SYM32F003 参考手册

ARM® Cortex®-M0+ 32 位微控制器

1 产品特性

- 内核：ARM® Cortex®-M0+ CPU
- 20K 字节 FLASH，具有擦写保护功能，支持 ISP、ICP、IAP，4 级安全保护
- 22 字节 OTP，通过 ISP 协议写入
- 3K 字节 RAM，具备硬件奇偶校验功能
- 复位和电源管理
  - 低功耗模式（Sleep，DeepSleep）
  - 上电和掉电复位（POR/BOR）
  - 可编程低电压检测器（LVD）
- 时钟管理
  - 4 ~ 32MHz 外部输入时钟
  - 内置 48MHz RC 振荡器
  - 内置 32.768kHz RC 振荡器
  - 内置 8kHz RC 振荡器
  - 内置 120kHz RC 振荡器
- 多达 21+1 路 I/O 管脚
  - 所有 I/O 口支持具备滤波的中断功能
  - 所有 I/O 口支持具备滤波的唤醒功能
- 定时器
  - 1 个 16 位高级定时器，支持 3 对具有死区的互补 PWM 输出
  - 1 个 16 位通用定时器，支持 4 路 PWM 输出或输入捕捉、支持编码计数
  - 3 个 16 位基本定时器
  - 1 个自动唤醒定时器，可编程唤醒周期
  - 1 个窗口看门狗，采用 PCLK 计数
  - 1 个独立看门狗，采用专用时钟计数
- 通信接口
  - 2 路低功耗 UART，支持异步模式、同步模式，支持小数波特率，支持低功耗模式接收数据
  - 1 路标准 SPI 接口，支持 4~16bit 位宽
  - 1 路标准 I2C 接口，速率可达 1M bps
  - 1 路 IR 调制器，可编程占空比和极性
- CRC 硬件计算单元，支持 4 种算法
- 14 位模数转换器（ADC）
  - 高达 1M SPS 转换速度
  - 内置电压参考、温度传感器
  - 内置电压跟随器
- 2 路电压比较器，内置可编程参考电压
- 1 路低电压检测器，内置 16 阶比较基准，可监测端口电压及电源电压
- 标准 SWD 调试接口
- 安全特性
  - 80 比特 UID / 48 比特 UID
  - 符合 IEC/UL 60730 相关标准
  - 异常存储空间访问报错
  - 特殊 SFR 保护，防止误操作
- 工作温度：-40℃ ~ 105℃
- 工作电压：1.65V ~ 5.5V
- 速度等级：
  - 0 ~ 24MHz @ 1.65 ~ 5.5V
  - 0 ~ 48MHz @ 1.80 ~ 5.5V

2 系统结构

该系列微控制器的系统结构包括 Arm® Cortex®-M0+处理器、总线结构和存储器体系，将在后继章节中描述。该处理器是新一代的处理器内核，具有许多新的特性，适合于需要高性能和低功耗微控制器的市场。该处理器通过 AHB-Lite 总线接口访问存储器及片内外设。

2.1 处理器

本芯片所采用的 Cortex®-M0+处理器具有门数低、效能高的特点；专为要求面积优化、低功耗处理器的微控制器及深度嵌入式应用而设计。该处理器基于 ARMv6-M 架构，同时支持 Thumb®指令集、单周期输入/输出端口、硬件乘法器和低延迟中断响应时间。

该处理器提供的系统外设如下所示：

- 内部总线矩阵连接 AHB-Lite 接口，调试访问端口(DAP)
- 嵌套向量中断控制器(NVIC)
- 可选唤醒中断控制器(WIC)
- 断点和观察点单元(BPU)
- 串行线调试端口(SW-DP)

欲了解内核相关信息，请参考 Arm® Cortex®-M0+技术参考手册。

2.2 总线架构

本微控制器系统包含：

- 1 个主设备：
  - ARM® Cortex®-M0+内核
- 多个从设备：
  - 片上 SRAM
  - 片上 FLASH
  - AHB 外设
  - 4 个 AHB 到 APB 桥及 APB 总线上的外设

主从设备通过 AHB 总线矩阵进行连接，总线架构如下图所示：

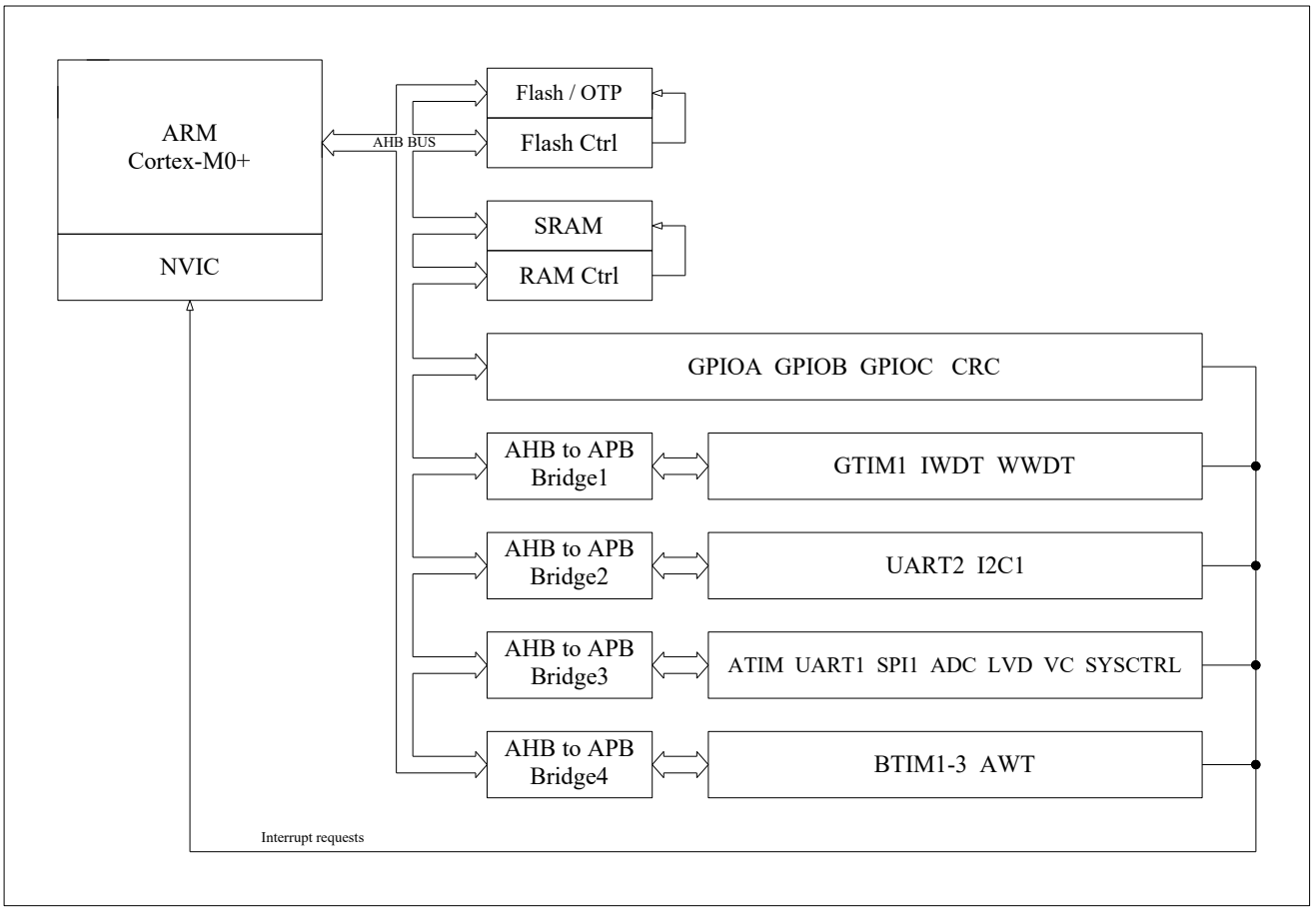

图　2-1 总线架构图

- AHB 总线

管理 M0+微处理器对 FLASH、SRAM 以及所有外设的访问存取仲裁。

- AHB 到 APB 桥 1/2/3/4

提供 AHB 总线到 APB1/APB2/APB3/APB4 总线的完全同步的连接，即 AHB 和 APB 总线的桥接。

2.3 存储器体系

2.3.1 概述

本芯片内核为 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。存储器以小端格式进行组织，即最低字节数据为数据的最低有效位，最高字节为数据的最高有效位。

例：
将 0x11223344 存放在地址为 0x2000 0000 的存储器空间中，实际存放结果是：
0x2000 0000 字节存放 0x44，
0x2000 0001 字节存放 0x33，
0x2000 0002 字节存放 0x22，
0x2000 0003 字节存放 0x11。

2.3.2 存储器映射

表 2-1 存储器映射表

名称	边界地址	大小
FLASH 存储器	0x0000 0000 - 0x0000 4FFF	20KB
OTP 存储器	0x0010 0770 - 0x0010 0785	22B
物理参数存储器	0x0010 0786 - 0x0010 07FF	122B
启动程序存储器	0x0010 0000 - 0x0010 073F	1.8KB
SRAM 存储器	0x2000 0000 - 0x2000 0BFF	3KB

2.3.2.1 FLASH 存储器

本芯片提供 20KB 的主 FLASH 存储器，支持字节、半字和字访问操作，该区域可用于存储固件或数据。可以通过使用在系统编程(ISP)、在应用编程(IAP)或在电路编程(ICP)来改写 FLASH 存储器内的数据。关于 FLASH 的详细介绍，请参考 FLASH 存储器章节。

2.3.2.2 OTP 存储器

本芯片提供 22B 的 OTP 存储器；仅支持读取与写入操作，不支持擦除操作。仅支持通过 ISP 协议向该区域写入数据；一旦写入不可擦除及改写。可用于存储不可更改的信息，如自定义的产品 ID 号、身份验证信息等。

2.3.2.3 物理参数存储器

本芯片具有 122B 的物理参数存储器，仅支持读取操作。该区域存放了内置模拟模块的校准值（HSI.TRIM、LSI.TRIM……）及数字签名数据。

2.3.2.4 启动程序存储器

当芯片复位后的 150us 内，若 SWDIO 管脚未检测到持续的 50kHz 的方波则执行 FLASH 存储区的用户程序。当芯片复位后的 150us 内，若 SWDIO 管脚检测到持续的 50kHz 的方波则执行启动程序存储区的 Bootloader 程序，用户可使用 ISP 协议操作 FLASH 存储器和 OTP 存储器。FLASH 存储器支持读、写、擦等操作；OTP 存储器支持读、写操作。

2.3.3 寄存器映射

表 2-2 外设地址映射表

名称	边界地址	大小	对应外设
APB1外设	0x4000 0400 - 0x4000 07FF	1KB	GTIM1
	0x4000 2C00 - 0x4000 2FFF	1KB	WWDT
	0x4000 3000 - 0x4000 33FF	1KB	IWDT
APB2外设	0x4000 4400 - 0x4000 47FF	1KB	UART2
	0x4000 5400 - 0x4000 57FF	1KB	I2C1
APB3外设	0x4001 0000 - 0x4001 03FF	1KB	SYSCTRL
	0x4001 2400 - 0x4001 27FF	1KB	ADC
	0x4001 2800 - 0x4001 2BFF	1KB	VC / LVD
	0x4001 2C00 - 0x4001 2FFF	1KB	ATIM
	0x4001 3000 - 0x4001 33FF	1KB	SPI1
	0x4001 3800 - 0x4001 3BFF	1KB	UART1
APB4外设	0x4001 4800 - 0x4001 4BFF	1KB	BTIM1 - 3
	0x4001 4C00 - 0x4001 4FFF	1KB	AWT
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
	0x4800 0400 - 0x4800 07FF	1KB	GPIOB
	0x4800 0800 - 0x4800 0BFF	1KB	GPIOC
M0+外设	0xE000 0000 - 0xE00F FFFF	1MB	M0+内核外设

2.3.3.1 AHB 外设

系统复位后，AHB 外设时钟总是关闭的，需通过 SYSCTRL_AHBEN 寄存器使能相应外设的时钟后方可访问这些外设的寄存器。用户还可通过 SYSCTRL_AHBRST 寄存器复位相应的外设。

2.3.3.2 APB 外设

系统复位后，APB 外设时钟总是关闭的，需通过 SYSCTRL_APBENx 寄存器使能相应外设的时钟后方可访问这些外设的寄存器。用户还可通过 SYSCTRL_APBRSTx 寄存器复位相应的外设。

3 电源控制与功耗

3.1 概述

本芯片由两组电源（DVCC-DVSS、AVCC-AVSS）进行供电；封装时 DVCC 与 AVCC 接到同一个管脚 VCC，DVSS 与 AVSS 接到同一个管脚 VSS。模拟电源 AVCC 主要为模拟模块供电；数字电源 DVCC 主要为数字模块供电。内置稳压器输出约 1.5V 的内核电压，供数字电路及部分振荡器使用。内置 BGR 输出三路精准的参考电压供模拟模块使用，这三个电压分别为 1.2V、1.5V、2.5V。

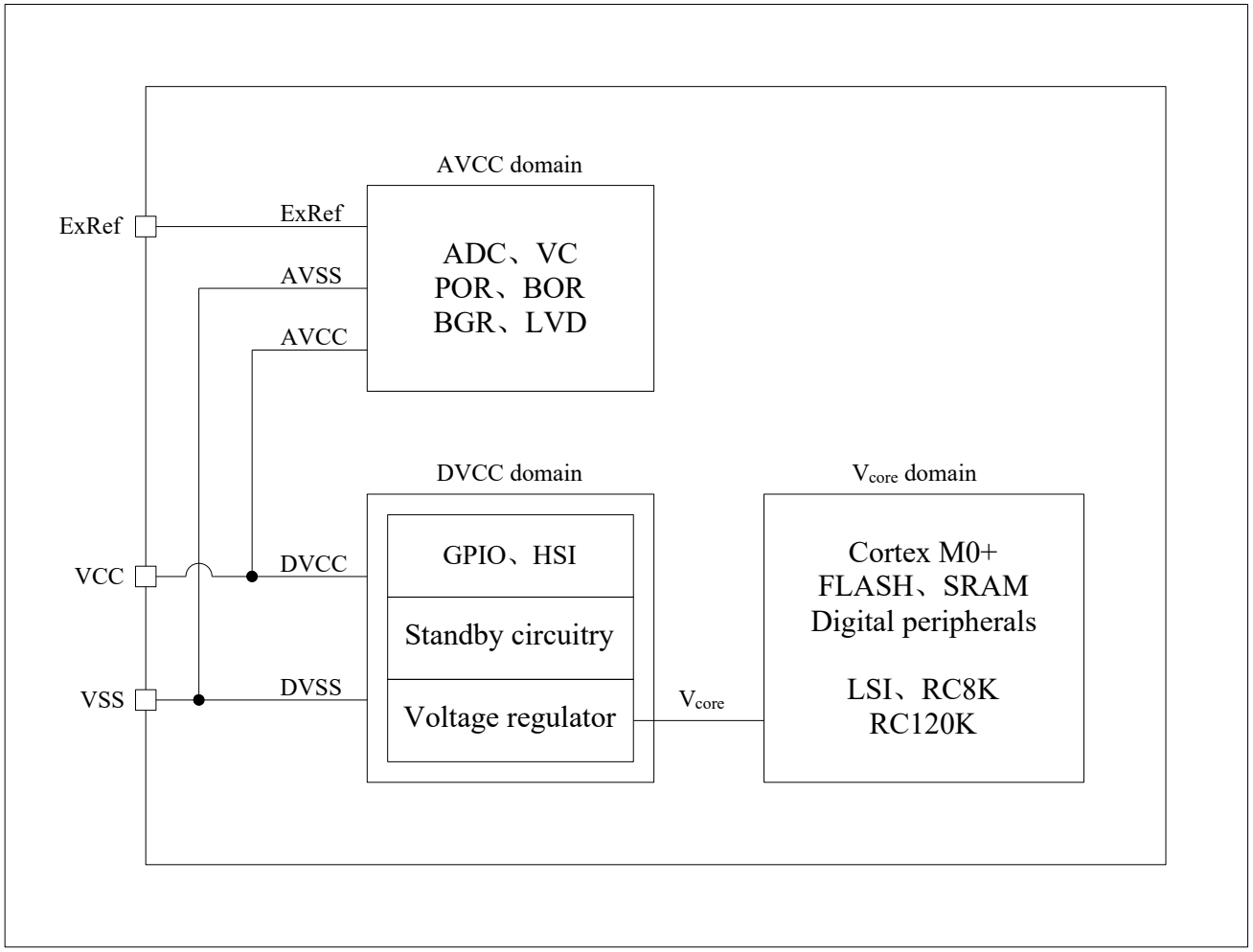

图　3-1 电源分配图

3.2 电源监控

3.2.1 上电复位/掉电复位(POR/BOR)

本芯片内部集成上电复位（POR）和掉电复位（BOR）电源监控电路。POR/BOR 同时监控电源电压；当监测到电源电压低于 V_{BOR} 阈值时系统会进入复位状态；当监测到电源电压高于 V_{POR} 阈值时系统会进入运行状态。

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：

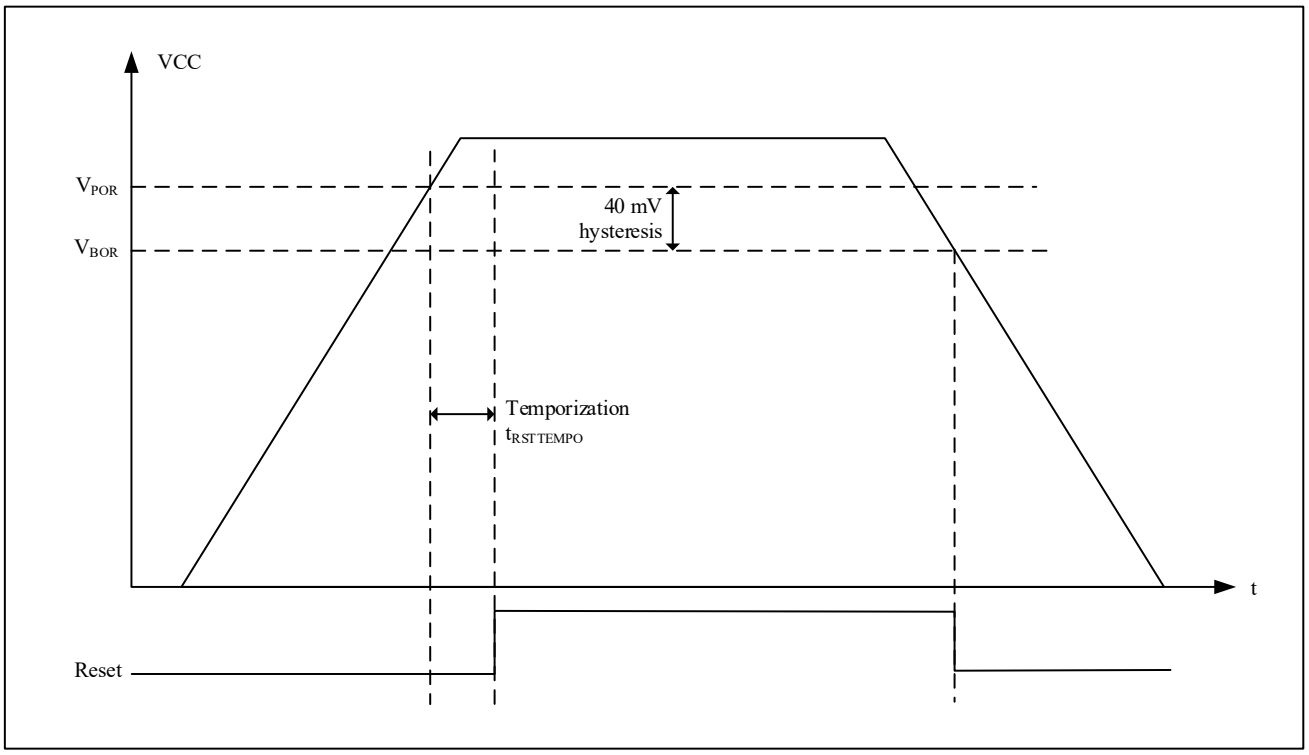

图 3-2 上电复位掉电复位时序图

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

3.2.2 低电压检测器(LVD)

本芯片内部集成低电压检测器(LVD)，可持续监测电源的电压或输入到 GPIO 的信号的电压。当监测结果符合触发条件时，将根据用户配置产生系统复位或中断请求。

低电压检测器（LVD）可通过寄存器配置监测目标、复位门限电压、数字滤波等参数。

3.3 工作模式

本产品在电源管理模块的控制下可实现三种工作模式。

各工作模式下可工作的功能模块不相同，如下所示：

- 运行模式（Active Mode）：　　CPU 运行，片内所有外设运行。
- 休眠模式（Sleep Mode）：　　CPU 停止，片内所有外设运行。
- 深度休眠模式（Deep Sleep Mode）：　CPU 停止，低功耗片内外设运行。

本产品工作时可在三种工作模式之间自由切换。在运行模式下执行 WFI 指令，可使本产品进入休眠模式或深度休眠模式。从休眠模式或深度休眠模式通过中断唤醒，回到运行模式。当系统进入休眠模式及深度休眠模式时，GPIO 管脚的状态与运行模式下相同。在深度休眠模式下，高速时钟（HEX、HSI）自动停止，低速时钟（LSI、RC8K、RC120K）保持原状态不变。

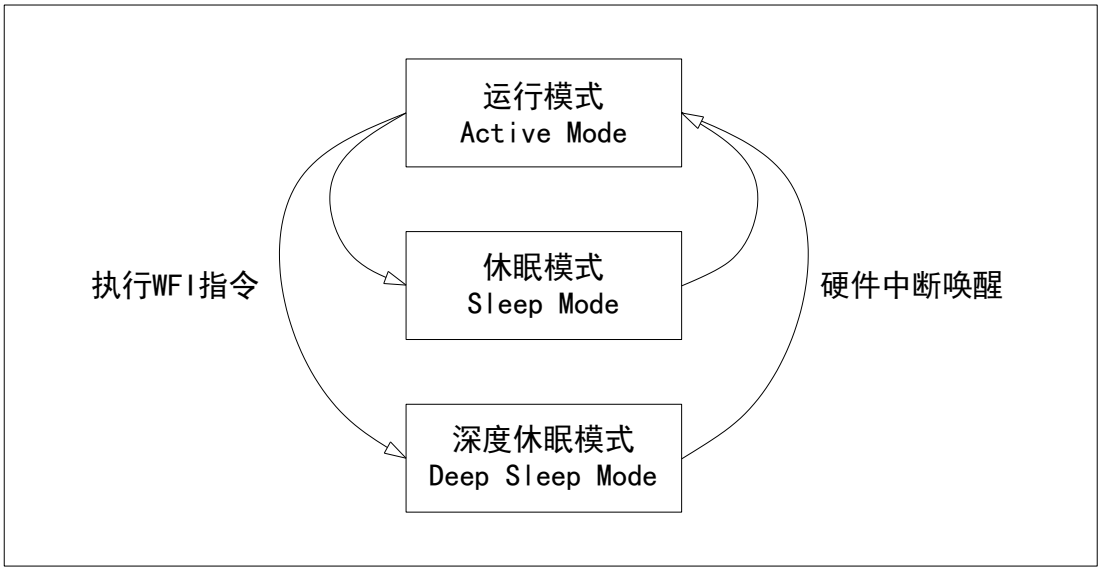

图　3-3 工作模式状态转换图

注意 1：进入深度休眠模式前应当将 HCLK 的时钟频率修改为不大于 4MHz。
注意 2：若使能了 VCx，则需等待 VCx_ISR.Ready 标志置 1 后才可以进入深度休眠模式。
注意 3：进入深度休眠模式前应当等待 FLASH_CR1.BUSY 标志变为 0 并设置 FLASH_CR1.MODE 为 0。

3.3.1 进入休眠模式或深度休眠模式

使用 M0+内核的 WFI 指令，配合 M0+内核的 SCR.SLEEPONEXIT 和 SCR.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。进入深度休眠模式前应当将 HCLK 的时钟频率修改为不大于 4MHz；若使能了 VCx，则需等待 VCx_ISR.Ready 标志变为 1；若 FLASH 正在进行擦写操作，则需等待 FLASH_CR1.BUSY 标志变为 0 并设置 FLASH_CR1.MODE 为 0。

进入休眠及深度休眠的详细方法如下表所示。

表 3-1 进入低功耗模式配置表

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

进入深度休眠模式时，系统将自动关闭高速时钟（HSI、HEX）；如需在深度休眠模式下使部分外设仍保持运行，则需要在进入深度休眠模式前，将外设的工作时钟设置为低速时钟并使能该低速时钟。

部分外设在深度休眠模式下可正常工作，具体如下所示：

- GPIOx，可产生中断；需配置滤波时钟为 NA、LSI、RC120K、RC8K；
- VCx，可正常工作并产生中断，需配置滤波时钟为 NA、RC120K；
- LVD，可正常工作并产生中断，需配置滤波时钟为 NA、RC120K；
- AWT，可正常工作并产生中断，需配置计数时钟为 LSI；
- IWDT，可正常工作并产生中断，需关闭低功耗暂停功能；
- UARTx，可正常接收并产生中断，需配置传输时钟为 LSI。

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。如果用户在中断服务程序中执行 WFI 命令进入休眠或深度休眠，则需要比当前中断更高优先级的中断才能唤醒 CPU；因此，建议用户不要在中断服务程序中执行 WFI 指令。在各种工作模式下，CPU 可响应的中断源如下表所示。

表 3-2 工作模式及中断响应对照表

中断源	运行模式	休眠模式	深度休眠模式
GPIOx	●	●	●
VCx	●	●	●
LVD	●	●	●
AWT	●	●	●
IWDT	●	●	●
UARTx	●	●	●
SYSCTRL	●	●	x
FLASH	●	●	x
RAM	●	●	x
BTIMx	●	●	x
GTIMx	●	●	x
ATIM	●	●	x
WWDT	●	●	x
I2Cx	●	●	x
SPIx	●	●	x
ADC	●	●	x

只有当外设的中断被允许时，该外设的中断信号才可唤醒休眠模式或深度休眠模式的 CPU。若使能了全局中断向量，则唤醒后 CPU 将立即执行相应的中断服务程序。若禁止全局中断向量，则唤醒后 CPU 将执行 WFI 指令的下一条语句；用户需使用代码清除内核的中断挂起状态。当禁止全局中断向量时 CPU 无需进行中断操作，可以更快执行用户代码，适用于对唤醒速度有要求的场景。

进入深度休眠模式时，系统将自动关闭高速时钟源（HSI、HEX）；在退出深度休眠模式时系统自动使能被关闭的高速时钟源（HIS、HEX）。HSI 时钟的启动时间仅数微秒，使用 HSI 作为唤醒后的时钟可实现快速唤醒执行任务。

3.4 功耗控制

- 当系统工作于运行模式时，可通过一定的措施降低运行功耗：
  - 设置 SYSCTRL_CR0.SYSCLK，降低系统时钟（SysClk）的频率
  - 设置 SYSCTRL_CR0.HCLKPRS，降低内核时钟（HCLK）的频率
  - 设置 SYSCTRL_CR0.PCLKPRS，降低外设时钟（PCLK）的频率
  - 设置 SYSCTRL_AHBENx、SYSCTRL_APBENx，关闭未使用的外设的时钟

- 当系统工作于休眠模式时，可通过一定的措施降低运行功耗：
  - 设置 SYSCTRL_CR0.SYSCLK，降低系统时钟（SysClk）的频率
  - 设置 SYSCTRL_CR0.HCLKPRS，降低内核时钟（HCLK）的频率
  - 设置 SYSCTRL_CR0.PCLKPRS，降低外设时钟（PCLK）的频率
  - 设置 SYSCTRL_AHBENx、SYSCTRL_APBENx，关闭未使用的外设的时钟
  - 设置 SCB->SCR.SLEEP-ON-EXIT，使 CPU 退出中断服务程序后自动进入休眠模式

- 当系统工作于深度休眠模式时，可通过一定的措施降低运行功耗：
  - 设置 SYSCTRL_CR0.SYSCLK 为高速时钟，进入深度休眠时被硬件自动关闭
  - 设置 SYSCTRL_APBENx，关闭无需在低功耗模式下工作的外设的时钟
  - 设置 SCB->SCR.SLEEP-ON-EXIT，使 CPU 退出中断服务程序后自动进入休眠模式

3.5 寄存器描述

SCB_SCR 内核系统控制寄存器

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，每次新的中断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	设为0时，执行WFI时，本产品进入普通休眠模式 设为1时，执行WFI时，本产品进入深度休眠模式
1	SLEEPONEXIT	RW	设置为0时，该特性就会被自动禁止 设为1时，当退出异常处理并返回程序线程时，处理器自动进入休眠模式(WFI)
0	RFU	-	保留位，请保持默认值

4 系统控制（SYSCTRL）

4.1 系统复位

本芯片支持以下 8 种系统复位，各复位源的复位范围略有不同。

复位源	复位范围
上电复位 POR	整个 MCU
掉电复位 BOR	整个 MCU
复位管脚复位 NRST	整个 MCU
独立看门狗复位 IWDT	M0+内核 / 外设（除 RAM 控制器外）
窗口看门狗复位 WWDT	M0+内核 / 外设（除 RAM 控制器外）
低电压检测器复位 LVD	M0+内核 / 外设（除 LVD 外）
内核 SYSRESETREQ 复位	M0+内核 / 外设（除 RAM 控制器和 LVD 外）
内核 LOCKUP 故障复位	M0+内核 / 外设（除 RAM 控制器和 LVD 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。复位完成后，用户可通过 SYSCTRL_RESETFLAG 寄存器查询本次系统复位的复位源；完成复位标志查询后需将该寄存器清零以便下次复位后查询复位源。

4.1.1 上电复位 POR

本芯片集成了专用的 POR 电路；在电源电压上升到 POR 阈值前芯片将保持在复位状态，以防止芯片在上电过程中出现误动作。

4.1.2 掉电复位 BOR

本芯片集成了专用的 BOR 电路；在电源电压下降到 BOR 阈值后芯片将保持在复位状态，以防止芯片在下电过程中出现误动作。

4.1.3 复位管脚复位 NRST

本芯片具有专门的复位输入管脚，输入一定宽度的低电平脉冲会引起系统复位。芯片内置了专用数字滤波电路，小于 20us 的低电平脉冲信号可能会被屏蔽；因此，施加到复位管脚上的低电平信号应大于 20us 以确保芯片可靠复位。

复位管脚内置了约 8kΩ 的上拉电阻，当用户在管脚外部连接上拉电阻时需考虑内部上拉电阻的影响。

4.1.4 IWDT 复位

本芯片集成了独立看门狗 IWDT，该看门狗采用专用的 RC8K 时钟进行计数，使 IWDT 可独立可靠地工作。用户需在 IWDT 溢出前对 IWDT 进行重载，否则 IWDT 将溢出并产生复位信号以复位整个芯片。

4.1.5 WWDT 复位

本芯片集成了窗口看门狗 WWDT，用户需要在特定的时间窗口内对 WWDT 进行重载，否则 WWDT 将产生复位信号以复位整个芯片。

4.1.6 LVD 低电压检测复位

本芯片集成了低电压检测器(LVD)，可持续监测电源的电压或输入到 GPIO 的信号的电压。当监测结果符合触发条件时，将根据用户配置产生系统复位或中断请求。
低电压检测器（LVD）通过寄存器配置监测目标、复位门限电压、数字滤波等参数。

4.1.7 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。

4.1.8 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 复位功能处于禁止状态，通过设置 SYSCTRL_CR2.LOCKUP 位域为 1 以使能 LOCKUP 复位功能。

4.2 外设复位

本芯片集成外设软复位功能，用户可使用软件对指定的外设进行复位，以使外设的寄存器及状态恢复到上电复位时状态。该功能由外设模块的复位寄存器控制，当设置相应的位域为 0 时，外设处于复位状态；当设置相应的位域为 1 时，外设处于工作状态。外设复位相关的三个寄存器分别为：SYSCTRL_AHBRST、SYSCTRL_APBRST1、SYSCTRL_APBRST2；具体每个外设的复位信号详见这三个寄存器位域的定义。

4.3 时钟源

本芯片内置多个时钟源产生电路，通过分频器和多路选择器提供丰富多样的时钟给 CPU 及各种片内外设使用。

4.3.1 时钟树

以下 3 个时钟源可以被配置为系统时钟：

- 外部输入时钟 HEX，可输入 4～32MHz 的时钟
- 内部高速 RC 振荡器时钟 HSI， 可生成 3～48MHz 的时钟
- 内部低速 RC 振荡器时钟 LSI， 可生成 32.768kHz 的时钟

以下 2 个时钟源可以被配置为外设的工作时钟或滤波时钟：

- 内部低速 RC 振荡器时钟 RC120K， 可生成约 120kHz 的时钟
- 内部低速 RC 振荡器时钟 RC8K， 可生成约 8.6kHz 的时钟

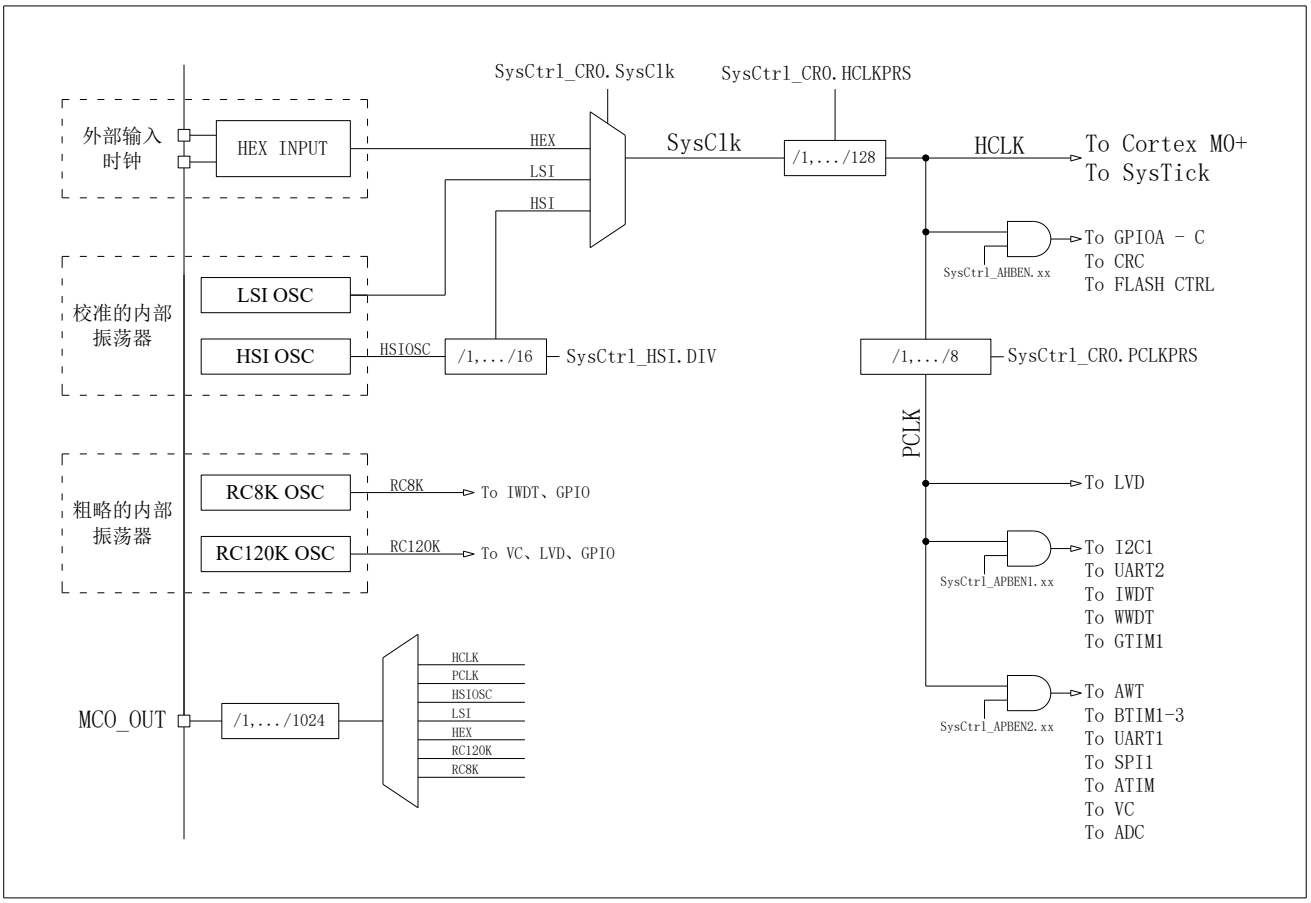

图 4-1 时钟树框图

4.3.2 HEX 时钟

外部输入时钟信号（HEX）是从芯片管脚输入的时钟信号。
通过 HEX.PINMUX 位域可选择输入的时钟信号是来自 PB00 还是 PB01；作为输入时钟信号的管脚需要设置为数字输入模式；未使用的另一只管脚可作为通用 GPIO。通过 HEX.PINEN 位域可使能或禁止管脚上的时钟信号输入到 HEX 电路。
从芯片管脚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，信号频率在 4 - 32MHz 之间。
使能 HEX 电路后，当从管脚输入的时钟信号的数量达到的 HEX.WAITCYCLE 的要求，HEX.STABLE 及 ISR.HEXRDY 标志被硬件置 1；若使能 HEX 起振稳定中断（IER.HEXRDY=1），则 CPU 会执行 HEX 起振稳定中断服务程序。

4.3.3 HSI 时钟

本芯片内部集成高频 RC 振荡器，上电后默认输出频率为 48MHz 的 HSIOSC 时钟信号；该时钟信号经过分频后输出频率为 3～48 MHz 的 HSI 时钟信号。
更改 HSI.TRIM 位域的数值即可微调 HSIOSC 时钟信号的频率：该位域的数值每增加 1 则 HSIOSC 时钟信号的频率增加约 0.2%。该 RC 振荡器输出频率的安全工作范围为 40～50MHz，通过调整该寄存器的数值即可实现输出非 48MHz 的时钟信号。
当芯片从 DeepSleep 状态唤醒时，HSIOSC 仅需 4~5us 即可完成起振并驱动 CPU 开始工作。建议在进入 DeepSleep 前将 SYSCLK 的来源切换为 HSI(4MHz)以实现快速唤醒并处理唤醒事件。

4.3.4 LSI 时钟

本芯片内部集成低频 RC 振荡器，上电后默认输出频率为 32.768kHz 的 LSI 时钟信号。
更改 LSI.TRIM 位域的数值即可微调 LSI 时钟信号的频率：该位域的数值每增加 1 则 LSI 时钟信号的频率增加约 0.4%。该 RC 振荡器输出频率的安全工作范围为 30～36kHz，通过调整该寄存器的数值即可实现输出非 32.768kHz 的时钟信号。

4.3.5 RC120K 时钟

本芯片内部集成低频 RC 振荡器，上电后默认输出频率约 120kHz 的 RC120K 时钟信号。
该时钟可用于 GPIO、LVD、VC 的滤波，开启滤波功能后该时钟自动使能。

4.3.6 RC8K 时钟

本芯片内部集成低频 RC 振荡器，上电后默认输出频率约 8.6kHz 的 RC8K 时钟信号。
该时钟可用于 GPIO 的滤波和 IWDT 的计数，开启 GPIO 滤波或使能 IWDT 后该时钟自动使能。

4.4 时钟控制

4.4.1 工作模式与振荡器使能

当系统工作于运行模式和休眠模式时，各时钟振荡器的使能由 SYSCTRL_CR1 相关位域控制。
当系统工作于深度休眠模式时，高频时钟 HEX、HSI 和 PLL 被自动关闭以降低系统功耗；低频时钟 LSI、RC120K、RC8K 保持原工作状态，采用低频时钟作为工作时钟的外设在深度休眠模式下可以正常工作，如下所示：

- GPIOx，可产生中断；需配置滤波时钟为 LSI、RC120K、RC8K；
- VCx，可正常工作并产生中断，需配置滤波时钟为 RC120K；
- LVD，可正常工作并产生中断，需配置滤波时钟为 RC120K；
- AWT，可正常工作并产生中断，需配置计数时钟为 LSI；
- IWDT，可正常工作并产生中断，需关闭低功耗暂停功能；
- UARTx，可正常接收并产生中断，需配置传输时钟为 LSI；

4.4.2 内核时钟控制

内核时钟 HCLK 来自系统时钟 SYSCLK 的 1～128 分频，由 SYSCTRL_CR0.HCLKPRS 配置。
SYSCLK 时钟来自 HSI、HEX、LSI，由 SYSCTRL_CR0.SYSCLK 配置。
当时钟源使能并稳定后才可被配置为 SYSCLK 的来源，建议关闭未使用的时钟源。

4.4.3 外设时钟控制

外设时钟 PCLK 来自内核时钟 HCLK 的 1～8 分频，由 SYSCTRL_CR0.PCLKPRS 配置。
片内外设的工作时钟及配置时钟可以通过寄存器使能或禁止；当需要外设工作时，必须使能该外设的工作及配置时钟；当不需要外设工作时，可关闭该外设的工作及配置时钟以减小芯片的整体功耗。

- 通过 SYSCTRL_AHBEN 寄存器可使能或禁止 AHB 外设的时钟：
  - FLASH、CRC、GPIOA、GPIOB、GPIOC
- 通过 SYSCTRL_APBEN1 寄存器可使能或禁止 APB 外设的时钟：
  - GTIM1、WWDT、IWDT、UART2、I2C1
- 通过 SYSCTRL_APBEN2 寄存器可使能或禁止 APB 外设的时钟：
  - ADC、VC、ATIM、SPI1、UART1、BTIM1-3、AWT

4.4.4 输出片内时钟信号

本芯片的管脚通过适当配置可输出各种片内时钟信号以方便调试。

- HCLK_OUT 管脚可以输出 HCLK 信号；
- PCLK_OUT 管脚可以输出 PCLK 信号；
- MCO_OUT 管脚可以输出 HCLK、PCLK、HSIOSC、LSI、HEX、RC120K、RC8K 时钟信号，并可配置输出信号的分频比为 1 ~ 1024。

4.5 系统时钟切换

复位后完成时，系统时钟 SysClk 的默认来源为 HSI。用户可根据实际需要修改 SysClk 来源为 HEX、HSI 或 LSI，SysClk 来源由 SYSCTRL_CR0.SYSCLK 位域控制。

4.5.1 标准的时钟切换流程

切换系统时钟来源需按照特定的操作步骤进行，如下所示：

Step1. 如新时钟源需要外部引脚，则需将该引脚设置为适当的模式；

Step2. 配置新时钟源的振荡参数；

Step3. 设置 SYSCTRL_CR1 寄存器的相应位域为 1，使能新时钟源振荡器；

Step4. 等待新时钟源输出稳定的时钟信号（STABLE 标志变为 1）；

Step5. 根据当前时钟源和新时钟源两者中较高的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step6. 修改 SYSCTRL_CR0.SYSCLK 位域，选择新时钟源作为 SysClk 的来源；

Step7. 根据新时钟源的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step8. 关闭不再使用的时钟源。

注意：SYSCTRL 及 FLASH 的部分寄存器的高位需要写入 KEY 才能改写寄存器的值。

4.5.2 HSI 时钟不同频率间切换示例

HSI 时钟不同频率间切换操作流程如下：

Step1. 根据当前 HSI 频率和目标 HSI 频率中较高的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step2. 修改 SYSCTRL_HSI.DIV 位域，以调整 HSIOSC 与 HSI 的分频比；

Step3. 根据当前 HSI 的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域。

4.5.3 从其它时钟源切换到 HEX 示例

从其它时钟源到切换到 HEX 操作流程如下：

Step1. 设置 PB00 或 PB01 管脚为数字输入模式；

Step2. 配置 SYSCTRL_HEX.PINMUX 位域，选择输入时钟信号的管脚；

Step3. 配置 SYSCTRL_HEX.WITCYCLE，选择 HEX 输入信号的稳定计数值；

Step4. 设置 SYSCTRL_HEX.PINEN 为 1，使能从管脚输入时钟信号；

Step5. 向 SYSCTRL_CR1 写入（SYSCTRL_CR1 | 0x5A5A0002），使能 HEX 电路；

Step6. 查询等待 SYSCTRL_HEX.STABLE 标志变为 1，HEX 电路输出稳定的时钟；

Step7. 根据当前时钟频率和 HEX 时钟频率中较高的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step8. 向 SYSCTRL_CR0 写入 0x5A5A0001，将 SysClk 时钟来源切换为 HEX；

Step9. 根据 HEX 时钟频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step10. 向 SYSCTRL_CR1 写入 0x5A5A0002，关闭除 HEX 外的其它时钟。

4.5.4 从其它时钟源切换到 LSI 示例

从其它时钟源到切换到 LSI 操作流程如下：

Step1. 从 FLASH 中读取 LSI 的 TRIM 值并写入 SYSCTRL_LSI.TRIM 位域；

Step2. 设置 SYSCTRL_LSI.WAITCYCLE 为 3，设定 LSI 稳定时间为 258 周期；

Step3. 向 SYSCTRL_CR1 写入（SYSCTRL_CR1 | 0x5A5A0008），使能 LSI 振荡电路；

Step4. 查询等待 SYSCTRL_LSI.STABLE 标志变为 1，LSI 振荡电路输出稳定的时钟；

Step5. 向 SYSCTRL_CR0 写入 0x5A5A0003，将 SysClk 时钟来源切换为 LSI；

Step6. 向 FLASH_CR2 写入 0x5A5A0000，设置 FLASH 访问速度为小于 24MHz；

Step7. 向 SYSCTRL_CR1 写入 0x5A5A0008，关闭除 LSI 外的其它时钟。

4.5.5 从其它时钟源切换到 HSI 示例

从其它时钟源到切换到 HSI 操作流程如下：

Step1. 从 FLASH 中读取 HSI 的 TRIM 值并写入 SYSCTRL_HSI.TRIM 位域；

Step2. 根据需要配置 SYSCTRL_HSI.DIV，设置 HSIOSC 与 HSI 的分频比；

Step3. 向 SYSCTRL_CR1 写入（SYSCTRL_CR1 | 0x5A5A0001），使能 HSI 振荡电路；

Step4. 查询等待 SYSCTRL_HSI.STABLE 标志变为 1，HSI 振荡电路输出稳定的时钟；

Step5. 根据当前时钟频率和 HSI 时钟频率中较高的频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step6. 向 SYSCTRL_CR0 写入 0x5A5A0000，将 SysClk 时钟来源切换为 HSI；

Step7. 根据 HSI 时钟频率，按照闪存存储器章节要求合理配置 FLASH_CR2.WAIT 位域；

Step8. 向 SYSCTRL_CR1 写入 0x5A5A0001，关闭除 HSI 外的其它时钟。

4.6 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
SYSCTRL_CR0	0x40010000 + 0x00	系统控制寄存器0
SYSCTRL_CR1	0x40010000 + 0x04	系统控制寄存器1
SYSCTRL_CR2	0x40010000 + 0x08	系统控制寄存器2
SYSCTRL_IER	0x40010000 + 0x0C	系统中断使能控制寄存器
SYSCTRL_ISR	0x40010000 + 0x10	系统中断标志寄存器
SYSCTRL_ICR	0x40010000 + 0x14	系统中断标志清除寄存器
SYSCTRL_HSI	0x40010000 + 0x18	内置高频时钟控制寄存器
SYSCTRL_HEX	0x40010000 + 0x1C	外置高频晶体控制寄存器
SYSCTRL_LSI	0x40010000 + 0x20	内置低频时钟控制寄存器
SYSCTRL_PLL	0x40010000 + 0x28	内置锁相环控制寄存器
SYSCTRL_DEBUGEN	0x40010000 + 0x2C	调试状态定时器控制寄存器
SYSCTRL_AHBEN	0x40010000 + 0x30	AHB外设时钟使能控制寄存器
SYSCTRL_APBEN2	0x40010000 + 0x34	APB外设时钟使能控制寄存器2
SYSCTRL_APBEN1	0x40010000 + 0x38	APB外设时钟使能控制寄存器1
SYSCTRL_AHBRST	0x40010000 + 0x40	AHB外设复位控制寄存器
SYSCTRL_APBRST2	0x40010000 + 0x44	APB外设复位控制寄存器2
SYSCTRL_APBRST1	0x40010000 + 0x48	APB外设复位控制寄存器1
SYSCTRL_RESETFLAG	0x40010000 + 0x4C	系统复位标志寄存器
SYSCTRL_GTIM1CAP	0x40010000 + 0x50	通用定时器1输入捕捉来源配置寄存器
SYSCTRL_ATIMETR	0x40010000 + 0x60	高级定时器ETR来源配置寄存器
SYSCTRL_GTIMETR	0x40010000 + 0x64	通用定时器ETR来源配置寄存器
SYSCTRL_ATIMGATE	0x40010000 + 0x68	高级定时器GATE来源配置寄存器
SYSCTRL_TIMITR	0x40010000 + 0x6C	定时器ITR来源配置寄存器
SYSCTRL_MCO	0x40010000 + 0x70	系统时钟输出控制寄存器

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
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HEX 011：设置SysClk的时钟源为LSI

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
3	LSIEN	RW	内部低速时钟LSI使能控制 0：关闭 1：使能
2	RFU	-	保留位，请保持默认值
1	HEXEN	RW	外部输入时钟HEX使能控制 0：关闭 1：使能
0	HSIEN	RW	内部高速时钟HSIOSC使能控制 0：关闭 1：使能

注意：

进入 DeepSleep 时，高速时钟（HSI/HEX）自动关闭；低速时钟（LSI）保持当前状态。

SYSCTRL_CR2 系统控制寄存器 2

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
15:4	RFU	-	保留位
3	WAKEUPCLK	RW	DeepSleep唤醒时，系统时钟的来源配置 0：唤醒后时钟源为原系统时钟源 1：唤醒后时钟源为HSI，原系统时钟源保持使能 注意：若使能该功能，则在进入DeepSleep前应配置好HSI的输出频率以及FLASH_CR2.WAIT位域的值。
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：PA02、PA05作为SWD端口，GPIO功能不可用。 1：PA02、PA05作为GPIO端口，SWD功能不可用。
0	RSTIO	RW	RST端口功能配置 0：PC05作为RESET端口，GPIO功能不可用。 1：PC05作为GPIO端口，RESET功能不可用。 注意：对本寄存器进行任何写操作都会导致此位被锁定。锁定后此位无法修改，只有POR上电可以解除此位的锁定。

SYSCTRL_HSI 内置高频时钟控制寄存器

Address offset: 0x18 Reset value: 0x0000 ----

位域	名称	权限	功能描述
31:16	RFU	-	保留位
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 / 6 0110：HSI = HSIOSC / 1 1000：HSI = HSIOSC / 2 1001：HSI = HSIOSC / 4 1010：HSI = HSIOSC / 6 1011：HSI = HSIOSC / 8 1100：HSI = HSIOSC / 10 1101：HSI = HSIOSC / 12 1110：HSI = HSIOSC / 14 1111：HSI = HSIOSC / 16
10:0	TRIM	RW	时钟频率调整，更改该寄存器位域的数值即可调整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 40 - 50MHz。从FLASH中读出预设的校准值，写入到本位域即可获得精准的HSIOSC频率。 校准值存储地址：0x0010 07B8 - 0x0010 07B9

SYSCTRL_LSI 内置低频时钟控制寄存器

Address offset: 0x20 Reset value: 0x0000 ----

位域	名称	权限	功能描述
31:16	RFU	-	保留位
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
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，更改该寄存器位域的数值即可调整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。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。 校准值存储地址：0x0010 07BA - 0x0010 07BB

SYSCTRL_HEX 外部输入时钟控制寄存器

Address offset: 0x1C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:20	RFU	-	保留位
19	STABLE	RO	HEX时钟稳定状态位 0：HEX时钟尚未稳定 1：HEX时钟已经稳定
18:8	RFU	-	保留位
7	PINMUX	RW	外部时钟输入管脚来源配置 0：外部输入时钟来自PB00管脚 1：外部输入时钟来自PB01管脚
6	PINEN	RW	外部时钟输入管脚使能控制 0：禁止管脚上的时钟信号输入到HEX电路 1：允许管脚上的时钟信号输入到HEX电路
5:4	WAITCYCLE	RW	HEX稳定时钟周期数量选择 00：等待8192个HEX时钟信号 01：等待32768个HEX时钟信号 10：等待131072个HEX时钟信号 11：等待262144个HEX时钟信号
3:0	RFU	-	保留位

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
15:4	RFU		保留位
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
1	HEXRDY	RW	HEX稳定中断使能控制 0：禁止HEX稳定中断 1：使能HEX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

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
31:15	RFU	-	保留位
14	LSISTABLE	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
13	RFU	-	保留位
12	HEXSTABLE	RO	HEX钟稳定状态位 0：HEX尚未稳定 1：HEX钟已经稳定
11	HSISTABLE	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
10:4	RFU	-	保留位
3	LSIRDY	RO	LSI时钟稳定标志位 0：LSI时钟尚未稳定 1：LSI时钟已经稳定
2	RFU	-	保留位
1	HEXDY	RO	HEX钟稳定标志位 0：HEX钟尚未稳定 1：HEX钟已经稳定
0	HSIRDY	RO	HSIOSC时钟稳定标志位 0：HSIOSC时钟尚未稳定 1：HSIOSC时钟已经稳定

SYSCTRL_ICR 系统中断标志清除寄存器

Address offset: 0x14 Reset value: 0x0000 000B

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3	LSIRDY	R1W0	LSI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
2	RFU	-	保留位
1	HEXDY	R1W0	HEX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时钟稳定标志位清零控制 W0：清除ISR寄存器中的相应标志 W1：无功能

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
31:7	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

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
31:12	RFU	-	保留位，请保持默认值
11	I2C1	RW	I2C1模块配置时钟及工作时钟使能控制 0：关闭 1：使能
10:8	RFU	-	保留位，请保持默认值
7	UART2	RW	UART2模块配置时钟使能控制 0：关闭 1：使能
6	RFU	-	保留位，请保持默认值
5	IWDT	RW	IWDT模块配置时钟使能控制 0：关闭 1：使能
4	WWDT	RW	WWDT模块配置时钟及工作时钟使能控制 0：关闭 1：使能
3:2	RFU	-	保留位，请保持默认值
1	GTIM1	RW	GTIM1模块配置时钟及工作时钟使能控制 0：关闭 1：使能
0	RFU	-	保留位，请保持默认值

SYSCTRL_APBEN2 APB 外设时钟使能控制寄存器 2

Address offset: 0x34 Reset value: 0x0000 8000

位域	名称	权限	功能描述
31:14	RFU	-	保留位，请保持默认值
13	AWT	RW	AWT模块配置时钟使能控制 0：关闭 1：使能
12	BTIM	RW	BTIM1-3模块配置时钟及工作时钟使能控制 0：关闭 1：使能
11:10	RFU	-	保留位，请保持默认值
9	UART1	RW	UART1模块配置时钟使能控制 0：关闭 1：使能
8	SPI1	RW	SPI1模块配置时钟及工作时钟使能控制 0：关闭 1：使能
7	ATIM	RW	ATIM模块配置时钟及工作时钟使能控制 0：关闭 1：使能
6:5	RFU	-	保留位，请保持默认值
4	VC	RW	VC模块配置时钟使能控制 0：关闭 1：使能
3	RFU	RW	保留位，请保持默认值
2	ADC	RW	ADC模块配置时钟及工作时钟使能控制 0：关闭 1：使能
1:0	RFU	-	保留位，请保持默认值

SYSCTRL_AHBRST AHB 外设复位控制寄存器

Address offset: 0x40 Reset value: 0x0000 0076

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

SYSCTRL_APBRST1 APB 外设复位控制寄存器 1

Address offset: 0x48 Reset value: 0x0000 08B3

位域	名称	权限	功能描述
31:12	RFU	-	保留位，请保持默认值
11	I2C1	RW	I2C1模块复位控制 0：模块处于复位状态 1：模块正常工作
10:8	RFU	-	保留位，请保持默认值
7	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
6	RFU	-	保留位，请保持默认值
5	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
4	WWDT	RW	WWDT模块复位控制 0：模块处于复位状态 1：模块正常工作
3:2	RFU	-	保留位，请保持默认值
1	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
0	RFU	-	保留位，请保持默认值

SYSCTRL_APBRST2 APB 外设复位控制寄存器 2

Address offset: 0x44　　Reset value: 0x0000 339C

位域	名称	权限	功能描述
31:14	RFU	-	保留位，请保持默认值
13	AWT	RW	AWT模块复位控制 0：模块处于复位状态 1：模块正常工作
12	BTIM	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
11:10	RFU	-	保留位，请保持默认值
9	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
8	SPI1	RW	SPI1模块复位控制 0：模块处于复位状态 1：模块正常工作
7	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
6:5	RFU	-	保留位，请保持默认值
4	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
3	RFU	-	保留位，请保持默认值
2	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
1:0	RFU	-	保留位，请保持默认值

SYSCTRL_RESETFLAG 系统复位标志寄存器

Address offset: 0x4C Reset value: 0x0000 0---

位域	名称	权限	功能描述
31:10	RFU	-	保留位
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
6	RSTB	RW0	NRST管脚复位标志 0：未发生管脚复位 1：已发生管脚复位 写0清除，写1无效
5	WWDT	RW0	WWDT复位标志，需要软件初始化及清除 0：未发生 WWDT 复位 1：已发生 WWDT 复位 写0清除，写1无效
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

SYSCTRL_DEBUGEN 调试状态定时器控制寄存器

Address offset: 0x2C Reset value: 0x0000 06E3

位域	名称	权限	功能描述
31:11	RFU	-	保留位
10	WWDT	RW	调试状态下，WWDT 计数功能配置 0：正常计数 1：暂停计数
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
8:7	RFU	-	保留位
6	AWT	RW	调试状态下，AWT 计数功能配置 0：正常计数 1：暂停计数
5	BTIM	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
4:2	RFU	-	保留位
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

SYSCTRL_GTIM1CAP 通用定时器 1 输入捕捉来源配置寄存器

Address offset: 0x50 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:15	RFU	-	保留位
14:12	CH4	RW	GTIM1的CH4输入捕捉来源配置 详见GTIM1AP[2:0]
11	RFU	-	保留位
10:8	CH3	RW	GTIM1的CH3输入捕捉来源配置 详见GTIM1AP[2:0]
7	RFU	-	保留位
6:4	CH2	RW	GTIM1的CH2输入捕捉来源配置 详见GTIM1AP[2:0]
3	RFU	-	保留位
2:0	CH1	RW	GTIM1CH1输入捕捉来源配置 000：由GPIOx_AFRH和GPIOx_AFRL配置 001：UART1的RXD信号 010：UART2的RXD信号 100：VC1的比较输出信号 101：VC2的比较输出信号 110：LVD的输出信号

SYSCTRL_ATIMETR 高级定时器 ETR 来源配置寄存器

Address offset: 0x60 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:3	RFU	-	保留位
2:0	ATIMETR	RW	ATIM的ETR来源配置 详见 GTIMETR[2:0]

SYSCTRL_GTIMETR 通用定时器 ETR 来源配置寄存器

Address offset: 0x64 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:3	RFU	-	保留位
2:0	GTIM1ETR	RW	GTIM1的ETR来源配置 000：由GPIOx_AFRH和GPIOx_AFRL配置 001：UART1的RXD信号 010：UART2的RXD信号 100：VC1的比较输出信号 101：VC2的比较输出信号 110：LVD的输出信号

SYSCTRL_ATIMGATE 高级定时器 GATE 来源配置寄存器

Address offset: 0x68 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:3	RFU	-	保留位
2:0	ATIMGATE	RW	ATIM的GATE来源配置 000：由GPIOx_AFRH和GPIOx_AFRL配置 001：UART1的RXD信号 010：UART2的RXD信号 100：VC1的比较输出信号 101：VC2的比较输出信号 110：LVD的输出信号

SYSCTRL_TIMITR 定时器 ITR 来源配置寄存器

Address offset: 0x6C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:24	RFU	-	保留位
23:21	BTIM3ITR	RW	BTIM3的ITR来源配置 详见 TIMITR [2:0]
20:18	BTIM2ITR	RW	BTIM2的ITR来源配置 详见 TIMITR [2:0]
17:15	BTIM1ITR	RW	BTIM1的ITR来源配置 详见 TIMITR [2:0]
14:6	RFU	-	保留位
5:3	GTIM1ITR	RW	GTIM1的ITR来源配置 详见 TIMITR [2:0]
2:0	ATIMITR	RW	ATIM的ITR来源配置 000：BTIM1的溢出信号 001：BTIM2的溢出信号 010：BTIM3的溢出信号 011：GTIM1的溢出信号 111：ATIM的主模式输出信号

注意：不可配置 TIM 的 ITR 来源为自己的溢出信号。

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
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：HEX 1000：RC120K 1001：RC8K

5 中断（INTERRUPT）

5.1 简介

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 IRQ0 和 IRQ1 均挂起时，并且两者的优先级相同，那么先处理 IRQ0。

5.4 中断向量表

本芯片中断向量表如下所示：

向量编号	外部中断号	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	0x0000 0008
3	-	-1	HardFault	硬件错误异常（fault）	0x0000 000C
4-10	-	-	-	-	-
11	-	可配置	SVCall	通过SWI指令调用的管理程序	0x0000 002C
12-13	-	-	-	-	-
14	-	可配置	PendSV	系统服务的可挂起请求	0x0000 0038
15	-	可配置	SysTick	系统节拍定时器	0x0000 003C
16	0	可配置	WDT	窗口看门狗和独立看门狗中断	0x0000 0040
17	1	可配置	LVD	LVD全局中断	0x0000 0044
18	2	-	-	-	0x0000 0048
19	3	可配置	FLASHRAM	FLASH/RAM全局中断	0x0000 004C
20	4	可配置	SYSCTRL	SYSCTRL全局中断	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	-	-	-	0x0000 0060
25	9	-	-	-	0x0000 0064
26	10	-	-	-	0x0000 0068
27	11	-	-	-	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
33	17	-	-	-	0x0000 0084
34	18	-	-	-	0x0000 0088
35	19	-	-	-	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	I2C1	I2C1全局中断	0x0000 009C
40	24	-	-	-	0x0000 00A0
41	25	可配置	SPI1	SPI1全局中断	0x0000 00A4
42	26	-	-	-	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	-	-	-	0x0000 00B4
46	30	可配置	AWT	AWT全局中断	0x0000 00B8
47	31	-	-	-	0x0000 00BC

5.5 中断向量重定位

利用向量表偏移寄存器 SCB->VTOR，Cortex-M0+处理器的向量表可以重定位。向量表重定位可用于多种情形：BootLoader、动态修改异常向量、加载应用到 RAM。

5.5.1 利用重定位实现 BootLoader

许多微控制器都在单独的 Bootloader ROM 里存放了 Bootloader 或启动固件，在执行 Flash 存储器里的应用代码前，处理器首先会执行启动 ROM 里的一小段代码。此时，处理器需要以启动 ROM 里的向量表启动，执行启动代码后设置 VTOR 以利用用户 Flash 存储器里的向量表并跳转到用户 Flash 存储器里的启动代码。

要从 Bootloader 切换到位于 0x1000 的用户应用的代码如下所示：

```
SCB->VTOR = 0x1000;
__set_MSP( *(__IO uint32_t *) 0x1000 );
(*((void (*)(void))( *(__IO uint32_t *)( 0x1000 + 4))))();
```

5.5.2 利用重定位实现动态修改异常向量

VTOR 的第二个常见用途为，在程序执行的不同阶段修改异常向量。在这个情况下，需要将整个向量表从程序映像复制到 SRAM，然后在需要时修改异常向量。此时应该多加留意，以免分配给向量表的存储器空间和应用剩余部分使用的 SRAM 空间重叠。

5.5.3 利用重定位加载应用到 RAM

应用可能存放在片外存储器中（如 SD 卡），需要加载到存储器系统中执行。在这种情况下，将程序映像复制到 RAM 后，如实现 BootLoader 小节所描述的方法跳转到 RAM 以执行加载完成的用户程序。

5.6 中断相关寄存器

5.6.1 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。

5.6.2 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。

5.6.3 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~IRQ32 的中断优先级，每个中断源使用 8 比特，在本芯片中仅使用了高 2 比特，最多可设置 4 个中断优先级。

5.6.4 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+内核有专用的 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 （使能中断）
void __disable_irq(void);       //设置 PRIMASK （禁止中断）
```

5.7 寄存器描述

寄存器列表

寄存器名称	偏移地址	描述
NVIC_ISER	0xE000 E000 + 0x100	IRQ0~IRQ31 中断使能设置寄存器
NVIC_ICER	0xE000 E000 + 0x180	IRQ0~IRQ31 中断使能清除寄存器
NVIC_ISPR	0xE000 E000 + 0x200	IRQ0~IRQ31 中断挂起设置寄存器
NVIC_ICPR	0xE000 E000 + 0x280	IRQ0~IRQ31 中断挂起清除寄存器
NVIC_IPR0	0xE000 E000 + 0x400	IRQ0~IRQ3 中断优先级控制寄存器0
NVIC_IPR1	0xE000 E000 + 0x404	IRQ4~IRQ7 中断优先级控制寄存器1
NVIC_IPR2	0xE000 E000 + 0x408	IRQ8~IRQ11 中断优先级控制寄存器2
NVIC_IPR3	0xE000 E000 + 0x40C	IRQ12~IRQ15 中断优先级控制寄存器3
NVIC_IPR4	0xE000 E000 + 0x410	IRQ16~IRQ19 中断优先级控制寄存器4
NVIC_IPR5	0xE000 E000 + 0x414	IRQ20~IRQ23 中断优先级控制寄存器5
NVIC_IPR6	0xE000 E000 + 0x418	IRQ24~IRQ27 中断优先级控制寄存器6
NVIC_IPR7	0xE000 E000 + 0x41C	IRQ28~IRQ31 中断优先级控制寄存器7

NVIC_ISER 中断使能设置寄存器

地址偏移：0x100 复位值：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

NVIC_ICER 中断使能清除寄存器

地址偏移：0x180 复位值：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

NVIC_ISPR 中断挂起设置寄存器

地址偏移：0x200 复位值：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

NVIC_ICPR 中断挂起清除寄存器

地址偏移：0x280 复位值：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

NVIC_IPR0 中断优先级控制寄存器 0

地址偏移：0x400 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ3	R/W	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，请保持默认值位，请保持默认值
23:22	IPRIRQ2	R/W	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	IPRIRQ2	R/W	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	R/W	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

NVIC_IPR1 中断优先级控制寄存器 1

地址偏移：0x404 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ7	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR2 中断优先级控制寄存器 2

地址偏移：0x408 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ11	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR3 中断优先级控制寄存器 3

地址偏移：0x40C 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ15	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR4 中断优先级控制寄存器 4

地址偏移：0x410 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ19	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR5 中断优先级控制寄存器 5

地址偏移：0x414 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ23	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR6 中断优先级控制寄存器 6

地址偏移：0x418 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ27	R/W	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	R/W	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	R/W	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	R/W	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

NVIC_IPR7 中断优先级控制寄存器 7

地址偏移：0x41C 复位值：0x0000 0000

位域	名称	权限	功能描述
31:30	IPRIRQ31	R/W	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	R/W	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	R/W	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	R/W	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 随机存储器（RAM）

6.1 概述

本芯片内置一块容量为 3KB 的随机存储器（RAM）。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。本存储器支持三种读写操作方式：字节（8 比特）、半字（16 比特）、字（32 比特）。

6.2 主要特性

- 多达 3KB 片上 RAM 存储器
- 访问速率为 HCLK
- 访问位宽支持字节（8 比特）、半字（16 比特）、字（32 比特）
- 具有奇偶校验功能

6.3 功能描述

本节将详细描述 RAM 存储器的相关功能。

6.3.1 存储器构成

本存储器的地址范围为 0x2000 0000 - 0x2000 0BFF，其大小为 3KB。

6.3.2 存储器操作

RAM 存储器支持两种操作：读操作、写操作；每种操作支持三种位宽：8bit、16bit 和 32bit。用户代码对 RAM 进行操作时需确保操作位宽与目标地址边界对齐，否则会导致 HardFault 硬件错误异常；对齐方式详见下表。

表 6-1 目标地址与位宽关系

目标地址最低两比特的值	00	01	10	11
支持的操作位宽	8bit / 16bit / 32bit	8bit	8bit / 16bit	8bit

6.3.2.1 读操作

对 RAM 的读操作可采用访问绝对地址方式，需注意操作位宽应与地址边界对齐。

8bit 位宽代码示例：
```
tmp8  = *((uint8_t  *)0x2000 0001);
```
16bit 位宽代码示例：
```
tmp16 = *((uint16_t *)0x2000 0002);
```
32bit 位宽代码示例：
```
tmp32 = *((uint32_t *)0x2000 0004);
```

6.3.2.2 写操作

对 RAM 的写操作可采用访问绝对地址方式，需注意操作位宽应与地址边界对齐。

8bit 位宽代码示例：
```
*((uint8_t  *)0x2000 0001)= 0x11;
```
16bit 位宽代码示例：
```
*((uint16_t *)0x2000 0002)= 0x1122;
```
32bit 位宽代码示例：
```
*((uint32_t *)0x2000 0004)= 0x11223344;
```

6.3.3 奇偶校验功能

本芯片支持 RAM 的奇偶校验功能，即每字节 RAM 存放在 9bit 的物理空间中，包括 8bit 数据位和 1bit 奇偶校验位。对 RAM 写入数据时，控制器将同步写入 8bit 数据及 1bit 校验值。从 RAM 读取数据时，控制器将读出 8bit 数据及 1bit 校验位；并验证其校验位是否正确。若校验位不正确则 RAM_ISR.PARITY 标志会被硬件置位，出错地址将存储到 RAM_ADDR 寄存器。当使能了奇偶校验出错中断（RAM_IER.PARITY=1），CPU 会执行校验出错中断服务程序。用户程序可设置 RAM_ICR.PARITY 为 0 以清除该中断标志。

6.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
RAM_IER	0x4002 2400 + 0x00	中断使能控制寄存器
RAM_ADDR	0x4002 2400 + 0x04	奇偶校验出错地址寄存器
RAM_ISR	0x4002 2400 + 0x08	中断标志寄存器
RAM_ICR	0x4002 2400 + 0x0C	中断标志清除寄存器

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

RAM_IER 中断使能控制寄存器

Address offset: 0x00 Reset value: 0x0000 0001

位域	名称	权限	功能描述
31:2	RFU	-	保留位
1	PARITY	RW	RAM奇偶校验出错中断使能控制 0：禁止 1：使能
0	RFU	-	保留位

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
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

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
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 闪存存储器（FLASH）

7.1 概述

本芯片内部集成了 20KB 嵌入式 FLASH 供用户使用，可用来存储应用程序及用户数据。FLASH 存储器控制器为嵌入式片上 FLASH 存储器提多种操作：编程、页面擦除和全片擦除功能以实现指令与数据的储存。

7.2 主要特性

- 多达 20KB 片上 FLASH 存储器
- 多达 22 字节 OTP 存储器
- 支持 3 种访问位宽：字节（Byte）、半字（HalfWord）、全字（Word）
- 支持 3 种操作模式：读出、写入、擦除
- 支持 3 种安全保护：读保护、写保护、擦保护

7.3 功能描述

本节将详细描述 FLASH 存储器的相关功能。

7.3.1 存储器构成

本芯片存储器由高达 20KB 的主 FLASH 存储区域、用于启动加载器的 2KB 信息区域和 22B 的 OTP 区域组成；其基地址、大小及保护设置详见下表。

OTP 区只可通过 ISP 协议写入，写入后不可擦除；适用于存储特定的不可修改数据。信息区在制造时写入，用户不可擦除或编程；用于存储启动程序及物理参数。主存储区域共包含 40 个可单独进行擦除的页面；用于存储用户代码及用户数据。

表 7-1 存储器地址分配

区块	名称	地址范围	擦写保护位	大小
主存储区	Page0	0x0000 0000 - 0x0000 01FF	PageLock.Lock0	512B
	……	……		……
	Page3	0x0000 0600 - 0x0000 07FF		512B
	Page4	0x0000 0800 - 0x0000 09FF	PageLock.Lock1	512B
	……	……		……
	Page7	0x0000 0E00 - 0x0000 0FFF		512B
	……	……	……	……
	Page36	0x0000 4800 - 0x0000 49FF	PageLock.Lock9	512B
	……	……		……
	Page39	0x0000 4E00 - 0x0000 4FFF		512B
OTP区	OTP存储区	0x0010 0770 - 0x0010 0785	-	22B
信息区	物理参数存储区	0x0010 0786 - 0x0010 07FF	-	122B
	启动程序存储区	0x0010 0000 - 0x0010 073F	-	1.8KB

7.3.2 访问延迟

本芯片内置的 FLASH 存储器支持的最快访问时钟频率为 24MHz，当 HCLK 频率大于 24MHz 时，需配置 CR2.WAIT 为合适的值，如下表所示。

表 7-2 访问周期配置

HCLK频率	WAIT位域值	FLASH的访问速率
HCLK ⩽ 24MHz	000	HCLK
HCLK ⩽ 48MHz	001	HCLK / 2

为满足 FLASH 访问速率的需要，在进行时钟切换时需满足以下两条规则：

- HCLK 从较低频率切换到较高频率：应先更改 WAIT 位域，然后再切换 HCLK 频率。
- HCLK 从较高频率切换到较低频率：应先切换 HCLK 频率，然后再更改 WAIT 位域。

7.3.3 存储器操作

FLASH 存储器支持三种操作模式：读出、擦除和编程；每种操作支持三种位宽：8bit、16bit 和 32bit。当页面处于锁定状态时，不能进行编程或擦除；在进行编程或擦除前应解锁待操作的页面；操作进行中（FLASH_CR1.BUSY=1）不可以使 MCU 进入 DeepSleep 模式；操作完成后（FLASH_CR1.BUSY=0）应及时将页面恢复成锁定状态并将 FLASH 操作模式更改为读取模式（FLASH_CR1.MODE=0）以防止程序异常时窜改数据。

FLASH 在执行擦除和编程操作时，需要使用校准的 HSI 时钟信号进行时序控制；在执行操作前，用户需向 SYSCTRL_HSI.TRIM 位域写入校准值。

用户代码对 FLASH 进行操作时需确保操作位宽与目标地址边界对齐，否则会导致 HardFault 硬件错误异常；对齐方式详见下表。

表 7-3 目标地址与位宽关系

目标地址最低两比特的值	00	01	10	11
支持的操作位宽	8bit / 16bit / 32bit	8bit	8bit / 16bit	8bit

7.3.3.1 全片擦除操作

FLASH 控制器提供了全片擦除功能用来复位 FLASH 内的所有数据，该操作完成后所有 FLASH 地址空间的数据内容均为 0xFF，OTP 地址空间的数据保持原值。只有当 CPU 从 RAM 执行程序时该操作才能执行成功。

如果对未解锁的 FLASH 进行全片擦除操作，该操作不会执行且 ISR.LOCK 标志位会被硬件置位。如果设置了 IER.LOCK 为 1，则 CPU 会执行相应的中断服务程序。用户可通过设置 ICR.LOCK 为 0 来清除 ISR.LOCK 中断标志。

对 FLASH 进行全片擦除的示例：

Step1. 设置 SYSCTRL_AHBEN.FLASH 为 1，使能 FLASH 模块时钟；

Step2. 向 FLASH_CR1 写入 0x5A5A0013，配置 FLASH 工作于片擦模式；

Step3. 向 FLASH_PAGELOCK 写入 0x5A5A003F，解锁所有页面；

Step4. 向 FLASH 内的任意地址写入任意数据，触发 FLASH 控制器进行全片擦除操作；

代码示例：`*(( uint8_t * ) 0x00 ) = 0x00;`

Step5. 查询等待 FLASH_CR1.BUSY 变成 0，全片擦除操作已完成；

Step6. 向 FLASH_PAGELOCK 写入 0x5A5A0000，锁定所有页面；

Step7. 向 FLASH_CR1 写入 0x5A5A0010，配置 FLASH 工作于读取模式。

注意：只有当 CPU 从 RAM 执行程序时全片擦除操作才能成功执行。

7.3.3.2 页擦除操作

FLASH 控制器提供了页擦除功能，用来复位指定的 FLASH 页面。每一页可以单独擦除而不影响其它页的内容；页擦除操作完成后，该页所有地址空间的数据内容均为 0xFF。OTP 存储区的数据不能被擦除。

如果对未解锁的 FLASH 页面进行页擦除操作，该操作不会执行且 ISR.LOCK 标志位会被硬件置位。如果设置了 IER.LOCK 为 1，则 CPU 会执行相应的中断服务程序。用户可通过设置 ICR.LOCK 为 0 来清除 ISR.LOCK 中断标志。

如果对 PC 指针所在的页面进行页擦除操作，该操作不会执行且 ISR.PC 标志位会被硬件置位。如果设置了 IER.PC 为 1，则 CPU 会执行相应的中断服务程序。用户可通过设置 ICR.PC 为 0 来清除 ISR.PC 中断标志。

对 FLASH 的 Page36 进行页擦除的示例：

Step1. 设置 SYSCTRL_AHBEN.FLASH 为 1，使能 FLASH 模块时钟；

Step2. 向 FLASH_CR1 写入 0x5A5A0012，配置 FLASH 工作于页擦模式；

Step3. 向 FLASH_PAGELOCK 写入 0x5A5A0020，解锁 Page36 – Page39；

Step4. 向 Page36 的首地址写入任意数据，触发 FLASH 控制器进行页擦除操作；

代码示例：`*(( uint8_t * )( 36 * 512 )) = 0x00;`

Step5. 查询等待 FLASH_CR1.BUSY 变成 0，页擦操作已完成；

Step6. 向 FLASH_PAGELOCK 写入 0x5A5A0000，锁定已完成页擦的页面；

Step7. 向 FLASH_CR1 写入 0x5A5A0010，配置 FLASH 工作于读取模式。

7.3.3.3 编程操作

FLASH 控制器提供了编程功能，用来修改 FLASH 的内容。编程操作只能将 FLASH 存储器中的数据由‘1’改写为‘0’，因此在进行编程之前需对待编程地址所在页进行页擦除操作。如需对 OTP 区域进行编程，请参阅 ISP 协议或使用驱动库中的函数。

本芯片在执行编程操作时，会检查目标存储空间内容是否为全‘1’，对应 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。如果对目标存储空间的数据不是全‘1’的地址进行编程操作，该操作不会执行且 ISR.PROG 标志位会被硬件置位。如果设置 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 ICR.PROG 为 0 来清除 ISR.PROG 中断标志。

如果对未解锁的存储空间进行编程操作，该操作操作不会执行且 ISR.LOCK 标志位会被硬件置位。如果设置 IER.LOCK 为 1，则 CPU 会执行对应的中断服务程序。用户可通过设置 ICR.LOCK 为 0 来清除 ISR.LOCK 中断标志。

如果对 PC 指针所在页面的存储空间进行编程操作，该操作不会执行且 ISR.PC 标志位会被硬件置位。如果设置 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 ICR.PC 为 0 来清除 ISR.PC 中断标志。

对 0x4800 地址进行编程的示例：

Step1. 设置 SYSCTRL_AHBEN.FLASH 为 1，使能 FLASH 模块时钟；

Step2. 向 FLASH_CR1 写入 0x5A5A0011，配置 FLASH 工作于编程模式；

Step3. 向 FLASH_PAGELOCK 写入 0x5A5A0020，解锁 Page36 – Page39

Step4. 向 0x4800 写入目标数据，触发 FLASH 控制器进行编程操作；

三种位宽的编程代码示例：

```
*(( uint8_t  * ) 0x4800 )) = 0x11;
*(( uint16_t * ) 0x4800 )) = 0x1122;
*(( uint32_t * ) 0x4800 )) = 0x11223344;
```

Step5. 查询等待 FLASH_CR1.BUSY 变成 0，编程操作已完成；

Step6. 向 FLASH_PAGELOCK 写入 0x5A5A0000，锁定已完成编程的页面；

Step7. 向 FLASH_CR1 写入 0x5A5A0010，配置 FLASH 工作于读取模式。

7.3.3.4 读操作

本芯片对 FLASH/OTP 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，三种位宽的读操作示例如下所示。

```
tmp8  = *((uint8_t  *) 0x01);
tmp16 = *((uint16_t *) 0x0002);
tmp32 = *((uint32_t *) 0x00000004);
```

7.3.4 安全保护

本芯片内置的 FLASH 存储器具有擦写保护及读保护功能，以防止 FLASH 存储器被非法擦写或读取数据。该功能对保护固件不受非法使用者侵害非常有用。

7.3.4.1 擦写保护

本芯片内置的 FLASH 存储器被划分为 40 页，每 4 个页面共用一个擦写保护位。当擦写保护位为 1 时，对应的页面可以进行擦写；当擦写保护位为 0 时，对应的页面不可以进行擦写。当用户程序在 FLASH 中运行并对 PC 指针所在的页面进行擦写，则该操作不会执行且 ISR.PC 标志位会被硬件置位。如果设置了 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 ICR.PC 为 0 以清除 ISR.PC 中断标志。

PAGELOCK 寄存器具有写保护功能；每次向该寄存器写数据时，待写入数据的高 16 比特必须为 0x5A5A 才能将数据写入。

锁定 Page0 – Page3 示例：

```
FLASH_PAGELOCK = 0x5A5A0000 | ( FLASH_PAGELOCK & 0xFFFE );
```

解锁 Page36 - Page39 示例：

```
FLASH_PAGELOCK = 0x5A5A0000 | ( FLASH_PAGELOCK | 0x0020 );
```

7.3.4.2 读保护

本芯片的FLASH存储器支持读保护功能。使能读保护后，无法通过ISP或SWD方式读出FLASH存储区的内容。无论是否使能 FLASH 读保护，CPU 均可正常执行指令和读取 FLASH 数据。
本存储器具有 4 个保护等级，当前保护等级可通过 CR1.SECURITY 来获取。读保护等级可通过 ISP 指令进行设置，具体设置方式请参阅 ISP 编程文档；读保护等级可通过驱动库函数进行设置，具体设置方式请参阅驱动库函数说明。
FLASH 的读保护功能如下表所示：

表 7-4 保护等级功能

保护等级	功能描述
Level0	未设置读保护。 可以通过SWD或ISP方式对FLASH进行读取操作。
Level1	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。
Level2	FLASH内容不可通过SWD或ISP方式读取。 可通过ISP方式将保护等级降低到Level0。 降级之后FLASH中内容为全0xFF。
Level3	FLASH内容不可通过SWD或ISP方式读取。 ISP和SWD功能都被禁止。 芯片不能再次进行编程。

7.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
FLASH_CR1	0x40022000 + 0x00	控制寄存器1
FLASH_CR2	0x40022000 + 0x04	控制寄存器2
FLASH_PAGELOCK	0x40022000 + 0x08	擦写锁定寄存器
FLASH_IER	0x40022000 + 0x20	中断使能寄存器
FLASH_ISR	0x40022000 + 0x24	中断标志寄存器
FLASH_ICR	0x40022000 + 0x28	中断标志清零寄存器

FLASH_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 0010

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才有效
15:8	RFU	-	保留位
7:6	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
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	STANDBY	RW	低功耗使能控制 0：当系统进入DeepSleep模式，FLASH不进入低功耗模式 1：当系统进入DeepSleep模式，FLASH进入低功耗模式 注意：请配置该控制位为1
3:2	RFU	-	保留位
1:0	MODE	RW	操作模式配置 00：读取模式，Read 01：编程模式，Program 10：页擦模式，PageErase 11：片擦模式，ChipErase

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
2:0	WAIT	RW	FLASH访问速率配置 000：HCLK， 适用于 HCLK≤24MHz 时 001：HCLK/2，适用于 HCLK≤48MHz 时

FLASH_PAGELOCK 擦写锁定寄存器 1

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:10	KEY	WO	仅当KEY为0x5A5A时，对该寄存器的写操作有效
9	LOCK9	RW	Page36 – Page39擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
8	LOCK8	RW	Page32 – Page35擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
…	…	…	…
1	LOCK1	RW	Page4 - Page7擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写
0	LOCK0	RW	Page0 – Page3擦写锁定配置 0：锁定，不可擦写 1：开放，可以擦写

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
4	PROG	RW	编程错误中断使能控制 0：禁止 1：使能
3:2	RFU	-	保留位，请保持默认值
1	LOCK	RW	擦写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

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
31:5	RFU	-	保留位
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
3:2	RFU	-	保留位，请保持默认值
1	LOCK	RO	擦写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

FLASH_ICR 中断标志清除寄存器

Address offset: 0x28 Reset value: 0x0000 000F

位域	名称	权限	功能描述
31:5	RFU	-	保留位
4	PROG	WO	编程错误标志清除 W0：清除编程错误标志 W1：无功能
3:2	RFU	-	保留位
1	LOCK	WO	擦写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	WO	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

8 循环冗余校验（CRC）

8.1 概述

循环冗余校验 (CRC) 计算单元将数据流作为输入，在生成多项式的控制下生成一个输出数。该输出数常应用于验证数据传输或数据存储的正确性和完整性。本芯片内置 CRC 计算单元，支持多种 CRC 算法。

8.2 主要特性

- 生成多项式：$x^{16} + x^{12} + x^5 + 1$
- 4 种常用的算法：
  - CRC16_CCITT
  - CRC16_CCITT_False
  - CRC16_X25
  - CRC16_XMODEM

8.3 功能描述

本节将详细描述循环冗余校验的相关功能。

8.3.1 算法模式

本芯片的 CRC 单元支持多种 CRC 算法。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，其对比如下表所示。

表 8-1 CRC 算法

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_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 $x^{16} + x^{12} + x^5 + 1$，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，该多项式的值为 0001 0000 0010 0001，即 0x1021。

- 初始值

在开始计算 CRC 校验值之前，先给 CRC 校验值赋的初值。

- 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。

8.3.2 计算流程及示例

CRC 计算单元支持的输入数据位宽为 8bit。

例如，用户需要计算 0x00，0x11，0x22，0x33，0x44，0x55，0x66，0x77 这一组数据的 CRC 校验值则依次将每一个字节写入 CRC_DR 寄存器即可。

采用 CRC_X25 算法对上述数据计算 CRC 的示例：

Step1. 设置 SYSCTRL_AHBEN.CRC 为 1，使能 CRC 模块时钟；

Step2. 设置 CRC_CR 为 6，配置 CRC 算法为 X25 并完成初始化；

Step3. 向 CRC_DR 写入 0x00；

Step4. 向 CRC_DR 写入 0x11；

Step5. 向 CRC_DR 写入 0x22；

Step6. 向 CRC_DR 写入 0x33；

Step7. 向 CRC_DR 写入 0x44；

Step8. 向 CRC_DR 写入 0x55；

Step9. 向 CRC_DR 写入 0x66；

Step10. 向 CRC_DR 写入 0x77；

Step11. 读取 CRC_RESULT 以获取 CRC 计算结果。

8.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
CRC_CR	0x4002 3000 + 0x00	控制寄存器
CRC_INIT	0x4002 3000 + 0x10	初始值寄存器
CRC_DR	0x4002 3000 + 0x08	数据寄存器
CRC_RESULT	0x4002 3000 + 0x0C	结果寄存器

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 0100：CRC16_CCITT 0101：CRC16_CCITT_False 0110：CRC16_X25 0111：CRC16_XMODEM

CRC_INIT 初始值寄存器

Address offset: 0x10 Reset value: 0x0000 FFFF

位域	名称	权限	功能描述
31:0	INIT	RW	CRC运算初始值，请保持默认值

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

CRC_RESULT 结果寄存器

Address offset: 0x0C 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	RESULT	RO	CRC计算结果

9 通用输入输出端口（GPIO）

9.1 概述

本芯片最多支持 21+1[注]个 GPIO 端口，实际可用的 GPIO 端口数量受封装限制，详见数据手册。GPIO 可配置为数字输入输出和模拟功能，支持外设功能复用，支持高电平、低电平、上升沿和下降沿 4 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。

注：经用户程序配置后，NRST 管脚可用作普通 GPIO 管脚。

9.2 主要特性

- 功能复用：每只管脚可配置为多种数字或模拟功能
- 输入模式：模拟输入、高阻输入、上/下拉输入
- 输出模式：模拟输出、推挽输出、开漏输出
- 信号输入：输入信号到达数据输入寄存器（GPIOx_IDR）及片内外设
- 信号输出：输出信号来源为数据输出寄存器（GPIOx_ODR）或片内外设
- 原子操作：具备原子操作寄存器（GPIOx_BSRR、GPIOx_BRR、GPIOx_TOG）
- 中断滤波：集成硬件滤波电路，屏蔽毛刺信号
- 中断类型：上升沿中断、下降沿中断、高电平中断、低电平中断
- 中断唤醒：每只管脚均可将 MCU 从深度休眠模式唤醒
- 辅助功能：输出各种内部时钟、片内外设输入信号重定向

9.3 功能描述

本节将详细描述通用输入输出端口的相关功能。

9.3.1 端口电路框图

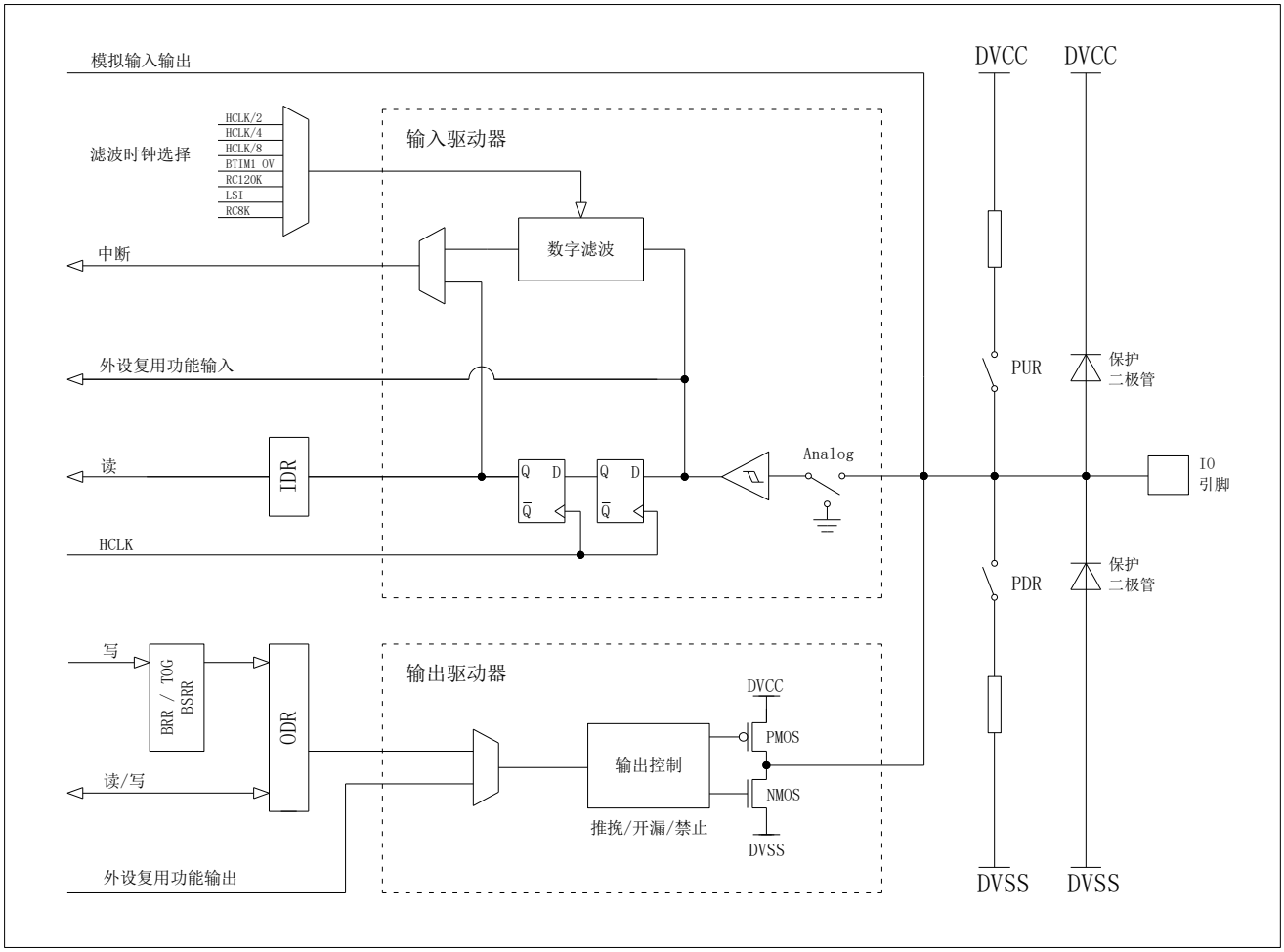

图 9-1 通用端口电路示意图

注意：由于每个 IO 引脚均具有保护二极管，故端口的输入电压必须小于 DVCC+0.3。

9.3.2 端口复位状态

在 MCU 复位期间及复位完成未配置 GPIO 寄存器之前，通用管脚均保持模拟输入无上下拉状态。特殊管脚的状态如下：

- SWCLK(PA05)：数字输入+上拉
- SWDIO(PA02)： 数字输入+上拉
- NRST(PC05) ： 数字输入+上拉

9.3.3 端口模式配置表

通过寄存器，可以将管脚配置为各种工作模式，其真值表如下方所示：

表 9-1 端口模式真值表

管脚工作模式	Analog	AFRL	DIR	OpenDrain	ODR	BRR	BSS	PUR	PDR
模拟输入输出	1	x	x	x	x	-	-	x	x
高阻输入	0	0	1	x	x	-	-	0	0
上拉输入	0	0	1	x	x	-	-	1	x
下拉输入	0	0	1	x	x	-	-	0	1
高阻输出	0	0	0	1	1	-	-	0	0
高阻输出	0	0	0	1	x	-	W1	0	0
上拉输出	0	0	0	1	1	-	-	1	x
上拉输出	0	0	0	1	x	-	W1	1	x
输出 0	0	0	0	1	0	-	-	x	x
输出 0	0	0	0	1	x	W1	-	x	x
输出 0	0	0	0	0	0	-	-	x	x
输出 0	0	0	0	0	x	W1	-	x	x
输出 1	0	0	0	0	1	-	-	x	x
输出 1	0	0	0	0	x	-	W1	x	x
外设输出	0	>0	0	0	x	-	-	x	x
外设上拉输入	0	>0	1	0	x	-	-	1	x
外设下拉输入	0	>0	1	0	x	-	-	0	1

9.3.4 模拟输入输出

当设置 GPIOx_ANALOGy 为 1 时端口被配置为模拟功能，模拟外设（ADC、VC、LVD……）可通过该端口进行工作。此时，端口数字功能被禁止：不能输出电平、读取管脚电平恒为 0、无内置上拉电阻及下拉电阻。具体每个管脚的模拟功能如下表所示。

表 9-2 模拟功能分配表

管脚名称	模拟功能	管脚名称	模拟功能
PA00	VC1_IN7、VC2_IN1、LVD_IN3	PB00	ADC_IN8、VC1_IN1
PA01	ADC_IN1、VC2_IN2	PB01	ADC_IN9、VC1_IN2
PA04	ADC_IN2、VC2_IN3	PB02	ADC_IN0、VC1_IN6、VC2_IN0
PA06	ADC_IN3、VC2_IN4	PB03	ADC_IN12、VC1_IN5、LVD_IN2
PA07	ADC_IN4、VC2_IN5	PB04	ADC_ExREF、VC1_IN4
PC00	ADC_IN5、VC2_IN6	PB05	ADC_IN11、VC1_IN3
PC01	ADC_IN6、VC2_IN7	PB06	ADC_IN10、LVD_IN1
PC02	ADC_IN7、VC1_IN0		

9.3.5 通用输入输出

当设置 GPIOx_ANALOGy 和 AFRLy 为 0 时端口被配置为通用数字输入输出功能。GPIOx_DIRy 为 1 时端口工作于输入模式；GPIOx_DIRy 为 0 时端口工作于输出模式。

当端口工作于输出模式时，可通过 GPIOx_OPENDRAINy 配置其输出方式为开漏或推挽；可通过 GPIOx_ODRy、GPIOx_BRRy、GPIOx_BSRRy、GPIOx_TOGy 配置其输出电平。其中 GPIOx_BRR、GPIOx_BSRR、GPIOx_TOG 寄存器是位操作寄存器，具备原子操作特性，可有效规避【读-修改-写】流程的不足之处。

当端口工作于输入模式时，可通过 GPIOx_PURy 使能或禁止端口内置的上拉电阻；可通过 GPIOx_PDRy 使能或禁止端口内置的下拉电阻；通过 GPIOx_IDRy 即可读取管脚的输入电平。GPIOx_IDR 寄存器与其前置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管脚电平跳变而造成的信号不稳定，但是会产生一定的读取延迟。

读端口管脚的同步时序如下图所示：

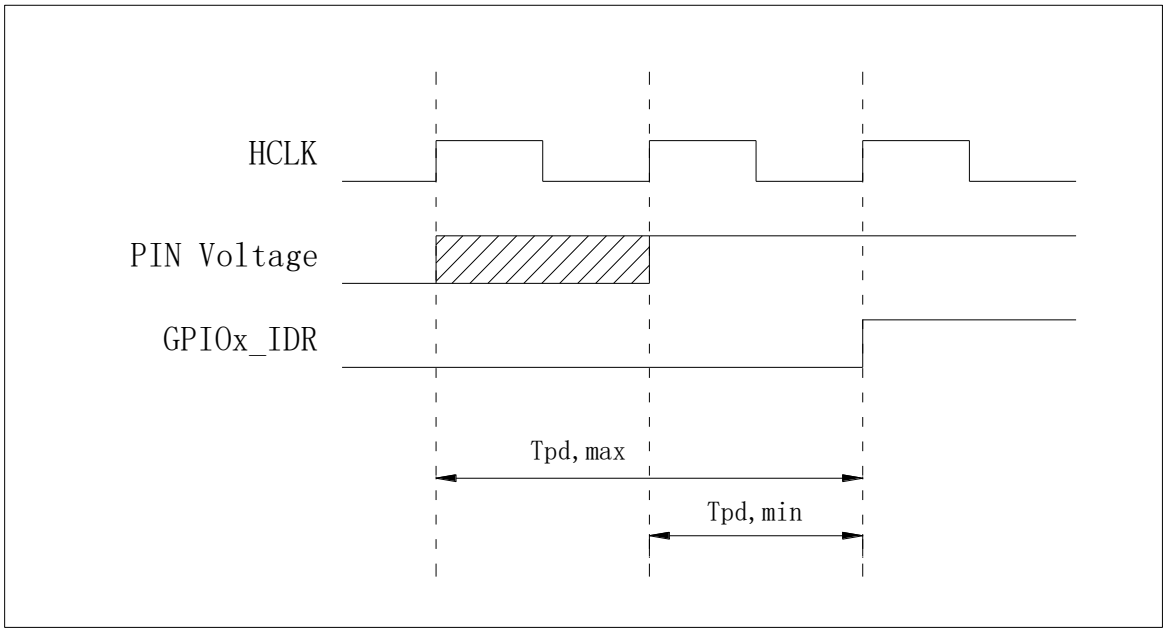

图 9-2 读取端口电平同步时序

在 HCLK 时钟上升沿之后的时钟周期，管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，稳定的管脚电平信号被读取，再一个 HCLK 时钟上升沿时，数据被锁存到 GPIOx_IDR 寄存器中。信号延迟 Tpd 为 1～2 个系统时钟。

9.3.6 数字复用功能

当设置 GPIOx_ANALOGy 为 0 且 AFRLy 大于 0 时端口被配置数字复用功能，数字外设（UART、SPI、TIM……）可通过该端口进行工作。GPIOx_DIRy 为 1 时端口工作于复用输入模式；GPIOx_DIRy 为 0 时端口工作于复用输出模式。

当端口工作于输入模式时，连接到端口的数字外设可自动读取管脚电平，用户可通过 GPIOx_IDR 读取管脚电平。配置 GPIOx_PURy、GPIOx_PDRy 可使能或禁止端口内置的上拉电阻及下拉电阻。

当端口工作于输出模式时，连接到端口的数字外设可自动控制其输出电平；GPIOx_ODR、GPIOx_BRR、GPIOx_BSRR、GPIOx_TOG 不再具有功能。通过 GPIOx_OPENDRAINy 配置其输出方式为开漏或推挽。

通过配置 GPIOx_AFRL.AFRy 为 1~7，即可将相应的端口配置为数字复用功能 AF1~AF7。具体每个管脚的复用功能如下表所示。

表 9-3 数字复用功能分配表

管脚名称	复用功能							
	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA00	GPIO	UART1_RXD	UART2_RTS	SPI1_SCK	ATIM_CH3A	GTIM1_CH4	BTIM1_TOGN	VC1_OUT
PA01	GPIO	UART2_TXD	VC2_OUT	SPI1_MOSI	ATIM_CH3B	GTIM1_CH1	BTIM2_TOGP	MCO_OUT
PA02	GPIO	UART1_RXD	UART2_TXD	I2C1_SDA	GTIM1_ETR	GTIM1_CH3	VC2_OUT	AWT_ETR
PA03	GPIO	UART2_TXD	UART1_RXD	PCLK_OUT	ATIM_BK	GTIM1_ETR	BTIM2_TOGP	LVD_OUT
PA04	GPIO	UART1_RXD	IR_OUT	SPI1_MISO	ATIM_CH3B	GTIM1_CH2	BTIM2_TOGN	GTIM1_ETR
PA05	GPIO	UART1_TXD	UART2_RXD	I2C1_SCL	ATIM_GATE	GTIM1_CH4	BTIM_ETR	MCO_OUT
PA06	GPIO	UART1_CTS	UART2_TXD	I2C1_SDA	ATIM_CH2B	GTIM1_CH3	BTIM3_TOGP	LVD_OUT
PA07	GPIO	UART1_RTS	UART2_RXD	VC1_OUT	ATIM_CH1B	GTIM1_CH4	BTIM3_TOGN	ATIM_BK
PB00	GPIO	UART1_RXD	I2C1_SDA	SPI1_CS	ATIM_CH1B	GTIM1_CH1	GTIM1_TOGP	AWT_ETR
PB01	GPIO	UART1_TXD	LVD_OUT	I2C1_SCL	ATIM_BK	GTIM1_CH2	GTIM1_TOGN	AWT_ETR
PB02	GPIO	UART1_TXD	UART2_CTS	SPI1_CS	ATIM_CH2B	GTIM1_CH3	BTIM1_TOGP	MCO_OUT
PB03	GPIO	UART2_RXD	I2C1_SDA	PCLK_OUT	ATIM_CH2A	GTIM1_CH2	BTIM3_TOGP	IR_OUT
PB04	GPIO	UART2_TXD	I2C1_SCL	GTIM1_ETR	ATIM_ETR	GTIM1_CH1	BTIM3_TOGN	ATIM_BK
PB05	GPIO	UART1_RXD	I2C1_SDA	BTIM_ETR	ATIM_CH1B	GTIM1_TOGN	BTIM2_TOGN	ATIM_BK
PB06	GPIO	UART1_TXD	I2C1_SCL	SPI1_CS	ATIM_CH1A	GTIM1_TOGP	BTIM2_TOGP	HCLK_OUT
PB07	GPIO	UART2_RXD	UART1_TXD	SPI1_SCK	ATIM_GATE	GTIM1_CH1	BTIM2_TOGN	BTIM_ETR
PC00	GPIO	UART2_RXD	UART1_TXD	SPI1_SCK	ATIM_CH1A	GTIM1_CH2	BTIM1_TOGP	HCLK_OUT
PC01	GPIO	UART2_TXD	GTIM1_ETR	SPI1_MISO	ATIM_CH2A	GTIM1_CH3	BTIM1_TOGN	VC1_OUT
PC02	GPIO	UART2_RXD	IR_OUT	SPI1_MOSI	ATIM_CH3A	GTIM1_CH4	HCLK_OUT	AWT_ETR
PC03	GPIO	UART1_TXD	SPI1_CS	SPI1_MISO	ATIM_CH3B	GTIM1_CH3	GTIM1_TOGP	ATIM_BK
PC04	GPIO	UART1_RXD	IR_OUT	SPI1_MOSI	ATIM_CH2B	GTIM1_CH4	GTIM1_TOGN	ATIM_GATE
PC05	GPIO							

9.3.7 SWD 管脚

芯片复位后，SYSCTRL_CR2.SWDIO 的值为 0，PA02、PA05 的功能为 SWD；这两只管脚的状态为数字输入+上拉电阻。当用户设置 SYSCTRL_CR2.SWDIO 为 1 后，PA02、PA05 的功能为 GPIO；这两只管脚的状态由 GPIO 相关寄存器控制。在应用中若需要用到这两只管脚的 GPIO 功能，需注意这两只管脚被配置为 GPIO 之前的状态为数字输入+上拉电阻，设计电路时应规避这段弱驱高电平持续时间对外围电路的影响。

9.3.8 RESET/PC05 管脚

芯片复位后，SYSCTRL_CR2.RSTIO 的值为 0，PC05 的功能为 RESET；这只管脚的状态为数字输入+上拉电阻。当用户设置 SYSCTRL_CR2.RSTIO 为 1 后，PC05 的功能为 GPIO；这只管脚的状态由 GPIO 相关寄存器控制。在应用中若需要用到这只管脚的 GPIO 功能，需注意这只管脚被配置为 GPIO 之前的状态为数字输入+上拉电阻，设计电路时应规避这段弱驱高电平持续时间对外围电路的影响。

芯片复位后，对 SYSCTRL_CR2 寄存器的首次写操作可配置 PC05 的功能为 GPIO 或 RESET；后续对 SYSCTRL_CR2 寄存器的写操作将不可改写 SYSCTRL_CR2.RSTIO 的值。

注意：RESET/PC05 管脚内置了约 8kΩ 上拉电阻，该电阻不可通过程序关闭。

9.3.9 组合管脚

当芯片封装的引脚数量不足以连接芯片内部的所有 GPIO 时，部分 GPIO 将以组合方式连接到同一个管脚上；该管脚的输出驱动能力为其它管脚的 2 倍。

四对组合管脚如下所示：

- GPIOA[4]/GPIOA[3]的输出驱动电路均连接到 PA04 管脚；
  GPIOA[4]/GPIOA[3]的输出控制电路均由 GPIOA[4]的数字逻辑控制；
  GPIOA[4]/GPIOA[3]的输入功能保持正常。
- GPIOB[6]/GPIOB[7]的输出驱动电路均连接到 PB06 管脚；
  GPIOB[6]/GPIOB[7]的输出控制电路均由 GPIOB[6]的数字逻辑控制；
  GPIOB[6]/GPIOB[7]的输入功能保持正常。
- GPIOB[0]/GPIOC[3]的输出驱动电路均连接到 PB00 管脚；
  GPIOB[0]/GPIOC[3]的输出控制电路均由 GPIOB[0]的数字逻辑控制；
  GPIOB[0]/GPIOC[3]的输入功能保持正常。
- GPIOB[1]/GPIOC[4]的输出驱动电路均连接到 PB01 管脚；
  GPIOB[1]/GPIOC[4]的输出控制电路均由 GPIOB[1]的数字逻辑控制；
  GPIOB[1]/GPIOC[4]的输入功能保持正常。

9.3.10 中断功能

每个被配置被为数字输入的管脚，均具有中断功能。每个管脚均支持四种类型的中断：高电平、低电平、上升沿、下降沿；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，GPIOx_ISR 的相应比特会被硬件置位，用户程序向 GPIOx_ICR 的相应比特写入“0”即可清除该中断标志。

本芯片的中断支持硬件数字滤波；设置 GPIOx_FILTER 的相应比特为 1，即可使能该功能。当输入信号的持续时间小于一个完整的滤波时钟周期，则会被硬件数字滤波电路滤除；当输入信号的持续时间大于两个完整的滤波时钟周期，则一定可以通过硬件数字滤波电路传递到后级中断产生电路；硬件数字滤波会使产生中断的时刻延后 1~2 个滤波时钟周期。本芯片配备了 7 种滤波时钟可供用户选择：HCLK/2、HCLK/4、HCLK/8、BTIM1 溢出、LSI、RC120K、RC8K。当 MCU 处于 Active 或 Sleep 模式时，从 GPIO 输入的信号在所有配置下均可正常生产中断；当 MCU 处于 DeepSleep 模式且关闭硬件滤波功能时，从 GPIO 输入的信号可正常产生中断；当 MCU 处于 DeepSleep 模式且使能硬件滤波并配置滤波时钟为 LSI、RC120K 或 RC8K 时，从 GPIO 输入的信号经硬件滤波后可正常产生中断。

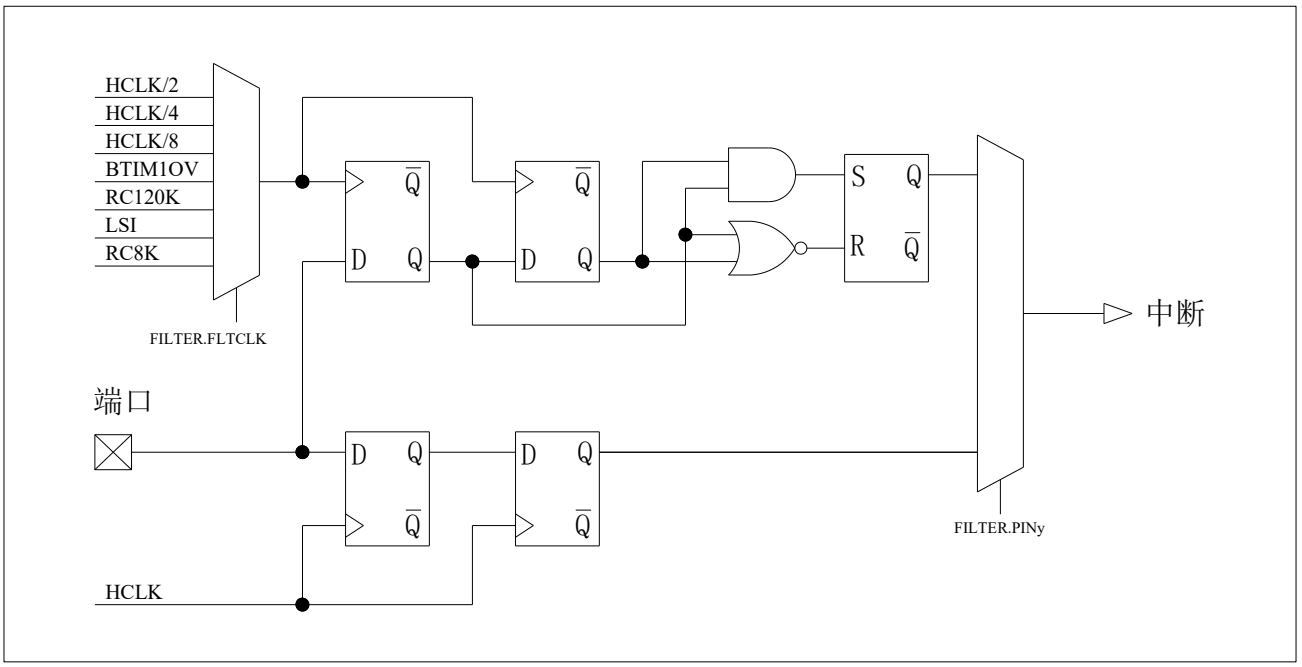

图 9-3 中断及数字滤波示意

9.4 寄存器描述

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_PDR	GPIOx_Base + 0x0C	GPIOx 下拉电阻寄存器
GPIOx_PUR	GPIOx_Base + 0x10	GPIOx 上拉电阻寄存器
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_HIGHIE	GPIOx_Base + 0x2C	GPIOx 高电平中断使能寄存器
GPIOx_LOWIE	GPIOx_Base + 0x30	GPIOx 低电平中断使能寄存器
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 滤波功能寄存器
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

GPIOA_Base = 0x4800 0000
GPIOB_Base = 0x4800 0400
GPIOC_Base = 0x4800 0800

GPIOx_ANALOG 端口模拟数字配置寄存器

Address offset: 0x1C

Reset value: 0x0000 001B(GPIOA) 0x0000 00FF(GPIOB) 0x0000 001B(GPIOC)

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_AFRL 端口复用功能配置寄存器低段

Address offset: 0x18 Reset value: 0x0000 0000

位域	名称	权限	功能描述
30:28	AFR7	RW	参见 AFR0
……	……	……	……
6:4	AFR1	RW	参见 AFR0
2:0	AFR0	RW	端口数字复用功能控制 000：GPIO 001：AF1 010：AF2 011：AF3 100：AF4 101：AF5 110：AF6 111：AF7

注：未列出的位为保留位，请保持默认值。

GPIOx_DIR 端口输入输出方向寄存器

Address offset: 0x00

Reset value: 0x0000 003F(GPIOA) 0x0000 00FF(GPIOB) 0x0000 003F(GPIOC)

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	RW	端口输入输出方向控制 0：将端口配置为输出 1：将端口配置为输入

GPIOx_OPENDRAIN 端口输出模式寄存器

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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	RW	端口输出方式控制位 0：推挽输出 1：开漏输出

GPIOx_PUR 端口上拉电阻寄存器

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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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

注：当 PDR.PINy、PUR.PINy 同时置 1 时，端口状态为使能上拉电阻，y = 1 ~ 7。

GPIOx_PDR 端口下拉电阻寄存器

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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	RW	端口下拉电阻使能控制 0：禁止下拉电阻 1：使能下拉电阻

注：当 PDR.PINy、PUR.PINy 同时置 1 时，端口状态为使能上拉电阻，y = 1 ~ 7。

GPIOx_FILTER 端口中断数字滤波器配置寄存器

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 100：RC120K 101：LSI 110：RC8K
15:8	RFU	-	保留位，请保持默认值
7	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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 RC120K、LSI、RC8K。

GPIOx_IDR 端口输入数据寄存器

Address offset: 0x50

Reset value: 0x0000 0024(GPIOA) 0x0000 0000(GPIOB) 0x0000 0024(GPIOC)

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
7	PIN7	RO	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_ODR 端口输出数据寄存器

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

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_BRR 端口位清零寄存器

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	PIN7	R0W1	参见 PIN0
……	……	……	……
0	PIN0	R0W1	端口位清零控制 W0：无功能 W1：设置 GPIOx_ODR 寄存器相应的比特为 0

GPIOx_BSRR 端口位置位清零寄存器

Address offset: 0x5C 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	BRR7	R0W1	参见 BRR0
……	……	……	……
16	BRR0	R0W1	端口位清零控制 W0：无功能 W1：设置 GPIOx_ODR 寄存器相应的比特为 0
15:8	RFU	-	保留位，请保持默认值
7	BSS7	R0W1	参见 BSS0
……	……	……	……
0	BSS0	R0W1	端口位置位控制 W0：无功能 W1：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，y = 1 ~ 7。

GPIOx_TOG 端口位翻转寄存器

Address offset: 0x60 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	PIN7	R0W1	参见 PIN0
……	……	……	……
0	PIN0	R0W1	端口位翻转控制 0：无功能 1：对 GPIOx_ODR 寄存器相应的比特取反

GPIOx_RISEIE 上升沿中断使能寄存器

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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_FALLIE 下降沿中断使能寄存器

Address offset: 0x28 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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_HIGHIE 高电平中断使能寄存器

Address offset: 0x2C 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	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	RW	端口高电平中断使能控制 0：禁止相应端口的高电平中断 1：使能相应端口的高电平中断

GPIOx_LOWIE 低电平中断使能寄存器

Address offset: 0x30 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	PIN7	RW	参见 PIN0
……	……	……	……
0	PIN0	RW	端口低电平中断使能控制 0：禁止相应端口的低电平中断 1：使能相应端口的低电平中断

GPIOx_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
31:8	RFU	-	保留位，请保持默认值
7	PIN7	RO	参见 PIN0
……	……	……	……
0	PIN0	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

GPIOx_ICR 中断标志清除寄存器

Address offset: 0x38

Reset value: 0x0000 00FF(GPIOA) 0x0000 00FF(GPIOB) 0x0000 00FF(GPIOC)

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	PIN7	R1W0	参见 PIN0
……	……	……	……
0	PIN0	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

10 基本定时器（BTIM）

10.1 概述

本芯片内部集成 3 个基本定时器 BTIM，每个 BTIM 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 支持定时器模式、计数器模式、触发启动模式和门控模式这 4 种工作模式，支持溢出事件触发中断请求。

10.2 主要特性

- 16bit 计数器
- 多种预分频器
- 单脉冲计数
- 门控功能
- 触发启动功能
- 外部计数功能
- 多个定时器级联工作

10.3 功能描述

10.3.1 功能框图

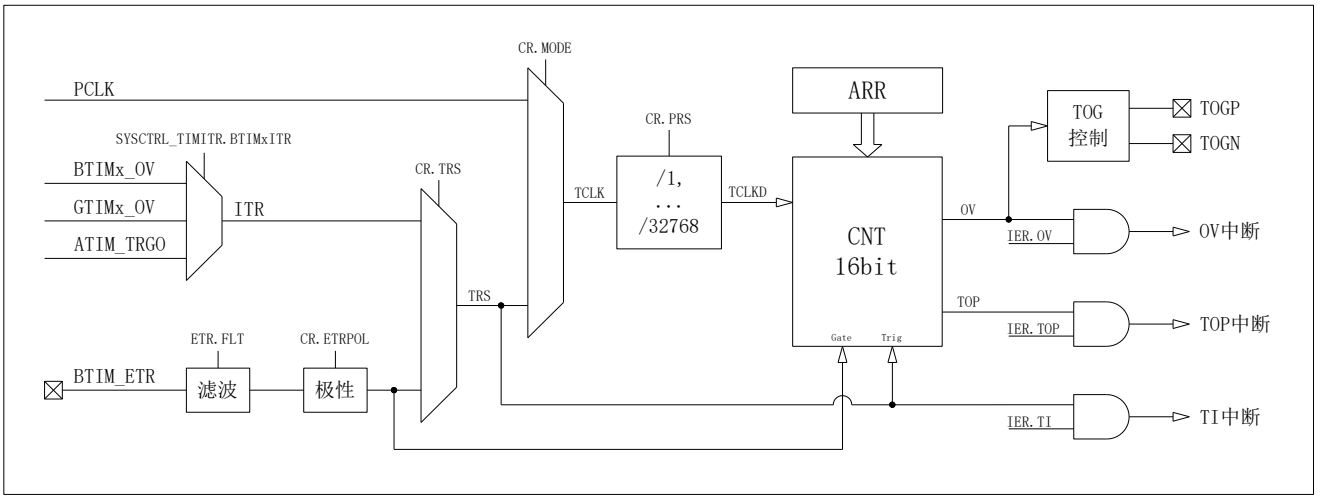

图 10-1 BTIM 功能框图

10.3.1.1 滤波和极性选择单元

来自外部管脚的 ETR 信号经滤波、极性选择处理后，供计数单元及触发单元使用。

滤波单元以一定的采样频率对待滤波信号进行采样，当连续采样到 N 个相同电平时才会输出该信号以滤除高频杂波信号。滤波单元的典型电路图如下方所示。

注意，待滤波信号的频率应低于采样频率的二分之一。

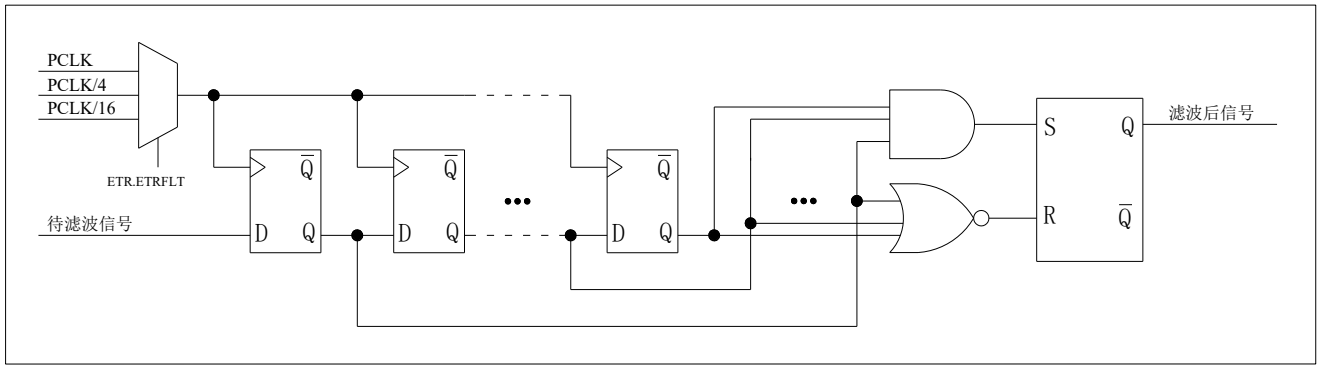

图 10-2 数字滤波电路示意

通过 CR.ETRPOL 位域，可配置 ETR 信号的有效沿是上升沿还是下降沿；ETR 信号的有效电平是高电平还是低电平。

10.3.1.2 预分频器

预分频器在 CR.PRS 位域的控制下对计数时钟 TCLK 进行分频，其分频系数为 1～32768。CR.PRS 被修改后，新的预分频系数会在定时器发生溢出时或 CR.EN 由 0 变为 1 时生效。

10.3.1.3 计数单元

在每个 TCLKD 信号的上升沿，内部计数器 CNT 自动加 1。每当计数器计数到 ARR 时，计数器将产生溢出信号 OV，然后再次自动从 0 开始加计数。计数器的计数范围为 0~ARR，计数单元的计数周期为 ARR+1。

每次溢出时，ISR.OV 会被硬件置 1，若 IER.OV 为 1 则会产生溢出中断，用户在退出中断服务程序前应清除该溢出标志。若 CR.TOGEN 为 1，则每溢出一次 TOGP/TOGN 输出电平翻转一次；用户可通过 GPIO 辅助功能将 TOGP/TOGN 信号输出到管脚用于驱动外部电路。

用户代码在定时器运行时可以修改 ARR 寄存器，且 ARR 的值将立即生效。若修改后的 ARR 值小于当前计数值，则计数器 CNT 将一直向上计数到 0xFFFF 后，产生 TOP 溢出信号，并从 0 重新开始计数。

定时器支持两种计数模式：单次计数模式和连续计数模式；可通过 CR.ONESHOT 位域配置。若定时器工作于连续计数模式，计数器将持续计数直到用户设置 CR.EN 为 0。若定时器工作于单次计数模式时，当计数值到达 ARR 后，硬件将自动设置 CR.EN 为 0 以停止计数。

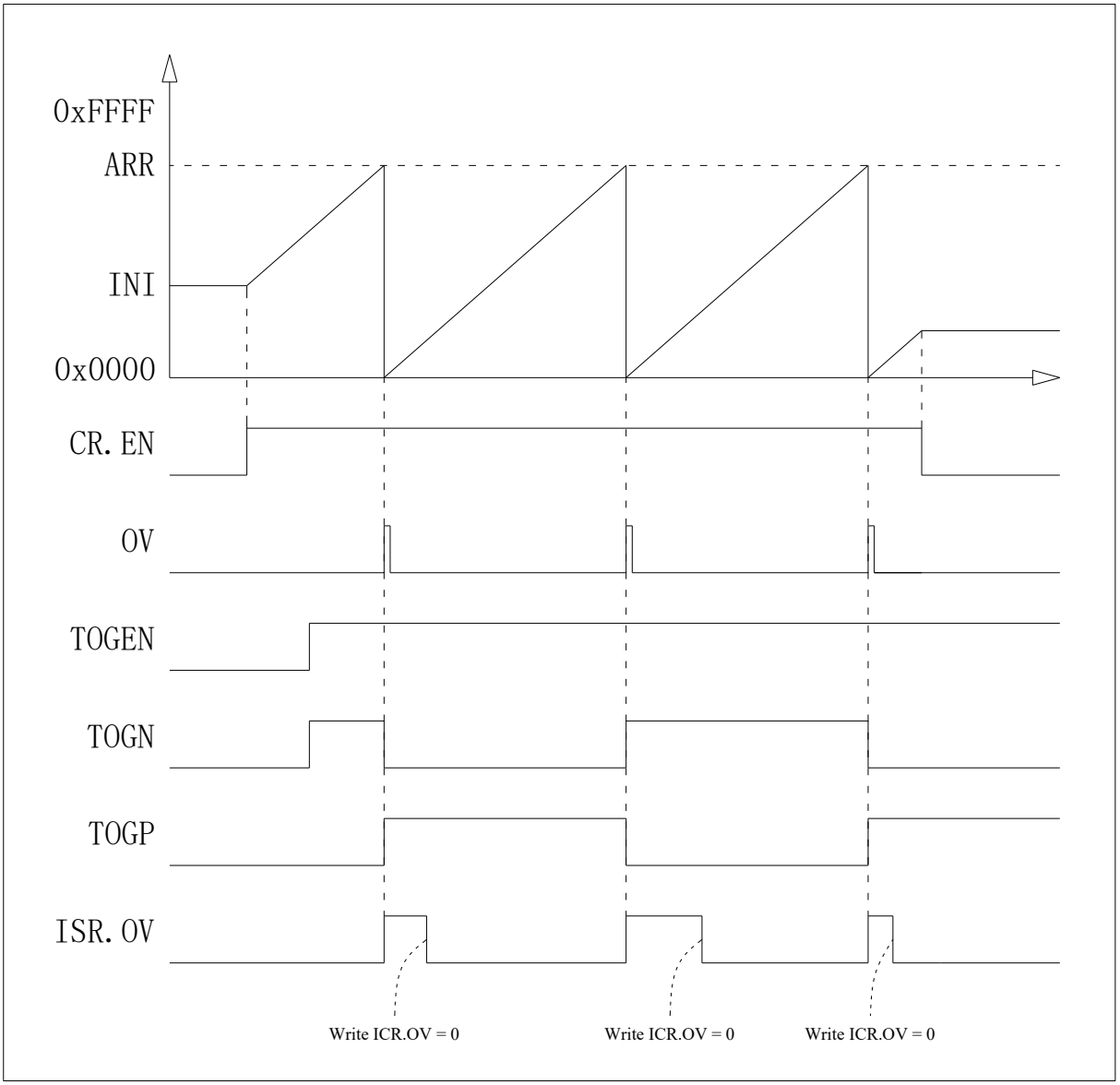

图 10-3 计数器波形

10.3.2工作模式

BTIM 支持 4 种工作模式：定时器模式、计数器模式、触发启动模式、门控模式。用户可根据实际需要灵活选择。

10.3.2.1 定时器模式

当设置 CR.MODE 为 0x00 时，BTIM 工作于定时器模式；该模式主要用于产生固定时间间隔的时基信号。计数单元的计数时钟为分频后的 PCLK，其功能框图如下所示。

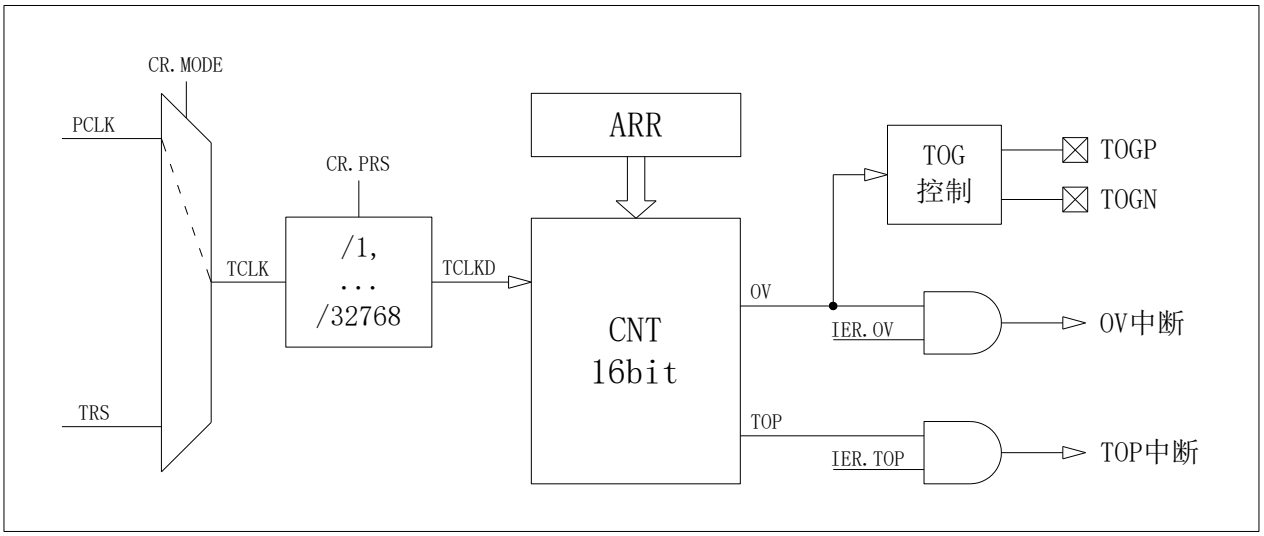

图 10-4 定时器模式框图

当设置 CR.EN 为 1 后，计数单元在 TCLKD（来自 PCLK）的驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位；若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

该工作模式的定时时间间隔计算公式如下：

$$T = \frac{(ARR + 1) \times 2^{PRS}}{f_{PCLK}}$$

其中，f_{PCLK} 为 PCLK 的频率，PRS 为预分频系数，ARR 为重载值。

10.3.2.2 计数器模式

当设置 CR.MODE 为 0x01 时，BTIM 工作于计数器模式；该模式主要用于测定某个事件发生的次数，可对 ITR 信号或 ETR 信号进行计数。

当 CR.TRS 为 1 时，计数源为内部 ITR 信号；当 CR.TRS 为 0 时，计数源为 ETR 信号。当选择 ITR 信号时，还可通过 SYSCTRL_TIMITR.BTIMxITR 选 ITR 信号的来源以实现多个 TIM 级联工作；当选择 ETR 信号时，还可通过 ETR.FLT 配置 ETR 信号的滤波参数，通过 CR.ETRPOL 配置 ETR 信号的极性。

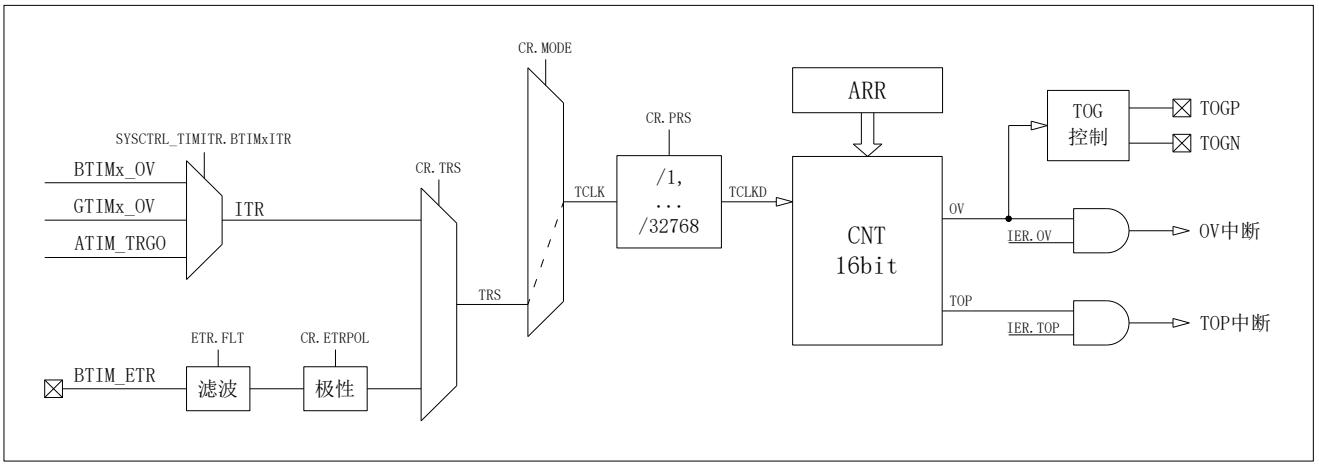

图 10-5 计数器模式框图

当设置 CR.EN 为 1 后，计数单元在 TCLKD（来自 TRS）的驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位；若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

10.3.2.3 触发启动模式

当设置 CR.MODE 为 0x02 时，BTIM 工作于触发启动模式，该模式主要用于在特定事件发生时启动定时器。有效的触发信号边沿或设置 CR.EN 为 1 均可启动定时器；启动后计数单元在 TCLKD（来自 PCLK）驱动下进行累加计数。当用户设置 BTIMx_CR.EN 为 0 时可暂停定时器，再次出现有效的触发信号边沿或设置 CR.EN 为 1 均可再次启动定时器。

当 CR.TRS 为 1 时触发源为内部 ITR 信号，用户还可通过 SYSCTRL_TIMITR.BTIMxITR 选 ITR 信号的来源以实现多个 TIM 同步启动。

当 CR.TRS 为 0 时触发源为 ETR 信号，用户还可通过 ETR.FLT 配置 ETR 信号的滤波参数，通过 CR.ETRPOL 配置 ETR 信号的极性。

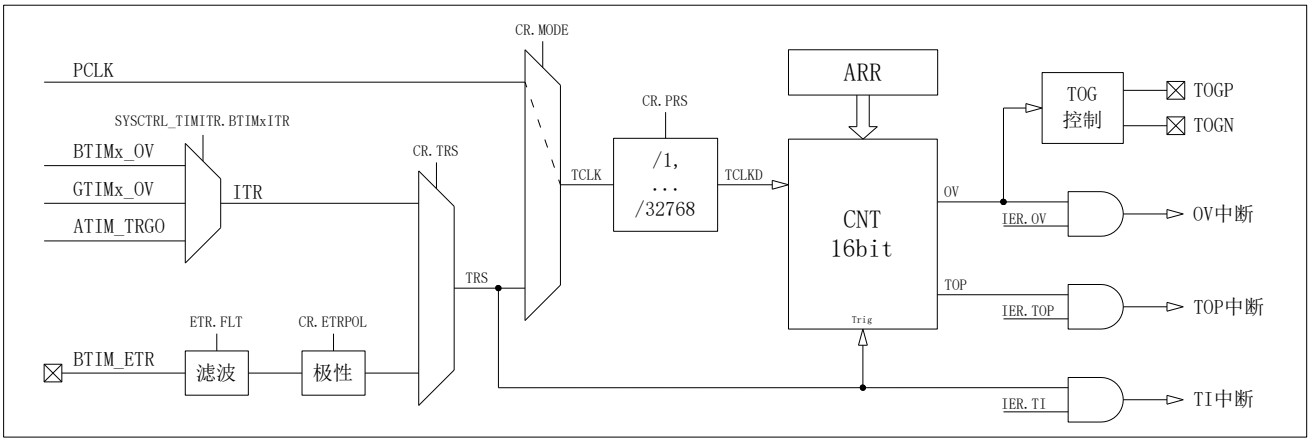

图 10-6 触发启动模式框图

定时器启动后，计数单元在 TCLKD（来自 PCLK）驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位；若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

每当出现有效的触发信号时 ISR.TI 标志会被硬件置位，若使能触发中断（IER.TI=1）则 CPU 将执行触发中断服务程序。

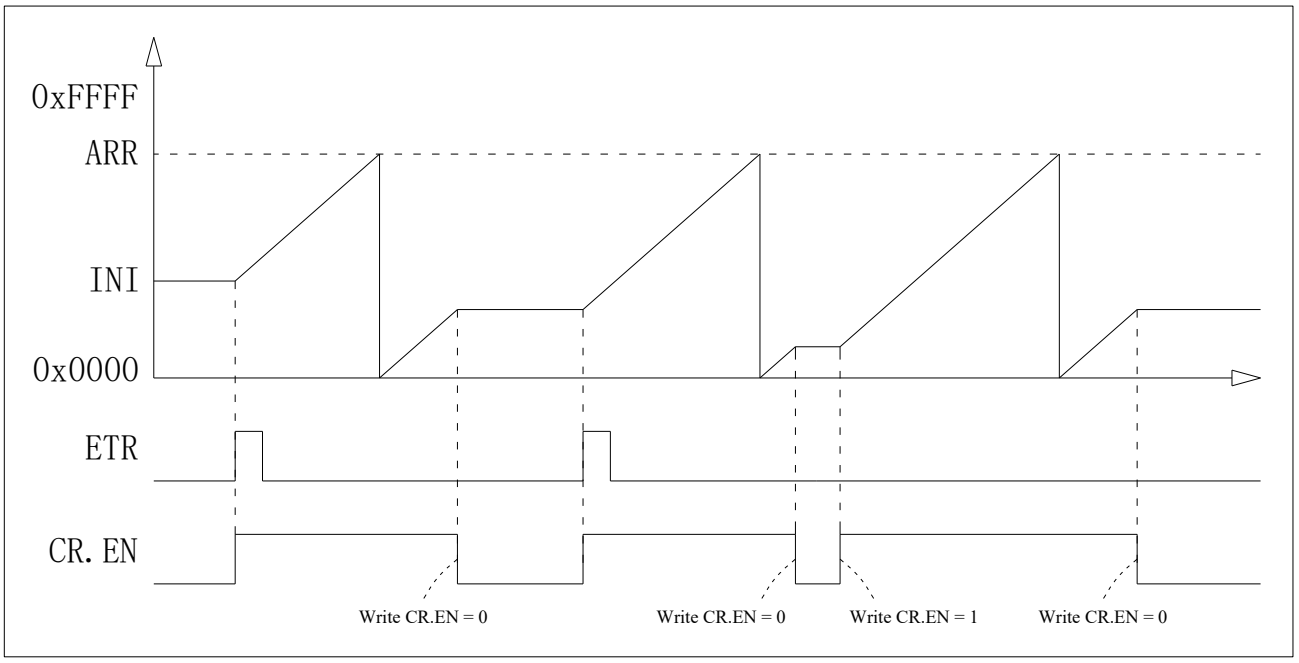

图 10-7 触发启动计数示意图

10.3.2.4 门控模式

当设置 CR.MODE 为 0x03 时，BTIM 工作于门控模式。当 CR.EN 为 1 且门控信号 Gate（来自 ETR）有效时，计数单元在 TCLKD（来自 PCLK）驱动下从初值开始加计数。可通过 ETR.FLT 配置 ETR 信号的滤波参数，通过 CR.ETRPOL 配置 ETR 有效电平的极性。

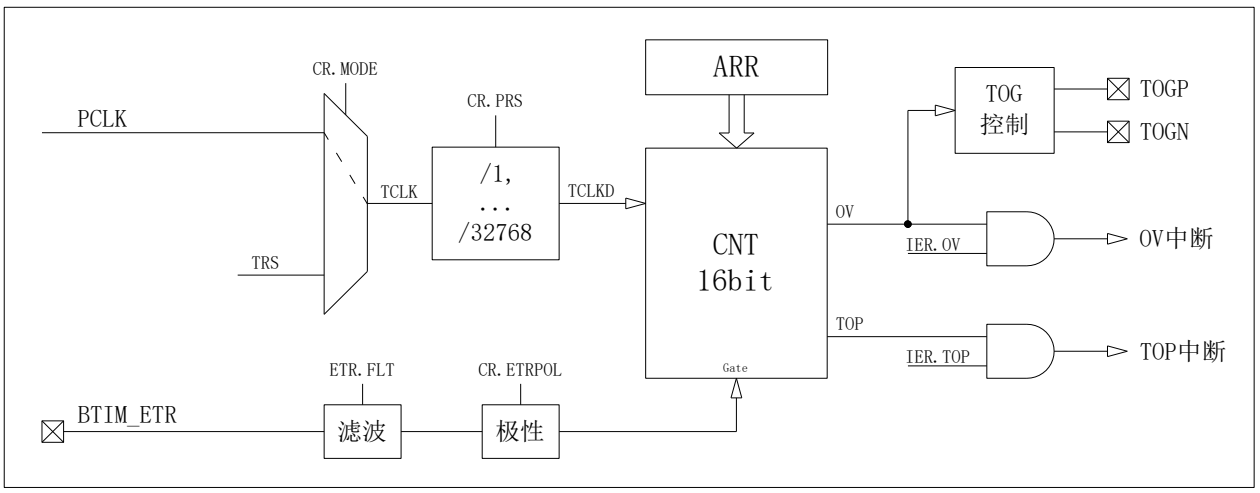

图 10-8 门控模式框图

当 CR.EN 为 1 且门控信号有效时，计数单元在 TCLKD（来自 PCLK）的驱动下从初值开始加计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位；若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

在任意时候设置 CR.EN 为 0 或门控信号无效时，计数单元立即暂停计数；当运行控制位 CR.EN 为 1 且门控信号有效时，计数单元继续计数。

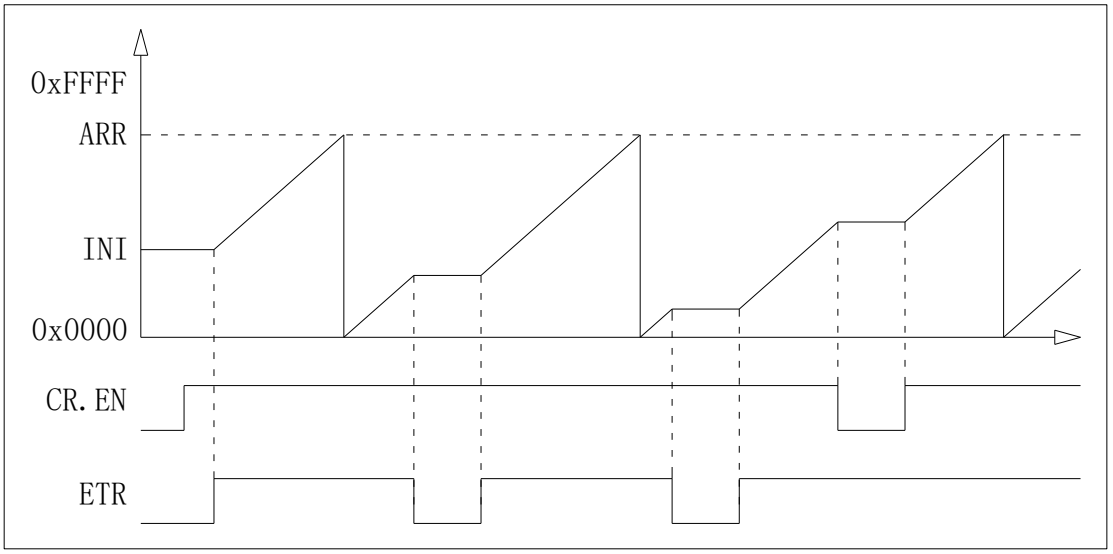

图 10-9 门控计数示意图

10.3.3 内部级联 ITR

当 BTIM 工作于计数模式且 TRS 来源为 ITR 时，即可将两个 TIM 级联使用。级联时，前一级 TIM 的溢出信号或主模式输出作为本级 TIM 的计数时钟。

级联可以将两个 TIM 组合成一个 32 位的定时器，也可以将前一级 TIM 作为后一级 TIM 的任意预分频器（1~65536）。

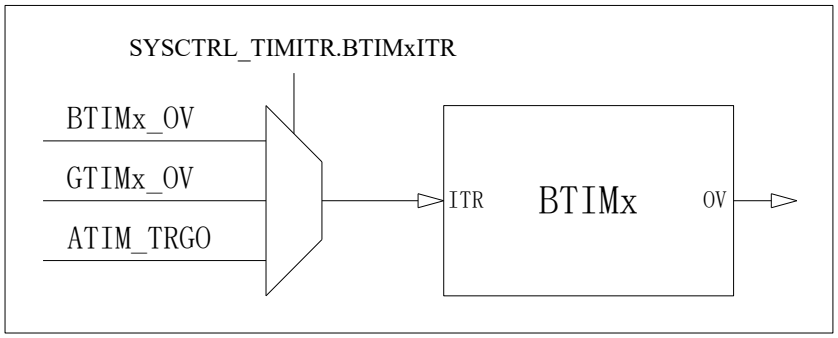

图 10-10 内部级联示意图

10.3.4 调试支持

BTIM 支持在调试模式下停止或继续计数，由 SYSCTRL_DEBUGEN.BTIM 控制。

调试状态下，SYSCTRL_DEBUGEN.BTIM 为 1 时，BTIMx 暂停工作；

调试状态下，SYSCTRL_DEBUGEN.BTIM 为 0 时，BTIMx 继续工作。

10.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
BTIMx_ARR	BTIMx_Base + 0x00	重载寄存器
BTIMx_CNT	BTIMx_Base + 0x04	计数寄存器
BTIMx_ETR	BTIMx_Base + 0x0C	外部触发控制寄存器
BTIMx_CR	BTIMx_Base + 0x10	控制寄存器
BTIMx_IER	BTIMx_Base + 0x14	中断使能寄存器
BTIMx_ISR	BTIMx_Base + 0x18	中断标志寄存器
BTIMx_ICR	BTIMx_Base + 0x1C	中断标志清除寄存器

BTIM1_Base = 0x4001 4800
BTIM2_Base = 0x4001 4900
BTIM3_Base = 0x4001 4A00

BTIMx_CR 控制寄存器

Address offset: 0x10 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
14:11	PRSSTATUS	RO	预分频器当前正在使用的分频系数
10:7	PRS	RW	预分频器分频系数配置 0：DIV1 4：DIV16 8：DIV256 12：DIV4096 1：DIV2 5：DIV32 9：DIV512 13：DIV8192 2：DIV4 6：DIV64 10：DIV1024 14：DIV16384 3：DIV8 7：DIV128 11：DIV2048 15：DIV32768
6	TOGEN	RW	TOG管脚输出波形使能控制 0：TOGP、TOGN输出电平均为0 1：TOGP、TOGN输出电平相反的信号
5	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
4	ETRPOL	RW	外部触发信号（ETR）极性选择 0：正向(触发模式上升沿有效，门控模式高电平有效) 1：反向(触发模式下降沿有效，门控模式低电平有效)
3	TRS	RW	触发源选择 0：ETR信号，来自ETR管脚的信号 1：ITR信号，其来源详见内部级联
2:1	MODE	RW	基本定时模式配置 00：定时器模式，对PCLK进行计数 01：计数器模式，对TRS进行计数 10：触发模式，由TRS信号触发计数器启动对PCLK计数 11：门控模式，在ETR信号控制下对PCLK进行计数
0	EN	RW	定时器运行控制 0：定时器停止 1：定时器运行

BTIMx_ARR 重载寄存器

Address offset: 0x00 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

BTIMx_CNT 计数寄存器

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	CNT	RW	定时器计数值

BTIMx_ETR 外部触发控制寄存器

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
6:4	FLT	RW	ETR信号数字滤波电路采样时钟及采样点数配置 000：无滤波 001：$f_{sample} = f_{PCLK}$，N=2 010：$f_{sample} = f_{PCLK}$，N=4 011：$f_{sample} = f_{PCLK}$，N=6 100：$f_{sample} = f_{PCLK}/4$，N=4 101：$f_{sample} = f_{PCLK}/4$，N=6 110：$f_{sample} = f_{PCLK}/8$，N=4 111：$f_{sample} = f_{PCLK}/8$，N=6
3:0	RFU	-	保留位，请保持默认值

BTIMx_IER 中断使能寄存器

Address offset: 0x14 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	TOP	RW	计数器TOP溢出中断使能控制 0：禁止TOP溢出中断 1：使能TOP溢出中断
1	TI	RW	触发中断使能控制 0：禁止触发中断 1：使能触发中断
0	OV	RW	计数器ARR溢出中断使能控制 0：禁止ARR溢出中断 1：使能ARR溢出中断

BTIMx_ISR 中断标志寄存器

Address offset: 0x18 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	TOP	RO	计数器TOP溢出标志 0：计数器未TOP溢出 1：计数器已TOP溢出
1	TI	RO	触发标志 0：未发生触发事件 1：已发生触发事件
0	OV	RO	计数器ARR溢出标志 0：计数器未ARR溢出 1：计数器已ARR溢出

BTIMx_ICR 中断标志清除寄存器

Address offset: 0x1C Reset value: 0x0000 0007

位域	名称	权限	功能描述
31:3	RFU	-	保留位，请保持默认值
2	TOP	R1W0	计数器TOP溢出标志清除 W0：清除计数器TOP溢出标志 W1：无功能
1	TI	R1W0	触发标志清除 W0：清除触发标志 W1：无功能
0	OV	R1W0	计数器ARR溢出标志清除 W0：清除计数器ARR溢出标志 W1：无功能

11 通用定时器（GTIM）

11.1 概述

本芯片内部集成 1 个通用定时器 GTIM，该 GTIM 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 支持定时器模式、计数器模式、触发启动模式、门控模式和编码计数模式，支持中断请求。

11.2 主要特性

- 16bit 计数器
- 多种预分频器
- 单脉冲计数
- 门控功能
- 触发启动功能
- 外部计数功能
- 4 路独立输入捕获 / 输出比较通道
- PWM 占空比调节范围为 0%～100%
- 编码计数器功能
- 多个定时器级联工作
- 片内外设互联

11.3 功能描述

11.3.1 功能框图

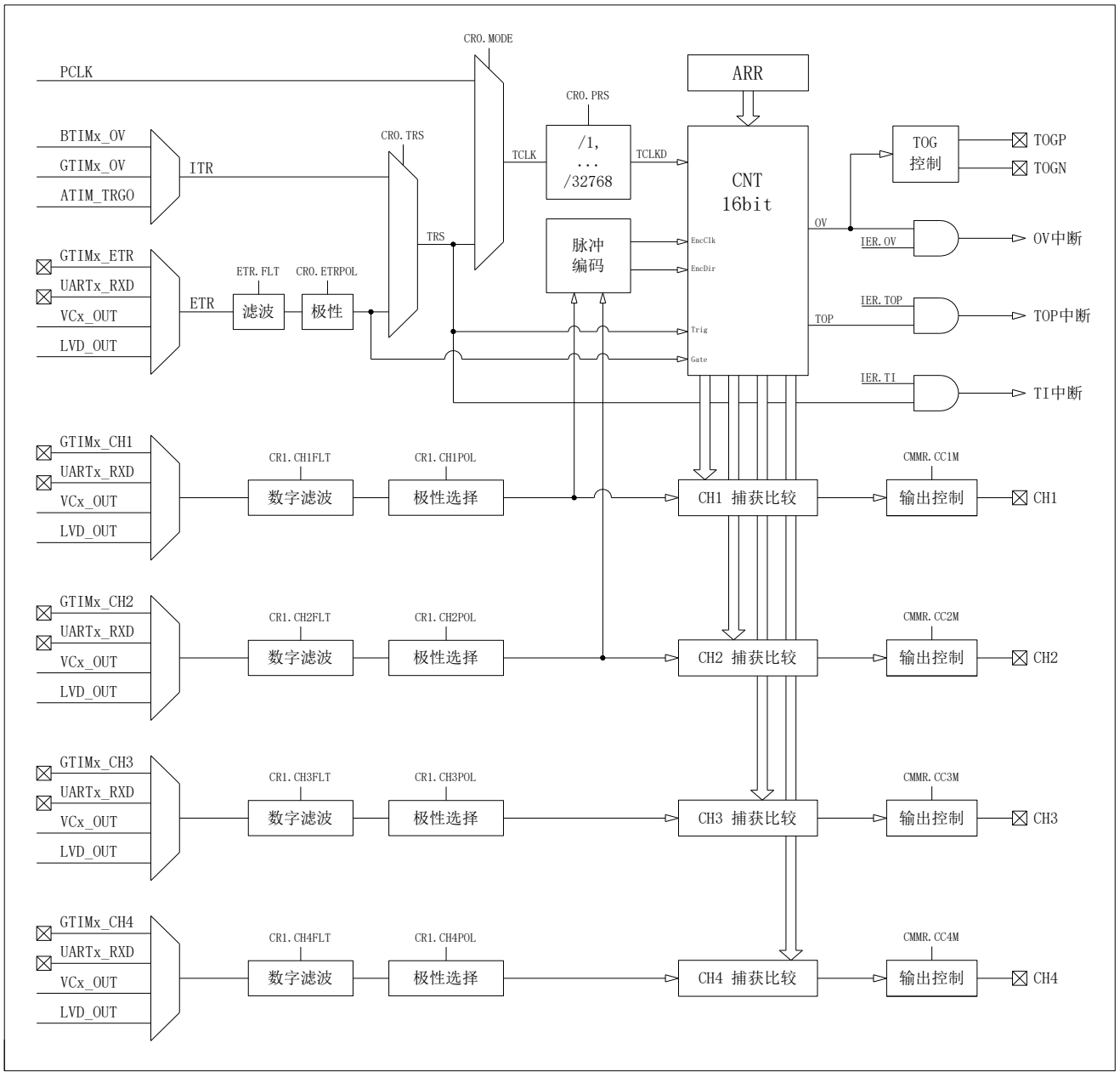

图 11-1 GTIM 功能框图

11.3.1.1 预分频器

预分频器在 CR0.PRS 位域的控制下对计数时钟 TCLK 进行分频，其分频系数为 1 ~ 32768。CR.PRS 被修改后，新的预分频系数会在定时器发生溢出时或 CR0.EN 由 0 变为 1 时生效。

11.3.1.2 计数单元

定时计数模式下，在每个 TCLKD 信号的上升沿，内部计数器 CNT 自动加 1。每当计数器计数到 ARR 时，计数器将产生溢出信号 OV，然后再次自动从 0 开始加计数。计数器的计数范围为 0~ARR，计数单元的计数周期为 ARR+1。每次溢出时，ISR.OV 会被硬件置 1，若使能溢出中断(IER.OV=1)，CPU 将会执行溢出中断服务程序。若 CR0.TOGEN 为 1，则每溢出一次 TOGP/TOGN 输出电平翻转一次；用户可通过 GPIO 辅助功能将 TOGP/TOGN 信号输出到管脚用于驱动外部电路。用户代码在定时器运行时可以修改 ARR 寄存器，且 ARR 的值将立即生效。若修改后的 ARR 值小于当前计数值，则计数器 CNT 将一直向上计数到 0xFFFF 后，产生 TOP 溢出信号，并从 0 重新开始计数。若定时器工作于单次计数模式时（CR0.ONSHOT=1），当计数值到达 ARR 后，硬件将自动设置 CR0.EN 为 0 以停止计数。

在编码计数模式下，在每个 EncClk 的上升沿，内部计数器 CNT 根据 EncDir 自动加 1 或减 1。每当加计数到 ARR 时，计数器将产生上溢出信号 OV 并再次自动从 0 开始加计数；每当减计数到 0 时，计数器将产生下溢出信号 UD 并从 0xFFFF 开始减计数。

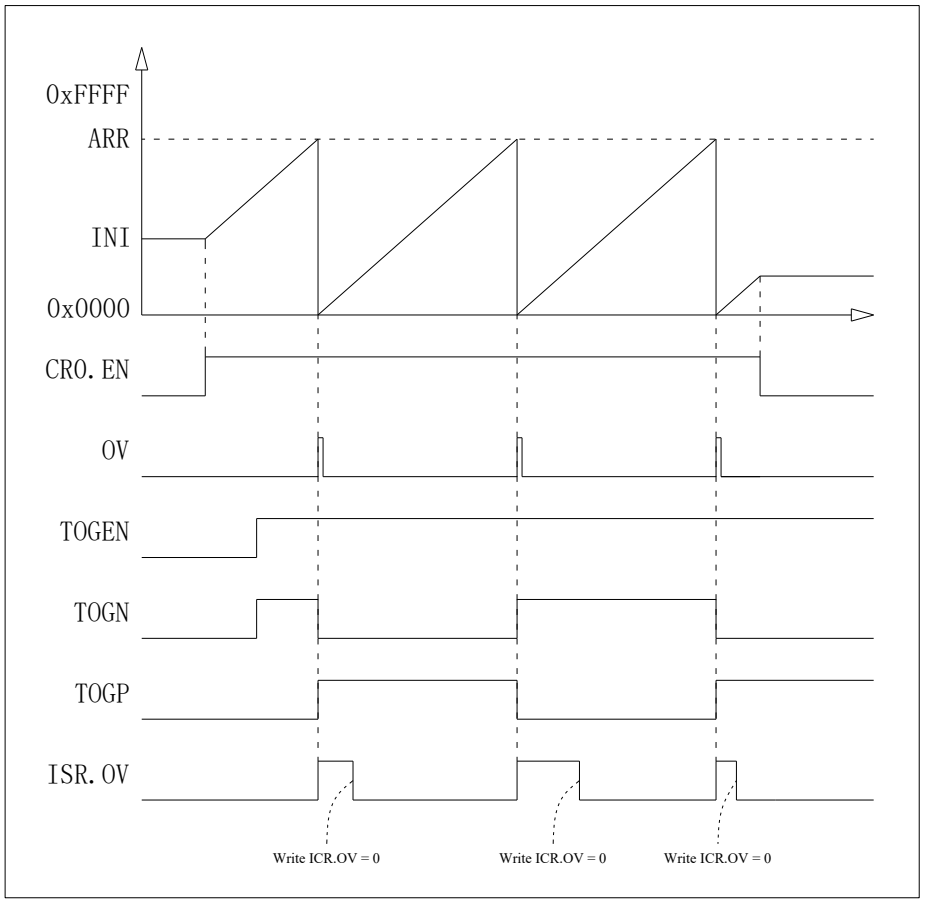

图 11-2 定时计数模式计数器波形

11.3.1.3 滤波和极性选择单元

滤波单元以一定的采样频率对待滤波信号进行采样，当连续采样到 N 个相同电平时才会输出该信号以滤除高频杂波信号。滤波单元的典型电路图如下方所示。
注意，待滤波信号的频率应低于采样频率的二分之一。

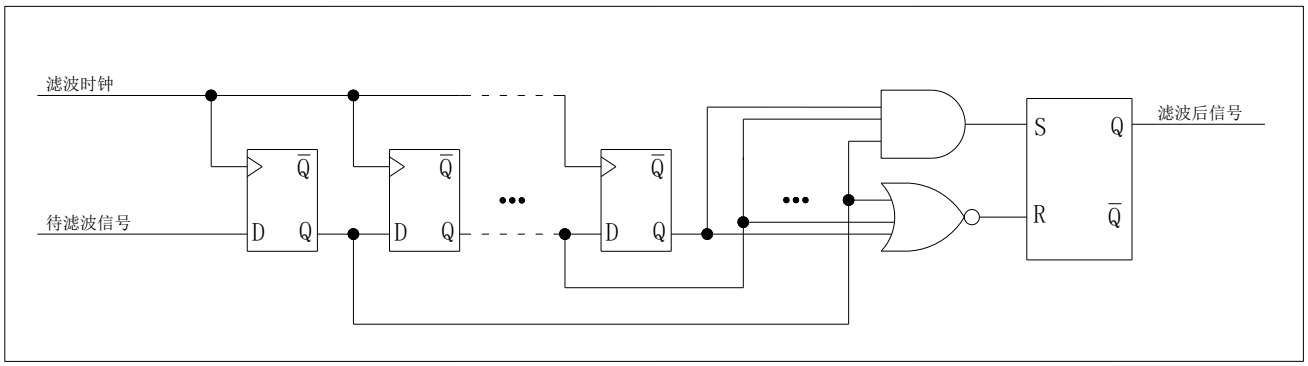

图 11-3 数字滤波电路示意

来自管脚或片内外设的 ETR 信号经滤波、极性选择处理后，供计数单元及触发单元使用。通过 ETR.FLT 位域，可配置滤波单元的采样时钟频率及采样点数。通过 CR0.ETRPOL 位域，可配置 ETR 信号的有效沿是上升沿还是下降沿；ETR 信号的有效电平是高电平还是低电平。
来自外部管脚或片内外设的 CHy 信号经滤波、极性选择处理后，供输入捕捉单元及编码计数单元使用。通过 CR1.CHyFLT 位域，可配置滤波单元的采样时钟频率及采样点数。

11.3.1.4 输入捕获输出比较单元

捕获比较通道由一个比较捕获寄存器和一个比较器组成，如下图所示。

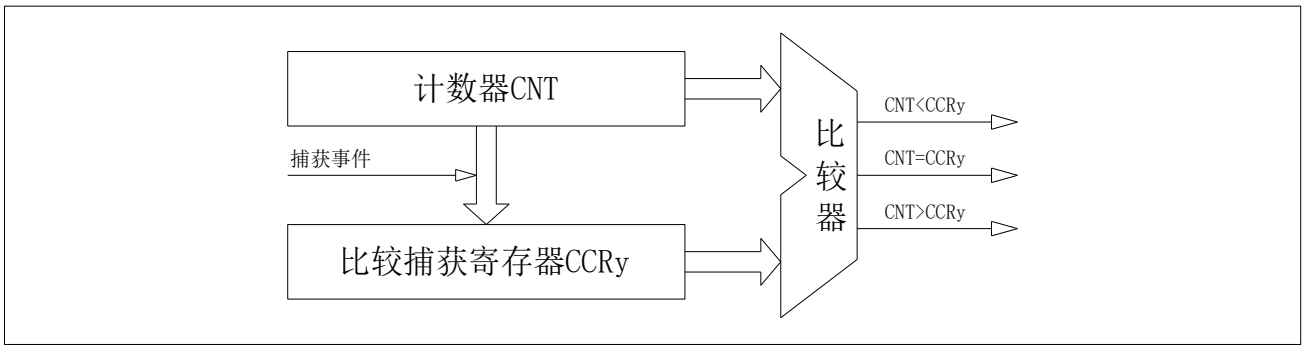

图 11-4 比较捕获电路示意

在输入捕获模式，一旦相应的通道发生捕获事件，计数器 CNT 的当前值立即被传输到比较捕获寄存器 CCRy 保存并置位 ISR.CCy 标志；如果相应通道的捕获比较中断使能，则 CPU 将执行中断服务程序。
在输出比较模式，比较器持续比较计数器 CNT 的值和比较捕获寄存器 CCRy 的值，其比较结果被送到输出控制单元。当计数器 CNT 的值与比较捕获寄存器 CCRy 的值相等时，ISR.CCy 标志被硬件置 1；如果相应通道的捕获比较中断使能，则 CPU 将执行相应的中断服务程序。

11.3.1.5 输出控制单元

输出控制单元用于比较匹配时，控制输出端口的波形。输出控制单元支持输出 6 种波形：低电平、高电平、匹配时清 0、匹配时置 1、PWM 正向信号、PWM 反向信号。

11.3.2基本工作模式

GTIM 支持 4 种基本工作模式：定时器模式、计数器模式、触发启动模式和门控模式。用户可根据实际需要灵活选择。

11.3.2.1 定时器模式

当设置 CR0.MODE 为 0x00 时，GTIM 工作于定时器模式。该模式主要用于产生固定时间间隔的时基信号。计数单元的计数时钟为分频后的 PCLK，其功能框图如下所示。

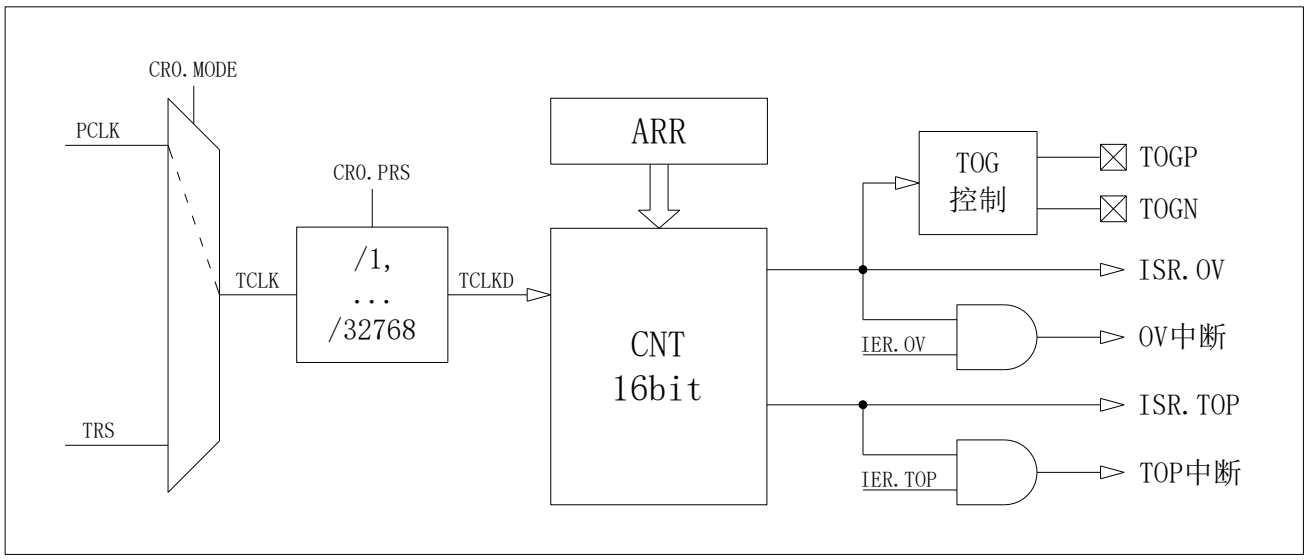

图 11-5 定时器模式框图

当设置 CR0.EN 为 1 后，计数单元在 TCLKD（来自 PCLK）的驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位，若 IER.OV 为 1，则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

该工作模式的定时时间间隔计算公式如下：

$$T = \frac{(ARR + 1) \times 2^{PRS}}{f_{PCLK}}$$

其中，f_{PCLK} 为 PCLK 的频率，PRS 为预分频系数，ARR 为重载值。

11.3.2.2 计数器模式

当设置 CR0.MODE 为 0x01 时，GTIM 工作于计数器模式，该模式主要用于测定某个事件发生的次数，可对 ITR 信号或 ETR 信号进行计数。

CR0.TRS 为 1 时计数源为 ITR 信号，用户还可通过 SYSCTRL_TIMITR.GTIMxITR 选 ITR 信号的来源以实现多个 TIM 级联工作。

当 CR0.TRS 为 0 时计数源为 ETR 信号，用户还可通过 SYSCTRL_GTIMETR.GTIMxETR 选 ETR 信号的来源为 GTIMx_ETR 管脚、UARTx_RXD、VCx_OUT 及 LVD_OUT；可通过 CR1.ETRFLT 配置 ETR 信号的滤波参数；通过 CR0.ETRPOL 配置 ETR 信号的极性。

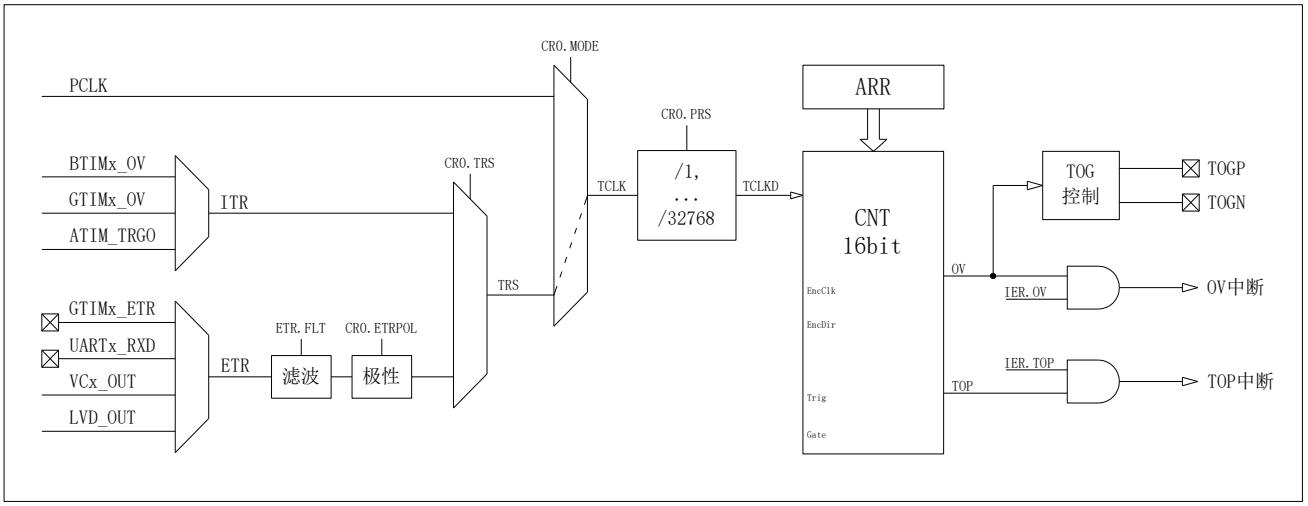

图 11-6 计数器模式功能框图

当设置 CR0.EN 为 1 后，计数单元在 TCLKD（来自 TRS）的驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位，若 IER.OV 为 1，则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

该工作模式的定时时间间隔计算公式如下：

$$T = \frac{(ARR + 1) \times 2^{PRS}}{f_{TRS}}$$

其中，f_{TRS} 为触发信号的频率，PRS 为预分频系数，ARR 为重载值。

11.3.2.3 触发启动模式

当设置 CR0.MODE 为 0x02，GTIM 工作于触发启动模式，该模式主要用于在特定事件发生时启动定时器。有效的触发信号边沿或设置 CR0.EN 为 1 均可启动定时器；启动后计数单元在 TCLKD（来自 PCLK）驱动下进行累加计数。当用户设置 CR0.EN 为 0 时可暂停定时器，再次出现有效的触发信号边沿或设置 CR0.EN 为 1 均可再次启动定时器。

当 CR0.TRS 为 1 时触发源为 ITR 信号，用户还可通过 SYSCTRL_TIMITR.GTIMxITR 选 ITR 信号的来源以实现多个 TIM 协同工作。

当 CR0.TRS 为 0 时触发源为 ETR 信号，用户可通过 SYSCTRL_GTIMETR.GTIMxETR 选择 ETR 信号的来源为 GTIMx_ETR 管脚、UARTx_RXD、VCx_OUT 及 LVD_OUT；可通过 CR1.ETRFLT 配置 ETR 信号的滤波参数；通过 CR0.ETRPOL 配置 ETR 信号的极性。

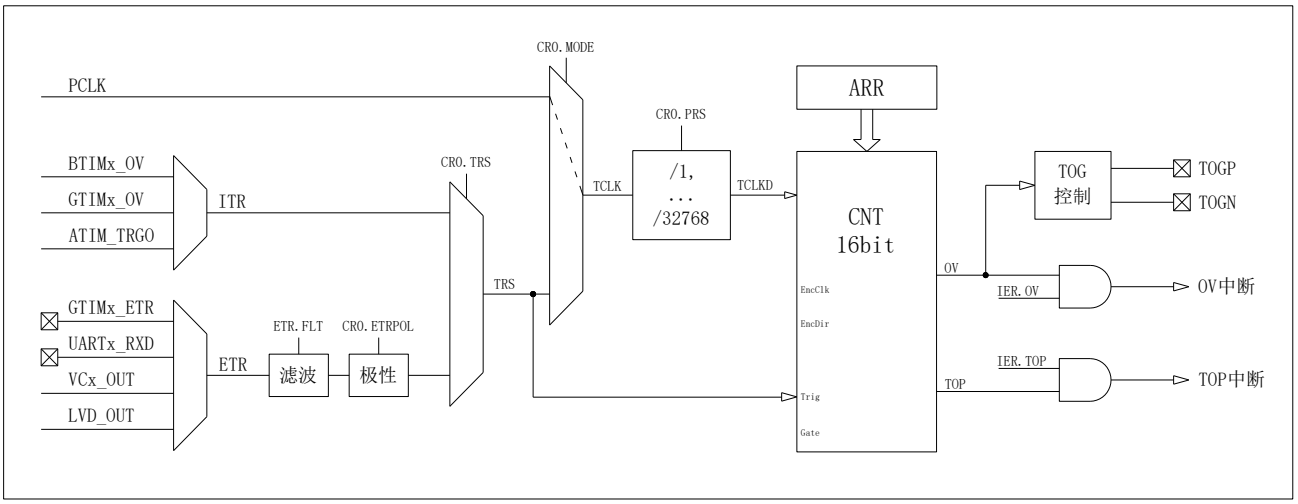

图 11-7 触发启动模式功能框图

定时器启动后，计数单元在 TCLKD（来自 PCLK）驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位，若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。每当出现有效的触发信号时，ISR.TI 标志会被硬件置位，若使能触发中断（IER.TI=1），则 CPU 将执行溢出中断服务程序。

该工作模式的定时时间间隔计算公式如下：

$$T = \frac{(ARR + 1) \times 2^{PRS}}{f_{PCLK}}$$

其中，f_{PCLK} 为 PCLK 的频率，PRS 为预分频系数，ARR 为重载值。

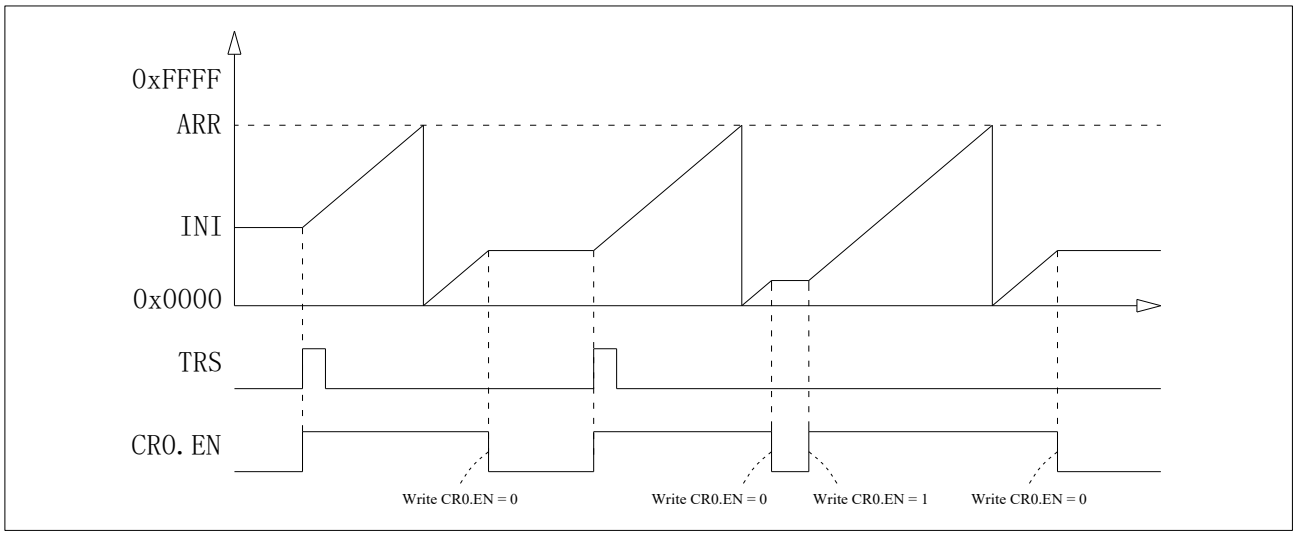

图 11-8 触发启动计数示意图

11.3.2.4 门控模式

当设置 CR0.MODE 为 0x03 时，GTIM 工作于门控模式。当 CR0.EN 为 1 且门控信号 Gate（来自 ETR）有效时，计数单元在 TCLKD（来自 PCLK）驱动下从初值开始加计数。可通过 ETR.FLT 配置 ETR 信号的滤波参数，通过 CR0.ETRPOL 配置 ETR 有效电平的极性。

用户可以通过 SYSCTRL_GTIMETR.GTIMxETR 选择 ETR 信号的来源为 GTIMx_ETR 管脚、UARTx_RXD、VCx_OUT 及 LVD_OUT；可通过 CR1.ETRFLT 配置 ETR 信号的滤波参数；通过 CR0.ETRPOL 配置 ETR 信号的极性。

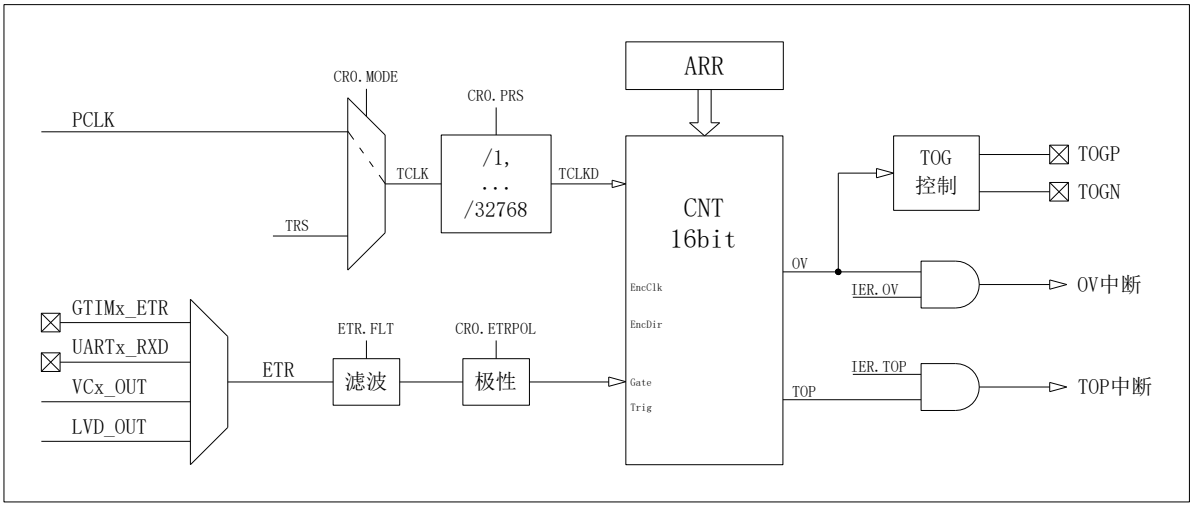

图 11-9 门控模式功能框图

当 CR0.EN 为 1 且门控信号有效时，计数单元在 TCLKD（来自 PCLK）的驱动下从初值开始加计数。当计数到 ARR 时产生 ARR 溢出，并重新从 0 开始计数。每次 ARR 溢出时 ISR.OV 标志会被硬件置位，若使能 ARR 溢出中断（IER.OV=1），则 CPU 将执行 ARR 溢出中断服务程序。当 CNT 的初值大于 ARR 时，则计数器会计数到 0xFFFF 时产生 TOP 溢出并置位 ISR.TOP 标志；若使能 TOP 溢出中断（IER.TOP=1），则 CPU 将执行 TOP 溢出中断服务程序。

在任意时候设置 CR0.EN 为 0 或门控信号无效时，计数单元立即暂停计数；当 CR0.EN 为 1 且门控信号有效时，计数单元继续计数。

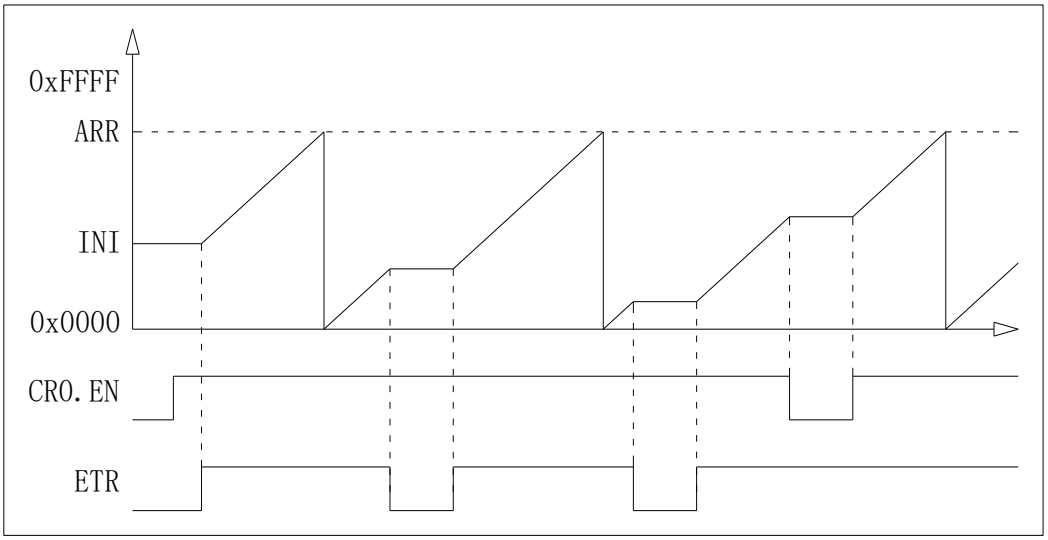

图 11-10 门控计数示意图

11.3.3输入捕获功能

当设置 CMMR.CCyM 为 1~3 时，相应的通道被配置为输入捕获模式，可用于测量输入到该通道的脉冲信号的宽度及频率。一旦该通道输入的信号边沿符合 CMMR.CCyM 的定义，计数器 CNT 的当前值立即被传输到比较捕获寄存器 CCRy 保存并置位 ISR.CCy 标志；若使能该通道的比较捕获中断（IER.CCy=1），则 CPU 将执行比较捕获中断服务程序。

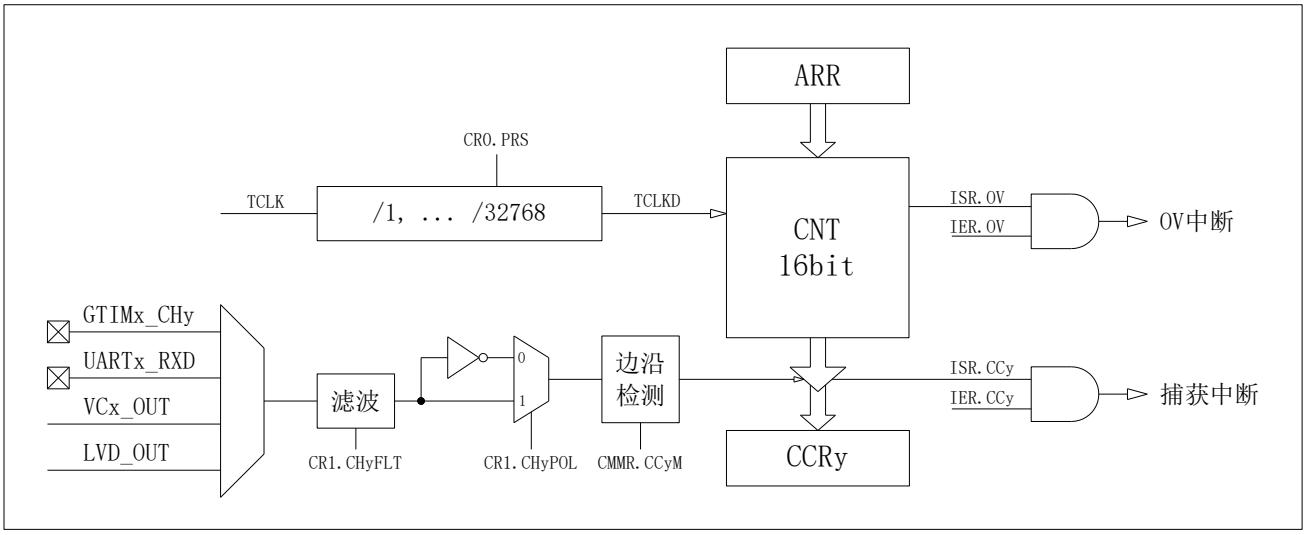

图 11-11 输入捕捉框图

通过 SYSCTRL_GTIMxCAP.Chy 位域可配置待捕获的信号的来源为 GTIMx_CHy 管脚、UARTx_RXD、VCx_OUT 及 LVD_OUT；通过 CR1.CHyFLT 位域可配置 CHy 信号的滤波参数；通过 CR1.CHyPOL 位域可配置待捕获信号的极性；通过 CMMR.CCyM 位域可配置需要执行捕获的输入信号的边沿为上升沿、下降沿或跳变沿。

当设置待捕获信号来自 UARTx_RXD 时，辅以适当的逻辑代码即可实现 UART 波特率自适应。

注：滤波模块详细介绍见【功能框图】小节。

下图展示了两个通道进行输入捕捉的时序；其中 CH1 在上升沿进行捕捉，CH2 在上升沿和下降沿同时进行捕捉。

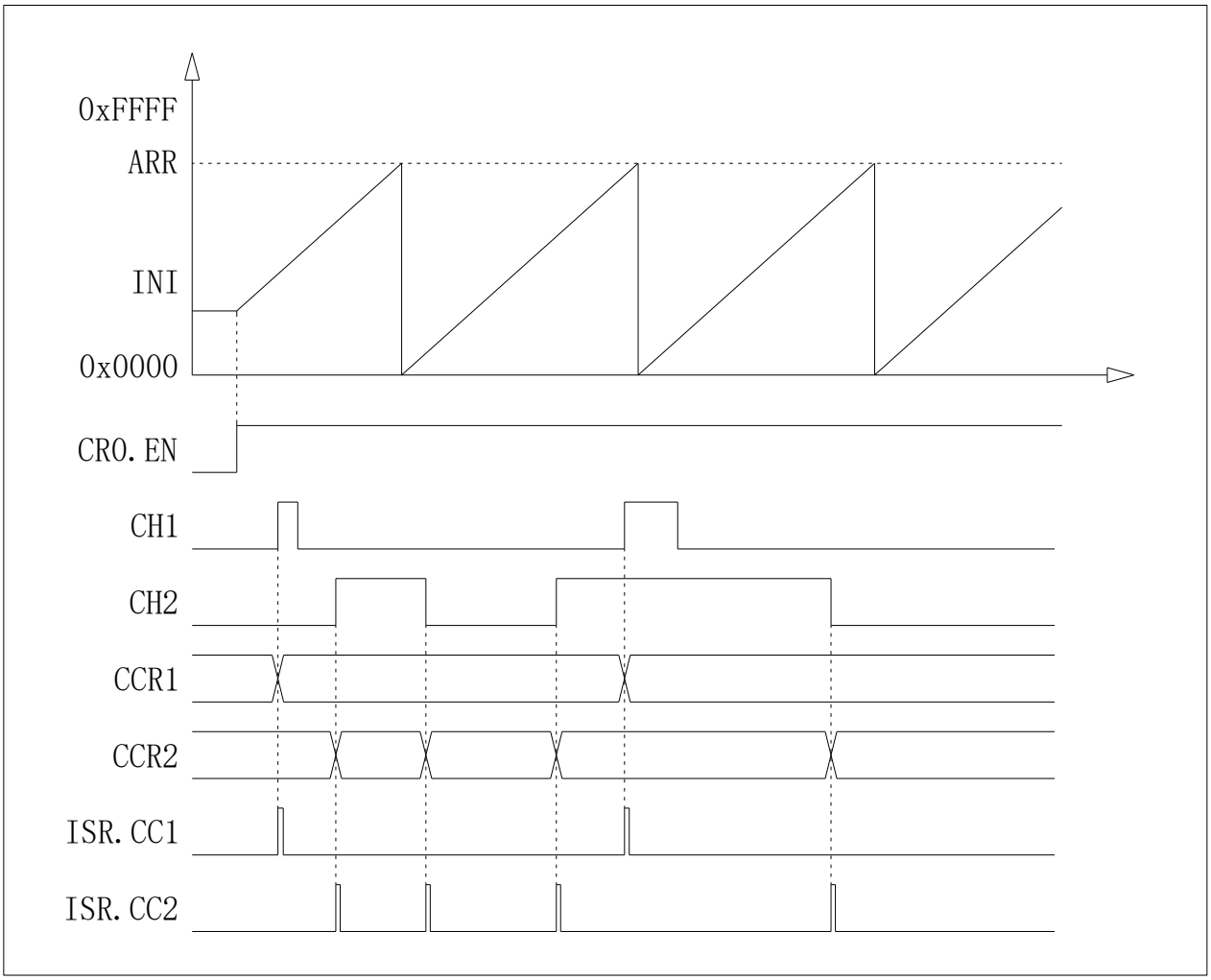

图 11-12 输入捕捉时序示意

11.3.4输出比较功能

当设置 CMMR.CCyM 为 8~15 时，相应的通道被配置为输出比较模式，可以输出多种波形。计数寄存器 GTIM_CNT 的值与比较捕获寄存器 GTIM_CCRy 的值持续进行比较，其比较结果用以控制该通道的输出信号：低电平、高电平、匹配时清 0、匹配时置 1、PWM 正向信号、PWM 反向信号。当 GTIM_CNT 的值与 GTIM_CCRy 的值相等时，ISR.CCy 标志被硬件置 1；若使能该通道的比较捕获中断（IER.CCy=1），则 CPU 将执行比较捕获中断服务程序。

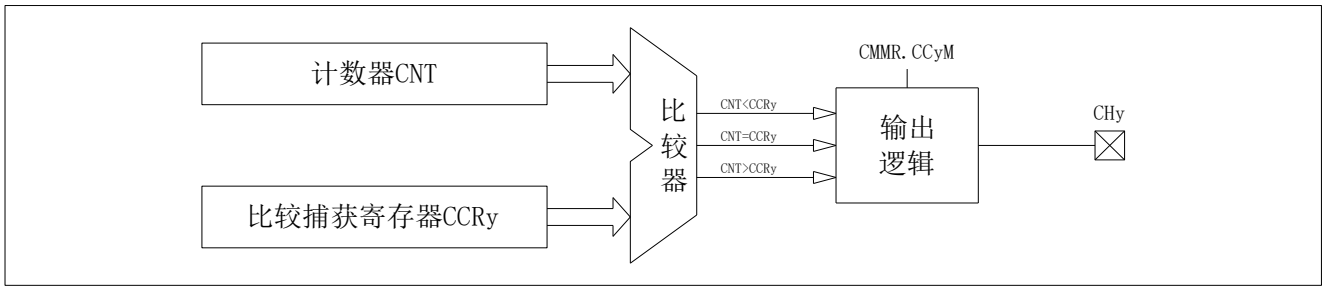

图 11-13 比较捕获电路示意

输出比较模式的配置及输出信号如下所示：

CMMR.CCyM	CHy通道输出信号
0x08	立即输出低电平
0x09	立即输出高电平
0x0A	在匹配时输出低电平
0x0B	在匹配时输出高电平
0x0E	输出正向PWM；CNT≥CCR时输出高电平
0x0F	输出反向PWM；CNT<CCR时输出高电平

正向 PWM 信号的频率为 f_{PCLK} / (ARR+1) / 2^{PRS}，占空比为(ARR – CCR + 1) / (ARR + 1)。当 CCR 为 0 时输出占空比为 100%；当 CCR 大于 ARR 时输出占空比为 0%。

反向 PWM 信号的频率为 f_{PCLK} / (ARR+1) / 2^{PRS}，占空比为 CCR / (ARR + 1)。当 CCR 为 0 时输出占空比为 0%；当 CCR 大于 ARR 时输出占空比为 100%。

下图展示了两个通道输出 PWM 波形的时序；其中 CH1 输出正向 PWM 波形，CH2 输出反向 PWM 波形。

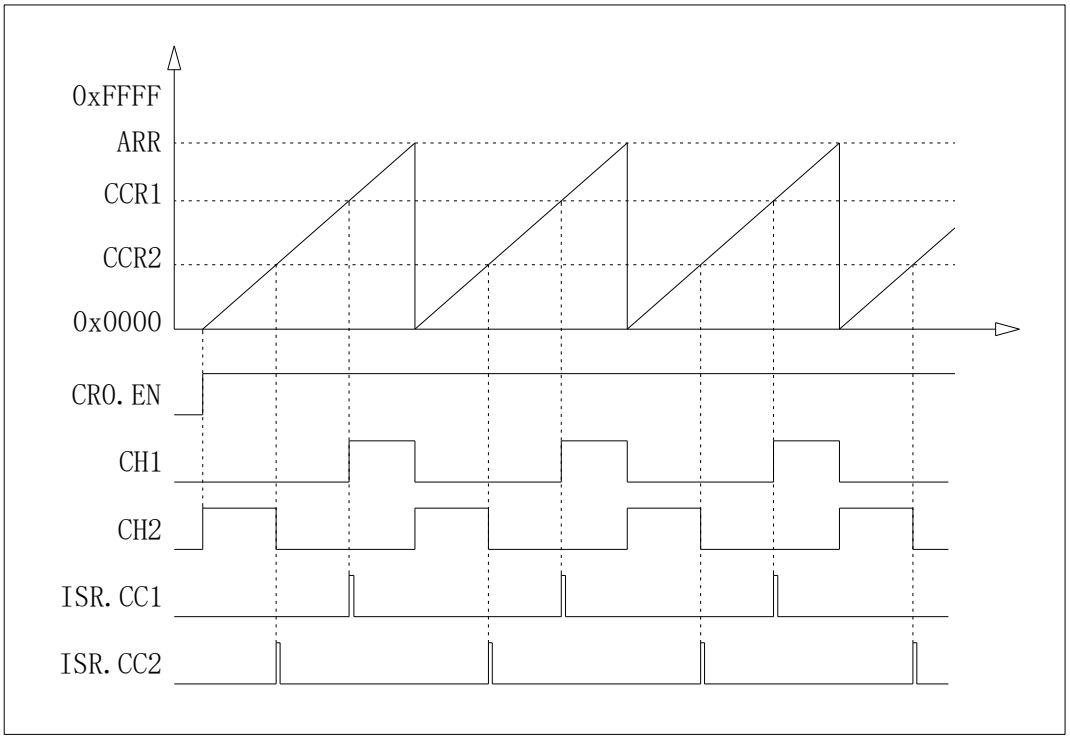

图 11-14 PWM 输出时序示意

11.3.5编码计数模式

当设置 CR0.ENCMODE 为 1~3 时，GTIM 工作于编码计数模式；可对输入到 CH1/CH2 的编码信号进行计数。该模式支持三种计数模式：CH1 边沿计数、CH2 边沿计数、CH1 及 CH2 边沿同时计数。通过 CR1.CHyPOL 及 CHyFLT 位域，可配置 CH1/CH2/CH3 的极性及滤波参数。通过 CR0.ENCRESET 及 CR0.ENCRELOAD 可配置 CH3 的边沿复位或重载计数器。

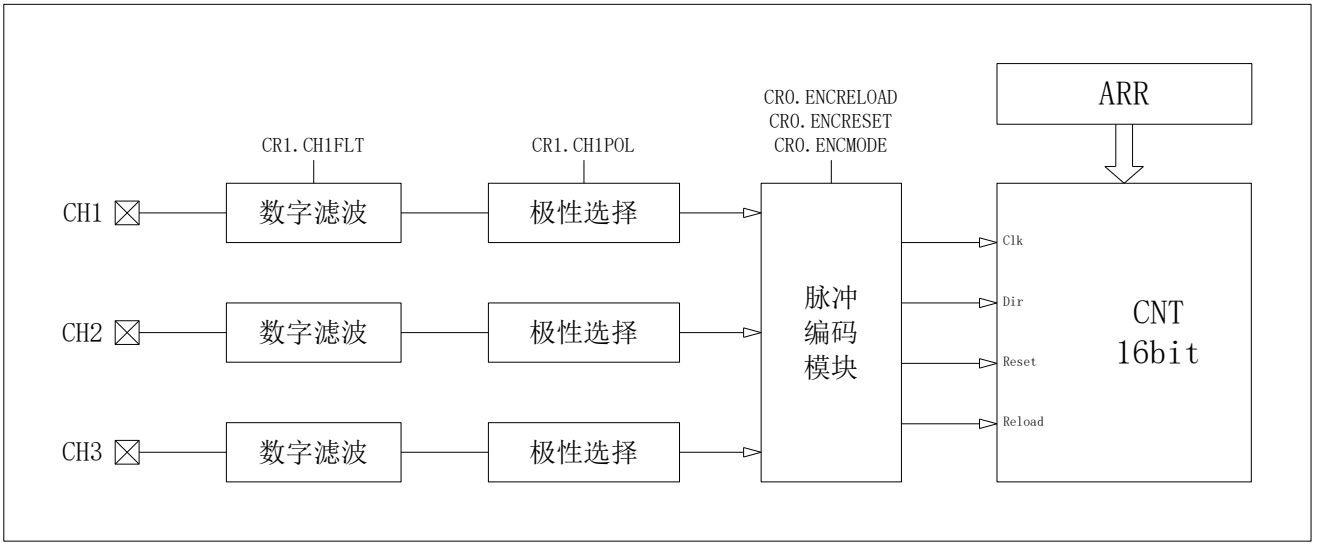

图 11-15 编码计数电路示意

当设置 CR0.EN 为 1 后计数器在脉冲编码模块输出的 Clk 的驱动下进行计数，计数方向由脉冲编码模块输出的 Dir 信号控制。当向上计数到 ARR 时产生上溢出，并重新从 0 开始计数；当向下计数到 0 时产生下溢出，并重新从 0xFFFF 开始计数。需要注意的是在编码计数模式时溢出中断无效，请勿在编码计数模式使用溢出中断。

通过读取 ISR.DIR 标志，可随时获取当前的计数方向为向下计数或向下计数；当计数方向改变时 ISR.DIRCHANGE 标志会被硬件置位，若使能计数方向改变中断（IER.DIRCHANGE=1），则 CPU 将执行计数方向改变中断服务程序。

下表给出了三种编码模式下编码器输入信号与计数方向的关系

编码模式	信号电平		CH1		CH2	
	CH2	CH1	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

下图是一个编码计数实例，显示了计数信号的产生和方向控制；还显示了双边沿计数模式下输入抖动是如何被抑制的。

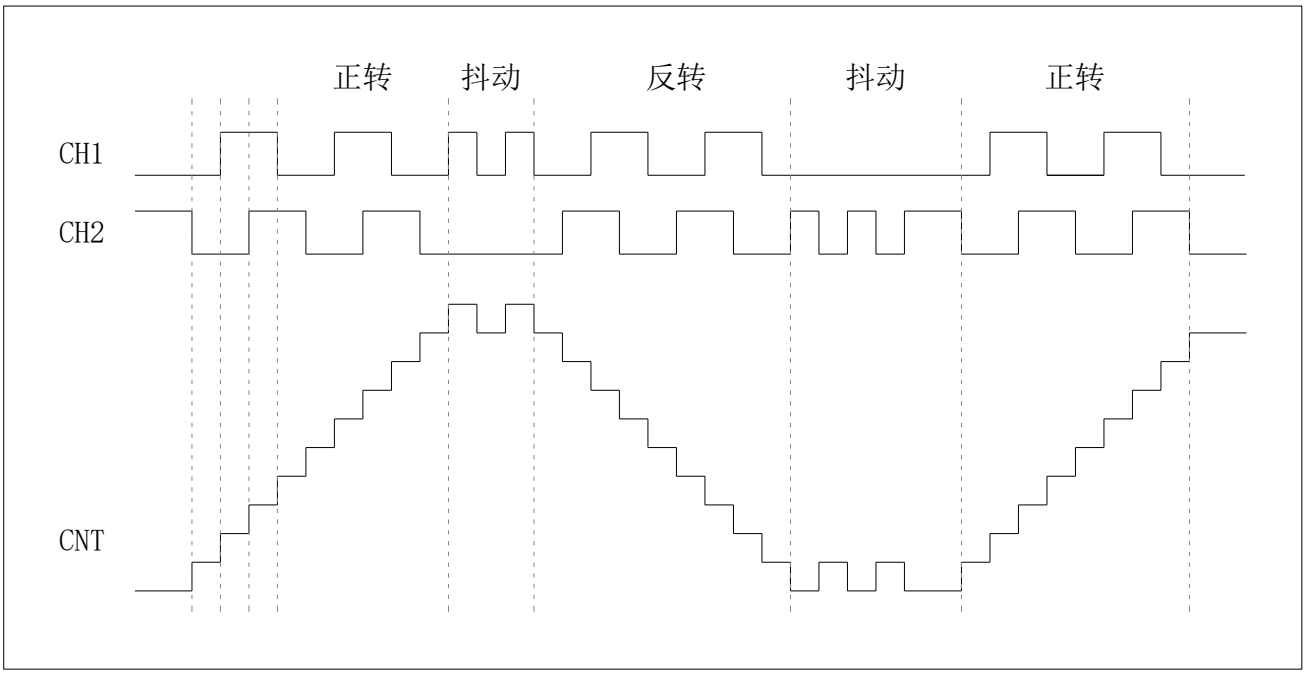

图 11-16 编码计数模式 3 计数实例

11.3.6 内部级联 ITR

当 GTIM 工作于计数模式且 TRS 来源为 ITR 时，即可将两个 TIM 级联使用。级联时，前一级 TIM 的溢出信号或主模式输出作为本级 TIM 的计数时钟。

级联可以将两个 TIM 组合成一个 32 位的定时器，也可以将前一级 TIM 作为后一级 TIM 的任意预分频器（1~65536）。

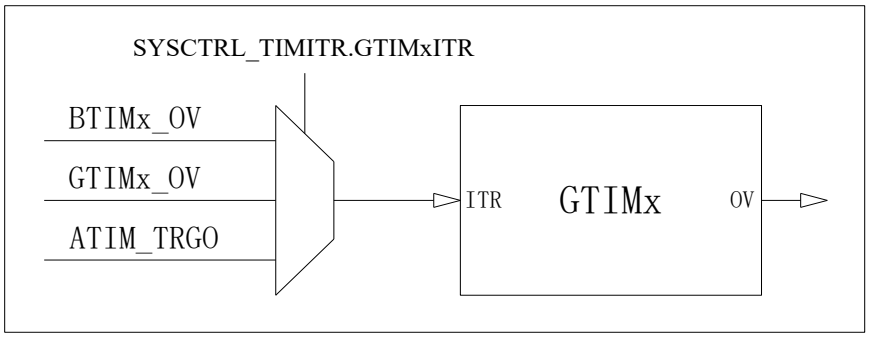

图 11-17 内部级联示意图

11.3.7 外设互联 ETR

通过 SYSCTRL_GTIMETR.GTIMxETR 位域可以配置 GTIM 的 ETR 来源为 GTIMx_ETR 管脚、UARTx_RXD、VCx_OUT 及 LVD_OUT。ETR 信号经滤波和极性选择后可用于触发或门控；结合 TIM 的功能即可实现 UART 波特率测量、VC 输出信号脉冲宽度测量、LVD 触发 TIM 启动以执行特殊流程等。

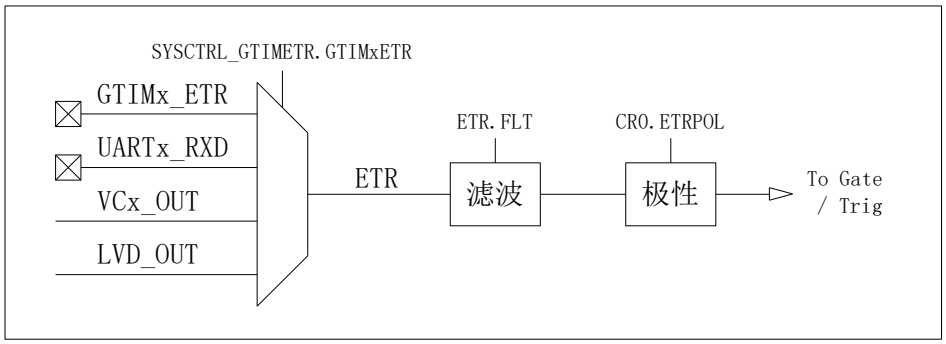

图 11-18 外设互联示意图

11.3.8 调试支持

GTIM 支持在调试模式下停止或继续计数，由 SYSCTRL_DEBUGEN.GTIMx 控制。

调试状态下，SYSCTRL_DEBUGEN.GTIMx 为 1 时，GTIMx 暂停工作；

调试状态下，SYSCTRL_DEBUGEN.GTIMx 为 0 时，GTIMx 继续工作。

11.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
GTIMx_ARR	GTIMx_Base + 0x300	重载寄存器
GTIMx_CNT	GTIMx_Base + 0x304	计数寄存器
GTIMx_CMMR	GTIMx_Base + 0x308	比较捕获控制寄存器
GTIMx_ETR	GTIMx_Base + 0x30C	外部触发控制寄存器
GTIMx_CR0	GTIMx_Base + 0x310	控制寄存器0
GTIMx_IER	GTIMx_Base + 0x314	中断使能寄存器
GTIMx_ISR	GTIMx_Base + 0x318	中断标志寄存器
GTIMx_ICR	GTIMx_Base + 0x31C	中断标志清除寄存器
GTIMx_CCR1	GTIMx_Base + 0x320	比较捕获寄存器1
GTIMx_CCR2	GTIMx_Base + 0x324	比较捕获寄存器2
GTIMx_CCR3	GTIMx_Base + 0x328	比较捕获寄存器3
GTIMx_CCR4	GTIMx_Base + 0x32C	比较捕获寄存器4
GTIMx_CR1	GTIMx_Base + 0x330	控制寄存器1

GTIM1_Base = 0x4000 0400

GTIMx_CR0 控制寄存器 0

Address offset: 0x310 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
20:19	ENCRELOAD	RW	编码计数模式，计数寄存器外部重载条件配置 00：无功能 01：CH3上升沿重载计数寄存器CNT 10：CH3下降沿重载计数寄存器CNT
18:17	ENCRESET	RW	编码计数模式，计数寄存器外部复位条件配置 00：无功能 01：CH3上升沿复位计数寄存器CNT 10：CH3下降沿复位计数寄存器CNT
16:15	ENCMODE	RW	编码计数模式使能配置 00：定时器功能由MODE位配置 01：编码计数模式1，CH1信号变化沿计数 10：编码计数模式2，CH2信号变化沿计数 11：编码计数模式3，CH1/CH2信号变化沿计数 注：在编码计数模式时溢出中断无效
14:11	PRSSTATUS	RO	分频电路当前正在使用的预分频系数
10:7	PRS	RW	预分频器分频系数配置 0：DIV1 8：DIV256 1：DIV2 9：DIV512 2：DIV4 10：DIV1024 3：DIV8 11：DIV2048 4：DIV16 12：DIV4096 5：DIV32 13：DIV8192 6：DIV64 14：DIV16384 7：DIV128 15：DIV32768
6	TOGEN	RW	TOG管脚输出使能控制 0：TOGP、TOGN输出电平均为0 1：TOGP、TOGN输出电平相反的信号
5	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
4	ETRPOL	RW	外部触发信号（ETR）极性选择 0：正向(触发模式上升沿有效，门控模式高电平有效) 1：反向(触发模式下降沿有效，门控模式低电平有效)
3	TRS	RW	触发源选择 0：ETR信号，其来源详见外部互联 1：ITR信号，其来源详见内部级联
2:1	MODE	RW	基本定时模式配置

			00：定时器模式，对PCLK进行计数 01：计数器模式，对TRS进行计数 10：触发模式，由TRS信号触发计数器启动对PCLK计数 11：门控模式，在ETR信号控制下对PCLK进行计数
0	EN	RW	定时器运行控制 0：定时器停止 1：定时器运行

GTIMx_CR1 控制寄存器 1

Address offset: 0x330 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	CH4POL	RW	CH4通道输入信号极性配置 0：CH4管脚反相信号送入内部电路 1：CH4管脚同相信号送入内部电路
14:12	CH4FLT	RW	CH4 输入信号滤波配置 功能描述详见 CH1FLT
11	CH3POL	RW	CH3通道输入信号极性配置 0：CH3管脚反相信号送入内部电路 1：CH3管脚同相信号送入内部电路
10:8	CH3FLT	RW	CH3 输入信号滤波配置 功能描述详见 CH1FLT
7	CH2POL	RW	CH2通道输入信号极性配置 0：CH2管脚反相信号送入内部电路 1：CH2管脚同相信号送入内部电路
6:4	CH2FLT	RW	CH2 输入信号滤波配置 功能描述详见 CH1FLT
3	CH1POL	RW	CH1通道输入信号极性配置 0：CH1管脚反相信号送入内部电路 1：CH1管脚同相信号送入内部电路
2:0	CH1FLT	RW	CH1输入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} = f_{PCLK}， N=2 010：f_{sample} = f_{PCLK}， N=4 011：f_{sample} = f_{PCLK}， N=6 100：f_{sample} = $f_{PCLK}/4$， N=4 101：f_{sample} = $f_{PCLK}/4$， N=6 110：f_{sample} = $f_{PCLK}/8$， N=4 111：f_{sample} = $f_{PCLK}/8$， N=6

GTIMx_ARR 重载寄存器

Address offset: 0x300　　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 注意：工作于编码计数模式时，ARR需设置为0xFFFF。

GTIMx_CNT 计数寄存器

Address offset: 0x304　　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	RW	定时器计数值

GTIMx_ETR 外部触发控制寄存器

Address offset: 0x30C　　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	FLT	RW	ETR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} = f_{PCLK}，N=2 010：f_{sample} = f_{PCLK}，N=4 011：f_{sample} = f_{PCLK}，N=6 100：f_{sample} = $f_{PCLK}/4$，N=4 101：f_{sample} = $f_{PCLK}/4$，N=6 110：f_{sample} = $f_{PCLK}/8$，N=4 111：f_{sample} = $f_{PCLK}/8$，N=6
3:0	RFU	-	保留位，请保持默认值

GTIMx_CMMR 比较捕获控制寄存器

Address offset: 0x308 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
15:12	CC4M	RW	CH4捕获比较模式配置 功能描述详见CC1M
11:8	CC3M	RW	CH3捕获比较模式配置 功能描述详见CC1M
7:4	CC2M	RW	CH2捕获比较模式配置 功能描述详见CC1M
3:0	CC1M	RW	CH1捕获比较模式配置 0000：无功能 0001：上升沿捕获 0010：下降沿捕获 0011：上下沿同时捕获 1000：强制输出低电平 1001：强制输出高电平 1010：在比较匹配时置0 1011：在比较匹配时置1 1110：PWM正向输出（CNT ≥ CCR输出高电平） 1111：PWM反向输出（CNT ＜ CCR输出高电平）

GTIMx_CCR1 比较捕获寄存器 1

Address offset: 0x320 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	CCR	RW	CH1通道比较/捕获值

GTIMx_CCR2 比较捕获寄存器 2

Address offset: 0x324 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	CCR	RW	CH2通道比较/捕获值

GTIMx_CCR3 比较捕获寄存器 3

Address offset: 0x328 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	CCR	RW	CH3通道比较/捕获值

GTIMx_CCR4 比较捕获寄存器 4

Address offset: 0x32C 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	CCR	RW	CH4通道比较/捕获值

GTIMx_IER 中断使能寄存器

Address offset: 0x314 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	DIRCHANGE	RW	编码器计数方向改变中断控制 0：禁止 1：使能
8:7	RFU	-	保留位，请保持默认值
6	CC4	RW	CH4捕获比较中断使能控制 0：禁止 1：使能
5	CC3	RW	CH3捕获比较中断使能控制 0：禁止 1：使能
4	CC2	RW	CH2捕获比较中断使能控制 0：禁止 1：使能
3	CC1	RW	CH1捕获比较中断使能控制 0：禁止 1：使能
2	TOP	RW	计数器TOP溢出中断使能控制 0：禁止TOP溢出中断 1：使能TOP溢出中断
1	TI	RW	触发中断使能控制 0：禁止触发中断 1：使能触发中断
0	OV	RW	计数器ARR溢出中断使能控制 0：禁止ARR溢出中断 1：使能ARR溢出中断

注：在编码计数模式溢出中断无效，请勿在编码计数模式使用溢出中断。

GTIMx_ISR 中断标志寄存器

Address offset: 0x318 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	DIR	RO	编码器当前计数方向标志 0：正向计数 1：反向计数
9	DIRCHANGE	RO	编码器计数方向改变中断标志 0：计数方向未改变 1：计数方向已改变
8:7	RFU	-	保留位，请保持默认值
6	CC4	RO	CH4比较捕获中断标志 0：未发生比较捕获事件 1：已发生比较捕获事件
5	CC3	RO	CH3比较捕获中断标志 0：未发生比较捕获事件 1：已发生比较捕获事件
4	CC2	RO	CH2比较捕获中断标志 0：未发生比较捕获事件 1：已发生比较捕获事件
3	CC1	RO	CH1比较捕获中断标志 0：未发生比较捕获事件 1：已发生比较捕获事件
2	TOP	RO	计数器TOP溢出标志 0：计数器未TOP溢出 1：计数器已TOP溢出
1	TI	RO	触发标志 0：未发生触发事件 1：已发生触发事件
0	OV	RO	计数器ARR溢出标志 0：计数器未ARR溢出 1：计数器已ARR溢出 注：当计数器值从ARR变为0时产生该标志

GTIMx_ICR 中断标志清除寄存器

Address offset: 0x31C Reset value: 0x0000 03FF

位域	名称	权限	功能描述
31:10	RFU	-	保留位，请保持默认值
9	DIRCHANGE	R1W0	编码器计数方向改变中断控制 W0：清除编码器计数方向改变中断标志 W1：无功能
8:7	RFU	-	保留位，请保持默认值
6	CC4	R1W0	CH4比较捕获中断标志清除 W0：清除比较捕获中断标志 W1：无功能
5	CC3	R1W0	CH3比较捕获中断标志清除 W0：清除比较捕获中断标志 W1：无功能
4	CC2	R1W0	CH2比较捕获中断标志清除 W0：清除比较捕获中断标志 W1：无功能
3	CC1	R1W0	CH1比较捕获中断标志清除 W0：清除比较捕获中断标志 W1：无功能
2	TOP	R1W0	计数器TOP溢出标志清除 W0：清除计数器TOP溢出标志 W1：无功能
1	TI	R1W0	触发标志清除 W0：清除触发标志 W1：无功能
0	OV	R1W0	计数器ARR溢出标志清除 W0：清除计数器ARR溢出标志 W1：无功能

12 高级定时器（ATIM）

12.1 概述

高级定时器(ATIM)由一个 16 位的自动重载计数器和 7 个比较单元组成，并由一个可编程的预分频器驱动。ATIM 支持 6 个独立的捕获/比较通道，可实现 6 路独立 PWM 输出或 3 对互补 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）。

12.2 主要特性

- 16 位向上、向下、向上/向下自动装载计数器
- 6 个独立通道输入捕获和输出比较
- PWM 输出，边沿或中央对齐模式
- 死区时间可编程的互补 PWM 输出
- 刹车功能
- 单脉冲模式输出
- 正交编码计数功能
- 支持中断和事件：
  - 计数器上溢、下溢
  - 捕获、比较事件
  - 捕获数据丢失
  - 刹车事件
  - 更新事件
  - 触发事件

12.3 功能描述

12.3.1 功能框图

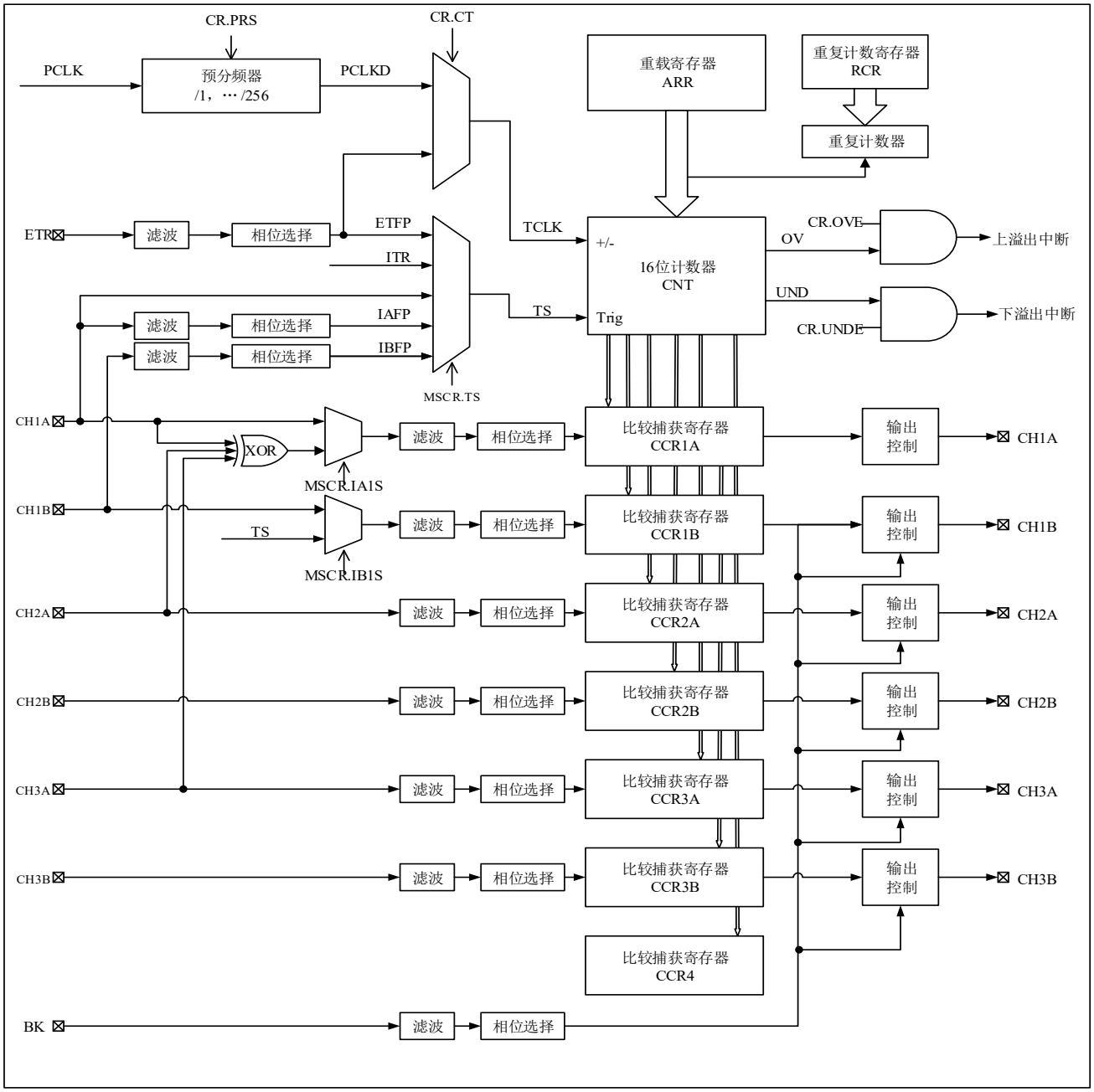

图 12-1 ATIM 功能框图

通过 16 位计数器和 7 路比较单元的结合，可以实现输入捕获、输出比较的功能，可以用来测量输入信号的脉宽、周期，可以输出 PWM 波形。

12.3.1.1 时钟源选择

计数单元的计数时钟源可选内部系统时钟 PCLK 或 ETR 输入信号，具体通过控制寄存器 ATIM_CR 的 CT 位域来选择。

当设置 ATIM_CR.CT 为 0 时，计数时钟源为内部系统时钟 PCLK，可通过 ATIM_CR.PRS 对内部时钟 PCLK 进行分频。

当设置 ATIM_CR.CT 为 1 时，计数时钟源为 ETR 输入信号（ETR 信号的具体来源参见片内外设互联 ETR 小节），可通过 ATIM_FLTR.FLTET 进行滤波控制，通过 ATIM_FLTR.ETP 选择 ETR 输入相位。

12.3.1.2 更新事件 UEV

更新事件 UEV（Update Event）的更新源通过控制寄存器 ATIM_CR 的 URS 位域设置，参见下表。

ATIM_CR.URS位域值	更新源
0	计数器上溢出、下溢出（且重复计数次数ATIM_RCR为0）； 软件置位UG；从模式复位
1	计数器上溢出、下溢出（且重复计数次数ATIM_RCR为0）

当发生更新事件时（内部 UEV 信号保持一个 PCLK 周期），ATIM 进行以下动作：

计数器被重置：

- 边沿对齐模式：向上计数时初始化为 0，向下计数时重新加载 ATIM_ARR
- 中央对齐模式：改变计数方向
- 如果使用了缓存寄存器功能，将更新 ATIM_ARR、ATIM_CHxCCRy 到其缓存寄存器
- 预分频器的计数器被清零，但不影响 ATIM_CR.PRS 中的预分频值
- 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_CR.UIE=1），将产生一个更新中断请求，设置 ATIM_ICR.UIE 为 0 清除该标志位。

注意，如果使用了重复计数器（ATIM_RCR.RCR 不为 0），只有在计数次数达到重复计数次数（ATIM_RCR.RCR 达到 0）时，才会产生更新事件 UEV。

12.3.1.3 计数模式

计数器可设置为向上计数（边沿对齐模式）、向下计数（边沿对齐模式），或向上向下双向计数（中央对齐模式）。

- 向上计数模式

在边沿对齐模式下设置控制寄存器 ATIM_CR 的 DIR 位为 0 时，计数器工作在向上计数模式。设置 ATIM_CR.EN 为 1 启动 ATIM，计数器 CNT 开始向上计数。当计数值到达重载值 ARR 后产生溢出信号 OV（溢出信号 OV 保持一个 PCLK 周期，然后自动清除）。下一个 PCLK 时，计数器被初始化为 0，同时计数器上溢出中断标志位 ATIM_ISR.OVF 被硬件置位，如果允许中断

（ATIM_CR.OVE=1），将产生中断请求，设置 ATIM_ICR.OVF 为 0 清除该标志位。

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数（ATIM_RCR.RCR）时，才会产生更新事件 UEV，否则每次计数器上溢出时都会产生更新事件 UEV。

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

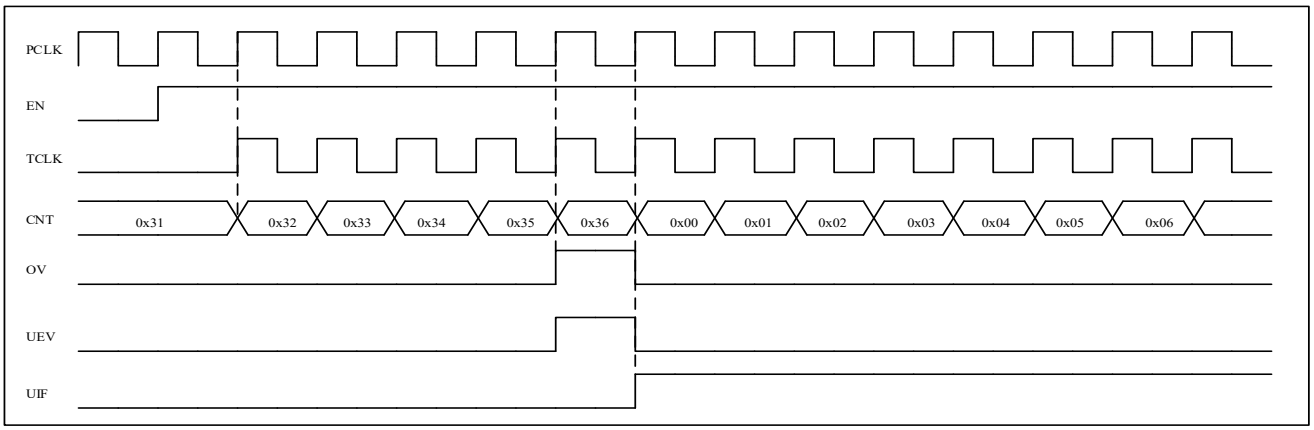

图 12-2 分频系数 1，向上计数

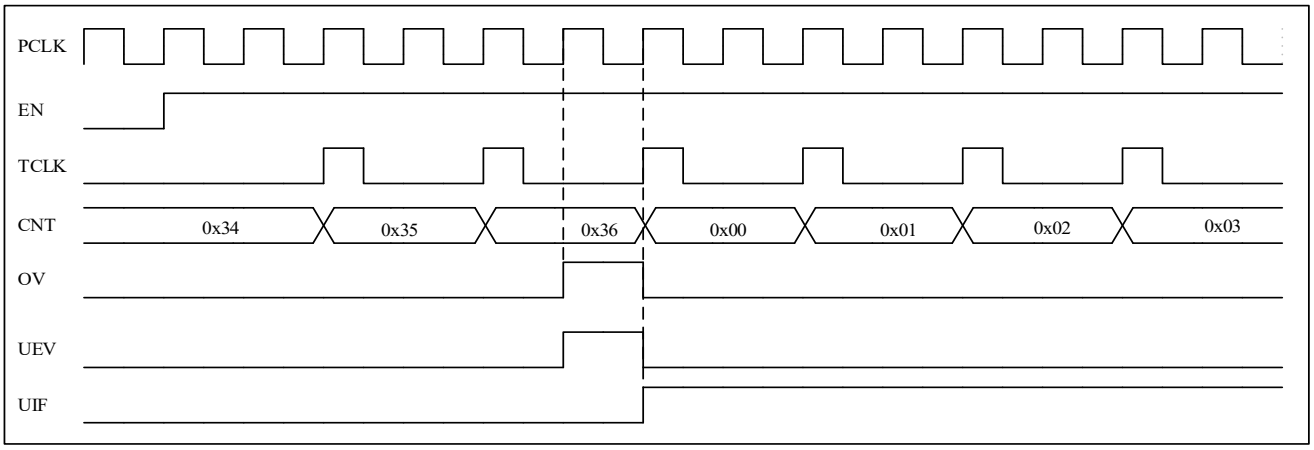

图 12-3 分频系数 2，向上计数

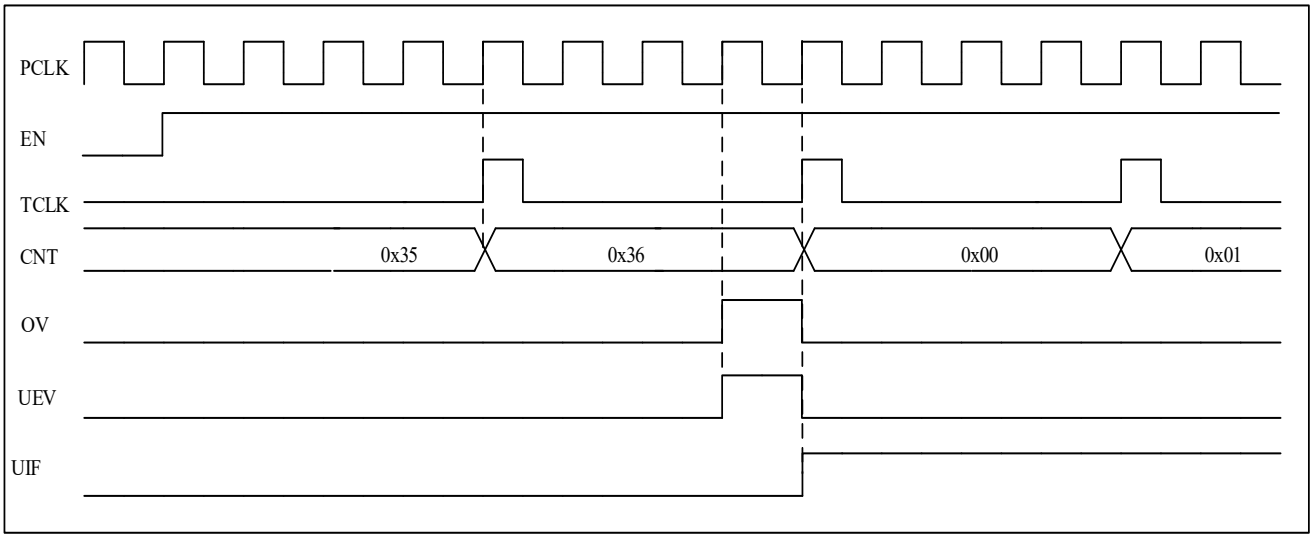

图 12-4 分频系数 4，向上计数

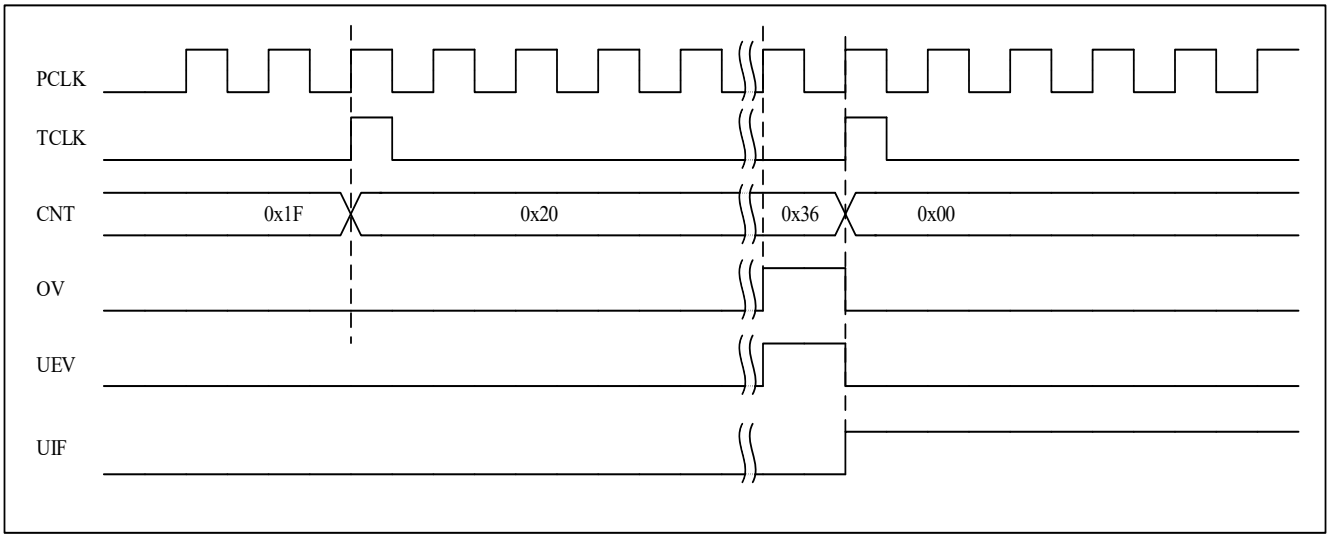

图 12-5 分频系数 N，向上计数

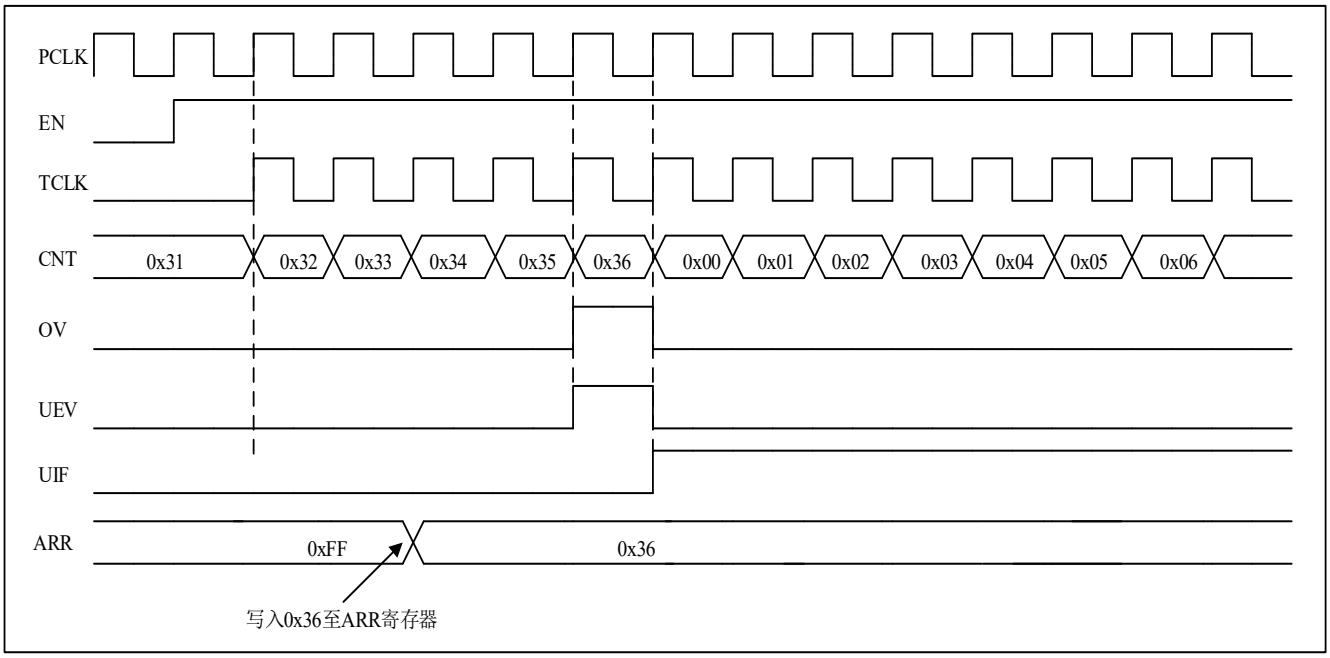

图 12-6 禁止缓存重载时更新时序

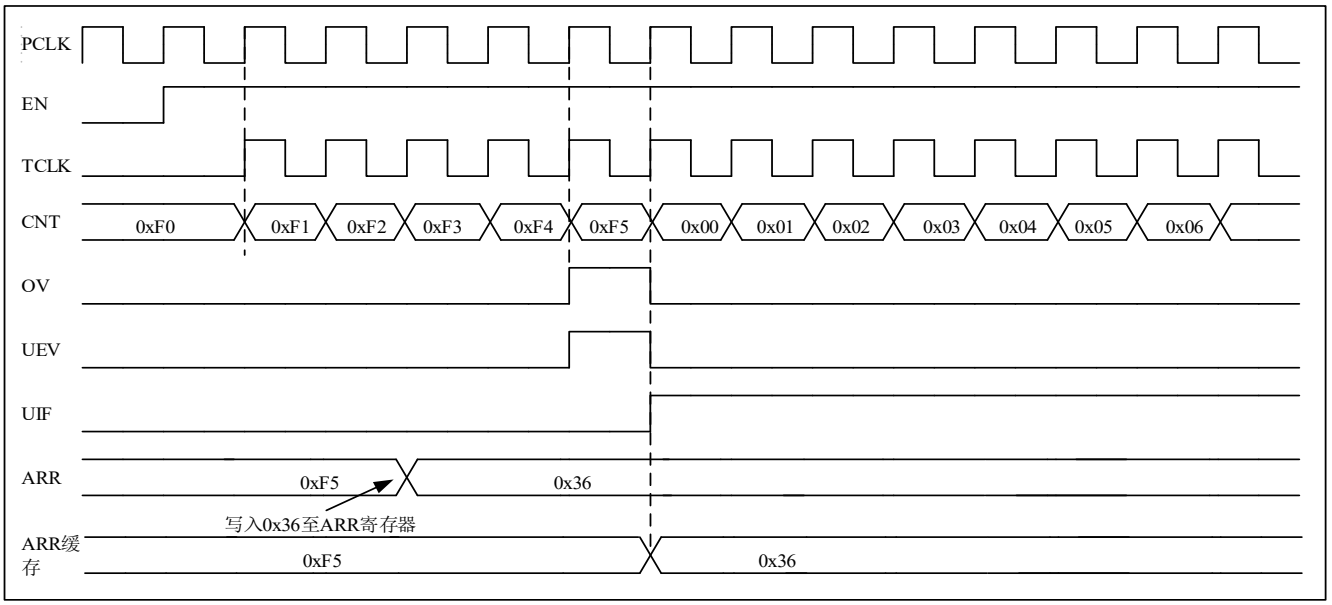

图 12-7 使能缓存重载时更新时序

- 向下计数模式

在边沿对齐模式下设置控制寄存器 ATIM_CR 的 DIR 位为 1 时，计数器工作在向下计数模式。设置 ATIM_CR.EN 为 1 启动 ATIM，硬件自动加载 ARR 到计数器 CNT，计数器开始向下计数。当计数值到达 0 后产生溢出信号 UND（溢出信号 UND 保持一个 PCLK 周期，然后自动清除）。溢出后一个 PCLK，计数器重装载 ARR 的值，计数器下溢出中断标志位 ATIM_ISR.UNDF 被硬件置位，如果允许中断（ATIM_CR.UNDE=1），将产生中断请求，设置 ATIM_ICR.UNDF 为 0 清除该标志位。

如果使用了重复计数器功能，在向下计数达到设置的重复计数次数（ATIM_RCR.RCR）时，才会产生更新事件 UEV，否则每次计数器下溢出时都会产生更新事件。

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

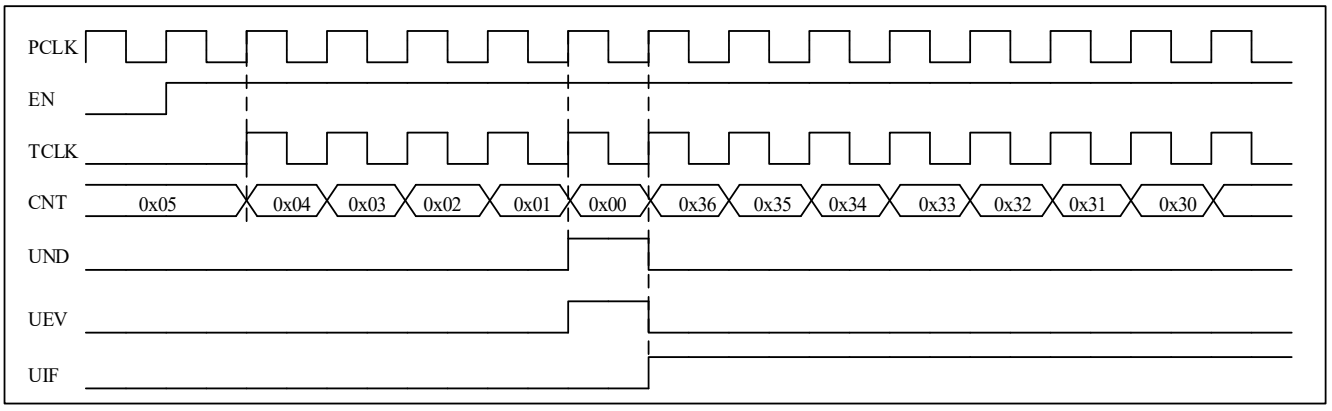

图 12-8 分频系数 1，向下计数

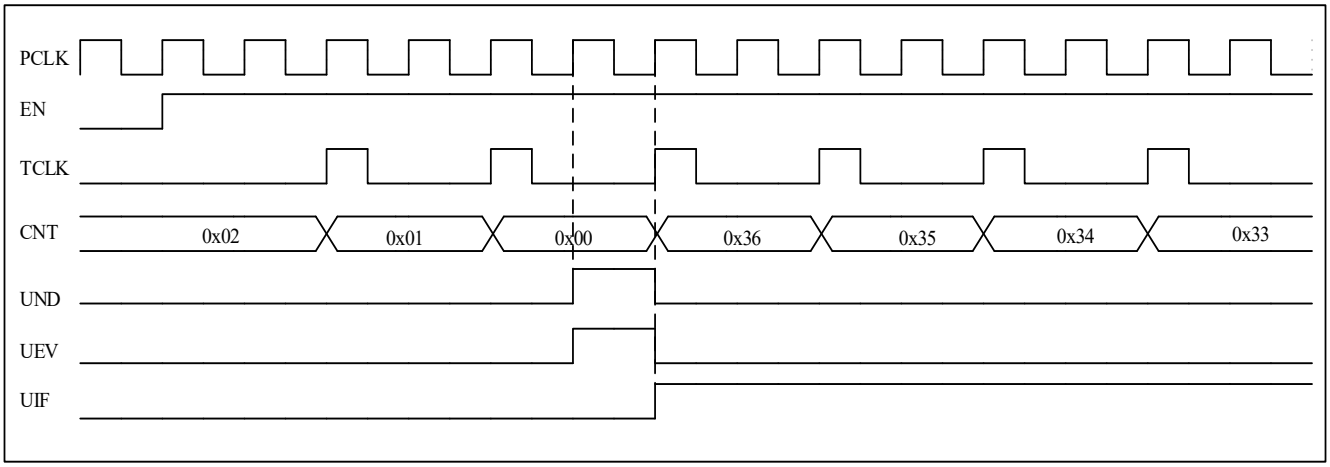

图 12-9 分频系数 2，向下计数

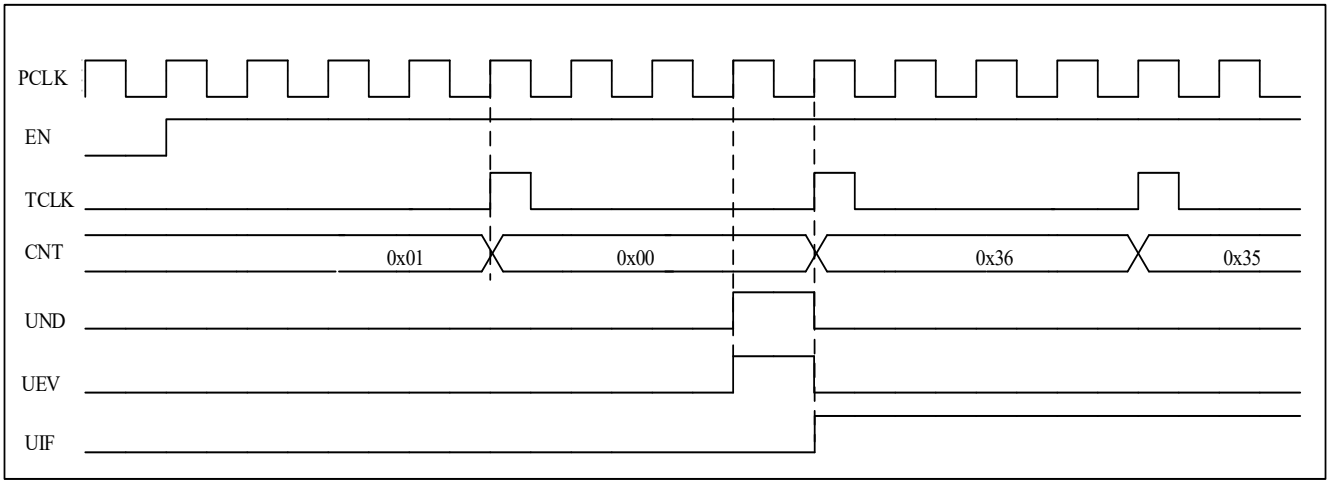

图 12-10 分频系数 4，向下计数

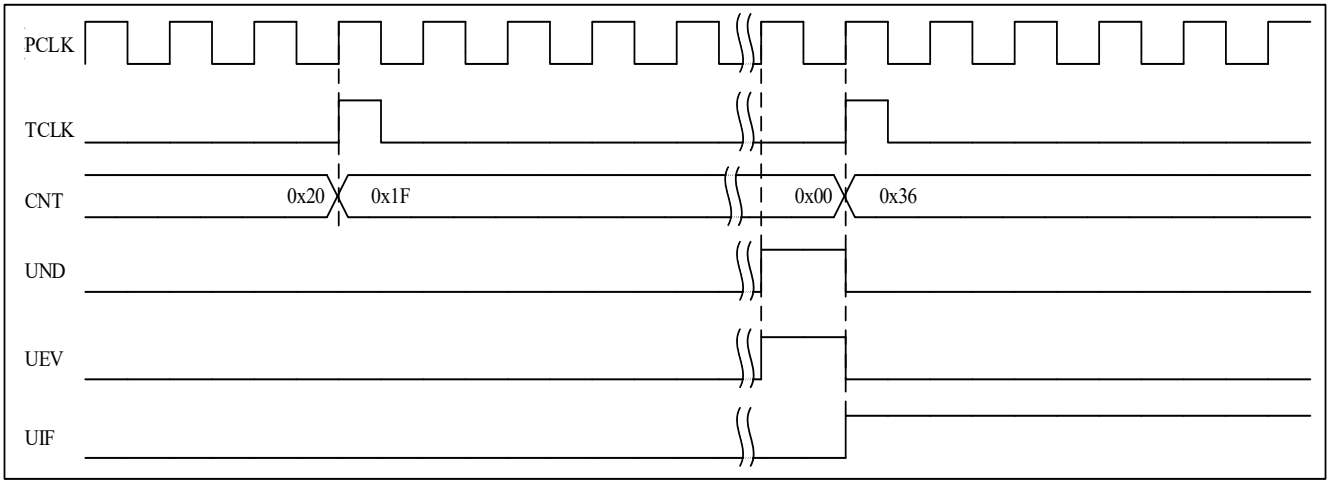

图 12-11 分频系数 N，向下计数

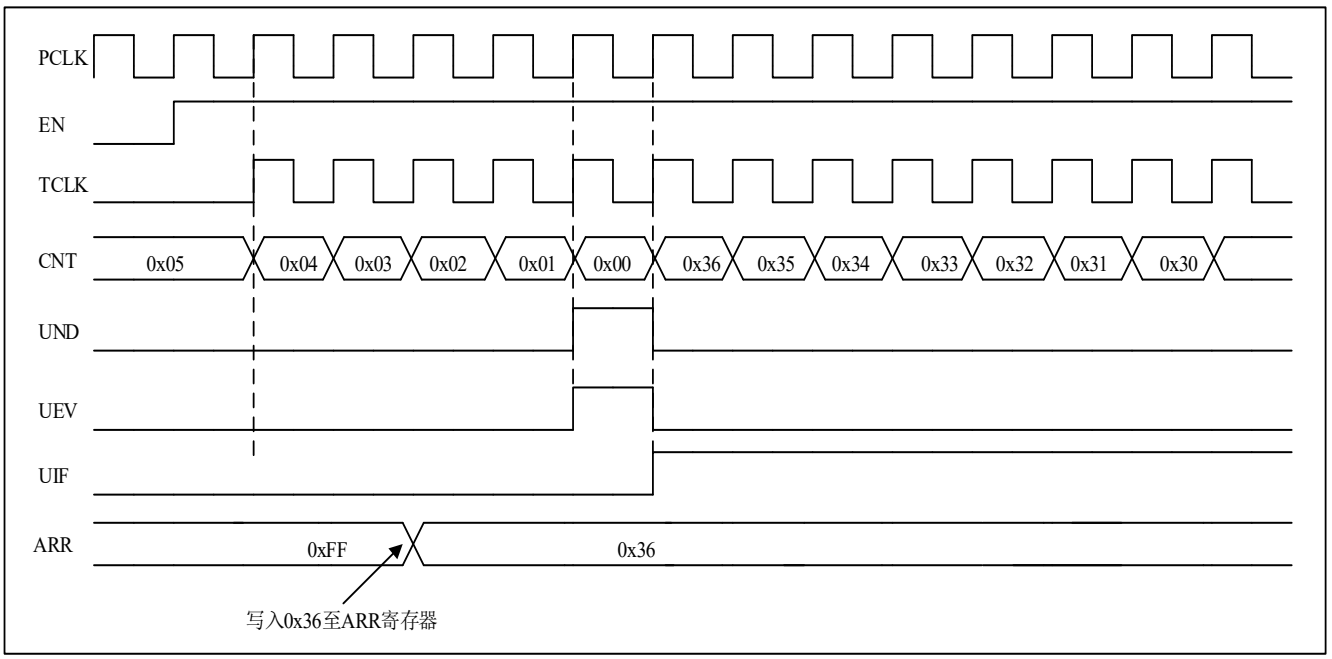

图 12-12 禁止缓存重载时更新时序

- 中央对齐模式（向上/向下计数）

在中央对齐模式下，计数器从 0 开始计数到重载值 ARR，产生一个计数器上溢出事件，然后向下计数到 0 并且产生一个计数器下溢出事件，然后再从 0 开始重新向上计数。

中央对齐模式下，控制寄存器 ATIM_CR 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。从其他模式切换到中央对齐模式时 DIR 位自动清 0。

如果使用了重复计数器功能，在向上/向下计数达到设置的重复计数次数（ATIM_RCR.RCR）时，才会产生更新事件 UEV，否则每次计数器上溢出/下溢出时都会产生更新事件。

以下是计数器在不同时钟频率下的操作示例：

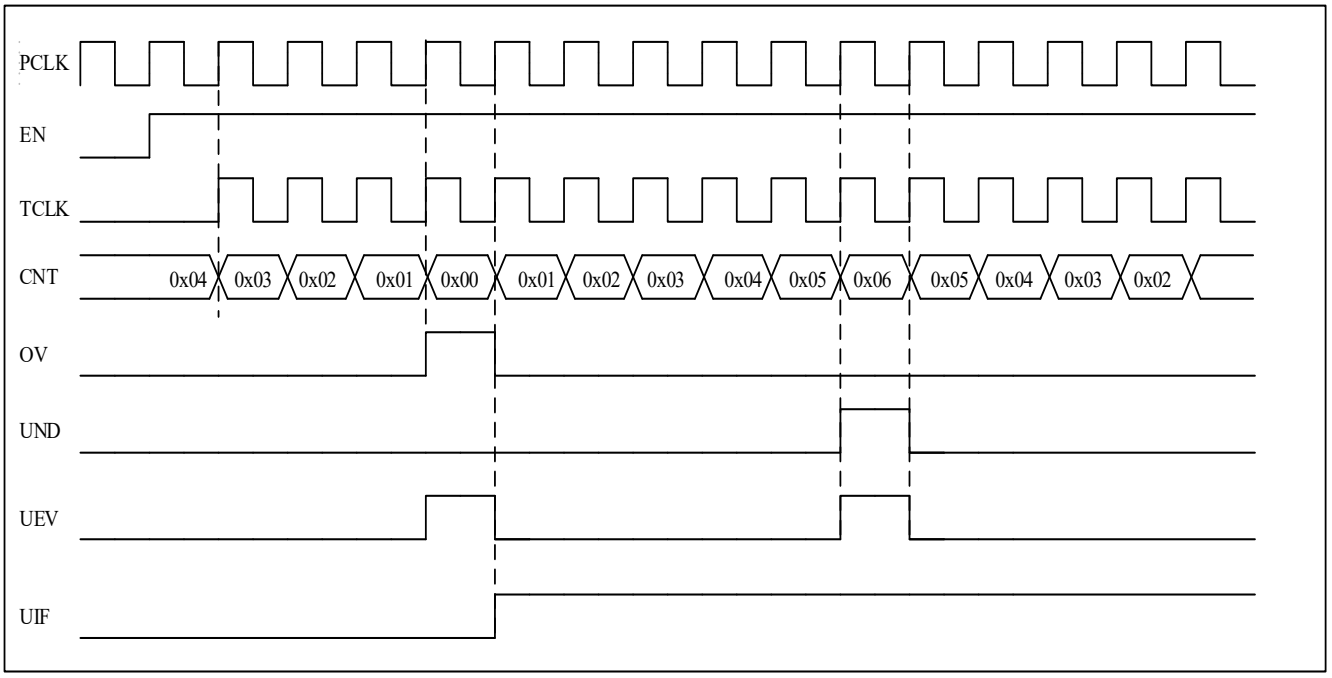

图 12-13 分频系数 1，中央对齐计数

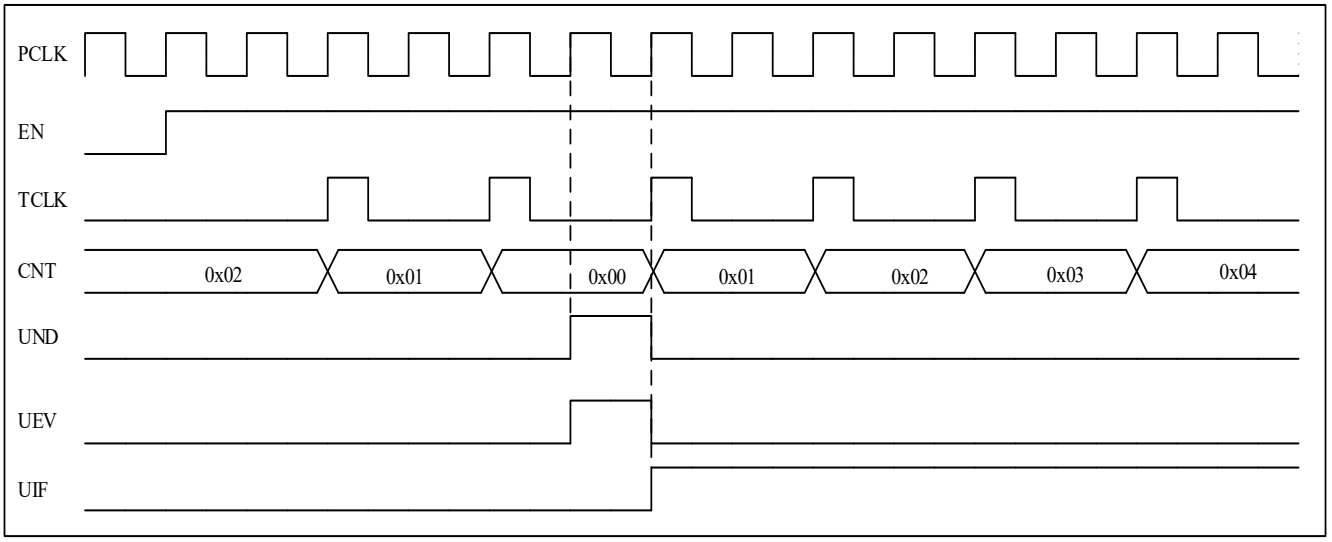

图 12-14 分频系数 2，中央对齐计数

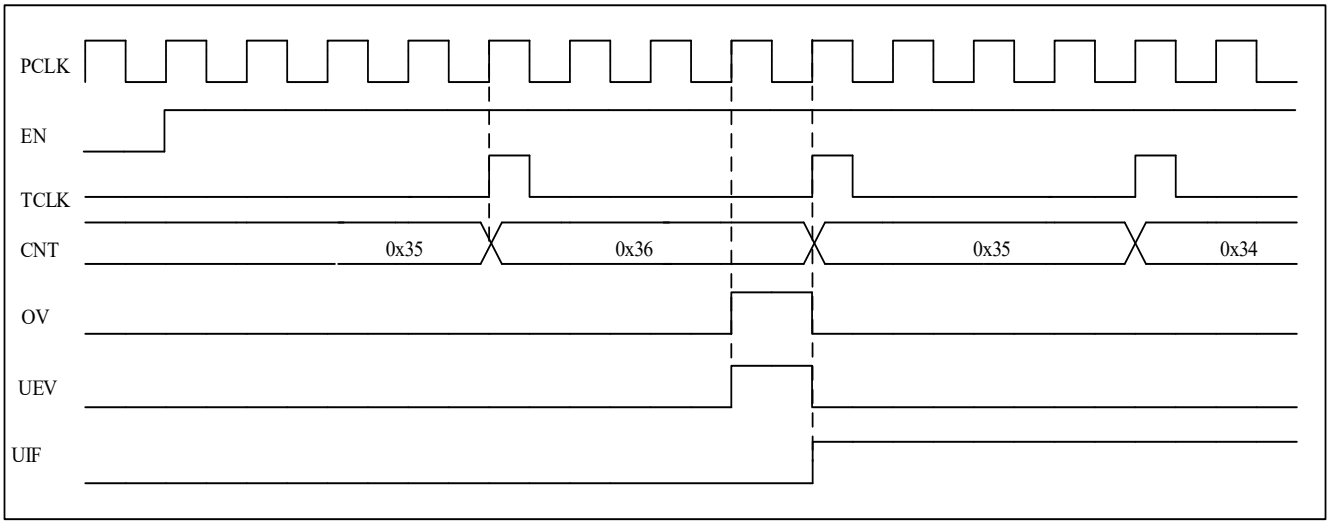

图 12-15 分频系数 4，中央对齐计数

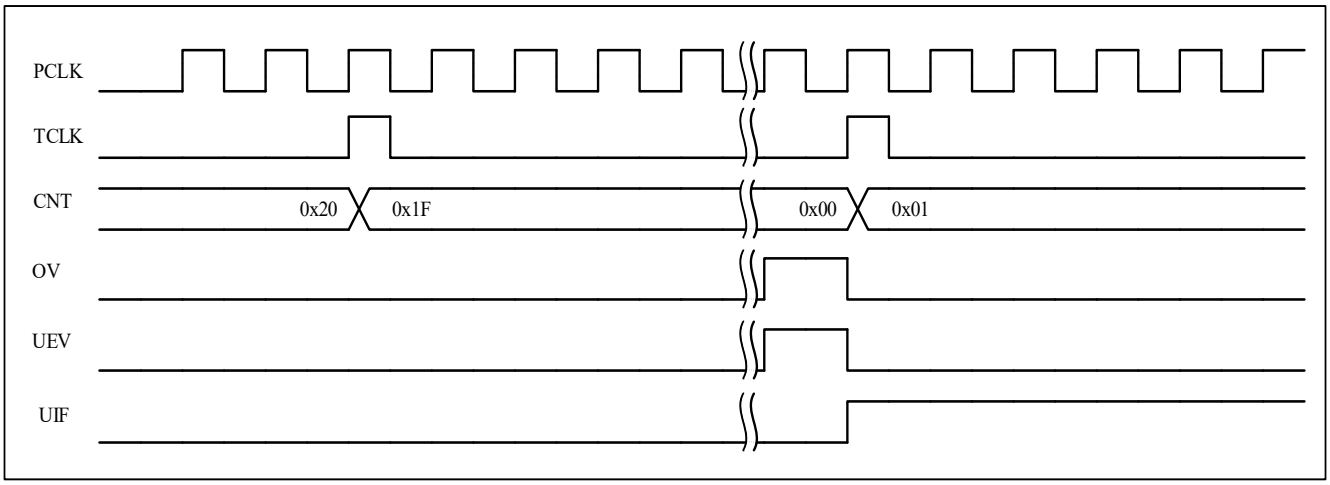

图 12-16 分频系数 N，中央对齐计数

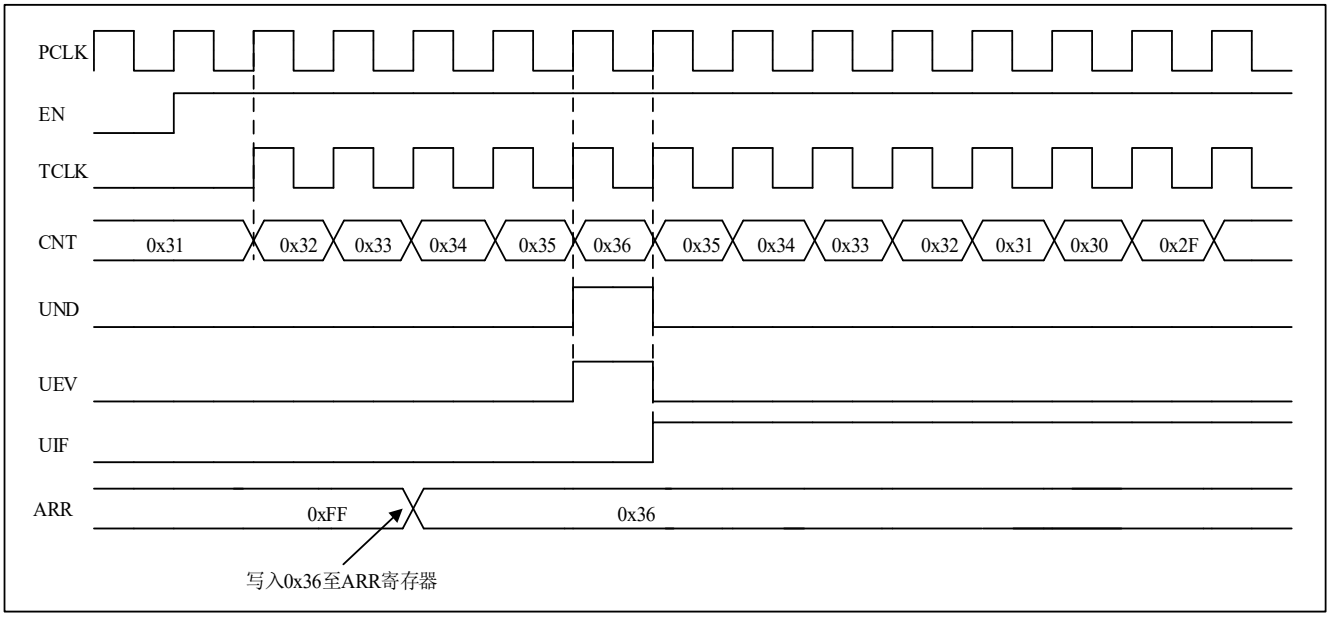

图 12-17 禁止缓存重载时更新时序

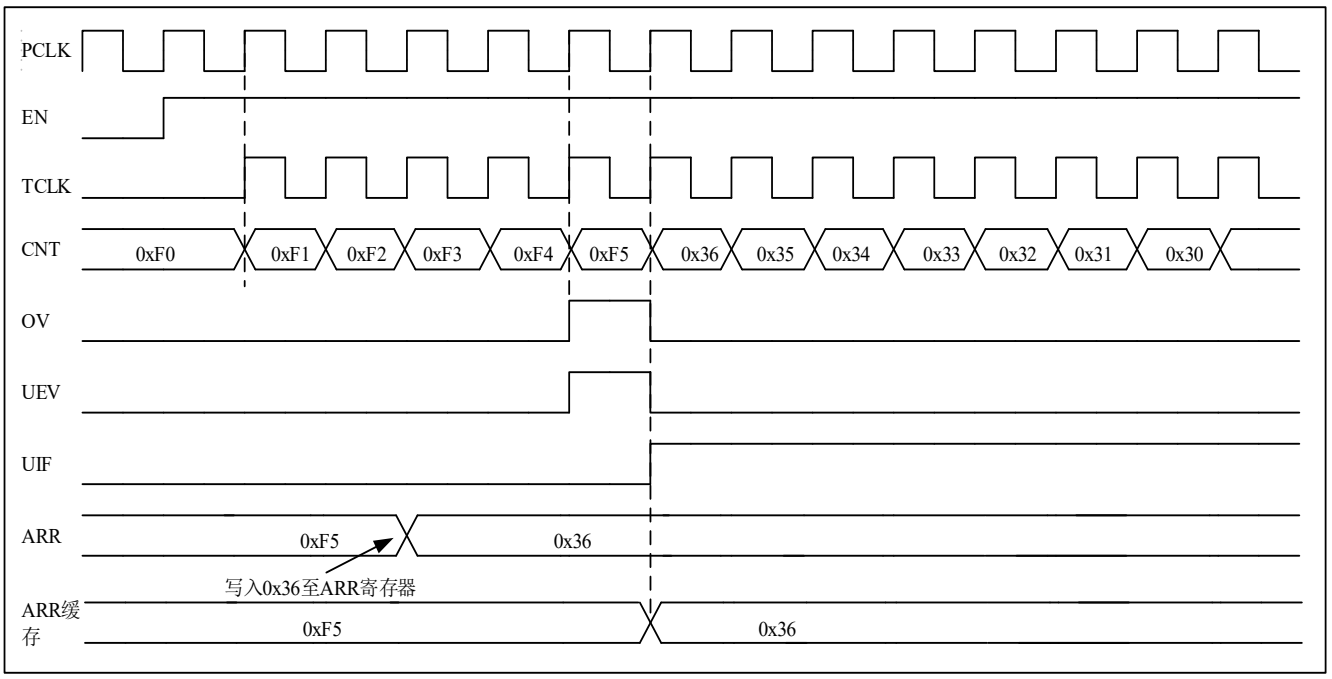

图 12-18 使能缓存重载时更新时序

12.3.1.4 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_CR 的 BUFPEN 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 将会立即更新缓存寄存器。

当定时器处于运行状态且缓存功能有效时，更新重载寄存器 ARR 将不会立即更新缓存寄存器，仅当有更新事件 UEV 时才会将重载寄存器 ARR 的值更新到缓存寄存器中。

12.3.1.5 重复计数器

启动 ATIM 会自动加载重复计数寄存器 ATIM_RCR 的 RCR 到重复计数器，在主计数器产生溢出时根据 ATIM_RCR 寄存器的 UD 和 OV 使能位自动减一，具体请参考下表：

主计数器溢出类型	工作模式	CR.DIR	RCR.UD	RCR.OV	重复计数器动作
上溢	边沿对齐模式	0，向上计数	-	0	减一
			-	1	不计数
	中央对齐模式	-	-	0	减一
			-	1	不计数
下溢	边沿对齐模式	1，向下计数	0	-	减一
			1	-	不计数
	中央对齐模式	-	0	-	减一
			1	-	不计数

使用重复计数器功能（ATIM_RCR 的 RCR 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 RCR 的内容将立即加载到重复计数器。

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。

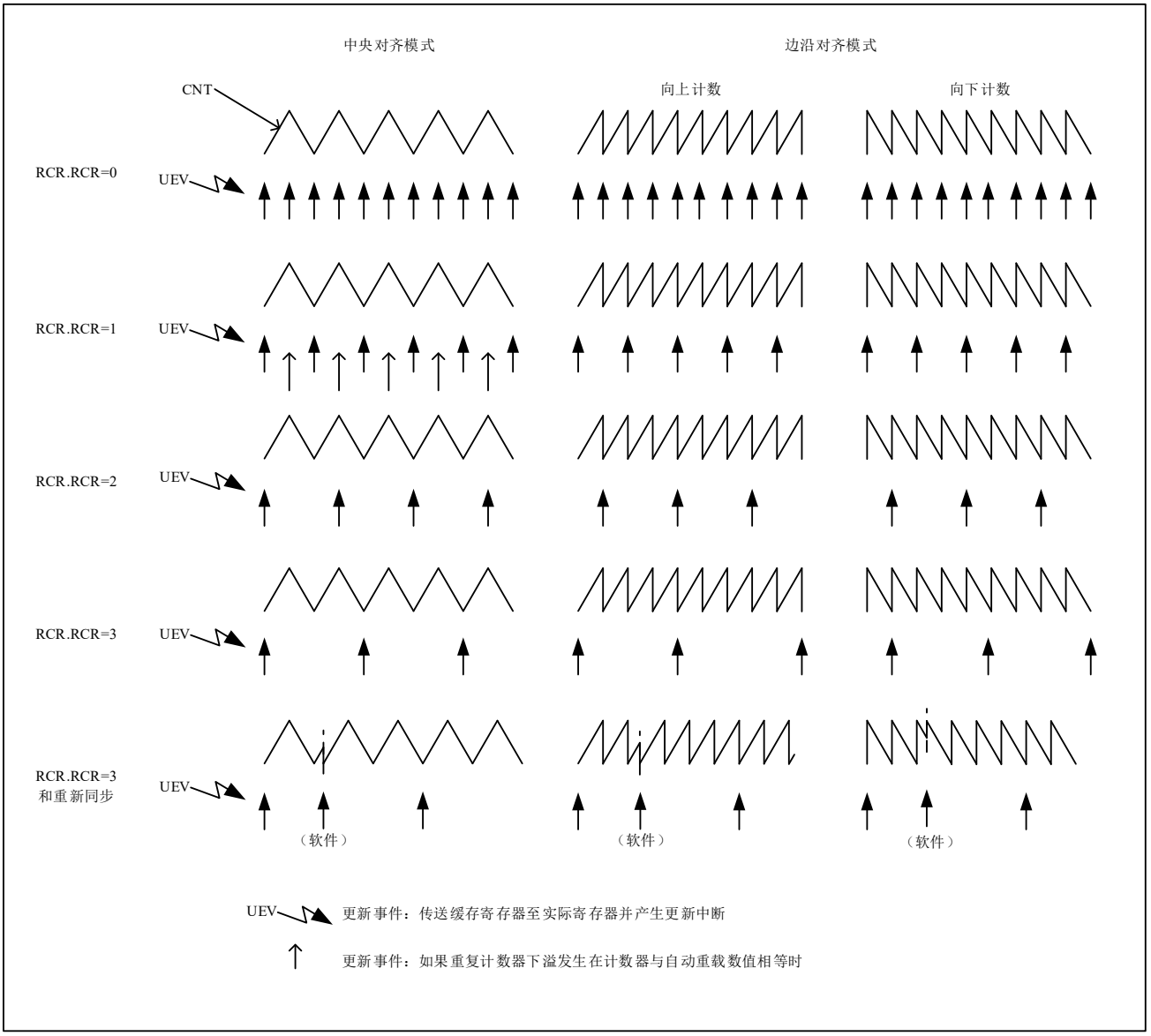

图 12-19 重复计数示例

12.3.1.6 比较捕获通道

ATIM 有 3 个独立比较捕获通道 CH1、CH2、CH3，每个通道都有 A、B 两路比较捕获寄存器及其缓存寄存器，以及输入信号处理单元（数字滤波和相位检测），比较单元，和匹配输出单元。因此 ATIM 可实现 6 路输入捕获，或者 6 路独立 PWM 输出，或 3 对互补 PWM 输出。ATIM 的通道 CH4 为芯片内部通道，无外部管脚，只有一路比较捕获寄存器(ATIM_CH4CCR)，且只能用于比较，不能用来捕获。

12.3.2输入捕获功能

12.3.2.1 输入捕获

边沿对齐模式和中央对齐模式都支持输入捕获功能。设置通道 x 控制寄存器 ATIM_CHxCR 的 CSy 位域为 1，使通道 CHxy（x=1~3，y=A,B）工作在输入捕获模式，可通过软件或硬件触发输入捕获。

软件触发：设置 ATIM_CHxCR.CCGy 为 1 时，立即软件触发一次捕获，主计数寄存器 ATIM_CNT 的值被锁存到对应通道的比较捕获寄存器 ATIM_CHxCCRy 中，完成一次捕获，捕获完成后 CCGy 位被硬件自动清零。

硬件触发：当检测到通道 CHxy 上的有效边沿后，主计数寄存器 ATIM_CNT 的值被锁存到对应通道的比较捕获寄存器 ATIM_CHxCCRy 中，完成一次捕获。触发捕获的有效边沿通过 ATIM_CHxCR 寄存器的 BKSy 位域来选择，参见下表：

高级定时器	通道	BKSy位域值	捕获触发条件
ATIM_CHxCR (x=1，2，3)	BKSy (y=A,B)	00	禁止
		01	上升沿时捕获
		10	下降沿时捕获
		11	上下沿时均捕获

当发生一次捕获时，对应通道的捕获中断标志位 ATIM_ISR.CxyF 被硬件置位，如果使能了中断（ATIM_CHxCR.CIEy=1）产生中断请求；如果发生捕获事件时，ATIM_ISR.CxyF 标志位已经为高，那么重复捕获标志位 ATIM_ISR.CxyE 将被硬件置位。设置 ATIM_ICR.CxyF 为 0 清除 ATIM_ISR.CxyF 标志位，设置 ATIM_ICR.CxyE 为 0 清除 ATIM_ISR. CxyE 标志位。

注意：即使 ATIM 未启动，ATIM_CNT 没有开始计数，当检测到通道 CHxy 的有效边沿后，仍会触发捕获并执行捕获动作。

12.3.2.2 PWM 输入模式

PWM 输入模式是输入捕获模式的一个应用。设置通道 x 控制寄存器 ATIM_CHxCR 的 BKSy 位域为 0x03，使定时器在 PWM 输入信号的上升沿和下降沿都发生捕获，经过三次捕获后，可计算出 PWM 信号的脉宽和周期。

下图是 CH3B 通道的 PWM 信号测量示意图。

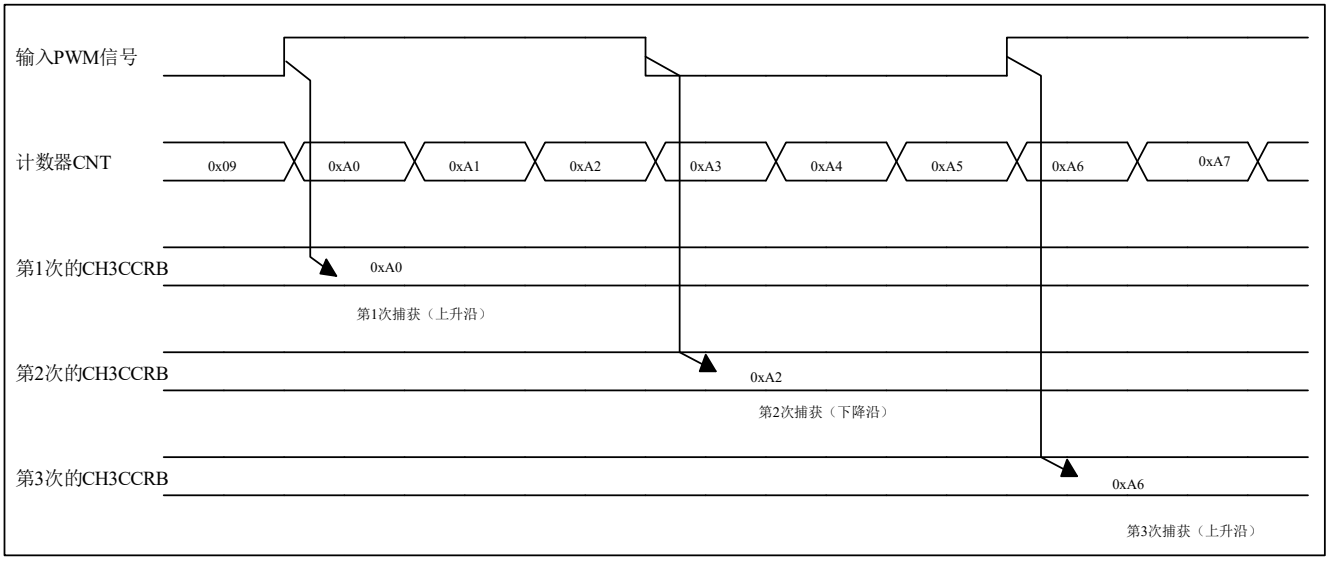

图 12-20 PWM 测量示意图

12.3.2.3 输入捕获触发源

ATIM 的输入捕获触发源可选择外部 CHxy 管脚输入信号，或者内部触发 TS 信号。

通道 CH2A、CH2B、CH3A、CH3B 的输入捕获只能选择对应的外部 GPIO 管脚输入，可通过滤波寄存器 ATIM_FLTR 的 OCMxyFLTxy 位域进行通道滤波设置，通过 ATIM_CHxCR.BKSy 位域进行相位的设置。

ATIM 的 CH1 是一个具有高级功能的输入通道，输入端 CH1A、CH1B 既可用作触发捕获，又可用作主计数器的触发启动。

CH1A 的输入捕获由主从模式控制寄存器 ATIM_MSCR 的 IA1S 位域确定，可选择 CH1A 输入或 CH1A、CH2A、CH3A 的异或输入。

当设置 ATIM_MSCR.IA1S 为 0 时选择 CH1A 输入，可通过 ATIM_FLTR.OCM1AFLT1A 设置通道滤波，通过 ATIM_ATIM_CH1CR.BKSA 设置相位；设置 ATIM_MSCR.IA1S 为 1 时选择 CH1A、CH2A、CH3A 的异或输入，该特性用于连接霍尔传感器。

CH1B 的输入捕获由主从模式控制寄存器 ATIM_MSCR 的 IB1S 位域确定，可选择 CH1B 输入或内部触发 TS 信号。

当设置 ATIM_MSCR.IB1S 为 0 时选择 CH1B 输入，可通过 ATIM_FLTR.OCM1BFLT1B 设置通道滤波，通过 ATIM_CH1CR.BKSB 设置相位；当设置 ATIM_MSCR.IB1S 为 1 时选择内部触发 TS 信号，通过 ATIM_MSCR.TS 配置 TS 信号的来源，参见下表。

ATIM_MSSR.TS	TS信号
000	ETR经滤波和相位选择后的信号ETFP
001	内部互联信号ITR
101	端口CH1A的边沿信号
110	端口CH1A经滤波和相位选择后的信号IAFP
111	端口CH1B经滤波和相位选择后的信号IBFP

12.3.3输出比较功能

12.3.3.1 输出比较

设置通道 x 控制寄存器 ATIM_CHxCR 的 CSy 位域为 0，使通道 CHxy 工作在输出比较模式。在输出比较模式下，将计数寄存器 ATIM_CNT 的值与对应通道 CHxy 的比较捕获寄存器 ATIM_CHxCCRy 的值相比较，当两者匹配时，比较捕获通道 CHxy 输出为可设定的电平状态。比较通道的输出状态由 ATIM_FLTR 寄存器的 OCMxyFLTxy 位域设置，参见下表。

ATIM_FLTR.OCMxyFLTxy	比较模式配置
000	强制CHxy输出低电平
001	强制CHxy输出高电平
010	在比较匹配时CHxy置0
011	在比较匹配时CHxy置1
100	在比较匹配时CHxy翻转输出
101	在比较匹配时输出一个计数周期的高电平
110	PWM模式1
111	PWM模式2

ATIM_FLTR 寄存器中的输出极性控制位 CCPxy，可设置 CHxy 的输出极性。

当发生比较匹配时，硬件将通道比较匹配中断标志位 ATIM_ISR.CxyF 置位，同时：

- 如果允许中断，即 ATIM_CHxCR.CIEy 为 1，将产生中断请求；
- 如果允许触发 ADC，即 ATIM_TRIG.ADTE 为 1，且 ATIM_TRIG.CMxyE 为 1，比较匹配将触发一次 ADC 转换请求。

比较捕获寄存器 ATIM_CHxCCRy 具有缓存功能，通过 ATIM_CHxCR 寄存器的 BUFEy 位选择是否使用缓存功能。

当 ATIM_CHxCR.BUFEy 为 0 时，禁止通道 CHxy 的比较缓存功能，可在任意时候通过软件更新 ATIM_CHxCCRy 寄存器，更新值立即生效并影响输出波形；

当 ATIM_CHxCR.BUFEy 为 1 时，使能通道 CHxy 的比较缓存功能，更新 ATIM_CHxCCRy 寄存器不会立即更新缓存寄存器，仅当发生更新事件 UEV 时才会将 ATIM_CHxCCRy 寄存器的值更新到缓存寄存器中。

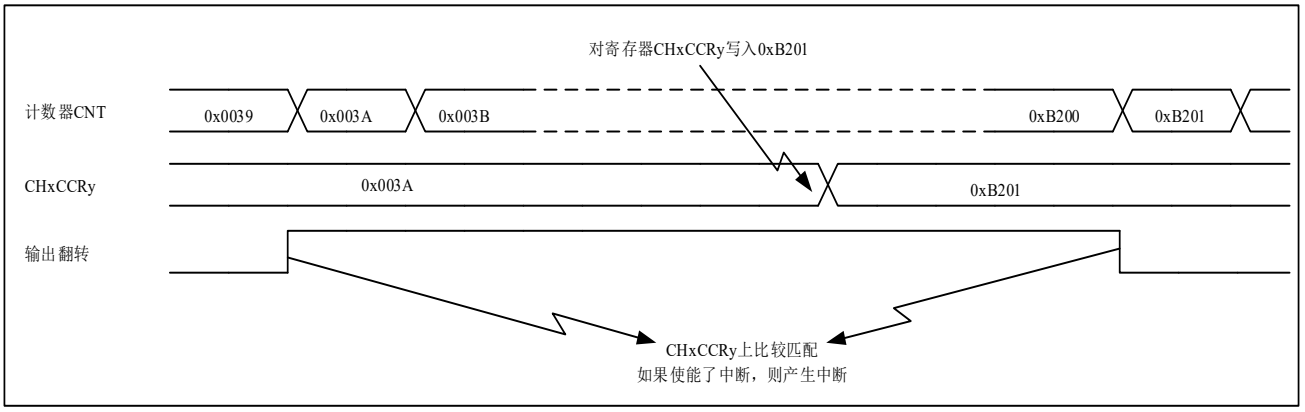

图 12-21 使能缓存，输出比较时序

12.3.3.2 强制输出

强制输出模式，比较通道 CHxy 的输出状态直接由软件强置为 0 或为 1 的状态，而不依赖于比较结果。ATIM_FLTR 寄存器中 OCMxyFLTxy 位域设置为 0x0，强置相应的 CHxy 通道输出低电平；ATIM_FLTR 寄存器中 OCMxyFLTxy 位域设置为 0x1，强置相应的 CHxy 通道输出高电平。

强制输出模式下，ATIM_CHxCCRy 寄存器和计数器 ATIM_CNT 持续比较，比较结果会影响相应标志位，也会产生相应的中断。

12.3.3.3 单脉冲输出

单脉冲模式是 PWM 输出模式的一种，计数器响应触发后，在一段延时之后产生一个脉冲输出，延时时间与脉冲宽度均可程序控制。

单脉冲模式需要设置 ATIM 从模式方式，关于从模式及其触发条件，请参考本章从模式小节。

输出控制/输入滤波寄存器 ATIM_FLTR 的 OCMxyFLTxy 位域，应设置为输出比较模式或 PWM 输出模式，同时将控制寄存器 ATIM_CR 的 ONESHOT 位域设置为 0x1，选择为单次触发模式，计数器将在产生下一个更新事件 UEV 时停止计数。

单脉冲模式启动前，需注意计数器 CNT 当前值与重载值 ARR、比较值 CHxCCRy，应满足以下条件：

- 向上计数方式： CNT < CHxCCRy ≤ ARR
- 向下计数方式： CNT > CHxCCRy

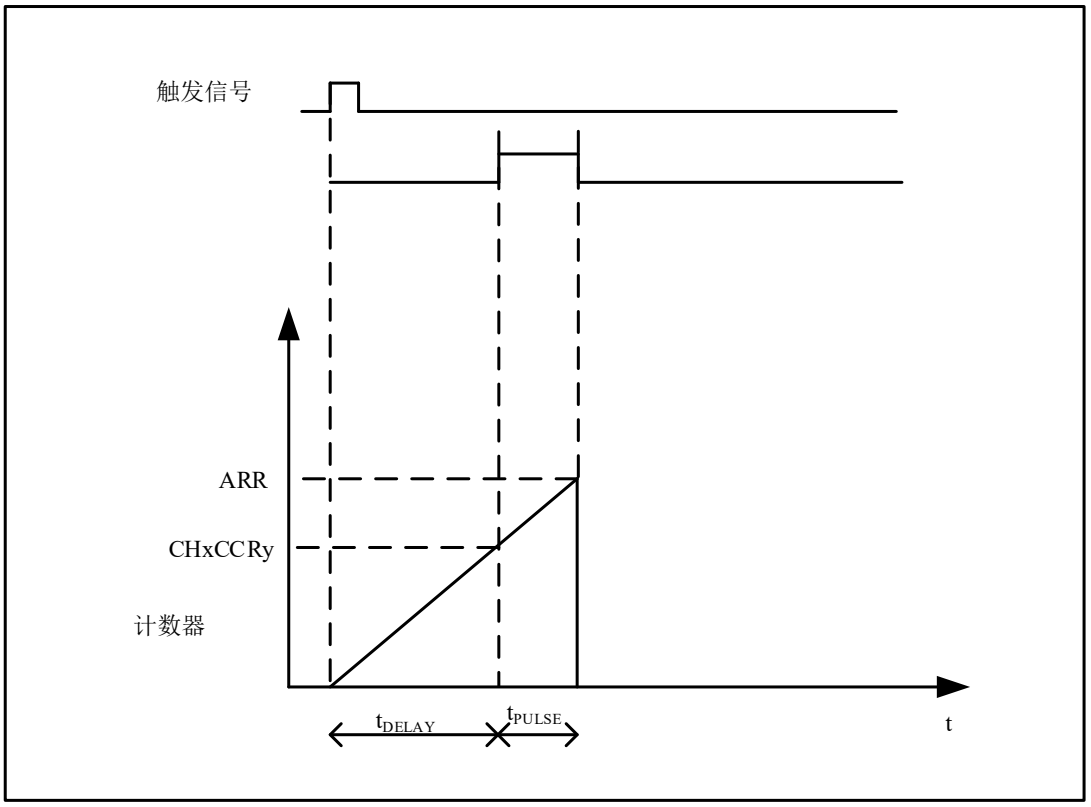

图 12-22 单脉冲输出示例

以从模式触发启动计时器后，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 的初值和比较捕获寄存器 ATIM_CHxCCRy 的差值确定，t_{PULSE} 由重载寄存器 ATIM_ARR 的值来确定。

12.3.3.4 独立 PWM 输出

ATIM 的 PWM（脉冲宽度调制）输出模式，可产生一个频率、占空比可编程的 PWM 波形，频率由重载寄存器 ATIM_ARR 确定，占空比由比较捕获寄存器 ATIM_CHxCCRy 确定。
PWM 输出模式需要设置控制寄存器 ATIM_CR、滤波寄存器 ATIM_FLTR 和死区寄存器。

PWM模式	寄存器	位域	位域值	说明
PWM 模式 1	ATIM_CR	COMP	0x0	独立PWM输出模式
	ATIM_FLTR	OCMxyFLTxy	0x6	PWM 模式 1
	ATIM_DTR	MOE	0x1	使能PWM输出
PWM 模式 2	ATIM_CR	COMP	0x0	独立PWM输出模式
	ATIM_FLTR	OCMxyFLTxy	0x7	PWM 模式 2
	ATIM_DTR	MOE	0x1	使能PWM输出

在 PWM 模式下，比较通道 CHx 的 A 路可通过控制寄存器 ATIM_CR 的 PWM2S 位域配置为单点比较或双点比较工作方式。在单点比较方式下，使用比较捕获寄存器 ATIM_CHxCCRA 控制比较输出；在双点比较方式下，使用比较捕获寄存器 ATIM_CHxCCRA 和 ATIM_CHxCCRB 控制比较输出。比较通道的 B 路只能使用单点比较，由比较捕获寄存器 ATIM_CHxCCRB 控制比较输出。

比较方式	工作模式	计数方向	PWM 模式 1	PWM 模式 2
单点	边沿对齐模式 中央对齐模式	向上	CNT≤CHxCCRy，输出高	CNT<CHxCCRy，输出低
		向下	CNT>CHxCCRy，输出低	CNT≥CHxCCRy，输出高
双点	边沿对齐模式	向上	CHxCCRA<CNT≤CHxCCRB 输出低	CHxCCRA≤CNT<CHxCCRB 输出高
		向下	CHxCCRA<CNT≤CHxCCRB 输出高	CHxCCRA≤CNT<CHxCCRB 输出低
	中央对齐模式	向上	CNT≤CHxCCRA，输出高	CNT<CHxCCRA，输出低
		向下	CNT>CHxCCRB，输出低	CNT≥CHxCCRB，输出高

以下是 ATIM 在不同模式下的独立 PWM 输出示例。

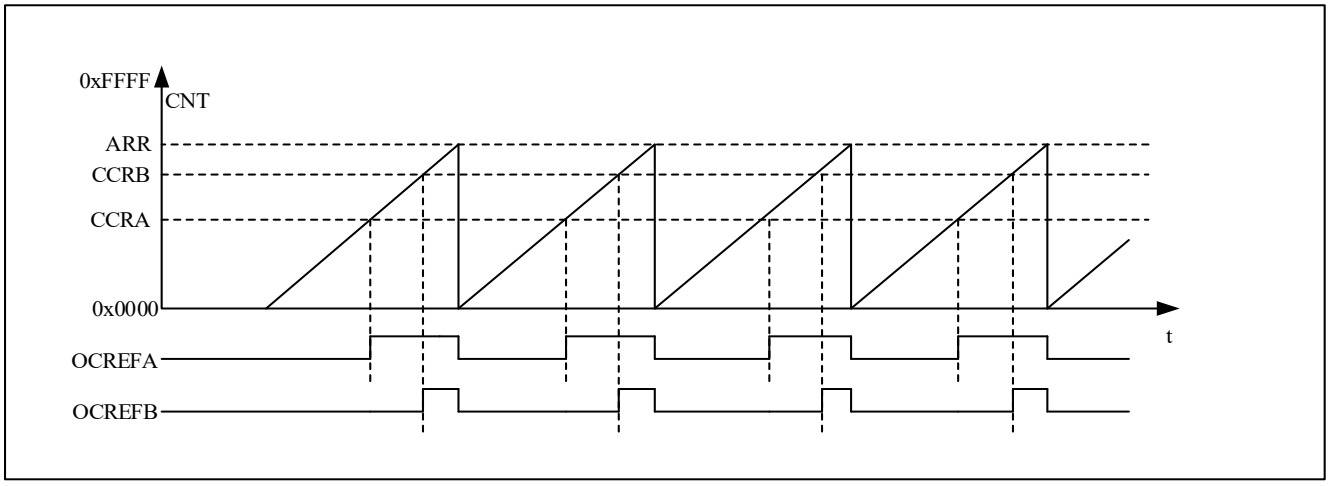

图 12-23 PWM 模式 2，边沿对齐、上计数、单点比较

操作步骤如下：

Step1. 设置 SYSCTRL_APBEN2.ATIM 位为 1，打开 ATIM 模块。

Step2. 设置寄存器 ATIM_CR.MODE 为 0x2，定时器工作在边沿对齐模式下

Step3. 设置寄存器 ATIM_CR.COMP 为 0x0，设置为独立 PWM 输出

Step4. 设置寄存器 ATIM_CR.DIR 为 0x0，边沿对齐计数方向为上计数

Step5. 根据 PWM 的周期设置 ATIM_ARR

Step6. 设置计数器 ATIM_CNT 的初值（初值必须小于 ATIM_ARR 的值）

Step7. 根据 PWM 的脉宽设置比较寄存器 ATIM_CHxCCRA，ATIM_CHxCCRB

Step8. 设置 ATIM_FLTR.OCMxB 和 ATIM_FLTR.OCMxA 为 0x7，配置为 PWM 模式 2

Step9. 设置 ATIM_ICR 寄存器相关位，清除相关中断标志

Step10. 如果需要设置相关的中断使能。

Step11. 设置 ATIM_FLTR.CCPxA 和 ATIM_FLTR.CCPxB 为 0，选择无反相输出。

Step12. 设置 ATIM_DTR.MOE 为 0x01，使能 PWM 的输出

Step13. 设置寄存器 ATIM_CR.EN 为 0x01，启动定时器

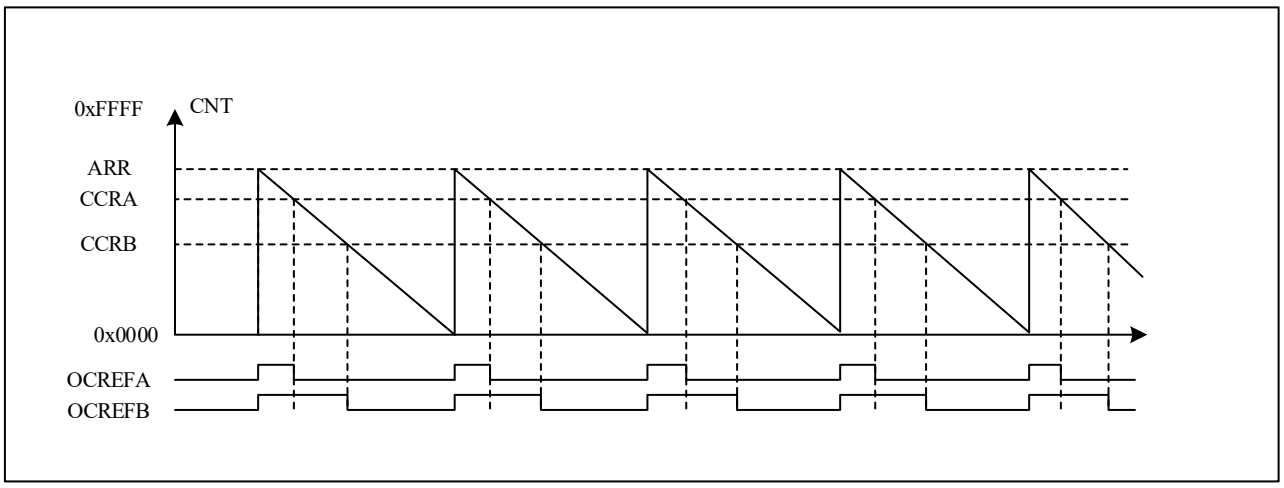

图 12-24 PWM 模式 2，边沿对齐、下计数、单点比较

操作步骤如下：

Step1. 设置 SYSCTRL_APBEN2.ATIM 位为 1，打开 ATIM 模块。

Step2. 设置寄存器 ATIM_CR.MODE 为 0x2，定时器工作在边沿对齐模式下

Step3. 设置寄存器 ATIM_CR.COMP 为 0x0，设置为独立 PWM 输出

Step4. 设置寄存器 ATIM_CR.DIR 为 0x1，边沿对齐计数方向为下计数

Step5. 根据 PWM 的周期设置 ATIM_ARR

Step6. 设置计数器 ATIM_CNT 的初值（初值必须小于 ATIM_ARR 的值）

Step7. 根据 PWM 的脉宽设置比较寄存器 ATIM_CHxCCRA，ATIM_CHxCCRB

Step8. 设置 ATIM_FLTR.OCMxB 和 ATIM_FLTR.OCMxA 为 0x7，配置为 PWM 模式 2

Step9. 设置 ATIM_ICR 寄存器相关位，清除相关中断标志

Step10. 如果需要设置相关的中断使能。

Step11. 设置 ATIM_FLTR.CCPxA 和 ATIM_FLTR.CCPxB 为 0，不需要反相输出。

Step12. 设置 ATIM_DTR.MOE 为 0x01，使能 PWM 的输出

Step13. 设置寄存器 ATIM_CR.EN 为 0x01，使能定时器

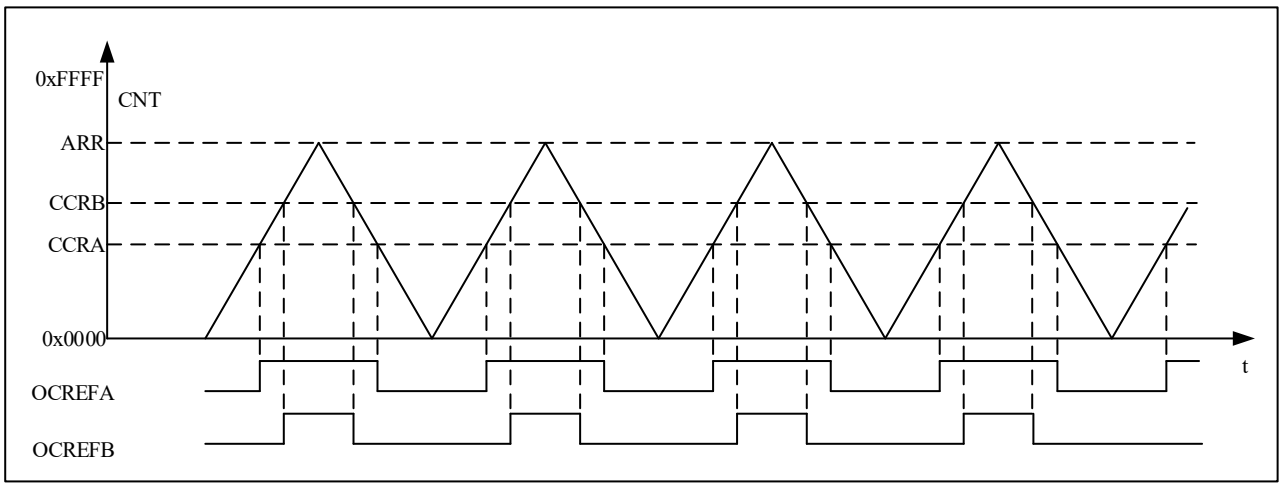

图 12-25 PWM 模式 2，中央对齐、单点比较

操作步骤如下：

Step1. 设置 SYSCTRL_APBEN2.ATIM 位为 1，打开 ATIM 模块。

Step2. 设置寄存器 ATIM_CR.MODE 为 0x3，定时器工作在中央对齐模式下

Step3. 设置寄存器 ATIM_CR.COMP 为 0x0，设置为独立 PWM 输出

Step4. 根据 PWM 的周期设置 ATIM_ARR

Step5. 设置计数器 ATIM_CNT 的初值（初值必须小于 ATIM_ARR 的值）

Step6. 根据 PWM 的脉宽设置比较寄存器 ATIM_CHxCCRA，ATIM_CHxCCRB

Step7. 设置 ATIM_FLTR.OCMxB 和 ATIM_FLTR.OCMxA 为 0x7，配置为 PWM 边沿对齐模式

Step8. 设置 ATIM_ICR 寄存器相关位，清除相关中断标志

Step9. 如果需要设置相关的中断使能。

Step10. 设置 ATIM_FLTR.CCPxA 和 ATIM_FLTR.CCPxB 为 0，不需要反相输出。

Step11. 设置 ATIM_DTR.MOE 为 0x01，使能 PWM 的输出

Step12. 设置寄存器 ATIM_CR.EN 为 0x01，使能定时器

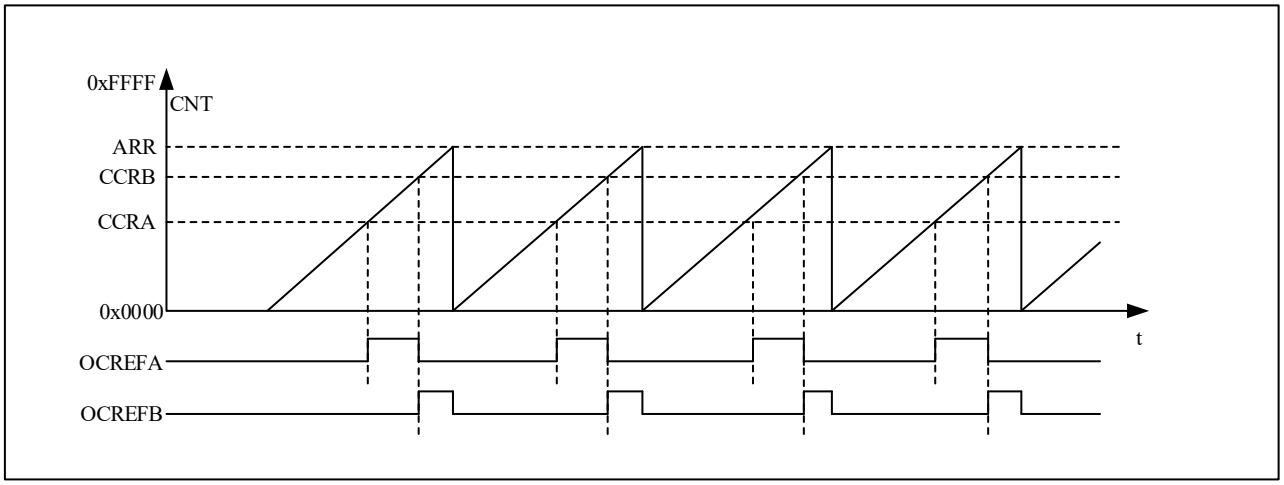

图 12-26 PWM 模式 2，中央对齐、双点比较

操作步骤如下：

Step1. 设置 SYSCTRL_APBEN2.ATIM 位为 1，打开 ATIM 模块。

Step2. 设置寄存器 ATIM_CR.MODE 为 0x2，定时器工作在边沿对齐模式下

Step3. 设置寄存器 ATIM_CR.COMP 为 0x0，设置为独立 PWM 输出

Step4. 设置寄存器 ATIM_CR.DIR 为 0x0，边沿对齐计数方向为上计数

Step5. 根据 PWM 的周期设置 ATIM_ARR

Step6. 设置计数器 ATIM_CNT 的初值（初值必须小于 ATIM_ARR 的值）

Step7. 根据 PWM 的脉宽设置比较寄存器 ATIM_CHxCCRA，ATIM_CHxCCRB

Step8. 设置 ATIM_FLTR.OCMxB 和 ATIM_FLTR.OCMxA 为 0x7，配置为 PWM 模式 2

Step9. 设置 ATIM_ICR 寄存器相关位，清除相关中断标志

Step10. 如果需要设置相关的中断使能。

Step11. 设置 ATIM_FLTR.CCPxA 和 ATIM_FLTR.CCPxB 为 0，不需要反相输出。

Step12. 设置 ATIM_DTR.MOE 为 0x01，使能 PWM 的输出

Step13. 设置寄存器 ATIM_CR.EN 为 0x01，使能定时器

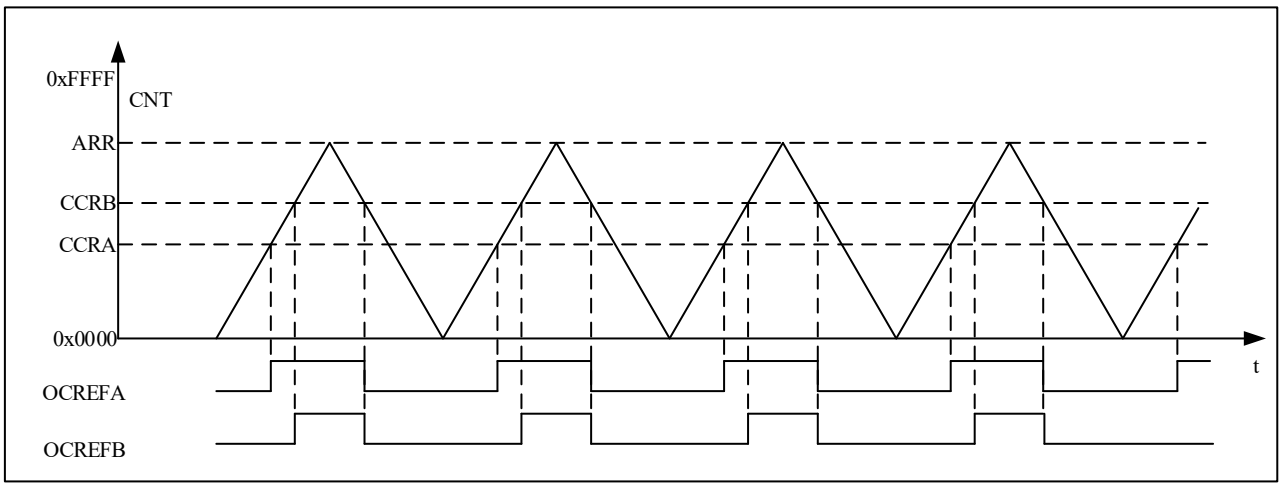

图 12-27 PWM 模式 2，中央对齐、双点比较

注：OCREFxA 按双点比较输出，OCREFxB 由 CHxCCRB 控制单点比较输出

操作步骤如下：

Step1. 设置 SYSCTRL_APBEN2.ATIM 位为 1，打开 ATIM 模块。

Step2. 设置寄存器 ATIM_CR.MODE 为 0x3，定时器工作在中央对齐模式下

Step3. 设置寄存器 ATIM_CR.COMP 为 0x0，设置为独立 PWM 输出

Step4. 根据 PWM 的周期设置 ATIM_ARR

Step5. 设置计数器 ATIM_CNT 的初值（初值必须小于 ATIM_ARR 的值）

Step6. 设置 ATIM_CR.PWM2S 为 0x0，使能双点比较功能

Step7. 根据 PWM 的脉宽设置比较寄存器 ATIM_CHxCCRA，ATIM_CHxCCRB

Step8. 设置 ATIM_FLTR.OCMxB 和 ATIM_FLTR.OCMxA 为 0x7，配置为 PWM 模式 2

Step9. 设置 ATIM_ICR 寄存器相关位，清除相关中断标志

Step10. 如果需要设置相关的中断使能。

Step11. 设置 ATIM_FLTR.CCPxA 和 ATIM_FLTR.CCPxB 为 0，不需要反相输出。

Step12. 设置 ATIM_DTR.MOE 为 0x01，使能 PWM 的输出

Step13. 设置寄存器 ATIM_CR.EN 为 0x01，使能定时器

12.3.3.5 互补 PWM 输出和死区插入

ATIM 可以输出三路互补 PWM 信号，可设置死区时间。

设置控制寄存器 ATIM_CR 的 COMP 位域为 1 选择互补 PWM 输出模式，比较输出通道 CHxA 与通道 CHxB 产生一对互补 PWM。设置 ATIM_FLTR 寄存器的 OCMxAFLTxA 位域为 0x6 选择 PWM 模式 1，为 0x7 选择 PWM 模式 2。设置 ATIM_DTR 寄存器的 MOE 位域为 1，使能 PWM 输出。

互补 PWM 输出模式，可通过控制寄存器 ATIM_CR 的 PWM2S 位域选择单点比较或双点比较工作方式：单点比较时使用比较捕获寄存器 ATIM_CHxCCRA 控制比较输出；双点比较时使用比较捕获寄存器 ATIM_CHxCCRA 和 ATIM_CHxCCRB 控制比较输出。

互补 PWM 输出模式，通道 CHx 的 A 路控制输出信号，B 路比较捕获寄存器 CHxCCRB 不再控制 CHxB 输出，但仍可用作内部控制，比如触发 ADC。

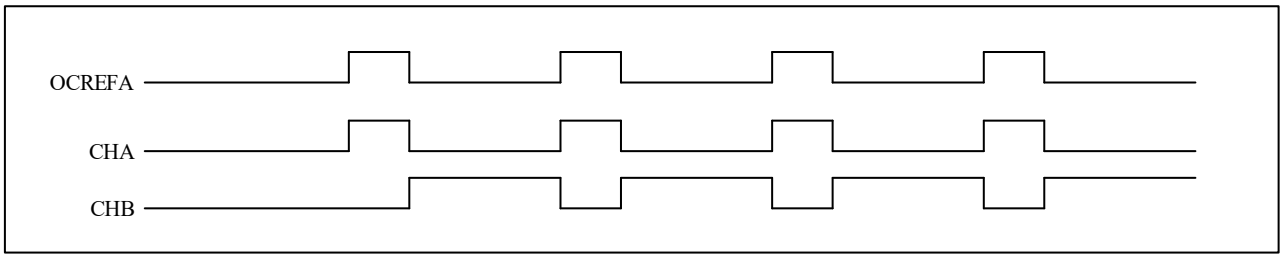

图 12-28 互补 PWM 输出波形

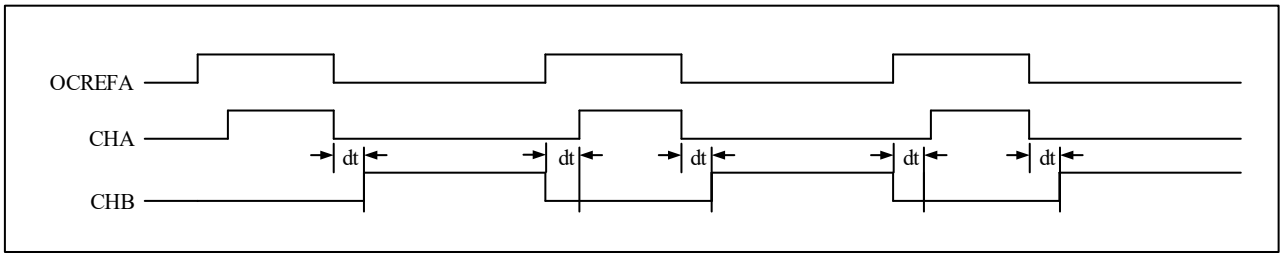

图 12-29 插入死区的互补 PWM 输出波形

设置死区时间寄存器 ATIM_DTR 的 DTEN 位域为 1，使能死区控制。

三个通道 CHx 的死区时间是统一设置的，由 ATIM_DTR 寄存器的 DTR 位域配置。

ATIM_DTR.DTR	步长	死区时间（单位：T_{TCLK}）	当T_{TCLK} = 125 ns时， 死区时间范围
DTR[7] = 0	1 T_{TCLK}	DTR[6:0] + 2	0.25 ~ 16.125 us
DTR[7:6] = 10	2 T_{TCLK}	((DTR[5:0] + 64) * 2 + 2) * 2	32.5 ~ 64 us
DTR[7:5] = 110	8 T_{TCLK}	((DTR[4:0] + 32) * 8 + 2) * 8	258 ~ 506 us
DTR[7:5] = 111	16T_{TCLK}	((DTR[4:0] + 32) * 16 + 2) * 16	1028 ~ 2020 us

12.3.3.6 刹车功能

设置死区时间寄存器 ATIM_DTR 的 BKE 位域为 1，使能刹车功能。

刹车信号为高电平有效，当刹车信号有效时，比较输出通道 CHxy 将立即设置为程序设定的输出状态，具体输出状态由通道控制寄存器 ATIM_CHxCR 的 BSKA 和 BSKB 位域设置，常见的刹车控制是 BSKA、BAKB 同时为高或低电平。

- 刹车触发源
  - 软件刹车，设置 ATIM_CR.BG 位域为 1 触发软件刹车，BG 位自动清零；
  - 电压比较器 VC 的比较输出，ATIM_DTR 寄存器的 VCE 位域为 1 使能；
  - 系统的异常，如：时钟异常，供电异常等，ATIM_DTR.SAFEEN 位域为 1 使能；
  - 外部 ATIM_BK 管脚输入（GPIO 功能复用设置），通过 ATIM_FLTR.FLTBK 位域对刹车输入信号进行滤波设置，ATIM_FLTR.BKP 位域选择刹车输入信号的相位。

当刹车信号有效时，刹车中断标志位 ATIM_ISR.BIF 会被硬件置位，如果允许刹车中断（ATIM_CR.BIE=1），将产生中断请求，设置 ATIM_ICR.BIF 为 0 清除该标志位。

12.3.3.7 比较匹配中断

ATIM 定时器启动后，计数寄存器 ATIM_CNT 和比较捕获寄存器 ATIM_CHxCCRy 开始比较，当二者值相等时，会产生比较匹配事件。

工作于边沿对齐模式时，无论计数方向是向上还是向下，当比较匹配时，对应通道的捕获/比较匹配标志位 ATIM_ISR.CxyF 会被硬件置位，如果允许中断（ATIM_CHxCR.CIEy=1），将产生中断请求，设置 ATIM_ICR.CxyF 为 0 清除该标志位。

工作于中央对齐模式时，比较通道的 A 路、B 路比较匹配方式不同：

- 比较通道的 A 路，由寄存器 ATIM_CR 的 CISA 位域控制，可选择比较匹配发生的时机在向上计数、向下计数或者双向计数过程中。当比较匹配时，对应通道的捕获/比较匹配标志位 ATIM_ISR.CxAF 会被硬件置位，如果允许中断（ATIM_CHxCR.CIEA=1），将产生中断请求，设置 ATIM_ICR.CxAF 为 0 清除该标志位。
- 比较通道的 B 路，由通道 x 控制寄存器 ATIM_CHxCR 的 CISB 位域控制，可选择比较匹配发生的时机，详见下表。（比较通道的 B 路的比较匹配单独控制，可更灵活触发 ADC，详见 ATIM 触发 ADC 小节。）

工作模式	A、B 路	计数方向	CISA、CISB 位域	比较匹配时是否产生中断
边沿对齐模式	A、B 路	向下、向上	--	是
中央对齐模式	A 路	向上	CISA=01b 或 11b	是
			CISA=10b 或 00b	否
		向下	CISA=10b 或 11b	是
			CISA=01b 或 00b	否
	B 路	向上	CISB=01b 或 11b	是
			CISB=10b 或 00b	否
		向下	CISB=10b 或 11b	是
			CISB=01b 或 00b	否

下图是比较匹配产生中断的示例。

图：边沿对齐模式比较匹配中断

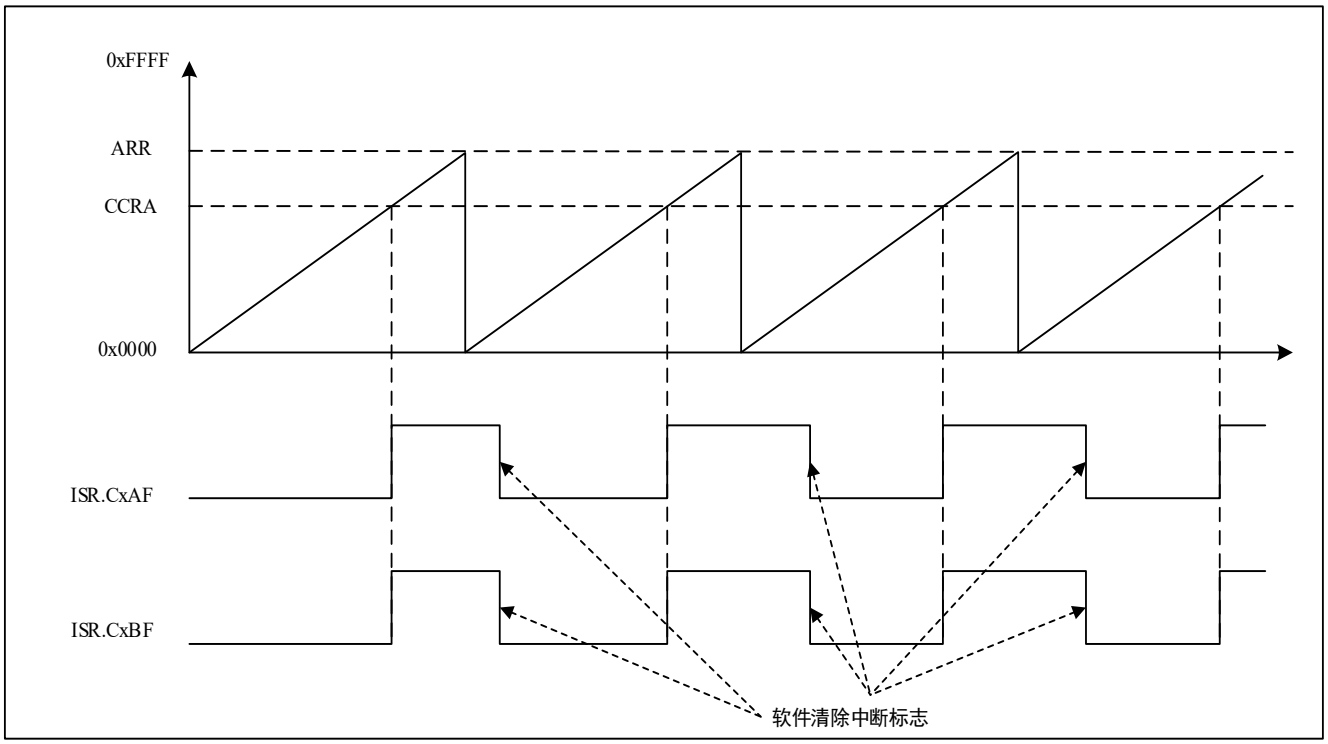

图 12-30 边沿对齐模式比较匹配中断

图：中央对齐模式，A 路双向比较匹配，B 路下计数比较匹配

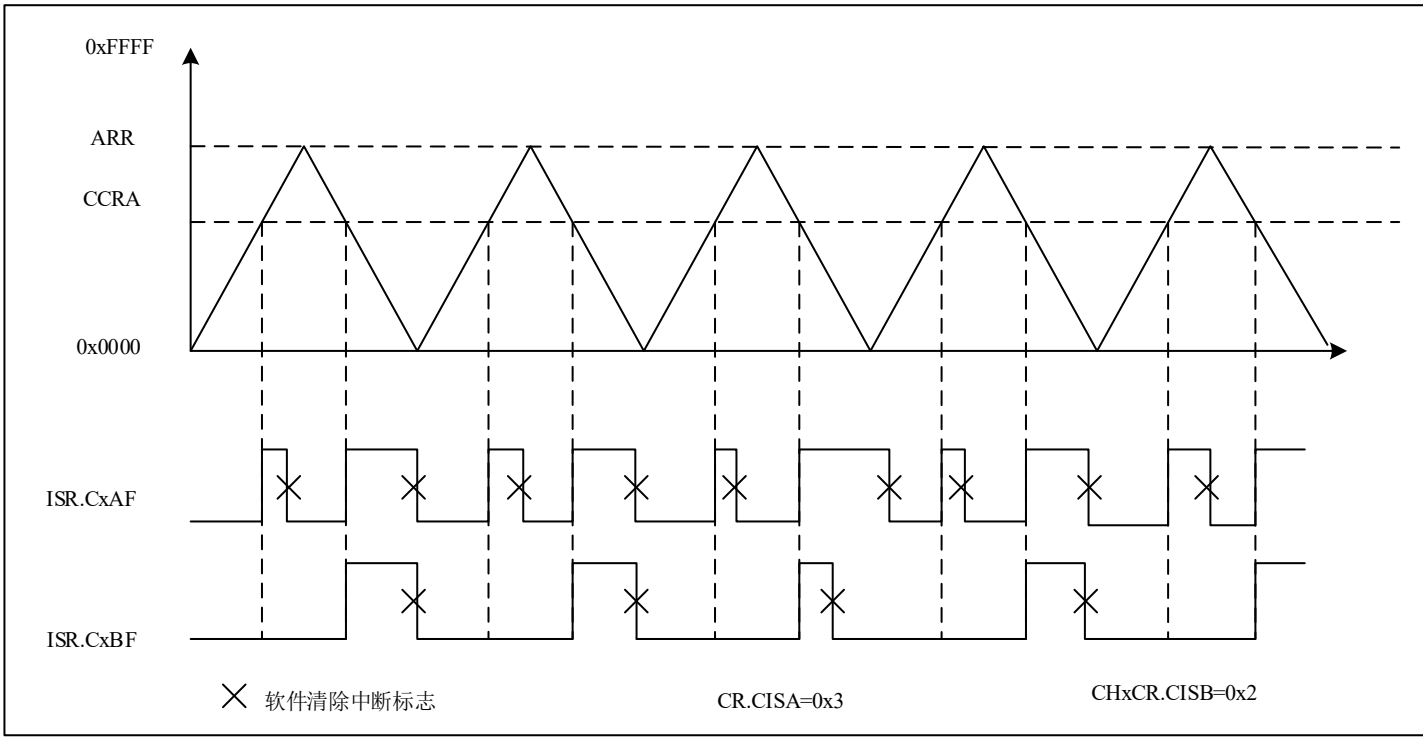

图 12-31 中央对齐模式，A 路双向比较匹配，B 路下计数比较匹配

12.3.4 正交编码计数

ATIM 工作于从模式时具有正交编码计数功能，通过 CH1A、CH1B 连接外部的正交编码器，根据输入信号的跳变顺序，实现计数器自动向上或向下计数。

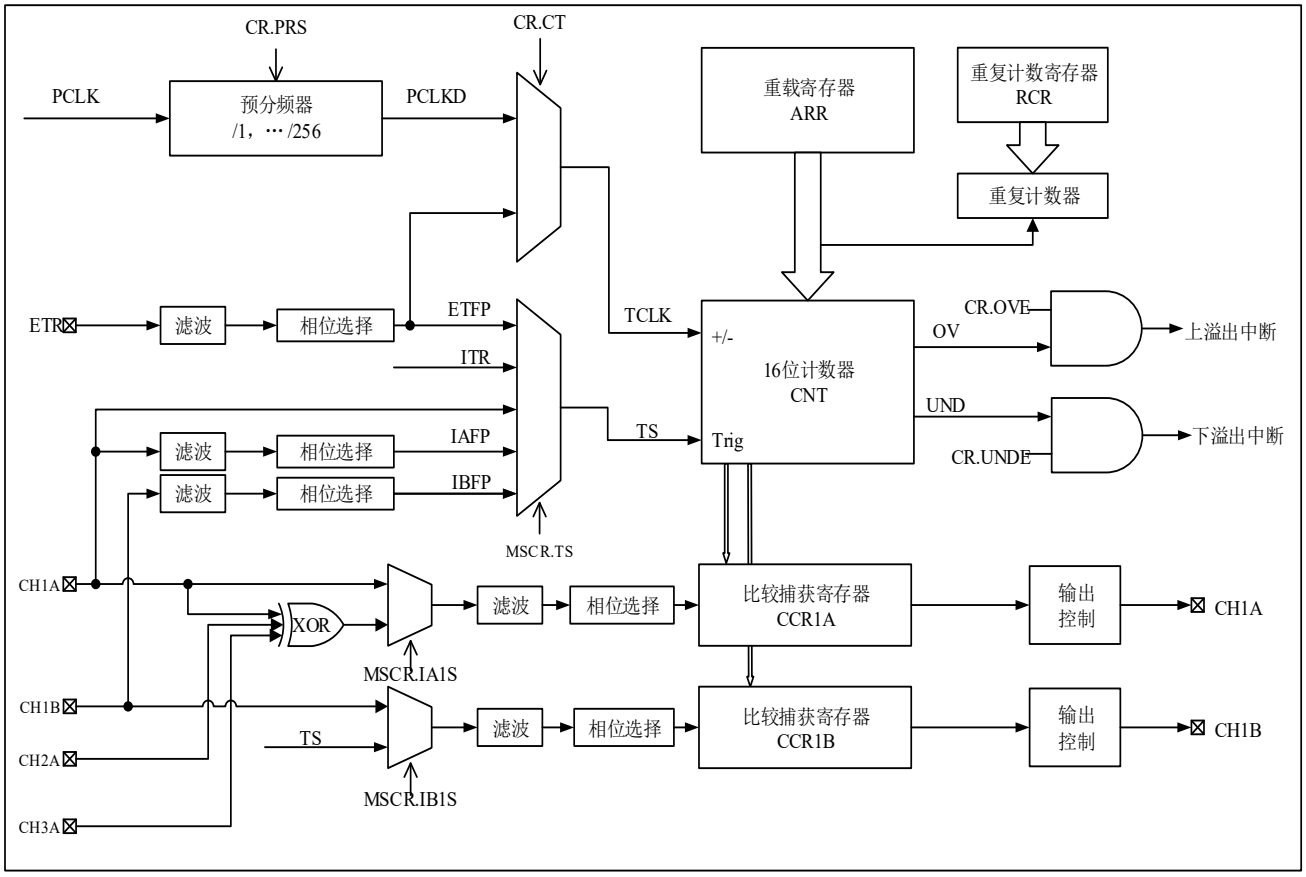

图 12-32 正交编码计数功能框图

CH1A、CH1B 输入信号具有滤波功能，分别由 ATIM_FLTR 寄存器的 OCM1AFLT1A 和 OCM1BFLT1B 位域设置；可配置相位，分别由 ATIM_FLTR 寄存器的 CCP1A 和 CCP1B 位域控制。

IAFP 和 IBFP 是 CH1A、CH1B 经过滤波和相位选择后的内部信号名称，用作编码器的输入信号接口，IAFP 和 IBFP 二者的相位关系决定计数方向，同时会影响控制寄存器 ATIM_CR 的方向位域 DIR。

主从模式控制寄存器 ATIM_MSCR 的 SMS 位域为 0x4、0x5、0x6，分别对应正交编码模式 1、模式 2、模式 3。正交编码的三种模式下计数方向和编码器信号的关系如下表所示：

工作模式	信号的电平		IAFP		IBFP	
	IAFP	IBFP	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，为保证计数方向和速度的正确性， CH1A 和 CH1B 的相位差应大于一个 TCLK 脉冲宽度；CH1A 或 CH1B 的信号脉冲宽度应当大于两个 TCLK 脉冲宽度。

下图是一个编码计数操作的实例，显示了计数信号的产生和方向控制。（同时展示了在中央对齐模式双边沿计数时，如何抑制输入抖动，抖动通常是在编码器转换方向时产生。）

示例配置如下：

- CH1A 经过滤波和不反相输入到 CH1CCRA 中
- CH1B 经过滤波和不反相输入到 CH1CCRB 中
- 设置 TIM_MSCR 寄存器的 SMS 位域为 0x6，选择正交编码模式为 3，CH1A、CH1B 的上升沿和下降沿计数
- 设置 ATIM_CR.EN 为 0x01，启动计数器

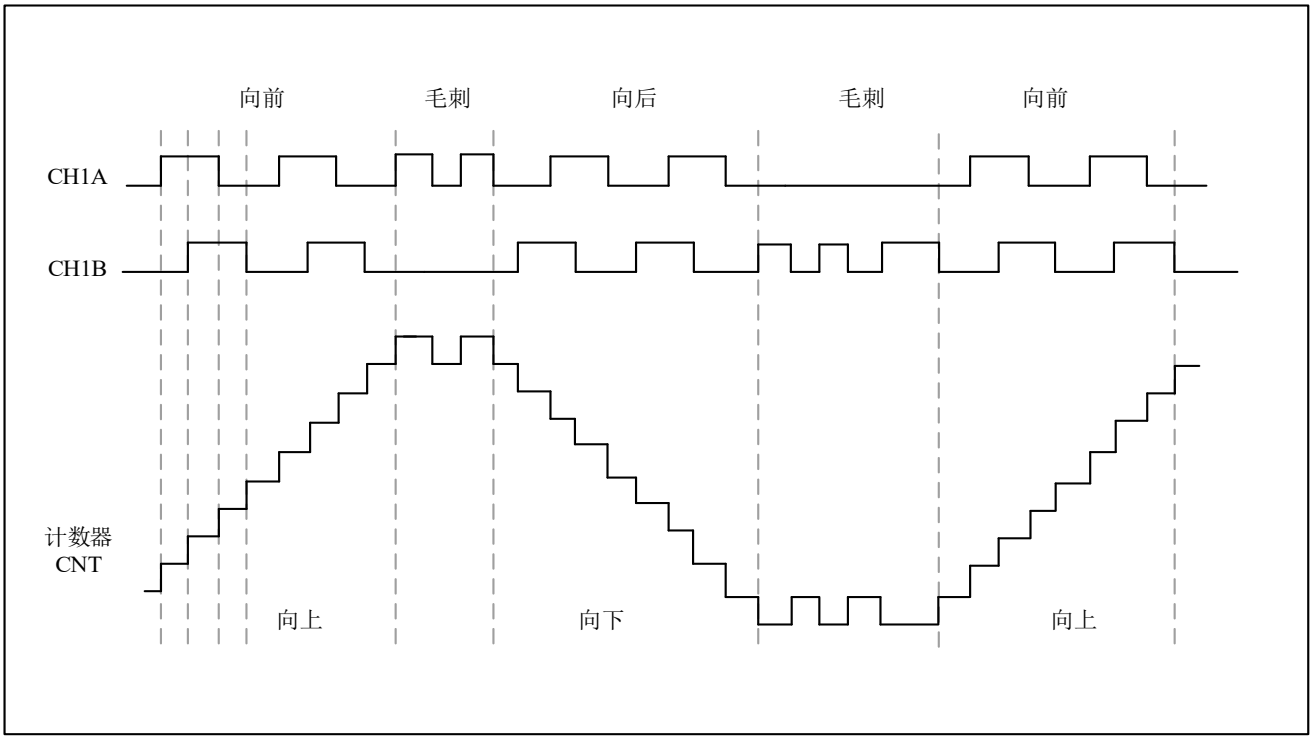

图 12-33 编码计数操作实例

12.3.5触发 ADC

ATIM 边沿对齐模式和中央对齐模式均支持比较匹配事件或更新事件触发 ADC。ADC 触发功能可应用于电机 FOC 控制。

设置触发 ADC 控制寄存器 ATIM_TRIG 的 ADTE 位域为 1，使能 ADC 触发全局控制，允许比较匹配或事件更新时触发启动 ADC。

设置 ATIM_TRIG.UEVE 为 1，更新事件时同步触发启动 ADC。

设置 ATIM_TRIG.CMxyE 为 1，通道 CHxy 比较匹配时触发启动 ADC。关于比较匹配的发生时机，请参考本章比较匹配中断小节。

12.3.6从模式

ATIM 定时器的从模式是指定时器的运行方式受触发信号控制，主从模式控制寄存器 ATIM_MSCR 的 SMS 位域可设置 ATIM 定时器在从模式下的不同工作状态。参见下表：

ATIM_MSCR.SMS	从模式功能选择
000	使用内部时钟
001	复位功能
010	触发模式
011	外部时钟模式
100	正交编码计数模式1
101	正交编码计数模式2
110	正交编码计数模式3
111	门控功能

12.3.6.1 使用内部时钟（主模式）

ATIM 主从模式控制寄存器 ATIM_MSCR 的 SMS 位域为 0x0 时，禁用定时器从模式，计数时钟源由控制寄存器 ATIM_CR 的 CT 位域选择，向 ATIM_CR.EN 位域写入 1，定时器立即开始计数。

12.3.6.2 复位功能（从模式）

ATIM 主从模式控制寄存器 ATIM_MSCR 的 SMS 位域为 0x1 时，ATIM 计数器的复位由 TS 信号控制。TS 信号有效时，初始化 ATIM 计数器 CNT 和预分频器的计数器。如果控制寄存器 ATIM_CR 的 URS 位域为 0，则产生一个更新事件 UEV，置位更新事件中断标志位 ATIM_ISR.UIF，同时更新 ATIM_ARR、ATIM_CHxCCRy 至其缓存寄存器。

TS 信号的来源由主从模式控制寄存器 ATIM_MSCR 的 TS 位控制，参见下表：

ATIM_MSCR.TS位域值	TS信号来源
000	ETR经滤波相位选择后的信号ETFP
001	内部互联信号ITR
101	端口CH1A的输入信号
110	端口CH1A经滤波相位选择后的信号IAFP
111	端口CH1B经滤波相位选择后的信号IBFP

ETR 输入信号来源可以是外部 ATIM_ETR 管脚，也可以是片内其它外设，请参考片内外设互联 ETR 小节。选择 ETR 为 TS 信号源时，可通过 ATIM_FLTR.FLTET 进行滤波控制，通过 ATIM_FLTR.ETP 选择 ETR 输入相位。

内部互联信号 ITR 来源为 BTIM 和 GTIM 的溢出信号，可通过定时器 ITR 来源配置寄存器 SYSCTRL_TIMITR 来选择，参见表 15-15 ATIM 的 ITR 信号来源。

外部输入端口 CH1A 和 CH1B 都具有滤波和相位选择功能，CH1A 和 CH1B 分别通过输出控制/输入滤波寄存器 ATIM_FLTR 的 OCM1AFLT1A 和 OCM1BFLT1B 位域进行滤波控制、CCP1A 和 CCP1B 位域进行相位控制。ATIM 的输入 CH1A、CH1B 管脚需要配置功能复用，可支持的管脚见表 15-14 ATIM 比较捕获通道管脚。

12.3.6.3 触发模式（从模式）

ATIM 主从模式控制寄存器 ATIM_MSCR 的 SMS 位域为 0x2，配置为触发模式，可通过软件或硬件触发启动计数器开始计数。

- 软件触发

设置控制寄存器 ATIM_CR 的 TG 位域为 1，立即启动计数器，TG 位自动清零。

- 硬件触发

触发信号 TS 有效时，启动计数器。TS 信号来源及相位选择参见复位功能小节相关描述。

当产生触发时，触发中断标志位 ATIM_ISR.TIF 会被硬件置位，如果设置触发中断使能位 ATIM_CR.TIE 为 1 将产生中断请求，设置 ATIM_ICR.TIF 为 0 清除该中断标志位。

注意：

如果使用下降沿触发，需要先选择触发极性，然后选择触发模式，以避免产生误触发。

12.3.6.4 外部时钟模式（从模式）

主从模式控制寄存器 ATIM_MSCR 的 SMS 位为 0x3，ATIM 计数器在 TS 信号的每个上升沿或下降沿计数。TS 信号来源及相位选择参见复位功能小节相关描述。

12.3.6.5 正交编码模式（从模式）

主从模式控制寄存器 ATIM_MSCR 的 SMS 位为 0x4~0x6，计数器依据通道 CH1A 和 CH1B 输入信号的跳变顺序进行向上或向下计数，具体描述请参考本章正交编码计数小节。

12.3.6.6 门控功能（从模式）

主从模式控制寄存器 ATIM_MSCR 的 SMS 位域为 0x7，ATIM 配置为门控模式。门控信号 TS 有效且 ATIM_CR.EN 为 1 时，启动计数器计数；门控信号 TS 无效或者 ATIM_CR.EN 为 0 时，暂停计数器计数。

TS 信号来源及相位选择参见复位功能小节相关描述。

12.3.7 内部级联 ITR

本芯片的所有定时器（ATIM、GTIM、BTIM）可以级联使用，前一级定时器的溢出信号作为下一级定时器的 ITR 输入。

ATIM 的 ITR 来源为 BTIM 和 GTIM 的溢出信号，可通过定时器 ITR 来源配置寄存器 SYSCTRL_TIMITR 来选择，参见下表：

SYSCTRL_TIMITR.ATIMITR位域值	ATIM的ITR信号来源
000	BTIM1的溢出信号
001	BTIM2的溢出信号
010	BTIM3的溢出信号
011	GTIM1的溢出信号

ATIM 也可以作为 BTIM 和 GTIM 的 ITR 来源。通过主从模式控制寄存器 ATIM_MSCR 的 MMS 位域，可以选择 ATIM 的主模式输出（ATIM_trgo），连接到其他定时器的 ITR 输入。如下图所示：

图 ATIM 内部级联 ITR

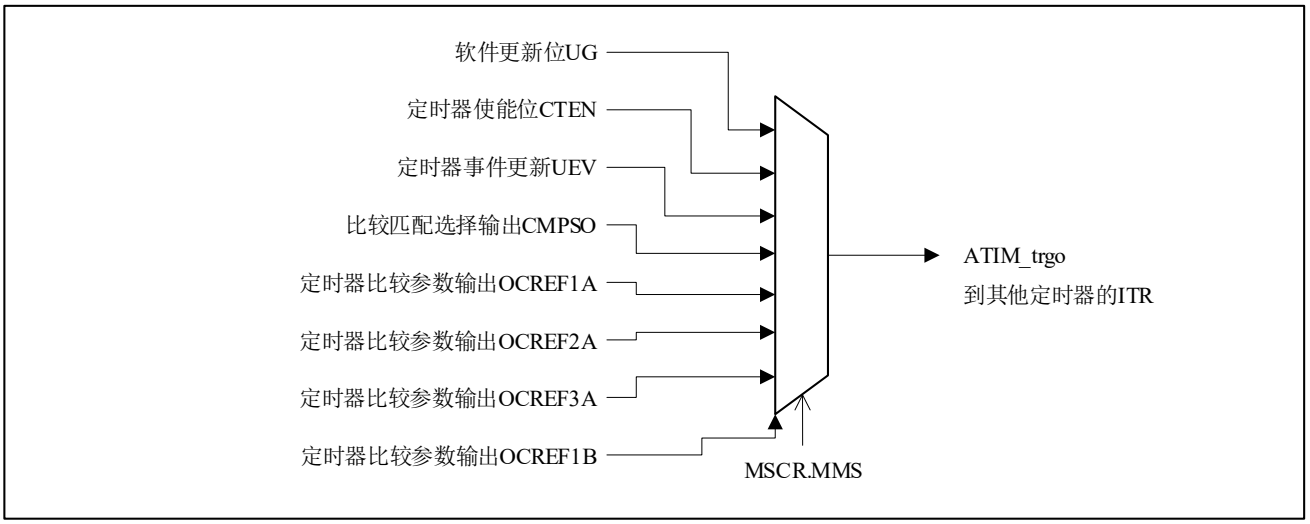

图 12-34 ATIM 内部级联 ITR

ATIM 主模式输出配置如下所示

ATIM_MSCR.MMS位域值	ATIM主模式输出（ATIM_trgo）
000	软件更新UG，写ATIM_CR.UG
001	定时器使能信号EN（ATIM_CR.EN）
010	定时器更新事件UEV
011	比较匹配选择输出CMPSO
100	定时器通道CH1A比较参考输出OCREF1A
101	定时器通道CH2A比较参考输出OCREF2A
110	定时器通道CH3A比较参考输出OCREF3A
111	定时器通道CH1B比较参考输出OCREF1B

12.3.8片内外设互联 ETR

ATIM 的 ETR 信号来源可以是 ATIM_ETR 管脚，也可以是片内其它外设，通过高级定时器 ETR 来源配置寄存器 SYSCTRL_ATIMETR 进行选择。

SYSCTRL_ATIMETR.ATIMETR 为 0x00 时，ETR 信号来自 ATIM_ETR 管脚输入；ATIMETR 为 0x01~0x06 时，ETR 信号来自片内其它外设，实现片内外设互联，参见下表：

SYSCTRL_ATIMETR.ATIMETR	ATIM的ETR来源
000	ATM_ETR管脚
001	UART1的RXD信号
010	UART2的RXD信号
100	VC1的比较输出信号
101	VC2的比较输出信号
110	LVD的输出信号

12.3.9 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_DEBUGEN 的 ATIM 位域来设置。

设置 SYSCTRL_DEBUGEN.ATIM 为 1，则在调试状态时暂停 ATIM 的计数器计数；

设置 SYSCTRL_DEBUGEN.ATIM 为 0，则在调试状态时 ATIM 的计数器继续计数。

12.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
ATIM_ARR	0x4001 2C00 + 0x00	重载寄存器
ATIM_CNT	0x4001 2C00 + 0x04	计数寄存器
ATIM_CR	0x4001 2C00 + 0x0C	控制寄存器
ATIM_ISR	0x4001 2C00 + 0x10	中断标志寄存器
ATIM_ICR	0x4001 2C00 + 0x14	中断标志清除寄存器
ATIM_MSCR	0x4001 2C00 + 0x18	主从模式控制寄存器
ATIM_FLTR	0x4001 2C00 + 0x1C	输出控制/输入滤波寄存器
ATIM_TRIG	0x4001 2C00 + 0x20	触发 ADC 控制寄存器
ATIM_CH1CR	0x4001 2C00 + 0x24	通道 1 控制寄存器
ATIM_CH2CR	0x4001 2C00 + 0x28	通道 2 控制寄存器
ATIM_CH3CR	0x4001 2C00 + 0x2C	通道 3 控制寄存器
ATIM_CH4CR	0x4001 2C00 + 0x58	通道 4 控制寄存器
ATIM_DTR	0x4001 2C00 + 0x30	死区时间寄存器
ATIM_RCR	0x4001 2C00 + 0x34	重复计数寄存器
ATIM_CH1CCRA	0x4001 2C00 + 0x3C	通道 1 比较捕获寄存器 A
ATIM_CH1CCRB	0x4001 2C00 + 0x40	通道 1 比较捕获寄存器 B
ATIM_CH2CCRA	0x4001 2C00 + 0x44	通道 2 比较捕获寄存器 A
ATIM_CH2CCRB	0x4001 2C00 + 0x48	通道 2 比较捕获寄存器 B
ATIM_CH3CCRA	0x4001 2C00 + 0x4C	通道 3 比较捕获寄存器 A
ATIM_CH3CCRB	0x4001 2C00 + 0x50	通道 3 比较捕获寄存器 B
ATIM_CH4CCR	0x4001 2C00 + 0x54	通道 4 比较捕获寄存器

ATIM_CR 控制寄存器

Address offset: 0x0C Reset value: 0x0060 0008

位域	名称	权限	功能描述
31:30	RFU	-	保留位，请保持默认值
29	UNDE	RW	下溢出中断使能 0: 禁止 1: 使能
28	OVE	RW	上溢出中断使能 0: 禁止 1: 使能
27	DIR	RW/RO	计数方向，只有在边沿对齐模式下可以写。其他模式下只读，写无效 0：上计数 1：下计数 注：从其他模式切换到中央对齐模式DIR自动清0。软件事件更新和从模式外部触发复位模式DIR自动清零。
26	BG	RW	软件刹车，自动清零 写0无效 写1产生软件刹车
25	UG	RW	软件更新，自动清零 写0无效 写1产生软件更新 初始化计数器并更新ATIM_ARR、ATIM_CHxCCRy到其缓存寄存器（缓存使能），预分频器的计数器也会被清零。
24	TG	RW	软件触发，自动清零。需要在触发模式(SMS=2)且MODE=2/3下都可以触发。 写0无效 写1产生软件触发
23	OCCE	RW	OCREF清除使能 0：OCREF输出不受OCREF_CLR影响 1：OCREF_CLR高电平信号可以清除OCREF输出
22:21	CISA	RW	中央对齐模式下，A通道的比较匹配中断设置 00：不产生中断 01：向上计数时匹配时产生中断 10：向下计数时匹配时产生中断 11：向上、下计数时匹配时产生中断 注：B通道的比较匹配中断设置在ATIM_CHxCR寄存器的CISB位域
20	BIE	RW	刹车中断使能 0：禁止 1：使能
19	TIE	RW	触发中断使能 0：禁止 1：使能
18	RFU	-	保留位，请保持默认值
17	URS	RW	更新源

			0：上溢出/下溢出/软件更新UG/从模式复位 1：上溢出/下溢出
16	OCCS	RW	OCREF清除源选择 0：电压比较器VC输出，VC选择在VCx_CR1寄存器设置 1：ETR端口滤波相位选择后的信号 当OCCE有效时，OCREF_CLR高电平信号可以清除OCREF的比较输出信号为零（当OCMxyFLTxy>1时有效），下一个更新事件UEV后继续比较输出
15	RFU	-	保留位，请保持默认值
14	ONESHOT	RW	单次触发模式选择 0：循环计数 1：发生事件更新后定时器停止
13:12	ALIGNMENT	RW	对齐方式 00：保留　　01：保留 10：边沿对齐模式　　11：中央对齐模式
11	RFU	-	保留位，请保持默认值
10	UIE	RW	UIE 更新中断使能 0：禁止　　1：使能
9:8	RFU	-	保留位，请保持默认值
7	BUFPEN	RW	重载寄存器缓存使能 1：缓存使能，写入后下个事件更新后才影响到周期值 0：缓存无效，写入后立刻影响周期值
6:4	PRS	RW	内部时钟PCLK分频选择 000：1　001：2　010：4　011：8 100：16　101：32　110：64　111：256
3	PWM2S	RW	OCREFA双点比较选择(缺省值为1) 0：双点比较使能，使用CHxCCRA、CHxCCRB比较控制OCREFA输出 1：单点比较使能，只使用CHxCCRA比较控制OCREFA输出 注：OCREFB不受影响，仍然使用CHxCCRB控制OCREFB输出
2	CT	RW	计数时钟选择 0：内部计数时钟PCLK 1：外部计数时钟ETR
1	COMP	RW	PWM互补输出模式选择 0：独立PWM输出 1：互补PWM输出
0	EN	RW	定时器使能 0：禁止 1：使能 在ONESHOT模式下结束时该位自动清零

ATIM_ARR 重载寄存器

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	ARR	RW	重载寄存器/周期寄存器，具有缓存功能 在计数器未使能或者缓存没有使能时，缓存寄存器可以立刻更新

ATIM_CNT 计数寄存器

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	CNT	RW	重载定时器计数寄存器

注意：使用 PWM 比较输出时，初始化 CNT 值需要小于 ARR 的值。

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
31:19	RFU	-	保留位，请保持默认值
18	C4AF	RO	通道 CH4 比较匹配标志
17	UNDF	RO	下溢出中断标志
16	OVF	RO	上溢出中断标志
15	TIF	RO	触发中断标志
14	BIF	RO	刹车中断标志
13	C3BE	RO	通道 CH3B 捕获数据丢失标志 0：无数据丢失； 1：数据丢失
12	C2BE	RO	通道 CH2B 捕获数据丢失标志 0：无数据丢失； 1：数据丢失
11	C1BE	RO	通道 CH1B 捕获数据丢失标志 0：无数据丢失； 1：数据丢失
10	C3AE	RO	通道 CH3A 捕获数据丢失标志 0：无数据丢失； 1：数据丢失

9	C2AE	RO	通道 CH2A 捕获数据丢失标志 0：无数据丢失； 1：数据丢失
8	C1AE	RO	通道 CH1A 捕获数据丢失标志 0：无数据丢失； 1：数据丢失
7	C3BF	RO	通道 CH3B 发生捕获/比较匹配标志 0：没有发生 1：发生
6	C2BF	RO	通道 CH2B 发生捕获/比较匹配标志 0：没有发生 1：发生
5	C1BF	RO	通道 CH1B 发生捕获/比较匹配标志 0：没有发生 1：发生
4	C3AF	RO	通道 CH3A 发生捕获/比较匹配标志 0：没有发生 1：发生
3	C2AF	RO	通道 CH2A 发生捕获/比较匹配标志 0：没有发生 1：发生
2	C1AF	RO	通道 CH1A 发生捕获/比较匹配标志 0：没有发生 1：发生
1	RFU	-	保留位，请保持默认值
0	UIF	RO	事件更新中断标志 0：没有发生 1：发生

ATIM_ICR 中断标志清除寄存器

Address offset: 0x14 Reset value: 0x0007 FFFF

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18	C4AF	R1W0	通道 CH4 比较匹配标志清除，写 0 清除
17	UNDF	R1W0	下溢出中断标志清除，写 0 清除
16	OVF	R1W0	上溢出中断标志清除，写 0 清除
15	TIF	R1W0	触发中断标志清除，写 0 清除
14	BIF	R1W0	刹车中断标志清除，写 0 清除
13	C3BE	R1W0	通道 CH3B 捕获数据丢失标志清除，写 0 清除
12	C2BE	R1W0	通道 CH2B 捕获数据丢失标志清除，写 0 清除
11	C1BE	R1W0	通道 CH1B 捕获数据丢失标志清除，写 0 清除
10	C3AE	R1W0	通道 CH3A 捕获数据丢失标志清除，写 0 清除
9	C2AE	R1W0	通道 CH2A 捕获数据丢失标志清除，写 0 清除
8	C1AE	R1W0	通道 CH1A 捕获数据丢失标志清除，写 0 清除
7	C3BF	R1W0	通道 CH3B 捕获/比较匹配标志清除，写 0 清除
6	C2BF	R1W0	通道 CH2B 捕获/比较匹配标志清除，写 0 清除
5	C1BF	R1W0	通道 CH1B 捕获/比较匹配标志清除，写 0 清除
4	C3AF	R1W0	通道 CH3A 捕获/比较匹配标志清除，写 0 清除
3	C2AF	R1W0	通道 CH2A 捕获/比较匹配标志清除，写 0 清除
2	C1AF	R1W0	通道 CH1A 捕获/比较匹配标志清除，写 0 清除
1	RFU	-	保留位，请保持默认值
0	UIF	R1W0	事件更新中断清除，写 0 清除

ATIM_MSCR 主从模式控制寄存器

Address offset: 0x18 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	IB1S	RW	CH1B 输入选择 0：CH1B 1：内部触发 TS 信号
11	IA1S	RW	CH1A 输入选择 0： CH1A 1： CH1A、CH2A、CH3A 三者异或 注：设置为 1 后端口的任意一个端口变化将导致输入变化
10:8	SMS	RW	从模式功能选择 000：使用内部时钟 100：正交编码计数模式 1 001：复位功能 101：正交编码计数模式 2 010：触发模式 110：正交编码计数模式 3 011：外部时钟模式 111：门控功能
7:5	TS	RW	触发选择 000：端口 ETR 的滤波相位选择后的信号 ETFP 001：内部互联信号 ITR 101：端口 CH1A 的边沿信号 110：端口 CH1A 的滤波相位选择后的信号 IAFP 111：端口 CH1B 的滤波相位选择后的信号 IBFP
4	MSM	RW	主从选择 0：无延时 1：延时使能，使主从计数器同时启动 注：使用触发模式时，从模式设置为 0，主模式设置为 1，可以使主从计数同时启动
3	RFU	-	保留位，请保持默认值
2:0	MMS	RW	主模式输出选择（ATIM_trgo），连接到其他定时器的 ITR 000：软件更新 UG，写 CR.UG 001：定时器使能信号 EN 010：定时器更新事件 UEV 011：比较匹配选择输出 CMPSO 100：定时器通道 CH1A 比较参数输出 OCREF1A 101：定时器通道 CH2A 比较参数输出 OCREF2A 110：定时器通道 CH3A 比较参数输出 OCREF3A 111：定时器通道 CH1B 比较参数输出 OCREF1B

ATIM_FLTR 输出控制/输入滤波寄存器

Address offset: 0x1C Reset value: 0x0000 0000

位域	名称	权限	功能描述
31	ETP	RW	ETR 输入相位选择 0：同相位 1：反向输入
30:28	FLTET	RW	ETR 滤波设置，3 个连续有效 0xx：滤波无效 100：PCLK 110：PCLK /16 101：PCLK /4 111：PCLK /64
27	BKP	RW	刹车 BK 输入相位选择 0：同相位 1：反向输入
26:24	FLTBK	RW	刹车输入滤波设置，3 个连续有效 0xx：滤波无效 100：PCLK 110：PCLK /16 101：PCLK /4 111：PCLK /64
23	CCP3B	RW	比较功能：CH3B 通道比较输出相位控制 0：正常输出 1：反向输出
22:20	OCM3B FLT3B	RW	比较功能：CH3B 通道比较控制，参考 OCM1BFLT1B 捕获功能：CH3B 输入通道滤波设置，参考 FLTBK
19	CCP3A	RW	比较功能：CH3A 通道比较输出相位控制 0：正常输出 1：反向输出
18:16	OCM3A FLT3A	RW	比较功能：CH3A 通道比较控制，参考 OCM1BFLT1B 捕获功能：CH3A 输入通道滤波设置，参考 FLTBK
15	CCP2B	RW	比较功能：CH2B 通道比较输出相位控制 0：正常输出 1：反向输出
14：12	OCM2B FLT2B	RW	比较功能：CH2B 通道比较控制，参考 OCM1BFLT1B 捕获功能：CH2B 输入通道滤波设置，参考 FLTBK
11	CCP2A	RW	比较功能：CH2A 通道比较输出相位控制 0：正常输出 1：反向输出
10:8	OCM2A FLT2A	RW	比较功能：CH2A 通道比较控制，参考 OCM1BFLT1B 捕获功能：CH2A 输入通道滤波设置，参考 FLTBK
7	CCP1B	RW	比较功能：CH1B 通道比较输出相位控制 0：正常输出 1：反向输出 编码计数与从模式门控功能：输入相位控制 从模式门控，复位，外部触发，外部时钟使用 CH1B 端口输

			入极性控制 0：正常输入 1：反向输入
6:4	OCM1B FLT1B	RW	比较功能：CH1B 通道比较控制 000：强制为 0 001：强制为 1 010：比较匹配时强制为 0 011：比较匹配时强制为 1 100：比较匹配时翻转 101：比较匹配时输出一个计数周期的高电平 110：PWM 模式 1 单点比较： 上计数时 CNT<CHxCCRy 输出高，下计数时 CNT>CHxCCRy 输出为低电平 双点比较： 1）边沿对齐上计数 CHxCCRA<CNT≤CHxCCRB 输出为低电平 2）边沿对齐下计数 CHxCCRA<CNT≤CHxCCRB 输出为高电平 3）中央对齐上计数 CNT<CHxCCRA 输出高，下计数 CNT>CHxCCRB 为低电平 111：PWM 模式 2 单点比较： 上计数时 CNT<CHxCCRy 输出低，下计数时 CNT>CHxCCRy 输出为高电平 双点比较： 1）边沿对齐上计数 CCRxA≤CNT<CCRxB 输出为高电平 2）边沿对齐下计数 CCRxA≤CNT<CCRxB 输出为低电平 3）中央对齐上计数 CNT<CCRxA 输出低，下计数 CNT>CCRxB 为高电平 捕获功能：CH1B 输入通道滤波设置，参考 FLTBK
3	CCP1A	RW	比较功能：CH1A 通道比较输出相位控制 0：正常输出 1：反向输出 编码计数与从模式门控功能：输入相位控制 从模式门控，复位，外部触发，外部时钟使用 CH1A 端口输入极性控制 0：正常输入 1：反向输入
2:0	OCM1A FLT1A	RW	比较功能：A 通道比较控制，参考 OCM1BFLT1B 捕获功能：CH1A 输入通道滤波设置，参考 FLTBK

ATIM_TRIG 触发 ADC 控制寄存器

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	ADTE	RW	使能 ADC 触发全局控制 0：禁止 1：使能
6	CM3BE	RW	通道 CH3B 比较匹配触发 ADC 使能 0：禁止 1：使能
5	CM2BE	RW	通道 CH2B 比较匹配触发 ADC 使能 0：禁止 1：使能
4	CM1BE	RW	通道 CH1B 比较匹配触发 ADC 使能 0：禁止 1：使能
3	CM3AE	RW	通道 CH3A 比较匹配触发 ADC 使能 0：禁止 1：使能
2	CM2AE	RW	通道 CH2A 比较匹配触发 ADC 使能 0：禁止 1：使能
1	CM1AE	RW	通道 CH1A 比较匹配触发 ADC 使能 0：禁止 1：使能
0	UEVE	RW	事件更新触发 ADC 使能 0：禁止 1：使能

ATIM_DTR 死区时间寄存器

Address offset: 0x30 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	VCE	RW	VC 比较输出刹车使能 0：禁止 1：使能
13	SAFEEN	RW	Safety 刹车使能（osc fail,brown down,lockup） 0：禁止 1：使能
12	MOE	RW	PWM 输出使能 0：禁止 1：使能
11	AOE	RW	PWM 输出自动使能 0：禁止 1：使能
10	BKE	RW	刹车使能 0：禁止 1：使能
9	DTEN	RW	死区控制使能 0：禁止 1：使能
8	RFU	-	保留位，请保持默认值
7:0	DTR	RW	死区时间寄存器 DTR[7] =0 T = DTR[6:0]+2 DTR[7:6] =10 T = (DTR[5:0]+64)*2 +2 DTR[7:5] =110 T = (DTR[4:0]+32)*8 +2 DTR[7:5] =111 T = (DTR[4:0]+32)*16 +2

ATIM_RCR 重复计数寄存器

Address offset: 0x34 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	UD	RW	重复计数控制下溢出屏蔽 0：向下计数溢出时重复计数器计数 1：向下计数溢出屏蔽
8	OV	RW	重复计数控制上溢出屏蔽 0：向上计数溢出时重复计数器计数 1：向上计数溢出屏蔽
7:0	RCR	RW	重复周期计数值 设置 RCR+1 个周期个由[9:8]控制的溢出产生事件更新，计数器上溢出或下溢出时内部 RCR_CNT 减 1，当计数到零后 RCR_CNT 重载 RCR 的值，并且产生事件更新 UEV 信号

ATIM_CH1CR 通道 1 控制寄存器

Address offset: 0x24 Reset value: 0x0000 3000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15	CCGB	RW	捕获比较 B 通道软件触发，硬件自动清零。 在比较模式下只产生中断 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中
14	CCGA	RW	捕获比较 A 通道软件触发，硬件自动清零。 在比较模式下只产生中断 在捕获模式下产生中断并且捕获计数器的值到捕获寄存器中
13:12	CISB	RW	中央对齐模式下，B 通道的比较匹配中断设置 00：不产生中断 01：向上计数时匹配时产生中断 10：向下计数时匹配时产生中断 11：向上、下计数时匹配时产生中断 注：A 通道的比较匹配中断设置在 ATIM_CR 寄存器的 CISA 位域
9	CIEB	RW	B 通道捕获比较触发中断使能 0：禁止 1：使能
8	CIEA	RW	A 通道捕获比较触发中断使能 0：禁止 1：使能
7	BUFEB	RW	比较功能：比较捕获寄存器 B 缓存使能控制 0：禁止 1：使能
6	BUFEA	RW	比较功能：比较捕获寄存器 A 缓存使能控制 0：禁止 1：使能
5	CSB	RW	B 通道捕获/比较功能选择 0：比较模式 1：捕获模式
4	CSA	RW	A 通道捕获/比较功能选择 0：比较模式 1：捕获模式
3:2	BKSB	RW	比较模式 B 通道比较功能输出刹车电平控制 00：高阻输出 01：对输出无影响 10：强制输出低电平 11：强制输出高电平 捕捉模式 B 通道捕获使能

			00：禁止 01：使能上升沿捕捉 10：使能下降沿捕捉 11：使能上升沿/下降沿捕捉
1:0	BKSA	RW	比较模式 A 通道比较功能输出刹车电平控制 00：高阻输出 01：对输出无影响 10：强制输出低电平 11：强制输出高电平 捕捉模式 A 通道捕获使能 00：禁止 01：使能上升沿捕捉 10：使能下降沿捕捉 11：使能上升沿/下降沿捕捉

ATIM_CH2CR 通道 2 控制寄存器

Address offset: 0x28 Reset value: 0x0000 3000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0			详见 ATIM_CH1CR

ATIM_CH3CR 通道 3 控制寄存器

Address offset：0x2C Reset value： 0x0000 3000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:0			详见 ATIM_CH1CR

ATIM_CH4CR 通道 4 控制寄存器

Address offset: 0x58 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	C4EN	RW	通道比较使能
4:3	CIS	RW	中央对齐模式下，通道 CH4 的比较匹配中断设置 00：不产生中断 01：向上计数时匹配时产生中断 10：向下计数时匹配时产生中断 11：向上、下计数时匹配时产生中断
2	CDE	RW	RFU
1	CIE	RW	比较触发中断使能 0：禁止 1：使能
0	BUFE	RW	比较功能：通道 4 比较捕获寄存器缓存使能控制 0：禁止 1：使能

ATIM_CH1CCRA 通道 1 比较捕获寄存器 A

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	CCR1A	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH1CCRB 通道 1 比较捕获寄存器 B

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	CCR1B	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH2CCRA 通道 2 比较捕获寄存器 A

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	CCR2A	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH2CCRB 通道 2 比较捕获寄存器 B

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	CCR2B	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH3CCRA 通道 3 比较捕获寄存器 A

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	CCR3A	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH3CCRB 通道 3 比较捕获寄存器 B

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	CCR3B	RW	比较捕获寄存器，比较具有缓存功能

ATIM_CH4CCR 通道 4 比较捕获寄存器

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	CCR4	RW	比较捕获寄存器，比较具有缓存功能

13 独立看门狗定时器（IWDT）

13.1 概述

本芯片内部集成高安全性的独立看门狗定时器(IWDT)，并配备专用的时钟源 RC8K。该独立看门狗可检测并解决由软件错误导致的故障，并在计数器达到给定的超时值时触发系统复位。IWDT 在深度休眠模式下可选择继续运行或暂停运行。

13.2 主要特性

- 12bit 递减计数器
- 专用时钟源 RC8K
- 溢出周期为 500us ~ 262s
- 溢出可触发中断或复位
- 深度休眠模式下可继续运行或暂停运行

13.3 功能描述

本节将详细描述独立看门狗定时器的相关功能。

13.3.1 功能框图

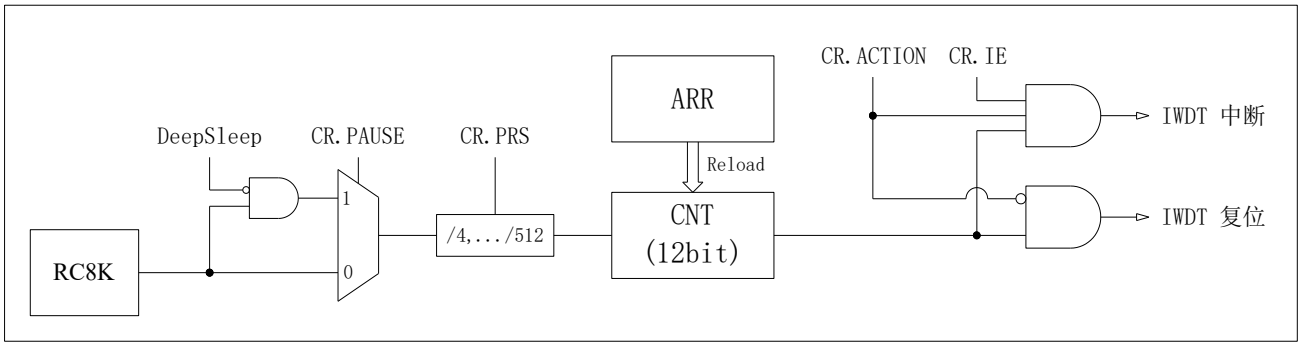

图 13-1 IWDT 功能框图

IWDT 由独立专用低频 RC 振荡器、预分频器、可重装载计数器组成。独立专用低频 RC 振荡器 RC8K 的输出时钟频率约 8.6kHz，该时钟信号经 4～512 分频后驱动计数器进行递减计数。
当计数器计数到 0 并再次接收到计数时钟时，计数器会产生溢出信号并置位 SR.OV 标志。该溢出信号会触发计数器重载，将 ARR 寄存器值加载到计数器。当 CR.ACTION 为 0 时，该溢出信号将产生复位信号并复位整个芯片；当 CR.ACTION 与 CR.IE 均为 1 时，该溢出信号将产生中断并执行的 IWDT 中断服务程序，用户在退出中断服务程序前应向 SR.OV 控制位写入 0 以清除 IWDT 溢出标志。

13.3.2 运行控制

向 KR 寄存器写入 0xCCCC，即可启动 IWDT；计数器从 0xFFF 开始向下计数。
向 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。
向 KR 寄存器依次写入 0x5A5A - 0xA5A5 即可暂停 IWDT，暂停后向 IWDT_KR 寄存器写入 0xCCCC 即可恢复运行。
当设置 CR.PAUSE 为 1 时，芯片进入 DeepSleep 模式时 IWDT 自动暂停，退出 DeepSleep 时自动恢复运行。

13.3.3 溢出周期

IWDT 运行时，其计数值范围为 ARR～0；其溢出周期如下所示：

$$T_{ov} = \frac{(ARR + 1) \times 4 \times 2^{CR.PRS}}{f_{RC8K}}$$

其中 f_{RC8K} 代表内置独立振荡器输出时钟的频率，其频率约为 8.6kHz；ARR 代表 ARR 寄存器的

值；CR.PRS 为预分频器的分频系数。
通过配置 ARR 及 CR.PRS，可获得的溢出周期的典型范围为 500us ~ 262s。

13.3.4 寄存器访问

CR、ARR 和 WINR 寄存器具有写保护；在写保护状态下其值不可修改，解除写保护状态后，可以修改这些寄存器的值。应向 KR 寄存器写入 0x5555 可以解除这些寄存器的写保护；向 IWDT_KR 寄存器写入其它值则会使能这些的寄存器写保护。
修改 CR、ARR 和 WINR 寄存器需一定的时间才能完成；对这些寄存器进行修改后需检查 WINRF、ARRF、CRF 标志以等待写操作完成。
注意：只有 IWDT 启动后，才能修改上述寄存器。

13.3.5 窗口模式

向 WINR 寄存器写入一个小于 ARR 的值，可使 IWDT 工作于窗口看门狗模式。使能 IWDT 窗口功能后，用户只能在计数器的计数值小于等于 WINR 且大于 0 时对计数器进行重载，否则将产生溢出事件。用户可以配置 IWDT 溢出后的动作以实现溢出后产生系统复位或者中断。通过读取 CNT 寄存器，可获取当前计数器的计数值以判断当前时刻是否可以对计数器进行重载操作。

13.4 编程示例

13.4.1 配置 IWDT 为独立看门狗

Step1. 设置 SYSCTRL_APBEN1.IWDT 为 1，使能 IWDT 的配置时钟；
Step2. 向 IWDT_KR 寄存器写入 0xCCCC，启动 IWDT；
Step3. 向 IWDT_KR 寄存器写入 0x5555，解除 IWDT 寄存器的写保护；
Step4. 向 IWDT_CR 寄存器写入需要的数据以配置看门狗计数时钟的预分频系数、溢出后动作、深度休眠模式下是否自动暂停；
Step5. 向 IWDT_ARR 寄存器写入需要的数据以配置看门狗的溢出周期；
Step6. 向 IWDT_WINR 寄存器写入 0xFFF，关闭窗口功能；
Step7. 等待 IWDT_SR.ARRF 和 IWDT_SR.CRF 变为 0，相关寄存器更新完成；
Step8. 向 IWDT_KR 寄存器写入 0xAAAA，执行重载操作以加载 ARR 值到计数器；
Step9. 在程序的主循环中，定时向 IWDT_KR 寄存器写入 0xAAAA 以执行重载操作。

13.4.2 配置 IWDT 为窗口看门狗

Step1. 设置 SYSCTRL_APBEN1.IWDT 为 1，使能 IWDT 的配置时钟；
Step2. 向 IWDT_KR 寄存器写入 0xCCCC，启动 IWDT；
Step3. 向 IWDT_KR 寄存器写入 0x5555，解除 IWDT 寄存器的写保护；
Step4. 向 IWDT_CR 寄存器写入需要的数据以配置看门狗计数时钟的预分频系数、溢出后动作、深度休眠模式下是否自动暂停；
Step5. 向 IWDT_ARR 寄存器写入需要的数据以配置看门狗的溢出周期；
Step6. 等待 IWDT_SR.ARRF 和 IWDT_SR.CRF 变为 0，相关寄存器更新完成；
Step7. 向 IWDT_KR 寄存器写入 0xAAAA，执行重载操作以加载 ARR 值到计数器；
Step8. 再次向 IWDT_KR 寄存器写入 0x5555，解除 IWDT 寄存器的写保护；
Step9. 向 IWDT_WINR 寄存器写入需要的数据（应小于 ARR）以配置看门狗的窗口区间；
Step10. 等待 IWDT_SR.WINRF 变为 0，窗口寄存器更新完成；
Step11. 向 IWDT_KR 寄存器写入 0xCCCC, 使能 IWDT 相关寄存器的写保护；
Step12. 在程序的主循环中，定时读取 IWDT_CNT 以查找重载时刻点，找到后向 IWDT_KR 寄存器写入 0xAAAA 然后等待 IWDT_SR.RELOAD 变为 0 以完成重载操作。

13.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
IWDT_KR	0x4000 3000 + 0x00	键值寄存器
IWDT_CR	0x4000 3000 + 0x04	控制寄存器
IWDT_ARR	0x4000 3000 + 0x08	重载寄存器
IWDT_SR	0x4000 3000 + 0x0C	状态寄存器
IWDT_WINR	0x4000 3000 + 0x10	窗口寄存器
IWDT_CNT	0x4000 3000 + 0x24	计数值寄存器

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：暂停看门狗 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

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RC8K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。

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 时，才可修改本寄存器。
- 当溢出后动作配置为复位时，本寄存器填写的数值需要大于 0。

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 注意：连续两次读到的数值相同，才可认为读出该计数值

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

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 注意：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 注意：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 注意：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 注意：写CR寄存器后需要等待该标志变为0

14 窗口看门狗定时器（WWDT）

14.1 概述

本芯片内部集成窗口看门狗定时器(WWDT)，可用来监测由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。用户需要在设定的时间窗口内对 WWDT 刷新，否则将触发系统复位。WWDT 采用 PCLK 作为其工作时钟，适合那些要求看门狗在精确计时窗口起作用的应用程序。

14.2 主要特性

- 开启后不可关闭
- 7 比特递减计数器
- 计数时钟来自 PCLK
- 当 PCLK 为 48MHz 时，溢出周期为 85us ~ 699ms
- 计数值递减到 0x40 时产生中断
- 计数值递减到 0x3F 时产生复位
- 在窗口外进行重载操作产生复位

14.3 功能描述

本节将详细描述窗口看门狗定时器的相关功能。

14.3.1 功能框图

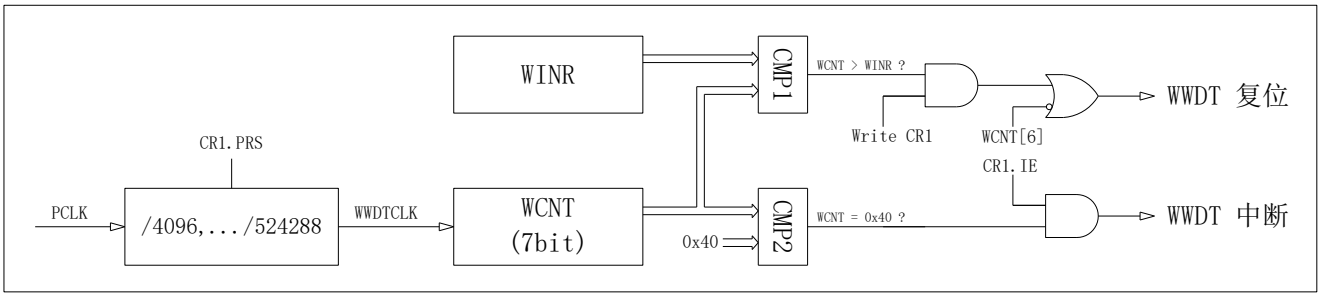

图 14-1 WWDT 功能框图

WWDT 由预分频器、递减计数器组成、窗口值寄存器组成。PCLK 时钟信号经 4096 ～ 524288 分频后驱动计数器进行递减计数。

14.3.2 运行控制

设置 CR0.EN 控制位为 1 即可启动 WWDT；WWDT 启动后不可停止，计数器从初值开始向下递减计数。

当计数器计数值为 0x40 时，WWDT 产生预溢出信号并置位 SR.POV 标志；若 CR1.IE 为 1，该预溢出信号将产生中断并执行的 WWDT 中断服务程序，用户在中断服务程序中应尽快重载 WWDT 并向 SR.POV 控制位写入 0 以清除 WWDT 预溢出标志。

当计数器计数值小于 0x40 时，WWDT 将输出复位信号并复位整个芯片。

当用户在窗口外（WCNT > WINR）更新 WCNT 时，WWDT 将输出复位信号并复位整个芯片。

当用户在窗口内更新（WCNT ≤ WINR）WCNT 时，WCNT 的数值可正常改写。

14.3.3 溢出周期

WWDT 运行时，其计数值范围为 WCNT～0x3F；其溢出周期如下所示：

$$T_{ov} = \frac{(WCNT - 0x3F) \times 4096 \times 2^{CR1.PRS}}{f_{PCLK}}$$

其中 f_{PCLK} 代表 PCLK 时钟的频率；WCNT 代表用户写入的计数器初始值；CR1.PRS 为预分频器的分频系数。当 PCLK 为 48MHz 时，其溢出时间为 85us～699ms。

14.3.4刷新计数器

WWDT 运行时，只有特定窗口期间（WINR ≥ WCNT ＞ 0x3F）才能刷新计数器，否则将产生系统复位。对 CR0 寄存器的写操作，即为刷新计数器；该操作可直接改写计数器的值。
用户应根据应用需求，合理配置 WINR 的数值及刷新时写入 CR0 的数值。

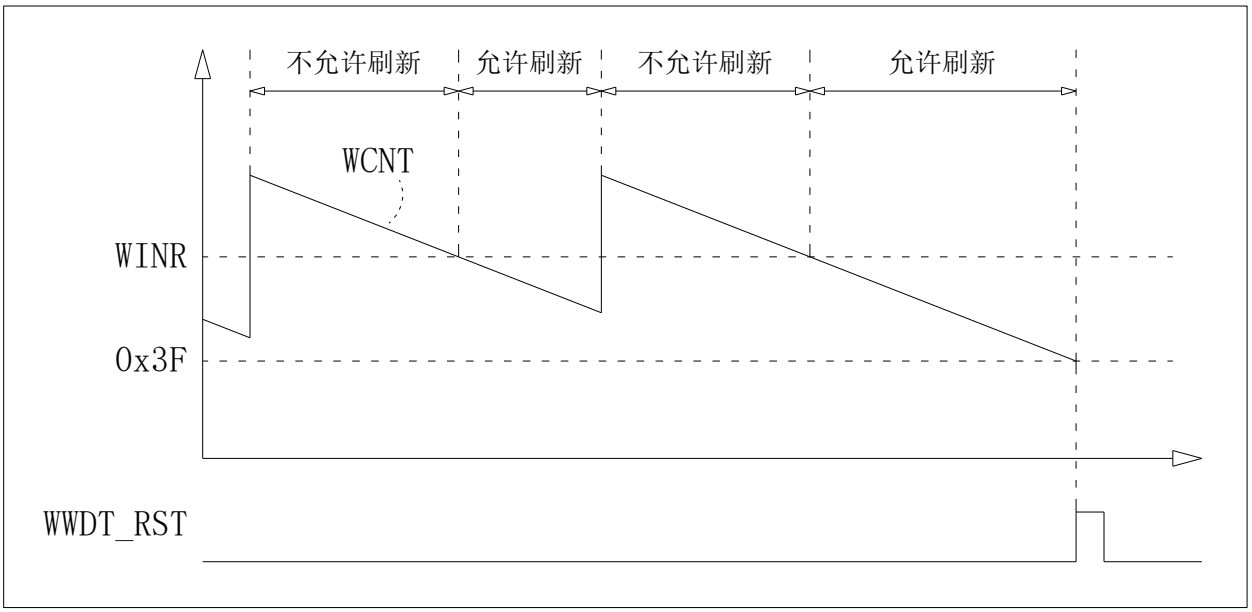

图 14-2 WWDT 时序图

14.4 编程示例

Step1. 设置 SYSCTRL_APBEN1.WWDT 为 1，使能 WWDT 的配置时钟及工作时钟；
Step2. 设置 CR1.PRS 为 2，配置 WWDT 的预分频系数为 16384；
Step3. 设置 CR0.WCNT 为 0x5F，配置计数周期为(0x5F - 0x3F) * 16384 个 PCLK；
Step4. 设置 CR1.WINR 为 0x4F，配置允许刷新窗口为计数周期的一半；
Step5. 设置 CR1.IE 为 1，使能预溢出中断；
Step6. 使能 WWDT 的中断向量表；
Step7. 设置 WWDT_CR0.EN 为 1，启动窗口看门狗；
Step8. 在中断服务程序中刷新计数器（设置 CR0.WCNT 为 0x5F），并清除 SR.POV 标志。

14.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
WWDT_CR0	0x4000 2C00 + 0x00	控制寄存器0
WWDT_CR1	0x4000 2C00 + 0x04	控制寄存器1
WWDT_SR	0x4000 2C00 + 0x08	状态寄存器

WWDT_CR0 控制寄存器 0

Address offset: 0x00 Reset value: 0x0000 007F

位域	名称	权限	功能描述
31:8	RFU	-	保留位，请保持默认值
7	EN	RW1	WWDT使能控制，使能后不能禁止 0：禁止 1：使能
6:0	WCNT	RW	读：WWDT计数器当前计数值 写：更新WWDT计数器当前计数值 注意：写入的值需要大于0x40

WWDT_CR1 控制寄存器 1

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

位域	名称	权限	功能描述
31:11	RFU	-	保留位，请保持默认值
10	IE	RW	WWDT预溢出中断使能控制 0：禁止预溢出中断 1：使能预溢出中断 注意：置1后不可清零
9:7	PRS	RW	WWDT计数时钟预分频值： 000：PCLK / 4096　　100：PCLK / 65536 001：PCLK / 8192　　101：PCLK / 131072 010：PCLK / 16384　　110：PCLK / 262144 011：PCLK / 32768　　111：PCLK / 524288
6:0	WINR	RW	看门狗窗口值

WWDT_SR 状态寄存器

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	POV	RW0	WWDT预溢出标志 R0：WWDT未发生预溢出 R1：WWDT已发生预溢出 W0：清除WWDT预溢出标志

15 自动唤醒定时器（AWT）

15.1 概述

本芯片内部集成 1 个自动唤醒定时器(AWT)；当计数器时钟源为 LSI 时，AWT 可周期性自动将 MCU 唤醒到运行模式。

15.2 主要特性

- 16bit 向下计数器
- 4 种计数时钟源：HEX / LSI / HSIOSC / ETR
- 可编程预分频器：2 ~ 32768 分频
- 支持 DeepSleep 模式下工作，溢出中断可唤醒 MCU
- 计数时钟源为 LSI 时，唤醒周期为 61us ~ 65536s

15.3 功能描述

本节将详细描述自动唤醒定时器的相关功能。

15.3.1 功能框图

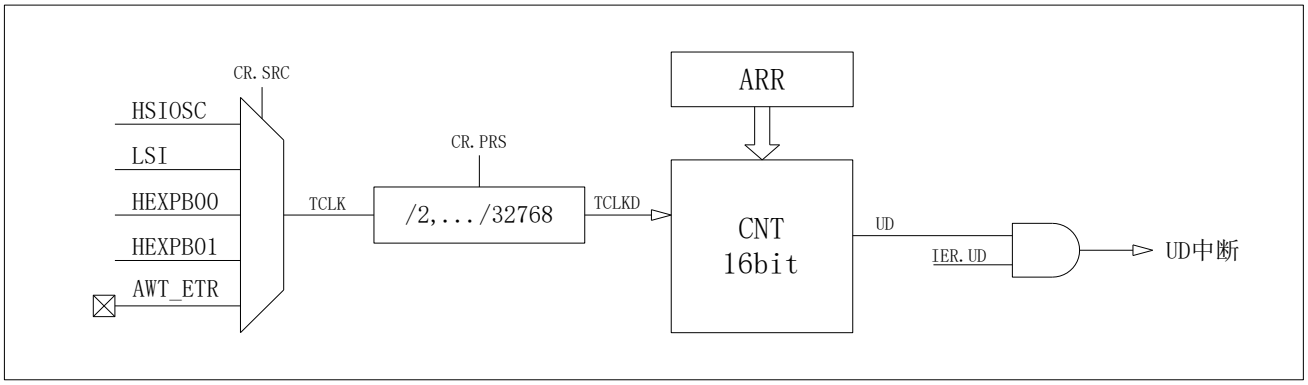

图 15-1 AWT 功能框图

- 时钟源

通过 AWT_CR.SRC 位域，可选择 4 种时钟源作为 AWT 的计数时钟；这 4 种时钟源分别为 HSIOSC、LSI、HEX、和 AWT_ETR 管脚输入的信号。

- 预分频器

AWT 内置预分频器，通过 AWT_CR.PRS 位域可配置其分频系数为 2～32768。

- 计数单元

计数单元的核心组件是一个 16 比特向下计数器 CNT 和一个 16 比特自动重载寄存器 ARR。当计数器 CNT 递减到 0 时发生下溢出，ARR 值被自动装载到 CNT，然后再次开始减计数。

15.3.2 定时及自动唤醒功能

当 AWT 的计数时钟来自内部时钟时（HSIOSC、LSI、HEX），AWT 工作于定时模式。该模式主要用于产生固定时间间隔的时基信号，其功能框图如下所示。

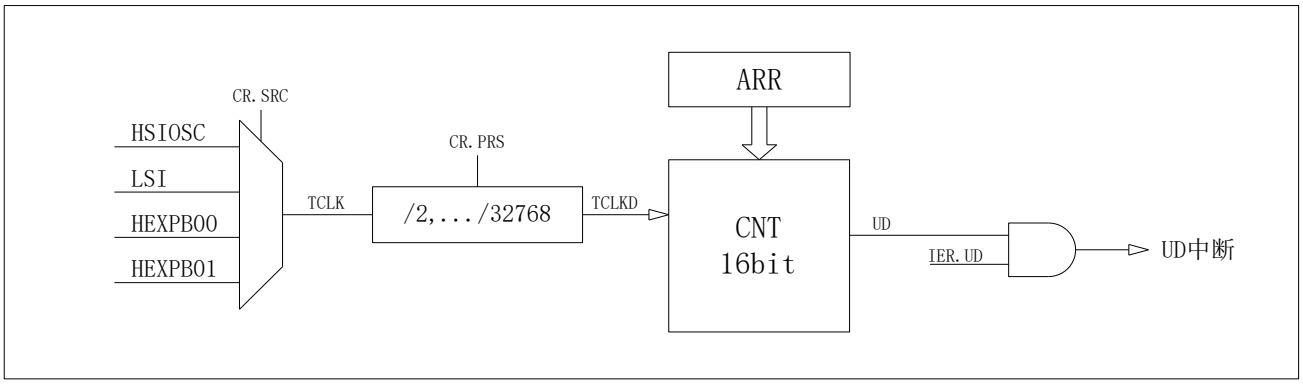

图 15-2 定时模式功能框图

通过 CR.SRC 可选择 TCLK 时钟的来源为 HSIOSC、LSI、HEX 或 LSI；通过 CR.PRS 可配置 TCLKD 时钟信号与 TCLK 时钟信号的分频比。当设置 CR.EN 为 1 后，计数单元在 TCLKD 的驱动下从 ARR 开始减计数。当计数到 0 时产生下溢出，并重新从 ARR 开始计数。每次溢出时 ISR.UD 标志会被硬件置位；若 IER.UD 为 1 则 CPU 将执行溢出中断服务程序。在退出中断服务程序之前，用户程序应向 ICR.UD 写入 0 以清除该标志。

该工作模式的定时时间间隔计算公式如下：

$$T = \frac{(ARR + 1) \times 2^{PRS}}{f_{TCLK}}$$

其中，f_{TCLK} 为 TCLK 的频率，PRS 为预分频系数，ARR 为重载值。

当 TCLK 时钟信号频率为 32768Hz 时，其定时间隔范围为 61us ~ 65536s。

当 TCLK 的来源为 LSI 时，AWT 在 DeepSleep 模式下仍可正常工作。每当 AWT 溢出产生中断时均可将 MCU 从 DeepSleep 模式唤醒到 Active 模式。利用该功能，可实现 MCU 在 DeepSleep 状态下定时自动唤醒以执行周期性任务。

15.3.3 计数功能

当 AWT 的计数时钟来自 AWT_ETR 管脚输入的信号时，AWT 工作于计数模式。该模式主要用于测定某个事件发生的次数，其功能框图如下所示。

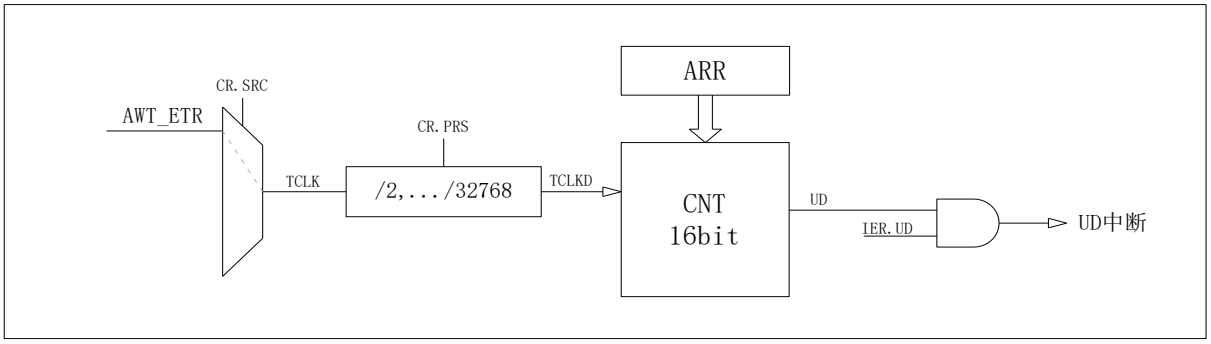

图 15-3 计数模式功能框图

通过 CR.SRC 选择 TCLK 时钟的来源为 AWT_ETR；通过 CR.PRS 可配置 TCLKD 时钟信号与 TCLK 时钟信号的分频比。当设置 CR.EN 为 1 后，计数单元在 TCLKD 的驱动下从 ARR 开始减计数。当计数到 0 时产生下溢出，并重新从 ARR 开始计数。每次溢出时 ISR.UD 标志会被硬件置位；若 IER.UD 为 1 则 CPU 将执行溢出中断服务程序。在退出中断服务程序之前，用户程序应向 ICR.UD 写入 0 以清除该标志。

注意 1：仅当 PCLK 有效时，AWT_ETR 输入的信号才能驱动计数器进行计数

注意 2：AWT_ETR 输入信号的频率应小于 PCLK 的频率

15.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
AWT_CR	0x4001 4C00 + 0x00	控制寄存器
AWT_ARR	0x4001 4C00 + 0x04	重载值寄存器
AWT_CNT	0x4001 4C00 + 0x08	计数值寄存器
AWT_IER	0x4001 4C00 + 0x10	中断使能寄存器
AWT_ISR	0x4001 4C00 + 0x14	中断标志寄存器
AWT_ICR	0x4001 4C00 + 0x1C	中断标志清除寄存器

AWT_CR 控制寄存器

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	ONESHOT	RW	保留位，请保持默认值
10:8	SRC	RW	计数时钟来源选择 000：HSIOSC时钟信号 001：LSI 010：从PB00输入的HEX时钟信号 011：从PB01输入的HEX时钟信号 100：AWT_ETR管脚输入的信号
7:4	PRS	RW	计数时钟预分频配置 0000：保留， 0001：2分频 0010：4分频 0011：8分频 0100：16分频 0101：32分频 0110：64分频 0111：128分频 1000：256分频 1001：512分频 1010：1024分频 1011：2048分频 1100：4096分频 1101：8192分频 1110：16384分频 1111：32768分频
3:1	MD	RW	工作模式配置 011：唤醒定时器模式 xxx：保留模式
0	EN	RW	定时器使能控制 0：禁止 1：使能

注：配置完成 SRC、RPS、MD 位域以及 ARR 寄存器之后才可将 EN 置 1。

AWT_ARR 重载值寄存器

Address offset: 0x04 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	计数器重载值

AWT_CNT 计数值寄存器

Address offset: 0x08 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	计数器计数值

AWT_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
31:4	RFU	-	保留位，请保持默认值
3	UD	RW	计数器下溢出中断使能配置 0：禁止 1：使能
2:0	RFU	-	保留位，请保持默认值

AWT_ISR 中断标志寄存器

Address offset: 0x14 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	UD	RO	计数器下溢出中断标志 0：定时器未溢出 1：定时器已溢出
2:0	TRIMDONE	-	保留位

AWT_ICR 中断标志清除寄存器

Address offset: 0x1C Reset value: 0x0000 003F

位域	名称	权限	功能描述
31:4	RFU	-	保留位，请保持默认值
3	UD	R1W0	计数器下溢出中断标志清除 W0：清除溢出标志 W1：无功能
2:0	TRIMDONE	-	保留位

16 通用异步收发器（UART）

16.1 概述

本芯片内部集成 2 个 UART 模块，支持异步全双工、单线半双工模式和同步半双工；支持硬件数据流控和多机通信；支持小数波特率发生器；支持深度休眠模式下接收数据。

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
  - 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
- 支持异步模式
  - 全双工
  - 单线半双工
- 支持同步模式
  - 单线半双工，同 SPI 主机模式单线半双工
  - 单工只发，同 SPI 主机模式单工只发
  - 单工只收，同 SPI 主机模式单工只收
- 支持硬件流控 RTS、CTS
- 支持多机通信
- 低功耗模式下接收数据

16.3 功能描述

本节将详细描述通用异步收发器的相关功能。

16.3.1 功能框图

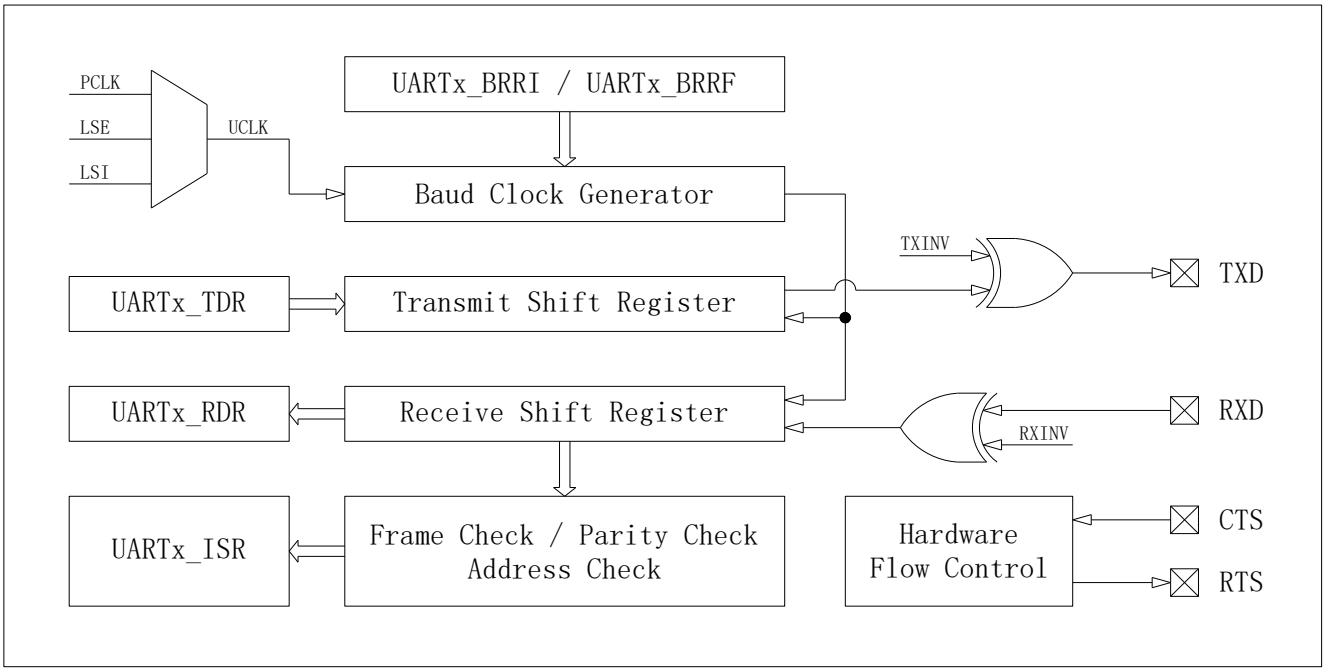

图 16-1 UART 功能框图

16.3.2工作时钟

UART 模块具有两个时钟：配置时钟 PCLK 和传输时钟 UCLK。

- 配置时钟

配置时钟固定为 PCLK，只有该时钟有效时才能对 UART 寄存器进行读写操作。该时钟可通过 SYSCTRL_APBENy 进行使能。

- 传输时钟

传输时钟用于 UART 的数据收发逻辑，通过 UART_CR2.SOURCE 可选择其来源为 PCLK 或 LSI。当其来源为 LSI 时，UART 在 DeepSleep 状态下仍可正常进行数据收发。

16.3.3 同步模式

当设置 UART_CR1.SYNC 为 1 时，UART 工作于同步模式；TXD、RXD 管脚的输入输出方向由硬件自动控制；UARTx_TXD 管脚输出同步移位时钟信号，UARTx_RXD 管脚进行数据的发送和接收。工作于同步模式的 UART 主机可以与工作于单工或单线半双工模式的 SPI 从机（CPOL=1，CPHA=1）进行通信。

16.3.3.1 波特率设置

同步模式下，波特率计算公式为：$BaudRate = \frac{f_{UCLK}}{12}$

其中，f_{UCLK} 是 UART 的传输时钟的频率。

16.3.3.2 数据收发

当设置 UARTx_CR1.TXEN 为 1 时，UART 工作于发送状态；将数据写入 UARTx_TDR 寄存器后，数据从 UARTx_RXD 管脚串行输出，同时 UARTx_TXD 管脚输出同步移位时钟信号。当数据发送完成后，UARTx_ISR.TC 会被硬件置 1。

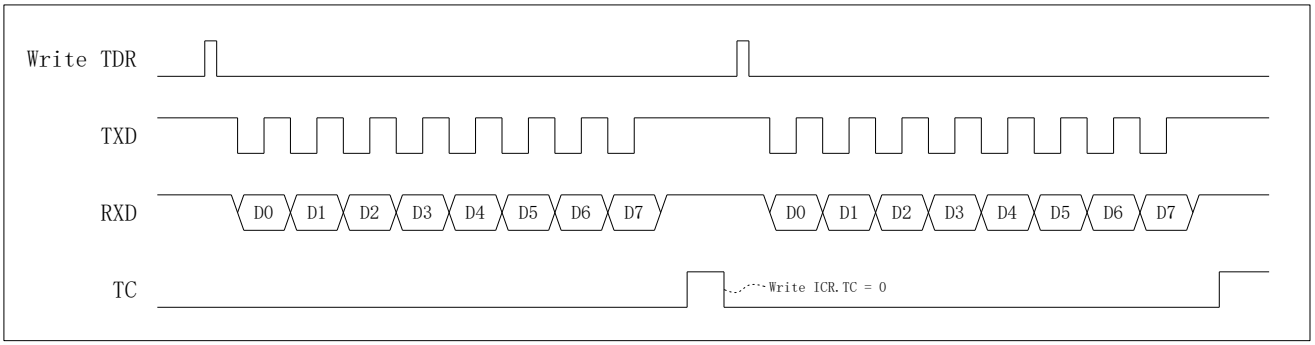

图 16-2 同步半双工数据发送示例

当设置 UARTx_CR1.RXEN 为 1 时，UART 工作于接收状态；UARTx_TXD 管脚将输出同步移位时钟信号，数据从 UARTx_RXD 管脚串行输入。每当 ISR.RC 标志为 0 时，TXD 引脚会输出 8 个同步移位时钟，数据将从 RXD 输入。每次将 ISR.RC 标志清零即可接收下一个字节。

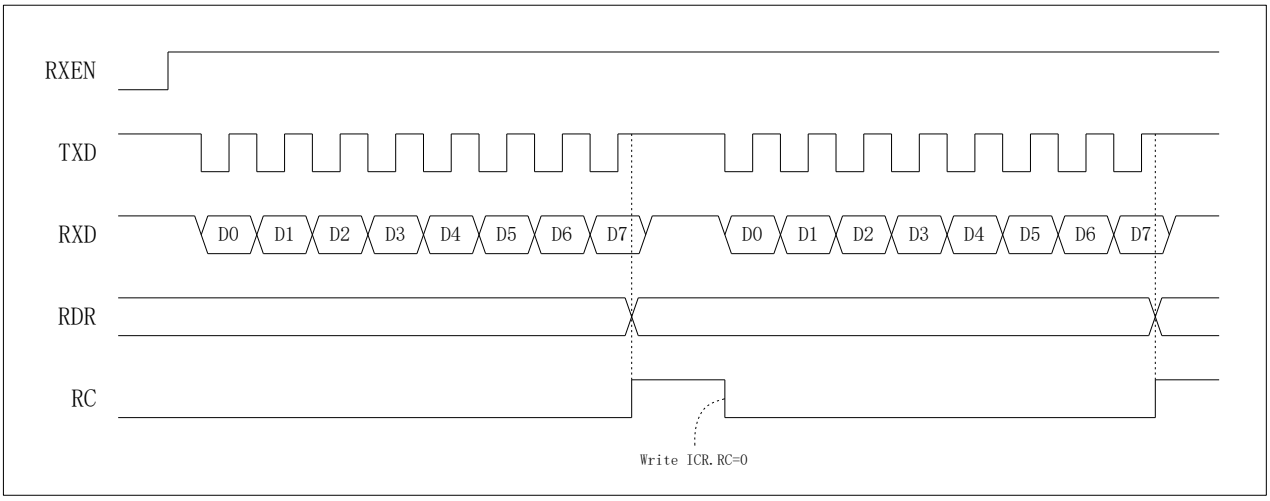

图 16-3 同步半双工数据接收示例

16.3.4异步模式

当设置 UART_CR1.SYNC 为 0 时，UART 工作于异步模式；TXD、RXD 管脚的输入输出方向由硬件自动控制；通过 UARTx_TXD 管脚发送数据，通过 UARTx_RXD 管脚接收数据。

16.3.4.1 数据帧格式

UART 异步通信以帧的形式进行数据传输；每个帧包括 1 个起始位、8 比特数据域、可选的校验位和 1~2 个停止位。

- 起始位

发送时，TXD 发送 1 比特宽度的低电平作为起始位。

接收时，可选择 RXD 信号的下降沿或低电平作为起始位的判定方式。建议选择 RXD 信号的下降沿作为起始位判定方式；低电平作为起始位的判定方式主要用于从深度休眠唤醒后接收数据。

- 数据域

数据域具有 8 比特数据位；

- 校验位

通过 UARTx_CR1.PARITY 位域可配置校验方式为无校验、奇校验、偶校验和自定义校验。

无校验：不发送校验位

奇校验：硬件自动配置校验位，使数据位和校验位中“1”的总数为奇数。

偶校验：硬件自动配置校验位，使数据位和校验位中“1”的总数为偶数。

自定义校验：软件配置校验位，主要应用于多机通信。

- 停止位

通过 UARTx_CR1.STOP 可配置停止位长度为 1、1.5 或 2 比特。

16.3.4.2 小数波特率发生器

UART 支持四种波特率发生方式：16 倍采样、8 倍采样、4 倍采样和专用采样这 4 种采样模式。波特率发生方式如下所示：

CR1.OVER	生成模式	计算公式	备注
00	16倍模式	$BaudRate = \frac{f_{UCLK}}{16 \times BRRI + BRRF}$	生成小数波特率
01	8倍模式	$BaudRate = \frac{f_{UCLK}}{8 \times BRRI}$	生成整数波特率 BRRF需保持为0
10	4倍模式	$BaudRate = \frac{f_{UCLK}}{4 \times BRRI}$	生成整数波特率 BRRF需保持为0
11	专用模式	$BaudRate = \frac{256 \times f_{UCLK}}{BRRI}$	用于LSI生成9600~2400 BRRF需保持为0

其中 f_{UCLK} 代表 UCLK 的频率，BRRI 代表 BRRI 寄存器的值，BRRF 代表 BRRF 寄存器的值。

例：在 16 倍模式下，当 UCLK 频率为 8MHz 时，配置通信波特率为 115200

BRRI = f_{UCLK} / BaudRate / 16 = 8000000 / 115200 / 16= 4.34，取 BRRI 为 4；

BRRF = f_{UCLK} / BaudRate - BRRI*16 = 8000000 / 115200 - 16 * 4 = 5.44，取 BRRF 为 5；

实际波特率 BaudRate = f_{UCLK} / (16 * BRRI + BRRF) = 8000000 / (16 * 4 + 5) = 115942；

偏差率为 115942 / 115200 - 1 = 0.64%。

16.3.4.3 常用波特率配置表

以下列举各种常用主频下的常用波特率配置

表 16-1 波特率配置表（f_{UCLK}=8MHz）

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%
19200	26	1	19184.65	-0.08%
38400	13	0	38461.54	0.16%
57600	8	11	57553.96	-0.08%
115200	4	5	115942.03	0.64%
230400	2	3	228571.43	-0.79%
500000	1	0	500000	0.00%

表 16-2 波特率配置表（f_{UCLK}=16MHz）

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%
19200	52	1	19207.68	0.04%
38400	26	1	38369.30	-0.08%
57600	17	6	57553.96	-0.08%
115200	8	11	115107.91	-0.08%
230400	4	5	231884.06	0.64%
500000	2	0	500000.00	0.00%
1000000	1	0	1000000.00	0.00%

表 16-3 波特率配置表（f_{UCLK}=24MHz）

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%
19200	78	2	19200.00	0.00%
38400	39	1	38400.00	0.00%
57600	26	1	57553.96	-0.08%
115200	13	0	115384.62	0.16%
230400	6	8	230769.23	0.16%
500000	3	0	500000.00	0.00%
1000000	1	8	1000000.00	0.00%

表 16-4 波特率配置表（f_{UCLK}=32MHz）

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%
19200	104	3	19196.16	-0.02%
38400	52	1	38415.37	0.04%
57600	34	12	57553.96	-0.08%
115200	17	6	115107.91	-0.08%
230400	8	11	230215.83	-0.08%
500000	4	0	500000.00	0.00%
1000000	2	0	1000000.00	0.00%
1500000	1	5	1523809.52	1.59%
2000000	1	0	2000000.00	0.00%

表 16-5 波特率配置表（f_{UCLK}=48MHz）

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%
19200	156	4	19200.00	0.00%
38400	78	2	38400.00	0.00%
57600	52	1	57623.05	0.04%
115200	26	1	115107.91	-0.08%
230400	13	0	230769.23	0.16%
500000	6	0	500000.00	0.00%
1000000	3	0	1000000.00	0.00%
1500000	2	0	1500000.00	0.00%
2000000	1	8	2000000.00	0.00%
3000000	1	0	3000000.00	0.00%

表 16-6 波特率配置表（f_{UCLK}=32768Hz）

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%

16.3.4.4 发送数据

当设置 CR1.TXEN 为 1 时，TXD 管脚可对外发送数据。向 TDR 寄存器写入数据后，数据会从最低位开始从 TXD 管脚按波特率串行输出。向 TDR 寄存器写入数据时，ISR.TXE 标志会被硬件清零；当 TDR 寄存器中的数据被转移到发送移位寄存器时，ISR.TXE 标志会被硬件置 1。只有 ISR.TXE 为 1 时，才能向 TDR 寄存器写入数据。每当发送移位寄存器中的数据帧发送完成时 ISR.TC 都会被硬件置位，代表已经发送完成一帧数据。当 TDR 寄存器或发送移位寄存器中有待发送的数据时，ISR.TXBUSY 标志会被硬件置 1；当 TDR 寄存器和发送移位寄存器中均没有待发送的数据时，ISR.TXBUSY 标志会被硬件清零。

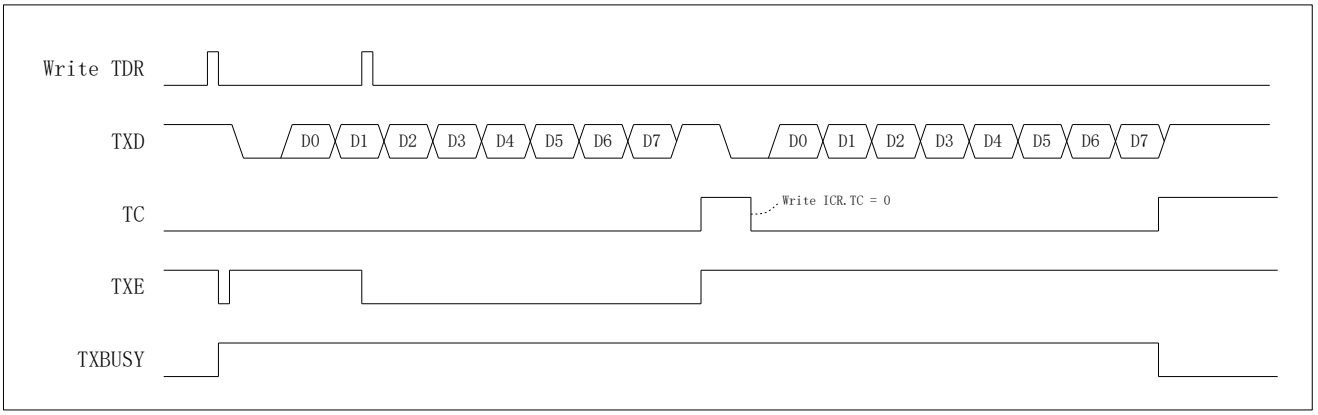

图 16-4 异步模式数据发送示例

16.3.4.5 接收数据

当设置 CR1.RXEN 为 1 时，可从 RXD 管脚接收数据。当 RXD 管脚检测到起始位后，接收电路按波特率从 RXD 管脚依次接收每一比特数据；接收完成后将数据存入 RDR 寄存器。当接收数据从移位寄存器转移到 RDR 寄存器时，ISR.RC 标志会被硬件置 1；用户检测到 ISR.RC 标志为 1 时应尽快从 RDR 寄存器中读出收到的数据并清除 ISR.RC 标志。如未能及时读取 RDR 寄存器内的数据，新接收的数据会覆盖 RDR 寄存器中的旧数据，造成数据丢失。如接收到的数据帧的校验位出错，ISR.PE 标志会被硬件置位。如果在预期的停止位时刻点未检测到高电平，ISR.FE 标志会被硬件置位以指示帧结构错误。

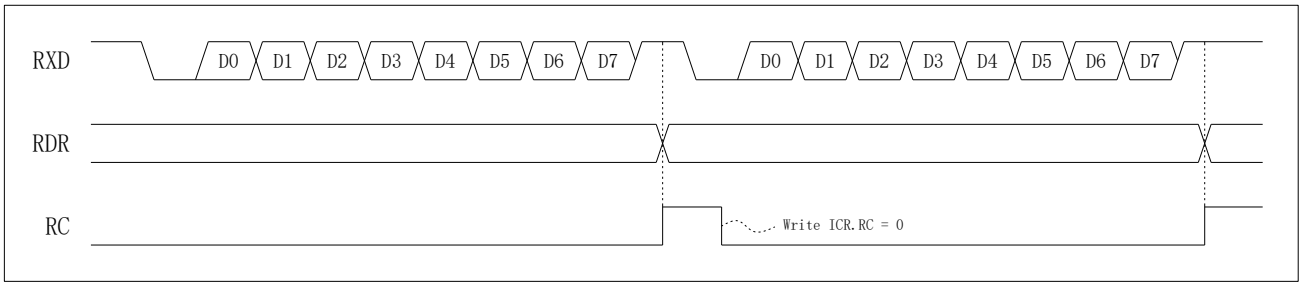

图 16-5 异步模式数据接收示例

16.3.4.6 单线半双工

当设置 CR2.SIGNAL 为 1 时，UART 工作于单线半双工模式。仅使用 TXD 管脚进行数据的发送和接收，RXD 管脚可用作其它功能。用户应将 TXD 管脚配置为开漏并外接上拉电阻。
当没有发送数据时，TXD 管脚被自动配置为输入状态，等待接收总线上其它主机发出数据。每当用户向 TDR 寄存器写入数据后，TXD 管脚被自动配置为输出态以便向总线上发送用户写到 TDR 内寄存器的数据。为防止单总线出现冲突，用户应采取适当的应用层保护措施，以确保不会出现多个主机同时向总线发送数据。

16.3.4.7 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个从机，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

主机需设置 CR1.PARITY 为 1，以发送和接收 9 比特数据。当 TDR[8]为 1 时，表示主机发送的是地址帧；当 TDR[8]为 0 时，表示主机发送的是数据帧。

- 从机接收

从机需设置 CR1.PARITY 和 CR2.ADDREN 为 1，发送和接收 9 比特数据并使能从机地址识别。当从机收到地址帧且地址匹配时，从机会将接收到的地址帧保存到 RDR 寄存器中并置位 ISR.RC 标志和 ISR.MATCH 标志。当从机收到不匹配的地址帧时，会自动清零 ISR.MATCH 标志。
从机程序查询到 RC 标志时，应通过 RDR[8]判断接收到的是地址帧还是数据帧。从机程序在发送数据帧时，需要将 TDR[8]设置为 0，以避免该数据帧被其它从机当作地址帧。

- 从机地址与地址掩码

从机地址由 ADDR 寄存器和 MASK 寄存器共同决定：当(UART_MATCH & UART_ADDR) 与 (UART_MATCH & UART_RDR)相等时，即为收到了匹配的地址。
例：从机设置 UART_ADDR=0xA0，UART_MASK=0xF0；则本从机可以响应主机发出的地址为 0xA0 ~ 0xAF 的地址帧。

- 广播地址

地址 0xFF 被定义为广播地址，主机发送广播地址帧时，所有的从机均可收到该地址帧。

16.3.4.8 硬件流控

本芯片支持 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。

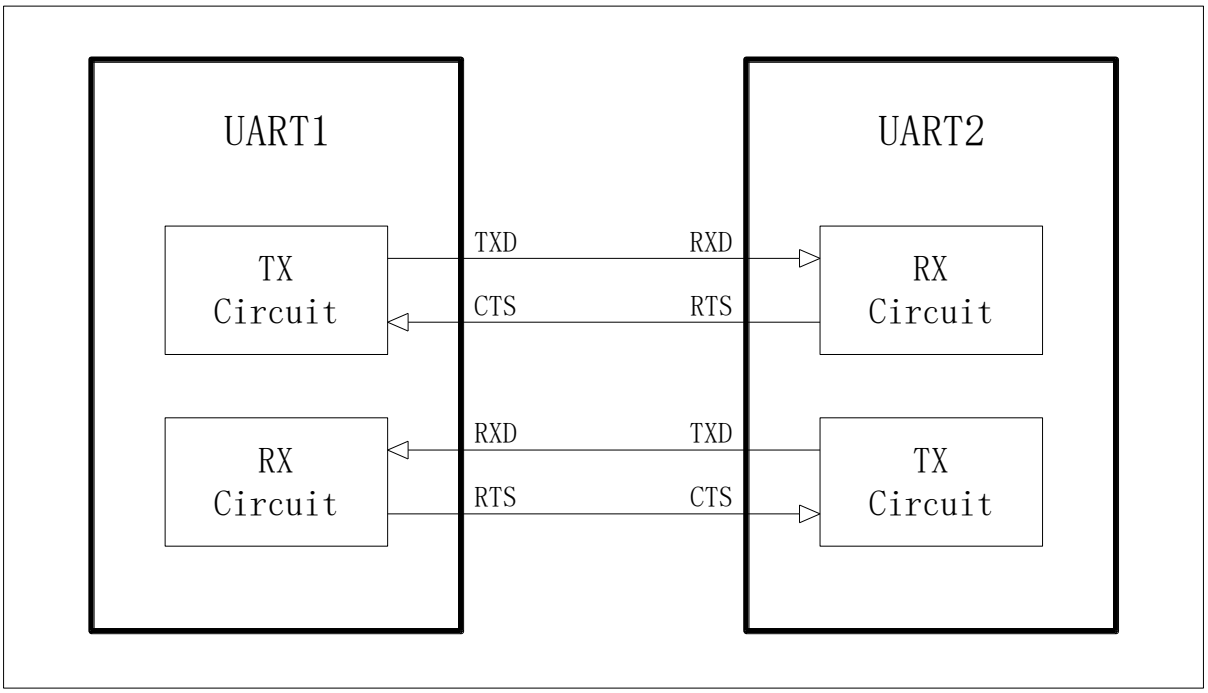

图 16-6 硬件流控示意图

- RTS 流控

当设置 CR2.RTSEN 为 1 时，本模块具有 RTS 流控功能。当本模块准备好接收新的数据帧时，RTS 管脚就会输出低电平；当本模块不能再接收新的数据帧时，RTS 管脚输出高电平。只有满足如下三个条件时，RTS 管脚才会输出低电平。

- CR1.RXEN 为 1，已使能接收电路
- RDR 寄存器内的数据已被用户读取
- ISR.PE/FE 为 0，没有奇偶校验错误、帧结构错误

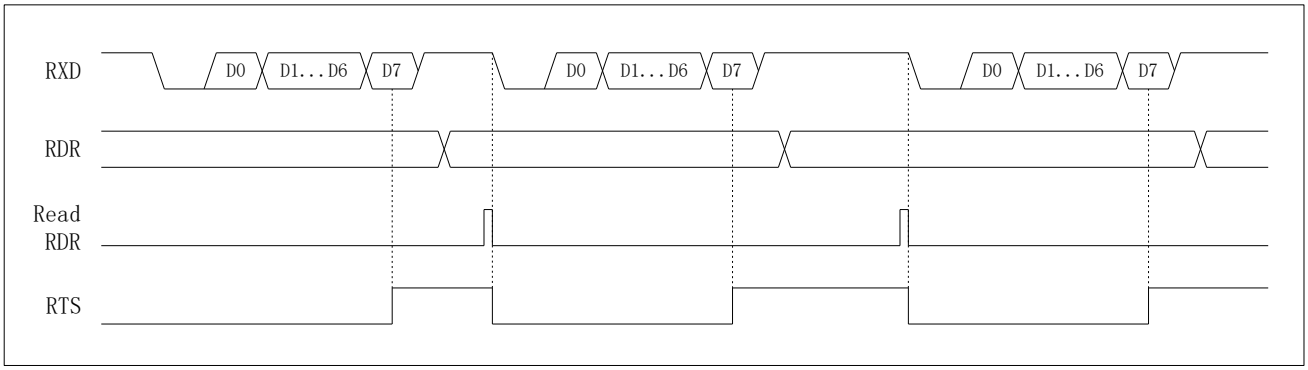

图 16-7 RTS 硬件流控时序图

- CTS 流控

当设置 CR.CTSEN 为 1 时，本模块具有 CTS 流控功能。当 CTS 管脚输入为高电平时，TXD 管脚不会向外发送数据；当 CTS 管脚输入为低电平时，TXD 管脚向外发送数据。若正在发送数据时，CTS 管脚输入信号变成高电平，则正在发送的数据帧不受影响；当前数据帧数送完成后暂停发送新的数据。

通过 ISR.CTSLV 标志可获取当前 CTS 管脚的电平状态。CTS 信号电平发生变化时，ISR.CTS 标志会被硬件置 1，若使能该中断（IER.CTS=1），则会产生 CTS 电平变化中断。

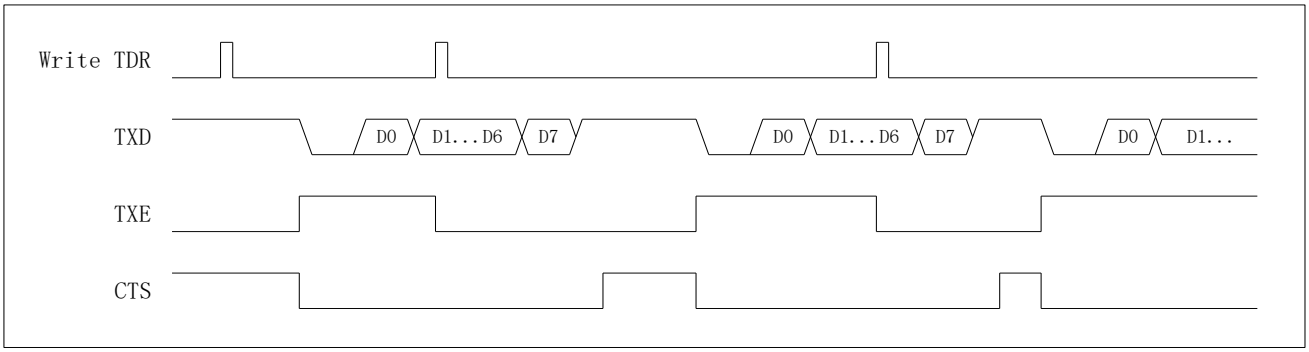

图 16-8 CTS 硬件流控时序图

16.3.4.9 深度休眠模式下接收数据

本 UART 模块支持两种深度休眠模式下接收数据：低速时钟接收数据、GPIO 辅助接收数据。

- 低速时钟接收数据

该方式需设置 UART 传输时钟（UCLK）来源为 LSI，使能 UART 接收完成中断。该方式支持的通信波特率为 9600、4800、2400。

每当 UART 接收完成一帧数据时，UART 接收完成中断会唤醒深度休眠模下的 CPU，用户代码应在该中断服务程序中对接收到的数据进行处理。

- GPIO 辅助接收数据

该方式需设置 START 位判定方式为低电平，设置系统时钟来源为 HSI，使能 RXD 所在 GPIO 的下降沿中断。该方式支持所有不大于 19200 的通信波特率。

每当 RXD 管脚出现下降沿时，GPIO 中断会唤醒深度休眠模式下的 CPU，UART 模块获得 PCLK 并开始接收数据帧，用户代码应在 GPIO 中断服务程序中等待当前数据帧接收完成并对接收到的数据进行处理。

16.4 编程示例

16.4.1 发送数据示例

Step1. 通过 SYSCTRL_AHBEN 寄存器使能 TXD 管脚所在 GPIO 的工作时钟；
Step2. 按 GPIO 章节相关描述，将 TXD 功能复用到选定的管脚；
Step3. 通过 SYSCTRL_APBENx 寄存器使能 UART 配置时钟；
Step4. 设置 CR1.SYNC 为 0，配置 UART 工作于异步全双工模式；
Step5. 通过 CR2.SOURCE 位域配置传输时钟来源；
Step6. 通过 CR1.OVER 位域配置波特率采样模式；
Step7. 通过 BRRI / BRRF 寄存器配置所需要的通信波特率；
Step8. 通过 CR1.PARITY 位域配置所需要的奇偶校验证方式；
Step9. 通过 CR1.STOP 位域配置所需要的 STOP 位长度；
Step10. 设置 CR1.TXEN 为 1，使能 UART 发送电路；
Step11. 查询等待 ISR.TXE 标志变为 1，TDR 寄存器为空；
Step12. 将待发送的数据写入 TDR 寄存器；
Step13. 如待发送的数据未完成，则跳转到 Step11 继续执行；
Step14. 查询等待 ISR.TXBUSY 标志变为 0，所有待发送数据均已通过 TXD 管脚发送完成。

16.4.2 接收数据示例

Step1. 通过 SYSCTRL_AHBEN 寄存器使能 RXD 管脚所在 GPIO 的工作时钟；
Step2. 按 GPIO 章节相关描述，将 RXD 功能复用到选定的管脚；
Step3. 通过 SYSCTRL_APBENx 寄存器使能 UART 配置时钟；
Step4. 设置 CR1.SYNC 为 0，配置 UART 工作于异步全双工模式；
Step5. 设置 CR1.START 为 0，配置 START 位的判定方式为 RXD 信号下降沿；
Step6. 通过 CR2.SOURCE 位域配置传输时钟来源；
Step7. 通过 CR1.OVER 位域配置波特率采样模式；
Step8. 通过 BRRI / BRRF 寄存器配置所需要的通信波特率；
Step9. 通过 CR1.PARITY 位域配置所需要的奇偶校验证方式；
Step10. 通过 CR1.STOP 位域配置所需要的 STOP 位长度；
Step11. 设置 CR1.RXEN 为 1，使能 UART 接收电路；
Step12. 查询等待 ISR.RC 标志变为 1，接收完成一帧数据；
Step13. 查询 ISR.PE 和 ISR.FE，以确认接收到的数据帧是否有效；若无效则进行出错处理，若有效则从 RDR 寄存器读出数据并保存；
Step14. 向 ICR.RC 写入 0x00，清除接收完成标志；
Step15. 如待接收的数据未完成，则跳转到 Step12 继续执行；

16.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	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_IER	UARTx_Base + 0x08	中断使能寄存器
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_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

UART1_Base = 0x4001 3800
UART2_Base = 0x4000 4400

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
31:11	RFU	-	保留位，请保持默认值
10:9	OVER	RW	波特率生成方式设置 00：16倍方式， BaudRate = UCLK / (BRRI* 16 + BRRF) 01：8倍方式， BaudRate = UCLK / (BRRI* 8) 10：4倍方式， BaudRate = UCLK / (BRRI* 4) 11：专用方式， BaudRate = UCLK * 256 / BRRI 注意1：非16倍方式模式，需设置BRRF为0 注意2：专用方式仅适合传输时钟为LSI且目标波特率为2400、4800、9600的情况
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	RFU	-	保留位，请保持默认值
6	SYNC	RW	同步或异步模式配置 0：异步模式 1：同步模式
5:4	STOP	RW	停止位长度设置 00：1 位 01：1.5 位 10：2 位 11：保留 注意：同步模式下STOP位须配置为1位
3:2	PARITY	RW	校验位设置 00：无奇偶校验 01：自定义校验 10：偶校验 11：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

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
31:10	RFU	-	保留位，请保持默认值
9:8	SOURCE	RW	UART传输时钟来源配置 00：UCLK来自PCLK 01：UCLK来自PCLK 11：UCLK来自LSI
7:6	RFU	-	保留位，请保持默认值
5	TXINV	RW	TXD管脚输出信号反相控制 0：TXD管脚输出同相信号 1：TXD管脚输出反相信号
4	RXINV	RW	RXD管脚输入信号反相控制 0：RXD管脚信号直接送入接收电路 1：RXD管脚信号反相送入接收电路
3	RTSEN	RW	RTS流控信号使能控制 0：禁止 1：使能
2	CTSEN	RW	CTS流控信号使能控制 0：禁止 1：使能
1	SINGLE	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
0	ADDREN	RW	从机地址识别使能控制 0：禁止 1：使能

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

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

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
31:9	RFU	-	保留位，请保持默认值
8:0	TDR	WO	待发送的数据 当CR1.PARITY设置为自定义校验时，TXD管脚会发送TDR[8]

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 当CR1.PARITY设置为自定义校验时，接收电路会接收RDR[8]

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
31:7	RFU	-	保留位，请保持默认值
6	CTS	RW	CTS信号变化中断使能控制 0：禁止 1：使能
5	RFU	-	保留位，请保持默认值
4	PE	RW	奇偶校验错误中断使能控制 0：禁止 1：使能
3	FE	RW	帧结构错误中断使能控制 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
0	TXE	RW	TDR寄存器空中断使能控制 0：禁止 1：使能

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
31:9	RFU	-	保留位，请保持默认值
8	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
7	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
6	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
5	MATCH	RO	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
4	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
3	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
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
1	TC	RO	发送完成中断标志 0：尚未完成一帧数据的发送 1：已经完成一帧数据的发送
0	TXE	RO	TDR寄存器空标志 0：TDR寄存器内有数据 1：TDR寄存器内无数据

UARTx_ICR 中断标志清除寄存器

Address offset: 0x20 Reset value: 0x0000 00FF

位域	名称	权限	功能描述
31:7	RFU	-	保留位，请保持默认值
6	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
4	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
3	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
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

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

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

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。本芯片内部集成 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
- 从机模式下通信速率高达 PCLK/4
- 支持多机通信模式
- 8 个带标志位的中断源

17.3 功能描述

本节将详细描述 SPI 接口的相关功能。

17.3.1 功能框图

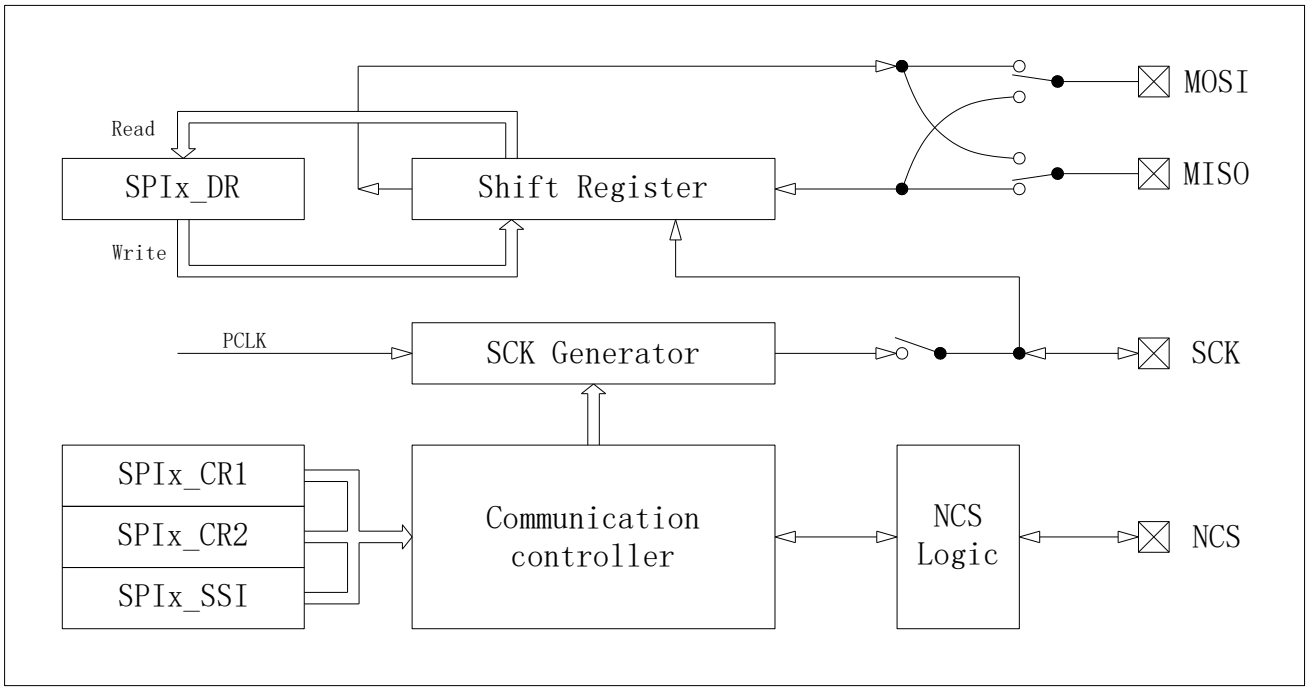

图 17-1 SPI 功能框图

SPI 一般通过 4 个管脚与外部器件通信：

- MOSI：主机输出/从机输入管脚；主机模式下发送数据，从机模式下接收数据
- MISO：主机输入/从机输出管脚；主机模式下接收数据，从机模式下发送数据
- SCK：同步串行时钟管脚；主机模式下发出时钟，从机模式下接收时钟
- NCS：从机选择管脚；选择从机或多主机冲突检测

17.3.2通信时序和数据格式

- 通信时序

在 SPI 通信过程中，NCS、SCK、MOSI 信号由主机产生，MISO 信号由从机产生。NCS 信号的下降沿是 SPI 通信的起始信号，在通信过程中 NCS 需保持低电平。在每个 SCK 的时钟周期中，MOSI 和 MISO 信号线传输一比特数据。SPI 通信格式取决于串行时钟极性、串行时钟相位和数据帧格式。数据帧格式又取决于数据宽度和数据比特位顺序。SPI 接口正常通信要求所有的主机和从机必须配置为相同的通信格式。

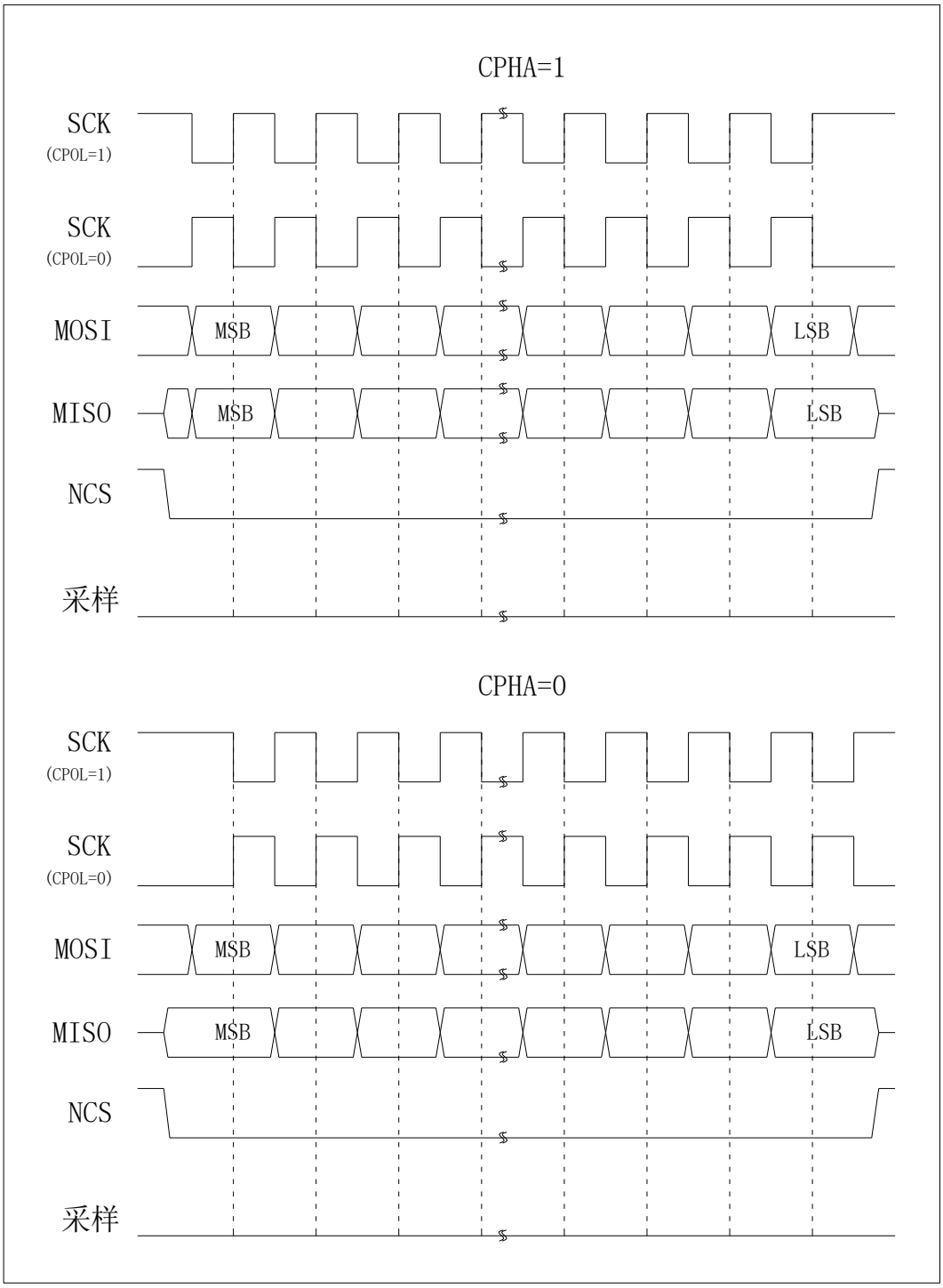

图 17-2 SPI 通信时序图

- 数据帧格式

数据帧宽度由CR1.WIDTH位域配置，可设置4～16bit任意数据位宽。数据的大小端由CR1.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）。

- 时钟频率

当SPI模块作为主机时，需向总线输出SCK时钟信号；通过CR1.BR位域可设置该时钟信号的频率为$f_{PCLK}/2$～$f_{PCLK}/128$。当SPI模块作为从机时，无需配置CR1.BR位域。

- 时钟极性、时钟相位

时钟极性（CPOL）是指总线空闲状态时SCK电平状态。当设置CR1.CPOL为0时，SCK时钟线在空闲时为低电平；当设置CR1.CPOL为1时，SCK时钟线在空闲时为高电平。

时钟相位（CPHA）是指数据的采样和移位时刻。当设置CR1.CPHA为0时，在SCK的前边沿采样、后边沿移位；当设置CR1.CPHA为1时，在SCK的前边沿移位、后边沿采样。

根据CPOL和CPHA的不同配置SPI总线可设置4种通信模式，主机和从机需要配置为相同的通信模式才能正常通信。

CPOL	CPHA	空闲时SCK电平	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 延迟采样

当 SPI 的速率太快时，信号在电路传输中的延迟造成 SPI 主机无法正确接收到从机发出的 MISO 信号。当设置 CR1.SMP 为 1，则 SPI 主机将在延迟半个 SCK 周期的时刻点对 MISO 进行采样。该功能可有效提高 SPI 主机的数据接收速率。

下图展示了 CPOL/CPHA 均为 0 时的延迟采样时序。主机和从机所发送与接收的 SCK 及 MISO 均存在一定的延迟；当环路延时造成主机在 SCK 的上升沿不能采样到正确的 MISO 电平时必须采用延迟采样来接收从机发来的数据。

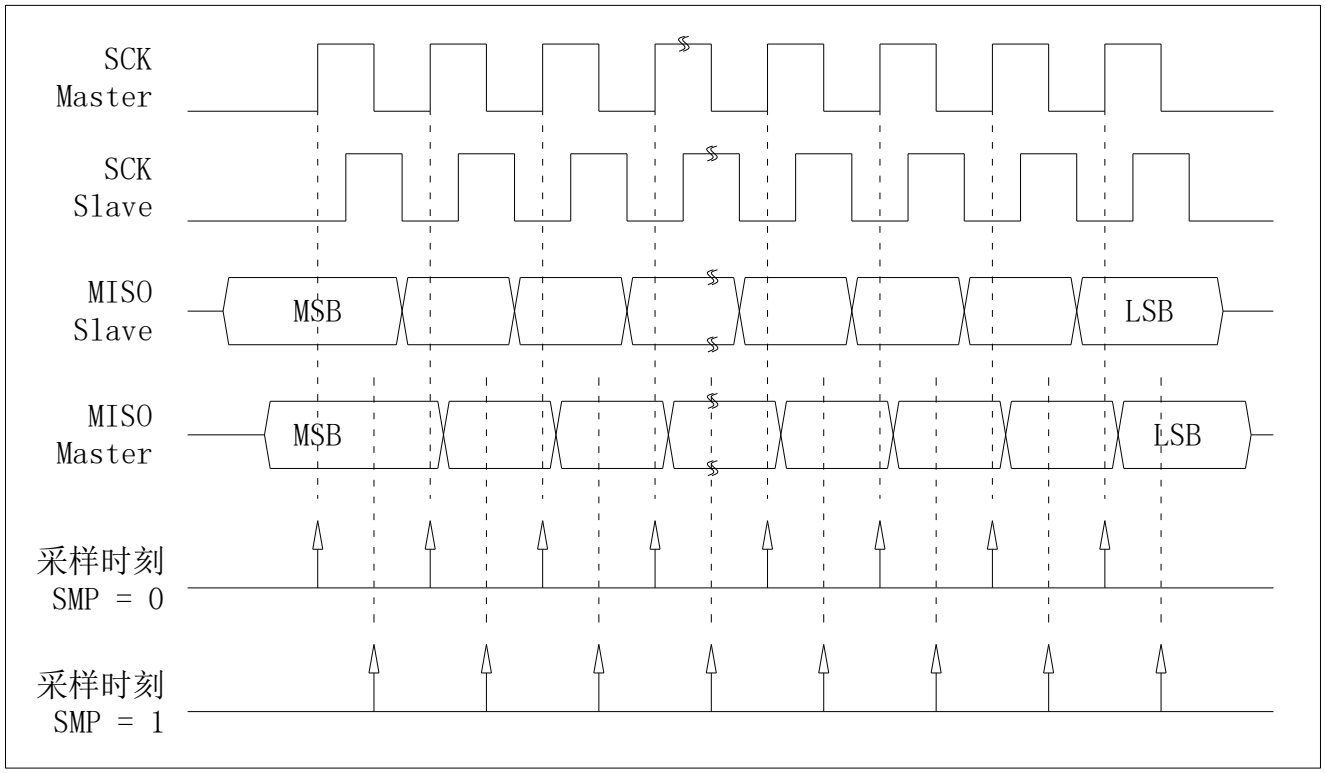

图 17-3 延迟采样时序图

17.3.4 从机选择信号

- SPI 主机模式

当 CR1.SSM 为 0 时：NCS 管脚需配置为输入，用于检测多主机模式下的总线冲突。
当 CR1.SSM 为 1 时：NCS 管脚需配置为输出，通过 SSI.SSI 设置 NCS 管脚的输出电平。

- SPI 从机模式

当 CR1.SSM 为 0 时：NCS 管脚需配置为输入，NCS 管脚的低电平可以选中本 SPI 从机。
当 CR1.SSM 为 1 时：无需 NCS 管脚，软件设置 SSI.SSI 为 0 可以选中本 SPI 从机。

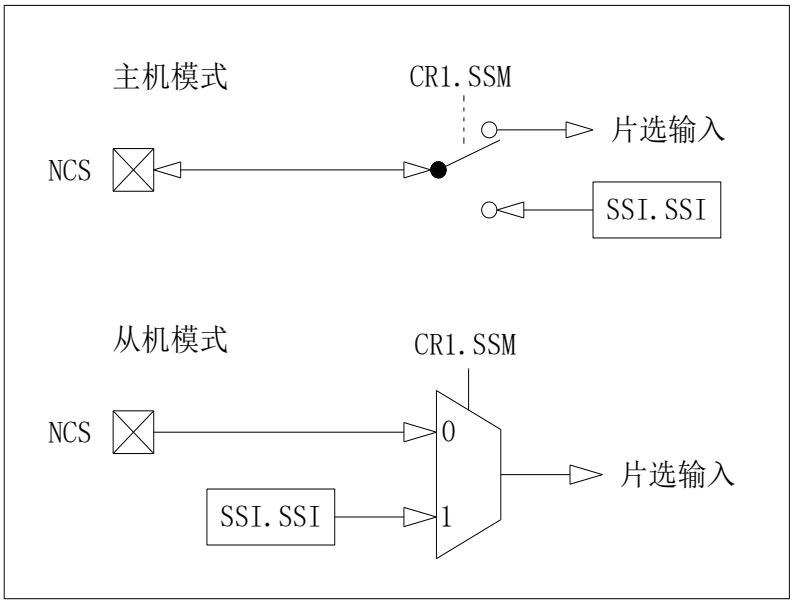

图 17-4 从机选择信号的配置

17.3.5 从机 MISO 管脚配置

当 SPI 模块作为从机且未被主机选中时，可通过寄存器配置 MISO 管脚的输出状态。
当设置 CR1.MISOHD 为 0 时，MISO 管脚始终为 CMOS 输出状态；仅适用于单一从机的场景。
当设置 CR1.MISOHD 为 1 时，MISO 管脚为高阻状态；适用于多从机的场景。

17.3.6全双工模式

当设置 CR1.MODE 为 0 时，SPI 工作于全双工通信模式。主机和从机通过 MOSI、MISO 两条单向数据线进行连接，在主机提供的 SCK 时钟信号控制下进行数据传输。

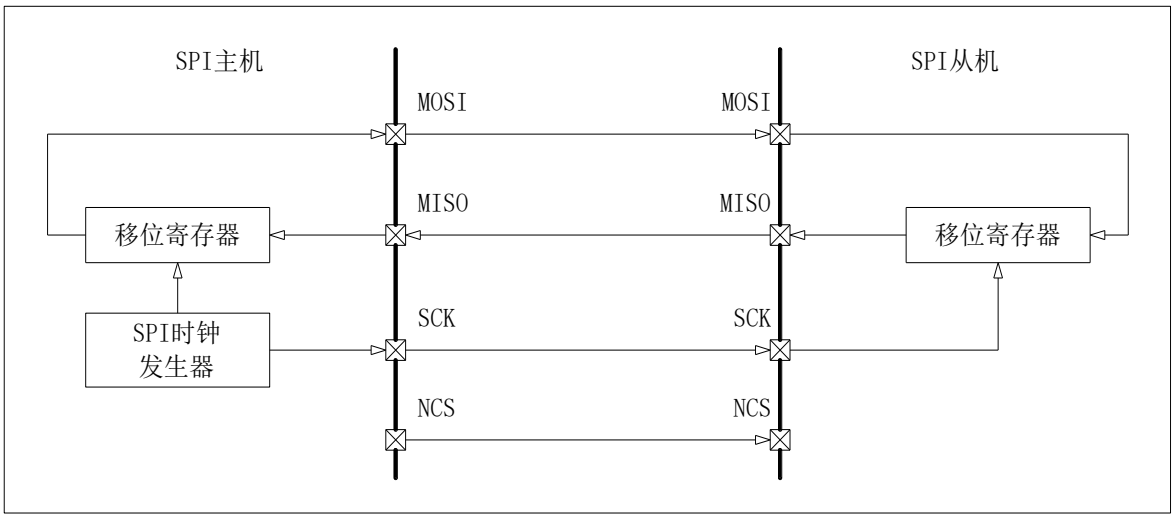

图 17-5 全双工通信连接图

- 主机收发

当设置 CR1.MSTR 为 1 时，SPI 模块工作于主机模式。

设置 SSI.SSI 为 0，NCS 管脚输出低电平以选中从机。当用户向 SPI_DR 寄存器写入待发送的数据后，发送移位寄存器在 SCK 的控制下从 MOSI 管脚依次输出待发送数据的每一个比特；同时接收移位寄存器在 SCK 的控制下从 MISO 管脚依次读入从机发送的每一个比特。一帧数据传输完成时，ISR.RXNE 由硬件置 1，用户应及时从 SPI_DR 寄存器中读出接收到的数据。

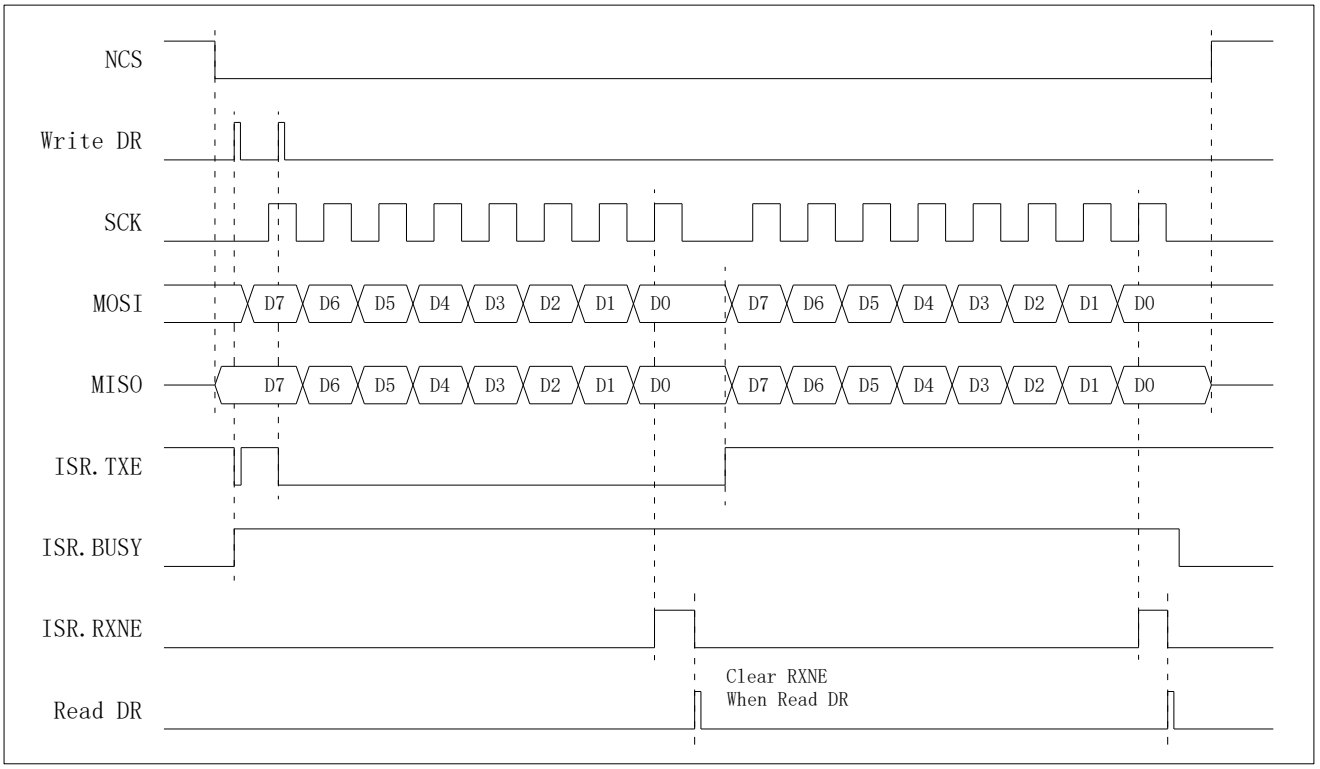

图 17-6 全双工主机收发示例

- 从机收发

当设置 CR1.MSTR 为 0 时，SPI 模块工作于从机模式。

当 NCS 信号被拉低后，发送移位寄存器在 SCK 的控制下从 MISO 管脚依次输出待发送数据的每一个比特；同时接收移位寄存器在 SCK 的控制下从 MOSI 管脚依次读入主机发送的每一个比特。每当 ISR.TXE 标志为 1 时，应及时向 SPI_DR 寄存器写入下一帧待发送的数据。每当数据接收完成时，ISR.RXNE 由硬件置 1，用户应及时从 SPI_DR 寄存器中读出接收到的数据。当检测到 NCS 信号被拉高后，结束本次通信。

从机在收到主机发送的 SCK 时钟前应准备好待发送的第一帧数据：向 ICR.FLUSH 写入 0 并将待发送的第一帧数据写入 SPI_DR 寄存器。

注意：主机在拉低 NCS 信号后，应留出足够的时间以便从机写入第一帧待发送的数据；从机也可以在主机拉低 NCS 信号之前就写入第一帧待发送的数据。

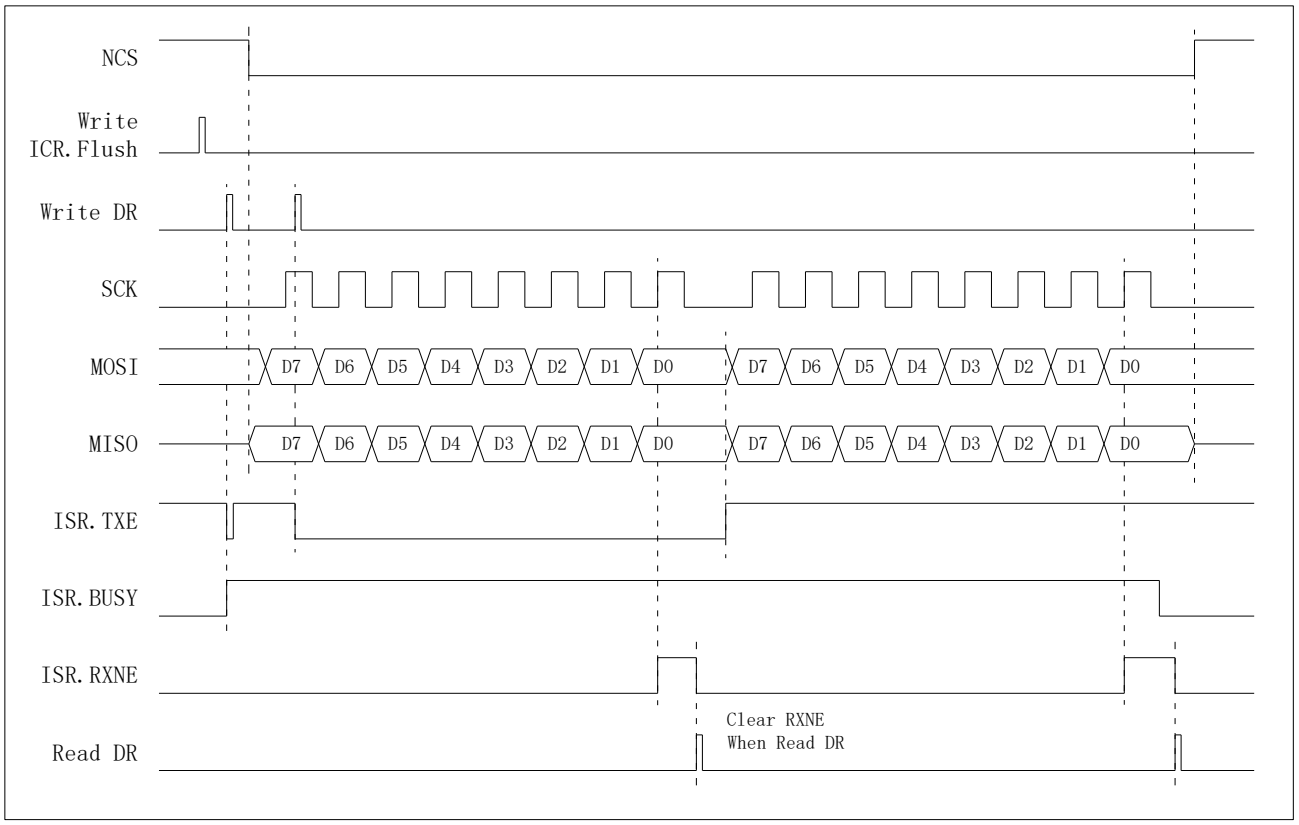

图 17-7 全双工从机收发示例

17.3.7 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工发送或单工接收通信。
主机使用 MOSI 信号线进行单工发送，使用 MISO 信号线进行单工接收；从机使用 MOSI 信号线进行单工接收，使用 MISO 信号线进行单工发送；未使用的信号线可用作其它功能。
设置 SPIx_CR1.MODE 位域为 0x1，SPI 工作于单工发送模式；设置 SPIx_CR1.MODE 位域为 0x2，SPI 工作于单工接收模式。
注意：SPI 工作于单工发送模式时，应忽略与接收相关的标志。

17.3.7.1 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。相关操作流程参见全双工模式。

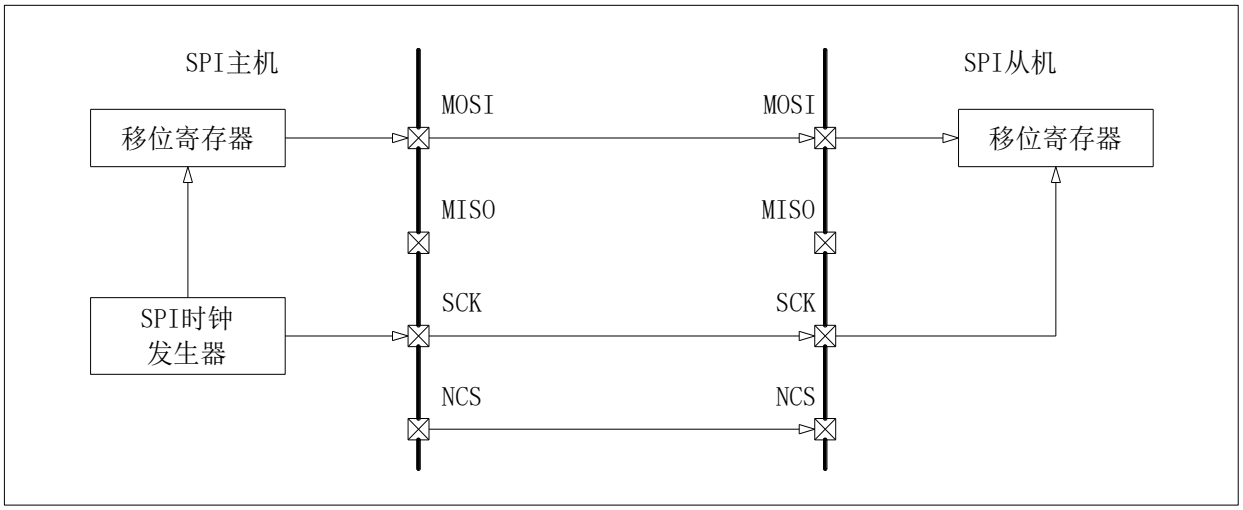

图 17-8 主机单发从机单收通信连接图

17.3.7.2 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。相关操作流程参见全双工模式。

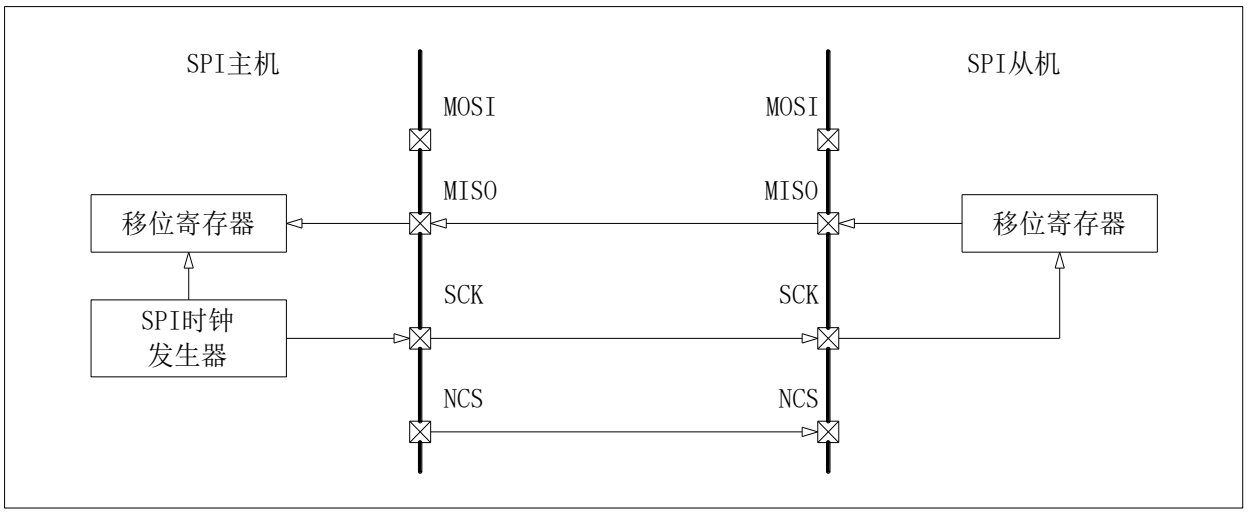

图 17-9 主机单发从机单收通信连接图

17.3.8 单线半双工模式

当设置 CR1.MODE 为 3 时，SPI 工作于单线半双工通信模式。主机和从机通过一条双向数据线进行连接，在主机提供的 SCK 时钟信号控制下进行数据传输。双向数据线的数据传输方向由 CR2.HDOE 位域控制；由于主机和从机的数据传输方向无法保证同步切换，建议在该数据线上串联一个电阻。当 SPI 作为从机并工作于单线半双工模式时，可与工作于同步模式的 UART 主机进行通信。

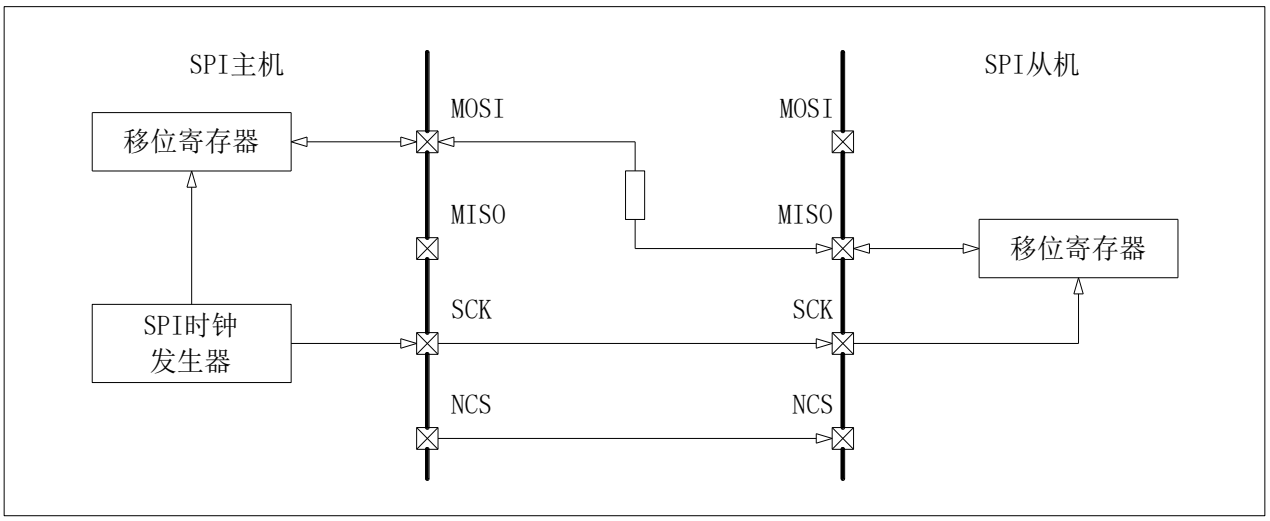

图 17-10 单线半双工通信连接图

- 主机发送

当设置 CR1.MSTR 为 1、CR2.HDOE 为 1 时，SPI 模块工作于主机发送模式。

设置 SSI.SSI 为 0，NCS 管脚输出低电平以选中从机。每当用户向 SPI_DR 寄存器写入待发送的数据后，发送移位寄存器在 SCK 的控制下从 MOSI 管脚依次输出待发送数据的每一个比特。待所有待发送的数据发送完成后，应设置 SSI.SSI 为 1 使 NCS 管脚输出高电平以结束本次通信。

- 主机接收

当设置 CR1.MSTR 为 1、CR2.HDOE 为 0 时，SPI 模块工作于主机接收模式。

设置 SSI.SSI 为 0，NCS 管脚输出低电平以选中从机。每当用户对 SPI_DR 寄存器进行一次写操作后，主机从 SCK 管脚输出时钟信号并从 MOSI 管脚读入一帧数据。每帧数据接收完成时，ISR.RXNE 由硬件置 1，用户可从 SPI_DR 寄存器中读出所接收到的数据。待所有待接收的数据接收完成后，应设置 SSI.SSI 为 1 使 NCS 管脚输出高电平以结束本次通信。

- 从机发送

当设置 CR1.MSTR 为 0、CR2.HDOE 为 1 时，SPI 模块工作于从机发送模式。

在 NCS 信号被拉低之前，从机需准备好待发送的第一帧数据：向 ICR.FLUSH 写入 0，将待发送的第一帧数据写入 SPI_DR 寄存器。当检测到 NCS 信号被拉低后，发送移位寄存器在 SCK 的控制下从 MISO 管脚依次输出待发送数据的每一个比特。每当 ISR.TXE 标志为 1 时，应及时向 SPI_DR 寄存器写入下一帧待发送的数据。当检测到 NCS 信号被拉高后，结束本次通信。

- 从机接收

当设置 CR1.MSTR 为 0、CR2.HDOE 为 0 时，SPI 模块工作于从机接收模式。

当检测到 NCS 信号被拉低后，接收移位寄存器在 SCK 的控制下从 MISO 管脚依次读入主机发送的每一个比特。每帧数据接收完成时，ISR.RXNE 由硬件置 1，用户可从 SPI_DR 寄存器中读出所接收到的数据。当检测到 NCS 信号被拉高后，结束本次通信。

17.3.9多机通信

在多机通信系统中，主机的 NCS 管脚应配置为输入模式（CR1.SSM=0），由 GPIO 负责输出从机选择信号，且主机的 NCS 输入由另一台主机的 GPIO 驱动。主机在尝试获取 SPI 总线控制权时，需结合 ISR.MODF 标志来检测总线是否发生冲突。多个从机的系统中，每台从机共享串行时钟和数据线，但是需要各自独立的从机选择信号，这就要求主机通过不同的 GPIO 向每个从机输出对应的从机选择信号。

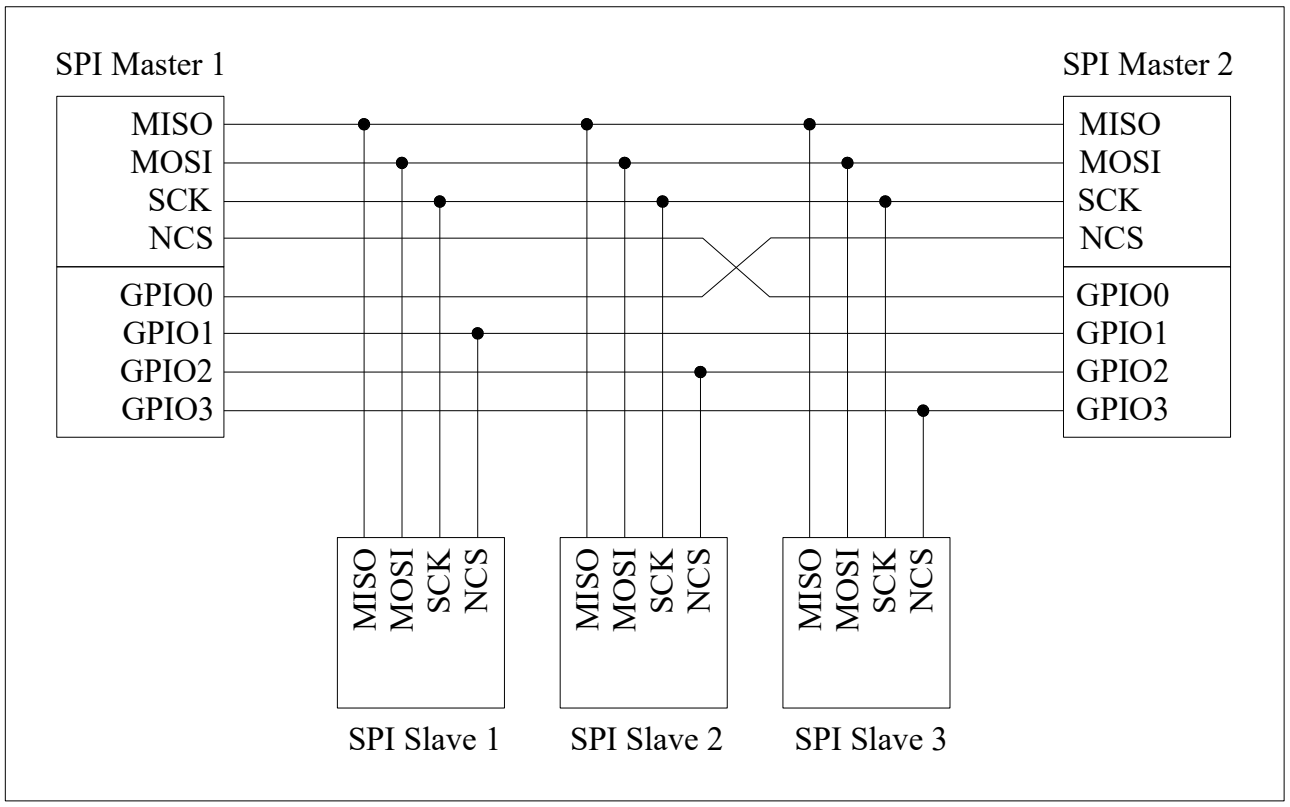

图 17-11 多机系统通信连接图

17.3.10 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（ISR.TXE）

当待发送数据从 SPIx_DR 寄存器转移到移位寄存器后，ISR.TXE 标志位会被硬件置 1，表示发送缓冲器已空，此时允许对 SPIx_DR 寄存器写入新的待发送数据。在对 SPIx_DR 寄存器写入数据的同时，ISR.TXE 标志位会被自动清 0。

- 接收缓冲非空标志位（ISR.RXNE）

当接收到的数据从移位寄存器转移到 SPIx_DR 寄存器后，ISR.RXNE 标志位会被硬件置 1，表示已接收到一帧数据，用户可从 SPIx_DR 寄存器中读取所接收到的数据。读 SPIx_DR 寄存器的同时将清除 ISR.RXNE 标志位；用户也可以通过写 ICR.RXNE 以清除 ISR.RXNE 标志。

- 总线忙标志位（ISR.BUSY）

ISR.BUSY 标志位，由硬件置 1 和清 0，用于表明当前的 SPI 通信状态。

主机模式下，当 SPIx_DR 或移位寄存器中尚有数据未发送完成时，该标志为 1；当 SPIx_DR 或移位寄存器中的数据均已发送完成时，该标志为 0。用户代码需查询到该标志为 0 时，才可拉高 NCS 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（ISR.SSLVL）

ISR.SSLVL 状态位，由硬件设置和清除，用于指示从机选择信号的电平状态。

ISR.SSLVL 为 0，从机选择信号为低电平，从机被选中；ISR.SSLVL 为 1，从机选择信号为高电平，从机未被选中。

- 从机选择输入上升沿标志（ISR.SSR）

ISR.SSR 标志位，用于指示从机选择信号是否出现了上升沿。

ISR.SSR 为 0，表示从机选择信号没有出现上升沿；ISR.SSR 为 1，表示从机选择信号出现了上升沿。

- 从机选择输入下降沿标志（ISR.SSF）

ISR.SSF 标志位，用于指示从机选择信号是否出现了下降沿。

ISR.SSF 为 0，表示从机选择信号没有出现下降沿；ISR.SSF 为 1，表示从机选择信号出现了下降沿。

17.3.11 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（ISR.MODF）

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 NCS 管脚被配置为输入时，如果 NCS 管脚输入为低电平，则会发生模式错误，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：
CR1.EN 被清零，SPI 被强制关闭；向 ICR.MODF 写入 0 即可清除 ISR.MODF 标志；之后需重新配置 SPI 控制器。

- 从机选择错误标志位（ISR.SSERR）

当 SPI 工作于从机模式，且正在进行数据传输时，NCS 输入信号被拉高，则会发生从机选择错误，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，用户应及时清除 ISR.SSERR 标志并在应用层进行出错处理。

- 接收缓冲上溢错误标志位（ISR.OV）

当 ISR.RXNE 标志为 1 且又收到新的一帧数据时，就会发生接收缓冲上溢错误，ISR.OV 标志位被硬件置位。发生该错误时，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之后，用户应及时清除 ISR.OV 标志并在应用层进行出错处理。

- 发送缓冲下溢错误标志位（ISR.UD）

当 SPI 工作于从机模式时，如果发送缓冲器空并收到来自主机的 SCK 时，就会发生发送缓冲下溢错误，ISR.UD 标志位会被硬件置位。该错误表示从机没能及时向 SPIx_DR 寄存器写入数据，错过了给主机发送数据的时机。
发生发送缓冲下溢错误之后，用户应及时清除 ISR.UD 标志并在应用层进行出错处理。

17.4 编程示例

17.4.1 主机查询方式发送数据

Step1. 设置 SYSCTRL_AHBEN.GPIOx 及 SYSCTRL_APBENx.SPIx 为 1，使能 SPI 管脚对应的 GPIO 时钟和 SPI 模块工作时钟；

Step2. 将 SCK、MOSI 和 NCS 信号配置到所选定的管脚并将该管脚配置为推挽复用输出模式；

Step3. 设置 CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；

Step4. 设置 CR1.MSTR 为 1，配置 SPI 为主机模式；

Step5. 设置 CR1.WIDTH 为 7，设置每帧数据的宽度 8 比特；

Step6. 设置 CR1.LSBF 为 0，设置先发送高有效位；

Step7. 根据需要配置 CR1.CPOL 及 CR1.CPHA，设置时钟极性及相位；

Step8. 设置 CR1.SSM 为 1，通过 SSI 寄存器控制 NCS 管脚输出电平；

Step9. 根据需要配置 CR1.BR，设置 SCK 时钟速率；

Step10. 设置 CR1.EN 为 1，使能 SPIx 模块；

Step11. 设置 SSI.SSI 为 0，NCS 管脚输出低电平以选中从机；

Step12. 查询等待 ISR.TXE 标志变为 1，发送缓冲器为空；

Step13. 将待发送的一帧数据写入 SPIx_DR 寄存器；

Step14. 如待发送的数据未完成，则跳转到 Step12 继续执行；

Step15. 查询等待 ISR.BUSY 标志变为 0，所有待发送数据均发送到 SPI 总线；

Step16. 设置 SSI.SSI 为 1，NCS 管脚输出高电平以结束通信。

17.4.2 主机查询方式接收数据

Step1. 设置 SYSCTRL_AHBEN.GPIOx 及 SYSCTRL_APBENx.SPIx 为 1，使能 SPI 管脚对应的 GPIO 时钟和 SPI 模块工作时钟；

Step2. 将 SCK 和 NCS 信号配置到所选定的管脚并将该管脚配置为推挽复用输出模式；
将 MISO 信号配置到所选定的管脚并将该管脚配置为浮空输入模式；

Step3. 设置 CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；

Step4. 设置 CR1.MSTR 为 1，配置 SPI 为主机模式；

Step5. 设置 CR1.WIDTH 为 7，设置每帧数据的宽度 8 比特；

Step6. 设置 CR1.LSBF 为 0，设置先发送高有效位；

Step7. 根据需要配置 CR1.CPOL 及 CR1.CPHA，设置时钟极性及相位；

Step8. 设置 CR1.SSM 为 1，通过 SSI 寄存器控制 NCS 管脚输出电平；

Step9. 根据需要配置 CR1.BR，设置 SCK 时钟速率；

Step10. 设置 CR1.EN 为 1，使能 SPIx 模块；

Step11. 设置 SSI.SSI 为 0，NCS 管脚输出低电平以选中从机；

Step12. 向 SPIx_DR 寄存器写入任意数据以启动一帧数据传输；

Step13. 查询等待 ISR.RXNE 标志变为 1，然后从 SPIx_DR 寄存器读出收到的数据并保存；

Step14. 如待接收的数据未完成，则跳转到 Step12 继续执行；

Step15. 设置 SSI.SSI 为 1，NCS 管脚输出高电平以结束通信。

17.4.3 从机查询方式接收数据

Step1. 设置 SYSCTRL_AHBEN.GPIOx 及 SYSCTRL_APBENx.SPIx 为 1，使能 SPI 管脚对应的 GPIO 时钟和 SPI 模块工作时钟；
Step2. 将 SCK、MOSI 和 NCS 信号配置到所选定的管脚并将该管脚配置为输入模式；
Step3. 设置 CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；
Step4. 设置 CR1.MSTR 为 0，配置 SPI 为从机模式；
Step5. 设置 CR1.WIDTH 为 7，设置每帧数据的宽度 8 比特；
Step6. 设置 CR1.LSBF 为 0，设置先发送高有效位；
Step7. 根据需要配置 CR1.CPOL 及 CR1.CPHA，设置时钟极性及相位；
Step8. 设置 CR1.EN 为 1，使能 SPIx 模块；
Step9. 查询等待 ISR.SSLVL 变为 0，主机拉低 NCS 信号以选中从机；
Step10. 查询等待 ISR.RXNE 变为 1，接收完成一帧数据；
Step11. 从 SPIx_DR 寄存器读出接收到的数据并保存；
Step12. 如待接收的数据未完成，则跳转到 Step10 继续执行；
Step13. 查询等待 ISR.SSLVL 变为 1，主机拉高 NCS 信号以结束通信。

17.4.4 从机查询方式发送数据

Step1. 设置 SYSCTRL_AHBEN.GPIOx 及 SYSCTRL_APBENx.SPIx 为 1，使能 SPI 管脚对应的 GPIO 时钟和 SPI 模块工作时钟；
Step2. 将 SCK 和 NCS 信号配置到所选定的管脚并将该管脚配置为输入模式；
将 MISO 信号配置到所选定的管脚并将该管脚配置为输出模式；
Step3. 设置 CR1.MODE 为 0x0，配置 SPI 为双向全双工通信模式；
Step4. 设置 CR1.MSTR 为 0，配置 SPI 为从机模式；
Step5. 设置 CR1.WIDTH 为 7，设置每帧数据的宽度 8 比特；
Step6. 设置 CR1.LSBF 为 0，设置先发送高有效位；
Step7. 根据需要配置 CR1.CPOL 及 CR1.CPHA，设置时钟极性及相位；
Step8. 设置 CR1.MISOHD 为 1，从机未被选中时 MISO 管脚为高阻态；
Step9. 设置 CR1.EN 为 1，使能 SPIx 模块；
Step10. 查询等待 ISR.SSLVL 变为 0，主机拉低 NCS 信号以选中从机；
Step11. 向 ICR.FLUSH 位域写入 0，清空发送缓冲区及移位寄存器；
Step12. 查询等待 ISR.TXE 变为 1，发送缓冲区已空；
Step13. 向 SPIx_DR 寄存器写入一帧待发送的数据；
Step14. 如待发送的数据未完成，则跳转到 Step12 继续执行；
Step15. 查询等待 ISR.SSLVL 变为 1，主机拉高 NCS 信号以结束通信。

注意：主机在拉低 NCS 信号后，应留出足够的时间以便从机写入第一帧待发送的数据。

17.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
SPIx_CR1	SPIx_Base + 0x00	控制寄存器1
SPIx_CR2	SPIx_Base + 0x08	控制寄存器2
SPIx_SSI	SPIx_Base + 0x0C	从机选择寄存器
SPIx_IER	SPIx_Base + 0x04	中断使能寄存器
SPIx_ISR	SPIx_Base + 0x10	中断标志寄存器
SPIx_ICR	SPIx_Base + 0x14	中断标志清除寄存器
SPIx_DR	SPIx_Base + 0x18	数据寄存器

SPI1_Base = 0x4001 3000

SPIx_CR1 控制寄存器 1

Address offset: 0x00 Reset value: 0x0000 1C04

位域	名称	权限	功能描述
31:19	RFU	-	保留位，请保持默认值
18	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
17:16	RFU	-	保留位，请保持默认值
15:14	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
13:10	WIDTH	RW	每帧数据的宽度为WIDTH+1，例如 0011：4bit 0100：5bit … 1110：15bit 1111：16bit
9	SSM	RW	从机选择配置（主机模式） 0：SPI_CS管脚为输入模式，管脚低电平可选中本机 1：SPI_CS管脚为输出模式，输出电平来自SSI 从机选择配置（从机模式） 0：SPI_CS管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
8	SMP	RW	主机模式延迟采样 0：传统采样方式，采样时刻由CPOL和CPHA的决定 1：延迟采样方式，采样时刻相比传统方式延迟半个SCK周期
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
6	EN	RW	SPI 模块使能控制 0：禁止 1：使能
5:3	BR	RW	主机模式SCK波特率配置 000：PCLK/2 001：PCLK/4 010：PCLK/8 011：PCLK/16 …… 110：PCLK/128

			111：保留
2	MSTR	RW	主从模式配置 0：从机模式 1：主机模式
1	CPOL	RW	串行时钟极性配置 0：待机时低电平 1：待机时高电平
0	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

SPIx_CR2 控制寄存器 2

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 注意：仅在单线半双工通信时有效

SPIx_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SPIx_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>SPIx_CS管脚输出低</td></tr>
<tr><td>1</td><td>SPIx_CS管脚输出高</td></tr>
</table></td></tr>
</table>

SPIx_DR 数据寄存器

Address offset: 0x18 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

SPIx_IER 中断使能寄存器

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	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：禁止 1：使能
2	SSF	RW	从机选择输入下降沿中断使能控制 0：禁止 1：使能
1	RXNE	RW	接收缓冲非空中断使能控制 0：禁止 1：使能
0	TXE	RW	发送缓冲空中断使能控制 0：禁止 1：使能

SPIx_ISR 中断标志寄存器

Address offset: 0x10 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
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
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

SPIx_ICR 中断标志清除寄存器

Address offset: 0x14　　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

18 I2C 接口（I2C）

18.1 概述

本芯片内部集成 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

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送端和接收端，发送端每发送 1 个字节数据，接收端必须回应 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)信号，从机地址及读写位，数据传输，停止信号(STOP)。如下图所示。

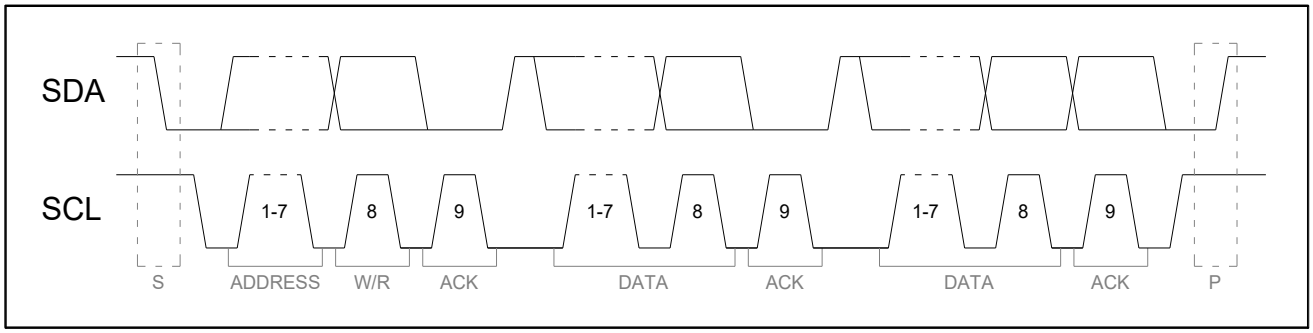

图 18-1 I2C 协议帧时序图

- 起始信号(START)

当总线处于空闲状态下（SCL 和 SDA 线同时为高）时，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，该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。

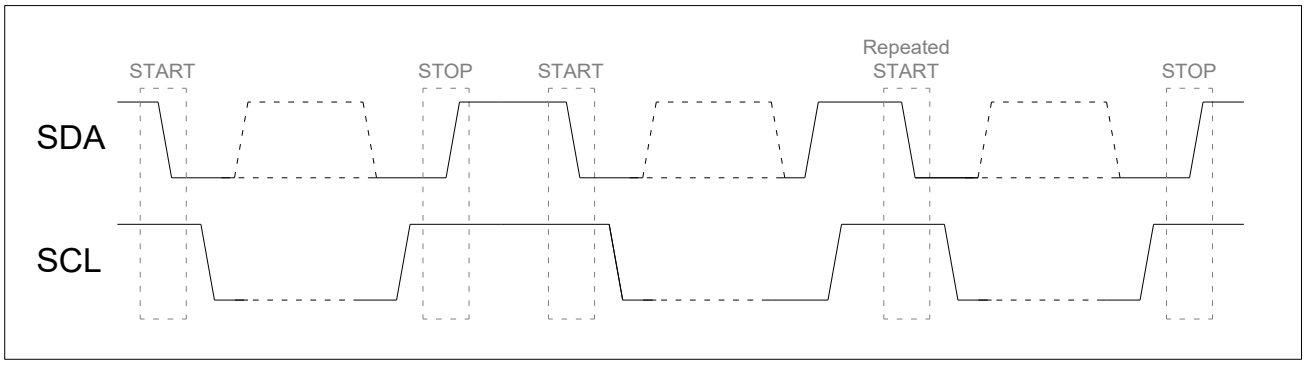

图 18-2 启始信号和停止信号时序图

- 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：7bit 从机地址+读写位。被寻址的从机在第 9 个 SCL 时钟周期内占用 SDA 总线，并将 SDA 设置为低电平作为 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，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。

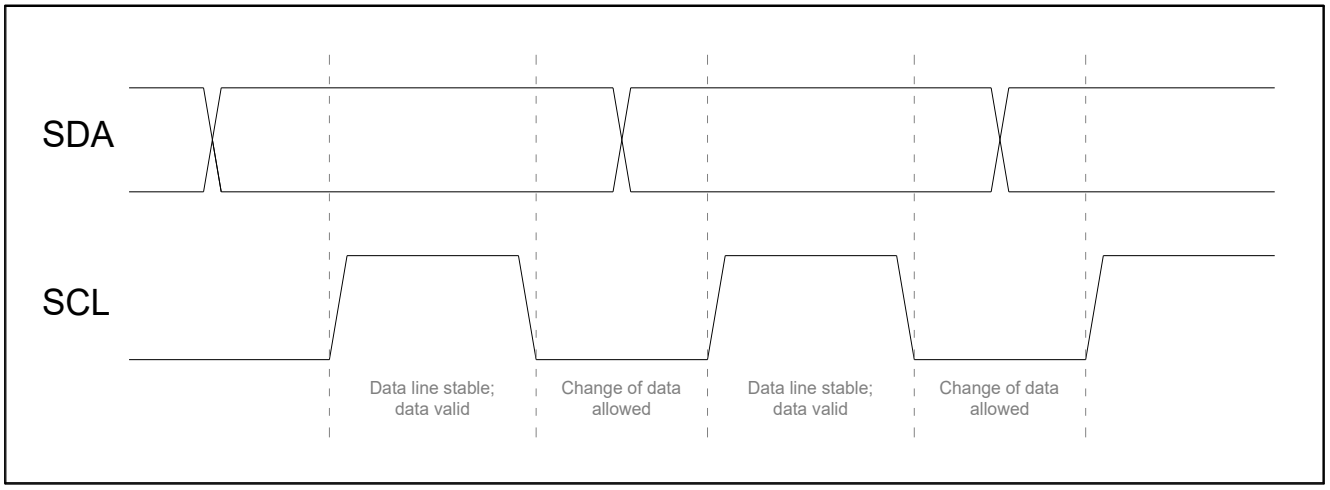

图 18-3 数据传输时序图

18.3.2传输应答

在总线上传输数据时，发送端每传输完成 1 个字节数据，在第 9 个 SCL 时钟周期发送端放弃对 SDA 的控制，接收端须在第 9 个 SCL 时钟周期回复应答信息：接收成功，发送 ACK 应答；接收异常发送 NACK 应答。

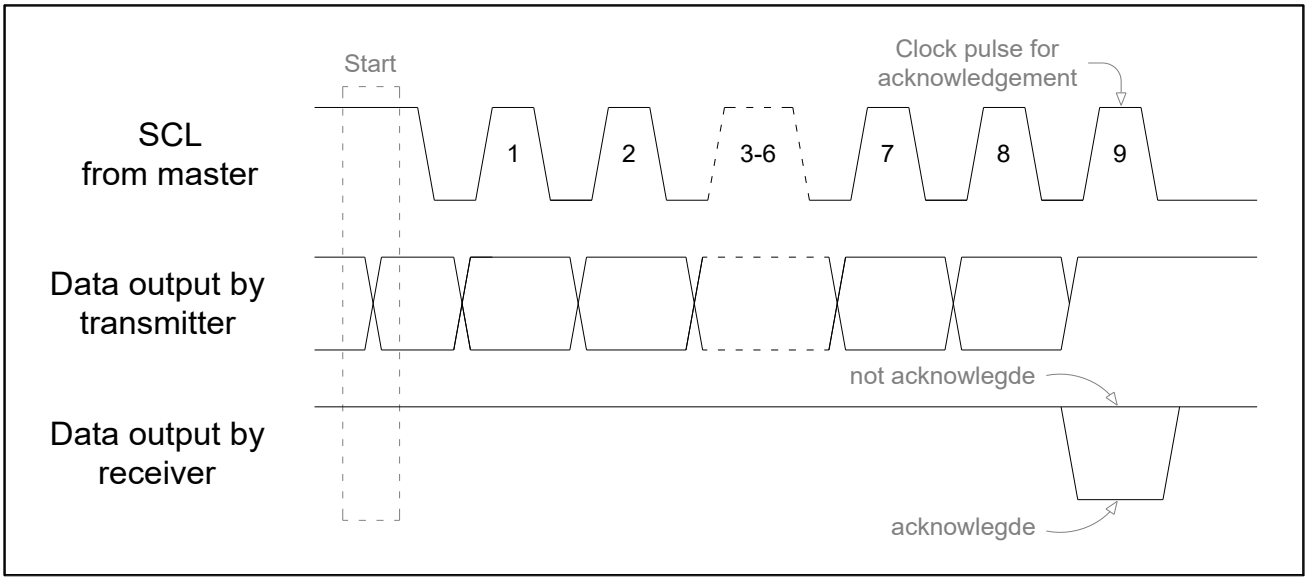

图 18-4 应答信号

18.3.3冲突检测与仲裁

I2C 总线上的仲裁分为两个部分：SCL 线上的同步和 SDA 线上的仲裁

- SCL 线上的同步（时钟同步）

  由于 I2C 总线具有线与的逻辑功能，SCL 线上只要有一个节点发送低电平，总线上就表现为低电平。当所有的节点都发送高电平时，总线才能表现为高电平。所以，时钟低电平的时间由时钟电平期最长的器件决定，而时钟的高电平时间由时钟高电平期最短的器件决定。由于 I2C 这种特性，当多个主机同时发送时钟信号时，在总线上表示的是统一的时钟信号。如果从机希望主机降低传送速度可以通过将 SCL 主动拉低延长其低电平时间来通知主机，当主机在准备下一次传送时发现 SCL 的电平被拉低时将进行等待，直到从机完成操作并释放 SCL 线的控制权。

- SDA 线上的仲裁

  SDA 线上的仲裁也具有线与的逻辑功能。主机在发送数据后，通过比较总线上的数据来决定是否退出竞争。丢失仲裁的主机立即切换到未被寻址的从机状态，以确保自身能被仲裁胜利的主机寻址到。仲裁失败的主机继续输出时钟脉冲（在 SCL 上），直到发送完当前的串行字节。通过这种原理可以保证 I2C 总线在多个主机企图控制总线时保证数据的不丢失。

18.4 功能描述

本节将详细描述 I2C 接口的相关功能。

18.4.1 功能框图

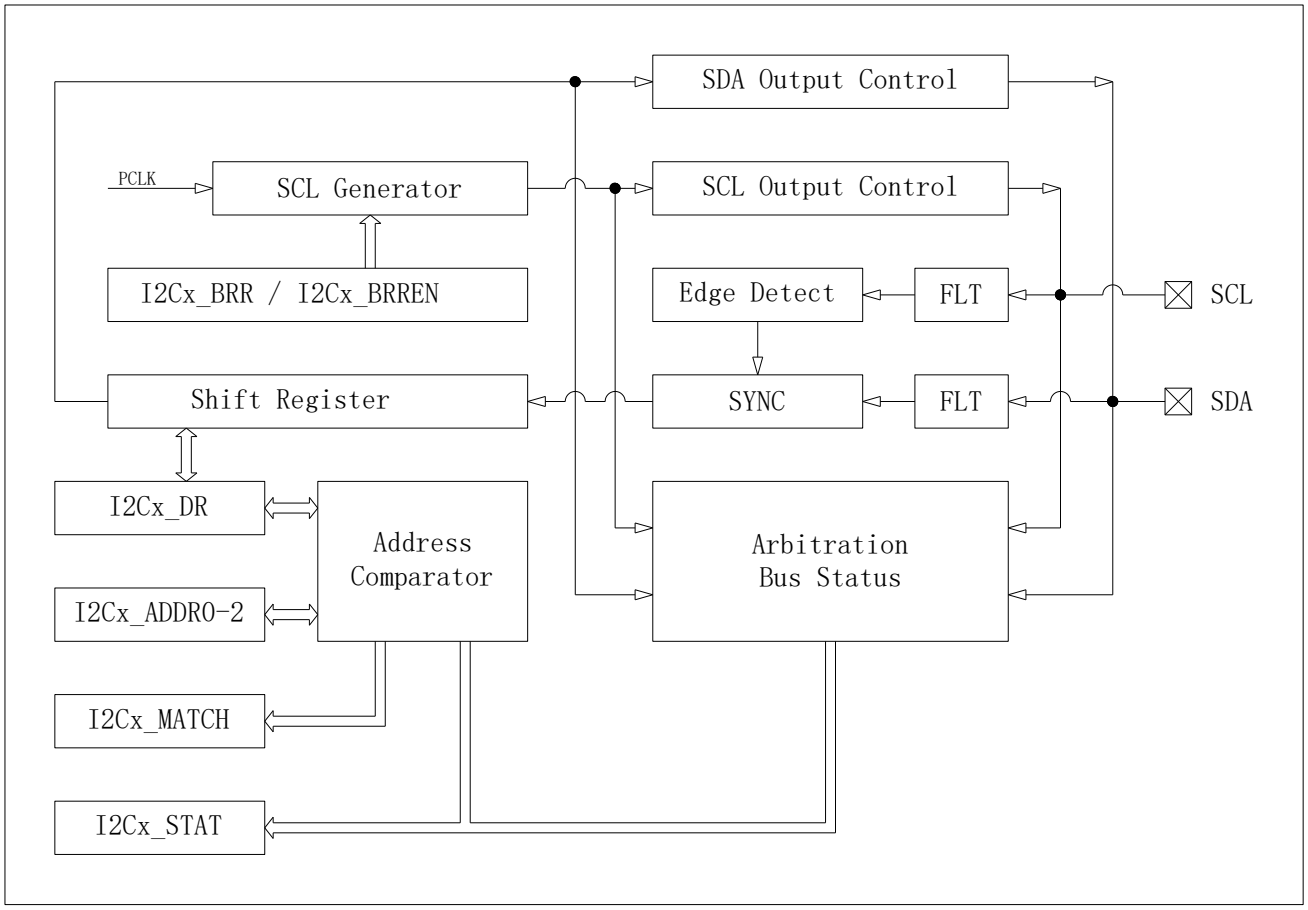

图 18-5 I2C 功能框图

18.4.2 串行时钟发生器

串行时钟发生器采用 PCLK 作为输入时钟，可生各种频率的 SCL 时钟，其计算公式如下所示：

$$f_{SCL} = \frac{f_{PCLK}}{8 \times (BRR + 1)} \ (BRR \geq 1)$$

其中，f_{SCL} 代表 SCL 时钟信号的频率，f_{PCLK} 代表 PCLK 时钟信号的频率，BRR 的值来自寄存器 I2Cx_BRR。

18.4.3 输入滤波器

输入滤波器可对 SDA 和 SCL 输入信号进行数字滤波处理，可有效消除总线上的干扰信号。通过 I2Cx_CR.FLT 控制位可将滤波器设为简单滤波模式或高级滤波模式。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。

作为主机时，如果 BRR 的值小于或等于 9，则应设置 I2Cx_CR.FLT 为 1；如果 BRR 的值大于 9，则应设置 I2Cx_CR.FLT 为 0。

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 I2Cx_CR.FLT 为 1；如果 PCLK 与 SCL 的频率比值大于 40，则应设置 I2Cx_CR.FLT 为 0。

18.4.4 地址比较器

I2C 控制器支持 3 个可编程的从机地址，可通过 I2Cx_ADDR0-2 寄存器进行配置。地址比较器将接收到的地址和 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，设置 I2Cx_CR.SI 为 1 并产生中断请求。应用程序可查询 I2Cx_MATCH 寄存器以获取匹配成功的地址序号。

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 I2Cx_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 I2Cx_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 数据传输的接收端必须在每个字节的第 9 个 SCL 时钟周期给发送端进行 ACK 或者 NACK 应答，发送端通过该应答位来获取接收端当前状态：回应 ACK 应答，则表明接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 一般表示接收端已不再接收任何数据。

接收端发送 ACK 还是 NACK，由接收端的 I2Cx_CR.AA 位域控制。当设置 I2Cx_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 I2Cx_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（接收端）在最后一个字节数据接收完成后回应 NACK 应答给从机（发送端），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 I2Cx_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（发送端），则表示从机（接收端）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 中断

I2C 控制寄存器 I2Cx_CR 的 SI 位域为中断标志位。当 I2Cx_STAT 寄存器发生改变（变成 0xF8 除外）时，I2Cx_CR.SI 标志位就会被置位，同时产生中断请求。用户应在中断服务程序中查询 I2Cx_STAT 寄存器的值以确定中断产生原因。设置 I2Cx_CR.SI 为 0 即可清除该标志位。

18.4.8 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。

18.4.8.1 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 时钟由主机产生，因此需要根据传输波特率设置 I2Cx_BRR、I2Cx_BRREN。主机设置 I2Cx_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 I2Cx_STAT 变为 0x08，中断标志位 I2Cx_CR.SI 被置 1。

主机发送完 START 起始信号后，需要软件设置 I2Cx_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 I2Cx_DR；清零 I2Cx_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 I2Cx_STAT 变为 0x18，中断标志位 I2Cx_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，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 I2Cx_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 I2Cx_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 I2Cx_STAT=0x38）。

主机发送模式下状态同步图如下图所示。

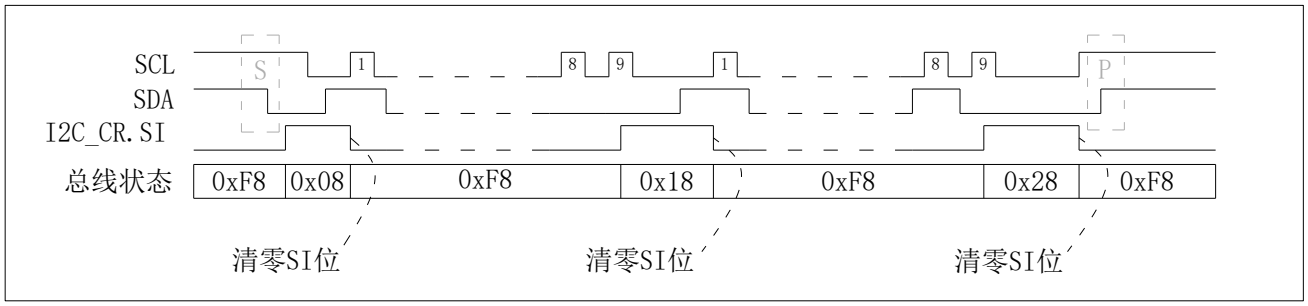

图 18-6 主机发送模式状态同步图

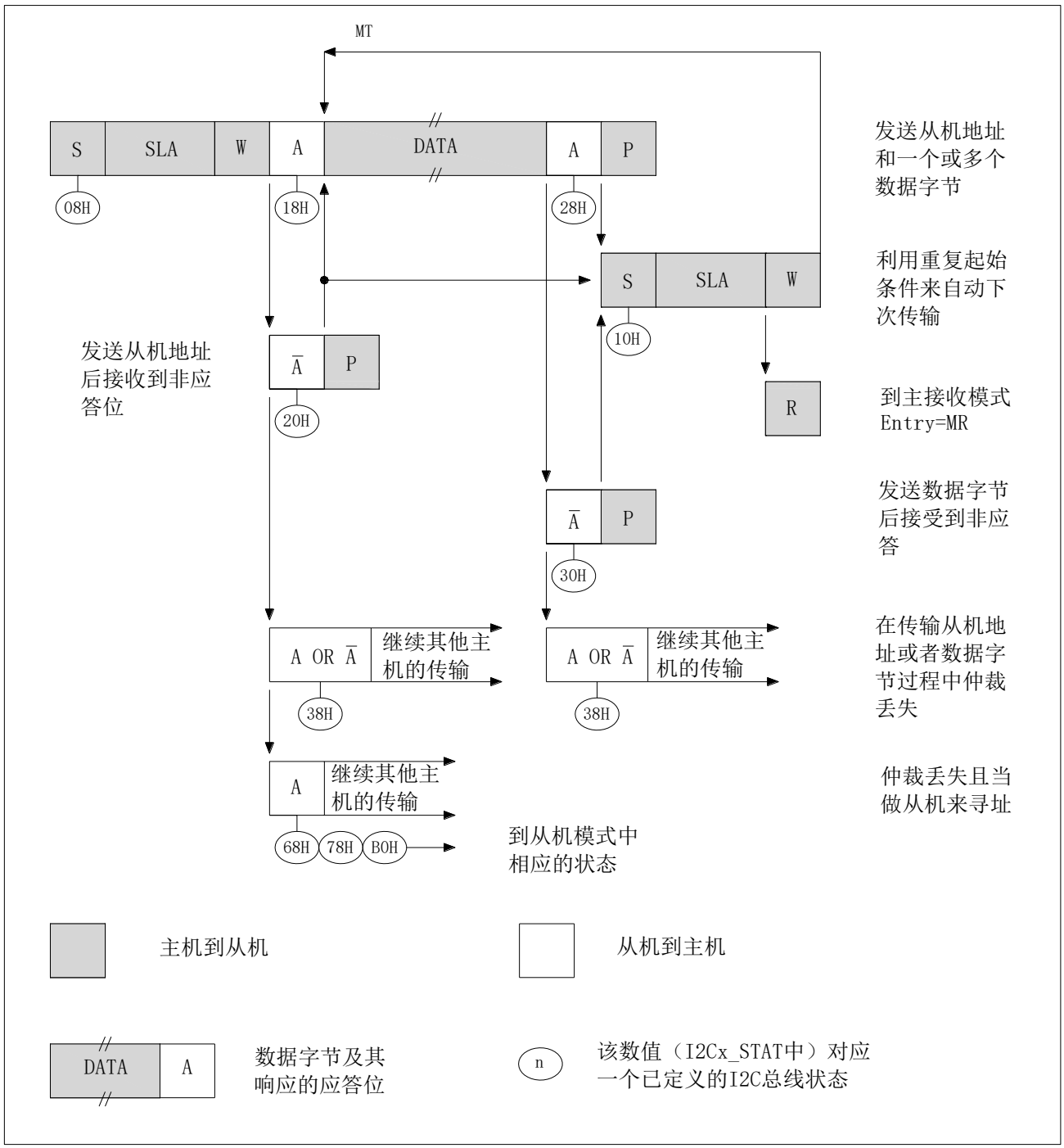

图 18-7 主机发送模式流程和寄存器状态

18.4.8.2 主机接收模式

主机接收模式用于接收从机发送的数据，主机每接收到 1 字节数据后应回应 ACK 应答信号。SCL 时钟由主机产生，因此需要根据传输波特率设置 I2Cx_BRR、I2Cx_BRREN。

主机设置 I2Cx_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 I2Cx_STAT 变为 0x08，中断标志位 I2Cx_CR.SI 被置 1。

主机发送完 START 起始信号后，需要软件设置 I2Cx_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 I2Cx_DR；清零 I2Cx_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 I2Cx_STAT 变为 0x40，中断标志位 I2Cx_CR.SI 被置 1。

此后，主机设置 I2Cx_CR.AA 为 1，并清零 I2Cx_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。在接收最后一个字节前需要将 I2Cx_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，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 I2Cx_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 I2Cx_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 I2Cx_STAT=0x38）。

主机接收模式下状态同步图如下图所示。

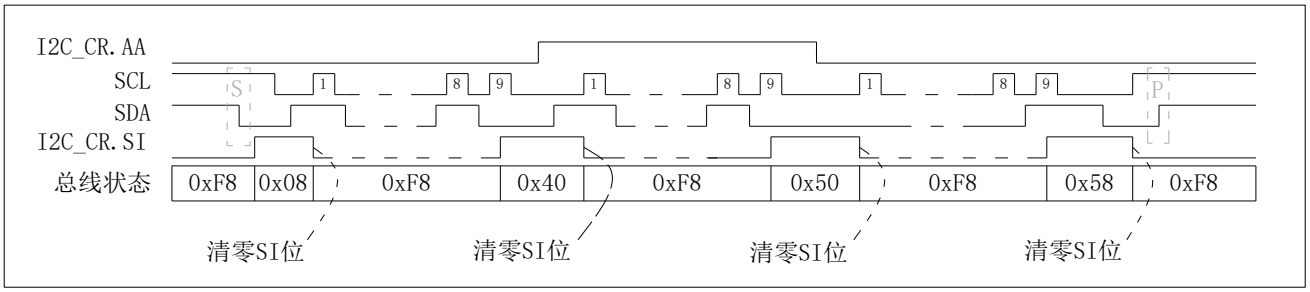

图 18-8 主机接收模式状态同步图

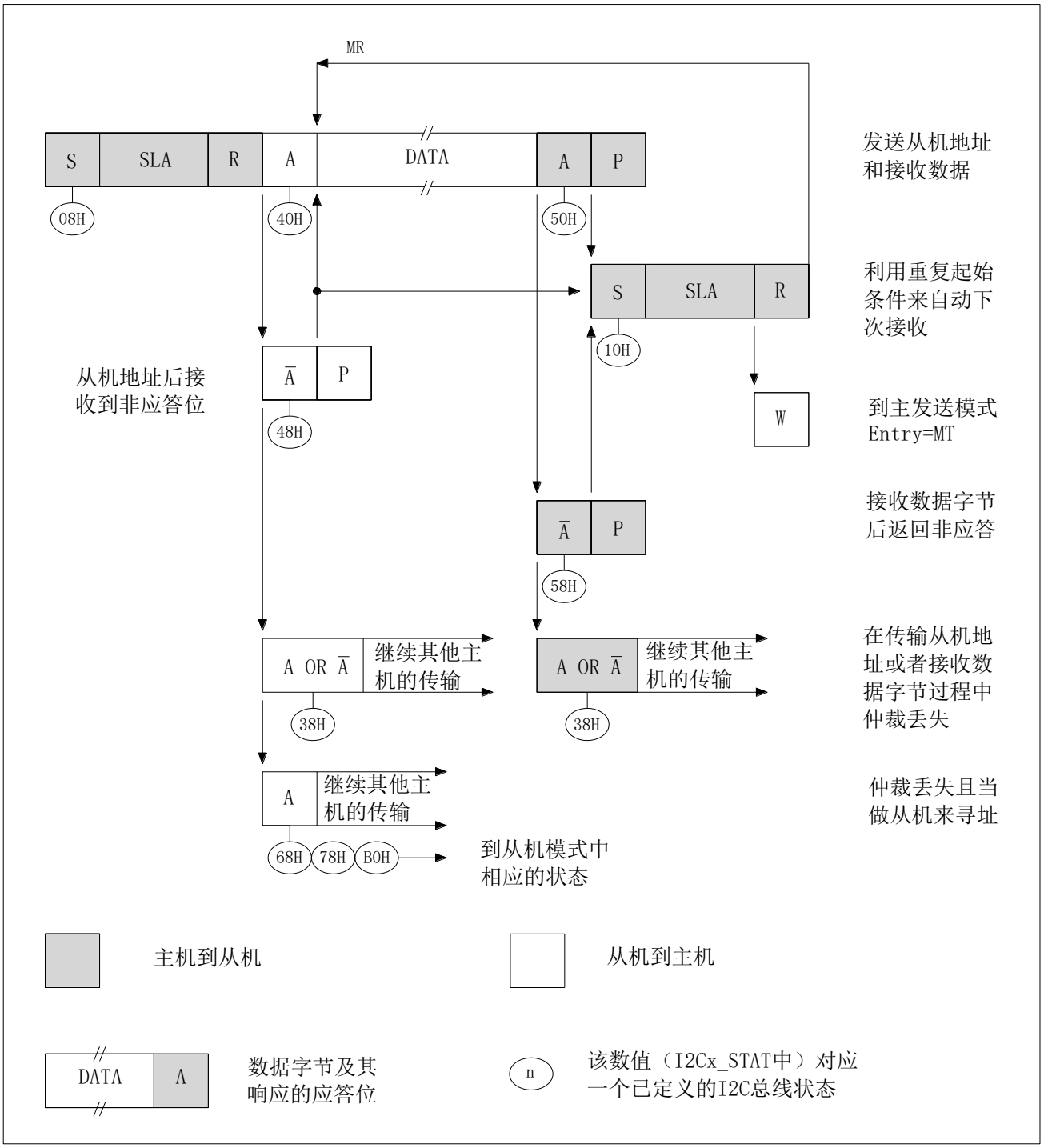

图 18-9 主机接收状态图

18.4.8.3 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 I2Cx_ADDR0-2 任意 1 个中；设置 I2Cx_CR.AA 为 1 以响应主机的寻址。

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 I2Cx_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 I2Cx_STAT 变为 0x60，中断标志位 I2Cx_CR.SI 同时被置 1。此时必须清除 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_STAT=0x68），则会进入已寻址从机接收模式。

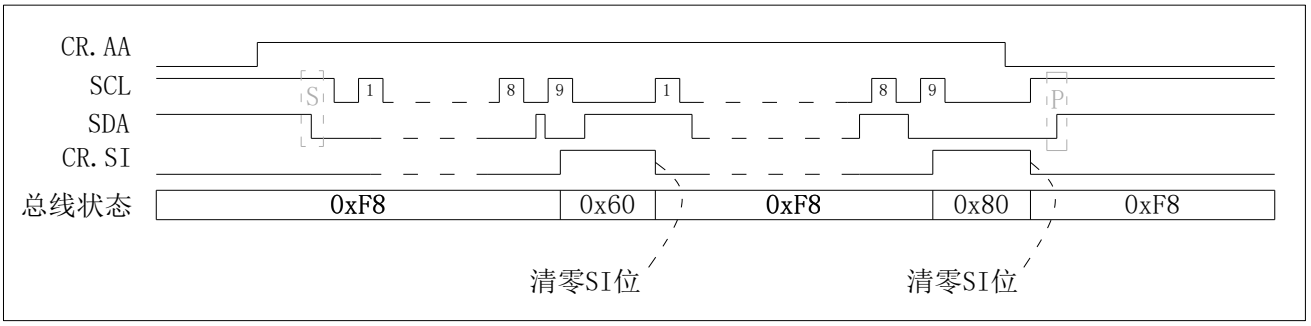

图 18-10 从机接收模式状态同步图

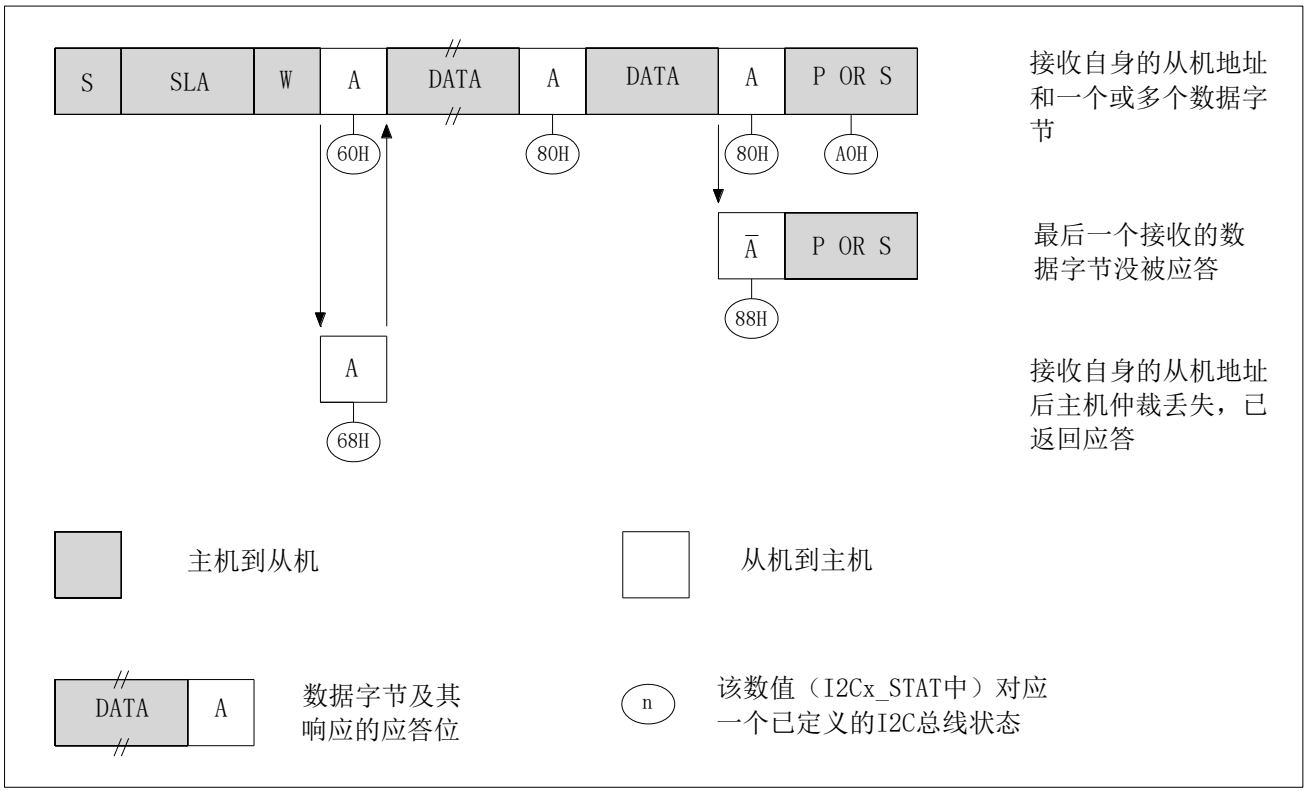

图 18-11 从机接收状态图

18.4.8.4 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 I2Cx_ADDR0-2 任意 1 个中；设置 I2Cx_CR.AA 为 1 以响应主机的寻址。

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 I2Cx_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 I2Cx_STAT 变为 0xA8，中断标志位 I2Cx_CR.SI 同时被置 1。此时应及时将待发送的数据写入 I2Cx_DR 寄存器，并清除 I2Cx_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。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 I2Cx_STAT=0xB0），则会进入已寻址从机发送模式。

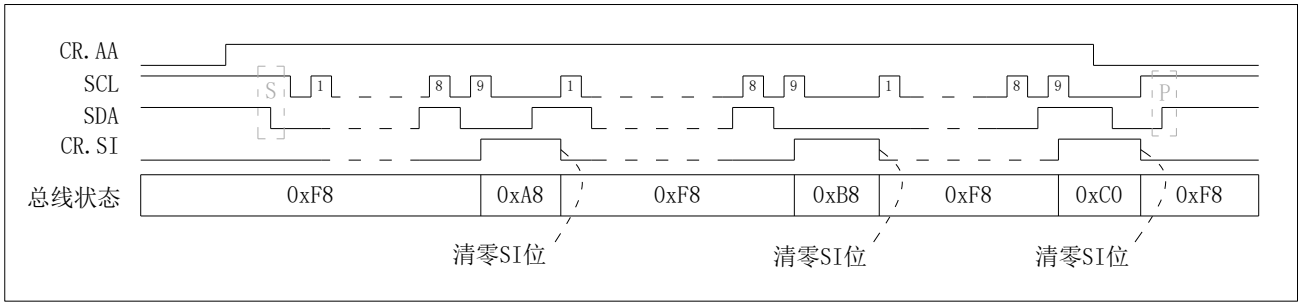

图 18-12 从机发送模式状态同步图

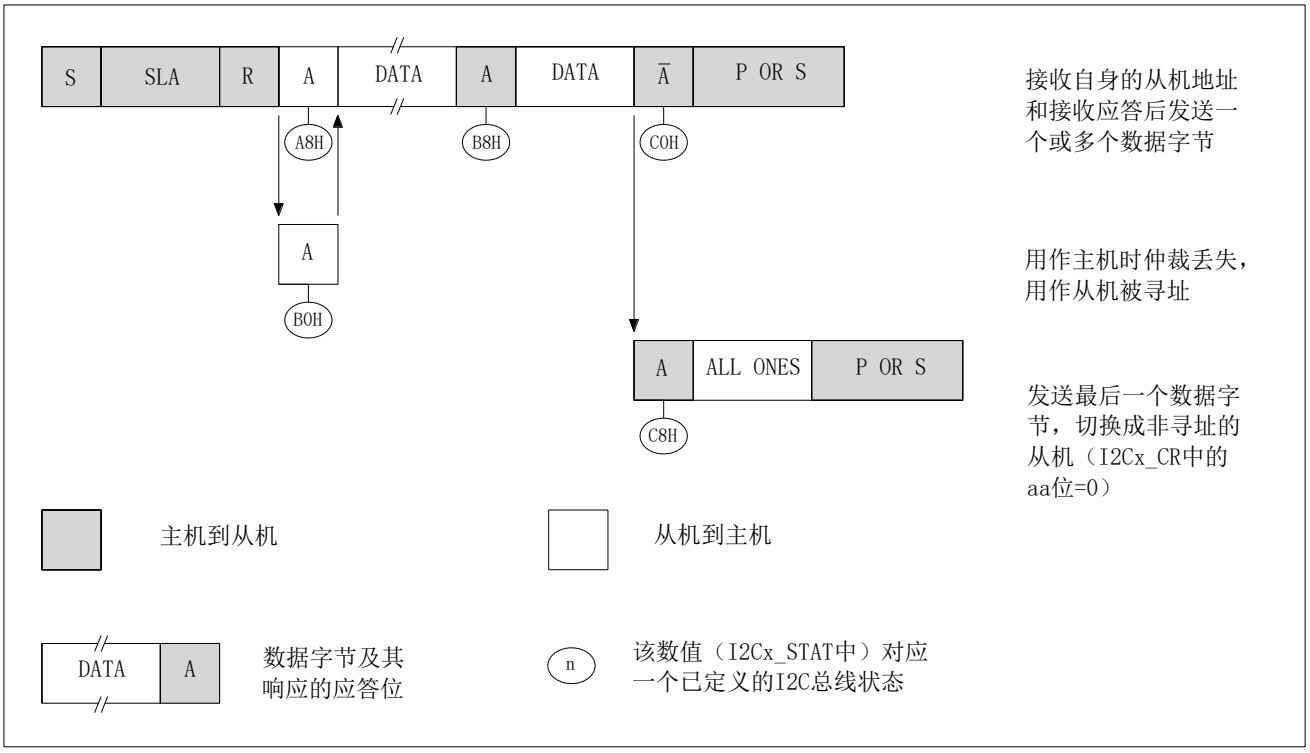

图 18-13 从机发送状态图

18.4.8.5 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 I2Cx_CR.AA 为 1 以及 I2Cx_ADDR0.GC 为 1 以响应主机的广播寻址和广播数据。

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 I2Cx_STAT 变为 0x70，中断标志位 I2Cx_CR.SI 同时被置 1。此时必须及时清除 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_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 I2Cx_STAT=0x78），则会进入已寻址广播接收模式。

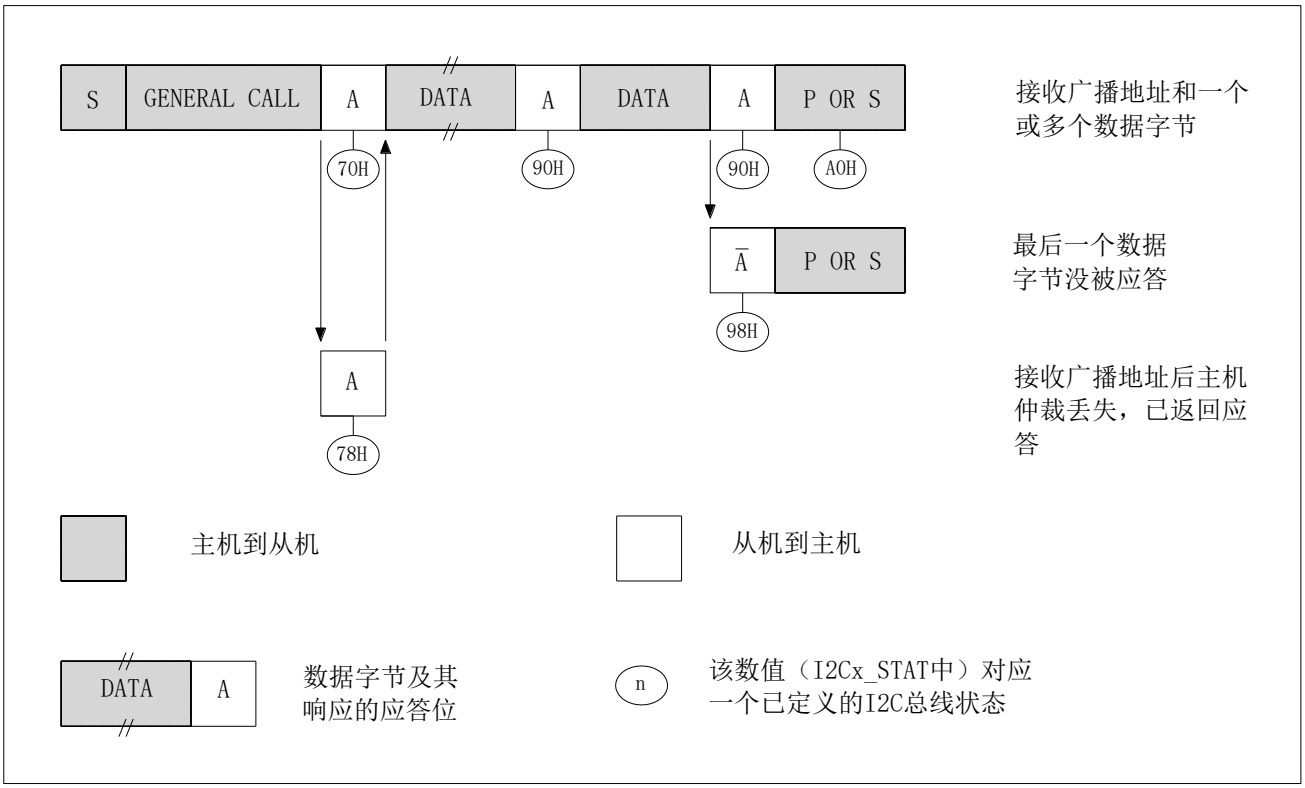

图 18-14 从机广播接收状态图

18.4.9状态码

I2C 总线状态通过 I2C 状态寄存器 I2Cx_STAT 来标识，其详细说明如下表所示。

表 18-1 状态码汇总表

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 I2Cx_DR 中的数据，已接收ACK
	30H	已发送 I2Cx_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
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，I2Cx_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

I2Cx_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，I2Cx_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 I2Cx_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：

- 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。
- 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 I2Cx_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。

18.5 编程示例

18.5.1 主机发送示例

Step1. 按 GPIO 章节管脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的管脚，并配置 SCL、SDA 管脚为开漏输出模式；

Step2. 设置 SYSCTRL_APBEN1.I2Cx 为 1，使能 I2Cx 模块时钟；

Step3. 配置 I2Cx_BRR，使 SCL 的时钟速率符合应用需求；

Step4. 设置 I2Cx_BRREN 为 1，使能 SCL 时钟发生器；

Step5. 设置 I2Cx_CR.EN 为 1，使能 I2C 模块；

Step6. 设置 I2Cx_CR.STA 为 1，总线尝试发送 START 信号；

Step7. 等待 I2Cx_CR.SI 变为 1，START 信号已发送到总线上；

Step8. 查询 I2Cx_STAT，如果该寄存器值为 0x08 或 0x10 执行下一步，否则进行出错处理；

Step9. 设置 I2Cx_CR.STA 为 0，向 I2Cx_DR 中写入 SLA+W，设置 I2Cx_CR.SI 为 0，发送 SLA+W；

Step10. 等待 I2Cx_CR.SI 变为 1，SLA+W 已发送到总线上；

Step11. 查询 I2Cx_STAT，如果该寄存器值为 0x18 继续执行下一步，否则进行出错处理；

Step12. 向 I2Cx_DR 写入待发送的数据，设置 I2Cx_CR.SI 为 0，发送数据；

Step13. 等待 I2Cx_CR.SI 变为 1，数据已发送到总线上；

Step14. 查询 I2Cx_STAT，如果该寄存器值为 0x28 继续执行下一步，否则进行出错处理；

Step15. 如待发送的数据未完成，则跳转到 Step12 继续执行；

Step16. 设置 I2Cx_CR.STO 为 1，设置 I2Cx_CR.SI 为 0，总线尝试发送 STOP 信号；

Step17. 等待 I2Cx_CR.STO 变为 0，STOP 信号已发送到总线上，结束本次数据传输。

18.5.2 主机接收示例

Step1. 按 GPIO 章节管脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的管脚，并配置 SCL、SDA 管脚为开漏输出模式；

Step2. 设置 SYSCTRL_APBEN1.I2Cx 为 1，使能 I2Cx 模块时钟；

Step3. 配置 I2Cx_BRR，使 SCL 的时钟速率符合应用需求；

Step4. 设置 I2Cx_BRREN 为 1，使能 SCL 时钟发生器；

Step5. 设置 I2Cx_CR.EN 为 1，使能 I2C 模块；

Step6. 设置 I2Cx_CR.STA 为 1，总线尝试发送 START 信号；

Step7. 等待 I2Cx_CR.SI 变为 1，START 信号已发送到总线上；

Step8. 查询 I2Cx_STAT，如果该寄存器值为 0x08 或 0x10 执行下一步，否则进行出错处理；

Step9. 设置 I2Cx_CR.STA 为 0，向 I2Cx_DR 写入 SLA+R，设置 I2Cx_CR.SI 为 0，发送 SLA+R；

Step10. 等待 I2Cx_CR.SI 变为 1，SLA+R 已发送到总线上；

Step11. 查询 I2Cx_STAT，如果寄存器值为 0x40 继续执行下一步，否则进行出错处理；

Step12. 设置 I2Cx_CR.AA 为 1，使能应答标志；

Step13. 设置 I2Cx_CR.SI 为 0，主机发出 SCL，从机发出 SDA；

Step14. 等待 I2Cx_CR.SI 变为 1，从 I2Cx_DR 读取已接收到的数据；

Step15. 查询 I2Cx_STAT，如果该寄存器值为 0x50 或 0x58 执行下一步，否则进行出错处理；

Step16. 如果待接收的数据只差最后一个字节，设置 I2Cx_CR.AA 为 0，使能非应答标志；

Step17. 如待接收的数据未完成，则跳转到 Step13 继续执行；

Step18. 设置 I2Cx_CR.STO 为 1，设置 I2Cx_CR.SI 为 0，总线尝试发送 STOP 信号；

Step19. 等待 I2Cx_CR.STO 变为 0，STOP 信号已发送到总线上，结束本次数据传输。

18.5.3 从机接收示例

Step1. 按 GPIO 章节管脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的管脚，并配置 SCL、SDA 管脚为开漏输出模式；
Step2. 设置 SYSCTRL_APBEN1.I2Cx 为 1，使能 I2Cx 模块时钟；
Step3. 设置 I2Cx_CR.EN 为 1，使能 I2C 模块；
Step4. 配置 I2Cx_ADDR0 为从机地址；
Step5. 设置 I2Cx_CR.AA 为 1，使能应答标志；
Step6. 等待 I2Cx_CR.SI 变为 1，被 SLA+W 寻址；
Step7. 查询 I2Cx_STAT，如果该寄存器值为 0x60 继续执行下一步，否则进行出错处理；
Step8. 设置 I2Cx_CR.SI 为 0，等待主机发送数据并回应 ACK 信号；
Step9. 等待 I2Cx_CR.SI 变为 1，从 I2Cx_DR 中读取已接收到的数据；
Step10. 查询 I2Cx_STAT，如果该寄存器值为 0x80 继续执行下一步，否则进行出错处理；
Step11. 如待接收的数据未完成，则跳转到 Step9 继续执行；
Step12. 设置 I2Cx_CR.AA 为 0，设置 I2Cx_CR.SI 为 0。

18.5.4 从机发送示例

Step1. 按 GPIO 章节管脚数字复用功能的相关描述，将 SCL、SDA 映射到需要的管脚，并配置 SCL、SDA 管脚为开漏输出模式；
Step2. 设置 SYSCTRL_APBEN1.I2Cx 为 1，使能 I2Cx 模块时钟；
Step3. 设置 I2Cx_CR.EN 为 1，使能 I2C 模块；
Step4. 配置 I2Cx_ADDR0 为从机地址；
Step5. 设置 I2Cx_CR.AA 为 1，使能应答标志；
Step6. 等待 I2Cx_CR.SI 变为 1，被 SLA+R 寻址；
Step7. 查询 I2Cx_STAT，如果该寄存器的值为 0xA8 继续执行下一步，否则进行出错处理；
Step8. 向 I2Cx_DR 写入待发送的数据，设置 I2Cx_CR.SI 为 0，准备发送数据；
Step9. 等待 I2Cx_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；
Step10. 查询 I2Cx_STAT，如果该寄存器的值为 0xB8 时继续执行下一步，否则进行出错处理；
Step11. 如待发送的数据未完成，则跳转到 Step8 继续执行；
Step12. 设置 I2Cx_CR.AA 为 0，设置 I2Cx_CR.SI 为 0。

18.6 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
I2Cx_BRREN	I2Cx_Base + 0x00	波特率生成器使能寄存器
I2Cx_BRR	I2Cx_Base + 0x04	波特率生成器配置寄存器
I2Cx_CR	I2Cx_Base + 0x08	控制寄存器
I2Cx_DR	I2Cx_Base + 0x0C	数据寄存器
I2Cx_ADDR0	I2Cx_Base + 0x10	从机地址0寄存器
I2Cx_STAT	I2Cx_Base + 0x14	状态寄存器
I2Cx_ADDR1	I2Cx_Base + 0x20	从机地址1寄存器
I2Cx_ADDR2	I2Cx_Base + 0x24	从机地址2寄存器
I2Cx_MATCH	I2Cx_Base + 0x28	从机地址匹配标志寄存器

I2C1_Base = 0x4000 5400

I2Cx_BRREN 波特率生成器使能寄存器

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 波特率生成器使能控制 0：禁止 1：使能

I2Cx_BRR 波特率生成器配置寄存器

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 ≥ 1

I2Cx_CR 控制寄存器

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	EN	RW	I2C模块使能控制 0：禁止 1：使能
5	STA	RW	I2C总线状态控制 W1：向总线发送START或Repeated START起始信号 注意：完成START信号发送后，应将STA清零
4	STO	RW	I2C总线状态控制 R0：发送STOP流程已经完成 R1：发送STOP流程尚未完成 W1：向总线发送STOP停止信号 注意：用户置位STO后，应等待STO变为0
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
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
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：详见本章输入滤波器小节。

I2Cx_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

I2Cx_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【状态码】小节

I2Cx_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
0	GC	RW	广播地址应答使能 0：禁止 1：使能

I2Cx_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
0	RFU	-	保留位，请保持默认值

I2Cx_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
0	RFU	-	保留位，请保持默认值

I2Cx_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三种情况下会清零：

- 模块复位时
- START/STOP 发送时

19 红外调制器（IRMOD）

19.1 概述

本芯片内部集成红外调制器(IRMOD)，借助 GTIM 的输出比较通道和 UART 的 TXD 信号，可输出多种红外调制波形。

19.2 主要特性

- 载波信号占空比可任意调节
- 多种调制信号：GTIM1_CH1 / GTIM1_CH2 / UARTx_TXD / BTIMx_TOGP
- 两种调制方式：逻辑与，逻辑或

19.3 功能描述

本芯片具有红外调制接口，调制后的信号驱动红外 LED 即可实现红外遥控功能。可选择 4 种信号（GTIM1_CH1/CH2，BTIM2_TOGP，BTIM3_TOGP）作为载波信号，可选择 5 种信号（UART1_TXD，UART2_TXD，BTIM2_TOGP，BTIM3_TOGP，CR.IRSW）作为调制信号。载波信号与调制信号可采用“逻辑或”、“逻辑与”进行调制，调制后的信号可以从 IR_OUT 管脚输出以驱动红外 LED。

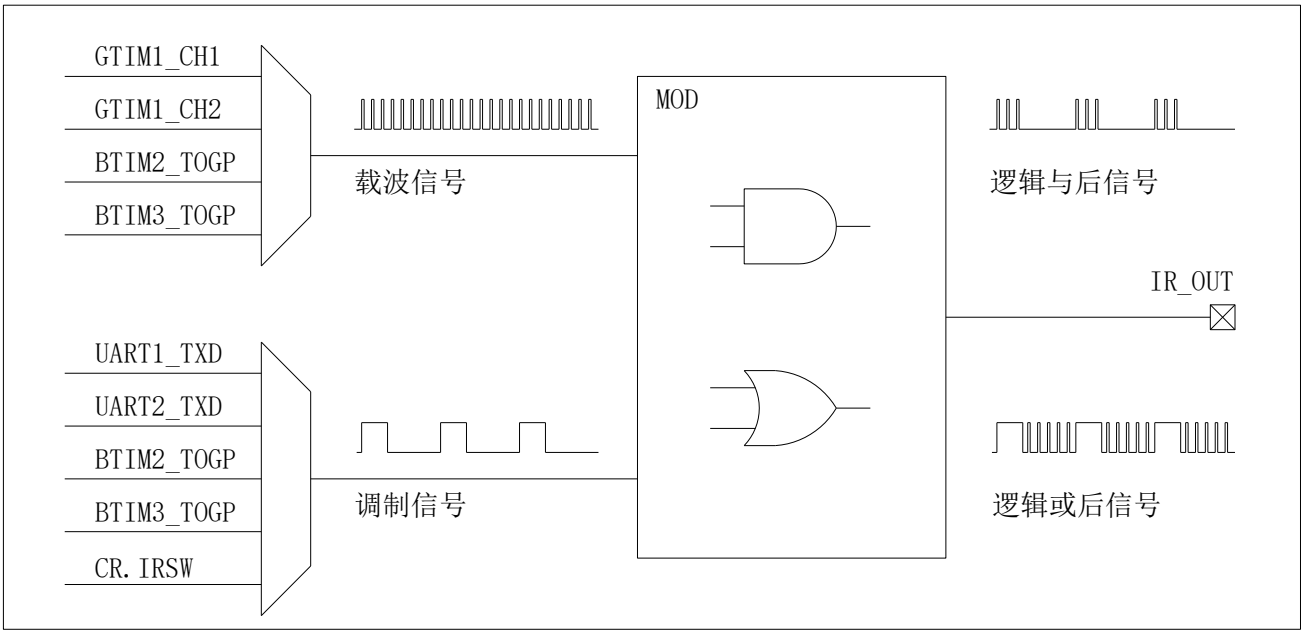

图 19-1 IRMOD 功能框图

19.4 编程示例

- 输出 UART1_TXD 与 GTIM1_CH1“逻辑或”的波形

  Step1. 设置 SYSCTRL_AHBEN.GPIOB 为 1，使能 GPIOB 时钟；

  Step2. 设置 GPIOB_ANALOG.PIN3 为 0，设置 PB03 为数字管脚；

  Step3. 设置 GPIOB_DIR.PIN3 为 0，设置 PB03 为输出模式；

  Step4. 设置 GPIOB_AFRL.AFR3 为 7，配置 PB03 的功能为 IR_OUT；

  Step5. 设置 IRMOD_CR 为 0x09，配置调制方式为 UART2_TXD | GTIM1_CH1；

  Step6. 设置 SYSCTRL_APBEN1.GTIM1 为 1，使能 GTIM1 时钟；

  Step7. 按 GTIM 章节描述，配置 GTIM1_CH1 输出 38kHz 占空比为 70%的方波；

  Step8. 按 UART 章节描述，配置 UART1 为通信参数为 4800-8-N-1；

  Step9. 向 UART1_ICR.TC 写入 0，清除发送完成标志；

  Step10. 将待发送的数据写入 UART1_TDR 寄存器，IR_OUT 管脚输出调制波形；

  Step11. 查询等待 UART1_ISR.TC 变为 1，当前写入数据发送完成；

  Step12. 重复 Step9 ~ Step11 以发送下一个待发送的数据。

19.5 寄存器描述

IRMOD_CR 红外调制控制寄存器

Address: 0x4001 0074　　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	IRSW	RW	红外调制软控制位 功能详见下方所示
3:0	MOD	RW	红外调制方式配置 0000：GTIM1_CH1 & BTIM2_TOGP 0001：GTIM1_CH1 & IRSW 0010：GTIM1_CH2 & BTIM3_TOGP 0011：GTIM1_CH2 & IRSW 0100：GTIM1_CH1 | BTIM2_TOGP 0101：GTIM1_CH1 | IRSW 0110：GTIM1_CH2 | BTIM3_TOGP 0111：GTIM1_CH2 | IRSW 1000：GTIM1_CH1 & UART1_TXD 1001：GTIM1_CH1 | UART1_TXD 1010：GTIM1_CH2 & UART1_TXD 1011：GTIM1_CH2 | UART1_TXD 1100：BTIM2_TOGP & UART2_TXD 1101：BTIM2_TOGP | UART2_TXD 1110：BTIM3_TOGP & UART2_TXD 1111：BTIM3_TOGP | UART2_TXD

20 模数转换器（ADC）

20.1 概述

本芯片内部集成一个14位分辨率、最高1M SPS转换速度的逐次比较型模数转换器(SAR ADC)，最多可将16路模拟信号转换为数字信号。

20.2 主要特性

- 14位分辨率
- 最高转换速率为1M SPS
- 16 个输入转换通道：
  - 13个外部管脚输入
  - 1个内置温度传感器
  - 1个内置BGR 1.2V基准
  - 1个AVCC/3电源电压
- 4种参考源（Vref）：
  - AVCC电源电压
  - ExREF管脚电压
  - 内置1.5V参考源
  - 内置2.5V参考源
- 采样电压输入范围：0~Vref
- 多种转换模式：
  - 单次转换
  - 多次转换
  - 连续转换
  - 序列扫描转换
  - 序列断续转换
- 支持输入通道电压阈值监测
- 内置电压跟随器，可转换高阻抗输入信号
- 支持片内外设自动触发ADC转换

20.3 功能描述

本节将详细描述模数转换器的相关功能。

20.3.1 功能框图

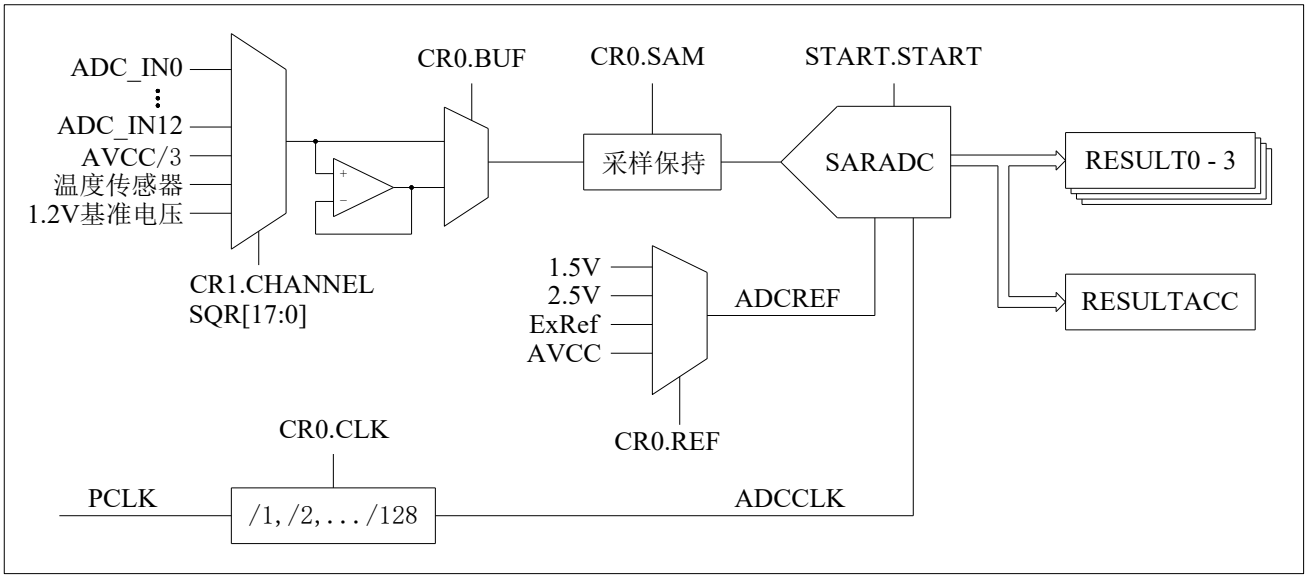

图 20-1 ADC 功能框图

20.3.2 启动时序

ADC 模拟电路从使能到可稳定工作需一定的时间。当设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667 后，模拟电路进入初始化流程；约 30us 后模拟电路初始化完成，可稳定工作。使能 ADC 后，可查询 ISR.READY 标志以等待模拟电路完成初始化流程。

20.3.3 模数转换过程

20.3.3.1 转换时序

当 ADC 初始化完成后，即可通过软件或硬件启动 ADC 转换。
一次典型的 ADC 转换时序如下图所示：

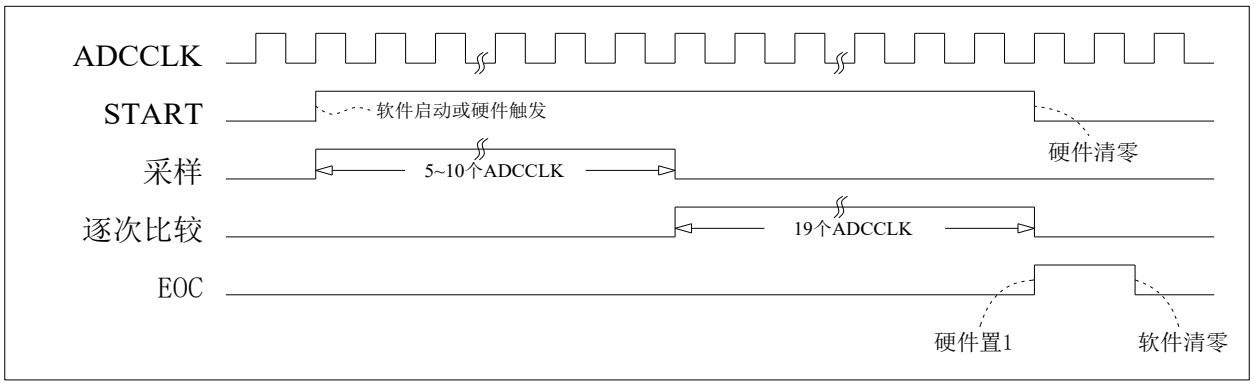

图 20-2 ADC 转换时序图

有两种方法可以启动 ADC 转换：向 ADC_START 寄存器写入 1 或通过 ADC_TRIGGER 寄存器所配置的外设触发。每次 ADC 转换由采样和逐次比较过程构成；其中采样过程需要 5~10 个 ADCCLK，逐次比较过程需要 19 个 ADCCLK。
ADC 采样过程的持续时间可配置为 5～10 个 ADCCLK，由 CR0.SAM 进行配置。该持续时间的合理时长由被采样信号的电气特性决定，用户应根据实测情况进行合理设置。
ADC 逐次比较过程完成后，转换结果存入 ADC_RESULT0～ADC_RESULT3，START 位被硬件清零，EOC 标志被硬件置 1。

20.3.3.2 转换速率

ADC 的转换速率与 ADCCLK 的频率及采样时钟个数相关。相关计算公式如下所示：

$$f_{conv} = \frac{f_{ADCCLK}}{(19 + N_{SA})}$$

f_{conv} 代表 ADC 转换速率，其单位为 SPS；

f_{ADCCLK} 代表 ADCCLK 频率，其单位为 Hz，由 CR0.CLK 配置；

N_{SA} 代表采样过程所需要的 ADCCLK 的数量，由 CR0.SAM 配置。

当 ADCCLK 的频率为 24MHz、N_{SA} 被配置为 5 时，ADC 的转换速率为 1M SPS，即每 1us 完成一次 ADC 转换。

ADC 转换速率与参考源、AVCC 电压及内部跟随器相关；最高转换速率如下表所示。

表 20-1 ADC 最高转换速率

参考源来源	AVCC电压	内置跟随器	最高转换速率	ADCCLK最高频率
内部1.5V	1.8V ~ 2.0V	-	100k SPS	2MHz
内部1.5V	2.0V ~ 5.5V	-	200k SPS	4MHz
内部2.5V	2.8V ~ 5.5V	-	200k SPS	4MHz
AVCC/ExREF	1.65V ~ 2.4V	-	200k SPS	4MHz
AVCC/ExREF	2.4V ~ 5.5V	使能	200k SPS	4MHz
AVCC/ExREF	2.4V ~ 2.7V	禁止	500k SPS	12MHz
AVCC/ExREF	2.7V ~ 5.5V	禁止	1M SPS	24MHz

20.3.3.3 参考源来源

ADC 模块提供了四种参考源供用户选择，由 CR0.REF 进行配置。

本芯片内部已集成两种参考源：1.5V 和 2.5V；当 AVCC 电压高于 1.8V 时可选择内置 1.5V 作为参考源；当 AVCC 电压高于 2.8V 时可选择内置 2.5V 作为参考源。当用户选择内部参考源作为 ADC 的参考源时，应同步配置 ADC 的转换速率不高于 200kSPS。

采用表达式*((uint16_t *) 0x001007BC)可读出内置 1.5V 参考源的电压值，其单位为 1mV；采用表达式*((uint16_t *) 0x001007BE)可读取内置 2.5V 参考源的电压值，其单位为 1mV。

若用户对参考源的精度要求较高，可以通过 ExREF 管脚外接独立高精度电压参考源；若 AVCC 的精度满足用户需求，可选择 AVCC 作为 ADC 的参考源；当选择 AVCC 或 ExREF 作为 ADC 的参考源时，应同步配置 ADC 的转换速率不高于 1MSPS。

20.3.3.4 内置电压跟随器

当待转换信号的驱动能力较弱时，采保电路在有限的采样持续时间内不能采样到待转换电压，进而造成 ADC 转换结果误差较大。当使能 ADC 内置的电压跟随器时（设置 CR0.BUF 为 1），待转换信号的驱动能力不再影响 ADC 转换的结果。内置待转信号（AVCC/3、温度传感器输出电压、内部 1.2V）的驱动能力较弱；对这些信号进行转换时，必须使能内置电压跟随器。

注意，当使能 ADC 内置的电压跟随器时，应同步配置 ADC 的转换速率不高于 200kSPS。

20.3.3.5 转换结果

当 ADC 工作于单通道转换模式时，14 位 ADC 转换结果存储在 RESULT0 中。
当 ADC 工作于序列转换模式时，14 位 ADC 转换结果存储在 RESULT0～RESULT3 中。
配置 CR1.ALIGN，可设定转换结果存储于 RESULTy[13:0]或 RESULTy[15:2]。
配置 CR1.DISCARD，可设定新转换完成的数据是否覆盖未被读取的旧数据。
ADC 转换结果与输入信号的电压及参考源的电压成比例关系，如下所示。

$$Result = \frac{V_{in}}{V_{ref}} \times 16383$$

20.3.4转换模式

通过 CR0.MODE 可将 ADC 模块配置为 7 种不同的工作模式：

- 单通道单次转换模式
  - ADC 启动后对所指定通道进行一次转换。
- 单通道多次转换模式
  - ADC 启动后对所指定通道进行多次（1～256 次）转换。
- 单通道持续转换模式
  - ADC 启动后对所指定通道持续进行转换。
- 序列扫描转换模式
  - ADC 启动后对待转换的所有序列均进行一次转换。
- 序列多次转换模式
  - ADC 启动后对待转换的所有序列共进行多次（1～256 次）转换。
- 序列持续转换模式
  - ADC 启动后对待转换的所有序列持续依次进行转换。
- 序列断续转换模式
  - ADC 启动后对待转换的所有序列中的一个进行一次转换，每转换一次换一个序列。

20.3.4.1 单通道单次转换模式

当配置 CR0.MODE 为 000 时，ADC 模块工作于单通道单次转换模式。ADC 启动后对所指定通道进行一次转换。

通过 CR1.CHANNEL 配置待转换的输入通道；通过 START.START 或 TRIGGER 启动 ADC 转换；每次转换完成时，转换结果存储到 RESULT0，START 位被硬件清零，ISR.EOC 标志被硬件置 1。若使能了转换完成中断（IER.EOC=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.EOC 控制位写入 0 以清除 ISR.EOC 标志。

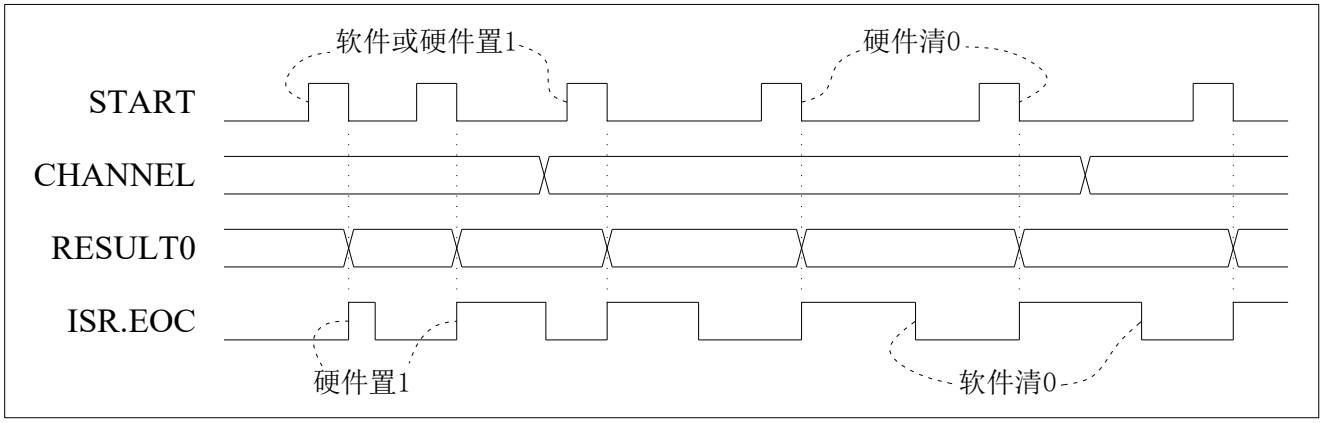

图 20-3 单通道单次转换时序图

- 单通道单次转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；
Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；
Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；
Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；
Step5. 设置 CR0.MODE 为 0，选择单通道单次转换模式；
Step6. 配置 CR0.REF，选择 ADC 的参考源；
当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。
Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；
Step8. 配置 CR1.CHANNEL，选择待转换的通道；
Step9. 设置 ICR.EOC 为 0，清除 ISR.EOC 标志；
Step10. 设置 START.START 为 1，启动 ADC 转换；
Step11. 查询等待 ISR.EOC 变为 1，从 RESULT0 寄存器中读出 ADC 转换结果；
Step12. 如需对其它通道进行转换，重复执行 Step8 至 Step11；
Step13. 设置 CR0.EN 为 0，关闭 ADC 模块。

- 单通道单次转换编程示例-硬件触发启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；
Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；
Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；
Step5. 设置 CR0.MODE 为 0，选择单通道单次转换模式；
Step6. 配置 CR0.REF，选择 ADC 的参考源；
当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。
Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；
Step8. 配置 TRIGGER 寄存器，选择触发启动信号来源；
Step9. 设置 IER.EOC 为 1，使能 ADC 转换完成中断；
Step10. 通过库函数，使能 ADC 的中断向量；
Step11. 配置 CR1.CHANNEL，选择待转换的通道；
Step12. 在 ADC 中断服务程序中，从 RESULT0 寄存器中读出 ADC 转换结果并设置 ICR.EOC 为 0 以清除 ISR.EOC 标志；
Step13. 如需对其它通道进行转换，重复执行 Step10 至 Step12；
Step14. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.2 单通道多次转换模式

当配置 CR0.MODE 为 001 时，ADC 模块工作于单通道多次转换模式。ADC 启动后对所指定通道进行多次（1 ~ 256 次）转换；该功能通常与累加功能联合使用以获取 ADC 转换结果的累加值或平均值。注意：在此模式下必须设置 CR2.ACCEN 为 1，ADC 才能正常工作。
通过 CR1.CHANNEL 配置待转换的输入通道；通过 CR2.CNT 配置转换次数；通过 CR2[9:8]使能并初始化累加功能；通过 START.START 或 TRIGGER 启动 ADC 转换。每次转换完成时，转换结果存储到 RESULT0，ISR.EOC 标志被硬件置 1；当达到所配置的转换次数时，累加结果存储到 RESULTACC，ISR.EOA 标志被硬件置 1，START.START 被硬件清零。
若使能了相应中断（IER.EOC=1 或 IER.EOA=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR 寄存器相应比特（ICR.EOC 或 ICR.EOA）写入 0 以清除中断标志。

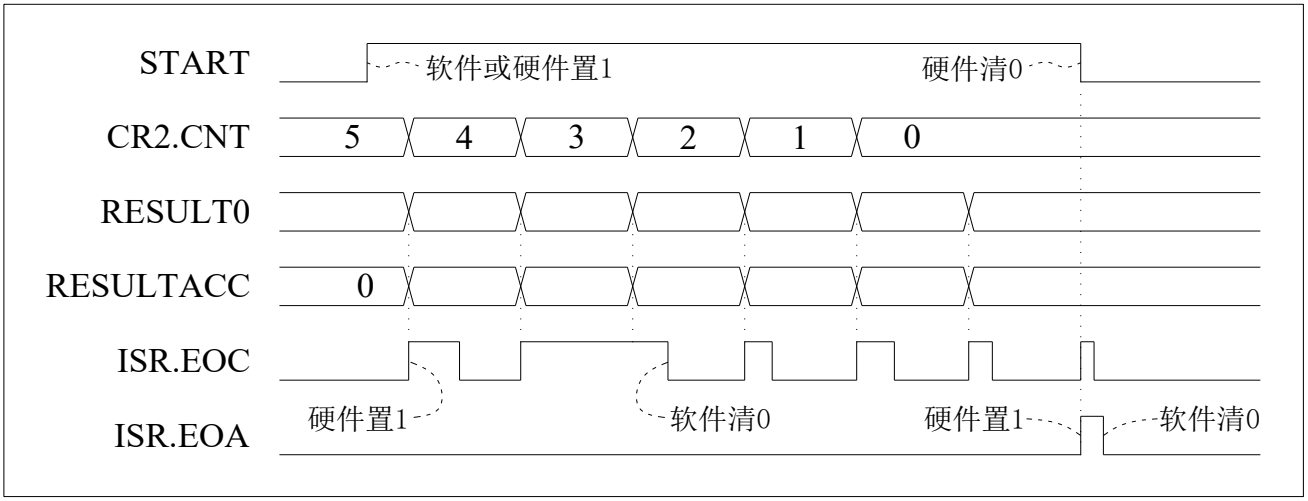

图 20-4 单通道多次转换时序图

- 单通道多次转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；
Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；
Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；
Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；
Step5. 设置 CR0.MODE 为 1，选择单通道多次转换模式；
Step6. 配置 CR0.REF，选择 ADC 的参考源；
当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。
Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；
Step8. 配置 CR1.CHANNEL，选择待转换的通道；
Step9. 设置 ICR.EOA 为 0，清除 ISR.EOA 标志；
Step10. 向 CR2.CNT 写入待转换的次数；
Step11. 设置 CR2.ACCEN 和 CR2.ACCRST 为 1，使能并初始化累加功能；
Step12. 设置 START.START 为 1，启动 ADC 转换；
Step13. 查询等待 ISR.EOA 变为 1，从 RESULTACC 寄存器中读出 ADC 转换结果累加值；
Step14. 如需对其它通道进行转换，重复执行 Step8 至 Step13；
Step15. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.3 单通道持续转换模式

当配置 CR0.MODE 为 010 时，ADC 模块工作于单通道持续转换模式。ADC 启动后对所指定通道持续进行转换直到用户设置 START.START 为 0 才会停止转换；用户在任意时刻均可获取当前 ADC 输入通道的最新转换值。

通过 CR1.CHANNEL 配置待转换的输入通道；通过 START.START 或 TRIGGER 启动 ADC 转换；每次转换完成时，转换结果存储到 RESULT0，ISR.EOC 标志被硬件置 1。若使能了转换完成中断（IER.EOC=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.EOC 控制位写入 0 以清除 ISR.EOC 标志。

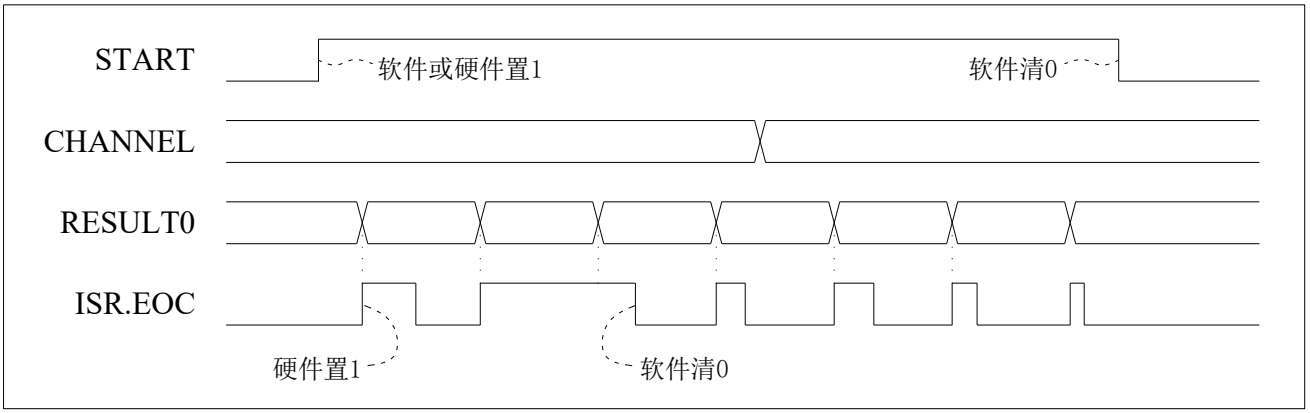

图 20-5 单通道持续转换时序图

- 单通道持续转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；

Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；

Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；

Step5. 设置 CR0.MODE 为 2，选择单通道持续转换模式；

Step6. 配置 CR0.REF，选择 ADC 的参考源；

当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。

Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；

Step8. 设置 CR1.DISCARD 为 0，选择新转换完成的结果覆盖未被读取的旧结果；

Step9. 配置 CR1.CHANNEL，选择待转换的通道；

Step10. 设置 START.START 为 1，启动 ADC 转换；

Step11. 第一次转换完成后，任意时刻均可从 RESULT0 读取最新转换完成的结果；

Step12. 如需对其它通道进行转换，直接修改 CR1.CHANNEL 寄存器即可；

Step13. 设置 START.START 为 0，即可停止 ADC 持续转换；

Step14. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.4 序列扫描转换模式

当配置 CR0.MODE 为 100 时，ADC 模块工作于序列扫描转换模式。ADC 启动后对待转换的所有序列均进行一次转换。

通过 SQR.SQRyCH（y = 0 - 3）配置每一个序列待转换的输入通道；通过 SQR.ENS 配置待转换序列的数量；通过 START.START 或 TRIGGER 启动 ADC 转换；硬件按照每个序列的配置自动设置待转换通道；每个序列转换完成时，转换结果存储到相应的 RESULTy 寄存器，ISR.EOC 标志被硬件置 1；当所有待转换序列转换完成时，ISR.EOS 标志被硬件置 1，START.START 被硬件清零。若使能了相应中断（IER.EOC=1 或 IER.EOS=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR 寄存器相应比特（ICR.EOC 或 ICR.EOS）写入 0 以清除中断标志。

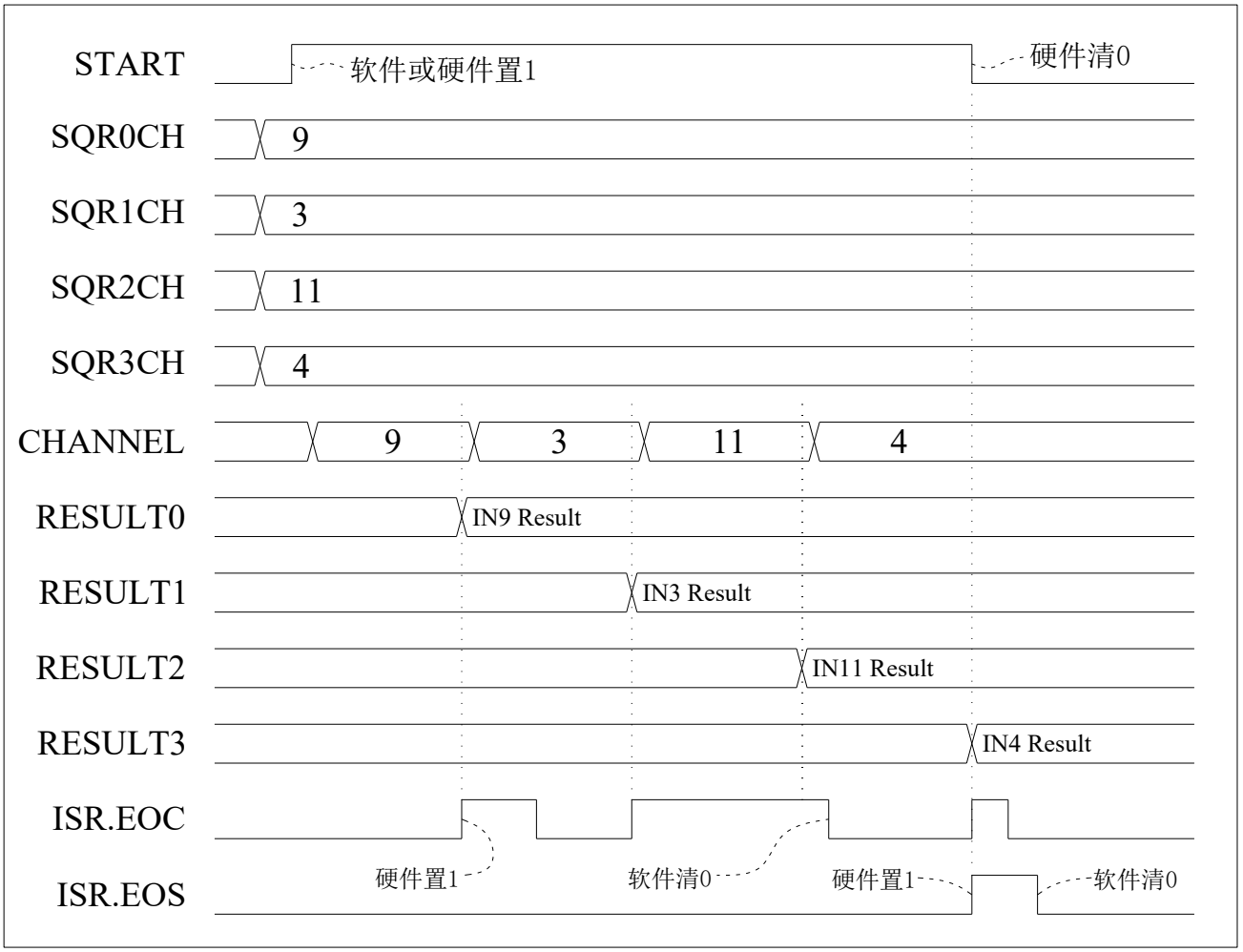

图 20-6 序列扫描转换时序图

- 序列扫描转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；

Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；

Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；

Step5. 设置 CR0.MODE 为 4，选择序列扫描转换模式；

Step6. 配置 CR0.REF，选择 ADC 的参考源；

当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。

Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；

Step8. 配置 SQR.SQRyCH，设置每个序列待转换的通道；

Step9. 配置 SQR.ENS，设置待转换的序列的数量；

Step10. 设置 ICR.EOS 为 0，清除 ISR.EOS 标志；

Step11. 设置 START.START 为 1，启动 ADC 转换；

Step12. 查询等待 ISR.EOS 变为 1，从 RESULT0～RESULT3 寄存器中读出序列转换的结果；

Step13. 如需对其它通道进行转换，重复执行 Step8 至 Step12；

Step14. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.5 序列多次转换模式

当配置 CR0.MODE 为 101 时，ADC 模块工作于序列多次转换模式。ADC 启动后按序列所配置的通道共进行多次（1 ~ 256 次）转换。该功能通常与累加功能联合使用以获取 ADC 转换结果的累加值或平均值。注意：在此模式下必须设置 CR2.ACCEN 为 1，ADC 才能正常工作。

通过 SQR.SQRyCH（y = 0 - 3）配置每一个序列待转换的输入通道；通过 SQR.ENS 配置待转换序列的数量；通过 CR2.CNT 配置转换次数；通过 CR2[9:8]使能并初始累加功能；通过 START.START 或 TRIGGER 启动 ADC 转换。每个序列转换完成时，转换结果存储到相应的 RESULT0 ~ RESULT3，ISR.EOC 标志被硬件置 1；当达到所配置的转换次数时，累加结果存储到 RESULTACC，ISR.EOA 标志被硬件置 1，START.START 被硬件清零。

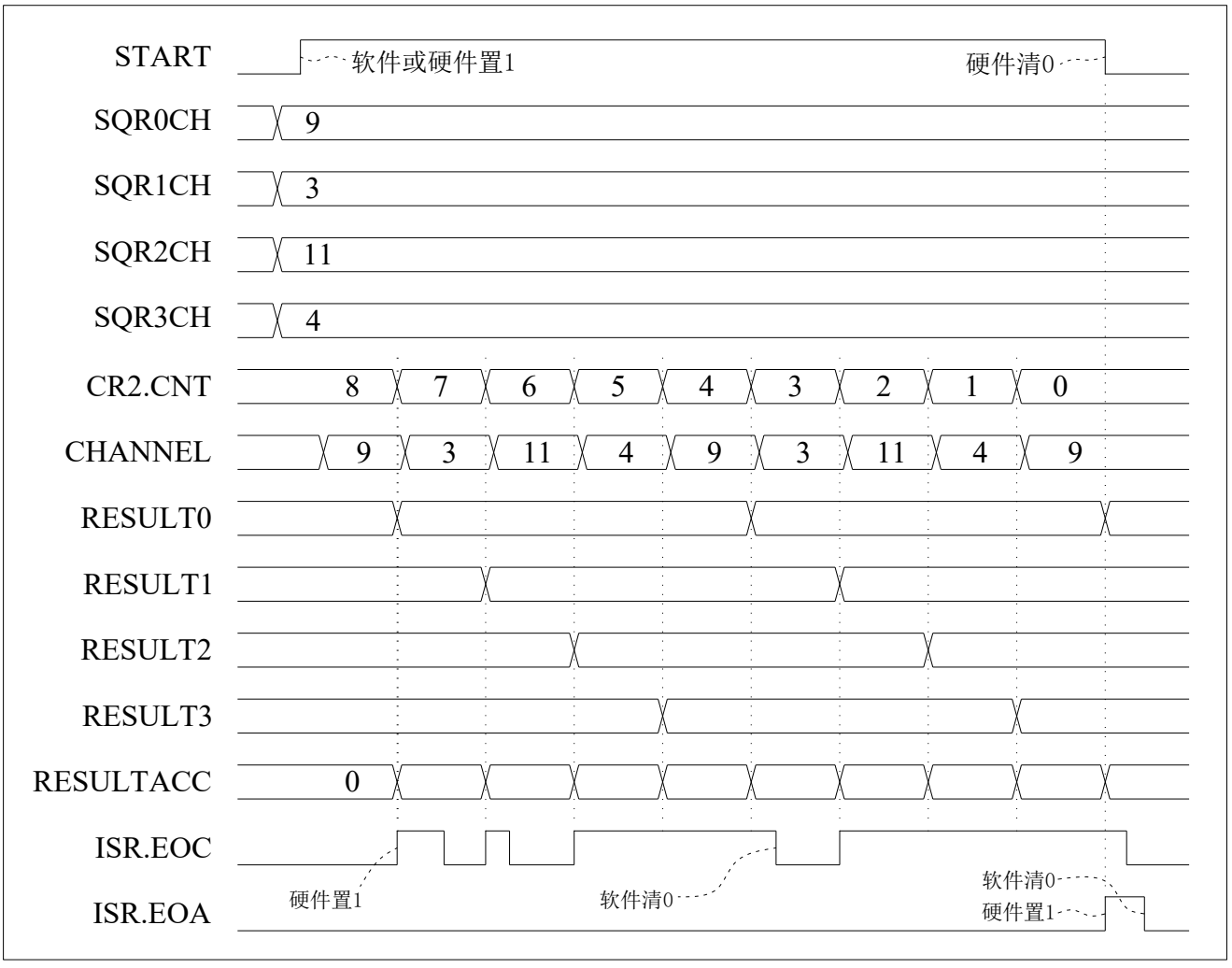

图 20-7 序列多次转换时序图

- 序列多次转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；

Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；

Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；

Step5. 设置 CR0.MODE 为 5，选择序列多次转换模式；

Step6. 配置 CR0.REF，选择 ADC 的参考源；

当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。

Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；

Step8. 配置 SQR.SQRyCH，设置每个序列待转换的通道；

Step9. 配置 SQR.ENS，设置待转换的序列的数量；

Step10. 向 CR2.CNT 写入待转换的次数；

Step11. 设置 CR2.ACCEN 和 CR2.ACCRST 为 1，使能并初始化累加功能；

Step12. 设置 ICR.EOA 为 0，清除 ISR.EOA 标志；

Step13. 设置 START.START 为 1，启动 ADC 转换；

Step14. 查询等待 ISR.EOA 变为 1，从 RESULTACC 寄存器中读出 ADC 转换结果累加值；

Step15. 如需对其它通道进行转换，重复执行 Step8 至 Step14；

Step16. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.6 序列持续转换模式

当配置 CR0.MODE 为 011 时，ADC 模块工作于序列持续转换模式。ADC 启动后按序列所配置的通道持续进行转换直到用户设置 START.START 为 0 才会停止转换；用户在任意时刻均可获取当前多个 ADC 输入通道的最新转换值。

通过 SQR.SQRyCH（y＝0 - 3）配置每一个序列待转换的输入通道；通过 SQR.ENS 配置待转换序列的数量；通过 START.START 或 TRIGGER 启动 ADC 转换。每个序列转换完成时，转换结果存储到相应的 RESULT0 ~ RESULT3 寄存器，ISR.EOC 标志被硬件置 1；若使能了转换完成中断（IER.EOC=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.EOC 控制位写入 0 以清除 ISR.EOC 标志。

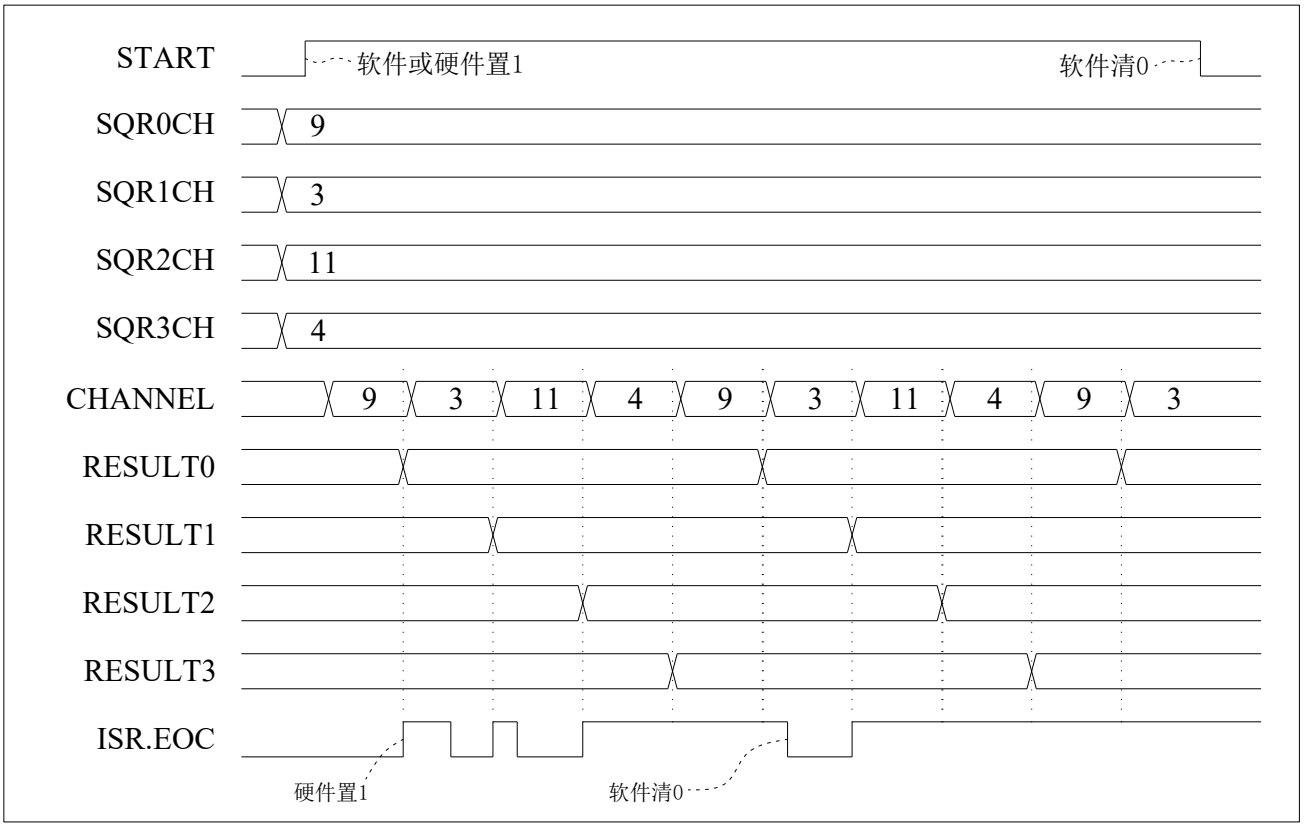

图 20-8 序列持续转换时序图

- 序列持续转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；

Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；

Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；

Step5. 设置 CR0.MODE 为 3，选择序列持续转换模式；

Step6. 配置 CR0.REF，选择 ADC 的参考源；

当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。

Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；

Step8. 设置 CR1.DISCARD 为 0，选择新转换完成的结果覆盖未被读取的旧结果；

Step9. 配置 SQR.SQRyCH，设置每个序列待转换的通道；

Step10. 配置 SQR.ENS，设置待转换的序列的数量；

Step11. 设置 START.START 为 1，启动 ADC 转换；

Step12. 当所有序列都转换完成一次后，任意时刻均可从 RESULT0~3 读取最新转换完成的结果；

Step13. 设置 START.START 为 0，即可停止 ADC 持续转换；

Step14. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.4.7 序列断续转换模式

当配置 CR0.MODE 为 110 时，ADC 模块工作于序列断续转换模式。ADC 每启动一次则对一个序列进行一次转换，转换完成后待转换的序列自动切换为下一个。

通过 SQR.SQRyCH（y＝0 - 3）配置每一个序列待转换的输入通道；通过 SQR.ENS 配置待转换序列的数量；通过 START.START 或 TRIGGER 启动 ADC 转换。转换完成时，转换结果存储到相应的 RESULT0～RESULT3，ISR.EOC 标志被硬件置 1，START.START 被硬件清零。

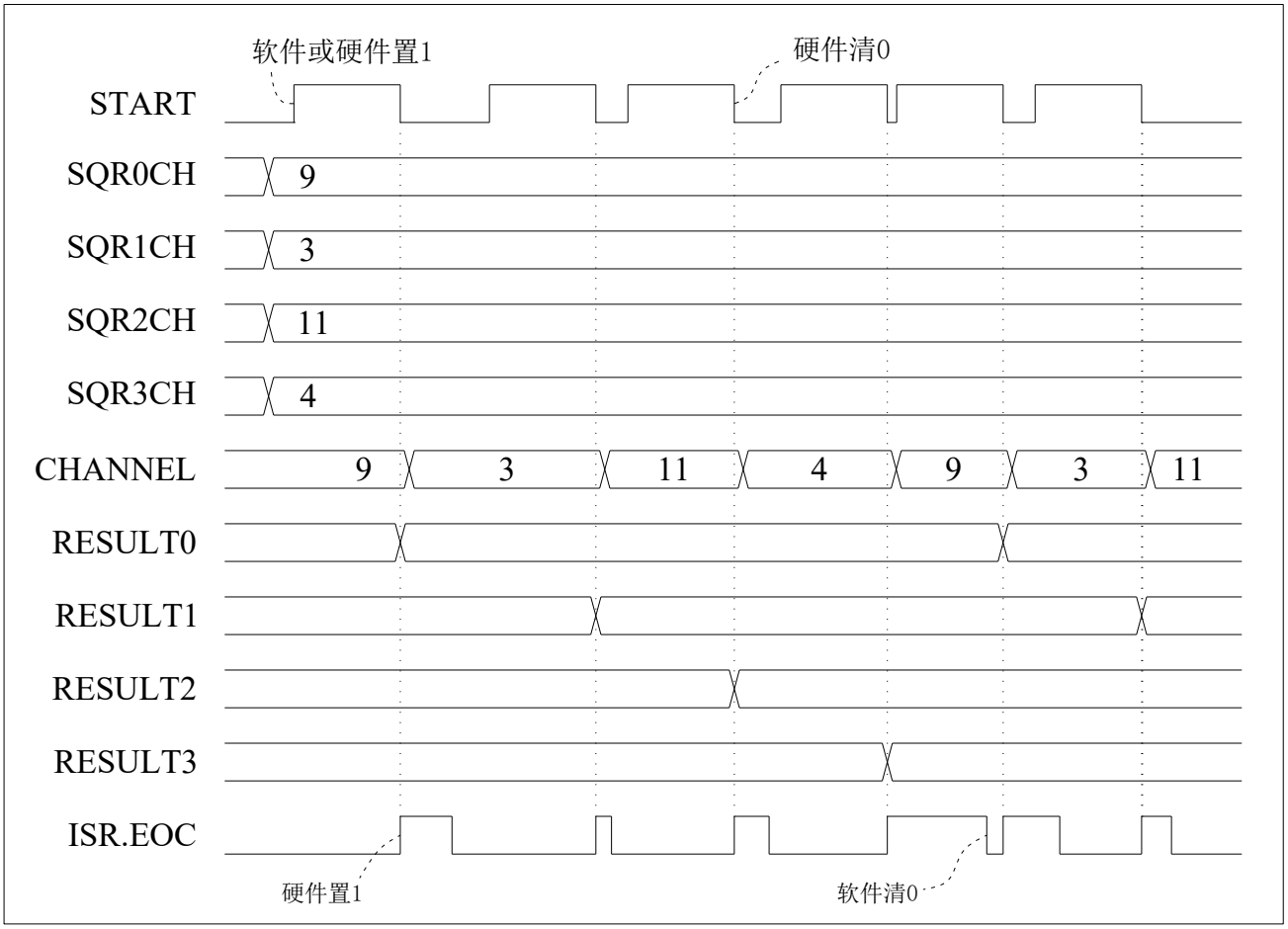

图 20-9 序列断续转换时序图

- 序列断续转换编程示例-软件启动

Step1. 设置待转换 ADC 通道所在的管脚为模拟功能；

Step2. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；

Step3. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；

Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；

Step5. 设置 CR0.MODE 为 6，选择序列断续转换模式；

Step6. 配置 CR0.REF，选择 ADC 的参考源；

当配置参考源为外部参考源时，需将外部参考源输入管脚设为模拟功能。

Step7. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；

Step8. 配置 SQR.SQRyCH，设置每个序列待转换的通道；

Step9. 配置 SQR.ENS，设置待转换的序列的数量；

Step10. 设置 ICR.EOC 为 0，清除 ISR.EOC 标志；
Step11. 设置 START.START 为 1，启动 ADC 转换；
Step12. 查询等待 ISR.EOC 变为 1，从 RESULT0 寄存器中读出序列 0 的 ADC 转换结果；
Step13. 设置 ICR.EOC 为 0，清除 ISR.EOC 标志；
Step14. 设置 START.START 为 1，启动 ADC 转换；
Step15. 查询等待 ISR.EOC 变为 1，从 RESULT1 寄存器中读出序列 1 的 ADC 转换结果；
Step16. 设置 ICR.EOC 为 0，清除 ISR.EOC 标志；
Step17. 设置 START.START 为 1，启动 ADC 转换；
Step18. 查询等待 ISR.EOC 变为 1，从 RESULT2 寄存器中读出序列 2 的 ADC 转换结果；
Step19. 设置 ICR.EOC 为 0，清除 ISR.EOC 标志；
Step20. 设置 START.START 为 1，启动 ADC 转换；
Step21. 查询等待 ISR.EOC 变为 1，从 RESULT3 寄存器中读出序列 3 的 ADC 转换结果；
Step22. 如需断续进行转换，重复执行 Step10 至 Step21；
Step23. 设置 CR0.EN 为 0，关闭 ADC 模块。

20.3.5转换结果累加

当设置 ADC_CR2.ACCEN 为 1 时，即可使能转换结果累加功能。每当 ADC 完成一次转换，累加值寄存器 RESULTACC 内的数值自动加上本次转换完成的结果。当设置 ADC_CR2.ACCRST 为 1 时，累加值寄存器 ADC_RESULTACC 内的数值清零。
无论 ADC 工作于何种工作模式，均可使用该功能获取 ADC 转换结果的累加值或平均值。

20.3.6自动关闭

当设置 ADC_START.AUTOSTOP 为 1 时，即可使能 ADC 自动关闭功能。当 ADC 工作于非持续转换模式下且 ADC 完成了所有待进行的转换时，START.START 和 ADC_CR0.EN 位同时被硬件自动清零。

20.3.7外部触发启动

ADC 支持片内外设触发启动，可实现 ADC 转换的时刻点与外设中断严格同步。
ADC 支持的片内外设触发源为：I2C 中断信号 / SPI 中断信号 / UART 中断信号 / BTIM 中断信号 / GTIM 中断信号 / ATIM 输出的触发信号。
在 ADC_TRIGGER 寄存器的触发使能位与相应外设的中断使能位均置 1 条件下，相应外设的中断标志由 0 变为 1 时，ADC 立即自动启动转换；转换完成的结果存入 ADC_RESULTy 寄存器；用户应及时清除相应外设的中断标志以便下一次触发能正常工作。

20.3.8模拟看门狗

模拟看门狗可实现对 ADC 输入的模拟量的自动监测，直到其转换结果符合用户预期时才产生中断申请用户程序介入。

当设置 CR1.WDTALL 为 1 时，模拟看门狗对每一个通道转换完成的 ADC 结果的高 12 比特进行监测并置位相应的区间标志；当设置 CR1.WDTALL 为 0 时，模拟看门狗对 CR1.WDTCH 所指定的通道转换完成的 ADC 结果的高 12 比特进行监测并置位相应的区间标志。

模拟看门狗支持三种监测区间：高阈值区间、低阈值区间和中阈值区间；通过 VTH 寄存器和 VTL 寄存器可配置每个区间的范围。

- 当 ADC 转换结果位于低阈值区间时（0≤Result[13:2]＜ADC_VTL），ISR.WDTL 被硬件置 1；若使能了低阈值中断（IER.WDTL=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.WDTL 控制位写入 0 以清除 ISR.WDTL 标志。
- 当 ADC 转换结果位于高阈值区间时（ADC_VTH≤Result[13:2]≤4095），ISR.WDTH 被硬件置 1；若使能了高阈值中断（IER.WDTH=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.WDTH 控制位写入 0 以清除 ISR.WDTH 标志。
- 当 ADC 转换结果位于中阈值区间内时（ADC_VTL≤Result[13:2]＜ADC_VTL），ISR.WDTM 被硬件置 1；若使能了中阈值中断（IER.WDTM=1），则会执行 ADC 的中断服务程序，用户在退出中断服务程序前应向 ICR.WDTM 控制位写入 0 以清除 ISR.WDTM 标志。

20.3.9温度传感器

本芯片内置温度传感器模块，传感器的输出电压与温度成正比关系，其输出电压的典型值如下表所示。不同芯片在相同温度下的输出电压存在一定的偏差，故出厂时对温度传感器进行了校准：记录校准时的环境温度存入 0x1007C5 地址；记录 ADC 参考电压为 1.5V 时对温度传感器输出电压进行转换的结果的高 12 比特存入 0x1007C6 - 0x1007C7 地址；记录 ADC 参考电压为 2.5V 时对温度传感器输出电压进行转换的结果的高 12 比特存入 0x1007C8 - 0x1007C9 地址。

表 20-2 温度传感器输出电压典型值

温度（℃）	电压（V）	温度（℃）	电压（V）	温度（℃）	电压（V）
105	1.00	55	0.87	5	0.74
95	0.97	45	0.84	-5	0.71
85	0.95	35	0.82	-15	0.69
75	0.92	25	0.79	-25	0.66
65	0.90	15	0.76	-35	0.63

使用 ADC 对温度传感器输出电压转换完成后，通过下方表达式即可计算当前的环境温度。

- 当 ADC 参考源为内部 1.5V 时环境温度计算表达式如下：

```
Ta = (*(uint8_t *)0x1007C5) * 0.5 + 0.0000231 * (*(int16_t *)0x1007BC) *
((int16_t)ADC_RESULT0 - (*(int16_t*)0x1007C6) * 4 );
```

- 当 ADC 参考源为内部 2.5V 时环境温度计算表达式如下：

```
Ta = (*(uint8_t *)0x1007C5) * 0.5 + 0.0000231 * (*(int16_t *)0x1007BE) *
((int16_t)ADC_RESULT0 - (*(int16_t *)0x1007C8) * 4 );
```

- 测量环境温度编程示例

Step1. 设置 SYSCTRL_APBEN2.ADC 为 1，使能 ADC 工作时钟；
Step2. 设置 CR0.EN 为 1 并向 RSTCAL 写入 0x6667，初始化 ADC 模块；
Step3. 设置 CR0.TSEN 为 1，使能内置温度传感器；
Step4. 查询等待 ISR.READY 位变为 1，等待 ADC 模块初始化完成；
Step5. 设置 CR1.CHANNEL 为 0x0E，选择待转换的通道为内部温度传感器；
Step6. 设置 CR0.MODE 为 0，选择单通道单次转换模式；
Step7. 配置 CR0.REF，选择 ADC 的参考源为内置 1.5V 或 2.5V；
Step8. 设置 CR0.BUF 为 1，使能输入通道内置跟随器；
Step9. 配置 CR0.CLK 及 CR0.SAM，设置 ADC 的转换速率；
注意：转换速率应小于 200kSPS。
Step10. 设置 START.START 为 1，启动 ADC 转换；
Step11. 查询等待 START.START 变为 0，从 RESULT0 寄存器中读出 ADC 转换结果；
Step12. 设置 CR0.EN 为 0，关闭 ADC 模块；
Step13. 依据环境温度计算表达式以计算当前环境温度。

20.4 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
ADC_CR0	0x40012400 + 0x00	ADC控制寄存器0
ADC_CR1	0x40012400 + 0x04	ADC控制寄存器1
ADC_START	0x40012400 + 0x08	ADC启动寄存器
ADC_SQR	0x40012400 + 0x0C	ADC序列配置寄存器
ADC_CR2	0x40012400 + 0x10	ADC控制寄存器2
ADC_VTH	0x40012400 + 0x14	ADC高阈值寄存器
ADC_VTL	0x40012400 + 0x18	ADC低阈值寄存器
ADC_TRIGGER	0x40012400 + 0x1C	ADC外部触发寄存器
ADC_RESULT0	0x40012400 + 0x20	ADC转换结果0寄存器
ADC_RESULT1	0x40012400 + 0x24	ADC转换结果1寄存器
ADC_RESULT2	0x40012400 + 0x28	ADC转换结果2寄存器
ADC_RESULT3	0x40012400 + 0x2C	ADC转换结果3寄存器
ADC_RESULTACC	0x40012400 + 0x30	ADC转换结果累加值寄存器
ADC_IER	0x40012400 + 0x34	ADC中断使能寄存器
ADC_ICR	0x40012400 + 0x38	ADC中断标志清除寄存器
ADC_ISR	0x40012400 + 0x3C	ADC中断标志寄存器
ADC_RSTCAL	0x40012400 + 0x88	ADC复位校准寄存器

ADC_CR0 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	RFU	-	保留位，请保持默认值
15:14	BIAS	RW	请保持默认值不变
13	BUF	RW	内置跟随器使能控制 0：关闭跟随器，外部输入信号与ADC直接相连 1：打开跟随器，外部输入信号通过跟随器后与ADC相连 注意1：使能该功能时，最大转换速度为200k SPS 注意2：以下条件必须使能该功能： 待转换的信号输出阻抗较高；待转换的信号为AVCC/3、内置温度传感器输出电压、内部基准1.2V输出电压
12:11	SAM	RW	ADC采样周期选择 00：5个ADCCLK时钟周期 10：8个ADCCLK时钟周期 01：6个ADCCLK时钟周期 11：10个ADCCLK时钟周期
10:8	CLK	RW	ADCCLK来源配置 000：PCLK 100：PCLK / 16 001：PCLK / 2 101：PCLK / 32 010：PCLK / 4 110：PCLK / 64 011：PCLK / 8 111：PCLK / 128
7:6	REF	RW	ADC参考源选择 00：内置1.5V参考源，其精确值存储于0x1007BC ~ 0x1007BD 01：内置2.5V参考源，其精确值存储于0x1007BE ~ 0x1007BF 10：外部参考源ExREF 11：AVCC电源电压
5	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
4	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
3:1	MODE	RW	ADC工作模式配置 000：单通道单次转换模式 100：序列扫描转换模式 001：单通道多次转换模式 101：序列多次转换模式 010：单通道持续转换模式 110：序列断续转换模式 011：序列持续转换模式
0	EN	RW	ADC使能控制 0：禁止ADC 1：使能ADC 注意：使能后，需等待ADC启动完成（ISR.READY变为1）。

ADC_CR1 控制寄存器 1

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:14	RFU	-	保留位，请保持默认值
13	WDTALL	RW	所有通道模拟看门狗使能控制 0：禁止模拟看门狗 1：使能模拟看门狗
12	RFU	-	保留位，请保持默认值
11:8	WDTCH	RW	单通道模拟看门狗监测通道配置 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：AVCC/3 0110：ADC_IN6 1110：内置温度传感器 0111：ADC_IN7 1111：1.2V内核电压基准源
6	ALIGN	RW	ADC转换结果对齐方式 0：右对齐，转换结果存储于[13:0] 1：左对齐，转换结果存储于[15:2]
5	DISCARD	RW	单通道ADC转换结果保存策略配置 0：覆盖未被读取的旧数据，保留新数据 1：丢弃新转换完成的数据，保留未读取的旧数据
4	RFU	-	保留位，请保持默认值
3:0	CHANNEL	RW	单通道转换模式待转换通道配置 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：AVCC/3 0110：ADC_IN6 1110：内置温度传感器 0111：ADC_IN7 1111：1.2V内核电压基准源

注意：当选择 1101～1111 通道时，需使能内置跟随器且转换速率应不大于 200kSPS。

ADC_CR2 控制寄存器 2

Address offset: 0x10 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	ACCRST	R0W1	累加值寄存器清0控制 W0：无功能 W1：清零转换结果累加值寄存器ADC_RESULTACC
8	ACCEN	RW	转换结果累加使能 0：禁止累加功能 1：使能累加功能 注意：多次转换模式必须使能该功能
7:0	CNT	RW	多次转换模式转换次数配置 转换次数为CNT+1

ADC_RSTCAL 复位校准寄存器

Address offset: 0x88 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	RSTCAL	WO	写入0x6667以启动复位校准

ADC_SQR 序列配置寄存器

Address offset: 0x0C 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
17:16	ENS	RW	待转换的序列配置 00：转换SQR0CH 10：转换SQR0CH - SQR2CH 01：转换SQR0CH - SQR1CH 11：转换SQR0CH - SQR3CH
15:12	SQR3CH	RW	序列3待转换通道配置 详见SQR0CH
11:8	SQR2CH	RW	序列2待转换通道配置 详见SQR0CH
7:4	SQR1CH	RW	序列1待转换通道配置 详见SQR0CH
3:0	SQR0CH	RW	序列0待转换通道配置 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：AVCC/3 0110：ADC_IN6 1110：内置温度传感器 0111：ADC_IN7 1111：1.2V内核电压基准源

注意：当选择 1101～1111 通道时，需使能内置跟随器且转换速率应不大于 200kSPS。

ADC_START 启动寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:2	RFU	-	保留位，请保持默认值
1	AUTOSTOP	RW	ADC自动禁止配置 0：转换完成后继续保持CR0.EN为1 1：转换完成后自动设置CR0.EN为0
0	START	RW	ADC启动转换控制 0：停止ADC转换 1：启动ADC转换 注意：启动多次转换模式前需清除EOA标志。

ADC_VTH 高阈值寄存器

Address offset: 0x14 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	VTH	RW	模拟看门狗检测高阈值

ADC_VTL 低阈值寄存器

Address offset: 0x18 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
11:0	VTL	RW	模拟看门狗检测低阈值

ADC_TRIGGER 外部触发寄存器

Address offset: 0x1C 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	I2C1	RW	I2C1中断触发ADC启动 0：禁止 1：使能
12	RFU	-	保留位，请保持默认值
11	SPI1	RW	SPI1中断触发ADC启动 0：禁止 1：使能
10	RFU	-	保留位，请保持默认值
9	UART2	RW	UART2中断触发ADC启动 0：禁止 1：使能
8	UART1	RW	UART1中断触发ADC启动 0：禁止 1：使能
7	BTIM3	RW	BTIM3中断触发ADC启动 0：禁止 1：使能
6	BTIM2	RW	BTIM2中断触发ADC启动 0：禁止 1：使能
5	BTIM1	RW	BTIM1中断触发ADC启动 0：禁止 1：使能
4:2	RFU	-	保留位，请保持默认值
1	GTIM1	RW	GTIM1中断触发ADC启动 0：禁止 1：使能
0	ATIM	RW	ATIM输出的触发信号触发ADC启动 0：禁止 1：使能

ADC_IER 中断使能寄存器

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	OVW	RW	转换结果溢出中断使能控制 0：禁止 1：使能
5	WDTM	RW	模拟看门狗中阈值区间中断使能控制 0：禁止 1：使能
4	WDTH	RW	模拟看门狗高阈值区间中断使能控制 0：禁止 1：使能
3	WDTL	RW	模拟看门狗低阈值区间中断使能控制 0：禁止 1：使能
2	EOA	RW	多次转换完成中断使能控制 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	转换完成中断使能控制 0：禁止 1：使能

ADC_ISR 中断标志寄存器

Address offset: 0x3C 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	READY	RO	模拟电路初始化完成标志 0：初始化未完成 1：初始化已完成，可以开始进行ADC转换
6	OVW	RO	转换结果溢出标志 0：未发生该事件 1：结果寄存器中的数据尚未被读取时，又有新的转换完成
5	WDTM	RO	模拟看门狗中阈值区间标志 0：转换结果高12Bit位于[ADC_VTL ， ADC_VTH)区间外 1：转换结果高12Bit位于[ADC_VTL ， ADC_VTH)区间内
4	WDTH	RO	模拟看门狗高阈值区间标志 0：转换结果高12Bit位于[ADC_VTH ， 4095]区间外 1：转换结果高12Bit位于[ADC_VTH ， 4095]区间内
3	WDTL	RO	模拟看门狗低阈值区间标志 0：转换结果高12Bit位于[0 ， ADC_VTL)区间外 1：转换结果高12Bit位于[0 ， ADC_VTL)区间内
2	EOA	RO	多次转换完成标志 0：多次转换未完成 1：多次转换已完成
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

ADC_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	OVW	R1W0	转换结果溢出标志清0控制 W0：清除ISR寄存器中的相应标志 W1：无功能
5	WDTM	R1W0	模拟看门狗中阈值区间标志清0控制 W0：清除ISR寄存器中的相应标志 W1：无功能
4	WDTH	R1W0	模拟看门狗高阈值区间标志清0控制 W0：清除ISR寄存器中的相应标志 W1：无功能
3	WDTL	R1W0	模拟看门狗低阈值区间标志清0控制 W0：清除ISR寄存器中的相应标志 W1：无功能
2	EOA	R1W0	多次转换完成标志清0控制 W0：清除ISR寄存器中的相应标志 W1：无功能
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

ADC_RESULT0 转换结果 0 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
15:0	RESULT	RO	ADC转换结果0寄存器

ADC_RESULT1 转换结果 1 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
15:0	RESULT	RO	ADC转换结果1寄存器

ADC_RESULT2 转换结果 2 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	RESULT	RO	ADC转换结果2寄存器

ADC_RESULT3 转换结果 3 寄存器

Address offset: 0x2C 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转换结果3寄存器

ADC_RESULTACC 转换结果累加值寄存器

Address offset: 0x30 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	RESULT	RO	ADC转换结果累加值寄存器

21 模拟电压比较器（VC）

21.1 概述

本芯片内部集成 2 个模拟电压比较器，支持从管脚输出比较结果。模拟电压比较器具备迟滞功能、滤波功能、极性选择功能、窗口比较功能。其中断，可唤醒 DeepSleep 状态下的 MCU。

21.2 主要特性

- 2 个模拟电压比较器单元
- 配置 64 阶电阻分压器
- 多达 8 路外部模拟信号输入
- 具备 4 路内部模拟输入信号：
  - 内置电阻分压器输出电压
  - 内置温度传感器输出电压
  - 内置 1.2V 基准电压
  - ADC 参考源
- 可选择输出极性
- 支持迟滞功能
- 支持窗口比较功能
- 支持数字滤波
- 支持 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
- 支持低功耗运行，可唤醒 DeepSleep 状态下的 MCU

21.3 功能描述

本节将详细描述模拟电压比较器的相关功能。

21.3.1 功能框图

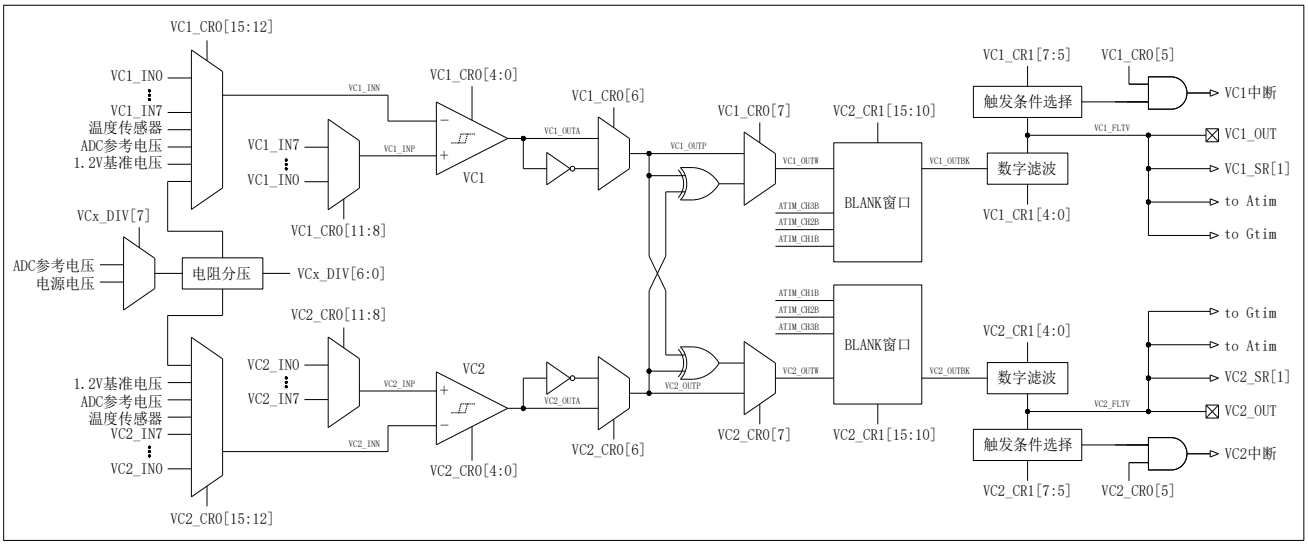

图 21-1 VC 功能框图

21.3.2 建立/响应时间

电压比较器的建立时间是指从 VC 使能到输出正确比较结果的时间，在建立时间完成前 VC 的输出结果可能是不符合预期的。使能 VC 后，应查询 SR.Ready 标志以等待 VC 建立完成。

电压比较器的响应时间是指 VC 使能后，从输入电压发生变化到输出正确比较结果的时间。响应时间由 VCx_CR0.RESP 位域配置，从 200ns 到 20us 四档可调。响应时间越短，功耗越大。

21.3.3 输入通道

电压比较器的正端和负端均有多个输入信号可供选择。配置 CR0.PCHANNEL 可选择电压比较器正端输入信号为 VCx_IN0～VCx_IN7；配置 CR0.NCHANNEL 可选择电压比较器负端输入信号为 VCx_IN0 ~ VCx_IN7、内置分压器的输出电压、ADC 模块的参考源、内置 1.2V 参考源、内置温度传感器的输出电压。注意：当选择内置参考源时，需使能生成该电压的模块，详见寄存器描述。

21.3.4 内置电阻分压器

电压比较器配备了内置电阻分压器，其输出电压为 $\frac{1}{64}Ref \sim \frac{64}{64}Ref$，通过 DIV[5:0]位域进行配置。分压器具有两种参考源来源，分别为 AVCC 和 ADC 模块所配置的参考源；通过 DIV[7]位域进行配置。

21.3.5迟滞功能

通过 CR0.HYS 可配置电压比较器的迟滞功能。使能迟滞功能后，当正负极输入信号电压接近时比较器的输出结果不会频繁发生翻转；只有当输入信号的电压差大于阈值电压时，比较器输出信号才会发生翻转。支持四档迟滞选项：没有迟滞、10mv 迟滞、20mv 迟滞和 30mv 迟滞。

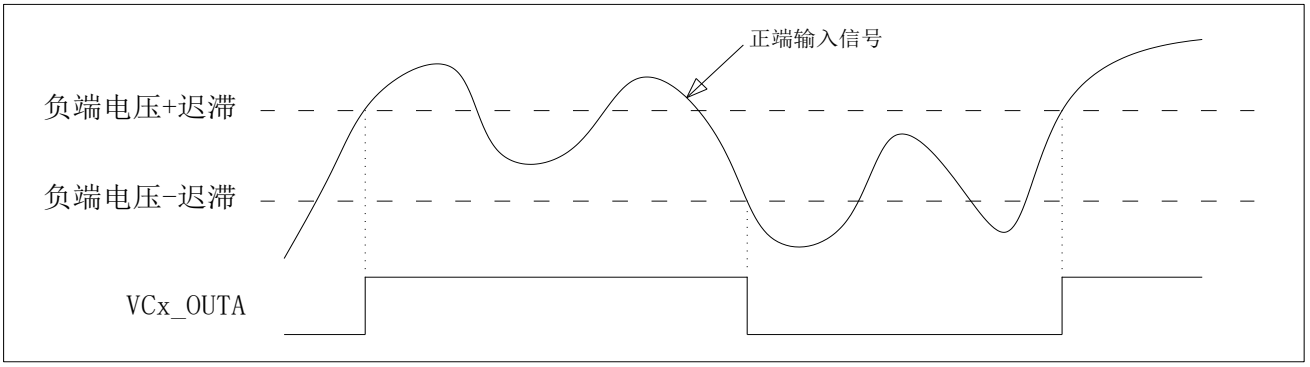

图 21-2 迟滞响应波形

使能迟滞功能时，外部输入的正端信号通过 R1 连接到比较器内核的正端；比较器内核的正端通过反馈电阻 R2 连接到比较器内核的输出端。在 R1 与 R2 的共同作用下，比较器实现了迟滞功能，如下所示。注意，当正端输入信号的内阻较大时，电阻 R1、R2 会影响 VCx_INP 管脚的输入电压。

R1 与 R2 的阻值由 VCx_CR0[4:3]控制，如下表所示。

VCx_CR0[4:3]	R1阻值	R2阻值
00	短路	开路
01	~4 kΩ	~2 MΩ
10	~10 kΩ	~2 MΩ
11	~16 kΩ	~2 MΩ

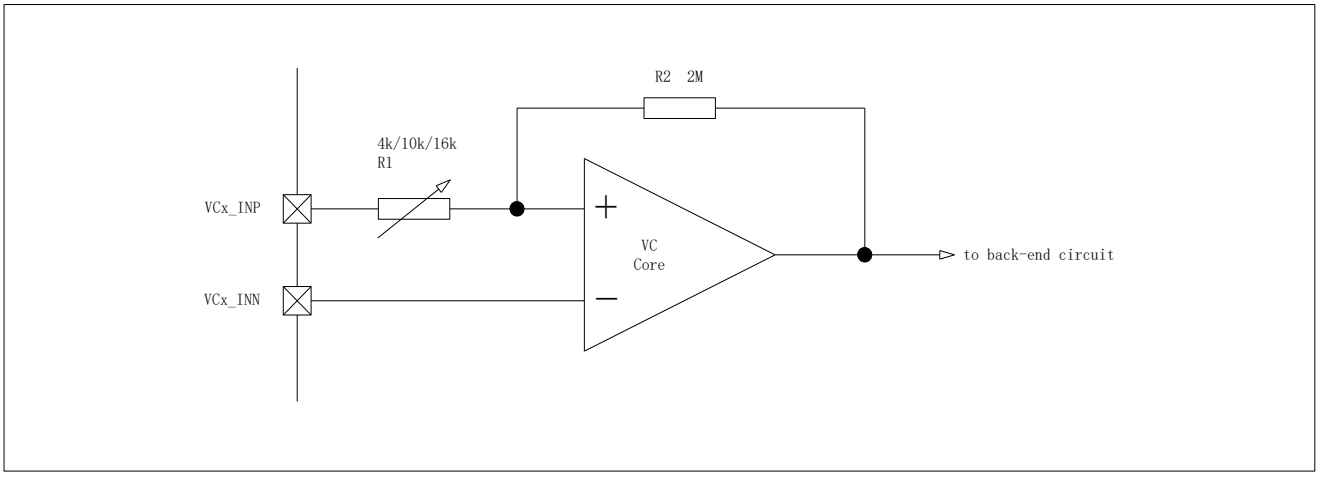

图 21-3 迟滞电路示意图

21.3.6极性选择

通过 CR0.POL 位域可改变电压比较器输出信号 VCx_OUTP 的极性为正向或反向。
当 CR0.POL 为 1 时：VCx_OUTP 与 VCx_OUTA 反向；即正端电压高于负端电压时输出低。
当 CR0.POL 为 0 时：VCx_OUTP 与 VCx_OUTA 同向；即正端电压高于负端电压时输出高。

21.3.7窗口比较功能

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。
当 CR0.WINDOW 为 1 时：VCx_OUTW = VC1_OUTP 异或 VC2_OUTP
当 CR0.WINDOW 为 0 时：VCx_OUTW = VCx_OUTP

21.3.8BLANK 窗口功能

BLANK 窗口与 ATIM 配合，用于屏蔽 PWM 信号的变化点附近的噪声干扰信号。
每当上升沿检测电路检测到上升沿时，计数器开始计数并持续 $2^{(BLANKTIME+2)}$个 PCLK 周期。计数器计数期间输出低电平，故 VCx_OUTBK 也为低电平；计数器停止期间输出高电平，故 VCx_OUTBK 信号与 VCx_OUTW 信号相同。

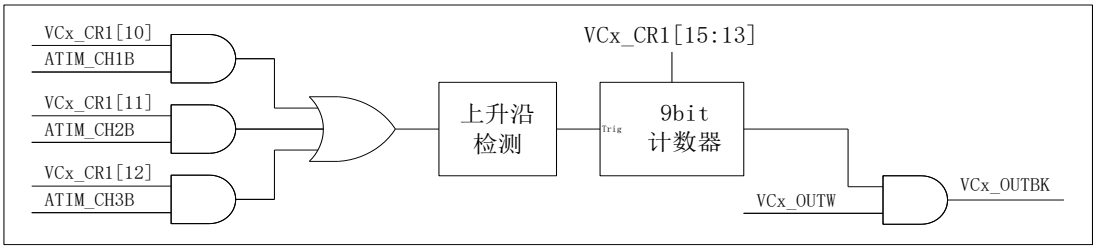

图 21-4 BLANK 窗口功能框图

21.3.9数字滤波

电压比较器内置数字滤波器，可对电压比较器的输出信号进行数字滤波。该功能可过滤输入信号中大于迟滞电压的噪声，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。通过 CR1.FLTEN 可使能或关闭数字滤波功能；通过 CR1.FLTCLK 可配置滤波时钟为 PCLK 或 RC120K；通过 CR1.FLTTIME 可配置滤波宽度为 1～4095 个滤波时钟。
通过状态寄存器 VCx_SR.FLTV 可以读出电压比较器经数字滤波后的输出 VCx_FLTV；配置 GPIO 功能为 VCx_OUT 时，该管脚即可实时输出数字滤波后的信号波形以方便观察测量。
当用户设置了 VCx_OUT 管脚为数字输出，且选择 VC 比较输出的复用功能时，VCx_OUT 管脚将输出电压比较器经数字滤波后的输出电平。
下图展示了滤波宽度为 1 时数字滤波器的输入与输出波形。

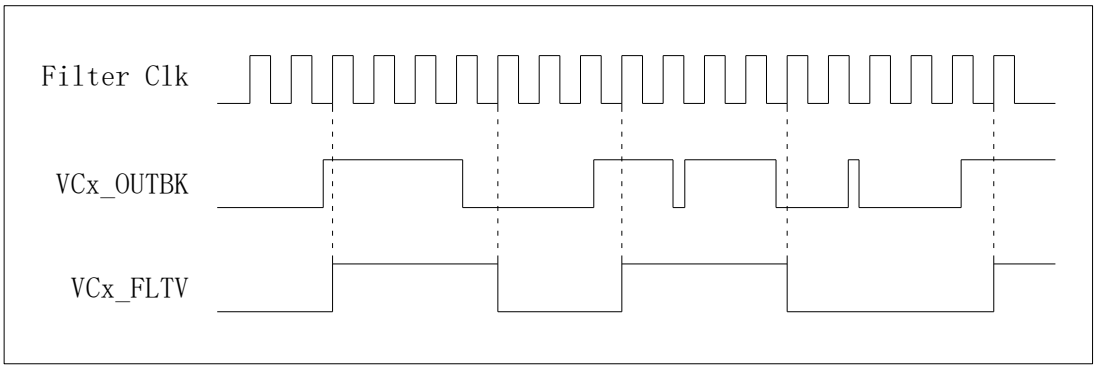

图 21-5 数字滤波输入输出波形

21.3.10 中断

本电压比较器在 DeepSleep 模式下可正常工作，其中断可将芯片从 DeepSleep 模式唤醒。通过配置 CR1[7:5]可设置中断标志的触发条件为上升沿、下降沿或高电平。当 VCx_FLTV 信号符合 CR1[7:5]所配置的触发条件时，SR.INTF 会被硬件置 1；若 CR0.IE 为 1 则会产生相应的 VCx 中断；在退出中断服务程序前，用户应向 SR.INTF 写入 0 以清除该中断标志。

21.4 编程示例

Step1. 配置 VCx_CR0.PCHANNEL，选择正端待监测的电压来源；
Step2. 配置 VCx_CR0.NCHANNEL，选择负端待监测的电压来源；
Step3. 配置 VCx_CR1.FLTTIME，选择滤波时间；
Step4. 配置 VCx_CR1.FLTCLK，选择滤波时钟；
Step5. 设置 VCx_CR1.FLTEN 为 1，使能 VCx 滤波；
Step6. 配置 VCx_CR1[7:5]，选择中断触发方式；
Step7. 设置 VCx_CR0.EN 为 1，使能 VCx；
Step8. 查询等待 VCx_SR.READY 为 1，VC 模块初始化完成；
Step9. 设置 VCx_CR0.IE 为 1，使能 VCx 中断；
Step10. 在中断服务程序执行需要进行的操作，在退出中断服务程序之前应清除中断标志。

21.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
VCx_DIV	VCx_Base + 0x00	电阻分压控制寄存器
VCx_CR0	VCx_Base + 0x04	控制寄存器0
VCx_CR1	VCx_Base + 0x08	控制寄存器1
VCx_SR	VCx_Base + 0x0C	状态寄存器

VC1_Base = 0x4001 2A00
VC2_Base = 0x4001 2A10

VCx_CR0 控制寄存器 0

Address offset: 0x04 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:12	NCHANNEL	RW	VCx 负端输入信号配置 0000：VCx_IN0 0100：VCx_IN4 0001：VCx_IN1 0101：VCx_IN5 0010：VCx_IN2 0110：VCx_IN6 0011：VCx_IN3 0111：VCx_IN7 1000：内置分压器输出电压 1001：ADC模块所配置的参考源（注意：需使能ADC_CR0.EN） 1010：内置1.2V基准电压（注意：需使能ADC_CR0.BGREN） 1011：内置温度传感器输出电压 （注意：需使能ADC_CR0.TSEN和ADC_CR0.BGREN）
11:8	PCHANNEL	RW	VCx 正端输入信号配置 0000：VCx_IN0 0100：VCx_IN4 0001：VCx_IN1 0101：VCx_IN5 0010：VCx_IN2 0110：VCx_IN6 0011：VCx_IN3 0111：VCx_IN7
7	WINDOW	RW	窗口比较功能配置 0：禁止窗口功能，VCx_OUTW = VCx_OUTP 1：使能窗口功能，VCx_OUTW = VC1_OUTP 异或 VC2_OUTP
6	POL	RW	VCx 输出信号极性设置 0：正端大于负端时，输出信号VCx_OUTA为高电平 1：正端大于负端时，输出信号VCx_OUTA为低电平
5	IE	RW	VCx 中断使能配置 0：禁止 1：使能
4:3	HYS	RW	VCx 迟滞窗口配置 00：没有迟滞 10：迟滞窗口大约20mV 01：迟滞窗口大约10mV 11：迟滞窗口大约30mV
2:1	RESP	RW	VCx 响应速度配置 00：极低速 10：中速 01：低速 11：高速 注：响应速度越快，功耗越大
0	EN	RW	VCx 使能控制 0：禁止 1：使能

VCx_CR1 控制寄存器 1

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:16	RFU	-	保留位，请保持默认值
15:13	BLANKTIME	RW	BLANK窗口持续时间配置 持续时间 =(2(BLANKTIME+2) + N) 个PCLK周期(N = 0~2)
12	BLANKCH3B	RW	ATIM的CH3B上升沿触发VCx启动BLANK窗口配置 0：禁止 1：使能
11	BLANKCH2B	RW	ATIM的CH2B上升沿触发VCx启动BLANK窗口配置 0：禁止 1：使能
10	BLANKCH1B	RW	ATIM的CH1B上升沿触发VCx启动BLANK窗口配置 0：禁止 1：使能
9	ATIMBK	RW	ATIM的BK刹车信号连接性配置 0：无功能 1：VCx输出连接到ATIM的BK刹车信号
8	ATIMCLR	RW	ATIM的OCREF_CLR信号连接性配置 0：无功能 1：VCx输出连接到ATIM的OCREF_CLR信号
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
4	FLTCLK	RW	数字滤波模块滤波时钟设置 0：内置RC120K振荡器时钟，其频率约120kHz 1：PCLK
3:1	FLTTIME	RW	数字滤波模块滤波时间配置 000：滤除宽度小于1个时钟周期的信号 001：滤除宽度小于3个时钟周期的信号 010：滤除宽度小于7个时钟周期的信号 011：滤除宽度小于15个时钟周期的信号 100：滤除宽度小于63个时钟周期的信号 101：滤除宽度小于255个时钟周期的信号

			110：滤除宽度小于1023个时钟周期的信号 111：滤除宽度小于4095个时钟周期的信号
0	FLTEN	RW	数字滤波模块使能配置 0：禁止 1：使能

VCx_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
31:8	RFU	-	保留位，请保持默认值
7	VIN	RW	电阻分压电路参考源设置 0：AVCC 1：ADC模块所配置的参考源（需使能ADC）
6	EN	RW	电阻分压电路使能控制 0：禁止 1：使能
5:0	DIV	RW	电阻分压电路输出电压配置 输出电压 = 参考源 × (1 + DIV) / 64

注意：VC1_DIV 和 VC2_DIV 指向同一实体寄存器，VC1 和 VC2 共用电阻分压器电路。

VCx_SR 状态寄存器

Address offset: 0x0C 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	READY	RO	VCx状态标志，通过读取此位判断VCx是否已经稳定 0：尚未稳定，不可以被使用 1：已经稳定，可以被使用 注意：若使能了VC，则进入DeepSleep前应查询等待该标志置1。
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 可编程电压检测器（LVD）

22.1 概述

LVD 可用于监测 AVCC 及芯片管脚的电压，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，用户可使用该信号处理一些紧急任务。

22.2 主要特性

- 4 路监测源，AVCC、LVD_IN1、LVD_IN2、LVD_IN3 管脚输入
- 16 阶阈值电压，范围 1.8V ~ 3.3V
- 8 种触发条件，高电平、上升沿、下降沿组合
- 2 种触发结果，复位、中断
- 8 阶滤波可配置
- 具备迟滞功能，强力抗干扰

22.3 功能描述

本节将详细描述低电压检测器的相关功能。

22.3.1 功能框图

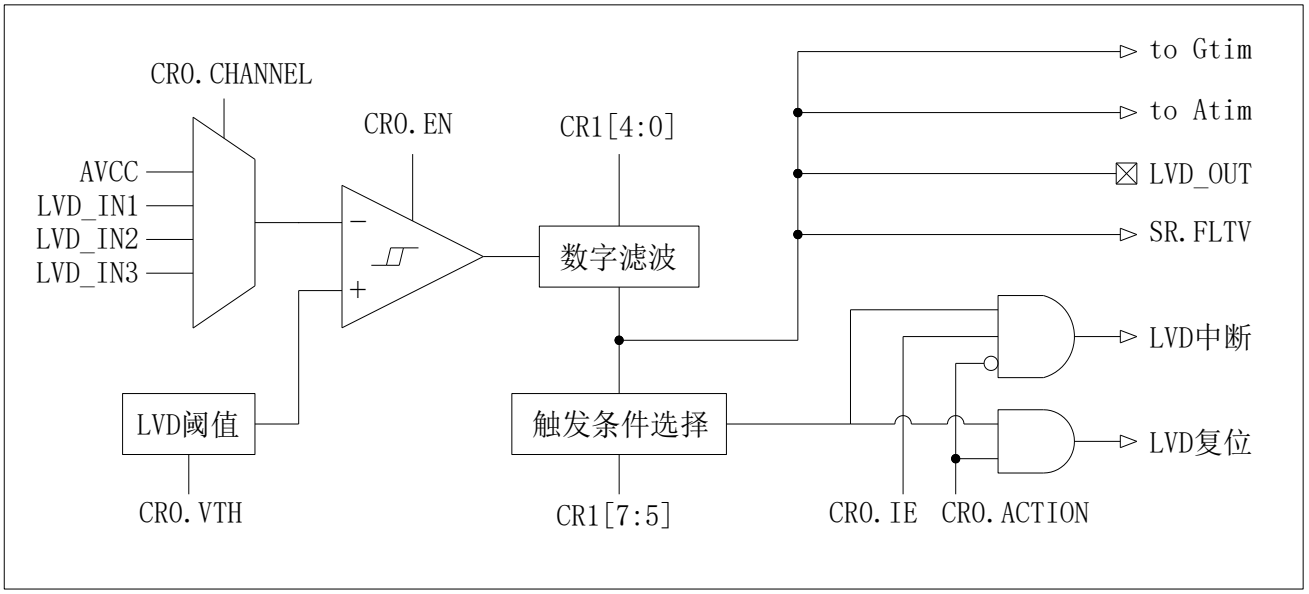

图 22-1 LVD 功能框图

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。

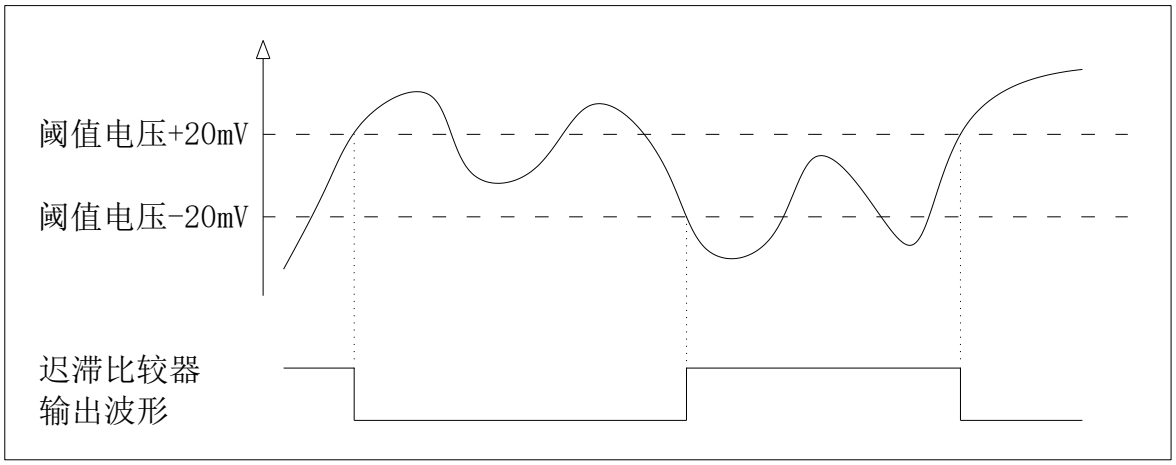

图 22-2 迟滞响应波形

22.3.3数字滤波

当设置 LVD_CR1.FLTEN 为 1 即可使能数字滤波功能。该滤波器可将迟滞比较器输出结果中小于滤波宽度的信号滤除，以增强系统鲁棒性。通过 LVD_CR1.FLTCLK 可选择 HSIOSC 或 RC120K 作为数字滤波器的滤波时钟；通过 LVD_CR1.FLTTIME 可配置滤波宽度为 1～4095 个滤波时钟的持续时间。注意，仅当 HSIOSC 时钟有效时，方可选择 HSIOSC 作为滤波时钟。

通过 LVD_SR.FLTV 位域，可实时读出经数字滤波后的信号电平；配置 GPIO 功能为 LVD_OUT 时，该管脚即可实时输出数字滤波后的信号波形以方便观察测量。

下图展示了滤波宽度为 1 时数字滤波器的输入与输出波形。

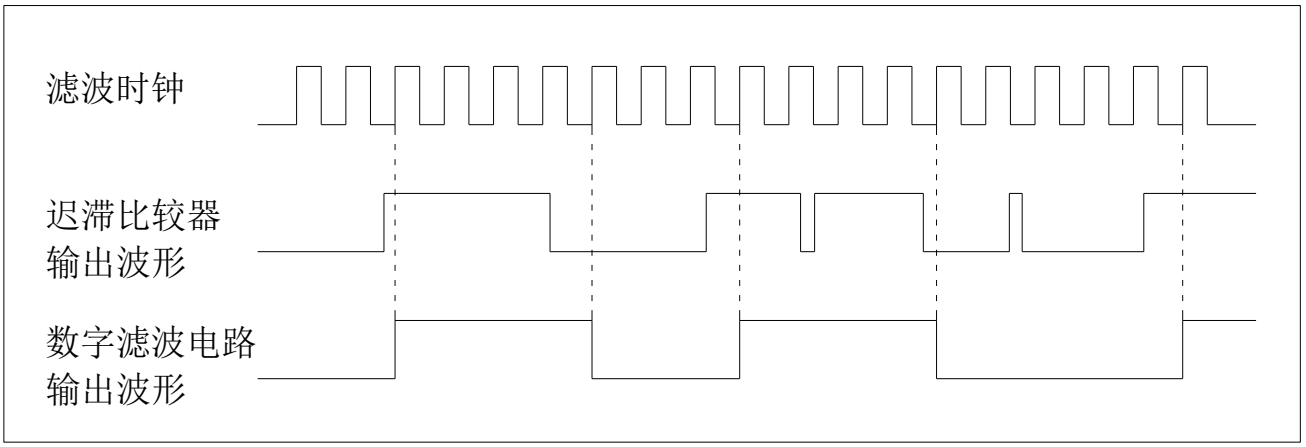

图 22-3 数字滤波输入输出波形

22.3.4触发条件

LVD 具有 3 种触发条件：被监测电压低于阈值电压、被监测电压跌落到阈值以下、被监测电压回升到阈值以上；详见 LVD_CR1[7:5] 位域的描述。用户可根据实际需要灵活配置使用何种条件触发动作。

22.3.5触发动作

LVD 具有 2 种触发动作：产生中断、产生复位；详见 LVD_CR0.ACTION 位域的描述。

- 当电压跌落时，需要处理紧急任务则建议选择产生中断。
- 当电压跌落时，需要防止 MCU 误动作则建议选择产生复位。

当 LVD 产生复位信号复位 MCU 时，LVD 自身的寄存器并不会被复位直到被监测电压上升到阈值电压以上。

22.4 编程示例

22.4.1 欠压复位示例

Step1. 配置 LVD_CR0.CHANNEL，选择监测目标；
Step2. 配置 LVD_CR0.VTH，设置阈值电压；
Step3. 配置 LVD_CR1.FLTCLK，选择滤波时钟；
Step4. 配置 LVD_CR1.FLTTIME，选择滤波时间；
Step5. 配置 LVD_CR1.FLTEN，使能 LVD 滤波；
Step6. 设置 LVD_CR1.LEVEL 为 1，选择被监测电压低于阈值时触发 LVD 动作；
Step7. 设置 LVD_CR0.ACTION 为 1，选择 LVD 触发动作为系统复位；
Step8. 设置 LVD_CR0.EN 为 1，使能 LVD。

22.4.2 电压变化中断示例

Step1. 配置 LVD_CR0.CHANNEL，选择监测目标；
Step2. 配置 LVD_CR0.VTH，选择阈值电压；
Step3. 配置 LVD_CR1.FLTCLK，选择滤波时钟；
Step4. 配置 LVD_CR1.FLTTIME，选择滤波时间；
Step5. 配置 LVD_CR1.FLTEN，使能 LVD 滤波；
Step6. 设置 LVD_CR1.RISE 和 FALL 均为 1，选择上升沿和下降沿触发；
Step7. 设置 LVD_CR0.ACTION 为 0，选择 LVD 触发动作为中断；
Step8. 设置 LVD_CR0.IE 为 1，使能 LVD 中断；
Step9. 使能 NVIC 中断向量表中的 LVD 中断；
Step10. 设置 LVD_CR0.EN 为 1，使能 LVD；
Step11. 在 LVD 中断服务程序中处理需要进行的任务，退出中断服务程序前应清除中断标志。

22.5 寄存器描述

寄存器列表

寄存器名称	寄存器地址	描述
LVD_CR0	0x4001 2A80 + 0x00	控制寄存器0
LVD_CR1	0x4001 2A80 + 0x04	控制寄存器1
LVD_SR	0x4001 2A80 + 0x08	状态寄存器

LVD_CR0 控制寄存器 0

Address offset: 0x00 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	IE	RW	中断使能控制 0：禁止LVD中断 1：使能LVD中断
8	RFU	-	保留位，请保持默认值
7:4	VTH	RW	阈值电压选择 0000：1.8V 1000：2.6V 0001：1.9V 1001：2.7V 0010：2.0V 1010：2.8V 0011：2.1V 1011：2.9V 0100：2.2V 1100：3.0V 0101：2.3V 1101：3.1V 0110：2.4V 1110：3.2V 0111：2.5V 1111：3.3V
3:2	CHANNEL	RW	监测输入通道配置 00：AVCC电源电压 01：LVD_IN1管脚电压 10：LVD_IN2管脚电压 11：LVD_IN3管脚电压
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

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	LEVEL	RW	被监测电压低于阈值电压触发 0：禁止 1：使能
6	FALL	RW	被监测电压跌落到阈值以下的下降沿触发 0：禁止 1：使能
5	RISE	RW	被监测电压回升到阈值以上的上升沿触发 0：禁止 1：使能
4	FLTCLK	RW	数字滤波模块滤波时钟设置 0：内置RC120K振荡器时钟，其频率约120kHz 1：HSIOSC，其频率约48MHz 注意：仅当HSISOC使能时，方可选择HSIOSC作为滤波时钟。
3:1	FLTTIME	RW	数字滤波模块滤波时间配置 000：滤除宽度小于1个时钟周期的信号 001：滤除宽度小于3个时钟周期的信号 010：滤除宽度小于7个时钟周期的信号 011：滤除宽度小于15个时钟周期的信号 100：滤除宽度小于63个时钟周期的信号 101：滤除宽度小于255个时钟周期的信号 110：滤除宽度小于1023个时钟周期的信号 111：滤除宽度小于4095个时钟周期的信号
0	FLTEN	RW	数字滤波模块使能配置 0：禁止 1：使能

LVD_SR 状态寄存器

Address offset: 0x08 Reset value: 0x0000 0000

位域	名称	复位值	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，被监测电压高于阈值电压 1：数字滤波器输出高电平，被监测电压低于阈值电压
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

本芯片的内核为 Cortex®-M0+，该内核包含用于高级调试功能的硬件扩展。调试扩展允许内核可以在取指（指令断点）或取访问数据（数据断点）时停止内核。内核停止时，可以查询内核的内部状态和系统的外部状态。查询完成后，将恢复内核和系统并恢复程序执行。
当调试主机与本芯片相连并进行调试时，将使用调试功能。

SWD 调试支持框图如下所示：

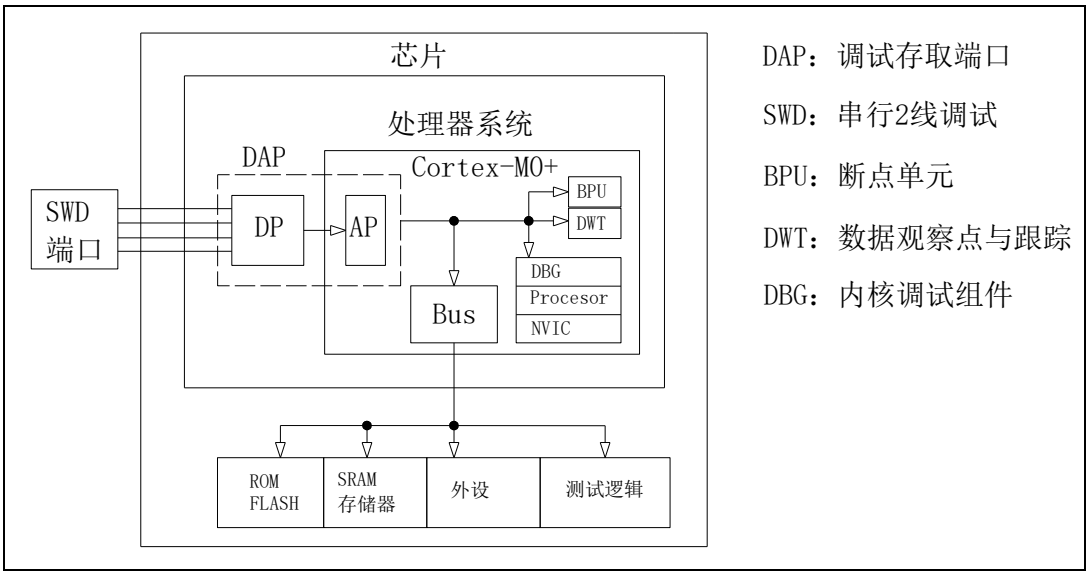

图 23-1 SWD 调试支持框图

23.2 Arm 参考文档

- Cortex®-M0+技术参考手册 (TRM)，可从 http://infocenter.arm.com 获取
- Arm 调试接口 V5
- Arm CoreSight 设计套件版本 r1p1 技术参考手册

23.3 SWD 端口管脚

本芯片的 SWD 接口管脚分配如下表所示，PA02/PA05 出厂时默认为 SWD 功能。

表 23-1 SWD 管脚分配

SWD端口名称	管脚功能	管脚分配
SWCLK	串行时钟输入	PA05
SWDIO	串行数据输入输出	PA02

PA02/PA05 管脚可配置为 SWD 功能或 GPIO 功能，由 SYSCTRL_CR2.SWDIO 位域进行功能配置。SWDIO 为 0，PA02/PA05 管脚被配置为 SWD 功能；SWDIO 为 1，则被配置为 GPIO 功能。

通常建议 SWD 管脚使用 100kΩ 上拉电阻，本芯片的 PA02/PA05 作为 SWD 功能时内置有上拉电阻，其阻值在 50kΩ - 200kΩ 之间，用户可以在外部增加上拉电阻，以提高抗干扰性能。

本芯片系列的 PA02/PA05 管脚默认为 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 管脚的配置与功能参见下表：

表 23-2 加密等级与 GPIO、SWD、ISP 功能

加密等级	SWDIO	GPIO	SWD	ISP
Level0	0	×	可擦可写可读	可擦可写可读
	1	√	不能连接	
Level1	0	×	可擦可写	可擦可写
	1	√	不能连接	
Level2	0	×	不能连接	可擦可写
	1	√	不能连接	
Level3	0	×	不能连接	不能连接
	1	√	不能连接	

注意：

加密等级可通过 ISP 协议进行设定，详见 FLASH 串行编程协议；

当芯片加密等级为 2 时，SWD 接口被禁止，只能通过 ISP 方式编程；

当芯片加密等级为 3 时，SWD、ISP 接口均被禁止，芯片无法再次烧录程序。

23.4 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。

23.4.1 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 010：WAIT 100：OK

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

23.4.2 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。

操作顺序如下：

Step1. 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；

Step2. 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；

Step3. 进入空闲状态后，对 DP-SW 的 ID CODE 寄存器执行读访问；

Step4. 按照包协议对需要访问的寄存器进行读写访问。

注意：如果不执行 Step3，则目标板在后续的包通信响应阶段，会发送 FAULT 响应。

23.4.3 SW-DP 和 SW-AP 的读写访问

- SW-DP/SW-AP 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。

23.4.4SW-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	详见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。 详见 DP SELECT寄存器位定义
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 寄存器位定义

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）

23.4.5SW-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），见下表:

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。

23.5 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，参见下表：

表 23-10 AP IDR 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

23.6 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，请参考《ARMv6-M Architecture Reference Manual》和 ARM® CoreSight 组件技术参考手册。

23.7 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，请参见《ARMv6-M Architecture Reference Manual》。

23.8 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_DEBUGEN 设置，可控制定时器、看门狗定时器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_DEBUGEN

位	名称	值	调试状态下断点期间计数器功能配置
10	WWDT	1	调试状态下，WWDT计数器暂停计数
		0	调试状态下，WWDT计数器正常计数
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
6	AWT	1	调试状态下，AWT计数器暂停计数
		0	调试状态下，AWT计数器正常计数
5	BTIM	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
4:2	RFU	-	保留位，请保持默认值
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。

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、RAM 容量、芯片管脚数量等信息，可通过 ISP、SWD 或 CPU 进行读取。数字签名相关信息在出厂时写入，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 UID 寄存器

芯片内置了两套唯一身份标识：长度为 80bit 的 LUID 和长度为 48bit 的 SUID。LUID 分散存储于 0x0010 084C - 0x0010 0850 / 0x0010 0854 / 0x0010 0858 / 0x0010 085C - 0x0010 085E；SUID 存储于 0x0010 07A0 - 0x001007A5。UID 在芯片生产时写入，用户无法修改。

唯一身份标识符典型应用场景：

- 用作设备序列号
- 在对 FLASH 进行编程时将 UID 加密后生成的密文写入 OTP/FLASH；在应用程序中采用相同的算法验证密文与 UID 的关系是否正确。
- 激活安全启动流程等

24.3 产品型号信息

产品型号信息记录在 0x0010 07D0 - 0x0010 07E1 地址，存储了产品型号的为 ASCII 码，最大长度为 18 字节。当产品型号长度不足时，在后方填充“00”。

例：“53594d3332463030334634503700000000000”代表“SYM32F003F4P7”。

24.4 FLASH 容量信息

FLASH 容量信息记录在 0x0010 07E4 -0x0010 07E7 地址，长度为 4 个字节。

例：0x0000 5000 代表 20KB。

24.5 RAM 容量信息

RAM 容量信息记录在 0x0010 07E8 -0x0010 07EB 地址，长度为 4 个字节。

例：0x0000 0C00 代表 3KB。

24.6 管脚数量信息

管脚数量信息记录在 0x0010 07E2 地址，长度为 1 个字节。

例：0x14 代表 20Pin，0x18 代表 24Pin。

附录 A SysTick 定时器

SysTick 定时器简介

OS 要想支持多任务，就需要周期执行上下文切换，这样就需要有定时器之类的硬件资源打断程序执行。当定时器中断产生时，处理器就会在异常处理中进行 OS 任务调度，同时还会进行 OS 维护的工作。Cortex-M0 处理器中有一个称为 SysTick 的简单定时器，用于产生周期性的中断请求。

SysTick 为 24 位的定时器，并且向下计数。定时器的计数减到 0 后，就会重新装载一个可编程的数值，并且同时产生 SysTick 异常（异常编号为 15），该异常事件会引起 SysTick 异常处理的执行，这个过程是 OS 的一部分。

对于不需要 OS 的系统，SysTick 定时器也可以用作其他用途，比如定时、计时或者为需要周期执行的任务提供中断源。SysTick 异常的产生是可控的，如果异常被禁止，仍然可以用轮询的方法使用 SysTick 定时器，比如检查当前的计数值或者轮询计数标志。

SysTick 配置示例

由于 SysTick 定时器的重载值和当前值在复位时都是未定义的，为了防止产生异常结果，对 SysTick 的配置需要遵循一定的流程：

Step1：配置 SysTick->CTRL.ENABLE 为 0，禁止 SysTick；

Step2：配置 SysTick->CTRL.CLKSOURCE，选择 SysTick 的时钟源；

Step3：配置 SysTick->LOAD，选择 SysTick 的溢出周期；

Step4：向 SysTick->VAL 写入任意值，清零 SysTick->VAL 及 SysTick->CTRL.COUNTFLAG；

Step5：配置 SysTick->CTRL.TICKINT 为 1，使能 SysTick 中断；

Step6：配置 SysTick->CTRL.ENABLE 为 1，使能 SysTick；

Step7：在中断服务程序中读取 SysTick->CTRL 以清除溢出标志。

寄存器描述

寄存器列表

寄存器名称	寄存器地址	寄存器描述
SYSTICK_CTRL	0xE000E010 + 0x00	SysTick控制和状态寄存器
SYSTICK_LOAD	0xE000E010 + 0x04	SysTick重装载寄存器
SYSTICK_VAL	0xE000E010 + 0x08	SysTick当前值寄存器
SYSTICK_CALIR	0xE000E010 + 0x0C	SysTick校准值寄存器

SYSTICK_CTRL 控制和状态寄存器

Address offset: 0x00　　Reset value: 0x0000 0000

位域	名称	权限	功能描述
31:17	Reserved	-	-
16	COUNTFLAG	RO	SysTick定时器溢出标志 0：SysTick定时器未发生溢出 1：SysTick定时器发生下溢出 读该寄存器，可清除COUNTFLAG标志
15:3	Reserved	-	-
2	CLKSOURCE	RW	SysTick时钟源选择 0：使用参考时钟4MHz，来自HSI分频 1：使用内核时钟HCLK
1	TICKINT	RW	SysTick定时器中断使能控制 0：禁止 1：使能
0	ENABLE	RW	SysTick定时器使能控制 0：禁止 1：使能

SYSTICK_LOAD 重载寄存器

Address offset: 0x04　　Reset value: 0xxxxx xxxx

位域	名称	权限	功能描述
31:24	Reserved	-	-
23:0	RELOAD	RW	SysTick 定时器重载值

SYSTICK_VAL 计数值寄存器

Address offset: 0x08 Reset value: 0xxxxx xxxx

位域	名称	权限	功能描述
31:24	Reserved	-	-
23:0	CURRENT	RW	读取该寄存器，获取SysTick定时器的当前计数值 写任意值到该寄存器，清零该寄存器及COUNTFLAG

SYSTICK_CALIB 校准值寄存器

Address offset: 0x0C Reset value: 0xxxxx xxxx

位域	名称	权限	功能描述
31	NOREF	RO	SysTick当前计数时钟标志 0：当前计数时钟为参考时钟 1：当前计数时钟为内核时钟
30	SKEW	RO	TENMS精度指示 0：TENMS值代表精确的10ms 1：TENMS值代表粗略的10ms
29:24	Reserved	-	
23:0	TENMS	RO	10毫秒校准值

附录 B 文档约定

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。
等宽英文	编程示例，汇编指令、伪代码和源代码示例。也用于正文中引用指令助记符、伪代码和源代码示例。
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。

寄存器协议

符号示例	描述
0x53CA	带‘0x’前缀的字符串表示十六进制数，数值为数字0～9或A～F大写英文
b00110	以b引导的0或1数字串表示二进制数
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寄存器的第7位~第5位
RW	可读可写，软件可以读写该位域
RO	只读，软件只能读取该位域
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

版本记录

版本	修订日期	修订说明
Beta 0.1	2021-03-12	内测版发布
Rev 1.0	2022-07-26	初版发布 由用户手册改为参考手册
Rev 1.01	2022-11-15	将CR2.DISCARD更正为CR1.DISCARD 将CR[9:8]更正为CR2[9:8] 修改笔误 修改VC章节的VCx_CR0中，WINDO位的对齐格式由两端对齐，改为左对齐 修改VC章节的CR0寄存器由CR0.NCHANNEL修改为CR0.INN 修改VC章节的CR0寄存器由CR0.PCHANNEL修改为CR0.INP 修改ATIM章节CR寄存，由CR.MODE修改为CR.ALIGNMENT 修改WWDT章节，编程示例中的step1步骤描述错误 修改VC章节的CR0寄存器由CR0.INN修改为CR0.NCHANNEL 修改VC章节的CR0寄存器由CR0.INP修改为CR0.PCHANNEL 修改BTIM章节的门控模式小结，添加“有效时” 修改FLASH章节的读保护小结中添加“读保护等级可通过驱动库函数进行设置，具体设置方式请参阅驱动库函数说明。” 修改“电源控制与功耗”章节“进入休眠模式或深入休眠模式”小结的AWT的ETR在深度休眠模式下不能正常计数 AWT章节的计数功能小结中添加AWT_ETR注意事项 更新BTIM计数器模式框图 删除ADC章节的ADC_CR1_DMAEN位 删除ATIM章节的ATIM_CH1CR_CDEB、CDEA位 删除UART章节的UART_ICR_MATCH位 存储器操作章节，增加HSI相关说明 【设置CRC_CR为4】更改为【设置CRC_CR为6】
Rev 1.02	2023-02-09	修改ADC相关描述 修改GPIO中断小节描述文字 修改BTIM计数器模式框图 修改SYSCTRL_CR2寄存器描述
Rev 1.03	2023-02-22	在GPIO小节的端口电路图下方增加IO输入电压相关描述
Rev 1.04	2023-04-06	修改SPI功能描述中从机选择信号、状态标志、错误标志的描述
Rev 1.05	2023-05-26	修改GTIM编码计数模式中关于溢出中断的描述 更新GTIM编码计数电路示意图 添加IWDT_ARR最小值注意事项 添加WWDT_CR0.WCNT写操作注意事项 修改UART章节描述中Baud为BaudRate
Rev 1.06	2023-07-22	增加UART波特率发生方式表中的备注说明：OVER非0时需保持BRRF为0 修改IWDT章节窗口看门狗模式的描述，修改IWDT章节窗口看门

		狗模式的编程示例 修改IDCODE的错误 修正一些笔误
Rev 1.07	2023-09-06	VC迟滞功能小节增加描述性图文 修改串口发送时序图 修正UARTx_CR2.SIGNAL为UARTx_CR2.SINGLE 修正一些笔误
Rev 1.08	2023-10-20	修正SPI延迟采样时序图 FLASH章节添加描述，FLASH_CR1.BUSY标志清0前不可以进入DeepSleep状态 工作模式章节添加进入DeepSleep的注意事项 修正AWT章节描述中ICR.OV为ICR.UD 修改LVD触发动作的文字描述 修正ADC_ISR寄存器模拟看门狗“转换结果位于”为“转换结果的高12Bit位于” 修正一些笔误
Rev 1.09	2024-04-02	AWT模块，增加自动校准HSI、LSI相关描述 修改WWDT章节寄存器相关描述 修改UART章节特性相关描述，修正UART同步模式时序图