

## 基于 ARM®32 位的 Cortex™-M4 微控制器，带 64 K 字节内部闪存 sLib、10 个定时器、1 个 ADC、1 个比较器、6 个通信接口功能

- 内核：ARM®32位的Cortex™-M4 CPU
  - 最高120 MHz工作频率，带存储器保护单元(MPU)，内建单周期乘法和硬件除法
  - 具有DSP指令集
- 存储器
  - 16 K字节至64 K字节的闪存程序/数据存储器
  - 4 K字节的系统存储器可作启动加载程序(Bootloader)用外，也可一次性配置成一般用户程序和数据区
  - 16 K字节的SRAM
  - sLib：将指定之主存储区设为执行代码安全库区，此区代码仅能调用无法读取
- 时钟、复位和电源管理
  - 2.4至3.6伏供电和I/O引脚
  - 上电/断电复位(POR/PDR)、可编程电压监测器(PVD)
  - 4至25 MHz晶体振荡器
    - 内嵌经出厂校准的48 MHz RC振荡器(25 °C达1 %精度，-40 °C至+105 °C达2.5 %精度)
    - 内嵌带校准的40 kHz RC振荡器
    - 带校准功能的32 kHz晶体振荡器
    - PLL可灵活配置31~500倍频和1~15分频系数
- 低功耗
  - 睡眠、停机、和待机模式
- 1个12位A/D转换器，0.5 μs转换时间(多达16个输入通道)
  - 转换范围：0至3.6 V
  - 一组采样和保持功能
  - 温度传感器
- 1个比较器
- DMA：5通道DMA控制器
  - 支持的外设：定时器、ADC、I<sup>2</sup>S、SPI、I<sup>2</sup>C、和USART
- 调试模式
  - 串行单线调试(SWD)接口
- 多达39个快速I/O端口
  - 39个多功能双向的I/O口，所有I/O口可以映像到16个外部中断；几乎所有I/O口可容忍5V输入信号
  - 所有I/O口均为快速I/O，寄存器存取速度最高
- 多达10个定时器
  - 多达5个16位定时器，每个定时器有多达4个用于输入捕获/输出比较/PWM或脉冲计数的通道和增量编码器输入
  - 1个16位带死区控制和紧急刹车，用于电机控制的PWM高级控制定时器
  - 1个基本定时器
  - 2个看门狗定时器(独立的和窗口型的)
  - 系统时间定时器：24位自减型计数器
- ERTC：增强型RTC，具亚秒级精度及硬件日历
- 多达6个通信接口
  - 2个I<sup>2</sup>C接口(支持SMBus/PMBus)
  - 2个USART接口(支持ISO7816, LIN, IrDA接口和调制解调控制)
  - 2个SPI接口(36 M位/秒)，2个均可复用为I<sup>2</sup>S接口
- CRC计算单元
- 96位的芯片唯一代码
- 封装
  - LQFP48 7 x 7 mm
  - LQFP32 7 x 7 mm
  - QFN32 5 x 5 mm
  - QFN32 4 x 4 mm
  - QFN28 4 x 4 mm
  - TSSOP20 6.5 x 4.4 mm

表1. 选型列表

闪存存储器	型号
64 K字节	AT32F421C8T7 AT32F421K8T7 AT32F421K8U7 AT32F421K8U7-4 AT32F421F8P7 AT32F421G8U7
32 K字节	AT32F421C6T7 AT32F421K6T7 AT32F421K6U7 AT32F421K6U7-4 AT32F421F6P7 AT32F421G6U7
16 K字节	AT32F421C4T7 AT32F421K4T7 AT32F421K4U7 AT32F421K4U7-4 AT32F421F4P7 AT32F421G4U7

## 目 录

1 系统架构 ( SYSTEM ) .....	37
1.1 系统概述 .....	37
1.1.1 总线架构 .....	39
1.1.2 ARM Cortex™-M4处理器 .....	39
1.2 地址映射 .....	40
1.2.1 寄存器映像 .....	42
1.2.2 位绑定 .....	44
1.2.3 片上SRAM .....	44
1.2.4 片上Flash .....	44
1.3 引导配置 .....	47
1.4 器件特征信息 .....	47
1.4.1 寄存器的缩写说明 .....	47
1.4.2 闪存容量寄存器 .....	48
1.4.3 器件电子签名 .....	48
2 电源控制 ( PWR ) .....	50
2.1 简介 .....	50
2.2 主要特点 .....	50
2.3 功能描述 .....	50
2.3.1 电源域 .....	50
2.3.1.1 VDD/VDDA电源域 .....	51
2.3.1.2 内核电源域 .....	52
2.3.2 低功耗模式 .....	53
2.3.2.1 睡眠模式 .....	53
2.3.2.2 停机模式 .....	54
2.3.2.3 待机模式 .....	55
2.3.2.4 调试模式 .....	56

2.3.3 自动唤醒 .....	56
2.4 PWR寄存器 .....	56
2.4.1 电源控制寄存器 ( PWR_CTRL ) .....	57
2.4.2 电源控制/状态寄存器 ( PWR_CTRLSTS ) .....	58
2.4.3 电源控制寄存器2 ( PWR_CTRL2 ) .....	59
3 复位和时钟控制 ( RCC ) .....	60
3.1 复位 .....	60
3.1.1 系统复位 .....	60
3.1.2 电源复位 .....	60
3.1.3 备份域复位 .....	61
3.2 时钟 .....	61
3.2.1 HSE时钟 .....	63
3.2.2 HSI时钟 .....	63
3.2.3 PLL.....	64
3.2.4 LSE时钟 .....	64
3.2.5 LSI时钟 .....	65
3.2.6 系统时钟 ( SYSCLK ) 选择.....	65
3.2.7 时钟失效检测 ( CFD ) .....	65
3.2.8 ERTC时钟 .....	65
3.2.9 看门狗时钟 .....	66
3.2.10 时钟输出.....	66
3.3 RCC寄存器描述 .....	66
3.3.1 时钟控制寄存器 ( RCC_CTRL ) .....	68
3.3.2 时钟配置寄存器 ( RCC_CFG ) .....	69
3.3.3 时钟中断寄存器 ( RCC_CLKINT ) .....	71
3.3.4 APB2外设复位寄存器 ( RCC_APB2RST ) .....	73
3.3.5 APB1外设复位寄存器 ( RCC_APB1RST ) .....	74

3.3.6 AHB外设时钟使能寄存器 ( RCC_AHBEN ) .....	76
3.3.7 APB2外设时钟使能寄存器 ( RCC_APB2EN ) .....	77
3.3.8 APB1外设时钟使能寄存器 ( RCC_APB1EN ) .....	78
3.3.9 备份域控制寄存器 ( RCC_BDC ) .....	79
3.3.10 控制/状态寄存器 ( RCC_CTRLSTS ) .....	80
3.3.11 AHB外设复位寄存器 ( RCC_AHBRST ) .....	81
3.3.12 PLL配置寄存器 ( RCC_PLL ) .....	82
3.3.13 额外寄存器 ( RCC_MISC ) .....	83
3.3.14 额外寄存器 ( RCC_MISC2 ) .....	84
4 内嵌闪存控制器 ( EFC ) .....	86
4.1 EFC简介 .....	86
4.2 主要特点 .....	86
4.2.1 闪存模块组织 .....	86
4.3 功能描述 .....	89
4.3.1 读操作 .....	89
4.3.1.1 取指令 .....	89
4.3.1.2 D-Code接口 .....	89
4.3.1.3 闪存访问控制器 .....	90
4.3.2 闪存编程和擦除控制器 ( FPEC ) .....	90
4.3.2.1 键值 .....	90
4.3.2.2 解除闪存锁 .....	90
4.3.2.3 主闪存编程 .....	90
4.3.2.4 闪存擦除 .....	91
4.3.2.5 选择字节编程 .....	93
4.3.3 保护 .....	94
4.3.3.1 写保护 .....	94
4.3.3.2 读保护 .....	94

4.3.3.3 选择字节块写保护 .....	95
4.3.4 选择字节说明 .....	96
4.3.5 特殊功能 .....	97
4.3.5.1 安全库区设定 .....	97
4.3.5.2 系统存储器区域作为主存扩展使用 .....	98
4.3.5.3 CRC校验 .....	98
4.4 EFC寄存器 .....	99
4.4.1 闪存访问控制寄存器 ( FLASH_ACR ) .....	100
4.4.2 FPEC键寄存器 ( FLASH_FCKEY ) .....	101
4.4.3 闪存OPTKEY寄存器 ( FLASH_OPTKEYR ) .....	102
4.4.4 闪存状态寄存器 ( FLASH_STS ) .....	102
4.4.5 闪存控制寄存器 ( FLASH_CTRL ) .....	103
4.4.6 闪存地址寄存器 ( FLASH_ADDR ) .....	104
4.4.7 选择字节寄存器 ( FLASH_UOB ) .....	104
4.4.8 写保护寄存器 ( FLASH_WRPRT ) .....	105
4.4.9 闪存安全库区状态寄存器0 ( FLASH_CDR0 ) .....	105
4.4.10 闪存安全库区状态寄存器1 ( FLASH_CDR1 ) .....	106
4.4.11 闪存安全库区密码寄存器 ( FSLIB_PSW ) .....	107
4.4.12 闪存安全库区密码设定状态寄存器 ( FSLIB_PSW_STS ) .....	107
4.4.13 闪存CRC校验起始位置 ( FLASH_CRC_AR ) .....	108
4.4.14 闪存CRC校验控制寄存器 ( FLASH_CRC_CTRL ) .....	108
4.4.15 闪存CRC校验结果寄存器 ( FLASH_CRC_OUTR ) .....	109
4.4.16 闪存安全库区密码设定寄存器 ( FSLIB_SET_PSW ) .....	109
4.4.17 闪存安全库区范围设定寄存器 ( FSLIB_SET_RANGE ) .....	110
4.4.18 闪存主存扩展区安全库区设定寄存器 ( SYS_MEM_SLIB_SET ) .....	110
4.4.19 闪存主存扩展区模式寄存器 ( SYS_MEM_BOOT_DIS_SET ) .....	110

4.4.20 闪存安全库区键寄存器 ( FSLIB_KEYR ) .....	111
5 CRC计算单元 ( CRC ) .....	112
5.1 CRC简介 .....	112
5.2 CRC主要特性 .....	112
5.3 CRC功能描述 .....	112
5.4 CRC寄存器 .....	112
5.4.1 数据寄存器 ( CRC_DR ) .....	113
5.4.2 独立数据寄存器 ( CRC_IDR ) .....	113
5.4.3 控制寄存器 ( CRC_CTRL ) .....	113
6 通用功能I/O ( GPIO ) .....	115
6.1 简介 .....	115
6.2 主要特征 .....	115
6.3 功能描述 .....	115
6.3.1 通用I/O(GPIO) .....	117
6.3.2 I/O引脚的复用功能和重映射 .....	117
6.3.3 I/O端口控制寄存器 .....	118
6.3.4 I/O端口数据寄存器 .....	118
6.3.5 I/O端口数据位处理 .....	118
6.3.6 GPIO锁定机制 .....	118
6.3.7 I/O复用功能输入/输出 .....	119
6.3.8 外部中断/唤醒线 .....	120
6.3.9 输入配置 .....	120
6.3.10 模拟输入配置 .....	121
6.3.11 输出配置 .....	122
6.3.12 复用功能 ( AF ) .....	123
6.3.13 把OSC32_IN/OSC32_OUT作为GPIO端口PC14/PC15.....	124
6.3.14 把OSC_IN/OSC_OUT作为GPIO端口PF0/PF1 .....	124

6.3.15 备份域供电下GPIO引脚的使用 .....	124
6.4 GPIO寄存器 .....	124
6.4.1 端口配置低寄存器 ( GPIOx_MODER ) ( x=A..C,F ) .....	126
6.4.2 端口输出类型寄存器 ( GPIOx_OTYPER ) ( x=A..C,F ) .....	126
6.4.3 电流推动/吸入能力切换控制寄存器 ( GPIOx_ODRVR ) ( x=A..C,F ) .....	126
6.4.4 端口上拉/下拉寄存器 ( GPIOx_PUPDR ) ( x=A..C,F ) .....	127
6.4.5 端口输入数据寄存器 ( GPIOx_IPTDT ) ( x=A..C,F ) .....	127
6.4.6 端口输出数据寄存器 ( GPIOx_OPTDT ) ( x=A..C,F ) .....	128
6.4.7 端口位设置/清除寄存器 ( GPIOx_BSRE ) ( x=A..C,F ) .....	128
6.4.8 端口配置锁定寄存器 ( GPIOx_LOCK ) ( x=A..C,F ) .....	128
6.4.9 端口复用功能低位寄存器 ( GPIOx_AFRL ) ( x=A..C,F ) .....	129
6.4.10 端口复用功能高位寄存器 ( GPIOx_AFRH ) ( x=A..C,F ) .....	129
6.4.11 端口位清除寄存器 ( GPIOx_BRE ) ( x=A..C,F ) .....	130
6.4.12 极大电流推动/吸入能力切换控制寄存器 ( GPIOx_HDRV ) ( x=A..C,F ) .....	130
6.4.13 电压转换速率切换控制寄存器 ( GPIOx_SRCTR ) ( x=A..C,F ) .....	131
7 系统配置控制器 ( SYSCFG ) .....	132
7.1 简介 .....	132
7.2 SYSCFG寄存器 .....	132
7.2.1 SYSCFG配置寄存器1 ( SYSCFG_CFGR1 ) .....	132
7.2.2 SYSCFG外部中断配置寄存器1 ( SYSCFG_EXTIC1 ) .....	133
7.2.3 SYSCFG外部中断配置寄存器2 ( SYSCFG_EXTIC2 ) .....	134
7.2.4 SYSCFG外部中断配置寄存器3 ( SYSCFG_EXTIC3 ) .....	134
7.2.5 SYSCFG外部中断配置寄存器4 ( SYSCFG_EXTIC4 ) .....	134
8 中断和事件 ( INT/EVENT ) .....	136
8.1 嵌套向量中断控制器 .....	136
8.1.1 系统嘀嗒 ( SysCNTRick ) 校准值寄存器 .....	136
8.1.2 中断和异常向量 .....	136

8.2 外部中断/事件控制器 ( EXTI ) .....	137
8.2.1 主要特性 .....	138
8.2.2 框图 .....	138
8.2.3 唤醒事件管理 .....	138
8.2.4 功能说明 .....	138
8.2.5 外部中断/事件线路映像 .....	139
8.3 EXTI寄存器描述 .....	140
8.3.1 中断屏蔽寄存器 ( EXTI_INTEN ) .....	141
8.3.2 事件屏蔽寄存器 ( EXTI_EVTEN ) .....	142
8.3.3 上升沿触发选择寄存器 ( EXTI_RTRSEL ) .....	142
8.3.4 下降沿触发选择寄存器 ( EXTI_FTRSEL ) .....	143
8.3.5 软件中断事件寄存器 ( EXTI_SWIE ) .....	143
8.3.6 挂起寄存器 ( EXTI_PND ) .....	144
9 DMA控制器 ( DMA ) .....	145
9.1 DMA简介 .....	145
9.2 DMA主要特性 .....	145
9.3 功能描述 .....	146
9.3.1 DMA处理 .....	146
9.3.2 仲裁器 .....	146
9.3.3 DMA通道 .....	147
9.3.4 可编程的数据传输宽度、对齐方式和数据大小端 .....	148
9.3.5 错误管理 .....	149
9.3.6 中断 .....	149
9.3.7 DMA请求映像 .....	149
9.4 DMA寄存器 .....	151
9.4.1 DMA中断状态寄存器 ( DMA_ISTS ) .....	153
9.4.2 DMA中断标志清除寄存器 ( DMA_ICLR ) .....	153

9.4.3 DMA通道x配置寄存器 ( DMA_CHCTRLx ) ( x = 1...5 ) .....	154
9.4.4 DMA通道x传输数量寄存器 ( DMA_TCNTx ) ( x = 1...5 ) .....	155
9.4.5 DMA通道x外设地址寄存器 ( DMA_CPBAX ) ( x = 1...5 ) .....	156
9.4.6 DMA通道x存储器地址寄存器 ( DMA_CMBAx ) ( x = 1...5 ) .....	156
10 定时器 ( TIMER ) .....	157
10.1 基本定时器 ( TMR6 ) .....	157
10.1.1 TRM6简介 .....	157
10.1.2 TRM6的主要特性 .....	157
10.1.3 TRM6的功能 .....	157
10.1.3.1 时基单元 .....	157
10.1.3.2 预分频器 .....	158
10.1.3.3 计数模式 .....	159
10.1.3.4 时钟源 .....	162
10.1.3.5 调试模式 .....	162
10.1.4 TRM6寄存器 .....	162
10.1.4.1 TMR6控制寄存器1 ( TMRx_CTRL1 ) .....	163
10.1.4.2 TMR6控制寄存器2 ( TMRx_CTRL2 ) .....	164
10.1.4.3 TMR6 DMA中断使能寄存器 ( TMRx_DIE ) .....	164
10.1.4.4 TMR6状态寄存器 ( TMRx_STS ) .....	165
10.1.4.5 TMR6事件产生寄存器 ( TMRx_EVEG ) .....	165
10.1.4.6 TMR6计数器 ( TMRx_CNT ) .....	165
10.1.4.7 TMR6预分频器 ( TMRx_DIV ) .....	166
10.1.4.8 TMR6自动重装载寄存器 ( TMRx_AR ) .....	166
10.2 通用定时器 ( TMR3 ) .....	166
10.2.1 TMRx简介 .....	166
10.2.2 TMRx主要功能 .....	166
10.2.3 TMRx功能描述 .....	167

10.2.3.1 时基单元 .....	167
10.2.3.2 计数器模式 .....	168
10.2.3.3 时钟选择 .....	176
10.2.3.4 捕获/比较通道 .....	179
10.2.3.5 输入捕获模式 .....	180
10.2.3.6 PWM输入模式 .....	181
10.2.3.7 强置输出模式 .....	182
10.2.3.8 输出比较模式 .....	182
10.2.3.9 PWM模式 .....	183
10.2.3.10 单脉冲模式 .....	185
10.2.3.11 在外部事件时清除OCxREF信号 .....	186
10.2.3.12 编码器接口模式 .....	187
10.2.3.13 定时器输入异或功能 .....	189
10.2.3.14 定时器和外部触发的同步 .....	189
10.2.3.15 定时器同步 .....	191
10.2.3.16 调试模式 .....	196
10.2.4 TMRx寄存器描述 .....	196
10.2.4.1 控制寄存器1 ( TMRx_CTRL1 ) .....	198
10.2.4.2 控制寄存器2 ( TMRx_CTRL2 ) .....	199
10.2.4.3 从模式控制寄存器 ( TMRx_SMC ) .....	200
10.2.4.4 DMA/中断使能寄存器 ( TMRx_DIE ) .....	201
10.2.4.5 状态寄存器 ( TMRx_STS ) .....	203
10.2.4.6 事件产生寄存器 ( TMRx_EVEG ) .....	204
10.2.4.7 捕获/比较模式寄存器1 ( TMRx_CCM1 ) .....	204
10.2.4.8 捕获/比较模式寄存器2 ( TMRx_CCM2 ) .....	207
10.2.4.9 捕获/比较使能寄存器 ( TMRx_CCE ) .....	208

10.2.4.10 计数器 ( TMRx_CNT ) .....	209
10.2.4.11 预分频器 ( TMRx_DIV ) .....	209
10.2.4.12 自动重装载寄存器 ( TMRx_AR ) .....	210
10.2.4.13 捕获/比较寄存器1 ( TMRx_CC1 ) .....	210
10.2.4.14 捕获/比较寄存器2 ( TMRx_CC2 ) .....	211
10.2.4.15 捕获/比较寄存器3 ( TMRx_CC3 ) .....	211
10.2.4.16 捕获/比较寄存器4 ( TMRx_CC4 ) .....	212
10.2.4.17 DMA控制寄存器 ( TMRx_DMAC ) .....	212
10.2.4.18 连续模式的DMA地址 ( TMRx_DMABA ) .....	213
10.3 通用定时器 ( TMR14 ) .....	214
10.3.1 TMRx简介 .....	214
10.3.2 TMRx主要功能 .....	214
10.3.2.1 TMR14主要功能 .....	214
10.3.3 TMRx功能描述 .....	215
10.3.3.1 时基单元 .....	215
10.3.3.2 计数器模式 .....	216
10.3.3.3 时钟选择 .....	219
10.3.3.4 捕获/比较通道 .....	219
10.3.3.5 输入捕获模式 .....	221
10.3.3.6 强置输出模式 .....	221
10.3.3.7 输出比较模式 .....	222
10.3.3.8 PWM模式 .....	222
10.3.3.9 调试模式 .....	223
10.3.4 TMR14寄存器描述 .....	223
10.3.4.1 控制寄存器 1 ( TMRx_CTRL1 ) .....	224
10.3.4.2 DMA/中断使能寄存器 ( TMRx_DIE ) .....	225

10.3.4.3 状态寄存器 ( TMRx_STS ) .....	226
10.3.4.4 事件产生寄存器 ( TMRx_EVEG ) .....	226
10.3.4.5 捕获/比较模式寄存器1 ( TMRx_CCM1 ) .....	227
10.3.4.6 捕获/比较使能寄存器 ( TMRx_CCE ) .....	228
10.3.4.7 计数器 ( TMRx_CNT ) .....	229
10.3.4.8 预分频器 ( TMRx_DIV ) .....	229
10.3.4.9 自动重装载寄存器 ( TMRx_AR ) .....	230
10.3.4.10 捕获/比较寄存器1 ( TMRx_CC1 ) .....	230
10.3.4.11 通道输入重映射寄存器 ( TMRx_RMP ) .....	230
10.4 通用定时器 ( TRM15 ) .....	232
10.4.1 TMR15简介 .....	232
10.4.2 TMR15主要特性 .....	232
10.4.3 TMR15功能描述 .....	233
10.4.3.1 时基单元 .....	233
10.4.3.2 计数器模式 .....	234
10.4.3.3 重复计数器 .....	237
10.4.3.4 时钟选择 .....	238
10.4.3.5 捕获/比较通道 .....	240
10.4.3.6 输入捕获模式 .....	242
10.4.3.7 PWM输入模式 .....	242
10.4.3.8 强制输出模式 .....	243
10.4.3.9 输出比较模式 .....	243
10.4.3.10 PWM模式 .....	244
10.4.3.11 互补输出和死区插入 .....	245
10.4.3.12 使用刹车功能 .....	246
10.4.3.13 单脉冲模式 .....	248

10.4.3.14 TMRx定时器和外部触发的同步 .....	249
10.4.3.15 定时器同步 .....	251
10.4.3.16 调试模式 .....	251
10.4.4 TMR15寄存器描述 .....	251
10.4.4.1 TMR15控制寄存器1 ( TMR15_CTRL1 ) .....	253
10.4.4.2 TMR15控制寄存器2 ( TMR15_CTRL2 ) .....	254
10.4.4.3 TMR15从模式控制寄存器 ( TMR15_SMC ) .....	255
10.4.4.4 TMR15 DMA/中断使能寄存器 ( TMR15_DIE ) .....	256
10.4.4.5 TMR15状态寄存器 ( TMR15_STS ) .....	257
10.4.4.6 TMR15事件产生寄存器 ( TMR15_EVEG ) .....	258
10.4.4.7 TMR15捕获/比较模式寄存器1 ( TMR15_CCM1 ) .....	259
10.4.4.8 TMR15捕获/比较使能寄存器 ( TMR15_CCE ) .....	261
10.4.4.9 TMR15计数器 ( TMR15_CNT ) .....	263
10.4.4.10 TMR15预分频器 ( TMR15_DIV ) .....	263
10.4.4.11 TMR15自动重装载寄存器 ( TMR15_AR ) .....	264
10.4.4.12 TMR15重复计数寄存器 ( TMR15_RC ) .....	264
10.4.4.13 TMR15捕获/比较寄存器 1 ( TMR15_CC1 ) .....	264
10.4.4.14 TMR15捕获/比较寄存器2 ( TMR15_CC2 ) .....	265
10.4.4.15 TMR15刹车和死区寄存器 ( TMR15_BRKDT ) .....	265
10.4.4.16 TMR15 DMA控制寄存器 ( TMR15_DMAC ) .....	267
10.4.4.17 TMR15连续模式的DMA地址 ( TMR15_DMABA ) .....	267
10.5 通用定时器 ( TMR16/17 ) .....	267
10.5.1 TMRx简介 .....	268
10.5.2 TMRx主要特性 .....	268
10.5.3 TMRx功能描述 .....	269
10.5.3.1 时基单元 .....	269

10.5.3.2 计数器模式 .....	270
10.5.3.3 重复计数器 .....	273
10.5.3.4 时钟选择 .....	274
10.5.3.5 捕获/比较通道 .....	275
10.5.3.6 输入捕获模式 .....	276
10.5.3.7 强置输出模式 .....	277
10.5.3.8 输出比较模式 .....	277
10.5.3.9 PWM模式 .....	278
10.5.3.10 互补输出和死区插入 .....	279
10.5.3.11 使用刹车功能 .....	280
10.5.3.12 单脉冲模式 .....	282
10.5.3.13 调试模式 .....	283
10.5.4 TMRx寄存器描述 .....	283
10.5.4.1 TMRx控制寄存器1 ( TMRx_CTRL1 ) ( x=16 , 17 ) .....	284
10.5.4.2 TMRx控制寄存器2 ( TMRx_CTRL2 ) ( x=16 , 17 ) .....	285
10.5.4.3 TMRx DMA/中断使能寄存器 ( TMRx_DIE ) ( x=16 , 17 ) .....	286
10.5.4.4 TMRx状态寄存器 ( TMRx_STS ) ( x=16 , 17 ) .....	286
10.5.4.5 TMRx事件产生寄存器 ( TMRx_EVEG ) ( x=16 , 17 ) .....	287
10.5.4.6 TMRx捕获/比较模式寄存器1 ( TMRx_CCM1 ) ( x=16 , 17 ) .....	288
10.5.4.7 TMRx捕获/比较使能寄存器 ( TMRx_CCE ) ( x=16 , 17 ) .....	290
10.5.4.8 TMRx计数器 ( TMRx_CNT ) ( x=16 , 17 ) .....	292
10.5.4.9 TMRx预分频器 ( TMRx_DIV ) ( x=16 , 17 ) .....	292
10.5.4.10 TMRx自动重装载寄存器 ( TMRx_AR ) ( x=16 , 17 ) .....	292
10.5.4.11 TMRx重复计数寄存器 ( TMRx_RC ) ( x=16 , 17 ) .....	293
10.5.4.12 TMRx捕获/比较寄存器1 ( TMRx_CC1 ) ( x=16 , 17 ) .....	293
10.5.4.13 TMRx刹车和死区寄存器 ( TMRx_BRKDT ) ( x=16 , 17 ) .....	293

10.5.4.14 TMRx DMA控制寄存器 ( TMRx_DMAC ) ( x=16 , 17 ) .....	295
10.5.4.15 TMRx连续模式的DMA地址 ( TMRx_DMABA ) ( x=16 , 17 ) .....	295
10.6 高级控制定时器 ( TMR1 ) .....	296
10.6.1 TMR1简介 .....	296
10.6.2 TMR1主要特性 .....	296
10.6.3 TMR1功能描述 .....	297
10.6.3.1 时基单元 .....	297
10.6.3.2 计数器模式 .....	298
10.6.3.3 重复计数器 .....	306
10.6.3.4 时钟选择 .....	307
10.6.3.5 捕获/比较通道 .....	310
10.6.3.6 输入捕获模式 .....	311
10.6.3.7 PWM输入模式 .....	312
10.6.3.8 强置输出模式 .....	313
10.6.3.9 输出比较模式 .....	313
10.6.3.10 PWM模式 .....	314
10.6.3.11 互补输出和死区插入 .....	316
10.6.3.12 使用刹车功能 .....	318
10.6.3.13 在外部事件时清除OCxREF信号 .....	319
10.6.3.14 产生六步PWM输出 .....	320
10.6.3.15 单脉冲模式 .....	321
10.6.3.16 编码器接口模式 .....	322
10.6.3.17 定时器输入异或功能 .....	324
10.6.3.18 与霍尔传感器的接口 .....	324
10.6.3.19 TMRx定时器和外部触发的同步 .....	326
10.6.3.20 定时器同步 .....	329

10.6.3.21 调试模式 .....	329
10.6.4 TMR1寄存器描述 .....	329
10.6.4.1 TMR1控制寄存器1 ( TMRx_CTRL1 ) .....	331
10.6.4.2 TMR1控制寄存器2 ( TMRx_CTRL2 ) .....	332
10.6.4.3 TMR1从模式控制寄存器 ( TMRx_SMC ) .....	333
10.6.4.4 TMR1 DMA/中断使能寄存器 ( TMRx_DIE ) .....	335
10.6.4.5 TMR1状态寄存器 ( TMRx_STS ) .....	336
10.6.4.6 TMR1事件产生寄存器 ( TMRx_EVEG ) .....	337
10.6.4.7 TMR1捕获/比较模式寄存器1 ( TMRx_CCM1 ) .....	338
10.6.4.8 TMR1捕获/比较模式寄存器2 ( TMRx_CCM2 ) .....	340
10.6.4.9 TMR1捕获/比较使能寄存器 ( TMRx_CCE ) .....	342
10.6.4.10 TMR1计数器 ( TMRx_CNT ) .....	344
10.6.4.11 TMR1预分频器 ( TMRx_DIV ) .....	344
10.6.4.12 TMR1自动重装载寄存器 ( TMRx_AR ) .....	345
10.6.4.13 TMR1重复计数寄存器 ( TMRx_RC ) .....	345
10.6.4.14 TMR1捕获/比较寄存器 1 ( TMRx_CC1 ) .....	345
10.6.4.15 TMR1捕获/比较寄存器2 ( TMRx_CC2 ) .....	346
10.6.4.16 TMR1捕获/比较寄存器3 ( TMRx_CC3 ) .....	346
10.6.4.17 TMR1捕获/比较寄存器4 ( TMRx_CC4 ) .....	346
10.6.4.18 TMR1刹车和死区寄存器 ( TMRx_BRKDT ) .....	347
10.6.4.19 TMR1 DMA控制寄存器 ( TMRx_DMAC ) .....	348
10.6.4.20 TMR1连续模式的DMA地址 ( TMRx_DMABA ) .....	349
11 红外线接口 ( IRTMR ) .....	350
12 看门狗 ( WDG ) .....	351
12.1 窗口看门狗 ( WWDG ) .....	351
12.1.1 WWDG简介 .....	351
12.1.2 WWDG主要特性 .....	351

12.1.3 WWDG功能描述 .....	351
12.1.4 如何编写看门狗超时程序 .....	352
12.1.5 调试模式 .....	353
12.1.6 寄存器描述 .....	353
12.1.6.1 控制寄存器 ( WWDG_CTRL ) .....	354
12.1.6.2 配置寄存器 ( WWDG_CFG ) .....	354
12.1.6.3 状态寄存器 ( WWDG_STS ) .....	355
12.2 独立看门狗 ( IWDG ) .....	355
12.2.1 简介 .....	355
12.2.2 IWDG主要性能 .....	355
12.2.3 IWDG功能描述 .....	355
12.2.3.1 硬件看门狗 .....	356
12.2.3.2 寄存器访问保护 .....	356
12.2.3.3 调试模式 .....	356
12.2.4 IWDG寄存器描述 .....	357
12.2.4.1 键寄存器 ( IWDG_KEY ) .....	357
12.2.4.2 预分频寄存器 ( IWDG_PR ) .....	357
12.2.4.3 重装载寄存器 ( IWDG_RLD ) .....	358
12.2.4.4 状态寄存器 ( IWDG_STS ) .....	358
13 实时时钟 ( ERTC ) .....	360
13.1 前言 .....	360
13.2 ERTC的主要特性 .....	360
13.3 ERTC功能说明 .....	361
13.3.1 时钟和预分频器 .....	361
13.3.2 实时时钟和日历 .....	362
13.3.3 可编程闹钟 .....	362
13.3.4 ERTC初始化和配置 .....	362

13.3.5 读取日历 .....	363
13.3.6 复位ERTC.....	364
13.3.7 ERTC同步 .....	364
13.3.8 ERTC参考时钟检测 .....	364
13.3.9 ERTC精密数字校准 .....	365
13.3.10 时间戳功能.....	366
13.3.11 入侵检测 .....	367
13.3.12 校准时钟输出 .....	368
13.3.13 闹钟输出 .....	368
13.4 ERTC和低功耗模式 .....	368
13.5 ERTC中断 .....	368
13.6 ERTC寄存器 .....	369
13.6.1 ERTC时间寄存器(ERTC_TIME) .....	370
13.6.2 ERTC日期寄存器(ERTC_DATE) .....	371
13.6.3 ERTC控制寄存器(ERTC_CTRL) .....	371
13.6.4 ERTC初始化和状态寄存器(ERTC_STS) .....	373
13.6.5 ERTC预分频器寄存器(ERTC_PSC).....	374
13.6.6 ERTC闹钟A寄存器(ERTC_ALA).....	375
13.6.7 ERTC写保护寄存器(ERTC_WPR) .....	376
13.6.8 ERTC亚秒寄存器(ERTC_SBSR) .....	376
13.6.9 ERTC平移控制寄存器(ERTC_SFCTR) .....	376
13.6.10 ERTC时间戳时间寄存器(ERTC_TSTM) .....	377
13.6.11 ERTC时间戳日期寄存器(ERTC_TSDT).....	377
13.6.12 ERTC时间戳亚秒寄存器(ERTC_TSSBS) .....	378
13.6.13 ERTC校准寄存器(ERTC_CCR) .....	378
13.6.14 ERTC入侵和复用功能配置寄存器(ERTC_TPAF).....	379

13.6.15 ERTC闹钟A亚秒寄存器(ERTC_ALASBS).....	380
13.6.16 ERTC备份寄存器(ERTC_BKPxDT).....	381
14 模拟/数字转换 ( ADC ) .....	382
14.1 ADC介绍 .....	382
14.2 ADC主要特征 .....	382
14.3 ADC功能描述 .....	382
14.3.1 ADC开关控制 .....	383
14.3.2 ADC时钟 .....	384
14.3.3 通道选择 .....	384
14.3.4 单次转换模式 .....	384
14.3.5 连续转换模式 .....	384
14.3.6 时序图 .....	385
14.3.7 模拟看门狗 .....	385
14.3.8 扫描模式 .....	386
14.3.9 注入通道管理 .....	386
14.3.10 间断模式 .....	387
14.3.11 校准 .....	388
14.3.12 数据对齐 .....	388
14.3.13 可编程的通道采样时间 .....	389
14.3.14 外部触发转换 .....	389
14.3.15 DMA请求 .....	389
14.3.16 温度传感器 .....	390
14.3.17 ADC中断 .....	391
14.4 ADC寄存器 .....	391
14.4.1 ADC状态寄存器 ( ADC_STS ) .....	392
14.4.2 ADC控制寄存器1 ( ADC_CTRL1 ) .....	393
14.4.3 ADC控制寄存器2 ( ADC_CTRL2 ) .....	395

14.4.4 ADC采样时间寄存器1 ( ADC_SMPT1 ) .....	397
14.4.5 ADC采样时间寄存器2 ( ADC_SMPT2 ) .....	397
14.4.6 ADC注入通道数据偏移寄存器x ( ADC_JOFSx ) ( x=1..4 ) .....	398
14.4.7 ADC看门狗高阈值寄存器 ( ADC_WHTR ) .....	398
14.4.8 ADC看门狗低阈值寄存器 ( ADC_WLTR ) .....	398
14.4.9 ADC规则序列寄存器1 ( ADC_RSQ1 ) .....	399
14.4.10 ADC规则序列寄存器2 ( ADC_RSQ2 ) .....	399
14.4.11 ADC规则序列寄存器3 ( ADC_RSQ3 ) .....	400
14.4.12 ADC注入序列寄存器 ( ADC_JSQ ) .....	400
14.4.13 ADC 注入数据寄存器x ( ADC_JDORx ) ( x= 1..4 ) .....	401
14.4.14 ADC规则数据寄存器 ( ADC_RDOR ) .....	401
15 I <sup>2</sup> C接口 ( I2C ) .....	402
15.1 I <sup>2</sup> C简介 .....	402
15.2 I <sup>2</sup> C主要特点 .....	402
15.3 I <sup>2</sup> C功能描述 .....	403
15.3.1 模式选择 .....	403
15.3.2 I <sup>2</sup> C从模式 .....	404
15.3.3 I <sup>2</sup> C主模式 .....	406
15.3.4 错误条件 .....	411
15.3.5 SDA/SCL线控制 .....	411
15.3.6 SMBus .....	412
15.3.7 DMA请求 .....	413
15.3.8 包错误校验(PEC) .....	414
15.3.9 I <sup>2</sup> C中断请求 .....	415
15.3.10 I <sup>2</sup> C调试模式 .....	416
15.4 I <sup>2</sup> C寄存器描述 .....	416
15.4.1 控制寄存器1(I <sup>2</sup> C_CTRL1) .....	417

15.4.2 控制寄存器2(I <sup>2</sup> C_CTRL2).....	419
15.4.3 自身地址寄存器1(I <sup>2</sup> C_OADDR1).....	420
15.4.4 自身地址寄存器2(I <sup>2</sup> C_OADDR2).....	421
15.4.5 数据寄存器(I <sup>2</sup> C_DT).....	421
15.4.6 状态寄存器1(I <sup>2</sup> C_STS1).....	421
15.4.7 状态寄存器2(I <sup>2</sup> C_STS2).....	424
15.4.8 时钟控制寄存器(I <sup>2</sup> C_CLKCTRL).....	425
15.4.9 TMRISE寄存器(I <sup>2</sup> C_TMRISE).....	426
16 通用同步异步收发器 ( USART ) .....	427
16.1 USART介绍 .....	427
16.2 USART主要特性 .....	427
16.3 USART功能概述 .....	428
16.3.1 USART特性描述.....	429
16.3.2 发送器 .....	430
16.3.2.1 字符发送 .....	430
16.3.2.2 可配置的停止位 .....	430
16.3.2.3 单字节通信 .....	431
16.3.2.4 断开帧 .....	432
16.3.2.5 空闲符号 .....	432
16.3.3 接收器 .....	433
16.3.3.1 起始位侦测 .....	433
16.3.3.2 字符接收 .....	433
16.3.3.3 断开帧 .....	434
16.3.3.4 空闲符号 .....	434
16.3.3.5 溢出错误 .....	434
16.3.3.6 帧错误 .....	435
16.3.3.7 接收期间可配置的停止位 .....	435

16.3.4 分数波特率的产生 .....	436
16.3.4.1 如何从USART_BAUDR寄存器值得到USARTDIV.....	436
16.3.5 USART接收器容忍时钟的变化 .....	438
16.3.6 多处理器通信 .....	438
16.3.6.1 空闲总线检测 ( WUMODE=0 ) .....	438
16.3.6.2 地址标记 ( address mark ) 检测 ( WUMODE=1 ) .....	439
16.3.7 校验控制 .....	439
16.3.8 LIN ( 局域互联网 ) 模式 .....	440
16.3.8.1 LIN发送 .....	440
16.3.8.2 LIN接收 .....	440
16.3.9 USART同步模式 .....	442
16.3.10 单线半双工通信 .....	444
16.3.11 智能卡 .....	445
16.3.12 IrDA SIR ENDEC功能模块 .....	446
16.3.13 利用DMA连续通信 .....	448
16.3.13.1 利用DMA发送 .....	448
16.3.13.2 利用DMA接收 .....	449
16.3.13.3 多缓冲器通信中的错误标志和中断产生 .....	450
16.3.14 硬件流控制 .....	450
16.3.14.1 RTS流控制 .....	450
16.3.14.2 CTS流控制 .....	451
16.4 USART中断请求 .....	451
16.5 USART模式配置 .....	452
16.6 USART寄存器描述 .....	452
16.6.1 USART寄存器地址映象 .....	452
16.6.2 状态寄存器 ( USART_STS ) .....	453

16.6.3 数据寄存器 ( USART_DT ) .....	455
16.6.4 波特比率寄存器 ( USART_BAUDR ) .....	455
16.6.5 控制寄存器1 ( USART_CTRL1 ) .....	456
16.6.6 控制寄存器2 ( USART_CTRL2 ) .....	458
16.6.7 控制寄存器3 ( USART_CTRL3 ) .....	459
16.6.8 保护时间和预分频寄存器 ( GTP ) .....	460
17 串行外设接口 ( SPI ) .....	462
17.1 SPI简介 .....	462
17.2 主要特点 .....	462
17.2.1 SPI特点 .....	462
17.2.2 I <sup>2</sup> S功能 .....	462
17.3 功能描述 .....	463
17.3.1 SPI功能描述 .....	463
17.3.1.1 概述 .....	463
17.3.1.2 配置SPI为从模式 .....	467
17.3.1.3 配置SPI为主模式 .....	468
17.3.1.4 配置SPI为单工通信 .....	469
17.3.1.5 数据发送与接收过程 .....	469
17.3.1.6 CRC计算 .....	474
17.3.1.7 状态标志 .....	475
17.3.1.8 关闭SPI .....	476
17.3.1.9 利用DMA的SPI通信 .....	477
17.3.1.10 错误标志 .....	478
17.3.1.11 SPI中断 .....	479
17.3.2 I <sup>2</sup> S功能描述 .....	479
17.3.2.1 I <sup>2</sup> S功能描述 .....	479
17.3.2.2 支持的音频协议 .....	481

17.3.2.3 时钟发生器 .....	486
17.3.2.4 I <sup>2</sup> S主模式 .....	488
17.3.2.5 I <sup>2</sup> S从模式 .....	490
17.3.2.6 状态标志位 .....	491
17.3.2.7 错误标志位 .....	491
17.3.2.8 I <sup>2</sup> S中断 .....	491
17.3.2.9 DMA功能 .....	492
17.4 SPI寄存器 .....	492
17.4.1 SPI控制寄存器1 ( SPI_CTRL1 ) ( I <sup>2</sup> S模式下不使用 ) .....	494
17.4.2 SPI控制寄存器2 ( SPI_CTRL2 ) .....	495
17.4.3 SPI状态寄存器 ( SPI_STS ) .....	496
17.4.4 SPI数据寄存器 ( SPI_DT ) .....	497
17.4.5 SPICRC多项式寄存器 ( SPI_CPOLY ) ( I <sup>2</sup> S模式下不使用 ) .....	497
17.4.6 SPIRxCRC寄存器 ( SPI_RCRC ) ( I <sup>2</sup> S模式下不使用 ) .....	498
17.4.7 SPITxCRC寄存器 ( SPI_TCRC ) .....	498
17.4.8 SPI_I2S配置寄存器 ( SPI_I2SCTRL ) .....	498
17.4.9 SPI_I2S预分频寄存器 ( SPI_I2SCLKP ) .....	499
18 MCU调试 ( MCUDBG ) .....	501
18.1 简介 .....	501
18.2 功能描述 .....	502
18.2.1 低功耗模式的调试支持 .....	502
18.2.2 支持定时器、看门狗和I <sup>2</sup> C的调试 .....	502
18.2.3 ID代码 .....	502
18.2.4 SW调试端口脚 .....	502
18.3 MCUDBG寄存器 .....	502
18.3.1 MCUDBG设备ID ( MCUDBG_IDCODE ) .....	503
18.3.2 MCUDBG控制寄存器 ( MCUDBG_CTRL ) .....	504

19 比较器(COMP) .....	507
19.1 COMP简介 .....	507
19.2 COMP的主要特性 .....	507
19.3 比较器功能描述 .....	507
19.3.1 比较器框图 .....	507
19.3.2 COMP引脚和内部信号 .....	508
19.3.3 比较器复位和时钟 .....	508
19.3.4 比较器锁定机制 .....	508
19.3.5 迟滞 .....	508
19.3.6 比较器输出消隐功能 .....	509
19.3.7 功率模式 .....	509
19.3.8 干扰滤波器 .....	510
19.4 COMP中断 .....	510
19.5 COMP寄存器 .....	510
19.5.1 比较器控制和状态寄存器1(COMP_CTRLSTS) .....	511
19.5.2 干扰滤波器使能寄存器(G_FILTER_EN) .....	513
19.5.3 干扰滤波器高脉冲数(HIGH_PULSE) .....	513
19.5.4 干扰滤波器低脉冲数(LOW_PULSE) .....	514
20 版本历史 .....	515

## 表格目录

表格 1-1	寄存器组起始地址 .....	42
表格 1-2	64KB 闪存模块的组织 .....	44
表格 1-3	32KB 闪存模块的组织 .....	45
表格 1-4	16KB 闪存模块的组织 .....	46
表格 1-5	启动模式 .....	47
表格 1-6	寄存器缩写表 .....	47
表格 2-1	SLP-NOW 模式 .....	54
表格 2-2	SLP-ON-EXIT 模式 .....	54
表格 2-3	停机模式 .....	55
表格 2-4	待机模式 .....	56
表格 2-5	PWR 寄存器的映像和复位值 .....	56
表格 3-1	RCC 寄存器的映像和复位值 .....	66
表格 4-1	64KB 闪存模块的组织 .....	86
表格 4-2	32KB 闪存模块的组织 .....	87
表格 4-3	16KB 闪存模块的组织 .....	88
表格 4-4	闪存存储器保护状态 .....	94
表格 4-5	读保护等级切换说明 .....	95
表格 4-6	选择字节格式 .....	96
表格 4-7	信息块的组织结构 .....	96
表格 4-8	用户选择字节说明 .....	96
表格 4-9	闪存接口—寄存器映像和复位值 .....	99
表格 5-1	CRC 计算单元寄存器映像 .....	113
表格 6-1	端口位配置表 .....	116
表格 6-2	通过 GPIOA_AFR 寄存器配置端口 A 的复用功能 .....	119
表格 6-3	通过 GPIOB_AFR 寄存器配置端口 B 的复用功能 .....	119
表格 6-4	通过 GPIOF_AFR 寄存器配置端口 F 的复用功能 .....	120
表格 6-5	GPIO 寄存器地址映射和复位值 .....	124
表格 7-1	SYSCFG 寄存器地址映射和复位值 .....	132
表格 8-1	AT32F421 产品的向量表 .....	136
表格 8-2	外部中断/事件控制器寄存器映像和复位值 .....	140
表格 9-1	可编程的数据传输宽度和大小端操作 (当 PINC = MINC = 1 ) .....	148

表格 9-2	DMA 中断请求 .....	149
表格 9-3	各个通道的 DMA 请求一览 .....	150
表格 9-4	DMA 寄存器映像和复位值 .....	151
表格 10-1	TMR6 寄存器图和复位值 .....	162
表格 10-2	计数方向与编码器信号的关系 .....	187
表格 10-3	TMRx – 寄存器图和复位值 .....	196
表格 10-4	TMRx 内部触发连接 <sup>(1)</sup> .....	201
表格 10-5	标准 OCx 通道的输出控制位 .....	209
表格 10-6	TMRx – 寄存器图和复位值 .....	223
表格 10-7	标准 OCx 通道的输出控制位 .....	229
表格 10-8	TMR15 寄存器图和复位值 .....	252
表格 10-9	TMRx 内部触发连接 .....	256
表格 10-10	带刹车功能的互补输出通道 OCx 和 OCxN 的控制位 .....	262
表格 10-11	TMR16/17 寄存器图和复位值 .....	283
表格 10-12	带刹车功能的互补输出通道 OCx 和 OCxN 的控制位 .....	291
表格 10-13	计数方向与编码器信号的关系 .....	323
表格 10-14	TMR1 寄存器图和复位值 .....	329
表格 10-15	TMRx 内部触发连接 .....	335
表格 10-16	带刹车功能的互补输出通道 OCx 和 OCxN 的控制位 .....	343
表格 12-1	WWDG 寄存器映像和复位值 .....	353
表格 12-2	看门狗超时时间 ( 40kHz 的输入时钟 ( LSI ) ) <sup>(1)</sup> .....	356
表格 12-3	IWDG 寄存器映像和复位值 .....	357
表格 13-1	低功耗模式对 ERTC 的作用 .....	368
表格 13-2	中断控制位 .....	369
表格 13-3	ERTC 寄存器映像和复位值 .....	369
表格 14-1	ADC 引脚 .....	383
表格 14-2	模拟看门狗通道选择 .....	385
表格 14-3	ADC1 用于规则通道的外部触发 .....	389
表格 14-4	ADC1 用于注入通道的外部触发 .....	389
表格 14-5	ADC 中断 .....	391
表格 14-6	ADC 寄存器映像和复位值 .....	391
表格 15-1	SMBus 与 I <sup>2</sup> C 的比较 .....	412

表格 15-2	I <sup>2</sup> C 中断请求表 .....	415
表格 15-3	I <sup>2</sup> C 寄存器地址映像和复位值 .....	416
表格 16-1	检测噪声的数据采样 .....	435
表格 16-2	设置波特率时的误差计算 .....	436
表格 16-3	当 DIV_Decimal=0 时，USART 接收器的容忍度 .....	438
表格 16-4	当 DIV_Decimal≠0 时，USART 接收器的容忍度 .....	438
表格 16-5	帧格式 .....	439
表格 16-6	USART 中断请求 .....	451
表格 16-7	USART 模式设置 (1) .....	452
表格 16-8	USART 寄存器地址映像和复位值 .....	452
表格 17-1	SPI 中断请求 .....	479
表格 17-2	使用系统时钟得到精确的音频频率 .....	487
表格 17-3	I <sup>2</sup> S 中断请求 .....	492
表格 17-4	SPI 寄存器列表及其复位值 .....	492
表格 18-1	MCUDBG 寄存器地址映像和复位值 .....	502
表格 19-1	COMP 寄存器图和复位值 .....	510

### 图表目录

图表 1-1	AT32F421 系列微控制器系统架构 .....	38
图表 1-2	Cortex <sup>TM</sup> -M4 内部框图 .....	39
图表 1-3	AT32F421 地址配置 .....	41
图表 2-1	各电源域框图 .....	50
图表 2-2	上电复位和掉电复位的波形图 .....	51
图表 2-3	PVD 的阈值与输出 .....	52
图表 3-1	复位电路 .....	60
图表 3-2	时钟树 .....	62
图表 3-3	HSE/LSE 时钟源 .....	63
图表 4-1	编程过程 .....	90
图表 4-2	闪存页擦除过程 .....	92
图表 4-3	闪存整片擦除过程 .....	92
图表 4-4	读保护等级切换状态图 .....	94
图表 5-1	CRC 单元框 .....	112
图表 6-1	I/O 端口位的基本结构 .....	115

图表 6-2	5 伏兼容 I/O 端口位的基本结构 .....	116
图表 6-3	输入浮空/上拉/下拉配置 .....	120
图表 6-4	高阻抗的模拟输入配置 .....	122
图表 6-5	输出配置 .....	123
图表 6-6	复用功能配置 .....	123
图表 8-1	外部中断/事件控制器框图 .....	138
图表 8-2	外部中断通用 I/O 映像 .....	140
图表 9-1	DMA 框图 .....	146
图表 9-2	DMA 请求映像 .....	149
图表 10-1	基本定时器框图 .....	157
图表 10-2	预分频系数从 1 变到 2 的计数器时序图 .....	158
图表 10-3	预分频系数从 1 变到 4 的计数器时序图 .....	158
图表 10-4	计数器时序图 , 内部时钟分频系数为 1 .....	159
图表 10-5	计数器时序图 , 内部时钟分频系数为 2 .....	160
图表 10-6	计数器时序图 , 内部时钟分频系数为 4 .....	160
图表 10-7	计数器时序图 , 内部时钟分频系数为 N .....	160
图表 10-8	计数器时序图 , 当 ARPEN=0 时的更新事件 ( TMRx_AR 没有预装载 ) .....	161
图表 10-9	计数器时序图 , 当 ARPEN=1 时的更新事件 ( 预装载 TMRx_AR ) .....	161
图表 10-10	普通模式时序图 , 内部时钟分频系数为 1 .....	162
图表 10-11	通用定时器框图 .....	167
图表 10-12	当预分频器的参数从 1 变到 2 时 , 计数器的时序图 .....	168
图表 10-13	当预分频器的参数从 1 变到 4 时 , 计数器的时序图 .....	168
图表 10-14	计数器时序图 , 内部时钟分频因子为 1 .....	169
图表 10-15	计数器时序图 , 内部时钟分频因子为 2 .....	170
图表 10-16	计数器时序图 , 内部时钟分频因子为 4 .....	170
图表 10-17	计数器时序图 , 内部时钟分频因子为 N .....	170
图表 10-18	计数器时序图 , 当 ARPEN=0 时的更新事件 ( 没有预装入了 TMRx_AR ) .....	171
图表 10-19	计数器时序图 , 当 ARPEN=1 时的更新事件 ( 预装入了 TMRx_AR ) .....	171
图表 10-20	计数器时序图 , 内部时钟分频因子为 1 .....	172
图表 10-21	计数器时序图 , 内部时钟分频因子为 2 .....	172
图表 10-22	计数器时序图 , 内部时钟分频因子为 4 .....	172
图表 10-23	计数器时序图 , 内部时钟分频因子为 N .....	173

图表 10-24 计数器时序图 , 当 ARPEN=0 时的更新事件.....	173
图表 10-25 计数器时序图 , 内部时钟分频因子为 1 , TMRx_AR=0x6.....	174
图表 10-26 计数器时序图 , 内部时钟分频因子为 2 .....	174
图表 10-27 计数器时序图 , 内部时钟分频因子为 4 , TMRx_AR=0x36.....	175
图表 10-28 计数器时序图 , 内部时钟分频因子为 N .....	175
图表 10-29 计数器时序图 , ARPEN=1 时的更新事件 ( 计数器下溢 ) .....	175
图表 10-30 计数器时序图 , ARPEN=1 时的更新事件 ( 计数器溢出 ) .....	176
图表 10-31 一般模式下的控制电路 , 内部时钟分频因子为 1.....	177
图表 10-32 TI2 外部时钟连接例子 .....	177
图表 10-33 外部时钟模式 1 下的控制电路 .....	177
图表 10-34 外部触发输入框图 .....	178
图表 10-35 外部时钟模式 2 下的控制电路 .....	178
图表 10-36 捕获/比较通道 ( 如 : 通道 1 输入部分 ) .....	179
图表 10-37 捕获/比较通道 1 的主电路 .....	179
图表 10-38 捕获/比较通道的输出部分 ( 通道 1 ) .....	180
图表 10-39 PWM 输入模式时序.....	181
图表 10-40 输出比较模式 , 翻转 OC1 .....	183
图表 10-41 边沿对齐的 PWM 波形 ( AR=8 ) .....	184
图表 10-42 中央对齐的 PWM 波形 ( AP=8 ) .....	184
图表 10-43 单脉冲模式的例子 .....	185
图表 10-44 清除 TMRx 的 OCxREF .....	187
图表 10-45 编码器模式下的计数器操作实例 .....	188
图表 10-46 IC1FP1 反相的编码器接口模式实例 .....	188
图表 10-47 复位模式下的控制电路 .....	189
图表 10-48 门控模式下的控制电路 .....	190
图表 10-49 触发器模式下的控制电路 .....	190
图表 10-50 外部时钟模式 2 + 触发模式下的控制电路 .....	191
图表 10-51 主/从定时器的例子 .....	191
图表 10-52 定时器 1 的 OC1REF 控制定时器 2 .....	192
图表 10-53 通过使能定时器 1 可以控制定时器 2 .....	193
图表 10-54 使用定时器 1 的更新触发定时器 2 .....	194
图表 10-55 利用定时器 1 的使能触发定时器 2 .....	195

图表 10-56 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2.....	196
图表 10-57 通用定时器 TMR14 框图.....	215
图表 10-58 当预分频器的参数从 1 变到 2 时，计数器的时序图.....	215
图表 10-59 当预分频器的参数从 1 变到 4 时，计数器的时序图.....	216
图表 10-60 计数器时序图，内部时钟分频因子为 1 .....	217
图表 10-61 计数器时序图，内部时钟分频因子为 2 .....	217
图表 10-62 计数器时序图，内部时钟分频因子为 4 .....	217
图表 10-63 计数器时序图，内部时钟分频因子为 N .....	218
图表 10-64 计数器时序图，当 ARPEN=0 时的更新事件 ( TMRx_AR 没有预装入 ) .....	218
图表 10-65 计数器时序图，当 ARPEN=1 时的更新事件 ( 预装入了 TMRx_AR ) .....	218
图表 10-66 一般模式下的控制电路，内部时钟分频因子为 1.....	219
图表 10-67 捕获/比较通道 ( 如：通道 1 输入部分 ) .....	219
图表 10-68 捕获/比较通道 1 的主电路 .....	220
图表 10-69 捕获/比较通道的输出部分 ( 通道 1 ) .....	220
图表 10-70 输出比较模式，翻转 OC1 .....	222
图表 10-71 边沿对齐的 PWM 波形 ( AR=8 ) .....	223
图表 10-72 通用控制定时器 15 框图.....	233
图表 10-73 当预分频器的参数从 1 变到 2 时，计数器的时序图.....	234
图表 10-74 当预分频器的参数从 1 变到 4 时，计数器的时序图.....	234
图表 10-75 计数器时序图，内部时钟分频因子为 1 .....	235
图表 10-76 计数器时序图，内部时钟分频因子为 2 .....	235
图表 10-77 计数器时序图，内部时钟分频因子为 4 .....	236
图表 10-78 计数器时序图，内部时钟分频因子为 N .....	236
图表 10-79 计数器时序图，当 ARPEN=0 时的更新事件 ( TMRx_AR 没有预装入 ) .....	236
图表 10-80 计数器时序图，当 ARPEN=1 时的更新事件 ( 预装入了 TMRx_AR ) .....	237
图表 10-81 不同模式下更新速率的例子，及 TMRx_RC 的寄存器设置.....	237
图表 10-82 一般模式下的控制电路，内部时钟分频因子为 1.....	238
图表 10-83 TI2 外部时钟连接例子 .....	239
图表 10-84 外部时钟模式 1 下的控制电路 .....	239
图表 10-85 捕获/比较通道 ( 如：通道 1 输入部分 ) .....	240
图表 10-86 捕获/比较通道 1 的主电路 .....	240
图表 10-87 捕获/比较通道的输出部分 ( 通道 1 ) .....	241

图表 10-88 捕获/比较通道的输出部分 ( 通道 2 ) .....	241
图表 10-89 PWM 输入模式时序.....	243
图表 10-90 输出比较模式 , 翻转 OC1 .....	244
图表 10-91 边沿对齐的 PWM 波形 ( AR=8 ) .....	245
图表 10-92 带死区插入的互补输出.....	245
图表 10-93 死区波形延迟大于负脉冲 .....	246
图表 10-94 死区波形延迟大于正脉冲 .....	246
图表 10-95 响应刹车的输出.....	247
图表 10-96 单脉冲模式的例子 .....	248
图表 10-97 复位模式下的控制电路.....	250
图表 10-98 门控模式下的控制电路 .....	250
图表 10-99 触发器模式下的控制电路 .....	251
图表 10-100 通用控制定时器 16/17 框图.....	269
图表 10-101 当预分频器的参数从 1 变到 2 时 , 计数器的时序图 .....	270
图表 10-102 当预分频器的参数从 1 变到 4 时 , 计数器的时序图 .....	270
图表 10-103 计数器时序图 , 内部时钟分频因子为 1 .....	271
图表 10-104 计数器时序图 , 内部时钟分频因子为 2 .....	271
图表 10-105 计数器时序图 , 内部时钟分频因子为 4 .....	271
图表 10-106 计数器时序图 , 内部时钟分频因子为 N .....	272
图表 10-107 计数器时序图 , 当 ARPEN=0 时的更新事件 ( TMRx_AR 没有预装入 ) .....	272
图表 10-108 计数器时序图 , 当 ARPEN=1 时的更新事件 ( 预装入了 TMRx_AR ) .....	272
图表 10-109 不同模式下更新速率的例子 , 及 TMRx_RC 的寄存器设置.....	273
图表 10-110 一般模式下的控制电路 , 内部时钟分频因子为 1.....	274
图表 10-111 捕获/比较通道 ( 如 : 通道 1 输入部分 ) .....	275
图表 10-112 捕获/比较通道 1 的主电路 .....	275
图表 10-113 捕获/比较通道的输出部分 ( 通道 1 ) .....	275
图表 10-114 输出比较模式 , 翻转 OC1 .....	277
图表 10-115 边沿对齐的 PWM 波形 ( AR=8 ) .....	278
图表 10-116 带死区插入的互补输出 .....	279
图表 10-117 死区波形延迟大于负脉冲 .....	280
图表 10-118 死区波形延迟大于正脉冲 .....	280
图表 10-119 响应刹车的输出 .....	281

图表 10-120 高级控制定时器框图.....	297
图表 10-121 当预分频器的参数从 1 变到 2 时，计数器的时序图.....	298
图表 10-122 当预分频器的参数从 1 变到 4 时，计数器的时序图.....	298
图表 10-123 计数器时序图，内部时钟分频因子为 1 .....	299
图表 10-124 计数器时序图，内部时钟分频因子为 2 .....	299
图表 10-125 计数器时序图，内部时钟分频因子为 4 .....	300
图表 10-126 计数器时序图，内部时钟分频因子为 N .....	300
图表 10-127 计数器时序图，当 ARPEN=0 时的更新事件 ( TMRx_AR 没有预装入 ) .....	300
图表 10-128 计数器时序图，当 ARPEN=1 时的更新事件 ( 预装入了 TMRx_AR ) .....	301
图表 10-129 计数器时序图，内部时钟分频因子为 1 .....	302
图表 10-130 计数器时序图，内部时钟分频因子为 2 .....	302
图表 10-131 计数器时序图，内部时钟分频因子为 4 .....	302
图表 10-132 计数器时序图，内部时钟分频因子为 N .....	303
图表 10-133 计数器时序图，当没有使用自动装载时的更新事件 .....	303
图表 10-134 计数器时序图，内部时钟分频因子为 1 , TMRx_AR=0x6.....	304
图表 10-135 计数器时序图，内部时钟分频因子为 2 .....	304
图表 10-136 计数器时序图，内部时钟分频因子为 4 , TMRx_AR=0x36.....	305
图表 10-137 计数器时序图，内部时钟分频因子为 N .....	305
图表 10-138 计数器时序图，ARPEN=1 时的更新事件 ( 计数器下溢 ) .....	306
图表 10-139 计数器时序图，ARPEN=1 时的更新事件 ( 计数器溢出 ) .....	306
图表 10-140 不同模式下更新速率的例子，及 TMRx_RC 的寄存器设置.....	307
图表 10-141 一般模式下的控制电路，内部时钟分频因子为 1.....	308
图表 10-142 TI2 外部时钟连接例子 .....	308
图表 10-143 外部时钟模式 1 下的控制电路 .....	308
图表 10-144 外部触发输入框图 .....	309
图表 10-145 外部时钟模式 2 下的控制电路 .....	309
图表 10-146 捕获/比较通道 ( 如 : 通道 1 输入部分 ) .....	310
图表 10-147 捕获/比较通道 1 的主电路 .....	310
图表 10-148 捕获/比较通道的输出部分 ( 通道 1 至 3 ) .....	311
图表 10-149 捕获/比较通道的输出部分 ( 通道 4 ) .....	311
图表 10-150 PWM 输入模式时序.....	313
图表 10-151 输出比较模式，翻转 OC1 .....	314

图表 10-152 边沿对齐的 PWM 波形 ( AR=8 ) .....	315
图表 10-153 中央对齐的 PWM 波形 ( AR=8 ) .....	315
图表 10-154 带死区插入的互补输出 .....	317
图表 10-155 死区波形延迟大于负脉冲 .....	317
图表 10-156 死区波形延迟大于正脉冲 .....	317
图表 10-157 响应刹车的输出 .....	319
图表 10-158 清除 TMRx 的 OCxREF .....	320
图表 10-159 产生六步 PWM , 使用 HALL 的例子 ( OSIMR=1 ) .....	320
图表 10-160 单脉冲模式的例子 .....	321
图表 10-161 编码器模式下的计数器操作实例 .....	323
图表 10-162 IC1FP1 反相的编码器接口模式实例 .....	324
图表 10-163 霍尔传感器接口的实例 .....	326
图表 10-164 复位模式下的控制电路 .....	327
图表 10-165 门控模式下的控制电路 .....	327
图表 10-166 触发器模式下的控制电路 .....	328
图表 10-167 外部时钟模式 2 + 触发模式下的控制电路 .....	329
图表 11-1 红外线内部连接示意图 .....	350
图表 12-1 看门狗框图 .....	351
图表 12-2 窗口看门狗时序图 .....	352
图表 12-3 独立看门狗框图 .....	356
图表 13-1 ERTC 框图 .....	360
图表 14-1 单个 ADC 框图 .....	382
图表 14-2 时序图 .....	385
图表 14-3 模拟看门狗警戒区 .....	385
图表 14-4 注入转换延时 .....	386
图表 14-5 校准时序图 .....	388
图表 14-6 数据右对齐 .....	388
图表 14-7 数据左对齐 .....	388
图表 14-8 温度传感器和 V <sub>REFINT</sub> 通道框图 .....	390
图表 15-1 I <sup>2</sup> C 总线协议 .....	403
图表 15-2 I <sup>2</sup> C 的功能框图 .....	403
图表 15-3 从发送器的传送序列图 .....	405

图表 15-4 从接收器的传送序列图 .....	405
图表 15-5 主发送器传送序列图 .....	407
图表 15-6 主接收器传送序列图 .....	408
图表 15-7 N>2 时主收流程图 .....	409
图表 15-8 N=2 时主收流程图 .....	410
图表 15-9 N=1 时主收流程图 .....	410
图表 15-10 I <sup>2</sup> C 中断映射图 .....	416
图表 16-1 USART 框图 .....	428
图表 16-2 字长设置 .....	429
图表 16-3 配置停止位 .....	431
图表 16-4 发送时 TRAC/TDE 的变化情况 .....	432
图表 16-5 起始位侦测 .....	433
图表 16-6 检测噪声的数据采样 .....	434
图表 16-7 利用空闲总线检测的静默模式 .....	439
图表 16-8 利用地址标记检测的静默模式 .....	439
图表 16-9 LIN 模式下的断开检测 ( 11 位断开长度-设置了 LBDLEN 位 ) .....	441
图表 16-10 LIN 模式下的断开检测与帧错误的检测 .....	442
图表 16-11 USART 同步传输的例子 .....	443
图表 16-12 USART 数据时钟时序示例 ( LEN=0 ) .....	443
图表 16-13 USART 数据时钟时序示例 ( LEN=1 ) .....	444
图表 16-14 RX 数据采样/保持时间 .....	444
图表 16-15 ISO7816-3 异步协议 .....	445
图表 16-16 使用 1.5 停止位检测奇偶检验错 .....	446
图表 16-17 IrDA SIR ENDEC 框图 .....	448
图表 16-18 IrDA 数据调制 ( 3/16 ) -普通模式 .....	448
图表 16-19 利用 DMA 发送 .....	449
图表 16-20 利用 DMA 接收 .....	450
图表 16-21 两个 USART 间的硬件流控制 .....	450
图表 16-22 RTS 流控制 .....	450
图表 16-23 CTS 流控制 .....	451
图表 16-24 USART 中断映像图 .....	452
图表 17-1 SPI 框图 .....	464

图表 17-2 单主和单从应用 .....	465
图表 17-3 硬件/软件的从选择管理 .....	466
图表 17-4 数据时钟时序图 .....	467
图表 17-5 主模式、全双工模式下 ( BDMODE=0 并且 RONLY=0 ) 连续输出时 , TE/RNE/BSY 的变化示意图 .....	471
图表 17-6 从模式、全双工模式下 ( BDMODE=0 并且 RONLY=0 ) 连续传输时 , TE/RNE/BSY 的变化示意图 .....	472
图表 17-7 主设备只发送模式 ( BDMODE=0 并且 RONLY=0 ) 下连续传输时 , TE/BSY 变化示意图 .	473
图表 17-8 从设备只发送模式 ( BDMODE=0 并且 RONLY=0 ) 下连续传输时 , TE/BSY 变化示意图 .	473
图表 17-9 只接收模式 ( BDMODE=0 并且 RONLY=1 ) 下连续传输时 , RNE 变化示意图 .....	474
图表 17-10 非连续传输发送 ( BDMODE=0 并且 RONLY=0 ) 时 , TE/BSY 变化示意图 .....	474
图表 17-11 使用 DMA 发送 .....	478
图表 17-12 使用 DMA 接收 .....	478
图表 17-13 I <sup>2</sup> S 框图 .....	479
图表 17-14 I <sup>2</sup> S 飞利浦协议波形 ( 16/32 位全精度 , CPOL=0 ) .....	481
图表 17-15 I <sup>2</sup> S 飞利浦协议标准波形 ( 24 位帧 , CPOL=0 ) .....	481
图表 17-16 发送 0x8EAA33 .....	482
图表 17-17 接收 0x8EAA33 .....	482
图表 17-18 I <sup>2</sup> S 飞利浦协议标准波形 ( 16 位扩展至 32 位包帧 , CPOL=0 ) .....	482
图表 17-19 示例 .....	483
图表 17-20 MSB 对齐 16 位或 32 位全精度 , CPOL=0 .....	483
图表 17-21 MSB 对齐 24 位数据 , CPOL=0 .....	483
图表 17-22 MSB 对齐 16 位数据扩展到 32 位包帧 , CPOL=0 .....	483
图表 17-23 LSB 对齐 16 位或 32 位全精度 , CPOL=0 .....	484
图表 17-24 LSB 对齐 24 位数据 , CPOL=0 .....	484
图表 17-25 要求发送 0x3478AE 的操作 .....	484
图表 17-26 要求接收 0x3478AE 的操作 .....	485
图表 17-27 LSB 对齐 16 位数据扩展到 32 位包帧 , CPOL=0 .....	485
图表 17-28 示例 .....	485
图表 17-29 PCM 标准波形 ( 16 位 ) .....	486
图表 17-30 PCM 标准波形 ( 16 位扩展到 32 位包帧 ) .....	486
图表 17-31 音频采样频率定义 .....	486

图表 17-32 I <sup>2</sup> S 时钟发生器结构 .....	487
图表 18-1 AT32F421 级别和 Cortex <sup>TM</sup> -M4 级别的调试框图 .....	501
图表 19-1 比较器的框图 .....	507
图表 19-2 比较器迟滞 .....	508
图表 19-3 比较器输出消隐 .....	509
图表 19-4 H_PULSE_CNT=1 和 L_PULSE_CNT =0 时干扰滤波器时序图 .....	510
图表 19-5 H_PULSE_CNT=2 和 L_PULSE_CNT = 1 时干扰滤波器时序图 .....	510

# 1 系统架构 (SYSTEM)

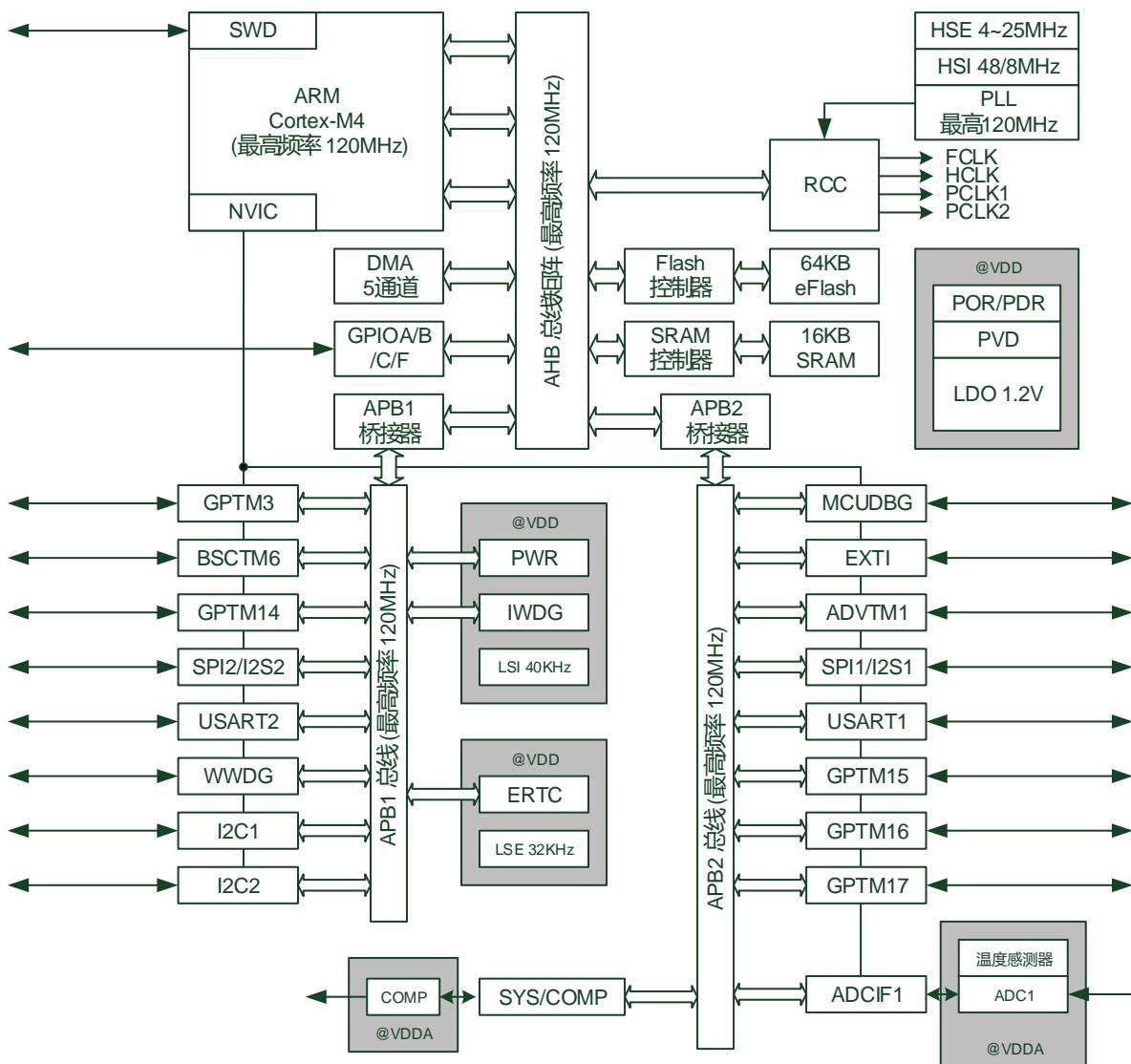
AT32F421 系列微控制器包括 ARM<sup>®</sup> Cortex<sup>TM</sup>-M4 处理器内核、总线架构、外设以及存储器构成。Cortex<sup>TM</sup>-M4 处理器是一种新时代的内核，拥有许多先进功能。对比于 Cortex<sup>TM</sup>-M3，Cortex<sup>TM</sup>-M4 处理器支持增强的高效 DSP 指令集，包含扩展的单周期 16/32 位乘法累加器（MAC）、双 16 位 MAC 指令、优化的 8/16 位 SIMD 运算及饱和运算指令。当设计中使用带 DSP 功能的 Cortex<sup>TM</sup>-M4 时就能格外节能，比软件解决方案更快，使 Cortex<sup>TM</sup>-M4 适用于那些要求微控制器提供高效能与低功耗的产品市场。

## 1.1 系统概述

AT32F421 主系统基于 AHB 总线矩阵整合而成。AHB 总线矩阵是基于 AMBA3.0 AHB-LITE 的多层次 AHB，可使系统中多个主设备与从设备之间建立并行的访问路径，确保总线带宽的有效利用。AHB 总线矩阵包含四个主设备：Cortex<sup>TM</sup>-M4 内核的 ICode 总线、DCode 总线、系统总线，以及 DMA 控制器。总线中包含六个从设备：闪存控制器（EFC）、片上 SRAM、GPIO 控制器、DMA 控制器以及两个 APB 总线网桥。系统总线主要用于加载/储存数据以及调试访问系统存储区。系统存储区可划分为片上 SRAM 区、外部存储区以及外设映像区。

系统 AHB 总线与所有的 AHB 外设相连，此外还包含两条 AHB-APB 总线桥接器，这样可以在系统 AHB 总线以及两个 APB 总线之间实现完全同步连接。两条 APB 总线则与所有的 APB 外设相连。APB1 总线的最高速度限制为 120MHz，APB2 总线的最高速度限制为 120MHz，以上所述设备通过多层次 AHB 总线架构相互连接，如图 1-1 所示：

图表 1-1 AT32F421系列微控制器系统架构



## 1.1.1 总线架构

### ICode 总线

- 该总线将 Cortex™-M4 内核的指令总线与闪存指令接口相连接。指令预取在此总线上完成。

### DCode 总线

- 该总线将 Cortex™-M4 内核的 DCode 总线与闪存存储器的数据接口相连接（常量加载和调试访问）。

### 系统总线

- 此总线连接 Cortex™-M4 内核的系统总线（外设总线）到总线矩阵，总线矩阵协调着内核和 DMA 间的访问。

### DMA 总线

- 此总线将 DMA 的 AHB 主控接口与总线矩阵相联，总线矩阵协调着 CPU 的 DCode 和 DMA 到 SRAM、闪存和外设的访问。

### 总线矩阵

- 总线矩阵协调内核系统总线和 DMA 主控总线之间的访问仲裁，仲裁利用轮换算法。AHB 外设通过总线矩阵与系统总线相连，允许 DMA 访问。

### AHB/APB 桥接器 (APB)

两个 AHB/APB 桥在 AHB 和 2 个 APB 总线间提供同步连接。APB1 操作速度限于 120MHz，APB2 操作速度限于 120MHz。有关连接到每个桥的不同外设的地址映像请参考 [1.2 地址映射章节](#)。在每一次复位以后，所有除 SRAM 和 EFC 以外的外设都被关闭，在使用一个外设之前，必须设置寄存器 RCC\_AHBEN 来打开该外设的时钟。

**注意：**当对 APB 寄存器进行 8 位或者 16 位访问时，该访问会被自动转换成 32 位的访问，桥接器会自动将 8 位或者 32 位的数据扩展以配合 32 位的向量。

## 1.1.2 ARM Cortex™-M4 处理器

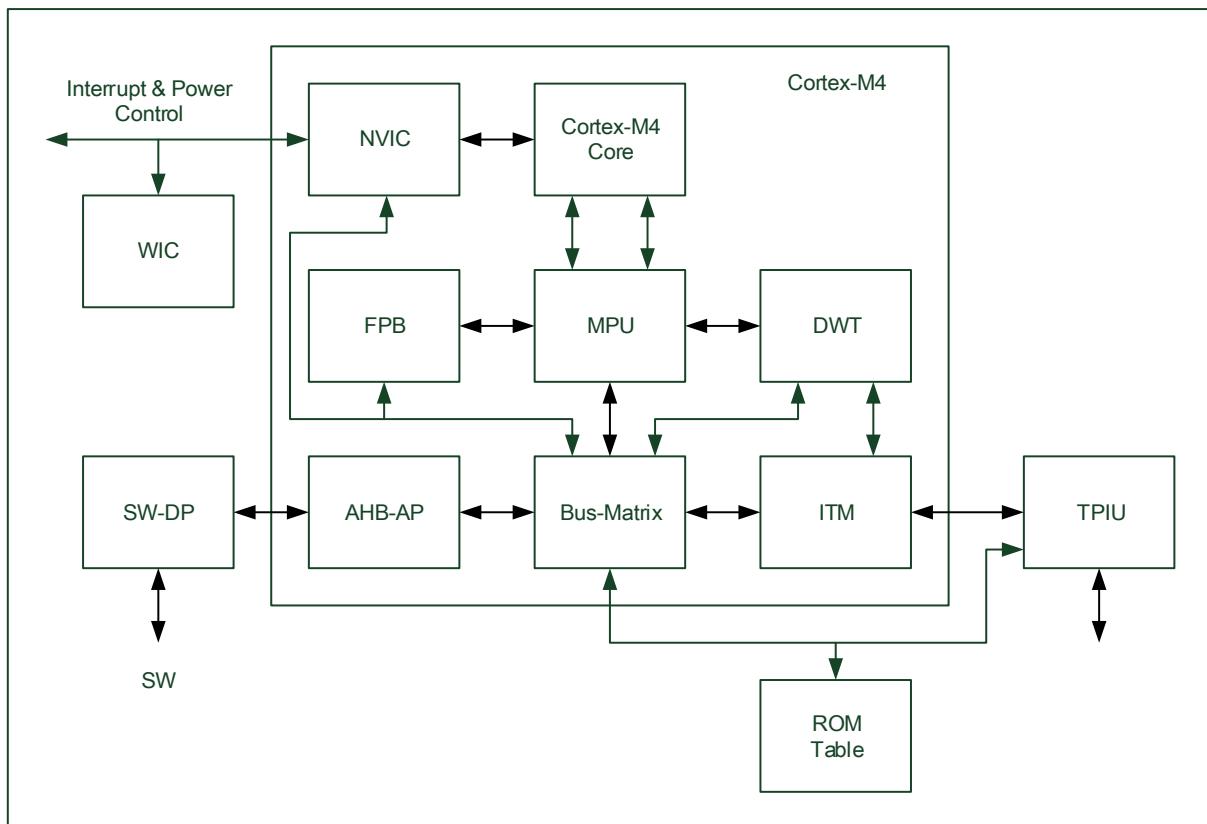
Cortex™-M4 处理器是一款低功耗处理器，具有低门数，低中断延迟和低成本调试。支持包括 DSP 指令集与浮点运算功能，特别适合用于深度嵌入式应用程序需要快速中断响应功能。Cortex™-M4 处理器是基于 ARMv7-M 架构，既支持 Thumb 指令集也支持 DSP 指令集。Cortex™-M4 处理器并提供以下系统级外设：

内部总线矩阵，用于实现 ICode 总线、DCode 总线、系统总线、专用外设总线 (PPB) 以及调试专用总线 (AHB-AP) 的互联：

- 嵌套式向量型中断控制器 (Nested Vectored Interrupt Controller, 简写为 NVIC)
- 闪存地址重载及断点单元 (Flash Patch and Breakpoint, 简写为 FPB)
- 内存保护单元 (Memory Protection Unit, 简写为 MPU)
- 数据观测点及跟踪单元 (Data Watchpoint and Trace, 简写为 DWT)
- 跟踪仪器宏单元 (Instrument Trace Macrocell, 简写为 ITM)
- 串行线调试接口 (Serial Wire Debug Port, 简写为 SW-DP)
- 跟踪端口接口单元 (Trace Port Interface Unit, 简写为 TPIU)

下图为 Cortex™-M4 处理器的内部框图，请参阅《ARM® Cortex-M4 技术参考手册》了解关于 Cortex™-M4 更详尽信息。

图表 1-2 Cortex™-M4 内部框图

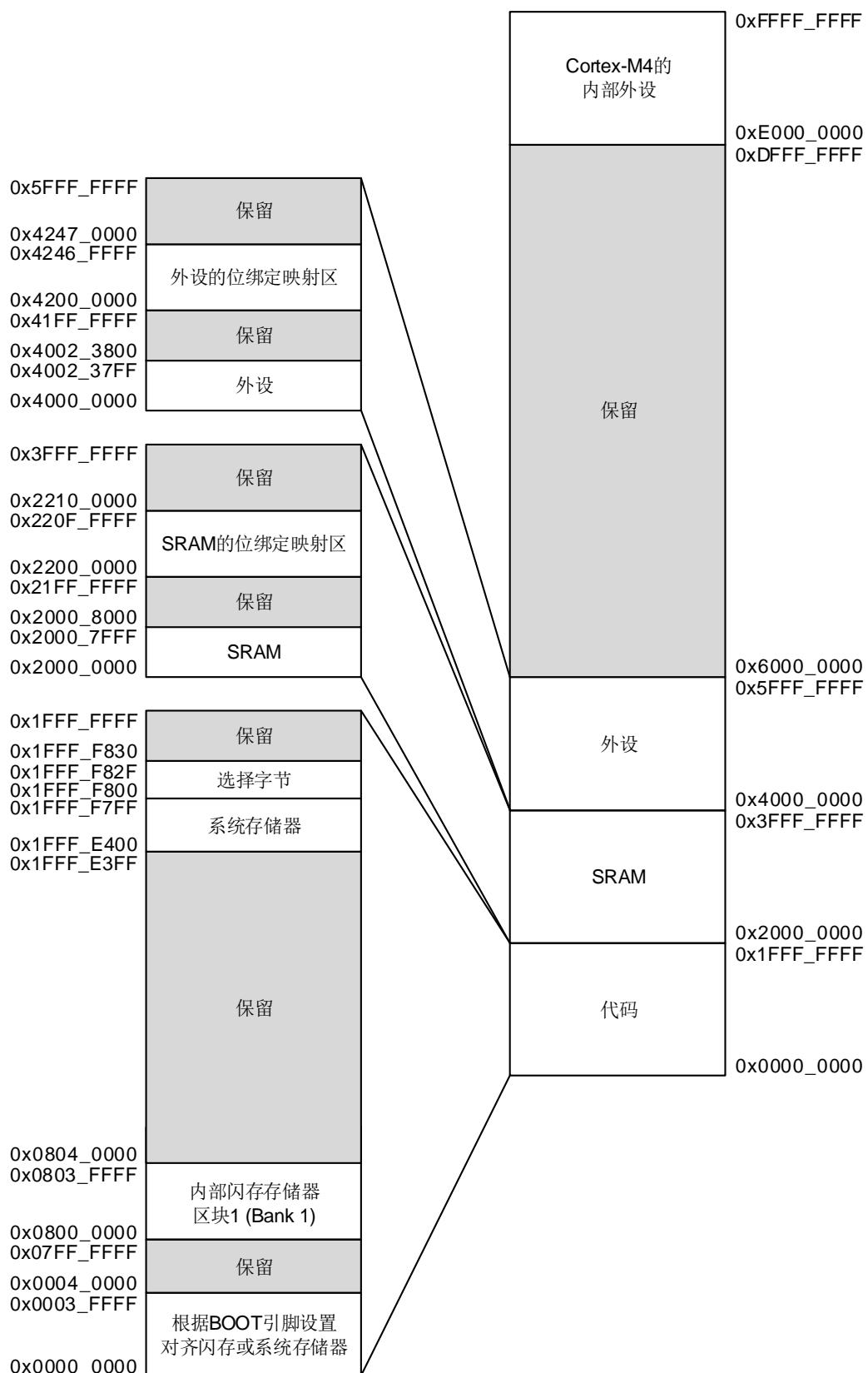


## 1.2 地址映射

程序存储器、数据存储器、寄存器和输入输出端口被组织在同一个 4GB 的线性地址空间内。数据字节以小端格式存放在存储器中。一个字里的最低地址字节被认为是该字的最低有效字节，而最高地址字节是最有效字节。

外设寄存器的映像请参考相关章节。可访问的存储器空间被分成 8 个主要块，每个块为 512MB。其他所有没有分配给片上存储器和外设的存储器空间都是保留的地址空间，请参考相应器件的数据手册中的存储器映像图。

图表 1-3 AT32F421地址配置



### 1.2.1 寄存器映像

请参考相应器件的数据手册中的存储器映像图。[表 1-1](#) 列出了所用 AT32F421 中外设的起始地址。

表格 1-1 寄存器组起始地址

起始地址	外设	总线	寄存器映像
0x6000 0000 - 0xFFFF FFFF	保留	AHB	
0x5004 0000 - 0x5FFF FFFF	保留		
0x5000 0000 - 0x5003 FFFF	保留		
0x4800 1400 - 0x4800 17FF	GPIOF		<a href="#">参见 6.4 节</a>
0x4800 0C00 - 0x4800 13FF	保留		
0x4800 0800 - 0x4800 0BFF	GPIOC		<a href="#">参见 6.4 节</a>
0x4800 0400 - 0x4800 07FF	GPIOB		<a href="#">参见 6.4 节</a>
0x4800 0000 - 0x4800 03FF	GPIOA		<a href="#">参见 6.4 节</a>
0x4002 8000 - 0x47FF FFFF	保留		
0x4002 3400 - 0x4002 7FFF	保留		
0x4002 3000 - 0x4002 33FF	CRC		<a href="#">参见 5.4 节</a>
0x4002 2000 - 0x4002 23FF	闪存存储器接口 (EFC)		<a href="#">参见 4.4 节</a>
0x4002 1400 - 0x4002 1FFF	保留		
0x4002 1000 - 0x4002 13FF	复位和时钟控制 (RCC)		<a href="#">参见 3.3 节</a>
0x4002 0800 - 0x4002 0FFF	保留		
0x4002 0400 - 0x4002 07FF	保留		
0x4002 0000 - 0x4002 03FF	DMA	APB2	<a href="#">参见 9.4 节</a>
0x4001 8400 - 0x4001 7FFF	保留		
0x4001 8000 - 0x4001 83FF	保留		
0x4001 7C00 - 0x4001 7FFF	保留		
0x4001 7800 - 0x4001 7BFF	保留		
0x4001 7400 - 0x4001 77FF	保留		
0x4001 7000 - 0x4001 73FF	保留		
0x4001 6C00 - 0x4001 6FFF	保留		
0x4001 6800 - 0x4001 6BFF	保留		
0x4001 6400 - 0x4001 67FF	保留		
0x4001 6000 - 0x4001 63FF	保留		
0x4001 5C00 - 0x4001 5FFF	保留		
0x4001 5800 - 0x4001 5BFF	保留		
0x4001 5400 - 0x4001 57FF	保留		
0x4001 5000 - 0x4001 53FF	保留		
0x4001 4C00 - 0x4001 4FFF	保留		
0x4001 4800 - 0x4001 4BFF	TMR17 定时器		<a href="#">参见 10.5.4 节</a>
0x4001 4400 - 0x4001 47FF	TMR16 定时器		<a href="#">参见 10.5.4 节</a>
0x4001 4000 - 0x4001 43FF	TMR15 定时器		<a href="#">参见 10.4.4 节</a>
0x4001 3C00 - 0x4001 4BFF	保留		

0x4001 3800 - 0x4001 3BFF	USART1	APB1	<a href="#">参见 16.6 节</a>
0x4001 3400 - 0x4001 37FF	保留		
0x4001 3000 - 0x4001 33FF	SPI1/I2S1		<a href="#">参见 17.4 节</a>
0x4001 2C00 - 0x4001 2FFF	TMR1 定时器		<a href="#">参见 10.6.4 节</a>
0x4001 2800 - 0x4001 2BFF	保留		
0x4001 2400 - 0x4001 27FF	ADC1		<a href="#">参见 14.4 节</a>
0x4001 2000 - 0x4001 23FF	保留		
0x4001 1C00 - 0x4001 1FFF	保留		
0x4001 1800 - 0x4001 1BFF	保留		
0x4001 1400 - 0x4001 17FF	保留		
0x4001 1000 - 0x4001 13FF	保留		
0X4001 0C00 - 0x4001 0FFF	保留		
0x4001 0800 - 0x4001 0BFF	保留		
0x4001 0400 - 0x4001 07FF	EXTI		<a href="#">参见 8.3 节</a>
0x4001 0000 - 0x4001 03FF	SYSCFG/COMP		<a href="#">参见 7.2 节</a>
0x4000 8000 - 0x4000 FFFF	保留		
0x4000 7C00 - 0x4000 7FFF	保留		
0x4000 7800 - 0x4000 7BFF	保留		
0x4000 7400 - 0x4000 77FF	保留		
0x4000 7000 - 0x4000 73FF	电源控制 (PWR)		<a href="#">参见 2.4 节</a>
0x4000 6800 - 0x4000 6BFF	保留		
0x4000 6400 - 0x4000 67FF	保留		
0x4000 6000 - 0x4000 63FF	保留		
0x4000 5C00 - 0x4000 5FFF	保留		
0x4000 5800 - 0x4000 5BFF	I2C2		<a href="#">参见 15.4 节</a>
0x4000 5400 - 0x4000 57FF	I2C1		<a href="#">参见 15.4 节</a>
0x4000 5000 - 0x4000 53FF	保留		
0x4000 4C00 - 0x4000 4FFF	保留		
0x4000 4800 - 0x4000 4BFF	保留		
0x4000 4400 - 0x4000 47FF	USART2		<a href="#">参见 16.6 节</a>
0x4000 4000 - 0x4000 43FF	保留		
0x4000 3C00 - 0x4000 3FFF	保留		
0x4000 3800 - 0x4000 3BFF	SPI2/I2S2		<a href="#">参见 17.4 节</a>
0x4000 3400 - 0x4000 37FF	保留		
0x4000 3000 - 0x4000 33FF	独立看门狗 (IWDG)		<a href="#">参见 12.2.4 节</a>
0x4000 2C00 - 0x4000 2FFF	窗口看门狗 (WWDG)		<a href="#">参见 12.1.6 节</a>
0x4000 2800 - 0x4000 2BFF	ERTC		<a href="#">参见 13.6 节</a>
0x4000 2400 - 0x4000 27FF	保留		
0x4000 2000 - 0x4000 23FF	TMR14 定时器		<a href="#">参见 10.3.4 节</a>
0x4000 1C00 - 0x4000 1FFF	保留		

0x4000 1800 - 0x4000 1BFF	保留	
0x4000 1400 - 0x4000 17FF	保留	
0x4000 1000 - 0x4000 13FF	TMR6 定时器	参见 <a href="#">10.1.4 节</a>
0x4000 0C00 - 0x4000 0FFF	保留	
0x4000 0800 - 0x4000 0BFF	保留	
0x4000 0400 - 0x4000 07FF	TMR3 定时器	参见 <a href="#">10.2.4 节</a>
0x4000 0000 - 0x4000 03FF	保留	

## 1.2.2 位绑定

Cortex™-M4 存储器映像包括两个位段 (bit-band) 区。这两个位段区将别名存储器区中的每个字映射到位段存储器区的一个位，在别名存储区写入一个字具有对位段区的目标位执行读-改-写操作的相同效果。在 AT32F421 系列里，外设寄存器和 SRAM 都被映射到一个位段区里，这允许执行单一的位段的写和读操作。

下面的映射公式给出了别名区中的每个字是如何对应位带区的相应位的：

位别名地址 = 位绑定地址 + (字节偏移 × 32) + (位偏移 × 4)。

其中：

位别名地址：位别名存储器区中字的地址，它映像到某个目标位。

位绑定地址：位别名区的起始地址。

字节偏移：包含目标位的字节在位段里的序号。

位偏移：目标位所在位置 (0-31)。

举例说明，若想对映射别名区中 SRAM 地址为 0x2000\_0200 的字节中的位 2 作位级操作：

位别名地址 = 0x2200\_0000 + (0x200 × 32) + (2 × 4) = 0x2200\_4008

如果对 0x2200\_4008 地址的写操作，与对 SRAM 中地址 0x2000\_0200 字节的位 2 执行读-改-写操作有着相同的效果。如果对 0x2200\_4008 地址进行读操作，将返回 SRAM 中地址 0x2000\_0200 字节的位 2 的值 (0x0000\_0001 或 0x0000\_0000)。请参阅《ARM® Cortex-M4 技术参考手册》了解关于位绑定更详尽信息。

## 1.2.3 片上 SRAM

AT32F421 系列内置 16K 字节的片上 SRAM，起始地址为 0x2000\_0000。它可以以字节、半字 (16 位) 或全字 (32 位) 访问。

## 1.2.4 片上 Flash

AT32F421 系列提供最大 64KB 的片上闪存，支持零等待延时的单周期 32 位读取操作。Flash 存储器的组织可划分为 **主存储块** 与 **信息块**：主存储块用于存放应用程序代码，可按字节 (8 位对齐) 访问、半字节 (16 位对齐)、全字 (32 位对齐) 访问。存储器可单次做一个字节 (8 位)、半字节 (16 位) 或一个全字 (32 位) 编程。64KB 及以下型号主存储块由每页为 1KB 的分页组成，每个分页都可以个别抹除，闪存存储器也支持单次全片抹除 (但不会抹除信息块)。信息块则包含系统存储器以及选项字节两个区段。系统存储器用于存储引导加载程序，于出厂时已编程，用户不可更改。而选项字节则存放用户可修改的选项。详细组织可参阅下表。

闪存存储器由闪存控制器操作，该控制器提供预取缓冲、选项字节加载、闪存编成与抹除以及读取/写入保护等功能。有关闪存控制器的操作与寄存器配置信息请参考[第 4 章节](#)。

表格 1-2 64KB 闪存模块的组织

块		名称	地址范围	长度 (字节)
主存储器	块 1	页 0	0x0800 0000 – 0x0800 03FF	1K

	64KB	页 1	0x0800 0400 – 0x0800 07FF	1K	
		页 2	0x0800 0800 – 0x0800 0BFF	1K	
		页 3	0x0800 0C00 – 0x0800 0FFF	1K	
		页 4	0x0800 1000 – 0x0800 13FF	1K	
		.	.	.	
		页 63	0x0800 FC00 – 0x0800 FFFF	1K	
信息块		系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K	
		用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512	
闪存存储器 接口寄存器		FLASH_ACR	0x4002 2000 – 0x4002 2003	4	
		FLASH_FCKEY	0x4002 2004 – 0x4002 2007	4	
		FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4	
		FLASH_STS	0x4002 200C – 0x4002 200F	4	
		FLASH_CTRL	0x4002 2010 – 0x4002 2013	4	
		FLASH_ADDR	0x4002 2014 – 0x4002 2017	4	
		保留	0x4002 2018 – 0x4002 201B	4	
		FLASH_UOB	0x4002 201C – 0x4002 201F	4	
		FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4	
		保留	0x4002 2024 - 0x4002 2073	80	
		FLASH_CDR0	0x4002 2074 - 0x4002 2077	4	
		FLASH_CDR1	0x4002 2078 - 0x4002 207B	4	
		FSLIB_PSW	0x4002 207C - 0x4002 207F	4	
		FSLIB_PSW_STS	0x4002 2080 - 0x4002 2083	4	
		FLASH_CRC_AR	0x4002 2084 - 0x4002 2087	4	
		FLASH_CRC_CTRL	0x4002 2088 - 0x4002 208B	4	
		FLASH_CRC_OUTR	0x4002 208C - 0x4002 208F	4	
		保留	0x4002 2090 - 0x4002 215F	208	
		FSLIB_SET_PSW	0x4002 2160 - 0x4002 2163	4	
		FSLIB_SET_RANGE	0x4002 2164 - 0x4002 2167	4	
		SYS_MEM_SLIB_SET	0x4002 2168 - 0x4002 216B	4	
		SYS_MEM_BOOT_DIS_SET	0x4002 216C - 0x4002 216F	4	
		FSLIB_KEYR	0x4002 2170 - 0x4002 2173	4	

表格 1-3 32KB闪存模块的组织

块		名称	地址范围	长度（字节）	
主存储器	块 1 32KB	页 0	0x0800 0000 – 0x0800 03FF	1K	
		页 1	0x0800 0400 – 0x0800 07FF	1K	
		页 2	0x0800 0800 – 0x0800 0BFF	1K	
		页 3	0x0800 0C00 – 0x0800 0FFF	1K	
		页 4	0x0800 1000 – 0x0800 13FF	1K	
		.	.	.	
		页 31	0x0800 7C00 – 0x0800 7FFF	1K	
信息块		系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K	
		用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512	
闪存存储器 接口寄存器		FLASH_ACR	0x4002 2000 – 0x4002 2003	4	
		FLASH_FCKEY	0x4002 2004 – 0x4002 2007	4	

	FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4
	FLASH_STS	0x4002 200C – 0x4002 200F	4
	FLASH_CTRL	0x4002 2010 – 0x4002 2013	4
	FLASH_ADDR	0x4002 2014 – 0x4002 2017	4
	保留	0x4002 2018 – 0x4002 201B	4
	FLASH_UOB	0x4002 201C – 0x4002 201F	4
	FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4
	保留	0x4002 2024 - 0x4002 2073	80
	FLASH_CDR0	0x4002 2074 - 0x4002 2077	4
	FLASH_CDR1	0x4002 2078 - 0x4002 207B	4
	FSLIB_PSW	0x4002 207C - 0x4002 207F	4
	FSLIB_PSW_STS	0x4002 2080 - 0x4002 2083	4
	FLASH_CRC_AR	0x4002 2084 - 0x4002 2087	4
	FLASH_CRC_CTRL	0x4002 2088 - 0x4002 208B	4
	FLASH_CRC_OUTR	0x4002 208C - 0x4002 208F	4
	保留	0x4002 2090 - 0x4002 215F	208
	FSLIB_SET_PSW	0x4002 2160 - 0x4002 2163	4
	FSLIB_SET_RANGE	0x4002 2164 - 0x4002 2167	4
	SYS_MEM_SLIB_SET	0x4002 2168 - 0x4002 216B	4
	SYS_MEM_BOOT_DIS_SET	0x4002 216C - 0x4002 216F	4
	FSLIB_KEYR	0x4002 2170 - 0x4002 2173	4

表格 1-4 16KB闪存模块的组织

块	名称	地址范围	长度（字节）
主存储器	页 0	0x0800 0000 – 0x0800 03FF	1K
	页 1	0x0800 0400 – 0x0800 07FF	1K
	页 2	0x0800 0800 – 0x0800 0BFF	1K
	页 3	0x0800 0C00 – 0x0800 0FFF	1K
	页 4	0x0800 1000 – 0x0800 13FF	1K
	.	.	.
	页 15	0x0800 3C00 – 0x0800 3FFF	1K
	信息块		
	系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K
	用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512
闪存存储器 接口寄存器	FLASH_ACR	0x4002 2000 – 0x4002 2003	4
	FLASH_FCKEY	0x4002 2004 – 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4
	FLASH_STS	0x4002 200C – 0x4002 200F	4
	FLASH_CTRL	0x4002 2010 – 0x4002 2013	4
	FLASH_ADDR	0x4002 2014 – 0x4002 2017	4
	保留	0x4002 2018 – 0x4002 201B	4
	FLASH_UOB	0x4002 201C – 0x4002 201F	4
	FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4
	保留	0x4002 2024 - 0x4002 2073	80
	FLASH_CDR0	0x4002 2074 - 0x4002 2077	4
	FLASH_CDR1	0x4002 2078 - 0x4002 207B	4
	FSLIB_PSW	0x4002 207C - 0x4002 207F	4
	FSLIB_PSW_STS	0x4002 2080 - 0x4002 2083	4

	FLASH_CRC_AR	0x4002 2084 - 0x4002 2087	4
	FLASH_CRC_CTRL	0x4002 2088 - 0x4002 208B	4
	FLASH_CRC_OUTR	0x4002 208C - 0x4002 208F	4
	保留	0x4002 2090 - 0x4002 215F	208
	FSLIB_SET_PSW	0x4002 2160 - 0x4002 2163	4
	FSLIB_SET_RANGE	0x4002 2164 - 0x4002 2167	4
	SYS_MEM_SLIB_SET	0x4002 2168 - 0x4002 216B	4
	SYS_MEM_BOOT_DIS_SET	0x4002 216C - 0x4002 216F	4
	FSLIB_KEYR	0x4002 2170 - 0x4002 2173	4

## 1.3 引导配置

在 AT32F421 里，可以通过 nBOOT1 和 BOOT0 引脚选择三种不同引导模式，如下表所示。

表格 1-5 启动模式

启动模式	说明	引脚配置	
		nBOOT1	BOOT0
主闪存存储器	主闪存存储器被选为启动区域	X	0
系统存储器	系统存储器被选为启动区域	0	1
片上 SRAM	内置 SRAM 被选为启动区域	1	1

在系统复位后，启动模式配置的值将被锁存。用户可以通过设置启动模式配置值，来选择在复位后的启动模式。从待机模式退出时，启动模式配置值将被重新锁存；因此，在待机模式下启动模式配置值应保持为需要的启动配置。

根据选定的启动模式，主闪存存储器、系统存储器或 SRAM 可以按照以下方式访问：

- 从主闪存存储器启动：主闪存存储器被映射到启动空间（0x0000 0000），但仍然能够在它原有的地址（0x0800 0000）访问它，即闪存存储器的内容可以在两个地址区域访问，0x0000 0000 或 0x0800 0000。
- 从系统存储器启动：系统存储器被映射到启动空间（0x0000 0000），但仍然能够在它原有的地址（0x1FFF E400）访问它。
- 从内置 SRAM 启动：只能在 0x2000 0000 开始的地址区访问 SRAM。

在启动延迟之后，CPU 从地址 0x0000 0000 获取堆栈顶的地址，并从启动存储器的 0x0000\_0004 指示的地址开始执行代码。因为固定的存储器映像，代码区始终从地址 0x0000\_0000 开始（通过 ICode 和 DCode 总线访问），而数据区（SRAM）始终从地址 0x2000\_0000 开始（通过系统总线访问）。Cortex™-M4 的 CPU 始终从 ICode 总线获取复位向量，即启动仅适合于从代码区开始（典型地从 Flash 启动）。

AT32F421 系列微控制器实现了一个特殊的机制，可以从片上 SRAM 启动。当从片上 SRAM 启动，在应用程序的初始化代码中，必须使用 NVIC 的异常表和偏移寄存器，从新映射向量表之 SRAM 中。

系统存储器中包含内嵌的引导加载程序，可用于对闪存存储器编程。引导加载程序可以通过 USART1 和 USART2 接口对闪存存储器进行重新编程。

## 1.4 器件特征信息

### 1.4.1 寄存器的缩写说明

本手册中对寄存器的描述中将使用下述缩写：

表格 1-6 寄存器缩写表

读/写 (rw)	软件可以读或写这些位。
----------	-------------

只读 (r)	软件只能读这些位。
只写 (w)	软件只能写这些位；如果读这些位，则返回它们的复位值。
读/清除 (rc_w0)	软件可以读并写'0'清除这些位，写'1'将不对该位产生影响。
读/设置 (rs)	软件可以读并写'1'设置这些位，写'0'将不对该位产生影响。
保留 (res)	保留位，必须保持在复位状态。

## 1.4.2 闪存容量寄存器

闪从容量寄存器提供该芯片闪存容量信息，用户可透过该寄存器取得闪存容量。

基地址: 0x1FFF F7E0

复位值: 0xXXXX XXXX (由生产厂设置)

15      14      13      12      11      10      9      8      7      6      5      4      3      2      1      0
F_SIZE
r
位15: 0      F_SIZE: 闪存容量，以 KByte 为单位 例如: 0x0040 = 64KByte

## 1.4.3 器件电子签名

器件电子签名包含产品容量信息和器件唯一 ID (96 位 UID)，它位于闪存的信息区块中。96 位器件唯一 ID 对任何器件来说都是独一无二的，且用户不可更改。ID 可以用来作为下列用途：

- 序列号
- 或者做为密钥的一部分

基地址: 0x1FFF F7E8

复位值: 0xXXXX XXXX (由生产厂设置)

31      30      29      28      27      26      25      24      23      22      21      20      19      18      17      16
UID[31: 16]
r
15      14      13      12      11      10      9      8      7      6      5      4      3      2      1      0
UID[15: 0]
r

基地址: 0x1FFF F7EC

复位值: 0xXXXX XXXX (由生产厂设置)

31      30      29      28      27      26      25      24      23      22      21      20      19      18      17      16
UID[63: 48]
r
15      14      13      12      11      10      9      8      7      6      5      4      3      2      1      0
UID[47: 32]
r

基地址: 0x1FFF F7F0

复位值: 0xXXXX XXXX (由生产厂设置)

31            30            29            28            27            26            25            24

30

29

28

27

26

25

24

UID[95: 88]

r

[23:0]

UID[87: 64]

r

注：U\_ID[95:88]为 Series ID，为 0x09。

## 2 电源控制 (PWR)

### 2.1 简介

功耗是 AT32F421 系列设备中最重要的问题之一, AT32F421 系列设备可工作电压范围为 2.4 V 至 3.6 V, 且可在 -40~+105°C 温度范围内正常工作。为了减少功耗, 且使得应用程序可以在 CPU 运行时间要求、速度和功耗的相互冲突中获得最佳折衷, 电源控制中提供了三种省电模式, 包括睡眠模式, 停机模式和待机模式。AT32F421 系列设备有两个电源域, 包括 VDD/VDDA 域和 1.2 V 域。VDD/VDDA 域由电源直接供电, 但 VDDA 和 VSSA 提供分离的模拟模块电源, 降低电源噪声干扰。在 VDD/VDDA 域中嵌入了一个 LDO, 用来为 1.2 V 域供电。

### 2.2 主要特点

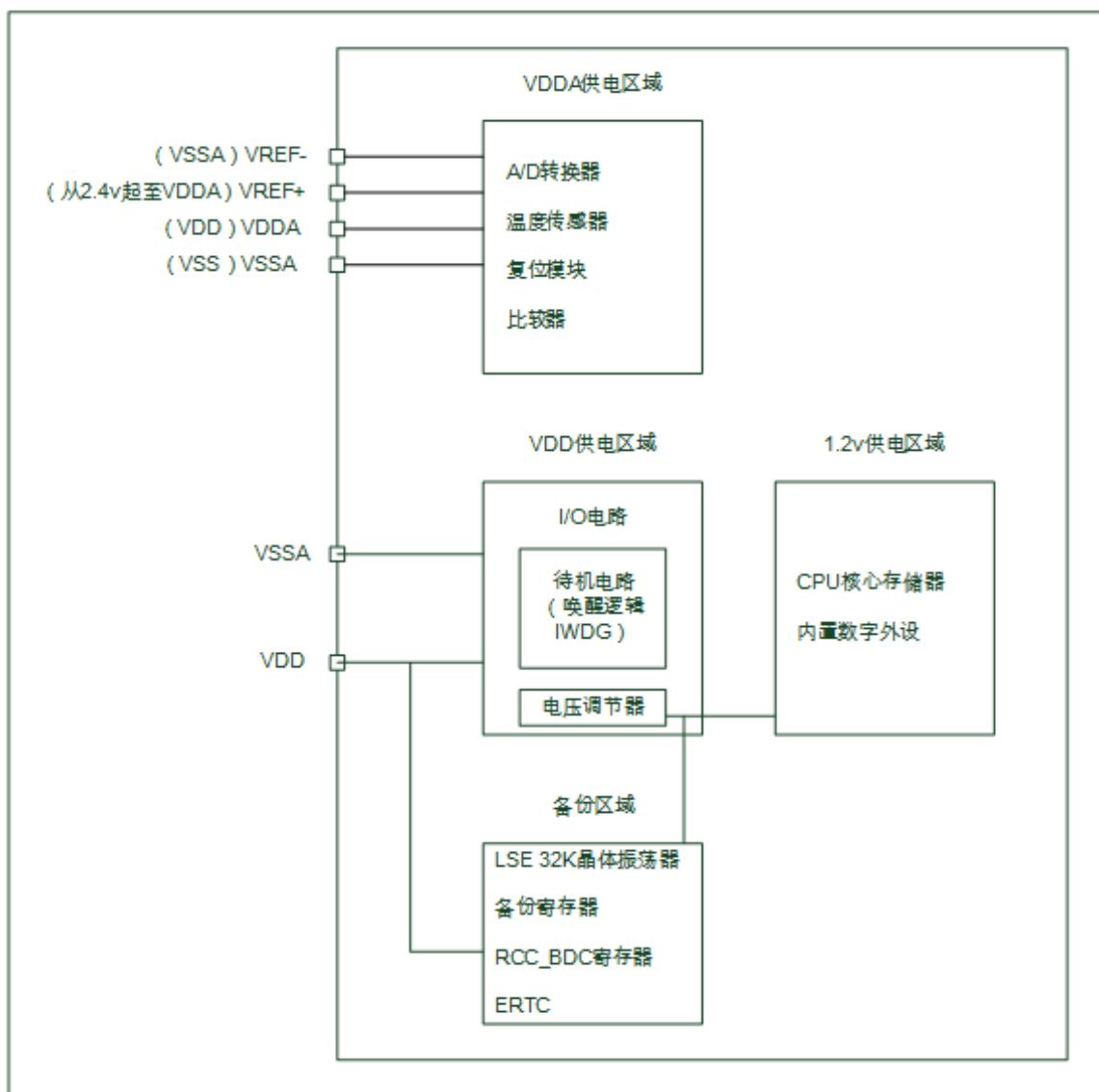
- 具备两个电源: VDD/VDDA 域、1.2 V 内核域。
- 支持三种省电模式: 睡眠模式、停机模式和待机模式。
- 内建电压调节器提供 1.2 V 给内核域。
- 提供电压检测器, 能于电压低于阈值时发出中断。

### 2.3 功能描述

#### 2.3.1 电源域

AT32F421 的工作电压 (VDD) 为 2.4~3.6 V。通过内置的电压调节器提供所需的 1.2 V 电源。

图表 2-1 各电源域框图



### 2.3.1.1 VDD/VDDA电源域

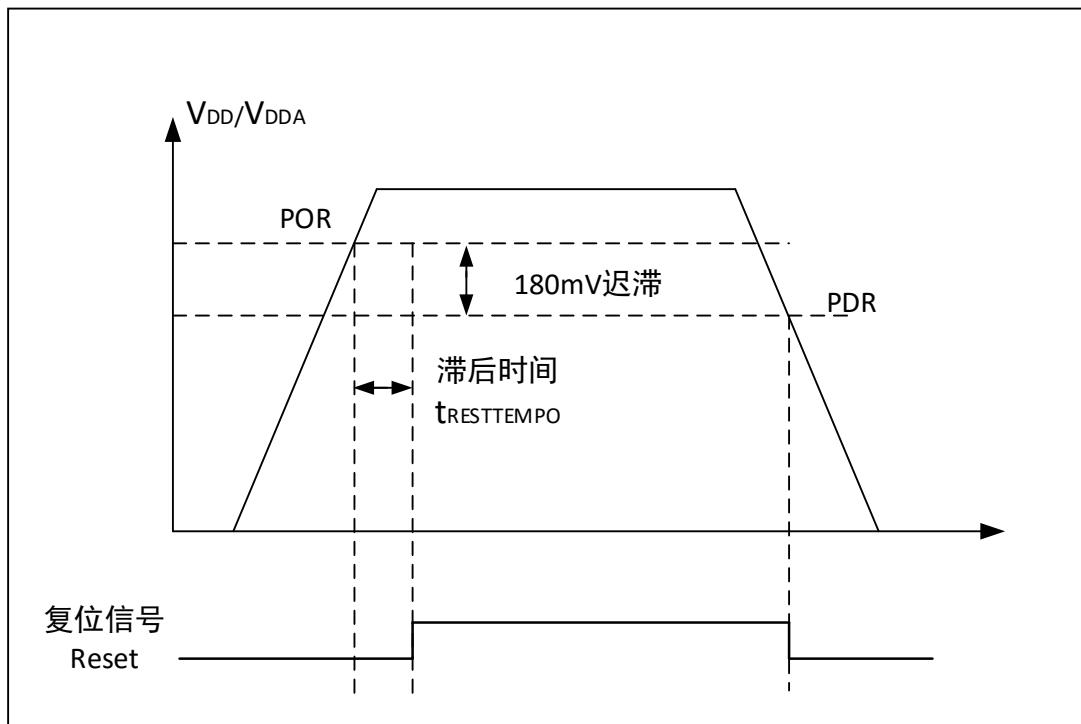
AT32F421 的工作电压 (VDD) 为 2.4~3.6V。通过内置的电压调节器提供所需的 1.2V 电源。

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

ADC 的电源引脚为 VDDA，独立的电源地 VSSA。

AT32F421 内部有一个完整的上电复位 (POR) 和掉电复位 (PDR) 电路，当供电电压达到 2.4V 时系统既能正常工作。当 VDD/VDDA 低于指定的限位电压 VPOR/VPDR 时，系统保持为复位状态，而无需外部复位电路。关于上电复位和掉电复位的细节请参考数据手册的电气特性部分。

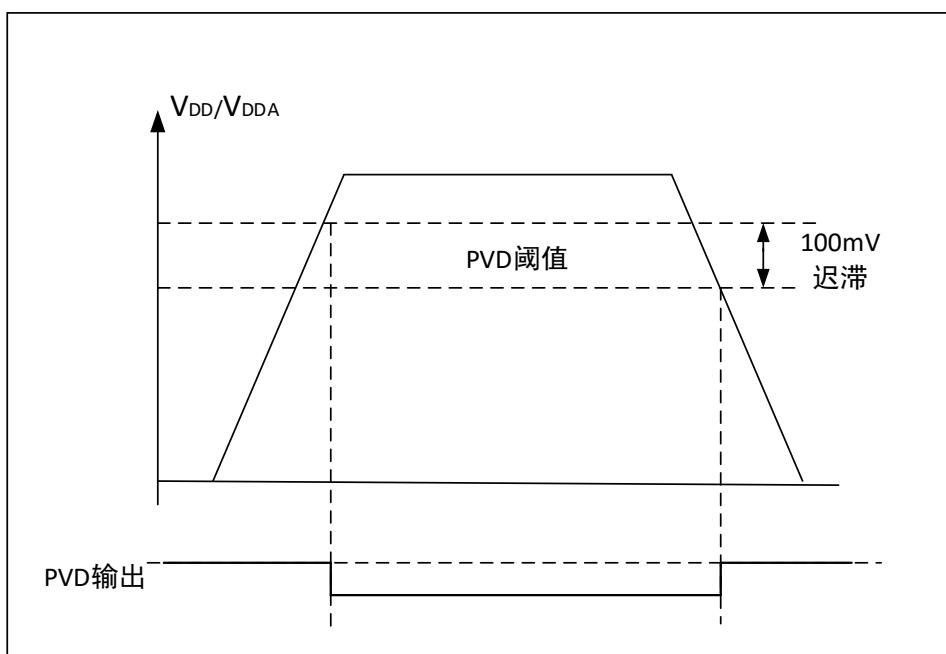
图表 2-2 上电复位和掉电复位的波形图



AT32F421 提供一个程序电压侦测器 PVD，用户可以利用 PVD 对 VDD 电压与电源控制寄存器（PWR\_CTRL）中的 PVDS[2: 0]位进行比较来监控电源，选择监控电压的阈值。

通过设置 PVDEN 位来使能 PVD。电源控制/状态寄存器（PWR\_CTRLSTS）中的 PVD 标志用来表明 VDD 是高于还是低于 PVD 的电压阈值。该事件在内部连接到外部中断的第 16 线，如果该中断在外部中断寄存器中是使能的，该事件就会产生中断。当 VDD 下降到 PVD 阈值以下和（或）当 VDD 上升到 PVD 阈值之上时，根据外部中断第 16 线的上升/下降边沿触发设置，就会产生 PVD 中断。例如，这一特性可用于执行紧急关闭任务。

图表 2-3 PVD的阈值与输出



### 2.3.1.2 内核电源域

内核电源域包含 CPU 核心、内存与内置数字外设。该电源域由一电压调节器供电。调节器于复位后总是使能的。根据应用方式它以 3 种不同的模式工作。

运转模式：调节器以正常功耗模式提供 1.2V 电源（内核，内存和外设）。

停机模式：调节器以低功耗模式提供 1.2V 电源，以保存寄存器和 SRAM 的内容。

待机模式：调节器停止供电。除了备用电路和备份域外，寄存器和 SRAM 的内容全部丢失。

备份区域由 VDD 供电：

- PC14 和 PC15 可以用于 GPIO 或 LSE 引脚；
- PC13 可以作为通用 I/O 口、TAMPER 引脚、ERTC 校准时钟、ERTC 闹钟、唤醒或秒输出。

## 2.3.2 低功耗模式

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

AT32F421 有三种低功耗模式：

- 睡眠模式（Cortex™-M4 内核停止，所有外设包括 Cortex™-M4 核心的外设，如 NVIC、系统时钟（SysTick）等仍在运行）；
- 停机模式（所有的时钟都已停止）；
- 待机模式（1.2V 电源关闭）。

模式	进入	唤醒	对 1.2V 区域时钟的影响	对 VDD 区域时钟的影响	电压调节器
睡眠 (SLP-NOW 或 SLP-ON-EXIT)	WFI	任一中断	CPU 时钟关，对其他时钟和 ADC 时钟无影响	无	开
	WFE	唤醒事件			
停机	PDDS 位 +SLEEPDEEP 位 +WFI 或 WFE	任一外部中断 (在外部中断寄存器中设置)	关闭所有 1.2V 区域的时钟	HSI 和 HSE 的振荡器关闭	开启或处于低功耗模式（依据电源控制寄存器（PWR_CTRL）的设定）
待机	PDDS 位 +SLEEPDEEP 位 +WFI 或 WFE	WKUPx 引脚的上升沿、ERTC 闹钟事件、NRST 引脚上的外部复位、IWDG 复位			关

此外，在运行模式下，可以通过以下方式中的一种降低功耗：

- 降低系统时钟。在运行模式下，通过对预分频寄存器进行编程，可以降低任意一个系统时钟（SYSCLK、HCLK、PCLK1、PCLK2）的速度。进入睡眠模式前，也可以利用预分频器来降低外设的时钟。详见[第3.3.2节：时钟配置寄存器（RCC\\_CFG）](#)。
- 关闭 APB 和 AHB 总线上未被使用的外设时钟。在运行模式下，任何时候都可以通过停止为外设和内存提供时钟（HCLK 和 PCLKx）来减少功耗。为了在睡眠模式下更多地减少功耗，可在执行 WFI 或 WFE 指令前关闭所有外设的时钟。通过设置 AHB 外设时钟使能寄存器（RCC\_AHBEN）、APB2 外设时钟使能寄存器（RCC\_APB2EN）和 APB1 外设时钟使能寄存器（RCC\_APB1EN）来开关各个外设模块的时钟。

### 2.3.2.1 睡眠模式

进入睡眠模式：

通过执行 WFI 或 WFE 指令进入睡眠状态。根据 Cortex™-M4 系统控制寄存器中的 SLEEPONEXIT 位的值，有两种选项可用于选择睡眠模式进入机制：

- **SLP-NOW:** 如果 SLEEPONEXIT 位被清除，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。
- **SLP-ON-EXIT:** 如果 SLEEPONEXIT 位被置位，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

在睡眠模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。关于如何进入睡眠模式，更多的细节参考 [表 2-1](#) 和 [表 2-2](#)。

退出睡眠模式：

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

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

- 在外设控制寄存器中使能一个中断，而不是在 NVIC（嵌套向量中断控制器）中使能，并且在 Cortex™-M4 系统控制寄存器中使能 SEVONPEND 位。当 MCU 从 WFE 中唤醒后，外设的中断挂起位和外设的 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）必须被清除。
- 配置一个外部或内部的 EXTI 线为事件模式。当 MCU 从 WFE 中唤醒后，因为与事件线对应的挂起位未被设置，不必清除外设的中断挂起位或外设的 NVIC 中断通道挂起位。

该模式唤醒所需的时间最短，因为没有时间损失在中断的进入或退出上。关于如何退出睡眠模式，更多的细节参考 [表 2-1](#) 和 [表 2-2](#)。

表格 2-1 SLP-NOW 模式

SLP-NOW 模式	说明
进入	在以下条件下执行 WFI（等待中断）或 WFE（等待事件）指令： - SLEEPDEEP = 0 和 - SLEEPONEXIT = 0 参考 Cortex™-M4 系统控制寄存器。
退出	如果执行 WFI 进入睡眠模式： 中断：参考中断向量表 ( <a href="#">表 8-1</a> ) 如果执行 WFE 进入睡眠模式： 唤醒事件：参考唤醒事件管理 ( <a href="#">第 8.2.3 节</a> )
唤醒延时	无

表格 2-2 SLP-ON-EXIT 模式

SLP-ON_EXIT 模式	说明
进入	在以下条件下执行 WFI 指令： - SLEEPDEEP = 0 和 - SLEEPONEXIT = 1 参考 Cortex™-M4 系统控制寄存器
退出	中断：参考中断向量表 ( <a href="#">表 8-1</a> )
唤醒延时	无

### 2.3.2.2 停机模式

停机模式是在 Cortex™-M4 的深睡眠模式基础上结合了外设的时钟控制机制，在停机模式下电压调节器可运行在正常或低功耗模式。此时在 1.2V 供电区域的所有时钟都被停止，SysTick 除外，PLL、HSI 和 HSE RC 振荡器的功能被禁止，SRAM 和寄存器内容被保留下。

在停机模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。

进入停机模式

关于如何进入停机模式，详见[表 2-3](#)。

如果正在进行闪存编程，直到对内存访问完成，系统才进入停机模式。如果正在进行对 APB 的访问，直到对 APB 访问完成，系统才进入停机模式。可以通过对独立的控制位进行编程，可选择以下功能：

- 独立看门狗（IWDG）：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。详见[12.2.3 节](#)。
- 实时时钟（ERTC）：通过备份域控制寄存器（RCC\_BDC）的 ERTCEN 位来设置。
- 内部 RC 振荡器（LSI RC）。
- 外部 32.768kHz 振荡器（LSE）：通过备份域控制寄存器（RCC\_BDC）的 LSEEN 位设置。在停机模式下，如果在进入该模式前 ADC 没有被关闭，那么这些外设仍然消耗电流。通过设置寄存器 ADC\_CTRL2 的 ADON 位可关闭这个外设。

### 退出停机模式

关于如何退出停机模式，详见下表。当一个中断或唤醒事件导致退出停机模式时，HSI RC 振荡器被选为系统时钟。

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

表格 2-3 停机模式

停机模式	说明
进入	<p>在以下条件下执行 WFI（等待中断）或 WFE（等待事件）指令：</p> <ul style="list-style-type: none"> <li>- 设置 Cortex™-M4 系统控制寄存器中的 SLEEPDEEP 位</li> <li>- 清除电源控制寄存器（PWR_CTRL）中的 PDDS 位</li> </ul> <p>注：为了进入停机模式，所有的外部中断的请求位（挂起寄存器（EXTI_PND））和 ERTC 的闹钟标志都必须被清除，否则停机模式的进入流程将被跳过，程序继续运行。</p>
退出	<p>如果执行 WFI 进入停机模式：</p> <p>设置任一外部中断线为中断模式（在 NVIC 中必须使能相应的外部中断向量）。参见中断向量表（<a href="#">表 8-1</a>）。</p> <p>如果执行 WFE 进入停机模式：</p> <p>设置任一外部中断线为事件模式。参见唤醒事件管理（<a href="#">第 8.2.3 节</a>）。</p>
唤醒延时	HSI RC 唤醒时间

### 2.3.2.3 待机模式

待机模式可实现系统的最低功耗。该模式是在 Cortex™-M4 深睡眠模式时关闭电压调节器。整个 1.2V 供电区域被断电。PLL、HSI 和 HSE 振荡器也被断电。SRAM 和寄存器内容丢失。只有备份的寄存器和待机电路维持供电（见[图 2-1](#)）。

#### 进入待机模式

关于如何进入待机模式，详见[表 2-4](#)。可以通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗（IWDG）：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。详见[12.2.3 节](#)。
- 实时时钟（ERTC）：通过备用区域控制寄存器（RCC\_BDC）的 ERTCEN 位来设置。
- 内部 RC 振荡器（LSI RC）
- 外部 32.768kHz 振荡器（LSE）：通过备用区域控制寄存器（RCC\_BDC）的 LSEEN 位设置。

### 退出待机模式

当一个外部复位 (NRST 引脚)、IWDG 复位、WKUPx 引脚上的上升沿或 ERTC 闹钟事件的上升沿发生时 (见图 13-1: 简化的 ERTC 框图), 微控制器从待机模式退出。从待机唤醒后, 除了电源控制/状态寄存器 (PWR\_CTRLSTS) (见第 2.4.2 节), 所有寄存器被复位。从待机模式唤醒后的代码执行等同于复位后的执行(采样启动模式引脚、读取复位向量等)。电源控制/状态寄存器 (PWR\_CTRLSTS) (见第 2.4.2 节) 将会指示内核由待机状态退出。

待机模式下的输入/输出端口状态在待机模式下, 所有的 I/O 引脚处于高阻态, 除了以下的引脚:

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

表格 2-4 待机模式

待机模式	说明
进入	在以下条件下执行 WFI (等待中断) 或 WFE (等待事件) 指令: - 设置 Cortex™-M4 系统控制寄存器中的 SLEEPDEEP 位 - 设置电源控制寄存器 (PWR_CTRL) 中的 PDSS 位 - 清除电源控制/状态寄存器 (PWR_CTRLSTS) 中的 WUF 位
退出	WKUPx 引脚的上升沿、ERTC 闹钟事件的上升沿、NRST 引脚上外部复位、IWDG 复位。
唤醒延时	复位持续时间 TRSTTEMPO

### 2.3.2.4 调试模式

默认情况下, 如果在进行调试微处理器时, 使微处理器进入停机或待机模式, 将失去调试连接。这是因为 Cortex™-M4 的内核失去了时钟。然而, 通过设置 MCUDBG\_CTRL 寄存器中的某些配置位, 可以在使用低功耗模式下调试软件。

更多的细节请参考[第 18.2.1 节: 低功耗模式的调试支持](#)。

### 2.3.3 自动唤醒

ERTC 可以在不需要依赖外部中断的情况下唤醒低功耗模式下的微控制器 (自动唤醒模式)。ERTC 提供一个可编程的时间基数, 用于周期性从停机或待机模式下唤醒。通过对备份区域控制寄存器 (RCC\_BDC) 的 ERTCSSEL[1: 0]位的编程, 三个 ERTC 时钟源中的二个时钟源可以选作实现此功能。

#### 低功耗 32.768kHz 外部晶振 (LSE)

该时钟源提供了一个低功耗且精确的时间基准。(在典型情形下消耗小于 1µA)

#### 低功耗内部 RC 振荡器 (LSI RC)

使用该时钟源, 节省了一个 32.768kHz 晶振的成本。但是 RC 振荡器将少许增加电源消耗。

为了用 ERTC 闹钟事件将系统从停机模式下唤醒, 必须进行如下操作:

配置外部中断线 17 为上升沿触发。

配置 ERTC 使其可产生 ERTC 闹钟事件。

如果要从待机模式中唤醒, 不必配置外部中断线 17。

## 2.4 PWR 寄存器

可以用半字 (16位) 或字 (32位) 的方式操作这些外设寄存器。

表格 2-5 PWR 寄存器的映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

0x00	PWR_CTRL	保留		DBP	PVDS[2: 0]		PVDEN	CLSBF	CLWUF	PDDS	LPDS
					0	0	0	0	0	0	0
0x04	PWR_CTRL_STS	保留		WUPEN7	WUPEN6	保留	WUPEN1	WUPEN2	保留	PVD	SBF
				0	0		0	0		0	0
0x20	PWR_CTRL_2	保留		LPDS1	保留						

## 2.4.1 电源控制寄存器 (PWR\_CTRL)

地址偏移: 0x000

复位值: 0x0000 0000 (从待机模式唤醒时清除)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				DBP	PVDS[2: 0]		PVDEN	CLSBF	CLWUF	PDDS	LPDS				
r				rw		rw		rw		rc_w1		rc_w1		rw	

位 31: 9	保留。始终读为 0。
位 8	<p><b>DBP:</b> 取消后备区域的写保护 在复位后, ERTC 和后备寄存器处于被保护状态以防意外写入。设置这位允许写入这些寄存器。 0: 禁止写入 ERTC 和后备寄存器 1: 允许写入 ERTC 和后备寄存器 注: 如果 ERTC 的时钟是 HSE/128, 该位必须保持为'1'。</p>
位 7: 5	<p><b>PVDS[2: 0]: PVD 电平选择</b> 这些位用于选择电源电压监测器的电压阈值 000: 保留 100: 2.6V 001: 2.3V 101: 2.7V 010: 2.4V 110: 2.8V 011: 2.5V 111: 2.9V 注: 详细说明参见数据手册中的电气特性部分。</p>
位 4	<p><b>PVDEN:</b> 电源电压监测器 (PVD) 使能 0: 禁止 PVD 1: 开启 PVD</p>
位 3	<p><b>CLSBF:</b> 清除待机位 始终读出为 0 0: 无功效 1: 清除 SBF 待机位 (写)</p>

位 2	CLWUF: 清除唤醒位 始终读出为 0 0: 无功效 1: 2 个系统时钟周期后清除 WUF 唤醒位 (写)
位 1	PDDS: 掉电深睡眠 0: CPU 进入深睡眠时进入停机模式 1: CPU 进入深睡眠时进入待机模式
位 0	LPDS: 深睡眠下的低功耗使能位, 在 PDDS=0 时有效 0: 在停机模式下电压调压器开启 1: 在停机模式下电压调压器处于低功耗模式

## 2.4.2 电源控制/状态寄存器 (PWR\_CTRLSTS)

地址偏移: 0x004

复位值: 0x0000 0000 (从待机模式唤醒时不被清除)

与标准的 APB 读相比, 读此寄存器需要额外的 APB 周期

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	WUP EN7	WUP EN6	保留	WUP EN2	WUP EN1	保留	res	r	r	PVD	SBF	WUF	res	rw	rw

位 31: 15	保留。始终读为 0。
位 14	WUPEN7: 使能 WKUP 引脚 7 0: WKUP 引脚 7 为通用 I/O。WKUP 引脚 7 上的事件不能将 CPU 从待机模式唤醒 1: WKUP 引脚 7 用于将 CPU 从待机模式唤醒, WKUP 引脚 7 被强置为输入下拉的配置 (WKUP 引脚 7 上的上升沿将系统从待机模式唤醒) 注: 在系统复位时清除这一位。
位 13	WUPEN6: 使能 WKUP 引脚 6 0: WKUP 引脚 6 为通用 I/O。WKUP 引脚 6 上的事件不能将 CPU 从待机模式唤醒 1: WKUP 引脚 6 用于将 CPU 从待机模式唤醒, WKUP 引脚 6 被强置为输入下拉的配置 (WKUP 引脚 6 上的上升沿将系统从待机模式唤醒) 注: 在系统复位时清除这一位。
位 12: 10	保留。始终读为 0。
位 9	WUPEN2: 使能 WKUP 引脚 2 0: WKUP 引脚 2 为通用 I/O。WKUP 引脚 2 上的事件不能将 CPU 从待机模式唤醒 1: WKUP 引脚 2 用于将 CPU 从待机模式唤醒, WKUP 引脚 2 被强置为输入下拉的配置 (WKUP 引脚 2 上的上升沿将系统从待机模式唤醒) 注: 在系统复位时清除这一位。
位 8	WUPEN1: 使能 WKUP 引脚 1 0: WKUP 引脚 1 为通用 I/O。WKUP 引脚上的事件不能将 CPU 从待机模式唤醒 1: WKUP 引脚 1 用于将 CPU 从待机模式唤醒, WKUP 引脚 1 被强置为输入下拉的配置 (WKUP 引脚 1 上的上升沿将系统从待机模式唤醒) 注: 在系统复位时清除这一位。
位 7: 3	保留。始终读为 0。

位 2	PVD: PVD 输出 当 PVD 被 PVDEN 位使能后该位才有效 0: VDD/VDDA 高于由 PVDS[2: 0]选定的 PVD 阈值 1: VDD/VDDA 低于由 PVDS[2: 0]选定的 PVD 阈值 注: 在待机模式下 PVD 被停止。因此, 待机模式后或复位后, 直到设置 PVDEN 位之前, 该位为 0。
位 1	SBF: 待机标志 该位由硬件设置, 并只能由 POR/PDR (上电/掉电复位) 或设置电源控制寄存器 (PWR_CTRL) 的 CLSBF 位清除。 0: 系统不在待机模式 1: 系统进入待机模式
位 0	WUF: 唤醒标志 该位由硬件设置, 并只能由 POR/PDR (上电/掉电复位) 或设置电源控制寄存器 (PWR_CTRL) 的 CLWUF 位清除。 0: 没有发生唤醒事件 1: 在 WKUPx 引脚上发生唤醒事件或出现 ERTC 闹钟事件。 注: 当 WKUPx 引脚已经是高电平时, 在 (通过设置 WUPENx 位) 使能 WKUPx 引脚时, 会检测到一个额外的事件。

### 2.4.3 电源控制寄存器2 (PWR\_CTRL2)

地址偏移: 0x020

复位值: 0x0000 0000 (从待机模式唤醒时不被清除)

与标准的 APB 读相比, 读此寄存器需要额外的 APB 周期

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								LPDS1		保留					
res		rw		res											

位 31: 6	保留。始终读为 0。
位 5	LPDS1: 内部电压调压器额外低功耗模式使能位, 与 PWR_CTRL 的 PDDS 和 LPDS 位协同工作, 在 LPDS = 1 时, 且芯片进入停机模式时才有效。 0: 内部电压调压器额外低功耗模式关闭。 1: 内部电压调压器额外低功耗模式开启。 注: 如需启动额外低功耗模式, 请先配置 LPDS1, 再配置 PDDS 和 LPDS 位。
位 4: 0	保留。

## 3 复位和时钟控制 (RCC)

### 3.1 复位

AT32F421 支持三种复位形式，分别为系统复位、上电复位和备份区域复位。

#### 3.1.1 系统复位

除了时钟控制器的 RCC\_CTRLSTS 寄存器中的复位标志位和备份区域中的寄存器以外，系统复位将复位所有寄存器至它们的复位状态。

当发生以下任一事件时，产生一个系统复位：

1. NRST 引脚上的低电平（外部复位）
2. 窗口看门狗计数终止（WWDG 复位）
3. 独立看门狗计数终止（IWDG 复位）
4. 软件复位（SW 复位）
5. 低功耗管理复位可通过查看 RCC\_CTRLSTS 控制状态寄存器中的复位状态标志位识别复位事件来源。

软件复位：

通过将 Cortex™-M4 中断应用和复位控制寄存器中的 SYSRESETREQ 位置‘1’，可实现软件复位。请参考 Cortex™-M4 技术参考手册获得进一步信息。

低功耗管理复位：

在以下两种情况下可产生低功耗管理复位：

1. 在进入待机模式时产生低功耗管理复位：通过将用户选择字节中的 nSTDBY\_RST 位置‘1’将使能该复位。这时，即使执行了进入待机模式的过程，系统将被复位而不是进入待机模式。
2. 在进入停机模式时产生低功耗管理复位：通过将用户选择字节中的 nSTP\_RST 位置‘1’将使能该复位。这时，即使执行了进入停机模式的过程，系统将被复位而不是进入停机模式。

#### 3.1.2 电源复位

当以下事件中之一发生时，产生电源复位：

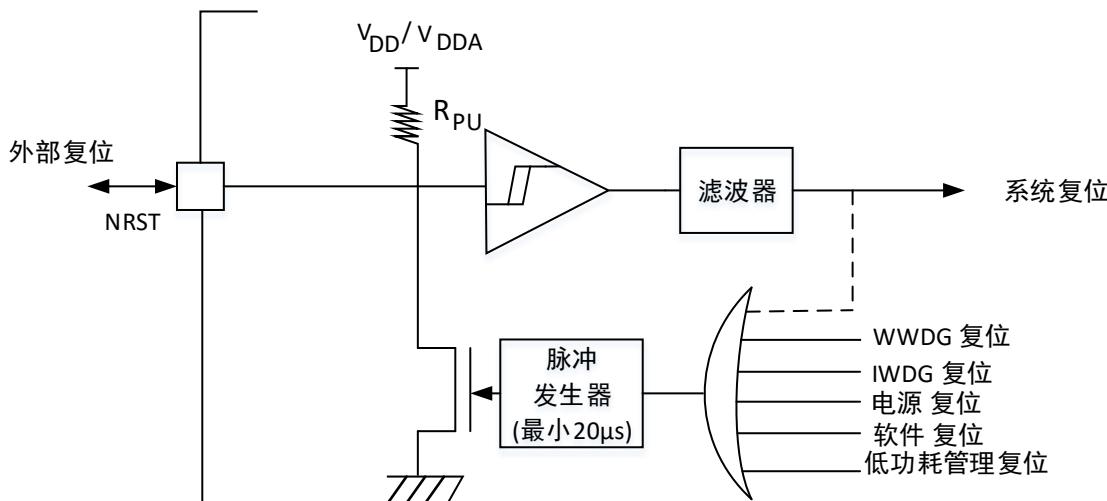
1. 上电/掉电复位（POR/PDR 复位）
2. 从待机模式中返回

电源复位将复位除了备份区域外的所有寄存器。（见图 2-1）

图中复位源将最终作用于 RESET 引脚，并在复位过程中保持低电平。复位入口矢量被固定在地址 0x0000\_0004。更多细节，参阅表 8-1：AT32F421 产品的向量表。

芯片内部的复位信号会在 NRST 引脚上输出，脉冲发生器保证每一个（外部或内部）复位源都能有至少 20μs 的脉冲延时；当 NRST 引脚被拉低产生外部复位时，它将产生复位脉冲。

图表 3-1 复位电路



### 3.1.3 备份域复位

备份区域拥有两个专门的复位，它们只影响备份区域（见图 2-1）。当以下事件中之一发生时，产生备份区域复位。

1. 软件复位，备份区域复位可由设置备份域控制寄存器（RCC\_BDC）（见 3.3.9 节）中的 BDRST 位产生。
2. 在 VDD 掉电的前提下，VDD 上电将引发备份区域复位。

## 3.2 时钟

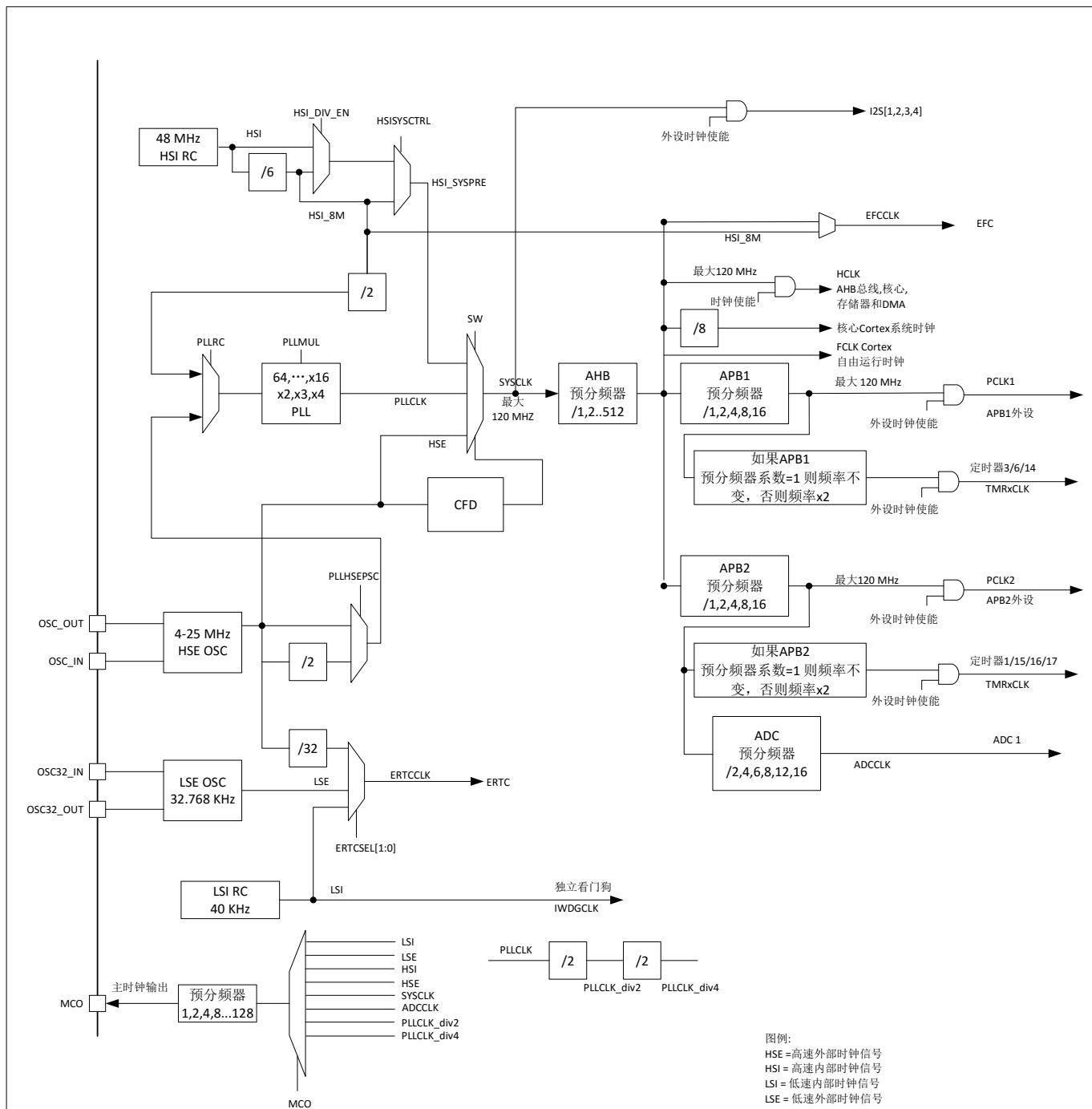
三种不同的时钟源可被用来驱动系统时钟（SYSCLK）：

- HSI 振荡器时钟
- HSE 振荡器时钟
- PLL 时钟

这些设备有以下 2 种二级时钟源：

- 40kHz 低速内部 RC，可以用于驱动独立看门狗和通过程序选择驱动 ERTC。ERTC 用于从停机/待机模式下自动唤醒系统。
- 32.768kHz 低速外部晶体也可用来通过程序选择驱动 ERTC（ERTCCLK）。当不被使用时，任一个时钟源都可被独立地启动或关闭，由此优化系统功耗。

图表 3-2 时钟树



- 当 HSI 被用于作为 PLL 时钟的输入时，系统时钟能得到的最大频率是 120MHz。
- 对于内部和外部时钟源的特性，请参考相应产品数据手册中“电气特性”章节。

用户可通过多个预分频器配置 AHB、高速 APB（APB2）和低速 APB（APB1）域的频率。AHB 域的最大频率是 120MHz。APB1 和 APB2 域的最大允许频率是 120MHz。

RCC 通过 AHB 时钟（HCLK）8 分频后作为 Cortex™-M4 系统定时器（SysTick）的外部时钟。通过对 SysTick 控制与状态寄存器的设置，可选择上述时钟或 Cortex™-M4（HCLK）时钟作为 SysTick 时钟。ADC 时钟由高速 APB2 时钟经 2、4、6、8、12、16 分频后获得。

定时器时钟频率分配由硬件按以下 2 种情况自动设置：

- 如果相应的 APB 预分频系数是 1，定时器的时钟频率与所在 APB 总线频率一致。

2. 否则，定时器的时钟频率被设为与其相连的 APB 总线频率的 2 倍。

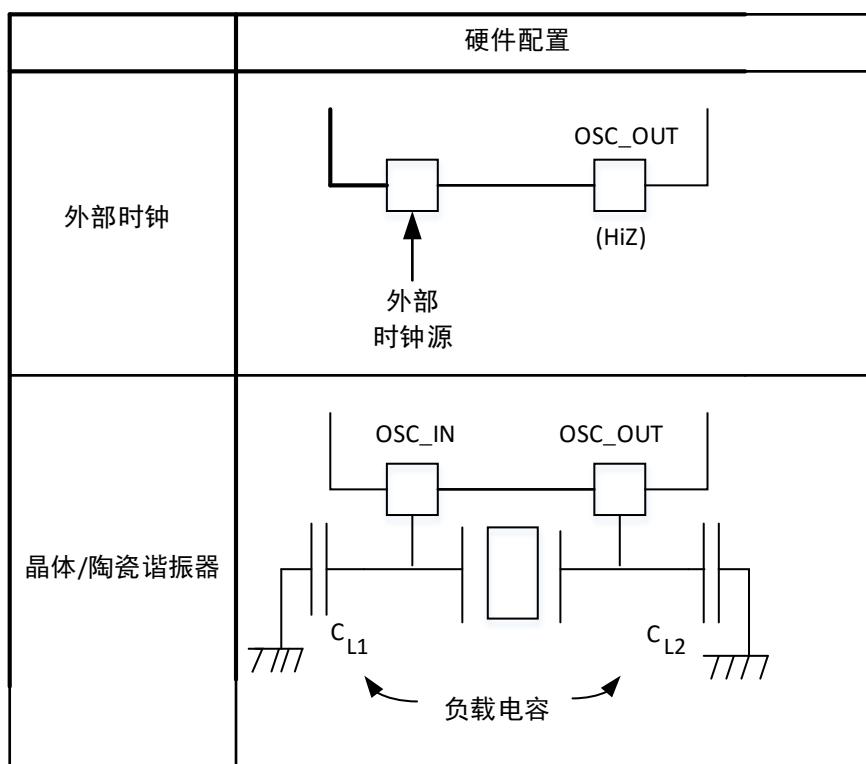
FCLK 是 Cortex™-M4 的自由运行时钟。详情见 ARM 的 Cortex™-M4 技术参考手册。

### 3.2.1 HSE时钟

高速外部时钟信号（HSE）由以下两种时钟源产生：

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

图表 3-3 HSE/LSE时钟源



#### 外部时钟源（HSE 旁路）

在这个模式里，必须提供外部时钟。它的频率最高可达 25MHz。用户可通过设置在时钟控制寄存器中的 HSEBYP 和 HSEEN 位来选择这一模式。外部时钟信号（50% 占空比的方波、正弦波或三角波）必须连到 OSC\_IN 引脚，同时保证 OSC\_OUT 引脚悬空。见图 3-3。

#### 外部晶体/陶瓷谐振器（HSE 晶体）

4~25MHz 外部振荡器可为系统提供更为精确的主时钟。相关的硬件配置可参考图 3-3，进一步信息可参考数据手册的电气特性部分。

在时钟控制寄存器 RCC\_CTRL 中的 HSESTBL 位用来指示高速外部振荡器是否稳定。在启动时，直到这一位被硬件置‘1’，时钟才被释放出来。如果在时钟中断寄存器 RCC\_CLKINT 中允许产生中断，将会产生相应中断。

HSE 晶体可以通过设置时钟控制寄存器里 RCC\_CTRL 中的 HSEEN 位被启动和关闭。

### 3.2.2 HSI时钟

HSI 时钟信号由内部 48 或 8 MHz 的 RC 振荡器产生，可直接作为系统时钟或在 2 分频后作为 PLL 输入。

HSI RC 振荡器能够在不需要任何外部器件的条件下提供系统时钟。它的启动时间比 HSE 晶体振荡器短。然而，即使在校准之后它的时钟频率精度仍较差。

校准制造工艺决定了不同芯片的 RC 振荡器频率会不同，这就是为什么每个芯片的 HSI 时钟频率在出厂前已经被 ATK 校准到 1% (25°C) 的原因。系统复位时，工厂校准值被装载到时钟控制寄存器的 HSICAL[7:0] 位。HSICAL[7:0] 可被软件进行再配置（需要把 HSICAL\_KEY 设成 0x5A），一旦经由软件配置过后，需重置系统才能重新装载工厂校准值。

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

时钟控制寄存器中的 HSISTBL 位用来指示 HSI RC 振荡器是否稳定。在时钟启动过程中，直到这一位被硬件置‘1’，HSI RC 输出时钟才被释放。HSI RC 可由时钟控制寄存器中的 HSIEN 位来启动和关闭。

如果 HSE 晶体振荡器失效，HSI 时钟会被作为备用时钟源。参考 [3.2.7 节时钟失效检测](#)。

### 3.2.3 PLL

内部 PLL 可以用来倍频 HSI RC 的输出时钟或 HSE 晶体输出时钟。参考 [图 3-2](#) 和时钟控制寄存器。

PLL 的设置（选择 HSI 振荡器除 2 或 HSE 振荡器或 HSE 振荡器除 2 为 PLL 的输入时钟，和选择倍频因子）必须在其被激活前完成。一旦 PLL 被激活，这些参数就不能被改动。

如果 PLL 中断在时钟中断寄存器里被允许，当 PLL 准备就绪时，可产生中断申请。

PLL 有两种配置使用方式，常规整数倍频配置模式和灵活配置模式：

#### 1) 常规整数倍频配置模式（默认模式）

根据 PLL 时钟计算公式：

$$\text{PLL 输出时钟} = \text{PLL 输入时钟} \times \text{PLL 倍频系数}$$

配置流程：

- 清除 PLLCFGGEN (RCC\_PLL[31])
- 设定当前 PLL 的输入时钟频率，具体参考 PLL\_FREF (RCC\_PLL[26:24])
- 设定当前 PLL 的倍频系数，具体参考 PLLMUL (RCC\_CFG[30:29], RCC\_CFG[21:18])
- 使能 PLL
- 等待 PLL 稳定工作

#### 2) 灵活配置模式

根据 PLL 时钟计算公式：

$$\text{PLL 输出时钟} = \text{PLL 输入时钟} \times \text{PLL 倍频系数} / (\text{PLL 预分频系数} \times \text{PLL 后分频系数})$$

$$500\text{MHz} \leq \text{PLL 输入时钟} \times \text{PLL 倍频系数} / \text{PLL 预分频系数} \leq 1000\text{MHz}$$

$$2\text{MHz} \leq \text{PLL 输入时钟} / \text{PLL 预分频系数} \leq 16\text{MHz}$$

配置流程：

- 计算出 PLL 倍频系数，PLL 预分频系数和 PLL 后分频系数后，对 RCC\_PLL 寄存器中的 PLL\_NS/PLL\_MS/PLL\_FR 对应位进行配置
- 使能 PLLCFGGEN (RCC\_PLL[31])
- 使能 PLL
- 等待 PLL 稳定工作

注：灵活配置模式，PLL\_FREF 和 PLLMUL 寄存器不起作用。

注：灵活配置模式，可以实现非整数倍频，

例如：当 PLL 输入时钟为 12.288 MHz 时，可以配置 PLL 输出频率 =  $12.288 \times 125 / (2 \times 8) = 96\text{MHz}$

例如：当 PLL 输入时钟为 5 MHz 时，可以配置 PLL 输出频率 =  $5 \times 108 / (5 \times 1) = 108\text{MHz}$

### 3.2.4 LSE 时钟

LSE 晶体是一个 32.768kHz 的低速外部晶体或陶瓷谐振器。它为实时时钟或者其他定时功能提供一个低

功耗且精确的时钟源。

**LSE** 晶体通过在备份域控制寄存器（RCC\_BDC）里的 LSEEN 位启动和关闭。在备份域控制寄存器（RCC\_BDC）里的 LSESTBL 指示 LSE 晶体振荡是否稳定。在启动阶段，直到这个位被硬件置‘1’后，LSE 时钟信号才被释放出来。如果在时钟中断寄存器里被允许，可产生中断申请。

外部时钟源（**LSE** 旁路）

在这个模式里必须提供一个 32.768kHz 频率的外部时钟源。你可以通过设置在备份域控制寄存器（RCC\_BDC）里的 LSEBYP\_S 和 LSEEN 位来选择这个模式。具有 50% 占空比的外部时钟信号（方波、正弦波或三角波）必须连到 OSC32\_IN 引脚，同时保证 OSC32\_OUT 引脚悬空，见图 3-3。

### 3.2.5 LSI时钟

**LSI RC** 担当一个低功耗时钟源的角色，它可以在停机和待机模式下保持运行，为独立看门狗和自动唤醒单元提供时钟。**LSI** 时钟频率大约 40kHz（在 30kHz 和 60kHz 之间）。进一步信息请参考数据手册中有关电气特性部分。

**LSI RC** 可以通过控制/状态寄存器（RCC\_CTRLSTS）里的 LSISTBL 位指示低速内部振荡器是否稳定。在启动阶段，直到这个位被硬件设置为‘1’后，此时钟才被释放。如果在时钟中断寄存器（RCC\_CLKINT）里被允许，将产生 **LSI** 中断申请。

**LSI** 校准可以通过校准内部低速振荡器 **LSI** 来补偿其频率偏移，从而获得精度可接受的 ERTC 时间基数，以及独立看门狗（**IWDG**）的超时时间（当这些外设以 **LSI** 为时钟源）。

### 3.2.6 系统时钟（**SYCLK**）选择

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

只有当目标时钟源准备就绪了（经过启动稳定阶段的延迟或 **PLL** 稳定），从一个时钟源到另一个时钟源的切换才会发生。在被选择时钟源没有就绪时，系统时钟的切换不会发生。直至目标时钟源就绪，才发生切换。

在时钟控制寄存器（RCC\_CTRL）里的状态位指示哪个时钟已经准备好了，哪个时钟目前被用作系统时钟。

### 3.2.7 时钟失效检测（**CFD**）

时钟失效检测可以通过软件被激活。一旦其被激活，时钟监测器将在 **HSE** 振荡器启动延迟后被使能，并在 **HSE** 时钟关闭后关闭。

如果 **HSE** 时钟发生故障，**HSE** 振荡器被自动关闭，时钟失效事件将被送到高级定时器（TMR1）的刹车输入端，并产生时钟安全中断 CFDI，允许软件完成营救操作。此 CFDI 中断连接到 Cortex™-M4 的 NMI 中断（不可屏蔽中断）。

**注意：** 一旦 **CFD** 被激活，并且 **HSE** 时钟出现故障，**CFD** 中断就产生，并且 **NMI** 也自动产生。**NMI** 将被不断执行，直到 **CFD** 中断挂起位被清除。因此，在 **NMI** 的处理程序中必须通过设置时钟中断寄存器（RCC\_CLKINT）里的 CFDFC 位来清除 **CFD** 中断。

如果 **HSE** 振荡器被直接或间接地作为系统时钟，（间接的意思是：它被作为 **PLL** 输入时钟，并且 **PLL** 时钟被作为系统时钟），时钟故障将导致系统时钟自动切换到 **HSI** 振荡器，同时外部 **HSE** 振荡器被关闭。在时钟失效时，如果 **HSE** 振荡器时钟（被分频或未被分频）是用作系统时钟的 **PLL** 的输入时钟，**PLL** 也将被关闭。

### 3.2.8 ERTC时钟

通过设置备份域控制寄存器（RCC\_BDC）里的 ERTCSEL[1:0]位，ERTCCLK 时钟源可以选择 **HSE/32**、**LSE** 或 **LSI**。除非备份域复位，此选择不能被改变。

在 STANDBY 模式下，**HSE** 时钟不能作为 ERTC 的时钟源，其余情况下，只要 VDD 正常供电，**HSE/32**、**LSE** 或 **LSI** 均可作为 ERTC 的时钟源。

### 3.2.9 看门狗时钟

如果独立看门狗已经由硬件选项或软件启动，LSI 振荡器将被强制在打开状态，并且不能被关闭。在 LSI 振荡器稳定后，时钟供应给 IWDG。

### 3.2.10 时钟输出

微控制器允许输出时钟信号到外部 CLKOUT 引脚。相应的 GPIO 端口寄存器必须被配置为相应功能。以下 8 个时钟信号可被选作 CLKOUT 时钟：

- ADC CLK
- SYSCLK
- LSI
- LSE
- HSI
- HSE
- 2 分频后的 PLL 时钟
- 4 分频后的 PLL 时钟

所有这些时钟输出都可以配置成为 1/2/4/8/16/64/128/256/512 分频，由 RCC\_MISC 的 MCOPRE[3:0]配置寄存器控制。

## 3.3 RCC 寄存器描述

下表列出了 RCC 寄存器的映像和复位值。

表格 3-1 RCC 寄存器的映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
000h	RCC_CTL	保留					PLLSTBL		PLLLEN		保留				CFDEN		HSICAL[7: 0]				HSITWK[5: 0]				HSISTBL				HSIEN																		
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															
004h	RCC_CFGR	保留		PLLMUL[5: 4]		ADCPSC[2]		保留		CLKOUT[2: 0]		保留		PLLMUL[3: 0]		PLLHSEPSC		PLLRC		ADCPSC[1: 0]		APB2_PSC[2: 0]		APB1_PSC[2: 0]		AHBPSC[3: 0]		SYSCLKSTS[1: 0]		SYSCLKSEL[1: 0]		HSISTBL		HSIEN													
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															
008h	RCC_CLKINT	保留						CFDFC		保留		PLLSTBLFC		HSESTBLFC		HSISTBLFC		LSESTBLFC		LSESTBLFC		保留		PLLSTBLIE		HSESTBLIE		HSISTBLIE		LSESTBLIE		CFDF		保留		PLLSTBLF		HSESTBLF		HSISTBLF		LSESTBLF		SYSCLKSEL[1: 0]			
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														
00Ch	RCC_APB2RSTR	保留								TMR17RST		TMR16RST		TMR15RST		USART1RST		保留		SP11RST		TMR1RST		保留		ADC1RST		保留		EXTIRST		SYSCFG_RST		保留		0		0		0		0		0		0	
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

010h	RCC_AP B1RSTR	保留	0	PWRRST	保留	I2C2RST I2C1RST	保留							
	复位值													
014h	RCC_AH BEN	保留	0	保留	保留	GPIOOPEN 保留	保留							
	复位值													
018h	RCC_AP B2EN	保留	0	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													
01Ch	RCC_AP B1EN	保留	0	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													
020h	RCC_BD C	保留	0	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													
024h	RCC_CTRLST S	保留	0	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													
028h	RCC_AH BRSTR	保留	0	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													
02Ch	RCC_PL L	保留	0	PLLREF[2:0]	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留
	复位值													

030h	RCC_MI SC	MCOPRE[3:0]				保留	HSI_DIV_EN	保留	CLKEFC_SRC	保留	CLKOUT[3]	保留				HSICAL_KEY[7:0]							
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
054h	RCC_MI SC2	保留												HSI_SYS_CTRL	保留	AUTO_STEP_EN	保留						
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 3.3.1 时钟控制寄存器 (RCC\_CTRL)

偏移地址: 0x00

复位值: 0x0000 XX83, X 代表未定义

访问: 无等待状态, 字, 半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		保留			PLL_STBL	PLLEN	保留			CFDEN	HSE_BYPS	HSE_STBL	HSE_EN		
		res			r	rw	res			rw	rw	r	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		HSICAL[7: 0]						HSITWK[7: 2]						HSI_STBL	HSI_EN
		rw						rw						r	rw

位 31: 26	保留, 始终读为 0。
位 25	PLLSTBL: PLL 时钟就绪标志 (PLL clock ready flag) PLL 锁定后由硬件置'1'。 0: PLL 未锁定; 1: PLL 锁定。
位 24	PLLEN: PLL 使能 (PLL enable) 由软件置'1'或清零。当进入待机和停机模式时, 该位由硬件清零。当 PLL 时钟被用作或被选择将要作为系统时钟时, 该位不能被清零。 0: PLL 关闭; 1: PLL 使能。
位 23: 20	保留, 始终读为 0。
位 19	CFDEN: 时钟失效检测使能 (Clock Failure Detection enable) 由软件置'1'或清零以使能时钟监测器。 0: 时钟监测器关闭; 1: 如果外部 4-25MHz 振荡器就绪, 时钟监测器开启。
位 18	HSEBYPSS: 外部高速时钟旁路 (External high-speed clock bypass) 在调试模式下由软件置'1'或清零来旁路外部晶体振荡器。只有在外部 4-25MHz 振荡器关闭的情况下, 才能写入该位。 0: 外部 4-25MHz 振荡器没有旁路; 1: 外部 4-25MHz 外部晶体振荡器被旁路。

位 17	HSESTBL: 外部高速时钟就绪标志 (External high-speed clock ready flag) 由硬件置'1'来指示外部 4-25MHz 振荡器已经稳定。在 HSEEN 位清零后，该位需要 6 个外部 4-25MHz 振荡器周期清零。 0: 外部 4-25MHz 振荡器没有就绪； 1: 外部 4-25MHz 振荡器就绪。
位 16	HSEEN: 外部高速时钟使能 (External high-speed clock enable) 由软件置'1'或清零。 当进入待机和停机模式时，该位由硬件清零，关闭 4-25MHz 外部振荡器。当外部 4-25MHz 振荡器被用作或被选择将要作为系统时钟时，该位不能被清零。 0: HSE 振荡器关闭； 1: HSE 振荡器开启。
位 15: 8	HSICAL[7: 0]: 内部高速时钟校准 (Internal high-speed clock calibration) 在系统启动时，这些位被自动初始化。 此字段只有在 HSICAL_KEY[7:0] 为 0x5A 的时候可被写入。
位 7: 2	HSITWK[5: 0]: 内部高速时钟调整 (Internal high-speed clock trimming) 由软件写入来调整内部高速时钟，它们被叠加在 HSICAL[7: 0] 数值上。这些位在 HSICAL[7: 0] 的基础上，让用户可以输入一个调整数值，根据电压和温度的变化调整内部 HSI RC 振荡器的频率。 默认数值为 32，可以把 HSI 调整到 48MHz±1%；每步 HSICAL 的变化调整约 40kHz。
位 1	HSISTBL: 内部高速时钟就绪标志 (Internal high-speed clock ready flag) 由硬件置'1'来指示内部 48MHz 振荡器已经稳定。在 HSIEN 位清零后，该位需要 6 个内部 48MHz 振荡器周期清零。 0: 内部 48MHz 振荡器没有就绪； 1: 内部 48MHz 振荡器就绪。
位 0	HSIEN: 内部高速时钟使能 (Internal high-speed clock enable) 由软件置'1'。 当从待机和停机模式返回或用作系统时钟的外部 4-25MHz 振荡器发生故障时，该位由硬件置'1'来启动内部 48MHz 的 RC 振荡器。当内部 48MHz 振荡器被直接或间接地用作或被选择将要作为系统时钟时，该位不能被清零。 0: 内部 48MHz 振荡器关闭； 1: 内部 48MHz 振荡器开启。

### 3.3.2 时钟配置寄存器 (RCC\_CFG)

偏移地址: 0x04

复位值: 0x0000 0000

访问: 0 到 2 个等待周期，字，半字和字节访问，只有当访问发生在时钟切换时，才会插入 1 或 2 个等待周期。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	PLLMUL [5: 4]	ADC PSC[2]	保留	CLKOUT[2: 0]	保留	保留	保留	保留	保留	保留	保留	保留	保留	PLLHSE PSC	PLL RC
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCPSC[1: 0]	APB2PSC[2: 0]	APB1PSC[2: 0]	AHBPSC[3: 0]	SYSCLKSTS [1: 0]	SYSCLKSEL [1: 0]										
rw	rw	rw	rw	r	r										
位 31	保留，始终读为 0。														

位 26: 24	<p><b>CLKOUT[3:0]:</b> 微控制器时钟输出 (Microcontroller clock output)  <b>CLKOUT[3] 在 RCC_MISC 寄存器位 16</b>  由软件置'1'或清零。  0000: 没有时钟输出;  0001: 没有时钟输出;  0010: 内部低速 RC 振荡器时钟 (LSI) 输出;  0011: 外部低速振荡器时钟 (LSE) 输出;  0100: 系统时钟 (SYSCLK) 输出;  0101: 内部 RC 振荡器时钟 (HSI) 输出;  0110: 外部振荡器时钟 (HSE) 输出;  0111: PLL 时钟 2 分频后输出;  1100: PLL 时钟 4 分频后输出;  1101: 没有时钟输出;  1110: ADC 时钟输出。  注意: - 该时钟输出在启动和切换 CLKOUT 时钟源时可能会被截断。  - 在系统时钟作为输出至 CLKOUT 引脚时, 请保证输出时钟频率不超过 50MHz (I/O 口最高频率)。 </p>
位 27 位 23: 22	保留, 始终读为 0。
位 30: 29 位 21: 18	<p><b>PLLMUL[5: 0]:</b> PLL 倍频系数 (PLL multiplication factor) { 位 30: 29, 位 21: 18}  仅在 PLLCFG=0 时配置有效, 由软件设置来确定 PLL 倍频系数。只有在 PLL 关闭的情况下才可被写入。  000000: PLL 2 倍频输出      001111: PLL 16 倍频输出  010000: PLL 17 倍频输出      011111: PLL 32 倍频输出  100000: PLL 33 倍频输出      101111: PLL 48 倍频输出  110000: PLL 49 倍频输出      111111: PLL 64 倍频输出 </p>
位 17	<p><b>PLLHSEPSC:</b> HSE 分频器作为 PLL 输入 (HSE divider for PLL entry)  由软件置'1'或清'0'来分频 HSE 后作为 PLL 输入时钟。只能在关闭 PLL 时才能写入此位。  0: HSE 不分频  1: HSE 2 分频 </p>
位 16	<p><b>PLLRC:</b> PLL 输入时钟源 (PLL entry clock source)  由软件置'1'或清'0'来选择 PLL 输入时钟源。只能在关闭 PLL 时才能写入此位。  0: HSI 振荡器时钟经 2 分频后作为 PLL 输入时钟  1: HSE 时钟作为 PLL 输入时钟。 </p>
位 28 位 15: 14	<p><b>ADCPSC[2: 0]:</b> ADC 预分频 (ADC prescaler)  <b>ADCPSC[2],</b> 由位 28 来做设定  由软件置'1'或清'0'来确定 ADC 时钟频率  000: PCLK2 2 分频后作为 ADC 时钟  001: PCLK2 4 分频后作为 ADC 时钟  010: PCLK2 6 分频后作为 ADC 时钟  011: PCLK2 8 分频后作为 ADC 时钟  100: PCLK2 2 分频后作为 ADC 时钟  101: PCLK2 12 分频后作为 ADC 时钟  110: PCLK2 8 分频后作为 ADC 时钟  111: PCLK2 16 分频后作为 ADC 时钟 </p>
位 13: 11	<p><b>APB2PSC[2: 0]:</b> APB 预分频 (APB2) (APB prescaler (APB2))  由软件置'1'或清'0'来控制 APB2 时钟 (PCLK2) 的预分频系数。  0xx: HCLK 不分频  100: HCLK 2 分频  101: HCLK 4 分频  110: HCLK 8 分频  111: HCLK 16 分频 </p>

位 10: 8	<b>APB1PSC[2: 0]:</b> APB 预分频 (APB1) (APB prescaler (APB1)) 由软件置'1'或清'0'来控制 APB1 时钟 (PCLK1) 的预分频系数。 0xx: HCLK 不分频 100: HCLK 2 分频 101: HCLK 4 分频 110: HCLK 8 分频 111: HCLK 16 分频
位 7: 4	<b>AHBPSC[3: 0]:</b> AHB 预分频 (AHB Prescaler) 由软件置'1'或清'0'来控制 AHB 时钟的预分频系数。 0xxx: SYSCLK 不分频 1000: SYSCLK 2 分频 1100: SYSCLK 64 分频 1001: SYSCLK 4 分频 1101: SYSCLK 128 分频 1010: SYSCLK 8 分频 1110: SYSCLK 256 分频 1011: SYSCLK 16 分频 1111: SYSCLK 512 分频 注意: 当 AHB 时钟的预分频系数大于 1 时, 必须开启预取缓冲器。
位 3: 2	<b>SYCLKSTS[1: 0]:</b> 系统时钟切换状态 (System clock switch status) 由硬件置'1'或清'0'来指示哪一个时钟源被作为系统时钟。 00: HSI 作为系统时钟; 01: HSE 作为系统时钟; 10: PLL 输出作为系统时钟; 11: 不可用。
位 1: 0	<b>SYCLKSEL[1: 0]:</b> 系统时钟切换 (System clock switch) 由软件置'1'或清'0'来选择系统时钟源。在从停机或待机模式中返回时或直接或间接作为系统时钟的 HSE 出现故障时, 由硬件强制选择 HSI 作为系统时钟 (如果时钟安全系统已经启动) 00: HSI 作为系统时钟; 01: HSE 作为系统时钟; 10: PLL 输出作为系统时钟; 11: 不可用。

### 3.3.3 时钟中断寄存器 (RCC\_CLKINT)

偏移地址: 0x08

复位值: 0x0000 0000

访问: 无等待周期, 字, 半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留				CFDFC	保留	PLLSTB LFC	HSEST BLFC	HSIST BLFC	LSES TBLFC	LSIST BLFC					
res				w	res	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	PLLS TBLIE	HSES TBLIE	HSIS TBLIE	LSES TBLIE	LSIST BLIE	CFDF	保留	PLLS TBLF	HSES TBLF	HSIS TBLF	LSES TBLF	LSIS TBLF			
res	rw	rw	rw	rw	rw	r	res	r	r	r	r	r	r	r	r

位 31: 24	保留, 始终读为 0。
位 23	CFDFC: 清除时钟失效中断 (Clock failure detection interrupt clear) 由软件置'1'来清除 CFDF 安全系统中断标志位 CFDF。 0: 无作用; 1: 清除 CFDF 安全系统中断标志位。
位 22: 21	保留, 始终读为 0。

位 20	<b>PLLSTBLFC:</b> 清除 PLL 就绪中断 (PLL ready interrupt clear) 由软件置'1'来清除 PLL 就绪中断标志位 PLLSTBLF。 0: 无作用; 1: 清除 PLL 就绪中断标志位 PLLSTBLF。
位 19	<b>HSESTBLFC:</b> 清除 HSE 就绪中断 (HSE ready interrupt clear) 由软件置'1'来清除 HSE 就绪中断标志位 HSESTBLF。 0: 无作用; 1: 清除 HSE 就绪中断标志位 HSESTBLF。
位 18	<b>HSISTBLFC:</b> 清除 HSI 就绪中断 (HSI ready interrupt clear) 由软件置'1'来清除 HSI 就绪中断标志位 HSISTBLF。 0: 无作用; 1: 清除 HSI 就绪中断标志位 HSISTBLF。
位 17	<b>LSESTBLFC:</b> 清除 LSE 就绪中断 (LSE ready interrupt clear) 由软件置'1'来清除 LSE 就绪中断标志位 LSESTBLF。 0: 无作用; 1: 清除 LSE 就绪中断标志位 LSESTBLF。
位 16	<b>LSISTBLFC:</b> 清除 LSI 就绪中断 (LSI ready interrupt clear) 由软件置'1'来清除 LSI 就绪中断标志位 LSISTBLF。 0: 无作用; 1: 清除 LSI 就绪中断标志位 LSISTBLF。
位 15: 13	保留, 始终读为 0。
位 12	<b>PLLSTBLIE:</b> PLL 就绪中断使能 (PLL ready interrupt enable) 由软件置'1'或清'0'来使能或关闭 PLL 就绪中断。 0: PLL 就绪中断关闭; 1: PLL 就绪中断使能。
位 11	<b>HSESTBLIE:</b> HSE 就绪中断使能 (HSE ready interrupt enable) 由软件置'1'或清'0'来使能或关闭外部 4-25MHz 振荡器就绪中断。 0: HSE 就绪中断关闭; 1: HSE 就绪中断使能。
位 10	<b>HSISTBLIE:</b> HSI 就绪中断使能 (HSI ready interrupt enable) 由软件置'1'或清'0'来使能或关闭内部 48MHz RC 振荡器就绪中断。 0: HSI 就绪中断关闭; 1: HSI 就绪中断使能。
位 9	<b>LSESTBLIE:</b> LSE 就绪中断使能 (LSE ready interrupt enable) 由软件置'1'或清'0'来使能或关闭外部 32kHz RC 振荡器就绪中断。 0: LSE 就绪中断关闭; 1: LSE 就绪中断使能。
位 8	<b>LSISTBLIE:</b> LSI 就绪中断使能 (LSI ready interrupt enable) 由软件置'1'或清'0'来使能或关闭内部 40kHz RC 振荡器就绪中断。 0: LSI 就绪中断关闭; 1: LSI 就绪中断使能。
位 7	<b>CFDF:</b> 时钟失效中断标志 (Clock Failure Detection interrupt flag) 在外部 4-25MHz 振荡器时钟出现故障时, 由硬件置'1'。由软件通过置'1' CFDFC 位来清除。 0: 无 HSE 时钟失效产生的安全系统中断; 1: HSE 时钟失效导致了时钟安全系统中断。
位 6: 5	保留, 始终读为 0。

位 4	<b>PLLSTBLF:</b> PLL 就绪中断标志 (PLL ready interrupt flag) 在 PLL 就绪且 <b>PLLSTBLIE</b> 位被置'1'时, 由硬件置'1'。由软件通过置'1' <b>PLLSTBLFC</b> 位来清除。 0: 无 PLL 上锁产生的时钟就绪中断; 1: PLL 上锁导致时钟就绪中断。
位 3	<b>HSESTBLF:</b> HSE 就绪中断标志 (HSE ready interrupt flag) 在外部低速时钟就绪且 <b>HSESTBLIE</b> 位被置'1'时, 由硬件置'1'。由软件通过置'1' <b>HSESTBLFC</b> 位来清除。 0: 无外部 4-25MHz 振荡器产生的时钟就绪中断; 1: 外部 4-25MHz 振荡器导致时钟就绪中断。
位 2	<b>HSISTBLF:</b> HSI 就绪中断标志 (HSI ready interrupt flag) 在内部高速时钟就绪且 <b>HSISTBLIE</b> 位被置'1'时, 由硬件置'1'。由软件通过置'1' <b>HSISTBLFC</b> 位来清除。 0: 无内部 48MHz RC 振荡器产生的时钟就绪中断; 1: 内部 48MHz RC 振荡器导致时钟就绪中断。
位 1	<b>LSESTBLF:</b> LSE 就绪中断标志 (LSE ready interrupt flag) 在外部低速时钟就绪且 <b>LSESTBLIE</b> 位被置'1'时, 由硬件置'1'。 由软件通过置'1' <b>LSESTBLFC</b> 位来清除。 0: 无外部 32kHz 振荡器产生的时钟就绪中断; 1: 外部 32kHz 振荡器导致时钟就绪中断。
位 0	<b>LSISTBLF:</b> LSI 就绪中断标志 (LSI ready interrupt flag) 在内部低速时钟就绪且 <b>LSISTBLIE</b> 位被置'1'时, 由硬件置'1'。由软件通过置'1' <b>LSISTBLFC</b> 位来清除。 0: 无内部 40kHz RC 振荡器产生的时钟就绪中断; 1: 内部 40kHz RC 振荡器导致时钟就绪中断。

### 3.3.4 APB2外设复位寄存器 (RCC\_APB2RST)

偏移地址: 0x0C

复位值: 0x0000 0000

访问: 无等待周期, 字, 半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16								
保留 BG														TMR17 RST	TMR16 RST	TMR15 RST							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
保留	USAR T1RST	保留	SPI1 RST	TMR1 RST	保留	ADC RST	保留						EXTI RST	SYSCFG COMPR ST									
res	rw	res	rw	rw	res	rw	res						rw	rw									
位 31: 19		保留, 始终读为 0。																					
位 18		<b>TMR17RST:</b> TMR17 定时器复位 (TMR17 timer reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 TMR17 定时器。																					
位 17		<b>TMR16RST:</b> TMR16 定时器复位 (TMR16 timer reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 TMR16 定时器。																					

位 16	<b>TMR15RST:</b> TMR15 定时器复位 (TMR15 timer reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 TMR15 定时器。
位 15	保留, 始终读为 0。
位 14	<b>USART1RST:</b> USART1 复位 (USART1 reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 USART1。
位 13	保留, 始终读为 0。
位 12	<b>SPI1RST:</b> SPI1 复位 (SPI 1 reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 SPI1。
位 11	<b>TMR1RST:</b> TMR1 定时器复位 (TMR1 timer reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 TMR 1 定时器。
位 10	保留, 始终读为 0。
位 9	<b>ADCRST:</b> ADC 接口复位 (ADC interface reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位 ADC 接口。
位 8:2	保留, 始终读为 0。
位 1	<b>EXTIRST:</b> EXTI 复位 (EXTI reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位。
位 0	<b>SYSCFGCOMPRST:</b> SYSCFG 和 COMP 复位 (SYSCFGCOMP reset) 由软件置'1'或清'0' 0: 无作用; 1: 复位。

### 3.3.5 APB1外设复位寄存器 (RCC\_APB1RST)

偏移地址: 0x10

复位值: 0x0000 0000

访问: 无等待周期, 字, 半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	PWR RST	保留	保留	I2C2 RST	I2C1 RST	保留	USART2 RST	保留	res	rw	res	rw	res	rw	rw
res	rw	res	rw	res	rw	res	rw	res	rw	rw	res	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	SPI2 RST	保留	WWDG RST	保留	TMR14 RST	保留	保留	保留	TMR6 RST	保留	TMR3 RST	保留	res	rw	res
rw	rw	res	rw	res	rw	rw	res	res	rw	res	rw	rw	rw	rw	res

位 31:29	保留，始终读为 0。
位 28	<b>PWRRST:</b> 电源接口复位 (Power interface reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位电源接口。
位 27:23	保留，始终读为 0。
位 22	<b>I2C2RST:</b> I2C 2 复位 (I2C 2 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 I2C 2。
位 21	<b>I2C1RST:</b> I2C 1 复位 (I2C 1 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 I2C 1。
位 20:18	保留，始终读为 0.
位 17	<b>USART2RST:</b> USART2 复位 (USART 2 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 USART2。
位 16:15	保留，始终读为 0。
位 14	<b>SPI2RST:</b> SPI2 复位 (SPI 2 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 SPI2。
位 13:12	保留，始终读为 0。
位 11	<b>WWDGRST:</b> 窗口看门狗复位 (WCNTWindow watchdog reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位窗口看门狗。
位 10:9	保留，始终读为 0。
位 8	<b>TMR14RST:</b> 定时器 14 复位 (Timer 14 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 TMR 14 定时器。
位 7:5	保留，始终读为 0。
位 4	<b>TMR6RST:</b> 定时器 6 复位 (Timer 6 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 TMR 6 定时器。
位 3:2	保留，始终读为 0。
位 1	<b>TMR3RST:</b> 定时器 3 复位 (Timer 3 reset) 由软件置'1'或清'0' 0: 无作用； 1: 复位 TMR 3 定时器。
位 0	保留，始终读为 0。

### 3.3.6 AHB外设时钟使能寄存器（RCC\_AHBEN）

偏移地址: 0x14

复位值: 0x0000 0014

访问: 无等待周期, 字、半字和字节访问

注意: 当外设时钟没有启用时, 软件不能读出外设寄存器的数值, 返回的数值始终是 0x0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								GPIO FEN	保留		GPIO CEN	GPIO BEN	GPIO AEN	保留	
res								rw	res		rw	rw	rw	res	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CCE	保留	FLASH EN	保留	SRAM EN	保留	DMA EN	
res								rw	res	rw	res	rw	res	rw	

位 31:23	保留, 始终读为 0。
位 22	<b>GPIOFEN:</b> IO 端口 E 时钟使能 (I/O port F clock enable) 由软件置'1'或清'0' 0: IO 端口 F 时钟关闭; 1: IO 端口 F 时钟开启。
位 21:20	保留, 始终读为 0。
位 19	<b>GPIOCEN:</b> IO 端口 C 时钟使能 (I/O port C clock enable) 由软件置'1'或清'0' 0: IO 端口 C 时钟关闭; 1: IO 端口 C 时钟开启。
位 18	<b>GPIOBEN:</b> IO 端口 B 时钟使能 (I/O port B clock enable) 由软件置'1'或清'0' 0: IO 端口 B 时钟关闭; 1: IO 端口 B 时钟开启。
位 17	<b>GPIOAEN:</b> IO 端口 A 时钟使能 (I/O port A clock enable) 由软件置'1'或清'0' 0: IO 端口 A 时钟关闭; 1: IO 端口 A 时钟开启。
位 16:7	保留, 始终读为 0。
位 6	<b>CCE:</b> CRC 时钟使能 (CRC clock enable) 由软件置'1'或清'0'。 0: CRC 时钟关闭; 1: CRC 时钟开启。
位 5	保留, 始终读为 0。
位 4	<b>FLASHEN:</b> 内部闪存接口电路时钟使能 (EFC clock enable) 由软件置'1'或清'0'。 0: EFC 时钟关闭; 1: EFC 时钟开启。
位 3	保留, 始终读为 0。
位 2	<b>SRAMEN:</b> SRAM 时钟使能 (SRAM interface clock enable) 由软件置'1'或清'0'来开启或关闭睡眠模式时 SRAM 时钟。 0: 睡眠模式时 SRAM 时钟关闭; 1: 睡眠模式时 SRAM 时钟开启。

位 1	保留，始终读为 0。
位 0	<b>DMAEN:</b> DMA 时钟使能 (DMA clock enable) 由软件置'1'或清'0'。 0: DMA 时钟关闭； 1: DMA 时钟开启。

### 3.3.7 APB2外设时钟使能寄存器 (RCC\_APB2EN)

偏移地址: 0x18

复位值: 0x0000 0000

访问: 字, 半字和字节访问

通常无访问等待周期。

但在 APB2 总线上的外设被访问时, 将插入等待状态直到 APB2 的外设访问结束。

注意: 当外设时钟没有启用时, 软件不能读出外设寄存器的数值, 返回的数值始终是 0x0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16						
保留														TMR17EN	TMR16EN	TMR15EN					
res														rw	rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
保留	USART1 EN	保留	SPI1 EN	TMR1 EN	保留	ADC EN	保留														SYSCFG COMP EN
res	rw	rw	rw	rw	rw	rw	res														rw

位 31: 19	保留, 始终读为 0。
位 18	<b>TMR17EN:</b> TMR17 定时器时钟使能 (Timer 17 clock enable) 由软件置'1'或清'0' 0: TMR17 定时器时钟关闭； 1: TMR17 定时器时钟开启。
位 17	<b>TMR16EN:</b> TMR16 定时器时钟使能 (Timer 16 clock enable) 由软件置'1'或清'0' 0: TMR16 定时器时钟关闭； 1: TMR16 定时器时钟开启。
位 16	<b>TMR15EN:</b> TMR15 定时器时钟使能 (Timer 15 clock enable) 由软件置'1'或清'0' 0: TMR15 定时器时钟关闭； 1: TMR15 定时器时钟开启。
位 15	保留, 始终读为 0。
位 14	<b>USART1EN:</b> USART1 时钟使能 (USART1 clock enable) 由软件置'1'或清'0' 0: USART1 时钟关闭； 1: USART1 时钟开启。
位 13	保留, 始终读为 0。

位 12	<b>SPI1EN:</b> SPI1 时钟使能 (SPI 1 clock enable) 由软件置'1'或清'0' 0: SPI1 时钟关闭; 1: SPI1 时钟开启。
位 11	<b>TMR1EN:</b> TMR1 定时器时钟使能 (Timer 1 clock enable) 由软件置'1'或清'0' 0: TMR1 定时器时钟关闭; 1: TMR1 定时器时钟开启。
位 10	保留, 始终读为 0。
位 9	<b>ADCEN:</b> ADC 接口时钟使能 (ADC interface clock enable) 由软件置'1'或清'0' 0: ADC 接口时钟关闭; 1: ADC 接口时钟开启。
位 8:1	保留, 始终读为 0。
位 0	<b>SYSCFGCOMPEN:</b> SYSCFG 和 COMP 时钟使能 由软件置'1'或清'0' 0: SYSCFG 和 COMP 时钟关闭; 1: SYSCFG 和 COMP 时钟开启;

### 3.3.8 APB1外设时钟使能寄存器 (RCC\_APB1EN)

偏移地址: 0x1C

复位值: 0x0000 0000

访问: 字、半字和字节访问

通常无访问等待周期。但在 APB1 总线上的外设被访问时, 将插入等待状态直到 APB1 外设访问结束。

注意: 当外设时钟没有启用时, 软件不能读出外设寄存器的数值, 返回的数值始终是 0x0。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	PWR EN	保留		I2C2 EN	I2C1 EN	保留	USART2 EN	保留							
rw	rew	rw	rw	res	rw	rw	res	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	SPI2 EN	保留	WWDG EN	保留	TMR14 EN	保留	TMR6 EN	保留	TMR3 EN	保留					
rw	rw	res	rw	res	rw	res	rw	res	rw	res	rw	rw	rw	rw	res

位 31:29	保留, 始终读为 0。
位 28	<b>PWREN:</b> 电源接口时钟使能 (Power interface clock enable) 由软件置'1'或清'0' 0: 电源接口时钟关闭; 1: 电源接口时钟开启。
位 27:23	保留, 始终读为 0。
位 22	<b>I2C2EN:</b> I2C 2 时钟使能 (I2C 2 clock enable) 由软件置'1'或清'0' 0: I2C 2 时钟关闭; 1: I2C 2 时钟开启。
位 21	<b>I2C1EN:</b> I2C 1 时钟使能 (I2C 1 clock enable) 由软件置'1'或清'0' 0: I2C 1 时钟关闭; 1: I2C 1 时钟开启。

位 20:18	保留, 始终读为 0。
位 17	<b>USART2EN:</b> USART2 时钟使能 (USART 2 clock enable) 由软件置'1'或清'0' 0: USART2 时钟关闭; 1: USART2 时钟开启。
位 16:15	保留, 始终读为 0。
位 14	<b>SPI2EN:</b> SPI 2 时钟使能 (SPI 2 clock enable) 由软件置'1'或清'0' 0: SPI 2 时钟关闭; 1: SPI 2 时钟开启。
位 13: 12	保留, 始终读为 0。
位 11	<b>WWDGEN:</b> 窗口看门狗时钟使能 (Window watchdog clock enable) 由软件置'1'或清'0' 0: 窗口看门狗时钟关闭; 1: 窗口看门狗时钟开启。
位 10:9	保留, 始终读为 0。
位 8	<b>TMR14EN:</b> 定时器 14 时钟使能 (Timer 14 clock enable) 由软件置'1'或清'0' 0: 定时器 14 时钟关闭; 1: 定时器 14 时钟开启。
位 7:5	保留, 始终读为 0。
位 4	<b>TMR6EN:</b> 定时器 6 时钟使能 (Timer 6 clock enable) 由软件置'1'或清'0' 0: 定时器 6 时钟关闭; 1: 定时器 6 时钟开启。
位 3:2	保留, 始终读为 0.
位 1	<b>TMR3EN:</b> 定时器 3 时钟使能 (Timer 3 clock enable) 由软件置'1'或清'0' 0: 定时器 3 时钟关闭; 1: 定时器 3 时钟开启。
位 0	保留, 始终读为 0.

### 3.3.9 备份域控制寄存器 (RCC\_BDC)

偏移地址: 0x20

复位值: 0x0000 0000, 只能由备份域复位有效复位

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

**注意:** 备份域控制寄存器中 (RCC\_BDC) LSEEN、LSEBYPs、ERTCSEL 和 ERTCEN 位处于备份域。因此, 这些位在复位后处于写保护状态, 只有在电源控制寄存器 (PWR\_CTRL) 中的 DBP 位置 '1' 后才能对这些位进行改动。进一步信息请参考 [3.1 节](#)。这些位只能由备份域复位清除 (见 [3.1.3 节](#))。任何内部或外部复位都不会影响这些位。

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

保留	BDRST
----	-------

		res		rw
15	14	13	12	11 10 9 8 7 6 5 4 3 2 1 0
ERTCEN	保留	ERTCSEL[1:0]	保留	LSE BYPS LSE STBL LSEEN
rw	res	rw	rw	rw r rw
位 31: 17	保留, 始终读为 0。			
位 16	<b>BDRST:</b> 备份域软件复位 (Backup domain software reset) 由软件置'1'或清'0' 0: 复位未激活; 1: 复位整个备份域。			
位 15	<b>ERTCEN:</b> ERTC 时钟使能 (ERTC clock enable) 由软件置'1'或清'0' 0: ERTC 时钟关闭; 1: ERTC 时钟开启。			
位 14: 10	保留, 始终读为 0。			
位 9: 8	<b>ERTCSEL[1: 0]:</b> ERTC 时钟源选择 (ERTC clock source selection) 由软件设置来选择 ERTC 时钟源。一旦 ERTC 时钟源被选定, 直到下次后备域被复位, 它不能在被 改变。可通过设置 BDRST 位来清除。 00: 无时钟; 01: LSE 振荡器作为 ERTC 时钟; 10: LSI 振荡器作为 ERTC 时钟; 11: HSE 振荡器在 32 分频后作为 ERTC 时钟。			
位 7: 3	保留, 始终读为 0。			
位 2	<b>LSEBYP:</b> 外部低速时钟振荡器旁路 (External low-speed oscillator bypass) 在调试模式下由软件置'1'或清'0'来旁路 LSE。只有在外部 32kHz 振荡器关闭时, 才能写入该位 0: LSE 时钟未被旁路; 1: LSE 时钟被旁路。			
位 1	<b>LSESTBL:</b> 外部低速 LSE 就绪 (External low-speed oscillator ready) 由硬件置'1'或清'0'来指示是否外部 32kHz 振荡器就绪。在 LSEEN 被清零后, 该位需要 6 个外部 低速振荡器的周期才被清零。 0: 外部 32kHz 振荡器未就绪; 1: 外部 32kHz 振荡器就绪。			
位 0	<b>LSEEN:</b> 外部低速振荡器使能 (External low-speed oscillator enable) 由软件置'1'或清'0' 0: 外部 32kHz 振荡器关闭; 1: 外部 32kHz 振荡器开启。			

### 3.3.10 控制/状态寄存器 (RCC\_CTRLSTS)

偏移地址: 0x24

复位值: 0x0C00 0000, 除复位标志外由系统复位清除, 复位标志只能由电源复位清除。访问: 0 到 3 等待周期, 字、半字和字节访问; 当连续对该寄存器进行访问时, 将插入等待状态。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LP RSTF	WWDG RSTF	IWDG RSTF	SW RSTF	POR RSTF	PIN RSTF	保留	RST FC	保留							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

	保留	res	LSI STBL	LSI EN
位 31	<b>LPRSTF:</b> 低功耗复位标志 (Low-power reset flag) 在低功耗管理复位发生时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无低功耗管理复位发生; 1: 发生低功耗管理复位。 关于低功耗管理复位的详细信息, 请参考 3.1.1 节的"低功耗管理复位"。		r	rw
位 30	<b>WWDGRSTF:</b> 窗口看门狗复位标志 (Window watchdog reset flag) 在窗口看门狗复位发生时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无窗口看门狗复位发生; 1: 发生窗口看门狗复位。			
位 29	<b>IWDGRSTF:</b> 独立看门狗复位标志 (Independent watchdog reset flag) 在独立看门狗复位发生在 VDD 区域时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无独立看门狗复位发生; 1: 发生独立看门狗复位。			
位 28	<b>SWRSTF:</b> 软件复位标志 (Software reset flag) 在软件复位发生时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无软件复位发生; 1: 发生软件复位。			
位 27	<b>PORRSTF:</b> 上电/掉电复位标志 (POR/PDR reset flag) 在上电/掉电复位发生时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无上电/掉电复位发生; 1: 发生上电/掉电复位。			
位 26	<b>PINRSTF:</b> NRST 引脚复位标志 (PIN reset flag) 在 NRST 引脚复位发生时由硬件置'1'; 由软件通过写 RSTFC 位清除。 0: 无 NRST 引脚复位发生; 1: 发生 NRST 引脚复位。			
位 25	保留, 读操作返回 0			
位 24	<b>RSTFC:</b> 清除复位标志 (Reset flag clear) 由软件置'1'来清除复位标志。 0: 无作用; 1: 清除复位标志。			
位 23: 2	保留, 读操作返回 0			
位 1	<b>LSISTBL:</b> 内部低速振荡器就绪 (Internal low-speed oscillator ready) 由硬件置'1'或清'0'来指示内部 40kHz RC 振荡器是否就绪。 0: 内部 40kHz RC 振荡器时钟未就绪; 1: 内部 40kHz RC 振荡器时钟就绪。			
位 0	<b>LSIEN:</b> 内部低速振荡器使能 (Internal low-speed oscillator enable) 由硬件置'1'来控制内部 40kHz RC 振荡器开启。当使用 LSI 时, 必须开启。 0: 内部 40kHz RC 振荡器时钟关闭; 1: 内部 40kHz RC 振荡器时钟开启。			

### 3.3.11 AHB外设复位寄存器 (RCC\_AHBRST)

偏移地址: 0x28

复位值: 0x0000 0000

访问：无等待周期，字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									GPIOF RST		保留	GPIOC RST	GPIOB RST	GPIOA RST	保留
					res				rw		res	rw	rw	rw	res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									保留						
									res						
位 31: 23	保留，始终读为 0。														
位 22	<b>GPIOFRST: GPIOF 复位 (GPIOF reset)</b> 由软件置'1'或清'0' 0: 无作用； 1: 复位 GPIOF。														
位 21:20	保留，始终读为 0。														
位 19	<b>GPIOCRST: GPIOC 复位 (GPIOC reset)</b> 由软件置'1'或清'0' 0: 无作用； 1: 复位 GPIOC。														
位 18	<b>GPIOBRST: GPIOB 复位 (GPIOB reset)</b> 由软件置'1'或清'0' 0: 无作用； 1: 复位 GPIOB。														
位 17	<b>GPIOARST: GPIOA 复位 (GPIOA reset)</b> 由软件置'1'或清'0' 0: 无作用； 1: 复位 GPIOA。														
位 11: 0	保留，始终读为 0。														

### 3.3.12 PLL配置寄存器 (RCC\_PLL)

偏移地址: 0x2C

复位值: 0x0000 1F10

访问：无等待周期，字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLL CFGGEN	保留					PLL_FREF				保留					PLL_NS [8]
rw	res					rw				res					rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									PLL_NS[7:0]		PLL_MS[3:0]		保留		PLL_FR[2:0]
									rw		rw		res		rw
位 31	<b>PLLCFGGEN: PLL 配置使能 (PLL Configure Enable)</b> 由软件置'1'或清'0' 0: PLL 使用常规整数倍频配置模式，使用 PLL_FREF 和 PLLMUL 寄存器配置 PLL 1: PLL 使用灵活配置模式，使用 PLL_MS/PLL_NS/PLL_FR 寄存器值配置 PLL。														
位 30: 27	保留，始终读为 0。														

位 26: 24	<b>PLL_REF:</b> PLL 输入时钟选择, 仅在 PLLCFGGEN=0 时起作用 000: PLL 使用 3.9 MHz ~ 5 MHz 输入时钟; 001: PLL 使用 5.2 MHz ~ 6.25 MHz 输入时钟; 010: PLL 使用 7.8125 MHz ~ 8.33 MHz 输入时钟; 011: PLL 使用 8.33 MHz ~ 12.5 MHz 输入时钟; 100: PLL 使用 15.625 MHz ~ 20.83 MHz 输入时钟; 101: PLL 使用 20.83 MHz ~ 31.255 MHz 输入时钟; 110: 保留; 111: 保留。
位 23: 17	保留, 始终读为 0。
位 16: 8	<b>PLL_NS:</b> PLL 倍频系数 PLL_NS 范围 (31~500)
位 7: 4	<b>PLL_MS:</b> PLL 预分频系数 PLL_MS 范围 (1~15)
位 3	保留, 始终读为 0。
位 2: 0	<b>PLL_FR:</b> PLL 后分频配置值 PLL_FR 范围 (0~5) 000: PLL 后分频系数为 1, 1 分频输出; 001: PLL 后分频系数为 2, 2 分频输出; 010: PLL 后分频系数为 4, 4 分频输出; 011: PLL 后分频系数为 8, 8 分频输出; 100: PLL 后分频系数为 16, 16 分频输出; 101: PLL 后分频系数为 32, 32 分频输出; 其他: 保留 请注意 PLL_FR 值和后分频系数对应关系

注意: PLL 时钟计算公式

$$\text{PLL 输出时钟} = \text{PLL 输入时钟} \times \text{PLL 倍频系数} / (\text{PLL 预分频系数} \times \text{PLL 后分频系数})$$

$$500\text{MHz} \leq \text{PLL 输入时钟} \times \text{PLL 倍频系数} / \text{PLL 预分频系数} \leq 1000\text{MHz}$$

$$2\text{MHz} \leq \text{PLL 输入时钟} / \text{PLL 预分频系数} \leq 16\text{MHz}$$

### 3.3.13 额外寄存器 (RCC\_MISC)

偏移地址: 0x30

复位值: 0x0010 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCOPRE		保留		HSI_DIV_EN		保留		CLK_EFC_SR_C		保留		CLKOUT[3]			
res		rw		res		res		rw		res		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								HSICAL_KEY[7:0]							
res								rw							

位 31: 28	<b>MCOPRE[3:0]:</b> 微控制器时钟输出预分频值 (Microcontroller clock output pre-divider) 1000: 预分频为 2; 1001: 预分频为 4; 1010: 预分频为 8; 1011: 预分频为 16; 1100: 预分频为 64; 1101: 预分频为 128; 1110: 预分频为 256; 1111: 预分频为 512; Others: 预分频为 1。
位 27: 26	保留, 读操作返回 0
位 25	<b>HSI_DIV_EN:</b> HSI 6 分频使能 (HSI divider 6 enable) 0: HSI 6 分频使能 1: HSI 6 分频不使能
位 24: 21	保留, 读操作返回 0
位 20	<b>CLKEFC_SRC:</b> 闪存控制器时钟来源 (EFC Clock Source) 0: EFC 时钟源是 8M HSI; 1: EFC 时钟源是 HCLK。
位 19: 17	保留, 读操作返回 0
位 16	<b>CLKOUT[3]:</b> 微控制器时钟输出 (Microcontroller clock output) 搭配 RCC_CFG 寄存器位 26:24 使用
位 15:8	保留, 读操作返回 0
位 7:0	<b>HSICAL_KEY[7:0]:</b> HSICAL 写入键值。 此字段为 0x5A 时, HSICAL [7:0]才可被写入。

### 3.3.14 额外寄存器 (RCC\_MISC2)

偏移地址: 0x54

复位值: 0x0000 000D

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															
res															
位 31:10	保留, 读操作返回 0														
位 9	<b>HSISYSCTRL:</b> HSI 作为系统时钟的频率选择位 当 SYSCLKSEL 选择 HSI 为时钟源时, SYSCLK 的频率为 0: 固定是 8Mhz, 即选择 HSI 时钟的 6 分频 1: 根据 HSI_DIV_EN 设定可能为 48M 或 8M														
位 8: 6	保留, 读操作返回 0														

位 5: 4	<b>AUTO_STEP_EN:</b> 自动滑顺频率切换使能 为使切换系统时钟源到 PLL 或是切换 AHB 预分频由大到小时平顺(系统频率由小变大) , 建议系统频率操作目标大于 120MHz 时启动自动滑顺频率切换。 当自动滑顺频率切换功能作用时, 硬件会暂停 AHB 总线, 直到整个自动滑顺频率切换完成才恢复。此期间 DMA 仍正常工作, 中断事件会被记忆并待 AHB 总线恢复后 NVIC 即可处理。 00: 关闭自动滑顺频率切换 01: 保留 10: 保留 11: 开启自动滑顺频率切换, 当 AHPBSC 或 SYSCLKSEL 两个寄存器被改动时, 会自动触发自动滑顺频率切换功能
位 3:0	保留

## 4 内嵌闪存控制器 (EFC)

### 4.1 EFC简介

内嵌的闪存存储器可以用于在线编程 (ICP) 或在程序中编程 (IAP) 烧写。

在线编程 (In-Circuit Programming - ICP) 方式用于更新闪存存储器的全部内容，它通过 SWD 协议或系统加载程序 (Bootloader) 下载用户应用程序到微控制器中。ICP 是一种快速有效的编程方法，消除了封装和管座的困扰。

与 ICP 方式对应，在程序中编程 (In-Application Programming - IAP) 可以使用微控制器支持的任一种通信接口（如 I/O 端口、UART、I2C、SPI 等）下载程序或数据到存储器中。IAP 允许用户在程序运行时重新烧写闪存存储器中的内容。然而，IAP 要求至少有一部分程序已经使用 ICP 烧到闪存存储器中。

闪存接口是在 AHB 协议上实现了对指令和数据的访问，它通过对存储器的预取缓存，加快了存储器的访问；闪存接口还实现了在所有工作电压下对闪存编程和擦除所需的逻辑电路，这里还包括访问和写入保护以及选择字节的控制。

### 4.2 主要特点

- 多达 64K 字节闪存
- 带预取缓冲器的读接口（可配置 2x32 位或者 4x32 位）
- 选择字节加载
- 闪存编程/擦除操作
- 读出/写入保护
- HEX 加密保护
- 系统存储器区域作为主存扩展使用
- 密码保护指定范围程序区（安全库区），安全库区包含主存存储区或主存扩展区
- 32 位 CRC 校验
- 低功耗模式

#### 4.2.1 闪存模块组织

存储器可组织成 64 个 1K 字节/页的主存储器块和一个信息块，见 [表 4-1](#)。

存储器可组织成 32 个 1K 字节/页的主存储器块和一个信息块，见 [表 4-2](#)。

存储器可组织成 16 个 1K 字节/页的主存储器块和一个信息块，见 [表 4-3](#)。

表格 4-1 64KB 闪存模块的组织

块	名称	地址范围	长度 (字节)
主存储器	页 0	0x0800 0000 – 0x0800 03FF	1K
	页 1	0x0800 0400 – 0x0800 07FF	1K
	页 2	0x0800 0800 – 0x0800 0BFF	1K
	页 3	0x0800 0C00 – 0x0800 0FFF	1K
	页 4	0x0800 1000 – 0x0800 13FF	1K
	...	...	...
	页 63	0x0800 FC00 – 0x0800 FFFF	1K
信息块	系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K
	用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512
闪存存储器 接口寄存器	FLASH_ACR	0x4002 2000 – 0x4002 2003	4
	FLASH_FCKEY	0x4002 2004 – 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4
	FLASH_STS	0x4002 200C – 0x4002 200F	4

	FLASH_CTRL	0x4002 2010 – 0x4002 2013	4
	FLASH_ADDR	0x4002 2014 – 0x4002 2017	4
	保留	0x4002 2018 – 0x4002 201B	4
	FLASH_UOB	0x4002 201C – 0x4002 201F	4
	FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4
	保留	0x4002 2024 - 0x4002 2073	80
	FLASH_CDR0	0x4002 2074 - 0x4002 2077	4
	FLASH_CDR1	0x4002 2078 - 0x4002 207B	4
	FSLIB_PSW	0x4002 207C - 0x4002 207F	4
	FSLIB_PSW_STS	0x4002 2080 - 0x4002 2083	4
	FLASH_CRC_AR	0x4002 2084 - 0x4002 2087	4
	FLASH_CRC_CTRL	0x4002 2088 - 0x4002 208B	4
	FLASH_CRC_OUTR	0x4002 208C - 0x4002 208F	4
	保留	0x4002 2090 - 0x4002 215F	208
	FSLIB_SET_PSW	0x4002 2160 - 0x4002 2163	4
	FSLIB_SET_RANGE	0x4002 2164 - 0x4002 2167	4
	SYS_MEM_SLIB_SET	0x4002 2168 - 0x4002 216B	4
	SYS_MEM_BOOT_DIS_SET	0x4002 216C - 0x4002 216F	4
	FSLIB_KEYR	0x4002 2170 - 0x4002 2173	4

表格 4-2 32KB闪存模块的组织

块	名称	地址范围	长度(字节)
主存储器	页 0	0x0800 0000 – 0x0800 03FF	1K
	页 1	0x0800 0400 – 0x0800 07FF	1K
	页 2	0x0800 0800 – 0x0800 0BFF	1K
	页 3	0x0800 0C00 – 0x0800 0FFF	1K
	页 4	0x0800 1000 – 0x0800 13FF	1K
	.	.	.
	页 31	0x0800 7C00 – 0x0800 7FFF	1K
信息块	系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K
	用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512
闪存存储器 接口寄存器	FLASH_ACR	0x4002 2000 – 0x4002 2003	4
	FLASH_FCKEY	0x4002 2004 – 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4
	FLASH_STS	0x4002 200C – 0x4002 200F	4
	FLASH_CTRL	0x4002 2010 – 0x4002 2013	4
	FLASH_ADDR	0x4002 2014 – 0x4002 2017	4
	保留	0x4002 2018 – 0x4002 201B	4
	FLASH_UOB	0x4002 201C – 0x4002 201F	4
	FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4
	保留	0x4002 2024 - 0x4002 2073	80
	FLASH_CDR0	0x4002 2074 - 0x4002 2077	4
	FLASH_CDR1	0x4002 2078 - 0x4002 207B	4
	FSLIB_PSW	0x4002 207C - 0x4002 207F	4
	FSLIB_PSW_STS	0x4002 2080 - 0x4002 2083	4
	FLASH_CRC_AR	0x4002 2084 - 0x4002 2087	4
	FLASH_CRC_CTRL	0x4002 2088 - 0x4002 208B	4

	FLASH_CRC_OUTR	0x4002 208C - 0x4002 208F	4
	保留	0x4002 2090 - 0x4002 215F	208
	FSLIB_SET_PSW	0x4002 2160 - 0x4002 2163	4
	FSLIB_SET_RANGE	0x4002 2164 - 0x4002 2167	4
	SYS_MEM_SLIB_SET	0x4002 2168 - 0x4002 216B	4
	SYS_MEM_BOOT_DIS_SET	0x4002 216C - 0x4002 216F	4
	FSLIB_KEYR	0x4002 2170 - 0x4002 2173	4

表格 4-3 16KB闪存模块的组织

块	名称	地址范围	长度(字节)
主存储器	页 0	0x0800 0000 – 0x0800 03FF	1K
	页 1	0x0800 0400 – 0x0800 07FF	1K
	页 2	0x0800 0800 – 0x0800 0BFF	1K
	页 3	0x0800 0C00 – 0x0800 0FFF	1K
	页 4	0x0800 1000 – 0x0800 13FF	1K
	.	.	.
	页 15	0x0800 3C00 – 0x0800 3FFF	1K
	系统存储器区	0x1FFF E400 – 0x1FFF F3FF	4K
	用户选择字节	0x1FFF F800 – 0x1FFF F9FF	512
闪存存储器 接口寄存器	FLASH_ACR	0x4002 2000 – 0x4002 2003	4
	FLASH_FKEY	0x4002 2004 – 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 – 0x4002 200B	4
	FLASH_STS	0x4002 200C – 0x4002 200F	4
	FLASH_CTRL	0x4002 2010 – 0x4002 2013	4
	FLASH_ADDR	0x4002 2014 – 0x4002 2017	4
	保留	0x4002 2018 – 0x4002 201B	4
	FLASH_UOB	0x4002 201C – 0x4002 201F	4
	FLASH_WRPRT	0x4002 2020 – 0x4002 2023	4
	保留	0x4002 2024 – 0x4002 2073	80
	FLASH_CDR0	0x4002 2074 – 0x4002 2077	4
	FLASH_CDR1	0x4002 2078 – 0x4002 207B	4
	FSLIB_PSW	0x4002 207C – 0x4002 207F	4
	FSLIB_PSW_STS	0x4002 2080 – 0x4002 2083	4
	FLASH_CRC_AR	0x4002 2084 – 0x4002 2087	4
	FLASH_CRC_CTRL	0x4002 2088 – 0x4002 208B	4
	FLASH_CRC_OUTR	0x4002 208C – 0x4002 208F	4
	保留	0x4002 2090 – 0x4002 215F	208
	FSLIB_SET_PSW	0x4002 2160 – 0x4002 2163	4
	FSLIB_SET_RANGE	0x4002 2164 – 0x4002 2167	4
	SYS_MEM_SLIB_SET	0x4002 2168 – 0x4002 216B	4
	SYS_MEM_BOOT_DIS_SET	0x4002 216C – 0x4002 216F	4
	FSLIB_KEYR	0x4002 2170 – 0x4002 2173	4

闪存存储器被组织成 32 位宽的存储器单元，可以存放代码和数据常数。每一个 AT32F421 微控制器的闪存模块都有一个特定的启始地址。

信息块分为两个部分：

- 系统存储器用于存放在系统存储器自举模式下的启动程序，这个区域只保留给 Artery

使用，启动程序使用 **USART1** 或 **USART2** 实现对闪存存储器的烧写；Artery 在生产线上对这个区域烧写并锁定以防止用户擦写，另外还可以把系统存储器做为用户代码区使用。

### ● 选择字节

对主存储器和信息块的写入由内嵌的闪存烧写/擦除控制器（**FPEC**）管理；烧写与擦除的高电压由内部产生。

闪存存储器有两种保护方式防止非法的访问（读、写、擦除）：

页写入保护。

读出保护。（详情请参考 [4.3.3 节](#)）

在执行闪存写操作时，任何对闪存的读操作都会锁住总线，在写操作完成后读操作才能正确地进行；即在进行写或擦除操作时，不能进行代码或数据的读取操作。

进行闪存操作时（写或擦除），必须打开内部的 RC 振荡器（**HSI**）。闪存存储器可以用 **ICP** 或 **IAP** 方式烧写。

注意：在低功耗模式下，所有闪存存储器的操作都被中止。

## 4.3 功能描述

### 4.3.1 读操作

内置闪存模块可以在通用地址空间直接寻址，任何 8/16/32 位数据的读操作都能访问闪存模块的内容并得到相应的数据。

读接口在闪存端包含一个读控制器，还包含一个 **AHB** 接口与 CPU 衔接。这个接口的主要工作是产生读闪存的控制信号并预取 CPU 要求的指令块，预取指令块仅用于在 **I-Code** 总线上的取指操作，数据常量是通过 **D-Code** 总线访问的。这两条总线的访问目标是相同的闪存模块，访问 **D-Code** 将比预取指令优先级高。

#### 4.3.1.1 取指令

**Cortex™-M4** 在 **I-Code** 总线上取指令，在 **D-Code** 总线上取数据。预取指令块可以有效地提高对 **I-Code** 总线访问的效率。

预取缓冲器包含两个数据块，每个数据块有 16 个字节；预取指令（数据）块直接映像到闪存中，因为数据块的大小与闪存的宽度相同，所以读取预取指令块可以在一个读周期完成。

设置预取缓冲器可以使 CPU 更快地执行，CPU 读取一个字的同时下一个字已经在预取缓冲器中等候，即当代码跳转的边界为 16 字节的倍数时，闪存的加速比例为 2。

预取控制器根据预取缓冲器中可用的空间决定是否访问闪存，预取缓冲器中有至少一块的空余空间时，预取控制器则启动一次读操作。

清除闪存访问控制寄存器中的一个控制位能够关闭预取缓冲器。芯片复位后预取缓冲器处于开启状态。

注意：当 **AHB** 时钟的预分频系数不为‘1’时，必须打开预取缓冲器（**FLASH\_ACR[4]=1**）。

如果在系统中没有高频率的时钟，即 **HCLK** 频率较低时，闪存的访问只需半个 **HCLK** 周期（半周期的闪存访问只能在时钟频率低于 8MHz 时进行，使用 **HSI** 或 **HSE** 并且关闭 **PLL** 时可得到这样的频率）；在闪存访问控制寄存器中有一个控制位可以选择这种工作方式。

注意：当使用了预取缓冲器和 **AHB** 时钟的预分频系数不为‘1’时，不能使用半周期访问方式。

访问时间调节器为了维持读闪存的控制信号，预取控制器的时钟周期与闪存访问时间的比例由闪存访问控制器控制；这个值给出了能够正确地读取数据时，闪存控制信号所需的时钟周期数目；芯片复位后，该值为‘0’，闪存访问仅为一个时钟周期。

#### 4.3.1.2 D-Code 接口

**D-Code** 接口包含 CPU 端简单的 **AHB** 接口和对闪存访问控制器的仲裁器提出访问请求的逻辑电路。**D-Code** 的访问优先于预取指令的访问。这个接口使用预取缓冲器的访问时间调节器模块。

### 4.3.1.3 闪存访问控制器

这个模块就是在 I-Code 上的指令预取请求和 D-Code 接口上读请求的仲裁器。D-Code 接口的请求优先于 I-Code 的请求。

## 4.3.2 闪存编程和擦除控制器 (FPEC)

FPEC 模块处理闪存的编程和擦除操作，它包括 7 个 32 位的寄存器，只要 CPU 不访问闪存，闪存操作不会延缓 CPU 的执行。

- FPEC键寄存器 (FLASH\_FCKEY)
- 选择字节键寄存器 (FLASH\_OPTKEYR)
- 闪存状态寄存器 (FLASH\_STS)
- 闪存控制寄存器 (FLASH\_CTRL)
- 闪存地址寄存器 (FLASH\_ADDR)
- 选择字节寄存器 (FLASH\_UOB)
- 写保护寄存器 (FLASH\_WRPRT)

### 4.3.2.1 键值

共有三个键值：

- RDPRTEN 键 = 0x00A5
- KEY1 = 0x45670123
- KEY2 = 0xCDEF89AB

### 4.3.2.2 解除闪存锁

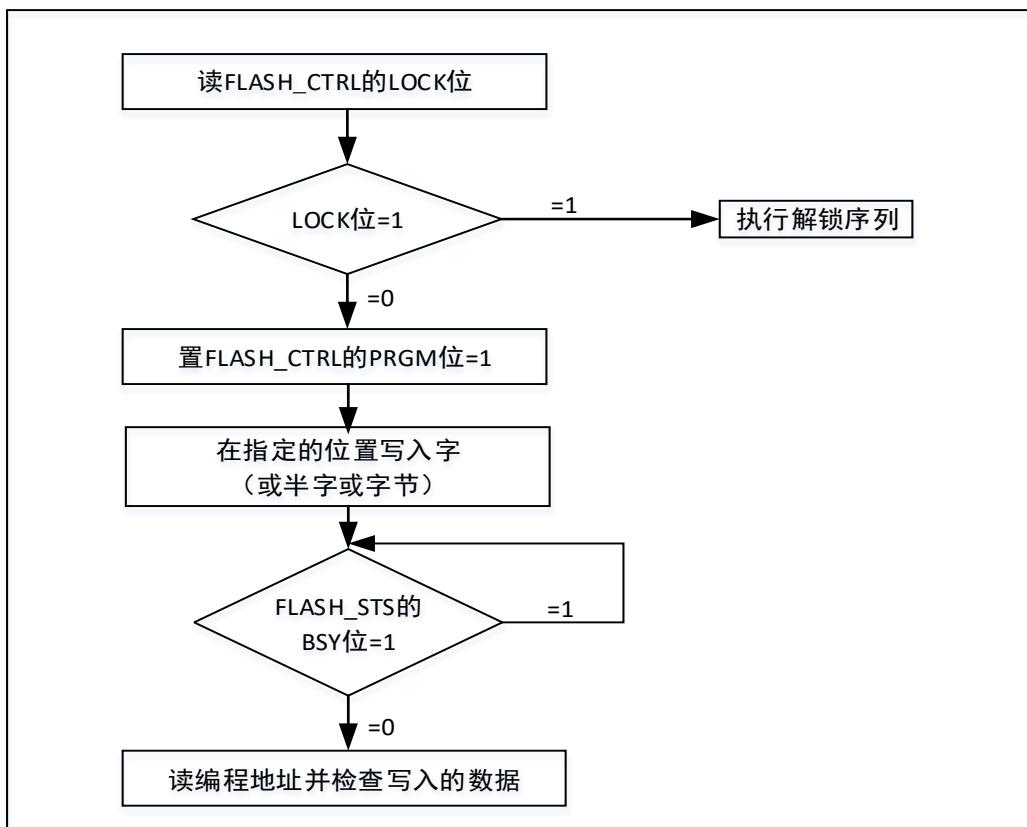
复位后，FPEC 模块是被保护的，不能写入 FLASH\_CTRL 寄存器；通过写入特定的序列到 FLASH\_FCKEY 寄存器可以打开 FPEC 模块，这个特定的序列是在 FLASH\_FCKEY 写入两个键值 (KEY1 和 KEY2，见 [4.3.2.1](#) 节)；错误的操作序列都会在下次复位前锁死 FPEC 模块和 FLASH\_CTRL 寄存器。

写入错误的键序列还会产生总线错误；总线错误发生在第一次写入的不是 KEY1，或第一次写入的是 KEY1 但第二次写入的不是 KEY2 时；FPEC 模块和 FLASH\_CTRL 寄存器可以由程序设置 FLASH\_CTRL 寄存器中的 LOCK 位锁住，这时可以通过在 FLASH\_FCKEY 中写入正确的键值对 FPEC 解锁。

### 4.3.2.3 主闪存编程

对主闪存编程每次可以写入 32,16 或 8 位。当 FLASH\_CTRL 寄存器的 PRGM 位为‘1’时，在一个闪存地址写入一个字(或半字或字节)将启动一次编程。在编程过程中 (BSY 位为‘1’)，任何读写闪存的操作都会使 CPU 暂停，直到此次闪存编程结束。

图表 4-1 编程过程



### 标准编程

这种模式下 CPU 以标准的写数据的方式烧写闪存, FLASH\_CTRL 寄存器的 PRGM 位必须置’1’。FPEC 先读出指定地址的内容并检查它是否被擦除, 如未被擦除则不执行编程并在 FLASH\_STS 寄存器的 PRGMFLR 位提出警告 (唯一的例外是当要烧写的数值是 0 时, 0 可被正确烧入且 PRGMFLR 位不被置位); 如果指定的地址在 FLASH\_WRPRT 中设定为写保护, 则不执行编程并在 FLASH\_STS 寄存器的 WRPTFLR 位置’1’ 提出警告。FLASH\_STS 寄存器的 PRCDN 为’1’ 时表示编程结束。

标准的闪存编程顺序如下:

- 检查FLASH\_STSx寄存器的BSY位, 以确认没有其他正在进行的编程操作;
- 设置FLASH\_CTRLx寄存器的PRGM位为’1’;
- 在指定的地址写入要编程的数据;
- 等待BSY位变为’0’;
- 读出写入的地址并验证数据。

**注意:** 当 FLASH\_STSx 寄存器的 BSY 位为’1’ 时, 不能对任何寄存器执行写操作。

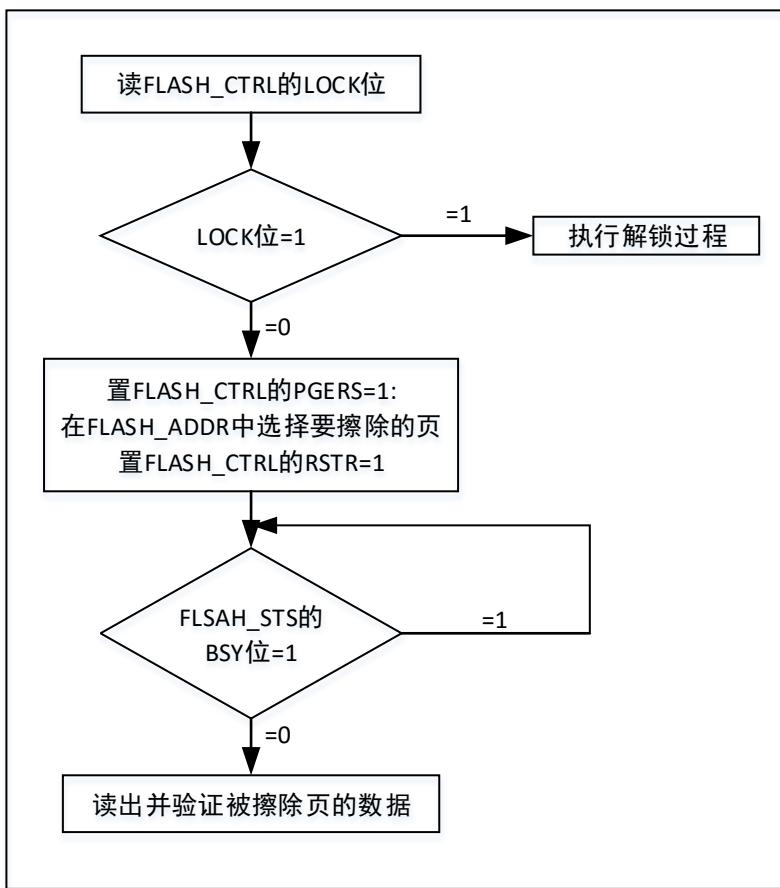
### 4.3.2.4 闪存擦除

闪存可以按页擦除, 也可以整片擦除。作为主存扩展区的系统存储器区域的页擦除实际为整个系统存储器区域擦除。如果系统存储器区域作为主存扩展区, 则整片擦除包括主存擦除和系统存储器区域擦除。

闪存的任何一页都可以通过 FPEC 的页擦除功能擦除; 擦除一页应遵守下述过程:

- 检查FLASH\_STSx寄存器的BSY位, 以确认没有其他正在进行的闪存操作;
- 设置FLASH\_CTRLx寄存器的PGERS位为’1’;
- 用FLASH\_ADDRx寄存器选择要擦除的页;
- 设置FLASH\_CTRLx寄存器的RSTR位为’1’;
- 等待BSY位变为’0’;
- 读出被擦除的页并做验证。

图表 4-2 闪存页擦除过程

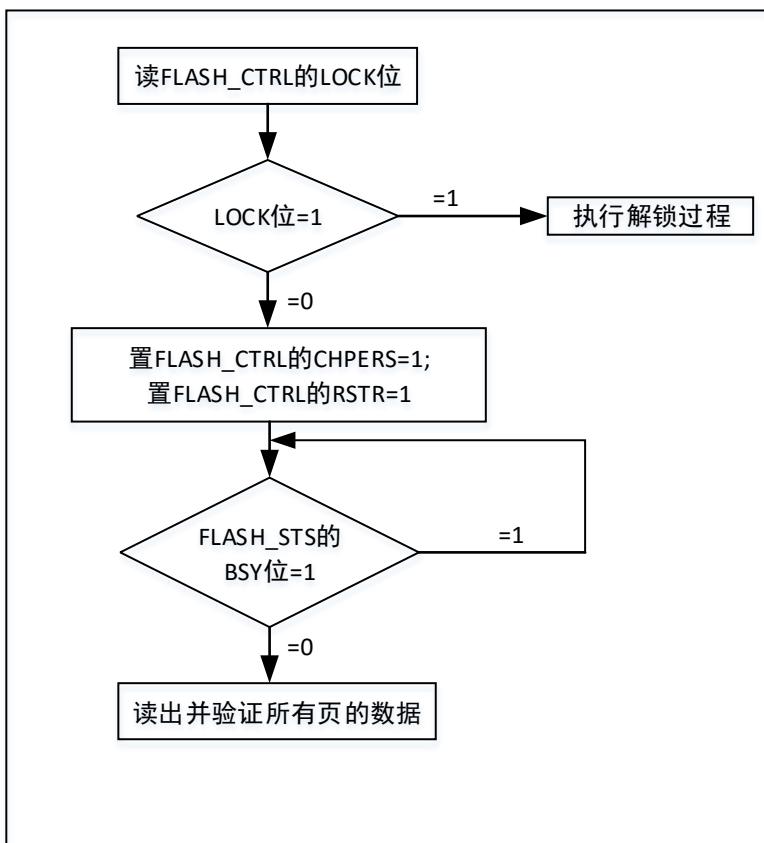


### 整片擦除

可以用整片擦除功能擦除所有用户区的闪存，信息块不受此操作影响。建议使用下述过程：

- 检查FLASH\_STS寄存器的BSY位，以确认没有其他正在进行的闪存操作；
- 设置FLASH\_CTRL寄存器的CHPERS位为‘1’；
- 设置FLASH\_CTRL寄存器的RSTR位为‘1’；
- 等待BSY位变为‘0’；
- 读出所有页并做验证。

图表 4-3 闪存整片擦除过程



#### 4.3.2.5 选择字节编程

对选择字节的编程与普通的用户地址不同。选择字节的数目有 256 个字节（4 个字节作为写保护，1 个字节作为读保护，1 个字节为配置选项，250 个字节存储用户数据）。对 FPEC 解锁后，必须分别写入 KEY1 和 KEY2（见 [4.3.2.1 节](#)）到 FLASH\_OPTKEYR 寄存器，FLASH\_CTRL 寄存器的 UOBWE 位会被设置为‘1’，此时可以对选择字节进行编程：设置 FLASH\_CTRL 寄存器的 UOBPRGM 位为‘1’ 后写入半字到指定的地址。

FPEC 先读出指定地址的选择字节内容并检查它是否已经被擦除，如未被擦除则不执行编程并在 FLASH\_STS 寄存器的 WRPRTFLR 位提出警告。FLASH\_STS 寄存器的 PRCDN 为‘1’ 时表示编程结束。

FPEC 使用半字中的低字节并自动地计算出高字节（高字节为低字节的反码），并开始编程操作，这将保证选择字节和它的反码始终是正确的。

烧写编程的顺序如下：

- 检查 FLASH\_STS 寄存器的 BSY 位，以确认没有其他正在进行的编程操作；
- 解锁 FLASH\_CTRL 寄存器的 UOBWE 位；
- 设置 FLASH\_CTRL 寄存器的 UOBPRGM 位为‘1’；
- 写入要编程的半字到指定的地址；
- 等待 BSY 位变为‘0’；
- 读出写入的地址并验证数据。

当读闪存保护选项从“保护”变为“未保护”时，在重新设置读保护选项前会自动执行一个整片擦除用户闪存的操作。如果用户要改变读保护之外的选项，则不会出现整片擦除操作。读保护选项上的这一擦除操作保护了闪存中的内容不被非法读出。

#### 擦除过程

选择字节的擦除操作顺序（OPTERASE）如下：

- 检查FLASH\_STS寄存器的BSY位，以确认没有其他正在进行的闪存操作；
- 解锁FLASH\_CTRL寄存器的UOBWE位；
- 设置FLASH\_CTRL寄存器的UOBERS位为'1'；
- 设置FLASH\_CTRL寄存器的RSTR位为'1'；
- 等待BSY位变为'0'；
- 读出被擦除的选择字节并做验证。

### 4.3.3 保护

闪存中的用户代码区开启读保护可以防止非法的读出；同样可以对闪存区的页加以写保护，防止在程序跑飞的情况下不被意外地改变，写保护的基本单位为1页。

#### 4.3.3.1 写保护

如果试图在一个受保护的页面进行编程或擦除操作，在闪存状态寄存器（FLASH\_STS）中会返回一个保护错误标志。

解除写保护操作顺序如下：

- 使用闪存控制寄存器（FLASH\_CTRL）的UOBERS位擦除整个选择字节区域；
- 写入正确的RDP代码0xA5，允许读访问；
- 进行系统复位，重装载选择字节（包含新的WRP [3: 0]字节）；写保护被解除。

#### 4.3.3.2 读保护

读保护是通过设置RDP选择字节启动的，并通过系统复位加载RDP选择字节。当保护字节被写入相应的值以后，实施下述保护：

- 只允许从用户代码中对主闪存存储器的读操作（以非调试方式从主闪存存储器启动）。
- 第0~3页被自动加上了写保护，其它部分的存储器可以通过在主闪存存储器中执行的代码进行编程（实现IAP或数据存储等功能），但不允许在调试模式下或在从内部SRAM启动后执行写或擦除操作（整片擦除除外）。

注意：若是系统存储器区域是用于存放用户代码，则系统存储器区域作为主闪存存储区的扩展，支持读保护功能

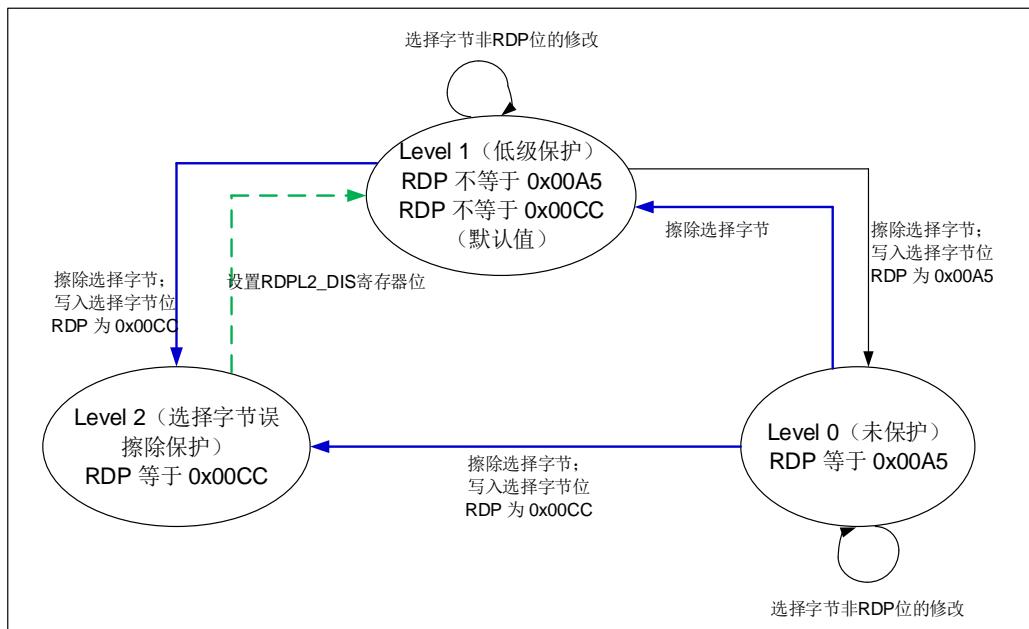
所有通过SWD向内置SRAM装载代码并执行代码的功能依然有效，亦可以通过SWD从内置SRAM启动，这个功能可以用来解除读保护。当读保护的选择字节转变为存储器未保护的数值时，将会执行整片擦除过程。

当RDP选择字节和它的反码包含下列数值对时，闪存被置于保护状态：

表格 4-4 闪存存储器保护状态

RDP字节的值	RDP反码的值	读保护状态
任意值	非RDP字节的反码	低级保护
0xCC	0x33	RDP使能及选择字节误擦除保护
0xA5	0x5A	未保护
任意值	非RDP字节的反码	保护

图表 4-4 读保护等级切换状态图



表格 4-5 读保护等级切换说明

主存区域	保护等级	调试模式, 或从SRAM启动模式			从主存启动或是从程序启动代码区启动		
		读	写	擦除	读	写	擦除
用户代码	低级保护	禁止		禁止(1)		允许	
	RDP使能及选择字节误擦除保护	禁止			允许		
选择字节区	低级保护	禁止	允许		允许		
	RDP使能及选择字节误擦除保护	禁止		禁止(2)	禁止	禁止(2)	

(1) 用户代码只会在解除读保护时被硬件自动擦除

(2) 选择字节只能通过设置寄存器位RDPL2\_DIS 被硬件自动擦除

**注意:** 擦除选择字节块将不会导致自动的整片擦除操作, 因为擦除的结果 0xFF 相当于保护状态。

解除 RDP 使能及选择字节误擦除保护的过程是:

- 设置寄存器位 RDPL2\_DIS;
- 等待 BSY 位变为 '0';
- 读出被擦除的选择字节并做验证;
- 进行复位 (任意复位) 以重新加载选择字节 (和新的 RDP 代码), 此时 RDP 使能及选择字节误擦除保护被解除, 低级保护被设定。

解除低级读保护的过程是:

- 擦除整个选择字节区域, 读保护码 (RDP) 将变为 0xFF, 此时读保护仍然有效;
- 写入正确的 RDP 代码 0xA5 以解除存储器的保护, 该操作将首先导致对所有用户闪存的整片擦除操作;
- 进行复位 (任意复位) 以重新加载选择字节 (和新的 RDP 代码), 此时读保护被解除。

**注意:** 可以使用系统启动程序解除读保护 (此时只需执行系统复位即可重新加载选择字节)。

### 4.3.3.3 选择字节块写保护

默认状态下, 选择字节块始终是可以读且被写保护。要想对选择字节块进行写操作 (编程/擦除) 首先要在 OPTKEYR 中写入正确的键序列 (与上锁时一样), 随后允许对选择字节块的写操作, FLASH\_CTRL

寄存器的 UOBWE 位标示允许写，清除这位将禁止写操作。

#### 4.3.4 选择字节说明

选择字节共有 256 个字节，由用户根据应用的需要配置；例如：可以选择使用硬件模式的看门狗或软件的看门狗。

在选择字节中每个 32 位的字被划分为下述格式：

表格 4-6 选择字节格式

位 31~24	位 23~16	位 15~8	位 7~0
选择字节 1 的反码	选择字节 1	选择字节 0 的反码	选择字节 0

选择字节块中选择字节的组织结构如下：

表格 4-7 信息块的组织结构

地址	[31: 24]	[23: 16]	[15: 8]	[7: 0]
0x1FF_F800	uUSER	USER	nRDP	RDP
0x1FF_F804	nData1	Data1	nData0	Data0
0x1FF_F808	nWRP1	WRP1	nWRP0	WRP0
0x1FF_F80C	nWRP3	WRP3	nWRP2	WRP2
0x1FF_F810	nData3	Data3	nData2	Data2
0x1FF_F814	nData5	Data5	nData4	Data4
...	..	..	..	..
0x1FF_F9FC	nData249	Data249	nData248	Data248

表格 4-8 用户选择字节说明

<b>RDP:</b> 读出保护选择字节（读保护设定结果存放在寄存器 FLASH_UOB[1]以及[26]） 读出保护功能帮助用户保护存在闪存中的代码。该功能由设置信息块中的一个选择字节启用。写入正确的数值（RDPRTE 键=0x00A5）到这个选择字节后，闪存被开放允许读出访问。 注意： 1) 写入数值 0x00CC 到这个选择字节后，将启动读保护使能及选择字节误擦除保护功能，闪存将被禁止读出访问，并且选择字节也将禁止更改。 2) 写入数值非 0x00A5 以及非 0x00CC 到这个选择字节后，将启动低级闪存读保护功能，闪存将被禁止读出访问。	
<b>USR:</b> 用户选择字节（存放在寄存器 FLASH_UOB[9:2]） 这个字节用于配置下列功能：	
- 选择看门狗事件：硬件或软件 - 进入停机（STOP）模式时的复位事件 - 进入待机模式时的复位事件 - 选择启动模式事件	
位 23: 21	0x7: 不用
位 20	nBOOT1: 和 BOOT0 一起决定启动模式，当 BOOT0 = 1 时 0: 由 SRAM 启动 1: 由系统存储器区启动
位 19	0x1: 不用
位 18	nRST_STDBY 0: 当进入待机模式时产生复位 1: 进入待机模式时不产生复位
位 17	nRST_STOP 0: 当进入停机（STOP）模式时产生复位 1: 进入停机（STOP）模式时不产生复位
位 16	WDG_SW 0: 硬件看门狗 1: 软件看门狗

<p><b>Datax:</b> 用户数据 这个地址可以使用选择字节的编程方式编程。 <b>Data249:</b> 用户数据 249 <b>Data248:</b> 用户数据 248 ... <b>Data3:</b> 用户数据 3 <b>Data2:</b> 用户数据 2 <b>Data1:</b> 用户数据 1 (存放在寄存器 FLASH_UOB[25:18]) <b>Data0:</b> 用户数据 0 (存放在寄存器 FLASH_UOB[17:10])</p>
<p><b>WRPx:</b> 闪存写保护选择字节 (存放在寄存器 FLASH_WRPRT) 选择字节 WRPx 中的每一个比特位用于保护主存储器中 4 个存储页 (1K 字节/页)，WRP3 的位 7 用于保护系统存储器区域 (当作为主存扩展区存放用户代码)： 0: 实施写保护 1: 不实施写保护 四个用户选择字节用于保护总共 64K 字节的主存储器。 <b>WRP0:</b> 第 0~31 页的写保护 (存放在寄存器 FLASH_WRPRT[7:0]) <b>WRP1:</b> 第 32~63 页的写保护 (存放在寄存器 FLASH_WRPRT[15:8]) <b>WRP2:</b> 不用 (存放在寄存器 FLASH_WRPRT[23:16]) <b>WRP3:</b> 位 0~6 不用；位 7 用于系统存储器区域的写保护 (存放在寄存器 FLASH_WRPRT[31:24])</p>

每次系统复位后，选择字节装载器读出信息块的数据并保存在寄存器中；每个选择位都在信息块中有它的反码位，在装载选择位时反码位用于验证选择位是否正确，如果有任何的差别，将产生一个选择字节错误标志 (UOBFLR)。当发生选择字节错误时，对应的选择字节被强置为 0xFF。当选择字节和它的反码均为 0xFF 时（擦除后的状态）——关闭上述验证功能。

所有的选择位（不包括它们的反码位）用于配置该微控制器，CPU 可以读选择寄存器，详见[第 4.4 章](#)寄存器说明。

## 4.3.5 特殊功能

### 4.3.5.1 安全库区设定

设定以密码保护指定范围的程序区，即安全库区。安全库区划分为指令区与唯读区。

#### 设定安全库区的益处：

以密码保护安全库区，方案商可刻录核心算法到此区域；

指令安全库区仅能执行，无法被读取 (I-Code 总线除外)，除非输入方案商指定密码，也无法删除(包含 ISP/IAP/SWD)；

其余空白程序区可以提供给方案商客户进行二次开发；

方案商可以藉由安全库功能销售核心算法，不需要每个客户都开发完整方案；

设定安全库区，可防止蓄意破坏或更改终端产品应用程序代码。

#### 注意：

安全库区代码必须以 1 页为单位进行烧录，并且起始地址与主存 (或主存扩展区) 地址对齐；

仅允许 I-Code 总线读取指令库区；

写入或擦除安全库区代码 (指令区和唯读区)，将在 FLASH\_STS 寄存器的 WRPRTFLR 位置'1' 提出警告；

可支持整个安全库区设定为指令区，或是设定为唯读区，或是部分指令区与部分唯读区。

默认状态下，安全库区设定寄存器始终是不可读且被写保护。要想对安全库区设定寄存器进行写操作，首先要对安全库区解锁，对 FSLIB\_KEYR 寄存器写入 0xA35F6D24 值，通过查看 FSLIB\_PSW\_STS 位 SLIB\_UNLOCK 确认解锁成功，随后允许对安全库区设定寄存器写入设定值。

启动主存安全库区的过程如下：

- 检查 FLASH\_STS 寄存器的 BSY 位，以确认没有其他正在进行的编程操作；
- 对 FSLIB\_KEYR 寄存器写入 0xA35F6D24，以进行安全库区解锁；
- 检查 FSLIB\_PSW\_STS 寄存器的 SLIB\_UNLOCK 位，以确认解锁成功；
- 如果将安全库区设在主存内，则在 FSLIB\_SET\_RANGE 寄存器设定要保护的区域，包

含指令区与唯读区的地址。如果将安全库区设在主存扩展区，则设定 **SYS\_MEM\_SLIB\_SET** 寄存器；只能将主存和主存扩展区二者之一设为安全库区，不能同时设为安全库区；

- 等待 **BSY** 位变为'0'；
- 在 **FSLIB\_SET\_PSW** 寄存器设定安全区域密码；
- 等待 **BSY** 位变为'0'；
- 烧录将存入安全库区的代码；
- 进行系统复位，重装载安全库区设定字；
- 读出 **FLASH\_CDR0/CDR1** 寄存器用于判断安全库区设定结果。

解除安全库区的过程是：

- 在 **FSLIB\_PSW** 寄存器写入先前设置的安全区域密码；
- 等待 **BSY** 位变为'0'；
- 进行系统复位，重装载安全库区设定字；
- 读出 **FLASH\_CDR0** 寄存器用于判断安全库区设定结果。

注意：解除安全库区将会执行主存的整片擦除（包括主存扩展区），以及安全库设定块擦除。

#### 4.3.5.2 系统存储器区域作为主存扩展使用

用户只有一次机会将系统存储器区域作为主存扩展使用。设定系统存储器区域作为主存扩展使用的过程是：

- 用户读取 **FLASH\_CDR0** 的位0，获知系统存储器区域当前的角色
- 对 **FSLIB\_KEYR** 寄存器写入 0xA35F6D24，以进行系统存储器区域设定解锁
- 写非 0xFF 到 **SYS\_MEM\_BOOT\_DIS\_SET** 寄存器的位 7-0
- 等待 **BSY** 位变为'0'；
- 进行系统复位，重装载设定字；
- 读出 **FLASH\_CDR0** 寄存器用于判断设定结果。

#### 4.3.5.3 CRC校验

以页为单位对安全库区代码或以 HEX 加密用户代码进行可选的 CRC 校验。

校验过程如下：

- 检查 **FLASH\_STS** 寄存器的 **BSY** 位，以确认没有其他正在进行的编程操作；
- 在 **FLASH\_CRC\_AR** 寄存器设定要校验的代码起始地址；
- 在 **FLASH\_CRC\_CTRL** 寄存器位 15~位 0，设定要校验的代码长度（单位是 1 页）；
- 在 **FLASH\_CRC\_AR** 寄存器设置位 16，启动 CRC 校验；
- 等待 **BSY** 位变为'0'；
- 读出 **FLASH\_CRC\_OUTR** 寄存器用于判断 CRC 校验结果。

注意：

**FLASH\_CRC\_AR** 寄存器设定值必须与页起始地址对齐；

不允许跨主存（或是跨主存扩展区）的 CRC 校验。

## 4.4 EFC寄存器

表格 4-9 闪存接口一寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
0x00	FLASH_ACR	保留																				PFRTB_HitCYC	PRFTBS2	PRFTBE2	PRFTBS	PRFTBE	HLFCYA	LATENCY[2: 0]	0														
		复位值																																									
0x04	FLASH_FCKEY	KEY[31: 0]																				0	0	0	1	1	0	0	0	0	0												
		复位值																																									
0x08	FLASH_OPTKEYR	OPTKEYR[31: 0]																				x	x	x	x	x	x	x	x	x	x												
		复位值																																									
0x0C	FLASH_STS	保留																				0	0	0	1	1	0	0	0	0	0												
		复位值																																									
0x10	FLASH_CTRL	保留																				DPDEN	RDPL2_DIS	PRCDN	WRPRTFLR	x	x	x	x	x	x												
		复位值																																									
0x14	FLASH_ADDR	TA[31: 0]																				0	0	0	0	0	0	0	0	0	0												
		复位值																																									
0x18		保留																				ERRIE	UOBWE	LOCK	RSTR	UOBERS	UOPRGM	WRPRTFLR	x	x	x	x											
0x1C	FLASH_UOB	Data1																																									
0x20	FLASH_WRPRT	WRPRTBMP[31: 0]																				nSTDBY_RST	nSCLKSEL_WD	RDPTEN	UOBFRL	CHPERS	PGERS	PRGMFLR	x	x	x	x											
		复位值																																									
0x74	FLASH_CDR0	SYS_MEM_SLIB_INSTR_PG																				保留																					

#### 4.4.1 闪存访问控制寄存器（FLASH\_ACR）

地址偏移: 0x00

复位值: 0x0000 0030

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							保留	PRFTB_HitCYC	PRFTBS2	PRFTBE2	PRFTBS	PRFTBE	HLFCYA	LATENCY	
res							rw	r	rw	r	rw	rw	rw	rw	rw
位 31~9	保留。必须保持为清除状态'0'														
位 8	<b>PRFTB_HitCYC:</b> 预取缓冲区访问时延使能 0: 启用预取缓冲区时延, 缓存访问需要等待 1 个时钟周期; 1: 关闭预取缓冲区时延, 缓存访问零等待。														
位 7	<b>PRFTBS2:</b> 预取缓冲区第二数据块状态 该位指示预取缓冲区第二数据块的状态 0: 预取缓冲区第二数据块关闭; 1: 预取缓冲区第二数据块开启。														
位 6	<b>PRFTBE2:</b> 预取缓冲区第二数据块使能 0: 关闭预取缓冲区第二数据块; 1: 启用预取缓冲区第二数据块。														
位 5	<b>PRFTBS:</b> 预取缓冲区状态 该位指示预取缓冲区的状态 0: 预取缓冲区关闭; 1: 预取缓冲区开启。														
位 4	<b>PRFTBE:</b> 预取缓冲区使能 0: 关闭预取缓冲区; 1: 启用预取缓冲区。														
位 3	<b>HLFCYA:</b> 闪存半周期访问使能 0: 禁止半周期访问; 1: 启用半周期访问。														
位 2~0	<b>LATENCY:</b> 时延 这些位表示 SYSCLK (系统时钟) 周期与闪存访问时间的比例 000: 零等待状态, 当 $0 < \text{SYSCLK} \leq 32 \text{ MHz}$ 001: 一个等待状态, 当 $32 \text{ MHz} < \text{SYSCLK} \leq 64 \text{ MHz}$ 010: 两个等待状态, 当 $64 \text{ MHz} < \text{SYSCLK} \leq 96 \text{ MHz}$ 011: 三个等待状态, 当 $96 \text{ MHz} < \text{SYSCLK} \leq 120 \text{ MHz}$														

#### 4.4.2 FPEC键寄存器 (FLASH\_FCKEY)

专用于闪存主存区域以及当系统存储器区域作为主存的扩展。

地址偏移: 0x04

复位值: XXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY[31: 16]															
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY[15: 0]															
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

注意: 所有这些位是只写的, 读出时返回 0。

位 31~0

**KEY: FPEC 键**

这些位用于输入 FPEC 的解锁键。

#### 4.4.3 闪存OPTKEY寄存器（FLASH\_OPTKEYR）

地址偏移: 0x08

复位值: xxxx xxxx

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPTKEYR[31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPTKEYR[15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

注意 所有这些位是只写的，读出时返回 0

位 31~0

**OPTKEYR: 选择字节键**

这些位用于输入选择字节的键以解除 UOBWE。

#### 4.4.4 闪存状态寄存器（FLASH\_STS）

专用于闪存区块。

地址偏移: 0x0C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															
RES								RW	RW	RW	RW	RW	RW	R	

位 31~6	保留。必须保持为清除状态'0'
位 5	<b>PRCDN: 操作结束</b> 当闪存操作（编程/擦除）完成时，硬件设置这位为'1'，写入'1'可以清除这位状态。 注：每次成功的编程或擦除都会设置 PRCDN 状态。
位 4	<b>WRPRTFLR: 写保护错误</b> 试图对写保护的闪存地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。
位 3	保留。必须保持为清除状态'0'
位 2	<b>PRGMFLR: 编程错误</b> 试图对内容不是'0xFFFF'的地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。 注：进行编程操作之前，必须先清除 FLASH_CTRL 寄存器的 RSTR 位。
位 1	保留。必须保持为清除状态'0'
位 0	<b>BSY: 繁忙标志</b> 该位指示闪存操作正在进行。在闪存操作开始时，该位被设置为'1'；在操作结束或发生错误时该位被清除为'0'。

#### 4.4.5 闪存控制寄存器（FLASH\_CTRL）

专用于闪存主存区块、用户选择字节以及当系统存储器区域作为主存的扩展。

地址偏移: 0x10

复位值: 0x0002 0080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留														DPDEN	RDPL2_DIS
res														rw	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	PRC DNIE	保 留	ER RIE	UOB WE	保 留	LOCK	RSTR	UOB ERS	UOBR GGM	保 留	CHP ERS	PGE RS	PR GM		
res	rw	res	rw	rw	res	rw	rw	rw	rw	res	rw	rw	rw	rw	rw
位 31~13		保留。必须保持为清除状态'0'													
位 17		<b>DPDEN:</b> 低功耗模式使能 0: 禁止低功耗模式; 1: 允许低功耗模式。													
位 16		<b>RDPL2_DIS:</b> 解除读保护使能及选择字节误擦除保护 设置该位, 将触发硬件自动对选择字节做擦除, 复位后将解除读保护使能及选择字节误擦除保护, 维持低级读保护。 当硬件收到该位为 1 后, 硬件自动清除此位。													
位 15~13		保留。必须保持为清除状态'0'													
位 12		<b>PRCDNIE:</b> 允许操作完成中断 该位允许在 <b>FLASH_STS</b> 寄存器中的 <b>PRCDN</b> 位变为'1'时产生中断。 0: 禁止产生中断; 1: 允许产生中断。													
位 11, 8, 3		保留。必须保持为清除状态'0'													
位 10		<b>ERRIE:</b> 允许错误状态中断 该位允许在发生 <b>FPEC</b> 错误时产生中断( <b>FLASH_STS</b> 寄存器中的 <b>PRGMFLR/WRPRTFLR</b> 置为'1'时)。 0: 禁止产生中断; 1: 允许产生中断。													
位 9		<b>UOBWE:</b> 允许写选择字节 当该位为'1'时, 允许对选择字节进行编程操作。当在 <b>FLASH_OPTKEYR</b> 寄存器写入正确的键序列后, 该位被置为'1'。 软件可清除此位。													
位 7		<b>LOCK:</b> 锁 只能写'1'。当该位为'1'时表示 <b>FPEC</b> 和 <b>FLASH_CTRL</b> 被锁住。在检测到正确的解锁序列后, 硬件清除此位为'0'。在一次不成功的解锁操作后, 下次系统复位前, 该位不能再被改变。													
位 6		<b>RSTR:</b> 开始 当该位为'1'时将触发一次擦除操作。该位只可由软件置为'1'并在 <b>BSY</b> 变为'0'时清为'0'。													
位 5		<b>UOBERS:</b> 擦除选择字节 擦除选择字节。													
位 4		<b>UOBPRGM:</b> 烧写选择字节 对选择字节编程。													

位 2	<b>CHPERS:</b> 全擦除 选择擦除所有用户页。
位 1	<b>PGERS:</b> 页擦除 选择擦除页。
位 0	<b>PRGM:</b> 编程 选择编程操作。

#### 4.4.6 闪存地址寄存器 (FLASH\_ADDR)

专用于闪存主区块以及当系统存储器区域作为主存的扩展。

地址偏移: 0x14

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TA[31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA[15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

这些位由硬件修改为当前/最后使用的地址。在页擦除操作中，软件必须修改这个寄存器以指定要擦除的页。

位 31~0	<b>TA:</b> 闪存地址 当进行编程时选择要编程的地址，当进行页擦除时选择要擦除的页。 注意：当 <b>FLASH_STS</b> 中的 <b>BSY</b> 位为'1'时，不能写这个寄存器
--------	--

#### 4.4.7 选择字节寄存器 (FLASH\_UOB)

地址偏移: 0x1C

复位值: 0x03FF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留				RDPRT_HL	USER_D1				USER_D0[7:6]						
res				R					R				R		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USER_D0[5:0]				未使用				nSTDBY_RST	nSTP_RST	KSEL_WDG	RDPR_TEN	UOB_FLR			
R				res				R	R	R	R	R	R	R	R

位 31~27	保留。必须保持为清除状态'0'。
位 26	<b>RDPRT_HL:</b> 读保护高位 读保护等级使用 {位 26, 位 1}联合判断。 0: 未启动读保护，且 RDP 值=0xA5 1: 启动低级读保护，且 RDP 值非 0xCC 以及 0xA5 2: 保留 3: 启动 RDP 使能及选择字节误擦除保护，且 RDP 值=0xCC
位 25~18	<b>USER_D1:</b> 用户数据 1
位 17~10	<b>USER_D0:</b> 用户数据 0

位 9~2	<b>USR:</b> 用户选择字节 这里包含 OBL 加载的用户选择字节 位[9: 5]: 未用 位 4: <b>nSTDBY_RST</b> 位 3: <b>nSTP_RST</b> 位 2: <b>SYSCLKSEL_WDG</b>
位 1	<b>RDPRTEN:</b> 读保护当设置为 1, 表示闪存存储器的读保护有效。 注意: 该位为只读。
位 0	<b>UOBFLR:</b> 选择字节错误 当该位为'1'时表示选择字节和它的反码不匹配。 注意: 该位为只读。

#### 4.4.8 写保护寄存器 (FLASH\_WRPRT)

地址偏移: 0x20

复位值: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRPRTBMP [31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRPRTBMP [15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
位 31~0		<b>WRPRTBMP:</b> 写保护 该寄存器包含由 OBL 加载的写保护选择字节。 0: 写保护生效 1: 写保护失效 注意: 这些位为只读。													

#### 4.4.9 闪存安全库区状态寄存器0 (FLASH\_CDR0)

专用于闪存安全库区。

地址偏移: 0x74

复位值: 0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16			
保留								SYS_MEM_SLIB_INSTR_PG										
RES															R			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
保留								FSLIB_EN	SYS_MEM_SLIB_EN	保留		SYS_MEM_BOOT_DIS						
RES								R	R	RES		R						
位 31~24		保留。必须保持为清除状态'0'																
位 23~16		<b>SYS_MEM_SLIB_INSTR_PG:</b> 主存扩展区域安全库区指令起始页地址 0x0: 页 0 0x1: 页 1 0x2: 页 2 0x3: 页 3 0xFF: 无指令安全区 其余设定值: 无效																

	<p><b>注意：</b>  <b>SYS_MEM_SLIB_INST_PG</b> 为 0~3，表示主存扩展区从页 0 至页 3 设定为安全库区，其中从页 0 至 <b>INSTR_PG</b>-1，设定为 唯读区，从 <b>INSTR_PG</b> 至页 3 设定为指令库区；  <b>SYS_MEM_SLIB_INSTR_PG</b> 为 0xFF，表示主存扩展区从页 0 至页 3 设定为安全库区，并且整个安全库区作为唯读区；</p>
位 15~4	保留。必须保持为清除状态'0'
位 3	<p><b>FSLIB_EN:</b> 允许存放安全库代码功能  该位设定时，表示闪存主存区域部分或是全部（依照 <b>FSLIB_CDR1</b> 设定），或是主存扩展区作为安全库代码。  0: 闪存存放安全库代码功能失效；  1: 闪存存放安全库代码功能生效。</p>
位 2	<p><b>SYS_MEM_SLIB_EN:</b> 允许主存扩展区作为存放安全库代码功能  该位设定时，表示闪存主存扩展区域作为安全库代码。  0: 主存扩展区域存放安全库代码功能失效；  1: 主存扩展区域存放安全库代码功能生效。</p>
位 1	保留。必须保持为清除状态'0'
位 0	<p><b>SYS_MEM_BOOT_DIS:</b> 禁止系统存储器区域作为存放启动代码功能  该位设定时，表示禁止系统存储器区域作为启动代码功能。  0: 系统存储器区域作为存放启动代码功能；  1: 系统存储器区域作为存放应用代码功能。</p>

#### 4.4.10 闪存安全库区状态寄存器1 (FLASH\_CDR1)

专用于闪存安全库区。

地址偏移：0x78

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLK_SLIB_END_PG[9:0]										BLK_SLIB_INSTR_PG[10:5]					
R											R				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BLK_SLIB_INSTR_PG[4:0]					BLK_SLIB_ST_PG[10:0]										
R															

位 31~22	<p><b>BLK_SLIB_END_PG:</b> 主存安全库区结束页号  0: 页0  1: 页1  2: 页2  ...  63: 页 63  对于 32KB 闪存，有效页号 0~31；  对于 16KB 闪存，有效页号 0~15；</p>
---------	---

位 21~11	<p><b>BLK_SLIB_INSTR_PG:</b> 主存指令安全库区起始页号            0: 页0            1: 页1            2: 页2            ...            63: 页63            0x7FF: 无指令安全库区            对于 32KB 闪存, 有效页号 0 – 31 以及 0x7FF;            对于 16KB 闪存, 有效页号 0 – 15 以及 0x7FF;</p> <p><b>注意:</b>  <b>BLK_SLIB_INSTR_PG</b> 为 ST_PG, 表示主存从 ST_PG 至 END_PG 设定为安全库区, 并且整个安全库区作为指令库区;  <b>BLK_SLIB_INSTR_PG</b> 为 1 – 63, 且不等于 ST_PG, 表示主存从 ST_PG 至 END_PG 设定为安全库区, 其中从 ST_PG 至 INSTR_PG -1, 设定为 唯读区, 从 INSTR_PG 至 END_PG 设定为指令库区;  <b>BLK_SLIB_INSTR_PG</b> 为 0x7FF, 表示主存从 ST_PG 至 END_PG 设定为安全库区, 并且整个安全库区作为唯读区;</p>
位 10~0	<p><b>BLK_SLIB_ST_PG:</b> 主存安全库区起始页号            0: 页0            1: 页1            2: 页2            ...            63: 页 63            对于 32KB 闪存, 有效页号 0 - 31;            对于 16KB 闪存, 有效页号 0 – 15;</p>

#### 4.4.11 闪存安全库区密码寄存器 (**FSLIB\_PSW**)

专用于闪存安全库区。

地址偏移: 0x7C

复位值: 0xFFFF FFFF

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	FSLIB_PSW[31: 16]
W W W W W W W W W W W W W W W W	
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	FSLIB_PSW[15: 0]
W W W W W W W W W W W W W W W W	
位 31~0	<p><b>FSLIB_PSW:</b> 安全库区密码寄存器            设定正确的安全库区密码, 将解除安全库区功能。            注意: 读该寄存器, 将返回全 1</p>

#### 4.4.12 闪存安全库区密码设定状态寄存器 (**FSLIB\_PSW\_STS**)

专用于闪存安全库区。

地址偏移: 0x80

复位值: 0x0000 0000

保留
----

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										SLIB_UNLOCK	SLIB_PSW_OK	SLIB_PSW_ERR			
res															
位 31~3	保留。必须保持为清除状态'0'														
位 2	<b>SLIB_UNLOCK:</b> SLIB 解锁状态 0:不能设定 FSLIB_SET_PSW/FSLIB_SET_RANGE/SYS_MEM_SLIB_SET/ SYS_MEM_BOOT_DIS_SET 1:可以设定 FSLIB_SET_PSW/FSLIB_SET_RANGE/SYS_MEM_SLIB_SET/ SYS_MEM_BOOT_DIS_SET														
位 1	<b>SLIB_PSW_OK:</b> 密码正确 当密码正确，并且设定的密码不等于 0xFFFF FFFF，该位被设置为'1'。														
位 0	<b>SLIB_PSW_ERR:</b> 密码错误 当密码错误，并且设定的密码不等于 0xFFFF FFFF，该位被设置为'1'。 注意：当该位置 1 后，硬件将不再接受重新设定密码寄存器，直到再次复位。														

#### 4.4.13 闪存CRC校验起始位置 (FLASH\_CRC\_AR)

专用于闪存主存以及主存扩展区域。

地址偏移: 0x84

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
FCRC_AR[31: 16]																
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
FCRC_AR[15: 0]																
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	
位 31~0		<b>FCRC_AR:</b> 用户代码或安全库代码 CRC 校验起始地址 用于主存或主存扩展区域中用户代码或安全库代码的 CRC 校验。 注意：必须与页起始地址对齐														

#### 4.4.14 闪存CRC校验控制寄存器 (FLASH\_CRC\_CTRL)

专用于闪存主存以及主存扩展区域。

地址偏移: 0x88

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
Res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W															

CRC_NUM															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
位 31~17	保留。必须保持为清除状态'0'														
位 16	<b>CRC_CHK_TRIG:</b> 启动 CRC 校验 设置该位去启动用户代码或是安全库代码的 CRC 校验功能 硬件启动 CRC 校验后，会自动清除该位。 注意：校验数据从 FCRC_AR ~ FCRC_AR+CRC_NUM*1 页														
位 15~0	<b>CRC_NUM:</b> CRC 校验数量 设定本次 CRC 校验的数据量，单位是页														

#### 4.4.15 闪存CRC校验结果寄存器（FLASH\_CRC\_OUTR）

专用于闪存主存以及主存扩展区域。

地址偏移: 0x8C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCRC_OUT[31: 16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCRC_OUT[15: 0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

注意：所有这些位是只读的，写入无反应。

位 31~0	<b>FCRC_OUT:</b> CRC 校验结果
--------	---------------------------

#### 4.4.16 闪存安全库区密码设定寄存器（FSLIB\_SET\_PSW）

专用于安全库区。

地址偏移: 0x160

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSLIB_SET_PSW[31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSLIB_SET_PSW[15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

注意：所有这些位是只写的，读出时返回 0。

位 31~0	<b>FSLIB_SET_PSW:</b> 安全库区密码设定
--------	--------------------------------

#### 4.4.17 闪存安全库区范围设定寄存器（**FSLIB\_SET\_RANGE**）

专用于安全库区。

地址偏移: 0x164

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSLIB_SET_END_PAGE[9:0]										FSLIB_SET_INST_START_PAGE[10:5]					
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSLIB_SET_INST_START_PAGE[4:0]										FSLIB_SET_START_PAGE[10:0]					
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

注意：所有这些位是只写的，读出时返回 0。

位 31~22	<b>FSLIB_SET_END_PAGE</b> : 主存安全库区结束页号
位 21~11	<b>FSLIB_SET_INST_START_PAGE</b> : 主存指令安全库区起始页号
位 10~0	<b>FSLIB_SET_START_PAGE</b> : 主存安全库区起始页号

#### 4.4.18 闪存主存扩展区安全库区设定寄存器 (**SYS\_MEM\_SLIB\_SET**)

专用于安全库区。

地址偏移: 0x168

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留										SYS_MEM_SET_INST_START_PAGE[7:0]					
Res	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYS_MEM_AS_SLIB															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

注意：所有这些位是只写的，读出时返回 0。

位 31~24	保留
位 23~16	<b>SYS_MEM_SET_INST_START_PAGE</b> : 主存扩展区域指令安全库区首地址
位 15~0	<b>SYS_MEM_AS_SLIB</b> : 配置主存扩展区作安全库区 注意：写入 0x5AA5 并且只在系统存储器区域作为主存扩展区的条件下有效

#### 4.4.19 闪存主存扩展区模式寄存器

### (SYS\_MEM\_BOOT\_DIS\_SET)

专用于主存扩展区。

地址偏移: 0x16C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								Res							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								BOOT_DIS_SET[7:0]							
Res								W	W	W	W	W	W	W	W

注意：所有这些位是只写的，读出时返回 0。

位 31~8	保留
位 7~0	<b>BOOT_DIS_SET:</b> 配置系统存储区域作为存放启动代码功能 0xFF: 系统存储区作为存放启动代码功能 其他值: 系统存储区域作为存放应用代码功能, 即作为主存扩展区使用 注意: 更改系统存储区域作为主存扩展区的角色需要在闪存未启动读报护的情况下进行。

### 4.4.20 闪存安全库区键寄存器 (FSLIB\_KEYR)

专用于安全库区。

地址偏移: 0x170

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSLIB_KEYR[31: 16]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FSLIB_KEYR[15: 0]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

注意：所有这些位是只写的，读出时返回 0。

位 31~0	<b>FSLIB_KEYR:</b> 安全库区键 这些位用于输入安全库区的解锁键。
--------	--

## 5 CRC计算单元（CRC）

### 5.1 CRC简介

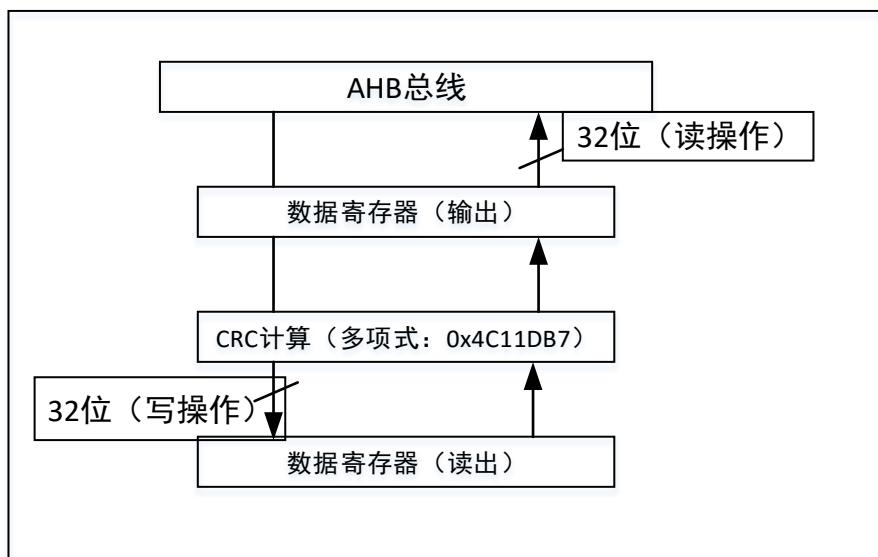
循环冗余校验（CRC）计算单元是根据固定的生成多项式得到任一32位全字的CRC计算结果。在其他的应用中，CRC技术主要应用于核实数据传输或者数据存储的正确性和完整性。标准EN/IEC 60335-1即提供了一种核实闪存存储器完整性的方法。CRC计算单元可以在程序运行时计算出软件的标识，之后与在连接时生成的参考标识比较，然后存放在指定的存储器空间。

### 5.2 CRC主要特性

- 使用CRC-32（以太网）多项式：0x4C11DB7
  - $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 一个32位数据寄存器用于输入 / 输出
- CRC计算时间：4个AHB时钟周期（HCLK）
- 通用8位寄存器（可用于存放临时数据）

下图为CRC计算单元框图

图表 5-1 CRC单元框



### 5.3 CRC功能描述

CRC计算单元含有1个32位数据寄存器：

- 对该寄存器进行写操作时，作为输入寄存器，可以输入要进行CRC计算的新数据。
- 对该寄存器进行读操作时，返回上一次CRC计算的结果。每一次写入数据寄存器，其计算结果是前一次CRC计算结果和新计算结果的组合（对整个32位字进行CRC计算，而不是逐字节地计算）

在CRC计算期间会暂停CPU的写操作，因此可以对寄存器CRC\_DR进行背靠背写入或者连续地写-读操作。

可以通过设置寄存器CRC\_CTRL的RESET位来重置寄存器CRC\_DR为0xFFFF FFFF。该操作不影响寄存器CRC\_IDR内的数据。

### 5.4 CRC寄存器

CRC计算单元包括2个数据寄存器和1个控制寄存器

下表列出了CRC的寄存器映像和复位值

表格 5-1 CRC计算单元寄存器映像

偏移	寄存器	31~24	23~16	15~8	7	6	5	4	3	2	1	0								
0x00	CRC_DR	数据寄存器 0xFFFF FFFF																		
	复位值																			
0x04	CRC_IDR	保留			独立的数据寄存器 0x00															
	复位值																			
0x08	CRC_CTRL	保留																		
	复位值																			

### 5.4.1 数据寄存器 (CRC\_DR)

地址偏移: 0x00

复位值: 0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR[31: 16]															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR[15: 0]															
rw															
位 31: 0	数据寄存器位 写入 CRC 计算器的新数据时，作为输入寄存器 读取时返回 CRC 计算的结果														

### 5.4.2 独立数据寄存器 (CRC\_IDR)

地址偏移: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16								
保留																							
res																							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
保留								IDR[7: 0]															
res																							
位 31: 8	保留																						
位 7: 0	通用 8 位数据寄存器位 可用于临时存放 1 字节的数据。 寄存器 CRC_CTRL 的 RESET 位产生的 CRC 复位对本寄存器没有影响																						

### 5.4.3 控制寄存器 (CRC\_CTRL)

地址偏移: 0x08

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															
res															
位 31: 1	保留														
位 0	<b>RESET 位</b> 复位 CRC 计算单元，设置数据寄存器为 0xFFFF FFFF。 只能对该位写'1'，它由硬件自动清'0'。														
rw															

# 6 通用功能I/O (GPIO)

## 6.1 简介

每个通用 I/O 口都有 6 个 32 位配置寄存器 (GPIOx\_MODER, GPIOx\_OTYPER, GPIOx\_ODRVR, GPIOx\_PUPDR, GPIOx\_HDRV 和 GPIOx\_SRCTR), 2 个 32 位数据寄存器 (GPIOx\_IPTDT 和 GPIOx\_OPTDT), 1 个 32 位置位/复位寄存器 (GPIOx\_BSRE) 和 1 个 32 位复位寄存器 (GPIOx\_BRE)。

每个通用 I/O 口还含有 1 个 32 位锁定寄存器 (GPIOx\_LOCK) 和 2 个 32 位复用功能寄存器 (GPIOx\_AFRH 和 GPIOx\_AFRL)。

## 6.2 主要特征

- 输出状态：推挽输出或开漏输出
- 从数据寄存器 (GPIOx\_OPTDT) 或外设 (复用功能输出) 输出数据
- 可选的每个 I/O 口的电流推动/吸入能力
- 输入状态：浮空、上拉/下拉、模拟输入
- 从数据寄存器 (GPIOx\_IPTDT) 或外设 (复用功能输入) 输入数据
- 位置位/复位寄存器 (GPIOx\_BSRE) 和位复位寄存器 (GPIOx\_BRE) 为对 GPIOx\_OPTDT 寄存器提供位访问能力
- 每个 I/O 口的锁定机制 (GPIOx\_LOCK) 配置
- 模拟功能
- 可选的复用功能
- 允许 GPIO 口和外设引脚的高灵活性复用

## 6.3 功能描述

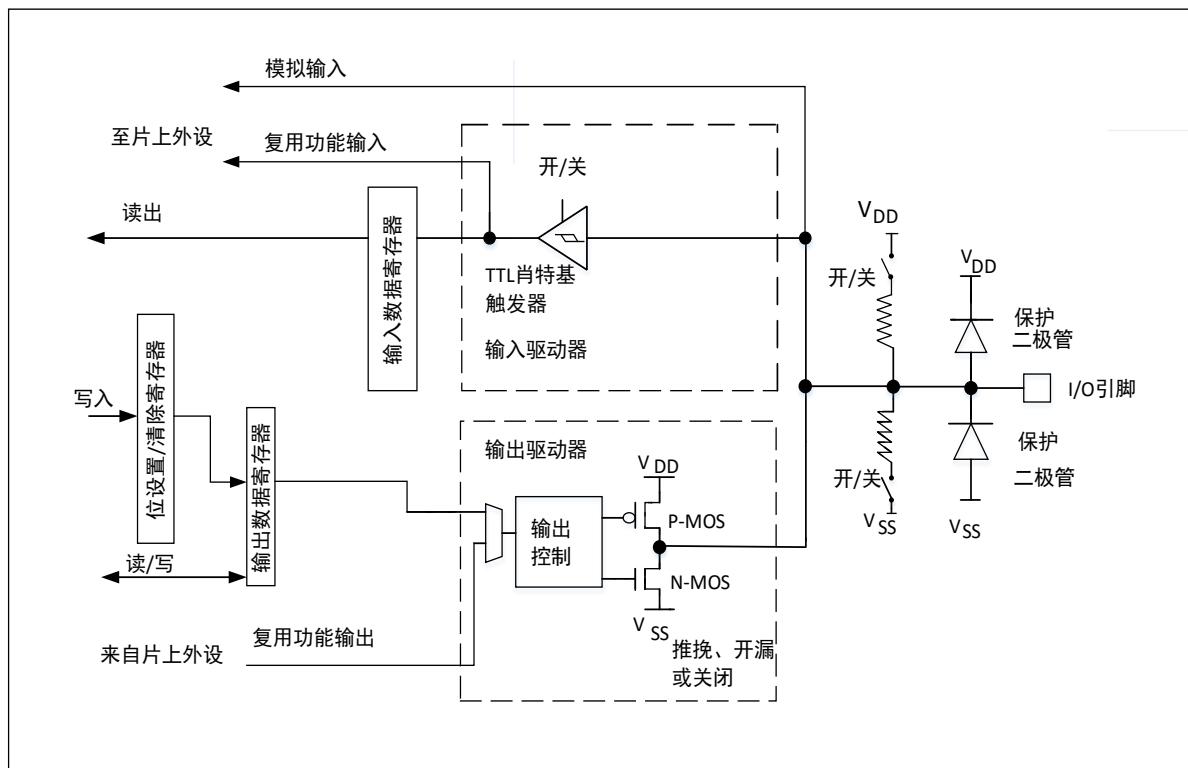
根据数据手册中列出的每个 I/O 端口的特定硬件特征，GPIO 端口的每个端口可以由软件分别配置成多种模式。

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟输入
- 开漏输出
- 推挽输出
- 复用功能的推挽输出
- 复用功能的开漏输出

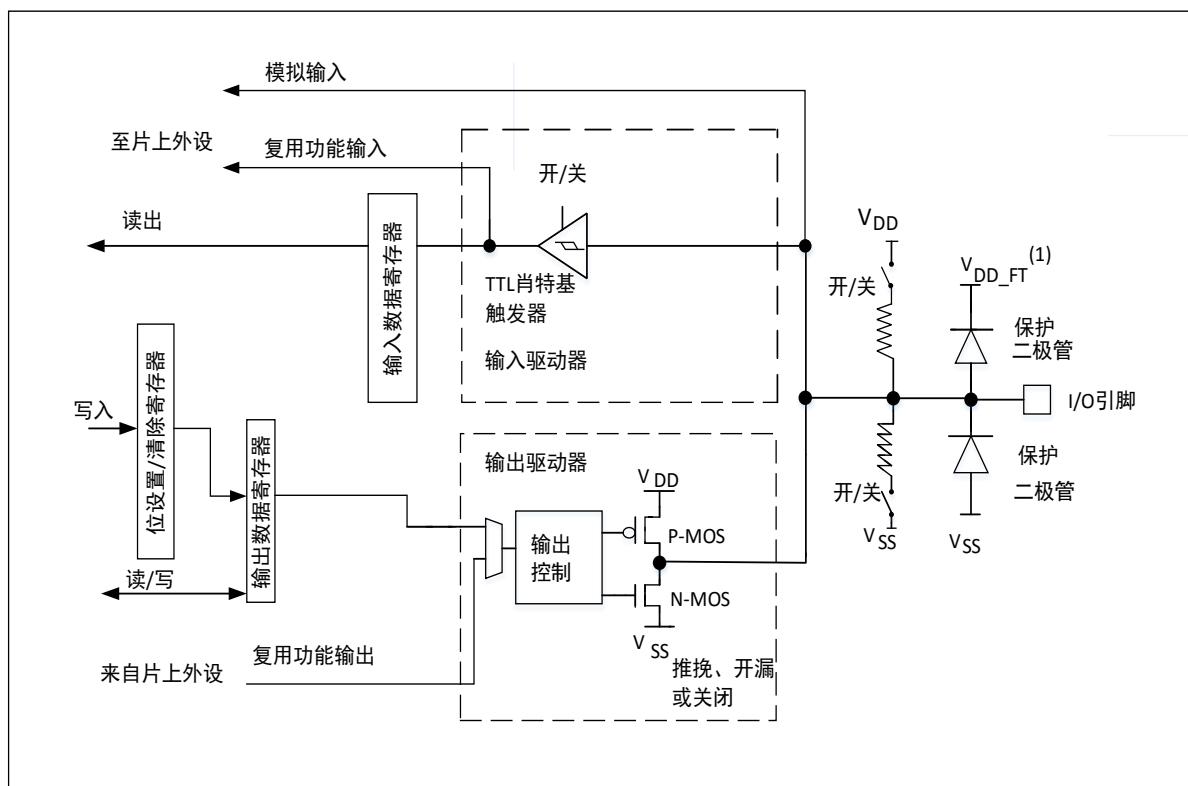
每个 I/O 端口位可以自由编程，然而 I/O 端口寄存器可以按 32 位字，半字或是字节访问。GPIOx\_BSRE 和 GPIOx\_BRE 寄存器允许对任何 GPIOx\_OPTDT 寄存器的读/更改的独立访问；这样，在读和更改访问之间产生 IRQ 时不会发生危险。

下图给出了一个 I/O 端口位的基本结构。

图表 6-1 I/O 端口位的基本结构



图表 6-2 5 伏兼容 I/O 端口位的基本结构



注意： $V_{DD\_FT}$  对 5 伏容忍 I/O 脚是特殊的，它与  $V_{DD}$  不同。

表格 6-1 端口位配置表

MODER[1: 0]	OTYPER	ODRVR[1: 0]	PUPDR[1: 0]	I/O配置	
01	0	ODRV[1: 0]	0	0	GP输出 PP
	0		0	1	GP输出 PP
	0		1	0	GP输出 PP
	0		1	1	保留
	1		0	0	GP输出 OD
	1		0	1	GP输出 OD
	1		1	0	GP输出 OD
	1		1	1	保留(GP输出 OD)
	0		0	0	AF输出 PP
	0		0	1	AF输出 PP
10	0	ODRV[1: 0]	1	0	AF输出 PP
	0		1	1	保留
	0		0	0	AF输出 OD
	1		0	1	AF输出 OD
	1		1	0	AF输出 OD
	1		1	1	保留
	0				
	0				
00	x	x	x	0	输入 浮空
	x	x	x	0	输入 PU
	x	x	x	1	输入 PD
	x	x	x	1	保留(浮空输入)
11	x	x	x	0	输入/输出 模拟
	x	x	x	0	保留
	x	x	x	1	
	x	x	x	1	

### 6.3.1 通用 I/O(GPIO)

复位期间和刚复位后，复用功能未开启，大部分 I/O 端口被配置成浮空输入模式。

当作为输出配置时，写到输出数据寄存器 (GPIOx\_OPTDT) 上的值会输出到相应的 I/O 引脚。可以以推挽模式或开漏模式（仅低电平被驱动，高电平表现为高阻）使用输出驱动器。

输入数据寄存器 (GPIOx\_IPTDT) 在每个 AHB 时钟周期捕捉 I/O 引脚上的数据。

所有 GPIO 引脚有一个内部弱上拉和弱下拉，它们被激活或断开有赖于 GPIOx\_PUPDR 寄存器的值。

### 6.3.2 I/O引脚的复用功能和重映射

器件 I/O 口线通过多路复用器连接到内嵌的外设/模块。微观上，同一时刻仅允许外设的复用功能一个引脚连接到一个 I/O 口线上。因此，同一 I/O 口线上不能有冲突的外设引脚分配。

每个 I/O 引脚有一个多达 16 个复用功能输入（从 AF0 到 AF15）的多路复用器，其可通过配置 GPIOx\_AFRL 寄存器（从引脚 0 到引脚 7）和 GPIOx\_AFRH 寄存器（从引脚 8 到引脚 15）来实现。复位后，所有的 I/O 口都连接到复用功能 0 (AF0)。

除了这种灵活的 I/O 复用结构，每个外设还有复用功能映射到不同的 I/O 引脚上，这种方法用于在小封装器件上优化更多的可用外设。

为了使用一个给定的 I/O 口配置，你必须按如下的原则执行：

- 调试功能：每个器件复位后，这些引脚立即配置为复用功能用来支持调用。
- GPIO：在GPIOx\_MODER寄存器中配置所需的I/O口为输出，输入或模拟输入。
- 外设的复用功能：
  - 连接 I/O 到所需的 AFx，AFx 定义在 GPIOx\_AFRL 或 GPIOx\_AFRH 寄存器中
  - 通过对 GPIOx\_OTYPER, GPIOx\_PUPDR 和 GPIOx\_ODRVR 寄存器来配置相应引脚的上拉/下拉和电流推动/吸入能力
  - 在 GPIOx\_MODER 寄存器中配置所需的 I/O 口为复用功能
- 附加功能：
  - 对于ADC，在GPIOx\_MODER寄存器中配置所需的I/O线为模拟方式并在ADC寄存器中配置所需的功能。
  - 对于附加功能如RTC、WKUPx和振荡器，在关联的ERTC、PWR和RCC寄存器配置相应所需的功能。

### 6.3.3 I/O端口控制寄存器

每个 GPIO 口都有 6 个 32 位的控制寄存器（GPIOx\_MODER, GPIOx\_OTYPER, GPIOx\_ODRVR, GPIOx\_PUPDR, GPIOx\_HDRV, GPIOx\_SRCTR）用来配置多达 16 根 I/O 口线。GPIOx\_MODER 寄存器用来选择 I/O 模式（如输入，输出、复用或模拟）。GPIOx\_OTYPER, GPIOx\_ODRVR, GPIOx\_HDRV 和 GPIOx\_SRCTR 寄存器用来选择输出类型（如推挽或开漏），电流推动/吸入能力和电压转换速率。GPIOx\_PUPDR 寄存器用来选择上拉/下拉方式。

### 6.3.4 I/O端口数据寄存器

每个 GPIO 口有两个 16 位数据寄存器：输入和输出数据寄存器（GPIOx\_IPTDT 和 GPIOx\_OPTDT）。GPIOx\_OPTDT 用于存储输出数据，其可进行读/写访问。从 I/O 口线的输入数据存放在(GPIOx\_IPTDT) 寄存器中，该寄存器为只读寄存器。

### 6.3.5 I/O端口数据位处理

位置位复位寄存器(GPIOx\_BSRE)是一个 32 位寄存器，其允许应用对输出数据寄存器(GPIOx\_OPTDT) 的每个位进行置位和复位操作。

对于 GPIOx\_OPTDT 中的每位，在 GPIOx\_BSRE 中有两位与之对应：BS(i) 和 BR(i)。当对位 BS(i) 写 1 时则设置相应的 OPTDT(i) 位。当对 BR(i) 写 1 时，则复位相应的 OPTDT(i) 位。

对 GPIOx\_BSRE 中的任意位写 0 都不会影响 GPIOx\_OPTDT 寄存器的值。若对 GPIOx\_BSRE 的 BS(i) 和 BR(i) 同时置 1，那么其置位操作具有优先权（即对相应位做置位操作）。

用 GPIOx\_BSRE 寄存器来改变 GPIOx\_OPTDT 的相应位，GPIOx\_OPTDT 位也可直接从这个寄存器进行访问。GPIOx\_BSRE 寄存器提供对 GPIOx\_OPTDT 寄存器原子位操作处理机制。

GPIOx\_OPTDT 用 GPIOx\_BSRE 置位或复位的访问机制不需要软件去关闭中断来访问 GPIOx\_OPTDT：在一个 AHB 写访问周期改变 1 位或多位数据是可能的。

### 6.3.6 GPIO锁定机制

用一个特定对 GPIOx\_LOCK 寄存器的写序列来冻结端口的控制寄存器是可行的。冻结的寄存器有：GPIOx\_MODER, GPIOx\_OTYPER, GPIOx\_ODRVR, GPIOx\_PUPDR, GPIOx\_AFRL, GPIOx\_AFRH, GPIOx\_HDRV 和 GPIOx\_SRCTR。

为了写 GPIOx\_LOCK 寄存器，须发出一个特定的写/读序列。当正确的锁定序列作用于这个寄存器的位 16 时，LOCK[15:0]的值用来锁定 I/O 口的配置(在写序列期间要保持 LOCK[15:0]值不变)。

当锁定序列已经作用于一个端口位，该端口位的值再也不能改变直到下一次 MCU 复位或 GPIO 复位。每个 GPIOx\_LOCK 位冻结控制寄存器中 (GPIOx\_MODER, GPIOx\_OTYPER, GPIOx\_ODRVR, GPIOx\_PUPDR, GPIOx\_AFRL, GPIOx\_AFRH, GPIOx\_HDRV 和 GPIOx\_SRCTR) 的相应位。

锁定序列只能用字(32位长)访问 GPIOx\_LOCK 寄存器, 基于 GPIOx\_LOCK 位 16 必须与[15:0]位同时设置的事实。

### 6.3.7 I/O复用功能输入/输出

选择每个端口线的有效复用功能之一是由两个寄存器来决定的。你可根据你应用的需求用这两寄存器连接复用功能模块到其他引脚。

这就表明每个 GPIO 口线可能做为多个外设的口线, 用 GPIOx\_AFRL 和 GPIOx\_AFRH 复用功能寄存器来配置这些外设口线。

表格 6-2 通过 GPIOA\_AFR 寄存器配置端口 A 的复用功能

引脚名	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PA0	-	USART2_CTS	-	-	I2C2_SCL	TMR1_ETR	-	COMP_OUT
PA1	EVENTOUT	USART2 RTS	-	-	I2C2_SDA	TMR15_CH1N	-	-
PA2	TMR15_CH1	USART2_TX	-	-	-	-	-	-
PA3	TMR15_CH2	USART2_RX	-	-	-	I2S2_MCK	-	-
PA4	SPI1_NSS/I2S1_WS	USART2_CK	-	-	TMR14_CH1	-	-	-
PA5	SPI1_SCK/I2S1_CK	-	-	-	-	-	-	-
PA6	SPI1_MISO/I2S1_MCK	TMR3_CH1	TMR1_BKIN	I2S2_MCK	-	TMR16_CH1	EVENTOUT	COMP_OUT
PA7	SPI1_MOSI/I2S1_SD	TMR3_CH2	TMR1_CH1N	-	TMR14_CH1	TMR17_CH1	EVENTOUT	-
PA8	CLKOUT	USART1_CK	TMR1_CH1	EVENTOUT	USART2_TX	-	-	I2C2_SCL
PA9	TMR15_BKIN	USART1_TX	TMR1_CH2	-	I2C1_SCL	CLKOUT	-	I2C2_SMBA
PA10	TMR17_BKIN	USART1_RX	TMR1_CH3	-	I2C1_SDA	-	-	-
PA11	EVENTOUT	USART1_CTS	TMR1_CH4	-	I2C1_SMBA	I2C2_SCL	-	COMP_OUT
PA12	EVENTOUT	USART1_RTS	TMR1_ETR	-	-	I2C2_SDA	-	-
PA13	SWDIO	IR_OUT	-	-	-	-	SPI2_MISO/I2S2_MCK	-
PA14	SWCLK	USART2_TX	-	-	-	-	SPI2_MOSI/I2S2_SD	-
PA15	SPI1_NSS/I2S1_WS	USART2_RX	-	EVENTOUT	-	-	SPI2_NSS/I2S2_WS	-

表格 6-3 通过 GPIOB\_AFR 寄存器配置端口 B 的复用功能

引脚名	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
PB0	EVENTOUT	TMR3_CH3	TMR1_CH2N	USART2_RX	-	-	I2S1_MCK	-
PB1	TMR14_CH1	TMR3_CH4	TMR1_CH3N	-	-	-	SPI2_SCK/I2S2_CK	-
PB2	-	-	TMR3_ETR	-	-	-	-	-
PB3	SPI1_SCK/I2S1_CK	EVENTOUT	-	-	-	-	SPI2_SCK/I2S2_CK	-
PB4	SPI1_MISO/I2S1_MCK	TMR3_CH1	EVENTOUT	-	-	TMR17_BKIN	SPI2_MISO/I2S2_MCK	I2C2_SDA
PB5	SPI1_MOSI/I2S1_SD	TMR3_CH2	TMR16_BKIN	I2C1_SMBA	-	-	SPI2_MOSI/I2S2_SD	-
PB6	USART1_TX	I2C1_SCL	TMR16_CH1N	-	-	-	I2S1_MCK	-
PB7	USART1_RX	I2C1_SDA	TMR17_CH1N	-	-	-	-	-
PB8	-	I2C1_SCL	TMR16_CH1	-	-	-	-	-
PB9	IR_OUT	I2C1_SDA	TMR17_CH1	EVENTOUT	-	I2S1_MCK	-	SPI2 NSS/I2S2_WS
PB10	-	I2C2_SCL	-	-	-	-	-	SPI2_SCK/I2S2_CK
PB11	EVENTOUT	I2C2_SDA	-	-	-	-	-	-
PB12	SPI2_NSS/I2S2_WS	EVENTOUT	TMR1_BKIN	-	-	TMR15_BKIN	-	I2C2_SMBA
PB13	SPI2_SCK/I2S2_CK	-	TMR1_CH1N	-	-	I2C2_SCL	-	-
PB14	SPI2_MISO/I2S2_MCK	TMR15_CH1	TMR1_CH2N	-	-	I2C2_SDA	-	-
PB15	SPI2_MOSI/I2S2_SD	TMR15_CH2	TMR1_CH3N	TMR15_CH1N	-	-	-	-

表格 6-4 通过GPIOF\_AFR寄存器配置端口F的复用功能

引脚名	AF0	AF1
PF0	-	I2C1_SDA
PF1	-	I2C1_SCL
PF6	I2C2_SCL	-
PF7	I2C2_SDA	-

### 6.3.8 外部中断/唤醒线

所有端口都有外部中断能力。为了使用外部中断线，端口必须配置成输入模式。更多的关于外部中断的信息，参考 [8.2 节外部中断/事件控制器（EXTI）](#)。

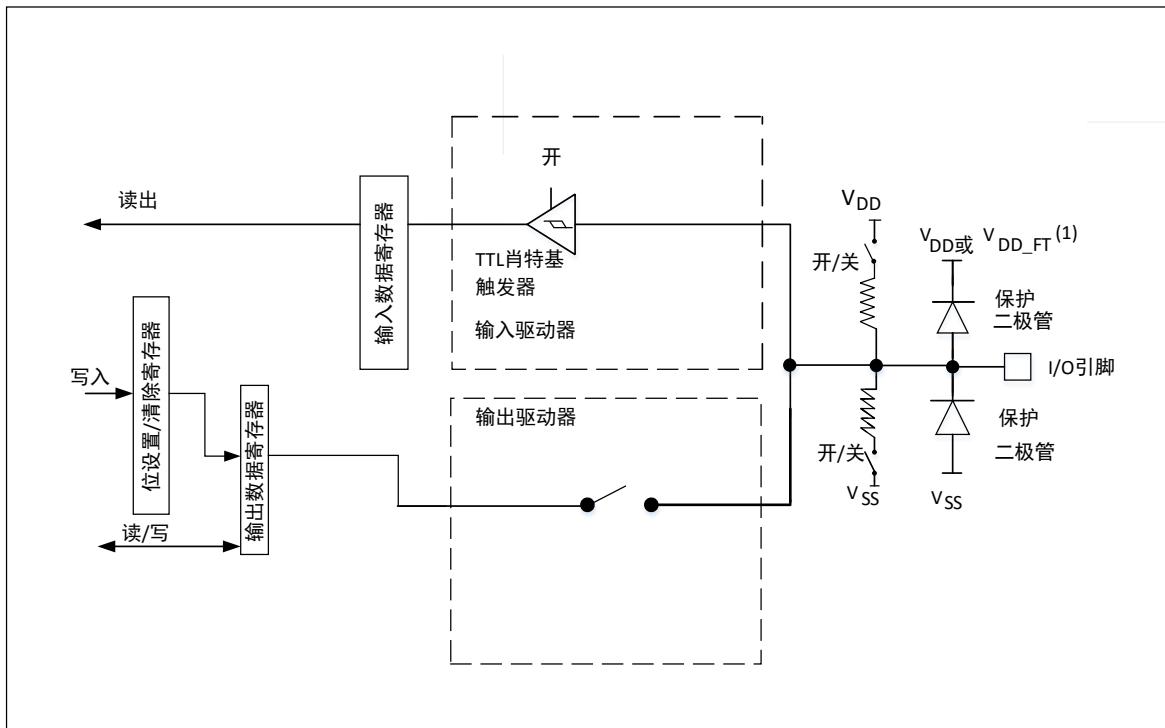
### 6.3.9 输入配置

当 I/O 端口配置为输入时：

- 输出缓冲器被禁止
- 施密特触发输入被激活
- 弱上拉和下拉电阻是否激活取决于 GPIOx\_PUPDR 寄存器的值
- 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可得到 I/O 状态

下图给出了 I/O 端口位的输入配置

图表 6-3 输入浮空/上拉/下拉配置



注意： $V_{DD\_FT}$  对 5 伏容忍 I/O 脚是特殊的，它与  $V_{DD}$  不同

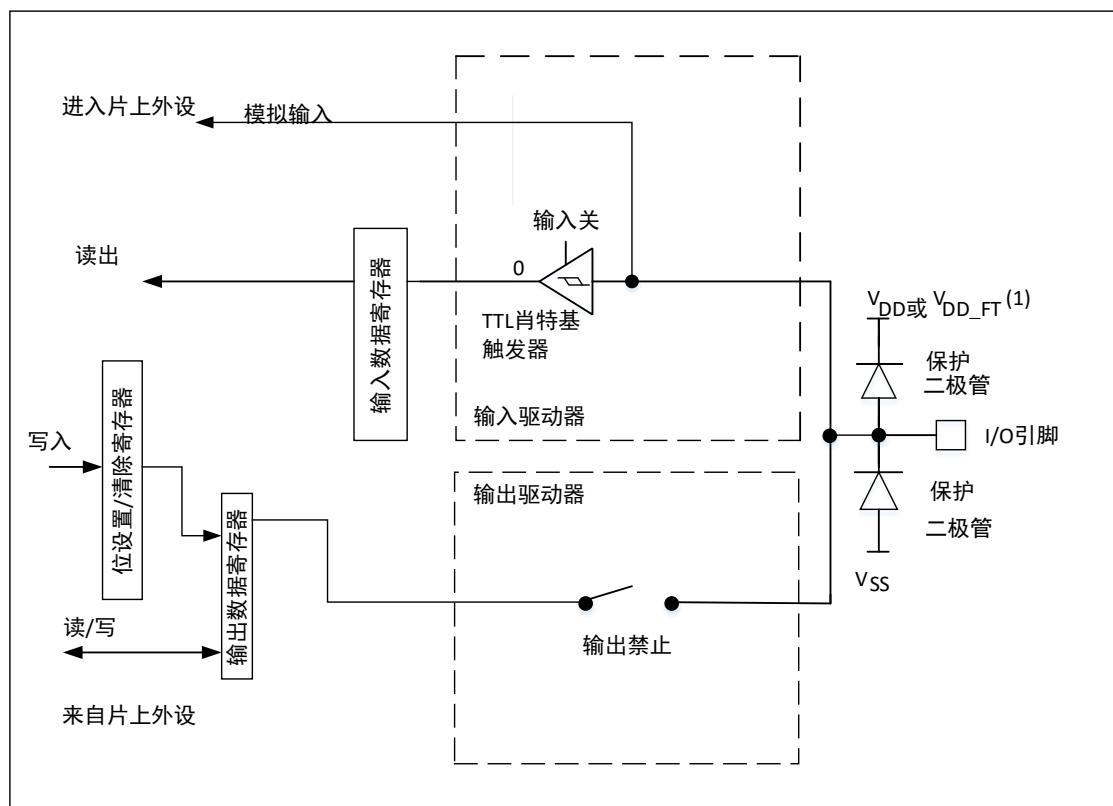
### 6.3.10 模拟输入配置

当 I/O 端口被配置为模拟输入配置时：

- 输出缓冲器被禁止
- 禁止施密特触发输入，实现了每个模拟 I/O 引脚上的零消耗。施密特触发输出值被强置为‘0’
- 弱上拉和下拉电阻被禁止
- 读取输入数据寄存器时数值为‘0’

下图给出了 I/O 端口位的高阻抗模拟输入配置：

图表 6-4 高阻抗的模拟输入配置



注意:  $V_{DD\_FT}$  对 5 伏兼容 I/O 脚是特殊的, 它与  $V_{DD}$  不同

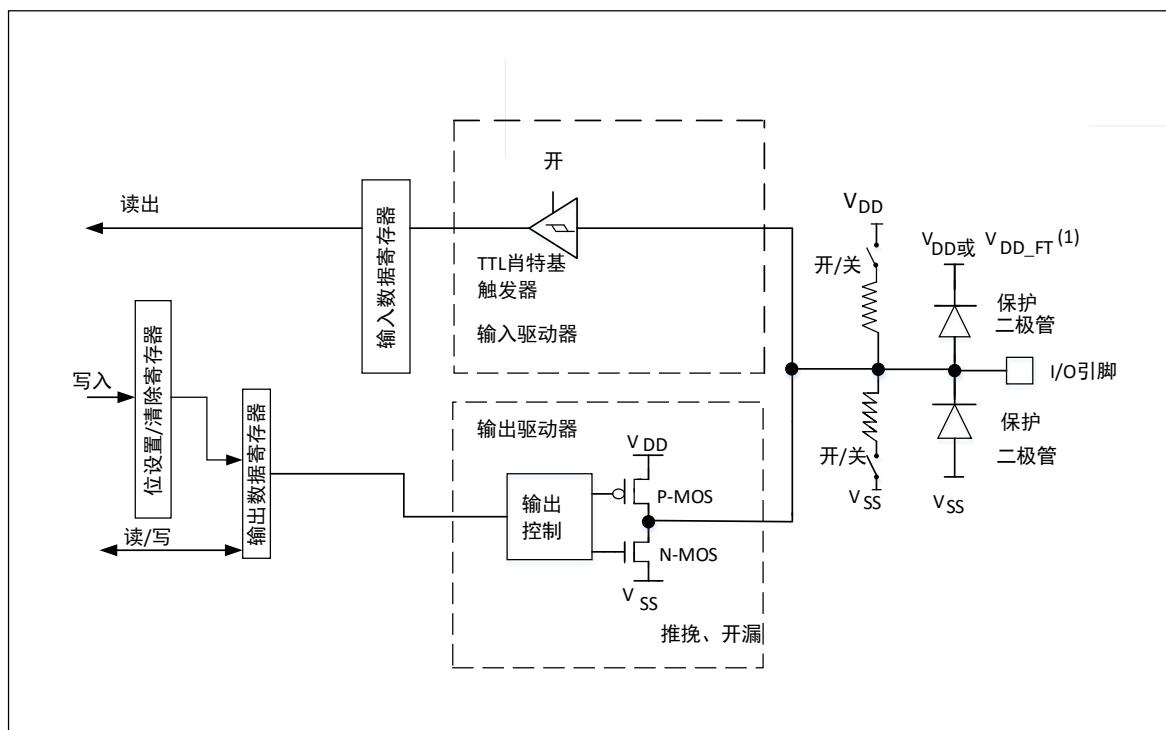
### 6.3.11 输出配置

当 I/O 端口被配置为输出时:

- 输出缓冲器被激活
  - 开漏模式: 输出寄存器上的 '0' 激活N-MOS, 而输出寄存器上的 '1' 将端口置于高阻状态 (P-MOS从不被激活)
  - 推挽模式: 输出寄存器上的 '0' 激活N-MOS, 而输出寄存器上的 '1' 将激活P-MOS
- 施密特触发输入被激活
- 弱上拉和下拉电阻被禁止
- 出现在I/O脚上的数据在每个AHB时钟被采样到输入数据寄存器
- 对输入数据寄存器的读访问可得到I/O状态
- 对输出数据寄存器的读访问得到最后一次写的值

下图给出了 I/O 端口位的输出配置。

图表 6-5 输出配置



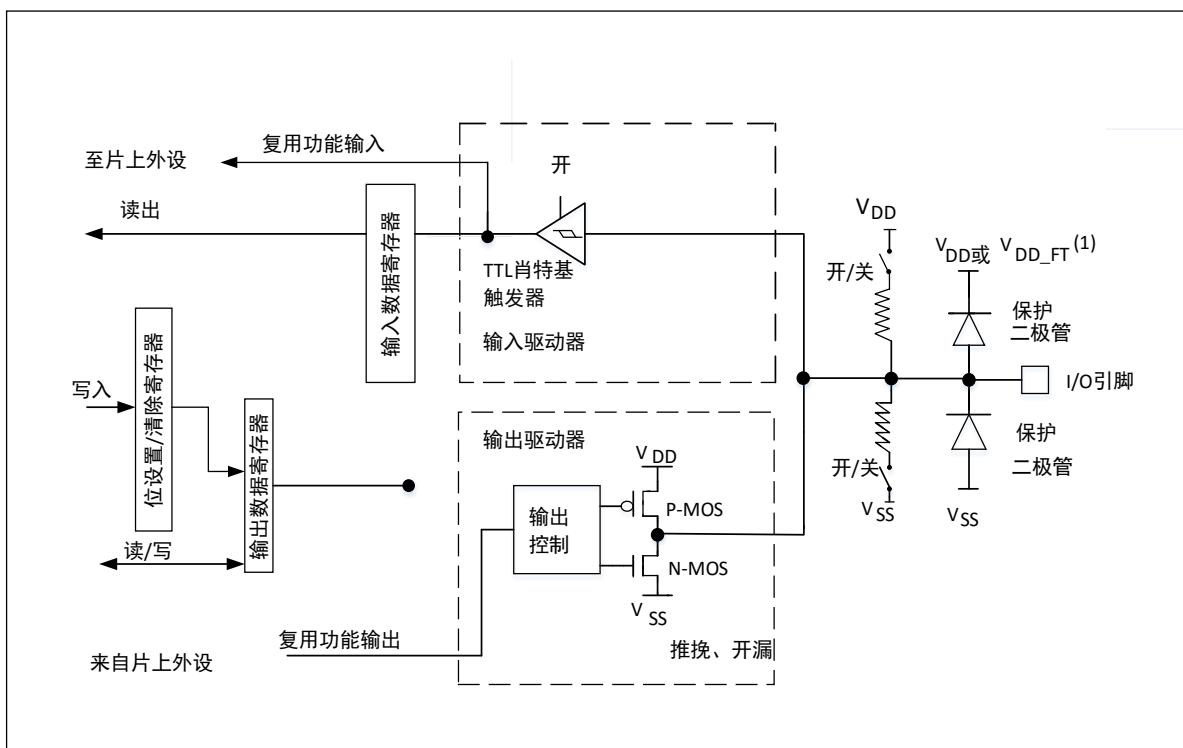
注意:  $V_{DD\_FT}$  对 5 伏兼容 I/O 脚是特殊的, 它与  $V_{DD}$  不同

### 6.3.12 复用功能 (AF)

当 I/O 端口被配置为复用功能时:

- 输出缓冲器可被配置为开漏或推挽模式
- 外设的信号驱动输出缓冲器 (复用功能输出)
- 施密特触发输入被激活
- 弱上拉和下拉电阻被禁止
- 在每个AHB时钟周期, 出现在I/O脚上的数据被采样到输入数据寄存器
- 读输入数据寄存器时可得到I/O口状态

图表 6-6 复用功能配置



注意:  $V_{DD\_FT}$  对 5 伏兼容 I/O 脚是特殊的, 它与  $V_{DD}$  不同

### 6.3.13 把OSC32\_IN/OSC32\_OUT作为GPIO端口PC14/PC15

当 LSE 振荡器关闭时, LSE 振荡器引脚 OSC32\_IN/OSC32\_OUT 可以分别用做 GPIO 的 PC14/PC15, LSE 功能始终优先于通用 I/O 口的功能。

- 注意: 1. 关闭 1.2V 电压区(入待机模式)时, 不能使用 PC14/PC15 的 GPIO 口功能;  
2. 见 [第 2.3.1.2 节](#) 有关 I/O 口使用的限制

### 6.3.14 把OSC\_IN/OSC\_OUT作为GPIO端口PF0/PF1

当 HSE 振荡器关闭时, HSE 振荡器引脚 OSC\_IN/OSC\_OUT 可以用做 GPIO 的 PF0/PF1, HSE 功能始终优先于 I/O 口的功能。

### 6.3.15 备份域供电下 GPIO 引脚的使用

当 1.2V 区域断电(当器件进入待机模式)时 GPIO 功能失去。在这种情况下, 若 PC13/PC14/PC15 配置为不被 ERTC 配置旁路, 这些引脚被设为模拟输入模式。

## 6.4 GPIO 寄存器

下面列出了 GPIO 寄存器映象和复位数值。必须以字(32 位)的方式操作这些外设寄存器。

表格 6-5 GPIO 寄存器地址映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
000h	GPIOA_MODER	MDE15 [1:0]	MDE14 [1:0]	MDE13 [1:0]	MDE12 [1:0]	MDE11 [1:0]	MDE10 [1:0]	MDE9[1:0]	MDE8[1:0]	MDE7[1:0]	MDE6[1:0]	MDE5[1:0]	MDE4[1:0]	MDE3[1:0]	MDE2[1:0]	MDE1[1:0]	MDE0[1:0]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		复位值	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

000h	GPIOx_MOD ER(x = B,C,F)	MDE15 [1:0]
	复位值	0 0 0 0 0 0
004h	GPIOx_OTY PER	保留
	复位值	
008h	GPIOx_ODR VR	ODRV15[1:0] ODRV14[1:0] ODRV13[1:0] ODRV12[1:0] ODRV11[1:0] ODRV10[1:0] ODRV9[1:0] ODRV8[1:0] ODRV7[1:0] ODRV6[1:0] ODRV5[1:0] ODRV4[1:0] ODRV3[1:0] ODRV2[1:0] ODRV1[1:0] ODRV0[1:0]
	复位值	
00Ch	GPIOA_PUP DR	PUPD15[1:0] PUPD14[1:0] PUPD13[1:0] PUPD12[1:0] PUPD11[1:0] PUPD10[1:0] PUPD9[1:0] PUPD8[1:0] PUPD7[1:0] PUPD6[1:0] PUPD5[1:0] PUPD4[1:0] PUPD3[1:0] PUPD2[1:0] PUPD1[1:0] PUPD0[1:0]
	复位值	
00Ch	GPIOx_PUP DR(x = B,C,F)	保留
	复位值	
010h	GPIOx_IPTD T	IPTDT[15: 0]
	复位值	
014h	GPIOx_OPT DT	OPTDT[15: 0]
	复位值	
018h	GPIOx_BSR E	BRE[15: 0]
	复位值	
01Ch	GPIOx_LOC K	保留
	复位值	LOCK[15: 0]
0x20	GPIOx_AFR L	AFSEL7[3:0] AFSEL6[3:0] AFSEL14[3:0] AFSEL13[3:0] AFSEL5[3:0] AFSEL4[3:0] AFSEL12[3:0] AFSEL11[3:0] AFSEL10[3:0] AFSEL9[3:0] AFSEL8[3:0]
	复位值	
0x24	GPIOx_AFR H	AFSEL15[3:0] AFSEL10[3:0] AFSEL9[3:0] AFSEL8[3:0]

#### 6.4.1 端口配置低寄存器（GPIOx\_MODER）（ $x=A..C,F$ ）

偏移地址： 0x00

复位值: 0x28000000 端口 A

0x00000000 其它端口

位 2v+1:2v

**MDEy[1:0]:** 端口 x 的模式位 (y=0...15)

软件通过这些位配置相应的 I/O 端口。

00: 输入模式（复位状态）

## 01：通用输出模式

## 10: 复用功能

#### 6.4.2 端口输出类型寄存器 (GPIOx\_OTYPER) (x=A..C,F)

偏移地址: 0x04

复位值: 0x00000000

### 6.4.3 电流推动/吸入能力切换控制寄存器 (**GPIOx\_ODRVR**) ( $x=A..C,F$ )

偏移地址: 0x08

复位值: 0x0C000000 端口 A

0x00000000 其它端口

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ODRV15 [1:0]		ODRV14 [1:0]		ODRV13 [1:0]		ODRV12 [1:0]		ODRV11 [1:0]		ODRV10 [1:0]		ODRV9 [1:0]		ODRV8 [1:0]	
rw	rw	rw	rw	rw	rw										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODRV7 [1:0]		ODRV6 [1:0]		ODRV5 [1:0]		ODRV4 [1:0]		ODRV3 [1:0]		ODRV2 [1:0]		ODRV1 [1:0]		ODRV0 [1:0]	
rw	rw	rw	rw	rw	rw										

位 2y+1:2y

**ODRVy[1:0]:** 端口 x 的模式位 (y=0...15)

软件通过这些位配置相应的 I/O 端口电流能力。

x0: 适中电流推动/吸入能力 (对应于原来的输出模式: 低速)

01: 较大电流推动/吸入能力 (对应于原来的输出模式: 中速)

11: 适中电流推动/吸入能力 (对应于原来的输出模式: 低速)

#### 6.4.4 端口上拉/下拉寄存器 (**GPIOx\_PUPDR**) (x=A..C,F)

偏移地址: 0x0C

复位值: 0x24000000 端口 A

0x00000000 其它端口

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPD15 [1:0]		PUPD14 [1:0]		PUPD13 [1:0]		PUPD12 [1:0]		PUPD11 [1:0]		PUPD10 [1:0]		PUPD9 [1:0]		PUPD8 [1:0]	
rw	rw	rw	rw	rw	rw										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPD7 [1:0]		PUPD6 [1:0]		PUPD5 [1:0]		PUPD4 [1:0]		PUPD3 [1:0]		PUPD2 [1:0]		PUPD1 [1:0]		PUPD0 [1:0]	
rw	rw	rw	rw	rw	rw										

位 2y+1:2y

**PUPDy[1:0]:** 端口 x 的模式位 (y=0...15)

软件通过这些位配置相应的 I/O 上拉或下拉。

00: 无上拉和下拉

01: 上拉

10: 下拉

11: 无上拉和下拉

注: 此寄存器只用在输入模式

#### 6.4.5 端口输入数据寄存器 (**GPIOx\_IPTDT**) (x=A..C,F)

地址偏移: 0x10

复位值: 0x0000XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPD T[1] [5]	IPD T[1] [4]	IPD T[1] [3]	IPD T[1] [2]	IPD T[1] [1]	IPTD T[10]	IPTD T[9]	IPTD T[8]	IPTD T[7]	IPTD T[6]	IPTD T[5]	IPTD T[4]	IPTD T[3]	IPTD T[2]	IPTD T[1]	IPTD T[0]
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位 31:16 保留, 始终读为 0。

位 15:0	<b>IPTDTy[15:0]:</b> 端口输入数据 ( $y=0\ldots15$ ) 这些位为只读。读出的值为对应 I/O 口的状态。
--------	---

#### 6.4.6 端口输出数据寄存器 (GPIOx\_OPTDT) ( $x=A..C,F$ )

地址偏移: 0x14

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPT DT[15]	OPT DT[14]	OPT DT[13]	OPT DT[12]	OPT DT[11]	OPT DT[10]	OPT DT[9]	OPT DT[8]	OPT DT[7]	OPT DT[6]	OPT DT[5]	OPT DT[4]	OPT DT[3]	OPT DT[2]	OPT DT[1]	OPT DT[0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31:16		保留, 始终读为 0。													
位 15:0		<b>OPTDTy[15:0]:</b> 端口输出数据 ( $y=0\ldots15$ ) 这些位可读可写。 注: 通过写 GPIOx_BSRE ( $x=A\ldots F$ ) 或 GPIOx_BRE ( $x=A\ldots F$ ) 寄存器, 可以分别地对各个 OPTDT 位进行独立的设置/清除。													

#### 6.4.7 端口位设置/清除寄存器 (GPIOx\_BSRE) ( $x=A..C,F$ )

地址偏移: 0x18

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BRE [15]	BRE [14]	BRE [13]	BRE [12]	BRE [11]	BRE [10]	BRE [9]	BRE [8]	BRE [7]	BRE [6]	BRE [5]	BRE [4]	BRE [3]	BRE [2]	BRE [1]	BRE [0]
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BST [15]	BST [14]	BST [13]	BST [12]	BST [11]	BST [10]	BST [9]	BST [8]	BST [7]	BST [6]	BST [5]	BST [4]	BST [3]	BST [2]	BST [1]	BST [0]
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:16	<b>BREy:</b> 清除端口 x 的位 y ( $y=0\ldots15$ ) 这些位只能写入。读这些位时返回 0x0000 数值。 0: 对对应的 OPTDTy 位不产生影响 1: 清除对应的 OPTDTy 位为 0 注: 如果同时设置了 BSTy 和 BREy 的对应位, BSTy 位起作用。
位 15:0	<b>BSTy:</b> 设置端口 x 的位 y ( $y=0\ldots15$ ) 这些位只能写入。读这些位时返回 0x0000 数值。 0: 对对应的 OPTDTy 位不产生影响 1: 设置对应的 OPTDTy 位为 1

#### 6.4.8 端口配置锁定寄存器 (GPIOx\_LOCK) ( $x=A..C,F$ )

当执行正确的写序列设置了位 16(LOCKK)时, 该寄存器用来锁定端口位的配置。位[15:0]用于锁定 GPIO 端口的配置。在规定的写入操作期间, 不能改变 LOCK[15:0]。当对相应的端口位执行了 LOCK 序列后, 在下次系统复位或 GPIO 复位之前将不能再更改端口位的配置。

注: 一个特定的写序列用于写 GPIOx\_LOCK 寄存器。在这个锁序列期间, 仅能字访问寄存器。

每个锁定位冻结一个指定的配置寄存器 (控制和复用功能寄存器)。

地址偏移: 0x1C

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															LOC KK
res															RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOC K[15]	LOC K[14]	LOC K[13]	LOC K[12]	LOC K[11]	LOC K[10]	LOC K[9]	LOC K[8]	LOC K[7]	LOC K[6]	LOC K[5]	LOC K[4]	LOC K[3]	LOC K[2]	LOC K[1]	LOC K[0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:17	保留。
位 16	<b>LOCKK:</b> 锁键 (Lock key) 该位可随时读出, 它只可通过锁键写入序列修改。 0: 端口配置锁键位未激活 1: 端口配置锁键位被激活, 下次系统复位或 GPIO 复位前 GPIOx_LOCK 寄存器被锁住。 锁键的写入序列: 写 1->写 0->写 1->读 0->读 1 最后一个读可省略, 但可以用来确认锁键已被激活。 注: 在操作锁键的写入序列时, 不能改变 LOCK[15:0] 的值。 操作锁键写入序列中的任何错误将不能激活锁键。
位 15:0	<b>LOCKy:</b> 端口 x 的锁位 y (y=0...15) (Portx Lock bit) 这些位可读可写但只能在 LOCKK 位为 0 时写入。 0: 不锁定端口的配置 1: 锁定端口的配置

#### 6.4.9 端口复用功能低位寄存器 (GPIOx\_AFRL) (x=A..C,F)

偏移地址: 0x20

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]			
rw 15	rw 14	rw 13	rw 12	rw 11	rw 10	rw 9	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1	rw 0
AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 4y+3:4y	<b>AFSELy[1:0]:</b> 端口 x 引脚 y 的复用功能选择 (y=0...7) 软件通过这些位配置相应的 I/O 口复用功能。 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1xxx: 保留
-----------	--

#### 6.4.10 端口复用功能高位寄存器 (GPIOx\_AFRH) (x=A..C,F)

偏移地址: 0x24

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]			

rw 15	rw 14	rw 13	rw 12	rw 11	rw 10	rw 9	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1	rw 0
AFSEL11[3:0]				AFSEL10[3:0]				AFSEL9[3:0]				AFSEL8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 4y+3:4y	AFSELy[1:0]: 端口 x 引脚 y 的复用功能选择 (y=8...15) 软件通过这些位配置相应的 I/O 口复用功能。
	0000: AF0
	0001: AF1
	0010: AF2
	0011: AF3
	0100: AF4
	0101: AF5
	0110: AF6
	0111: AF7
	1xxx: 保留

#### 6.4.11 端口位清除寄存器 (GPIOx\_BRE) (x=A..C,F)

地址偏移: 0x28

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRE y[15]	BRE y[14]	BRE y[13]	BRE y[12]	BRE y[11]	BRE y[10]	BRE y[9]	BRE y[8]	BRE y[7]	BRE y[6]	BRE y[5]	BRE y[4]	BRE y[3]	BRE y[2]	BRE y[1]	BRE y[0]
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位 31:16	保留。
位 15:0	<b>BREy:</b> 清除端口 x 的位 y (y=0...15) 这些位只能写入。读这些位时返回 0x0000 数值。 0: 对对应的 OPTDTy 位不产生影响 1: 清除对应的 OPTDTy 位为 0

#### 6.4.12 极大电流推动/吸入能力切换控制寄存器 (GPIOx\_HDRV) (x=A..C,F)

地址偏移: 0x3C

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HD RV[ 15] [14]	HD RV[ 13]	HD RV1 [12]	HD RV[ 11]	HD RV1 [1]	HD RV0 [9]	HD RV [8]	HD RV [7]	HD RV [6]	HD RV [5]	HD RV [4]	HD RV [3]	HD RV [2]	HD RV [1]	HD RV [0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31:16	保留。
---------	-----

位 15:0	<b>HDRV:</b> 极大电流推动/吸入能力切换控制寄存器 0: 无效 1: GPIO 切换为极大电流推动/吸入能力 (对应于原来的输出模式: 高速)
--------	--

### 6.4.13 电压转换速率切换控制寄存器 (**GPIOx\_SRCTR**) (**x=A..C,F**)

地址偏移: 0x40

复位值: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SR CTR [15]	SR CTR [14]	SR CTR [13]	SR CTR [12]	SR CTR [11]	SR CTR [10]	SR CT R [9]	SR CT R [8]	SR CT R [7]	SR CT R [6]	SR CT R [5]	SR CT R [4]	SR CT R [3]	SR CT R [2]	SR CTR [1]	SR CT R[0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw						

位 31:16	保留。
位 15:0	<b>SRCTR:</b> 电压转换速率切换控制位 0: 适中转换速率 1: 增强转换速率

# 7 系统配置控制器 (SYSCFG)

## 7.1 简介

该器件具有一组配置寄存器。系统配置控制器的主要用途如下：

- 重映射部分DMA触发源到其它不同DMA通道上
- 管理连接到GPIO口的外部中断

## 7.2 SYSCFG寄存器

下面列出了 SYSCFG 寄存器映象和复位数值。必须以字（32 位）的方式操作这些外设寄存器。

表格 7-1 SYSCFG寄存器地址映射和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	SYSCFG_CFGR1	保留																		TMR17_DMA_RMP	TMR16_DMA_RMP	USART1_RX_DMA_RMP	USART1_TX_DMA_RMP	ADC_DMA_RMP	IR_MOD[1:0]	IR_POL	PA11_12_RMP	保留		MEM_MODE			
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x				
008H	SYSCFG_EXTIC1	保留																		EXTINT3 [3: 0]	EXTINT2 [3: 0]	EXTINT1 [3: 0]	EXTINT0 [3: 0]										
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
00CH	SYSCFG_EXTIC2	保留																		EXTINT7 [3: 0]	EXTINT6 [3: 0]	EXTINT5 [3: 0]	EXTINT4 [3: 0]										
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
010H	SYSCFG_EXTIC3	保留																		EXTINT11 [3: 0]	EXTINT10 [3: 0]	EXTINT9 [3: 0]	EXTINT8 [3: 0]										
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
014H	SYSCFG_EXTIC4	保留																		EXTINT15 [3: 0]	EXTINT14 [3: 0]	EXTINT13 [3: 0]	EXTINT12 [3: 0]										
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

### 7.2.1 SYSCFG配置寄存器1 (SYSCFG\_CFGR1)

偏移地址：0x000

复位值：0x0000\_000X

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	res	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																		保留		MEM_MODE												
res		rw		rw		rw		res		r																						
位 31~13										保留																		MEM_MODE				

位 12	<b>TMR17_DMA_RMP:</b> TMR17 DMA 请求重映射位 由软件设置和清除该位。 0x0: 无重映射 (TMR17_CH1 与 TMR17_UP 的 DMA 请求映射在 DMA 通道 1 上) 0x1: 重映射 1 (TMR17_CH1 与 TMR17_UP 的 DMA 请求映射在 DMA 通道 2 上)
位 11	<b>TMR16_DMA_RMP:</b> TMR16 DMA 请求重映射位 由软件设置和清除该位。 0x0: 无重映射 (TMR16_CH1 与 TMR16_UP 的 DMA 请求映射在 DMA 通道 3 上) 0x1: 重映射 1 (TMR16_CH1 与 TMR16_UP 的 DMA 请求映射在 DMA 通道 4 上)
位 10	<b>USART1_RX_DMA_RMP:</b> USART1 RX DMA 请求重映射位 由软件设置和清除该位。它控制着 USART1 RX DMA 通道请求的重映射。 0: 无重映射 (USART1_RX 的 DMA 请求映射在 DMA 通道 3 上) 1: 重映射 (USART1_RX 的 DMA 请求映射在 DMA 通道 5 上)
位 9	<b>USART1_TX_DMA_RMP:</b> USART1 TX DMA 请求重映射位 由软件设置和清除该位。它控制着 USART1 TX DMA 通道请求的重映射。 0: 无重映射 (USART1_TX 的 DMA 请求映射在 DMA 通道 2 上) 1: 重映射 (USART1_TX 的 DMA 请求映射在 DMA 通道 4 上)
位 8	<b>ADC_DMA_RMP:</b> ADC DMA 请求重映射位 由软件设置和清除该位。它控制着 ADC DMA 通道请求的重映射。 0: 无重映射 (ADC 的 DMA 请求映射在 DMA 通道 1 上) 1: 重映射 (ADC 的 DMA 请求映射在 DMA 通道 2 上)
位 7: 6	<b>IR_MOD[1:0]:</b> 红外调制包络信号源选择 用于选择红外调制包络信号源: 00: TMR16 01: USART1 10: USART2 11: 保留
位 5	<b>IR_POL:</b> 红外输出极性选择 0: 红外线发射输出 (IR_OUT) 不反向 1: 红外线发射输出 (IR_OUT) 反向
位 4	<b>PA11_12_RMP:</b> 小封装(20 脚)上 PA11 和 PA12 的重映射 由软件设置与清除该位。它控制着小封装上 PA9/10 或 PA11/12 脚的映射 0: 没有重映射(PA9/PA10 脚对应 PA9/PA10 脚) 1: 重映射 (PA11/PA12 脚映射到 PA9/PA10 脚上)
位 3:2	保留
位 1:0	<b>MEM_MODE[1:0]:</b> 启动模式状态位 此位仅供读取，显示复位后的启动区域。这些位值由 BOOT0 引脚和 nBOOT1 的配置值决定。 X0: 从主存存储器启动 01: 从系统存储器启动 11: 从内置 SRAM 启动

## 7.2.2 SYSCFG 外部中断配置寄存器1 (SYSCFG\_EXTIC1)

地址偏移:0x08

复位值:0x0000\_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTINT3[3:0]				EXTINT2[3:0]				EXTINT1[3:0]				EXTINT0[3:0]			
rw				rw				rw				rw			
位 31:16				保留。											

位 15:0	<b>EXTINTx[3:0]: EXTINTx 配置 (x=0...3) (EXTINTx configuration)</b> 这些位可由软件读写, 用于选择 EXTINTx 外部中断的输入源。 0000: PA[x]引脚 0001: PB[x]引脚 0101: PF[x]引脚
--------	---

### 7.2.3 SYSCFG外部中断配置寄存器2 (SYSCFG\_EXTIC2)

地址偏移:0x0C

复位值:0x0000\_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16												
保留																											
res																											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
EXTINT7[3:0]				EXTINT6[3:0]				EXTINT5[3:0]				EXTINT4[3:0]															
rw				rw				rw				rw															
位 31:16		保留。																									
位 15:0		<b>EXTINTx[3:0]: EXTINTx 配置 (x=4...7) (EXTINTx configuration)</b> 这些位可由软件读写, 用于选择 EXTINTx 外部中断的输入源。 0000: PA[x]引脚 0001: PB[x]引脚 0101: PF[x]引脚																									

### 7.2.4 SYSCFG外部中断配置寄存器3 (SYSCFG\_EXTIC3)

地址偏移:0x10

复位值:0x0000\_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16												
保留																											
res																											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
EXTINT11[3:0]				EXTINT10[3:0]				EXTINT9[3:0]				EXTINT8[3:0]															
rw				rw				rw				rw															
位 31:16		保留。																									
位 15:0		<b>EXTINTx[3:0]: EXTINTx 配置 (x=8...11) (EXTINTx configuration)</b> 这些位可由软件读写, 用于选择 EXTINTx 外部中断的输入源。 0000: PA[x]引脚 0001: PB[x]引脚																									

### 7.2.5 SYSCFG外部中断配置寄存器4 (SYSCFG\_EXTIC4)

地址偏移:0x14

复位值:0x0000\_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

EXTINT15[3:0]	EXTINT14[3:0]	EXTINT13[3:0]	EXTINT12[3:0]
rw	rw	rw	rw
位 31:16	保留。		
位 15:0	<b>EXTINTx[3:0]: EXTINTx 配置 (x=12...15) (EXTINTx configuration)</b> 这些位可由软件读写，用于选择 EXTINTx 外部中断的输入源。 0000: PA[x]引脚 0001: PB[x]引脚 0010: PC[x]引脚		

# 8 中断和事件 (INT/EVENT)

## 8.1 嵌套向量中断控制器

### 特性

- 28 个可屏蔽中断通道（不包含 16 个 Cortex™-M4 的中断线）；
- 16 个可编程的优先等级（使用了 4 位中断优先级）；
- 低延迟的异常和中断处理；
- 电源管理控制；
- 系统控制寄存器的实现；

嵌套向量中断控制器 (NVIC) 和处理器核的接口紧密相连，可以实现低延迟的中断处理和高效地处理晚到的中断。

嵌套向量中断控制器管理着包括内核异常等中断。

### 8.1.1 系统嘀嗒 (SysCNTRick) 校准值寄存器

系统嘀嗒校准值固定为 9000，例如：当 HCLK=72 MHz，系统嘀嗒时钟设定为 9MHz (HCLK/8)，产生 1ms 时间基准。

### 8.1.2 中断和异常向量

下面两个表，分别列出了 AT32F421 产品的向量表。

表格 8-1 AT32F421 产品的向量表

位置	优先 级	优先级 类型	名称	说明	地址
	-	-	-	保留	0x0000_0000
	-3	固定	Reset	复位	0x0000_0004
	-2	固定	NMI	不可屏蔽中断 RCC 时钟失效检测 (CFD) 联接到 NMI 向量	0x0000_0008
	-1	固定	硬件失效 (HardFault)	所有类型的失效	0x0000_000C
	0	可设置	存储管理 (MemoryManage)	存储器管理	0x0000_0010
	1	可设置	总线错误 (BusFault)	预取指失败，存储器访问失败	0x0000_0014
	2	可设置	错误应用 (UsageFault)	未定义的指令或非法状态	0x0000_0018
	-	-	-	保留	0x0000_001C ~0x0000_002B
	3	可设置	SVCall	通过 SWI 指令的系统服务调用	0x0000_002C
	4	可设置	调试监控 (DebugLENonitor)	调试监控器	0x0000_0030
	-	-	-	保留	0x0000_0034
	5	可设置	PendSV	可挂起的系统服务	0x0000_0038
	6	可设置	SysCNTRick	系统嘀嗒定时器	0x0000_003C
0	7	可设置	WWDG	窗口定时器中断	0x0000_0040
1	8	可设置	PVD	连到 EXTI 线 16 的电源电压检测 (PVD) 中断	0x0000_0044

2	9	可设置	ERTC	连接到 EXTI 线 17, 19 的 ERTC 中断	0x0000_0048
3	10	可设置	FLASH	闪存全局中断	0x0000_004C
4	11	可设置	RCC	复位和时钟控制 (RCC) 中断	0x0000_0050
5	12	可设置	EXTINT1_0	EXTI 线 1_0 中断	0x0000_0054
6	13	可设置	EXTINT3_2	EXTI 线 3_2 中断	0x0000_0058
7	14	可设置	EXTINT15_4	EXTI 线 15_4 中断	0x0000_005C
8	15	可设置	-	保留	0x0000_0060
9	16	可设置	DMA 通道 1	DMA 通道 1 全局中断	0x0000_0064
10	17	可设置	DMA 通道 3_2	DMA 通道 3_2 全局中断	0x0000_0068
11	18	可设置	DMA 通道 5_4	DMA 通道 5_4 全局中断	0x0000_006C
12	19	可设置	ADC_COMP	ADC 和 COMP 全局中断	0x0000_0070
13	20	可设置	TMR1_BRK_UP_TRG_C OM	TMR1 中断	0x0000_0074
14	21	可设置	TMR1_CC	TMR1 捕获比较中断	0x0000_0078
15	22	可设置	-	保留	0x0000_007C
16	23	可设置	TMR3	TMR3 全局中断	0x0000_0080
17	24	可设置	TMR6	TMR6 全局中断	0x0000_0084
18	25	可设置	-	保留	0x0000_0088
19	26	可设置	TMR14	TMR14 全局中断	0x0000_008C
20	27	可设置	TMR15	TMR15 全局中断	0x0000_0090
21	28	可设置	TMR16	TMR16 全局中断	0x0000_0094
22	29	可设置	TMR17	TMR17 全局中断	0x0000_0098
23	30	可设置	I <sup>2</sup> C1_EV	I <sup>2</sup> C1 事件中断	0x0000_009C
24	31	可设置	I <sup>2</sup> C2_EV	I <sup>2</sup> C2 事件中断	0x0000_00A0
25	32	可设置	SPI1	SPI1 全局中断	0x0000_00A4
26	33	可设置	SPI2	SPI2 全局中断	0x0000_00A8
27	34	可设置	USART1	USART1 全局中断	0x0000_00AC
28	35	可设置	USART2	USART2 全局中断	0x0000_00B0
29	36	可设置	-	保留	0x0000_00B4
30	37	可设置	-	保留	0x0000_00B8
31	38	可设置	-	保留	0x0000_00BC
32	39	可设置	I <sup>2</sup> C1_ER	I <sup>2</sup> C1 错误中断	0x0000_00C0
33	40	可设置	-	保留	0x0000_00C4
34	41	可设置	I <sup>2</sup> C2_ER	I <sup>2</sup> C2 错误中断	0x0000_00C8

## 8.2 外部中断/事件控制器 (EXTI)

它有 22 个能产生事件/中断请求的边沿检测器。每个输入线可以独立地配置输入类型（事件或中断）和对应的触发事件（上升沿或下降沿或者双边沿都触发）。每个输入线都可以独立地被屏蔽。挂起寄存器保持着状态线的中断请求。

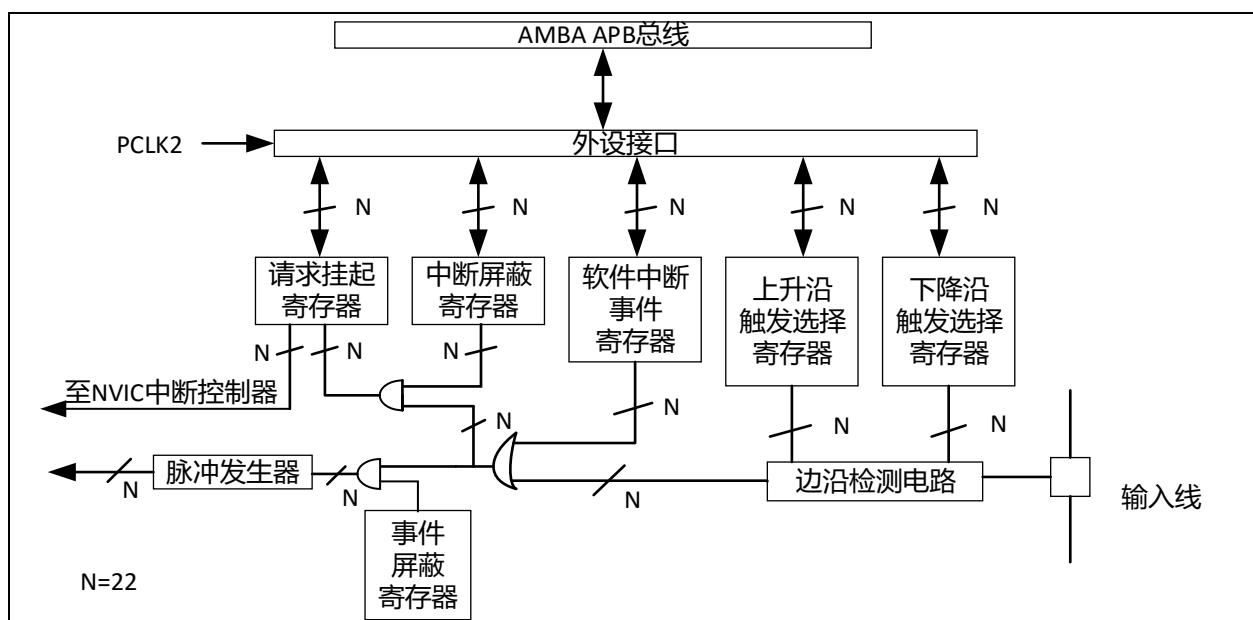
### 8.2.1 主要特性

EXTI 控制器的主要特性如下：

- 每个中断/事件都有独立的触发和屏蔽
- 每个中断线都有专用的状态位
- 支持多达 20 个软件的中断/事件请求
- 检测脉冲宽度低于 APB2 时钟宽度的外部信号。参见数据手册中电气特性部分的相关参数。

### 8.2.2 框图

图表 8-1 外部中断/事件控制器框图



### 8.2.3 唤醒事件管理

AT32F421 可以处理外部或内部事件来唤醒内核 (WFE)。唤醒事件可以通过下述配置产生：

- 在外设的控制寄存器使能一个中断，但不在 NVIC 中使能，同时在 Cortex™-M4 的系统控制寄存器中使能 SEVONPEND 位。当 CPU 从 WFE 恢复后，需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位（在 NVIC 中断清除挂起寄存器中）。
- 配置一个外部或内部 EXTI 线为事件模式，当 CPU 从 WFE 恢复后，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或 NVIC 中断通道挂起位。

使用外部 I/O 端口作为唤醒事件，请参见 [8.2.4 节的功能说明](#)。

### 8.2.4 功能说明

要产生中断，必须先配置好并使能中断线。根据需要的边沿检测设置 2 个触发寄存器，同时在中断屏蔽寄存器的相应位写‘1’允许中断请求。当外部中断线上发生了期待的边沿时，将产生一个中断请求，对应的挂起位也随之被置‘1’。在挂起寄存器的对应位写‘1’，将清除该中断请求。

如果需要产生事件，必须先配置好并使能事件线。根据需要的边沿检测通过设置 2 个触发寄存器，同时在事件屏蔽寄存器的相应位写‘1’允许事件请求。当事件线上发生了需要的边沿时，将产生一个事件请求脉冲，对应的挂起位不被置‘1’。

通过在软件中断/事件寄存器写‘1’，也可以通过软件产生中断/事件请求。

**硬件中断选择**

通过下面的过程来配置 22 个线路做为中断源：

- 配置 22 个中断线的屏蔽位（`EXTI_INTEN`）
- 配置所选中断线的触发选择位（`EXTI_RTRSEL` 和 `EXTI_FTRSEL`）；
- 配置对应到外部中断控制器（`EXTI`）的 NVIC 中断通道的使能和屏蔽位，使得 22 个中断线中的请求可以被正确地响应。

#### 硬件事件选择

通过下面的过程，可以配置 22 个线路为事件源

- 配置 22 个事件线的屏蔽位（`EXTI_EVTEN`）
- 配置事件线的触发选择位（`EXTI_RTRSEL` 和 `EXTI_FTRSEL`）

#### 软件中断/事件的选择

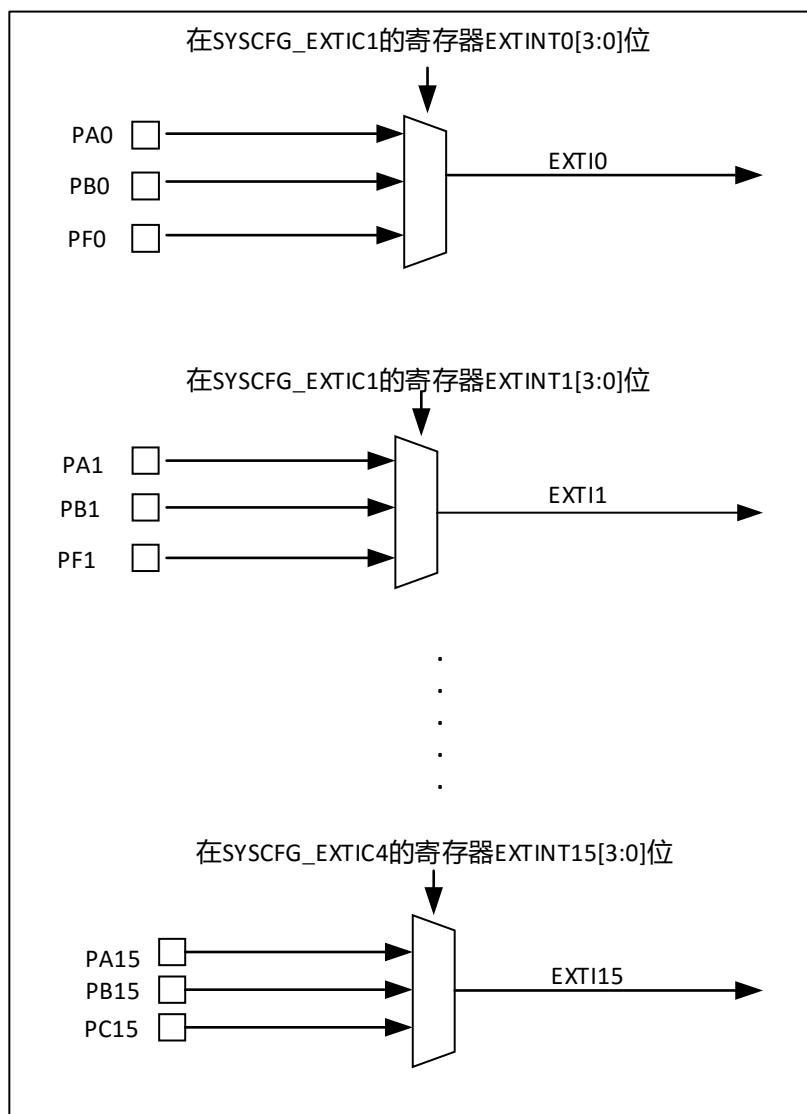
22 个线路可以被配置成软件中断/事件线。下面是产生软件中断的过程：

- 配置 22 个中断/事件线屏蔽位（`EXTI_INTEN`, `EXTI_EVTEN`）
- 设置软件中断寄存器的请求位（`EXTI_SWIE`）

### 8.2.5 外部中断/事件线路映像

55 通用 I/O 端口以下图的方式连接到 16 个外部中断/事件线上：

图表 8-2 外部中断通用 I/O 映像



通过 SYSCFG\_EXTIx 配置 GPIO 线上的外部中断/事件。参见 [7.2 节](#)。

另外四个 EXTI 线的连接方式如下：

- EXTI 线 16 连接到 PVD 输出
- EXTI 线 17 连接到 RTC 闹钟事件
- EXTI 线 18 保留未用
- EXTI 线 19 连接到 RTC 入侵和时间戳事件
- EXTI 线 20 保留未用
- EXTI 线 21 连接到 COMP 事件

### 8.3 EXTI 寄存器描述

必须以字（32 位）的方式操作这些外设寄存器。

下表列出了 EXTI 寄存器的映像和复位值。

表格 8-2 外部中断/事件控制器寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	MR21	21
0x00	EXTI_INTEN	保留												
0x04	EXTI_EVTEN	保留												
0x08	EXTI_RTRSEL	保留												
0x0C	EXTI_FTRSEL	保留												
0x10	EXTI_SWIE	保留												
0x14	EXTI_PND	保留												

### 8.3.1 中断屏蔽寄存器 (EXTI\_INTEN)

偏移地址: 0x00

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								MR21	保留	MR19	保留	MR17	MR16	MR15	MR14
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MRO

rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 22															保留, 必须始终保持为复位状态 (0)。
位 21															<b>MRx:</b> 线 x 上的中断屏蔽 (Interrupt LENask on line x) 0: 屏蔽来自线 x 上的中断请求; 1: 开放来自线 x 上的中断请求。
位 20															保留, 必须始终保持为复位状态 (0)。
位 19															<b>MRx:</b> 线 x 上的中断屏蔽 (Interrupt LENask on line x) 0: 屏蔽来自线 x 上的中断请求; 1: 开放来自线 x 上的中断请求。
位 18															保留, 必须始终保持为复位状态 (0)。

位 17: 0	<b>MRx:</b> 线 x 上的中断屏蔽 (Interrupt LENask on line x) 0: 屏蔽来自线 x 上的中断请求; 1: 开放来自线 x 上的中断请求。
---------	---

### 8.3.2 事件屏蔽寄存器 (EXTI\_EVTEN)

偏移地址: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								MR21	保留		MR19	保留		MR17	MR16
res									rw	res	rw	res	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 22	保留, 必须始终保持为复位状态 (0)。
位 21	<b>MRx:</b> 线 x 上的事件屏蔽 (Event LENask on line x) 0: 屏蔽来自线 x 上的事件请求; 1: 开放来自线 x 上的事件请求。
位 20	保留, 必须始终保持为复位状态 (0)。
位 19	<b>MRx:</b> 线 x 上的事件屏蔽 (Event LENask on line x) 0: 屏蔽来自线 x 上的事件请求; 1: 开放来自线 x 上的事件请求。
位 18	保留, 必须始终保持为复位状态 (0)。
位 17: 0	<b>MRx:</b> 线 x 上的事件屏蔽 (Event LENask on line x) 0: 屏蔽来自线 x 上的事件请求; 1: 开放来自线 x 上的事件请求。

### 8.3.3 上升沿触发选择寄存器 (EXTI\_RTRSEL)

偏移地址: 0x08

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								TR21	保留		TR19	保留		TR17	TR16
res									rw	res	rw	res	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 22	保留, 必须始终保持为复位状态 (0)。
位 21	<b>TRx:</b> 线 x 上的上升沿触发事件配置位 (Rising trigger event configuration bit of line x) 0: 禁止输入线 x 上的上升沿触发 (中断和事件) 1: 允许输入线 x 上的上升沿触发 (中断和事件)
位 20	保留, 必须始终保持为复位状态 (0)。
位 19	<b>TRx:</b> 线 x 上的上升沿触发事件配置位 (Rising trigger event configuration bit of line x) 0: 禁止输入线 x 上的上升沿触发 (中断和事件) 1: 允许输入线 x 上的上升沿触发 (中断和事件)

位 18	保留，必须始终保持为复位状态（0）。
位 17: 0	<b>TRx:</b> 线 x 上的上升沿触发事件配置位（Rising trigger event configuration bit of line x） 0: 禁止输入线 x 上的上升沿触发（中断和事件） 1: 允许输入线 x 上的上升沿触发（中断和事件）

注意：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。在写 **EXTI\_RTRSEL** 寄存器时，在外部中断线上的上升沿信号不能被识别，挂起位也不会被置位。在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

### 8.3.4 下降沿触发选择寄存器（**EXTI\_FTRSEL**）

偏移地址: 0x0C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								TR21	保留	TR19	保留	TR17	保留	TR16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 22	保留，必须始终保持为复位状态（0）。
位 21	<b>TRx:</b> 线 x 上的下降沿触发事件配置位（Falling trigger event configuration bit of line x） 0: 禁止输入线 x 上的下降沿触发（中断和事件） 1: 允许输入线 x 上的下降沿触发（中断和事件）
位 20	保留，必须始终保持为复位状态（0）。
位 19	<b>TRx:</b> 线 x 上的下降沿触发事件配置位（Falling trigger event configuration bit of line x） 0: 禁止输入线 x 上的下降沿触发（中断和事件） 1: 允许输入线 x 上的下降沿触发（中断和事件）
位 18	保留，必须始终保持为复位状态（0）。
位 17: 0	<b>TRx:</b> 线 x 上的下降沿触发事件配置位（Falling trigger event configuration bit of line x） 0: 禁止输入线 x 上的下降沿触发（中断和事件） 1: 允许输入线 x 上的下降沿触发（中断和事件）

注意：外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。在写 **EXTI\_FTRSEL** 寄存器时，在外部中断线上的下降沿信号不能被识别，挂起位不会被置位。在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

### 8.3.5 软件中断事件寄存器（**EXTI\_SWIE**）

偏移地址: 0x10

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								SW IER21	保留	SW IER19	保留	SW IER17	保留	SW IER16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW IER15	SW IER14	SW IER13	SW IER12	SW IER11	SW IER10	SW IER9	SW IER8	SW IER7	SW IER6	SW IER5	SW IER4	SW IER3	SW IER2	SW IER1	SW IER0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 22	保留，必须始终保持为复位状态（0）。														
位 21	<b>SWIERx:</b> 线 x 上的软件中断 （Software interrupt on line x） 如果在 EXTI_INTEN 中这条线的对应位被置“1”，在该位为“0”时对该位写“1”将置起 EXTI_PND 中的对应位，并产生中断请求。 注：通过清除 EXTI_PND 的对应位（写入‘1’），可以清除该位为‘0’。														
位 20	保留，必须始终保持为复位状态（0）。														
位 19	<b>SWIERx:</b> 线 x 上的软件中断 （Software interrupt on line x） 如果在 EXTI_INTEN 中这条线的对应位被置“1”，在该位为“0”时对该位写“1”将置起 EXTI_PND 中的对应位，并产生中断请求。 注：通过清除 EXTI_PND 的对应位（写入‘1’），可以清除该位为‘0’。														
位 18	保留，必须始终保持为复位状态（0）。														
位 17: 0	<b>SWIERx:</b> 线 x 上的软件中断 （Software interrupt on line x） 如果在 EXTI_INTEN 中这条线的对应位被置“1”，在该位为“0”时对该位写“1”将置起 EXTI_PND 中的对应位，并产生中断请求。 注：通过清除 EXTI_PND 的对应位（写入‘1’），可以清除该位为‘0’。														

### 8.3.6 挂起寄存器 (EXTI\_PND)

偏移地址: 0x14

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								PR21	保留	PR19	保留	PR17	保留	PR16	
				res								rw	res	rw	res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 22	保留，必须始终保持为复位状态（0）。														
位 21	<b>PRx:</b> 挂起位 (Pending bit) 0: 没有发生触发请求 1: 发生了选择的触发请求当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。														
位 20	保留，必须始终保持为复位状态（0）。														
位 19	<b>PRx:</b> 挂起位 (Pending bit) 0: 没有发生触发请求 1: 发生了选择的触发请求当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。														
位 18	保留，必须始终保持为复位状态（0）。														
位 17: 0	<b>PRx:</b> 挂起位 (Pending bit) 0: 没有发生触发请求 1: 发生了选择的触发请求当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。														

## 9 DMA控制器 (DMA)

### 9.1 DMA简介

直接存储器存取 (DMA) 用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。无须 CPU 干预，数据可以通过 DMA 快速地移动，这就节省了 CPU 的资源来做其他操作。

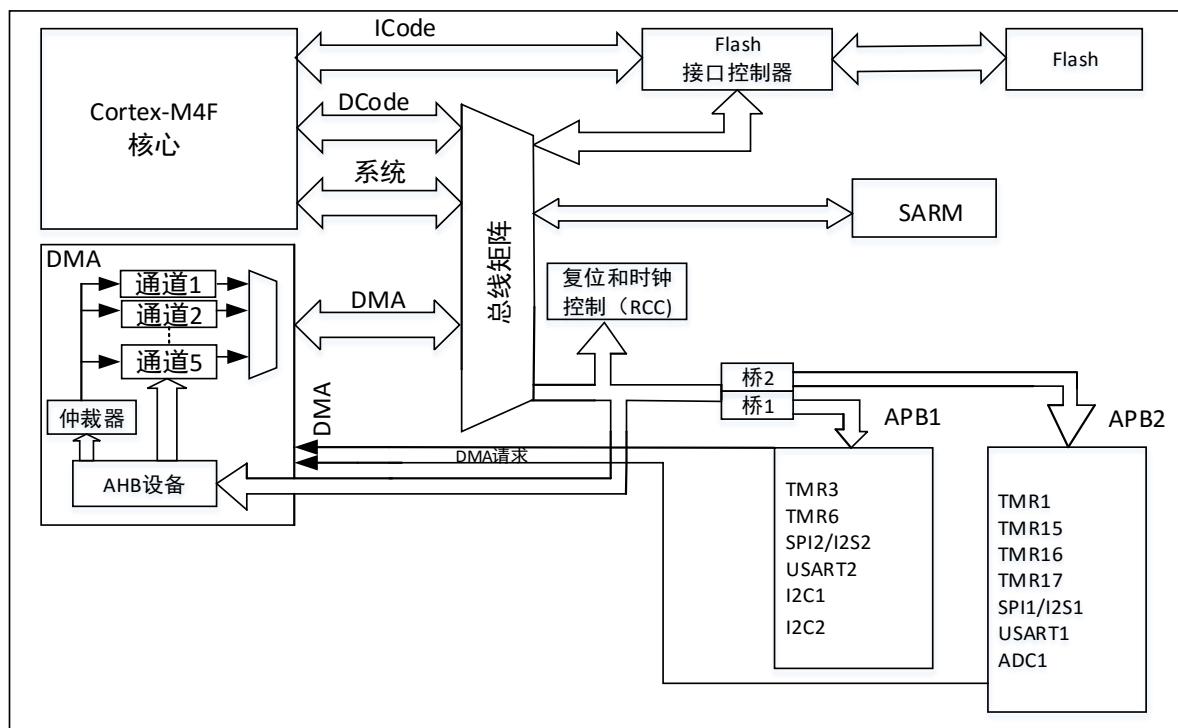
DMA 控制器有 5 个通道，每个通道专门用来管理来自于一个或多个外设对存储器访问的请求。还有一个仲裁器来协调各个 DMA 请求的优先权。

### 9.2 DMA主要特性

- 5个独立的可配置的通道（请求）
- 每个通道都直接连接专用的硬件DMA请求，每个通道都同样支持软件触发。这些功能通过软件来配置
- 在同一个DMA模块上，多个请求间的优先权可以通过软件编程设置（共有四级：很高、高、中等和低），优先权设置相等时由硬件决定（请求0优先于请求1，依此类推）
- 独立数据源和目标数据区的传输宽度（字节、半字、全字），模拟打包和拆包的过程。源和目标地址必须按数据传输宽度对齐
- 支持循环的缓冲器管理
- 每个通道都有3个事件标志（DMA半传输、DMA传输完成和DMA传输出错），这3个事件标志逻辑或成为一个单独的中断请求
- 存储器和存储器间的传输
- 外设和存储器、存储器和外设之间的传输
- 闪存、SRAM、外设的SRAM、APB1、APB2和AHB外设均可作为访问的源和目标。
- 可编程的数据传输数目：最大为65535

下面为功能框图：

图表 9-1 DMA框图



注意：根据不同型号，图中 DMA 外设可能会有所减少。

## 9.3 功能描述

DMA 控制器和 Cortex™-M4 核心共享系统数据总线，执行直接存储器数据传输。当 CPU 和 DMA 同时访问相同的目标（RAM 或外设）时，DMA 请求会暂停 CPU 访问系统总线达若干个周期，总线仲裁器执行循环调度，以保证 CPU 至少可以得到一半的系统总线（存储器或外设）带宽。

### 9.3.1 DMA处理

在发生一个事件后，外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道的优先权处理请求。当 DMA 控制器开始访问发出请求的外设时，DMA 控制器立即发送给它一个应答信号。当从 DMA 控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，DMA 控制器同时撤销应答信号。如果有更多的请求时，外设可以启动下一个周期。

总之，每次 DMA 传送由 3 个操作组成：

- 从外设数据寄存器或者从当前外设/存储器地址寄存器指示的存储器地址取数据，第一次传输时的开始地址是 DMA\_CPBAX 或 DMA\_CMBAX 寄存器指定的外设基址或存储器单元
- 存数据到外设数据寄存器或者当前外设/存储器地址寄存器指示的存储器地址，第一次传输时的开始地址是 DMA\_CPBAX 或 DMA\_CMBAX 寄存器指定的外设基址或存储器单元
- 执行一次 DMA\_TCNTx 寄存器的递减操作，该寄存器包含未完成的操作数目

### 9.3.2 仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。

优先权管理分 2 个阶段：

- 软件：每个通道的优先权可以在 DMA\_CHCTRLx 寄存器中设置，有 4 个等级

- 最高优先级
  - 高优先级
  - 中等优先级
  - 低优先级
- 硬件：如果2个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有较高的优先权。举个例子，通道2优先于通道4

### 9.3.3 DMA通道

每个通道都可以在有固定地址的外设寄存器和存储器地址之间执行 DMA 传输。DMA 传输的数据量是可编程的，最大达到 65535。包含要传输的数据项数量的寄存器，在每次传输后递减。

#### 可编程的数据量

外设和存储器的传输数据量可以通过 DMA\_CHCTRLx 寄存器中的 PWIDTH 和 MWIDTH 编程。

#### 指针增量

通过设置 DMA\_CHCTRLx 寄存器中的 PINC 和 MINC 标志位，外设和存储器的指针在每次传输后可以有选择地完成自动增量。当设置为增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决于所选的数据宽度为 1、2 或 4。第一个传输的地址是存放在 DMA\_CPBAX/DMA\_CMBAx 寄存器中地址。在传输过程中，这些寄存器保持它们初始的数值，软件不能改变和读出目前正在传输的地址（它在内部的当前外设/存储器地址寄存器中）。

当通道配置为非循环模式时，传输结束后（即传输计数变为 0）将不再产生 DMA 操作。要开始新的 DMA 传输，需要在关闭 DMA 通道的情况下，在 DMA\_TCNTx 寄存器中重新写入传输数目。

**注意：** 如果一个 DMA 通道关闭使能，DMA 寄存器值不会被复位。DMA 通道寄存器（DMA\_CCRx, DMA\_CPARx 和 DMA\_CMARx）保持上一次的设置值。

在循环模式下，最后一次传输结束时，DMA\_TCNTx 寄存器的内容会自动地被重新加载为其初始数值，内部的当前外设/存储器地址寄存器也被重新加载为 DMA\_CPBAX/DMA\_CMBAx 寄存器设定的初始地址。

#### 通道配置过程

下面是配置 DMA 通道 x 的过程（x 代表通道号）：

1. 在 DMA\_CPBAX 寄存器中设置外设寄存器的地址。发生外设数据传输请求时，这个地址将是数据传输的源或目标。
2. 在 DMA\_CMBAx 寄存器中设置数据存储器的地址。发生外设数据传输请求时，传输的数据将从这个地址读出或写入这个地址。
3. 在 DMA\_TCNTx 寄存器中设置要传输的数据量。在每个数据传输后，这个数值递减。
4. 在 DMA\_CHCTRLx 寄存器的 CHPL[1: 0] 位中设置通道的优先级。
5. 在 DMA\_CHCTRLx 寄存器中设置数据传输的方向、循环模式、外设和存储器的增量模式、外设和存储器的数据宽度、传输一半产生中断或传输完成产生中断。
6. 设置 DMA\_CHCTRLx 寄存器的 ENABLE 位，启动该通道。

一旦启动了 DMA 通道，它既可响应连到该通道上的外设的 DMA 请求。

当传输一半的数据后，半传输标志（HTIF）被置 1，当设置了允许半传输中断位（HTIE）时，将产生一个中断请求。在数据传输结束后，传输完成标志（TCIF）被置 1，当设置了允许传输完成中断位（TCIE）时，将产生一个中断请求。

#### 循环模式

循环模式用于处理循环缓冲区和连续的数据传输（如 ADC 的扫描模式）。在 DMA\_CHCTRLx 寄存器中的 CIRM 位用于开启这一功能。当启动了循环模式，数据传输的数目变为 0 时，将会自动地被恢复成配置通道时设置的初值，DMA 操作将会继续进行。

#### 存储器到存储器模式

DMA 通道的操作可以在没有外设请求的情况下进行，这种操作就是存储器到存储器模式。

当设置了 DMA\_CHCTRLx 寄存器中的 MEMTOMEM 位之后，在软件设置了 DMA\_CHCTRLx 寄存器中的 CHEN 位启动 DMA 通道时，DMA 传输将马上开始。当 DMA\_TCNTx 寄存器变为 0 时，DMA 传输结束。存储器到存储器模式不能与循环模式同时使用。

### 9.3.4 可编程的数据传输宽度、对齐方式和数据大小端

当 PWIDTH 和 MWIDTH 不相同时，DMA 模块按照下表进行数据对齐。

表格 9-1 可编程的数据传输宽度和大小端操作（当 PINC = MINC = 1）

源端宽度	目标宽度	传输数目	源: 地址/数据	传输操作	目标: 地址/数据
8	8	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在 0x0 读 B0[7: 0], 在 0x0 写 B0[7: 0] 2: 在 0x1 读 B1[7: 0], 在 0x1 写 B1[7: 0] 3: 在 0x2 读 B2[7: 0], 在 0x2 写 B2[7: 0] 4: 在 0x3 读 B3[7: 0], 在 0x3 写 B3[7: 0]	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3
8	16	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在 0x0 读 B0[7: 0], 在 0x0 写 00B0[15: 0] 2: 在 0x1 读 B1[7: 0], 在 0x2 写 00B1[15: 0] 3: 在 0x2 读 B2[7: 0], 在 0x4 写 00B2[15: 0] 4: 在 0x3 读 B3[7: 0], 在 0x6 写 00B3[15: 0]	0x0 / 00B0 0x2 / 00B1 0x4 / 00B2 0x6 / 00B3
8	32	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在 0x0 读 B0[7: 0], 在 0x0 写 000000B0[31: 0] 2: 在 0x1 读 B1[7: 0], 在 0x4 写 000000B1[31: 0] 3: 在 0x2 读 B2[7: 0], 在 0x8 写 000000B2[31: 0] 4: 在 0x3 读 B3[7: 0], 在 0xC 写 000000B3[31: 0]	0x0 / 000000B0 0x4 / 000000B1 0x8 / 000000B2 0xC / 000000B3
16	8	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在 0x0 读 B1B0[15: 0], 在 0x0 写 B0[7: 0] 2: 在 0x2 读 B3B2[15: 0], 在 0x1 写 B2[7: 0] 3: 在 0x4 读 B5B4[15: 0], 在 0x2 写 B4[7: 0] 4: 在 0x6 读 B7B6[15: 0], 在 0x3 写 B6[7: 0]	0x0 / B0 0x1 / B2 0x2 / B4 0x3 / B6
16	16	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在 0x0 读 B1B0[15: 0], 在 0x0 写 B1B0[15: 0] 2: 在 0x2 读 B3B2[15: 0], 在 0x2 写 B3B2[15: 0] 3: 在 0x4 读 B5B4[15: 0], 在 0x4 写 B5B4[15: 0] 4: 在 0x6 读 B7B6[15: 0], 在 0x6 写 B7B6[15: 0]	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6
16	32	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在 0x0 读 B1B0[15: 0], 在 0x0 写 0000B1B0[31: 0] 2: 在 0x2 读 B3B2[15: 0], 在 0x4 写 0000B3B2[31: 0] 3: 在 0x4 读 B5B4[15: 0], 在 0x8 写 0000B5B4[31: 0] 4: 在 0x6 读 B7B6[15: 0], 在 0xC 写 0000B7B6[31: 0]	0x0 / 0000B1B0 0x4 / 0000B3B2 0x8 / 0000B5B4 0xC / 0000B7B6
32	8	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在 0x0 读 B3B2B1B0[31: 0], 在 0x0 写 B0[7: 0] 2: 在 0x4 读 B7B6B5B4[31: 0], 在 0x1 写 B4[7: 0] 3: 在 0x8 读 BBBAB9B8[31: 0], 在 0x2 写 B8[7: 0] 4: 在 0xC 读 BFBEBDBC[31: 0], 在 0x3 写 BC[7: 0]	0x0 / B0 0x1 / B4 0x2 / B8 0x3 / BC
32	16	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在 0x0 读 B3B2B1B0[31: 0], 在 0x0 写 B1B0[15: 0] 2: 在 0x4 读 B7B6B5B4[31: 0], 在 0x2 写 B5B4[15: 0] 3: 在 0x8 读 BBBAB9B8[31: 0], 在 0x4 写 B9B8[15: 0] 4: 在 0xC 读 BFBEBDBC[31: 0], 在 0x6 写 BDBC[15: 0]	0x0 / B1B0 0x2 / B5B4 0x4 / B9B8 0x6 / BDBC
32	32	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在 0x0 读 B3B2B1B0[31: 0], 在 0x0 写 B3B2B1B0[31: 0] 2: 在 0x4 读 B7B6B5B4[31: 0], 在 0x4 写 B7B6B5B4[31: 0] 3: 在 0x8 读 BBBAB9B8[31: 0], 在 0x8 写 BBBAB9B8[31: 0] 4: 在 0xC 读 BFBEBDBC[31: 0], 在 0xC 写 BFBEBDBC[31: 0]	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC

#### 操作一个不支持字节或半字写的 AHB 设备

当 DMA 模块开始一个 AHB 的字节或半字写操作时，数据将在 HWDATA[31: 0]总线中未使用的部分重複。因此，如果 DMA 以字节或半字写入不支持字节或半字写操作的 AHB 设备时（即 HSIZE 不适于该模块），不会发生错误，DMA 将按照下面两个例子写入 32 位 HWDATA 数据：

- 当HSIZE=半字时，写入半字’0xABCD’，DMA将设置HWDATA总线为’0xABCDABCD’
- 当HSIZE=字节时，写入字节’0xAB’，DMA将设置HWDATA总线为’0xABABABAB’

假定 AHB/APB 桥是一个 AHB 的 32 位从设备，它不处理 HSIZE 参数，它将按照下述方式把任何 AHB 上的字节或半字按 32 位传送到 APB 上：

- 一个 AHB 上对地址 0x0（或 0x1、0x2 或 0x3）的写字节数据' 0xB0' 操作，将转换到 APB 上对地址 0x0 的写字数据' 0xB0B0B0B0' 操作
- 一个 AHB 上对地址 0x0（或 0x2）的写半字数据' 0xB1B0' 操作，将转换到 APB 上对地址 0x0 的写字数据' 0xB1B0B1B0' 操作

例如，如果要写入 APB 后备寄存器（与 32 位地址对齐的 16 位寄存器），需要配置存储器数据源宽度（MWIDTH）为' 16 位'，外设目标数据宽度（PWIDTH）为' 32 位'。

### 9.3.5 错误管理

读写一个保留的地址区域，将会产生 DMA 传输错误。当在 DMA 读写操作时发生 DMA 传输错误时，硬件会自动地清除发生错误的通道所对应的通道配置寄存器（DMA\_CHCTRLx）的 CHEN 位，该通道操作被停止。此时，在 DMA\_IFR 寄存器中对应该通道的传输错误中断标志位（ERRIF）将被置位，如果在 DMA\_CHCTRLx 寄存器中设置了传输错误中断允许位，则将产生中断。

### 9.3.6 中断

每个 DMA 通道都可以在 DMA 传输过半、传输完成和传输错误时产生中断。为应用的灵活性考虑，通过设置寄存器的不同位来打开这些中断。

表格 9-2 DMA 中断请求

中断事件	事件标志位	使能控制位
传输过半	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	ERRIF	ERRIE

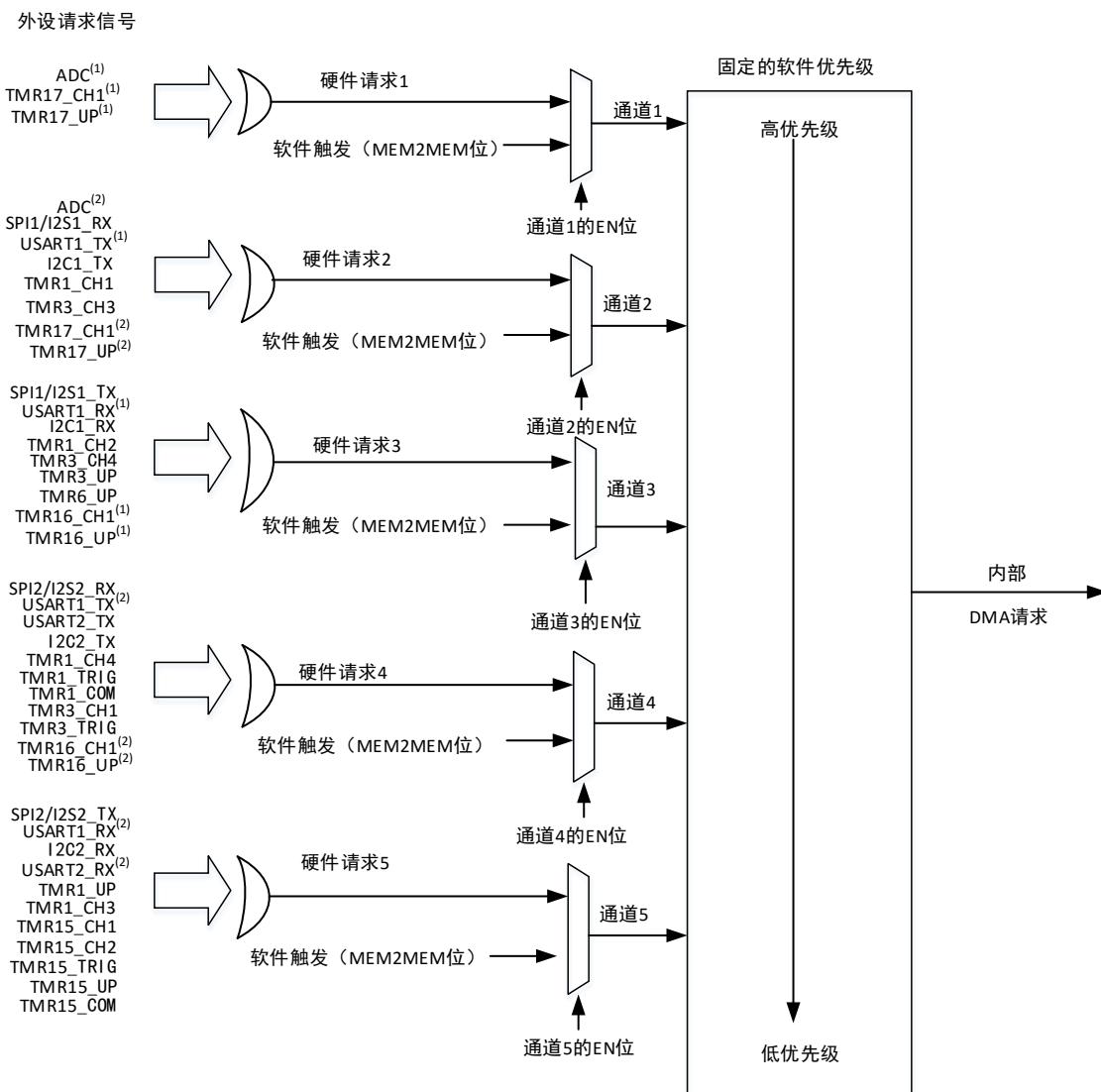
### 9.3.7 DMA 请求映像

#### DMA 控制器

从外设产生的 DMA 请求，通过逻辑或输入到 DMA 控制器，这意味着同时只能有一个请求有效。参见下图的 DMA 请求映像。

外设的 DMA 请求，可以通过设置相应外设寄存器中的控制位，被独立地开启或关闭。

图表 9-2 DMA 请求映像



表格 9-3 各个通道的DMA请求一览

外设	通道 1	通道 2	通道 3	通道 4	通道 5
ADC	ADC <sup>(1)</sup>	ADC <sup>(2)</sup>			
SPI/I2S		SPI1/I2S1_RX	SPI1/I2S1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX
USART1		USART1_TX <sup>(1)</sup>	USART1_RX <sup>(1)</sup>	USART1_TX <sup>(2)</sup>	USART1_RX <sup>(2)</sup>
USART2				USART2_TX	USART2_RX
I2C		I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX
TMR1		TMR1_CH1	TMR1_CH2	TMR1_CH4 TMR1_TRIG TMR1_COM	TMR1_CH3 TMR1_UP

<b>TMR3</b>		TMR3_CH3	TMR3_CH4 TMR3_UP	TMR3_CH1 TMR3_TRIG	
<b>TMR6</b>			TMR6_UP		
<b>TMR15</b>					TMR15_CH1 TMR15_UP TMR15_TRIG TMR15_COM TMR15_CH2
<b>TMR16</b>			TMR16_CH1 <sup>(1)</sup> TMR16_UP <sup>(1)</sup>	TMR16_CH1 <sup>(2)</sup> TMR16_UP <sup>(2)</sup>	
<b>TMR17</b>	TMR17_CH1 <sup>(1)</sup> TMR17_UP <sup>(1)</sup>	TMR17_CH1 <sup>(2)</sup> TMR17_UP <sup>(2)</sup>			

1: SYSCFG\_CFGR1中相应的remap位为0时，相应DMA请求映射到此通道

2: SYSCFG\_CFGR1中相应的remap位为1时，相应DMA请求映射到此通道

## 9.4 DMA寄存器

可以用字(8位)、半字(16位)或字(32位)的方式操作这些外设寄存器。

表格 9-4 DMA寄存器映像和复位值

020h	DMA_TCNT 2	保留	CNT[15: 0]																																									
	复位值		0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																									
024h	DMA_CPB A2	PA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
028h	DMA_CMBA A2	MA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
02Ch	保留																																											
030h	DMA_CHCTRL 3	保留	MEMTOMEM	CHPL[1: 0]		MWIDTH[1: 0]	PWIDTH[1: 0]		MINC	PINC	CIRM	DIR	ERRIE	HTIE																														
	复位值			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																					
034h	DMA_TCNT 3	保留	CNT[15: 0]																																									
	复位值		0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																									
038h	DMA_CPB A3	PA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
03Ch	DMA_CMBA A3	MA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
040h	保留																																											
044h	DMA_CHCTRL 4	保留	MEMTOMEM	CHPL[1: 0]		MWIDTH[1: 0]	PWIDTH[1: 0]		MINC	PINC	CIRM	DIR	ERRIE	HTIE																														
	复位值			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																					
048h	DMA_TCNT 4	保留	CNT[15: 0]																																									
	复位值		0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																									
04Ch	DMA_CPB A4	PA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
050h	DMA_CMBA A4	MA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
054h	保留																																											
058h	DMA_CHCTRL 5	保留	MEMTOMEM	CHPL[1: 0]		MWIDTH[1: 0]	PWIDTH[1: 0]		MINC	PINC	CIRM	DIR	ERRIE	HTIE																														
	复位值			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0			0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																					
05Ch	DMA_TCNT 5	保留	CNT[15: 0]																																									
	复位值		0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																									
060h	DMA_CPB A5	PA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										
064h	DMA_CMBA A5	MA[31: 0]																																										
	复位值	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																																										

### 9.4.1 DMA中断状态寄存器 (DMA\_ISTS)

偏移地址: 0x00

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												ERR IF5	HT IF5	TCI F5	GI F5
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR IF4	HT IF4	TC IF4	GI F4	ER RIF3	HT IF3	TC IF3	GI F3	ERR IF2	HT IF2	TC IF2	GI F2	ER RIF1	HT IF1	TC IF1	GI F1

位 31: 20	保留, 始终读为 0。
位 19, 15, 11, 7, 3	<b>ERRIFx:</b> 通道 x 的传输错误标志 ( $x = 1 \dots 5$ ) (Channel x transfer error flag) 硬件设置这些位。在 DMA_ICLR 寄存器的相应位写入'1'可以清除这里对应的标志位。 0: 在通道 x 没有传输错误 (TE) ; 1: 在通道 x 发生了传输错误 (TE) 。
位 18, 14, 10, 6, 2	<b>HTIFx:</b> 通道 x 的半传输标志 ( $x = 1 \dots 5$ ) (Channel x half transfer flag) 硬件设置这些位。在 DMA_ICLR 寄存器的相应位写入'1'可以清除这里对应的标志位。 0: 在通道 x 没有半传输事件 (HT) ; 1: 在通道 x 产生了半传输事件 (HT) 。
位 17, 13, 9, 5, 1	<b>TCIFx:</b> 通道 x 的传输完成标志 ( $x = 1 \dots 5$ ) (Channel x transfer complete flag) 硬件设置这些位。在 DMA_ICLR 寄存器的相应位写入'1'可以清除这里对应的标志位。 0: 在通道 x 没有传输完成事件 (TC) ; 1: 在通道 x 产生了传输完成事件 (TC) 。
位 16, 12, 8, 4, 0	<b>GIFx:</b> 通道 x 的全局中断标志 ( $x = 1 \dots 5$ ) (Channel x global interrupt flag) 硬件设置这些位。在 DMA_ICLR 寄存器的相应位写入'1'可以清除这里对应的标志位。 0: 在通道 x 没有 TE、HT 或 TC 事件; 1: 在通道 x 产生了 TE、HT 或 TC 事件。

### 9.4.2 DMA中断标志清除寄存器 (DMA\_ICLR)

偏移地址: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												CERRI F5	CHTI F5	CTCI F5	CGI F5
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERRI F4	CHTI F4	CTCI F4	CGI F4	CERRI F3	CHTI F3	CTCI F3	CGI F3	CERRI F2	CHTI F2	CTCI F2	CGI F2	CERRI F1	CHTI F1	CTCI F1	CGI F1

位 31: 20	保留, 始终读为 0。
位 19, 15, 11, 7, 3	<b>CERRIFx:</b> 清除通道 x 的传输错误标志 ( $x = 1 \dots 5$ ) (Channel x transfer error clear) 这些位由软件设置和清除。 0: 不起作用 1: 清除 DMA_ISTS 寄存器中的对应 ERRIF 标志。

位 18, 14, 10, 6, 2	<b>CHTIFx:</b> 清除通道 x 的半传输标志 ( $x = 1 \dots 5$ ) (Channel x half transfer clear) 这些位由软件设置和清除。 0: 不起作用 1: 清除 DMA_ISTS 寄存器中的对应 HTIF 标志。
位 17, 13 , 9, 5, 1	<b>CTCIFx:</b> 清除通道 x 的传输完成标志 ( $x = 1 \dots 5$ ) (Channel x transfer complete clear) 这些位由软件设置和清除。 0: 不起作用 1: 清除 DMA_ISTS 寄存器中的对应 TCIF 标志。
位 16, 12, 8, 4, 0	<b>CGIFx:</b> 清除通道 x 的全局中断标志 ( $x = 1 \dots 5$ ) (Channel x global interrupt clear) 这些位由软件设置和清除。 0: 不起作用 1: 清除 DMA_ISTS 寄存器中的对应的 GIF、ERRIF、HTIF 和 TCIF 标志。

### 9.4.3 DMA通道x配置寄存器 (DMA\_CHCTRLx) ( $x = 1 \dots 5$ )

偏移地址: 0x08 + 20 x (通道编号 - 1)

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	MEM TO MEM	CHPL[1:00]	MWIDTH [1: 0]	PWIDTH [1: 0]	MIN C	PINC	CIR M	DIR	ERRI E	HTIE	TCIE	CHE N			

位 31: 15	保留, 始终读为 0。
位 14	<b>MEMTOMEM:</b> 存储器到存储器模式 (Memory to memory mode) 该位由软件设置和清除。 0: 非存储器到存储器模式; 1: 启动存储器到存储器模式。
位 13: 12	<b>CHPL[1: 0]:</b> 通道优先级 (Channel priority level) 这些位由软件设置和清除。 00: 低 01: 中 10: 高 11: 最高
位 11: 10	<b>MWIDTH[1: 0]:</b> 存储器数据宽度 (Memory size) 这些位由软件设置和清除。 00: 8 位 01: 16 位 10: 32 位 11: 保留
位 9: 8	<b>PWIDTH[1: 0]:</b> 外设数据宽度 (Peripheral size) 这些位由软件设置和清除。 00: 8 位 01: 16 位 10: 32 位 11: 保留

位 7	<b>MINC:</b> 存储器地址增量模式 (Memory increment mode) 该位由软件设置和清除。 0: 不执行存储器地址增量操作 1: 执行存储器地址增量操作
位 6	<b>PINC:</b> 外设地址增量模式 (Peripheral increment mode) 该位由软件设置和清除。 0: 不执行外设地址增量操作 1: 执行外设地址增量操作
位 5	<b>CIRM:</b> 循环模式 (Circular mode) 该位由软件设置和清除。 0: 不执行循环操作 1: 执行循环操作
位 4	<b>DIR:</b> 数据传输方向 (Data transfer direction) 该位由软件设置和清除。 0: 从外设读 1: 从存储器读
位 3	<b>ERRIE:</b> 允许传输错误中断 (Transfer error interrupt enable) 该位由软件设置和清除。 0: 禁止 TE 中断 1: 允许 TE 中断
位 2	<b>HTIE:</b> 允许半传输中断 (Half transfer interrupt enable) 该位由软件设置和清除。 0: 禁止 HT 中断 1: 允许 HT 中断
位 1	<b>TCIE:</b> 允许传输完成中断 (Transfer complete interrupt enable) 该位由软件设置和清除。 0: 禁止 TC 中断 1: 允许 TC 中断
位 0	<b>CHEN:</b> 允许传输完成中断 (Transfer complete interrupt enable) 该位由软件设置和清除。 0: 禁止 TC 中断 1: 允许 TC 中断

#### 9.4.4 DMA通道x传输数量寄存器 (DMA\_TCNTx) (x = 1…5)

偏移地址: 0x0C + 20 x (通道编号 - 1)

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
保留																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
位 31: 16		CNT[15: 0]														
位 31: 16		保留, 始终读为 0。														

位 15: 0	<b>CNT[15: 0]:</b> 数据传输数量 (Number of data to transfer) 数据传输数量为 0 至 65535。这个寄存器只能在通道不工作 (DMA_CHCTRLx 的 CHEN=0) 时写入。通道开启后该寄存器变为只读，指示剩余的待传输字节数目。寄存器内容在每次 DMA 传输后递减。 数据传输结束后，寄存器的内容或者变为 0；或者当该通道配置为自动重加载模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当寄存器的内容为 0 时，无论通道是否开启，都不会发生任何数据传输。
---------	--

#### 9.4.5 DMA通道x外设地址寄存器 (DMA\_CPBAX) (x = 1…5)

偏移地址: 0x10 + 20 x (通道编号 - 1)

复位值: 0x0000 0000

当开启通道 (DMA\_CHCTRLx 的 CHEN=1) 时不能写该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PA[31: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PA[31: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 0	<b>PA[31: 0]:</b> 外设地址 (Peripheral address) 外设数据寄存器的基地址，作为数据传输的源或目标。 当 PWIDHT='01' (16 位)，不使用 PA[0]位。操作自动地与半字地址对齐。 当 PWIDHT='10' (32 位)，不使用 PA[1: 0]位。操作自动地与字地址对齐。
---------	---

#### 9.4.6 DMA通道x存储器地址寄存器 (DMA\_CMBAx) (x = 1…5)

偏移地址: 0x14 + 20 x (通道编号 - 1)

复位值: 0x0000 0000

当开启通道 (DMA\_CHCTRLx 的 CHEN=1) 时不能写该寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MA[31: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MA[31: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 0	<b>MA[31: 0]:</b> 储存器地址 储存器地址作为数据传输的源或目标。 当 MWIDHT='01' (16 位)，不使用 MA[0]位。操作自动地与半字地址对齐。 当 MWIDHT='10' (32 位)，不使用 MA[1: 0]位。操作自动地与字地址对齐。
---------	--

# 10 定时器 (TIMER)

## 10.1 基本定时器 (TMR6)

### 10.1.1 TRM6简介

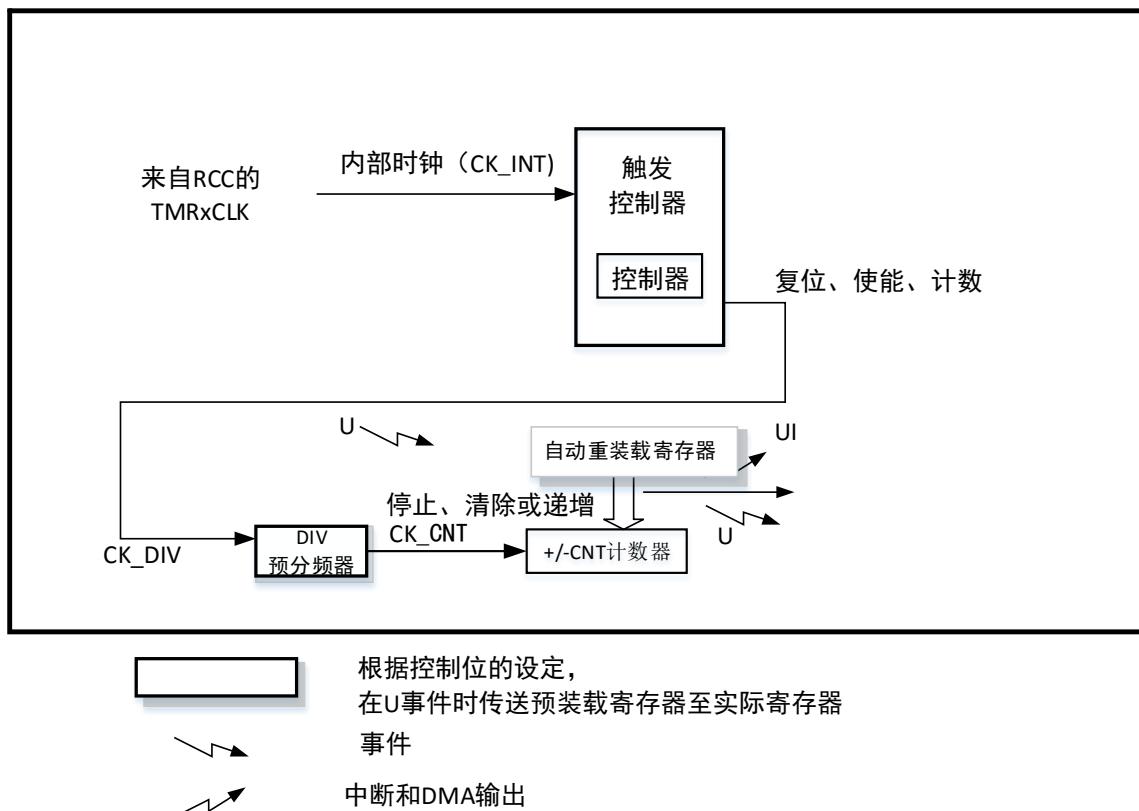
基本定时器 TMR6 包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。它们可以作为通用定时器提供时间基准。

### 10.1.2 TRM6的主要特性

TMR6 定时器的主要功能包括：

- 16位自动重装载累加计数器
- 16位可编程（可实时修改）预分频器，用于对输入时钟按系数为1~65536之间的任意数值分频
- 在更新事件（计数器溢出）时产生中断/DMA请求

图表 10-1 基本定时器框图



### 10.1.3 TRM6的功能

#### 10.1.3.1 时基单元

这个可编程定时器的主要部分是一个带有自动重装载的 16 位累加计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，即使计数器运行时也可以操作。

时基单元包含：

- 计数器寄存器 (TMRx\_CNT)

- 预分频寄存器 (TMRx\_DIV)
- 自动重装载寄存器 (TMRx\_AR)

自动重装载寄存器是预加载的，每次读写自动重装载寄存器时，实际上是通过读写预加载寄存器实现。根据 TMRx\_CTRL1 寄存器中的自动重装载预加载使能位 (ARPEN)，写入预加载寄存器的内容能够立即或在每次更新事件时，传送到它的影子寄存器。当 TMRx\_CTRL1 寄存器的 UEVDIS 位为 '0'，则每当计数器达到溢出值时，硬件发出更新事件；软件也可以产生更新事件；关于更新事件的产生，随后会有详细的介绍。

计数器由预分频输出 CK\_CNT 驱动，设置 TMRx\_CTRL1 寄存器中的计数器使能位 (CNTEN) 使能计数器计数。

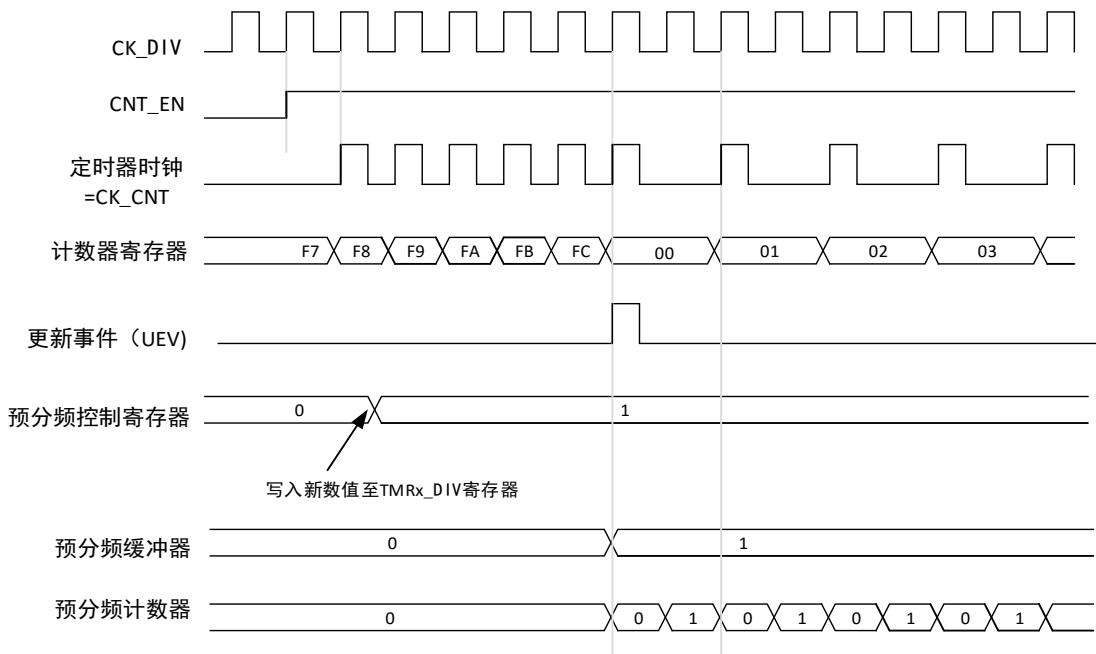
注意： 实际的设置计数器使能信号 CNT\_EN 相对于 CNTEN 滞后一个时钟周期。

### 10.1.3.2 预分频器

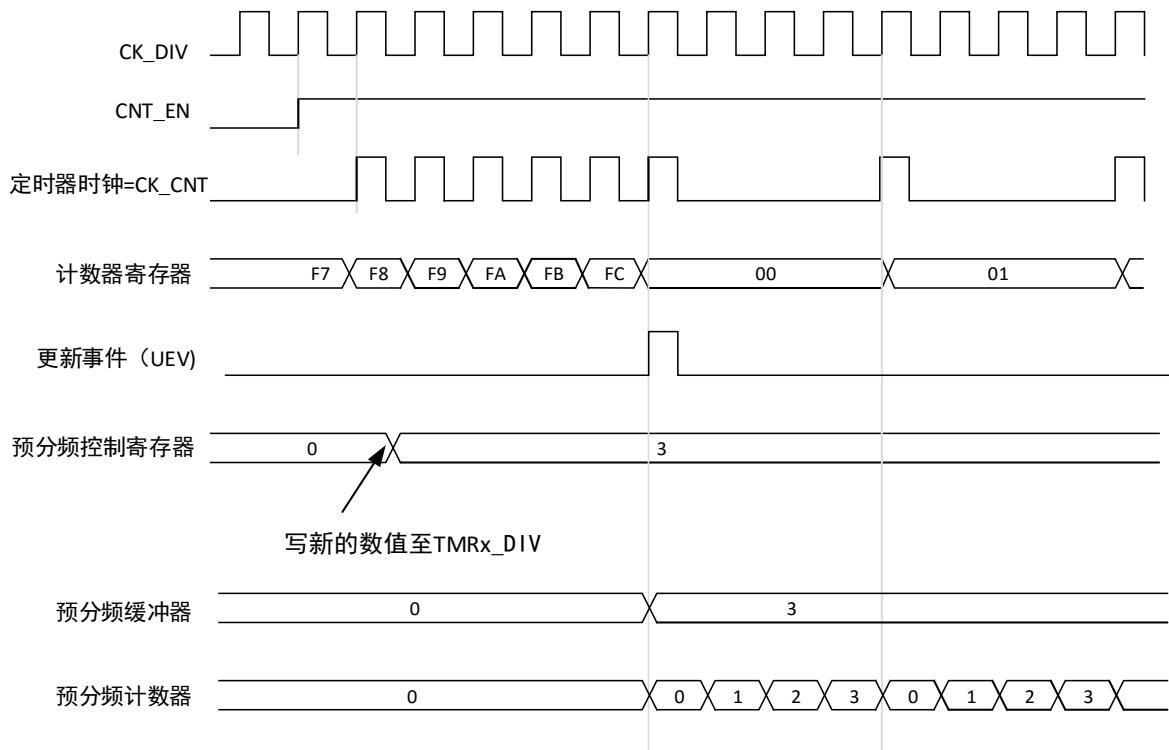
预分频可以以系数介于 1 至 65536 之间的任意数值对计数器时钟分频。它是通过一个 16 位寄存器 (TMRx\_DIV) 的计数实现分频。因为 TMRx\_DIV 控制寄存器具有缓冲，可以在运行过程中改变它的数值，新的预分频数值将在下一个更新事件时起作用。

以下两图是在运行过程中改变预分频系数的例子。

图表 10-2 预分频系数从1变到2的计数器时序图



图表 10-3 预分频系数从1变到4的计数器时序图



### 10.1.3.3 计数模式

计数器从 0 累加计数到自动重装载数值（TMR<sub>x</sub>\_AR 寄存器），然后重新从 0 开始计数并产生一个计数器溢出事件。

每次计数器溢出时可以产生更新事件；（通过软件或使用从模式控制器）设置 TMR<sub>x</sub>\_EVEG 寄存器的 UEVG 位也可以产生更新事件。

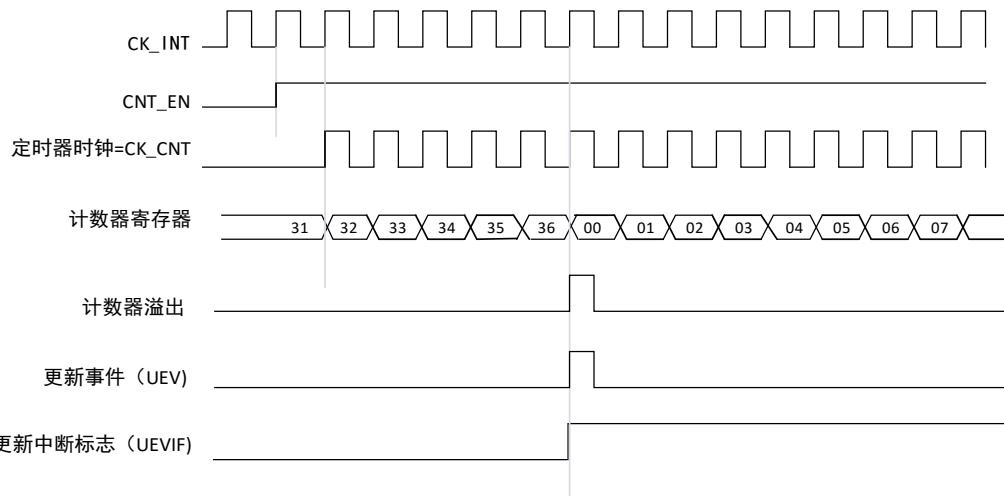
设置 TMR<sub>x</sub>\_CTRL1 中的 UEVDIS 位可以禁止产生 UEV 事件，这可以避免在写入预加载寄存器时更改影子寄存器。在清除 UEVDIS 位为‘0’之前，将不再产生更新事件，但计数器和预分频器依然会在应产生更新事件时重新从 0 开始计数(但预分频系数不变)。另外，如果设置了 TMR<sub>x</sub>\_CTRL1 寄存器中的 UVERS (选择更新请求)，设置 UEVG 位可以产生一次更新事件 UEV，但不设置 UEVIF 标志 (即没有中断或 DMA 请求)。

当发生一次更新事件时，所有寄存器会被更新并（根据 UVERS 位）设置更新标志（TMR<sub>x</sub>\_STS 寄存器的 UEVIF 位）：

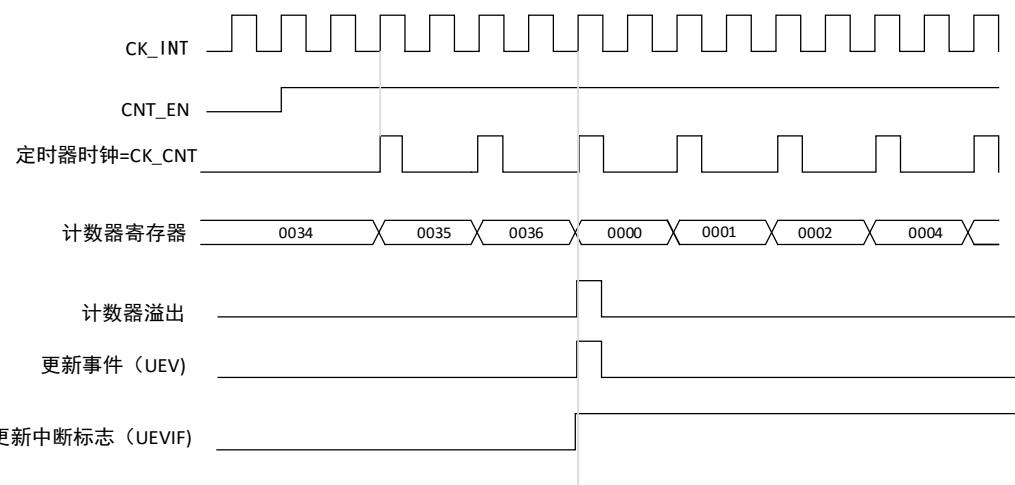
- 传送预装载值（TMR<sub>x</sub>\_DIV 寄存器的内容）至预分频器的缓冲区。
- 自动重装载影子寄存器被更新为预装载值（TMR<sub>x</sub>\_AR）。

以下是一些在 TMR<sub>x</sub>\_AR=0x36 时不同时钟频率下计数器工作的图示例子。

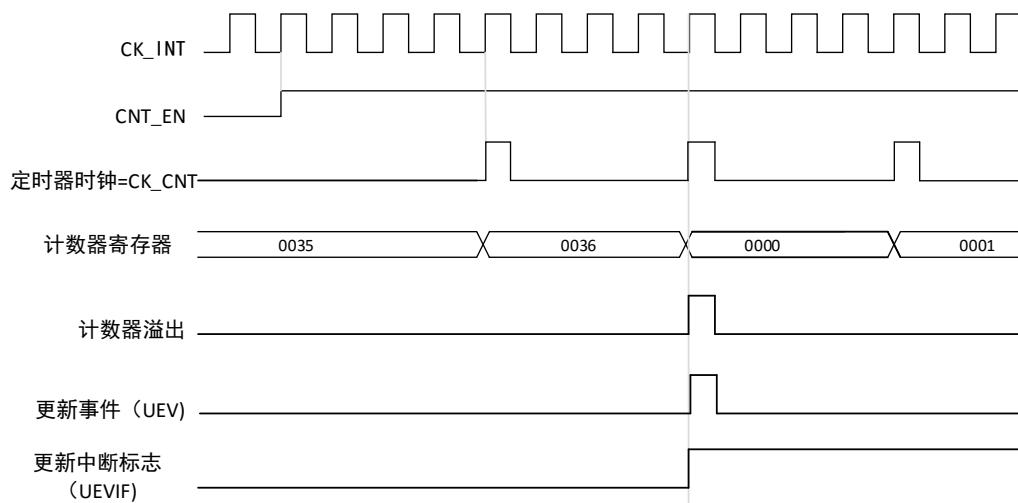
图表 10-4 计数器时序图，内部时钟分频系数为 1



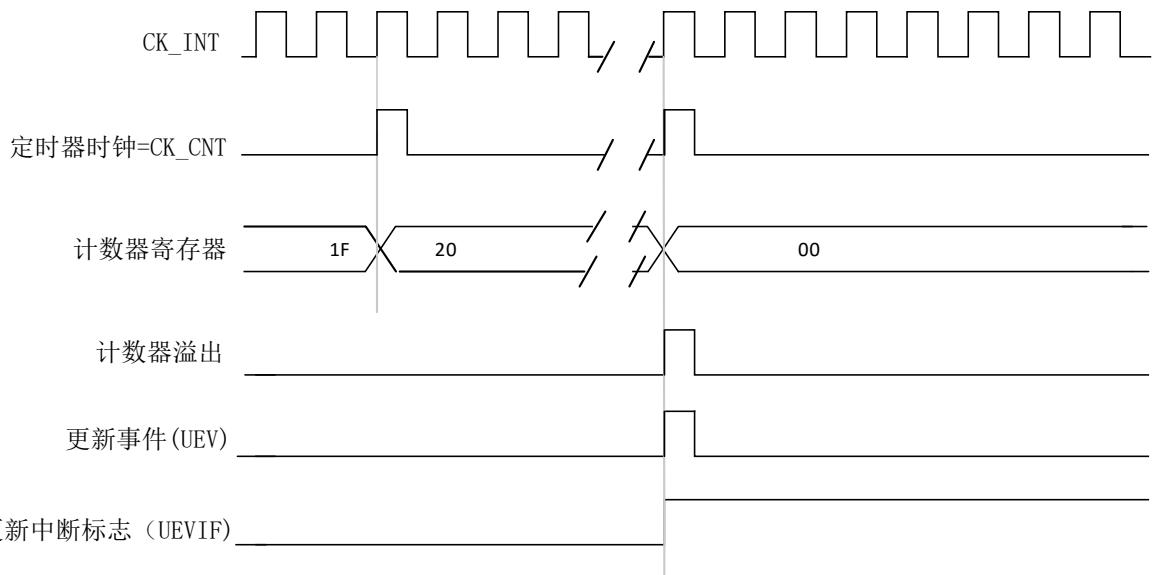
图表 10-5 计数器时序图, 内部时钟分频系数为2



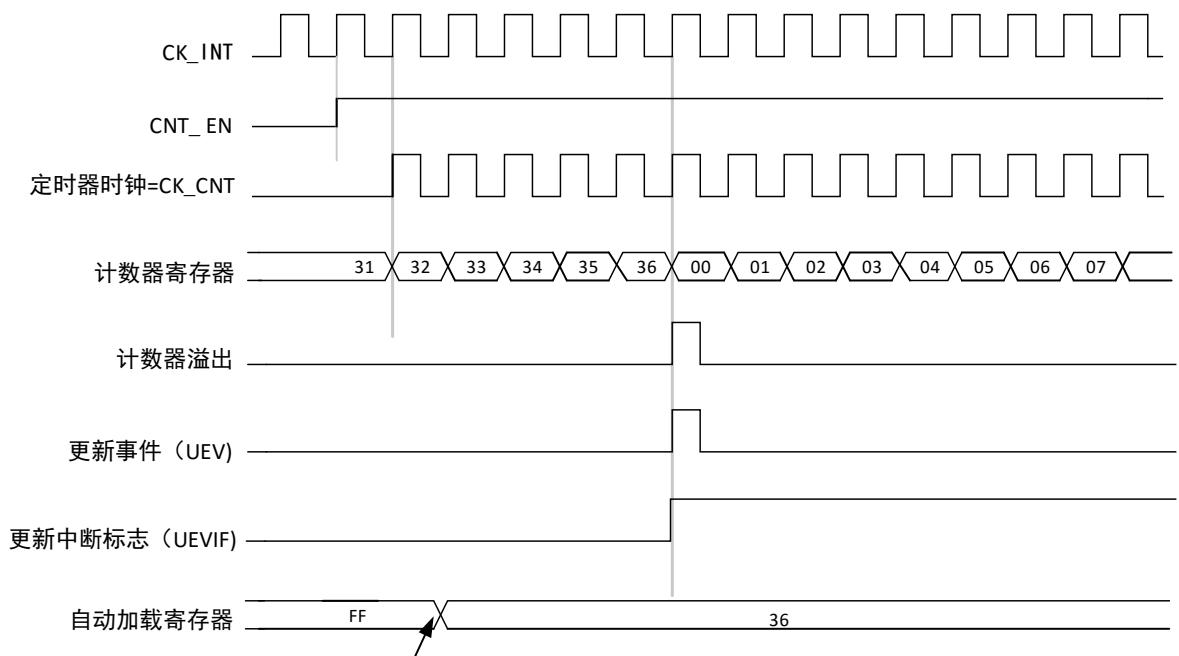
图表 10-6 计数器时序图, 内部时钟分频系数为4



图表 10-7 计数器时序图, 内部时钟分频系数为N

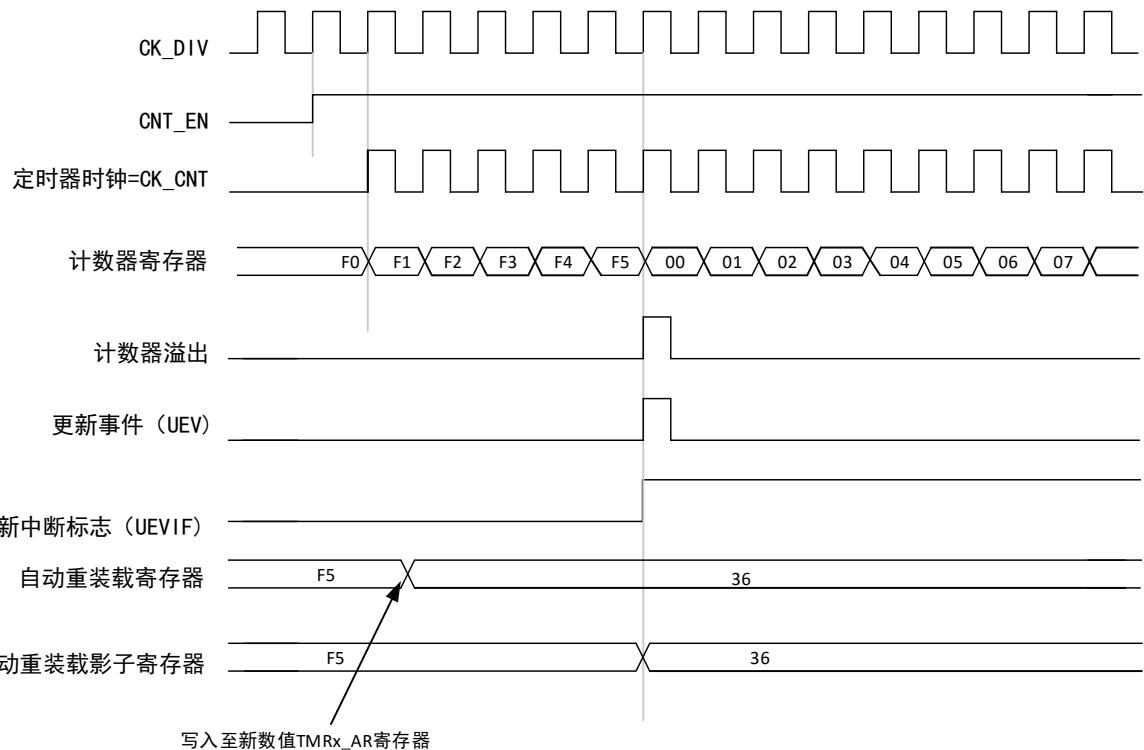


图表 10-8 计数器时序图，当ARPEN=0时的更新事件（TMRx\_AR没有预装载）



在TMRx\_AR写入新数值

图表 10-9 计数器时序图，当ARPEN=1时的更新事件（预装载TMRx\_AR）

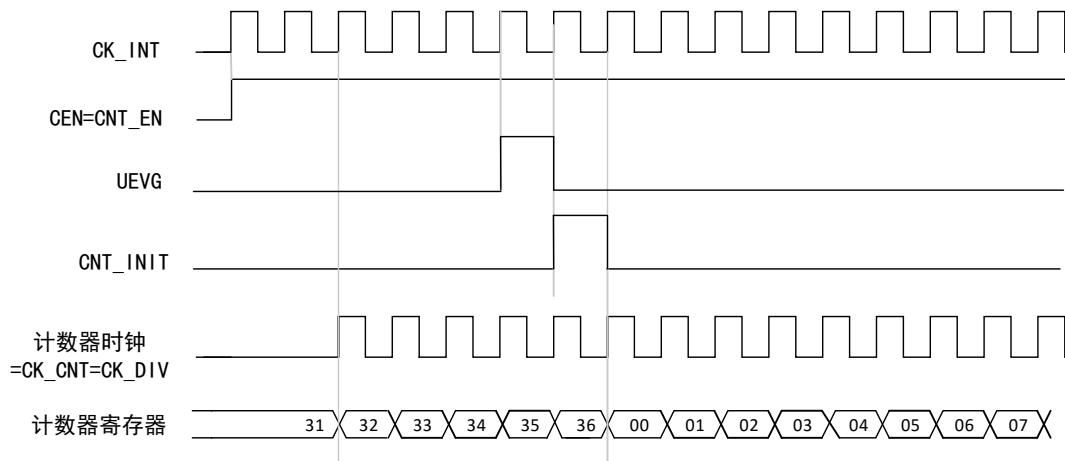


#### 10.1.3.4 时钟源

计数器的时钟由内部时钟 (CK\_INT) 提供。

TMRx\_CTRL1 寄存器的 CNTEN 位和 TMRx\_EVEG 寄存器的 UEVG 位是实际的控制位，(除了 UEVG 位被自动清除外) 只能通过软件改变它们。一旦置 CNTEN 位为'1'，内部时钟即向预分频器提供时钟。下图示出控制电路和向上计数器在普通模式下，没有预分频器时的操作。

图表 10-10 普通模式时序图，内部时钟分频系数为1



#### 10.1.3.5 调试模式

当微控制器进入调试模式 (Cortex™-M4 核心停止) 时，根据 DBG 模块中的配置位 DBG\_TMRx\_STOP 的设置，TMRx 计数器或者继续计数或者停止工作。详见[第 18.2.2 节](#)。

#### 10.1.4 TRM6 寄存器

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

下表中将 TMRx 的所有寄存器映射到一个 16 位可寻址（编址）空间。

表格 10-1 TMR6 寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
0x00	TMRx_CTRL1	保留																ARPEN	保留			UVRS	UEVDIS	CNTEN	0	0	0	0	0									
	复位值																	0				0	0	0	0													
0x04	TMRx_CTRL2	保留																MMSEL[2: 0]	保留			保留						保留										
	复位值																	0																				
0x0C	TMRx_DIE	保留																UEVDE	保留			保留						保留										
	复位值																	0																				
0x10	TMRx_STS	保留																0	保留			保留						保留										
	复位值																	0																				
0x14	TMRx_EVEG	保留																0	保留			保留						保留										
	复位值																	0																				
0x24	TMRx_CNT	保留								CNT[15: 0]																保留												
	复位值									0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												
0x28	TMRx_DIV	保留								DIV[15: 0]																保留												
	复位值									0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												
0x2C	TMRx_AR	保留								AR[15: 0]																保留												
	复位值									0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																												

#### 10.1.4.1 TMR6控制寄存器1 (TMRx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
保留								ARPEN	保留								OPMODE	UVERS		UEVDIS		CNTEN	

位 15: 8	保留, 始终读为 0。
位 7	<b>ARPEN:</b> 自动重装载预装载使能 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲 1: TMRx_AR 寄存器具有缓冲
位 6: 4	保留, 始终读为 0。
位 3	<b>OPMODE:</b> 单脉冲模式 (One-pulse mode) 0: 在发生更新事件时, 计数器不停止 1: 在发生下次更新事件时, 计数器停止计数 (清除 CNTEN 位)。

位 2	<b>UVERS:</b> 更新请求源 (Update request source) 该位由软件设置和清除, 以选择 UEV 事件的请求源。 0: 如果使能了中断或 DMA, 以下任一事件可以产生一个更新中断或 DMA 请求: - 计数器溢出 - 设置 UEVG 位 - 通过从模式控制器产生的更新 1: 如果使能了中断或 DMA, 只有计数器上溢或下溢可以产生更新中断或 DMA 请求。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 该位由软件设置和清除, 以使能或禁止 UEV 事件的产生。 0: UEV 使能。更新事件 (UEV) 可以由下列事件产生: - 计数器溢出 - 设置 UEVG 位 - 通过从模式控制器产生的更新产生更新事件后, 带缓冲的寄存器被加载为预加载数值。 1: 禁止 UEV。不产生更新事件 (UEV), 影子寄存器保持它的内容 (AR、DIV)。但是如果设置了 UEVG 位或从模式控制器产生了一个硬件复位, 则计数器和预分频器将被重新初始化。
位 0	<b>CNTEN:</b> 计数器使能 (Counter enable) 0: 关闭计数器 1: 使能计数器 注: 门控模式只能在软件已经设置了 CNTEN 位时有效, 而触发模式可以自动地由硬件设置 CNTEN 位。在单脉冲模式下, 当产生更新事件时 CNTEN 被自动清除。

### 10.1.4.2 TMR6控制寄存器2 (TMRx\_CTRL2)

偏移地址: 0x04

复位值: 0x0000

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

位 15: 7	保留, 始终读为 0。
位 6: 4	<b>MMSEL:</b> 主模式选择 (LENaster mode selection) 这些位用于选择在主模式下向从定时器发送的同步信息 (TRGO), 有以下几种组合: 000: 复位 - 使用 TMRx_EVEG 寄存器的 UEVG 位作为触发输出 (TRGO)。如果触发输入产生了复位 (从模式控制器配置为复位模式), 则相对于实际的复位信号, TRGO 上的信号有一定的延迟。 001: 使能 - 计数器使能信号 CNT_EN 被用作为触发输出 (TRGO)。它可用于在同一时刻启动多个定时器, 或控制使能从定时器的时机。计数器使能信号是通过 CNTEN 控制位和配置为门控模式时的触发输入的逻辑或'产生。 当计数器使能信号是通过触发输入控制时, 在 TRGO 输出上会有一些延迟, 除非选择了主/从模式 (见 TMRx_SMC 寄存器的 MSMODE 位)。 010: 更新 - 更新事件被用作为触发输出 (TRGO)。例如一个主定时器可以作为从定时器的预分频器使用。
位 3: 0	保留, 始终读为 0。

### 10.1.4.3 TMR6 DMA中断使能寄存器 (TMRx\_DIE)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				UEVD E		保留				保留				UEV IE	
res				rw		res				rw					

位 15: 9	保留, 始终读为 0。
位 8	<b>UEVDE:</b> 更新 DMA 请求使能 (Update DMA request enable) 0: 禁止更新 DMA 请求 1: 使能更新 DMA 请求
位 7: 1	保留, 始终读为 0。
位 0	<b>UEVIE:</b> 更新中断使能 (Update interrupt enable) 0: 禁止更新中断 1: 使能更新中断

#### 10.1.4.4 TMR6状态寄存器 (TMRx\_STS)

偏移地址: 0x10

复位值: 0x0000

位 15: 1	保留, 始终读为 0。
位 0	<p><b>UEVIF:</b> 更新中断标志 (Update interrupt flag) 硬件在更新中断时设置该位, 它由软件清除。</p> <p>0: 没有产生更新。 1: 产生了更新中断。下述情况下由硬件设置该位:</p> <ul style="list-style-type: none"> <li>- 计数器产生溢出并且 TMRx_CTRL1 中的 UEVDIS=0;</li> <li>- 如果 TMRx_CTRL1 中的 UVERS=0 并且 UEVDIS=0, 置位 UEVG (TMRx_EVEG) 位重新初始化计数器时。</li> </ul>

#### 10.1.4.5 TMR6事件产生寄存器（TMRx\_EVEG）

偏移地址: 0x14

复位值: 0x0000

位 15: 1	保留, 始终读为 0。
位 0	<p><b>UEVG:</b> 产生更新事件 (Update generation)      该位由软件设置, 由硬件自动清除。</p> <p>0: 无作用      1: 重新初始化定时器的计数器并产生对寄存器的更新。注意: 预分频器也被清除 (但预分频系数不变)。</p>

#### 10.1.4.6 TMR6计数器 (TMRx\_CNT)

偏移地址: 0x24

复位值: 0x0000

位 15: 0

CNT[15: 0]: 计数器数值 (Counter value)

### 10.1.4.7 TMR6预分频器 (TMRx\_DIV)

偏移地址: 0x28

复位值: 0x0000

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

位 15: 0

DIV[15: 0]: 预分频器数值 (Prescaler value)

计数器的时钟频率 CK\_CNT 等于 fCK\_DIV / (DIV[15: 0]+1)。

在每一次更新事件时, DIV 的数值被传送到实际的预分频寄存器中。

### 10.1.4.8 TMR6自动重装载寄存器 (TMRx\_AR)

偏移地址: 0x2C

复位值: 0x0000

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

位 15: 0

AR[15: 0]: 自动重装载数值 (Auto reload value)

AR 的数值将传送到实际的自动重装载寄存器中。关于 AR 的更新和作用, 详见 [10.1.3.1 节](#)。

如果自动重装载数值为 0, 则计数器停止。

## 10.2 通用定时器 (TMR3)

### 10.2.1 TMRx简介

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

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

每个定时器都是完全独立的, 没有互相共享任何资源。它们可以一起同步操作, 参见 [10.2.3.15 节](#)。

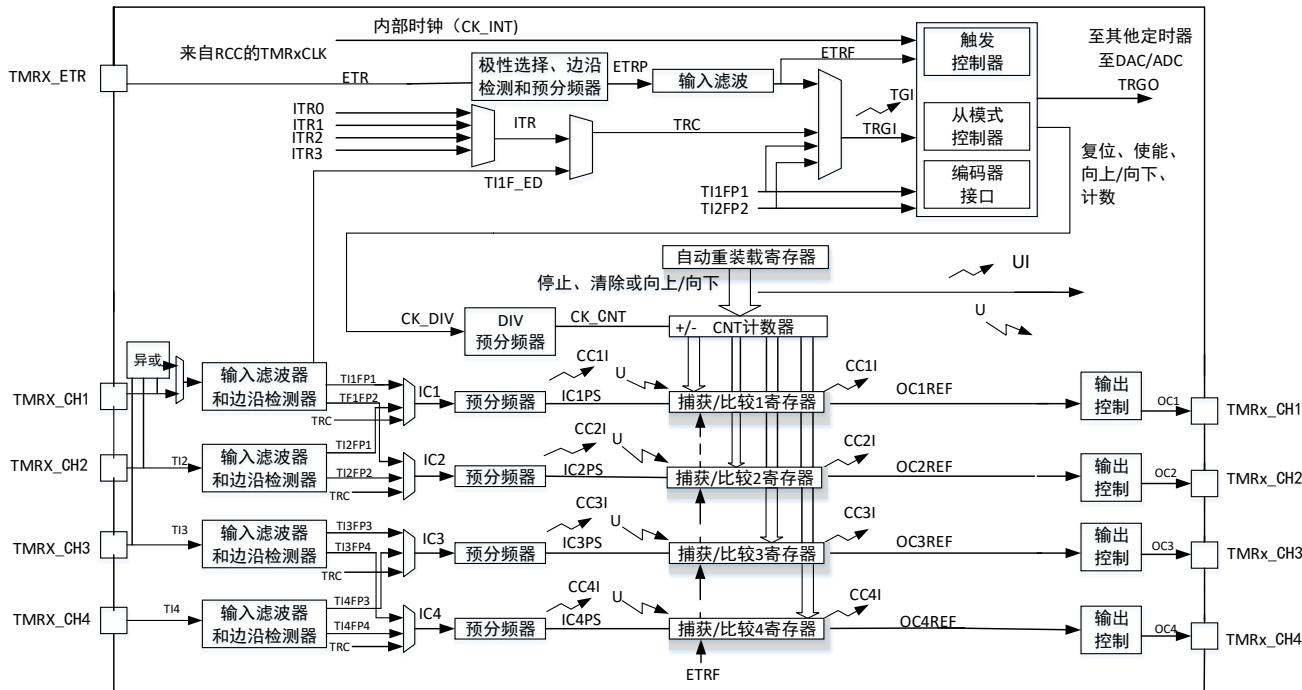
### 10.2.2 TMRx主要功能

通用 TMRx (TMR3) 定时器功能包括:

- 16位向上、向下、向上/向下自动装载计数器
- 16位可编程 (可以实时修改) 预分频器, 计数器时钟频率的分频系数为 1~65536 之间的任意数值
- 4个独立通道:
  - 输入捕获
  - 输出比较
  - PWM生成 (边缘或中间对齐模式)
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互连的同步电路

- 如下事件发生时产生中断/DMA:
  - 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
- 支持针对定位的增量（正交）编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

图表 10-11 通用定时器框图



注意：  
Reg 根据控制位的设定，在 **U** 事件时传送预加载寄存器的内容至工作寄存器  
 ↗ 事件  
 ↗ 中断和 DMA 输出

### 10.2.3 TMRx 功能描述

#### 10.2.3.1 时基单元

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

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

时基单元包含：

- 计数器寄存器 (TMRx\_CNT)
- 预分频器寄存器 (TMRx\_DIV)
- 自动装载寄存器 (TMRx\_AR)

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TMRx\_CTRL1 寄存器中的自动装载预装载使能位 (ARPEN) 的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TMRx\_CTRL1 寄存器中的 UEVDIS 位等于 '0' 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK\_CNT 驱动，仅当设置了计数器 TMRx\_CTRL1 寄存器中的计数器使能位 (CNTEN) 时，CK\_CNT 才有效。(有关计数器使能的细节，请参见控制器的从模式描述)。

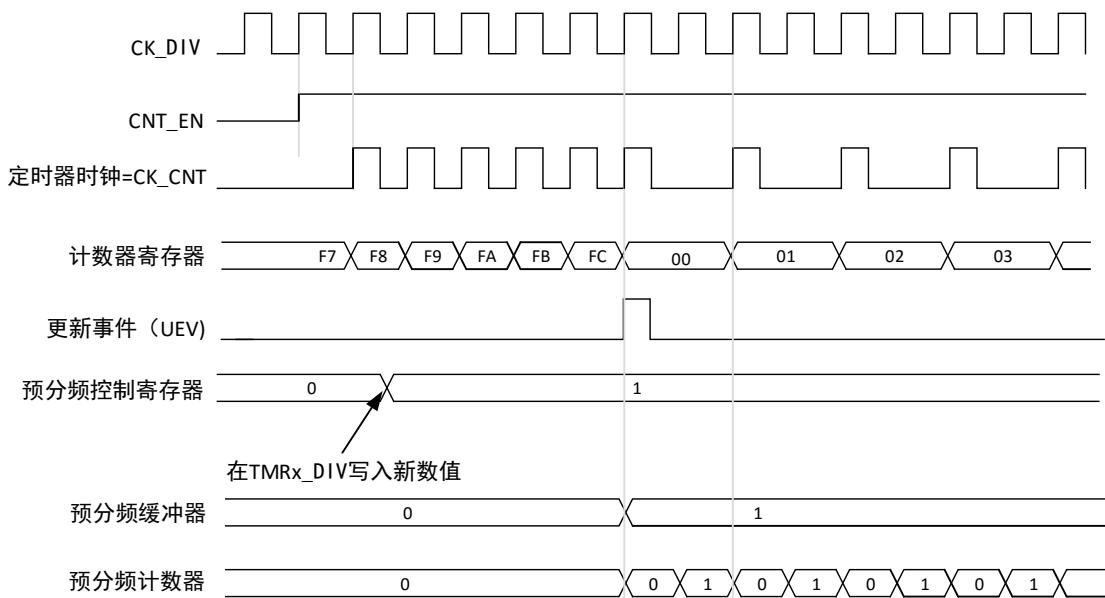
**注意：** 真正的计数器使能信号 CNT\_EN 是在 CNTEN 的一个时钟周期后被设置。

### 预分频器

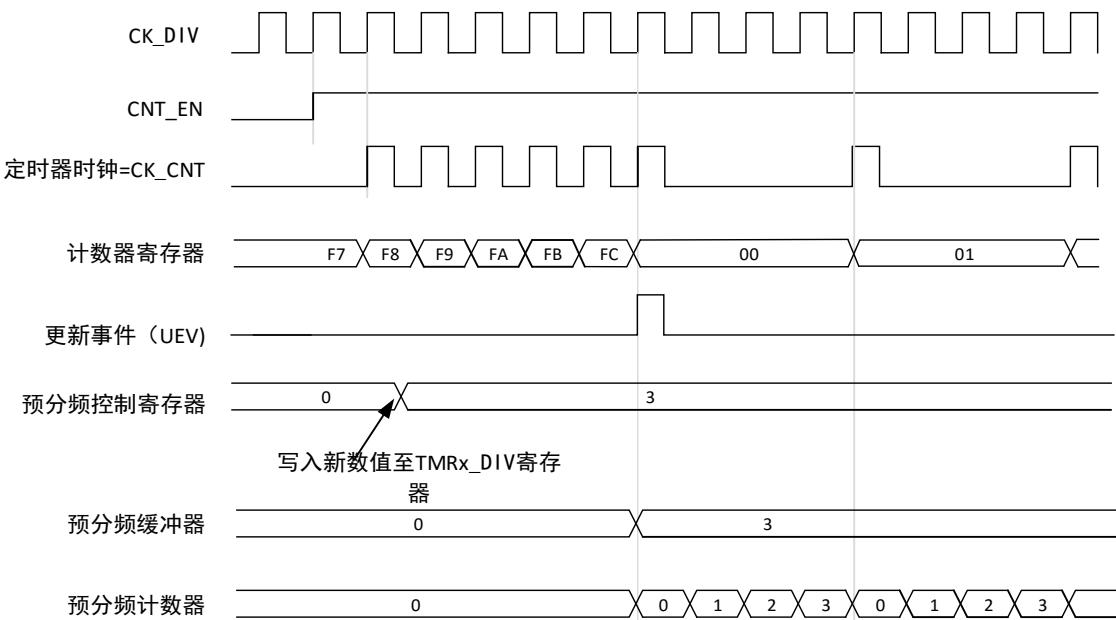
预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个 (在 TMRx\_DIV 寄存器中的) 16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器参数在下一次更新事件到来时被采用。

[图 10-12](#) 和 [图 10-13](#) 给出了在预分频器运行时，更改计数器参数的例子。

图表 10-12 当预分频器的参数从1变到2时，计数器的时序图



图表 10-13 当预分频器的参数从1变到4时，计数器的时序图



### 10.2.3.2 计数器模式

#### 向上计数模式

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

每次计数器溢出时可以产生更新事件，在 **TMRx\_EVEG** 寄存器中（通过软件方式或者使用从模式控制器）设置 **UEVG** 位也同样可以产生一个更新事件。

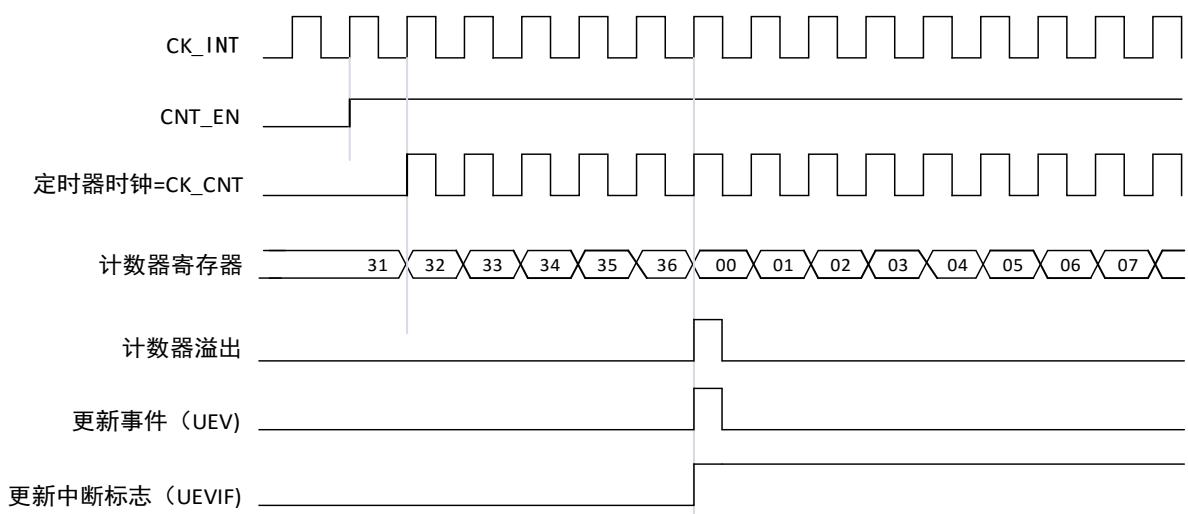
设置 **TMRx\_CTRL1** 寄存器中的 **UEVDIS** 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 **UEVDIS** 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清 0（但预分频系数不变）。此外，如果设置了 **TMRx\_CTRL1** 寄存器中的 **UEVRS** 位（选择更新请求），设置 **UEVG** 位将产生一个更新事件 **UEV**，但硬件不设置 **UEVIF** 标志（即不产生中断或 DMA 请求）；这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 **UEVRS** 位）设置更新标志位（**TMRx\_STS** 寄存器中的 **UEVIF** 位）。

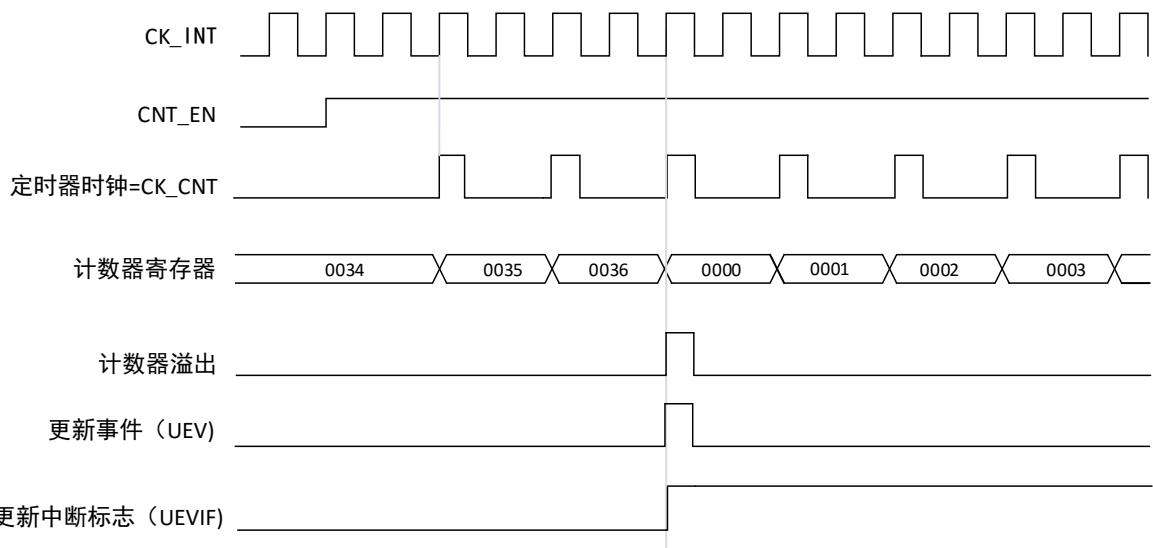
- 预分频器的缓冲区被置入预装载寄存器的值（**TMRx\_DIV** 寄存器的内容）。
- 自动装载影子寄存器被重新置入预装载寄存器的值（**TMRx\_AR**）。

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

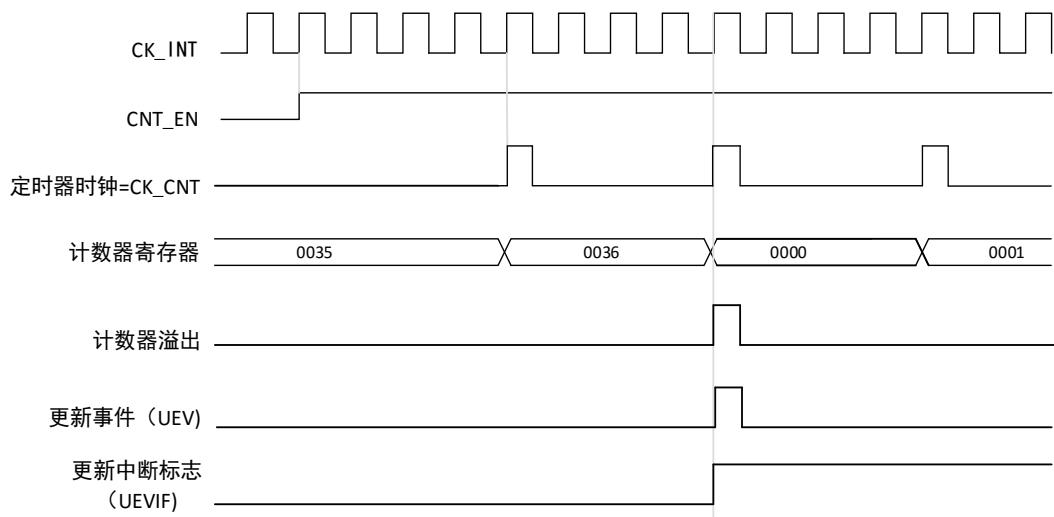
图表 10-14 计数器时序图，内部时钟分频因子为 1



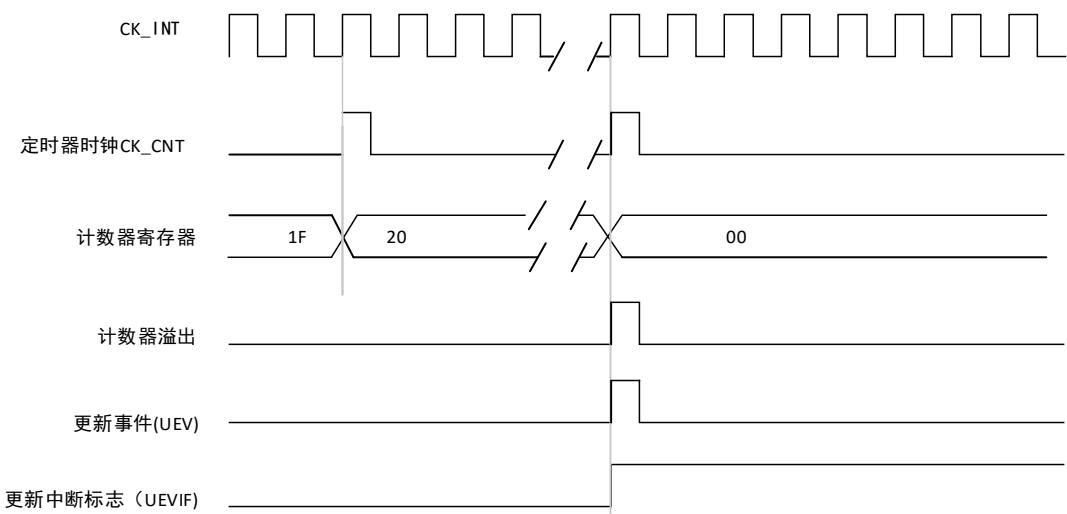
图表 10-15 计数器时序图，内部时钟分频因子为2



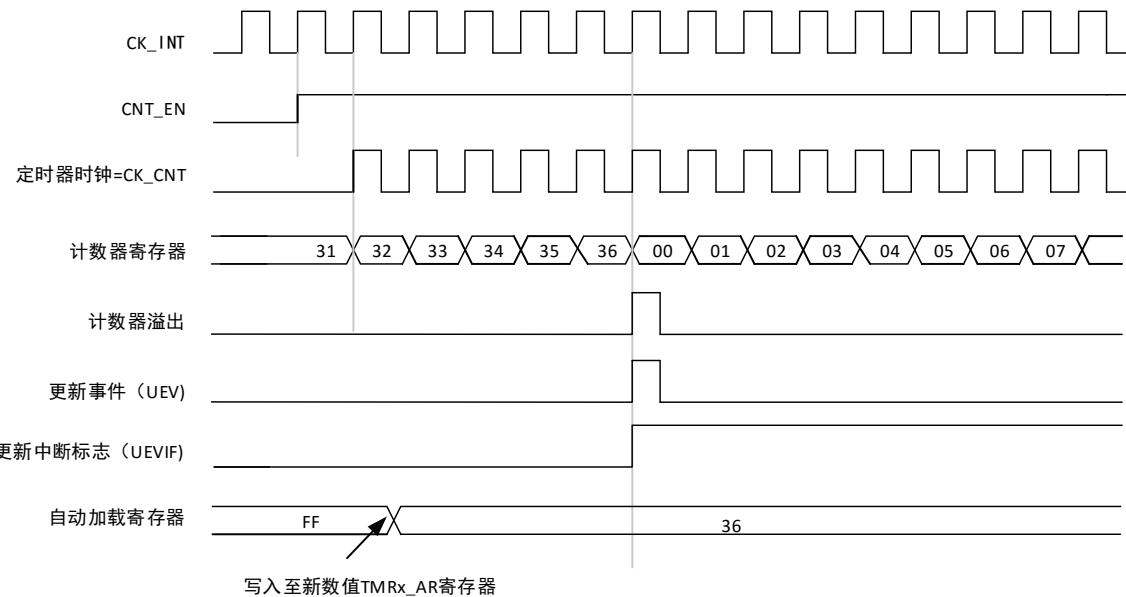
图表 10-16 计数器时序图，内部时钟分频因子为4



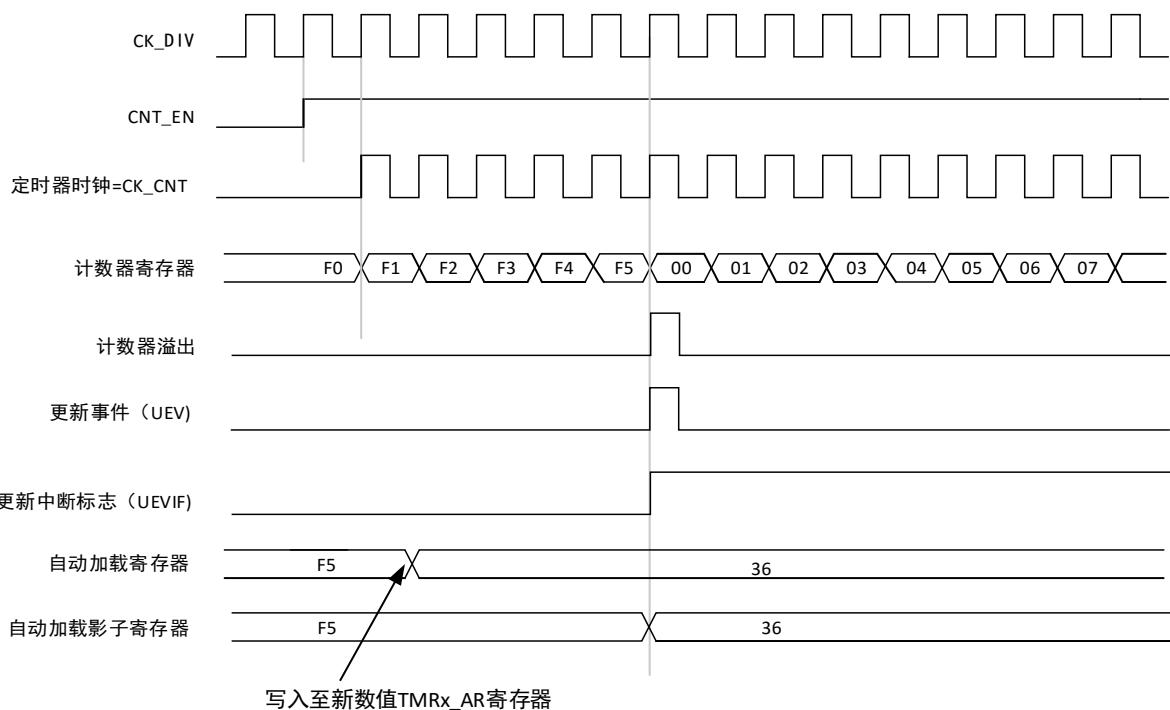
图表 10-17 计数器时序图，内部时钟分频因子为N



图表 10-18 计数器时序图, 当ARPEN=0时的更新事件 (没有预装入了TMRx\_AR)



图表 10-19 计数器时序图, 当ARPEN=1时的更新事件 (预装入了TMRx\_AR)



### 向下计数模式

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

每次计数器溢出时可以产生更新事件, 在 TMRx\_EVEG 寄存器中(通过软件方式或者使用从模式控制器)设置 UEVG 位, 也同样可以产生一个更新事件。

设置 TMRx\_CTRL1 寄存器的 UEVDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UEVDIS 位被清为'0'之前不会产生更新事件。然而, 计数器仍会从当前自动加载值重新开始计数, 同时预分频器的计数器重新从 0 开始 (但预分频系数不变)。此外, 如果设置了 TMRx\_CTRL1 寄存器中的 UEVRS 位 (选择更新请求), 设置 UEVG 位将产生一个更新事件 UEV 但不设置 UEVIF 标志 (因此不产生中断和 DMA 请求), 这是为了避免在发生捕获事件并清除计数器时, 同时产生更新和捕获中断。

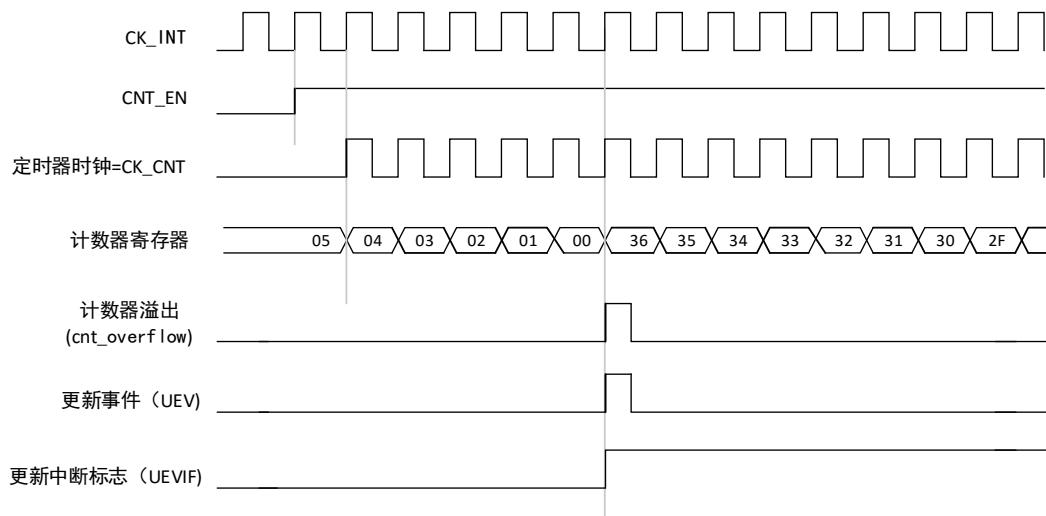
当发生更新事件时，所有的寄存器都被更新，并且（根据 UEVRS 位的设置）更新标志位（TMRx\_STS 寄存器中的 UEVIF 位）也被设置。

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

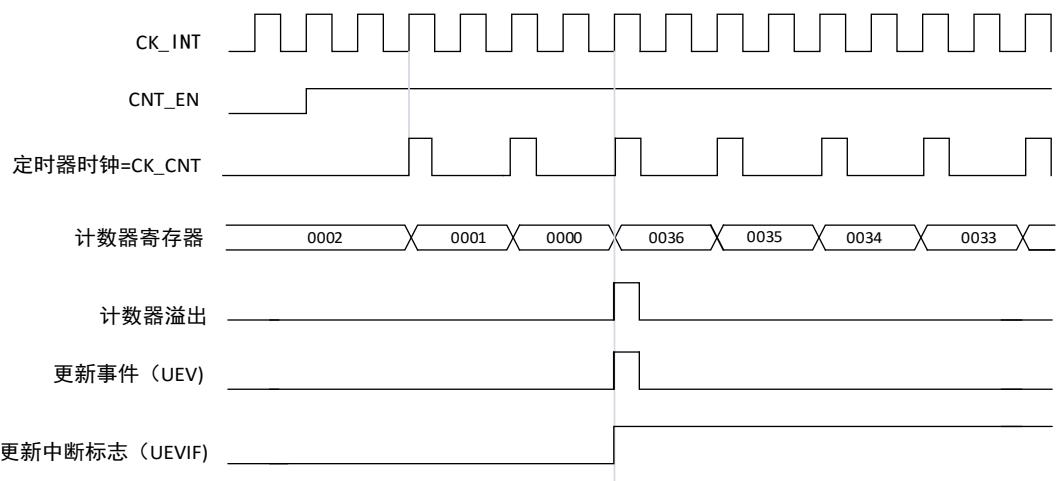
**注意：** 自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

以下是一些当 TMRx\_AR=0x36 时，计数器在不同时钟频率下的操作例子。

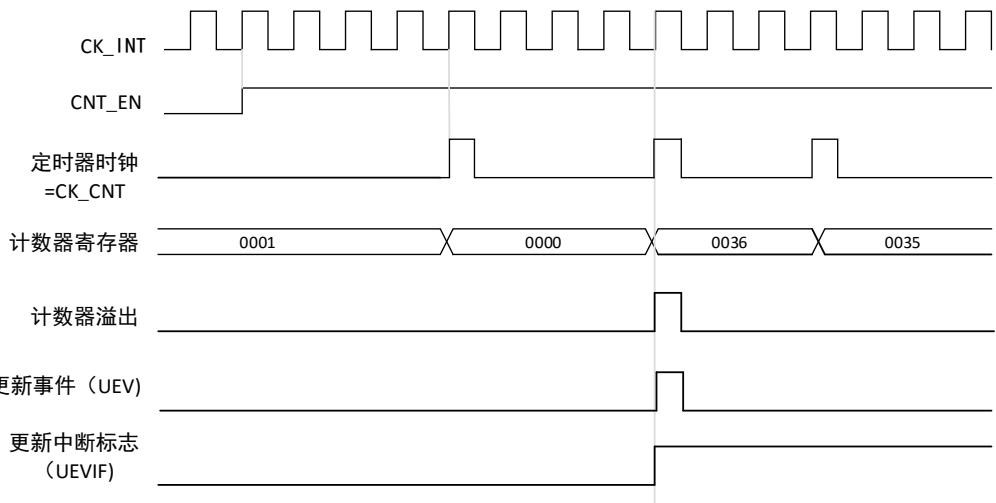
图表 10-20 计数器时序图，内部时钟分频因子为1



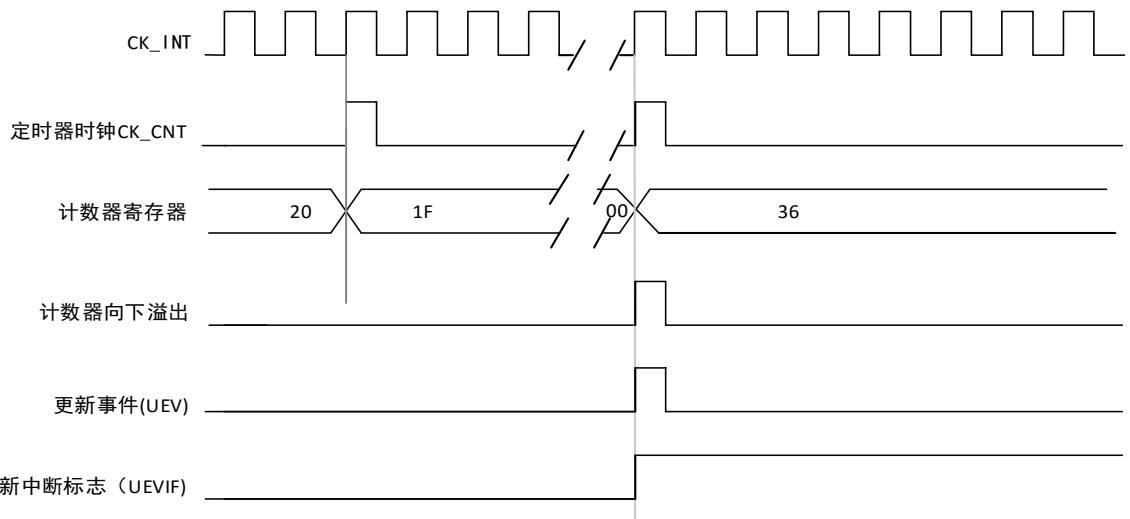
图表 10-21 计数器时序图，内部时钟分频因子为2



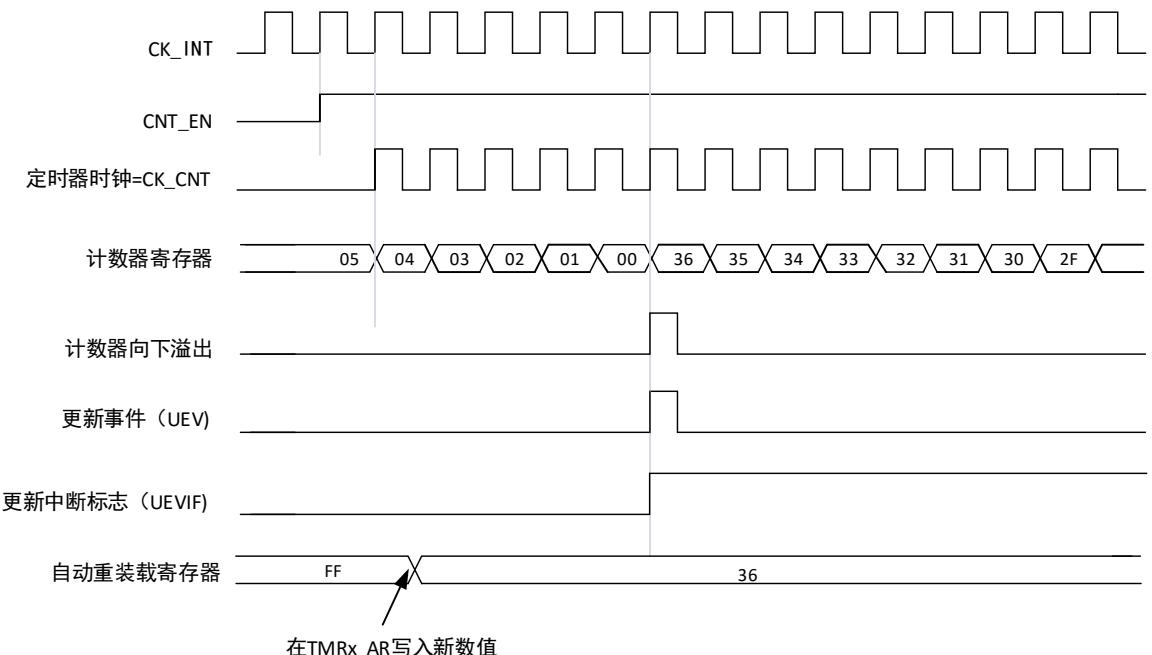
图表 10-22 计数器时序图，内部时钟分频因子为4



图表 10-23 计数器时序图，内部时钟分频因子为N



图表 10-24 计数器时序图，当ARPEN=0时的更新事件



### 中央对齐模式（向上/向下计数）

在中央对齐模式，计数器从 0 开始计数到自动加载的值 (TMRx\_AR 寄存器) -1，产生一个计数器溢出事

件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在这个模式，不能写入 TMRx\_CTRL1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

设置 TMRx\_CTRL1 寄存器中的 UEVDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UEVDIS 位被清为'0'之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。此外，如果设置了 TMRx\_CTRL1 寄存器中的 UEVRS 位（选择更新请求），设置 UEVG 位将产生一个更新事件 UEV 但不设置 UEVIF 标志（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

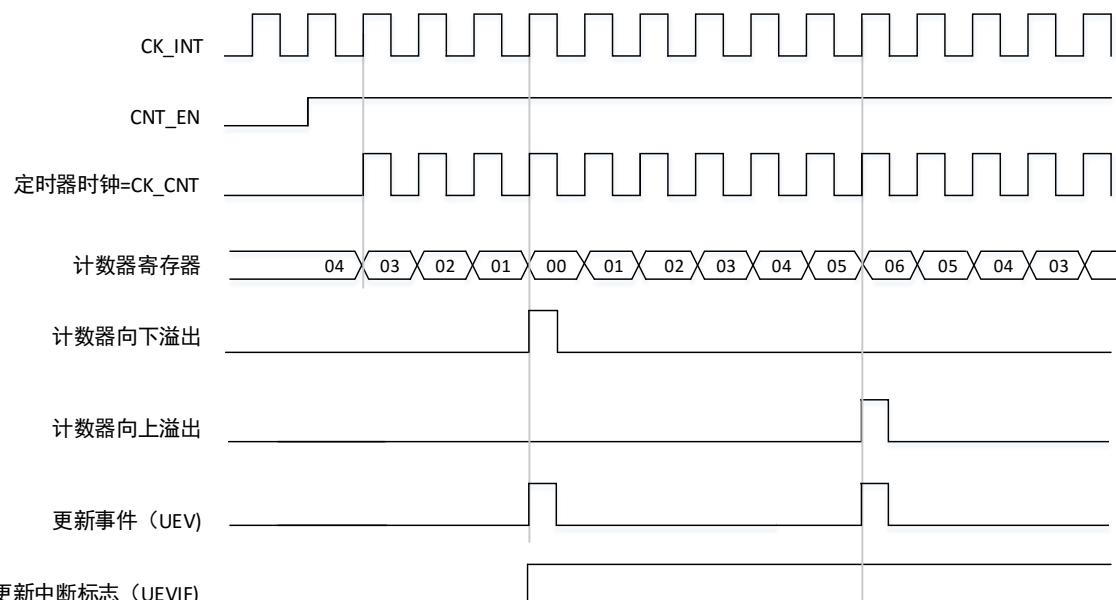
当发生更新事件时，所有的寄存器都被更新，并且（根据 UEVRS 位的设置）更新标志位（TMRx\_STS 寄存器中的 UEVIF 位）也被设置。

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

**注意：**如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值（计数器被装载为新的值）。

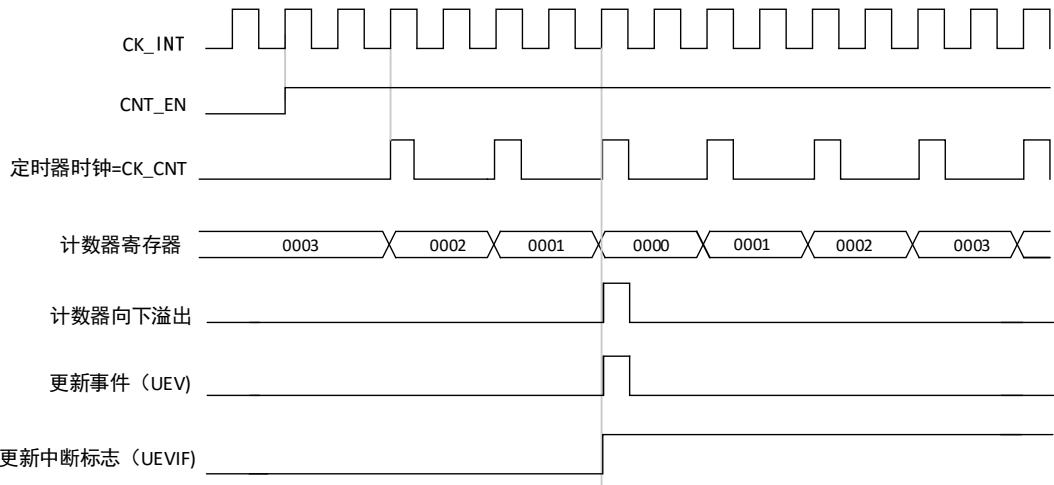
以下是一些计数器在不同时钟频率下的操作的例子：

图表 10-25 计数器时序图，内部时钟分频因子为 1，TMRx\_AR=0x6

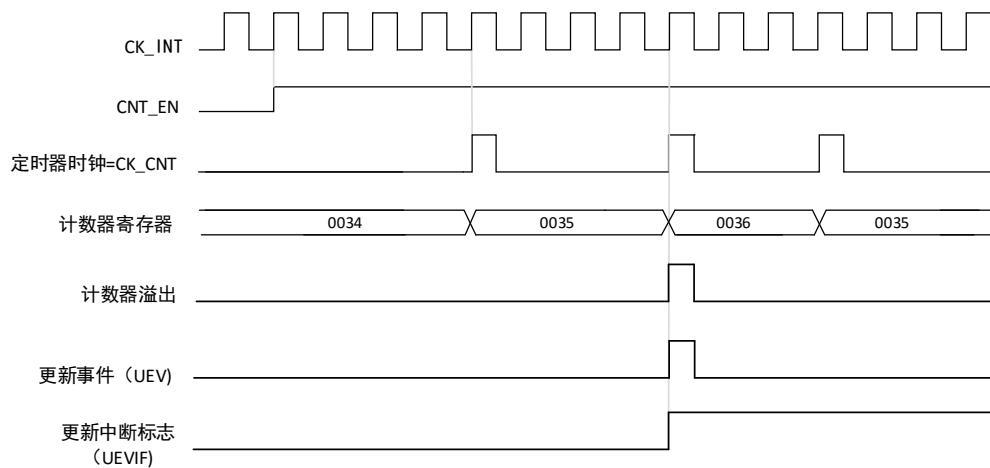


**注意：**这里使用了中心对齐模式 1（寄存器的配置详见 [10.2.4.1 节](#)）。

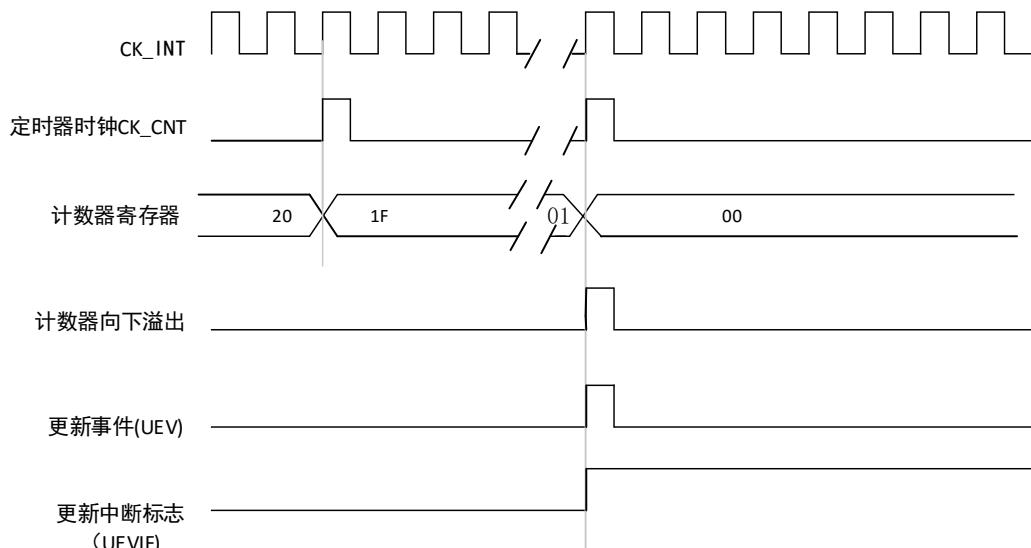
图表 10-26 计数器时序图，内部时钟分频因子为2



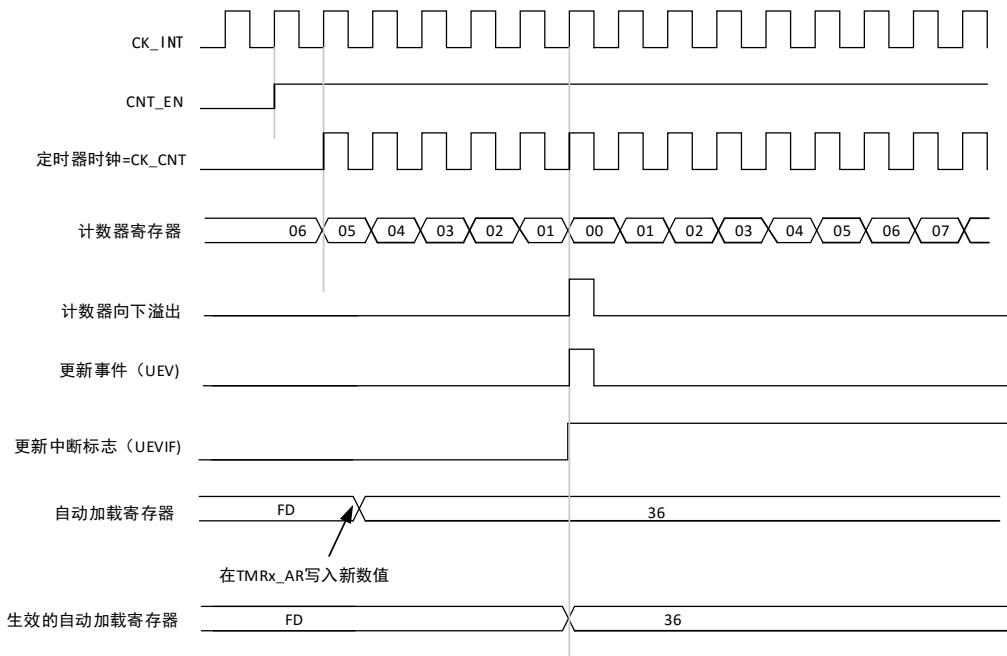
图表 10-27 计数器时序图，内部时钟分频因子为4，TMRx\_AR=0x36



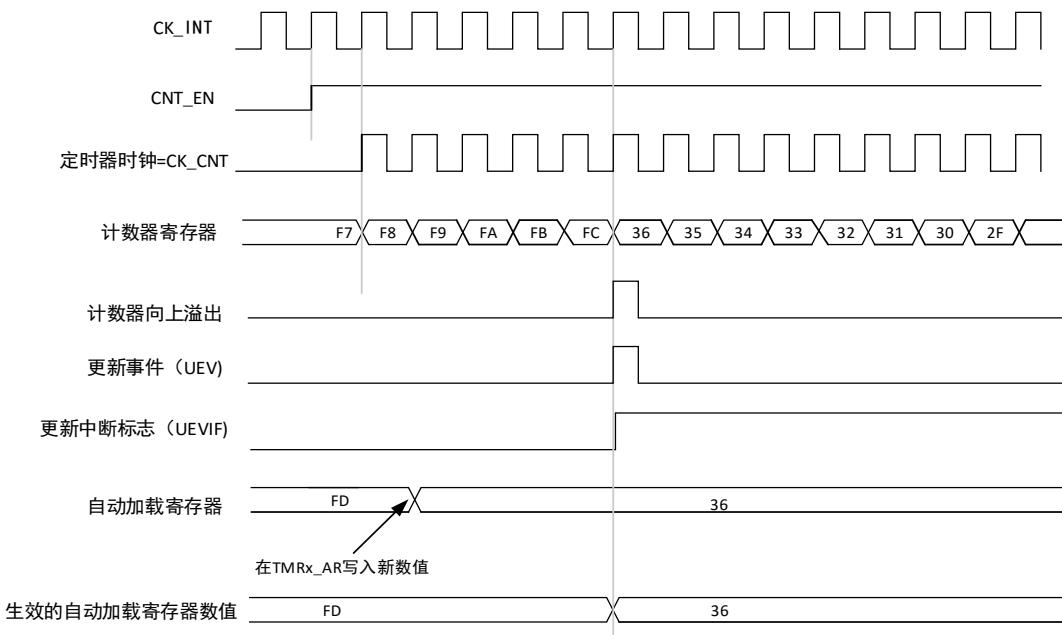
图表 10-28 计数器时序图，内部时钟分频因子为N



图表 10-29 计数器时序图，ARPEN=1时的更新事件（计数器下溢）



图表 10-30 计数器时序图，ARPEN=1时的更新事件（计数器溢出）



### 10.2.3.3 时钟选择

计数器时钟可由下列时钟源提供：

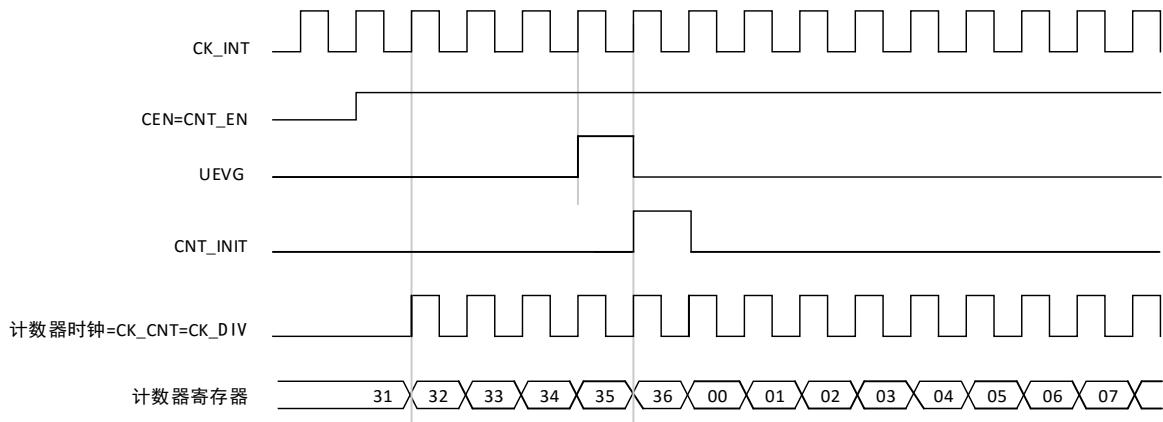
- 内部时钟 (CK\_INT)
- 外部时钟模式1：外部输入脚 (TI<sub>x</sub>)
- 外部时钟模式2：外部触发输入 (ETR)
- 内部触发输入 (ITRx)：使用一个定时器作为另一个定时器的预分频器，如可以配置一个定时器Timer1而作为另一个定时器Timer2的预分频器。参见 [10.2.3.15](#)。

#### 内部时钟源 (CK\_INT)

如果禁止了从模式控制器 (TMRx\_SMC 寄存器的 SMSEL=000)，则 CNTEN、DIR (TMRx\_CTRL1 寄存器) 和 UEVG 位 (TMRx\_EVEG 寄存器) 是事实上的控制位，并且只能被软件修改 (UEVG 位仍被自动清除)。只要 CNTEN 位被写成'1'，预分频器的时钟就由内部时钟 CK\_INT 提供。

下图显示了控制电路和向上计数器在一般模式下，不带预分频器时的操作。

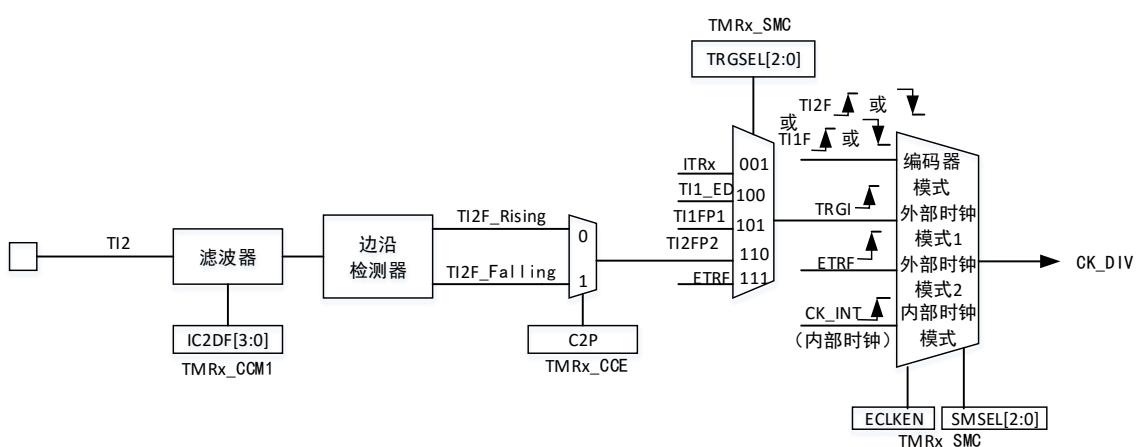
图表 10-31 一般模式下的控制电路，内部时钟分频因子为1



### 外部时钟模式 1

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

图表 10-32 TI2 外部时钟连接例子



例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 配置 TMRx\_CCM1 寄存器 C2SEL='01'，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TMRx\_CCM1 寄存器的 IC2DF[3: 0]，选择输入滤波器带宽（如果不需滤波器，保持 IC2DF=0000）。

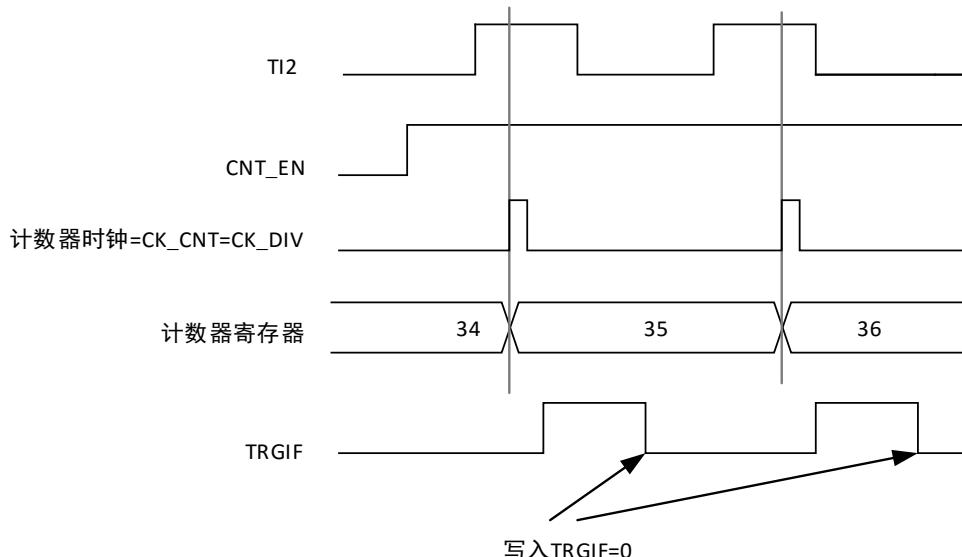
注意：捕获预分频器不用作触发，所以不需要对它进行配置

3. 配置 TMRx\_CCE 寄存器的 C2P='0'，选定上升沿极性
4. 配置 TMRx\_SMC 寄存器的 SMSEL='111'，选择定时器外部时钟模式 1
5. 配置 TMRx\_SMC 寄存器中的 TRGSEL='110'，选定 TI2 作为触发输入源
6. 设置 TMRx\_CTRL1 寄存器的 CNTEN='1'，启动计数器

当上升沿出现在 TI2，计数器计数一次，且 TRGIF 标志被设置。

在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。

图表 10-33 外部时钟模式 1 下的控制电路



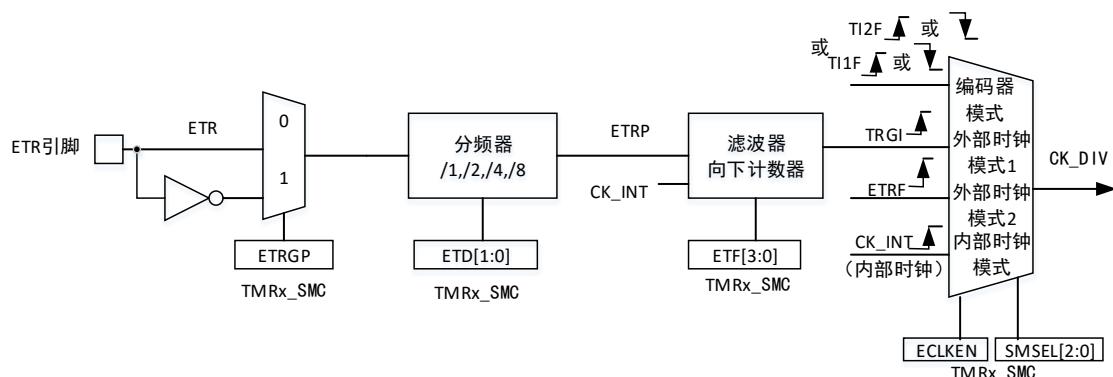
### 外部时钟源模式 2

选定此模式的方法为：令 TMRx\_SMC 寄存器中的 ECLKEN=1

计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图。

图表 10-34 外部触发输入框图



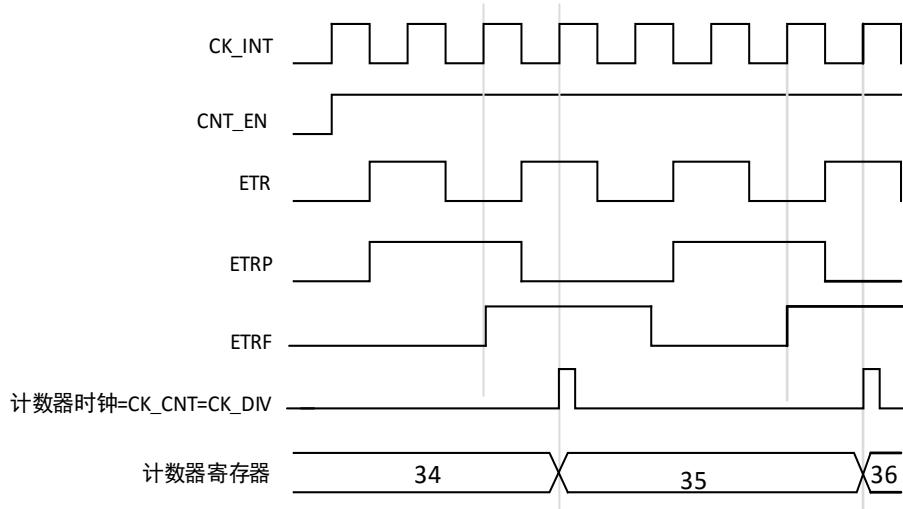
例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：

1. 本例中不需要滤波器，置 TMRx\_SMC 寄存器中的 ETDF[3: 0]=0000，
2. 设置预分频器，置 TMRx\_SMC 寄存器中的 ETD[1: 0]=01，
3. 设置在 ETR 的上升沿检测，置 TMRx\_SMC 寄存器中的 ETRGP=0，
4. 开启外部时钟模式 2，置 TMRx\_SMC 寄存器中的 ECLKEN=1，
5. 启动计数器，置 TMRx\_CTRL1 寄存器中的 CNTEN=1

计数器在每 2 个 ETR 上升沿计数一次。

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

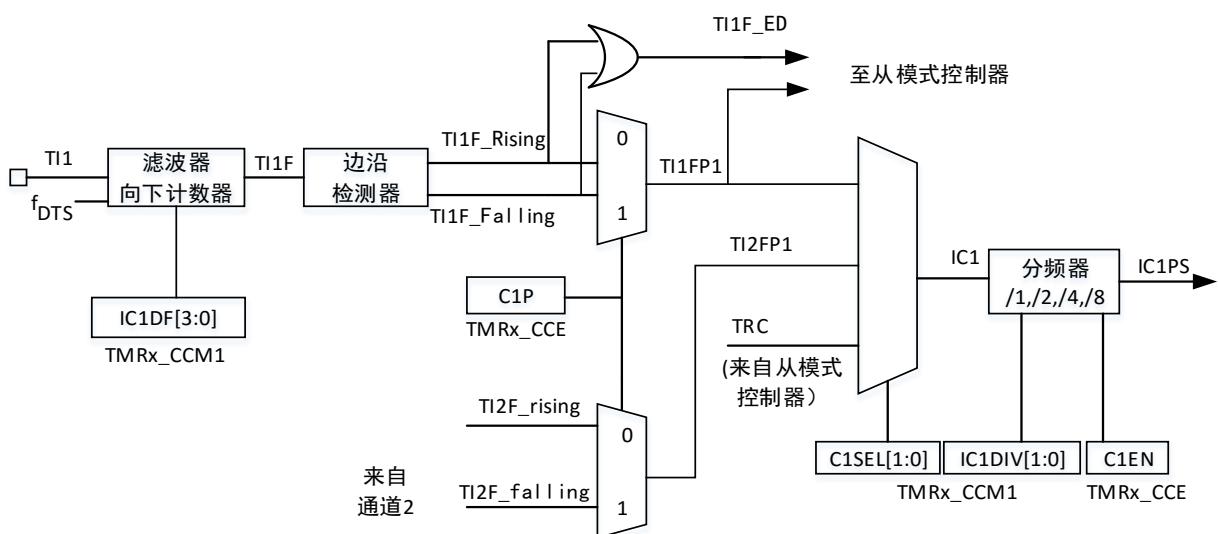
图表 10-35 外部时钟模式2下的控制电路



#### 10.2.3.4 捕获/比较通道

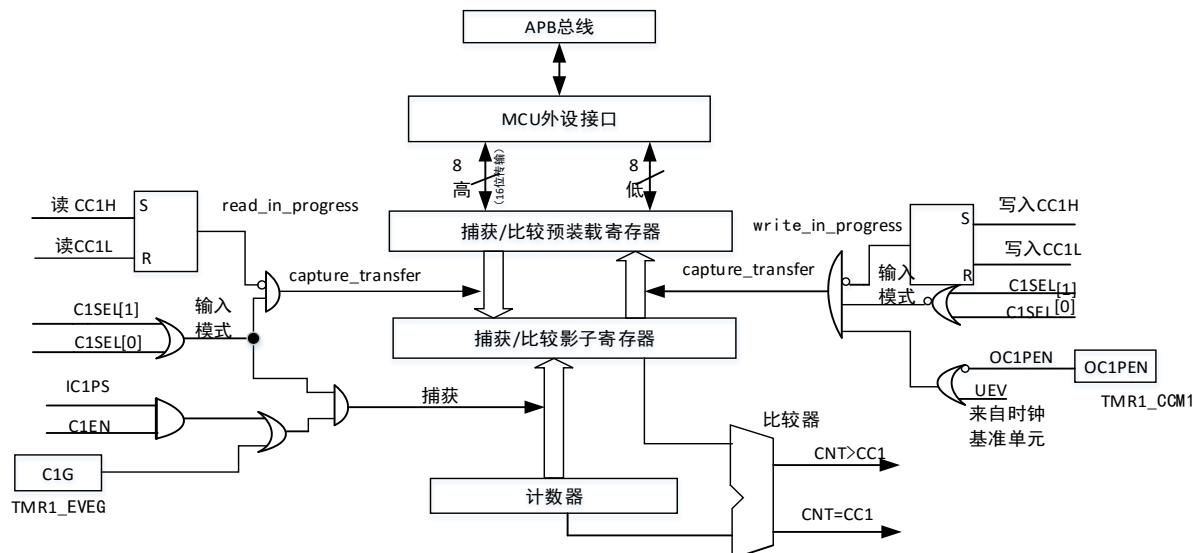
每一个捕获/比较通道都是围绕着一个捕获/比较寄存器（包含影子寄存器），包括捕获的输入部分（数字滤波、多路复用和预分频器），和输出部分（比较器和输出控制）。下面几张图是一个捕获/比较通道概览。输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘检测器产生一个信号 (TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器。

图表 10-36 捕获/比较通道（如：通道1输入部分）

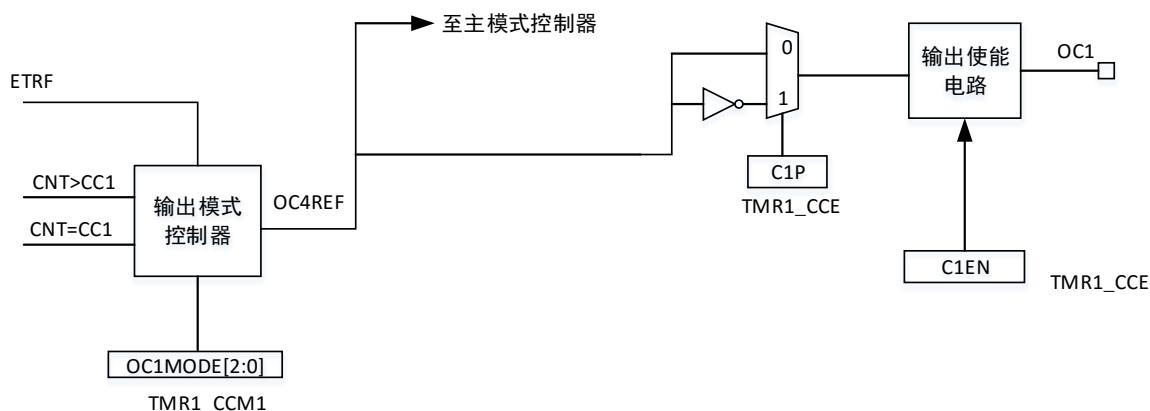


输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

图表 10-37 捕获/比较通道1的主电路



图表 10-38 捕获/比较通道的输出部分（通道 1）



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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

### 10.2.3.5 输入捕获模式

在输入捕获模式下，当检测到  $ICx$  信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 ( $TMRx\_CCx$ ) 中。当捕获事件发生时，相应的  $CxIF$  标志 ( $TMRx\_STS$  寄存器) 被置'1'，如果使能了中断或者 DMA 操作，则将产生中断或者 DMA 操作。如果捕获事件发生时  $CxIF$  标志已经为高，那么重复捕获标志  $CxOF$  ( $TMRx\_STS$  寄存器) 被置'1'。写  $CxIF=0$  可清除  $CxIF$ ，或读取存储在  $TMRx\_CCx$  寄存器中的捕获数据也可清除  $CxIF$ 。写  $CxOF=0$  可清除  $CxOF$ 。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到  $TMRx\_CC1$  寄存器中，步骤如下：

- 选择有效输入端：  $TMRx\_CC1$  必须连接到  $TI1$  输入，所以写入  $TMRx\_CC1$  寄存器中的  $C1SEL=01$ ，只要  $C1SEL$  不为'00'，通道被配置为输入，并且  $TM1\_CC1$  寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为  $TIx$  时，输入滤波器控制位是  $TMRx\_CCMx$  寄存器中的  $ICxDF$  位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期。因此我们可以（以  $f_{DTS}$  频率）连续采样 8 次，以确认在  $TI1$  上一次真实的边沿变换，即在  $TMRx\_CCM1$  寄存器中写入  $IC1DF=0011$ 。
- 选择  $TI1$  通道的有效转换边沿，在  $TMRx\_CCE$  寄存器中写入  $C1P=0$ （上升沿）。

- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写TMRx\_CCM1寄存器的IC1DIV=00）。
- 设置TMRx\_CCE寄存器的C1EN=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置TMRx\_DIE寄存器中的C1IE位允许相关中断请求，通过设置TMRx\_DIE寄存器中的C1DE位允许DMA请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到TMRx\_CC1寄存器。
- C1IF标志被设置（中断标志）。当发生至少2个连续的捕获时，而C1IF未曾被清除，C1OF也被置'1'。
- 如设置了C1IE位，则会产生一个中断。
- 如设置了C1DE位，则还会产生一个DMA请求。

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

注意：设置TMRx\_EVEG寄存器中相应的CxG位，可以通过软件产生输入捕获中断和/或DMA请求。

### 10.2.3.6 PWM输入模式

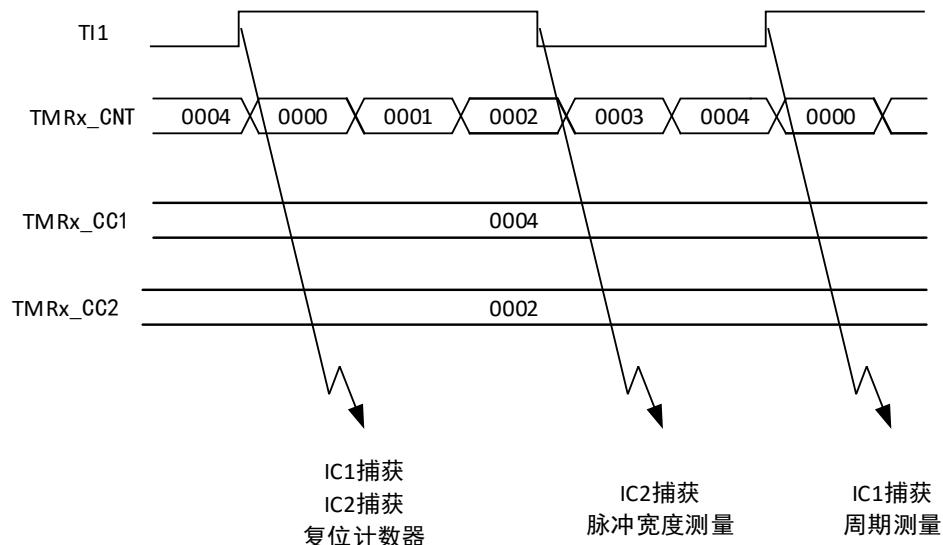
该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

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

例如，你需要测量输入到TI1上的PWM信号的长度（TMRx\_CC1寄存器）和占空比（TMRx\_CC2寄存器），具体步骤如下（取决于CK\_INT的频率和预分频器的值）。

- 选择TMRx\_CC1的有效输入：置TMRx\_CCM1寄存器的C1SEL=01（选择TI1）。
- 选择TI1FP1的有效极性（用来捕获数据到TMRx\_CC1中和清除计数器）：置C1P=0（上升沿有效）。
- 选择TMRx\_CC2的有效输入：置TMRx\_CCM1寄存器的C2SEL=10（选择TI1）。
- 选择TI1FP2的有效极性（捕获数据到TMRx\_CC2）：置C2P=1（下降沿有效）。
- 选择有效的触发输入信号：置TMRx\_SMC寄存器中的TRGSEL=101（选择TI1FP1）。
- 配置从模式控制器为复位模式：置TMRx\_SMC中的SMSEL=100。
- 使能捕获：置TMRx\_CCE寄存器中C1EN=1且C2EN=1。

图表 10-39 PWM输入模式时序



由于只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TMRx\_CH1/TMRx\_CH2 信号。

### 10.2.3.7 强置输出模式

在输出模式 (TMRx\_CCMx 寄存器中 CxSEL=00) 下，输出比较信号 (OCxREF 和相应的 OCx) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 TMRx\_CCMx 寄存器中相应的 OCxMODE=101，即可强置输出比较信号 (OCxREF/OCx) 为有效状态。这样 OCxREF 被强置为高电平 (OCxREF 始终为高电平有效)，同时 OCx 得到 CxP 极性位相反的值。

例如：CxP=0 (OCx 高电平有效)，则 OCx 被强置为高电平。

置 TMRx\_CCMx 寄存器中的 OCxMODE=100，可强置 OCxREF 信号为低。

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

### 10.2.3.8 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式 (TMRx\_CCMx 寄存器中的 OCxMODE 位) 和输出极性 (TMRx\_CCE 寄存器中的 CxP 位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 (OCxMODE=000)、被设置成有效电平 (OCxMODE=001)、被设置成无效电平 (OCxMODE=010) 或进行翻转 (OCxMODE=011)。
- 设置中断状态寄存器中的标志位 (TMRx\_STS 寄存器中的 CxIF 位)。
- 若设置了相应的中断屏蔽 (TMRx\_DIE 寄存器中的 CxIE 位)，则产生一个中断。
- 若设置了相应的使能位 (TMRx\_DIE 寄存器中的 CxDE 位，TMRx\_CTRL2 寄存器中的 CDSEL 位选择 DMA 请求功能)，则产生一个 DMA 请求。

TMRx\_CCMx 中的 OCxPEN 位选择 TMRx\_CCx 寄存器是否需要使用预装载寄存器。

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

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

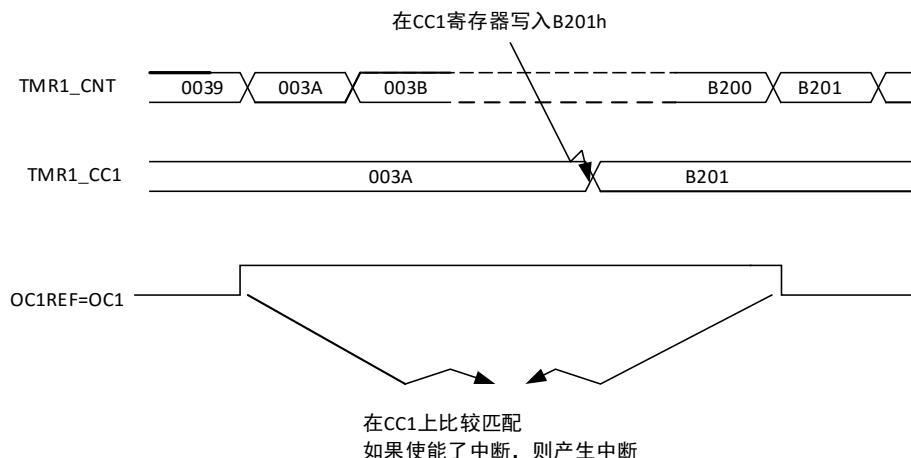
输出比较模式的配置步骤：

- 选择计数器时钟 (内部，外部，预分频器)。
- 将相应的数据写入 TMRx\_AR 和 TMRx\_CCx 寄存器中。

3. 如果要产生一个中断请求和/或一个 DMA 请求, 设置 CxIE 位和/或 CxDE 位。
4. 选择输出模式, 例如当计数器 CNT 与 CC<sub>x</sub> 匹配时翻转 OC<sub>x</sub> 的输出引脚, CC<sub>x</sub> 预装载未用, 开启 OC<sub>x</sub> 输出且高电平有效, 则必须设置 OC<sub>x</sub>MODE='011'、OC<sub>x</sub>PEN='0'、CxP='0'和 CxEN='1'。
5. 设置 TMR<sub>x</sub>\_CTRL1 寄存器的 CNTEN 位启动计数器

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

图表 10-40 输出比较模式, 翻转 OC1



### 10.2.3.9 PWM模式

脉冲宽度调制模式可以产生一个由 TMR<sub>x</sub>\_AR 寄存器确定频率、由 TMR<sub>x</sub>\_CC<sub>x</sub> 寄存器确定占空比的信号。

在 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 OC<sub>x</sub>MODE 位写入'110' (PWM 模式 1) 或'111' (PWM 模式 2), 能够独立地设置每个 OC<sub>x</sub> 输出通道产生一路 PWM。必须设置 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器 OC<sub>x</sub>PEN 位以使能相应的预装载寄存器, 最后还要设置 TMR<sub>x</sub>\_CTRL1 寄存器的 ARPEN 位, (在向上计数或中心对称模式中) 使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候, 预装载寄存器才能被传送到影子寄存器, 因此在计数器开始计数之前, 必须通过设置 TMR<sub>x</sub>\_EVEG 寄存器中的 UEVG 位来初始化所有的寄存器。OC<sub>x</sub> 的极性可以通过软件在 TMR<sub>x</sub>\_CCE 寄存器中的 CxP 位设置, 它可以设置为高电平有效或低电平有效。TMR<sub>x</sub>\_CCE 寄存器中的 CxEN 位控制 OC<sub>x</sub> 输出使能。详见 [10.2.4.9 节 TMR<sub>x</sub> CCE 寄存器](#) 的描述。

在 PWM 模式 (模式 1 或模式 2) 下, TMR<sub>x</sub>\_CNT 和 TMR<sub>x</sub>\_CC<sub>x</sub> 始终在进行比较, (依据计数器的计数方向) 以确定是否符合 TMR<sub>x</sub>\_CC<sub>x</sub> ≤ TMR<sub>x</sub>\_CNT 或者 TMR<sub>x</sub>\_CNT ≤ TMR<sub>x</sub>\_CC<sub>x</sub>。然而为了与 OCREF\_CLR 的功能(在下一个 PWM 周期之前, ETR 信号上的一个外部事件能够清除 OC<sub>x</sub>REF)一致, OC<sub>x</sub>REF 信号只能在下述条件下产生:

- 当比较的结果改变
- 当输出比较模式 (TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 OC<sub>x</sub>MODE 位) 从“冻结” (无比较, OC<sub>x</sub>MODE='000') 切换到某个 PWM 模式 (OC<sub>x</sub>MODE='110'或'111')。

这样在运行中可以通过软件强置 PWM 输出。

根据 TMR<sub>x</sub>\_CTRL1 寄存器中 CMSEL 位的状态, 定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

#### PWM 边沿对齐模式

##### 向上计数配置

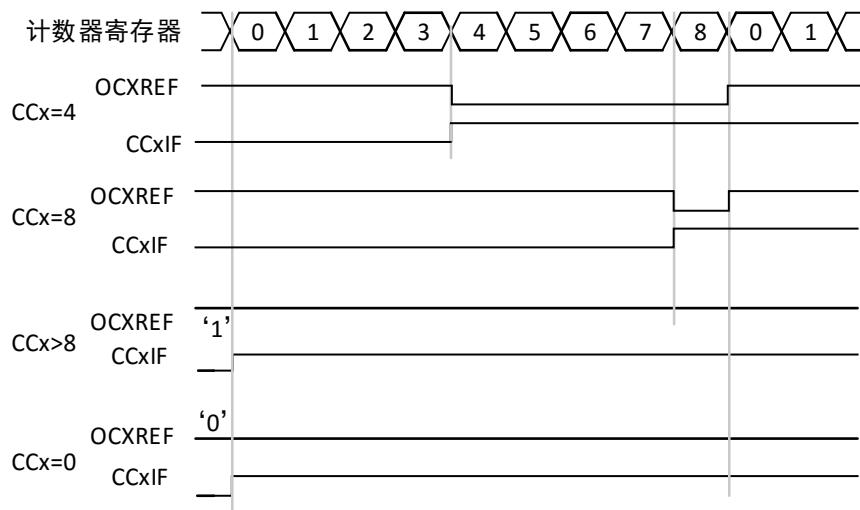
当 TMR<sub>x</sub>\_CTRL1 寄存器中的 DIR 位为低的时候执行向上计数。参看 [10.2.3.2 节](#)。

下面是一个 PWM 模式 1 的例子。当 TMR<sub>x</sub>\_CNT<TMR<sub>x</sub>\_CC<sub>x</sub> 时 PWM 信号参考 OC<sub>x</sub>REF 为高, 否则

为低。如果  $\text{TMRx\_CCx}$  中的比较值大于自动重装载值 ( $\text{TMRx\_AR}$ )，则  $\text{OCxREF}$  保持为'1'。

如果比较值为 0，则  $\text{OCxREF}$  保持为'0'。下图为  $\text{TMRx\_AR}=8$  时边沿对齐的 PWM 波形实例。

图表 10-41 边沿对齐的 PWM 波形 ( $\text{AR}=8$ )



### 向下计数的配置

当  $\text{TMRx\_CTRL1}$  寄存器的 DIR 位为高时执行向下计数。参看 [10.2.3.2 节](#)。

在 PWM 式 1，当  $\text{TMRx\_CNT} > \text{TMRx\_CCx}$  时参考信号  $\text{OCxREF}$  为低，否则高。如果  $\text{TMRx\_CCx}$  中的比较值大于  $\text{TMRx\_AR}$  中的自动重装载值，则  $\text{OCxREF}$  保持为'1'。该模式下不能产生 0% 的 PWM 波形。

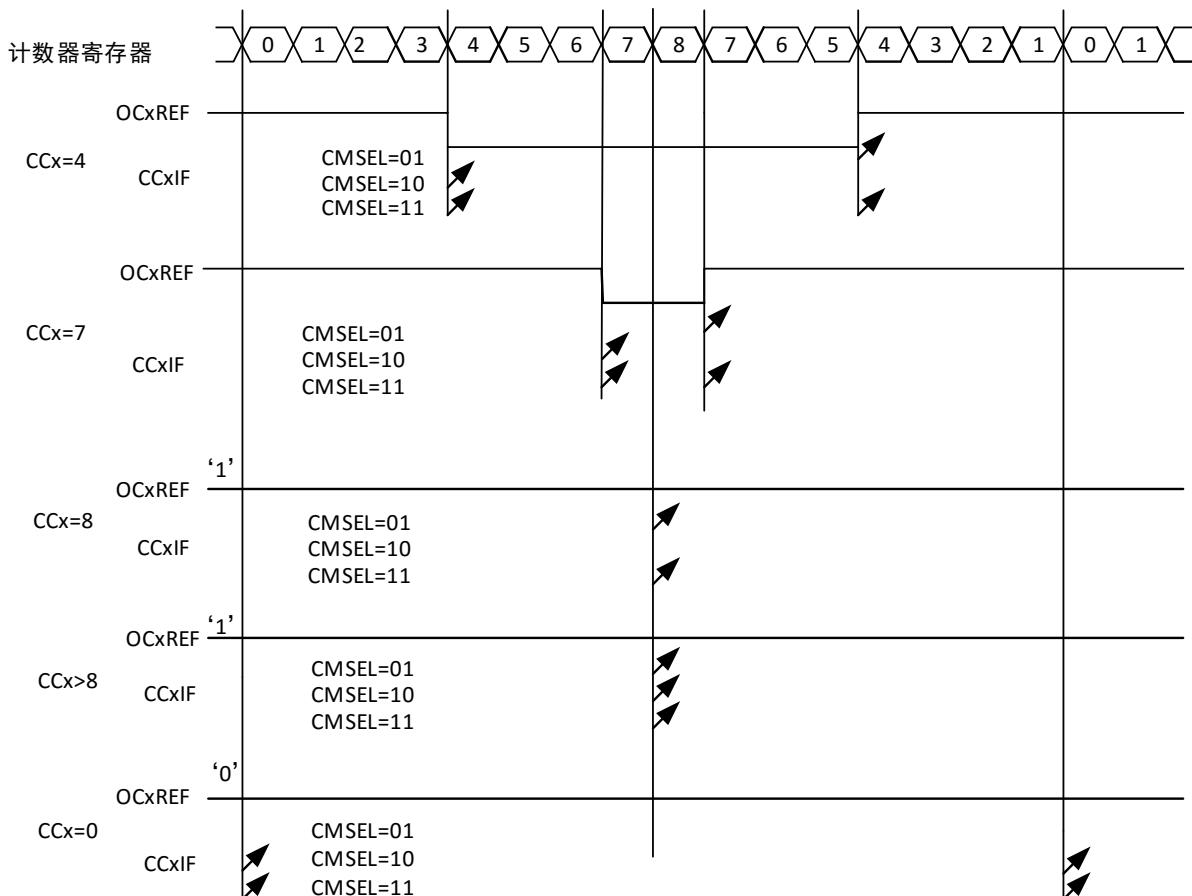
### PWM 中央对齐模式

当  $\text{TMRx\_CTRL1}$  寄存器中的 CMSEL 位不为'00'时，为中央对齐模式（所有其他的配置对  $\text{OCxREF}/\text{OCx}$  信号都有相同的作用）。根据不同的 CMSEL 位设置，比较标志可以在计数器向上计数时被置'1'、在计数器向下计数时被置'1'、或在计数器向上和向下计数时被置'1'。 $\text{TMRx\_CTRL1}$  寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。参看 [10.2.3.2 节](#) 的中央对齐模式。

下图给出了一些中央对齐的 PWM 波形的例子

- $\text{TMRx\_AR}=8$
- PWM 模式 1
- $\text{TMRx\_CTRL1}$  寄存器中的  $\text{CMSEL}=01$ ，在中央对齐模式 1 时，当计数器向下计数时设置比较标志。

图表 10-42 中央对齐的 PWM 波形 ( $\text{AP}=8$ )



#### 使用中央对齐模式的提示:

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这意味着计数器向上还是向下计数取决于TMRx\_CTRL1寄存器中DIR位的当前值。此外，软件不能同时修改DIR和CMSEL位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值（TMRx\_CNT > TMRx\_AR），则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将0或者TMRx\_AR的值写入计数器，方向被更新，但不产生更新事件UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置TMRx\_EVEG位中的UEVG位），不要在计数进行过程中修改计数器的值。

#### 10.2.3.10 单脉冲模式

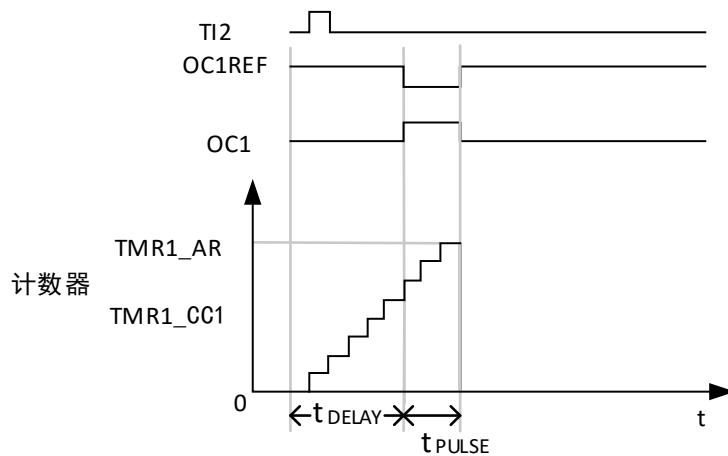
单脉冲模式（OPM）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后，产生一个脉宽可程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TMRx\_CTRL1 寄存器中的 OPMODE 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

向上计数方式： $CNT < CCx \leq AR$  （特别地， $0 < CCx$ ），

向下计数方式： $CNT > CCx$ 。

图表 10-43 单脉冲模式的例子



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

假定 TI2FP2 作为触发 1：

- 置  $TMRx\_CCM1$  寄存器中的  $C2SEL='01'$ ，把  $TI2FP2$  映像到  $TI2$ 。
- 置  $TMRx\_CCE$  寄存器中的  $C2P='0'$ ，使  $TI2FP2$  能够检测上升沿。
- 置  $TMRx\_SMC$  寄存器中的  $TRGSEL='110'$ ， $TI2FP2$  作为从模式控制器的触发 ( $TRGI$ )。
- 置  $TMRx\_SMC$  寄存器中的  $SMSEL='110'$  (触发模式)， $TI2FP2$  被用来启动计数器。

OPM 波形由写入比较寄存器的数值决定 (要考虑时钟频率和计数器预分频器)

- $t_{DELAY}$  由写入  $TMRx\_CC1$  寄存器中的值定义。
- $t_{PULSE}$  由自动装载值和比较值之间的差值定义 ( $TMRx\_AR - TMRx\_CC1$ )。
- 假定当发生比较匹配时要产生从 '0' 到 '1' 的波形，当计数器到达预装载值时要产生一个从 '1' 到 '0' 的波形；首先要置  $TMRx\_CCM1$  寄存器的  $OC1MODE='111'$ ，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置  $TMRx\_CCM1$  中的  $OC1PEN='1'$  和  $TMRx\_CTRL1$  寄存器中的  $ARPEN$ ；然后在  $TMRx\_CC1$  寄存器中填写比较值，在  $TMRx\_AR$  寄存器中填写自动装载值，修改  $UEVG$  位来产生一个更新事件，然后等待在  $TI2$  上的一个外部触发事件。本例中， $C1P='0'$ 。

在这个例子中， $TMRx\_CTRL1$  寄存器中的  $DIR$  和  $CMSEL$  位应该置低。因为只需一个脉冲，所以必须设置  $TMRx\_CTRL1$  寄存器中的  $OPMODE='1'$ ，在下一个更新事件 (当计数器从自动装载值翻转到 0) 时停止计数。

**特殊情况：OCx 快速使能：**

在单脉冲模式下，在  $TIx$  输入脚的边沿检测逻辑设置  $CNTEN$  位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时  $t_{DELAY}$ 。如果要以最小延时输出波形，可以设置  $TMRx\_CCMx$  寄存器中的  $OCxFEN$  位；此时  $OCxREF$  (和  $OCx$ ) 被强制响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。 $OCxFEN$  只在通道配置为 PWM1 和 PWM2 模式时起作用。

### 10.2.3.11 在外部事件时清除 OCxREF 信号

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

该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

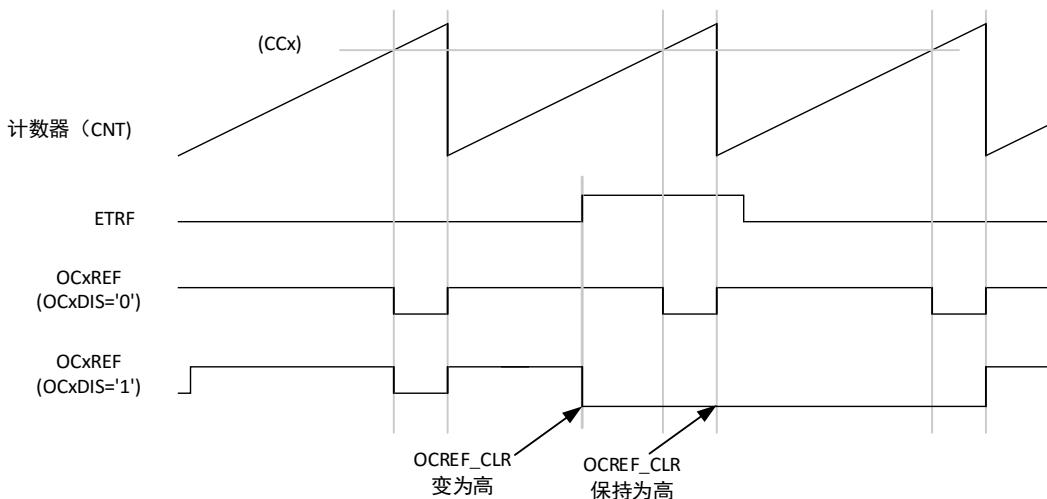
例如， $OCxREF$  信号可以联到一个比较器的输出，用于控制电流。这时， $ETR$  必须配置如下：

1. 外部触发预分频器必须处于关闭： $TMRx\_SMC$  寄存器中的  $ETD[1: 0]='00'$ 。
2. 必须禁止外部时钟模式 2： $TMRx\_SMC$  寄存器中的  $ECLKEN='0'$ 。

3. 外部触发极性 (ETRGP) 和外部触发滤波器 (ETDF) 可以根据需要配置。

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

图表 10-44 清除 TMRx 的 OCxREF



### 10.2.3.12 编码器接口模式

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

通过设置 TMRx\_CCE 寄存器中的 C1P 和 C2P 位, 可以选择 TI1 和 TI2 极性; 如果需要, 还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 10-2, 假定计数器已经启动 (TMRx\_CTRL1 寄存器中的 CNTEEN='1'), 计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号; 如果没有滤波和变相, 则 TI1FP1=TI1, TI2FP2=TI2。根据两个输入信号的跳变顺序, 产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序, 计数器向上或向下计数, 同时硬件对 TMRx\_CTRL1 寄存器的 DIR 位进行相应的设置。

不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数。在任一输入端 (TI1 或者 TI2) 的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TMRx\_AR 寄存器的自动装载值之间连续计数 (根据方向, 或是 0 到 AR 计数, 或是 AR 到 0 计数)。所以在开始计数之前必须配置 TMRx\_AR; 同样, 捕获器、比较器、预分频器、触发输出特性等仍工作如常。

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

表格 10-2 计数方向与编码器信号的关系

有效边沿	相对信号的电平 (TI1FP1 对应 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数

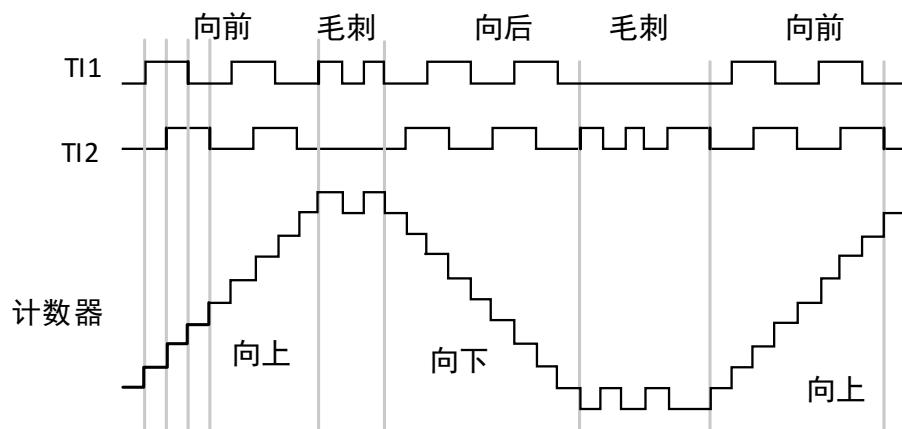
	低	向上计数	向下计数	向下计数	向上计数
--	---	------	------	------	------

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

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

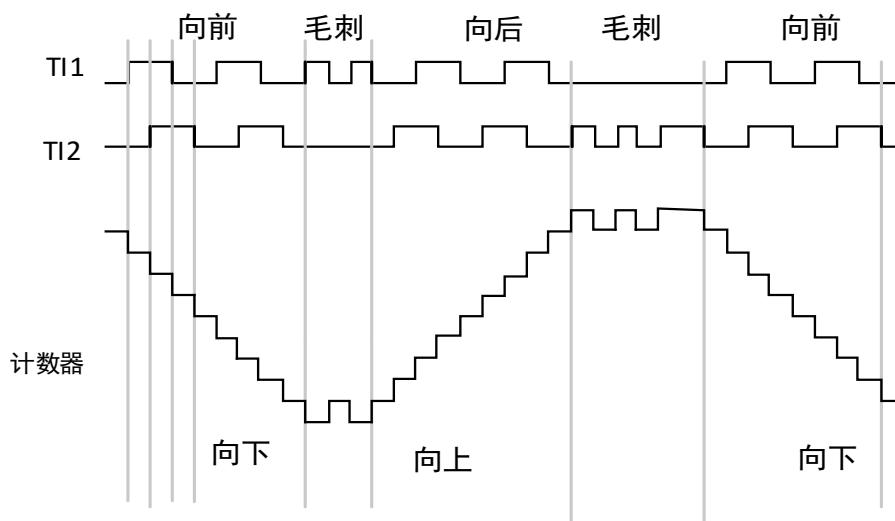
- C1SEL='01' (TMRx\_CCM1 寄存器, IC1FP1 映射到 TI1)。
- C2SEL='01' (TMRx\_CCM1 寄存器, IC2FP2 映射到 TI2)。
- C1P='0' (TMRx\_CCE 寄存器, IC1FP1 不反相, IC1FP1=TI1)。
- C2P='0' (TMRx\_CCE 寄存器, IC2FP2 不反相, IC2FP2=TI2)。
- SMSEL='011' (TMRx\_SMC 寄存器, 所有的输入均在上升沿和下降沿有效)。
- CNTEEN='1' (TMRx\_CTRL1 寄存器, 计数器使能)。

图表 10-45 编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例 (C1P='1', 其他配置与上例相同)

图表 10-46 IC1FP1 反相的编码器接口模式实例



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

以通过一个由实时时钟产生的 DMA 请求来读取它的值。

### 10.2.3.13 定时器输入异或功能

TMRx\_CTRL2 寄存器中的 TI1SEL 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TMRx\_CH1、TMRx\_CH2 和 TMRx\_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。本章的 [10.6.3.18 节](#)给出了此特性用于连接霍尔传感器的例子。

### 10.2.3.14 定时器和外部触发的同步

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

#### 从模式：复位模式

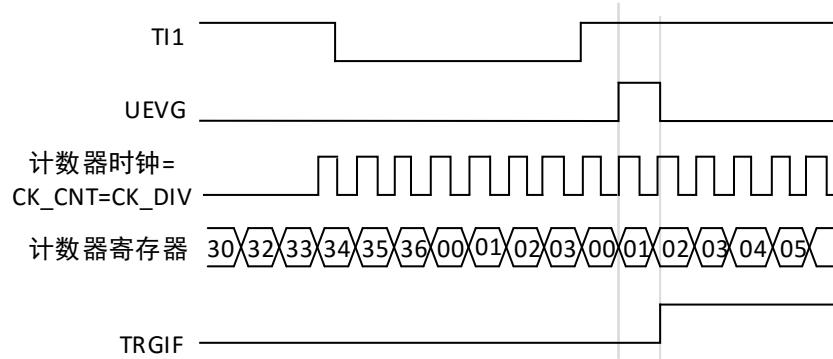
在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TMRx\_CTRL1 寄存器的 UEVRS 位为低，还会产生一个更新事件 UEV；然后所有的预装载寄存器(TMRx\_AR, TMRx\_CC<sub>x</sub>)都会被更新。

在下面的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1DF=0000）。触发操作中不使用捕获预分频器，所以不需要配置它。C1SEL 位只选择输入捕获源，即 TMRx\_CCM1 寄存器中 C1SEL=01。置 TMRx\_CCE 寄存器中 C1P=0 以确定极性（只检测上升沿）。
- 置 TMRx\_SMC 寄存器中 SMSEL=100，配置定时器为复位模式；置 TMRx\_SMC 寄存器中 TRGSEL =101，选择 TI1 作为输入源。
- 置 TMRx\_CTRL1 寄存器中 CNTEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志 (TMRx\_STS 寄存器中的 TRGIF 位) 被设置，根据 TMRx\_DIE 寄存器中 TRGIE (中断使能) 位和 TRGDE (DMA 使能) 位的设置，产生一个中断请求或一个 DMA 请求。下图显示当自动重装载寄存器 TMRx\_AR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时，取决于 TI1 输入端的重同步电路。

图表 10-47 复位模式下的控制电路



#### 从模式：门控模式

按照选中的输入端电平使能计数。

在如下的例子中，计数器只在 TI1 为低时向上计数：

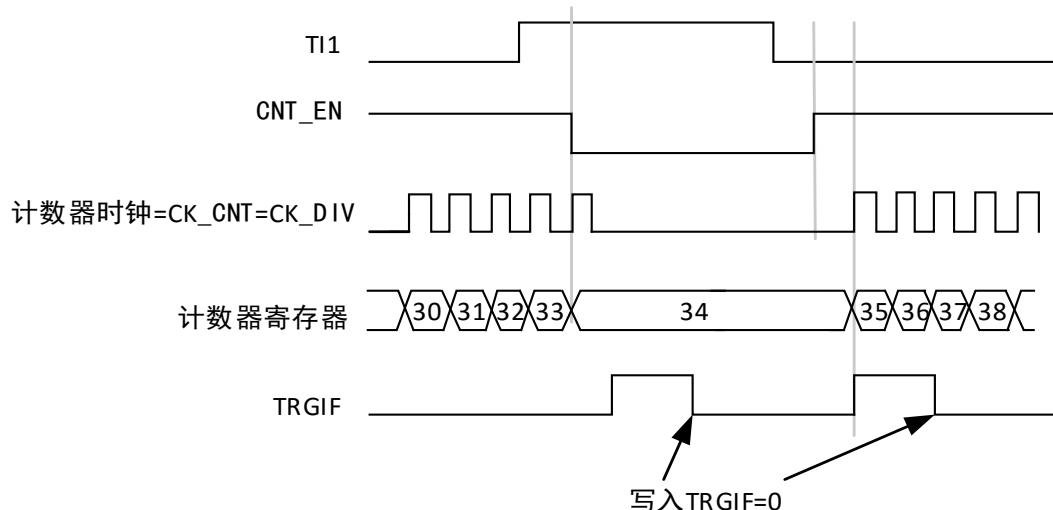
- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1DF=0000）。触发操作中不使用捕获预分频器，所以不需要配置。C1SEL 位用于选择输入捕获源，置 TMRx\_CCM1 寄存器中 C1SEL=01。置 TMRx\_CCE 寄存器中 C1P=1 以确定极性（只检测低电平）。
- 置 TMRx\_SMC 寄存器中 SMSEL=101，配置定时器为门控模式；置 TMRx\_SMC 寄存器中 TRGSEL=101，选择 TI1 作为输入源。

- 置 TMRx\_CTRL1 寄存器中 CNTEN=1，启动计数器。在门控模式下，如果 CNTEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，在 TI1 变高时停止计数。当计数器开始或停止时都设置 TMRx\_STS 中的 TRGIF 标置。

TI1 上升沿和计数器实际停止之间的延时，取决于 TI1 输入端的重同步电路。

图表 10-48 门控模式下的控制电路



#### 从模式：触发模式

输入端上选中的事件使能计数器。

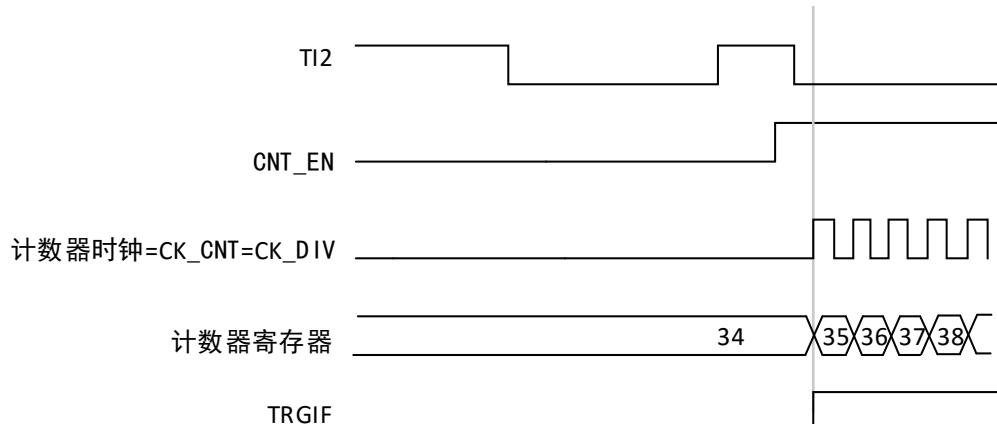
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道2检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2DF=0000）。触发操作中不使用捕获预分频器，不需要配置。C2SEL位只用于选择输入捕获源，置 TMRx\_CCM1 寄存器中 C2SEL=01。置 TMRx\_CCE 寄存器中 C2P=0以确定极性（只检测低电平）。
- 置 TMRx\_SMC 寄存器中 SMSEL=110，配置定时器为触发模式；置 TMRx\_SMC 寄存器中 TRGSEL=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TRGIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图表 10-49 触发器模式下的控制电路



#### 从模式：外部时钟模式 2 + 触发模式

外部时钟模式 2 可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。这时，ETR 信号

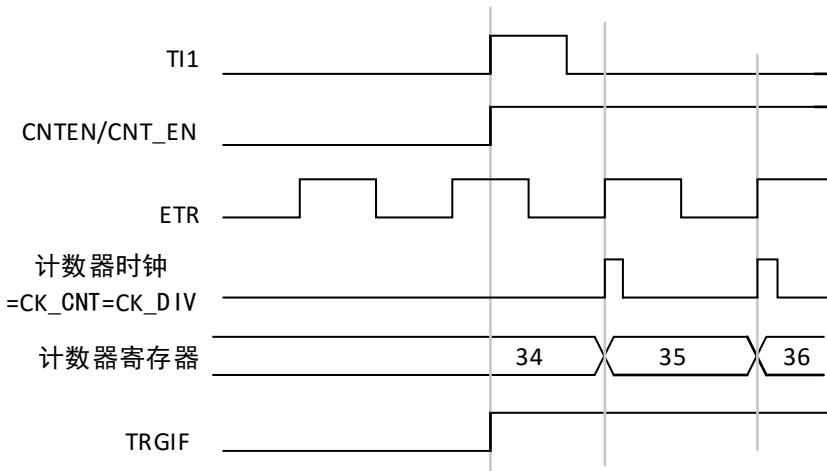
被用作外部时钟的输入，在复位模式、门控模式或触发模式时可以选择另一个输入作为触发输入。不建议使用 **TMRx\_SMC** 寄存器的 **TRGSEL** 位选择 **ETR** 作为 **TRGI**。

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

1. 通过 **TMRx\_SMC** 寄存器配置外部触发输入电路：
  - **ETDF=0000**: 没有滤波
  - **ETD=00**: 不用预分频器
  - **ETRGP=0**: 检测 **ETR** 的上升沿，置 **ECLKEN=1** 使能外部时钟模式 2
2. 按如下配置通道 1，检测 **TI** 的上升沿：
  - **IC1DF=0000**: 没有滤波
  - 触发操作中不使用捕获预分频器，不需要配置
  - 置 **TMRx\_CCM1** 寄存器中 **C1SEL=01**，选择输入捕获源
  - 置 **TMRx\_CCE** 寄存器中 **C1P=0** 以确定极性（只检测上升沿）
3. 置 **TMRx\_SMC** 寄存器中 **SMSEL=110**，配置定时器为触发模式。置 **TMRx\_SMC** 寄存器中 **TRGSEL=101**，选择 **TI1** 作为输入源。

当 **TI1** 上出现一个上升沿时，**TRGIF** 标志被设置，计数器开始在 **ETR** 的上升沿计数。**ETR** 信号的上升沿和计数器实际复位间的延时，取决于 **ETRP** 输入端的重同步电路。

图表 10-50 外部时钟模式2+触发模式下的控制电路



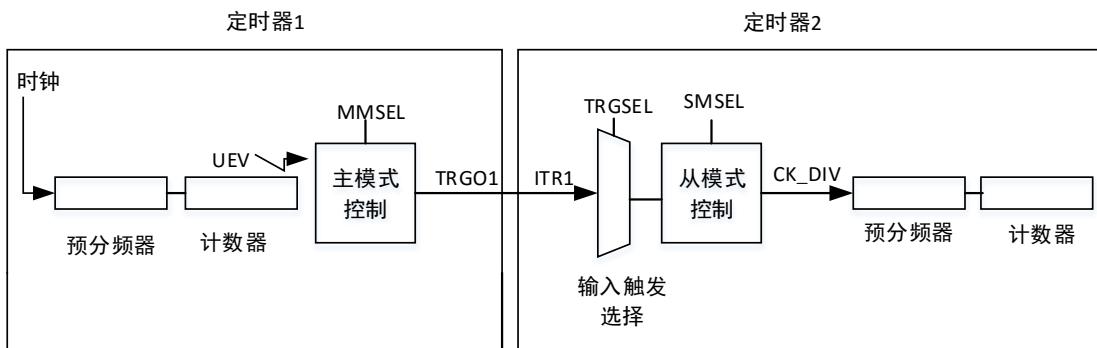
### 10.2.3.15 定时器同步

所有 **TMRx** 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况。

**使用一个定时器作为另一个定时器的预分频器**

图表 10-51 主/从定时器的例子



如：可以配置定时器 1 作为定时器 2 的预分频器。参考[图 10-51](#)，进行下述操作：

- 配置定时器1为主模式，它可以在每一个更新事件UEV时输出一个周期性的触发信号。在TMR1\_CTRL2寄存器的MMSEL='010'时，每当产生一个更新事件时在TRGO1上输出一个上升沿信号。
- 连接定时器1的TRGO1输出至定时器2，设置TMR2\_SMC寄存器的TRGSEL='000'，配置定时器2为使用ITR1作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式1（TMR2\_SMC寄存器的SMSEL=111）；这样定时器2即可由定时器1周期性的上升沿（即定时器1的计数器溢出）信号驱动。
- 最后，必须设置相应（TMRx\_CTRL1寄存器）的CNTEN位分别启动两个定时器。

**注意：**如果 OCx 已被选中为定时器 1 的触发输出（MMSEL=1xx），它的上升沿用于驱动定时器 2 的计数器。

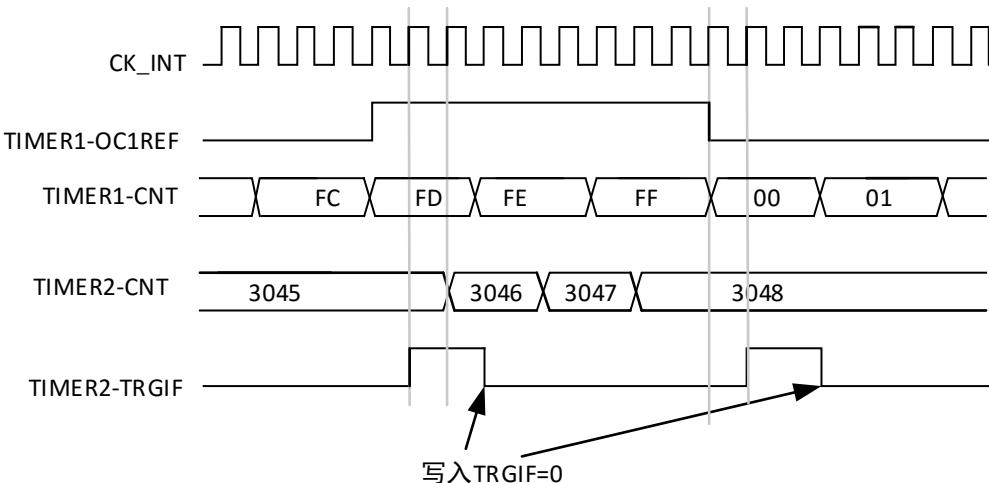
#### 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考[图 10-51](#)的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK\_INT 除以 3 ( $f_{CK\_CNT}=f_{CK\_INT}/3$ ) 得到。

- 配置定时器1为主模式，送出它的输出比较参考信号（OC1REF）为触发输出（TMR1\_CTRL2寄存器的MMSEL=100）
- 配置定时器1的OC1REF波形（TMR1\_CCM1寄存器）
- 配置定时器2从定时器1获得输入触发（TMR2\_SMC寄存器的TRGSEL=000）
- 配置定时器2为门控模式（TMR2\_SMC寄存器的SMSEL=101）
- 置 TMR2\_CTRL1寄存器的CNTEN=1 以使能定时器2
- 置 TMR1\_CTRL1寄存器的CNTEN=1 以启动定时器1

**注意：**定时器 2 的时钟不与定时器 1 的时钟同步，这个模式只影响定时器 2 计数器的使能信号。

图表 10-52 定时器1的OC1REF控制定时器2

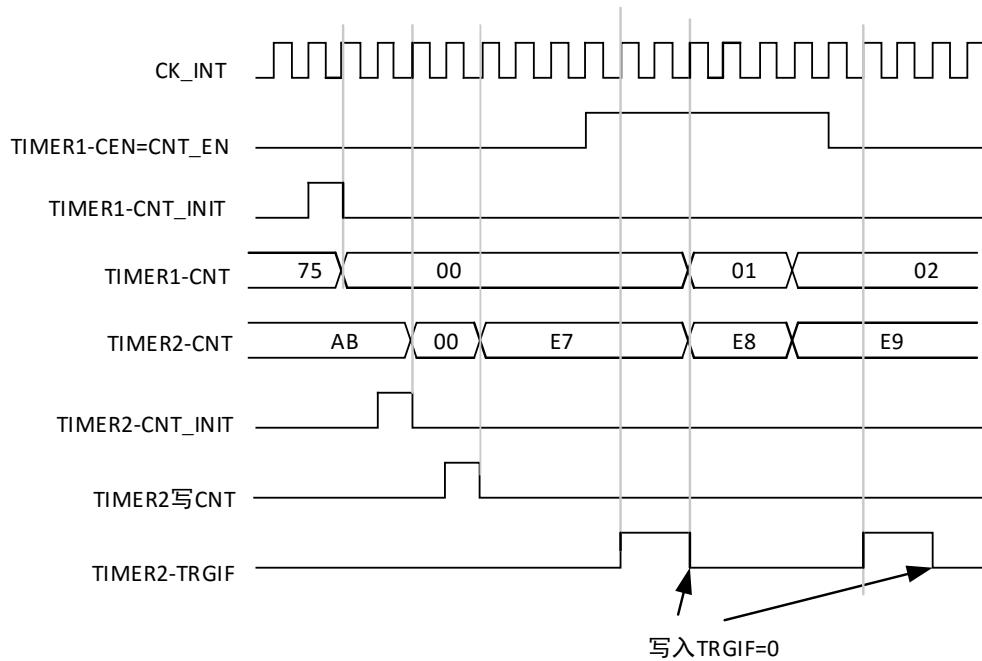


在图 10-52 的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 1 之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入需要的任意数值。写 TMRx\_EVEG 寄存器的 UEVG 位即可复位定时器。

在下一个例子中，需要同步定时器 1 和定时器 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写'0'到 TMR1\_CTRL1 的 CNTEN 位将禁止定时器 1，定时器 2 随即停止。

- 配置定时器 1 为主模式，送出输出比较 1 参考信号（OC1REF）做为触发输出（TMR1\_CTRL2 寄存器的 MMSEL=100）。
- 配置定时器 1 的 OC1REF 波形（TMR1\_CCM1 寄存器）。
- 配置定时器 2 从定时器 1 获得输入触发（TMR2\_SMC 寄存器的 TRGSEL=000）
- 配置定时器 2 为门控模式（TMR2\_SMC 寄存器的 SMSEL=101）
- 置 TMR1\_EVEG 寄存器的 UEVG='1'，复位定时器 1。
- 置 TMR2\_EVEG 寄存器的 UEVG='1'，复位定时器 2。
- 写'0xE7'至定时器 2 的计数器（TMR2\_CNT），初始化它为 0xE7。
- 置 TMR2\_CTRL1 寄存器的 CNTEN='1'以使能定时器 2。
- 置 TMR1\_CTRL1 寄存器的 CNTEN='1'以启动定时器 1。
- 置 TMR1\_CTRL1 寄存器的 CNTEN='0'以停止定时器 1。

图表 10-53 通过使能定时器 1 可以控制定时器 2

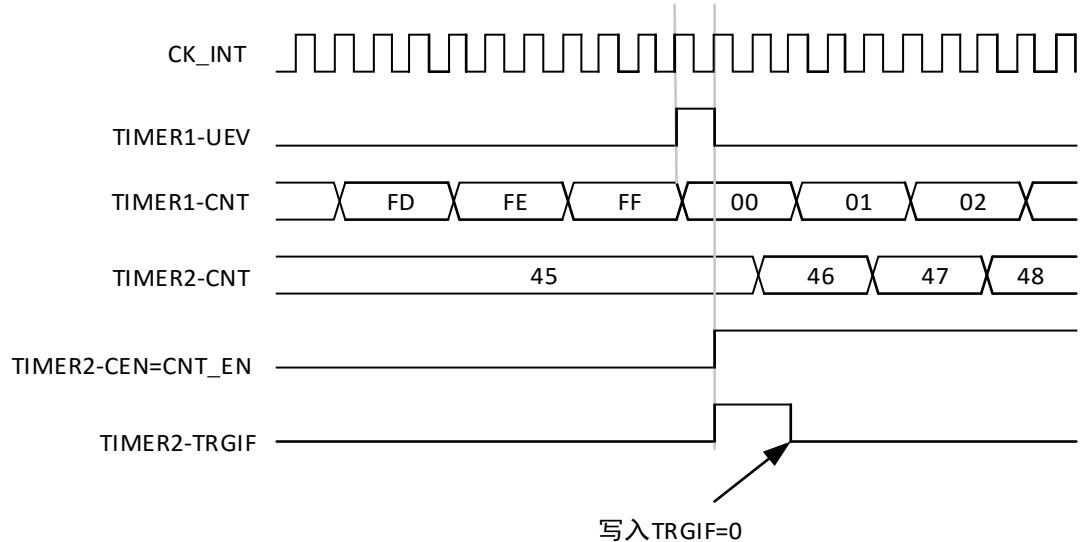


### 使用一个定时器去启动另一个定时器

在这个例子中，使用定时器 1 的更新事件使能定时器 2。参考[图 10-51](#)的连接。一旦定时器 1 产生更新事件，定时器 2 即从它当前的数值（可以是非 0）按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CNTEN 位被自动地置‘1’，同时计数器开始计数直到写‘0’到 TMR2\_CTRL1 寄存器的 CNTEN 位。两个定时器的时钟频率都是由预分频器对 CK\_INT 除以 3 ( $f_{CK\_CNT}=f_{CK\_INT}/3$ )。

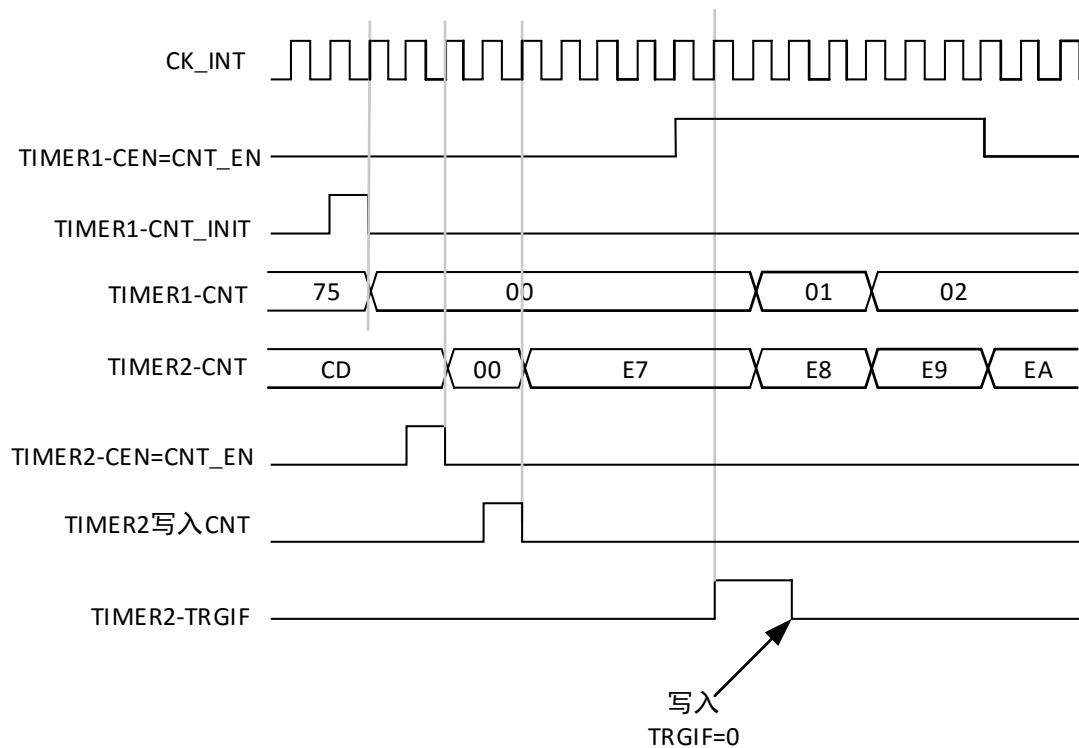
- 配置定时器1为主模式，送出它的更新事件（UEV）做为触发输出（TMR1\_CTRL2寄存器的MMSEL=010）。
- 配置定时器1的周期（TMR1\_AR寄存器）。
- 配置定时器2从定时器1获得输入触发（TMR2\_SMC寄存器的TRGSEL=000）
- 配置定时器2为触发模式（TMR2\_SMC寄存器的SMSEL=110）
- 置TMR1\_CTRL1寄存器的CNTEN=1以启动定时器1。

图表 10-54 使用定时器1的更新触发定时器2



在上一个例子中，可以在启动计数之前初始化两个计数器。[图 10-55](#) 显示在与上例相同配置情况下，使用触发模式而不是门控模式（TMR2\_SMC 寄存器的 SMSEL=110）的动作。

图表 10-55 利用定时器1的使能触发定时器2



### 使用一个定时器作为另一个的预分频器

这个例子使用定时器 1 作为定时器 2 的预分频器。参考[图 10-51](#)的连接，配置如下：

- 配置定时器1为主模式，送出它的更新事件UEV做为触发输出（TMR1\_CTRL2寄存器的MMSEL='010'）。然后每次计数器溢出时输出一个周期信号。
- 配置定时器1的周期（TMR1\_AR寄存器）。
- 配置定时器2从定时器1获得输入触发（TMR2\_SMC寄存器的TRGSEL=000）。
- 配置定时器2使用外部时钟模式（TMR2\_SMC寄存器的SMSEL=111）。
- 置TMR1\_CTRL2寄存器的CNTEN=1以启动定时器2。
- 置TMR1\_CTRL1寄存器的CNTEN=1以启动定时器1。

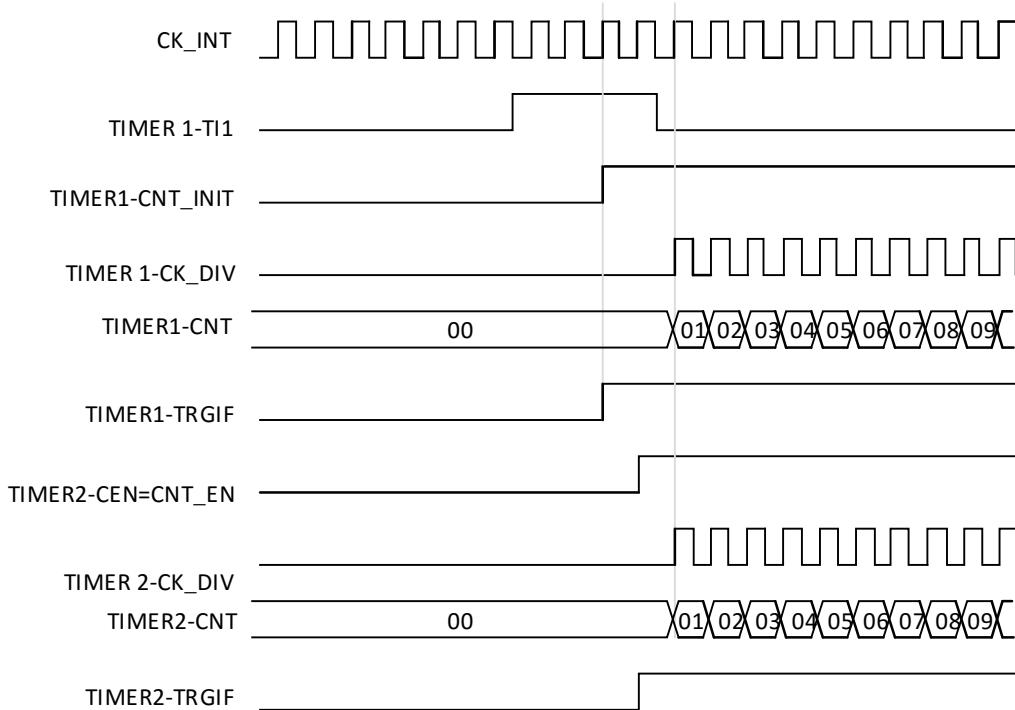
### 使用一个外部触发同步地启动 2 个定时器。

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见[图 10-51](#)。为保证计数器的对齐，定时器 1 必须配置为主/从模式（对于 TI1 为从，对于定时器 2 为主）：

- 配置定时器1为主模式，送出它的使能做为触发输出（TMR1\_CTRL2寄存器的MMSEL='001'）。
- 配置定时器1为从模式，从TI1获得输入触发（TMR1\_SMC寄存器的TRGSEL='100'）。
- 配置定时器1为触发模式（TMR1\_SMC寄存器的SMSEL='110'）。
- 配置定时器1为主/从模式，TMR1\_SMC寄存器的MSMODE='1'。
- 配置定时器2从定时器1获得输入触发（TMR2\_SMC寄存器的TRGSEL=000）
- 配置定时器2为触发模式（TMR2\_SMC寄存器的SMSEL='110'）。当定时器1的TI1上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个TRGIF标志也同时被设置。

**注意：** 在这个例子中，在启动之前两个定时器都被初始化（设置相应的UEVG位），两个计数器都从0开始，但可以通过写入任意一个计数器寄存器（TMRx\_CNT）在定时器间插入一个偏移。下图中能看到主/从模式下在定时器1的CNT\_EN和CK\_DIV之间有个延迟。

图表 10-56 使用定时器1的TI1输入触发定时器1和定时器2



### 10.2.3.16 调试模式

当微控制器进入调试模式(Cortex™-M4 核心停止),根据 DBG 模块中 `DBG_TMRx_STOP` 的设置, TMRx 计数器或者继续正常操作, 或者停止。详见随后[第 18.2.2 节: 支持定时器、看门狗和 I2C 的调试](#)。

### 10.2.4 TMRx 寄存器描述

可以用半字(16位)或字(32位)的方式操作这些外设寄存器。

下表中将 TMRx 的所有寄存器映射到一个 16 位可寻址(编址)空间。

表格 10-3 TMRx – 寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TMRx_CTRL1	保留																				NEMD	CLKDIV[1:0]	ARPEN	DIR	OPMODE	UVERS	UEDIS	CNTEN	0			
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x04	TMRx_CTRL2	保留																				T1SEL	MMSEL[2:0]	CMSEL[1:0]	DIR	CDSEL	保留						
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x08	TMRx_SMC	保留												ETRGP	ECLKEN	ETD[1:0]	ETDF[3:0]	MSMODE	TRGSEL[2:0]	TRGSEL[2:0]	MMSEL[2:0]	CMSEL[1:0]	DIR	CDSEL	保留								
	复位值	0	0	0	0	0	0	0	0	0	0	0	0																				
0x0C	TMRx_DIE	保留												TRGDE	保留	C4DE	C3DE	C2DE	C1DE	UEVDE	保留	TRGE	保留	C4IE	C3IE	C2IE	C1IE	UEVIE	0	0	0		



0x40	TMRx_CC4	CC4[31: 16]												CC4[15: 0]																				
	复位值	0 0												0 0																				
0x48	TMRx_DMAB	保留												DBLEN[4: 0]				保留	ADDR[4: 0]				0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0											
	复位值	0 0												0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0					0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0				0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0											
0x4C	TMRx_DMABA	保留												DMABA[15: 0]												0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0								
	复位值	0 0												0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0												0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0								

### 10.2.4.1 控制寄存器1 (TMRx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

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

位 15: 11	保留, 始终读为 0。
位 10	<b>PMEN:</b> 增强模式使能 (Plus Mode Enable) 开启 TMRx 增强模式, 该模式下 TMRx_CNT, TMRx_AR, TMRx_CC1/2/3/4 由 16 位扩展为 32 位。 0: 不开启增强模式; 1: 开启增强模式。 注: TMR2 和 TMR5 才具有此功能, 其它 TMR 设置此位无效。
位 9: 8	<b>CLKDIV[1: 0]:</b> 时钟分频因子 (Clock division) 定义在定时器时钟 (CK_INT) 频率与数字滤波器 (ETR, TIx) 使用的采样频率之间的分频比例。 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: 保留
位 7	<b>ARPEN:</b> 自动重装载预装载允许位 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲; 1: TMRx_AR 寄存器被装入缓冲器。
位 6: 5	<b>CMSEL[1: 0]:</b> 选择中央对齐模式 (Center-aligned mode selection) 00: 边沿对齐模式。计数器依据方向位 (DIR) 向上或向下计数。 01: 中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 只在计数器向下计数时被设置。 10: 中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 只在计数器向上计数时被设置。 11: 中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 在计数器向上和向下计数时均被设置。 注: 在计数器开启时 (CNTEN=1), 不允许从边沿对齐模式转换到中央对齐模式。
位 4	<b>DIR:</b> 方向 (Direction) 0: 计数器向上计数; 1: 计数器向下计数。 注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。
位 3	<b>OPMODE:</b> 单脉冲模式 (One pulse mode) 0: 在发生更新事件时, 计数器不停止; 1: 在发生下一次更新事件 (清除 CNTEN 位) 时, 计数器停止。

位 2	<b>UEVRS:</b> 更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生： - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 1: 禁止 UEV。不产生更新事件，影子寄存器 (AR、DIV、CCx) 保持它们的值。如果设置 UEVG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
位 0	<b>CNTEN:</b> 使能计数器 0: 禁止计数器； 1: 使能计数器。 注：在软件设置了 CNTEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CNTEN 位。 在单脉冲模式下，当发生更新事件时，CNTEN 被自动清除。

#### 10.2.4.2 控制寄存器2 (TMRx\_CTRL2)

偏移地址: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				TI1SEL EL		MMSEL[2: 0]		CDS EL	保留		res				
res		rw		rw		rw		rw		rw		rw		res	

位 15: 8	保留，始终读为 0。
位 7	<b>TI1SEL:</b> TI1 选择 (TI1 selection) 0: TMRx_CH1 引脚连到 TI1 输入； 1: TMRx_CH1、TMRx_CH2 和 TMRx_CH3 引脚经异或后连到 TI1 输入。 见本章 <a href="#">10.6.3.18</a> 的与霍尔传感器的接口一节。

位 6: 4	<b>MMSEL[2: 0]:</b> 主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 – TMRx_EVEG 寄存器的 UEVG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001: 使能 – 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CNTEN 控制位和门控模式下的触发输入信号的逻辑或产生。 当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TMRx_SMC 寄存器中 MSMODE 位的描述)。 010: 更新 – 更新事件被选为触发输入 (TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 C1IF 标志时 (即使它已经为高)，触发输出送出一个正脉冲 (TRGO)。 100: 比较 – OC1REF 信号被用于作为触发输出 (TRGO)。 101: 比较 – OC2REF 信号被用于作为触发输出 (TRGO)。 110: 比较 – OC3REF 信号被用于作为触发输出 (TRGO)。 111: 比较 – OC4REF 信号被用于作为触发输出 (TRGO)。
	<b>CDSEL:</b> 捕获/比较的 DMA 选择 0: 当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1: 当发生更新事件时，送出 CCx 的 DMA 请求。
	位 2: 0 保留，始终读为 0。

### 10.2.4.3 从模式控制寄存器 (TMRx\_SMC)

偏移地址: 0x08

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETR GP	ECL KEN	ETD[1: 0]		ETDF[3: 0]	MSM ODE	TRGSEL[2: 0]	保留	SMSEL[2: 0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	res	rw	rw	rw

位 15	<b>ETRGP:</b> 外部触发极性 (External trigger polarity) 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0: ETR 不反相，高电平或上升沿有效； 1: ETR 被反相，低电平或下降沿有效。
位 14	<b>ECLKEN:</b> 外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0: 禁止外部时钟模式 2； 1: 使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1: 设置 ECLKEN 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF (SMSEL=111 和 TRGSEL=111) 具有相同功效。 注 2: 下述从模式可以与外部时钟模式 2 同时使用：复位模式、门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF (TRGSEL 位不能是'111')。 注 3: 外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
位 13: 12	<b>ETD[1: 0]:</b> 外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须最多是 CK_INT 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00: 关闭预分频； 01: ETRP 频率除以 2； 10: ETRP 频率除以 4； 11: ETRP 频率除以 8。

位 11: 8	<p><b>ETDF[3: 0]: 外部触发滤波 (External trigger filter)</b>          这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。</p> <table border="0"> <tbody> <tr> <td>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</td><td>1000: 采样频率 <math>f_{SAMPLING} = f_{DTS}/8</math>, N=6</td></tr> <tr> <td>0001: 采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=2</td><td>1001: 采样频率 <math>f_{SAMPLING} = f_{DTS}/8</math>, N=8</td></tr> <tr> <td>0010: 采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=4</td><td>1010: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=5</td></tr> <tr> <td>0011: 采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=8</td><td>1011: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=6</td></tr> <tr> <td>0100: 采样频率 <math>f_{SAMPLING} = f_{DTS}/2</math>, N=6</td><td>1100: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=8</td></tr> <tr> <td>0101: 采样频率 <math>f_{SAMPLING} = f_{DTS}/2</math>, N=8</td><td>1101: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=5</td></tr> <tr> <td>0110: 采样频率 <math>f_{SAMPLING} = f_{DTS}/4</math>, N=6</td><td>1110: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=6</td></tr> <tr> <td>0111: 采样频率 <math>f_{SAMPLING} = f_{DTS}/4</math>, N=8</td><td>1111: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=8</td></tr> </tbody> </table>	0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6	0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8	0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5	0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6	0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8	0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5	0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6	0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8
0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6																
0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8																
0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5																
0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6																
0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8																
0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5																
0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6																
0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8																
位 7	<p><b>MSMODE: 主/从模式 (Master/slave mode)</b>          0: 无作用；          1: 触发输入 (TRGI) 上的事件被延迟了，以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。</p>																
位 6: 4	<p><b>TRGSEL[2: 0]: 触发选择 (Trigger selection)</b>          这 3 位选择用于同步计数器的触发输入。</p> <table border="0"> <tbody> <tr> <td>000: 内部触发 0 (ITR0), TMR1</td> <td>100: TI1 的边沿检测器 (TI1F_ED)</td> </tr> <tr> <td>001: 内部触发 1 (ITR1), TMR2</td> <td>101: 滤波后的定时器输入 1 (TI1FP1)</td> </tr> <tr> <td>010: 内部触发 2 (ITR2), TMR3</td> <td>110: 滤波后的定时器输入 2 (TI2FP2)</td> </tr> <tr> <td>011: 内部触发 3 (ITR3), TMR4</td> <td>111: 外部触发输入 (ETRF)</td> </tr> </tbody> </table> <p>关于每个定时器中 ITRx 的细节，参见表 10-4。          注：这些位只能在未用到（如 SMSEL=000）时被改变，以避免在改变时产生错误的边沿检测。</p>	000: 内部触发 0 (ITR0), TMR1	100: TI1 的边沿检测器 (TI1F_ED)	001: 内部触发 1 (ITR1), TMR2	101: 滤波后的定时器输入 1 (TI1FP1)	010: 内部触发 2 (ITR2), TMR3	110: 滤波后的定时器输入 2 (TI2FP2)	011: 内部触发 3 (ITR3), TMR4	111: 外部触发输入 (ETRF)								
000: 内部触发 0 (ITR0), TMR1	100: TI1 的边沿检测器 (TI1F_ED)																
001: 内部触发 1 (ITR1), TMR2	101: 滤波后的定时器输入 1 (TI1FP1)																
010: 内部触发 2 (ITR2), TMR3	110: 滤波后的定时器输入 2 (TI2FP2)																
011: 内部触发 3 (ITR3), TMR4	111: 外部触发输入 (ETRF)																
位 3	保留，始终读为 0。																
位 2: 0	<p><b>SMSEL[2: 0]: 从模式选择 (Slave mode selection)</b>          当选择了外部信号，触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关（见输入控制寄存器和控制寄存器的说明）</p> <table border="0"> <tbody> <tr> <td>000: 关闭从模式 – 如果 CNTEN=1，则预分频器直接由内部时钟驱动。</td> </tr> <tr> <td>001: 编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。</td> </tr> <tr> <td>010: 编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。</td> </tr> <tr> <td>011: 编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。</td> </tr> <tr> <td>100: 复位模式 – 选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。</td> </tr> <tr> <td>101: 门控模式 – 当触发输入 (TRGI) 为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止（但不复位）。计数器的启动和停止都是受控的。</td> </tr> <tr> <td>110: 触发模式 – 计数器在触发输入 TRGI 的上升沿启动（但不复位），只有计数器的启动是受控的。</td> </tr> <tr> <td>111: 外部时钟模式 1 – 选中的触发输入 (TRGI) 的上升沿驱动计数器。</td> </tr> </tbody> </table> <p>注：如果 TI1F_EN 被选为触发输入 (TRGSEL=100) 时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。</p>	000: 关闭从模式 – 如果 CNTEN=1，则预分频器直接由内部时钟驱动。	001: 编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。	010: 编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。	011: 编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。	100: 复位模式 – 选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。	101: 门控模式 – 当触发输入 (TRGI) 为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止（但不复位）。计数器的启动和停止都是受控的。	110: 触发模式 – 计数器在触发输入 TRGI 的上升沿启动（但不复位），只有计数器的启动是受控的。	111: 外部时钟模式 1 – 选中的触发输入 (TRGI) 的上升沿驱动计数器。								
000: 关闭从模式 – 如果 CNTEN=1，则预分频器直接由内部时钟驱动。																	
001: 编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。																	
010: 编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。																	
011: 编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。																	
100: 复位模式 – 选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。																	
101: 门控模式 – 当触发输入 (TRGI) 为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止（但不复位）。计数器的启动和停止都是受控的。																	
110: 触发模式 – 计数器在触发输入 TRGI 的上升沿启动（但不复位），只有计数器的启动是受控的。																	
111: 外部时钟模式 1 – 选中的触发输入 (TRGI) 的上升沿驱动计数器。																	

表格 10-4 TMRx 内部触发连接<sup>(1)</sup>

从定时器	ITR0 (TRGSEL = 000)	ITR1 (TRGSEL = 001)	ITR2 (TRGSEL = 010)	ITR3 (TRGSEL = 011)
TMR3	TMR1_TRGO	0	TMR15_TRGO	0

注意：如果某个产品中没有相应的定时器，则对应的触发信号 ITRx 也不存在。

#### 10.2.4.4 DMA/中断使能寄存器 (TMRx\_DIE)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	TRG DE	保留	C4D E	C3D E	C2D E	C1D E	UEV DE	保留	TRG IE	保留	C4I E	C3I E	C2I E	C1I E	UEV IE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	保留, 始终读为 0。
位 14	<b>TRGDE:</b> 允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求; 1: 允许触发 DMA 请求。
位 13	保留, 始终读为 0。
位 12	<b>C4DE:</b> 允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较 4 的 DMA 请求; 1: 允许捕获/比较 4 的 DMA 请求。
位 11	<b>C3DE:</b> 允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较 3 的 DMA 请求; 1: 允许捕获/比较 3 的 DMA 请求。
位 10	<b>C2DE:</b> 允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求; 1: 允许捕获/比较 2 的 DMA 请求。
位 9	<b>C1DE:</b> 允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求; 1: 允许捕获/比较 1 的 DMA 请求。
位 8	<b>UEVDE:</b> 允许更新的 DMA 请求 (Update DMA request enable) 0: 禁止更新的 DMA 请求; 1: 允许更新的 DMA 请求。
位 7	保留, 始终读为 0。
位 6	<b>TRGIE:</b> 触发中断使能 (Trigger interrupt enable) 0: 禁止触发中断; 1: 使能触发中断。
位 5	保留, 始终读为 0。
位 4	<b>C4IE:</b> 允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0: 禁止捕获/比较 4 中断; 1: 允许捕获/比较 4 中断。
位 3	<b>C3IE:</b> 允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较 3 中断; 1: 允许捕获/比较 3 中断。
位 2	<b>C2IE:</b> 允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断; 1: 允许捕获/比较 2 中断。
位 1	<b>C1IE:</b> 允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断; 1: 允许捕获/比较 1 中断。
位 0	<b>UEVIE:</b> 允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

### 10.2.4.5 状态寄存器 (TMRx\_STS)

偏移地址: 0x10

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	C4OF F	C3OF F	C2OF F	C1OF F	保留	保留	TRGIF	保留	C4IF F	C3IF F	C2IF F	C1IF F	UEV IF		

res            rw            rw            rw            rw            res            rw            res            rw            rw            rw            rw            rw            rw            rw

位 15: 13	保留, 始终读为 0。
位 12	<b>C4OF:</b> 捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 C1OF 描述。
位 11	<b>C3OF:</b> 捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 C1OF 描述。
位 10	<b>C2OF:</b> 捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 C1OF 描述。
位 9	<b>C1OF:</b> 捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置'1'。写'0'可清除该位。 0: 无重复捕获产生; 1: 当计数器的值被捕获到 TMRx_CC1 寄存器时, C1IF 的状态已经为'1'。
位 8: 7	保留, 始终读为 0。
位 6	<b>TRGIF:</b> 触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置'1'。它由软件清'0'。 0: 无触发器事件产生; 1: 触发器中断等待响应。
位 5	保留, 始终读为 0。
位 4	<b>C4IF:</b> 捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 C1IF 描述。
位 3	<b>C3IF:</b> 捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 C1IF 描述。
位 2	<b>C2IF:</b> 捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 C1IF 描述。
位 1	<b>C1IF:</b> 捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) <b>如果通道 CC1 配置为输出模式:</b> 当计数器值与比较值匹配时该位由硬件置'1', 但在中心对称模式下除外 (参考 TMRx_CTRL1 寄存器的 CMSEL 位)。它由软件清'0'。 0: 无匹配发生; 1: TMRx_CNT 的值与 TMRx_CC1 的值匹配。 <b>如果通道 CC1 配置为输入模式:</b> 当捕获事件发生时该位由硬件置'1', 它由软件清'0'或通过读 TMRx_CC1 清'0'。 0: 无输入捕获产生; 1: 计数器值已被捕获 (拷贝) 至 TMRx_CC1 (在 IC1 上检测到与所选极性相同的边沿)。

位 0	<p><b>UEVIF:</b> 更新中断标记（Update interrupt flag） 当产生更新事件时该位由硬件置'1'。它由软件清'0'。</p> <p>0: 无更新事件产生； 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1':</p> <ul style="list-style-type: none"> <li>- 若 TMRx_CTRL1 寄存器的 UEVDIS=0、UEVRS=0，当 TMRx_EVEG 寄存器的 UEVG=1 时产生更新事件（软件对计数器 CNT 重新初始化）；</li> <li>- 若 TMRx_CTRL1 寄存器的 UEVDIS=0、UEVRS=0，当计数器 CNT 被触发事件重初始化时产生更新事件。（参考同步控制寄存器的说明）</li> </ul>
-----	---

#### 10.2.4.6 事件产生寄存器 (TMRx\_EVEG)

偏移地址: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				保留					TRG G	保留	C4G	C3G	C2G	C1G	UEV G
				res					rw	res	rw	rw	rw	rw	rw

位 15: 7	保留, 始终读为 0。
位 6	<p><b>TRGG:</b> 产生触发事件 (Trigger generation) 该位由软件置'1', 用于产生一个触发事件, 由硬件自动清'0'。 0: 无动作; 1: TMRx_STS 寄存器的 TRGIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。</p>
位 5	保留, 始终读为 0。
位 4	<p><b>C4G:</b> 产生捕获/比较 4 事件 (Capture/compare 4 generation) 参考 C1G 描述。</p>
位 3	<p><b>C3G:</b> 产生捕获/比较 3 事件 (Capture/compare 3 generation) 参考 C1G 描述。</p>
位 2	<p><b>C2G:</b> 产生捕获/比较 2 事件 (Capture/compare 2 generation) 参考 C1G 描述。</p>
位 1	<p><b>C1G:</b> 产生捕获/比较 1 事件 (Capture/compare 1 generation) 该位由软件置'1', 用于产生一个捕获/比较事件, 由硬件自动清'0'。 0: 无动作; 1: 在通道 CC1 上产生一个捕获/比较事件: <b>若通道 CC1 配置为输出:</b> 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。 <b>若通道 CC1 配置为输入:</b> 当前的计数器值捕获至 TMRx_CC1 寄存器; 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 C1IF 已经为 1, 则设置 C1OF=1。</p>
位 0	<p><b>UEVG:</b> 产生更新事件 (Update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0 (向上计数) 则计数器被清'0', 若 DIR=1 (向下计数) 则计数器取 TMRx_AR 的值。</p>

#### 10.2.4.7 捕获/比较模式寄存器1 (TMRx CCM1)

偏移地址: 0x18

复位值: 0x0000

通道可用于输入（捕获模式）或输出（比较模式），通道的方向由相应的 **CxSEL** 定义。该寄存器其它位的作用在输入和输出模式下不同。**OCxx** 描述了通道在输出模式下的功能，**ICxx** 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

#### 输出比较模式：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2 DIS	OC2MODE[2: 0]	OC2 PEN	OC2 FEN	C2SEL[1: 0]	OC1 DIS	OC1MODE[2: 0]	OC1 PEN	OC1 FEN	C1SEL[1: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>OC2DIS:</b> 输出比较 2 清 0 使能 （Output compare 2 clear enable）
位 14: 12	<b>OC2MODE[2: 0]:</b> 输出比较 2 模式 （Output compare 2 mode）
位 11	<b>OC2PEN:</b> 输出比较 2 预装载使能 （Output compare 2 preload enable）
位 10	<b>OC2FEN:</b> 输出比较 2 快速使能 （Output compare 2 fast enable）
位 9: 8	<b>C2SEL[1: 0]:</b> 捕获/比较 2 选择 （Capture/Compare 2 selection） 该位定义通道的方向（输入/输出），及输入脚的选择： 00: CC2 通道被配置为输出； 01: CC2 通道被配置为输入，IC2 映射在 TI2 上； 10: CC2 通道被配置为输入，IC2 映射在 TI1 上； 11: CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TMRx_SMC 寄存器的 TRGSEL 位选择）。 注：C2SEL 仅在通道关闭时（TMRx_CCE 寄存器的 C2EN='0'）才是可写的。
位 7	<b>OC1DIS:</b> 输出比较 1 清 0 使能 （Output compare 1 clear enable） 0: OC1REF 不受 ETRF 输入的影响； 1: 一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
位 6: 4	<b>OC1MODE[2: 0]:</b> 输出比较 1 模式 （Output compare 1 enable） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1 的值。OC1REF 是高电平有效，而 OC1 的有效电平取决于 C1P 位。 000: 冻结。输出比较寄存器 TMRx_CC1 与计数器 TMRx_CNT 间的比较对 OC1REF 不起作用； 001: 匹配时设置通道 1 为有效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为低。 011: 翻转。当 TMRx_CC1=TMRx_CNT 时，翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1— 在向上计数时，一旦 TMRx_CNT<TMRx_CC1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TMRx_CNT>TMRx_CC1 时通道 1 为无效电平（OC1REF=0），否则为有效电平（OC1REF=1）。 111: PWM 模式 2— 在向上计数时，一旦 TMRx_CNT<TMRx_CC1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TMRx_CNT>TMRx_CC1 时通道 1 为有效电平，否则为无效电平。 注：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。

位 3	<b>OC1PEN:</b> 输出比较 1 预装载使能 (Output compare 1 preload enable) 0: 禁止 TMRx_CC1 寄存器的预装载功能, 可随时写入 TMRx_CC1 寄存器, 并且新写入的数值立即起作用。 1: 开启 TMRx_CC1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TMRx_CC1 的预装载值在更新事件到来时被传送至当前寄存器中。 注: 仅在单脉冲模式下 (TMRx_CTRL1 寄存器的 OPMODE='1'), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。
位 2	<b>OC1FEN:</b> 输出比较 1 快速使能 (Output compare 1 fast enable) 该位用于加快 CC 输出对触发器输入事件的响应。 0: 根据计数器与 CC1 的值, CC1 正常操作, 即使触发器是打开的。当触发器的输入出现一个有效沿时, 激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此, OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 该位只在通道被配置成 PWM1 或 PWM2 模式时起作用。
位 1: 0	<b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN='0') 才是可写的。

## 输入捕获模式:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2DF[3: 0]		IC2DIV[1: 0]		C2SEL[1: 0]		IC1DF[3: 0]		IC1DIV[1: 0]		C1SEL[1: 0]					
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 12	<b>IC2DF[3: 0]:</b> 输入捕获 2 滤波器 (Input capture 2 filter)
位 11: 10	<b>IC2DIV[1: 0]:</b> 输入/捕获 2 预分频器 (input capture 2 prescaler)
位 9: 8	<b>C2SEL[1: 0]:</b> 捕获/比较 2 选择 (Capture/compare 2 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C2SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C2EN='0') 才是可写的。
位 7: 4	<b>IC1DF[3: 0]:</b> 输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成, 它记录到 N 个事件后会产生一个输出的跳变: 0000: 无滤波器, 以 $f_{DTS}$ 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8

位 3: 2	<b>IC1DIV[1: 0]:</b> 输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入 (IC1) 的预分频系数。 一旦 C1EN='0' (TMRx_CCE 寄存器中)，则预分频器复位。 00: 无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01: 每 2 个事件触发一次捕获； 10: 每 4 个事件触发一次捕获； 11: 每 8 个事件触发一次捕获。
位 1: 0	<b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC1 通道被配置为输出； 01: CC1 通道被配置为输入，IC1 映射在 TI1 上； 10: CC1 通道被配置为输入，IC1 映射在 TI2 上； 11: CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注：C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN='0') 才是可写的。

#### 10.2.4.8 捕获/比较模式寄存器2 (TMRx\_CCM2)

偏移地址: 0x1C

复位值: 0x0000 参看以上 CCM1 寄存器的描述

输出比较模式：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4 DIS	OC4MODE[2: 0]	OC4 PEN	OC4 FEN	C4SEL[1: 0]	OC3 DIS	OC3MODE[2: 0]	OC3 PEN	OC3 FEN	C3SEL[1: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>OC4DIS:</b> 输出比较 4 清 0 使能 (Output compare 4 clear enable)
位 14: 12	<b>OC4MODE[2: 0]:</b> 输出比较 4 模式 (Output compare 4 mode)
位 11	<b>OC4PEN:</b> 输出比较 4 预装载使能 (Output compare 4 preload enable)
位 10	<b>OC4FEN:</b> 输出比较 4 快速使能 (Output compare 4 fast enable)
位 9: 8	<b>C4SEL[1: 0]:</b> 捕获/比较 4 选择 (Capture/Compare 4 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC4 通道被配置为输出； 01: CC4 通道被配置为输入，IC4 映射在 TI4 上； 10: CC4 通道被配置为输入，IC4 映射在 TI3 上； 11: CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注：C4SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C4EN='0') 才是可写的。
位 7	<b>OC3DIS:</b> 输出比较 3 清 0 使能 (Output compare 3 clear enable)
位 6: 4	<b>OC3MODE[2: 0]:</b> 输出比较 3 模式 (Output compare 3 mode)
位 3	<b>OC3PEN:</b> 输出比较 3 预装载使能 (Output compare 3 preload enable)
位 2	<b>OC3FEN:</b> 输出比较 3 快速使能 (Output compare 3 fast enable)
位 1: 0	<b>C3SEL[1: 0]:</b> 捕获/比较 3 选择 (Capture/Compare 3 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC3 通道被配置为输出； 01: CC3 通道被配置为输入，IC3 映射在 TI3 上； 10: CC3 通道被配置为输入，IC3 映射在 TI4 上； 11: CC3 通道被配置为输入，IC3 映射在 TRGI 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注：C3SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C3EN='0') 才是可写的。

## 输入捕获模式:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				IC4DF[3: 0]	IC4DIV[1: 0]	C4SEL[1: 0]		IC3DF[3: 0]		IC3DIV[1: 0]		C3SEL[1: 0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 12	<b>IC4DF[3: 0]:</b> 输入捕获 4 滤波器 (Input capture 4 filter)														
位 11: 10	<b>IC4DIV[1: 0]:</b> 输入/捕获 4 预分频器 (input capture 4 prescaler)														
位 9: 8	<b>C4SEL[1: 0]:</b> 捕获/比较 4 选择 (Capture/compare 4 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC4 通道被配置为输出； 01: CC4 通道被配置为输入，IC4 映射在 TI4 上； 10: CC4 通道被配置为输入，IC4 映射在 TI3 上； 11: CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TMRx_SMC 寄存器的 TRGSEL 位选择）。 注：C4SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C4EN='0') 才是可写的。														
位 7: 4	<b>IC3DF[3: 0]:</b> 输入捕获 3 滤波器 (Input capture 3 filter)														
位 3: 2	<b>IC3DIV[1: 0]:</b> 输入/捕获 3 预分频器 (Input capture 3 prescaler)														
位 1: 0	<b>C3SEL[1: 0]:</b> 捕获/比较 3 选择 (Capture/Compare 3 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC3 通道被配置为输出； 01: CC3 通道被配置为输入，IC3 映射在 TI3 上； 10: CC3 通道被配置为输入，IC3 映射在 TI4 上； 11: CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TMRx_SMC 寄存器的 TRGSEL 位选择）。 注：C3SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C3EN='0') 才是可写的。														

## 10.2.4.9 捕获/比较使能寄存器 (TMRx\_CCE)

偏移地址: 0x20

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	C4P	C4E N	保留	C3P	C3E N	保留	C2P	C2E N	保留	C1P	C1E N				
res	rw	rw	res	rw	rw	res	rw	rw	res	rw	rw	res	rw	rw	rw
位 15: 14	保留，始终读为 0。														
位 13	<b>C4P:</b> 输入/捕获 4 输出极性 (Capture/Compare 4 output polarity) 参考 C1P 的描述。														
位 12	<b>C4EN:</b> 输入/捕获 4 输出使能 (Capture/Compare 4 output enable) 参考 C1EN 的描述。														
位 11: 10	保留，始终读为 0。														
位 9	<b>C3P:</b> 输入/捕获 3 输出极性 (Capture/Compare 3 output polarity) 参考 C1P 的描述。														
位 8	<b>C3EN:</b> 输入/捕获 3 输出使能 (Capture/Compare 3 output enable) 参考 C1EN 的描述。														
位 7: 6	保留，始终读为 0。														

位 5	<b>C2P:</b> 输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 C1P 的描述。
位 4	<b>C2EN:</b> 输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 C1EN 的描述。
位 3: 2	保留, 始终读为 0。
位 1	<b>C1P:</b> 输入/捕获 1 输出极性 (Capture/Compare 1 output polarity) <b>CC1 通道配置为输出:</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1 通道配置为输入:</b> 该位选择是 IC1 还是 IC1 的反相信号作为触发或捕获信号。 0: 不反相: 捕获发生在 IC1 的上升沿; 当用作外部触发器时, IC1 不反相。 1: 反相: 捕获发生在 IC1 的下降沿; 当用作外部触发器时, IC1 反相。
位 0	<b>C1EN:</b> 输入/捕获 1 输出使能 (Capture/Compare 1 output enable) <b>CC1 通道配置为输出:</b> 0: 关闭 - OC1 禁止输出。 1: 开启 - OC1 信号输出到对应的输出引脚。 <b>CC1 通道配置为输入:</b> 该位决定了计数器的值是否能捕获入 TMRx_CC1 寄存器。 0: 捕获禁止; 1: 捕获使能。

表格 10-5 标准 OCx 通道的输出控制位

CxEN 位	OCx 输出状态
0	禁止输出 (OCx=0, OCx_EN=0)
1	OCx = OCxREF + 极性, OCx_EN=1

注意: 连接到标准 OCx 通道的外部 I/O 引脚状态, 取决于 OCx 通道状态和 GPIO 以及 AFIO 寄存器。

#### 10.2.4.10 计数器 (TMRx\_CNT)

偏移地址: 0x24

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31: 16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 16		<b>CNT[31: 16]:</b> 计数器的值 (Counter value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位), CNT 被扩展为 32 位。													
位 15: 0		<b>CNT[15: 0]:</b> 计数器的值 (Counter value)													

#### 10.2.4.11 预分频器 (TMRx\_DIV)

偏移地址: 0x28

复位值: 0x0000

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

位 15: 0	<b>DIV[15: 0]:</b> 预分频器的值 (Prescaler value) 计数器的时钟频率 CK_CNT 等于 $f_{CK\_DIV}/(DIV[15: 0]+1)$ 。 DIV 包含了当更新事件产生时装入当前预分频器寄存器的值。

#### 10.2.4.12 自动重装载寄存器 (TMRx\_AR)

偏移地址: 0x2C

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AR[31: 16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 16	<b>AR[31: 16]:</b> 自动重装载的值 (Auto reload value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)， AR 被扩展为 32 位。
位 15: 0	<b>AR[15: 0]:</b> 自动重装载的值 (Auto reload value) AR 包含了将要传送至实际的自动重装载寄存器的数值。当自动重装载的值为空时，计数器不工作。 详细参考 <a href="#">10.2.3.1</a> 节：有关 AR 的更新和动作。

#### 10.2.4.13 捕获/比较寄存器1 (TMRx\_CC1)

偏移地址: 0x34

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CC1[31: 16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC1[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 16	<b>CC1[31: 16]:</b> 捕获/比较 1 的值 (Capture/Compare 1 value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)， CC1 被扩展为 32 位。
----------	---

位 15: 0	<b>CC1[15: 0]:</b> 捕获/比较 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出: CC1 包含了装入当前捕获/比较 1 寄存器的值 (预装载值)。 如果在 TMRx_CCM1 寄存器 (OC1PEN 位) 中未选择预装载特性, 写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比寄存器 1 中。 当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入: CC1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。
---------	--

#### 10.2.4.14 捕获/比较寄存器2 (TMRx\_CC2)

偏移地址: 0x38

复位值: 0x0000

位 31: 16	<b>CC2[31: 16]:</b> 捕获/比较 2 的值 (Capture/Compare 2 value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位), CC2 被扩展为 32 位。
位 15: 0	<b>CC2[15: 0]:</b> 捕获/比较 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出: CC2 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。 如果在 TMRx_CCM2 寄存器 (OC2PEN 位) 中未选择预装载特性, 写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 2 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入: CC2 包含了由上一次输入捕获 2 事件 (IC2) 传输的计数器值。

#### 10.2.4.15 捕获/比较寄存器3 (TMRx\_CC3)

偏移地址: 0x3C

复位值: 0x0000

位 31: 16	<b>CC3[31: 16]:</b> 捕获/比较 3 的值 (Capture/Compare3 value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位), CC3 被扩展为 32 位。
位 15: 0	<b>CC3[15: 0]:</b> 捕获/比较 3 的值 (Capture/Compare3 value)

位 15: 0	<b>CC3[15: 0]:</b> 捕获/比较 3 的值 (Capture/Compare 3value) 若 CC3 通道配置为输出: CC3 包含了装入当前捕获/比较 3 寄存器的值 (预装载值)。 如果在 TMRx_CCM3 寄存器 (OC3PEN 位) 中未选择预装载特性, 写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较寄存器 3 中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入: CC3 包含了由上一次输入捕获 3 事件 (IC3) 传输的计数器值。

#### 10.2.4.16 捕获/比较寄存器4 (TMRx\_CC4)

偏移地址: 0x40

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CC4[31: 16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 16	<b>CC4[31: 16]:</b> 捕获/比较 4 的值 (Capture/Compare4 value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位), CC4 被扩展为 32 位。														
	<b>CC4[15: 0]:</b> 捕获/比较 4 的值 (Capture/Compare 4value) 若 CC4 通道配置为输出: CC4 包含了装入当前捕获/比较 4 寄存器的值 (预装载值)。 如果在 TMRx_CCM4 寄存器 (OC4PEN 位) 中未选择预装载特性, 写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较寄存器 4 中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入: CC4 包含了由上一次输入捕获 4 事件 (IC4) 传输的计数器值。														

#### 10.2.4.17 DMA控制寄存器 (TMRx\_DMAMC)

偏移地址: 0x48

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		DBLEN[4: 0]				保留		ADDR[4: 0]							
res	rw	rw	rw	rw	rw	rw	res	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 13		保留, 始终读为 0。													
位 12: 8		<b>DBLEN[4: 0]:</b> DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度 (当对 TMRx_DMABA 寄存器进行读或写时, 定时器则进行一次连续传送), 即: 定义传输的字节数目: 00000: 1 个字节      00001: 2 个字节 00010: 3 个字节      ..... .....      10001: 18 个字节													

位 7: 5	保留, 始终读为 0。
位 4: 0	<b>ADDR[4: 0]: DMA 基地址 (DMA base address)</b> 这些位定义了 DMA 在连续模式下的基地址 (当对 TMRx_DMABA 寄存器进行读或写时), ADDR 定义为从 TMRx_CTRL1 寄存器所在地址开始的偏移量: 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_SMC, .....

#### 10.2.4.18 连续模式的DMA地址 (TMRx\_DMABA)

偏移地址: 0x4C

复位值: 0x0000

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

位 15: 0	<b>DMABA[15: 0]: DMA 连续传送寄存器 (DMA register for burst accesses)</b> 对 TMRx_DMABA 寄存器的读或写会导致对以下地址所在寄存器的存取操作: TMRx_CTRL1 地址 + ADDR + DMA 索引, 其中: “TMRx_CTRL1 地址”是控制寄存器 1 (TMRx_CTRL1) 所在的地址; “ADDR”是 TMRx_DMAC 寄存器中定义的基地址; “DMA 索引”是由 DMA 自动控制的偏移量, 它取决于 TMRx_DMAC 寄存器中定义的 DBLEN。
---------	---

## 10.3 通用定时器 (TMR14)

### 10.3.1 TMRx简介

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

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

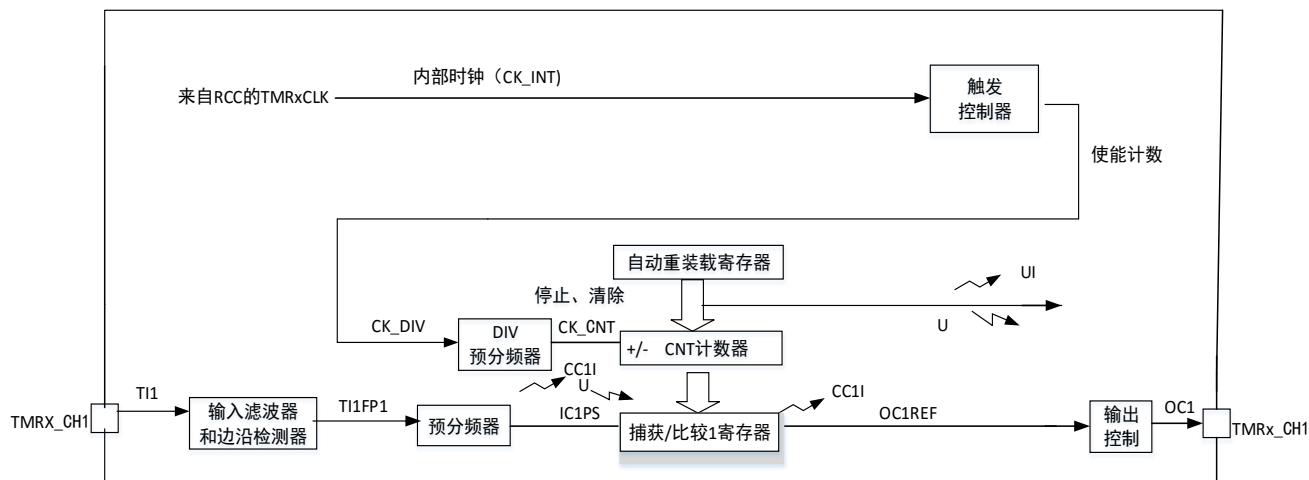
### 10.3.2 TMRx主要功能

#### 10.3.2.1 TMR14主要功能

通用 TMRx (TMR14) 定时器功能包括：

- 16位向上自动装载计数器
- 16位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为1~65536之间的任意数值
- 1个独立通道：
  - 输入捕获
  - 输出比较
  - PWM生成（边缘对齐模式）
  - 单脉冲模式输出
- 使用外部信号控制定时器和定时器互连的同步电路
- 如下事件发生时产生中断：
  - 更新：计数器向上溢出，计数器初始化（通过软件或者内部）
  - 触发事件（计数器启动、停止、初始化或者由内部触发计数）
  - 输入捕获
  - 输出比较

图表 10-57 通用定时器TMR14框图



注意： 根据控制位的设定，在  $U$  事件时传送预加载寄存器的内容至工作寄存器  
 ↗ 事件  
 ↗ 中断

### 10.3.3 TMRx功能描述

#### 10.3.3.1 时基单元

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

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

时基单元包含：

- 计数器寄存器（TMRx\_CNT）
- 预分频器寄存器（TMRx\_DIV）
- 自动装载寄存器（TMRx\_AR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TMRx\_CTRL1 寄存器中的自动装载预装载使能位(ARPEN)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件并当 TMRx\_CTRL1 寄存器中的 UEVDIS 位等于'0'时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK\_CNT 驱动，仅当设置了计数器 TMRx\_CTRL1 寄存器中的计数器使能位 (CNTEN) 时，CK\_CNT 才有效。（有关计数器使能的细节，请参见控制器的从模式描述）。

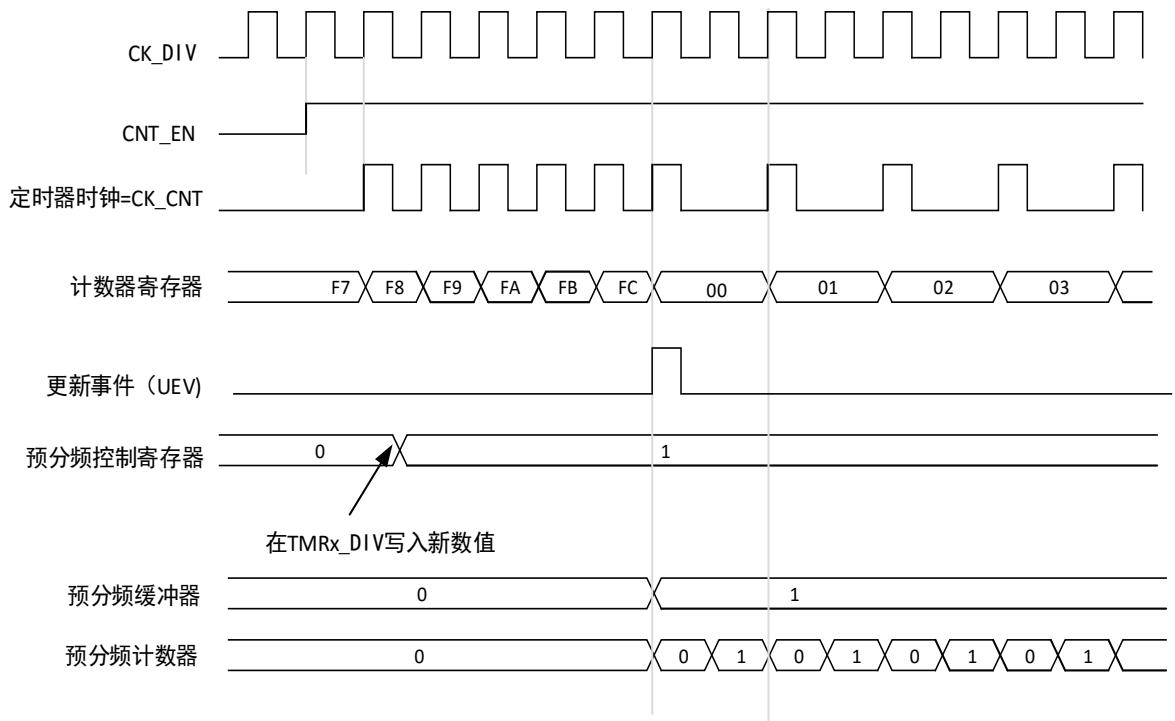
注意： 真正的计数器使能信号 CNT\_EN 是在 CNTEN 的一个时钟周期后被设置。

#### 预分频器

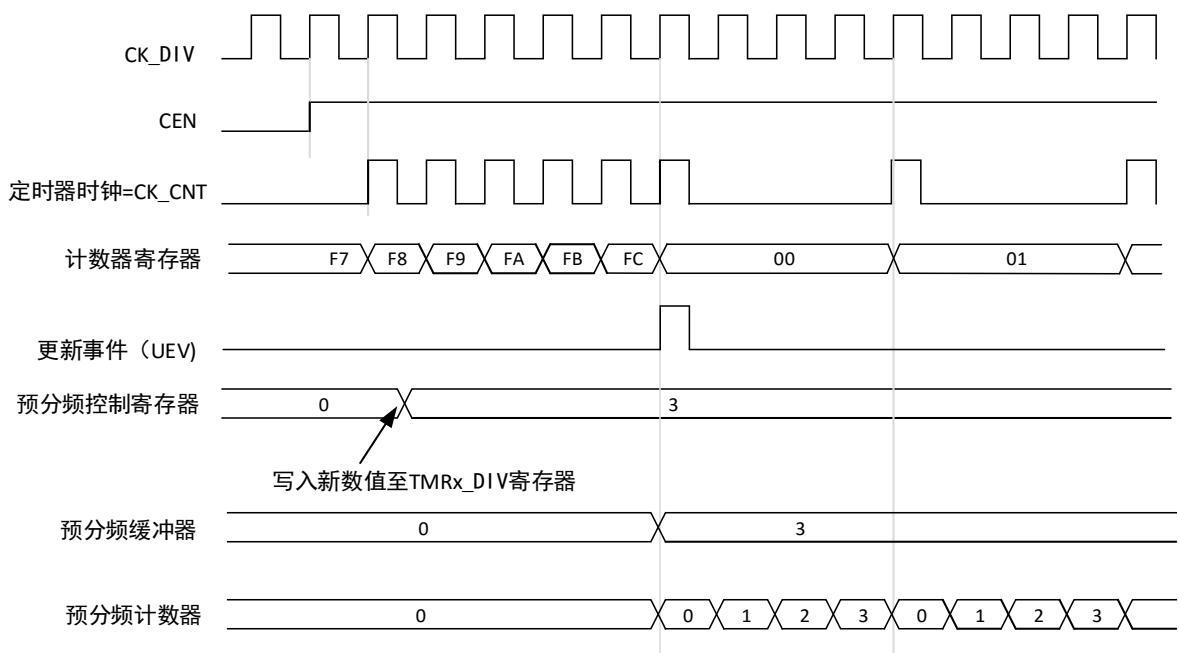
预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TMRx\_DIV 寄存器中的）16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器参数在下一次更新事件到来时被采用。

[图 10-58](#) 和 [图 10-59](#) 给出了在预分频器运行时，更改计数器参数的例子。

图表 10-58 当预分频器的参数从1变到2时，计数器的时序图



图表 10-59 当预分频器的参数从1变到4时，计数器的时序图



### 10.3.3.2 计数器模式

#### 向上计数模式

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

每次计数器溢出时可以产生更新事件，在 TMRx\_EVEG 寄存器中置 UEVG 位也同样可以产生一个更新事件。

设置 TMRx\_CTRL1 寄存器中的 UEVDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UEVDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，

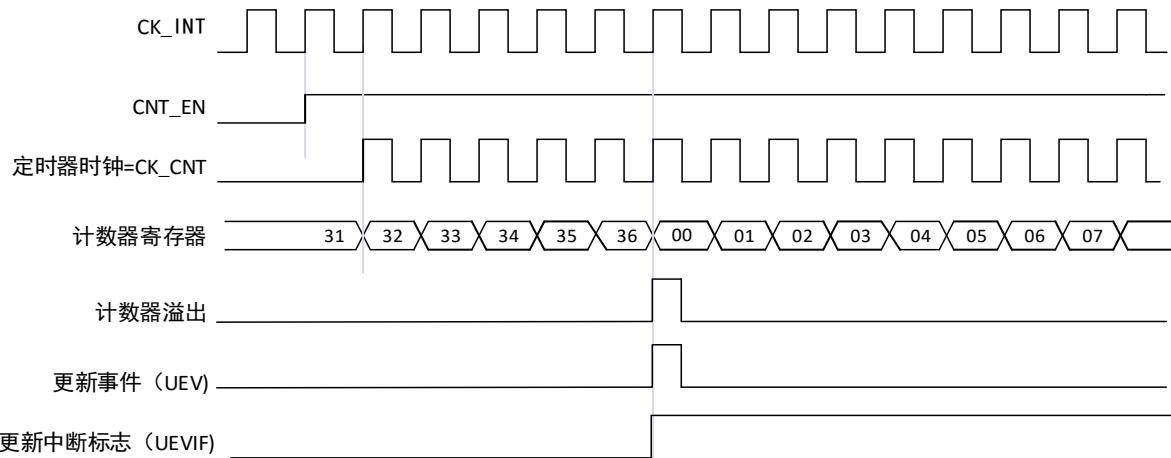
计数器仍会被清'0', 同时预分频器的计数也被清 0(但预分频系数不变)。此外, 如果设置了 TMRx\_CTRL1 寄存器中的 UEVRS 位(选择更新请求), 设置 UEVG 位将产生一个更新事件 UEV, 但硬件不设置 UEVIF 标志(即不产生中断或 DMA 请求); 这是为了避免在捕获模式下清除计数器时, 同时产生更新和捕获中断。

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

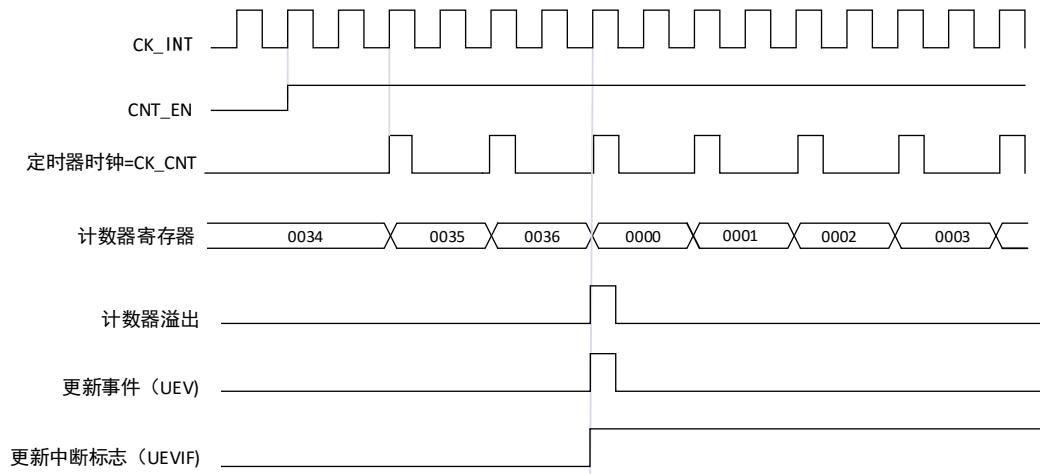
- 预分频器的缓冲区被置入预装载寄存器的值 (TMRx\_DIV 寄存器的内容)。
- 自动装载影子寄存器被重新置入预装载寄存器的值 (TMRx\_AR)。

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

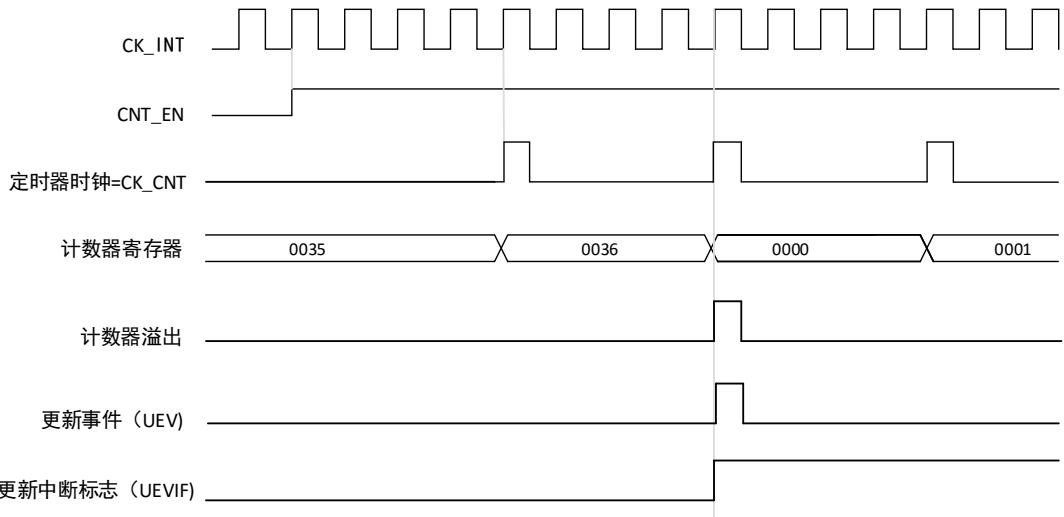
图表 10-60 计数器时序图, 内部时钟分频因子为 1



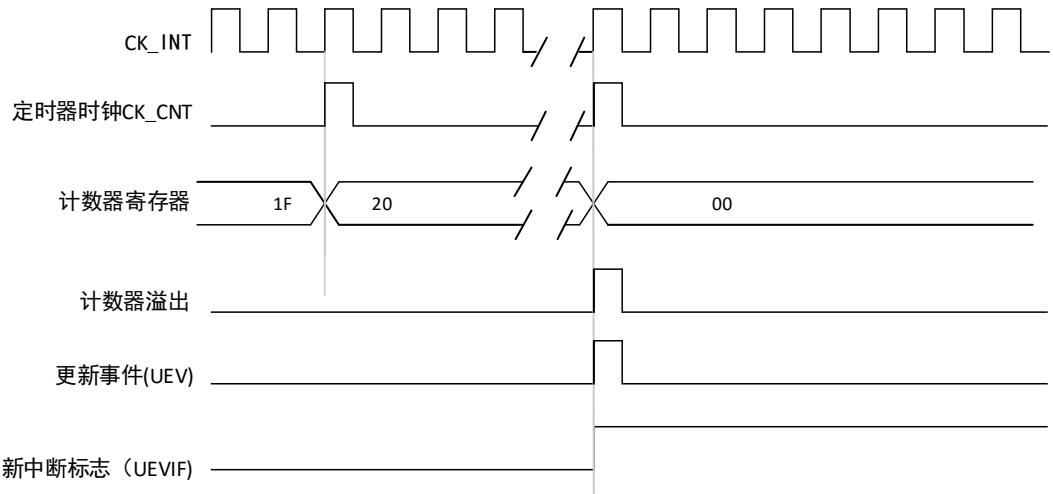
图表 10-61 计数器时序图, 内部时钟分频因子为 2



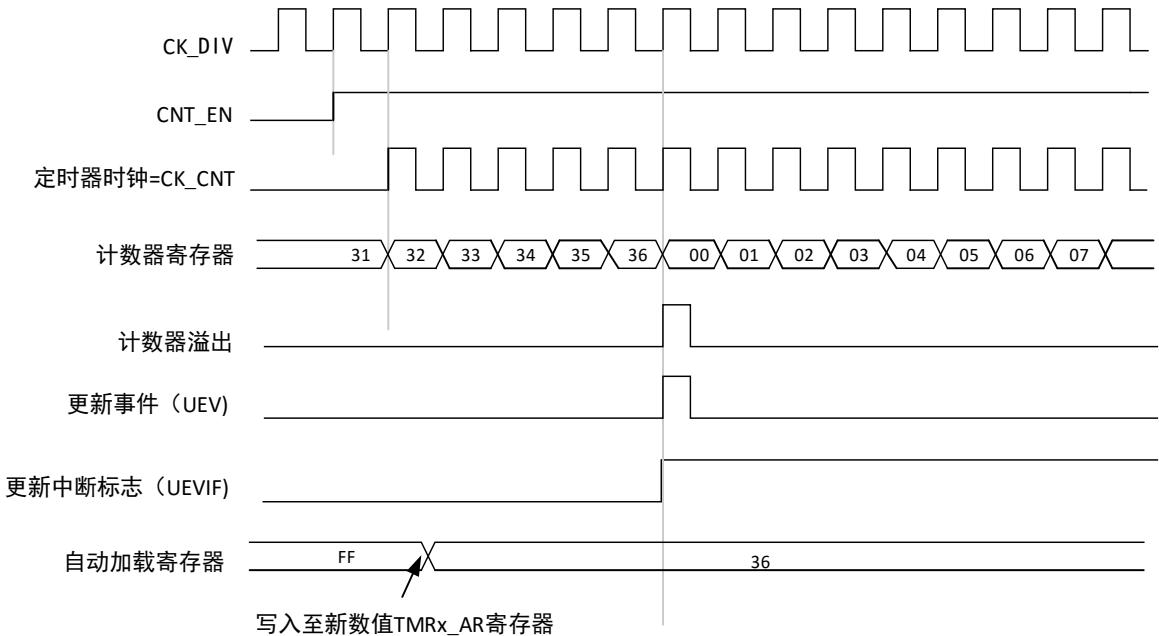
图表 10-62 计数器时序图, 内部时钟分频因子为 4



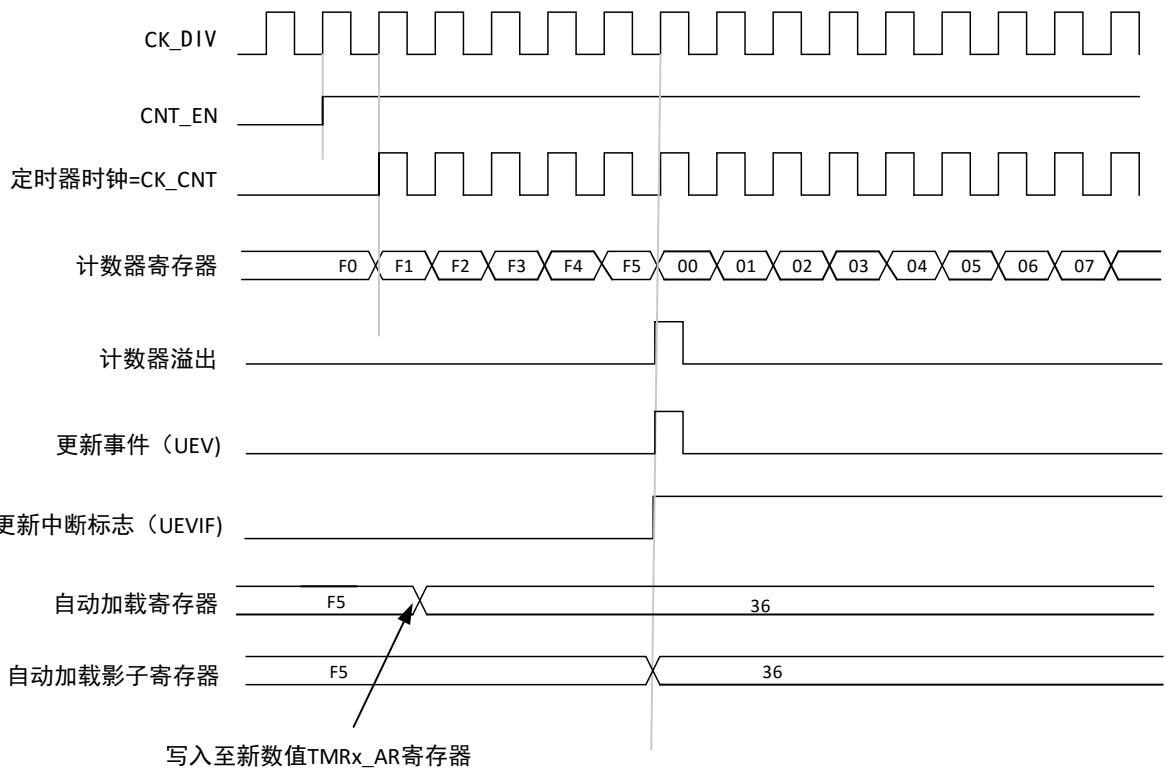
图表 10-63 计数器时序图，内部时钟分频因子为N



图表 10-64 计数器时序图，当ARPEN=0时的更新事件 (TMRx\_AR没有预装入)



图表 10-65 计数器时序图，当ARPEN=1时的更新事件 (预装入了TMRx\_AR)



### 10.3.3.3 时钟选择

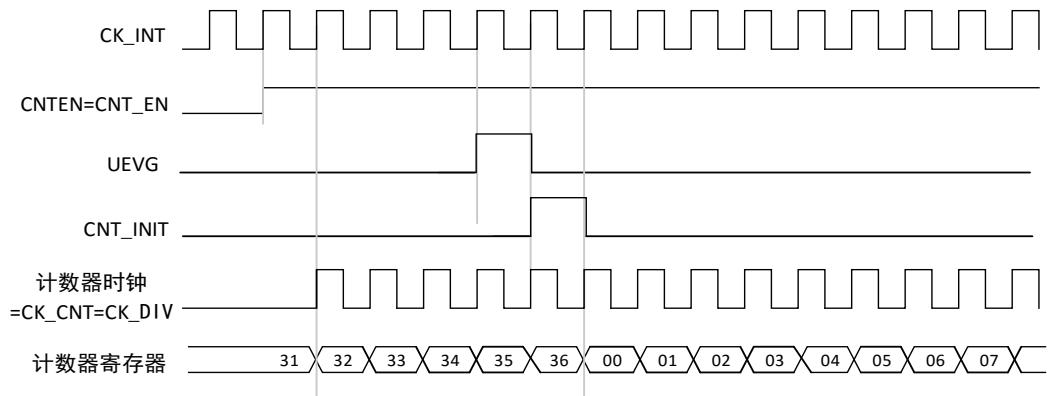
计数器时钟可由下列时钟源提供：

- 内部时钟 (CK\_INT)

#### 内部时钟源 (CK\_INT)

对于 TMR14，内部时钟源是默认的时钟源。

图表 10-66 一般模式下的控制电路，内部时钟分频因子为1

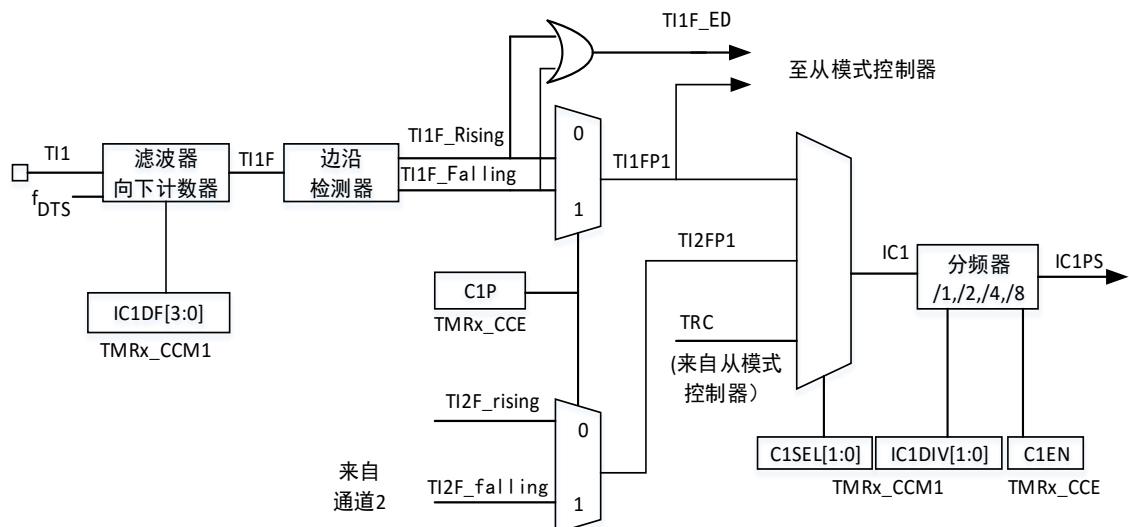


### 10.3.3.4 捕获/比较通道

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

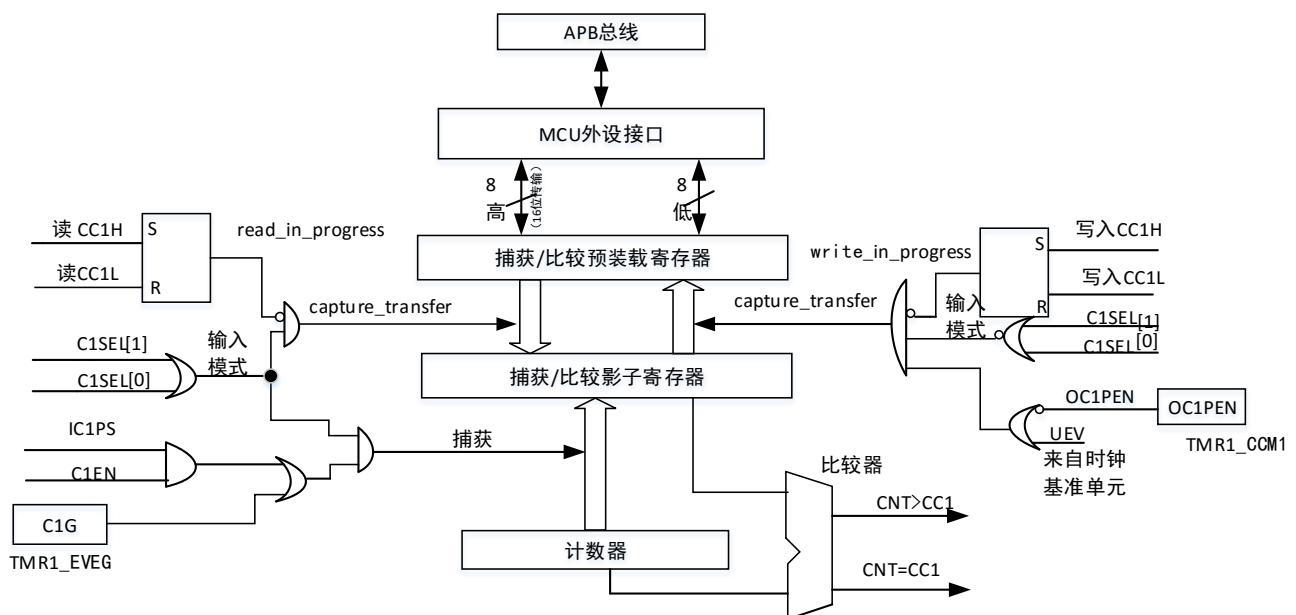
下面几张图是一个捕获/比较通道概览。输入部分对相应的 TI<sub>x</sub> 输入信号采样，并产生一个滤波后的信号 TI<sub>x</sub>F。然后，一个带极性选择的边缘检测器产生一个信号 (TI<sub>x</sub>FP<sub>x</sub>)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器。

图表 10-67 捕获/比较通道（如：通道1输入部分）

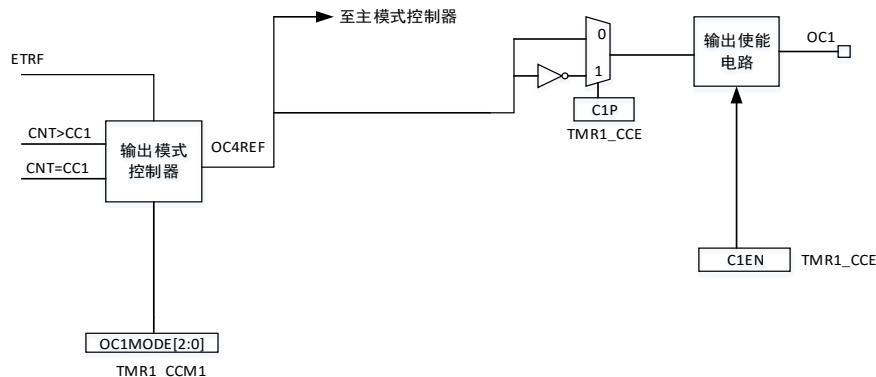


输出部分产生一个中间波形 OCxRef (高有效) 作为基准, 链的末端决定最终输出信号的极性。

图表 10-68 捕获/比较通道 1 的主电路



图表 10-69 捕获/比较通道的输出部分 (通道 1)



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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

### 10.3.3.5 输入捕获模式

在输入捕获模式下，当检测到 IC<sub>x</sub> 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TMR<sub>x</sub>\_CC<sub>x</sub>）中。当捕获事件发生时，相应的 CxIF 标志（TMR<sub>x</sub>\_STS 寄存器）被置'1'，如果使能了中断或者 DMA 操作，则将产生中断或者 DMA 操作。如果捕获事件发生时 CxIF 标志已经为高，那么重复捕获标志 CxOF（TMR<sub>x</sub>\_STS 寄存器）被置'1'。写 CxIF=0 可清除 CxIF，或读取存储在 TMR<sub>x</sub>\_CC<sub>x</sub> 寄存器中的捕获数据也可清除 CxIF。写 CxOF=0 可清除 CxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TMR<sub>x</sub>\_CC1 寄存器中，步骤如下：

- 选择有效输入端：TMR<sub>x</sub>\_CC1 必须连接到 TI1 输入，所以写入 TMR<sub>x</sub>\_CC1 寄存器中的 C1SEL=01，只要 C1SEL 不为 '00'，通道被配置为输入，并且 TM1\_CC1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TI<sub>x</sub> 时，输入滤波器控制位是 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 IC<sub>x</sub>DF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期。因此我们可以（以 f<sub>DTS</sub> 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TMR<sub>x</sub>\_CCM1 寄存器中写入 IC1DF=0011。
- 选择 TI1 通道的有效转换边沿，在 TMR<sub>x</sub>\_CCE 寄存器中写入 C1P=0（上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TMR<sub>x</sub>\_CCM1 寄存器的 IC1DIV=00）。
- 设置 TMR<sub>x</sub>\_CCE 寄存器的 C1EN=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TMR<sub>x</sub>\_CC1 寄存器。
- C1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 C1OF 未曾被清除，C1OF 也被置'1'。
- 如设置了 C1IE 位，则会产生一个中断。
- 为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注意：设置 TMR<sub>x</sub>\_EVEG 寄存器中相应的 CxG 位，可以通过软件产生输入捕获中断。

### 10.3.3.6 强置输出模式

在输出模式（TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中 CxSEL=00）下，输出比较信号（OC<sub>x</sub>REF 和相应的 OC<sub>x</sub>）能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 TMRx\_CCMx 寄存器中相应的 OCxMODE=101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CxP 极性位相反的值。

例如：CxP=0（OCx 高电平有效），则 OCx 被强置为高电平。

置 TMRx\_CCMx 寄存器中的 OCxMODE=100，可强置 OCxREF 信号为低。

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

### 10.3.3.7 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式（TMRx\_CCMx 寄存器中的 OCxMODE 位）和输出极性（TMRx\_CCE 寄存器中的 CxP 位）定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平（OCxMODE=000）、被设置成有效电平（OCxMODE=001）、被设置成无效电平（OCxMODE=010）或进行翻转（OCxMODE=011）。
- 设置中断状态寄存器中的标志位（TMRx\_STS 寄存器中的 CxIF 位）。
- 若设置了相应的中断屏蔽（TMRx\_DIE 寄存器中的 CxIE 位），则产生一个中断。

TMRx\_CCMx 中的 OCxPEN 位选择 TMRx\_CCx 寄存器是否需要使用预装载寄存器。

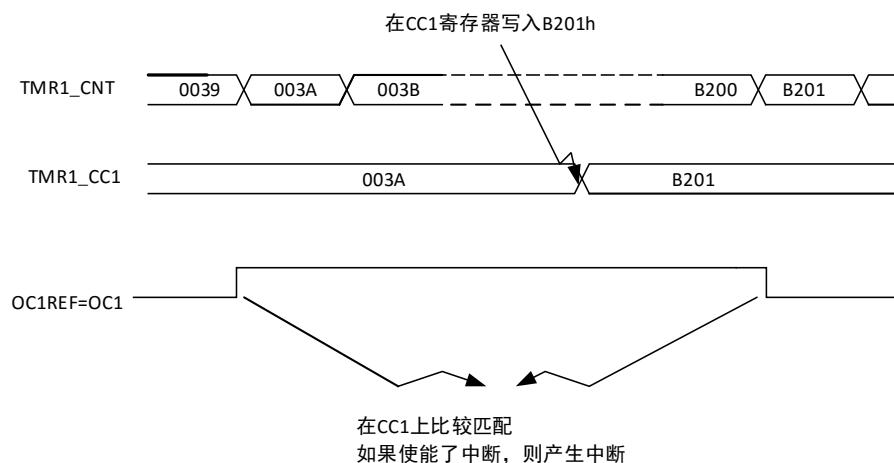
在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部，外部，预分频器）。
2. 将相应的数据写入 TMRx\_AR 和 TMRx\_CCx 寄存器中。
3. 如果要产生一个中断请求和/或一个 DMA 请求，设置 CxIE 位和/或 CxDE 位。
4. 选择输出模式，例如当计数器 CNT 与 CCx 匹配时翻转 OCx 的输出引脚，CCx 预装载未用，开启 OCx 输出且高电平有效，则必须设置 OCxMODE='011'、OCxPEN='0'、CxP='0' 和 CxEN='1'。
5. 设置 TMRx\_CTRL1 寄存器的 CNTEN 位启动计数器

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

图表 10-70 输出比较模式，翻转 OC1



### 10.3.3.8 PWM模式

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

在 TMRx\_CCMx 寄存器中的 OCxMODE 位写入'110' (PWM 模式 1) 或'111' (PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须设置 TMRx\_CCMx 寄存器 OCxPEN 位以使能相应的预装载寄存器，最后还要设置 TMRx\_CTRL1 寄存器的 ARPEN 位，(在向上计数或中心对称模式中) 使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TMRx\_EVEG 寄存器中的 UEVG 位来初始化所有的寄存器。OCx 的极性可以通过软件在 TMRx\_CCE 寄存器中的 CxP 位设置，它可以设置为高电平有效或低电平有效。TMRx\_CCE 寄存器中的 CxEN 位控制 OCx 输出使能。详见 TMRx\_CCE 寄存器的描述。

在 PWM 模式 (模式 1 或模式 2) 下，TMRx\_CNT 和 TMRx\_CCx 始终在进行比较，以确定是否符合  $TMRx\_CCx \leq TMRx\_CNT$  或者  $TMRx\_CNT \leq TMRx\_CCx$ 。

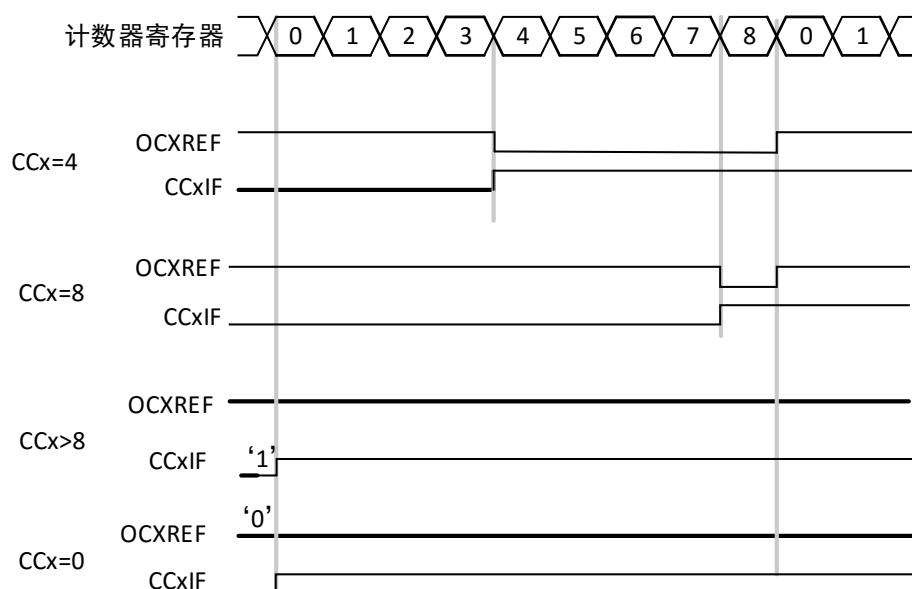
定时器在边沿对齐模式仅当计数器向上计数时能够产生 PWM。

### PWM 边沿对齐模式

下面是一个 PWM 模式 1 的例子。当  $TMRx\_CNT < TMRx\_CCx$  时 PWM 信号参考 OCxREF 为高，否则为低。如果  $TMRx\_CCx$  中的比较值大于自动重装载值 (TMRx\_AR)，则 OCxREF 保持为'1'。

如果比较值为 0，则 OCxREF 保持为'0'。下图为 TMRx\_AR=8 时边沿对齐的 PWM 波形实例。

图表 10-71 边沿对齐的 PWM 波形 (AR=8)



### 10.3.3.9 调试模式

当微控制器进入调试模式(Cortex™-M4 核心停止)，根据 DBG 模块中 DBG\_TMRx\_STOP 的设置，TMRx 计数器或者继续正常操作，或者停止。详见随后[第 18.2.2 节：支持定时器、看门狗和 I2C 的调试](#)。

### 10.3.4 TMR14寄存器描述

可以用半字 (16 位) 或字 (32 位) 的方式操作这些外设寄存器。

下表中将 TMRx 的所有寄存器映射到一个 16 位可寻址 (编址) 空间。

表格 10-6 TMRx – 寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

0x00	TMRx_CTRL1	保留	CLKDIV[1: 0] 0 0 0	保留	UEVRS 0 0	UEVDIS 0 0	CNTEN 0
	复位值						
0x0C	TMRx_DIE	保留	ARPEN 0	保留	C1IE 0	UEVIE 0	CNTEN 0
	复位值						
0x10	TMRx_STS	保留	C1OF 0	保留	C1IF 0	UEVIF 0	CNTEN 0
	复位值						
0x14	TMRx_EVEG	保留	C1G 0	保留	C1EVG 0	UEVG 0	CNTEN 0
	复位值						
0x18	TMRx_CCM1 输出 比较模式	保留	OC1MODE[2: 0] 0 0 0	保留	C1SE[1: 0] 0 0 0	C1DF[3: 0] 0 0 0 0	C1IPEN 0 0 0
	复位值						
0x18	TMRx_CCM1 输入 捕获模式	保留	IC1DF[3: 0] 0 0 0 0	保留	C1IP 0 0 0	C1IFEN 0 0 0	C1SEL[1: 0] 0 0 0
	复位值						
0x20	TMRx_CCE	保留	C1NP 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						
0x24	TMRx_CNT	保留	CNT[15: 0] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						
0x28	TMRx_DIV	保留	DIV[15: 0] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						
0x2C	TMRx_AR	保留	AR[15: 0] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						
0x34	TMRx_CC1	保留	CC1[15: 0] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						
0x50	TMRx_RMP	保留	T11_RMP [1: 0] 0 0	保留	C1P 0 0 0	C1EN 0 0 0	C1NP 0
	复位值						

#### 10.3.4.1 控制寄存器 1 (TMRx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		CLKDIV [1: 0]	ARP EN		保留		UEV RS	UEV DIS	CNT EN						
res		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 15: 10	保留, 始终读为 0。
位 9: 8	<b>CLKDIV[1: 0]:</b> 时钟分频因子 (Clock division) 定义在定时器时钟 (CK_INT) 频率与数字滤波器 (ETR, TIx) 使用的采样频率之间的分频比例。 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: 保留
位 7	<b>ARPEN:</b> 自动重装载预装载允许位 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲; 1: TMRx_AR 寄存器被装入缓冲器。
位 6: 3	保留, 始终读为 0。
位 2	<b>UEVRS:</b> 更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断, 则下述任一事件产生更新中断: - 计数器溢出 - 设置 UEVG 位 1: 如果使能了更新中断请求, 则只有计数器溢出才产生更新中断。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: - 计数器溢出 - 设置 UEVG 位 1: 禁止 UEV。不产生更新事件, 影子寄存器 (AR、DIV、CCx) 保持它们的值。如果设置了 UEVG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化。
位 0	<b>CNTEN:</b> 使能计数器 0: 禁止计数器; 1: 使能计数器。 在单脉冲模式下, 当发生更新事件时, CNTEN 被自动清除。

#### 10.3.4.2 DMA/中断使能寄存器 (TMRx\_DIE)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
保留														C1I E	UEV IE
位 15: 2	保留, 始终读为 0。														
位 1	<b>C1IE:</b> 允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断; 1: 允许捕获/比较 1 中断。														
位 0	<b>UEVIE:</b> 允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。														

### 10.3.4.3 状态寄存器 (TMRx\_STS)

偏移地址: 0x10

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		C1OF	保留		C1IF	UEV IF									
res	rw	rw	rw	rw	res	rw	res	rw							
位 15: 10	保留, 始终读为 0。														
位 9	<b>C1OF:</b> 捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置'1'。写'0'可清除该位。 0: 无重复捕获产生; 1: 当计数器的值被捕获到 TMRx_CC1 寄存器时, C1IF 的状态已经为'1'。														
位 8: 2	保留, 始终读为 0。														
位 1	<b>C1IF:</b> 捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) <b>如果通道 CC1 配置为输出模式:</b> 当计数器值与比较值匹配时该位由硬件置'1', 由软件清'0'。 0: 无匹配发生; 1: TMRx_CNT 的值与 TMRx_CC1 的值匹配。当计数溢出时, C1F 只'1'。 <b>如果通道 CC1 配置为输入模式:</b> 当捕获事件发生时该位由硬件置'1', 它由软件清'0'或通过读 TMRx_CC1 清'0'。 0: 无输入捕获产生; 1: 计数器值已被捕获(拷贝)至 TMRx_CC1 (在 IC1 上检测到与所选极性相同的边沿)。														
位 0	<b>UEVIF:</b> 更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': - 若 TMRx_CTRL1 寄存器的 UEVDIS=0、UEVRS=0, 当 TMRx_EVEG 寄存器的 UEVG=1 时产生更新事件 (软件对计数器 CNT 重新初始化); - 若 TMRx_CTRL1 寄存器的 UEVDIS=0, 计数器溢出。														

### 10.3.4.4 事件产生寄存器 (TMRx\_EVEG)

偏移地址: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														C1G	UEVG
res														rw	rw
位 15: 2	保留, 始终读为 0。														
位 1	<b>C1G:</b> 产生捕获/比较 1 事件 (Capture/compare 1 generation) 该位由软件置'1', 用于产生一个捕获/比较事件, 由硬件自动清'0'。 0: 无动作; 1: 在通道 CC1 上产生一个捕获/比较事件: <b>若通道 CC1 配置为输出:</b> 设置 C1IF=1, 若开启对应的中断, 则产生相应的中断。 <b>若通道 CC1 配置为输入:</b> 当前的计数器值捕获至 TMRx_CC1 寄存器; 设置 C1IF=1, 若开启对应的中断, 则产生相应的中断。若 C1IF 已经为 1, 则设置 C1OF=1。														

位 0	<b>UEVG:</b> 产生更新事件 (Update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。计数器被清'0'。
-----	--

### 10.3.4.5 捕获/比较模式寄存器1 (TMRx\_CCM1)

偏移地址: 0x18

复位值: 0x0000

通道可用于输入 (捕获模式) 或输出 (比较模式), 通道的方向由相应的 CxSEL 定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能, ICxx 描述了通道在输出模式下的功能。因此必须注意, 同一个位在输出模式和输入模式下的功能是不同的。

**输出比较模式:**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								OC1MODE[2: 0]	OC1PEN	OC1FEN	C1SEL[1: 0]				
res								rw	rw	rw	rw	rw	rw	rw	rw

位 15: 7	保留, 始终读为 0。
位 6: 4	<b>OC1MODE[2: 0]:</b> 输出比较 1 模式 (Output compare 1 enable) 该 3 位定义了输出参考信号 OC1REF 的动作, 而 OC1REF 决定了 OC1 的值。OC1REF 是高电平有效, 而 OC1 的有效电平取决于 C1P 位。 000: 冻结。输出比较寄存器 TMRx_CC1 与计数器 TMRx_CNT 间的比较对 OC1REF 不起作用; 001: 匹配时设置通道 1 为有效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时, 强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时, 强制 OC1REF 为低。 011: 翻转。当 TMRx_CC1=TMRx_CNT 时, 翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1—一旦 TMRx_CNT<TMRx_CC1 时通道 1 为有效电平, 否则为无效电平; 111: PWM 模式 2—一旦 TMRx_CNT<TMRx_CC1 时通道 1 为无效电平, 否则为有效电平; 注: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC1REF 电平才改变。
位 3	<b>OC1PEN:</b> 输出比较 1 预装载使能 (Output compare 1 preload enable) 0: 禁止 TMRx_CC1 寄存器的预装载功能, 可随时写入 TMRx_CC1 寄存器, 并且新写入的数值立即起作用。 1: 开启 TMRx_CC1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TMRx_CC1 的预装载值在更新事件到来时被传送至当前寄存器中。 注: 仅在单脉冲模式下 (TMRx_CTRL1 寄存器的 OPMODE='1'), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。
位 2	<b>OC1FEN:</b> 输出比较 1 快速使能 (Output compare 1 fast enable) 该位用于加快 CC 输出对触发器输入事件的响应。 0: 根据计数器与 CC1 的值, CC1 正常操作, 即使触发器是打开的。当触发器的输入出现一个有效沿时, 激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此, OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 该位只在通道被配置成 PWM1 或 PWM2 模式时起作用。

<b>位 1: 0</b>	<p><b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 selection)          这 2 位定义通道的方向 (输入/输出)，及输入脚的选择：            00: CC1 通道被配置为输出；            01: CC1 通道被配置为输入， IC1 映射在 TI1 上；            10: 保留；            11: 保留。  <i>注：C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=’0’) 才是可写的。</i> </p>
---------------	--

输入捕获模式:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
res				IC1DF[3: 0]				IC1DIV [1: 0]		C1SEL [1: 0]					

位 15: 8	保留, 始终读为 0。																																
位 7: 4	<p><b>IC1DF[3: 0]:</b> 输入捕获 1 滤波器 (Input capture 1 filter)            这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成, 它记录到 N 个事件后会产生一个输出的跳变:</p> <table> <tbody> <tr><td>0000:</td><td>无滤波器, 以 <math>f_{DTS}</math> 采样</td><td>1000:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/8</math>, N=6</td></tr> <tr><td>0001:</td><td>采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=2</td><td>1001:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/8</math>, N=8</td></tr> <tr><td>0010:</td><td>采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=4</td><td>1010:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=5</td></tr> <tr><td>0011:</td><td>采样频率 <math>f_{SAMPLING} = f_{CK\_INT}</math>, N=8</td><td>1011:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=6</td></tr> <tr><td>0100:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/2</math>, N=6</td><td>1100:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/16</math>, N=8</td></tr> <tr><td>0101:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/2</math>, N=8</td><td>1101:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=5</td></tr> <tr><td>0110:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/4</math>, N=6</td><td>1110:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=6</td></tr> <tr><td>0111:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/4</math>, N=8</td><td>1111:</td><td>采样频率 <math>f_{SAMPLING} = f_{DTS}/32</math>, N=8</td></tr> </tbody> </table>	0000:	无滤波器, 以 $f_{DTS}$ 采样	1000:	采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6	0001:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2	1001:	采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8	0010:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4	1010:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5	0011:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8	1011:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6	0100:	采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6	1100:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8	0101:	采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8	1101:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5	0110:	采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6	1110:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6	0111:	采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8	1111:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8
0000:	无滤波器, 以 $f_{DTS}$ 采样	1000:	采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6																														
0001:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2	1001:	采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8																														
0010:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4	1010:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5																														
0011:	采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8	1011:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6																														
0100:	采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6	1100:	采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8																														
0101:	采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8	1101:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5																														
0110:	采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6	1110:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6																														
0111:	采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8	1111:	采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8																														
位 3: 2	<p><b>IC1DIV[1: 0]:</b> 输入/捕获 1 预分频器 (Input capture 1 prescaler)            这 2 位定义了 CC1 输入 (IC1) 的预分频系数。            一旦 C1EN='0' (TMRx_CCE 寄存器中), 则预分频器复位。</p> <table> <tbody> <tr><td>00:</td><td>无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获;</td></tr> <tr><td>01:</td><td>每 2 个事件触发一次捕获;</td></tr> <tr><td>10:</td><td>每 4 个事件触发一次捕获;</td></tr> <tr><td>11:</td><td>每 8 个事件触发一次捕获。</td></tr> </tbody> </table>	00:	无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获;	01:	每 2 个事件触发一次捕获;	10:	每 4 个事件触发一次捕获;	11:	每 8 个事件触发一次捕获。																								
00:	无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获;																																
01:	每 2 个事件触发一次捕获;																																
10:	每 4 个事件触发一次捕获;																																
11:	每 8 个事件触发一次捕获。																																
位 1: 0	<p><b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 selection)            这 2 位定义通道的方向 (输入/输出), 及输入脚的选择:</p> <table> <tbody> <tr><td>00:</td><td>CC1 通道被配置为输出;</td></tr> <tr><td>01:</td><td>CC1 通道被配置为输入, IC1 映射在 TI1 上;</td></tr> <tr><td>10:</td><td>保留;</td></tr> <tr><td>11:</td><td>保留。</td></tr> </tbody> </table> <p>注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN='0') 才是可写的。</p>	00:	CC1 通道被配置为输出;	01:	CC1 通道被配置为输入, IC1 映射在 TI1 上;	10:	保留;	11:	保留。																								
00:	CC1 通道被配置为输出;																																
01:	CC1 通道被配置为输入, IC1 映射在 TI1 上;																																
10:	保留;																																
11:	保留。																																

#### 10.3.4.6 捕获/比较使能寄存器 (TMRx\_CCE)

偏移地址: 0x20

复位值: 0x0000

位 3	<b>C1NP:</b> 输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) CC1 通道配置为输出: C1NP 必须保持清除。 CC1 通道配置为输入: C1NP 结合 C1P 定义 TI1FP1 极性 (参考 C1P 描述)
位 2	保留, 始终读为 0。
位 1	<b>C1P:</b> 输入/捕获 1 输出极性 (Capture/Compare 1 output polarity) <b>CC1 通道配置为输出:</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1 通道配置为输入:</b> CC1NP/CC1P 位选择 TI1FP1 的极性信号作为触发或捕获信号。 00: 不反相/上升沿。电路对 TI1FP1 上升沿敏感 (捕获模式), TI1FP1 不反相。 01: 反相/下降沿。电路对 TI1FP1 下降沿敏感 (捕获模式), TI1FP1 反相。 10: 保留。 11: 不反相/双沿。电路同时对 TI1FP1 上升沿和下降沿敏感 (捕获模式), TI1FP1 是不反相。
位 0	<b>C1EN:</b> 输入/捕获 1 输出使能 (Capture/Compare 1 output enable) CC1 通道配置为输出: 0: 关闭 - OC1 禁止输出。 1: 开启 - OC1 信号输出到对应的输出引脚。 CC1 通道配置为输入: 该位决定了计数器的值是否能捕获入 TMRx_CC1 寄存器。 0: 捕获禁止。 1: 捕获使能。

表格 10-7 标准 OCx 通道的输出控制位

CxEN 位	OCx 输出状态
0	禁止输出 (OCx=0, OCx_EN=0)
1	OCx = OCxREF + 极性, OCx_EN=1

注意: 连接到标准 OCx 通道的外部 I/O 引脚状态, 取决于 OCx 通道状态和 GPIO 以及 AFIO 寄存器。

#### 10.3.4.7 计数器 (TMRx\_CNT)

偏移地址: 0x24

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
位 15: 0		<b>CNT[15: 0]:</b> 计数器的值 (Counter value)													

#### 10.3.4.8 预分频器 (TMRx\_DIV)

偏移地址: 0x28

复位值: 0x0000

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

位 15: 0	<b>DIV[15: 0]:</b> 预分频器的值 (Prescaler value) 计数器的时钟频率 CK_CNT 等于 $f_{CK\_DIV} / (\text{DIV}[15: 0] + 1)$ 。 DIV 包含了当更新事件产生时装入当前预分频器寄存器的值。
---------	--

#### 10.3.4.9 自动重装载寄存器 (TMRx\_AR)

偏移地址: 0x2C

复位值: 0x0000

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

位 15: 0	<b>AR[15: 0]:</b> 自动重装载的值 (Auto reload value) AR 包含了将要传送至实际的自动重装载寄存器的数值。 详细参考 <a href="#">10.3.3.1 节</a> : 有关 AR 的更新和动作当自动重装载的值为空时, 计数器不工作。
---------	---

#### 10.3.4.10 捕获/比较寄存器1 (TMRx\_CC1)

偏移地址: 0x34

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC1[15: 0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 15: 0	<b>CC1[15: 0]:</b> 捕获/比较 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出: CC1 包含了装入当前捕获/比较 1 寄存器的值 (预装载值)。 如果在 TMRx_CCM1 寄存器 (OC1PEN 位) 中未选择预装载特性, 写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较寄存器 1 中。 当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入: CC1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。
---------	---

#### 10.3.4.11 通道输入重映射寄存器 (TMRx\_RMP)

偏移地址: 0x50

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留													TI1_RMP		
res													rw		

位 15: 2	保留，始终读为 0。
位 1: 0	<b>TI1_RMP[1: 0]:TMR14 通道 1 输入重映射</b> 00: TMR14 通道 1 连接到 GPIO 01: TMR14 通道 1 连接到 RTCCLK 10: TMR14 通道 1 连接到 HSE/32 CLOCK 11: TMR14 通道 1 连接到 MCO

## 10.4 通用定时器 (TRM15)

### 10.4.1 TMR15简介

通用定时器 (TMR15) 由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM 等）。

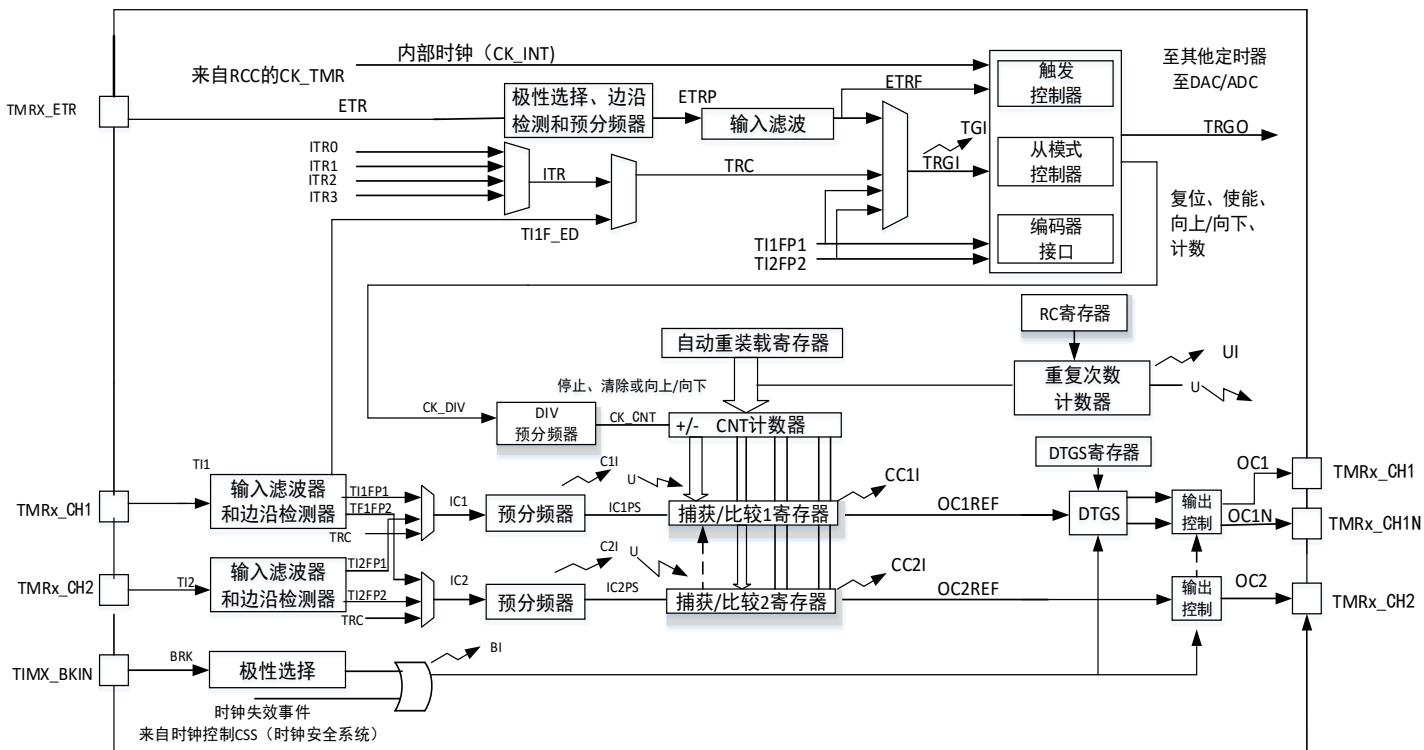
使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

### 10.4.2 TMR15主要特性

TMR15 定时器的功能包括：

- 16位自动装载计数器
- 16位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为1~65535之间的任意数值
- 多达2个独立通道：
  - 输入捕获
  - 输出比较
  - PWM生成（边缘对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入

图表 10-72 通用控制定时器15框图



注意： 根据控制位的设定，在 U（更新）事件时传送预加载寄存器的内容至工作寄存器

事件

中断和 DMA 输出

## 10.4.3 TMR15功能描述

### 10.4.3.1 时基单元

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

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器（**TMRx\_CNT**）
- 预分频器寄存器（**TMRx\_DIV**）
- 自动装载寄存器（**TMRx\_AR**）
- 重复次数寄存器（**TMRx\_RC**）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 **TMRx\_CTRL1** 寄存器中的自动装载预装载使能位(**ARPEN**)的设置，预装载寄存器的内容被立即或在每次的更新事件 **UEV** 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 **TMRx\_CTRL1** 寄存器中的 **UEVDIS** 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 **CK\_CNT** 驱动，仅当设置了计数器 **TMRx\_CTRL1** 寄存器中的计数器使能位 (**CNTEN**) 时，**CK\_CNT** 才有效。（更多有关使能计数器的细节，请参见控制器的从模式描述）。

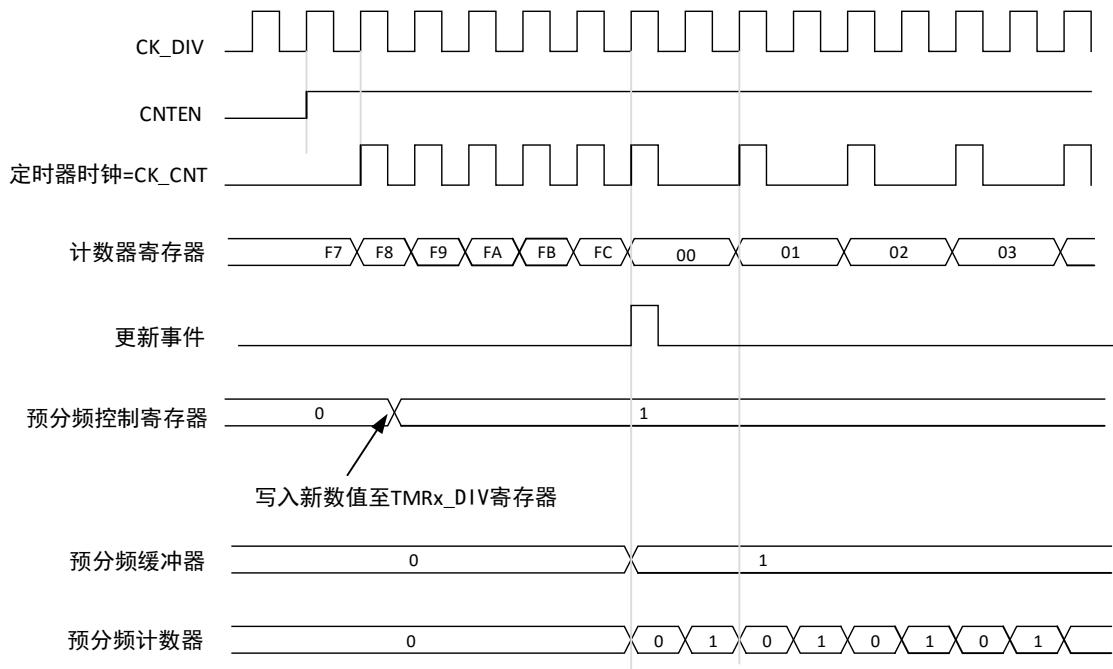
注意，在设置了 **TMRx\_CTRL** 寄存器的 **CNTEN** 位的一个时钟周期后，计数器开始计数。

#### 预分频器描述

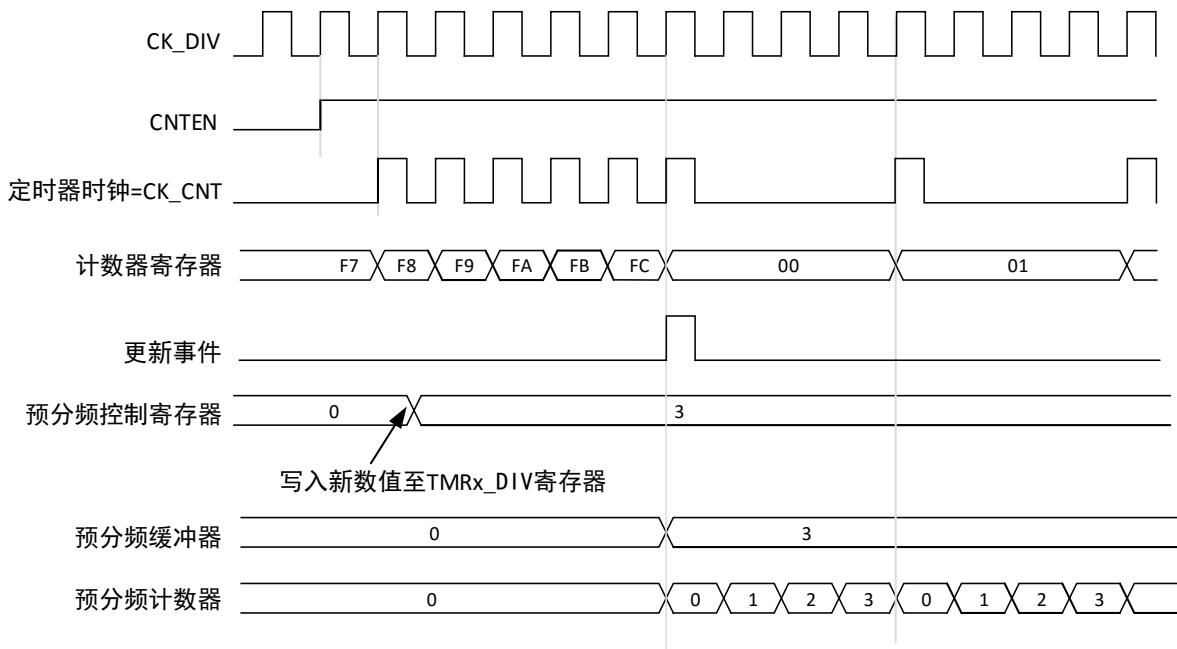
预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TMRx\_DIV 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

[图 10-73](#) 和 [图 10-74](#) 给出了在预分频器运行时，更改计数器参数的例子。

图表 10-73 当预分频器的参数从1变到2时，计数器的时序图



图表 10-74 当预分频器的参数从1变到4时，计数器的时序图



### 10.4.3.2 计数器模式

#### 向上计数模式

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

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

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

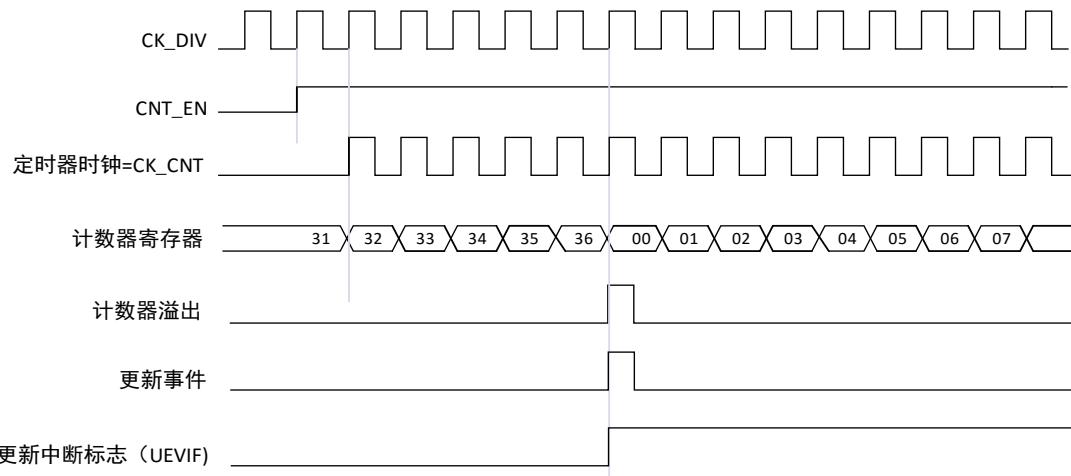
设置 **TMRx\_CTRL1** 寄存器中的 **UEVDIS** 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 **UEVDIS** 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清 0（但预分频器的数值不变）。此外，如果设置了 **TMRx\_CTRL1** 寄存器中的 **UVERS** 位（选择更新请求），设置 **UEVG** 位将产生一个更新事件 **UEV**，但硬件不设置 **UEVIF** 标志（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 **UVERS** 位）设置更新标志位（**TMRx\_STS** 寄存器中的 **UEVIF** 位）。

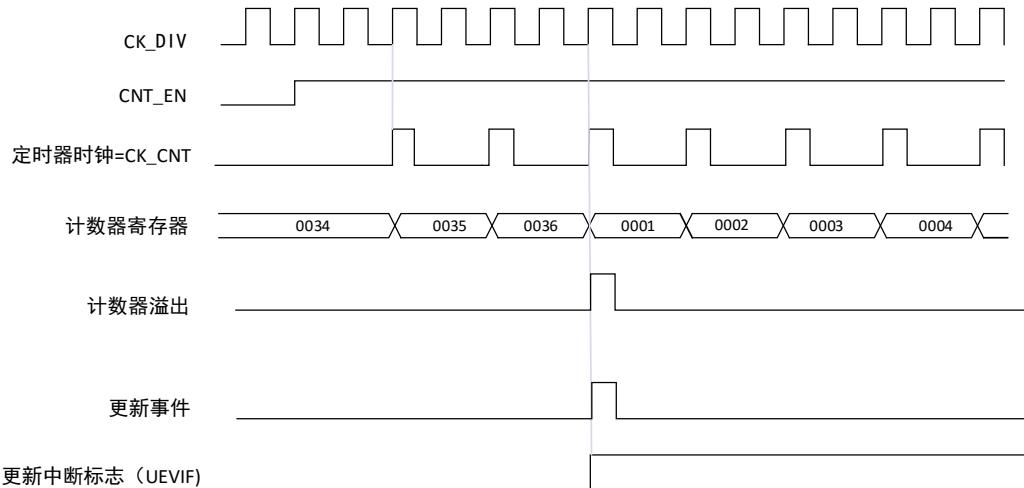
- 重复计数器被重新加载为 **TMRx\_RC** 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值（**TMRx\_AR**）。
- 预分频器的缓冲区被置入预装载寄存器的值（**TMRx\_DIV** 寄存器的内容）。

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

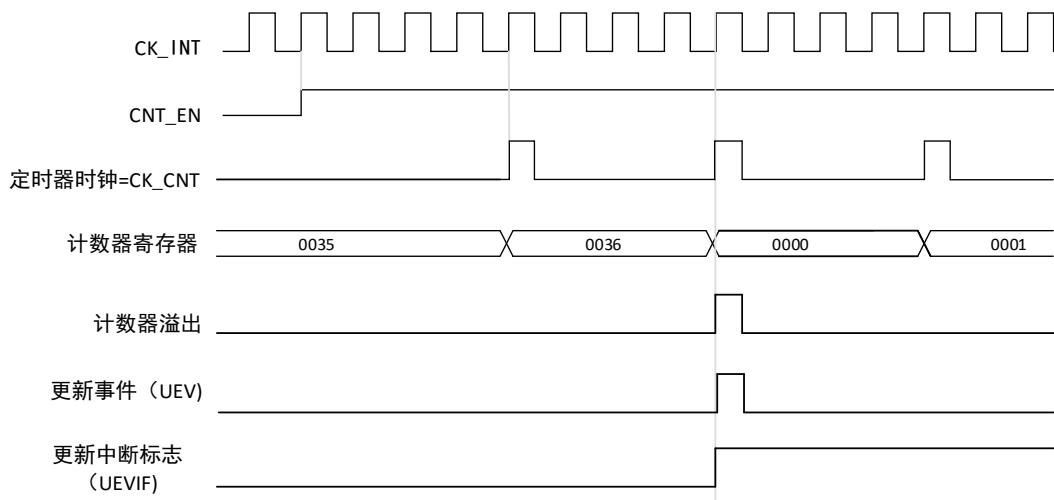
图表 10-75 计数器时序图，内部时钟分频因子为1



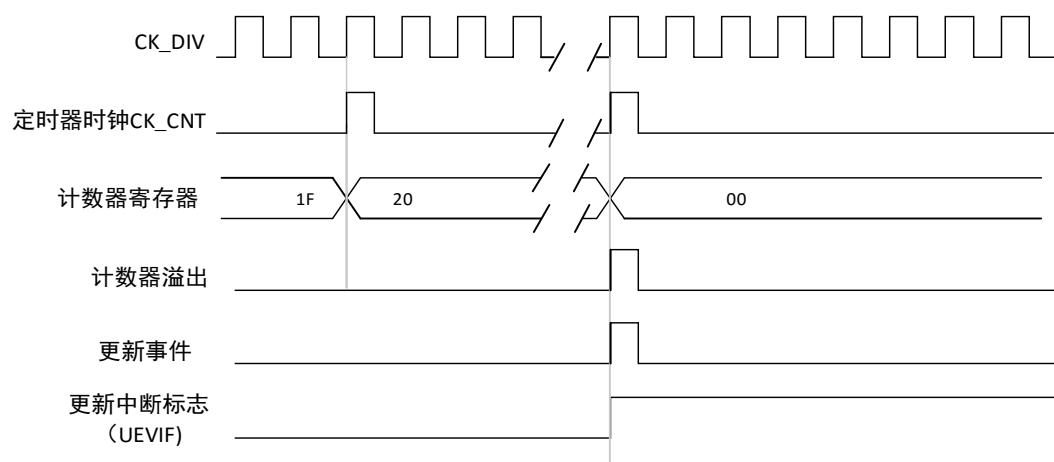
图表 10-76 计数器时序图，内部时钟分频因子为2



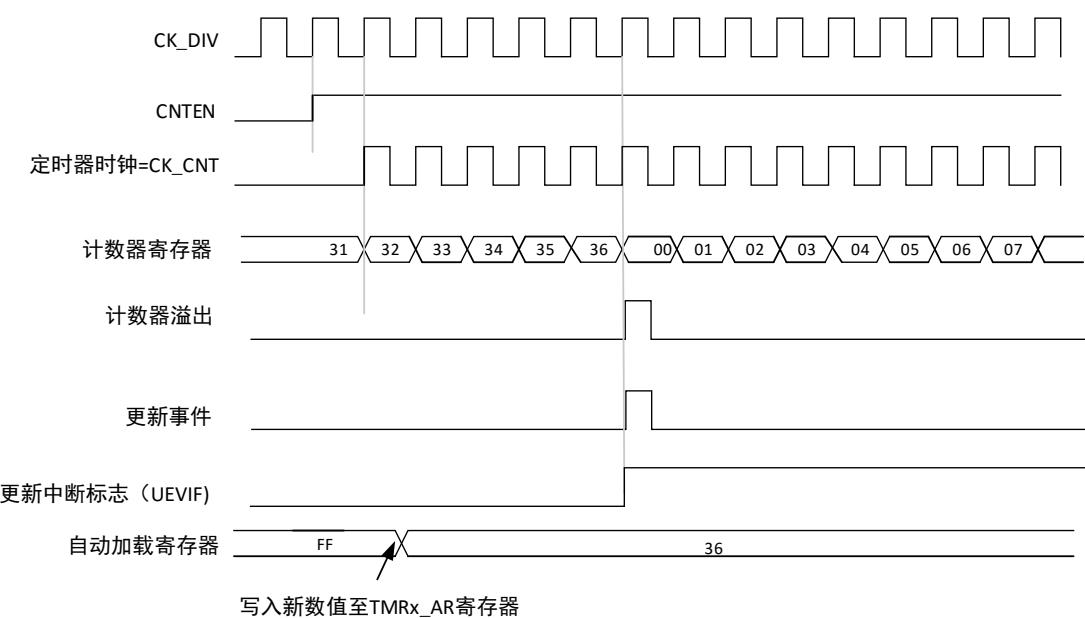
图表 10-77 计数器时序图，内部时钟分频因子为4



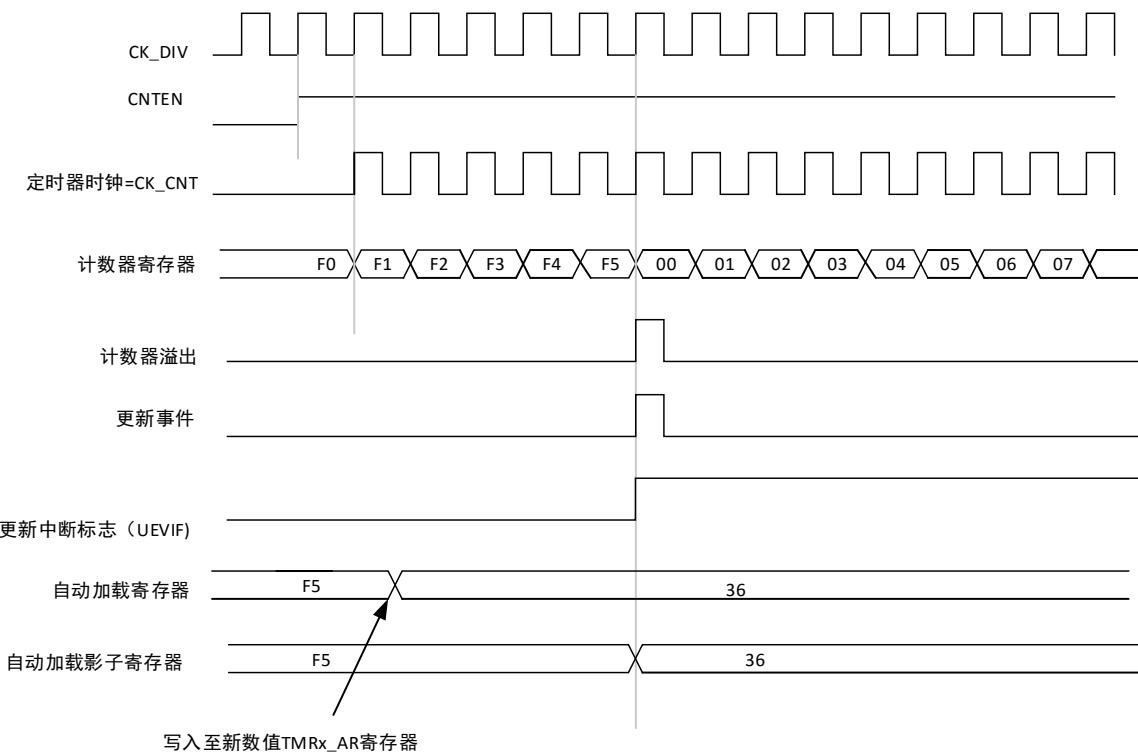
图表 10-78 计数器时序图，内部时钟分频因子为N



图表 10-79 计数器时序图，当ARPEN=0时的更新事件 (TMRx\_AR没有预装入)



图表 10-80 计数器时序图，当ARPEN=1时的更新事件（预装入了TMRx\_AR）



### 10.4.3.3 重复计数器

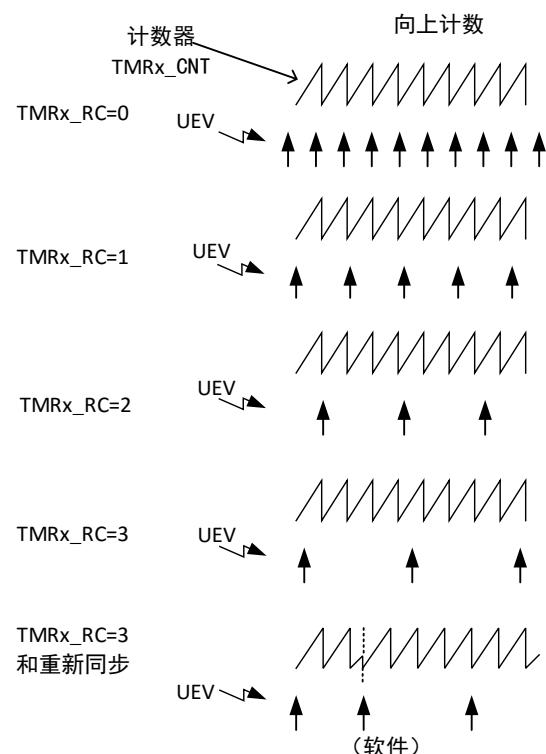
10.4.3.1 “时基单元”解释了计数器上溢时更新事件是如何产生的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢时，数据从预装载寄存器传输到影子寄存器 (TMRx\_AR 自动重载入寄存器，TMRx\_DIV 预装载寄存器，还有在比较模式下的捕获/比较寄存 (TMRx\_CCx)，N 是 TMRx\_RC 重复计数寄存器中的值)。

- 重复计数器在向上计数模式下每次计数器溢出时递减

重复计数器是自动加载的，重复速率是由 TMRx\_RC 寄存器的值定义 (参看图 10-81)。当更新事件由软件产生 (通过设置 TMRx\_EVEG 中的 UEVG 位) 或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TMRx\_RC 寄存器中的内容被重载入到重复计数器。

图表 10-81 不同模式下更新速率的例子，及TMRx\_RC的寄存器设置



UEV  $\searrow$  更新事件：传送预装载寄存器至实际寄存器并产生更新中断

#### 10.4.3.4 时钟选择

计数器时钟可由下列时钟源提供：

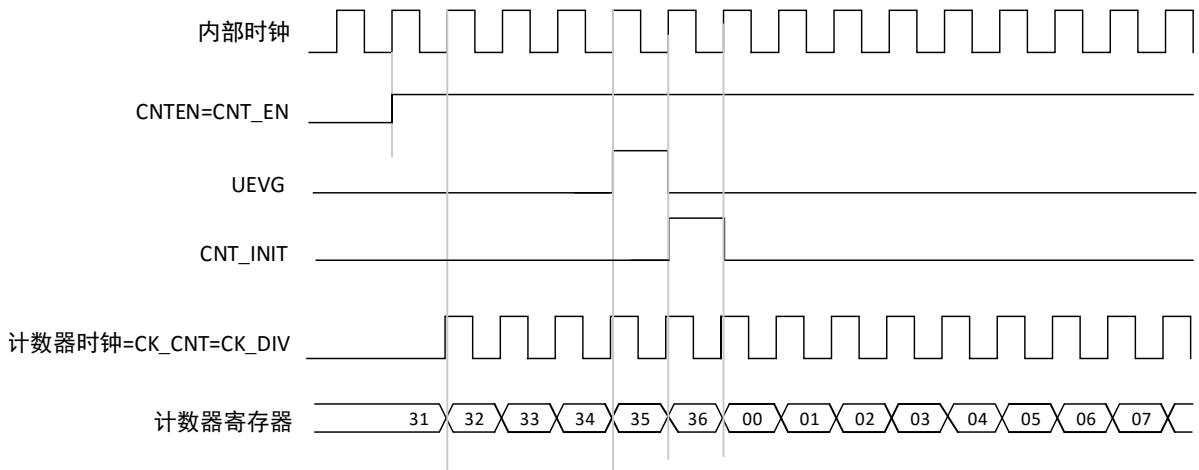
- 内部时钟（CK\_INT）
- 外部时钟模式1：外部输入引脚
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器Timer1而作为另一个定时器Timer15的预分频器。详见 [10.4.4.10节](#)。

##### 内部时钟源（CK\_INT）

如果禁止了从模式控制器（SMSEL=000），则 CNTEN、DIR（TMRx\_CTRL1 寄存器）和 UEVG 位（TMRx\_EVEG 寄存器）是实际的控制位，并且只能被软件修改（UEVG 位仍被自动清除）。只要 CNTEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK\_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

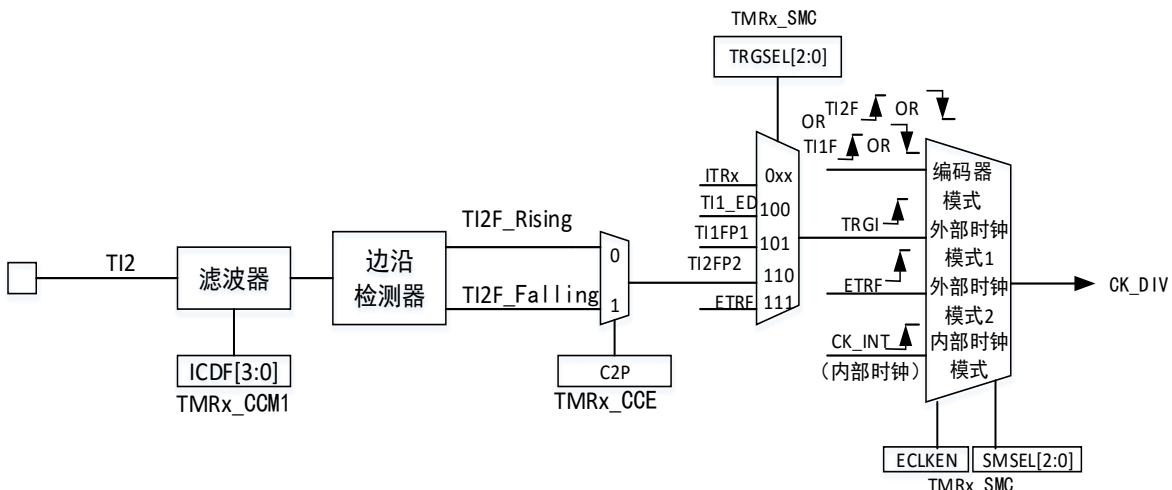
图表 10-82 一般模式下的控制电路，内部时钟分频因子为1



### 外部时钟源模式 1

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

图表 10-83 TI2 外部时钟连接例子



例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

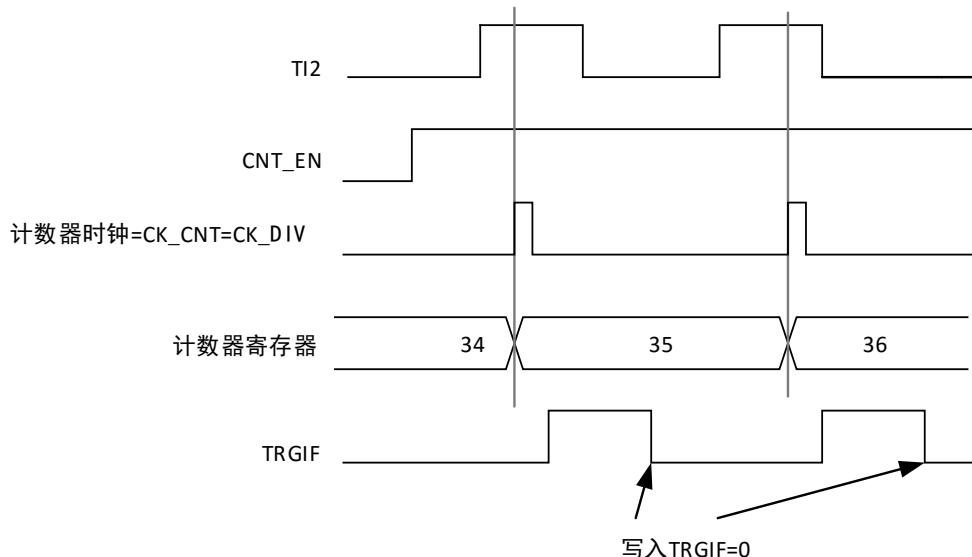
1. 配置 TMRx\_CCM1 寄存器 C2SEL=01，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TMRx\_CCM1 寄存器的 IC2DF[3:0]，选择输入滤波器带宽(如果不需要滤波器，保持 IC2DF = 0000)。
3. 配置 TMRx\_CCE 寄存器的 C2P=0，选定上升沿极性。
4. 配置 TMRx\_SMC 寄存器的 SMSEL=111，选择定时器外部时钟模式 1。
5. 配置 TMRx\_SMC 寄存器中的 TRGSEL=110，选定 TI2 作为触发输入源。
6. 设置 TMRx\_CTRL1 寄存器的 CNTEN=1，启动计数器。

**注意：**捕获预分频器不用作触发，所以不需要对它进行配置。

当上升沿出现在 TI2，计数器计数一次，且 TRGIF 标志被设置。

在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。

图表 10-84 外部时钟模式 1 下的控制电路



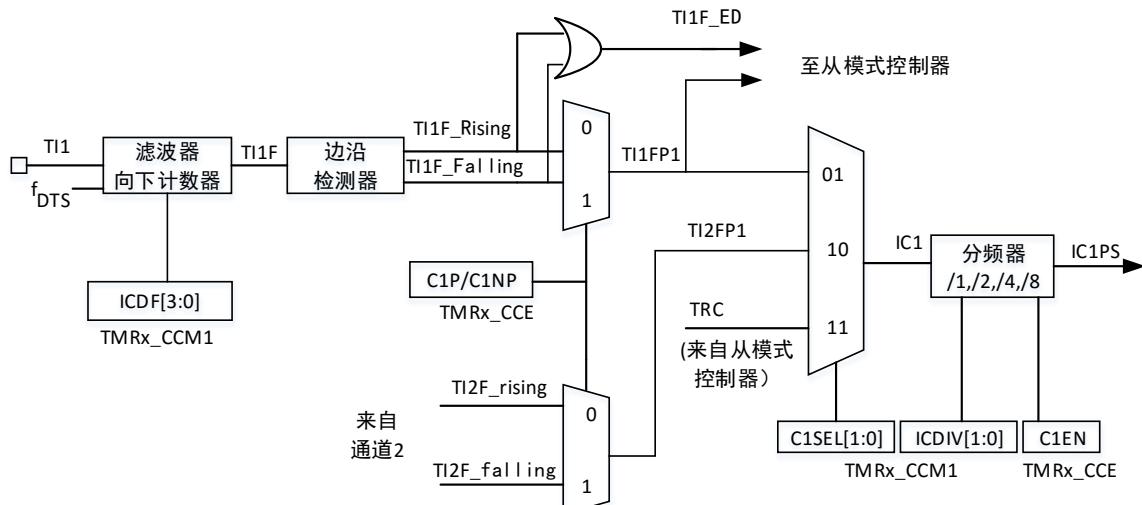
#### 10.4.3.5 捕获/比较通道

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

[图 10-85 至图 10-88](#) 是一个捕获/比较通道概览。

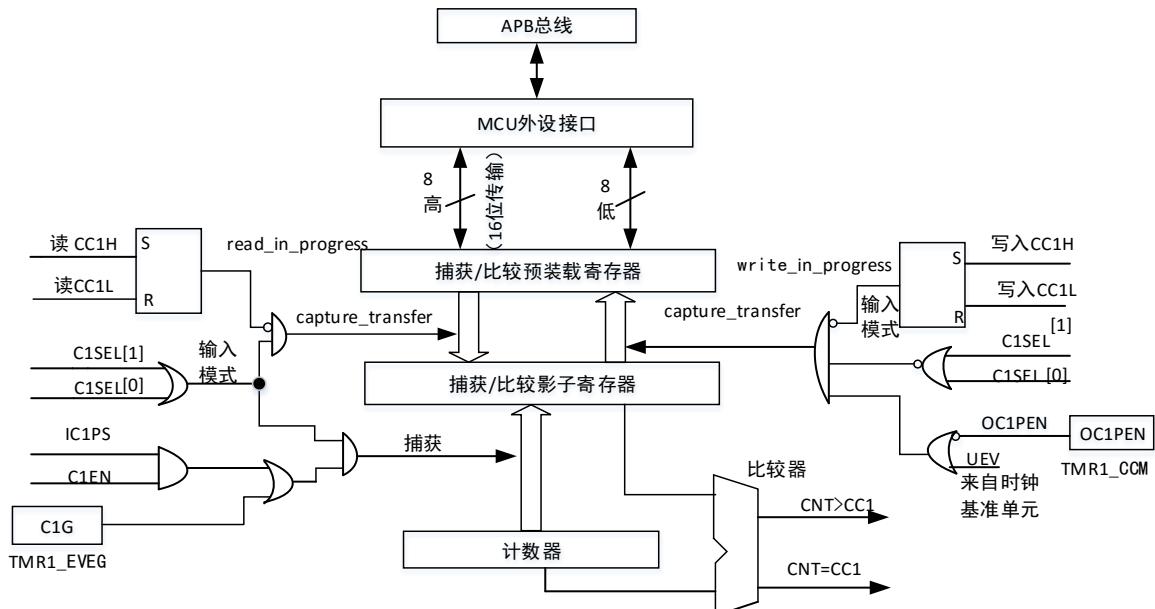
输入部分对相应的  $\text{TIx}$  输入信号采样，并产生一个滤波后的信号  $\text{TIxF}$ 。然后，一个带极性选择的边缘检测器产生一个信号 ( $\text{TIxFpX}$ )，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器 ( $\text{ICxPS}$ )。

图表 10-85 捕获/比较通道（如：通道1输入部分）

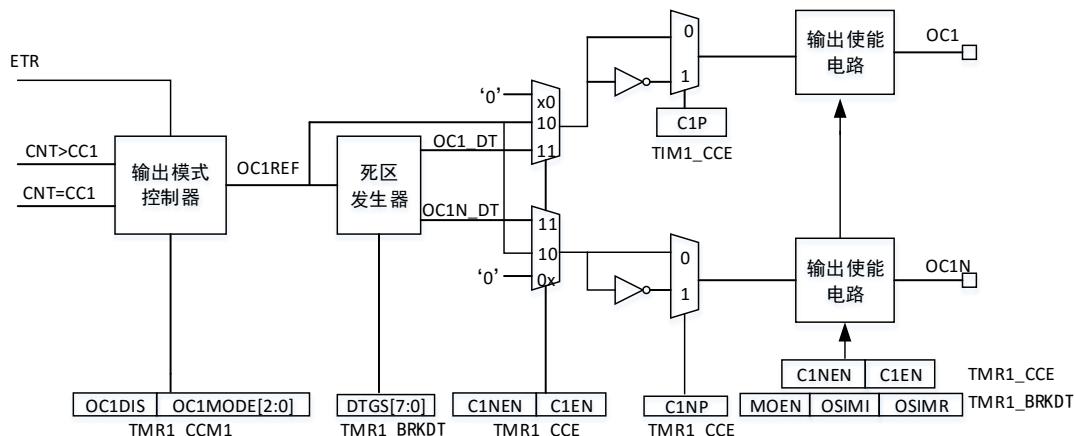


输出部分产生一个中间波形  $\text{OCxRef}$ （高有效）作为基准，链的末端决定最终输出信号的极性。

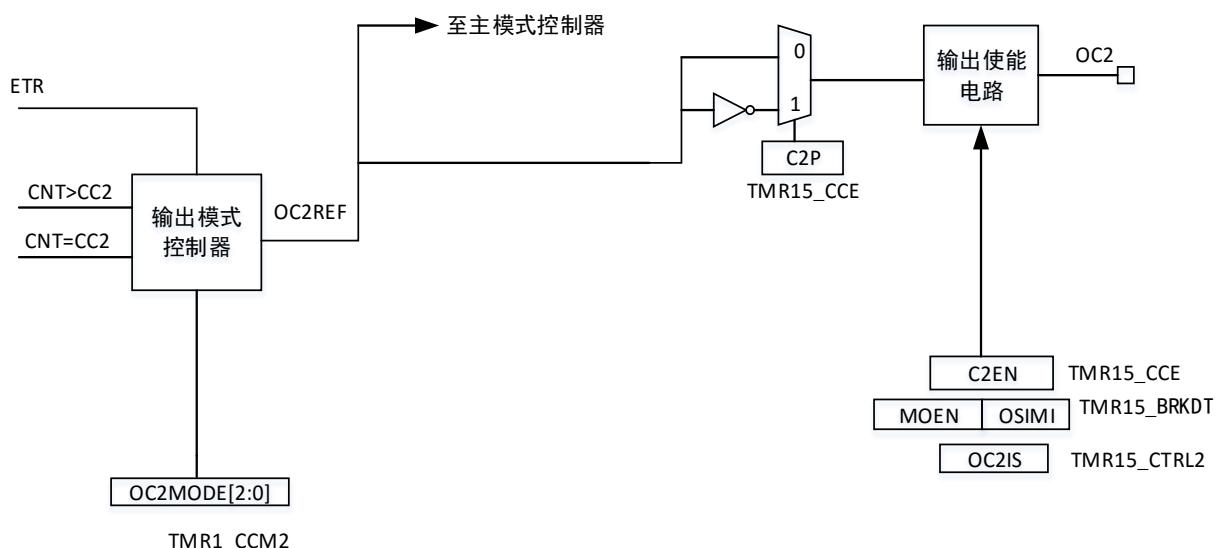
图表 10-86 捕获/比较通道1的主电路



图表 10-87 捕获/比较通道的输出部分（通道1）



图表 10-88 捕获/比较通道的输出部分（通道2）



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

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

#### 10.4.3.6 输入捕获模式

在输入捕获模式下，当检测到 IC<sub>x</sub> 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 (TMR<sub>x</sub>\_CC<sub>x</sub>) 中。当发生捕获事件时，相应的 CxIF 标志 (TMR<sub>x</sub>\_STS 寄存器) 被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CxIF 标志已经为高，那么重复捕获标志 CxOF (TMR<sub>x</sub>\_STS 寄存器) 被置 1。写 CxIF=0 可清除 CxIF，或读取存储在 TMR<sub>x</sub>\_CC<sub>x</sub> 寄存器中的捕获数据也可清除 CxIF。写 CxOF=0 可清除 CxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TMR<sub>x</sub>\_CC1 寄存器中，步骤如下：

- 选择有效输入端：TMR<sub>x</sub>\_CC1 必须连接到 TI1 输入，所以写入 TMR<sub>x</sub>\_CC1 寄存器中的 C1SEL=01，只要 C1SEL 不为 '00'，通道被配置为输入，并且 TMR<sub>x</sub>\_CC1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TI<sub>x</sub> 时，输入滤波器控制位是 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 IC<sub>x</sub>DF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以  $f_{DTS}$  频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TMR<sub>x</sub>\_CCM1 寄存器中写入 IC1DF=0011。
- 选择 TI1 通道的有效转换边沿，在 TMR<sub>x</sub>\_CCE 寄存器中写入 C1P=0（上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TMR<sub>x</sub>\_CCM1 寄存器的 IC1PS=00）。
- 设置 TMR<sub>x</sub>\_CCE 寄存器的 C1EN=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1IE 位允许相关中断请求，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TMR<sub>x</sub>\_CC1 寄存器。
- C1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 C1IF 未曾被清除，C1OF 也被置 1。
- 如设置了 C1IE 位，则会产生一个中断。
- 如设置了 C1DE 位，则还会产生一个 DMA 请求。

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

注意：设置 TMR<sub>x</sub>\_EVEG 寄存器中相应的 CxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

#### 10.4.3.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

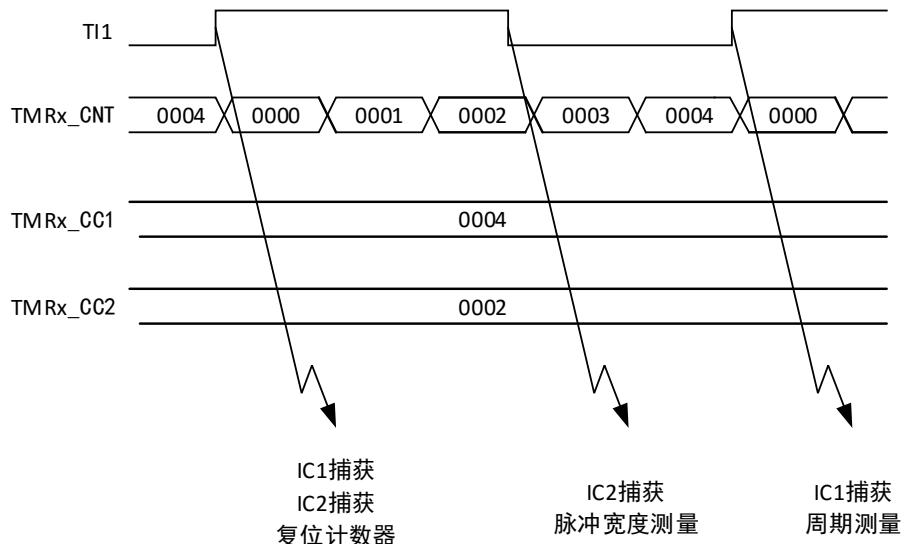
- 两个 IC<sub>x</sub> 信号被映射至同一个 TI<sub>x</sub> 输入。
- 这 2 个 IC<sub>x</sub> 信号为边沿有效，但是极性相反。
- 其中一个 TI<sub>x</sub>FP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的长度 (TMR<sub>x</sub>\_CC1 寄存器) 和占空比 (TMR<sub>x</sub>\_CC2 寄存器)，具体步骤如下（取决于 CK\_INT 的频率和预分频器的值）

- 选择 TMR<sub>x</sub>\_CC1 的有效输入：置 TMR<sub>x</sub>\_CCM1 寄存器的 C1SEL=01（选中 TI1）。
- 选择 TI1FP1 的有效极性（用来捕获数据到 TMR<sub>x</sub>\_CC1 中和清除计数器）：置 C1P=0（上升沿有效）。
- 选择 TMR<sub>x</sub>\_CC2 的有效输入：置 TMR<sub>x</sub>\_CCM1 寄存器的 C2SEL=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TMR<sub>x</sub>\_CC2）：置 C2P=1（下降沿有效）。

- 选择有效的触发输入信号：置 TMRx\_SMC 寄存器中的 TRGSEL=101（选择 TI1FP1）。
- 配置从模式控制器为复位模式：置 TMRx\_SMC 中的 SMSEL=100。
- 使能捕获：置 TMRx\_CCE 寄存器中 C1EN=1 且 C2EN=1。

图表 10-89 PWM 输入模式时序



注意：因为只有 *TI1FP1* 和 *TI2FP2* 连到了从模式控制器，所以 *PWM* 输入模式只能使用 *TMRx\_CH1/TMRx\_CH2* 信号。

#### 10.4.3.8 强制输出模式

在输出模式 (*TMRx\_CCMx* 寄存器中 *CxSEL=00*) 下，输出比较信号 (*OCxREF* 和相应的 *OCx/OCxN*) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 *TMRx\_CCMx* 寄存器中相应的 *OCxMODE=101*，即可强置输出比较信号 (*OCxREF/OCx*) 为有效状态。这样 *OCxREF* 被强置为高电平 (*OCxREF* 始终为高电平有效)，同时 *OCx* 得到 *CxP* 极性相反的信号。

例如：*CxP=0* (*OCx* 高电平有效)，则 *OCx* 被强置为高电平。

置 *TMRx\_CCMx* 寄存器中的 *OCxMODE=100*，可强置 *OCxREF* 信号为低。

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

#### 10.4.3.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式 (*TMRx\_CCMx* 寄存器中的 *OCxMODE* 位) 和输出极性 (*TMRx\_CCE* 寄存器中的 *CxP* 位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 (*OCxMODE=000*)、被设置成有效电平 (*OCxMODE=001*)、被设置成无效电平 (*OCxMODE=010*) 或进行翻转 (*OCxMODE=011*)。
- 设置中断状态寄存器中的标志位 (*TMRx\_STS* 寄存器中的 *CxIF* 位)。
- 若设置了相应的中断屏蔽 (*TMRx\_DIE* 寄存器中的 *CxIE* 位)，则产生一个中断。
- 若设置了相应的使能位 (*TMRx\_DIE* 寄存器中的 *CxDE* 位，*TMRx\_CTRL2* 寄存器中的 *CDSEL* 位选择 DMA 请求功能)，则产生一个 DMA 请求。

*TMRx\_CCMx* 中的 *OCxPEN* 位选择 *TMRx\_CCx* 寄存器是否需要使用预装载寄存器。

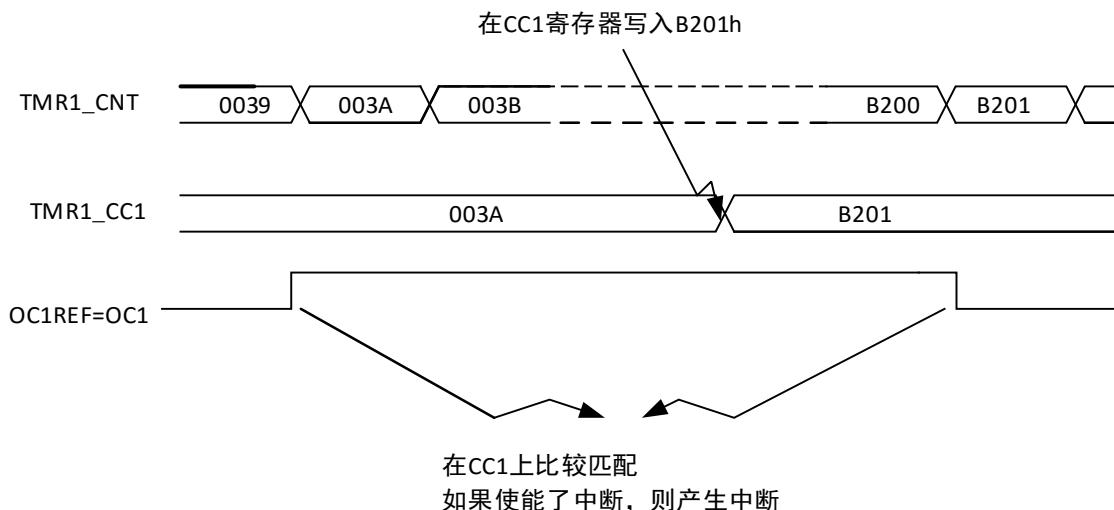
在输出比较模式下，更新事件 UEV 对 *OCxREF* 和 *OCx* 输出没有影响。

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

1. 选择计数器时钟(内部, 外部, 预分频器)。
2. 将相应的数据写入 **TMRx\_AR** 和 **TMRx\_CCx** 寄存器中。
3. 如果要产生一个中断请求, 设置 **CCxE** 位。
4. 选择输出模式, 例如:
  - 要求计数器与 **CCx** 匹配时翻转 **OCx** 的输出引脚, 设置 **OCxMODE=011**
  - 置 **OCxPEN = 0** 禁用预装载寄存器
  - 置 **CxP = 0** 选择极性为高电平有效
  - 置 **CxEN = 1** 使能输出
5. 设置 **TMRx\_CTRL1** 寄存器的 **CNTEN** 位启动计数器

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

图表 10-90 输出比较模式, 翻转OC1



#### 10.4.3.10 PWM模式

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

在 **TMRx\_CCMx** 寄存器中的 **OCxMODE** 位写入'110'(PWM模式1)或'111'(PWM模式2), 能够独立地设置每个 **OCx** 输出通道产生一路PWM。必须通过设置 **TMRx\_CCMx** 寄存器的 **OCxPEN** 位使能相应的预装载寄存器, 最后还要设置 **TMRx\_CTRL1** 寄存器的 **ARPEN** 位, (在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

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

**OCx** 的极性可以通过软件在 **TMRx\_CCE** 寄存器中的 **CxP** 位设置, 它可以设置为高电平有效或低电平有效。**OCx** 的输出使能通过(**TMRx\_CCE** 和 **TMRx\_BRKDT** 寄存器中) **CxEN**、**CxNEN**、**MOEN**、**OSIMI** 和 **OSIMR** 位的组合控制。详见 **TMRx\_CCE** 寄存器的描述。

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

根据 **TMRx\_CTRL1** 寄存器中 **CMSEL** 位的状态, 定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

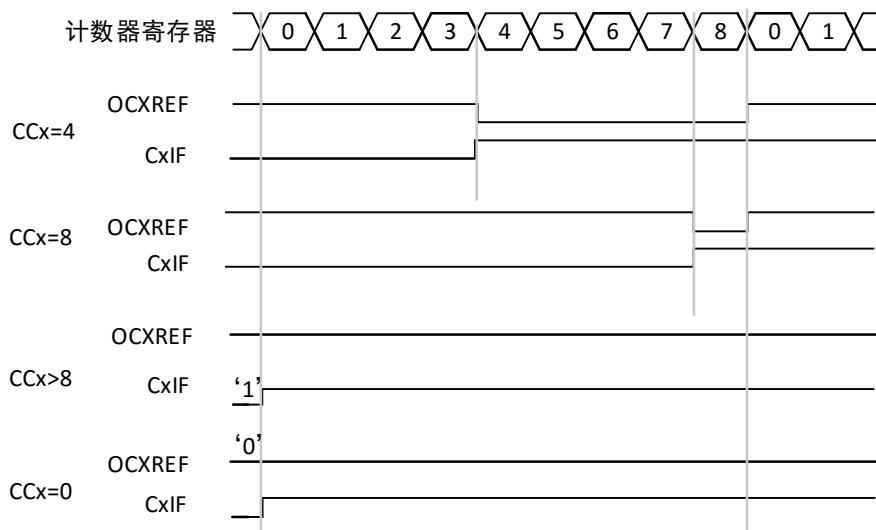
### PWM 边沿对齐模式

- 向上计数配置

当 TMRx\_CTRL1 寄存器中的 DIR 位为低的时候执行向上计数。参看 [10.4.3.2 节](#)。

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

图表 10-91 边沿对齐的 PWM 波形 (AR=8)



### 10.4.3.11 互补输出和死区插入

通用定时器 (TMR15) 能够输出一路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

配置 TMRx\_CCE 寄存器中的 CxP 和 CxNP 位，可以为每一个输出独立地选择极性（主输出 OCx 或互补输出 OCxN）。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TMRx\_CCE 寄存器的 CxEN 和 CxNEN 位，TMRx\_BRKDT 和 TMRx\_CTRL2 寄存器中的 MOEN、OCxIS、OCxNIS、OSIMI 和 OSIMR 位，详见 [表 10-13 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位](#)。特别的是，在转换到 IDLE 状态时 (MOEN 下降到 0) 死区被激活。

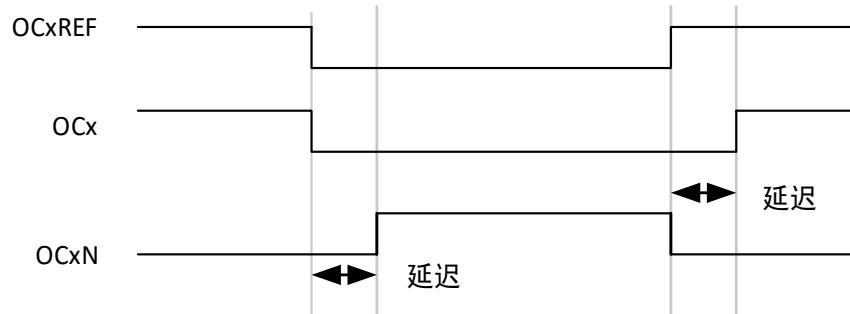
同时设置 CxEN 和 CxNEN 位将插入死区，如果存在刹车电路，则还要设置 MOEN 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

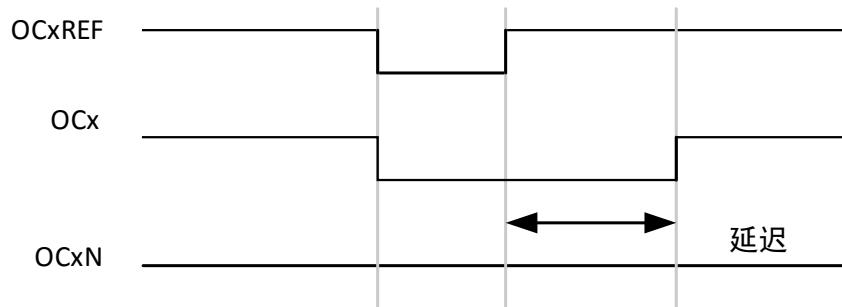
如果延迟大于当前有效的输出宽度 (OCx 或者 OCxN)，则不会产生相应的脉冲。

下列几张图显示了死区发生器的出信号和当前参考信号 OCxREF 之间的关系。(假设 CxP=0、CxNP=0、MOEN=1、CxEN=1 并且 CxNEN=1)

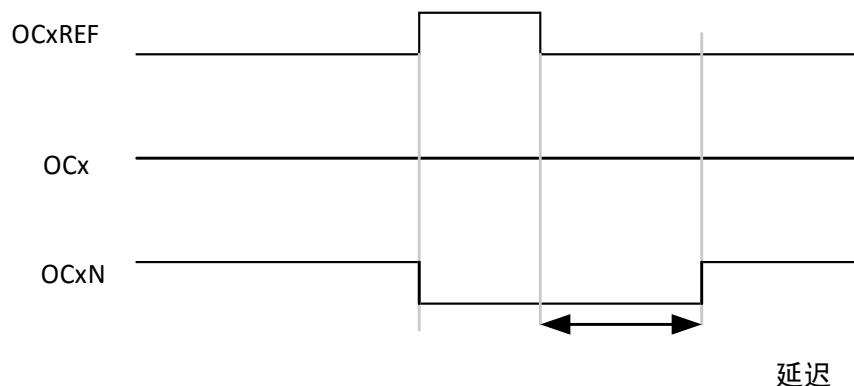
图表 10-92 带死区插入的互补输出



图表 10-93 死区波形延迟大于负脉冲



图表 10-94 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TMRx\_BRKDT 寄存器中的 DTGS 位编程配置。详见 [10.4.4.15 TMR15 刹车和死区寄存器 \(TMRx\\_BRKDT\)](#) 中的延时计算。

#### 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置、输出比较或 PWM)，通过配置 TMRx\_CCE 寄存器的 CxEN 和 CxNEN 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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

#### 10.4.3.12 使用刹车功能

当使用刹车功能时，依据相应的控制位 (TMRx\_BRKDT 寄存器中的 MOEN、OSIMI 和 OSIMR 位，TMRx\_CTRL2 寄存器中的 OCxIS 和 OCxNIS 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见 [表 10-13 带刹车功能的互补输出通道 OCx](#)

和 OCxN 的控制位。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生，详见 3.2.7 节时钟失效检测（CFD）。

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

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

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOEN 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSIMI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOEN=0，每一个输出通道输出由 TMRx\_CTRL2 寄存器中的 OCxIS 位设定的电平。如果 OSIMI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OCxIS 和 OCxNIS 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。
- 注意，因为重新同步 MOEN，死区时间比通常情况下长一些（大约 2 个 ck\_tim 的时钟周期）。
- 如果 OSIMI=0，定时器释放使能输出，否则保持使能输出；或一旦 CxEN 与 CxNEN 之一变高时，使能输出变为高。
- 如果设置了 TMRx\_DIE 寄存器中的 BRKIE 位，当刹车状态标志（TMRx\_STS 寄存器中的 BRKIF 位）为'1'时，则产生一个中断。如果设置了 TMRx\_DIE 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 如果设置了 TMRx\_BRKDT 寄存器中的 AOEN 位，在下一个更新事件 UEV 时 MOEN 位被自动置位；例如，这可以用来进行整形。否则，MOEN 始终保持低直到被再次置'1'；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注意：刹车输入为电平有效。所以，当刹车输入有效时，不能同时（自动地或者通过软件）设置 MOEN。同时，状态标志 BRKIF 不能被清除。

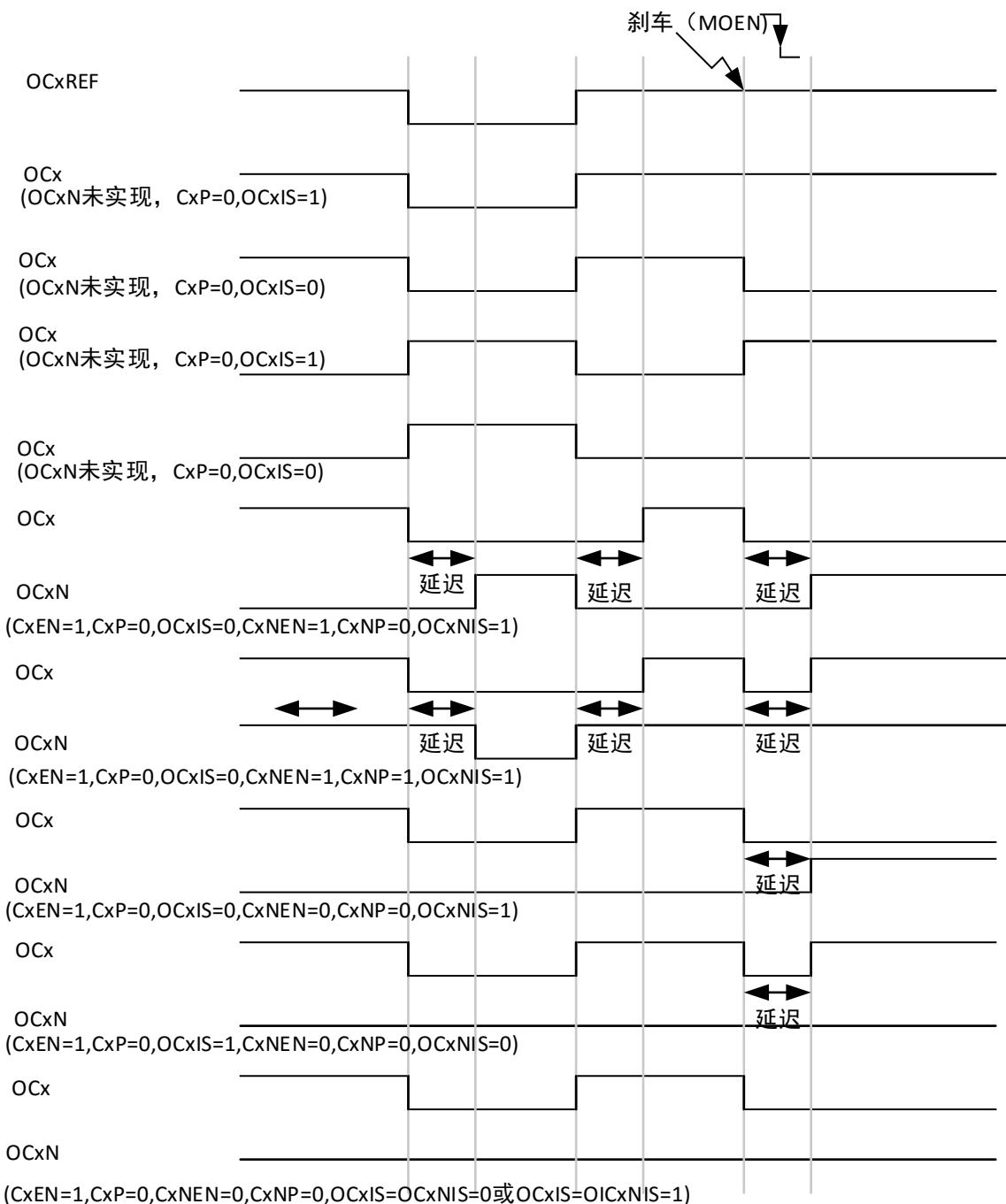
刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TMRx\_BRKDT 寄存器中的 BRKEN 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxMODE 配置，刹车使能和极性）。

用户可以通过 TMRx\_BRKDT 寄存器中的 LOCKC 位，从三级保护中选择一种，参看 [10.6.4.18 节 TMR1 刹车和死区寄存器（TMRx\\_BRKDT）](#)。在 MCU 复位后 LOCKC 位只能被修改一次。

下图显示响应刹车的输出实例。

图表 10-95    响应刹车的输出



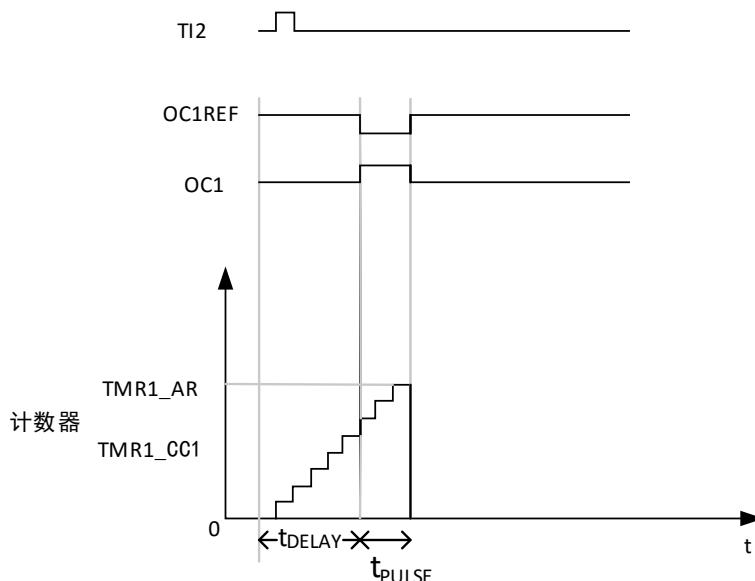
#### 10.4.3.13 单脉冲模式

单脉冲模式 (OPMODE) 是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TMRx\_CTRL1 寄存器中的 OPMODE 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

- 向上计数方式：计数器  $CNT < CCx \leq AR$  （特别地， $0 < CCx$ ）；
- 向下计数方式：计数器  $CNT > CCx$ 。

图表 10-96 单脉冲模式的例子



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

假定 TI2FP2 作为触发 1：

- 置 TMRx\_CCM1 寄存器中的 C2SEL=01，把 TI2FP2 映像到 TI2。
- 置 TMRx\_CCE 寄存器中的 C2P=0，使 TI2FP2 能够检测上升沿。
- 置 TMRx\_SMC 寄存器中的 TRGSEL=110，TI2FP2 作为从模式控制器的触发 (TRGI)。
- 置 TMRx\_SMC 寄存器中的 SMSEL=110 (触发模式)，TI2FP2 被用来启动计数器。

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

- $t_{DELAY}$  由 TMRx\_CC1 寄存器中的值定义。
- $t_{PULSE}$  由自动装载值和比较值之间的差值定义 ( $TMRx\_AR - TMRx\_CC1$ )。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TMRx\_CCM1 寄存器的 OC1MODE=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TMRx\_CCM1 中的 OC1PEN=1 和 TMRx\_CTRL1 寄存器中的 ARPEN；然后在 TMRx\_CC1 寄存器中填写比较值，在 TMRx\_AR 寄存器中填写自动装载值，设置 UEVG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，C1P=0。

在这个例子中，TMRx\_CTRL1 寄存器中的 DIR 和 CMSEL 位应该置低。因为只需要一个脉冲，所以必须设置 TMRx\_CTRL1 寄存器中的 OPMODE=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

**特殊情况：OCx 快速使能：**

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

如果要以最小延时输出波形，可以设置 TMRx\_CCMx 寄存器中的 OCxFEN 位；此时 OCxREF(和 OCx) 直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFEN 只在通道配置为 PWM1 和 PWM2 模式时起作用。

#### 10.4.3.14 TMRx 定时器和外部触发的同步

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

**从模式：复位模式**

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化。同时，如果 TMRx\_CTRL1 寄存

器的 UVERS 位为低, 还产生一个更新事件 UEV。然后所有的预装载寄存器(TMRx\_AR , TMRx\_CCx)都被更新了。

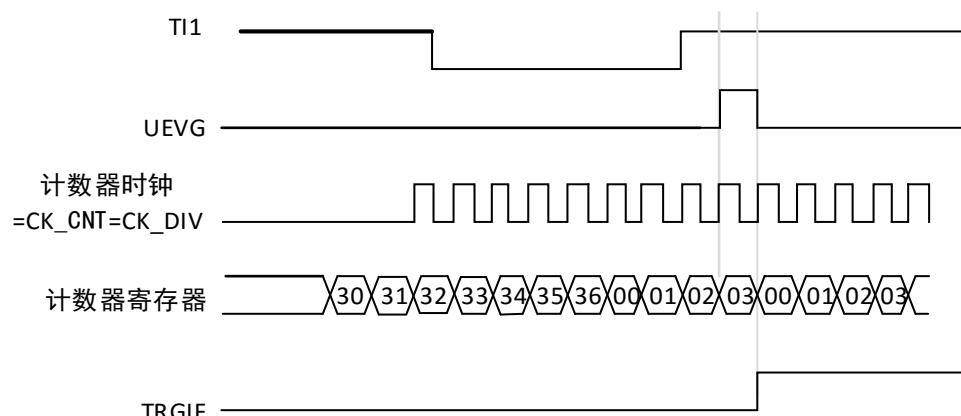
在以下的例子中, TI1 输入端的上升沿导致向上计数器被清零:

- 配置通道1以检测TI1的上升沿。配置输入滤波器的带宽(在本例中, 不需要任何滤波器, 因此保持IC1DF=0000)。触发操作中不使用捕获预分频器, 所以不需要配置。C1SEL位只选择输入捕获源, 即TMRx\_CCM1寄存器中C1SEL=01。置TMRx\_CCE寄存器中C1P=0以确定极性(只检测上升沿)。
- 置TMRx\_SMC寄存器中SMSEL=100, 配置定时器为复位模式; 置TMRx\_SMC寄存器中TRGSEL=101, 选择TI1作为输入源。
- 置TMRx\_CTRL1寄存器中CNTEN=1, 启动计数器。

计数器开始依据内部时钟计数, 然后正常运转直到TI1出现一个上升沿; 此时, 计数器被清零然后从0重新开始计数。同时, 触发标志(TMRx\_STS寄存器中的TRGIF位)被设置, 根据TMRx\_DIE寄存器中TRGIE(中断使能)位和TRGDE(DMA使能)位的设置, 产生一个中断请求或一个DMA请求。

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

图表 10-97 复位模式下的控制电路



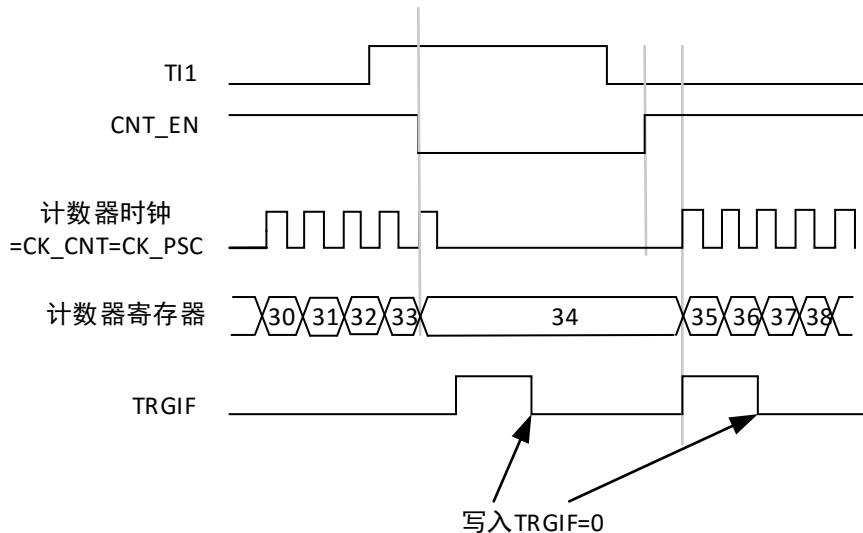
### 从模式: 门控模式

按照选中的输入端电平使能计数器。在如下的例子中, 计数器只在TI1为低时向上计数:

- 配置通道1以检测TI1上的低电平。配置输入滤波器带宽(本例中, 不需要滤波, 所以保持IC1DF=0000)。触发操作中不使用捕获预分频器, 所以不需要配置。C1SEL位用于选择输入捕获源, 置TMRx\_CCM1寄存器中C1SEL=01。置TMRx\_CCE寄存器中C1P=1以确定极性(只检测低电平)。
- 置TMRx\_SMC寄存器中SMSEL=101, 配置定时器为门控模式; 置TMRx\_SMC寄存器中TRGSEL=101, 选择TI1作为输入源。
- 置TMRx\_CTRL1寄存器中CNTEN=1, 启动计数器。在门控模式下, 如果CNTEN=0, 则计数器不能启动, 不论触发输入电平如何。只要TI1为低, 计数器开始依据内部时钟计数, 一旦TI1变高则停止计数。当计数器开始或停止时都设置TMRx\_STS中的TRGIF标志。

TI1上升沿和计数器实际停止之间的延时取决于TI1输入端的重同步电路。

图表 10-98 门控模式下的控制电路



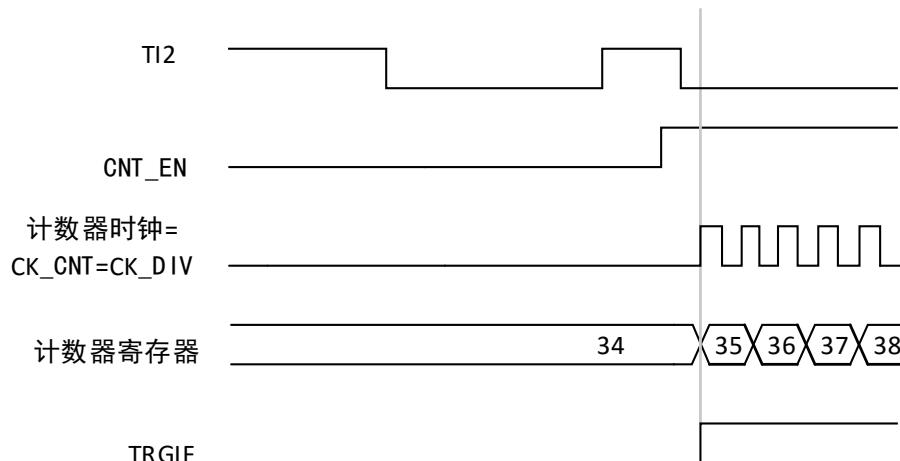
### 从模式：触发模式

输入端上选中的事件使能计数器。在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道2检测TI2的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持IC2DF=0000）。触发操作中不使用捕获预分频器，不需要配置。C2SEL位只用于选择输入捕获源，置TMRx\_CCM1寄存器中C2SEL=01。置TMRx\_CCE寄存器中C2P=0以确定极性（只检测低电平）。
- 置TMRx\_SMC寄存器中SMSEL=110，配置定时器为触发模式；置TMRx\_SMC寄存器中TRGSEL=110，选择TI2作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TRGIF 标志。TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图表 10-99 触发器模式下的控制电路



### 10.4.3.15 定时器同步

所有 TMR 定时器在内部相连，用于定时器同步或链接。详见 [10.2.3.15 节](#)。

### 10.4.3.16 调试模式

当微控制器进入调试模式时（Cortex™-M4 核心停止），根据 DBG 模块中 DBG\_TMRx\_STOP 的设置，TMRx 计数器可以或者继续正常操作，或者停止。详见 [第 18.2.2 节](#)。

## 10.4.4 TMR15寄存器描述

可以用半字（16位）或字（32位）的方式操作这些外设寄存器。

下表中将 TMR15 的所有寄存器映射到一个 16 位可寻址（编址）空间

表格 10-8 TMR15 寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8
0x00	TMRx_CTRL1	保留																			CLKDIV[1: 0]				
	复位值																								
0x04	TMRx_CTRL2	保留																			OC4IS				
	复位值																				OC3NIS				
0x08	TMRx_SMC	保留																			OC3IS				
	复位值																				OC2NIS				
0x0C	TMRx_DIE	保留																			OC2IS				
	复位值																				OC1NIS				
0x10	TMRx_STS	保留																			CLKDIV[1: 0]				
	复位值																				OC1IS				
0x14	TMRx_EVEG	保留																			ARPEN				
	复位值																				0				
0x18	TMRx_CCM1 输出 比较模式	保留																			保留				
	复位值																				MMSEL[2: 0]				
	TMRx_CCM1 输入 捕获模式	保留																			0				
0x1C	TMRx_CCM2 输出 比较模式	保留																			DIR				
	复位值																				0				

	TMRx_CCM2 输入 捕获模式	保留	IC4DF[3: 0]				IC4DIV[1: 0]				C4SI[1: 0]				IC3DF[3: 0]				IC3DIV[1: 0]				C3SE[1: 0]			
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	TMRx_CCE	保留	C4P	C4EN	C3NP	C3NEN	C3P	C3EN	C2NP	C2NEN	C2P	C2NEN	C1NP	C1NEN	C1P	C1EN	C1EN	C1EN	C1EN	C1EN	C1EN	C1EN	C1EN	C1EN	C1EN	
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	TMRx_CNT	保留	CNT[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TMRx_DIV	保留	DIV[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TMRx_AR	保留	AR[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	TMRx_RC	保留	RC[7: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	TMRx_CC1	保留	CC1[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	TMRx_CC2	保留	CC2[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	TMRx_CC3	保留	CC3[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	TMRx_CC4	保留	CC4[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TMRx_BRKDT	保留	MOEN	AOEN	BRKP	BRKEN	OSIMR	OSIMI	LOCKC[1: 0]	DTGS[7: 0]				DTGS[7: 0]				DTGS[7: 0]				DTGS[7: 0]				
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TMRx_DMAC	保留	DBLEN[4: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	TMRx_DMABA	保留	DMABA[15: 0]				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	复位值	保留	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 10.4.4.1 TMR15控制寄存器1 (TMR15\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留						CLKDIV [1: 0]	ARP EN	保留		保留	OPMODE	UVERS	UEVDIS	CNT EN			
res															rw		
位 15: 10		保留, 始终读为 0。															

位 9: 8	<b>CLKDIV[1: 0]:</b> 时钟分频因子 (Clock division) 这 2 位定义在定时器时钟 (CK_INT) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: 保留, 不要使用这个配置
位 7	<b>ARPEN:</b> 自动重装载预装载允许位 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲; 1: TMRx_AR 寄存器被装入缓冲器。
位 6: 4	保留
位 3	<b>OPMODE:</b> 单脉冲模式 (One pulse mode) 0: 在发生更新事件时, 计数器不停止; 1: 在发生下一次更新事件 (清除 CNTEN 位) 时, 计数器停止。
位 2	<b>UVERS:</b> 更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器) 1: 禁止 UEV。不产生更新事件, 影子寄存器 (AR、DIV、CCx) 保持它们的值。如果设置了 UEVG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化。
位 0	<b>CNTEN:</b> 使能计数器 (Counter enable) 0: 禁止计数器; 1: 使能计数器。 注: 在软件设置了 CNTEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CNTEN 位。

#### 10.4.4.2 TMR15控制寄存器2 (TMR15\_CTRL2)

偏移地址: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				OC2 IS	OC1 NIS	OC1 IS	TI1S EL		MMSEL[2: 0]	CDS EL	CUS EL	保留	CPC		
res				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	res	rw

位 15: 11	保留, 始终读为 0。
位 10	<b>OC2IS:</b> 输出空闲状态 2 (OC2 输出)。参见 OC1IS 位。
位 9	<b>OC1NIS:</b> 输出空闲状态 1 (OC1N 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 死区后 OC1N=0; 1: 当 MOEN=0 时, 死区后 OC1N=1。 注: 已经设置了 LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。

位 8	<b>OC1IS:</b> 输出空闲状态 1 (OC1 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=0。 1: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=1。 注: 已经设置了 LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。
位 7	保留
位 6: 4	<b>MMSEL[2: 0]:</b> 主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 – TMRx_EVEG 寄存器的 UEVG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式), 则 TRGO 上的信号相对实际的复位会有一个延迟。 001: 使能–计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CNTEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时, TRGO 上会有一个延迟, 除非选择了主/从模式 (见 TMRx_SMC 寄存器中 MSMODE 位的描述)。 010: 更新 – 更新事件被选为触发输入 (TRGO)。例如, 一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 – 在发生一次捕获或一次比较成功时, 当要设置 C1IF 标志时 (即使它已经为高), 触发输出送出一个正脉冲 (TRGO)。 100: 比较 – OC1REF 信号被用于作为触发输出 (TRGO)。 101: 比较 – OC2REF 信号被用于作为触发输出 (TRGO)。
位 3	<b>CDSEL:</b> 捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0: 当发生 CCx 事件时, 送出 CCx 的 DMA 请求; 1: 当发生更新事件时, 送出 CCx 的 DMA 请求。
位 2	<b>CUSEL:</b> 捕获/比较控制更新选择 (Capture/compare control update selection) 0: 如果捕获/比较控制位是预装载的 (CPC=1), 只能通过设置 HALL 位更新它们; 1: 如果捕获/比较控制位是预装载的 (CPC=1), 可以通过设置 HALL 位或 TRGI 上的一个上升沿更新它们。 注: 该位只对具有互补输出的通道起作用。
位 1	保留, 始终读为 0。
位 0	<b>CPC:</b> 捕获/比较预装载控制位 (Capture/compare preloaded control) 0: CxEN, CxNEN 和 OCxMODE 位不是预装载的; 1: CxEN, CxNEN 和 OCxMODE 位是预装载的; 设置该位后, 它们只在发生一个 HALL 事件的时候 (设置了 HALL 位或检测到 TRGI 的上升沿, 依据 CUSEL 位) 被更新。 注: 该位只对具有互补输出的通道起作用。

#### 10.4.4.3 TMR15从模式控制寄存器 (TMR15\_SMC)

偏移地址: 0x08

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				MSM ODE	TRGSEL[2: 0]				保留		SMSEL[2: 0]				
res		rw		rw		rw		rw		res		rw		rw	

位 15: 8	保留
位 7	<b>MSMODE:</b> 主/从模式 (Master/slave mode) 0: 无作用; 1: 触发输入 (TRGI) 上的事件被延迟了, 以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。

位 6: 4	<b>TRGSEL[2: 0]: 触发选择 (Trigger selection)</b> 这 3 位选择用于同步计数器的触发输入。 000: 保留 100: TI1 的边沿检测器 (TI1F_ED) 001: 内部触发 1 (ITR1) 101: 滤波后的定时器输入 1 (TI1FP1) 010: 内部触发 2 (ITR2) 110: 滤波后的定时器输入 2 (TI2FP2) 011: 内部触发 3 (ITR3) 111: 保留 更多有关 ITRx 的细节, 参见表 10-13。 注: 这些位只能在未用到 (如 SMSEL=000) 时被改变, 以避免在改变时产生错误的边沿检测。
	位 3 保留, 始终读为 0。
	<b>SMSEL[2: 0]: 从模式选择 (Slave mode selection)</b> 当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关 (见输入控制寄存器和控制寄存器的说明) 000: 关闭从模式 – 如果 CNTEN=1, 则预分频器直接由内部时钟驱动。 001: 保留 010: 保留 011: 保留 100: 复位模式 – 选中的触发输入 (TRGI) 的上升沿重新初始化计数器, 并且产生一个更新寄存器的信号。 101: 门控模式 – 当触发输入 (TRGI) 为高时, 计数器的时钟开启。一旦触发输入变为低, 则计数器停止 (但不复位)。计数器的启动和停止都是受控的。 110: 触发模式 – 计数器在触发输入 TRGI 的上升沿启动 (但不复位), 只有计数器的启动是受控的。 111: 外部时钟模式 1 – 选中的触发输入 (TRGI) 的上升沿驱动计数器。 注: 如果 TI1F_EN 被选为触发输入 (TRGSEL=100) 时, 不要使用门控模式。这是因为, TI1F_ED 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。
	位 2: 0

表格 10-9 TMRx 内部触发连接

从定时器	ITR0 (TRGSEL=000)	ITR1 (TRGSEL=001)	ITR2 (TRGSEL=010)	ITR3 (TRGSEL=011)
TMR15	0	TMR3_TRGO	TMR16_OC	TMR17_OC

#### 10.4.4.4 TMR15 DMA/中断使能寄存器 (TMR15\_DIE)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	TRG DE	HALL DE	保留	C2D E	C1D E	UEV DE	BRKI E	TRGI E	HALL IE	保留	C2IE	C1IE	UEVI E		
res	rw	rw	res	res	rw	rw	rw	rw	rw	rw	res	res	rw	rw	rw

位 15	保留, 始终读为 0。
位 14	<b>TRGDE:</b> 允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求; 1: 允许触发 DMA 请求。
位 13	<b>HALLDE:</b> 允许 HALL 的 DMA 请求 (HALL DMA request enable) 0: 禁止 HALL 的 DMA 请求; 1: 允许 HALL 的 DMA 请求。
位 12: 11	保留
位 10	<b>C2DE:</b> 允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求; 1: 允许捕获/比较 2 的 DMA 请求。

位 9	<b>C1DE:</b> 允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求; 1: 允许捕获/比较 1 的 DMA 请求。
位 8	<b>UEVDE:</b> 允许更新的 DMA 请求 (Update DMA request enable) 0: 禁止更新的 DMA 请求; 1: 允许更新的 DMA 请求。
位 7	<b>BRKIE:</b> 允许刹车中断 (Break interrupt enable) 0: 禁止刹车中断; 1: 允许刹车中断。
位 6	<b>TRGIE:</b> 触发中断使能 (Trigger interrupt enable) 0: 禁止触发中断; 1: 使能触发中断。
位 5	<b>HALLIE:</b> 允许 HALL 中断 (HALL interrupt enable) 0: 禁止 HALL 中断; 1: 允许 HALL 中断。
位 4: 3	保留
位 2	<b>C2IE:</b> 允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断; 1: 允许捕获/比较 2 中断。
位 1	<b>C1IE:</b> 允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断; 1: 允许捕获/比较 1 中断。
位 0	<b>UEVIE:</b> 允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

#### 10.4.4.5 TMR15状态寄存器 (TMR15\_STS)

偏移地址: 0x10

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				C2OF F	C1OF F	保留	BRK IF	TRG IF	HAL LIF	保留	C2IF F	C1IF F	UEV IF		
res				rw	rw	res	rw	rw	rw	res	res	rw	rw	rw	rw

位 15: 11	保留, 始终读为 0。
位 10	<b>C2OF:</b> 捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 C1OF 描述。
位 9	<b>C1OF:</b> 捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 计数器的值被捕获到 TMRx_CC1 寄存器时, C1IF 的状态已经为'1'。
位 8	保留, 始终读为 0。
位 7	<b>BRKIF:</b> 刹车中断标记 (Break interrupt flag) 一旦刹车输入有效, 由硬件对该位置'1'。如果刹车输入无效, 则该位可由软件清'0'。 0: 无刹车事件产生; 1: 刹车输入上检测到有效电平。

位 6	<b>TRGIF:</b> 触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置'1'。它由软件清'0'。 0: 无触发器事件产生; 1: 触发中断等待响应。
位 5	<b>HALLIF:</b> HALL 中断标记 (HALL interrupt flag) 一旦产生 HALL 事件 (当捕获/比较控制位: CxEN、CxNEN、OCxMODE 已被更新) 该位由硬件置'1'。它由软件清'0'。 0: 无 HALL 事件产生; 1: HALL 中断等待响应。
位 4: 3	保留
位 2	<b>C2IF:</b> 捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 C1IF 描述。
位 1	<b>C1IF:</b> 捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式: 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外 (参考 TMRx_CTRL1 寄存器的 CMSEL 位)。它由软件清'0'。 0: 无匹配发生; 1: TMRx_CNT 的值与 TMRx_CC1 的值匹配。 当 TMRx_CC1 的内容大于 TMRx_AR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, C1IF 位变高。 如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置'1', 它由软件清'0'或通过读 TMRx_CC1 清'0'。 0: 无输入捕获产生; 1: 计数器值已被捕获 (拷贝) 至 TMRx_CC1 (在 IC1 上检测到与所选极性相同的边沿)。
位 0	<b>UEVIF:</b> 更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': - 若 TMRx_CTRL1 寄存器的 UEVDIS=0, 当重复计数器数值上溢或下溢时 (重复计数器=0 时产生更新事件)。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当设置 TMRx_EVEG 寄存器的 UEVG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当计数器 CNT 被触发事件重新初始化时。 (参考 <a href="#">10.6.4.3: TMR1 从模式控制寄存器 (TMRx_SMC)</a> )。

#### 10.4.4.6 TMR15事件产生寄存器 (TMR15\_EVEG)

偏移地址: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				BRK G	TRG G	HALL G	保留		C2G	C1G	UEV G				
res				rw	rw	rw	res	res	rw	rw	rw				

位 15: 8	保留, 始终读为 0。
位 7	<b>BRKG:</b> 产生刹车事件 (Break generation) 该位由软件置'1', 用于产生一个刹车事件, 由硬件自动清'0'。 0: 无动作; 1: 产生一个刹车事件。此时 MOEN=0、BRKIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。

位 6	<b>TRGG:</b> 产生触发事件 (Trigger generation) 该位由软件置'1', 用于产生一个触发事件, 由硬件自动清'0'。 0: 无动作; 1: TMRx_STS 寄存器的 TRGIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。
位 5	<b>HALLG:</b> 捕获/比较事件, 产生控制更新 (Capture/Compare control update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 当 CPC=1, 允许更新 CxEN、CxNEN、OCxMODE 位。 注: 该位只对拥有互补输出的通道有效。
位 4: 3	保留
位 2	<b>C2G:</b> 产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 C1G 描述。
位 1	<b>C1G:</b> 产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1', 用于产生一个捕获/比较事件, 由硬件自动清'0'。 0: 无动作; 1: 在通道 CC1 上产生一个捕获/比较事件: 若通道 CC1 配置为输出: 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。 若通道 CC1 配置为输入: 当前的计数器值被捕获至 TMRx_CC1 寄存器; 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 C1IF 已经为 1, 则设置 C1OF=1。
位 0	<b>UEVG:</b> 产生更新事件 (Update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0 (向上计数) 则计数器被清'0'; 若 DIR=1 (向下计数) 则取计数器取 TMRx_AR 的值。

#### 10.4.4.7 TMR15 捕获/比较模式寄存器1 (TMR15\_CCM1)

偏移地址: 0x18

复位值: 0x0000

通道可用于输入 (捕获模式) 或输出 (比较模式), 通道的方向由相应的 CxSEL 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能, ICxx 描述了通道在输入模式下的功能。因此必须注意, 同一个位在输出模式和输入模式下的功能是不同的。

##### 输出比较模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	OC2MODE[2: 0]	OC2 PEN	OC2 FEN	C2SEL[1: 0]	保留	OC1MODE[2: 0]	OC1 PEN	OC1 FEN	C1SEL[1: 0]						

res	rw	rw	rw	rw	rw	rw	rw	res	rw						
位 15	保留														
位 14: 12	<b>OC2MODE[2: 0]:</b> 输出比较 2 模式 (Output Compare 2 mode)														
位 11	<b>OC2PEN:</b> 输出比较 2 预装载使能 (Output Compare 2 preload enable)														
位 10	<b>OC2FEN:</b> 输出比较 2 快速使能 (Output Compare 2 fast enable)														

位 9: 8	<p><b>C2SEL[1: 0]:</b> 捕获/比较 2 选择。 (Capture/Compare 2 selection)          该位定义通道的方向 (输入/输出), 及输入脚的选择:          00: CC2 通道被配置为输出;          01: CC2 通道被配置为输入, IC2 映射在 TI2 上;          10: CC2 通道被配置为输入, IC2 映射在 TI1 上;          11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。          注: C2SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C2NEN=0) 才是可写的。</p>
位 7	保留
位 6: 4	<p><b>OC1MODE[2: 0]:</b> 输出比较 1 模式 (Output Compare 1 mode)          该 3 位定义了输出参考信号 OC1REF 的动作, 而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效, 而 OC1、OC1N 的有效电平取决于 C1P、C1NP 位。          000: 冻结。输出比较寄存器 TMRx_CC1 与计数器 TMRx_CNT 间的比较对 OC1REF 不起作用;          001: 匹配时设置通道 1 为有效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时, 强制 OC1REF 为高。          010: 匹配时设置通道 1 为无效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时, 强制 OC1REF 为低。          011: 翻转。当 TMRx_CC1=TMRx_CNT 时, 翻转 OC1REF 的电平。          100: 强制为无效电平。强制 OC1REF 为低。          101: 强制为有效电平。强制 OC1REF 为高。          110: PWM 模式 1— 在向上计数时, 一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为有效电平, 否则为无效电平; 在向下计数时, 一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为无效电平 (OC1REF=0), 否则为有效电平 (OC1REF=1)。          111: PWM 模式 2—在向上计数时, 一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为无效电平, 否则为有效电平; 在向下计数时, 一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为有效电平, 否则为无效电平。          注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。          注 2: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC1REF 电平才改变。</p>
位 3	<p><b>OC1PEN:</b> 输出比较 1 预装载使能 (Output Compare 1 preload enable)          0: 禁止 TMRx_CC1 寄存器的预装载功能, 可随时写入 TMRx_CC1 寄存器, 并且新写入的数值立即起作用。          1: 开启 TMRx_CC1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TMRx_CC1 的预装载值在更新事件到来时被加载至当前寄存器中。          注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。          注 2: 仅在单脉冲模式下 (TMRx_CTRL1 寄存器的 OPMODE=1), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。</p>
位 2	<p><b>OC1FEN:</b> 输出比较 1 快速使能 (Output Compare 1 fast enable)          该位用于加快 CC 输出对触发输入事件的响应。          0: 根据计数器与 CC1 的值, CC1 正常操作, 即使触发器是打开的。当触发器的输入有一个有效沿时, 激活 CC1 输出的最小延时为 5 个时钟周期。          1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此, OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OC1FEN 只在通道被配置成 PWM1 或 PWM2 模式时起作用。</p>
位 1: 0	<p><b>C1SEL[1: 0]:</b> 捕获/比较 1 选择。 (Capture/Compare 1 selection)          这 2 位定义通道的方向 (输入/输出), 及输入脚的选择:          00: CC1 通道被配置为输出;          01: CC1 通道被配置为输入, IC1 映射在 TI1 上;          10: CC1 通道被配置为输入, IC1 映射在 TI2 上;          11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。          注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=0) 才是可写的。</p>

## 输入捕获模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2DF[3: 0]		IC2DIV[1: 0]		C2SEL[1: 0]		IC1DF[3: 0]		IC1DIV[1: 0]		C1SEL[1: 0]					
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 12	<b>IC2DF[3: 0]:</b> 输入捕获 2 滤波器 (Input capture 2 filter)
位 11: 10	<b>IC2DIV[1: 0]:</b> 输入/捕获 2 预分频器 (Input capture 2 prescaler)
位 9: 8	<b>C2SEL[1: 0]:</b> 捕获/比较 2 选择 (Capture/Compare 2 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C2SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C2EN=0) 才是可写的。
位 7: 4	<b>ETDF[3: 0]:</b> 外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上, 数字滤波器是一个事件计数器, 它记录到 N 个事件后会产生一个输出的跳变。 0000: 无滤波器, 以 $f_{DTS}$ 采样                    1000: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=6 0001: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=2        1001: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=8 0010: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=4        1010: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=5 0011: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=8        1011: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=6 0100: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=6        1100: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=8 0101: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=8        1101: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=5 0110: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=6        1110: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=6 0111: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=8        1111: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=8
位 3: 2	<b>IC1DIV[1: 0]:</b> 输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入 (IC1) 的预分频系数.一旦 C1EN=0 (TMRx_CCE 寄存器中), 则预分频器复位。 00: 无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获; 01: 每 2 个事件触发一次捕获; 10: 每 4 个事件触发一次捕获; 11: 每 8 个事件触发一次捕获。
位 1: 0	<b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=0) 才是可写的。

## 10.4.4.8 TMR15捕获/比较使能寄存器 (TMR15\_CCE)

偏移地址: 0x20

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留		C2NP		保留		C2P		C2NEN		C1NP		C1NEN		C1P		C1EN	
res		rw		res		rw		rw		rw		rw		rw		rw	

位 15: 8	保留, 始终读为 0。
位 7	<b>C2NP:</b> 输入/捕获 2 互补输出极性 (Capture/Compare 2 complementary output polarity) 参考 C1NP 的描述。
位 6	保留
位 5	<b>C2P:</b> 输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 C1P 的描述。
位 4	<b>C2NEN:</b> 输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 C1OE 的描述。
位 3	<b>C1NP:</b> 输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) <b>CC1 通道配置为输出:</b> 0: OC1N 高电平有效; 1: OC1N 低电平有效。 <b>CC1 通道配置为输入:</b> C1NP 结合 C1P 定义 TI1FP1/TI2FP1 极性 (参考 C1P 描述) 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2 且 C1SEL=00 (通道配置为输出) 则该位不能被修改。
位 2	<b>C1NEN:</b> 输入/捕获 1 互补输出使能 (Capture/Compare 1 complementary output enable) 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。
位 1	<b>C1P:</b> 输入/捕获 1 输出极性 (Capture/Compare 1 output polarity) <b>CC1 通道配置为输出:</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1 通道配置为输入:</b> CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的极性信号作为触发或捕获信号。 00: 不反相/上升沿。电路 TIxFP1 上升沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 不反相 (触发中的门控模式)。 01: 反相/下降沿。电路对 TIxFP1 下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 反相 (触发中的门控模式)。 10: 保留。 11: 不反相/双沿。电路同时对 TIxFP1 上升沿和下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 是不反相 (触发中门控模式)。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2, 则该位不能被修改。
位 0	<b>C1EN:</b> 输入/捕获 1 输出使能 (Capture/Compare 1 output enable) <b>CC1 通道配置为输出:</b> 0 : 关闭— OC1 禁止输出, 因此 OC1 的输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1NEN 位的值。 1 : 开启— OC1 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1NEN 位的值。 <b>CC1 通道配置为输入:</b> 该位决定了计数器的值是否能捕获入 TMRx_CC1 寄存器。 0: 捕获禁止; 1: 捕获使能。

表格 10-10 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态 (1)	
MOEN 位	OSIMI 位	OSIMR 位	CxEN 位	CxNEN 位	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0

		0	0	1	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		0	1	0	OCxREF+极性, OCx= OCxREF xor CxP, OCx_EN=1	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1
		1	0	0	输出禁止 (与定时器断开) OCx=CxP, OCx_EN=0	输出禁止 (与定时器断开) OCxN=CxNP, OCxN_EN=0
		1	0	1	关闭状态 (输出使能且为无效电平) OCx=CxP, OCx_EN=1	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		1	1	0	OCxREF + 极性, OCx= OCxREF xor CxP, OCx_EN=1	关闭状态 (输出使能且为无效电平) OCxN=CxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1
0	X	0	0	0	输出禁止 (与定时器断开) 异步地: OCx=CxP , OCx_EN=0 , OCxN=CxNP , OCxN_EN=0; 若时钟存在: 经过一个死区时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	
		0	1	0		
		1	0	1		
		0	1	0		
		1	0	1	关闭状态 (输出使能且为无效电平) 异步地: OCx=CxP , OCx_EN=1 , OCxN=CxNP , OCxN_EN=1; 若时钟存在: 经过一个死区 时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	
		0	1	0		
		1	0	1		
		1	1	0		

注意：如果一个通道的 2 个输出都没有使用( $CxEN = CxNEN = 0$ )，那么  $OC1IS$ ,  $OCxNIS$ ,  $CxP$  和  $CxNP$  都必须清零。

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

#### 10.4.4.9 TMR15计数器 (TMR15\_CNT)

偏移地址: 0x24

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0		CNT[15: 0]: 计数器的值 (Counter value)													

#### 10.4.4.10 TMR15预分频器 (TMR15\_DIV)

偏移地址: 0x28

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>DIV[15: 0]:</b> 预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_DIV} / (DIV[15: 0] + 1)$ 。 DIV 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TMR_EVEG 的 UEVG 位清'0'或被工作在复位模式的从控制器清'0'。														

#### 10.4.4.11 TMR15自动重装载寄存器 (TMR15\_AR)

偏移地址: 0x2C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>AR[15: 0]:</b> 自动重装载的值 (Prescaler value) AR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 <a href="#">10.4.3.1 节</a> ：有关 AR 的更新和动作。当自动重装载的值为空时，计数器不工作。														

#### 10.4.4.12 TMR15重复计数寄存器 (TMR15\_RC)

偏移地址: 0x30

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								RC[7: 0]							
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 8	保留，始终读为 0。														
位 7: 0	<b>RC[7: 0]:</b> 重复计数器的值 (Repetition counter value) 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）。如果允许产生更新中断，则会同时影响产生更新中断的速率。每次向下计数器 RC_CNT 达到 0，会产生一个更新事件并且计数器 RC_CNT 重新从 RC 值开始计数。由于 RC_CNT 只有在周期更新事件 U_RC 发生时才重载 RC 值，因此对 TMRx_RC 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(RC+1) 对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；														

#### 10.4.4.13 TMR15捕获/比较寄存器 1 (TMR15\_CC1)

偏移地址: 0x34

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC1[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 0	<p><b>CC1[15: 0]:</b> 捕获/比较通道 1 的值 (Capture/Compare 1 value)  <b>若 CC1 通道配置为输出:</b>            CC1 包含了装入当前捕获/比较 1 寄存器的值 (预装载值)。如果在 TMRx_CCM1 寄存器 (OC1PEN 位) 中未选择预装载功能, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC1 端口上产生输出信号。  <b>若 CC1 通道配置为输入:</b>            CC1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。</p>
---------	---

#### 10.4.4.14 TMR15捕获/比较寄存器2 (TMR15\_CC2)

偏移地址: 0x38

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC2[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 0	<p><b>CC2[15: 0]:</b> 捕获/比较通道 2 的值 (Capture/Compare 2 value)  <b>若 CC2 通道配置为输出:</b>            CC2 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TMRx_CCM2 寄存器 (OC2PEN 位) 中未选择预装载特性, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 2 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC2 端口上产生输出信号。  <b>若 CC2 通道配置为输入:</b>            CC2 包含了由上一次输入捕获 2 事件 (IC2) 传输的计数器值。</p>
---------	---

#### 10.4.4.15 TMR15刹车和死区寄存器 (TMR15\_BRKDT)

偏移地址: 0x44

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MO EN	AO EN	BR KP	BRK EN	OSI MR	OSI MI	LOCKC[1: 0]	DTGS[7: 0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注意: 根据锁定设置, AOEN、BRKP、BRKEN、OSIMR 和 DTGS[7: 0]位均可被写保护, 有必要在第一次写入 TMRx\_BRKDT 寄存器时对它们进行配置。

位 15	<p><b>MOEN:</b> 主输出使能 (Main output enable)            一旦刹车输入有效, 该位被硬件异步清'0'。根据 AOEN 位的设置值, 该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。            0: 禁止 OC 和 OCN 输出或强制为空闲状态;            1: 如果设置了相应的使能位 (TMRx_CCE 寄存器的 CxEN、CxNEN 位), 则开启 OC 和 OCN 输出。            有关 OC/OCN 使能的细节, 参见 <a href="#">10.6.4.9 节: TMR1 捕获/比较使能寄存器 (TMRx_CCE)</a>。</p>
位 14	<p><b>AOEN:</b> 自动输出使能 (Automatic output enable)            0: MOEN 只能被软件置'1';            1: MOEN 能被软件置'1'或在下一个更新事件被自动置'1' (如果刹车输入无效)。            注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。</p>

位 13	<b>BRKP:</b> 刹车输入极性 (Break polarity) 0: 刹车输入低电平有效; 1: 刹车输入高电平有效。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
位 12	<b>BRKEN:</b> 刹车功能使能 (Break enable) 0: 禁止刹车输入 (BRK 及 CFD 时钟失效事件); 1: 开启刹车输入 (BRK 及 CFD 时钟失效事件)。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
位 11	<b>OSIMR:</b> 运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOEN=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSIMR 位。 参考 OC/OCN 使能的详细说明 ( <a href="#">10.6.4.9 节</a> , TMR1 捕获/比较使能寄存器 (TMRx_CCE))。 0: 当定时器不工作时, 禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0); 1: 当定时器不工作时, 一旦 CxEN=1 或 CxNEN=1, 首先开启 OC/OCN 并输出无效电平, 然后置 OC/OCN 使能输出信号=1。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 2, 则该位不能被修改。
位 10	<b>OSIMI:</b> 空闲模式下“关闭状态”选择 (Off-state selection for Idle mode) 该位用于当 MOEN=0 且通道设为输出时。参考 OC/OCN 使能的详细说明 ( <a href="#">10.6.4.9 节</a> , TMR1 捕获/比较使能寄存器 (TMRx_CCE))。 0 : 当定时器不工作时, 禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0) ; 1 : 当定时器不工作时, 一旦 CxEN=1 或 CxNEN=1 , OC/OCN 首先输出其空闲电平, 然后 OC/OCN 使能输出信号=1。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 2, 则该位不能被修改。
位 9: 8	<b>LOCKC[1: 0]:</b> 锁定设置 (LENock configuration) 该位为防止软件错误而提供写保护。 00: 锁定关闭, 寄存器无写保护; 01: 锁定级别 1, 不能写入 TMRx_BRKDT 寄存器的 DTGS、BRKEN、BRKP、AOEN 位和 TMRx_CTRL2 寄存器的 OCXIS/OCXNIS 位; 10: 锁定级别 2, 不能写入锁定级别 1 中的各位, 也不能写入 CC 极性位 (一旦相关通道通过 CxSEL 位设为输出, CC 极性位是 TMRx_CCE 寄存器的 CxP/CCNxP 位) 以及 OSIMR/OSIMI 位; 11: 锁定级别 3, 不能写入锁定级别 2 中的各位, 也不能写入 CC 控制位 (一旦相关通道通过 CxSEL 位设为输出, CC 控制位是 TMRx_CCMx 寄存器的 OCxMODE/OCxPEN 位); 注: 在系统复位后, 只能写一次 LOCKC 位, 一旦写入 TMRx_BRKDT 寄存器, 则其内容冻结直至复位。
位 7: 0	<b>DTGS[7: 0]:</b> 死区发生器设置 (Dead-time generator setup) 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间: DTGS[7: 5]=0xx => DT=DTGS[7: 0] × Tdtg,      Tdtg = T <sub>DTS</sub> ; DTGS[7: 5]=10x => DT= (64+DTGS[5: 0]) × Tdtg,    Tdtg = 2 × T <sub>DTS</sub> ; DTGS[7: 5]=110 => DT= (32+DTGS[4: 0]) × Tdtg,    Tdtg = 8 × T <sub>DTS</sub> ; DTGS[7: 5]=111 => DT= (32+DTGS[4: 0]) × Tdtg,    Tdtg = 16 × T <sub>DTS</sub> ; 例: 若 T <sub>DTS</sub> = 125ns (8MHZ), 可能的死区时间为: 0 到 15875ns, 若步长时间为 125ns; 16us 到 31750ns, 若步长时间为 250ns; 32us 到 63us, 若步长时间为 1us; 64us 到 126us, 若步长时间为 2us; 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 1、2 或 3, 则不能修改这些位。

#### 10.4.4.16 TMR15 DMA控制寄存器 (TMR15\_DMAR)

偏移地址: 0x48

复位值: 0x0000

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

位 15: 13	保留, 始终读为 0。
位 12: 8	<b>DBLEN[4: 0]:</b> DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度 (当对 TMRx_DMABA 寄存器进行读或写时, 定时器则进行一次连续传送), 即: 定义传输的次数, 传输可以是半字 (双字节) 或字节: 00000: 1 次传输      00001: 2 次传输 00010: 3 次传输      ..... ....      ..      10001: 18 次传输 例: 我们考虑这样的传输: DBLEN=7, ADDR=TMR2_CTRL1 - 如果 DBLEN=7, ADDR=TMR2_CTRL1 表示待传输数据的地址, 那么传输的地址由下式给出: (TMRx_CTRL1 的地址) + ADDR + (DMA 索引), 其中 DMA 索引=DBLEN 其中 (TMRx_CTRL1 的地址) + ADDR 再加上 7, 给出了将要写入或者读出数据的地址, 这样数据的传输将发生在从地址 (TMRx_CTRL1 的地址) + ADDR 开始的 7 个寄存器。 根据 DMA 数据长度的设置, 可能发生以下情况: - 如果设置数据为半字 (16 位), 那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节, 数据仍然会传输给全部 7 个寄存器: 第一个寄存器包含第一个 MSB 字节, 第二个寄存器包含第一个 LSB 字节, 以此类推。因此对于定时器, 用户必须指定由 DMA 传输的数据宽度。
位 7: 5	保留, 始终读为 0。
位 4: 0	<b>ADDR[4: 0]:</b> DMA 基址 (DMA base address) 这些位定义了 DMA 在连续模式下的基址 (当对 TMRx_DMAR 寄存器进行读或写时), ADDR 定义为从 TMRx_CTRL1 寄存器所在地址开始的偏移量: 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_SMC, ....

#### 10.4.4.17 TMR15连续模式的DMA地址 (TMR15\_DMABA)

偏移地址: 0x4C

复位值: 0x0000

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

位 15: 0	<b>DMABA[15: 0]:</b> DMA 连续传送寄存器 (DMA register for burst accesses) 对 TMRx_DMABA 寄存器的读或写会导致对以下地址所在寄存器的存取操作: TMRx_CTRL1 地址 + ADDR + DMA 索引, 其中: “TMRx_CTRL1 地址”是控制寄存器 1 (TMRx_CTRL1) 所在的地址; “ADDR”是 TMRx_DMAR 寄存器中定义的基地址; “DMA 索引”是由 DMA 自动控制的偏移量, 它取决于 TMRx_DMAR 寄存器中定义的 DBLEN。
---------	---

### 10.5 通用定时器 (TMR16/17)

### 10.5.1 TMRx简介

通用定时器（TMRx）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM 等）。

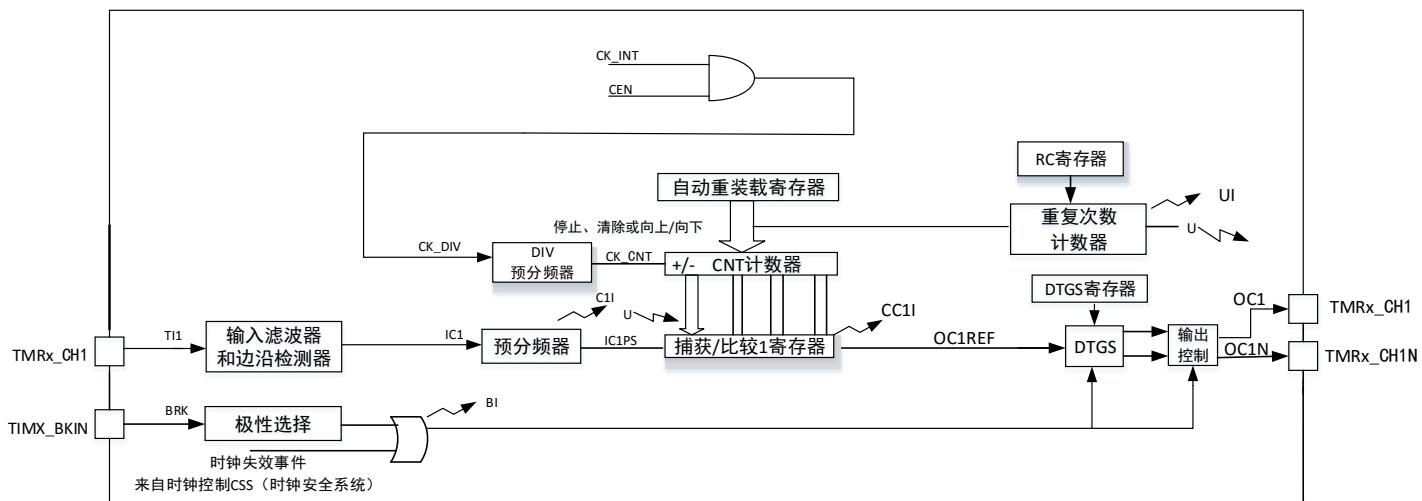
使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

### 10.5.2 TMRx主要特性

TMRx 定时器的功能包括：

- 16位自动装载计数器
- 16位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为1~65535之间的任意数值
- 1个独立通道：
  - 输入捕获
  - 输出比较
  - PWM生成（边缘对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器溢出
  - 输入捕获
  - 输出比较
  - 刹车信号输入

图表 10-100 通用控制定时器 16/17 框图



注意： 根据控制位的设定，在 U (更新) 事件时传送预加载寄存器的内容至工作寄存器

▲ 事件

↗ 中断和 DMA 输出

## 10.5.3 TMRx 功能描述

### 10.5.3.1 时基单元

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

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器 (TMRx\_CNT)
- 预分频器寄存器 (TMRx\_DIV)
- 自动装载寄存器 (TMRx\_AR)
- 重复次数寄存器 (TMRx\_RC)

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TMRx\_CTRL1 寄存器中的自动装载预装载使能位(ARPEN)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TMRx\_CTRL1 寄存器中的 UEVDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK\_CNT 驱动，仅当设置了计数器 TMRx\_CTRL1 寄存器中的计数器使能位 (CNTEN) 时，CK\_CNT 才有效。（更多有关使能计数器的细节，请参见控制器的从模式描述）。

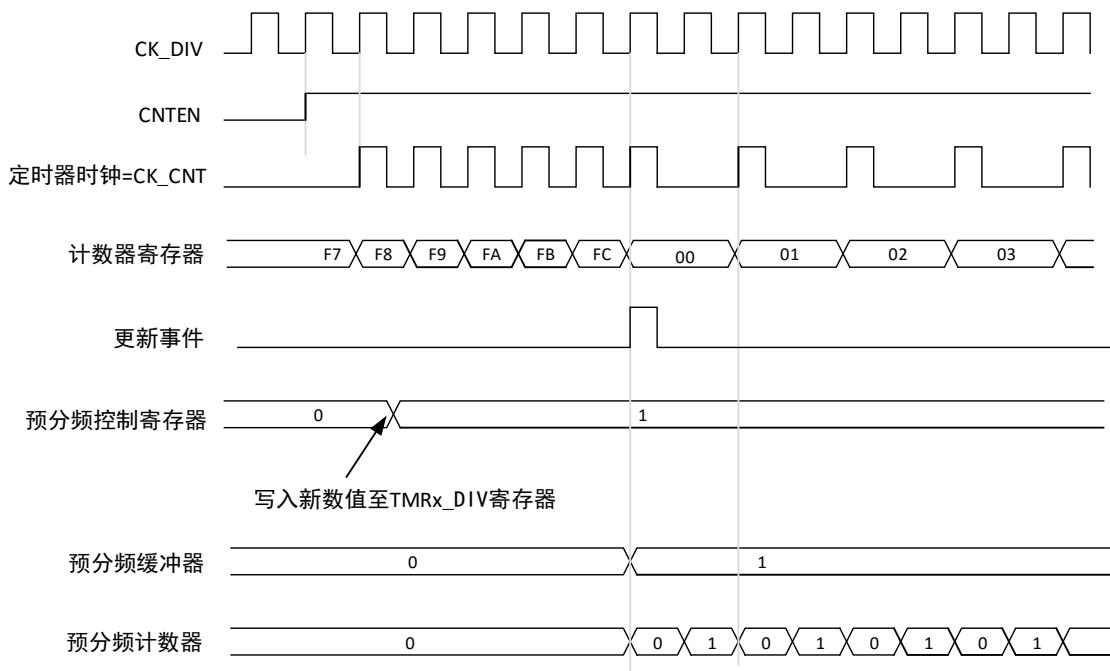
注意，在设置了 TMRx\_CTRL 寄存器的 CNTEN 位的一个时钟周期后，计数器开始计数。

#### 预分频器描述

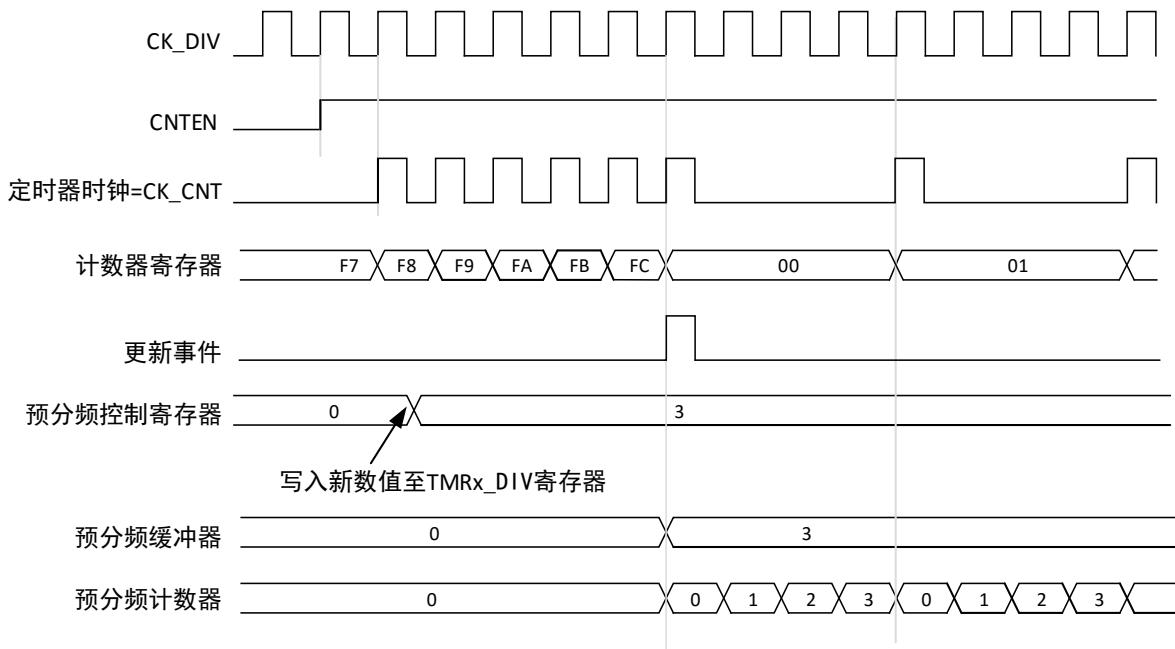
预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TMRx\_DIV 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

[图 10-101](#) 和 [图 10-102](#) 给出了在预分频器运行时，更改计数器参数的例子。

图表 10-101 当预分频器的参数从1变到2时，计数器的时序图



图表 10-102 当预分频器的参数从1变到4时，计数器的时序图



### 10.5.3.2 计数器模式

#### 向上计数模式

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

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

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

设置 TMRx\_CTRL1 寄存器中的 UEVDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UEVDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，

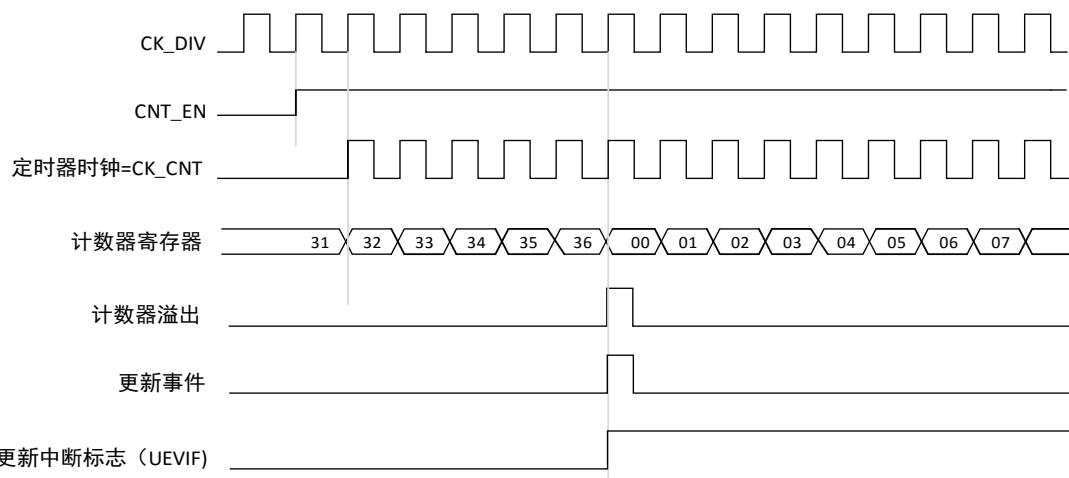
计数器仍会被清'0'，同时预分频器的计数也被清 0（但预分频器的数值不变）。此外，如果设置了 TMRx\_CTRL1 寄存器中的 UVERS 位（选择更新请求），设置 UEVG 位将产生一个更新事件 UEV，但硬件不设置 UEVIF 标志（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

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

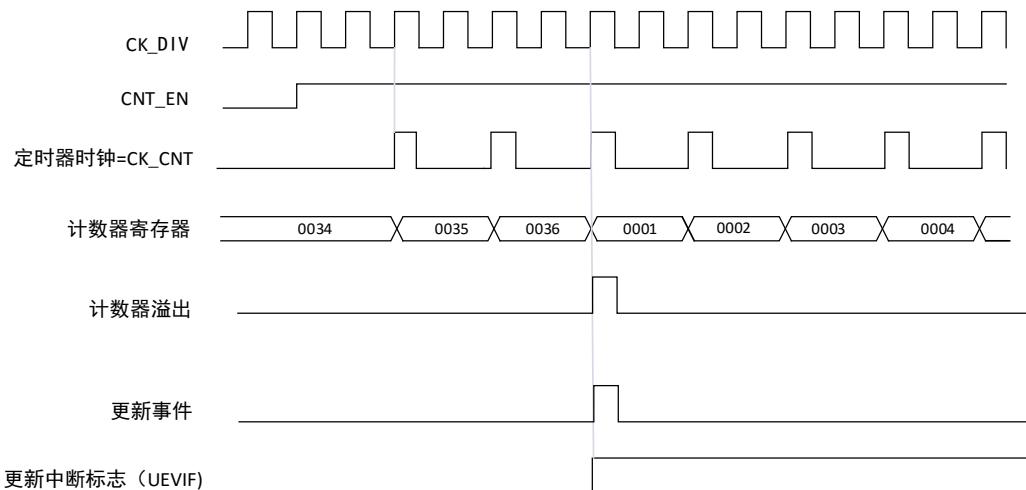
- 重复计数器被重新加载为 TMRx\_RC 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值（TMRx\_AR）。
- 预分频器的缓冲区被置入预装载寄存器的值（TMRx\_DIV 寄存器的内容）。

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

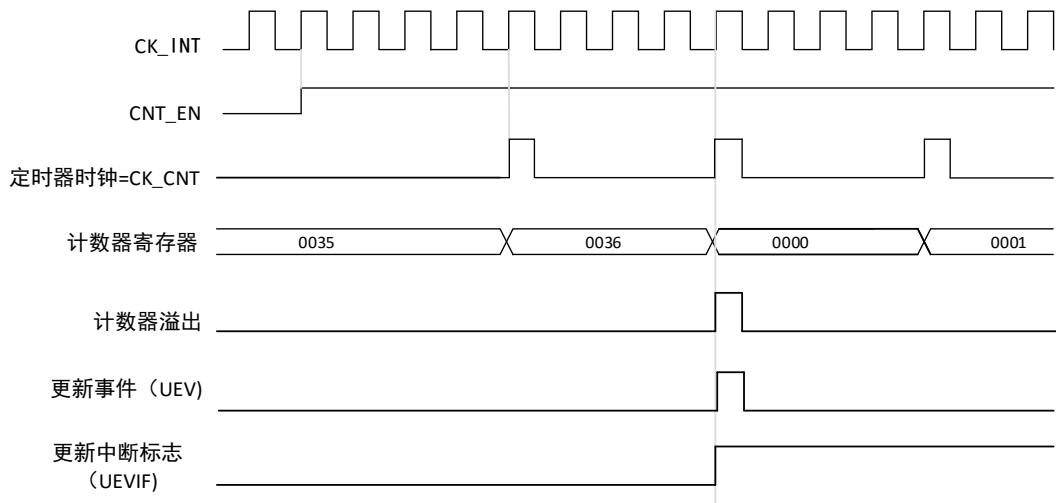
图表 10-103 计数器时序图，内部时钟分频因子为1



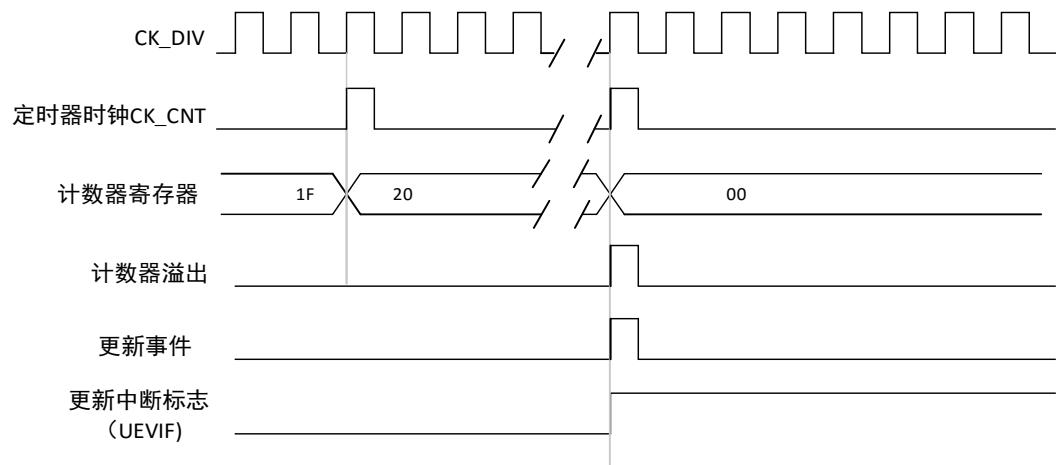
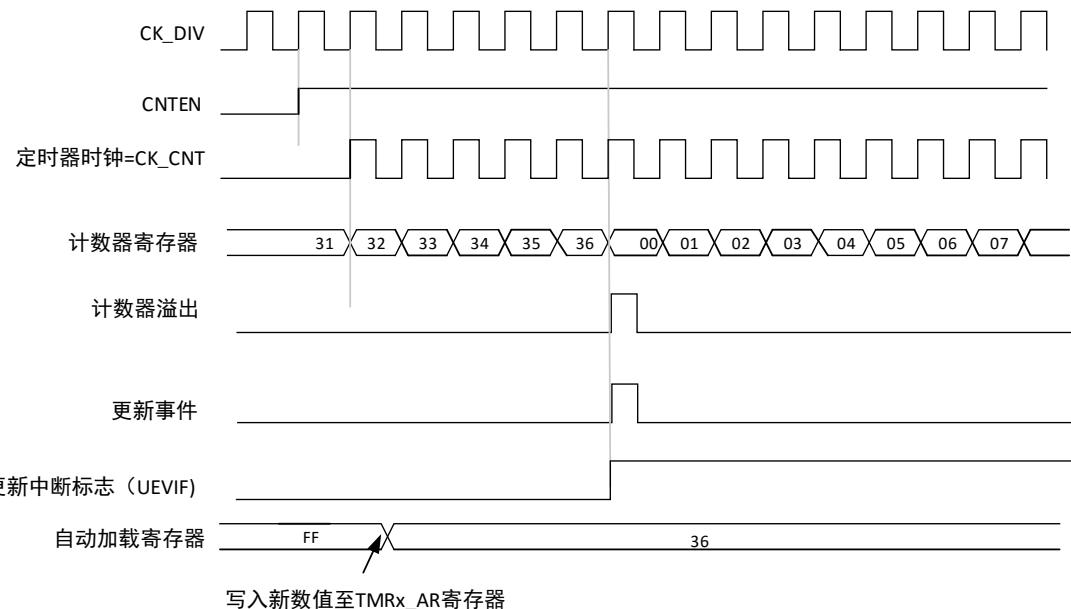
图表 10-104 计数器时序图，内部时钟分频因子为2

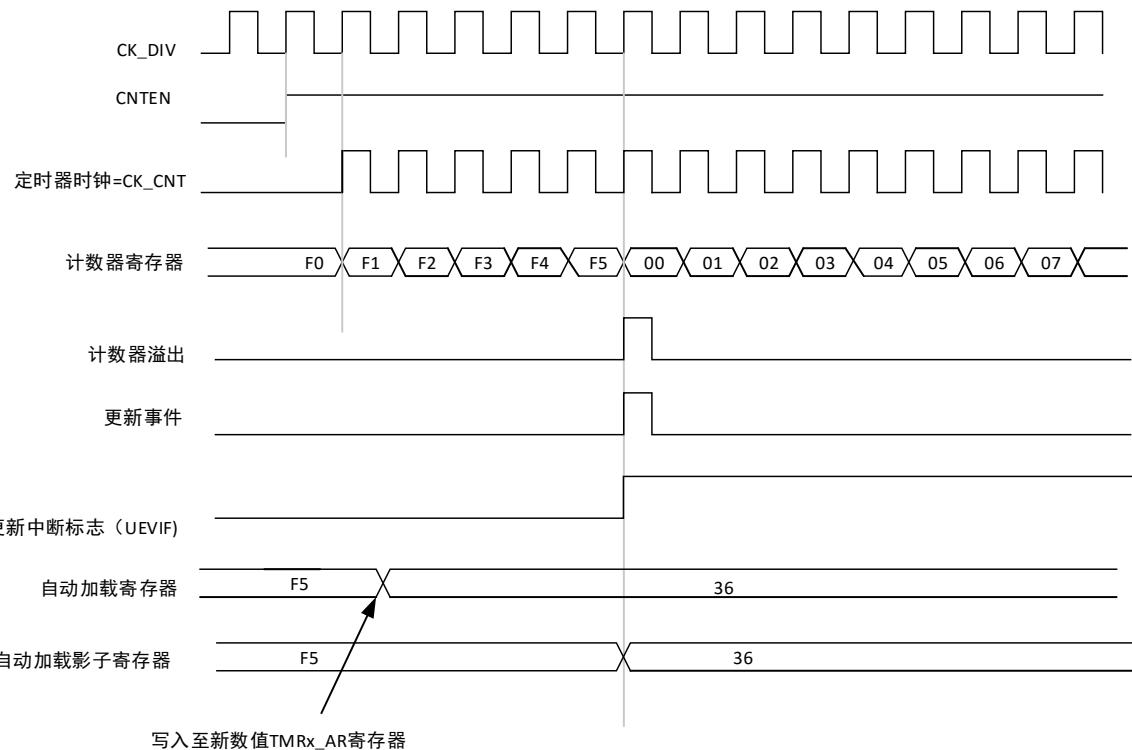


图表 10-105 计数器时序图，内部时钟分频因子为4



图表 10-106 计数器时序图，内部时钟分频因子为N

图表 10-107 计数器时序图，当 $ARPEN=0$ 时的更新事件 ( $TMRx\_AR$ 没有预装入)图表 10-108 计数器时序图，当 $ARPEN=1$ 时的更新事件 (预装入了 $TMRx\_AR$ )



### 10.5.3.3 重复计数器

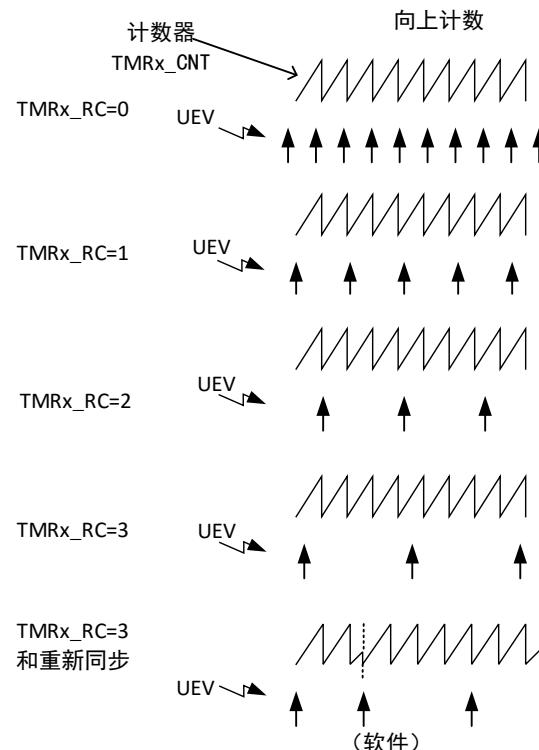
10.5.3.1 “时基单元”解释了计数器上溢时更新事件是如何产生的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢时，数据从预装载寄存器传输到影子寄存器 (TMRx\_AR 自动重载入寄存器，TMRx\_DIV 预装载寄存器，还有在比较模式下的捕获/比较寄存 (TMRx\_CCx)，N 是 TMRx\_RC 重复计数寄存器中的值。

- 重复计数器在向上计数模式下每次计数器溢出时时递减

重复计数器是自动加载的，重复速率是由 TMRx\_RC 寄存器的值定义（参看图 10-109）。当更新事件由软件产生（通过设置 TMRx\_EVEG 中的 UEVG 位）或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TMRx\_RC 寄存器中的内容被重载入到重复计数器。

图表 10-109 不同模式下更新速率的例子，及 TMRx\_RC 的寄存器设置



UEV  更新事件：传送预装载寄存器至实际寄存器并产生更新中断

#### 10.5.3.4 时钟选择

计数器时钟可由下列时钟源提供：

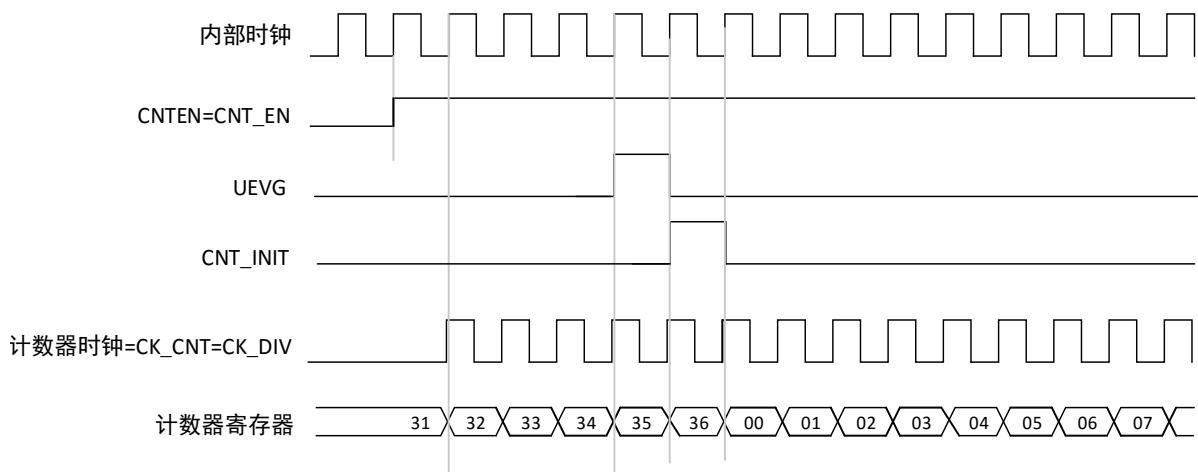
- ### ● 内部时钟 (CK\_INT)

### 内部时钟源（CK\_INT）

如果禁止了从模式控制器 (**SMSEL=000**)，则 **CNTEN**、**DIR** (**TMRx\_CTRL1** 寄存器) 和 **UEVG** 位 (**TMRx\_EVEG** 寄存器) 是实际的控制位，并且只能被软件修改 (**UEVG** 位仍被自动清除)。只要 **CNTEN** 位被写成'1'，预分频器的时钟就由内部时钟 **CK\_INT** 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

图表 10-110 一般模式下的控制电路，内部时钟分频因子为1



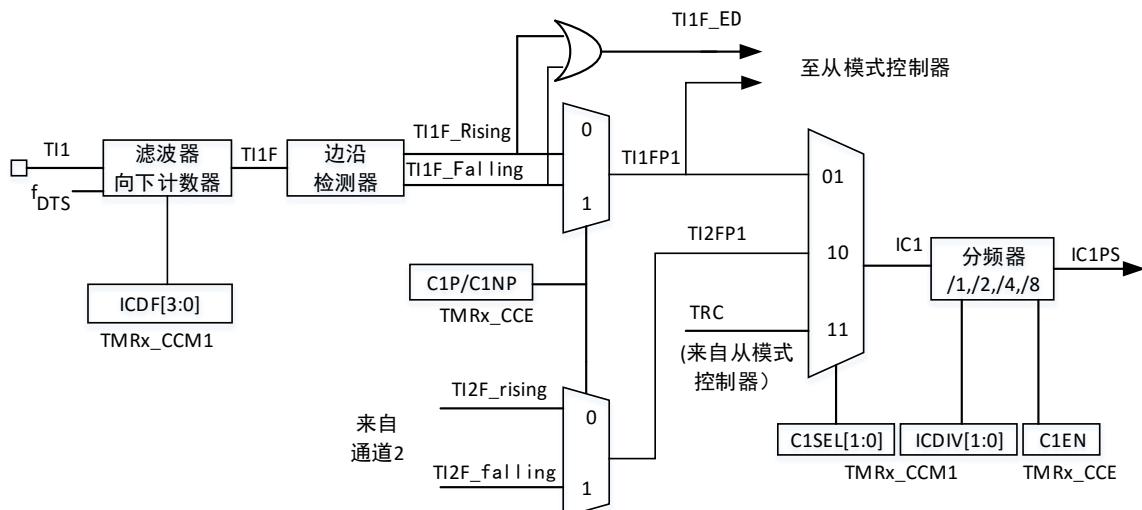
### 10.5.3.5 捕获/比较通道

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

[图 10-111 至图 10-113](#) 是一个捕获/比较通道概览。

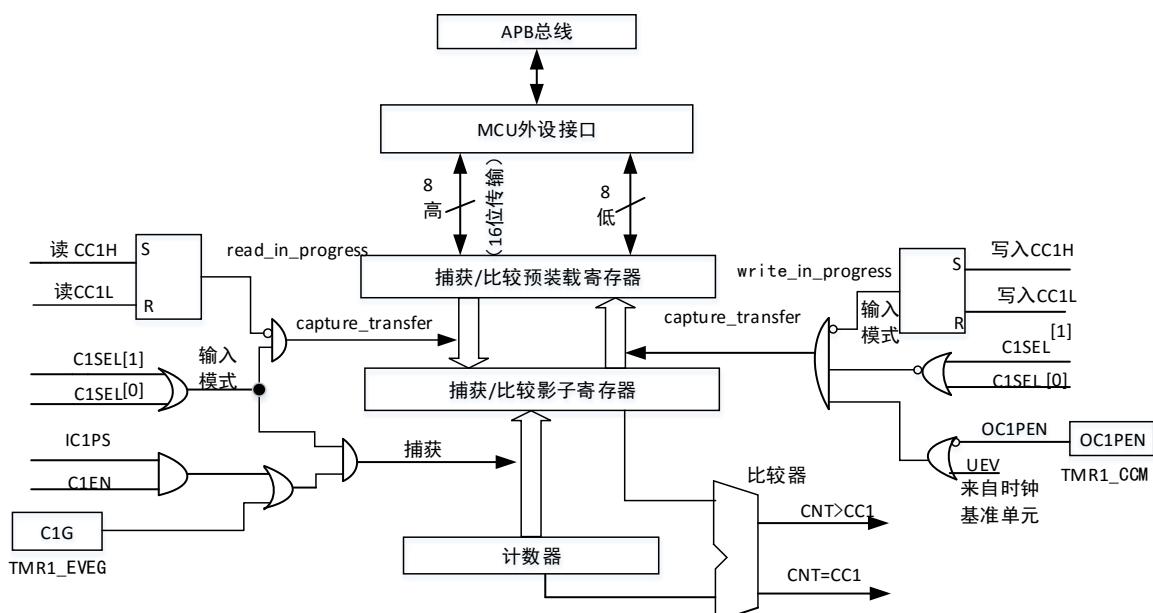
输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘检测器产生一个信号 (TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器 (ICxPS)。

图表 10-111 捕获/比较通道（如：通道 1 输入部分）

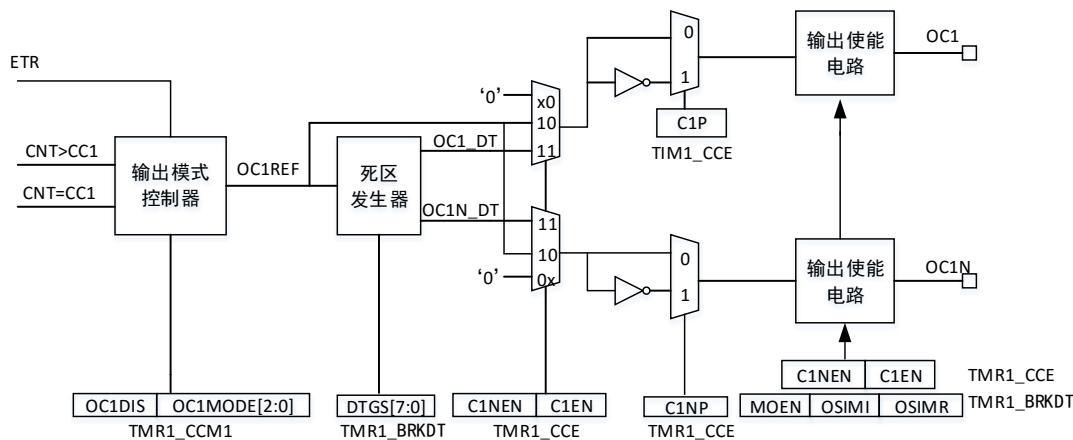


输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

图表 10-112 捕获/比较通道 1 的主电路



图表 10-113 捕获/比较通道的输出部分（通道 1）



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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

### 10.5.3.6 输入捕获模式

在输入捕获模式下，当检测到 IC<sub>x</sub> 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 (TMR<sub>x</sub>\_CC<sub>x</sub>) 中。当发生捕获事件时，相应的 CxIF 标志 (TMR<sub>x</sub>\_STS 寄存器) 被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CxIF 标志已经为高，那么重复捕获标志 CxOF (TMR<sub>x</sub>\_STS 寄存器) 被置 1。写 CxIF=0 可清除 CxIF，或读取存储在 TMR<sub>x</sub>\_CC<sub>x</sub> 寄存器中的捕获数据也可清除 CxIF。写 CxOF=0 可清除 CxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TMR<sub>x</sub>\_CC1 寄存器中，步骤如下：

- 选择有效输入端：TMR<sub>x</sub>\_CC1 必须连接到 TI1 输入，所以写入 TMR<sub>x</sub>\_CC1 寄存器中的 C1SEL=01，只要 C1SEL 不为 '00'，通道被配置为输入，并且 TMR<sub>x</sub>\_CC1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TI<sub>x</sub> 时，输入滤波器控制位是 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 IC<sub>x</sub>DF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以 f<sub>DTS</sub> 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TMR<sub>x</sub>\_CCM1 寄存器中写入 IC1DF=0011。
- 选择 TI1 通道的有效转换边沿，在 TMR<sub>x</sub>\_CCE 寄存器中写入 C1P=0（上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TMR<sub>x</sub>\_CCM1 寄存器的 IC1PS=00）。
- 设置 TMR<sub>x</sub>\_CCE 寄存器的 C1EN=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1IE 位允许相关中断请求，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传递到 TMR<sub>x</sub>\_CC1 寄存器。
- C1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 C1IF 未曾被清除，C1OF 也被置 1。
- 如设置了 C1IE 位，则会产生一个中断。
- 如设置了 C1DE 位，则还会产生一个 DMA 请求。

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

注意：设置  $TMRx\_EVEG$  寄存器中相应的  $CxG$  位，可以通过软件产生输入捕获中断和/或 DMA 请求。

### 10.5.3.7 强置输出模式

在输出模式 ( $TMRx\_CCMx$  寄存器中  $CxSEL=00$ ) 下，输出比较信号 ( $OCxREF$  和相应的  $OCx/OCxN$ ) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置  $TMRx\_CCMx$  寄存器中相应的  $OCxMODE=101$ ，即可强置输出比较信号 ( $OCxREF/OCx$ ) 为有效状态。这样  $OCxREF$  被强置为高电平 ( $OCxREF$  始终为高电平有效)，同时  $OCx$  得到  $CxP$  极性相反的信号。

例如： $CxP=0$  ( $OCx$  高电平有效)，则  $OCx$  被强置为高电平。

置  $TMRx\_CCMx$  寄存器中的  $OCxMODE=100$ ，可强置  $OCxREF$  信号为低。

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

### 10.5.3.8 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式 ( $TMRx\_CCMx$  寄存器中的  $OCxMODE$  位) 和输出极性 ( $TMRx\_CCE$  寄存器中的  $CxP$  位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 ( $OCxMODE=000$ )、被设置成有效电平 ( $OCxMODE=001$ )、被设置成无效电平 ( $OCxMODE=010$ ) 或进行翻转 ( $OCxMODE=011$ )。
- 设置中断状态寄存器中的标志位 ( $TMRx\_STS$  寄存器中的  $CxIF$  位)。
- 若设置了相应的中断屏蔽 ( $TMRx\_DIE$  寄存器中的  $CxIE$  位)，则产生一个中断。
- 若设置了相应的使能位 ( $TMRx\_DIE$  寄存器中的  $CxDE$  位， $TMRx\_CTRL2$  寄存器中的  $CDSEL$  位选择 DMA 请求功能)，则产生一个 DMA 请求。

$TMRx\_CCMx$  中的  $OCxPEN$  位选择  $TMRx\_CCx$  寄存器是否需要使用预装载寄存器。

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

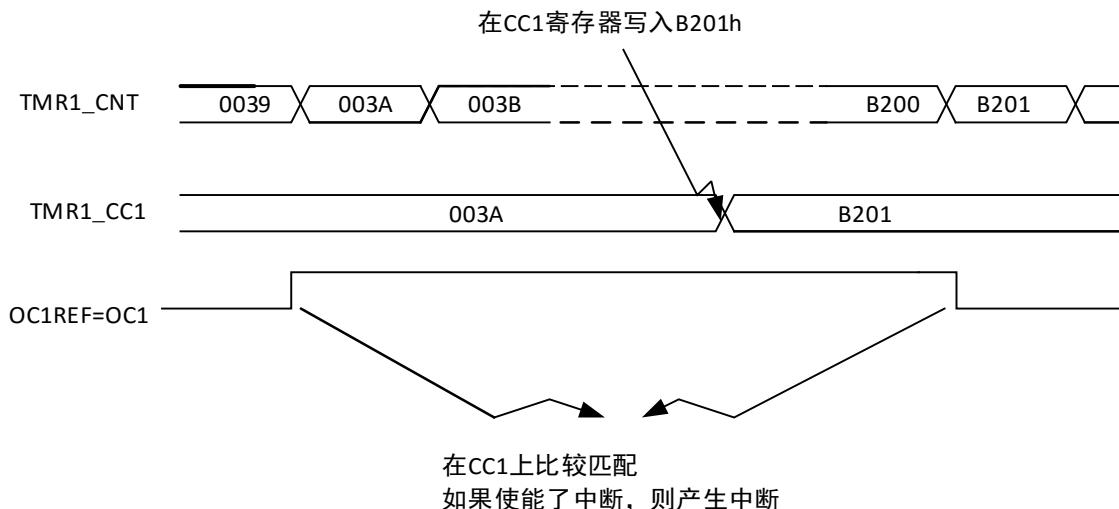
同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟 (内部，外部，预分频器)。
2. 将相应的数据写入  $TMRx\_AR$  和  $TMRx\_CCx$  寄存器中。
3. 如果要产生一个中断请求，设置  $CCxIE$  位。
4. 选择输出模式，例如：
  - 要求计数器与  $CCx$  匹配时翻转  $OCx$  的输出引脚，设置  $OCxMODE=011$
  - 置  $OCxPEN = 0$  禁用预装载寄存器
  - 置  $CxP = 0$  选择极性为高电平有效
  - 置  $CxEN = 1$  使能输出
5. 设置  $TMRx\_CTRL1$  寄存器的  $CNTEN$  位启动计数器

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

图表 10-114 输出比较模式，翻转 OC1



### 10.5.3.9 PWM模式

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

在 TMRx\_CCMx 寄存器中的 OCxMODE 位写入'110'（PWM 模式 1）或'111'（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TMRx\_CCMx 寄存器的 OCxPEN 位使能相应的预装载寄存器，最后还要设置 TMRx\_CTRL1 寄存器的 ARPEN 位，（在向上计数或中心对称模式中）使能自动重装载的预装载寄存器。

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

OCx 的极性可以通过软件在 TMRx\_CCE 寄存器中的 CxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TMRx\_CCE 和 TMRx\_BRKDT 寄存器中）CxEN、CxNEN、MOEN、OSIMI 和 OSIMR 位的组合控制。详见 TMRx\_CCE 寄存器的描述。

在 PWM 模式（模式 1 或模式 2）下，TMRx\_CNT 和 TMRx\_CCx 始终在进行比较，（依据计数器的计数方向）以确定是否符合  $TMRx\_CCx \leq TMRx\_CNT$  或者  $TMRx\_CNT \leq TMRx\_CCx$ 。

根据 TMRx\_CTRL1 寄存器中 CMSEL 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

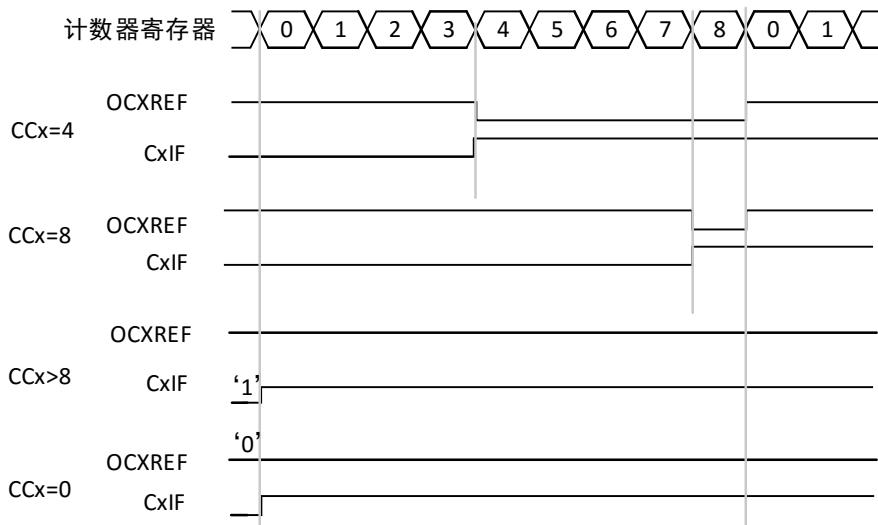
#### PWM 边沿对齐模式

- 向上计数配置

当 TMRx\_CTRL1 寄存器中的 DIR 位为低的时候执行向上计数。参看 10.5.3.2 节。

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

图表 10-115 边沿对齐的 PWM 波形（AR=8）



### 10.5.3.10 互补输出和死区插入

通用定时器（TMR15）能够输出一路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

配置 TMRx\_CCE 寄存器中的 CxP 和 CxNP 位，可以为每一个输出独立地选择极性（主输出 OCx 或互补输出 OCxN）。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TMRx\_CCE 寄存器的 CxEN 和 CxNEN 位，TMRx\_BRKDT 和 TMRx\_CTRL2 寄存器中的 MOEN、OCxIS、OCxNIS、OSIMI 和 OSIMR 位，详见表 10-13 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。特别的是，在转换到 IDLE 状态时（MOEN 下降到 0）死区被激活。

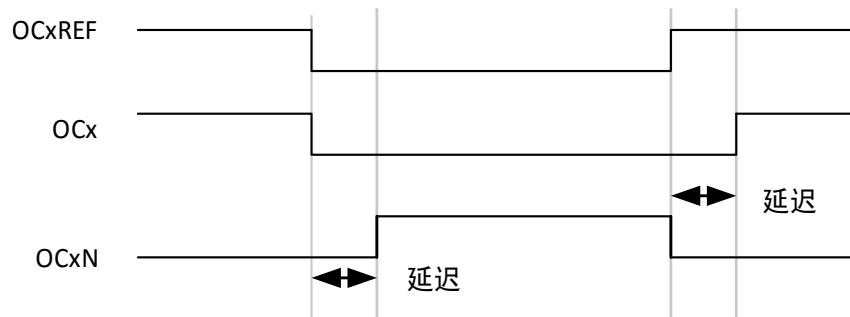
同时设置 CxEN 和 CxNEN 位将插入死区，如果存在刹车电路，则还要设置 MOEN 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

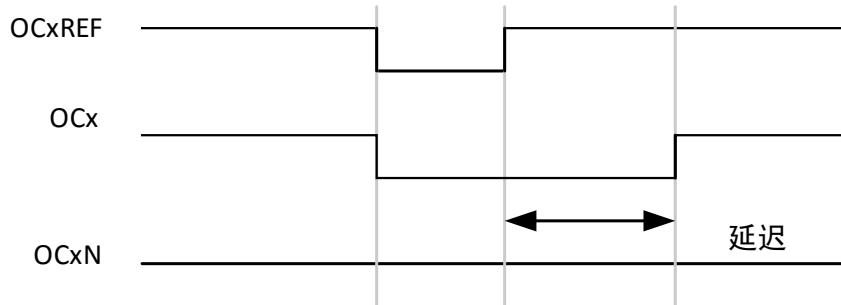
如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。

下列几张图显示了死区发生器的 出信号和当前参考信号 OCxREF 之间的关系。（假设 CxP=0、CxNP=0、MOEN=1、CxEN=1 并且 CxNEN=1）

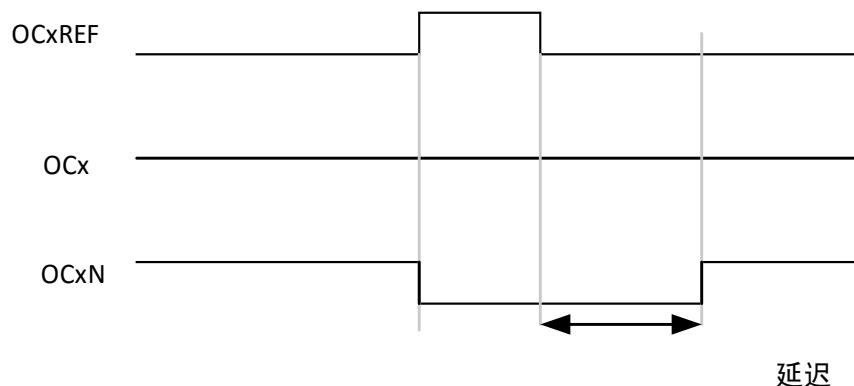
图表 10-116 带死区插入的互补输出



图表 10-117 死区波形延迟大于负脉冲



图表 10-118 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TMRx\_BRKDT 寄存器中的 DTGS 位编程配置。详见 10.5.4.15 节 TMR15 刹车和死区寄存器 (TMRx\_BRKDT) 中的延时计算。

#### 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置、输出比较或 PWM)，通过配置 TMRx\_CCE 寄存器的 CxEN 和 CxNEN 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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

#### 10.5.3.11 使用刹车功能

当使用刹车功能时，依据相应的控制位 (TMRx\_BRKDT 寄存器中的 MOEN、OSIMI 和 OSIMR 位，TMRx\_CTRL2 寄存器中的 OCxIS 和 OCxNIS 位)，输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见表 10-13 带刹车功能的互补输出通道 OCx

和 OCxN 的控制位。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生，[详见 3.2.7 节时钟失效检测（CFD）](#)。

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

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

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOEN 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSIMI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOEN=0，每一个输出通道输出由 TMRx\_CTRL2 寄存器中的 OCxIS 位设定的电平。如果 OSIMI=0，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OCxIS 和 OCxNIS 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。
- 注意，因为重新同步 MOEN，死区时间比通常情况下长一些（大约 2 个 ck\_tim 的时钟周期）。
- 如果 OSIMI=0，定时器释放使能输出，否则保持使能输出；或一旦 CxEN 与 CxNEN 之一变高时，使能输出变为高。
- 如果设置了 TMRx\_DIE 寄存器中的 BRKIE 位，当刹车状态标志（TMRx\_STS 寄存器中的 BRKIF 位）为'1'时，则产生一个中断。如果设置了 TMRx\_DIE 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 如果设置了 TMRx\_BRKDT 寄存器中的 AOEN 位，在下一个更新事件 UEV 时 MOEN 位被自动置位；例如，这可以用来进行整形。否则，MOEN 始终保持低直到被再次置'1'；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注意：刹车输入为电平有效。所以，当刹车输入有效时，不能同时（自动地或者通过软件）设置 MOEN。同时，状态标志 BRKIF 不能被清除。

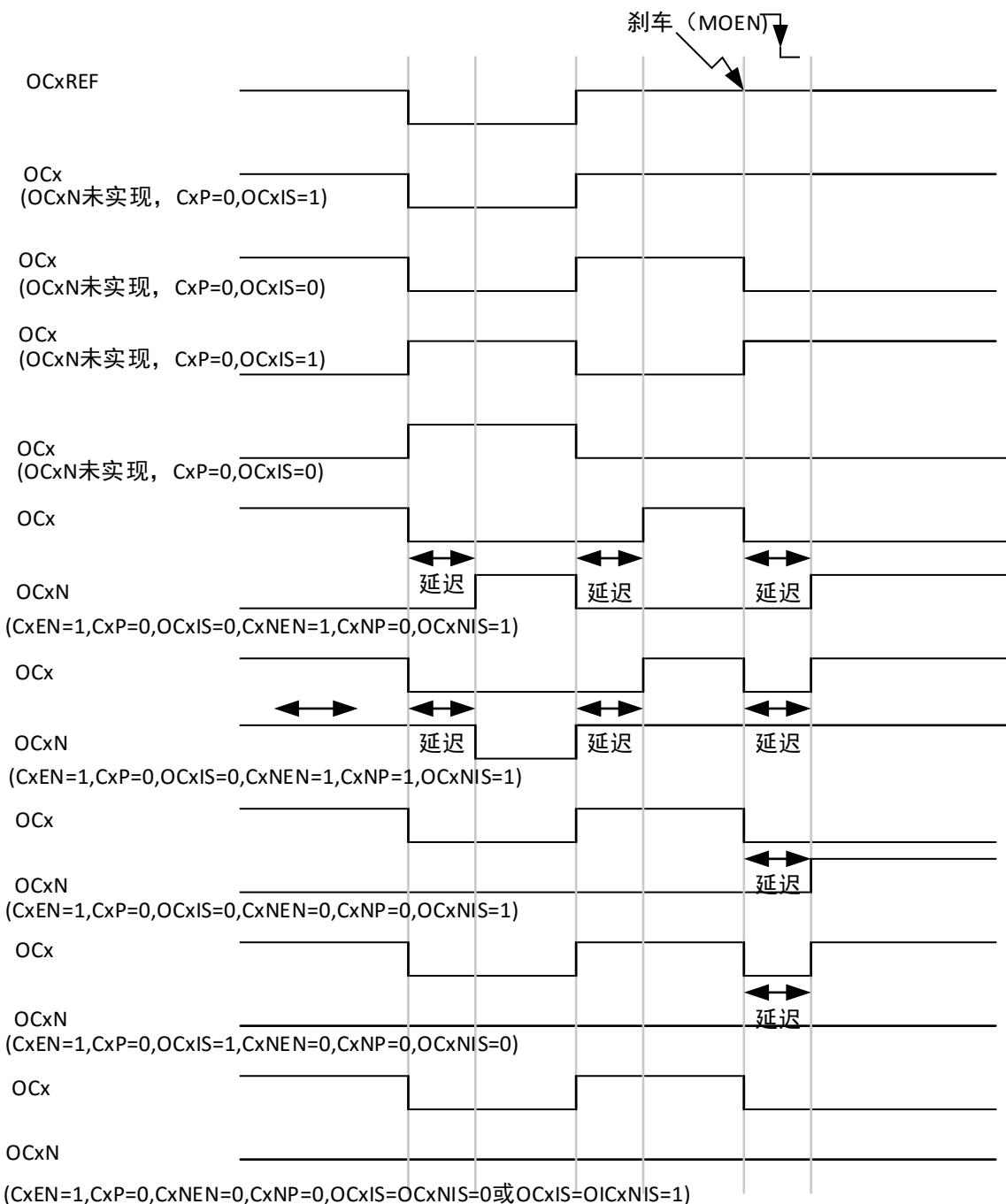
刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TMRx\_BRKDT 寄存器中的 BRKEN 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxMODE 配置，刹车使能和极性）。

用户可以通过 TMRx\_BRKDT 寄存器中的 LOCKC 位，从三级保护中选择一种，参看 [10.6.4.18 节 TMR1 刹车和死区寄存器（TMRx\\_BRKDT）](#)。在 MCU 复位后 LOCKC 位只能被修改一次。

下图显示响应刹车的输出实例。

图表 10-119 响应刹车的输出



### 10.5.3.12 单脉冲模式

单脉冲模式（OPMODE）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TMRx\_CTRL1 寄存器中的 OPMODE 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

- 向上计数方式：计数器  $CNT < CCx \leq AR$  （特别地， $0 < CCx$ ）；
- 向下计数方式：计数器  $CNT > CCx$ 。

特殊情况：OCx 快速使能：

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

如果要以最小延时输出波形，可以设置  $TMRx\_CCMx$  寄存器中的 OCxFEN 位；此时 OCxREF(和 OCx) 直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFEN 只在通道配置为 PWM1 和 PWM2 模式时起作用。

### 10.5.3.13 调试模式

当微控制器进入调试模式时（Cortex™-M4 核心停止），根据 DBG 模块中  $DBG\_TMRx\_STOP$  的设置， $TMRx$  计数器可以或者继续正常操作，或者停止。详见[第 18.2.2 节](#)。

### 10.5.4 TMRx 寄存器描述

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

下表中将  $TMR16/17$  的所有寄存器映射到一个 16 位可寻址（编址）空间

表格 10-11 TMR16/17 寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
0x00	TMRx_CTRL1	保留												CLKDIV[1: 0]		ARPEN		保留		保留		OPMODE		UVERS		UEVDIS		CNTEN									
		复位值		保留												0 0 0 0		0 0		0 0		0 0		0 0		0 0		0 0									
0x04	TMRx_CTRL2	保留												OC1NIS		OC1IS		保留		保留		CDSEL		CUSEL		保留		0 0		0 0							
		复位值		保留												0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0							
0x0C	TMRx_DIE	保留												C1DE		BRKIE		保留		保留		保留		保留		保留		保留		保留		保留					
		复位值		保留												0 0 0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0					
0x10	TMRx_STS	保留												C1OF		BRKIF		保留		保留		保留		保留		保留		保留		保留		保留					
		复位值		保留												0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0					
0x14	TMRx_EVEG	保留												BRKG		HALLG		保留		保留		保留		保留		保留		保留		保留		保留					
		复位值		保留												0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0		0 0					
0x18	TMRx_CCM1 输出 比较模式	保留												OC1MODE[2 : 0]		OC1PEN		OC1FEN		C1G		C1IF		C1IE		C1EVIE		C1EVIF		UEVG		UEVIF		CPC		CPC	
		复位值		保留												0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0		0 0 0 0			

	TMRx_CCM1 输入 捕获模式	保留										[C1DF[3: 0]				[C1DIV[1: 0]				[C1SEL[1: 0]			
	复位值											0	0	0	0	0	0	0	0	0	0	0	0
0x20	TMRx_CCE	保留										C1NP	C1NEN	C1P	C1EN	0	0	0	0	0	0	0	0
	复位值																						
0x24	TMRx_CNT	保留				CNT[15: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	TMRx_DIV	保留				DIV[15: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	TMRx_AR	保留				AR[15: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	TMRx_RC	保留				RC[7: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	TMRx_CC1	保留				CC1[15: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	TMRx_BRKDT	保留				MOEN	AOEN	BRKP	BRKEN	OSIMR	OSIMI	LOCKC[1: 0]		DTGS[7: 0]									
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	TMRx_DMAC	保留				DBLEN[4: 0]										保留	ADDR[4: 0]						
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	TMRx_DMABA	保留				DMABA[15: 0]																	
	复位值					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 10.5.4.1 TMRx控制寄存器1 (TMRx\_CTRL1) (x=16, 17)

偏移地址: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
保留						CLKDIV[1: 0]	ARP EN	保留		保留		OPMODE	UVERS	UEVDIS	CNTEN	res	RW						

位 15: 10	保留, 始终读为 0。
位 9: 8	<b>CLKDIV[1: 0]:</b> 时钟分频因子 (Clock division) 这 2 位定义在定时器时钟 (CK_INT) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: 保留, 不要使用这个配置

位 7	<b>ARPEN:</b> 自动重装载预装载允许位 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲; 1: TMRx_AR 寄存器被装入缓冲器。
位 6: 4	保留
位 3	<b>OPMODE:</b> 单脉冲模式 (One pulse mode) 0: 在发生更新事件时, 计数器不停止; 1: 在发生下一次更新事件 (清除 CNTEN 位) 时, 计数器停止。
位 2	<b>UVERS:</b> 更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UEVG 位</li> <li>- 从模式控制器产生的更新</li> </ul> 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UEVG 位</li> <li>- 从模式控制器产生的更新具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器)</li> </ul> 1: 禁止 UEV。不产生更新事件, 影子寄存器 (AR、DIV、CCx) 保持它们的值。如果设置了 UEVG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化。
位 0	<b>CNTEN:</b> 使能计数器 (Counter enable) 0: 禁止计数器; 1: 使能计数器。 <b>注:</b> 在软件设置了 CNTEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CNTEN 位。

#### 10.5.4.2 TMRx控制寄存器2 (TMRx\_CTRL2) (x=16, 17)

偏移地址: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						保留	OC1 NIS	OC1 IS		保留	CDS EL	CUS EL	保留	CPC	
						res	rw	rw		res	rw	rw	res	rw	

位 15: 10	保留, 始终读为 0。
位 9	<b>OC1NIS:</b> 输出空闲状态 1 (OC1N 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 死区后 OC1N=0; 1: 当 MOEN=0 时, 死区后 OC1N=1。 注: 已经设置了LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。
位 8	<b>OC1IS:</b> 输出空闲状态 1 (OC1 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=0。 1: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=1。 注: 已经设置了LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。
位 7: 4	保留

位 3	<b>CDSEL:</b> 捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0: 当发生 CCx 事件时, 送出 CCx 的 DMA 请求; 1: 当发生更新事件时, 送出 CCx 的 DMA 请求。
位 2	<b>CUSEL:</b> 捕获/比较控制更新选择 (Capture/compare control update selection) 0: 如果捕获/比较控制位是预装载的 (CPC=1), 只能通过设置 HALL 位更新它们; 1: 如果捕获/比较控制位是预装载的 (CPC=1), 可以通过设置 HALL 位或 TRGI 上的一个上升沿更新它们。 注: 该位只对具有互补输出的通道起作用。
位 1	保留, 始终读为 0。
位 0	<b>CPC:</b> 捕获/比较预装载控制位 (Capture/compare preloaded control) 0: CxEN, CxNEN 和 OCxMODE 位不是预装载的; 1: CxEN, CxNEN 和 OCxMODE 位是预装载的; 设置该位后, 它们只在发生一个 HALL 事件的时候 (设置了 HALL 位或检测到 TRGI 的上升沿, 依据 CUSEL 位) 被更新。 注: 该位只对具有互补输出的通道起作用。

#### 10.5.4.3 TMRx DMA/中断使能寄存器 (TMRx\_DIE) (x=16, 17)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		C1DE	UEVDE	BRKIE	保留	HALLIE	保留	保留	保留	C1IE	UEVIE				
res		rw	rw	rw	rw	rw	res	res	res	rw	rw				

位 15: 10	保留, 始终读为 0。
位 9	<b>C1DE:</b> 允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求; 1: 允许捕获/比较 1 的 DMA 请求。
位 8	<b>UEVDE:</b> 允许更新的 DMA 请求 (Update DMA request enable) 0: 禁止更新的 DMA 请求; 1: 允许更新的 DMA 请求。
位 7	<b>BRKIE:</b> 允许刹车中断 (Break interrupt enable) 0: 禁止刹车中断; 1: 允许刹车中断。
位 6	保留
位 5	<b>HALLIE:</b> 允许 HALL 中断 (HALL interrupt enable) 0: 禁止 HALL 中断; 1: 允许 HALL 中断。
位 4: 2	保留
位 1	<b>C1IE:</b> 允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断; 1: 允许捕获/比较 1 中断。
位 0	<b>UEVIE:</b> 允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

#### 10.5.4.4 TMRx 状态寄存器 (TMRx\_STS) (x=16, 17)

偏移地址: 0x10

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					C1OF	保留	BRKIF	保留	HALLIF	保留	保留	保留	C1IF	UEVIF	
res					rw	res	rw	res	rw	res	res	res	rw	rw	

位 15: 10	保留, 始终读为 0。
位 9	<b>C1OF:</b> 捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 计数器的值被捕获到 TMRx_CC1 寄存器时, C1IF 的状态已经为'1'。
位 8	保留, 始终读为 0。
位 7	<b>BRKIF:</b> 刹车中断标记 (Break interrupt flag) 一旦刹车输入有效, 由硬件对该位置'1'。如果刹车输入无效, 则该位可由软件清'0'。 0: 无刹车事件产生; 1: 刹车输入上检测到有效电平。
位 6	保留
位 5	<b>HALLIF:</b> HALL 中断标记 (HALL interrupt flag) 一旦产生 HALL 事件 (当捕获/比较控制位: CxEN、CxNEN、OCxMODE 已被更新) 该位由硬件置'1'。它由软件清'0'。 0: 无 HALL 事件产生; 1: HALL 中断等待响应。
位 4: 2	保留
位 1	<b>C1IF:</b> 捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) <b>如果通道 CC1 配置为输出模式:</b> 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外 (参考 TMRx_CTRL1 寄存器的 CMSEL 位)。它由软件清'0'。 0: 无匹配发生; 1: TMRx_CNT 的值与 TMRx_CC1 的值匹配。 当 TMRx_CC1 的内容大于 TMRx_AR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, C1IF 位变高。 <b>如果通道 CC1 配置为输入模式:</b> 当捕获事件发生时该位由硬件置'1', 它由软件清'0'或通过读 TMRx_CC1 清'0'。 0: 无输入捕获产生; 1: 计数器值已被捕获 (拷贝) 至 TMRx_CC1 (在 IC1 上检测到与所选极性相同的边沿)。
位 0	<b>UEVIF:</b> 更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': - 若 TMRx_CTRL1 寄存器的 UEVDIS=0, 当重复计数器数值上溢或下溢时 (重复计数器=0 时产生更新事件)。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当设置 TMRx_EVEG 寄存器的 UEVG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当计数器 CNT 被触发事件重新初始化时。 (参考 <a href="#">10.6.4.3: TMR1 从模式控制寄存器 (TMRx_SMC)</a> )。

#### 10.5.4.5 TMRx事件产生寄存器 (TMRx\_EVEG) (x=16, 17)

偏移地址: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					保留		BRKG	保留	HALLG	保留	保留	保留	C1G	UEVG	

	res	rw	res	rw	res	res	res	rw	rw
位 15: 8	保留, 始终读为 0。								
位 7	<b>BRKG:</b> 产生刹车事件 (Break generation) 该位由软件置'1', 用于产生一个刹车事件, 由硬件自动清'0'。 0: 无动作; 1: 产生一个刹车事件。此时 MOEN=0、BRKIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。								
位 6	保留								
位 5	<b>HALLG:</b> 捕获/比较事件, 产生控制更新 (Capture/Compare control update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 当 CPC=1, 允许更新 CxEN、CxNEN、OCxMODE 位。 注: 该位只对拥有互补输出的通道有效。								
位 4: 2	保留								
位 1	<b>C1G:</b> 产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1', 用于产生一个捕获/比较事件, 由硬件自动清'0'。 0: 无动作; 1: 在通道 CC1 上产生一个捕获/比较事件: 若通道 CC1 配置为输出: 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。 若通道 CC1 配置为输入: 当前的计数器值被捕获至 TMRx_CC1 寄存器; 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 C1IF 已经为 1, 则设置 C1OF=1。								
位 0	<b>UEVG:</b> 产生更新事件 (Update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0 (向上计数) 则计数器被清'0'; 若 DIR=1 (向下计数) 则取计数器取 TMRx_AR 的值。								

#### 10.5.4.6 TMRx捕获/比较模式寄存器1 (TMRx\_CCM1) (x=16, 17)

偏移地址: 0x18

复位值: 0x0000

通道可用于输入 (捕获模式) 或输出 (比较模式), 通道的方向由相应的 CxSEL 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能, ICxx 描述了通道在输入模式下的功能。因此必须注意, 同一个位在输出模式和输入模式下的功能是不同的。

##### 输出比较模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					OC1MODE[2: 0]			OC1 PEN	OC1 FEN	C1SEL[1: 0]					
res					rw		rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 7	保留														

位 6: 4	<p><b>OC1MODE[2: 0]:</b> 输出比较 1 模式 (Output Compare 1 mode)          该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。          OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 C1P、C1NP 位。          000: 冻结。输出比较寄存器 TMRx_CC1 与计数器 TMRx_CNT 间的比较对 OC1REF 不起作用；          001: 匹配时设置通道 1 为有效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为高。          010: 匹配时设置通道 1 为无效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为低。          011: 翻转。当 TMRx_CC1=TMRx_CNT 时，翻转 OC1REF 的电平。          100: 强制为无效电平。强制 OC1REF 为低。          101: 强制为有效电平。强制 OC1REF 为高。          110: PWM 模式 1— 在向上计数时，一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为无效电平 (OC1REF=0)，否则为有效电平 (OC1REF=1)。          111: PWM 模式 2— 在向上计数时，一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为有效电平，否则为无效电平。          注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。          注 2: 在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。</p>
位 3	<p><b>OC1PEN:</b> 输出比较 1 预装载使能 (Output Compare 1 preload enable)          0: 禁止 TMRx_CC1 寄存器的预装载功能，可随时写入 TMRx_CC1 寄存器，并且新写入的数值立即起作用。          1: 开启 TMRx_CC1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TMRx_CC1 的预装载值在更新事件到来时被加载至当前寄存器中。          注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。          注 2: 仅在单脉冲模式下 (TMRx_CTRL1 寄存器的 OPMODE=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。</p>
位 2	<p><b>OC1FEN:</b> 输出比较 1 快速使能 (Output Compare 1 fast enable)          该位用于加快 CC 输出对触发输入事件的响应。          0: 根据计数器与 CC1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。          1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。          OC1FEN 只在通道被配置成 PWM1 或 PWM2 模式时起作用。</p>
位 1: 0	<p><b>C1SEL[1: 0]:</b> 捕获/比较 1 选择。 (Capture/Compare 1 selection)          这 2 位定义通道的方向 (输入/输出)，及输入脚的选择：          00: CC1 通道被配置为输出；          01: CC1 通道被配置为输入，IC1 映射在 TI1 上；          10: CC1 通道被配置为输入，IC1 映射在 TI2 上；          11: CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。          注：C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=0) 才是可写的。</p>

### 输入捕获模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	IC1DF[3: 0]	IC1DIV[1: 0]	C1SEL[1: 0]												
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		
位 15: 8	保留														

位 7: 4	<b>ETDF[3: 0]:</b> 外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000: 无滤波器，以 $f_{DTS}$ 采样      1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6 0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2      1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8 0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4      1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5 0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8      1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6      1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8      1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6      1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8      1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8

#### 10.5.4.7 TMRx捕获/比较使能寄存器 (TMRx\_CCE) (x=16, 17)

偏移地址: 0x20

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										C1 NP	C1 NEN	C1P	C1 EN		
res										rw	rw	rw	rw		

位 15: 4	保留, 始终读为 0。
位 3	<b>C1NP:</b> 输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) <b>CC1 通道配置为输出:</b> 0: OC1N 高电平有效; 1: OC1N 低电平有效。 <b>CC1 通道配置为输入:</b> C1NP 结合 C1P 定义 TI1FP1/TI2FP1 极性 (参考 C1P 描述) 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2 且 C1SEL=00 (通道配置为输出) 则该位不能被修改。
位 2	<b>C1NEN:</b> 输入/捕获 1 互补输出使能 (Capture/Compare 1 complementary output enable) 0: 关闭 - OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。 1: 启开 - OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。

位 1	<p><b>C1P:</b> 输入/捕获 1 输出极性 (Capture/Compare 1 output polarity)  <b>CC1 通道配置为输出:</b>            0: OC1 高电平有效            1: OC1 低电平有效  <b>CC1 通道配置为输入:</b>            CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的极性信号作为触发或捕获信号。            00: 不反相/上升沿。电路 TIxFP1 上升沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 不反相 (触发中的门控模式)。            01: 反相/下降沿。电路对 TIxFP1 下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 反相 (触发中的门控模式)。            10: 保留。            11: 不反相/双沿。电路同时对 TIxFP1 上升沿和下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 是不反相 (触发中门控模式)。  <b>注:</b> 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2, 则该位不能被修改。</p>
	<p><b>C1EN:</b> 输入/捕获 1 输出使能 (Capture/Compare 1 output enable)  <b>CC1 通道配置为输出:</b>            0 : 关闭— OC1 禁止输出, 因此 OC1 的输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1NEN 位的值。            1 : 开启— OC1 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN 、 OSIMI 、 OSIMR 、 OC1IS、 OC1NIS 和 C1NEN 位的值。  <b>CC1 通道配置为输入:</b>            该位决定了计数器的值是否能捕获入 TMRx_CC1 寄存器。            0: 捕获禁止;            1: 捕获使能。</p>

表格 10-12 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态 (1)	
MOEN 位	OSIMI 位	OSIMR 位	CxEN 位	CxNEN 位	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	0	1	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		0	1	0	OCxREF+极性, OCx= OCxREF xor CxP, OCx_EN=1	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1
		1	0	0	输出禁止 (与定时器断开) OCx=CxP, OCx_EN=0	输出禁止 (与定时器断开) OCxN=CxNP, OCxN_EN=0
		1	0	1	关闭状态 (输出使能且为无效电平) OCx=CxP, OCx_EN=1	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		1	1	0	OCxREF + 极性, OCx= OCxREF xor CxP, OCx_EN=1	关闭状态 (输出使能且为无效电平) OCxN=CxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1
0	0	X	0	0	输出禁止 (与定时器断开) 异步地: OCx=CxP , OCx_EN=0 , OCxN=CxNP , OCxN_EN=0;	
	0		0	1		

	0		1	0	若时钟存在：经过一个死区时间后 OCx=OC1IS， OCxN=OCxNIS，假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。							
	0		1	1								
	1		0	0	关闭状态（输出使能且为无效电平） 异步地： OCx=CxP , OCx_EN=1 , OCxN=CxNP , OCxN_EN=1;							
	1		0	1								
	1		1	0	若时钟存在：经过一个死区时间后 OCx=OC1IS， OCxN=OCxNIS，假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。							
	1		1	1								

注意：如果一个通道的 2 个输出都没有使用(CxEN = CxNEN = 0)，那么 OC1IS, OCxNIS, CxP 和 CxNP 都必须清零。

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

#### 10.5.4.8 TMRx计数器 (TMRx\_CNT) (x=16, 17)

偏移地址: 0x24

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>CNT[15: 0]:</b> 计数器的值 (Counter value)														

#### 10.5.4.9 TMRx预分频器 (TMRx\_DIV) (x=16, 17)

偏移地址: 0x28

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>DIV[15: 0]:</b> 预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_DIV} / (DIV[15: 0] + 1)$ 。 DIV 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TMR_EVEG 的 UEVG 位清'0'或被工作在复位模式的从控制器清'0'。														

#### 10.5.4.10 TMRx自动重装载寄存器 (TMRx\_AR) (x=16, 17)

偏移地址: 0x2C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>AR[15: 0]:</b> 自动重装载的值 (Prescaler value) AR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 10.5.3.1 节：有关 AR 的更新和动作。当自动重装载的值为空时，计数器不工作。														

### 10.5.4.11 TMRx重复计数寄存器（TMRx\_RC）（x=16, 17）

偏移地址: 0x30

复位值: 0x0000

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

位 15: 8	保留, 始终读为 0。
位 7: 0	<p><b>RC[7: 0]:</b> 重复计数器的值（Repetition counter value）          启用了预装载功能后, 这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)。如果允许产生更新中断, 则会同时影响产生更新中断的速率。每次向下计数器 RC_CNT 达到 0, 会产生一个更新事件并且计数器 RC_CNT 重新从 RC 值开始计数。由于 RC_CNT 只有在周期更新事件 U_RC 发生时才重载 RC 值, 因此对 TMRx_RC 寄存器写入的新值只在下次周期更新事件发生时才起作用。          这意味着在 PWM 模式中, (RC+1) 对应着:          - 在边沿对齐模式下, PWM 周期的数目;          - 在中心对称模式下, PWM 半周期的数目;</p>

### 10.5.4.12 TMRx捕获/比较寄存器1（TMRx\_CC1）（x=16, 17）

偏移地址: 0x34

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC1[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 0	<p><b>CC1[15: 0]:</b> 捕获/比较通道 1 的值（Capture/Compare 1 value）          若 CC1 通道配置为输出:          CC1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。如果在 TMRx_CCM1 寄存器（OC1PEN 位）中未选择预装载功能, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC1 端口上产生输出信号。          若 CC1 通道配置为输入:          CC1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。</p>
---------	---

### 10.5.4.13 TMRx刹车和死区寄存器（TMRx\_BRKDT）（x=16, 17）

偏移地址: 0x44

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
MO EN	AO EN	BR KP	BRK EN	OSI MR	OSI MI	LOCKC[1: 0]	DTGS[7: 0]													
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw					

注意: 根据锁定设置, AOEN、BRKP、BRKEN、OSIMR 和 DTGS[7: 0]位均可被写保护, 有必要在第一次写入 TMRx\_BRKDT 寄存器时对它们进行配置。

位 15	<p><b>MOEN:</b> 主输出使能 (Main output enable)          一旦刹车输入有效，该位被硬件异步清'0'。根据 AOEN 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。          0: 禁止 OC 和 OCN 输出或强制为空闲状态；          1: 如果设置了相应的使能位 (TMRx_CCE 寄存器的 CxEN、CxNEN 位)，则开启 OC 和 OCN 输出。          有关 OC/OCN 使能的细节，参见 10.5.4.7 节：TMR1 捕获/比较使能寄存器 (TMRx_CCE)。</p>
位 14	<p><b>AOEN:</b> 自动输出使能 (Automatic output enable)          0: MOEN 只能被软件置'1'；          1: MOEN 能被软件置'1'或在下一个更新事件被自动置'1' (如果刹车输入无效)。          注：一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1'，则该位不能被修改。</p>
位 13	<p><b>BRKP:</b> 刹车输入极性 (Break polarity)          0: 刹车输入低电平有效；          1: 刹车输入高电平有效。          注：一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1'，则该位不能被修改。          注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。</p>
位 12	<p><b>BRKEN:</b> 刹车功能使能 (Break enable)          0: 禁止刹车输入 (BRK 及 CFD 时钟失效事件)；          1: 开启刹车输入 (BRK 及 CFD 时钟失效事件)。          注：一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1'，则该位不能被修改。          注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。</p>
位 11	<p><b>OSIMR:</b> 运行模式下“关闭状态”选择 (Off-state selection for Run mode)          该位用于当 MOEN=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSIMR 位。参考 OC/OCN 使能的详细说明 (10.5.4.7 节，TMR1 捕获/比较使能寄存器 (TMRx_CCE))。          0: 当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)；          1: 当定时器不工作时，一旦 CxEN=1 或 CxNEN=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。          注：一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 2，则该位不能被修改。</p>
位 10	<p><b>OSIMI:</b> 空闲模式下“关闭状态”选择 (Off-state selection for Idle mode)          该位用于当 MOEN=0 且通道设为输出时。参考 OC/OCN 使能的详细说明 (10.5.4.7 节，TMR1 捕获/比较使能寄存器 (TMRx_CCE))。          0 : 当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)；          1 : 当定时器不工作时，一旦 CxEN=1 或 CxNEN=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。          注：一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 2，则该位不能被修改。</p>
位 9: 8	<p><b>LOCKC[1: 0]:</b> 锁定设置 (LENock configuration)          该位为防止软件错误而提供写保护。          00: 锁定关闭，寄存器无写保护；          01: 锁定级别 1，不能写入 TMRx_BRKDT 寄存器的 DTGS、BRKEN、BRKP、AOEN 位和 TMRx_CTRL2 寄存器的 OCXIS/OCxNIS 位；          10: 锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位 (一旦相关通道通过 CxSEL 位设为输出，CC 极性位是 TMRx_CCE 寄存器的 CxP/CCNxP 位) 以及 OSIMR/OSIMI 位；          11: 锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位 (一旦相关通道通过 CxSEL 位设为输出，CC 控制位是 TMRx_CCMx 寄存器的 OCxMODE/OCxPEN 位)；          注：在系统复位后，只能写入一次 LOCKC 位，一旦写入 TMRx_BRKDT 寄存器，则其内容冻结直至复位。</p>

位 7: 0	<p><b>DTGS[7: 0]:</b> 死区发生器设置 (Dead-time generator setup)          这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间:  <math>DTGS[7: 5]=0xx \Rightarrow DT=DTGS[7: 0] \times Tdtg, \quad Tdtg = T_{DTS};</math>  <math>DTGS[7: 5]=10x \Rightarrow DT= (64+DTGS[5: 0]) \times Tdtg, \quad Tdtg = 2 \times T_{DTS};</math>  <math>DTGS[7: 5]=110 \Rightarrow DT= (32+DTGS[4: 0]) \times Tdtg, \quad Tdtg = 8 \times T_{DTS};</math>  <math>DTGS[7: 5]=111 \Rightarrow DT= (32+DTGS[4: 0]) \times Tdtg, \quad Tdtg = 16 \times T_{DTS};</math>          例: 若 <math>T_{DTS} = 125\text{ns}</math> (8MHZ), 可能的死区时间为:          0 到 15875ns, 若步长时间为 125ns;          16us 到 31750ns, 若步长时间为 250ns;          32us 到 63us, 若步长时间为 1us;          64us 到 126us, 若步长时间为 2us;          注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 1、2 或 3, 则不能修改这些位。</p>
--------	---

#### 10.5.4.14 TMRx DMA控制寄存器 (TMRx\_DMAR) (x=16, 17)

偏移地址: 0x48

复位值: 0x0000

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

位 15: 13	保留, 始终读为 0。
位 12: 8	<p><b>DBLEN[4: 0]:</b> DMA 连续传送长度 (DMA burst length)          这些位定义了 DMA 在连续模式下的传送长度(当对 TMRx_DMABA 寄存器进行读或写时, 定时器则 进行一次连续传送), 即: 定义传输的次数, 传输可以是半字(双字节)或字节:          00000: 1 次传输      00001: 2 次传输          00010: 3 次传输      .....          ....      ..      10001: 18 次传输          例: 我们考虑这样的传输: DBLEN=7, ADDR=TMR2_CTRL1          - 如果 DBLEN=7, ADDR=TMR2_CTRL1 表示待传输数据的地址, 那么传输的地址由下式给出: (TMRx_CTRL1 的地址) + ADDR + (DMA 索引), 其中 DMA 索引=DBLEN 其中 (TMRx_CTRL1 的地址) + ADDR 再加上 7, 给出了将要写入或者读出数据的地址, 这样数据的传输将发生在从地址 (TMRx_CTRL1 的地址) + ADDR 开始的 7 个寄存器。          根据 DMA 数据长度的设置, 可能发生以下情况:          - 如果设置数据为半字 (16 位), 那么数据就会传输给全部 7 个寄存器。          - 如果设置数据为字节, 数据仍然会传输给全部 7 个寄存器: 第一个寄存器包含第一个 MSB 字节, 第二个寄存器包含第一个 LSB 字节, 以此类推。因此对于定时器, 用户必须指定由 DMA 传输的数据宽度。</p>
位 7: 5	保留, 始终读为 0。
位 4: 0	<p><b>ADDR[4: 0]:</b> DMA 基址 (DMA base address)          这些位定义了 DMA 在连续模式下的基址 (当对 TMRx_DMABA 寄存器进行读或写时), ADDR 定义为从 TMRx_CTRL1 寄存器所在地址开始的偏移量:          00000: TMRx_CTRL1,          00001: TMRx_CTRL2,          00010: TMRx_SMC,          .....</p>

#### 10.5.4.15 TMRx 连续模式的 DMA 地址 (TMRx\_DMABA) (x=16, 17)

偏移地址: 0x4C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DMABA[15: 0]																
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
位 15: 0		<b>DMABA[15: 0]:</b> DMA 连续传送寄存器 (DMA register for burst accesses) 对 TMRx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作: TMRx_CTRL1 地址 + ADDR + DMA 索引, 其中: “TMRx_CTRL1 地址”是控制寄存器 1 (TMRx_CTRL1) 所在的地址; “ADDR”是 TMRx_DMAR 寄存器中定义的基地址; “DMA 索引”是由 DMA 自动控制的偏移量, 它取决于 TMRx_DMAR 寄存器中定义的 DBLEN。														

## 10.6 高级控制定时器 (TMR1)

### 10.6.1 TMR1简介

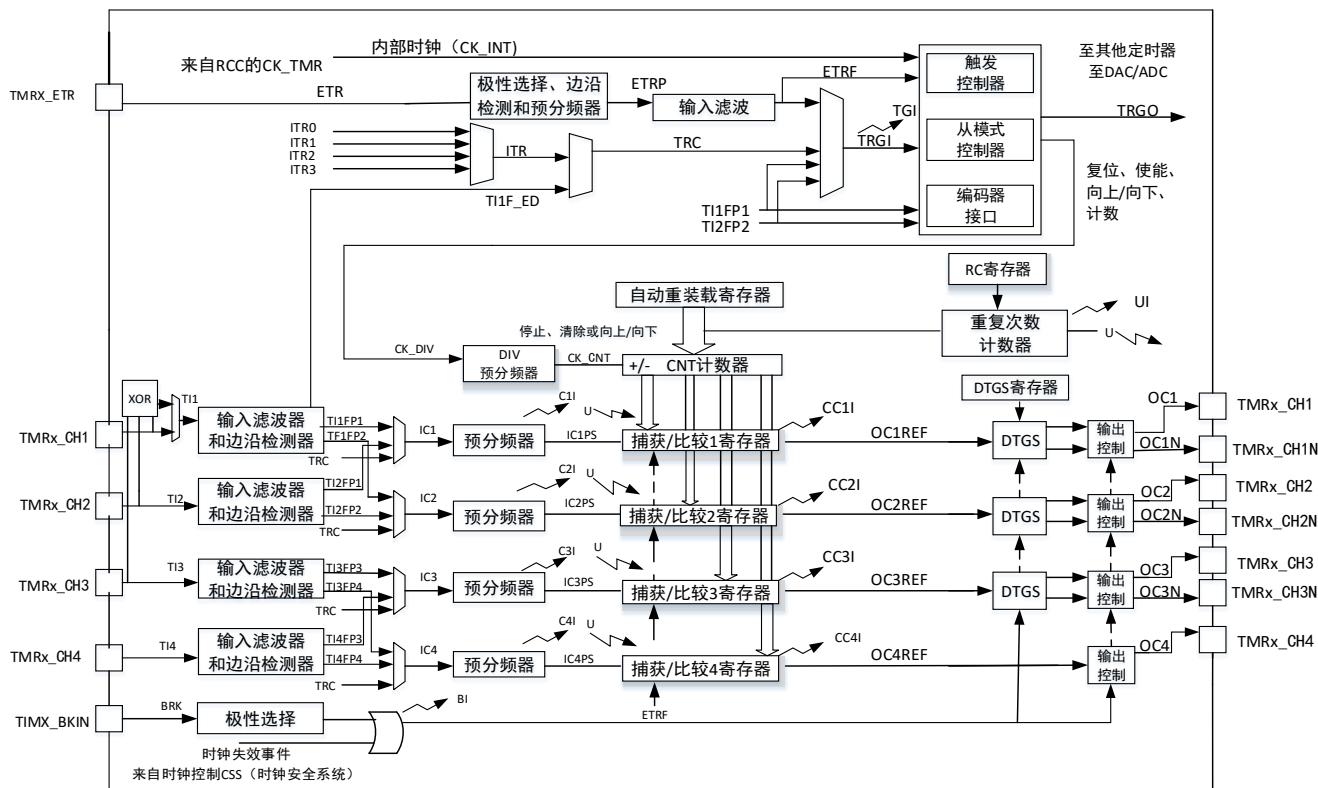
高级控制定时器 (TMR1) 由一个 16 位的自动装载计数器组成, 它由一个可编程的预分频器驱动。它适合多种用途, 包含测量输入信号的脉冲宽度 (输入捕获), 或者产生输出波形 (输出比较、PWM、嵌入死区时间的互补 PWM 等)。使用定时器预分频器和 RCC 时钟控制预分频器, 可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。高级控制定时器 (TMR1) 和通用定时器 (TMRx) 是完全独立的, 它们不共享任何资源。它们可以同步操作, 具体描述参看 [10.6.3.20 节](#)。

### 10.6.2 TMR1主要特性

TMR1 定时器的功能包括:

- 16位向上、向下、向上/下自动装载计数器
- 16位可编程 (可以实时修改) 预分频器, 计数器时钟频率的分频系数为 1~65536 之间的任意数值
- 多达 4 个独立通道:
  - 输入捕获
  - 输出比较
  - PWM 生成 (边缘或中间对齐模式)
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA:
  - 更新: 计数器向上溢出/向下溢出, 计数器初始化 (通过软件或者内部/外部触发)
  - 触发事件 (计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 支持针对定位的增量 (正交) 编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

图表 10-120 高级控制定时器框图



注意： 根据控制位的设定，在 U (更新) 事件时传送预加载寄存器的内容至工作寄存器

事件

中断和 DMA 输出

## 10.6.3 TMR1功能描述

### 10.6.3.1 时基单元

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

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器 (TMRx\_CNT)
- 预分频器寄存器 (TMRx\_DIV)
- 自动装载寄存器 (TMRx\_AR)
- 重复次数寄存器 (TMRx\_RC)

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TMRx\_CTRL1 寄存器中的自动装载预装载使能位(ARPEN)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TMRx\_CTRL1 寄存器中的 UEVDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK\_CNT 驱动，仅当设置了计数器 TMRx\_CTRL1 寄存器中的计数器使能位 (CNTEN) 时，CK\_CNT 才有效。（更多有关使能计数器的细节，请参见控制器的从模式描述）。

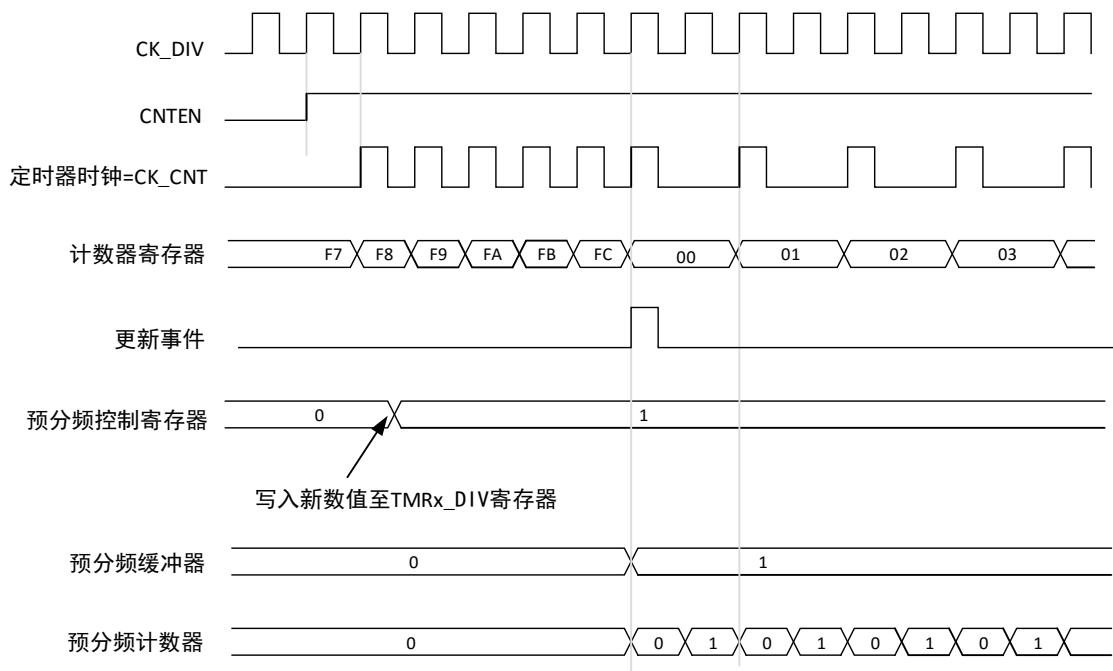
注意，在设置了 TMRx\_CTRL 寄存器的 CNTEN 位的一个时钟周期后，计数器开始计数。

#### 预分频器描述

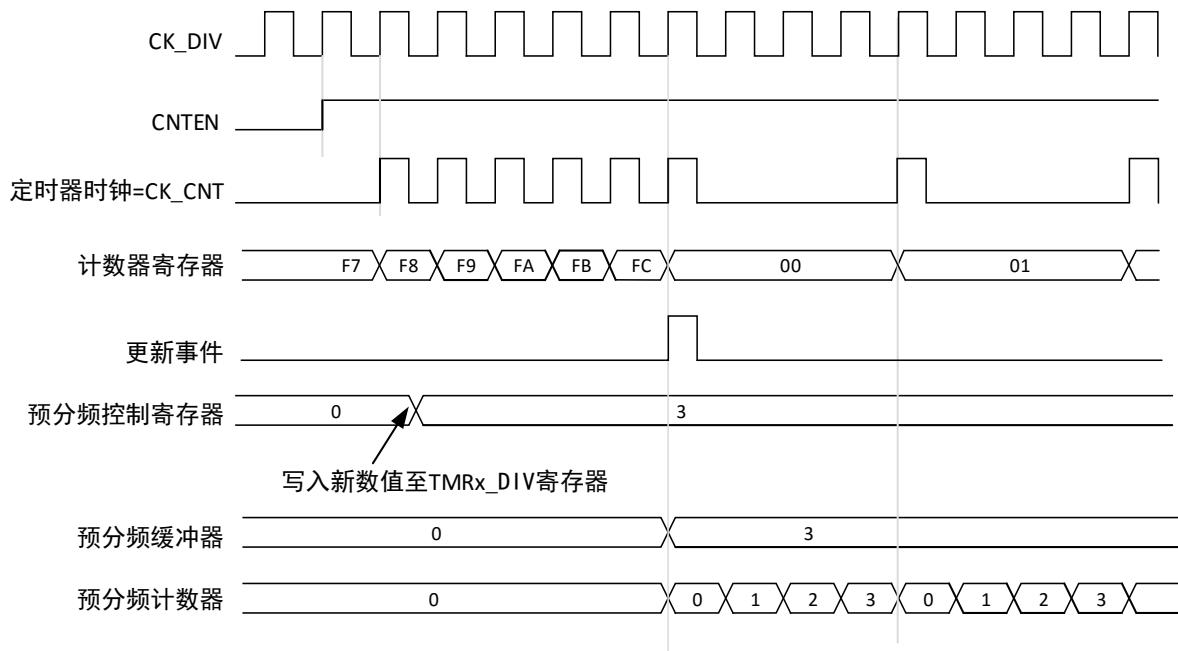
预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TMRx\_DIV 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

[图 10-121](#) 和 [图 10-122](#) 给出了在预分频器运行时，更改计数器参数的例子。

图表 10-121 当预分频器的参数从1变到2时，计数器的时序图



图表 10-122 当预分频器的参数从1变到4时，计数器的时序图



### 10.6.3.2 计数器模式

#### 向上计数模式

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

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

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

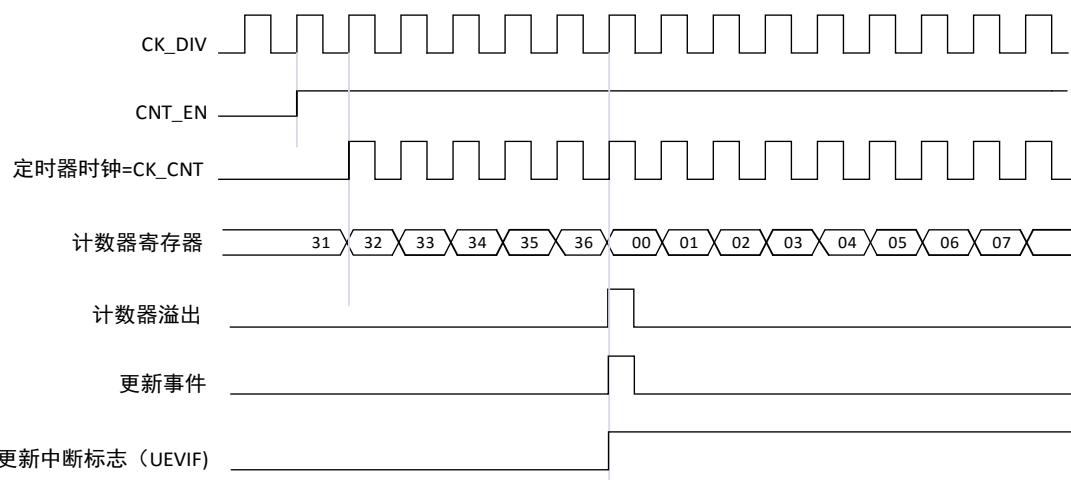
设置 **TMRx\_CTRL1** 寄存器中的 **UEVDIS** 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 **UEVDIS** 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清'0'，同时预分频器的计数也被清 0（但预分频器的数值不变）。此外，如果设置了 **TMRx\_CTRL1** 寄存器中的 **UVERS** 位（选择更新请求），设置 **UEVG** 位将产生一个更新事件 **UEV**，但硬件不设置 **UEVIF** 标志（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 **UVERS** 位）设置更新标志位（**TMRx\_STS** 寄存器中的 **UEVIF** 位）。

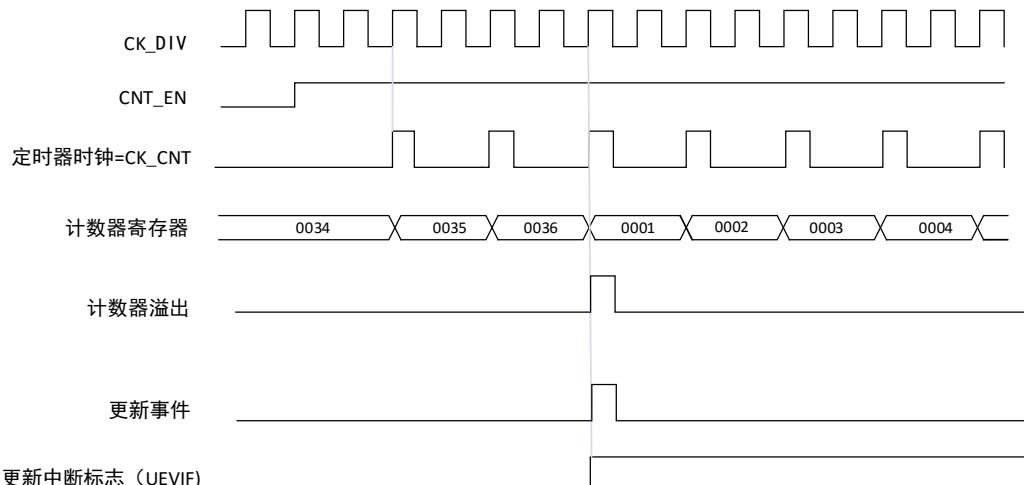
- 重复计数器被重新加载为 **TMRx\_RC** 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值（**TMRx\_AR**）。
- 预分频器的缓冲区被置入预装载寄存器的值（**TMRx\_DIV** 寄存器的内容）。

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

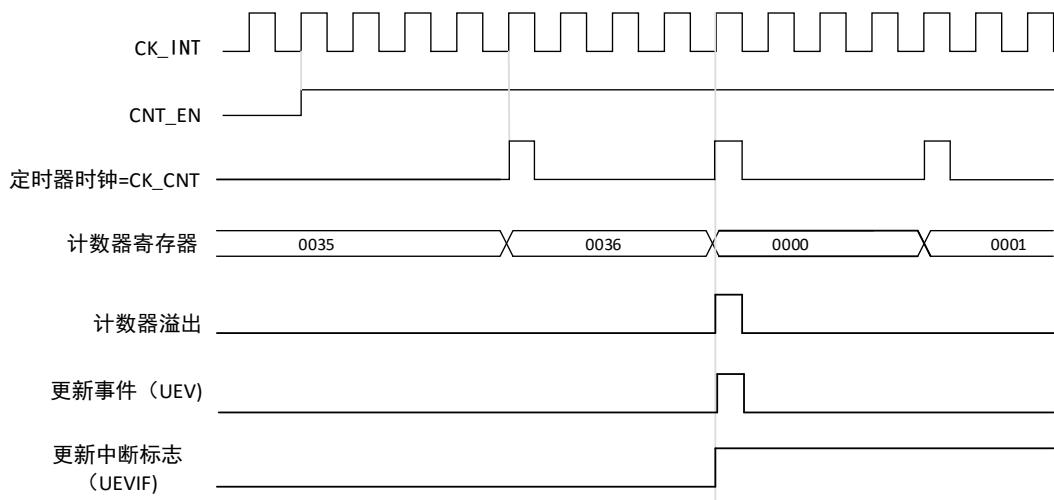
图表 10-123 计数器时序图，内部时钟分频因子为1



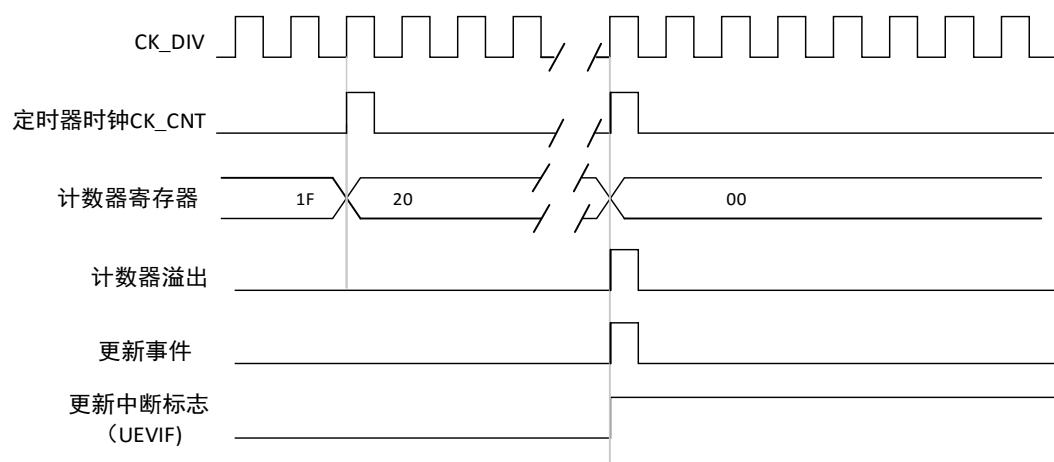
图表 10-124 计数器时序图，内部时钟分频因子为2



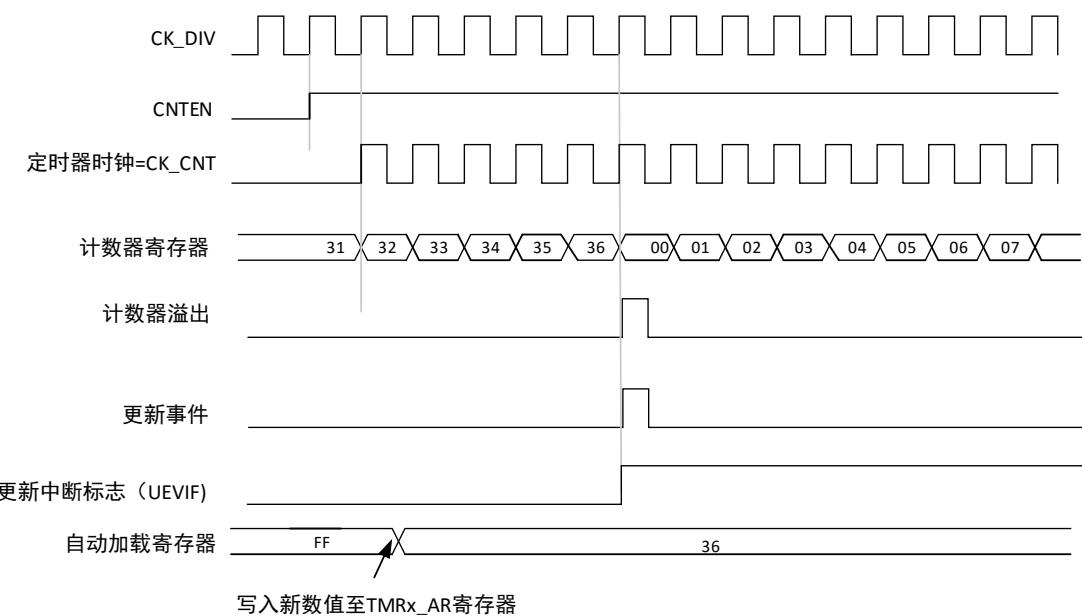
图表 10-125 计数器时序图，内部时钟分频因子为4



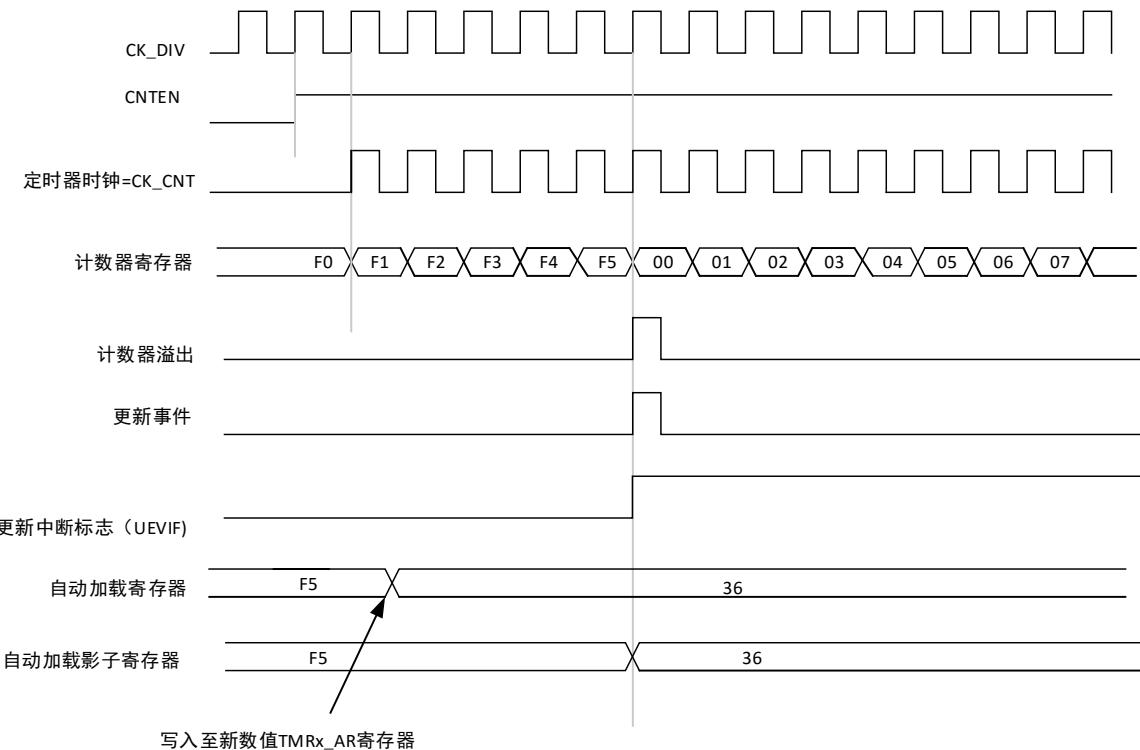
图表 10-126 计数器时序图，内部时钟分频因子为N



图表 10-127 计数器时序图，当ARPEN=0时的更新事件 (TMRx\_AR没有预装入)



图表 10-128 计数器时序图, 当ARPEN=1时的更新事件 (预装入了TMRx\_AR)



### 向下计数模式

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

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

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

设置 TMRx\_CTRL1 寄存器的 UEVDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UEVDIS 位被清为 0 之前不会产生更新事件。然而, 计数器仍会从当前自动加载值重新开始计数, 并且预分频器的计数器重新从 0 开始 (但预分频系数不变)。此外, 如果设置了 TMRx\_CTRL1 寄存器中的 UVERS 位 (选择更新请求), 设置 UEVG 位将产生一个更新事件 UEV 但不设置 UEVIF 标志 (因此不产生中断和 DMA 请求), 这是为了避免在发生捕获事件并清除计数器时, 同时产生更新和捕获中断。

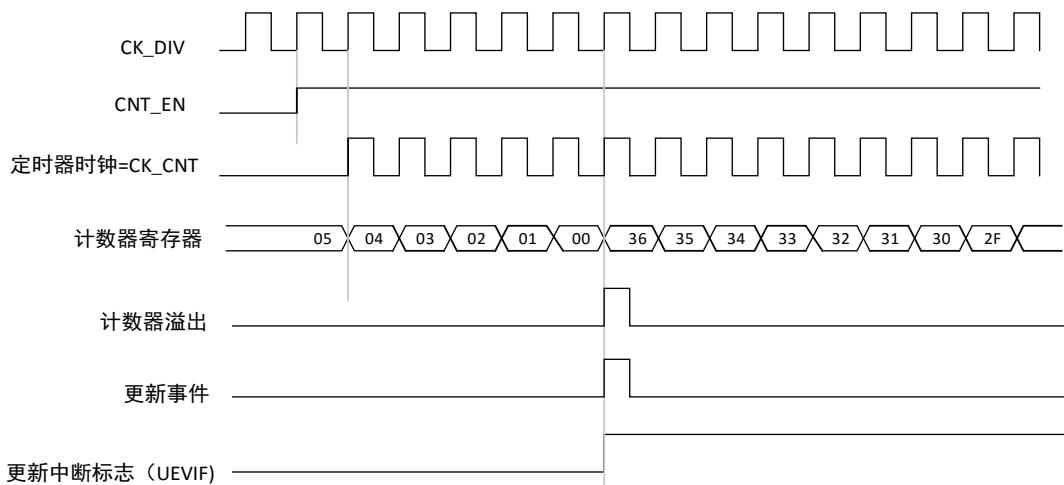
当发生更新事件时, 所有的寄存器都被更新, 并且 (根据 UVERS 位的设置) 更新标志位 (TMRx\_STS 寄存器中的 UEVIF 位) 也被设置。

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

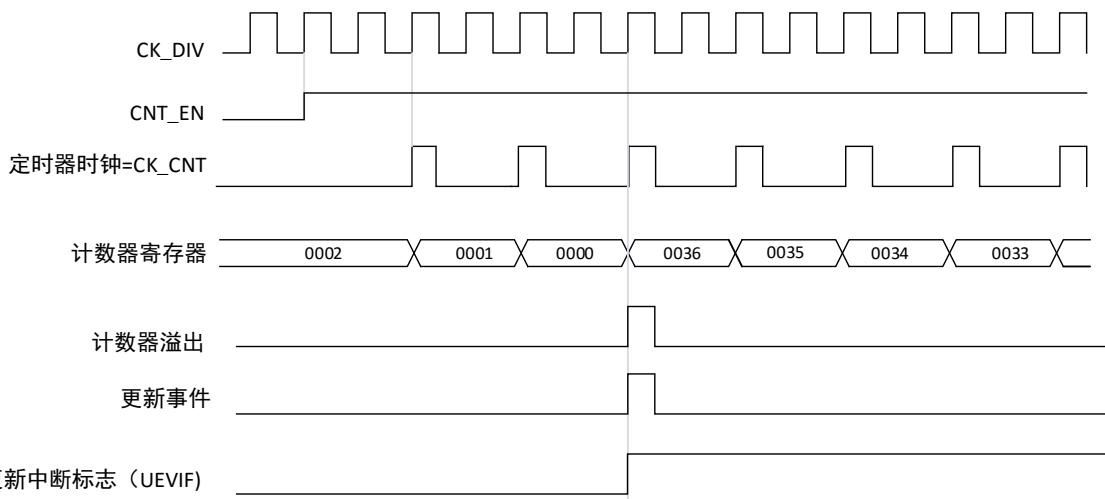
**注意:** 自动装载在计数器重载入之前被更新, 因此下一个周期将是预期的值。

以下是一些当 TMRx\_AR=0x36 时, 计数器在不同时钟频率下的操作例子。

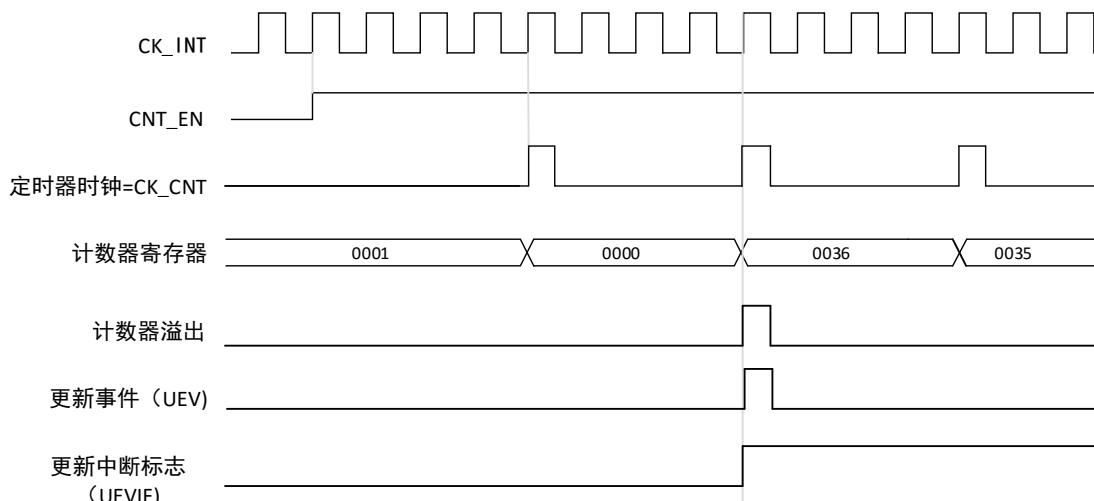
图表 10-129 计数器时序图，内部时钟分频因子为1



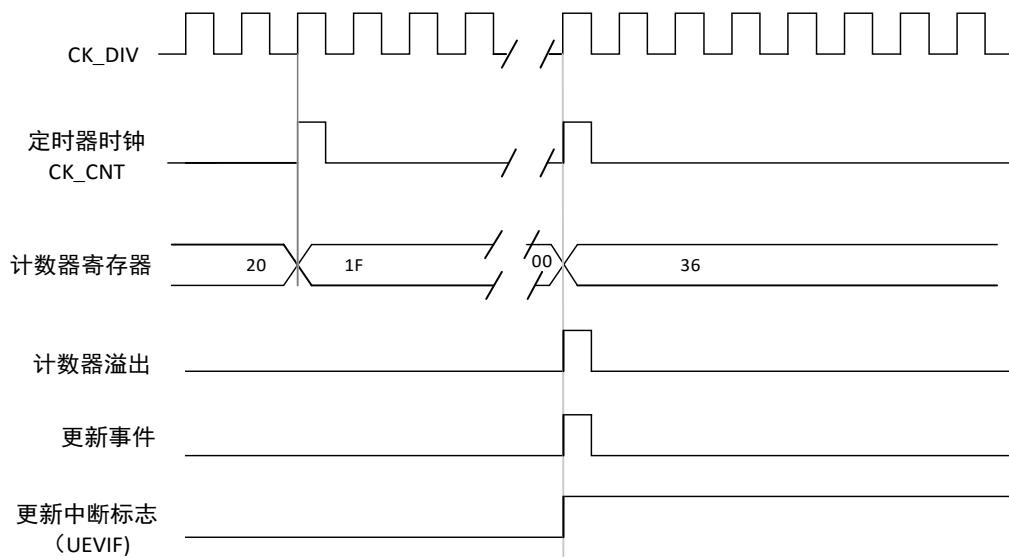
图表 10-130 计数器时序图，内部时钟分频因子为2



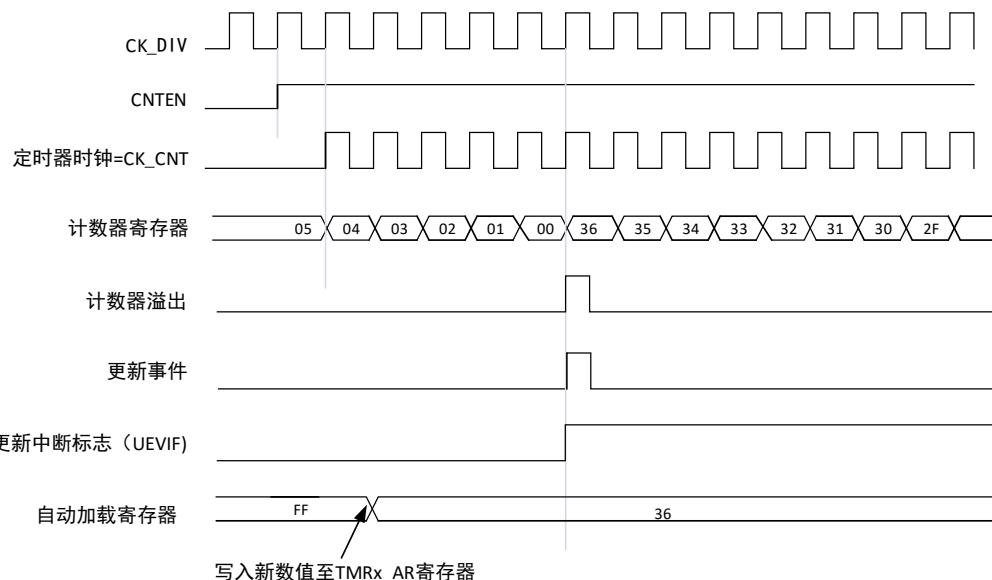
图表 10-131 计数器时序图，内部时钟分频因子为4



图表 10-132 计数器时序图，内部时钟分频因子为N



图表 10-133 计数器时序图，当没有使用自动装载时的更新事件



### 中央对齐模式（向上/向下计数）

在中央对齐模式，计数器从 0 开始计数到自动加载的值 (TMRx\_AR 寄存器) -1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在此模式下，不能写入 TMRx\_CTRL1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

设置 TMRx\_CTRL1 寄存器中的 UEVDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UEVDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。此外，如果设置了 TMRx\_CTRL1 寄存器中的 UVERS 位（选择更新请求），设置 UEVG 位将产生一个更新事件 UEV 但不设置 UEVIF 标志（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

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

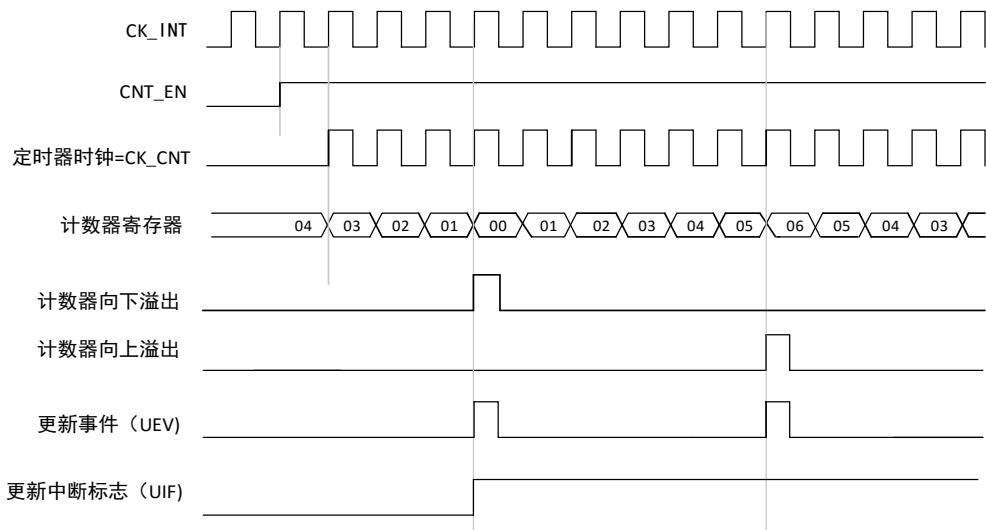
- 重复计数器被重置为 TMRx\_RC 寄存器中的内容

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

**注意：**如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值（计数器被装载为新的值）。

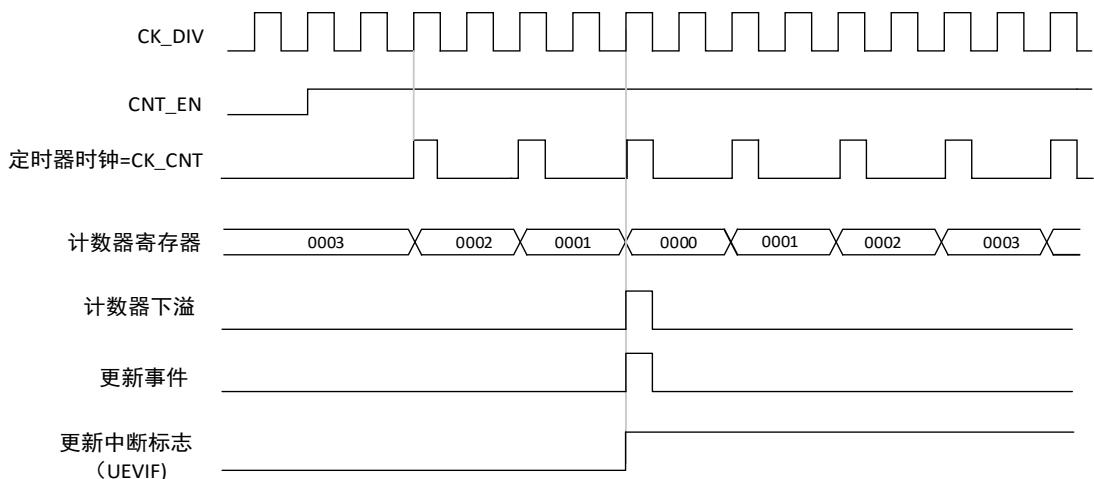
以下是一些计数器在不同时钟频率下的操作的例子：

图表 10-134 计数器时序图，内部时钟分频因子为1，TMRx\_AR=0x6

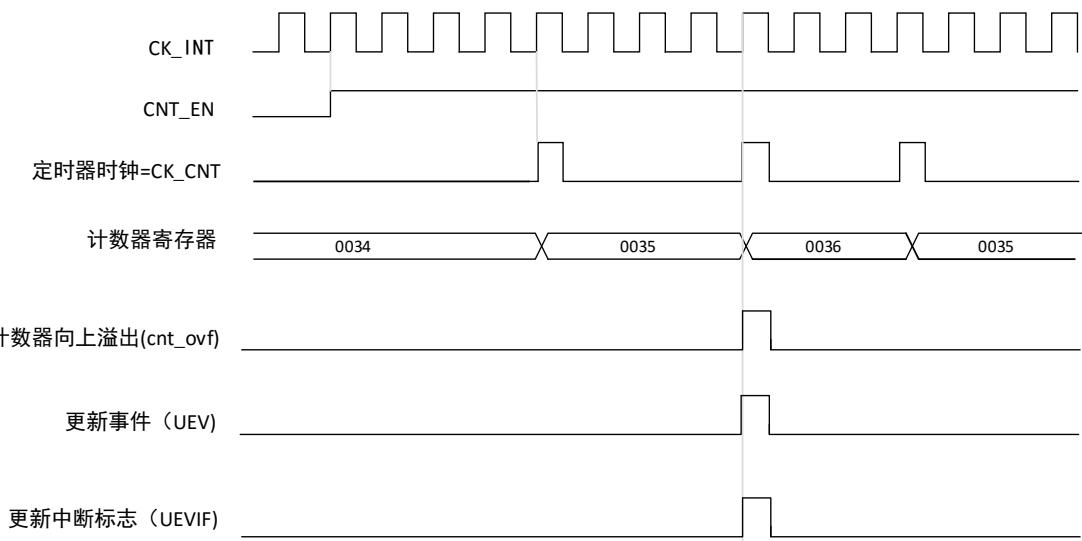


**注意：**这里使用了中心对齐模式1（详见 [10.6.4.1 寄存器描述](#)）。

图表 10-135 计数器时序图，内部时钟分频因子为2

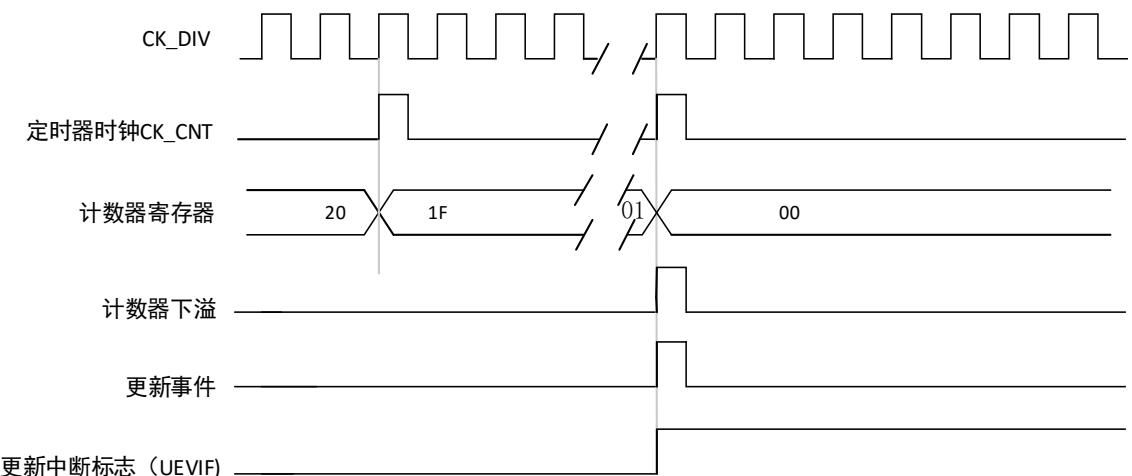


图表 10-136 计数器时序图，内部时钟分频因子为4，TMRx\_AR=0x36

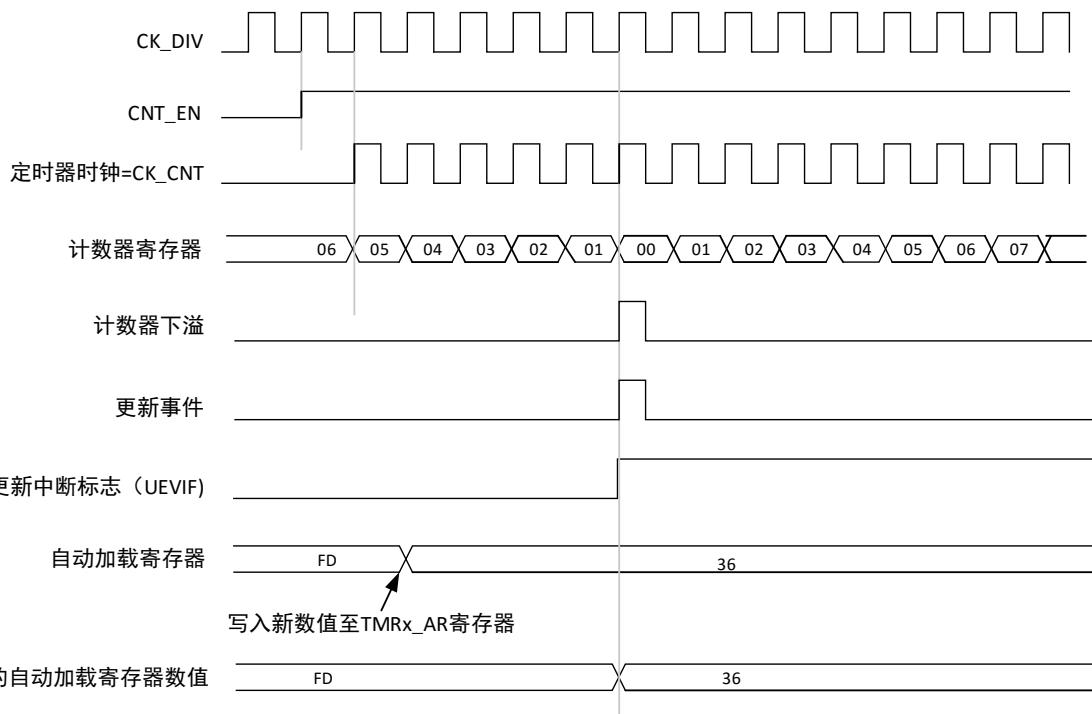


注：这里使用了中心对齐模式2或3，计数器溢出时设置UEVIF

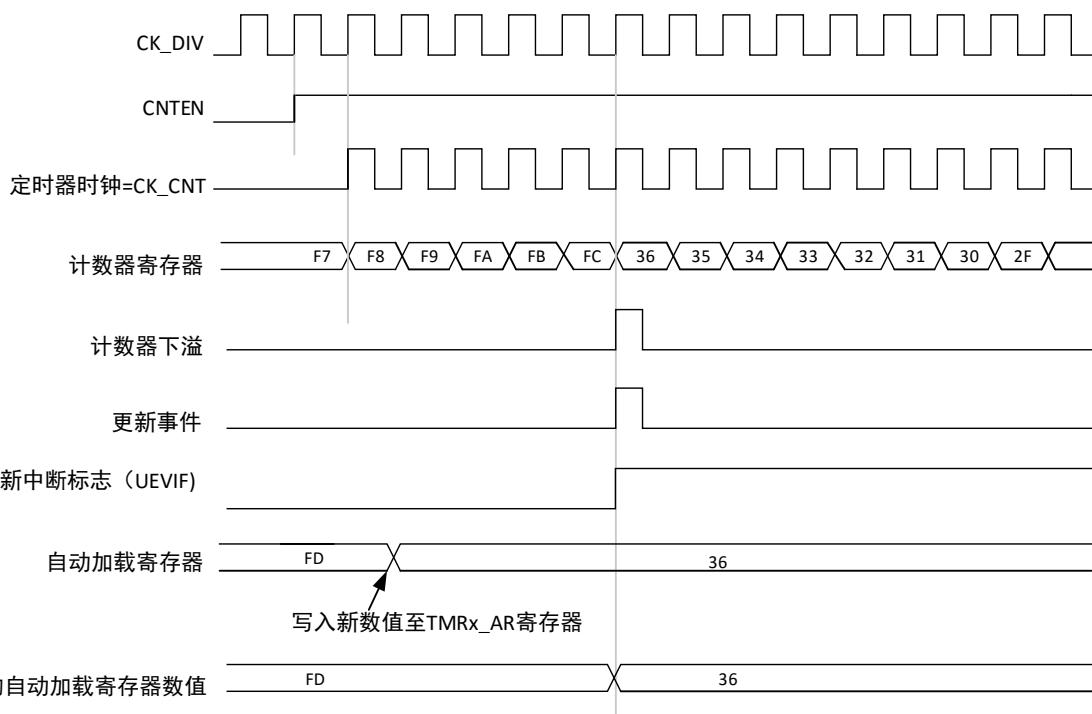
图表 10-137 计数器时序图，内部时钟分频因子为N



图表 10-138 计数器时序图, ARPEN=1时的更新事件（计数器下溢）



图表 10-139 计数器时序图, ARPEN=1时的更新事件（计数器溢出）



### 10.6.3.3 重复计数器

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

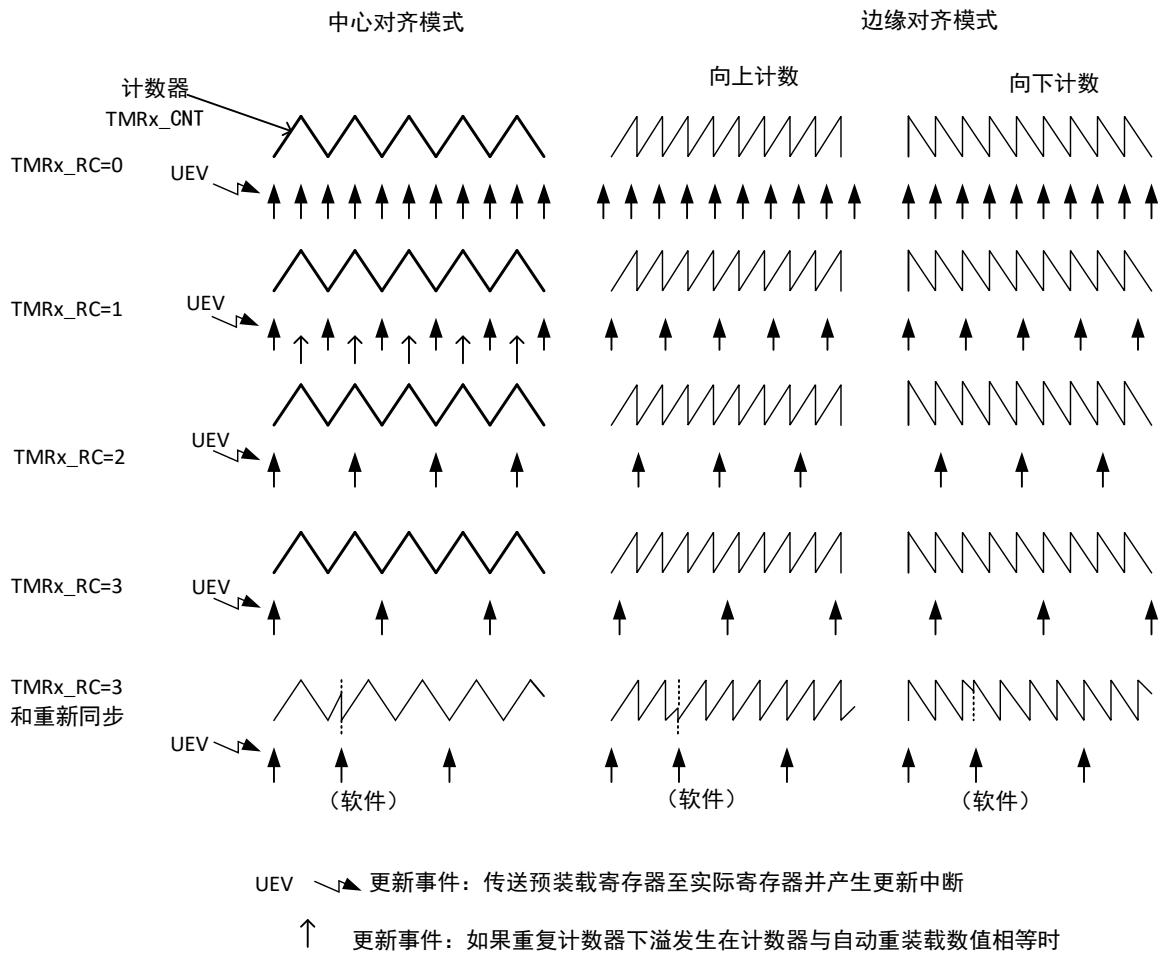
这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器 (TMRx\_AR 自动重载入寄存器, TMRx\_DIV 预装载寄存器, 还有在比较模式下的捕获/比较寄存 (TMRx\_CCx), N 是 TMRx\_RC 重复计数寄存器中的值)。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时
- 向下计数模式下每次计数器下溢时
- 中央对齐模式下每次上溢和每次下溢时。虽然这样限制了PWM的最大循环周期为128，但它能够在每个PWM周期2次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个PWM周期中仅刷新一次比较寄存器，则最大的分辨率为 $2 \times T_{ck}$ 。

重复计数器是自动加载的，重复速率是由TMRx\_RC寄存器的值定义（参看图10-140）。当更新事件由软件产生（通过设置TMRx\_EVEG中的UEVG位）或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且TMRx\_RC寄存器中的内容被重载入到重复计数器。

图表 10-140 不同模式下更新速率的例子，及TMRx\_RC的寄存器设置



#### 10.6.3.4 时钟选择

计数器时钟可由下列时钟源提供：

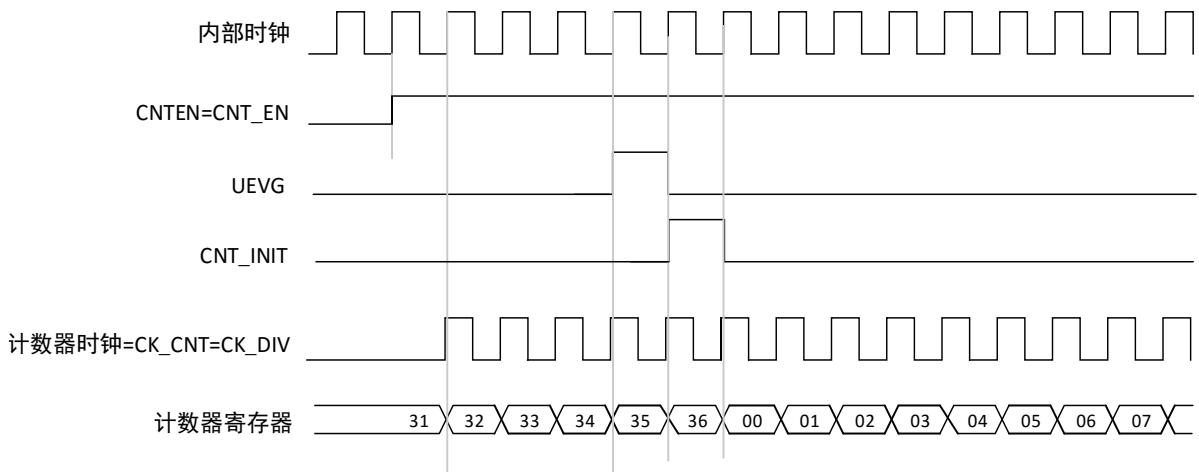
- 内部时钟（CK\_INT）
- 外部时钟模式1：外部输入引脚
- 外部时钟模式2：外部触发输入ETR
- 内部触发输入（ITRx）：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器Timer1而作为另一个定时器Timer2的预分频器。详见10.6.3.20节。

##### 内部时钟源（CK\_INT）

如果禁止了从模式控制器（SMSEL=000），则CNTEN、DIR（TMRx\_CTRL1寄存器）和UEVG位（TMRx\_EVEG寄存器）是实际的控制位，并且只能被软件修改（UEVG位仍被自动清除）。只要CNTEN位被写成‘1’，预分频器的时钟就由内部时钟CK\_INT提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

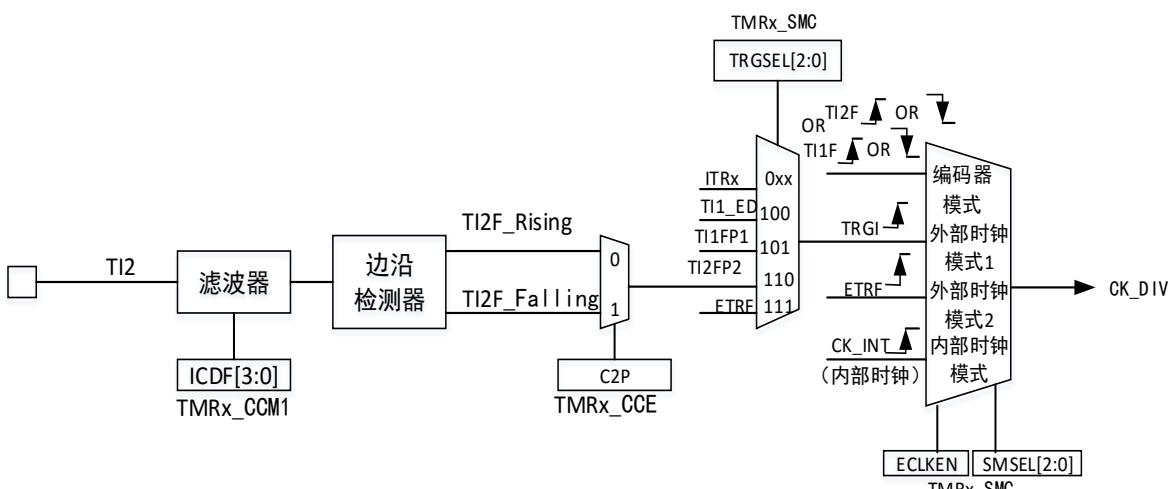
图表 10-141 一般模式下的控制电路，内部时钟分频因子为1



### 外部时钟源模式 1

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

图表 10-142 TI2 外部时钟连接例子



例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

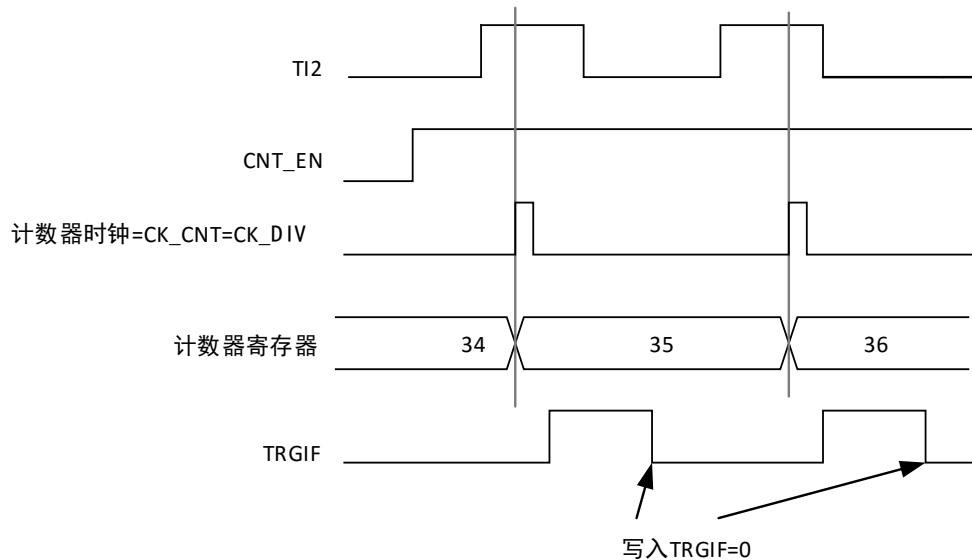
1. 配置 TMRx\_CCM1 寄存器 C2SEL=01，配置通道 2 检测 TI2 输入的上升沿。
2. 配置 TMRx\_CCM1 寄存器的 IC2DF[3:0]，选择输入滤波器带宽(如果不需滤波器，保持 IC2DF = 0000)。
3. 配置 TMRx\_CCE 寄存器的 C2P=0，选定上升沿极性。
4. 配置 TMRx\_SMC 寄存器的 SMSEL=111，选择定时器外部时钟模式 1。
5. 配置 TMRx\_SMC 寄存器中的 TRGSEL=110，选定 TI2 作为触发输入源。
6. 设置 TMRx\_CTRL1 寄存器的 CNTEN=1，启动计数器。

**注意：**捕获预分频器不用作触发，所以不需要对它进行配置。

当上升沿出现在 TI2，计数器计数一次，且 TRGIF 标志被设置。

在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。

图表 10-143 外部时钟模式 1 下的控制电路



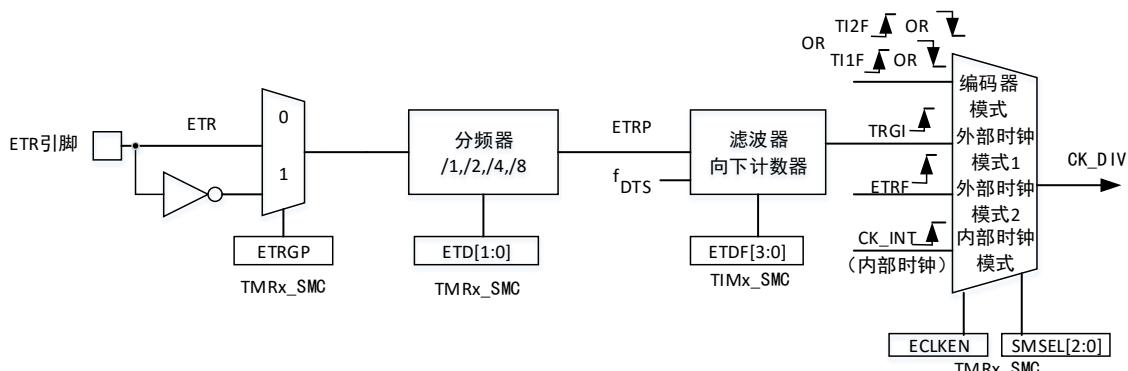
### 外部时钟源模式 2

选定此模式的方法为：令 TMRx\_SMC 寄存器中的 ECLKEN=1。

计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图：

图表 10-144 外部触发输入框图



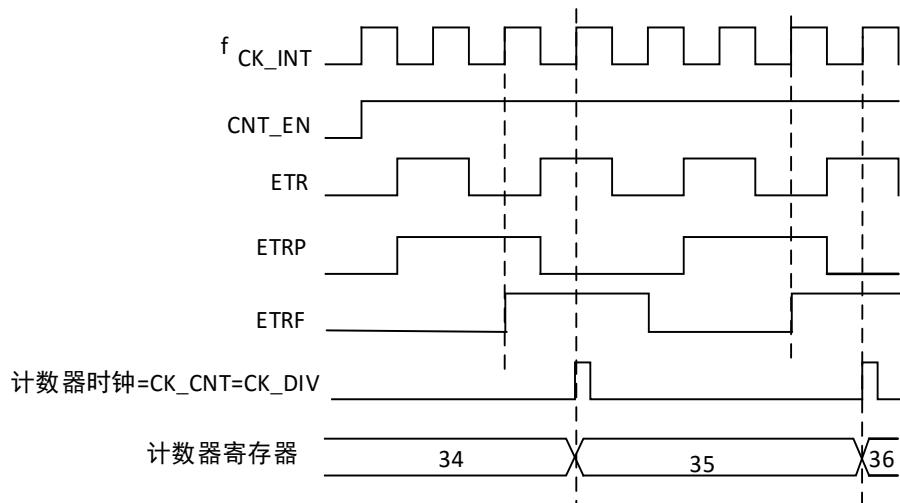
例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：

1. 本例中不需要滤波器，设置 TMRx\_SMC 寄存器中的 ETDF[3: 0]=0000。
2. 设置预分频器，置 TMRx\_SMC 寄存器中的 ETD[1: 0]=01。
3. 选择 ETR 的上升沿检测，置 TMRx\_SMC 寄存器中的 ETRGP=0。
4. 开启外部时钟模式 2，写 TMRx\_SMC 寄存器中的 ECLKEN=1。
5. 启动计数器，写 TMRx\_CTRL1 寄存器中的 CNTEN=1。

计数器在每 2 个 ETR 上升沿计数一次。

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

图表 10-145 外部时钟模式2下的控制电路



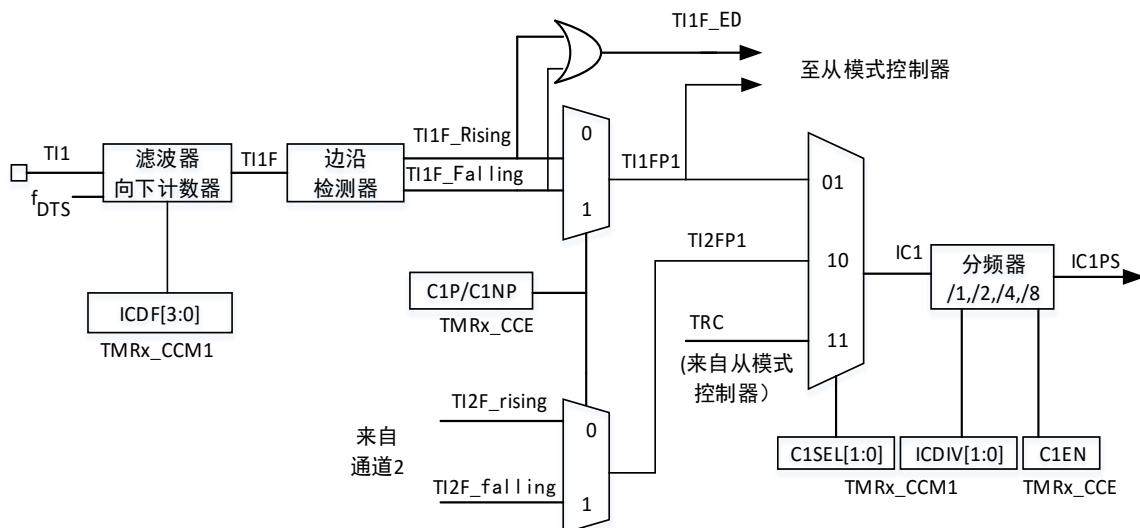
### 10.6.3.5 捕获/比较通道

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

[图 10-146 至图 10-149](#) 是一个捕获/比较通道概览。

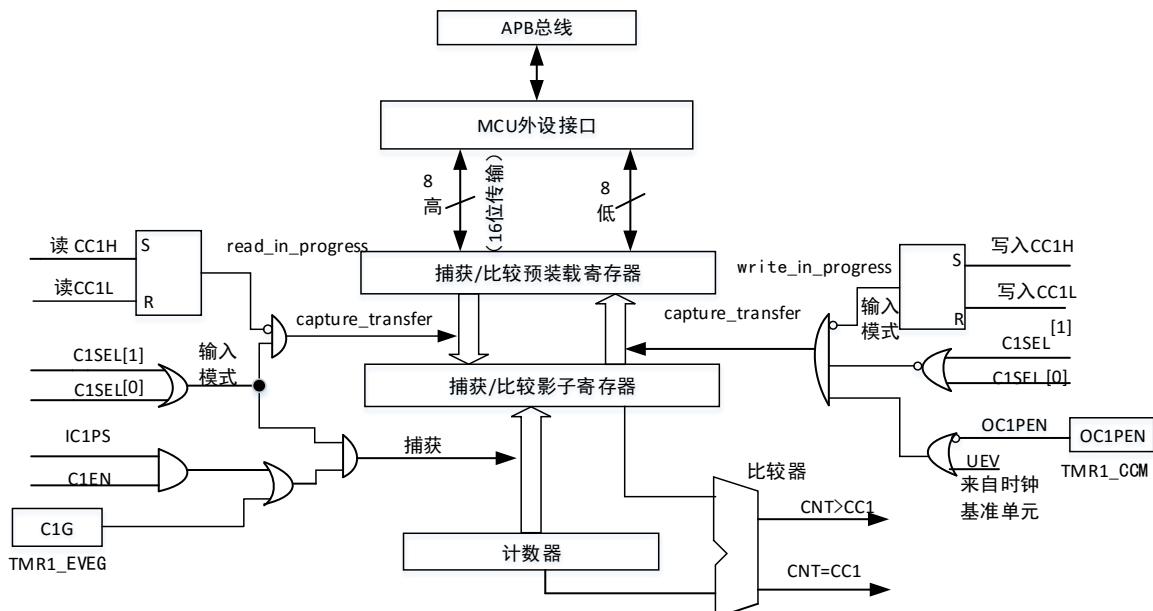
输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号 (TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器 (ICxPS)。

图表 10-146 捕获/比较通道（如：通道1输入部分）

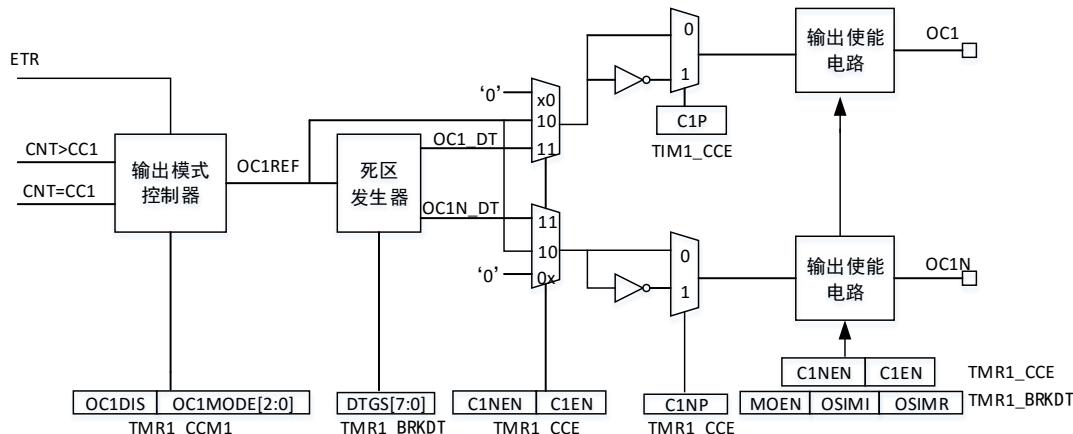


输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

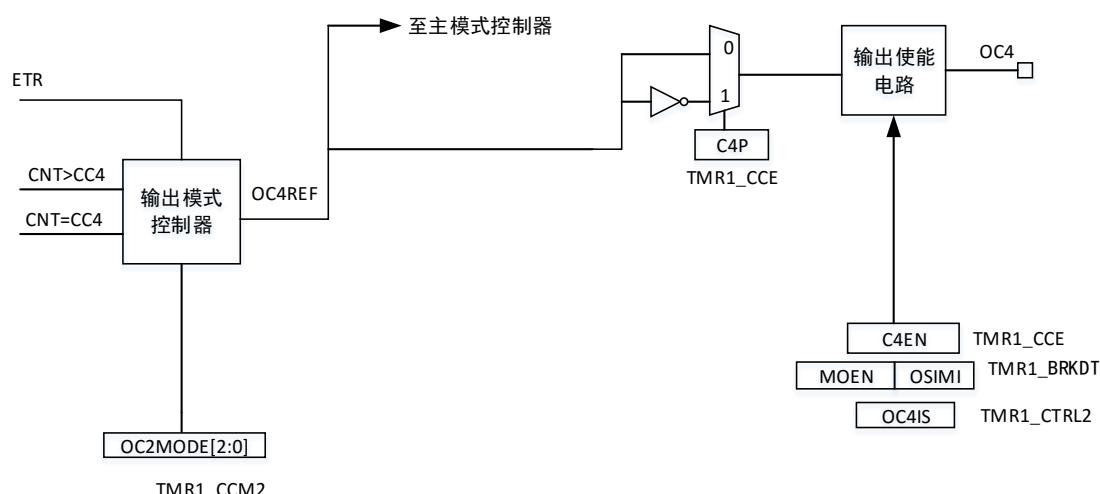
图表 10-147 捕获/比较通道1的主电路



图表 10-148 捕获/比较通道的输出部分（通道 1 至 3）



图表 10-149 捕获/比较通道的输出部分（通道 4）



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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

### 10.6.3.6 输入捕获模式

在输入捕获模式下，当检测到 IC<sub>x</sub> 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器 (TMR<sub>x</sub>\_CC<sub>x</sub>) 中。当发生捕获事件时，相应的 CxIF 标志 (TMR<sub>x</sub>\_STS 寄存器) 被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CxIF 标志已经为高，那么重复捕获标志 CxOF (TMR<sub>x</sub>\_STS 寄存器) 被置 1。写 CxIF=0 可清除 CxIF，或读取存储在 TMR<sub>x</sub>\_CC<sub>x</sub> 寄存器中的捕获数据也可清除 CxIF。写 CxOF=0 可清除 CxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TMR<sub>x</sub>\_CC1 寄存器中，步骤如下：

- 选择有效输入端：TMR<sub>x</sub>\_CC1 必须连接到 TI1 输入，所以写入 TMR<sub>x</sub>\_CC1 寄存器中的 C1SEL=01，只要 C1SEL 不为 '00'，通道被配置为输入，并且 TMR<sub>x</sub>\_CC1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TI<sub>x</sub> 时，输入滤波器控制位是 TMR<sub>x</sub>\_CCM<sub>x</sub> 寄存器中的 IC<sub>x</sub>DF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以  $f_{DTS}$  频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TMR<sub>x</sub>\_CCM1 寄存器中写入 IC1DF=0011。
- 选择 TI1 通道的有效转换边沿，在 TMR<sub>x</sub>\_CCE 寄存器中写入 C1P=0（上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TMR<sub>x</sub>\_CCM1 寄存器的 IC1PS=00）。
- 设置 TMR<sub>x</sub>\_CCE 寄存器的 C1EN=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1IE 位允许相关中断请求，通过设置 TMR<sub>x</sub>\_DIE 寄存器中的 C1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TMR<sub>x</sub>\_CC1 寄存器。
- C1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 C1IF 未曾被清除，C1OF 也被置 1。
- 如设置了 C1IE 位，则会产生一个中断。
- 如设置了 C1DE 位，则还会产生一个 DMA 请求。

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

注意：设置 TMR<sub>x</sub>\_EVEG 寄存器中相应的 CxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

### 10.6.3.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

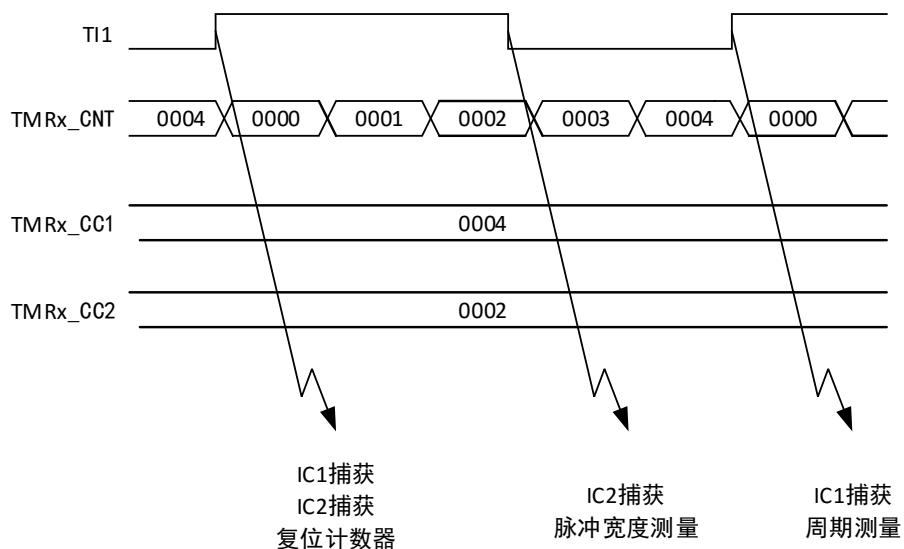
- 两个 IC<sub>x</sub> 信号被映射至同一个 TI<sub>x</sub> 输入。
- 这 2 个 IC<sub>x</sub> 信号为边沿有效，但是极性相反。
- 其中一个 TI<sub>x</sub>FP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。

例如，你需要测量输入到 TI1 上的 PWM 信号的长度 (TMR<sub>x</sub>\_CC1 寄存器) 和占空比 (TMR<sub>x</sub>\_CC2 寄存器)，具体步骤如下（取决于 CK\_INT 的频率和预分频器的值）

- 选择 TMR<sub>x</sub>\_CC1 的有效输入：置 TMR<sub>x</sub>\_CCM1 寄存器的 C1SEL=01（选中 TI1）。
- 选择 TI1FP1 的有效极性（用来捕获数据到 TMR<sub>x</sub>\_CC1 中和清除计数器）：置 C1P=0（上升沿有效）。
- 选择 TMR<sub>x</sub>\_CC2 的有效输入：置 TMR<sub>x</sub>\_CCM1 寄存器的 C2SEL=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TMR<sub>x</sub>\_CC2）：置 C2P=1（下降沿有效）。
- 选择有效的触发输入信号：置 TMR<sub>x</sub>\_SMC 寄存器中的 TRGSEL=101（选择 TI1FP1）。
- 配置从模式控制器为复位模式：置 TMR<sub>x</sub>\_SMC 中的 SMSEL=100。

- 使能捕获：置 TMRx\_CCE 寄存器中 C1EN=1 且 C2EN=1。

图表 10-150 PWM 输入模式时序



注意：因为只有 *TI1FP1* 和 *TI2FP2* 连到了从模式控制器，所以 PWM 输入模式只能使用 *TMRx\_CH1/TMRx\_CH2* 信号。

#### 10.6.3.8 强置输出模式

在输出模式 (*TMRx\_CCMx* 寄存器中 *CxSEL=00*) 下，输出比较信号 (*OCxREF* 和相应的 *OCx/OCxN*) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 *TMRx\_CCMx* 寄存器中相应的 *OCxMODE=101*，即可强置输出比较信号 (*OCxREF/OCx*) 为有效状态。这样 *OCxREF* 被强置为高电平 (*OCxREF* 始终为高电平有效)，同时 *OCx* 得到 *CxP* 极性相反的信号。

例如：*CxP=0* (*OCx* 高电平有效)，则 *OCx* 被强置为高电平。

置 *TMRx\_CCMx* 寄存器中的 *OCxMODE=100*，可强置 *OCxREF* 信号为低。

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

#### 10.6.3.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式 (*TMRx\_CCMx* 寄存器中的 *OCxMODE* 位) 和输出极性 (*TMRx\_CCE* 寄存器中的 *CxP* 位) 定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平 (*OCxMODE=000*)、被设置成有效电平 (*OCxMODE=001*)、被设置成无效电平 (*OCxMODE=010*) 或进行翻转 (*OCxMODE=011*)。
- 设置中断状态寄存器中的标志位 (*TMRx\_STS* 寄存器中的 *CxIF* 位)。
- 若设置了相应的中断屏蔽 (*TMRx\_DIE* 寄存器中的 *CxIE* 位)，则产生一个中断。
- 若设置了相应的使能位 (*TMRx\_DIE* 寄存器中的 *CxDE* 位，*TMRx\_CTRL2* 寄存器中的 *CDSEL* 位选择 DMA 请求功能)，则产生一个 DMA 请求。

*TMRx\_CCMx* 中的 *OCxPEN* 位选择 *TMRx\_CCx* 寄存器是否需要使用预装载寄存器。

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

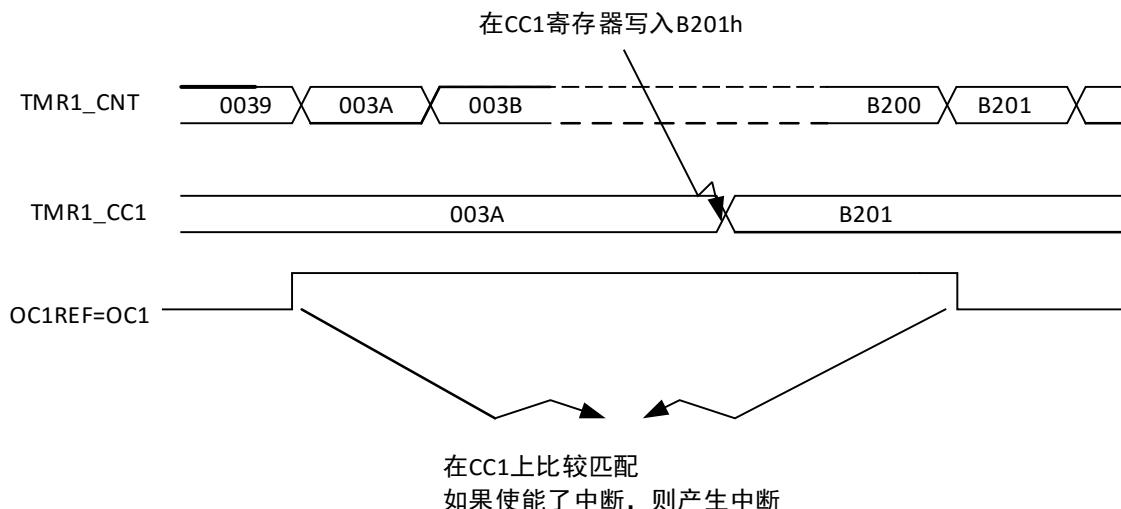
同步的精度可以达到计数器的一个计数周期。输出比较模式 (在单脉冲模式下) 也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部，外部，预分频器）。
2. 将相应的数据写入 TMRx\_AR 和 TMRx\_CCx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxEIE 位。
4. 选择输出模式，例如：
  - 要求计数器与 CCx 匹配时翻转 OCx 的输出引脚，设置 OCxMODE=011
  - 置 OCxPEN = 0 禁用预装载寄存器
  - 置 CxP = 0 选择极性为高电平有效
  - 置 CxEN = 1 使能输出
5. 设置 TMRx\_CTRL1 寄存器的 CNTEN 位启动计数器

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

图表 10-151 输出比较模式，翻转OC1



### 10.6.3.10 PWM模式

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

在 TMRx\_CCMx 寄存器中的 OCxMODE 位写入'110' (PWM 模式 1) 或'111' (PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TMRx\_CCMx 寄存器的 OCxPEN 位使能相应的预装载寄存器，最后还要设置 TMRx\_CTRL1 寄存器的 AR PEN 位，(在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

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

OCx 的极性可以通过软件在 TMRx\_CCE 寄存器中的 CxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过 (TMRx\_CCE 和 TMRx\_BRKDT 寄存器中) CxEN、CxNEN、MOEN、OSIMI 和 OSIMR 位的组合控制。详见 TMRx\_CCE 寄存器的描述。

在 PWM 模式 (模式 1 或模式 2) 下，TMRx\_CNT 和 TMRx\_CCx 始终在进行比较，(依据计数器的计数方向) 以确定是否符合  $TMRx\_CCx \leq TMRx\_CNT$  或者  $TMRx\_CNT \leq TMRx\_CCx$ 。

根据 TMRx\_CTRL1 寄存器中 CMSEL 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

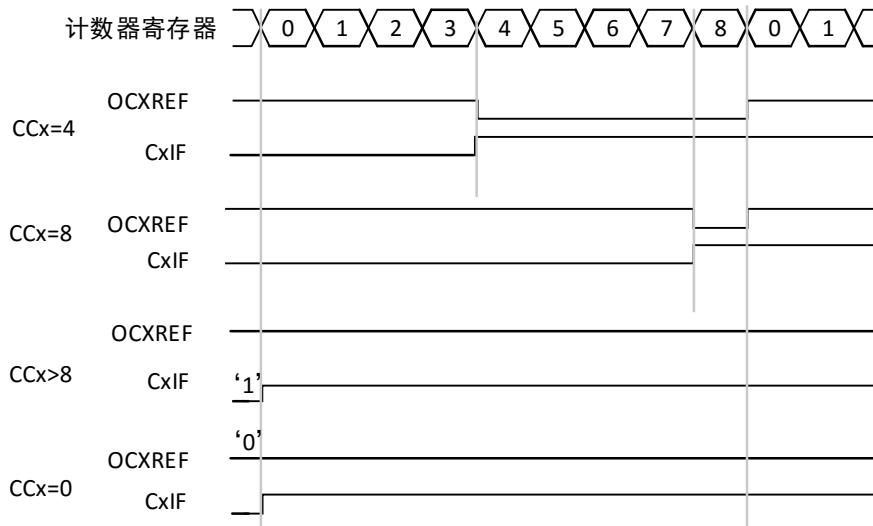
#### PWM 边沿对齐模式

- 向上计数配置

当 TMRx\_CTRL1 寄存器中的 DIR 位为低的时候执行向上计数。参看 10.6.3.2 节。

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

图表 10-152 边沿对齐的 PWM 波形 (AR=8)



- 向下计数的配置

当 TMRx\_CTRL1 寄存器的 DIR 位为高时执行向下计数。参看 10.6.3.2 节。

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

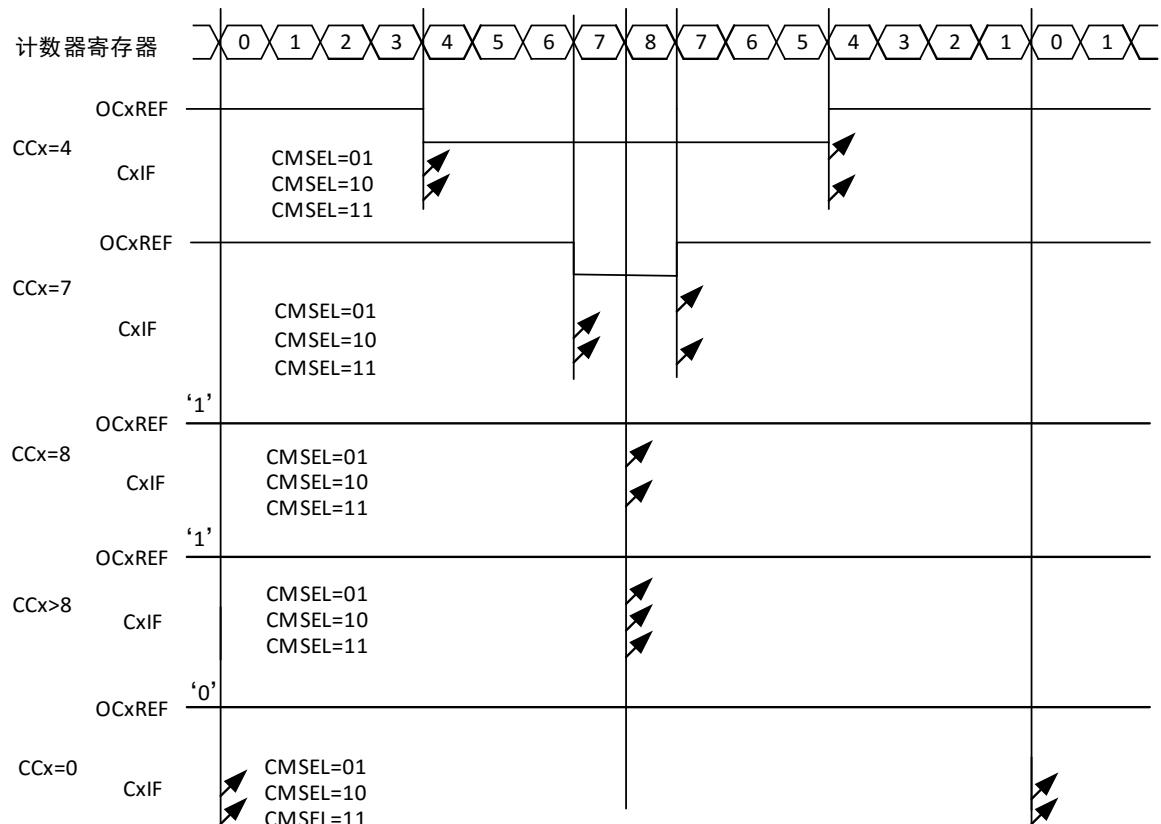
### PWM 中央对齐模式

当 TMRx\_CTRL1 寄存器中的 CMSEL 位不为'00'时为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMSEL 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TMRx\_CTRL1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。参看 [10.6.3.2 节](#) 的中央对齐模式。

下图给出了一些中央对齐的 PWM 波形的例子

- TMRx\_AR=8
- PWM 模式 1
- TMRx\_CTRL1 寄存器的 CMSEL=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图表 10-153 中央对齐的 PWM 波形 (AR=8)



#### 使用中央对齐模式的提示:

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于TMRx\_CTRL1寄存器中DIR位的当前值。此外，软件不能同时修改DIR和CMSEL位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值(TMRx\_CNT>TMRx\_AR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将0或者TMRx\_AR的值写入计数器，方向被更新，但不产生更新事件UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置TMRx\_EVEG位中的UEVG位），并且不要在计数进行过程中修改计数器的值。

#### 10.6.3.11 互补输出和死区插入

高级控制定时器(TMR1)能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

配置TMRx\_CCE寄存器中的CxP和CxNP位，可以为每一个输出独立地选择极性(主输出OCx或互补输出OCxN)。

互补信号OCx和OCxN通过下列控制位的组合进行控制：TMRx\_CCE寄存器的CxEN和CxNEN位，TMRx\_BRKDT和TMRx\_CTRL2寄存器中的MOEN、OCxIS、OCxNIS、OSIMI和OSIMR位，详见表10-13带刹车功能的互补输出通道OCx和OCxN的控制位。特别的是，在转换到IDLEF状态时(MOEN下降到0)死区被激活。

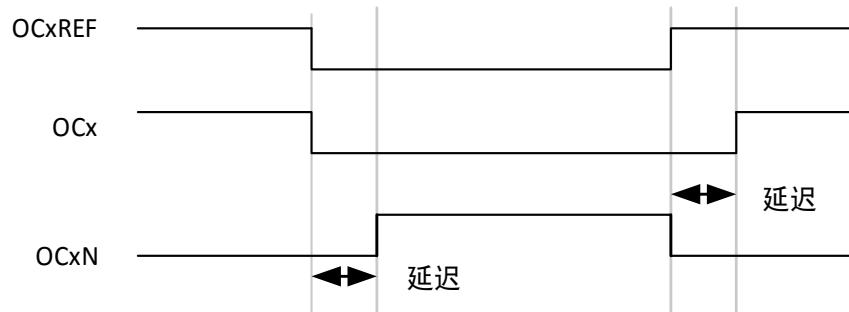
同时设置CxEN和CxNEN位将插入死区，如果存在刹车电路，则还要设置MOEN位。每一个通道都有一个10位的死区发生器。参考信号OCxREF可以产生2路输出OCx和OCxN。如果OCx和OCxN为高有效：

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

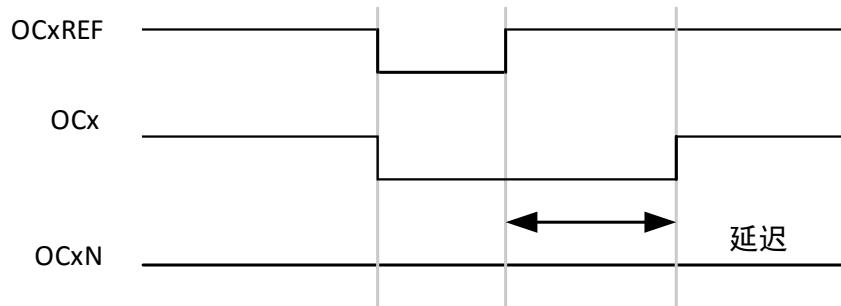
如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。

下列几张图显示了死区发生器的出信号和当前参考信号 OCxREF 之间的关系。（假设 CxP=0、CxNP=0、MOEN=1、CxEN=1 并且 CxNEN=1）

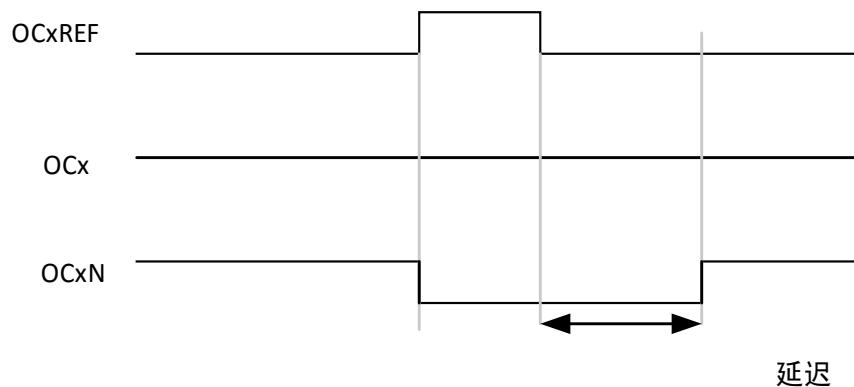
图表 10-154 带死区插入的互补输出



图表 10-155 死区波形延迟大于负脉冲



图表 10-156 死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TMRx\_BRKDT 寄存器中的 DTGS 位编程配置。详见 [10.6.4.18 节 TMR1 刹车和死区寄存器 \(TMRx\\_BRKDT\)](#) 中的延时计算。

#### 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强置、输出比较或 PWM），通过配置 TMRx\_CCE 寄存器的 CxEN 和 CxNEN 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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

### 10.6.3.12 使用刹车功能

当使用刹车功能时，依据相应的控制位 (**TMRx\_BRKDT** 寄存器中的 **MOEN**、**OSIMI** 和 **OSIMR** 位，**TMRx\_CTRL2** 寄存器中的 **OCxis** 和 **OCxnis** 位)，输出使能信号和无效电平都会被修改。但无论何时， $OCx$  和  $OCxN$  输出不能在同一时间同时处于有效电平上。详见[表 10-13 带刹车功能的互补输出通道  \$OCx\$  和  \$OCxN\$  的控制位](#)。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生，[详见 3.2.7 节时钟失效检测 \(CFD\)](#)。

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

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

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- **MOEN** 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 **OSIMI** 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 **MOEN=0**，每一个输出通道输出由 **TMRx\_CTRL2** 寄存器中的 **OCxis** 位设定的电平。如果 **OSIMI=0**，则定时器释放使能输出，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 **OCxis** 和 **OCxnis** 位指示的电平驱动输出端口。即使在这种情况下， $OCx$  和  $OCxN$  也不能被同时驱动到有效的电平。
- 注意，因为重新同步 **MOEN**，死区时间比通常情况下长一些（大约 2 个 **ck\_tim** 的时钟周期）。
- 如果 **OSIMI=0**，定时器释放使能输出，否则保持使能输出；或一旦 **CxEN** 与 **CxNEN** 之一变高时，使能输出变为高。
- 如果设置了 **TMRx\_DIE** 寄存器中的 **BRKIE** 位，当刹车状态标志（**TMRx\_STS** 寄存器中的 **BRKIF** 位）为'1'时，则产生一个中断。如果设置了 **TMRx\_DIE** 寄存器中的 **BDE** 位，则产生一个 DMA 请求。
- 如果设置了 **TMRx\_BRKDT** 寄存器中的 **AOEN** 位，在下一个更新事件 **UEV** 时 **MOEN** 位被自动置位；例如，这可以用来进行整形。否则，**MOEN** 始终保持低直到被再次置'1'；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注意：刹车输入为电平有效。所以，当刹车输入有效时，不能同时（自动地或者通过软件）设置 **MOEN**。同时，状态标志 **BRKIF** 不能被清除。

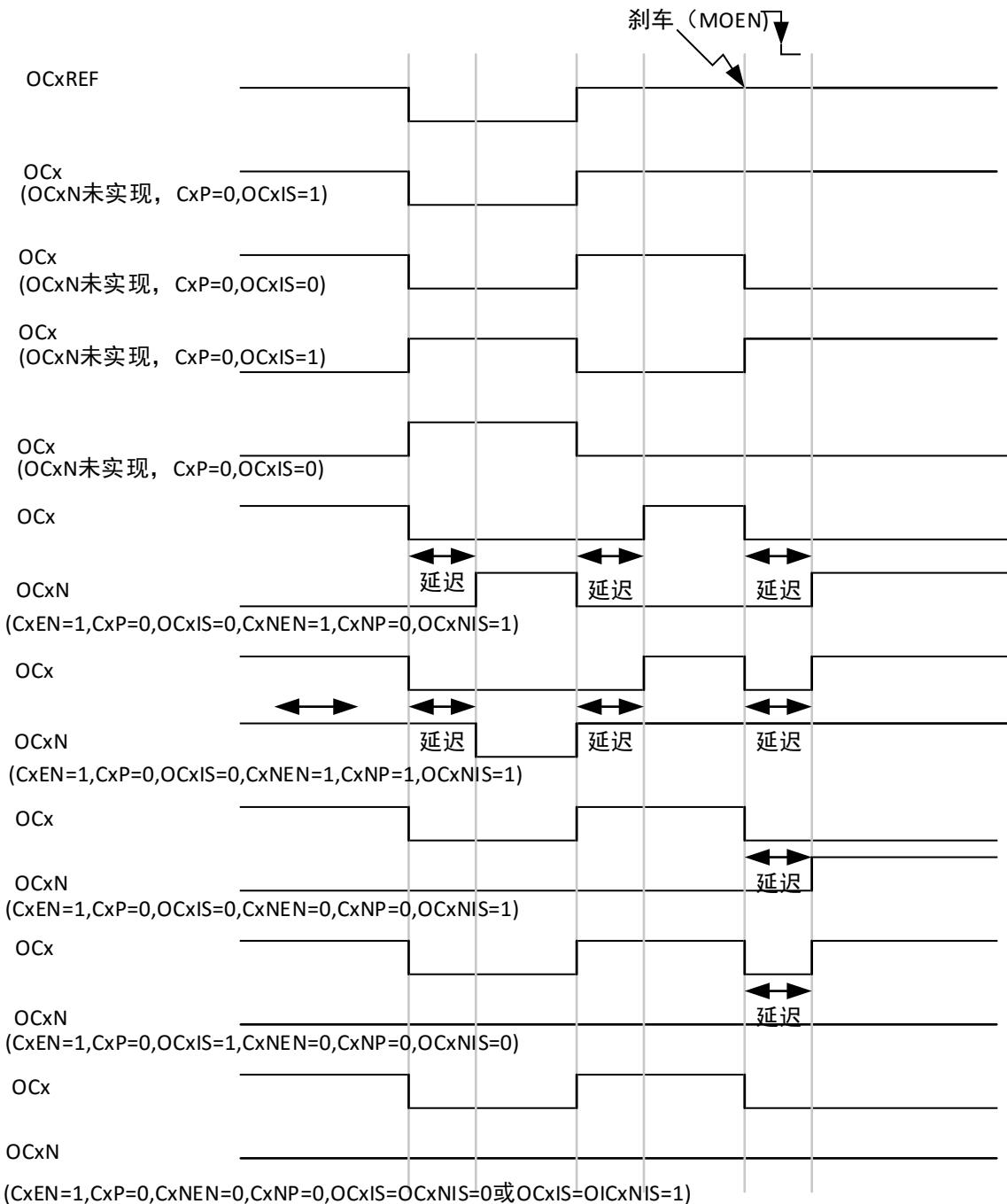
刹车由 **BRK** 输入产生，它的有效极性是可编程的，且由 **TMRx\_BRKDT** 寄存器中的 **BRKEN** 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度， $OCx/OCxN$  极性和被禁止的状态， $OCxMODE$  配置，刹车使能和极性）。

用户可以通过 **TMRx\_BRKDT** 寄存器中的 **LOCKC** 位，从三级保护中选择一种，参看 [10.6.4.18 节 TMR1 刹车和死区寄存器 \(TMRx\\_BRKDT\)](#)。在 MCU 复位后 **LOCKC** 位只能被修改一次。

下图显示响应刹车的输出实例。

图表 10-157 响应刹车的输出



### 10.6.3.13 在外部事件时清除 OCxREF 信号

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

该功能只能用于输出比较和 **PWM** 模式，而不能用于强置模式。

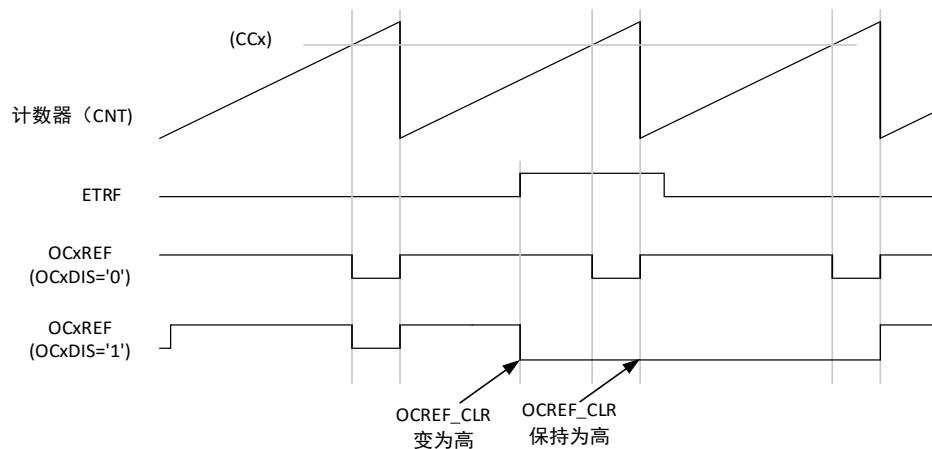
例如，**OCxREF** 信号可以联到一个比较器的输出，用于控制电流。这时，**ETR** 必须配置如下：

1. 外部触发预分频器必须处于关闭： **TMRx\_SMC** 寄存器中的 **ETD[1: 0]=00**。

2. 必须禁止外部时钟模式 2: TMRx\_SMC 寄存器中的 ECLKEN=0。
3. 外部触发极性 (ETRGP) 和外部触发滤波器 (ETDF) 可以根据需要配置。

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

图表 10-158 清除 TMRx 的 OCxREF



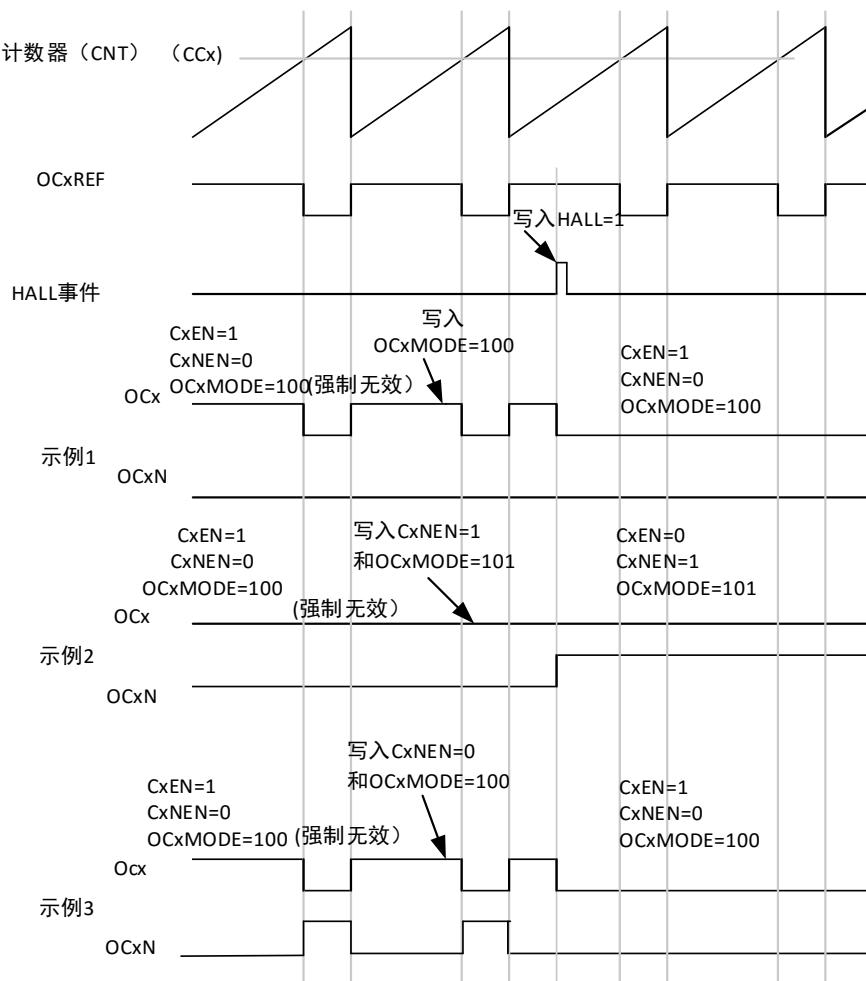
#### 10.6.3.14 产生六步PWM输出

当在一个通道上需要互补输出时, 预装载位有 OCxMODE、CxEN 和 CxNEN。在发生 HALL 换相事件时, 这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置, 并在同一个时刻同时修改所有通道的配置。HALL 可以通过设置 TMRx\_EVEG 寄存器的 HALL 位由软件产生, 或在 TRGI 上升沿由硬件产生。

当发生 HALL 事件时会设置一个标志位 (TMRx\_STS 寄存器中的 HALLIF 位), 这时如果已设置了 TMRx\_DIE 寄存器的 HALLIE 位, 则产生一个中断; 如果已设置了 TMRx\_DIE 寄存器的 HALLDE 位, 则产生一个 DMA 请求。

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

图表 10-159 产生六步PWM, 使用HALL的例子 (OSIMR=1)



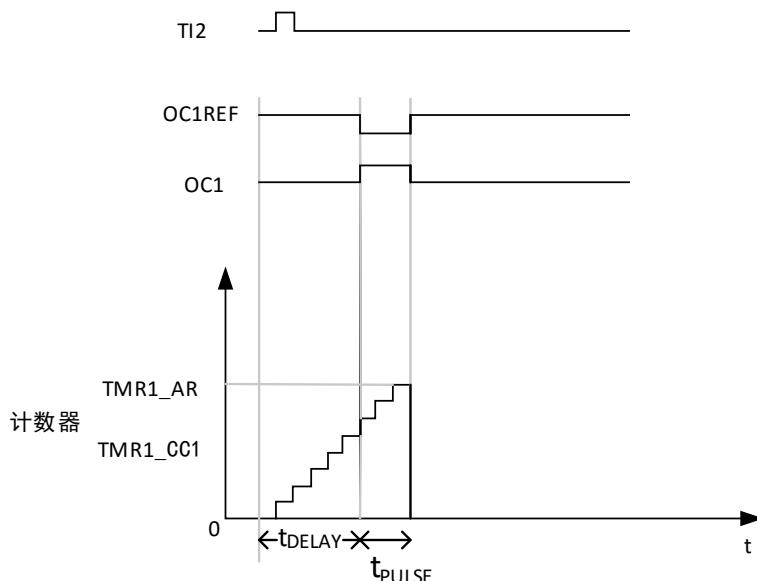
### 10.6.3.15 单脉冲模式

单脉冲模式（OPMODE）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TMRx\_CTRL1 寄存器中的 OPMODE 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

- 向上计数方式：计数器  $CNT < CCx \leq AR$  （特别地， $0 < CCx$ ）；
- 向下计数方式：计数器  $CNT > CCx$ 。

图表 10-160 单脉冲模式的例子



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

假定 TI2FP2 作为触发 1：

- 置 TMRx\_CCM1 寄存器中的 C2SEL=01，把 TI2FP2 映像到 TI2。
- 置 TMRx\_CCE 寄存器中的 C2P=0，使 TI2FP2 能够检测上升沿。
- 置 TMRx\_SMC 寄存器中的 TRGSEL=110，TI2FP2 作为从模式控制器的触发 (TRGI)。
- 置 TMRx\_SMC 寄存器中的 SMSEL=110 (触发模式)，TI2FP2 被用来启动计数器。

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

- $t_{DELAY}$  由 TMRx\_CC1 寄存器中的值定义。
- $t_{PULSE}$  由自动装载值和比较值之间的差值定义 ( $TMRx\_AR - TMRx\_CC1$ )。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TMRx\_CCM1 寄存器的 OC1MODE=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TMRx\_CCM1 中的 OC1PEN=1 和 TMRx\_CTRL1 寄存器中的 ARPEN；然后在 TMRx\_CC1 寄存器中填写比较值，在 TMRx\_AR 寄存器中填写自动装载值，设置 UEVG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，C1P=0。

在这个例子中，TMRx\_CTRL1 寄存器中的 DIR 和 CMSEL 位应该置低。因为只需要一个脉冲，所以必须设置 TMRx\_CTRL1 寄存器中的 OPMODE=1，在下一个更新事件 (当计数器从自动装载值翻转到 0) 时停止计数。

**特殊情况：OCx 快速使能：**

在单脉冲模式下，在  $TIx$  输入脚的边沿检测逻辑设置 CNTEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时  $t_{DELAY}$ 。如果要以最小延时输出波形，可以设置 TMRx\_CCMx 寄存器中的 OCxFEN 位；此时 OCxREF (和 OCx) 直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFEN 只在通道配置为 PWM1 和 PWM2 模式时起作用。

### 10.6.3.16 编码器接口模式

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

通过设置 **TMRx\_CCE** 寄存器中的 C1P 和 C2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 10-11，假定计数器已经启动 (**TMRx\_CTRL1** 寄存器中的 **CNTEN=1**)，则计数器由每次在 **TI1FP1** 或 **TI2FP2** 上的有效跳变驱动。**TI1FP1** 和 **TI2FP2** 是 **TI1** 和 **TI2** 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 **TI1FP1=TI1**, **TI2FP2=TI2**。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 **TMRx\_CTRL1** 寄存器的 **DIR** 位进行相应的设置。不管计数器是依靠 **TI1** 计数、依靠 **TI2** 计数或者同时依靠 **TI1** 和 **TI2** 计数，在任一输入端 (**TI1** 或者 **TI2**) 的跳变都会重新计算 **DIR** 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 **TMRx\_AR** 寄存器的自动装载值之间连续计数（根据方向，或是 0 到 AR 计数，或是 AR 到 0 计数）。所以在开始计数之前必须配置 **TMRx\_AR**；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作正常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

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

表格 10-13 计数方向与编码器信号的关系

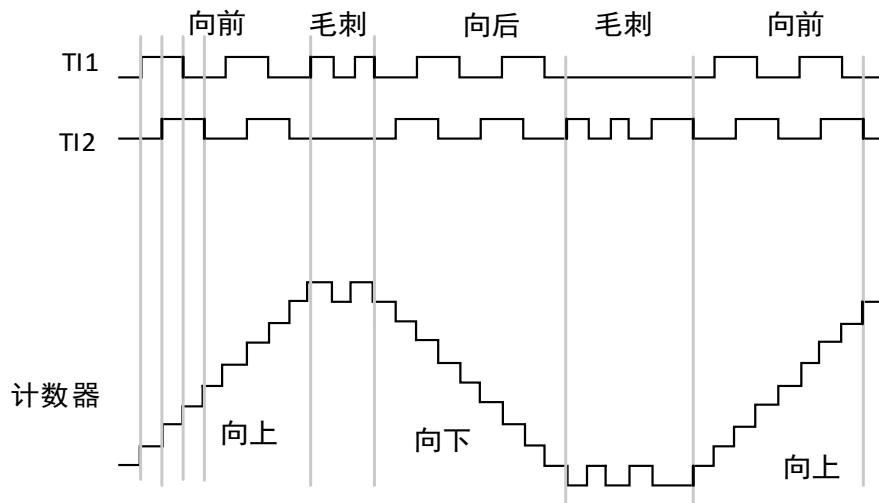
有效边沿	相对信号的电平 ( <b>TI1FP1</b> 对应 <b>TI2</b> , <b>TI2FP2</b> 对应 <b>TI1</b> )	<b>TI1FP1</b> 信号		<b>TI2FP2</b> 信号	
		上升	下降	上升	下降
仅在 <b>TI1</b> 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 <b>TI2</b> 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 <b>TI1</b> 和 <b>TI2</b> 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

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

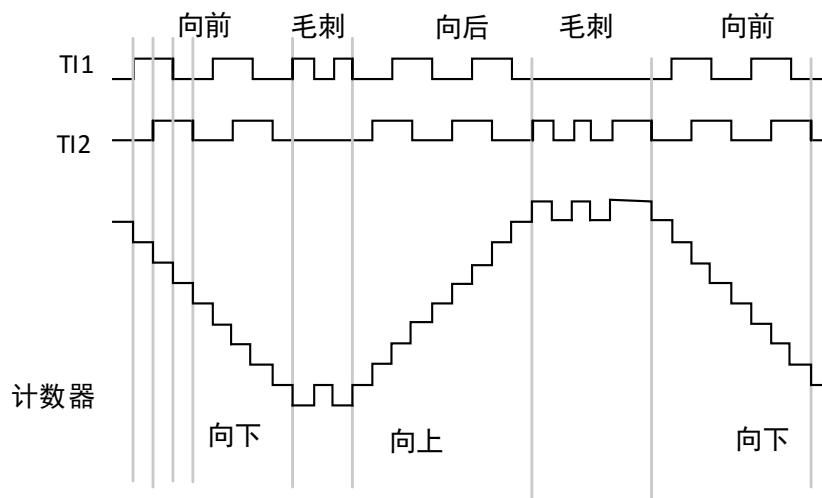
- **C1SEL='01'** （**TMRx\_CCM1** 寄存器， **IC1FP1** 映射到 **TI1**）
- **C2SEL='01'** （**TMRx\_CCM1** 寄存器， **IC2FP2** 映射到 **TI2**）
- **C1P='0'** （**TMRx\_CCE** 寄存器， **IC1FP1** 不反相， **IC1FP1=TI1**）
- **C2P='0'** （**TMRx\_CCE** 寄存器， **IC2FP2** 不反相， **IC2FP2=TI2**）
- **SMSEL='011'** （**TMRx\_SMC** 寄存器， 所有的输入均在上升沿和下降沿有效）.
- **CNTEN='1'** （**TMRx\_CTRL1** 寄存器， 计数器使能）

图表 10-161 编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例（C1P='1'，其他配置与上例相同）。

图表 10-162 IC1FP1反相的编码器接口模式实例



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

### 10.6.3.17 定时器输入异或功能

TMRx\_CTRL2 寄存器中的 TI1SEL 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TMRx\_CH1、TMRx\_CH2 和 TMRx\_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下一节给出了此特性用于连接霍尔传感器的例子。

### 10.6.3.18 与霍尔传感器的接口

使用高级控制定时器（TMR1）产生 PWM 信号驱动马达时，可以用另一个通用 TMRx（TMR2、TMR3、TMR4 或 TMR5）定时器作为“接口定时器”来连接霍尔传感器，见图 10-163，3 个定时器输入脚（CC1、CC2、CC3）通过一个异或门连接到 TI1 输入通道（通过设置 TMRx\_CTRL2 寄存器中的 TI1SEL 位来选择），“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F\_ED。每当 3 个输入之一变化时，计数器重新从 0 开始

计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC（见图 10-146）。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以（通过触发一个 HALL 事件）用于改变高级定时器 TMR1 各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。

因此“接口定时器”通道必须编程为在一个指定的延时（输出比较或 PWM 模式）之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器 TMR1。

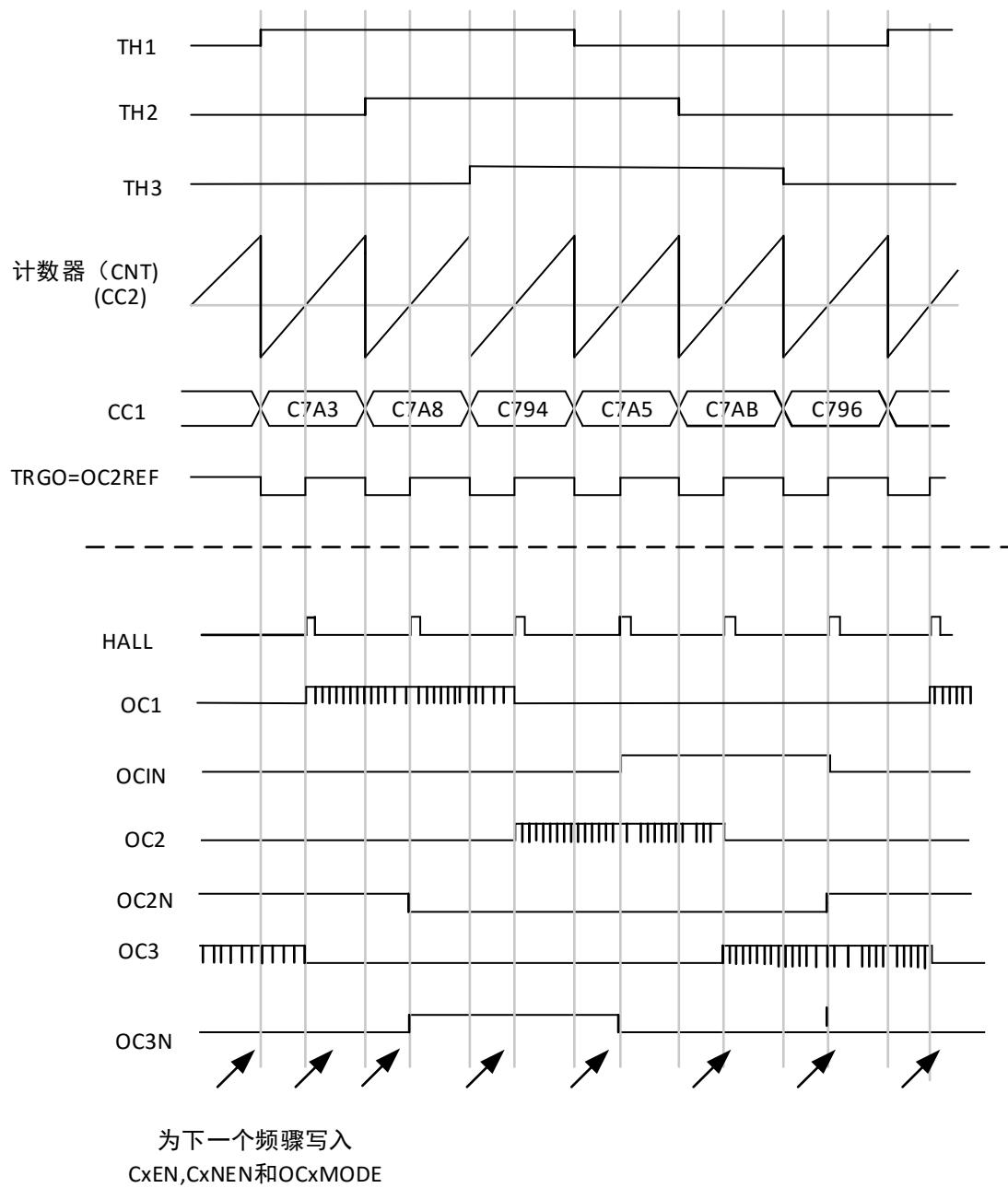
举例：霍尔输入连接到 TMRx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TMRx 的 PWM 配置。

- 置 TMRx\_CTRL2 寄存器的 TI1SEL 位为‘1’，配置三个定时器输入逻辑或到 TI1 输入，
- 时基编程：置 TMRx\_AR 为其最大值（计数器必须通过 TI1 的变化清零）。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
- 设置通道 1 为捕获模式（选中 TRC）：置 TMRx\_CCM1 寄存器中 C1SEL=11，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM2 模式，并具有要求的延时：置 TMRx\_CCM1 寄存器中的 OC2MODE=111 和 C2SEL=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TMRx\_CTRL2 寄存器中的 MMSEL=101。

在高级控制寄存器 TMR1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的（TMRx\_CTRL2 寄存器中 CPC=1），同时触发输入控制 HALL 事件（TMRx\_CTRL2 寄存器中 CUSEL=1）。在一次 HALL 事件后，写入下一步的 PWM 控制位（CxEN、OCxMODE），这可以在处理 OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例

图表 10-163 霍尔传感器接口的实例



### 10.6.3.19 TMRx定时器和外部触发的同步

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

#### 从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化。同时，如果 TMRx\_CTRL1 寄存器的 UVERS 位为低，还产生一个更新事件 UEV。然后所有的预装载寄存器(TMRx\_AR , TMRx\_CCx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道1以检测TI1的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持IC1DF=0000）。触发操作中不使用捕获预分频器，所以不需要配置。C1SEL位只选择输入捕获源，即TMRx\_CCM1寄存器中C1SEL=01。置TMRx\_CCE寄存器中C1P=0以确定极性（只检测上升沿）。
- 置TMRx\_SMC寄存器中SMSEL=100，配置定时器为复位模式；置TMRx\_SMC寄存器

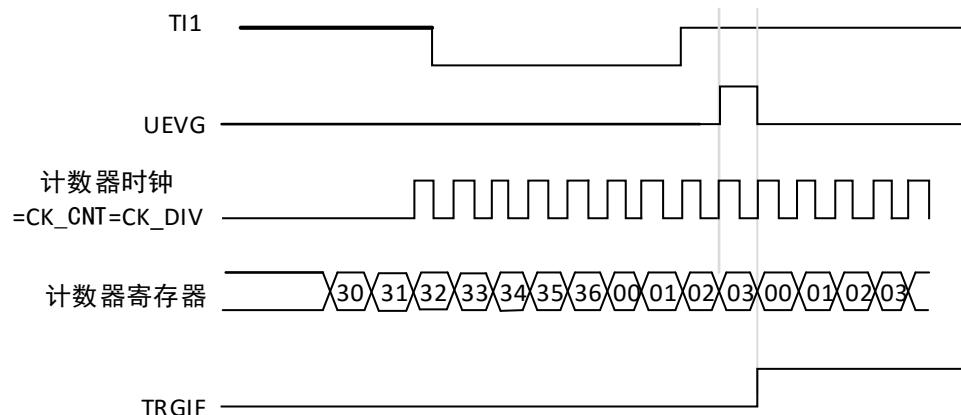
中  $\text{TRGSEL}=101$ , 选择 TI1 作为输入源。

- 置  $\text{TMRx\_CTRL1}$  寄存器中  $\text{CNTEN}=1$ , 启动计数器。

计数器开始依据内部时钟计数, 然后正常运转直到 TI1 出现一个上升沿; 此时, 计数器被清零然后从 0 重新开始计数。同时, 触发标志 ( $\text{TMRx\_STS}$  寄存器中的  $\text{TRGIF}$  位) 被设置, 根据  $\text{TMRx\_DIE}$  寄存器中  $\text{TRGIE}$  (中断使能) 位和  $\text{TRGDE}$  (DMA 使能) 位的设置, 产生一个中断请求或一个 DMA 请求。

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

图表 10-164 复位模式下的控制电路



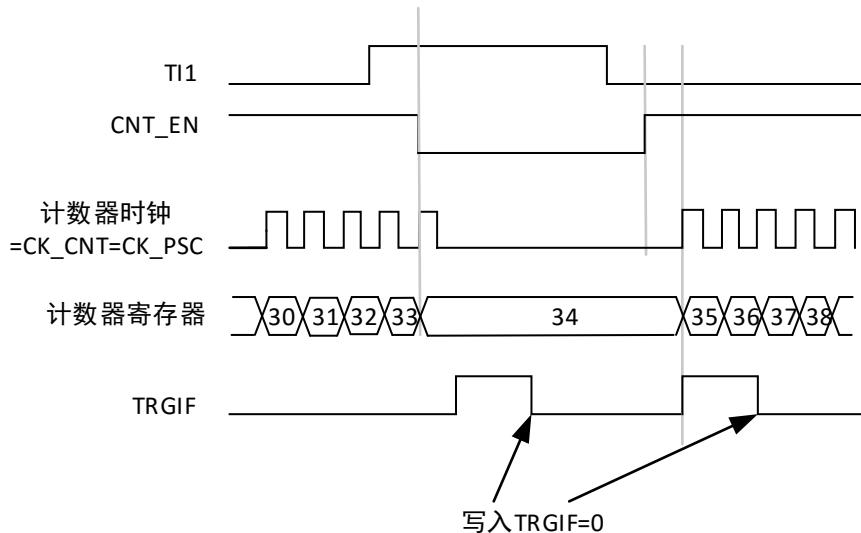
#### 从模式: 门控模式

按照选中的输入端电平使能计数器。在如下的例子中, 计数器只在 TI1 为低时向上计数:

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽 (本例中, 不需要滤波, 所以保持  $\text{IC1DF}=0000$ )。触发操作中不使用捕获预分频器, 所以不需要配置。 $\text{C1SEL}$  位用于选择输入捕获源, 置  $\text{TMRx\_CCM1}$  寄存器中  $\text{C1SEL}=01$ 。置  $\text{TMRx\_CCE}$  寄存器中  $\text{C1P}=1$  以确定极性 (只检测低电平)。
- 置  $\text{TMRx\_SMC}$  寄存器中  $\text{SMSEL}=101$ , 配置定时器为门控模式; 置  $\text{TMRx\_SMC}$  寄存器中  $\text{TRGSEL}=101$ , 选择 TI1 作为输入源。
- 置  $\text{TMRx\_CTRL1}$  寄存器中  $\text{CNTEN}=1$ , 启动计数器。在门控模式下, 如果  $\text{CNTEN}=0$ , 则计数器不能启动, 不论触发输入电平如何。只要 TI1 为低, 计数器开始依据内部时钟计数, 一旦 TI1 变高则停止计数。当计数器开始或停止时都设置  $\text{TMRx\_STS}$  中的  $\text{TRGIF}$  标志。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

图表 10-165 门控模式下的控制电路



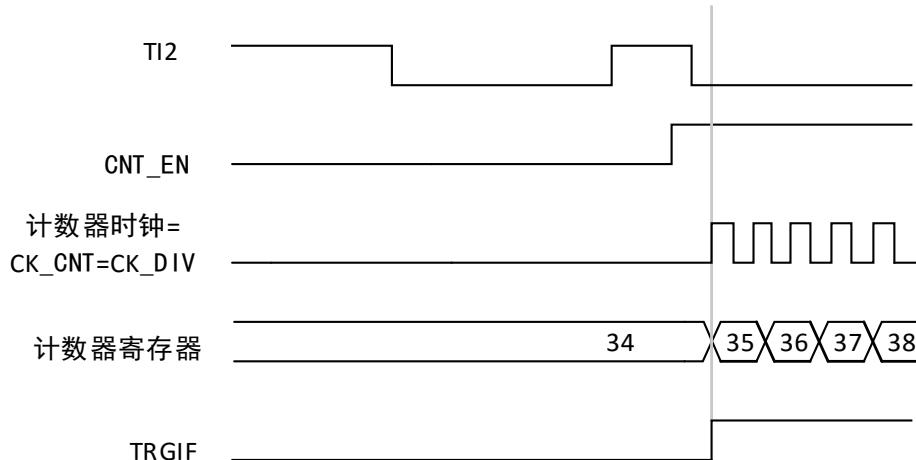
### 从模式：触发模式

输入端上选中的事件使能计数器。在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道2检测TI2的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持IC2DF=0000）。触发操作中不使用捕获预分频器，不需要配置。C2SEL位只用于选择输入捕获源，置TMRx\_CCM1寄存器中C2SEL=01。置TMRx\_CCE寄存器中C2P=0以确定极性（只检测低电平）。
- 置TMRx\_SMC寄存器中SMSEL=110，配置定时器为触发模式；置TMRx\_SMC寄存器中TRGSEL=110，选择TI2作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TRGIF 标志。TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

图表 10-166 触发器模式下的控制电路



### 从模式：外部时钟模式 2 + 触发模式

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

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

1. 通过 TMRx\_SMC 寄存器配置外部触发输入电路：
  - ETDF=0000: 没有滤波
  - ETD=00: 不用预分频器
  - ETRGP=0: 检测 ETR 的上升沿，置 ECLKEN=1 使能外部时钟模式 2。

2. 按如下配置通道 1，检测 TI 的上升沿：

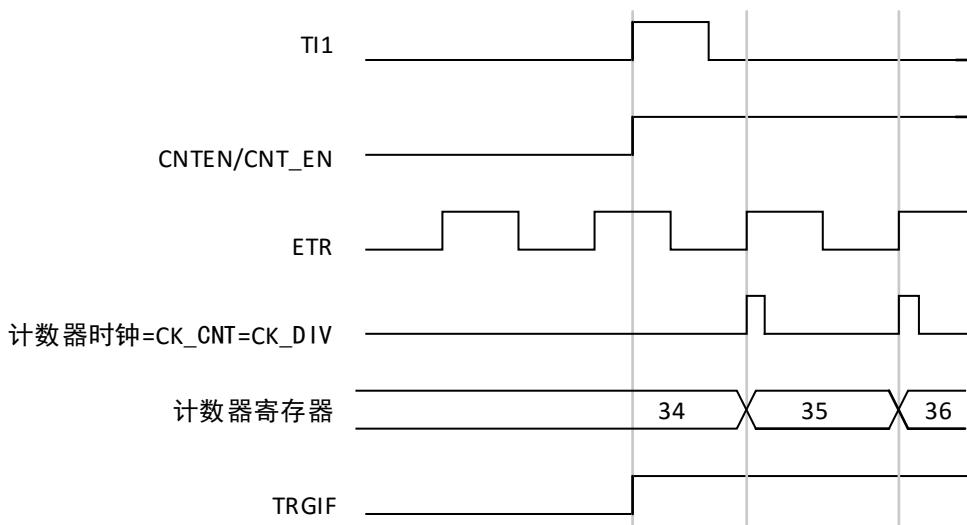
- IC1DF=0000：没有滤波
- 触发操作中不使用捕获预分频器，不需要配置
- 置 TMRx\_CCM1 寄存器中 C1SEL=01，选择输入捕获源
- 置 TMRx\_CCE 寄存器中 C1P=0 以确定极性（只检测上升沿）

3. 置 TMRx\_SMC 寄存器中 SMSEL=110，配置定时器为触发模式。置 TMRx\_SMC 寄存器中 TRGSEL=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TRGIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

图表 10-167 外部时钟模式2+触发模式下的控制电路



### 10.6.3.20 定时器同步

所有 TMR 定时器在内部相连，用于定时器同步或链接。详见 [10.2.3.15 节](#)。

### 10.6.3.21 调试模式

当微控制器进入调试模式时（Cortex™-M4 核心停止），根据 DBG 模块中 DBG\_TMRx\_STOP 的设置，TMRx 计数器可以或者继续正常操作，或者停止。详见 [第 18.2.2 节](#)。

## 10.6.4 TMR1寄存器描述

可以用半字（16位）或字（32位）的方式操作这些外设寄存器。

下表中将 TMR1 的所有寄存器映射到一个 16 位可寻址（编址）空间

表格 10-14 TMR1寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

0x00	TMRx_CTRL1	保留											
	复位值												
0x04	TMRx_CTRL2	保留											
	复位值												
0x08	TMRx_SMC	保留											
	复位值												
0x0C	TMRx_DIE	保留											
	复位值												
0x10	TMRx_STS	保留											
	复位值												
0x14	TMRx_EVEG	保留											
	复位值												
0x18	TMRx_CCM1 输出 比较模式	保留											
	复位值												
0x1C	TMRx_CCM1 输入 捕获模式	保留											
	复位值												
0x1C	TMRx_CCM2 输出 比较模式	保留											
	复位值												
0x1C	TMRx_CCM2 输入 捕获模式	保留											
	复位值												

0x20	TMRx_CCE	保留	C4P	C4EN	C3NP	C3NEN	C3P	C3EN	C2NP	C2NEN	C2P	C2NEN	C1NP	C1NEN	C1P	C1EN											
			0	0	0	0	0	0	0	0	0	0	0	0	0	0											
0x24	TMRx_CNT	保留	CNT[15: 0]						0	0	0	0	0	0	0	0											
			0	0	0	0	0	0	0	0	0	0	0	0	0	0											
0x28	TMRx_DIV	保留	DIV[15: 0]						0	0	0	0	0	0	0	0											
			0	0	0	0	0	0	0	0	0	0	0	0	0	0											
0x2C	TMRx_AR	保留	AR[15: 0]						0	0	0	0	0	0	0	0											
			0	0	0	0	0	0	0	0	0	0	0	0	0	0											
0x30	TMRx_RC	保留							RC[7: 0]						0	0	0	0	0	0	0	0					
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
0x34	TMRx_CC1	保留	CC1[15: 0]						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x38	TMRx_CC2	保留	CC2[15: 0]						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x3C	TMRx_CC3	保留	CC3[15: 0]						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x40	TMRx_CC4	保留	CC4[15: 0]						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x44	TMRx_BRKDT	保留	MOEN	AOEN	BRKP	BRKEN	OSIMR	OSIMI	LOCKC[1: 0]		DTGS[7: 0]																
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x48	TMRx_DMAC	保留	DBLEN[4: 0]						保留	ADDR[4: 0]						保留	0						0	0	0	0	0
			0	0	0	0	0	0		0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	
0x4C	TMRx_DMABA	保留	DMABA[15: 0]						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 10.6.4.1 TMR1控制寄存器1 (TMRx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				CLKDIV [1: 0]		ARP EN	CMSEL [1: 0]		DIR	OPMODE		UVERS	UEVDIS		CNT EN
res				rw		rw	rw		rw	rw		rw	rw		rw

位 15: 10	保留, 始终读为 0。
位 9: 8	<b>CLKDIV[1: 0]:</b> 时钟分频因子 (Clock division) 这 2 位定义在定时器时钟 (CK_INT) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。 00: $t_{DTS} = t_{CK\_INT}$ 01: $t_{DTS} = 2 \times t_{CK\_INT}$ 10: $t_{DTS} = 4 \times t_{CK\_INT}$ 11: 保留, 不要使用这个配置

位 7	<b>ARPEN:</b> 自动重装载预装载允许位 (Auto-reload preload enable) 0: TMRx_AR 寄存器没有缓冲; 1: TMRx_AR 寄存器被装入缓冲器。
位 6: 5	<b>CSEL[1: 0]:</b> 选择中央对齐模式 (Center-aligned mode selection) 00: 边沿对齐模式。计数器依据方向位 (DIR) 向上或向下计数。 01: 中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 只在计数器向下计数时被设置。 10: 中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 只在计数器向上计数时被设置。 11: 中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道 (TMRx_CCMx 寄存器中 CxSEL=00) 的输出比较中断标志位, 在计数器向上和向下计数时均被设置。 <i>注: 在计数器开启时 (CNTEN=1), 不允许从边沿对齐模式转换到中央对齐模式。</i>
位 4	<b>DIR:</b> 方向 (Direction) 0: 计数器向上计数; 1: 计数器向下计数。 <i>注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。</i>
位 3	<b>OPMODE:</b> 单脉冲模式 (One pulse mode) 0: 在发生更新事件时, 计数器不停止; 1: 在发生下一次更新事件 (清除 CNTEN 位) 时, 计数器停止。
位 2	<b>UVERS:</b> 更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
位 1	<b>UEVDIS:</b> 禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: - 计数器溢出/下溢 - 设置 UEVG 位 - 从模式控制器产生的更新具有缓存的寄存器被装入它们的预装载值。(更新影子寄存器) 1: 禁止 UEV。不产生更新事件, 影子寄存器 (AR、DIV、CCx) 保持它们的值。如果设置了 UEVG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化。
位 0	<b>CNTEN:</b> 使能计数器 (Counter enable) 0: 禁止计数器; 1: 使能计数器。 <i>注: 在软件设置了 CNTEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CNTEN 位。</i>

#### 10.6.4.2 TMR1控制寄存器2 (TMRx\_CTRL2)

偏移地址: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留	OC4 IS	OC3 NIS	OC3 IS	OC2 NIS	OC2 IS	OC1 NIS	OC1 IS	TI1S EL	MMSEL[2: 0]	CDS EL	CUS EL	保留	CPC			
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	res	rw		
位 15		保留, 始终读为 0。														

位 14	<b>OC4IS:</b> 输出空闲状态 4 (OC4 输出)。参见 OC1IS 位。
位 13	<b>OC3NIS:</b> 输出空闲状态 3 (OC3N 输出)。参见 OC1NIS 位。
位 12	<b>OC3IS:</b> 输出空闲状态 3 (OC3 输出)。参见 OC1IS 位。
位 11	<b>OC2NIS:</b> 输出空闲状态 2 (OC2N 输出)。参见 OC1NIS 位。
位 10	<b>OC2IS:</b> 输出空闲状态 2 (OC2 输出)。参见 OC1IS 位。
位 9	<b>OC1NIS:</b> 输出空闲状态 1 (OC1N 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 死区后 OC1N=0; 1: 当 MOEN=0 时, 死区后 OC1N=1。 注: 已经设置了LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。
位 8	<b>OC1IS:</b> 输出空闲状态 1 (OC1 输出) (Output Idle state 1) 0: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=0。 1: 当 MOEN=0 时, 如果实现了 OC1N, 则死区后 OC1=1。 注: 已经设置了LOCK (TMRx_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。
位 7	<b>TI1SEL:</b> TI1 选择 (TI1 selection) 0: TMRx_CH1 引脚连到 TI1 输入。 1: TMRx_CH1、TMRx_CH2 和 TMRx_CH3 引脚经异或后连到 TI1 输入。
位 6: 4	<b>MMSEL[2: 0]:</b> 主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 – TMRx_EVEG 寄存器的 UEVG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式), 则 TRGO 上的信号相对实际的复位会有一个延迟。 001: 使能–计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CNTEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时, TRGO 上会有一个延迟, 除非选择了主/从模式 (见 TMRx_SMC 寄存器中 MSMODE 位的描述)。 010: 更新 – 更新事件被选为触发输入 (TRGO)。例如, 一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 – 在发生一次捕获或一次比较成功时, 当要设置 C1IF 标志时 (即使它已经为高), 触发输出送出一个正脉冲 (TRGO)。 100: 比较 – OC1REF 信号被用于作为触发输出 (TRGO)。 101: 比较 – OC2REF 信号被用于作为触发输出 (TRGO)。 110: 比较 – OC3REF 信号被用于作为触发输出 (TRGO)。 111: 比较 – OC4REF 信号被用于作为触发输出 (TRGO)。
位 3	<b>CDSEL:</b> 捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0: 当发生 CCx 事件时, 送出 CCx 的 DMA 请求; 1: 当发生更新事件时, 送出 CCx 的 DMA 请求。
位 2	<b>CUSEL:</b> 捕获/比较控制更新选择 (Capture/compare control update selection) 0: 如果捕获/比较控制位是预装载的 (CPC=1), 只能通过设置 HALL 位更新它们; 1: 如果捕获/比较控制位是预装载的 (CPC=1), 可以通过设置 HALL 位或 TRGI 上的一个上升沿更新它们。 注: 该位只对具有互补输出的通道起作用。
位 1	保留, 始终读为 0。
位 0	<b>CPC:</b> 捕获/比较预装载控制位 (Capture/compare preloaded control) 0: CxEN, CxNEN 和 OCxMODE 位不是预装载的; 1: CxEN, CxNEN 和 OCxMODE 位是预装载的; 设置该位后, 它们只在发生一个 HALL 事件的时候 (设置了 HALL 位或检测到 TRGI 的上升沿, 依据 CUSEL 位) 被更新。 注: 该位只对具有互补输出的通道起作用。

### 10.6.4.3 TMR1从模式控制寄存器 (TMRx\_SMC)

偏移地址: 0x08

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETR GP	ECL KEN	ETD[1: 0]		ETDF[3: 0]		MSM ODE	TRGSEL[2: 0]	保留	SMSEL[2: 0]						
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	res	RW	RW	RW

位 15	<b>ETRGP:</b> 外部触发极性 (External trigger polarity) 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0: ETR 不反相, 高电平或上升沿有效; 1: ETR 被反相, 低电平或下降沿有效。
位 14	<b>ECLKEN:</b> 外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0: 禁止外部时钟模式 2; 1: 使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1: 设置 ECLKEN 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF (SMSEL=111 和 TRGSEL=111) 具有相同功效。 注 2: 下述从模式可以与外部时钟模式 2 同时使用: 复位模式, 门控模式和触发模式; 但是, 这时 TRGI 不能连到 ETRF (TRGSEL 位不能是'111')。 注 3: 外部时钟模式 1 和外部时钟模式 2 同时被使能时, 外部时钟的输入是 ETRF。
位 13: 12	<b>ETD[1: 0]:</b> 外部触发预分频 (External trigger divide) 外部触发信号 ETRP 的频率必须最多是 TMRxCLK 频率的 1/4。当输入较快的外部时钟时, 可以使用预分频降低 ETRP 的频率。 00: 关闭预分频; 01: ETRP 频率除以 2; 10: ETRP 频率除以 4; 11: ETRP 频率除以 8。
位 11: 8	<b>ETDF[3: 0]:</b> 外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上, 数字滤波器是一个事件计数器, 它记录到 N 个事件后会产生一个输出的跳变。 0000: 无滤波器, 以 $f_{DTS}$ 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , $N=2$ 0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , $N=4$ 0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , $N=8$ 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , $N=6$ 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , $N=8$ 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , $N=6$ 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , $N=8$ 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , $N=6$ 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , $N=8$ 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , $N=5$ 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , $N=6$ 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , $N=8$ 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , $N=5$ 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , $N=6$ 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , $N=8$
位 7	<b>MSMODE:</b> 主/从模式 (Master/slave mode) 0: 无作用; 1: 触发输入 (TRGI) 上的事件被延迟了, 以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
位 6: 4	<b>TRGSEL[2: 0]:</b> 触发选择 (Trigger selection) 这 3 位选择用于同步计数器的触发输入。 000: 内部触发 0 (ITR0) 100: TI1 的边沿检测器 (TI1F_ED) 001: 内部触发 1 (ITR1) 101: 滤波后的定时器输入 1 (TI1FP1) 010: 内部触发 2 (ITR2) 110: 滤波后的定时器输入 2 (TI2FP2) 011: 内部触发 3 (ITR3) 111: 外部触发输入 (ETRF) 更多有关 ITRx 的细节, 参见表 10-15。 注: 这些位只能在未用到 (如 SMSEL=000) 时被改变, 以避免在改变时产生错误的边沿检测。
位 3	保留, 始终读为 0。

位 2: 0	<b>SMSEL[2: 0]:</b> 从模式选择 (Slave mode selection)
	当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关 (见输入控制寄存器和控制寄存器的说明)
	000: 关闭从模式 – 如果 CNTEN=1, 则预分频器直接由内部时钟驱动。
	001: 编码器模式 1 – 根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿向上/下计数。
	010: 编码器模式 2 – 根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿向上/下计数。
	011: 编码器模式 3 – 根据另一个信号的输入电平, 计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。
	100: 复位模式 – 选中的触发输入 (TRGI) 的上升沿重新初始化计数器, 并且产生一个更新寄存器的信号。
	101: 门控模式 – 当触发输入 (TRGI) 为高时, 计数器的时钟开启。一旦触发输入变为低, 则计数器停止 (但不复位)。计数器的启动和停止都是受控的。
	110: 触发模式 – 计数器在触发输入 TRGI 的上升沿启动 (但不复位), 只有计数器的启动是受控的。
	111: 外部时钟模式 1 – 选中的触发输入 (TRGI) 的上升沿驱动计数器。
	注: 如果 TI1F_EN 被选为触发输入 (TRGSEL=100) 时, 不要使用门控模式。这是因为, TI1F_ED 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。

表格 10-15 TMRx 内部触发连接

从定时器	ITR0 (TRGSEL=000)	ITR1 (TRGSEL=001)	ITR2 (TRGSEL=010)	ITR3 (TRGSEL=011)
TMR1	TMR15_TRGO	0	TMR3_TRGO	0

#### 10.6.4.4 TMR1 DMA/中断使能寄存器 (TMRx\_DIE)

偏移地址: 0x0C

复位值: 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	TRG DE	HAL LDE	C4DE	C3D E	C2D E	C1D E	UEV DE	BRKI E	TRGI E	HALL IE	C4IE	C3IE	C2IE	C1IE	UEVI E	
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	保留, 始终读为 0。
位 14	<b>TRGDE:</b> 允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求; 1: 允许触发 DMA 请求。
位 13	<b>HALLDE:</b> 允许 HALL 的 DMA 请求 (HALL DMA request enable) 0: 禁止 HALL 的 DMA 请求; 1: 允许 HALL 的 DMA 请求。
位 12	<b>C4DE:</b> 允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较 4 的 DMA 请求; 1: 允许捕获/比较 4 的 DMA 请求。
位 11	<b>C3DE:</b> 允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较 3 的 DMA 请求; 1: 允许捕获/比较 3 的 DMA 请求。
位 10	<b>C2DE:</b> 允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求; 1: 允许捕获/比较 2 的 DMA 请求。
位 9	<b>C1DE:</b> 允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求; 1: 允许捕获/比较 1 的 DMA 请求。

位 8	<b>UEVDE:</b> 允许更新的 DMA 请求 (Update DMA request enable) 0: 禁止更新的 DMA 请求; 1: 允许更新的 DMA 请求。
位 7	<b>BRKIE:</b> 允许刹车中断 (Break interrupt enable) 0: 禁止刹车中断; 1: 允许刹车中断。
位 6	<b>TRGIE:</b> 触发中断使能 (Trigger interrupt enable) 0: 禁止触发中断; 1: 使能触发中断。
位 5	<b>HALLIE:</b> 允许 HALL 中断 (HALL interrupt enable) 0: 禁止 HALL 中断; 1: 允许 HALL 中断。
位 4	<b>C4IE:</b> 允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0: 禁止捕获/比较 4 中断; 1: 允许捕获/比较 4 中断。
位 3	<b>C3IE:</b> 允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较 3 中断; 1: 允许捕获/比较 3 中断。
位 2	<b>C2IE:</b> 允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断; 1: 允许捕获/比较 2 中断。
位 1	<b>C1IE:</b> 允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断; 1: 允许捕获/比较 1 中断。
位 0	<b>UEVIE:</b> 允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

#### 10.6.4.5 TMR1状态寄存器 (TMRx\_STS)

偏移地址: 0x10

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	C4OF F	C3OF F	C2OF F	C1OF F	保留	BRK IF	TRG IF	HAL LIF	C4I F	C3I F	C2I F	C1I F	UEV IF		
res	rw	rw	rw	rw	res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 13	保留, 始终读为 0。
位 12	<b>C4OF:</b> 捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 C1OF 描述。
位 11	<b>C3OF:</b> 捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 C1OF 描述。
位 10	<b>C2OF:</b> 捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 C1OF 描述。
位 9	<b>C1OF:</b> 捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 计数器的值被捕获到 TMRx_CC1 寄存器时, C1IF 的状态已经为'1'。
位 8	保留, 始终读为 0。

位 7	<b>BRKIF:</b> 刹车中断标记 (Break interrupt flag) 一旦刹车输入有效, 由硬件对该位置'1'。如果刹车输入无效, 则该位可由软件清'0'。 0: 无刹车事件产生; 1: 刹车输入上检测到有效电平。
位 6	<b>TRGIF:</b> 触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置'1'。它由软件清'0'。 0: 无触发器事件产生; 1: 触发中断等待响应。
位 5	<b>HALLIF:</b> HALL 中断标记 (HALL interrupt flag) 一旦产生 HALL 事件 (当捕获/比较控制位: CxEN、CxNEN、OCxMODE 已被更新) 该位由硬件置'1'。它由软件清'0'。 0: 无 HALL 事件产生; 1: HALL 中断等待响应。
位 4	<b>C4IF:</b> 捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 C1IF 描述。
位 3	<b>C3IF:</b> 捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 C1IF 描述。
位 2	<b>C2IF:</b> 捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 C1IF 描述。
位 1	<b>C1IF:</b> 捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) <b>如果通道 CC1 配置为输出模式:</b> 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外 (参考 TMRx_CTRL1 寄存器的 CMSEL 位)。它由软件清'0'。 0: 无匹配发生; 1: TMRx_CNT 的值与 TMRx_CC1 的值匹配。 当 TMRx_CC1 的内容大于 TMRx_AR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, C1IF 位变高。 <b>如果通道 CC1 配置为输入模式:</b> 当捕获事件发生时该位由硬件置'1', 它由软件清'0'或通过读 TMRx_CC1 清'0'。 0: 无输入捕获产生; 1: 计数器值已被捕获 (拷贝) 至 TMRx_CC1 (在 IC1 上检测到与所选极性相同的边沿)。
位 0	<b>UEVIF:</b> 更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': - 若 TMRx_CTRL1 寄存器的 UEVDIS=0, 当重复计数器数值上溢或下溢时 (重复计数器=0 时产生更新事件)。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当设置 TMRx_EVEG 寄存器的 UEVG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 - 若 TMRx_CTRL1 寄存器的 UVERS=0、UEVDIS=0, 当计数器 CNT 被触发事件重新初始化时。 (参考 <a href="#">10.6.4.3: TMR1 从模式控制寄存器 (TMRx_SMC)</a> )。

#### 10.6.4.6 TMR1事件产生寄存器 (TMRx\_EVEG)

偏移地址: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								BRK G	TRG G	HALL G	C4G	C3G	C2G	C1G	UEV G
					保留		res		rw	rw	rw	rw	rw	rw	rw

位 15: 8      保留, 始终读为 0。

位 7	<b>BRKG:</b> 产生刹车事件 (Break generation) 该位由软件置'1', 用于产生一个刹车事件, 由硬件自动清'0'。 0: 无动作; 1: 产生一个刹车事件。此时 MOEN=0、BRKIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。
位 6	<b>TRGG:</b> 产生触发事件 (Trigger generation) 该位由软件置'1', 用于产生一个触发事件, 由硬件自动清'0'。 0: 无动作; 1: TMRx_STS 寄存器的 TRGIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。
位 5	<b>HALLG:</b> 捕获/比较事件, 产生控制更新 (Capture/Compare control update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 当 CPC=1, 允许更新 CxEN、CxNEN、OCxMODE 位。 注: 该位只对拥有互补输出的通道有效。
位 4	<b>C4G:</b> 产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 C1G 描述。
位 3	<b>C3G:</b> 产生捕获/比较 3 事件 (Capture/Compare 3 generation) 参考 C1G 描述。
位 2	<b>C2G:</b> 产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 C1G 描述。
位 1	<b>C1G:</b> 产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1', 用于产生一个捕获/比较事件, 由硬件自动清'0'。 0: 无动作; 1: 在通道 CC1 上产生一个捕获/比较事件: 若通道 CC1 配置为输出: 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。 若通道 CC1 配置为输入: 当前的计数器值被捕获至 TMRx_CC1 寄存器; 设置 C1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 C1IF 已经为 1, 则设置 C1OF=1。
位 0	<b>UEVG:</b> 产生更新事件 (Update generation) 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0 (向上计数) 则计数器被清'0'; 若 DIR=1 (向下计数) 则取计数器取 TMRx_AR 的值。

#### 10.6.4.7 TMR1捕获/比较模式寄存器1 (TMRx\_CCM1)

偏移地址: 0x18

复位值: 0x0000

通道可用于输入 (捕获模式) 或输出 (比较模式), 通道的方向由相应的 CxSEL 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能, ICxx 描述了通道在输入模式下的功能。因此必须注意, 同一个位在输出模式和输入模式下的功能是不同的。

##### 输出比较模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2 DIS	OC2MODE[2: 0]	OC2 PEN	OC2 FEN	C2SEL[1: 0]	OC1 DIS	OC1MODE[2: 0]	OC1 PEN	OC1 FEN							C1SEL[1: 0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>OC2DIS:</b> 输出比较 2 清 0 使能 (Output Compare 2 clear enable)
------	--

位 14: 12	<b>OC2MODE[2: 0]:</b> 输出比较 2 模式 (Output Compare 2 mode)
位 11	<b>OC2PEN:</b> 输出比较 2 预装载使能 (Output Compare 2 preload enable)
位 10	<b>OC2FEN:</b> 输出比较 2 快速使能 (Output Compare 2 fast enable)
位 9: 8	<p><b>C2SEL[1: 0]:</b> 捕获/比较 2 选择。 (Capture/Compare 2 selection) 该位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC2 通道被配置为输出； 01: CC2 通道被配置为输入， IC2 映射在 TI2 上； 10: CC2 通道被配置为输入， IC2 映射在 TI1 上； 11: CC2 通道被配置为输入， IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C2SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C2NEN=0) 才是可写的。</p>
位 7	<p><b>OC1DIS:</b> 输出比较 1 清'0'使能 (Output Compare 1 clear enable) 0: OC1REF 不受 ETRF 输入的影响； 1: 一旦检测到 ETRF 输入高电平，清除 OC1REF=0。</p>
位 6: 4	<p><b>OC1MODE[2: 0]:</b> 输出比较 1 模式 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。 OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 C1P、C1NP 位。 000: 冻结。输出比较寄存器 TMRx_CC1 与计数器 TMRx_CNT 间的比较对 OC1REF 不起作用； 001: 匹配时设置通道 1 为有效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 TMRx_CNT 的值与捕获 / 比较寄存器 1 (TMRx_CC1) 相同时，强制 OC1REF 为低。 011: 翻转。当 TMRx_CC1=TMRx_CNT 时，翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1— 在向上计数时，一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为无效电平 (OC1REF=0)，否则为有效电平 (OC1REF=1)。 111: PWM 模式 2— 在向上计数时，一旦 TMRx_CNT&lt;TMRx_CC1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TMRx_CNT&gt;TMRx_CC1 时通道 1 为有效电平，否则为无效电平。 注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。 注 2: 在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。</p>
位 3	<p><b>OC1PEN:</b> 输出比较 1 预装载使能 (Output Compare 1 preload enable) 0: 禁止 TMRx_CC1 寄存器的预装载功能，可随时写入 TMRx_CC1 寄存器，并且新写入的数值立即起作用。 1: 开启 TMRx_CC1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TMRx_CC1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1: 一旦 LOCK 级别设为 3 (TMRx_BRKDT 寄存器中的 LOCKC 位) 并且 C1SEL=00 (该通道配置成输出) 则该位不能被修改。 注 2: 仅在单脉冲模式下 (TMRx_CTRL1 寄存器的 OPMODE=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。</p>
位 2	<p><b>OC1FEN:</b> 输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快 CC 输出对触发输入事件的响应。 0: 根据计数器与 CC1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OC1FEN 只在通道被配置成 PWM1 或 PWM2 模式时起作用。</p>

位 1: 0	<b>C1SEL[1: 0]:</b> 捕获/比较 1 选择。 (Capture/Compare 1 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=0) 才是可写的。														

### 输入捕获模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2DF[3: 0]			IC2DIV[1: 0]		C2SEL[1: 0]		IC1DF[3: 0]			IC1DIV[1: 0]		C1SEL[1: 0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 12	<b>IC2DF[3: 0]:</b> 输入捕获 2 滤波器 (Input capture 2 filter)														
位 11: 10	<b>IC2DIV[1: 0]:</b> 输入/捕获 2 预分频器 (Input capture 2 prescaler)														
位 9: 8	<b>C2SEL[1: 0]:</b> 捕获/比较 2 选择 (Capture/Compare 2 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C2SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C2EN=0) 才是可写的。														
位 7: 4	<b>IC1DF [3: 0]:</b> 输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成, 它记录到 N 个事件后会产生一个输出的跳变: 0000: 无滤波器, 以 $f_{DTS}$ 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK\_INT}$ , N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$ , N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$ , N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$ , N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$ , N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$ , N=8														
位 3: 2	<b>IC1DIV[1: 0]:</b> 输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入 (IC1) 的预分频系数.一旦 C1EN=0 (TMRx_CCE 寄存器中), 则预分频器复位。 00: 无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获; 01: 每 2 个事件触发一次捕获; 10: 每 4 个事件触发一次捕获; 11: 每 8 个事件触发一次捕获。														
位 1: 0	<b>C1SEL[1: 0]:</b> 捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C1SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C1EN=0) 才是可写的。														

### 10.6.4.8 TMR1捕获/比较模式寄存器2 (TMRx\_CCM2)

偏移地址: 0x1C

复位值: 0x0000

参看以上 CCM1 寄存器描述

#### 输出比较模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4 DIS	OC4MODE[2: 0]	OC4 PE	OC4 FEN	C4SEL[1: 0]	OC3 DIS	OC3MODE[2: 0]	OC3 PEN	OC3 FEN	C3SEL[1: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>OC4DIS:</b> 输出比较 4 清 0 使能 (Output compare 4 clear enable)
位 14: 12	<b>OC4MODE[2: 0]:</b> 输出比较 4 模式 (Output compare 4 mode)
位 11	<b>OC4PEN:</b> 输出比较 4 预装载使能 (Output compare 4 preload enable)
位 10	<b>OC4FEN:</b> 输出比较 4 快速使能 (Output compare 4 fast enable)
位 9: 8	<b>C4SEL[1: 0]:</b> 捕获/比较 4 选择 (Capture/Compare 4 selection) 该 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC4 通道被配置为输出; 01: CC4 通道被配置为输入, IC4 映射在 TI4 上; 10: CC4 通道被配置为输入, IC4 映射在 TI3 上; 11: CC4 通道被配置为输入, IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: CC4S 仅在通道关闭时 (TMRx_CCE 寄存器的 C4EN=0) 才是可写的。
位 7	<b>OC3DIS:</b> 输出比较 3 清 0 使能 (Output compare 3 clear enable)
位 6: 4	<b>OC3MODE[2: 0]:</b> 输出比较 3 模式 (Output compare 3 mode)
位 3	<b>OC3PEN:</b> 输出比较 3 预装载使能 (Output compare 3 preload enable)
位 2	<b>OC3FEN:</b> 输出比较 3 快速使能 (Output compare 3 fast enable)
位 1: 0	<b>C3SEL[1: 0]:</b> 捕获/比较 3 选择 (Capture/Compare 3 selection) 这 2 位定义通道的方向 (输入/输出), 及输入脚的选择: 00: CC3 通道被配置为输出; 01: CC3 通道被配置为输入, IC3 映射在 TI3 上; 10: CC3 通道被配置为输入, IC3 映射在 TI4 上; 11: CC3 通道被配置为输入, IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C3SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C3EN=0) 才是可写的。

#### 输入捕获模式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC4DF[3: 0]	IC4DIV[1: 0]	C4S[1: 0]		IC3DF[3: 0]	IC3DIV[1: 0]	C3SEL[1: 0]									
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 12	<b>IC4DF[3: 0]:</b> 输入捕获 4 滤波器 (Input capture 4 filter)
位 11: 10	<b>IC4DIV[1: 0]:</b> 输入/捕获 4 预分频器 (Input capture 4 prescaler)

位 9: 8	<b>C4S[1: 0]:</b> 捕获/比较 4 选择 (Capture/Compare 4 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC4 通道被配置为输出； 01: CC4 通道被配置为输入，IC4 映射在 TI4 上； 10: CC4 通道被配置为输入，IC4 映射在 TI3 上； 11: CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: CC4S 仅在通道关闭时 (TMRx_CCE 寄存器的 C4EN=0) 才是可写的。
位 7: 4	<b>IC3DF[3: 0]:</b> 输入捕获 3 滤波器 (Input capture 3 filter)
位 3: 2	<b>IC3DIV[1: 0]:</b> 输入/捕获 3 预分频器 (Input capture 3 prescaler)
位 1: 0	<b>C3SEL[1: 0]:</b> 捕获/比较 3 选择 (Capture/compare 3 selection) 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC3 通道被配置为输出； 01: CC3 通道被配置为输入，IC3 映射在 TI3 上； 10: CC3 通道被配置为输入，IC3 映射在 TI4 上； 11: CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TMRx_SMC 寄存器的 TRGSEL 位选择)。 注: C3SEL 仅在通道关闭时 (TMRx_CCE 寄存器的 C3EN=0) 才是可写的。

#### 10.6.4.9 TMR1捕获/比较使能寄存器 (TMRx\_CCE)

偏移地址: 0x20

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	C4P	C4EN	C3NP	C3NEN	C3P	C3EN	C2NP	C2NEN	C2P	C2NEN	C1NP	C1NEN	C1P	C1EN	
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15: 14	保留，始终读为 0。
位 13	<b>C4P:</b> 输入/捕获 4 输出极性 (Capture/Compare 4 output polarity) 参考 C1P 的描述。
位 12	<b>C4EN:</b> 输入/捕获 4 输出使能 (Capture/Compare 4 output enable) 参考 C1OE 的描述。
位 11	<b>C3NP:</b> 输入/捕获 3 互补输出极性 (Capture/Compare 3 complementary output polarity) 参考 C1NP 的描述。
位 10	<b>C3NEN:</b> 输入/捕获 3 互补输出使能 (Capture/Compare 3 complementary output enable) 参考 C1NEN 的描述。
位 9	<b>C3P:</b> 输入/捕获 3 输出极性 (Capture/Compare 3 output polarity) 参考 C1P 的描述。
位 8	<b>C3EN:</b> 输入/捕获 3 输出使能 (Capture/Compare 3 output enable) 参考 C1OE 的描述。
位 7	<b>C2NP:</b> 输入/捕获 2 互补输出极性 (Capture/Compare 2 complementary output polarity) 参考 C1NP 的描述。
位 6	<b>C2NEN:</b> 输入/捕获 2 互补输出使能 (Capture/Compare 2 complementary output enable) 参考 C1NEN 的描述。
位 5	<b>C2P:</b> 输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 C1P 的描述。

位 4	<b>C2NEN:</b> 输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 C1OE 的描述。
位 3	<b>C1NP:</b> 输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) <b>CC1 通道配置为输出:</b> 0: OC1N 高电平有效; 1: OC1N 低电平有效。 <b>CC1 通道配置为输入:</b> C1NP 结合 C1P 定义 TI1FP1/TI2FP1 极性 (参考 C1P 描述) 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2 且 C1SEL=00 (通道配置为输出) 则该位不能被修改。
位 2	<b>C1NEN:</b> 输入/捕获 1 互补输出使能 (Capture/Compare 1 complementary output enable) 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。 1: 启开— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1EN 位的值。
位 1	<b>C1P:</b> 输入/捕获 1 输出极性 (Capture/Compare 1 output polarity) <b>CC1 通道配置为输出:</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1 通道配置为输入:</b> CC1NP/CC1P 位选择 TI1FP1 和 TI2FP1 的极性信号作为触发或捕获信号。 00: 不反相/上升沿。电路 TIxFP1 上升沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 不反相 (触发中的门控模式)。 01: 反相/下降沿。电路对 TIxFP1 下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 反相 (触发中的门控模式)。 10: 保留。 11: 不反相/双沿。电路同时对 TIxFP1 上升沿和下降沿敏感 (捕获, 触发中的复位, 外部时钟或触发模式), TIxFP1 是不反相 (触发中门控模式)。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 3 或 2, 则该位不能被修改。
位 0	<b>C1EN:</b> 输入/捕获 1 输出使能 (Capture/Compare 1 output enable) <b>CC1 通道配置为输出:</b> 0 : 关闭— OC1 禁止输出, 因此 OC1 的输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1NEN 位的值。 1 : 启开— OC1 信号输出到对应的输出引脚, 其输出电平依赖于 MOEN、OSIMI、OSIMR、OC1IS、OC1NIS 和 C1NEN 位的值。 <b>CC1 通道配置为输入:</b> 该位决定了计数器的值是否能捕获入 TMRx_CC1 寄存器。 0: 捕获禁止; 1: 捕获使能。

表格 10-16 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态 (1)	
MOEN 位	OSIMI 位	OSIMR 位	CxEN 位	CxNEN 位	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	0	1	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		0	1	0	OCxREF+极性, OCx= OCxREF xor CxP, OCx_EN=1	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1

		1	0	0	输出禁止（与定时器断开） OCx=CxP, OCx_EN=0	输出禁止（与定时器断开） OCxN=CxNP, OCxN_EN=0
		1	0	1	关闭状态 (输出使能且为无效电平) OCx=CxP, OCx_EN=1	OCxREF + 极性, OCxN= OCxREF xor CxNP, OCxN_EN=1
		1	1	0	OCxREF + 极性, OCx= OCxREF xor CxP, OCx_EN=1	关闭状态 (输出使能且为无效电平) OCxN=CxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区, OCx_EN=1	OCxREF 反相+极性+死区, OCxN_EN=1
0	0	X	0	0	输出禁止（与定时器断开） 异步地: OCx=CxP , OCx_EN=0 , OCxN=CxNP , OCxN_EN=0;	
	0		0	1	若时钟存在: 经过一个死区时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	
	0		1	0	关闭状态 (输出使能且为无效电平) 异步地: OCx=CxP , OCx_EN=1 , OCxN=CxNP , OCxN_EN=1;	
	0		1	1	若时钟存在: 经过一个死区 时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	
	1		0	0	关闭状态 (输出使能且为无效电平) 异步地: OCx=CxP , OCx_EN=1 , OCxN=CxNP , OCxN_EN=1;	
	1		0	1	若时钟存在: 经过一个死区 时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	
	1		1	0	关闭状态 (输出使能且为无效电平) 异步地: OCx=CxP , OCx_EN=1 , OCxN=CxNP , OCxN_EN=1;	
	1		1	1	若时钟存在: 经过一个死区 时间后 OCx=OC1IS, OCxN=OCxNIS, 假设 OC1IS 与 OCxNIS 并不都对应 OCx 和 OCxN 的有效电平。	

注意： 如果一个通道的 2 个输出都没有使用( $CxEN = CxNEN = 0$ ), 那么  $OC1IS$ ,  $OCxNIS$ ,  $CxP$  和  $CxNP$  都必须清零。

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

#### 10.6.4.10 TMR1计数器 (TMRx\_CNT)

偏移地址: 0x24

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0		<b>CNT[15: 0]:</b> 计数器的值 (Counter value)													

#### 10.6.4.11 TMR1预分频器 (TMRx\_DIV)

偏移地址: 0x28

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0		<b>DIV[15: 0]:</b> 预分频器的值 (Prescaler value) 计数器的时钟频率 ( $CK_{CNT}$ ) 等于 $f_{CK\_DIV} / ( DIV[15: 0]+1 )$ 。 DIV 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TMR_EVEG 的 UEVG 位清'0'或被工作在复位模式的从控制器清'0'。													

#### 10.6.4.12 TMR1自动重装载寄存器（TMRx\_AR）

偏移地址: 0x2C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AR[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>AR[15: 0]:</b> 自动重装载的值（Prescaler value） AR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 10.6.3.1 节：有关 AR 的更新和动作。当自动重装载的值为空时，计数器不工作。														

#### 10.6.4.13 TMR1重复计数寄存器（TMRx\_RC）

偏移地址: 0x30

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										RC[7: 0]					
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 8	保留，始终读为 0。														
位 7: 0	<b>RC[7: 0]:</b> 重复计数器的值（Repetition counter value） 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）。如果允许产生更新中断，则会同时影响产生更新中断的速率。每次向下计数器 RC_CNT 达到 0，会产生一个更新事件并且计数器 RC_CNT 重新从 RC 值开始计数。由于 RC_CNT 只有在周期更新事件 U_RC 发生时才重载 RC 值，因此对 TMRx_RC 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(RC+1) 对应着： — 在边沿对齐模式下，PWM 周期的数目； — 在中心对称模式下，PWM 半周期的数目；														

#### 10.6.4.14 TMR1捕获/比较寄存器 1 (TMRx\_CC1)

偏移地址: 0x34

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC1[15: 0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 0	<b>CC1[15: 0]:</b> 捕获/比较通道 1 的值（Capture/Compare 1 value） 若 CC1 通道配置为输出： CC1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。如果在 TMRx_CCM1 寄存器（OC1PEN 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CC1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。														

#### 10.6.4.15 TMR1捕获/比较寄存器2（TMRx\_CC2）

偏移地址: 0x38

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC2[15: 0]															

位 15: 0	<b>CC2[15: 0]:</b> 捕获/比较通道 2 的值 (Capture/Compare 2 value)
	<p>若 CC2 通道配置为输出: CC2 包含了装入当前捕获/比较 2 寄存器的值 (预装载值)。如果在 TMRx_CCM2 寄存器 (OC2PEN 位) 中未选择预装载特性, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 2 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC2 端口上产生输出信号。</p> <p>若 CC2 通道配置为输入: CC2 包含了由上一次输入捕获 2 事件 (IC2) 传输的计数器值。</p>

#### 10.6.4.16 TMR1捕获/比较寄存器3（TMRx\_CC3）

偏移地址: 0x3C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC3[15: 0]															

位 15: 0	<b>CC3[15: 0]:</b> 捕获/比较通道 3 的值 (Capture/Compare 3 value)
	<p>若 CC3 通道配置为输出: CC3 包含了装入当前捕获/比较 3 寄存器的值 (预装载值)。</p> <p>如果在 TMRx_CCM3 寄存器 (OC3PEN 位) 中未选择预装载特性, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 3 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC3 端口上产生输出信号。</p> <p>若 CC3 通道配置为输入: CC3 包含了由上一次输入捕获 3 事件 (IC3) 传输的计数器值。</p>

#### 10.6.4.17 TMR1捕获/比较寄存器4（TMRx\_CC4）

偏移地址: 0x40

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4[15: 0]															

位 15: 0	<p><b>CC4[15: 0]:</b> 捕获/比较通道 4 的值 (Capture/Compare 4 value)            若 CC4 通道配置为输出:            CC4 包含了装入当前捕获/比较 4 寄存器的值 (预装载值)。            如果在 TMRx_CCM4 寄存器 (OC4PEN 位) 中未选择预装载特性, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕获/比较 4 寄存器中。当前捕获/比较寄存器参与同计数器 TMRx_CNT 的比较, 并在 OC4 端口上产生输出信号。            若 CC4 通道配置为输入:            CC4 包含了由上一次输入捕获 4 事件 (IC4) 传输的计数器值。         </p>
---------	--

#### 10.6.4.18 TMR1刹车和死区寄存器 (TMRx\_BRKDT)

偏移地址: 0x44

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MO EN	AO EN	BR KP	BRK EN	OSI MR	OSI MI	LOCKC[1: 0]									DTGS[7: 0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

注意: 根据锁定设置, AOEN、BRKP、BRKEN、OSIMR 和 DTGS[7: 0]位均可被写保护, 有必要在第一次写入 TMRx\_BRKDT 寄存器时对它们进行配置。

位 15	<b>MOEN:</b> 主输出使能 (Main output enable) 一旦刹车输入有效, 该位被硬件异步清'0'。根据 AOEN 位的设置值, 该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 0: 禁止 OC 和 OCN 输出或强制为空闲状态; 1: 如果设置了相应的使能位 (TMRx_CCE 寄存器的 CxEN、CxNEN 位), 则开启 OC 和 OCN 输出。 有关 OC/OCN 使能的细节, 参见 <a href="#">10.6.4.9 节: TMR1 捕获/比较使能寄存器(TMRx_CCE)</a> 。
位 14	<b>AOEN:</b> 自动输出使能 (Automatic output enable) 0: MOEN 只能被软件置'1'; 1: MOEN 能被软件置'1'或在下一个更新事件被自动置'1' (如果刹车输入无效)。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。
位 13	<b>BRKP:</b> 刹车输入极性 (Break polarity) 0: 刹车输入低电平有效; 1: 刹车输入高电平有效。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
位 12	<b>BRKEN:</b> 刹车功能使能 (Break enable) 0: 禁止刹车输入 (BRK 及 CFD 时钟失效事件); 1: 开启刹车输入 (BRK 及 CFD 时钟失效事件)。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为'1', 则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
位 11	<b>OSIMR:</b> 运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOEN=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSIMR 位。 参考 OC/OCN 使能的详细说明 ( <a href="#">10.6.4.9 节: TMR1 捕获/比较使能寄存器(TMRx_CCE)</a> )。 0: 当定时器不工作时, 禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0); 1: 当定时器不工作时, 一旦 CxEN=1 或 CxNEN=1, 首先开启 OC/OCN 并输出无效电平, 然后置 OC/OCN 使能输出信号=1。 注: 一旦 LOCK 级别 (TMRx_BRKDT 寄存器中的 LOCKC 位) 设为 2, 则该位不能被修改。

位 10	<p><b>OSIMI:</b> 空闲模式下“关闭状态”选择（Off-state selection for Idle mode） 该位用于当 MOEN=0 且通道设为输出时。参考 OC/OCN 使能的详细说明（<a href="#">10.6.4.9 节</a>，TMR1 捕获/比较使能寄存器（TMRx_CCE））。 0：当定时器不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）； 1：当定时器不工作时，一旦 CxEN=1 或 CxNEN=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别（TMRx_BRKDT 寄存器中的 LOCKC 位）设为 2，则该位不能被修改。</p>
位 9: 8	<p><b>LOCKC[1: 0]:</b> 锁定设置（LENock configuration） 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TMRx_BRKDT 寄存器的 DTGS、BRKEN、BRKP、AOEN 位和 TMRx_CTRL2 寄存器的 OCXIS/OCxNIS 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CxSEL 位设为输出，CC 极性位是 TMRx_CCE 寄存器的 CxP/CCNxP 位）以及 OSIMR/OSIMI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CxSEL 位设为输出，CC 控制位是 TMRx_CCMx 寄存器的 OCxMODE/OCxPEN 位）； 注：在系统复位后，只能写一次 LOCKC 位，一旦写入 TMRx_BRKDT 寄存器，则其内容冻结直至复位。</p>
位 7: 0	<p><b>DTGS[7: 0]:</b> 死区发生器设置（Dead-time generator setup） 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间： <math>DTGS[7: 5]=0xx \Rightarrow DT=DTGS[7: 0] \times T_{dtg}, \quad T_{dtg} = T_{DTS};</math> <math>DTGS[7: 5]=10x \Rightarrow DT= (64+DTGS[5: 0]) \times T_{dtg}, \quad T_{dtg} = 2 \times T_{DTS};</math> <math>DTGS[7: 5]=110 \Rightarrow DT= (32+DTGS[4: 0]) \times T_{dtg}, \quad T_{dtg} = 8 \times T_{DTS};</math> <math>DTGS[7: 5]=111 \Rightarrow DT= (32+DTGS[4: 0]) \times T_{dtg}, \quad T_{dtg} = 16 \times T_{DTS};</math> 例：若 <math>T_{DTS} = 125\text{ns}</math> (8MHz)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别（TMRx_BRKDT 寄存器中的 LOCKC 位）设为 1、2 或 3，则不能修改这些位。</p>

#### 10.6.4.19 TMR1 DMA控制寄存器（TMRx\_DMAC）

偏移地址：0x48

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	DBLEN[4: 0]				保留	ADDR[4: 0]									
res	rw	rw	rw	rw	rw	res	rw								
位 15: 13	保留，始终读为 0。														

位 12: 8	<p><b>DBLEN[4: 0]: DMA 连续传送长度 (DMA burst length)</b></p> <p>这些位定义了 DMA 在连续模式下的传送长度(当对 TMRx_DMABA 寄存器进行读或写时, 定时器则进行一次连续传送), 即: 定义传输的次数, 传输可以是半字(双字节)或字节:</p> <table border="0"> <tr> <td>00000: 1 次传输</td><td>00001: 2 次传输</td></tr> <tr> <td>00010: 3 次传输</td><td>.....</td></tr> <tr> <td>....</td><td>.....</td></tr> <tr> <td>10001: 18 次传输</td><td></td></tr> </table> <p>例: 我们考虑这样的传输: DBLEN=7, ADDR=TMR2_CTRL1</p> <ul style="list-style-type: none"> <li>- 如果 DBLEN=7, ADDR=TMR2_CTRL1 表示待传输数据的地址, 那么传输的地址由下式给出: (TMRx_CTRL1 的地址) + ADDR + (DMA 索引), 其中 DMA 索引= DBLEN 其中 (TMRx_CTRL1 的地址) + ADDR 再加上 7, 给出了将要写入或者读出数据的地址, 这样数据的传输将发生在从地址 (TMRx_CTRL1 的地址) + ADDR 开始的 7 个寄存器。</li> </ul> <p>根据 DMA 数据长度的设置, 可能发生以下情况:</p> <ul style="list-style-type: none"> <li>- 如果设置数据为半字 (16 位), 那么数据就会传输给全部 7 个寄存器。</li> <li>- 如果设置数据为字节, 数据仍然会传输给全部 7 个寄存器: 第一个寄存器包含第一个 MSB 字节, 第二个寄存器包含第一个 LSB 字节, 以此类推。因此对于定时器, 用户必须指定由 DMA 传输的数据宽度。</li> </ul>	00000: 1 次传输	00001: 2 次传输	00010: 3 次传输	.....	....	.....	10001: 18 次传输	
00000: 1 次传输	00001: 2 次传输								
00010: 3 次传输	.....								
....	.....								
10001: 18 次传输									
位 7: 5	保留, 始终读为 0。								
位 4: 0	<p><b>ADDR[4: 0]: DMA 基地址 (DMA base address)</b></p> <p>这些位定义了 DMA 在连续模式下的基地址 (当对 TMRx_DMAR 寄存器进行读或写时), ADDR 定义为从 TMRx_CTRL1 寄存器所在地址开始的偏移量:</p> <table border="0"> <tr> <td>00000: TMRx_CTRL1,</td> </tr> <tr> <td>00001: TMRx_CTRL2,</td> </tr> <tr> <td>00010: TMRx_SMC,</td> </tr> <tr> <td>.....</td> </tr> </table>	00000: TMRx_CTRL1,	00001: TMRx_CTRL2,	00010: TMRx_SMC,	.....				
00000: TMRx_CTRL1,									
00001: TMRx_CTRL2,									
00010: TMRx_SMC,									
.....									

#### 10.6.4.20 TMR1连续模式的DMA地址 (TMRx\_DMABA)

偏移地址: 0x4C

复位值: 0x0000

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

位 15: 0	<p><b>DMABA[15: 0]: DMA 连续传送寄存器 (DMA register for burst accesses)</b></p> <p>对 TMRx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作:</p> <p>TMRx_CTRL1 地址 + ADDR + DMA 索引, 其中:</p> <ul style="list-style-type: none"> <li>“TMRx_CTRL1 地址”是控制寄存器 1 (TMRx_CTRL1) 所在的地址;</li> <li>“ADDR”是 TMRx_DMAR 寄存器中定义的基地址;</li> <li>“DMA 索引”是由 DMA 自动控制的偏移量, 它取决于 TMRx_DMAR 寄存器中定义的 DBLEN。</li> </ul>
---------	--

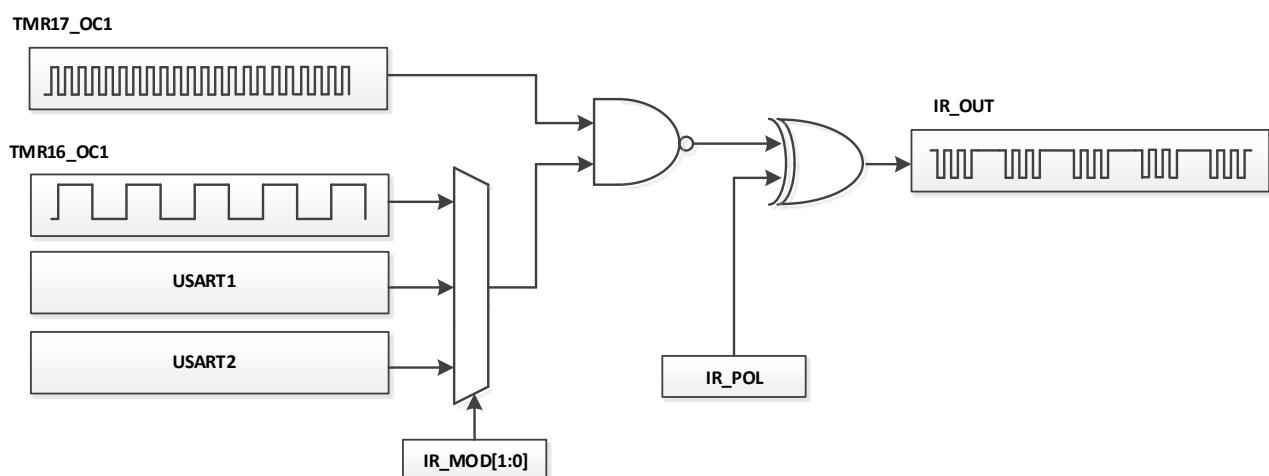
## 11 红外线接口 (IRTMR)

红外线接口用来控制发射红外光的 LED，该 LED 可用来发射红外数据来实现红外遥控。

TMR16\_OC1, USART1, USART2 和 TMR17\_OC1 在内部连接，通过 TMR16\_OC1/USART1/USART2（通过 SYSCFG\_CFGR1 寄存器中的 IR\_MOD[1: 0]位选择）来产生低频调制包络信号，TMR17\_OC1 产生高频载波信号，来产生需要的红外控制信号，通过极性控制位可使红外输出反相（SYSCFG\_CFGR1 寄存器中的 IR\_POL 位）。

使用红外线接口时需要配置 PB9 或者 PA13 为复用模式，并使能 PB9 或者 PA13。

图表 11-1 红外线内部连接示意图



# 12 看门狗 (WDG)

## 12.1 窗口看门狗 (WWDG)

### 12.1.1 WWDG简介

AT32F421 内置两个看门狗，提供了更高的安全性、时间的精确性和使用的灵活性。两个看门狗设备（独立看门狗和窗口看门狗）可用来检测和解决由软件错误引起的故障；当计数器达到给定的超时值时，触发一个中断（仅适用于窗口型看门狗）或产生系统复位。

独立看门狗 (IWDG) 由专用的低速时钟 (LSI) 驱动，即使主时钟发生故障它也仍然有效。窗口看门狗由从 APB1 时钟分频后得到的时钟驱动，通过可配置的时间窗口来检测应用程序非正常的过迟或过早的操作。

窗口看门狗通常被用来监测，由外部干扰或不可预见的逻辑条件造成应用程序背离正常的运行序列而产生的软件故障。除非递减计数器的值在 CNTR6 位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个 MCU 复位。在递减计数器达到窗口寄存器数值之前，如果 7 位的递减计数器数值（在控制寄存器中）被刷新，那么也将产生一个 MCU 复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。

IWDG 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。WWDG 最适合那些要求看门狗在精确计时窗口起作用的应用程序。

关于独立看门狗的详情，请参看第 12.2 节。

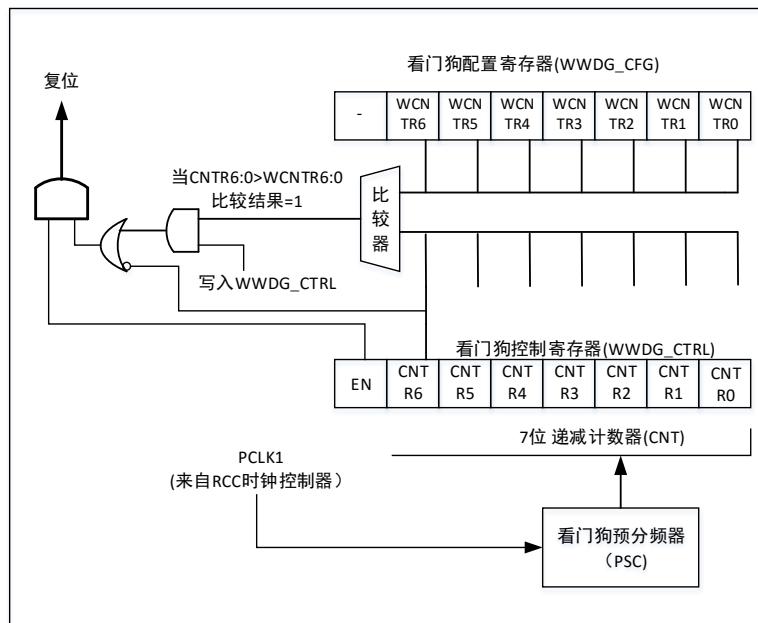
### 12.1.2 WWDG主要特性

- 可编程的自由运行递减计数器
- 条件复位
  - 当递减计数器的值小于 0x40，（若看门狗被启动）则产生复位。
  - 当递减计数器在窗口外被重新装载，（若看门狗被启动）则产生复位。
- 如果启动了看门狗并且允许中断，当递减计数器等于 0x40 时产生早期唤醒中断 (EWIEN)，它可以被用于重装载计数器以避免 WWDG 复位。

### 12.1.3 WWDG功能描述

如果看门狗被启动 (WWDG\_CTRL 寄存器中的 EN 位被置‘1’)，并且当 7 位 (CNTR[6: 0]) 递减计数器从 0x40 翻转到 0x3F (CNTR6 位清零) 时，则产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。

图表 12-1 看门狗框图



应用程序在正常运行过程中必须定期地写入 WWDG\_CTRL 寄存器以防止 MCU 发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在 WWDG\_CTRL 寄存器中的数值必须在 0xFF 和 0xC0 之间：

- 启动看门狗

在系统复位后，看门狗总是处于关闭状态，设置 WWDG\_CTRL 寄存器的 EN 位能够开启看门狗，随后它不能再被关闭，除非发生复位。

- 控制递减计数器

递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时，CNTR6 位必须被设置，以防止立即产生一个复位。CNTR[5: 0] 位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG\_CTRL 寄存器时，预分频值是未知的。配置寄存器 (WWDG\_CFG) 中包含窗口的上限值：要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载，[图 12-2](#) 描述了窗口寄存器的工作过程。另一个重装载计数器的方法是利用早期唤醒中断 (EWIEN)。设置 WWDG\_CFG 寄存器中的 EWIEN 位开启该中断。当递减计数器到达 0x40 时，则产生此中断，相应的中断服务程序 (ISTS) 可以用来加载计数器以防止 WWDG 复位。在 WWDG\_STS 寄存器中写 ‘0’ 可以清除该中断。

注意：可以用 CNTR6 位产生一个软件复位（设置 EN 位为 ‘1’，CNTR6 位为 ‘0’）。

#### 12.1.4 如何编写看门狗超时程序

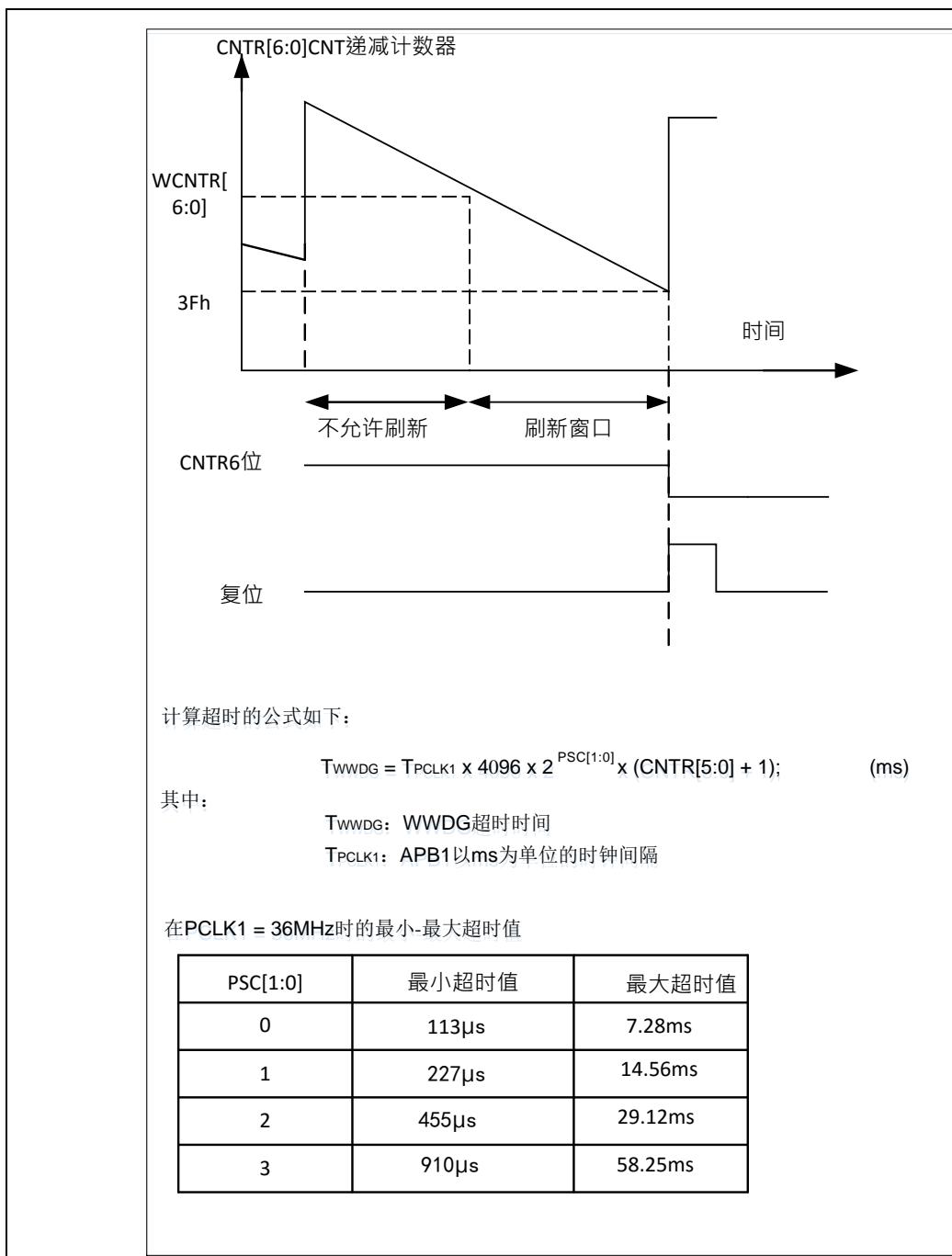
可以使用 [图 12-2](#) 提供的公式计算窗口看门狗的超时时间。

---

**警告：**当写入 WWDG\_CTRL 寄存器时，始终置 CNTR6 位为 ‘1’ 以避免立即产生一个复位。

---

图表 12-2 窗口看门狗时序图



## 12.1.5 调试模式

当微控制器进入调试模式时（Cortex™-M4 核心停止），根据调试模块中的 **DBG\_WWDG\_STOP** 配置位的状态，WWDG 的计数器能够继续工作或停止。详见第 18.2.2 节。

## 12.1.6 寄存器描述

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

表格 12-1 WWDG 寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

0x00	WWDG_CTL RL	保留	EN	CNTR[6: 0]							
				0	1	1	1	1	1	1	1
0x04	WWDG_CFG G	保留	EWIEN PSC[1: 0]	WCNTR[6: 0]							
				0	0	0	1	1	1	1	1
0x08	WWDG_STS	保留	EWIF 0								

### 12.1.6.1 控制寄存器 (WWDG\_CTRL)

地址偏移量: 0x00

复位值: 0x7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16									
保留																								
res																								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0									
保留								EN	CNTR[6: 0]															
res								rs	rw															
位 31: 8		保留。																						
位 7		<b>EN:</b> 激活位 (Activation bit) 此位由软件置'1'，但仅能由硬件在复位后清'0'。当 EN=1 时，看门狗可以产生复位。 0: 禁止看门狗 1: 启用看门狗																						
位 6: 0		<b>CNTR[6: 0]:</b> 7 位计数器 (MSB 至 LSB) (7-bit counter) 这些位用来存储看门狗的计数器值。每 ( $4096 \times 2^{PSC}$ ) 个 PCLK1 周期减 1。当计数器值从 40h 变为 3Fh 时 (CNTR6 变成 0)，产生看门狗复位。																						

### 12.1.6.2 配置寄存器 (WWDG\_CFG)

地址偏移量: 0x04

复位值: 0x7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
保留																	
res																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留								EWIEN	PSC[1: 0]	WCNTR[6: 0]							
res								rs	rw	rw							

位 31: 8	保留。
位 9	<b>EWIEN:</b> 提前唤醒中断 (Early wakeup interrupt) 此位若置'1', 则当计数器值达到 40h, 即产生中断。此中断只能由硬件在复位后清除。
位 8: 7	<b>PSC[1: 0]:</b> 时基 预分频器的时基可以设置如下: 00: CK 计时器时钟 (PCLK1 除以 4096) 除以 1 01: CK 计时器时钟 (PCLK1 除以 4096) 除以 2 10: CK 计时器时钟 (PCLK1 除以 4096) 除以 4 11: CK 计时器时钟 (PCLK1 除以 4096) 除以 8
位 6: 0	<b>WCNTR[6: 0]:</b> 7 位窗口值 (7-bit window value) 这些位包含了用来与递减计数器进行比较用的窗口值。

### 12.1.6.3 状态寄存器 (WWDG\_STS)

地址偏移量: 0x08

复位值: 0x00



位 31: 1	保留。
位 0	<b>EWIF:</b> 提前唤醒中断标志 (Early wakeup interrupt flag) 当计数器值达到 40h 时, 此位由硬件置'1'。它必须通过软件写'0'来清除。对此位写'1'无效。若中断未被使能, 此位也会被置'1'。

## 12.2 独立看门狗 (IWDG)

### 12.2.1 简介

独立看门狗 (IWDG) 由专用的低速时钟 (LSI) 驱动, 即使主时钟发生故障它也仍然有效。

IWDG 最适合应用于那些需要看门狗作为一个在主程序之外, 能够完全独立工作, 并且对时间精度要求较低的场合。

### 12.2.2 IWDG 主要性能

- 自由运行的递减计数器
- 时钟由独立的RC振荡器提供 (可在停机和待机模式下工作)
- 看门狗被激活后, 则在计数器计数至0x000时产生复位

### 12.2.3 IWDG 功能描述

图 12-3 为独立看门狗模块的功能框图。在键寄存器 (IWDG\_KEY) 中写入 0xCCCC, 开始启用独立看门狗; 此时计数器开始从其复位值 0xFFFF 递减计数。当计数器计数到末尾 0x000 时, 会产生一个复位信号 (IWDG\_RESET)。

无论何时, 只要在键寄存器 IWDG\_KEY 中写入 0xAAAA, IWDG\_RLD 中的值就会被重新加载到计数器, 从而避免产生看门狗复位。

### 12.2.3.1 硬件看门狗

如果用户在选择字节中启用了“硬件看门狗”功能，在系统上电复位后，看门狗会自动开始运行；如果在计数器计数结束前，若软件没有向键寄存器写入相应的值，则系统会产生复位。

### 12.2.3.2 寄存器访问保护

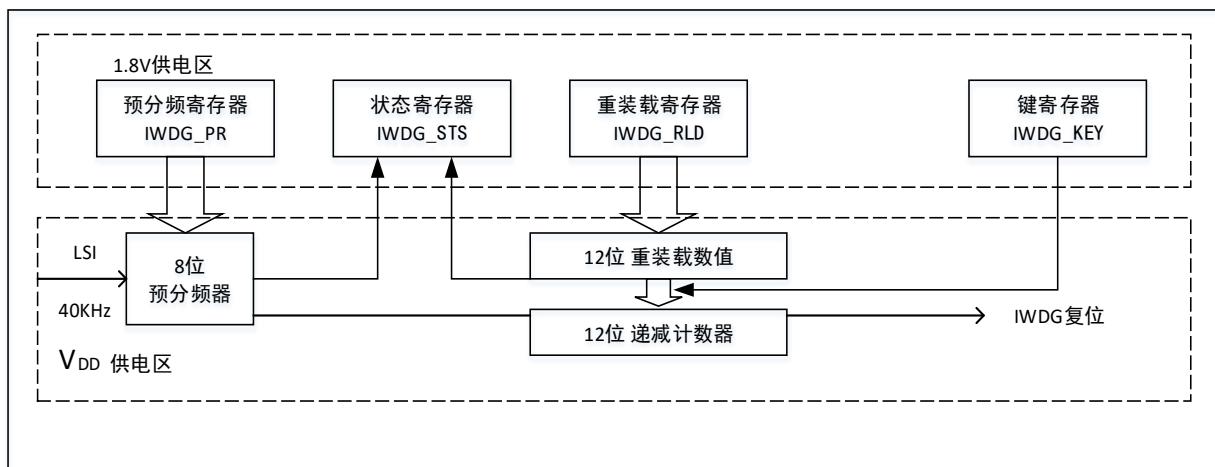
IWDG\_PR 和 IWDG\_RLD 寄存器具有写保护功能。要修改这两个寄存器的值，必须先向 IWDG\_KEY 寄存器中写入 0x5555。以不同的值写入这个寄存器将会打乱操作顺序，寄存器将重新被保护。重装载操作（即写入 0xAAAA）也会启动写保护功能。

状态寄存器指示预分频值和递减计数器是否正在被更新。

### 12.2.3.3 调试模式

当微控制器进入调试模式时（Cortex™-M4 核心停止），根据调试模块中的 DBG\_IWDG\_STOP 配置位的状态，IWDG 的计数器能够继续工作或停止。详见有关调试模块的章节。

图表 12-3 独立看门狗框图



注意：看门狗功能处于 VDD 供电区，即在停机和待机模式时仍能正常工作。

表格 12-2 看门狗超时时间 (40kHz的输入时钟 (LSI))<sup>(1)</sup>

预分频系数	PR[2: 0]位	最短时间 (ms) RLD[11: 0] = 0x000	最长时间 (ms) RLD[11: 0] = 0xFFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 或 7)	6.4	26214.4

注意：这些时间是按照 40kHz 时钟给出。实际上，MCU 内部的 RC 频率会在 30kHz 到 60kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。通过对 LSI 进行校准可获得相对精确的看门狗超时时间。有关 LSI 校准的问题，详见 [3.2.5 节](#)。

## 12.2.4 IWDG 寄存器描述

关于在寄存器描述里面所用到的缩写，详见表 1-4。可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

表格 12-3 IWDG 寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KEY	保留												KEY[15: 0]																			
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x04	IWDG_PR	保留												PR[2: 0]																0	0	0	
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0x08	IWDG_RLD	保留												RLD[11: 0]															RLDF		PSCF		
	复位值	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0		
0x08	IWDG_STS	保留												保留															RLDF		PSCF		
	复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 12.2.4.1 键寄存器 (IWDG\_KEY)

地址偏移: 0x00

复位值: 0x0000 0000 (在待机模式复位)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	保留														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	res														
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
位 31: 16		保留，始终读为 0。																												
位 15: 0		<b>KEY[15: 0]:</b> 键值 (只写寄存器, 读出值为 0x0000) (Key value) 软件必须以一定的间隔写入 0xAAAA, 否则, 当计数器为 0 时, 看门狗会产生复位。 写入 0x5555 表示允许访问 IWDG_PR 和 IWDG_RLD 寄存器。(见 12.2.3.2 节) 写入 0xCCCC, 启动看门狗工作 (若选择了硬件看门狗则不受此命令字限制)。																												

### 12.2.4.2 预分频寄存器 (IWDG\_PR)

地址偏移: 0x04

复位值: 0x0000 0000

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

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

位 31: 3	保留, 始终读为 0。
位 2: 0	<p><b>PR[2: 0]:</b> 预分频因子 (Prescaler divider) 这些位具有写保护设置, 参见 <a href="#">12.2.3.2 节</a>。通过设置这些位来选择计数器时钟的预分频因子。要改变预分频因子, IWDG_STS 寄存器的 PSCF 位必须为 0。</p> <p>000: 预分频因子=4 100: 预分频因子=64 001: 预分频因子=8 101: 预分频因子=128 010: 预分频因子=16 110: 预分频因子=256 011: 预分频因子=32 111: 预分频因子=256</p> <p>注意: 对此寄存器进行读操作, 将从 VDD 电压域返回预分频值。如果写操作正在进行, 则读回的值可能是无效的。因此, 只有当 IWDG_STS 寄存器的 PSCF 位为 0 时, 读出的值才有效。</p>

#### 12.2.4.3 重装载寄存器 (IWDG\_RLD)

地址偏移: 0x08

复位值: 0x0000 0FFF (待机模式时复位)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
保留																
res																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留	RLD[11: 0]															
res	rw															

位 31: 12	保留, 始终读为 0。
位 11: 0	<p><b>RLD[11: 0]:</b> 看门狗计数器重装载值 (WCNTRatchdog counter reload value) 这些位具有写保护功能, 参看 <a href="#">12.2.3.2 节</a>。用于定义看门狗计数器的重装载值, 每当向 IWDG_KEY 寄存器写入 0xAAAA 时, 重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此重装载值和时钟预分频值来计算, 参照表 12-2。</p> <p>只有当 IWDG_STS 寄存器中的 RLDF 位为 0 时, 才能对此寄存器进行修改。</p> <p>注意: 对此寄存器进行读操作, 将从 VDD 电压域返回重装载值。如果写操作正在进行, 则读回的值可能是无效的。因此, 只有当 IWDG_STS 寄存器的 RLDF 位为 0 时, 读出的值才有效。</p>

#### 12.2.4.4 状态寄存器 (IWDG\_STS)

地址偏移: 0x0C

复位值: 0x0000 0000 (待机模式时不复位)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
保留																
res																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留	版本 1.00															

保留		RLDF	PSCF
	res	r	r
位 31: 2	保留。		
位 1	<b>RLDF:</b> 看门狗计数器重装载值更新 (WCNTRatchdog counter reload value update) 此位由硬件置'1'用来指示重装载值的更新正在进行中。当在 VDD 域中的重装载更新结束后，此位由硬件清'0'（最多需 5 个 40kHz 的 RC 周期）。重装载值只有在 RLDF 位被清'0'后才可更新。		
位 0	<b>PSCF:</b> 看门狗预分频值更新 (WCNTRatchdog prescaler value update) 此位由硬件置'1'用来指示预分频值的更新正在进行中。当在 VDD 域中的预分频值更新结束后，此位由硬件清'0'（最多需 5 个 40kHz 的 RC 周期）。预分频值只有在 PSCF 位被清'0'后才可更新。		

**注意：**如果在应用程序中使用了多个重装载值或预分频值，则必须在 **RLDF** 位被清除后才能重新改变预装载值，在 **PSCF** 位被清除后才能重新改变预分频值。然而，在预分频和 / 或重装值更新后，不必等待 **RLDF** 或 **PSCF** 复位，可继续执行下面的代码。（即是在低功耗模式下，此写操作仍会被继续执行完成。）

# 13 实时时钟 (ERTC)

## 13.1 前言

实时时钟 (ERTC) 是一个独立的 BCD 定时器/计数器。ERTC 提供一个日历时钟、一个可编程闹钟中断。两个 32 位寄存器包含二进码十进数格式 (BCD) 的秒、分钟、小时 (12 或 24 小时制)、星期几、日期、月份和年份。此外，还可提供二进制格式的亚秒值。

系统可以自动将月份的天数补偿为 28、29 (闰年)、30 和 31 天。并且还可以进行夏令时补偿。

其它 32 位寄存器还包含可编程的闹钟亚秒、秒、分钟、小时、星期几和日期。

此外，还可以使用数字校准功能对晶振精度的偏差进行补偿。

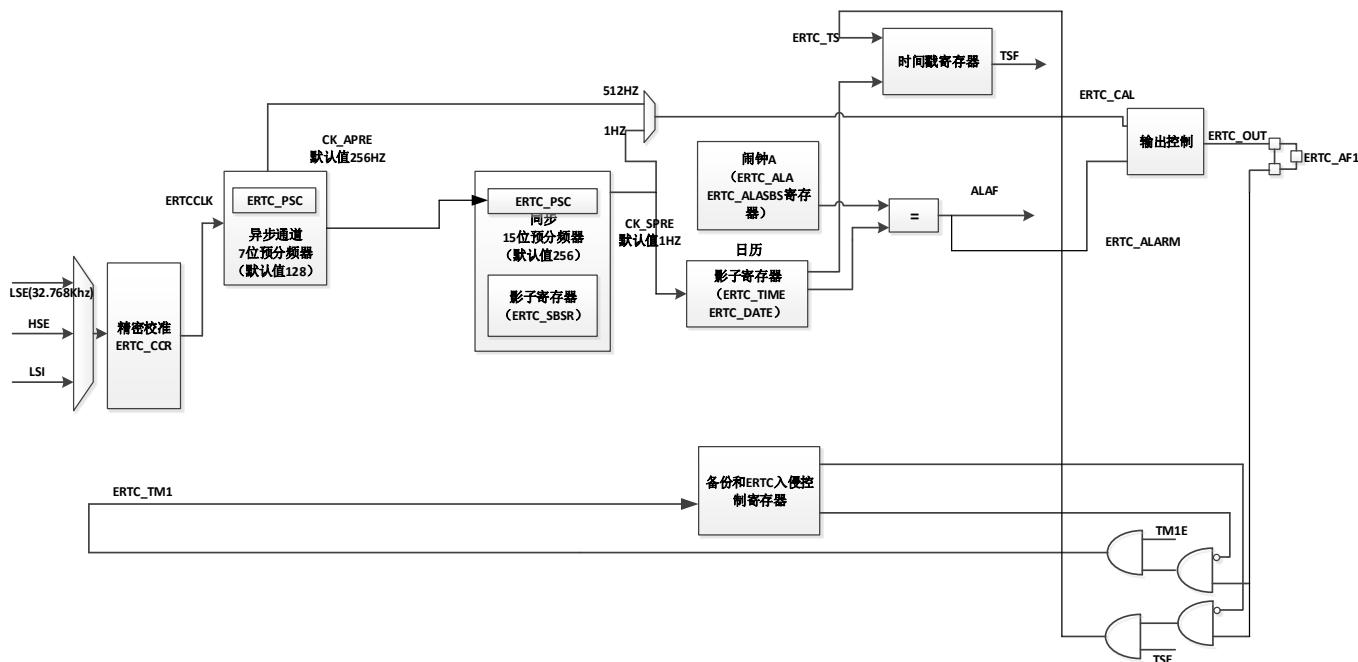
上电复位后，所有 ERTC 寄存器都会受到保护，以防止可能的非正常写访问。

无论器件状态如何 (运行模式、低功耗模式或处于复位状态)，只要电源电压保持在工作范围内，ERTC 便不会停止工作。

## 13.2 ERTC的主要特性

- 包含亚秒、秒、分钟、小时 (12/24 小时制)、星期几、日期、月份和年份的日历。
- 软件可编程的夏令时补偿。
- 一个具有中断功能的可编程闹钟。可通过任意日历字段的组合驱动闹钟。
- 参考时钟检测：可使用更加精确的第二时钟源 (50 Hz 或 60 Hz) 来提高日历的精确度。
- 利用亚秒级移位特性与外部时钟实现精确同步。
- 可屏蔽中断/事件：
  - 闹钟 A
  - 时间戳
  - 入侵检测
- 数字校准电路 (周期性计数器调整)
  - 精度为 0.95 ppm，在数秒钟的校准窗口中获得
- 用于事件保存的时间戳功能 (1 个事件)
- 入侵检测：
  - 1 个带可配置过滤器和内部上拉的入侵事件
- 5 个备份寄存器 (20 字节)。发生入侵检测事件时，将复位备份寄存器。
- 复用功能输出 (ERTC\_OUT)，可选择以下两个输出之一：
  - ERTC\_CAL：512 Hz 或 1 Hz 时钟输出 (LSE 频率为 32.768 kHz)。可通过将 ERTC\_CTRL 寄存器中的 CALOE[23] 位置 1 来使能此输出。该输出可连接到器件 ERTC\_AF1 功能。
  - ERTC\_ALARM (闹钟 A)。可通过配置 ERTC\_CTRL 寄存器的 OSEL[1:0] 位选择此输出。该输出可连接到器件 ERTC\_AF1 功能。
- ERTC 复用功能输入：
  - ERTC\_TS：时间戳事件检测。该输入可连接到器件 ERTC\_AF1。
  - ERTC\_TAMP：TAMPER 事件检测。该输入可连接到器件 ERTC\_AF1。
  - ERTC\_REFIN：参考时钟输入 (通常为市电，50 Hz 或 60 Hz)。

图表 13-1 ERTC框图



## 13.3 ERTC功能说明

### 13.3.1 时钟和预分频器

ERTC 时钟源 (ERTCCLK) 通过时钟控制器从 LSE 时钟、LSI 振荡器时钟以及 HSE 时钟三者中选择。有关 ERTC 时钟源配置的更多信息，请参见第 3 节：复位和时钟控制 (RCC)。

可编程的预分频器阶段可生成 1 Hz 的时钟，用于更新日历。为最大程度地降低功耗，预分频器分为 2 个可编程的预分频器（参见图 13-1：ERTC 框图）：

- 一个通过 ERTC\_PSC 寄存器的 PRDIV\_A 位配置的 7 位异步预分频器。
- 一个通过 ERTC\_PSC 寄存器的 PRDIV\_S 位配置的 15 位同步预分频器。

注意： 使用两个预分频器时，推荐将异步预分频器配置为较高的值，以最大程度降低功耗。

要使用频率为 32.768 kHz 的 LSE 获得频率为 1 Hz 的内部时钟 (ck\_spre)，需要将异步预分频系数设置为 128，并将同步预分频系数设置为 256。

分频系数的最小值为 1，最大值为  $2^{22}$ 。

这对应于约为 4 MHz 的最大输入频率。

$f_{ck\_apre}$  可根据以下公式得出：

$$f_{ck\_apre} = \frac{f_{ERTC\_CLK}}{PRDIV\_A + 1}$$

$ck\_apre$  时钟用于为二进制 ERTC\_SBSR 亚秒递减计数器提供时钟。当该计数器计数到 0 时，会使用 PRDIV\_S 的内容重载 ERTC\_SBSR。

$f_{ck\_spre}$  可根据以下公式得出：

$$f_{ck\_spre} = \frac{f_{ERTC\_CLK}}{(PRDIV\_S + 1) \times (PRDIV\_A + 1)}$$

### 13.3.2 实时时钟和日历

ERTC 日历时间和日期寄存器可通过与 PCLK1 (APB1 时钟) 同步的影子寄存器来访问。这些时间和日期寄存器也可以直接访问，这样可避免等待同步的持续时间。

- ERTC\_SBSR 对应于亚秒
- ERTC\_TIME 对应于时间
- ERTC\_DATE 对应于日期

每隔两个 ERTCCCLK 周期，便将当前日历值复制到影子寄存器，并将 ERTC\_STS 寄存器的 RSF 位置 1 (请参见第 13.6.4 节)。在停机和待机模式下不会执行复制操作。退出这两种模式时，影子寄存器会在最长 2 个 ERTCCCLK 周期后进行更新。

当应用读取日历寄存器时，它会访问影子寄存器的内容。也可以通过将 ERTC\_CTRL 寄存器的 BYPSHDW 控制位置 1 来直接访问日历寄存器。默认情况下，该位被清零，用户访问影子寄存器。

在 BYPSHDW=0 模式下读取 ERTC\_SBSR、ERTC\_TIME 或 ERTC\_DATE 寄存器时，APB 时钟频率 ( $f_{APB}$ ) 必须至少为 ERTC 时钟频率 ( $f_{ERTCCLK}$ ) 的 7 倍。

影子寄存器通过系统复位来复位。

### 13.3.3 可编程闹钟

ERTC 单元提供 1 个可编程闹钟，即闹钟 A。

可通过将 ERTC\_CTRL 寄存器中的 ALAE 位置 1 来使能可编程闹钟功能。如果日历、亚秒、秒、分钟、小时、日期或日分别与闹钟寄存器 ERTC\_ALASBS/ERTC\_ALA 中编程的值相匹配，则 ALAF 标志会被置为 1。可通过 ERTC\_ALA 寄存器的 MASKx 位以及 ERTC\_ALASBS 寄存器的 MASKSBS 位单独选择各日历字段。可通过 ERTC\_CTRL 寄存器中的 ALAIE 位使能闹钟中断。

闹钟 A (如果已通过 ERTC\_CTRL 寄存器中的位 OSEL[0:1]使能) 可连接到 ERTC\_ALARM 输出。可通过 ERTC\_CTRL 寄存器的 OPOL 位配置 ERTC\_ALARM 极性。

**注意：** 如果选择秒字段 (ERTC\_ALA 中的 MASK1 位复位)，则 ERTC\_PSC 寄存器中设置的同步预分频器分频系数必须至少为 3，才能确保闹钟正确地运行。

### 13.3.4 ERTC 初始化和配置

#### ERTC 寄存器访问

ERTC 寄存器为 32 位寄存器。除了当 BYPSHDW=0 时对日历影子寄存器执行的读访问之外，APB 接口会在访问 ERTC 寄存器时引入 2 个等待周期。

#### ERTC 寄存器写保护

系统复位后，可通过 PWR 电源控制寄存器 (PWR\_CTRL) 的 DBP 位保护 ERTC 寄存器以防止非正常的写访问。必须将 DBP 位置 1 才能使能 ERTC 寄存器的写访问。

上电复位后，所有 ERTC 寄存器均受到写保护。通过向写保护寄存器 (ERTC\_WPR) 写入一个密钥来使能对 ERTC 寄存器的写操作。

要解锁所有 ERTC 寄存器 (ERTC\_STS[13:8]、ERTC\_TPAF 和 ERTC\_BKPxDT 除外) 的写保护，需要执行以下步骤：

1. 将“0xCA”写入 ERTC\_WPR 寄存器。
2. 将“0x53”写入 ERTC\_WPR 寄存器。

写入一个错误的关键字会再次激活写保护。

保护机制不受系统复位影响。ERTC 寄存器写保护

#### 日历初始化和配置

要编程包括时间格式和预分频器配置在内的初始时间和日期日历值，需按照以下顺序操作：

1. 将 ERTC\_STS 寄存器中的 INITM 位置 1 以进入初始化模式。在此模式下，日历计数器将停止工作并且其值可更新。
2. 轮询 ERTC\_STS 寄存器中的 INITF 位。当 INITF 置 1 时进入初始化阶段模式。大约需要 2 个 ERTCCLK 时钟周期（由于时钟同步）。
3. 要为日历计数器生成 1 Hz 时钟，应首先编程 ERTC\_PSC 寄存器中的同步预分频系数，然后编程异步预分频系数。即使只需要更改这两个字段中之一，也必须对 ERTC\_PSC 寄存器执行两次单独的写访问。
4. 在影子寄存器（ERTC\_TIME 和 ERTC\_DATE）中加载初始时间和日期值，然后通过 ERTC\_CTRL 寄存器中的 HFM 位配置时间格式（12 或 24 小时制）。
5. 通过清零 INITM 位退出初始化模式。随后，自动加载实际日历计数器值，在 4 个 ERTCCLK 时钟周期后重新开始计数。

当初始化序列完成之后，日历开始计数。

**注意：** 系统复位后，应用可读取 ERTC\_STS 寄存器中的 INITS 标志，以检查日历是否已初始化。如果该标志为 0，表明日历尚未初始化，因为年份字段设置为其上电复位时的默认值 (0x00)。

要在初始化之后读取日历，必须首先用软件检查 ERTC\_STS 寄存器的 RSF 标志是否置 1。

### 夏令时

可通过 ERTC\_CTRL 寄存器的 SUB1H、ADD1H 和 BKP 位管理夏令时。

利用 SUB1H 或 ADD1H，软件只需单次操作便可在日历中减去或增加一个小时，无需执行整个初始化步骤。

此外，软件还可以使用 BKP 位来记录是否曾经执行过此操作。

### 编程闹钟

要对可编程的闹钟（闹钟 A）进行编程或更新，必须执行类似的步骤：

1. 将 ERTC\_CTRL 寄存器中的 ALAE 位清零以禁止闹钟 A。
2. 轮询 ERTC\_STS 寄存器中的 ALAWF 位，直到其中一个置 1，以确保闹钟寄存器可以访问。大约需要 2 个 RTCCLK 时钟周期（由于时钟同步）。
3. 编程闹钟 A 寄存器（ERTC\_ALASBS/ERTC\_ALA）。
4. 将 ERTC\_CTRL 寄存器中的 ALAE 位置 1 以再次使能闹钟 A。

**注意：** 约 2 个 ERTCCLK 时钟周期（由于时钟同步）后，将执行对 ERTC\_CTRL 寄存器的更改。

## 13.3.5 读取日历

### 当 ERTC\_CTRL 寄存器中的 BYPSHDW 控制位清零时

要正确读取 ERTC 日历寄存器（ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE），APB1 时钟频率 ( $f_{PCLK1}$ ) 必须等于或大于 ERTC 时钟频率的七倍。这可以确保同步机制行为的安全性。

如果 APB1 时钟频率低于 ERTC 时钟频率的七倍，则软件必须分两次读取日历时间寄存器和日期寄存器。这样，当两次读取的 ERTC\_TIME 结果相同时，才能确保数据正确。否则必须执行第三次读访问。任何情况下，APB1 的时钟频率都不能低于 ERTC 的时钟频率。

每次将日历寄存器中的值复制到 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 影子寄存器时，ERTC\_STS 寄存器中的 RSF 位都会置 1。每两个 ERTC CLK 周期执行一次复制。为确保这 3 个值来自同一时刻点，读取 ERTC\_SBSR 或 ERTC\_TIME 时会锁定高阶日历影子寄存器中的值，直到读取 ERTC\_DATE。为避免软件对日历执行读访问的时间间隔小于 2 个 ERTCCLK 周期：第一次读取日历之后必须通过软件将 RSF 清零，并且软件必须等待到 RSF 置 1 之后才可再次读取 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 寄存器。

从低功耗模式（停机模式或待机模式）唤醒之后，必须通过软件将 RSF 清零。之后，软件必须等待至 RSF 再次置 1 之后才可以读取 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 寄存器。

RSF 位必须在唤醒之后而不是进入低功耗模式之前进行清零。

注意： 系统复位之后，软件必须等待至 RSF 置 1 之后才可以读取 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 寄存器。实际上，系统复位会将影子寄存器复位为其默认值。

初始化之后（请参见第 13.3.4 节的日历初始化和配置）：软件必须等待至 RSF 置 1 之后才可以读取 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 寄存器。

同步之后（请参见第 13.3.7 节：ERTC 同步）：软件必须等待至 RSF 置 1 之后才可以读取 ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE 寄存器。

#### 当 ERTC\_CTRL 寄存器中的 BYPSHDW 控制位置 1 时（旁路影子寄存器）

读取日历寄存器时会直接从日历计数器获取值，这样便无需等待至 RSF 位置 1。这对于从低功耗模式（停机模式或待机模式）退出后的情况特别有用，因为影子寄存器在这些模式下不更新。

当 BYPSHDW 位置 1 时，如果在对寄存器的两次读访问之间出现 ERTCCLK 沿，则不同寄存器的结果彼此可能不一致。此外，如果在读操作期间出现 ERTCCLK 沿，则可能导致其中一个寄存器的值不正确。软件必须分两次读取所有寄存器，然后将两次结果加以比较来确认数据是否一致和正确。此外，软件也可以只比较两次读取日历寄存器得到的结果的最低位。

注意： 当 BYPSHDW=1 时，读取日历寄存器的指令需要一个额外的 APB 周期才能完成。

### 13.3.6 复位ERTC

日历影子寄存器（ERTC\_SBSR、ERTC\_TIME 和 ERTC\_DATE）以及 ERTC 状态寄存器(ERTC\_STS)的某些位通过所有可用的系统复位源复位为各自的默认值。

相反，以下寄存器则通过上电复位来复位为各自的默认值并且不受系统复位的影响：ERTC 当前日历寄存器、ERTC 控制寄存器(ERTC\_CTRL)、预分频器寄存器(ERTC\_PSC)、ERTC 校准寄存器(ERTC\_CCR)、ERTC 移位寄存器(ERTC\_SFCTR)、ERTC 时间戳寄存器（ERTC\_TSTM、ERTC\_TS DT 和 ERTC\_TSSBS）、ERTC 入侵和复用功能配置寄存器(ERTC\_TPAF)、ERTC 备份寄存器(ERTC\_BKPxDT)以及闹钟 A 寄存器（ERTC\_ALA/RTC\_ALASBS）。

此外，如果复位源不是上电复位源，则发生系统复位时，ERTC 会继续工作。发生上电复位时，ERTC 会停止工作，并且所有 ERTC 寄存器都会设置为各自的复位值。

### 13.3.7 ERTC同步

ERTC 可与高精度的远程时钟同步。在读取亚秒字段后（ERTC\_SBSR 或 ERTC\_TSSBS），即可计算远程时钟的时间与 ERTC 之间的精准偏差。之后，可使用 ERTC\_SFCTR 对 ERTC 的时钟进行零点几秒的“平移”，经过调整后可消除此偏差。

ERTC\_SBSR 包含同步预分频器计数器的值。这样，便可计算分辨率低至  $1/(PRDIV_S + 1)$  秒的 ERTC 的准确时间。因此，可通过增大同步预分频器的值 (PRDIV\_S[14:0]) 来提高 分辨率。将 PRDIV\_S 设置为 0x7FFF 时，可得到允许的最大分辨率 ( $30.52 \mu s$ ，时钟频率为  $32768 Hz$ )。

但是，提高 PRDIV\_S 意味着必须降低 PRDIV\_A 才能将同步预分频器的输出维持在  $1 Hz$ 。这样，异步预分频器的输出频率会增大，ERTC 的动态功耗也会相应增加。

可以使用 ERTC 平移控制寄存器 (ERTC\_SFCTR) 对 ERTC 进行微调。可以用大小为  $1/(PRDIV_S + 1)$  秒的分辨率对 ERTC\_SFCTR 进行写操作，将时钟平移（延迟或提前）最长 1 秒。在这种平移操作中，会将 SUBSBS[14:0] 值加到同步预分频器计数器 SBS[15:0] 中：这将使时钟产生延迟。如果同时将 ADD1S 位置 1，则会增加一秒，与此同时减去的时间为零点几秒，因此将使时钟提前。

对 ERTC\_SFCTR 寄存器执行写操作以启动平移操作时，硬件会将 SFP 标志置 1 以指示平移操作挂起。完成平移操作时，硬件会将该位清零。

注意： 初始化平移操作前，用户必须检查确认 SBS[15] = 0，以确保不会发生上溢。

注意： 该同步功能与参考时钟检测功能不兼容：当 RFCKON=1 时，固件不能对 ERTC\_SFCTR 执行写操作。

### 13.3.8 ERTC参考时钟检测

ERTC 日历更新可与参考时钟 ERTC\_REFIN（通常为市电， $50 Hz$  或  $60 Hz$ ）同步。ERTC\_REFIN 参

考时钟的精度应高于 32.768 kHz LSE 时钟。使能 ERTC\_REFIN 检测时（将 ERTC\_CTRL 的 RFCKON 位置 1），日历仍由 LSE 提供时钟，而 ERTC\_REFIN 用于补偿不准确的日历更新频率 (1 Hz)。

每个 1 Hz 时钟边沿都与最近的参考时钟边沿进行比较（如果在给定的时间窗口内发现一个边沿）。在大多数情况下，两个时钟边沿恰好对齐。当 1 Hz 时钟由于 LSE 时钟不精确而发生偏离时，ERTC 会稍微偏移 1 Hz 时钟，以便后续的 1 Hz 时钟边沿能够对齐。利用这种机制，可使日历像参考时钟一样精确。

ERTC 使用 32.768 kHz 石英产生的 256 Hz 时钟 (ck\_apre) 检测是否存在参考时钟源。大约在日历每次更新时（每 1 秒钟），便会在时间窗口期间执行一次检测。检测到第一个参考时钟边沿时，该窗口等于 7 个 ck\_apre 周期。随后的日历更新使用长度为 3 个 ck\_apre 周期的较小窗口。

每次在窗口中检测到参考时钟时，都会行强制输出 ck\_apre 时钟的异步预分频器进行重载。当参考时钟与 1 Hz 时钟对齐时，此操作不起作用，因为预分频器会在同一时刻重载。当时钟不对齐时，重载操作会微调后续的 1 Hz 时钟边沿，使其与参考时钟对齐。

如果参考时钟停止（在 3 个 ck\_apre 窗口内未出现参考时钟边沿），日历将仅根据 LSE 时钟进行连续更新。ERTC 随后使用 ck\_spre 边沿上居中的大检测窗口（7 个 ck\_apre 周期）等待参考时钟。

使能参考时钟检测后，必须将 PRDIV\_A 和 PRDIV\_S 设置为各自的默认值：

- PRDIV\_A = 0x007F
- PREVID\_S = 0x00FF

注意：参考时钟检测在待机模式下不可用

### 13.3.9 ERTC精密数字校准

ERTC 频率可采用约 0.954 ppm 的分辨率进行数字校准，校准范围为 -487.1 ppm 到 +488.5 ppm。使用一系列微调（增加和/或减少单独的 ERTCCCLK 脉冲）进行频率校正。这些微调的分布非常均匀，因此 ERTC 的校准效果相当好，即使在短时间内持续观察也是如此。

当输入频率为 32768 Hz 时，精密数字校准的周期约为  $2^{20}$  个 ERTCCCLK 脉冲或 32 秒。

精密数字校准寄存器 (ERTC\_CCR) 可指定 32 秒周期内要减少的 ERTCCCLK 时钟周期数：

- 将 CALM[0] 置 1 时，32 秒周期内将只减少 1 个周期。
- 将 CALM[1] 置 1 时，将减少 2 个周期。
- 将 CALM[2] 置 1 时，将减少 4 个周期。
- 依此类推，将 CALM[8] 置 1 时，将减少 256 个周期。

使用适当分辨率时，CALM 可使 ERTC 频率减少最多 487.1 ppm，而 CALAD 可用于使频率增加 488.5 ppm。将 CALAD 置“1”，可每隔  $2^{11}$  个 ERTCCCLK 周期有效插入一个额外的 ERTCCCLK 脉冲，这意味着每 32 秒周期可增加 512 个时钟。

与 CALM 和 CALAD 配合使用时，可在 32 秒周期内增加一个范围为 -511 到 +512 ERTCCCLK 周期的偏差，对应的校准范围为 -487.1 ppm 到 +488.5 ppm，分辨率约为 0.954 ppm。

若输入频率 ( $F_{ERTCCCLK}$ ) 已知，可通过以下公式计算有效校准频率 ( $F_{CAL}$ )：

$$F_{CAL} = F_{ERTCCCLK} \times \left[ 1 + \frac{CALAD \times 512 - CALM}{2^{20} + CALM - CALAD \times 512} \right]$$

#### PRDIV\_A<3 条件下的校准

当异步预分频器值 (ERTC\_PSC 寄存器中的 PRDIV\_A 位) 小于 3 时，不能将 CALAD 位置 1。如果 CALAD 已置 1 并且 PRDIV\_A 位的值小于 3，则会忽略 CALAD，即假定 CALAD 等于 0 而执行校准。

要在 PRDIV\_A 小于 3 的条件下执行校准，应降低同步预分频器值 (PRDIV\_S) 以便每秒内可加速 8 个 ERTCCCLK 时钟周期，这意味着每 32 秒可增加 256 个时钟周期。因此，仅使用 CALM 位，可在每

32 秒内有效增加 255 到 256 个时钟脉冲（对应的校准范围为 243.3 ppm 到 244.1 ppm）。

在标称 ERTCCLK 频率 32768 Hz 下，当 PRDIV\_A 等于 1 时（分频系数为 2），应将 PRDIV\_S 设置为 16379 而不是 16383（少 4）。唯一相关的其它情况是，当 PRDIV\_A 等于 0 时，应将 PRDIV\_S 设置为 32759 而不是 32767（少 8）。

如果以这种方式减少 PRDIV\_S，则采用以下公式计算校准输入时钟的有效频率：

$$F_{CAL} = F_{ERTCCLK} \times \left[ 1 + \frac{256 - CALM}{2^{20} + CALM - 256} \right]$$

在这种情况下，如果 ERTCCLK 恰好为 32768.00 Hz，则当 CALM[7:0] 等于 0x100 时（CALM 范围的中值），说明设置正确。

### 验证 ERTC 校准

通过测量 ERTC CLK 的精确频率，计算正确的 CALM 和 CALAD 值以实现 ERTC 精度。此外，还为应用提供了一个可选的 1 Hz 输出，用来测量和验证 ERTC 精度。

如果在有限的间隔内测量 ERTC 的精确频率，则会导致测量期间产生最多 2 个 ERTCCLK 时钟周期的测量误差，具体取决于数字校准周期与测量周期的对齐方式。

但是，如果测量周期与校准周期的长度相同，则可以消除此测量误差。在这种情况下，观测到的唯一误差是由数字校准的分辨率导致的误差。

- 默认情况下，校准周期为 32 秒。在此模式下，测量整个 32 秒内 1 Hz 输出的精度，可确保测量误差在 0.477 ppm 内（32 秒内为 0.5 个 ERTCCLK 周期，受校准分辨率限制）。
- 可将 ERTC\_CCR 寄存器的 CAL16 位置 1，以强制 16 秒的校准周期。此时，可在 16 秒内测量 ERTC 精度，产生的最大误差为 0.954 ppm（16 秒内为 0.5 个 ERTCCLK 周期）。但是，由于校准分辨率降低，长期的 ERTC 精度也会降到 0.954 ppm：将 CAL16 置 1 时，CALM[0] 位将始终保持为 0。
- 可将 ERTC\_CCR 寄存器的 CAL8 位置 1，以强制 8 秒的校准周期。此时，可在 8 秒内测量 ERTC 精度，产生的最大误差为 1.907 ppm（8 秒内为 0.5 个 ERTCCLK 周期）。长期的 ERTC 精度也会降到 1.907 ppm：将 CAL8 置 1 时，CALM[1:0] 位将始终保持为 00。

### 动态重校准

当 ERTC\_STS/INITF=0 时，可动态更新校准寄存器 (ERTC\_CCR)，具体步骤如下：

1. 轮询 ERTC\_STS/RECALPDF（重新校准挂起标志）。
2. 如果该标志为 0，则可以根据需要向 ERTC\_CCR 写入新值。随后 RECALPDF 位会被自动置为 1。
3. 新校准设置将在对 ERTC\_CCR 执行写操作之后的三个 ck\_apre 周期内生效。

### 13.3.10 时间戳功能

将 ERTC\_CTRL 寄存器的 TSE 位置 1 可使能时间戳。

当在 TIMESTAMP 备用功能映射到的引脚上检测到时间戳事件时，日历会保存到时间戳寄存器 (ERTC\_TSSBS、ERTC\_TSTM 和 ERTC\_TS DT) 中。发生时间戳事件时，ERTC\_STS 寄存器中的时间戳标志位 (TSF) 将置 1。

通过将 ERTC\_CTRL 寄存器中的 TSIE 位置 1，可在发生时间戳事件时生成中断。

如果在时间戳标志 (TSF) 已置 1 的条件下检测到新的时间戳事件，则时间戳上溢标志 (TSOF) 将置 1，而时间戳寄存器 (ERTC\_TSTM 和 ERTC\_TS DT) 将保持上一事件的结果。

注意：由于同步过程，TSF 将在时间戳事件后 2 个 ck\_apre 周期置 1。

将 TSOF 置 1 时不存在延迟。这意味着，如果两个时间戳事件接连发生，则 TSOF 可能为“1”而 TSF 仍为“0”。因此，建议只在检测到 TSF 为“1”后再轮询 TSOF。

注意：如果在 TSF 位清零后紧接着发生时间戳事件，则 TSF 和 TSOF 位都将置 1。为防止在时间戳事件发生的同时屏蔽该事件，除非已将 TSF 位读取为“1”，否则应用程序不得将“0”写入 TSF 位。

此外，入侵事件可能导致时间戳被记录。有关 TMTS 控制位的说明，请参见第 13.6.14 节：ERTC 入侵和复用功能配置寄存器 (ERTC\_TPAF)。如果时间戳事件与配置为过滤模式的入侵事件 (TMFLT 设置为非零值) 使用同一引脚，则必须通过将 ERTC\_TPAF 寄存器中的 TMTS 置“1”来选择入侵检测事件的时间戳模式。

### 13.3.11 入侵检测

ERTC 有一个入侵检测输入可用。该输入既可配置为边沿检测，也可配置为带过滤的电平检测。

#### ERTC 备份寄存器

备份寄存器 (ERTC\_BKPxDT) 包括 5 个 32 位寄存器，用于存储 20 字节的用户应用数据。这些寄存器在备份域中实现，备份寄存器不会在系统复位时复位，也不会在器件从待机模式唤醒时复位。

发生入侵检测事件时，将复位备份寄存器。(请参见第 13.6.16 节：ERTC 备份寄存器 (ERTC\_BKPxDT))  
**入侵检测初始化**

入侵检测输入与 ERTC\_STS 寄存器中的标志 TPF 相关联。可通过将 ERTC\_TPAF 寄存器中相应的 TM1E 位置 1 来使能各输入。

入侵检测事件会复位所有备份寄存器 (ERTC\_BKPxDT)。

通过将 ERTC\_TPAF 寄存器中的 TMIE 位置 1，可在发生入侵检测事件时生成中断。

#### 入侵事件的时间戳

当 TMTS 置“1”时，任何入侵事件都会导致时间戳事件的发生。在这种情况下，如同发生正常时间戳事件一样，ERTC\_STS 中的 TSF 位或 TSOF 位会置 1。在 TSF 或 TSOF 置 1 的同时，受影响的入侵标志寄存器 (TPF) 也会随之置 1。

#### 对入侵输入的边沿检测

如果 TMFLT 位为“00”，则在观测到上升沿或下降沿时（根据相应的 TM1TRG 位），TAMPER 引脚会生成入侵检测事件 (ERTC\_TAMP)。选择边沿检测时，会禁止入侵输入上的内部上拉电阻。

注意：为避免丢失入侵检测事件，用于边沿检测的信号与 TM1E 使用逻辑与操作来检测入侵事件，以避免丢失在使能 TAMPER 引脚前发生的入侵事件。

- 当 TM1TRG = 0 时：如果 TAMPER 复用功能在使能入侵检测之前已变为高电平 (TM1E 位置 1)，则使能 TAMPER 后便会立即检测到入侵事件，即使在 TM1E 置 1 后 TAMPER 引脚上未出现上升沿。
- 当 TM1TRG = 1 时：如果 TAMPER 复用功能在使能入侵检测之前已变为低电平，则使能 TAMPER 后便可立即检测到入侵事件（即使在 TM1E 置 1 后 TAMPER 引脚上未出现下降沿）。

检测到入侵事件并清零后，应当在重新编程备份寄存器 (ERTC\_BKPxDT) 之前禁止 TAMPER 复用功能，然后再重新使能 (TM1E 置 1)。这可防止应用在 TAMPER 值仍指示入侵检测时，对备份寄存器执行写操作。这相当于对 TAMPER 复用功能的电平检测。

注意：当 VDD 电源关闭时，入侵检测仍有效。要避免意外复位备份寄存器，应将 TAMPER 复用功能映射到的引脚从外部连接到正确的电平。

#### 对入侵输入的带过滤电平检测

通过将 TMFLT 设置为非零值可执行带过滤的电平检测。在 TM1TRG 位指定的电平连续出现 2、4 或 8 个（取决于 TMFLT 值）采样时生成入侵检测事件。

除非通过将 TMPUDIS 置 1 禁止入侵输入，否则在入侵输入的状态被采样前，将通过 I/O 内部上拉电阻对这些输入预充电。预充电的持续时间由 TMPRCH 位确定，允许增大入侵输入上的电容。

可使用 TMFREQ 确定用于电平检测的采样频率，以便使入侵检测延迟与上拉电阻功耗之间达到最佳平衡。

### 13.3.12 校准时钟输出

将 ERTC\_CTRL 寄存器中的 CALOE 位置 1 时，会在 ERTC\_CAL 器件输出上提供一个参考时钟。如果 ERTC\_CTRL 寄存器中的 CALSEL 位复位且 PRDIV\_A = 0x7F，则 ERTC\_CAL 频率为  $f_{ERTCCLK}/64$ 。这相当于 ERTCCLK 频率为 32.768 kHz 时，512 Hz 的校准输出。

ERTC\_CAL 占空比是不规则的：下降沿上存在轻微抖动。因此推荐使用上升沿。

如果 CALSEL 置 1 且“PRDIV\_S+1”为 256 的非零整数倍（即：PRDIV\_S[7:0] = 0xFF），则 ERTC\_CAL 频率为  $f_{ERTCCLK}/(256 * (PRDIV_A+1))$ 。这相当于 ERTCCLK 频率为 32.768 kHz 时，1 Hz 的校准输出，其中预分频器为默认值（PRDIV\_A = 0x7F、PRDIV\_S = 0xFF）。

#### 校准复用功能输出

当将 ERTC\_CTRL 寄存器中的 CALOE 位置 1 时，ERTC\_AF1 上会使能校准复用功能 (ERTC\_CAL)。

### 13.3.13 闹钟输出

闹钟输出有一种功能可供选择：ALAF。这些功能可反映 ERTC\_STS 寄存器中相应标志的内容。

ERTC\_CTRL 寄存器中的 OSEL[1:0] 控制位用于激活 ERTC\_AF1 中的闹钟复用功能输出 (ERTC\_ALARM)，以及选择 ERTC\_ALARM 输出上的功能。

输出的极性由 ERTC\_CTRL 中的 OPOL 控制位确定，这样当 OPOL 置 1 时会输出选定标志位的相反值。

#### 闹钟复用功能输出

使用 ERTC\_TPAF 寄存器中的控制位 ALAOUTTYPE 可将 ERTC\_ALARM 配置为开漏输出或推挽输出。

注意：使能 ERTC\_ALARM 之后，其优先级高于 ERTC\_CAL (CALOE 位的设置与 ERTC\_AF1 无关)。

选择 ERTC\_CAL 或 ERTC\_ALARM 时，ERTC\_AF1 会自动配置为输出复用功能。

## 13.4 ERTC 和低功耗模式

表格 13-1 低功耗模式对 ERTC 的作用

模式	说明
睡眠	无影响 ERTC 中断可使器件退出睡眠模式。
停机	当 ERTC 时钟源为 LSE 或 LSI 时，ERTC 保持工作状态。ERTC 闹钟、ERTC 入侵事件、ERTC 时 间戳事件会使器件退出停机模式。
待机	当 ERTC 时钟源为 LSE 或 LSI 时，ERTC 保持工作状态。ERTC 闹钟、ERTC 入侵事件、ERTC 时 间戳事件会使器件退出待机模式。

## 13.5 ERTC 中断

所有 ERTC 中断均与 EXTI 控制器相连。

要使能 ERTC 闹钟中断，需按照以下顺序操作：

1. 将 EXTI 线 17 配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 ERTC IRQ 通道并将其使能。
3. 配置 ERTC 以生成 ERTC 闹钟（闹钟 A）。

要使能 ERTC 入侵中断，需按照以下顺序操作：

1. 将 EXTI 线 19 配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 ERTC IRQ 通道并将其使能。
3. 配置 ERTC 以检测 ERTC 入侵事件。

要使能 ERTC 时间戳中断，需按照以下顺序操作：

1. 将 EXTI 线 19 配置为中断模式并将其使能，然后选择上升沿有效。
2. 配置 NVIC 中的 ERTC IRQ 通道并将其使能。

3. 配置 ERTC 以检测 ERTC 时间戳事件。

表格 13-2 中断控制位

中断事件	时间标志	使能控制位	退出睡眠模式	退出停机模式	退出待机模式
闹钟 A	ALAF	ALAIE	是	是(1)	是(1)
TimeStamp	TSF	TSIE	是	是(1)	是(1)
Tamper 检测 (2)	TPF	TMIE	是	是(1)	是(1)

1. 仅当 ERTC 时钟源为 LSE 或 LSI 时，才能从停机和待机模式唤醒。

2. 如果存在 ERTC\_TAMPER2 引脚。请参见器件数据手册的引脚排列。

## 13.6 ERTC寄存器

表格 13-3 ERTC寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
000h	ERTC_TIME	保留					AMPM		HT[1:0]		HU[3:0]		保留		MT[2:0]		MU[3:0]		保留		ST[2:0]		DU[3:0]		SU[3:0]		保留														
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
004h	ERTC_DATE	保留					YT[3:0]		YU[3:0]		WK[2:0]		MT		MU[3:0]		保留		DT[1:0]		DU[3:0]		保留		DT[1:0]		DU[3:0]														
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1											
008h	ERTC_CTR_L	保留					CALOE		OSEL[1:0]		OPOL		CALSEL		BKP		SUB1H		ADD1H		TSE		保留		保留		ALAE		保留												
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1											
00Ch	ERTC_STS	保留										RECALPDF		保留		TPF		TSOF		TSF		保留		保留		ALAF		INITM		INITF		RSF		BYPSHDW		REFECKON		TSEDGE		保留	
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1										
010h	ERTC_PSC	保留					PRDIV_A[6:0]					保留		PRDIV_S[6:0]					保留					保留					ALAWF												
		1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1											
01Ch	ERTC_ALA	MASK4		WKSEL		DT[1:0]		DU[3:0]		MASK3		AMPM		HT[1:0]		HU[3:0]		MSKA2		MT[2:0]		MU[3:0]		MASK1		ST[2:0]		SU[3:0]		保留											
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											
024h	ERTC_WPR	保留															KEY[7:0]										保留														
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0											

028h	ERTC_SBS R	保留								SBS[15:0]															
	复位值									0	0	0	0	0	0	0	0	0	0	0	0	0			
02Ch	ERTC_SFC TR	ADD1S	保留								SUBSBS[14:0]														
	复位值		0	0	0	0	0	0	0	0	0	0	0	0	0										
030h	ERTC_TST M	保留	AMPMM HT[1:0]	HU[3:0]	保留	MT[2:0]	MU[3:0]	保留	ST[2:0]	SU[3:0]	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留				
	复位值																								
034h	ERTC_TSD T	保留	WK[2:0]	MT	MU[3:0]	保留	DT[1:0]	DU[3:0]	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留				
	复位值																								
038h	ERTC_TSS BS	保留	SBS[15:0]												保留										
	复位值		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
03Ch	ERTC_CCR	保留	CALAD	CAL18	CAL16	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留				
	复位值														0	0	0	0	0	0	0	0			
040h	ERTC_TPA F	保留	ALAYOUTPE	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留				
	复位值														0	0	0	0	0	0	0	0			
044h	ERTC_ALA SBS	保留	MASKSBS [3:0]	保留	保留	保留	SBS[14:0]												保留						
	复位值						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
050h- 060h	ERTC_BKPx DT	保留	保留								D[15:0]														
	复位值										0	0	0	0	0	0	0	0							

### 13.6.1 ERTC时间寄存器(ERTC\_TIME)

ERTC\_TIME 是日历时间影子寄存器。只能在初始化模式下对该寄存器执行写操作。请参见第 [13.3.4 节 ERTC 初始化和配置](#), [13.3.5 节 读取日历](#)。

偏移地址: 0x00

上电复位值: 0x0000 0000

系统复位: 当 BYPSHDW = 0 时为 0x0000 0000; 当 BYPSHDW = 1 时不受影响。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16								
RES									AMPM	HT[1:0]	HU[3:0]												

15	14	13	12	11	10	9	8	7	RW	RW	RW	RW	RW	RW	RW	RW
RES	MT[2:0]			MU[3:0]				RES	ST[2:0]				SU[3:0]			
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW

位 31:24	保留
位 23	保留, 必须保持复位值
位 22	AMPM:AM/PM 符号 0: AM 或 24 小时制 1: PM
位 21:20	HT[1:0]:小时的十位 (BCD 格式)
位 16:16	HU[3:0]:小时的个位 (BCD 格式)
位 15	保留, 必须保持复位值
位 14:12	MT[2:0]:分钟的十位 (BCD 格式)
位 11:8	MU[3:0]:分钟的个位 (BCD 格式)
位 7	保留, 必须保持复位值
位 6:4	ST[2:0]:秒的十位 (BCD 格式)
位 3:0	SU[3:0]:秒的个位 (BCD 格式)

注意：此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

注意：为保证读取到的时间值为最新值，建议读取 ERTC\_TIME 前读取一次 ERTC\_STS 中 RSF 位（寄存器同步标志）。

### 13.6.2 ERTC日期寄存器(ERTC\_DATE)

ERTC\_DATE 是日历日期影子寄存器。只能在初始化模式下对该寄存器执行写操作。请参见第 [13.3.4 节](#) ERTC 初始化和配置，[13.3.5 节](#) 读取日历。

偏移地址: 0x04

上电复位值: 0x0000 2101

系统复位：当 BYPSHDW = 0 时为 0x0000 2101；当 BYPSHDW = 1 时不受影响。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES								YT[3:0]	YU[3:0]						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WK[2:0]		MT		MU[3:0]				RES		DT[1:0]	DU[3:0]				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31:24	保留
位 23:20	YT[3:0]:年份的十位
位 19:16	YU[3:0]:年份的个位
位 15:13	WK[2:0]:星期几 000: 禁止 001: 星期一 ... 111: 星期日
位 12	MT:月份的十位 (BCD 格式)
位 11:8	MU:月份的个位 (BCD)
位 7:6	保留, 必须保持复位值
位 5:4	DT[1:0]:日期的十位
位 3:0	DU[3:0]:日期的个位

注意：此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

注意：为保证读取到的时间值为最新值，建议读取 ERTC\_DATE 前读取一次 ERTC\_STS 中 RSF 位（寄存器同步标志）。

### 13.6.3 ERTC控制寄存器(ERTC\_CTRL)

偏移地址: 0x08

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES									C AL O E	OSEL[1:0]	OPOL	CAL SEL	B K P	SUB 1H	ADD 1H
R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS IE	RES	ALAIE	T S E	RES	ALA E	R E S	HF M	BYP SHD W	RFCK ON	TSE DGE	RES				
R W		R W	R W		R W	R W	R W	R W	R W	R W					

位 31:24	保留,必须保持复位值
位 23	CALOE:校准输出使能 该位使能 ERTC_CAL 输出 0: 禁止校准输出 1: 使能校准输出
位 22:21	OSEL[1:0]:输出选择 这些位用于选择要连接到 ERTC_ALARM 输出的标志 00: 禁止输出 01: 使能闹钟 A 输出 10: 保留 11: 保留
位 20	OPOL: 输出极性 该位用于配置 ERTC_ALARM 输出的极性 0: 当 ALAF 置 1 时 (取决于 OSEL[1:0]), 该引脚为高电平 1: 当 ALAF 置 1 时 (取决于 OSEL[1:0]), 该引脚为低电平
位 19	CALSEL: 校准输出选择 当 CALOE=1 时, 该位可选择 ERTC_CAL 上输出的信号。 0: 校准输出为 512 Hz 1: 校准输出为 1 Hz 在 ERTCCLK 为 32.768 kHz 且预分频器为其默认值 (PRDIV_A=127 且 PRDIV_S=255) 的条件下, 这些频率有效。请参见第 <a href="#">13.3.12 节: 校准时钟输出</a> 。
位 18	BKP: 备份 (Backup) 用户可对此位执行写操作以记录是否已对夏令时进行更改。
位 17	SUB1H: 减少 1 小时 (冬季时间更改) 当该位在初始化模式以外的模式下置 1 时, 如果当前小时不是 0, 则日历时间将减少 1 小时。此位始终读为 0。 当前小时为 0 时, 将此位置 1 没有任何作用。 0: 无作用。 1: 将当前时间减少 1 小时。这可用于冬季时间更改。
位 16	ADD1H: 增加 1 小时 (夏季时间更改) 当该位在初始化模式以外的模式下置 1 时, 日历时间将增加 1 小时。此位始终读为 0。 0: 无作用。 1: 将当前时间增加 1 小时。这可用于夏季时间更改
位 15	TSIE: 时间戳中断使能 0: 禁止时间戳中断 1: 使能时间戳中断
位 14	保留
位 13	保留
位 12	ALAIE: 闹钟 A 中断使能 0: 禁止闹钟 A 中断 1: 使能闹钟 A 中断
位 11	TSE: 时间戳使能 0: 禁止时间戳

	1: 使能时间戳
位 10	保留
位 9	保留
位 8	ALAE: 闹钟 A 使能 0: 禁止闹钟 A 1: 使能闹钟 A
位 7	保留
位 6	HFM: 小时格式 0: 24 小时/天格式 1: AM/PM 小时格式
位 5	BYPSHDW: 旁路影子寄存器 0: 日历值 (从 EERTC_SBSR、EERTC_TIME 和 EERTC_DATE 读取时) 取自影子寄存器, 该影子寄存器每两个 ERTCCCLK 周期更新一次。 1: 日历值 (从 EERTC_SBSR、EERTC_TIME 和 EERTC_DATE 读取时) 直接取自日历计数器。 注意: 如果 APB1 时钟的频率低于 7 倍的 ERTCCCLK 频率, 则必须将 BYPSHDW 置“1”。
位 4	RFCKON: 参考时钟检测使能 (50 Hz 或 60 Hz) 0: 禁止参考时钟检测 1: 使能参考时钟检测 注意: PRDIV_S 必须为 0x00FF。
位 3	TSEDGE: 时间戳事件有效边沿 0: TIMESTAMP 上升沿生成时间戳事件 1: TIMESTAMP 下降沿生成时间戳事件 TSEDGE 发生更改时, 必须复位 TSE 以避免将 TSF 意外置 1
位 2:0	保留

注意: 只能在初始化模式下 (ERTC\_STS 中 INITF = 1) 对该寄存器的位 6 和 4 执行写操作。

建议不要在日历小时递增时更改小时, 因为这样做会屏蔽日历小时的增量。

ADD1H 和 SUB1H 的更改在下一秒生效。

此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.4 ERTC 初始化和状态寄存器(ERTC\_STS)

偏移地址: 0x0C

上电复位值: 0x0000 0007

系统复位值: 不受影响 (INITM、INITF 和 RSF 除外, 它们在复位时被清零)。

31	30	29	28	27	2 6	2 5	24	23	22	21	20	19	1 8	1 7	16
RES														RECALPD F	
RES														R	
15	14	13	12	11	1 0	9	8	7	6	5	4	3	2	1	0
RE S	RE S	TPF	TSOF	TSF	RES	ALA F	INIT M	INIT F	RSF	INIT S	SF P	RES	RES	ALAWF	
		RC_W 0	RC_W 0	RC_W 0		RC- W0	RW	R	RC_W 0	R	RC- W0			R	

位 31:17	保留
位 16	RECALPDF: 重新校准挂起标志 (Recalibration pending Flag) 当软件对 ERTC_CCR 寄存器执行写操作时, RECALPDF 状态标志将自动置“1”, 指示 ERTC_CCR 寄存器已屏蔽。当采用新的校准设置时, 该位恢复为“0”。请参见动态重校准一节。
位 15	保留, 必须保持复位值
位 14	保留, 必须保持复位值
位 13	TPF: 入侵检测标志 当检测到入侵检测事件时, 由硬件将此标志置 1。该标志由软件写零清除。

位 12	<b>TSOF:</b> 时间戳溢出标志 当在 TSF 已置 1 的情况下发生时间戳事件时，由硬件将此标志置 1。 该标志由软件写零清除。 建议仅在 TSF 位清零之后再检查并清零 TSOF 位。否则，如果时间戳事件恰好在清零 TSF 位之前刚刚发生，则溢出事件可能会被漏掉。
位 11	<b>TSF:</b> 时间戳标志 发生时间戳事件时，由硬件将此标志置 1。 该标志由软件写零清除。
位 10	保留
位 9	保留
位 8	<b>ALAF:</b> 阵钟 A 标志 当时间/日期寄存器 (ERTC_TIME 和 ERTC_DATE) 与闹钟 A 寄存器 (ERTC_ALA) 匹配时，由硬件将该标志置 1。 该标志由软件写零清除。
位 7	<b>INITM:</b> 初始化模式 0: 自由运行模式。 1: 初始化模式，用于编程时间和日期寄存器 (ERTC_TIME 和 ERTC_DATE) 以及预分频器寄存器 (ERTC_PSC)。计数器停止计数，当 INITM 被复位后，计数器从新值开始计数。
位 6	<b>INITF:</b> 初始化标志 当此位置 1 时，ERTC 处于初始化状态，此时可更新事件、日期和预分频器寄存器。 0: 不允许更新日历寄存器。 1: 允许更新日历寄存器。
位 5	<b>RSF:</b> 寄存器同步标志 每次将日历寄存器的值复制到影子寄存器 (ERTC_SBSRx、ERTC_TIMEx 和 ERTC_DATEx) 时，都会由硬件将此位置 1。在初始化模式下、平移操作挂起时 (SFP=1) 或在旁路影子寄存器模式 (BYPSHDW=1) 下，该位由硬件清零。该位还可由软件清零。 0: 日历影子寄存器尚未同步 1: 日历影子寄存器已同步
位 4	<b>INITS:</b> 初始化状态标志 当日历年份字段不为 0 时（上电复位状态），由硬件将该位置 1。 0: 日历尚未初始化 1: 日历已经初始化
位 3	<b>SFP:</b> 平移操作挂起 0: 没有平移操作挂起 1: 某个平移操作挂起 只要通过对 ERTC_SFCTR 寄存器执行写操作来启动平移操作，此标志便由硬件置 1。执行完相应的平移操作后，此标志由硬件清零。对 SFP 执行写入操作不起作用。
位 2	保留
位 1	保留
位 0	<b>ALAWF:</b> 阵钟 A 写标志 在 ERTC_CTRL 寄存器中的 ALAE 位置 0 后，当闹钟 A 的值可更改时，由硬件将该位置 1。该位在初始化模式下由硬件清零。 0: 不允许更新闹钟 A 1: 允许更新闹钟 A

注意： 将 ALAF 和 TSF 位编程为 0 之后 2 个 APB 时钟周期，清零生效。

此寄存器受写保护 (ERTC\_STS[13:8] 位除外)。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.5 ERTC预分频器寄存器(ERTC\_PSC)

偏移地址: 0x10

上电复位值: 0x007F 00FF

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES														PRDIV_A[6:0]	
									RW	RW	RW	RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	PRDIV_S[14:0]														
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31:24	保留
位 23	保留, 必须保持复位值
位 22:16	PRDIV_A[6:0]: 异步预分频系数 下面是异步分频系数的公式: $ck_{apre}$ 频率 = ERTCCLK 频率/(PRDIV_A+1) 注意: PRDIV_A [6:0]= 000000 为禁用值。
位 15	保留, 必须保持复位值
位 14:0	PRDIV_S[14:0]: 同步预分频系数 下面是同步分频系数的公式: $ck_{spre}$ 频率 = $ck_{apre}$ 频率/(PRDIV_S+1)

注意: 只能在初始化模式下对该寄存器执行写操作。必须通过两次独立的写访问执行初始化。请参见第 [13.3.4 节 ERTC 初始化和配置](#)。

此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.6 ERTC闹钟A寄存器(ERTC\_ALA)

偏移地址: 0x1C

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASK 4	WKSEL L	DT[1:0]		DU[3:0]				MASK 3	AMPM	HT[1:0]		HU[3:0]			
RW	RW	R W	R W	R W	R W	R W	R W	RW	RW	R W	R W	R W	R W	R W	R W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK 2	MT[2:0]			MU[3:0]				MSK1	ST[2:0]			SU[3:0]			
RW	RW	R W	R W	R W	R W	R W	R W	RW	RW	R W	R W	R W	R W	R W	R W

位 31	MASK4: 闹钟 A 日期掩码 0: 如果日期/日匹配, 则闹钟 A 置 1 1: 在闹钟 A 比较中, 日期/日无关
位 30	WKSEL: 星期几选择 0: DU[3:0] 代表日期的个位 1: DU[3:0] 代表星期几。DT[1:0] 为无关位。
位 29:28	DT[1:0]: 日期的十位 (BCD 格式)
位 27:24	DU[3:0]: 日期的个位或日 (BCD 格式)
位 23	MASK3: 闹钟 A 小时掩码 0: 如果小时匹配, 则闹钟 A 置 1 1: 在闹钟 A 比较中, 小时无关
位 22	AMPM: AM/PM 符号 0: AM 或 24 小时制 1: PM
位 21:20	HT[1:0]: 小时的十位 (BCD 格式)
位 19:16	HU[3:0]: 小时的个位 (BCD 格式)
位 15	MASK2: 闹钟 A 分钟掩码 0: 如果分钟匹配, 则闹钟 A 置 1 1: 在闹钟 A 比较中, 分钟无关
位 14:12	MT[2:0]: 分钟的十位 (BCD 格式)
位 11:8	MU[3:0]: 分钟的个位 (BCD 格式)
位 7	MASK1: 闹钟 A 秒掩码 0: 如果秒匹配, 则闹钟 A 置 1

	1: 在闹钟 A 比较中，秒无关
位 6:4	ST[2:0]: 秒的十位 (BCD 格式)
位 3:0	SU[3:0]: 秒的个位 (BCD 格式)

注意：仅当 ERTC\_STS 中的 ALAWF 置 1 时或在初始化模式下，才可以对该寄存器执行写操作。

此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.7 ERTC写保护寄存器(ERTC\_WPR)

偏移地址: 0x24

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES								KEY[7:0]							

位 31:8	保留，必须保持复位值
位 7:0	<p>KEY[7:0]: 写保护关键字 可通过软件对该字节执行写操作。 读取该字节时，始终返回 0x00。 有关如何解锁 ERTC 寄存器写保护的介绍，请参见 <a href="#">13.3.4 节</a> ERTC 寄存器写保护。</p>

### 13.6.8 ERTC亚秒寄存器(ERTC\_SBSR)

偏移地址: 0x28

上电复位值: 0x0000 0000

系统复位: 当 BYPSHDW = 0 时为 0x0000 0000; 当 BYPSHDW = 1 时不受影响。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SBS[15:0]															

位 31:16	保留
位 15:0	<p>SBS: 亚秒值 SBS[15:0] 是同步预分频器计数器的值。 此亚秒值可根据以下公式得出： 亚秒值 = ( PRDIV_S - SBS ) / ( PRDIV_S + 1 ) 注意：仅当执行平移操作之后，SBS 才能大于 PRDIV_S。在这种情况下，正确的时间 / 日期比 ERTC_TIME/ERTC_DATE 所指示的时间 / 日期慢一秒钟。</p>

注意：为保证读取到的时间值为最新值，建议读取ERTC\_SBSR前读取一次ERTC\_STS中RSF位（寄存器同步标志）。

### 13.6.9 ERTC平移控制寄存器(ERTC\_SFCTR)

偏移地址: 0x2C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD1S	RES														
W	RES														

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	SUBSBS[14:0]														
	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

位 31	ADD1S: 增加一秒钟 0: 无作用 1: 对时钟/日历增加一秒钟 此位为只写位且始终读为 0。当平移操作挂起 (ERTC_STS 中的 SFP=1) 时，对该位执行写操作无作用。 此函数应与 SUBSBS 配合使用（请参见下文介绍），以便有效地向原子操作机制的时钟添加亚秒值。
位 31:15	保留
位 14:0	SUBSBS: 减少亚秒值 此位为只写位且始终读为 0。当平移操作挂起 (ERTC_STS 中的 SFP=1) 时，对该位执行写操作无作用。 写入 SUBSBS 的值将加到同步预分频器计数器中。由于该计数器递减计数，此操作可有效地从时钟减去（延迟）以下时间： $\text{延迟 (秒)} = \text{SUBSBS} / (\text{PRDIV\_S} + 1)$ 当 ADD1S 函数与 SUBSBS 结合使用时，可有效地将亚秒值增加到时钟（提前时钟），使时钟提前以下时间： $\text{提前 (秒)} = (1 - (\text{SUBSBS} / (\text{PRDIV\_S} + 1)))$ 注意：对 SUBSBS 执行写操作将使 RSF 清零。软件随后会等待至 RSF=1 以确定影子寄存器已更新为平移后的时间。 请参见第 13.3.7 节：ERTC 同步。

注意：此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.10 ERTC时间戳时间寄存器(ERTC\_TSTM)

偏移地址: 0x30

上电复位值: 0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES									AMPM	HT[1:0]		HU[3:0]			
15	14	13	12	11	10	9	8	7	R	R	R	R	R	R	R
RES	MT[2:0]		MU[3:0]				RES	ST[2:0]			SU[3:0]				
	R	R	R	R	R	R		R	R	R	R	R	R	R	R

位 31:23	保留，必须保持复位值
位 22	AMPM: AM/PM 符号 0: AM 或 24 小时制 1: PM
位 21:20	HT[1:0]: 小时的十位 (BCD 格式)
位 19:16	HU[3:0]: 小时的个位 (BCD 格式)
位 15	保留，必须保持复位值。
位 14:12	MT[2:0]: 分钟的十位 (BCD 格式)
位 11:8	MU[3:0]: 分钟的个位 (BCD 格式)
位 7	保留，必须保持复位值。
位 6:4	ST[2:0]: 秒的十位 (BCD 格式)
位 3:0	SU[3:0]: 秒的个位 (BCD 格式)

注意：仅当 ERTC\_STS 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

### 13.6.11 ERTC时间戳日期寄存器(ERTC\_TSDT)

偏移地址: 0x34

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WK[1:0]	MT	MU[3:0]						RES		DT[1:0]		DU[3:0]			
R	R	R	R	R	R	R	R			R	R	R	R	R	R

位 31:16	保留, 必须保持复位值
位 15:13	WK[2:0]: 星期几
位 12	MT: 月份的十位 (BCD 格式)
位 11:8	MU[3:0]: 月份的个位 (BCD 格式)
位 7:6	保留, 必须保持复位值。
位 5:4	DT[1:0]: 日期的十位 (BCD 格式)
位 3:0	DU[3:0]: 日期的个位 (BCD 格式)

注意: 仅当 ERTC\_STS 中的 TSF 置 1 时, 该寄存器的内容才有效。当 TSF 位复位时, 清零该寄存器。

### 13.6.12 ERTC时间戳亚秒寄存器(ERTC\_TSSBS)

偏移地址: 0x38

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SBS[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

位 31:16	保留
位 15:0	SBS: 亚秒值 当发生时间戳事件时, SBS[15:0] 是同步预分频器计数器的值。

注意: 仅当 ERTC\_STS 中 TSF 置 1 时, 该寄存器的内容才有效。当 TSF 位复位时, 清零该寄存器。

### 13.6.13 ERTC校准寄存器(ERTC\_CCR)

偏移地址: 0x3C

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CALAD	CAL8	CAL16	RES				CALM[8:0]								
RW	RW	RW	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31:16	保留
位 15	CALAD: 将 ERTC 的频率增加 488.5 ppm 0: 不增加 ERTCCLK 脉冲。 1: 每 $2^{11}$ 个脉冲有效插入一个 ERTCCLK 脉冲（将频率增加 488.5 ppm）。 此功能应与 CALM 结合使用，后者在高分辨率下会降低日历的频率。如果输入频率为 32768 Hz，则在 32 秒窗口中增加的 ERTCCLK 脉冲数按如下公式计算: $(512 * \text{CALAD}) - \text{CALM}$ 。 请参见第 <a href="#">13.3.9 节: ERTC 精密数字校准</a> 。
位 14	CAL8: 使用 8 秒校准周期 当 CAL8 置“1”时，选择 8 秒校准周期。 CAL8=“1”时，CALM[1:0] 将始终保持为“00”。 请参见第 <a href="#">13.3.9 节: ERTC 精密数字校准</a> 。
位 13	CAL16: 使用 16 秒校准周期 当 CAL16 置“1”时，选择 16 秒校准周期。如果 CAL8=1，则必须将该位置“1”。 注意：当 CAL16=“1”时，CALM[0] 将始终保持为“0”。 请参见第 <a href="#">13.3.9 节: ERTC 精密数字校准</a> 。
位 12:9	保留
位 8:0	CALM[8:0]: 负校准 在 $2^{20}$ 个 ERTCCLK 脉冲内屏蔽 CALM 个脉冲（如果输入频率为 32768 Hz，则为 32 秒）来降低日历的频率。其分辨率为 0.9537 ppm。 要提高日历的频率，则应将此功能与 CALAD 结合使用。请参见第 <a href="#">13.3.9 节: ERTC 精密数字校准</a> 。

### 13.6.14 ERTC 入侵和复用功能配置寄存器(ERTC\_TPAF)

偏移地址: 0x40

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	2 2	2 1	20	19	18	17	16
RES													ALAOUTTYP E	RES	RES
													RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMPUDI S	TMPRCH[1: 0]		TMFLT[1: 0]		TMFREQ[2:0]			TMT S	RES	RE S	RE S	TMIE		TM1TR G	TM1 E
RW	RW	RW	RW	RW	R W	R W	R W	RW		RW	RW	RW		RW	RW

位 31:19	保留
位 18	ALAOUTTYPE: ERTC_ALARM 输出类型 0: ERTC_ALARM 为开漏输出 1: ERTC_ALARM 为推挽输出
位 17:16	保留
位 15	TMPUDIS: TAMPER 禁止内部上拉 该位决定在每次采样之前是否对入侵引脚都进行预充电。 0: 采样之前对入侵引脚进行预充电（使能内部上拉） 1: 禁止对入侵引脚进行预充电
位 14:13	TMFRCH[1:0]: 入侵预充电持续时间 这些位决定了在每次采样之前激活上拉的持续时间。TMFRCH 对每个入侵输入都有效。 0x0: 1 个 ERTCCLK 周期 0x1: 2 个 ERTCCLK 周期 0x2: 4 个 ERTCCLK 周期 0x3: 8 个 ERTCCLK 周期
位 12:11	TMFLT[1:0]: 入侵过滤器计数 这些位决定了为激活入侵事件所需的指定电平 (TAMP*TRG) 上的连续采样次数。TMFLT 对每个入侵输入都有效。 0x0: 在入侵输入转变为有效电平的边沿激活入侵（入侵输入上无内部上拉）。

	0x1: 在有效电平上连续执行 2 次采样后激活入侵。 0x2: 在有效电平上连续执行 4 次采样后激活入侵。 0x3: 在有效电平上连续执行 8 次采样后激活入侵。
位 10:8	TMREQ[2:0]: 入侵采样频率 决定对每个入侵输入进行采样时的频率。 0x0: ERTCCLK / 32768 (ERTCCLK = 32768 Hz 时为 1 Hz) 0x1: ERTCCLK / 16384 (ERTCCLK = 32768 Hz 时为 2 Hz) 0x2: ERTCCLK / 8192 (ERTCCLK = 32768 Hz 时为 4 Hz) 0x3: ERTCCLK / 4096 (ERTCCLK = 32768 Hz 时为 8 Hz) 0x4: ERTCCLK / 2048 (ERTCCLK = 32768 Hz 时为 16 Hz) 0x5: ERTCCLK / 1024 (ERTCCLK = 32768 Hz 时为 32 Hz) 0x6: ERTCCLK / 512 (ERTCCLK = 32768 Hz 时为 64 Hz) 0x7: ERTCCLK / 256 (ERTCCLK = 32768 Hz 时为 128 Hz)
位 7	TMTS: 发生入侵检测事件时激活时间戳 0: 发生入侵检测事件时不保存时间戳 1: 发生入侵检测事件时保存时间戳 即便 ERTC_CTRL 寄存器中的 TSE=0, TMTS 仍有效。
位 6:5	保留。始终读为 0
位 4:3	保留
位 2	TMIE: 入侵中断使能 0: 禁止入侵中断 1: 使能入侵中断
位 1	TM1TRG: 入侵 1 的有效电平 如果 TMFLT ≠ 00 0: TAMPER1 保持低电平会触发入侵检测事件。 1: TAMPER1 保持高电平会触发入侵检测事件。 如果 TMFLT = 00: 0: TAMPER1 上升沿会触发入侵检测事件。 1: TAMPER1 下降沿会触发入侵检测事件。 注意: 如果 TMFLT = 0, 则更改 TM1TRG 时必须复位 TM1E, 以避免将 TPF 意外置 1。
位 0	TM1E: 入侵 1 检测使能 0: 禁止入侵 1 检测 1: 使能入侵 1 检测

### 13.6.15 ERTC闹钟A亚秒寄存器(ERTC\_ALASBS)

偏移地址: 0x44

上电复位值: 0x0000 0000

系统复位: 不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES				MASKSBS[3:0]				RES							
R	R	R	R	RW	RW	RW	RW	R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	SBS[14:0]														
R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31:28	保留
位 27:24	MASKSBS[3:0]: 屏蔽从此位开始的最高有效位 0: 不对闹钟 A 的亚秒进行比较。当秒单元递增时设置闹钟（假定其余字段均匹配）。 1: 在闹钟 A 比较中, SBS[14:1] 为无关位。仅比较 SBS [0]。 2: 在闹钟 A 比较中, SBS [14:2] 为无关位。仅比较 SBS [1:0]。 3: 在闹钟 A 比较中, SBS [14:3] 为无关位。仅比较 SBS [2:0]。 ... 12: 在闹钟 A 比较中, SBS [14:12] 为无关位。比较 SBS [11:0]。 13: 在闹钟 A 比较中, SBS [14:13] 为无关位。比较 SBS [12:0]。 14: 在闹钟 A 比较中, SBS [14] 为无关位。比较 SBS [13:0]。 15: 所有 15 个 SBS 位均进行比较, 并且必须全部匹配才能激活闹钟。同步计数器的溢出位 (位 15) 从不进行比较。仅当执行平移操作之后, 此位才不为 0。
位 23:15	保留

位 14:0	SBS[14:0]: 亚秒值 该值与同步预分频器计数器的内容进行比较以确定是否要激活闹钟 A。仅比较位 0 到 MASKSBS-1。
--------	---

注意：仅当 ERTC\_CTRL 中 ALAE 复位或初始化模式下，才可以对该寄存器执行写操作。

此寄存器受写保护。第 [13.3.4 节](#) 的 ERTC 寄存器写保护中介绍了写访问的过程。

### 13.6.16 ERTC备份寄存器(ERTC\_BKPxDT)

偏移地址：0x50 到 0x60

上电复位值：0x0000 0000

系统复位：不受影响

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D[31:16]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31:0	D[31:0] 应用可向/从这些寄存器写入/读取数据。 系统复位时，这些寄存器不会复位，并且当器件在低功耗模式下工作时，寄存器的内容仍然有效。 发生入侵检测事件时该寄存器会被复位，并且只要 TAMPxF=1，该寄存器就一直保持复位。
--------	---

# 14 模拟/数字转换 (ADC)

## 14.1 ADC介绍

12位ADC是一种逐次逼近型模拟数字转换器。它有多达18个通道，可测量16个外部和2个内部信号源。各通道的A/D转换可以单次、连续、扫描或间断模式执行。ADC的结果可以左对齐或右对齐方式存储在16位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阈值。

ADC的输入时钟不得超过28MHz，它是由PCLK2经分频产生，参见图3-2。

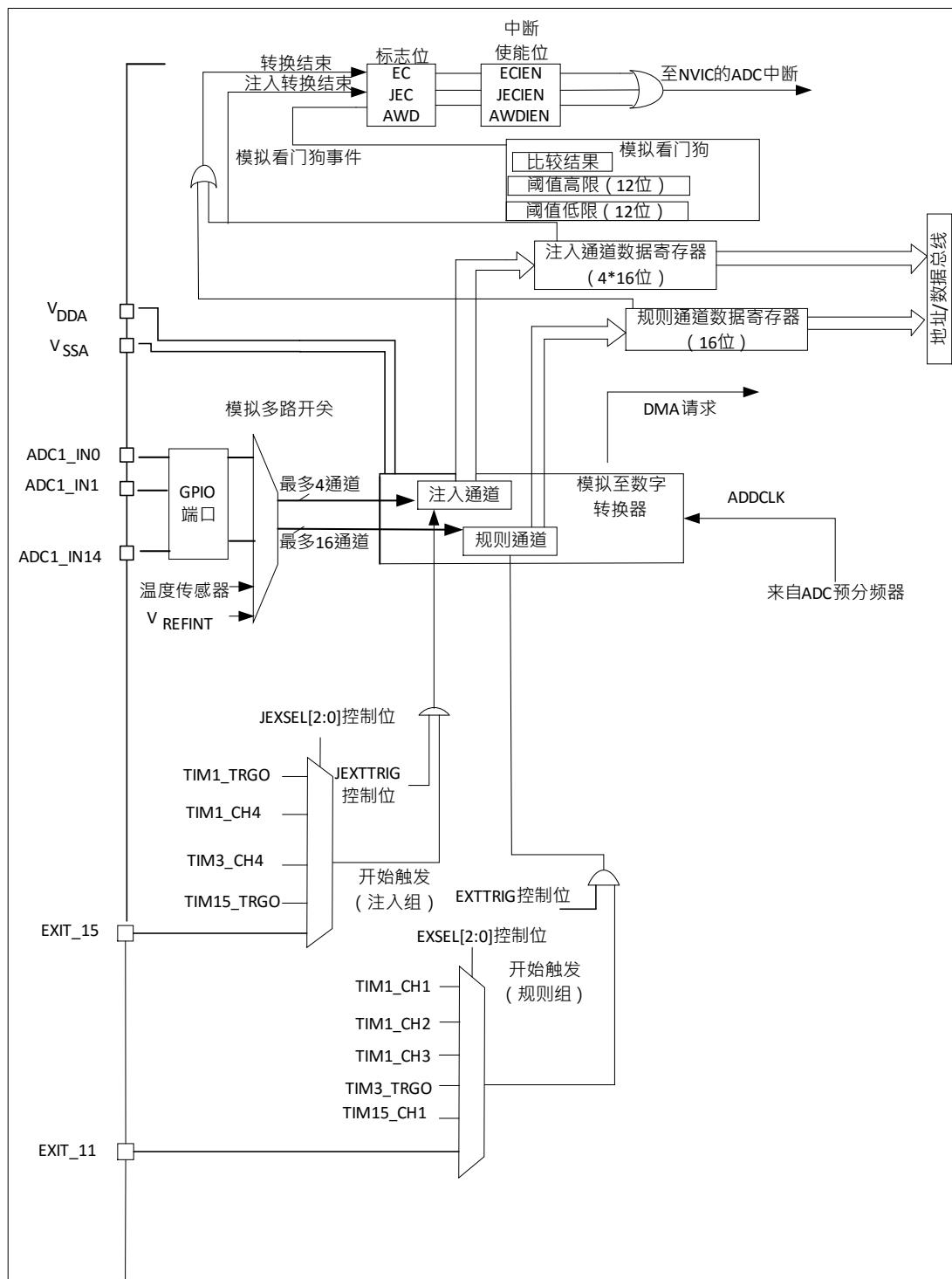
## 14.2 ADC主要特征

- 12位分辨率
- 转换结束、注入转换结束和发生模拟看门狗事件时产生中断
- 单次和连续转换模式
- 从通道0到通道n的自动扫描模式
- 自校准时间：156个ADC时钟周期
- 带内嵌数据一致性的数据对齐
- 采样间隔可以按通道分别编程
- 规则转换和注入转换均有外部触发选项
- 间断模式
- ADC转换时间
  - 时钟为28MHz时为0.5 μs
- ADC供电要求：2.4V到3.6V
- ADC输入范围： $V_{SSA} \leq V_{IN} \leq V_{DDA}$
- 规则通道转换期间有DMA请求产生

## 14.3 ADC功能描述

下图为一个ADC模块的框图，表14-1为ADC引脚的说明。

图表 14-1 单个ADC框图



表格 14-1 ADC引脚

名称	信号类型	注解
V <sub>DDA</sub> <sup>(1)</sup>	输入, 模拟电源	等效于 V <sub>DD</sub> 的模拟电源且: 2.4V ≤ V <sub>DDA</sub> ≤ V <sub>DD</sub> (3.6V)
V <sub>SSA</sub> <sup>(1)</sup>	输入, 模拟电源地	等效于 V <sub>SS</sub> 的模拟电源地
ADC1_IN[14: 0]	模拟输入信号	15 个模拟输入通道

注意： V<sub>DDA</sub> 和 V<sub>SSA</sub> 应该分别连接到 V<sub>DD</sub> 和 V<sub>SS</sub>。

### 14.3.1 ADC开关控制

通过设置 ADC\_CTRL2 寄存器的 ADON 位可给 ADC 上电。当第一次设置 ADON 位时，它将 ADC 从断

电状态下唤醒。

ADC 上电延迟一段时间后 ( $t_{STAB}$ )，再次设置 ADON 位时开始进行转换。通过清除 ADON 位可以停止转换，并将 ADC 置于断电模式。在这个模式中，ADC 几乎不耗电（仅几个  $\mu A$ ）。

### 14.3.2 ADC时钟

由时钟控制器提供的 ADCCLK 时钟和 PCLK2 (APB2 时钟) 同步。RCC 控制器为 ADC 时钟提供一个专用的可编程预分频器，详见复位和时钟控制 (RCC) 章节。

### 14.3.3 通道选择

有 18 个多路通道。可以把转换组织成两组：规则组和注入组。在任意多个通道上以任意顺序进行的一系列转换构成组转换。例如，可以如下顺序完成转换：通道 3、通道 8、通道 2、通道 2、通道 0、通道 2、通道 2。

- 规则组由多达 16 个转换组成。规则通道和它们的转换顺序在 ADC\_RSQx 寄存器中选择。规则组中转换的总数应写入 ADC\_RSQ1 寄存器的 LEN[3: 0] 位中
- 注入组由多达 4 个转换组成。注入通道和它们的转换顺序在 ADC\_JSQ 寄存器中选择。注入组里的转换总数目应写入 ADC\_JSQ 寄存器的 LEN[1: 0] 位中

如果 ADC\_RSQx 或 ADC\_JSQ 寄存器在转换期间被更改，当前的转换被清除，一个新的启动脉冲将发送到 ADC 以转换新选择的组。

#### 温度传感器 VREFINT 内部通道

温度传感器和通道 ADC1\_IN16 相连接，内部参照电压 VREFINT 和 ADC1\_IN17 相连接。可以按注入或规则通道对这两个内部通道进行转换。

### 14.3.4 单次转换模式

单次转换模式下，ADC 只执行一次转换。该模式既可通过设置 ADC\_CTRL2 寄存器的 ADON 位（只适用于规则通道）启动也可通过外部触发启动（适用于规则通道或注入通道），这时 CON 位为 0。

一旦选择通道的转换完成：

- 如果一个规则通道被转换
  - 转换数据被储存在 16 位 ADC\_RDOR 寄存器中
  - EC (转换结束) 标志被设置
  - 如果设置了 ECIEN，则产生中断
- 如果一个注入通道被转换
  - 转换数据被储存在 16 位的 ADC\_JDOR1 寄存器中
  - JEC (注入转换结束) 标志被设置
  - 如果设置了 JECIEN 位，则产生中断

然后 ADC 停止。

### 14.3.5 连续转换模式

在连续转换模式中，当前面 ADC 转换一结束马上就启动另一次转换。此模式可通过外部触发启动或通过设置 ADC\_CTRL2 寄存器上的 ADON 位启动，此时 CON 位是 1。

每个转换后：

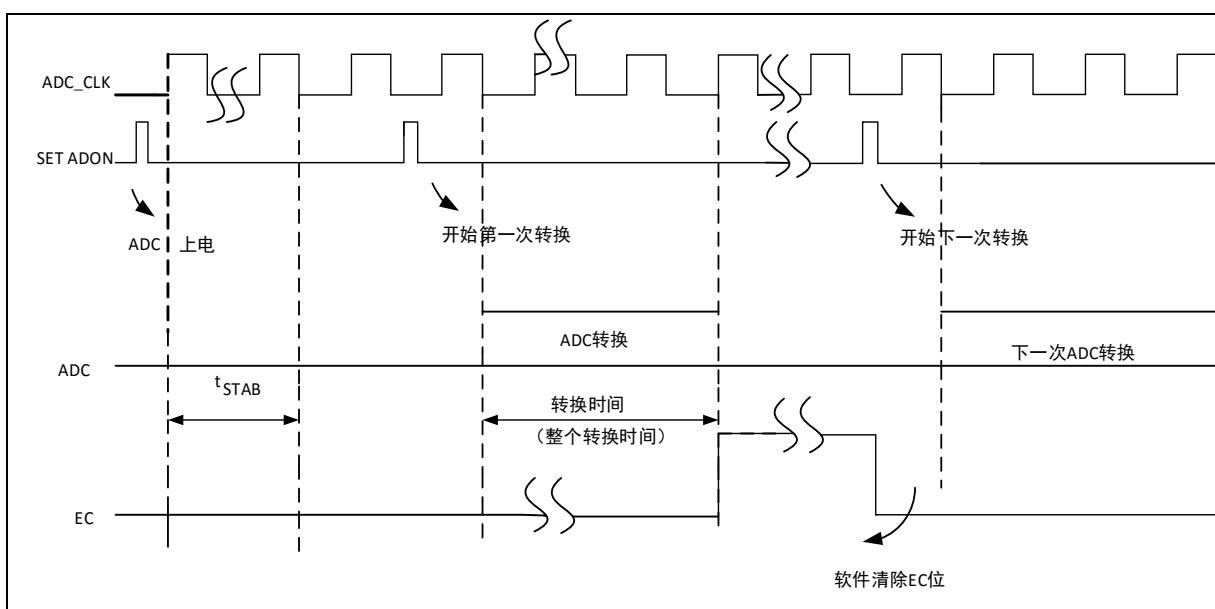
- 如果一个规则通道被转换
  - 转换数据被储存在 16 位的 ADC\_RDOR 寄存器中
  - EC (转换结束) 标志被设置
  - 如果设置了 ECIEN，则产生中断
- 如果一个注入通道被转换

- 转换数据被储存在16位的ADC\_JDOR1寄存器中
- JEC（注入转换结束）标志被设置
- 如果设置了JECIEN位，则产生中断

### 14.3.6 时序图

如下图所示，ADC 在开始精确转换前需要一个稳定时间  $t_{STAB}$ 。在开始 ADC 转换和 14 个时钟周期后，EC 标志被设置，16 位 ADC 数据寄存器包含转换的结果。

图表 14-2 时序图



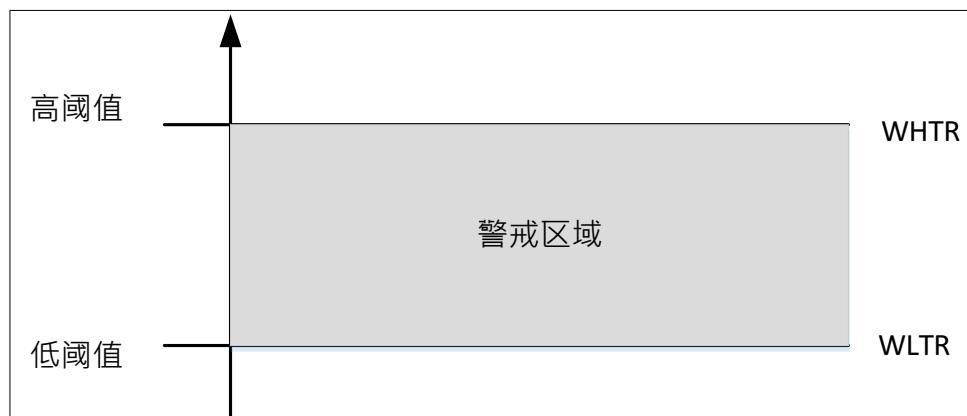
### 14.3.7 模拟看门狗

如果被 ADC 转换的模拟电压低于低阈值或高于高阈值，AWD 模拟看门狗状态位被设置。阈值位于 ADC\_WHTR 和 ADC\_WLTR 寄存器的最低 12 个有效位中。通过设置 ADC\_CTRL1 寄存器的 AWDIEN 位以允许产生相应中断。

阈值独立于由 ADC\_CTRL2 寄存器上的 DALIGN 位选择的数据对齐模式。比较是在对齐之前完成的（见 [14.3.12 节](#)）。

通过配置 ADC\_CTRL1 寄存器，模拟看门狗可以作用于 1 个或多个通道，如表 14-2 所示。

图表 14-3 模拟看门狗警戒区



表格 14-2 模拟看门狗通道选择

模拟看门狗警戒的通道	ADC_CTRL1 寄存器控制位
------------	------------------

	AWDSGE 位	AWDEN 位	JAWDEN 位
无	任意值	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有注入和规则通道	0	1	1
单一的 <sup>(1)</sup> 注入通道	1	0	1
单一的 <sup>(1)</sup> 规则通道	1	1	0
单一的 <sup>(1)</sup> 注入或规则通道	1	1	1

注意：由 AWDCS[4: 0]位选择。

### 14.3.8 扫描模式

此模式用来扫描一组模拟通道。

扫描模式可通过设置 ADC\_CTRL1 寄存器的 SCN 位来选择。一旦这个位被设置，ADC 扫描所有被 ADC\_RSQX 寄存器（对规则通道）或 ADC\_JSQ（对注入通道）选中的所有通道。在每个组的每个通道上执行单次转换。在每个转换结束时，同一组的下一个通道被自动转换。如果设置了 CON 位，转换不会在选择组的最后一个通道上停止，而是再次从选择组的第一个通道继续转换。

如果设置了 DMAEN 位，在每次 RDOR 寄存器更新后，DMA 控制器把规则组通道的转换数据传输到 SRAM 中。而注入通道转换的数据总是存储在 ADC\_JDORx 寄存器中。

### 14.3.9 注入通道管理

#### 触发注入

清除 ADC\_CTRL1 寄存器的 JAUT 位，并且设置 SCN 位，即可使用触发注入功能。

- 利用外部触发或通过设置 ADC\_CTRL2 寄存器的 ADON 位，启动一组规则通道的转换。
- 如果在规则通道转换期间产生一外部注入触发，当前转换被复位，注入通道序列被以单次扫描方式进行转换。
- 然后，恢复上次被中断的规则组通道转换。如果在注入转换期间产生一规则事件，注入转换不会被中断，但是规则序列将在注入序列结束后被执行。[图 14-4](#) 是其定时图。

注意：当使用触发的注入转换时，必须保证触发事件的间隔长于注入序列。例如：序列长度为 28 个 ADC 时钟周期（即 2 个具有 1.5 个时钟间隔采样时间的转换），触发之间最小的间隔必须是 29 个 ADC 时钟周期。

#### 自动注入

如果设置了 JAUT 位，在规则组通道之后，注入组通道被自动转换。这可以用来转换在 ADC\_RSQx 和 ADC\_JSQ 寄存器中设置的多至 20 个转换序列。

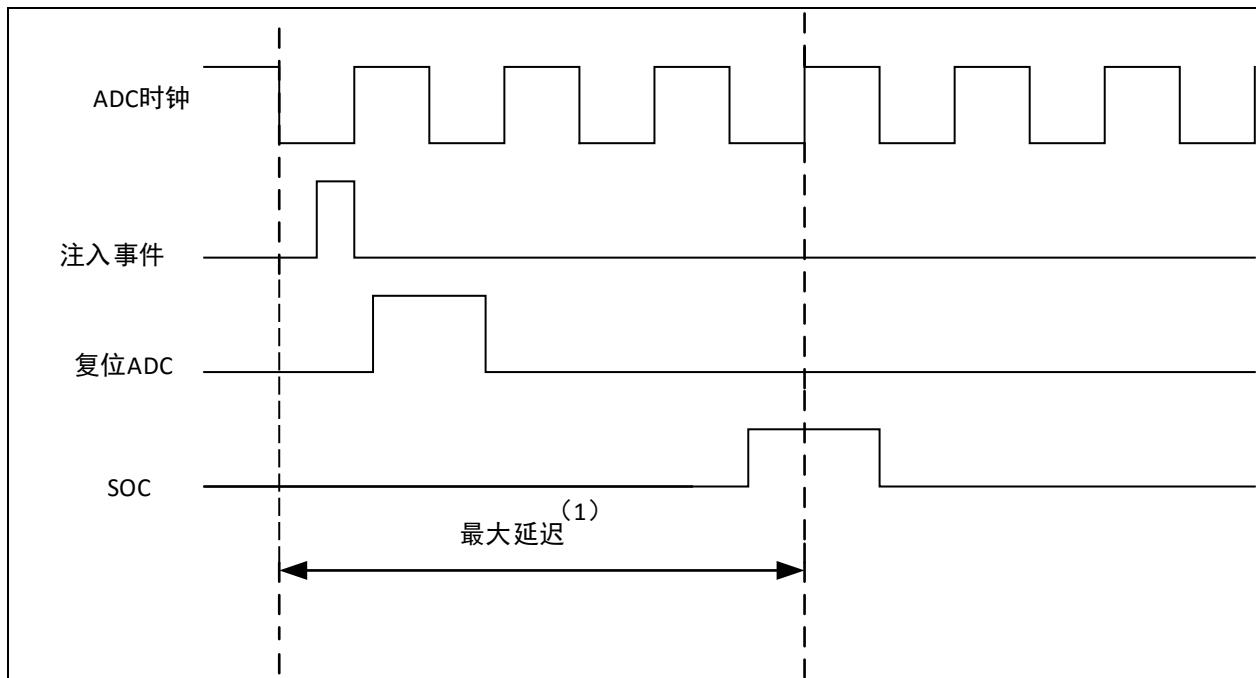
在此模式里，必须禁止注入通道的外部触发。

如果除 JAUT 位外还设置了 CON 位，规则通道至注入通道的转换序列被连续执行。

对于 ADC 时钟预分频系数为 4 至 8 时，当从规则转换切换到注入序列或从注入转换切换到规则序列时，会自动插入 1 个 ADC 时钟间隔；当 ADC 时钟预分频系数为 2 时，则有 2 个 ADC 时钟间隔的延迟。

注意：不可能同时使用自动注入和间断模式。

图表 14-4 注入转换延时



注意： 最大延迟数值请参考 AT32F421 系列数据手册中有关电气特性部分。

### 14.3.10 间断模式

#### 规则组

此模式通过设置 ADC\_CTRL1 寄存器上的 RDISEN 位激活。它可以用来执行一个短序列的  $n$  次转换 ( $n \leq 8$ )，此转换是 ADC\_RSQx 寄存器所选择的转换序列的一部分。数值  $n$  由 ADC\_CTRL1 寄存器的 DISN[2: 0]位给出。

一个外部触发信号可以启动 ADC\_RSQx 寄存器中描述的下一轮  $n$  次转换，直到此序列所有的转换完成为止。总的序列长度由 ADC\_RSQ1 寄存器的 LEN[3: 0]定义。

举例：

$n=3$ , 被转换的通道 = 0、1、2、3、6、7、9、5

第一次触发：转换的序列为 0、1、2，并产生 EC 事件

第二次触发：转换的序列为 3、6、7，并产生 EC 事件

第三次触发：转换的序列为 9、5，并产生 EC 事件

第四次触发：转换的序列为 0、1、2，并产生 EC 事件

注意：当以间断模式转换一个规则组时，转换序列结束后不自动从头开始。当所有子组被转换完成，下一次触发启动第一个子组的转换。在上面的例子中，第四次触发重新转换第一子组的通道 0、1 和 2。

#### 注入组

此模式通过设置 ADC\_CTRL1 寄存器的 JDISEN 位激活。在一个外部触发事件后，该模式按通道顺序逐个转换 ADC\_JSQ 寄存器中选择的序列。

一个外部触发信号可以启动 ADC\_JSQ 寄存器选择的下一个通道序列的转换，直到序列中所有的转换完成为止。总的序列长度由 ADC\_JSQ 寄存器的 JLEN[1: 0]位定义。

例子：

$n=1$ , 被转换的通道 = 1、2、3

第一次触发：通道 1 被转换

第二次触发：通道 2 被转换

第三次触发：通道 3 被转换，并且产生 EC 和 JEC 事件

第四次触发：通道 1 被转换

- 注意：
1. 当完成所有注入通道转换，下个触发启动第 1 个注入通道的转换。在上述例子中，第四个触发重新转换第 1 个注入通道 1。
  2. 不能同时使用自动注入和间断模式。
  3. 必须避免同时为规则和注入组设置间断模式。间断模式只能作用于一组转换。

### 14.3.11 校准

ADC 有一个内置自校准模式。校准可大幅减小因内部电容器组的变化而造成的精度误差。在校准期间，在每个电容器上都会计算出一个误差修正码（数字值），这个码用于消除在随后的转换中每个电容器上产生的误差。

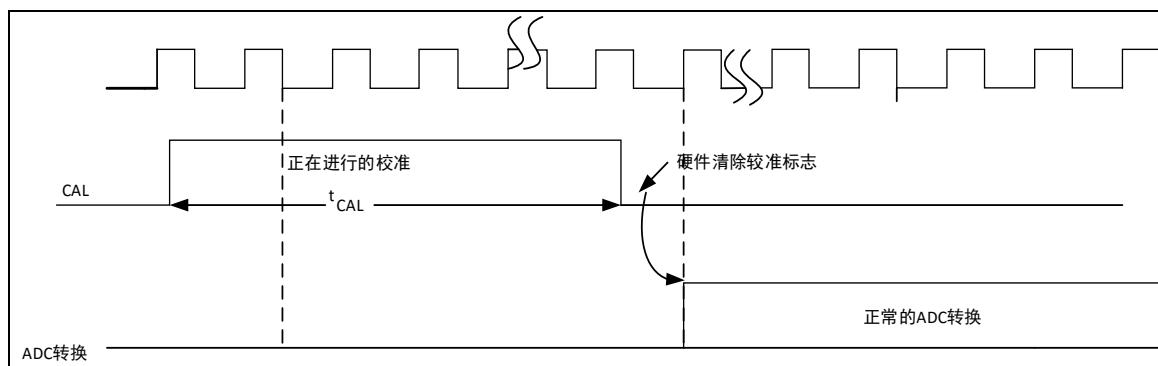
通过设置 ADC\_CTRL2 寄存器的 CAL 位启动校准。一旦校准结束，CAL 位被硬件复位，可以开始正常转换。建议在上电时执行一次 ADC 校准。校准阶段结束后，校准码储存在 ADC\_RDOR 中。

注意：

1. 建议在每次上电后执行一次校准。

2. 启动校准前，ADC 必须处于上电状态 (**ADON=’1’**) 超过至少两个 ADC 时钟周期。

图表 14-5 校准时序图



### 14.3.12 数据对齐

ADC\_CTRL2 寄存器中的 DALIGN 位选择转换后数据储存的对齐方式。数据可以左对齐或右对齐，如图 14-6 和图 14-7 所示。

注入组通道转换的数据值已经减去了在 ADC\_JOFSx 寄存器中定义的偏移量，因此结果可以是一个负值。SEXT 位是扩展的符号值。

对于规则组通道，不需减去偏移值，因此只有 12 个位有效。

图表 14-6 数据右对齐

注入组

SEXT	SEXT	SEXT	SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
------	------	------	------	-----	-----	----	----	----	----	----	----	----	----	----	----

规则组

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
---	---	---	---	-----	-----	----	----	----	----	----	----	----	----	----	----

图表 14-7 数据左对齐

注入组

SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0
------	-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---

规则组

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---	---

### 14.3.13 可编程的通道采样时间

ADC 使用若干个 ADC\_CLK 周期对输入电压采样，采样周期数目可以通过 ADC\_SMPT1 和 ADC\_SMPT2 寄存器中的 SMP[2: 0]位更改。每个通道可以分别用不同的时间采样。总转换时间如下计算：

$$T_{CONV} = \text{采样时间} + 12.5 \text{ 个周期}$$

例如：

当 ADCCLK=14MHz，采样时间为 1.5 周期

$$T_{CONV} = 1.5 + 12.5 = 14 \text{ 周期} = 1 \mu\text{s}$$

### 14.3.14 外部触发转换

转换可以由外部事件触发（例如定时器捕获，EXTI 线）。如果设置了 EXTTRIG 控制位，则外部事件就能够触发转换。EXSEL[2: 0]和 JEXSEL[2: 0]控制位允许应用程序选择多个可能的事件中的某一个，可以触发规则和注入组的采样。

注意：当外部触发信号被选为 ADC 规则或注入转换时，只有它的上升沿可以启动转换。

表格 14-3 ADC1 用于规则通道的外部触发

触发源	连接类型	EXSEL[2: 0]
TMR1_CC1 事件	来自片上定时器的内部信号	000
TMR1_CC2 事件		001
TMR1_CC3 事件		010
TMR3_TRGO 事件		100
TMR15_CC1 事件		101
EXTI 线 11	外部引脚	110
SWSTR	软件控制位	111

表格 14-4 ADC1 用于注入通道的外部触发

触发源	连接类型	JEXSEL[2: 0]
TMR1_TRGO 事件	来自片上定时器的内部信号	000
TMR1_CC4 事件		001
TMR3_CC4 事件		100
TMR15_TRGO 事件		101
EXTI 线 15	外部引脚	110
JSWSTR	软件控制位	111

软件触发事件通过对寄存器 ADC\_CTRL2 的 SWSTR 或 JSWSTR 位置‘1’产生。

规则组的转换可以被注入触发打断。

### 14.3.15 DMA请求

因为规则通道转换的值储存在一个仅有的数据寄存器中，所以当转换多个规则通道时需要使用 DMA，这可以避免丢失已经存储在 ADC\_RDOR 寄存器中的数据。

只有在规则通道的转换结束时才产生 DMA 请求，并将转换的数据从 ADC\_RDOR 寄存器传输到用户指定的目的地址。

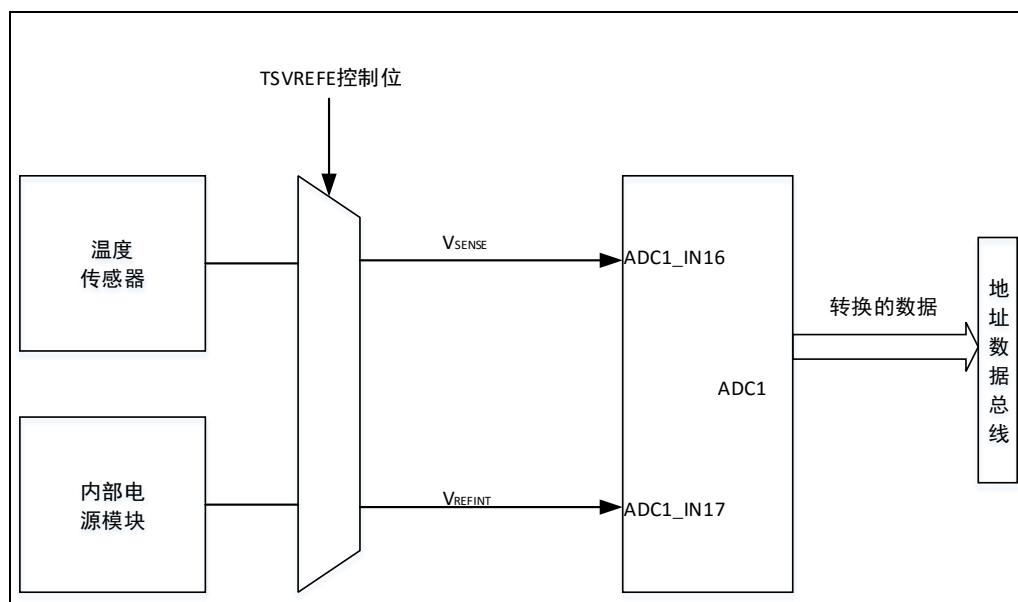
### 14.3.16 温度传感器

温度传感器可以用来测量器件周围的温度 ( $T_A$ )。温度传感器在内部和 ADC1\_IN16 输入通道相连接，此通道把传感器输出的电压转换成数字值。温度传感器模拟输入推荐采样时间是  $8.6 \mu s$ 。[图 14-18](#) 是温度传感器的方框图。当没有被使用时，传感器可以置于关电模式。

**注意：** 必须设置 TSREF 位激活内部通道：ADC1\_IN16（温度传感器）和 ADC1\_IN17（VREFINT）的转换。

温度传感器输出电压随温度线性变化，由于生产过程的变化，温度变化曲线的偏移在不同芯片上会有不同（最多相差  $45^\circ C$ ）。内部温度传感器更适合于检测温度的变化，而不是测量绝对的温度。如果需要测量精确的温度，应该使用一个外置的温度传感器。

图表 14-8 温度传感器和 V<sub>REFINT</sub> 通道框图



#### 读温度

为使用传感器：

1. 选择 ADC1\_IN16 输入通道。
2. 选择采样时间为  $8.6 \mu s$ 。
3. 设置 ADC 控制寄存器 2 (ADC\_CTRL2) 的 TSREF 位，以唤醒关电模式下的温度传感器。
4. 通过设置 ADON 位启动 ADC 转换 (或用外部触发)。
5. 读 ADC 数据寄存器上的 VSENSE 数据结果。
6. 利用下列公式得出温度。

$$\text{温度 } (\text{ }^\circ \text{C}) = \{ (V_{25} - V_{\text{SENSE}}) / \text{Avg\_Slope} \} + 25$$

这里：

$$V_{25} = V_{\text{SENSE}} \text{ 在 } 25^\circ \text{C 时的数值}$$

Avg\_Slope = 温度与 VSENSE 曲线的平均斜率 (单位为 mV/ $^\circ C$  或  $\mu V/\text{ }^\circ C$ )

参考数据手册的电气特性章节中  $V_{25}$  和 Avg\_Slope 的实际值。

**注意：** 传感器从关电模式唤醒后到可以输出正确水平的 VSENSE 前，有一个建立时间。ADC 在上电后也有一个建立时间，因此为了缩短延时，应该同时设置 ADON 和 TSREF 位。

### 14.3.17 ADC 中断

规则和注入组转换结束时能产生中断，当模拟看门狗状态位被设置时也能产生中断。它们都有独立的中断使能位。

ADC\_STS 寄存器中有 2 个其他标志，但是它们没有相关联的中断：

- JSTR (注入组通道转换的启动)
- RSTR (规则组通道转换的启动)

表格 14-5 ADC 中断

中断事件	事件标志	使能控制位
规则组转换结束	EC	ECIEN
注入组转换结束	JEC	JECIEN
设置了模拟看门狗状态位	AWD	AWDIEN

## 14.4 ADC 寄存器

必须以字（32 位）的方式操作这些外设寄存器。

表格 14-6 ADC 寄存器映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	ADC_STS	保留																									RSTR	JSTR	JEC	EC	AWD		
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
004h	ADC_CTRL1	保留						保留						保留						保留						AWDCS[4: 0]							
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
008h	ADC_CTRL2	保留						保留						保留						保留						保留							
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00Ch	ADC_SMPT1	保留						保留						保留						保留						保留							
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
010h	ADC_SMPT2	保留			SMP9[2: 0]			SMP8[2: 0]			SMP7[2: 0]			SMP6[2: 0]			SMP5[2: 0]			SMP4[2: 0]			SMP3[2: 0]			SMP2[2: 0]			SMP1[2: 0]				
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
014h	ADC_JOFS1	保留																		JOFST1[11: 0]													
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
018h	ADC_JOFS2	保留																		JOFST2[11: 0]													
		复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01Ch	ADC_JOFS3	保留																		JOFST3[11: 0]													

#### 14.4.1 ADC状态寄存器 (ADC\_STS)

地址偏移: 0x00

复位值: 0x0000 0000

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

保留

res

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

保留 RSTR JSTR JEC EC AWD

res

rc

1

© w0

rg

1

© w0

rc

位 31, 15

保留，必须保持为0。

位 4	<b>RSTR:</b> 规则通道开始位 (Regular channel Start flag) 该位由硬件在规则通道转换开始时设置, 由软件清除。 0: 规则通道转换未开始; 1: 规则通道转换已开始。
位 3	<b>JSTR:</b> 注入通道开始位 (Injected channel Start flag) 该位由硬件在注入通道组转换开始时设置, 由软件清除。 0: 注入通道组转换未开始; 1: 注入通道组转换已开始。
位 2	<b>JEC:</b> 注入通道转换结束位 (Injected channel end of conversion) 该位由硬件在所有注入通道组转换结束时设置, 由软件清除 0: 转换未完成; 1: 转换完成。
位 1	<b>EC:</b> 转换结束位 (End of conversion) 该位由硬件在(规则或注入)通道组转换结束时设置, 由软件清除或由读取 ADC_RDOR 时清除 0: 转换未完成; 1: 转换完成。
位 0	<b>AWD:</b> 模拟看门狗标志位 (Analog watchdog flag) 该位由硬件在转换的电压值超出了 ADC_WLTR 和 ADC_WHTR 寄存器定义的范围时 设置, 由软件清除。 0: 没有发生模拟看门狗事件; 1: 发生模拟看门狗事件。

#### 14.4.2 ADC控制寄存器1 (ADC\_CTRL1)

地址偏移: 0x04

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留					AWDEN	JAWDEN	保留								
res					rw		rw	res	rw	rw	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISN[2: 0]	JDISEN	RDISEN	JAUT	AWDSGE	SCN	JECIEN	AWDIEN	ECIEN		AWDCS[4: 0]					
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 24	保留。必须保持为 0。
位 23	<b>AWDEN:</b> 在规则通道上开启模拟看门狗 (Analog watchdog enable on regular channels) 该位由软件设置和清除。 0: 在规则通道上禁用模拟看门狗; 1: 在规则通道上使用模拟看门狗。
位 22	<b>JAWDEN:</b> 在注入通道上开启模拟看门狗 (Analog watchdog enable on injected channels) 该位由软件设置和清除。 0: 在注入通道上禁用模拟看门狗; 1: 在注入通道上使用模拟看门狗。
位 21: 16	保留。必须保持为 0。

位 15: 13	<b>DISN[2: 0]:</b> 间断模式通道计数 (Discontinuous mode channel count) 软件通过这些位定义在间断模式下, 收到外部触发后转换规则通道的数目 000: 1 个通道 001: 2 个通道 ..... 111: 8 个通道
位 12	<b>JDISEN:</b> 在注入通道上的间断模式 (Discontinuous mode on injected channels) 该位由软件设置和清除, 用于开启或关闭注入通道组上的间断模式 0: 注入通道组上禁用间断模式; 1: 注入通道组上使用间断模式。
位 11	<b>RDISEN:</b> 在规则通道上的间断模式 (Discontinuous mode on regular channels) 该位由软件设置和清除, 用于开启或关闭规则通道组上的间断模式 0: 规则通道组上禁用间断模式; 1: 规则通道组上使用间断模式。
位 10	<b>JAUT:</b> 自动的注入通道组转换 (Automatic Injected Group conversion) 该位由软件设置和清除, 用于开启或关闭规则通道组转换结束后自动的注入通道组转换 0: 关闭自动的注入通道组转换; 1: 开启自动的注入通道组转换。
位 9	<b>AWDSGE:</b> 扫描模式中在一个单一的通道上使用看门狗 (Enable the watchdog on a single channel in scan mode) 该位由软件设置和清除, 用于开启或关闭由 AWDCS[4: 0]位指定的通道上的模拟看门狗功能 0: 在所有的通道上使用模拟看门狗; 1: 在单一通道上使用模拟看门狗。
位 8	<b>SCN:</b> 扫描模式 (Scan mode) 该位由软件设置和清除, 用于开启或关闭扫描模式。在扫描模式中, 转换由 ADC_RSQx 或 ADC_JSQx 寄存器选中的通道。 0: 关闭扫描模式; 1: 使用扫描模式。 <b>注:</b> 如果分别设置了 ECIEN 或 JECIEN 位, 只在最后一个通道转换完毕后才会产生 EC 或 JEC 中断。
位 7	<b>JECIEN:</b> 允许产生注入通道转换结束中断 (Interrupt enable for injected channels) 该位由软件设置和清除, 用于禁止或允许所有注入通道转换结束后产生中断。 0: 禁止 JEC 中断; 1: 允许 JEC 中断。当硬件设置 JEC 位时产生中断。
位 6	<b>AWDIEN:</b> 允许产生模拟看门狗中断 (Analog watchdog interrupt enable) 该位由软件设置和清除, 用于禁止或允许模拟看门狗产生中断。在扫描模式下, 如果看门狗检测到超范围的数值时, 只有在设置了该位时扫描才会中止。 0: 禁止模拟看门狗中断; 1: 允许模拟看门狗中断。
位 5	<b>ECIEN:</b> 允许产生 EC 中断 (Interrupt enable for EC) 该位由软件设置和清除, 用于禁止或允许转换结束后产生中断。 0: 禁止 EC 中断; 1: 允许 EC 中断。当硬件设置 EC 位时产生中断。

位 4: 0	<p><b>AWDCS[4: 0]:</b> 模拟看门狗通道选择位 (Analog watchdog channel select bits) 这些位由软件设置和清除，用于选择模拟看门狗保护的输入通道。</p> <p>00000: ADC 模拟输入通道 0 00001: ADC 模拟输入通道 1 ..... 01111: ADC 模拟输入通道 15 10000: ADC 模拟输入通道 16 10001: ADC 模拟输入通道 17 保留所有其他数值。 注: ADC1 的模拟输入通道 16 和通道 17 在芯片内部分别连到了温度传感器和 V<sub>REFINT</sub>。</p>
--------	--

#### 14.4.3 ADC控制寄存器2 (ADC\_CTRL2)

地址偏移: 0x08

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留				TSR EF	SWS TR	JSWS TR	EXTT RIG	EXSEL[2:0]			保留				
res				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	res
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JEXT TRIG	JEXSEL[2:0]	DALI GN	保留	DM AEN	保留			RST CAL	CAL	CON	ADON	rw	rw	rw	rw
rw	rw	rw	rw	rw	res	rw	res			rw	rw	rw	rw	rw	rw

位 31: 24	保留。必须保持为 0。
位 23	<p><b>TSREF:</b> 温度传感器和 V<sub>REFINT</sub> 使能 (Temperature sensor and V<sub>REFINT</sub> enable) 该位由软件设置和清除，用于开启或禁止温度传感器和 V<sub>REFINT</sub> 通道。</p> <p>0: 禁止温度传感器和 V<sub>REFINT</sub>; 1: 启用温度传感器和 V<sub>REFINT</sub>。</p>
位 22	<p><b>SWSTR:</b> 开始转换规则通道 (Start conversion of regular channels) 由软件设置该位以启动转换，转换开始后硬件马上清除此位。如果在 EXSEL[2: 0] 位中选择了 SWSTR 为触发事件，该位用于启动一组规则通道的转换， 0: 复位状态; 1: 开始转换规则通道。</p>
位 21	<p><b>JSWSTR:</b> 开始转换注入通道 (Start conversion of injected channels) 由软件设置该位以启动转换，软件可清除此位或在转换开始后硬件马上清除此位。如果在 JEXSEL[2: 0] 位中选择了 JSWSTR 为触发事件，该位用于启动一组注入通道的转换。 0: 复位状态; 1: 开始转换注入通道。</p>
位 20	<p><b>EXTTRIG:</b> 规则通道的外部触发转换模式 (External trigger conversion mode for regular channels) 该位由软件设置和清除，用于开启或禁止可以启动规则通道组转换的外部触发事件。 0: 不用外部事件启动转换; 1: 使用外部事件启动转换。</p>

位 19: 17	<p><b>EXSEL[2: 0]:</b> 选择启动规则通道组转换的外部事件 (External event select for regular group)          这些位选择用于启动规则通道组转换的外部事件  <b>ADC1 的触发配置如下</b>          000: 定时器 1 的 CC1 事件          001: 定时器 1 的 CC2 事件          010: 定时器 1 的 CC3 事件          100: 定时器 3 的 TRGO 事件          101: 定时器 15 的 CC1 事件          110: EXTI 线 11 事件          111: SWSTR          其余: 无</p>
位 16	保留。必须保持为 0。
位 15	<p><b>JEXTTRIG:</b> 注入通道的外部触发转换模式 (External trigger conversion mode for injected channels)          该位由软件设置和清除，用于开启或禁止可以启动注入通道组转换的外部触发事件。          0: 不用外部事件启动转换；          1: 使用外部事件启动转换。</p>
位 14: 12	<p><b>JEXSEL[2: 0]:</b> 选择启动注入通道组转换的外部事件 (External event select for injected group)          这些位选择用于启动注入通道组转换的外部事件。  <b>ADC1 的触发配置如下</b>          000: 定时器 1 的 TRGO 事件          001: 定时器 1 的 CC4 事件          100: 定时器 3 的 CC4 事件          101: 定时器 15 的 TRGO 事件          110: EXTI 线 15 事件          111: JSWSTR          其余: 无</p>
位 11	<p><b>DALIGN:</b> 数据对齐 (Data alignment)          该位由软件设置和清除。参考 <a href="#">图 14-6</a> 和 <a href="#">图 14-7</a>。          0: 右对齐；          1: 左对齐。</p>
位 10: 9	保留。必须保持为 0。
位 8	<p><b>DMAEN:</b> 直接存储器访问模式 (Direct memory access mode)          该位由软件设置和清除。详见 DMA 控制器章节。          0: 不使用 DMA 模式；          1: 使用 DMA 模式。</p>
位 7: 4	保留。必须保持为 0。
位 3	<p><b>RSTCAL:</b> 复位校准 (Reset calibration)          该位由软件设置并由硬件清除。在校准寄存器被初始化后该位将被清除。          0: 校准寄存器已初始化；          1: 初始化校准寄存器。  <b>注:</b> 如果正在进行转换时设置 RSTCAL，清除校准寄存器需要额外的周期。</p>
位 2	<p><b>CAL:</b> A/D 校准 (A/D Calibration)          该位由软件设置以开始校准，并在校准结束时由硬件清除。          0: 校准完成；          1: 开始校准。</p>
位 1	<p><b>CON:</b> 连续转换 (Continuous conversion)          该位由软件设置和清除。如果设置了此位，则转换将连续进行直到该位被清除。          0: 单次转换模式；          1: 连续转换模式。</p>

位 0	<b>ADON: 开/关 A/D 转换器 (A/D converter ON / OFF)</b> 该位由软件设置和清除。当该位为'0'时，写入'1'将把 ADC 从断电模式下唤醒。 当该位为'1'时，写入'1'将启动转换。应用程序需注意，在转换器上电至转换开始有一个延迟 $t_{STAB}$ ，参见图 14-2。 0: 关闭 ADC 转换/校准，并进入断电模式； 1: 开启 ADC 并启动转换。 <b>注：</b> 如果在这个寄存器中与 ADON 一起还有其他位被改变，则转换不被触发。这是为了防止触发错误的转换。														
-----	---	--	--	--	--	--	--	--	--	--	--	--	--	--	--

#### 14.4.4 ADC采样时间寄存器1 (ADC\_SMPT1)

地址偏移: 0x0C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15[0]	SMP14[2: 0]		SMP13[2: 0]		SMP12[2: 0]		SMP11[2: 0]		SMP10[2: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 24	保留。必须保持为 0。														
位 23: 0	<b>SMPx[2: 0]: 选择通道 x 的采样时间 (Channel x Sample time selection)</b> 这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。														
	000: 1.5 周期      100: 41.5 周期 001: 7.5 周期      101: 55.5 周期 010: 13.5 周期      110: 71.5 周期 011: 28.5 周期      111: 239.5 周期														

#### 14.4.5 ADC采样时间寄存器2 (ADC\_SMPT2)

地址偏移: 0x10

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留		SMP9[2: 0]		SMP8[2: 0]		SMP7[2: 0]		SMP6[2: 0]		SMP5[2: 1]					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP5[0]	SMP4[2: 0]		SMP3[2: 0]		SMP2[2: 0]		SMP1[2: 0]		SMPO[2: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 31: 30	保留。必须保持为 0。														
位 29: 0	<b>SMPx[2: 0]: 选择通道 x 的采样时间 (Channel x Sample time selection)</b> 这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。														
	000: 1.5 周期      100: 41.5 周期 001: 7.5 周期      101: 55.5 周期 010: 13.5 周期      110: 71.5 周期 011: 28.5 周期      111: 239.5 周期														

#### 14.4.6 ADC注入通道数据偏移寄存器x (ADC\_JOFSx) (x=1..4)

地址偏移: 0x14-0x20

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		JOFSTx[11: 0]													
res		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 12		保留。必须保持为 0。													
位 11: 0		JOFSTx[11: 0]: 注入通道 x 的数据偏移 (Data offset for injected channel x) 当转换注入通道时, 这些位定义了用于从原始转换数据中减去的数值。转换的结果可以在 ADC_JDORx 寄存器中读出。													

#### 14.4.7 ADC看门狗高阈值寄存器 (ADC\_WHTR)

地址偏移: 0x24

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		AWHT[11: 0]													
res		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 12		保留。必须保持为 0。													
位 11: 0		AWHT[11: 0]: 模拟看门狗高阈值 (Analog watchdog high threshold) 这些位定义了模拟看门狗的阈值高限。													

#### 14.4.8 ADC看门狗低阈值寄存器 (ADC\_WLTR)

地址偏移: 0x28

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		AWLT[11: 0]													
res		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 12		保留。必须保持为 0。													
位 11: 0		AWLT[11: 0]: 模拟看门狗低阈值 (Analog watchdog low threshold) 这些位定义了模拟看门狗的阈值低限。													

### 14.4.9 ADC规则序列寄存器1 (ADC\_RSQ1)

地址偏移: 0x2C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								LEN[3: 0]				SQ16[4: 1]			
res								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ16[0]	SQ15[4: 0]				SQ14[4: 0]				SQ13[4: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 24		保留。必须保持为 0。													
位 23: 20		<b>LEN[3: 0]:</b> 规则通道序列长度 (Regular channel sequence length) 这些位由软件定义在规则通道转换序列中的通道数目。 0000: 1 个转换 0001: 2 个转换 ..... 1111: 16 个转换													
位 19: 15		<b>SQ16[4: 0]:</b> 规则序列中的第 16 个转换 (16th conversion in regular sequence) 这些位由软件定义转换序列中的第 16 个转换通道的编号 (0~17)。													
位 14: 10		<b>SQ15[4: 0]:</b> 规则序列中的第 15 个转换 (15th conversion in regular sequence)													
位 9: 5		<b>SQ14[4: 0]:</b> 规则序列中的第 14 个转换 (14th conversion in regular sequence)													
位 4: 0		<b>SQ13[4: 0]:</b> 规则序列中的第 13 个转换 (13th conversion in regular sequence)													

### 14.4.10 ADC规则序列寄存器2 (ADC\_RSQ2)

地址偏移: 0x30

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留		SQ12[4: 0]				SQ11[4: 0]				SQ10[4: 1]					
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ10[0]	SQ9[4: 0]				SQ8[4: 0]				SQ7[4: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 30		保留。必须保持为 0。													
位 29: 25		<b>SQ12[4: 0]:</b> 规则序列中的第 12 个转换 (12th conversion in regular sequence) 这些位由软件定义转换序列中的第 12 个转换通道的编号 (0~17)。													
位 24: 20		<b>SQ11[4: 0]:</b> 规则序列中的第 11 个转换 (11th conversion in regular sequence)													
位 19: 15		<b>SQ10[4: 0]:</b> 规则序列中的第 10 个转换 (10th conversion in regular sequence)													
位 14: 10		<b>SQ9[4: 0]:</b> 规则序列中的第 9 个转换 (9th conversion in regular sequence)													
位 9: 5		<b>SQ8[4: 0]:</b> 规则序列中的第 8 个转换 (8th conversion in regular sequence)													
位 4: 0		<b>SQ7[4: 0]:</b> 规则序列中的第 7 个转换 (7th conversion in regular sequence)													

### 14.4.11 ADC规则序列寄存器3 (ADC\_RSQ3)

地址偏移: 0x34

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	SQ6[4: 0]				SQ5[4: 0]				SQ4[4: 1]						
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ4[0]	SQ3[4: 0]				SQ2[4: 0]				SQ1[4: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 30	保留。必须保持为 0。														
位 29: 25	<b>SQ6[4: 0]:</b> 规则序列中的第 6 个转换 (6th conversion in regular sequence) 这些位由软件定义转换序列中的第 6 个转换通道的编号 (0~9)。														
位 24: 20	<b>SQ5[4: 0]:</b> 规则序列中的第 5 个转换 (5th conversion in regular sequence)														
位 19: 15	<b>SQ4[4: 0]:</b> 规则序列中的第 4 个转换 (4th conversion in regular sequence)														
位 14: 10	<b>SQ3[4: 0]:</b> 规则序列中的第 3 个转换 (3rd conversion in regular sequence)														
位 9: 5	<b>SQ2[4: 0]:</b> 规则序列中的第 2 个转换 (2nd conversion in regular sequence)														
位 4: 0	<b>SQ1[4: 0]:</b> 规则序列中的第 1 个转换 (1st conversion in regular sequence)														

### 14.4.12 ADC注入序列寄存器 (ADC\_JSQ)

地址偏移: 0x38

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								JLEN[3: 0]		JSQ4[4: 0]					
								res		rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JSQ4[0]	JSQ3[4: 0]				JSQ2[4: 0]				JSQ1[4: 0]						
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 31: 22	保留。必须保持为 0。														
位 21: 20	<b>JLEN[1: 0]:</b> 注入通道序列长度 (Injected sequence length) 这些位由软件定义在注入通道转换序列中的通道数目。 00: 1 个转换 01: 2 个转换 10: 3 个转换 11: 4 个转换														
位 19: 15	<b>JSQ4[4: 0]:</b> 注入序列中的第 4 个转换 (4th conversion in injected sequence) 这些位由软件定义转换序列中的第 4 个转换通道的编号 (0~9)。 注: 不同于规则转换序列, 如果 JLEN[1: 0]的长度小于 4, 则转换的序列顺序是从 (4-JLEN) 开始。例如: ADC_JSQ[21: 0] = 10 00011 00011 00111 00010, 意味着扫描转换将按下列通道顺序转换: 7、3、3, 而不是 2、7、3。														

位 14: 10	<b>JSQ3[4: 0]:</b> 注入序列中的第 3 个转换 (3rd conversion in injected sequence)
位 9: 5	<b>JSQ2[4: 0]:</b> 注入序列中的第 2 个转换 (2nd conversion in injected sequence)
位 4: 0	<b>JSQ1[4: 0]:</b> 注入序列中的第 1 个转换 (1st conversion in injected sequence)

#### 14.4.13 ADC 注入数据寄存器x (ADC\_JDORx) (x=1..4)

地址偏移: 0x3C – 0x48

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JD[15: 0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
位 31: 16															
保留。必须保持为 0。															
位 21: 20															
<b>JD[15: 0]:</b> 注入转换的数据 (Injected data) 这些位为只读，包含了注入通道的转换结果。数据是左对齐或右对齐，如图 14-6 和图 14-7 所示。															

#### 14.4.14 ADC 规则数据寄存器 (ADC\_RDOR)

地址偏移: 0x4C

复位值: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D[15: 0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
位 31: 16															
保留															
位 15: 0															
<b>D[15: 0]:</b> 规则转换的数据 (Regular data) 这些位为只读，包含了规则通道的转换结果。数据是左对齐或右对齐，如图 14-6 和图 14-7 所示。															

# 15 I<sup>2</sup>C接口 (I<sup>2</sup>C)

## 15.1 I<sup>2</sup>C简介

I<sup>2</sup>C(芯片间)总线接口连接微控制器和串行 I<sup>2</sup>C 总线。它提供多主机功能，控制所有 I<sup>2</sup>C 总线特定的时序、协议、仲裁和定时。支持标准和快速两种模式，同时与 SMBus2.0 兼容。I<sup>2</sup>C 模块有多种用途，包括 CRC 码的生成和校验、SMBus(系统管理总线—System Management Bus)和 PMBus(电源管理总线—Power Management Bus)。根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

## 15.2 I<sup>2</sup>C主要特点

- 并行总线/I<sup>2</sup>C总线协议转换器
- 多主机功能：该模块既可做主设备也可做从设备
- I<sup>2</sup>C主设备功能
  - 产生时钟
  - 产生起始和停止信号
- I<sup>2</sup>C从设备功能
  - 可编程的I<sup>2</sup>C地址检测
  - 可响应2个从地址的双地址能力
  - 停止位检测
- 产生和检测7位/10位地址和广播呼叫
- 支持不同的通讯速度
  - 标准速度(高达100 kHz)
  - 快速(高达400 kHz)
- 状态标志：
  - 发送器/接收器模式标志
  - 字节发送结束标志
  - I<sup>2</sup>C总线忙标志
- 错误标志
  - 主模式时的仲裁丢失
  - 地址/数据传输后的应答(ACK)错误
  - 检测到错位的起始或停止条件
  - 上溢或下溢(在禁止时钟延长的情况下)
- 2个中断向量
  - 1个中断用于地址/数据通讯成功
  - 1个中断用于错误
- 可选的禁止时钟延长功能
- 具单字节缓冲器的DMA
- 可配置的PEC(信息包错误检测)的产生或校验
  - 发送模式中PEC值可以作为最后一个字节传输
  - 用于最后一个接收字节的PEC错误校验
- 兼容SMBus 2.0
  - 25 ms时钟低超时延时
  - 10 ms主设备累积时钟低扩展时间
  - 25 ms从设备累积时钟低扩展时间

- 带ACK控制的硬件PEC产生/校验
- 支持地址分辨协议(ARP)
- 兼容PMBus

## 15.3 I<sup>2</sup>C功能描述

I<sup>2</sup>C 模块接收和发送数据，并将数据从串行转换成并行，或并行转换成串行。可以开启或禁止中断。接口通过数据引脚(SDA)和时钟引脚(SCL)连接到 I<sup>2</sup>C 总线。允许连接到标准(高达 100kHz)或快速(高达 400kHz)的 I<sup>2</sup>C 总线。

### 15.3.1 模式选择

接口可以运行下述模式中的一种：

- 从发送器模式
- 从接收器模式
- 主发送器模式
- 主接收器模式

在默认状态下，接口工作于从模式。接口在生成起始条件后自动地从从模式切换到主模式；当仲裁丢失或产生停止信号时，则从主模式切换到从模式。允许多主机功能。

#### 通信流

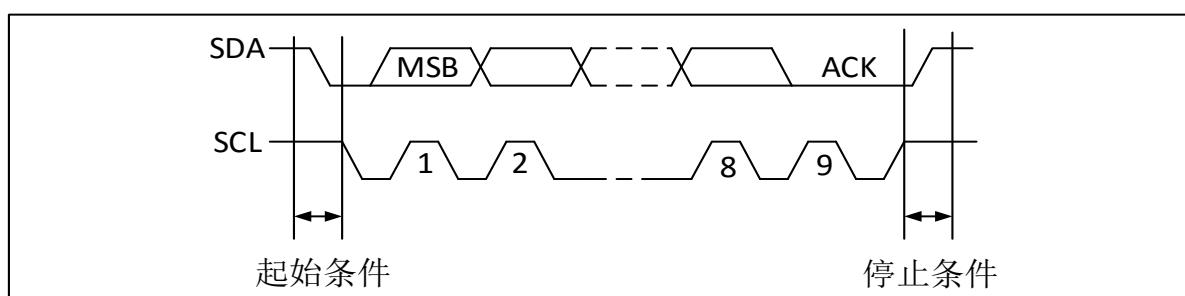
主模式时，I<sup>2</sup>C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。起始条件和停止条件都是在主模式下由软件控制产生。

从模式时，I<sup>2</sup>C 接口能识别它自己的地址(7位或 10位)和广播呼叫地址。软件能够开启或禁止广播呼叫地址的识别功能。

数据和地址按 8 位(也即一个字节)进行传输，高位在前。跟在起始条件后的 1 或 2 个字节是地址(7位模式为 1 个字节，10位模式为 2 个字节)。地址只在主模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位(ACK)给发送器。参考下图。

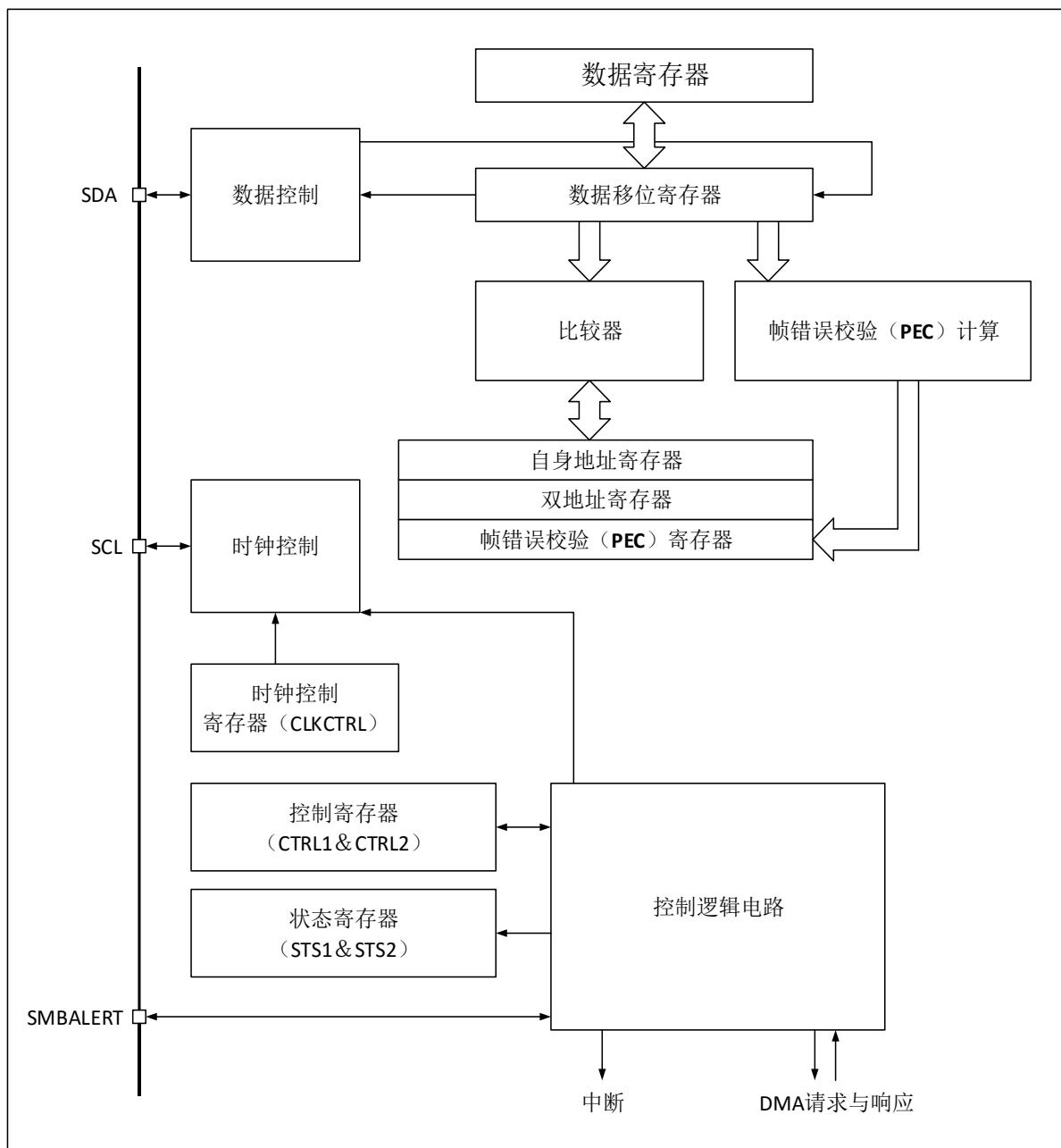
图表 15-1 I<sup>2</sup>C总线协议



软件可以开启或禁止应答(ACKEN)，并可以设置 I<sup>2</sup>C 接口的地址(7位、10位地址或广播呼叫地址)。

I<sup>2</sup>C 接口的功能框图示于下图。

图表 15-2 I<sup>2</sup>C的功能框图



注意：在 SMBus 模式下，SMBALERT 是可选信号。如果禁止了 SMBus，则不能使用该信号。

### 15.3.2 I<sup>2</sup>C从模式

默认情况下，I<sup>2</sup>C 接口总是工作在从模式。从从模式切换到主模式，需要产生一个起始条件。为了产生正确的时序，必须在 I<sup>2</sup>C\_CTRL2 寄存器中设定该模块的输入时钟。输入时钟的频率必须至少是：

- 标准模式下为：2MHz
- 快速模式下为：4MHz

一旦检测到起始条件，在 SDA 线上接收到的地址被送到移位寄存器。然后与芯片自己的地址 OADDR1 和 OADDR2(当 DUALEN=1)或者广播呼叫地址(如果 GCEN=1)相比较。

注意：在 10 位地址模式时，比较包括头段序列(11110xx0)，其中的 xx 是地址的两个最高有效位。

**头段或地址不匹配：** I<sup>2</sup>C 接口将其忽略并等待另一个起始条件。

**头段匹配(仅 10 位模式)：** 如果 ACKEN 位被置'1'，I<sup>2</sup>C 接口产生一个应答脉冲并等待 8 位从地址。

**地址匹配:** I<sup>2</sup>C 接口产生以下时序:

- 如果 ACKEN 被置'1', 则产生一个应答脉冲
- 硬件设置 ADDRF 位; 如果设置了 EVTITEN 位, 则产生一个中断
- 如果 DUALEN=1, 软件必须读 DUALF 位, 以确认响应了哪个从地址

在 10 位模式, 接收到地址序列后, 从设备总是处于接收器模式。当接收到重复的起始条件后, 在收到与地址匹配的头序列并且最低位为'1'(即 11110xx1)时, 将进入发送器模式。

从模式下 TRF 位指示当前是处于接收器模式还是发送器模式。

### 从发送器

在接收到地址和清除 ADDRF 位后, 从发送器将字节从 DT 寄存器经由内部移位寄存器发送到 SDA 线上。

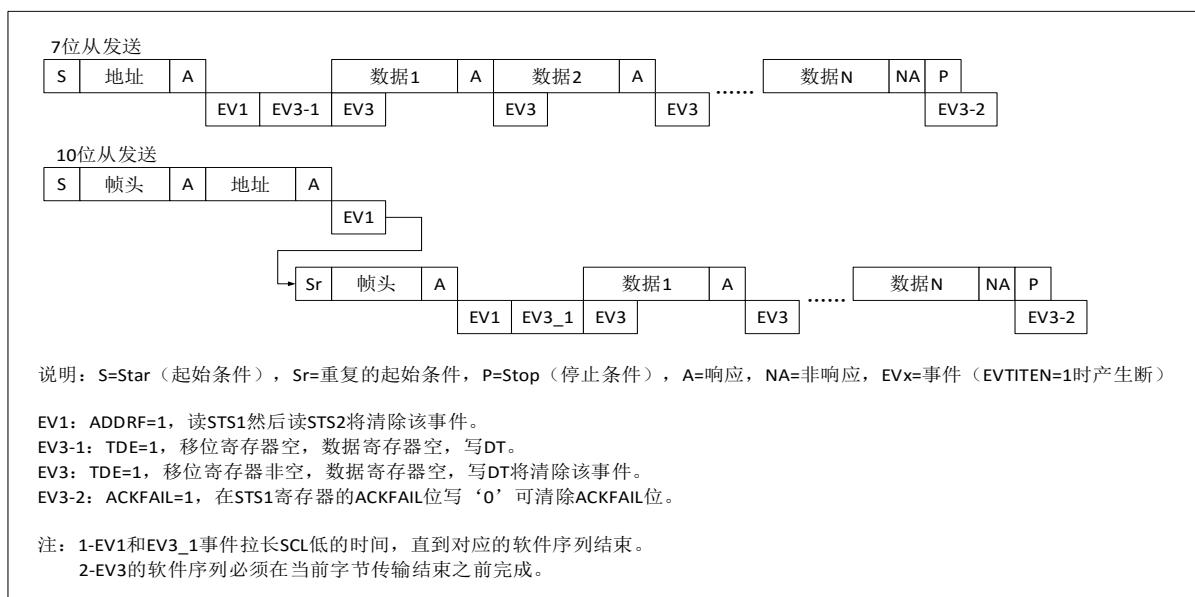
从设备保持 SCL 为低电平, 直到 ADDRF 位被清除并且待发送数据已写入 DT 寄存器。(见图 15-3 中的 EV1 和 EV3)。

当收到应答脉冲时:

- TDE 位被硬件置位, 如果设置了 EVTITEN 和 BUFITEN 位, 则产生一个中断

如果 TDE 位被置位, 但在下一个数据发送结束之前没有新数据写入到 I2C\_DT 寄存器, 则 BTFF 位被置位, 在清除 BTFF 之前 I<sup>2</sup>C 接口将保持 SCL 为低电平; 读出 I2C\_STS1 之后再写入 I2C\_DT 寄存器将清除 BTFF 位。

图表 15-3 从发送器的传送序列图



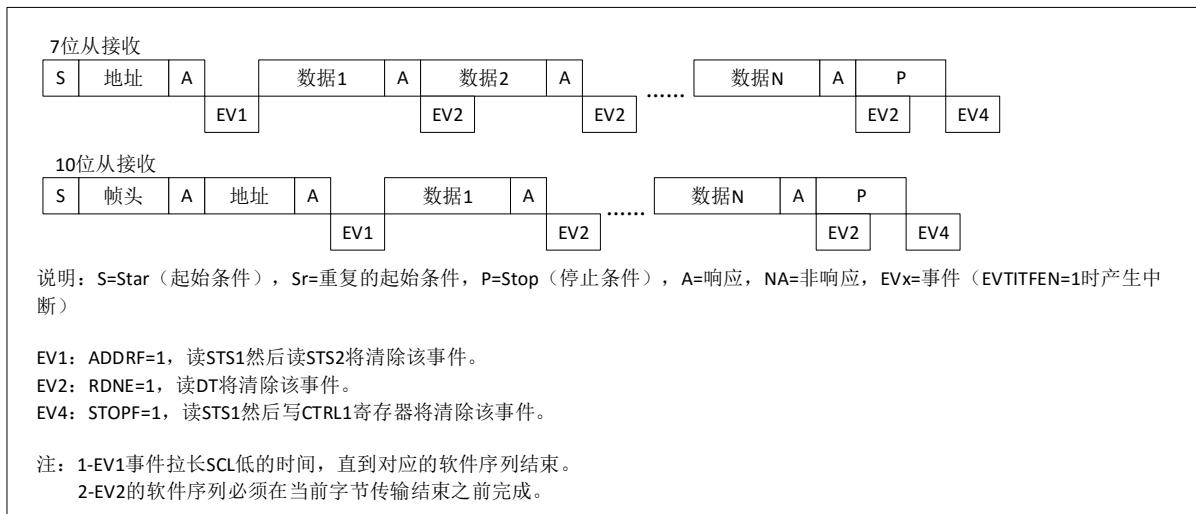
### 从接收器

在接收到地址并清除 ADDRF 后, 从接收器将通过内部移位寄存器从 SDA 线接收到的字节存进 DT 寄存器。I<sup>2</sup>C 接口在接收到每个字节后都执行下列操作:

- 如果设置了 ACKEN 位, 则产生一个应答脉冲
- 硬件设置 RDNE=1 如果设置了 EVTITEN 和 BUFITEN 位, 则产生一个中断。

如果 RDNE 被置位, 并且在接收新的数据结束之前 DT 寄存器未被读出, BTFF 位被置位, 在清除 BTFF 之前 I<sup>2</sup>C 接口将保持 SCL 为低电平; 读出 I2C\_STS1 之后再读出 I2C\_DT 寄存器将清除 BTFF 位。

图表 15-4 从接收器的传送序列图



### 关闭从通信

在传输完最后一个数据字节后，主设备产生一个停止条件，I<sup>2</sup>C 接口检测到这一条件时：

- 设置STOPF=1，如果设置了EVTITEN位，则产生一个中断

然后 I<sup>2</sup>C 接口等待读 STS1 寄存器，再写 CTRL1 寄存器，完成这个操作后 STOPF 位被清零。（见[图 15-4](#) 中的 EV4）。

### 15.3.3 I<sup>2</sup>C 主模式

在主模式时，I<sup>2</sup>C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。当通过 STARTGEN 位在总线上产生了起始条件，设备就进入了主模式。

以下是主模式所要求的操作顺序：

- 在I2C\_CTRL2寄存器中设定该模块的输入时钟以产生正确的时序
- 配置时钟控制寄存器
- 配置上升时间寄存器
- 编程I2C\_CTRL1寄存器启动外设
- 置I2C\_CTRL1寄存器中的STARTGEN位为1，产生起始条件

I<sup>2</sup>C 模块的输入时钟频率必须至少是：

- 标准模式下为：2MHz
- 快速模式下为：4MHz

### 主机时钟的产生

使用 CLKCTRL 来产生 I<sup>2</sup>C 时钟的高低电平，它从时钟的上升沿或者下降沿开始。由于从机可能会拉低时钟线，因此外设此后等待一个 TMRISE 寄存器设置的最大上升时间后去检测时钟线的电平。

- 如果时钟线为低，说明从机拉低了总线，并且计数高电平的计数器停止计数直到检测到时钟线变高。这样可以保证时钟的高电平宽度
- 如果时钟线为高，高电平计数器持续计数

### 起始条件

当 BUSYF=0 时，设置 STARTGEN=1，I<sup>2</sup>C 接口将产生一个开始条件并切换至主模式(MSF 位置位)。

注意：在主模式下，设置 STARTGEN 位将在当前字节传输完后由硬件产生一个重开始条件。

一旦发出开始条件：

- STARTF位被硬件置位，如果设置了EVTITEN位，则会产生一个中断

然后主设备等待读 STS1 寄存器，紧跟着将从地址写入 DT 寄存器(见[图 15-5](#) 和[图 15-6](#) 的 EV5)。

### 从地址的发送

从地址通过内部移位寄存器被送到 SDA 线上。

- 在10位地址模式时，发送一个头段序列产生以下事件

- ADDR10F位被硬件置位，如果设置了EVTITEN位，则产生一个中断。

然后主设备等待读STS1寄存器，再将第二个地址字节写入DT寄存器(见[图15-5](#)和[图15-6](#))

- ADDRF位被硬件置位，如果设置了EVTITEN位，则产生一个中断。

随后主设备等待一次读STS1寄存器，跟着读STS2寄存器(见[图15-5](#)和[图15-6](#))

- 在7位地址模式时，只需送出一个地址字节

一旦该地址字节被送出，

- ADDRF位被硬件置位，如果设置了EVTITEN位，则产生一个中断

随后主设备等待一次读STS1寄存器，跟着读STS2寄存器(见[图15-5](#)和[图15-6](#))。

根据送出从地址的最低位，主设备决定进入发送器模式还是进入接收器模式。

- 在7位地址模式时

— 要进入发送器模式，主设备发送从地址时置最低位为'0'

— 要进入接收器模式，主设备发送从地址时置最低位为'1'

- 在10位地址模式时

— 要进入发送器模式，主设备先送头字节(11110xx0)，然后送从地址。(这里xx代表10位地址中的最高2位。)

— 要进入接收器模式，主设备先送头字节(11110xx0)，然后送从地址。然后再重新发送一个开始条件，后面跟着头字节(11110xx1)(这里xx代表10位地址中的最高2位。)

TRF位指示主设备是在接收器模式还是发送器模式。

### 主发送器

在发送了地址和清除了ADDRF位后，主设备通过内部移位寄存器将字节从DT寄存器发送到SDA线上。

主设备等待，直到数据写入DT寄存器，TDE被清除，(见[图15-5](#)中的EV8)。

当收到应答脉冲时：

- TDE位被硬件置位，如果设置了EVTITEN和BUFITEN位，则产生一个中断

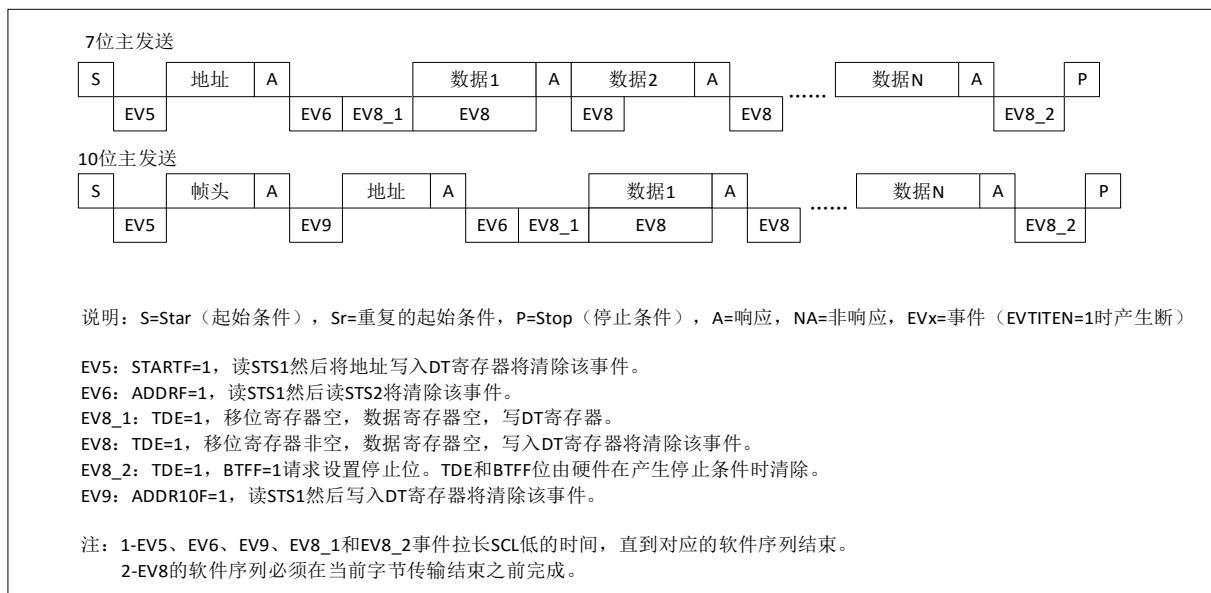
如果TDE被置位并且在上一次数据发送结束之前没有写新的数据字节到DT寄存器，则BTFF被硬件置位，在清除BTFF之前I<sup>2</sup>C接口将保持SCL为低电平；读出I<sup>2</sup>C\_STS1之后再写入I<sup>2</sup>C\_DT寄存器将清除BTFF位。

### 关闭通信

在DT寄存器中写入最后一个字节后，通过设置STOPGEN位产生一个停止条件(见[图15-5](#)的EV8\_2)，然后I<sup>2</sup>C接口将自动回到从模式(MSF位清除)。

注意：当TDE或BTFF位置位时，停止条件应安排在出现EV8\_2事件时。

图表 15-5 主发送器传送序列图



## 主接收器

在发送地址和清除 ADDRDF 之后, I<sup>2</sup>C 接口进入主接收器模式。在此模式下, I<sup>2</sup>C 接口从 SDA 线接收数据字节, 并通过内部移位寄存器送至 DT 寄存器。在每个字节后, I<sup>2</sup>C 接口依次执行以下操作:

- 如果 ACKEN 位被置位, 发出一个应答脉冲
- 硬件设置 RDNE=1, 如果设置了 EVTITEN 和 BUFITEN 位, 则会产生一个中断(见 [图 15-6](#) 中的 EV7)

如果 RDNE 位被置位, 并且在接收新数据结束前, DT 寄存器中的数据没有被读走, 硬件将设置 BTFF=1, 在清除 BTFF 之前 I<sup>2</sup>C 接口将保持 SCL 为低电平; 读出 I<sup>2</sup>C\_STS1 之后再读出 I<sup>2</sup>C\_DT 寄存器将清除 BTFF 位。

## 关闭通信

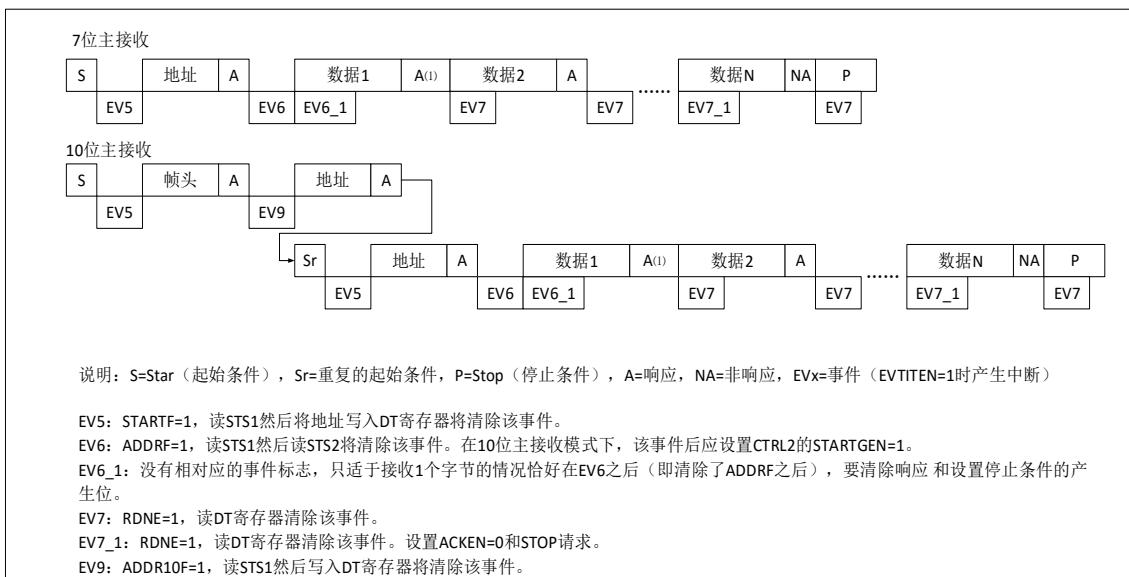
### 第一种情况: I<sup>2</sup>C 中断设为最高优先级

主设备在从设备接收到最后一个字节后发送一个 NACK。接收到 NACK 后, 从设备释放对 SCL 和 SDA 线的控制; 主设备就可以发送一个停止/重起始条件。

- 为了在收到最后一个字节后产生一个 NACK 脉冲, 在读倒数第二个数据字节之后(在倒数第二个 RDNE 事件之后)必须清除 ACKEN 位
- 为了产生一个停止/重起始条件, 软件必须在读倒数第二个数据字节之后(在倒数第二个 RDNE 事件之后)设置 STOPGEN/STARTGEN 位
- 只接收一个字节时, 刚好在 EV6 之后(EV6\_1 时, 清除 ADDRDF 之后)要关闭应答和停止条件的产生位

在产生了停止条件后, I<sup>2</sup>C 接口自动回到从模式(MSF 位被清除)。

图表 15-6 主接收器传送序列图



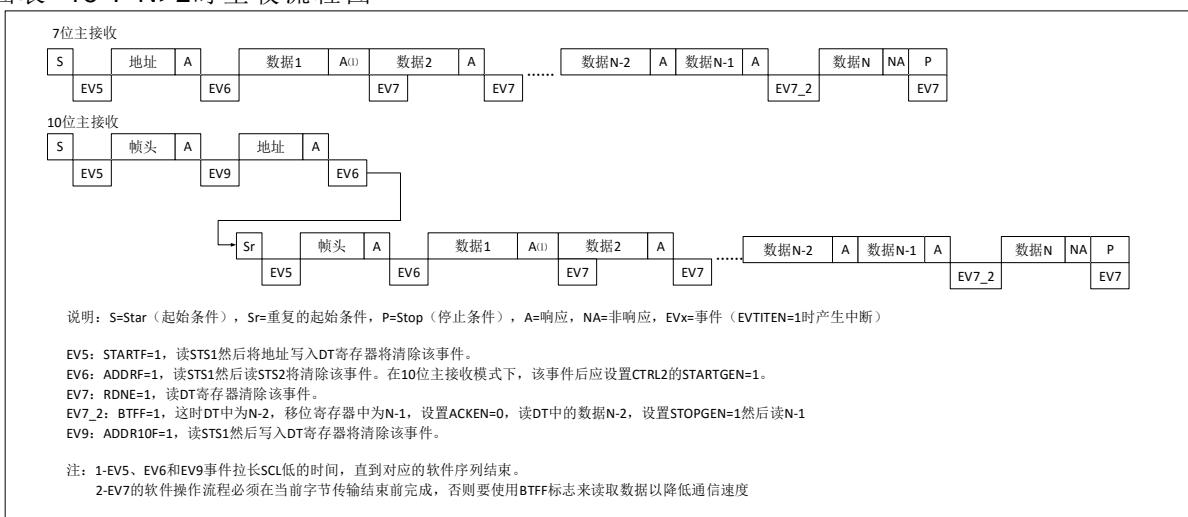
注意:

1. 如果收到一个单独的字节, 则是 NA。
2. EV5、EV6 和 EV9 事件拉长 SCL 低电平, 直到对应的软件序列结束。
3. EV7 的软件序列必须在当前字节传输结束前完成。
4. EV6\_1 或 EV7\_1 的软件序列必须在当前传输字节的 ACK 脉冲之前完成。

第二种情况: I<sup>2</sup>C 中断未设为最高优先级并且接收字节总数 N>2

在这种条件下, 接收到数据 N-2 时并不读取, 当 N-1 也收到后, 时钟被拉低, 通信暂停。此时清 ACKEN 位, 然后依次读数据 N-2, 设置 STOPGEN/STARTGEN, 读数据 N-1, 再待 RDNE 置位后, 读数据 N, 如下图所示:

图表 15-7 N>2 时主收流程图



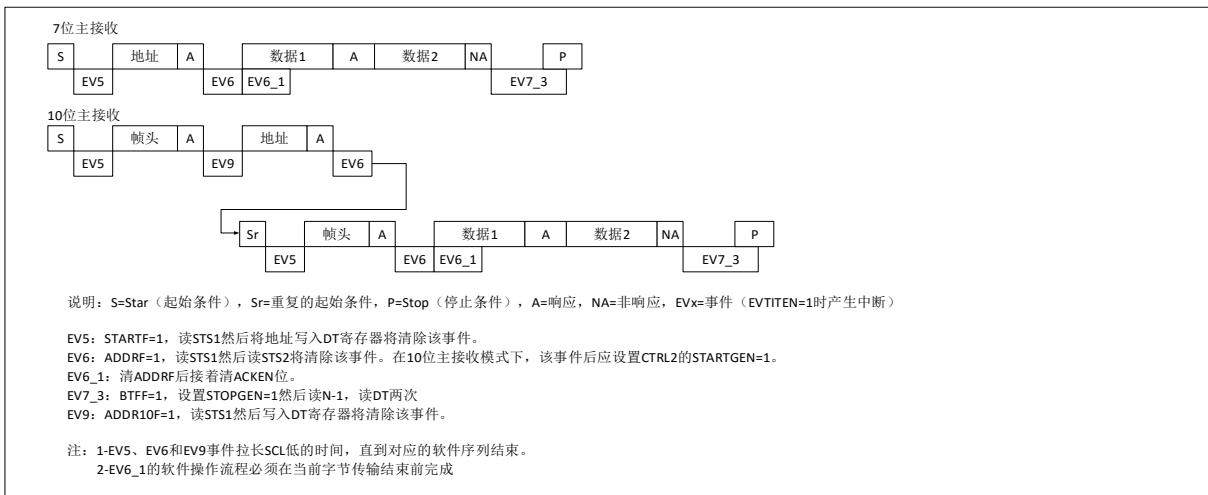
当剩下三个字节时软件流程如下:

- RDNE=1 => 不做任何操作
- 收到数据 N-1
- BTFF=1, DT 中 N-2, 移位寄存器中 N-1, 时钟被拉低
- 清 ACKEN 位
- 读数据 N-2, 然后总线会开始接收数据 N
- 收到数据 N, 并回复 NACK
- 设置 STARTGEN/STOPGEN 位
- RDNE=1
- 读数据 N

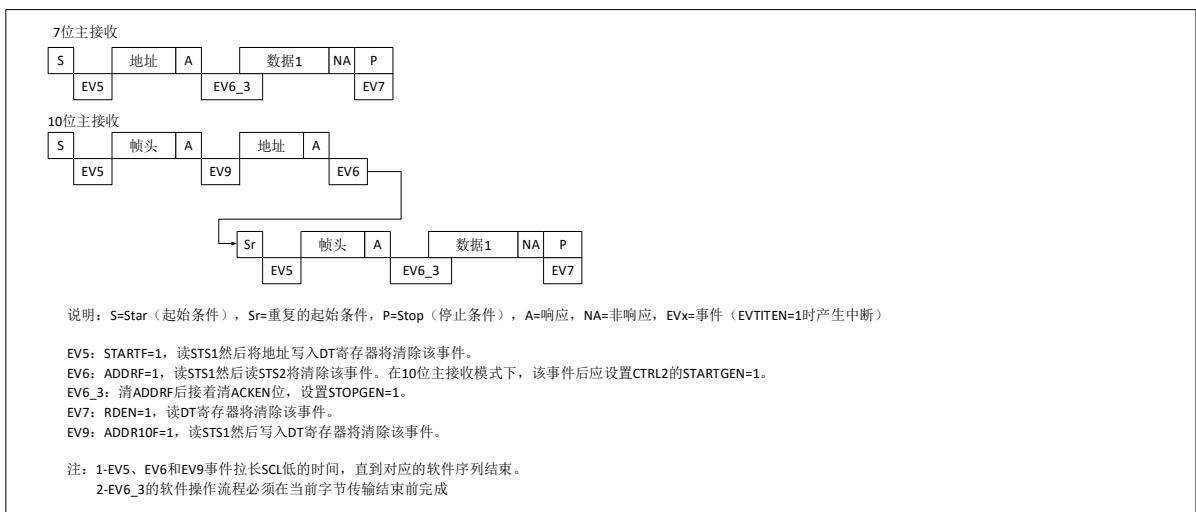
**第三种情况：I<sup>2</sup>C 中断未设为最高优先级并且接收字节总数 N=2 或 N=1**

- 接收一个字节
  - ADDR事件时清ACKEN位
  - 清ADDR位
  - 设置STARTGEN/STOPGEN位
  - RDNE置位时读取数据
- 接收两个字节
  - 设置POSEN和ACKEN位
  - 等待ADDRF置位
  - 清ADDRF位
  - 清ACKEN位
  - 等待BTFF置位
  - 置STOPGEN位
  - 读两次DT寄存器

图表 15-8 N=2时主收流程图



图表 15-9 N=1时主收流程图



### 15.3.4 错误条件

以下条件可能造成通讯失败。

#### 总线错误(BUSERR)

在一个地址或数据字节传输期间，当 I<sup>2</sup>C 接口检测到一个外部的停止或起始条件则产生总线错误。此时：

- BUSERR位被置位为'1'；如果设置了ERRITEN位，则产生一个中断
- 在从模式情况下，数据被丢弃，硬件释放总线
  - 如果是错误的开始条件，从设备必须清BUSERR位，然后等待下一次开始条件
  - 如果是错误的停止条件，从设备按正常的停止条件操作，同时硬件释放总线
- 在主模式情况下，硬件不释放总线，同时不影响当前的传输状态。此时由软件决定是否要中止当前的传输

#### 应答错误(ACKFAIL)

当接口检测到一个无应答位时，产生应答错误。此时：

- ACKFAIL位被置位，如果设置了ERRITEN位，则产生一个中断
- 当发送器接收到一个NACK时，必须复位通讯
  - 如果是处于从模式，硬件释放总线
  - 如果是处于主模式，软件必须生成一个停止条件或者重复的起始条件

#### 仲裁丢失(ARLOST)

当 I<sup>2</sup>C 接口检测到仲裁丢失时产生仲裁丢失错误，此时：

- ARLOST位被硬件置位，如果设置了ERRITEN位，则产生一个中断
- I<sup>2</sup>C 接口自动回到从模式(MSF位被清除)。当I<sup>2</sup>C接口丢失了仲裁，则它无法在同一个传输中响应它的从地址，但它可以在赢得总线的主设备发送重起始条件之后响应
- 硬件释放总线

#### 过载/欠载错误(OVRUN)

在从模式下，如果禁止时钟延长，I<sup>2</sup>C 接口正在接收数据时，当它已经接收到一个字节(RDNE=1)，但在 DT 寄存器中前一个字节数据还没有被读出，则发生过载错误。此时：

- 最后接收的数据被丢弃
- 在过载错误时，软件应清除RDNE位，发送器应该重新发送最后一次发送的字节

在从模式下，如果禁止时钟延长，I<sup>2</sup>C 接口正在发送数据时，在下一个字节的时钟到达之前，新的数据还未写入 DT 寄存器(TDE=1)，则发生欠载错误。此时：

- 在DT寄存器中的前一个字节将被重复发出
- 用户应该确定在发生欠载错时，接收端应丢弃重复接收到的数据。发送端应按I<sup>2</sup>C总线标准在规定的时间更新DT寄存器

在发送第一个字节时，必须在清除ADDRF之后并且第一个SCL上升沿之前写入DT寄存器；如果不能做到这点，则接收方应该丢弃第一个数据。

### 15.3.5 SDA/SCL线控制

- 如果允许时钟延长：
  - 发送器模式：如果TDE=1且BTFF=1：I<sup>2</sup>C接口在传输前保持时钟线为低，以等待软件读取STS1，然后把数据写进数据寄存器(缓冲器和移位寄存器都是空的)
  - 接收器模式：如果RDNE=1且BTFF=1：I<sup>2</sup>C接口在接收到数据字节后保持时钟线为低，以等待软件读STS1，然后读数据寄存器DT(缓冲器和移位寄存器都是满的)
- 如果在从模式中禁止时钟延长
  - 如果RDNE=1，在接收到下个字节前DT还没有被读出，则发生过载错。接收到的最后一个字节丢失

- 如果TDE=1，在必须发送下个字节之前却没有新数据写进DT，则发生欠载错。相同的字节将被重复发出
- 不控制重复写冲突

## 15.3.6 SMBus

### 介绍

系统管理总线(SMBus)是一个双线接口。通过它，各设备之间以及设备与系统的其他部分之间可以互相通信。它基于I<sup>2</sup>C操作原理。SMBus为系统和电源管理相关的任务提供一条控制总线。一个系统利用SMBus可以和多个设备互传信息，而不需使用独立的控制线路。

系统管理总线(SMBus)标准涉及三类设备。从设备：接收或响应命令的设备。主设备：用来发送命令、产生时钟和终止发送的设备。主机：一种专用的主设备，它提供与系统CPU的主接口。主机必须具有主-从机功能并且必须支持SMBus提醒协议。一个系统里只允许有一个主机。

### SMBus 和 I<sup>2</sup>C 之间的相似点

- 2条线的总线协议(1个时钟，1个数据) + 可选的SMBus提醒线
- 主-从通信，主设备提供时钟
- 多主机功能
- SMBus数据格式类似于I<sup>2</sup>C的7位地址格式(见图15-1)

### SMBus 和 I<sup>2</sup>C 之间的不同点

下表列出了SMBus和I<sup>2</sup>C的不同点。

表格 15-1 SMBus与I<sup>2</sup>C的比较

SMBus	I <sup>2</sup> C
最大传输速度 100kHz	最大传输速度 400kHz
最小传输速度 10kHz	无最小传输速度
35ms 时钟低超时	无时钟超时
固定的逻辑电平	逻辑电平由VDD决定
不同的地址类型(保留的、动态的等)	7位、10位和广播呼叫从地址类型
不同的总线协议(快速命令、处理呼叫等)	无总线协议

### SMBus 应用用途

利用系统管理总线，设备可提供制造商信息，告诉系统它的型号/部件号，保存暂停事件的状态，报告不同类型的错误，接收控制参数，和返回它的状态。SMBus为系统和电源管理相关的任务提供控制总线。

### 设备标识

在系统管理总线上，任何一个作为从模式的设备都有一个唯一的地址，叫做从地址。保留的从地址表请参考2.0版的SMBus规范(<http://smbus.org/specs/>)。

### 总线协议

SMBus技术规范支持9个总线协议。有关这些协议的详细资料和SMBus地址类型，请参考2.0版的SMBus规范(<http://smbus.org/specs/>)。这些协议由用户的软件来执行。

### 地址解析协议(ARP)

通过给每个从设备动态地分配一个新的唯一地址，可以解决SMBus的从地址冲突。地址解析协议(ARP)具有以下的特性：

- 使用标准SMBus物理层仲裁机制分配地址
- 当设备维持供电期间，分配的地址仍保持不变，也允许设备在断电后保留其地址
- 在地址分配后，没有额外的SMBus的打包开销(也就是说访问分配地址的设备与访问

- 固定地址的设备所用时间是一样的)
- 任何一个SMBus主设备可以遍历总线

### 唯一的设备标识符(UDID)

为了分配地址，需要一种区分每个设备的机制，每个设备必须拥有一个唯一的设备标识符。

关于在 ARP 上 128 位的 UDID 的详细信息，参考 2.0 版的 SMBus 规范(<http://smbus.org/specs/>)。

### SMBus 提醒模式

SMBus 提醒是一个带中断线的可选信号，用于那些希望扩展它们的控制能力而牺牲一个引脚的设备。SMBALERT 和 SCL、SDA 信号一样，是一种线与信号。SMBALERT 通常和 SMBus 广播呼叫地址一起使用。与 SMBus 有关的消息为 2 字节。

一个只具有从功能的设备，可以通过设置 I2C\_CTRL1 寄存器上的 SMBALERT 位，使用 SMBALERT 给主机发信号表示它希望进行通信。主机处理该中断并通过提醒响应地址 ARA(Alert Response Address，地址值为 0001100x)访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应答 ARA。此状态是由 I2C\_STS1 寄存器中的 SMBALERTF 状态标记来标识的。主机执行一个修改过的接收字节操作。由从发送设备提供的 7 位设备地址被放在字节的 7 个最高位上，第八个位可以是'0'或'1'。

如果多个设备把 SMBALERT 拉低，最高优先级设备(最小的地址)将在地址传输期间通过标准仲裁赢得通信权。在确认从地址后，此设备不得再拉低它的 SMBALERT，如果当信息传输完成后，主机仍看到 SMBALERT 低，就知道需要再次读 ARA。

没有实现 SMBALERT 信号的主机可以定期访问 ARA。

有关 SMBus 提醒模式的更多详细资料，请参考 2.0 版的 SMBus 规范(<http://smbus.org/specs/>)。

### 超时错误

在定时规范上 I<sup>2</sup>C 和 SMBus 之间有很多差别。

SMBus 定义了一个时钟低超时，35ms 的超时。SMBus 规定 TLOW: SEXT 为从设备的累积时钟低扩展时间。SMBus 规定 TLOW: MEXT 为主设备的累积时钟低扩展时间。更多超时细节请参考 2.0 版的 SMBus 规范(<http://smbus.org/specs/>)。

I2C\_STS1 中的状态标志 Timeout 或 Tlow 错误表明了这个特性的状态。

### 如何使用 SMBus 模式的接口

为了从 I<sup>2</sup>C 模式切换到 SMBus 模式，应该执行下列步骤：

- 设置 I2C\_CTRL1 寄存器中的 SMBMODE 位
- 按应用要求配置 I2C\_CTRL1 寄存器中的 SMBTYPE 和 ARPEN 位

如果要把设备配置成主设备，产生起始条件的步骤见 [15.3.3 节 I<sup>2</sup>C 主模式](#)。否则，参见 [15.3.2 节 I<sup>2</sup>C 从模式](#)。

软件程序必须处理多种 SMBus 协议。

- 如果 ARPEN=1 且 SMBTYPE=0，使用 SMB 设备默认地址
- 如果 ARPEN=1 且 SMBTYPE=1，使用 SMB 主设备头字段
- 如果 SMBALERTF=1，使用 SMB 提醒响应地址

## 15.3.7 DMA请求

DMA 请求(当被使能时)仅用于数据传输。发送时数据寄存器变空或接收时数据寄存器变满，则产生 DMA 请求。DMA 请求必须在当前字节传输结束之前被响应。当为相应 DMA 通道设置的数据传输量已经完成时，DMA 控制器发送传输结束信号 EOT 到 I<sup>2</sup>C 接口，并且在中断允许时产生一个传输完成中断：

- 主发送器：在 EOT 中断服务程序中，需禁止 DMA 请求，然后在等到 BTFF 事件后设置停止条件
- 主接收器：当要接收的数据数目大于或等于 2 时，DMA 控制器发送一个硬件信号 EOT\_1，它对应 DMA 传输(字节数 - 1)。如果在 I2C\_CTRL2 寄存器中设置了 DMALAST 位，硬件在发送完 EOT\_1 后的下一个字节，将自动发送 NACK。在中断允许的情况下，用户可以在 DMA 传输完成的中断服务程序中产生一个停止条件

### 利用 DMA 发送

通过设置 I2C\_CTRL2 寄存器中的 DMAEN 位可以激活 DMA 模式。只要 TDE 位被置位，数据将由 DMA 从预置的存储区装载进 I2C\_DT 寄存器。为 I<sup>2</sup>C 分配一个 DMA 通道，须执行以下步骤(x 是通道号)：

1. 在 DMA\_CPBAX 寄存器中设置 I2C\_DT 寄存器地址。数据将在每个 TDE 事件后从存储器传送至这个地址。
2. 在 DMA\_CMBAx 寄存器中设置存储器地址。数据在每个 TDE 事件后从这个存储区传送至 I2C\_DT。
3. 在 DMA\_TCNTx 寄存器中设置所需的传输字节数。在每个 TDE 事件后，此值将被递减。
4. 利用 DMA\_CHCTRLx 寄存器中的 CHPL[0: 1]位配置通道优先级。
5. 设置 DMA\_CHCTRLx 寄存器中的 DIR 位，并根据应用要求可以配置在整个传输完成一半或全部完成时发出中断请求。
6. 通过设置 DMA\_CHCTRLx 寄存器上的 CHEN 位激活通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I<sup>2</sup>C 接口发送一个传输结束的 EOT / EOT\_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注意：如果使用 DMA 进行发送时，不要设置 I2C\_CTRL2 寄存器的 BUFITEN 位。

### 利用 DMA 接收

通过设置 I2C\_CTRL2 寄存器中的 DMAEN 位可以激活 DMA 接收模式。每次接收到数据字节时，将由 DMA 把 I2C\_DT 寄存器的数据传送到设置的存储区(参考 DMA 说明)。设置 DMA 通道进行 I<sup>2</sup>C 接收，须执行以下步骤(x 是通道号)：

1. 在 DMA\_CPBAX 寄存器中设置 I2C\_DT 寄存器的地址。数据将在每次 RDNE 事件后从此地址传送到存储区。
2. 在 DMA\_CMBAx 寄存器中设置存储区地址。数据将在每次 RDNE 事件后从 I2C\_DT 寄存器传送到此存储区。
3. 在 DMA\_TCNTx 寄存器中设置所需的传输字节数。在每个 RDNE 事件后，此值将被递减。
4. 利用 DMA\_CHCTRLx 寄存器中的 CHPL[0: 1]配置通道优先级。
5. 清除 DMA\_CHCTRLx 寄存器中的 DIR 位，根据应用要求可以设置在数据传输完成一半或全部完成时发出中断请求。
6. 设置 DMA\_CHCTRLx 寄存器中的 CHEN 位激活该通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I<sup>2</sup>C 接口发送一个传输结束的 EOT / EOT\_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注意：如果使用 DMA 进行接收时，不要设置 I2C\_CTRL2 寄存器的 BUFITEN 位。

### 15.3.8 包错误校验(PEC)

包错误校验(PEC)计算器是用于提高通信的可靠性，这个计算器使用下述 CRC-8 多项式对每一位串行数据进行计算：

$$C(x) = x^8 + x^2 + x + 1$$

- PEC 计算由 I2C\_CTRL1 寄存器的 PECEN 位激活。PEC 使用 CRC-8 算法对所有信息字节进行计算，包括地址和读/写位在内
  - 在发送时：在最后一个 TDE 事件后设置 I2C\_CTRL1 寄存器的 PECTRA 传输位，PEC 将在最后一个字节后被发送
  - 在接收时：在最后一个 RDNE 事件之后设置 I2C\_CTRL1 寄存器的 PECTRA 位，如果下一个接收到的字节不等于内部计算的 PECVAL，接收器发送一个 NACK。如果是主接收器，不管校对的结果如何，PEC 后都将发送 NACK。PECTRA 位必须在接收当前字节的 ACK 脉冲之前设置
- 在 I2C\_STS1 寄存器中可获得 PECERR 错误标记/中断
- 如果 DMA 和 PEC 计算器都被激活

- 在发送时：当 I<sup>2</sup>C 接口从 DMA 控制器处接收到 EOT 信号时，它在最后一个字节后自动发送 PECVAL
- 在接收时：当 I<sup>2</sup>C 接口从 DMA 处接收到一个 EOT\_1 信号时，它将自动把下一个字节作为 PECVAL，并且将检查它。在接收到 PECVAL 后产生一个 DMA 请求
- 为了允许中间 PEC 传输，在 I2C\_CTRL2 寄存器中有一个控制位(DMALAST)用于判断是否真是最后一个 DMA 传输。如果确实是最后一个主接收器的 DMA 请求，在接收到最后一个字节后自动发送 NACK
- 仲裁丢失时 PEC 计算失效

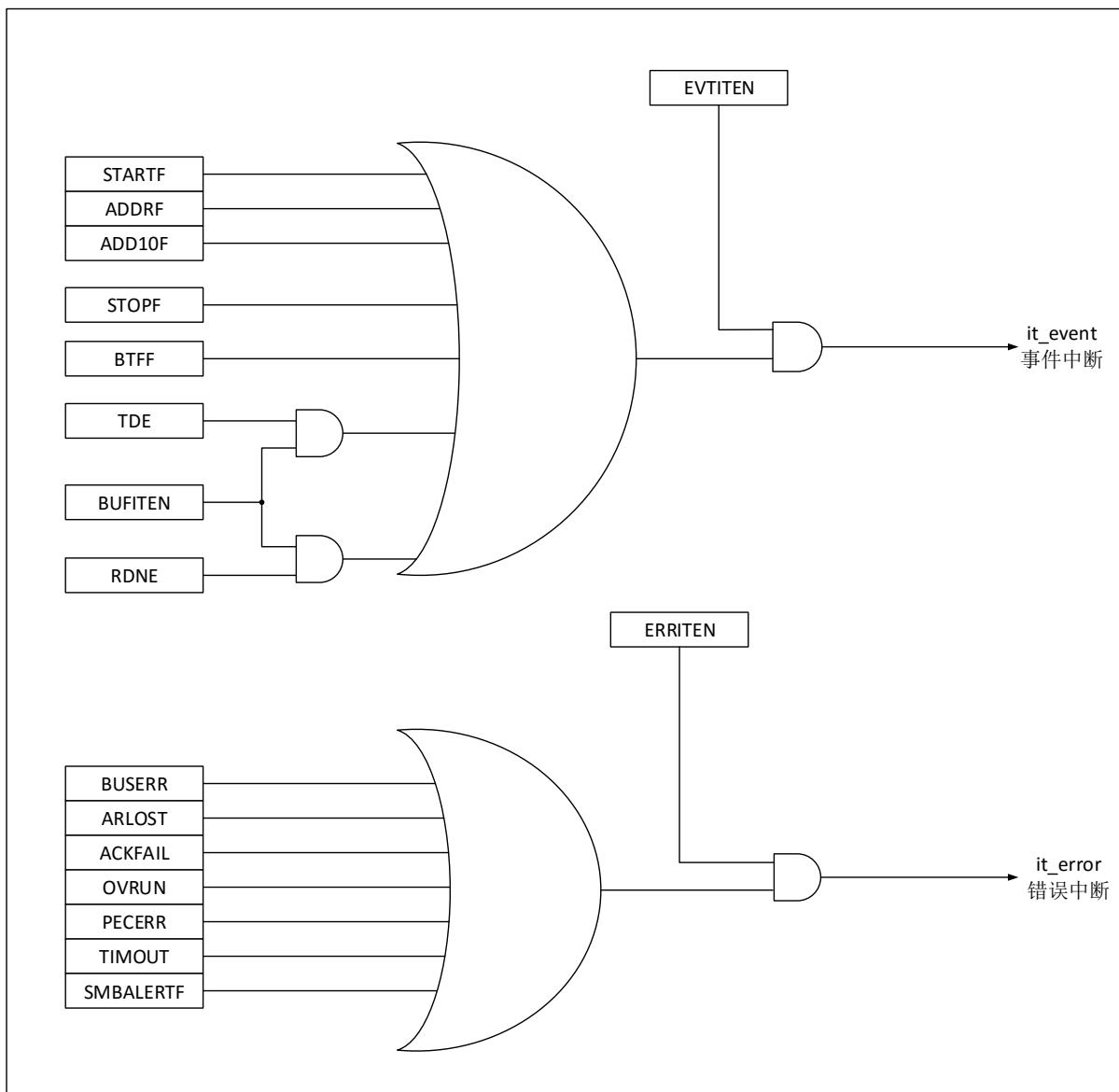
### 15.3.9 I<sup>2</sup>C 中断请求

下表列出了所有的 I<sup>2</sup>C 中断请求。

表格 15-2 I<sup>2</sup>C 中断请求表

中断事件	事件标志	开启控制位
起始位已发送(主)	STARTF	EVTITEN
地址已发送(主)或地址匹配(从)	ADDRF	
10 位头段已发送(主)	ADDR10F	
已收到停止(从)	STOPF	
数据字节传输完成	BTFF	
接收缓冲区非空	RDNE	EVTITEN 和 BUFITEN
发送缓冲区空	TDE	
总线错误	BUSERR	ERRITEN
仲裁丢失(主)	ARLOST	
响应失败	ACKFAIL	
过载/欠载	OVRUN	
PEC 错误	PECERR	
超时/Tlow 错误	TIMOUT	
SMBus 提醒	SMBALERTF	

注意： 1. STARTF、ADDRF、ADDR10F、STOPF、BTFF、RDNE 和 TDE 通过逻辑或汇到同一个中断通道中。  
 2. BUSERR、ARLOST、ACKFAIL、OVRUN、PECERR、TIMOUT 和 SMBALERTF 通过逻辑或汇到同一个中断通道中。

图表 15-10 I<sup>2</sup>C中断映射图

### 15.3.10 I<sup>2</sup>C调试模式

当微控制器进入调试模式 (Cortex™-M4 核心处于停止状态) 时，根据 DBG 模块中的 `DBG_I2Cx_SMBUS_TIMEOUT` 配置位，SMBUS 超时控制或者继续正常工作或者可以停止。详见 [18.3.1 节](#)。

## 15.4 I<sup>2</sup>C寄存器描述

可以用半字(16位)或字(32位)的方式操作这些外设寄存器。

表格 15-3 I<sup>2</sup>C寄存器地址映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

000h	I2C_CTRL1	保留	SWRESET	保留	SMBALERT	PECTRA	POSN	ACKEN	STOPGEN	ERRIT EN	STARTGEN	NOCLKSTRECH	GCEN	PECEN	ARPEN	SMBTYPE	保留	SMBMODE	PEN	
			0																	
004h	I2C_CTRL2	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	CLKFREQ[7: 0]			
008h	I2C_OADDR1	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	ADDR[7: 1]		
00Ch	I2C_OADDR2	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	ADDR2[7: 1]		
010h	I2C_DT	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	DT[7: 0]		
014h	I2C_STS1	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	DUALEN 0		
018h	I2C_STS2	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	PECVAL[7: 0]		
01Ch	I2C_CLKCTRL	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	CLKCTRL[11: 0]		
020h	I2C_TMRISE	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	保留	TMRISE[5: 0]		

#### 15.4.1 控制寄存器1(I<sup>2</sup>C\_CTRL1)

地址偏移: 0x00

复位值: 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SW RESET	保留	SMB ALERT	PEC TRA	POS EN	ACK EN	STOP GEN	START GEN	NOCL KSTR ETCH	GCEN	PEC EN	ARP EN	SMB TYPE	保留	SMB MODE	PEN
	rw	res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	res	rw	rw

位 15	<b>SWRESET:</b> 软件复位(Software reset) 当被置位时, I <sup>2</sup> C 处于复位状态。在复位该位前确信 I <sup>2</sup> C 的引脚被释放, 总线是空的。 0: I <sup>2</sup> C 模块不处于复位状态; 1: I <sup>2</sup> C 模块处于复位状态。 注: 该位可以用于 BUSYF 位为'1', 在总线上又没有检测到停止条件时。
位 14	保留位, 硬件强制为 0
位 13	<b>SMBALERT:</b> SMBus 提醒(SMBus alert) 软件可以设置或清除该位; 当 PEN=0 时, 由硬件清除。 0: 释放 SMBAlert 引脚使其变高。提醒响应地址头紧跟在 NACK 信号后面; 1: 驱动 SMBAlert 引脚使其变低。提醒响应地址头紧跟在 ACK 信号后面。
位 12	<b>PECTRA:</b> 数据包出错检测(Packet error checking) 软件可以设置或清除该位; 当传送 PECVAL 后, 或起始或停止条件时, 或当 PEN=0 时硬件将其清除。 0: 无 PEC 传输; 1: PEC 传输(在发送或接收模式)。 注: 仲裁丢失时, PEC 的计算失效。
位 11	<b>POSEN:</b> 应答/PEC 位置(用于数据接收)(Acknowledge/PEC Position (for data reception)) 软件可以设置或清除该位, 或当 PEN=0 时, 由硬件清除。 0: ACKEN 位控制当前移位寄存器内正在接收的字节的(N)ACK。PECTRA 位表明当前移位寄存器内的字节是 PECVAL; 1: ACKEN 位控制在移位寄存器里接收的下一个字节的(N)ACK。PECTRA 位表明在移位寄存器里接收的下一个字节是 PECVAL。 注: POSEN 位只能用在 2 字节的接收配置中, 必须在接收数据之前配置。 为了 NACK 第 2 个字节, 必须在清除 ADDR 位之后清除 ACKEN 位。 为了检测第 2 个字节的 PECVAL, 必须在配置了 POSEN 位之后, 拉伸 ADDR 事件时设置 PECTRA 位。
位 10	<b>ACKEN:</b> 应答使能(Acknowledge enable) 软件可以设置或清除该位, 或当 PEN=0 时, 由硬件清除。 0: 无应答返回; 1: 在接收到一个字节后返回一个应答(匹配的地址或数据)。
位 9	<b>STOPGEN:</b> 停止条件产生(Stop generation) 软件可以设置或清除该位; 或当检测到停止条件时, 由硬件清除; 当检测到超时错误时, 硬件将其置位。 在主模式下: 0: 无停止条件产生; 1: 当数据寄存器和移位寄存器传输完成后或在当前起始条件发出后产生停止条件。 在从模式下: 0: 无停止条件产生; 1: 在当前字节传输完成之后释放 SCL 和 SDA 线。 注: 当设置了 STOPGEN、STARTGEN 或 PECTRA 位, 在硬件清除这个位之前, 软件不要执行任何对 I2C_CTRL1 的写操作; 否则有可能会第 2 次设置 STOPGEN、STARTGEN 或 PECTRA 位。

位 8	<b>STARTGEN:</b> 起始条件产生(Start generation) 软件可以设置或清除该位，或当起始条件发出后或 PEN=0 时，由硬件清除。 在主模式下： 0: 无起始条件产生； 1: 重复产生起始条件。 在从模式下： 0: 无起始条件产生； 1: 当总线空闲时，产生起始条件。
位 7	<b>NOCLKSTRETCH:</b> 禁止时钟延长(从模式) (Clock stretching disable (Slave mode)) 该位用于当 ADDRFF 或 BTFF 标志被置位，在从模式下禁止时钟延长，直到它被软件复位。 0: 允许时钟延长； 1: 禁止时钟延长。
位 6	<b>GCEN:</b> 广播呼叫使能(General call enable) 0: 禁止广播呼叫。不应答地址 00h； 1: 允许广播呼叫。应答地址 00h。
位 5	<b>PECEN:</b> PEC 使能(PEC enable) 0: 禁止 PEC 计算； 1: 开启 PEC 计算。
位 4	<b>ARPEN:</b> ARP 使能(APR enable) 0: 禁止 ARP； 1: 使能 ARP。 如果 SMBTYPE=0，使用 SMBus 设备的默认地址。如果 SMBTYPE=1，使用 SMBus 的主地址。
位 3	<b>SMBTYPE:</b> SMBus 类型(SMBus type) 0: SMBus 设备； 1: SMBus 主机。
位 2	保留位，硬件强制为 0。
位 1	<b>SMBMODE:</b> SMBus 模式(SMBus mode) 0: I <sup>2</sup> C 模式； 1: SMBus 模式。
位 0	<b>PEN:</b> I <sup>2</sup> C 模块使能(Peripheral enable) 0: 禁用 I <sup>2</sup> C 模块； 1: 启用 I <sup>2</sup> C 模块：根据 SMBus 位的设置，相应的 I/O 口需配置为复用功能。 <b>注：</b> 如果清除该位时通讯正在进行，在当前通讯结束后，I <sup>2</sup> C 模块被禁用并返回空闲状态。 由于在通讯结束后发生 PEN=0，所有的位被清除。 在主模式下，通讯结束之前，绝不能清除该位。

注意：当 STARTGEN、STOPGEN 或 PECTRA 设置后，软件应该在相应位被硬件清零后写 I<sup>2</sup>C\_CTRL1，否则有可能产生第二次 STARTGEN、STOPGEN 或 PECTRA 请求。

### 15.4.2 控制寄存器2(I<sup>2</sup>C\_CTRL2)

地址偏移: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	DMA LAST	DM AEN	BUF ITEN	EVT ITEN	ERR ITEN	CLKFREQ[7: 0]									
res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 13	保留位，硬件强制为 0														

位 12	<b>DMALAST:</b> DMA 最后一次传输(DMA last transfer) 0: 下一次 DMA 的 EOT 不是最后的传输; 1: 下一次 DMA 的 EOT 是最后的传输。 注: 该位在主接收模式使用, 使得在最后一次接收数据时可以产生一个 NACK。
位 11	<b>DMAEN:</b> DMA 请求使能(DMA requests enable) 0: 禁止 DMA 请求; 1: 当 TDE=1 或 RDNE =1 时, 允许 DMA 请求。
位 10	<b>BUFITEN:</b> 缓冲器中断使能(Buffer interrupt enable) 0: 当 TDE=1 或 RDNE=1 时, 不产生任何中断; 1: 当 TDE=1 或 RDNE=1 时, 产生事件中断(不管 DMAEN 是何种状态)。
位 9	<b>EVTITEN:</b> 事件中断使能(Event interrupt enable) 0: 禁止事件中断; 1: 允许事件中断。 在下列条件下, 将产生该中断: – STARTF = 1 (主模式) – ADDR10F = 1 (主/从模式) – ADDR10F = 1 (主模式) – STOPF = 1 (从模式) – BTFF = 1, 但是没有 TDE 或 RDNE 事件 – 如果 BUFITEN = 1, TDE 事件为 1 – 如果 BUFITEN = 1, RDNE 事件为 1
位 8	<b>ERRITEN:</b> 出错中断使能(Error interrupt enable) 0: 禁止出错中断; 1: 允许出错中断。 在下列条件下, 将产生该中断: – BUSERR = 1 – ARLOST = 1 – ACKFAIL = 1 – OVRUN = 1 – PECERR = 1 – TIMOUT = 1 – SMBALERTF = 1
位 7: 0	<b>CLKFREQ[7: 0]:</b> I <sup>2</sup> C 模块时钟频率(Peripheral clock frequency) 必须设置正确的输入时钟频率以产生正确的时序, 允许的范围在 2~100MHz 之间: 00000000: 禁用 00000001: 禁用 00000010: 2MHz ... 01100100: 100MHz

### 15.4.3 自身地址寄存器1(I<sup>2</sup>C\_OADDR1)

地址偏移: 0x08

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRMODE	保留	保留	保留	保留	ADDR[9: 8]	ADDR[9: 8]	ADDR[7: 1]	ADDR[7: 1]	ADDR0						

位 15	<b>ADDRMODE:</b> 寻址模式(从模式) (Addressing mode (slave mode)) 0: 7 位从地址(不响应 10 位地址); 1: 10 位从地址(不响应 7 位地址)。
位 14: 10	保留位, 硬件强制为 0。

位 9: 8	<b>ADDR[9: 8]:</b> 接口地址(Interface address) 7 位地址模式时不用关心。 10 位地址模式时为地址的 9~8 位。
位 7: 1	<b>ADDR[7: 1]:</b> 接口地址(Interface address) 地址的 7~1 位。
位 0	<b>ADDR0:</b> 接口地址 (Interface address) 7 位地址模式时不用关心。 10 位地址模式时为地址第 0 位。

#### 15.4.4 自身地址寄存器2(I<sup>2</sup>C\_OADDR2)

地址偏移: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								ADDR2[7: 1]							
res								rw	rw	rw	rw	rw	rw	rw	rw

位 15: 8	保留位, 硬件强制为 0
位 7: 1	<b>ADDR2[7: 1]:</b> 接口地址(Interface address) 在双地址模式下地址的 7~1 位。
位 0	<b>DUALEN:</b> 双地址模式使能位(Dual addressing mode enable) 0: 在 7 位地址模式下, 只有 OADDR1 被识别; 1: 在 7 位地址模式下, OADDR1 和 OADDR2 都被识别。

#### 15.4.5 数据寄存器(I<sup>2</sup>C\_DT)

地址偏移: 0x10

复位值: 0x0000

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

位 15: 8	保留位, 硬件强制为 0
位 7: 0	<b>DT[7: 0]:</b> 8 位数据寄存器(8-bit data register) 用于存放接收到的数据或放置用于发送到总线的数据 发送器模式: 当写一个字节至 DT 寄存器时, 自动启动数据传输。一旦传输开始 (TDE=1), 如果能及时把下一个需传输的数据写入 DT 寄存器, I <sup>2</sup> C 模块将保持连续的数据流。 接收器模式: 接收到的字节被拷贝到 DT 寄存器(RDNE=1)。在接收到下一个字 (RDNE=1)之前读出数据寄存器, 即可实现连续的数据传送。 注: 在从模式下, 地址不会被拷贝进数据寄存器 DT; 注: 硬件不管理写冲突(如果 TDE=0, 仍能写入数据寄存器); 注: 如果在处理 ACK 脉冲时发生 ARLOST 事件, 接收到的字节不会被拷贝到数据寄存器里, 因此不能读到它。

#### 15.4.6 状态寄存器1(I<sup>2</sup>C\_STS1)

地址偏移: 0x14

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMB ALER TF	TIM OUT	保留	PEC ERR	OVR UN	ACK FAIL	AR LOS T	BUS ERR	TDE	RDN E	保留	STO PF	ADD R10F	BTF F	ADD RF	STA RTF
rcw0	rcw0	res	rcw0	rcw0	rcw0	rcw0	rcw0	r	r	res	r	r	r	r	r

位 15	<b>SMBALERTF:</b> SMBus 提醒(SMBus alert) 在 SMBus 主机模式下： 0: 无 SMBus 提醒； 1: 在引脚上产生 SMBALERTF 提醒事件。 在 SMBus 从机模式下： 0: 没有 SMBAlert 响应地址头序列； 1: 收到 SMBAlert 响应地址头序列至 SMBAlert 变低。 – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除。
位 14	<b>TIMOUT:</b> 超时或 Tlow 错误(Timeout or Tlow error) 0: 无超时错误； 1 : SCL 处于低电平已达到25ms(超时)； 或者 主机低电平累积时钟扩展时间超过 10ms(Tlow: mext); 或从设备低电平累积时钟扩展时间超过 25ms(Tlow: sext)。 – 当在从模式下设置该位：从设备复位通讯，硬件释放总线 – 当在主模式下设置该位：硬件发出停止条件 – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除 注：这个功能仅在 SMBUS 模式下有效
位 13	保留位，硬件强制为 0。
位 12	<b>PECERR:</b> 在接收时发生 PEC 错误(PEC Error in reception) 0: 无 PEC 错误：接收到 PEC 后接收器返回 ACK(如果 ACKEN=1); 1: 有 PEC 错误：接收到 PEC 后接收器返回 NACK(不管 ACKEN 是什么值)。 – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除
位 11	<b>OVRUN:</b> 过载/欠载(Overrun/Underrun) 0: 无过载/欠载； 1: 出现过载/欠载。 – 当 NOCLKSTRETCH=1 时，在从模式下该位被硬件置位，同时 – 在接收模式中当收到一个新的字节时(包括 ACK 应答脉冲)，数据寄存器里的内容还未被读出，则新接收的字节将丢失 – 在发送模式中当要发送一个新的字节时，却没有新的数据写入数据寄存器，同样的字节将被发送两次 – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除 注：如果数据寄存器的写操作发生时间非常接近 SCL 的上升沿，发送的数据是不确定的，并发生保持时间错误。
位 10	<b>ACKFAIL:</b> 应答失败(Acknowledge failure) 0: 没有应答失败； 1: 应答失败。 – 当没有返回应答时，硬件将置该位为'1' – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除
位 9	<b>ARLOST:</b> 仲裁丢失 (主模式) (Arbitration lost (master mode)) 0: 没有检测到仲裁丢失； 1: 检测到仲裁丢失。当接口失去对总线的控制给另一个主机时，硬件将置该位为'1'。 – 该位由软件写'0'清除，或在 PEN=0 时由硬件清除 在 ARLOST 事件之后，I <sup>2</sup> C 接口自动切换回从模式(MSF=0)。 注：在 SMBUS 模式下，在从模式下对数据的仲裁仅仅发生在数据阶段，或应答传输区间(不包括地址的应答)。

位 8	<b>BUSERR:</b> 总线出错(Bus error) 0: 无起始或停止条件出错; 1: 起始或停止条件出错。 – 当接口检测到错误的起始或停止条件, 硬件将该位置'1' – 该位由软件写'0'清除, 或在PEN=0时由硬件清除。 在发生BUSERR错误时, 软件应及时清除该标志位, 然后才能保证重新正常通信。
位 7	<b>TDE:</b> 数据寄存器为空 (发送时) (Data register empty (transmitters)) 0: 数据寄存器非空; 1: 数据寄存器空。 – 在发送数据时, 数据寄存器为空时该位被置'1', 在发送地址阶段不设置该位 – 软件写数据到 DT 寄存器可清除该位; 或在发生一个起始或停止条件后, 或当 PEN=0 时由硬件自动清除 如果收到一个 NACK, 或下一个要发送的字节是 PECVAL(PECTRA=1), 该位不被置位。 <b>注:</b> 在写入第 1 个要发送的数据后, 或设置了 BTFF 时写入数据, 都不能清除 TDE 位, 这是因为数据寄存器仍然为空。
位 6	<b>RDNE:</b> 数据寄存器非空 (接收时) (Data register not empty (receivers)) 0: 数据寄存器为空; 1: 数据寄存器非空。 – 在接收时, 当数据寄存器不为空, 该位被置'1'。在接收地址阶段, 该位不被置位 – 软件对数据寄存器的读写操作清除该位, 或当 PEN=0 时由硬件清除 在发生 ARLOST 事件时, RDNE 不被置位。 <b>注:</b> 当设置了 BTFF 时, 读取数据不能清除 RDNE 位, 因为数据寄存器仍然为满。
位 5	保留位, 硬件强制为 0
位 4	<b>STOPF:</b> 停止条件检测位 (从模式) (Stop detection (slave mode)) 0: 没有检测到停止条件; 1: 检测到停止条件。 – 在一个应答之后(如果 ACKEN=1), 当从设备在总线上检测到停止条件时, 硬件将该位置'1' – 软件读取 STS1 寄存器后, 对 CTRL1 寄存器的写操作将清除该位, 或当 PEN=0 时, 硬件清除该位 <b>注:</b> 在收到 NACK 后, STOPF 位不被置位。
位 3	<b>ADDR10F:</b> 10 位头序列已发送 (主模式) (10-bit header sent (Master mode)) 0: 没有 ADDR10F 事件发生; 1: 主设备已经将第一个地址字节发送出去。 – 在 10 位地址模式下, 当主设备已经将第一个字节发送出去时, 硬件将该位置'1' – 软件读取 STS1 寄存器后, 对 CTRL1 寄存器的写操作将清除该位, 或当 PEN=0 时, 硬件清除该位 <b>注:</b> 收到一个 NACK 后, ADDR10F 位不被置位。
位 2	<b>BTFF:</b> 字节发送结束(Byte transfer finished) 0: 字节发送未完成; 1: 字节发送结束。 当 NOCLKSTRETCH=0 时, 在下列情况下硬件将该位置'1': – 在接收时, 当收到一个新字节(包括 ACK 脉冲)且数据寄存器还未被读取(RDNE=1) – 在发送时, 当一个新数据将被发送且数据寄存器还未被写入新的数据(TDE=1) – 在软件读取 STS1 寄存器后, 对数据寄存器的读或写操作将清除该位; 或在传输中发送一个起始或停止条件后, 或当 PEN=0 时, 由硬件清除该位 <b>注:</b> 在收到一个 NACK 后, BTFF 位不会被置位。 如果下一个要传输的字节是 PECVAL(I2C_STS2 寄存器中 TRF 为'1', 同时 I2C_CTRL1 寄存器中 PECTRA 为'1'), BTFF 位不会被置位。

位 1	<p><b>ADDRF:</b> 地址已被发送(主模式)/地址匹配(从模式)(Address sent (master mode)/matched(slave mode)) 在软件读取 STS1 寄存器后, 对 STS2 寄存器的读操作将清除该位, 或当 PEN=0 时, 由硬件清除该位。</p> <p><b>地址匹配(从模式)</b></p> <ul style="list-style-type: none"> <li>0: 地址不匹配或没有收到地址;</li> <li>1: 收到的地址匹配。       <ul style="list-style-type: none"> <li>- 当收到的从地址与 OADDR 寄存器中的内容相匹配、或发生广播呼叫、或 SMBus 设备默认地址 或 SMBus 主机识别出 SMBus 提醒时, 硬件就将该位置'1'(当对应的设置被使能时)</li> </ul> </li> </ul> <p><b>地址已被发送(主模式)</b></p> <ul style="list-style-type: none"> <li>0: 地址发送没有结束;</li> <li>1: 地址发送结束。       <ul style="list-style-type: none"> <li>- 10 位地址模式时, 当收到地址的第二个字节的 ACK 后该位被置'1'</li> <li>- 7 位地址模式时, 当收到地址的 ACK 后该位被置'1'</li> </ul> </li> </ul> <p>注: 在收到 NACK 后, ADDRF 位不会被置位。</p>
位 0	<p><b>STARTF:</b> 起始位(主模式)(Start bit(Master mode)) 0: 未发送起始条件; 1: 起始条件已发送。       <ul style="list-style-type: none"> <li>- 当发出起始条件时该位被置'1'</li> <li>- 软件读取 STS1 寄存器后, 写数据寄存器的操作将清除该位, 或当 PEN=0 时, 硬件清除该位</li> </ul> </p>

### 15.4.7 状态寄存器2(I<sup>2</sup>C\_STS2)

地址偏移: 0x18

复位值: 0x0000

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

位 15: 8	<p><b>PECVAL[7: 0]:</b> 数据包出错检测(Packet error checking register) 当 PECEN=1 时, PECVAL[7: 0]存放内部的 PEC 的值, 当 PECEN 重置时清零。</p>
位 7	<p><b>DUALF:</b> 双标志(从模式)(Dual flag (Slave mode)) 0: 接收到的地址与 OADDR1 内的内容相匹配; 1: 接收到的地址与 OADDR2 内的内容相匹配。       <ul style="list-style-type: none"> <li>- 在产生一个停止条件或一个重复的起始条件时, 或 PEN=0 时, 硬件将该位清除</li> </ul> </p>
位 6	<p><b>SMBHOSTADDRF:</b> SMBus 主机头系列(从模式) (SMBus host header (Slave mode)) 0: 未收到 SMBus 主机的地址; 1: 当 SMBTYPE=1 且 ARPEN=1 时, 收到 SMBus 主机地址。       <ul style="list-style-type: none"> <li>- 在产生一个停止条件或一个重复的起始条件时, 或 PEN=0 时, 硬件将该位清除</li> </ul> </p>
位 5	<p><b>SMBDEFTADDRF:</b> SMBus 设备默认地址(从模式) (SMBus device default address (Slave mode)) 0: 未收到 SMBus 设备的默认地址; 1: 当 ARPEN=1 时, 收到 SMBus 设备的默认地址。       <ul style="list-style-type: none"> <li>- 在产生一个停止条件或一个重复的起始条件时, 或 PEN=0 时, 硬件将该位清除</li> </ul> </p>
位 4	<p><b>GCADDRF:</b> 广播呼叫地址(从模式) (General call address (Slave mode)) 0: 未收到广播呼叫地址; 1: 当 GCEN=1 时, 收到广播呼叫的地址。       <ul style="list-style-type: none"> <li>- 在产生一个停止条件或一个重复的起始条件时, 或 PEN=0 时, 硬件将该位清除</li> </ul> </p>

位 3	保留位, 硬件强制为 0
位 2	<b>TRF:</b> 发送/接收(Transmitter/receiver) 0: 接收到数据; 1: 数据已发送; 在整个地址传输阶段的结尾, 该位根据地址字节的 R/W 位来设定在检测到停止条件(STOPF=1)、重复的起始条件或总线仲裁丢失(ARLOST=1)后, 或当 PEN=0 时, 硬件将其清除。
位 1	<b>BUSYF:</b> 总线忙(Bus busy) 0: 在总线上无数据通讯; 1: 在总线上正在进行数据通讯。 - 在检测到 SDA 或 SCI 为低电平时, 硬件将该位置'1' - 当检测到一个停止条件时, 硬件将该位清除 该位指示当前正在进行的总线通讯, 当接口被禁用(PEN=0)时该信息仍然被更新。
位 0	<b>MSF:</b> 主从模式(Master/slave) 0: 从模式; 1: 主模式。 - 当接口处于主模式(STARTF=1)时, 硬件将该位置位 - 当总线上检测到一个停止条件、仲裁丢失(ARLOST=1 时)、或当 PEN=0 时, 硬件清除该位

### 15.4.8 时钟控制寄存器(I<sup>2</sup>C\_CLKCTRL)

地址偏移: 0x1C

复位值: 0x0000

- 注意
- 要求 FPCLK1 应当是 10MHz 的整数倍, 这样可以正确地产生 400KHz 的快速时钟。
  - CLKCTRL 寄存器只有在关闭 I<sup>2</sup>C 时(PEN=0)才能设置

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
F/S MODE	FM DUTY	保留	CLKCTRL[11: 0]													
rw	rw	res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>F/S MODE:</b> I <sup>2</sup> C 主模式选项 (I <sup>2</sup> C master mode selection) 0: 标准模式的 I <sup>2</sup> C; 1: 快速模式的 I <sup>2</sup> C。
位 14	<b>FMDUTY:</b> 快速模式时的占空比 (Fast mode duty cycle) 0: 快速模式下: $T_{low}/T_{high} = 2$ ; 1: 快速模式下: $T_{low}/T_{high} = 16/9$ (见 CLKCTRL)。
位 13: 12	保留位, 硬件强制为 0。

位 11: 0	<p><b>CLKCTRL[11: 0]</b> : 快速 / 标准模式下的时钟控制分频系数(主模式)(Clock control register in Fast/Standard mode (Master mode))          该分频系数用于设置主模式下的 SCL 时钟。  <u>在 I<sup>2</sup>C 标准模式或 SMBus 模式下:</u></p> $T_{high} = CLKCTRL \times T_{PCLK1}$ $T_{low} = CLKCTRL \times T_{PCLK1}$ <p><u>在 I<sup>2</sup>C 快速模式下:</u></p> <p>如果 FMDUTY = 0:</p> $T_{high} = CLKCTRL \times T_{PCLK1}$ $T_{low} = 2 \times CLKCTRL \times T_{PCLK1}$ <p>如果 FMDUTY = 1: (速度达 400kHz)</p> $T_{high} = 9 \times CLKCTRL \times T_{PCLK1}$ $T_{low} = 16 \times CLKCTRL \times T_{PCLK1}$ <p>例如: 在标准模式下, 产生 100kHz 的 SCL 的频率:          如果 CLKFREQ = 08, T<sub>PCLK1</sub> = 125ns, 则 CLKCTRL 必须写入 0x28(40×125ns = 5000ns)。</p> <p>注: 1. 允许设定的最小值为 0x04, 在快速 DUTY 模式下允许的最小值为 0x01;          2. T<sub>high</sub>=t<sub>r</sub>(SCL)+t<sub>w</sub>(SCLH), 详见数据手册中对这些参数的定义;          3. T<sub>low</sub>=t<sub>r</sub>(SCL)+t<sub>w</sub>(SCLL), 详见数据手册中对这些参数的定义;          4. 这些延时没有过滤器;          5. 只有在关闭 I<sup>2</sup>C 时(PEN = 0)才能设置 CLKCTRL 寄存器;          6. f<sub>CK</sub> 应当是 10MHz 的整数倍, 这样可以正确产生 400kHz 的快速时钟。</p>
---------	--

### 15.4.9 TMRISE寄存器(I<sup>2</sup>C\_TMRISE)

地址偏移: 0x20

复位值: 0x0002

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

res

rw

位 15: 6	保留位, 硬件强制为 0
位 5: 0	<p><b>TMRISE[5: 0]</b> : 在快速/标准模式下的最大上升时间(主模式)(Maximum rise time in Fast/Standard mode (Master mode))          这些位必须设置为 I<sup>2</sup>C 总线规范里给出的最大 SCL 上升时间, 增长步幅为 1。          例如: 标准模式中最大允许 SCL 上升时间为 1000ns。如果在 I2C_CTRL2 寄存器中 CLKFREQ[7: 0]中的值等于 0x08 且 T<sub>PCLK1</sub>=125ns, 故 TMRISE[5: 0]中必须写入 09h(1000ns/125 ns = 8+1)。滤波器的值也可以加到 TMRISE[5: 0]内。          如果结果不是一个整数, 则将整数部分写入 TMRISE[5: 0]以确保 t<sub>HIGH</sub> 参数。          注: 只有当 I<sup>2</sup>C 被禁用(PEN=0)时, 才能设置 TMRISE[5: 0]。</p>

# 16 通用同步异步收发器 (USART)

## 16.1 USART介绍

通用同步异步收发器 (USART) 提供了一种灵活的方法与使用工业标准 NRZ 异步串行数据格式的外部设备之间进行全双工数据交换。USART 利用分数波特率发生器提供宽范围的波特率选择。

它支持同步单向通信和半双工单线通信，也支持 LIN (局部互连网)，智能卡协议和 IrDA (红外数据组织) SIRENDEC 规范，以及 CTS/RTS (Clear To Send/Request To Send) 硬件流操作。它还允许多处理器通信。

使用多缓冲器配置的 DMA 方式，可以实现高速数据通信。

## 16.2 USART主要特性

- 全双工异步通信
- 单线半双工通信
- NRZ 标准格式 (Mark/Space)
- 可编程的波特率发生器
  - 发送和接收共用的可编程波特率，最高达 6.25MBits/s
- 可编程数据字长度 (8 位或 9 位)
- 可配置的停止位-支持 1 或 2 个停止位
- LIN 主机有发送断开符的能力以及 LIN 从机有检测断开符的能力
  - 当 USART 硬件配置成 LIN 时，生成 13 位断开符；检测 10/11 位断开符
- 发送方为同步传输提供时钟
- IrDA SIR 编码器解码器
  - 在普通模式下支持 3/16 位的持续时间
- ISO7816-3 标准里定义的异步智能卡协议
  - 智能卡模式支持 0.5 或 1.5 个停止位
- 可配置的 DMA 多缓冲器通信
  - 利用 DMA 缓冲接收/发送数据
- 单独的发送器和接收器使能位
- 检测标志
  - 接收缓冲器满
  - 发送缓冲器空
  - 传输结束标志
- 校验控制
  - 发送校验位
  - 对接收数据进行校验
- 四个错误检测标志
  - 溢出错误
  - 噪音错误
  - 帧错误
  - 校验错误
- 10 个带标志的中断源

- CTSF 改变
  - LIN 断开符检测
  - 发送数据寄存器空
  - 发送完成
  - 接收数据寄存器满
  - 检测到总线为空闲
  - 溢出错误
  - 帧错误
  - 噪音错误
  - 校验错误
- 多处理器通信--如果地址不匹配，则进入静默模式
  - 从静默模式中唤醒（通过空闲总线检测或地址检测）
  - 两种唤醒接收器的方式：地址位（MSB，第9位），总线空闲

### 16.3 USART功能概述

接口通过三个引脚与其他设备连接在一起（见图 16-1）。任何 USART 双向通信至少需要两个脚：接收数据输入（RX）和发送数据输出（TX）。

**RX：**串行数据输入端。利用过采样技术识别数据和噪音以恢复数据。

**TX：**串行数据输出端。当发送器被禁止时，输出引脚恢复到它的 I/O 端口配置。当发送器被激活，并且不发送数据时，TX 引脚处于高电平。在单线和智能卡模式里，此 I/O 口被同时用于数据的发送和接收。

- 总线在发送或接收前应处于空闲状态
- 一个起始位
- 一个数据字（8 或 9 位），最低有效位在前
- 0.5, 1, 1.5, 2 个停止位，表示数据帧的结束
- 使用分数波特率发生器——12 位整数和 4 位小数的表示方法。
- 一个状态寄存器（USART\_STS）
- 数据寄存器（USART\_DT）
- 一个波特率寄存器（USART\_BAUDR），12 位的整数和 4 位小数
- 一个智能卡模式下的保护时间寄存器（GTVVAL）

关于以上寄存器中每个位的具体定义，请参考寄存器描述第 16.6 节：USART 寄存器描述。

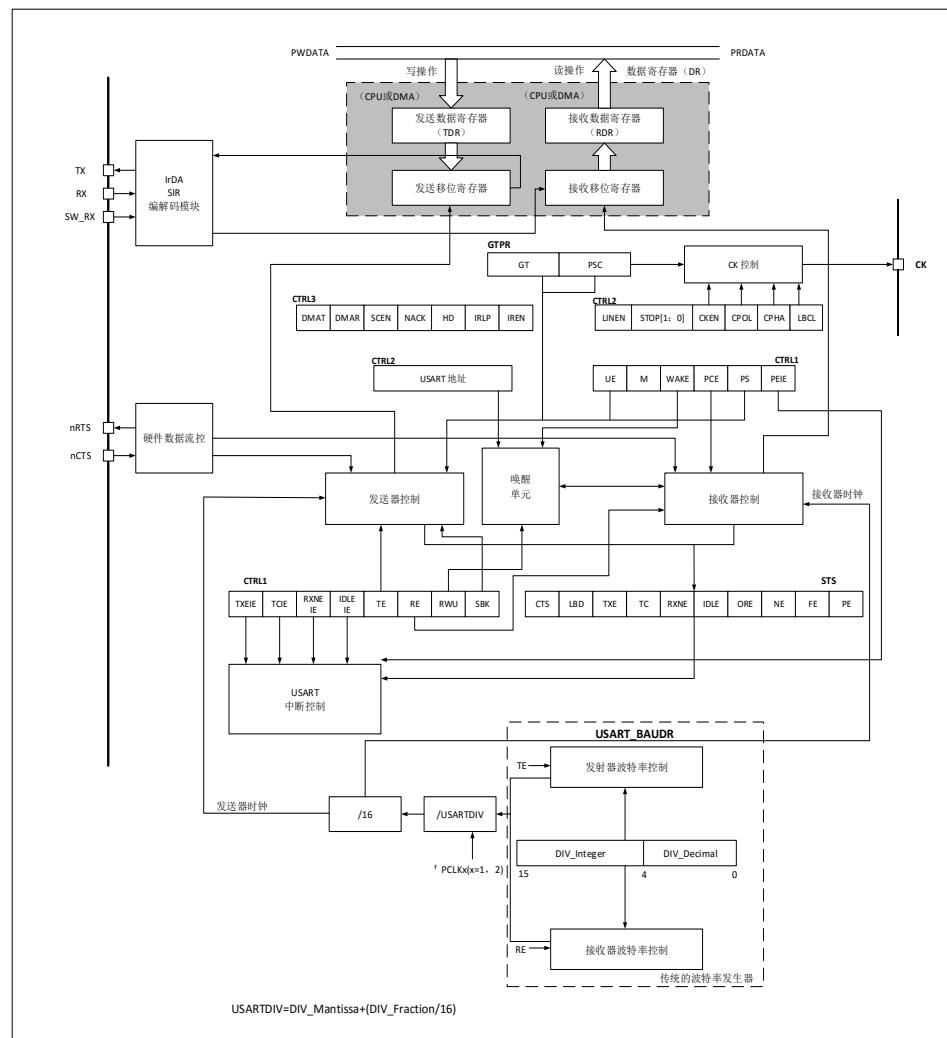
在同步模式中需要下列引脚：

- **CK：**发送器时钟输出。此引脚输出用于同步传输的时钟，(在 Start 位和 Stop 位上没有时钟脉冲，软件可选地，可以在最后一个数据位送出一个时钟脉冲)。数据可以在 RX 上同步被接收。这可以用来控制带有移位寄存器的外部设备（例如 LCD 驱动器）。时钟相位和极性都是软件可编程的。在智能卡模式里，CK 可以为智能卡提供时钟。

在硬件流控模式中需要下列引脚：

- **CTS：**接收清零，若为低电平，表明在当前数据传输结束时可继续下一次的数据发送；若为高电平，在当前数据传输结束时阻断下一次的数据发送。
- **RTS：**发送请求，若为低电平，表明 USART 准备好接收数据。

图表 16-1 USART 框图



### 16.3.1 USART特性描述

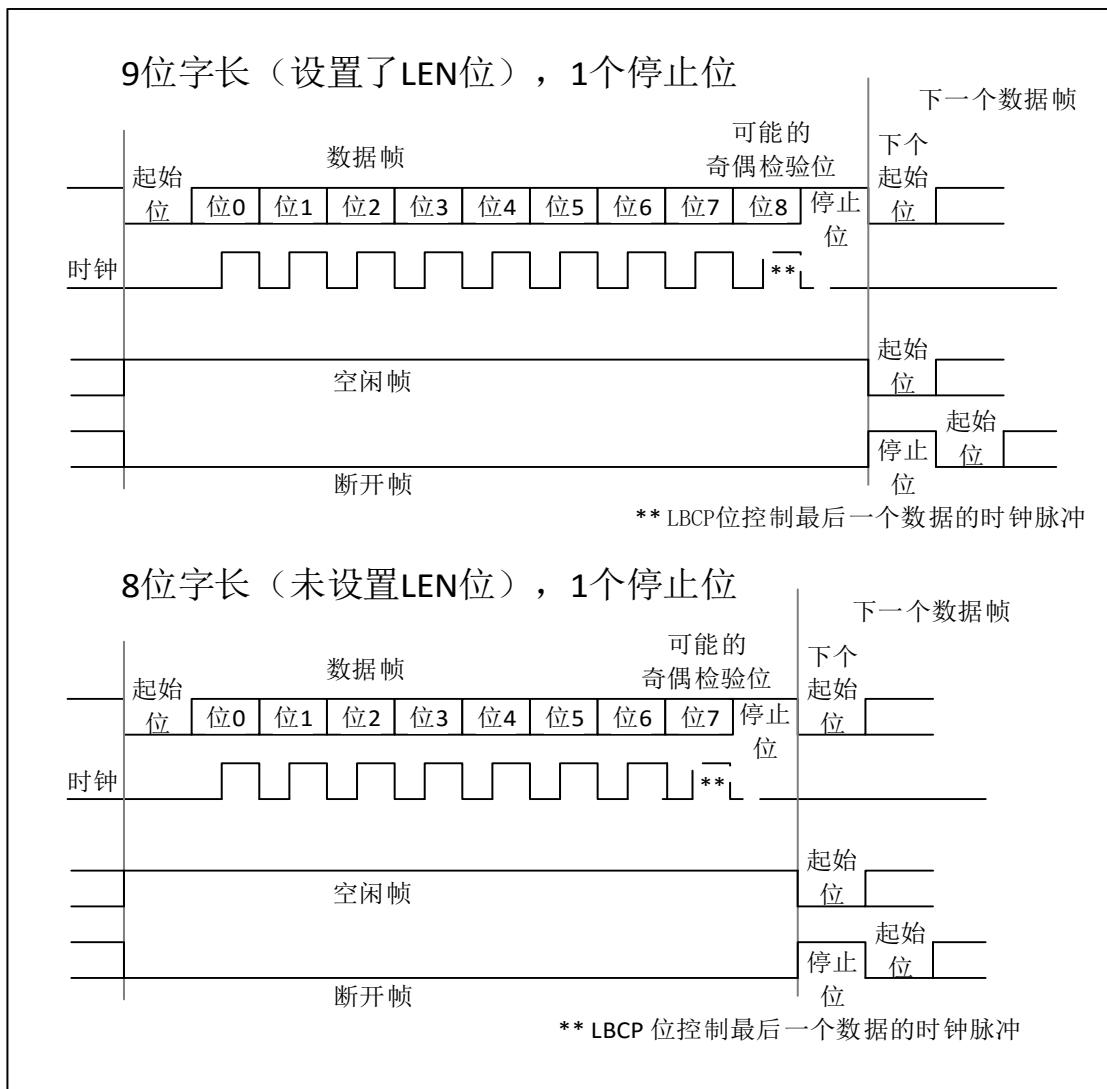
字长可以通过编程 **USART\_CTRL1** 寄存器中的 **LEN** 位，选择成 8 或 9 位（见图 16-2）。在起始位期间，**TX** 脚处于低电平，在停止位期间处于高电平。

空闲帧：全部由‘1’组成的一个完整的数据帧，也包含了数据的停止位。例如：如果 **LEN=0**，则空闲帧由 10 个‘1’组成；如果 **LEN=1**，则空闲帧由 11 个‘1’组成。

断开帧：被视为在一个帧周期内全部收到‘0’（包括停止位期间，也是‘0’）。在断开帧结束时，发送器再插入 1 或 2 个停止位（‘1’）来应答起始位。

发送和接收均由一个共用的波特时钟发生器驱动，当发送器和接收器的使能位分别置位时，分别为其产生波特时钟。

图表 16-2 字长设置



**注意：** 在本章中，若未特殊说明，置位均表示某个寄存器被置为状态‘1’，复位或清零均表示某个寄存器被置为状态‘0’；硬件或者程序均可能置位或者清零某个寄存器，请参考本章具体内容。

### 16.3.2 发送器

发送器根据 LEN 位的状态发送 8 位或 9 位的数据字。当发送使能位（TEN）被置位时，发送移位寄存器中的数据在 TX 脚上输出；通过配置，相应的时钟脉冲在 CK 脚上输出。

#### 16.3.2.1 字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式里，USART\_DT 寄存器包含了一个内部总线和发送移位寄存器之间的缓冲器（见图 16-1）。

每个字符之前都有一个低电平的起始位；之后跟着的停止位，其数目可配置。

USART 支持多种停止位的配置：0.5、1、1.5 和 2 个停止位。

**注意：** 1. 在数据传输期间不能复位 TEN 位，否则将破坏 TX 脚上的数据，因为波特率计数器停止计数。正在传输的当前数据将丢失。  
2. TEN 位被激活后，USART 将自动发送一个空闲帧。

#### 16.3.2.2 可配置的停止位

每个字符发送的停止位的位数可以通过控制寄存器 2（USART\_CTRL2）的位 13、12 进行编程。

1. 1 个停止位：停止位位数的默认值。

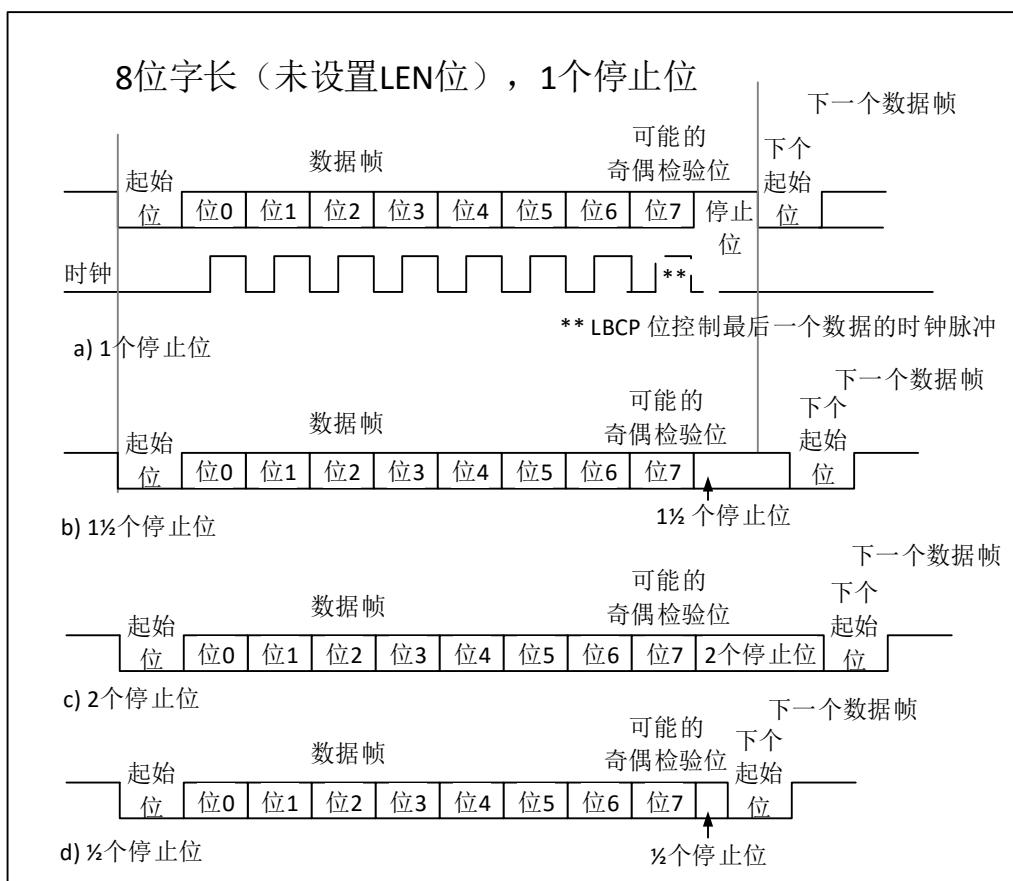
2. 2个停止位：可用于常规 USART 模式、单线模式以及调制解调器模式。

3. 0.5 个停止位：在智能卡模式下接收数据时使用。

4. 1.5 个停止位：在智能卡模式下发送和接收数据时使用。空闲帧包括了停止位。

断开帧是 10 位低电平，后跟停止位（当 LEN=0 时）；或者 11 位低电平，后跟停止位（LEN=1 时）。不可能传输更长的断开帧（长度大于 10 或者 11 位）。

图表 16-3 配置停止位



配置步骤：

1. 通过对 USART\_CTRL1 寄存器的 UEN 位写入 1 来激活 USART。
2. 配置 USART\_CTRL1 的 LEN 位来定义字长。
3. 配置 USART\_CTRL2 的 STOPB 位来定义停止位。
4. 如果采用多缓冲器通信，配置 USART\_CTRL3 中的 DMA 使能位 (DMATEN)。按多缓冲器通信中的描述配置 DMA 寄存器。
5. 配置 USART\_BAUDR 寄存器来定义波特率。
6. 配置 USART\_CTRL1 中的 TEN 位（也即向 TEN 位写入 1），USART 会自动发送一个空闲帧作为第一次数据发送。
7. 把要发送的数据写进 USART\_DT 寄存器（此动作清除 TDE 位）。在只有一个缓冲器的情况下，对每个待发送的数据重复步骤 7。
8. 在 USART\_DT 寄存器中写入最后一个数据字后，要等待 TRAC=1，它表示最后一个数据帧的传输结束。当需要关闭 USART 或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次数据传输。

### 16.3.2.3 单字节通信

清零 TDE 位总是通过对数据寄存器的写操作来完成的。

TDE 位由硬件（也即 USART IP 核）来置位，这表明：

- 数据已经从 TDR 寄存器（Transmitter Data Register）移送到移位寄存器，数据发送已经开始。
- TDR 寄存器被清空。
- 下一个数据可以被写进 USART\_DT 寄存器而不会覆盖先前的数据。

如果 TDEIEN 位被设置，此标志将产生一个中断。如果此时 USART 正在发送数据，对 USART\_DT 寄存器的写操作把数据存入 TDR 寄存器，并在当前传输结束时把该数据复制到移位寄存器。

如果此时 USART 没有发送数据，处于空闲状态，对 USART\_DT 寄存器的写操作直接把数据放进移位寄存器，数据传输开始，TDE 位立即被置起（也即写入“1”）。

当一帧发送完成时（停止位发送后）并且设置了 TDE 位，TRAC 位被置起，如果 USART\_CTRL1 寄存器中的 TRACIEN 位被置起时，则会产生中断。

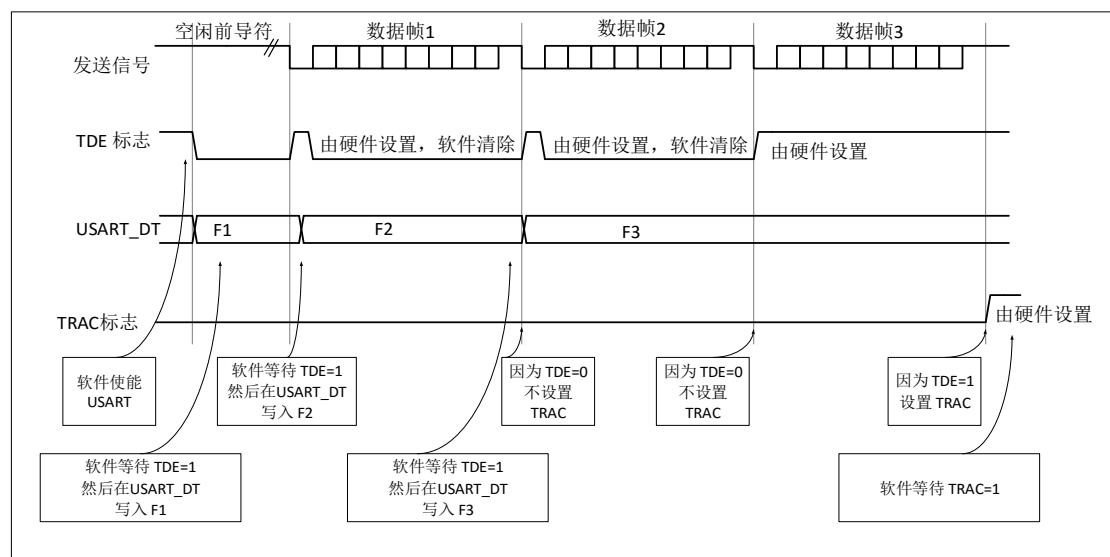
在 USART\_DT 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式（见图 16-4）之前，必须先等待 TRAC=1。

使用下列软件过程清除 TRAC 位：

1. 读一次 USART\_STS 寄存器；
2. 写一次 USART\_DT 寄存器。

**注意：** TRAC 位也可以通过软件对它写‘0’来清除。此清零方式只推荐在 DMA 多缓冲器通信模式下使用。

图表 16-4 发送时 TRAC/TDE 的变化情况



#### 16.3.2.4 断开帧

设置 SBRK 可发送一个断开帧。断开帧长度取决 LEN 位（见图 16-2）。如果设置 SBRK=1，在完成当前数据发送后，将在 TX 线上发送一个断开帧。断开字符发送完成时（在断开帧的停止位时）SBRK 被硬件复位。USART 在最后一个断开帧的结束处插入一逻辑‘1’，以保证能识别下一帧的起始位。

**注意：** 若在开始发送断开帧之前，软件又复位了 SBRK 位，断开帧将不被发送。如果要发送两个连续的断开帧，SBRK 位应该在前一个断开帧的停止位之后置起。

#### 16.3.2.5 空闲符号

置位 TEN 将使得 USART 在第一个数据帧前发送一空闲帧。

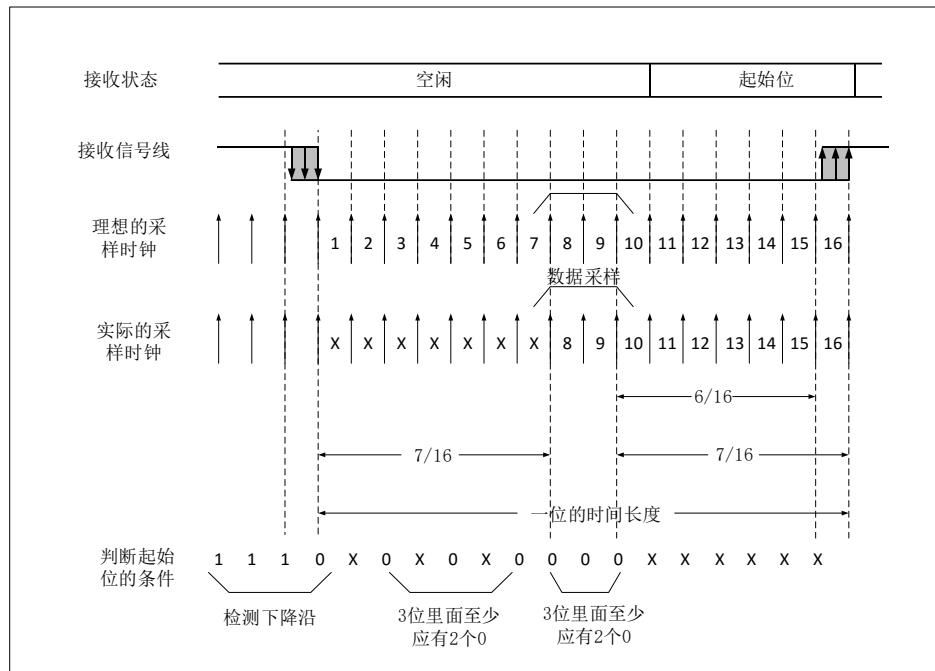
### 16.3.3 接收器

USART 可以根据 USART\_CTRL1 的 LEN 位接收 8 位或 9 位的数据字。

#### 16.3.3.1 起始位侦测

在 USART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。该序列为：1110X0X0X0000。

图表 16-5 起始位侦测



**注意：**如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态（不设置任何标志位）等待下降沿。

1. 在第 3、5、7 位的采样，以及在第 8、9、10 位的采样都为‘0’（也即 6 个‘0’），则确认收到起始位，并且不会置位 NERR 噪声标志位。
2. 第 3、5、7 位的采样有两个‘0’，与此同时，第 8、9、10 位的采样有两个‘0’点，那么确认收到起始位，但是会置位 NERR 噪声标志位。
3. 第 3、5、7 位的采样有两个‘0’，与此同时，第 8、9、10 位的采样有三个‘0’点，那么确认收到起始位，但是会置位 NERR 噪声标志位。
4. 第 3、5、7 位的采样有三个‘0’，与此同时，第 8、9、10 位的采样有两个‘0’点，那么确认收到起始位，但是会置位 NERR 噪声标志位。
5. 如果在第 3、5、7、8、9、10 位的采样值满足不了上面四种要求，则该 USART 接收器认为没有接受到正确的起始位，将退出起始位侦测并回到空闲状态等待下降沿。

#### 16.3.3.2 字符接收

在 USART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里，USART\_DT 寄存器包含的缓冲器位于内部 APB 总线和接收移位寄存器之间。

配置步骤：

1. 通过对 USART\_CTRL1 寄存器的 UEN 位写入 1 来激活 USART。
2. 配置 USART\_CTRL1 的 LEN 位来定义字长。
3. 配置 USART\_CTRL2 的 STOPB 位来定义停止位。

4. 如果采用多缓冲器通信，配置 USART\_CTRL3 中的 DMA 使能位 (DMAREN)。按多缓冲器通信中的描述配置 DMA 寄存器。
5. 配置 USART\_BAUDR 寄存器来定义波特率。
6. 配置 USART\_CTRL1 中的 REN 位 (也即向 REN 位写入 1)。激活接收器，使它开始寻找起始位。

当一字符被接收到时：

- RDNE 位被置位。它表明移位寄存器的内容被转移到 RDR (Receiver Data Register)。换句话说，数据已经被接收并且可以被读出（包括与之有关的错误标志）。
- 如果 RDNEIEN 位被设置，则产生中断。
- 在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起。
- 在多缓冲器通信时，RDNE 在每个字节接收后被置起，并由 DMA 对数据寄存器的读操作而清零。
- 在单缓冲器模式里，由软件读 USART\_DT 寄存器完成对 RDNE 位清除。RDNE 标志也可以通过对其写 0 来清除。RDNE 位必须在下一帧数据接收结束前被清零，以避免溢出错误。

注意：在接收数据时，REN 位不应该被复位。如果 REN 位在接收时被清零，当前字节的接收被丢失。

### 16.3.3.3 断开帧

当接收到一个断开帧时，USART 按处理帧错误一样处理它。

备注：在 LIN 模式下，当接收到一个断开帧时，将按断开帧处理，LBDF 将会置起。

### 16.3.3.4 空闲符号

当一空闲帧被检测到时，其处理步骤和接收到普通数据帧一样，但如果 IDLEIEN 位被设置将产生一个中断。

### 16.3.3.5 溢出错误

如果接收的数据未及时被读走，导致 RDNE 没有及时被复位，又接收到一个字符，则发生溢出错误。数据只有当 RDNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。RDNE 标记是接收到每个字节后被硬件置位的。如果下一个数据已被收到或先前 DMA 请求还没被服务时，RDNE 标志仍是置起的，溢出错误产生。

当溢出错误产生时：

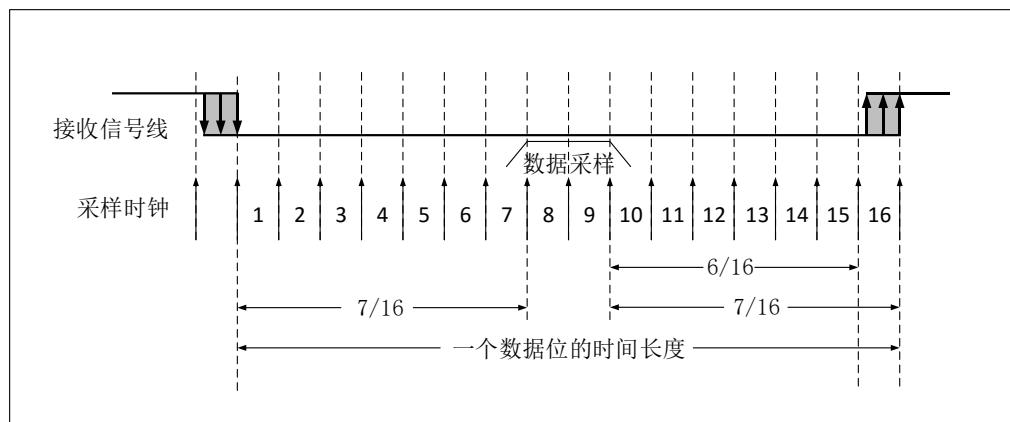
- ORERR 位被置位。
- RDR 内容将不会丢失。读 USART\_DT 寄存器仍能得到先前的数据。
- 移位寄存器中的内容将被覆盖。随后接收到的数据都将丢失。
- 如果 RDNEIEN 位被设置或 ERRIEN 和 DMAREN 位都被设置，中断产生。
- 顺序执行对 USART\_STS 和 USART\_DT 寄存器的读操作，可复位 ORERR。

注意：当 ORERR 置位时，表明至少有 1 个数据已经丢失。有两种可能性：

- 如果 RDNE=1，上一个有效数据还在接收寄存器 RDR 上，可以被读出。
- 如果 RDNE=0，这意味着上一个有效数据已经被读走，RDR 已经没有东西可读。当上一个有效数据在 RDR 中被读取的同时又接收到新的（也就是丢失的）数据时，此种情况可能发生。在读序列期间（在 USART\_STS 寄存器读访问和 USART\_DT 读访问之间）接收到新的数据，此种情况也可能发生。

噪音错误使用过采样技术（同步模式除外），通过区别有效输入数据和噪音来进行数据恢复。

图表 16-6 检测噪声的数据采样



表格 16-1 检测噪音的数据采样

采样值	NERR 状态	接收的位	数据有效性
000	0	0	有效
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效
111	0	1	有效

当在接收帧中检测到噪音时：

- 在 RDNE 位的上升沿设置 NERR 标志。
- 无效数据从移位寄存器传送到 USART\_DT 寄存器。
- 在单个字节通信情况下，没有噪音中断产生。然而，因为 NERR 标志位和 RDNE 标志位是同时被置位，RDNE 将产生中断。在多缓冲器通信情况下，如果已经设置了 USART\_CTRL3 寄存器中 ERRIEN 位，将产生一个中断。

顺序读 USART\_STS，再读 USART\_DT 寄存器，将清除 NERR 标志位。

### 16.3.3.6 帧错误

当以下情况发生时检测到帧错误：由于数据未同步或大量噪音的原因，停止位没有在预期的时间上接收和识别出来。当帧错误被检测到时：

- FERR 位被硬件置起
- 无效数据从移位寄存器传送到 USART\_DT 寄存器。
- 在单字节通信时，没有帧错误中断产生。然而，这个位和 RDNE 位同时置起，后者将产生中断。在多缓冲器通信情况下，如果 USART\_CTRL3 寄存器中 ERRIEN 位被置位的话，将产生中断。

顺序执行对 USART\_STS 和 USART\_DT 寄存器的读操作，可复位 FERR 位。

### 16.3.3.7 接收期间可配置的停止位

在数据接收期间，数据停止位的个数可以通过控制寄存器 2 (USART\_CTRL2 中的 STOPB) 的控制位来配置，在普通模式时，可以是 1 或 2 个。在智能卡模式里可能是 0.5 或 1.5 个。

1. 0.5 个停止位（智能卡模式中的接收）：不对 0.5 个停止位进行采样。因此，如果选择 0.5 个停止位则不能检测帧错误和断开帧。
2. 1 个停止位：对 1 个停止位的采样在第 8, 第 9 和第 10 采样点上进行。
3. 1.5 个停止位（智能卡模式）：当以智能卡模式发送时，器件必须检查数据是否被正确的发送出去。所以接收器功能块必须被激活（USART\_CTRL1 寄存器中的 REN=1），并且在停止位的发送期间采样数据线上的信号。如果出现校验错误，智能卡会在发送方采样 NACK 信号时，即总线上停止位对应的时间内时，拉低数据线，以此表示出现了帧错误。发送端的 FERR 在 1.5 个停止位结束时被置起。对 1.5 个停止位的采样是在第 16, 第 17 和第 18 采样点进行的。对于智能卡发送端而言，1.5 个的停止位可以被分成 2 部分：一个是 0.5 个数据位的周期，期间不做任何事情；随后是 1 个数据位的周期的停止位，在这段时间的中点处采样。对于智能卡接收端而言，1.5 个的停止位也可以被分成 2 部分：一个是 0.5 个数据位的周期，判断是否有奇偶校验错误；如果有奇偶校验错误，则在随后的 1 个数据位的周期拉低数据总线；如果没有奇偶校验错误，则在随后的 1 个数据位的周期，释放数据总线（数据总线处于高电平）。详见第 16.3.11 节：智能卡。
4. 2 个停止位：对 2 个停止位的采样是在第一停止位的第 8, 第 9 和第 10 个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被设置。第二个停止位不再检查帧错误。在第一个停止位结束时 RDNE 标志将被设置。

### 16.3.4 分数波特率的产生

接收器和发送器的波特率在 USARTDIV 的整数和小数寄存器中的值是相同的。

$$\text{TX/RX 波特率} = \frac{f_{CK}}{16 * \text{USARTDIV}}$$

这里的  $f_{CK}$  是给外设的时钟（PCLK1 用于 USART2, PCLK2 用于 USART1。）

USARTDIV 是一个无符号的定点数。该值设置在 USART\_BAUDR 寄存器。

**注意：** 在写入 USART\_BAUDR 之后，波特率计数器会被波特率寄存器的新值替换。因此，不要在通信过程中改变波特率寄存器的数值。

#### 16.3.4.1 如何从 USART\_BAUDR 寄存器值得到 USARTDIV

例 1：

如果 DIV\_Integer=27, DIV\_Decimal=12 (USART\_BAUDR=0x1BC)，于是

DIV\_Integer (USARTDIV) = 27

DIV\_Decimal (USARTDIV) = 12/16=0.75

所以 USARTDIV=27.75

例 2：

要求 USARTDIV=25.62, 就有：

DIV\_Decimal=16\*0.62=9.92

最接近的整数是：10=0xA

DIV\_Integer= DIV\_Integer (25.620) = 25=0x19

于是，USART\_BAUDR=0x19A

例 3：

要求 USARTDIV=50.99 就有：

DIV\_Decimal=16\*0.99=15.84

最接近的整数是：16=0x10=> DIV\_Decimal[3: 0]溢出=>进位必须加到小数部分

DIV\_Integer= DIV\_Integer (0d50.990+进位) = 51=0x33

于是：USART\_BAUDR=0x330, USARTDIV=0d51.000

表格 16-2 设置波特率时的误差计算

波特率		fPCLK=36MHz			fPCLK=72MHz		
序号	Kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.4	937.5	0%	2.4	1875	0%
2	9.6	9.6	234.375	0%	9.6	468.75	0%
3	19.2	19.2	117.1875	0%	19.2	234.375	0%
4	57.6	57.6	39.0625	0%	57.6	78.125	0%
5	115.2	115.384	19.5	0.15%	115.2	39.0625	0%
6	230.4	230.769	9.75	0.16%	230.769	19.5	0.16%
7	460.8	461.538	4.875	0.16%	461.538	9.75	0.16%
8	921.6	923.076	2.4375	0.16%	923.076	4.875	0.16%
9	2250	2250	1	0%	2250	2	0%
10	4500	不可能	不可能	不可能	4500	1	0%
11	6250	不可能	不可能	不可能	不可能	不可能	不可能
12	6750	不可能	不可能	不可能	不可能	不可能	不可能

波特率		fPCLK=100MHz			fPCLK=120MHz		
序号	Kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.40004	2604.125	0%	2.4	3125	0%
2	9.6	9.60061	651	0%	9.6	781.25	0%
3	19.2	19.2012	325.5	0%	19.2	390.625	0%
4	57.6	57.6037	108.5	0%	57.61	130.1875	0%
5	115.2	115.207	54.25	0%	115.16	65.125	0%
6	230.4	230.415	27.125	0.01%	230.33	32.5625	0%
7	460.8	460.829	13.5625	0.01%	461.54	16.25	0.16%
8	921.6	925.926	6.75	0.47%	923.08	8.125	0.16%
9	2250	2272.73	2.75	1%	2264.15	3.3125	0.63%
10	4500	4545.45	1.375	1%	4444.44	1.6875	1.23%
11	6250	6250	1	0%	6315.79	1.1875	1.05%
12	7500	不可能	不可能	不可能	7500	1	0%

注意：

1. CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。
2. USART1 使用 PCLK2（最高 120MHz）。USART2 使用 PCLK1（最高 120MHz）。

### 16.3.5 USART接收器容忍时钟的变化

只有当整体的时钟系统地变化小于 USART 异步接收器能够容忍的范围，USART 异步接收器才能正常地工作。影响这些变化的因素有：

- **DTRA:** 由于发送器误差而产生的变化（包括发送器端振荡器的变化）
- **DQUANT:** 接收器端波特率取整所产生的误差
- **DREC:** 接收器端振荡器的变化
- **DTCL:** 由于传输线路产生的变化（通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成）。

需要满足： $DTRA + DQUANT + DREC + DTCL < \text{USART 接收器的容忍度}$

对于正常接收数据，USART 接收器的容忍度等于最大能容忍的变化，它依赖于下述选择：

- 由 USART\_CTRL1 寄存器的 LEN 位定义的 10 或 11 位字符长度
- 是否使用分数波特率产生

表格 16-3 当 DIV\_Decimal=0 时，USART 接收器的容忍度

M 位	认为 NF 是错误	不认为 NF 是错误
0	3.75%	4.375%
1	3.41%	3.97%

表格 16-4 当 DIV\_Decimal ≠ 0 时，USART 接收器的容忍度

M 位	认为 NF 是错误	不认为 NF 是错误
0	3.33%	3.88%
1	3.03%	3.53%

注意：在特殊的情况下，即当收到的帧包含一些在 LEN=0 时，正好是 10 位 (LEN=1 时是 11 位) 的空闲帧，上面 2 个表格中的数据可能会有少许出入。

### 16.3.6 多处理器通信

通过 USART 可以实现多处理器通信（将几个 USART 连在一个网络里）。例如某个 USART 设备可以是主，它的 TX 输出和其他 USART 从设备的 RX 输入相连接；USART 从设备各自的 TX 输出逻辑地与在一起，并且和主设备的 RX 输入相连接。

在多处理器配置中，通常希望只有被寻址的接收器才被激活，来接收随后的数据，这样就可以减少由未被寻址的接收器的参与带来的多余的 USART 服务开销。未被寻址的设备可启用其静默功能置于静默模式。在静默模式里：

- 任何接收状态位都不会被置位。
- 所有接收中断被禁止。
- USART\_CTRL1 寄存器中的 RECMUTE 位被置 1。RECMUTE 可以被硬件自动控制或在某个条件下由软件写入。

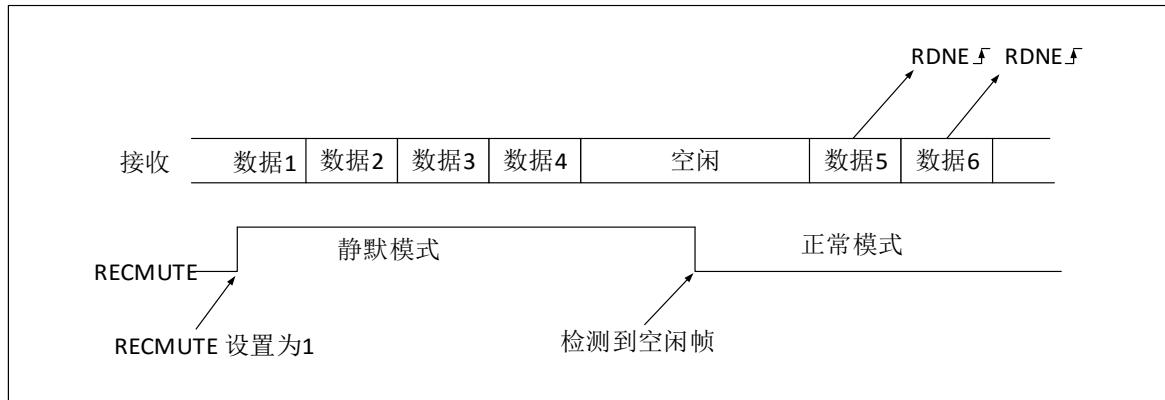
根据 USART\_CTRL1 寄存器中的 WUMODE 位状态，USART 可以用二种方法进入或退出静默模式。

- 如果 WUMODE 位被复位：进行空闲总线检测。
- 如果 WUMODE 位被置位：进行地址标记检测。

#### 16.3.6.1 空闲总线检测 (WUMODE=0)

当 RECMUTE 位被写 1 时，USART 进入静默模式。当检测到一空闲帧时，它被唤醒。然后，RECMUTE 被硬件清零，但是 USART\_STS 寄存器中的 IDLEF 位并不置起。RECMUTE 还可以被软件写 0。下图给出利用空闲总线检测来唤醒和进入静默模式的一个例子。

图表 16-7 利用空闲总线检测的静默模式



### 16.3.6.2 地址标记 (address mark) 检测 (WUMODE=1)

在这个模式里, 如果 MSB 是 1, 该字节被认为是地址, 否则被认为是数据。在一个地址字节中, 目标接收器的地址被放在 4 个 LSB 中。这个 4 位地址被接收器同它自己地址做比较, 接收器的地址被编程在 USART\_CTRL2 寄存器的 ADDR。

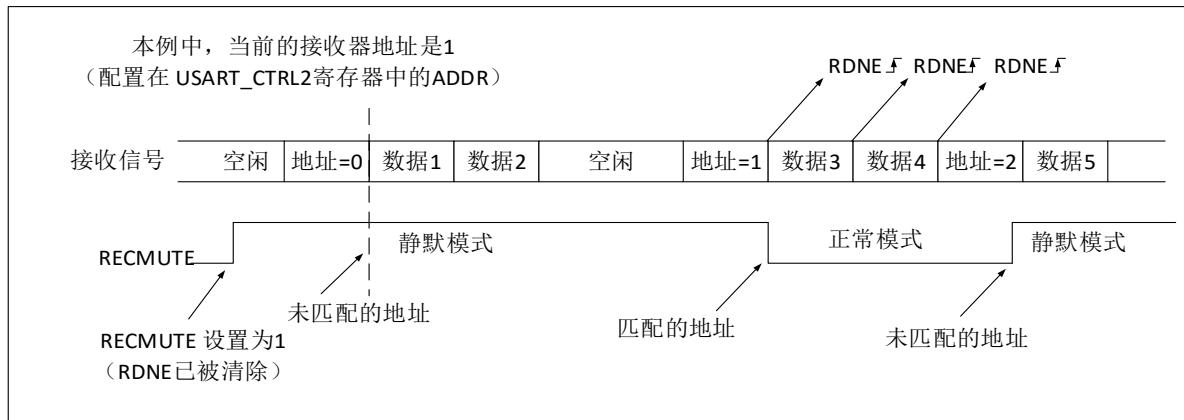
关闭奇偶校验功能时, 当 LEN=0 时, MSB 是 USART\_DT[7], 当 LEN=1 时, MSB 是 USART\_DT[8]; 开启奇偶校验功能时, 当 LEN=0 时, MSB 是 USART\_DT[6], 当 LEN=1 时, MSB 是 USART\_DT[7]。

如果接收到的字节与它的编程地址不匹配时, USART 进入静默模式。此时, 硬件设置 RECMUTE 位。接收该字节既不会设置 RDNE 标志也不会产生中断或发出 DMA 请求, 因为 USART 已经在静默模式。

当接收到的字节与接收器内编程地址匹配时, USART 退出静默模式。然后 RECMUTE 位被清零, 随后的字节被正常接收。收到这个匹配的地址字节时将设置 RDNE 位, 因为 RECMUTE 位已被清零。

当接收缓冲器不包含数据时 (USART\_STS 的 RDNE=0), RECMUTE 位可以被写 0 或 1。否则, 该次写操作被忽略。下图给出利用地址标记检测来唤醒和进入静默模式的例子。

图表 16-8 利用地址标记检测的静默模式



### 16.3.7 校验控制

设置 USART\_CTRL1 寄存器上的 PCEN 位, 可以使能奇偶控制 (发送时生成一个奇偶位, 接收时进行奇偶校验)。根据 LEN 位定义的帧长度, 可能的 USART 帧格式列在下表中。

表格 16-5 帧格式

M 位	PCEN 位	USART 帧
-----	--------	---------

0	0	起始位 8位数据 停止位
0	1	起始位 7位数据 奇偶检验位 停止位
1	0	起始位 9位数据 停止位
1	1	起始位 8位数据 奇偶检验位 停止位

注意：地址标记唤醒设备时，地址的匹配只考虑到数据的 MSB 位，而不用关心校验位。（MSB 是数据位中最后发出的，后面紧跟校验位或者停止位。）

偶校验：校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中 ‘1’ 的个数为偶数。例如：数据 =00110101，有 4 个 ‘1’，如果选择偶校验（在 USART\_CTRL1 中的 PSEL=0），校验位将是 ‘0’。

奇校验：此校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中 ‘1’ 的个数为奇数。例如：数据 =00110101，有 4 个 ‘1’，如果选择奇校验（在 USART\_CTRL1 中的 PSEL=1），校验位将是 ‘1’。

传输模式：如果 USART\_CTRL1 的 PCEN 位被置位，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去（如果选择偶校验偶数个 ‘1’，如果选择奇校验奇数个 ‘1’）。如果奇偶校验失败，USART\_STS 寄存器中的 PERR 标志被置 ‘1’，并且如果 USART\_CTRL1 寄存器的 PERRIEN 在被预先设置的话，中断产生。

### 16.3.8 LIN（局域互联网）模式

LIN 模式是通过设置 USART\_CTRL2 寄存器的 LINEN 位选择。在 LIN 模式下，下列控制寄存器必须保持为 0：

- USART\_CTRL2 寄存器的 CLKEN 位，
- USART\_CTRL3 寄存器的 STOPB[1: 0], SCMEN, HALFSEL 和 IRDAEN

#### 16.3.8.1 LIN发送

16.3.2 节里所描述的同样步骤适用于 LIN 主发送，但和普通 USART 发送有以下区别：

- 清零 LEN 位以配置 8 位字长
- 置位 LINEN 位以进入 LIN 模式。这时，置位 SBRK 将发送 13 位 ‘0’ 作为断开帧；断开帧包括 13 位 ‘0’ 和一位 ‘1’，以允许对下一个开始位的检测。

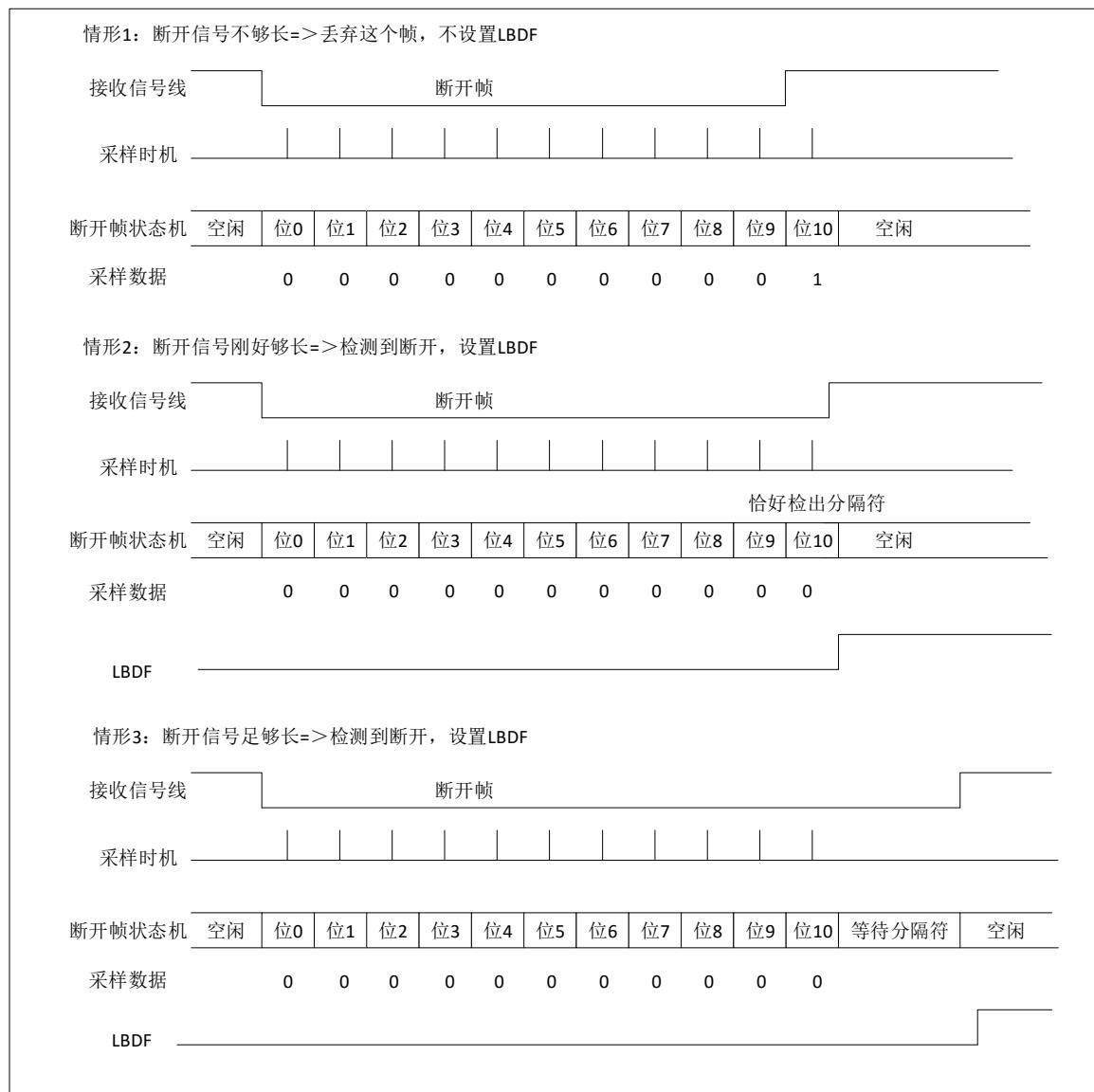
#### 16.3.8.2 LIN接收

当 LIN 模式被使能时，断开帧检测电路被激活。该检测完全独立于 USART 接收器。断开帧只要一出现就能检测到，不管是在总线空闲时还是在接收某数据帧期间（例如，数据帧还未完成，又插入了断开帧的发送）。

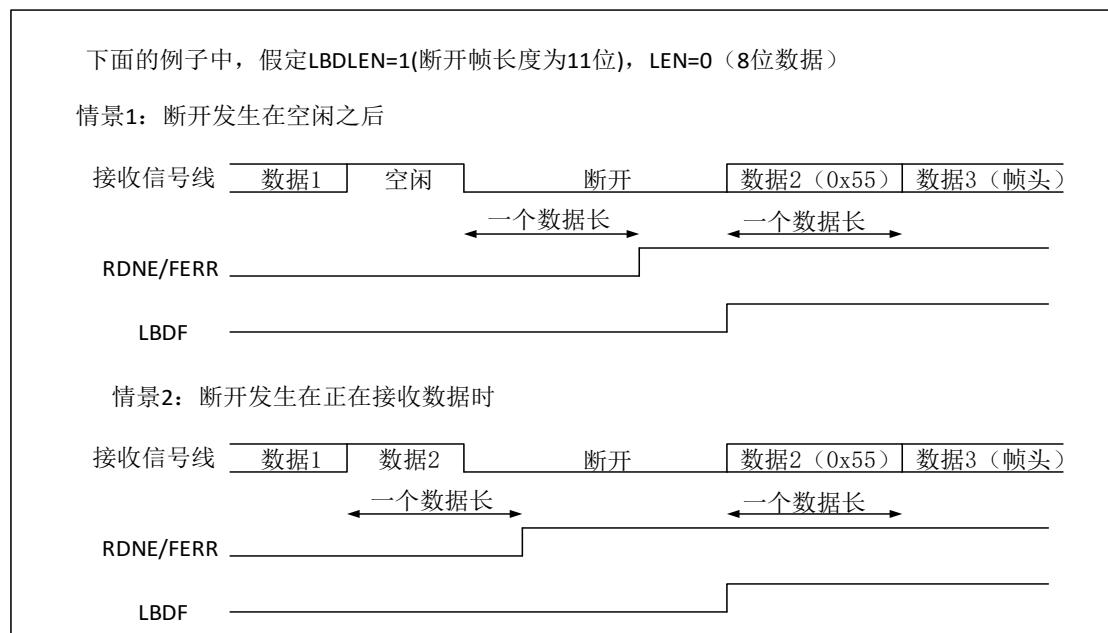
当接收器被激活时（USART\_CTRL1 的 REN=1），电路监测 RX 上的起始信号。监测起始位的方法同检测断开帧或数据帧是一样的。当起始位被检测到后，电路对每个接下来的位，在每个位的第 8, 9, 10 个过采样时钟点上进行采样。如果 10 个（当 USART\_CTRL2 的 LBDLEN=0）或 11 个（当 USART\_CTRL2 的 LBDLEN=1）连续位都是 ‘0’，并且又跟着一个定界符（定界符，也即是 RX 的上升沿），USART\_STS 的 LBDF 标志被设置。如果 LBDIEN 位=1，中断产生。在确认断开帧前，要检查定界符，因为它意味 RX 线已经回到高电平。

如果在第 10 或 11 个采样点之前采样到了 ‘1’，检测电路取消当前检测并重新寻找起始位。如果 LIN 模式被禁止，接收器继续如普通 USART 那样工作，并不检测断开帧。如果 LIN 模式没有被激活（LINEN=0），接收器仍然正常工作于 USART 模式，不会进行断开检测。如果 LIN 模式被激活（LINEN=1），只要一发生帧错误（也就是停止位检测到 ‘0’，这种情况出现在断开帧），接收器就停止，直到断开帧检测电路接收到一个 ‘1’（这种情况发生于断开帧没有完整的发出来），或一个定界符（这种情况发生于已经检测到一个完整的断开帧）。图 16-9 说明了断开帧检测器状态机的行为和断开帧标志的关系。图 16-10 给出了一个断开帧的例子。

图表 16-9 LIN模式下的断开检测（11位断开长度 - 设置了LBDLEN位）



图表 16-10 LIN模式下的断开检测与帧错误的检测



### 16.3.9 USART同步模式

通过在 USART\_CTRL2 寄存器上写 CLKEN 位选择同步模式。在同步模式里，下列控制位必须保持清零状态：

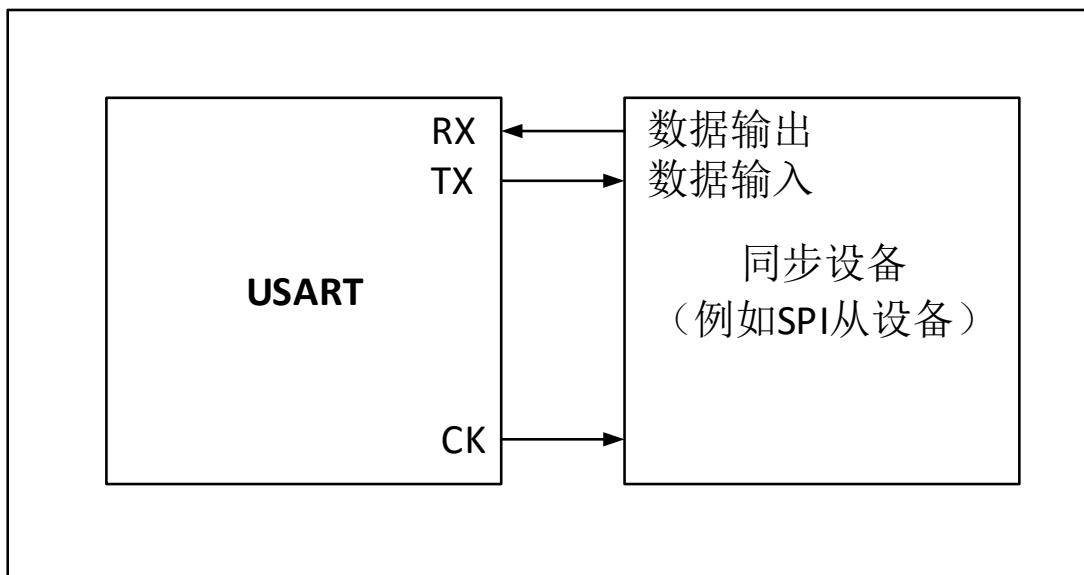
- USART\_CTRL2 寄存器中的 LINEN 位
- USART\_CTRL3 寄存器中的 SCMEN,HALFSEL 和 IRDAEN 位

USART 允许用户以主模式方式控制双向同步串行通信。CK 脚是 USART 发送器时钟的输出。在起始位和停止位期间，CK 脚上没有时钟脉冲。根据 USART\_CTRL2 寄存器中 LBCP 位的状态，决定在最后一个有效数据位期间产生或不产生时钟脉冲。USART\_CTRL2 寄存器的 CLKPOL 位允许用户选择时钟极性，USART\_CTRL2 寄存器上的 CLKPHA 位允许用户选择外部时钟的相位（见图 16-11、图 16-12 和图 16-13）。在总线空闲期间，实际数据到来之前以及发送断开帧的时候，外部 CK 时钟不被激活。同步模式时，USART 发送器和异步模式里工作一模一样。但是因为 CK 是与 TX 同步的（根据 CLKPOL 和 CLKPHA），所以 TX 上的数据是随 CK 同步发出的。

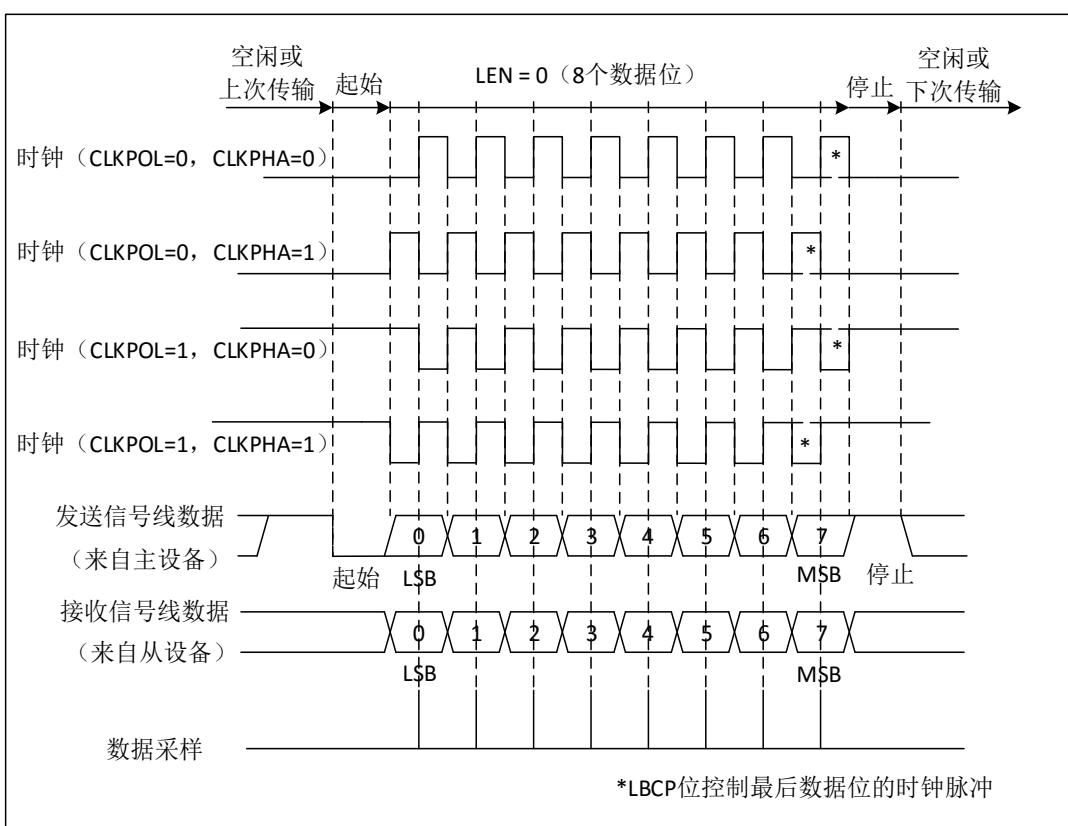
同步模式的 USART 接收器工作方式与异步模式不同。如果 REN=1，数据在 CK 上采样（根据 CLKPOL 和 CLKPHA 决定在上升沿还是下降沿），不需要任何的过采样。但必须考虑建立时间和持续时间（取决于波特率，1/16 位时间）。

- 注意：**
1. CK 脚同 TX 脚一起联合工作。因而，只有在使能了发送器 (*TEN*=1)，并且发送数据时（写入数据至 USART\_DT 寄存器）才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。
  2. LBCP, CLKPOL 和 CLKPHA 位的配置，应该在发送器和接收器都被禁止时；当使能了发送器或接收器时，这些位不能被改变。
  3. 建议在同一条指令中设置 *TEN* 和 *REN*，以减少接收器的建立时间和保持时间。
  4. USART 只支持主模式：它不能用来自其他设备的输入时钟接收或发送数据（CK 永远是输出）。

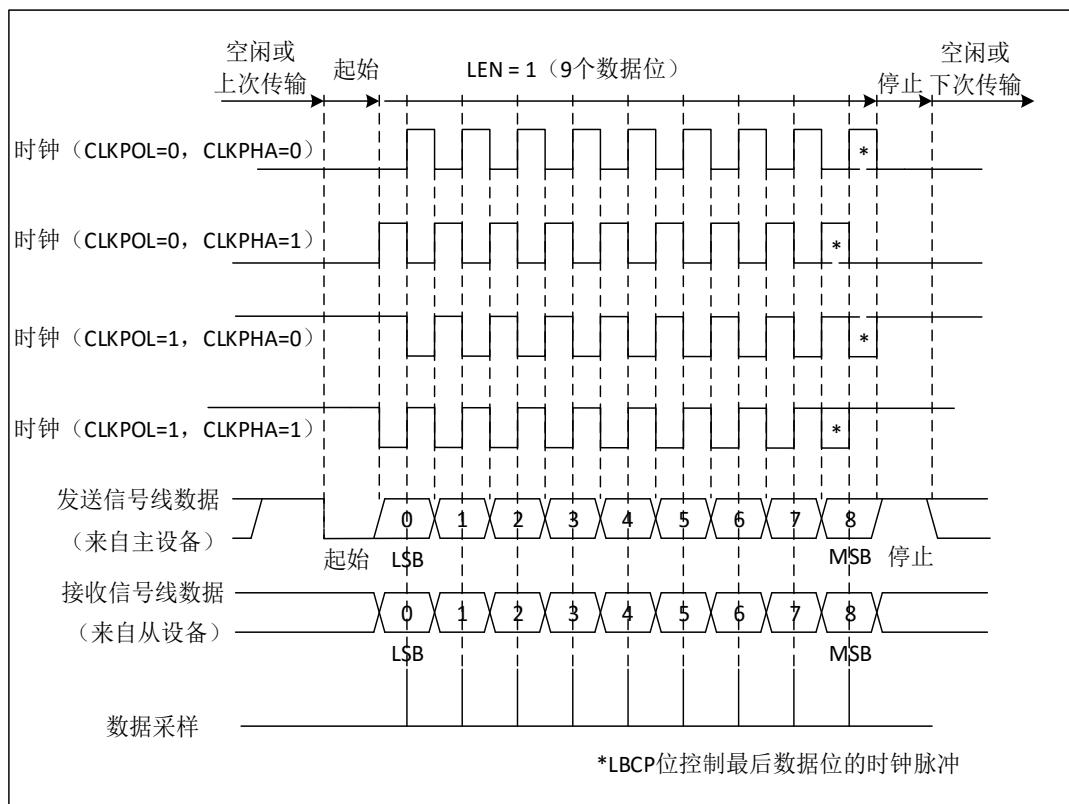
图表 16-11 USART同步传输的例子



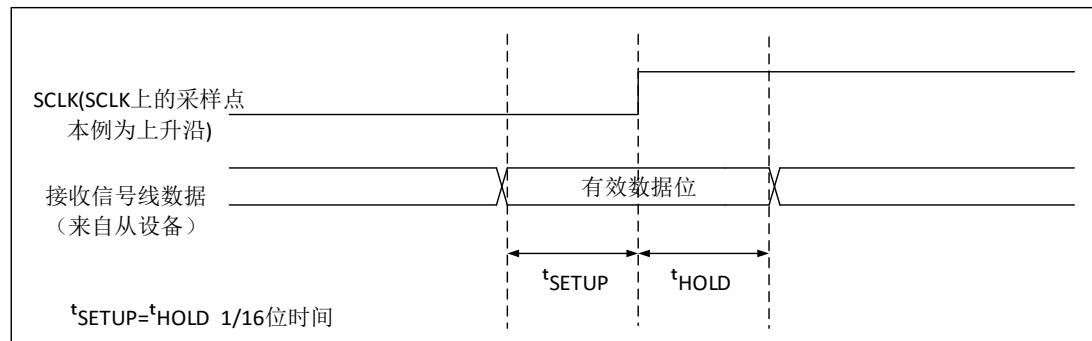
图表 16-12 USART数据时钟时序示例 (LEN=0)



图表 16-13 USART数据时钟时序示例 (LEN=1)



图表 16-14 RX 数据采样/保持时间



注意：在智能卡模式下 CK 的功能不同，有关细节请参考智能卡模式部分。

### 16.3.10 单线半双工通信

单线半双方模式通过设置 USART\_CTRL3 寄存器的 HALFSEL 位选择。在这个模式里，下面的位必须保持清零状态：

- USART\_CTRL2 寄存器的 LINEN 和 CLKEN 位
- USART\_CTRL3 寄存器的 SCMEN 和 IRDAEN 位

USART 可以配置成遵循单线半双工协议。在单线半双工模式下，TX 和 RX 引脚在芯片内部互连。使用控制位“HALFSEL”（USART\_CTRL3 中的 HALFSEL 位）选择半双工和全双工通信。

当 HALFSEL 为‘1’时

- RX 不再被使用。
- 当没有数据传输时，TX 总是被释放。因此，它在空闲状态的或接收状态时表现为一个标准 I/O 口。这就意味该 I/O 在不被 USART 驱动时，必须配置成悬空输入（或开漏的输出高）。除此以

外，通信与普通 USART 模式类似。由软件来管理线上的冲突（例如通过使用一个中央仲裁器）。特别的是，发送从不会被硬件所阻碍，也即如果 USART 处于发送状态时，USART 是无法接收数据的。换句话说，在半双工模式下，发送具有比接收有更高的优先权，发送和接收的冲突应该由软件处理。当 TEN 位被设置时，只要数据一写到数据寄存器上，发送就继续。

### 16.3.11 智能卡

设置 USART\_CTRL3 寄存器的 SCLEN 位选择智能卡模式。在智能卡模式下，下列位必须保持清零：

- USART\_CTRL2 寄存器的 LINEN 位
- USART\_CTRL3 寄存器的 HALFSEL 位和 IRDAEN 位

此外，CLKEN 位可以被设置，以提供时钟给智能卡。

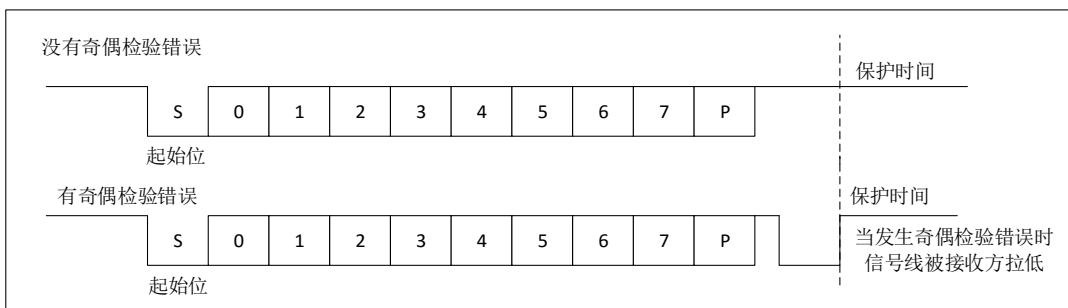
该接口符合 ISO7816-3 标准，支持智能卡异步协议。USART 应该被设置为：

- 8 位数据位加校验位：此时 USART\_CTRL1 寄存器中 LEN=1、PCEN=1
- 发送和接收时为 1.5 个停止位：即 USART\_CTRL2 寄存器的 STOPB=11

**注意：** 也可以在接收时选择 0.5 个停止位，但为了避免在 2 种配置间转换，建议在发送和接收时都使用 1.5 个停止位。

[图 16-15](#) 给出的例子说明了数据线上，在有校验错误和没校验错误两种情况下的信号。

图表 16-15 ISO7816-3 异步协议



当与智能卡相连接时，USART 的 TX 驱动一根智能卡也驱动的双向线；TX 需配置成开漏。

智能卡是一个单线半双工通信协议。

从发送移位寄存器把数据发送出去，要被延时最小 1/2 波特时钟。在普通操作时，一个满的发送移位寄存器将在下一个波特时钟沿开始向外移出数据。在智能卡模式下，此发送被延迟 1/2 波特时钟。

- 如果在接收一个设置为 0.5 或 1.5 个停止位的数据帧期间，检测到一奇偶校验错误，在完成接收该帧后（即停止位结束时），发送线被拉低一个波特时钟周期。这是告诉智能卡发送到 USART 的数据没有被正确地接收到。此 NACK 信号（拉低发送线一个波特时钟周期）在发送端将产生一个帧错误（发送端被配置成 1.5 个停止位）。应用程序可以根据协议处理重新发送数据。如果设置了 NACKEN 控制位，发生校验错误时接收器会给出一个 NACK 信号；否则就不会发送 NACK。
- 若发生奇偶校验错误，发送端的 FERR 在 1.5 个停止位结束时被置起。对于智能卡发送端而言，1.5 个的停止位可以被分成 2 部分：一个是 0.5 个数据位的周期，期间不做任何事情；随后是 1 个数据位的周期的停止位，在这段时间的中点处采样。对于智能卡接收端而言，1.5 个的停止位也可以被分成 2 部分：一个是 0.5 个数据位的周期，判断是否有奇偶校验错误；如果有奇偶校验错误，则在随后的 1 个数据位的周期拉低数据总线；如果没有奇偶校验错误，则在随后的 1 个数据位的周期，释放数据总线（数据总线处于高电平）。
- TRAC 标志的置起可以通过编程保护时间寄存器 (GTVL) 得以延时。在普通操作时，当发送移位寄存器变空并且没有新的发送请求出现时，TRAC 被立即置起。在智能卡模式里，空的发送移位寄存器将触发保护时间计数器开始向上计数，直到保护时间寄存器 (GTVL) 中的值。TRAC 在这段时间（也就是 GTVL 个 bit 的时间宽度）被强制拉低。当保护时间计数器达到保护时间寄存器中的值时，TRAC 才能置起（也即置 1）。

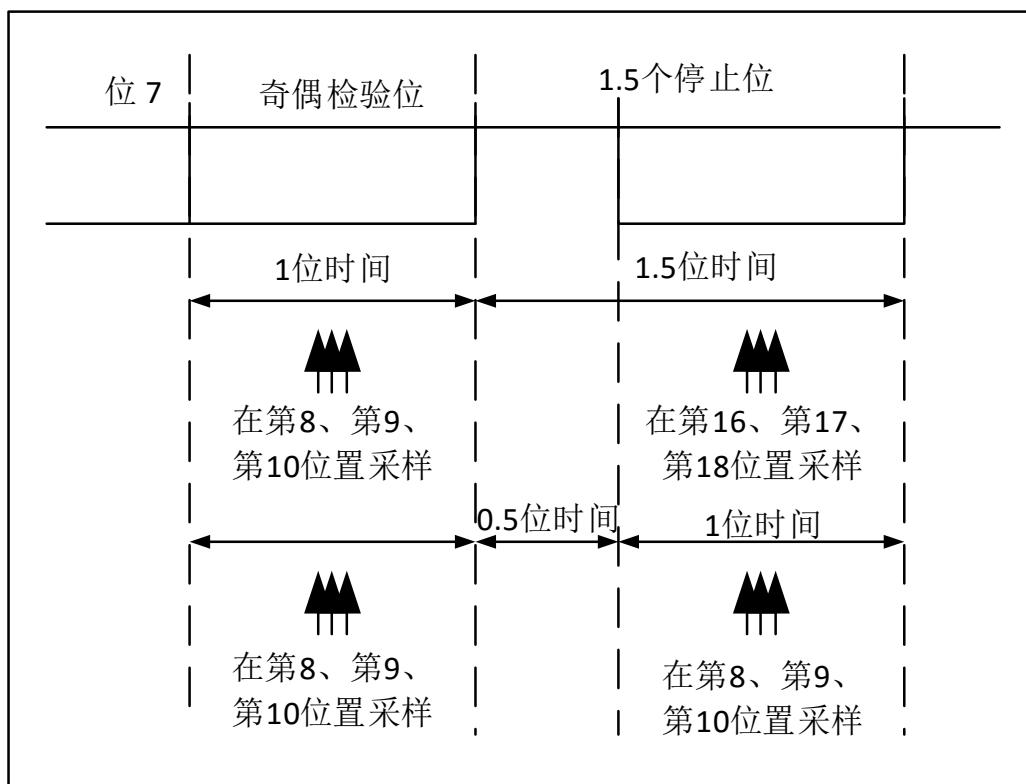
- TRAC 标志的清零不受智能卡模式的影响。
- 如果发送器检测到一个帧错误（收到接收器的 NACK 信号），发送器的接收功能模块不会把 NACK 信号当作起始位检测。根据 ISO 协议，接收到的 NACK 信号的持续时间可以是 1 或 2 波特时钟周期。
- 在接收器这边，如果一个校验错误被检测到，并且 NACK 被发送，接收器不会把 NACK 检测成起始位。

注意： 1. 断开帧在智能卡模式里没有意义。一个带帧错误的  $00h$  数据将被当成数据而不是断开帧。

2. 当来回切换 TEN 位时，没有 IDLE 帧被发送。ISO 协议没有定义 IDLE 帧。

下图详述了 USART 是如何采样 NACK 信号的。在这个例子里，USART 正在发送数据，并且被配置成 1.5 个停止位。为了检查数据的完整性和 NACK 信号，USART 的接收功能块被激活。

图表 16-16 使用 1.5 停止位检测奇偶检验错



USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式里，CK 不和通信直接关联，而是先通过一个 5 位预分频器简单地用内部的外设输入时钟来驱动智能卡的时钟。分频率在预分频寄存器 GTVAL 中配置。CK 频率可以从  $f_{CK}/2$  到  $f_{CK}/62$ ，这里的  $f_{CK}$  是外设输入时钟。

### 16.3.12 IrDA SIR ENDEC 功能模块

通过设置 USART\_CTRL3 寄存器的 IRDAEN 位选择 IrDA 模式。在 IrDA 模式里，下列位必须保持清零：

- USART\_CTRL2 寄存器的 LINEN,STOPB 和 CLKEN 位
- USART\_CTRL3 寄存器的 SCMEN 和 HALFSEL 位。

IrDA SIR 物理层规定使用反相归零调制方案 (RZI)，该方案用一个红外光脉冲代表逻辑‘0’（见图 16-17）。SIR 发送编码器对从 USART 输出的 NRZ (非归零) 比特流进行调制。输出脉冲流被传送到一个外部输出驱动器和红外 LED。USART 为 SIRENDEC 最高只支持到 115.2Kbps 速率。在普通模式里，脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到

USART。在空闲状态里，解码器输入通常是高（标记状态 marking state）。发送编码器输出的极性和解码器的输入相反。当解码器输入低时，检测到一个起始位。

- IrDA 是一个半双工通信协议。如果发送器忙（也就是 USART 正在送数据给 IrDA 编码器），IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙（也就是 USART 正在接收从 IrDA 解码器来的解码数据），从 USART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- SIR 发送逻辑把' 0' 作为高脉冲发送，把' 1' 作为低电平发送。脉冲的宽度规定为普通模式时位周期的 3/16 (见图 16-18)。
- SIR 接收逻辑把高电平状态解释为' 1'，把低脉冲解释为' 0'。
- 发送编码器输出与解码器输入有着相反的极性。当空闲时，SIR 输出处于低状态。
- SIR 解码器把 IrDA 兼容的接收信号转变成给 USART 的比特流。
- IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收器端的尖峰脉冲检测逻辑滤除宽度小于 2 个 DIV 周期的脉冲 (DIV 是在 IrDA 低功耗波特率寄存器 GTP 中编程的预分频值)。宽度小于 1 个 DIV 周期的脉冲一定被滤除掉，但是那些宽度大于 1 个而小于 2 个 DIV 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。当 DIV=0 时，IrDA 编码器/解码器不工作。
- 接收器可以与一低功耗发送器通信。
- 在 IrDA 模式里，USART\_CTRL2 寄存器上的 STOPB 位必须配置成 1 个停止位。

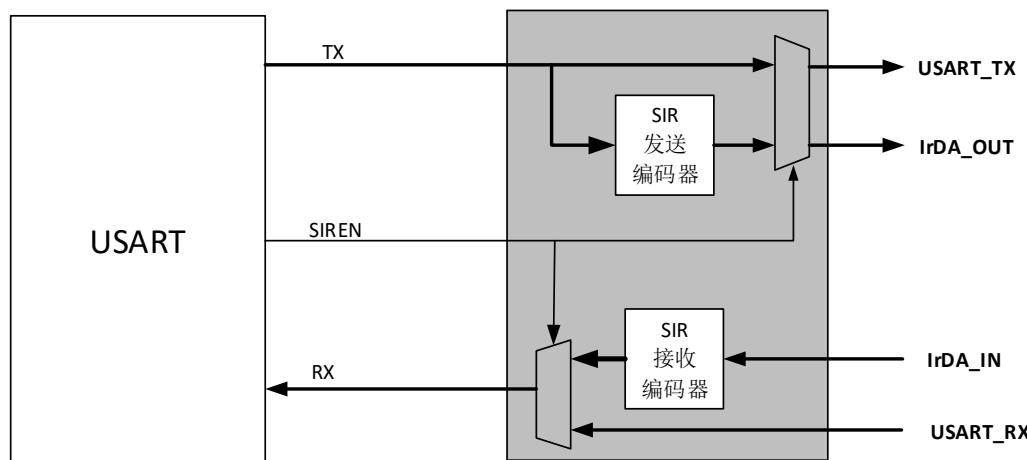
### IrDA 低功耗模式

发送器在低功耗模式，脉冲宽度不再持续 3/16 个位周期。取而代之，脉冲的宽度是低功耗波特率的 3 倍，它最小可以是 1.42MHz。通常这个值是 1.8432MHz ( $1.42\text{MHz} < \text{DIV} < 2.12\text{MHz}$ )。一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。低功耗模式的接收器类似于普通模式的接收器。为了滤除尖峰干扰脉冲，USART 应该滤除宽度短于 1 个 DIV 的脉冲。只有持续时间大于 2 个周期的 IrDA 低功耗波特率时钟 (GTP 中的 DIV) 的低电平信号才被接受为有效的信号。

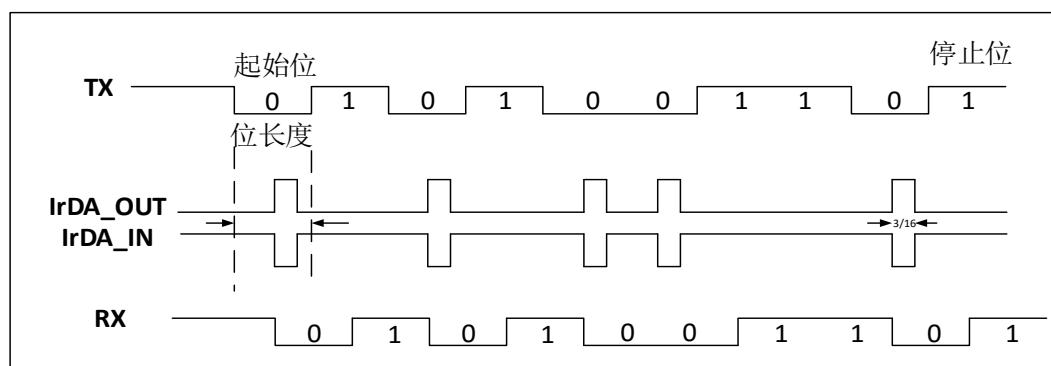
注意：1. 宽度小于 2 个，但大于 1 个 DIV 周期的脉冲可能会也可能不会被滤除。

2. 接收器的建立时间应该由软件管理。IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时 (IrDA 是一个半双工协议)。

图表 16-17 IrDA SIR ENDEC框图



图表 16-18 IrDA数据调制（3/16） - 普通模式



### 16.3.13 利用DMA连续通信

USART 可以利用 DMA 连续通信。RX 缓冲器和 TX 缓冲器的 DMA 请求是分别产生的。

**注意：**参考产品技术说明以确定是否可用 DMA 控制器。如果所用产品无 DMA 功能，应按 [16.3.2 节](#)或[16.3.3 节](#)里所描述的方法使用 USART。在 USART\_STS 寄存器里，可以清零 TDE/RDNE 标志来实现连续通信。

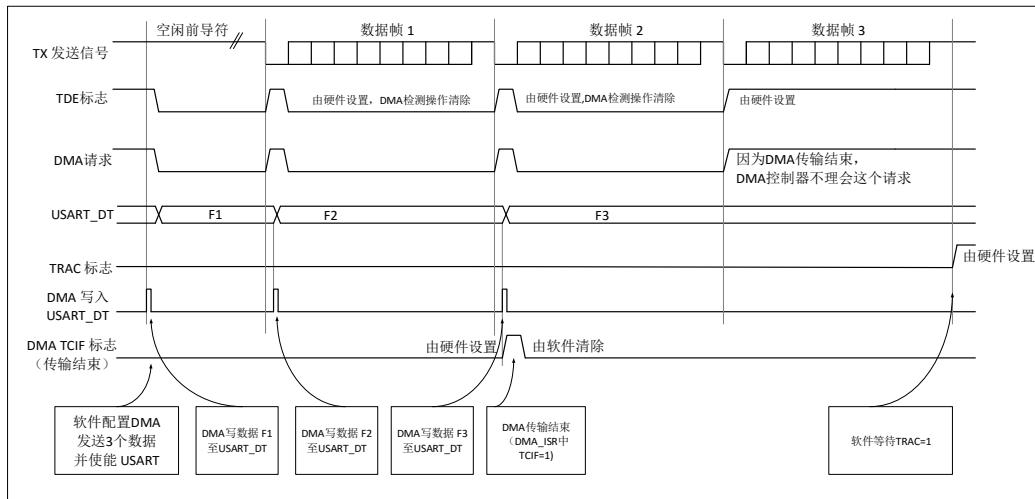
#### 16.3.13.1 利用DMA发送

使用 DMA 进行发送，可以通过设置 USART\_CTRL3 寄存器上的 DMATEN 位激活。当 TDE 位被置为‘1’时，DMA 就从指定的 SRAM 区传送数据到 USART\_DT 寄存器。为 USART 的发送分配一个 DMA 通道的步骤如下（x 表示通道号）：

1. 在 DMA 控制寄存器上将 USART\_DT 寄存器地址配置成 DMA 传输的目的地址。在每个 TDE 事件后，数据将被传送到这个地址。
2. 在 DMA 控制寄存器上将存储器地址配置成 DMA 传输的源地址。在每个 TDE 事件后，将从此存储器区读出数据并传送到 USART\_DT 寄存器。
3. 在 DMA 控制寄存器中配置要传输的总的字节数。
4. 在 DMA 寄存器上配置通道优先级。
5. 根据应用程序的要求，配置在传输完成一半还是全部完成时产生 DMA 中断。
6. 在 DMA 寄存器上激活该通道。当传输完成 DMA 控制器指定的数据量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。在发送模式下，当 DMA 传输完所有要发送的数据时，DMA 控制器设置 DMA\_ISTS 寄存器的 TCIF 标志；监视 USART\_STS 寄存器的 TRAC 标志可以确认 USART 通信是否结束，这样可以在关闭 USART 或进入停机模式之前避免破坏最后一次传输的

数据；软件需要先等待 **TDE=1**，再等待 **TRAC=1**。

图表 16-19 利用 DMA 发送

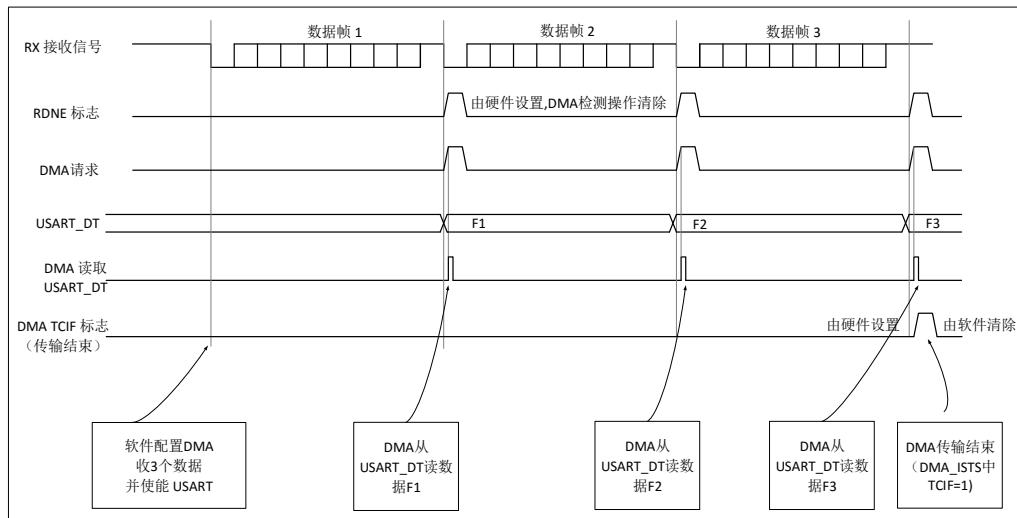


### 16.3.13.2 利用 DMA 接收

可以通过设置 **USART\_CTRL3** 寄存器的 **DMAREN** 位激活使用 DMA 进行接收，每次接收到一个字节，DMA 控制器就就把数据从 **USART\_DT** 寄存器传送到指定的 **SRAM** 区(参考 DMA 相关说明)。为 **USART** 的接收分配一个 DMA 通道的步骤如下 (x 表示通道号)：

1. 通过 DMA 控制寄存器把 **USART\_DT** 寄存器地址配置成传输的源地址。在每个 **RDNE** 事件后，将从此地址读出数据并传输到存储器。
2. 通过 DMA 控制寄存器把存储器地址配置成传输的目的地址。在每个 **RDNE** 事件后，数据将从 **USART\_DT** 传输到此存储器区。
3. 在 DMA 控制寄存器中配置要传输的总的字节数。
4. 在 DMA 寄存器上配置通道优先级。
5. 根据应用程序的要求配置在传输完成一半还是全部完成时产生 DMA 中断。
6. 在 DMA 控制寄存器上激活该通道。当接收完成 DMA 控制器指定的传输量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

图表 16-20 利用 DMA 接收



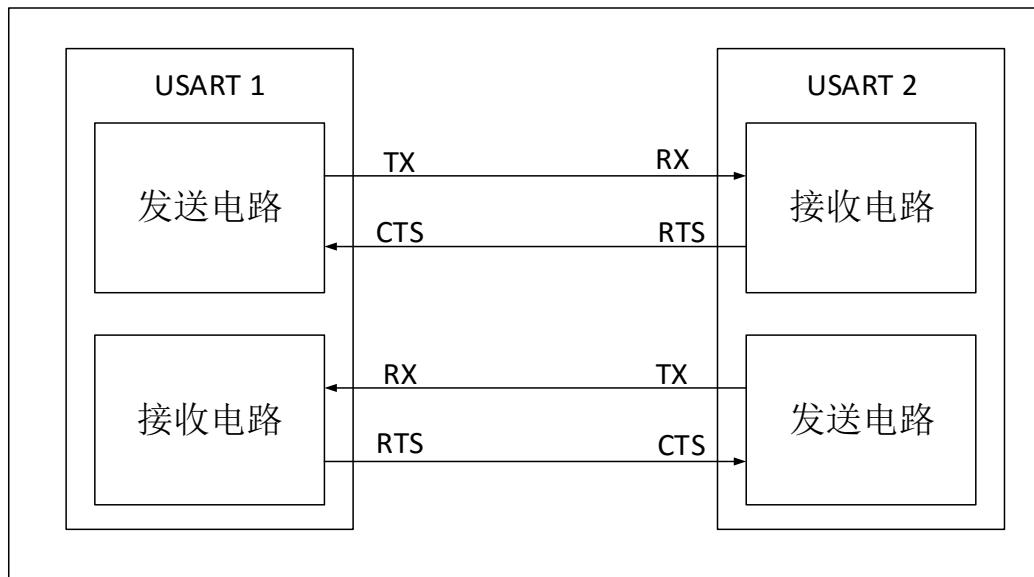
### 16.3.13.3 多缓冲器通信中的错误标志和中断产生

在多缓冲器通信的情况下，通信期间如果发生任何错误，在当前字节传输后将置起错误标志。如果中断使能位被设置，将产生中断。在单个字节接收的情况下，和 RDNE 一起被置起的帧错误、溢出错误和噪音标志，有单独的错误标志中断使能位；如果设置了，会在当前字节传输结束后，产生中断。

### 16.3.14 硬件流控制

利用 CTS 输入和 RTS 输出可以控制 2 个设备间的串行数据流。下图表明在这个模式里如何连接 2 个设备。

图表 16-21 两个USART间的硬件流控制

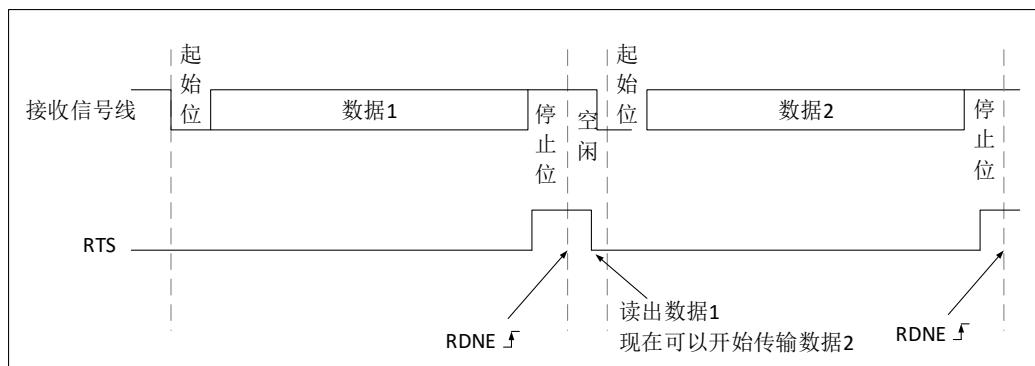


通过将 UASRT\_CTRL3 中的 RTSEN 和 CTSEN 置位，可以分别独立地使能 RTS 和 CTS 流控制。

#### 16.3.14.1 RTS流控制

如果 RTS 流控制被使能 (RTSEN=1)，只要 USART 接收器准备好接收新的数据，RTS 就变成有效（下拉为低电平）。当接收寄存器内有数据到达时（在每个 stop 位开始时），RTS 被置位，由此表明希望在当前帧结束时停止数据传输；当 RDNE 置起后，读 DR 会使得 RDNE 被清零，从而 RTS 也被清零，表示接收机可以接收下一个数据。下图是一个启用 RTS 流控制的通信的例子。

图表 16-22 RTS流控制

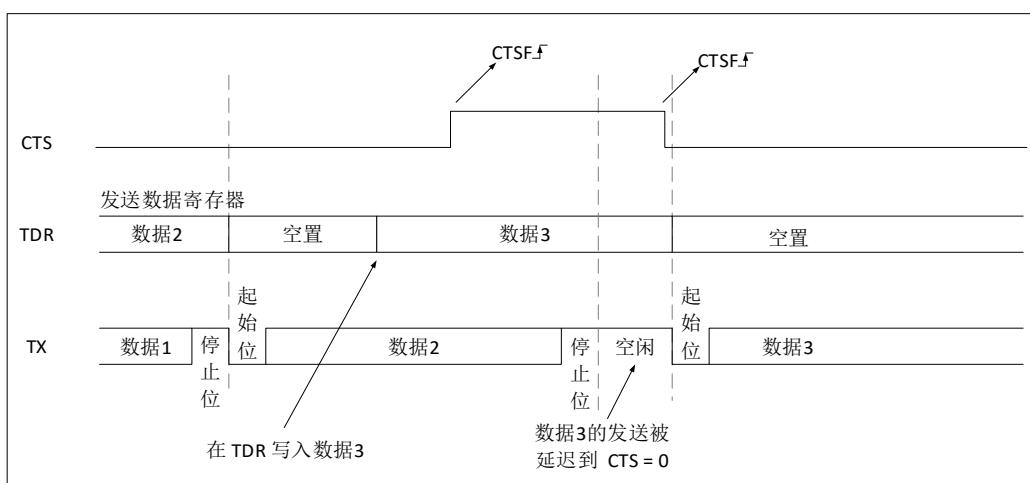


### 16.3.14.2 CTS流控制

如果 CTS 流控制被使能 (CTSEN=1)，发送器在发送下一帧前检查 CTS 输入。如果 CTS 有效(也即 CTS 为低电平)，则下一个数据被发送 (假设那个数据是准备发送的，也就是 TDE=0)；若 CTS 在传输期间被变成无效(也即 CTS 为高电平)，当前的传输完成后停止发送。

当 CTSEN=1 时，只要 CTS 电平发生变化，硬件就自动设置 CTSF 状态位。它表明接收器是否准备好进行通信。如果设置了 USART\_CTL3 寄存器的 CTSIEN 位，则产生中断。下图是一个启用 CTS 流控制通信的例子。

图表 16-23 CTS流控制



## 16.4 USART中断请求

表格 16-6 USART中断请求

中断事件	事件标志	使能位
发送数据寄存器空	TDE	TDEIEN
CTS 标志	CTSF	CTSIEN
发送完成	TRAC	TRACIEN
接收数据就绪可读	RDNE	RDNEIEN
检测到数据溢出	ORERR	
检测到空闲线路	IDLEF	IDLEIEN
奇偶检验错	PERR	PERRIEN
断开标志	LBDF	LBDIEN
噪声标志，多缓冲通信中的溢出错误和帧错误	NERR 或 ORERR 或 FERR	ERRIEN <sup>(1)</sup>

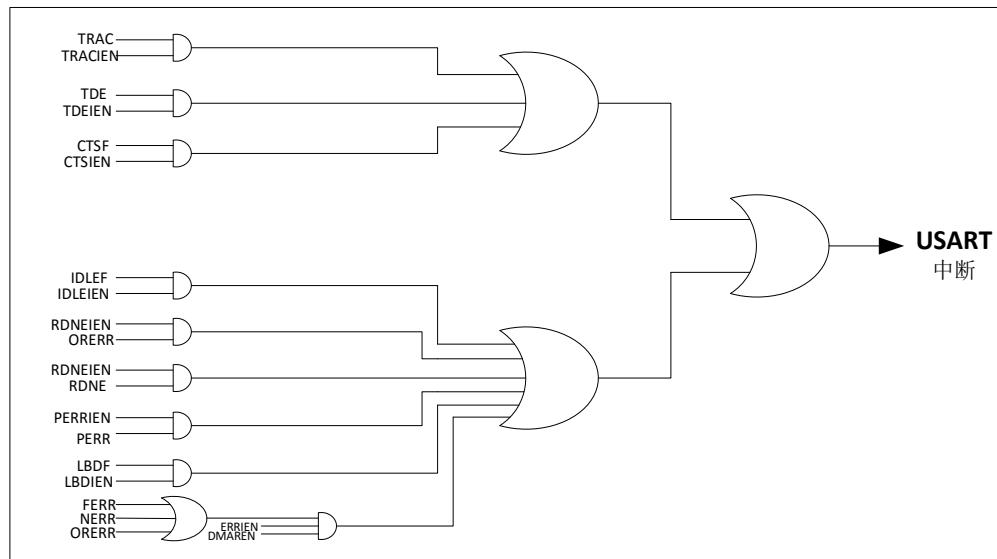
注意：仅当使用 DMA 接收数据时，才使用这个标志位。

USART 的各种中断事件被连接到同一个中断向量（见下图），有以下各种中断事件：

- 发送期间：发送完成、CTS 变化、发送数据寄存器空。
- 接收期间：空闲总线检测、溢出错误、接收数据寄存器非空、校验错误、LIN 断开帧检测、噪音标志（仅在多缓冲器通信）和帧错误（仅在多缓冲器通信）。

如果设置了对应的使能控制位，这些事件就可以产生各自的中断。

图表 16-24 USART 中断映像图



## 16.5 USART 模式配置

表格 16-7 USART 模式设置 (1)

USART 模式	USART1	USART2
异步模式	X	X
硬件流控制	X	X
多缓存通讯 (DMA)	X	X
多处理器通讯	X	X
同步	X	X
智能卡	X	X
半双工 (单线模式)	X	X
IrDA	X	X
LIN	X	X

注：X=支持，NA=不支持该应用。

## 16.6 USART 寄存器描述

有关寄存器描述里所使用的缩写，请参考“寄存器描述表中使用的缩写列表”。

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

### 16.6.1 USART 寄存器地址映象

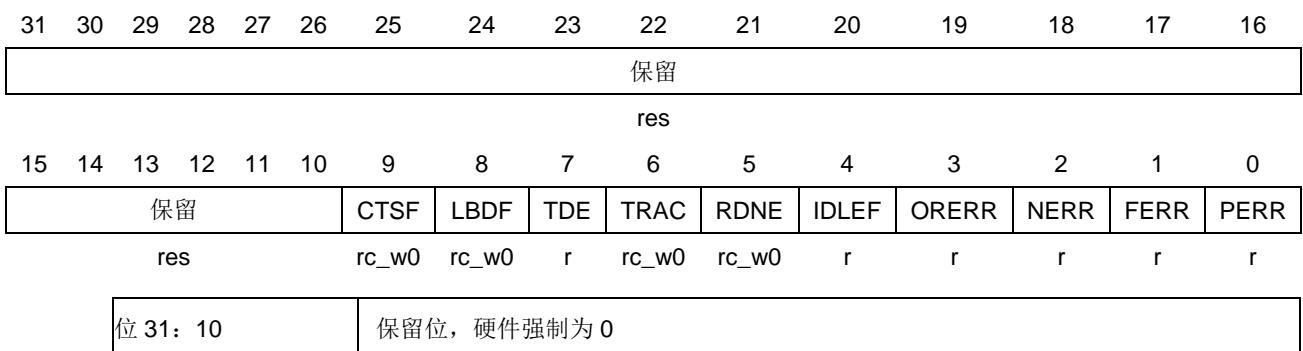
表格 16-8 USART 寄存器地址映像和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
0x00	USART_STS	保留																									CTSF	LBDF	TDE	TRAC	RDNE	IDLEF				
	0x00C0																															PERR	0			
0x04	USART_DT	保留																									DT[8: 0]									
	0x0000																										0	0	0	0	0	0				
0x08	USART_BAUDR	保留												DIV_Integer[11: 0]												DIV_Decimal[3: 0]										
	0x0000													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x0C	USART_CTRL1	保留												UEN	LEN	WUMODE	PCEN	PSEL	PERRN	TDEIEN	TRACIEN	RDNEIEN	IDLEIEN	TEN	REN	RECUME	SBRK	7	6	5	4	3	2	1	0	
	0x0000													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x10	USART_CTRL2	保留												SWAP	LINEN	STOPB[1: 0]	CLKEN	CLKPOL	CLKRHA	LBCP	保留	LBDIEN	保留	保留	0	0	0	0	0	0	0	0	0	0	0	
	0x0000													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x14	USART_CTRL3	保留												CTSien	CTSEN	RTSEN	DMA1EN	DMA1REN	SCMEN	NACKEN	HALFSEL	IRDALP	IRDAEN	ERRIEN	0	0	0	0	0	0	0	0	0	0	0	
	0x0000													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x18	USART_GTP	保留												GTVAL[7: 0]								DIV[7: 0]														
	0x0000													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			

## 16.6.2 状态寄存器 (USART\_STS)

地址偏移: 0x00

复位值: 0x00C0



位 9	<b>CTSF:</b> CTSF 标志 (CTS flag) 如果设置了 CTSEN 位, 当 CTS 输入变化状态时, 该位被硬件置高。由软件将其清零。如果 USART_CTRL3 中的 CTSIEN 为'1', 则产生中断。 0: CTS 状态线上没有变化; 1: CTS 状态线上发生变化。
位 8	<b>LBDF:</b> LIN 断开检测标志 (LIN break detection flag) 当探测到 LIN 断开时, 该位由硬件置'1', 由软件清'0' (向该位写 0)。如果 USART_CTRL3 中的 LBDIEN=1, 则产生中断。 0: 没有检测到 LIN 断开; 1: 检测到 LIN 断开。 注意: 若 LBDIEN=1, 当 LBDF 为'1'时要产生中断。
位 7	<b>TDE:</b> 发送数据寄存器空 (Transmit data register empty) 当 TDR 寄存器中的数据被硬件转移到移位寄存器的时候, 该位被硬件置位。如果 USART_CTRL1 寄存器中的 TDEIEN 为 1, 则产生中断。对 USART_DT 的写操作, 将清零该位。 0: 数据还没有被转移到移位寄存器; 1: 数据已经被转移到移位寄存器。 注意: 单缓冲器传输中使用该位。
位 6	<b>TRAC:</b> 发送完成 (Transmission complete) 当包含有数据的一帧发送完成后, 并且 TDE=1 时, 由硬件将该位置'1'。如果 USART_CTRL1 中的 TRACIEN 为'1', 则产生中断。由软件序列清除该位 (先读 USART_STS, 然后写入 USART_DT)。TRAC 位也可以通过写入'0'来清除, 只有在多缓存通讯中才推荐这种清除程序。 0: 发送还未完成; 1: 发送完成。
位 5	<b>RDNE:</b> 读数据寄存器非空 (Read data register not empty) 当 RDR 移位寄存器中的数据被转移到 USART_DT 寄存器中, 该位被硬件置位。如果 USART_CTRL1 寄存器中的 RDNEIEN 为 1, 则产生中断。对 USART_DT 的读操作可以将该位清零。RDNE 位也可以通过写入 0 来清除, 只有在多缓存通讯中才推荐这种清除程序。 0: 数据没有收到; 1: 收到数据, 可以读出。
位 4	<b>IDLEF:</b> 监测到总线空闲 (IDLE line detected) 当检测到总线空闲 (空闲帧) 时, 该位被硬件置位。如果 USART_CTRL1 中的 IDLEIEN 为'1', 则产生中断。由软件序列清除该位 (先读 USART_STS, 然后读 USART_DT)。 0: 没有检测到空闲总线, 也即没有检测到空闲帧; 1: 检测到空闲总线。 注意: IDLEF 位不会再次被置高直到 RDNE 位被置起 (即又检测到一次空闲总线)
位 3	<b>ORERR:</b> 过载错误 (Overrun error) 当 RDNE 仍然是'1'的时候, 当前被接收在移位寄存器中的数据, 需要传送至 RDR 寄存器时, 硬件将该位置位。如果 USART_CTRL1 中的 RDNEIEN 为'1'的话, 则产生中断。由软件序列将其清零 (先读 USART_STS, 然后读 USART_DT)。 0: 没有过载错误; 1: 检测到过载错误。 注意: 该位被置位时, RDR 寄存器中的值不会丢失, 但是移位寄存器中的数据会被覆盖。如果设置了 ERRIEN 位, 在多缓冲器通信模式下, ORERR 标志置位会产生中断的。

位 2	<b>NERR:</b> 噪声错误标志 (Noise error flag) 在接收到的帧检测到噪音时，由硬件对该位置位。由软件序列对其清零（先读 USART_STS，再读 USART_DT）。 0: 没有检测到噪声； 1: 检测到噪音。 注意：该位不会产生中断，因为它和 RDNE 一起出现，硬件会在设置 RDNE 标志时产生中断。在多缓冲区通信模式下，如果设置了 ERRIEN 位，则设置 NERR 标志时会产生中断。
位 1	<b>FERR:</b> 帧错误 (Framing error) 当检测到同步错位，过多的噪声或者检测到断开符，该位被硬件置位。由软件序列将其清零（先读 USART_STS，再读 USART_DT）。 0: 没有检测到帧错误； 1: 检测到帧错误或者断开帧 (break frame)。 注意：该位不会产生中断，因为它和 RDNE 一起出现，硬件会在设置 RDNE 标志时产生中断。如果当前传输的数据既产生了帧错误，又产生了过载错误，硬件还是会继续该数据的传输，并且只设置 ORERR 标志位。 在多缓冲区通信模式下，如果设置了 ERRIEN 位，则设置 FERR 标志时会产生中断。
位 0	<b>PERR:</b> 校验错误 (Parity error) 在接收模式下，如果出现奇偶校验错误，硬件对该位置位。由软件序列对其清零（依次读 USART_STS 和 USART_DT）。在清除 PERR 位前，软件必须等待 RDNE 标志位被置'1'。如果 USART_CTRL1 中的 PERRIEN 为'1'，则产生中断。 0: 没有奇偶校验错误； 1: 奇偶校验错误。

### 16.6.3 数据寄存器 (USART\_DT)

地址偏移: 0x04

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16						
保留																					
res																					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
保留										DT[8: 0]											
res																					
位 31: 9		保留位，硬件强制为 0																			
位 8: 0		<b>DT[8: 0]:</b> 数据值 (Data value) 包含了发送或接收的数据。由于它是由两个寄存器组成的，一个给发送用 (TDR)，一个给接收用 (RDR)，该寄存器兼具读和写的功能。TDR 寄存器提供了内部总线和输出移位寄存器之间的并行接口 (参见图 16-1)。RDR 寄存器提供了输入移位寄存器和内部总线之间的并行接口。 当使能校验位 (USART_CTRL1 中 PCEN 位被置位) 进行发送时，写到 MSB 的值 (根据数据的长度不同，MSB 是第 7 位或者第 8 位) 会被校验位取代。当使能校验位进行接收时，读到的 MSB 位是接收到的校验位。																			

### 16.6.4 波特比率寄存器 (USART\_BAUDR)

地址偏移: 0x08

复位值: 0x0000

注意：如果 TE 或 RE 被分别禁止，波特计数器停止计数。

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

### 16.6.5 控制寄存器1 (USART\_CTRL1)

地址偏移: 0x0C

复位值: 0x0000

位 9	<b>PSEL:</b> 校验选择 (Parity selection) 当校验控制使能后，该位用来选择是采用偶校验还是奇校验。软件对它置'1'或清'0'。当前字节传输完成后，该选择生效。 0: 偶校验； 1: 奇校验。
位 8	<b>PERRIEN:</b> PERR 中断使能 (PERR interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断； 1: 当 USART_STS 中的 PERR 为'1'时，产生 USART 中断。
位 7	<b>TDEIEN:</b> 发送缓冲区空中断使能 (TDE interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断； 1: 当 USART_STS 中的 TDE 为'1'时，产生 USART 中断。
位 6	<b>TRACIEN:</b> 发送完成中断使能 (TRAC interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断； 1: 当 USART_STS 中的 TRAC 为'1'时，产生 USART 中断。
位 5	<b>RDNEIEN:</b> 接收缓冲区非空中断使能 (RDNE interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断； 1: 当 USART_SR 中的 RDNE 或者 ORERR 为'1'时，产生 USART 中断。
位 4	<b>IDLEIEN:</b> IDLE 中断使能 (IDLE interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断； 1: 当 USART_STS 中的 IDLEF 为'1'时，产生 USART 中断。
位 3	<b>TEN:</b> 发送使能 (Transmitter enable) 该位使能发送器。该位由软件设置或清除。 0: 禁止发送； 1: 使能发送。  注意: 1. 在数据传输过程中，除了在智能卡模式下，如果 TEN 位上有个 0 脉冲（即设置为'0'之后再设置为'1'），会在当前数据字传输完成后，发送一个“前导符”（空闲总线，也即空闲帧）。 2. 当 TEN 被设置后，在真正发送开始之前，有一个比特时间的延迟。
位 2	<b>REN:</b> 接收使能 (Receiver enable) 该位由软件设置或清除。 0: 禁止接收； 1: 使能接收，并开始搜寻 RX 引脚上的起始位。
位 1	<b>RECMUTE:</b> 接收唤醒 (Receiver wakeup) 该位用来决定是否把 USART 置于静默模式。该位由软件设置或清除。当唤醒序列到来时，硬件也会将其清零。 0: 接收器处于普通工作模式； 1: 接收器处于静默模式。  注意: 1. 在把 USART 置于静默模式（设置 RECMUTE 位）之前，USART 要先接收一个数据。否则在静默模式下，不能被空闲总线检测唤醒。 2. 当配置成地址标记检测唤醒 (WUMODE 位=1)，在 RDNE 位被置位时，不能用软件修改 RECMUTE 位。
位 0	<b>SBRK:</b> 发送断开帧 (Send break) 使用该位来发送断开字符。该位可以由软件设置或清除。操作过程应该是软件设置该位，然后在断开帧的停止位时，由硬件将该位复位。 0: 没有发送断开字符； 1: 将要发送断开字符。

### 16.6.6 控制寄存器2 (USART\_CTRL2)

地址偏移: 0x10

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWAP	LIN EN	STOP B[1: 0]	CLK EN	CLK POL	CLKP HA	LBCP	保留	LBD IEN	LBD LEN	保留	ADDR[3: 0]				
rw	rw	rw	rw	rw	rw	rw	rw	res	rw	rw	res	rw	rw	rw	rw

位 31: 16	保留位, 硬件强制为 0。
位 15	<b>SWAP:</b> 交换 TX/RX 管脚 0: TX 和 RX 管脚功能不被交换 1: TX 和 RX 管脚功能被交换 当 USART 被使能(UEN=1)时, 该位域不能改写
位 14	<b>LINEN:</b> LIN 模式使能 (LIN mode enable) 该位由软件设置或清除。 0: 禁止 LIN 模式; 1: 使能 LIN 模式。在 LIN 模式下, 可以用 USART_CTRL1 寄存器中的 SBRK 位发送 LIN 同步断开符 (也即 13 位低电平), 以及检测 LIN 同步断开符。
位 13: 12	<b>STOPB:</b> 停止位 (STOP bits) 这 2 位用来设置停止位的位数 00: 1 个停止位; 01: 0.5 个停止位; 10: 2 个停止位; 11: 1.5 个停止位;
位 11	<b>CLKEN:</b> 时钟使能 (Clock enable) 该位用来使能 CK 引脚 0: 禁止 CK 引脚; 1: 使能 CK 引脚。
位 10	<b>CLKPOL:</b> 时钟极性 (Clock polarity) 在同步模式下, 可以用该位选择 SLCK 引脚上时钟输出的极性。和 CLKPHA 位一起配合来产生需要的时钟/数据的采样关系 0: 总线空闲时 CK 引脚上保持低电平; 1: 总线空闲时 CK 引脚上保持高电平。
位 9	<b>CLKPHA:</b> 时钟相位 (Clock phase) 在同步模式下, 可以用该位选择 SLCK 引脚上时钟输出的相位。和 CLKPOL 位一起配合来产生需要的时钟/数据的采样关系 (参见 <a href="#">图 16-12</a> 和 <a href="#">图 16-13</a> )。 0: 在时钟的第一个边沿进行数据捕获; 1: 在时钟的第二个边沿进行数据捕获。
位 8	<b>LBCP:</b> 最后一位时钟脉冲 (Last bit clock pulse) 在同步模式下, 使用该位来控制是否在 CK 引脚上输出最后发送的那个数据字节 (MSB) 对应的时钟脉冲 0: 最后一位数据的时钟脉冲不从 CK 输出; 1: 最后一位数据的时钟脉冲会从 CK 输出。 注意: 最后一个数据位就是第 8 或者第 9 个发送的位 (根据 USART_CTRL1 寄存器中的 LEN 位所定义的 8 或者 9 位数据帧格式)。
位 7	保留位, 硬件强制为 0

位 6	<b>LBDIEN:</b> LIN 断开符检测中断使能 (LIN break detection interrupt enable) 断开符中断屏蔽 (使用断开分隔符来检测断开符) 0: 禁止中断; 1: 只要 USART_STS 寄存器中的 LBDF 为'1'就产生中断。
位 5	<b>LBDLEN:</b> LIN 断开符检测长度 (LIN break detection length) 该位用来选择是 11 位还是 10 位的断开符检测 0: 10 位的断开符检测; 1: 11 位的断开符检测。 注意: LBDLEN 可用于 LIN 模式及其他模式下的断开帧的检测长度控制, 且检测长度和 LIN 模式相同。
位 4	保留位, 硬件强制为 0
位 3: 0	<b>ADDR[3: 0]:</b> 本设备的 USART 节点地址 该位域给出本设备 USART 节点的地址。这是在多处理器通信下的静默模式中使用的, 使用地址标记来唤醒某个 USART 设备。

注意: 在使能发送后不能改写这三个位 (CLKPOL、CLKPHA、LBCP)。

### 16.6.7 控制寄存器3 (USART\_CTRL3)

地址偏移: 0x14

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
res															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	CTS IEN	CTS EN	RTS EN	DMA TEN	DMA REN	SCM EN	NAC KEN	HALF SEL	IRD ALP	IRD AEN	ERR IEN				
res	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

位 31: 11	保留位, 硬件强制为 0
位 10	<b>CTSIEN:</b> CTSF 中断使能 (CTSF interrupt enable) 0: 禁止中断; 1: USART_STS 寄存器中的 CTSF 为'1'时生中断。
位 9	<b>CTSEN:</b> CTS 使能 (CTS enable) 0: 禁止 CTS 硬件流控制; 1: CTS 模式使能, 只有 CTS 输入信号有效 (拉成低电平) 时才能发送数据。如果在数据传输的过程中, CTS 信号变成无效, 那么发完这个数据后, 传输就停止下来。如果当 CTS 为无效时, 往数据寄存器里写数据, 则要等到 CTS 有效时才会发送这个数据。
位 8	<b>RTSEN:</b> RTS 使能 (RTS enable) 0: 禁止 RTS 硬件流控制; 1: RTS 中断使能, 只有接收缓冲区内有空余的空间时才请求下一个数据。当前数据发送完成后, 发送操作就需要暂停下来。如果可以接收数据了, 将 RTS 输出置为有效 (拉至低电平)。
位 7	<b>DMATEN:</b> DMA 使能发送 (DMA transmitter enable) 该位由软件设置或清除。 0: 禁止发送时的 DMA 模式。 1: 使能发送时的 DMA 模式;

位 6	<b>DMAREN:</b> DMA 使能接收 (DMA receiver enable) 该位由软件设置或清除。 0: 禁止接收时的 DMA 模式。 1: 使能接收时的 DMA 模式;
位 5	<b>SCMEN:</b> 智能卡模式使能 (Smart card mode enable) 该位用来使能智能卡模式 0: 禁止智能卡模式; 1: 使能智能卡模式。
位 4	<b>NACKEN:</b> 智能卡 NACKEN 使能 (Smart card NACK enable) 0: 校验错误出现时, 不发送 NACK; 1: 校验错误出现时, 发送 NACK。
位 3	<b>HALFSEL:</b> 半双工选择 (Half-duplex selection) 选择单线半双工模式 0: 不选择半双工模式; 1: 选择半双工模式。
位 2	<b>IRDALP:</b> 红外低功耗 (IrDA low-power) 该位用来选择普通模式还是低功耗红外模式 0: 普通模式; 1: 低功耗模式。
位 1	<b>IRDAEN:</b> 红外模式使能 (IrDA mode enable) 该位由软件设置或清除。 0: 不使能红外模式; 1: 使能红外模式。
位 0	<b>ERRIEN:</b> 错误中断使能 (Error interrupt enable) 在多缓冲区通信模式下, 当有帧错误、过载或者噪声错误时 (USART_STS 中的 FERR=1, 或者 ORERR=1, 或者 NERR=1) 产生中断。 0: 禁止中断; 1: 只要 USART_CTRL3 中的 DMAREN=1, 并且 USART_STS 中的 FERR=1, 或者 ORERR=1, 或者 NERR=1, 则产生中断

### 16.6.8 保护时间和预分频寄存器 (GTP)

地址偏移: 0x18

复位值: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	保留
res																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
GTVAL[7: 0]								DIV[7: 0]								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
位 31: 16		保留位, 硬件强制为 0														

位 15: 8	<p><b>GTVAL[7: 0]:</b> 保护时间值 (Guard time value) 该位域规定了以波特时钟为单位的保护时间。在智能卡模式下，需要这个功能。当保护时间过去后，才会设置发送完成标志。</p>
位 7: 0	<p><b>DIV[7: 0]:</b> 预分频器值 (Prescaler value) -在红外 (IrDA) 低功耗模式下： DIV[7: 0]=红外低功耗波特率对系统时钟分频以获得低功耗模式下的频率： 源时钟被寄存器中的值（仅有 8 位有效）分频： 00000000: 保留–不要写入该值； 00000001: 对源时钟 1 分频； 00000010: 对源时钟 2 分频； ..... -在红外 (IrDA) 的普通模式下： DIV 只能设置为 00000001 -在智能卡模式下： DIV[4: 0]: 预分频值对系统时钟进行分频，给智能卡提供时钟。 寄存器中给出的值（低 5 位有效）乘以 2 后，作为对源时钟的分频因子 00000: 保留–不要写入该值； 00001: 对源时钟进行 2 分频； 00010: 对源时钟进行 4 分频； 00011: 对源时钟进行 6 分频； ..... 注意：位[7: 5]在智能卡模式下没有意义。</p>

# 17 串行外设接口（SPI）

## 17.1 SPI简介

SPI 接口可以配置为支持 SPI 协议或者支持 I<sup>2</sup>S 音频协议。

SPI 接口默认工作在 SPI 方式，可以通过软件把功能从 SPI 模式切换到 I<sup>2</sup>S 模式。

串行外设接口（SPI）允许芯片与外部设备以半/全双工、同步、串行方式通信。此接口可以被配置成主模式，并为外部从设备提供通信时钟（SCK）。接口还能以多主配置方式工作。它可用于多种用途，包括使用一条双向数据线的双线单工同步传输，还可使用 CRC 校验的可靠通信。

I<sup>2</sup>S 也是一种 3 引脚的同步串行接口通讯协议。它支持四种音频标准，包括飞利浦 I<sup>2</sup>S 标准，MSB 和 LSB 对齐标准，以及 PCM 标准。它在半双工通讯中，可以工作在主和从 2 种模式下。当它作为主设备时，通过接口向外部的从设备提供时钟信号。

## 17.2 主要特点

### 17.2.1 SPI特点

- 3线全双工同步传输
- 带或不带第三根双向数据线的双线单工同步传输
- 8或16位传输帧格式选择
- 主或从操作
- 支持多主模式
- 10个主模式波特率预分频系数（最大为f<sub>PCLK</sub>/2）
- 从模式频率（最大为f<sub>PCLK</sub>/2）
- 主模式和从模式的快速通信
- 主模式和从模式下均可以由软件或硬件进行 NSS 管理：主/从操作模式的动态改变
- 可编程的时钟极性和相位
- 可编程的数据顺序，MSB 在前或 LSB 在前
- 可触发中断的专用发送和接收标志
- SPI总线忙状态标志
- 支持可靠通信的硬件CRC
  - 在发送模式下，CRC 值可以被作为最后一个字节发送
  - 在全双工模式中对接收到的最后一个字节自动进行 CRC 校验
- 可触发中断的主模式故障、过载以及 CRC 错误标志
- 支持 DMA 功能的 2 字节发送和接收缓冲器：产生发送和接受请求

### 17.2.2 I<sup>2</sup>S功能

- 单工通信（仅发送或接收）
- 主或者从操作
- 8位线性可编程预分频器，获得精确的音频采样频率（8KHz 到 192kHz）
- 数据格式可以是 16 位，24 位或者 32 位
- 音频信道固定数据包帧为 16 位（16 位数据帧）或 32 位（16、24 或 32 位数据帧）
- 可编程的时钟极性（稳定态）
- 从发送模式下的下溢标志位和主/从接收模式下的上溢标志位

- 16位数据寄存器用来发送和接收，在通道两端各有一个寄存器
- 支持的I<sup>2</sup>S协议：
  - I<sup>2</sup>S飞利浦标准
  - MSB对齐标准（左对齐）
  - LSB对齐标准（右对齐）
  - PCM标准（16位通道帧上带长或短帧同步或者16位数据帧扩展为32位通道帧）
- 数据方向总是MSB在先
- 发送和接收都具有DMA能力
- 主时钟可以输出到外部音频设备，比率固定为256xFs（Fs为音频采样频率）

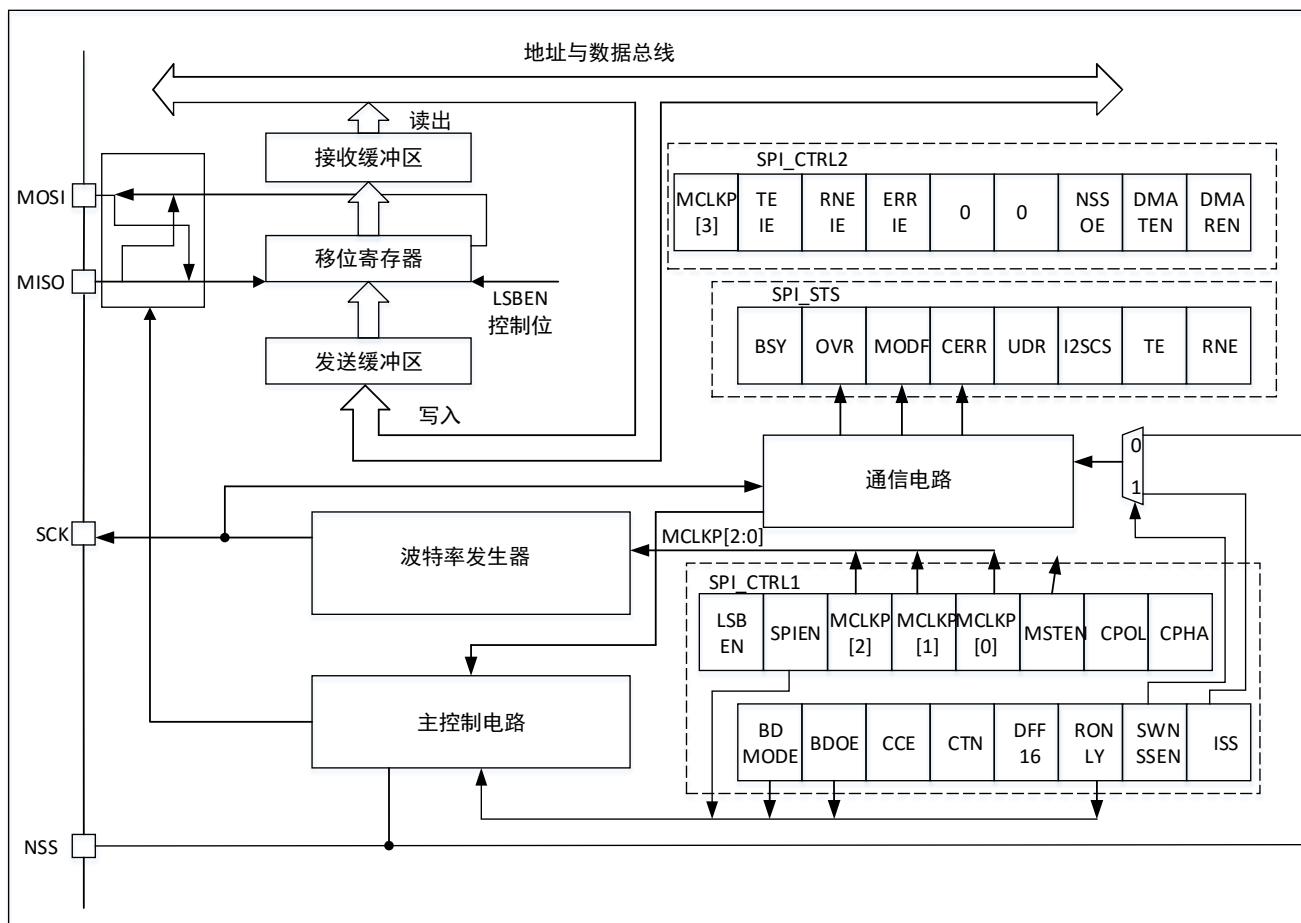
## 17.3 功能描述

### 17.3.1 SPI功能描述

#### 17.3.1.1 概述

SPI的方框图见下图。

图表 17-1 SPI框图

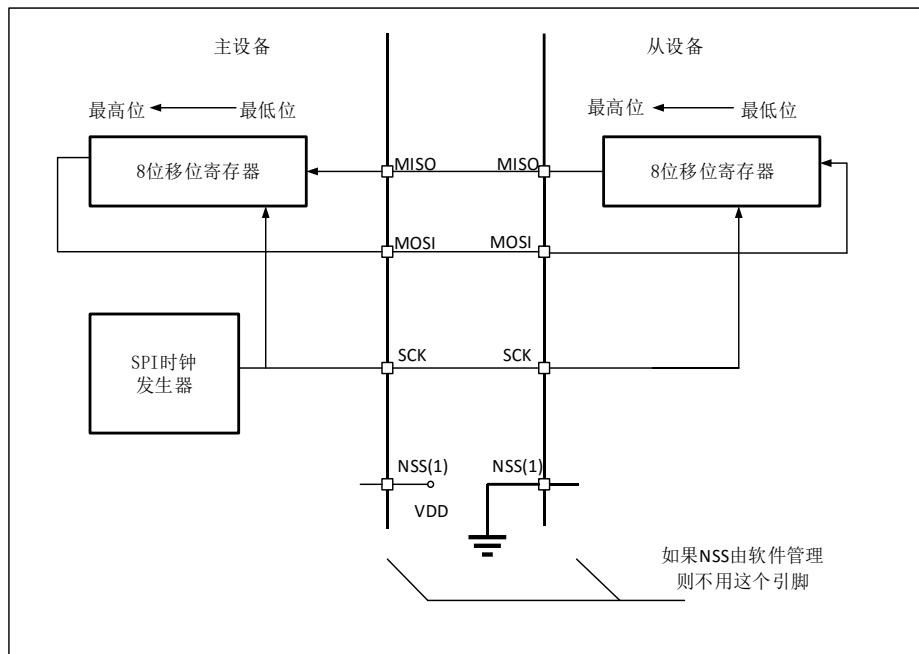


通常 SPI 通过 4 个引脚与外部器件相连：

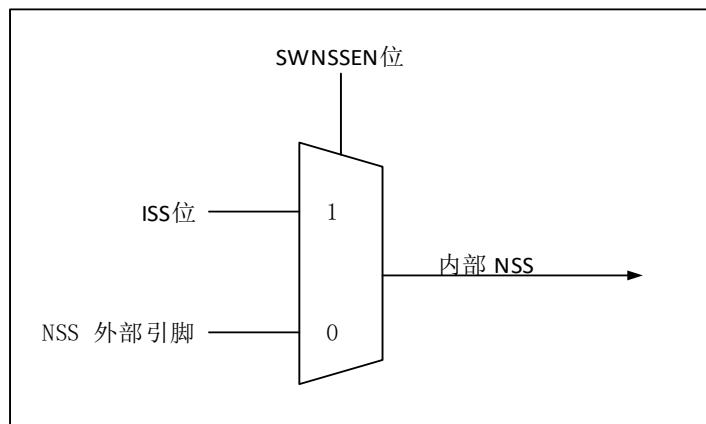
- **MISO:** 主设备输入/从设备输出引脚。该引脚在从模式下发送数据，在主模式下接收数据。
- **MOSI:** 主设备输出/从设备输入引脚。该引脚在主模式下发送数据，在从模式下接收数据。
- **SCK:** 串口时钟，作为主设备的输出，从设备的输入。
- **NSS:** 从设备选择。这是一个可选的引脚，用来选择主/从设备。它的功能是用来作为“片选引脚”，让主设备可以单独地与特定从设备通讯，避免数据线上的冲突。从设备的 NSS 引脚可以由主设备的一个标准 I/O 引脚来驱动。一旦被使能（NSSOE 位），NSS 引脚也可以作为输出引脚，并在 SPI 处于主模式时拉低；此时，所有的 SPI 设备，如果它们的 NSS 引脚连接到主设备的 NSS 引脚，则会检测到低电平，如果它们被设置为 NSS 硬件模式，就会自动进入从设备状态。当配置为主设备、NSS 配置为输入引脚（MSTEN=1, NSSOE=0）时，如果 NSS 被拉低，则这个 SPI 设备进入主模式失败状态：即 MSTEN 位被自动清除，此设备进入从模式。

下图是一个单主和单从设备互连的例子。

图表 17-2 单主和单从应用



图表 17-3 硬件/软件的从选择管理



#### 时钟信号的相位和极性

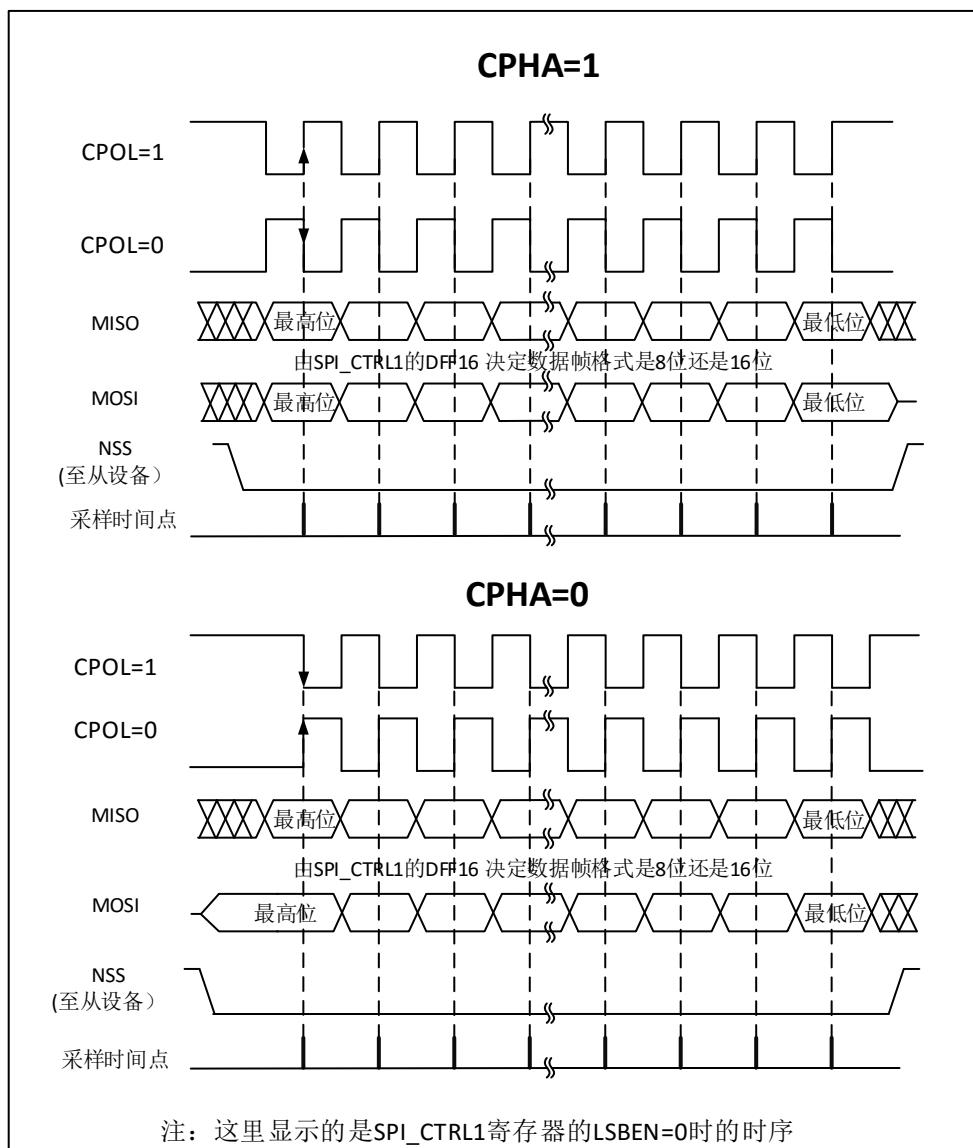
SPI\_CTRL1 寄存器的 CPOL 和 CPHA 位，能够组合成四种可能的时序关系。CPOL（时钟极性）位控制在没有数据传输时时钟的空闲状态电平，此位对主模式和从模式下的设备都有效。如果 CPOL 被清‘0’，SCK 引脚在空闲状态保持低电平；如果 CPOL 被置‘1’，SCK 引脚在空闲状态保持高电平。

如果 CPHA（时钟相位）位被置‘1’，SCK 时钟的第二个边沿（CPOL 位为‘0’时就是下降沿，CPOL 位为‘1’时就是上升沿）进行数据位的采样，数据在第二个时钟边沿被锁存。如果 CPHA 位被清‘0’，SCK 时钟的第一边沿（CPOL 位为‘0’时就是上升沿，CPOL 位为‘1’时就是下降沿）进行数据位采样，数据在第一个时钟边沿被锁存。

CPOL 时钟极性和 CPHA 时钟相位的组合选择数据捕捉的时钟边沿。[图 17-4](#) 显示了 SPI 传输的 4 种 CPHA 和 CPOL 位组合。此图可以解释为主设备和从设备的 SCK 脚、MISO 脚、MOSI 脚直接连接的主或从时序图。

- 注意：
1. 在改变 CPOL/CPHA 位之前，必须清除 SPIEN 位将 SPI 禁止。
  2. 主和从必须配置成相同的时序模式。
  3. SCK 的空闲状态必须和 SPI\_CTRL1 寄存器指定的极性一致（CPOL 为‘1’时，空闲时应上拉 SCK 为高电平；CPOL 为‘0’时，空闲时应下拉 SCK 为低电平）。
  4. 数据帧格式（8 位或 16 位）由 SPI\_CTRL1 寄存器的 DFF16 位选择，并且决定发送/接收的数据长度。

图表 17-4 数据时钟时序图



### 数据帧格式

根据 SPI\_CTRL1 寄存器中的 LSBEN 位，输出数据位时可以 MSB 在先也可以 LSB 在先。根据 SPI\_CTRL1 寄存器的 DFF16 位，每个数据帧可以是 8 位或是 16 位。所选择的数据帧格式对发送和/或接收都有效。

#### 17.3.1.2 配置 SPI 为从模式

在从模式下，SCK 引脚用于接收从主设备来的串行时钟。SPI\_CTRL1 寄存器中 MCLKP[3: 0]的设置不影响数据传输速率。

**注意：** 建议在主设备发送时钟之前使能 SPI 从设备，否则可能会发生意外的数据传输。在通信时钟的第一个边沿到来之前或正在进行的通信结束之前，从设备的数据寄存器必须就绪。在使能从设备和主设备之前，通信时钟的极性必须处于稳定的数值。

请按照以下步骤配置 SPI 为从模式：

#### 配置步骤

1. 设置 DFF16 位以定义数据帧格式为 8 位或 16 位。
2. 选择 CPOL 和 CPHA 位来定义数据传输和串行时钟之间的相位关系（见图 17-4）。为保证正确的数据传输，从设备和主设备的 CPOL 和 CPHA 位必须配置成相同的方式。
3. 帧格式（SPI\_CTRL1 寄存器中的 LSBEN 位定义的“MSB 在前”还是“LSB 在前”）必

须与主设备相同。

4. 硬件模式下（参考从选择（NSS）脚管理部分），在完整的数据帧（8位或16位）传输过程中，NSS引脚必须为低电平。在NSS软件模式下，设置SPI\_CTRL1寄存器中的SWNSSEN位并清除ISS位。
5. 清除MSTEN位、设置SPIEN位(SPI\_CTRL1寄存器)，使相应引脚工作于SPI模式下。在这个配置中，MOSI引脚是数据输入，MISO引脚是数据输出。

#### 数据发送过程

在写操作中，数据被写入发送缓冲器。

当从设备收到时钟信号，并且在MOSI引脚上出现第一个数据位时，发送过程开始（注：此时第一个位被发送出去）。余下的位（对于8位数据帧格式，还有7位；对于16位数据帧格式，还有15位）被装进移位寄存器。当发送缓冲器中的数据传输到移位寄存器时，SPI\_STS寄存器的TE标志被设置，如果设置了SPI\_CTRL2寄存器的TEIE位，将会产生中断。

#### 数据接收过程

对于接收器，当数据接收完成时：

- 移位寄存器中的数据传送到接收缓冲器，SPI\_STS寄存器中的RNE标志被设置。
- 如果设置了SPI\_CTRL2寄存器中的RNEIE位，则产生中断。

在最后一个采样时钟边沿后，RNE位被置‘1’，移位寄存器中接收到的数据字节被传送到接收缓冲器。当读SPI\_DT寄存器时，SPI设备返回这个接收缓冲器的数值。

读SPI\_DT寄存器时，RNE位被清除。

### 17.3.1.3 配置SPI为主模式

在主配置时，在SCK脚产生串行时钟。

#### 配置步骤

1. 通过SPI\_CTRL1寄存器的MCLKP[3: 0]位定义串行时钟波特率。
2. 选择CPOL和CPHA位，定义数据传输和串行时钟间的相位关系（见图17-4）。
3. 设置DFF16位来定义8位或16位数据帧格式。
4. 配置SPI\_CTRL1寄存器的LSBEN位定义帧格式。
5. 如果需要NSS引脚工作在输入模式，硬件模式下，在整个数据帧传输期间应把NSS脚连接到高电平；在软件模式下，需设置SPI\_CTRL1寄存器的SWNSSEN位和ISS位。如果NSS引脚工作在输出模式，则只需设置NSSOE位。
6. 必须设置MSTEN位和SPIEN位（只当NSS脚被连到高电平，这些位才能保持置位）。在这个配置中，MOSI引脚是数据输出，而MISO引脚是数据输入。

#### 数据发送过程

当写入数据至发送缓冲器时，发送过程开始。在发送第一个数据位时，数据字被并行地（通过内部总线）传入移位寄存器，而后串行地移出到MOSI脚上；MSB在先还是LSB在先，取决于SPI\_CTRL1寄存器中的LSBEN位的设置。数据从发送缓冲器传输到移位寄存器时TE标志将被置位，如果设置了SPI\_CTRL1寄存器中的TEIE位，将产生中断。

#### 数据接收过程

对于接收器来说，当数据传输完成时：

- 传送移位寄存器里的数据到接收缓冲器，并且RNE标志被置位。
- 如果设置了SPI\_CTRL2寄存器中的RNEIE位，则产生中断。

在最后采样时钟沿，RNE位被设置，在移位寄存器中接收到的数据字被传送到接收缓冲器。读SPI\_DT寄存器时，SPI设备返回接收缓冲器中的数据。

读SPI\_DT寄存器将清除RNE位。一旦传输开始，如果下一个将发送的数据被放进了发送缓冲器，就可以维持一个连续的传输流。在试图写发送缓冲器之前，需确认TE标志应该为‘1’。

注意：在 NSS 硬件模式下，从设备的 NSS 输入由 NSS 引脚控制或另一个由软件驱动的 GPIO 引脚控制。

### 17.3.1.4 配置SPI为单工通信

SPI 模块能够以两种配置工作于单工方式：

- 1条时钟线和1条双向数据线；
- 1条时钟线和1条数据线（只接收或只发送）；

#### 1条时钟线和1条双向数据线（BDMODE=1）

设置 SPI\_CTRL1 寄存器中的 BDMODE 位而启用此模式。在这个模式下，SCK 引脚作为时钟，主设备使用 MOSI 引脚而从设备使用 MISO 引脚作为数据通信。传输的方向由 SPI\_CTRL1 寄存器里的 BDOE 控制，当这个位是‘1’的时候，数据线是输出，否则是输入。

#### 1条时钟和1条单向数据线（BDMODE=0）

在这个模式下，SPI 模块可以或者作为只发送，或者作为只接收。

- 只发送模式类似于全双工模式（BDMODE=0, RONLY=0）：数据在发送引脚（主模式时是MOSI、从模式时是MISO）上传输，而接收引脚（主模式时是MISO、从模式时是MOSI）可以作为通用的I/O使用。此时，软件不必理会接收缓冲器中的数据（如果读出数据寄存器，它不包含任何接收数据）。
- 在只接收模式，可以通过设置SPI\_CTRL1寄存器的RONLY位而关闭SPI的输出功能；此时，发送引脚（主模式时是MOSI、从模式时是MISO）被释放，可以作为其它功能使用。

配置并使能 SPI 模块为只接收模式的方式是：

- 在主模式时，一旦使能SPI，通信立即启动，当清除SPIEN位时立即停止当前的接收。在此模式下，不必读取BSY标志，在SPI通信期间这个标志始终为‘1’。
- 在从模式时，只要NSS被拉低（或在NSS软件模式时，ISS位为‘0’）同时SCK有时钟脉冲，SPI就一直在接收。

### 17.3.1.5 数据发送与接收过程

#### 接收与发送缓冲器

在接收时，接收到的数据被存放在一个内部的接收缓冲器中；在发送时，在被发送之前，数据将首先被存放在一个内部的发送缓冲器中。

对 SPI\_DT 寄存器的读操作，将返回接收缓冲器的内容；写入 SPI\_DT 寄存器的数据将被写入发送缓冲器中。

#### 主模式下开始传输

- 全双工模式（BDMODE=0并且RONLY=0）
  - 当写入数据到 SPI\_DT 寄存器（发送缓冲器）后，传输开始；
  - 在传送第一位数据的同时，数据被并行地从发送缓冲器传送到 8 位的移位寄存器中，然后按顺序被串行地移位送到 MOSI 引脚上；
  - 与此同时，在 MISO 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI\_DT 寄存器（接收缓冲器）中。
- 单向的只接收模式（BDMODE=0并且RONLY=1）
  - SPIEN=1 时，传输开始；
  - 只有接收器被激活，在 MISO 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI\_DT 寄存器（接收缓冲器）中。
- 双向模式，发送时（BDMODE=1并且BDOE=1）
  - 当写入数据到 SPI\_DT 寄存器（发送缓冲器）后，传输开始；

- 在传送第一位数据的同时，数据被并行地从发送缓冲器传送到 8 位的移位寄存器中，然后按顺序被串行地移位送到 MOSI 引脚上；
- 不接收数据。
- 双向模式，接收时（BDMODE=1 并且 BDOE=0）
  - SPIEN=1 并且 BDOE=0 时，传输开始；
  - 在 MOSI 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI\_DT 寄存器（接收缓冲器）中。
  - 不激活发送器，没有数据被串行地送到 MOSI 引脚上。

### 从模式下开始传输

- 全双工模式（BDMODE=0 并且 RONLY=0）
  - 当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始，随后的数据位依次移动进入移位寄存器；
  - 与此同时，在传输第一个数据位时，发送缓冲器中的数据被并行地传送到 8 位的移位寄存器，随后被串行地发送到 MISO 引脚上。软件必须保证在 SPI 主设备开始数据传输之前在发送寄存器中写入要发送的数据。
- 单向的只接收模式（BDMODE=0 并且 RONLY=1）
  - 当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始，随后数据位依次移动进入移位寄存器；
  - 不启动发送器，没有数据被串行地传送到 MISO 引脚上。
- 双向模式，发送时（BDMODE=1 并且 BDOE=1）
  - 当从设备接收到时钟信号并且发送缓冲器中的第一个数据位被传送到 MISO 引脚上的时候，数据传输开始；
  - 在第一个数据位被传送到 MISO 引脚上的同时，发送缓冲器中要发送的数据被平行地传送到 8 位的移位寄存器中，随后被串行地发送到 MISO 引脚上。软件必须保证在 SPI 主设备开始数据传输之前在发送寄存器中写入要发送的数据；
  - 不接收数据。
- 双向模式，接收时（BDMODE=1 并且 BDOE=0）
  - 当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始；
  - 从 MISO 引脚上接收到的数据被串行地传送到 8 位的移位寄存器中，然后被平行地传送到 SPI\_DT 寄存器（接收缓冲器）；
  - 不启动发送器，没有数据被串行地传送到 MISO 引脚上。

### 处理数据的发送与接收

当数据从发送缓冲器传送到移位寄存器时，设置 TE 标志（发送缓冲器空），它表示内部的发送缓冲器可以接收下一个数据；如果在 SPI\_CTRL2 寄存器中设置了 TEIE 位，则此时会产生一个中断；写入 SPI\_DT 寄存器即可清除 TE 位。

**注意：** 在写入发送缓冲器之前，软件必须确认 TE 标志为‘1’，否则新的数据会覆盖已经在发送缓冲器中的数据。

在采样时钟的最后一个边沿，当数据被从移位寄存器传送到接收缓冲器时，设置 RNE 标志（接收缓冲器非空）；它表示数据已经就绪，可以从 SPI\_DT 寄存器读出；如果在 SPI\_CTRL2 寄存器中设置了 RXNIE 位，则此时会产生一个中断；读出 SPI\_DT 寄存器即可清除 RXNIE 标志位。

在一些配置中，传输最后一个数据时，可以使用 BSY 标志等待数据传输的结束。

### 主或从模式下（BDMODE=0 并且 RONLY=0）全双工发送和接收过程模式

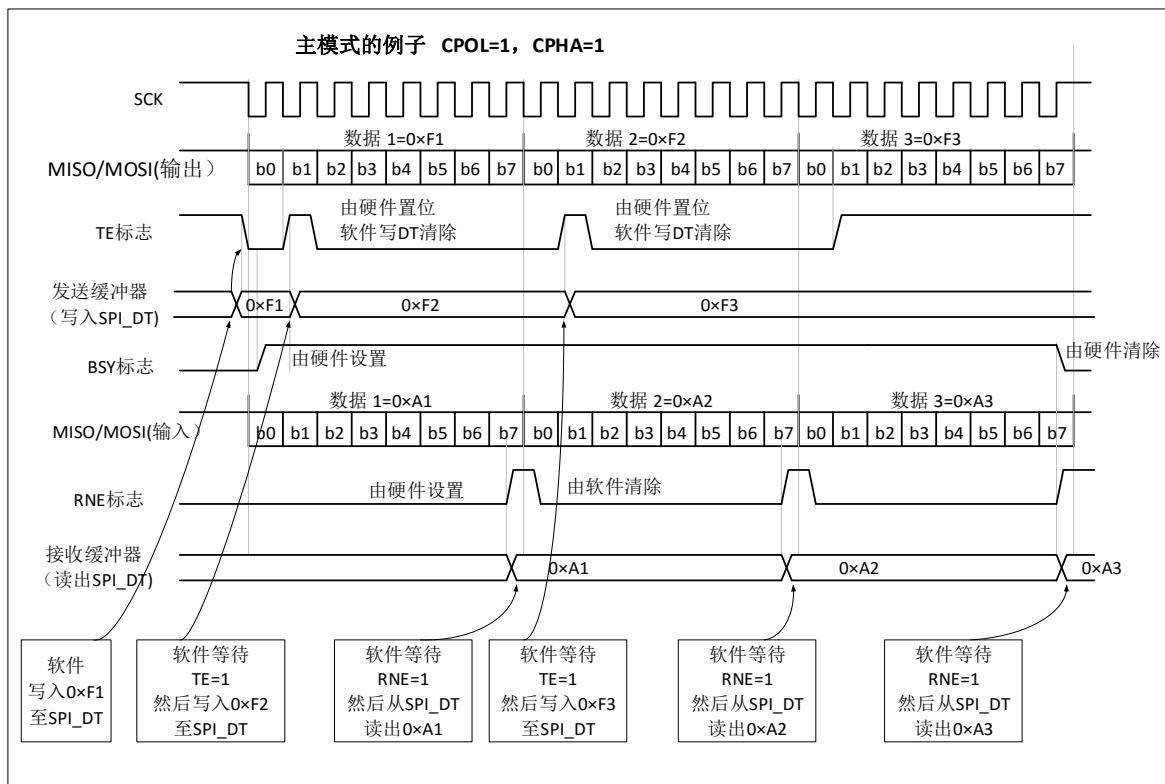
软件必须遵循下述过程，发送和接收数据（见图 17-5 和图 17-6）：

1. 设置 SPIEN 位为‘1’，使能 SPI 模块；

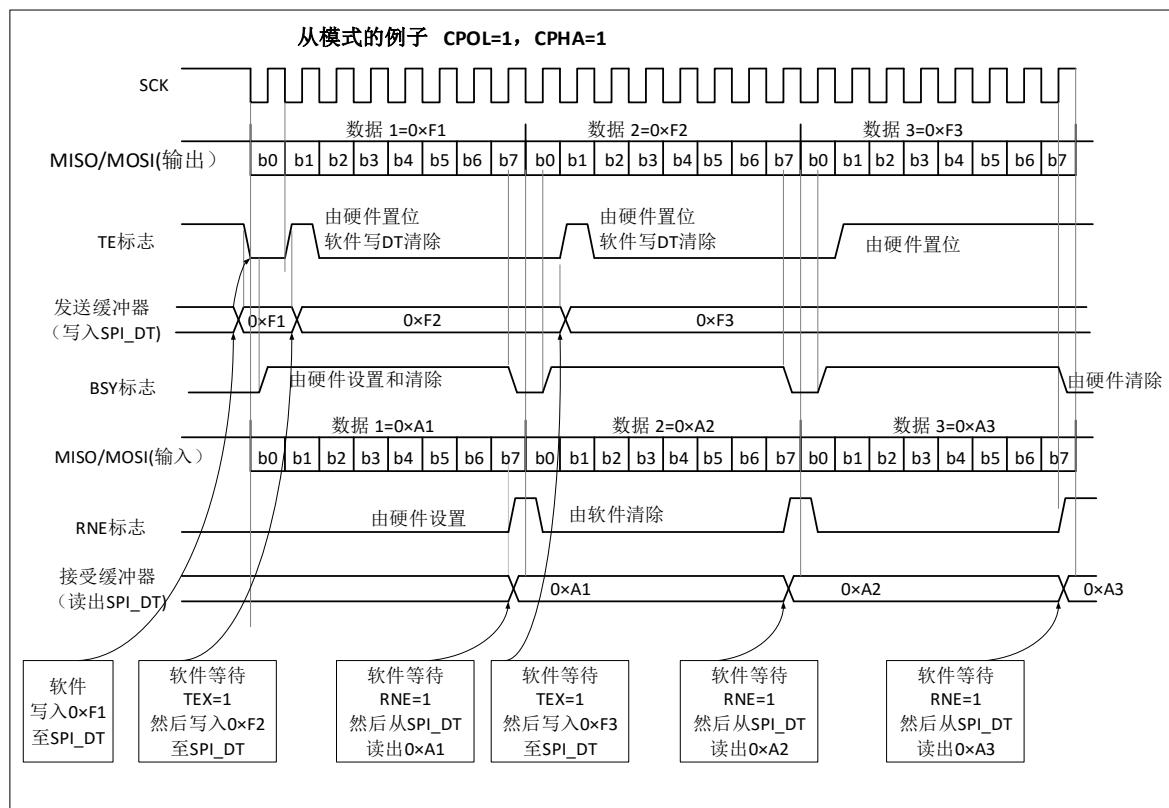
2. 在SPI\_DT寄存器中写入第一个要发送的数据，这个操作会清除TE标志；
3. 等待TE=1，然后写入第二个要发送的数据。等待RNE=1，然后读出SPI\_DT寄存器并获得第一个接收到的数据，读SPI\_DT的同时清除了RNE位。重复这些操作，发送后续的数据同时接收n-1个数据；
4. 等待RNE=1，然后接收最后一个数据；
5. 等待TE=1，在BSY=0之后关闭SPI模块。

也可以在响应RNE或TE标志的上升沿产生的中断的处理程序中实现这个过程。

图表 17-5 主模式、全双工模式下（BDMODE=0并且RONLY=0）连续输出时，TE/RNE/BSY的变化示意图



图表 17-6 从模式、全双工模式下（BDMODE=0 并且 RONLY=0）连续传输时，TE/RNE/BSY 的变化示意图。



### 只发送过程（BDMODE=0 并且 RONLY=0）

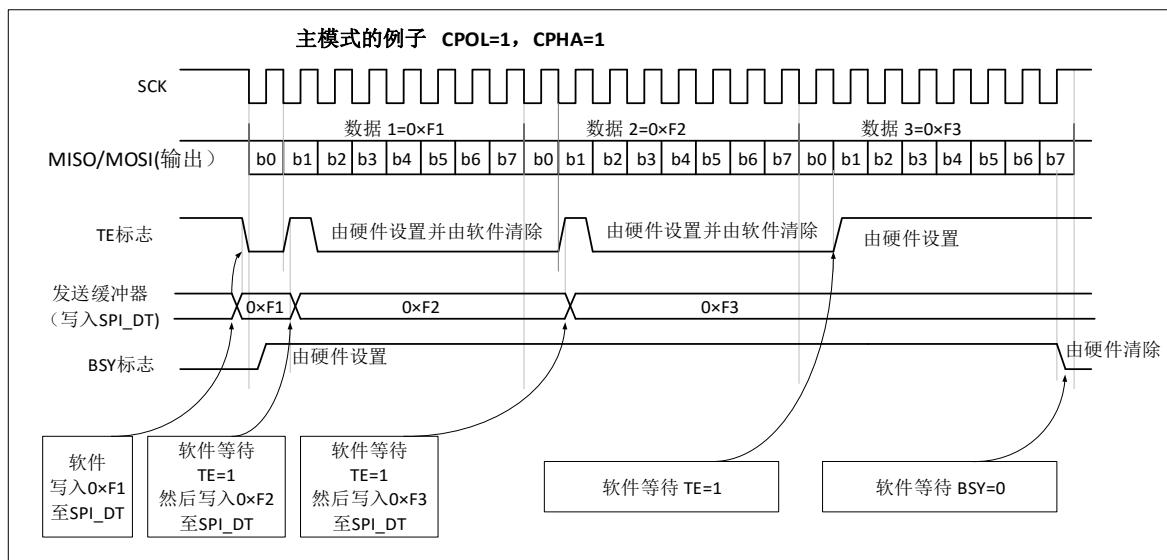
在此模式下，传输过程可以简要说明如下，使用 BSY 位等待传输的结束（见图 17-7 和图 17-8）：

1. 设置 SPIEN 位为‘1’，使能 SPI 模块；
2. 在 SPI\_DT 寄存器中写入第一个要发送的数据，这个操作会清除 TE 标志；
3. 等待 TE=1，然后写入第二个要发送的数据。重复这个操作，发送后续的数据；
4. 写入最后一个数据到 SPI\_DT 寄存器之后，等待 TE=1；然后等待 BSY=0，这表示最后一个数据的传输已经完成。

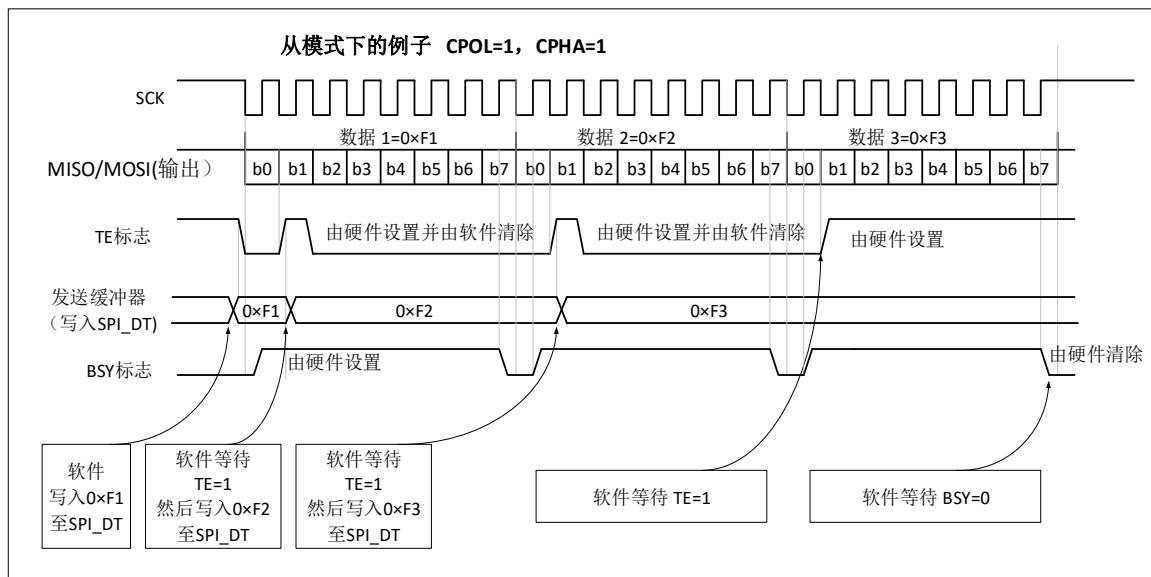
也可以在响应 TE 标志的上升沿产生的中断的处理程序中实现这个过程。

- 注意：**
1. 对于不连续的传输，在写入 SPI\_DT 寄存器的操作与设置 BSY 位之间有 2 个 APB 时钟周期的延迟，因此在只发送模式下，写入最后一个数据后，最好先等待 TE=1，然后再等待 BSY=0。
  2. 只发送模式下，在传输 2 个数据之后，由于不会读出接收到的数据，SPI\_STS 寄存器中的 OVR 位会变为‘1’。（注：软件不必理会这个 OVR 标志位）

图表 17-7 主设备只发送模式（BDMODE=0并且RONLY=0）下连续传输时，TE/BSY变化示意图



图表 17-8 从设备只发送模式（BDMODE=0并且RONLY=0）下连续传输时，TE/BSY变化示意图



### 双向发送过程（BDMODE=1 并且 BDOE=1）

在此模式下，操作过程类似于只发送模式，不同的是：在使能 SPI 模块之前，需要在 SPI\_CTRL1 寄存器中同时设置 BDMODE 和 BDOE 位为‘1’。

### 单向只接收模式（BDMODE=0 并且 RONLY=1）

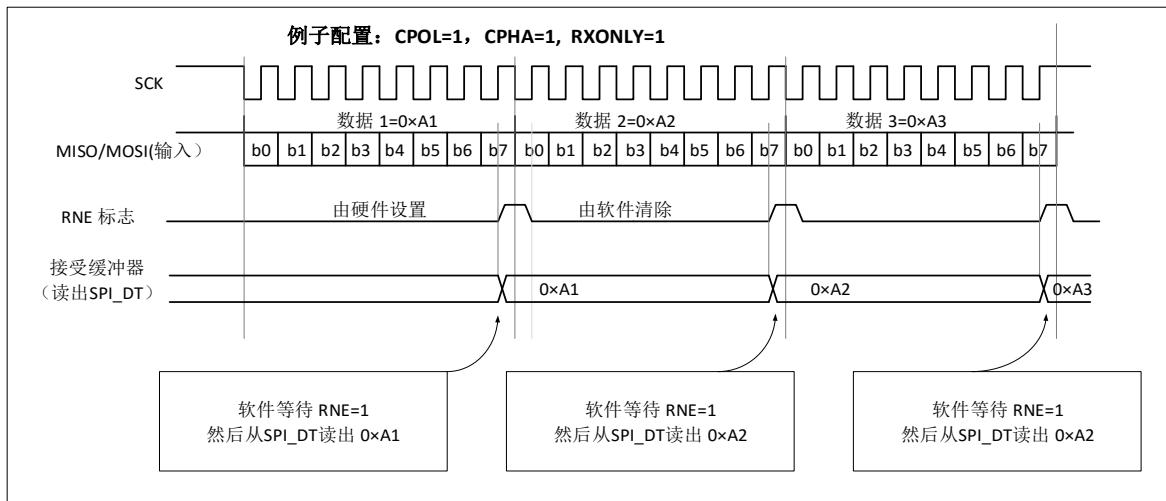
在此模式下，传输过程可以简要说明如下（见图 17-9）：

1. 在 SPI\_CTRL1 寄存器中，设置 RONLY=1；
2. 设置 SPIEN=1，使能 SPI 模块：
  - a) 主模式下，立刻产生 SCK 时钟信号，在关闭 SPI (SPIEN=0) 之前，不断地接收串行数据；
  - b) 从模式下，当 SPI 主设备拉低 NSS 信号并产生 SCK 时钟时，接收串行数据。
3. 等待 RNE=1，然后读出 SPI\_DT 寄存器以获得收到的数据（同时会清除 RNE 位）。重复这个操作接收所有数据。

也可以在响应 RNE 标志的上升沿产生的中断的处理程序中实现这个过程。

**注意：**如果在最后一个数据传输结束后关闭 SPI 模块，请按照[第 17.3.1.9 节](#)的建议操作。

图表 17-9 只接收模式（BDMODE=0 并且 RONLY=1）下连续传输时，RNE 变化示意图



### 双向接收过程（BDMODE=1 并且 BDOE=0）

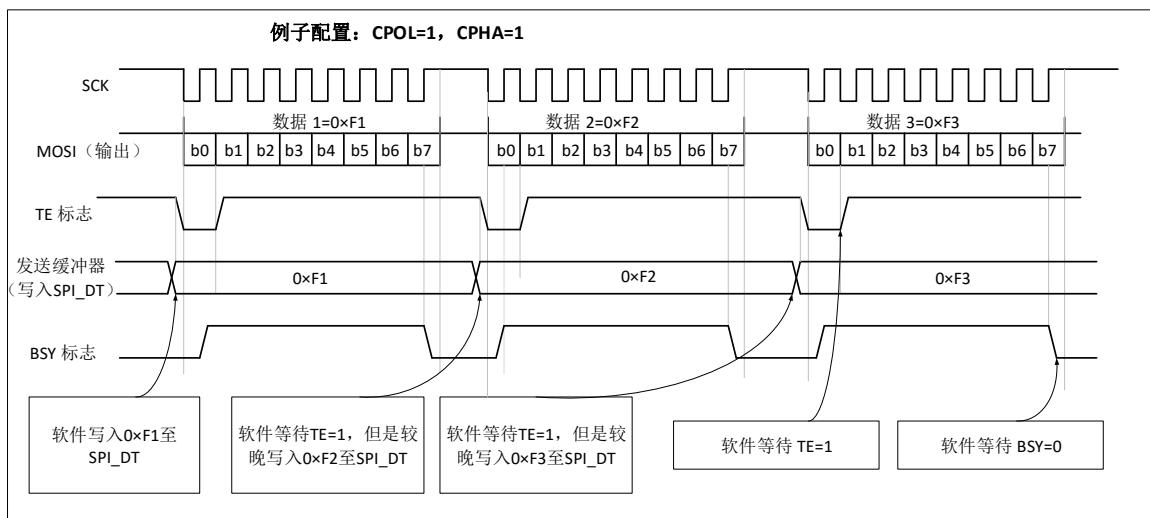
在此模式下，操作过程类似于只接收模式，不同的是：在使能 SPI 模块之前，需要在 SPI\_CTRL1 寄存器中设置 BDMODE 为‘1’并清除 BDOE 位为‘0’。

#### 连续和非连续传输

当在主模式下发送数据时，如果软件足够快，能够在检测到每次 TE 的上升沿（或 TE 中断），并立即在正在进行的传输结束之前写入 SPI\_DT 寄存器，则能够实现连续的通信；此时，在每个数据项的传输之间的 SPI 时钟保持连续，同时 BSY 位不会被清除。

如果软件不够快，则会导致不连续的通信；这时，在每个数据传输之间会被清除（见下图）。在主模式的只接收模式下（RONLY=1），通信总是连续的，而且 BSY 标志始终为‘1’。在从模式下，通信的连续性由 SPI 主设备决定。不管怎样，即使通信是连续的，BSY 标志会在每个数据项之间至少有一个 SPI 时钟周期为低（见[图 17-8](#)）。

图表 17-10 非连续传输发送（BDMODE=0 并且 RONLY=0）时，TE/BSY 变化示意图



### 17.3.1.6 CRC计算

CRC 校验用于保证全双工通信的可靠性。数据发送和数据接收分别使用单独的 CRC 计算器。通过对每一个接收位进行可编程的多项式运算来计算 CRC。CRC 的计算是在由 SPI\_CTRL1 寄存器中 CPHA 和 CPOL 位定义的采样时钟边沿进行的。

注意：该 SPI 接口提供了两种 CRC 计算方法，取决于所选的发送和/或接收的数据帧格式：  
8 位数据帧采用 CRC8；16 位数据帧采用 CRC16。

CRC 计算是通过设置 SPI\_CTRL1 寄存器中的 CCE 位启用的。设置 CCE 位时同时复位 CRC 寄存器 (SPI\_RCRC 和 SPI\_TCRC)。当设置了 SPI\_CTRL1 的 CTN 位，SPI\_TCRC 的内容将在当前字节发送之后发出。

在传输 SPI\_TCRC 的内容时，如果在移位寄存器中收到的数值与 SPI\_RCRC 的内容不匹配，则 SPI\_STS 寄存器的 CERR 标志位被置 1。

如果在 TX 缓冲器中还有数据，CRC 的数值仅在数据字节传输结束后传送。在传输 CRC 期间，CRC 计算器关闭，寄存器的数值保持不变。

注意：请参考产品说明书，以确认有此功能（不是所有型号都有此功能）。

SPI 通信可以通过以下步骤使用 CRC：

- 设置 CPOL、CPHA、LSBEN、MCLKP、SWNSSEN、ISS 和 MSTEN 的值；
- 在 SPI\_CPOLY 寄存器输入多项式；
- 通过设置 SPI\_CTRL1 寄存器 CCE 位使能 CRC 计算，该操作也会清除寄存器 SPI\_RCRC 和 SPI\_TCRC；
- 设置 SPI\_CTRL1 寄存器的 SPIEN 位启动 SPI 功能；
- 启动通信并且维持通信，直到只剩最后一个字节或者半字；
- 在把最后一个字节或半字写进发送缓冲器时，设置 SPI\_CTRL1 的 CTN 位，指示硬件在发送完成最后一个数据之后，发送 CRC 的数值。在发送 CRC 数值期间，停止 CRC 计算；
- 当最后一个字节或半字被发送后，SPI 发送 CRC 数值，CTN 位被清除。同样，接收到的 CRC 与 SPI\_RCRC 值进行比较，如果比较不相配，则设置 SPI\_STS 上的 CERR 标志位，当设置了 SPI\_CTRL2 寄存器的 ERRIE 时，则产生中断。

注意：1、当 SPI 模块处于从设备模式时，请注意在时钟稳定之后再使能 CRC 计算，否则可能会得到错误的 CRC 计算结果。事实上，只要设置了 CCE 位，只要在 SCK 引脚上有输入时钟，不管 SPIEN 位的状态，都会进行 CRC 的计算。  
2、当 SPI 时钟频率较高时，用户在发送 CRC 时必须小心。在 CRC 传输期间，使用 CPU 的时间应尽可能少；为了避免在接收最后的数据和 CRC 时出错，在发送 CRC 过程中应禁止函数调用。必须在发送/接收最后一个数据之前完成设置 CTN 位的操作。  
3、当 SPI 时钟频率较高时，因为 CPU 的操作会影响 SPI 的带宽，建议采用 DMA 模式以避免 SPI 降低的速度。  
4、当 AT32F421 配置为从模式并且使用了 NSS 硬件模式，NSS 引脚应该在数据传输和 CRC 传输期间保持为低。

当配置 SPI 为从模式并且使用 CRC 的功能，即使 NSS 引脚为高时仍然会执行 CRC 的计算（注：当 NSS 信号为高时，如果 SCK 引脚上有时钟脉冲，则 CRC 计算会继续执行）。例如：当主设备交替地与多个从设备进行通信时，将会出现这种情况（注：此时要想办法避免 CRC 的误操作）。

在不选中一个从设备（NSS 信号为高）转换到选中一个新的从设备（NSS 信号为低）的时候，为了保持主从设备端下次 CRC 计算结果的同步，应该清除主从两端的 CRC 数值。

按照下述步骤清除 CRC 数值：

1. 关闭 SPI 模块 (SPIEN=0)；
2. 清除 CCE 位为 '0'；
3. 设置 CCE 位为 '1'；
4. 使能 SPI 模块 (SPIEN=1)。

### 17.3.1.7 状态标志

应用程序通过 3 个状态标志可以完全监控 SPI 总线的状态。

#### 发送缓冲器空闲标志 (TE)

此标志为' 1' 时表明发送缓冲器为空，可以写下一个待发送的数据进入缓冲器中。当写入 SPI\_DT 时，TE 标志被清除。

#### 接收缓冲器非空 (RNE)

此标志为' 1' 时表明在接收缓冲器中包含有效的接收数据。读 SPI 数据寄存器可以清除此标志。

#### 忙 (Busy) 标志

BSY 标志由硬件设置与清除（写入此位无效果），此标志表明 SPI 通信层的状态。

当它被设置为' 1' 时，表明 SPI 正忙于通信，但有一个例外：在主模式的双向接收模式下（MSTEN=1、BDMODE =1 并且 BDOE=0），在接收期间 BSY 标志保持为低。

在软件要关闭 SPI 模块并进入停机模式（或关闭设备时钟）之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输，因此需要严格按照下述过程执行。

BSY 标志还可以用于在多主系统中避免写冲突。

除了主模式的双向接收模式（MSTEN=1、BDMODE =1 并且 BDOE=0），当传输开始时，BSY 标志被置' 1'。

以下情况时此标志将被清除为' 0'：

- 当传输结束（主模式下，如果是连续通信的情况例外）；
- 当关闭 SPI 模块；
- 当产生主模式失效（MODF=1）。

如果通信不是连续的，则在每个数据项的传输之间，BSY 标志为低。

当通信是连续时：

- 主模式下：在整个传输过程中，BSY 标志保持为高；
- 从模式下：在每个数据项的传输之间，BSY 标志在一个 SPI 时钟周期中为低。

注意：不要使用 BSY 标志处理每一个数据项的发送和接收，最好使用 TE 和 RNE 标志。

### 17.3.1.8 关闭 SPI

当通讯结束，可以通过关闭 SPI 模块来终止通讯。清除 SPIEN 位即可关闭 SPI。在某些配置下，如果在传输还未完成时，就关闭 SPI 模块并进入停机模式，则可能导致当前的传输被破坏，而且 BSY 标志也变得不可信。

为了避免发生这种情况，关闭 SPI 模块时，建议按照下述步骤操作：

#### 在主或从模式下的全双工模式 (BDMODE=0, RONLY=0)

1. 等待 RNE=1 并接收最后一个数据；
2. 等待 TE=1；
3. 等待 BSY=0；
4. 关闭 SPI (SPIEN=0)，最后进入停机模式（或关闭该模块的时钟）。

#### 在主或从模式下的单向只发送模式 (BDMODE=0, RONLY=0) 或双向的发送模式 (BDMODE=1, BDOE=1)

在 SPI\_DT 寄存器中写入最后一个数据后：

1. 等待 TE=1；
2. 等待 BSY=0；
3. 关闭 SPI (SPIEN=0)，最后进入停机模式（或关闭该模块的时钟）。

#### 在主模式下的单向只接收模式 (MSTEN=1, BDMODE=0, RONLY=1) 或双向的接收模式 (MSTEN=1, BDMODE=1, BDOE=0)

这种情况需要特别地处理，以保证 SPI 不会开始一次新的传输：

1. 等待倒数第二个（第 n-1 个）RNE=1；
2. 在关闭 SPI (SPIEN=0) 之前等待一个 SPI 时钟周期（使用软件延迟）；
3. 在进入停机模式（或关闭该模块的时钟）之前等待最后一个 RNE=1。

**注意：**在主模式下的双向只接收模式 (MSTEN=1, BDMODE=1, BDOE=0) 时，传输过程中 BSY 标志始终为低。

在从模式下的只接收模式 (MSTEN=0, BDMODE=0, RONLY=1) 或双向的接收模式 (MSTEN=0, BDMODE=1, BDOE=0)

1. 可以在任何时候关闭 SPI (SPIEN=0)，SPI 会在当前的传输结束后被关闭；
2. 如果希望进入停机模式，在进入停机模式（或关闭该模块的时钟）之前必须首先等待 BSY=0。

### 17.3.1.9 利用 DMA 的 SPI 通信

为了达到最大通信速度，需要及时往 SPI 发送缓冲器填数据，同样接收缓冲器中的数据也必须及时读走以防止溢出。为了方便高速率的数据传输，SPI 实现了一种采用简单的请求/应答的 DMA 机制。

当 SPI\_CTRL2 寄存器上的对应使能位被设置时，SPI 模块可以发出 DMA 传输请求。发送缓冲器和接收缓冲器亦有各自的 DMA 请求。

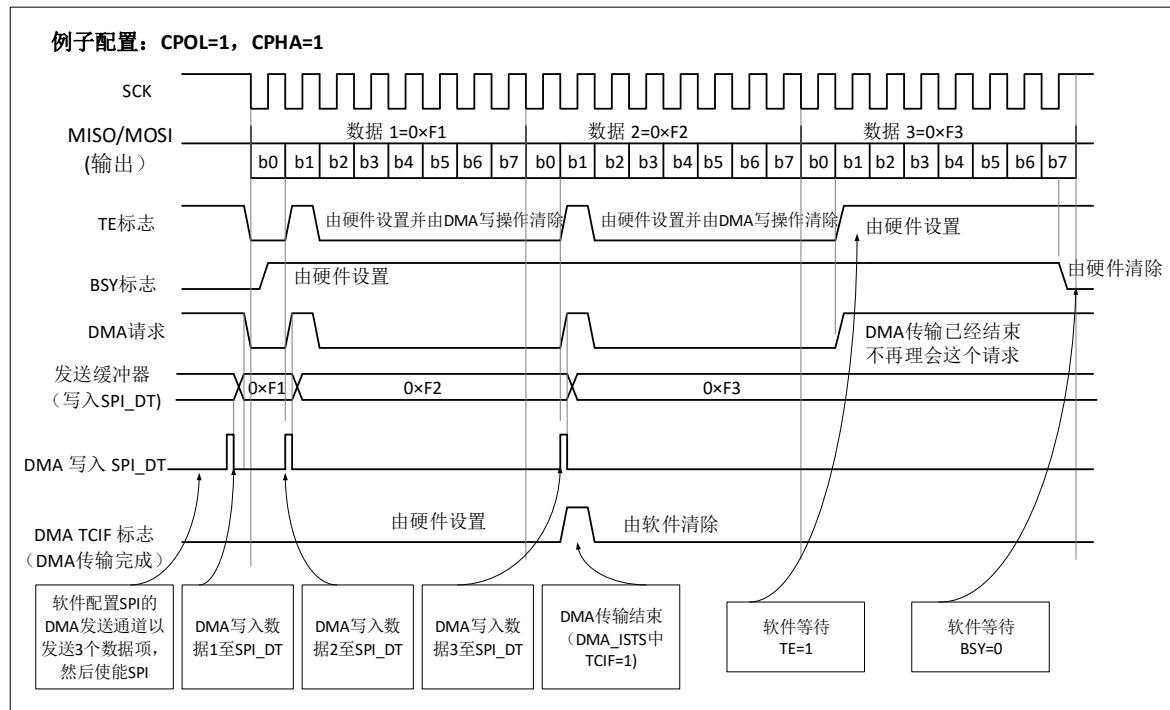
- 发送时，在每次 TE 被设置为‘1’时发出 DMA 请求，DMA 控制器则写数据至 SPI\_DT 寄存器，TE 标志因此而被清除。
- 接收时，在每次 RNE 被设置为‘1’时发出 DMA 请求，DMA 控制器则从 SPI\_DT 寄存器读出数据，RNE 标志因此而被清除。

当只使用 SPI 发送数据时，只需使能 SPI 的发送 DMA 通道。此时，因为没有读取收到的数据，OVR 被置为‘1’（译注：软件不必理会这个标志）。当只使用 SPI 接收数据时，只需使能 SPI 的接收 DMA 通道。

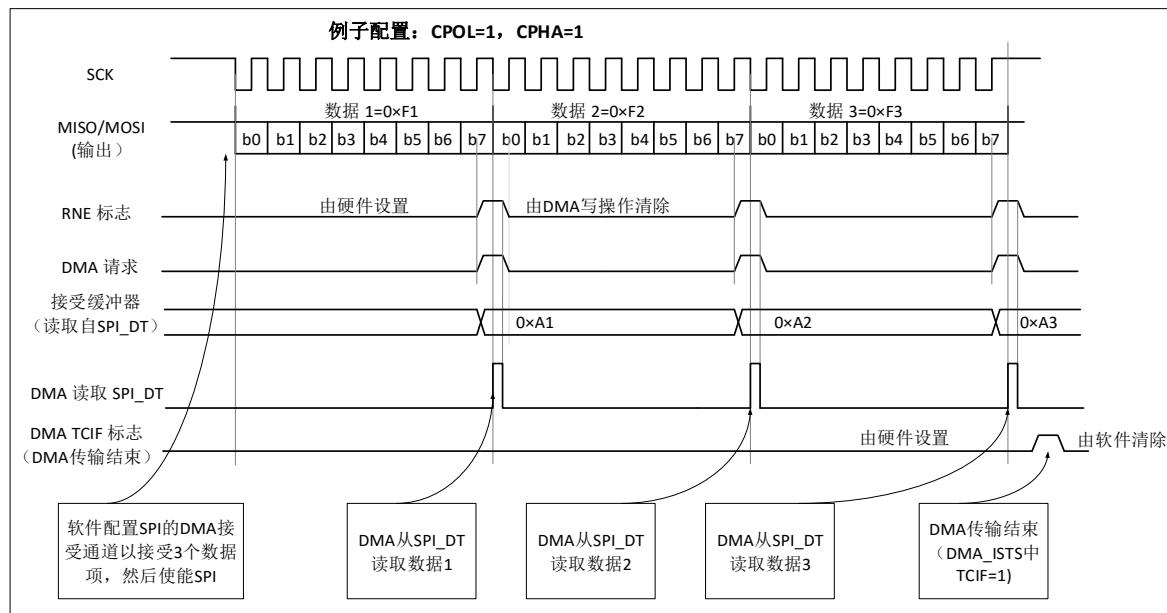
在发送模式下，当 DMA 已经传输了所有要发送的数据 (DMA\_ISTS 寄存器的 TCIF 标志变为‘1’) 后，可以通过监视 BSY 标志以确认 SPI 通信结束，这样可以避免在关闭 SPI 或进入停机模式时，破坏最后一个数据的传输。因此软件需要先等待 TE=1，然后等待 BSY=0。

**注意：**在不连续的通信中，在写数据到 SPI\_DT 的操作与 BSY 位被置为‘1’之间，有 2 个 APB 时钟周期的延迟，因此，在写完最后一个数据后需要先等待 TE=1 再等待 BSY=0。

图表 17-11 使用 DMA 发送



图表 17-12 使用 DMA 接收



### 带 CRC 的 DMA 功能

当使能 SPI 使用 CRC 检验并且启用 DMA 模式时，在通信结束时，CRC 字节的发送和接收是自动完成的。

数据和 CRC 传输结束时，SPI\_STS 寄存器的 CERR 标志为' 1' 表示在传输期间发生错误。

#### 17.3.1.10 错误标志

##### 主模式失效错误 (MODF)

主模式失效仅发生在：NSS 引脚硬件模式管理下，主设备的 NSS 脚被拉低；或者在 NSS 引脚软件模式管理下，ISS 位被置为' 0' 时；MODF 位被自动置位。主模式失效对 SPI 设备有以下影响：

- MODF位被置为'1'，如果设置了ERRIE位，则产生SPI中断；
- SPIEN位被清为'0'。这将停止一切输出，并且关闭SPI接口；
- MSTEN位被清为'0'，因此强迫此设备进入从模式。

下面的步骤用于清除 MODF 位：

1. 当 MODF 位被置为'1'时，执行一次对 SPI\_STS 寄存器的读或写操作；
2. 然后写 SPI\_CTRL1 寄存器。

在有多个 MCU 的系统中，为了避免出现多个从设备的冲突，必须先拉高该主设备的 NSS 脚，再对 MODF 位进行清零。在完成清零之后，SPIEN 和 MSTEN 位可以恢复到它们的原始状态。

出于安全的考虑，当 MODF 位为'1'时，硬件不允许设置 SPIEN 和 MSTEN 位。

通常配置下，从设备的 MODF 位不能被置为'1'。然而，在多主配置里，一个设备可以在设置了 MODF 位的情况下，处于从设备模式；此时，MODF 位表示可能出现了多主冲突。中断程序可以执行一个复位或返回到默认状态来从错误状态中恢复。

### 溢出错误

当主设备已经发送了数据字节，而从设备还没有清除前一个数据字节产生的 RNE 时，即为溢出错误。当产生溢出错误时：

- OVR位被置为'1'；当设置了ERRIE位时，则产生中断。

此时，接收器缓冲器的数据不是主设备发送的新数据，读 SPI\_DT 寄存器返回的是之前未读的数据，所有随后传送的数据都被丢弃。

依次读出 SPI\_DT 寄存器和 SPI\_STS 寄存器可将 OVR 清除。

### CRC 错误

当设置了 SPI\_CTRL 寄存器上的 CCE 位时，CRC 错误标志用来核对接收数据的有效性。如果移位寄存器中接收到的值（发送方发送的 SPI\_TCRC 数值）与接收方 SPI\_RCRC 寄存器中的数值不匹配，则 SPI\_STS 寄存器上的 CERR 标志被置位为'1'。

## 17.3.1.11 SPI中断

表格 17-1 SPI中断请求

中断事件	事件标志	使能控制位
发送缓冲器空标志	TE	TEIE
接收缓冲器非空标志	RNE	RNEIE
主模式失效事件	MODF	ERRIE
溢出错误	OVR	
CRC 错误标志	CERR	

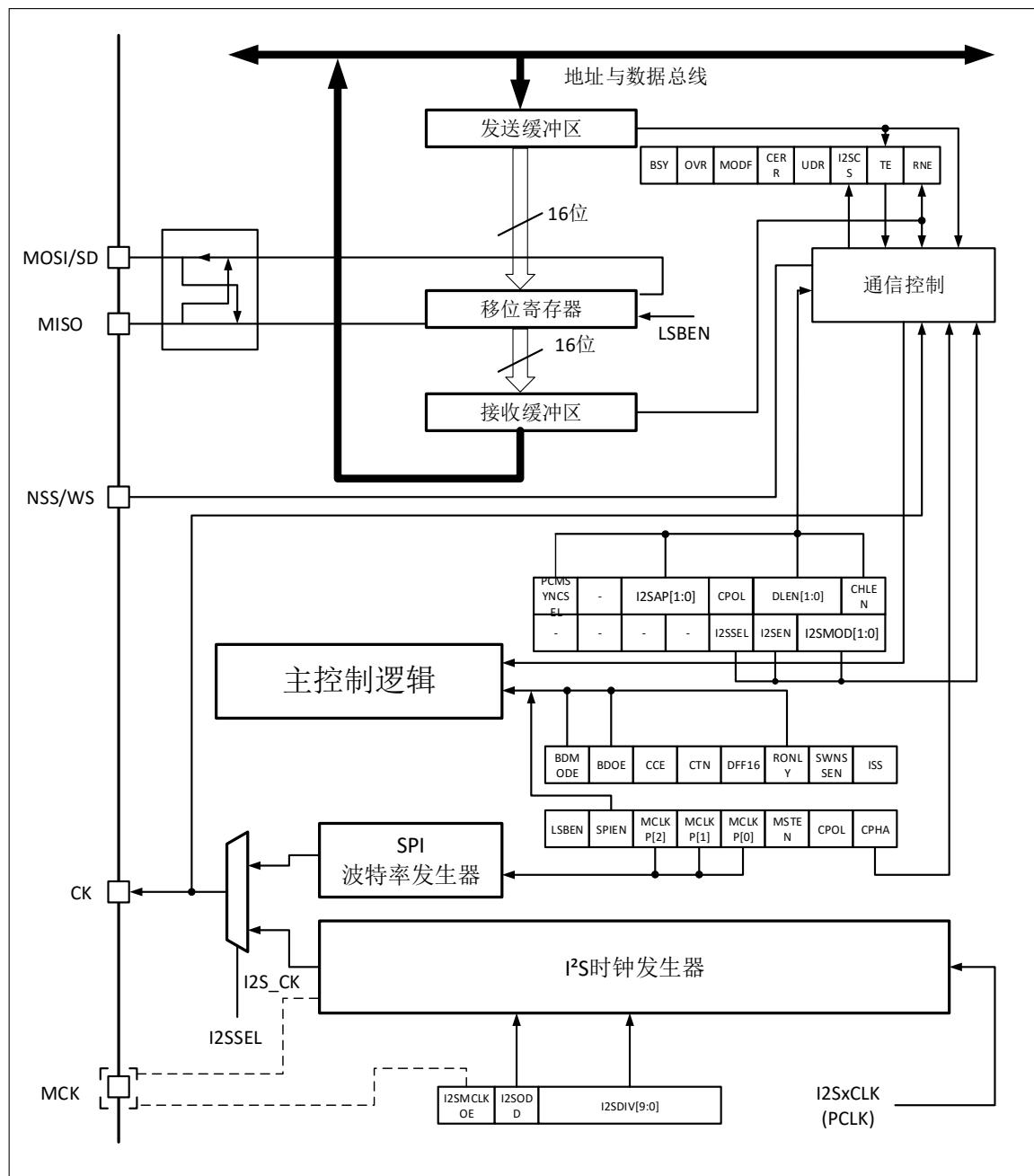
## 17.3.2 I<sup>2</sup>S功能描述

所有的 AT32F421 产品均支持 I<sup>2</sup>S 音频协议。

### 17.3.2.1 I<sup>2</sup>S功能描述

I<sup>2</sup>S 的框图如下图所示：

图表 17-13 I<sup>2</sup>S框图



通过将寄存器 SPI\_I2SCTRL 的 I2SEL 位置为'1'，即可使能 I<sup>2</sup>S 功能。此时，可以把 SPI 模块用作 I<sup>2</sup>S 音频接口。I<sup>2</sup>S 接口与 SPI 接口使用大致相同的引脚、标志和中断。

I<sup>2</sup>S 与 SPI 共用 3 个引脚：

- SD：串行数据（映射至 MOSI 引脚），用来发送和接收 2 路时分复用通道的数据；
- WS：字选（映射至 NSS 引脚），主模式下作为数据控制信号输出，从模式下作为输入；
- CK：串行时钟（映射至 SCK 引脚），主模式下作为时钟信号输出，从模式下作为输入。

在某些外部音频设备需要主时钟时，可以另有一个附加引脚输出时钟：

- MCLK：主时钟（独立映射），在 I<sup>2</sup>S 配置为主模式，寄存器 SPI\_I2SCLKP 的 I2SMCLKOE 位为'1'时，作为输出额外的时钟信号引脚使用。输出时钟信号的频率预先设置为  $256 \times F_s$ ，其中  $F_s$  是音频信号的采样频率。

设置成主模式时，I<sup>2</sup>S 使用自身的时钟发生器来产生通信用的时钟信号。这个时钟发生器也是主时钟输出的时钟源。I<sup>2</sup>S 模式下有 2 个额外的寄存器，一个是与时钟发生器配置相关的寄存器 SPI\_I2SCLKP，另一个是 I<sup>2</sup>S 通用配置寄存器 SPI\_I2SCTRL（可设置音频标准、从/主模式、数据格式、数据包帧、时钟极性等参数）。

在 I<sup>2</sup>S 模式下不使用寄存器 SPI\_CTRL1 和所有的 CRC 寄存器。同样，I<sup>2</sup>S 模式下也不使用寄存器 SPI\_CTRL2 的 NSSOE 位，和寄存器 SPI\_STS 的 MODF 位和 CERR 位。I<sup>2</sup>S 使用与 SPI 相同的寄存器 SPI\_DT 用作 16 位宽模式数据传输。

### 17.3.2.2 支持的音频协议

三线总线支持 2 个声道上音频数据的时分复用：左声道和右声道，但是只有一个 16 位寄存器用作发送或接收。因此，软件必须在对数据寄存器写入数据时，根据当前传输中的声道写入相应数据；同样，在读取寄存器数据时，通过检查寄存器 SPI\_STS 的 I2SCS 位来判明接收到的数据属于哪个声道。左声道总是先于右声道发送数据（I<sup>2</sup>SCS 位在 PCM 协议下无意义）。

有四种可用的数据和包帧组合。可以通过以下四种数据格式发送数据：

- 16位数据打包进16位帧
- 16位数据打包进32位帧
- 24位数据打包进32位帧
- 32位数据打包进32位帧

在使用 16 位数据扩展到 32 位帧时，前 16 位（MSB）是有意义的数据，后 16 位（LSB）被强制为 0，该操作不需要软件干预，也不需要有 DMA 请求（仅需要一次读/写操作）。

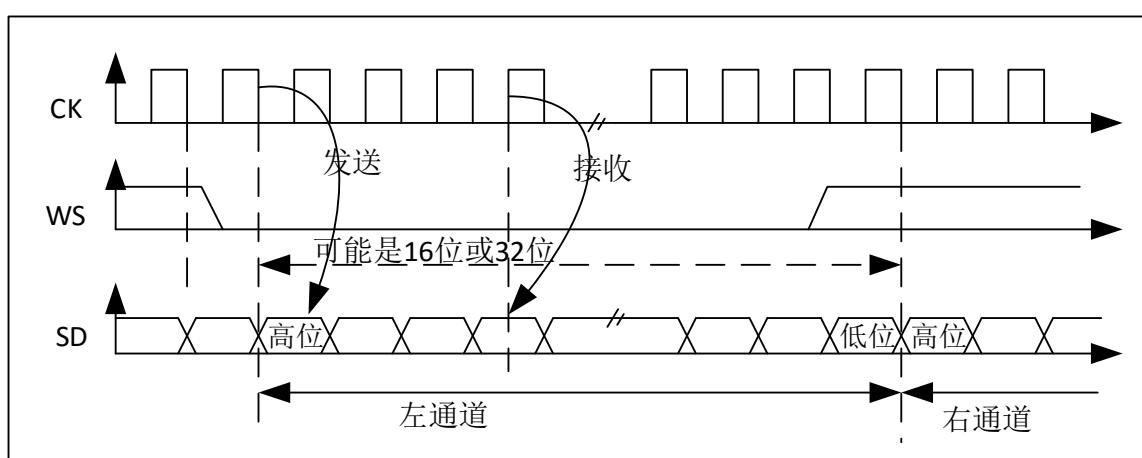
24 位和 32 位数据帧需要 CPU 对寄存器 SPI\_DT 进行 2 次读或写操作，在使用 DMA 时，需要 2 次 DMA 传输。对于 24 位数据，扩展到 32 位后，最低 8 位由硬件置 0。对于所有的数据格式和通讯标准，总是先发送最高位（MSB）。

I<sup>2</sup>S 接口支持四种音频标准，可以通过设置寄存器 SPI\_I2SCTRL 的 I2SAP[1: 0]位和 PCMSYNCSEL 位来选择。

#### I<sup>2</sup>S 飞利浦标准

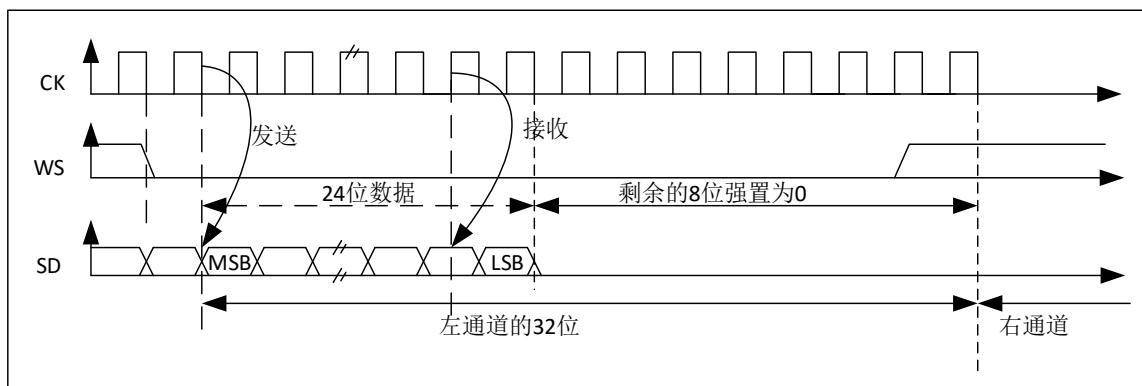
在此标准下，引脚 WS 用来指示正在发送的数据属于哪个声道。在发送第一位数据（MSB）前 1 个时钟周期，该引脚即为有效。

图表 17-14 I<sup>2</sup>S 飞利浦协议波形（16/32位全精度，CPOL=0）



发送方在时钟信号（CK）的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在时钟信号的下降沿变化。

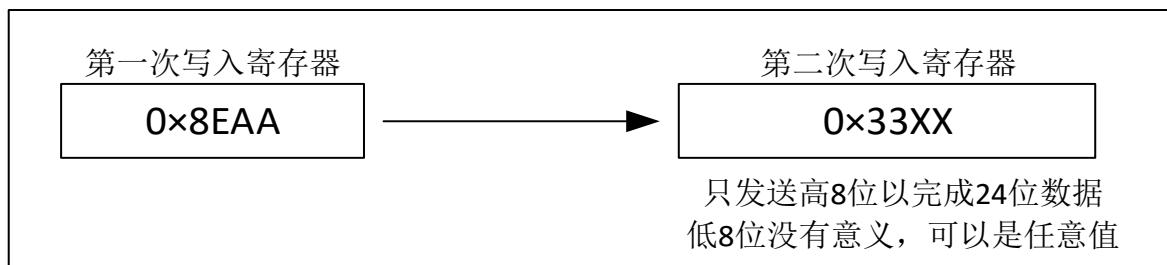
图表 17-15 I<sup>2</sup>S 飞利浦协议标准波形（24位帧，CPOL=0）



此模式需要对寄存器 SPI\_DT 进行 2 次读或写操作。

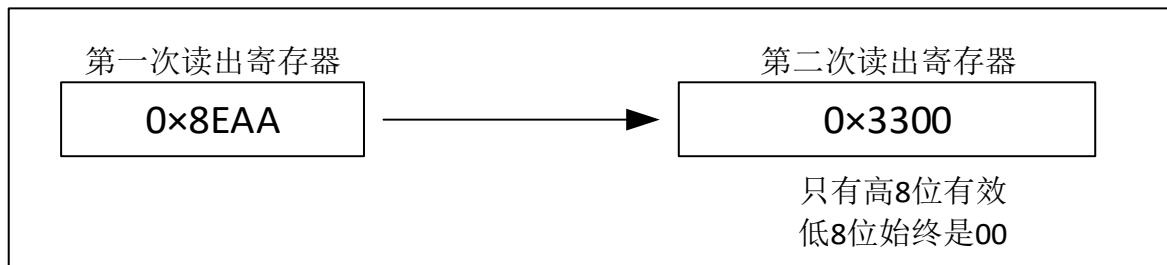
- 在发送模式下：如果需要发送 0x8EAA33（24位）：

图表 17-16 发送 0x8EAA33

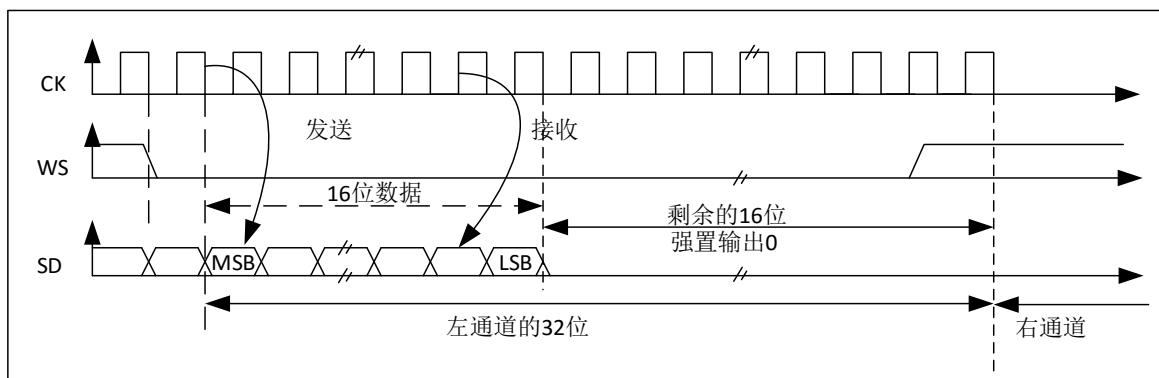


- 在接收模式下：如果接收 0x8EAA33：

图表 17-17 接收 0x8EAA33



图表 17-18 I<sup>2</sup>S 飞利浦协议标准波形（16位扩展至32位包帧，CPOL=0）



在 I<sup>2</sup>S 配置阶段，如果选择将 16 位数据扩展到 32 位声道帧，只需要访问一次寄存器 SPI\_DT。用来扩展到 32 位的低 16 位被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3（扩展到 32 位是 0x76A30000），需要的操作如下图所示。

图表 17-19 示例



在发送时需要将 **MSB** 写入寄存器 SPI\_DT；标志位 **TE** 为'1' 表示可以写入新的数据，如果允许了相应的中断，则可以产生中断。发送是由硬件完成的，即使还未发送出后 16 位的 0x0000，也会设置 **TE** 并产生相应的中断。

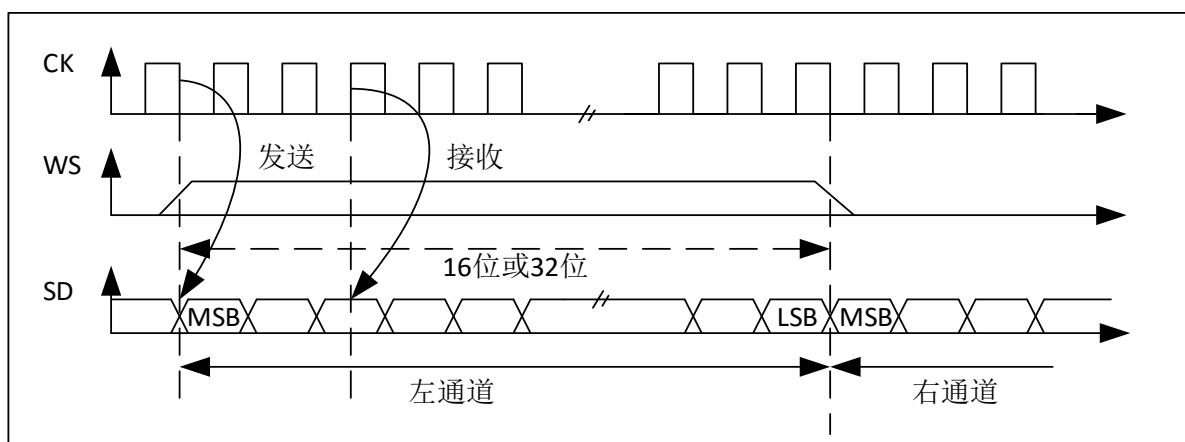
接收时，每次收到高 16 位半字（**MSB**）后，标志位 **RNE** 置'1'，如果允许了相应的中断，则可以产生中断。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

### MSB 对齐标准

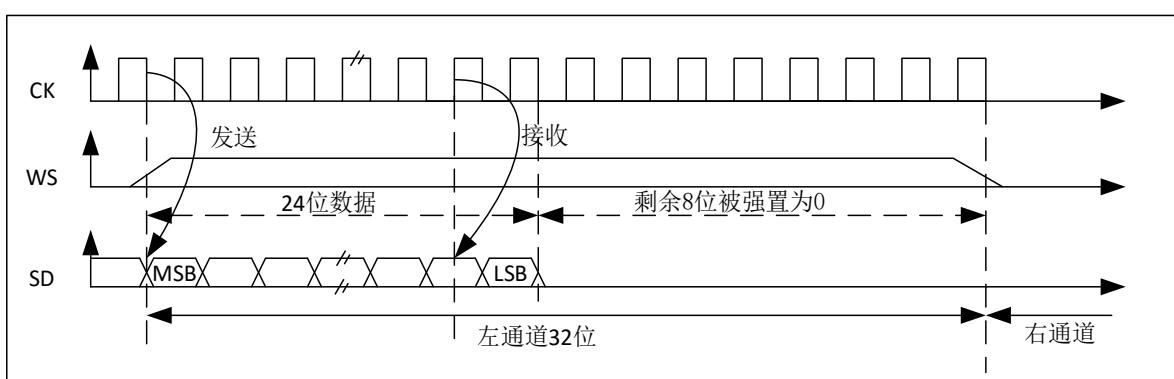
在此标准下，WS 信号和第一个数据位，即最高位（**MSB**）同时产生。

图表 17-20 MSB对齐16位或32位全精度，CPOL=0

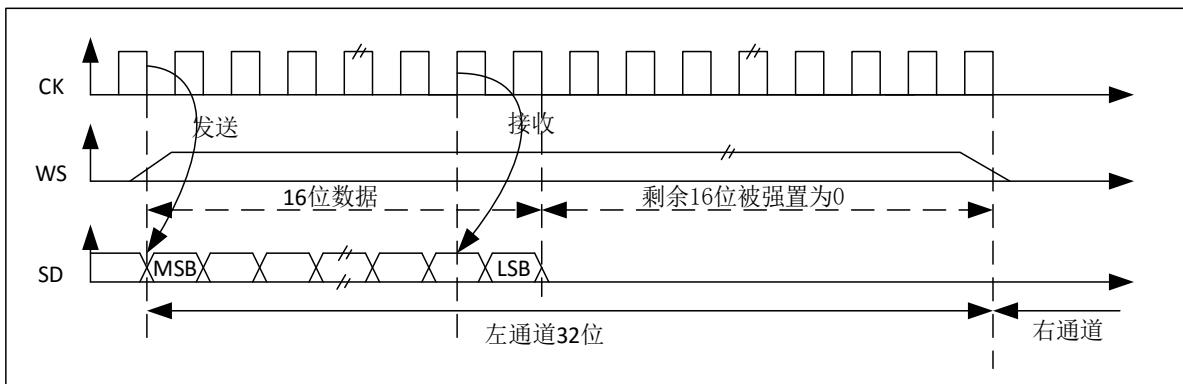


发送方在时钟信号的下降沿改变数据；接收方是在上升沿读取数据。

图表 17-21 MSB对齐24位数据，CPOL=0



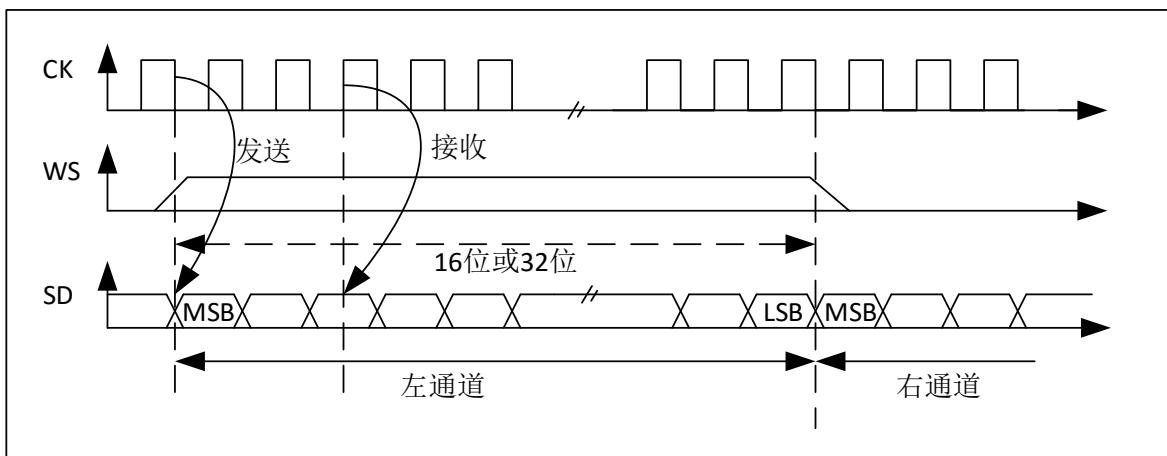
图表 17-22 MSB对齐16位数据扩展到32位包帧，CPOL=0



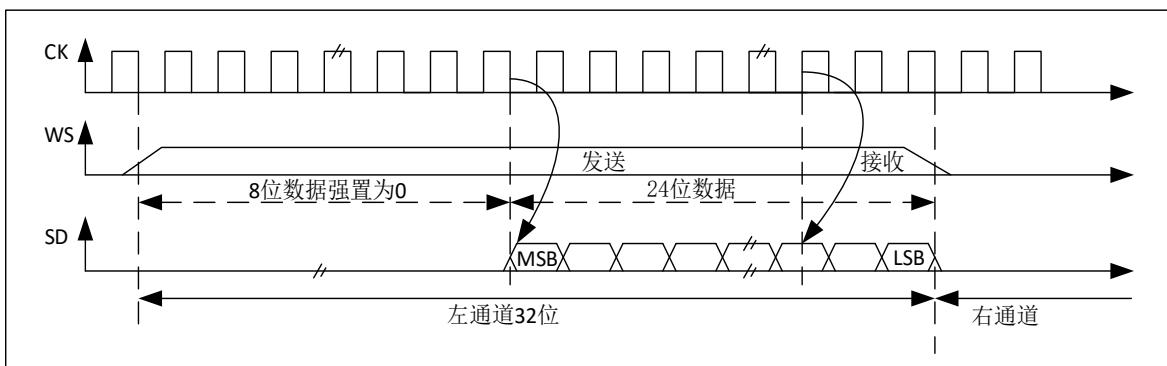
### LSB 对齐标准

此标准与 MSB 对齐标准类似（在 16 位或 32 位全精度帧格式下无区别）。

图表 17-23 LSB对齐16位或32位全精度，CPOL=0



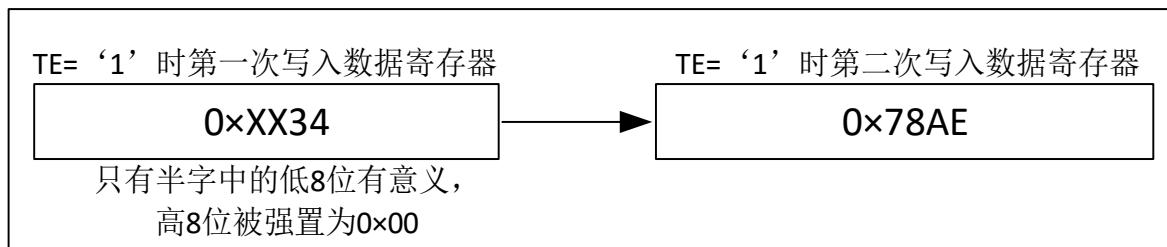
图表 17-24 LSB对齐24位数据，CPOL=0



- 在发送模式下

如果要发送数据 0x3478AE，需要通过软件或者 DMA 对寄存器 SPI\_DT 进行 2 次写操作。操作流程如下图所示。

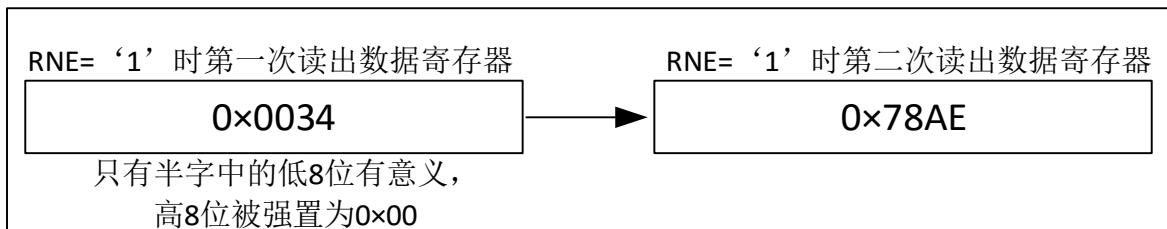
图表 17-25 要求发送0x3478AE的操作



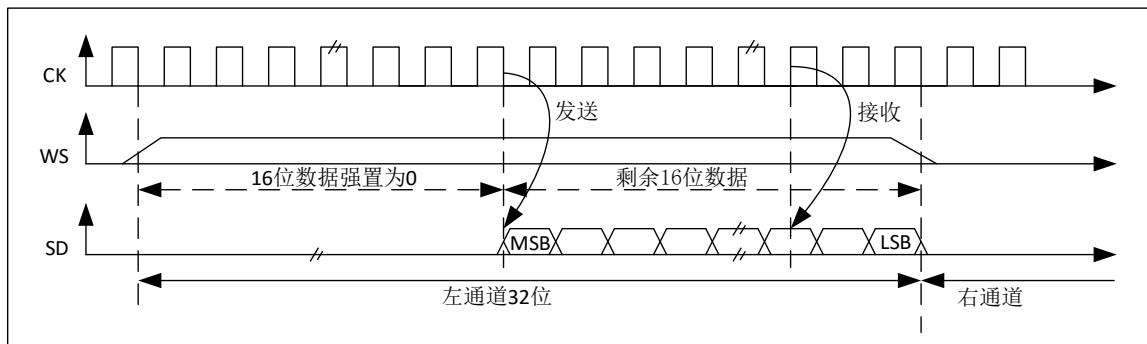
- 在接收模式下

如果要接收数据 0x3478AE，需要在 2 个连续的 RNE 事件发生时，分别对寄存器 SPI\_DT 进行 1 次读操作。

图表 17-26 要求接收0x3478AE的操作



图表 17-27 LSB对齐16位数据扩展到32位包帧，CPOL=0



在 I<sup>2</sup>S 配置阶段，如果选择将 16 位数据扩展到 32 声道帧，只需要访问一次寄存器 SPI\_DT。此时，扩展到 32 位后的高半字（16 位 MSB）被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3（扩展到 32 位是 0x000076A3），需要的操作如下图所示。

图表 17-28 示例

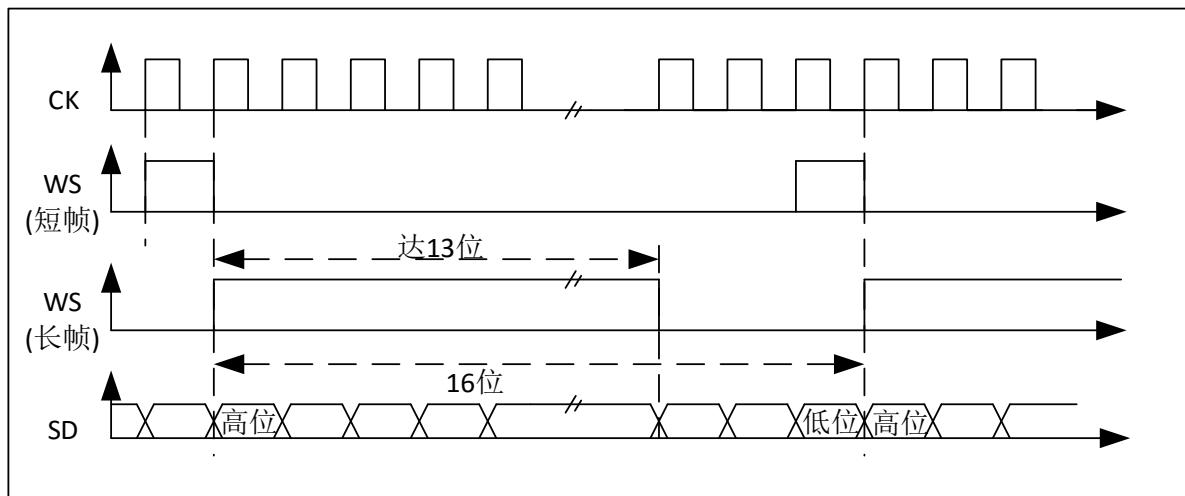


在发送时，如果 TE 为‘1’，用户需要写入待发送的数据（即 0x76A3）。用来扩展到 32 位的 0x0000 部分由硬件首先发送出去，一旦有效数据开始从 SD 引脚送出，即发生下一次 TE 事件。在接收时，一旦接收到有效数据（而不是 0x0000 部分），即发生 RNE 事件。这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

### PCM 标准

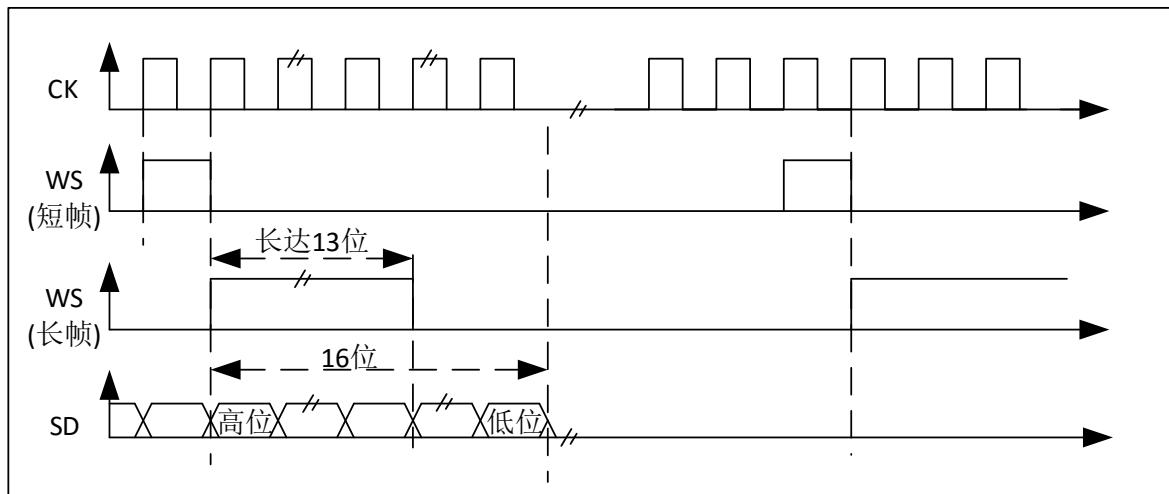
在 PCM 标准下，不存在声道选择的信息。PCM 标准有 2 种可用的帧结构，短帧或者长帧，可以通过设置寄存器 SPI\_I2SCTRL 的 PCMSYNCSEL 位来选择。

图表 17-29 PCM标准波形（16位）



对于长帧，主模式下，用来同步的 WS 信号有效的时间固定为 13 位。对于短帧，用来同步的 WS 信号长度只有 1 位。

图表 17-30 PCM标准波形（16位扩展到32位包帧）



**注意：**无论哪种模式（主或从）、哪种同步方式（短帧或长帧），连续的 2 帧数据之间和 2 个同步信号之间的时间差，（即使是从模式）需要通过设置 SPI\_I2SCTRL 寄存器的 DLEN 位和 CHLEN 位来确定。

### 17.3.2.3 时钟发生器

I<sup>2</sup>S 的比特率即确定了在 I<sup>2</sup>S 数据线上的数据流和 I<sup>2</sup>S 的时钟信号频率。

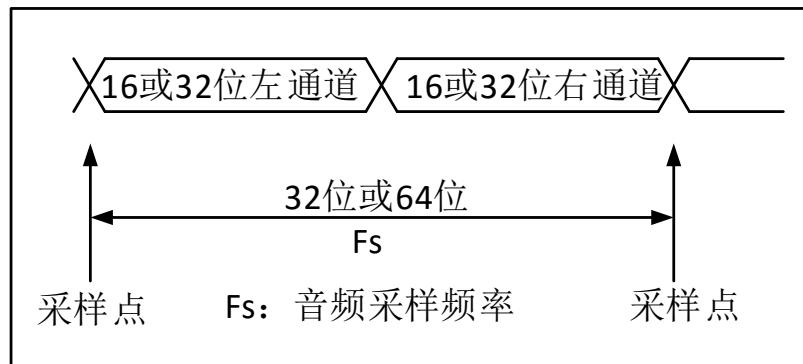
I<sup>2</sup>S 比特率=每个声道的比特数×声道数目×音频采样频率

对于一个具有左右声道和 16 位音频信号，I<sup>2</sup>S 比特率计算如下：

I<sup>2</sup>S 比特率=16×2×Fs

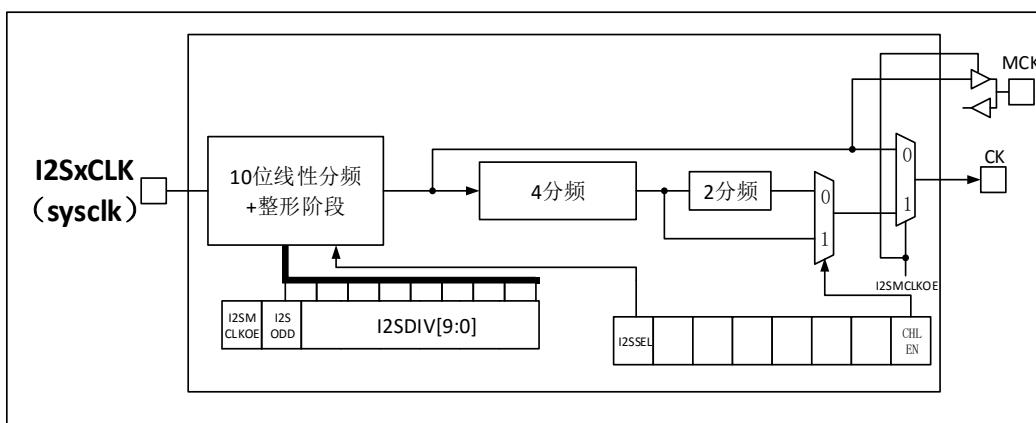
如果包长为 32 位，则有：I<sup>2</sup>S 比特率=32×2×Fs

图表 17-31 音频采样频率定义



在主模式下，为了获得需要的音频频率，需要正确地对线性分频器进行设置。

图表 17-32 I<sup>2</sup>S时钟发生器结构



上图中 I2SxCLK 的时钟源是系统时钟（即驱动 AHB 时钟的 HSI、HSE 或 PLL）。

音频的采样频率可以是 192KHz、96kHz、48kHz、44.1kHz、32kHz、22.05kHz、16kHz、11.025kHz 或者 8kHz（或任何此范围内的数值）。为了获得需要的频率，需按照以下公式设置线性分频器：

当需要生成主时钟时（寄存器 SPI\_I2SCLKP 的 I2SMCLKOE 位为‘1’）：

$$\text{声道的帧长为 16 位时, } \text{Fs} = \text{I2SxCLK} / [(16 * 2) * ((2 * \text{I2SDIV}) + \text{I2SODD}) * 8]$$

$$\text{声道的帧长为 32 位时, } \text{Fs} = \text{I2SxCLK} / [(32 * 2) * ((2 * \text{I2SDIV}) + \text{I2SODD}) * 4]$$

当关闭主时钟时（I2SMCLKOE 位为‘0’）：

$$\text{声道的帧长为 16 位时, } \text{Fs} = \text{I2SxCLK} / [(16 * 2) * ((2 * \text{I2SDIV}) + \text{I2SODD})]$$

$$\text{声道的帧长为 32 位时, } \text{Fs} = \text{I2SxCLK} / [(32 * 2) * ((2 * \text{I2SDIV}) + \text{I2SODD})]$$

下面 2 张表给出了不同时钟配置时，精确参数的例子。

**注意：** 可以使用其它配置以达到优化时钟精确度的目的。

表格 17-2 使用系统时钟得到精确的音频频率

SysCLK (MHz)	MCLK	TargetFs (Hz)	16bit				32bit			
			I2sDI V	I2S _O DD	RealFs	Error	I2sDI V	I2S _O DD	RealFs	Error
120	NO	192000	10	0	187500	2.34%	10	0	187500	2.34%
120	No	96000	19	1	96153	0.16%	10	0	93750	2.34%
120	No	48000	39	0	48076	0.16%	19	1	48076	0.16%
120	No	44100	42	1	44117	0.04%	21	1	43604	1.12%
120	No	32000	58	1	32051	0.16%	29	1	31779	0.69%
120	No	22050	85	0	22058	0.04%	42	1	22058	0.04%
120	No	16000	117	0	16025	0.16%	58	1	16025	0.16%

120	No	11025	170	0	11029	0.04%	85	0	11029	0.04%
120	No	8000	234	1	7995	0.05%	117	1	8012	0.16%
120	Yes	96000	2	1	93750	2.34%	2	1	937500	2.34%
120	Yes	48000	5	0	46875	2.34%	5	0	46875	2.34%
120	Yes	44100	5	1	42613	3.37%	5	1	42613	3.37%
120	Yes	32000	7	1	31250	2.34%	7	1	31250	2.34%
120	Yes	22050	10	1	22321	1.23%	10	1	22321	1.23%
120	Yes	16000	14	1	16163	1.02%	14	1	16163	1.02%
120	Yes	11025	21	1	10901	1.023%	21	1	10901	1.023%
120	Yes	8000	29	1	7944	0.68%	29	1	7944	0.68%
108	No	192000	9	0	187500.0	2.34%	4	1	187500.0	2.34%
108	No	96000	17	1	96428.57	0.446%	9	0	93750	2.34%
108	No	48000	35	0	48214.29	0.446%	17	1	48214.29	0.446%
108	No	44100	38	1	43831.11	0.61%	19	0	44407.89	0.698%
108	No	32000	52	1	32142.86	0.446%	26	1	31839.62	0.501%
108	No	22050	76	1	22058.82	0.04%	38	1	21915.58	0.609%
108	No	16000	110	1	15995.26	0.029%	55	0	16071.43	0.446%
108	No	11025	153	0	11029.41	0.04%	76	1	11029.41	0.04%
108	No	8000	221	0	7997.63	0.029%	105	1	7997.63	0.029%
108	Yes	96000	2	0	105468.8	9.86%	2	0	105468.8	9.86%
108	Yes	48000	4	1	46875.0	2.34%	4	1	46875.0	2.34%
108	Yes	44100	5	0	42187.5	2.34%	5	0	42187.5	2.34%
108	Yes	32000	6	0	32451.92	1.41%	6	0	32451.92	1.41%
108	Yes	22050	9	1	22203.95	0.698%	9	1	22203.95	0.698%
108	Yes	16000	13	0	16225.96	1.41%	13	0	16225.96	1.41%
108	Yes	11025	19	0	11101.97	0.698%	17	1	11101.97	0.698%
108	Yes	8000	26	1	7959.906	0.501%	26	1	7959.906	0.501%
72	No	192000	6	0	187500	2.34%	3	0	187500	2.34%
72	No	96000	11	1	97826.09	1.90%	6	0	93750	2.34%
72	No	48000	32	1	34615.38	27.88%	11	1	48913.04	1.90%
72	No	44100	25	1	44117.65	0.04%	13	0	43269.23	1.88%
72	No	32000	35	0	32142.86	0.45%	17	1	32142.86	0.45%
72	No	22050	51	0	22058.82	0.04%	25	1	22058.82	0.04%
72	No	16000	70	1	15957.45	0.27%	35	0	16071.43	0.45%
72	No	11025	102	0	11029.41	0.04%	51	0	11029.41	0.04%
72	No	8000	140	1	8007.117	0.09%	70	1	7978.723	0.27%
72	Yes	96000	2	0	70312.5	26.76%	2	0	70312.5	26.76%
72	Yes	48000	3	0	46875	2.34%	3	0	46875	2.34%
72	Yes	44100	3	0	46875	6.29%	3	0	46875	6.29%
72	Yes	32000	4	1	31250	2.34%	4	1	31250	2.34%
72	Yes	22050	6	1	21634.62	1.88%	6	1	21634.62	1.88%
72	Yes	16000	9	0	15625	2.34%	9	0	15625	2.34%
72	Yes	11025	13	0	10817.31	1.88%	13	0	10817.31	1.88%
72	Yes	8000	17	1	8035.714	0.45%	17	1	8035.714	0.45%

#### 17.3.2.4 I<sup>2</sup>S主模式

设置 I<sup>2</sup>S 工作在主模式，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 SPI\_I2SCLKP 的 I2SMCLKOE 位来选择输出或者不输出主时钟（MCLK）。

## 流程

1. 设置寄存器 SPI\_I2SCLKP 的 I2SDIV[9: 0]定义与音频采样频率相符的串行时钟波特率。同时也要定义寄存器 SPI\_I2SCLKP 的 I2SODD 位。
2. 设置 CPOL 位定义通信用时钟在空闲时的电平状态。如果需要向外部的 ADC 音频器件提供主时钟 MCLK，将寄存器 SPI\_I2SCLKP 的 I2SMCLKOE 位置为'1'。（按照不同的 MCLK 输出状态，计算 I2SDIV 和 I2SODD 的值，详见 [17.4.3 节](#)）。
3. 设置寄存器 SPI\_I2SCTRL 的 I2SEL 位为'1'激活 I<sup>2</sup>S 功能，设置 I2SAP[1: 0]和 PCMSYNCSEL 位选择所用的 I<sup>2</sup>S 标准，设置 CHLEN 选择每个声道的数据位数。还要设置寄存器 SPI\_I2SCTRL 的 I2SMOD[1: 0]选择 I<sup>2</sup>S 主模式和方向（发送端还是接收端）。
4. 如果需要，可以通过设置寄存器 SPI\_CTRL2 来打开所需的中断功能和 DMA 功能。
5. 必须将寄存器 SPI\_I2SCTRL 的 I2SEN 位置为'1'。
6. 引脚 WS 和 CK 需要配置为输出模式。如果寄存器 SPI\_I2SCLKP 的 I2SMCLKOE 位为'1'，引脚 MCLK 也要配置成输出模式。

## 发送流程

当写入 1 个半字（16 位）的数据至发送缓存，发送流程开始。假设第一个写入发送缓存的数据对应的是左声道数据。当数据从发送缓存移到移位寄存器时，标志位 TE 置'1'，这时，要把对应右声道的数据写入发送缓存。标志位 I2SCS 提示了目前待传输的数据对应哪个声道。标志位 I2SCS 的值在 TE 为'1'时更新，因此它在 TE 为'1'时有意义。

在先左声道后右声道的数据都传输完成后，才能被认为是一个完整的数据帧。不可以只传输部分数据帧，如仅有左声道的数据。

当发出第一位数据的同时，半字数据被并行地传送至 16 位移位寄存器，然后后面的位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓存移至移位寄存器时，标志位 TE 置为'1'，如果寄存器 SPI\_CTRL2 的 TEIE 位为'1'，则产生中断。

写入数据的操作取决于所选择的 I<sup>2</sup>S 标准，详见 [17.4.2 节](#)。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI\_DT 写入下一个要传输的数据。

当写入最后一个数据，等待 TE=1 及 BSY=0 后再清除 I2SEN 位关闭 I<sup>2</sup>S 功能。

## 接收流程

接收流程的配置步骤除了第 3 点外，与发送流程的一致（参见前述的“发送流程”），需要通过配置 I2SMOD[1: 0]来选择主接收模式。无论何种数据和声道长度，音频数据总是以 16 位包的形式接收。即每次填满接收缓存后，标志位 RNE 置'1'，如果寄存器 SPI\_CTRL2 的 RNEIE 位为'1'，则产生中断。根据配置的数据和声道长度，收到左声道或右声道的数据会需要 1 次或者 2 次把数据传送到接收缓存的过程。

对寄存器 SPI\_DT 进行读操作即可清除 RNE 标志位。

每次接收以后即更新 I2SCS。它的值取决于 I<sup>2</sup>S 单元产生的 WS 信号。

读取数据的操作取决于所选择的 I<sup>2</sup>S 标准，详见 [17.4.2 节](#)。

如果前一个接收到的数据还没有被读取，又接收到新数据，即发生上溢，标志位 OVR 被置为'1'，如果寄存器 SPI\_CTRL2 的 ERRIE 位为'1'，则产生中断，表示发生了错误。

若要关闭 I<sup>2</sup>S 功能，需要执行特别的操作，以保证 I<sup>2</sup>S 模块可以正常地完成传输周期而不会开始新的数据传输。操作过程与数据配置和通道长度、以及音频协议的模式相关：

- 16位数据扩展到32位通道长度（DLEN=00并且CHLEN=1），使用 LSB（低位）对齐模式（I2SAP=10）
  - 等待倒数第二个（n-1）RNE=1；
  - 等待 17 个 I<sup>2</sup>S 时钟周期（使用软件延迟）；
  - 关闭 I<sup>2</sup>S（I2SEN=0）。
- 16位数据扩展到32位通道长度（DLEN=00并且CHLEN=1），使用 MSB（高位）对齐、I<sup>2</sup>S 或 PCM 模式（分别为 I2SAP=00，I2SAP=01 或 I2SAP=11）
  - 等待最后一个 RNE=1；
  - 等待 1 个 I<sup>2</sup>S 时钟周期（使用软件延迟）；

- c) 关闭 I<sup>2</sup>S (I2SEN=0)。
- 所有其它 DLEN 和 CHLEN 的组合, I2SAP 选择的任意音频模式, 使用下述方式关闭 I<sup>2</sup>S:
  - a) 等待倒数第二个 (n-1) RNE=1;
  - b) 等待一个 I<sup>2</sup>S 时钟周期 (使用软件延迟);
  - c) 关闭 I<sup>2</sup>S (I2SEN=0)。

**注意:** 在传输期间 BSY 标志始终为低。

### 17.3.2.5 I<sup>2</sup>S 从模式

在从模式下, I<sup>2</sup>S 可以设置成发送和接收模式。从模式的配置方式基本遵循和配置主模式一样的流程。在从模式下, 不需要 I<sup>2</sup>S 接口提供时钟。时钟信号和 WS 信号都由外部主 I<sup>2</sup>S 设备提供, 连接到相应的引脚上。因此用户无需配置时钟。

配置步骤列举如下:

1. 设置寄存器 SPI\_I2SCTRL 的 I2SSEL 位激活 I<sup>2</sup>S 功能; 设置 I2SAP[1: 0]来选择所用的 I<sup>2</sup>S 标准; 设置 DLEN[1: 0]选择数据的比特数; 设置 CHLEN 选择每个声道的数据位数。设置寄存器 SPI\_I2SCTRL 的 I2SMOD[1: 0]选择 I<sup>2</sup>S 从模式的数据方向 (发送端还是接收端)。
2. 根据需要, 设置寄存器 SPI\_CTRL2 打开所需的中断功能和 DMA 功能。
3. 必须设置寄存器 SPI\_I2SCTRL 的 I2SEN 位为'1'。

#### 发送流程

当外部主设备发送时钟信号, 并且当 NSS\_WS 信号请求传输数据时, 发送流程开始。必须先使能从设备, 并且写入 I<sup>2</sup>S 数据寄存器之后, 外部主设备才能开始通信。

对于 I<sup>2</sup>S 的 MSB 对齐和 LSB 对齐模式, 第一个写入数据寄存器的数据项对应左声道的数据。当开始通信时, 数据从发送缓冲器传送到移位寄存器, 然后标志位 TE 置为'1'; 这时, 要把对应右声道的数据项写入 I<sup>2</sup>S 数据寄存器。

标志位 I2SCS 提示了目前待传输的数据对应哪个声道。与主模式的发送流程相比, 在从模式中, I2SCS 取决于来自外部主 I<sup>2</sup>S 的 WS 信号。这意味着从 I<sup>2</sup>S 在接收到主端生成的时钟信号之前, 就要准备好第一个要发送的数据。对于 I<sup>2</sup>S 的 MSB 对齐和 LSB 对齐模式, WS 信号为'1'表示先发送左声道。

**注意:** 设置 I2SEN 位为'1'的时间, 应当比 CK 引脚上的主 I<sup>2</sup>S 时钟信号早至少 2 个 PCLK 时钟周期。

当发出第一位数据的时候, 半字数据并行地通过 I<sup>2</sup>S 内部总线传输至 16 位移位寄存器, 然后其它位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓冲器传送至移位寄存器时, 标志位 TE 置'1', 如果寄存器 SPI\_CTRL2 的 TEIE 位为'1', 则产生中断。

**注意:** 在对发送缓冲器写入数据前, 要确认标志位 TE 为'1'。

写入数据的操作取决于所选中的 I<sup>2</sup>S 标准, 详见 [17.4.2 节](#)。为了保证连续的音频数据传输, 建议在当前传输完成之前, 对寄存器 SPI\_DT 写入下一个要传输的数据。如果在代表下一个数据传输的第一个时钟边沿到达之前, 新的数据仍然没有写入寄存器 SPI\_DT, 下溢标志位会置'1', 并可能产生中断; 它指示软件发送数据错误。如果寄存器 SPI\_CTRL2 的 ERRIE 位为'1', 在寄存器 SPI\_STS 的标志位 UDR 为高时, 就会产生中断。建议在这时关闭 I<sup>2</sup>S, 然后重新从左声道开始发送数据。

当写入最后一个数据, 等待 TE=1 及 BSY=0 后再清除 I2SEN 位关闭 I<sup>2</sup>S 功能。

#### 接收流程

配置步驟除了第 1 点外, 与发送流程一致。需要通过配置 I2SMOD[1: 0]来选择主接收模式。无论何种数据和声道长度, 音频数据总是以 16 位包的形式接收, 即每次填满接收缓存, 标志位 RNE 置'1', 如果寄存器 SPI\_CTRL2 的 RNEIE 位为'1', 则产生中断。按照不同的数据和声道长度设置, 收到左声道或者右声道数据会需要 1 次或者 2 次传输数据至接收缓冲器的过程。

每次接收到数据 (将要从 SPI\_DT 读出) 以后即更新 I2SCS, 它对应 I<sup>2</sup>S 单元产生的 WS 信号。

读取 SPI\_DT 寄存器, 将清除 RNE 位。

读取数据的操作取决于所选中的 I<sup>2</sup>S 标准, 详见 [17.4.2 节](#)。

在还没有读出前一个接收到的数据, 又接收到新数据时, 即产生上溢, 并设置标志位 OVR 为'1'; 如果寄存器 SPI\_CTRL2 的 ERRIE 位为'1', 则产生中断, 指示发生了错误。

要关闭 I<sup>2</sup>S 功能时, 需要在接收到最后一次 RNE=1 时将 I2SEN 位清'0'。

注意：外部主 I<sup>2</sup>S 器件需要有通过音频声道发送/接收 16 位或 32 位数据包的功能。

### 17.3.2.6 状态标志位

有 3 个状态标志位供用户监控 I<sup>2</sup>S 总线的状态。

#### 忙标志位（BSY）

BSY 标志由硬件设置与清除（写入此位无效果），该标志位指示 I<sup>2</sup>S 通信层的状态。

该位为'1'时表明 I<sup>2</sup>S 通讯正在进行中，但有一个例外：主接收模式（I2SMOD=11）下，在接收期间 BSY 标志始终为低。

在软件要关闭 SPI 模块之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输，因此需要严格按照下述过程执行。

当传输开始时，BSY 标志被置为'1'，除非 I<sup>2</sup>S 模块处于主接收模式。

下述情况时，该标志位被清除：

- 当传输结束时（除了主发送模式，这种模式下通信是连续的）；
- 当关闭 I<sup>2</sup>S 模块时。

当通信是连续的时候：

- 在主发送模式时，整个传输期间，BSY 标志始终为高；
- 在从模式时，每个数据项传输之间，BSY 标志在 1 个 I<sup>2</sup>S 时钟周期内变低。

注意：不要使用 BSY 标志处理每一个数据项的发送和接收，最好使用 TE 和 RNE 标志。

#### 发送缓存空标志位（TE）

该标志位为'1'表示发送缓冲器为空，可以对发送缓冲器写入新的待发送数据。在发送缓冲器中已有数据时，标志位清'0'。在 I<sup>2</sup>S 被关闭时（I2SEN 位为'0'），该标志位也为'0'。

#### 接收缓存非空标志位（RNE）

该标志位置'1'表示在接收缓存里有接收到的有效数据。在读取寄存器 SPI\_DT 时，该位清'0'。

#### 声道标志位（I2SCS）

在发送模式下，该标志位在 TE 为高时刷新，指示从 SD 引脚上发送的数据所在的声音。如果在从发送模式下发生了下溢错误，该标志位的值无效，在重新开始通讯前需要把 I<sup>2</sup>S 关闭再打开。

在接收模式下，该标志位在寄存器 SPI\_DT 接收到数据时刷新，指示接收到的数据所在的声音。

注意：如果发生错误（如上溢 OVR），该标志位无意义，需要将 I<sup>2</sup>S 关闭再打开（同时，如果必要修改 I<sup>2</sup>S 的配置）。

在 PCM 标准下，无论短帧格式还是长帧格式，这个标志位都没有意义。

如果寄存器 SPI\_STS 的标志位 OVR 或 UDR 为'1'，且寄存器 SPI\_CTRL2 的 ERRIE 位为'1'，则会产生中断。（中断源已经被清除后）可以通过读寄存器 SPI\_STS 来清除中断标志。

### 17.3.2.7 错误标志位

I<sup>2</sup>S 单元有 2 个错误标志位。

#### 下溢标志位（UDR）

在从发送模式下，如果数据传输的第一个时钟边沿到达时，新的数据仍然没有写入 SPI\_DT 寄存器，该标志位会被置'1'。在寄存器 SPI\_I2SCTRL 的 I2SEL 位置'1'后，该标志位才有效。如果寄存器 SPI\_CTRL2 的 ERRIE 位为'1'，就会产生中断。

通过对寄存器 SPI\_STS 进行读操作来清除该标志位。

#### 上溢标志位（OVR）

如果还没有读出前一个接收到的数据时，又接收到新的数据，即产生上溢，该标志位置'1'，如果寄存器 SPI\_CTRL2 的 ERRIE 位为'1'，则产生中断指示发生了错误。

这时，接收缓存的内容，不会刷新为从发送设备送来的新数据。对寄存器 SPI\_DT 的读操作返回最后一个正确接收到的数据。其他所有在上溢发生后由发送设备发出的 16 位数据都会丢失。

通过先读寄存器 SPI\_STS 再读寄存器 SPI\_DT，来清除该标志位。

### 17.3.2.8 I<sup>2</sup>S 中断

下表列举了全部 I<sup>2</sup>S 中断。

表格 17-3 I<sup>2</sup>S中断请求

中断事件	事件标志位	使能标志位
发送缓冲器空标志位	TE	TEIE
接收缓冲器非空标志位	RNE	RNEIE
上溢标志位	OVR	ERRIE
下溢标志位	UDR	

### 17.3.2.9 DMA功能

DMA 的工作方式在 I<sup>2</sup>S 模式除了 CRC 功能不可用以外，与在 SPI 模式完全相同。

因为在 I<sup>2</sup>S 模式下没有数据传输保护系统。

## 17.4 SPI寄存器

可以用半字（16位）或字（32位）的方式操作这些外设寄存器。

表格 17-4 SPI寄存器列表及其复位值

0x1C	SPI_I2SCTR L	保留	I2SEL	I2SEN	I2SMOD[1: 0]	PCMSSYNCSEL 保留	I2SAP[1: 0]	CPOL	DLEN[1: 0]	CHLEN
			0	0	0		0			
0x20	SPI_I2SCLK P	保留	I2SDIV[9: 8]	I2SMCLKOE	I2SODD	I2SDIV[7: 0]	I2SAP[1: 0]	CPOL	DLEN[1: 0]	CHLEN
			0	0	0		0			

### 17.4.1 SPI控制寄存器1 (SPI\_CTRL1) (I<sup>2</sup>S模式下不使用)

地址偏移: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BD MO DE	BD OE	CCE	CTN	DFF 16	RO NLY	SW NSS EN	ISS	LSB EN	SPI EN	MCLKP[2: 0]	MST EN	CP OL	CPH A		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位 15	<b>BDMODE:</b> 双向数据模式使能 (Bidirectional data mode enable) 0: 选择“双线单向”模式; 1: 选择“单线双向”模式。 注: I <sup>2</sup> S 模式下不使用。
位 14	<b>BDOE:</b> 双向模式下的输出使能 (Output enable in bidirectional mode) 和 BDMODE 位一起决定在“单线双向”模式下数据的输出方向 0: 输出禁止 (只收模式); 1: 输出使能 (只发模式)。 这个“单线”数据线在主设备端为 MOSI 引脚, 在从设备端为 MISO 引脚。 注: I <sup>2</sup> S 模式下不使用。
位 13	<b>CCE:</b> 硬件 CRC 校验使能 (Hardware CRC calculation enable) 0: 禁止 CRC 计算; 1: 启动 CRC 计算。 注: 只有在禁止 SPI 时 (SPIEN=0), 才能写该位, 否则出错。该位只能在全双工模式下使用。 注: I <sup>2</sup> S 模式下不使用。
位 12	<b>CTN:</b> 下一个发送 CRC (Transmit CRC next) 0: 下一个发送的值来自发送缓冲区。 1: 下一个发送的值来自发送 CRC 寄存器。 注: 在 SPI_DT 寄存器写入最后一个数据后应马上设置该位。 注: I <sup>2</sup> S 模式下不使用。
位 11	<b>DFF16:</b> 数据帧格式 (Data frame format) 0: 使用 8 位数据帧格式进行发送/接收; 1: 使用 16 位数据帧格式进行发送/接收。 注: 只有当 SPI 禁止 (SPIEN=0) 时, 才能写该位, 否则出错。 注: I <sup>2</sup> S 模式下不使用。
位 10	<b>RONLY:</b> 只接收 (Receive only) 该位和 BDMODE 位一起决定在“双线双向”模式下的传输方向。在多个从设备的配置中, 在未被访问的从设备上该位被置 1, 使得只有被访问的从设备有输出, 从而不会造成数据线上数据冲突。 0: 全双工 (发送和接收); 1: 禁止输出 (只接收模式)。 注: I <sup>2</sup> S 模式下不使用。
位 9	<b>SWNSSEN:</b> 软件从设备管理 (Software slave management) 当 SWNSSEN 被置位时, NSS 引脚上的电平由 ISS 位的值决定。 0: 禁止软件从设备管理; 1: 启用软件从设备管理。 注: I <sup>2</sup> S 模式下不使用。
位 8	<b>ISS:</b> 内部从设备选择 (Internal slave select) 该位只在 SWNSSEN 位为'1'时有意义。它决定了 NSS 上的电平, 在 NSS 引脚上的 I/O 操作无效。 注: I <sup>2</sup> S 模式下不使用。

位 7	<b>LSBEN:</b> 帧格式 (Frame format) 0: 先发送 MSB; 1: 先发送 LSB。 注: 当通信在进行时不能改变该位的值。 注: I <sup>2</sup> S 模式下不使用。
位 6	<b>SPIEN:</b> SPI 使能 (SPI enable) 0: 禁止 SPI 设备; 1: 开启 SPI 设备。 注: I <sup>2</sup> S 模式下不使用。 注: 当关闭 SPI 设备时, 请按照 <a href="#">第 17.3.1.8 节</a> 的过程操作。
位 5: 3	<b>MCLKP[2: 0]:</b> 波特率控制 (Baudrate control) MCLKP[3]位在 SPI_CTRL2 寄存器, MCLKP[3:0]: 0000: f <sub>PCLK</sub> /2 0001: f <sub>PCLK</sub> /4 0010: f <sub>PCLK</sub> /8 0011: f <sub>PCLK</sub> /16 0100: f <sub>PCLK</sub> /32 0101: f <sub>PCLK</sub> /64 0110: f <sub>PCLK</sub> /128 0111: f <sub>PCLK</sub> /256 1000: f <sub>PCLK</sub> /512 1001: f <sub>PCLK</sub> /1024 当通信正在进行的时候, 不能修改这些位。 注: I <sup>2</sup> S 模式下不使用。
位 2	<b>MSTEN:</b> 主设备选择 (Master selection) 0: 配置为从设备; 1: 配置为主设备。 注: 当通信正在进行的时候, 不能修改该位。 注: I <sup>2</sup> S 模式下不使用。
位 1	<b>CPOL:</b> 时钟极性 (Clock polarity) 0: 空闲状态时, SCK 保持低电平; 1: 空闲状态时, SCK 保持高电平。 注: 当通信正在进行的时候, 不能修改该位。 注: I <sup>2</sup> S 模式下不使用。
位 0	<b>CPHA:</b> 时钟相位 (Clock phase) 0: 数据采样从第一个时钟边沿开始; 1: 数据采样从第二个时钟边沿开始。 注: 当通信正在进行的时候, 不能修改该位。 注: I <sup>2</sup> S 模式下不使用。

## 17.4.2 SPI控制寄存器2 (SPI\_CTRL2)

地址偏移: 0x04

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					保留		MCL KP[3]	TEIE	RNEI E	ERRI E		保留	NSS OE	DMA TEN	DMA REN
					res		rw	rw	rw	rw		res	rw	rw	rw

位 15: 9	保留位, 硬件强制为 0
位 8	<b>MCLKP[3]:</b> 波特率控制 (Baudrate control) 详见 MCLKP[2: 0]在 SPI_CTRL1 寄存器。
位 7	<b>TEIE:</b> 发送缓冲区空中断使能 (Rx buffer empty interrupt enable) 0: 禁止 TE 中断; 1: 允许 TE 中断, 当 TE 标志置位为'1'时产生中断请求。
位 6	<b>RNEIE:</b> 接收缓冲区非空中断使能 (RX buffer not empty interrupt enable) 0: 禁止 RNE 中断; 1: 允许 RNE 中断, 当 RNE 标志置位时产生中断请求。

位 5	<b>ERRIE:</b> 错误中断使能 (Error interrupt enable) 当错误 (CERR、OVR、MODF) 产生时, 该位控制是否产生中断 0: 禁止错误中断; 1: 允许错误中断。
位 4: 3	保留位, 硬件强制为 0。
位 2	<b>NSSOE:</b> SS 输出使能 (SS output enable) 0: 禁止在主模式下 SS 输出, 该设备可以工作在多主设备模式; 1: 设备开启时, 开启主模式下 SS 输出, 该设备不能工作在多主设备模式。 注: I <sup>2</sup> S 模式下不使用。
位 1	<b>DMATEN:</b> 发送缓冲区 DMA 使能 (DMA Tx enable) 当该位被设置时, TE 标志一旦被置位就发出 DMA 请求 0: 禁止发送缓冲区 DMA; 1: 启动发送缓冲区 DMA。
位 0	<b>DMAREN:</b> 接收缓冲区 DMA 使能 (DMA Rx enable) 当该位被设置时, RNE 标志一旦被置位就发出 DMA 请求 0: 禁止接收缓冲区 DMA; 1: 启动接收缓冲区 DMA。

### 17.4.3 SPI状态寄存器 (SPI\_STS)

地址偏移: 0x08

复位值: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					保留			BSY	OVR	MO DF	CERR	UD R	I <sup>2</sup> S CS	TE	RNE
res						r	r	r	r	r_w0	r	r	r	r	r

位 15: 8	保留位, 硬件强制为 0
位 7	<b>BSY:</b> 忙标志 (Busy flag) 0: SPI 不忙; 1: SPI 正忙于通信, 或者发送缓冲非空。 该位由硬件置位或者复位。 注: 使用这个标志时需要特别注意, 详见 <a href="#">第 17.3.1.7 节</a> 和 <a href="#">第 17.3.1.8 节</a> 。
位 6	<b>OVR:</b> 溢出标志 (Over run flag) 0: 没有出现溢出错误; 1: 出现溢出错误。 该位由硬件置位, 由软件序列复位。 关于软件序列的详细信息, 参考 <a href="#">17.3.2.7 节</a> 。
位 5	<b>MODF:</b> 模式错误 (Mode fault) 0: 没有出现模式错误; 1: 出现模式错误。 该位由硬件置位, 由软件序列复位。 关于软件序列的详细信息, 参考 <a href="#">17.3.2.7 节</a> 。 注: I <sup>2</sup> S 模式下不使用。
位 4	<b>CERR:</b> CRC 错误标志 (CRC error flag) 0: 收到的 CRC 值和 SPI_RCRC 寄存器中的值匹配; 1: 收到的 CRC 值和 SPI_RCRC 寄存器中的值不匹配。 该位由硬件置位, 由软件写'0'而复位。 注: I <sup>2</sup> S 模式下不使用。

位 3	<b>UDR:</b> 下溢标志位 (Under run flag) 0: 未发生下溢; 1: 发生下溢。 该标志位由硬件置'1'，由一个软件序列清'0'，详见 <a href="#">17.3.2.7 节</a> 。 注：在 SPI 模式下不使用。
位 2	<b>I2SCS:</b> 声道 (Channel side) 0: 需要传输或者接收左声道; 1: 需要传输或者接收右声道。 注：在 SPI 模式下不使用。在 PCM 模式下无意义。
位 1	<b>TE:</b> 发送缓冲为空 (Transmit buffer empty) 0: 发送缓冲非空; 1: 发送缓冲为空。
位 0	<b>RNE:</b> 接收缓冲非空 (Receive buffer not empty) 0: 接收缓冲为空; 1: 接收缓冲非空。

#### 17.4.4 SPI数据寄存器 (SPI\_DT)

地址偏移: 0x0C

复位值: 0x0000

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

位 15: 0	<b>DT[15: 0]:</b> 数据寄存器 (Dataregister) 待发送或者已经收到的数据 数据寄存器对应两个缓冲区：一个用于写（发送缓冲）；另外一个用于读（接收缓冲）。 写操作将数据写到发送缓冲区；读操作将返回接收缓冲区里的数据。 对 SPI 模式的注释：根据 SPI_CTRL1 的 DFF16 位对数据帧格式的选择，数据的发送和接收可以是 8 位或者 16 位的。为保证正确的操作，需要在启用 SPI 之前就确定好数据帧格式。 对于 8 位的数据，缓冲器是 8 位的，发送和接收时只会用到 SPI_DT[7: 0]。在接收时，SPI_DT[15: 8]被强制为 0。 对于 16 位的数据，缓冲器是 16 位的，发送和接收时会用到整个数据寄存器，即 SPI_DT[15: 0]
---------	---

#### 17.4.5 SPICRC多项式寄存器 (SPI\_CPOLY) (I<sup>2</sup>S模式下不使用)

地址偏移: 0x10

复位值: 0x0007

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

位 15: 0	CPOLY[15: 0]: CRC 多项式寄存器 (CRCpolynomialregister) 该寄存器包含了 CRC 计算时用到的多项式。其复位值为 0x0007，根据应用可以设置其他数值。 注：在 I <sup>2</sup> S 模式下不使用。
---------	--

#### 17.4.6 SPIxCRC 寄存器 (SPI\_RCRC) (I<sup>2</sup>S 模式下不使用)

地址偏移: 0x14

复位值: 0x0000

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

位 15: 0	<b>RCRC[15: 0]:</b> 接收 CRC 寄存器 在启用 CRC 计算时，RCRC[15: 0]中包含了依据收到的字节计算的 CRC 数值。当在 SPI_CTRL1 的 CCE 位写入'1'时，该寄存器被复位。CRC 计算使用 SPI_CPOLY 中的多项式。当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的标准。 注：当 BSY 标志为'1'时读该寄存器，将可能读到不正确的数值。 注：在 I <sup>2</sup> S 模式下不使用。
---------	--

#### 17.4.7 SPITxCRC 寄存器 (SPI\_TCRC)

地址偏移: 0x18

复位值: 0x0000

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

位 15: 0	<b>TCRC[15: 0]:</b> 发送 CRC 寄存器 在启用 CRC 计算时，TCRC[15: 0]中包含了依据将要发送的字节计算的 CRC 数值。当在 SPI_CTRL1 中的 CCE 位写入'1'时，该寄存器被复位。CRC 计算使用 SPI_CPOLY 中的多项式。 当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 个位都参与计算，并且按照 CRC16 的标准。 注：当 BSY 标志为'1'时读该寄存器，将可能读到不正确的数值。 注：在 I <sup>2</sup> S 模式下不使用。
---------	--

#### 17.4.8 SPI\_I2S 配置寄存器 (SPI\_I2SCTRL)

地址偏移: 0x1C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	I2SS_El	I2SE_N	I2SMOD[1: 0]	PCMS_YNCS_El	保留	I2SAP[1: 0]	CPO_L	DLEN[1: 0]	CHL_EN						
res	rw	rw	rw	rw	rw	res	rw	rw	rw	rw	rw	rw	rw	rw	rw

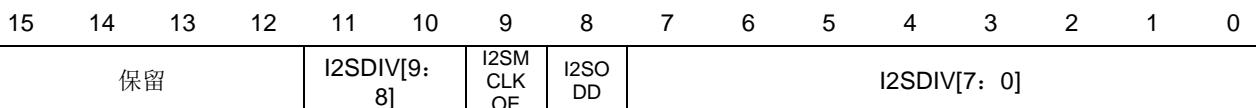
位 15: 12	保留位，硬件强制为 0
----------	-------------

位 11	<b>I2SEL:</b> I <sup>2</sup> S 模式选择 (I <sup>2</sup> Smodeselection) 0: 选择 SPI 模式; 1: 选择 I <sup>2</sup> S 模式。 注: 该位只有在关闭了 SPI 或者 I <sup>2</sup> S 时才能设置。
位 10	<b>I2SEN:</b> I <sup>2</sup> S 使能 (I <sup>2</sup> Senable) 0: 关闭 I <sup>2</sup> S; 1: I <sup>2</sup> S 使能。 注: 在 SPI 模式下不使用。
位 9: 8	I2SMOD[1: 0]: I <sup>2</sup> S 模式设置 (I <sup>2</sup> Sconfigurationmode) 00: 从设备发送; 01: 从设备接收; 10: 主设备发送; 11: 主设备接收。 注: 该位只有在关闭了 I <sup>2</sup> S 时才能设置。 在 SPI 模式下不使用。
位 7	<b>PCMSYNCSEL:</b> PCM 帧同步 (PCMframesynchronization) 0: 短帧同步; 1: 长帧同步。 注: 该位只在 I2SAP=11 (使用 PCM 标准) 时有意义。 在 SPI 模式下不使用。
位 6	保留位, 硬件强制为 0。
位 5: 4	<b>I2SAP[1: 0]:</b> I <sup>2</sup> S 标准选择 (I <sup>2</sup> Sstandardselection) 00: I <sup>2</sup> S 飞利浦标准; 01: 高字节对齐标准 (左对齐); 10: 低字节对齐标准 (右对齐); 11: PCM 标准。关于 I <sup>2</sup> S 标准的细节, 详见 <a href="#">17.3.2.2 节</a> 。 注: 为了正确操作, 只有在关闭了 I <sup>2</sup> S 时才能设置该位。 在 SPI 模式下不使用。
位 3	<b>CPOL:</b> 静止态时钟极性 (Steadystateclockpolarity) 0: I <sup>2</sup> S 时钟静止态为低电平; 1: I <sup>2</sup> S 时钟静止态为高电平。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。 在 SPI 模式下不使用。
位 2: 1	<b>DLEN[1: 0]:</b> 待传输数据长度 (Data length to be transferred) 00: 16 位数据长度; 01: 24 位数据长度; 10: 32 位数据长度; 11: 不允许。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。 在 SPI 模式下不使用。
位 0	<b>CHLEN:</b> 声道长度 (每个音频声道的数据位数) (Channel length (number of bits per audio channel)) 0: 16 位宽; 1: 32 位宽。 只有在 DLEN=00 时该位的写操作才有意义, 否则声道长度都由硬件固定为 32 位。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。 在 SPI 模式下不使用。

### 17.4.9 SPI\_I2S预分频寄存器 (SPI\_I2SCLKP)

地址偏移: 0x20

复位值: 0x0002



res	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位 15: 12	保留位, 硬件强制为 0											
位 12: 11	<b>I2SDIV[9: 8]: I<sup>2</sup>S 线性预分频 (I<sup>2</sup>Slinearprescaler)</b> 描述见 I2SDIV[7: 0].											
位 9	<b>I2SMCLKOE:</b> 主设备时钟输出使能 (LENasterclockoutputenable) 0: 关闭主设备时钟输出; 1: 主设备时钟输出使能。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。仅在 I <sup>2</sup> S 主设备模式下使用该位。在 SPI 模式下不使用。											
位 8	<b>I2SODD:</b> 奇系数预分频 (Oddfactorfortheprescaler) 0: 实际分频系数=I2SDIV*2; 1: 实际分频系数= (I2SDIV*2) +1。 参见 <a href="#">17.3.2.3 节</a> 。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。仅在 I <sup>2</sup> S 主设备模式下使用该位。在 SPI 模式下不使用。											
位 7: 0	<b>I2SDIV[7: 0]: I<sup>2</sup>S 线性预分频 (I<sup>2</sup>Slinearprescaler)</b> I2SDIV[9: 8]设置在 bit12: 11。 禁止设置 I2SDIV[9: 0]=0 或者 I2SDIV[9: 0]=1 参见 <a href="#">17.3.2.3 节</a> 。 注: 为了正确操作, 该位只有在关闭了 I <sup>2</sup> S 时才能设置。仅在 I <sup>2</sup> S 主设备模式下使用该位。在 SPI 模式下不使用。											

# 18 MCU调试 (MCUDBG)

## 18.1 简介

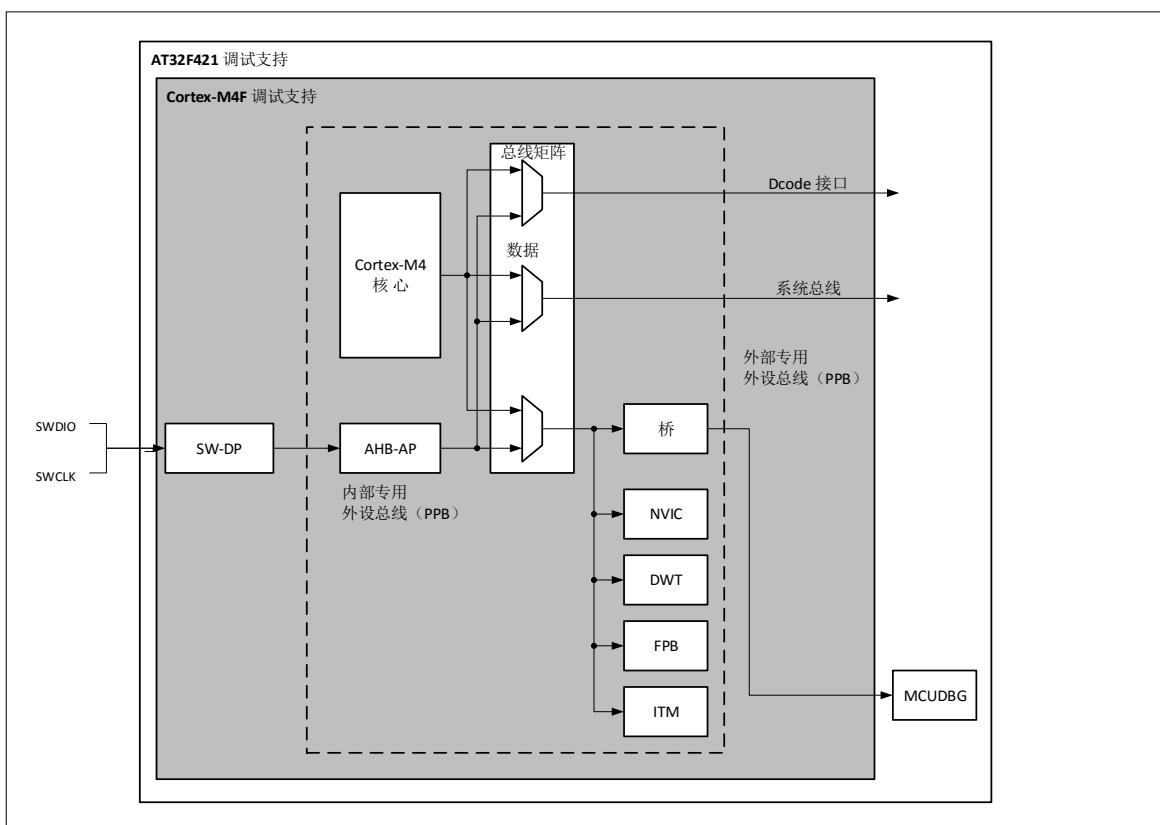
AT32F421 使用 Cortex™-M4 内核，该内核内含硬件调试模块，支持复杂的调试操作。硬件调试模块允许内核在取指(指令断点)或访问数据(数据断点)时停止。内核停止时，内核的内部状态和系统的外部状态都是可以查询的。完成查询后，内核和外设可以被复原，程序将继续执行。

当 AT32F421 微控制器连接到调试器并开始调试时，调试器将使用内核的硬件调试模块进行调试操作。

支持一种调试接口：

串行调试

图表 18-1 AT32F421级别和Cortex™-M4级别的调试框图



ARM Cortex™-M4 内核提供集成的片上调试功能，可参考：

- Cortex™-M4技术参考手册(TRM)
- ARM调试接口V5
- ARM CoreSight 开发工具集(r1p0版)技术参考手册

MCUDBG 模块可帮助调试器调试低功耗模式，定时器，ERTC，I<sup>2</sup>C，WWDG 与 IWDG。当相应位置位，在低功耗模式下提供时钟或保持计数器定时器，WWDG，IWDG 或 I<sup>2</sup>C 的当前状态。MCU 调试模块协助调试器提供以下功能：

- 低功耗模式的调试支持
- 在断点时提供定时器、看门狗和I<sup>2</sup>C的时钟控制
- ID代码

## 18.2 功能描述

### 18.2.1 低功耗模式的调试支持

使用 WFI 和 WFE 可以进入低功耗模式。MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的能耗。内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。

为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

- 在睡眠模式下，调试器必须先置位 MCUDBG\_CTRL 寄存器的 DBG\_SLEEP 位。这将为 HCLK 提供与 FCLK（由代码配置的系统时钟）相同的时钟。
- 在停机模式下，调试器必须先置位 DBG\_STOP 位。这将激活内部 RC 振荡器，在停机模式下为 FCLK 和 HCLK 提供时钟。

### 18.2.2 支持定时器、看门狗和 I<sup>2</sup>C 的调试

在产生断点时，有必要根据定时器和看门狗的不同用途选择计数器的工作模式：

- 在产生断点时，计数器继续计数。这在输出 PWM 控制电机时常常要用到。
- 在产生断点时，计数器停止计数。这对于看门狗的计数器是必需的。

对于 I<sup>2</sup>C，用户可以选择在断点期间阻止 SMBUS 超时。

### 18.2.3 ID 代码

在 AT32F421 微控制器内部有多个 ID 编码，强烈建议工具设计者使用映射在外部 PPB 存储器上地址为 0xE0042000 的 MCUDEVICEID 来锁定调试器。这个 ID 定义了 MCU 的部件号和硅片版本。它是 MCUDBG 的一个组成部分，并且映射到外部 PPB 总线上。

使用 SW 调试口（2 个引脚）或通过用户代码都可以访问此编码。即使当 MCU 处于系统复位状态下这个编码也可以被访问。

### 18.2.4 SW 调试端口脚

AT32F421 的 2 个普通 I/O 口可用作 SW-DP 接口引脚。这些引脚在所有的封装里都存在。复位 (SYSRESETn 或 PORESETn) 以后，属于 SW-DP 的所有 2 个引脚都立即被初始化为可被调试器使用的专用引脚。

AT32F421 微控制器可以用复用重映射和调试 I/O 配置寄存器(见 [6.4.9 以及 6.4.10 节](#))来禁止 SW-DP 接口的功能，这些专用引脚将被释放以用作普通 I/O 口，对此寄存器的设置将由用户代码而不是调试器完成。

## 18.3 MCUDBG 寄存器

下面列出了 MCUDBG 寄存器映象和复位数值。必须以字（32 位）的方式操作这些外设寄存器。

表格 18-1 MCUDBG 寄存器地址映像和复位值

地址	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xE004 2000	MCUDBG_I DCODE																																
	复位值																																

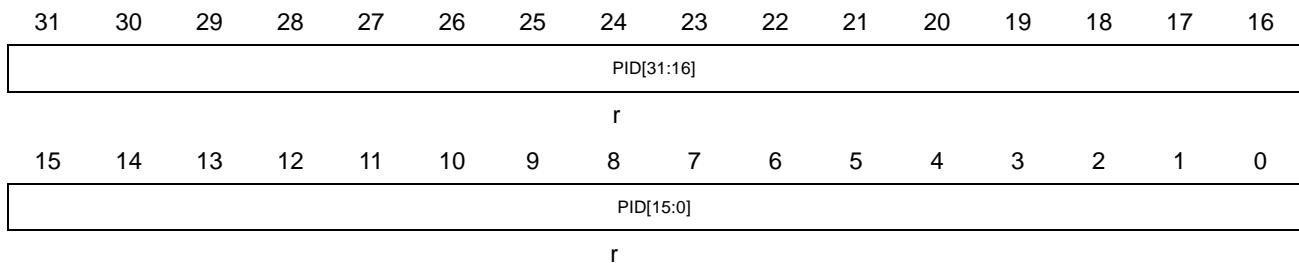
0xE004 2004	MCUDBG_CTRL	保留	保留	DBG_TMR14_STOP	保留														
				0															

### 18.3.1 MCUDBG设备ID（MCUDBG\_IDCODE）

MCU 集成了 ID code，通过 ID 可以识别 MCU 的版本编号。MCUDBG\_IDCODE 寄存器被映射到外部 PPB 总线，基址为 0xE0042000。寄存器可以由用户软件通过 SW debug port 来访问。AT32F421 Series ID 相关信息详见 [1.4.3 节](#)。

地址：0xE0042000（只支持 32 位访问）

POR 复位：0xXXXXXXXXX（不被系统复位所复位）



PID [31: 0]	AT32 型号	FLASH 大小	封装
0x50020100	AT32F421C8T7	64KB	LQFP48
0x50020101	AT32F421K8T7	64KB	LQFP32
0x50020102	AT32F421K8U7	64KB	QFN32 (5x5)
0x50020103	AT32F421K8U7-4	64KB	QFN32 (4x4)
0x50020104	AT32F421F8U7	64KB	QFN20
0x50020105	AT32F421F8P7	64KB	TSSOP20
0x50020086	AT32F421C6T7	32KB	LQFP48
0x50020087	AT32F421K6T7	32KB	LQFP32
0x50020088	AT32F421K6U7	32KB	QFN32 (5x5)
0x50020089	AT32F421K6U7-4	32KB	QFN32 (4x4)
0x5002008A	AT32F421F6U7	32KB	QFN20
0x5002008B	AT32F421F6P7	32KB	TSSOP20
0x5001000C	AT32F421C4T7	16KB	LQFP48
0x5001000D	AT32F421K4T7	16KB	LQFP32
0x5001000E	AT32F421K4U7	16KB	QFN32 (5x5)
0x5001000F	AT32F421K4U7-4	16KB	QFN32 (4x4)
0x50010010	AT32F421F4U7	16KB	QFN20
0x50010011	AT32F421F4P7	16KB	TSSOP20
0x50020112	AT32F421G8U7	64KB	QFN28

0x50020093	AT32F421G6U7	32KB	QFN28
0x50010014	AT32F421G4U7	16KB	QFN28

### 18.3.2 MCUDBG控制寄存器 (MCUDBG\_CTRL)

MCUDBG\_CTRL 寄存器被映射到外部 PPB 总线，基址为 0xE0042004。寄存器由 PORESET 异步复位（不被系统复位所复位）。当内核处于复位状态下时，调试器可写。

地址: 0xE0042004 (只支持 32 位访问)

POR 复位: 0x00000000 (不被系统复位所复位)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		保留		DBG_TMR1_4_ST_OP		保留		DBG_TMR1_7_ST_OP	DBG_TMR1_6_ST_OP	DBG_TMR1_5_ST_OP	DBG_ERTC_512_STOP		保留	DBG_TMR6_STO_P	
RW	RW	RW	RW	RW	RW	RW		res	RW	res	RW	RW	RW	RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG_I2C1_SMBUS_TIM_EOUT	DBG_ERTC_STOP	保留	DBG_TMR3_STO_P	保留	DBG_TMR1_STO_P	DBG_WWDG_STO_P	DBG_IWWDG_STO_P		保留		DBG_STANDBY	DBG_STOP	DBG_SLEEP		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	res	RW	RW	RW	RW	RW

位 31: 28	保留, 必须保持为 0。
位 27	<b>DBG_TMRx_STOP:</b> 当核心停止时停止定时器计数器 (x=14) 0: 当核心停止时, 仍然向相关定时器的计数器提供时钟, 定时器输出工作正常; 1: 当核心停止时, 切断相关定时器的计数器的时钟, 同时关闭定时器的输出 (就好象对某一暂停事件的紧急响应, 停止定时器)。
位 26: 25	保留, 必须保持为 0。
位 24: 22	<b>DBG_TMRx_STOP:</b> 当核心停止时停止定时器计数器 (x=17...15) 0: 当核心停止时, 仍然向相关定时器的计数器提供时钟, 定时器输出工作正常; 1: 当核心停止时, 切断相关定时器的计数器的时钟, 同时关闭定时器的输出 (就好象对某一暂停事件的紧急响应, 停止定时器)。
位 21	<b>DBG_ERTC_512_STOP:</b> 当内核进入调试状态时。 0: RTC 512Hz 时钟仍然正常输出; 1: RTC 512Hz 时钟不输出。
位 20	保留, 必须保持为 0。

位 19	<b>DBG_TMRx_STOP:</b> 当核心停止时停止定时器计数器 ( $x=6$ ) 0: 当核心停止时, 仍然向相关定时器的计数器提供时钟, 定时器输出工作正常; 1: 当核心停止时, 切断相关定时器的计数器的时钟, 同时关闭定时器的输出 (就好象对某一暂停事件的紧急响应, 停止定时器)。
位 18: 17	保留, 必须保持为 0。
位 16	<b>DBG_I2C2_SMBUS_TIMEOUT:</b> 当核心停止时停止 SMBUS 超时模式。 0: 与正常模式操作相同; 1: 冻结 SMBUS 的超时控制。
位 15	<b>DBG_I2C1_SMBUS_TIMEOUT:</b> 当核心停止时停止 SMBUS 超时模式。 0: 与正常模式操作相同; 1: 冻结 SMBUS 的超时控制。
位 14	<b>DBG_ERTC_STOP:</b> 当内核进入调试状态时, RTC 停止运行。 0: RTC 仍然正常运行; 1: RTC 的接收寄存器不继续接收数据。
位 13	保留, 必须保持为 0。
位 12	<b>DBG_TMRx_STOP:</b> 当内核进入调试状态时计数器停止工作 $x=3$ 。 0: 选中定时器的计数器仍然正常工作; 1: 选中定时器的计数器停止工作。
位 11	保留, 必须保持为 0。
位 10	<b>DBG_TMRx_STOP:</b> 当内核进入调试状态时计数器停止工作 $x=1$ 。 0: 选中定时器的计数器仍然正常工作; 1: 选中定时器的计数器停止工作。
位 9	<b>DBG_WWDG_STOP:</b> 当内核进入调试状态时调试窗口看门狗停止工作。 0: 窗口看门狗计数器仍然正常工作; 1: 窗口看门狗计数器停止工作。
位 8	<b>DBG_IWDG_STOP:</b> 当内核进入调试状态时看门狗停止工作 0: 看门狗计数器仍然正常工作; 1: 看门狗计数器停止工作。
位 7: 3	保留, 必须保持为 0。
位 2	<b>DBG_STANDBY:</b> 调试待机模式。 0: (FCLK 关, HCLK 关) 整个数字电路部分都断电。从软件的观点看, 退出 STANDBY 模式与复位是一样的 (除了一些状态位指示了微控制器刚从 STANDBY 状态退出)。 1: (FCLK 开, HCLK 开) 数字电路部分不下电, FCLK 和 HCLK 时钟由内部 RLD 振荡器提供时钟。另外, 微控制器通过产生系统复位来退出 STANDBY 模式和复位是一样的。

位 1	<p><b>DBG_STOP:</b> 调试停机模式。</p> <p>0: (FCLK 关, HCLK 关) 在停机模式时, 时钟控制器禁止一切时钟 (包括 HCLK 和 FCLK)。当从 STOP 模式退出时, 时钟的配置和复位之后的配置一样 (微控制器由 8MHz 的内部 RC 振荡器 (HSI) 提供时钟)。因此, 软件必需重新配置时钟控制系统启动 PLL, 晶振等。</p> <p>1: (FCLK 开, HCLK 开) 在停机模式时, FCLK 和 HCLK 时钟由内部 RC 振荡器提供。当退出停机模式时, 软件必需重新配置时钟系统启动 PLL, 晶振等 (与配置此比特位为 0 时的操作一样)。</p>
位 0	<p><b>DBG_SLEEP:</b> 调试睡眠模式</p> <p>0: (FCLK 开, HCLK 关) 在睡眠模式时, FCLK 由原先已配置好的系统时钟提供, HCLK 则关闭。由于睡眠模式不会复位已配置好的时钟系统, 因此从睡眠模式退出时, 软件不需要重新配置时钟系统。</p> <p>1: (FCLK 开, HCLK 开) 在睡眠模式时, FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。</p>

# 19 比较器(COMP)

## 19.1 COMP简介

AT32F421 内置一个超低功耗比较器 COMP，它可用作独立器件（I/O 上提供了全部接口），也可以与定时器结合使用。

比较器可用于多种功能，包括：

- 由模拟信号触发从低功耗模式唤醒
- 模拟信号调理
- 与定时器的 PWM 输出结合使用时，组成逐周期的电流控制环路

## 19.2 COMP的主要特性

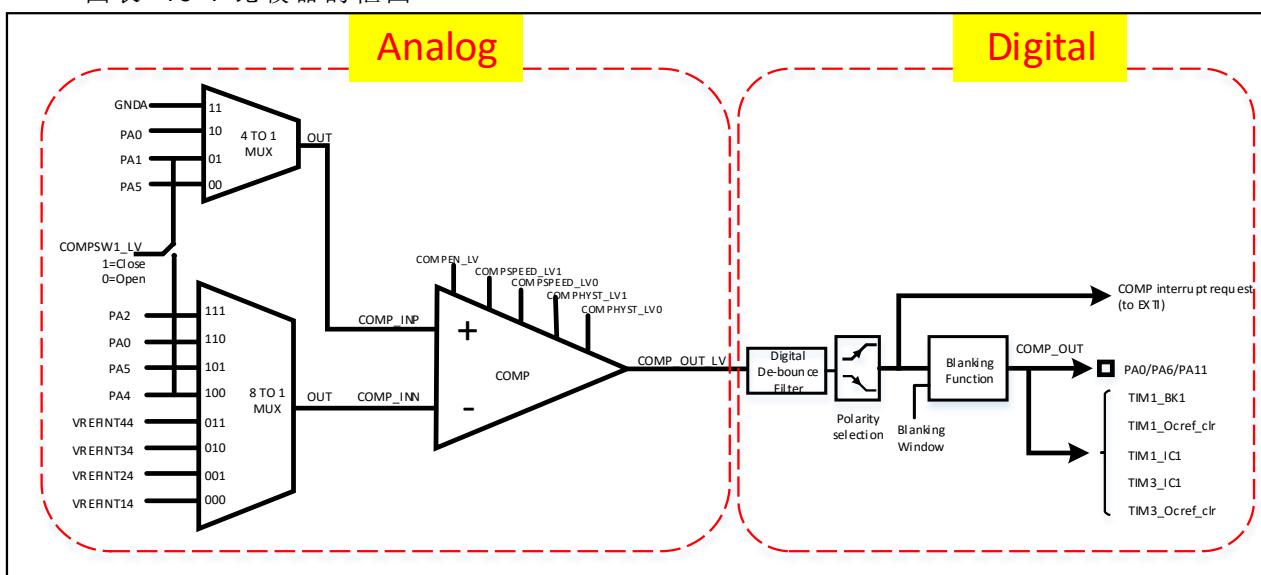
- 比较器具有轨到轨输入以及快速或慢速两种模式
- 每个比较器有可配置的正输入和可配置的负输入，用于灵活选择电压：
  - I/O 引脚
  - 内部参考电压和三个系数分压值(1/4, 1/2, 3/4)
- 可编程速度/功耗
- 可编程迟滞输出
- 输出可以重定向到用于触发以下事件的I/O或定时器输入：
  - 捕捉事件
  - OCREF\_CLR事件（逐周期的电流控制环路）
  - 快速关闭PWM的刹车事件
- 比较器带消隐输出
- 比较器能产生中断，用于使器件从睡眠模式和停机模式唤醒(通过EXTI控制器)

## 19.3 比较器功能描述

### 19.3.1 比较器框图

比较器的框图如图 19-1 比较器的框图所示。

图表 19-1 比较器的框图



### 19.3.2 COMP引脚和内部信号

I/Os 用作比较器输入时，必须在 GPIOs 寄存器配置为模拟模式。

比较器输出可以使用数据手册的“复用功能映射”表给出的复用功能通道连接到 I/O。

输出可以在内部重定向到用作以下用途的各种定时器的输入：

- 紧急关闭 PWM 信号，使用 BKIN
- 逐周期电流控制，使用 OCREF\_CLR 输入
- 用于时序测量的输入捕获

可以在内部和外部同时对比较器输出进行重定向。

### 19.3.3 比较器复位和时钟

比较器的时钟和复位由时钟控制器（RCC）提供，与 PCLK 同步（APB 时钟）。

在 RCC 控制器中没有单独提供这个外设的时钟使能控制位。COMP 时钟使能控制位与 SYSCFG 共用。COMP 只被系统复位重置。

注意：极性选择逻辑和输出端口的重定向工作独立于 PCLK 时钟。这使得比较器在停机模式下仍然可以工作。

### 19.3.4 比较器锁定机制

这两个比较器可用于过流或热保护等安全用途。对于具有特定功能安全要求的应用，必须保证在发生意外的寄存器访问或程序计数器损坏时，不能更改比较器的编程。

为此，可以对比较器控制和状态寄存器进行写保护（只读）。

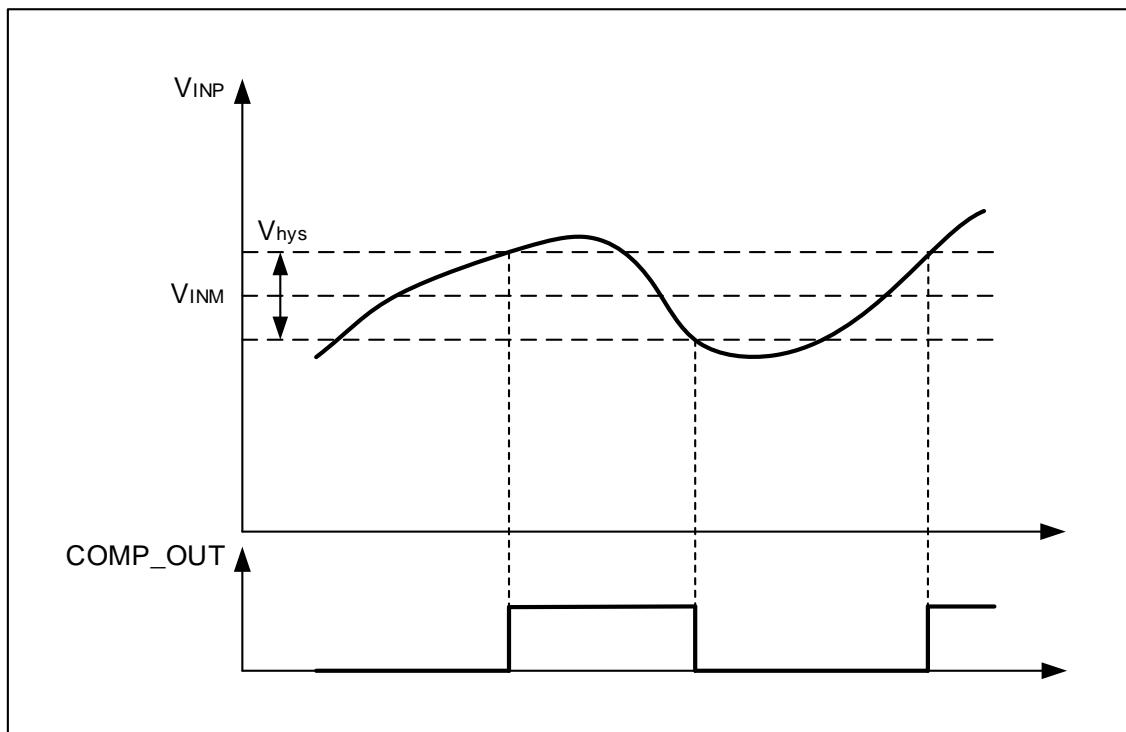
一旦编程完成，COMPLOCK 位可以设置为 1。这使得整个 COMP\_CTRLSTS 寄存器变为只读，包括 COMPLOCK 位。

只能通过系统复位才能复位写保护。

### 19.3.5 迟滞

比较器包含了一个可编程迟滞去避开噪声信号带来的虚假传输信号。如果不需要迟滞，可以关闭掉（例如退出低功耗模式），使得能够使用外部组件去保持迟滞值。

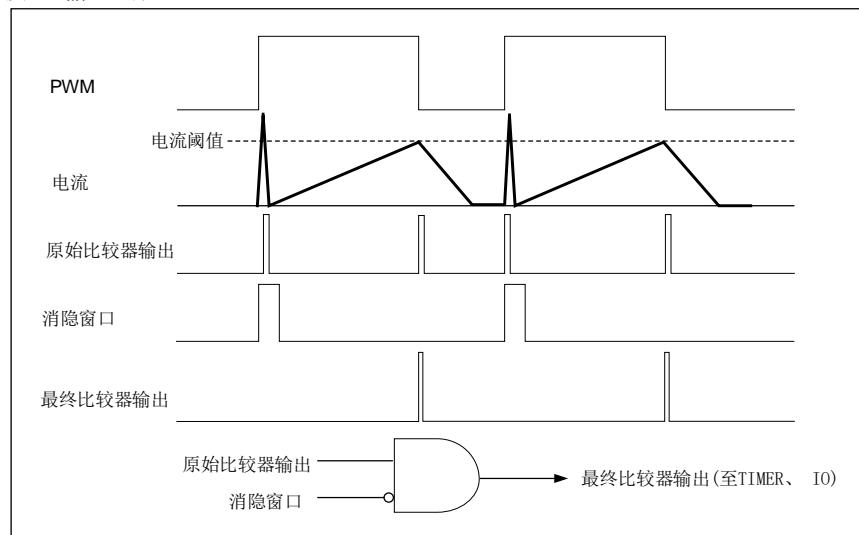
图表 19-2 比较器迟滞



### 19.3.6 比较器输出消隐功能

比较器的输出消隐功能的目的，是防止电流调节由于在 PWM 起始时刻产生的尖峰电流而引起的跳闸（通常情况下，该尖峰电流存在于由反并联二极管组成的电源开关电路）。输出消隐功能通过一个可选的消隐窗口实现，来源于定时器的输出比较信号，消隐窗口通过软件进行配置（参考比较器的寄存器描述选择可用的消隐源）。然后，将消隐信号取反与比较器输出进行与门操作得到想要的比较器输出。实例如下图所示：

图表 19-3 比较器输出消隐



### 19.3.7 功率模式

对于给定的应用，可以调节 COMP 功耗与传输延迟以获得最佳平衡。

可以对 COMP\_CTRLSTS 寄存器中的 COMPMODE 位进行编程，以提供更高速度/功耗或更低速度/功耗。

### 19.3.8 干扰滤波器

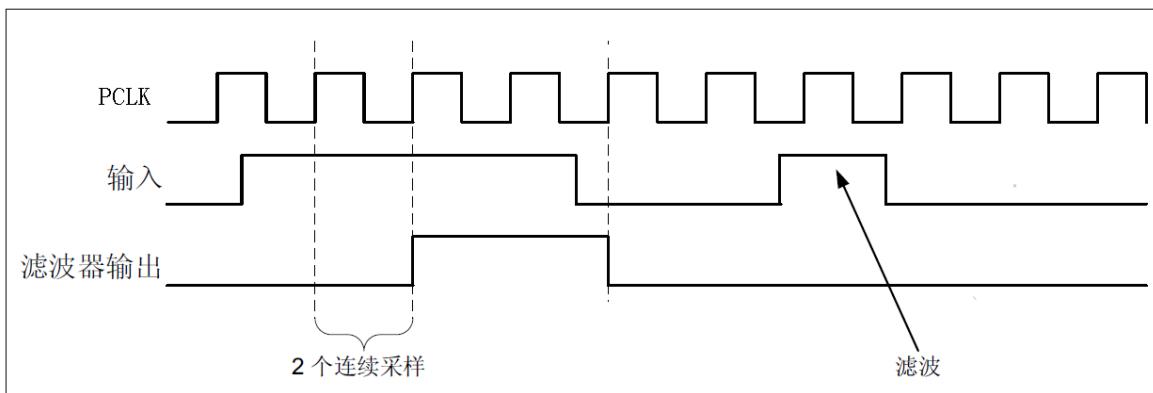
干扰滤波器可以用来滤除毛刺和噪声干扰。

滤波器的敏感性由 H\_PULSE\_CNT 和 L\_PULSE\_CNT 位控制。滤波器的敏感性会影响相同的连续采样的数量，在滤波器输入上检测到此类连续采样时，才能将某信号电平变化视为有效切换。

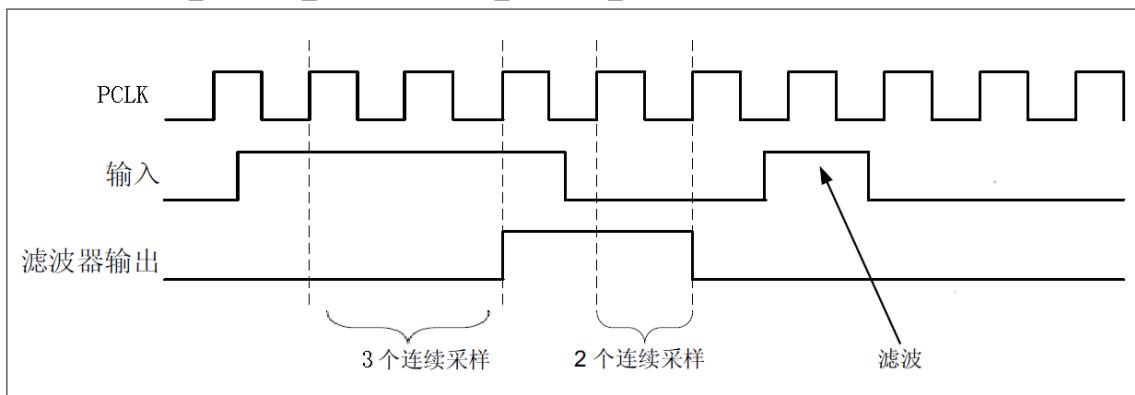
[图 19-4 到 图 19-5](#) 显示了在不同 H\_PULSE\_CNT 和 L\_PULSE\_CNT 值下的时序图。

注：因为滤波器采样数据需要时钟，系统在停机模式下关闭比较器时钟，因此，要让比较器在停机模式下工作，必须在进入停机模式前关闭滤波器（G\_FILTER\_EN 寄存器中的 GFE=0）。

图表 19-4 H\_PULSE\_CNT=1 和 L\_PULSE\_CNT =0 时干扰滤波器时序图



图表 19-5 H\_PULSE\_CNT=2 和 L\_PULSE\_CNT = 1 时干扰滤波器时序图



### 19.4 COMP中断

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

比较器通过 EXTI 线 21 来产生中断或事件。

更多详细信息，请参见中断和事件部分。

### 19.5 COMP寄存器

表格 19-1 COMP寄存器图和复位值

偏移	寄存器	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	-----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

01CH	COMP_CTR LSTS	COMPLOCK	保留	SCALEN	BRGEN 保留	COMPBLANKING	COMPHYST	COMPSEL	COMPINSEL	COMPINMSEL	COMPmode	COMPsw	COMPEN	
		COMPVALUE		0 0		0 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0	0 1	0 0 0 0 0 0	0 0 0 0 0 0	0 0	0	
024H	G_FILTER_EN	保留										GFE 0		
		复位值												
028H	HIGH_PULSE	保留										H_PULSE_CNT 0 0 0 0 0 0		
		复位值												
02CH	LOW_PULSE	保留										L_PULSE_CNT 0 0 0 0 0 0		
		复位值												

### 19.5.1 比较器控制和状态寄存器1(COMP\_CTRLSTS)

COMP\_CTRLSTS为比较器控制/状态寄存器。此寄存器包含与比较器相关的位/标志。

偏移地址：0x1C

复位值：0x0000 0080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COMLOCK	COMPVALUE	保留				SCALEN	BRGEN	保留	COMPBLANKING [2:0]			COMPHYST [1:0]			
rwo	r	res				rw	rw	res	rw	res	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COMPOL	保留		COMPOUTSEL [2:0]		保留	COMPINSEL [1:0]	COMPINMSEL [2:0]			COMPmode [1:0]	COMPsw		COMPEN		
rw	res		rw		res	rw	rw			rw	rw		rw		

位 31	<b>COMPLOCK:</b> 比较器锁定 该位只能写一次, 由软件置 1, 由系统复位清零。它将锁定比较器控制寄存器 COMP_CTRLSTS[31:0]的全部内容 0: 比较器的 COMP_CTRLSTS[31:0]可读/可写 1: 比较器的 COMP_CTRLSTS[31:0]只读
位 30:24	保留, 保持复位值
位 23	<b>SCALEN:</b> 内部等分电压使能 该位使能内部等分电压 0: 内部等分电压不使能 (VREFINT44 = VREFINT34 = VREFINT24 = VREFINT14 = 0V) 1: 内部等分电压使能 (需要 COMPEN = 1, COMPINMSEL[2] = 0 )
位 22	<b>BRGEN:</b> 等分电压桥使能 该位使能等分电压桥 0: 等分电压桥不使能, VREFINT44 = VREFINT34 = VREFINT24 = VREFINT14 = 1.2V (需要 SCALEN = 1, COMPEN = 1, COMPINMSEL[2] = 0 ) 1: 等分电压桥使能, VREFINT44 = 1.2V, VREFINT34 = 0.9V, VREFINT24 = 0.6V, VREFINT14 = 0.3V (需要 SCALEN = 1, COMPEN = 1, COMPINMSEL[2] = 0 )
位 21	保留, 保持复位值
位 20:18	<b>COMPBLANKING[2:0]:</b> 比较器消隐窗口源选择位 这些位选择定时器输出控制比较器输出消隐 000: 没有输出消隐功能 001: TMR1 OC4 作为消隐窗口源 010: 保留 011: TMR3 OC3 作为消隐窗口源 100: TMR15 OC2 作为消隐窗口源 101: 保留 110: TMR15 OC1 作为消隐窗口源 111: 保留
位 17:16	<b>COMPHYST[1:0]:</b> 比较器迟滞 这些位控制比较器的迟滞程度 00: 没有迟滞 01: 低度迟滞 10: 中度迟滞 11: 高度迟滞 参见迟滞程度的电气特性
位 15	<b>COMPPOL:</b> 比较器极性选择位 该位选择比较器的输出极性 0: 比较器输出值不反相 1: 比较器输出值反相
位 14:13	保留, 保持复位值
位 12:10	<b>COMPOUTSEL[2:0]:</b> 比较器输出选择位 这些位控制比较器的输出目的地 000: 无选择 001: 定时器 1 刹车输入 010: 定时器 1 输入捕获 1 011: 定时器 1 OCrefclear 输入 100: 保留 101: 保留 110: 定时器 3 输入捕获 1 111: 定时器 3 OCrefclear 输入
位 9	保留, 保持复位值
位 8:7	<b>COMPINPSEL:</b> 比较器同相输入选择 这些位用于选择比较器同相端输入信号源。复位后为值 01, 即选择 PA1。 00: PA5 01: PA1(默认输入) 10: PA0 11: VSSA
位 6:4	<b>COMPINMSEL[2:0]:</b> 比较器的反相端输入选择

	这些位用于选择输入到比较器反相端的信号源 000: 1/4 V <sub>REFINT</sub> 001: 1/2 V <sub>REFINT</sub> 010: 3/4 V <sub>REFINT</sub> 011: V <sub>REFINT</sub> 100: PA4 101: PA5 110: PA0 111: PA2
位 3:2	<b>COMPMODE[1:0]:</b> 比较器的操作模式选择 这些位用于控制比较器的操作模式，允许调整速率和功耗 00: 高速/最大功耗 01: 中速/中功耗 10: 低速/低功耗 11: 超低速/超低功耗速
位 1	<b>COMPSSW:</b> 比较器同相输入连接开关 该位用于开关比较器同相输入端 PA1 与 PA4 之间的连接 0: 开关断开 1: 开关闭合 注：此开关仅用于重定向信号到高阻输入，例如比较器的同相输入(高阻开关)
位 0	<b>COMPEN:</b> 比较器使能位 该位控制打开或关闭比较器 0: 关闭比较器 1: 开启比较器

### 19.5.2 干扰滤波器使能寄存器(G\_FILTER\_EN)

偏移地址: 0x24

复位值: 0x0000

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

res

rw

位 15:1	保留，始终读为 0。
位 0	<b>GFE:</b> 干扰滤波器使能 (Glitch filter enable) 0: 无作用 1: 使能滤波器

### 19.5.3 干扰滤波器高脉冲数(HIGH\_PULSE)

偏移地址: 0x28

复位值: 0x0000

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

res

rw

位 15:6	保留，始终读为 0。
位 5:0	<b>H_PULSE_CNT:</b> 高脉冲计数值 (High pulse Count) 滤波器输入信号有效电平变化必须至少稳定 H_PULSE_CNT+1 个时钟周期，才能将其视为有效输入，输出才会改变为高电平。 0: 1 个 pclk 时钟 1: 2 个 pclk 时钟

	2: 3 个 pclk 时钟 ..... 62: 63 个 pclk 时钟 63: 64 个 pclk 时钟
--	---

#### 19.5.4 干扰滤波器低脉冲数(LOW\_PULSE)

偏移地址: 0x2C

复位值: 0x0000

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

位 15:6	保留, 始终读为 0。
位 5:0	<b>L_PULSE_CNT:</b> 高脉冲计数值 (Low pulse Count) 滤波器输入信号无效电平变化必须至少稳定 L_PULSE_CNT+1 个时钟周期, 才能将其视为有效输入, 输出才会改变为低电平。 0: 1 个 pclk 时钟 1: 2 个 pclk 时钟 2: 3 个 pclk 时钟 ..... 62: 63 个 pclk 时钟 63: 64 个 pclk 时钟

## 20 版本历史

文档版本历史

日期	版本	变更
2020.07.22	1.00	最初版本

**重要通知 - 请仔细阅读**

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和 / 或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力的产品不得应用于武器。此外，雅特力产品也不是为下列用途而设计并不得应用于下列用途：(A) 对安全性有特别要求的应用，例如：生命支持、主动植入设备或对产品功能安全有要求的系统；(B) 航空应用；(C) 汽车应用或汽车环境，且 / 或 (D) 航天应用或航天环境。如果雅特力产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，采购商仍将独自承担因此而导致的任何风险，雅特力的产品设计规格明确指定的汽车、汽车安全或医疗工业领域专用产品除外。根据相关政府主管部门的规定，ESCC、QML 或 JAN 正式认证产品适用于航天应用。

经销的雅特力产品如有不同于本文档中提出的声明和 / 或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。