寄存器的更新频率（即，从预装载寄存器向活动寄存器周期性传输数据）；使能更新中断时，也可设置更新中断的生成速率。 与 REP_CNT 相关的减计数器每次计数到 0 时，都将生成一个更新事件并且计数器从 REP 值重新开始计数。由于只有生成重复更新事件 U_RC 时，REP_CNT 才会重载 REP 值，因此在生成下一重复更新事件之前，无论向 ATIM_RCR 寄存器写入何值都无影响。 这意味着 PWM 模式下 (REP+1) 相当于： – 边沿对齐模式下的 PWM 周期数。 – 中心对齐模式下的 PWM 半周期数。

14.9.19 ATIM_CCR1 捕获 / 比较寄存器 1

Address offset: 0x34 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR1	RW	捕获 / 比较 1 值 **如果通道 CC1 配置为输出：** CCR1 为要装载到有效捕获 / 比较 1 寄存器的值（预装载值）。如果没有通过 ATIM_CCMR1CMP 寄存器中的 OC1PE 位来使能预装载功能，则该值立刻生效；否则只在发生更新事件时生效（拷贝到实际起作用的捕获 / 比较寄存器 1）。 实际捕获 / 比较寄存器中包含要与计数器 ATIM_CNT 进行比较并在 OC1 输出上发出信号的值。 **如果通道 CC1 配置为输入：** CCR1 为上一个输入捕获 1 事件 (IC1) 发生时的计数器值。只能读取 ATIM_CCR1 寄存器，无法对其进行编程。

14.9.20 ATIM_CCR2 捕获 / 比较寄存器 2

Address offset: 0x38 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR2	RW	捕获 / 比较 2 值，请参见 CCR1 说明

14.9.21 ATIM_CCR3 捕获 / 比较寄存器 3

Address offset: 0x3C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR3	RW	捕获 / 比较 3 值，请参见 CCR1 说明

14.9.22 ATIM_CCR4 捕获 / 比较寄存器 4

Address offset: 0x40 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	CCR4	RW	捕获 / 比较 4 值，请参见 CCR1 说明

14.9.23 ATIM_CCR5 捕获 / 比较寄存器 5

Address offset: 0x48 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	GC5C6	RW	CH6/CH5 组合波形配置 参见 GC5C1 说明
30	GC5C5	RW	CH5/CH5 组合波形配置 参见 GC5C1 说明
29	GC5C4	RW	CH4/CH5 组合波形配置 参见 GC5C1 说明
28	GC5C3	RW	CH3/CH5 组合波形配置 参见 GC5C1 说明
27	GC5C2	RW	CH2/CH5 组合波形配置 参见 GC5C1 说明
26	GC5C1	RW	CH1/CH5 组合波形配置 0：OC1REFC 与 OC5REF 无关 1：OC1REFC 输出 OC1REF 和 OC5REF 的逻辑与
25:16	RFU	-	保留位，请保持默认值
15:0	CCR5	RW	捕获 / 比较 5 值，请参见 CCR1 说明

14.9.24 ATIM_CCR6 捕获 / 比较寄存器 6

Address offset: 0x4C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	GC6C6	RW	CH6/CH6 组合波形配置 参见 GC6C1 说明
30	GC6C5	RW	CH5/CH6 组合波形配置 参见 GC6C1 说明
29	GC6C4	RW	CH4/CH6 组合波形配置 参见 GC6C1 说明
28	GC6C3	RW	CH3/CH6 组合波形配置 参见 GC6C1 说明
27	GC6C2	RW	CH2/CH6 组合波形配置 参见 GC6C1 说明
26	GC6C1	RW	CH1/CH6 组合波形配置 0：OC1REFC 与 OC6REF 无关 1：递增计数时 OC1REFC 输出 OC1REF；递减计数时 OC1REFC 输出 OC6REF
25:16	RFU	-	保留位，请保持默认值
15:0	CCR6	RW	捕获 / 比较 6 值，请参见 CCR1 说明

14.9.25 ATIM_BDTR 刹车和死区寄存器

Address offset: 0x44 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:26	RFU	-	保留位，请保持默认值
25	BK2P	RW	刹车 2 极性 0：刹车输入 BRK2 为低电平有效 1：刹车输入 BRK2 为高电平有效 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。* *注 2：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。*
24	BK2E	RW	刹车 2 使能 0：禁止刹车输入 BRK2 1：使能刹车输入 BRK2 *注 1：BRK2 必须只在 OSSR=OSSI=1 时使用。* *注 2：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。* *注 3：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。*

位域	名称	权限	功能描述
23:20	BK2F	RW	刹车 2 滤波器 此位域可定义 BRK2 输入的采样频率和适用于 BRK2 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿。 0000：无滤波器，BRK2 异步工作 0001：$f_{SAMPLING} = f_{PCLK}$，N=2 0010：$f_{SAMPLING} = f_{PCLK}$，N=4 0011：$f_{SAMPLING} = f_{PCLK}$，N=8 0100：$f_{SAMPLING} = f_{DTS}/2$，N=6 0101：$f_{SAMPLING} = f_{DTS}/2$，N=8 0110：$f_{SAMPLING} = f_{DTS}/4$，N=6 0111：$f_{SAMPLING} = f_{DTS}/4$，N=8 1000：$f_{SAMPLING} = f_{DTS}/8$，N=6 1001：$f_{SAMPLING} = f_{DTS}/8$，N=8 1010：$f_{SAMPLING} = f_{DTS}/16$，N=5 1011：$f_{SAMPLING} = f_{DTS}/16$，N=6 1100：$f_{SAMPLING} = f_{DTS}/16$，N=8 1101：$f_{SAMPLING} = f_{DTS}/32$，N=5 1110：$f_{SAMPLING} = f_{DTS}/32$，N=6 1111：$f_{SAMPLING} = f_{DTS}/32$，N=8 *注：编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。*
19:16	BKF	RW	刹车滤波器 此位域可定义 BRK 输入的采样频率和适用于 BRK 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿。 0000：无滤波器，BRK 异步工作 0001：$f_{SAMPLING} = f_{PCLK}$，N=2 0010：$f_{SAMPLING} = f_{PCLK}$，N=4 0011：$f_{SAMPLING} = f_{PCLK}$，N=8 0100：$f_{SAMPLING} = f_{DTS}/2$，N=6 0101：$f_{SAMPLING} = f_{DTS}/2$，N=8 0110：$f_{SAMPLING} = f_{DTS}/4$，N=6 0111：$f_{SAMPLING} = f_{DTS}/4$，N=8 1000：$f_{SAMPLING} = f_{DTS}/8$，N=6 1001：$f_{SAMPLING} = f_{DTS}/8$，N=8 1010：$f_{SAMPLING} = f_{DTS}/16$，N=5 1011：$f_{SAMPLING} = f_{DTS}/16$，N=6 1100：$f_{SAMPLING} = f_{DTS}/16$，N=8 1101：$f_{SAMPLING} = f_{DTS}/32$，N=5 1110：$f_{SAMPLING} = f_{DTS}/32$，N=6 1111：$f_{SAMPLING} = f_{DTS}/32$，N=8 *注：编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。*

位域	名称	权限	功能描述
15	MOE	RW	主输出使能 只要刹车输入（BRK 或 BRK2）为有效状态，此位便由硬件异步清零。此位由软件置 1，也可根据 AOE 位状态自动置 1。此位仅对配置为输出的通道有效。 0：响应刹车事件（2 个）。禁止 OC 和 OCN 输出 响应刹车事件或向 MOE 写入 0 时：OC 和 OCN 输出被禁止或被强制为空闲状态，具体取决于 OSSI 位。 1：如果 OC 和 OCN 输出的相应使能位（ATIM_CCER 寄存器中的 CCyE 和 CCyNE 位）均置 1，则使能 OC 和 OCN 输出。
14	AOE	RW	自动输出使能 0：MOE 只能由软件置 1 1：MOE 可由软件置 1，也可在发生下一更新事件时自动置 1（如果刹车输入 BRK 和 BRK2 有效时，此位无效） *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
13	BKP	RW	刹车极性 0：刹车输入 BRK 为低电平有效 1：刹车输入 BRK 为高电平有效 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。* *注 2：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。*
12	BKE	RW	刹车使能 该位可使能完整的刹车保护（包括内部所有源和相应的 BKIN 源）。 0：禁止刹车功能 1：使能刹车功能 *注 1：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1 后，此位即无法修改。* *注 2：对该位执行任何写操作后，都需要经过 1 个 APB 时钟周期的延迟才生效。*
11	OSSR	RW	运行模式下的关闭状态选择 此位在 MOE=1 时作用于配置为输出模式且具有互补输出的通道。如果定时器中没有互补输出，则不存在 OSSR。 0：处于无效状态时，禁止 OC/OCN 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。 1：处于无效状态时，一旦 CCyE=1 或 CCyNE=1，便使能 OC/OCN 输出并将其设为无效电平（输出仍由定时器控制）。 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 2 后，此位即无法修改。*

位域	名称	权限	功能描述
10	OSSI	RW	空闲模式下的关闭状态选择 当由于刹车事件或软件写操作而使 MOE=0 时，此位作用于配置为输出的通道。 0：处于无效状态时，禁止 OC/OCN 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。 1：处于无效状态时，首先将 OC/OCN 输出强制为其无效电平，然后在死区时间后将其强制为空闲电平。定时器始终控制输出。 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 2 后，此位即无法修改。*
9:8	LOCK	RW	锁定配置 这些位用于针对软件错误提供写保护。 00：关闭锁定——不对任何位提供写保护。 01：锁定级别 1，此时无法对 ATIM_CR2 寄存器中的 OISy 和 OISyN 位、ATIM_DTR2 寄存器中的 DTGF、DATE 和 DTPE 位、ATIM_BDTR 寄存器中的 BK2P、BK2E、BK2F[3:0]、BKF[3:0]、AOE、BKP、BKE、LOCK 和 DTG 位、ATIM_AF1 寄存器的所有位域、ATIM_AF2 寄存器的所有位域执行写操作。 10：锁定级别 2，此时无法对锁定级别 1 中适用的各位、CC 极性位（ATIM_CCER 寄存器中的 CCyP/CCyNP 位，只要通过 CCyS 位将相关通道配置为输出）、ATIM_BDTR 寄存器中的 OSSR 和 OSSI 位执行写操作。 11：锁定级别 3，此时无法对锁定级别 2 中适用的各位、CC 控制位（ATIM_CCMRxCMP 寄存器中的 OCyM 和 OCyPE 位，只要通过 CCyS 位将相关通道配置为输出）执行写操作。 *注：复位后只能对 LOCK 位执行一次写操作。对 ATIM_BDTR 寄存器执行写操作后其中的内容将冻结，直到下一次复位。*
7:0	DTG	RW	配置死区发生器 此位域定义插入到互补输出之间的死区持续时间。DT 与该持续时间相对应。 DTG[7:5]=0xx => DT=DTG[7:0]×t_{dtg}，其中 t_{dtg}=t_{DTS}。 DTG[7:5]=10x => DT=(64+DTG[5:0]) ×t_{dtg}，其中 t_{dtg} = 2×t_{DTS}。 DTG[7:5]=110 => DT=(32+DTG[4:0]) ×t_{dtg}，其中 t_{dtg} = 8×t_{DTS}。 DTG[7:5]=111 => DT=(32+DTG[4:0]) ×t_{dtg}，其中 t_{dtg} = 16×t_{DTS}。 示例：如果 t_{DTS} = 125ns (8MHz)，则可能的死区值为： – 0 到 15875ns（步长为 125ns） – 16µs 到 31750ns（步长为 250ns） – 32µs 到 63µs（步长为 1µs） – 64µs 到 126µs（步长为 2µs） *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位域即无法修改。*

注：

由于可以根据 LOCK 配置锁定位 BK2P、BK2E、BK2F[3:0]、BKF[3:0]、AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0] 的写操作，因此必须在第一次对 ATIM_BDTR 寄存器执行写访问时对这些位进行配置。

14.9.26 ATIM_DTR2 死区时间寄存器 2

Address offset: 0x54 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17	DTPE	RW	死区时间值预加载使能 0：禁止死区时间值预加载 1：使能死区时间值预加载
16	DTAE	RW	死区时间不对称使能 0：上升沿和下降沿的死区时间是相同的，由 DTG[7:0] 定义。 1：上升沿的死区时间由 DTG[7:0] 定义，下降沿的死区时间由 DTGF[7:0] 定义。 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位域即无法修改。*
15:8	RFU	-	保留位，请保持默认值
7:0	DTGF	RW	下降沿的死区时间发生器 此位域定义插入到互补输出下降沿之间的死区持续时间。 DTGF[7:5]=0xx => DTF=DTGF[7:0]×t_{dtg}，其中 $t_{dtg}=t_{DTS}$。 DTGF[7:5]=10x => DTF=(64+DTG[5:0])×t_{dtg}，其中 $t_{dtg}=2\times t_{DTS}$。 DTGF[7:5]=110 => DTF=(32+DTG[4:0])×t_{dtg}，其中 $t_{dtg}=8\times t_{DTS}$。 DTGF[7:5]=111 => DTF=(32+DTG[4:0])×t_{dtg}，其中 $t_{dtg}=16\times t_{DTS}$。 示例：如果 t_{DTS} = 125ns (8MHz)，则可能的死区值为： – 0 到 15875ns（步长为 125ns） – 16µs 到 31750ns（步长为 250ns） – 32µs 到 63µs（步长为 1µs） – 64µs 到 126µs（步长为 2µs） *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1、2 或 3，此位域即无法修改。*

14.9.27 ATIM_ECR 编码控制寄存器

Address offset: 0x58 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:6	IPOS	RW	编码索引定位 在正交编码器模式（SMS[3:0] = 0001、0010、0011、1110、1111）中，此位表示索引事件在哪个 AB 输入配置中重置计数器。 00：当 AB = 00 时，索引会重置计数器 01：当 AB = 01 时，索引会重置计数器 10：当 AB = 10 时，索引会重置计数器 11：当 AB = 11 时，索引会重置计数器 在定向时钟编码器模式或时钟加方向编码器模式（SMS[3:0] = 1010、1011、1100、1101）中，这些位指示索引事件在哪个电平上重置计数器。在双向时钟编码器模式下，这一点都适用于两个时钟输入。 x0：当时钟信号为低电平时，索引将重置计数器 x1：当时钟信号为高电平时，索引将重置计数器
5	FIDX	RW	该位表示是否只考虑第一个索引 0：索引始终处于活动状态 1：仅第一个索引重置计数器
4:3	RFU	-	保留位，请保持默认值
2:1	IDIR	RW	索引方向 此位域表示索引在哪种计数方向上重置计数器 00：在任何计数方向上重置计数器 01：仅在向上计数时重置计数器 10：仅在向下计数时重置计数器 11：保留
0	IE	RW	索引使能 0：禁止索引 1：使能索引

14.9.28 ATIM_TISEL1 TI 输入选择寄存器 1

Address offset: 0x5C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28	RFU	-	保留位，请保持默认值
27:24	TI4SEL	RW	TI4 通道输入捕获信号来源选择，参见 TI1SEL 说明
23:20	RFU	-	保留位，请保持默认值
19:16	TI3SEL	RW	TI3 通道输入捕获信号来源选择，参见 TI1SEL 说明
15:12	RFU	-	保留位，请保持默认值
11:8	TI2SEL	RW	TI2 通道输入捕获信号来源选择，参见 TI1SEL 说明
7:4	RFU	-	保留位，请保持默认值
3:0	TI1SEL	RW	TI1 通道输入捕获信号来源选择 0000：ATIM_CH1 0001：VC1_OUT 0010：VC2_OUT 0011：UART1_RXD 0100：UART2_RXD 0101：UART3_RXD 0110：HSE_FAULT 0111：LSE_FAULT 1000：RTC_OUT 1001：LSI_OUT 1010：BTIM1_Trgo 1011：BTIM2_Trgo 1100：BTIM3_Trgo 1101：GTIM1_Trgo 1110：GTIM2_Trgo 1111：MCO_OUT

14.9.29 ATIM_TISEL2 TI 输入选择寄存器 2

Address offset: 0x6C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11:8	TI6SEL	RW	TI6 通道输入捕获信号来源选择，参见 TI1SEL 说明
7:4	RFU	-	保留位，请保持默认值
3:0	TI5SEL	RW	TI5 通道输入捕获信号来源选择，参见 TI1SEL 说明

14.9.30 ATIM_AF1 复用功能选项寄存器 1

Address offset: 0x60 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17:14	ETRSEL	RW	ETR 信号来源选择 0000：ATIM_ETR 引脚 0001：VC1_OUT 0010：VC2_OUT 0101：ADC_AWD 1000：LVD_OUT 1001：GTIM1_ETR 引脚 1010：GTIM2_ETR 引脚 1011：UART1_TXD 1100：UART2_TXD 1101：UART3_TXD 1110：HSE_FAULT 1111：LSE_FAULT *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，这些位即无法修改。*
13:12	RFU	-	保留位，请保持默认值
11	BKVC2P	RW	BRK VC2 输入极性 此位选择 VC2 输入电平灵敏度，必须与 BKP 极性位一起编程。 0：VC2 输入为高电平有效 1：VC2 输入为低电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
10	BKVC1P	RW	BRK VC1 输入极性 此位选择 VC1 输入电平灵敏度，必须与 BKP 极性位一起编程。 0：VC1 输入为高电平有效 1：VC1 输入为低电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
9	BKINP	RW	BRK BKIN 输入极性 此位选择 BKIN 复用功能输入电平灵敏度，必须与 BKP 极性位一起编程。 0：BKIN 输入为高电平有效 1：BKIN 输入为低电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
8:3	RFU	-	保留位，请保持默认值

位域	名称	权限	功能描述
2	BKVC2E	RW	BRK VC2 使能 此位使能定时器 BRK 输入的 VC2。VC2 输出与其他 BRK 源进行“或”运算。 0：禁止 VC2 输入 1：使能 VC2 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
1	BKVC1E	RW	BRK VC1 使能 此位使能定时器 BRK 输入的 VC1。VC1 输出与其他 BRK 源进行“或”运算。 0：禁止 VC1 输入 1：使能 VC1 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
0	BKINE	RW	BRK BKIN 输入使能 此位使能定时器 BRK 输入的 BKIN 复用功能。BKIN 输入与其他 BRK 源进行“或”运算。 0：禁止 BKIN 输入 1：使能 BKIN 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*

14.9.31 ATIM_AF2 复用功能选项寄存器 2

Address offset: 0x64 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18:16	OCRSEL	RW	OCREF_CLR 源选择 000：VC1_OUT 001：VC2_OUT 100：ADC_AWD 111：ETRF *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
15:12	RFU	-	保留位，请保持默认值
11	BK2VC2P	RW	BRK2 VC2 输入极性 此位选择 VC2 输入电平灵敏度，必须与 BK2P 极性位一起编程。 0：VC2 输入为低电平有效 1：VC2 输入为高电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*

位域	名称	权限	功能描述
10	BK2VC1P	RW	BRK2 VC1 输入极性 此位选择 VC1 输入电平灵敏度，必须与 BK2P 极性位一起编程。 0：VC1 输入为低电平有效 1：VC1 输入为高电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
9	BK2INP	RW	BRK2 BKIN2 输入极性 此位选择 BKIN2 复用功能输入电平灵敏度，必须与 BK2P 极性位一起编程。 0：BKIN2 输入为低电平有效 1：BKIN2 输入为高电平有效 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
8:3	RFU	-	保留位，请保持默认值
2	BK2VC2E	RW	BRK2 VC2 使能 此位使能定时器 BRK2 输入的 VC2。VC2 输出与其他 BRK2 源进行“或”运算。 0：禁止 VC2 输入 1：使能 VC2 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
1	BK2VC1E	RW	BRK2 VC1 使能 此位使能定时器 BRK2 输入的 VC1。VC1 输出与其他 BRK2 源进行“或”运算。 0：禁止 VC1 输入 1：使能 VC1 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*
0	BK2INE	RW	BRK2 BKIN 输入使能 此位使能定时器 BRK2 输入的 BKIN2 复用功能。BKIN2 输入与其他 BRK2 源进行“或”运算。 0：禁止 BKIN2 输入 1：使能 BKIN2 输入 *注：只要编程了 LOCK（ATIM_BDTR 寄存器中的 LOCK 位）级别 1，此位即无法修改。*

15 独立看门狗定时器（IWDT）

15.1 概述

CW32L011 内部集成独立看门狗定时器 (IWDT)。一旦启动 IWDT，用户需要在规定时间间隔内对 IWDT 的计数器进行重载，否则计数器溢出会触发复位或产生中断信号。IWDT 启动后，可停止计数。可选择在深度休眠模式下 IWDT 保持运行或暂停计数。

专门设置的键值寄存器，可以锁定 IWDT 的关键寄存器，防止寄存器被意外修改。

15.2 主要特性

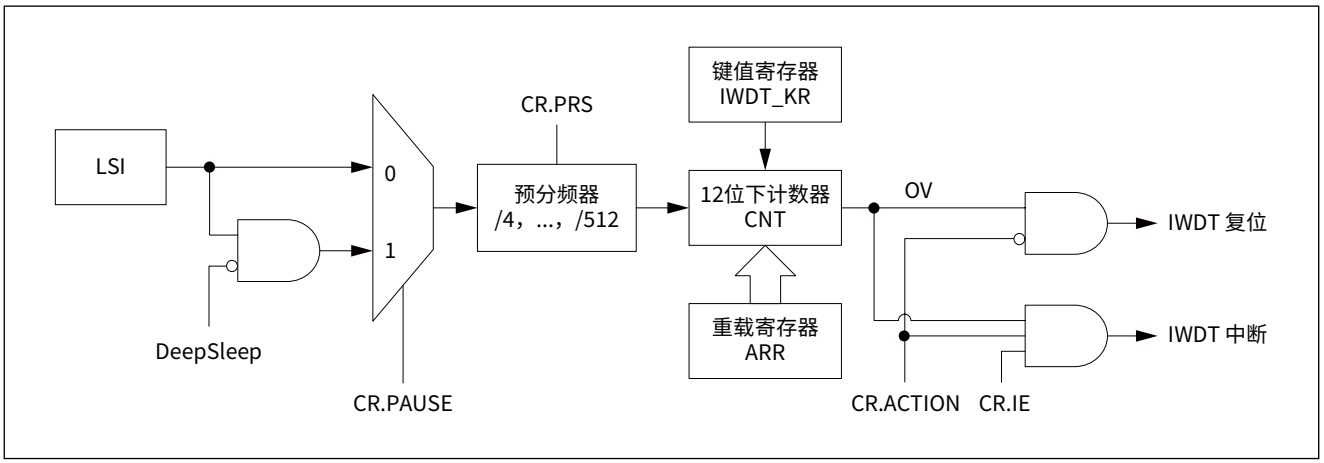
- 12bit 的向下计数器
- 可编程时钟预分频周期
- 溢出可触发中断或复位
- 寄存器保护锁功能
- 深度休眠模式下可暂停计数

15.3 功能描述

15.3.1 功能框图

IWDT 功能框图如下图所示：

图 15-1 IWDT 功能框图



IWDT 由一个 12 位可重载的向下计数器实现，其计数时钟源为内部低速 LSI 时钟，通过控制寄存器 IWDT_CR 的 PRS 位域可对其时钟源 LSI 信号进行 4 ~ 512 的预分频。IWDT 计数器发生溢出时可选择产生中断和复位信号。

15.3.2 工作方式

启动 IWDT 的计数器，需要向键值寄存器 IWDT_KR 写入 0xCCCC，计数器开始从 0xFFF 向下计数。

在计数器减到 0 之前，向 IWDT_KR 寄存器写入 0xAAAA，会触发计数器重载，将 ARR 寄存器值加载到计数器。当 IWDT 计数器值递减到 0，会产生溢出事件，溢出事件可触发 MCU 复位或产生 IWDT 中断信号，同时触发计数器重载。控制寄存器 IWDT_CR 的 ACTION 和 IE 位域，用于控制看门狗溢出时是否产生中断和复位，如下表所示：

表 15-1 工作方式

IWDT_CR.ACTION	IWDT_CR.IE	IWDT 溢出后动作
0	X	复位
1	1	不复位，只产生中断
1	0	不复位，不产生中断

IWDT 在 MCU 进入深度休眠模式时，可选择暂停 IWDT 计数，从而达到更低的系统整体功耗：控制寄存器 IWDT_CR 的 PAUSE 位域为 0，深度休眠模式时保持 IWDT 定时器运行；为 1 时暂停计数，当 MCU 退出深度休眠模式时 IWDT 自动恢复计数。

15.3.3 窗口选项

向窗口寄存器 IWDT_WINR 写入一个小于重载寄存器 IWDT_ARR 的值，可使 IWDT 工作于窗口看门狗模式。IWDT_WINR 寄存器的默认值是 0x0FFF，即窗口选项默认是关闭的。

修改 IWDT_WINR 寄存器的值，会触发重载操作，将 ARR 寄存器值加载到计数器。

使用 IWDT 窗口功能，用户应在计数器值小于等于 IWDT_WINR 窗口值，并且递减到 0 之前进行重载操作，以避免产生复位或看门狗定时器溢出。在看门狗计数器的值大于窗口值时进行重载操作，将触发系统复位。

15.3.4 寄存器锁定功能

通过 IWDT 的键值寄存器 IWDT_KR，可配置 IWDT 的重要寄存器 IWDT_CR、IWDT_ARR、IWDT_WINR 为锁定状态或解除锁定，锁定状态下不可对寄存器进行修改操作。

向 IWDT_KR 寄存器写入 0x5555，解除对 IWDT_CR、IWDT_ARR、IWDT_WINR 寄存器的锁定；向 IWDT_KR 寄存器写入其他任何值，启动锁定保护。

CW32L011 在上电复位后，IWDT_CR、IWDT_ARR、IWDT_WINR 寄存器默认处于锁定状态，用户需先解除锁定，才可对其进行修改操作。

15.3.5 启动刷新与停止

配置 IWDT_KR 寄存器，可实现 IWDT 的启动、刷新和停止操作：

- 写入 0xCCCC，启动 IWDT。
- 写入 0xAAAA，重载计数器，即刷新 IWDT。
- 顺序写入 0x5A5A、0xA5A5，停止 IWDT。

注意：

上述操作会同时启动 IWDT 寄存器锁定保护。

15.3.6 状态寄存器

状态寄存器 IWDT_SR，指示 IWDT 当前运行状态或寄存器更新状态，如下表所示：

表 15-2 IWDT 状态指示

IWDT_SR 位	名称	描述	1	0
4	RUN	运行标志	运行	未运行
3	OV	溢出标志	产生溢出	未溢出
5	RELOAD	计数器重载标志	正在重载	重载完成
2	WINRF	WINR 寄存器更新标志	正在更新	更新完成
1	ARRF	ARR 寄存器更新标志	正在更新	更新完成
0	CRF	CR 寄存器更新标志	正在更新	更新完成

为确保对 IWDT 寄存器的操作正确性，用户在更新 IWDT_WINR、IWDT_ARR、IWDT_CR 之后，需要检查 WINRF、ARRF、CRF 标志位是否为 0，以确认操作是否完成。

为确保对 IWDT 的重载操作，用户在进行重载操作后，应当检查 RELOAD 标志位是否为 0。

15.3.7 定时时长设定

IWDT 的计数时钟源为内部低速 LSI 时钟（时钟频率约为 32.8kHz，具体请参阅数据手册），通过控制寄存器 IWDT_CR 的 PRS 位域，可对其时钟源 LSI 信号进行分频，如下表所示：

表 15-3 IWDT 分频系数表

IWDT_CR.PRS	预分频值
000	4
001	8
010	16
011	32
100	64
101	128
110	256
111	512

看门狗定时时长计算公式：

$$T = (4 \times 2^{PRS} / f) \times (ARR + 1)$$

其中，f 为时钟源 LSI 的频率，PRS 为预分频系数，ARR 为重载值。

故，当时钟源 LSI 的频率为 32800Hz 时，IWDT 的最长和最短定时范围：

$$\text{IWDT 最短定时} = (4 \times 2^{0} / 32800) \times (0x000 + 1) \approx 122\ \mu s$$

$$\text{IWDT 最长定时} = (4 \times 2^{7} / 32800) \times (0xFFF + 1) \approx 63.9\ s$$

例：当时钟源 LSI 的频率为 32800Hz 时，设置预分频值为 64，重载值为 512，则：

$$\text{IWDT 定时时长} = (4 \times 2^{4} / 32800) \times (512 + 1) = 1\ s$$

15.4 编程示例

15.4.1 配置 IWDT 为独立看门狗

步骤 1：设置 SYSCTRL_APBEN2.IWDT 为 1，使能 IWDT 的配置时钟；

步骤 2：向 IWDT_KR 寄存器写入 0xCCCC，启动 IWDT；

注意：

需要启动 IWDT 后，才可对相关寄存器进行修改。

步骤 3：向 IWDT_KR 寄存器写入 0x5555，解除 IWDT 寄存器锁定功能；

步骤 4：配置 IWDT_CR，配置看门狗计数时钟与 LSI 振荡器的预分频值、溢出后动作、深度休眠模式下是否自动暂停；

步骤 5：配置 IWDT_ARR，配置看门狗的溢出周期；

步骤 6：等待 IWDT_SR.ARRF 和 IWDT_SR.CRF 变为 0，等待重载值和 CR 寄存器更新完成；

步骤 7：向 IWDT_KR 寄存器写入 0xAAAA，加载 ARR 到 IWDT 计数器。

15.4.2 配置 IWDT 为窗口看门狗

步骤 1：设置 SYSCTRL_APBEN2.IWDT 为 1，使能 IWDT 的配置时钟；

步骤 2：向 IWDT_KR 寄存器写入 0xCCCC，启动 IWDT；

注意：

需要启动 IWDT 后，才可对相关寄存器进行修改。

步骤 3：向 IWDT_KR 寄存器写入 0x5555，解除 IWDT 寄存器锁定功能；

步骤 4：配置 IWDT_CR，配置看门狗计数时钟与 LSI 振荡器的预分频值、溢出后动作、深度休眠模式下是否自动暂停；

步骤 5：配置 IWDT_ARR，配置看门狗的重载值；

步骤 6：配置 IWDT_WINR，配置窗口大小，注意 IWDT_WINR 必须小于 IWDT_ARR 重载值；

步骤 7：等待 IWDT_SR.ARRF、IWDT_SR.WINRF 和 IWDT_SR.CRF 变为 0，等待重载值、窗口寄存器和 CR 寄存器更新完成；

步骤 8：向 IWDT_KR 寄存器写入 0xAAAA，加载 ARR 到 IWDT 计数器。

15.4.3 刷新 IWDT（喂狗操作）

步骤 1：向 IWDT_KR 寄存器写入 0xAAAA，加载 ARR 到计数器；

步骤 2：等待 IWDT_SR.RELOAD 变为 0，等待重载操作完成。

15.5 寄存器列表

IWDT 基地址：IWDT_BASE = 0x4000 5000

表 15-4 IWDT 寄存器列表

寄存器名称	寄存器地址	寄存器描述
IWDT_KR	IWDT_BASE + 0x00	键值寄存器
IWDT_CR	IWDT_BASE + 0x04	控制寄存器
IWDT_ARR	IWDT_BASE + 0x08	重载寄存器
IWDT_SR	IWDT_BASE + 0x0C	状态寄存器
IWDT_WINR	IWDT_BASE + 0x10	窗口寄存器
IWDT_CNT	IWDT_BASE + 0x24	计数值寄存器

15.6 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

15.6.1 IWDT_KR 键值寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	KR	WO	写入 0xCCCC：启动 IWDT 计数器 写入 0xAAAA：重载 IWDT 计数值 依次写入 0x5A5A、0xA5A5：停止看门狗 写 入 0x5555： 解 除 IWDT_ARR、IWDT_WINR、IWDT_CR 的写保护 写入非 0x5555：使能 IWDT_ARR、IWDT_WINR、IWDT_CR 的写保护

15.6.2 IWDT_CR 控制寄存器

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:6	RFU	-	保留位，请保持默认值
5	PAUSE	RW	Deepsleep 模式下 IWDT 暂停控制 0：DeepSleep 模式下 IWDT 继续运行 1：DeepSleep 模式下 IWDT 自动暂停
4	IE	RW	IWDT 中断使能控制 0：禁止看门狗中断 1：使能看门狗中断
3	ACTION	RW	IWDT 溢出后动作配置 0：看门狗溢出后产生复位 1：看门狗溢出后产生中断
2:0	PRS	RW	配置看门狗计数时钟与 LSI 振荡器的分频比 000：4 分频 001：8 分频 010：16 分频 011：32 分频 100：64 分频 101：128 分频 110：256 分频 111：512 分频

注意：

向 IWDT_KR 寄存器写入 0x5555 后，才可修改本寄存器；IWDT_SR.CRF 为 0 时，才可修改 IWDT_CR。

15.6.3 IWDT_ARR 重载寄存器

Address offset: 0x08 Reset value: 0x0000 0FFF

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11:0	ARR	RW	IWDT 重载值

注意：

向 IWDT_KR 寄存器写入 0x5555 后，才可修改本寄存器；IWDT_SR.ARRF 为 0，才可修改本寄存器。

15.6.4 IWDT_CNT 计数值寄存器

Address offset: 0x24 Reset value: 0x0000 0FFF

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11:0	CNT	RO	计数值寄存器 *注：连续两次读到的数值相同，才可认为读出该计数值*

15.6.5 IWDT_WINR 窗口寄存器

Address offset: 0x10 Reset value: 0x0000 0FFF

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11:0	WINR	RW	窗口比较器比较值

15.6.6 IWDT_SR 状态寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:6	RFU	-	保留位，请保持默认值
5	RELOAD	RO	WDT 计数器重载标志 0：WDT 计数器已完成重载操作 1：WDT 计数器正在进行重载操作 *注：窗口模式下，进行重载操作后需要等待该标志变为 0*
4	RUN	RO	WDT 运行标志 0：WDT 没有运行 1：WDT 正在运行
3	OV	RW0	WDT 溢出标志 R0：WDT 没有溢出 R1：WDT 溢出 W0：清除 WDT 溢出标志
2	WINRF	RO	WINR 寄存器更新标志 0：WINR 寄存器更新完成 1：WINR 寄存器正在更新 *注：写 WINR 寄存器后需要等待该标志变为 0*
1	ARRF	RO	ARR 寄存器更新标志 0：ARR 寄存器更新完成 1：ARR 寄存器正在更新 *注：写 ARR 寄存器后需要等待该标志变为 0*
0	CRF	RO	CR 寄存器更新标志 0：CR 寄存器更新完成 1：CR 寄存器正在更新 *注：写 CR 寄存器后需要等待该标志变为 0*

16 通用异步收发器（UART）

16.1 概述

CW32L011 内部集成 3 个通用异步收发器 (UART)，支持异步全双工、同步半双工和单线半双工模式，支持硬件数据流控和多机通信，还支持 LIN（局域互连网络）；可编程数据帧结构，可以通过小数波特率发生器提供宽范围的波特率选择；内置定时器模块，支持等待超时检测、接收空闲检测、自动波特率检测和通用定时功能。

UART 控制器工作在双时钟域下，允许在深度休眠模式下进行数据的接收，接收完成中断可以唤醒 MCU 回到运行模式。

注意：

仅 UART1 支持 LIN 和定时器功能；UART2/3 可通过片内外设互联与 BTIM/GTIM/ATIM 的从模式协同工作实现超时定时器相关功能。

16.2 主要特性

- 支持双时钟域驱动
  - 配置时钟 PCLK
  - 传输时钟 UCLK
- 可编程数据帧结构：
  - 数据字长：8、9 位，LSB/MSB 在前
  - 校验位：无校验、奇校验、偶校验
  - 停止位长度：1、1.5、2 位
- 16 位整数、4 位小数波特率发生器
- 支持异步全双工、同步半双工、单线半双工
- 单独的发送器和接收器使能位
- 单独的发送和接收信号极性控制
- TXD/RXD 引脚配置可交换
- 支持 LoopBack 模式
- 支持接收数据匹配检测
- 间隔段帧和空闲字符发送和接收功能
- 硬件流控 RTS、CTS 和 RS485 驱动器使能
- 发送缓冲器空 / 接收数据完成触发启动 ADC
- 支持多机通信，自动地址识别
- 13 个带中断标志的中断源
- 自动波特率检测模式 1/2
- 等待超时检测 / 接收空闲检测
- 内置定时器模块支持通用定时器功能
- 错误检测：奇偶校验错误、帧结构错误、溢出错误、噪声错误
- 低功耗模式下收发数据，中断唤醒 MCU
- 支持与工作电压低于 MCU 的器件通信（借助 VC）
- LIN 主模式同步间隔段发送功能和 LIN 从模式同步间隔段检测功能：
  - 支持长度可配置的同步间隔段发送
  - 支持 10/11 位同步间隔段的检测

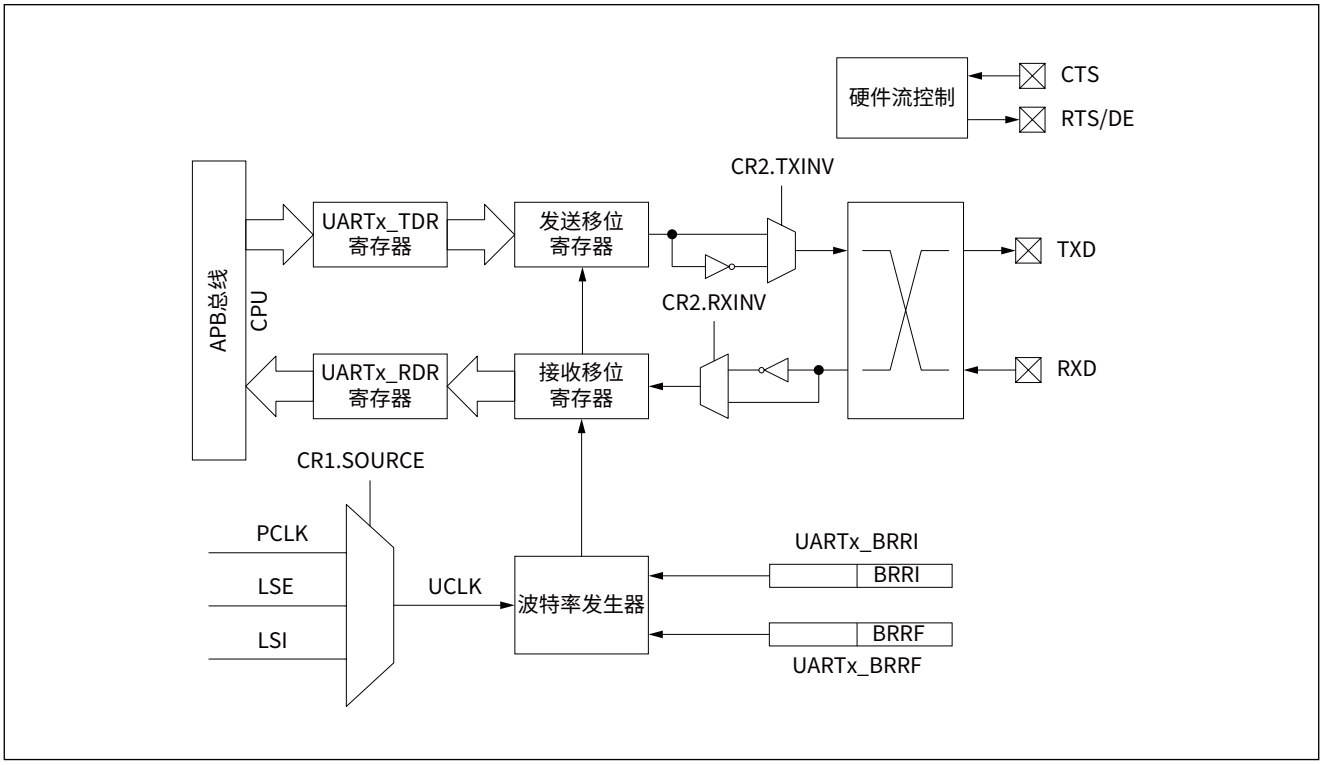
16.3 功能描述

16.3.1 功能框图

UART 控制器挂载到 APB 总线上，配置时钟域 PCLK，固定为 APB 总线时钟 PCLK，用于寄存器配置逻辑工作；传输时钟域 UCLK，用于数据收发逻辑工作，其来源可选择 PCLK 时钟、外部低速时钟（LSE）以及内部低速时钟（LSI）。双时钟域的设计更便于波特率的设置，支持从深度休眠模式下唤醒控制器。

UART 发射器和接收器有单独的使能位，支持发送和接收的单独信号极性控制，且 TXD/RXD 引脚配置可交换，方便用户配置。UART 控制器的功能框图如下图所示：

图 16-1 UART 功能框图



UART 控制器支持多种工作模式，在不同的工作模式下，各引脚具有不同的功能，引脚配置也不同，如下表所示（其中 x = 1、2、3）：

表 16-1 UART 端口配置

工作模式	UART 引脚	作用	GPIO 配置
异步全双工	UARTx_TXD	数据串行输出	数字，推挽输出 / 开漏输出，复用
	UARTx_RXD	数据串行输入	数字，上拉输入，复用
	UARTx_RTS [1]	请求发送	数字，推挽输出 / 开漏输出，复用
	UARTx_CTS	允许发送	数字，上拉输入，复用
同步半双工	UARTx_TXD	时钟信号输出	数字，推挽输出 / 开漏输出，复用
	UARTx_RXD	数据的发送和接收	数字，推挽输出 / 开漏输出，复用
单线半双工	UARTx_TXD	数据的发送和接收	数字，开漏输出，复用（需外接上拉）
	UARTx_RXD	不使用	可作通用 IO 使用

注 1：设置 RS485 驱动器使能信号时，RTS 引脚用作 DE 功能。

16.3.2 同步模式

UART 支持同步半双工工作模式。在该模式下，UARTx_TXD 引脚输出同步移位时钟信号，UARTx_RXD 引脚进行数据的发送和接收。UARTx_TXD 和 UARTx_RXD 引脚的具体配置参见表 16-1 UART 端口配置。

UART 的同步模式可以与 SPI 的单线半双工模式进行通信，此时 SPI 必须配置为固定的电平模式：时钟极性 CPOL 为 1、时钟相位 CPHA 为 1。

设置控制寄存器 UARTx_CR1 的 SYNC 位域为 1，使 UART 工作于同步半双工模式。此时 UART 只能作为主机，数据字长只能是 8 位，且 UARTx_CR1.SIGNAL 和 UARTx_CR2.ADDREN 必须清零。

16.3.2.1 波特率设置

同步半双工模式下，波特率计算公式：

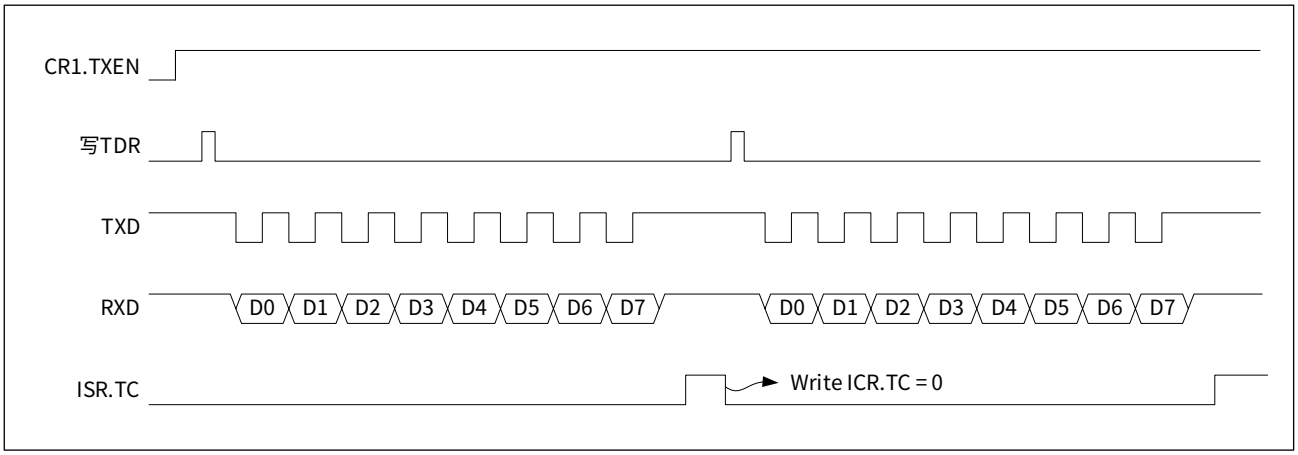
$$BaudRate = UCLK / 12$$

其中，UCLK 是 UART 的传输时钟，其来源可以是 PCLK、LSE 或 LSI，通过控制寄存器 UARTx_CR1 的 SOURCE 位域来选择。

16.3.2.2 数据收发

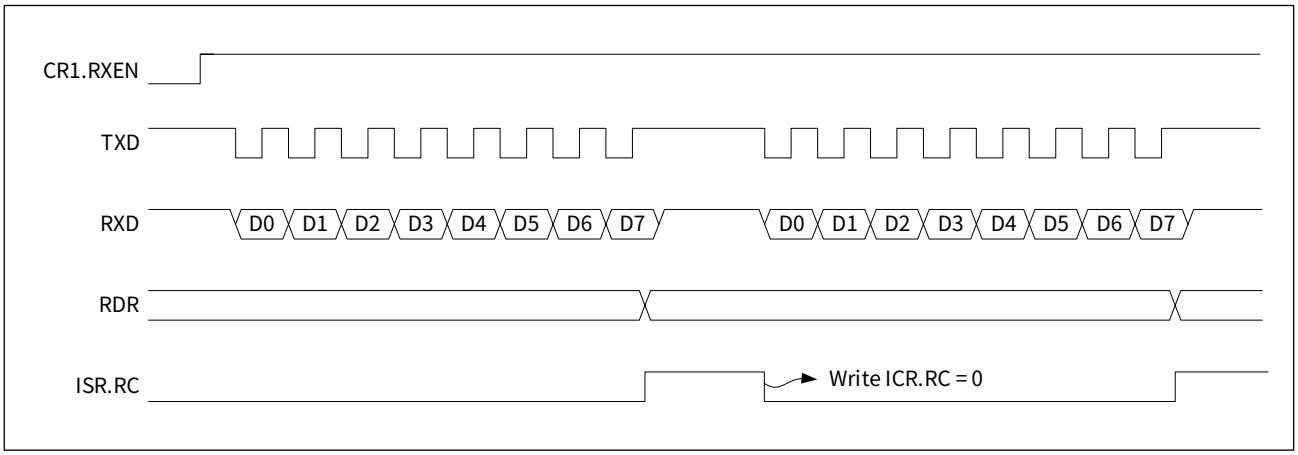
将 UART 发送使能位 UARTx_CR1.TXEN 置 1 使 UART 工作于发送状态，将数据写入 UARTx_TDR 寄存器后，数据将从 UARTx_RXD 引脚串行输出，同时 UARTx_TXD 引脚输出同步移位时钟信号。当 UARTx_TDR 寄存器和移位寄存器中的数据均已发送完成后，发送完成中断标志位 UARTx_ISR.TC 会被硬件置位。

图 16-2 同步半双工模式发送时序（发送两个字节数据）



将 UART 接收使能位 UARTx_CR1.RXEN 置 1 使 UART 工作于接收状态，UARTx_TXD 引脚将输出同步移位时钟信号，数据从 UARTx_RXD 引脚串行输入。数据接收完成后，接收完成中断标志位 UARTx_ISR.RC 会被硬件置位，此时可以读取 UARTx_RDR 寄存器。接收下一个字节之前，必须将 UARTx_ISR.RC 标志位清零，RC 标志清零后，立即开始接收下一个字节数据。

图 16-3 同步半双工模式接收时序（接收两个字节数据）



16.3.3 异步模式

16.3.3.1 数据帧格式

在 UART 异步通信中，数据是以数据帧的形式发送和接收的，一帧数据包括一个起始位、一个数据域、一个可选的校验位和宽度可编程的停止位。

起始位

起始位长度固定为 1 位，起始位的判定方式可以选择下降沿或低电平，通过控制寄存器 UARTx_CR1 的 START 位域来选择。一般选择下降沿作为起始位的判定方式，低电平方式判定起始位主要应用于低功耗模式，请参见 16.4 低功耗模式。

数据域

数据字长可以设置为 8 位或 9 位，通过控制寄存器 UARTx_CR1 的 CHLEN 位域来配置。当禁止奇偶校验时（设置 UARTx_CR1.PARITYEN 为 0），数据字长可以设置为 8 位或 9 位; 当使能奇偶校验时（设置 UARTx_CR1.PARITYEN 为 1），数据字长必须设置为 9 位，如果字符长度设置为 8 位，PARITYEN 位域会自动清零。请参见表 16-2 数据帧结构。

数据位传输顺序可选择 MSB 或 LSB 在前，通过控制寄存器 UARTx_CR1 的 MSBF 位域来选择。

校验位

校验方式支持偶校验和奇校验，具体通过控制寄存器 UARTx_CR1 的 PARITY 位域来选择。

- 奇校验：校验位使一帧数据中数据位和校验位中“1”的总数为奇数。
- 偶校验：校验位使一帧数据中数据位和校验位中“1”的总数为偶数。

停止位

停止位长度可以配置为 1、1.5 或 2 位，具体通过控制寄存器 UARTx_CR1 的 STOP 位域来配置。

表 16-2 数据帧结构

UARTx_CR1.CHLEN	UARTx_CR1.PARITYEN	数据帧
0	X	起始位 +8 位数据 + 停止位
1	0	起始位 +9 位数据 + 停止位
1	1	起始位 +8 位数据 + 奇偶校验位 + 停止位

16.3.3.2 小数波特率发生器

波特率的产生

UART 的接收和发送波特率是相同的，由同一个波特率发生器产生。

波特率发生器支持 16 倍采样、8 倍采样、4 倍采样和专用采样这 4 种采样模式，具体的采样模式通过控制寄存器 UARTx_CR1 的 OVER 位域来选择。

1. OVER = 00，设置 16 倍采样，波特率计算公式：

$$BaudRate = UCLK / (16 \times BRRI + BRRF)$$

UCLK 是 UART 的传输时钟，其来源可以是 PCLK、LSE 或 LSI，具体来源通过 UARTx_CR1.SOURCE 来选择。

BRRI（UARTx_BRRI［15:0］），是波特率计数器的整数部分，可设置范围为 1 ~ 65535。

BRRF（UARTx_BRRF［3:0］），是波特率计数器的小数部分，可设置范围为 0 ~ 15。

例 1：

当传输时钟 UCLK 的频率为 24MHz 时，设置 BRRI = 156（即 UARTx_BRRI = 0x9C），BRRF = 4（即 UARTx_BRRF = 0x04），则：

$$BaudRate = 24000000 / (16 \times 156 + 4) = 9600 \text{ bps}$$

例 2：

当传输时钟 UCLK 的频率为 24MHz 时，要求配置 BaudRate = 115200 bps，计算

$$16 \times BRRI + BRRF = 24000000 / 115200 = 208.33$$

则：

BRRI = 208.33 / 16 = 13.02，最接近的整数是：13（0x0D）

BRRF = 0.02×16 = 0.32，最接近的整数是：0（0x00）

即需要设置 UARTx_BRRI 为 0x0D，UARTx_BRRF 为 0x00

此时，实际波特率 BaudRate = 115384.62 bps，误差率为 0.16%

2. OVER = 01，设置 8 倍采样，波特率计算公式：

$$BaudRate = UCLK / (8 \times BRRI)$$

3. OVER = 10，设置 4 倍采样，波特率计算公式：

$$BaudRate = UCLK / (4 \times BRRI)$$

4. OVER = 11，设置专用采样，波特率计算公式：

$$BaudRate = (256 \times UCLK) / BRRI$$

注：

专用采样仅适合传输时钟源为 LSE 或者 LSI 时，进行 2400bps、4800bps 或 9600bps 波特率下的 UART 通信。

例 1：

传输时钟 UCLK 选择 LSE，频率为 32.768kHz，要求配置 9600 bps 波特率，则：

BRRI = 256×32768 / 9600 = 873.81，最接近的整数是：874（0x36A）

即需要设置 UARTx_BRRI 为 0x36A

此时，实际波特率 BaudRate = 9597.95 bps，误差为 0.02%。

波特率设置示例

表 16-3 ~ 表 16-9 列举了常用 MCU 频率时，波特率计数寄存器的设定值及波特率误差。

表 16-3　UCLK 为 4MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	52	1	4801.92	0.04%
9600	26	1	9592.33	-0.08%
14400	17	6	14388.49	-0.08%
19200	13	0	19230.77	0.16%
38400	6	8	38461.54	0.16%
56000	4	7	56338.03	0.60%
57600	4	5	57971.01	0.64%
115200	2	3	114285.71	-0.79%
256000	1	0	250000	-2.34%

表 16-4　UCLK 为 8MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	104	3	4799.04	-0.02%
9600	52	1	9603.84	0.04%
14400	34	12	14388.49	-0.08%
19200	26	1	19184.65	-0.08%
38400	13	0	38461.54	0.16%
56000	8	15	55944.06	-0.10%
57600	8	11	57553.96	-0.08%
115200	4	5	115942.03	0.64%
500000	1	0	500000	0.00%

表 16-5 UCLK 为 16MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	208	5	4800.48	0.01%
9600	104	3	9598.08	-0.02%
14400	69	7	14401.44	0.01%
19200	52	1	19207.68	0.04%
38400	26	1	38369.30	-0.08%
56000	17	14	55944.06	-0.10%
57600	17	6	57553.96	-0.08%
115200	8	11	115107.91	-0.08%
1000000	1	0	1000000	0.00%

表 16-6 UCLK 为 24MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	312	8	4800.00	0.00%
9600	156	4	9600.00	0.00%
14400	104	3	14397.12	-0.02%
19200	78	2	19200.00	0.00%
38400	39	1	38400.00	0.00%
56000	26	13	55944.06	-0.10%
57600	26	1	57553.96	-0.08%
115200	13	0	115384.62	0.16%
1500000	1	0	1500000	0.00%

表 16-7　UCLK 为 32MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	416	11	4799.76	0.00%
9600	208	5	9600.96	0.01%
14400	138	14	14401.44	0.01%
19200	104	3	19196.16	-0.02%
38400	52	1	38415.37	0.04%
56000	35	11	56042.03	0.08%
57600	34	12	57553.96	-0.08%
115200	17	6	115107.91	-0.08%
2000000	1	0	2000000	0.00%

表 16-8　UCLK 为 48MHz 波特率设置示例（OVER = 00）

波特率（bps）	BRRI	BRRF	实际波特率	误差率
4800	625	0	4800.00	0.00%
9600	312	8	9600.00	0.00%
14400	208	5	14401.44	0.01%
19200	156	4	19200.00	0.00%
38400	78	2	38400.00	0.00%
56000	53	9	56009.33	0.02%
57600	52	1	57623.05	0.04%
115200	26	1	115107.91	-0.08%
3000000	1	0	3000000	0.00%

表 16-9　UCLK 为 32.768kHz 波特率设置示例（OVER = 11）

波特率（bps）	BRRI	实际波特率	误差率
2400	3495	2400.17	0.01%
4800	1748	4798.97	-0.02%
9600	874	9597.95	-0.02%

16.3.3.3 发送控制

数据发送

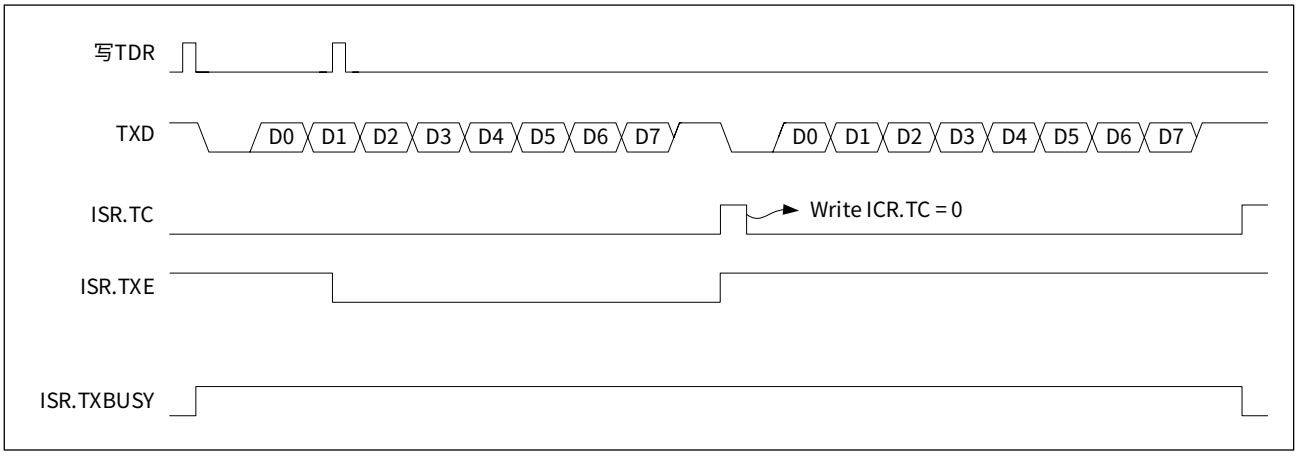
设置 UARTx_CR1.TXEN 为 1 使能发送电路。将数据写入 UARTx_TDR 寄存器后，数据会被硬件转移到发送移位寄存器，发送移位寄存器将数据串行移出（数据位传输顺序可选择 MSB 或 LSB 在前）。具体发送流程的寄存器配置请参见 16.6 编程示例。

当数据被硬件转移到发送移位寄存器后，发送缓冲器空中断标志位 UARTx_ISR.TXE 会被硬件置位，表示 UARTx_TDR 寄存器已空，此时允许对 UARTx_TDR 寄存器写入新的待发送数据。在对 UARTx_TDR 寄存器写入数据的同时，UARTx_ISR.TXE 标志位会被自动清零。如果此时发送移位寄存器中尚有未发送完成的数据，那么新写入的数据会在 UARTx_TDR 寄存器中暂存，在当前数据发送完成后，UARTx_TDR 寄存器中的数据会被硬件转移到发送移位寄存器中继续发送。

当发送移位寄存器中的数据帧发送完成后，如果 UARTx_TDR 寄存器中没有待发送的数据（即 UARTx_ISR.TXE = 1），发送完成中断标志位 UARTx_ISR.TC 会被硬件置位，同时 UART 发送忙标志位 UARTx_ISR.TXBUSY 会被硬件清零，表示数据已全部发送完成，UART 发送器空闲。

UART 发送控制环节的时序图如下图所示：

图 16-4 异步工作模式发送控制时序（发送两个字节数据）



间隔段帧

间隔段帧是完全由‘0’组成的一个完整的数据帧（停止位期间也是‘0’）。在间隔段帧结束时，发送器再插入 2 个比特的逻辑‘1’，以保证能识别下一帧的起始位。

设置 UARTx_TDR.BREAK 为 1 将发送间隔段帧，间隔段帧长度由 UART 数据字长决定。如果写入 BREAK 时，存在正在发送的数据，则等待当前数据发送完成后，将在 TXD 上发送一个间隔段帧。

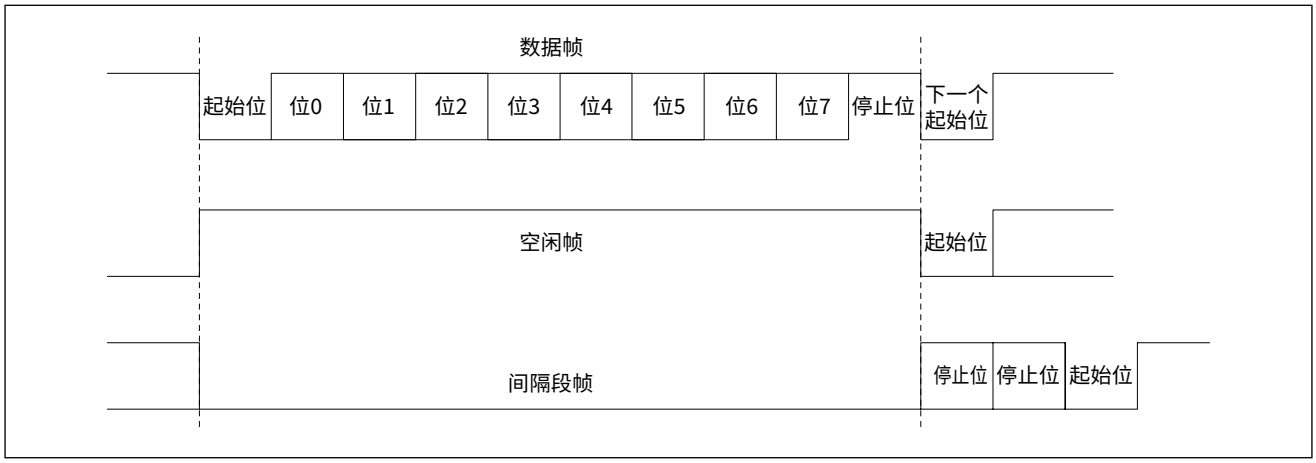
空闲帧

空闲帧是完全由‘1’组成的一个完整的数据帧（停止位期间也是‘1’），后面跟着包含了数据的下一帧的起始位。

设置 UARTx_TDR.IDLE 为 1 将发送空闲帧，空闲帧长度由 UART 数据字长决定。如果写入 IDLE 时，存在正在发送的数据，则等待当前数据发送完成后，将在 TXD 上发送一个空闲帧。

间隔段帧和空闲帧的时序图如下图所示，其中数据字长为 8 位：

图 16-5 间隔段帧和空闲帧时序



16.3.3.4 接收控制

数据接收

设置 UARTx_CR1.RXEN 为 1 使能接收电路。当接收器侦测到起始位后，开始接收数据，接收期间，数据从最低位或最高位开始串行移入接收移位寄存器，并行转移到 UARTx_RDR 寄存器，此时数据可以被读出。具体接收流程的寄存器配置请参见 16.6 编程示例。

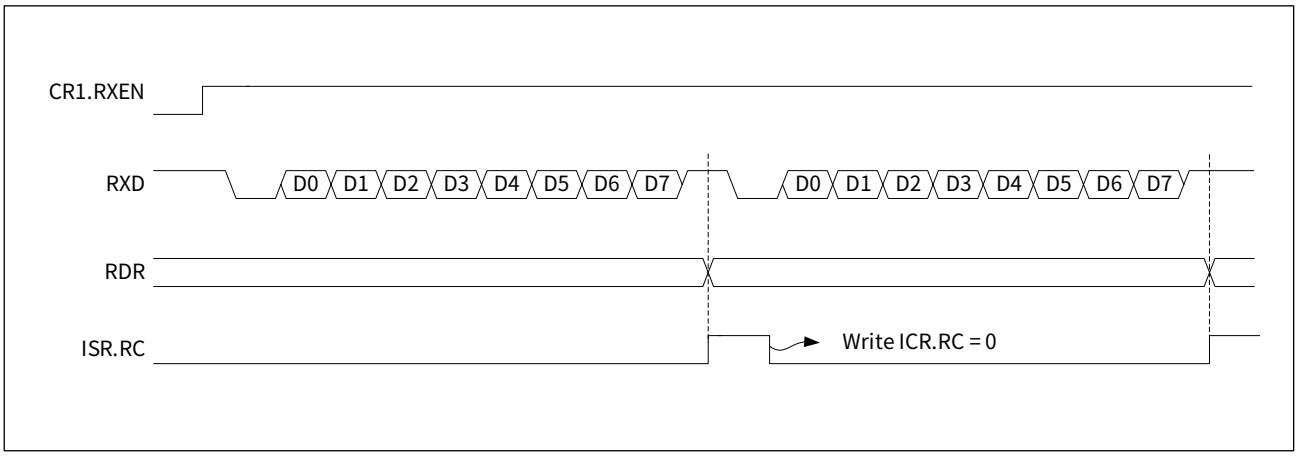
当接收数据从移位寄存器转移到 UARTx_RDR 寄存器后，接收完成中断标志位 UARTx_ISR.RC 会被硬件置位，表示已经完成一帧数据的接收。用户程序中，一旦检测到 UARTx_ISR.RC 标志位为 1，应尽快读取 UARTx_RDR 寄存器，并清除 UARTx_ISR.RC 标志位。如果未及时读取 UARTx_RDR 寄存器，新接收的数据会覆盖 UARTx_RDR 寄存器中的数据，造成数据丢失。

在接收一个数据帧时，会自动进行奇偶校验，若检测到奇偶校验错误，UARTx_ISR.PE 标志位会被硬件置位，表示奇偶校验出错。

在接收一个数据帧时，如果在预期时间内未识别到正确的停止位，则判定发生帧结构错误，UARTx_ISR.FE 标志位会被硬件置位。

UART 接收控制环节的时序图如下图所示：

图 16-6 异步工作模式接收控制时序（接收两个字节数据）



溢出错误

如果 UARTx_ISR.RC 标志位还没有清零，又接收到一个字符，则发生溢出错误，UARTx_ISR.ORE 标志位会被硬件置位。发生溢出错误后，新接收的数据会覆盖 UARTx_RDR 寄存器中的数据，造成数据丢失。

如果设置 UARTx_IER.ORE 为 1，将产生溢出中断，设置 UARTx_ICR.ORE 为 0，可清除溢出标志。

噪声错误

当在接收帧中检测到噪声时，在 UARTx_ISR.RC 标志位的上升沿 UARTx_ISR.NE 标志位被硬件置位，无效数据从移位寄存器转移到 UARTx_RDR 寄存器。如果设置 UARTx_IER.NE 为 1，将产生噪声中断，设置 UARTx_ICR.NE 为 0，可清除噪声标志。注意噪声错误仅在 16 倍采样或 8 倍采样时有效。

接收数据匹配

设置 UARTx_CR2.RXMATCHEN 为 1 可使能接收数据匹配检测。当接收到的数据与 UARTx_RXMATCH 寄存器配置的目标接收数据相同时，UARTx_ISR. RXMATCH 标志位会被硬件置位。如果设置 UARTx_IER. RXMATCH 为 1，将产生接收数据匹配中断，设置 UARTx_ICR. RXMATCH 为 0，可清除该标志。

间隔段帧

当 UART 接收到间隔段帧时，RXBRK、RC 和 FE 标志都会置位，UARTx_RDR 寄存器接收到 0，如果使能相应的中断，将产生对应的中断请求。

空闲帧

每当接收完成一个字符时，才会启动空闲字符检测。当检测到空闲字符时，RXIDLE 标志置位，如果使能相应的中断，将产生中断请求。

16.3.3.5 单线半双工模式

设置 UARTx_CR1.SIGNAL 为 1 使 UART 工作于单线半双工工作模式。在该模式下，使用 UARTx_TXD 引脚进行数据的发送和接收，不占用 UARTx_RXD 引脚（UARTx_RXD 可作通用 IO 使用），引脚具体配置参见表 16-1 UART 端口配置。

写数据到 UARTx_TDR 寄存器后，UARTx_TXD 引脚立即进入发送状态，输出 UARTx_TDR 寄存器中的数据。数据发送完成后，UARTx_TXD 引脚恢复到常态的接收状态。

没有发送数据时，UARTx_TXD 引脚处于接收状态，数据接收完成后，接收完成标志位 UARTx_ISR.RC 会被硬件置位，此时应尽快读取 UARTx_RDR 寄存器，并清除 UARTx_ISR.RC 标志位。

注意：

用户应采取适当的应用层保护机制，以确保不会出现多主机同时向总线发送数据。

16.3.3.6 多机通信

UART 支持多机通信方式。在该模式下，UART 总线上有一个主机和多台从机，每个从机有唯一的从机地址，通信时主机会先发送地址帧对从机寻址，只有地址匹配的从机才被激活，接收随后主机发送的数据帧。

主机发送

多机通信模式下，主机需将帧数据长度设置为 9 位，并禁止奇偶校验。

UARTx_TDR.TDR 的最高位决定主机发送地址帧还是数据帧，UARTx_TDR[8] 为 1 表示主机发送的是地址帧，UARTx_TDR[8] 为 0 表示主机发送的是数据帧。

从机接收

多机通信模式下，从机需将帧数据长度设置为 9 位，并禁止奇偶校验，同时设置 UARTx_CR2.ADDREN 为 1，使能从机地址识别，从机硬件自动检测主机发送的地址与本机地址是否匹配。

如果地址匹配，从机会将接收到的地址帧保存到 UARTx_RDR 寄存器中，UARTx_ISR.RC 标志位被硬件置位，同时 UARTx_ISR.SLVMATCH 标志位被硬件置位，从机接收随后主机发送的数据帧。通信过程中，从机需

1. 应用程序在接收完成中断 RC 里查询 UARTx_RDR[8]，以判断接收到的是地址帧还是数据帧。
2. 从机在发送数据帧时，需要将 UARTx_TDR[8] 设置为 0，以避免被其它从机当作地址帧。

如果地址不匹配，从机不会接收主机发送的数据帧，也不产生接收完成中断，已置位的 UARTx_ISR.SLVMATCH 标志位将被清零。

从机地址与地址掩码

从机地址由 UARTx_ADDR 寄存器配置，从机地址应配置为唯一。UARTx_MASK 寄存器是地址掩码，UARTx_MASK［7:0］中为‘1’的位对应的从机地址位参与从机地址匹配运算，为‘0’的位对应的从机地址位则不参与从机地址匹配运算。

当 UARTx_MASK［7:0］中全部位设置为‘1’时，即向 UARTx_MASK 寄存器写入 0xFF，则从机地址的 8 位地址位全部参与从机地址匹配，主机能唯一识别到从机。

当 UARTx_MASK［7:0］中部分位设置为‘0’时，对应的从机地址位不参与从机地址匹配，用以实现多个从机响应主机发出的同一地址帧，即主机对多个从机同时寻址。

例 1：

对从机 A 设置：UARTx_ADDR = 0xA0，UARTx_MASK = 0xFE
对从机 B 设置：UARTx_ADDR = 0xA1，UARTx_MASK = 0xFE
对从机 C 设置：UARTx_ADDR = 0xA2，UARTx_MASK = 0xFC
对从机 D 设置：UARTx_ADDR = 0xA3，UARTx_MASK = 0xFC
对从机 E 设置：UARTx_ADDR = 0xA5，UARTx_MASK = 0xFF
主机发送 0xA0 或 0xA1 地址帧时，可以寻址从机 A、从机 B、从机 C、从机 D
主机发送 0xA2 或 0xA3 地址帧时，可以寻址从机 C、从机 D
主机发送 0xA4 地址帧时，没有从机被寻址
主机发送 0xA5 地址帧时，可以寻址从机 E

广播地址

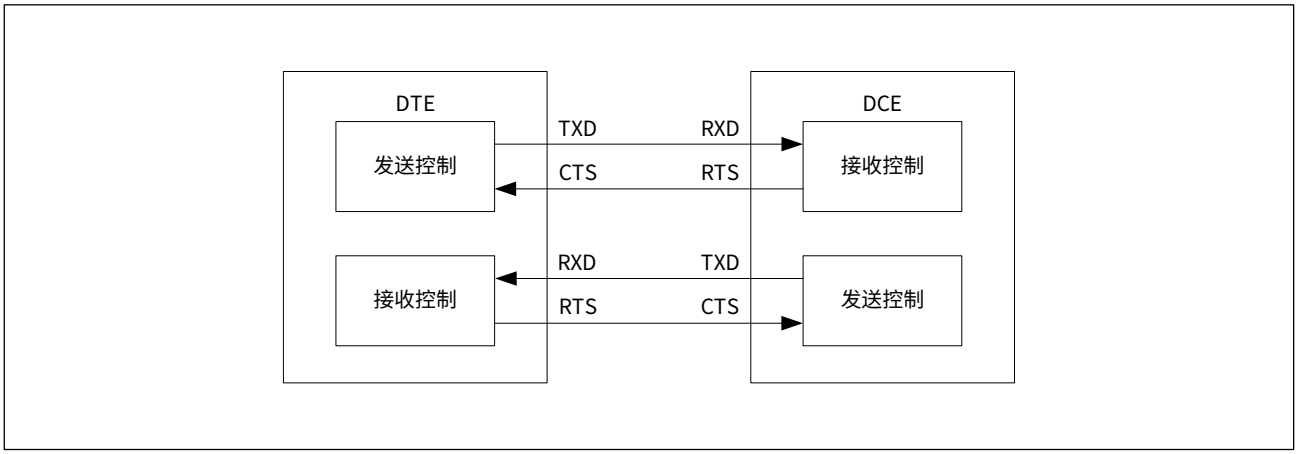
多机通信中，地址 0xFF 被定义为广播地址，主机发送广播地址帧时，所有地址的从机都被寻址。

16.3.3.7 硬件流控

CW32L011 支持 UART 硬件流控模式，即支持请求发送 RTS 和允许发送 CTS，用于自动控制数据的发送和接收。

数据通信系统中，数据终端设备 DTE 与数据电路终接设备 DCE 之间的通信硬件流控示意图如下图所示：

图 16-7 硬件流控示意图



请求发送 RTS

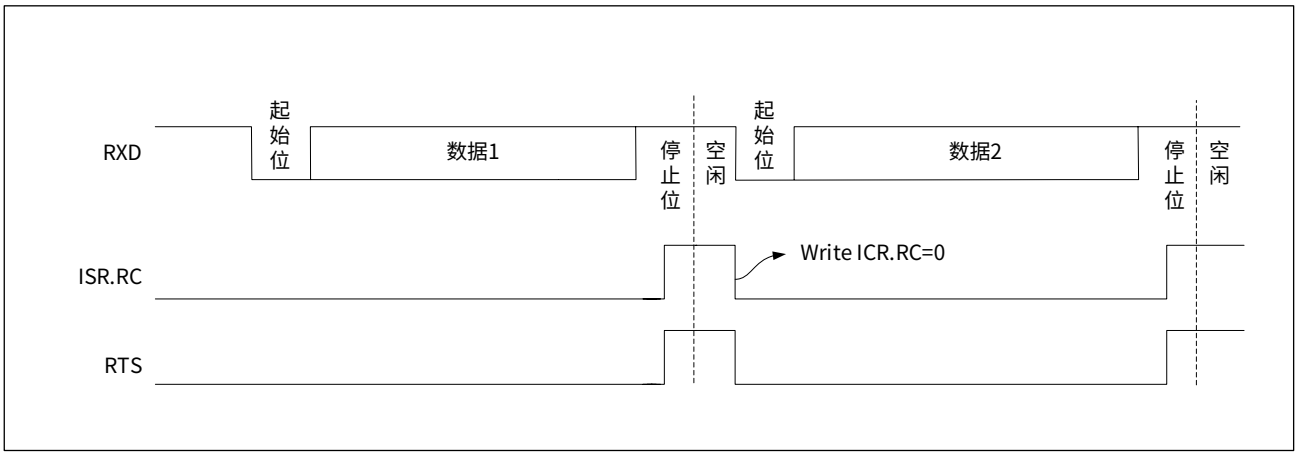
将 UARTx_CR2.RTSEN 置 1 使能 RTS 流控，RTS 表示请求发送，低电平有效。

当接收端准备好接收新的数据时（即 UARTx_ISR.RC = 0），RTS 引脚就会输出低电平，请求发送端发送一帧数据。

当接收端接收完成一帧数据时（即 UARTx_ISR.RC = 1），RTS 引脚就会输出高电平，通知发送端暂停发送。

用户应用中，读取 UARTx_RDR 寄存器中的数据后，如需继续接收下一帧数据，须软件将 UARTx_ISR.RC 标志位清零，此时 RTS 引脚将输出低电平，请求发送端发送下一帧数据。

图 16-8 RTS 流控时序



允许发送 CTS

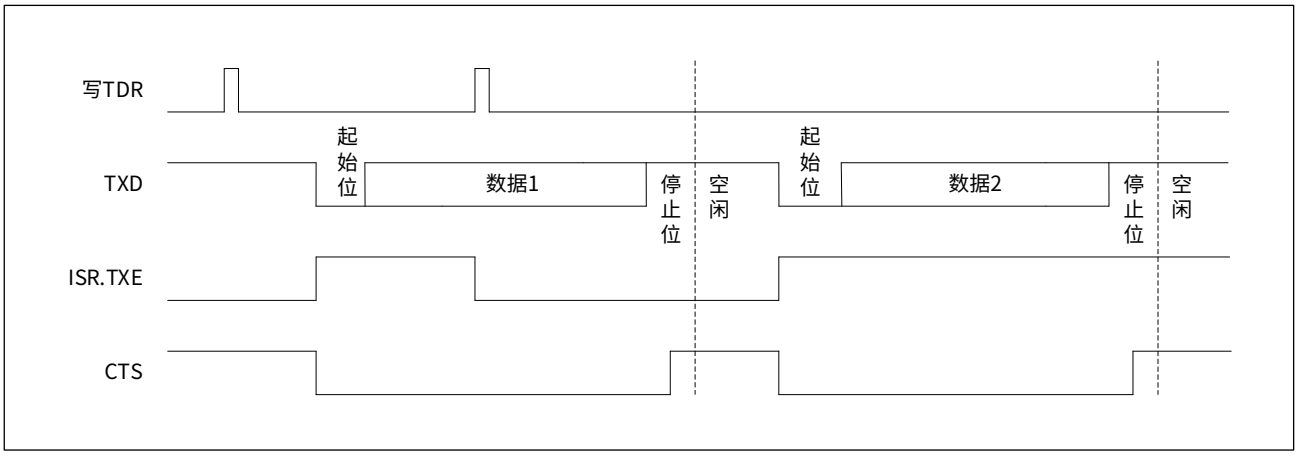
将 UARTx_CR.CTSEN 置 1 使能 CTS 流控，CTS 表示允许发送，低电平有效。

当发送端检测到 CTS 信号为低电平时，如果有待发送的数据（即 UARTx_ISR.TXE = 0），且当前没有正在发送的数据，则会发送这一帧数据。

当发送端检测到 CTS 信号为高电平时，如果有正在进行的数据发送，当前发送会继续进行，当前的发送完成后将停止发送。

在 CTS 流控过程中，CTS 信号电平标志位 UARTx_ISR.CTSLV 指示了当前 CTS 引脚的电平状态；CTS 信号变化标志位 UARTx_ISR.CTS 指示了 CTS 信号是否发生了变化，CTS 信号发生变化时，UARTx_ISR.CTS 标志位会被硬件置位，当 CTS 中断允许时（即设置 UARTx_IER.CTS 为 1），则会产生中断，写 UARTx_ICR.CTS 为 0 清除该中断标志。

图 16-9 CTS 流控时序



16.3.3.8 RS485 驱动器使能

设置控制寄存器 UARTx_CR3 的 DEM 位域为 1，可使能 RS485 驱动器使能功能，此时 RTS 引脚将为 DE（驱动器使能）功能，用户需将此引脚配置为数字输出的复用功能。

使能该功能后，用户可通过 DE 信号激活外部收发器的发送模式。DE 信号的极性可通过 UARTx_CR3 寄存器的 DEP 位域来配置，可选择 DE 信号高电平有效或低电平有效。

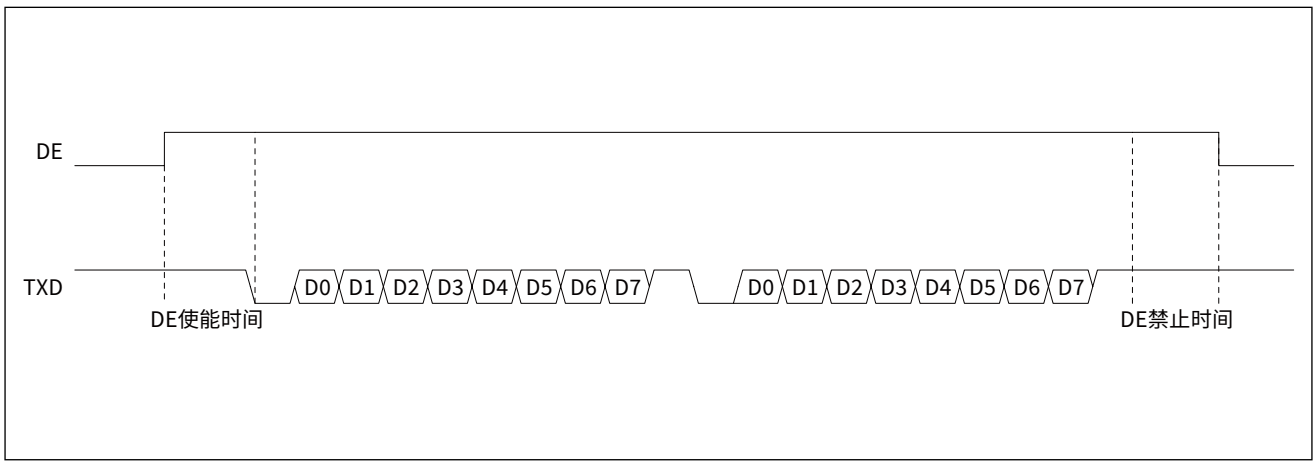
驱动器的使能时间为输出 DE 有效电平到开始发送 START 位之间的时间，禁止时间为最后一个 STOP 位发送完成到输出 DE 无效电平之间的时间，驱动器使能时间和禁止时间均通过 UARTx_CR3 寄存器的 DETIME 位域来设置，计算公式如下：

$$\text{驱动器使能和禁止时间} = \text{采样时间} \times \text{DETIME}$$

其中采样时间为 1/4、1/8、1/16 比特时间，专用采样（OVER=3）时不支持 DE 功能。

当连续发送几个数据时，最后一个 STOP 位发送完成后，DE 信号才会变为无效电平。DE 高电平有效时，其时序如下图所示：

图 16-10 驱动器使能时序



16.3.3.9 LoopBack 模式

设置 UARTx_CR2.LOOP 为 1 使 UART 工作于 LoopBack 模式。在该模式下，RXD 信号来源为内部的 TXD 信号，不占用外部的 UARTx_TXD 和 UARTx_RXD 引脚（UARTx_TXD 和 UARTx_RXD 可作通用 GPIO 使用）。

写数据到 UARTx_TDR 寄存器后，RXD 从内部接收数据到 UARTx_RDR 寄存器，发送和接收完成后，TC 和 RC 标志位都会置 1，如果使能了中断将产生对应的中断请求。

16.3.3.10 输入信号来源 / 电平转换

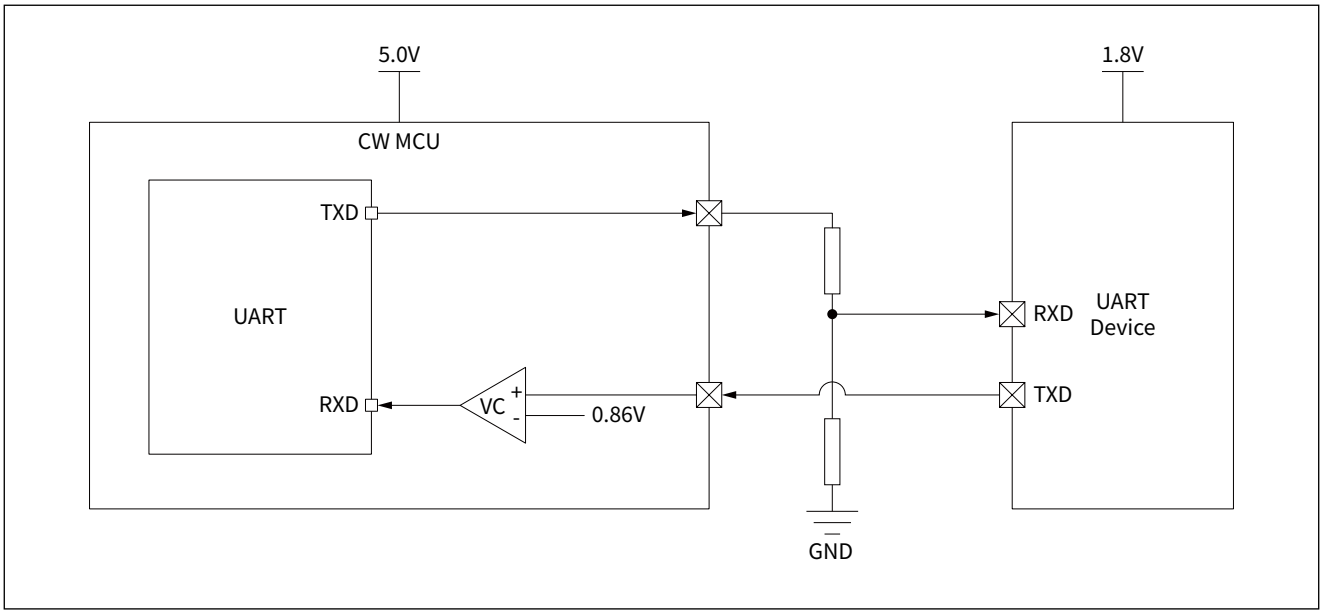
本模块的 RXD 输入信号可配置为多种来源，如下表所示：

表 16-10　RXD 输入信号来源

UARTx_CR2.RXSRC	UARTx_CR1.SIGNAL	UARTx_CR2.SWAP	RXD 输入信号来源
000	0	0	RXD 引脚
		1	TXD 引脚
	1	0	TXD 引脚
		1	RXD 引脚
001	-	-	VC1_OUT
010			VC2_OUT

当配置 RXD 的输入信号来源为 VCx_OUT 时，本模块可与低于本芯片工作电压的 UART 器件进行通信，VCx 在通信链路中充当了电平转换的角色，典型工作电路如下图所示：

图 16-11　不同电压器件通信示意图

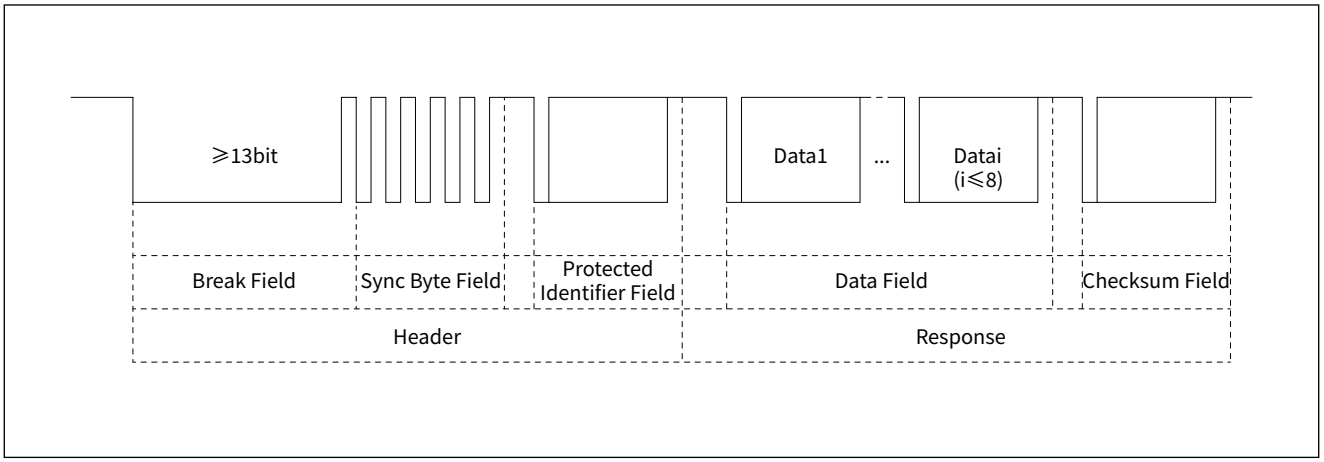


16.3.4 LIN 模式

通过将 UARTx_CR3 寄存器中的 LIN 位域设置为 1 来选择 LIN 模式。在 LIN 模式下，必须将 UARTx_CR1 寄存器中的 SIGNAL、SYNC、STOP 和 PARITY 位清零。

LIN 的一帧 (Frame) 数据包含帧头 (Header) 和应答 (Response) 两部分。主机任务负责发送帧头，从机任务接收帧头并对帧头所包含信息进行解析，并对帧头作出反应 (接收 / 发送 / 忽略应答部分)。LIN 的帧结构如下图所示：

图 16-12 LIN 帧的结构



在 LIN 的一帧当中，除同步间隔段（Break Field），其它各段都是通过字节域的格式传输的。字节域包括 1 位起始位 +8 位数据位 +1 位停止位，数据传输都是最低有效位（LSB）在前。

16.3.4.1 LIN 发送

同步间隔段的发送

同步间隔段表示一帧数据的开始，LIN 间隔段的比特数量由 UARTx_TDR.TDR 设置，注意同步间隔是至少持续 13 位 (以主机节点的位速率为准) 的显性电平。

在 LIN 模式下，设置 UARTx_TDR.BREAK 为 1，同时向 UARTx_TDR.TDR 中写入间隔段的比特数量，将立即发送设定长度的间隔段。

同步段的发送

向 UARTx_TDR 寄存器中写入 0x55，将发送同步段。

受保护 ID 段的发送

受保护 ID 段的前 6 位是帧 ID，加上两个奇偶校验位后称作受保护 ID。

根据校验公式，计算出帧 ID 对应的奇偶校验位，然后将受保护 ID 写入数据寄存器 UARTx_TDR，将发送受保护 ID 段。

16.3.4.2 LIN 接收

同步间隔段的检测

使能 LIN 模式后，将激活同步间隔段检测电路。通过控制寄存器 UARTx_CR3 的 BRKL 位域设置接收间隔段的长度阈值。

使能接收电路后，硬件开始监测 RXD 引脚输入的起始信号。检测到起始位后，电路会对接下来的位进行采样，当检测到 RXD 通信线低电平宽度大于等于 BRKL 位域设定的间隔段长度阈值，且其后跟随间隔符，则 UARTx_ISR.RXBRK 标志位会被硬件置位，表示间隔段接收完成。如果 UARTx_IER.RXBRK 为 1，则会产生中断请求。

如果在设定的间隔段长度阈值之前，已经采样到高电平，则同步间隔段检测电路会取消当前检测，并重新搜索起始位。

LIN 模式下，当发生帧结构错误时，接收器立即停止，直到间隔段检测电路接收到“1”（间隔段不完整）或接收到间隔符（检测到间隔段）。

间隔段检测时，帧结构错误标志位 UARTx_ISR.FE 会被硬件置位，清除 UARTx_ISR.RXBRK 标志位的同时，需要清除 UARTx_ISR.FE 标志，并读取 UARTx_RDR 寄存器，以确认接收到的数据为 0x00。

同步段的检测

从机节点通过接收主机节点发出的同步段，计算出主机节点位速率，根据计算结果对自身的位速率重新作调整。从机节点应设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x6，配置为自动波特率侦测模式 2，通过计数时钟 UCLK 测量同步段的频率。位速率计算公式如下：

1 位时间 =（TIMCNT+（TIMARR+1）×ARR 溢出次数）/（8×UCLK）

如果设置了 UART 为 16 倍采样（OVER=00），则调整后的波特率计数器值如下：

BRRI=（uint16_t）（（TIMCNT+（TIMARR+1）×ARR 溢出次数 /8）≫ 4）

BRRF=（uint16_t）（（TIMCNT+（TIMARR+1）×ARR 溢出次数 /8）& 0x0F）

16.3.5 定时器工作模式

UART 内置定时器模块，包含一个 24bit 向上计数器 TIMCNT 和一个 24bit 自动重载寄存器 TIMARR。定时器支持 5 种工作模式，如下表所示：

表 16-11 定时器工作模式

UARTx_CR2. TIMCR	工作模式
010	等待超时检测模式
011	接收空闲检测模式
101	自动波特率侦测模式 1
110	自动波特率侦测模式 2
111	通用定时模式

16.3.5.1 等待超时检测模式

设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x2，定时器工作于等待超时检测模式，超时时间为：

$$(TIMARR+1)/f_{BaudRate}$$

设置 UARTx_CR2.TIMCR 为 0x2，将立即启动计数器 TIMCNT 从 0 开始向上计数，计数时钟为波特率时钟，每收到一个停止位时从 0 开始重新计数；当计数到重载值 TIMARR 后定时器溢出，同时定时器溢出标志位 UARTx_ISR. TIMOV 被硬件置位，表示接收等待超时，定时器自动停止；如果使能了定时器溢出中断（设置 UARTx_IER.TIMOV 为 1），将执行对应的中断服务程序，退出中断服务程序之前应清除对应的标志位，以避免重复进入中断服务程序。

16.3.5.2 接收空闲检测模式

设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x3，定时器工作于接收空闲检测模式，检测空闲时间为：

$$(TIMARR+1)/f_{BaudRate}$$

每收到一个停止位时，计数器 TIMCNT 在波特率时钟的驱动下从 0 开始向上计数；当计数到重载值 TIMARR 后定时器溢出，同时定时器溢出标志位 UARTx_ISR.TIMOV 被硬件置位，表示接收空闲超时，定时器自动停止；如果使能了定时器溢出中断（设置 UARTx_IER.TIMOV 为 1），将执行对应的中断服务程序，退出中断服务程序之前应清除对应的标志位，以避免重复进入中断服务程序。

16.3.5.3 自动波特率侦测模式

UART 支持两种自动波特率侦测模式，设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x5，配置为自动波特率侦测模式 1；设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x6，配置为自动波特率侦测模式 2。

自动波特率侦测模式 1

UARTx_RXD 引脚的下降沿触发定时器启动，计数器 TIMCNT 在计数时钟 UCLK 的驱动下从 0 开始向上计数，计数到重载值 TIMARR 后定时器溢出，同时定时器溢出标志位 UARTx_ISR.TIMOV 被硬件置位，计数器保持 TIMARR 值不变。

UARTx_RXD 引脚的上升沿停止定时器计数，同时 UARTx_ISR.BAUD 标志位会被硬件置位，表示自动波特率检测已完成。

自动波特率侦测模式 2

自动波特率侦测模式 2 主要用于 LIN 模式下的时钟同步，请参见 16.3.4 LIN 模式。

UARTx_RXD 引脚的下降沿触发定时器启动，计数器 TIMCNT 在计数时钟 UCLK 的驱动下从 0 开始向上计数，计数到重载值 TIMARR 后定时器溢出，同时定时器溢出标志位 UARTx_ISR.TIMOV 被硬件置位，计数器保持 TIMARR 值不变。

UARTx_RXD 引脚的第 4 个下降沿停止定时器计数，同时 UARTx_ISR.BAUD 标志位会被硬件置位，表示自动波特率检测已完成。

16.3.5.4 通用定时模式

UART 内置的定时器模块可作为通用定时器使用，包含一个 24bit 向上计数器，并由计数时钟 UCLK 驱动。

设置控制寄存器 UARTx_CR2 的 TIMCR 位域为 0x7，将立即启动定时器，计数器 TIMCNT 在计数时钟 UCLK 的驱动下累加计数。当超时（TIMARR+1）个 UCLK 时钟后，定时器自动停止。

16.4 低功耗模式

UART 控制器工作在双时钟域下，支持在深度休眠模式下进行正常的数据收发，并通过接收完成中断唤醒 MCU 回到运行模式。

如果设置了传输时钟 UCLK 来源为低速时钟，当系统进入深度休眠模式后，高速时钟将停止，低速时钟保持运行，UART 仍可以进行正常的数据收发（波特率仅支持 2400 bps、4800 bps 和 9600 bps）。

要实现深度休眠模式下使用 UART 唤醒功能，需在进入深度休眠模式之前使能 UART 接收完成中断（即设置 UARTx_IER.RC 为 1），数据接收完成时，接收完成中断将唤醒 MCU 恢复到运行模式。

如果设置了传输时钟 UCLK 来源为高速时钟，当系统进入深度休眠模式后，高速时钟会停止运行，UART 不会接收数据。此时，仍可通过 GPIO 中断唤醒 MCU，实现在深度休眠模式下接收数据，参考配置步骤如下：

步骤 1：使能 UARTx_RXD 对应引脚的 GPIO 下降沿中断；

步骤 2：设置 UARTx_CR1.START 为 1，选择 RXD 信号起始位判定方式为低电平；

步骤 3：使能 UART 接收（即设置 UARTx_CR1.RXEN 为 1）；

步骤 4：进入深度休眠模式；

步骤 5：等待主机发送数据，产生 GPIO 下降沿中断，唤醒 MCU；

步骤 6：关闭 RXD 对应引脚的 GPIO 中断功能，等待 RXD 接收完成。

更多关于低功耗模式进入与唤醒的优化设计，请参见 3 电源控制（PWR）与功耗章节。

16.5 UART 中断

UART 控制器支持 13 个中断源，当 UART 中断触发事件发生时，中断标志位会被硬件置位，如果设置了对应的中断使能控制位，将产生中断请求。

CW32L011 的一个 UART 模块使用一个相同的系统 UART 中断，UART 中断是否产生中断跳转由嵌套向量中断控制器 (NVIC) 的中断使能设置寄存器 NVIC_ISER 的相应位控制。

在用户 UART 中断服务程序中，应查询相关 UART 中断标志位，以进行相应的处理，在退出中断服务程序之前，要清除该中断标志位，避免重复进入中断程序。

各 UART 中断源的标志位、中断使能位、中断标志清除位或清除方法，如下表所示：

表 16-12 UART 中断控制

中断事件	中断标志位	中断使能位	标志清除方法
发送缓冲器空	ISR.TXE	IER.TXE	写 UARTx_TDR 寄存器
发送完成	ISR.TC	IER.TC	写 0 到 ICR.TC
接收完成	ISR.RC	IER.RC	写 0 到 ICR.RC
空闲字符接收完成	ISR.RXIDLE	IER.RXIDLE	写 0 到 ICR.RXIDLE
间隔段接收完成	ISR.RXBRK	IER.RXBRK	写 0 到 ICR. RXBRK
自动波特率检测完成	ISR.BAUD	IER.BAUD	写 0 到 ICR. BAUD
定时器溢出	ISR.TIMOV	IER.TIMOV	写 0 到 ICR. TIMOV
CTS 信号变化	ISR.CTS	IER.CTS	写 0 到 ICR.CTS
帧结构错误	ISR.FE	IER.FE	写 0 到 ICR.FE
奇偶校验错误	ISR.PE	IER.PE	写 0 到 ICR.PE
噪声标志	ISR.NE	IER.NE	写 0 到 ICR.NE
溢出错误	ISR.ORE	IER.ORE	写 0 到 ICR.ORE
接收数据匹配	ISR.RXMATCH	IER.RXMATCH	写 0 到 ICR.RXMATCH

16.6 编程示例

16.6.1 异步全双工编程示例

16.6.1.1 查询方式发送数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 UARTx_TXD 引脚配置成推挽复用输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 0，配置 UARTx 为异步全双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：设置 UARTx_CR1.TXEN 为 1 使能发送；

步骤 9：设置 UARTx_ICR.TC 为 0，清除发送完成标志位；

步骤 10：将要发送的一帧数据写入 UARTx_TDR 寄存器；

步骤 11：查询等待 UARTx_ISR.TC 标志位置 1，确认一帧数据发送完成；

步骤 12：重复步骤 9 至步骤 12，发送下一帧数据。

16.6.1.2 查询方式接收数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 RXD 引脚配置成上拉输入复用模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 0，配置 UARTx 为异步全双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：向 UARTx_ICR 寄存器写入 0x00，清除所有标志位；

步骤 9：设置 UARTx_CR1.RXEN 为 1 使能接收；

步骤 10：查询等待 UARTx_ISR.RC 标志位置 1，确认接收完一帧数据；

步骤 11：查询错误标志 UARTx_ISR.PE 和 UARTx_ISR.FE，确认数据是否有效，如果数据无效，则进行出错处理，如果数据有效，则读取 UARTx_RDR 寄存器并保存数据；

步骤 12：设置 UARTx_ICR.RC 为 0，清除接收完成标志位；

步骤 13：重复步骤 10 至步骤 13，接收下一帧数据。

16.6.1.3 中断方式发送数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 TXD 引脚配置成推挽复用输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 0，配置 UARTx 为异步全双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：配置 NVIC 控制器，请参见 5 中断章节；

步骤 9：设置 UARTx_IER.TXE 为 1，使能发送缓冲器空中断；

步骤 10：设置 UARTx_CR1.TXEN 为 1 使能发送；

步骤 11：将要发送的一帧数据写入 UARTx_TDR 寄存器；

步骤 12：数据开始发送，发送缓冲器空，进入中断服务函数：查询判断 UARTx_ISR.TXE 标志位，如果标志位为 1，写一帧新的数据到 UARTx_TDR 寄存器；

步骤 13：查询等待 UARTx_ISR.TXBUSY 标志位清零，关闭 UART。

16.6.1.4 中断方式接收数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 RXD 引脚配置成上拉输入复用模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 0，配置 UARTx 为异步全双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：配置 NVIC 控制器，请参见 5 中断章节；

步骤 9：设置 UARTx_IER.RC 为 1 使能接收完成中断；

步骤 10：设置 UARTx_CR1.RXEN 为 1 使能接收；

步骤 11：等待一帧数据接收完成，进入中断服务函数：查询 UARTx_ISR.PE 和 UARTx_ISR.FE 标志位，确认数据是否有效，如果无效，则进行出错处理；如果有效，查询判断 UARTx_ISR.RC 标志位，如果标志位为 1，则读取 UARTx_RDR 寄存器并保存数据，并清除该标志位。

16.6.2 同步半双工编程示例

16.6.2.1 查询方式发送数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 GPIO 时钟和 UART 配置时钟；

步骤 2：将 UARTx_TXD、UARTx_RXD 引脚配置成推挽复用输出模式，从机片选引脚 NCS 配置成推挽输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 1，配置 UARTx 为同步半双工通信模式；

步骤 4：配置 UARTx_CR1.SOURCE，选择传输时钟源，配置波特率，具体配置请参见 16.3.2 同步模式；

步骤 5：拉低 NCS 信号，选择某一个从机；

步骤 6：设置 UARTx_CR1.TXEN 为 1 使能发送；

步骤 7：设置 UARTx_ICR.TC 为 0，清除发送完成标志位；

步骤 8：将要发送的 8 位数据写入 UARTx_TDR 寄存器；

步骤 9：查询等待 UARTx_ISR.TC 标志位置 1，确认数据发送完成；

步骤 10：重复步骤 7 至步骤 10，发送下一帧数据；

步骤 11：拉高 NCS 信号，结束通信。

16.6.2.2 查询方式接收数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 GPIO 时钟和 UART 配置时钟；

步骤 2：将 UARTx_TXD、UARTx_RXD 引脚配置成推挽复用输出模式，从机片选引脚 NCS 配置成推挽输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SYNC 为 1，配置 UARTx 为同步半双工通信模式；

步骤 4：配置 UARTx_CR1.SOURCE，选择传输时钟源，配置波特率，具体配置请参见 16.3.2 同步模式；

步骤 5：拉低 NCS 信号，选择某一个从机；

步骤 6：向 UARTx_ICR 写入 0x00，清除所有标志位；

步骤 7：设置 UARTx_CR1.RXEN 为 1 使能接收；

步骤 8：查询等待 UARTx_ISR.RC 标志位置 1，确认数据接收完成；

步骤 9：读 UARTx_RDR 寄存器并保存数据；

步骤 10：设置 UARTx_ICR.RC 为 0，清除接收完成标志位；

步骤 11：重复步骤 8 至步骤 11，接收下一个 8 位数据；

步骤 12：拉高 NCS 信号，结束通信。

16.6.3 单线半双工编程示例

16.6.3.1 查询方式发送数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 UARTx_TXD 引脚配置成开漏复用输出模式，并外接上拉电阻，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SIGNAL 为 1，配置 UARTx 为单线半双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：设置 UARTx_CR1.TXEN 为 1，UARTx_CR1.RXEN 为 1 使能发送和接收；

步骤 9：设置 UARTx_ICR.TC 为 0，清除发送完成标志位；

步骤 10：将要发送的一帧数据写入 UARTx_TDR 寄存器；

步骤 11：查询等待 UARTx_ISR.TC 标志位置 1，确认数据发送完成；

步骤 12：重复步骤 9 至步骤 12，发送下一帧数据。

16.6.3.2 查询方式接收数据

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBENx.UARTx 为 1，使能 UART 引脚对应的 GPIO 时钟和 UART 配置时钟；

步骤 2：将 UARTx_TXD 引脚配置成开漏复用输出模式，并外接上拉电阻，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 UARTx_CR1.SIGNAL 为 1，配置 UARTx 为单线半双工通信模式；

步骤 4：配置数据帧；

1. 起始位判定方式：配置 UARTx_CR1.START
2. 数据字长：配置 UARTx_CR1.CHLEN
3. 校验位：配置 UARTx_CR1.PARITYEN 和 UARTx_CR1.PARITY
4. 停止位：配置 UARTx_CR1.STOP

步骤 5：配置 UARTx_CR1.SOURCE，选择传输时钟源；

步骤 6：配置 UARTx_CR1.OVER，选择采样模式；

步骤 7：配置 UARTx_BRRI 和 UARTx_BRRF 寄存器，配置波特率，具体配置请参见 16.3.3.2 小数波特率发生器；

步骤 8：向 UARTx_ICR 写入 0x00，清除所有标志位；

步骤 9：设置 UARTx_CR1.TXEN 为 1，UARTx_CR1.RXEN 为 1 使能发送和接收；

步骤 10：查询等待 UARTx_ISR.RC 标志位置 1，确认接收完一帧数据；

步骤 11：查询错误标志 UARTx_ISR.PE 和 UARTx_ISR.FE，确认数据是否有效，如果数据无效，则进行出错处理，如果数据有效，则读取 UARTx_RDR 寄存器并保存数据；

步骤 12：设置 UARTx_ICR.RC 为 0，清除接收完成标志位；

步骤 13：重复步骤 10 至步骤 13，接收下一帧数据。

16.7 寄存器列表

UART1 基地址：UART1_BASE = 0x4000 0C00

UART2 基地址：UART2_BASE = 0x4000 1000

UART3 基地址：UART3_BASE = 0x4000 2000

表 16-13 UART 寄存器列表

寄存器名称	寄存器地址	寄存器描述
UARTx_CR1	UARTx_BASE + 0x00	控制寄存器 1
UARTx_CR2	UARTx _BASE + 0x04	控制寄存器 2
UARTx_CR3	UARTx_BASE + 0x38	控制寄存器 3
UARTx_BRRI	UARTx_BASE + 0x0C	波特率计数器整数部分寄存器
UARTx_BRRF	UARTx_BASE + 0x10	波特率计数器小数部分寄存器
UARTx_TIMARR	UARTx_BASE + 0x14	定时器重载值寄存器
UARTx_TIMCNT	UARTx_BASE + 0x18	定时器计数值寄存器
UARTx_IER	UARTx_BASE + 0x08	中断使能寄存器
UARTx_ISR	UARTx_BASE + 0x1C	中断标志寄存器
UARTx_ICR	UARTx_BASE + 0x20	中断标志清除寄存器
UARTx_RDR	UARTx_BASE + 0x24	接收数据寄存器
UARTx_TDR	UARTx_BASE + 0x28	发送数据寄存器
UARTx_ADDR	UARTx_BASE + 0x30	从机地址寄存器
UARTx_MASK	UARTx_BASE + 0x34	从机地址掩码寄存器
UARTx_RXMATCH	UARTx_BASE + 0x3C	接收数据匹配寄存器

16.8 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

16.8.1 UARTx_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:15	RFU	-	保留位，请保持默认值
14	SYNC	RW	传输同步异步模式设置 0：工作于异步模式 1：工作于同步模式 *注：LIN 模式下，必须保持该控制位为 0。*
13:12	SOURCE	RW	UART 传输时钟来源配置 00：UCLK 来自 PCLK 01：UCLK 来自 PCLK 10：UCLK 来自 LSE 11：UCLK 来自 LSI
11	SIGNAL	RW	单总线模式使能控制 0：禁止，通过 TXD 端口发送数据，通过 RXD 端口接收数据 1：使能，通过 TXD 端口进行数据收发 *注：LIN 模式下，必须保持该控制位为 0。*
10:9	OVER	RW	采样方式设置 00：16 倍采样， Baud = UCLK / (BRRI×16 + BRRF) 01：8 倍采样， Baud = UCLK / (BRRI×8) 10：4 倍采样， Baud = UCLK / (BRRI×4) 11：专用采样， Baud = UCLK×256 / BRRI *注 1：专用采样仅适合传输时钟为 LSI/LSE 时进行 2400、4800、9600 通信。* *注 2：当 BRRF 为非零值，自动采用 16 倍采样。*
8	START	RW	RXD 信号 START 位判定方式设置 0：RXD 信号下降沿 1：RXD 信号低电平
7	MSBF	RW	数据帧高低位顺序选择 0：最低有效位 LSB 收发在前 1：最高有效位 MSB 收发在前
6	CHLEN	RW	字符长度 0：8bit 1：9bit *注：如果使能奇偶校验，则字符的最后 1Bit 为奇偶校验位。*

位域	名称	权限	功能描述
5:4	STOP	RW	停止位长度设置 00：1 位 01：1.5 位 10：2 位 11：保留 *注 1：同步模式下 STOP 位必须清零。* *注 2：LIN 模式下 STOP 位必须清零。*
3	PARITYEN	RW	奇偶校验使能 0：禁止 1：使能 *注：当字符长度为 8bit 时，奇偶校验使能自动清零。*
2	PARITY	RW	奇偶校验选择 0：偶校验 1：奇校验
1	RXEN	RW	接收电路使能控制 0：禁止 1：使能
0	TXEN	RW	发送电路使能控制 0：禁止 1：使能

16.8.2 UARTx_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17:15	RXSRC	RW	RXD 输入信号来源配置 000：UARTx_RXD 引脚 001：VC1_OUT 010：VC2_OUT 其他：保留
14	LOOP	RW	LoopBack 模式使能控制 0：RXD 选择外部 RXD 引脚 1：RXD 选择内部 TXD
13	ADCTX	RW	发送缓冲区空触发 ADC 使能控制 0：禁止 1：使能
12	ADCRX	RW	接收数据完成触发 ADC 使能控制 0：禁止 1：使能
11	SWAP	RW	TXD/RXD 引脚交换使能控制 0：不进行引脚交换 1：进行引脚交换
10:8	TIMCR	RW	定时器工作模式配置 000：立即停止定时器 010：等待超时检测模式 立即启动定时器从 0 开始计数，每收到一个 STOP 位时从 0 开始计数；计数时钟为波特率时钟；超时后自动停止。 011：接收空闲检测模式 每收到一个 STOP 位时从 0 开始计数；计数时钟为波特率时钟；超时后自动停止。 101：自动波特率侦测模式 1 RXD 下降沿使定时器启动从 0 开始计数，上升沿停止定时器；计数时钟为 UCLK；检测完成自动停止。 110：自动波特率侦测模式 2 RXD 下降沿使定时器启动从 0 开始计数，第 4 个下降沿停止定时器；计数时钟为 UCLK；检测完成自动停止。 111：通用定时功能 立即启动定时器从 0 开始计数；计数时钟为 UCLK；超时（TIMARR+1）个 UCLK 后自动停止。
7:6	RFU	-	保留位，请保持默认值
5	TXINV	RW	TXD 引脚输出信号反相控制 0：TXD 引脚输出正相信号 1：TXD 引脚输出反相信号

位域	名称	权限	功能描述
4	RXINV	RW	RXD 引脚输入信号反相控制 0：RXD 引脚信号直接送入接收电路 1：RXD 引脚信号反相送入接收电路
3	RTSEN	RW	RTS 硬件信号使能控制 0：禁止 1：使能
2	CTSEN	RW	CTS 硬件信号使能控制 0：禁止 1：使能
1	RXMATCHEN	RW	接收数据匹配检测使能控制 0：禁止 1：使能
0	ADDREN	RW	从机地址识别使能控制 0：禁止 1：使能

16.8.3 UARTx_CR3 控制寄存器 3

Address offset: 0x38 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8	BRKL	RW	LIN 模式，接收间隔段的长度阈值 0：10bit 1：11bit
7	LIN	RW	LIN 工作模式使能 0：工作于 UART 模式 1：工作于 LIN 模式
6:2	DETIME	RW	驱动器使能及禁止时间 输出 DE 有效电平并等待该时间后，开始发送 START 位； STOP 位发送完成并等待该时间后，输出 DE 无效电平。 该时间 = 采样时间 ×DETIME *注 1：采样时间为 1/4、1/8、1/16 比特时间。* *注 2：OVER=3（专用采样）时不支持 DE 功能。*
1	DEP	RW	驱动器输出信号极性配置 0：DE 信号高电平有效 1：DE 信号低电平有效
0	DEM	RW	驱动器使能配置 0：禁止驱动器功能，RTS 引脚为 RTS 功能 1：使能驱动器功能，RTS 引脚为 DE 功能 *注：该控制位优先级高于 UARTx_CR2.RTSEN。*

16.8.4 UARTx_BRRI 波特率计数器整数部分寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	BRRI	RW	波特率计数器整数部分

16.8.5 UARTx_BRRF 波特率计数器小数部分寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3:0	BRRF	RW	波特率计数器小数部分

16.8.6 UARTx_TIMARR 定时器重载值寄存器

Address offset: 0x14 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23:0	TIMARR	RW	定时器重载值

16.8.7 UARTx_TIMCNT 定时器计数值寄存器

Address offset: 0x18 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位，请保持默认值
23:0	TIMCNT	RO	定时器计数值

16.8.8 UARTx_TDR 发送数据寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:11	RFU	-	保留位，请保持默认值
10	BREAK	WO	发送间隔段帧 0：发送正常数据 1：发送间隔段帧 *注：LIN 模式的间隔段长度由 UARTx_TDR[8:0] 的值决定；UART 模式的间隔段长度由 UART 字长决定。*
9	IDLE	WO	发送空闲帧 0：发送正常数据 1：发送空闲帧 *注 1：空闲帧长度由 UART 字长决定；* *注 2：空闲帧优先级高于间隔帧。*
8:0	TDR	WO	待发送的数据或 LIN 模式间隔段的长度 *注：LIN 模式下发送间隔段时，写入此寄存器的数据作为间隔段的比特数量。*

16.8.9 UARTx_RDR 接收数据寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8:0	RDR	RO	已接收的数据

16.8.10 UARTx_IER 中断使能寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:13	RFU	-	保留位，请保持默认值
12	RXMATCH	RW	接收数据匹配中断使能控制 0：禁止 1：使能
11	ORE	RW	溢出中断使能控制 0：禁止 1：使能
10	NE	RW	噪声中断使能控制 0：禁止 1：使能
9	PE	RW	奇偶校验错误中断使能控制 0：禁止 1：使能
8	FE	RW	帧结构错误中断使能控制 0：禁止 1：使能
7	CTS	RW	CTS 信号变化中断使能控制 0：禁止 1：使能
6	TIMOV	RW	定时器溢出中断使能控制 0：禁止 1：使能
5	BAUD	RW	自动波特率检测完成中断使能控制 0：禁止 1：使能
4	RXBRK	RW	间隔段接收完成中断使能控制 0：禁止 1：使能
3	RXIDLE	RW	空闲字符接收完成中断使能控制 0：禁止 1：使能
2	RC	RW	接收完成中断使能控制 0：禁止 1：使能
1	TC	RW	发送完成中断使能控制 0：禁止 1：使能

位域	名称	权限	功能描述
0	TXE	RW	发送缓冲器空中断使能控制 0：禁止 1：使能

16.8.11 UARTx_ISR 中断标志寄存器

Address offset: 0x1C Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	CTSLV	RO	CTS 信号电平状态 0：CTS 信号为低电平 1：CTS 信号为高电平
14	TXBUSY	RO	TXD 发送状态 0：TDR 寄存器及发送移位寄存器空 1：TDR 寄存器或发送移位寄存器非空
13	SLVMATCH	RO	接收从机地址匹配标志 0：未检测到匹配的地址 1：已检测到匹配的地址
12	RXMATCH	RO	接收数据匹配标志 0：接收到的数据与 UARTx_RXMATCH 不同 1：接收到的数据与 UARTx_RXMATCH 相同
11	ORE	RO	溢出标志 UARTx_ISR.RC 为 1 时，当移位寄存器中当前正在接收的数据准备好传输至 UARTx_RDR 寄存器时，硬件将该位置位。 0：无溢出错误 1：检测到溢出错误 *注：发生溢出错误后，新接收的数据会覆盖 UARTx_RDR 寄存器中的数据。*
10	NE	RO	噪声标志 0：未检测到噪声 1：检测到噪声 *注：OVER 为 2 或 3 时，该位无效，保持为 0。*
9	PE	RO	奇偶校验错误中断标志 0：未检测到奇偶校验错误 1：已检测到奇偶校验错误
8	FE	RO	帧结构错误中断标志 0：未检测到帧结构错误 1：已检测到帧结构错误

位域	名称	权限	功能描述
7	CTS	RO	CTS 信号变化中断标志 0：未检测到 CTS 电平变化 1：已检测到 CTS 电平变化
6	TIMOV	RO	定时器溢出标志 0：定时器未溢出（TIMCNT ＜ TIMARR） 1：定时器已溢出（TIMCNT ≧ TIMARR）
5	BAUD	RO	自动波特率检测完成标志 0：自动波特率检测未完成 1：自动波特率检测已完成
4	RXBRK	RO	间隔段接收完成标志 0：未接收到间隔段 1：已接收到间隔段
3	RXIDLE	RO	空闲字符接收完成标志 0：未接收到空闲字符 1：已接收到空闲字符 *注：每当接收完成一个字符时，才会启动空闲字符检测。*
2	RC	RO	接收完成中断标志 0：尚未完成一帧数据的接收 1：已经完成一帧数据的接收 *注：STOP 为 1bit，RC 收到 1bit STOP 位置 1；* *STOP 为 1.5bit，RC 收到 0.5bit STOP 位置 1；* *STOP 为 2bit，RC 收到 0.5bit STOP 位置 1。*
1	TC	RO	发送完成中断标志 0：发送缓冲器或移位寄存器中的数据尚未发送完成 1：发送缓冲器和移位寄存器中的数据均已发送完成
0	TXE	RO	发送缓冲器空标志 0：发送缓冲器内有数据 1：发送缓冲器内无数据 *注：写 UARTx_TDR 寄存器将清除该标志位。*

16.8.12 UARTx_ICR 中断标志清除寄存器

Address offset: 0x20 Reset value: 0x0000 1FFF

位域	名称	权限	功能描述
31:13	RFU	-	保留位，请保持默认值
12	RXMATCH	R1W0	接收数据匹配标志清除控制 W0：清除接收数据匹配标志 W1：无功能
11	ORE	R1W0	溢出标志清除控制 W0：清除溢出标志 W1：无功能
10	NE	R1W0	噪声标志清除控制 W0：清除噪声标志 W1：无功能
9	PE	R1W0	奇偶校验错误中断标志清除控制 W0：清除奇偶校验错误中断标志 W1：无功能
8	FE	R1W0	帧结构错误中断标志清除控制 W0：清除帧结构错误中断标志 W1：无功能
7	CTS	R1W0	CTS 信号变化中断标志清除控制 W0：清除 CTS 信号变化中断标志 W1：无功能
6	TIMOV	R1W0	定时器溢出标志清除控制 W0：清除定时器溢出标志 W1：无功能
5	BAUD	R1W0	自动波特率检测完成标志清除控制 W0：清除自动波特率检测完成标志 W1：无功能
4	RXBRK	R1W0	间隔段接收完成标志清除控制 W0：清除间隔段接收完成标志 W1：无功能
3	RXIDLE	R1W0	空闲字符接收完成标志清除控制 W0：清除空闲字符接收完成标志 W1：无功能
2	RC	R1W0	接收完成中断标志清除控制 W0：清除接收完成中断标志 W1：无功能
1	TC	R1W0	发送完成中断标志清除控制 W0：清除发送完成中断标志 W1：无功能
0	RFU	-	保留位，请保持默认值

16.8.13 UARTx_ADDR 从机地址寄存器

Address offset: 0x30 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	ADDR	RW	从机地址寄存器

16.8.14 UARTx_MASK 从机地址掩码寄存器

Address offset: 0x34 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	MASK	RW	从机地址掩码寄存器

16.8.15 UARTx_RXMATCH 接收数据匹配寄存器

Address offset: 0x3C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8:0	RXMATCH	RW	待匹配的目标接收数据

17 串行外设接口（SPI）

17.1 概述

串行外设接口（SPI）是一种同步串行数据通信接口，常用于 MCU 与外部设备之间进行同步串行通信。CW32L011 内部集成 1 个串行外设 SPI 接口，支持双向全双工、单线半双工和单工通信模式，可配置 MCU 作为主机或从机，支持多主机通信模式。

17.2 主要特性

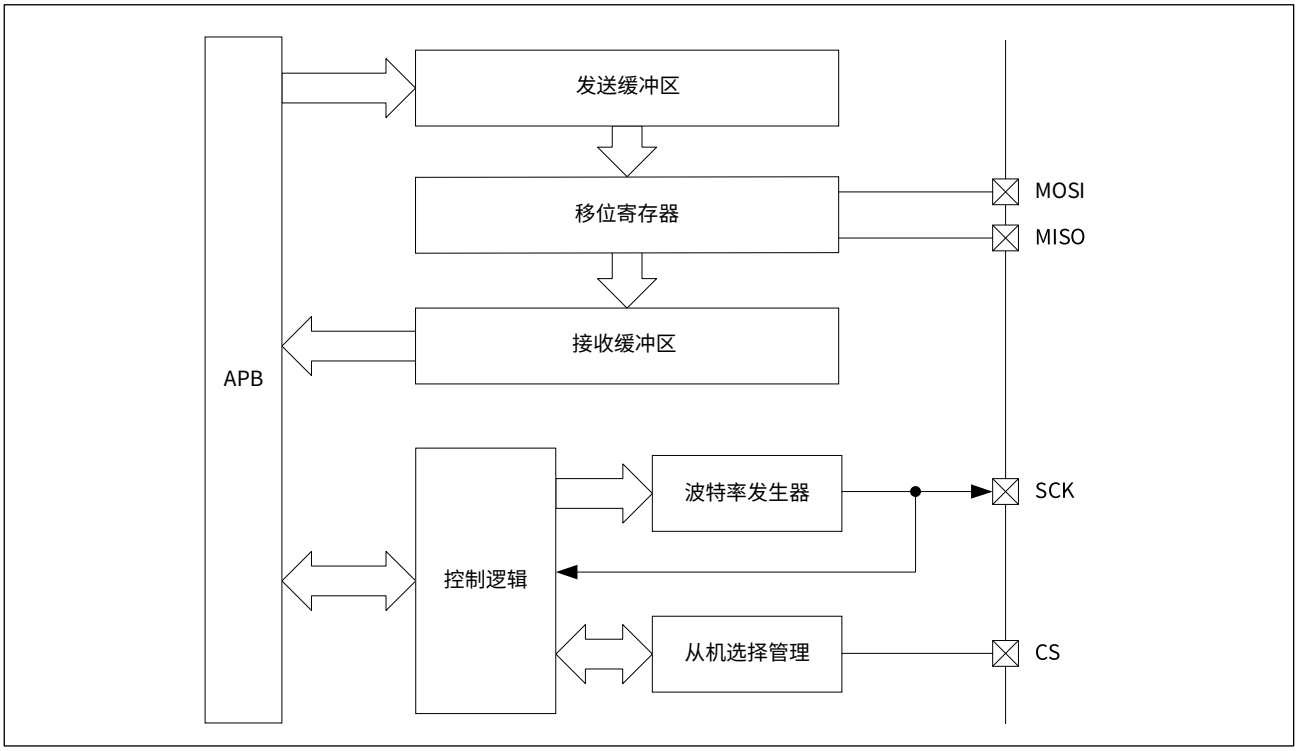
- 支持主机模式、从机模式
- 支持全双工、单线半双工、单工
- 可选的 4 位到 16 位数据帧宽度
- 支持收发数据 LSB 或 MSB 在前
- 可编程时钟极性和时钟相位
- 主机模式下通信速率高达 PCLK/2
- 支持多机通信模式
- 8 个带标志位的中断源
- 主机模式支持帧间间隔调整
- 发送缓冲器空 / 接收数据完成触发启动 ADC

17.3 功能描述

17.3.1 功能框图

SPI 控制器的功能框图如下图所示：

图 17-1 SPI 功能框图



SPI 一般通过 4 个引脚与外部设备相连：

- MOSI
  主机输出 / 从机输入，用于主机模式下的数据发送和从机模式下的数据接收。
- MISO
  主机输入 / 从机输出，用于主机模式下的数据接收和从机模式下的数据发送。
- SCK
  同步串行时钟，主机时钟输出和从机时钟输入，发送或接收主机同步时钟。
- CS
  从机选择，也用于多机通信时总线冲突检测，参见 17.3.3 从机选择引脚 CS。

CW32L011 支持用户灵活选择 GPIO 作为 SPI 通信引脚，通过 AFR 功能复用实现，具体 SPI 引脚请参考数据手册引脚定义。

SPI 支持多种工作模式，在不同的工作模式下，各引脚具有不同的功能，引脚配置如下表所示：

表 17-1 SPI 端口配置

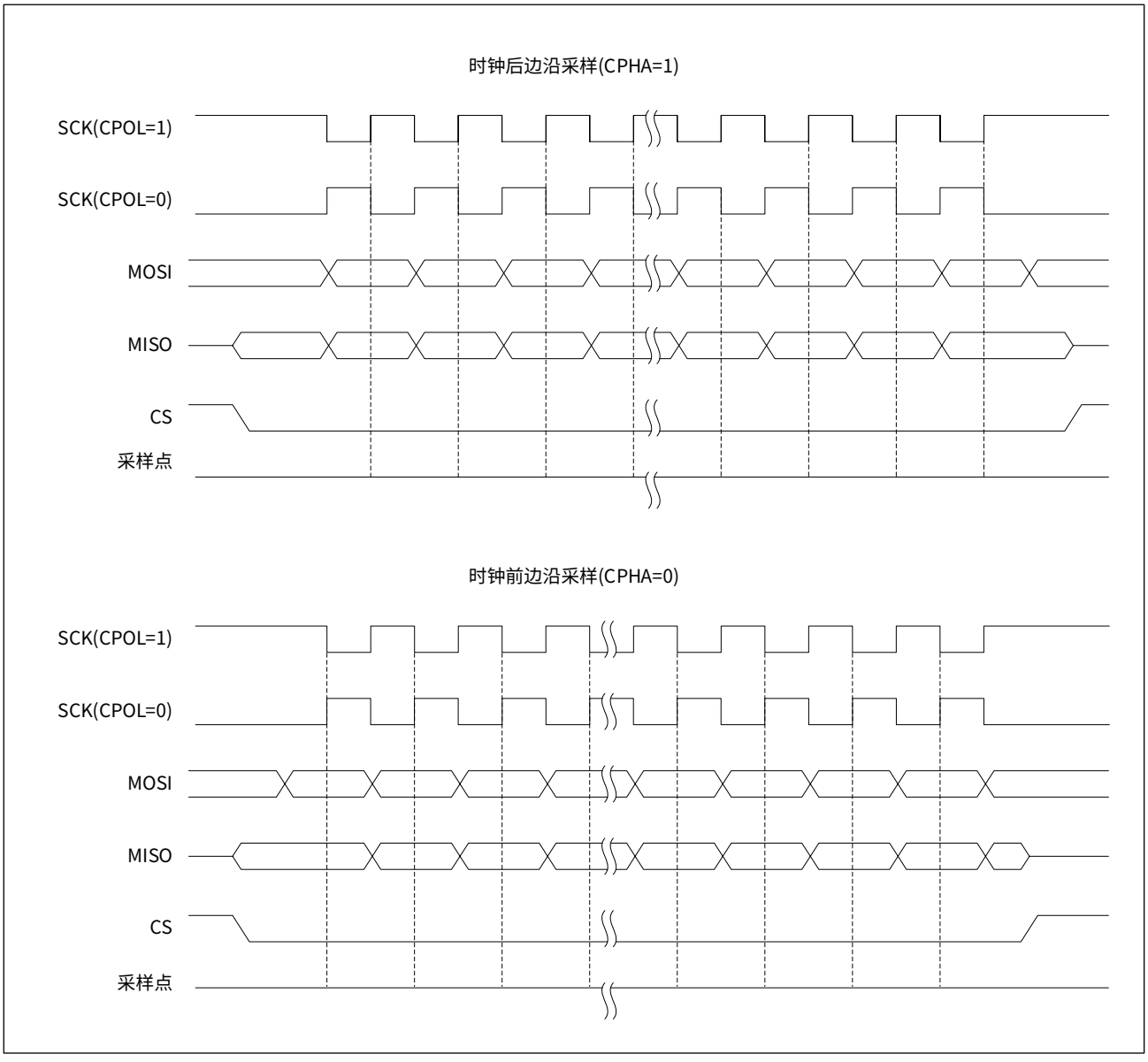
SPI 引脚	模式配置	作用	GPIO 配置
SPI_SCK	主机	串行时钟输出	数字，推挽输出，复用
	从机	串行时钟输入	数字，浮空输入，复用
SPI_MOSI	全双工 / 主机	主机数据输出	数字，推挽输出，复用
	全双工 / 从机	从机数据输入	数字，浮空输入，复用
	单线半双工 / 主机	主机数据收发	数字，推挽输出，复用
	单线半双工 / 从机	不使用	可作为通用 I/O
	单工单发 / 主机	主机数据输出	数字，推挽输出，复用
	单工单发 / 从机	不使用	可作为通用 I/O
	单工单收 / 主机	不使用	可作为通用 I/O
	单工单收 / 从机	从机数据输入	数字，浮空输入，复用
SPI_MISO	全双工 / 主机	主机数据输入	数字，浮空输入，复用
	全双工 / 从机	从机数据输出	数字，推挽输出，复用
	单线半双工 / 主机	不使用	可作为通用 I/O
	单线半双工 / 从机	从机数据收发	数字，推挽输出，复用
	单工单发 / 主机	不使用	可作为通用 I/O
	单工单发 / 从机	从机数据输出	数字，推挽输出，复用
	单工单收 / 主机	主机数据输出	数字，推挽输出，复用
	单工单收 / 从机	不使用	可作为通用 I/O
SPI_CS	主机（SSM=0）	从机选择输入	数字，浮空输入，复用
	主机（SSM=1）	从机选择输出	数字，推挽输出，复用
	从机（SSM=0）	从机选择输入	数字，浮空输入，复用
	从机（SSM=1）	不使用	可作为通用 I/O

17.3.2 通信时序和数据格式

通信时序

下图是 SPI 的通信时序，CS、SCK、MOSI 信号都由主机控制产生，MISO 信号是从机响应信号。从机选择 CS 信号由高电平变低，是 SPI 通信的起始信号，当从机检测到起始信号后，开始与主机通信。在每个 SCK 的时钟周期中，MOSI 和 MISO 信号线传输一位数据。CS 由低电平变高，是通信的停止信号，表示本次通信结束。

图 17-2 SPI 通信时序



数据帧格式

数据帧宽度由控制寄存器 SPI_CR1 的 WIDTH 位域配置，可设置 4 ~ 16bit 任意数据位宽。

数据的大小端由控制寄存器 SPI_CR1 的 LSBF 位域配置，可选择最高有效位在前（MSB）或最低有效位在前（LSB）。

主机模式帧间隔由控制寄存器 SPI_CR1 的 GAP 位域配置，可选择 0~15 个 SCK 时钟周期间隔。

时钟频率

同步串行时钟 SCK 信号由 SPI 主机控制产生，其时钟来源是 PCLK，通过配置控制寄存器 SPI_CR1 的 BR 位域来设置分频因子，可选择 2 ~ 256 分频。对于 SPI 从机，配置 SPI_CR1.BR 无影响。

当 SPI 通信速率明显低于 PCLK 时钟频率（$f_{SCK} \leqslant f_{PCLK}/16$）时，可设置 SPI_CR1.FLTEN 为 1 使能 SPI 滤波功能。

时钟极性、时钟相位

时钟极性 CPOL，指设备处于没有数据传输的空闲状态时，SCK 串行时钟线的电平状态。通过控制寄存器 SPI_CR1 的 CPOL 位域进行配置：设置 SPI_CR1.CPOL 为 0，SCK 时钟线在空闲时为低电平；设置 SPI_CR1.CPOL 为 1，SCK 时钟线在空闲时为高电平。

时钟相位 CPHA，指数据的采样和移位时刻。通过控制寄存器 SPI_CR1 的 CPHA 进行配置：设置 SPI_CR1.CPHA 为 0，在 SCK 的前边沿（SCK 由空闲状态变为非空闲状态的时钟边沿）采样、后边沿（SCK 由非空闲状态变为空闲状态的时钟边沿）移位；设置 SPI_CR1.CPHA 为 1，在 SCK 的前边沿移位、后边沿采样。

根据时钟极性 CPOL 和时钟相位 CPHA 的不同配置，SPI 可设置 4 种电平模式，主机和从机需要配置成相同的电平模式才能保证正常通信。

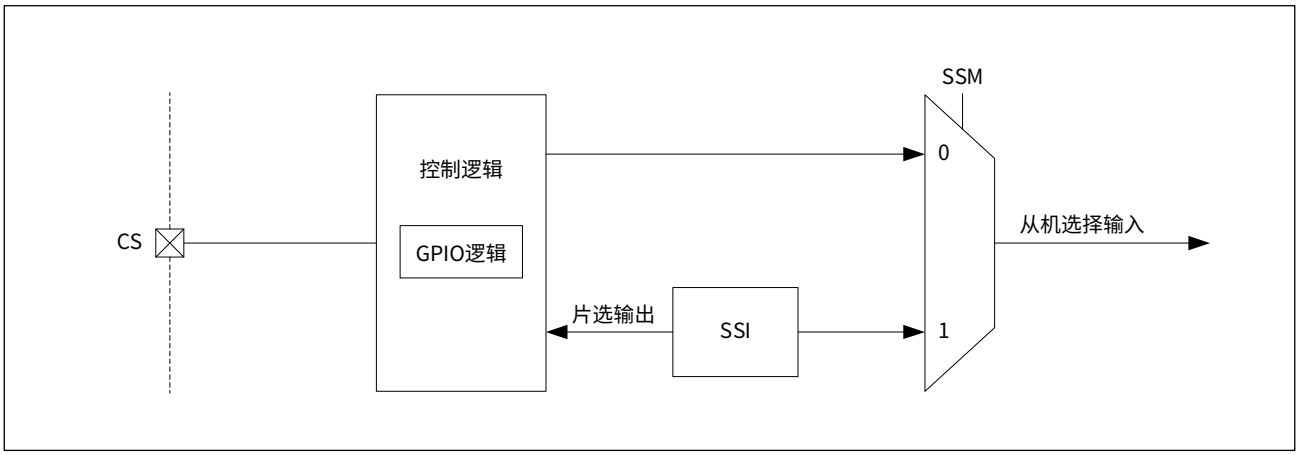
表 17-2　SPI 电平模式

CPOL	CPHA	空闲时 SCK 时钟	采样时刻
0	0	低电平	前边沿
0	1	低电平	后边沿
1	0	高电平	前边沿
1	1	高电平	后边沿

17.3.3 从机选择引脚 CS

下图所示为从机选择引脚 CS 的配置管理示意图。在主机模式下，CS 可选择作为输入或输出，作为输入时用于检测多主机模式下的总线冲突，作为输出时用于产生从机片选信号。在从机模式下，CS 作为一个器件片选输入。

图 17-3 从机选择引脚管理



在主机和从机模式下，通过控制寄存器 SPI_CR1 的 SSM 位域来管理从机选择引脚 CS。

主机模式下 CS 引脚

设置 SPI_CR1.SSM 为 0，CS 被配置为输入，用于检测多机通信系统中的总线是否发生冲突（注意，此时主机需通过其它 GPIO 来选择需通信的从机），请参见 17.3.7 多机通信。

设置 SPI_CR1.SSM 为 1，从机选择引脚 CS 被配置为输出，用于产生从机选择信号。CS 引脚输出电平由从机选择寄存器 SPI_SSI 的 SSI 位域决定：设置 SPI_SSI.SSI 为 1，CS 输出高电平；设置 SPI_SSI.SSI 为 0，CS 输出低电平。

从机模式下 CS 引脚

设置 SPI_CR1.SSM 为 0，使用 CS 信号作为器件选择。CS 电平决定本机是否被选中：CS 为低电平时本机被选中；CS 为高电平时本机被取消选中。

设置 SPI_CR1.SSM 为 1，不使用 CS 作为器件选择，从机选择由 SPI_SSI 寄存器的 SSI 位域值决定。设置 SPI_SSI.SSI 为 0，本机被选中；设置 SPI_SSI.SSI 为 1，本机被取消选中。这种配置下用户程序以软件方式设置本机是否被选中，CS 引脚可作普通 IO 使用。

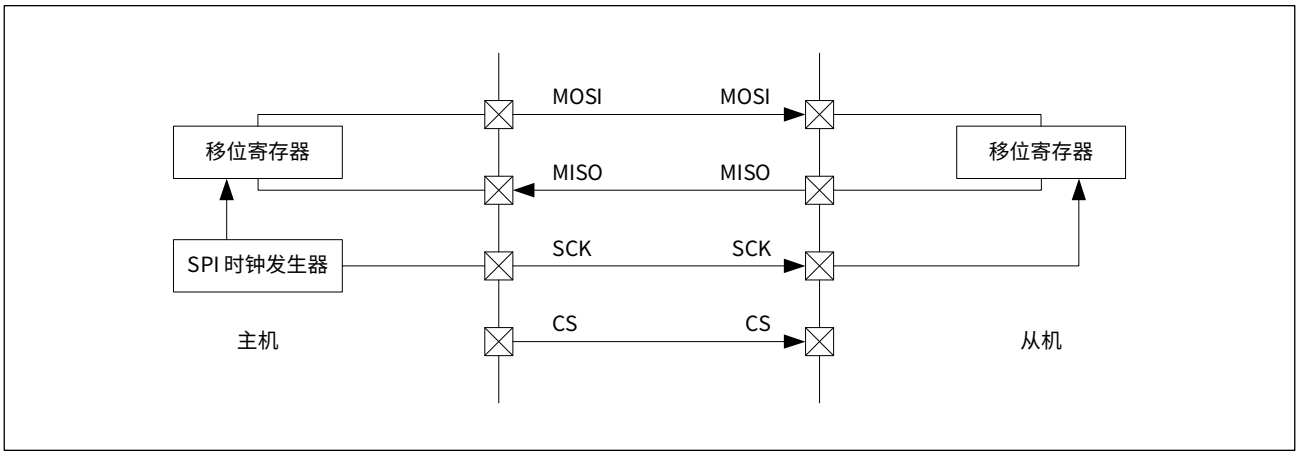
17.3.4 全双工模式

SPI 支持全双工通信模式，在该模式下，主机和从机的移位寄存器通过 MOSI、MISO 两条单向数据线进行连接，在主机提供的 SCK 时钟信号的控制下同时进行两个线路数据传输。

设置控制寄存器 SPI_CR1 的 MODE 位域为 0x0，使 SPI 工作于全双工通信模式。

全双工模式的典型应用框图下图所示：

图 17-4 全双工通信模式



主机收发

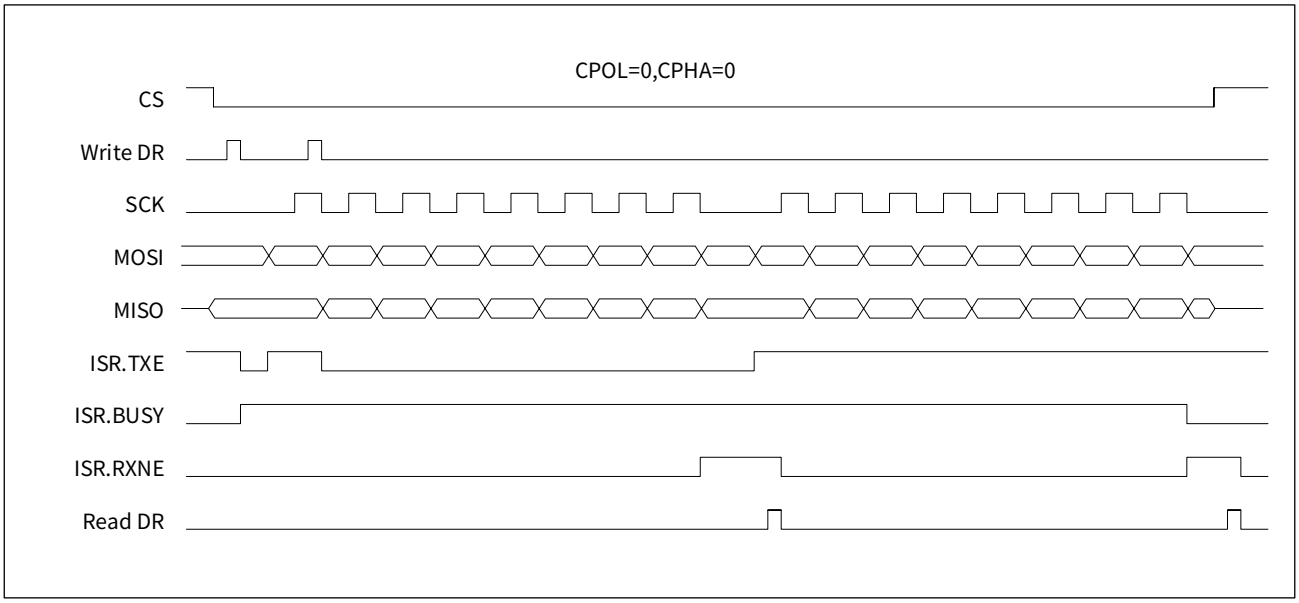
设置 SPI_CR1.MSTR 为 1，SPI 工作于主机模式。

主机设置 SPI_SSI.SSI 为 0，在从机选择 CS 引脚输出低电平，作为通信起始信号。

当发送缓冲器为空时，即 SPI_ISR.TXE 标志位为 1，将待发送的一帧数据写入 SPI_DR 寄存器，数据在同步移位时钟信号的控制下，从 MOSI 引脚输出，同时将 MISO 引脚的数据接收到移位寄存器。发送一个数据帧结束时，接收缓冲器非空标志位 SPI_ISR.RXNE 由硬件置 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

设置 SPI_SSI.SSI 为 1，从机选择引脚 CS 输出高电平，结束本次通信。

图 17-5 全双工模式下主机收发时序（两个字节）



从机收发

设置 SPI_CR1.MSTR 为 0，SPI 工作于从机模式。

在从机选择 CS 信号被拉低之前，从机需要设置 SPI_ICR.FLUSH 为 0，以清空发送缓冲区和移位寄存器，并将待发送的第一帧数据写入 SPI_DR 寄存器。

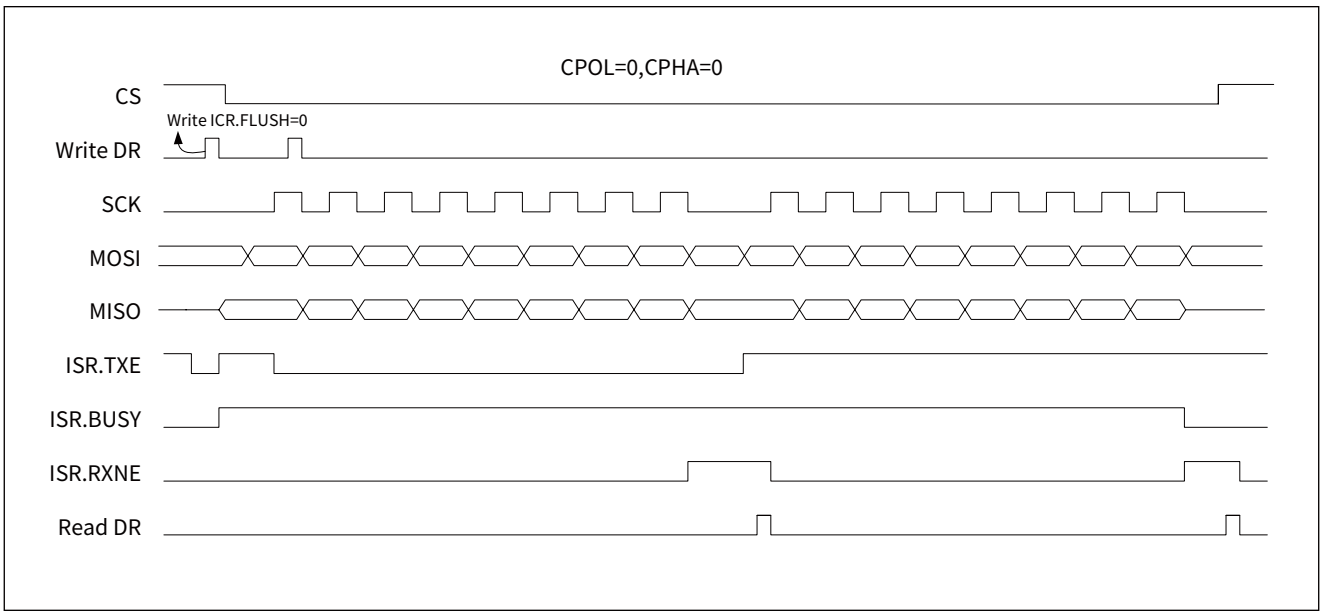
当 CS 信号被拉低后，被写入的数据将在主机的同步移位时钟信号的控制下，从 MISO 引脚输出，同时接收 MOSI 引脚的数据到移位寄存器。

如果是多数据帧连续通信，用户应当不断查询 SPI_ISR.TXE 标志位，一旦标志为 1，立即将待发送的数据写入 SPI_DR 寄存器，以免出现数据漏发。

当接收缓冲器非空时，即 SPI_ISR.RXNE 标志位为 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

当检测到 CS 引脚变为高电平时，本次通信结束。

图 17-6　全双工模式下从机收发时序（两个字节）



17.3.5 单线半双工模式

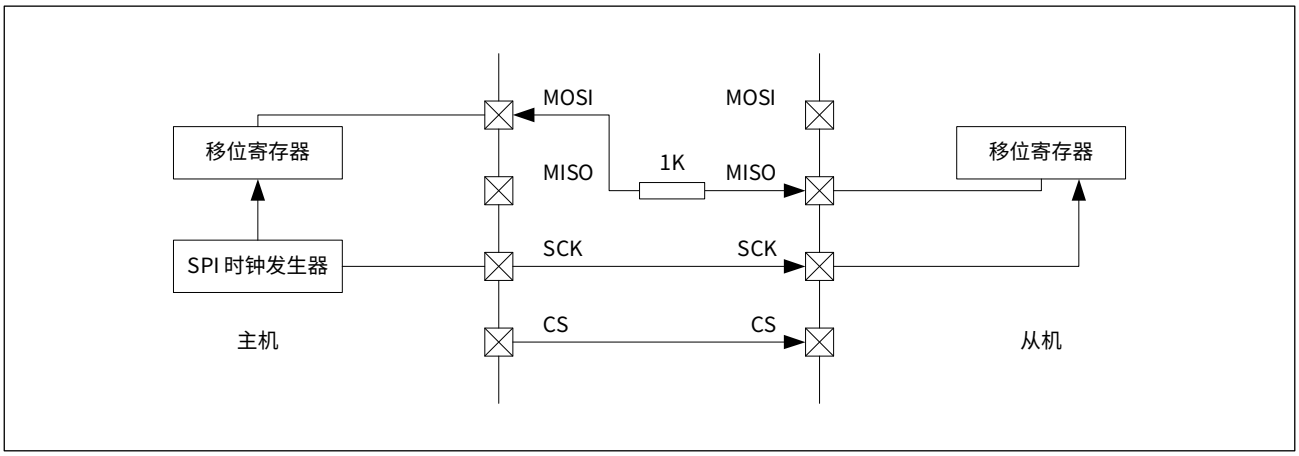
SPI 支持单线半双工通信模式，在该模式下，主机和从机通过一条双向数据线进行数据通信，主机使用 MOSI，从机使用 MISO，未使用的其他 SPI 信号线可供其它功能使用。

设置控制寄存器 SPI_CR1 的 MODE 位域为 0x3，使 SPI 工作于单线半双工模式。

控制寄存器 SPI_CR3 的 HDOE 位域，选择收发方向，HDOE 为 0 仅接收，HDOE 为 1 仅发送。

单线半双工模式的典型应用框图，如下图所示：

图 17-7 单线半双工模式



单线半双工通信时，由于主机和从机的通信方向无法保证同步切换 ，有可能会出现驱动冲突而导致器件损害，建议在主机和从机之间的数据线上串联一个电阻，以限制电流。

当 SPI 作为从机时，使用单线半双工模式，可以实现与一个标准的 UART 收发器进行同步模式通信，此时 SPI 必须配置为固定的电平模式：时钟极性 CPOL 为 1，时钟相位 CPHA 为 1。

主机发送

设置 SPI_CR1.MSTR 为 1，SPI 工作于主机模式；设置 SPI_CR3.HDOE 为 1，主机仅发送数据。

主机设置 SPI_SSI.SSI 为 0，在从机选择引脚 CS 输出低电平，作为通信起始信号。

当发送缓冲器为空时，即 SPI_ISR.TXE 标志位为 1，将待发送的一帧数据写入 SPI_DR 寄存器，数据在同步移位时钟信号的控制下从 MOSI 引脚输出。

当写入最后一帧数据后，必须等待发送缓冲器为空，即 SPI_ISR.TXE 标志位变为 1，同时 SPI_ISR.BUSY 标志位变为 0，以确保数据发送完毕。然后设置 SPI_SSI.SSI 为 1，在从机选择 CS 引脚输出高电平，结束本次通信。

主机接收

设置 SPI_CR1.MSTR 为 1，SPI 工作于主机模式；设置 SPI_CR3.HDOE 为 0，主机仅接收数据。

主机设置 SPI_SSI.SSI 为 0，在从机选择 CS 引脚输出低电平，作为通信起始信号。

当发送缓冲器为空时，即 SPI_ISR.TXE 标志位为 1，向 SPI_DR 寄存器写入一帧虚拟数据以启动传输。

当接收缓冲器非空时，即 SPI_ISR.RXNE 标志位变为 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

如果要接收多帧数据，重复以上步骤写入 SPI_DR 并从 SPI_DR 读取接收到的数据。

当接收完所有数据帧后，设置 SPI_SSI.SSI 为 1，在从机选择 CS 引脚输出高电平，结束本次通信。

从机发送

设置 SPI_CR1.MSTR 为 0，SPI 工作于从机模式；设置 SPI_CR3.HDOE 为 1，从机仅发送数据。

在从机选择 CS 信号被拉低之前，从机需要设置 SPI_ICR.FLUSH 为 0，以清空发送缓冲区和移位寄存器，并将待发送的第一帧数据写入 SPI_DR 寄存器。

当 CS 信号被拉低后，被写入的数据将在主机的同步移位时钟信号的控制下，从 MISO 引脚输出。

如果是多数据帧连续通信，用户应当不断查询 SPI_ISR.TXE 标志位，一旦标志为 1，立即将待发送的数据写入 SPI_DR 寄存器，以免出现数据漏发。

当检测到 CS 引脚变为高电平时，本次通信结束。

从机接收

设置 SPI_CR1.MSTR 为 0，SPI 工作于从机模式；设置 SPI_CR3.HDOE 为 0，从机仅接收数据。

当检测到 CS 引脚变为低电平时，从机开始与主机通信。

当接收缓冲器非空时，即 SPI_ISR.RXNE 标志位变为 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

当检测到 CS 引脚变为高电平时，本次通信结束。

17.3.6 单工模式

SPI 支持单工通信模式，主机和从机通过一根单向数据线进行单发或单收通信。

主机使用 MOSI 信号线进行单发通信，使用 MISO 信号线进行单收通信；从机使用 MOSI 信号线进行单收通信，使用 MISO 信号线进行单发通信，未使用的信号线可供其它功能使用。

设置控制寄存器 SPI_CR1 的 MODE 位域为 0x1，SPI 工作于单工单发通信模式；设置 MODE 位域为 0x2，SPI 工作于单工单收通信模式。

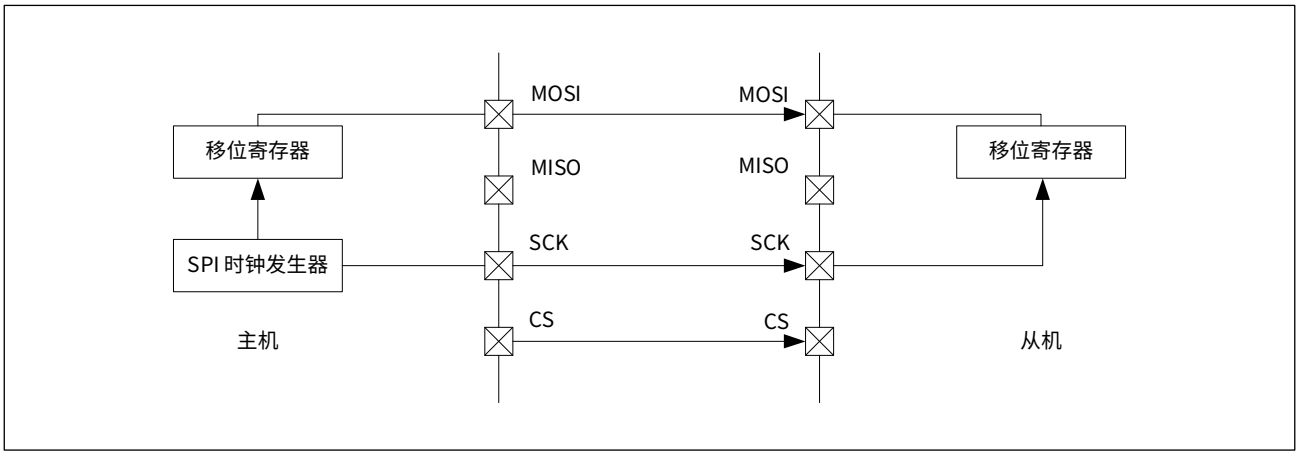
注意：

在单工单发模式下，必须忽略所有与发送端接收流相关的事件，例如接收缓冲上溢错误标志位 SPI_ISR.OV。

17.3.6.1 主机单发 / 从机单收

在主机单发、从机单收的应用场景下，主机和从机使用一根 MOSI 数据线进行通信，其应用框图如下图所示：

图 17-8 主机单发、从机单收模式



主机单发

设置 SPI_CR1.MODE 为 0x1，SPI 工作于单工单发通信模式；设置 SPI_CR1.MSTR 为 1，SPI 工作于主机模式。

设置 SPI_SSI.SSI 为 0，在从机选择 CS 引脚输出低电平，作为通信起始信号。

当发送缓冲器为空时，即 SPI_ISR.TXE 标志位为 1，将待发送的一帧数据写入 SPI_DR 寄存器，数据在同步移位时钟信号的控制下从 MOSI 引脚输出。

当写入最后一帧数据后，必须等待发送缓冲空标志位 SPI_ISR.TXE 变为 1，同时 SPI 总线忙标志位 SPI_ISR.BUSY 变为 0，以确保数据发送完毕。然后设置 SPI_SSI.SSI 为 1，使从机选择 CS 引脚输出高电平，结束本次通信。

从机单收

设置 SPI_CR1.MODE 为 0x2，SPI 工作于单工单收通信模式；设置 SPI_CR1.MSTR 为 0，SPI 工作于从机模式。

当检测到 CS 引脚变为低电平时，从机开始与主机通信。

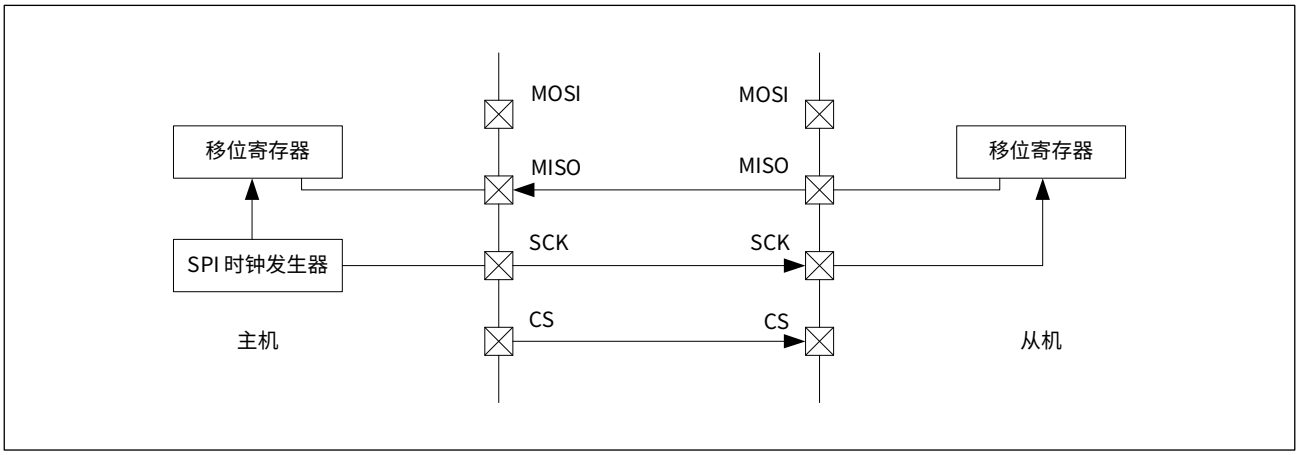
当接收缓冲器非空时，即 SPI_ISR.RXNE 标志位为 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

当检测到 CS 引脚变为高电平时，本次通信结束。

17.3.6.2 主机单收 / 从机单发

在主机单收、从机单发的应用场景下，主机和从机使用一根 MISO 数据线进行通信，其应用框图如下图所示：

图 17-9 主机单收、从机单发模式



主机单收

设置 SPI_CR1.MODE 为 0x2，SPI 工作于单工单收通信模式；设置 SPI_CR1.MSTR 为 1，SPI 工作于主机模式。

设置 SPI_SSI.SSI 为 0，在从机选择 CS 引脚输出低电平，作为通信起始信号。

当发送缓冲器为空时，即 SPI_ISR.TXE 标志位为 1，向 SPI_DR 寄存器写入一帧虚拟数据以启动传输。

当接收缓冲器非空时，即 SPI_ISR.RXNE 标志位为 1，表示已经接收完成一帧数据，此时可以读取 SPI_DR 寄存器。

如果要接收多帧数据，重复以上步骤写入 SPI_DR 并从 SPI_DR 读取接收到的数据。

当接收完所有数据帧后，设置 SPI_SSI.SSI 为 1，从机选择 CS 引脚输出高电平，结束本次通信。

从机单发

设置 SPI_CR1.MODE 为 0x1，SPI 工作于单工单发通信模式；设置 SPI_CR1.MSTR 为 0，SPI 工作于从机模式。

在从机选择信号 CS 被拉低之前，从机需要设置 SPI_ICR.FLUSH 为 0，以清空发送缓冲区和移位寄存器，并将待发送的第一帧数据写入 SPI_DR 寄存器。

当 CS 信号被拉低后，被写入的数据在主机的同步移位时钟信号的控制下，从 MISO 引脚输出。

如果是多数据帧连续通信，用户应当不断查询 SPI_ISR.TXE 标志位，一旦标志为 1，立即将待发送的数据写入 SPI_DR 寄存器，以免出现数据漏发。

当检测到 CS 引脚变为高电平时，本次通信结束。

17.3.7 多机通信

SPI 支持多机通信模式。在该模式下，主机的从机选择 CS 引脚应配置为输入，与其他主机的总线申请信号相连，用于检测 SPI 总线是否发生冲突。

如果某一主机的从机选择 CS 引脚被拉为低电平，说明有其它主机在占用总线，该主机会产生模式错误（请参见 17.3.9 错误标志），此主机 SPI 将处于禁用状态，以避免 SPI 总线冲突。

如果某一主机的从机选择 CS 引脚检测到高电平，说明 SPI 总线空闲，此时，若该主机想要与从机进行通信，必须先在总线申请信号输出低电平到其它主机的 CS 信号，获得 SPI 总线控制权，然后选择某一从机开始通信。

下图所示是一个典型的多机通信系统，以主机 A 和从机 1 进行通信为例，说明其操作流程如下：

步骤 1：主机 A 检测 CS 引脚，直到出现高电平；

步骤 2：主机 A 的 GPIO0 输出低电平，获取总线控制权，主机 B 被禁用；

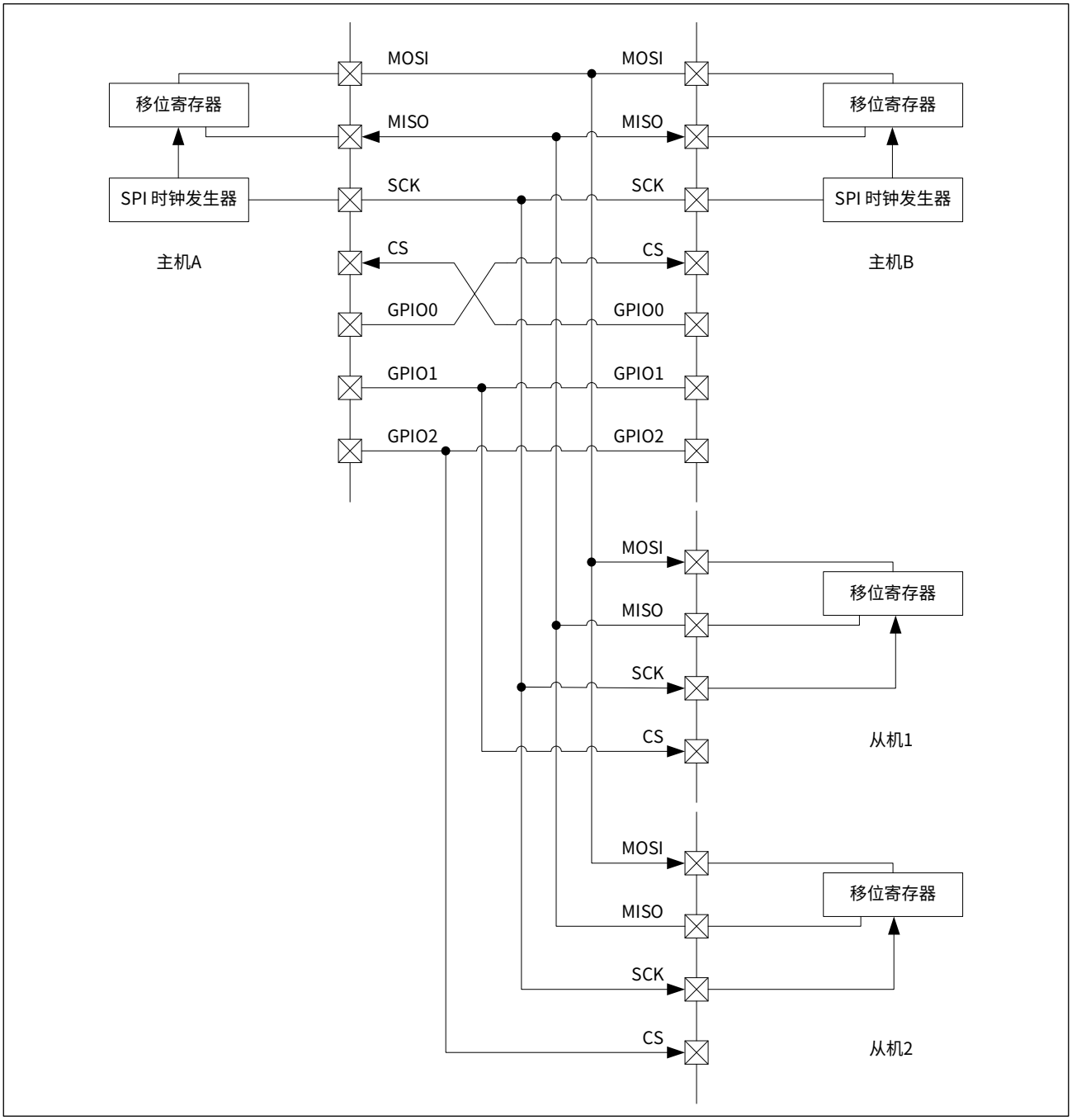
步骤 3：主机 A 的 GPIO1 输出低电平，选中从机 1；

步骤 4：主机 A 与从机 1 进行通信；

步骤 5：主机 A 的 GPIO1 输出高电平，释放从机 1；

步骤 6：主机 A 的 GPIO0 输出高电平，释放 SPI 总线。

图 17-10　多机通信系统



为保证多机通信模式下通信可靠性，从机的控制寄存器 SPI_CR1 的 MISOHD 位域应设置为 1，当从机被选中时，MISO 引脚为 CMOS 输出；未被选中时，MISO 引脚为高阻输出。

17.3.8 状态标志

SPI 控制器存在 6 个状态标志，用来指示 SPI 的一般工作状态。

发送缓冲空标志位（SPI_ISR.TXE）

当发送数据从 SPI_DR 寄存器转移到移位寄存器后，SPI_ISR.TXE 标志位会被硬件置位，表示发送缓冲器已空，此时允许对 SPI_DR 寄存器写入新的待发送数据。在对 SPI_DR 寄存器写入数据的同时，SPI_ISR.TXE 标志位会被自动清零。

接收缓冲非空标志位（SPI_ISR.RXNE）

当接收数据从移位寄存器转移到 SPI_DR 寄存器后，SPI_ISR.RXNE 标志位会被硬件置位，表示已经完成一帧数据的接收，此时可以读取 SPI_DR 寄存器，读 SPI_DR 寄存器的同时将清除 SPI_ISR.RXNE 标志位。

用户也可以通过软件手动清除 SPI_ISR.RXNE 标志位。

总线忙标志位（SPI_ISR.BUSY）

SPI_ISR.BUSY 标志位，由硬件设置和清除，用于表明当前的 SPI 通信状态。

主机模式下，当正在准备进行或正在进行数据传输时（发送缓冲区有数据或移位寄存器有数据），BUSY 标志位会被硬件置位，表示 SPI 接口处于忙碌状态；当没有数据传输时（发送缓冲区无数据且移位寄存器无数据），BUSY 标志位被硬件自动清零，表示 SPI 接口处于空闲状态。用户代码需查询到该标志为 0 时，才可拉高 CS 信号以结束本轮通信。

从机模式下，不建议使用该标志。

从机选择信号状态（SPI_ISR.SSLVL）

SPI_ISR.SSLVL 状态位，由硬件设置和清除，用于指示从机选择输入信号的电平状态。

SPI_ISR.SSLVL 为 0，从机选择 CS 输入是低电平，从机被选中；SPI_ISR.SSLVL 为 1，从机选择 CS 输入是高电平，从机未被选中。

从机选择输入上升沿标志（SPI_ISR.SSR）

SPI_ISR.SSR 标志位，用于指示从机选择 CS 输入信号是否出现了上升沿。仅在从机模式下 CS 使用硬件引脚时（MSTR=0 且 SSM=0）才有意义。

SPI_ISR.SSR 为 0，表示从机选择 CS 输入没有出现上升沿；SPI_ISR.SSR 为 1，表示从机选择输入出现了上升沿。

从机选择输入下降沿标志（SPI_ISR.SSF）

SPI_ISR.SSF 标志位，用于指示从机选择 CS 输入信号是否出现了下降沿。仅在从机模式下 CS 使用硬件引脚时（MSTR=0 且 SSM=0）才有意义。

SPI_ISR.SSF 为 0，表示从机选择 CS 输入没有出现下降沿；SPI_ISR.SSF 为 1，表示从机选择 CS 输入出现了下降沿。

17.3.9 错误标志

SPI 控制器存在 4 个错误标志，用来指示 SPI 是否发生了错误。

模式错误标志（SPI_ISR.MODF）

SPI_ISR.MODF 标志位，用于检测 SPI 多机通信模式下的总线冲突。

当 SPI 工作于主机模式，且从机选择 CS 引脚被配置为输入时，如果 CS 引脚输入为低电平，则会发生模式错误，SPI_ISR.MODF 标志位被硬件置位，表示此时有其他 SPI 主机在占用总线。

发生模式错误后，SPI_CR2.EN 被清零，SPI 被强制关闭。

设置 SPI_ICR.MODF 为 0，可清除 SPI_ISR.MODF 标志位，完成清零后，用户需要重新配置 SPI 控制器。

发生模式错误后，如果重新使能了 SPI 控制器，SPI 仍可以进行正常的数据传输，但 SPI_ISR.MODF 标志位会保持置位状态，知道用户软件清除。

从机模式下的从机选择错误标志位（SPI_ISR.SSERR）

当 SPI 工作于从机模式，且正在进行有效的数据传输时，如果从机选择 CS 输入被拉高，则会发生从机选择错误，SPI_ISR.SSERR 标志位被硬件置位。

发生从机选择错误会导致当前的数据传输出错，从机进入未选中状态。

发生从机选择错误之后，如果从机选择 CS 输入再次被拉低，SPI 仍可以进行正常的数据传输，但 SPI_ISR.SSERR 标志位会保持置位状态，直到用户软件清除。

接收缓冲上溢错误标志位（SPI_ISR.OV）

当主机或从机在收到一帧数据后，没有清除 SPI_ISR.RXNE 标志位，即，既没有读 SPI_DR 寄存器又没有软件清除 SPI_ICR.RXNE 为 0，然后收到新的一帧数据时，就会发生接收缓冲上溢错误，SPI_ISR.OV 标志位被硬件置位。

发生接收缓冲上溢错误后，新接收到的数据将覆盖缓冲器中原有的数据。

发生接收缓冲上溢错误后，SPI 仍可以进行正常的数据传输，但 SPI_ISR.OV 标志位会保持置位状态，直到用户软件清除。

从机模式下的发送缓冲下溢错误标志位（SPI_ISR.UD）

当 SPI 工作于从机模式时，如果发送缓冲器空，且新一帧的传输已经开始，就会发生发送缓冲下溢错误，SPI_ISR.UD 标志位会被硬件置位，表示从机没有及时向 SPI_DR 寄存器写入数据，错过了给主机发送数据。

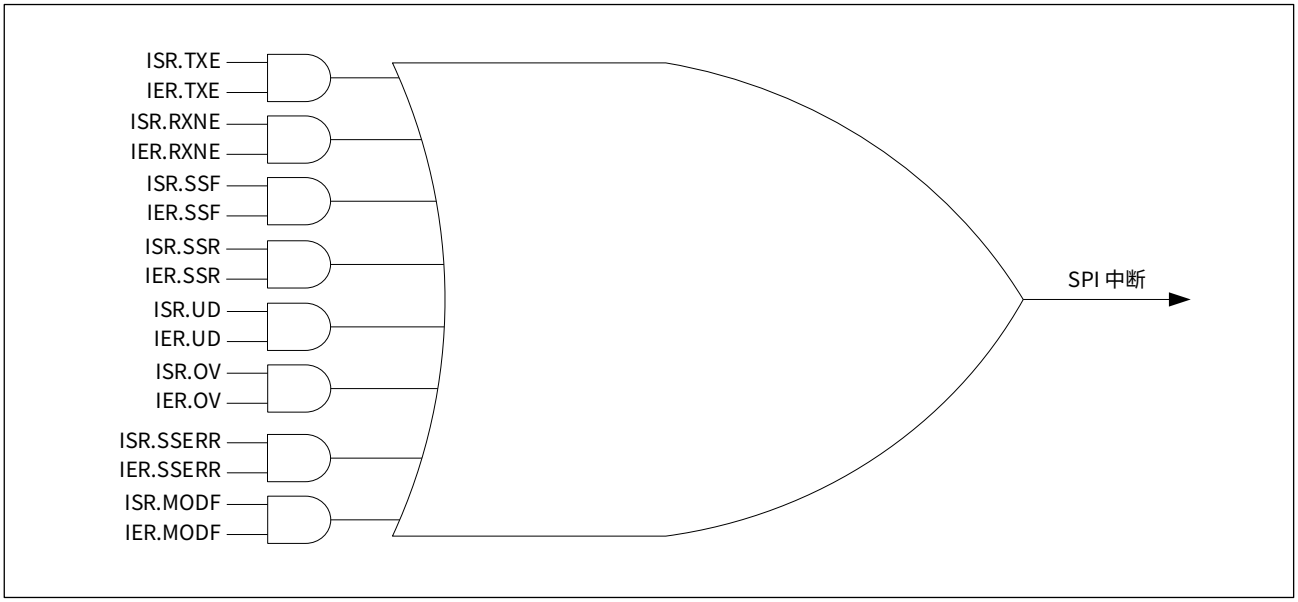
发生发送缓冲下溢错误之后，SPI 仍可以进行正常的数据传输，但 SPI_ISR.UD 标志位会保持置位状态，直到用户软件清除。

17.4 SPI 中断

SPI 控制器支持 8 个中断源，当 SPI 中断触发事件发生时，中断标志位会被硬件置位，如果设置了对应的中断使能控制位，将产生中断请求。

CW32L011 的一个 SPI 模块使用一个相同的系统 SPI 中断，SPI 中断是否产生中断跳转由中断使能设置寄存器 NVIC_ISER 的相应位控制。系统 SPI 中断示意图如下图所示：

图 17-11 SPI 中断



在用户 SPI 中断服务程序中，应查询相关 SPI 中断标志位，以进行相应的处理，在退出中断服务程序之前，要清除该中断标志位，避免重复进入中断程序。

SPI 的各中断源的标志位、中断使能位、中断标志清除位及清除方法，如下表所示：

表 17-3 SPI 中断控制

中断事件	中断标志位	中断使能位	标志清除方法
发送缓冲器空	ISR.TXE	IER.TXE	写 SPI_DR 寄存器
接收缓冲器非空	ISR.RXNE	IER.RXNE	读 SPI_DR 寄存器，或写 0 到 ICR.RXNE
从机选择输入出现下降沿	ISR.SSF	IER.SSF	写 0 到 ICR.SSF
从机选择输入出现上升沿	ISR.SSR	IER.SSR	写 0 到 ICR.SSR
从机模式下发送缓冲下溢	ISR.UD	IER.UD	写 0 到 ICR.UD
接收缓冲上溢	ISR.OV	IER.OV	写 0 到 ICR.OV
从机模式下从机选择错误	ISR.SSERR	IER.SSERR	写 0 到 ICR.SSERR
模式错误	ISR.MODF	IER.MODF	写 0 到 ICR.MODF

17.5 编程示例

17.5.1 全双工 / 主模式

17.5.1.1 查询方式收发

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_MOSI 和 SPI_CS 引脚配置为推挽复用输出模式，SPI_MISO 引脚配置为浮空输入、复用模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 13：查询等待 SPI_ISR.TXE 标志位置 1，确认发送缓冲器为空；

步骤 14：将待发送的一帧数据写入 SPI_DR 寄存器；

步骤 15：查询等待 SPI_ISR.RXNE 标志位置 1，确认接收完成一帧数据；

步骤 16：读取 SPI_DR 寄存器并保存数据；

步骤 17：重复步骤 13 至步骤 17，进行下一帧数据的收发，直到全部数据收发完毕；

步骤 18：设置 SPI_SSI.SSI 为 1，通信结束。

17.5.1.2 中断方式收发

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_MOSI 和 SPI_CS 引脚配置为推挽复用输出模式，SPI_MISO 引脚配置为浮空输入、复用模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：向 SPI_ICR 写入 0x00，清除所有标志位；

步骤 13：配置 NVIC 控制器，请参见 5 中断章节；

步骤 14：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 15：设置 SPI_IER.RXNE 为 1 使能接收缓冲非空中断；

步骤 16：设置 SPI_IER.TXE 为 1 使能发送缓冲空中断；

步骤 17：进入中断服务函数：查询判断 SPI_ISR.TXE 标志位，如果标志位为 1，则将待发送的一帧数据写入 SPI_DR 寄存器；查询判断 SPI_ISR.RXNE 标志位，如果标志位为 1，则读取 SPI_DR 寄存器并保存数据；

步骤 18：设置 SPI_SSI.SSI 为 1，通信结束。

17.5.2 单线半双工 / 主模式

17.5.2.1 查询方式发送

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_MOSI 和 SPI_CS 引脚配置为推挽复用输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 0x3，配置 SPI 为单线半双工通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 13：设置 SPI_CR3. HDOE 为 1，仅发送；

步骤 14：查询等待 SPI_ISR.TXE 标志位置 1，确认发送缓冲器为空；

步骤 15：将待发送的一帧数据写入 SPI_DR 寄存器；

步骤 16：如果还有数据待发送，重复步骤 14 至步骤 16，发送下一帧数据；

步骤 17：查询等待 SPI_ISR.TXE 标志位置 1，最后一帧数据开始发送；

步骤 18：查询等待 SPI_ISR.BUSY 标志位变为 0，SPI 总线空闲；

步骤 19：设置 SPI_SSI.SSI 为 1，通信结束。

17.5.2.2 查询方式接收

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_MOSI 和 SPI_CS 引脚配置为推挽复用输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 11，配置 SPI 为单线半双工通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 13：设置 SPI_CR3.HDOE 为 0，仅接收；

步骤 14：查询等待 SPI_ISR.TXE 标志位置 1，确认发送缓冲器为空；

步骤 15：将一帧虚拟数据写入 SPI_DR 寄存器；

步骤 16：查询等待 SPI_ISR.RXNE 标志位置 1，确认接收完成一帧数据；

步骤 17：读取 SPI_DR 寄存器并保存数据；

步骤 18：重复步骤 14 至步骤 18，接收下一帧数据，直到全部数据接收完毕；

步骤 19：设置 SPI_SSI.SSI 为 1，通信结束。

17.5.3 单工模式 / 主模式

17.5.3.1 查询方式发送

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_MOSI 和 SPI_CS 引脚配置为推挽复用输出模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 01，配置 SPI 为单工单发通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 13：查询等待 SPI_ISR.TXE 标志位置 1，确认发送缓冲器为空；

步骤 14：将待发送的一帧数据写入 SPI_DR 寄存器；

步骤 15：如果还有数据待发送，重复步骤 13 至步骤 15，发送下一帧数据；

步骤 16：查询等待 SPI_ISR.TXE 标志位置 1，最后一帧数据开始发送；

步骤 17：查询等待 SPI_ISR.BUSY 标志位变为 0，SPI 总线空闲；

步骤 18：设置 SPI_SSI.SSI 为 1，通信结束。

17.5.3.2 查询方式接收

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.SPI 为 1，使能 SPI 引脚对应的 GPIO 时钟和 SPI 工作时钟；

步骤 2：将 SPI_SCK、SPI_CS 引脚配置为推挽复用输出模式，SPI_MISO 引脚配置为浮空输入、复用模式，具体寄存器配置步骤请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 SPI_CR1.MODE 为 10，配置 SPI 为单工单收通信模式；

步骤 4：设置 SPI_CR1.MSTR 为 1，配置 SPI 为主机模式；

步骤 5：配置 SPI_CR1.WIDTH，设置每帧数据的宽度；

步骤 6：配置 SPI_CR1.LSBF，设置数据位收发顺序；

步骤 7：配置 SPI_CR1.CPOL，设置时钟极性；

步骤 8：配置 SPI_CR1.CPHA，设置时钟相位；

步骤 9：设置 SPI_CR1.SSM 为 1，通过 SSI 寄存器决定从机选择输出值；

步骤 10：配置 SPI_CR1.BR，设置 SCK 波特率；

步骤 11：设置 SPI_CR2.EN 为 1，使能 SPI 控制器；

步骤 12：设置 SPI_SSI.SSI 为 0，通信开始；

步骤 13：查询等待 SPI_ISR.TXE 标志位置 1，确认发送缓冲器为空；

步骤 14：将一帧虚拟数据写入 SPI_DR 寄存器；

步骤 15：查询等待 SPI_ISR.RXNE 标志位置 1，确认接收完成一帧数据；

步骤 16：读取 SPI_DR 寄存器并保存数据；

步骤 17：重复步骤 13 至步骤 17，接收下一帧数据，直到全部数据接收完毕；

步骤 18：设置 SPI_SSI.SSI 为 1，通信结束。

17.6 寄存器列表

SPI 基地址：SPI_BASE = 0x4000 0800

表 17-4 SPI 寄存器列表

寄存器名称	寄存器地址	寄存器描述
SPI_CR1	SPI_BASE + 0x00	控制寄存器 1
SPI_CR2	SPI_BASE + 0x04	控制寄存器 2
SPI_CR3	SPI_BASE + 0x08	控制寄存器 3
SPI_SSI	SPI_BASE + 0x0C	从机选择寄存器
SPI_IER	SPI_BASE + 0x10	中断使能寄存器
SPI_ISR	SPI_BASE + 0x14	中断标志寄存器
SPI_ICR	SPI_BASE + 0x18	中断标志清除寄存器
SPI_DR	SPI_BASE + 0x1C	数据寄存器

17.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

17.7.1 SPI_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0700

位域	名称	权限	功能描述
31:21	RFU	-	保留位，请保持默认值
20	SMP	RW	主机模式延后采样 0：正常采样方式，采样时刻由 CPOL 和 CPHA 决定 1：延后采样方式，采样时刻相比正常采样方式延后约 20ns
19	MISOHD	RW	从机 MISO 输出配置 0：MISO 始终为 COMS 输出 1：从机被选中时 MISO 为 COMS 输出，未被选中时为高阻输出
18	FLTEN	RW	滤波功能使能 0：禁止 1：使能 *注：数字滤波电路可以滤除 SCK、MISO、MOSI 信号上小于一个 PCLK 周期的小脉冲，但会增加约 2 个 PCLK 时钟的额外延时。数字滤波功能仅在 SPI 通信速率明显低于 PCLK 时钟频率时使用（$f_{SCK} < f_{PCLK}/8$）。*
17:16	MODE	RW	通信模式配置 00：双向全双工 01：单工单发 10：单工单收 11：单线半双工
15:12	GAP	RW	主机模式帧间隔 0000：无间隔 0001：$1T_{SCK}$ 0010：$2T_{SCK}$ …… 1111：$15T_{SCK}$
11:8	WIDTH	RW	每帧数据的宽度为 WIDTH+1， 例如 0011：4bit 0100：5bit … 1110：15bit 1111：16bit *注：写 0~2 自动变为 8bit。*
7	LSBF	RW	数据帧高低位顺序选择 0：最高有效位（MSB）收发在前 1：最低有效位（LSB）收发在前

位域	名称	权限	功能描述
6:4	BR	RW	主机模式 SCK 波特率配置 000：PCLK/2 001：PCLK/4 010：PCLK/8 011：PCLK/16 100：PCLK/32 101：PCLK/64 110：PCLK/128 111：PCLK/256
3	CPOL	RW	串行时钟极性配置 0：待机时低电平 1：待机时高电平
2	CPHA	RW	串行时钟相位配置 0：前边沿采样 / 后边沿移位 1：前边沿移位 / 后边沿采样
1	SSM	RW	从机选择配置（主机模式） 0：SPI_CS 引脚为输入模式，引脚低电平可选中本机 1：SPI_CS 引脚为输出模式，输出电平来自 SSI 从机选择配置（从机模式） 0：SPI_CS 引脚电平决定本机是否被选中 1：SSI 寄存器的值决定本机是否被选中
0	MSTR	RW	工作模式配置 0：从机模式 1：主机模式

注意：

只有当 SPI_CR2.EN 为 0 时，才可以修改本寄存器。

17.7.2 SPI_CR2 控制寄存器 2

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:5	RFU	-	保留位，请保持默认值
4	ADCTX	RW	发送缓冲器空触发 ADC 使能控制 0：禁止 1：使能
3	ADCRX	RW	接收数据完成触发 ADC 使能控制 0：禁止 1：使能
2:1	RFU	-	保留位，请保持默认值
0	EN	RW	使能控制 0：禁止 SPI 模块 1：使能 SPI 模块

17.7.3 SPI_CR3 控制寄存器 3

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	HDOE	RW	单线半双工时，数据发送 / 接收状态控制 0：仅接收 1：仅发送 *注：仅在单线半双工通信时有效。*

17.7.4 SPI_SSI 从机选择寄存器

Address offset: 0x0C Reset value: 0x0000 0001

<table>
<tr><th>位域</th><th>名称</th><th>权限</th><th>功能描述</th></tr>
<tr><td>31:1</td><td>RFU</td><td>-</td><td>保留位，请保持默认值</td></tr>
<tr><td>0</td><td>SSI</td><td>RW</td><td>从机选择，当 SPI_CR1.SSM 为 1 时有效
<table>
<tr><th>SSM</th><th>MSTR</th><th>SSI</th><th>描述</th></tr>
<tr><td>0</td><td>-</td><td>-</td><td>无效</td></tr>
<tr><td rowspan="4">1</td><td rowspan="2">0(从机)</td><td>0</td><td>设置本机为选中态</td></tr>
<tr><td>1</td><td>设置本机为未选中态</td></tr>
<tr><td rowspan="2">1(主机)</td><td>0</td><td>SPI_CS 引脚输出低</td></tr>
<tr><td>1</td><td>SPI_CS 引脚输出高</td></tr>
</table></td></tr>
</table>

17.7.5 SPI_IER 中断使能寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	MODF	RW	模式错误中断使能控制 0：禁止 1：使能
6	SSERR	RW	从机模式下的从机选择错误中断使能控制 0：禁止 1：使能
5	OV	RW	接收缓冲上溢错误中断使能控制 0：禁止 1：使能
4	UD	RW	从机模式下发送缓冲下溢错误中断使能控制 0：禁止 1：使能
3	SSR	RW	从机选择输入上升沿中断使能控制 0：禁用 1：使能
2	SSF	RW	从机选择输入下降沿中断使能控制 0：禁用 1：使能
1	RXNE	RW	接收缓冲非空中断使能控制 0：禁用 1：使能
0	TXE	RW	发送缓冲空中断使能控制 0：禁用 1：使能

17.7.6 SPI_ISR 中断标志寄存器

Address offset: 0x14 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:10	RFU	-	保留位，请保持默认值
9	SSLVL	RO	从机选择信号状态 0：从机选择信号为低电平 1：从机选择信号为高电平
8	BUSY	RO	总线忙标志 0：总线空闲 1：总线忙
7	MODF	RO	模式错误标志 0：正常 1：出错 *注：主机模式下从机选择输入为低则触发 MODF。*
6	SSERR	RO	从机模式下的从机选择错误标志位 0：正常 1：出错
5	OV	RO	接收缓冲上溢错误标志位 0：正常 1：出错
4	UD	RO	从机模式下的发送缓冲下溢错误标志位 0：正常 1：出错
3	SSR	RO	从机选择输入上升沿标志位 0：未出现上升沿 1：出现了上升沿 *注：仅在从机模式下 CS 使用硬件引脚时（MSTR=0 且 SSM=0）才有意义。*
2	SSF	RO	从机选择输入下降沿标志位 0：未出现下降沿 1：出现了下降沿 *注：仅在从机模式下 CS 使用硬件引脚时（MSTR=0 且 SSM=0）才有意义。*
1	RXNE	RO	接收缓冲非空标志位 0：接收缓冲空 1：接收缓冲非空 *注：对 DR 寄存器的读操作或对 ICR.RXNE 写 0 均可以清除该标志。*
0	TXE	RO	发送缓冲空标志位 0：发送缓冲非空 1：发送缓冲空

17.7.7 SPI_ICR 中断标志清除寄存器

Address offset: 0x18 Reset value: 0x0000 00FF

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	MODF	R1W0	模式错误标志清除 W0：清除模式错误标志 W1：无功能
6	SSERR	R1W0	从机模式下的从机选择错误标志清除 W0：清除从机模式下的从机选择错误标志 W1：无功能
5	OV	R1W0	接收缓冲上溢错误标志清除 W0：清除接收缓冲上溢错误标志 W1：无功能
4	UD	R1W0	从机模式下的发送缓冲下溢错误标志清除 W0：清除从机模式下的发送缓冲下溢错误标志 W1：无功能
3	SSR	R1W0	从机选择输入上升沿标志清除 W0：清除从机选择输入上升沿标志 W1：无功能
2	SSF	R1W0	从机选择输入下降沿标志清除 W0：清除从机选择输入下降沿标志 W1：无功能
1	RXNE	R1W0	接收缓冲非空标志清除 W0：清除接收缓冲非空标志 W1：无功能
0	FLUSH	R1W0	清空发送缓冲区和移位寄存器 W0：清空发送缓冲区和移位寄存器（清除后 ISR.TXE 置 1） W1：无功能

17.7.8 SPI_DR 数据寄存器

Address offset: 0x1C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	DR	RW	写入为待发送的数据 读出为接收到的数据

18 I2C 接口

18.1 概述

CW32L011 内部集成 1 个 I2C 控制器，能按照设定的传输速率（标准，快速，高速）将需要发送的数据按照 I2C 规范串行发送到 I2C 总线上，并对通信过程中的状态进行检测，另外还支持多主机通信中的总线冲突和仲裁处理。

18.2 主要特性

- 支持主机发送 / 接收，从机发送 / 接收四种工作模式
- 支持时钟延展 (时钟同步) 和多主机通信冲突仲裁
- 支持标准 (100Kbps)/ 快速 (400Kbps)/ 高速 (1Mbps) 三种工作速率
- 支持 7bit 寻址功能
- 支持 3 个从机地址
- 支持广播地址
- 支持输入信号噪声过滤功能
- 支持中断状态查询功能
- 支持 SMBUS 超时检测
- 支持与工作电压低于 MCU 的器件通信（借助 VC）

18.3 协议描述

I2C 总线使用两根信号线（数据线 SDA 和时钟线 SCL）在设备间传输数据。SCL 为单向时钟线，固定由主机驱动。SDA 为双向数据线，在数据传输过程中由收发两端分时驱动。

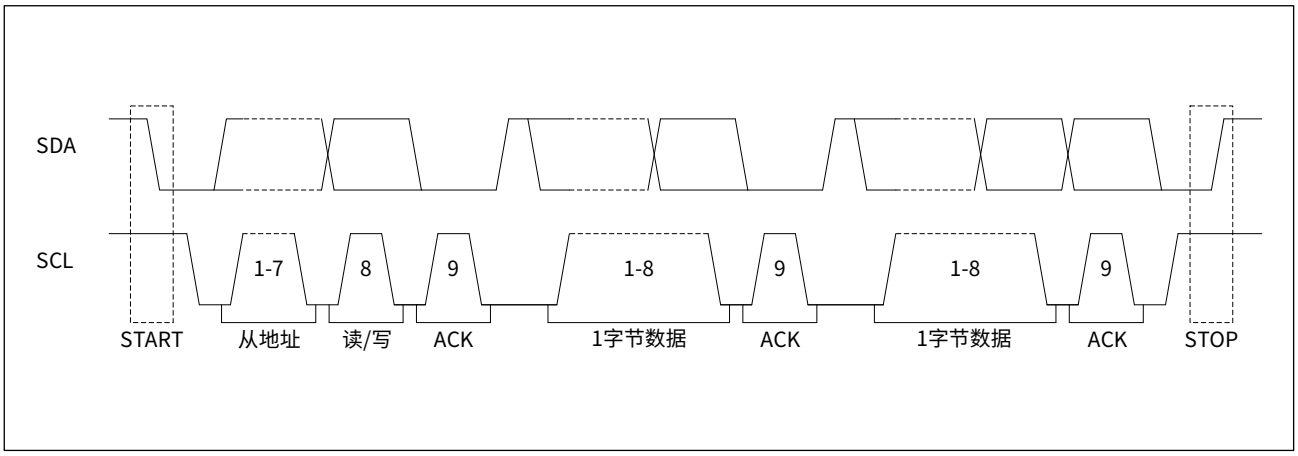
I2C 总线上可以连接多个设备，所有设备在没有进行数据传输时都处于空闲状态（未寻址从机接收模式），任一设备都可以作为主机发送 START 起始信号来开始数据传输，在 STOP 停止信号出现在总线上之前，总线一直处于被占用状态。

I2C 通信采用主从结构，并由主机发起和结束通信。主机通过发送 START 起始信号来发起通信，之后发送 SLA+W/R 共 8bit 数据（其中，SLA 为 7bit 从机地址，W/R 为读写位），并在第 9 个 SCL 时钟释放 SDA 总线，对应的从机在第 9 个 SCL 时钟占用 SDA 总线并输出 ACK 应答信号，完成从机寻址。此后根据主机发送的第 1 字节的 W/R 位来决定数据通信的发端和收端，发端每发送 1 个字节数据，收端必须回应 1 个 ACK 应答信号。数据传输完成后，主机发送 STOP 信号结束本次通信。

18.3.1 协议帧格式

标准 I2C 传输协议帧包含四个部分：起始信号（START）或重复起始信号（Repeated START），从机地址及读写位，数据传输，停止信号（STOP）。如下图所示：

图 18-1 I2C 协议帧



- 起始信号（START）

  当总线处于空闲状态时（SCL 和 SDA 线同时为高），SDA 线上出现由高到低的下降沿信号，则被定义为起始信号。主机向总线发出起始信号后开始数据传输，并占用总线。

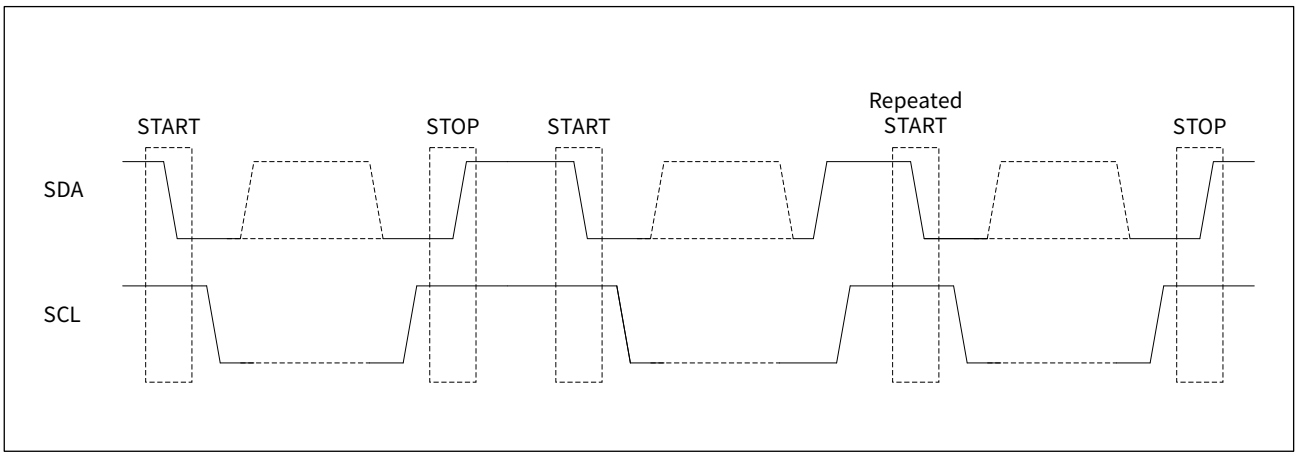
- 重复起始信号（Repeated START）

  当一个起始信号后未出现停止信号之前，出现了新的起始信号，新的起始信号被定义为重复起始信号。在主机发送停止信号前，SDA 总线一直处于占用状态，其它主机无法占用总线。

- 停止信号（STOP）

  当 SCL 线为高时，SDA 线上出现由低到高的上升沿信号，则被定义为停止信号。主机向总线发出停止信号以结束数据传输，并释放总线。

图 18-2　起始和停止信号



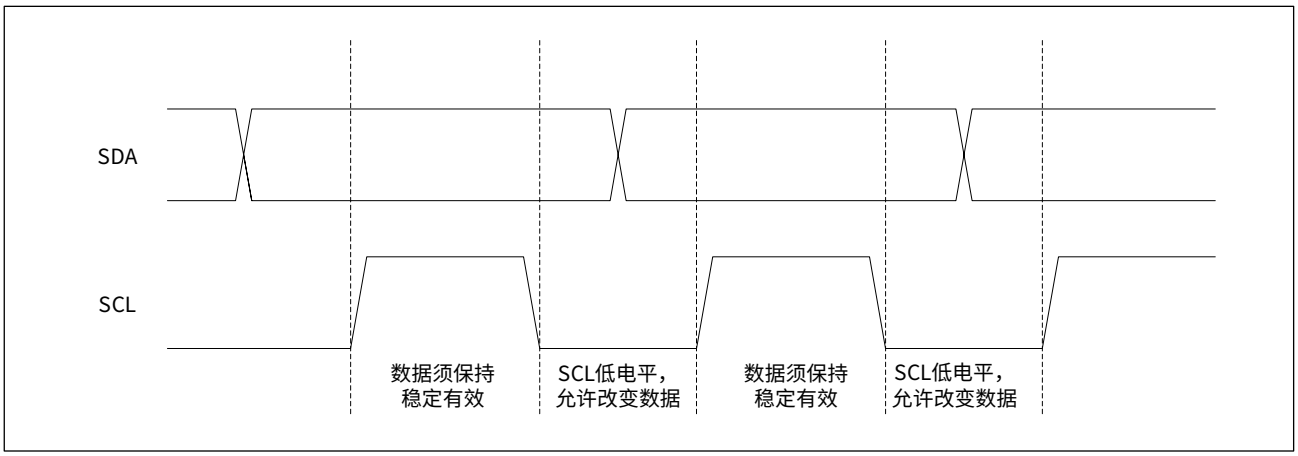
- 从机地址及读写位

  当起始信号产生后，主机开始传输第 1 字节数据：7 bit 从机地址 + 读写位。读写位（1：读；0：写）控制总线上数据传输方向。被寻址的从机在第 9 个 SCL 时钟周期内占用 SDA 总线，并将 SDA 置为低电平作为 ACK 应答。

- 数据传输

  主机在 SCL 线上输出串行时钟信号，主从机通过 SDA 线进行数据传输。数据传输过程中，1 个 SCL 时钟脉冲传输 1 个数据位（最高有效位 MSB 在前），且 SDA 线上的数据只在 SCL 为低时改变，每传输 1 个字节跟随 1 个应答位。如下图所示：

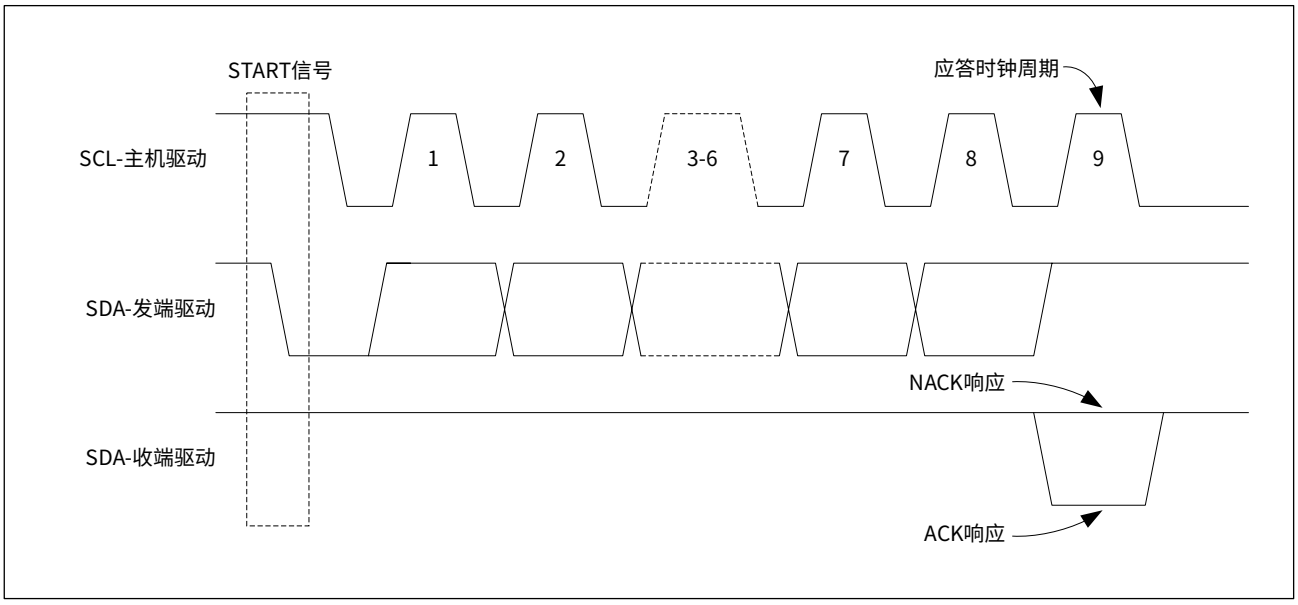
图 18-3　数据传输



18.3.2 传输应答

在总线上传输数据时，发端每传输完 1 个字节数据，在第 9 个 SCL 时钟周期发端放弃对 SDA 的控制，收端须在第 9 个 SCL 时钟周期回复 1 个应答位：接收成功，发送 ACK 应答，接收异常发送 NACK 应答。

图 18-4 传输应答



18.3.3 冲突检测与仲裁

在多主机通信系统中，总线上的每个节点都有从机地址。每个节点可以作为从节点被其它节点访问，也可以作为主节点向其它的节点发送控制字节和传送数据。如果有两个或两个以上的节点同时向总线发出起始信号并开始传输数据，就会造成总线冲突。I2C 控制器内置一个仲裁器，可对 I2C 总线冲突进行检测和仲裁，以保证数据通信的可靠性和完整性。

- 冲突检测原理

  在物理实现上，SDA 和 SCL 引脚电路结构相同，引脚的输出驱动与输入缓冲连在一起。输出结构为漏极开路的场效应管、输入结构为高输入阻抗的同相器。基于该结构：

  1. 由于 SDA、SCL 为漏极开路结构，借助于外部的上拉电阻实现了信号的“线与”逻辑；
  2. 设备向总线写数据的同时读取数据，可用来检测总线冲突，实现 “时钟同步”和“总线仲裁”。

  根据“线与”逻辑，如果 2 个主机同时发送逻辑 1 或逻辑 0，则 2 个主机都检测不到冲突，需要等到下一位数据发送再继续检测冲突；如果 2 个主机一个发送逻辑 1，一个发送逻辑 0，此时总线上为逻辑 0，发送逻辑 1 的主机检测到冲突，发送逻辑 0 的主机没有检测到冲突。

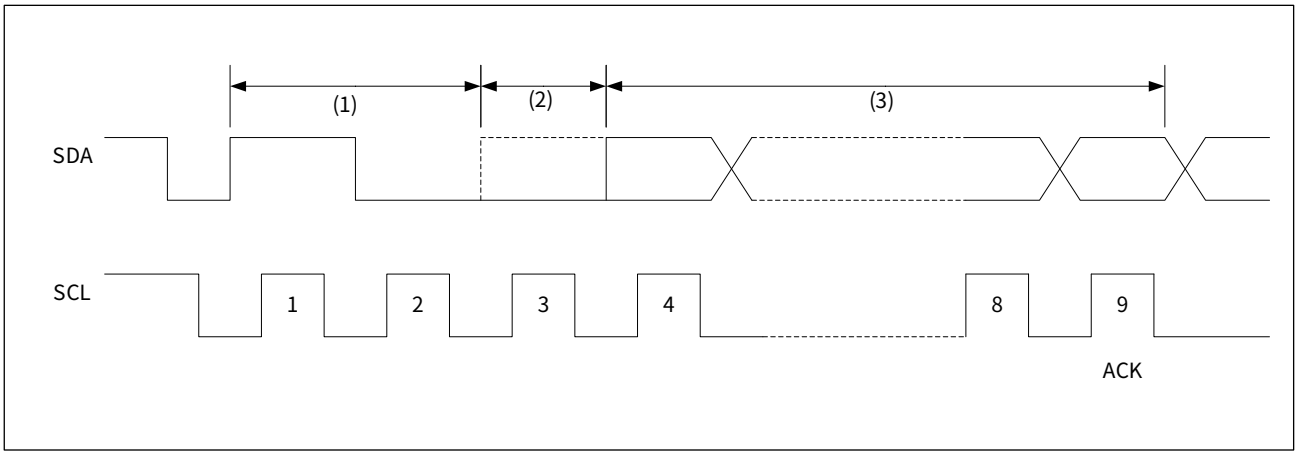
- 冲突仲裁原理

  当主机检测到总线冲突后，该主机丢失仲裁，退出主机发送模式，进入未寻址从机模式，释放 SDA 数据线，并回到地址侦测状态，之后根据接收到的 SLA+W/R 进入相应的从机模式 (SLA 地址匹配进入已寻址从机模式，SLA 地址不匹配则进入未寻址从机模式)。仲裁失败的主机，仍会发送 SCL 串行时钟，直到当前字节传输结束。

  当主机没有检测到总线冲突，该主机赢得仲裁，继续主导本次数据传输，直到通信完成。

下图为一个 I2C 总线上 A 主机和 B 主机发送冲突和仲裁示意图（假设 A，B 两个主机的发送时钟同步）：

图 18-5 冲突检测及仲裁



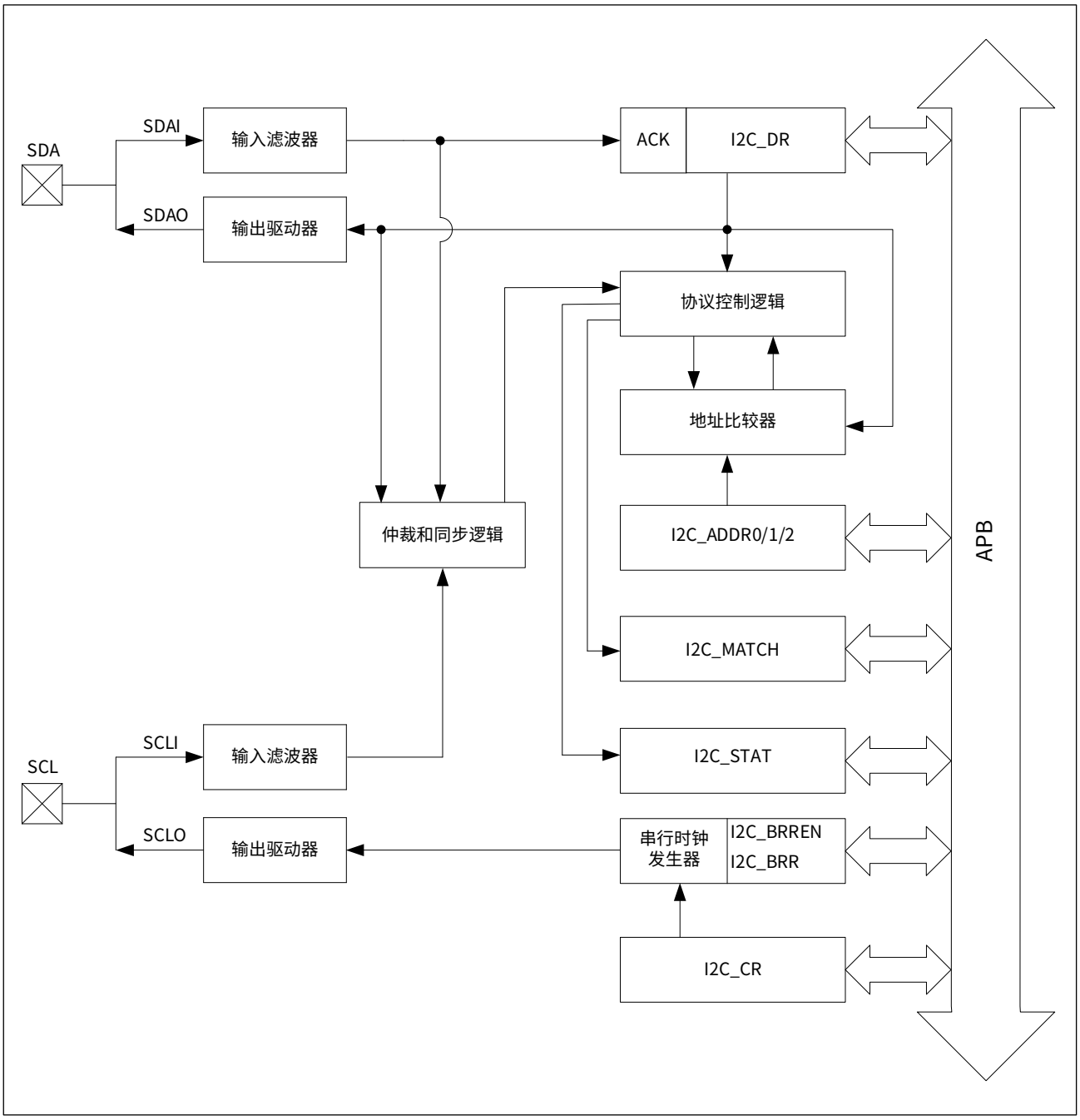
1. 在阶段（1），A 主机和 B 主机均先后发送逻辑 1 和逻辑 0，由于数据相同且同步，2 个主机都不会检测到总线冲突。
2. 在阶段（2），A 主机发送逻辑 1，B 主机发送逻辑 0，SDA 总线上数据为逻辑 0。A 主机控制器检测到数据错误，退出发送竞争，即丢失仲裁，A 主机进入未被寻址的从机接收模式。B 主机没有检测到总线冲突，赢得仲裁，继续本次数据传输。
3. 在阶段（3），A 主机处于未被寻址的从机接收模式，但仍产生 SCL 时钟脉冲，直到当前字节传输结束。此后 A 主机 I2C 控制器不再发送时钟信号，B 主机由于赢得仲裁，SCL 和 SDA 都由 B 主机来主导控制传输。

18.4 功能描述

18.4.1 功能框图

I2C 模块主要包括时钟发生器、输入滤波器、地址比较器、协议控制逻辑、仲裁和同步逻辑、以及相关寄存器等。其功能框图如下图所示：

图 18-6 I2C 功能框图



CW32L011 支持用户灵活选择 GPIO 作为 I2C 通信引脚，如下表所示：

表 18-1 I2C 引脚选择

I2C	引脚
I2C_SCL	PA04/PA09/PA14/PB06/PC15
I2C_SDA	PA05/PA10/PA13/PB07/PC14

18.4.2 串行时钟发生器

串行时钟发生器用来产生 I2C 通信的波特率时钟 SCL。串行时钟发生器采用 PCLK 作为输入时钟，通过 1 个 8bit 的计数器计数，输出所需波特率的 I2C 时钟信号。

SCL 时钟频率计算公式：

$$f_{SCL} = f_{PCLK} / 8 / (BRR + 1)$$

其中，BRR 通过波特率计数器配置寄存器 I2C_BRR 配置，BRR 有效范围为 1 ~ 255。

串行时钟发生器的计数器计数由 I2C_BRREN 寄存器的 EN 位域使能，EN 为 1 使能，为 0 禁止。主机时应设置 EN 为 1，从机时该位不影响。

PCLK、BRR 组合和 SCL 时钟频率的对应关系如下表所示：

表 18-2 I2C 传输速率和配置举例

f_{SCL}(kHz) / f_{PCLK} (kHz)	I2C_BRR						
	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	**100**	83	71	62
6000	375	250	187	150	124	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	**1000**	666	500	**400**	333	285	250

18.4.3 输入滤波器

输入滤波器可对 SDA 和 SCL 输入信号进行滤波处理，小于 1 个 PCLK 周期的尖峰脉冲信号会被滤除。

输入滤波器可配置为两种模式：简单滤波模式和高级滤波模式。简单滤波具有更快的通信速率；高级模式则具有更高的抗干扰性能。通过 I2C 控制寄存器 I2C_CR 的 FLT 位域配置滤波模式，设置 I2C_CR.FLT 为 1 则为简单滤波模式，设置 I2C_CR.FLT 为 0 则为高级滤波模式。

当作为主机时，如果 BRR 的值小于或等于 9，则应设置 I2C_CR.FLT 为 1；如果 BRR 的值大于 9，则应设置 I2C_CR.FLT 为 0。

当作为从机时，如果 PCLK 与 SCL 的频率比值小于或等于 40，则应设置 I2C_CR.FLT 为 1；如果 PCLK 与 SCL 的频率比值大于 40，则应设置 I2C_CR.FLT 为 0。

18.4.4 地址比较器

I2C 总线上各设备都有从机地址，且各从机地址均不同。主机根据从机地址寻址从机，从机通过地址比较器自动检测主机发送的 7bit 寻址地址与本机地址是否匹配，以确定是否与主机通信。

I2C 控制器支持 3 个可编程的从机地址，具体地址信息通过从机地址寄存器 I2C_ADDR0 / I2C_ADDR1 / I2C_ADDR2 进行配置。

从机的地址比较器将接收到的 7bit 寻址地址和 3 个从机地址以及广播地址（0x00）相比较。如果符合 4 个地址中的任何一个，则认为地址匹配，同时 I2C 中断标志位 I2C_CR.SI 会被置 1，并产生一个中断请求。应用程序可通过查询从机地址匹配寄存器 I2C_MATCH 获取匹配成功的地址序号。

如果地址匹配到 I2C_ADD0 / 1 / 2，则从机进入相应的已寻址从机接收模式（接收到 SLA+W）或者已寻址从机发送模式（接收到 SLA+R）；如果地址匹配到广播地址 0x00（接收到 SLA+W），则从机进入广播接收模式。

18.4.5 仲裁和同步逻辑

18.4.5.1 SCL 同步

I2C 支持时钟同步（时钟延展）功能，SCL 时钟低电平的时间由 SCL 时钟低电平宽度最长的器件决定，而 SCL 时钟高电平时间由时钟高电平宽度最短的器件决定。

如果从机希望主机降低传输速度，可以在收到数据并回应 ACK 后，保持 I2C_CR.SI 为 1，则从机 I2C 控制器将保持 SCL 为低电平状态，以此来通知主机。当主机准备下一个字节数据传输（发送或者接收）时检测到 SCL 的电平被拉低，则进行等待，直到从机完成操作并将 I2C_CR.SI 清 0，从机控制器释放 SCL 的拉低控制，主机检测到 SCL 为高电平后继续下一个字节数据的传输。

18.4.5.2 SDA 仲裁

I2C 支持 SDA 冲突检测和仲裁，可以保证在多个主机企图控制 I2C 总线时，I2C 总线上的数据不被破坏。

每个主机发送数据时，都会同时比较总线上的数据与自己发送的数据是否一致，不一致则认为检测到总线冲突，会退出发送竞争，即丢失仲裁。丢失仲裁的主机会立即切换到未被寻址的从机状态，以确保自身能被仲裁成功的主机寻址到。丢失仲裁的主机会继续输出 SCL 串行时钟，直到当前字节传输完成。

SDA 仲裁一般发生在主机发送 SLA+W/R 数据阶段，如果两个主机同时向一个从机发送数据，即两个主机发送的从机地址相同，则仲裁会在第二个字节持续。

18.4.6 应答控制

I2C 数据传输的收端必须在每个字节的第 9 个 SCL 时钟周期给发端进行 ACK 或者 NACK 应答，发端通过该应答位来获取收端当前状态：回应 ACK 应答，则表明收端已正确接收该字节数据，可以继续接收下一字节数据；回应 NACK 一般表示收端已不再接收任何数据。

收端发送 ACK 还是 NACK，由收端的 I2C 控制寄存器 I2C_CR 的 AA 位域来控制。当设置 I2C_CR.AA 为 1 时，I2C 模块每收到 1 字节数据后会回应 ACK 应答，当设置 I2C_CR.AA 为 0 时，I2C 模块每收到 1 字节数据后回应 NACK 应答。

在主机接收数据过程中，主机作为通信发起方，控制着收发字节个数，主机（收端）在最后一个字节数据接收完成后回应 NACK 应答给从机（发端），从机收到 NACK 应答后将切换为未寻址从机接收模式，并释放 SDA 总线，以便主机发送 STOP 停止信号或 Repeated START 重复起始信号。

在从机发送数据过程中，如果自身的 I2C_CR.AA 应答控制位被应用程序清零，则从机在发送完最后 1 字节有效数据后，将自身切换为未寻址从机接收模式，并释放 SDA 总线，主机从总线上读数据将得到 0xFF。此时主机应能判断从机处于无响应状态，并发送 STOP 停止信号或 Repeated START 重复起始信号。

在从机接收数据过程中，如果回应 NACK 给主机（发端），则表示从机（收端）主动结束本次通信，不再接收主机发送的数据，且将自身切换为未寻址从机接收模式，并释放 SDA 总线，此时主机应发送 STOP 停止信号或 Repeated START 重复起始信号。

18.4.7 输入信号来源 / 电平转换

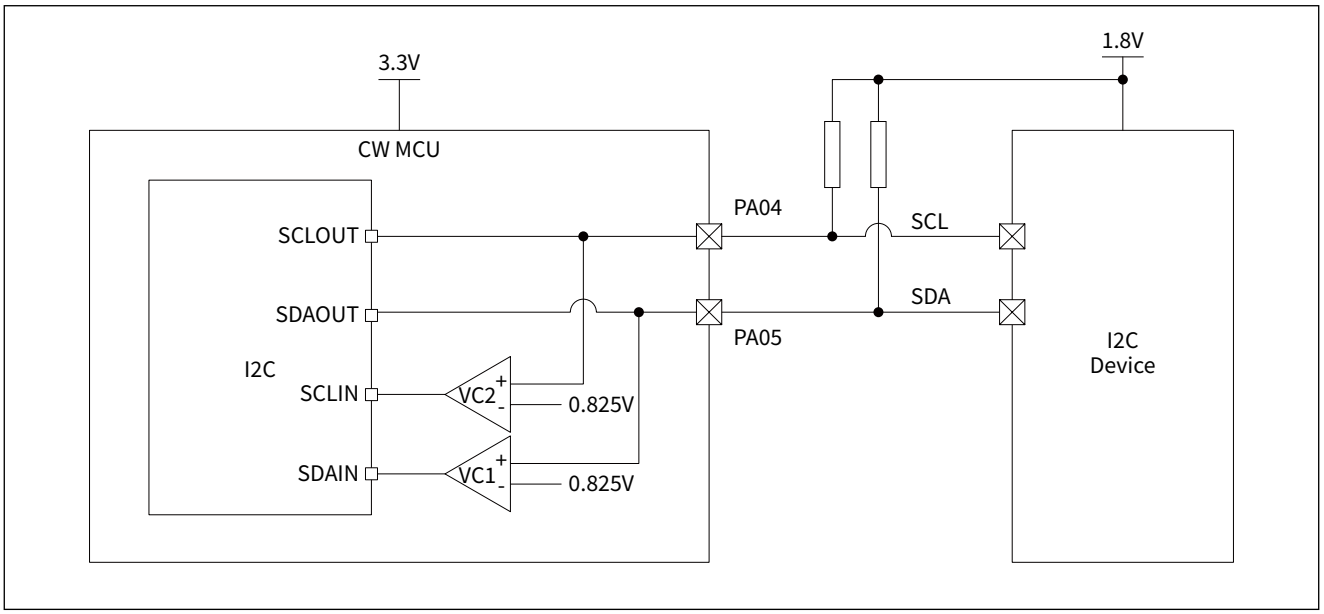
本模块的 SCL/SDA 输入信号可配置为多种来源，如下表所示：

表 18-3 SCL/SDA 输入信号来源

I2C_CR.SCLINSRC	SCL 输入信号来源	I2C_CR.SDAINSRC	SDA 输入信号来源
000	I2C_SCL 引脚	000	I2C_SDA 引脚
001	VC1_OUT	001	VC1_OUT
010	VC2_OUT	010	VC2_OUT

当配置 SCL/SDA 的输入信号来源为 VCx_OUT 时，本模块可与低于本芯片工作电压的 I2C 器件进行通信，VCx 在通信链路中充当了电平转换的角色，典型工作电路如下图所示：

图 18-7 不同电压器件通信示意图



注意，上图所示的 PA04 引脚具有 I2C_SCL 功能及 VC1_CH2 通道，PA05 引脚具有 I2C_SDA 功能及 VC2_CH2 通道。使用时，需配置 PA04 的数字功能为 I2C_SCL 并使能开漏，需配置 PA05 的数字功能为 I2C_SDA 并使能开漏；需配置 I2C_SCL 的输入信号来源为 VC1_OUT，需配置 I2C_SDA 的输入信号来源为 VC2_OUT；需配置 VC1 的正端为 CH2（PA04 引脚），需配置 VC2 的正端为 CH2（PA05 引脚）；需配置 VC1/VC2 的负端均为电阻分压输出的 0.825V。

18.4.8 SMBUS 超时时间检测

I2C 与 BTIM1 通过内部互联信号协同工作即可检测 SMBUS 超时。每当 BTIM1 检测到 SCL 下降沿时对计数值进行清零，每当 SCL 为低电平时 BTIM1 进行累加计数，当低电平持续时间超过 ARR 所设定的预期时，BTIM1 发生溢出以指示检测到 SMBUS 超时。

主要配置步骤如下：

步骤 1：设置 BITM1_SMCR.SMS 为 010，使 BTIM1 工作于门控计数模式；

步骤 2：设置 BITM1_SMCR.TRGISRC 为 0100，BTIM1 门控信号来源为 I2C_SCL；

步骤 3：设置 BITM1_SMCR.TRGIPOL 为 1，BTIM1 门控电平为低电平有效；

步骤 4：设置 BITM1_SMCR.RSTISRC 为 0100，BTIM1 计数值清零信号来源为 I2C_SCL；

步骤 5：设置 BITM1_SMCR.RSTIPOL 为 1， BTIM1 计数值清零信号为下降沿有效；

步骤 6：根据 SMBUS 所规定的 SCL 低电平超时时间，合理配置 BTIM1_ARR。

18.4.9 I2C 中断

I2C 控制寄存器 I2C_CR 的 SI 位域为中断标志位。当 I2C 状态寄存器 I2C_STAT 的 STAT 位域值发生改变（变成 0xF8 除外）时，I2C_CR.SI 标志位就会被置位，同时产生中断请求。

在用户 I2C 中断服务程序中，应查询 I2C 状态寄存器 I2C_STAT 的 STAT 位域值获取 I2C 总线的状态，以确定中断产生原因。设置 I2C_CR.SI 为 0 清除该标志位。

18.4.10 工作模式

I2C 控制器支持 4 种工作模式：主机发送模式、主机接收模式、从机发送模式、从机接收模式。另外还支持广播接收模式，其工作方式和从机接收模式类似。

18.4.10.1 主机发送模式

主机发送模式下，主机主动发送多个字节到从机。

SCL 串行时钟由主机控制产生，因此需要先根据传输波特率设置 I2C_BRR 寄存器并设置 I2C_BRREN 寄存器的 EN 位域为 1，然后启动传输。

主机设置 I2C_CR.STA 为 1，通知控制器发送 START 起始信号，控制器收到通知后检测总线是否空闲，当总线空闲时，主机控制器向总线发送一个 START 起始信号。如果发送成功，状态码 I2C_STAT 变为 0x08，中断标志位 I2C_CR.SI 被置 1。

主机发送完 START 起始信号后，需要软件设置 I2C_CR.STA 为 0，然后将从机地址和写标志位（SLA+W）写入到 I2C 数据寄存器 I2C_DR；清零 I2C_CR.SI 位，主机控制器将发送 SLA+W 到 I2C 总线上；当主机发送完 SLA+W 并收到从机 ACK 应答信号后，状态码 I2C_STAT 变为 0x18，中断标志位 I2C_CR.SI 被置 1。

此后主机根据应用需要，发送多个字节的用户自定义数据并检测 ACK 应答信号，每个字节的发送过程和发送 SLA+W 数据帧类似。

主机在发送数据过程中，如果收到 NACK 应答信号，表明从机不再接收主机发送的数据（从机的 I2C_CR.AA 被清零），主机应发送 STOP 信号结束本次数据传输，或者发送 Repeated START 重复起始信号开启新一轮传输。

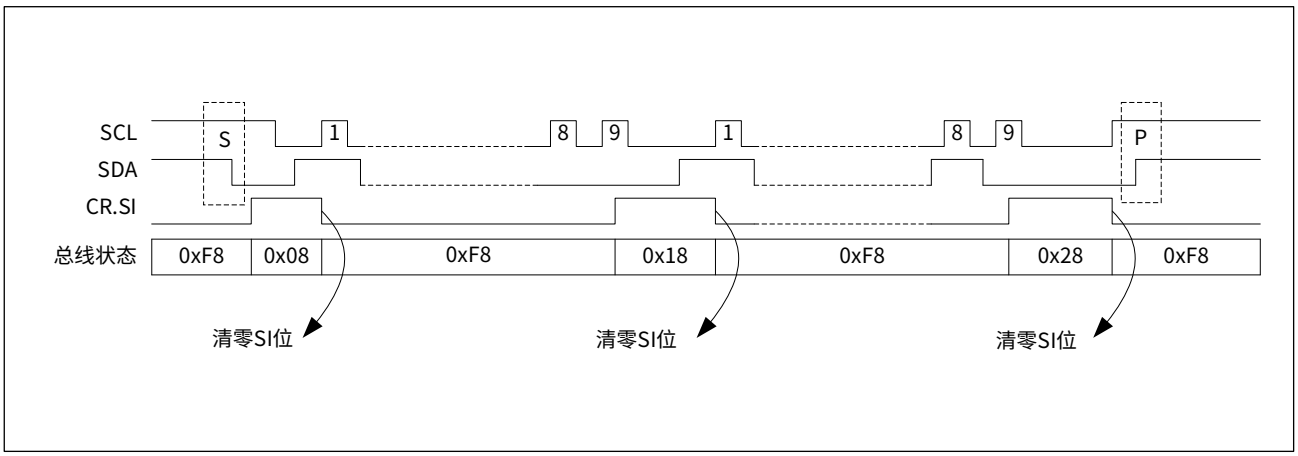
当主机发送完成所有数据后，设置 I2C_CR.STO 为 1，通知控制器数据已发送完成，待发送 STOP 停止信号。清零 I2C_CR.SI 位，主机控制器将发送 STOP 停止信号到 I2C 总线上，完成本次数据传输，释放总线。

当主机发送完成所有数据后，也可以不发送 STOP 停止信号，而是直接发送 Repeated START 重复起始信号，继续占用总线进行新一轮数据传输。

当主机由于总线冲突丢失仲裁时，会进入未寻址从机接收模式（状态码 I2C_STAT = 0x38）。

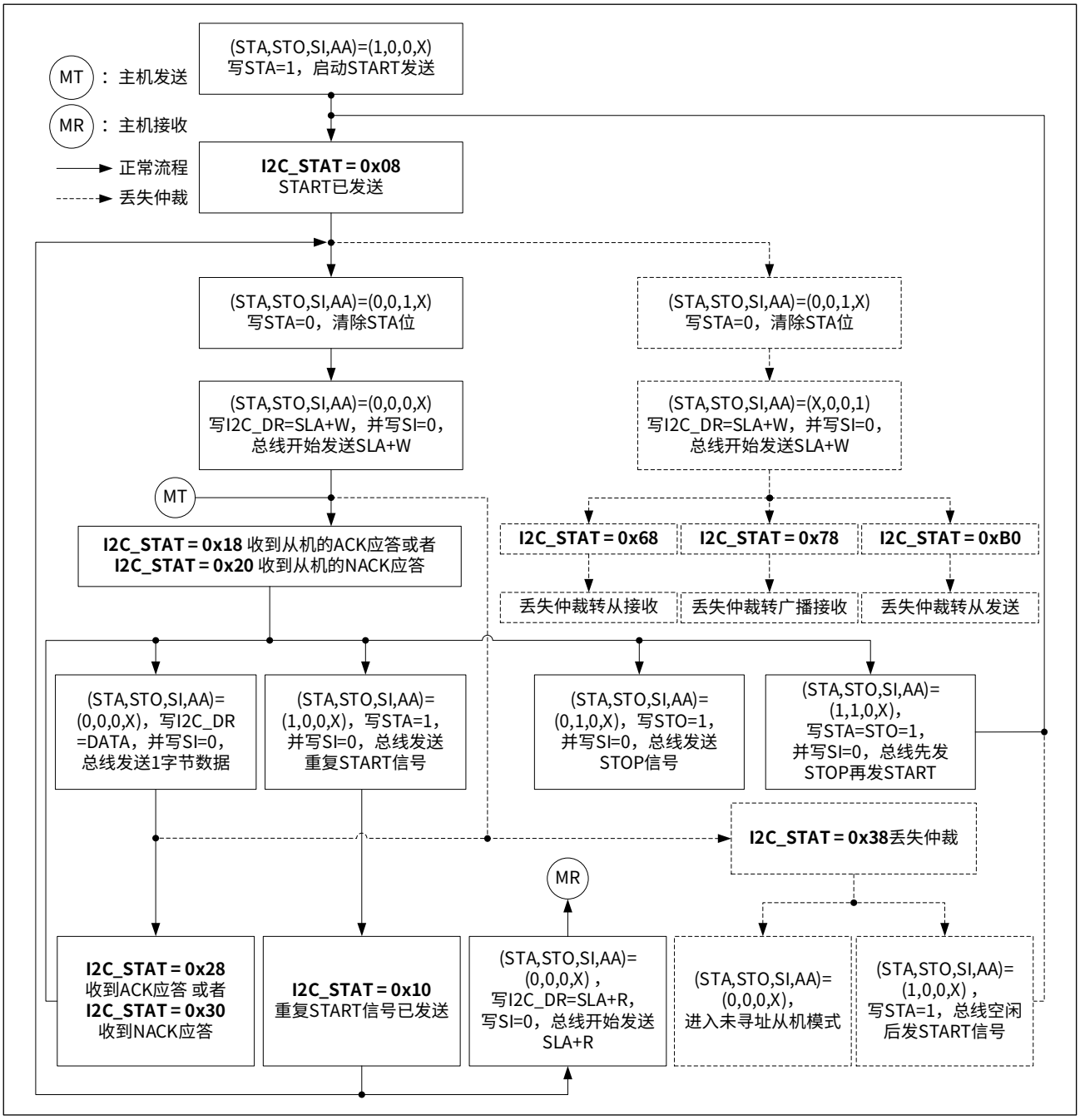
主机发送模式下状态同步图如下图所示：

图 18-8 主机发送模式状态同步图



主机发送模式下数据发送流程图及状态寄存器值如下图所示：

图 18-9　主机发送模式流程和寄存器状态



18.4.10.2 主机接收模式

主机接收模式用于接收从机发送的多个数据，主机每接收到 1 字节数据后会回应 ACK 应答信号。

SCL 串行时钟由主机控制产生，因此需要先根据传输波特率设置主机的 I2C_BRR 寄存器并设置 I2C_BRREN 寄存器的 EN 位域为 1，然后启动传输。

主机设置 I2C_CR.STA 为 1，通知控制器发送 START 起始信号，控制器收到通知后检测总线是否空闲，当总线空闲时，主机控制器向总线发送一个 START 起始信号。如果发送成功，状态码 I2C_STAT 变为 0x08，中断标志位 I2C_CR.SI 被置 1。

主机发送完 START 起始信号后，需要软件设置 I2C_CR.STA 为 0，然后将从机地址和读标志位（SLA+R）写入到 I2C 数据寄存器 I2C_DR；清零 I2C_CR.SI 标志位，主机控制器将发送 SLA+R 到 I2C 总线上；当主机发送完 SLA+R 并收到从机 ACK 应答信号后，状态码 I2C_STAT 变为 0x40，中断标志位 I2C_CR.SI 被置 1。

此后，主机设置 I2C_CR.AA 为 1，并清零 I2C_CR.SI 位，开始接收从机发送的数据，每接收完 1 字节数据后，都要回复一个 ACK 应答信号。在主机接收过程中，应注意：

1. 为保证每接收到 1 字节数据后都能正确产生 I2C_CR.SI 中断信号，需要在收到 1 字节数据后及时将 I2C_CR.SI 位清除。
2. 在接收最后一个字节前需要将 I2C_CR.AA 清零，即主机在接收到最后一个字节时不产生 ACK 应答信号，以此来通知从机停止数据发送。

主机在接收数据过程中，如果从机由于某种原因不再发送主机所需要的数据（从机的 I2C_CR.AA 被清零），主机后续将收到全 1 信号，此时主机需要在应用层对数据进行判决，判定从机为无响应状态，应发送 STOP 停止信号来结束本次传输，或发送 Repeated START 重复起始信号开启新一轮传输。

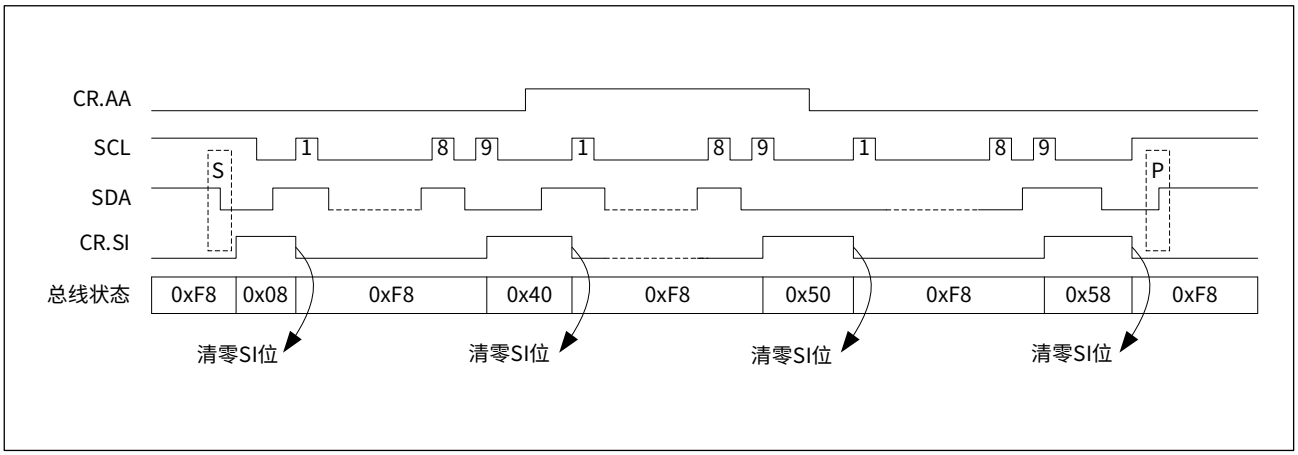
当主机接收完成所有数据后，设置 I2C_CR.STO 为 1，通知控制器数据已接收完成，待发送 STOP 停止信号。清零 I2C_CR.SI 位，主机控制器发送 STOP 停止信号到 I2C 总线上，完成本次数据传输，释放总线。

当主机接收完成所有数据后，也可以不发送 STOP 停止信号，而是直接发送 Repeated START 重复起始信号，继续占用总线进行新一轮数据传输。

当主机由于总线冲突丢失仲裁时，会进入未寻址从机接收模式（状态码 I2C_STAT = 0x38）。

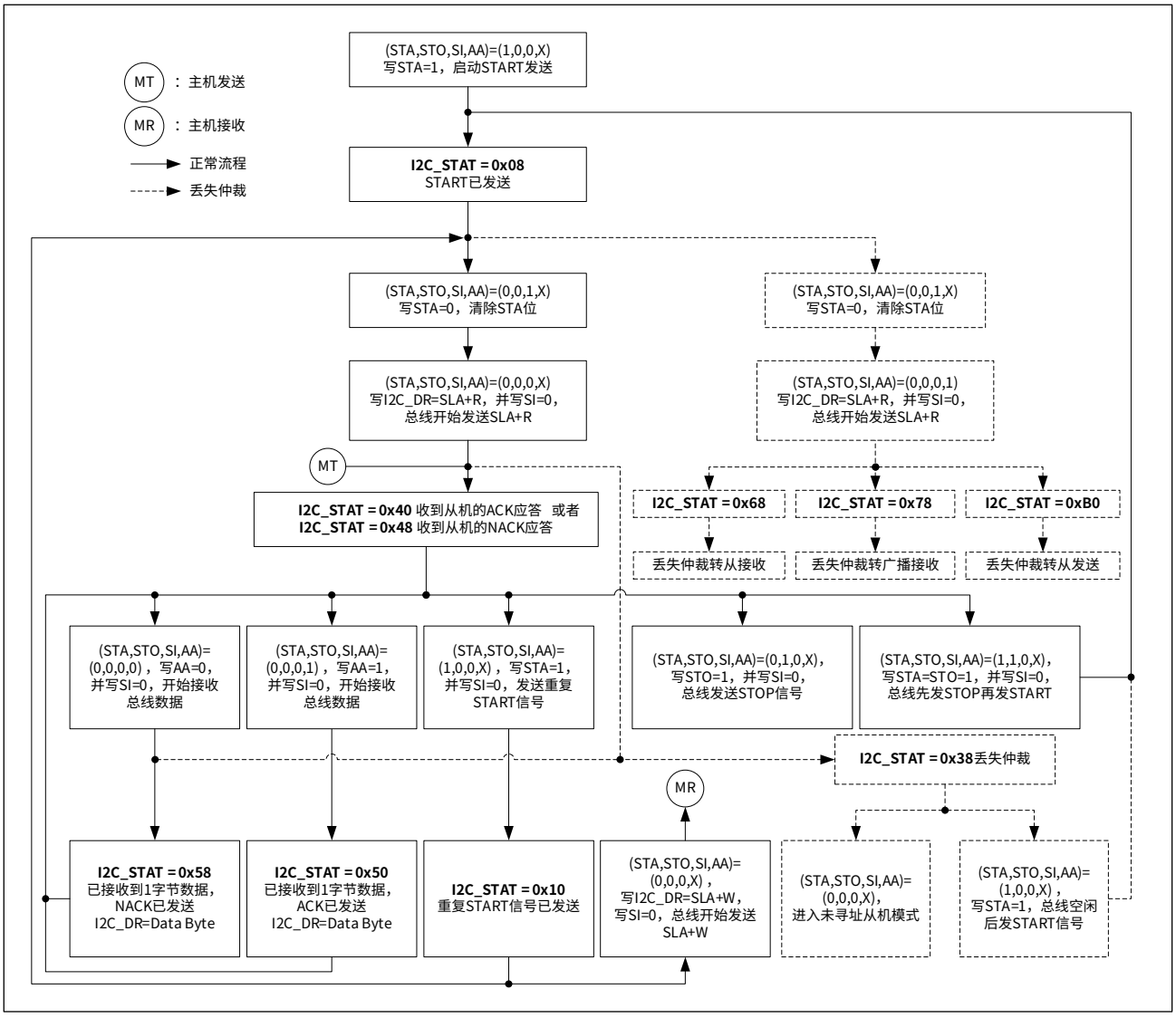
主机接收模式下状态同步图如下图所示：

图 18-10　主机接收模式状态同步图



主机接收模式下数据接收流程图及状态寄存器值如下图所示：

图 18-11　主机接收模式流程和寄存器状态



18.4.10.3 从机接收模式

从机接收模式下，从机接收主机发送的数据。

在传输之前，从机需要设定从机地址：将从机地址写入到 3 个从机地址寄存器 I2C_ADDR0、I2C_ADDR1、I2C_ADDR2 任意 1 个中；设置 I2C_CR.AA 为 1 以响应主机的寻址；I2C_BRR 寄存器设置值无效，不用设置。

完成上述初始化工作后，从机进入空闲状态（未寻址从机接收模式），等待被主机发送的写信号（SLA+W）寻址。当从机接收到 SLA+W 后，如果地址匹配到 I2C_ADDR0/1/2，则从机回应 ACK 应答，并进入已寻址从机接收模式，状态码 I2C_STAT 变为 0x60，中断标志位 I2C_CR.SI 同时被置 1。此时必须清除 I2C_CR.SI 位，以便从总线上接收主机发送的数据。

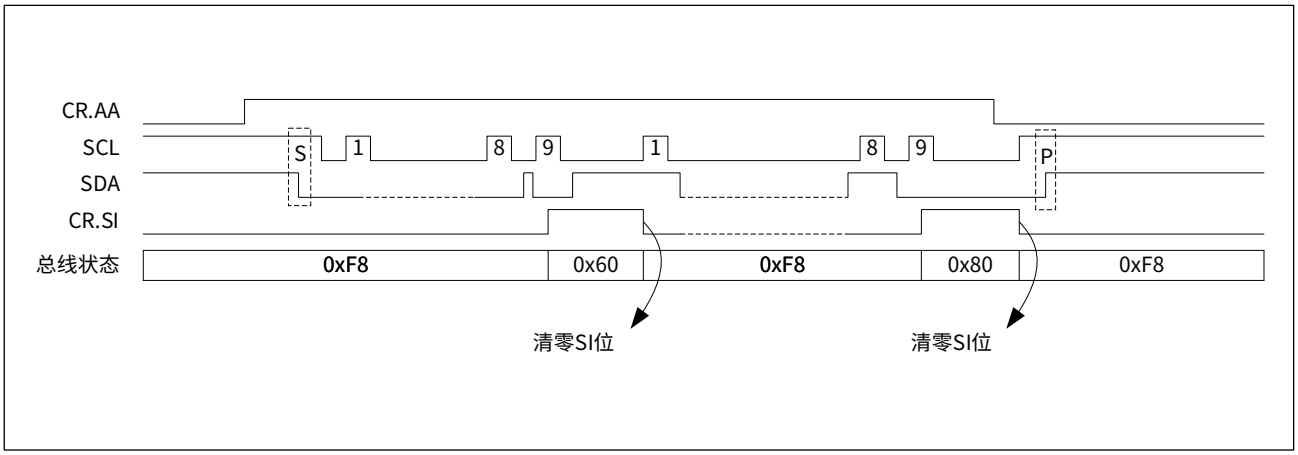
从机每接收到 1 字节数据都要回应 1 个 ACK 应答，应用程序读取完该字节数据后，必须将 I2C_CR.SI 位清零，为接收下一字节数据做好准备。

从机在接收数据过程中，如果 I2C_CR.AA 被清零，则从机将在接收到下一字节时返回 NACK 信号，从机自身状态也切换到未寻址从机接收模式，结束与主机的通信，不再接收数据，且 I2C_DR 寄存器保持之前接收到的数据。由该特性可知，从机应用程序可通过设置 I2C_CR.AA 为 0 主动将从机从已寻址从机接收模式切换到未寻址从机接收模式。

当主机在 SLA+ 读写阶段由于总线冲突丢失仲裁时会进入未寻址从机接收模式，之后如果接收到符合本机地址的 SLA+W 并回应 ACK 后（状态码 I2C_STAT = 0x68），则会进入已寻址从机接收模式。

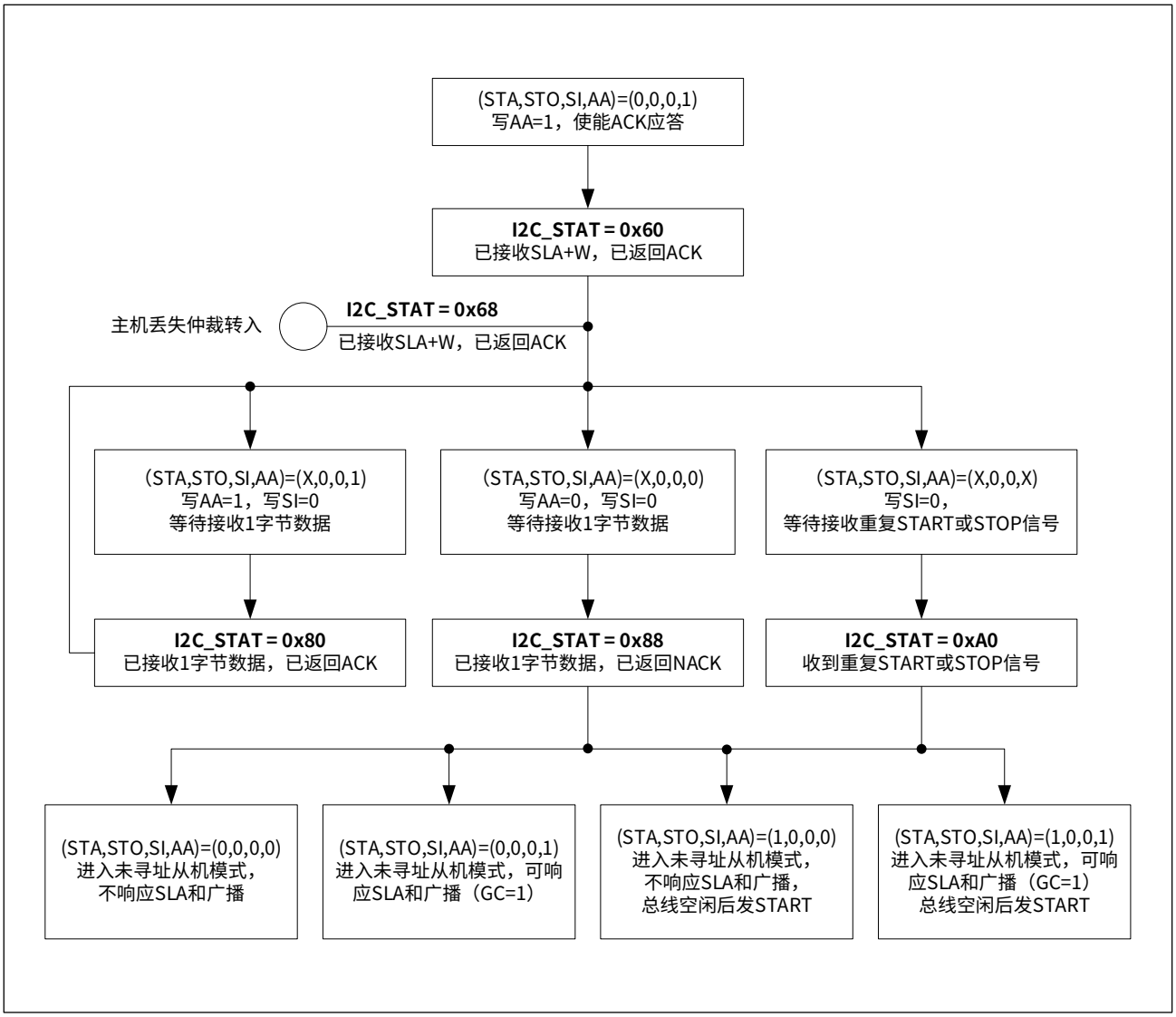
从机接收模式下状态同步图如下图所示：

图 18-12　从机接收模式状态同步图



从机接收模式下数据接收流程图及状态寄存器值如下图所示：

图 18-13　从机接收模式流程和寄存器状态



18.4.10.4 从机发送模式

从机发送模式下，数据由从机发送给主机。

在传输之前，从机需要设定从机地址：将从机地址写入到 3 个从机地址寄存器 I2C_ADDR0、I2C_ADDR1、I2C_ADDR2 任意 1 个中；设置 I2C_CR.AA 为 1 以响应主机的寻址；I2C_BRR 寄存器设置值无效，不用设置。

完成上述初始化工作后，从机进入空闲状态（未寻址从机接收模式），等待被主机发送的读信号（SLA+R）寻址。当从机接收到 SLA+R 后，如果地址匹配到 I2C_ADDR0/1/2，则从机回应 ACK 应答，并进入已寻址从机发送模式，状态码 I2C_STAT 变为 0xA8，中断标志位 I2C_CR.SI 同时被置 1。此时应及时将待发送的数据写入 I2C_DR 寄存器，并清除 I2C_CR.SI 位，等待发送 1 字节数据，并在每个字节数据发送完成后进行 ACK 应答确认，直到全部数据发送完毕。

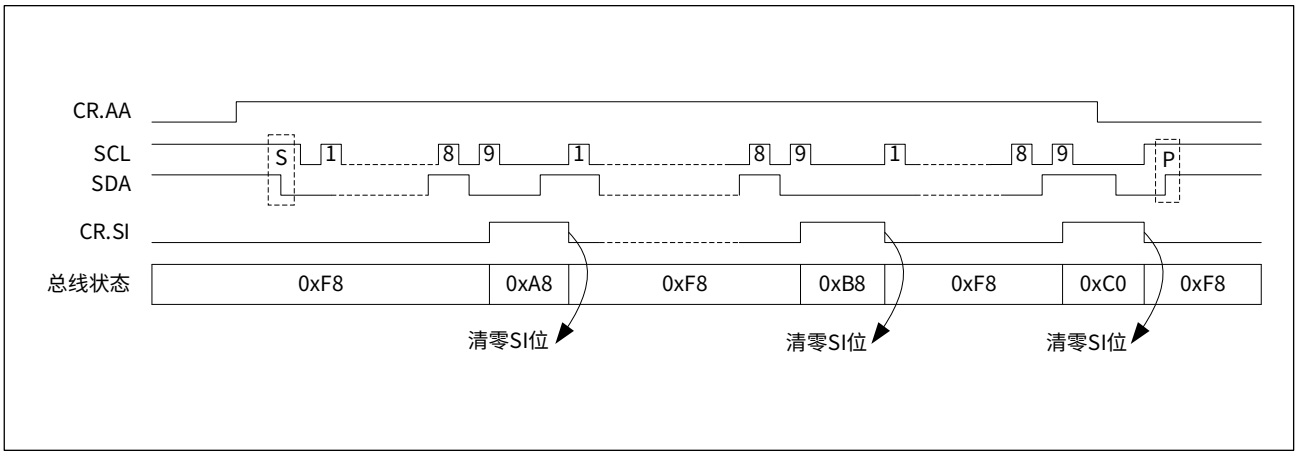
如果在传输过程中从机收到 NACK 应答，则从机不再发送数据，并进入未寻址从机接收模式。

如果在传输过程中从机主动将 I2C_CR.AA 设置为 0，则从机在发送完最后 1 字节有效数据后，将自身切换为未寻址从机接收模式，此后主机从总线上读数据将得到 0xFF。

当主机在 SLA+ 读写阶段由于总线冲突丢失仲裁时会进入未寻址从机接收模式，之后如果接收到符合本机地址的 SLA+R 并回应 ACK 后（状态码 I2C_STAT = 0xB0），则会进入已寻址从机发送模式。

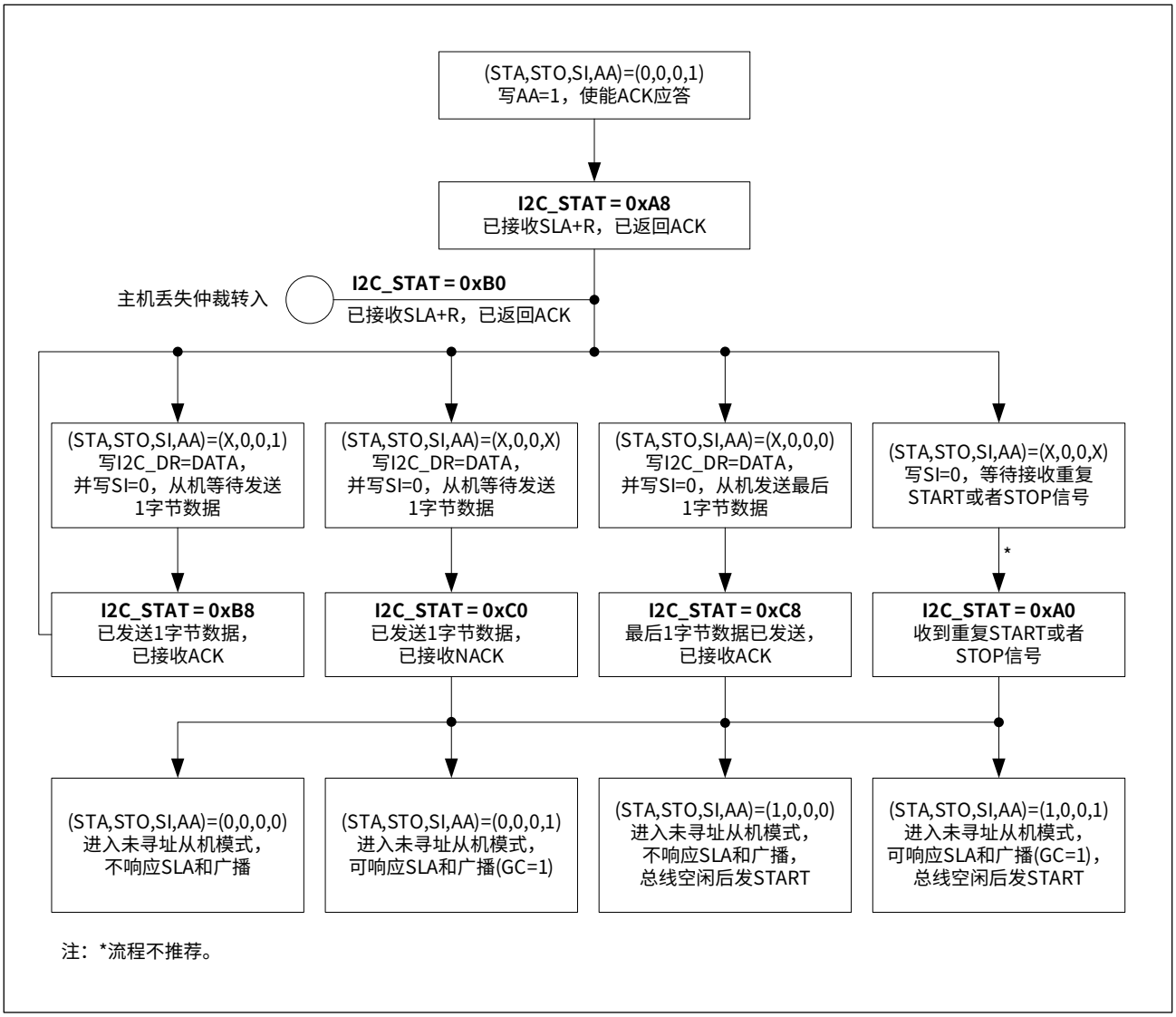
从机发送模式下状态同步图如下图所示：

图 18-14　从机发送模式状态同步图



从机发送模式下数据发送流程图及状态寄存器值如下图所示：

图 18-15　从机发送模式流程和寄存器状态



18.4.10.5 广播接收模式

广播接收模式是一种特殊的从机接收模式，当从机处于空闲状态（未寻址从机接收模式），收到主机发送的 SLA+W，且 SLA 为广播地址 0x00 时进入广播接收模式。该模式下数据接收过程和从机接收模式相同，但 I2C 总线状态码不同。

从机要接收主机发送的广播信息，需设置 I2C_CR.AA 为 1 以及 I2C_ADDR0.GC 为 1 以响应主机的广播寻址和广播数据； 3 个从机地址寄存器 I2C_ADDR0/1/2 不用设置；I2C_BRR 设置值无效，不用设置。

完成上述初始化工作后，从机进入空闲状态（未寻址从机接收模式），等待被主机发送的写信号（SLA+W）寻址。当从机接收到 SLA+W 后，如果地址匹配到广播地址 0x00，则从机回应 ACK 应答，并进入广播接收模式，状态码 I2C_STAT 变为 0x70，中断标志位 I2C_CR.SI 同时被置 1。此时必须及时清除 I2C_CR.SI 位，以便从总线上接收主机发送的数据。

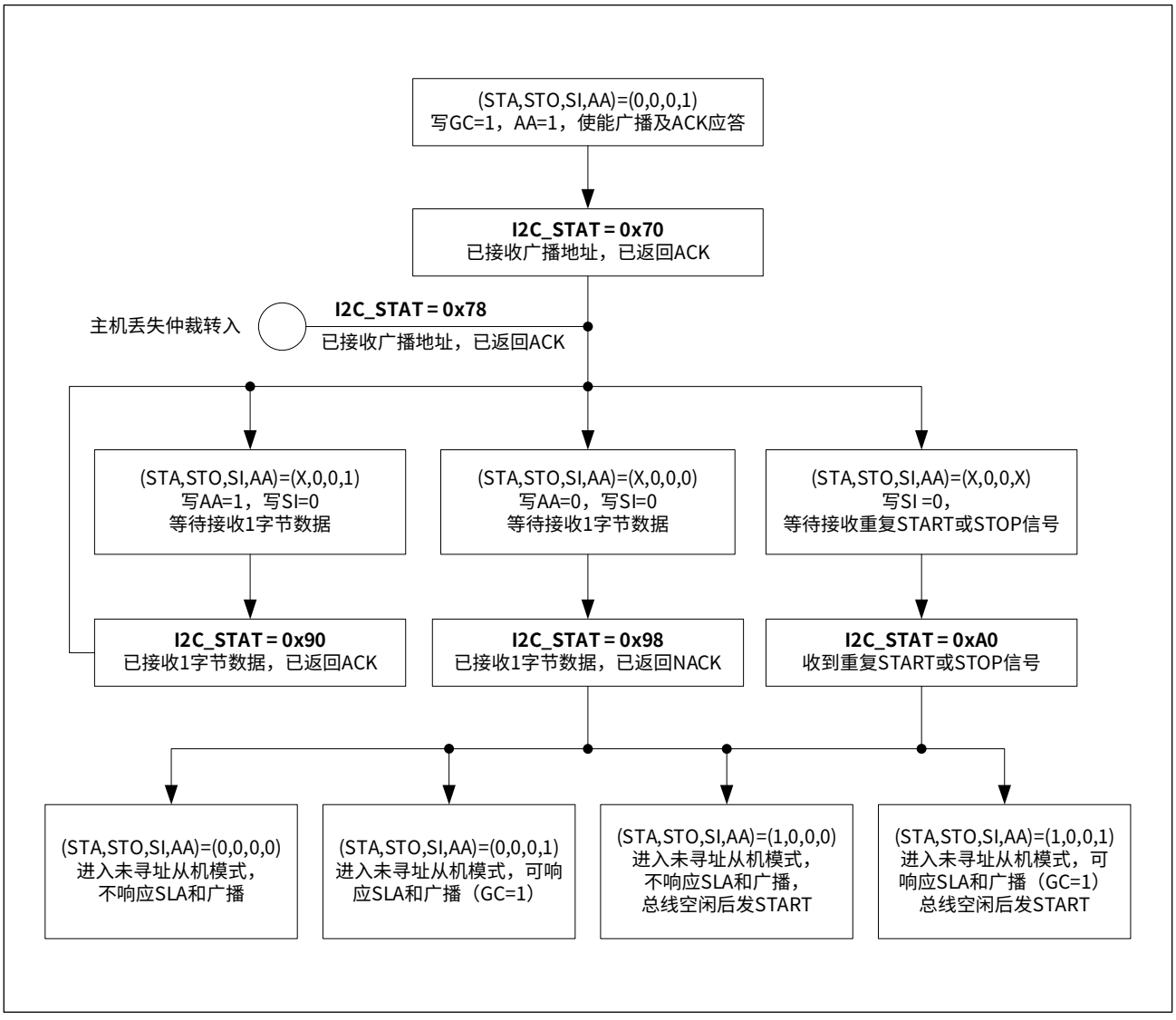
从机每接收到 1 字节数据都要回应 1 个 ACK 应答，应用程序读取完该字节数据后，必须将 I2C_CR.SI 位清零，为接收下 1 字节数据做好准备。

从机在接收数据过程中，如果 I2C_CR.AA 被清零，则从机将在接收到下一字节时返回 NACK 信号，从机自身状态也切换为未寻址从机接收模式，结束与主机的通信，不再接收数据，且 I2C_DR 寄存器保持之前接收到的数据。由该特性可知，从机应用程序可通过设置 I2C_CR.AA 为 0 主动将从机从已寻址广播接收模式切换未寻址从机接收模式。

当主机在 SLA+ 读写阶段由于总线冲突丢失仲裁时会进入未寻址从机接收模式，之后如果接收到符合广播地址的 SLA+W 并回应 ACK 后（状态码 I2C_STAT = 0x78），则会进入已寻址广播接收模式。

广播接收模式下数据接收流程图及状态寄存器值如下图所示：

图 18-16　广播接收模式流程和寄存器状态



18.4.11 多主机通信

在一些应用中，1 个 I2C 总线上有 2 个或多个主机同时访问从机，并有可能同时在传送数据，此时 SDA 总线上会存在数据冲突。

CW32L011 的 I2C 能进行 SDA 总线上的数据冲突检测和仲裁，实现多主机应用。如果两个主机同时发送数据，检测到冲突的主机会丢失仲裁并进入未寻址从机模式，未检测到冲突的主机会赢得仲裁并继续主导本次数据通信流程。

18.4.12 I2C 状态码

I2C 总线状态通过 I2C 状态寄存器 I2C_STAT 来标识，共 26 个正常接收或发送状态，和 2 个特殊状态（0xF8:I2C 总线无可用信息；0x00：总线错误）。

I2C 无论处于主机发送、主机接收、从机接收、从机发送或广播接收模式，当状态寄存器 I2C_STAT 的内容改变时，都会将 I2C_CR.SI 置位，且产生 I2C 中断。

I2C 状态码如下表所示：

表 18-4 I2C 状态码

工作模式	状态码	含义
主机发送模式	08H	已发送起始信号
	10H	已发送重复起始信号
	18H	已发送 SLA+W，已接收 ACK
	20H	已发送 SLA+W，已接收 NACK
	28H	已发送 I2C_DR 中的数据，已接收 ACK
	30H	已发送 I2C_DR 中的数据，已接收 NACK
	38H	主机在发送 SLA+W 阶段或者发送数据阶段丢失仲裁
主机接收模式	08H	已发送起始信号
	10H	已发送重复起始信号
	38H	主机在发送 SLA+R 阶段或者回应 NACK 阶段丢失仲裁
	40H	已发送 SLA+R，已接收 ACK
	48H	已发送 SLA+R，已接收 NACK
	50H	已接收数据字节，ACK 已返回
	58H	已接收数据字节，NACK 已返回
从机接收模式	60H	已接收自身的 SLA+W，已返回 ACK
	68H	当主机时在 SLA+ 读写阶段丢失仲裁，已接收自身的 SLA+W，已返回 ACK
	80H	前一次寻址使用自身从地址，已接收数据字节，已返回 ACK
	88H	前一次寻址使用自身从地址，已接收数据字节，已返回 NACK
	A0H	已寻址从机等待接收数据时，接收到停止条件或重复起始条件

工作模式	状态码	含义
从机发送模式	A8H	已接收自身的 SLA+R，已返回 ACK
	B0H	当主机时在 SLA+ 读写阶段丢失仲裁，已接收自身 SLA+R，已返回 ACK
	B8H	已发送数据字节，已接收 ACK
	C0H	已发送数据字节，已接收 NACK
	C8H	从机最后一个数据字节已被发送，并已接收 ACK
广播接收模式	70H	已接收广播地址（0x00），已返回 ACK
	78H	当主机时在 SLA+ 读写阶段丢失仲裁，已接收广播地址，已返回 ACK
	90H	前一次寻址使用广播地址，已接收数据字节，已返回 ACK
	98H	前一次寻址使用广播地址，已接收数据字节，已返回 NACK
	A0H	已寻址从机等待接收数据时，接收到停止条件或重复起始条件
其它	F8H	无可用的相关状态信息，I2C_CR.SI=0
	00H	传输过程出现总线错误，或外部干扰使 I2C 进入未定义的状态

特殊状态码 F8H，表示当前时刻没有任何有用信息，还不能确定当前总线的状态，因为 I2C_CR.SI 还没有被置位，无中断产生。这种情况在其它状态和 I2C 模块还未开始执行串行传输之前出现。

特殊状态码 00H，表示 I2C 串行传输过程中出现了总线错误，如 START 或者 STOP 信号出现在数据帧的错误位置上（包括在串行传输过程中的地址字节、数据字节或应答位）或者当外部干扰影响到内部 I2C 模块信号等。总线错误出现时，I2C_CR.SI 标志位会立即被置位，且设备立即被切换到未寻址从机接收模式，释放 SDA 和 SCL，并将 I2C_DR 寄存器清零。

当检测到 I2C 总线的 STAT 为总线错误（状态码为 00H）时，由于 I2C 总线为持续使能状态，并且对 I2C 模块来说错误并没有清除，SI 会持续保持为 1，即 SI 无法被清除。

总线错误清除方法：

1. 向总线发送 STOP 信号：置位 STO 位并清除 SI 位（由于当前处于总线错误状态，控制器并不会将 STOP 信号实际发送到总线上），STO 位会被硬件自动清 0，释放总线到正常空闲状态。
2. 如果置 STO 位仍然无法清除 SI 位，说明时序已被打乱，需要依次设置 I2C_CR.EN 为 0 和 1，即对 I2C 模块进行关闭并重启，然后设置 SI 为 0 清除 SI 位。

在各工作模式下，I2C 总线状态转换图如下图所示。注意，两种正常状态之间转换时，当未完成动作（如发送 SLA 过程中），即进入新的状态之前，状态码会出现短时的过渡状态，0xF8。用户可不关心，且不会产生中断。

图 18-17　主机发送 / 接收模式状态转换图

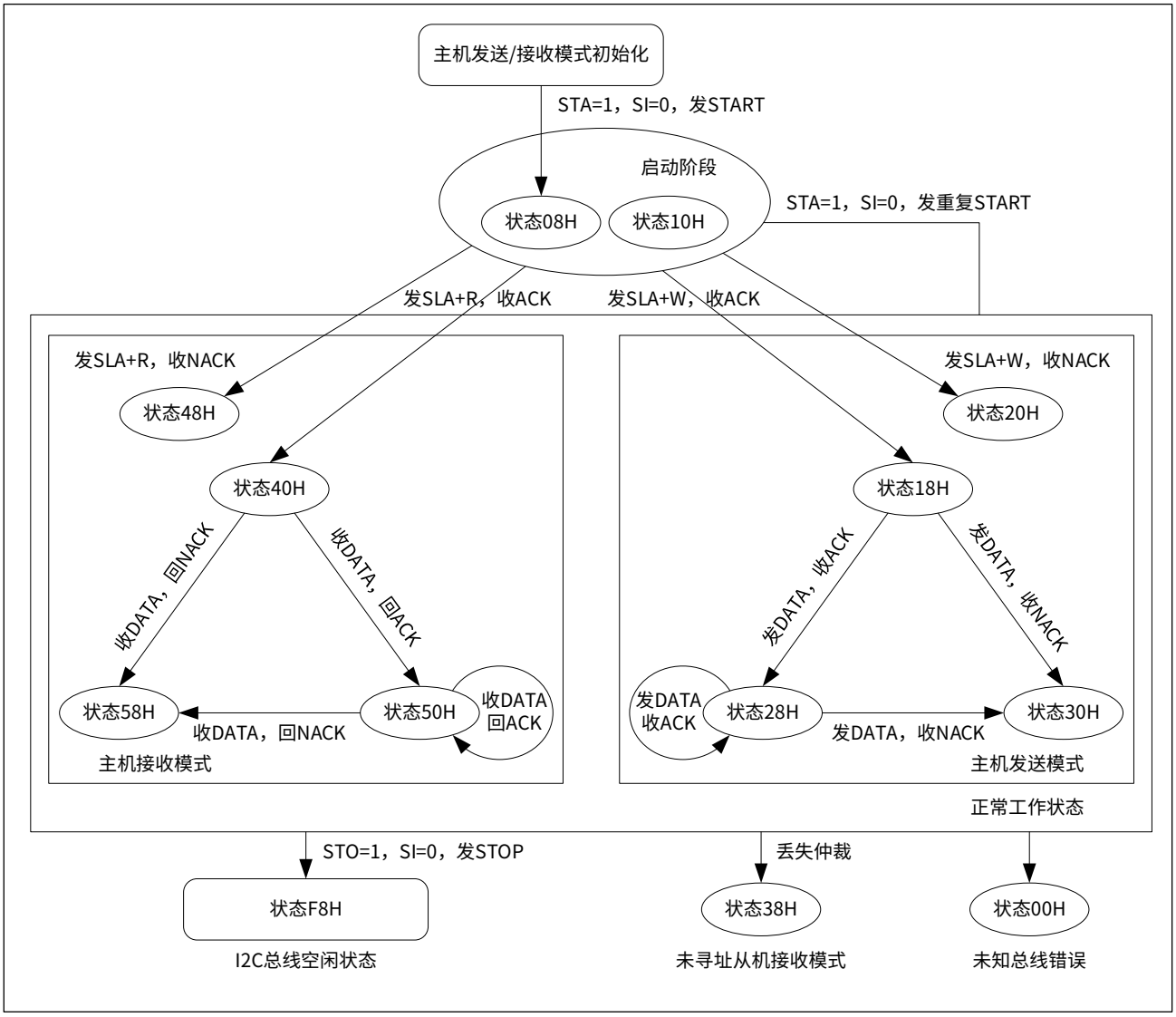
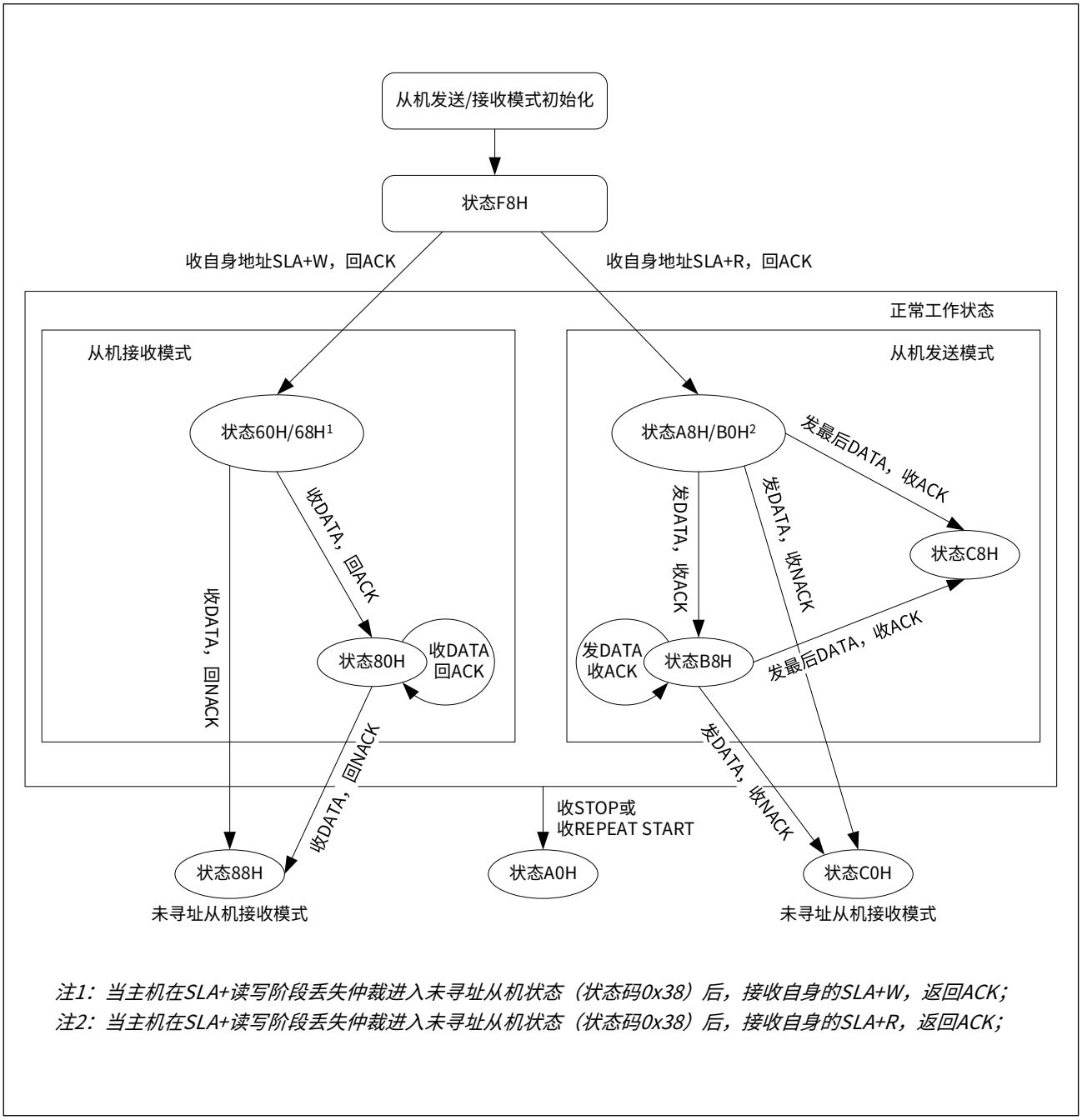
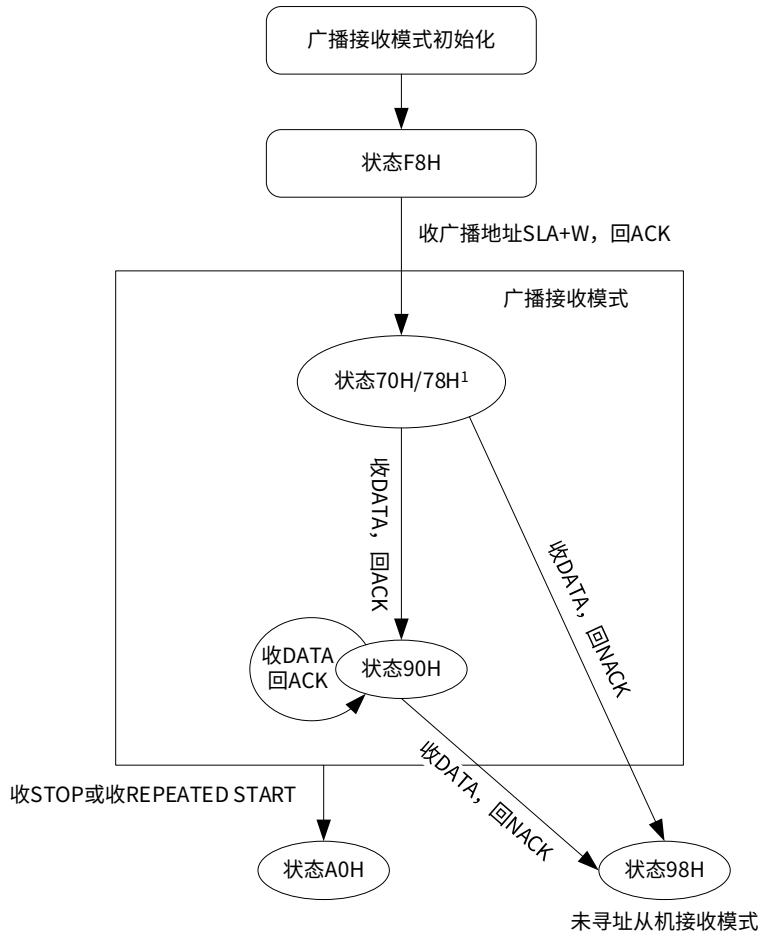


图 18-18　从机发送 / 接收模式状态转换图



注1：当主机在SLA+读写阶段丢失仲裁进入未寻址从机状态（状态码0x38）后，接收自身的SLA+W，返回ACK；
注2：当主机在SLA+读写阶段丢失仲裁进入未寻址从机状态（状态码0x38）后，接收自身的SLA+R，返回ACK；

图 18-19　广播接收模式状态转换图



注1：当主机在SLA+读写阶段丢失仲裁进入未寻址从机状态（状态码0x38）后，接收广播地址SLA+W，返回ACK；

18.5 编程示例

18.5.1 主机发送示例

步骤 1：按 GPIO 章节引脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的引脚，并配置 SCL、SDA 引脚为开漏输出模式；

步骤 2：设置 SYSCTRL_APBEN2.I2C 为 1，使能 I2C 模块时钟；

步骤 3：向 SYSCTRL_APBRST2.I2C 依次写入 0、1，复位 I2C 模块；

步骤 4：配置 I2C_BRR，使 SCL 的时钟速率符合应用需求；

步骤 5：设置 I2C_BRREN 为 1，使能 SCL 时钟发生器；

步骤 6：设置 I2C_CR.EN 为 1，使能 I2C 模块；

步骤 7：设置 I2C_CR.STA 为 1，总线尝试发送 START 信号；

步骤 8：等待 I2C_CR.SI 变为 1，START 信号已发送到总线上；

步骤 9：查询 I2C_STAT，如果该寄存器值为 0x08 或 0x10，继续执行下一步骤，否则进行出错处理；

步骤 10：向 I2C_DR 中写入 SLA+W，设置 I2C_CR.STA 为 0，设置 I2C_CR.SI 为 0，发送 SLA+W；

步骤 11：等待 I2C_CR.SI 变为 1，SLA+W 已发送到总线上；

步骤 12：查询 I2C_STAT，如果该寄存器值为 0x18，继续执行下一步骤，否则进行出错处理；

步骤 13：向 I2C_DR 写入待发送的数据，设置 I2C_CR.SI 为 0，发送数据；

步骤 14：等待 I2C_CR.SI 变为 1，数据已发送到总线上；

步骤 15：查询 I2C_STAT，如果该寄存器值为 0x28，继续执行下一步骤，否则进行出错处理；

步骤 16：如待发送的数据未完成，则跳转到步骤 13 继续执行；

步骤 17：设置 I2C_CR.STO 为 1，设置 I2C_CR.SI 为 0，发送 STOP 停止信号，结束本次数据传输。

18.5.2 主机接收示例

步骤 1：按 GPIO 章节引脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的引脚，并配置 SCL、SDA 引脚为开漏输出模式；

步骤 2：设置 SYSCTRL_APBEN2.I2C 为 1，使能 I2C 模块时钟；

步骤 3：向 SYSCTRL_APBRST2.I2C 依次写入 0、1，复位 I2C 模块；

步骤 4：配置 I2C_BRR，使 SCL 的时钟速率符合应用需求；

步骤 5：设置 I2C_BRREN 为 1，使能 SCL 时钟发生器；

步骤 6：设置 I2C_CR.EN 为 1，使能 I2C 模块；

步骤 7：设置 I2C_CR.STA 为 1，总线尝试发送 START 信号；

步骤 8：等待 I2C_CR.SI 变为 1，START 信号已发送到总线上；

步骤 9：查询 I2C_STAT，如果寄存器值为 0x08 或 0x10，继续执行下一步骤，否则进行出错处理；

步骤 10：向 I2C_DR 写入 SLA+R，设置 I2C_CR.STA 为 0，设置 I2C_CR.SI 为 0，发送 SLA+R；

步骤 11：等待 I2C_CR.SI 变为 1，SLA+R 已发送到总线上；

步骤 12：查询 I2C_STAT，如果寄存器值为 0x40（已收到 ACK），继续执行下一步骤，否则进行出错处理；

步骤 13：设置 I2C_CR.AA 为 1，使能应答标志；

步骤 14：设置 I2C_CR.SI 为 0，等待接收 1 字节数据（主机发送时钟，从机在时钟作用下发送数据）；

步骤 15：等待 I2C_CR.SI 变为 1（主机完成 1 字节数据接收并已回应 ACK 信号），从 I2C_DR 读取已接收到的数据；

步骤 16：查询 I2C_STAT，如果该寄存器值为 0x50 或 0x58，继续执行下一步骤，否则进行出错处理；

步骤 17：如果待接收的数据只差最后一个字节，设置 I2C_CR.AA 为 0，使能非应答标志；

步骤 18：如待接收的数据未完成，则跳转到步骤 14 继续执行；

步骤 19：设置 I2C_CR.STO 为 1，设置 I2C_CR.SI 为 0，发送 STOP 停止信号，结束本次数据传输。

18.5.3 从机接收示例

步骤 1：按 GPIO 章节引脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的引脚，并配置 SCL、SDA 引脚为开漏输出模式；

步骤 2：设置 SYSCTRL_APBEN2.I2C 为 1，使能 I2C 模块时钟；

步骤 3：向 SYSCTRL_APBRST2.I2C 依次写入 0、1，复位 I2C 模块；

步骤 4：设置 I2C_CR.EN 为 1，使能 I2C 模块；

步骤 5：配置 I2C_ADDR0 为从机地址；

步骤 6：设置 I2C_CR.AA 为 1，使能应答标志；

步骤 7：等待 I2C_CR.SI 变为 1，被 SLA+W 寻址；

步骤 8：查询 I2C_STAT，如果该寄存器值为 0x60，继续执行下一步骤，否则进行出错处理；

步骤 9：设置 I2C_CR.SI 为 0，等待主机发送数据，并回应 ACK 信号；

步骤 10：等待 I2C_CR.SI 变为 1，从 I2C_DR 中读取已接收到的数据；

步骤 11：查询 I2C_STAT，如果该寄存器值为 0x80，继续执行下一步骤，否则进行出错处理；

步骤 12：如待接收的数据未完成，则跳转到步骤 9 继续执行；

步骤 13：设置 I2C_CR.AA 为 0，设置 I2C_CR.SI 为 0，从机切换到未寻址从机接收模式，且不响应主机寻址。

18.5.4 从机发送示例

步骤 1：按 GPIO 章节引脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的引脚，并配置 SCL、SDA 引脚为开漏输出模式；

步骤 2：设置 SYSCTRL_APBEN2.I2C 为 1，使能 I2C 模块时钟；

步骤 3：向 SYSCTRL_APBRST2.I2C 依次写入 0、1，复位 I2C 模块；

步骤 4：设置 I2C_CR.EN 为 1，使能 I2C 模块；

步骤 5：配置 I2C_ADDR0 为从机地址；

步骤 6：设置 I2C_CR.AA 为 1，使能应答标志；

步骤 7：等待 I2C_CR.SI 变为 1，被 SLA+R 寻址；

步骤 8：查询 I2C_STAT，如果该寄存器的值为 0xA8，继续执行下一步骤，否则进行出错处理；

步骤 9：向 I2C_DR 写入待发送的数据，设置 I2C_CR.SI 为 0，准备发送数据；

步骤 10：等待 I2C_CR.SI 变为 1，表示数据已发送到总线上，并收到 ACK 或者 NACK 应答；

步骤 11：查询 I2C_STAT，如果该寄存器的值为 0xB8 时，继续执行下一步骤，否则进行出错处理；

步骤 12：如待发送的数据未完成，则跳转到步骤 9 继续执行；

步骤 13：设置 I2C_CR.AA 为 0，设置 I2C_CR.SI 为 0，从机切换到未寻址从机接收模式，且不响应主机寻址。

18.6 寄存器列表

I2C 基地址：I2C_BASE = 0x4000 5800

表 18-5 I2C 寄存器列表

寄存器名称	寄存器地址	寄存器描述
I2C_BRREN	I2C_BASE + 0x00	波特率计数器使能寄存器
I2C_BRR	I2C_BASE + 0x04	波特率计数器配置寄存器
I2C_CR	I2C_BASE + 0x08	控制寄存器
I2C_DR	I2C_BASE + 0x0C	数据寄存器
I2C_ADDR0	I2C_BASE + 0x10	从机地址 0 寄存器
I2C_STAT	I2C_BASE + 0x14	状态寄存器
I2C_ADDR1	I2C_BASE + 0x20	从机地址 1 寄存器
I2C_ADDR2	I2C_BASE + 0x24	从机地址 2 寄存器
I2C_MATCH	I2C_BASE + 0x28	从机地址匹配标志寄存器

18.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

18.7.1 I2C_BRREN 波特率计数器使能寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	EN	RW	I2C 总线 SCL 波特率计数器使能控制 0：禁止 1：使能 *注：主机时应使能 EN，从机时该位不影响。*

18.7.2 I2C_BRR 波特率计数器配置寄存器

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	BRR	RW	I2C 总线 SCL 波特率配置 $f_{SCL} = f_{PCLK} / 8 / (BRR+1)$ ，其中 BRR > 0

18.7.3 I2C_CR 控制寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:14	RFU	-	保留位，请保持默认值
13:11	SDAINSRC	RW	SDA 输入信号来源配置 000：I2C_SDA 引脚 001：VC1_OUT 010：VC2_OUT 其他：保留
10:8	SCLINSRC	RW	SCL 输入信号来源配置 000：I2C_SCL 引脚 001：VC1_OUT 010：VC2_OUT 其他：保留
7	RFU	-	保留位，请保持默认值
6	EN	RW	模块使能控制 0：禁止 1：使能
5	STA	RW	总线状态控制 W0：清零 STA W1：向总线发送 START 起始信号 *注 1：设置 STA 为 1 后，如果总线空闲，则发送 START 起始信号，如果总线忙则等待 I2C 停止后，发送 START 信号。* *注 2：如果设备已经在主机模式且已发送一个或多个字节，此时再设置 STA，I2C 总线将产生 Repeated START 重复起始信号。* *注 3：STA 可在任何时间置 1，包括从机模式。但硬件不会在完成 START 或 repeat START 信号发送后自动清 0，需要用户手动清除 STA。*
4	STO	RW	总线状态控制 W0：无功能 W1：向总线发送 STOP 停止信号 *注 1：硬件会在完成 STOP 信号发送后自动对 STO 清 0。* *注 2：如果在主机模式下 STA 和 STO 同时置 1，I2C 总线在发送 STOP 后马上发送 START。在从机模式下，禁止 STA 及 STO 同时置 1，以避免发出非法 I2C 帧。* *注 3：当总线上产生错误状态 (STAT 状态字为 00H) STO 也会置 1，但这种情况下 I2C 总线不会发送 STOP 停止信号。*

位域	名称	权限	功能描述
3	SI	RW	I2C 中断标志 R0：未发生 I2C 中断 R1：已发生 I2C 中断 W0：清除 I2C 中断标志并使状态机执行下一个动作 W1：无功能 *注 1: I2C 所有 26 种状态中出现一种，硬件就会置 1 此位 (F8H 除外)，此时软件通过 I2C_STAT 寄存器值，来确认总线当前状态。* *注 2：SI 需要软件清零。* *注 3：在 SI 被清 0 之前，SCL 低电平周期延长，传输暂停，该状态对于从机处理接收到的数据非常有用，可以确保准确处理前一数据再接收下一个数据。* *注 4：在软件清除 SI 前，软件应该准备好合适的寄存器设置。在 SI 被清除后，I2C 总线将会根据寄存器设置执行相应的操作。*
2	AA	RW	应答控制 0：在应答阶段发送 NACK 1：在应答阶段发送 ACK *注 1：对于已被寻址的从机，在从机接收模式下未回复 ACK 应答位或在从机发送模式下未接收到 ACK 应答位，该从机将切换为未寻址从机接收模式，无法接收数据直到其 AA 被置 1，且重新被主机寻址。* *注 2：特殊情况：从机发送模式时，从机发送最后一个字节给主机之前，清除 AA，发送完最后一个字节的位后，从机将切换为未被寻址的从机模式，和主机断开，状态寄存器 I2C_STAT 为 C8H。主机若再从总线上读数据，将得到 0xFF。*
1	RFU	-	保留位，请保持默认值
0	FLT	RW	I2C 滤波参数配置 0：高级滤波，更高的抗干扰性能 1：简单滤波，更快的通信速率 *注：详见 18.4.3 输入滤波器。*

18.7.4 I2C_DR 数据寄存器

Address offset: 0x0C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	DR	RW	数据寄存器 在发送模式下，写入待发送的数据 在接收模式下，读出接收到的数据

18.7.5 I2C_STAT 状态寄存器

Address offset: 0x14 Reset value: 0x0000 00F8

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:0	STAT	RO	I2C 状态寄存器，状态值的具体定义详见 18.4.12 I2C 状态码; STAT = F8H 时，表示无可用的相关状态信息，SI 将保持为 0。 其它 26 种状态，都会让 SI 置 1，且产生中断请求。

18.7.6 I2C_ADDR0 从机地址 0 寄存器

Address offset: 0x10 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:1	ADDR0	RW	从机模式地址 0 *注 1：主机模式无效。* *注 2：主机需要寻址该从机，需通过在 START 或 Repeated START 之后的第一个字节值地址信息与此地址相同。如果 AA 为 1，该从机响应主机，成为被寻址从机，否则主机广播寻址信息会被忽略。* *注 3：I2C_ADDR0[7:1] 不能写为全 0，因为 0x00 为广播方式寻址专用。*
0	GC	RW	广播地址应答使能 0：禁止 1：使能 *注 1：主机模式无效。* *注 2：使能 GC 后，如果 AA 置 1，则使能广播接收模式，若 AA 清 0，则忽略总线上的广播寻址信息。*

18.7.7 I2C_ADDR1 从机地址 1 寄存器

Address offset: 0x20 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:1	ADDR1	RW	从机模式地址 1 *注：主机模式无效。*
0	RFU	-	保留位，请保持默认值

18.7.8 I2C_ADDR2 从机地址 2 寄存器

Address offset: 0x24 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:1	ADDR2	RW	从机模式地址 2 *注：主机模式无效。*
0	RFU	-	保留位，请保持默认值

18.7.9 I2C_MATCH 从机地址匹配寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:3	RFU	-	保留位，请保持默认值
2	ADDR2	RO	I2C 从机模式地址 2 匹配标志位 0：从总线收到的设备地址与 ADDR2 不相同 1：从总线收到的设备地址与 ADDR2 相同
1	ADDR1	RO	I2C 从机模式地址 1 匹配标志位 0：从总线收到的设备地址与 ADDR1 不相同 1：从总线收到的设备地址与 ADDR1 相同
0	ADDR0	RO	I2C 从机模式地址 0 匹配标志位 0：从总线收到的设备地址与 ADDR0 不相同 1：从总线收到的设备地址与 ADDR0 相同

注：

地址匹配标志位在以下情况会清零：

- 模块复位时；
- START/STOP 发送时。

19 红外调制发送器（IR）

19.1 概述

CW32L011 内部集成红外调制发送器 (IR)，通过定时器、UART 及 IRSW 软控制位的配合使用，可方便实现各种标准的 PWM 或 PPM 编码方式，也可实现 UART 数据的红外调制发送。

19.2 主要特性

- 支持 IrDA 标准 1.0 的 SIR
- 最高数据速率 115.2kbps
- 可适应高低电平红外发射管

19.3 功能描述

实现红外调制发送器时，使用一个定时器通道产生一个固定频率的方波信号，另一个定时器或 UART 用以产生调制数据，配合 IRSW 软控制位进行‘与’或‘或’运算后，从 IR_OUT 引脚输出，具体 IR_OUT 引脚请参考数据手册引脚定义。

IR 红外调制控制寄存器 IRMOD_CR 的 MOD 位域，用于选择载波信号和数据信号的来源，以及与 IRSW 软控制位的‘与’‘或’操作，选择‘与’‘或’由用户的硬件红外发射管的驱动电平决定。通过 INV 位域可对 IR_OUT 输出进行反向控制。

载波信号频率用户可自行设置，最常见的载波频率是 38kHz。

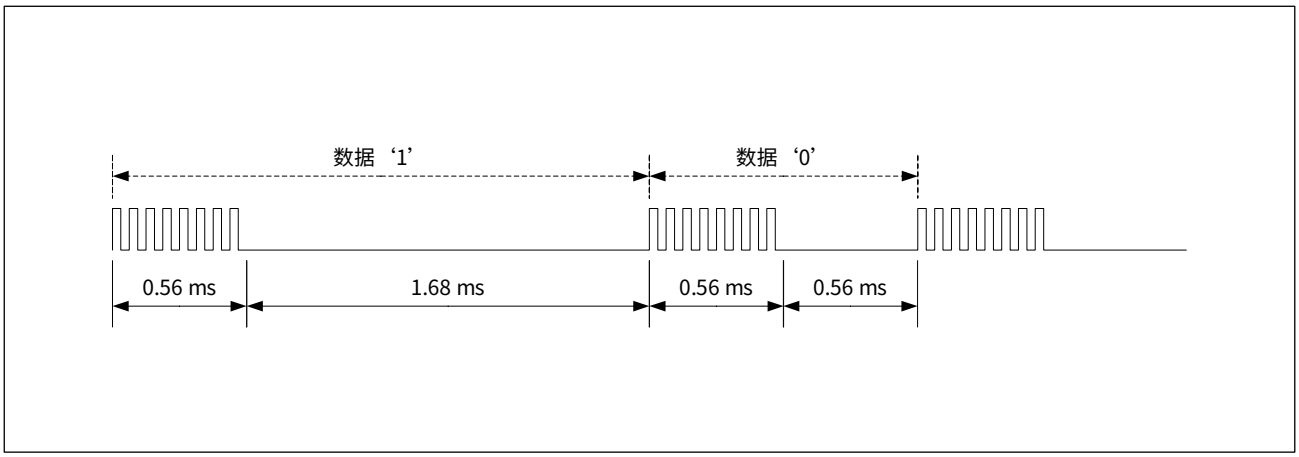
为降低发射功耗，载波的占空比一般设置为 20% 左右，产生载波的定时器可以使用 PWM 模式，请参见 13 通用定时器（GTIM）章节。为提高传输距离，可适当提高占空比。在 IrDA 标准 1.0 中，为实现快速的 IR 通信速率，脉冲的宽度规定为一个位周期的 3/16 或者为固定的 1.63μs，最小不能低于 1.41μs。

19.3.1 红外调制方式

常见 IR 红外编码方式有两种，脉宽调制 PWM 和脉位调制 PPM。

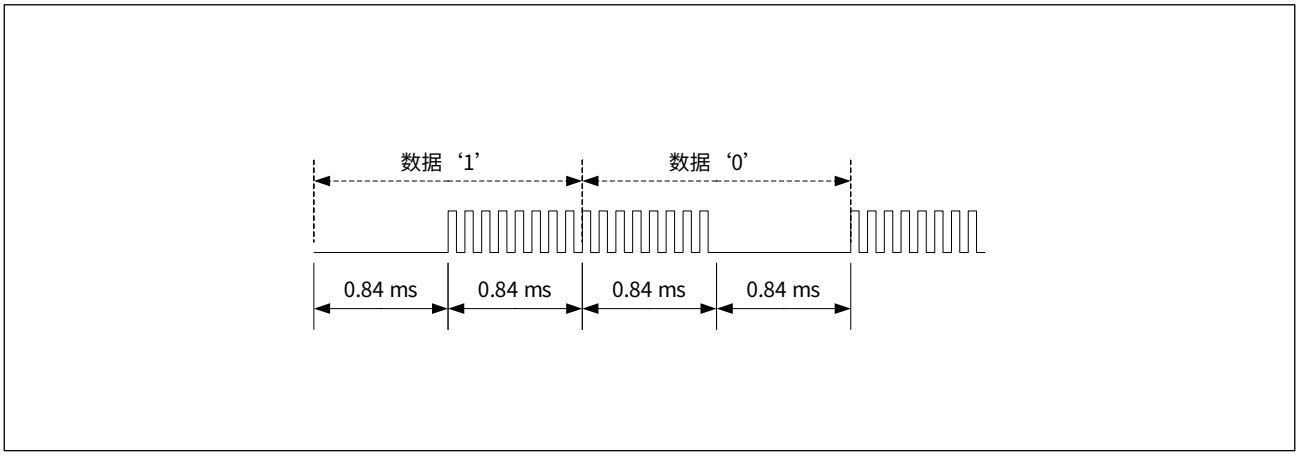
PWM 脉冲宽度调制，以发射红外载波的占空比代表‘0’和‘1’。比如 UPD6121，载波发射 0.56ms，不发射 0.56ms，表示‘0’；载波发射 0.56ms，不发射 1.68ms，表示‘1’。参考示意图如下图所示：

图 19-1 PWM 调制方式



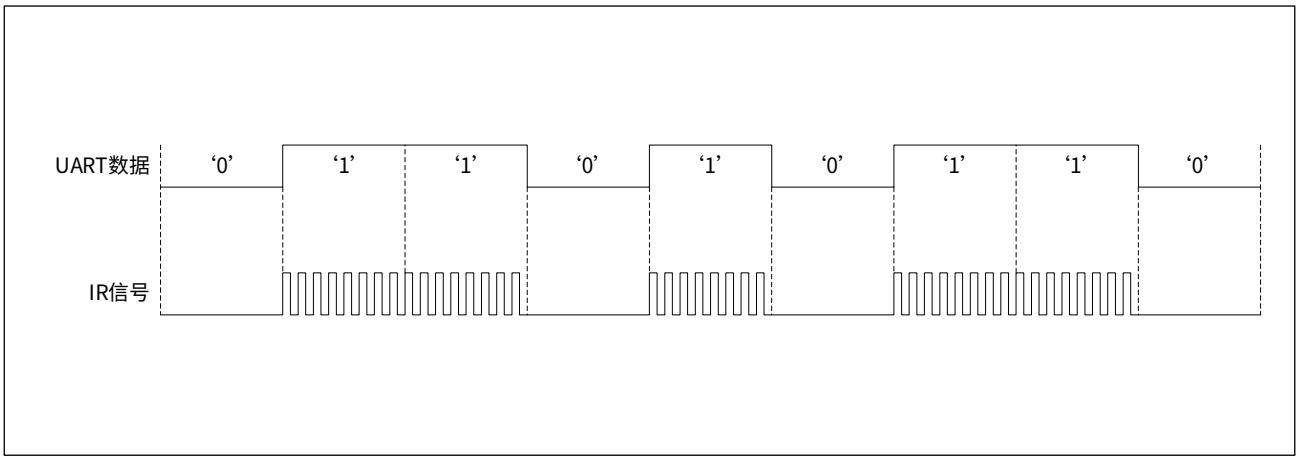
PPM 脉冲位置调制，以发射载波的位置表示‘0’和‘1’。从发射载波到不发射载波为‘0’，从不发射载波到发射载波为‘1’。比如 SAA3010，载波发射 0.84ms，不发射 0.84ms，表示‘0’；不发射 0.84ms，载波发射 0.84ms，表示‘1’。参考示意图如下图所示：

图 19-2　PPM 调制方式



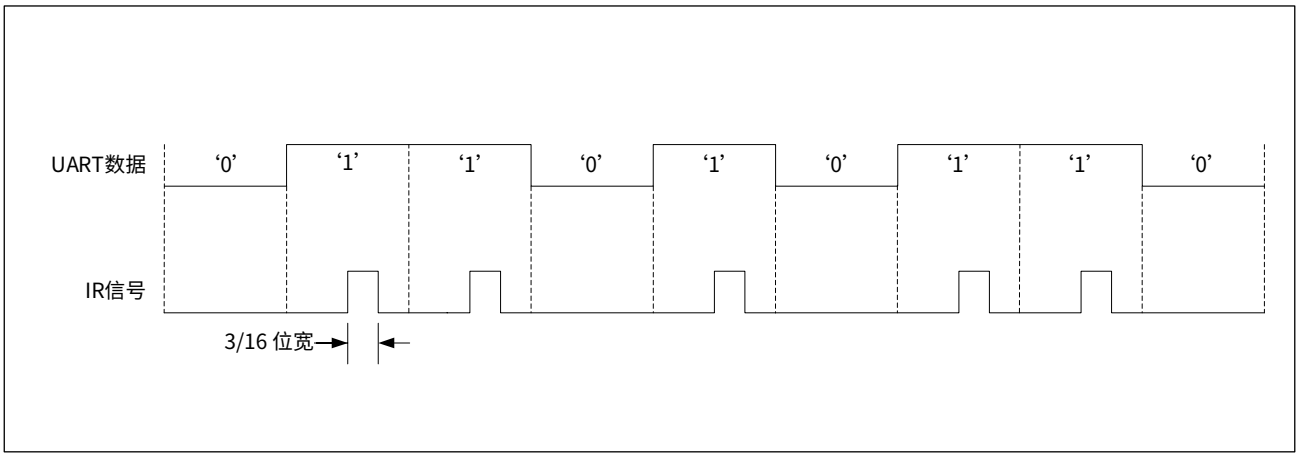
UART 调制是一种比较简单的红外调制方式，直接将串行数据与载波信号进行‘与’或‘或’运算。当 UART 串行波特率低于载波频率时，参考示意图如下图所示：

图 19-3　UART 调制方式



当 UART 串行波特率高于载波频率时，载波脉冲宽度应设置为位宽度的 3/16，载波频率需保持与串行频率一致。比如当 UART 波特率设置为 115.2Kbps 时，载波频率也设置为 115.2kHz，载波脉冲的宽度为 1.63μs（3/16×1/115200Hz = 1.63μs），参考示意图如下图所示：

图 19-4 高速 UART 调制方式



为方便调试，通常一个标准的数据帧还包括引导码、结束位、重发码等，请用户自行参阅相关数据手册。

19.3.2 红外调制初始化配置

一个完整的 IR 红外调制发送器的初始化过程要包括对定时器、UART、GPIO 的配置，参考步骤如下：

步骤 1：设置 SYSCTRL_APBEN1、SYSCTRL_APBEN2 寄存器，使能将要选择的定时器、UART 的外设时钟；

步骤 2：设置 SYSCTRL_AHBEN 寄存器中 IR_OUT 引脚对应的 GPIO 时钟使能允许位；

步骤 3：配置选择作为载波的定时器，设置载波频率、占空比（常用频率 38kHz，1/3 占空比），设置定时器工作模式（推荐为 PWM 输出）；

步骤 4：配置选择作为数据的定时器或 UART，设定位宽度，UART 工作模式（不需定义输出引脚）；

步骤 5：设定作为 IR_OUT 输出端口的 GPIOx_ANALOG 为数字模式，设置 GPIOx_DIR 为输出，同时设置适合的 GPIOx_OPENDRAIN、GPIOx_PUR；

步骤 6：启动作为载波和数据的定时器、UART；

步骤 7：设置 IR 红外调制控制寄存器 IRMOD_CR 的 MOD 位域，设定红外调制方式配置；

步骤 8：设定 IR_OUT 输出端口的 GPIOx_AFRH/GPIOx_AFRL 为 IR_OUT，启动 IR 复用功能。

19.3.3 红外接收

CW32L011 内部没有 IR 接收解调模块，在 IR 接收应用中，需要使用带有解调功能的一体化红外接收头，配合 GTIM 的捕捉功能（UART 方式可直接使用 RXD 引脚输入），可方便的实现 IR 接收功能。

19.4 IRMOD_CR 红外调制控制寄存器

Address: 0x4000 4000 + 0x80 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:6	RFU	-	保留位，请保持默认值
5	INV	RW	IR 输出反向控制 0：正向输出 1：反向输出
4	IRSW	RW	红外调制软控制位 功能详见 MOD 位域
3:0	MOD	RW	红外调制方式配置 0000：GTIM1_CH1 & GTIM2_CH1 & IRSW 0001：GTIM1_CH1 & GTIM2_CH2 & IRSW 0010：GTIM1_CH2 & GTIM2_CH1 & IRSW 0011：GTIM1_CH2 & GTIM2_CH2 & IRSW 0100：GTIM1_CH1 | GTIM2_CH1 | IRSW 0101：GTIM1_CH1 | GTIM2_CH2 | IRSW 0110：GTIM1_CH2 | GTIM2_CH1 | IRSW 0111：GTIM1_CH2 | GTIM2_CH2 | IRSW 1000：UART1_TXD & GTIM1_CH1 & IRSW 1001：UART1_TXD | GTIM1_CH1 | IRSW 1010：UART1_TXD & GTIM1_CH2 & IRSW 1011：UART1_TXD | GTIM1_CH2 | IRSW 1100：UART2_TXD & GTIM2_CH1 & IRSW 1101：UART2_TXD | GTIM2_CH1 | IRSW 1110：UART2_TXD & GTIM2_CH2 & IRSW 1111：UART2_TXD | GTIM2_CH2 | IRSW

20 模数转换器（ADC）

20.1 概述

CW32L011 内部集成一个 12 位精度、最高 1M SPS 转换速度的逐次逼近型模数转换器 (SAR ADC)，最多可将 16 路模拟信号转换为数字信号。现实世界中的绝大多数信号都是模拟量，如光、电、声、图像信号等，都要由 ADC 转换成数字信号，才能由 MCU 进行数字化处理。

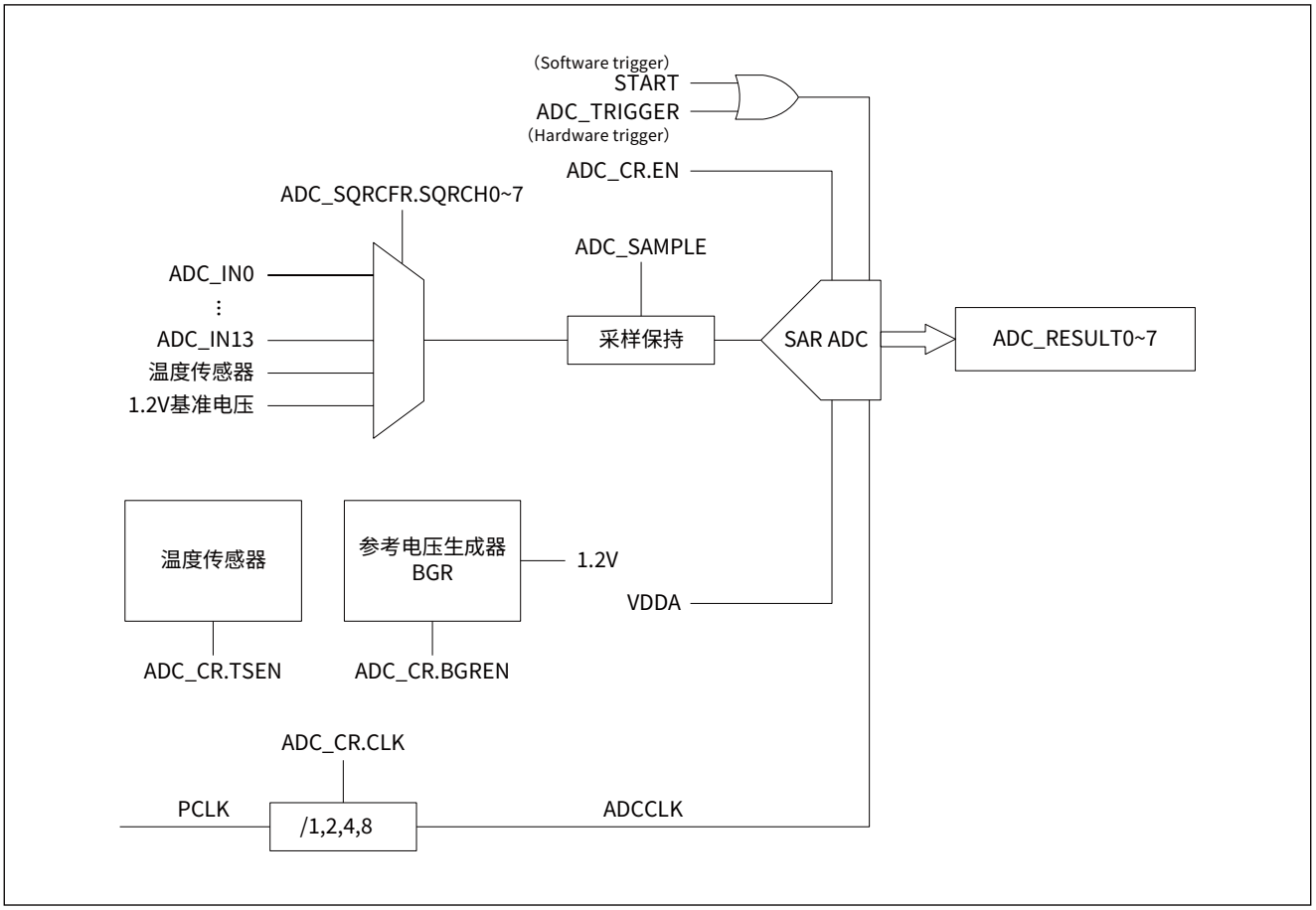
20.2 主要特性

- 12 位精度
- 各序列通道转换速度可单独配置，最高达 1M SPS
- 16 路输入转换通道
  - 14 路外部引脚输入
  - 内置温度传感器
  - 内置 BGR 1.2V 基准
- VDDA 电源电压作为参考电压源（Vref）
- 采样电压输入范围：0 ~ V_{ref}
- 支持序列通道转换模式
  - 支持单次和连续转换
  - 支持最多 8 个转换通道，每个通道可选 16 个转换源之一
- 支持输入通道电压阈值监测
- 支持片内外设自动触发 ADC 转换

20.3 功能框图

ADC 功能框图如下图所示：

图 20-1 ADC 功能框图

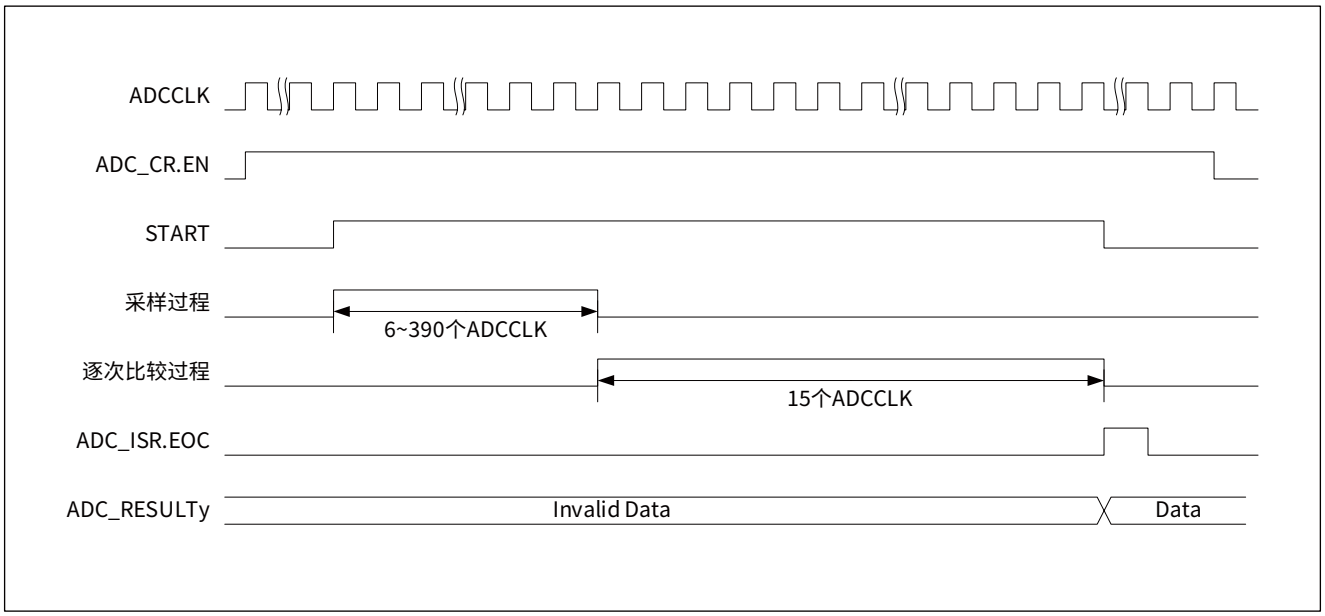


20.4 转换时序、转换速度、转换精度以及转换结果

20.4.1 转换时序

ADC 的转换时序如下图所示：

图 20-2 ADC 转换时序图



向 ADC 控制寄存器 ADC_CR 的 EN 位域写入 1，使能 ADC 模块。

ADC_CR.EN 由 0 变为 1 约 1μs 后模拟电路初始化完成，可以开始进行 ADC 转换。

向 ADC 启动寄存器 ADC_START 的 START 位域写入 1，启动 ADC 转换，转换完成后硬件自动清零。

ADC 工作时钟 ADCCLK，由系统时钟 PCLK 经预分频器分频得到，通过控制寄存器 ADC_CR 的 CLK 位域可选择 1、2、4、8 分频，如下表所示：

表 20-1 ADC 时钟配置表

ADC_CR.CLK	ADCCLK
000	PCLK
001	PCLK/2
010	PCLK/4
011	PCLK/8

一次完整的 ADC 转换需要 21 ~ 405 个 ADCCLK 时钟周期，包括采样阶段和逐次比较两个阶段：

1. 采样阶段：需要 6 ~ 390 个 ADCCLK 时钟周期。各序列转换通道的采样周期可单独配置，通过采样周期配置寄存器 ADC_SAMPLE 的 SQRCHy 位域进行设置，如下表所示：

表 20-2 ADC 各序列通道采样周期

ADC_SAMPLE.SQRCHy	采样周期（ADCCLK 个数）
0000	6
0001	7
0010	9
0011	12
0100	18
0101	24
0110	30
0111	42
1000	54
1001	70
1010	102
1011	134
1100	166
1101	198
1110	262
1111	390

通过外部 ADC_SAM 引脚，可输出 ADC 的采样时刻和采样周期，具体引脚请参阅《数据手册》引脚定义。

ADC 采样周期长度由用户对采样的速度要求和采样信号的电气特性决定，用户应选择合适的采样周期，以达到最佳的转换效果。

2. 逐次比较阶段：需要 15 个 ADCCLK 时钟周期。

ADC 每次转换完成之后，转换完成标志位 ADC_ISR.EOC 会被硬件置 1，ADC 转换结果存储在对应的序列转换结果寄存器 ADC_RESULTy（y=0、1、2、3、4、5、6、7）中，用户可通过设置 ADC_ICR.EOC 为 0 清除该标志位。

20.4.2 转换速度

ADC 转换速度与 VDDA 电源电压密切相关，各种条件下的最高转换速度，如下表所示：

表 20-3 ADC 转换速度与电压对照表

VDDA 电压	最高转换速度	最大 ADCCLK 频率
1.7V ~ 1.8V	200K SPS	6MHz
1.8V ~ 2.8V	500K SPS	12MHz
2.8V ~ 3.3V	1M SPS	24MHz
3.3V ~ 5.5V	1M SPS	48MHz

ADC 的转换速度的与工作时钟 ADCCLK 的对应关系如下：

$$ADC\ 转换速率 = f_{ADCCLK} / N_T$$

其中，f_{ADCCLK} 为 ADCCLK 时钟频率，N_T 为一次 ADC 转换所需要的 ADCCLK 个数。

20.4.3 转换结果

ADC 每次转换完成后，当前转换完成的序列通道 SQRCHy（y=0、1、2、3、4、5、6、7）的转换结果保存在对应的 ADC_RESULTy（y=0、1、2、3、4、5、6、7）寄存器中。

序列转换结果寄存器是 16 位宽，转换结果右对齐，有效值存储于 ADC_RESULTy 寄存器的低 12 位（位 11:0），高位（位 15:12）自动补 0。

20.5 工作模式

ADC 支持序列通道转换模式，可对最多 8 个序列通道（SQRCH0~SQRCH7）进行轮流转换，序列待转换的通道数量由 ADC_CR 寄存器的 ENS 位域配置。每个序列通道 SQRCHy（y=0、1、2、3、4、5、6、7）可选择 16 个转换源之一，具体由序列转换通道配置寄存器 ADC_SQRCFR 的 SQRCHy 位域进行设置，如下表所示：

表 20-4 序列通道待转换源

ADC_SQRCFR.SQRCHy	待转换源	GPIO
0000	ADC_IN0	PA00
0001	ADC_IN1	PA01
0010	ADC_IN2	PA02
0011	ADC_IN3	PA03
0100	ADC_IN4	PA04
0101	ADC_IN5	PA05
0110	ADC_IN6	PA06
0111	ADC_IN7	PA07
1000	ADC_IN8	PB00
1001	ADC_IN9	PB01
1010	ADC_IN10	PA08
1011	ADC_IN11	PA09
1100	ADC_IN12	PA10
1101	ADC_IN13	PA11
1110	TS 内置温度传感器	-
1111	1.2V 内核电压基准源	-

启动 ADC 转换，可通过向 ADC 启动寄存器 ADC_START 的 START 位域写 1；也可通过其他外设来触发，参见 20.6 外部触发启动源。

ADC 支持单次和连续模式，单次模式下（设置 ADC_CR.CONT 为 0），只执行一次 ENS 位域所配置的转换；连续模式下（设置 ADC_CR.CONT 为 1），持续进行 ENS 位域所配置的转换。

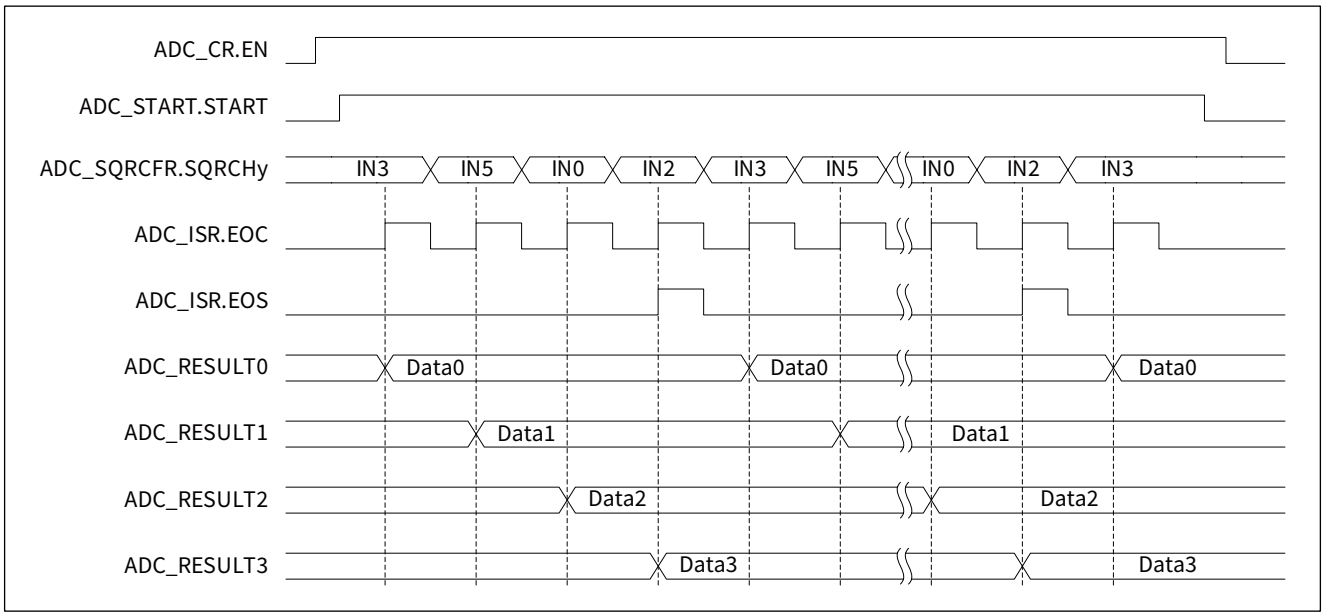
20.5.1 序列连续转换模式（CONT=1）

设置 ADC_CR.CONT 为 1 使能连续转换模式。此模式下，无论是通过软件 START 位启动 ADC，还是外部触发启动，一旦启动 ADC 将对 ENS 位域配置的序列转换通道持续进行转换，直到 ADC_START.START 位清 0，才会停止转换。

每次 ADC 转换完成后，ADC_ISR.EOC 标志位自动置 1，转换结果保存在与序列转换通道 SQRCH0 ~ SQRCH7 相同序号的序列转换结果寄存器 ADC_RESULT0 ~ ADC_RESULT7 中。当所选择的序列通道全部转换完成后，序列转换完成标志位 ADC_ISR.EOS 被置 1。用户应及时读取转换结果，以避免转换结果溢出。ADC_START.START 位清 0 可停止转换，同时复位序列的转换通道（即再次启动 ADC 时将重新从序列的通道 0 开始转换）。

以序列转换通道 SQRCH0 ~ SQRCH3 为例，序列连续转换模式的时序如下图所示：

图 20-3 序列连续转换时序图



通过 START 位启动 ADC 序列连续转换，参考操作流程如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ADC 为 1，使能 ADC 通道对应的 GPIO 时钟和 ADC 工作时钟；

步骤 2：设置 ADC 通道对应的 GPIO 引脚为模拟功能，具体寄存器配置请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 ADC_CR.EN 为 1，使能 ADC 模块；

步骤 4：延时约 1μs 等待模拟电路初始化完成；

步骤 5：设置 ADC_CR.CONT 为 1，选择连续转换模式；

步骤 6：配置 ADC_SAMPLE 寄存器相应位域及 ADC_CR.CLK，设置 ADC 的采样速度及时钟选择；

步骤 7：配置 ADC_CR.ENS，选择序列待转换的通道数量，如图 20-3 所示，设置 ADC_CR.ENS 为 3，序列转换通道为 SQRCH0~SQRCH3；

步骤 8：配置 ADC_SQRCFR.SQRCH0，选择序列通道 0 待转换源，如图 20-3 所示，设置 ADC_SQRCFR.SQRCH0 为 3，通道 0 的待转换源为 ADC_IN3；

步骤 9：配置 ADC_SQRCFR.SQRCH1，选择序列通道 1 待转换源，如图 20-3 所示，设置 ADC_SQRCFR.SQRCH1 为 5，通道 1 的待转换源为 ADC_IN5；

步骤 10：配置 ADC_SQRCFR.SQRCH2，选择序列通道 2 待转换源，如图 20-3 所示，设置 ADC_SQRCFR.SQRCH2 为 0，通道 2 的待转换源为 ADC_IN0；

步骤 11：配置 ADC_SQRCFR.SQRCH3，选择序列通道 3 待转换源，如图 20-3 所示，设置 ADC_SQRCFR.SQRCH3 为 2，通道 3 的待转换源为 ADC_IN2；

步骤 12：设置 ADC_ICR 为 0，清除 ADC 中断标志；

步骤 13：设置 ADC_START.START 为 1，启动 ADC 转换；

步骤 14：等待 ADC_ISR.EOS 变为 1，依次读取 ADC_RESULT0~ADC_RESULT3 寄存器，以获取对应通道的 ADC 转换结果。当 ADC_ISR.EOS 变为 1 时，表示一次 4 个通道的序列转换完成；

步骤 15：设置 ADC_START.START 为 0，停止 ADC 转换；

步骤 16：如需对其它通道进行转换，重复执行步骤 6 ~ 步骤 15；

步骤 17：设置 ADC_CR.EN 为 0，关闭 ADC 模块。

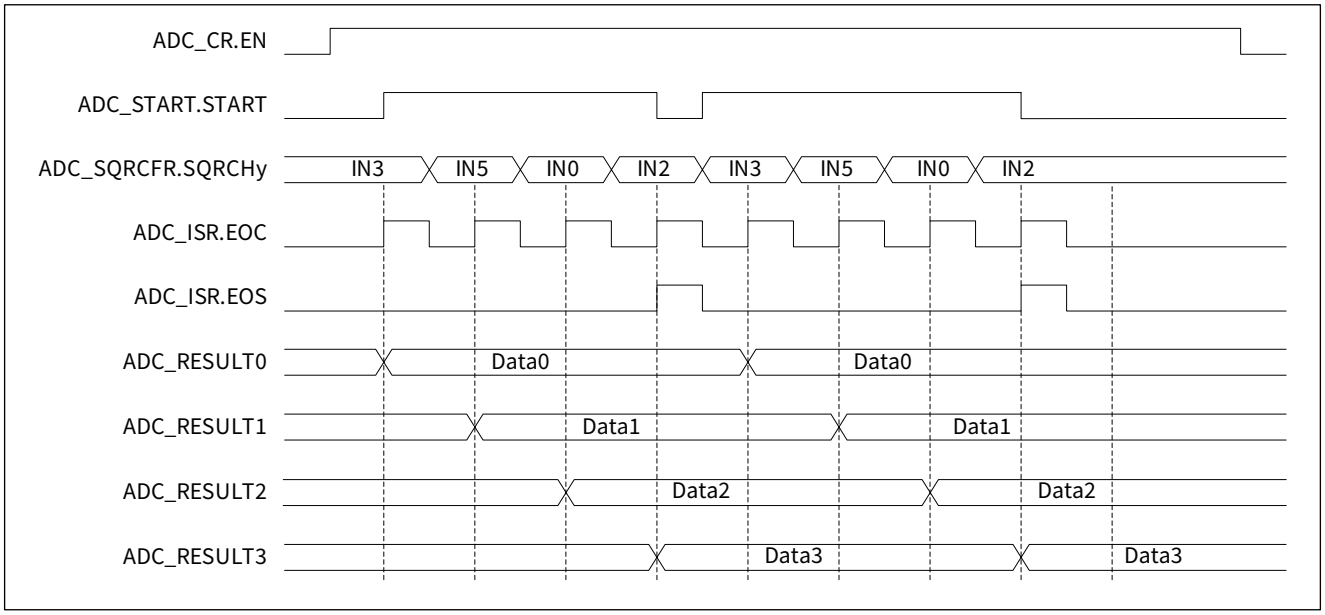
20.5.2 序列扫描转换模式（CONT=0）

设置 ADC_CR.CONT 为 0 使能单次转换模式。此模式下，无论是通过软件 START 位启动 ADC，还是外部触发启动，启动一次 ADC，将只对 ENS 位域配置的序列转换通道全部进行一次转换。

每次 ADC 转换完成后，ADC_ISR.EOC 标志位自动置 1，转换结果保存在与序列转换通道 SQRCH0 ~ SQRCH7 相同序号的序列转换结果寄存器 ADC_RESULT0~ADC_RESULT7 中。当所选择的序列通道全部转换完成后，序列转换完成标志位 ADC_ISR.EOS 被置 1，ADC_START.START 位自动清 0，ADC 停止转换。

以序列转换通道 SQRCH0 ~ SQRCH3 为例，序列扫描转换模式的时序如下图所示：

图 20-4 序列扫描转换时序图



通过 START 位启动 ADC 序列扫描转换，参考操作流程如下：

步骤 1：设置 SYSCTRL_AHBEN.GPIOx 为 1，SYSCTRL_APBEN1.ADC 为 1，使能 ADC 通道对应的 GPIO 时钟和 ADC 工作时钟；

步骤 2：设置 ADC 通道对应的 GPIO 引脚为模拟功能，具体寄存器配置请参见 8 通用输入输出端口（GPIO）章节；

步骤 3：设置 ADC_CR.EN 为 1，使能 ADC 模块；

步骤 4：延时约 1μs 等待模拟电路初始化完成；

步骤 5：设置 ADC_CR.CONT 为 0，选择单次转换模式；

步骤 6：配置 ADC_SAMPLE 寄存器相应位域及 ADC_CR.CLK，设置 ADC 的采样速度及时钟选择；

步骤 7：配置 ADC_CR.ENS，选择序列待转换的通道数量，如图 20-4 所示，设置 ADC_CR.ENS 为 3，序列转换通道为 SQRCH0 ~ SQRCH3；

步骤 8：配置 ADC_SQRCFR.SQRCH0，选择序列通道 0 待转换源，如图 20-4 所示，设置 ADC_SQRCFR.SQRCH0 为 3，通道 0 的待转换源为 ADC_IN3；

步骤 9：配置 ADC_SQRCFR.SQRCH1，选择序列通道 1 待转换源，如图 20-4 所示，设置 ADC_SQRCFR.SQRCH1 为 5，通道 1 的待转换源为 ADC_IN5；

步骤 10：配置 ADC_SQRCFR.SQRCH2，选择序列通道 2 待转换源，如图 20-4 所示，设置 ADC_SQRCFR.SQRCH2 为 0，通道 2 的待转换源为 ADC_IN0；

步骤 11：配置 ADC_SQRCFR.SQRCH3，选择序列通道 3 待转换源，如图 20-4 所示，设置 ADC_SQRCFR.SQRCH3 为 2，通道 3 的待转换源为 ADC_IN2；

步骤 12：设置 ADC_ICR 为 0，清除 ADC 中断标志；

步骤 13：设置 ADC_START.START 为 1，启动 ADC 转换；

步骤 14：等待 ADC_ISR.EOS 变为 1，依次读取 ADC_RESULT0 ~ ADC_RESULT3 寄存器，以获取对应通道的 ADC 转换结果。当 ADC_ISR.EOS 变为 1 时，表示一次 4 个通道的序列转换完成，ADC_START.START 位自动清 0，ADC 转换停止；

步骤 15：如需对其它通道进行转换，重复执行步骤 6~ 步骤 14；

步骤 16：设置 ADC_CR.EN 为 0，关闭 ADC 模块。

20.6 外部触发启动源

ADC 转换既可以通过软件启动（即设置 ADC_START.START 为 1），也可通过外部触发启动，触发源由外部触发启动寄存器 ADC_TRIGGER 选择，有 25 种触发 ADC 方式，详见相关寄存器描述。

20.7 模拟看门狗

模拟看门狗功能常用于对模拟量的自动监测，即将 ADC 转换结果与用户设定的阈值进行比较，并可产生中断。

模拟看门狗可通过编程 ADC_AWDCR 寄存器的 INy（y=0 ~ 15）位域来保护多条已选的通道，即 INy 位域置 1 时，相应通道的看门狗将使能。

模拟看门狗支持上阈值和下阈值比较，阈值上限和阈值下限分别通过 ADC_AWDTR 寄存器的 VTH 和 VTL 位域来设置。当 ADC 转换的模拟电压低于阈值下限或高于阈值上限时，模拟看门狗相应标志位会置 1，如果设置了中断使能寄存器 ADC_IER 的相应位域（AWDH、AWDL），将产生中断请求。上阈值和下阈值比较如下所示：

上阈值比较：当转换结果位于 (AWDTR.VTH，4095] 区间内时，ADC_ISR.AWDH 标志位置 1。

下阈值比较：当转换结果位于 [0，AWDTR.VTL）区间内时，ADC_ISR.AWDL 标志位置 1。

模拟看门狗输出的内部硬件信号 ADC_AWD 还可直接连接到 GTIM 和 ATIM 的 ETR 输入，或作为定时器参考信号 OCREF 的清除源，请参考相应定时器章节。

20.8 温度传感器

CW32L011 内置温度传感器模块，传感器的输出电压随温度变化，设置 ADC 模块的采样通道为内部温度传感器，通过 ADC 测量结果可计算得到当前的环境温度。

温度传感器默认处于关闭状态，通过设置控制寄存器 ADC_CR 的 TSEN 位域为 1，使能温度传感器，使能约 30μs 后才可以转换 TS 通道。

环境温度计算公式如下：

$$环境温度 = T_0 \times 0.5 + 0.095765 \times (V_{ref} \times AdcValue - 4.095 \times Trim)$$

其中：

V_{ref} 是当前 ADC 模块的参考电压，取值为 V_{DDA}，单位是 V。

T_0 是 8 位的初始校准温度值，记录在芯片的 FLASH 存储器中，其地址是 0x0010 07CD，单位是 0.5 摄氏度，读取出来的值需要除以 2，才是实际的温度。

AdcValue 是 ADC 模块测量温度传感器输出电压的 ADC 转换结果，取值范围为 0 ~ 4095。

Trim 是 16 位的校准值，计算时需要从芯片的 FLASH 存储器中读出，其存放地址如下表所示：

表 20-5 ADC 校准值地址

ADC 参考电压	校准值存放地址	校准值精度
V_{DDA}	0x0010 07CE - 0x0010 07CF	±3°C

计算示例如下：

条件：V_{ref} = 3.3、AdcValue = 0x3CD、Trim = 0x311、T_0 = 0x32

温度：$0x32 \times 0.5 + 0.095765 \times (3.3 \times 0x3CD - 4.095 \times 0x311) = 24.65°C$

通过 ADC 测量环境温度的参考操作流程如下：

步骤 1：设置 SYSCTRL_APBEN1.ADC 为 1，使能 ADC 配置时钟及工作时钟；

步骤 2：设置 ADC_CR.EN 为 1，使能 ADC 模块；

步骤 3：延时约 1μs 等待模拟电路初始化完成；

步骤 4：设置 ADC_CR.ENS 为 0，转换 SQRCH0；

步骤 5：设置 ADC_SQRCFR.SQRCH0 为 0xE，选择序列通道 0 待转换源为内置温度传感器；

步骤 6：配置 ADC_SAMPLE.SQRCH0 及 ADC_CR.CLK，设置 ADC 的转换速度；

步骤 7：设置 ADC_CR.TSEN 为 1，使能温度传感器，并等待约 30μs 使其稳定；

步骤 8：设置 ADC_ICR.EOC 为 0，清除 ADC_ISR.EOC 标志；

步骤 9：设置 ADC_START.START 为 1，启动 ADC 转换；

步骤 10：等待 ADC_ISR.EOC 变为 1，读取 ADC_RESULT0 寄存器，以获取 ADC 转换结果；

步骤 11：设置 ADC_CR.EN 为 0，关闭 ADC 模块；

步骤 12：读取 T_0 及 Trim，根据公式计算出当前的环境温度。

20.9 ADC 中断

ADC 中断请求，如下表所示：

表 20-6 ADC 中断源与中断标志

中断源	中断标志	中断使能	标志清除
转换结果 >ADC_AWDTR.VTH	ADC_ISR.AWDH	ADC_IER.AWDH 置 1	ADC_ICR.AWDH 清 0
转换结果 <ADC_AWDTR.VTL	ADC_ISR.AWDL	ADC_IER.AWDL 置 1	ADC_ICR.AWDL 清 0
ADC 序列转换完成	ADC_ISR.EOS	ADC_IER.EOS 置 1	ADC_ICR.EOS 清 0
ADC 转换完成	ADC_ISR.EOC	ADC_IER.EOC 置 1	ADC_ICR.EOC 清 0

20.10 寄存器列表

ADC 基地址：ADC_BASE = 0x4000 0000

表 20-7 ADC 寄存器列表

寄存器名称	寄存器地址	寄存器描述
ADC_CR	ADC_BASE + 0x00	ADC 控制寄存器
ADC_START	ADC_BASE + 0x08	ADC 启动寄存器
ADC_SAMPLE	ADC_BASE + 0x28	ADC 采样周期配置寄存器
ADC_SQRCFR	ADC_BASE + 0x2C	ADC 序列转换通道配置寄存器
ADC_AWDTR	ADC_BASE + 0x10	ADC 模拟看门狗阈值寄存器
ADC_AWDCR	ADC_BASE + 0x20	ADC 模拟看门狗配置寄存器
ADC_TRIGGER	ADC_BASE + 0x18	ADC 外部触发启动寄存器
ADC_IER	ADC_BASE + 0x74	ADC 中断使能寄存器
ADC_ISR	ADC_BASE + 0x7C	ADC 中断标志寄存器
ADC_ICR	ADC_BASE + 0x78	ADC 中断标志清除寄存器
ADC_RESULT0	ADC_BASE + 0x40	ADC 序列转换结果寄存器 0
ADC_RESULT1	ADC_BASE + 0x44	ADC 序列转换结果寄存器 1
ADC_RESULT2	ADC_BASE + 0x48	ADC 序列转换结果寄存器 2
ADC_RESULT3	ADC_BASE + 0x4C	ADC 序列转换结果寄存器 3
ADC_RESULT4	ADC_BASE + 0x50	ADC 序列转换结果寄存器 4
ADC_RESULT5	ADC_BASE + 0x54	ADC 序列转换结果寄存器 5
ADC_RESULT6	ADC_BASE + 0x58	ADC 序列转换结果寄存器 6
ADC_RESULT7	ADC_BASE + 0x5C	ADC 序列转换结果寄存器 7

20.11 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

20.11.1 ADC_CR 控制寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8:6	ENS	RW	序列待转换的通道数量配置 000：转换 SQRCH0 001：转换 SQRCH0 – SQRCH1 010：转换 SQRCH0 – SQRCH2 011：转换 SQRCH0 – SQRCH3 100：转换 SQRCH0 – SQRCH4 101：转换 SQRCH0 – SQRCH5 110：转换 SQRCH0 – SQRCH6 111：转换 SQRCH0 – SQRCH7
5:4	CLK	RW	ADC 工作时钟 ADCCLK 来源配置 00：ADCCLK = PCLK / 1 01：ADCCLK = PCLK / 2 10：ADCCLK = PCLK / 4 11：ADCCLK = PCLK / 8
3	CONT	RW	持续转换使能配置 0：单次模式（执行一次 ENS 所配置的转换） 1：持续模式（持续进行 ENS 所配置的转换）
2	TSEN	RW	内置温度传感器使能控制 0：禁止内置温度传感器 1：使能内置温度传感器 *注：使能后约 30μs 才可以转换 TS 通道。*
1	BGREN	RW	内置 BGR 基准使能控制 0：禁止内置 BGR 基准 1：使能内置 BGR 基准 *注：使能后约 30μs 才可以转换 BGR1.2V 通道。*
0	EN	RW	ADC 使能控制 0：禁止 ADC 1：使能 ADC *注：使能后约 1μs 才可以转换外部通道。*

20.11.2 ADC_START 启动寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:1	RFU	-	保留位，请保持默认值
0	START	RW	ADC 启动转换控制 0：停止 ADC 转换 1：启动 ADC 转换 *注：通过软件写 0 可停止转换并复位序列的转换通道。*

20.11.3 ADC_SAMPLE 采样周期配置寄存器

Address offset: 0x28 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28	SQRCH7	RW	ADC 序列通道 7 采样时间配置 参见 SQRCH0 位域
27:24	SQRCH6	RW	ADC 序列通道 6 采样时间配置 参见 SQRCH0 位域
23:20	SQRCH5	RW	ADC 序列通道 5 采样时间配置 参见 SQRCH0 位域
19:16	SQRCH4	RW	ADC 序列通道 4 采样时间配置 参见 SQRCH0 位域
15:12	SQRCH3	RW	ADC 序列通道 3 采样时间配置 参见 SQRCH0 位域
11:8	SQRCH2	RW	ADC 序列通道 2 采样时间配置 参见 SQRCH0 位域
7:4	SQRCH1	RW	ADC 序列通道 1 采样时间配置 参见 SQRCH0 位域
3:0	SQRCH0	RW	ADC 序列通道 0 采样时间配置 0000：6 个 ADCCLK 时钟周期　1000：54 个 ADCCLK 时钟周期 0001：7 个 ADCCLK 时钟周期　1001：70 个 ADCCLK 时钟周期 0010：9 个 ADCCLK 时钟周期　1010：102 个 ADCCLK 时钟周期 0011：12 个 ADCCLK 时钟周期　1011：134 个 ADCCLK 时钟周期 0100：18 个 ADCCLK 时钟周期　1100：166 个 ADCCLK 时钟周期 0101：24 个 ADCCLK 时钟周期　1101：198 个 ADCCLK 时钟周期 0110：30 个 ADCCLK 时钟周期　1110：262 个 ADCCLK 时钟周期 0111：42 个 ADCCLK 时钟周期　1111：390 个 ADCCLK 时钟周期 *注 1：转换时间 = 采样时间 + 逐次比较时间 (固定 15 个 ADCCLK)。* *注 2：当被转换通道为 BGR 或 TS 时，采样持续时间至少需要 40μs。*

20.11.4 ADC_SQRCFR 序列转换通道配置寄存器

Address offset: 0x2C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:28	SQRCH7	RW	序列通道 7 待转换源配置 详见 SQRCH0
27:24	SQRCH6	RW	序列通道 6 待转换源配置 详见 SQRCH0
23:20	SQRCH5	RW	序列通道 5 待转换源配置 详见 SQRCH0
19:16	SQRCH4	RW	序列通道 4 待转换源配置 详见 SQRCH0
15:12	SQRCH3	RW	序列通道 3 待转换源配置 详见 SQRCH0
11:8	SQRCH2	RW	序列通道 2 待转换源配置 详见 SQRCH0
7:4	SQRCH1	RW	序列通道 1 待转换源配置 详见 SQRCH0
3:0	SQRCH0	RW	序列通道 0 待转换源配置 0000：ADC_IN0　　1000：ADC_IN8 0001：ADC_IN1　　1001：ADC_IN9 0010：ADC_IN2　　1010：ADC_IN10 0011：ADC_IN3　　1011：ADC_IN11 0100：ADC_IN4　　1100：ADC_IN12 0101：ADC_IN5　　1101：ADC_IN13 0110：ADC_IN6　　1110：内置温度传感器 0111：ADC_IN7　　1111：1.2V 内核电压基准源 *注：当选择 1110~1111 通道时，采样持续时间至少需要 40μs。*

20.11.5 ADC_AWDTR 模拟看门狗阈值寄存器

Address offset: 0x10 Reset value: 0x0FFF 0000

位域	名称	权限	功能描述
31:28	RFU	-	保留位，请保持默认值
27:16	VTH	RW	模拟看门狗阈值上限值
15:12	RFU	-	保留位，请保持默认值
11:0	VTL	RW	模拟看门狗阈值下限值

20.11.6 ADC_AWDCR 模拟看门狗配置寄存器

Address offset: 0x20 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	IN15	RW	模拟看门狗监测 ADC_IN15 使能控制（即 1.2V 内核电压基准源） 0：禁止 1：使能
14	IN14	RW	模拟看门狗监测 ADC_IN14 使能控制（即内置温度传感器输出信号） 0：禁止 1：使能
13	IN13	RW	模拟看门狗监测 ADC_IN13 使能控制 0：禁止 1：使能
12	IN12	RW	模拟看门狗监测 ADC_IN12 使能控制 0：禁止 1：使能
11	IN11	RW	模拟看门狗监测 ADC_IN11 使能控制 0：禁止 1：使能
10	IN10	RW	模拟看门狗监测 ADC_IN10 使能控制 0：禁止 1：使能
9	IN9	RW	模拟看门狗监测 ADC_IN9 使能控制 0：禁止 1：使能
8	IN8	RW	模拟看门狗监测 ADC_IN8 使能控制 0：禁止 1：使能
7	IN7	RW	模拟看门狗监测 ADC_IN7 使能控制 0：禁止 1：使能
6	IN6	RW	模拟看门狗监测 ADC_IN6 使能控制 0：禁止 1：使能
5	IN5	RW	模拟看门狗监测 ADC_IN5 使能控制 0：禁止 1：使能

位域	名称	权限	功能描述
4	IN4	RW	模拟看门狗监测 ADC_IN4 使能控制 0：禁止 1：使能
3	IN3	RW	模拟看门狗监测 ADC_IN3 使能控制 0：禁止 1：使能
2	IN2	RW	模拟看门狗监测 ADC_IN2 使能控制 0：禁止 1：使能
1	IN1	RW	模拟看门狗监测 ADC_IN1 使能控制 0：禁止 1：使能
0	IN0	RW	模拟看门狗监测 ADC_IN0 使能控制 0：禁止 1：使能

20.11.7 ADC_TRIGGER 外部触发启动寄存器

Address offset: 0x18 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:25	RFU	-	保留位，请保持默认值
24	GTIM2CC4	RW	GTIM2 输出比较 CC4 信号触发 ADC 启动 0：禁止 1：使能
23	GTIM2CC3	RW	GTIM2 输出比较 CC3 信号触发 ADC 启动 0：禁止 1：使能
22	GTIM2CC2	RW	GTIM2 输出比较 CC2 信号触发 ADC 启动 0：禁止 1：使能
21	GTIM2CC1	RW	GTIM2 输出比较 CC1 信号触发 ADC 启动 0：禁止 1：使能
20	GTIM2TRGO	RW	GTIM2 TRGO 信号触发 ADC 启动 0：禁止 1：使能
19	UART3	RW	UART3 触发 ADC 启动 0：禁止 1：使能

位域	名称	权限	功能描述
18	UART2	RW	UART2 触发 ADC 启动 0：禁止 1：使能
17	UART1	RW	UART1 触发 ADC 启动 0：禁止 1：使能
16	SPI	RW	SPI 触发 ADC 启动 0：禁止 1：使能
15	BTIM3TRGO	RW	BTIM3 TRGO 信号触发 ADC 启动 0：禁止 1：使能
14	BTIM2TRGO	RW	BTIM2 TRGO 信号触发 ADC 启动 0：禁止 1：使能
13	BTIM1TRGO	RW	BTIM1 TRGO 信号触发 ADC 启动 0：禁止 1：使能
12	GTIM1CC4	RW	GTIM1 输出比较 CC4 信号触发 ADC 启动 0：禁止 1：使能
11	GTIM1CC3	RW	GTIM1 输出比较 CC3 信号触发 ADC 启动 0：禁止 1：使能
10	GTIM1CC2	RW	GTIM1 输出比较 CC2 信号触发 ADC 启动 0：禁止 1：使能
9	GTIM1CC1	RW	GTIM1 输出比较 CC1 信号触发 ADC 启动 0：禁止 1：使能
8	GTIM1TRGO	RW	GTIM1 TRGO 信号触发 ADC 启动 0：禁止 1：使能
7	ATIMCC6	RW	ATIM 输出比较 CC6 信号触发 ADC 启动 0：禁止 1：使能
6	ATIMCC5	RW	ATIM 输出比较 CC5 信号触发 ADC 启动 0：禁止 1：使能

位域	名称	权限	功能描述
5	ATIMCC4	RW	ATIM 输出比较 CC4 信号触发 ADC 启动 0：禁止 1：使能
4	ATIMCC3	RW	ATIM 输出比较 CC3 信号触发 ADC 启动 0：禁止 1：使能
3	ATIMCC2	RW	ATIM 输出比较 CC2 信号触发 ADC 启动 0：禁止 1：使能
2	ATIMCC1	RW	ATIM 输出比较 CC1 信号触发 ADC 启动 0：禁止 1：使能
1	ATIMTRGO2	RW	ATIM TRGO2 信号触发 ADC 启动 0：禁止 1：使能
0	ATIMTRGO	RW	ATIM TRGO 信号触发 ADC 启动 0：禁止 1：使能

20.11.8 ADC_IER 中断使能寄存器

Address offset: 0x74 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3	AWDH	RW	模拟看门狗阈值上限中断使能控制 0：禁止 1：使能
2	AWDL	RW	模拟看门狗阈值下限中断使能控制 0：禁止 1：使能
1	EOS	RW	序列转换完成中断使能控制 0：禁止 1：使能
0	EOC	RW	单次转换完成中断使能控制 0：禁止 1：使能

20.11.9 ADC_ISR 中断标志寄存器

Address offset: 0x7C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3	AWDH	RO	模拟看门狗阈值上限标志 0：转换结果位于 [0, AWDTR.VTH] 区间 1：转换结果位于 (AWDTR.VTH, 4095] 区间
2	AWDL	RO	模拟看门狗阈值下限标志 0：转换结果位于 [AWDTR.VTL, 4095] 区间 1：转换结果位于 [0, AWDTR.VTL) 区间
1	EOS	RO	序列转换完成标志 0：序列转换未完成 1：序列转换已完成
0	EOC	RO	转换完成标志 0：一次 ADC 转换未完成 1：一次 ADC 转换已完成

20.11.10 ADC_ICR 中断标志清除寄存器

Address offset: 0x78 Reset value: 0x0000 000F

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3	AWDH	R1W0	模拟看门狗阈值上限标志清 0 控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
2	AWDL	R1W0	模拟看门狗阈值下限标志清 0 控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
1	EOS	R1W0	序列转换完成标志清 0 控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能
0	EOC	R1W0	转换完成标志清 0 控制 W0：清除 ISR 寄存器中的相应标志 W1：无功能

20.11.11 ADC_RESULT0 序列转换结果寄存器 0

Address offset: 0x40 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 0

20.11.12 ADC_RESULT1 序列转换结果寄存器 1

Address offset: 0x44 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 1

20.11.13 ADC_RESULT2 序列转换结果寄存器 2

Address offset: 0x48 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 2

20.11.14 ADC_RESULT3 序列转换结果寄存器 3

Address offset: 0x4C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 3

20.11.15 ADC_RESULT4 序列转换结果寄存器 4

Address offset: 0x50 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 4

20.11.16 ADC_RESULT5 序列转换结果寄存器 5

Address offset: 0x54 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 5

20.11.17 ADC_RESULT6 序列转换结果寄存器 6

Address offset: 0x58 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 6

20.11.18 ADC_RESULT7 序列转换结果寄存器 7

Address offset: 0x5C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0	RESULT	RO	ADC 序列转换结果 7

21 模拟电压比较器（VC）

21.1 概述

CW32L011 内部集成 2 个模拟电压比较器（VC），用于比较两路模拟输入电压，并将比较结果从引脚输出。电压比较器的正端输入支持 4 路外部模拟输入，负端既支持 2 路外部模拟输入，又支持内部 1.2V 电压基准和内部电阻分压器输出电压。比较结果输出具有滤波功能、迟滞窗口功能，以及极性选择。支持比较中断，可用于低功耗模式下唤醒 MCU。

21.2 主要特性

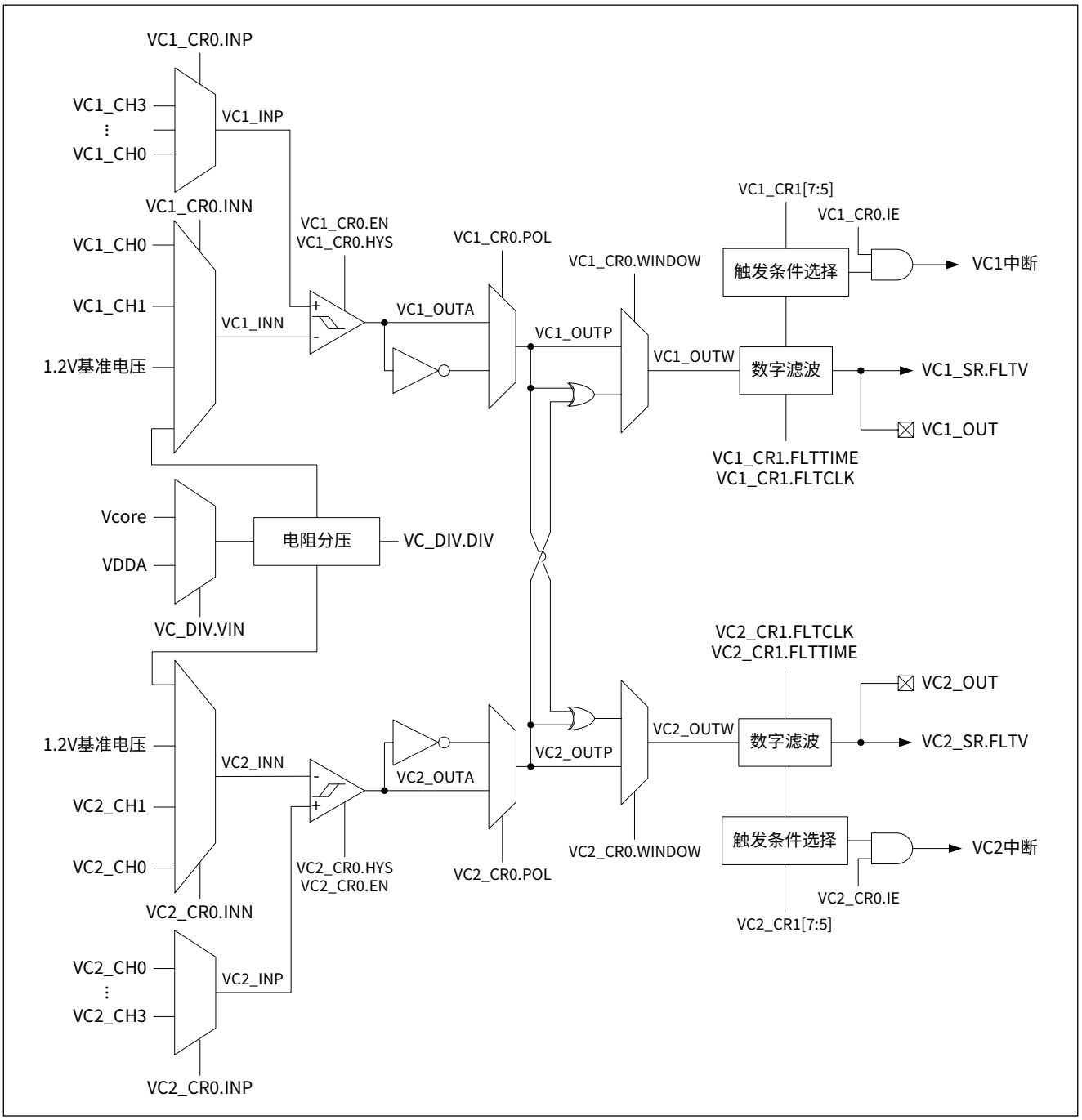
- 双路的模拟电压比较器 VC1、VC2
- 内部 8 阶电阻分压器
- 4 路外部模拟信号输入
- 2 路片内模拟输入信号
  - 内置电阻分压器输出电压
  - 内置 1.2V 基准电压
- 可选择输出极性
- 支持迟滞窗口比较功能
- 可编程的滤波器和滤波时间
- 3 种中断触发方式，可组合使用
  - 高电平触发
  - 上升沿触发
  - 下降沿触发
- 支持低功耗模式下运行，中断唤醒 MCU

21.3 功能描述

21.3.1 功能框图

模拟电压比较器 (VC) 的功能框图如下图所示：

图 21-1 VC 功能框图



VC1、VC2 的正负端输入选择，由控制寄存器 VCx_CR0 的 INP、INN 位域选择，如下表所示：

表 21-1 VC 正负端输入信号配置

VCx_CR0.INP	VCx 正端输入信号	VCx_CR0.INN	VCx 负端输入信号
00	VCx_CH0	00	VCx_CH0
01	VCx_CH1	01	VCx_CH1
10	VCx_CH2	10	内置 1.2V 基准电压
11	VCx_CH3	11	内置电阻分压器输出电压

内置的电阻分压配置由 VC_DIV 寄存器 [1] 的三个位域控制：

- EN 位域，使能电阻分压器。
- VIN 位域，选择分压器的输入电压。
  - VIN 为 0，选择 V_{DDA}。
  - VIN 为 1，选择 V_{core}，约 1.6V。
- DIV 位域，分压比例系数，有效范围为 0 ~ 7，选择分压器的输出电压，计算公式如下：

$$分压器输出电压 = 输入电压 \times (1 + DIV) / 8$$

注 1：

VC1 和 VC2 共用电阻分压器电路。

在选择内置 1.2V 基准电压作为比较器的负端输入时，需要使能芯片内部的 BGR 模块，内部 BGR 的启动时间大约为 30μs，VC 电压比较器需要等待内部 BGR 稳定后才能正常工作。

21.3.2 输入输出引脚

模拟电压比较器支持 4 路外部模拟信号输入，用户必须将对应 GPIO 端口配置为模拟功能（GPIOx_ANALOG.PINy = 1）。

模拟电压比较器支持将比较结果从引脚输出，用户必须将对应 GPIO 端口配置为数字输出，同时选择功能复用。

VC1、VC2 支持的输入输出引脚如下表所示：

表 21-2 VC 输入输出引脚配置

VC1 输入输出	GPIO	配置	VC2 输入输出	GPIO	配置
VC1_CH0	PA00	模拟	VC2_CH0	PA01	模拟
VC1_CH1	PA02	模拟	VC2_CH1	PA03	模拟
VC1_CH2	PA04	模拟	VC2_CH2	PA05	模拟
VC1_CH3	PA06	模拟	VC2_CH3	PA07	模拟
VC1_OUT	PA00	数字输出（AFR=4）	VC2_OUT	PA02	数字输出（AFR=4）
	PA11	数字输出（AFR=2）		PA12	数字输出（AFR=2）

21.3.3 延迟 / 响应时间

设置控制寄存器 VCx_CR0 的 EN 位域为 1，使能 VC 模块。

从 VC 使能或 VC 的正负两端输入电压变化，到电压比较器输出正确比较结果的时间，被定义为比较器的延迟 / 响应时间。延迟 / 响应时间由控制寄存器 VCx_CR0 的 RESP 位域配置，且响应时间越短，VC 模块的功耗越大。

21.3.4 极性选择

电压比较器 VC1、VC2 的输出信号的极性，由控制寄存器 VCx_CR0 的 POL 位域设置：

- POL 为 1，VCx_OUTP 信号与 VCx_OUTA 信号极性相反，即正端大于负端时 VCx 输出低电平。
- POL 为 0，VCx_OUTP 信号与 VCx_OUTA 信号极性相同，即正端大于负端时 VCx 输出高电平。

21.3.5 数字滤波

电压比较器内置的数字滤波器，用于对电压比较器的输出信号进行数字滤波。用户可使用滤波功能过滤系统噪声，比如马达停止时的大电流噪声等，避免比较器的噪声输出引起系统的误动作。

数字滤波器的时钟由控制寄存器 VCx_CR1 的 FLTCLK 位域选择：

- FLTCLK 为 1，使用 PCLK 作为滤波时钟。
- FLTCLK 为 0，使用内置低速 RC 振荡器时钟 LSI 作为滤波时钟。

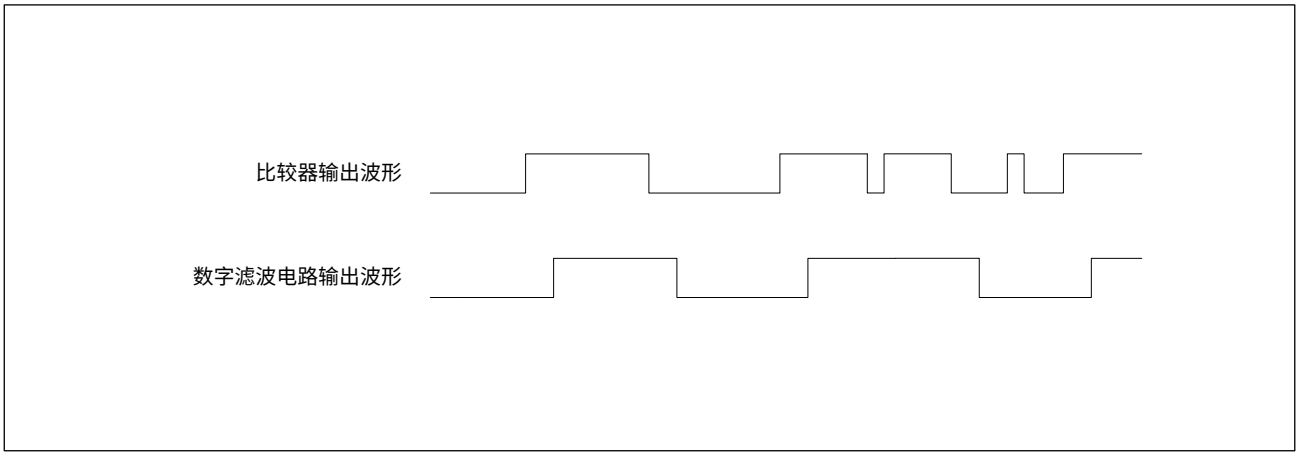
滤波器以一定的采样频率对输出信号进行采样，当连续采样到 N 个相同电平时信号有效，否则信号无效，以此滤除高频杂波信号。滤波单元的采样时钟为 FLTCLK 或 FLTCLK 的分频，通过 VCx_CR1 寄存器的 FLTTIME 位域可以选择 FLTCLK 的分频比及采样点个数 N。

电压比较器经数字滤波后的输出电平，可通过状态寄存器 VCx_SR 的 FLTV 位域读出。

当用户设置了 VCx_OUT 引脚为数字输出，且选择 VC 比较输出的复用功能时，VCx_OUT 引脚将输出电压比较器经数字滤波后的输出电平。

VC 的滤波响应波形，如下图所示：

图 21-2 VC 滤波响应时间



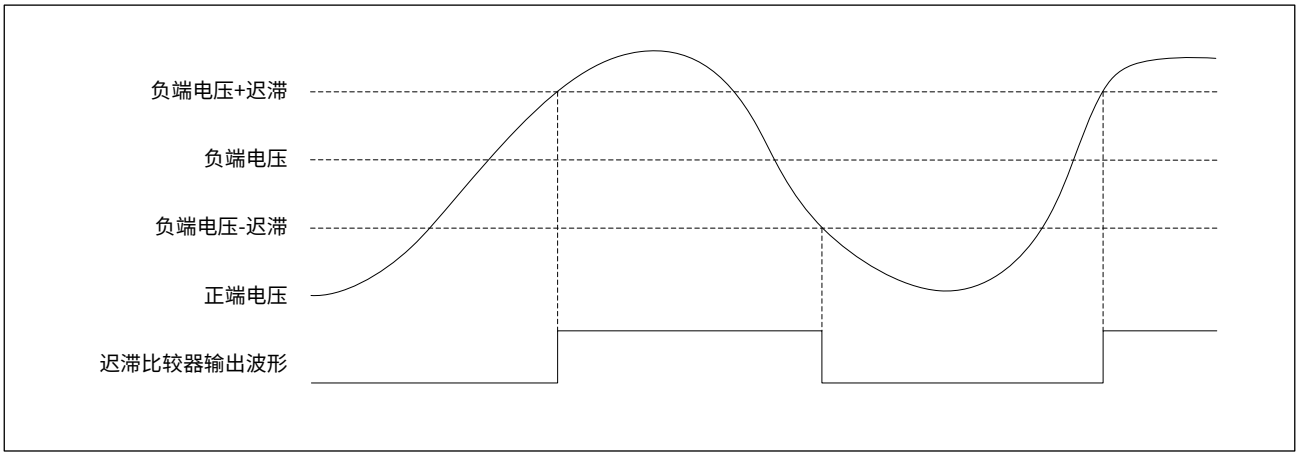
电压比较器的输出可以作为高级定时器 (ATIM) 的刹车输入或 OCREF_CLR 输入，关于刹车和 OCREF_CLR 的详细介绍请参见 14 高级定时器 (ATIM) 章节。此功能通常应用于电机保护场合，详细请参考相关应用笔记。

21.3.6 迟滞功能

模拟电压比较器支持迟滞功能，使用迟滞功能后，比较器的输出结果不会随输入信号的变化而立即翻转，而是在两路输入信号的偏移值高于或低于迟滞阈值电压后才发生翻转。

VC 迟滞功能波形示意图，如下图所示：

图 21-3 VC 迟滞功能



迟滞功能可以有效增强芯片的抗干扰能力，避免因输入信号的小幅度抖动，导致比较器的输出端不必要的频繁翻转。

迟滞阈值电压由控制寄存器 VCx_CR0 的 HYS 位域决定，如下表所示：

表 21-3 VC 迟滞阈值电压

VCx_CR0.HYS	迟滞窗口配置
0	没有迟滞
1	迟滞窗口大约 20mV

21.3.7 窗口比较功能

模拟电压比较器支持窗口比较功能，可将 VC1 和 VC2 的比较结果进行异或操作后输出，由控制寄存器 VCx_CR0 的 WINDOW 位域使能：

- WINDOW 为 1 时，VCx_OUTW 信号为 VC1_OUTP 信号与 VC2_OUTP 信号的异或值。
- WINDOW 为 0 时，VCx_OUTW 信号与 VCx_OUTP 信号电平相同。

21.3.8 BLANK 窗口功能

在保持 VCx 模块工作的同时，如果想暂时停止电压比较功能，或者为避免某些应用系统（比如电机控制）中，被监测信号短时间的合理波动造成电压比较器的输出电平发生不必要的翻转，本芯片的电压比较器增加了 BLANK 窗口功能，即，当指定的外部触发条件启动 BLANK 窗口时，在设定的 BLANK 窗口期内，不进行电压比较，电压比较器输出设定的高或低电平。BLANK 窗口期之后，电压比较器恢复正常工作。

BLANK 窗口持续时间，由控制寄存器 VCx_CR1 的 BLANKTIME 位域配置，窗口持续时间为

$$(2^{(BLANKTIME+2)} \sim 2^{(BLANKTIME+2)} + 2) \text{ 个 PCLK 周期}$$

如下表所示：

表 21-4 BLANK 窗口持续时间配置

VCx_CR1.BLANKTIME	BLANK 窗口 PCLK 时钟个数
000	4 ~ 6
001	8 ~ 10
010	16 ~ 18
011	32 ~ 34
100	64 ~ 66
101	128 ~ 130
110	256 ~ 258
111	512 ~ 514

BLANK 窗口持续期间电压比较器的输出电平由控制寄存器 VCx_CR1 的 BLANKLVL 位域进行设置，BLANKLVL 为 0 输出低电平，BLANKLVL 为 1 则输出高电平。

BLANK 窗口的触发启动条件，由控制寄存器 VCx_CR1 的 BLANKCH1、BLANKCH2、BLANKCH3、BLANKCH4、BLANKCH5、BLANKCH6 位域配置，分别由 ATIM 的 OC1REFC、OC2REFC、OC3REFC、OC4REFC、OC5REFC、OC6REFC 上升沿触发启动 BLANK 窗口。

21.4 VC 中断

CW32L011 的电压比较器支持在低功耗模式下工作，比较中断可将芯片从低功耗模式下唤醒。

设置控制寄存器 VCx_CR0 的 IE 位域为 1，使能 VCx 中断，产生中断时状态寄存器 VCx_SR 的中断标志位 INTF 会被硬件置 1，用户可以向 INTF 位写 0，清除中断标志。

设置控制寄存器 VCx_CR1 的 HIGHIE、RISEIE、FALLIE 位域，可选择不同的中断触发方式：

- HIGHIE 为 1，VCx_OUT 输出信号高电平触发中断。
- RISEIE 为 1，VCx_OUT 输出信号上升沿触发中断。
- FALLIE 为 1，VCx_OUT 输出信号下降沿触发中断。

21.5 编程示例

在此示例中，使用数字滤波功能，并配置比较中断，配置方法如下所示：

步骤 1：配置 VC_DIV.EN 为 1， 使能分压器；

步骤 2：配置 VC_DIV.DIV，设置分压系数；

步骤 3：配置 VCx_CR0.INP，选择正端待监测的电压来源；

步骤 4：配置 VCx_CR0.INN，选择负端待监测的电压来源；

步骤 5：配置 VCx_CR1.FLTTIME，选择滤波时间；

步骤 6：配置 VCx_CR1.FLTCLK，选择滤波时钟；

步骤 7：设置 VCx_CR1 寄存器的 HIGHIE、RISEIE、FALLIE 为 1，选择中断触发方式；

步骤 8：设置 VCx_CR0.IE 为 1，使能 VCx 中断；

步骤 9：设置 VCx_CR0.EN 为 1，使能 VCx；

步骤 10：在 VC 模块的初始化程序和 VC 模块的中断服务程序中，对 VCx_SR 的 INTF 位写入 0，清除中断标志后，允许 VCx 中断的产生。

21.6 寄存器列表

VC 基地址：VC_BASE = 0x4000 00A0

表 21-5 VC 寄存器列表

寄存器名称	寄存器地址	寄存器描述
VC_DIV	VC_BASE + 0x00	电阻分压控制寄存器
VCx_CR0	VC_BASE + 0x04 + 16×（x-1）	控制寄存器 0，x=1 ~ 2
VCx_CR1	VC_BASE + 0x08 + 16×（x-1）	控制寄存器 1，x=1 ~ 2
VCx_SR	VC_BASE + 0x0C + 16×（x-1）	状态寄存器，x=1 ~ 2

21.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

21.7.1 VC_DIV 电阻分压控制寄存器

Address offset: 0x00 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:6	RFU	-	保留位，请保持默认值
5	VIN	RW	电阻分压电路输入电压设置 0：VDDA 1：Vcore（约 1.6V）
4	EN	RW	电阻分压电路使能控制 0：禁止 1：使能
3	RFU	-	保留位，请保持默认值
2:0	DIV	RW	电阻分压电路输出电压配置 输出电压 = 输入电压 ×(1 + DIV) / 8

21.7.2 VCx_CR0 控制寄存器 0

Address offset: 0x04 + 16×（x-1） （x=1~2） Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:10	RFU	-	保留位，请保持默认值
9:8	INN	RW	VCx 负端输入信号配置 00：VCx_CH0 01：VCx_CH1 10：内置 1.2V 基准电压 11：内置分压器输出电压 *注：当输入信号超过（V_{DD}+0.2V）时，BGR 模块输出的 1.2V 电压会不精确。*
7:6	INP	RW	VCx 正端输入信号配置 00：VCx_CH0 01：VCx_CH1 10：VCx_CH2 11：VCx_CH3
5	WINDOW	RW	窗口比较功能配置 0：禁止窗口功能，VCx_OUTW 信号等于 VCx_OUTP 信号 1：使能窗口功能，VCx_OUTW 信号等于 VC1_OUTP 与 VC2_OUTP 的异或值
4	POL	RW	VCx 输出信号极性设置 0：正端大于负端时 VCx 输出高电平 1：正端大于负端时 VCx 输出低电平
3	IE	RW	VCx 中断使能配置 0：禁止 1：使能
2	HYS	RW	VCx 迟滞窗口配置 0：没有迟滞 1：迟滞窗口大约 20mV
1	RESP	RW	VCx 响应速度配置 0：低速 1：高速
0	EN	RW	VCx 使能控制 0：禁止 1：使能

21.7.3 VCx_CR1 控制寄存器 1

Address offset: 0x08 + 16×（x-1） （x=1~2） Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:18	RFU	-	保留位，请保持默认值
17	BLANKLVL	RW	BLANK 窗口持续期间输出电平配置 0：输出低电平 1：输出高电平
16:14	BLANKTIME	RW	BLANK 窗口持续时间配置： （$2^{(BLANKTIME+2)}$ ~ $2^{(BLANKTIME+2)}$+2）个 PCLK 周期 *注：ATIM 的 OC1REFC、OC2REFC、OC3REFC、OC4REFC、OC5REFC、 OC6REFC 任意一个上升沿都会触发启动 BLANK 窗口，在 BLANK 窗口持续时间内，不进行电压比较。*
13	BLANKCH6	RW	ATIM_OC6REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
12	BLANKCH5	RW	ATIM_OC5REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
11	BLANKCH4	RW	ATIM_OC4REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
10	BLANKCH3	RW	ATIM_OC3REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
9	BLANKCH2	RW	ATIM_OC2REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
8	BLANKCH1	RW	ATIM_OC1REFC 上升沿触发 VCx 启动 BLANK 窗口配置 0：禁止 1：使能
7	HIGHIE	RW	VCx 输出信号高电平触发中断使能 0：禁止 1：使能
6	RISEIE	RW	VCx 输出信号上升沿触发中断使能 0：禁止 1：使能
5	FALLIE	RW	VCx 输出信号下降沿触发中断使能 0：禁止 1：使能

位域	名称	权限	功能描述
4	FLTCLK	RW	数字滤波模块滤波时钟设置 0：LSI 1：PCLK *注：进入 DeepSleep 前，无需数字滤波功能时，请将滤波时钟源设置为 PCLK，否则 LSI 在 DeepSleep 下运行会产生功耗。*
3:0	FLTTIME	RW	数字滤波模块滤波时间配置 0000：无滤波 0001：$f_{SAMPLING} = f_{FLTCLK}/1$， N=2 0010：$f_{SAMPLING} = f_{FLTCLK}/1$， N=4 0011：$f_{SAMPLING} = f_{FLTCLK}/1$， N=8 0100：$f_{SAMPLING} = f_{FLTCLK}/2$， N=6 0101：$f_{SAMPLING} = f_{FLTCLK}/2$， N=8 0110：$f_{SAMPLING} = f_{FLTCLK}/4$， N=6 0111：$f_{SAMPLING} = f_{FLTCLK}/4$， N=8 1000：$f_{SAMPLING} = f_{FLTCLK}/8$， N=6 1001：$f_{SAMPLING} = f_{FLTCLK}/8$， N=8 1010：$f_{SAMPLING} = f_{FLTCLK}/16$， N=5 1011：$f_{SAMPLING} = f_{FLTCLK}/16$， N=6 1100：$f_{SAMPLING} = f_{FLTCLK}/16$， N=8 1101：$f_{SAMPLING} = f_{FLTCLK}/32$， N=5 1110：$f_{SAMPLING} = f_{FLTCLK}/32$， N=6 1111：$f_{SAMPLING} = f_{FLTCLK}/32$， N=8

21.7.4 VCx_SR 状态寄存器

Address offset: 0x0C + 16×（x-1） （x=1~2） Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:2	RFU	-	保留位，请保持默认值
1	FLTV	RO	数字滤波器输出的电平值 0：数字滤波器输出低电平 1：数字滤波器输出高电平
0	INTF	RW0	中断标志 R0：未发生 VC 中断 R1：已发生 VC 中断 W0：清除 VC 中断标志 W1：无功能

22 低电压检测器（LVD）

22.1 概述

低电压检测器 (LVD) 用于监测 VDDA 电源电压或外部引脚输入电压，当被监测电压与 LVD 阈值的比较结果满足触发条件时，将产生 LVD 中断或复位信号，通常用于处理一些紧急任务。

LVD 产生的中断和复位标志，只能由软件清零；只有当中断或复位标志被清零后，在再次达到触发条件时，LVD 才能再次产生中断或者复位信号。

22.2 主要特性

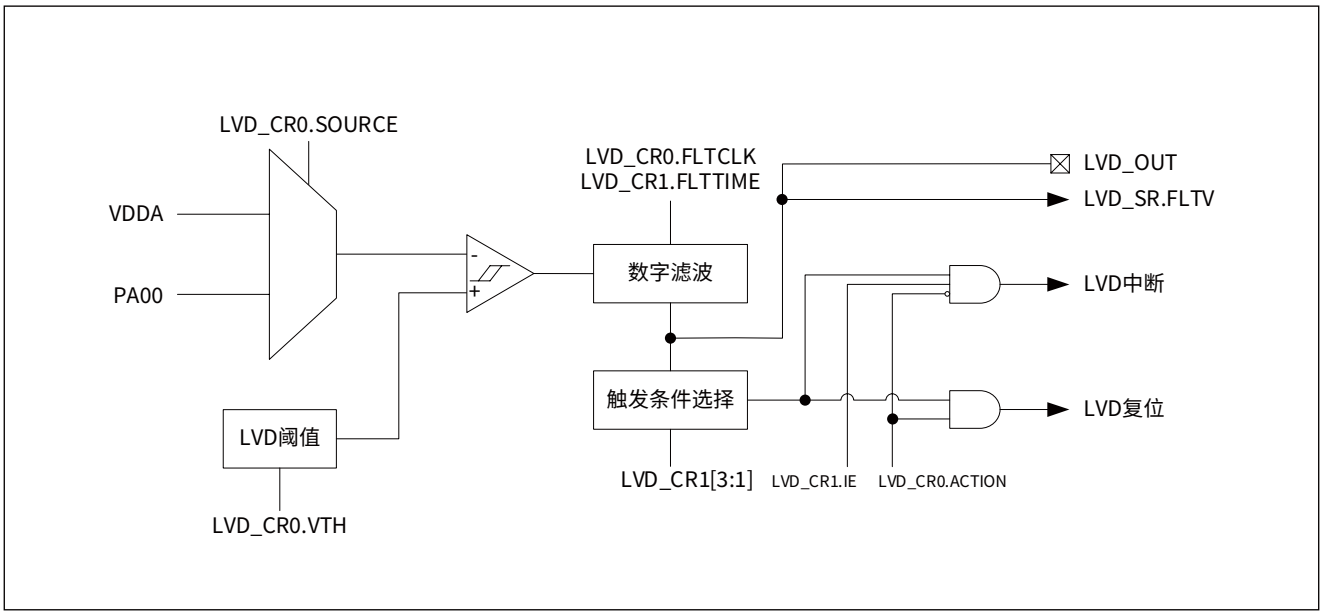
- 2 路监测电压源：VDDA 电源电压、PA00 引脚输入。
- 8 阶阈值电压，范围 1.8V ~ 4.6V。
- 3 种触发条件，可组合使用：
  - 电平触发：电压低于阈值。
  - 下降沿触发：电压跌落到阈值以下的下降沿。
  - 上升沿触发：电压回升到阈值以上的上升沿。
- 可触发产生中断或复位信号，二者不能同时产生。
- 可编程的滤波器和滤波时间。
- 支持迟滞功能。
- 支持低功耗模式下运行，中断唤醒 MCU。

22.3 功能描述

22.3.1 功能框图

低电压检测器 (LVD) 的功能框图如下图所示：

图 22-1 LVD 功能框图



LVD 可以监测 VDDA 电源电压，也可监测外部引脚输入电压（PA00），具体通过控制寄存器 LVD_CR0 的 SOURCE 位域来选择。当使用外部模拟信号输入时，用户必须将对应 GPIO 端口配置为模拟功能（GPIOx_ANALOG.PINy = 1）。

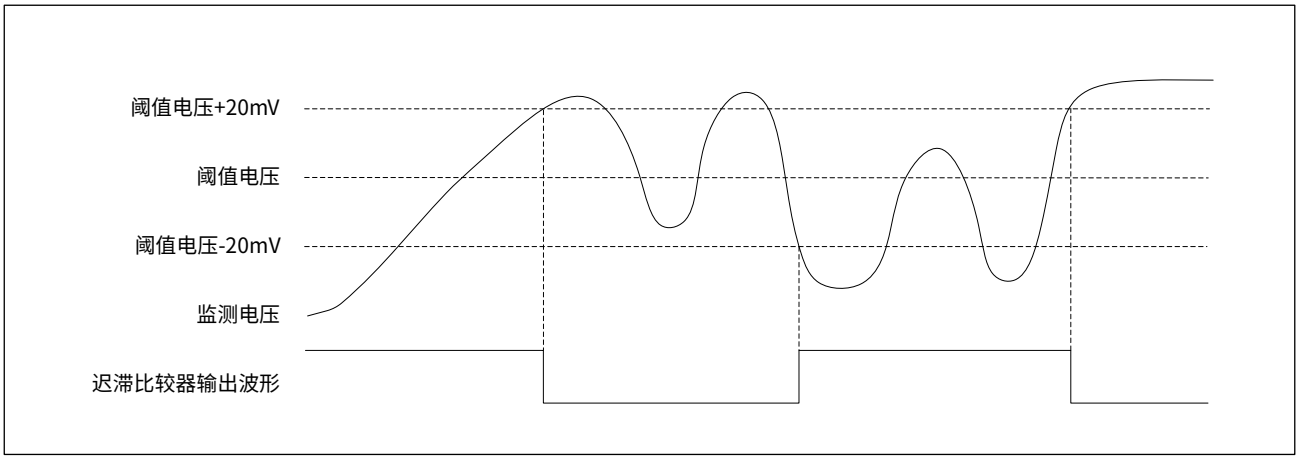
LVD 输出结果可以从 PA08/PC13 引脚输出，用户必须将对应 GPIO 端口配置为数字输出，同时选择功能复用。

22.3.2 迟滞功能

LVD 内置的电压比较器具有迟滞功能，可避免当 LVD 的被监测电压在阈值电压附近时，电压比较器的输出结果发生频繁翻转，增强系统抗干扰能力。

只有当被监测电压高于或低于阈值电压达到 20mV 时，比较器输出信号才会发生翻转。具体波形如下图所示：

图 22-2 LVD 迟滞响应



LVD 的阈值电压由控制寄存器 LVD_CR0 的 VTH 位值决定，有效值 0 ~ 7，如下表所示：

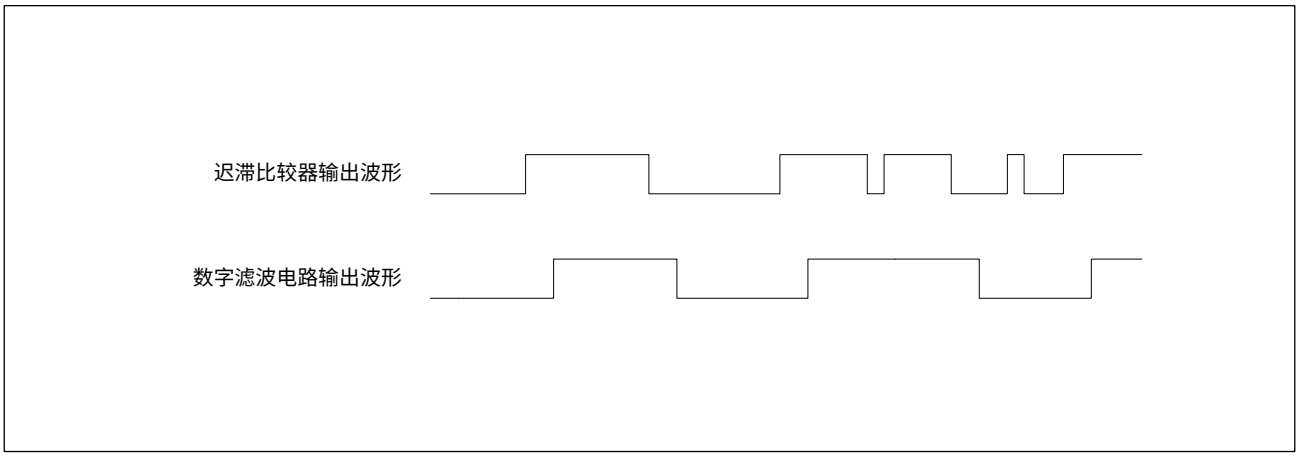
表 22-1 LVD 阈值电压

LVD_CR0.VTH	阈值电压（单位：V）	LVD_CR0.VTH	阈值电压（单位：V）
000	1.8	100	3.4
001	2.2	101	3.8
010	2.6	110	4.2
011	3.0	111	4.6

22.3.3 数字滤波

为增强系统的鲁棒性，LVD 支持数字滤波功能，可将 LVD 电压比较的输出结果信号进行数字滤波，小于滤波宽度的信号被滤除，不会触发中断或复位。具体波形如下图所示：

图 22-3 LVD 滤波输出



控制寄存器 LVD_CR0 的 FLTCLK 位域用于选择数字滤波的时钟：

- FLTCLK 位为 1，选择 SYSCLK 作为滤波时钟。
- FLTCLK 位为 0，选择内置低速 RC 振荡器时钟 LSI 作为滤波时钟。

滤波器以一定的采样频率对输出信号进行采样，当连续采样到 N 个相同电平时信号有效，否则信号无效，以此滤除高频杂波信号。滤波单元的采样时钟为 FLTCLK 或 FLTCLK 的分频，通过 LVD_CR1 寄存器的 FLTTIME 位域可以选择 FLTCLK 的分频比及采样点个数 N。

从 LVD 状态寄存器 LVD_SR 的 FLTV 位域，可以读出经 LVD 数字滤波后的信号电平；当 GPIO 的功能复用为 LVD_OUT 时，数字滤波后的信号就可以从 GPIO 输出，以方便观察测量。

22.4 LVD 中断

LVD 支持在低功耗模式下工作，中断输出可将芯片从低功耗模式下唤醒。

当被监测电压与 LVD 阈值的比较结果满足触发条件时，可产生中断或复位信号。产生中断还是复位信号由控制寄存器 LVD_CR0 的 ACTION 位域控制：

- ACTION 为 1，LVD 触发产生复位。
- ACTION 为 0，LVD 触发产生中断。

设置控制寄存器 LVD_CR1 的 IE 位域为 1，使能 LVD 中断，满足触发条件时将产生 LVD 中断，中断标志位 LVD_SR.INTF 会被硬件置 1，用户可以向 INTF 位写 0，清除中断标志。

设置控制寄存器 LVD_CR1 的 LEVEL、FALL、RISE 位域，可选择不同的中断或复位触发方式，三者可组合使用：

- LEVEL 为 1，被监测电压低于阈值时触发中断或产生复位。
- FALL 为 1，被监测电压跌落到阈值以下的下降沿触发中断或产生复位。
- RISE 为 1，被监测电压回升到阈值以上的上升沿触发中断或产生复位。

22.5 编程示例

22.5.1 欠压复位编程示例

在此示例中，被监测电压低于阈值电压时复位 MCU，使用数字滤波功能。

配置方法如下所示：

步骤 1：配置 LVD_CR0.SOURCE，选择待监测的电压来源；

步骤 2：配置 LVD_CR0.VTH，设置阈值电压；

步骤 3：配置 LVD_CR1.FLTTIME，选择 LVD 滤波时间；

步骤 4：配置 LVD_CR0.FLTCLK，选择滤波时钟；

步骤 5：设置 LVD_CR1.LEVEL 为 1，选择被监测电压低于阈值时触发 LVD 动作；

步骤 6：设置 LVD_CR0.ACTION 为 1，选择 LVD 触发动作为系统复位；

步骤 7：设置 LVD_CR0.EN 为 1，使能 LVD。

22.5.2 中断编程示例

在此示例中，被监测电压高于或低于阈值电压时产生中断，使用数字滤波功能。

配置方法如下所示：

步骤 1：配置 LVD_CR0.SOURCE，选择待监测的电压来源；

步骤 2：配置 LVD_CR0.VTH，选择阈值电压；

步骤 3：配置 LVD_CR1.FLTTIME，选择 LVD 滤波时间；

步骤 4：配置 LVD_CR0.FLTCLK，选择滤波时钟；

步骤 5：设置 LVD_CR1.RISE 和 FALL 均为 1，选择上升沿和下降沿触发；

步骤 6：设置 LVD_CR0.ACTION 为 0，选择 LVD 触发动作为中断；

步骤 7：设置 LVD_CR1.IE 为 1，使能 LVD 中断；

步骤 8：使能 NVIC 中断向量表中的 LVD 中断；

步骤 9：设置 LVD_CR0.EN 为 1，使能 LVD；

步骤 10：分别在 LVD 初始化程序和 LVD 中断服务程序中，对 LVD_SR.INTF 位写入 0，以清除中断标志，允许产生新的 LVD 中断。

22.6 寄存器列表

LVD 基地址：LVD_BASE = 0x4000 00D0

表 22-2 LVD 寄存器列表

寄存器名称	寄存器地址	寄存器描述
LVD_CR0	LVD_BASE + 0x00	控制寄存器 0
LVD_CR1	LVD_BASE + 0x04	控制寄存器 1
LVD_SR	LVD_BASE + 0x08	状态寄存器

22.7 寄存器描述

有关寄存器描述里所使用的缩写，请参见 1 文档约定章节。

22.7.1 LVD_CR0 控制寄存器 0

Address offset: 0x00 Reset value: 0x0000 0100

位域	名称	权限	功能描述
31:9	RFU	-	保留位，请保持默认值
8	FLTCLK	RW	数字滤波模块滤波时钟设置 0：LSI 1：SYSCLK *注：进入 DeepSleep 前，无需数字滤波功能时，请将滤波时钟源设置为 SYSCLK，否则 LSI 在 DeepSleep 下运行会产生功耗。*
7	RFU	-	保留位，请保持默认值
6:4	VTH	RW	阈值电压选择 000： 1.8V 001： 2.2V 010： 2.6V 011： 3.0V 100： 3.4V 101： 3.8V 110： 4.2V 111： 4.6V
3	RFU	-	保留位，请保持默认值
2	SOURCE	RW	监测来源配置 0：VDDA 电源电压 1：PA00 端口输入电压
1	ACTION	RW	触发动作配置 0：NVIC 中断 1：系统复位
0	EN	RW	使能控制 0：禁止 LVD 1：使能 LVD

22.7.2 LVD_CR1 控制寄存器 1

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7:4	FLTTIME	RW	数字滤波模块滤波时间配置 0000：无滤波 0001：$f_{SAMPLING} = f_{FLTCLK}/1$， N=2 0010：$f_{SAMPLING} = f_{FLTCLK}/1$， N=4 0011：$f_{SAMPLING} = f_{FLTCLK}/1$， N=8 0100：$f_{SAMPLING} = f_{FLTCLK}/2$， N=6 0101：$f_{SAMPLING} = f_{FLTCLK}/2$， N=8 0110：$f_{SAMPLING} = f_{FLTCLK}/4$， N=6 0111：$f_{SAMPLING} = f_{FLTCLK}/4$， N=8 1000：$f_{SAMPLING} = f_{FLTCLK}/8$， N=6 1001：$f_{SAMPLING} = f_{FLTCLK}/8$， N=8 1010：$f_{SAMPLING} = f_{FLTCLK}/16$， N=5 1011：$f_{SAMPLING} = f_{FLTCLK}/16$， N=6 1100：$f_{SAMPLING} = f_{FLTCLK}/16$， N=8 1101：$f_{SAMPLING} = f_{FLTCLK}/32$， N=5 1110：$f_{SAMPLING} = f_{FLTCLK}/32$， N=6 1111：$f_{SAMPLING} = f_{FLTCLK}/32$， N=8
3	FALL	RW	被监测电压跌落到阈值以下的下降沿触发 （即 LVD 输出信号上升沿触发） 0：禁止 1：使能
2	RISE	RW	被监测电压回升到阈值以上的上升沿触发 （即 LVD 输出信号下降沿触发） 0：禁止 1：使能
1	LEVEL	RW	被监测电压低于阈值电压触发 （即 LVD 输出信号高电平触发） 0：禁止 1：使能
0	IE	RW	LVD 中断使能控制 0：禁止 1：使能

22.7.3 LVD_SR 状态寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:2	RFU	-	保留位，请保持默认值
1	FLTV	RO	数字滤波器输出的电平值 0：数字滤波器输出低电平 1：数字滤波器输出高电平
0	INTF	RW0	中断标志 R0：未发生 LVD 中断 R1：已发生 LVD 中断 W0：清除 LVD 中断标志 W1：无功能

23 调试接口（DBG）

23.1 概述

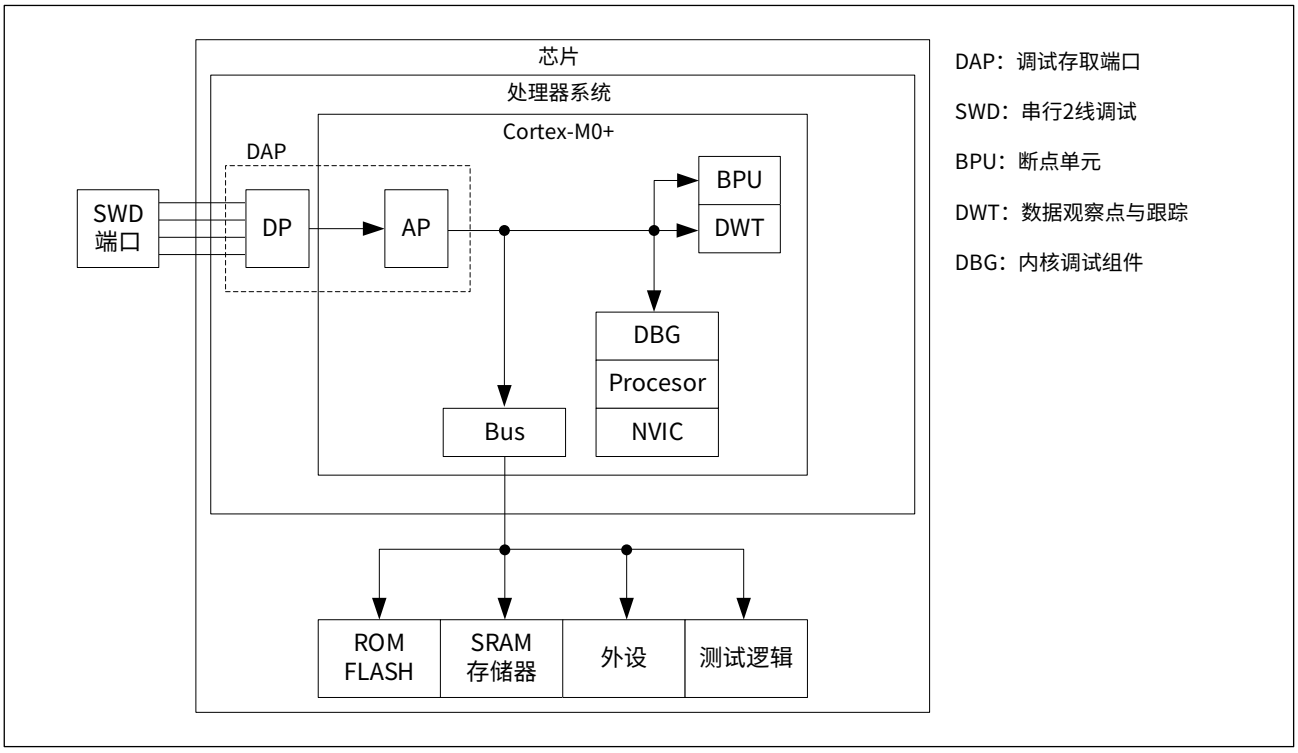
CW32L011 的内核为 ARM® Cortex®-M0+，内核内置 DAP 硬件调试模块，支持 SWD 模式调试。硬件调试模块可实现在取指 (指令断点) 或访问数据 (数据断点) 时挂起，程序停止运行，调试器可通过 DAP 对 M0 的内核状态和片内的外设状态及存储单元进行查询；且内核和外设可以被复原，程序继续执行。

当使用调试仿真工具通过 SWD 接口连接到 CW32L011，进入调试模式，通过芯片内核中的 DAP 硬件调试模块进行调试操作。

23.2 串行线调试端口 SWD

使用 CW 专用调试器或通用调试仿真工具的 SWD 接口和目标芯片内部的 DAP 调试模块连接，通过包传输协议进行数据交换，实现调试操作。SWD 方式是 2 线串行通信，包括一条时钟线 SWCLK 和一条双向数据线 SWDIO。如下图所示：

图 23-1 SWD 功能框图



CW32L011 系列的 SWD 接口引脚分配如下表所示，PA13/PA14 引脚在芯片出厂时默认为 SWD 功能。

表 23-1 SWD 引脚分配

SWD 端口名称	引脚功能	引脚分配
SWCLK	串行时钟输入	PA14
SWDIO	串行数据输入输出	PA13

PA13/PA14 引脚可配置为 SWD 功能或 GPIO/ISP 功能，由系统控制寄存器 SYSCTRL_CR2 的 SWDIO 位域进行功能配置。SWDIO 为 0，PA13/PA14 引脚被配置为 SWD 功能；SWDIO 为 1，则被配置为 GPIO/ISP 功能。

通常建议 SWD 引脚使用 10KΩ 上拉电阻，CW32L011 的 PA13/PA14 作为 SWD 功能时内置有上拉电阻，其阻值在 55kΩ ~ 70kΩ 之间，用户可以在外部增加上拉电阻，以提高抗干扰性能。

CW32L011 系列的 PA13/PA14 引脚默认为 SWD 功能，如果用户设定了加密等级，则需要根据设定的等级来判别是否支持 SWD 功能，SWD 引脚的配置与功能如下表所示：

表 23-2 SWD 引脚功能

加密等级	SYSCTRL_CR2.SWDIO	PA13/PA14 功能
Level0/1 加密	0（默认）	SWD 功能
	1	GPIO/ISP 功能
Level2 加密	0（默认）	NA，无任何功能
	1	GPIO/ISP 功能
Level3 加密	0（默认）	NA，无任何功能
	1	GPIO 功能

注意事项：

1. 加密等级可通过 ISP 通信方式进行设定，详情请参阅 ISP 通信协议文档；

2. 芯片上电后 SWDIO 和 SWCLK 均默认为内部上拉，用户可不用外接上拉电阻；

3. 当芯片加密等级设定为 2 时，SWD 功能被禁止，只能通过 ISP 方式烧录；

4. 当芯片加密等级设定为 3 时，SWD 功能及 ISP 功能均被禁止，芯片无法再次烧录新程序。

23.3 调试通信协议

调试器和目标芯片的 DAP 调试模块通过 SWD 包传输协议进行通信，包传输协议为 2 线同步串行协议，使用 SWCLK 时钟信号和 SWDIO 数据信号：

- SWCLK 为单向时钟信号，由调试器输出给目标芯片。
- SWDIO 为双向数据信号，由调试器和目标芯片双向分时驱动。

协议定义了长度为一个 SWCLK 周期的收发端转换时间，在收发端转换时间内，调试器和目标芯片都不驱动 SWDIO，SWDIO 由上拉电阻上拉到高电平。

SWDIO 信号线上传输数据时，遵循最低位 LSB 最先传输，最高位 MSB 最后传输原则。

通过包传输协议，调试器可以对目标芯片 DAP 调试模块内的 DP 寄存器和 AP 寄存器进行读写访问，下文中写作 SW-DP 和 SW-AP。

23.3.1 传输协议格式

DAP 调试模块包含有 DP 调试端口寄存器和 AP 存取端口寄存器，调试器和目标芯片的 DAP 调试模块进行通信实际上就是对 DP 寄存器和 AP 寄存器的读写操作。

传输通信帧通常包含 3 个字段：

- 包请求：长度为 8bit，调试器到目标芯片。
- 响应：长度为 3bit，目标芯片到调试器。
- 数据传输：长度为 33bit，目标芯片到调试器或者调试器到目标芯片，可选字段。

包请求各位的功能定义如下：

表 23-3 包请求位定义

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
3:4	A[3:2]	DP 或 AP 寄存器的地址字段，bit3 为 A2，bit4 为 A3
5	奇偶校验	前面 5bit 的 bit 奇偶校验位
6	停止	0
7	驻留	不受主机驱动。由于存在上拉，目标芯片读出为 1

请求包后面始终为收发端转换时间（默认 1bit），此时主机和目标都不驱动 SWDIO。

目标发送的响应位定义如下：

表 23-4 目标发送响应位定义

位	名称	说明
0:2	ACK	001：FAULT（bit0:0, bit1:0, bit2:1） 010：WAIT （bit0:0, bit1:1, bit2:0） 100：OK （bit0:1, bit1:0, bit2:0）

ACK 回应后为收发端转换时间（默认 1bit），此时主机和目标都不会驱动 SWDIO。

调试器或目标芯片发送的数据，位定义如下：

表 23-5 数据传输位定义

位	名称	说明
0:31	WDATA 或 RDATA	写入或读取数据
32	奇偶校验	32bit 数据的 bit 奇偶校验位

数据传输阶段不是必须的，如下两种情况才有数据传输：

- 数据读或者写请求后收到的回应是 OK 响应。
- DP-CTRL/STAT 寄存器的 ORUNDETECT 标志位被置位 1，此时无论收到何种回应（包括 WAIT 和 FAULT）都要求必须有数据传输。

23.3.2 SW-DP 状态机

SW-DP 内部有一个 ID CODE 寄存器，固定值为 0x0BC11477（为 ARM 公司 Cortex®-M0+ 识别码）。在采用 SWD 包协议进行目标寄存器读写之前，必须先读取此 ID 号以激活目标芯片的 SW-DP 逻辑，否则目标芯片的 SW-DP 的状态机不工作。

操作步骤如下：

步骤 1：在上电复位后或者 SWDIO 线路处于高电平超过 50 个周期后，SW-DP 状态机进入复位状态；

步骤 2：进入复位状态后，保持 SWDIO 线路处于低电平至少 2 个周期，SW-DP 状态机进入空闲状态；

步骤 3：进入空闲状态后，对 DP-SW 的 ID CODE 寄存器执行读访问；

注意：

如果不执行此操作，则目标板在后续的包通信响应阶段，会发送 FAULT 响应。

步骤 4：按照包协议对需要访问的寄存器进行读写访问。

23.3.3 SW-DP 和 SW-AP 的读写访问

- SW-DP/SW-DP 的写访问：调试器接收到 ACK 后，经过 1 个时钟周期的转换时间后，必须立即发送数据。
- SW-DP 的读访问：
  - 当目标芯片已做好数据发送准备，则发送 OK 响应，然后立即发送数据。
  - 当目标芯片 DAP 未做好数据发送准备，则发送 WAIT 响应结束本次通信。
  - 当目标芯片 DAP 状态错误时，发送 FAULT 响应，结束本次通信。
- SW-AP 的读访问：对 AP 的读访问为寄存式，即本次读 AP 操作并不能返回所需要的结果，而是等到下次 AP 操作才能返回本次读操作的结果；如果要执行的下次访问不是 AP 读访问操作，则必须读取 DP-RDBUFF 寄存器来获取本次读操作的结果。
- 每次进行 SW-AP 读访问或 DP-RDBUFF 读请求时都会更新 M0 内核的调试控制 / 状态寄存器 DP-CTRL/STAT 寄存器的 READOK 标志位，以指示 AP 读访问是否成功。
- SW-DP 有写缓冲区（用于 SW-DP 或 SW-AP 写入），在读写操作未完成时，可以接受下一个写入操作。如果 SW-DP 的写缓冲区已满，则芯片 SW-DP 逻辑会回复 WAIT 响应以通知主机暂缓操作。特殊操作除外，如 IDCODE 读取、DP-CTRL/STAT 读取或 ABORT 写入操作，这几项操作在写缓冲区已满时也会被接受。
- 由于 SWCLK 和 HCLK 不是同步时钟，属于异步时钟域，因此写操作后（数据传输的奇偶校验位后）还需要两个额外的 SWCLK 周期，以保证写入操作生效。在主机驱动这 2 个 SWCLK 周期时应将 SWDIO 线路驱动为低电平（空闲状态）。在对 DP-CTRL/STAT 寄存器写入上电请求操作时，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，将导致执行失败。

23.3.4 SW-DP 寄存器

当 APnDP 为 0 时，访问 DP 寄存器，由 A[3:2] 寻址，地址相同时因读写操作不同，寄存器含义可能不相同，具体如下表所示：

表 23-6 DP 寄存器列表及定义

A[3:2]	读 / 写	寄存器名称	寄存器内容说明
00	读	IDCODE	固定为 0x0BC1 1477（用于标识 SW-DP）， ARM 的 Cortex® -M0+ 的识别码
	写	ABORT	见表 23-7 DP ABORT 寄存器位定义
01	读 / 写	DP-CTRL/STAT (SELECT.CTRLSEL = 0)	主要用于： 1、请求系统或调试上电； 2、配置 AP 访问的传输操作； 3、控制比较和验证操作； 4、读取一些状态标志（上溢和上电确认）。
		WIRE CONTROL (SELECT.CTRLSEL = 1)	用于配置物理串行端口协议（如转换时间的持续时间等）
10	读	READ RESEND	允许从已损坏的调试传输中恢复读取数据，无需重复执行原始 AP 传输。
	写	SELECT	用于选择当前的访问 AP 端口和以及 AP 端口内 4 字寄存器 BANK。 见表 23-8 DP SELECT 寄存器
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 事务时提供读取本次 AP 请求的结果）。此读取缓冲区捕获 AP 中的数据，显示为前一次读取的结果，无需启动新操作。

表 23-7 DP ABORT 寄存器位定义

位	名称	定义
31:5	保留	
4	ORUNERRCLR	写 1 清除 STICKYORUN 错误标志
3	WDATAERR	写 1 清除 WDATAERR 错误标志
2	STICKYERR	写 1 清除 STICKYERR 错误标志
1	STICKYCMP	写 1 清除 STICKYCMP 标志
0	DAPABORT	写 1 产生 DAP ABORT 信号，放弃当前存取操作；当目标芯片回应 WAIT 响应时必须执行此操作来中止此次操作。

表 23-8 DP SELECT 寄存器

位	名称	定义
31:24	APSEL	选择当前 AP 端口，固定为 b00000000
23:8	保留	
7:4	APBANKSEL	在当前 AP 上选择活动的 4 字寄存器 BANK
3:1	保留	
0	CTRLSEL	DP 端口寄存器选择： 复位后默认为 0，选择 CTRL/STAT 寄存器； 设置为 1，选择 WCR 寄存器（Wire Control Register）。

23.3.5 SW-AP 寄存器

当 APnDP 为 1 时，访问 AP 寄存器。由于 SW-AP 寄存器较多，需要将 A[3:2] 和 SW-DP 选择寄存器 SELECT 的 APBANKSEL 位域值组合才能对 SW-AP 寄存器进行唯一寻址。

本文不对 SW-AP 逐一赘述，仅介绍一个常用的 IDR 公共寄存器（地址为 0xFC），如下表所示：

表 23-9 AP IDR 公共寄存器

位	定义
31:28	AP 设计的版本 Rervision。
27:24	AP 设计者标识码，本芯片固定为 0x04。
23:17	AP 设计者标识码，本芯片固定为 0x3B。
16:13	AP 类型。如存储器存取端口类型标识码为 b1000，未定义类型存取端口类型标识码为 b0000。
12:8	保留
7:0	AP 识别码。如 MEM-AP 或者 JTAG-AP 等。

23.4 内核调试

通过操作内核调试寄存器可实现对内核的调试，调试器通过 DAP 调试访问这些寄存器。

内核调试寄存器主要包括 4 个寄存器，如下表所示：

表 23-10 内核调试寄存器

寄存器	说明
DHCSR	调试暂停控制和状态寄存器，32bit； 控制处理器的暂停、单步和重启等动作。
DCRSR	调试内核寄存器选择器寄存器，17bit； 在暂停期间控制对内核寄存器的读和写。
DCRDR	调试内核寄存器数据寄存器，32bit； 暂停期间读写内核内核寄存器的数据传输寄存器。
DEMCR	调试异常监视控制寄存器，32bit； 用于使能数据监视点单元和向量捕捉特性，利用向量捕捉，调试器可以在处理器复位或者硬件错误产生时暂停处理器。

这些寄存器只能通过上电复位来复位。更多详细信息请参阅《Cortex®-M0+ Technical Reference Manual》。

23.5 断点单元 BPU

Cortex®-M0+ BPU 断点单元提供四个断点寄存器，实现了基于 PC 指针的断点功能。

有关 BPU CoreSight 组件的标识寄存器和访问类型的更多信息，请参阅《ARMv6-M Architecture Reference Manual》和 ARM® CoreSight 组件技术参考手册。

23.6 数据观察点与跟踪 DWT

Cortex®-M0+ DWT 提供了两个观察点寄存器组。实现如下功能：

- 设置数据监视点：数据或者外设的地址可以被标记为监视变量，对该地址的访问会产生调试事件，会暂停程序执行。
- ARMv6-M 中可选的 DWT 程序计数器采样寄存器 (DWT_PCSR) 功能。允许调试程序定期采样 PC 指针，提供粗略分析，无需停止处理器。

有关更多信息，请参阅《ARMv6-M Architecture Reference Manual》。

23.7 调试组件 DBG

调试器通过调试组件 DBG 实现断点期间的定时器、看门狗、RTC 等外设的时钟控制支持。通过调试状态定时器控制寄存器 SYSCTRL_DEBUG 设置，可控制定时器、看门狗定时器、RTC 等在调试状态下断点期间正常运行或暂停，详见下表：

表 23-11 调试状态定时器控制寄存器 SYSCTRL_DEBUG

位	名称	值	调试状态下断点期间计数器功能配置
9	IWDT	1	调试状态下，IWDT 计数器暂停计数
		0	调试状态下，IWDT 计数器正常计数
8	RTC	1	调试状态下，RTC 计数器暂停计数
		0	调试状态下，RTC 计数器正常计数
6	LPTIM	1	调试状态下，LPTIM 计数器暂停计数
		0	调试状态下，LPTIM 计数器正常计数
5	BTIM123	1	调试状态下，BTIM1、BTIM2、BTIM3 计数器暂停计数
		0	调试状态下，BTIM1、BTIM2、BTIM3 计数器正常计数
2	GTIM2	1	调试状态下，GTIM2 计数器暂停计数
		0	调试状态下，GTIM2 计数器正常计数
1	GTIM1	1	调试状态下，GTIM1 计数器暂停计数
		0	调试状态下，GTIM1 计数器正常计数
0	ATIM	1	调试状态下，ATIM 计数器暂停计数
		0	调试状态下，ATIM 计数器正常计数

如当定时器输出 PWM 进行电机控制时，定时器不能停止，否则会造成电机损坏。又如，看门狗定时器在断点期间要停止计数，以防止系统发生不希望的复位。

23.8 注意事项

CW32L011 在调试期间需要使用 HCLK 进行调试连接，不允许在调试会话期间关闭 HCLK，因此在调试环境下执行 WFI，会关闭内核 HCLK，导致调试功能失效。

如目标芯片当前已经进入深度休眠模式，此时 DAP 硬件调试模块不工作，调试器无法和目标芯片的 DAP 硬件调试模块进行连接，无法实现调试功能。

用户必须保证芯片处于运行模式（Active mode）或休眠模式（Sleep mode）才能使用调试器进行调试。

因此建议用户在合并深度休眠的相关代码前，先完成非深度休眠模式下设备的所有功能调试，最后再进行深度休眠相关代码的调试。

24 数字签名

24.1 概述

数字签名主要用来存放芯片唯一身份标识（UID）、产品型号、FLASH 容量、SRAM 容量、芯片封装引脚数等信息，可以通过 SWD 或者 CPU 读取。数字签名相关信息在出厂时编程，用户固件或外部设备可通过读取数字签名来对芯片的合法性进行验证。

24.2 产品唯一身份标识（UID）寄存器（80bit）

UID 寄存器存储了芯片的唯一身份标识符，其地址为 0x0010 07B0 - 0x0010 07B9，共 80bit。UID 在芯片生产时写入，用户无法修改。UID 寄存器支持以单字节 / 半字 / 全字等方式读取，然后使用自定义算法连接起来。

唯一身份标识符典型应用场景：

- 用作设备序列号
- 设备合法性验证，防止盗版

  用户在设备生产时采用私有密钥对 UID 进行加密运算，并将计算结果存放在主 FLASH 存储器或 OTP 存储器，程序在设备启动后，读取 UID 并采用同样的密钥进行加密运算，并将运算结果和之前存储的计算结果进行比较，相同则认为该设备是合法的，否则程序不启动，可有效防止用户设备被非法复制（盗版）。
- 作为安全密钥使用

  用户结合 UID 和私有算法，可在用户对 FLASH 编程前进行安全校验，提高 FLASH 内代码的安全性。
- 激活安全启动流程等

24.3 产品型号寄存器

产品型号寄存器存储了产品型号的 ASCII 码，其地址为 0x0010 07E0 - 0x0010 07EF，共 16 字节。产品型号不足 16 字节时，用 0x00 进行填充。

如芯片的型号为 CW32L011K8T6，对应存储 (从低地址开始) 的数据为：0x43 0x57 0x33 0x32 0x4C 0x30 0x31 0x31 0x4B 0x38 0x54 0x36 0x00 0x00 0x00 0x00。

注：

产品型号中如果有字母，则字母需要大写。

24.4 FLASH 容量寄存器

FLASH 容量寄存器存储了芯片内置 FLASH 存储器的容量大小，其地址为 0x0010 07D8 - 0x0010 07DB，共 4 字节。

从 FLASH 容量寄存器读出的 FLASH 容量大小以字节为单位，如 0x0001 0000 代表 64KB，0x0000 8000 代表 32KB。

24.5 SRAM 容量寄存器

SRAM 容量寄存器存储了芯片内置 SRAM 存储器的容量大小，其地址为 0x0010 07D6 - 0x0010 07D7，共 2 字节。

从 SRAM 容量寄存器读出的 SRAM 容量大小以字节为单位，如 0x1000 代表 4KB。

24.6 引脚数量寄存器

引脚数量寄存器存储了芯片引脚数，其地址为 0x0010 07D5，共 1 字节。如 0x20 代表 32Pin。

25 版本信息

表 25-1 文档修订信息

日期	版本	变更信息
2025-05-07	Rev 1.0	初始发布
2025-09-19	Rev 1.1	更新图 16-4 异步工作模式发送控制时序（发送两个字节数据）； 更新 ARM 公司 Cortex®-M0+ 识别码。