RX32H710 参考手册

文档编号：RM00008

基于 32 位 RISC 内核的专业电机微控制器

版本：V2.1

目录

1 简介 30
2 文档约定 31
2.1 寄存器缩写列表 31
2.2 词汇表 31
3 系统和存储器概述 32
3.1 系统架构 32
3.2 存储器结构 32
3.2.1 介绍 32
3.2.2 存储器映像和寄存器边界地址 33
3.3 内置 SRAM 36
3.4 Flash 概述 36
3.5 Boot 配置 36
4 内置 Flash 存储器（FLASH） 37
4.1 介绍 37
4.2 Flash 主要特征 37
4.3 Flash 功能描述 37
4.3.1 Flash 结构 37
4.3.2 读取等待周期 37
4.3.3 指令预取 38
4.3.4 Flash 擦除和编程 38
4.3.5 解锁 Flash 38
4.3.6 Flash 擦除流程 39
4.3.7 FLASH 写入流程 39
4.4 保护 39
4.4.1 读保护 39
4.4.2 专用代码读出保护（PCROP） 40
4.4.3 写保护 41
4.5 选项字节 41
4.5.1 选项字节描述 41
4.5.2 选项字节擦除 41
4.5.3 选项字节写入 41
4.5.4 选项字节组织 42
4.5.5 用户和读保护选项字节 42
4.5.6 PCROP 起始地址选项字节 44
4.5.7 PCROP 末端地址选项字节 44
4.5.8 写保护 A 区地址选项字节 45
4.5.9 写保护 B 区地址选项字节 45
4.6 Flash 寄存器描述 46
4.6.1 Flash 访问控制寄存器（FLASH_ACR） 46
4.6.2 Flash 键寄存器（FLASH_KEYR） 46
4.6.3 Flash OPTKEY 寄存器（FLASH_OPTKEYR） 47

4.6.4 Flash 状态寄存器（FLASH_SR） 47
4.6.5 Flash 控制寄存器（FLASH_CR） 48
4.6.6 Flash 选项字节寄存器（FLASH_OPTR） 50
4.6.7 Flash PCROP 起始地址寄存器（FLASH_PCROPSR） 52
4.6.8 Flash PCROP 末端地址寄存器（FLASH_PCROPER） 52
4.6.9 Flash 写保护 A 区地址寄存器（FLASH_WRP1AR） 53
4.6.10 Flash 写保护 B 区地址寄存器（FLASH_WRP1BR） 53
5 循环冗余校验计算单元（CRC） 54
5.1 CRC 简介 54
5.2 CRC 主要特性 54
5.3 CRC 功能说明 54
5.4 CRC 寄存器 55
5.4.1 数据寄存器（CRC_DR） 55
5.4.2 独立数据寄存器（CRC_IDR） 55
5.4.3 控制寄存器（CRC_CR） 55
6 电源控制（PWR） 56
6.1 概述 56
6.2 电源管理器 56
6.2.1 上电复位（POR）和掉电复位（PDR） 56
6.2.1 可编程电压检测器（PVD） 57
6.3 低功耗模式 58
6.3.1 运行模式 59
6.3.2 睡眠模式 59
6.3.3 停止模式 59
6.3.4 待机模式 59
6.3.5 低功耗模式下的自动唤醒（AWU） 60
6.4 PWR 寄存器描述 61
6.4.1 电源控制寄存器（PWR_CR1） 61
6.4.2 电源控制寄存器（PWR_CR2） 61
6.4.3 电源控制寄存器（PWR_CR3） 62
6.4.4 电源控制寄存器（PWR_CR4） 63
6.4.5 电源状态寄存器 1（PWR_SR1） 64
6.4.6 电源状态寄存器 2（PWR_SR2） 65
6.4.7 电源状态清除寄存器（PWR_SCR） 65
7 复位和时钟控制（RCC） 66
7.1 复位 66
7.1.1 电源复位 66
7.1.2 系统复位 66
7.1.3 RTC 域复位 67
7.2 时钟 67
7.2.1 HSE 时钟 70
7.2.2 HSI 时钟 71
7.2.3 PLL 内部锁相环 72

7.2.4 LSI 时钟 72
7.2.5 系统时钟（SYSCLK）选择 72
7.2.6 时钟安全系统（CSS） 73
7.2.7 ADC 时钟 73
7.2.8 RTC 时钟 73
7.2.9 定时器时钟 73
7.2.10 看门狗时钟 74
7.2.11 时钟输出 74
7.2.12 使用 TIM15/TIM2/TIM3/TIMA/TIMB 进行内部/外部时钟测量 74
7.2.13 外设时钟使能寄存器（RCC_AHBENR, RCC_APBxENR) 76
7.3 低功耗模式 76
7.4 RCC 寄存器 77
7.4.1 时钟控制寄存器（RCC_CR） 77
7.4.2 时钟配置寄存器（RCC_CFGR） 79
7.4.3 时钟中断寄存器（RCC_CIR） 82
7.4.4 APB2 外设复位寄存器（RCC_APB2RSTR） 84
7.4.5 APB1 外设复位寄存器（RCC_APB1RSTR） 85
7.4.6 AHB 外设时钟使能寄存器（RCC_AHBENR） 87
7.4.7 APB2 外设使能寄存器（RCC_APB2ENR） 89
7.4.8 APB1 外设使能寄存器（RCC_APB1ENR） 90
7.4.9 RTC 域控制寄存器（RCC_BDCR） 92
7.4.10 控制/状态寄存器（RCC_CSR） 93
7.4.11 AHB 外设时钟复位寄存器（RCC_AHBRSTR） 95
7.4.12 内部时钟源校准寄存器（RCC_ICSCR） 96
8 通用 I/O（GPIO） 97
8.1 简介 97
8.2 GPIO 主要特征 97
8.3 GPIO 功能描述 97
8.3.1 通用 IO（GPIO） 100
8.3.2 I/O 引脚复用功能复用器和映射 100
8.3.3 I/O 端口控制寄存器 101
8.3.4 I/O 端口数据寄存器 101
8.3.5 I/O 数据位操作 101
8.3.6 I/O 锁定机制 101
8.3.7 I/O 复用功能输入/输出 102
8.3.8 外部中断/唤醒线 102
8.3.9 输入配置 102
8.3.10 输出模式配置 103
8.3.11 复用功能配置 103
8.3.12 模拟配置 104
8.3.13 将 HSE 振荡器引脚用作 GPIO 105
8.4 GPIO 寄存器 106
8.4.1 GPIO 端口模式寄存器（GPIOx_MODER）（x = A...D） 106

8.4.2 GPIO 端口输出类型寄存器（GPIOx_OTYPER）（x = A...D） 106
8.4.3 GPIO 端口输出速度寄存器（GPIOx_OSPEEDR）（x = A...D） 107
8.4.4 GPIO 端口上拉/下拉寄存器（GPIOx_PUPDR）（x = A...D） 107
8.4.5 GPIO 端口输入数据寄存器（GPIOx_IDR）（x = A...D） 108
8.4.6 GPIO 端口输出数据寄存器（GPIOx_ODR）（x = A...D） 108
8.4.7 GPIO 端口置位/复位寄存器（GPIOx_BSRR）（x = A...D） 108
8.4.8 GPIO 端口配置锁定寄存器（GPIOx_LCKR）（x = A...D） 109
8.4.9 GPIO 复用功能低位寄存器（GPIOx_AFRL）（x = A...D） 110
8.4.10 GPIO 复用功能高位寄存器（GPIOx_AFRH）（x = A...D） 110
8.4.11 GPIO 端口位复位寄存器（GPIOx_BRR）（x = A...D） 111
9 系统配置控制器（SYSCFG） 112
9.1 SYSCFG 主要特征 112
9.2 SYSCFG 寄存器 112
9.2.1 SYSCFG 外部中断配置寄存器 1（SYSCFG_EXTICR1） 112
9.2.2 SYSCFG 外部中断配置寄存器 2（SYSCFG_EXTICR2） 112
9.2.3 SYSCFG 外部中断配置寄存器 3（SYSCFG_EXTICR3） 113
9.2.4 SYSCFG 外部中断配置寄存器 4（SYSCFG_EXTICR4） 113
9.2.5 SYSCFG 配置寄存器 2（SYSCFG_CFGR2） 114
10 外部中断和事件控制器（EXTI） 115
10.1 简介 115
10.2 EXTI 主要特性 115
10.3 EXTI 功能介绍 115
10.3.1 EXTI 框图 116
10.3.2 硬件中断选择 116
10.3.3 硬件事件选择 117
10.3.4 软件中断/事件选择 117
10.4 EXTI 中断/事件线映射 117
10.5 EXTI 寄存器描述 119
10.5.1 中断屏蔽寄存器（EXTI_IMR） 119
10.5.2 事件屏蔽寄存器（EXTI_EMR） 120
10.5.3 上升沿触发选择寄存器（EXTI_RTSR） 121
10.5.4 下降沿触发选择寄存器（EXTI_FTSR） 122
10.5.5 软件中断事件寄存器（EXTI_SWIER） 123
10.5.6 挂起寄存器（EXTI_PR） 124
11 CORDIC 协同处理器（CORDIC） 125
11.1 CORDIC 简介 125
11.2 CORDIC 主要特征 125
11.3 CORDIC 功能描述 125
11.3.1 通用描述 125
11.3.2 CORDIC 函数 125
11.3.3 定点表示法 129
11.3.4 缩放系数 129
11.3.5 精度 129

11.3.6 零开销模式 130
11.3.7 轮询方式 131
11.3.8 中断模式 131
11.4 CORDIC 寄存器描述 132
11.4.1 CORDIC 控制/状态寄存器（CORDIC_CSR） 132
11.4.2 CORDIC 参数寄存器（CORDIC_WDATA） 133
11.4.3 CORDIC 结果寄存器（CORDIC_RDATA） 133
12 数学运算协同处理器（ME） 134
12.1 简介 134
12.2 功能描述 134
12.3 ME 寄存器 135
12.3.1 ME 控制寄存器（ME_CR） 135
12.3.2 ME ARG1/RES1 寄存器（ARG1/RES1） 136
12.3.3 ME ARG2/RES2 寄存器（ARG2/RES2） 136
13 模数转换器（ADC） 137
13.1 ADC 简介 137
13.2 ADC 主要特征 137
13.3 ADC 功能描述 138
13.3.1 ADC 框图 138
13.3.2 ADC 引脚和内部信号 139
13.3.3 ADC 时钟 140
13.3.4 ADC 通道连接 140
13.3.5 校准 142
13.3.6 ADC 开关控制 142
13.3.7 通道选择 142
13.3.8 单次转换模式 143
13.3.9 连续转换模式 143
13.3.10 时序图 143
13.3.11 模拟看门狗 144
13.3.12 扫描模式 145
13.3.13 注入通道管理 145
13.3.14 间断模式 146
13.3.15 注入组序列间断计数清零 147
13.3.16 数据对齐 148
13.3.17 可编程的通道采样时间 148
13.3.18 外部触发转换 149
13.4 双 ADC 模式 150
13.4.1 同步注入模式 152
13.4.2 同步规则模式 152
13.4.3 快速交叉模式 153
13.4.4 慢速交叉模式 153
13.4.5 交替触发模式 154
13.4.6 独立模式 155

13.4.7 混合的规则/注入同步模式 155
13.4.8 混合的同步规则+交替触发模式 155
13.4.9 混合同步注入+交叉模式 156
13.5 温度传感器（TPS） 156
13.6 ADC 中断 157
13.7 ADC 寄存器描述 158
13.7.1 ADC 状态寄存器（ADC_SR） 158
13.7.2 ADC 控制寄存器 1（ADC_CR1） 159
13.7.3 ADC 控制寄存器 2（ADC_CR2） 161
13.7.4 ADC 采样时间寄存器 1（ADC_SMPR1） 164
13.7.5 ADC 采样时间寄存器 2（ADC_SMPR2） 164
13.7.6 ADC 注入通道数据偏移寄存器 x（ADC_JOFRx）（x = 1~4） 165
13.7.7 ADC 看门狗高阈值寄存器（ADC_HTR） 165
13.7.8 ADC 看门狗低阈值寄存器（ADC_LTR） 166
13.7.9 ADC 规则序列寄存器 1（ADC_SQR1） 167
13.7.10 ADC 规则序列寄存器 2（ADC_SQR2） 167
13.7.11 ADC 规则序列寄存器 3（ADC_SQR3） 167
13.7.12 ADC 注入序列寄存器（ADC_JSQR） 168
13.7.13 ADC 注入数据寄存器 x（ADC_JDRx）(x = 1~4) 168
13.7.14 ADC 规则数据寄存器（ADC_DR） 169
13.7.15 ADC 控制寄存器 3（ADC_CR3） 169
13.7.16 ADC 校准寄存器（ADC_CAL） 169
13.7.17 ADC 控制寄存器 4（ADC_CR4） 170
13.7.18 ADC TPS 校准寄存器（ADC_TPSOFFSET） 170
13.7.19 ADC 规则数据寄存器 x（ADC_DRx）(x = 1~8) 170
14 数模转换器（DAC） 171
14.1 DAC 简介 171
14.2 DAC 主要特征 171
14.3 DAC 实现 171
14.4 DAC 功能描述 172
14.4.1 DAC 框图 172
14.4.2 DAC 引脚和内部信号 172
14.4.3 DAC 通道使能 173
14.4.4 DAC 数据格式 173
14.4.5 DAC 转换 173
14.4.6 DAC 输出电压 174
14.5 DAC 低功耗模式 174
14.6 DAC 寄存器描述 175
14.6.1 DAC 控制寄存器（DAC_CR） 175
14.6.2 DAC 通道 1 10 位右对齐数据保持寄存器（DAC_DHR10R1） 175
14.6.3 DAC 通道 1 数据输出寄存器（DAC_DOR1） 176
14.6.4 DAC 模式控制寄存器（DAC_MCR） 176
15 直接存储器访问控制器（DMA） 177

15.1 简介 177
15.2 DMA 主要特征 177
15.3 DMA 实现 177
15.3.1 DMA1 实现 177
15.4 DMA 功能描述 178
15.4.1 DMA 框图 178
15.4.2 DMA 引脚和内部信号 178
15.4.3 DMA 请求映像 179
15.4.4 DMA 仲裁器传输 180
15.4.5 DMA 仲裁器 180
15.4.6 DMA 通道 181
15.4.7 DMA 数据宽度、对齐方式和数据大小端 184
15.4.8 DMA 错误管理 185
15.4.9 DMA 中断 185
15.5 DMA 寄存器描述 186
15.5.1 DMA 中断状态寄存器（DMA_ISR） 186
15.5.2 DMA 中断寄存器标志清除寄存器（DMA_IFCR） 187
15.5.3 DMA 通道 x 配置寄存器（DMA_CCRx） 188
15.5.4 DMA 通道 x 传输数量寄存器（DMA_CNDTRx） 191
15.5.5 DMA 通道 x 外设地址寄存器（DMA_CPARx） 192
15.5.6 DMA 通道 x 外设存储地址寄存器（DMA_CMARx） 192
16 比较器（COMP） 193
16.1 简介 193
16.2 主要特性 193
16.3 功能描述 193
16.3.1 COMP 框图 193
16.3.2 COMP 引脚和内部信号 195
16.3.3 COMP 复位和时钟 195
16.3.4 COMP 锁定机制 195
16.3.5 COMP 防误触发功能 196
16.3.6 COMP 迟滞 198
16.3.7 COMP 输出遮蔽功能 198
16.3.8 COMP 低功耗模式 199
16.3.9 COMP 中断 199
16.4 COMP 寄存器描述 200
16.4.1 比较器 x 控制和状态寄存器（COMP_CxCR）（x=1/2/3/4） 200
16.4.2 比较器控制寄存器 1（COMP_CR1） 203
17 运算放大器（OPAMP） 204
17.1 简介 204
17.2 OPAMP 主要特性 204
17.3 OPAMP 功能描述 204
17.3.1 OPAMP 输出重定向至内部 ADC/COMP 通道 204
17.3.2 OPAMP 复位和时钟 204

17.3.3 初始化配置 204
17.3.4 信号路由 205
17.3.5 OPAMP 模式 206
17.3.6 OPAMP 低功耗模式 210
17.4 OPAMP 寄存器描述 210
17.4.1 OPAMP1 控制状态寄存器（OPAMP1_CSR） 210
17.4.2 OPAMPx 控制状态寄存器（OPAMPx_CSR）（x=2...4） 212
17.4.3 OPAMPx 控制寄存器 2（OPAMPx_CR2）（x=1...4） 214
18 高级控制定时器（TIM1/TIM8） 215
18.1 TIM1/TIM8 简介 215
18.2 TIM1/TIM8 主要特征 215
18.3 TIM1/TIM8 功能描述 216
18.3.1 框图 216
18.3.2 TIM1/TIM8 引脚和内部信号 217
18.3.3 时基单元 220
18.3.4 计数模式 222
18.3.5 重复计数器 232
18.3.6 时钟选择 233
18.3.7 捕获比较通道 234
18.3.8 强制输出模式 236
18.3.9 输出比较模式 236
18.3.10 PWM 模式 237
18.3.11 非对称 PWM 模式 245
18.3.12 组合 PWM 模式 246
18.3.13 三相组合 PWM 模式 247
18.3.14 互补输出和死区插入 248
18.3.15 使用刹车功能 251
18.3.16 双向刹车输入 255
18.3.17 在外部事件时清除 tim_ocxref 信号 257
18.3.18 生成 6 步 PWM 258
18.3.19 单脉冲模式 259
18.3.20 可重触发单脉冲模式 260
18.3.21 在比较模式下的脉冲 261
18.3.22 方向位输出 263
18.3.23 UIF 位重映射 263
18.3.24 定时器同步 264
18.3.25 ADC 触发器 267
18.3.26 调试模式 267
18.4 TIM1/TIM8 低功耗模式 268
18.5 TIM1/TIM8 中断 268
18.6 TIM1/TIM8 寄存器描述 269
18.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 1，8） 269
18.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 1，8） 270

18.6.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = 1，8） 274
18.6.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x = 1，8） 276
18.6.5 TIMx 状态寄存器（TIMx_SR）（x = 1，8） 277
18.6.6 TIMx 事件产生寄存器（TIMx_EGR）（x = 1，8） 279
18.6.7 TIMx 捕获/比较模式寄存器 1[复用]（TIMx_CCMR1）（x = 1，8） 280
18.6.8 TIMx 捕获/比较模式寄存器 2 [复用]（TIMx_CCMR2）（x = 1，8） 282
18.6.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = 1，8） 284
18.6.10 TIMx 计数器（TIMx_CNT）（x = 1，8） 287
18.6.11 TIMx 预分频器（TIMx_PSC）（x = 1，8） 287
18.6.12 TIMx 自动重装载寄存器（TIMx_ARR）（x = 1，8） 288
18.6.13 TIMx 重复计数寄存器（TIMx_RCR）（x = 1，8） 288
18.6.14 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x = 1，8） 289
18.6.15 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x = 1，8） 289
18.6.16 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x = 1，8） 290
18.6.17 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x = 1，8） 290
18.6.18 TIMx 刹车和死区寄存器（TIMx_BDTR）（x = 1，8） 291
18.6.19 TIMx 捕获/比较寄存器 5（TIMx_CCR5）（x = 1，8） 296
18.6.20 TIMx 捕获/比较寄存器 6（TIMx_CCR6）（x = 1，8） 297
18.6.21 TIMx 捕获/比较模式寄存器 3（TIMx_CCMR3）（x = 1，8） 297
18.6.22 TIMx 定时器死区寄存器 2（TIMx_DTR2）（x = 1，8） 298
18.6.23 TIMx 定时器编码器控制寄存器（TIMx_ECR）（x = 1，8） 299
18.6.24 TIMx 复用功能选择寄存器 1（TIMx_AF1）（x = 1，8） 299
18.6.25 TIMx 复用功能选择寄存器 2（TIMx_AF2）（x = 1，8） 301
18.6.26 TIMx 捕获/比较寄存器 1N（TIMx_CCR1N）（x = 1，8） 304
18.6.27 TIMx 捕获/比较寄存器 2N（TIMx_CCR2N）（x = 1，8） 304
18.6.28 TIMx 捕获/比较寄存器 3N（TIMx_CCR3N）（x = 1，8） 305
18.6.29 TIMx 捕获/比较寄存器 4N（TIMx_CCR4N）（x = 1，8） 305
18.6.30 TIMx 刹车输入控制选择（TIMx_BICS）（x = 1，8） 306
19 通用定时器（TIM2/TIM3） 307
19.1 TIM2/TIM3 简介 307
19.2 TIM2/TIM3 主要功能 307
19.3 TIM2/TIM3 实现 307
19.4 TIM2/TIM3 功能描述 308
19.4.1 框图 308
19.4.2 TIM2/TIM3 引脚和内部信号 309
19.4.3 时基单元 311
19.4.4 计数器模式 313
19.4.5 时钟选择 323
19.4.6 捕获/比较通道 325
19.4.7 输入捕获模式 327
19.4.8 PWM 输入模式 327
19.4.9 强制输出模式 328
19.4.10 输出比较模式 329

19.4.11 PWM 模式 330
19.4.12 组合 PWM 模式 337
19.4.13 在外部事件时清除 tim_ocxref 信号 339
19.4.14 单脉冲模式 339
19.4.15 可重触发单脉冲模式 341
19.4.16 UIF 位重映射 341
19.4.17 定时器和外部触发的同步 342
19.4.18 定时器同步 345
19.4.19 ADC 触发 349
19.4.20 调试模式 351
19.5 TIM2/TIM3 低功耗模式 351
19.6 TIM2/TIM3 中断 351
19.7 TIM2/TIM3 寄存器描述 352
19.7.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 2，3） 352
19.7.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 2，3） 354
19.7.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = 2，3） 355
19.7.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x = 2，3） 357
19.7.5 TIMx 状态寄存器（TIMx_SR）（x = 2，3） 357
19.7.6 TIMx 事件产生寄存器（TIMx_EGR）（x = 2，3） 359
19.7.7 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = 2，3） 360
19.7.8 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = 2，3） 362
19.7.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = 2，3） 364
19.7.10 TIM3 计数器（TIM3_CNT） 366
19.7.11 TIM2 计数器（TIM2_CNT） 366
19.7.12 TIMx 预分频器（TIMx_PSC）（x = 2，3） 367
19.7.13 TIM3 自动重装载寄存器（TIM3_ARR） 367
19.7.14 TIM2 自动重装载寄存器（TIM2_ARR） 367
19.7.15 TIM3 捕获/比较寄存器 1（TIM3_CCR1） 368
19.7.16 TIM2 捕获/比较寄存器 1（TIM2_CCR1） 368
19.7.17 TIM3 捕获/比较寄存器 2（TIM3_CCR2） 369
19.7.18 TIM2 捕获/比较寄存器 2（TIM2_CCR2） 370
19.7.19 TIMx 编码器控制寄存器（TIMx_ECR）（x = 2，3） 371
19.7.20 TIMx 定时器输入选择寄存器（TIMx_TISEL）（x = 2，3） 371
19.7.21 TIMx 复用功能选择寄存器 2（TIMx_AF2）（x = 2，3） 372
20 通用定时器（TIM15） 373
20.1 TIM15 简介 373
20.2 TIM15 主要功能 373
20.3 TIM15 功能描述 374
20.3.1 框图 374
20.3.2 TIM15 引脚和内部信号 375
20.3.3 时基单元 378
20.3.4 计数器模式 380
20.3.5 重复计数器 383

20.3.6 时钟选择 384
20.3.7 捕获/比较通道 386
20.3.8 输入捕获模式 388
20.3.9 PWM 输入模式 389
20.3.10 强制输出模式 390
20.3.11 输出比较模式 390
20.3.12 PWM 模式 391
20.3.13 组合 PWM 模式 396
20.3.14 互补输出和死区插入 397
20.3.15 使用刹车功能 400
20.3.16 双向刹车输入 404
20.3.17 在外部事件时清除 tim_ocxref 信号 405
20.3.18 单脉冲模式 406
20.3.19 可重触发单脉冲模式 407
20.3.20 UIF 位重映射 408
20.3.21 定时器输入异或功能 408
20.3.22 定时器和外部触发的同步 409
20.3.23 定时器同步 412
20.3.24 ADC 触发 412
20.3.25 调试模式 412
20.4 TIM15 低功耗模式 412
20.5 TIM15 中断 412
20.6 TIM15 寄存器描述 413
20.6.1 TIM15 控制寄存器 1（TIM15_CR1） 413
20.6.2 TIM15 控制寄存器 2（TIM15_CR2） 415
20.6.3 TIM15 从模式控制寄存器（TIM15_SMCR） 417
20.6.4 TIM15 DMA/中断使能寄存器（TIM15_DIER） 418
20.6.5 TIM15 状态寄存器（TIM15_SR） 419
20.6.6 TIM15 事件产生寄存器（TIM15_EGR） 421
20.6.7 TIM15 捕获/比较模式寄存器 1 [复用]（TIM15_CCMR1） 422
20.6.8 TIM15 捕获/比较模式寄存器 1 [复用]（TIM15_CCMR1） 424
20.6.9 TIM15 捕获/比较模式寄存器 2 [复用]（TIM15_CCMR2） 426
20.6.10 TIM15 捕获/比较模式寄存器 2 [复用]（TIM15_CCMR2） 427
20.6.11 TIM15 捕获/比较使能寄存器（TIM15_CCER） 428
20.6.12 TIM15 计数器（TIM15_CNT） 431
20.6.13 TIM15 预分频器（TIM15_PSC） 431
20.6.14 TIM15 自动重装载寄存器（TIM15_ARR） 432
20.6.15 TIM15 重复计数寄存器（TIM15_RCR） 432
20.6.16 TIM15 捕获/比较寄存器 1（TIM15_CCR1） 433
20.6.17 TIM15 捕获/比较寄存器 2（TIM15_CCR2） 434
20.6.18 TIM15 捕获/比较寄存器 3（TIM15_CCR3） 435
20.6.19 TIM15 刹车和死区寄存器（TIM15_BDTR） 436
20.6.20 TIM15 定时器死区寄存器 2（TIM15_DTR2） 440

20.6.21 TIM15 定时器输入选择寄存器（TIM15_TISEL）.....441
20.6.22 TIM15 复用功能选择寄存器 1（TIM15_AF1）.....442
20.6.23 TIM15 复用功能选择寄存器 2（TIM15_AF2）.....444
21 捕获定时器（TIMA/TIMB）.....445
21.1 TIMA/TIMB 简介.....445
21.2 TIMA/TIMB 主要功能.....445
21.3 TIMA/TIMB 功能描述.....446
21.3.1 框图.....446
21.3.2 TIMA/TIMB 引脚和内部信号.....447
21.3.3 时基单元.....449
21.3.4 计数器模式.....451
21.3.5 时钟选择.....460
21.3.6 输入捕获模式.....463
21.3.7 捕获通道.....464
21.3.8 PWM 输入模式.....465
21.3.9 编码器接口模式.....466
21.3.10 UIF 位重映射.....484
21.3.11 定时器输入异或功能.....484
21.3.12 与霍尔传感器接口.....484
21.3.13 定时器和外部触发的同步.....485
21.3.14 定时器同步.....489
21.3.15 调试模式.....494
21.4 TIMA/TIMB 低功耗模式.....494
21.5 TIMA/TIMB 中断.....494
21.6 TIMA/TIMB 寄存器描述.....495
21.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = A，B）.....495
21.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = A，B）.....496
21.6.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = A，B）.....498
21.6.4 TIMx 中断使能寄存器（TIMx_DIER）（x = A，B）.....501
21.6.5 TIMx 状态寄存器（TIMx_SR）（x = A，B）.....502
21.6.6 TIMx 事件产生寄存器（TIMx_EGR）（x = A，B）.....504
21.6.7 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = A，B）.....505
21.6.8 TIMx 捕获/比较模式寄存器 2 [复用]（TIMx_CCMR2）（x = A，B）.....507
21.6.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = A，B）.....507
21.6.10 TIMx 计数器（TIMx_CNT）（x = A，B）.....509
21.6.11 TIMx 预分频器（TIMx_PSC）（x = A，B）.....509
21.6.12 TIMx 自动重装载寄存器（TIMx_ARR）（x = A，B）.....509
21.6.13 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x = A，B）.....510
21.6.14 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x = A，B）.....510
21.6.15 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x = A，B）.....510
21.6.16 TIMx 编码器控制寄存器（TIMx_ECR）（x = A，B）.....511
21.6.17 TIMx 定时器输入选择寄存器（TIMx_TISEL）（x = A，B）.....512
22 基本定时器（TIM6/TIM7）.....513

22.1 TIM6/TIM7 简介 513
22.2 TIM6/TIM7 主要功能 513
22.3 TIM6/TIM7 功能描述 513
22.3.1 框图 513
22.3.2 TIM6/TIM7 内部信号 514
22.3.3 TIM6/TIM7 时钟 514
22.3.4 时基单元 514
22.3.5 计数器模式 516
22.3.6 UIF 位重映射 520
22.3.7 ADC 触发 520
22.3.8 调试模式 521
22.4 TIM6/TIM7 低功耗模式 521
22.5 TIM6/TIM7 中断 521
22.6 TIMx 寄存器描述 522
22.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 6/7） 522
22.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 6/7） 524
22.6.3 TIMx/DMA/中断使能寄存器（TIMx_DIER）（x = 6/7） 524
22.6.4 TIMx 状态寄存器（TIMx_SR）（x = 6/7） 525
22.6.5 TIMx 事件产生寄存器（TIMx_EGR）（x = 6/7） 525
22.6.6 TIMx 计数器（TIMx_CNT）（x = 6/7） 525
22.6.7 TIMx 预分频器（TIMx_PSC）（x = 6/7） 526
22.6.8 TIMx 自动重装载寄存器（TIMx_ARR）（x = 6/7） 526
23 实时时钟（RTC） 527
23.1 RTC 简介 527
23.2 主要特性 527
23.3 功能描述 528
23.3.1 概述 528
23.3.2 复位过程 529
23.3.3 读 RTC 寄存器 529
23.3.4 配置 RTC 寄存器 529
23.3.5 RTC 标志的设置 530
23.4 RTC 寄存器描述 531
23.4.1 RTC 控制寄存器高位（RTC_CRH） 531
23.4.2 RTC 控制寄存器低位（RTC_CRL） 531
23.4.3 RTC 预分频装载寄存器（RTC_PRLH/RTC_PRLL） 533
23.4.4 RTC 预分频器余数寄存器（RTC_DIVH / RTC_DIVL） 534
23.4.5 RTC 计数器寄存器（RTC_CNTH/RTC_CNTL） 534
23.4.6 RTC 闹钟寄存器（RTC_ALRH/RTC_ALRL） 535
24 参考电压缓冲器（VREFBUF） 536
24.1 VREFBUF 简介 536
24.2 VREFBUF 功能描述 536
24.3 VREFBUF 校准 537
24.4 VREFBUF 寄存器描述 537

24.4.1 VREFBUF 控制和状态寄存器（VREFBUF_CSR） 537
24.4.2 VREFBUF 校准控制寄存器（VREFBUF_CCR） 538
25 独立看门狗（IWDG） 539
25.1 简介 539
25.2 IWDG 主要特性 539
25.3 IWDG 功能描述 539
25.3.1 硬件看门狗 539
25.3.2 寄存器访问保护 539
25.3.3 调试模式 539
25.4 IWDG 寄存器描述 541
25.4.1 键寄存器（IWDG_KR） 541
25.4.2 预分频寄存器（IWDG_PR） 541
25.4.3 重装载寄存器（IWDG_RLR） 542
25.4.4 状态寄存器（IWDG_SR） 542
26 FD 控制器局域网络（FDCAN） 543
26.1 简介 543
26.2 FDCAN 主要特性 544
26.3 FDCAN 功能描述 545
26.3.1 FDCAN 框图 545
26.3.2 FDCAN 引脚和内部信号 546
26.3.3 位定时 547
26.3.4 工作模式 548
26.3.5 错误管理 557
26.3.6 消息 RAM 558
26.3.7 FIFO 确认处理 567
26.3.8 FDCAN 接收 FIFO 元素 567
26.3.9 FDCAN 发送缓冲区元素 569
26.3.10 FDCAN 发送事件 FIFO 元素 571
26.3.11 FDCAN 标准消息 ID 过滤器元素 572
26.3.12 FDCAN 扩展消息 ID 过滤器元素 573
26.4 FDCAN 寄存器 574
26.4.1 FDCAN 内核释放寄存器 (FDCAN_CREL) 574
26.4.2 FDCAN 字节序寄存器（FDCAN_ENDN） 574
26.4.3 FDCAN 数据位定时和预分频器寄存器（FDCAN_DBTP） 575
26.4.4 FDCAN 测试寄存器（FDCAN_TEST） 576
26.4.5 FDCAN RAM 看门狗寄存器（FDCAN_RWD） 576
26.4.6 FDCAN CC 控制寄存器（FDCAN_CCCR） 577
26.4.7 FDCAN 标称位定时和预分频器寄存器（FDCAN_NBTP） 579
26.4.8 FDCAN 时间戳计数器配置寄存器（FDCAN_TSCC） 580
26.4.9 FDCAN 时间戳计数器值寄存器（FDCAN_TSCV） 580
26.4.10 FDCAN 超时计数器配置寄存器（FDCAN_TOCC） 581
26.4.11 FDCAN 超时计数器值寄存器（FDCAN_TOCV） 582
26.4.12 FDCAN 错误计数器寄存器（FDCAN_ECR） 582

26.4.13 FDCAN 协议状态寄存器（FDCAN_PSR） 583
26.4.14 FDCAN 发送器延迟补偿寄存器（FDCAN_TDCR） 585
26.4.15 FDCAN 中断寄存器（FDCAN_IR） 585
26.4.16 FDCAN 中断使能寄存器（FDCAN_IE） 588
26.4.17 FDCAN 中断线选择寄存器（FDCAN_ILS） 590
26.4.18 FDCAN 中断线使能寄存器（FDCAN_ILE） 591
26.4.19 FDCAN 全局过滤器配置寄存器（FDCAN_RXGFC） 591
26.4.20 FDCAN 扩展 ID 和掩码寄存器（FDCAN_XIDAM） 593
26.4.21 FDCAN 高优先级消息状态寄存器（FDCAN_HPMS） 593
26.4.22 FDCAN 接收 FIFO 0 状态寄存器（FDCAN_RXF0S） 594
26.4.23 FDCAN 接收 FIFO 0 确认寄存器（FDCAN_RXF0A） 595
26.4.24 FDCAN 接收 FIFO 1 状态寄存器（FDCAN_RXF1S） 595
26.4.25 FDCAN 接收 FIFO 1 确认寄存器（FDCAN_RXF1A） 596
26.4.26 FDCAN 发送缓冲区配置寄存器（FDCAN_TXBC） 596
26.4.27 FDCAN 发送 FIFO/队列状态寄存器（FDCAN_TXFQS） 597
26.4.28 FDCAN 发送缓冲区请求待处理寄存器（FDCAN_TXBRP） 597
26.4.29 FDCAN 发送缓冲区添加请求寄存器（FDCAN_TXBAR） 598
26.4.30 FDCAN 发送缓冲区取消请求寄存器（FDCAN_TXBCR） 599
26.4.31 FDCAN 发送缓冲区发送已发生寄存器（FDCAN_TXBTO） 599
26.4.32 FDCAN 发送缓冲区取消完成寄存器（FDCAN_TXBCF） 600
26.4.33 FDCAN 发送缓冲区发送中断使能寄存器（FDCAN_TXBTIE） 600
26.4.34 FDCAN 发送缓冲区取消完成中断使能寄存器（FDCAN_TXBCIE） 600
26.4.35 FDCAN 发送事件 FIFO 状态寄存器（FDCAN_TXEFS） 601
26.4.36 FDCAN 发送事件 FIFO 确认寄存器（FDCAN_TXEFA） 602
26.4.37 FDCAN 配置时钟分频寄存器（FDCAN_CKDIV） 602
27 串行外设接口（SPI） 603
27.1 SPI 简介 603
27.2 SPI 主要特性 603
27.3 SPI 功能描述 603
27.3.1 概述 603
27.3.2 配置 SPI 为从模式 608
27.3.3 配置 SPI 为主模式 609
27.3.4 配置 SPI 为单工通信 610
27.3.5 数据发送与接收过程 610
27.3.6 CRC 计算 616
27.3.7 状态标志 617
27.3.8 关闭 SPI 618
27.3.9 利用 DMA 的 SPI 通信 619
27.3.10 错误标志 621
27.3.11 SPI 中断 621
27.4 SPI 寄存器描述 622
27.4.1 SPI 控制寄存器 1（SPI_CR1） 622
27.4.2 SPI 控制寄存器 2（SPI_CR2） 624

27.4.3 SPI 状态寄存器（SPI_SR） 625
27.4.4 SPI 数据寄存器（SPI_DR） 626
27.4.5 SPI CRC 多项式寄存器（SPI_CRCPR） 626
27.4.6 SPI RX CRC 寄存器（SPI_RXCRCR） 627
27.4.7 SPI TX CRC 寄存器（SPI_TXCRCR） 627
28 I²C 接口（I2C） 628
28.1 I²C 简介 628
28.2 I²C 主要特点 628
28.3 I²C 功能描述 629
28.3.1 模式选择 629
28.3.2 I²C 从模式 631
28.3.3 I²C 主模式 633
28.3.4 错误条件 639
28.3.5 SDA/SCL 线控制 640
28.3.6 DMA 请求 640
28.4 I²C 中断请求 642
28.5 I²C 寄存器描述 643
28.5.1 控制寄存器 1（I2C_CR1） 643
28.5.2 控制寄存器 2（I2C_CR2） 645
28.5.3 自身地址寄存器 1（I2C_OAR1） 647
28.5.4 自身地址寄存器 2（I2C_OAR2） 647
28.5.5 数据寄存器（I2C_DR） 648
28.5.6 状态寄存器 1（I2C_SR1） 648
28.5.7 状态寄存器 2（I2C_SR2） 651
28.5.8 时钟控制寄存器（I2C_CCR） 653
28.5.9 TRISE 寄存器（I2C_TRISE） 654
29 通用异步收发器（UART） 655
29.1 UART 介绍 655
29.2 UART 主要特性 655
29.3 UART 功能概述 656
29.3.1 UART 特性描述 657
29.3.2 发送器 657
29.3.3 接收器 661
29.3.4 分数波特率的产生 664
29.3.5 UART 接收器容忍时钟的变化 665
29.3.6 校验控制 666
29.3.7 LIN（局域互联网）模式 667
29.3.8 单线半双工通信 669
29.3.9 利用 DMA 连续通信 669
29.4 UART 中断请求 671
29.5 UART 模式配置 672
29.6 UART 寄存器描述 673
29.6.1 状态寄存器（UART_SR） 673

29.6.2 数据寄存器（UART_DR） 675
29.6.3 波特比率寄存器（UART_BRR） 675
29.6.4 控制寄存器 1（UART_CR1） 676
29.6.5 控制寄存器 2（UART_CR2） 678
29.6.6 控制寄存器 3（UART_CR3） 679
30 调试支持（DBG） 680
30.1.1 对定时器和看门狗的调试支持 680
30.1.2 MCU 设备 ID 代码 680
30.1.3 调试 MCU 控制寄存器（DBGMCU_CR） 681
31 器件电子签名 683
31.1 闪存容量寄存器 683
31.2 产品唯一身份标识寄存器（96 位） 683
32 版本历史 685

表目录

表 3.1 RX32H710 系列存储器映像和外设边界地址 34
表 3.2 Boot 模式 36
表 4.1 Flash 结构 37
表 4.2 等待周期和 CPU 时钟频率对应关系 38
表 4.3 PCROP 保护机制(1) 40
表 4.4 选项字节格式 41
表 6.1 低功耗模式一览 58
表 8.1 端口位配置表 (1) 99
表 10.1 EXTI 线路连接 118
表 11.1 CORDIC 函数 126
表 11.2 余弦参数 126
表 11.3 正弦参数 127
表 11.4 相位参数 127
表 11.5 模参数 128
表 11.6 反正切参数 128
表 11.7 精度与迭代次数 130
表 13.1 ADC 内部输入/输出信号 139
表 13.2 ADC 输入/输出引脚 139
表 13.3 ADC1 通道 141
表 13.4 ADC2 通道 141
表 13.5 模拟看门狗通道选择 145
表 13.6 ADC1/2 间断计数清零信号 148
表 13.7 ADC1/2 规则通道的外部触发信号 149
表 13.8 ADC1/2 注入通道的外部触发信号 149
表 13.9 ADC 中断 157
表 14.1 DAC 特征 171
表 14.2 DAC 输入/输出引脚 172
表 14.3 DAC 输入/输出信号 172
表 14.4 HFSEL 描述 174
表 14.5 低功耗模式对 DAC 的影响 174
表 15.1 DMA1 实现 177
表 15.2 DMA 内部输入输出信号 178
表 15.3 各个通道的 DMA1 请求一览 179
表 15.4 可编程的数据传输宽度和大小端操作（当 PINC = MINC = 1） 184
表 15.5 DMA 中断请求 185
表 16.1 COMPx 正向输入分配 195
表 16.2 COMPx 反向输入分配 195
表 16.3 COMPx_P 端和 ADC 的 IO 对应关系 196
表 16.4 COMPx_N 端和 ADC 的 IO 对应关系 197
表 16.5 遮蔽源 198
表 16.6 比较器在低功耗模式下的行为 199

表 17.1 运算放大器的可能连接方式 205
表 17.2 低功耗模式对 OPAMP 的影响 210
表 18.1 TIM 输入/输出引脚 217
表 18.2 TIM 内部输入/输出信号 217
表 18.3 TIMx 内部触发连接 218
表 18.4 定时器刹车互连 218
表 18.5 定时器刹车 2 互连 218
表 18.6 系统中断互连 219
表 18.7 连接到 ocref_clr 的输入多路复用器 219
表 18.8 CCR 和 ARR 寄存器的变化抖动模式 244
表 18.9 CCR 寄存器在中心对齐 PWM 模式下的变化抖动模式 245
表 18.10 刹车保护解除条件 256
表 18.11 低功耗模式对 TIM1/TIM8 的影响 268
表 18.12 中断请求 268
表 18.13 带刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位 286
表 19.1 RX32H710 系列通用定时器 307
表 19.2 TIM 输入/输出引脚 309
表 19.3 TIM 内部输入/输出信号 309
表 19.4 连接到 tim_ti1 的输入多路复用器 309
表 19.5 连接到 tim_ti2 的输入多路复用器 310
表 19.6 TIMx 内部触发连接 310
表 19.7 连接到 ocref_clr 的输入多路复用器 310
表 19.8 CCR 和 ARR 寄存器的变化抖动模式 336
表 19.9 CCR 寄存器在中心对齐 PWM 模式下的变化抖动模式 337
表 19.10 低功耗模式对 TIM2/TIM3 的影响 351
表 19.11 中断请求 351
表 19.12 标准 tim_ocx 通道的输出控制位 365
表 20.1 TIM15 输入/输出引脚 375
表 20.2 TIM15 内部输入/输出信号 375
表 20.3 TIM15 内部触发连接 376
表 20.4 定时器刹车互连 377
表 20.5 系统中断互连 377
表 20.6 连接到 ocref_clr 的输入多路复用器 377
表 20.7 CCR 和 ARR 寄存器的变化抖动模式 396
表 20.8 刹车保护解除条件 404
表 20.9 低功耗模式对 TIM15 的影响 412
表 20.10 中断请求 412
表 20.11 带刹车功能的互补输出通道 tim_oc1 和 tim_oc1n 的控制位（TIM15） 430
表 21.1 TIM 输入/输出引脚 447
表 21.2 TIM 内部输入/输出信号 447
表 21.3 连接到 tim_ti1 的输入多路复用器 447
表 21.4 连接到 tim_ti2 的输入多路复用器 448
表 21.5 连接到 tim_ti3 的输入多路复用器 448

表 21.6 TIMx 内部触发连接 448
表 21.7 连接到 tim_etr 的输入多路复用器 448
表 21.8 计数方向与编码器信号的关系（CC1P = CC2P = 0） 467
表 21.9 计数方向与编码器信号及极性设置的关系 471
表 21.10 低功耗模式对 TIMA/TIMB 的影响 494
表 21.11 中断请求 494
表 22.1 TIM 内部输入/输出信号 514
表 22.2 低功耗模式对 TIM6/TIM7 的影响 521
表 22.3 中断请求 521
表 24.1 VREF 缓冲器模式 536
表 25.1 看门狗超时时间（32kHz 的输入时钟（LSI））(1) 540
表 26.1 CAN 子系统 I/O 信号 546
表 26.2 CAN 子系统 I/O 引脚 546
表 26.3 FDCAN 中的 DLC 编码 550
表 26.4 帧发送的可能配置 564
表 26.5 接收 FIFO 元素 567
表 26.6 接收 FIFO 元素说明 567
表 26.7 发送缓冲区和 FIFO 元素 569
表 26.8 发送缓冲区元素说明 569
表 26.9 发送事件 FIFO 元素 571
表 26.10 发送事件 FIFO 元素说明 571
表 26.11 标准消息 ID 过滤器元素 572
表 26.12 标准消息 ID 过滤器元素位域说明 572
表 26.13 扩展消息 ID 过滤器元素 573
表 26.14 扩展消息 ID 过滤器元素位域说明 573
表 27.1 SPI 中断请求 621
表 28.1 I²C 中断请求 642
表 29.1 检测噪声的数据采样 663
表 29.2 设置波特率时的误差计算 664
表 29.3 当 DIV_Fraction=0 时，UART 接收器的容忍度 665
表 29.4 当 DIV_Fraction!=0 时，UART 接收器的容忍度 665
表 29.5 帧格式 666
表 29.6 UART 中断请求 671
表 29.7 UART 模式设置 (1) 672
表 32.1 版本历史 685

图目录

图 3.1 存储器映像 33
图 5.1 CRC 计算单元框图 54
图 6.1 电源框图 56
图 6.2 上电复位和掉电复位波形图 57
图 6.3 PVD 门限 57
图 7.1 复位电路框图 66
图 7.2 时钟树 69
图 7.3 HSE 时钟源 70
图 7.4 使用 TIM15 捕获模式进行频率测量 74
图 7.5 使用 TIM2/TIM3 捕获模式进行频率测量 75
图 7.6 使用 TIMA/TIMB 捕获模式进行频率测量 75
图 8.1 I/O 端口位基本结构 98
图 8.2 5V 容忍 I/O 端口位的基本结构 98
图 8.3 输入浮空/上拉/下拉配置 102
图 8.4 输出配置 103
图 8.5 复用功能配置 104
图 8.6 高阻态模拟配置 104
图 10.1 可配置的中断/事件框图 116
图 10.2 外部中断/事件 GPIO 映射 117
图 11.1 CORDIC 在三角函数中的收敛情况 129
图 13.1 ADC 框图 138
图 13.2 ADC 时钟方案 140
图 13.3 校准时序图 142
图 13.4 ADC 转换时序图 144
图 13.5 模拟看门狗警戒区 144
图 13.6 注入转换延迟 146
图 13.7 使用间断计数清零来复位触发转换错位 147
图 13.8 数据右对齐 148
图 13.9 数据左对齐 148
图 13.10 双 ADC 框图 151
图 13.11 4 个通道的同步注入模式 152
图 13.12 16 个通道的同步规则模式 152
图 13.13 在 1 个通道上连续转换模式下的快速交叉模式 153
图 13.14 在 1 个通道上的慢速交叉模式 154
图 13.15 交替触发：每个 ADC1 的注入通道组 154
图 13.16 交替触发：在间断模式下每个 ADC 上的 4 个注入通道 155
图 13.17 交替+规则同步 155
图 13.18 触发事件发生在注入转换期间 156
图 13.19 交叉的单通道转换被注入序列 CH11 和 CH12 中断 156
图 13.20 温度传感器和 V_{REFINT} 通道框图 157
图 14.1 DAC 框图 172

图 14.2 DAC 单通道模式下的数据寄存器 173
图 14.3 DAC 转换时序图 174
图 15.1 DMA 框图 178
图 15.2 DMA1 请求映像 179
图 16.1 COMP1 框图 193
图 16.2 COMP2 框图 194
图 16.3 COMP3 框图 194
图 16.4 COMP4 框图 194
图 16.5 比较器迟滞 198
图 16.6 比较器输出遮蔽图 199
图 17.1 独立模式：外部增益设置模式 206
图 17.2 电压跟随器模式 207
图 17.3 PGA 模式单端输入（P 端无偏置电压） 209
图 17.4 PGA 模式差分输入（P 端接偏置电压） 209
图 18.1 高级控制定时器框图 216
图 18.2 当预分频器的参数从 1 变成 2 时，计数器的时序图 221
图 18.3 当预分频器的参数从 1 变成 4 时，计数器的时序图 221
图 18.4 计数器时序图，内部时钟分频系数为 1 222
图 18.5 计数器时序图，内部时钟分频系数为 2 223
图 18.6 计数器时序图，内部时钟分频系数为 4 223
图 18.7 计数器时序图，内部时钟分频系数为 N 224
图 18.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入） 224
图 18.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR） 225
图 18.10 计数器时序图，内部时钟分频系数为 1 226
图 18.11 计数器时序图，内部时钟分频系数为 2 226
图 18.12 计数器时序图，内部时钟分频系数为 4 227
图 18.13 计数器时序图，内部时钟分频系数为 N 227
图 18.14 计数器时序图，当没有使用重复计数器时的更新事件 228
图 18.15 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6 229
图 18.16 计数器时序图，内部时钟分频系数为 2 230
图 18.17 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36 230
图 18.18 计数器时序图，内部时钟分频系数为 N 231
图 18.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 231
图 18.20 计数器时序图，ARPE=1 时的更新事件（计数器上溢） 232
图 18.21 不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置 233
图 18.22 一般模式下的控制电路，内部时钟分频系数为 1 234
图 18.23 捕获/比较通道 1 的主电路 234
图 18.24 捕获/比较通道的输出阶段（通道 1，通道 2、3 和 4 同上） 235
图 18.25 捕获/比较通道的输出阶段（通道 5，通道 6 同上） 235
图 18.26 输出比较模式，翻转 OC1 237
图 18.27 边沿对齐的 PWM 波形（ARR=8） 238
图 18.28 中央对齐的 PWM 波形（ARR=8） 239
图 18.29 抖动原理 240

图 18.30 抖动模式下的数据格式和寄存器编码 241
图 18.31 PWM 分辨率 vs 频率 242
图 18.32 PWM 抖动模式 243
图 18.33 中心对齐 PWM 模式下抖动对占空比的影响 244
图 18.34 生成两个具有 50%占空比的相位偏移 PWM 信号 246
图 18.35 通道 1 和通道 3 的组合 PWM 模式 247
图 18.36 每周期具有多个触发脉冲的三相组合 PWM 信号 248
图 18.37 带死区插入的互补输出 249
图 18.38 非对称死区时间 250
图 18.39 死区波形延迟大于负脉冲 250
图 18.40 死区波形延迟大于正脉冲 250
图 18.41 刹车和刹车 2 电路概述 252
图 18.42 tim_brk 在发生刹车事件时的各种输出行为（OSSI=1） 254
图 18.43 当 tim_brk/tim_brk2 有效（OSSI=0）时 PWM 输出状态 255
图 18.44 输出重定向（未表示 tim_brk2 请求） 256
图 18.45 tim_ocref_clr 输入选择复用器 257
图 18.46 COM 事件生成 6 步 PWM 的示例（OSSR=1） 258
图 18.47 单脉冲模式的例子 259
图 18.48 可重触发单脉冲模式 260
图 18.49 脉冲发生器电路 261
图 18.50 在比较事件上的脉冲生成，适用于边沿对齐模式 261
图 18.51 在同时触发的情况下延长脉冲宽度 262
图 18.52 复位模式下的控制电路 264
图 18.53 门控模式下的控制电路 265
图 18.54 触发器模式下的控制电路 266
图 19.1 通用定时器框图 308
图 19.2 当预分频器的参数从 1 变到 2 时，计数器的时序图 312
图 19.3 当预分频器的参数从 1 变到 4 时，计数器的时序图 312
图 19.4 计数器时序图，内部时钟分频系数为 1 313
图 19.5 计数器时序图，内部时钟分频系数为 2 314
图 19.6 计数器时序图，内部时钟分频系数为 4 314
图 19.7 计数器时序图，内部时钟分频系数为 N 315
图 19.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入） 315
图 19.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR） 316
图 19.10 计数器时序图，内部时钟分频系数为 1 317
图 19.11 计数器时序图，内部时钟分频系数为 2 317
图 19.12 计数器时序图，内部时钟分频系数为 4 318
图 19.13 计数器时序图，内部时钟分频系数为 N 318
图 19.14 计数器时序图，更新事件 319
图 19.15 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6 320
图 19.16 计数器时序图，内部时钟分频系数为 2 321
图 19.17 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36 321
图 19.18 计数器时序图，内部时钟分频系数为 N 322

图 19.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢） 322
图 19.20 计数器时序图，ARPE=1 时的更新事件（计数器溢出） 323
图 19.21 一般模式下的控制电路，内部时钟分频系数为 1 324
图 19.22 tim_ti2 外部时钟连接例子 324
图 19.23 外部时钟模式 1 下的控制电路 325
图 19.24 捕获/比较通道（如：通道 1 输入部分） 325
图 19.25 捕获/比较通道 1 的主电路 326
图 19.26 捕获/比较通道的输出阶段（通道 1，通道 2、3 和 4 同上） 326
图 19.27 PWM 输入模式时序 328
图 19.28 输出比较模式，翻转 tim_oc1 329
图 19.29 边沿对齐的 PWM 波形（ARR=8） 331
图 19.30 中央对齐的 PWM 波形（APR=8） 332
图 19.31 抖动原理 333
图 19.32 抖动模式下的数据格式和寄存器编码 334
图 19.33 PWM 分辨率 vs 频率（16 位模式） 335
图 19.34 PWM 分辨率 vs 频率（28 位模式） 335
图 19.35 PWM 抖动模式 336
图 19.36 中心对齐 PWM 模式下抖动对占空比的影响 337
图 19.37 通道 1 和通道 3 的组合 PWM 模式 338
图 19.38 tim_ocref_clr 输入选择复用器 339
图 19.39 单脉冲模式的例子 340
图 19.40 可重触发单脉冲模式 341
图 19.41 复位模式下的控制电路 342
图 19.42 门控模式下的控制电路 343
图 19.43 触发器模式下的控制电路 344
图 19.44 主/从定时器的例子 345
图 19.45 仅定时器通道 1 主/从连接示例 345
图 19.46 TIM_mstr 的 tim_oc1ref 控制 TIM_slv 346
图 19.47 通过使能 TIM_mstr 可以控制 TIM_slv 347
图 19.48 使用 TIM_mstr 的更新触发 TIM_slv 348
图 19.49 利用 TIM_mstr 的使能触发 TIM_slv 348
图 19.50 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv 349
图 20.1 TIM15 框图 374
图 20.2 当预分频器的参数从 1 变到 2 时，计数器的时序图 379
图 20.3 当预分频器的参数从 1 变到 4 时，计数器的时序图 379
图 20.4 计数器时序图，内部时钟分频系数为 1 380
图 20.5 计数器时序图，内部时钟分频系数为 2 381
图 20.6 计数器时序图，内部时钟分频系数为 4 381
图 20.7 计数器时序图，内部时钟分频系数为 N 382
图 20.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入） 382
图 20.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR） 383
图 20.10 取决于模式和 TIMx_RCR 寄存器设置的更新频率示例 384
图 20.11 一般模式下的控制电路，内部时钟分频系数为 1 385

图 20.12 tim_ti2 外部时钟连接例子 385
图 20.13 外部时钟模式 1 下的控制电路 386
图 20.14 捕获/比较通道（如：通道 1 输入部分） 386
图 20.15 捕获/比较通道 1 的主电路 387
图 20.16 捕获/比较通道的输出阶段（通道 1） 387
图 20.17 捕获比较通道的输出阶段（TIM15 的通道 2） 388
图 20.18 PWM 输入模式时序 390
图 20.19 输出比较模式，翻转 tim_oc1 391
图 20.20 边沿对齐的 PWM 波形（ARR=8） 392
图 20.21 抖动原理 393
图 20.22 抖动模式下的数据格式和寄存器编码 394
图 20.23 PWM 分辨率 vs 频率 395
图 20.24 PWM 抖动模式 395
图 20.25 通道 1 和通道 2 的组合 PWM 模式 397
图 20.26 带死区插入的互补输出 398
图 20.27 非对称死区时间 399
图 20.28 死区波形延迟大于负脉冲 399
图 20.29 死区波形延迟大于正脉冲 399
图 20.30 刹车电路概述 401
图 20.31 tim_brk 在发生刹车事件时的各种输出行为（OSSI=1） 403
图 20.32 输出重定向 405
图 20.33 tim_ocref_clr 输入选择复用器 405
图 20.34 单脉冲模式的例子 406
图 20.35 可重触发单脉冲模式 408
图 20.36 测量两个信号边沿之间的间隔时间 408
图 20.37 复位模式下的控制电路 409
图 20.38 门控模式下的控制电路 410
图 20.39 触发器模式下的控制电路 411
图 21.1 捕获定时器框图 446
图 21.2 当预分频器的参数从 1 变到 2 时，计数器的时序图 450
图 21.3 当预分频器的参数从 1 变到 4 时，计数器的时序图 450
图 21.4 计数器时序图，内部时钟分频系数为 1 451
图 21.5 计数器时序图，内部时钟分频系数为 2 452
图 21.6 计数器时序图，内部时钟分频系数为 4 452
图 21.7 计数器时序图，内部时钟分频系数为 N 453
图 21.8 计数器时序图，更新事件 453
图 21.9 计数器时序图，内部时钟分频系数为 1 454
图 21.10 计数器时序图，内部时钟分频系数为 2 455
图 21.11 计数器时序图，内部时钟分频系数为 4 455
图 21.12 计数器时序图，内部时钟分频系数为 N 456
图 21.13 计数器时序图，更新事件 456
图 21.14 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6 458
图 21.15 计数器时序图，内部时钟分频系数为 2 458

图 21.16 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36 459
图 21.17 计数器时序图，内部时钟分频系数为 N 459
图 21.18 一般模式下的控制电路，内部时钟分频系数为 1 460
图 21.19 tim_ti2 外部时钟连接例子 461
图 21.20 外部时钟模式 1 下的控制电路 461
图 21.21 外部触发输入框图 462
图 21.22 外部时钟模式 2 下的控制电路 463
图 21.23 捕获通道（如：通道 1 输入部分） 464
图 21.24 捕获通道 1 的主电路 465
图 21.25 PWM 输入模式时序 466
图 21.26 编码器接口模式下的计数器工作示例 467
图 21.27 tim_ti1fp1 极性反相时的编码器接口模式示例 468
图 21.28 正交编码器的计数模式 468
图 21.29 方向加时钟编码器模式 469
图 21.30 方向时钟编码器模式（CC1P=CC2P=0） 470
图 21.31 方向时钟编码器模式（CC1P=CC2P=1） 470
图 21.32 索引门控选项 471
图 21.33 抖动的索引信号 472
图 21.34 IPOS[1:0] = 11 时的索引生成 473
图 21.35 通道 A 上带索引门控的计数器读取（IPOS[1:0] = 11） 473
图 21.36 非门控模式下索引的计数器读取（IPOS[1:0] = 00） 474
图 21.37 通道 A 和 B 上带索引门控的计数器读取 474
图 21.38 窄索引脉冲情况下的编码器模式行为（IPOS[1:0] = 11） 475
图 21.39 计数器重置窄索引脉冲（更近距离，ARR = 0x07） 476
图 21.40 x1 和 x2 模式下的索引行为（IPOS[1:0] = 01） 477
图 21.41 定向索引灵敏度 478
图 21.42 作为 FIDX 位设置的功能的计数器重置 478
图 21.43 时钟+方向模式下的索引行为，IPOS[0] = 1 479
图 21.44 方向时钟模式下的索引行为，IPOS[0] = 1 479
图 21.45 正交编码信号的状态图 480
图 21.46 递增计数编码器错误检测 481
图 21.47 递减计数编码器错误检测 482
图 21.48 更新时带有预加载转移的编码器模式更改（SMSPS = 0） 483
图 21.49 复位模式下的控制电路 485
图 21.50 门控模式下的控制电路 486
图 21.51 触发器模式下的控制电路 487
图 21.52 外部时钟模式 2+触发模式下的控制电路 488
图 21.53 主/从定时器的例子 489
图 21.54 仅定时器通道 1 主/从连接示例 489
图 21.55 TIM_mstr 的 tim_oc1ref 控制 TIM_slv 490
图 21.56 通过使能 TIM_mstr 可以控制 TIM_slv 491
图 21.57 使用 TIM_mstr 的更新触发 TIM_slv 492
图 21.58 利用 TIM_mstr 的使能触发 TIM_slv 492

图 21.59 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv 493
图 22.1 基本定时器框图 513
图 22.2 一般模式下的控制电路，内部时钟分频系数为 1 514
图 22.3 当预分频器的参数从 1 变到 2 时，计数器的时序图 515
图 22.4 当预分频器的参数从 1 变到 4 时，计数器的时序图 516
图 22.5 计数器时序图，内部时钟分频系数为 1 517
图 22.6 计数器时序图，内部时钟分频系数为 2 517
图 22.7 计数器时序图，内部时钟分频系数为 4 518
图 22.8 计数器时序图，内部时钟分频系数为 N 518
图 22.9 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入） 519
图 22.10 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR） 520
图 23.1 简化的 RTC 框图 528
图 23.2 RTC 秒和闹钟波形图示例，PR=0003，ALARM=00004 530
图 23.3 RTC 溢出波形图示例，PR=0003 530
图 25.1 独立看门狗框图 540
图 26.1 CAN 子系统 543
图 26.2 FDCAN 框图 545
图 26.3 位定时 547
图 26.4 收发延迟测量 552
图 26.5 总线监控模式下的引脚控制 553
图 26.6 环回模式下的引脚控制 556
图 26.7 CAN 错误状态图 557
图 26.8 消息 RAM 配置 558
图 26.9 标准消息 ID 过滤器路径 561
图 26.10 扩展消息 ID 过滤器路径 562
图 27.1 SPI 框图 604
图 27.2 单主和单从应用 605
图 27.3 数据时钟时序图 607
图 27.4 主/全双工模式下（BIDIMODE=0 且 RXONLY=0）连续传输时，TXE/RXNE/BSY 行为 612
图 27.5 从/全双工模式下（BIDIMODE=0 且 RXONLY=0）连续传输时，TXE/RXNE/BSY 行为 613
图 27.6 主设备只发送模式（BIDIMODE=0 并且 RXONLY=0）下连续传输时，TXE/BSY 行为 614
图 27.7 从设备只发送模式（BIDIMODE=0 并且 RXONLY=0）下连续传输时，TXE/BSY 行为 614
图 27.8 只接收模式（BIDIMODE=0 并且 RXONLY=1）下连续传输时，RXNE 行为 615
图 27.9 非连续传输发送（BIDIMODE=0 并且 RXONLY=0）时，TXE/BSY 行为 616
图 27.10 使用 DMA 发送 620
图 27.11 使用 DMA 接收 620
图 28.1 I²C 总线协议 629
图 28.2 I²C 的功能框图 630
图 28.3 从发送器的传送序列图 632
图 28.4 从接收器的传送序列图 633
图 28.5 主发送器传送序列图 635
图 28.6 方法 1：主接收器传送序列图 636
图 28.7 方法 2：N>2 时主接收器传送序列图 637

图 28.8 方法 2：N=2 时主接收器传送序列图 638
图 28.9 方法 2：N=1 时主接收器传送序列图 639
图 28.10 I²C 中断映射图 642
图 29.1 UART 框图 656
图 29.2 字长设置 657
图 29.3 配置停止位 658
图 29.4 发送时 TC/TXE 的变化情况 660
图 29.5 起始位侦测 661
图 29.6 检测噪声的数据采样 663
图 29.7 LIN 模式下的断开检测（11 位断开长度 – 设置了 LBDL 位） 668
图 29.8 LIN 模式下的断开检测与帧错误的检测 668
图 29.9 利用 DMA 发送 670
图 29.10 利用 DMA 接收 671
图 29.11 UART 中断映像图 672

1 简介

本参考手册用于帮助开发者理解并使用睿兴科技（南京）有限公司（后文简称“睿兴”或“RX”）的芯片。本文提供了 RX32H710 系列微控制器的存储和外设的完整信息。

相关封装和电气特性请参考相应的数据手册。

本参考手册和数据手册均可从睿兴官网 www.rxtek-icore.com 获得。

2 文档约定

2.1 寄存器缩写列表

read/write（rw）	软件可以读取和写入这个位。
read-only（r）	软件只能读取这个位。
write-only（w）	软件只能写入这个位。读取这个位将返回复位值。
read/clear write0 (rc_w0)	软件可以读取这个位，并且可以通过写入 0 来清零这个位。写入 1 对这个位没有影响。
read/clear write1 (rc_w1)	软件可以读取这个位，并且可以通过写入 1 来清零这个位。写入 0 对这个位没有影响。
read/set (rs)	软件可以读取和设置这个位。写入 0 对这个位没有影响。
read/write once (rwo)	软件只能写入这个位一次，但可以在任何时候读取它。只有复位操作才能将这个位返回到其复位值。
Reserved (Res.)	保留位，必须保持在复位值。

2.2 词汇表

Quad world：四字，128 位长度数据。
Double world：双字，64 位长度数据。
Word：字，32 位长度数据。
Half-word：半字，16 位长度数据。
Byte：字节，8 位长度数据。

3 系统和存储器概述

3.1 系统架构

RX32H710 系列集成了 32 位 RISC 内核，是电机专用的高性能、低功耗微控制器。

- 两个驱动单元
  - 32 位 RISC 内核及先进高性能总线
  - 通用 DMA1
- 三个被动单元
  - 内置 Flash
  - 内置 SRAM
  - AHB 外设包括 AHB 到 APB 的桥和 APB 外设（连接在 APB1 和 APB2）

3.2 存储器结构

3.2.1 介绍

程序存储器，数据存储器，寄存器和 I/O 口统一编址在 4G 的线性地址空间里。

字节在存储器中以小端格式编码。一个字中编号最低的字节被认为是该字的最低有效字节，最高地址字节是最高有效字节。

3.2.2 存储器映像和寄存器边界地址

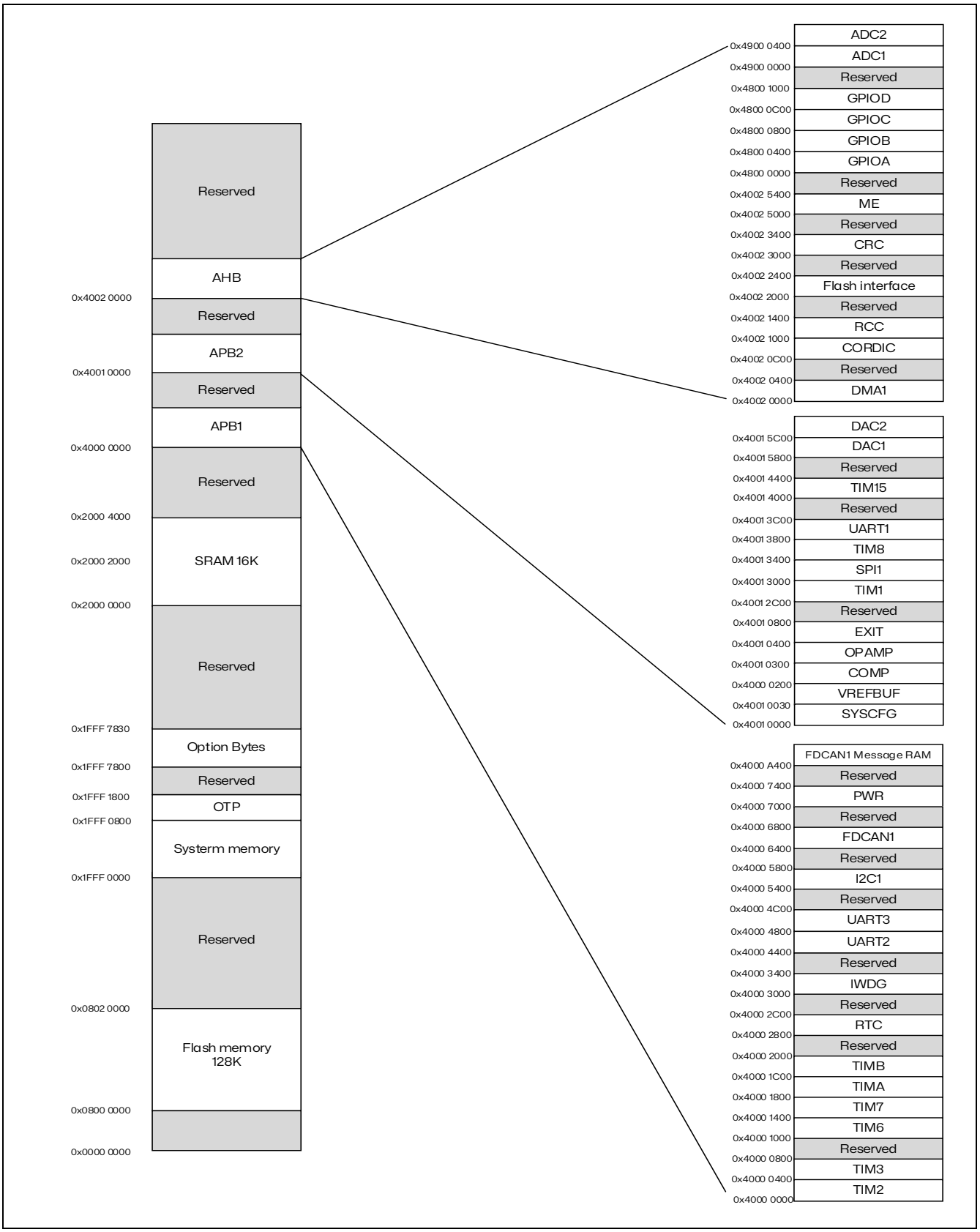


图 3.1 存储器映像

所有未分配给片上存储器和外围设备的内存映射区域都被配置为“Reserved”或者“保留”。有关可用内存和寄存器区域的详细映射，请参阅下表。

下表列出了设备中可用外设的边界地址。

表 3.1 RX32H710 系列存储器映像和外设边界地址

总线	边界地址	外设
AHB	0x4900 0400 - 0x4900 07FF	ADC2
	0x4900 0000 - 0x4900 03FF	ADC1
	0x4800 1000 - 0x48FF FFFF	保留
	0x4800 0C00 - 0x4800 0FFF	GPIOD
	0x4800 0800 - 0x4800 0BFF	GPIOC
	0x4800 0400 - 0x4800 07FF	GPIOB
	0x4800 0000 - 0x4800 03FF	GPIOA
	0x4002 5400 - 0x47FF FFFF	保留
	0x4002 5000 - 0x4002 53FF	ME
	0x4002 3400 - 0x4002 4FFF	保留
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2400 - 0x4002 2FFF	保留
	0x4002 2000 - 0x4002 23FF	Flash 接口
	0x4002 1400 - 0x4002 1FFF	保留
	0x4002 1000 - 0x4002 13FF	RCC
	0x4002 0C00 - 0x4002 0FFF	CORDIC
	0x4002 0400 - 0x4002 0BFF	保留
	0x4002 0000 - 0x4002 03FF	DMA1
APB2	0x4001 6000 - 0x4001 FFFF	保留
	0x4001 5C00 - 0x4001 5FFF	DAC2
	0x4001 5800 - 0x4001 5BFF	DAC1
	0x4001 4400 - 0x4001 47FF	保留
	0x4001 4000 - 0x4001 43FF	TIM15
	0x4001 3C00 - 0x4001 3FFF	保留
	0x4001 3800 - 0x4001 3BFF	UART1
	0x4001 3400 - 0x4001 37FF	TIM8
	0x4001 3000 - 0x4001 33FF	SPI1
	0x4001 2C00 - 0x4001 2FFF	TIM1
	0x4001 0800 - 0x4001 2BFF	保留
	0x4001 0400 - 0x4001 07FF	EXTI
	0x4001 0300 - 0x4001 03FF	OPAMP
	0x4001 0200 - 0x4001 02FF	COMP
	0x4001 0030 - 0x4001 01FF	VREFBUF
	0x4001 0000 - 0x4001 0029	SYSCFG
APB1	0x4000 A800 - 0x4000 FFFF	保留
	0x4000 A400 - 0x4000 A7FF	FDCAN1 Message RAM

总线	边界地址	外设
	0x4000 7400 - 0x4000 A3FF	保留
	0x4000 7000 - 0x4000 73FF	PWR
	0x4000 6800 - 0x4000 6FFF	保留
	0x4000 6400 - 0x4000 67FF	FDCAN1
	0x4000 5800 - 0x4000 63FF	保留
	0x4000 5400 - 0x4000 57FF	I2C1
	0x4000 4C00 - 0x4000 53FF	保留
	0x4000 4800 - 0x4000 4BFF	UART3
	0x4000 4400 - 0x4000 47FF	UART2
	0x4000 3400 - 0x4000 43FF	保留
	0x4000 3000 - 0x4000 33FF	IWDG
	0x4000 2C00 - 0x4000 2FFF	保留
	0x4000 2800 - 0x4000 2BFF	RTC
	0x4000 2000 - 0x4000 27FF	保留
	0x4000 1C00 - 0x4000 1FFF	TIMB
	0x4000 1800 - 0x4000 1BFF	TIMA
	0x4000 1400 - 0x4000 17FF	TIM7
	0x4000 1000 - 0x4000 13FF	TIM6
	0x4000 0800 - 0x4000 0FFF	保留
	0x4000 0400 - 0x4000 07FF	TIM3
	0x4000 0000 - 0x4000 03FF	TIM2

3.3 内置 SRAM

RX32H710 系列内置总计 16KB 的 SRAM。

- 16KB 的 SRAM（映射在 0x2000 0000）

这些 SRAM 可以以字节（8 位）、半字节（16 字节）或字（32 位）进行访问。这些存储器可以在没有等待周期的情况下由 CPU 或者 DMA 寻址。

3.4 Flash 概述

Flash 有两个不同的物理区域组成：

- 主 Flash 块。用于存储应用代码和数据。
- 信息块，主要包含三个部分：
  - 选项字节：用于硬件和存储器保护。
  - 系统存储器：包含 RX 专用代码。
  - OTP：一次性编程区域

Flash 接口基于 AHB 协议进行指令和数据访问。Flash 寄存器具有 Flash 的操作（擦除/写入）功能。

3.5 Boot 配置

RX32H710 的 BOOT 配置可以通过配置选项字节的方式实现，如下表所示：

表 3.2 Boot 模式

nBOOT1 FLASH_OPTR[23]	nBOOT0 FLASH_OPTR[27]	nSWBOOT0 FLASH_OPTR[26]	Boot 区域
x	x	1	主 Flash 块
x	1	0	主 Flash 块
0	0	0	SRAM
1	0	0	系统存储区

内置 Bootloader

内置 Bootloader 代码位于系统存储器（映射在 0x1FFF 0000）中，用于以下串口接口重新编程 Flash：

- 引脚 PB3/PB4 上的 UART1。

4 内置 Flash 存储器（FLASH）

4.1 介绍

Flash 接口管理 CPU 总线对 Flash 进行访问，具有 Flash 擦除和写入功能以及读写保护机制。Flash 接口通过一个包含指令预取的系统来加速代码的执行。

4.2 Flash 主要特征

- 总计 128 KB 的 Flash，128 bit（1×128 bit）的位宽
- Flash 读取/写入支持 128 bit 操作
- 写保护
- 读保护
- 支持页（1KB）擦除和全擦除
- 指令预取（128 bit）
- 选项字节加载器
- 4 KB 的一次性可编程存储区（OTP）

4.3 Flash 功能描述

4.3.1 Flash 结构

Flash 包含 128 页（每页 1KB）的主存储块和信息块，如下表所示。

表 4.1 Flash 结构

模块	名称	地址	大小（字节）
主存储块	页 0	0x0800 0000 - 0x0800 03FF	1K
	页 1	0x0800 0400 - 0x0800 07FF	1K
	页 2	0x0800 0800 - 0x0800 0BFF	1K
	…	…	…
	页 127	0x0801 FC00 - 0x0801 FFFF	1K
信息块	系统存储器	0x1FFF 0000 - 0x1FFF 07FF	2K
	OTP	0x1FFF 0800 – 0x1FFF 17FF	4K
	Option Bytes	0x1FFF 7800 - 0x1FFF 782F	48

4.3.2 读取等待周期

为了正确读取 Flash 的指令和数据，必须根据 CPU 的时钟（HCLK）在 Flash 访问控制寄存器（FLASH_ACR）中配置等待周期（Latency）。等待周期和 CPU 时钟频率的关系如下表所示。

表 4.2 等待周期和 CPU 时钟频率对应关系

等待周期（WS）（Latency）	HCLK（MHz）
0WS（1 CPU cycles）	≤38
1WS（2 CPU cycles）	≤76
2WS（3 CPU cycles）	≤114
3WS（4 CPU cycles）	≤152

复位后，CPU 时钟频率是 16MHz，FLASH_ACR 中配置为 1 等待周期。

修改 CPU 频率时，必须执行以下步骤以保证足够的 Flash 访问等待周期：

增加 CPU 频率：

1. 修改 Flash 访问控制寄存器（FLASH_ACR）中 Latency 的等待周期。
2. 读取并确认 FLASH_ACR 寄存器中的等待周期。
3. 修改 CPU 频率。
4. 确认 CPU 频率。

降低 CPU 频率：

1. 修改 CPU 频率。
2. 确认 CPU 频率。
3. 修改 Flash 访问控制寄存器（FLASH_ACR）中 Latency 的等待周期。
4. 读取并确认 FLASH_ACR 寄存器中的等待周期。

4.3.3 指令预取

预取缓冲区（128 bit）：CPU 每次取指最多为 32 位的字，取一条指令时，下一条指令已经在缓冲区中等待，这样 CPU 可以工作在更高的主频。

复位后预取缓冲区默认关闭。只有在 SYSCLK 小于 24MHz 且 AHB 没有预分频）（SYSCLK 必须等于 HCLK）时才可以开启或关闭预取缓冲区。通常情况下，建议在芯片初始化阶开启或关闭缓冲区，此时芯片运行在内部 16MHz 晶振上。

4.3.4 Flash 擦除和编程

Flash 支持在线编程和应用编程。

在线编程（in-circuit programming，ICP）是指使用串行调试接口或者 bootloader 更新 Flash 的内容。

应用编程（in-application programming，IAP）是指使用芯片支持的通讯协议更新 Flash 的内容。IAP 允许用户在芯片运行过程中更新 Flash 的内容。然而要实现 IAP，需要预先使用 ICP 将部分应用代码编写进 Flash 中。

如果在 Flash 操作期间复位芯片将无法保证 Flash 的内容正确。

4.3.5 解锁 Flash

复位后，Flash 控制寄存器（Flash control register，FLASH_CR）不可写入，这是为了防止 Flash 由于干扰产生误操作。通过以下流程可以解锁 FLASH_CR：

1. 在 Flash 密钥寄存器（Flash key register，FLASH_FKEYR）写入 KEY1 = 0x45670123。
2. 在 Flash 密钥寄存器（Flash key register，FLASH_FKEYR）写入 KEY2 = 0xCDEF89AB。

任何错误的流程将会导致 FLASH_CR 上锁直到系统复位。一旦执行了错误的写入密钥流程，将会产

生一个总线错误和一个硬件错误中断。

FLASH_CR 可以通过写入其中的 LOCK 位来上锁。

注：FLASH_CR 寄存器在 Flash 状态寄存器（Flash status register，FLASH_SR）的 BSY 位置起时不可被写入。在 BSY 位置起时，任何写入动作将会导致 AHB 总线挂起直到 BSY 位被清除。

4.3.6 Flash 擦除流程

Flash 擦除可以分为页擦和全擦。全擦不会擦除信息区（系统存储器和选项字节）。

页擦

页擦流程如下：

1. 确认 FLASH_SR 寄存器的 BSY 位以保证没有正在执行的 Flash 操作。
2. 检查并清除之前的 Flash 操作造成的错误。
3. 将 FLASH_CR 的 PER 位置 1，配置 PNB 选择要擦除的页。
4. 将 FLASH_CR 的 STRT 位置 1。
5. 等 FLASH_SR 中 BSY 被清除。

全擦

全擦流程如下：

1. 确认 FLASH_SR 寄存器的 BSY 位以保证没有正在执行的 Flash 操作。
2. 检查并清除之前的 Flash 操作造成的错误。
3. 将 FLASH_CR 的 MER1 位置 1。
4. 将 FLASH_CR 的 STRT 位置 1。
5. 等 FLASH_SR 中 BSY 被清除。

注：Flash 超出 128KB 不可擦除，也不可写入，不可读出，且不会进硬件错误中断。

注：OTP 不可擦除

4.3.7 FLASH 写入流程

主 FLASH 一次写入一个 128 bit 的数据。写入流程如下：

1. 确认 FLASH_SR 的 BSY 位以保证没有正在执行的 Flash 操作。
2. 将 FLASH_CR 的 PG 位置 1。
3. 向目标地址写入数据（128bit）。
4. 等待 FLASH_SR 的 BSY 位清零。
5. 读取并验证写入的数据。

4.4 保护

闪存中的用户代码区可以防止非法的读出，同样可以对闪存区的页加以保护，防止在程序跑飞的情况下不被意外地改变，写保护的基本单位是 1KB。

闪存的部分区域也可通过专有代码读出保护（PCROP）机制防止第三方进行读写操作。PCROP 的保护粒度为双字（64 位）。

4.4.1 读保护

读保护是通过设置 RDP 选项字节，然后在系统重新复位加载了新的 RDP 选项字节后启动的。当保护字节被写入相应的值后，只允许 Flash 中的代码对 Flash 进行读操作。

解除读保护

1. 擦除整个选项字节区域，读保护码（RDP）将会变为 0xFF，此时读保护仍然有效。
2. 写入正确的 RDP 代码 0xAA 以解除 Flash 的保护，该操作会导致整块 Flash 被擦除。
3. 进行系统复位以重新加载选项字节，读保护被解除。

4.4.2 专用代码读出保护（PCROP）

闪存存储器的部分区域可被保护，以防止第三方进行读写操作。受保护区域仅支持执行操作：只能由 RX32 CPU 作为指令代码访问该区域，而所有其他访问（包括 DMA、调试以及 CPU 的数据读、写和擦除操作）均被严格禁止。PCROP 区域的粒度为双字（64 位）。一个额外的选项位（PCROP_RDP）允许在 RDP 保护级别从级别 1 切换到级别 0 时，选择是否擦除 PCROP 区域。

每个 PCROP 区域均由相对于物理闪存地址的起始页偏移量和末端页偏移量定义。这些偏移量在 PCROP 地址寄存器中定义，包括闪存 PCROP 起始地址寄存器（FLASH_PCROPSR）和闪存 PCROP 结束地址寄存器（FLASH_PCROPER）。

- PCROP 区域的范围是从地址：[0x0800 0000]（闪存基地址）+ [PCROP_STRT × 0x10]（包含该地址）到地址：[0x0800 0000]（闪存基地址）+ [(PCROP_END + 1) × 0x10]（不包含该地址）。PCROP 区域的最小大小为两个双字（128 bit）。

例如，若要通过 PCROP 保护从地址 0x0800 2F80（包含该地址）到地址 0x0801 000C（包含该地址）的区域：

- 若选择从闪存引导，则必须对 FLASH_PCROPSR 和 FLASH_PCROPER 寄存器进行如下编程：
  - PCROP_STRT = 0x02F8
  - PCROP1_END = 0x1000

任何 PCROP 保护地址同时也受写保护，对这些地址任何写访问都会触发 WRPERR（写保护错误）。

任何 PCROP 区域同样受擦除保护。因此，无法对该区域内的任何页进行擦除操作（包括包含该区域起始地址和结束地址的页）。此外，如果某个区域受 PCROP 保护，则无法执行软件全片擦除。

仅当 RDP 保护级别从级别 1 切换到级别 0 时，才能禁用 PCROP。如果尝试通过修改用户选项来清除 PCROP 或缩小 PCROP 区域，选项编程操作会启动，但 PCROP 区域会保持不变。相反，可以扩大 PCROP 区域。

当选项位 PCROP_RDP 被清零，且 RDP 保护级别从级别 1 切换到级别 0 时，为保留 PCROP 区域，全片擦除操作会被部分全片擦除所替代。在这种情况下，PCROP_STRT 和 PCROP_END 也不会被擦除。

注：使用 PCROP_RDP 时，建议将 PCROP 区域与页粒度对齐，或者在 PCROP 区域的起始页或末端页中留出未使用的空间。

表 4.3 PCROP 保护机制(1)

PCROP 寄存器值	PCROP 保护区域
PCROP_offset_strt > PCROP_offset_end	无 PCROP 区域
PCROP_offset_strt > PCROP_offset_end	PCROP_offset_strt 与 PCROP_offset_end 之间的区域受 PCROP 保护。可对以下寄存器进行编程配置： – 将 PCROP_offset_strt 设置为较低地址偏移量 – 将 PCROP_offset_end 设置为较高地址偏移量

1. PCROP 区域的最小容量为 2 个双字（128 bit），由 PCROP_offset_strt 与 PCROP_offset_end 界定。

4.4.3 写保护

RX32H710 系列的写保护以 1KB 为单位。如果在已经开启了写保护的区域进行写入或者擦除操作，FLASH_SR 中的 WRPERR 将会置起。

写保护是通过设置 FLASH_WRP1AR、FLASH_WRP1BR，然后在系统重新复位加载了新的 FLASH_WRP1AR、FLASH_WRP1BR 选项字节后启动的。

解除写保护

1. 擦除整个选项字节区域。
2. 写入正确的 FLASH_WRP1AR、FLASH_WRP1BR 代码。
3. 进行系统复位以重新加载选项字节，写保护被解除。

注：全擦选项字节会将 RDP 擦除为 0xFF，系统重启后会启动读保护，因此需要将 RDP 进行修改。

4.5 选项字节

4.5.1 选项字节描述

选项字节一共有 48 字节，选项字节可以由终端用户根据应用情况进行配置。

选项字节中，一个 64 位的字被做如下划分。

表 4.4 选项字节格式

63-56	55-48	47-40	39-32	31-24	23-16	15-8	7-0
选项字节 3	选项字节 2	选项字节 1	选项字节 0	选项字节 3	选项字节 2	选项字节 1	选项字节 0

选项字节的内容可以通过直接访问选项字节所在地址获取，也可以通过访问选项字节寄存器（FLASH_OPTR）获取。

注：选项字节的新配置在系统重启后才会被加载。

4.5.2 选项字节擦除

选项字节擦除由 FPEC 在选项字节写入过程中自动执行。

4.5.3 选项字节写入

选项字节写入流程如下：

1. 对 FPEC 解锁。
2. 对 FLASH_OPTKEYR 写入 KEY1 = 0x0819 2A3B。
3. 对 FLASH_OPTKEYR 写入 KEY2 = 0x4C5D 6E7F。
4. 检查 FLASH_SR 的 BSY 位，以确认没有其他正在进行的 Flash 操作。
5. 写选项值至 FLASH_OPTR/FLASH_PCROPSR/FLASH_PCROPER/FLASH_WRP1AR/FLASH_WRP1BR。
6. 将 FLASH_CR 寄存器的 OPTSTRT 位置 1，启动选项字节编程。
7. 等待 FLASH_SR 的 BSY 位清零。
8. 读取并验证写入的半字。

4.5.4 选项字节组织

Address	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
1FFF7800	用户选项			读保护	用户选项			读保护
1FFF7808	保留		保留和 PCROP 区起始偏移		保留		保留和 PCROP 区起始偏移	
1FFF7810	PCROP 读保护和保留		保留和 PCROP 区末端偏移		PCROP 读保护和保留		保留和 PCROP 区末端偏移	
1FFF7818	保留	写保护 A 区末端偏移	保留	写保护 A 区起始偏移	保留	写保护 A 区末端偏移	保留	写保护 A 区起始偏移
1FFF7820	保留	写保护 B 区末端偏移	保留	写保护 B 区起始偏移	保留	写保护 B 区末端偏移	保留	写保护 B 区起始偏移

4.5.5 用户和读保护选项字节

地址：0x1FFF 7800

复位值：0x184F 30AA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	RDP1_PC_CHK_MODE	Res	nRST_MODE	n BOOT0	nSW BOOT0	Reserved		nBOOT1	Reserved				IWDG_STDBY	IWDG_STOP	IWDG_SW
	rw		rw	rw	rw			rw					rw	rw	rw

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

Bit 31 保留，必须保持为复位值。

Bit 30 **RDP1_PC_CHK_MODE**：在 RDP1 情况下，访问 Flash 时对 PC 位置的检测模式

0：检测 PC 位置，程序在 SRAM 中运行时不可以访问 Flash

1：不检测 PC 位置，程序在 SRAM 中运行时可以访问 Flash

注：仅在读保护等级为 0 时（RDP0）时才可以修改此位。

Bit 29 保留，必须保持为复位值。

Bit 28 **nRST_MODE**：PA2 引脚模式

0：GPIO

1：复位输入/输出

Bit 27 **nBOOT0**：BOOT0 选择位

0：Boot0=0

1：Boot0=1

Bit 26 **nSWBOOT0**：选择 BOOT0 模式

0：Boot0 取决于 nBOOT0 位

1：Flash Boot

Bit 25:24 保留，必须保持为复位值。

Bit 23 **nBOOT1**：引导配置

0：nBOOT1=0

1：nBOOT1=1

Bit 22:19 保留，必须保持为复位值。

Bit 18 **IWDG_STDBY**：待机模式时独立看门狗计数冻结

0：进入待机模式时独立看门狗计数冻结。

1：进入待机模式时独立看门狗计数运行。

Bit 17 **IWDG_STOP**：停机模式时独立看门狗计数冻结

0：进入停机模式时独立看门狗计数冻结。

1：进入停机模式时独立看门狗计数运行。

Bit 16 **IWDG_SW**：独立看门狗选择

0：硬件独立看门狗

1：软件独立看门狗

Bit 15:14 保留，必须保持为复位值。

Bit 13 **nRST_STDBY**：

0：当进入待机模式时产生复位

1：当进入待机模式时不复位

Bit 12 **nRST_STOP**：

0：当进入停机模式时产生复位

1：当进入停机模式时不复位

Bit 11:8 保留，必须保持为复位值。

Bit 7:0 **RDP[7:0]**：读保护级别

0xAA：级别为 0，读保护未激活。

0xCC：级别为 2，芯片读保护激活。

其他：级别为 1，存储器读保护激活。

4.5.6 PCROP 起始地址选项字节

地址：0x1FFF 7808

复位值：0x0000 01FF

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

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

Bit 31:13 保留，必须保持为复位值。

Bit 12:0 **PCROP_STRT[12:0]：**PCROP 区域起始地址偏移

PCROP_STRT 包含 PCROP 区域的第一个双字地址。

4.5.7 PCROP 末端地址选项字节

地址：0x1FFF 7810

复位值：0x0000 0000

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

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

Bit 31 **PCROP_RDP：**RDP 降级时保留 PCROP 区域

该位仅能通过硬件置位，当 RDP 从 Level 1 降级到 Level 0 并执行全片擦除后复位。

0：当 RDP 从 Level 1 降级到 Level 0 时，不擦除 PCROP 区域。

1：当 RDP 从 Level 1 降级到 Level 0 时，擦除 PCROP 区域（全片擦除）。

Bit 30:13 保留，必须保持为复位值。

Bit 12:0 **PCROP_END[12:0]：**PCROP 区域末端地址偏移

PCROP_END 包含 PCROP 区域的最后一个双字地址。

4.5.8 写保护 A 区地址选项字节

地址：0x1FFF 7818

复位值：0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									WRP1A_END[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
Reserved									WRP1A_STRT[6:0]						
									rw	rw	rw	rw	rw	rw	rw

Bit 31:23 保留，必须保持为复位值。

Bit 22:16 **WRP1A_END[6:0]：**WRP 第一区域“A”区末端地址偏移（单位：1K 字节）
包含 WRP 第一个区域的最后一个页面。

Bit 15:7 保留，必须保持为复位值。

Bit 6:0 **WRP1A_STRT[6:0]：**WRP 第一区域“A”区起始地址偏移（单位：1K 字节）
包含 WRP 第一个区域的第一个页面。

4.5.9 写保护 B 区地址选项字节

地址：0x1FFF 7820

复位值：0x0000 007F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									WRP1B_END[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
Reserved									WRP1B_STRT[6:0]						
									rw	rw	rw	rw	rw	rw	rw

Bit 31:23 保留，必须保持为复位值。

Bit 22:16 **WRP1B_END[6:0]：**WRP 第二区域“B”区末端地址偏移（单位：1K 字节）
包含 WRP 第二个区域的最后一个页面。

Bit 15:7 保留，必须保持为复位值。

Bit 6:0 **WRP1B_STRT[6:0]：**WRP 第二区域“B”区起始地址偏移（单位：1K 字节）
包含 WRP 第二个区域的第一个页面。

4.6 Flash 寄存器描述

4.6.1 Flash 访问控制寄存器（FLASH_ACR）

地址偏移：0x00
复位值：0x0000 0001
访问：无等待，支持字，半字和字节访问

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

Bit 31:9 保留，必须保持为复位值。

Bit 8 **PRFTEN**：预取缓冲区使能
0：关闭预取缓冲区
1：使能预取缓冲区

Bit 7:4 保留，必须保持为复位值。

Bit 3:0 **LATENCY[3:0]**：等待周期
该位表示 SYSCLK（系统时钟）周期与 Flash 访问时间的比例。
000：零等待周期
001：一个等待周期
010：两个等待周期
011：三个等待周期
其他：无

4.6.2 Flash 键寄存器（FLASH_KEYR）

地址偏移：0x08
复位值：0x0000 0000
访问：无等待，支持字访问

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

Bit 31:0 **KEYR**：Flash 键值
KEY1：0x4567 0123
KEY2：0xCDEF 89AB

4.6.3 Flash OPTKEY 寄存器（FLASH_OPTKEYR）

地址偏移：0x0C
复位值：0x0000 0000
访问：无等待，支持字访问

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

Bit 31:0 **OPTKEYR：**选项字节键值

KEY1：0x0819 2A3B
KEY2：0x4C5D 6E7F

4.6.4 Flash 状态寄存器（FLASH_SR）

地址偏移：0x10
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															BSY
															r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OPTV ERR	Reserved							PGS ERR	SIZE ERR	PGA ERR	WRP ERR	PROG ERR	Res	OPE RR	EOP
rc_w1								rc_w1	rc_w1	rc_w1	rc_w1	rc_w1		rc_w1	rc_w1

Bit 31:17 保留，必须保持为 0。

Bit 16 **BSY：**忙碌

表示正在执行一个 Flash 操作。在 Flash 操作开始时该位由硬件置 1，当操作结束或者发生错误时该位由硬件清零。

Bit 15 **OPTVERR：**选项有效性错误

当选项字节没有被正确加载时，该位被硬件置位。
写 1 清零。

Bit 14:8 保留，必须保持为复位值。

Bit 7 **PGSERR：**编程顺序错误

写入闪存时发生顺序错误由硬件置位，或者当 PROGERR、SIZERR、PGAERR、WRPERR 因之前的写入错误而置位时该位由硬件置位。
写 1 清零。

Bit 6 **SIZERR：**字节大小错误

仅允许四字编程，否则该位由硬件置位。
写 1 清零。

Bit 5 **PGAERR**：写入对齐错误

待编程的数据不能被包含在同一个 128 位闪存行中，由硬件置位。

写 1 清零。

Bit 4 **WRPERR**：写保护错误

写入开启了写保护的 Flash 区域时由硬件置位。

写 1 清零。

Bit 3 **PROGERR**：写入错误

当一个四字地址被编程时，写入值不是“0xFFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF”的 Flash 区域时由硬件置位。

写 1 清零。

Bit 2 保留，必须保持为 0。

Bit 1 **OPERR**：操作错误

Flash 区域操作（编程/擦除）未成功完成时，由硬件置位。

写 1 清零。

Bit 0 **EOP**：操作完成

一个或多个 Flash 操作（写入/擦除）成功完成时，由硬件置位。

写 1 清零。

*注：*EOP *只有在*Flash *成功写入或者擦除结束时生效。*

4.6.5 Flash 控制寄存器（FLASH_CR）

地址偏移：0x14

复位值：0xC000 0000

访问：当未执行 Flash 内存操作时，访问无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPT LOCK	Reserved		OBL_LAUNCH	Res	ERRIE	EOPIE	Reserved						OPT STRT	STRT
rs	rs			rc_w1		rw	rw							rs	rs

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PNB[6:0]							MER1	PER	PG
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **LOCK**：锁存位

软件写 1 有效。当该位置 1 时，FPEC 和 FLASH_CR 被锁住。检测到正确的解锁序列后，硬件清零此位。

在一次解锁不成功后，下次系统复位前，该位一直保持置位。

Bit 30 **OPTLOCK**：选项字节锁存位

当该位置位时，FLASH_CR 寄存器中与选项字节有关的所有位和选项页都被锁定。硬件检测到解锁顺序后，清除该位。在对 OPTLOCK 位进行解锁顺序之前，必须先清除 LOCK 位。

在解锁操作不成功的情况下，该位始终保持设置，直到下一次重置。

Bit 29:28 保留，必须保持为复位值。

Bit 27 **OBL_LAUNCH**：强制加载选项字节

当设置 1 时，强制重新加载选项字节。只有选项字节加载完成，该位才被清除。

如果 OPTLOCK 被设置，则该位不能被写入。
0：选项字节加载完成
1：请求加载选项字节

Bit 26 保留，必须保持为复位值。

Bit 25 **ERRIE**：错误中断使能
该位允许在 FLASH_SR 中的 PGERR/WRPERR 被置位时产生中断。
0：关闭 ERR 中断
1：使能 ERR 中断

Bit 24 **EOPIE**：操作结束中断使能
该位允许在 FLASH_SR 中的 EOP 被置位时产生中断。
0：关闭 EOP 中断
1：使能 EOP 中断

Bit 23:18 保留，必须保持为复位值。

Bit 17 **OPTSTRT**：选项修改开始
该位允许修改选项。该位仅由软件设置，在 BSY 置 1 时由硬件清零。

Bit 16 **STRT**：开始
软件写 1 后触发一次擦除操作。在 BSY 置 1 时由硬件清零。

Bit 15:10 保留，必须保持为复位值。

Bit 9:3 **PNB[6:0]**：擦除页码选择
0000000：page 0
0000001：page 1
0000010：page 2
....
0111111：page 63
1111111：page 127

Bit 2 **MER1**：全擦除操作
该位置 1 可使能对存储区的全擦操作。

Bit 1 **PER**：页擦除操作
该位置 1 可使能对主 Flash 的页擦操作。

Bit 0 **PG**：编程操作
该位置 1 可使能对主 Flash 的编程操作。

4.6.6 Flash 选项字节寄存器（FLASH_OPTR）

地址偏移：0x20

复位值：0xFxxx xxxx 该寄存器从 Flash OBL 中加载。

访问：当未执行 Flash 内存操作时，访问无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	RDP1_PC_CHK_MODE	Res	nRST_MODE	n BOOT0	nSW BOOT0	Reserved		nBOOT1	Reserved				IWDG_STDBY	IWDG_STOP	IWDG_SW
	rw		rw	rw	rw			rw					rw	rw	rw

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

Bit 31 保留，必须保持为复位值。

Bit 30 **RDP1_PC_CHK_MODE**：在 RDP1 情况下，访问 Flash 时对 PC 位置的检测模式

0：检测 PC 位置，程序在 SRAM 中运行时不可以访问 Flash

1：不检测 PC 位置，程序在 SRAM 中运行时可以访问 Flash

注：仅在读保护等级为 0 时（RDP0）时才可以修改此位。

Bit 29 保留，必须保持为复位值。

Bit 28 **nRST_MODE**：PA2 引脚模式

0：GPIO

1：复位输入/输出

Bit 27 **nBOOT0**：BOOT0 选择位

0：Boot0=0

1：Boot0=1

Bit 26 **nSWBOOT0**：选择 BOOT0 模式

0：Boot0 取决于 nBOOT0 位

1：Flash Boot

Bit 25:24 保留，必须保持为复位值。

Bit 23 **nBOOT1**：引导配置

0：nBOOT1=0

1：nBOOT1=1

Bit 22:19 保留，必须保持为复位值。

Bit 18 **IWDG_STDBY**：待机模式时独立看门狗计数冻结

0：进入待机模式时独立看门狗计数冻结。

1：进入待机模式时独立看门狗计数运行。

Bit 17 **IWDG_STOP**：停机模式时独立看门狗计数冻结

0：进入停机模式时独立看门狗计数冻结。

1：进入停机模式时独立看门狗计数运行。

Bit 16 **IWDG_SW**：独立看门狗选择

0：硬件独立看门狗

1：软件独立看门狗

Bit 15:14 保留，必须保持为复位值。

Bit 13 **nRST_STDBY**：

0：当进入待机模式时产生复位

1：当进入待机模式时不复位

Bit 12 **nRST_STOP**：

0：当进入停机模式时产生复位

1：当进入停机模式时不复位

Bit 11:8 保留，必须保持为复位值。

Bit 7:0 **RDP[7:0]**：读保护级别

0xAA：级别为 0，读保护未激活。

0xCC：级别为 2，芯片读保护激活。

其他：级别为 1，存储器读保护激活。

4.6.7 Flash PCROP 起始地址寄存器（FLASH_PCROPSR）

地址偏移：0x24
复位值：0x0000 01FF
寄存器位在选项字节加载（OBL）时从 Flash 内存中加载值
访问：当未执行 Flash 内存操作时，访问无等待，支持字访问

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

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

Bit 31:13 保留，必须保持为复位值。

Bit 12:0 **PCROP_STRT[12:0]：**PCROP 区域起始地址偏移
PCROP_STRT 包含 PCROP 区域的第一个双字地址。

4.6.8 Flash PCROP 末端地址寄存器（FLASH_PCROPER）

地址偏移：0x28
复位值：0x0000 0000
寄存器位在选项字节加载（OBL）时从 Flash 内存中加载值
访问：当未执行 Flash 内存操作时，访问无等待，支持字，半字访问。PCROP_RDP 位可字节访问

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

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

Bit 31 **PCROP_RDP：**RDP 降级时保留 PCROP 区域
该位仅能通过硬件置位，当 RDP 从 Level 1 降级到 Level 0 并执行全片擦除后复位。
0：当 RDP 从 Level 1 降级到 Level 0 时，不擦除 PCROP 区域。
1：当 RDP 从 Level 1 降级到 Level 0 时，擦除 PCROP 区域（全片擦除）。

Bit 30:13 保留，必须保持为复位值。

Bit 12:0 **PCROP_END[12:0]：**PCROP 区域末端地址偏移
PCROP_END 包含 PCROP 区域的最后一个双字地址。

4.6.9 Flash 写保护 A 区地址寄存器（FLASH_WRP1AR）

地址偏移：0x2C
复位值：0x0000 007F
寄存器位在选项字节加载（OBL）时从 Flash 内存中加载值
访问：当未执行 Flash 内存操作时，访问无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									WRP1A_END[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
Reserved									WRP1A_STRT[6:0]						
									rw	rw	rw	rw	rw	rw	rw

Bit 31:23 保留，必须保持为复位值。

Bit 22:16 **WRP1A_END[6:0]：**WRP 第一区域“A”区末端地址偏移（单位：1K 字节）
包含 WRP 第一个区域的最后一个页面。

Bit 15:7 保留，必须保持为复位值。

Bit 6:0 **WRP1A_STRT[6:0]：**WRP 第一区域“A”区起始地址偏移（单位：1K 字节）
包含 WRP 第一个区域的第一个页面。

4.6.10 Flash 写保护 B 区地址寄存器（FLASH_WRP1BR）

地址偏移：0x30
复位值：0x0000 007F
寄存器位在选项字节加载（OBL）时从 Flash 内存中加载值
访问：当未执行 Flash 内存操作时，访问无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									WRP1B_END[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
Reserved									WRP1B_STRT[6:0]						
									rw	rw	rw	rw	rw	rw	rw

Bit 31:23 保留，必须保持为复位值。

Bit 22:16 **WRP1B_END[6:0]：**WRP 第二区域“B”区末端地址偏移（单位：1K 字节）
包含 WRP 第二个区域的最后一个页面。

Bit 15:7 保留，必须保持为复位值。

Bit 6:0 **WRP1B_STRT[6:0]：**WRP 第二区域“B”区起始地址偏移（单位：1K 字节）
包含 WRP 第二个区域的第一个页面。

5 循环冗余校验计算单元（CRC）

5.1 CRC 简介

CRC（循环冗余校验）计算单元使用一个固定的多项式发生器从一个 32 位的数据字中产生 CRC 码。在众多的应用中，基于 CRC 的技术还常用来验证数据传输或存储的完整性。根据 EN/IEC 60335-1 标准的规定，这些技术提供了验证 Flash 完整性的方法。CRC 计算单元有助于在运行期间计算软件的签名，并将该签名与链接时生成并存储在指定存储单元的参考签名加以比较。

5.2 CRC 主要特性

- 使用 CRC-32（以太网）多项式：0x4C11DB7
  - $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 单输入/输出 32 位数据寄存器
- CRC 计算在 4 个 AHB 时钟周期（HCLK）内完成
- 8 位通用寄存器（可用于临时存储）

下图为 CRC 计算单元框图

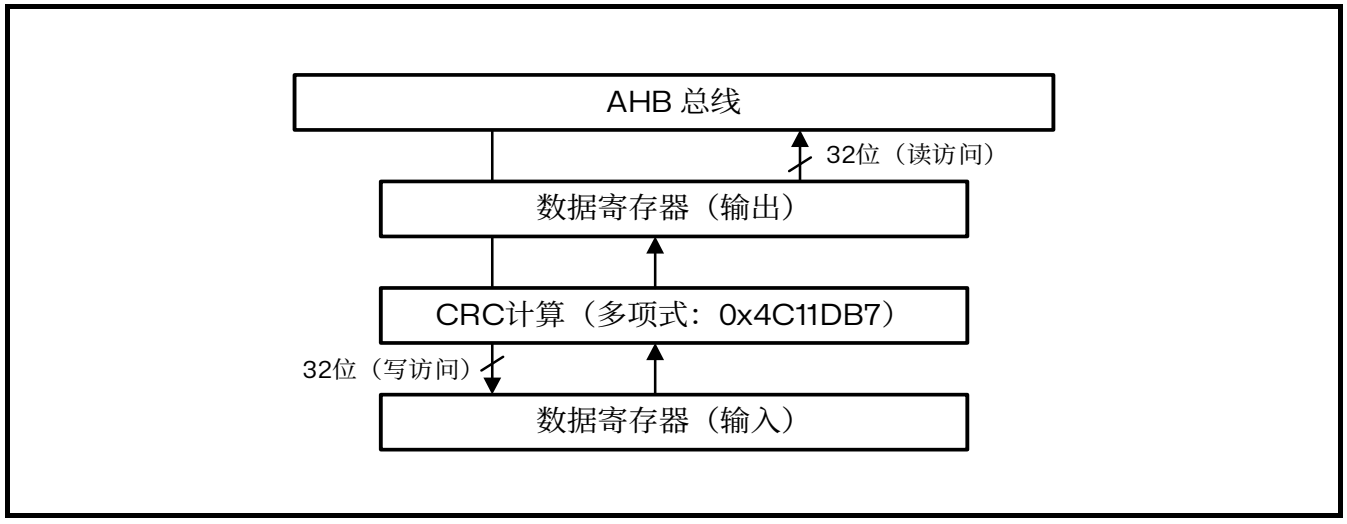


图 5.1 CRC 计算单元框图

5.3 CRC 功能说明

CRC 计算单元主要由单个 32 位数据寄存器组成，该寄存器：

- 用作输入寄存器，向 CRC 计算器中输入新数据（向寄存器写入数据时）
- 可保存之前的 CRC 计算结果（读取寄存器时）

对数据寄存器的每个写操作都会把当前新输入的数值和之前生成在数据寄存器中的 CRC 值再做一次 CRC 计算（CRC 计算针对整个 32 位数据字完成，而非逐字节进行）。

CRC 计算的时候，写操作被阻塞，因此允许执行背靠背写访问或连续的写读访问。

使用 CRC_CR 寄存器中的 RESET 控制位即可将 CRC 计算器复位为 0xFFFF FFFF。此操作不影响 CRC_IDR 寄存器的内容。

5.4 CRC 寄存器

CRC 计算单元包含两个数据寄存器和一个控制寄存器。CRC 寄存器必须按字（32 位）访问。

5.4.1 数据寄存器（CRC_DR）

偏移地址：0x00

复位值：0xFFFF FFFF

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

Bit 31:0 **DR[31:16]**：数据寄存器位
向 CRC 计算器写入新数据时用作输入寄存器。
读取寄存器时可读出之前的 CRC 计算结果。

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

Bit 31:8 保留，必须保持为复位值。

Bit 15:0 **IDR[15:0]**：通用 16 位数据寄存器位
可用作一个字节的临时存储单元。
此寄存器不受 CRC_CR 寄存器中 RESET 位产生的 CRC 复位影响。

5.4.3 控制寄存器（CRC_CR）

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

Bit 31:1 保留，必须保持为复位值。

Bit 0 **RESET**：
复位 CRC 计算单元并将数据寄存器设为 0xFFFF FFFF。
此位只能置 1，将由硬件自动进行清零。

6 电源控制（PWR）

6.1 概述

RX32H710 的工作电压（VDD/VDDA）为 2.5~3.6V。VDD 和 VDDA 必须保持一致。

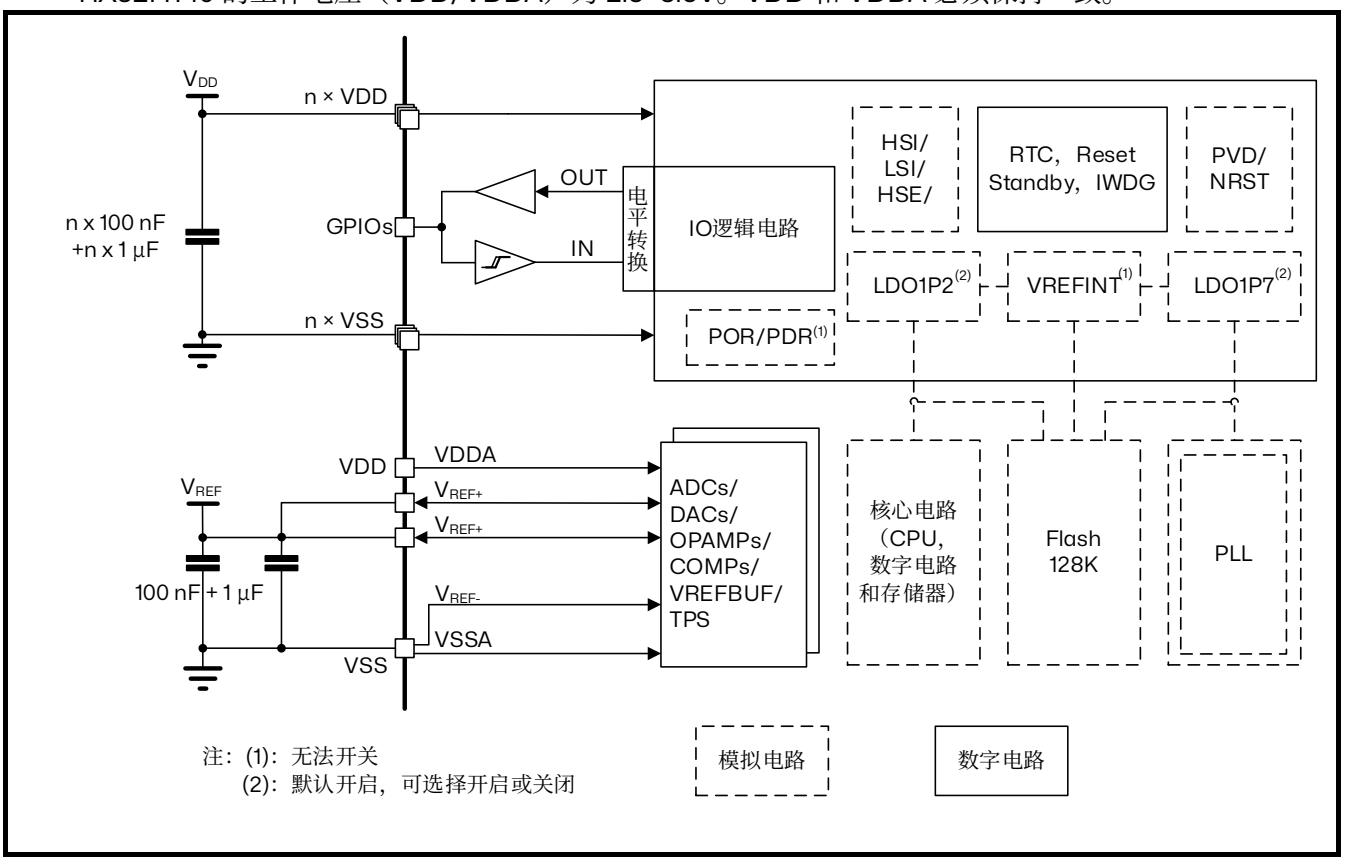


图 6.1 电源框图

6.2 电源管理器

6.2.1 上电复位（POR）和掉电复位（PDR）

RX32H710 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 2.5V 时，系统既能正常工作。

当 VDD/VDDA 低于指定的限位电压 V_{POR}/V_{PDR} 时，系统保持为复位状态，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。

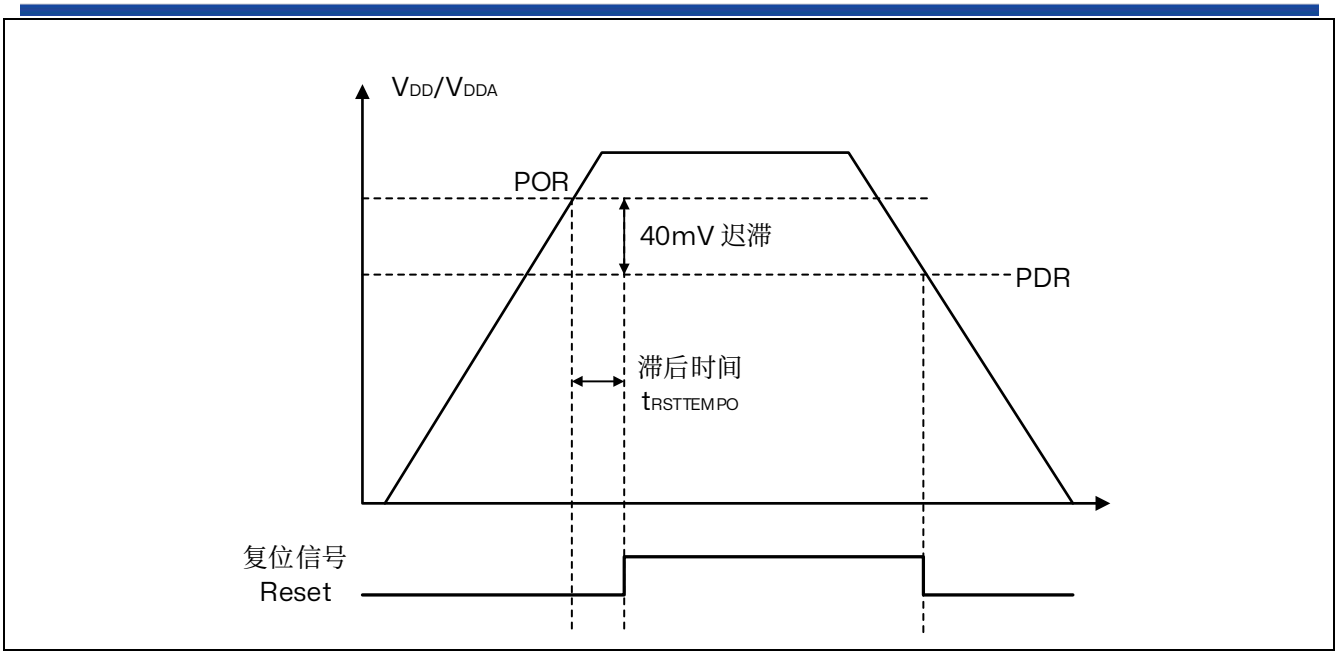


图 6.2 上电复位和掉电复位波形图

6.2.1 可编程电压检测器（PVD）

用户可以利用 PVD 对 VDD 电压与电源控制寄存器 2（PWR_CR2）中的 PLS[2:0]位进行比较来监控电源，这几位选择监控电压的阈值。通过设置 PVDE 位来使能 PVD。

电源状态寄存器 2（PWR_SR2）中的 PVDO 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线 16，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。

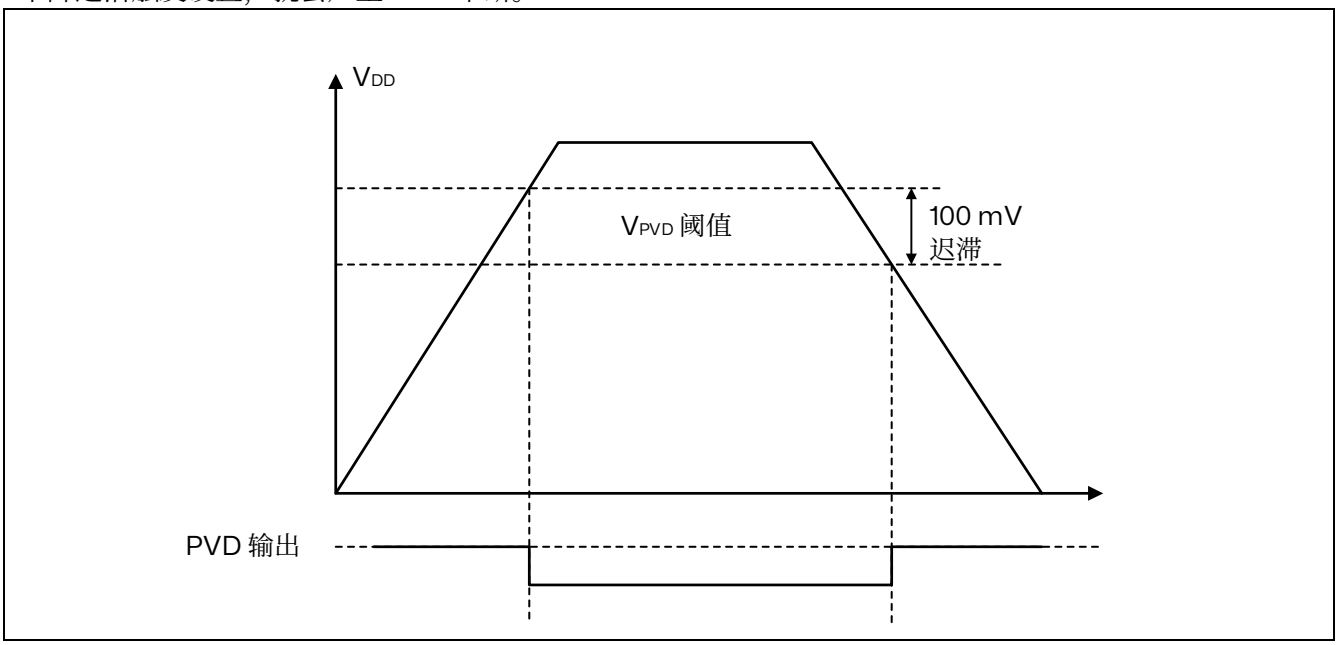


图 6.3 PVD 门限

6.3 低功耗模式

默认情况下，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（例如等待外部事件时），可以利用多种低功耗模式来节省功耗。用户需要根据最低电源消耗、最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。

RX32H710 支持三种低功耗模式，用户可以在以下模式中进行选择：

- 睡眠模式：CPU 时钟被关闭。所有外设继续运行，并且 CPU 可以被中断或者事件唤醒。
- 停止模式：保留 SRAM 和寄存器内容。关闭除 LSI 以外所有时钟。该系列可以被任一外部中断线从停止模式中唤醒。
- 待机模式：待机模式用于实现最低的功耗，关闭 LDO1P7 和 LDO1P2，关闭除 LSI 以外所有时钟。该系列可以被外部复位（NRST 引脚）、IWDG 复位、RTC 唤醒以及 WKUP 唤醒引脚的上升沿和下降沿从待机模式中唤醒。

*注：配置上升沿唤醒时，*WKUP *引脚不要外部上拉。配置下降沿唤醒时，*WKUP *引脚不要外部下拉。*

此外，可以通过以下方式降低运行模式的功耗：

- 降低系统时钟速度
- 当 APB 和 AHB 外设未使用时对它们进行时钟门控。

表 6.1 低功耗模式一览

模式	唤醒	唤醒后时钟	对时钟影响	LDO	
				1P7	1P2
睡眠 (立即睡眠或 退出时睡眠)	任一中断 唤醒事件	与进入睡眠模式前一致	CPU 时钟关闭	正常	正常
停止	任意外部中断 （在 EXTI 寄存器中配置） 指定外设事件	HSI	关闭除 LSI 以外所有时钟	正常	正常
待机	WKUP 引脚电平变化、 RTC 唤醒事件、 NRST 复位、 IWDG 复位	HSI	关闭除 LSI 以外所有时钟	关闭	关闭

调试模式

默认情况下，如果在进行调试微处理器时，使微处理器进入停止或待机模式，将失去调试连接。这是因为内核失去了时钟。

通过设置 DBGMCU_CR 寄存器中的对应配置位，可以在低功耗模式下调试软件。有关更多详细信息，请参阅章节：调试支持（DBG）。

6.3.1 运行模式

降低系统时钟速度

在运行模式下，可通过对预分频寄存器编程来降低系统时钟（SYSCLK、HCLK、PCLK）速度。进入睡眠模式之前，也可以使用这些预分频器降低外设速度。

更多详细信息，请参阅章节：时钟配置寄存器（RCC_CFGR）。

外设时钟门控

在运行模式下，可随时停止各外设和存储器的 HCLK 和 PCLK 以降低功耗。

要进一步降低睡眠模式的功耗，可在进入睡眠模式之前禁止外设时钟。

外设时钟门控由 RCC_AHBENR 和 RCC_APBxENR 寄存器控制。

6.3.2 睡眠模式

睡眠模式下的 I/O 状态

睡眠模式下，所有 I/O 保持与运行模式相同的状态。

6.3.3 停止模式

停止模式基于 32 位 RISC 内核深度睡眠模式基础上结合了外设的时钟控制机制。该模式下，内部 LDO 配置为正常模式。关闭除 LSI 以外所有时钟。内部 SRAM 和寄存器内容都将保留。

停止模式下的 I/O 状态

在停止模式下，所有 I/O 引脚保持与运行模式相同的状态。

6.3.4 待机模式

待机模式可以实现 MCU 的最低功耗。该模式是在 32 位 RISC 内核深度睡眠模式时关闭内部线性稳压器 LDO1P7 和 LDO1P2。PLL、HSI 和 HSE 振荡器也被关闭。

SRAM 和寄存器内容会丢失，除了待机电路中的寄存器。

待机模式下的 I/O 状态

在待机模式下，I/O 可以保持在模拟状态。

4 个唤醒引脚（WKUPx，x=1,2,3,4）也是可用的。

6.3.5 低功耗模式下的自动唤醒（AWU）

RTC 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）。RTC 提供一个可编程的时间基数，用于周期性从停止（0 或 1）或待机模式下唤醒。RTC 时钟源选择 LSI 可以实现此功能。为此，可以通过在 RTC 域控制寄存器（RCC_BDCR）中编程 RTCSEL[1:0]位来选择 LSI 作为 RTC 时钟源：

- 低功耗内部 RC 振荡器（LSI）
  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 RTC 闹钟事件将系统从停止模式下唤醒，必须进行如下操作：

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

如果要从待机模式中唤醒，不必配置外部中断线 17。

6.4 PWR 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。

6.4.1 电源控制寄存器（PWR_CR1）

地址偏移：0x00

复位值：0x0000 0000

从待机模式唤醒后，此寄存器将被重置。

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

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

Bit 31:9 保留，必须保持为复位值。

Bit 8 **DBP**：取消 RTC 的写保护

在复位后，RTC 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 RTC

1：允许写入 RTC

注：如果 RTC 的时钟是 HSE/128，该位必须保持为 '1'。

Bit 7:2 保留，必须保持为复位值。

Bit 1:0 **LPMS[1:0]**：低功耗模式选择

00：停止模式

01：保留

10：保留

11：待机模式

6.4.2 电源控制寄存器（PWR_CR2）

地址偏移：0x04

复位值：0x0000 0008

从待机模式唤醒后，此寄存器将被重置。

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

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

Bit 31:4 保留，必须保持为复位值。

Bit 3:1 **PLS[2:0]**：PVD 电平选择

这些位用于选择电源电压监测器的电压阈值

000：2.2V　　100：2.6V

001：2.3V　　101：2.7V

010：2.4V　　110：2.8V

011：2.5V　　111：2.9V

注：当 SYSCFG_CFGR2 寄存器的 PVDL 位被设置时此位被写保护。
此保护仅能通过系统复位来重置。

Bit 0 **PVDE**：电源电压监测器(PVD)使能

0：禁止 PVD

1：开启 PVD

注：当 SYSCFG_CFGR2 寄存器的 PVDL 位被设置时此位被写保护。
此保护仅能通过系统复位来重置。

6.4.3 电源控制寄存器（PWR_CR3）

地址偏移：0x08

复位值：0x0000 0000

退出待机模式以及 RCC_APB1RSTR1 寄存器中的 PWRRST 位被设置时，该寄存器不会被重置。

访问：访问此寄存器需要比标准 APB 访问更多的 APB 周期（写操作需要 3 个周期，读操作需要 2 个周期）。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIWUL	Reserved											EWUP4	EWUP3	EWUP2	EWUP1
rw												rw	rw	rw	rw

Bit 31:16 保留，必须保持为复位值。

Bit 15 **EIWUL**：使能内部唤醒线（RTC/IWDG）

0：禁止内部唤醒线唤醒

1：使能内部唤醒线

Bit 14:4 保留，必须保持为复位值。

Bit 3 **EWUP4**：使能 WKUP 引脚 4

0：WKUP 引脚 4 为通用 I/O。WKUP 引脚 4 上的事件不能将 CPU 从待机模式唤醒

1：WKUP 引脚 4 用于将 CPU 从待机模式唤醒，WKUP 引脚 4 上的上升沿/下降沿将系统从待机模式唤醒，有效边沿通过 PWR_CR4 的 WP4 设置。

Bit 2 **EWUP3**：使能 WKUP 引脚 3

0：WKUP 引脚 3 为通用 I/O。WKUP 引脚 3 上的事件不能将 CPU 从待机模式唤醒

1：WKUP 引脚 3 用于将 CPU 从待机模式唤醒，WKUP 引脚 3 上的上升沿/下降沿将系统从待机模式唤醒，有效边沿通过 PWR_CR4 的 WP3 设置。

Bit 1 **EWUP2**：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 上的上升沿/下降沿将系统从待机模式唤醒，有效边沿通过 PWR_CR4 的 WP2 设置。

Bit 0 **EWUP1**：使能 WKUP 引脚 1

0：WKUP 引脚 1 为通用 I/O。WKUP 引脚 1 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 上的上升沿/下降沿将系统从待机模式唤醒，有效边沿通过 PWR_CR4 的 WP1 设置。

6.4.4 电源控制寄存器（PWR_CR4）

地址偏移：0x0C
复位值：0x0000 0000
退出待机模式以及 RCC_APB1RSTR1 寄存器中的 PWRRST 位被设置时，该寄存器不会被重置。
访问：访问此寄存器需要比标准 APB 访问更多的 APB 周期（写操作需要 3 个周期，读操作需要 2 个周期）。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												WP4	WP3	WP2	WP1
												rw	rw	rw	rw

Bit 31:4 保留，必须保持为复位值。

Bit 3 **WP4：**WKUP4 引脚唤醒极性
此位定义了用于外部唤醒引脚上的事件检测极性，WKUP4
0：检测到高电平（上升沿）
1：检测到低电平（下升沿）

*注：配置上升沿唤醒时，*WKUP *引脚不要外部上拉。配置下降沿唤醒时，*WKUP *引脚不要外部下拉。*

Bit 2 **WP3：**WKUP3 引脚唤醒极性
此位定义了用于外部唤醒引脚上的事件检测极性，WKUP3
0：检测到高电平（上升沿）
1：检测到低电平（下升沿）

*注：配置上升沿唤醒时，*WKUP *引脚不要外部上拉。配置下降沿唤醒时，*WKUP *引脚不要外部下拉。*

Bit 1 **WP2：**WKUP2 引脚唤醒极性
此位定义了用于外部唤醒引脚上的事件检测极性，WKUP2
0：检测到高电平（上升沿）
1：检测到低电平（下升沿）

*注：配置上升沿唤醒时，*WKUP *引脚不要外部上拉。配置下降沿唤醒时，*WKUP *引脚不要外部下拉。*

Bit 0 **WP1：**WKUP1 引脚唤醒极性
此位定义了用于外部唤醒引脚上的事件检测极性，WKUP1
0：检测到高电平（上升沿）
1：检测到低电平（下升沿）

*注：配置上升沿唤醒时，*WKUP *引脚不要外部上拉。配置下降沿唤醒时，*WKUP *引脚不要外部下拉。*

6.4.5 电源状态寄存器 1（PWR_SR1）

地址偏移：0x10

复位值：0x0000 0000

退出待机模式以及 RCC_APB1RSTR1 寄存器中的 PWRRST 位被设置时，该寄存器不会被重置。

访问：与此标准 APB 读取相比，读取此寄存器需要额外的 2 个 APB 周期。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WUFI	Reserved						SBF	Reserved				WUF4	WUF3	WUF2	WUF1
r							r					r	r	r	r

Bit 31:16 保留，必须保持为复位值。

Bit 15 **WUFI**：内部唤醒标志（IWDG/RTC 唤醒）

当在内部唤醒线上检查到唤醒时，设置此位。当所有内部唤醒源被清除时该位被清除。

Bit 14:9 保留，必须保持为复位值。

Bit 8 **SBF**：待机标志

该位由硬件设置，并只能由 POR/PDR（上电/掉电复位）或设置电源状态清除寄存器（PWR_SCR）的 CSBF 位清除。

0：系统不在待机模式

1：系统进入待机模式

Bit 7:4 保留，必须保持为复位值。

Bit 3 **WUF4**：WKUP4 引脚唤醒标志

当在唤醒引脚 WKUP4 检测到一个唤醒事件时此位被设置，被清除通过向电源状态清除寄存器（PWR_SCR）的 CWUF4 位写“1”。

Bit 2 **WUF3**：WKUP3 引脚唤醒标志

当在唤醒引脚 WKUP3 检测到一个唤醒事件时此位被设置，被清除通过向电源状态清除寄存器（PWR_SCR）的 CWUF3 位写“1”。

Bit 1 **WUF2**：WKUP2 引脚唤醒标志

当在唤醒引脚 WKUP2 检测到一个唤醒事件时此位被设置，被清除通过向电源状态清除寄存器（PWR_SCR）的 CWUF2 位写“1”。

Bit 0 **WUF1**：WKUP1 引脚唤醒标志

当在唤醒引脚 WKUP1 检测到一个唤醒事件时此位被设置，被清除通过向电源状态清除寄存器（PWR_SCR）的 CWUF1 位写“1”。

6.4.6 电源状态寄存器 2（PWR_SR2）

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
Reserved															

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

Bit 31:12 保留，必须保持为复位值。

Bit 11 **PVDO**：PVD 输出

当 PVD 被 PVDE 位使能后该位才有效

0：V_{DD}/V_{DDA} 高于由 PLS[2:0]选定的 PVD 阈值

1：V_{DD}/V_{DDA} 低于由 PLS[2:0]选定的 PVD 阈值

*注：在待机模式下*PVD *被停止。因此，待机模式后或复位后，直到设置*PVDE *位之前，该位为*0。

Bit 10:0 保留，必须保持为复位值。

6.4.7 电源状态清除寄存器（PWR_SCR）

地址偏移：0x18

复位值：0x0000 0000

访问：与此标准 APB 写入相比，写入此寄存器需要额外的 3 个 APB 周期。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CWUFI	Reserved						CSBF	Reserved				CWUF4	CWUF3	CWUF2	CWUF1
w							w					w	w	w	w

Bit 31:16 保留，必须保持为复位值。

Bit 15 **CWUFI**：清除内部唤醒标志

写 1 清除 PWR_SR1 寄存器的内部唤醒标志 WUFI

Bit 14:9 保留，必须保持为复位值。

Bit 8 **CSBF**：清除待机标志

写 1 清除 PWR_SR1 寄存器的待机标志 SBF

Bit 7:4 保留，必须保持为复位值。

Bit 3 **CWUF4**：清除唤醒标志 4

写 1 清除 PWR_SR1 寄存器的 WKUP4 引脚唤醒标志 WUF4

Bit 2 **CWUF3**：清除唤醒标志 3

写 1 清除 PWR_SR1 寄存器的 WKUP3 引脚唤醒标志 WUF3

Bit 1 **CWUF2**：清除唤醒标志 2

写 1 清除 PWR_SR1 寄存器的 WKUP2 引脚唤醒标志 WUF2

Bit 0 **CWUF1**：清除唤醒标志 1

写 1 清除 PWR_SR1 寄存器的 WKUP1 引脚唤醒标志 WUF1

7 复位和时钟控制（RCC）

7.1 复位

有三种类型的复位，分别定义为系统复位、电源复位和 RTC 域复位。

7.1.1 电源复位

当发生以下任一事件时，会产生电源复位：

1. 上电复位（POR）或掉电复位（PDR 复位）
2. 退出待机模式时

上电或掉电复位（POR/PDR），会将所有寄存器设置为其复位值，RTC 域除外。

当退出待机模式时，V_{CORE} 域中的所有寄存器都会被设置为其复位值。V_{CORE} 域之外的寄存器（RTC、WKUP、IWDG 以及待机模式控制）不会受到影响。

7.1.2 系统复位

系统复位会将所有寄存器设置为其复位值，时钟控制/状态寄存器（RCC_CSR）中的复位标志和 RTC 域中的寄存器除外。

系统复位在发生以下任一事件时生成：

1. NRST 引脚出现低电平（外部复位）
2. 独立看门狗事件（IWDG 复位）
3. 软件复位（SW 复位）（见软件复位）
4. 低功耗模式安全复位（见低功耗模式安全复位）
5. 选项字节加载复位（见选项字节加载复位）

可以通过检查控制/状态寄存器 RCC_CSR 中的复位标志来识别复位源。

NRST 引脚（外部复位）

引脚上的任何有效复位信号都会传播到设备内部逻辑，并且所有内部复位源都会通过脉冲发生器外部驱动至该引脚。脉冲发生器保证每个内部复位源在 NRST 引脚上输出时具有至少 20µs 的最小复位脉冲持续时间。

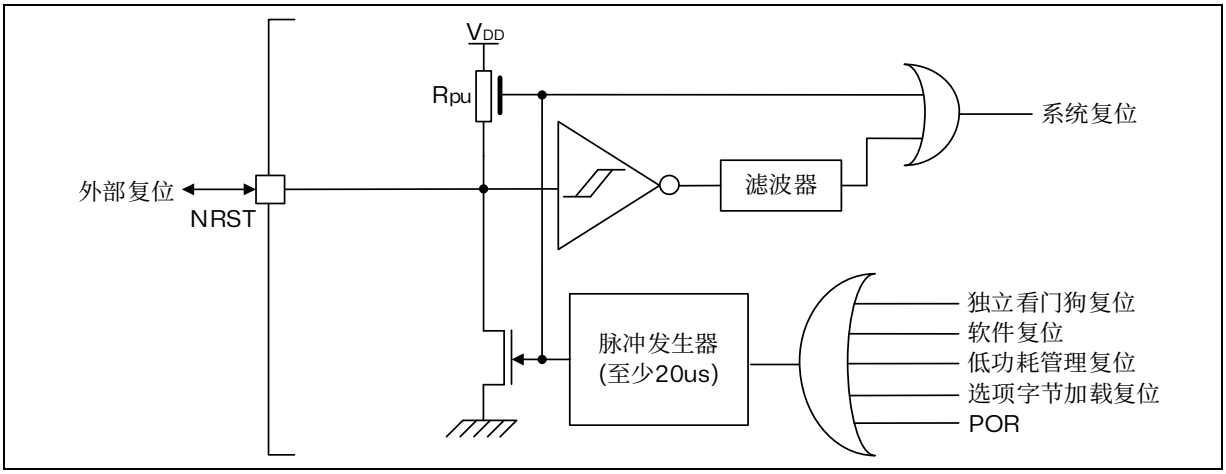


图 7.1 复位电路框图

低功耗模式安全复位

为防止关键应用错误地进入低功耗模式，提供了两种低功耗模式安全复位。如果在选项字节中使能，则在以下条件下生成复位：

1. 进入待机模式：通过清零用户选项字节中的 nRST_STDBY 位来使能此类型复位。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. 进入停止模式：通过清零用户选项字节中的 nRST_STOP 位来使能此类型复位。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。

有关用户选项字节的更多信息，请参阅 4.5.1 节：选项字节描述。

选项字节加载复位

当 FLASH_CR 寄存器中的 OBL_LAUNCH 位（第 27 位）被置位时，会生成选项字节加载复位。该位用于通过软件启动选项字节加载。

7.1.3 RTC 域复位

RTC 域有两种特定的复位。

当发生以下任一事件时，会生成 RTC 域复位：

1. 软件复位，通过设置 RTC 域控制寄存器（RCC_BDCR）中的 BDRST 位来触发。
2. 在 V_{DD} 掉电的前提下，V_{DD} 上电将引发 RTC 域复位。

RTC 域复位仅影响 RTC 和 RCC RTC 域控制寄存器。

7.2 时钟

四种不同的时钟源可被用来驱动系统时钟（SYSCLK）：

- LSI（低速内部）32KHz RC 振荡器时钟
- HSI（高速内部）16 MHz RC 振荡器时钟
- HSE 振荡器时钟，频率 16 MHz
- PLL 时钟

复位后启动，HSI 被用作系统时钟源。

同时设备的 LSI 还作为附加时钟源：

- 32 kHz 低速内部 RC（LSI RC），驱动独立看门狗，并可选择驱动用于从停止和待机模式自动唤醒的实时时钟（RTC）。

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。

可使用多个预分频器来配置 AHB 频率、APB1 和 APB2 域。AHB 和 APB2 域的最大频率为 152MHz，APB1 域的最大频率为 80MHz。

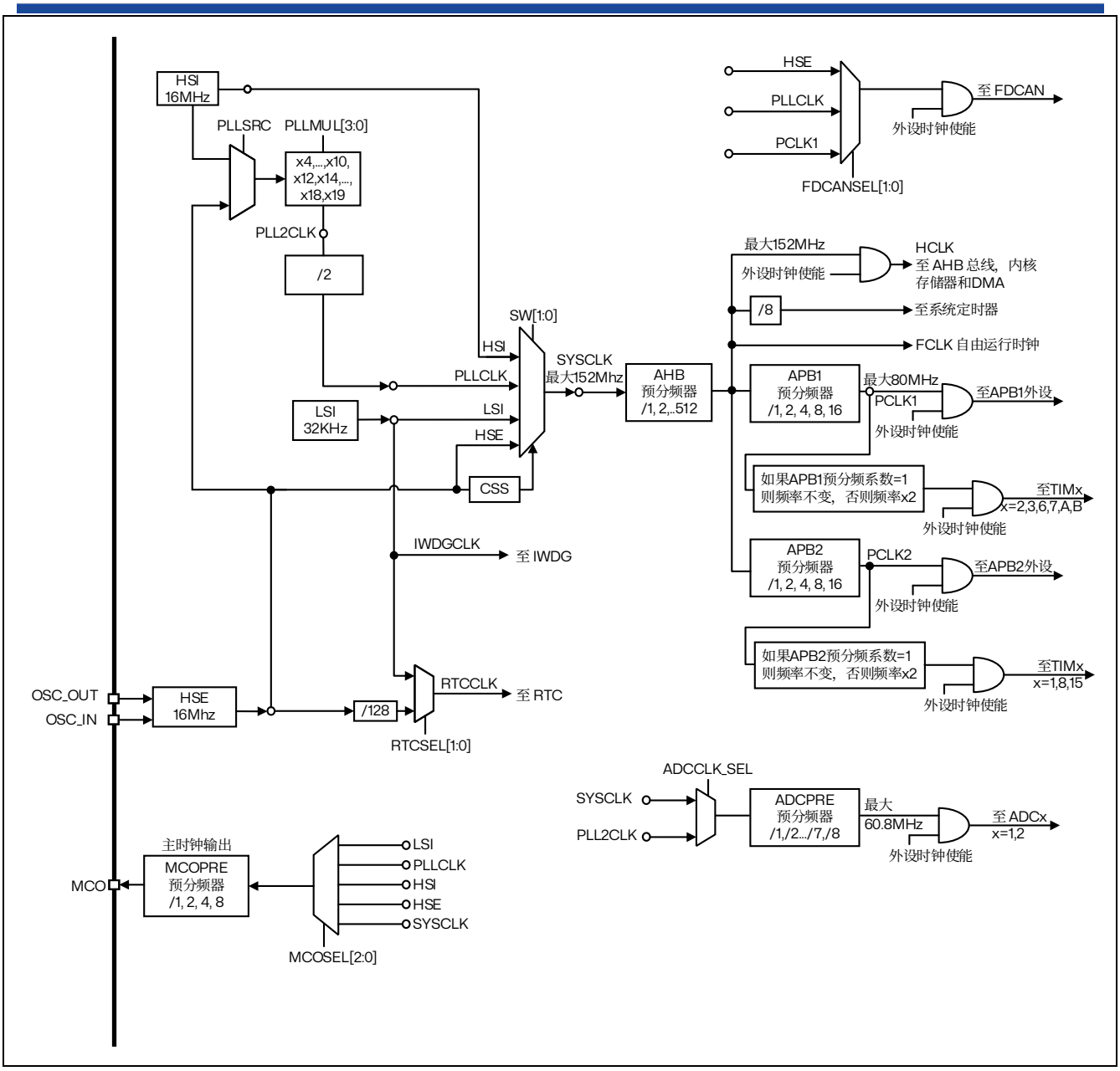
除以下情况外，所有外设时钟均源自其总线时钟（HCLK、PCLK1 或 PCLK2）：

- ADC 时钟，可通过软件从以下源中选择一个来派生：
  - 系统时钟（SYSCLK）
  - PLL2CLK 时钟
- UART 时钟，可通过软件从以下两个源中选择一个来派生：
  - APB1 或 APB2 时钟（根据 UART 映射到哪个 APB，选择 PCLK1 或 PCLK2）
- I2C 时钟，可通过软件从以下源中选择一个来派生：
  - APB1 时钟（PCLK1）
- RTC 时钟，可通过软件从以下两个源中选择一个来派生：
  - LSI 时钟
  - HSE 时钟除以 128

  仅当时钟为 LSI 时，才支持停止模式（包括唤醒）功能。
- IWDG 时钟始终为 LSI 时钟。
- FDCAN 时钟，可通过软件从以下两个源中选择一个来派生：
  - HSE 时钟
  - PLLCLK 时钟
  - PCLK1 时钟

RCC 通过 AHB 时钟（HCLK）8 分频后作为内核系统定时器的外部时钟。

FCLK 作为 32 位 RISC 内核的自由运行时钟。



1. 有关内部和外部时钟源特性的详细信息，请参阅设备数据手册中的“电气特性”部分。

图 7.2 时钟树

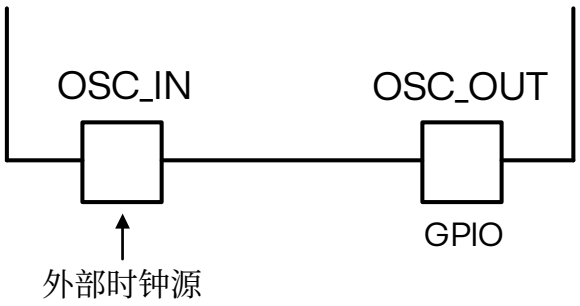
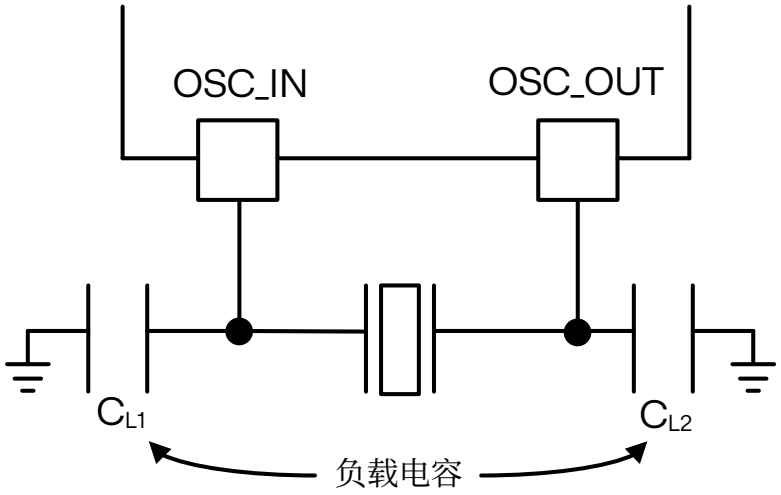
7.2.1 HSE 时钟

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 7.3 HSE 时钟源

时钟源	硬件配置
外部时钟	
晶体/陶瓷谐振器	

外部晶体/陶瓷谐振器（HSE 晶体）

16 MHz 的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上图，进一步信息可参考数据手册的电气特性部分。

时钟控制寄存器（RCC_CR）中的 HSERDY 标志指示 HSE 振荡器是否稳定。启动时，直到此位由硬件设置，时钟才会被释放。

可以使用时钟控制寄存器（RCC_CR）中的 HSEON 位来打开和关闭 HSE 晶体。

外部源（HSE 旁路）

在此模式下，必须提供外部时钟源。其频率可达 16MHz。通过设置时钟控制寄存器（RCC_CR）中的 HSEBYP 和 HSEON 位来选择此模式。外部时钟信号（方波、正弦波或三角波）需具有约 40-60%的占空比（具体取决于频率，请参阅数据手册），并驱动 OSC_IN 引脚，而 OSC_OUT 引脚可用作 GPIO。见上图。

7.2.2 HSI 时钟

HSI 时钟信号由内部 16 MHz 的 RC 振荡器生成。

HSI RC 振荡器的优势在于以低成本（无需外部组件）提供时钟源。其启动时间也比 HSE 晶体振荡器快，然而，即使经过校准，其频率准确性仍低于外部晶体振荡器或陶瓷谐振器。

HSI 时钟可在从停止模式（Stop）唤醒后选择为系统时钟。请参阅第 7.3 节：低功耗模式。如果 HSE 晶体振荡器出现故障，它也可以作为备用时钟源（辅助时钟）。请参阅第 7.2.6 节：时钟安全系统（CSS）。

校准

由于制造工艺的差异，RC 振荡器频率在不同芯片之间可能会有所不同，因此睿兴会在工厂对每颗芯片进行校准，以确保在 T_A=25°C 时具有 1%的准确性。

复位后，工厂校准值会被加载到内部时钟源校准寄存器（RCC_ICSCR）的 HSICAL[8:0]位中。

时钟控制寄存器（RCC_CR）中的 HSIRDY 标志指示 HSI 振荡器是否稳定。启动时，直到此位由硬件设置，HSI 输出时钟才会被释放。

可以使用时钟控制寄存器（RCC_CR）中的 HSION 位来打开和关闭 HSI 振荡器。

如果 HSE 晶体振荡器出现故障，HSI 信号也可以作为备用源（辅助时钟）。请参阅第 7.2.6 节：时钟安全系统（CSS）。

7.2.3 PLL 内部锁相环

内部锁相环 PLL 可用于将 HSI 或 HSE 的输出时钟频率进行倍频。PLL 的输入频率必须在设备数据手册中定义的范围内。所选时钟源会通过可编程倍频因子 PLLMUL（4 至 10，12，14 至 19）进行倍频，以提供所需输入范围内的时钟频率。请参阅图：时钟树和时钟配置寄存器（RCC_CFGR）。

必须在使能 PLL 之前完成 PLL 配置（包括输入时钟选择和倍频因子的设定）。一旦 PLL 被使能，这些参数便无法更改。如需修改 PLL 配置，请按照以下步骤操作：

1. 在时钟控制寄存器（RCC_CR）中将 PLLON 设置为 0，以禁用 PLL。
2. 等待 PLLRDY 被清除。此时 PLL 已完全停止。
3. 更改所需参数。
4. 在时钟控制寄存器（RCC_CR）中将 PLLON 设置为 1，以再次使能 PLL。

如果在时钟中断寄存器（RCC_CIR）中使能，当 PLL 准备就绪时，可以生成中断。

PLL 的输出频率不得超过 152 MHz。

如果 PLLCLK 被用作系统时钟，则不能清除 PLLON。

7.2.4 LSI 时钟

LSI RC 作为低功耗时钟源，可在停止和待机模式下为独立看门狗（IWDG）和实时时钟（RTC）持续运行。时钟频率为 32KHz。有关更多详细信息，请参阅数据手册中的电气特性部分。

LSI RC 可通过控制/状态寄存器（RCC_CSR）中的 LSION 位进行开启和关闭。

控制/状态寄存器（RCC_CSR）中的 LSIRDY 标志用于指示 LSI 振荡器是否稳定。在系统启动时，直到此位由硬件设置，时钟才会被释放。如果在时钟中断寄存器（RCC_CIR）中使能，可以生成中断。

7.2.5 系统时钟（SYSCLK）选择

系统时钟（SYSCLK）可由四种不同的时钟源驱动：

- LSI 振荡器
- HSI 振荡器
- HSE 振荡器
- PLL 锁相环

系统时钟的最大频率为 152 MHz。系统复位后，默认选择 HSI 振荡器作为系统时钟。当某个时钟源被直接用作或通过 PLL 用作系统时钟时，该时钟源无法停止。

仅当目标时钟源准备就绪（启动延迟后时钟稳定或 PLL 锁定）时，才能从一个时钟源切换到另一个时钟源。如果选择了尚未准备就绪的时钟源，则会在该时钟源准备就绪时进行切换。时钟控制寄存器（RCC_CR）和控制/状态寄存器（RCC_CSR）中的状态位指示哪些时钟已准备就绪。时钟配置寄存器（RCC_CFGR）中的 SWS[1:0]系统时钟切换状态指示当前正在使用哪个时钟作为系统时钟。

建议从低速系统时钟切换到高速系统时钟，或从高速切换到低速时，使用中等速度时钟作为过渡状态，持续至少 1μs。

时钟源切换条件：

- 从 HSE 或 HSI 切换到 AHB 频率（HCLK）高于 80 MHz 的 PLL
- 从 HCLK 高于 80 MHz 的 PLL 切换到 HSE 或 HSI

过渡状态：

- 将 AHB 预分频器 HPRE[3:0]位设置为将系统频率除以 2
- 将系统时钟切换到 PLL
- 等待至少 1μs，然后重新配置 AHB 预分频器位以获得所需的 HCLK 频率

7.2.6 时钟安全系统（CSS）

时钟安全系统可通过软件激活。在此情况下，时钟检测器在 HSE 振荡器启动延迟后使能，并在该振荡器停止时禁用。若检测到 HSE 时钟故障，HSE 振荡器将自动禁用，同时向高级控制定时器（TIM1/TIM8 及 TIM15）的中断输入发送时钟故障事件，并生成中断以通知软件出现故障（时钟安全系统中断 CSSI），从而使微控制器（MCU）能够执行救援操作。CSSI 与 32 位 RISC 内核的非屏蔽中断相关联。

注：一旦使能 CSS 且 HSE 时钟出现故障，将触发 CSS 中断并自动生成非屏蔽中断。非屏蔽中断将无限执行，除非清除 CSS 中断挂起位。因此，在非屏蔽中断中断服务例程（ISR）中，用户必须通过设置时钟中断寄存器（RCC_CIR）中的 CSSC 位来清除 CSS 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。

7.2.7 ADC 时钟

ADC 时钟源自系统时钟或 PLL2CLK 输出。

有关 ADC 时钟的最大值，请参阅设备数据手册。

7.2.8 RTC 时钟

RTCCLK 时钟源可以是 HSE/128 或 LSI 时钟。通过编程 RTC 域控制寄存器（RCC_BDCR）中的 RTCSEL[1:0]位来选择时钟源。除非 RTC 域复位，此选择不能被改变。为确保 RTC 正常工作，系统必须始终配置为使得 PCLK 频率大于或等于 RTCCLK 频率。

HSE 和 LSI 时钟不在 RTC 域内。因此：

- 若选择 LSI 作为 RTC 时钟：
  - 若 V_{DD} 供电关闭，则 RTC 状态无法保证。
- 若使用经过预分频的 HSE 时钟作为 RTC 时钟：
  - 若 V_{DD} 供电关闭或内部电压调节器关闭（从 V_{CORE} 域断电），则 RTC 状态无法保证。

当 RTC 时钟为 LSI 时，在系统复位期间，RTC 仍保持时钟供应并处于功能状态。

7.2.9 定时器时钟

定时器时钟频率由 APB 域频率和 APB 预分频值共同决定。若 APB 预分频值为 1，则定时器时钟频率被设置为与 APB 域相同的频率；否则，定时器时钟频率被设置为 APB 域频率的 2 倍。

7.2.10 看门狗时钟

如果独立看门狗（IWDG）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。

7.2.11 时钟输出

- MCO
  微控制器时钟输出（MCO）功能允许将时钟信号输出到外部的 MCO 引脚。
  可从五个时钟信号中选择一个作为 MCO 时钟：
  - LSI
  - SYSCLK
  - HSI
  - HSE
  - PLLCLK

  选择由时钟配置寄存器（RCC_CFGR）中的 MCOSEL[2:0]位域控制。所选时钟可通过时钟配置寄存器（RCC_CFGR）中的 MCOPRE[1:0]位域进行分频。

MCO 时钟输出需要 MCO 引脚上选择相应的复用功能。

7.2.12 使用 TIM15/TIM2/TIM3/TIMA/TIMB 进行内部/外部时钟测量

可通过 TIM15 的通道 3 输入捕获功能间接测量部分板上时钟源的频率，如下列图所示。

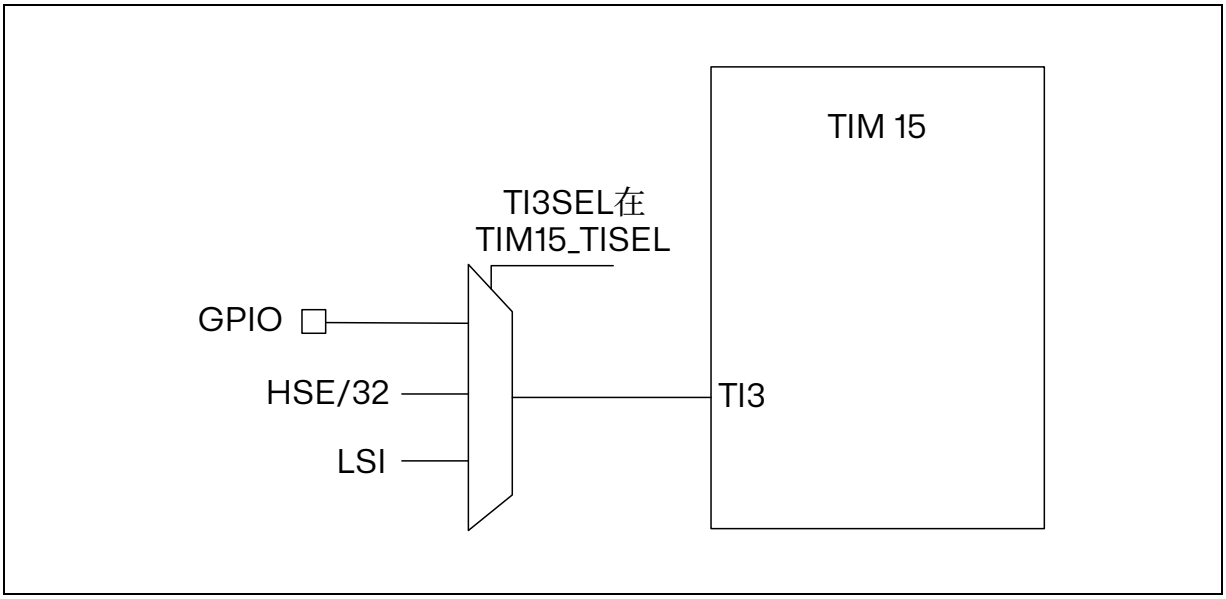


图 7.4 使用 TIM15 捕获模式进行频率测量

定时器 15 的输入捕获通道可以是一个 GPIO 引脚或 MCU 的内部时钟。可能性如下：

- TIM15 通道 3 连接到 GPIO。请参考设备数据手册中的复用功能映射。
- TIM15 通道 3 连接到 HSE/32 时钟。
- TIM15 通道 3 连接到 LSI 时钟。

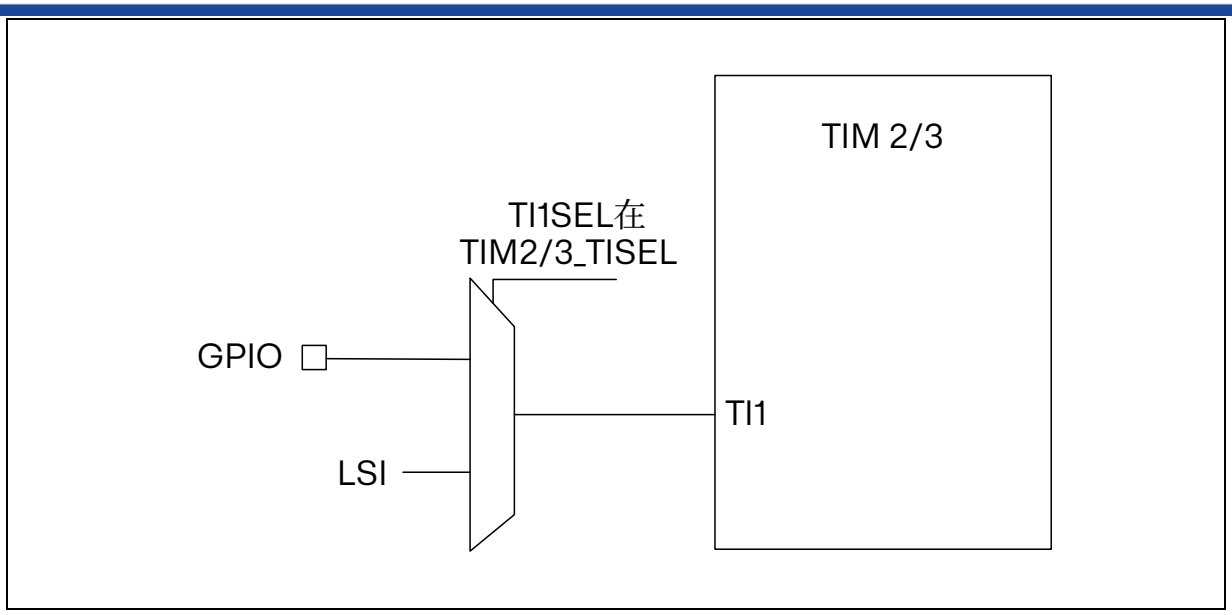


图 7.5 使用 TIM2/TIM3 捕获模式进行频率测量

定时器 2 和 3 的输入捕获通道可以是一个 GPIO 引脚或 MCU 的内部时钟。可能性如下:

- TIM2/TIM3 通道 1 连接到 GPIO。请参考设备数据手册中的复用功能映射。
- TIM2/TIM3 通道 1 连接到 LSI 时钟。

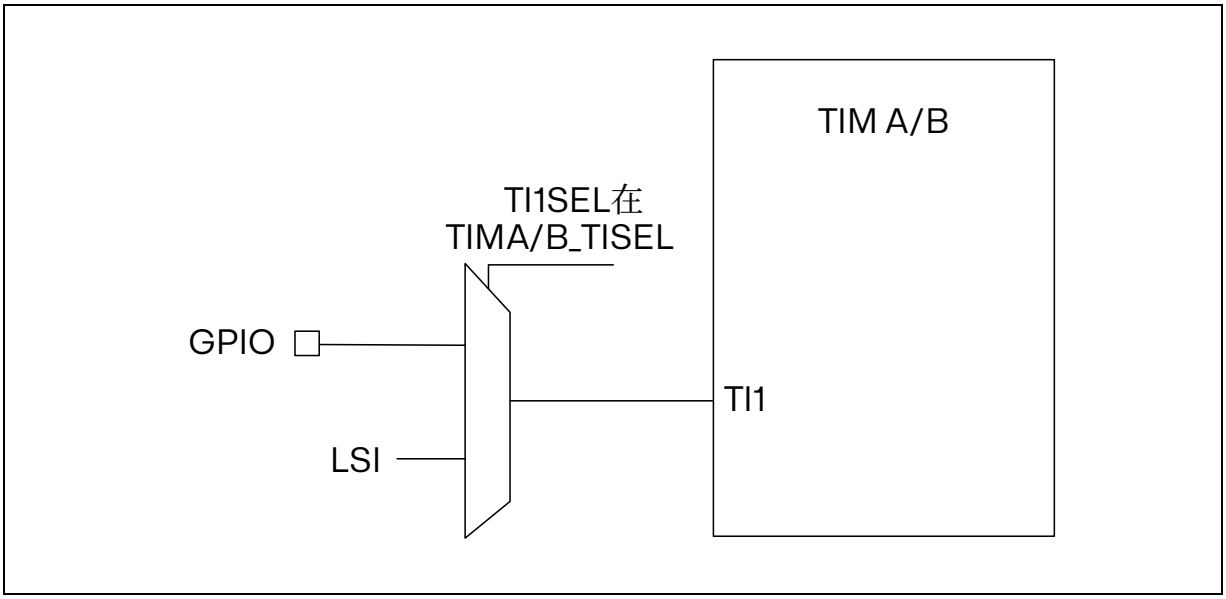


图 7.6 使用 TIMA/TIMB 捕获模式进行频率测量

定时器 A 和 B 的输入捕获通道可以是一个 GPIO 引脚或 MCU 的内部时钟。可能性如下:

- TIMA/TIMB 通道 1 连接到 GPIO。请参考设备数据手册中的复用功能映射。
- TIMA/TIMB 通道 1 连接到 LSI 时钟。

7.2.13 外设时钟使能寄存器（RCC_AHBENR, RCC_APBxENR)

每个外设时钟都可以通过 RCC_AHBENR、RCC_APBxENR 寄存器中的 xxxxEN 位来使能。

当外设时钟未激活时，不支持对外设寄存器的读或写访问。

使能位具有同步机制，以用来为外设创建无抖动时钟。设置使能位后，时钟在激活前有 2 个时钟周期的延迟。

注：在使能外设时钟后，软件必须等待一段延迟时间，然后才能访问外设寄存器。

7.3 低功耗模式

- AHB 和 APB 外设时钟（包括 DMA 时钟）可通过软件禁用。
- 睡眠模式会关闭 CPU 时钟。
- 停止模式会关闭除 LSI 以外所有时钟。
- 待机模式会关闭除 LSI 以外所有时钟。

可以通过设置 DBGMCU_CR 寄存器中的 DBG_STOP 或 DBG_STANDBY 位来覆盖 CPU 的深度睡眠模式，以便进行调试。

当退出停止模式（停止或待机）时，系统时钟为 HSI。

如果正在进行 Flash 存储器编程操作，则停止模式、待机模式的进入会延迟，直到 Flash 存储器接口访问完成。如果正在访问 APB 域，则停止模式、待机模式的进入会延迟，直到 APB 访问完成。

7.4 RCC 寄存器

7.4.1 时钟控制寄存器（RCC_CR）

偏移地址：0x00
复位值：0x0010 0003
HSEBYP 不受复位影响
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	HSE_CTR	HSI_CTR[1:0]		Reserved		PLL RDY	PLLON	Reserved			HSE BIAS	CSSON	HSE BYP	HSE RDY	HSEON
	rw	rw	rw			r	rw				rw	rs	rw	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														HSI RDY	HSION
														r	rw

Bit 31 保留，必须保持为复位值。

Bit 30 **HSE_CTR**：HSE 时钟滤波控制位
0：0.0ns（默认）
1：5ns

Bit 29:28 **HSI_CTR**：HSI 时钟滤波控制位
00：0.0ns（默认）
01：0.9ns
10：4.1ns
11：6.1ns

Bit 27:26 保留，必须保持为复位值。

Bit 25 **PLLRDY**：主 PLL 时钟就绪标志
由硬件设置，表示主 PLL 已锁定。
0：PLL 未锁定
1：PLL 已锁定

Bit 24 **PLLON**：主 PLL 使能
由软件设置和清除以启用主 PLL。
在进入停止和待机模式时由硬件清除。如果 PLL 时钟用作系统时钟，则无法复位此位。
0：PLL 关闭
1：PLL 使能

Bit 23:21 保留，必须保持为复位值。

Bit 20 **HSEBIAS**：HSE 起振电阻控制
0：断开 HSE 起振电阻
1：连接 HSE 起振电阻（默认）

Bit 19 **CSSON**：时钟安全系统使能

由软件设置以启用时钟安全系统。当 CSSON 设置时，如果 HSE 振荡器就绪，则硬件使能时钟检测器，如果检测到 HSE 时钟故障，则硬件禁用时钟检测器。此位仅可设置，复位时清除。

0：时钟安全系统关闭（时钟检测器关闭）

1：时钟安全系统使能（如果 HSE 振荡器稳定，则时钟检测器开启；否则关闭）

Bit 18 **HSEBYP**：HSE 晶体振荡器旁路

由软件设置和清除以使用外部时钟旁路振荡器。外部时钟必须在 HSEON 位设置后才能被设备使用。仅当 HSE 振荡器禁用时，才可以写入 HSEBYP 位。

0：HSE 晶体振荡器未旁路

1：HSE 晶体振荡器使用外部时钟旁路

Bit 17 **HSERDY**：HSE 时钟就绪标志

由硬件设置，表示 HSE 振荡器已稳定。

0：HSE 振荡器未就绪

1：HSE 振荡器就绪

注：一旦 HSEON *位清除，* HSERDY *将在* 6 *个* HSE *时钟周期后变为低电平。*

Bit 16 **HSEON**：HSE 时钟使能

由软件设置和清除。

在进入停止和待机模式时由硬件清除以停止 HSE 振荡器。如果 HSE 振荡器直接或间接用作系统时钟，则无法复位此位。

0：HSE 振荡器关闭

1：HSE 振荡器使能

Bit 15:2 保留，必须保持为复位值。

Bit 1 **HSIRDY**：HSI 时钟就绪标志

由硬件设置，表示 HSI 振荡器已稳定。此位仅在通过设置 HSION 由软件使能 HSI 时设置。

0：HSI 振荡器未就绪

1：HSI 振荡器已就绪

注：一旦 HSION *位清除，* HSIRDY *将在* 6 *个* HSI *时钟周期后变为低电平。*

Bit 0 **HSION**：HSI 时钟使能

由软件设置和清除。

在进入停止和待机模式时由硬件清除以停止 HSI 振荡器。

在离开停止模式或者 HSE 晶体振荡器发生故障，由硬件设置强制 HSI 振荡器开启。

如果 HSI 直接或间接用作系统时钟，则由硬件设置此位。

0：HSI 振荡器关闭

1：HSI 振荡器使能

7.4.2 时钟配置寄存器（RCC_CFGR）

偏移地址：0x04
复位值：0x8000 C000
访问：0≤等待周期≤2，支持字，半字和字节访问
仅在时钟源切换期间插入 1 或 2 个等待状态；
若访问发生在 APB 或 AHB 预分频器值更新过程中，则插入 0 至 15 个等待状态。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADC PRE[2]	Res	ADCCLK_SEL	MCOPRE[1:0]		MCOSEL[2:0]			Res	FDCANSEL[1:0]		PLLMUL[3:0]				PLLSRC
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
ADCPRE[1:0]		PPRE2[2:0]			PPRE1[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

Bit 31 **ADCPRE[2]**：ADC 时钟预分频
同 ADCPRE[1:0]说明。

Bit 30 保留，必须保持为复位值。

Bit 29 **ADCCLK_SEL**：ADCCLK 时钟选择
软件置位或清零来选择 ADC 时钟源。
0：SYSCLK 时钟分频后作为 ADC 时钟
1：PLL2CLK 时钟作为 ADC 时钟

Bit 28:27 **MCOPRE[1:0]**：微控制器时钟输出预分频
这些位由软件设置和清除。
强烈建议在使能 MCO 输出之前更改此预分频器的值。
00：MCO 不分频
01：MCO 分频为 2
10：MCO 分频为 4
11：MCO 分频为 8

Bit 26:24 **MCOSEL[2:0]**：微控制器时钟输出选择
这些位由软件设置和清除。
000：MCO 输出禁用，MCO 无时钟
001：选择 SYSCLK 系统时钟
010：保留
011：选择 HSI 时钟
100：选择 HSE 时钟
101：选择 PLL 时钟
110：选择 LSI 时钟
其他：保留

*注：在启动或*MCO *时钟源切换期间，此时钟输出可能会有一些截断周期。*

Bit 23 保留，必须保持为复位值。

Bit 22:21 **FDCANSEL[1:0]**：FDCAN 时钟选择

00：HSE

01：PLLCLK

10：PCLK1

11：保留

Bit 20:17 **PLLMUL[3:0]**：PLL 倍频系数选择

0000：保留	1000：10 倍频输出
0001：保留	1001：12 倍频输出
0010：4 倍频输出	1010：14 倍频输出
0011：5 倍频输出	1011：15 倍频输出
0100：6 倍频输出	1100：16 倍频输出
0101：7 倍频输出	1101：17 倍频输出
0110：8 倍频输出	1110：18 倍频输出
0111：9 倍频输出	1111：19 倍频输出

Bit 16 **PLLSRC**：PLL 时钟选择

此位由软件设置和清除来选择 PLL 输入时钟源。

0：HSI

1：HSE

注：只能在关闭 PLL *时才能写入此位。*

Bit 31, 15:14 **ADCPRE[2:0]**：ADC 时钟预分频

此位由软件设置和清除来选择 ADC 时钟频率。

000：ADCCLK 不分频

001：ADCCLK 2 分频后作为 ADC 时钟

010：ADCCLK 3 分频后作为 ADC 时钟

011：ADCCLK 4 分频后作为 ADC 时钟

100：ADCCLK 5 分频后作为 ADC 时钟

101：ADCCLK 6 分频后作为 ADC 时钟

110：ADCCLK 7 分频后作为 ADC 时钟

111：ADCCLK 8 分频后作为 ADC 时钟

Bit 13:11 **PPRE2[2:0]**：APB2 预分频

由软件设置和清除，用于控制 APB2 时钟（PCLK2）的分频系数。

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

Bit 10:8 **PPRE1[2:0]**：APB1 预分频

由软件设置和清除，用于控制 APB1 时钟（PCLK1）的分频系数。

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

Bit 7:4 **HPRE[3:0]**：AHB 预分频

由软件设置和清除，用于控制 AHB 时钟的分频系数。

0xxx：SYSCLK 不分频

1000：SYSCLK 2 分频

1001：SYSCLK 4 分频

1010：SYSCLK 8 分频

1011：SYSCLK 16 分频

1100：SYSCLK 64 分频

1101：SYSCLK 128 分频

1110：SYSCLK 256 分频

1111：SYSCLK 512 分频

Bit 3:2 **SWS[1:0]**：系统时钟切换状态

由硬件设置和清除，以指示哪个时钟源被用作系统时钟。

00：HSI 用作系统时钟

01：HSE 用作系统时钟

10：PLL 用作系统时钟

11：LSI 用作系统时钟

Bit 1:0 **SW[1:0]**：系统时钟切换

由软件设置和清除，以选择系统时钟源（SYSCLK）。

在退出停止和待机模式或 HSE 振荡器故障时，由硬件配置以强制选择 HSI 振荡器。

00：选择 HSI 作为系统时钟

01：选择 HSE 作为系统时钟

10：选择 PLL 作为系统时钟

11：选择 LSI 作为系统时钟

7.4.3 时钟中断寄存器（RCC_CIR）

偏移地址：0x08
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CSSC	Reserved		PLL RDYC	Res	HSI RDYC	Res	LSI RDYC
								w			w		w		w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			PLL RDYIE	Res	HSI RDYIE	Res	LSI RDYIE	CSSF	Reserved		PLL RDYF	Res	HSI RDYF	Res	LSI RDYF
			rw		rw		rw	r			r		r		r

Bit 31:24 保留，必须保持为复位值。

Bit 23 **CSSC**：时钟安全系统中断清除
此位由软件设置以清除 CSSF 标志。
0：无影响
1：清除 CSSF 标志

Bit 22:21 保留，必须保持为复位值。

Bit 20 **PLLRDYC**：PLL 就绪中断清除
此位由软件设置以清除 PLLRDYF 标志。
0：无影响
1：清除 PLLRDYF 标志

Bit 19 保留，必须保持为复位值。

Bit 18 **HSIRDYC**: HSI 就绪中断清除
此位由软件设置以清除 HSIRDYF 标志。
0：无影响
1：清除 HSIRDYF 标志

Bit 17 保留，必须保持为复位值。

Bit 16 **LSIRDYC**：LSI 就绪中断清除
此位由软件设置以清除 LSIRDYF 标志。
0：无影响
1：清除 LSIRDYF 标志

Bit 15:13 保留，必须保持为复位值。

Bit 12 **PLLRDYIE**：PLL 就绪中断使能
由软件设置和清除，以使能/禁用由 PLL 锁定引起的中断。
0：PLL 锁定中断禁用
1：PLL 锁定中断使能

Bit 11 保留，必须保持为复位值。

Bit 10 **HSIRDYIE**：HSI 就绪中断使能

由软件设置和清除，以使能/禁用由 HSI 振荡器稳定引起的中断。

0：HSI 就绪中断禁用

1：HSI 就绪中断使能

Bit 9 保留，必须保持为复位值。

Bit 8 **LSIRDYIE**：LSI 就绪中断使能

由软件设置和清除，以使能/禁用由 LSI 振荡器稳定引起的中断。

0：LSI 就绪中断禁用

1：LSI 就绪中断使能

Bit 7 **CSSF**：时钟安全系统中断标志

当检测到 HSE 振荡器故障时，由硬件设置。

通过软件设置 CSSC 位来清除。

0：无 HSE 时钟故障引起的时钟安全中断

1：HSE 时钟故障引起的时钟安全中断

Bit 6:5 保留，必须保持为复位值。

Bit 4 **PLLRDYF**：PLL 就绪中断标志

当 PLL 锁定且 PLLRDYDIE 被设置时，由硬件设置。

通过软件设置 PLLRDYC 位来清除。

0：无 PLL 锁定引起的时钟就绪中断

1：PLL 锁定引起的时钟就绪中断

Bit 3 保留，必须保持为复位值。

Bit 2 **HSIRDYF**：HSI 就绪中断标志

当 HSI 时钟稳定且 HSIRDYDIE 被设置时（响应 HSION 的设置，参见时钟控制寄存器（RCC_CR)），由硬件设置。若 HSION 未设置但 HSI 振荡器通过外设的时钟请求被使能，则此位不被设置且不产生中断。

通过软件设置 HSIRDYC 位来清除。

0：无 HSI 振荡器引起的时钟就绪中断

1：HSI 振荡器引起的时钟就绪中断

Bit 1 保留，必须保持为复位值。

Bit 0 **LSIRDYF**：LSI 就绪中断标志

当 LSI 时钟稳定且 LSIRDYDIE 被设置时，由硬件设置。

通过软件设置 LSIRDYC 位来清除。

0：无 LSI 振荡器引起的时钟就绪中断

1：LSI 振荡器引起的时钟就绪中断

7.4.4 APB2 外设复位寄存器（RCC_APB2RSTR）

偏移地址：0x0C
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											DAC2RST	DAC1RST	Reserved		TIM15RST
											rw	rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	UART1RST	TIM8RST	SPI1RST	TIM1RST	Reserved										SYSCFGRST
	rw	rw	rw	rw											rw

Bit 31:21 保留，必须保持为复位值。

Bit 20 **DAC2RST**：DAC2 复位
由软件设置和清除。
0：无影响
1：复位 DAC2

Bit 19 **DAC1RST**：DAC1 复位
由软件设置和清除。
0：无影响
1：复位 DAC1

Bit 18:17 保留，必须保持为复位值。

Bit 16 **TIM15RST**：TIM15 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM15 定时器

Bit 15 保留，必须保持为复位值。

Bit 14 **UART1RST**：UART1 复位
由软件设置和清除。
0：无影响
1：复位 UART1

Bit 13 **TIM8RST**：TIM8 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM8 定时器

Bit 12 **SPI1RST**：SPI1 复位
由软件设置和清除。
0：无影响
1：复位 SPI1

Bit 11 **TIM1RST**：TIM1 定时器复位

由软件设置和清除。

0：无影响

1：复位 TIM1 定时器

Bit 10:1 保留，必须保持为复位值。

Bit 0 **SYSCFGRST**：SYSCFG + COMP + OPAMP + VREFBUF 复位

0：无影响

1：复位 SYSCFG、COMP、OPAMP 和 VREFBUF

7.4.5 APB1 外设复位寄存器（RCC_APB1RSTR）

偏移地址：0x10

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PWR RST	Reserved		FDCAN RST	Reserved			I2C1 RST	Reserved		UART3 RST	UART2 RST	Res
			rw			rw				rw			rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TIMB RST	TIMA RST	TIM7 RST	TIM6 RST	Reserved		TIM3 RST	TIM2 RST
								rw	rw	rw	rw			rw	rw

Bit 31:29 保留，必须保持为复位值。

Bit 28 **PWRRST**：电源接口复位

由软件设置和清除。

0：无影响

1：复位 PWR

Bit 27:26 保留，必须保持为复位值。

Bit 25 **FDCANRST**：FDCAN 复位

由软件设置和清除。

0：无影响

1：复位 FDCAN

Bit 24:22 保留，必须保持为复位值。

Bit 21 **I2C1RST**：I2C1 复位

由软件设置和清除。

0：无影响

1：复位 I2C1

Bit 20:19 保留，必须保持为复位值。

Bit 18 **UART3RST**：UART3 复位
由软件设置和清除。
0：无影响
1：复位 UART3

Bit 17 **UART2RST**：UART2 复位
由软件设置和清除。
0：无影响
1：复位 UART2

Bit 16:8 保留，必须保持为复位值。

Bit 7 **TIMBRST**：TIMB 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIMB

Bit 6 **TIMARST**：TIMA 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIMA

Bit 5 **TIM7RST**：TIM7 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM7

Bit 4 **TIM6RST**：TIM6 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM6

Bit 3:2 保留，必须保持为复位值。

Bit 1 **TIM3RST**：TIM3 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM3

Bit 0 **TIM2RST**：TIM2 定时器复位
由软件设置和清除。
0：无影响
1：复位 TIM2

7.4.6 AHB 外设时钟使能寄存器（RCC_AHBENR）

偏移地址：0x14

复位值：0x0000 0014

访问：无等待，支持字，半字和字节访问

注：当外设时钟未激活时，不支持对外设寄存器的读或写访问。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					ADC2EN	Res	ADC1EN	Reserved				IOPDEN	IOPCEN	IOPBEN	IOPAEN
					rw		rw					rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				MEEN	Reserved				CRCEN	Res	FLITFEN	CORDICEN	SRAMEN	Res	DMA1EN
				rw					rw		rw	rw	rw		rw

Bit 31:27 保留，必须保持为复位值。

Bit 26 **ADC2EN**：ADC2 时钟使能

由软件设置和清除。

0：ADC2 时钟禁用

1：ADC2 时钟使能

Bit 25 保留，必须保持为复位值。

Bit 24 **ADC1EN**：ADC1 时钟使能

由软件设置和清除。

0：ADC1 时钟禁用

1：ADC1 时钟使能

Bit 23:20 保留，必须保持为复位值。

Bit 19 **GPIODEN**：GPIO 端口 D 时钟使能

由软件设置和清除。

0：IO 端口 D 时钟禁用

1：IO 端口 D 时钟使能

Bit 18 **GPIOCEN**：GPIO 端口 C 时钟使能

由软件设置和清除。

0：IO 端口 C 时钟禁用

1：IO 端口 C 时钟使能

Bit 17 **GPIOBEN**：GPIO 端口 B 时钟使能

由软件设置和清除。

0：IO 端口 B 时钟禁用

1：IO 端口 B 时钟使能

Bit 16 **GPIOAEN**：GPIO 端口 A 时钟使能

由软件设置和清除。

0：IO 端口 A 时钟禁用

1：IO 端口 A 时钟使能

Bit 15:12 保留，必须保持为复位值。

Bit 11 **MEEN**：ME 时钟使能
由软件设置和清除。
0：ME 时钟禁用
1：ME 时钟使能

Bit 10:7 保留，必须保持为复位值。

Bit 6 **CRCEN**：CRC 时钟使能
由软件设置和清除。
0：CRC 时钟禁用
1：CRC 时钟使能

Bit 5 保留，必须保持为复位值。

Bit 4 **FLITFEN**：FLITF 时钟使能
由软件设置和清除。
0：睡眠模式时闪存接口电路时钟禁用
1：睡眠模式时闪存接口电路时钟使能

Bit 3 **CORDICEN**：CORDIC 时钟使能
由软件设置和清除。
0：CORDIC 时钟禁用
1：CORDIC 时钟使能

Bit 2 **SRAMEN**：SRAM 时钟使能
由软件设置和清除。
0：睡眠模式时 SRAM 时钟禁用
1：睡眠模式时 SRAM 时钟使能

Bit 1 保留，必须保持为复位值。

Bit 0 **DMA1EN**：DMA1 时钟使能
由软件设置和清除。
0：DMA1 时钟禁用
1：DMA1 时钟使能

7.4.7 APB2 外设使能寄存器（RCC_APB2ENR）

偏移地址：0x18
复位值：0x0000 0000
访问：支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											DAC2EN	DAC1EN	Reserved		TIM15EN
											rw	rw			rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	UART1EN	TIM8EN	SPI1EN	TIM1EN	Reserved										SYSCFGEN
	rw	rw	rw	rw											rw

Bit 31:21 保留，必须保持为复位值。

Bit 20 **DAC2EN**：DAC2 时钟使能
由软件设置和清除。
0：DAC2 时钟禁用
1：DAC2 时钟使能

Bit 19 **DAC1EN**：DAC1 时钟使能
由软件设置和清除。
0：DAC1 时钟禁用
1：DAC1 时钟使能

Bit 18:17 保留，必须保持为复位值。

Bit 16 **TIM15EN**：TIM15 定时器时钟使能
由软件设置和清除。
0：TIM15 定时器时钟禁用
1：TIM15 定时器时钟使能

Bit 15 保留，必须保持为复位值。

Bit 14 **UART1EN**：UART1 时钟使能
由软件设置和清除。
0：UART1 时钟禁用
1：UART1 时钟使能

Bit 13 **TIM8EN**：TIM8 定时器时钟使能
由软件设置和清除。
0：TIM8 定时器时钟禁用
1：TIM8 定时器时钟使能

Bit 12 **SPI1EN**：SPI1 时钟使能
由软件设置和清除。
0：SPI1 时钟禁用
1：SPI1 时钟使能

Bit 11 **TIM1EN**：TIM1 定时器时钟使能

由软件设置和清除。

0：TIM1 定时器时钟禁用

1：TIM1 定时器时钟使能

Bit 10:1 保留，必须保持为复位值。

Bit 0 **SYSCFGEN**：SYSCFG + COMP + OPAMP + VREFBUF 时钟使能

0：SYSCFG + COMP + OPAMP + VREFBUF 时钟禁用

1：SYSCFG + COMP + OPAMP + VREFBUF 时钟使能

7.4.8 APB1 外设使能寄存器（RCC_APB1ENR）

偏移地址：0x1C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			PWR EN	Reserved		FDCAN EN	Reserved			I2C1 EN	Reserved		UART3 EN	UART2 EN	Res
			rw			rw				rw			rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TIMB EN	TIMA EN	TIM7 EN	TIM6 EN	Reserved		TIM3 EN	TIM2 EN
								rw	rw	rw	rw			rw	rw

Bit 31:29 保留，必须保持为复位值。

Bit 28 **PWREN**：电源接口时钟使能

由软件设置和清除。

0：电源接口时钟禁用

1：电源接口时钟使能

Bit 27:26 保留，必须保持为复位值。

Bit 25 **FDCANEN**：FDCAN 时钟使能

由软件设置和清除。

0：FDCAN 时钟禁用

1：FDCAN 时钟使能

Bit 24:22 保留，必须保持为复位值。

Bit 21 **I2C1EN**：I2C1 时钟使能

由软件设置和清除。

0：I2C1 时钟禁用

1：I2C1 时钟使能

Bit 20:19 保留，必须保持为复位值。

Bit 18 **UART3EN**：UART3 时钟使能
由软件设置和清除。
0：UART3 时钟禁用
1：UART3 时钟使能
Bit 17 **UART2EN**：UART2 时钟使能
由软件设置和清除。
0：UART2 时钟禁用
1：UART2 时钟使能

Bit 16:8 保留，必须保持为复位值。

Bit 7 **TIMBEN**：TIMB 定时器时钟使能
由软件设置和清除。
0：TIMB 定时器时钟禁用
1：TIMB 定时器时钟使能
Bit 6 **TIMAEN**：TIMA 定时器时钟使能
由软件设置和清除。
0：TIMA 定时器时钟禁用
1：TIMA 定时器时钟使能
Bit 5 **TIM7EN**：TIM7 定时器时钟使能
由软件设置和清除。
0：TIM7 定时器时钟禁用
1：TIM7 定时器时钟使能
Bit 4 **TIM6EN**：TIM6 定时器时钟使能
由软件设置和清除。
0：TIM6 定时器时钟禁用
1：TIM6 定时器时钟使能

Bit 3:2 保留，必须保持为复位值。

Bit 1 **TIM3EN**：TIM3 定时器时钟使能
由软件设置和清除。
0：TIM3 定时器时钟禁用
1：TIM3 定时器时钟使能
Bit 0 **TIM2EN**：TIM2 定时器时钟使能
由软件设置和清除。
0：TIM2 定时器时钟禁用
1：TIM2 定时器时钟使能

7.4.9 RTC 域控制寄存器（RCC_BDCR）

偏移地址：0x20
复位值：0x0000 0000
访问：0 ≤ 等待状态 ≤ 3，支持字、半字和字节访问。
对该寄存器的连续访问会插入等待状态。

注：此寄存器的位不在V_{CORE}*域内。因此，复位后，这些位受写保护，必须先设置*PWR *电源控制寄存器*1*（*PWR_CR1*）中的*DBP *位，才能对这些位进行修改。这些位（*BDRST *除外）仅在*RTC *域复位后复位（见章节：*RTC *域复位）。任何内部或外部复位对这些位均无效。*

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

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

Bit 31:17 保留，必须保持为复位值。

Bit 16 **BDRST**：RTC 域软件复位
由软件设置和清除。
0：未激活复位
1：复位整个 RTC 域

Bit 15 **RTCEN**：RTC 时钟使能
由软件设置和清除。
0：禁用 RTC 时钟
1：使能 RTC 时钟

Bit 14:10 保留，必须保持为复位值。

Bit 9:8 **RTCSEL[1:0]**：RTC 时钟源选择
由软件设置，以选择 RTC 的时钟源。一旦选择了 RTC 时钟源，除非 RTC 域被复位，否则无法更改。可以使用 BDRST 位来复位它们。
00：无时钟
01：无时钟
10：使用 LSI 振荡器时钟作为 RTC 时钟
11：使用分频 128 后的 HSE 振荡器时钟作为 RTC 时钟

Bit 7:0 保留，必须保持为复位值。

7.4.10 控制/状态寄存器（RCC_CSR）

偏移地址：0x24
复位值：0x0C00 0000
由系统复位重置，但仅由电源复位重置复位标志。
访问：0 ≤ 等待状态 ≤ 3，支持字、半字和字节访问。
对该寄存器的连续访问会插入等待状态。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWR RSTF	Res	IWDG RSTF	SFT RSTF	POR RSTF	PIN RSTF	OBL RSTF	RMVF	Reserved							
r		r	r	r	r	r	rw								

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

Bit 31 **LPWRRSTF**：低功耗复位标志
当因非法进入停止和待机模式而发生复位时，由硬件设置。
通过写入 RMVF 位清除。
0：未发生非法模式复位
1：发生非法模式复位

Bit 30 保留，必须保持为复位值。

Bit 29 **IWDGRSTF**：独立窗口看门狗复位标志
当发生独立看门狗复位域时，由硬件设置。
通过写入 RMVF 位清除。
0：未发生独立看门狗复位
1：发生独立看门狗复位

Bit 28 **SFTRSTF**：软件复位标志
当发生软件复位时，由硬件设置。
通过写入 RMVF 位清除。
0：未发生软件复位
1：发生软件复位

Bit 27 **PORRSTF**：上电/掉电复位标志
当发生上电/掉电复位时，由硬件设置。
通过写入 RMVF 位清除。
0：未发生上电/掉电复位
1：发生上电/掉电复位

Bit 26 **PINRSTF**：引脚复位标志

当 NRST 引脚发生复位时，由硬件设置。

通过写入 RMVF 位清除。

0：NRST 引脚未发生复位

1：NRST 引脚发生复位

Bit 25 **OBLRSTF**：选项字节加载复位标志

当因选项字节加载而发生复位时，由硬件设置。

通过写入 RMVF 位清除。

0：未因选项字节加载而发生复位

1：因选项字节加载而发生复位

Bit 24 **RMVF**：移除复位标志

通过软件设置以清除复位标志。

0：无影响

1：清除复位标志

Bit 23:2 保留，必须保持为复位值。

Bit 1 **LSIRDY**：LSI 振荡器就绪

由硬件设置和清除，以指示 LSI 振荡器是否稳定。在 LSION 位被清除后，LSIRDY 将在 3 个 LSI 振荡器时钟周期后变为低电平。即使 LSION = 0，如果 LSI 被独立看门狗或 RTC 请求，此位也可以被设置。

0：LSI 振荡器未就绪

1：LSI 振荡器就绪

Bit 0 **LSION**：LSI 振荡器使能

由软件设置和清除。

0：LSI 振荡器关闭

1：LSI 振荡器开启

7.4.11 AHB 外设时钟复位寄存器（RCC_AHBRSTR）

偏移地址：0x28
复位值：0x0000 0014
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					ADC2RST	Res	ADC1RST	Reserved				IOPDRST	IOPCRST	IOPBRST	IOPARST
					rw		rw					rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				MERST	Reserved				CRCRST	Reserved		CORDICRST	Reserved		DMA1RST
				rw					rw			rw			rw

Bit 31:27 保留，必须保持为复位值。

Bit 26 **ADC2RST**：ADC2 复位
由软件设置和清除。
0：无影响
1：复位 ADC2

Bit 25 保留，必须保持为复位值。

Bit 24 **ADC1RST**：ADC1 复位
由软件设置和清除。
0：无影响
1：复位 ADC1

Bit 23:20 保留，必须保持为复位值。

Bit 19 **GPIODRST**：GPIO 端口 D 复位
由软件设置和清除。
0：无影响
1：复位 IO 端口 D

Bit 18 **GPIOCRST**：GPIO 端口 C 复位
由软件设置和清除。
0：无影响
1：复位 IO 端口 C

Bit 17 **GPIOBRST**：GPIO 端口 B 复位
由软件设置和清除。
0：无影响
1：复位 IO 端口 B

Bit 16 **GPIOARST**：GPIO 端口 A 复位
由软件设置和清除。
0：无影响
1：复位 IO 端口 A

Bit 15:12 保留，必须保持为复位值。

Bit 11 **MERST**：ME 复位
由软件设置和清除。

0：无影响
1：复位 ME

Bit 10:7 保留，必须保持为复位值。

Bit 6 **CRCRST**：CRC 复位
由软件设置和清除。
0：无影响
1：复位 CRC

Bit 5:4 保留，必须保持为复位值。

Bit 3 **CORDICRST**：CORDIC 复位
由软件设置和清除。
0：无影响
1：复位 CORDIC

Bit 2:1 保留，必须保持为复位值。

Bit 0 **DMA1RST**：DMA1 时钟使能
由软件设置和清除。
0：无影响
1：复位 DMA1

7.4.12 内部时钟源校准寄存器（RCC_ICSCR）

偏移地址：0xAC
复位值：0x0000 00C0
访问：无等待，支持字，半字和字节访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							HSITRIM[8:0]								
							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
Reserved							HSICAL[8:0]								
							r	r	r	r	r	r	r	r	r

Bit 31:25 保留，必须保持为复位值。

Bit 24:16 **HSITRIM[8:0]**：HSI 时钟调整
上电时载入工厂生产预设值，软件写入来调整内部高速时钟。

Bit 15:9 保留，必须保持为复位值。

Bit 8:0 **HSICAL[8:0]**：HSI 时钟校准
这些位在启动时会初始化出厂编程设定的 HSI 校准微调值。

8 通用 I/O（GPIO）

8.1 简介

每个通用 I/O 端口包括 4 个 32 位配置寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR 和 GPIOx_PUPDR）、2 个 32 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR）、1 个 32 位置位/复位寄存器（GPIOx_BSRR）和 1 个 16 位复位寄存器（GPIOx_BRR）。此外，所有 GPIO 都包括 1 个 32 位锁定寄存器（GPIOx_LCKR）和 2 个 32 位复用功能选择寄存器（GPIOx_AFRH 和 GPIOx_AFRL）。

8.2 GPIO 主要特征

- 输出状态：推挽或开漏+上拉/下拉
- 从输出数据寄存器（GPIOx_ODR）或外设（复用功能输出）输出数据
- 可为每个 I/O 选择不同的速度
- 输入状态：浮空、上拉/下拉、模拟
- 将数据输入到输入数据寄存器（GPIOx_IDR）或外设（复用功能输入）
- 置位和复位寄存器（GPIOx_BSRR），对 GPIOx_ODR 具有按位写权限
- 锁定机制（GPIOx_LCKR），可冻结 I/O 端口配置
- 模拟功能
- 复用功能选择寄存器
- 快速翻转，每次翻转最快只需要两个时钟周期
- 引脚复用非常灵活，允许将 I/O 引脚用作 GPIO 或多种外设功能中的一种

8.3 GPIO 功能描述

根据数据手册中列出的每个 I/O 端口的特性，可通过软件将通用 I/O（GPIO）端口的各个端口位分别配置为多种模式：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟
- 具有上拉或下拉功能的开漏输出
- 具有上拉或下拉功能的推挽输出
- 具有上拉或下拉功能的复用功能推挽
- 具有上拉或下拉功能的复用功能开漏

每个 I/O 端口位均可自由编程，但 I/O 端口寄存器必须按 32 位字、半字或字节进行访问。GPIOx_BSRR 寄存器和 GPIOx_BRR 寄存器旨在实现对 GPIOx_ODR 寄存器进行原子读取/修改访问。这样便可确保在读取和修改访问之间发生中断请求也不会有问题。

下两张图分别显示了标准和 5V 容忍 I/O 端口位的基本结构。下表给出了可能的端口位配置方案。

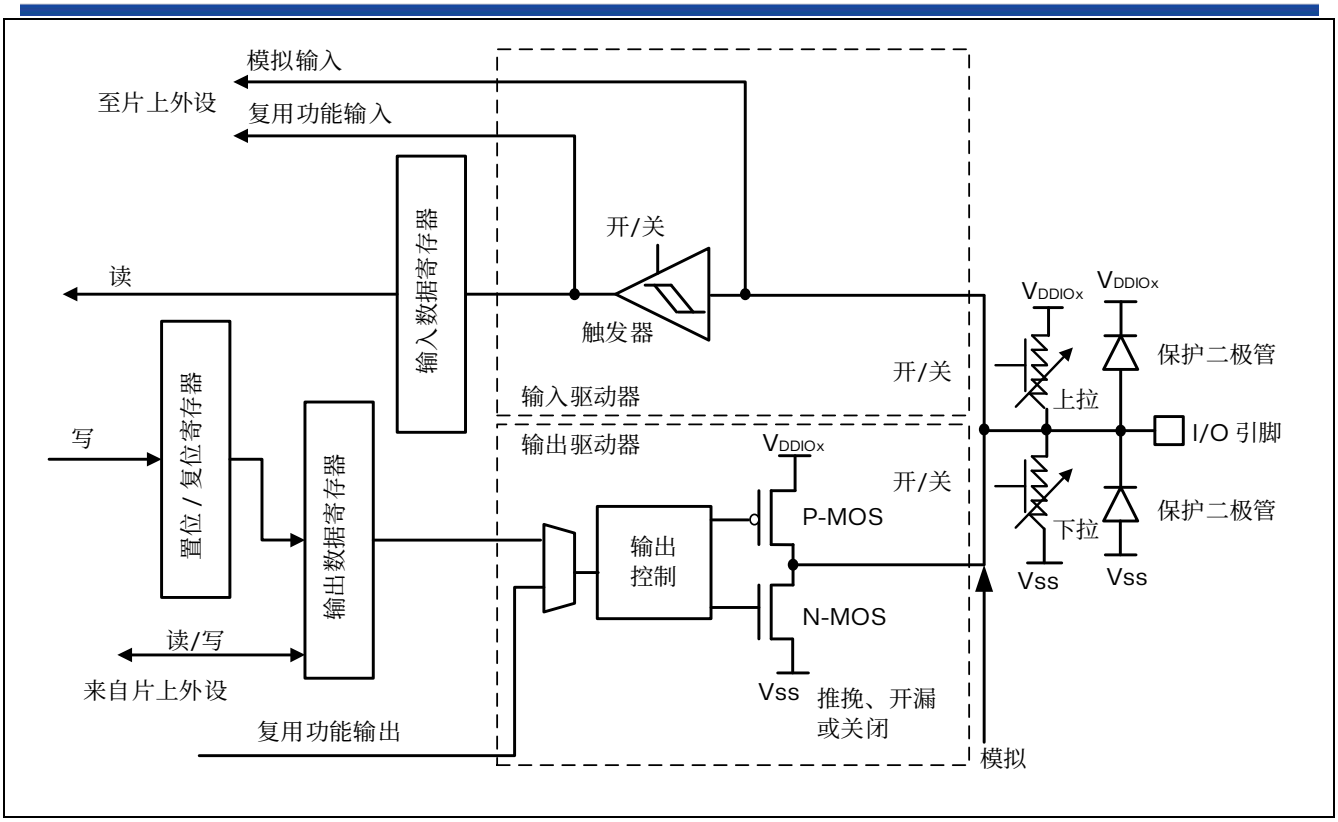
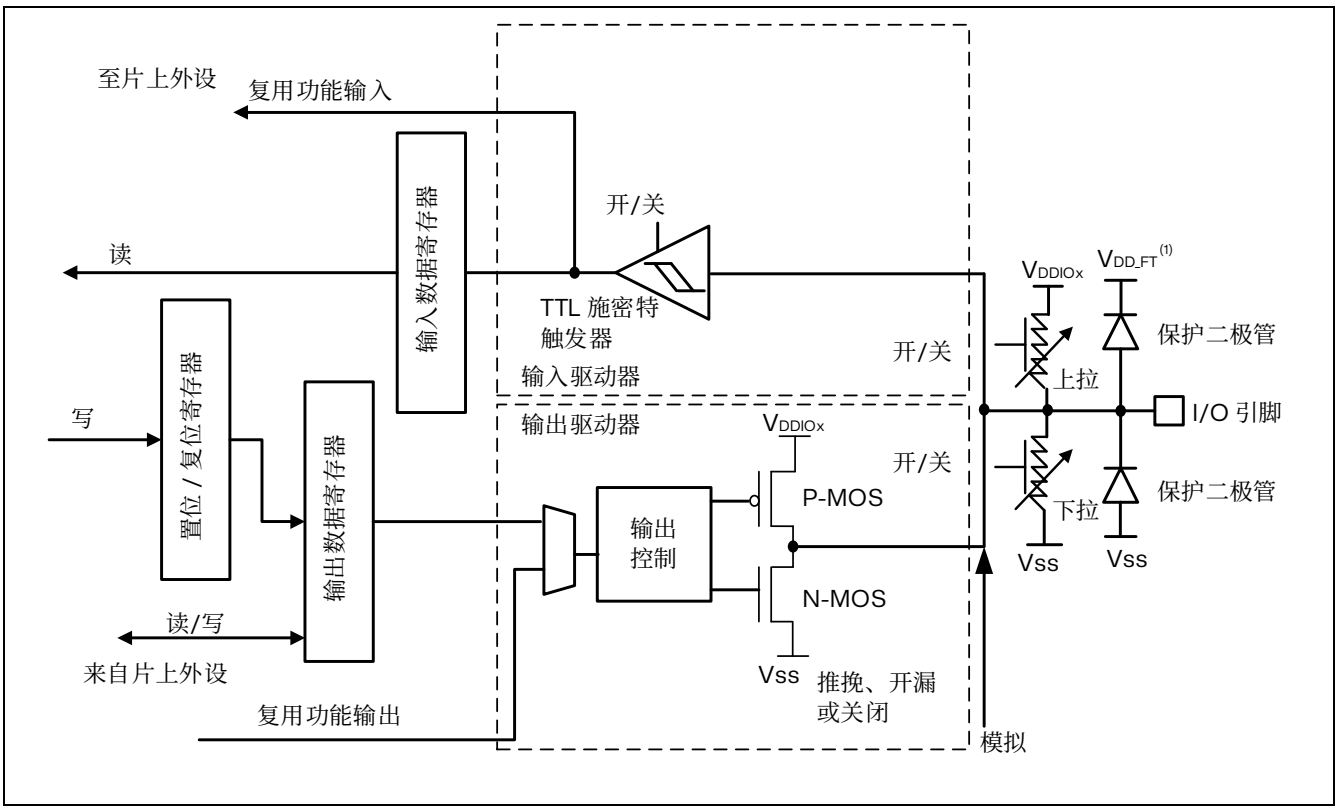


图 8.1 I/O 端口位基本结构



1. VDD_FT 是和 5V 容忍 I/O 相关的电位，与 VDD 不同。

图 8.2 5V 容忍 I/O 端口位的基本结构

表 8.1 端口位配置表 (1)

MODE[1:0]	OTYPE	OSPEED[1:0]		PUPD[1:0]		I/O 配置	
01	0	SPEED[1:0]		0	0	GP 输出	PP
	0			0	1	GP 输出	PP + PU
	0			1	0	GP 输出	PP + PD
	0			1	1	保留	
	1			0	0	GP 输出	OD
	1			0	1	GP 输出	OD + PU
	1			1	0	GP 输出	OD + PD
	1			1	1	保留（GP 输出 OD）	
10	0	SPEED[1:0]		0	0	AF	PP
	0			0	1	AF	PP + PU
	0			1	0	AF	PP + PD
	0			1	1	保留	
	1			0	0	AF	OD
	1			0	1	AF	OD + PU
	1			1	0	AF	OD + PD
	1			1	1	保留	
00	x	x	x	0	0	输入	浮空
	x	x	x	0	1	输入	PU
	x	x	x	1	0	输入	PD
	x	x	x	1	1	保留（输入浮空）	
11	x	x	x	0	0	输入/输出	模拟
	x	x	x	0	1	保留	
	x	x	x	1	0		
	x	x	x	1	1		

1. GP =通用、PP =推挽、PU =上拉、PD =下拉、OD =开漏、AF =复用功能。

8.3.1 通用 IO（GPIO）

在复位期间及复位刚刚完成后，复用功能尚未激活，大多数 I/O 端口被配置为模拟模式。

复位后，调试引脚处于复用功能上拉/下拉状态：

- PA1：DBGCLK 处于下拉状态
- PA0：DBGIO 处于上拉状态

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

输入数据寄存器（GPIOx_IDR）每个 AHB 时钟周期捕获一次 I/O 引脚的数据。

所有 GPIO 引脚都具有内部弱上拉及下拉电阻，可根据 GPIOx_PUPDR 寄存器中的值来打开/关闭。

8.3.2 I/O 引脚复用功能复用器和映射

器件 I/O 引脚通过一个复用器连接到板载外设/模块，该复用器一次仅允许一个外设的复用功能（AF）连接到 I/O 引脚。这可以确保共用同一个 I/O 引脚的外设之间不会发生冲突。

每个 I/O 引脚都有一个复用器，该复用器采用多达 8 路复用功能输入（AF0 到 AF7），可通过 GPIOx_AFRL（针对引脚 0 到 7）和 GPIOx_AFRH（针对引脚 8 到 15）寄存器对这些输入进行配置：

- 复位后，复用器选择为复用功能 0（AF0）。在复用模式下通过 GPIOx_MODER 寄存器配置 I/O。
- 器件数据手册中详细说明了每个引脚的特定复用功能分配。

除了这种灵活的 I/O 复用架构之外，各外设还可以将复用功能映射到不同 I/O 引脚，这可以优化小型封装中可用外设的数量。

要在指定配置下使用 I/O，用户必须按照以下步骤操作：

- **调试功能：**每个器件复位后，立即将这些引脚分配为可由调试主机使用的复用功能引脚。
- **GPIO：**在 GPIOx_MODER 寄存器中将所需 I/O 配置为输出、输入或模拟通道。
- **外设复用功能：**
  - 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，将 I/O 连接到所需的 AFx。
  - 通过 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDER 寄存器，分别选择类型、上拉/下拉以及输出速度。
  - 在 GPIOx_MODER 寄存器中将所需 I/O 配置为复用功能。
- **模拟功能：**
  - 对于 ADC、OPAMP 和 COMP，在 GPIOx_MODER 寄存器中将所需 I/O 配置为模拟模式，并在 ADC、OPAMP 和 COMP 寄存器中配置所需功能。

    如上所述，对于附加功能（如 OPAMP），输出由相应的外设控制。在使能外设控制寄存器中的附加功能输出之前，必须谨慎选择 I/O 端口的模拟功能。
  - 对于 RTC、WKUPx 和振荡器的其它功能，在相关的 RTC、PWR 和 RCC 寄存器中配置所需功能。这些功能优先于标准 GPIO 寄存器中的配置。

有关复用功能 I/O 引脚映射的详细信息，请参见器件数据手册中的“复用功能映射”表。

8.3.3 I/O 端口控制寄存器

每个 GPIO 端口有 4 个 32 位存储器映射的控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR），可配置多达 16 个 I/O。GPIOx_MODER 寄存器用于选择 I/O 模式（输入、输出、复用、模拟）。GPIOx_OTYPER 和 GPIOx_OSPEEDR 寄存器用于选择输出类型（推挽或开漏）和速度。无论采用哪种 I/O 方向，GPIOx_PUPDR 寄存器都用于选择上拉/下拉。

8.3.4 I/O 端口数据寄存器

每个 GPIO 都具有 2 个 16 位数据寄存器：输入和输出数据寄存器（GPIOx_IDR 和 GPIOx_ODR）。GPIOx_ODR 用于存储待输出数据，可对其进行读/写访问。通过 I/O 输入的数据存储到输入数据寄存器（GPIOx_IDR）中，它是一个只读寄存器。

有关寄存器说明的详细信息，请参见章节：GPIO 端口输入数据寄存器（GPIOx_IDR）（x = A...D）和 GPIO 端口输出数据寄存器（GPIOx_ODR）（x = A...D）。

8.3.5 I/O 数据位操作

置位复位寄存器（GPIOx_BSRR）是一个 32 位寄存器，它允许应用程序在输出数据寄存器（GPIOx_ODR）中对各个单独的数据位执行置位和复位操作。置位复位寄存器的大小是 GPIOx_ODR 的二倍。

GPIOx_ODR 中的每个数据位对应于 GPIOx_BSRR 中的两个控制位：BS(i)和 BR(i)。当写入 1 时，BS(i)位会**置位**对应的 ODR(i)位。当写入 1 时，BR(i)位会**复位** ODR(i)对应的位。

在 GPIOx_BSRR 中向任何位写入 0 都不会对 GPIOx_ODR 中的对应位产生任何影响。如果在 GPIOx_BSRR 中同时尝试对某个位执行置位和复位操作，则置位操作优先。

使用 GPIOx_BSRR 寄存器更改 GPIOx_ODR 中各个位的值是一个“单次”操作，不会锁定 GPIOx_ODR 位。随时都可以直接访问 GPIOx_ODR 位。GPIOx_BSRR 寄存器提供了一种执行原子按位处理的方法。

在对 GPIOx_ODR 进行位操作时，软件无需禁止中断：在一次原子 AHB 写访问中，可以修改一个或多个位。

8.3.6 I/O 锁定机制

通过将特定的写序列应用到 GPIOx_LCKR 寄存器，可以冻结 GPIO 控制寄存器。冻结的寄存器包括 GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFRL 和 GPIOx_AFRH。

要对 GPIOx_LCKR 寄存器执行写操作，必须应用特定的写/读序列。当正确的 LOCK 序列应用到此寄存器的第 16 位后，会使用 LCKR[15:0]的值来锁定 I/O 的配置（在写序列期间，LCKR[15:0]的值必须相同）。将 LOCK 序列应用到某个端口位后，在执行下一次 MCU 复位或外设复位之前，将无法对该端口位的值进行修改。每个 GPIOx_LCKR 位都会冻结控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR、GPIOx_AFRL 和 GPIOx_AFRH）中的对应位。

LOCK 序列（参见章节：GPIO 端口配置锁定寄存器（GPIOx_LCKR）（x = A...D）只能通过对 GPIOx_LCKR 寄存器进行字（32 位长）访问的方式来执行，因为 GPIOx_LCKR 的第 16 位必须与[15:0]位同时置位。有关详细信息，请参见章节：GPIO 端口配置锁定寄存器（GPIOx_LCKR）（x = A...D）中的 LCKR 寄存器说明。

8.3.7 I/O 复用功能输入/输出

有两个寄存器可用来从每个 I/O 可用的复用功能输入/输出中进行选择。借助这些寄存器，用户可根据应用程序的要求将某个复用功能连接到其它某个引脚。

这意味着可使用 GPIOx_AFRL 和 GPIOx_AFRH 复用功能寄存器在每个 GPIO 上复用多个可用的外设功能。这样一来，应用程序可为每个 I/O 选择任何一个可用功能。由于 AF 选择信号由复用功能输入和复用功能输出共用，所以只需为指定 I/O 的复用功能输入/输出选择一个通道即可。

要了解在每个 GPIO 引脚上复用了哪些功能，请参见器件数据手册。

8.3.8 外部中断/唤醒线

所有端口都具有外部中断功能。要使用外部中断线，必须将端口配置为输入模式。请参见章节：外部中断和事件控制器（EXTI）。

8.3.9 输入配置

当 I/O 端口被配置为输入时：

- 输出缓冲器被禁止
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 输入数据寄存器每个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样
- 对输入数据寄存器的读访问可获取 I/O 状态

下图说明了 I/O 端口位的输入配置。

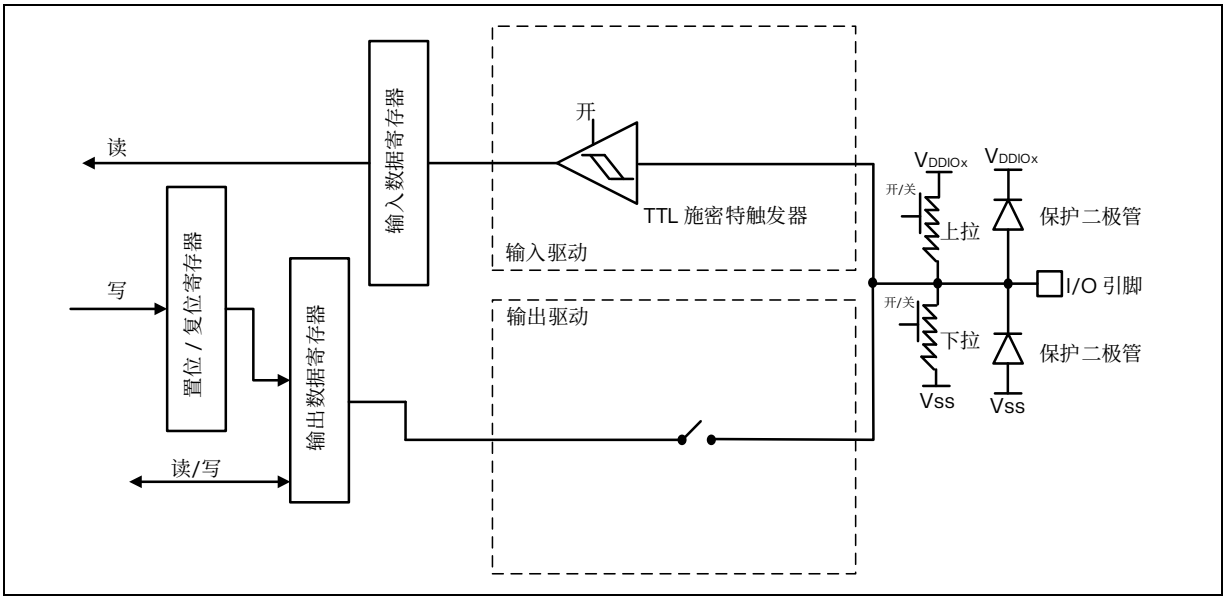


图 8.3 输入浮空/上拉/下拉配置

8.3.10 输出模式配置

当 I/O 端口被配置为输出时：

- 输出缓冲器被打开：
  - 开漏模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”会使端口保持高阻态（Hi-Z）（P-MOS 始终不激活）
  - 推挽模式：输出寄存器中的“0”可激活 N-MOS，而输出寄存器中的“1”可激活 P-MOS
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开上拉和下拉电阻
- 对输入数据寄存器的读访问可获取 I/O 状态
- 对输出数据寄存器的读访问可获取最后的写入值

下图说明了 I/O 端口位的输出配置。

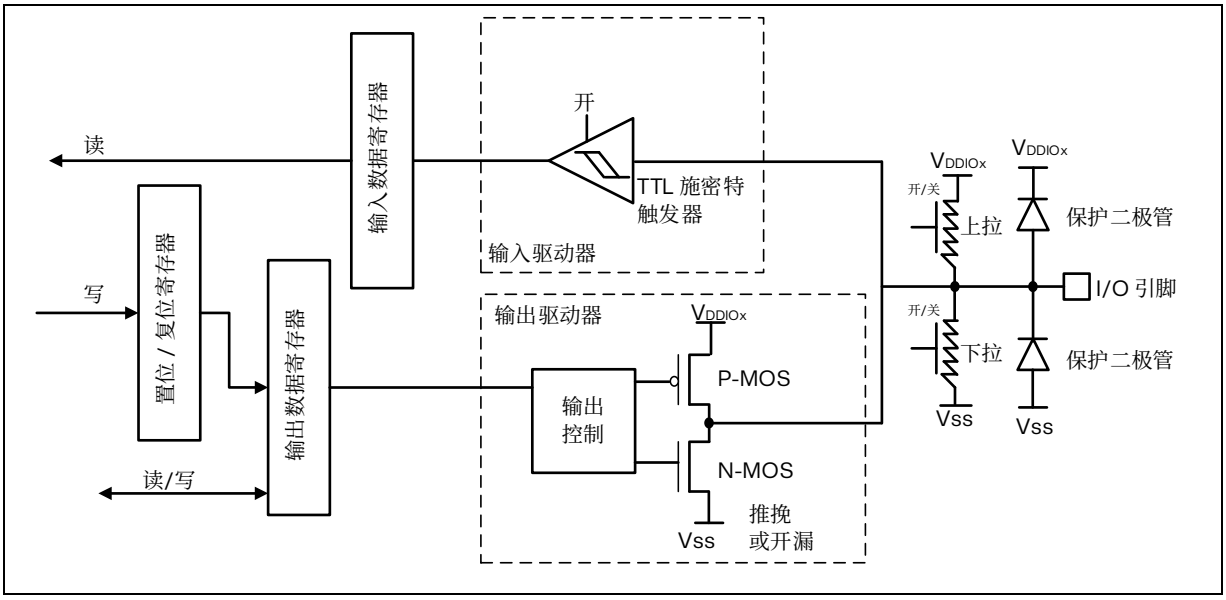


图 8.4 输出配置

8.3.11 复用功能配置

当 I/O 端口被配置为复用功能时：

- 可将输出缓冲器配置为开漏或推挽模式
- 输出缓冲器由来自外设的信号驱动（发送器使能和数据）
- 施密特触发器输入被打开
- 根据 GPIOx_PUPDR 寄存器中的值决定是否打开弱上拉电阻和下拉电阻
- 输入数据寄存器每个 AHB 时钟周期对 I/O 引脚上的数据进行一次采样
- 对输入数据寄存器的读访问可获取 I/O 状态

下图说明了 I/O 端口位的复用功能配置。

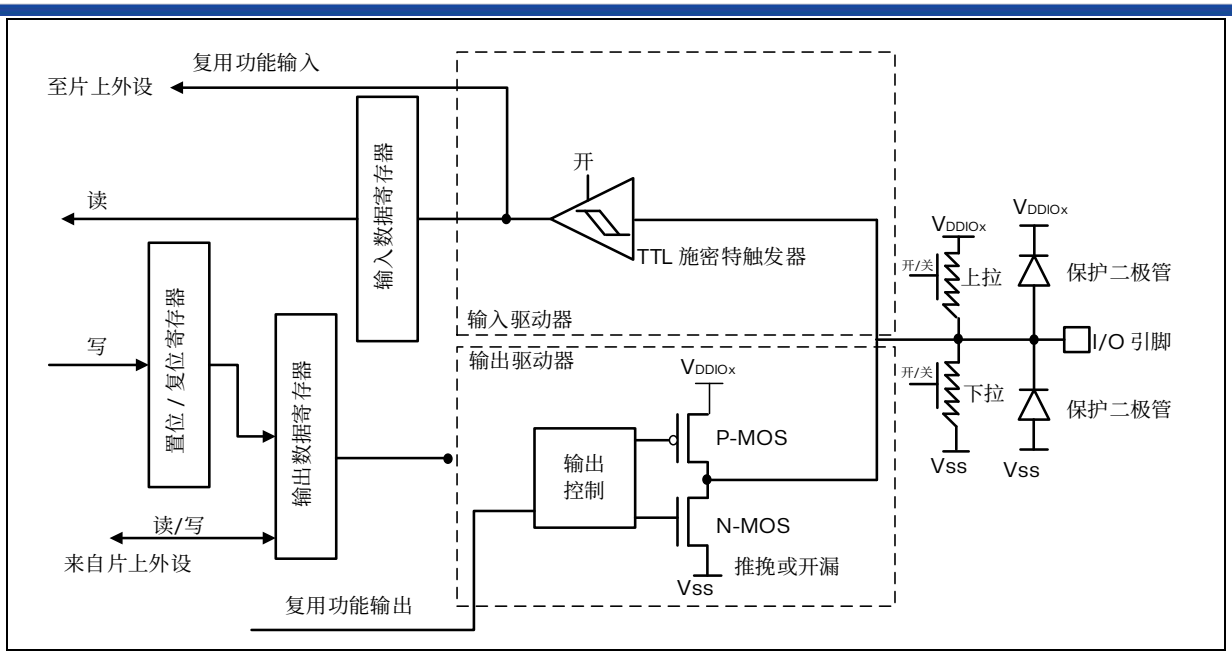


图 8.5 复用功能配置

8.3.12 模拟配置

当 I/O 端口配置为模拟时：

- 输出缓冲器被禁止
- 施密特触发器输入停用，I/O 引脚的每个模拟输入的功耗变为零。施密特触发器的输出被强制处理为恒定值（0）
- 弱上拉和下拉电阻被硬件关闭
- 对输入数据寄存器的读访问值为“0”

下图说明了 I/O 端口位的高阻态模拟输入配置。

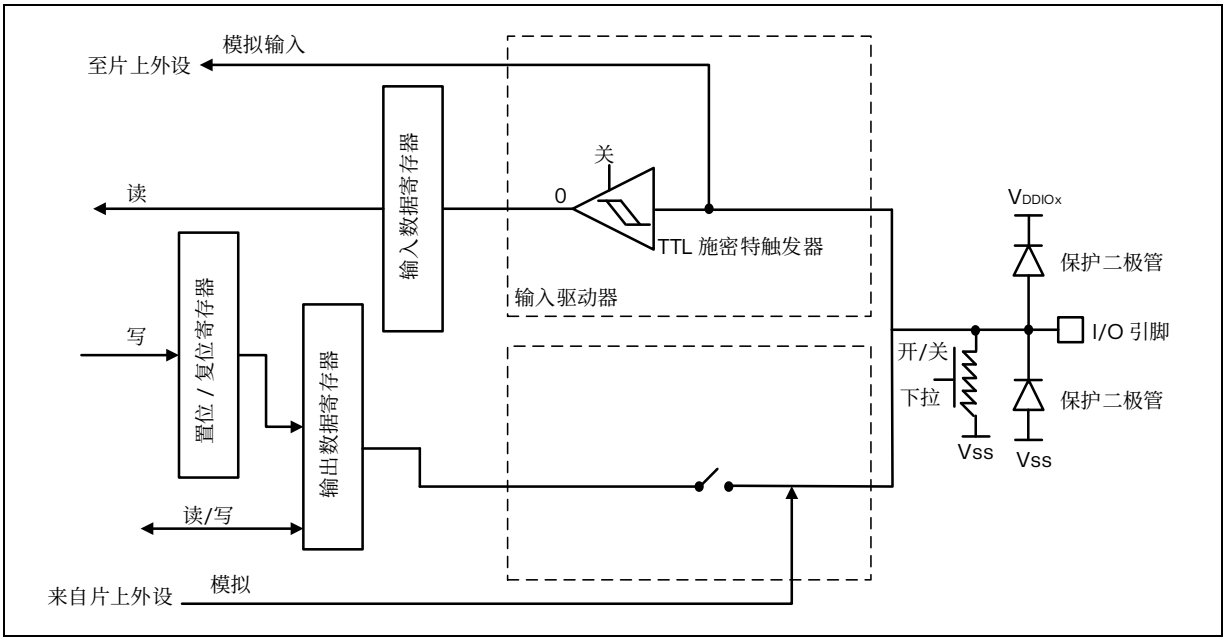


图 8.6 高阻态模拟配置

8.3.13 将 HSE 振荡器引脚用作 GPIO

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

当 HSE 振荡器打开（通过设置 RCC_CR 寄存器中的 HSEON 位）时，振荡器会控制与其相关联的引脚，这些引脚的 GPIO 配置不起作用。

将振荡器配置为用户外部时钟模式时，仅为时钟输入保留 OSC_IN 引脚，OSC_OUT 引脚仍可用作常规 GPIO。

8.4 GPIO 寄存器

本节对 GPIO 寄存器进行了详细介绍。
可按字、半字或字节模式写入外设寄存器。

8.4.1 GPIO 端口模式寄存器（GPIOx_MODER）（x = A...D）

偏移地址：0x00
复位值：0xFFFF FFFA（端口 A）
复位值：0xFFFF FFFF（其他端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]	
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
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **MODE[15:0][1:0]：**端口 x 配置 I/O 引脚 y（y = 0...15）
这些位通过软件写入，用于配置 I/O 模式。
00：输入模式
01：通用输出模式
10：复用功能模式
11：模拟模式（复位状态）

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

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
Reserved															

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **OT[15:0]：**端口 x 配置 I/O 引脚 y（y = 0...15）
这些位通过软件写入，用于配置 I/O 输出类型。
0：推挽输出（复位状态）
1：开漏输出

8.4.3 GPIO 端口输出速度寄存器（GPIOx_OSPEEDR）（x = A...D）

偏移地址：0x08
复位值：0x0000 0003（端口 A）
复位值：0x0000 0000（其他端口）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSPEED15[1:0]		OSPEED14[1:0]		OSPEED13[1:0]		OSPEED12[1:0]		OSPEED11[1:0]		OSPEED10[1:0]		OSPEED9[1:0]		OSPEED8[1:0]	
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
OSPEED7[1:0]		OSPEED6[1:0]		OSPEED5[1:0]		OSPEED4[1:0]		OSPEED3[1:0]		OSPEED2[1:0]		OSPEED1[1:0]		OSPEED0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **OSPEED[15:0][1:0]：**端口 x 配置 I/O 引脚 y（y = 0...15）

这些位通过软件写入，用于配置 I/O 输出速度。

00：低速

01：中速

10：高速

11：超高速

8.4.4 GPIO 端口上拉/下拉寄存器（GPIOx_PUPDR）（x = A...D）

偏移地址：0x0C
复位值：0x0000 0009（端口 A）
复位值：0x0000 0000（其他端口）

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]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]		PUPD7[1:0]	
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
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	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **PUPD[15:0][1:0]：**端口 x 配置 I/O 引脚 y（y = 0...15）

这些位通过软件写入，用于配置 I/O 上拉或下拉。

00：无上拉或下拉（浮空）

01：上拉

10：下拉

11：保留

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

偏移地址：0x10

复位值：0x0000 XXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **ID[15:0]：**端口 x 输入数据 I/O 引脚 y（y = 0...15）

这些位为只读。只能在字模式下访问。它们包含相应 I/O 端口的输入值。

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

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
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **OD[15:0]：**端口 x 输出数据 I/O 引脚 y（y = 0...15）

这些位可通过软件读取和写入。

注：对于原子置位/复位，通过写入 GPIOx_BSRR *或* GPIOx_BRR *寄存器，可分别置位和/或复位* ODR *位* (x = A...E)。

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

偏移地址：0x18

复位值：0x0000 0000

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

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

Bit 31:16 **BR[15:0]：**端口 x 复位 I/O 引脚 y（y = 0...15）

这些位为只写。读取这些位可返回值 0x0000。

0：对应 ODx 位无操作

1：复位对应的 ODx 位

注：如果同时对 BSx *和* BRx *置位，则* BSx *的优先级更高。*

Bit 15:0 **BS[15:0]：**端口 x 置位 I/O 引脚 y（y = 0...15）

这些位为只写。读取这些位可返回值 0x0000。

0：对应 ODx 位无操作

1：置位对应的 ODx 位

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

当正确的写序列应用到第 16 位（LCKK）时，此寄存器将用于锁定端口位的配置。bit[15:0]的值用于锁定 GPIO 的配置。在写序列期间，不能更改 LCKR[15:0]的值。将 LOCK 序列应用到某个端口位后，在执行下一次 MCU 复位或外设复位之前，将无法对该端口位的值进行修改。

注：可使用特定的写序列对 GPIOx_LCKR 寄存器执行写操作。在此锁定序列期间只允许使用字访问（32 位长）。

每个锁定位冻结一个特定的配置寄存器（控制寄存器和复用功能寄存器）。

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															LCKK
															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:17 保留，必须保持为复位值。

Bit 16 **LCKK：**锁定键

可随时读取此位。可使用锁定键写序列对其进行修改。

0：端口配置锁定键未激活

1：端口配置锁定键已激活。到下一次 MCU 复位或外设复位前，GPIOx_LCKR 寄存器将一直处于锁定状态。

锁定键写序列：

WR LCKR[16] = “1” + LCKR[15:0]

WR LCKR[16] = “0” + LCKR[15:0]

WR LCKR[16] = “1” + LCKR[15:0]

RD LCKR

RD LCKR[16] = “1”（此读操作为可选操作，但它可确认锁定已激活）

注：在锁定键写序列期间，不能更改 LCK[15:0]的值。

锁定序列中的任何错误都将中止锁定操作。

在任一端口位上的第一个锁定序列之后，对 LCKK 位的任何读访问都将返回“1”，直到下一次 MCU 复位或外设复位为止

Bit 15:0 **LCK[15:0]：**端口 x 锁定 I/O 引脚 y（y = 0...15）

这些位都是读/写位，但只能在 LCKK 位等于“0”时执行写操作。

0：端口配置未锁定

1：端口配置已锁定

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

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]			
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
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

Bit 31:0 **AFSEL[7:0][3:0]：**端口 x 引脚 y 的复用功能选择（y = 0...7）

这些位通过软件写入，用于配置复用功能 I/O。

0000：AF0

0001：AF1

0010：AF2

0011：AF3

0100：AF4

0101：AF5

0110：AF6

0111：AF7

其他：无作用

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

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
AFSEL15[3:0]				AFSEL14[3:0]				AFSEL13[3:0]				AFSEL12[3:0]			
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
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

Bit 31:0 **AFSEL[15:8][3:0]：**端口 x 引脚 y 的复用功能选择（y = 8...15）

这些位通过软件写入，用于配置复用功能 I/O。

0000：AF0

0001：AF1

0010：AF2

0011：AF3

0100：AF4

0101：AF5

0110：AF6

0111：AF7

其他：无作用

8.4.11 GPIO 端口位复位寄存器（GPIOx_BRR）（x = A...D）

偏移地址：0x28

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit 31:16 保留，必须保持为复位值

Bit 15:0 **BR[15:0]：**端口 x 复位 I/O 引脚 y（y = 0...15）

这些位只能写。读取这些位将返回值 0x0000。

0：对应 ODx 位无操作

1：复位对应的 ODx 位

9 系统配置控制器（SYSCFG）

9.1 SYSCFG 主要特征

RX32H710 系列器件配备了一组配置寄存器。系统配置控制器的主要特征如下：

- 管理外部中断线与 GPIOs 的连接
- 管理健壮性功能

9.2 SYSCFG 寄存器

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

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 0 到 3）

这些位通过软件写入，以选择外部中断/事件检测的 EXTI 输入的源输入。

0000：PA[x]引脚

0001：PB[x]引脚

0010：PC[x]引脚

0011：PD[x]引脚

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

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 4 到 7）

这些位通过软件写入，以选择外部中断/事件检测的 EXTI 输入的源输入。

0000：PA[x]引脚

0001：PB[x]引脚

0010：PC[x]引脚

0011：PD[x]引脚

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

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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI11[3:0]				EXTI10[3:0]				EXTI9[3:0]				EXTI8[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 8 到 11）

这些位通过软件写入，以选择外部中断/事件检测的 EXTI 输入的源输入。

0000：PA[x]引脚

0001：PB[x]引脚

0010：PC[x]引脚

0011：PD[x]引脚

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

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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXTI15[3:0]				EXTI14[3:0]				EXTI13[3:0]				EXTI12[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **EXTIx[3:0]**：EXTI x 配置（x = 12 到 15）

这些位通过软件写入，以选择外部中断/事件检测的 EXTI 输入的源输入。

0000：PA[x]引脚

0001：PB[x]引脚

0010：PC[x]引脚

0011：PD[x]引脚

9.2.5 SYSCFG 配置寄存器 2（SYSCFG_CFGR2）

地址偏移：0x1C

复位值：0x0000 0000

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

Bit 31:3 保留，必须保持为复位值。

Bit 2 **PVDL**：PVD 锁定使能位

此位由软件设置，并且只能通过系统复位来清除。它可用于使能并锁定 PVD 连接到 TIM1/8/15 刹车输入，以及 PWR_CR2 寄存器中的 PVDE 和 PLS[2:0]。

0：PVD 中断与 TIM1/8/15 刹车输入断开。PVDE 和 PLS[2:0]位可由应用程序编程。

1：PVD 中断连接到 TIM1/8/15 刹车输入，PVDE 和 PLS[2:0]位为只读。

Bit 1 保留，必须保持为复位值。

Bit 0 **CLL**：内核锁住信号输出使能位

此位由软件设置，并且只能通过系统复位来清除。它可用于使能并将内核锁住信号输出连接到 TIM1/8/15 刹车输入。

0：内核锁住输出与 TIM1/8/15 刹车输入断开。

1：内核锁住输出连接到 TIM1/8/15 刹车输入。

10 外部中断和事件控制器（EXTI）

10.1 简介

EXTI 的主要特征包括：

- 可生成多达 21 个事件/中断请求
  - 21 个可配置线
- 每个事件/中断线具有独立屏蔽功能
- 可配置上升沿或下降沿触发
- 专用状态位
- 模拟事件/中断请求

10.2 EXTI 主要特性

外部中断和事件控制器（EXTI）负责管理外部和内部异步事件/中断，并向 CPU/中断控制器生成事件请求，以及向电源控制器生成唤醒请求。

EXTI 能够管理多达 21 条可从停止模式唤醒的事件线。

这些事件线为可配置型：

- 可配置型事件线：可独立选择激活边沿，并通过状态标志指示中断来源。此类事件线主要用于 I/O 的外部中断以及部分外设。

每条线均可独立屏蔽，以用于中断或事件生成。

此外，该控制器还允许通过向专用寄存器写入数据，以软件方式模拟事件或中断，并与相应的硬件事件线进行多路复用。

10.3 EXTI 功能介绍

对于可配置的中断线，需要对其进行配置并使能才能生成中断。这通过编程两个触发寄存器以设置所需的边沿检测，并通过在中断屏蔽寄存器的相应位写入 1 来使能中断请求来实现。当中断线上出现所选边沿时，将生成一个中断请求，并且与中断线对应的挂起位也会被设置。通过向挂起寄存器中写入 1，可以清除此请求。

直接中断线其中断在中断屏蔽寄存器中默认是使能的，并且在挂起寄存器中没有与之对应的挂起位。

为了生成事件，需要配置并使能事件线。这同样通过编程两个触发寄存器以设置所需的边沿检测，并通过在事件屏蔽寄存器的相应位写入 1 来使能事件请求来实现。当事件线上出现所选边沿时，将生成一个事件脉冲，但与事件线对应的挂起位不会被设置。

对于可配置线路，还可以通过在软件中断/事件寄存器中写入 1 来由软件生成中断/事件请求。

注：与直接线路相关联的中断或事件仅在系统处于停止模式时被触发。如果系统仍在运行，EXTI 不会生成任何中断/事件。

10.3.1 EXTI 框图

外部中断/事件框图如下图所示。

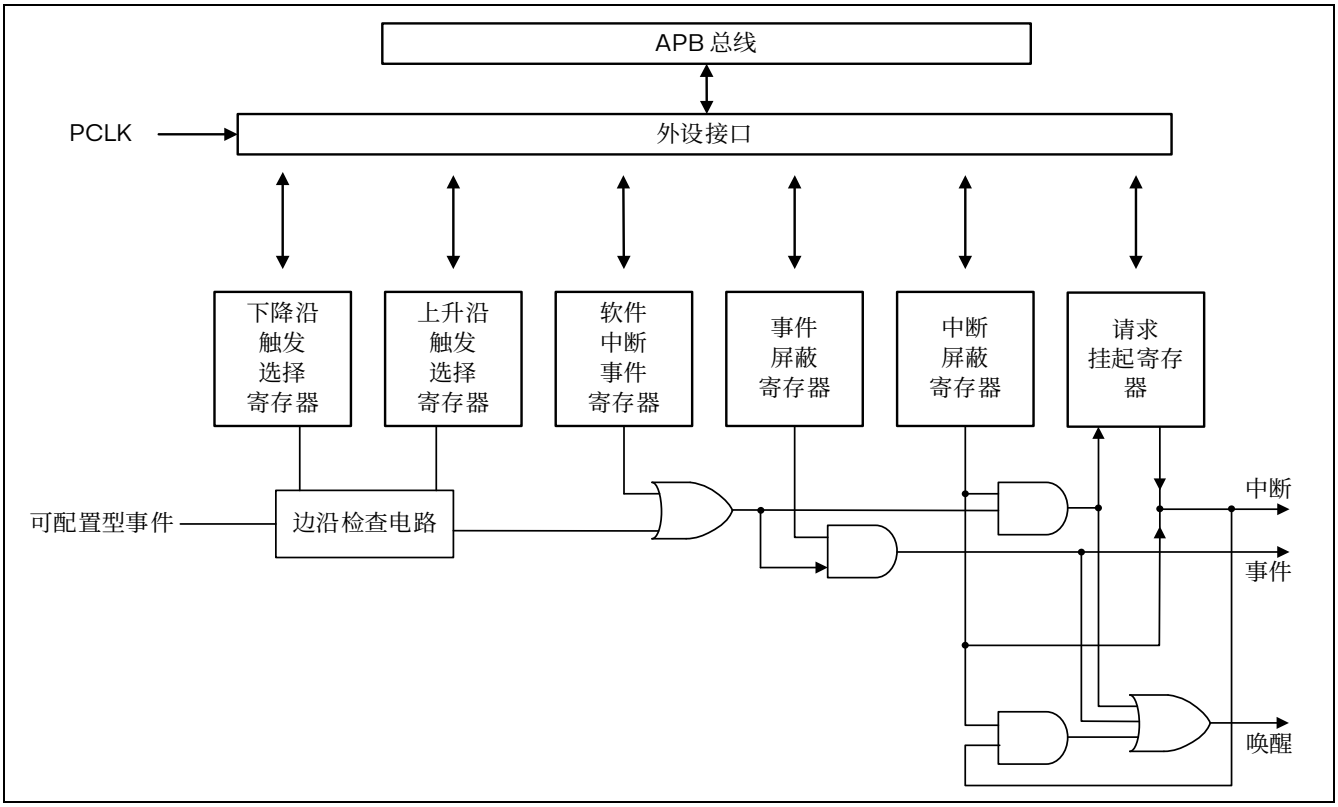


图 10.1 可配置的中断/事件框图

10.3.2 硬件中断选择

要将某条线路配置为中断源，请按照以下步骤操作：

1. 配置 EXTI_IMR 寄存器中的相应屏蔽位。
2. 配置中断线路的触发选择位（EXTI_RTSR 和 EXTI_FTSR）。
3. 配置控制映射到 EXTI 的中断通道的使能和屏蔽位，以便能够正确响应来自 EXTI 线路的中断请求。

10.3.3 硬件事件选择

要将某条线路配置为事件源，请按照以下步骤操作：

1. 配置 EXTI_EMR 寄存器中的相应屏蔽位。
2. 配置事件线路的触发选择位（EXTI_RTSR 和 EXTI_FTSR）。

10.3.4 软件中断/事件选择

任何可配置线路都可以配置为软件中断/事件线路。生成软件中断的步骤如下：

1. 配置相应的屏蔽位（EXTI_IMR，EXTI_EMR）。
2. 设置软件中断寄存器（EXTI_SWIER）中的所需位。

10.4 EXTI 中断/事件线映射

GPIO 连接到 16 条可配置的中断/事件线（参见下图）。

注：GPIOD 端口仅 10 条可配置的中断/事件线。

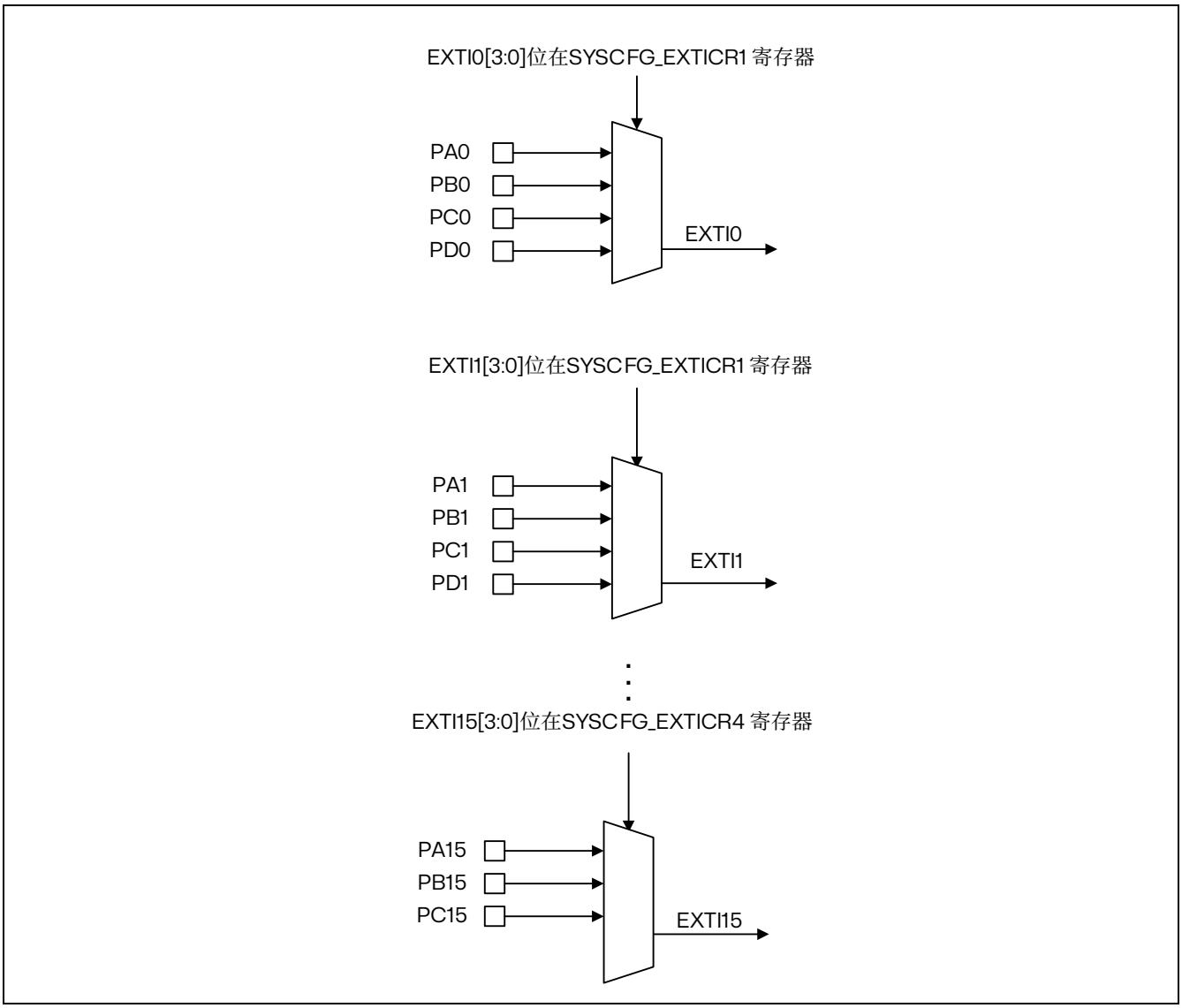


图 10.2 外部中断/事件 GPIO 映射

表 10.1 EXTI 线路连接

EXTI 线	线路源	线路类型
0~15	GPIO	可配置
16	PVD	可配置
17	RTC 闹钟事件	可配置
18	保留	-
19	保留	-
20	保留	-
21	COMP1 输出	可配置
22	COMP2 输出	可配置
23	保留	-
24	保留	-
25	保留	-
26	COMP3 输出	可配置
27	保留	-
28	保留	-
29	COMP4 输出	可配置

10.5 EXTI 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。
外设寄存器必须按字（32 位）进行访问。

10.5.1 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x00
复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		IM29	Reserved		IM26	Reserved			IM22	IM21	Reserved			IM17	IM16
		rw			rw				rw	rw				rw	rw

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

Bit 31:30 保留，必须保持为复位值。

Bit 29 **IM29：**线 29 上的中断屏蔽
0：屏蔽来自线 29 上的中断请求
1：开放来自线 29 上的中断请求

Bit 28:27 保留，必须保持为复位值。

Bit 26 **IM26：**线 26 上的中断屏蔽
0：屏蔽来自线 26 上的中断请求
1：开放来自线 26 上的中断请求

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **IMx：**线 x 上的中断屏蔽（x = 22~21）
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

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **IMx：**线 x 上的中断屏蔽（x = 17~0）
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

10.5.2 事件屏蔽寄存器（EXTI_EMR）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		EM29	Reserved		EM26	Reserved			EM22	EM21	Reserved			EM17	EM16
		rw			rw				rw	rw				rw	rw

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

Bit 31:30 保留，必须保持为复位值。

Bit 29 **EM29：**线 29 上的事件屏蔽

0：屏蔽来自线 29 上的事件请求

1：开放来自线 29 上的事件请求

Bit 28:27 保留，必须保持为复位值。

Bit 26 **EM26：**线 26 上的事件屏蔽

0：屏蔽来自线 26 上的事件请求

1：开放来自线 26 上的事件请求

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **EMx：**线 x 上的事件屏蔽（x = 22~21）

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

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **EMx：**线 x 上的事件屏蔽（x = 17~0）

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

10.5.3 上升沿触发选择寄存器（EXTI_RTSR）

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		RT29	Reserved		RT26	Reserved			RT22	RT21	Reserved			RT17	RT16
		rw			rw				rw	rw				rw	rw

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

Bit 31:30 保留，必须保持为复位值。

Bit 29 **RT29：**线 29 上的上升沿触发事件配置位

0：禁止输入线 29 上的上升沿触发（中断和事件）

1：使能输入线 29 上的上升沿触发（中断和事件）

Bit 28:27 保留，必须保持为复位值。

Bit 26 **RT26：**线 26 上的上升沿触发事件配置位

0：禁止输入线 26 上的上升沿触发（中断和事件）

1：使能输入线 26 上的上升沿触发（中断和事件）

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **RTx：**线 x 的上升触发事件配置位（x = 22~21）

0：禁止输入线 x 上的上升沿触发（中断和事件）

1：使能输入线 x 上的上升沿触发（中断和事件）

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **RTx：**线 x 的上升触发事件配置位（x = 17~0）

0：禁止输入线 x 上的上升沿触发（中断和事件）

1：使能输入线 x 上的上升沿触发（中断和事件）

注：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_RTSR 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。

10.5.4 下降沿触发选择寄存器（EXTI_FTSR）

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		FT29	Reserved		FT26	Reserved			FT22	FT21	Reserved			FT17	FT16
		rw			rw				rw	rw				rw	rw

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

Bit 31:30 保留，必须保持为复位值。

Bit 29 **FT29：**线 29 上的下降沿触发事件配置位

0：禁止输入线 29 上的下升沿触发（中断和事件）

1：使能输入线 29 上的下升沿触发（中断和事件）

Bit 28:27 保留，必须保持为复位值。

Bit 26 **FT26：**线 26 上的下降沿触发事件配置位

0：禁止输入线 26 上的下升沿触发（中断和事件）

1：使能输入线 26 上的下升沿触发（中断和事件）

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **FTx：**线 x 上的下降沿触发事件配置位（x = 22~21）

0：禁止输入线 x 上的下升沿触发（中断和事件）

1：使能输入线 x 上的下升沿触发（中断和事件）

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **FTx：**线 x 上的下降沿触发事件配置位（x = 17~0）

0：禁止输入线 x 上的下升沿触发（中断和事件）

1：使能输入线 x 上的下升沿触发（中断和事件）

注：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_FTSR *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。

10.5.5 软件中断事件寄存器（EXTI_SWIER）

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SWI29	Reserved		SWI26	Reserved			SWI22	SWI21	Reserved			SWI17	SWI16
		rw			rw				rw	rw				rw	rw

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

Bit 31:30 保留，必须保持为复位值。

Bit 29 **SWI29：**线 29 上的软件中断

当该位为“0”时，写“1”将设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。

通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

Bit 28:27 保留，必须保持为复位值。

Bit 26 **SWI26：**线 26 上的软件中断

当该位为“0”时，写“1”将设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。

通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **SWIx：**线 x 上的软件中断（x = 22~21）

当该位为“0”时，写“1”将设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。

通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **SWIx：**线 x 上的软件中断（x = 17~0）

当该位为“0”时，写“1”将设置 EXTI_PR 中相应的挂起位。如果在 EXTI_IMR 和 EXTI_EMR 中允许产生该中断，则此时将产生一个中断。

通过清除 EXTI_PR 的对应位（写入“1”），可以清除该位为“0”。

10.5.6 挂起寄存器（EXTI_PR）

偏移地址：0x14

复位值： 0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PIF29	Reserved		PIF26	Reserved			PIF22	PIF21	Reserved			PIF17	PIF16
		rc_w1			rc_w1				rc_w1	rc_w1				rc_w1	rc_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIF15	PIF14	PIF13	PIF12	PIF11	PIF10	PIF9	PIF8	PIF7	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit 31:30 保留，必须保持为复位值。

Bit 29 **PIF29：**线 29 上挂起的中断标志

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”可以清除该位。

Bit 28:27 保留，必须保持为复位值。

Bit 26 **PIF26：**线 26 上挂起的中断标志

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”可以清除该位。

Bit 25:23 保留，必须保持为复位值。

Bit 22:21 **PIFx：**线 x 上挂起的中断标志（x = 22~21）

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”可以清除该位。

Bit 20:18 保留，必须保持为复位值。

Bit 17:0 **PIFx：**线 x 上挂起的中断标志（x = 17~0）

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”可以清除该位。

11 CORDIC 协同处理器（CORDIC）

11.1 CORDIC 简介

CORDIC 协处理器为电机控制、计量、信号处理以及许多其他应用中常用的数学函数（主要是三角函数）提供硬件加速。

与软件实现相比，它加快了这些函数的计算速度，使得可以使用更低的操作频率，或者释放处理器周期以执行其他任务。

11.2 CORDIC 主要特征

- 24 位 CORDIC 旋转引擎
- 圆形模式
- 旋转和向量模式
- 函数：正弦 sin、余弦 cos、反正切 atan、模 modulus、
- 可编程精度
- 低延迟 AHB 从接口
- 结果准备好后即可读取，无需轮询或中断

11.3 CORDIC 功能描述

11.3.1 通用描述

CORDIC 是求解三角函数的一种经济高效的连续逼近算法。

在三角函数（圆形）模式下，通过以递减的角度旋转单位向量[1, 0]，直到旋转角度的累积和等于输入角度 θ，从而确定角度 θ 的正弦和余弦。旋转后向量的 x 和 y 笛卡尔分量分别对应于 θ 的余弦和正弦。相反地，对应于反正切（y/x）的向量[x, y]的角度，是通过将[x, y]以递减的角度逐次旋转以获得单位向量[1,0] 来确定的。旋转角度的累积和给出了原始向量的角度。

其他函数可以从上述基本函数中派生出来。

11.3.2 CORDIC 函数

使用协处理器的第一步是通过相应编程 CORDIC_CR 寄存器的 FUNC 位域来选择所需的函数。

下表列出了 CORDIC 协处理器支持的函数。

表 11.1 CORDIC 函数

函数	描述	主要参数（ARG1）	次要参数（ARG2）	主要结果（RES1）	次要结果（RES2）
cos	余弦	角度 θ	模 m	m • cos θ	m • sin θ
sin	正弦	角度 θ	模 m	m • sin θ	m • cos θ
phase	相位	x	y	atan2(y,x)	$\sqrt{x^2+y^2}$
modulus	模	x	y	$\sqrt{x^2+y^2}$	atan2(y,x)
atan	反正切	x	无	$\tan^{-1}$ x	无

一些函数接受两个输入参数（ARG1 和 ARG2），并且一些函数同时生成两个结果（RES1 和 RES2）。这是算法的一个附加功能，意味着只需一次操作即可获得两个值。例如，在执行极坐标到直角坐标转换时就是这种情况：sin θ 同时生成 cos θ，cos θ 同时生成 sin θ。类似地，对于直角坐标到极坐标转换（phase(x,y)，modulus(x,y)）也是如此。

对于某些函数（atan），可以应用一个缩放系数（见第 12.3.4 节）来扩展函数的范围，使其超出 q1.15 定点格式支持的最大范围[-1, 1]。对于所有其他圆形函数，缩放系数必须设置为 0。

cos 余弦

表 11.2 余弦参数

参数	描述	范围
ARG1	θ 角度（以弧度为单位），除以 π	[-1, 1]
ARG2	模 m	[0, 1]
RES1	m • cos θ	[-1, 1]
RES2	m • sin θ	[-1, 1]
SCALE	不适用	0

此函数计算范围在-π 到 π 之间的角度的余弦值。它也可以用于执行极坐标到直角坐标的转换。

主要参数是以弧度表示的角度θ。在编程 ARG1 之前，必须将其除以 π。

次要参数是模 m。如果 m 大于 1，则必须在软件中应用缩放以适应 ARG2 的 q1.15 范围。

主要结果 RES1 是角度的余弦值乘以模。

次要结果 RES2 是角度的正弦值乘以模。

sin 正弦

表 11.3 正弦参数

参数	描述	范围
ARG1	θ 角度（以弧度为单位），除以 π	[-1, 1]
ARG2	模 m	[0, 1]
RES1	m • sin θ	[-1, 1]
RES2	m • cos θ	[-1, 1]
SCALE	不适用	0

此函数计算范围在-π 到 π 之间的角度的正弦值。它也可以用于执行极坐标到直角坐标的转换。

主要参数是以弧度表示的角度θ。在编程 ARG1 之前，必须将其除以 π。

次要参数是模 m。如果 m 大于 1，则必须在软件中应用缩放以适应 ARG2 的 q1.15 范围。

主要结果 RES1 是角度的正弦值乘以模。

次要结果 RES2 是角度的余弦值乘以模。

phase 相位

表 11.4 相位参数

参数	描述	范围
ARG1	x 坐标	[-1, 1]
ARG2	y 坐标	[-1, 1]
RES1	相位角θ（以弧度为单位），除以 π	[-1, 1]
RES2	m • cos θ	[0, 1]
SCALE	不适用	0

此函数计算向量 **v** = [x y]在-π 到 π 范围内的相位角（也称为 atan2(y,x)）。它也可以用于执行直角坐标到极坐标的转换。

主要参数是 x 坐标，即向量在 x 轴方向上的大小。如果|x| > 1，则必须在软件中应用缩放以适应 ARG1 的 q1.15 范围。

次要参数是 y 坐标，即向量在 y 轴方向上的大小。如果|y| > 1，则必须在软件中应用缩放以适应 ARG2 的 q1.15 范围。

主要结果 RES1 是向量 **v** 的相位角 θ。为了获得以弧度表示的角度，RES1 必须乘以 π。请注意，由于相位角的循环性质，接近 π 的值有时会回绕到-π。

次要结果 RES2 是模，由下式给出：|v|=$\sqrt{x^2 + y^2}$ 。如果|v| > 1，则 RES2 中的结果将饱和到 1。

modulus 模

表 11.5 模参数

参数	描述	范围
ARG1	x 坐标	[-1, 1]
ARG2	y 坐标	[-1, 1]
RES1	模 m	[0, 1]
RES2	相位角θ	[-1, 1]
SCALE	不适用	0

此函数计算向量 **v** = [x y]的大小（或称模）。它也可以用于执行直角坐标到极坐标的转换。

主要参数是 x 坐标，即向量在 x 轴方向上的分量大小。如果|x| > 1，则必须在软件中应用缩放以适应 ARG1 的 q1.15 范围。

次要参数是 y 坐标，即向量在 y 轴方向上的分量大小。如果|y| > 1，则必须在软件中应用缩放以适应 ARG2 的 q1.15 范围。

主要结果 RES1 是模，由下式给出：$|v|=\sqrt{x^2+y^2}$。如果|**v**| > 1，则 RES1 的结果将饱和到 1。

次要结果 RES2 是向量 **v** 的相位角 θ。为了获得以弧度表示的角度，RES2 必须乘以 π。请注意，由于相位角的循环性质，接近 π 的值有时会回绕到-π。

atan 反正切

表 11.6 反正切参数

参数	描述	范围
ARG1	$x \cdot 2^{-n}$	[-1, 1]
ARG2	不适用	-
RES1	$2^{-n} \cdot \tan^{-1} x$（以弧度为单位），除以π	[-1, 1]
RES2	不适用	-
SCALE	n	[0 7]

此函数计算输入参数 x 的反正切值，即反正切（或逆正切）。

主要参数 ARG1 是输入值 x = tanθ。如果|x| > 1，则必须在软件中应用一个 2^{-n} 的缩放系数，使得$-1 < x \cdot 2^{-n} < 1$。将缩放后的值 $x \cdot 2^{-n}$ 编程到 ARG1 中，并将缩放系数 n 编程到 SCALE 参数中。

请注意，允许的最大输入值是 tanθ = 128，这对应于角度 θ = 89.55 度。对于|x| > 128 的情况，必须使用软件方法来求解 $\tan^{-1} x$。

次要参数 ARG2 未使用。

主要结果 RES1 是角度 $\theta = \tan^{-1} x$。为了获得以弧度表示的角度，RES1 必须乘以 $2^n \cdot \pi$。

次要结果 RES2 未使用。

11.3.3 定点表示法

CORDIC 采用定点有符号整数格式运算。输入和输出值固定为 q1.15 格式。

在 q1.15 格式中，数值范围为 1（0x8000）至 $1-2^{-15}$（0x7FFF）。此格式的优点在于，两个输入参数可以打包到一个 32 位写操作中，且两个结果可以通过一次 32 位读取操作获取。

11.3.4 缩放系数

上述章节中列出的几个函数指定了一个缩放系数 SCALE。这使得函数输入范围得以扩展，以覆盖 CORDIC 支持的全值范围，而不会使输入、输出或内部寄存器饱和。如果需要使用缩放系数，则必须通过软件计算并将其编程到 CORDIC_CSR 寄存器的 SCALE 位域中。

在将缩放后的值编程到 CORDIC_WDATA 寄存器之前，必须相应地对输入参数进行缩放。同时，在从 CORDIC_RDATA 寄存器读取结果时，也必须对缩放进行还原。

注：由于缩放值的截断，缩放系数会导致精度损失。

11.3.5 精度

结果的精度取决于 CORDIC（坐标旋转数字计算）的迭代次数。对于三角函数（正弦、余弦、相位、模量），该算法以每次迭代一个二进制位的恒定速率收敛，见下图。

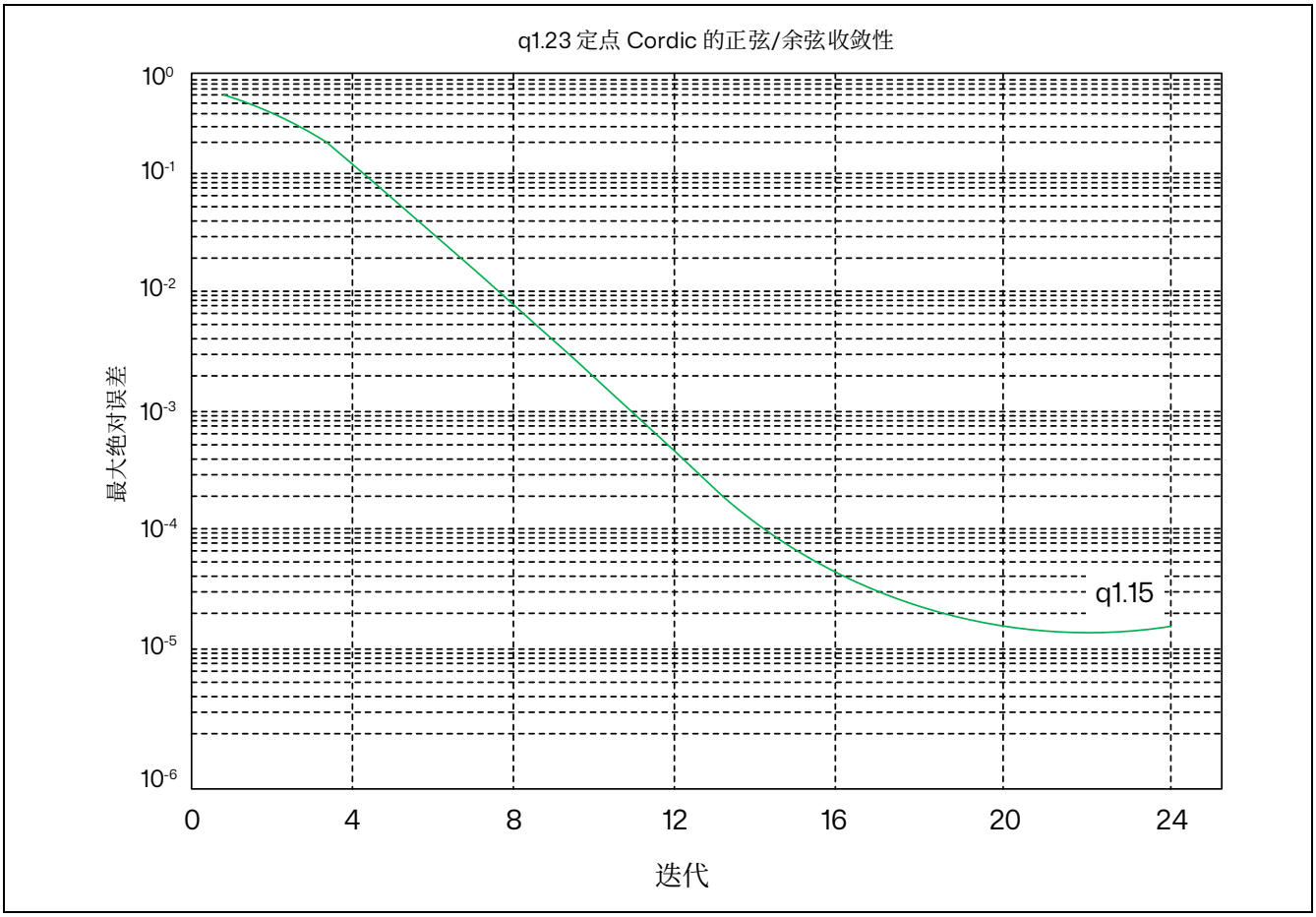


图 11.1 CORDIC 在三角函数中的收敛情况

注：当量化误差开始变得显著时，收敛速率会降低。

CORDIC 每个时钟周期可执行一次迭代。对于每个函数，下表中列出了每一次迭代后剩余的最大误差，以及达到该精度所需的时钟周期数。根据此表，可以确定所需的周期数，并将其编程到 CORDIC_CR 寄存器的 PRECISION 位域中。协处理器在完成编程的迭代次数后立即停止，并且可以立即读取结果。

表 11.7 精度与迭代次数

函数	迭代次数	循环次数	最大剩余误差(1)
			q1.15 格式
Sin, Cos, Phase(2), Mod, Atan(3)	1	1	2^{0}
	2	2	2^{-1}
	3	3	2^{-2}
	4	4	2^{-3}
	5	5	2^{-4}
	6	6	2^{-5}

1. 最大剩余误差是指在给定迭代次数后剩余的最大误差，与以双精度浮点执行的相同计算相比。可能会产生额外的舍入误差，对于 q15 格式最多为 2^{-16}。
2. 对于模量>0.5 的情况，随着量化误差变得显著，可实现的精度会按比例降低。
3. SCALE=0。如果使用更高的缩放系数，可实现的精度会按比例降低。

11.3.6 零开销模式

使用协处理器的最快方法是在 CORDIC_CSR 寄存器中预编程要执行的功能（FUNC）、所需的时钟周期数（PRECISION）以及适用的缩放系数（SCALE）。

计算通过将输入参数写入 CORDIC_WDATA 寄存器来触发。一旦写入了正确数量的输入参数（且任何正在进行的计算已完成），就会使用这些输入参数和当前的 CORDIC_CSR 设置启动新的计算。如果没有更改，则无需重新编程 CORDIC_CSR 寄存器。

输入参数默认为两个 16 位参数，必须将它们打包到一个 32 位字中，其中 ARG1 位于最低有效半字中，ARG2 位于最高有效半字中。然后，将打包的 32 位字写入 CORDIC_WDATA 寄存器。

一旦计算开始，任何尝试读取 CORDIC_RDATA 寄存器的操作都会插入总线等待状态，直到计算完成并返回结果。然后，软件可以写入输入并立即读取结果，而无需轮询查看其是否有效。或者，处理器可以在读取结果之前等待适当数量的时钟周期。这段时间可以用于为下一次计算编程 CORDIC_CSR 寄存器，并在需要时准备下一个输入数据。在计算进行中，可以重新编程 CORDIC_CSR 寄存器，而不会影响正在进行的计算的结果。同样，一旦考虑到前一个参数，就可以使用下一个参数更新 CORDIC_WDATA 寄存器。下一个参数和设置将保持挂起状态，直到前一个计算完成。

当计算完成时，可以从 CORDIC_RDATA 寄存器读取结果。结果默认为 2 个 16 位参数，即 CORDIC_RDATA 的一次读取会获取打包到一个 32 位字中的两个结果。RES1 位于较低半字中，RES2 位于较高半字中。

当下一个计算所需的参数数量已被写入，且已读取预期数量的结果时，下一个计算就会开始。这意味着在任何时候，都可能有一个计算正在进行中，或等待读取结果，以及一个操作处于挂起状态。当操作处于挂起状态时，对 CORDIC_WDATA 的任何进一步访问都会取消它并覆盖数据。

以下序列总结了在零开销模式下使用 CORDIC_IP 的步骤:

1. 使用适当的设置编程 CORDIC_CSR 寄存器
2. 在 CORDIC_WDATA 寄存器中为第一次计算编程参数。这将启动第一次计算。
3. 如果需要，为下一次计算更新 CORDIC_CSR 寄存器设置。
4. 在 CORDIC_WDATA 寄存器中为下一次计算编程参数。
5. 从 CORDIC_RDATA 寄存器读取结果。这将触发下一次计算。
6. 转到步骤 3。

11.3.7 轮询方式

当 CORDIC_RDATA 寄存器中有新结果可用时，CORDIC_CSR 寄存器中会设置 RRDY 标志。通过读取该寄存器可以轮询此标志。读取 CORDIC_RDATA 寄存器可以复位该标志。

轮询 RRDY 标志的时间比直接读取 CORDIC_RDATA 寄存器稍长，因为在结果可用时不会立即读取。读取 CORDIC_CSR 寄存器时，处理器和总线接口不会停滞，因此如果无法接受处理器停滞（例如，必须处理低延迟中断），则此模式可能适用。

11.3.8 中断模式

通过在 CORDIC_CSR 寄存器中设置中断使能（IE）位，每当设置 RRDY 标志时，就会生成一个中断。当标志复位时，中断被清除。

此模式允许在中断服务例程中读取计算结果，因此相对于其他任务具有优先级。然而，由于中断处理延迟，它比直接读取结果或轮询标志更慢。

11.4 CORDIC 寄存器描述

CORDIC 寄存器仅支持 32 位字格式访问

11.4.1 CORDIC 控制/状态寄存器（CORDIC_CSR）

偏移地址：0x00

复位值：0x0000 0050

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

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

Bit 31 **RRDY**：结果就绪标志

0：输出寄存器无新数据

1：CORDIC_RDATA 寄存器有新数据

该位在 CORDIC 操作完成时由硬件设置，当 CORDIC_RDATA 寄存器被读取（NRES+1）次后由硬件清除。此位设置时，若 IEN 位也设置，则触发 CORDIC 中断；此位设置期间，不会启动新计算。

Bit 30:17 保留，必须保持为复位值。

Bit 16 **IEN**：使能中断。

0：禁用。不生成中断请求。

1：使能。当 RRDY 标志设置时，生成中断请求。

此位由软件设置和清除。读取返回该位的当前状态。

Bit 15:11 保留，必须保持为复位值。

Bit 10:8 **SCALE[2:0]**：缩放系数

此位域的值表示应用于参数和/或结果的缩放系数。值 n 意味着参数已乘以 2^{-n}，和/或结果需要乘以 2^{n}。有关每个功能的缩放系数适用性和适当范围，请参阅第 12.3.2 节。

Bit 7:4 **PRECISION[3:0]**：精度要求（迭代次数）

0：保留

1 至 6：（迭代次数）/1

要确定给定精度所需的迭代次数，请参阅表：精度与迭代次数。注意，对于大多数功能，此位域的推荐范围为 3 至 6。

Bit 3:0 **FUNC[3:0]**：功能

0：余弦 cos

1：正弦 sin

2：相位 phase

3：模 modulus

4：反正切 atan

5 至 15：保留

11.4.2 CORDIC 参数寄存器（CORDIC_WDATA）

偏移地址：0x04

复位值：0xXXXX XXXX

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

Bit 31:0 **ARG[31:0]：**函数输入参数

此寄存器用于编程存储在 CORDIC_CSR 寄存器 FUNC 位域中选定函数的输入参数。

写入此寄存器即包含两个 16 位参数。主要参数（ARG1）位于低位半字（ARG[15:0]），次要参数（ARG2）位于高位半字（ARG[31:16]）。

有关每个函数所需的参数及其允许范围，请参阅第章节：CORDIC 函数。

当所需数量的参数已写入后，若之前的计算已完成，CORDIC 将使用提供的输入参数评估 CORDIC_CSR.FUNC 指定的函数。如果计算正在进行中，ARG1 和 ARG2 的值将保持等待状态，直至计算完成并读取结果。在此期间，对此寄存器的写入将取消等待中的操作并覆盖参数数据。

11.4.3 CORDIC 结果寄存器（CORDIC_RDATA）

偏移地址：0x08

复位值：0x0000 0000

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

Bit 31:0 **RES[31:0]：**函数的结果

读取此寄存器即包含两个 16 位参数。此寄存器包含主要结果（RES1）在低位半字（RES[15:0]），次要结果（RES2）在高位半字（RES[31:16]）。

对此寄存器的读取会复位 CORDIC_CSR 寄存器中的 RRDY 标志。

12 数学运算协同处理器（ME）

12.1 简介

该设备提供一个数学运算协同处理器（Math Engine），简称 ME。包含除法运算和一个开平方运算。

12.2 功能描述

ARGx 和 RESx 属于同一个寄存器，写入时，是写入到 ARGx，当运算完成（ME_CR 寄存器 BSY 位为 0 时）读取 RESx 值获取运算结果。

表 12.1 函数运算表

函数	ARG1（写入）	ARG2（写入）	RES1（读取）	RES2（读取）
DIV	x	y	x/y	x%y
SQRT	x	无	$\sqrt{x}$	无

运算步骤：

1. 选择 DIV 除法运算（FUN=0）还是 SQRT 开平方运算（FUN=1）；
2. ARG1 填入后自动启动运算，有 1 个以上参数需要先填入其他参数，再填入 ARG1；
3. 运算启动后，BSY=1；
4. 运算完成，BSY=0。

注意：1. 当 BSY=1 时，再次填入 ARG1 会被忽略；

2. 除法运算时 ARG2 不能为 0，填 0 会触发 CALF=1；

3. 除法运算时 RES1 为 ARG1 除 ARG2 的整数部分，RES2 为余数。

12.3 ME 寄存器

本节对 ME 寄存器进行了详细介绍。
可按字、半字或字节模式写入外设寄存器。

12.3.1 ME 控制寄存器（ME_CR）

偏移地址：0x00
复位值：0x0000 0000

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

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

Bit 31 **BSY**：判忙标志位
0：空闲或运算完成
1：运算中

Bit 30 **CALF**：运算错误标志位
0：运算正常
1：运算错误，除数为 0

Bit 29:5 保留，必须保持为复位值。

Bit 4 **SN**：符号选择
0：无符号
1：有符号

Bit 3:1 保留，必须保持为复位值。

Bit 0 **FUN**：函数运算选择
0：DIV
1：SQRT

12.3.2 ME ARG1/RES1 寄存器（ARG1/RES1）

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
ARG1/RES1[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
ARG1/RES1[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **ARG1/RES1[31:0]：**写入 ARG1，填入启动运算

读出 RES1，仅在运算完成（BSY=0）时才有效

12.3.3 ME ARG2/RES2 寄存器（ARG2/RES2）

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
ARG2/RES2[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
ARG2/RES2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **ARG2/RES2[31:0]：**写入 ARG2

读出 RES2，仅在运算完成（BSY=0）时才有效

13 模数转换器（ADC）

13.1 ADC 简介

本部分介绍多达 2 个 ADC 的实现。

每个 ADC 由 12 位逐次逼近模数转换器组成。

每个 ADC 的复用通道最多达 17 条。各种不同通道的 A/D 转换可在单次、连续、扫描或间断采样模式下进行。ADC 的结果存储在一个左对齐或右对齐的 16 位数据寄存器中。

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

ADC 的输入时钟不得超过 60.8MHz，它是由 SYSCLK 或 PLL2CLK 产生。详见章节：ADC 时钟。

13.2 ADC 主要特征

- 高性能特性
  - 多达 2 个 12 位分辨率 ADC
    ADC1 连接至 12 个外部通道和 5 个内部通道
    ADC2 连接至 12 个外部通道和 4 个内部通道
  - ADC 转换时间与 AHB 总线时钟频率无关
  - 自校准
  - 可独立设置各通道采样时间
  - 多达四条注入通道（对规则通道或注入通道的模拟输入分配完全可配置）
  - 数据对齐以保持内置数据一致性
  - 4 个专用注入数据寄存器供注入通道使用
  - 8 个专用规则数据寄存器供规则通道使用
- 每个 ADC 都有外部模拟输入通道
  - 专用 GPIO 焊盘有最多达 3 条快速通道
  - 专用 GPIO 焊盘有最多达 9 条慢速通道
- 此外，还有几条内部专用通道
  - 内部基准电压（V_{REFINT}），连接到 ADC1
  - 内部温度传感器（V_{TPS}），连接到 ADC2
  - 参考电压（V_{REF}/2）监测通道，连接到 ADC1
  - OPAMP1/4 内部输出，连接到 ADC1
  - OPAMP3/2 内部输出，连接到 ADC2
- 可通过以下方式启动转换过程：
  - 通过软件启动常规转换和注入转换
  - 通过极性可配置的硬件触发器（内部定时器事件或 GPIO 输入事件）来启动常规转换和注入转换
- ADC1 和 ADC2 的双重 ADC 模式
- 转换模式
  - 每个 ADC 均可转换单条通道，也可扫描一系列通道
  - 单次模式会在每次触发时对选定的输入执行一次转换
  - 连续模式可连续转换选定的输入
  - 间断采样模式

- 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
- ADC 输入范围：$V_{SSA} \le V_{IN} \le V_{REF}$

请参考设备数据手册中的 OPAMP 电气特性部分，了解在转换 OPAMP 输出电压时应应用的 ADC 采样时间值。

13.3 ADC 功能描述

13.3.1 ADC 框图

下图为 ADC 框图。

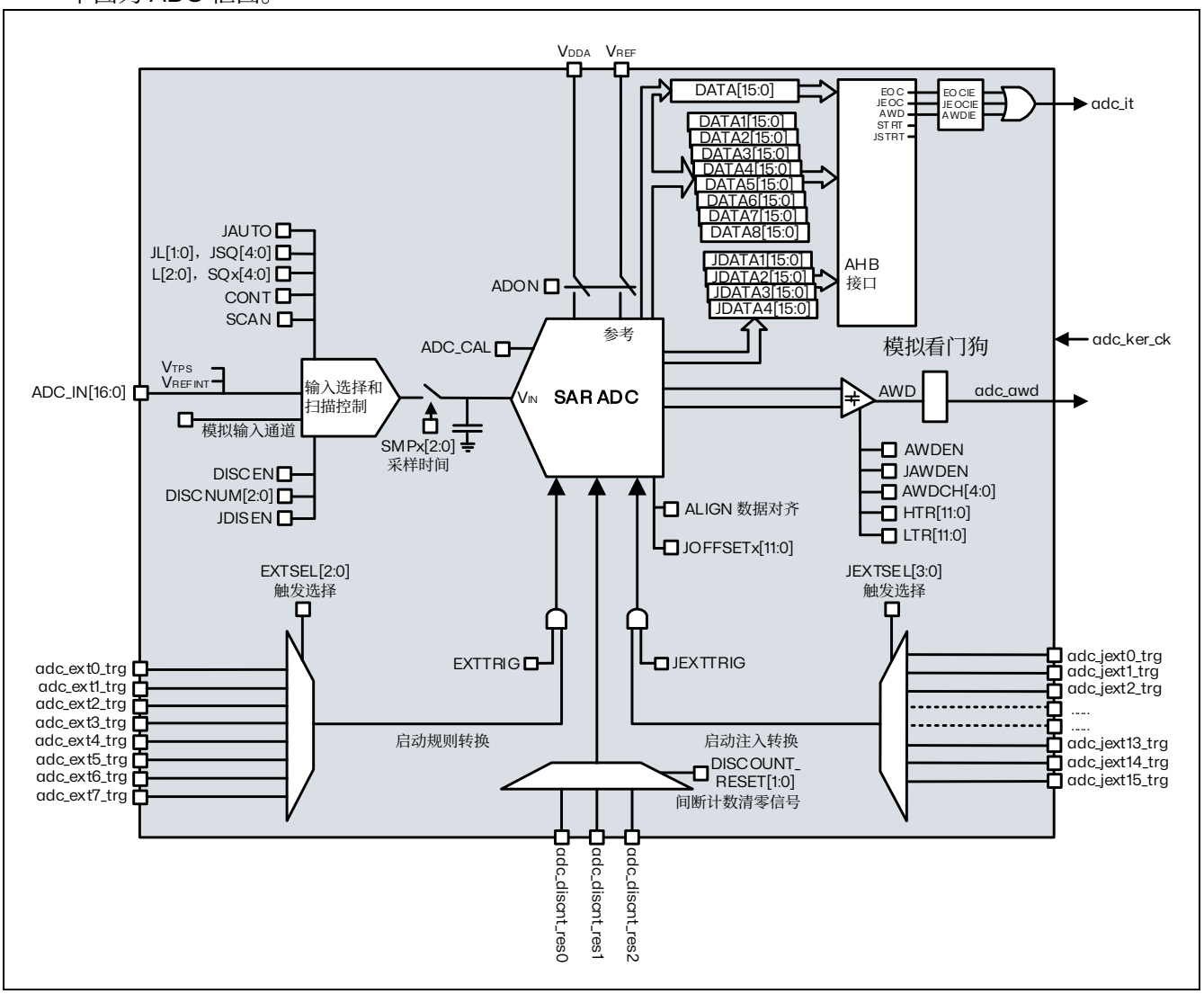


图 13.1 ADC 框图

13.3.2 ADC 引脚和内部信号

表 13.1 ADC 内部输入/输出信号

内部信号名称	信号类型	说明
adc_ext_trg[7:0]	输入	共有多达 8 个外部触发输入用于常规转换（可连接至片上定时器）。
adc_jext_trg[15:0]	输入	共有多达 16 个外部触发输入用于注入转换（可连接至片上定时器）。
adc_discnt_res[2:0]	输入	共有多达 3 个外部触发输入用于间断计数清零（可连接至片上定时器）。
adc_ker_ck	输入	ADC 内核时钟
adc_awd	输出	内部模拟看门狗输出信号，连接至片上定时器。
adc_it	输出	ADC 中断
V_{TPS}	模拟输入	内部温度传感器的输出电压
V_{REFINT}	模拟输入	内部基准电压的输出电压
V_{REF}	模拟输入	参考电压

表 13.2 ADC 输入/输出引脚

引脚名称	信号类型	说明
VREF	输入，模拟参考	ADC 使用的参考电压
VDDA	输入，模拟电源	等效于 V_{DD} 的模拟电源
VSSA	输入，模拟电源地	等效于 V_{SS} 的模拟电源地
ADCx_IN[16:0]	模拟输入信号	17 个模拟输入通道

13.3.3 ADC 时钟

ADC 时钟可以是名为 adc_ker_ck 的特定时钟源。

两个 ADC 的输入时钟相同，可从两个不同的时钟源中选择（请参见下图：ADC 时钟方案）：

– SYSCLK 系统时钟

– PLL2CLK 时钟

选择完输入时钟后，再经 ADCPRE 预分频器选择预分频可得到最终输入的时钟。该时钟可在 RCC 中配置（有关如何生成 ADC 时钟（adc_ker_ck）专用时钟的更多信息，请参见 RCC 部分）。

注：ADC 的输入时钟不得超过 60.8MHz。

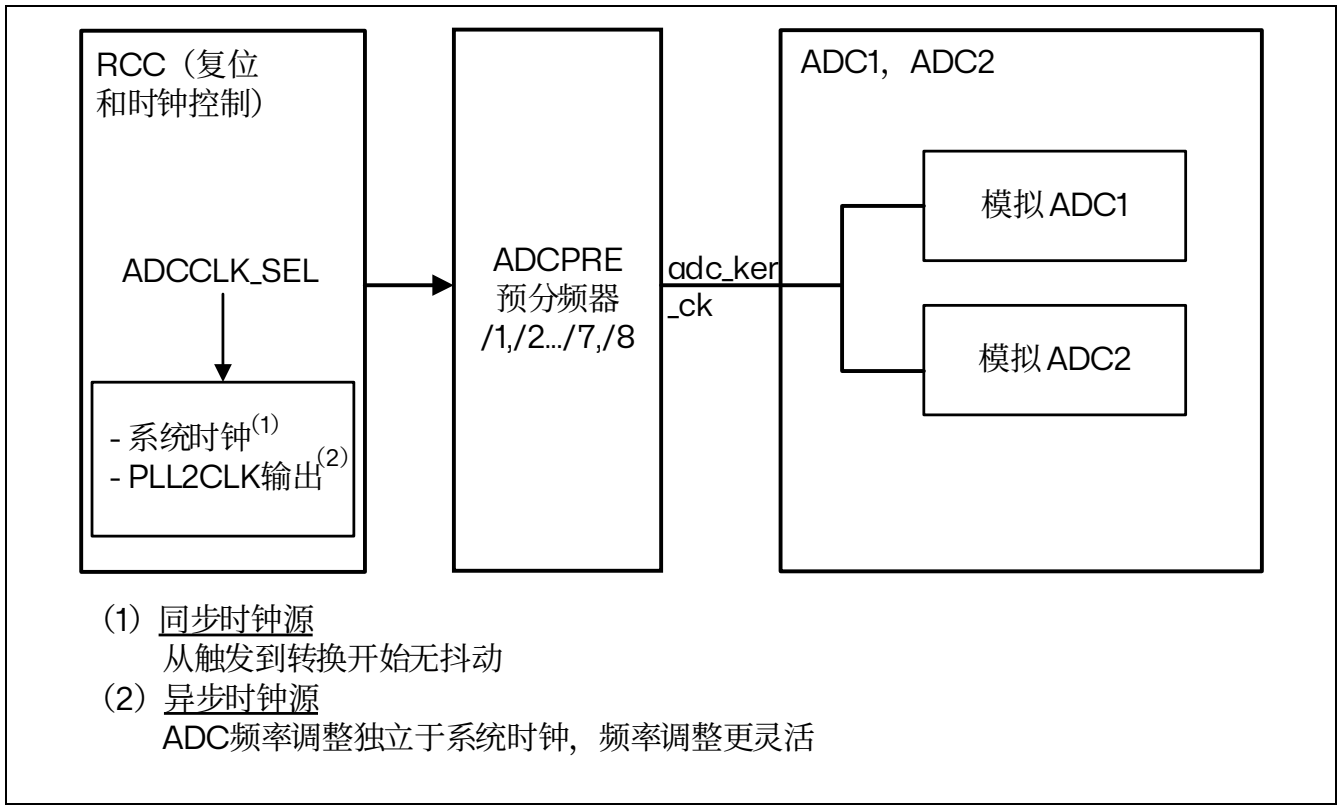


图 13.2 ADC 时钟方案

13.3.4 ADC 通道连接

ADC1、ADC2 的通道如下列表所示。

通道编号，即通道 x 对应设备数据手册中定义的 ADCy_INx 引脚。

注：若 OPAMP *工作在低功耗模式（*OPAMPx_CSR *寄存器* OPALPM=1*）下，对内部* OPAMP *通道（*ADC *通道* 13/14*）采样时，采样时间至少使用* 17.5 *周期及以上档位（速度*≤2Msps *采样）。*

注：当需要使用以下两个 ADC *通道需要先修改* OPAMPx_CSR *寄存器* VM_SEL[1:0]=01 *来避免因模拟功能复用而导致的* ADC *精度影响。*

- PB0：ADC2_IN9 & OPAMP2_N1
- PB9：ADC1_IN11 & OPAMP1_N1

表 13.3 ADC1 通道

通道编号	模拟信号来源	通道速度
通道 0	VSSA	FAST_4M
通道 1	PC7	SLOW_2M
通道 2	PC9	SLOW_2M
通道 3	PC6	SLOW_2M
通道 4	PA7	SLOW_2M
通道 5	PA8	SLOW_2M
通道 6	PA9	FAST_4M
通道 7	PA10	FAST_4M
通道 8	PA11	FAST_4M
通道 9	PA4	SLOW_2M
通道 10	PB8	SLOW_2M
通道 11	PB9	SLOW_2M
通道 12	PA3	SLOW_2M
通道 13	OPAMP1_OUT	FAST_4M
通道 14	OPAMP4_OUT	FAST_4M
通道 15	VREF/2	SLOW_1M
通道 16	VREFINT	SLOW_1M

表 13.4 ADC2 通道

通道编号	模拟信号来源	通道速度
通道 0	VSSA	FAST_4M
通道 1	PB2	SLOW_2M
通道 2	PB3	SLOW_2M
通道 3	PB4	SLOW_2M
通道 4	PB5	SLOW_2M
通道 5	PB1	SLOW_2M
通道 6	PC2	FAST_4M
通道 7	PA6	FAST_4M
通道 8	PB11	FAST_4M
通道 9	PB0	SLOW_2M
通道 10	PC3	SLOW_2M
通道 11	PB12	SLOW_2M
通道 12	PB13	SLOW_2M
通道 13	OPAMP3_OUT	FAST_4M
通道 14	OPAMP2_OUT	FAST_4M
通道 15	保留	SLOW_1M
通道 16	TPS	SLOW_1M

*注：1. 如果需要采样*VREFINT/TPS *通道，需要置位对应*ADC1/2_CR2 *寄存器的*TSVREFE *位*

2. *如果需要采样*TPS *通道，采样时间要设定为*640.5T *（选择最高档位）*

13.3.5 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准精度误差。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_CR2 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_CAL 中。

*注：建议在每次上电后执行一次校准。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。*

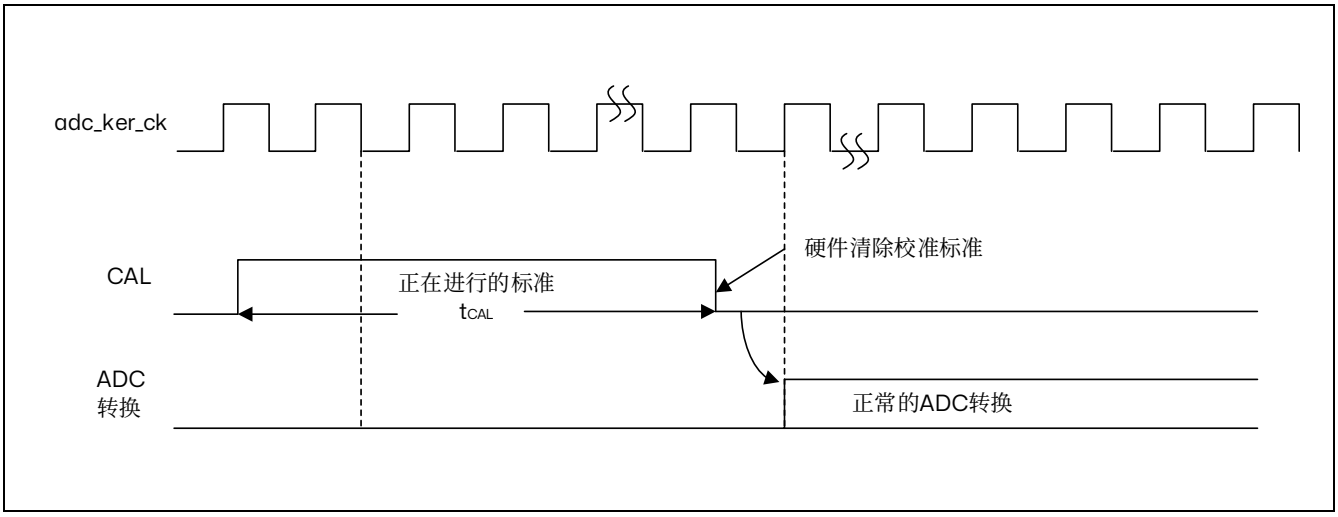


图 13.3 校准时序图

13.3.6 ADC 开关控制

可通过将 ADC_CR2 寄存器中的 ADON 位置 1 来为 ADC 供电。首次将 ADON 位置 1 时，会将 ADC 从掉电模式中唤醒。

SWSTART 或 JSWSTART 位置 1 时，启动 AD 转换。

可通过将 ADON 位清零来停止转换并使 ADC 进入掉电模式。在此模式下，ADC 几乎不耗电（只有几 μA）。

13.3.7 通道选择

RX32H710 每个 ADC 有多路通道。可以将转换分为两组：规则转换和注入转换。每个组包含一个转换序列，该序列可按任意顺序在任意通道上完成。例如，可按以下顺序对序列进行转换：ADC_IN3、ADC_IN8、ADC_IN2、ADC_IN2、ADC_IN0、ADC_IN2、ADC_IN2、ADC_IN15。

- **规则组：**由多达 8 个转换组成。规则通道和它们的转换顺序在 ADC_SQRx 寄存器中选择。规则组中转换的总数应写入 ADC_SQR1 寄存器的 L[2:0]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_JSQR 寄存器中选择。注入组里的转换总数目应写入 ADC_JSQR 寄存器的 JL[1:0]位中。

如果 ADC_SQRx 或 ADC_JSQR 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。

13.3.8 单次转换模式

在单次转换模式下，ADC 执行一次转换。CONT 位为 0 时，可通过以下方式启动此模式：

- 将 ADC_CR2 寄存器中的 SWSTART 位置 1（仅适用于规则通道）
- 将 JSWSTART 位置 1（适用于注入通道）
- 外部触发（适用于规则通道或注入通道）

完成所选通道的转换之后：

- 如果转换了规则通道：
  - 转换数据存储在 16 位 ADC_DR 寄存器中
  - EOC（转换结束）标志置 1
  - EOCIE 位置 1 时将产生中断
- 如果转换了注入通道：
  - 转换数据存储在 16 位 ADC_JDR1 寄存器中
  - JEOC（注入转换结束）标志置 1
  - JEOCIE 位置 1 时将产生中断

注： ADC_CR4 *寄存器的* EOC_OPT *位可选择控制* EOC *（转换结束）标志*
OPT = 0b*：规则或注入通道组转换结束时设置* ADC_SR *寄存器的* EOC
OPT = 1b *（默认）：规则通道组转换结束时设置* ADC_SR *寄存器的* EOC

然后 ADC 停止。

13.3.9 连续转换模式

在连续转换模式下，ADC 结束一个转换后立即启动一个新的转换。CONT 位为 1 时，可通过外部触发或将 ADC_CR2 寄存器中的 SWSTRT 位置 1 来启动此模式（仅适用于规则通道）。

每次转换之后：

- 如果转换了规则通道组：
  - 上次转换的数据存储在 16 位 ADC_DR 寄存器中
  - EOC（转换结束）标志置 1
  - EOCIE 位置 1 时将产生中断
- 如果转换了注入通道：
  - 转换数据存储在 16 位 ADC_JDR1 寄存器中
  - JEOC（注入转换结束）标志置 1
  - JEOCIE 位置 1 时将产生中断

注： ADC_CR4 *寄存器的* EOC_OPT *位可选择控制* EOC *（转换结束）标志*
OPT = 0b*：规则或注入通道组转换结束时设置* ADC_SR *寄存器的* EOC
OPT = 1b *（默认）：规则通道组转换结束时设置* ADC_SR *寄存器的* EOC

13.3.10 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 15 个时钟周期后，EOC 标志被设置，16 位 ADC 数据寄存器包含转换的结果。

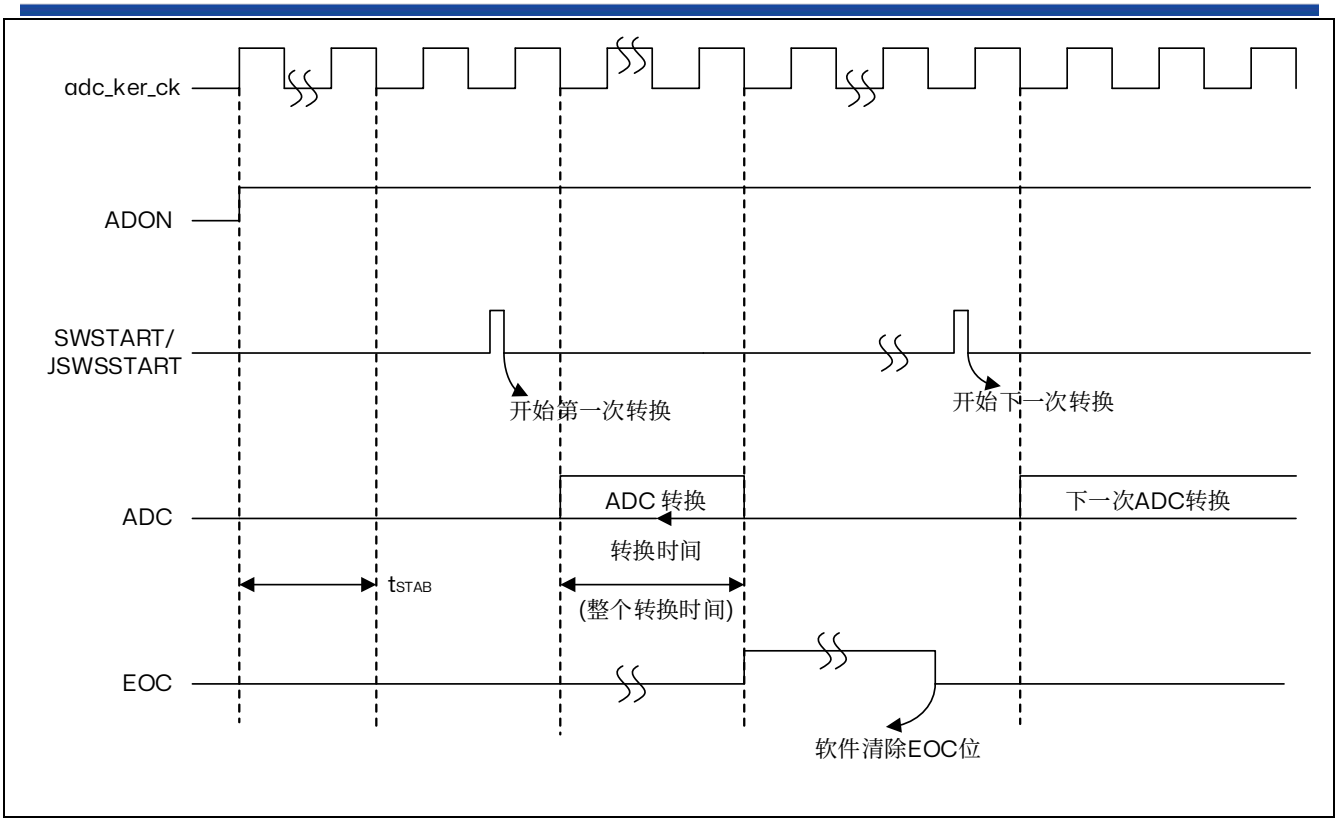


图 13.4 ADC 转换时序图

13.3.11 模拟看门狗

如果被 ADC 转换的模拟电压低于低阀值或高于高阀值，AWD 模拟看门狗状态位被设置。阀值位于 ADC_HTR 和 ADC_LTR 寄存器的最低 12 个有效位中。通过设置 ADC_CR1 寄存器的 AWDIE 位以允许产生相应中断。

阀值独立于由 ADC_CR2 寄存器上的 ALIGN 位选择的数据对齐模式。比较是在对齐之前完成的。

通过配置 ADC_CR1 寄存器，模拟看门狗可以作用于 1 个或多个通道。如果设置模拟看门狗作用于单一通道，可以使能 ADC_HTR 和 ADC_LTR 寄存器中的 HTREN 和 LTREN 位，并通过 HTRCH[5:0]和 LTRCH[5:0]位域来选择作用于哪一个通道。

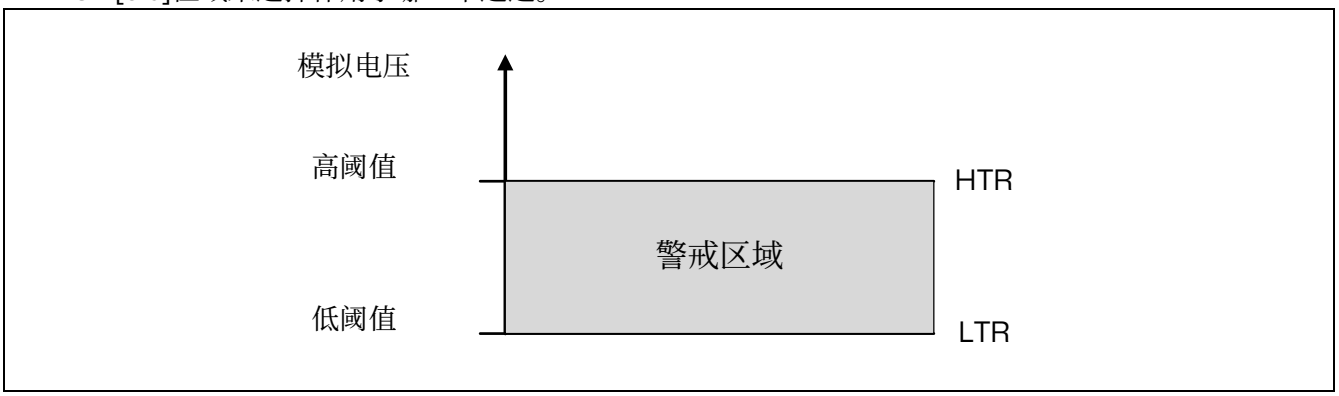


图 13.5 模拟看门狗警戒区

表 13.5 模拟看门狗通道选择

模拟看门狗警戒通道	ADC_CR1 寄存器控制位		
	AWDSGL 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 (1) 注入通道	1	0	1
单一的 (1) 规则通道	1	1	0
单一的 (1) 注入或规则通道	1	1	1

1. 由 AWDCH[4:0]位选择

13.3.12 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_CR1 寄存器的 SCAN 位来选择。一旦这个位被设置，ADC 扫描所有被 ADC_SQRx 寄存器（对规则通道）或 ADC_JSQR（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 CONT 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。

规则通道转换的对应数据被存储在 16 位 ADC_DRx（x = 1~8）规则数据寄存器 x 中

注入通道转换的对应数据被存储在 16 位 ADC_JDRx（x = 1~4）注入数据寄存器 x 中

13.3.13 注入通道管理

触发注入

清除 ADC_CR1 寄存器的 JAUTO 位，并且设置 SCAN 位，即可使用触发注入功能。

1. 通过外部触发或将 ADC_CR2 寄存器中的 SWSTART 位置 1 来启动规则通道组转换。
2. 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。
3. 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（前一次的规则序列已经完成），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。

注：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 30 个 ADC 时钟周期（即 2 个具有 2.5 个时钟间隔采样时间的转换），触发之间最小的间隔必须是 31 个 ADC 时钟周期。

自动注入

如果设置了 JAUTO 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_SQRx 和 ADC_JSQR 寄存器中设置的多至 12 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 JAUTO 位外还设置了 CONT 位，规则通道至注入通道的转换序列被连续执行。

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 2 个 ADC 时钟间隔的延迟。

注：不可能同时使用自动注入和间断模式。

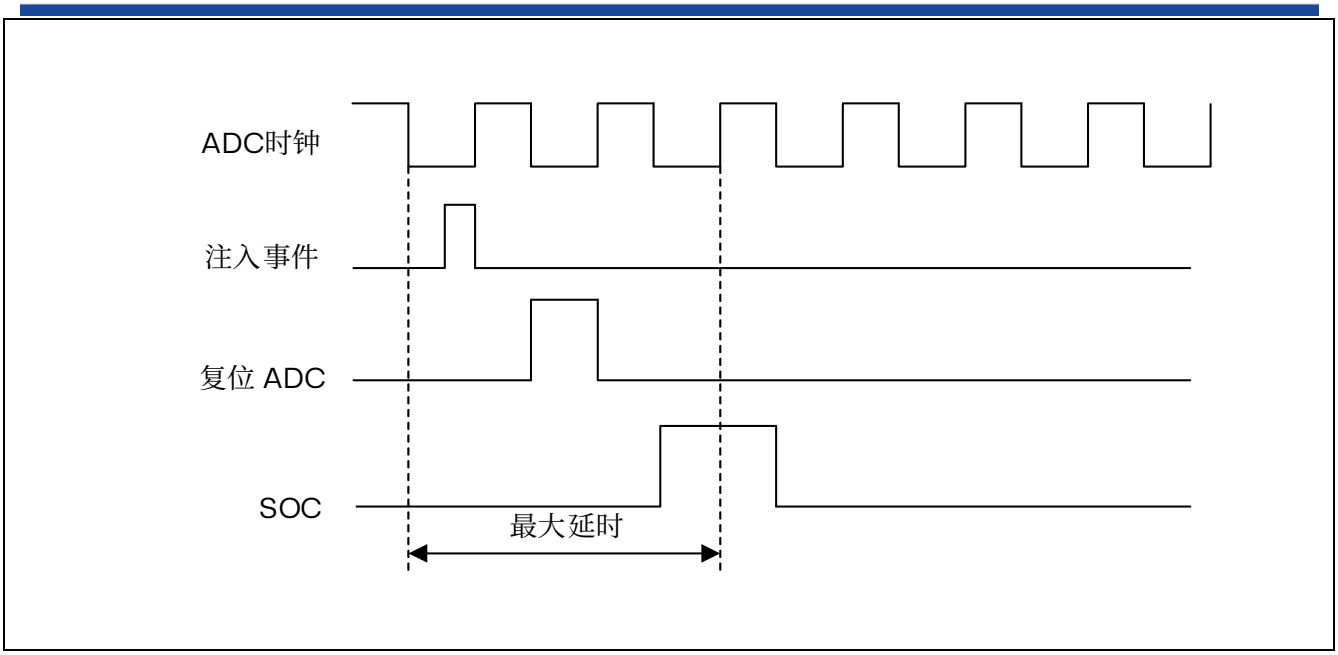


图 13.6 注入转换延迟

13.3.14 间断模式

规则组

此模式通过设置 ADC_CR1 寄存器上的 DISCEN 位激活。它可以用来执行一个短序列的 n 次转换（n<=8），此转换是 ADC_SQRx 寄存器所选择的转换序列的一部分。数值 n 由 ADC_CR1 寄存器的 DISCNUM[2:0]位给出。

一个外部触发信号可以启动 ADC_SQRx 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_SQR1 寄存器的 L[2:0]定义。

举例：

n = 3，被转换通道 = 0、1、2、3、6、7、9、10

第一次触发：转换的序列为 0、1、2

第二次触发：转换的序列为 3、6、7

第三次触发：转换的序列为 9、10，并产生 EOC 事件

第四次触发：转换的序列为 0、1、2

注：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（默认）。
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：为避免规则组因触发次数不足间断子序列个数，导致后续序列的触发受到错位影响，可以先将 ADC_CR1 寄存器 DISCEN=0 关闭规则组间断模式，再将 ADC_CR1 寄存器 DISCEN=1 重新打开来复位规则组间断计数以从头开始转换。

注入组

此模式通过设置 ADC_CR1 寄存器的 JDISCEN 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_JSQR 寄存器中选择的序列。

一个外部触发信号可以启动 ADC_JSQR 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_JSQR 寄存器的 JL[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 EOC 和 JEOC 事件

第四次触发：通道 1 被转换

注：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（默认）。在上述例子中，第四个触发重新转换第1 个注入通道1。

注：为避免注入组因触发次数不足间断子序列个数，导致后续序列的触发受到错位影响，可以通过 ADC_CR2 寄存器的DISCOUNT_RESET[1:0]来选择间断计数清零信号以从头开始转换。间断计数清零详情参见下一章节：注入组序列间断计数清零。

注：不能同时使用自动注入和间断模式。

不得同时为规则组和注入组设置间断模式。只能针对一个组使能间断模式。

13.3.15 注入组序列间断计数清零

清零信号可以由定时器事件触发。ADC_CR2 寄存器的 DISCOUNT_RESET[1:0]控制位允许应用程序选择所有可能的事件中的某一个，可以触发注入组序列间断计数清零。

在实际应用中，当 ADC 触发次数不足间断子序列个数，可能导致后续序列的触发会受到错位影响。此时通过间断计数清零可以使间断模式从新开始计数。

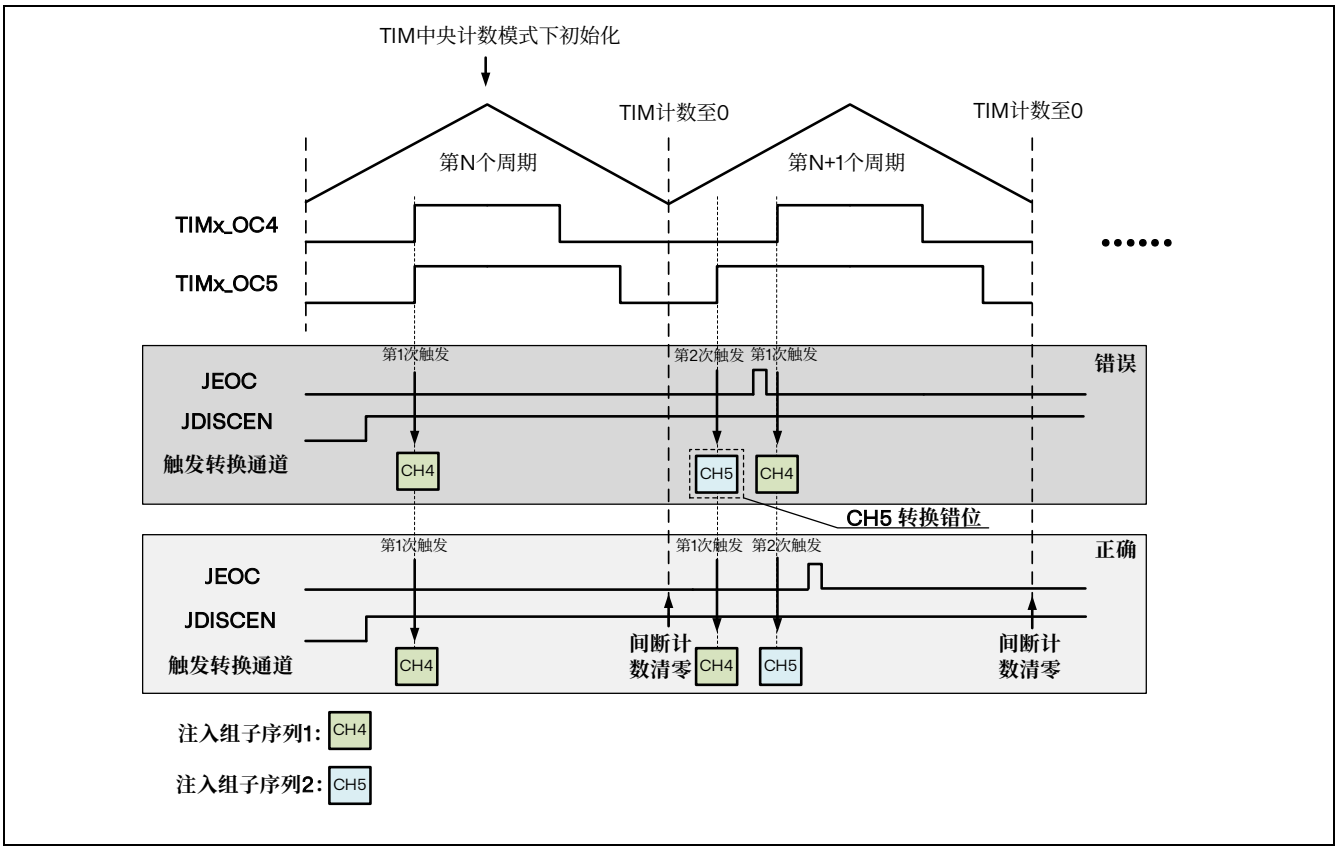


图 13.7 使用间断计数清零来复位触发转换错位

表 13.6 ADC1/2 间断计数清零信号

名称	触发源	连接类型	DISCOUNT_RESET [1:0]
adc_discnt_res0	TIM1_TRGO 事件	片上定时器的内部信号	01
adc_discnt_res1	TIM1_TRGO2 事件		10
adc_discnt_res2	TIM8_TRGO 事件		11

13.3.16 数据对齐

ADC_CR2 寄存器中的 ALIGN 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。

注入组通道转换的数据值已经减去了在 ADC_JOFRx 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。

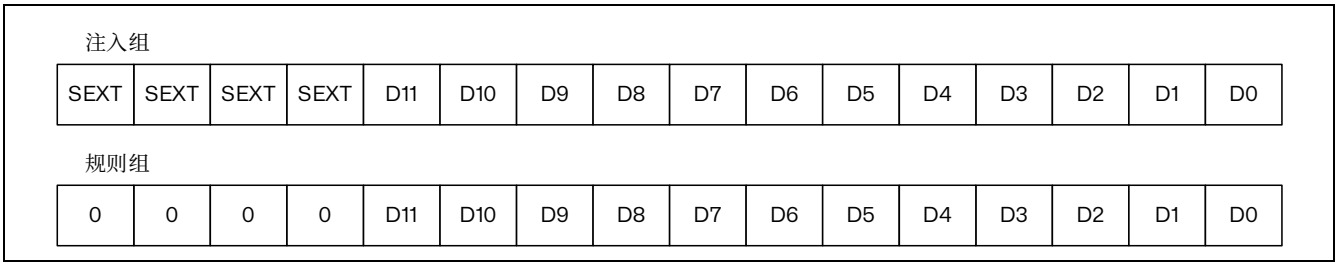


图 13.8 数据右对齐

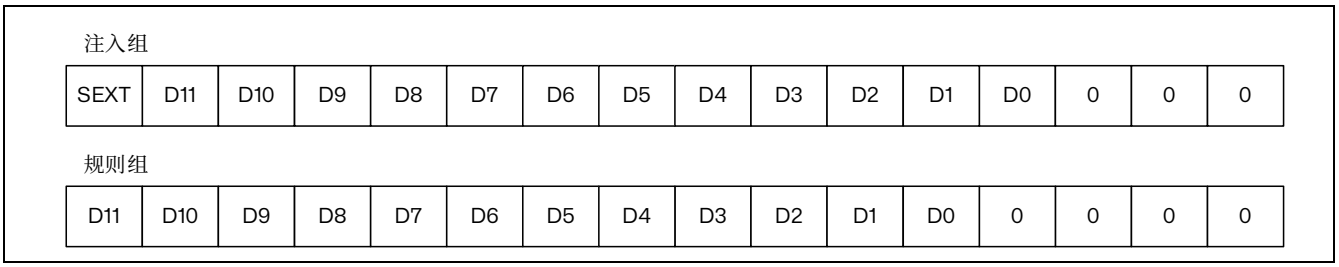


图 13.9 数据左对齐

13.3.17 可编程的通道采样时间

ADC 使用若干个 adc_ker_ck 周期对输入电压采样，采样周期数目可以通过 ADC_SMPR1 和 ADC_SMPR2 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} = 采样时间 + 12.5 个周期

例如：

当 adc_ker_ck = 60MHz，采样时间为 2.5 周期

T_{CONV} = 2.5 + 12.5 = 15 个周期 = 250ns

13.3.18 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。EXTSEL[2:0]和 JEXTSEL[3:0]控制位允许应用程序选择所有可能的事件中的某一个，可以触发规则和注入组的采样

注：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 13.7 ADC1/2 规则通道的外部触发信号

名称	触发源	连接类型	EXTSEL[2:0]
adc_ext_trg0	TIM8_TRGO2 事件	片上定时器的内部信号	000
adc_ext_trg1	TIM6_TRGO 事件		001
adc_ext_trg2	TIM7_TRGO 事件		010
adc_ext_trg3	TIM1_TRGO2 事件		011
adc_ext_trg4	TIM2_TRGO 事件		100
adc_ext_trg5	TIM3_TRGO 事件		101
adc_ext_trg6	EXTI 线 13（ADC1）	外部引脚	110
	EXTI 线 14（ADC2）		
adc_ext_trg7	SWSTART	软件控制位	111

表 13.8 ADC1/2 注入通道的外部触发信号

名称	触发源	连接类型	JEXTSEL[3:0]
adc_jext_trg0	TIM8_TRGO 事件	片上定时器的内部信号	0000
adc_jext_trg1	TIM8_OC4 事件		0001
adc_jext_trg2	TIM2_TRGO 事件		0010
adc_jext_trg3	TIM8_OC5 事件		0011
adc_jext_trg4	TIM3_OC1 事件		0100
adc_jext_trg5	TIM3_TRGO 事件		0101
adc_jext_trg6	EXTI 线 2（ADC1）	外部引脚	0110
	EXTI 线 3（ADC2）		
adc_jext_trg7	JSWSTART	软件控制位	0111
adc_jext_trg8	TIM8_TRGO2 事件	片上定时器的内部信号	1000
adc_jext_trg9	TIM6_TRGO 事件		1001
adc_jext_trg10	TIM7_TRGO 事件		1010
adc_jext_trg11	TIM8_OC4/OC6 事件		1011
adc_jext_trg12	TIM8_OC4/OC5 事件		1100
adc_jext_trg13	TIM1_TRGO 事件		1101
adc_jext_trg14	TIM15_TRGO 事件		1110
adc_jext_trg15	TIM1_TRGO2 事件		1111

软件触发事件可以通过对寄存器 ADC_CR2 的 SWSTART 或 JSWSTART 位置 1 产生。
规则组的转换可以被注入触发打断。

13.4 双 ADC 模式

在双 ADC 模式里，根据 ADC1_CR1 寄存器中 CR1_DUALMOD[3:0]位所选的模式，转换的启动可以是 ADC1 主和 ADC2 从的交替触发或同步触发。

注：在双 ADC 模式里，当转换配置成由外部事件触发时，用户必须将其设置成仅触发主 ADC，从 ADC 设置成软件触发，这样可以防止意外的触发从转换。但是，主和从 ADC 的外部触发必须同时被激活。

共有 6 种模式：

- 同步注入模式
- 同步规则模式
- 快速交叉模式
- 慢速交叉模式
- 交替触发模式
- 独立模式

还有可以用下列方式组合使用上面的模式：

- 同步注入模式 + 同步规则模式
- 同步规则模式 + 交替触发模式
- 同步注入模式 + 交叉模式

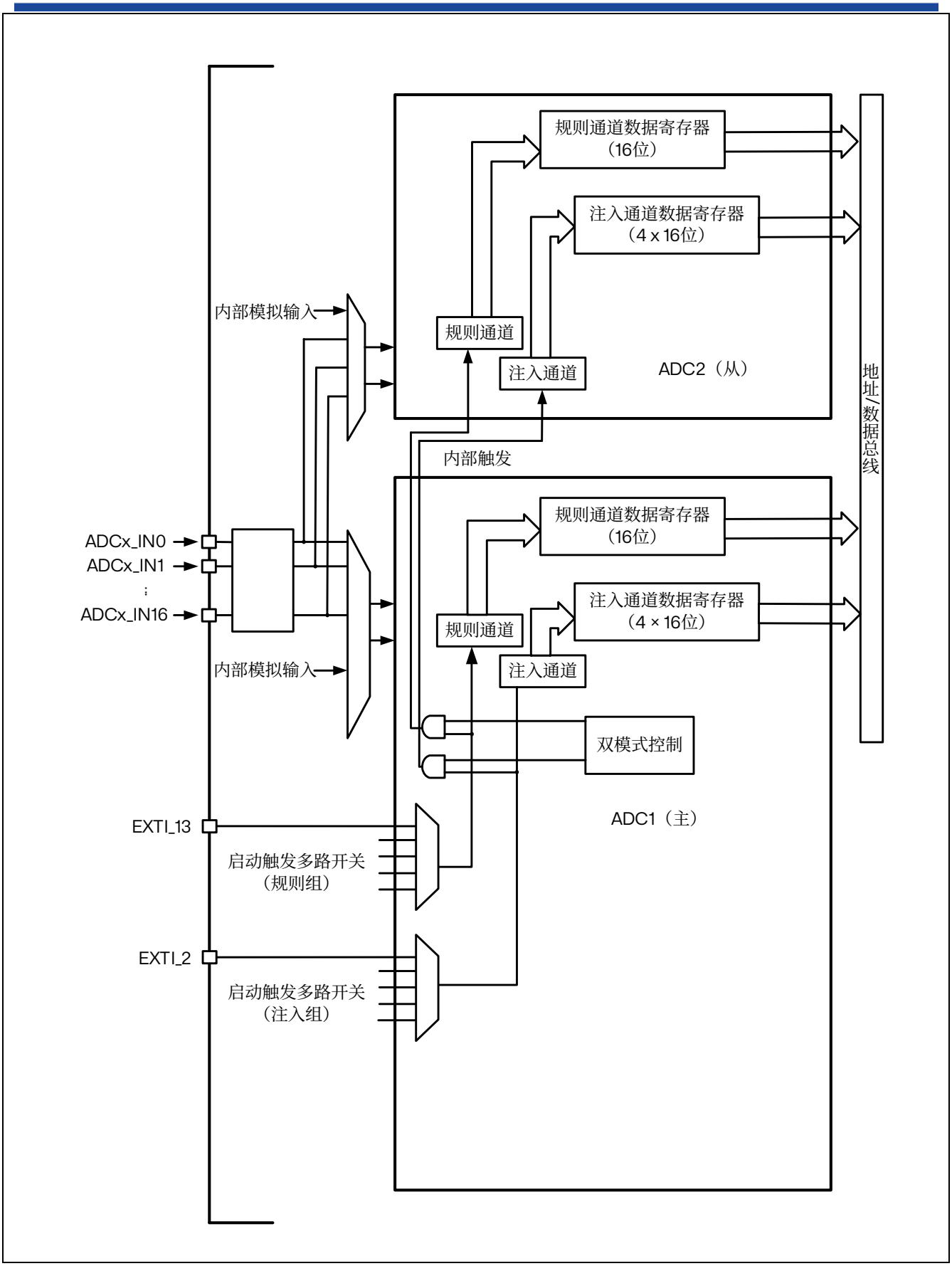


图 13.10 双 ADC 框图

13.4.1 同步注入模式

此模式转换一个注入通道组。外部触发来自 ADC1 的注入组多路开关（由 ADC1_CR2 寄存器的 JEXTSEL[3:0]选择），它同时给 ADC2 提供同步触发。

注：不要在 2 个 ADC 上转换相同的通道（两个 ADC 在同一个通道上的采样时间不能重叠）。

在 ADC1 或 ADC2 的转换结束时：

- 转换的数据存储在每个 ADC 接口的 ADC_JDRx 寄存器中。
- 当所有 ADC1/ADC2 注入通道都被转换时，产生 JEOC 中断（若任一 ADC 接口开放了中断）。

注：在同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

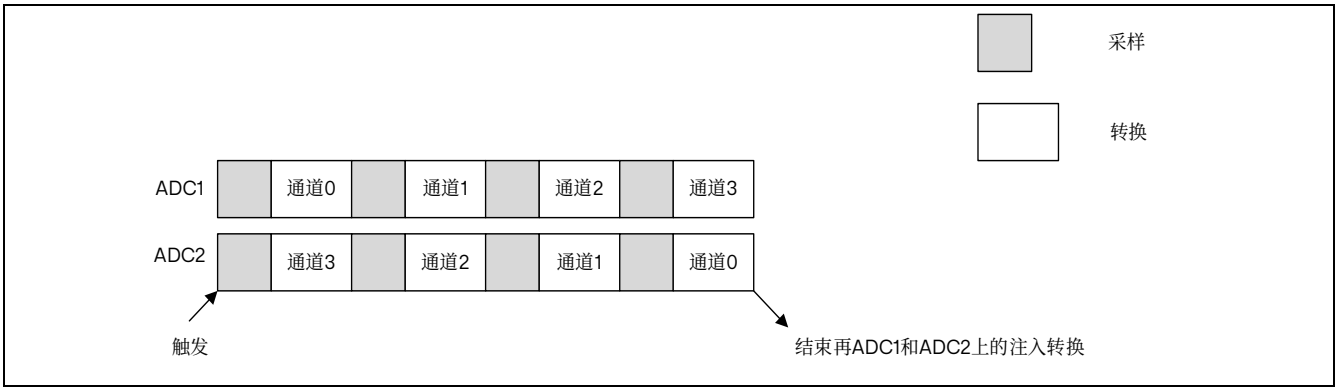


图 13.11 4 个通道的同步注入模式

13.4.2 同步规则模式

此模式在规则通道组上执行。外部触发来自 ADC1 的规则组多路开关（由 ADC1_CR2 寄存器的 EXTSEL[3:0]选择），它同时给 ADC2 提供同步触发。

注：不要在 2 个 ADC 上转换相同的通道（两个 ADC 在同一个通道上的采样时间不能重叠）。

在 ADC1 或 ADC2 的转换结束时：

- 转换的数据存储在每个 ADC 接口的 ADC_DRx 寄存器中。
- 当所有 ADC1/ADC2 规则通道都被转换完时，产生 EOC 中断（若任一 ADC 接口开放了中断）。

注：在同步规则模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

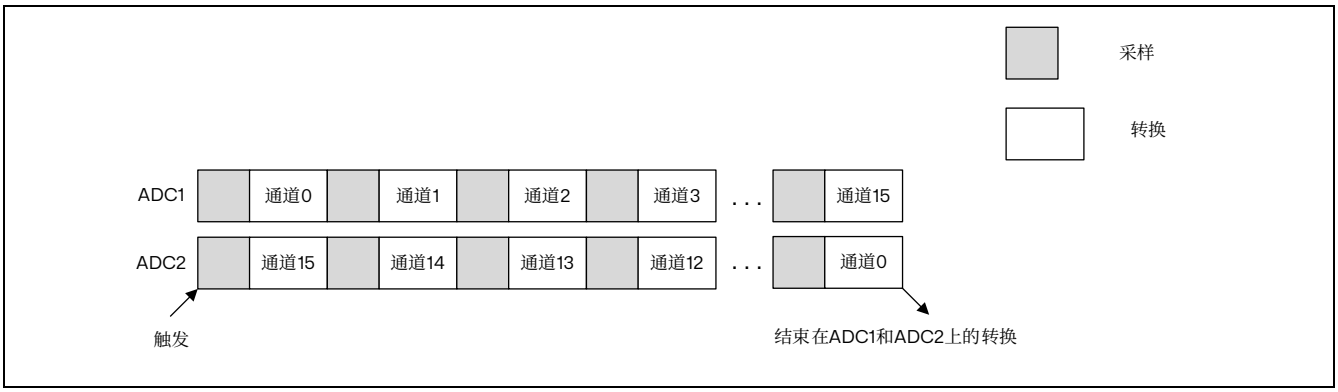


图 13.12 16 个通道的同步规则模式

13.4.3 快速交叉模式

此模式只适用于规则通道组。外部触发来自 ADC1 的规则通道多路开关。外部触发产生后：

- ADC2 立即启动
- ADC1 在延迟 7 个 ADC 时钟周期后启动

如果同时设置了 ADC1 和 ADC2 的 CONT 位，所选的两个 ADC 规则通道将被连续地转换。

注：最大允许采样时间<7 个adc_ker_ck 周期，避免 ADC1 和ADC2 转换相同通道时发生两个采样周期的重叠。

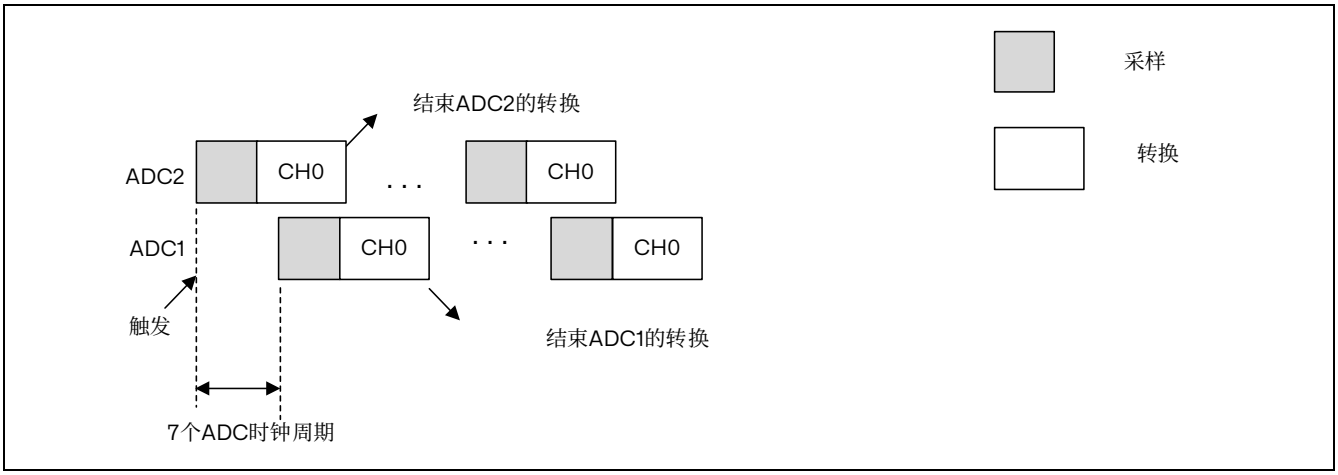


图 13.13 在 1 个通道上连续转换模式下的快速交叉模式

13.4.4 慢速交叉模式

此模式只适用于规则通道组。外部触发来自 ADC1 的规则通道多路开关。外部触发产生后：

- ADC2 立即启动
- ADC1 在延迟 14 个 ADC 时钟周期后启动
- 在延迟第二次 14 个 ADC 周期后 ADC2 再次启动，如此循环。

注：最大允许采样时间<14 个ADCCLK 周期，以避免和下个转换重叠。

在 28 个 ADC 时钟周期后自动启动新的 ADC2 转换。
在这个模式下不能设置 CONT 位，因为它将连续转换所选择的规则通道。
注意：应用程序必须确保当使用交叉模式时，不能有注入通道的外部触发产生。

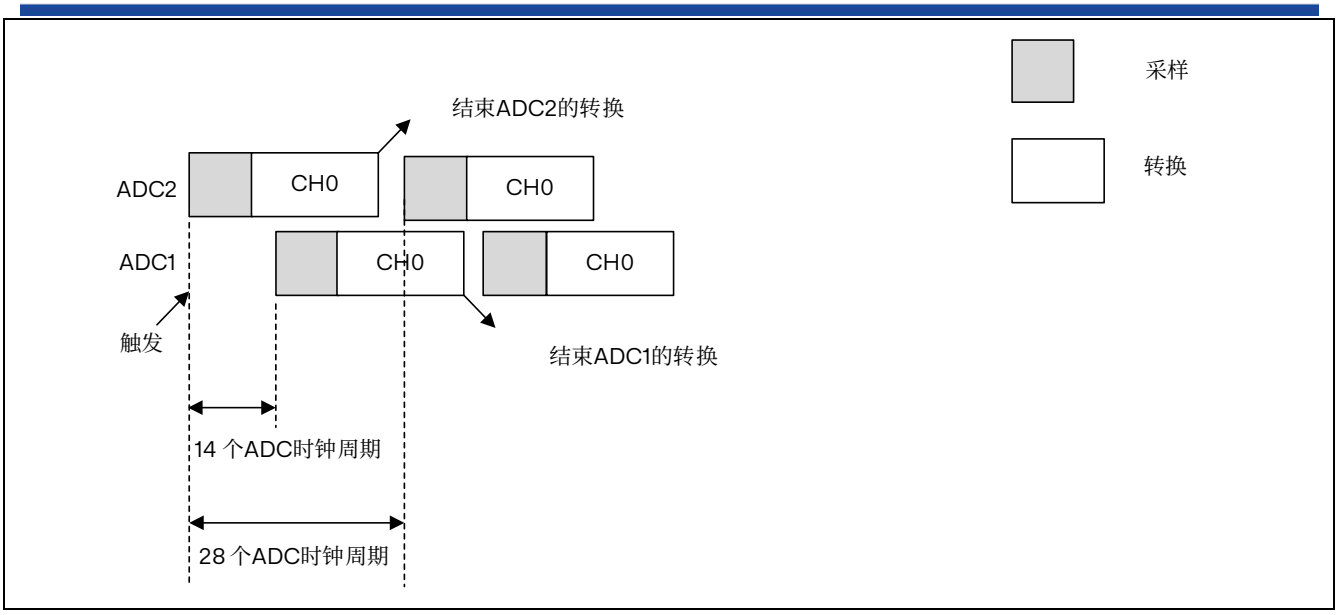


图 13.14 在 1 个通道上的慢速交叉模式

13.4.5 交替触发模式

此模式只适用于注入通道组。外部触发源来自 ADC1 的注入通道多路开关。

- 当第一个触发产生时，ADC1 上的所有注入组通道被转换。
- 当第二个触发到达时，ADC2 上的所有注入组通道被转换。
- 如此循环.....

如果允许产生 JEOC 中断，在所有 ADC1 注入组通道转换后产生一个 JEOC 中断。

如果允许产生 JEOC 中断，在所有 ADC2 注入组通道转换后产生一个 JEOC 中断。

当所有注入组通道都转换完后，如果又有另一个外部触发，交替触发处理从转换 ADC1 注入组通道重新开始。

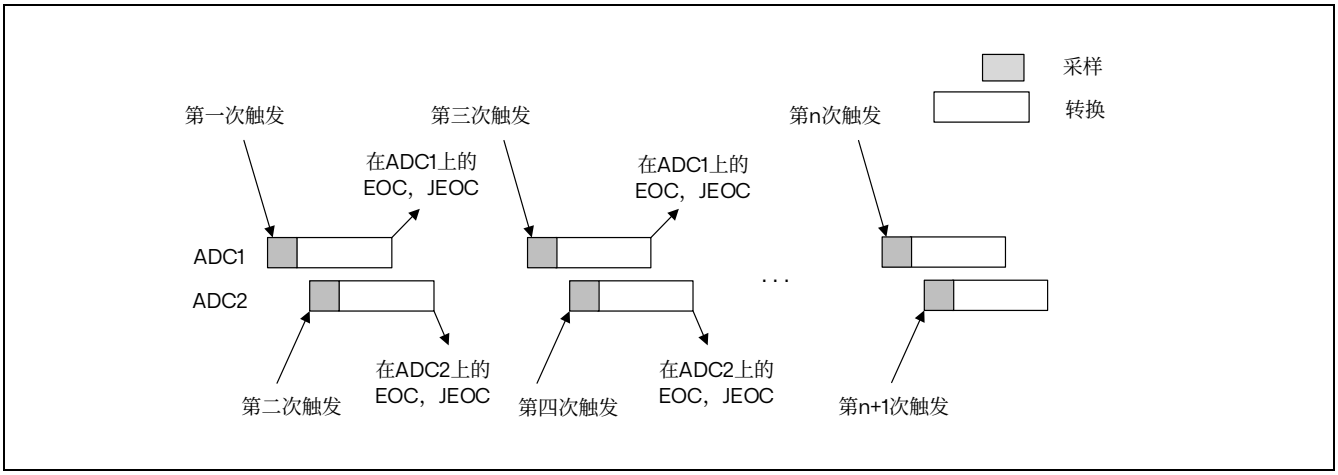


图 13.15 交替触发：每个 ADC1 的注入通道组

如果 ADC1 和 ADC2 上同时使用了注入间断模式：

- 当第一个触发产生时，ADC1 上的第一个注入通道被转换。
- 当第二个触发到达时，ADC2 上的第一个注入通道被转换。
- 如此循环.....

如果允许产生 JEOC 中断，在所有 ADC1 注入组通道转换后产生一个 JEOC 中断。

如果允许产生 JEOC 中断，在所有 ADC2 注入组通道转换后产生一个 JEOC 中断。
当所有注入组通道都转换完后，如果又有另一个外部触发，则重新开始交替触发过程。

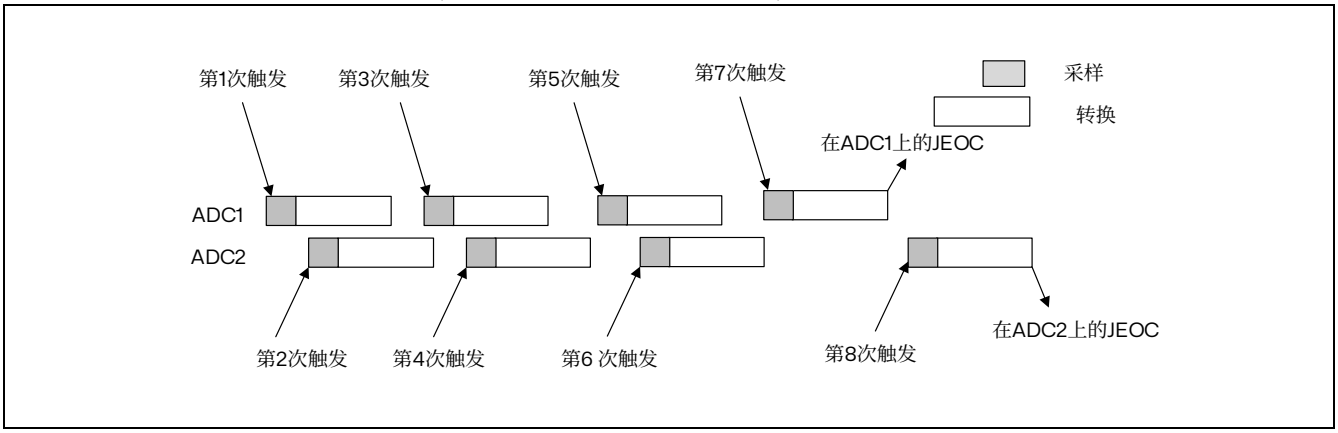


图 13.16 交替触发：在间断模式下每个 ADC 上的 4 个注入通道

13.4.6 独立模式

此模式里，双 ADC 同步不工作，每个 ADC 接口独立工作。

13.4.7 混合的规则/注入同步模式

规则组同步转换可以被中断，以启动注入组的同步转换。

注：在混合的规则/注入同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

13.4.8 混合的同步规则+交替触发模式

规则组同步转换可以被中断，以启动注入组交替触发转换。下图显示了一个规则同步转换被交替触发所中断。

注入交替转换在注入事件到达后立即启动。如果规则转换已经在运行，为了在注入转换后确保同步，所有的 ADC（主和从）的规则转换被停止，并在注入转换结束时同步恢复。

注：在混合的同步规则+交替触发模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

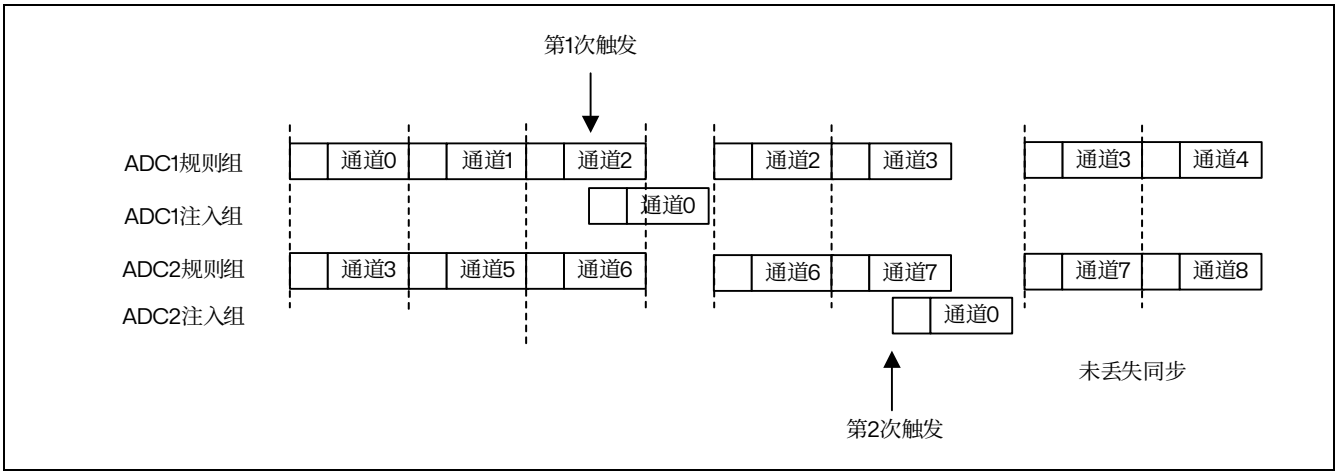


图 13.17 交替+规则同步

如果触发事件发生在一个中断了规则转换的注入转换期间，这个触发事件将被忽略。下图示出了这种情况的操作（第 2 个触发被忽略）。

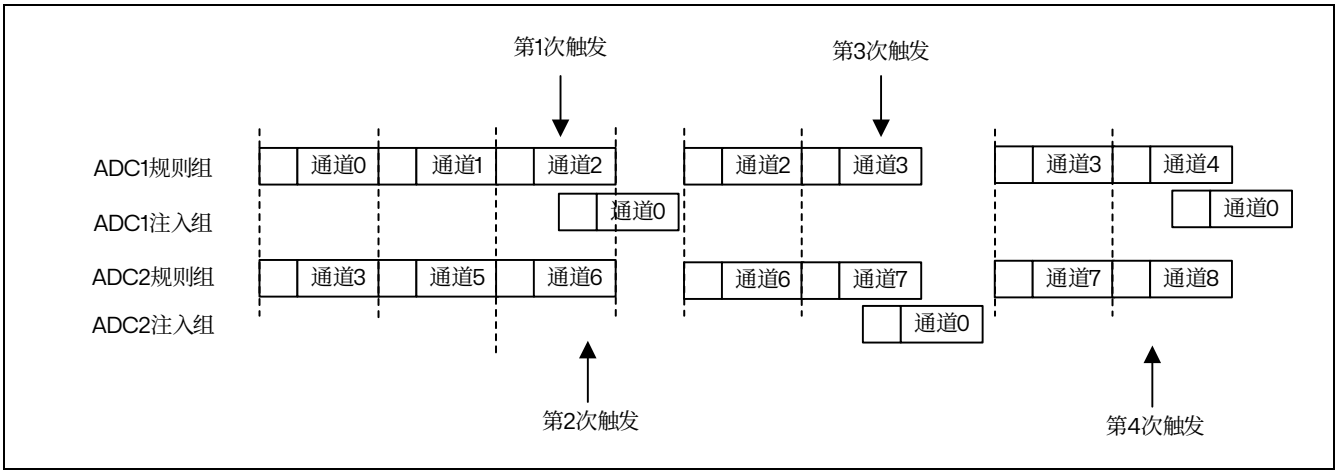


图 13.18 触发事件发生在注入转换期间

13.4.9 混合同步注入+交叉模式

一个注入事件可以中断一个交叉转换。这种情况下，交叉转换被中断，注入转换被启动，在注入序列转换结束时，交叉转换被恢复。下图是这种情况的一个例子。

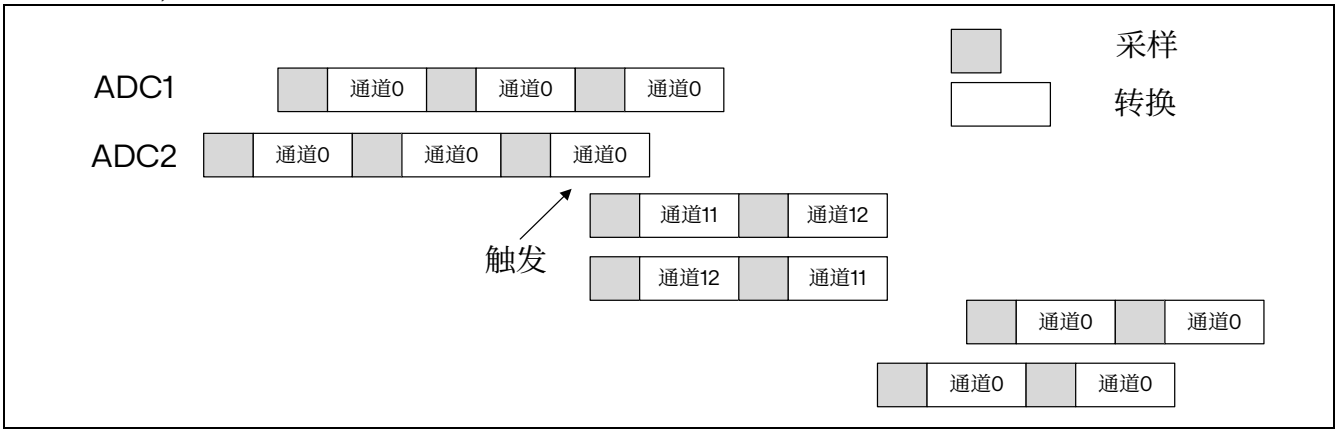


图 13.19 交叉的单通道转换被注入序列 CH11 和 CH12 中断

13.5 温度传感器（TPS）

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 ADC2 的通道 16 相连，此通道把传感器输出的电压转换成数字值。

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°C）。

注：必须设置 TSVREFE *位激活内部通道：* ADC2_IN16 *（*TPS*）和* ADC1_IN16 *（*VREFINT*）的转换。*

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。

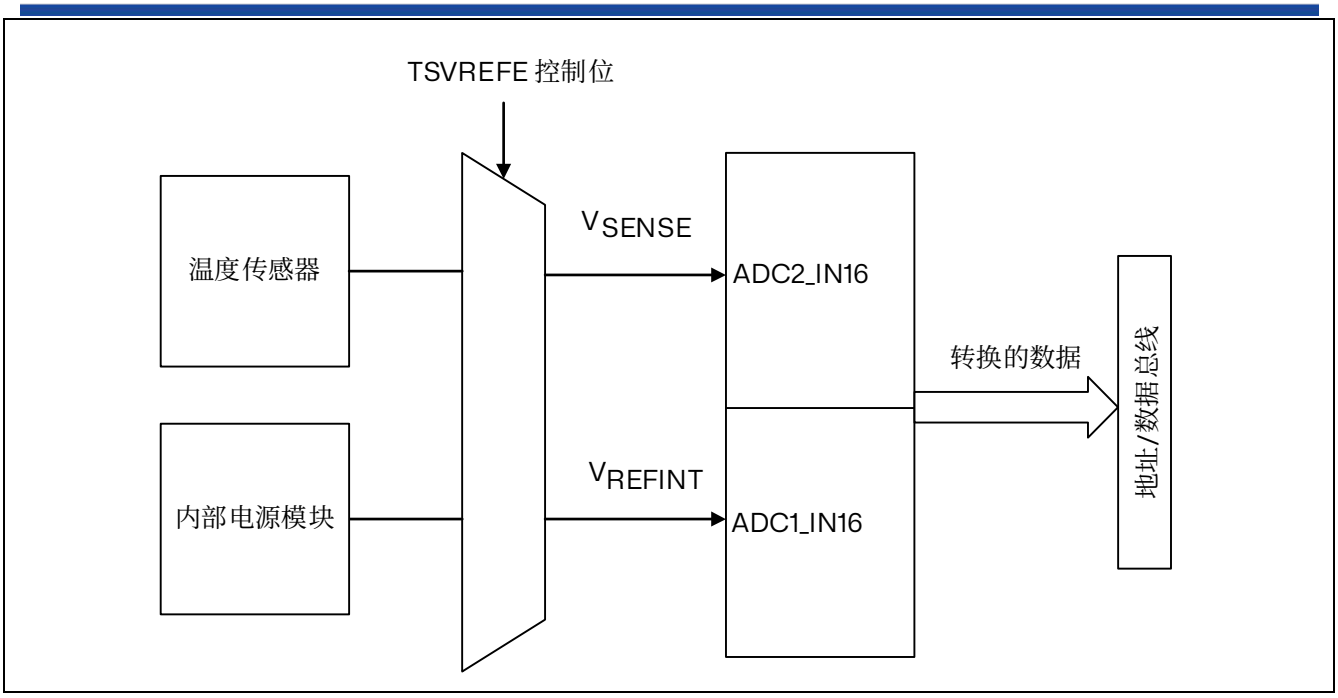


图 13.20 温度传感器和 V_{REFINT} 通道框图

13.6 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。

注：ADC1 和 ADC2 的中断映射在各自的中断向量上。

ADC_SR 寄存器中有 2 个其他标志，但是它们没有相关联的中断：

- JSTRT（注入组通道转换的启动）
- STRT（规则组通道转换的启动）

表 13.9 ADC 中断

中断事件	事件标志	使能控制位
规则组转换结束	EOC	EOCIE
注入组转换结束	JEOC	JEOCIE
设置了模拟看门狗状态位	AWD	AWDIE

13.7 ADC 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。
外设寄存器必须按字（32 位）进行访问。

13.7.1 ADC 状态寄存器（ADC_SR）

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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STRT	JSTRT	JEOC	EOC	AWD
											rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit 31:5 保留，必须保持为复位值。

Bit 4 **STRT**：规则通道开始位

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

Bit 3 **JSTRT**：注入通道开始位

该位由硬件在注入通道转换开始时设置，由软件清除。

0：注入通道转换未开始

1：注入通道转换已开始

Bit 2 **JEOC**：注入通道转换结束位

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

Bit 1 **EOC**：转换结束位

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_DR 时清除。

0：转换未完成

1：转换完成

Bit 0 **AWD**：模拟看门狗标志位

该位由硬件在转换的电压值超出了 ADC_LTR 和 ADC_HTR 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

13.7.2 ADC 控制寄存器 1（ADC_CR1）

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
Reserved					AWDFILT[2:0]			AWDEN	JAWDEN	Reserved		DUALMOD[3:0]			
					rc_w0	rc_w0	rc_w0	rw	rw			rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:27 保留，必须保持为复位值。

Bit 26:24 **AWDFILT[2:0]：**模拟看门狗滤波参数

当 AWDEN 和 JAWDEN 位都为 0 或监测通道改变时，AWDFILT 计数器清零。

000：无滤波

001：连续 2 次检测到超过模拟看门狗阈值才会置位 AWD 标志位

…

111：连续 8 次检测到超过模拟看门狗阈值才会置位 AWD 标志位

Bit 23 **AWDEN：**在规则通道上开启模拟看门狗

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

Bit 22 **JAWDEN：**在注入通道上开启模拟看门狗

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

Bit 21:20 保留，必须保持为复位值。

Bit 19:16 **DUALMOD[3:0]：**双模式选择

软件使用这些位选择操作模式。

0000：独立模式

0001：混合的同步规则+注入同步模式

0010：混合的同步规则+交替触发模式

0011：混合同步注入+快速交叉模式

0100：混合同步注入+慢速交叉模式

0101：注入同步模式

0110：规则同步模式

0111：快速交叉模式

1000：慢速交叉模式

1001：交替触发模式

注：在 ADC2 中这些位为保留位

在双模式中，改变通道的配置会产生一个重新开始的条件，这将导致同步丢失。建议在进行任何配置改变前关闭双模式。

Bit 15:13 **DISCNUM[2:0]：**间断模式通道计数

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

Bit 12 **JDISCEN：**注入通道上的间断模式

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

Bit 11 **DISCEN：**规则通道上的间断模式

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

Bit 10 **JAUTO：**自动的注入通道组转换

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

Bit 9 **AWDSGL：**扫描模式中在一个单一的通道上使用看门狗

该位由软件设置和清除，用于开启或关闭由 AWDCH[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

Bit 8 **SCAN：**扫描模式

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_SQRx 或 ADC_JSQR 寄存器选中的通道。

0：关闭扫描模式

1：使用扫描模式

*注：如果分别设置了*EOCIE *或*JEOCIE *位，只在最后一个通道转换完毕后才会产生*EOC *或*JEOC *中断。*

Bit 7 **JEOCIE：**允许产生注入通道转换结束中断

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 JEOC 中断

1：允许 JEOC 中断。当硬件设置 JEOC 位时产生中断。

Bit 6 **AWDIE：**允许产生模拟看门狗中断

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

Bit 5 **EOCIE：**允许产生 EOC 中断

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 EOC 中断

1：允许 EOC 中断。当硬件设置 EOC 位时产生中断。

Bit 4:0 **AWDCH[4:0]：**模拟看门狗通道选择位

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。

00000：ADC 模拟输入通道 0

00001：ADC 模拟输入通道 1

.....

10111：ADC 模拟输入通道 22

13.7.3 ADC 控制寄存器 2（ADC_CR2）

地址偏移：0x08

复位值：0x0100 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TS VREFE	SW START	JSW START	EXT TRIG	EXTSEL[2:0]			Res
								r	rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JEXT TRIG	JEXTSEL[3:0]				ALIGN	Reserved			DISCOUNT_ RESET[1:0]		Res	RST CAL	CAL	CONT	ADON
rw	rw	rw	rw	rw	rw				rw	rw		rw	rw	rw	rw

Bit 31:24 保留，必须保持为复位值。

Bit 23 **TSVREFE：**温度传感器和 V_{REFINT} 使能

该位由软件设置和清除，用于开启或禁止温度传感器和 V_{REFINT} 通道。

0：禁止温度传感器和 V_{REFINT}

1：使能温度传感器和 V_{REFINT}

Bit 22 **SWSTART：**开始转换规则通道

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 EXTSEL[2:0]位中选择了 SWSTART 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。

Bit 21 **JSWSTART：**开始转换注入通道

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 JEXTSEL[3:0]位中选择了 JSWSTART 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

Bit 20 **EXTTRIG**：规则通道的外部触发转换模式

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

Bit 19:17 **EXTSEL[2:0]**：选择启动规则通道组转换的外部事件

这些位选择用于启动规则通道组转换的外部事件。

ADC1/2 的触发配置如下：

000：TIM8_TRGO2

001：TIM6_TRGO

010：TIM7_TRGO

011：TIM1_TRGO2

100：TIM2_TRGO

101：TIM3_TRGO

110：ADC1：EXTI13，ADC2：EXTI14

111：SWSTART

Bit 16 保留，必须保持为复位值。

Bit 15 **JEXTTRIG**：注入通道的外部触发转换模式

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：禁止外部事件启动转换

1：使用外部事件启动转换

Bit 14:11 **JEXTSEL[3:0]**：选择启动注入通道组转换的外部事件

这些位选择用于启动注入通道组转换的外部事件。

ADC1/2 的触发配置如下：

0000：TIM8_TRGO

0001：TIM8_OC4

0010：TIM2_TRGO

0011：TIM8_OC5

0100：TIM3_OC1

0101：TIM3_TRGO

0110：ADC1：EXTI2，ADC2：EXTI3

0111：JSWSTART

1000：TIM8_TRGO2

1001：TIM6_TRGO

1010：TIM7_TRGO

1011：TIM8_OC4&OC6

1100：TIM8_OC4&OC5

1101：TIM1_TRGO

1110：TIM15_TRGO

1111：TIM1_TRGO2

Bit 10 **ALIGN**：数据对齐

该位由软件设置和清除。详情请参阅参考章节：数据对齐。

0：右对齐

1：左对齐

Bit 9:7 保留，必须保持为复位值。

Bit 6:5 **DISCOUNT_RESET[1:0]**：注入序列间断计数清零信号

00：不清零

01：TIM1_TRGO

10：TIM1_TRGO2

11：TIM8_TRGO

Bit 4 保留，必须保持为复位值。

Bit 3 **RSTCAL**：复位校准

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

*注：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。*

Bit 2 **CAL**：A/D 校准

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

Bit 1 **CONT**：连续转换

该位由软件设置和清除。如果设置了此位，则规则组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

Bit 0 **ADON**：开/关 A/D 转换器

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。

0：关闭 ADC 转换/校准，并进入断电模式

1：开启 ADC 并启动转换

*注：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。*

*注：*ADC *工作过程中，若需要重新配置*ADC *的某些寄存器（即*ADC *寄存器会动态变化），在配置之前需要先关闭*CR2_ADON，*配置完成后再开启*CR2_ADON。

13.7.4 ADC 采样时间寄存器 1（ADC_SMPR1）

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
Reserved		SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
		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
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

Bit 31:30 保留，必须保持为复位值。

Bit 29:0 **SMPx[2:0]：**通道 x 的采样时间（x = 19...10）

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。

000：2.5 周期

001：3.5 周期

010：4.5 周期

011：6.5 周期

100：11.5 周期

101：17.5 周期

110：48.5 周期

111：640.5 周期

13.7.5 ADC 采样时间寄存器 2（ADC_SMPR2）

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
Reserved		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		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
SMP5[0]	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:30 保留，必须保持为复位值。

Bit 29:0 **SMPx[2:0]：**通道 x 的采样时间（x = 9..0）

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。

000：2.5 周期

001：3.5 周期

010：4.5 周期

011：6.5 周期

100：11.5 周期

101：17.5 周期

110：48.5 周期

111：640.5 周期

13.7.6 ADC 注入通道数据偏移寄存器 x（ADC_JOFRx）（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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFFSETx[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:12 保留，必须保持为复位值。

Bit 11:0 **JOFFSETx[11:0]：**注入通道 x 的数据偏移（x = 1...4）

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_JDRx 寄存器中读出。

13.7.7 ADC 看门狗高阈值寄存器（ADC_HTR）

地址偏移：0x24

复位值：0x0000 0FFF

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

Bit 31 **HTREN：**该位始能后，ADC 转换 HTRCH 通道后，会将 ADC 转换值读取到 HTR 寄存器

0：禁止

1：使能

Bit 30:21 保留，必须保持为复位值。

Bit 20:16 **HTRCH[5:0]：**HTR 通道选择位

00000：通道 0

...

10000：通道 16

Bit 15:12 保留，必须保持为复位值。

Bit 11:0 **HTR[11:0]：**模拟看门狗高阈值

这些位定义了模拟看门狗的阈值高限。

13.7.8 ADC 看门狗低阈值寄存器（ADC_LTR）

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
LTREN	Reserved										LTRCH[5:0]				
rw											rw	rw	rw	rw	rw

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

Bit 31 **LTREN：**该位始能后，ADC 转换 LTRCH 通道后，会将 ADC 转换值读取到 LTR 寄存器

0：禁止

1：使能

Bit 30:21 保留，必须保持为复位值。

Bit 20:16 **LTRCH[5:0]：**LTR 通道选择位

00000：通道 0

...

10000：通道 16

Bit 15:12 保留，必须保持为复位值。

Bit 11:0 **LTR[11:0]：**模拟看门狗低阈值

这些位定义了模拟看门狗的阈值低限。

13.7.9 ADC 规则序列寄存器 1（ADC_SQR1）

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
Reserved									L[2:0]			Reserved			
									rw	rw	rw				

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

Bit 31:23 保留，必须保持为复位值。

Bit 22:20 **L[2:0]：**规则通道序列长度

这些位由软件定义在规则通道转换序列中的通道数目。

000：1 个转换

001：2 个转换

.....

111：8 个转换

Bit 19:0 保留，必须保持为复位值。

13.7.10 ADC 规则序列寄存器 2（ADC_SQR2）

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
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SQ8[4:0]					SQ7[4:0]				
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:10 保留，必须保持为复位值。

Bit 9:0 **SQx[3:0]：**规则序列中的第 x 个转换（x = 8...7）

这些位由软件定义转换序列中的第 x 个转换通道的编号

13.7.11 ADC 规则序列寄存器 3（ADC_SQR3）

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
Reserved		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		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

Bit 31:30 保留，必须保持为复位值。

Bit 29:0 **SQx[3:0]：**规则序列中的第 x 个转换（x = 6...1）

这些位由软件定义转换序列中的第 x 个转换通道的编号

13.7.12 ADC 注入序列寄存器（ADC_JSQR）

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
Reserved										JL[1:0]		JSQ4[4:1]			
										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

Bit 31:22 保留，必须保持为复位值。

Bit 21:20 **JL[1:0]：**注入通道序列长度

这些位由软件定义在规则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

Bit 19:0 **JSQx [4:0]：**注入序列中的第 x 个转换（x = 4...1）

这些位由软件定义转换序列中的第 x 个转换通道的编号

注：不同于规则转换序列，如果 JL[1:0] *的长度小于* 4*，则转换的序列顺序是从（*4-JL*）开始。例如：*ADC_JSQR[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。

13.7.13 ADC 注入数据寄存器 x（ADC_JDRx）(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
Reserved															

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **JDATA[15:0]：**注入转换的数据

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。详情请参阅参考章节：数据对齐。

13.7.14 ADC 规则数据寄存器（ADC_DR）

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **DATA[15:0]：**规则转换的数据

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。详情请参阅参考章节：数据对齐。

13.7.15 ADC 控制寄存器 3（ADC_CR3）

地址偏移：0x50

复位值：0x0000 0000

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

Bit 31:17 保留，必须保持为复位值。

Bit 16 **BASW：**ADC 采样遮蔽长度

0：遮蔽长度为 ADC 采样周期

1：遮蔽长度为 ADC 采样周期+12.5T

Bit 15:0 保留，必须保持为复位值。

13.7.16 ADC 校准寄存器（ADC_CAL）

地址偏移：0x54

复位值：0x0000 0040

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

Bit 31:7 保留，必须保持为复位值。

Bit 6:0 **DATA[6:0]：**ADC 自校准的值写入该寄存器

可以手动写入值，在 ADC 转换中生效

13.7.17 ADC 控制寄存器 4（ADC_CR4）

地址偏移：0x5C

复位值：0x0000 0020

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

Bit 31:6 保留，必须保持为复位值。

Bit 5 **EOC_OPT：**转换结束标志位 EOC 控制位

0：规则或注入通道组转换结束时设置 ADC_SR 寄存器的 EOC

1：规则通道组转换结束时设置 ADC_SR 寄存器的 EOC

Bit 4:0 保留，必须保持为复位值。

13.7.18 ADC TPS 校准寄存器（ADC_TPSOFFSET）

地址偏移：0x64

复位值：0x0000 0000

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

Bit 31:8 保留，必须保持为复位值。

Bit 7:0 **OFFSET[7:0]：**TPS 的校准值写入该寄存器

可以手动写入值，也可以自加载

转换 TPS 时生效，转换值+OFFSET

其中 OFFSET[7]为符号位

13.7.19 ADC 规则数据寄存器 x（ADC_DRx）(x = 1~8)

地址偏移：0x6C-0x88

复位值：0x0000 0000

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **DRx_DATA[15:0]：**转换的规则通道 x 数据

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。

14 数模转换器（DAC）

14.1 DAC 简介

DAC 模块是 10 位电压输出数模转换器。10 位数据默认采用右对齐。每个 DAC 各有一个输出通道，每个通道对应一个转换器。可通过一个输入参考电压引脚 VREF（与其他模拟外设共享）来提高分辨率。还可以在同一输入上设置内部参考。请参见参考电压缓冲器（VREFBUF）一节。

14.2 DAC 主要特征

DAC 的主要特性如下（请参见下图：DAC 框图）

- 2 个 10 位 DAC 通道
- 10 位数据采用右对齐
- DAC 输出可与片上外设连接
- 来自 V_{REF} 引脚的输入参考电压或内部 VREFBUF 参考电压

下图所示为 DAC 通道的框图，下表则给出了通道说明。

14.3 DAC 实现

表 14.1 DAC 特征

DAC 特征	DAC1	DAC2
通道(1)	DAC1_CH1	DAC2_CH1
V_{REF} 引脚	可作为参考电压	

(1) 当选择 DACx_CH1 作为 COMPx 的 N 端使用时，为防止开关噪声，需先使能 DACx，再将 COMPx 使能。

14.4 DAC 功能描述

14.4.1 DAC 框图

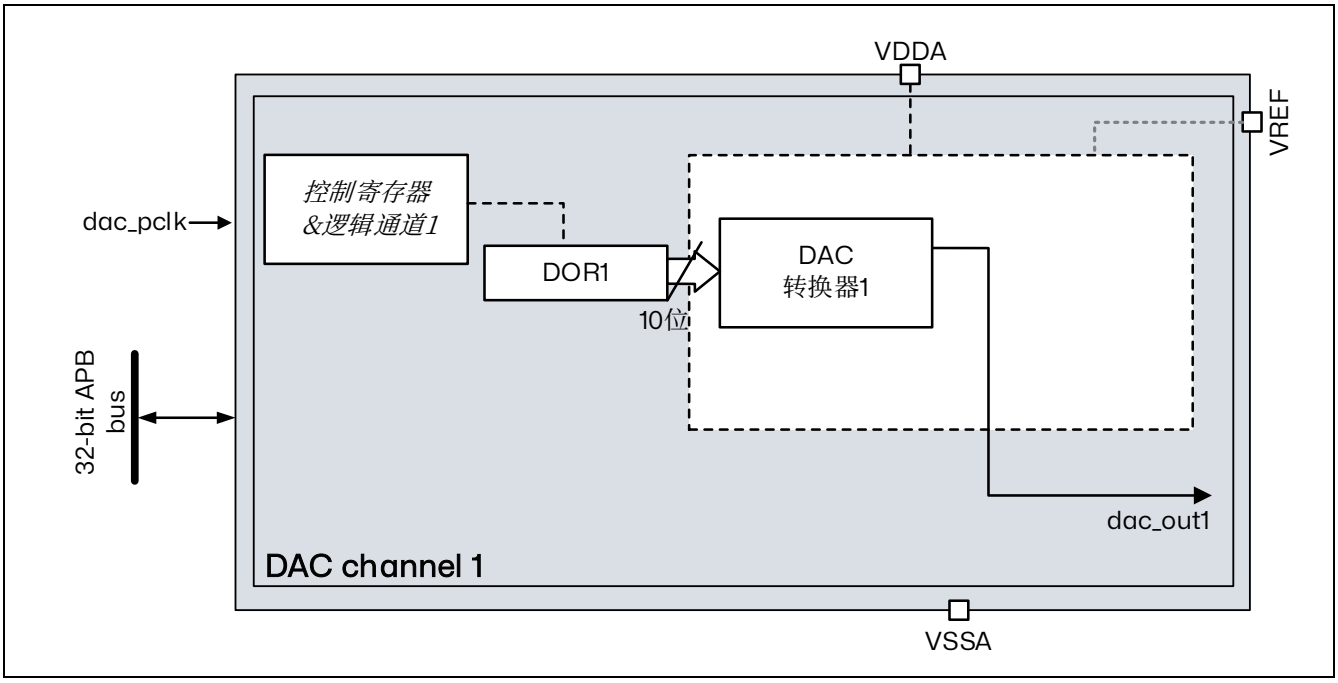


图 14.1 DAC 框图

14.4.2 DAC 引脚和内部信号

DAC 包含：

- 单独输出通道
- dac_out1 可使用与片上外设（比如比较器）的内部引脚连接。

每个 DAC 包含一条单独输出通道。两个 DAC 各自的输出通道均可连接到片上外设，比如 COMP。

当选择 DAC 通道内部连接至 COMP 反向输入端时，为防止开关噪声，需要先使能 DAC，再使能 COMP。

表 14.2 DAC 输入/输出引脚

引脚名称	信号类型	注释
VREF	模拟参考电压输入	DAC 高/正参考电压，$V_{REF} \le V_{DDAmax}$（请参见数据手册）
VDDA	模拟电源输入	等效于 V_{DD} 的模拟电源
VSSA	模拟电源地输入	等效于 V_{SS} 的模拟电源地

表 14.3 DAC 输入/输出信号

内部信号名称	信号类型	说明
dac_pclk	输入	DAC 外设时钟
dac_out1	模拟输出	片上外设的 DAC 通道 1 输出

14.4.3 DAC 通道使能

将 DACx_CR 寄存器中的相应 EN1 位置 1，即可使能对应 DAC 通道 1。经过一段启动时间 t_{WAKEUP} 后，DAC 通道 1 被真正使能。

注：EN1 位只会使能模拟 DAC 通道 1。即使 EN1 位复位，DAC 通道 1 数字接口仍处于使能状态。

14.4.4 DAC 数据格式

数据必须按如下方式写入指定寄存器：

- 对于 DAC 通道 1，采样 10 位右对齐的方式：
  - 10 位右对齐：软件必须将数据加载到 DAC_DHR10R1[9:0]位（存储到 DHR1[9:0]位）

根据加载的 DAC_DHR10R1 寄存器，用户写入的数据将移位并存储到相应的 DHR1（数据保持寄存器 1，即内部非存储器映射寄存器）。之后，DHR1 寄存器将被自动加载，或者通过软件或外部事件触发加载到 DOR1 寄存器。

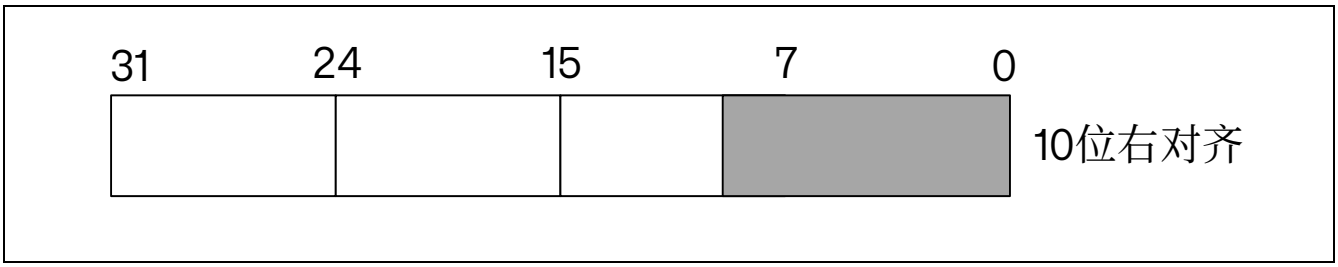


图 14.2 DAC 单通道模式下的数据寄存器

14.4.5 DAC 转换

DAC_DOR1 无法直接写入，任何数据都必须通过加载 DAC_DHR1 寄存器（写入 DAC_DHR10R1）才能传输到 DAC 通道 1。

DAC_DHR1 寄存器中存储的数据在经过一个 dac_pclk 时钟周期后，将自动转移到 DAC_DOR1 寄存器。

当 DAC_DOR1 加载了 DAC_DHR1 内容时，模拟输出电压将在一段时间 $t_{SETTLING}$ 后可用，具体时间取决于电源电压和模拟输出负载。

当 dac_pclk 时钟速度超过 80MHz 时，必须设置 DAC_MCR 的 HFSEL 位。它增加了从 DAC_DHR1 寄存器到 DAC_DOR1 寄存器传输的额外延迟。

请参考下面 HFSEL 的描述表，了解 DAC_DOR1 更新速率限制与 HFSEL 位和 dac_pclk 时钟频率的关系。

如果在不允许的期间内更新数据，则外设行为不可预测。

上述时序仅与 DAC 接口的限制有关。请参考产品数据表中的 $t_{SETTLING}$ 参数值。

表 14.4 HFSEL 描述

HFSEL[1:0]	APB 频率	功能
00	< 80 MHz	DAC_DOR 更新速率为 3 个 APB 时钟周期
01	≥ 80 MHz(1)	DAC_DOR 更新速率为 5 个 APB 时钟周期
10	≥ 160 MHz	DAC_DOR 更新速率为 7 个 APB 时钟周期
11	保留	-

1、请参考设备数据手册以获取最大 APB 频率值。

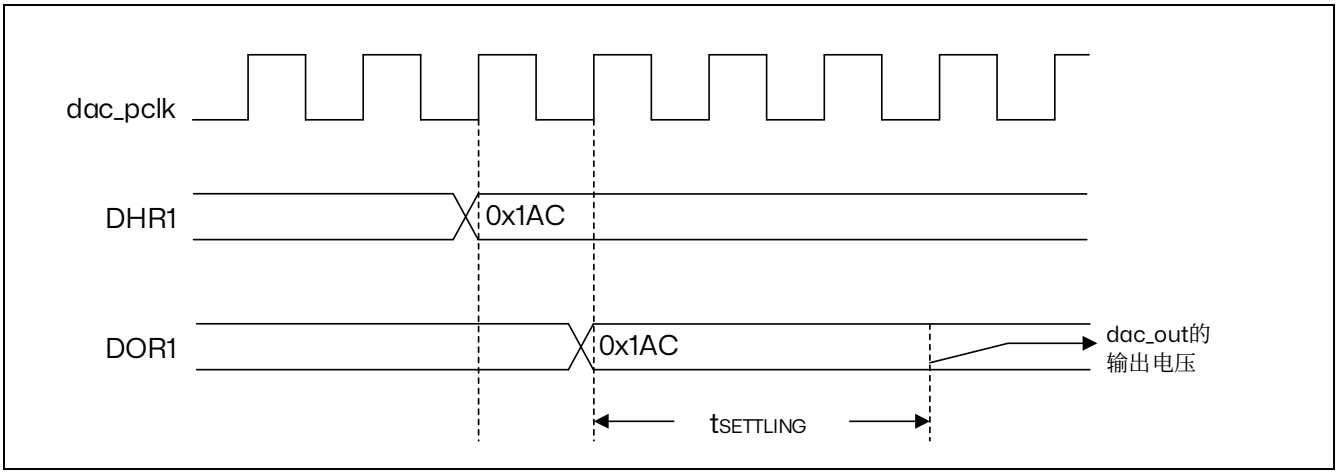


图 14.3 DAC 转换时序图

14.4.6 DAC 输出电压

经过线性转换后，数字输入会转换为 0 到 V_{REF} 之间的输出电压。

各 DAC 通道引脚的模拟输出电压通过以下公式确定：

$$DACoutput = V_{REF} \times \frac{DOR}{1024}$$

14.5 DAC 低功耗模式

表 14.5 低功耗模式对 DAC 的影响

模式	说明
睡眠	无影响。
停止	DAC 工作被停止并保留其寄存器内容。
待机	DAC 外设掉电，退出待机模式后必须重新初始化。

14.6 DAC 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。
外设寄存器必须按字（32 位）进行访问。

14.6.1 DAC 控制寄存器（DAC_CR）

偏移地址：0x00
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN1
															rw

Bit 31:1 保留，必须保持为复位值。

Bit 0 **EN1：**DAC 通道 1 使能
此位由软件置 1 和清零，以使能/禁止 DAC 通道 1。
0：禁止 DAC 通道 1
1：使能 DAC 通道 1

14.6.2 DAC 通道 1 10 位右对齐数据保持寄存器（DAC_DHR10R1）

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
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DACC1DHR[9:0]									
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:10 保留，必须保持为复位值。

Bit 9:0 **DACC1DHR[9:0]：**DAC 通道 1 的 10 位右对齐数据
这些位通过软件写入，用于指定 DAC 通道 1 的 10 位数据。

14.6.3 DAC 通道 1 数据输出寄存器（DAC_DOR1）

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
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						DACC1DOR[9:0]									
						r	r	r	r	r	r	r	r	r	r

Bit 31:10 保留，必须保持为复位值。

Bit 9:0 **DACC1DOR[9:0]：**DAC 通道 1 数据输出

这些位为只读，其中包含 DAC 通道 1 的数据输出。

14.6.4 DAC 模式控制寄存器（DAC_MCR）

偏移地址：0x3C

复位值：0x0000 0000

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

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:14 **HFSEL[1:0]：**高频接口模式选择

00：禁止高频接口模式，DAC_DOR 更新速率为 3 个 APB 时钟周期

01：使能与 APB >80 MHz 兼容的高频接口模式，使 DAC_DOR 更新速率达到 5 个 APB 时钟周期

10：使能与 APB >160 MHz 兼容的高频接口模式，使 DAC_DOR 更新速率达到 7 个 APB 时钟周期

11：保留

Bit 13:0 保留，必须保持为复位值。

15 直接存储器访问控制器（DMA）

15.1 简介

直接存储器访问（DMA）用于在外设与存储器之间以及存储器与存储器之间提供高速数据传输。可以在无需任何 CPU 操作的情况下通过 DMA 快速移动数据。这样节省的 CPU 资源可供其它操作使用。

DMA 控制器采用单个 AHB 主架构。

DMA 有一个实例：DMA1（请参阅下表：DMA1 支持的通道数量的实现）。

每个通道都专门用于管理来自一个或多个外设的存储器访问请求。每个 DMA 都包含一个仲裁器，用于处理 DMA 请求之间的优先级。

15.2 DMA 主要特征

- 单 AHB 主设备
- 外设到存储器，存储器到外设，存储器到存储器和外设到外设数据传输
- 以源和目标身份访问片上存储器映射设备，如闪存、SRAM 以及 AHB 和 APB 外设
- 所有 DMA 通道均可独立配置：
  - 每个通道都与来自外设的 DMA 请求信号相关联，或与存储器到存储器传输中的软件触发相关联。此配置由软件完成。
  - 请求之间的优先级可通过软件编程（每个通道有四个级别：非常高、高、中、低），在优先级相等的情况下也可通过硬件确定（例如，对通道 1 的请求比对通道 2 的请求具有优先权）。
  - 源和目标传输大小独立（字节、半字、字），模拟打包和解包。源地址和目标地址必须根据数据大小对齐。
  - 支持使用循环缓冲区管理从/向外设到/从存储器的传输
  - 可编程的数据传输数量：0 到 2^{16} - 1
- 每个通道生成中断请求。每个中断请求由以下三个 DMA 事件中的任意一个引起：传输完成、半传输或传输错误。

15.3 DMA 实现

15.3.1 DMA1 实现

DMA1 的实现采用了下表所示的硬件配置参数。

表 15.1 DMA1 实现

特征	DMA1
数量通道	3

15.4 DMA 功能描述

15.4.1 DMA 框图

DMA 框图如下图所示。

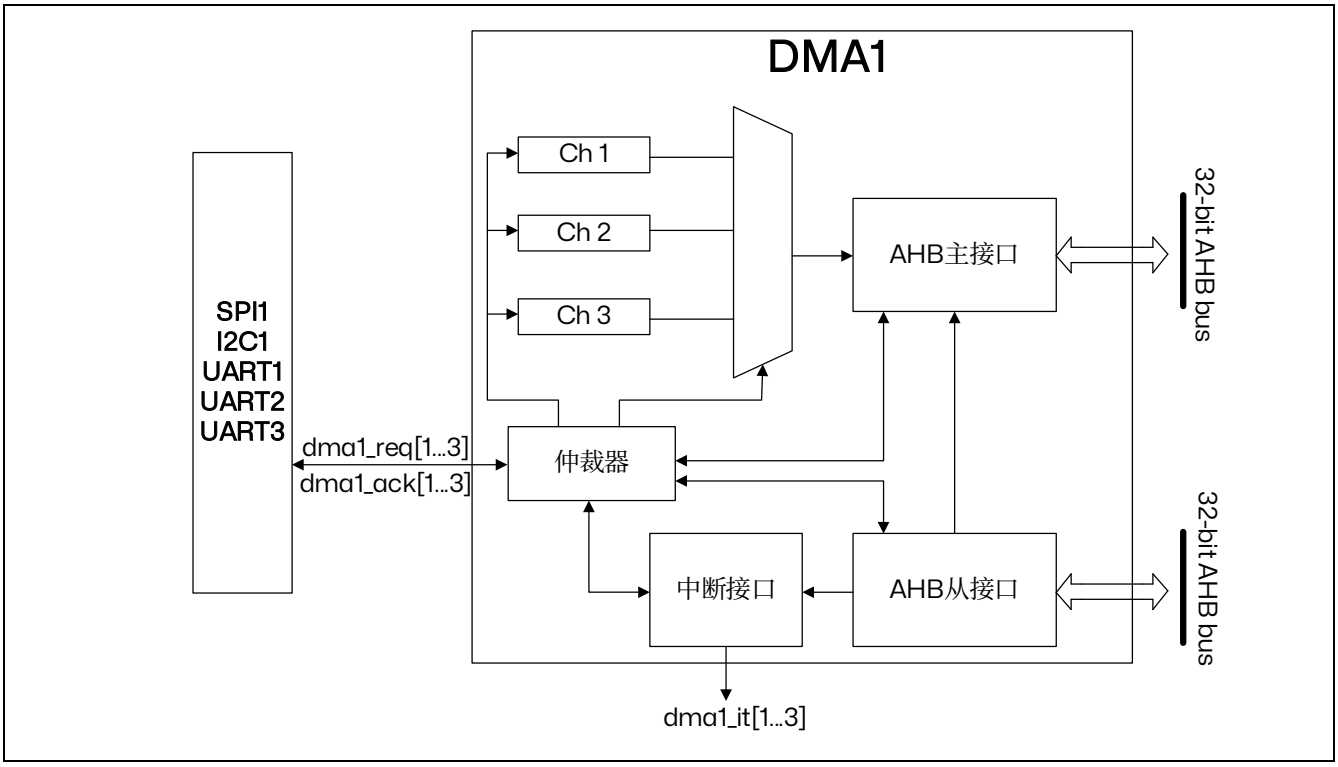


图 15.1 DMA 框图

注：请参阅上图：DMA 框图，了解功能实现。

DMA 控制器通过与其他系统主设备共享 AHB 系统总线来执行直接存储器传输。总线矩阵采用轮询调度。当 CPU 和 DMA 指向同一目标（存储器或外设）时，DMA 请求可能会阻止 CPU 访问系统总线几个总线周期。

根据通过 AHB 从接口的配置，DMA 控制器在 DMA 通道及其相关接收请求之间进行仲裁。DMA 控制器还通过单个 AHB 端口主设备调度 DMA 数据传输。

DMA 控制器为每个通道向中断控制器生成一个中断。

15.4.2 DMA 引脚和内部信号

表 15.2 DMA 内部输入输出信号

信号名称	信号类型	描述
dma_req[x]	输入	通道 x 请求
dma_ack[x]	输出	通道 x 应答
dma_it[x]	输出	通道 x 中断

15.4.3 DMA 请求映像

DMA1 控制器

从外设（SPI1、I2C1 和 UARTx[x=1、2、3]）产生的 3 个请求，通过逻辑或输入到 DMA1 控制器，这意味着同时只能有一个请求有效。参见下图的 DMA1 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。

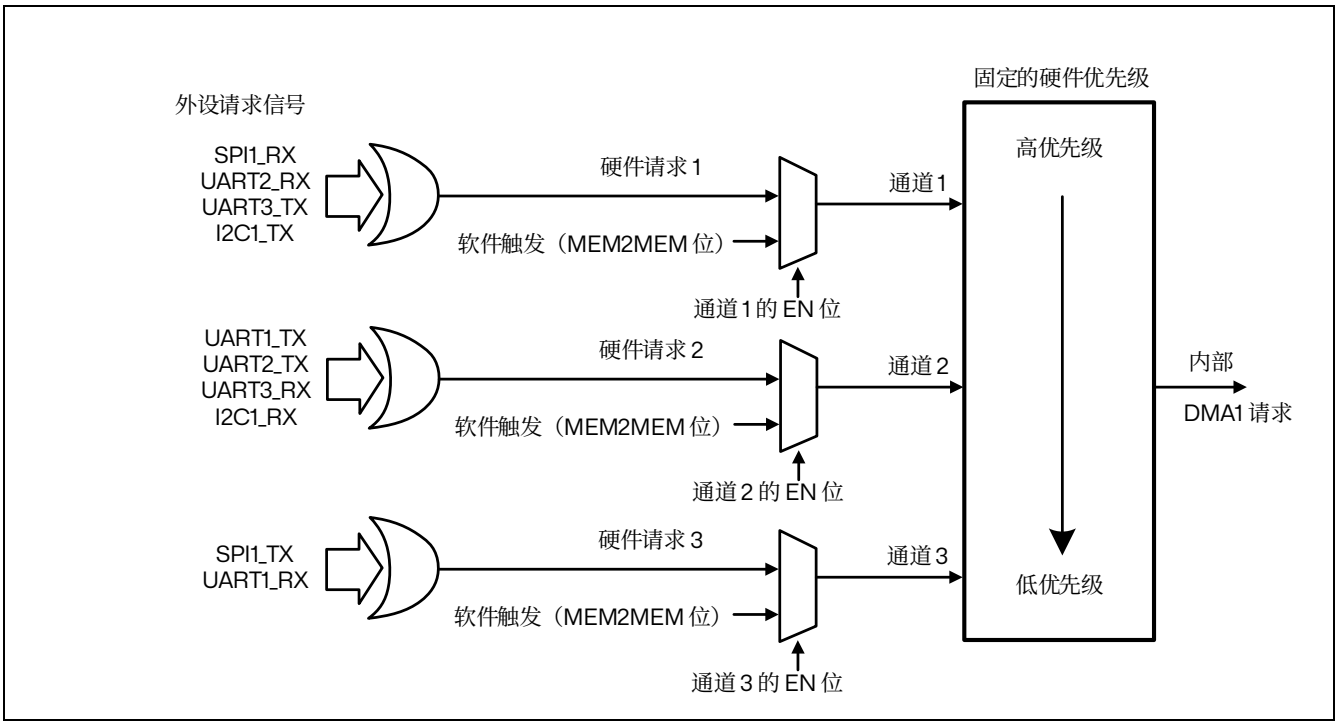


图 15.2 DMA1 请求映像

表 15.3 各个通道的 DMA1 请求一览

外设	通道 1	通道 2	通道 3
SPI	SPI1_RX	-	SPI1_TX
UART	UART2_RX UART3_TX	UART1_TX UART2_TX UART3_RX	UART1_RX
I2C	I2C1_TX	I2C1_RX	-

15.4.4 DMA 仲裁器传输

软件在通道级别配置 DMA 控制器，以执行由一系列 AHB 总线传输组成的块传输。

可以从外设请求 DMA 块传输，或者在内存到内存的传输情况下由软件触发。

在事件发生后，单个 DMA 传输的后续步骤如下：

1. 外设向 DMA 控制器发送单个 DMA 请求信号。
2. DMA 控制器根据与此外设请求相关联的通道的优先级来处理请求。
3. 一旦 DMA 控制器授予外设权限，DMA 控制器就会向外设发送确认信号。
4. 外设在从 DMA 控制器收到确认后，立即释放其请求。
5. 一旦外设取消请求，DMA 控制器就会释放确认信号。

外设可以发出另一个单请求并启动另一个 DMA 单次传输。

当外设是传输的源或目标时，会使用请求/确认协议。例如，在存储器到外设的传输中，外设通过将其单个请求信号发送到 DMA 控制器来启动传输。然后，DMA 控制器从存储器中的某个位置（通过内部当前外设/存储器地址寄存器寻址）读取单个数据，并将该数据写入外设。

对于给定通道 x，DMA 块传输由以下重复序列组成：

- 单个 DMA 传输，在 DMA AHB 总线主设备上封装两个单个数据的 AHB 传输：
  - 从外设数据寄存器或存储器中的某个位置（通过内部当前外设/存储器地址寄存器寻址）读取单个数据（字节、半字或字）。
    用于第一次单次传输的起始地址是外设或存储器的基地址，并在 DMA_CPARx 或 DMA_CMARx 寄存器中编程。
  - 向外设数据寄存器或存储器中的某个位置（通过内部当前外设/存储器地址寄存器寻址）写入单个数据（字节、半字或字）。
    用于第一次传输的起始地址是外设或存储器的基地址，并在 DMA_CPARx 或 DMA_CMARx 寄存器中编程。
- 对已编程的 DMA_CNDTRx 寄存器进行后递减
- 该寄存器包含要传输的数据项剩余数量（AHB“读取后跟写入”传输的数量）。
  此序列会一直重复，直到 DMA_CNDTRx 为零。

注：AHB 主总线源/目标地址必须与传输到源/目标的单个数据的编程大小对齐。

15.4.5 DMA 仲裁器

DMA 仲裁器管理不同通道之间的优先级。

当仲裁器（硬件请求或软件触发）授予活动通道 x 时，会发出一次 DMA 传输（例如，AHB 的“先读后写”单数据传输）。然后，仲裁器会再次考虑活动通道集合，并选择优先级最高的通道。

优先级的管理分为两个阶段：

- 软件：在 DMA_CCRx 寄存器中，将每个通道的优先级配置为四个不同等级之一：
  - 最高
  - 高
  - 中
  - 低
- 硬件：如果两个请求的软件优先级相同，则编号最低的通道获得优先权。例如，通道 2 的优先级高于通道 4。

当通道 x 被编程为存储器到存储器模式的块传输时，在该通道 x 的每次 DMA 传输之间都会进行再次仲裁。每当有另一个并发活动的请求通道时，DMA 仲裁器会自动切换并授予另一个优先级最高的请求通道，该通道的优先级可能低于存储器到存储器通道。

15.4.6 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。包含要传输的数据项数量的寄存器，在每次传输后递减。

DMA 通道是在块传输级别上进行编程的。

可编程的数据量

通过 DMA_CCRx 寄存器的 PSIZE[1:0]和 MSIZE[1:0]位域，可以分别对外设和存储器中的单个数据（字节、半字或字）的传输数据量进行编程。

指针增量

通过设置 DMA_CCRx 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，增量值取决与所选的数据宽度为 1、2 或 4。第一个传输的地址是存放在 DMA_CPARx/DMA_CMARx 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当前正在传输的地址（它在内部的当前外设/存储器地址寄存器中）。

当通道 x 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_CNDTRx 寄存器中重新写入传输数目。

注：如果通道 x 被禁用，DMA 寄存器不会重置。DMA 通道寄存器（DMA_CCRx、DMA_CPARx 和 DMA_CMARx）保留在通道配置阶段编程的初始值。

在**循环模式**下，最后一次传输结束时，DMA_CNDTRx 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_CPARx 和 DMA_CMARx 寄存器设定的初始基地址。

通道配置过程

下面是配置 DMA 通道 x 的过程，需要按照以下顺序操作：

1. 在 DMA_CPARx 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_CMARx 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_CNDTRx 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_CCRx 寄存器中配置以下参数：
   - 通道优先级
   - 数据传输方向
   - 循环模式
   - 外设和存储器递增模式
   - 外设和存储器数据大小
   - 半传输和/或全传输和/或传输错误时的中断使能

5. 设置 DMA_CCRx 寄存器的 EN 位，启动该通道。

通道一旦使能，就可以处理连接到该通道的外设发出的任何 DMA 请求，或者可以启动存储器到存储器的块传输。

注：通道配置程序的最后两个步骤可以合并成对 DMA_CCRx 寄存器的单次访问，以配置和使能通道。

通道状态和禁用通道

处于活动状态的通道 x 是一个已使能的通道（读取 DMA_CCRx.EN = 1）。活动通道 x 必须是已由软件使能（DMA_CCRx.EN 设置为 1）且之后未发生传输错误（DMA_ISR.TEIFx = 0）的通道。如果发生传输错误，硬件会自动禁用该通道（DMA_CCRx.EN = 0）。

以下三种用例可能发生：

- 暂停和恢复通道
  这对应于以下两个操作：
  – 软件禁用活动通道（在 DMA_CCRx.EN = 1 时写入 DMA_CCRx.EN = 0）。
  – 软件在不重新配置其他通道寄存器（如 DMA_CNDTRx、DMA_CPARx 和 DMA_CMARx）的情况下再次使能通道（将 DMA_CCRx.EN 设置为 1）。
  DMA 硬件不支持此用例，不保证剩余的数据传输能正确执行。
- 停止和中止通道
  如果应用程序不再需要该通道，软件可以禁用此活动通道。通道会停止并中止，但 DMA_CNDTRx 寄存器的内容可能无法正确反映与已中止的源和目标缓冲区/寄存器相比的剩余数据传输。
- 中止并重新启动通道
  这对应于软件的以下步骤：禁用活动通道，然后重新配置通道并再次使能它。
  如果满足以下条件，硬件支持此操作：
  – 应用程序确保在软件禁用通道时，没有通过其主端口同时进行 DMA 数据传输。例如，应用程序可以先禁用外设的 DMA 模式，以确保没有来自该外设的挂起硬件 DMA 请求。
  – 软件必须对相同的 DMA_CCRx 寄存器执行单独的写访问操作：首先禁用通道。其次，如果需要更改配置，则为下一个块传输（包括 DMA_CCRx）重新配置通道。当 DMA_CCRx.EN=1 时，存在只读 DMA_CCRx 寄存器位域。最后，再次使能通道。

当发生通道传输错误时，DMA_CCRx 寄存器的 EN 位由硬件清除。在 DMA_ISR 寄存器的 TEIFx 位被设置之前，软件无法再次设置此 EN 位以重新激活通道 x。

循环模式（在存储器到外设/外设到存储器传输中）

循环模式用于处理循环缓冲区和连续的数据传输（如 SPI 连续通信）。在 DMA_CCRx 寄存器中的 CIRC 位用于开启这一功能。

注：在存储器到存储器模式下不得使用循环模式。在使能循环模式（CIRC = 1）的通道之前，软件必须清除 DMA_CCRx 寄存器的 MEM2MEM 位。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。

为了停止循环传输，软件需要在禁用 DMA 通道之前停止外设生成 DMA 请求（如停止 SPI 通信）。在开始/使能传输之前以及在停止循环传输之后，软件必须明确地为 DMA_CNDTRx 编程值。

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_CCRx 寄存器中的 MEM2MEM 位之后，在软件设置了 DMA_CCRx 寄存器中的 EN 位启动 DMA 通道时，DMA 传输将马上开始。当 DMA_CNDTRx 寄存器变为 0 时，DMA 传输结束。

*注：在循环模式下不得使用存储器到存储器模式。在使能存储器到存储器模式（*MEM2MEM = 1*）的通道之前，软件必须清除* DMA_CCRx *寄存器的* CIRC *位。*

外设到外设模式

任何 DMA 通道都可以在外设到外设模式下运行：

- 当选择外设的硬件请求来触发 DMA 通道时
  此外设是 DMA 发起者，并控制从此外设到属于另一个存储器映射外设的寄存器的数据传输（该外设未配置为 DMA 模式）。
- 当没有选择并连接到 DMA 通道的外设请求时
  软件通过设置 DMA_CCRx 寄存器的 MEM2MEM 位来配置寄存器到寄存器的传输。

编程传输方向，分配源/目标

DMA_CCRx 寄存器的 DIR 位的值设置传输方向，并因此确定源和目标，而不考虑源/目标类型（外设或存储器）：

- DIR = 1 通常定义存储器到外设的传输。更一般地说，如果 DIR = 1：
  - **源**属性由 DMA_MARx 寄存器、DMA_CCRx 寄存器的 MSIZE[1:0]位域和 MINC 位定义。无论它们的通常命名如何，这些“存储器”寄存器、位域和位都用于在外设到外设模式下定义源外设。
  - **目标**属性由 DMA_PARx 寄存器、DMA_CCRx 寄存器的 PSIZE[1:0]位域和 PINC 位定义。无论它们的通常命名如何，这些“外设”寄存器、位域和位都用于在存储器到存储器模式下定义目标存储器。
- DIR = 0 通常定义外设到存储器的传输。更一般地说，如果 DIR = 0：
  - **源**属性由 DMA_PARx 寄存器、DMA_CCRx 寄存器的 PSIZE[1:0]位域和 PINC 位定义。无论它们的通常命名如何，这些“外设”寄存器、位域和位都用于在存储器到存储器模式下定义源存储器。
  - **目标**属性由 DMA_MARx 寄存器、DMA_CCRx 寄存器的 MSIZE[1:0]位域和 MINC 位定义。无论它们通常如何命名，这些“存储器”寄存器、位域和位都用于在外设到外设模式下定义目标外设。

15.4.7 DMA 数据宽度、对齐方式和数据大小端

当 PSIZE[1:0]和 MSIZE[1:0]不相等时，DMA 控制器会执行一些数据对齐操作，如下表所述。

表 15.4 可编程的数据传输宽度和大小端操作（当 PINC = MINC = 1）

源端宽度（MSIZE 如果 DIR=1, 则为 PSIZE）	目标宽度（PSIZE 如果 DIR=1, 则为 MSIZE）	传输数目（NDT）	源：地址/数据（DMA_CMARx 如果 DIR=1, 则为 DMA_CPARx）	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[15:0]，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复。

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，如果要写入某些 APB 外设的特定功能寄存器时（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（MSIZE）为 16 位，外设目标数据宽度（PSIZE）为 32 位。

15.4.8 DMA 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 x 所对应的通道配置寄存器 DMA_CCRx 的 EN 位，该通道操作被停止。

DMA_ISR 寄存器中对应该通道的传输错误中断标志位 TEIFx 将被置位，如果在 DMA_CCRx 寄存器中设置了传输错误中断允许位，则将产生中断。

在 DMA_ISR 寄存器的 TEIFx 位被清除（通过置位 DMA_IFCR 寄存器的 CTEIFx 位）之前，软件无法再次置位 DMA_CCRx 寄存器的 EN 位（通道 x 重新激活）。

当软件收到涉及外设的通道传输错误通知时，软件必须首先停止该外设的 DMA 模式，以禁用任何挂起或未来的 DMA 请求。然后，软件可以正常重新配置 DMA 和外设在 DMA 模式下进行新的传输。

15.4.9 DMA 中断

每个 DMA 通道 x 可以在半传输、传输完成或传输错误时生成中断。为提供灵活性，提供了单独的中断使能位。

表 15.5 DMA 中断请求

中断请求	中断事件	事件标志	中断使能位
通道 x 中断	通道 x 半传输	HTIFx	HTIEx
	通道 x 传输完成	TCIFx	TCIEx
	通道 x 传输错误	TEIFx	TEIEx
	通道 x 半传输或传输完成或传输错误	GIFx	-

15.5 DMA 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。

外设寄存器必须按字（32 位）进行访问。

注：请参阅 DMA *框图以了解功能实现。*

15.5.1 DMA 中断状态寄存器（DMA_ISR）

偏移地址：0x00

复位值：0x0000 0000

当在 DMA_IFCR 寄存器中设置相应清除位或相应的全局清除位 CGIFx 时，硬件会清除每个状态位。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
				r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:12 保留，必须保持为复位值。

Bit 4*(x-1)+3 **TEIFx：**通道 x 的传输错误标志（x = 1...3）

硬件设置这些位。在 DMA_IFCR 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）。

Bit 4*(x-1)+2 **HTIFx：**通道 x 的半传输标志（x = 1...3）

硬件设置这些位。在 DMA_IFCR 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）。

Bit 4*(x-1)+1 **TCIFx：**通道 x 的传输完成标志（x = 1...3）

硬件设置这些位。在 DMA_IFCR 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）。

Bit 4*(x-1)+0 **GIFx：**通道 x 的全局中断标志（x = 1...3）

硬件设置这些位。在 DMA_IFCR 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。

15.5.2 DMA 中断寄存器标志清除寄存器（DMA_IFCR）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CTEIF3	CHTIF3	CTCIF3	CGIF3	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
				w	w	w	w	w	w	w	w	w	w	w	w

Bit 31:12 保留，必须保持为复位值。

Bit 4*(x-1)+3 **CTEIFx：**清除通道 x 的传输错误标志（x = 1...3）

这些位由软件设置和清除。

0：无作用

1：清除 DMA_ISR 寄存器中的对应 TEIF 标志。

Bit 4*(x-1)+2 CHTIFx：清除通道 x 的半传输标志（x = 1...3）

这些位由软件设置和清除。

0：无作用

0：清除 DMA_ISR 寄存器中的对应 HTIF 标志。

Bit 4*(x-1)+1 CTCIFx：清除通道 x 的传输完成标志（x = 1...3）

这些位由软件设置和清除。

0：无作用

0：清除 DMA_ISR 寄存器中的对应 TCIF 标志。

Bit 4*(x-1)+0 CGIFx：清除通道 x 的传全局中断标志（x = 1...3）

这些位由软件设置和清除。

0：无作用

0：清除 DMA_ISR 寄存器中的对应 GIF、TEIF、HTIF、TCIF 标志。

15.5.3 DMA 通道 x 配置寄存器（DMA_CCRx）

偏移地址：0x08 + 20 * (x-1)，（x = 1 ~ 3）
复位值：0x0000
当 EN=1 时，MEM2MEM、PL[1:0]、MSIZE[1:0]、PSIZE[1:0]、MINC、PINC 和 DIR 是只读的。
MEM2MEM 和 CIRC 位的状态不能同时为高。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:15 保留，必须保持为复位值。

Bit 14 **MEM2MEM：**存储器到存储器模式

0：禁止

1：使能

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位是只读的。

Bit 13:12 **PL[1:0]：**通道优先级

00：低

01：中

10：高

11：最高

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域是只读的。

Bit 11:10 **MSIZE[1:0]：**存储器数据宽度

定义了每次 DMA 传输到指定存储器的数据宽度。

在存储器到存储器模式下如果 DIR=1，该位域表示存储器源；如果 DIR=0，则表示存储器目标。

在外设到外设模式下如果 DIR=1，该位域表示外设源；如果 DIR=0，则表示外设目标。

00：8 位

01：16 位

10：32 位

11：保留

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域是只读的。

Bit 9:8 **PSIZE[1:0]：**外设数据宽度

这些位由软件设置和清除。

在存储器到存储器模式下如果 DIR=1，该位域表示存储器目标；如果 DIR=0，则表示存储器源。

在外设到外设模式下如果 DIR=1，该位域表示外设目标；如果 DIR=0，则表示外设源。

00：8 位

01：16 位

10：32 位

11：保留

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域是只读的。

Bit 7 **MINC：**存储器递增模式

定义了每次 DMA 传输到指定存储器时的递增模式。

在存储器到存储器模式下如果 DIR=1，该位表示存储器源；如果 DIR=0，则表示存储器目标。

在外设到外设模式下如果 DIR=1，该位表示外设源；如果 DIR=0，则表示外设目标。

0：禁止

1：使能

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位是只读的。

Bit 6 **PINC：**外设地址增量模式

该位由软件设置和清除。

在存储器到存储器模式下如果 DIR=1，该位表示存储器目标；如果 DIR=0，则表示存储器源。

在外设到外设模式下如果 DIR=1，该位表示外设目标；如果 DIR=0，则表示外设源。

0：禁止

1：使能

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位是只读的。

Bit 5 **CIRC：**循环模式

0：禁止

1：使能

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位是只读的。

Bit 4 **DIR**：数据传输方向

此位仅在存储器到外设和外设到存储器模式下设置。

0：从外设读取

– 源属性由 PSIZE 和 PINC 以及 DMA_CPARx 寄存器定义。
在存储器到存储器模式下仍然有效。

– 目标属性由 MSIZE 和 MINC 以及 DMA_CMARx 寄存器定义。
在外设到外设模式下仍然有效。

1：从存储器读取

– 目标属性由 PSIZE 和 PINC 以及 DMA_CPARx 寄存器定义。
在存储器到存储器模式下仍然有效。

– 源属性由 MSIZE 和 MINC 以及 DMA_CMARx 寄存器定义。
在外设到外设模式下仍然有效。

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位是只读的。

Bit 3 **TEIE**：传输错误中断使能

该位由软件设置和清除。

0：禁止 TE 中断

0：使能 TE 中断

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位不是只读的。

Bit 2 **HTIE**：半传输中断使能

该位由软件设置和清除。

0：禁止 HT 中断

0：使能 HT 中断

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位不是只读的。

Bit 1 **TCIE**：传输完成中断使能

该位由软件设置和清除。

0：禁止 TC 中断

0：使能 TC 中断

注：此位由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位不是只读的。

Bit 0 **EN**：通道使能

当发生通道传输错误时，此位由硬件清除。在 DMA_ISR 寄存器的 TEIFx 位被清除（通过设置 DMA_IFCR 寄存器的 CTEIFx 位）之前，软件无法再次设置此位（重新激活通道 x）。

0：通道禁止

1：通道使能

注：此位由软件设置和清除。

15.5.4 DMA 通道 x 传输数量寄存器（DMA_CNDTRx）

偏移地址：0x0C + 20 * (x-1)，（x = 1 ~ 3）

复位值：0x0000

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **NDT[15:0]：**数据传输数量（0 至 65535）

当通道启用时，此位域由硬件更新：

– 每次 DMA“先读后写”传输后，该值递减，表示剩余要传输的数据项数量。

– 如果传输的编程数据量已达到，且通道不在循环模式（DMA_CCRx 寄存器中 CIRC = 0），则该值保持为零。

– 如果通道处于循环模式（CIRC = 1），则在传输完成后，该值会自动重新加载为之前编程的值。

如果此位域为零，则无论通道状态如何（使能或禁止），都无法进行传输。

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域是只读的。

15.5.5 DMA 通道 x 外设地址寄存器（DMA_CPARx）

偏移地址：0x10 + 20 * (x-1)，（x = 1 ~ 3）

复位值：0x0000 0000

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

Bit 31:16 **PA[31:0]：**外设地址

它包含从/向外设数据寄存器读取/写入数据的基地址。

当 PSIZE[1:0] = 01（16 位），PA[31:0]的位 0 被忽略。访问将自动对齐到半字地址。

当 PSIZE[1:0] = 10（32 位），PA[31:0]的位 1 和位 0 被忽略。访问将自动对齐到字地址。

在存储器到存储器模式下如果 DIR = 1，该位域表示存储器目标地址；如果 DIR = 0，则表示存储器源地址。

在外设到外设模式下如果 DIR = 1，该位域表示外设目标地址；如果 DIR = 0，则表示外设源地址。

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域不是只读的。

15.5.6 DMA 通道 x 外设存储地址寄存器（DMA_CMARx）

偏移地址：0x14 + 20 * (x-1)，（x = 1 ~ 3）

复位值：0x0000 0000

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

Bit 31:0 **MA[31:0]：**存储器地址

它包含从/向存储器读取/写入数据的基地址。

当 MSIZE[1:0] = 01（16 位），MA[31:0]的位 0 被忽略。访问将自动对齐到半字地址。

当 MSIZE[1:0] = 10（32 位），MA[31:0]的位 1 和位 0 被忽略。访问将自动对齐到字地址。

在存储器到存储器模式下如果 DIR = 1，该位域表示存储器源地址；如果 DIR = 0，则表示存储器目标地址。

在外设到外设模式下如果 DIR = 1，该位域表示外设源地址；如果 DIR = 0，则表示外设目标地址。

注：此位域由软件设置和清除。在通道使能（EN=1）时，不得对其进行写入。在通道使能（EN=1）时，此位域不是只读的。

16 比较器（COMP）

16.1 简介

该设备嵌入了 4 个超高速模拟比较器。
这些比较器可用于各种功能，包括：

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

16.2 主要特性

- 每个比较器具有可配置的正负输入，用于灵活的电压选择：
  - 多路 I/O 引脚
  - OPAMP 通道
  - DAC 通道
  - 内置的星型电阻网络 SRN 通道
- 可编程迟滞
- 输出重定向到 I/O 或定时器输入，用于触发快速 PWM 关断的刹车事件
- 输出遮蔽，以抵抗开关噪声
- 每个通道生成的中断，可用于从睡眠和停止模式唤醒

16.3 功能描述

16.3.1 COMP 框图

四个比较器各自的 COMP 框图分别如下列图所示：

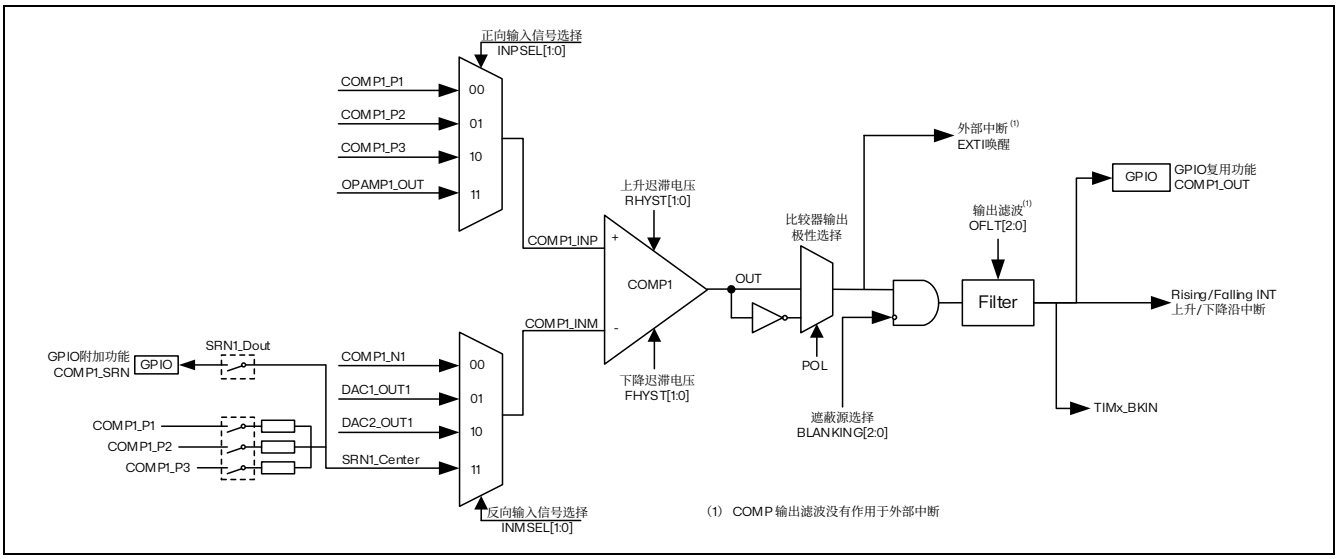


图 16.1 COMP1 框图

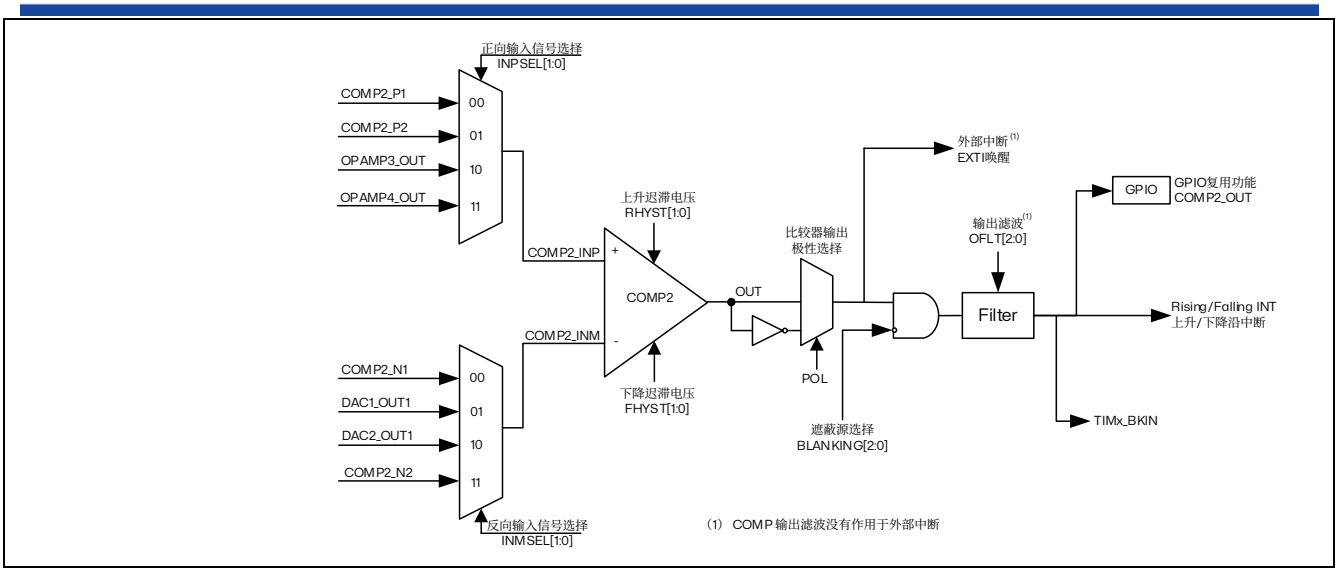


图 16.2 COMP2 框图

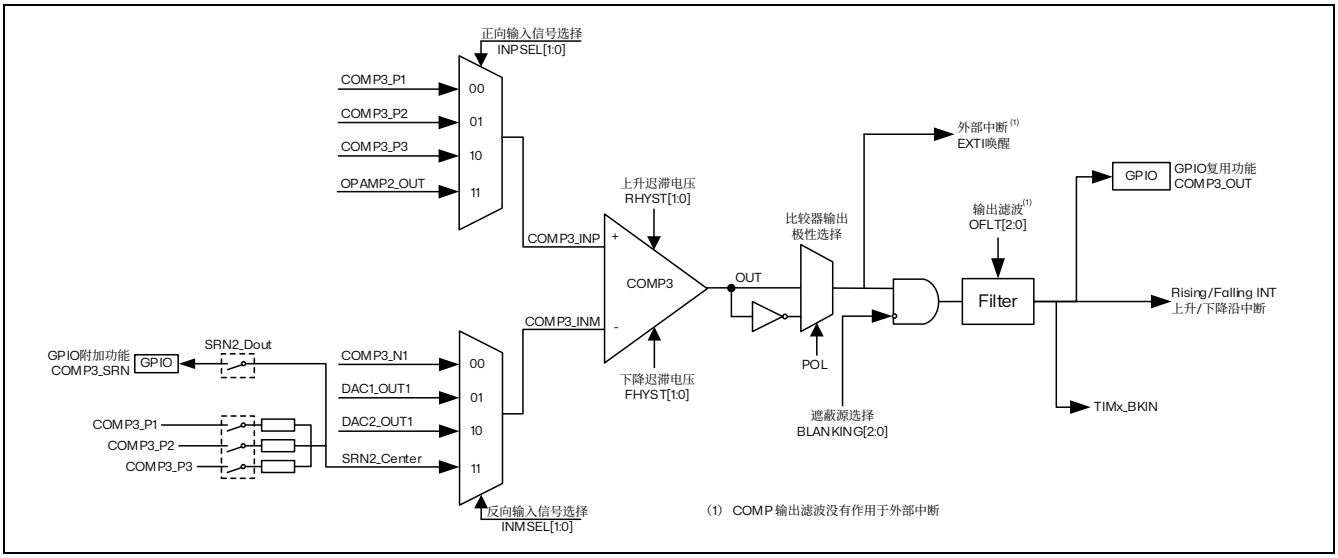


图 16.3 COMP3 框图

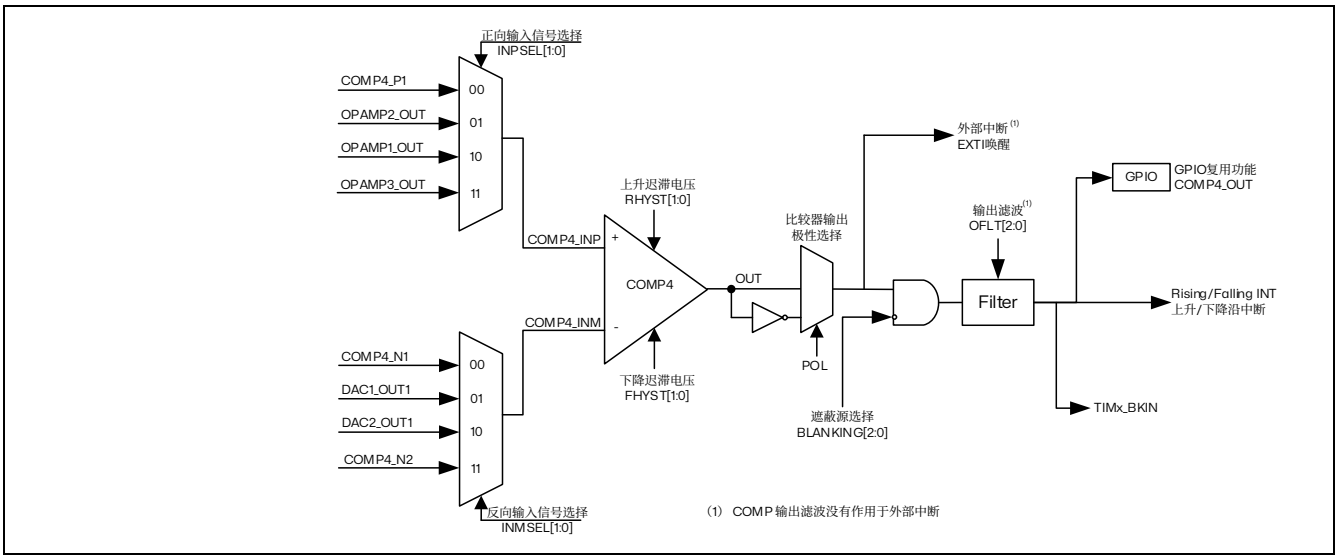


图 16.4 COMP4 框图

16.3.2 COMP 引脚和内部信号

用作比较器输入的 I/O 必须在 GPIO 寄存器中配置为模拟模式。

比较器输出可以使用数据手册“复用功能映射”表中给出的复用功能通道连接到 I/O。

输出也可以在内部重定向到用于以下用途的各种定时器输入：

- 使用 BKIN 和 BKIN2 输入紧急关断 PWM 信号
- 使用 OCREF_CLR 输入进行逐周期电流控制
- 用于时序测量的输入捕捉

可以同时将比较器输出内部和外部重定向。

表 16.1 COMPx 正向输入分配

INPSEL[1:0]	COMP1_INP	COMP2_INP	COMP3_INP	COMP4_INP
00	PA11	PA4	PB11	PB12
01 (1)	PA10	PB4	PC2	OPAMP2_OUT
10 (1)	PA9	OPAMP3_OUT	PA6	OPAMP1_OUT
11 (1)	OPAMP1_OUT	OPAMP4_OUT	OPAMP2_OUT	OPAMP3_OUT

(1) 当选择 OPAMPx_OUT 作为 COMPx 的 P 端使用时，为防止开关噪声，需先使能 OPAMPx，再将 COMPx 使能。

表 16.2 COMPx 反向输入分配

INMSEL[1:0]	COMP1_INM	COMP2_INM	COMP3_INM	COMP4_INM
00	PA7	PA8	PC3	PB13
01 (1)	DAC1_CH1	DAC1_CH1	DAC1_CH1	DAC1_CH1
10 (1)	DAC2_CH1	DAC2_CH1	DAC2_CH1	DAC2_CH1
11	SRN1_Center	PB3	SRN2_Center	PB5

(1) 当选择 DACx_CH1 作为 COMPx 的 N 端使用时，为防止开关噪声，需先使能 DACx，再将 COMPx 使能。

注：P 端选择 OPAMP，N 端使用不会选择 SRN；当 N 端选择 SRN 时，P 端只会选择 P1/2/3。

16.3.3 COMP 复位和时钟

由时钟控制器提供的 COMP 时钟与 APB2 时钟同步。

RCC 控制器中没有专用于 COMP 时钟使能控制位。复位和时钟使能位是 COMP 和 SYSCFG 共用的。

*注：**重要：**极性选择逻辑和输出重定向到端口独立于 APB 时钟工作。这使得比较器即使在停止模式下也能工作。*

16.3.4 COMP 锁定机制

比较器能用于安全的用途，比如过流或者过热保护。在某些特定的安全需求的应用中，有必要保证比较器设置不能被无效寄存器访问或者程序计数器破坏所改变。

为此，比较器控制和状态寄存器可以是写保护的（只读）。

一旦编程完成，就可以设置 COMP_CxCR 寄存器的 LOCK 位。这会导致大部分寄存器变成只读的，具体位参阅寄存器章节。

只能通过 MCU 复位来移除写保护。

16.3.5 COMP 防误触发功能

（1）当 COMP 和 ADC 同时使用时

当 COMP 的 P 或 N 端与 ADC 采样通道是同一个 IO 时，当外部接 RC 电路或 ADC 在通道转换时造成压差，此时 COMP 可能被 ADC 在通道转换时产生的压差影响而导致误触发。

当 COMP_CxCR 寄存器的 BAS_EN=1 时，根据 COMP_CxCR 寄存器的 INPSEL[1:0]和 COMP_CxCR 寄存器的 INMSEL[1:0]（x=1/2/3/4）的配置，在对应的 ADC 采样通道采样时对 COMP 输出进行遮蔽，这样可以防止由 ADC 的转换压差和 RC 电路造成的 COMP 误触发。

下两张表分别为 COMPx 的 P 端/N 端和 ADC 的 IO 对应关系。

表 16.3 COMPx_P 端和 ADC 的 IO 对应关系

CxCR_INPSEL	值	P 端输入	ADC 采样通道
C1CR_INPSEL	00	COMP1_P1	ADC1_IN8_H
	01	COMP1_P2	ADC1_IN7_H
	10	COMP1_P3	ADC1_IN6_H
	11	OPAMP1_OUT (1)	ADC1_IN4 ADC1_IN5 ADC1_IN10 ADC1_IN11 ADC1_IN12
C2CR_INPSEL	00	COMP2_P1	ADC1_IN9
	01	COMP2_P2	ADC2_IN3
	10	OPAMP3_OUT (1)	ADC2_IN4
	11	OPAMP4_OUT (1)	ADC2_IN1
C3CR_INPSEL	00	COMP3_P1	ADC2_IN8_H
	01	COMP3_P2	ADC2_IN6_H
	10	COMP3_P3	ADC2_IN7_H
	11	OPAMP2_OUT (1)	ADC2_IN5 ADC2_IN9 ADC2_IN10
C4CR_INPSEL	00	COMP4_P1	ADC2_IN11
	01	OPAMP2_OUT (1)	ADC2_IN5 ADC2_IN9 ADC2_IN10
	10	OPAMP1_OUT (1)	ADC1_IN4 ADC1_IN5 ADC1_IN10 ADC1_IN11 ADC1_IN12
	11	OPAMP3_OUT (1)	ADC2_IN4

(1) OPAMPx_OUT 输出内部连接至 COMPx 的正向输入，OPAMPx_OUT 输出和 OPAMPx_P/ N 输入相关联。

表 16.4 COMPx_N 端和 ADC 的 IO 对应关系

CxCR_INMSEL	值	N 端输入	ADC 采样通道
C1CR_INMSEL	00	COMP1_N1	ADC1_IN4
C2CR_INMSEL	00	COMP2_N1	ADC1_IN5
	11	COMP2_N2	ADC2_IN2
C3CR_INMSEL	00	COMP3_N1	ADC2_IN10
C4CR_INMSEL	00	COMP4_N1	ADC2_IN12
	11	COMP4_N2	ADC2_IN4

注：P 端选择OPAMP，N 端使用不会选择SRN；当N 端选择SRN 时，P 端只会选择P1/2/3。且ADC 采样也只会采对应的P 端（OPAMP 和SRN 不会同时使用）。

（2）当 COMP 切换引脚时

当 COMP 的 P 端或 N 端切换引脚会产生电压变化，此时 COMP 会被影响而导致误触发。

当需要切换引脚时，COMP 分别提供了硬件和软件两种遮蔽方式。

硬件遮蔽默认开启，软件遮蔽需要额外配置，具体操作如下：

硬件遮蔽：

- COMP 的通道切换默认硬件遮蔽。即 COMP_CxCR 寄存器的 CH_BLANK_EN 位默认为 1

软件遮蔽：

- 置位 COMP_CxCR 寄存器的 CH_DIS_BLANKING 位来打开遮蔽
- 置位 COMP_CxCR 寄存器的 CH_DIS 位来关闭 COMP 的 P 和 N 通道
- 切换 P 或 N 通道
- 清零 COMP_CxCR 寄存器的 CH_DIS 位来开启 COMP 的 P 和 N 通道
- 清零 COMP_CxCR 寄存器的 CH_DIS_BLANKING 位来关闭遮蔽

16.3.6 COMP 迟滞

该比较器具有可编程迟滞，可在有噪声信号时避免发生意外输出转换。它是对称的，既作用于比较器输出的下降沿，又作用于比较器输出的上升沿。迟滞可在不需要时（例如，退出低功耗模式时）禁止，以便使用外部组件强制迟滞值。

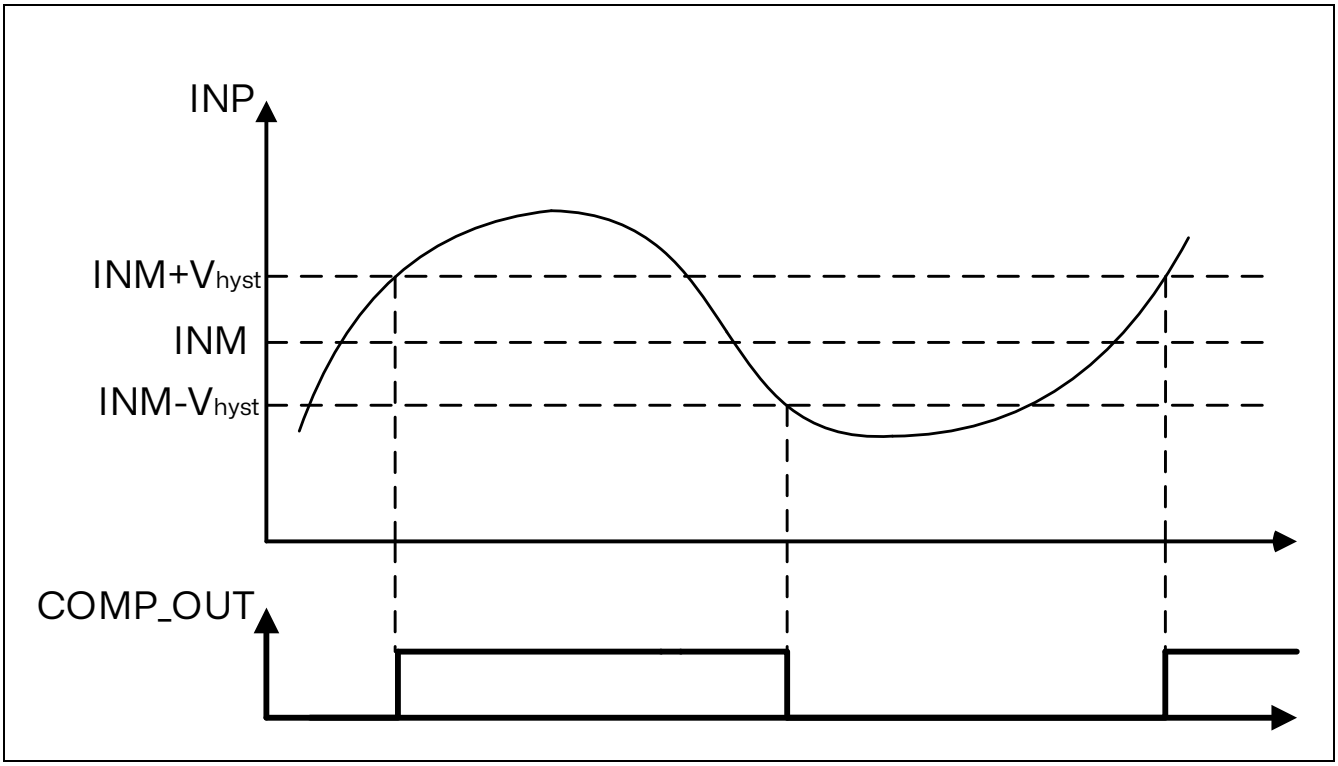


图 16.5 比较器迟滞

16.3.7 COMP 输出遮蔽功能

消除功能的目的是防止电流调节在 PWM 周期开始时（通常是功率开关反并联二极管中的恢复电流）出现短时尖峰时跳闸。 这里通过设置一个带有定时器输出比较信号的遮蔽窗来实现的。每个比较器通道有软件通过对应的 COMP_CxCR 寄存器的 BLANKING[2:0]位域单独选择遮蔽源，如下表：遮蔽源所示。

表 16.5 遮蔽源

BLANKSEL[2:0]	COMP1	COMP2	COMP3	COMP4
001	TIM1_OC5	TIM1_OC5	TIM1_OC5	TIM1_OC5
010	TIM1_OC4	TIM1_OC4	TIM1_OC4	TIM1_OC4
011	TIM2_OC1	TIM2_OC1	TIM2_OC1	TIM2_OC1
100	TIM8_OC5	TIM8_OC5	TIM8_OC5	TIM8_OC5
101	TIM8_OC4	TIM8_OC4	TIM8_OC4	TIM8_OC4
110	TIM15_OC1	TIM15_OC1	TIM15_OC1	TIM15_OC1
111	TIM15_OC2	TIM15_OC2	TIM15_OC2	TIM15_OC2

对遮蔽信号进行取反后与比较器级联输出进行逻辑与操作，以产生比较器通道 x 的输出。请参考下图中提供的示例。

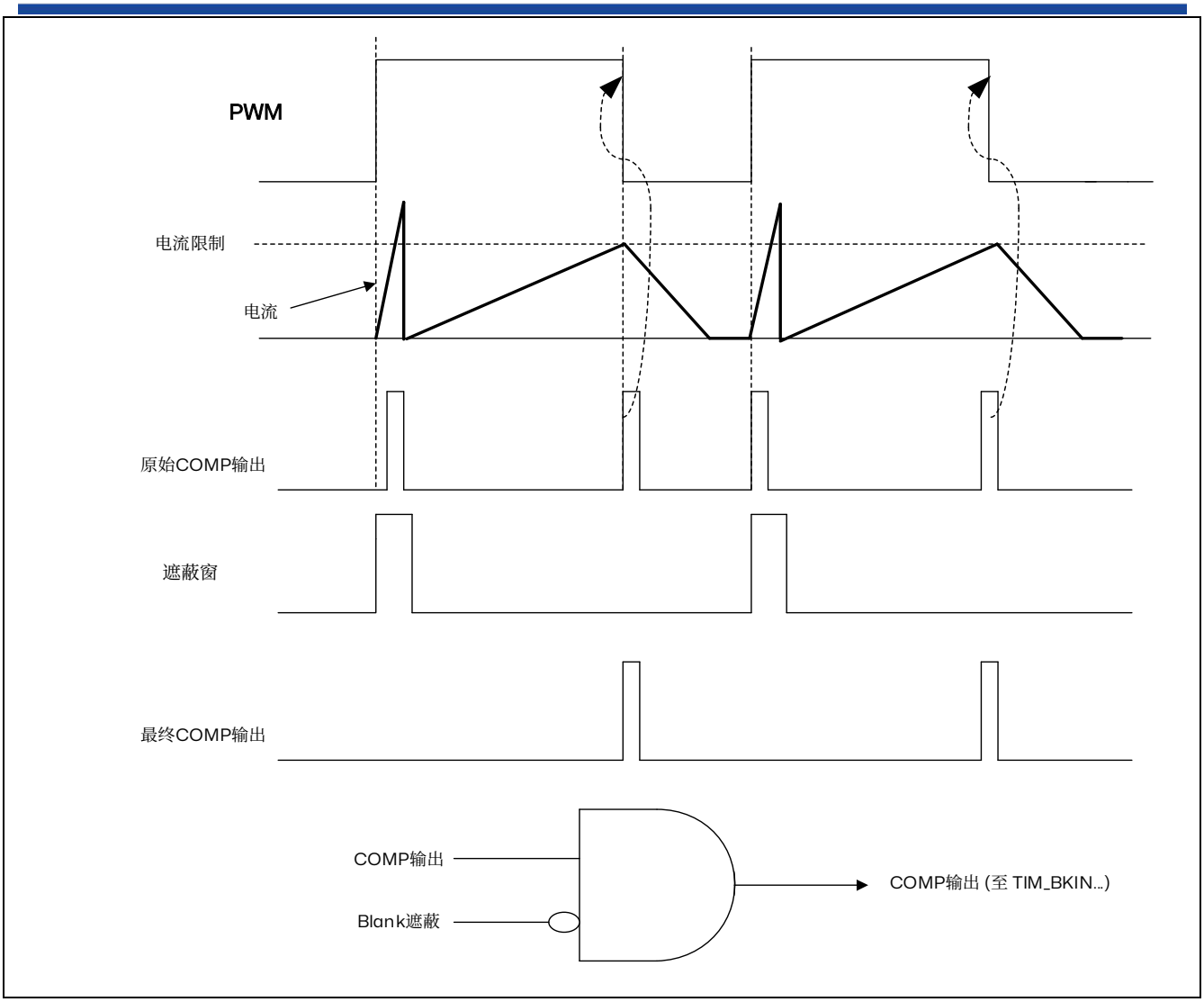


图 16.6 比较器输出遮蔽图

16.3.8 COMP 低功耗模式

表 16.6 比较器在低功耗模式下的行为

模式	描述
Sleep	对比较器没有影响。 比较器中断可使器件退出睡眠模式。
Stop	对比较器没有影响。 比较器中断可使器件退出停止模式。
Standby	导致比较器寄存器断电。 退出待机模式后必须重新初始化。

16.3.9 COMP 中断

比较器输出从内部连接到扩展中断和事件控制器。每个比较器都具有其各自的 EXTI 线，能够产生中断或事件，相同的机制用于退出睡眠和停止的低功耗模式。

要使能 COMPx 中断，需要遵循以下顺序：

1. 将 EXTI 线配置为中断模式，选择上升沿、下降沿或任一边沿有效，然后使能与 COMPx 输出事件对应的 EXTI 线。
2. 配置并使能映射到相应 EXTI 线的中断通道。
3. 使能 COMPx。

中断事件通过 COMP_CxCR 标志进行标记。

16.4 COMP 寄存器描述

16.4.1 比较器 x 控制和状态寄存器（COMP_CxCR）（x=1/2/3/4）

地址偏移：0x000+0x04*(x-1)

复位值：0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OUT	LIF	FIF	RIF	HIF	FHYST[1:0]		LIE	HIE	BLANKING[2:0]			CMP_BAS_EN_OPA	RHYST[1:0]	
rw	r	rc_w0	rc_w0	rc_w0	rc_w0	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
POL	OFLT[2:0]			CH_DIS_BLANKING	CH_DIS	BAS_EN	INPSEL[1:0]		Res	INMSEL[1:0]		FIE	RIE	CH_BLANK_EN	EN
rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

Bit 31 **LOCK：**COMP_CxCR 寄存器锁定

该位由软件设置和清除。此位设置后，除了 CH_BALNK_EN、INPSEL、POL、CH_DIS、CH_DIS_BLANKING、RIF、FIF、HIF、LIF、RIE、FIE、HIE、LIE 以外，CxCR[31:0]的其他位只能读（对应的 CAL 寄存器也会锁上，只有 APB2RSTR_SYSCFGRST 置 1 可解除 lock ）

0：解锁，除了 OUT 只读以外，CxCR[31:0]的其他位能读/写

1：锁定

Bit 30 **OUT：**比较器 x 输出状态

只读标志反映了在极性选择器和遮蔽之前的比较器 x 的输出电平

0：低输出

1：高输出

Bit 29 **LIF：**低电平输出中断事件标志位

0：无低电平输出中断事件

1：低电平输出中断事件发生

Bit 28 **FIF：**下降沿中断事件标志位

0：无下降沿中断事件

1：下降沿中断事件发生

Bit 27 **RIF：**上升沿中断事件标志位

0：无上升沿中断事件

1：上升沿中断事件发生

Bit 26 **HIF：**高电平输出中断事件标志位

0：无高电平输出中断事件

1：高电平输出中断事件发生

Bit 25:24 **FHYST[1:0]：**下降迟滞电压

00：0mv
01：20mv
10：40mv
11：60mv

Bit 23 **LIE**：低电平输出中断使能
0：禁止低电平输出中断
1：使能低电平输出中断

Bit 22 **HIE**：高电平输出中断使能
0：禁止高电平输出中断
1：使能高电平输出中断

Bit 21:19 **BLANKING[2:0]**：比较器 x 遮蔽源选择
此位域由软件控制，为比较器通道 x 选择遮蔽源，详见章节：COMP 输出遮蔽功能。
000：无遮蔽
001：TIM1 OC5 被选为遮蔽源
010：TIM1 OC4 被选为遮蔽源
011：TIM2 OC1 被选为遮蔽源
100：TIM8 OC5 被选为遮蔽源
101：TIM8 OC4 被选为遮蔽源
110：TIM15 OC1 被选为遮蔽源
111：TIM15 OC2 被选为遮蔽源

注：遮蔽需要屏蔽掉比较器输出 OUT *和中断事件* RIF、FIF

Bit 18 **CMP_BAS_EN_OPA**：OPAMP 通道 ADC 采样的遮蔽
0：OPAMP 通道遮蔽 ADC 采样禁止
1：OPAMP 通道遮蔽 ADC 采样使能

Bit 17:16 **RHYST[1:0]**：上升迟滞电压
00：0mv
01：20mv
10：40mv
11：60mv

Bit 15 **POL**：比较器输出极性
此位由软件选择控制比较器 x 的输出极性
0：同向输出
1：反向输出

Bit 14:12 **OFLT[2:0]**：输出滤波，连续的 PCLK 时钟比较输出不变则认为有效，否则保持不变
000：无滤波　　100：512 个时钟周期
001：32 个时钟周期　　101：1024 个时钟周期
010：128 个时钟周期　　110：2048 个时钟周期
011：256 个时钟周期　　111：4096 个时钟周期

*注：*COMP *输出滤波没有作用于外部中断*，详情请参见 COMP 框图

Bit 11 **CH_DIS_BLANKING**：切换通道前进行 BLANKING 遮蔽
0：通道切换前不开启遮蔽
1：通道切换前开启遮蔽

Bit 10 **CH_DIS**：P 端或 N 端通道切换时软件主动关闭通道

0：通道切换时不主动关闭通道

1：通道切换时软件主动关闭通道，避免切换引脚造成 COMP 误触发

Bit 9 **BAS_EN**：ADC 采样的遮蔽使能

0：禁止遮蔽 ADC 采样

1：使能遮蔽 ADC 采样（参见章节：COMP 防误触发功能）

Bit 8:7 **INPSEL[1:0]**：比较器 x 正向输入信号选择

此位由软件控制来选择比较器 x 的正向输入 COMPx_INP 的信号，详情参见表：

COMPx 正向输入分配

00：COMPx_P1

01：COMPx_P2，OPAMP2_out(COMP4)

10：COMPx_P3(COMP1/3)，OPAMP3_out(COMP2)，OPAMP1_out(COMP4)

11：OPAMP1_out(COMP1)，OPAMP4_out(COMP2)

OPAMP2_out(COMP3)，OPAMP3_out(COMP4)

Bit 6 保留，必须保持为复位值。

Bit 5:4 **INMSEL[1:0]**：比较器 x 反向输入信号选择

此位域由软件控制来选择比较器 x 的反向输入 COMPx_INM 的信号，详情参见表：

COMPx 反向输入分配

00：COMPx_N1

01：DAC1_CH1

10：DAC2_CH1

11：COMPx_N2(COMP2/4)，COMPx_SRN_Center(COMP1/3)

注：当选择 DACx_CH1 *作为* COMPx *的* N *端使用时，为防止开关噪声，需先使能* DACx，*再将* COMPx *使能。*

Bit 3 **FIE**：下降沿中断使能

0：禁止下降沿中断

1：使能下降沿中断

Bit 2 **RIE**：上升沿中断使能

0：禁止上升沿中断

1：使能上升沿中断

Bit 1 **CH_BLANK_EN**：硬件切换 P 端和 N 端通道遮蔽

由硬件默认为 1 开启 P 端和 N 端通道切换遮蔽，可软件关闭。

具体参考章节：比较器防误触发功能。

0：关闭硬件遮蔽

1：使能硬件遮蔽

Bit 0 **EN**：比较器 x 使能

0：禁止

1：使能

注：比较器的时钟由 RCC_APB2ENR *的* SYSCFGEN *位控制。*

16.4.2 比较器控制寄存器 1（COMP_CR1）

地址偏移：0x080

复位值：0x0000 0808

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										SRN2_Dout	SRN2_EN	Res	SRN_LOCK	SRN1_Dout	SRN1_EN
										rw	rw		rw	rw	rw

Bit 31:6 保留，必须保持为复位值。

Bit 5 **SRN2_Dout**：COMP3 星型电阻网络 2 输出使能位

0：COMP3 星型电阻网络 2 不可通过 GPIO 测量

1：COMP3 星型电阻网络 2 可通过 GPIO 测量

注：通过 GPIO *测量必须置位* CR1_SRN2_EN

Bit 4 **SRN2_EN**：COMP3 星型电阻网络 2 使能位

0：COMP3_P1/P2/P3 不接入星型电阻网络 2

1：COMP3_P1/P2/P3 接入星型电阻网络 2

Bit 3 保留，必须保持为复位值。

Bit 2 **SRN_LOCK**：COMP_CR1 寄存器锁定

0：解锁，CR1 的所有位能读/写

1：锁定，CR1 的所有位只读

Bit 1 **SRN1_Dout**：COMP1 星型电阻网络 1 输出使能位

0：COMP1 星型电阻网络 1 不可通过 GPIO 测量

1：COMP1 星型电阻网络 1 可通过 GPIO 测量

注：通过 GPIO *测量必须置位* CR1_SRN1_EN

Bit 0 **SRN1_EN**：COMP1 星型电阻网络 1 使能位

0：COMP1_P1/P2/P3 不接入星型电阻网络 1

1：COMP1_P1/P2/P3 接入星型电阻网络 1

17 运算放大器（OPAMP）

17.1 简介

该设备内置了四个运算放大器，每个放大器具有两个输入端和一个输出端。这三个输入/输出端（I/O）可以连接到外部引脚，从而实现任何类型的外部互连。运算放大器可以在内部配置为跟随器、非反相增益范围为 4 至 16 的放大器。

17.2 OPAMP 主要特性

- 轨对轨输入电压范围
- 低输入偏置电流
- 低输入偏置电压
- 高频增益带宽
- 低功耗模式，以实现更低的运行功耗

17.3 OPAMP 功能描述

OPAMP 具有多种模式。
每个 OPAMP 可以单独使能，当禁用时，其输出为高阻抗状态。
使能后，OPAMP 处于功能模式。
在功能模式下，OPAMP 的输入和输出按照章节：信号路由中的描述进行连接。

17.3.1 OPAMP 输出重定向至内部 ADC/COMP 通道

通过设置 OPAMPx_CSR 寄存器中的 TOINT_EN 位，可以将运算放大器的输出在内部重定向至一个 ADC/COMP 通道。在这种情况下，原本映射给特定 OPAMPx_OUT 输出的 GPIO 引脚将被释放，可以用于其他用途。ADC 可以内部测量 OPAMPx_OUT 的电压。COMP 可以内部选择 OPAMPx_OUT 输入。请参阅下表：运算放大器的可能连接方式，了解 OPAMPx 内部输出与 ADC/COMP 通道之间的分配关系。

17.3.2 OPAMP 复位和时钟

由时钟控制器提供的 OPAMP 时钟与 PCLK2（APB2 时钟）同步。在 RCC 控制器中没有提供时钟使能控制位。要使用 OPAMP 的时钟源，必须在 RCC 控制器中设置 SYSCFG 时钟使能控制位。

OPAEN 位用于使能和禁用 OPAMP 的操作。为了避免对输出产生伪影效应，必须在使能 OPAEN 位之前更改 OPAMP 的寄存器配置。

当不再需要 OPAMP 的输出时，可以禁用 OPAMP 以节省功率。在 OPAMP 被禁用期间，之前设置的所有配置都将保持不变。

17.3.3 初始化配置

OPAMP 的默认配置为功能模式，其中三个输入/输出端连接到外部引脚。默认配置使用正常模式，提供标准性能。可以通过设置 OPALPM 位，将 OPAMP 切换到低功耗模式，以获得更低的运行功耗。正常模式和低功耗模式的特性均在数据手册中定义。

一旦设置 OPAMPx_CSR 寄存器中的 OPAEN 位，OPAMP 即开始工作。两个输入引脚和一个输出引脚的连接方式如章节：信号路由中所述，且默认连接设置可以更改。

*注：在对应的*GPIOx_MODER *寄存器中，输入和输出引脚必须配置为模拟模式（默认状态）。*

17.3.4 信号路由

运算放大器引脚的路由由 OPAMPx_CSR 和 OPAMPx_CR2 寄存器决定。
下表描述了四个运算放大器（OPAMPx，x = 1...4）的连接情况。

表 17.1 运算放大器的可能连接方式

信号	引脚	内部的	描述
OPAMP1_VINM	PB9 (VINM_N1) PA8 (VINM_N2)	OPAMP1_OUT 或 PGA	由位 GAIN 和 VM_SEL 控制
OPAMP1_VINP	PB8 (VINP_P1) PA7 (VINP_P2)	外接偏置 V_{REFINT}/2 或 V_{ref}/2 或 GND	引脚由位 VP_SEL 控制，偏置由位 VOL_SEL 和 VOL_EN 控制
OPAMP1_OUT	PA3	ADC1_IN13 COMP1_INP COMP4_INP(1)	当 OPAMP 使能且 OPAMP 内部输出禁用时，该引脚被连接。 ADC 输入由 ADC 控制。 COMP 输入由 COMP 控制。
OPAMP2_VINM	PB0 (VINM_N1)	OPAMP2_OUT 或 PGA	由位 GAIN 和 VM_SEL 控制
OPAMP2_VINP	PB1 (VINP_P1)	外接偏置 V_{REFINT}/2 或 V_{ref}/2 或 GND	引脚由位 VP_SEL 控制，偏置由位 VOL_SEL 和 VOL_EN 控制
OPAMP2_OUT	PC3	ADC2_IN14 COMP3_INP COMP4_INP(1)	当 OPAMP 使能且 OPAMP 内部输出禁用时，该引脚被连接。 ADC 输入由 ADC 控制。 COMP 输入由 COMP 控制。
OPAMP3_VINM	PA13 (VINM_N1)	OPAMP3_OUT 或 PGA	由位 GAIN 和 VM_SEL 控制
OPAMP3_VINP	PA12(VINP_P1)	外接偏置 V_{REFINT}/2 或 V_{ref}/2 或 GND	引脚由位 VP_SEL 控制，偏置由位 VOL_SEL 和 VOL_EN 控制
OPAMP3_OUT	PB5	ADC2_IN13 COMP2_INP COMP4_INP(1)	当 OPAMP 使能且 OPAMP 内部输出禁用时，该引脚被连接。 ADC 输入由 ADC 控制。 COMP 输入由 COMP 控制。
OPAMP4_VINM	PB7 (VINM_N1)	OPAMP4_OUT 或 PGA	由位 GAIN 和 VM_SEL 控制
OPAMP4_VINP	PB6 (VINP_P1)	外接偏置 V_{REFINT}/2 或 V_{ref}/2 或 GND	引脚由位 VP_SEL 控制，偏置由位 VOL_SEL 和 VOL_EN 控制
OPAMP4_OUT	PB2	ADC1_IN14 COMP2_INP(1)	当 OPAMP 使能且 OPAMP 内部输出禁用时，该引脚被连接。 ADC 输入由 ADC 控制。 COMP 输入由 COMP 控制。

1. 当设置 TOINT_EN 位时，此 ADCx 通道和 COMPx_INP 在内部连接到 OPAMPx_OUT。在这种情况下，原本用于 OPAMPx_OUT 的 I/O 可以用于其他用途。

17.3.5 OPAMP 模式

运算放大器的输入和输出端均可通过引脚访问。这些放大器可在多种配置环境中使用：

- 独立模式（外部增益设置模式）
- 电压跟随器模式
- PGA 模式

注：必须将信号阻抗保持在一定水平以下，以避免输入泄漏产生明显伪影（由于源端的电阻压降）。有关详细信息，请参阅数据手册中的电气特性部分。

独立模式（外部增益设置模式）

以下是配置 OPAMP 作为独立模式使用的**必要步骤**：

1. 独立模式 P 端选择（仅 OPAMP1 可选，OPAMP2/3/4 默认为 VINP_P1）：
   - VINP_P1：OPAMP1_CSR 寄存器 VP_SEL=0
   - VINP_P2：OPAMP1_CSR 寄存器 VP_SEL=1
2. 独立模式 N 端选择：
   - VINM_N1：OPAMPx_CSR(x=1/2/3/4)寄存器 VM_SEL[1:0]=00
   - VINM_N2：OPAMP1_CSR 寄存器 VM_SEL[1:0]=11
3. OPAMP 输出选择：
   - 输出至外部 GPIO：OPAMPx_CSR 寄存器 TOEXT_EN=1
4. OPAMP 使能：
   - OPAMPx_CSR 寄存器 OPAEN=1

注：可将OPAHSM 设置为“运算放大器低功耗”模式，以获得更低的功耗表现。

注：下图中示例的独立模式设置的外部电阻增益 = (R1+R2)/R1

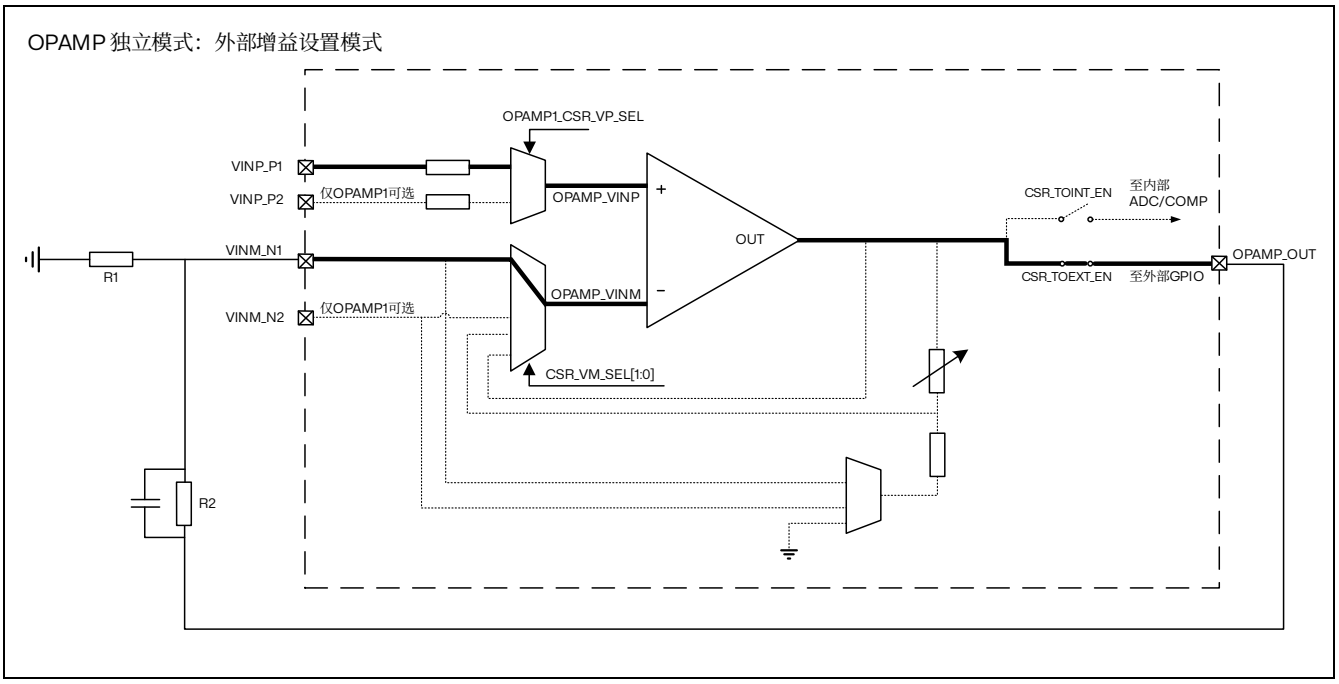


图 17.1 独立模式：外部增益设置模式

电压跟随器模式配置方式

以下是配置 OPAMP 作为电压跟随器模式使用的**必要步骤**：

1. OPAMPx 模式选择：
   - 电压跟随器模式：OPAMPx_CSR 寄存器 VM_SEL[1:0]=10
2. 电压跟随器模式 P 端选择（仅 OPAMP1 可选，OPAMP2/3/4 默认为 VINP_P1）：
   - VINP_P1：OPAMP1_CSR 寄存器 VP_SEL=0
   - VINP_P2：OPAMP1_CSR 寄存器 VP_SEL=1

*注：与*OPAMPx_VINM *对应的引脚可用于其他用途。*

3. OPAMP 输出选择：
   - 输出至内部 ADC/COMP：OPAMPx_CSR 寄存器 TOINT_EN=1
   - 输出至外部 GPIO：OPAMPx_CSR 寄存器 TOEXT_EN=1

*注：若选择输出至内部*ADC/COMP *通道，外部映射有*OPAMPx_OUT *的*I/O *口空闲，可用于其他用途。*

4. OPAMP 使能：
   - OPAMPx_CSR 寄存器 OPAEN=1

*注：一旦设置*OPAEN *位，*OPAMPx_VINP *引脚上的电压就会被缓冲至*OPAMPx_OUT *引脚。*

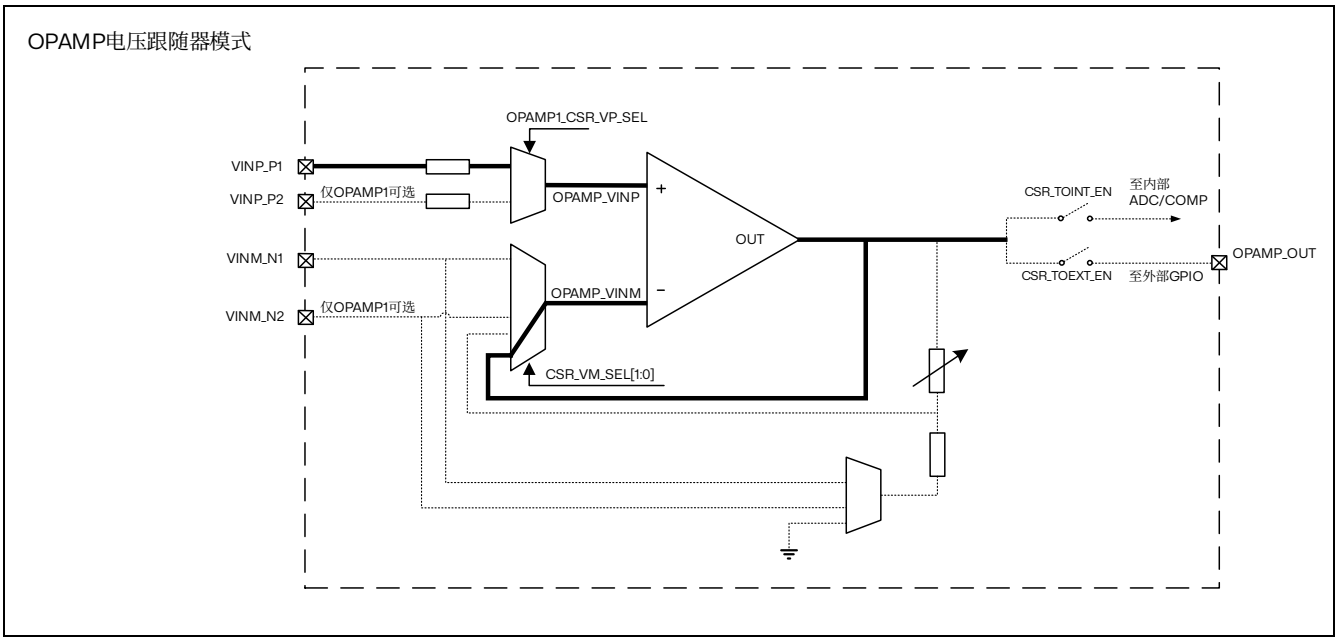


图 17.2 电压跟随器模式

PGA 模式

以下是配置 OPAMP 作为 PGA 模式使用的**必要步骤**:

1. OPAMPx 模式选择:
   - PGA 模式：OPAMPx_CSR 寄存器 VM_SEL[1:0]=01
2. PGA 模式 P 端选择（仅 OPAMP1 可选，OPAMP2/3/4 默认为 VINP_P1）:
   - VINP_P1：OPAMP1_CSR 寄存器 VP_SEL=0
   - VINP_P2：OPAMP1_CSR 寄存器 VP_SEL=1
3. PGA 模式 P 端内部偏置电压选择（先将 OPAMPx_CR2 寄存器 VOL_EN=0，GND_EN=0）:
   - 偏置 $V_{REFINT}/2$：OPAMPx_CR2 寄存器 VOL_EN=1，VOL_SEL=0，VREFINT_EN=1
   - 偏置 $V_{ref}/2$：OPAMPx_CR2 寄存器 VOL_EN=1，VOL_SEL=1
   - 偏置 GND：OPAMPx_CR2 寄存器 GND_EN=1

   注：V_{REFINT} *为内部稳定的带隙 1V 电压输出。*

   注：$V_{ref}/2$ *由 VREFBUF 配置，可选连接到外部 VREF+引脚，或连接内部 2.0V/2.5V/2.8V。*

   注：若$V_{ref}/2$ *选择连接内部*2.0V/2.5V/2.8V*，需要置位*VREFBUF_CSR *寄存器的*VCM *位以使能*$V_{REFBUF}/2$ *缓冲器，详见章节*25.4.1 VREFBUF *控制和状态寄存器（*VREFBUF_CSR*）。*

   *注：*OPAMP1 *和*OPAMP4 *共用一个偏置电压，*VOL_SEL *位在*OPAMP1_CR2 *寄存器设置。*

   *注：*OPAMP2 *和*OPAMP3 *共用一个偏置电压，*VOL_SEL *位在*OPAMP2_CR2 *寄存器设置。*

4. PGA 模式 N 端选择:
   - 内部接地（单端）：OPAMPx_CSR 寄存器 OUTCONNECT[1:0]=01
   - VINM_N1（差分）：OPAMPx_CSR(x=1/2/3/4)寄存器 OUTCONNECT[1:0]=10
   - VINM_N2（差分）：OPAMP1_CSR 寄存器 OUTCONNECT[1:0]=11
5. PGA 模式增益放大倍数选择:
   - 4X：OPAMPx_CSR 寄存器 GAIN[1:0]=00
   - 8X：OPAMPx_CSR 寄存器 GAIN[1:0]=01
   - 12X：OPAMPx_CSR 寄存器 GAIN[1:0]=10
   - 16X：OPAMPx_CSR 寄存器 GAIN[1:0]=11

   *注：为避免饱和，输入电压必须保持在*VDDA *除以所选增益所得的电压值以下。*

6. OPAMP 输出选择:
   - 输出至内部 ADC/COMP：OPAMPx_CSR 寄存器 TOINT_EN=1
   - 输出至外部 GPIO：OPAMPx_CSR 寄存器 TOEXT_EN=1
7. OPAMP 使能:
   - OPAMPx_CSR 寄存器 OPAEN=1

   *注：一旦设置*OPAEN *位，*OPAMPx_VINP *引脚上的电压就会按所选增益进行放大。*

   *注：*PGA *模式输出 =（*VINP - VINM）* GAIN + *偏置电压*

例 1：PGA 模式下 N 端选择内部接地，此时 P 端作为单端输入，偏置 GND。

– PGA 模式输出 =（VINP - 0）* GAIN + 0 = VINP * GAIN

例 2：PGA 模式下 N 端选择 VINM_N1，此时 P 端 N 端作为差分输入，偏置 $V_{REFINT}/2$。

– PGA 模式输出 =（VINP - VINM）* GAIN + $V_{REFINT}/2$

OPAMP PGA模式：**例1 N端选择内部接地，此时P端作为单端输入，偏置GND。**

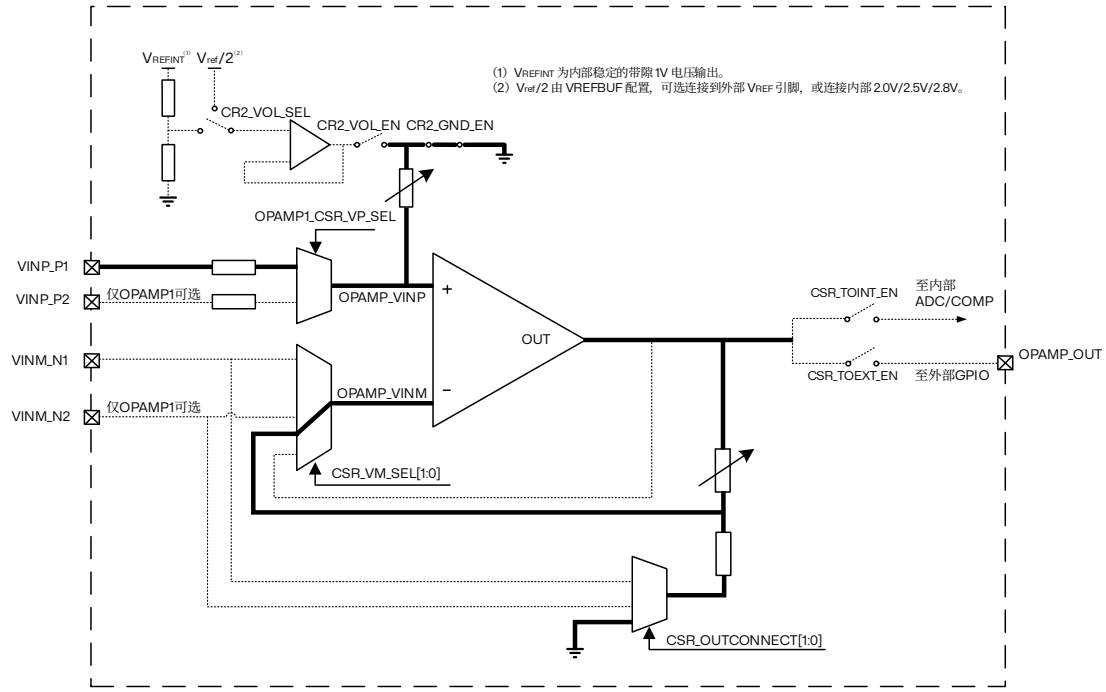


图 17.3 PGA 模式单端输入（P 端无偏置电压）

OPAMP PGA模式：**例2 N端选择VINM_N1，此时P端N端作为差分输入，偏置$V_{REFINT}/2$。**

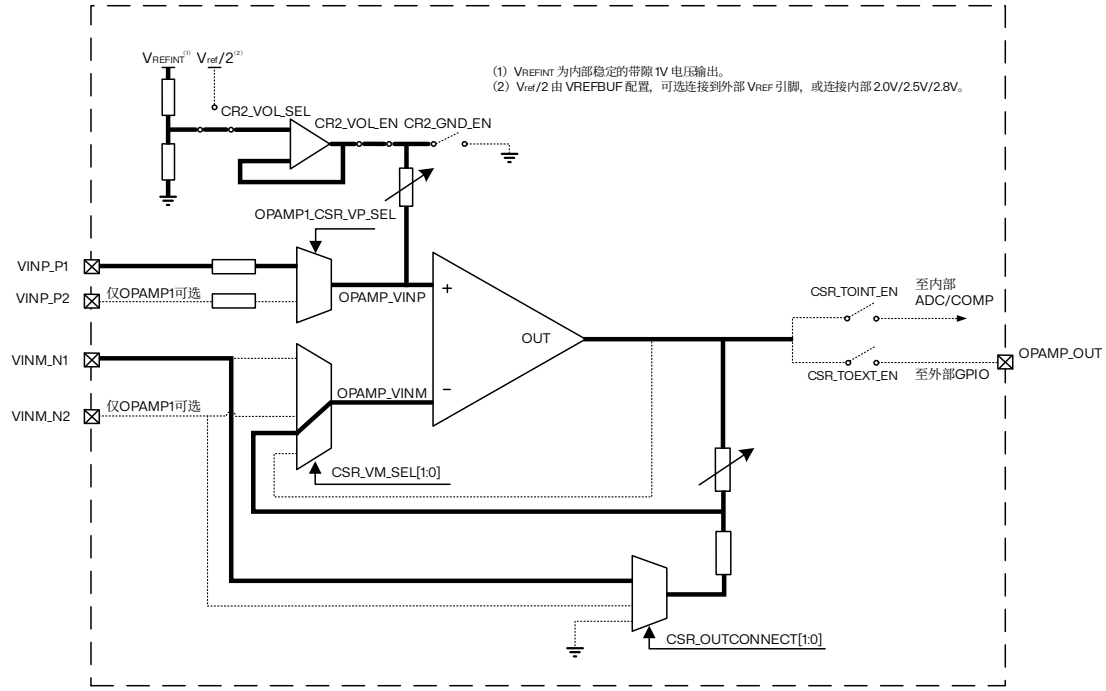


图 17.4 PGA 模式差分输入（P 端接偏置电压）

17.3.6 OPAMP 低功耗模式

表 17.2 低功耗模式对 OPAMP 的影响

模式	描述
Sleep	无影响。
Stop	无影响，OPAMP 寄存器内容保留。
Standby	OPAMP 寄存器断电，必须在退出待机模式后重新初始化。

17.4 OPAMP 寄存器描述

17.4.1 OPAMP1 控制状态寄存器（OPAMP1_CSR）

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
LOCK	Res	TOINT_EN	Reserved										OUTCONNECT[1:0]		Res
rw		rw											rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GAIN[1:0]		Reserved				VP_SEL	TOEXT_EN	OPA LPM	Reserved			VM_SEL[1:0]		Res	OPAEN
rw	rw					rw	rw	rw				rw	rw		rw

Bit 31 **LOCK**：OPAMP1_CSR 和 OPAMP1_CR2 锁定

此位为一次性写入位。由软件设置。只能通过系统复位清除。

此位用于将 OPAMP1_CSR 和 OPAMP1_CR2 寄存器配置为只读。

0：OPAMP1_CSR 和 OPAMP1_CR2 可读写

1：OPAMP1_CSR 和 OPAMP1_CR2 只读

注：通过 RCC_APB2RSTR *寄存器的* SYSCFGRST *位写入* 1 *解除* Lock *状态。*

Bit 30 保留，必须保持为复位值。

Bit 29 **TOINT_EN**：OPAMP1 内部输出使能

0：OPAMPx 输出至内部 ADC/COMP 禁止

1：OPAMPx 输出至内部 ADC/COMP 使能

Bit 28:19 保留，必须保持为复位值。

Bit 18:17 **OUTCONNECT[1:0]**：PGA 模式下 OPAMP1 的输出连接

00：不连接

01：OPAMP1 的输出经过内部电阻接地

10：OPAMP1 的输出经过内部电阻接到 OPAMP1_N1

11：OPAMP1 的输出经过内部电阻接到 OPAMP1_N2

注：仅 OPAMP1 *可选“*OPAMP1_N2*”，*OPAMP2/3/4 *只有一个* N *端输入。*

Bit 16 保留，必须保持为复位值。

Bit 15:14 **GAIN[1:0]：**OPAMP1 PGA 模式放大倍率选择

00：4x

01：8x

10：12x

11：16x

Bit 13:10 保留，必须保持为复位值。

Bit 9 **VP_SEL：**非反相输入选择

0：OPAMP1_P1

1：OPAMP1_P2

*注：仅*OPAMP1 *可选“*OPAMP1_P2*”，*OPAMP2/3/4 *只有一个*P *端输入。*

Bit 8 **TOEXT_EN：**OPAMP1 输出至外部 IO 引脚使能

0：OPAMP1 输出至外部 IO 引脚禁止

1：OPAMP1 输出至外部 IO 引脚使能

Bit 7 **OPALPM：**OPAMP1 低功耗模式使能

0：正常模式

1：低功耗模式

*注：若*OPAMP *工作在低功耗模式（*OPAMPx_CSR *寄存器*OPALPM=1*）下，对内部* OPAMP *通道（*ADC *通道*13/14*）采样时，采样时间至少使用*17.5 *周期及以上档位（速度*≤2Msps *采样）。详见章节：*14.3.4 ADC *通道连接。*

Bit 6:4 保留，必须保持为复位值。

Bit 3:2 **VM_SEL[1:0]：**反相输入选择

00：OPAMP1_N1 引脚连接到 OPAMP VINM 输入（独立模式）

01：反馈电阻连接到 OPAMP 的 VINM 输入（PGA 模式），PGA 模式放大倍率取决于 GAIN[1:0]的设置

10：OPAMP_OUT 连接到 OPAMP VINM 输入（电压跟随器模式）

11：OPAMP1_N2 引脚连接到 OPAMP VINM 输入（独立模式）

*注：*PGA *模式时，*OPAMP1 *的反相输入选择由*OUTCONNECT[1:0]*决定。*

*注：仅*OPAMP1 *可选“*OPAMP1_N2*”，*OPAMP2/3/4 *只有一个*N *端输入。*

Bit 1 保留，必须保持为复位值。

Bit 0 **OPAEN：**OPAMP1 使能

0：OPAMP1 禁止

1：OPAMP1 使能

17.4.2 OPAMPx 控制状态寄存器（OPAMPx_CSR）（x=2...4）

地址偏移：0x00 + 0x04*(x-1)

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GAIN[1:0]		Reserved					TOEXT_EN	OPALPM	Reserved			VM_SEL[1:0]		Res	OPAEN
rw	rw						rw	rw				rw	rw		rw

Bit 31 **LOCK**：OPAMPx_CSR 和 OPAMPx_CR2 锁定

此位为一次性写入位。由软件设置。只能通过系统复位清除。

此位用于将 OPAMPx_CSR 和 OPAMPx_CR2 寄存器配置为只读。

0：OPAMPx_CSR 和 OPAMPx_CR2 可读写

1：OPAMPx_CSR 和 OPAMPx_CR2 只读

注：通过 RCC_APB2RSTR *寄存器的* SYSCFGRST *位写入* 1 *解除* Lock *状态。*

Bit 30 保留，必须保持为复位值。

Bit 29 **TOINT_EN**：OPAMPx 内部输出使能

0：OPAMPx 输出至内部 ADC/COMP 禁止

1：OPAMPx 输出至内部 ADC/COMP 使能

Bit 28:19 保留，必须保持为复位值。

Bit 18:17 **OUTCONNECT[1:0]**：PGA 模式下 OPAMP2/3/4 的输出连接

00：不连接

01：OPAMPx 的输出经过内部电阻接地

10：OPAMPx 的输出经过内部电阻接到 OPAMPx_N1

11：保留

注：仅 OPAMP1 *可选“*OPAMP1_N2*”，* OPAMP2/3/4 *只有一个* N *端输入。*

Bit 16 保留，必须保持为复位值。

Bit 15:14 **GAIN[1:0]**：OPAMPx PGA 模式放大倍率选择

00：4x

01：8x

10：12x

11：16x

Bit 13:9 保留，必须保持为复位值。

Bit 8 **TOEXT_EN**：OPAMPx 输出至外部 IO 引脚使能

0：OPAMPx 输出至外部 IO 引脚禁止

1：OPAMPx 输出至外部 IO 引脚使能

Bit 7 **OPALPM**：OPAMPx 低功耗模式使能

0：正常模式

1：低功耗模式

*注：若*OPAMP *工作在低功耗模式（*OPAMPx_CSR *寄存器*OPALPM=1*）下，对内部* OPAMP *通道（*ADC *通道*13/14*）采样时，采样时间至少使用*17.5 *周期及以上档位（速度*≤2Msps *采样）。详见章节：*14.3.4 ADC *通道连接。*

Bit 6:4 保留，必须保持为复位值。

Bit 3:2 **VM_SEL[1:0]**：反相输入选择

00：OPAMPx_N1 引脚连接到 OPAMP VINM 输入（独立模式）

01：反馈电阻连接到 OPAMP 的 VINM 输入（PGA 模式），PGA 模式放大倍率取决于 GAIN[1:0]的设置

10：OPAMP_OUT 连接到 OPAMP VINM 输入（电压跟随器模式）

11：保留

*注：*PGA *模式时，*OPAMP2/3/4 *的反相输入选择由*OUTCONNECT[1:0]*决定。*

*注：仅*OPAMP1 *可选“*OPAMP1_N2*”，*OPAMP2/3/4 *只有一个*N *端输入。*

Bit 1 保留，必须保持为复位值。

Bit 0 **OPAEN**：OPAMPx 使能

0：OPAMPx 禁止

1：OPAMPx 使能

17.4.3 OPAMPx 控制寄存器 2（OPAMPx_CR2）（x=1…4）

地址偏移：0x10 + 0x04*(x-1)

复位值：0x0000 0000

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

Bit 31:4 保留，必须保持为复位值。

Bit 3 **VREFINT_EN**：V_{REFINT} 使能信号

0：禁止 V_{REFINT}

1：使能 V_{REFINT}

Bit 2 **VOL_SEL**：OPAMP1/2 P 端输入外接偏置电压选择

0：偏置电压选择为 $V_{REFINT}/2$

1：偏置电压选择为 $V_{ref}/2$

注：当偏置电压选择为 $V_{REFINT}/2$，V_{REFINT} 为内部稳定的带隙 1V 电压输出。需要置位 OPAMPx_CR2 寄存器的 VREFINT_EN 位来使能 V_{REFINT} 信号。

注：当偏置电压选择为 $V_{ref}/2$ 时，V_{ref} 电压由 VREFBUF 配置来选择，可选择连接到外部 V_{REF} 引脚，或选择连接内部参考电压缓冲器 2.0V/2.5V/2.8V。详见章节：参考电压缓冲器（VREFBUF）。

注：当 V_{ref} 电压选择连接内部参考电压缓冲器 V_{REFBUF} 的 2.0V/2.5V/2.8V，需要置位 VREFBUF_CSR 寄存器的 VCM 位来使能 $V_{REFBUF}/2$ 缓冲器。详见章节：参考电压缓冲器（VREFBUF）。

注：OPAMP4 和 OPAMP1 共用一个偏置，OPAMP3 和 OPAMP2 共用一个偏置。

注：仅 OPAMP1 和 OPAMP2 具有此位。

Bit 1 **GND_EN**：OPAMPx P 端输入外接偏置 GND 使能

0：禁止 P 端输入外接偏置 GND

1：使能 P 端输入外接偏置 GND

注：使能此位前先将 OPAMPx_CR2_VOL_EN 和 OPAMPx_CR2_GND_EN 清 0。

注：禁止同时使能 OPAMPx_CR2_VOL_EN。

Bit 0 **VOL_EN**：OPAMPx P 端输入外接偏置电压使能

0：禁止 P 端输入外接偏置电压

1：使能 P 端输入外接偏置电压

注：使能此位前先将 OPAMPx_CR2_VOL_EN 和 OPAMPx_CR2_GND_EN 清 0。

注：禁止同时使能 OPAMPx_CR2_GND_EN。

18 高级控制定时器（TIM1/TIM8）

18.1 TIM1/TIM8 简介

高级控制定时器（TIM1/TIM8）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，包括产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM）。

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

高级控制定时器（TIM1/TIM8）和通用定时器（TIMy）是完全独立的，它们不共享任何资源。它们可以同步操作。

18.2 TIM1/TIM8 主要特征

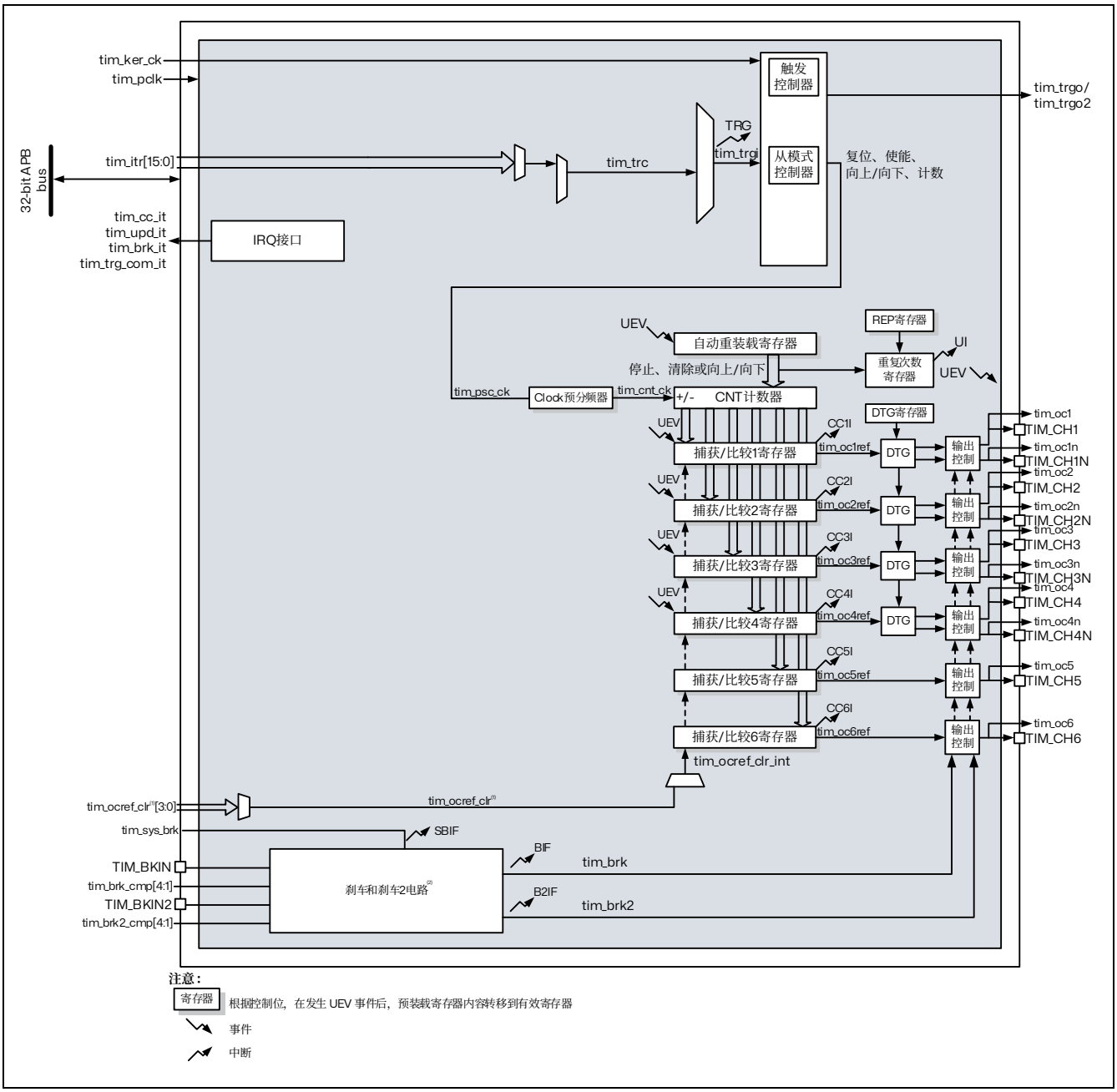
TIM1/TIM8 定时器的功能包括：

- 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 6 个独立通道：
  - 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
- 用于将定时器输出信号置于安全的用户选择配置的两路刹车输入信号
- 如下事件发生时产生中断请求：
  - 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）
  - 输出比较
- 触发输入作为外部时钟或者按周期的电流管理
- 非对称模式

18.3 TIM1/TIM8 功能描述

18.3.1 框图

图 18.1 高级控制定时器框图



1. 这一特性详见下一节：TIM1/TIM8 引脚和内部信号
2. 参见刹车和刹车 2 电路以获得更多细节

18.3.2 TIM1/TIM8 引脚和内部信号

本章节中的表格总结了 TIM 的输入和输出。

表 18.1 TIM 输入/输出引脚

引脚名称	信号类型	描述
TIM_CH1 TIM_CH2 TIM_CH3 TIM_CH4	输入/输出	定时器多用途通道。 每个通道都可以用于比较或 PWM（脉冲宽度调制）。
TIM_CH1N TIM_CH2N TIM_CH3N TIM_CH4N	输出	由 TIM_CHx 输出派生的定时器的互补输出， 具有插入死区时间的能力。
TIM_BKIN TIM_BKIN2	输入/输出	刹车和刹车 2 输入。这些输入也可以配置为双向模式。

表 18.2 TIM 内部输入/输出信号

内部信号名称	信号类型	描述
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或作为输入时钟(tim_ker_ck 时钟的 1/4 以下)。
tim_trgo/tim_trgo2	输出	内部触发器输出。这些触发器被其他定时器或其他外设使用。
tim_ocref_clr[3:0]	输入	定时器 tim_ocref_clr 输入总线。这些输入可以用来清除 tim_ocxref 信号，通常用于硬件的逐周期脉冲宽度控制。
tim_brk_cmp[4:1]	输入	内部信号的刹车输入。
tim_brk2_cmp[4:1]	输入	内部信号的刹车 2 输入。
tim_sys_brk[n:0]	输入	系统刹车输入。该输入收集 MCU 的系统级错误。
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟。
tim_cc_it	输出	定时器捕获/比较中断。
tim_upd_it	输出	定时器更新事件中断。
tim_brk_it	输出	定时器刹车和刹车 2 中断。
tim_trg_com_it	输出	定时器触发和换向中断。

下表列出连接到 tim_itr 输入多路复用器的内部源。

表 18.3 TIMx 内部触发连接

tim_itrx 输入	TIM1	TIM8
tim_itr0	tim2_trgo	tim2_trgo
tim_itr1	tim3_trgo	tim3_trgo
tim_itr2	tim15_trgo	tim15_trgo
tim_itr3	tim8_trgo	tim1_trgo
tim_itr4	timA_trgo	timA_trgo
tim_itr5	timB_trgo	timB_trgo
tim_itr[15:6]	保留	

下三张表列出了连接到 tim_brk 和 tim_brk2 输入的源。

表 18.4 定时器刹车互连

tim_brk 输入	TIM1	TIM8
TIM_BKIN	TIM1_BKIN pin	TIM8_BKIN pin
tim_brk_cmp1	comp1_out	comp1_out
tim_brk_cmp2	comp2_out	comp2_out
tim_brk_cmp3	comp3_out	comp3_out
tim_brk_cmp4	comp4_out	comp4_out

表 18.5 定时器刹车 2 互连

tim_brk2 输入	TIM1	TIM8
TIM_BKIN2	TIM1_BKIN2 pin	TIM8_BKIN2 pin
tim_brk2_cmp1	comp1_out	comp1_out
tim_brk2_cmp2	comp2_out	comp2_out
tim_brk2_cmp3	comp3_out	comp3_out
tim_brk2_cmp4	comp4_out	comp4_out

表 18.6 系统中断互连

tim_sys_brk 输入	TIM1 / TIM8	SYSCFG_CFGR2 寄存器的使能位
tim_sys_brk0	内核锁住	CLL
tim_sys_brk1	可编程电压检测器（PVD）	PVDL
tim_sys_brk2	时钟安全系统（CSS）	无（始终使能）

下表列出连接到 tim_ocref_clr 输入多路复用器的内部源。

表 18.7 连接到 ocref_clr 的输入多路复用器

定时器 OCREF 清除信号	定时器 tim_ocref_clr 信号分配	
	TIM1	TIM8
tim_ocref_clr0	comp1_out	comp1_out
tim_ocref_clr1	comp2_out	comp2_out
tim_ocref_clr2	comp3_out	comp3_out
tim_ocref_clr3	comp4_out	comp4_out

18.3.3 时基单元

可编程高级控制定时器的主要模块是一个 16 位计数器及其相关的自动重载寄存器。计数器可递增计数、递减计数或同时进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包含：

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）
- 重复次数寄存器（TIMx_RCR）

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件（UEV）时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE）。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生 进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位（CEN）置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

注：计数器将在 TIMx_CR1 *寄存器的* CEN *位置* 1 *时刻的一个时钟周期后开始计数。*

预分频器描述

预分频器可对计数器时钟频率进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此可对预分频器进行实时更改。而新的预分频比将在下一更新事件发生时被采用。

下两张图以一些示例说明在预分频比实时变化时计数器的行为：

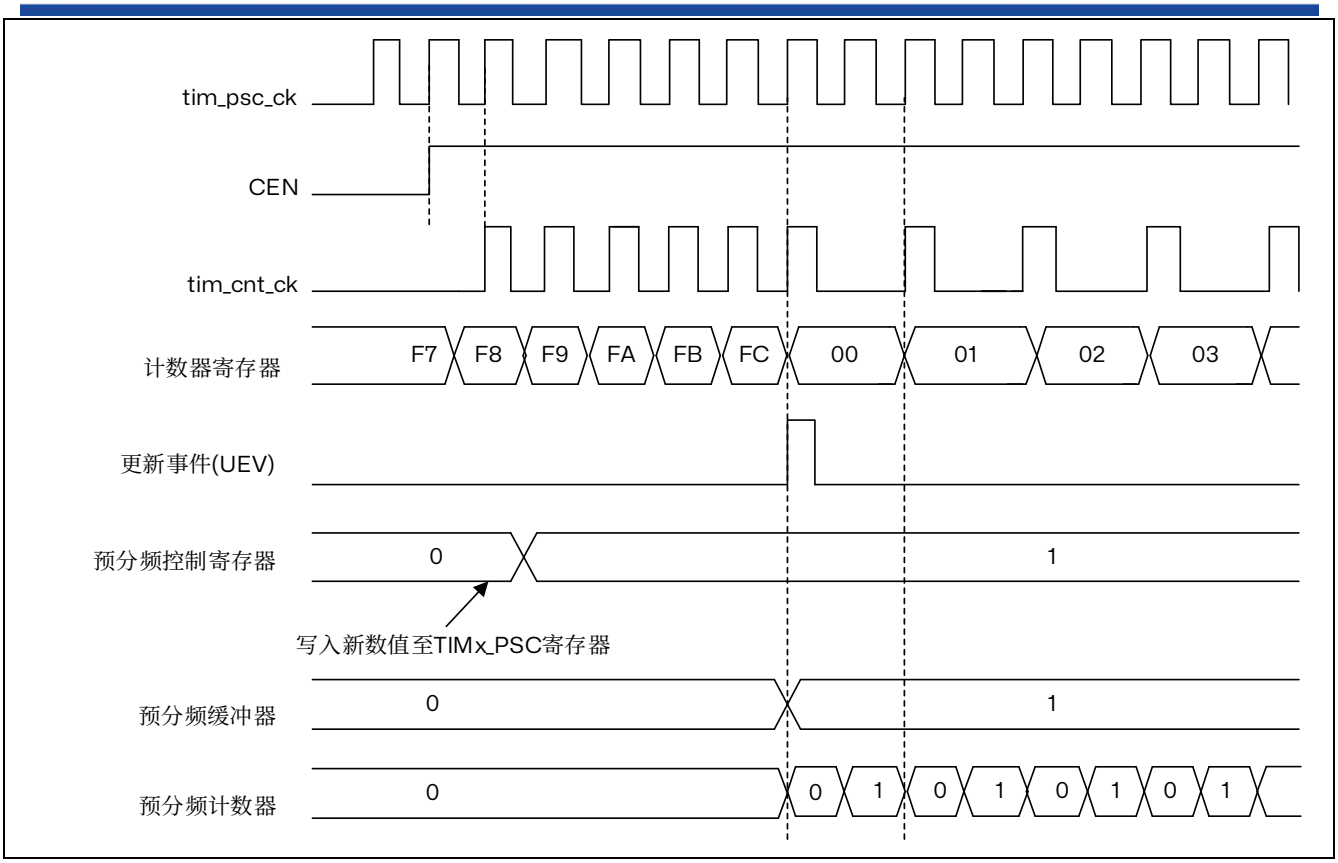


图 18.2 当预分频器的参数从 1 变成 2 时，计数器的时序图

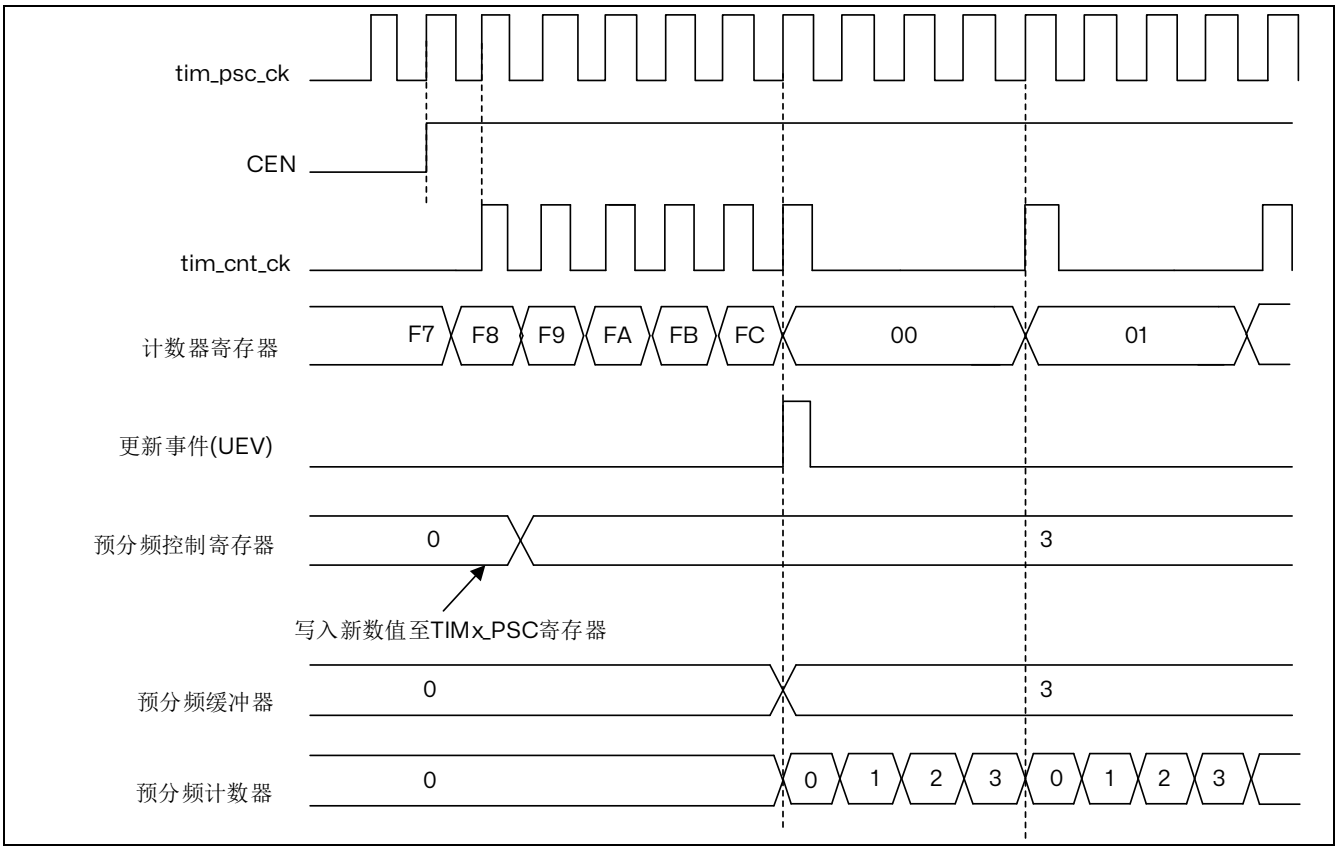


图 18.3 当预分频器的参数从 1 变成 4 时，计数器的时序图

18.3.4 计数模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中编程的次数加一次（TIMx_RCR+1）后，将生成更新事件(UEV)。否则，将在每次计数器上溢时产生更新事件。

将 TIMx_EGR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断）。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容
- 自动重载影子寄存器将以预装载值(TIMx_ARR)进行更新
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

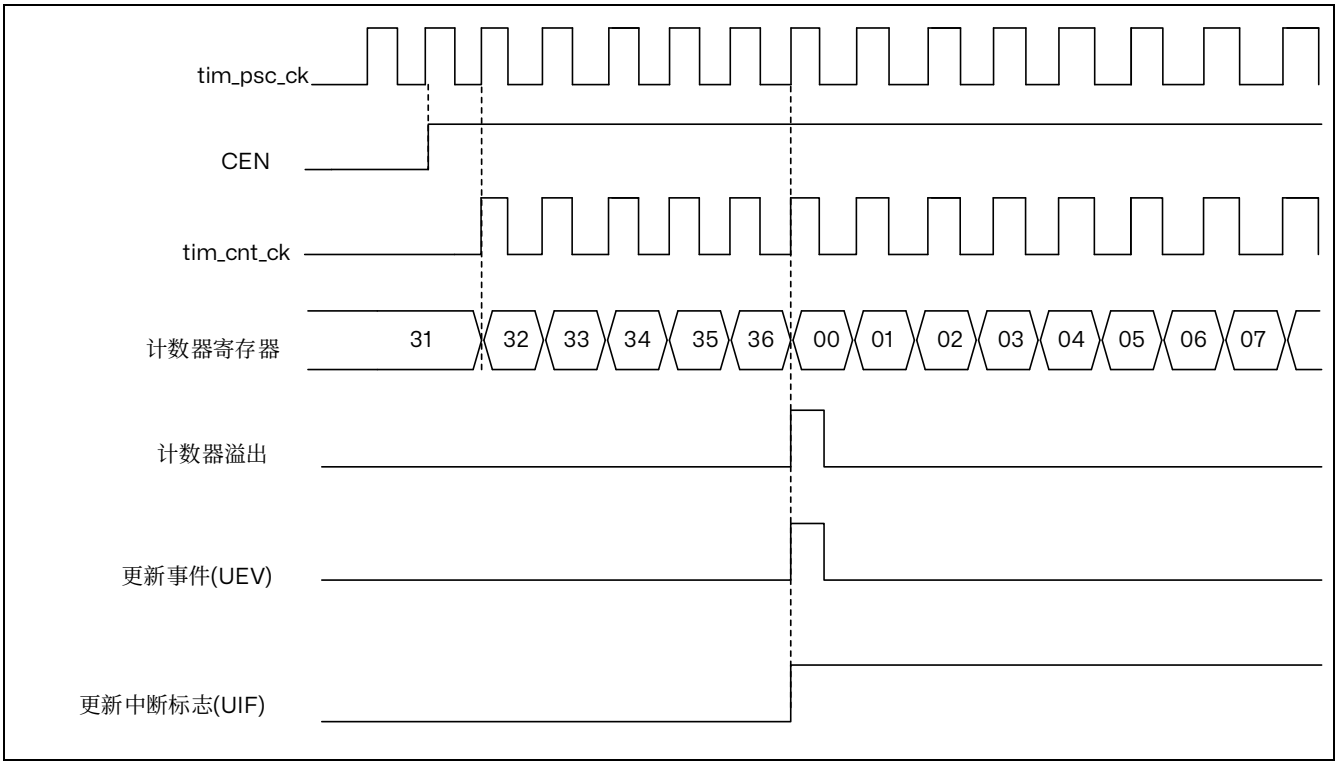


图 18.4 计数器时序图，内部时钟分频系数为 1

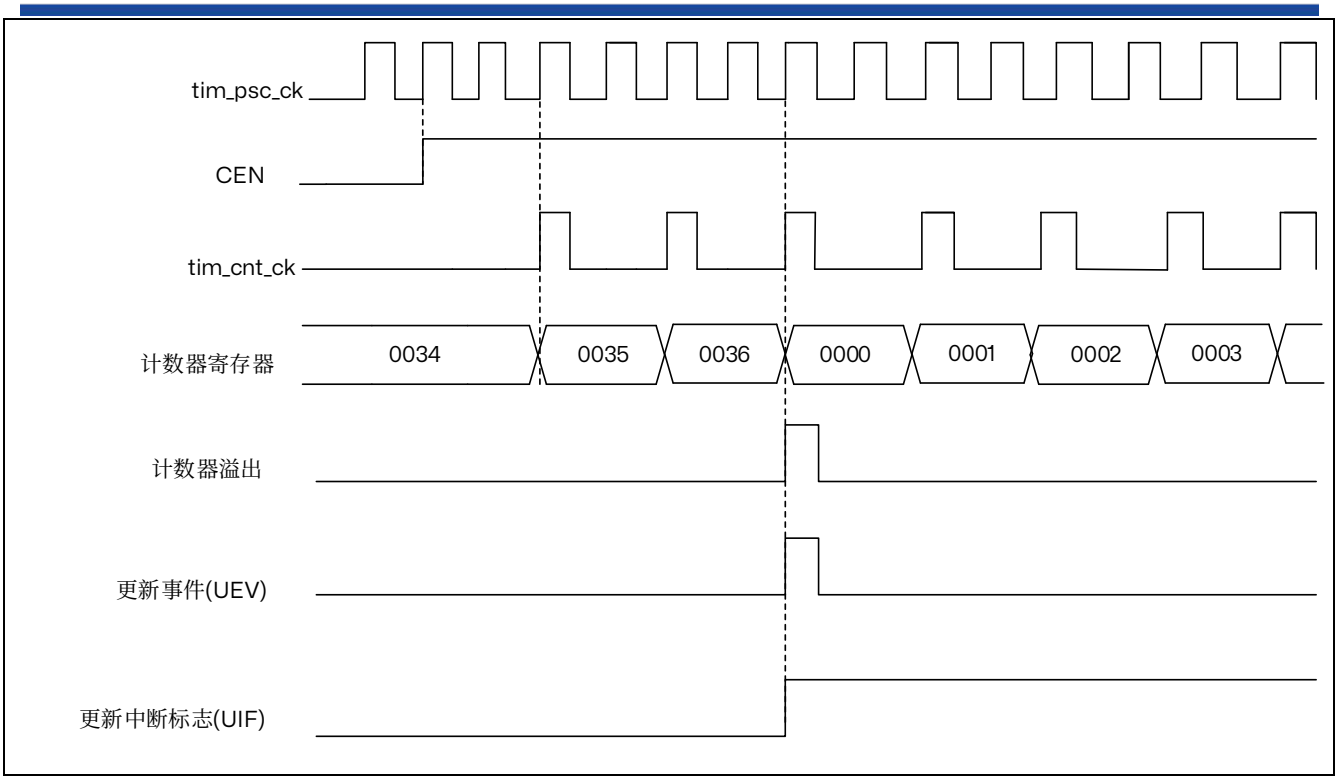


图 18.5 计数器时序图，内部时钟分频系数为 2

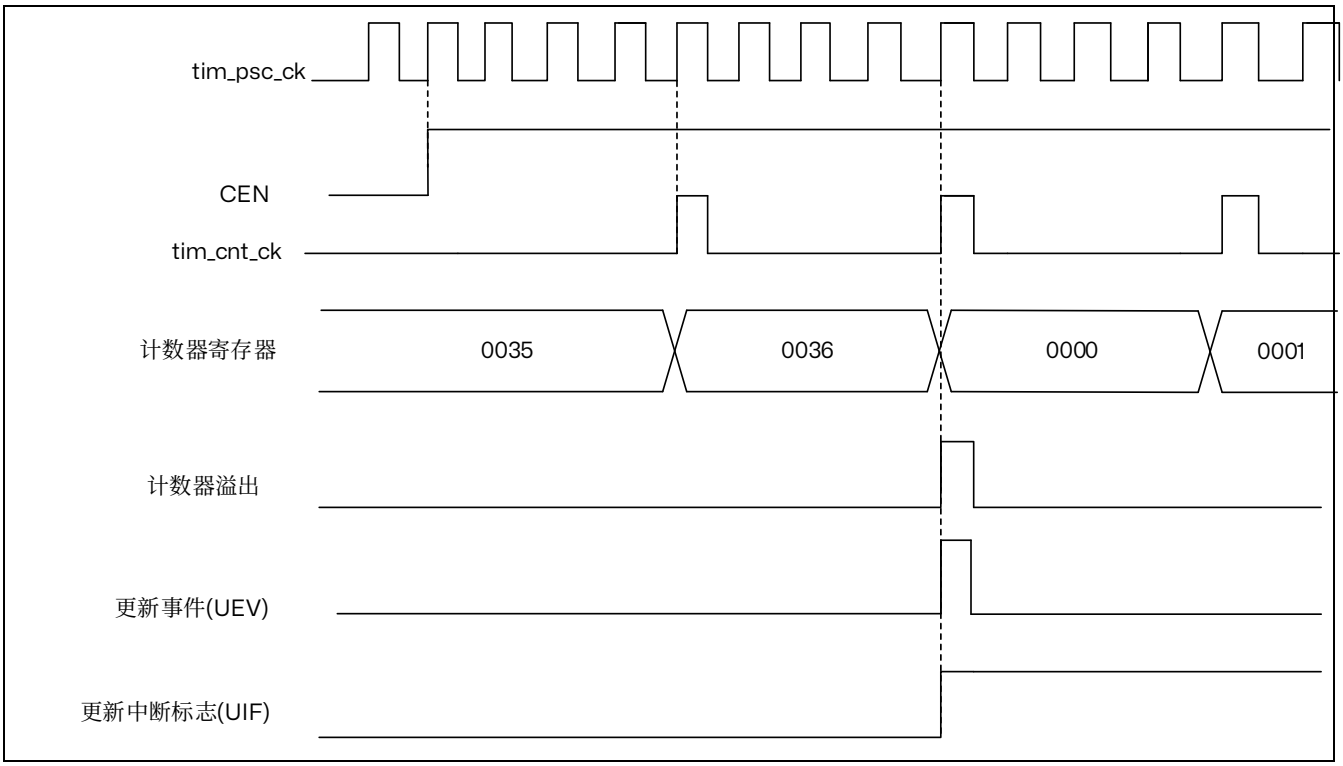


图 18.6 计数器时序图，内部时钟分频系数为 4

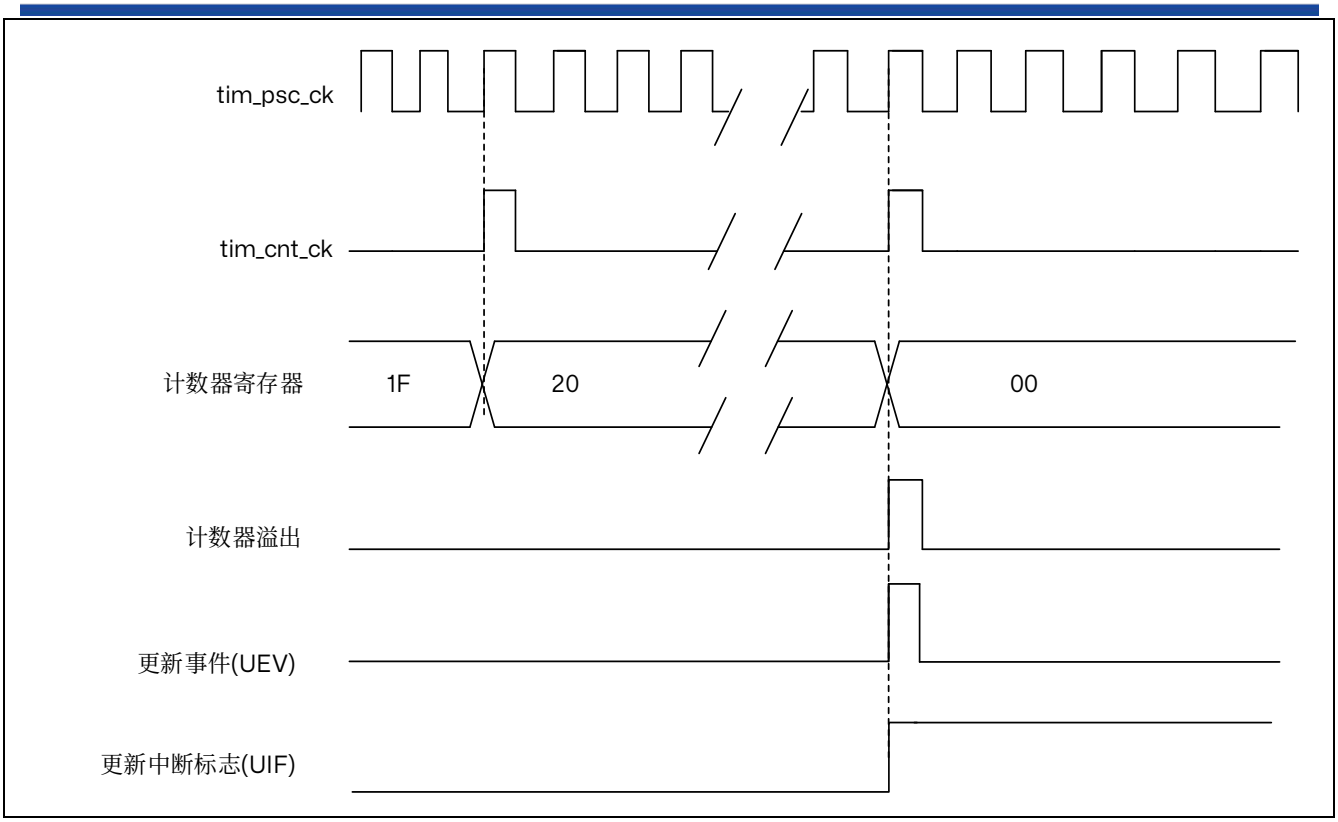


图 18.7 计数器时序图，内部时钟分频系数为 N

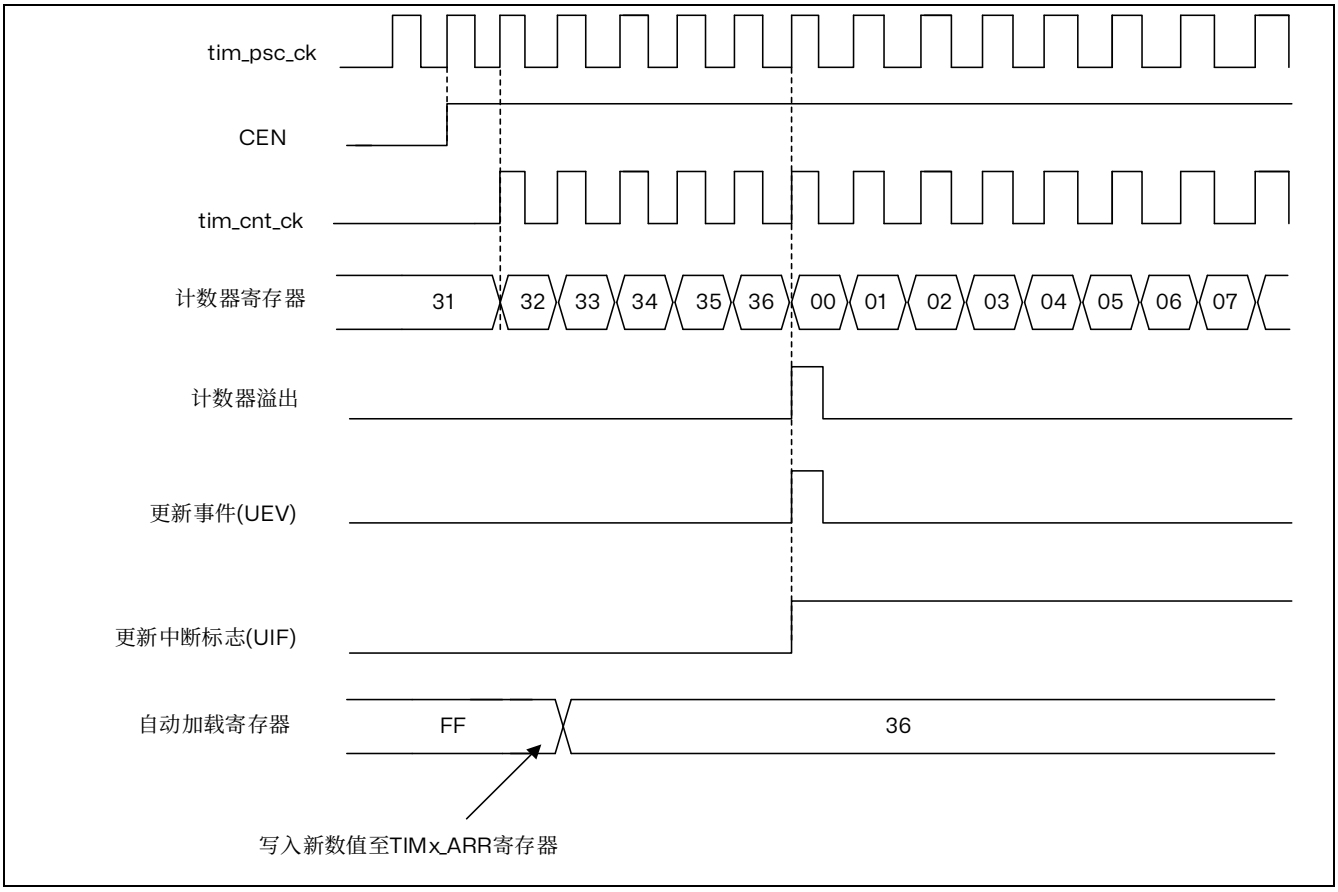


图 18.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

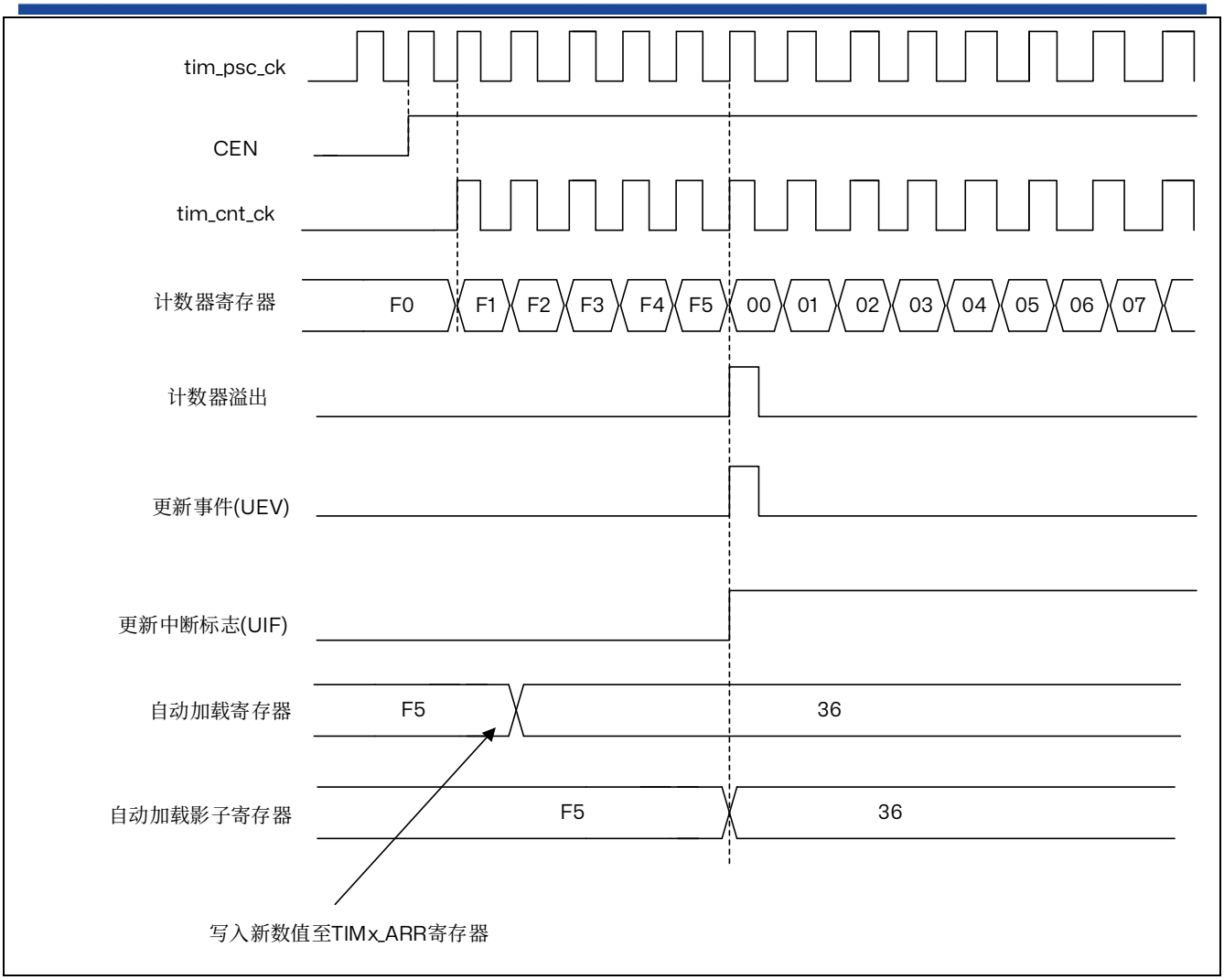


图 18.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

如果使用重复计数器，则当递减计数的重复次数达到重复计数器寄存器中编程的次数加一次（TIMx_RCR+1）后，将生成更新事件（UEV）。否则，将在每次计数器下溢时产生更新事件。

将 TIMx_EGR 寄存器的 UG 位置 1（通过软件或使用从模式控制器）时，也将产生更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器会重新从当前自动重载值开始计数，而预分频器计数器则重新从 0 开始计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，自动重载寄存器会在计数器重载之前得到更新，因此，下一个计数周期就是我们所希望的新的周期长度

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

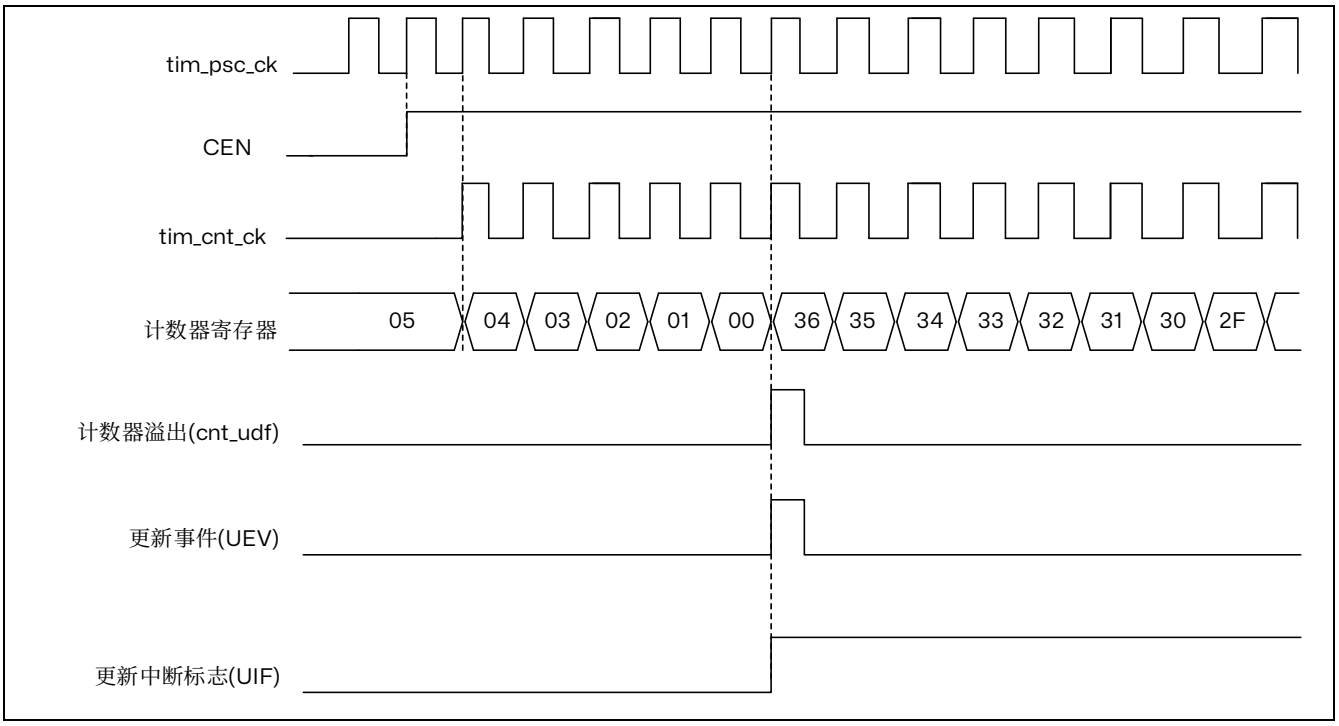


图 18.10 计数器时序图，内部时钟分频系数为 1

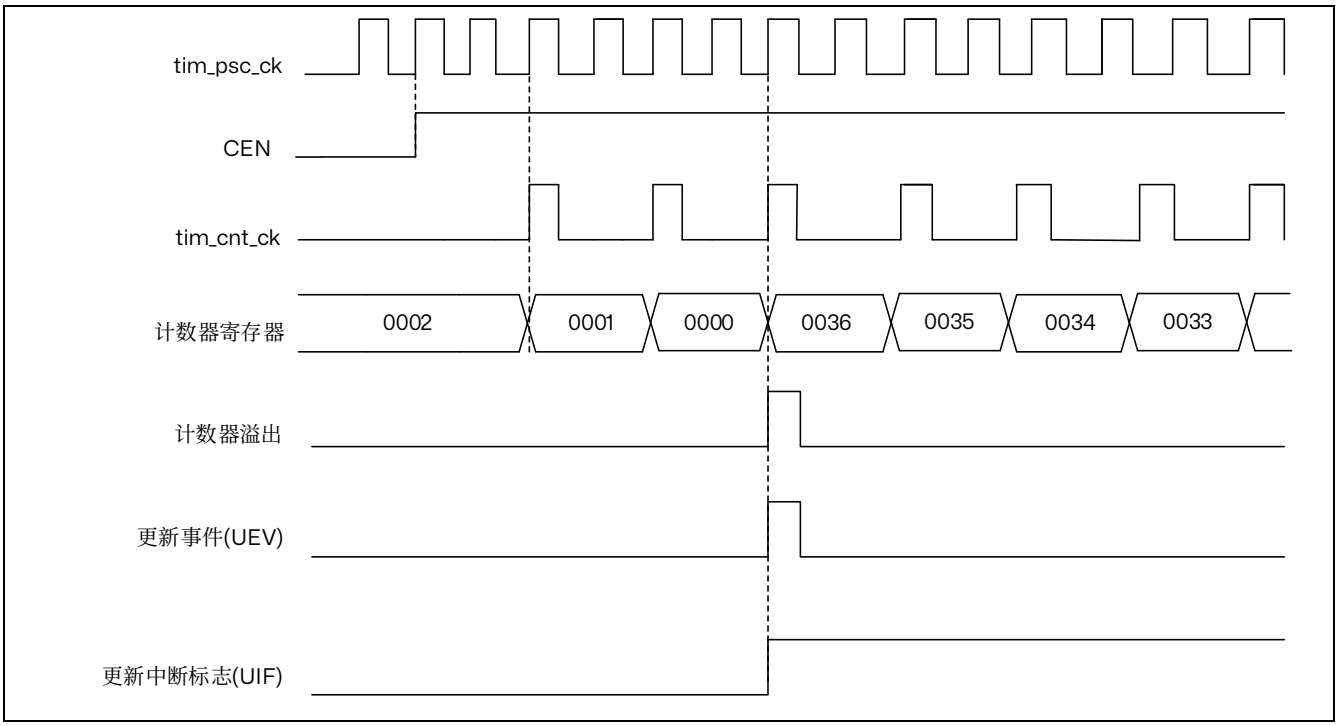


图 18.11 计数器时序图，内部时钟分频系数为 2

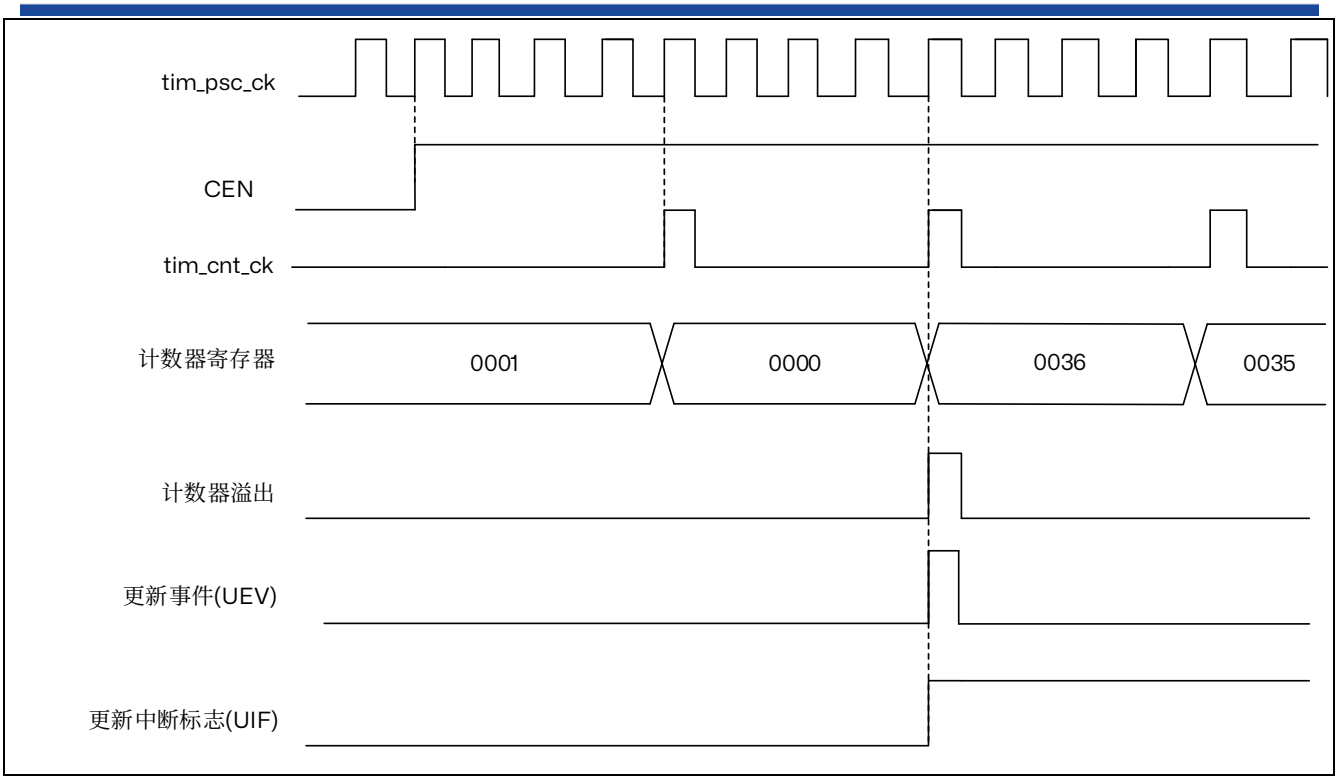


图 18.12 计数器时序图，内部时钟分频系数为 4

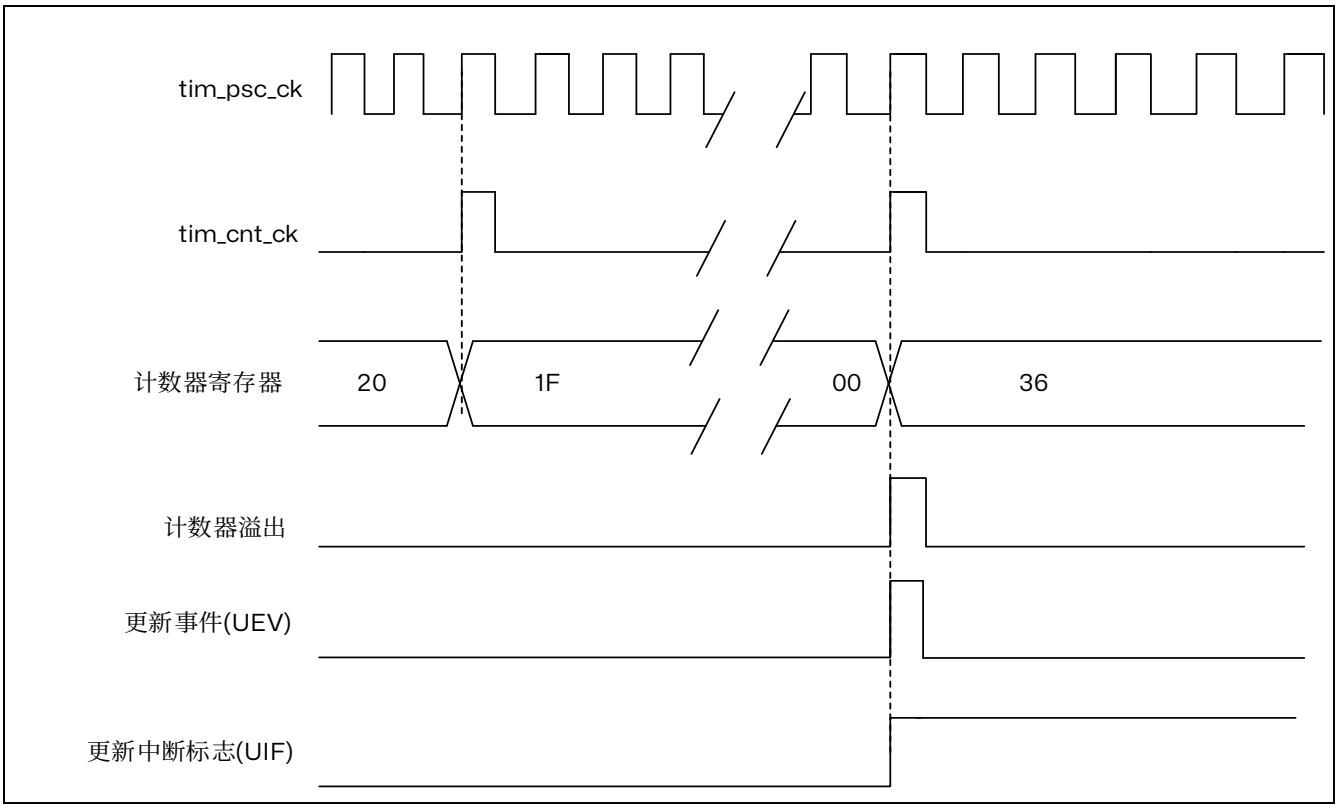


图 18.13 计数器时序图，内部时钟分频系数为 N

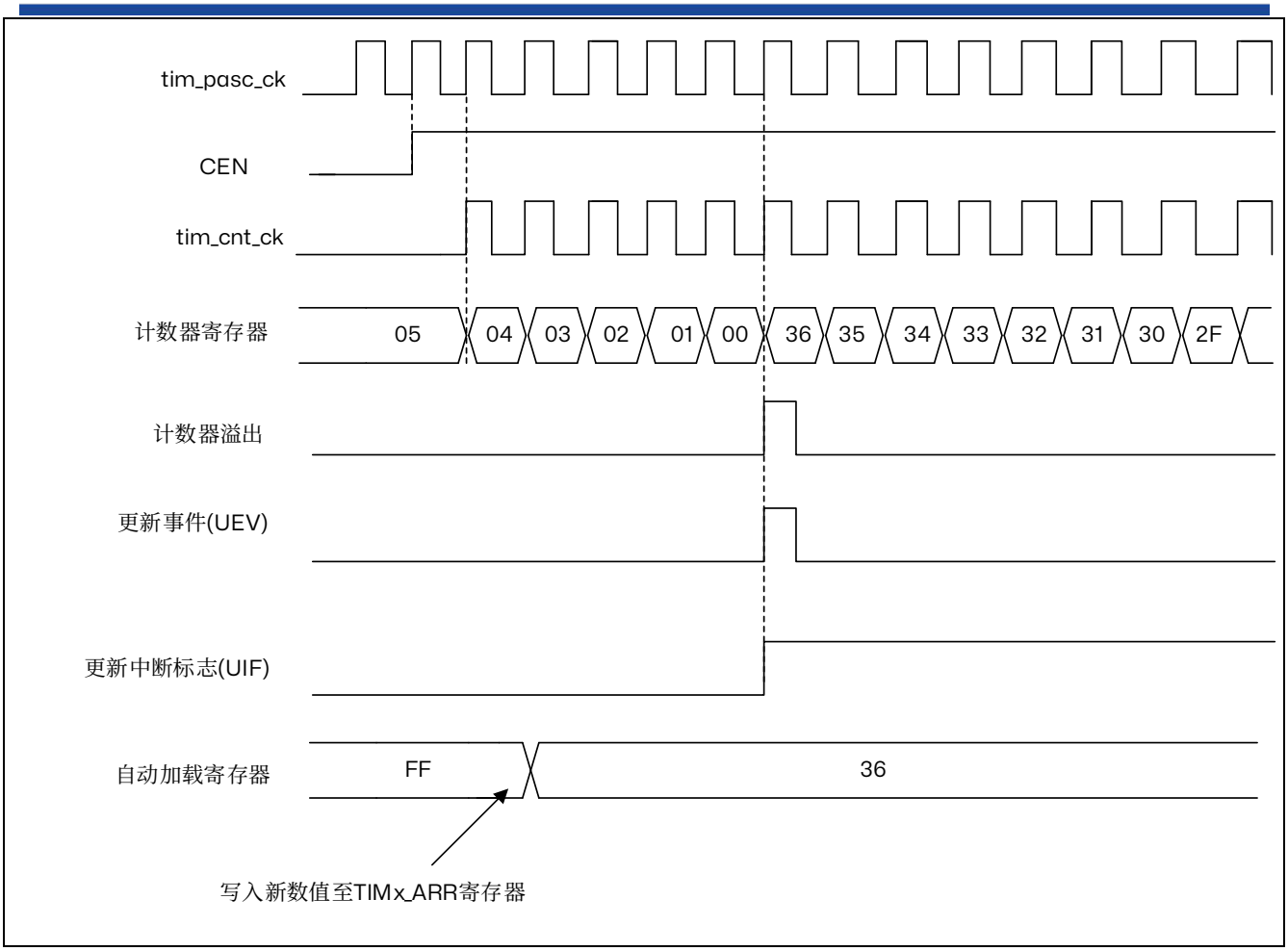


图 18.14 计数器时序图，当没有使用重复计数器时的更新事件

中心对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS = “01”）、计数器递增计数（中心对齐模式 2，CMS = “10”）以及计数器递增/递减计数（中心对齐模式 3，CMS = “11”）。

在此模式下，TIMx_CR1 寄存器的 DIR 方向位不可写入值，而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成 UEV 更新事件，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于

URS 位）：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，如果更新操作是由计数器上溢触发的，则自动重载寄存器在重载计数器之前更新，因此，下一个计数周期就是我们所希望的新的周期长度（计数器被重载新的值）。

以下各图以一些示例说明不同时钟频率下计数器的行为。

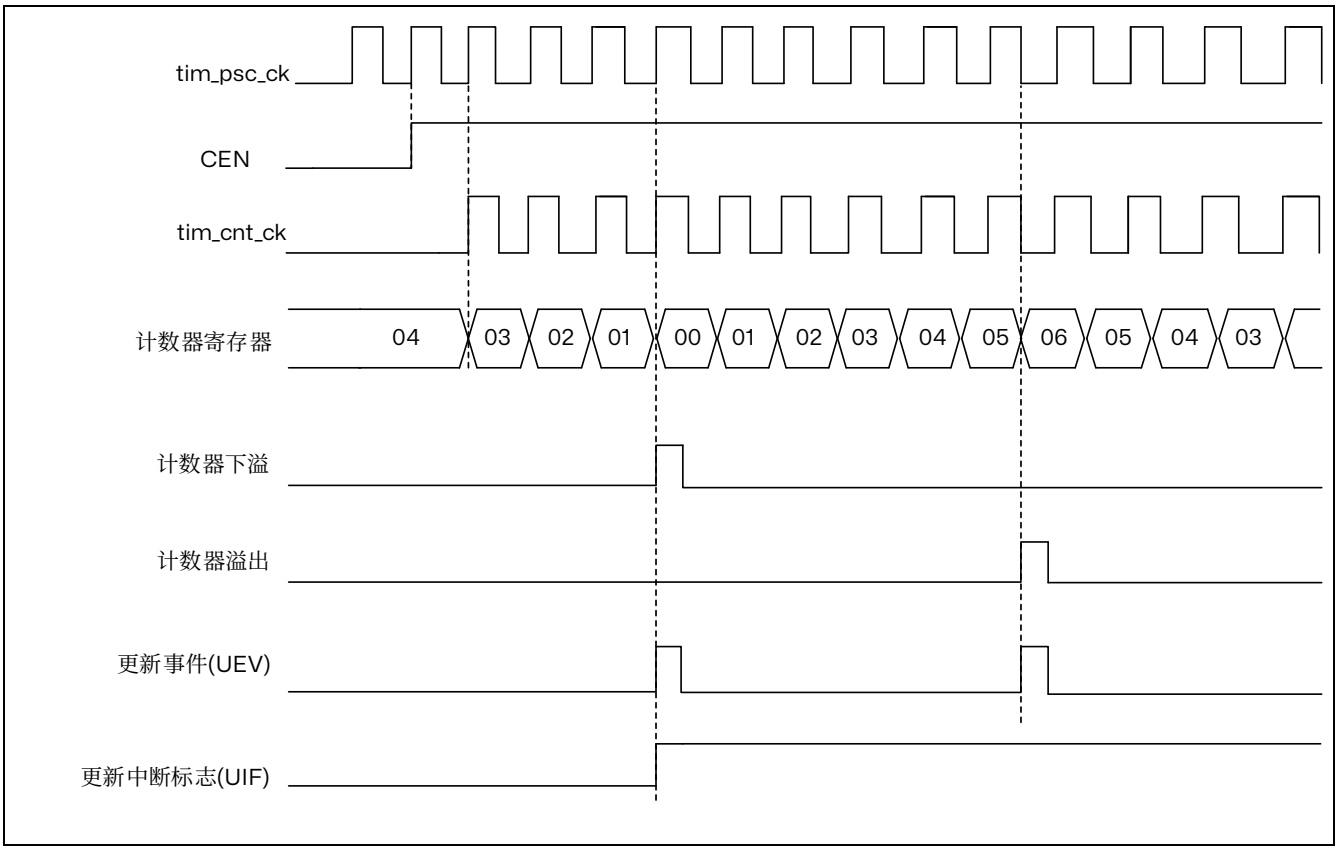


图 18.15 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6

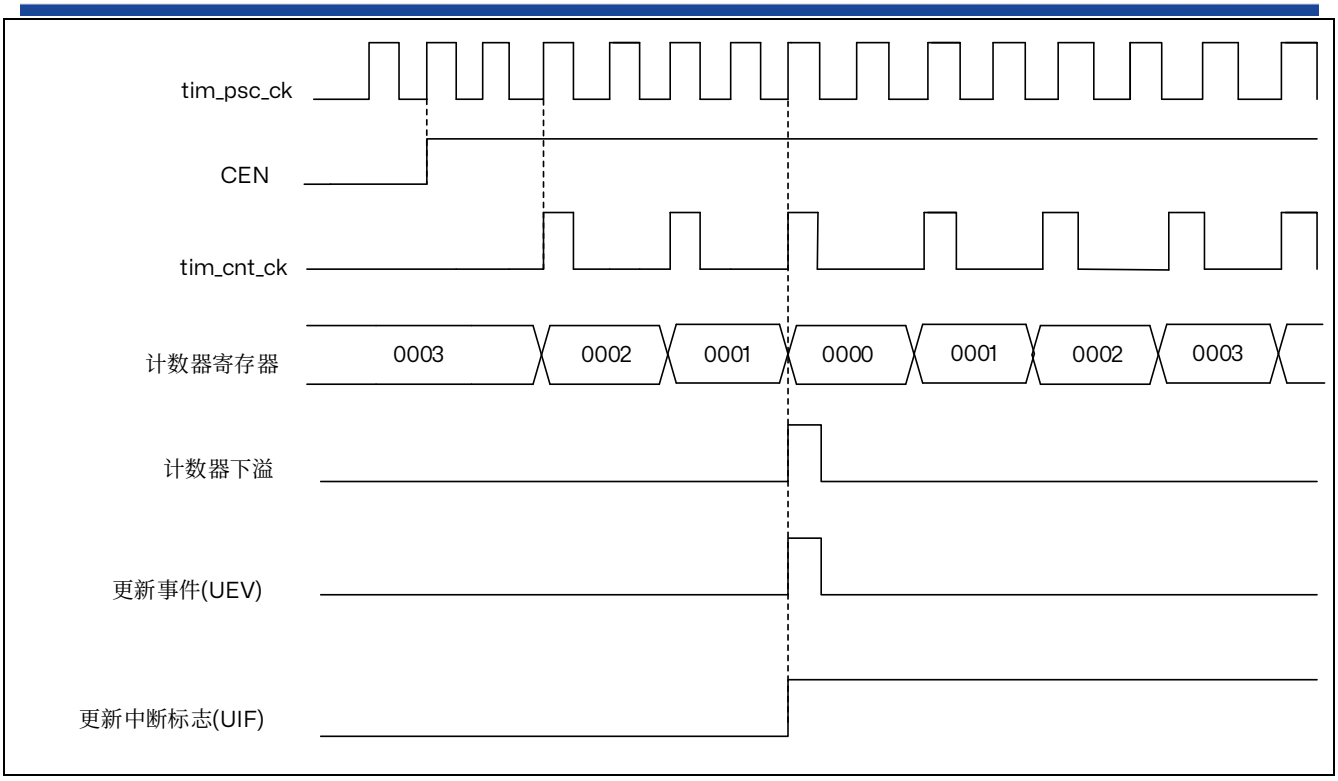


图 18.16 计数器时序图，内部时钟分频系数为 2

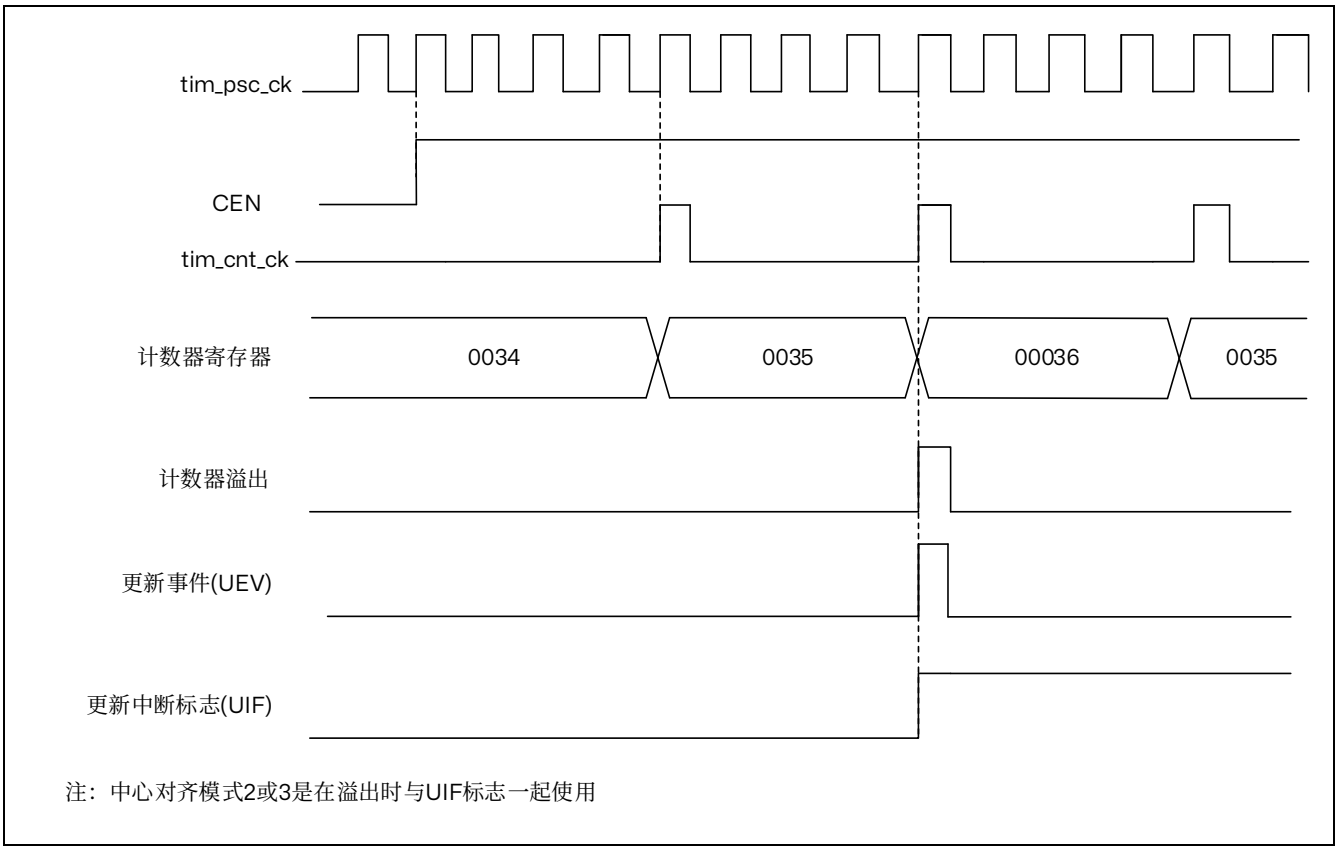


图 18.17 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36

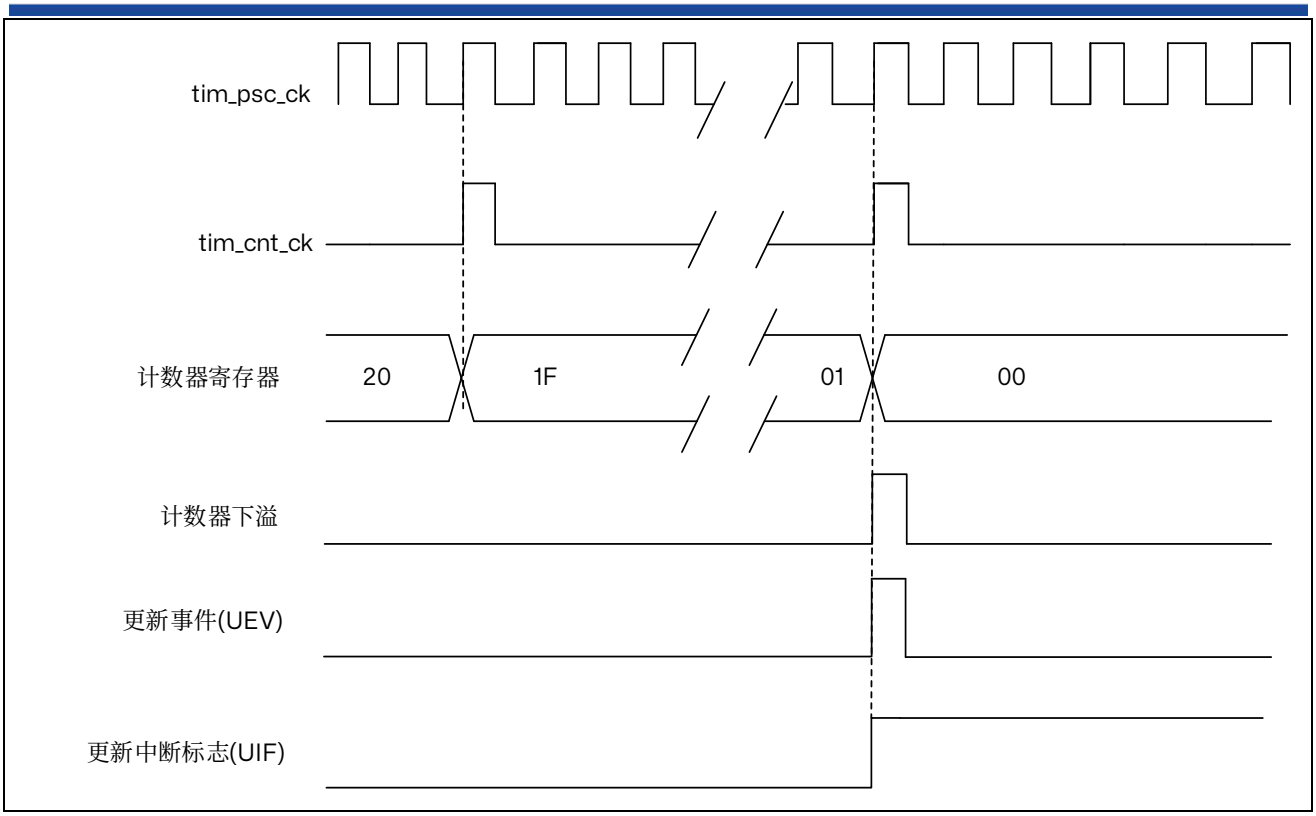


图 18.18 计数器时序图，内部时钟分频系数为 N

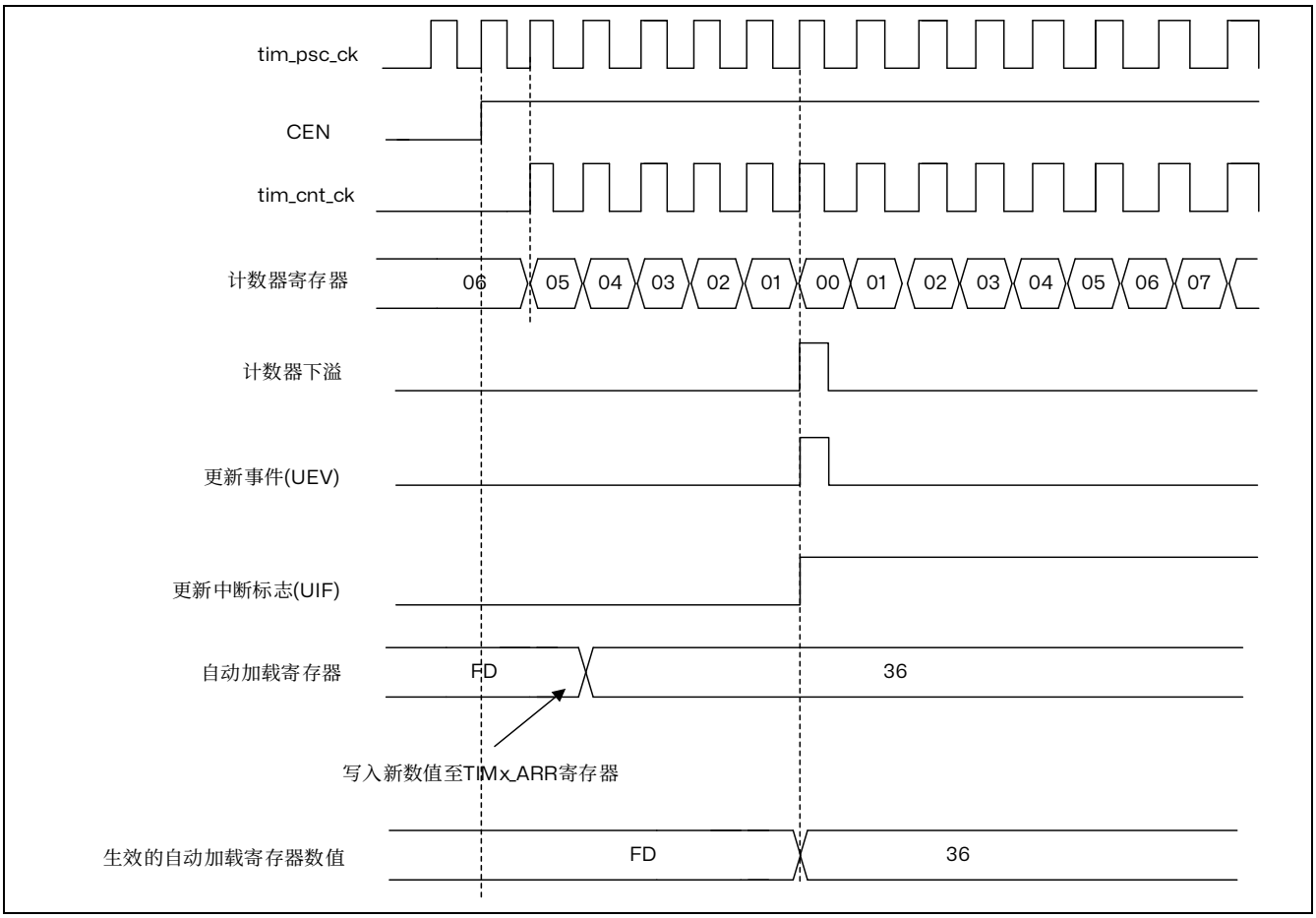


图 18.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

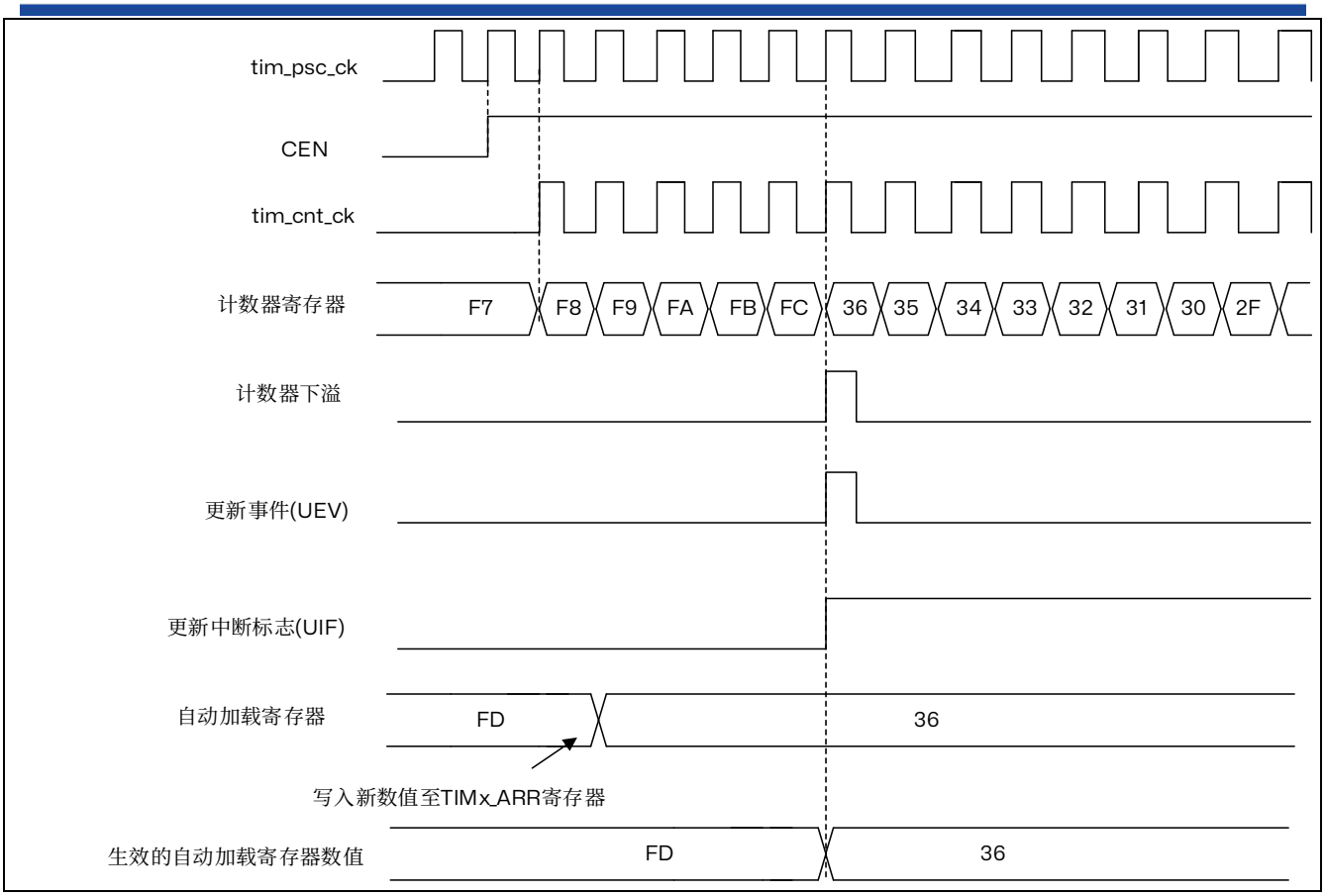


图 18.20 计数器时序图，ARPE=1 时的更新事件（计数器上溢）

18.3.5 重复计数器

时基单元介绍如何因计数器上溢/下溢而生成更新事件（UEV）。实际上，只有当重复计数器达到零时，才会生成更新事件。这在生成 PWM 信号时很有用。

这意味着，每当发生 N+1 个计数器上溢或下溢（其中，N 是 TIMx_RCR 重复计数器寄存器中的值），数据就将从预装载寄存器转移到影子寄存器（TIMx_ARR 自动重载寄存器、TIMx_PSC 预分频器寄存器以及比较模式下的 TIMx_CCRx 捕获/比较寄存器）。

重复计数器在下列情况下递减：

- 递增计数模式下的每个计数器上溢。
- 递减计数模式下的每个计数器下溢。
- 中心对齐模式下每个计数器上溢和计数器下溢。

  尽管这限制了最大重复次数为 32768 个 PWM 周期，但它使得每个 PWM 周期可以更新两次占空比。在中心对齐模式下，每个 PWM 周期仅刷新一次比较寄存器时，由于模式的对称性，最大分辨率为 $2xT_{ck}$。

重复计数器是自动重载类型；其重复率为 TIMx_RCR 寄存器所定义的值。当更新事件由软件（通过将 TIMx_EGR 寄存器的 UG 位置 1）或硬件（通过从模式控制器）生成时，无论重复计数器的值为多少，更新事件都将立即发生，并且在重复计数器中重新装载 TIMx_RCR 寄存器的内容。

在中心对齐模式下，如果 RCR 值为奇数，更新事件将在上溢或下溢时发生，这取决于何时写入 RCR 寄存器以及何时启动计数器。如果在启动计数器前写入 RCR，则 UEV 在上溢时发生。如果在启动计数器后写入 RCR，则 UEV 在下溢时发生。

例如，如果 RCR = 3，UEV 将在每个周期的第四个上溢或下溢事件时生成（取决于何时写入 RCR）。

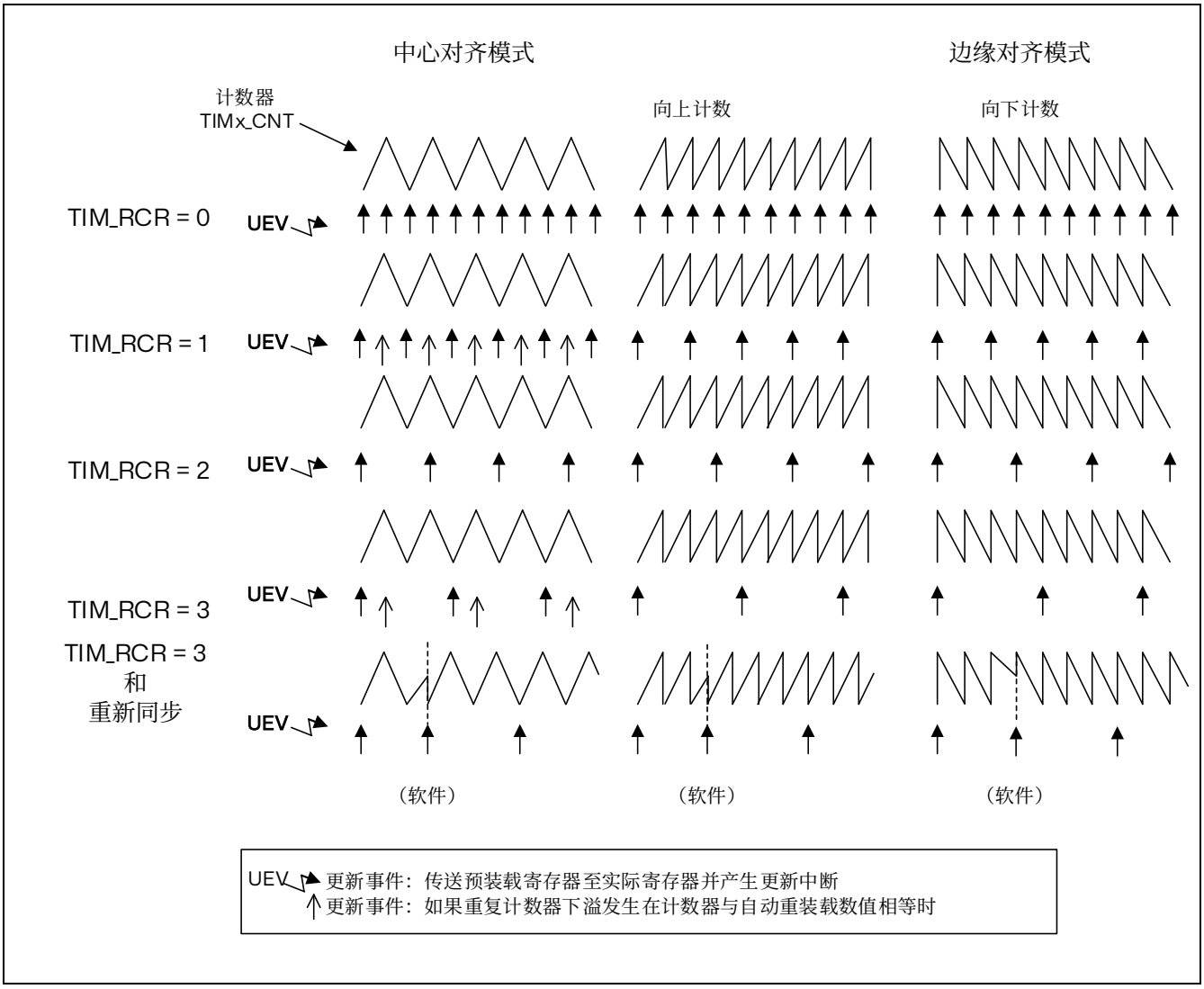


图 18.21 不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置

18.3.6 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（tim_ker_ck）

内部时钟源（tim_ker_ck）

如果禁止从模式控制器（SMS=000），则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍保持自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

下图显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

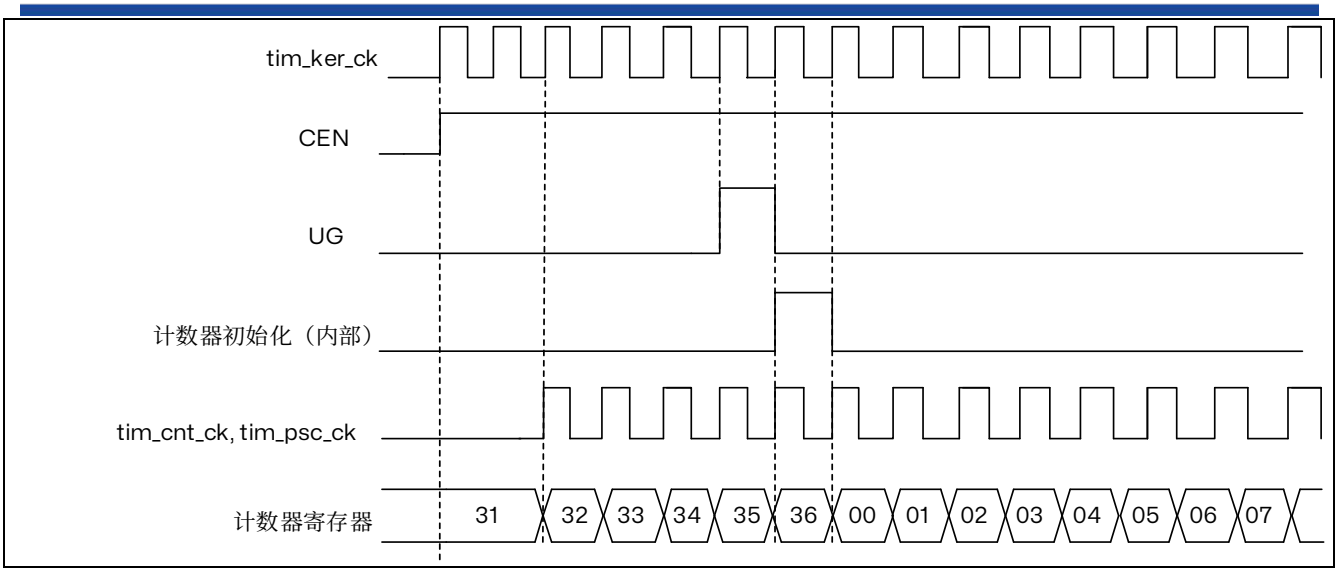


图 18.22 一般模式下的控制电路，内部时钟分频系数为 1

18.3.7 捕获比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）和一个输出阶段（比较器和输出控制）构建而成。

下列图提供了一个捕获/比较通道的概述。

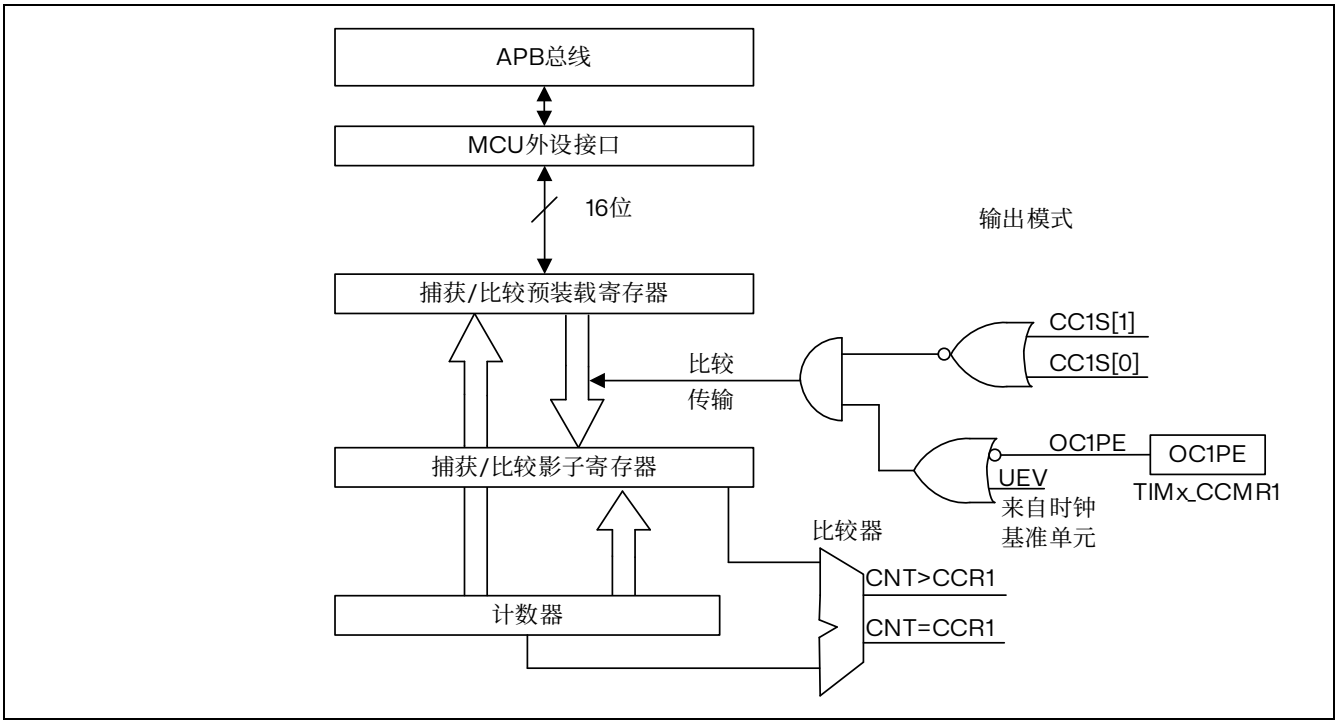


图 18.23 捕获/比较通道 1 的主电路

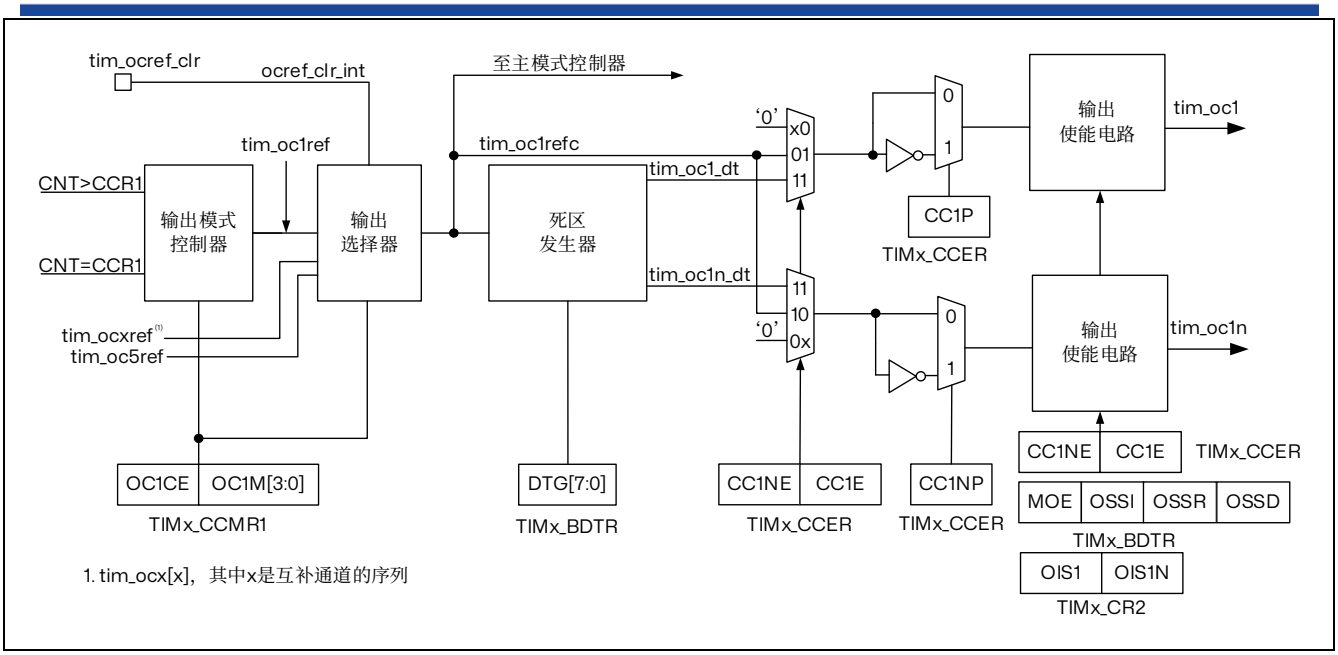


图 18.24 捕获/比较通道的输出阶段（通道 1，通道 2、3 和 4 同上）

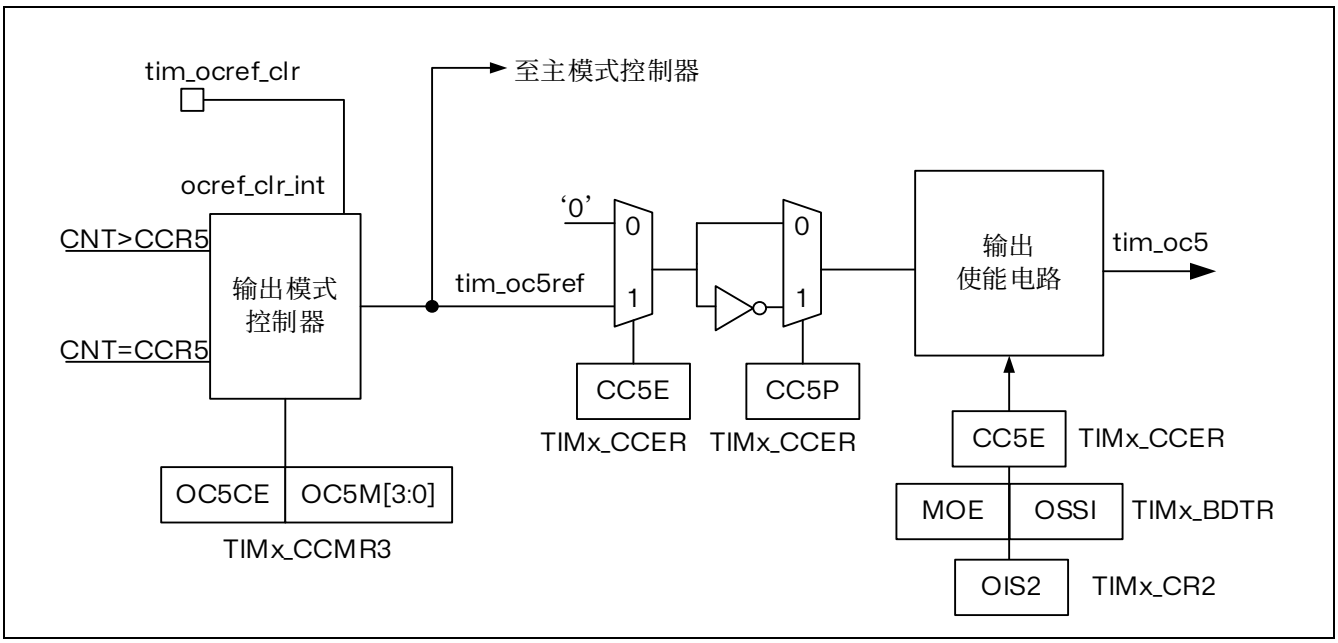


图 18.25 捕获/比较通道的输出阶段（通道 5，通道 6 同上）

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。始终可通过读写操作访问预装载寄存器。

在比较模式下，预装载寄存器的内容将被复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

18.3.8 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS 位= 00）下，可直接由软件将每个输出比较信号（tim_ocxref 和 tim_ocx/tim_ocxn）强制设置为有效电平或无效电平，而无需考虑输出比较寄存器和计数器之间的任何比较结果。

要将输出比较信号（tim_ocxref/tim_ocx）强制设置为有效电平，只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 0101。tim_ocxref 进而强制设置为高电平（tim_ocxref 始终为高电平有效），同时 tim_ocx 获取 CCxP 极性位的相反值。

例如：CCxP=0（tim_ocx 高电平有效）=> tim_ocx 强制设置为高电平。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 100，可将 tim_ocxref 信号强制设置为低电平。无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可发送相应的中断请求。下面的输出比较模式一节对此进行了介绍。

18.3.9 输出比较模式

此功能用于控制输出波形，或指示已经过某一时间段。通道 1 至 4 可以输出，而通道 5 和 6 仅可在微控制器内部使用（例如，用于复合波形生成或 ADC 触发）。

当捕获/比较寄存器与计数器之间相匹配时，输出比较功能：

- 将为相应的输出引脚分配一个可编程值，该值由输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）定义。匹配时，输出引脚既可保持其电平（OCXM=0000），也可设置为有效电平（OCXM=0001）、无效电平（OCXM=0010）或进行翻转（OCxM=0011）。
- 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
- 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装载寄存器。

在输出比较模式下，更新事件 UEV 对 tim_ocxref 和 tim_ocx 出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，则需将 CCxIE 位置 1。
4. 选择输出模式。例如：
   — 当 CNT 与 CCRx 匹配时，写入 OCxM =0011 以翻转 tim_ocx 输出引脚
   — 写入 OCxPE = 0 以禁止预装载寄存器
   — 写入 CCxP = 0 以选择高电平有效极性
   — 写入 CCxE = 1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可通过软件随时更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预加载寄存器（OCxPE=“0”，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。下图给出了一个示例。

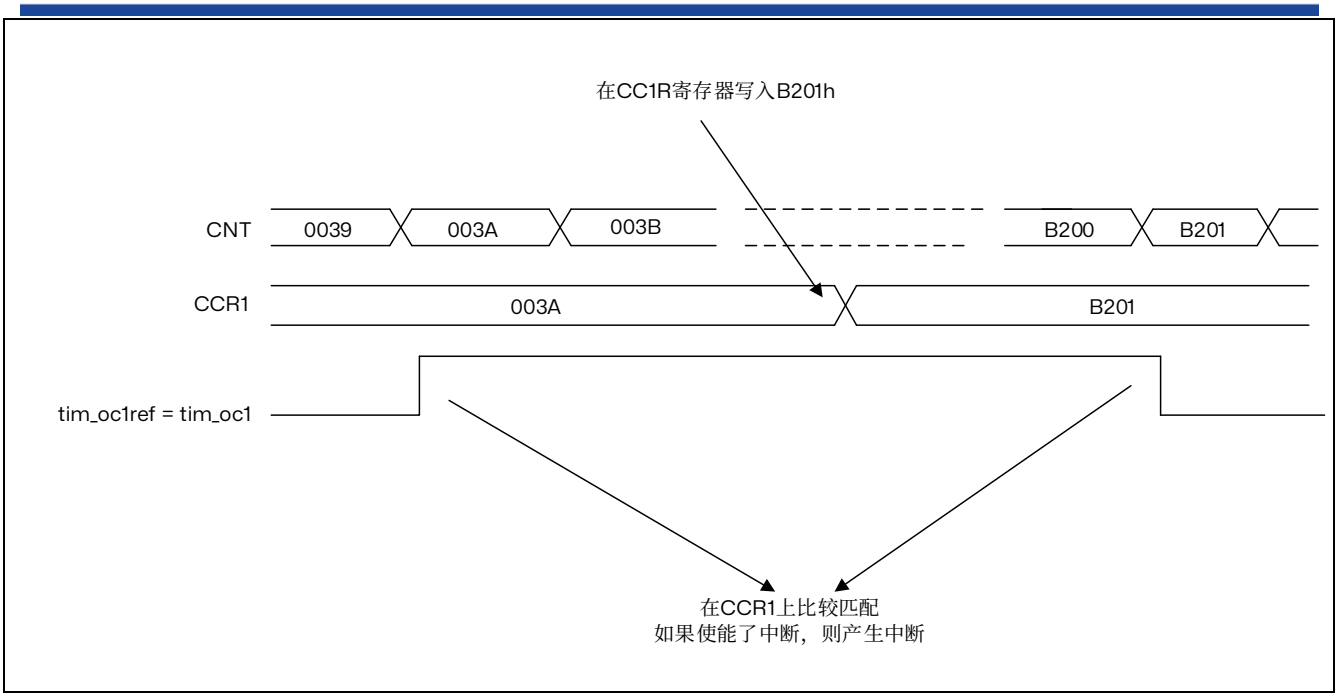


图 18.26 输出比较模式，翻转 OC1

18.3.10 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入“0110”（PWM 模式 1）或“0111”（PWM 模式 2），可以独立选择各通道（每个 OCx 输出对应一个 PWM）的 PWM 模式。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器（在递增计数或中心对齐模式下）。

由于只有在发生更新事件时预装载寄存器才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

tim_ocx 极性可使用 TIMx_CCER 寄存器的 CCxP 位来编程。既可以设为高电平有效，也可以设为低电平有效。通过 CCxE、CCxNE、MOE、OSSI、OSSR、OSSD 位（TIMx_CCER 和 TIMx_BDTR 寄存器）的组合使能 OCx 输出。有关详细信息，请参见 TIMx_CCER 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 总是与 TIMx_CCRx 进行比较，以确定是 TIMx_CCRx ≤ TIMx_CNT 还是 TIMx_CNT ≤ TIMx_CCRx（取决于计数器计数方向）。

根据 TIMx_CR1 寄存器中的 CMS 位状态，定时器能够产生边沿对齐模式或中心对齐模式的 PWM 信号。

PWM 边沿对齐模式

- 递增计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低时执行递增计数。请参见递增计数模式一节。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 tim_ocxref 便为高电平，否则为低电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。如果比较值为 0，则 tim_ocxref 保持为“0”。下图举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

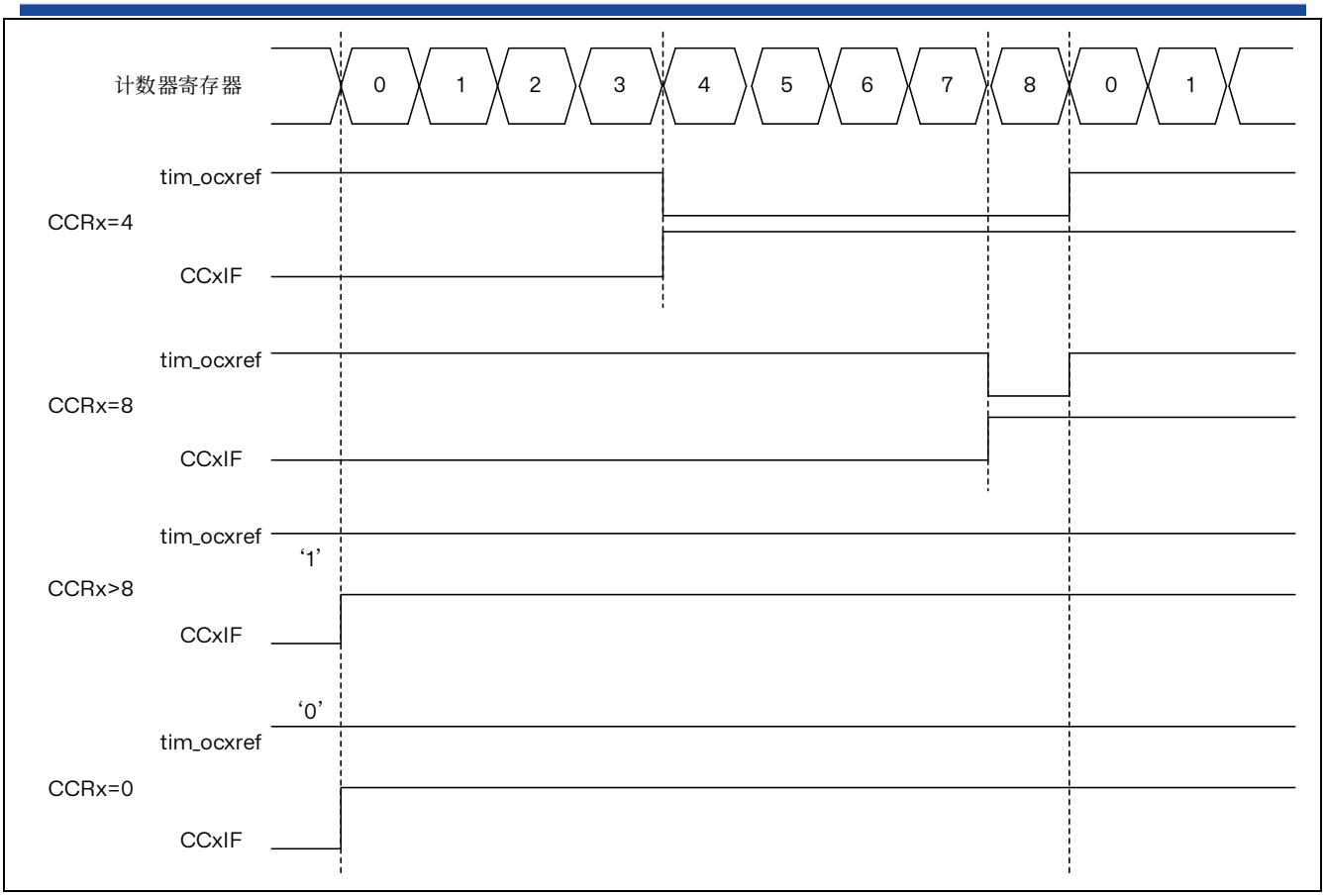


图 18.27 边沿对齐的 PWM 波形（ARR=8）

- 递减计数配置

当 TIMx_CR1 寄存器中的 DIR 位为高时执行递减计数。请参见递减计数模式一节。

在 PWM 模式 1 下，只要 TIMx_CNT > TIMx_CCRx，参考信号 tim_ocxref 即为低电平，否则其为高电平。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重载值，tim_ocxref 保持为“1”。此模式下不可能产生 0% 的 PWM 波形。

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”（其余所有配置对 tim_ocxref/tim_ocx 信号具有相同的作用），中心对齐模式生效。根据 CMS 位的配置，可以在计数器递增计数、递减计数或同时递增和递减计数时将比较标志置 1。TIMx_CR1 寄存器中的方向位（DIR）由硬件更新，不得通过软件更改。请参见中心对齐模式（递增/递减计数）。

下图显示了中心对齐模式的 PWM 波形，在此例中：

- TIMx_ARR=8
- PWM 模式为 PWM 模式 1,
- 在根据 TIMx_CR1 寄存器中 CMS=01 而选择的中心对齐模式 1 下，当计数器递减计数时，比较标志置 1。

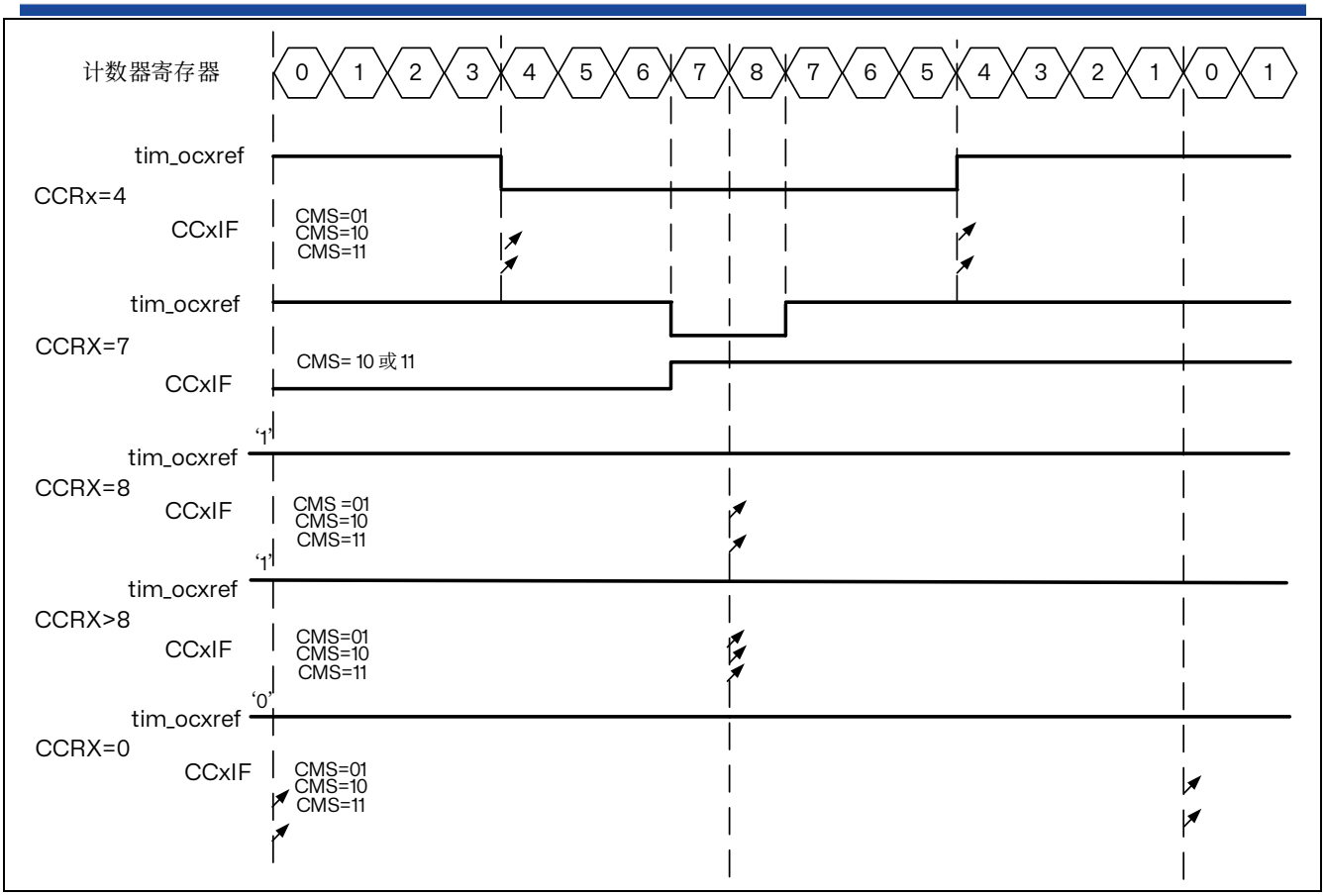


图 18.28 中央对齐的 PWM 波形（ARR=8）

关于使用中央对齐模式的提示：

- 启动中心对齐模式时将使用当前的递增/递减计数配置。这意味着计数器将根据写入 TIMx_CR1 寄存器中 DIR 位的值进行递增或递减计数。此外，不得同时通过软件修改 DIR 和 CMS 位。
- 不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：
  — 如果写入计数器中的值大于自动重载值（TIMx_CNT>TIMx_ARR），计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  — 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

抖动模式

通过使能抖动模式，使用 TIMx_CR1 寄存器中的 DITHEN 位，可以提高 PWM 模式的有效分辨率。这既适用于 CCR（用于增加占空比分辨率），也适用于 ARR（用于增加 PWM 频率分辨率）。

工作原理是在 16 个连续的 PWM 周期中，以预定义的模式使实际的 CCR（或 ARR）值略有变化（增加或减少一个定时器时钟周期）。这使得占空比或 PWM 期间的平均值提高了 16 倍分辨率。下图展示了应用于 4 个连续 PWM 周期的抖动原理。

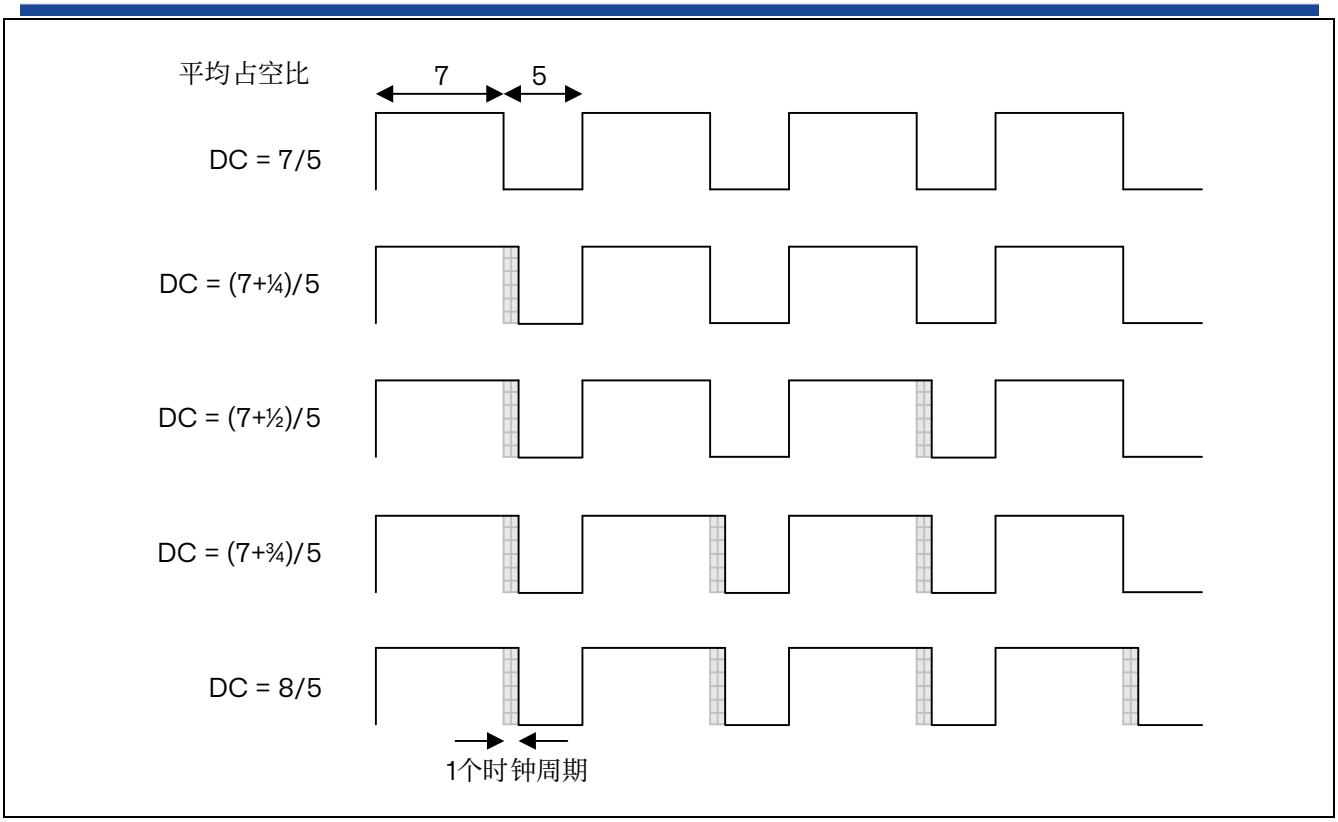


图 18.29 抖动原理

当抖动模式被使能时，寄存器编码将如下所示进行变化（参见下图示例）

- 4 个最低有效位（LSB）用于编码增强的分辨率部分（小数部分）
- 最高有效位（MSB）左移至 19:4 位，用于编码基本值

*注：如果设置清除*DITHEN *位，*ARR *和*CCR *值将自动更新。（例如，如果*ARR=0x05 *且*DITHEN=0，*它将更新为*ARR=0x50 *且*DITHEN=1*）。*

1. *必须重置*CEN *和*ARPE *位*
2. ARR[3:0]*位必须重置*
3. DITHEN *位必须重置*
4. CCIF *标志必须清除*
5. *可以设置*CEN *位（最终设置*ARPE=1*）。*

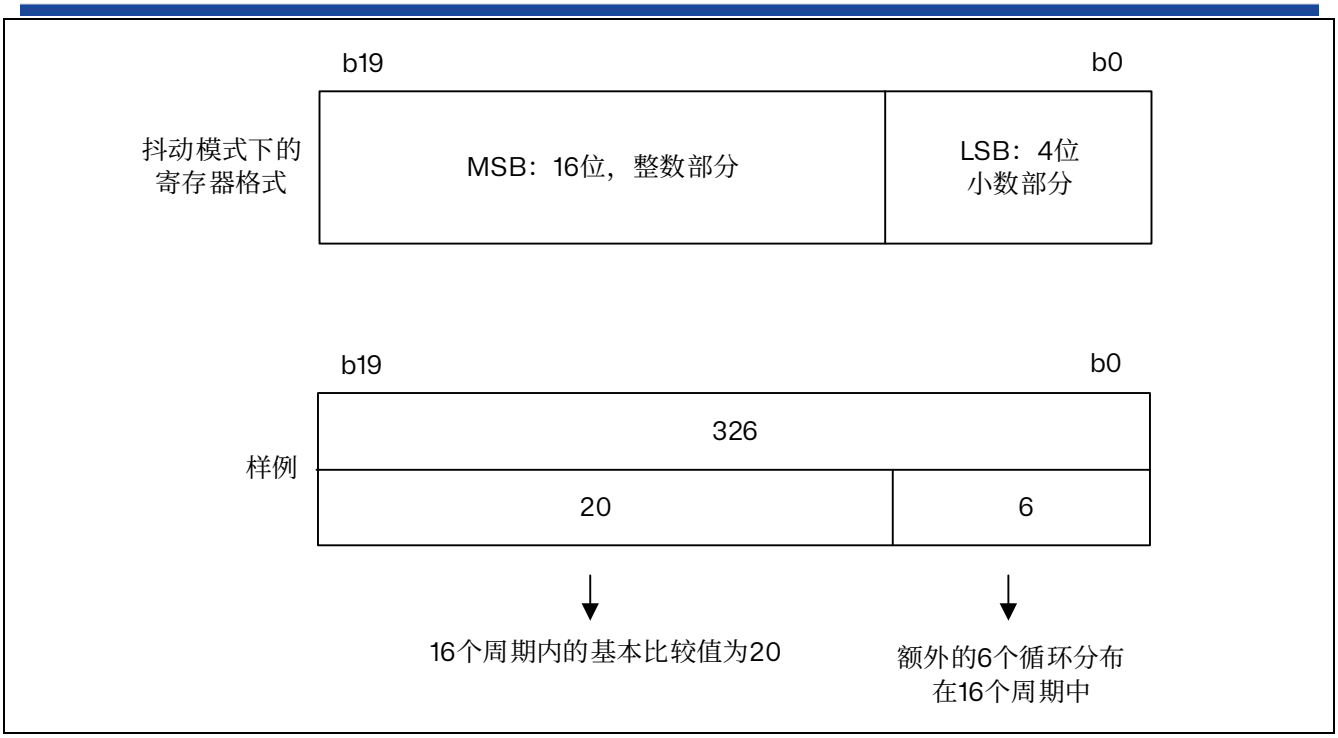


图 18.30 抖动模式下的数据格式和寄存器编码

最小频率由以下公式给出：

$$\text{分辨率} = \frac{\text{FTim}}{\text{Fpwm}} \Rightarrow \text{FpwmMin} = \frac{\text{FTim}}{\text{最大分辨率}}$$

$$\text{禁用抖动模式：FpwmMin} = \frac{\text{FTim}}{65536}$$

$$\text{使能抖动模式：FpwmMin} = \frac{\text{FTim}}{65535 + \frac{15}{16}}$$

*注意：在抖动模式下，*TIMx_ARR *和*TIMx_CCRy *的最大值限制为*0xFFFEF *（对应于整数部分的*65534 *和抖动部分的*15*）。*

如下图所示，无论 PWM 频率如何，抖动模式都可以增加 PWM 分辨率。

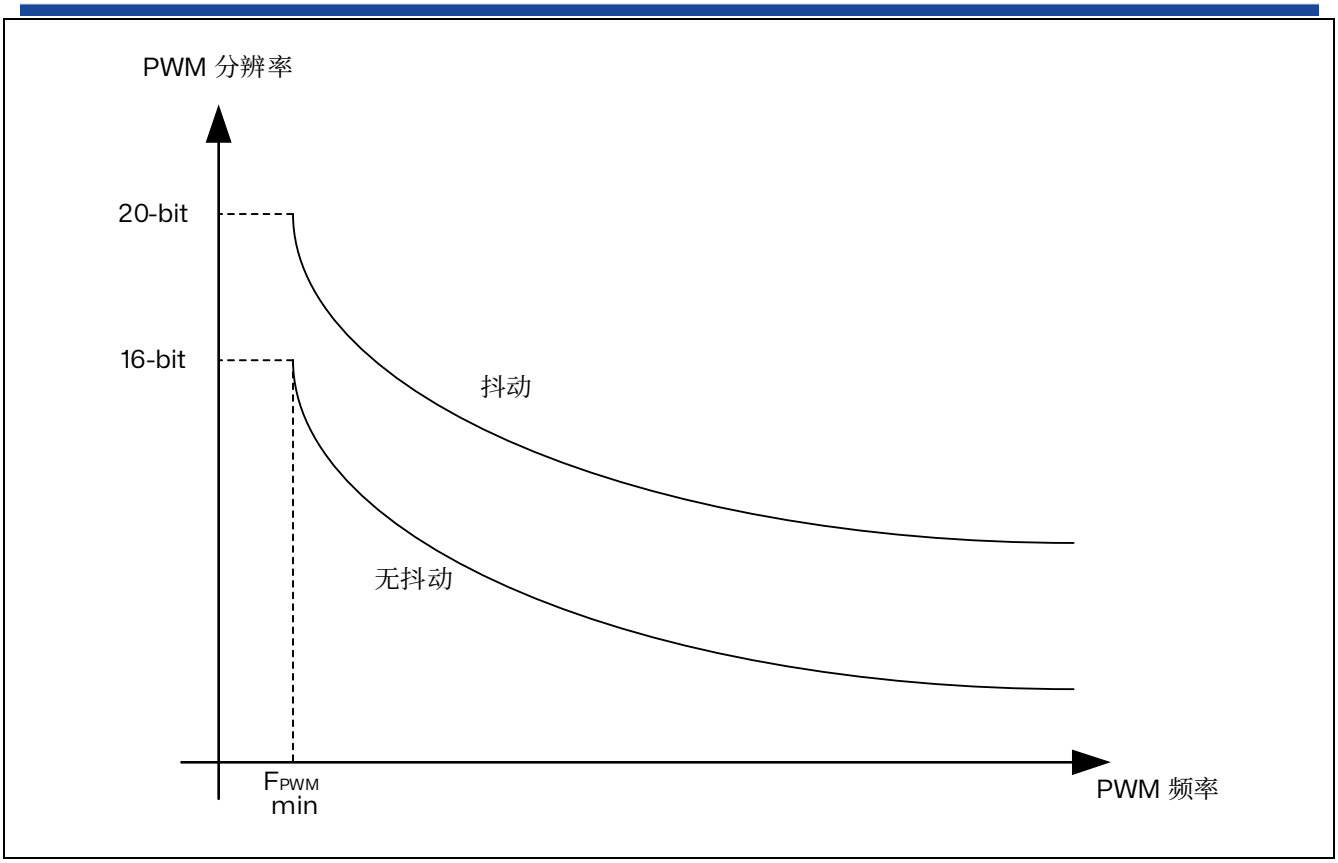


图 18.31 PWM 分辨率 vs 频率

如下图所示，占空比和/或周期的变化分散在 16 个连续周期。

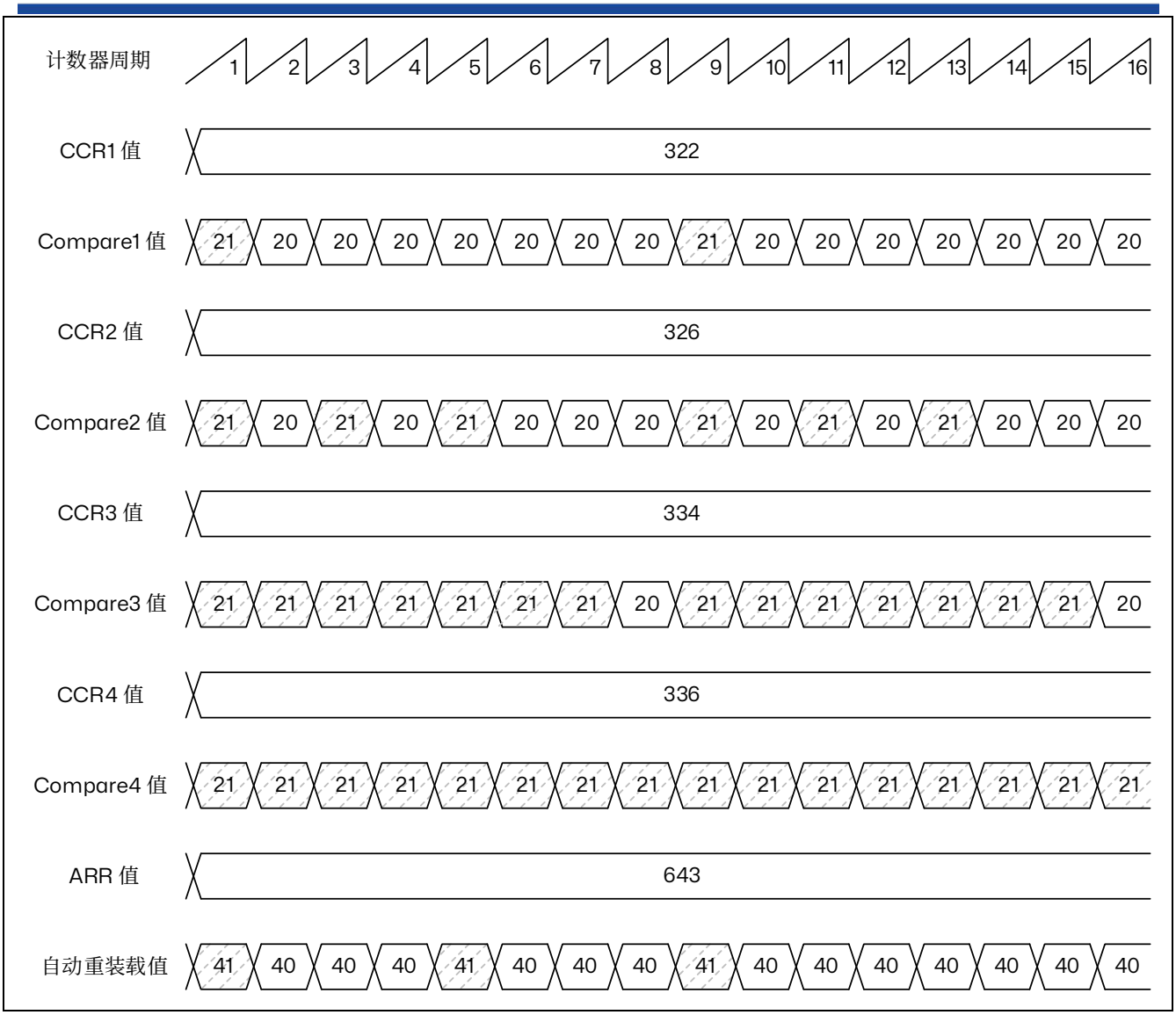


图 18.32 PWM 抖动模式

自动重新加载和比较值的增量按照下表中描述的特定模式分布。抖动序列是为了使增量分布尽可能均匀并最大限度地减少整体波动。

表 18.8 CCR 和 ARR 寄存器的变化抖动模式

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

在中心对齐 PWM 模式下（TIMx_CR1 寄存器中的 CMS 位不为“00”），也提供了抖动模式。在这种模式下，抖动模式考虑到递增递减计数阶段，应用于 8 个连续的 PWM 周期，如下图所示。

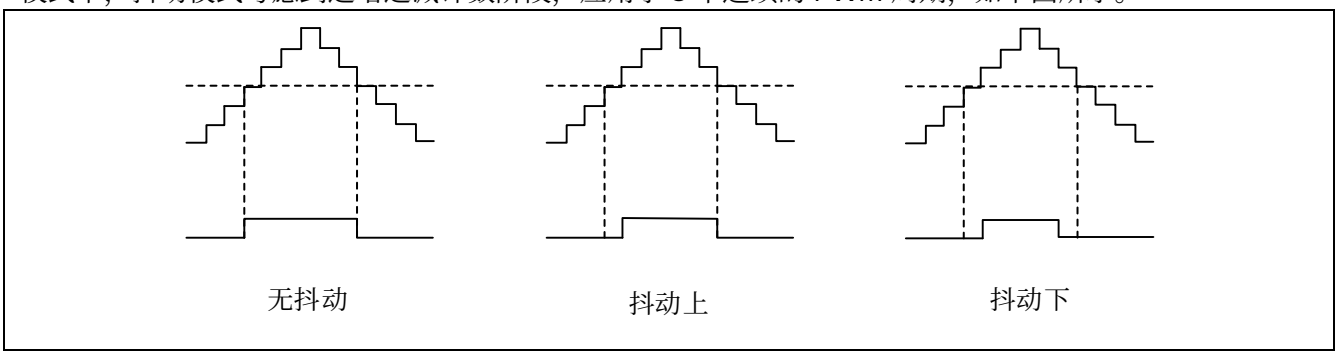


图 18.33 中心对齐 PWM 模式下抖动对占空比的影响

下表显示了如何在中心对齐 PWM 模式下添加抖动模式。

表 18.9 CCR 寄存器在中心对齐 PWM 模式下的变化抖动模式

LSB 值	PWM 周期															
	1		2		3		4		5		6		7		8	
	上	下	上	下	上	下	上	下	上	下	上	下	上	下	上	下
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

18.3.11 非对称 PWM 模式

非对称模式允许生成两个中心对齐的 PWM 信号，并具有可编程的相位偏移。虽然频率由 TIMx_ARR 寄存器的值决定，但占空比和相位偏移由一对 TIMx_CCRx 寄存器确定。一个寄存器控制 PWM 在计数上升期间，第二个寄存器在计数下降期间，这样 PWM 每半个 PWM 周期调整一次：

— tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制

— tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

通过将“1110”（非对称 PWM 模式 1）或“1111”（非对称 PWM 模式 2）写入 TIMx_CCMRx 寄存器的 OCxM 位，可以在两个通道上独立选择非对称 PWM 模式（每个 CCR 寄存器对一个 tim_ocx 输出）。

*注意：为了兼容性原因，*OCxM[3:0]*位域被分为两个部分，最高有效位与最低* 3 *位不连续。*

当某个通道用作非对称 PWM 通道时，其互补通道也可以使用。例如，如果 tim_oc1refc 信号在通道 1 上生产（非对称 PWM 模式 1），可以在通道 2 上输出 tim_oc1refc 信号，或者产生非对称 PWM 模式 1 的结果 tim_oc2refc 信号。

下图标识使用非对称 PWM 模式生成信号的示例（通道 1 到 4 配置为非对称 PWM 模式 2）。与死区时间生成器一起使用，可以控制全桥移相 DC 到 DC 转换器。

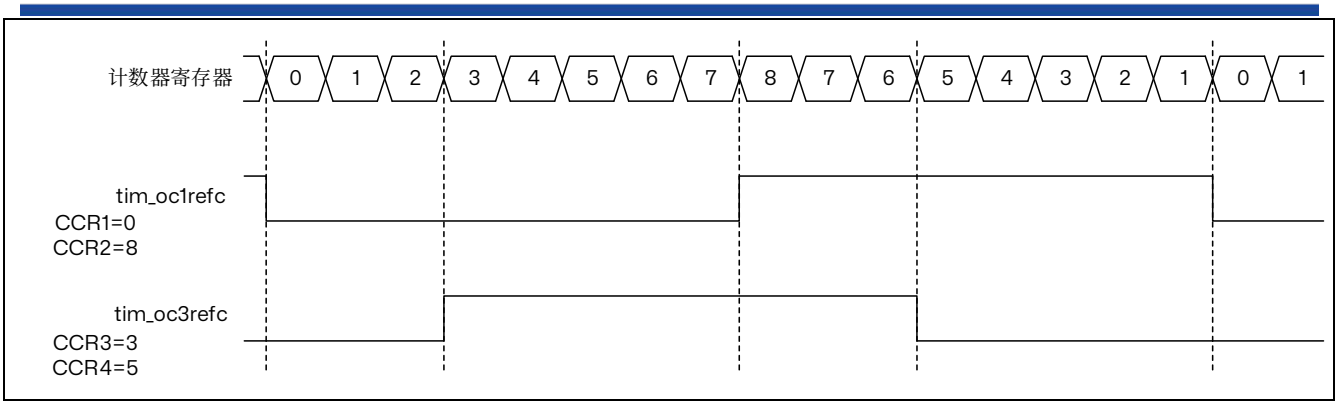


图 18.34 生成两个具有 50%占空比的相位偏移 PWM 信号

18.3.12 组合 PWM 模式

组合 PWM 模式允许生成两个边缘对齐或中心对齐的 PWM 信号，各自脉冲之间的可编程延迟和相位偏移。虽然频率由 TIMx_ARR 寄存器的值决定，但占空比和延迟由两个 TIMx_CCRx 寄存器确定。结果信号 tim_ocxrefc 由两个参考 PWM 的 OR 或 AND 逻辑组合而成：

— tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制

— tim_oc3refc（或 tim_oc4refc）由 TIMx_CCR3 和 TIMx_CCR4 控制

通过在 TIMx_CCMRx 寄存器的 OCxM 位中写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2），可以在两个通道上独立选组合 PWM 模式（每个 CCR 寄存器对一个 tim_ocx 输出）。

当某个通道用作组合 PWM 通道时，其互补通道必须配置为相反的 PWM 模式（例如，一个通道为组合 PWM 模式 1，另一个通道为组合 PWM 模式 2）。

*注意：为了兼容性原因，*OCxM[3:0]*位域被分为两个部分，最高有效位与最低*3 *位不连续。*

下图表示使用组合 PWM 模式生成信号的示例，该示例的配置如下：

— 通道 1 配置为组合 PWM 模式 2。

— 通道 2 配置为 PWM 模式 1。

— 通道 3 配置为组合 PWM 模式 2。

— 通道 4 配置为 PWM 模式 1。

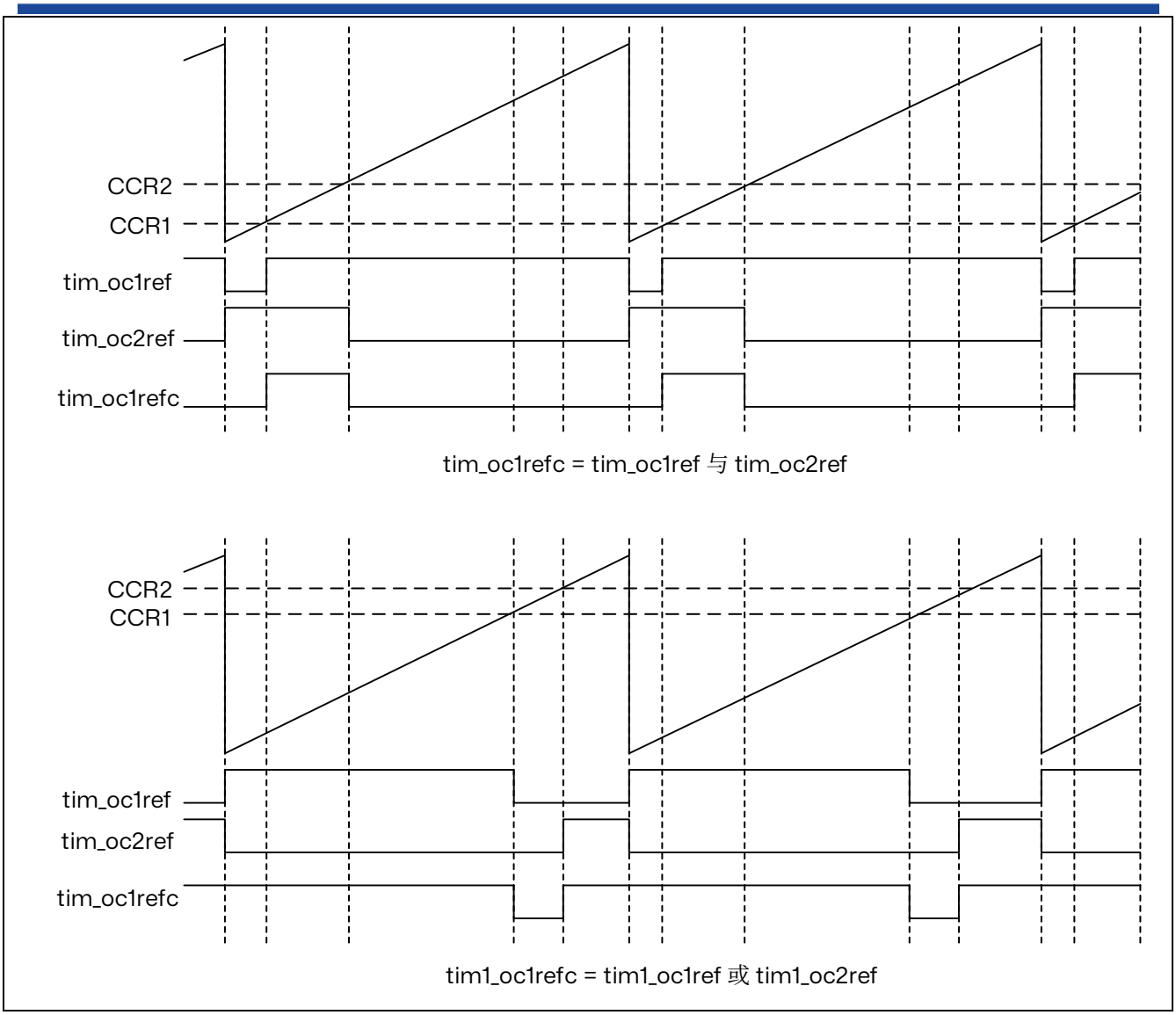


图 18.35 通道 1 和通道 3 的组合 PWM 模式

18.3.13 三相组合 PWM 模式

三相组合 PWM 模式允许生成一个到三个以脉冲中间的单个可编程信号 AND 为中心对齐的 PWM 信号。使用 tim_oc5ref 信号定义结果组合信号。TIMx_CCR5 中的 3 位 GC5C[3:1]允许选择 tim_oc5ref 与哪个参考信号组合。结果信号 tim_ocxrefc 由两个参考 PWM 的 AND 逻辑组合而成。

— 如果设置了 GC5C1，则 tim_oc1refc 由 TIMx_CCR1 和 TIMx_CCR5 控制
— 如果设置了 GC5C2，则 tim_oc2refc 由 TIMx_CCR2 和 TIMx_CCR5 控制
— 如果设置了 GC5C3，则 tim_oc3refc 由 TIMx_CCR3 和 TIMx_CCR5 控制

通过设置 3 位 GC5C[3:1]中的至少一位，可以在通道 1 到 3 上独立选择三相组合 PWM 模式。

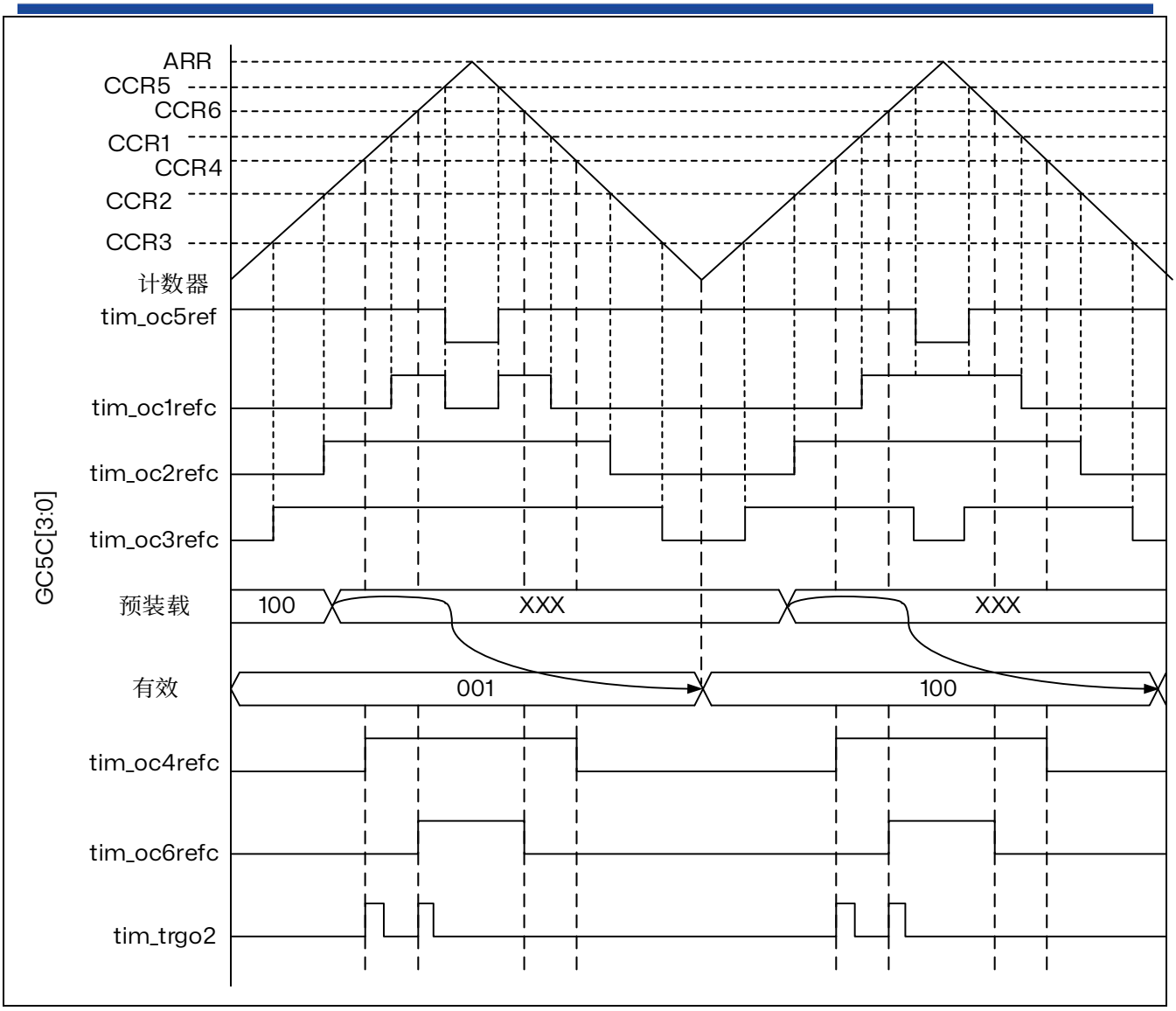


图 18.36 每周期具有多个触发脉冲的三相组合 PWM 信号

tim_trgo2 波形显示了 ADC 如何对给定的三相 PWM 信号进行同步。更多详细信息请参考本章后续小节：ADC 触发器。

18.3.14 互补输出和死区插入

高级控制定时器（TIM1/TIM8）可以输出两路互补信号，并管理输出的关断与接通瞬间。

这段时间通常称为死区，用户必须根据与输出相连接的器件及其特性（电平转换器的固有延迟、开关器件产生的延迟...）来调整死区时间。

每路输出可以独立选择输出极性（主输出 tim_ocx 或互补输出 tim_ocxn）。可通过对 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位执行写操作来完成极性选择。

互补信号 tim_ocx 和 tim_ocxn 通过以下多个控制位的组合进行激活：TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位以及 TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI、OSSR 和 OSSD 位。更多详细信息，请参考表格：带刹车功能的互补输出通道 tim_ocx 和 tim_ocxn 的控制位。应当注意，切换至 idle（MOE 下降到 0）的时刻，死区仍然有效。

死区时间插入功能是同时设置 CCxE 和 CCxNE 位以及（如果存在刹车电路）MOE 位来使能的。每

个通道都有一个 10 位的死区时间生成器。将基于参考波形 tim_ocxref 生成 2 个输出 tim_ocx 和 tim_ocxn。如果 tim_ocx 和 tim_ocxn 为高电平有效：

- 输出信号 tim_ocx 与参考信号相同，只是其上升沿相对参考上升沿存在延迟。
- 输出信号 tim_ocxn 与参考信号相反，并且其上升沿相对参考下降沿存在延迟。

如果延迟时间大于有效输出（tim_ocx 或 tim_ocxn）的宽度，则不会产生相应的脉冲。

下图所示为死区发生器的输出信号与参考信号 tim_ocxref 之间的关系。（在这些示例中，假定 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

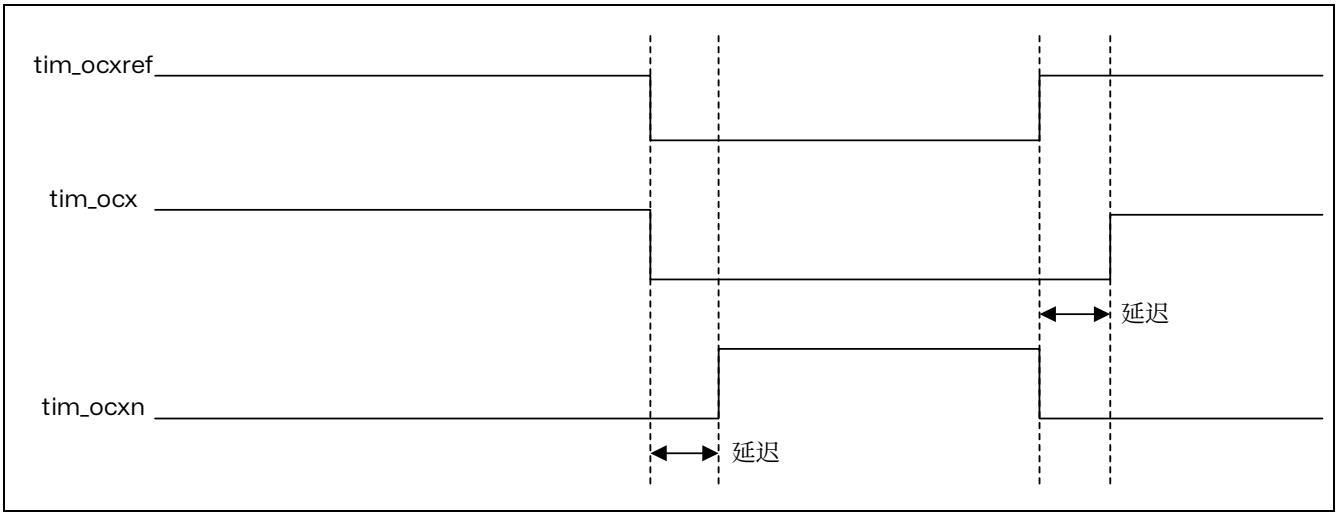


图 18.37 带死区插入的互补输出

TIMx_DTR2 中的 DTAE 位允许对参考信号上升沿和下降沿的死区值进行区分，如下图所示。

在非对称模式（DTAE=1）下，上升沿参考死区由 TIMx_BDTR 寄存器中的 DTG[7:0]位域定义，而下降沿参考死区由 TIMx_DTR2 寄存器中的 DTGF[7:0]位域定义。在使能计数器之前，必须写入 DTAE 位，并且在 CEN=1 时不得修改。

使用预加载机制可以在 PWM 操作期间动态更新死区值。当 TIMx_DTR2 寄存器中的 DTPE 位被设置时，死区位域 DTG[7:0]和 DTGF[7:0]将被预加载。预加载值将在下一个更新事件上加载到活动寄存器中。

注意：如果 DTPE *位在计数器使能时被使能。自上次更新以来写入的新值将被丢弃，并使用先前的值。*

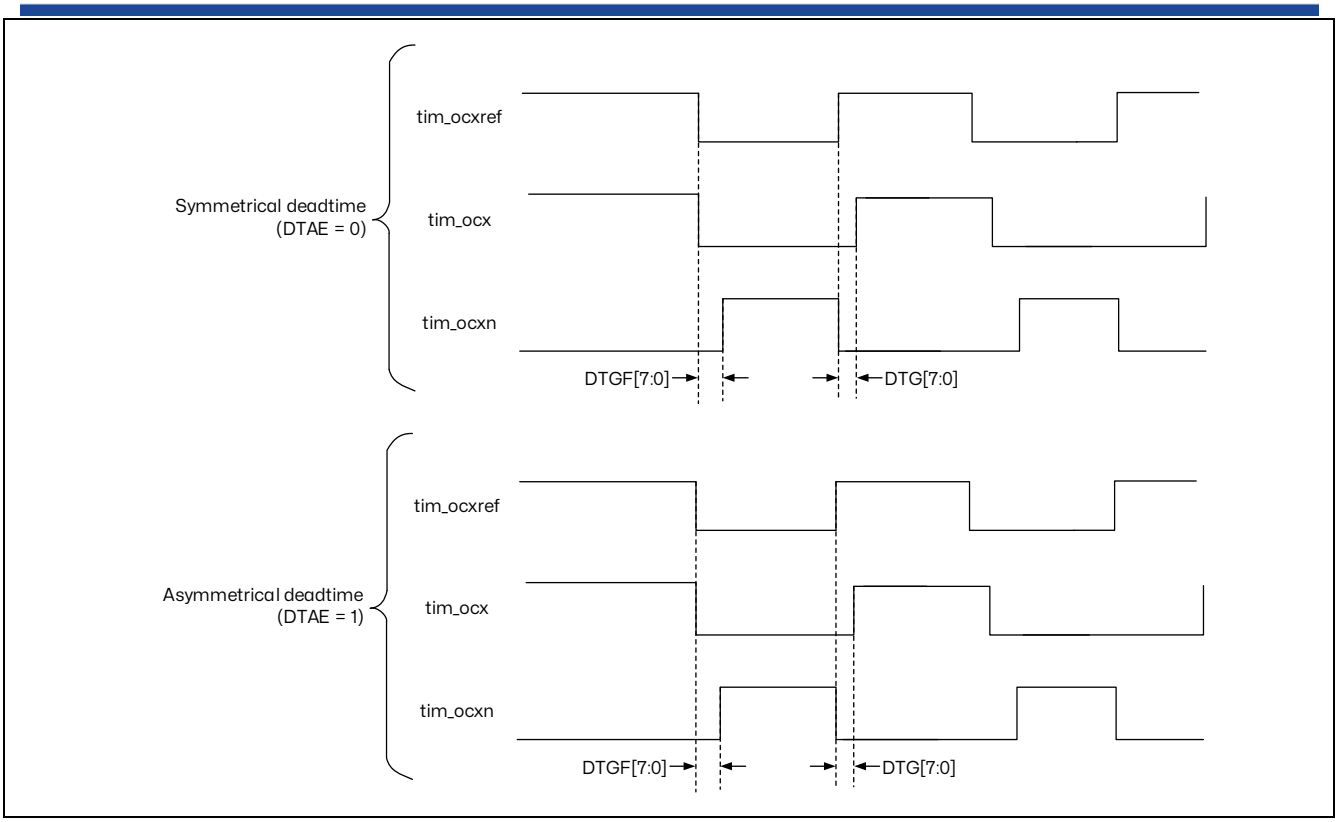


图 18.38 非对称死区时间

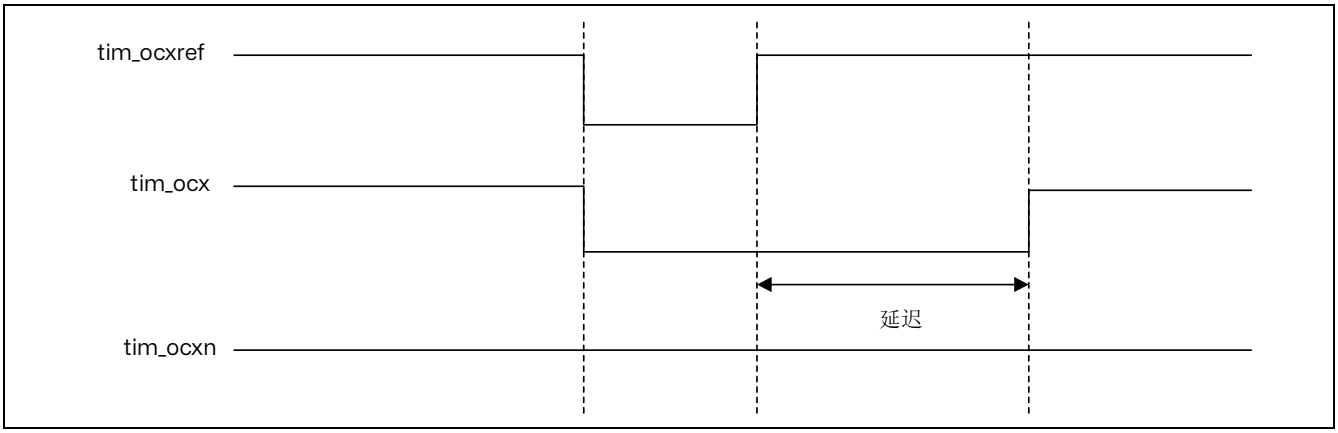


图 18.39 死区波形延迟大于负脉冲

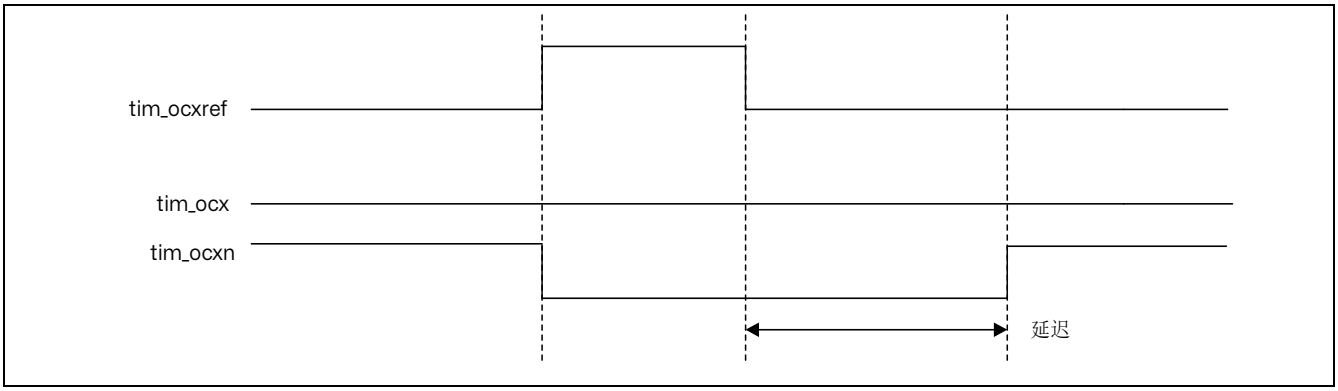


图 18.40 死区波形延迟大于正脉冲

死区延迟对于所有通道均相同，可通过 TIMx_BDTR 寄存器中的 DTG 位进行编程。有关延迟时间计算的信息，请参见后续小节：TIMx 刹车和死区寄存器（TIMx_BDTR）（x=1，8）。

将 OCxREF 重定向到 OCx 或 OCxN

在输出模式（强制输出模式、输出比较模式或 PWM 模式）下，通过配置 TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位，可将 tim_ocxref 重定向到 tim_ocx 输出或 tim_ocxn 输出。

通过此功能，可以在一个输出上发送特定波形（如 PWM 或静态有效电平），而同时使互补输出保持其无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时处于有效电平，两者互补并且带死区。

注：如果仅使能 tim_ocxn (CCxE=0, CCxNE=1)，*两者不互补，一旦* tim_ocxref *为高电平，* tim_ocxn *即变为有效。例如，如果* CCxNP=0，*则* tim_ocxn= tim_ocxref。*另一方面，如果同时使能* tim_ocxn *和* tim_ocxref (CCxE=CCxNE=1)，tim_ocx *在* tim_ocxref *为高电平时变为有效，而* tim_ocxn *则与之互补，在* tim_ocxref *为低电平时变为有效。*

18.3.15 使用刹车功能

刹车功能的目的是保护由定时器产生的 PWM 信号驱动的功率开关。两个刹车输入通常连接到功率级和三相逆变器的故障输出。当激活时，刹车电路会关闭 PWM 输出并强制它们进入预定义的安全状态。还可以选择许多内部 MCU 事件来触发输出关闭。

刹车具有两个通道。一个刹车通道可以收集系统级故障（时钟故障、ECC/奇偶校验错误，.....）和应用故障（来自输入引脚和内置比较器），并可以在死区时间后将输出强制到一个预定义的水平（无论是有效还是无效）。一个刹车 2 通道只包括应用故障，并能够将输出强制到无效状态。

刹车期间输出使能信号和输出电平取决于几个控制位：

- TIMx_BDTR 寄存器中的 MOE 位允许通过软件使能/禁用输出，并在发生刹车或刹车 2 事件时重置。
- TIMx_BDTR 寄存器中的 OSSI 位定义了定时器是否在无效状态下控制输出或将其控制权释放给 GPIO 控制器（通常是为了使其处于高阻模式）。
- TIMx_CR2 寄存器中的 OISx 和 OISxN 位设置输出关闭电平，可以是有效电平或无效电平。无论 OISx 和 OISxN 的值如何，tim_ocx 和 tim_ocxn 输出不能同时在给定时间内设置为有效电平。更多详细信息请参考后续表格：具有刹车功能的互补 tim_ocx 和 tim_ocxn 通道的输出控制位。

当退出重置状态时，刹车电路被禁用，并且 MOE 位为低电平。通过设置 TIMx_BDTR 寄存器中的 BKE 和 BK2E 位，可以使能刹车功能。通过配置相同的寄存器中的 BKP 和 BK2P 位，可以选择刹车输入极性。BKEx 和 BKPx 可以同时修改。在写入 BKEx 和 BKPx 位后，需要等待 1 个 APB 时钟周期才能使写入生效。因此，在写入操作后，需要等待 1 个 APB 时钟周期才能正确地读取该位。

由于 MOE 下降沿可能是异步的，因此在实际信号（作用于输出）和同步控制位（在 TIMx_BDTR 寄存器中访问）之间插入了一个重新同步电路。这导致异步信号和同步信号之间存在一些延迟。特别是当 MOE 被设置为 1 而之前它是低电平时，在正确读取之前必须插入一个延迟（伪指令）。这是因为写入作用于异步信号，而读取反映的是同步信号。

刹车（tim_brk）通道的源为：

- 连接到 TIMx_BKIN 引脚的外部源（根据 GPIO 备用功能选择寄存器中的选择而定），具有极性选择和可选的数字滤波
- 内部源：

— 来自 tim_brk_cmpx 输入（请参考章节：TIM1/TIM8 引脚和内部信号）
— 来自 tim_sys_brk 输入的系统中断请求（请参考章节：TIM1/TIM8 引脚和内部信号）

刹车 2（tim_brk2）的源为：

- 连接到 TIMx_BKIN2 引脚的外部源（根据 GPIO 备用功能选择寄存器中的选择而定），具有极性选择和可选的数字滤波。
- 来自 tim_brk2_cmpx 输入的内部源（请参考章节：TIM1/TIM8 引脚和内部信号）

也可以通过软件使用 TIMx_EGR 寄存器中的 BG 和 B2G 位生成刹车事件。

所有源在进入定时器的 tim_brk 或 tim_brk2 输入之前进行 OR 操作，如下图所示。

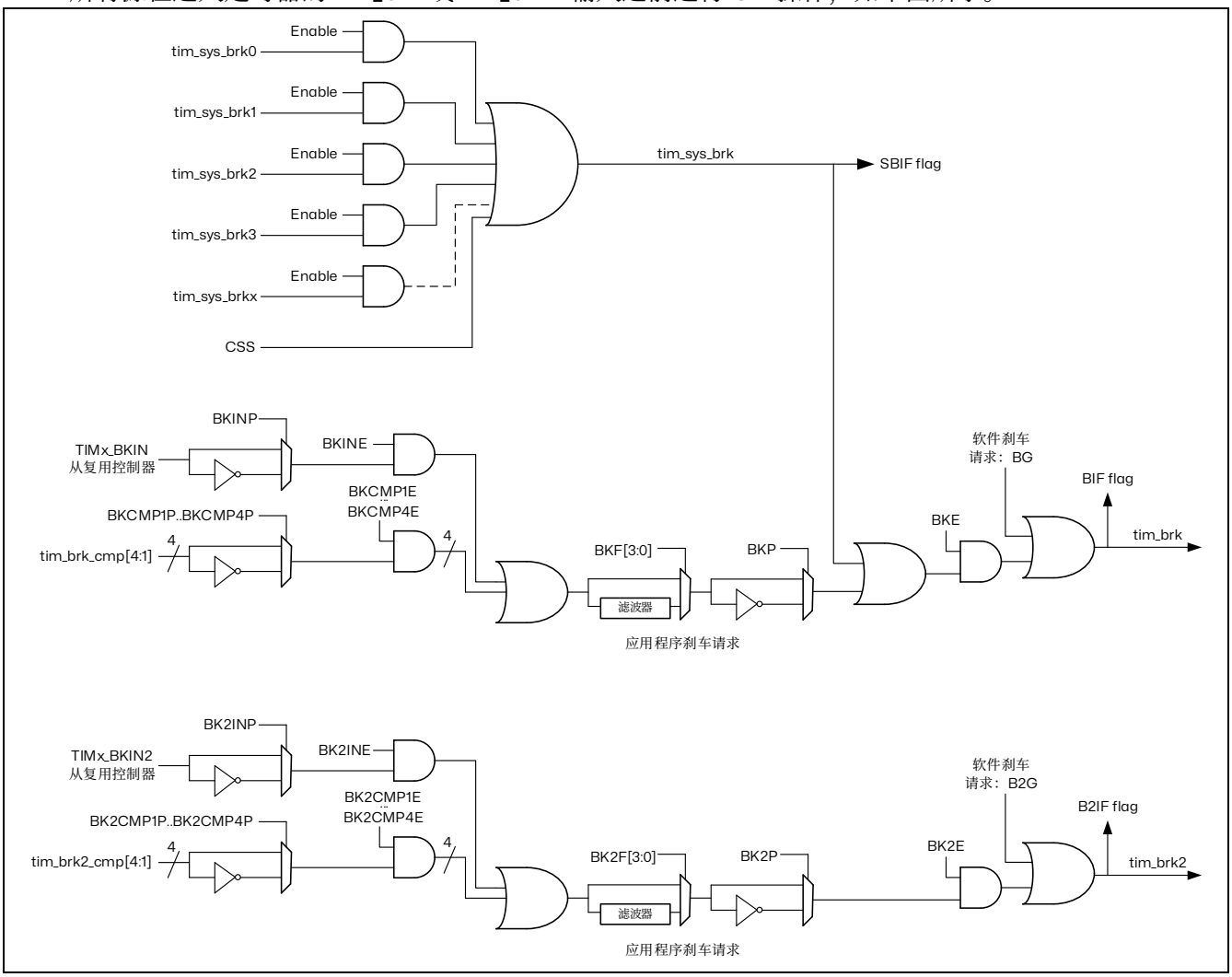


图 18.41 刹车和刹车 2 电路概述

注：只有当可编程滤波器禁用时，才保证异步（无时钟）操作。如果使能，必须使用故障安全时钟模式（例如使用内部PLL 和/或CSS）来保证刹车事件得到处理。

当其中一个刹车发生时（在所选的刹车输入线上检测到选定电平）：

- MOE 位会被异步清除，使输出处于无效状态、空闲状态或甚至将控制权释放给 GPIO 控制器（由 OSSI 位选择）。即使 MCU 振荡器关闭，此功能也会使能。
- 一旦 MOE=0，每个输出通道将驱动 TIMx_CR2 寄存器中 OISx 位编程的电平。如果 OSSI=0，定时器将释放输出控制（由 GPIO 控制器接管），否则使能输出保持高电平。
- 当使用互补输出时：

— 输出首先处于无效状态（取决于极性）。这是异步完成的，因此即使没有向定时器提供时钟也可以正常工作。

— 如果定时器的时钟仍然存在，则重新激活死区生成器，以便在死区后驱动 OISx 和 OISxN 位中编程的电平的输出。即使在这种情况下，tim_ocx 和 tim_ocxn 也不能同时驱动到其活动电平。请注意，由于在 MOE 上的重新同步，死区持续时间比平常稍长（大约 2 个 tim_ker_ck 时钟周期）。

— 如果 OSSI=0，定时器将释放输出控制（由 GPIO 控制器接管，该控制器强制高阻态），否则使能输出保持或变为高电平，一旦 CCxE 或 CCxNE 位为高电平。

- 设置刹车标志位（TIMx_SR 寄存器中的 SBIF、BIF 和 B2IF 位）。如果 TIMx_DIER 寄存器中的 BIE 位已设置，则生成中断。
- 如果 TIMx_BDTR 寄存器中的 AOE 位已设置，则在下一个更新事件（UEV）时自动重新设置 MOE 位。例如，这可以用于执行调节。否则，MOE 保持低电平，直到应用程序再次将其设置为“1”。在这种情况下，它可以用于安全，并将中断输入连接到来自功率驱动器、温度传感器或任何安全组件的警报。

注：如果 MOE 被 CPU 复位，而 AOE 位已设置，则输出处于空闲状态，并强制处于无效电平或高阻态，具体取决于 OSSI 值。如果 MOE 和 AOE 位都被 CPU 复位，则输出处于禁用状态，并驱动在 TIMx_CR2 寄存器中 OISx 位编程的电平。

注：刹车输入在电平上有效。因此，当刹车输入处于有效状态时（无论是自动还是通过软件设置），MOE 都无法设置。同时，状态标志 BIF 和 B2IF 不能清除。

除刹车输入和输出管理外，刹车电路内部还实施了写保护，用以保护应用的安全。通过该功能，用户可冻结多个参数配置（死区持续时间、tim_ocx/tim_ocxn 极性和禁止时的状态、OCxM 配置、刹车使能和极性）。可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从 3 种保护级别中进行选择。请参考章节：TIMx 刹车和死区寄存器（TIMx_BDTR）(x=1，8)。MCU 复位后只能对 LOCK 位执行一次写操作。

下图所示为输出对刹车响应行为的示例。

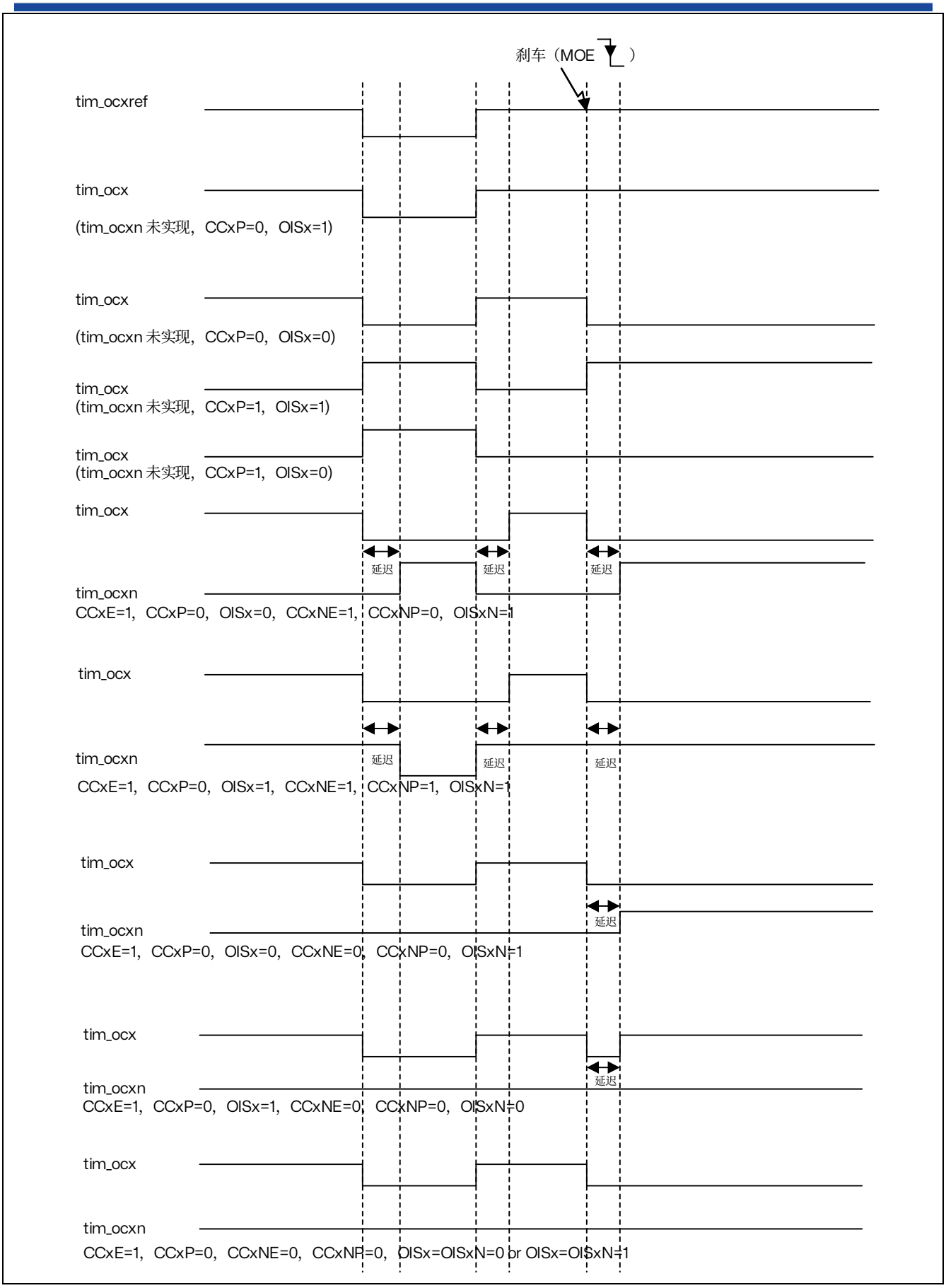


图 18.42 tim_brk 在发生刹车事件时的各种输出行为（OSSI=1）

两个中断输入在计时器输出上具有相同的行为：

— tim_brk/ tim_brk2 输入可禁止（无效状态）PWM 输出，也可将 PWM 输出强制为预定义的安全状态。

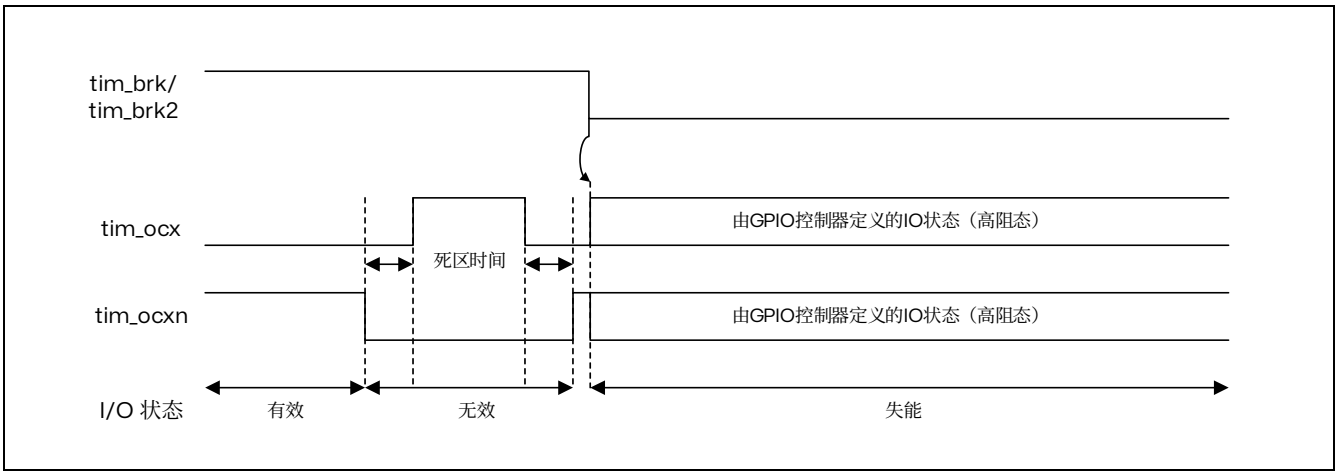


图 18.43 当 tim_brk/tim_brk2 有效（OSSI=0）时 PWM 输出状态

18.3.16 双向刹车输入

TIM1/TIM8 具有双向刹车 I/O，如下图所示。

这为以下方面提供支持：

- 板级全局刹车信号，用于向外部 MCU 或门驱动器发送故障信号，使用唯一的引脚作为输入和输出状态引脚
- 内部刹车源和多个外部开漏源 OR 连接在一起，以触发唯一的刹车事件，当多个内部和外部刹车源必须合并时

使用 TIMxBDTR 寄存器中的 BKBID 和 BK2BID 位配置 tim_brk 和 tim_brk2 输入为双向模式。BKBID 编程位可以使用 TIMxBDTR 寄存器中的 LOCK 位锁定为只读模式（在 LOCK 级别 1 或更高）。

tim_brk 和 tim_brk2 输入都支持双向模式，并且要求 I/O 配置为开漏模式，使用低电平有效极性（使用 BKINP、BKP、BK2INP 和 BK2P 位）。来自系统（例如 CSS）、片上外设或刹车输入的任何刹车请求都会强制将刹车输入置于低电平，以指示故障事件。出于安全原因，如果极性位没有正确设置（高电平有效），则双向模式将被抑制。

刹车软件事件（BG 和 B2G）也会强制将刹车 I/O 置为“0”，以指示外部组件定时器已进入刹车状态。然而，只有使能了刹车（BKE 或 B2KE = 1）时，这才有效。当使用 BKE 或 B2KE = 0 生成软件刹车事件时，输出将处于安全状态，并且会设置刹车标志，但 TIMx_BKIN 和 TIMx_BKIN2 I/O 没有影响。

安全解发机制可防止系统被彻底锁定（刹车输入低电平触发刹车，强制该输入为低电平）。

当 BKDSRM（BK2DSRM）位设置为 1 时，这将释放刹车输出以清除故障信号并允许重新给系统装载。

在任何时候都不能禁用刹车保护电路：

- 刹车输入路径始终处于有效状态：即使设置了 BKDSRM（BK2DSRM）位并释放了开漏控制，刹车事件也是有效的。这可以防止在存在刹车条件时重新启动 PWM 输出。
- 只要输出使能（MOE 位设置），BKDSRM（BK2DSRM）位就不能解除刹车保护（参见下表）。

表 18.10 刹车保护解除条件

MOE	BKBID (BK2BID)	BKDSRM (BK2DSRM)	刹车保护状态
0	0	X	装载
0	1	0	装载
0	1	1	解除
1	X	X	装载

装载和重装载刹车电路

默认情况下（外设复位配置），刹车电路（输入或双向模式）处于装载状态。
在发生刹车（刹车 2）事件后，必须按照以下步骤重新装载保护：

- BKDSRM（BK2DSRM）位必须设置为释放输出控制
- 软件必须等待系统刹车条件消失（如果有的话），并清除 SBIF 状态标志（或在重新装载之前系统地清除它）
- 软件必须轮询 BKDSRM（BK2DSRM）位，直到硬件清除该位（当应用程序刹车条件消失时）

从这一点开始，刹车电路处于装载和有效状态，并且可以设置 MOE 位以重新使能 PWM 输出。

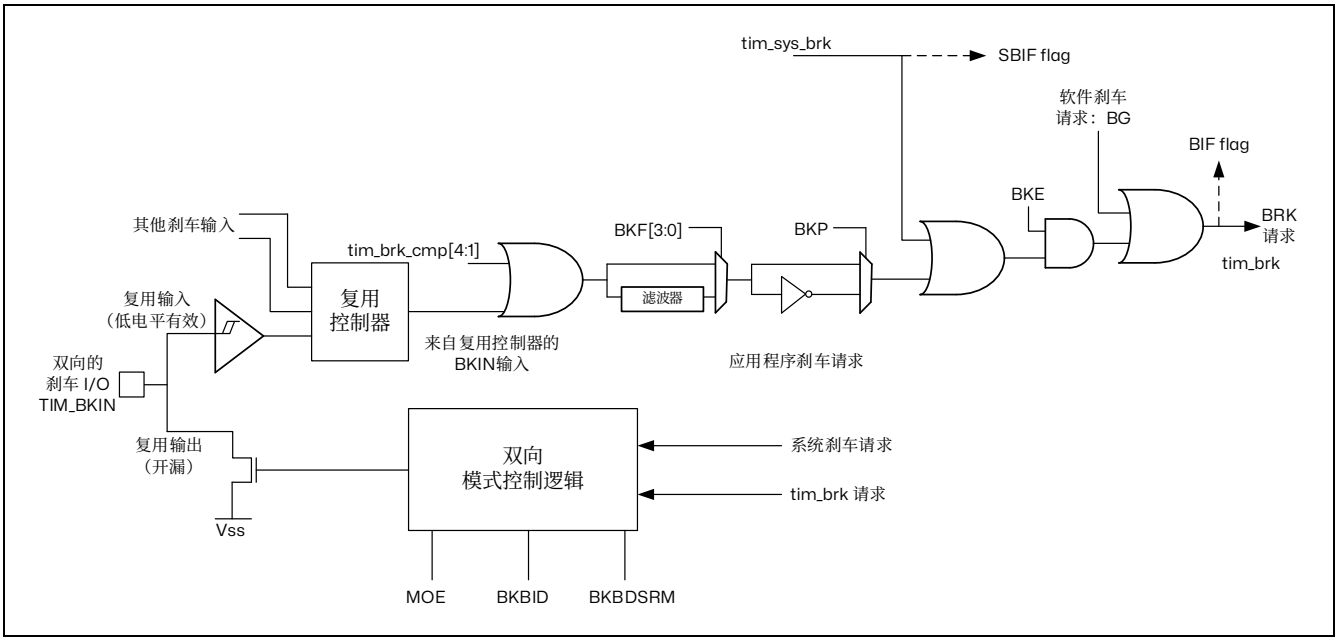


图 18.44 输出重定向（未表示 tim_brk2 请求）

18.3.17 在外部事件时清除 tim_ocxref 信号

对于给定通道，在 tim_ocref_clr_int 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可使 tim_ocxref 信号变为低电平。tim_ocxref 信号将保持低电平，直到发生下一更新事件（UEV）（特别注意，当 TIMx_RCR>0 时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。

此功能仅能用于输出比较模式和 PWM 模式，而不适用于强制输出模式。

tim_ocref_clr 输入可以在多个输入中进行选择，使用 TIMx_AF2 寄存器中的 OCRSEL[2:0]位域，如下图所示。请参考章节：TIM1/TIM8 引脚和内部信号，以获取产品中可用源的列表。

当配置组合 PWM 模式和三相组合 PWM 模式，对于给定通道，在 tim_ocref_clr_int 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可使组合结果 tim_ocxrefc 信号变为低电平。tim_ocxrefc 信号将保持低电平，直到发生下一更新事件（UEV）（特别注意，当 TIMx_RCR>0 时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。

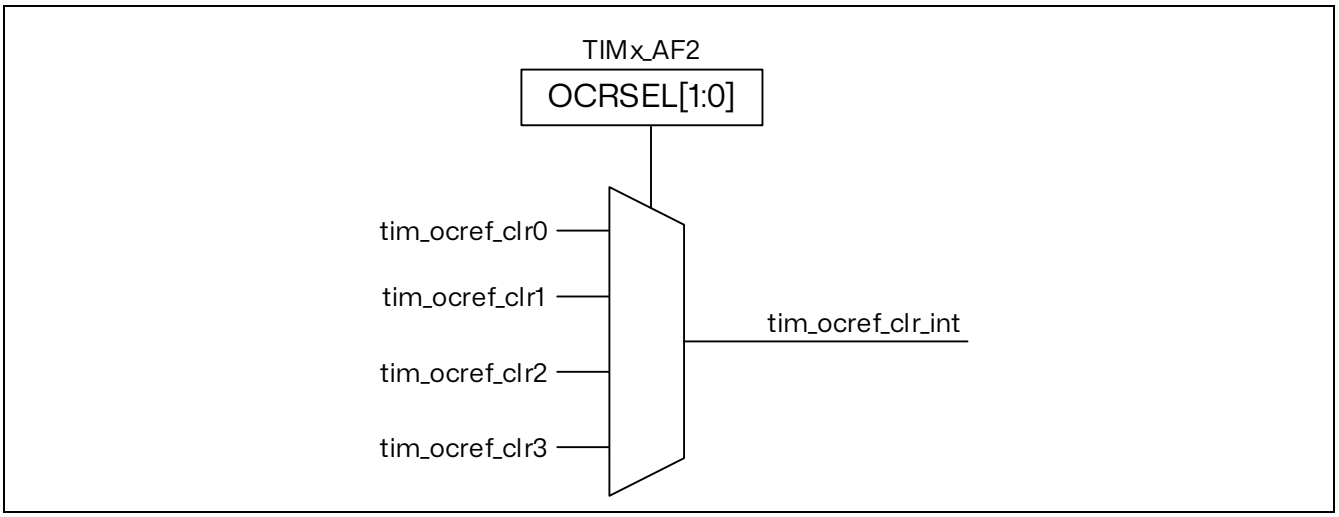


图 18.45 tim_ocref_clr 输入选择复用器

18.3.18 生成 6 步 PWM

当通道使用互补输出时，OCxM、CCxE 和 CCxNE 位上提供预装载位。发生 COM 换向事件时，这些预装载位将传输到影子位。因此，用户可以预先编程下一步骤的配置，并同时更改所有通道的配置。COM 可由软件通过将 TIMx_EGR 寄存器中的 COM 位置 1 而生成，也可以由硬件在 tim_trgi 上升沿生成。

发生 COM 事件时，某个标志位（TIMx_SR 寄存器中的 COMIF 位）将会置 1。这时将产生中断（如果 TIMx_DIER 寄存器中的 COMIE 位置 1）。

下图以 3 种不同的编程配置为例，显示了发生 COM 事件时 tim_ocx 和 tim_ocxn 输出的行为。

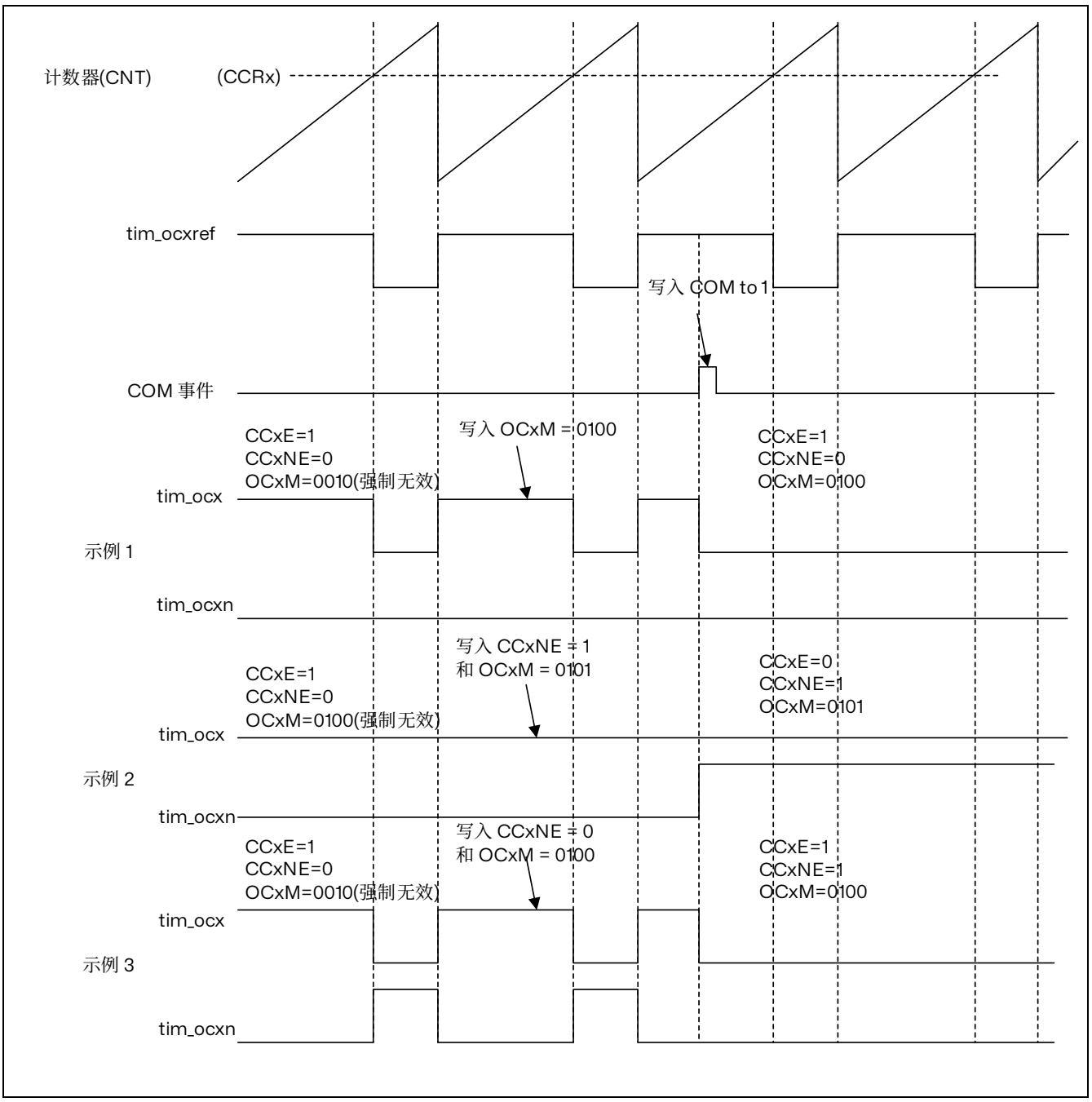


图 18.46 COM 事件生成 6 步 PWM 的示例（OSSR=1）

18.3.19 单脉冲模式

单脉冲模式（OPM）是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- 递增计数模式下：CNT < CCRx ≤ ARR（特别注意，0<CCRx）
- 递减计数模式下：CNT > CCRx

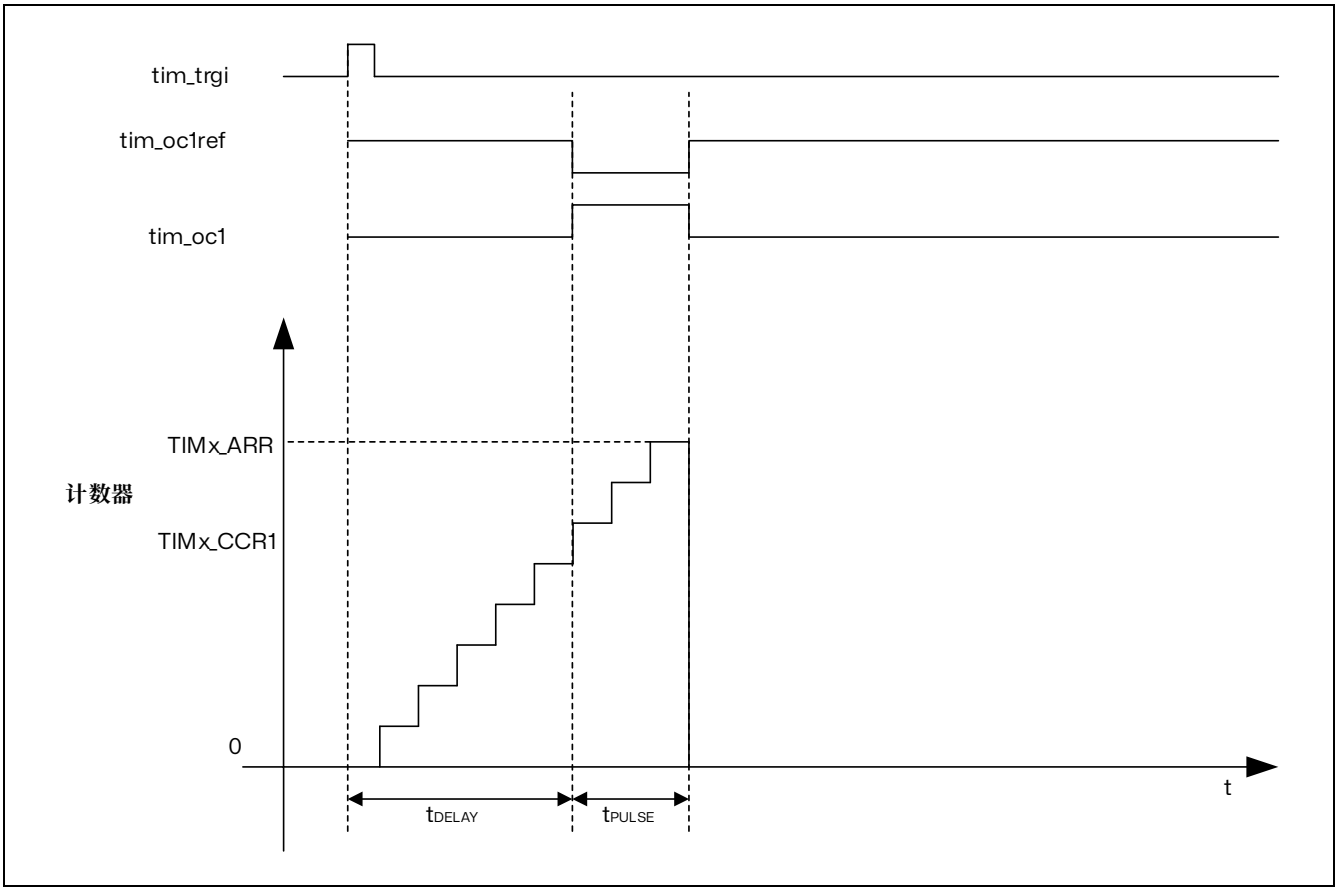


图 18.47 单脉冲模式的例子

例如，当 tim_trgi 信号到来时，经过 t_{DELAY} 的延迟，在 tim_oc1 上产生一个长度为 t_{PULSE} 的正脉冲。使用 tim_itr0 作为触发：

1. 在 TIMx_SMCR 寄存器中写入 TS=0000，将 tim_itr0 配置为从模式控制器的触发（tim_trgi）。
2. 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），使用 tim_itr0 启动计数器。

OPM 波形通过对比较寄存器执行写操作来定义（考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIMx_CCR1 寄存器的值定义。
- t_{PULSE} 由自动重载值与比较值（TIMx_ARR - TIMx_CCR1）之差来定义。
- 假设希望产生这样的波形：信号在发生比较匹配时从“0”变为“1”，在计数器达到自动重载值时由“1”变为“0”。为此，应在 TIMx_CCMR1 寄存器中写入 OC1M=111，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入

“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 tim_trgo 触发事件。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应为低。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位清“0”时，即选择重复模式。

特例：tim_ocx 快速使能：

在单脉冲模式下，tim_tix 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 tim_ocxref（和 tim_ocx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用

18.3.20 可重触发单脉冲模式

此模式允许计数器在接收到刺激后开始计数，并生成一个具有可编程长度的脉冲，但与上一节中描述的非重触发单脉冲模式存在以下差异：

— 只要触发器发生，脉冲就会立即开始（没有可编程的延迟）

— 如果在先前触发的脉冲完成之前发生新的触发，则脉冲会被延长

定时器必须处于从模式，TIMx_SMCR 寄存器的 SMS[3:0]位设置为“1000”（组合重置+触发模式），OCxM[3:0]位设置为“1000”或“1001”，用于可重触发 OPM 模式 1 或 2。

如果定时器配置为递增计数模式，则相应的 CCRx 必须设置为 0（ARR 寄存器设置脉冲长度）。如果定时器配置为递减计数模式，CCRx 必须大于或等于 ARR。

注：OCxM[[3:0]和 SMS[3:0]位域由于兼容性的原因被分成两个部分，最高有效位与最低 3 位不是连续的。OCxM 这种模式不能与中央对齐 PWM 模式一起使用。TIMx_CR1 寄存器中的 CMS[1:0]必须设置为 00。

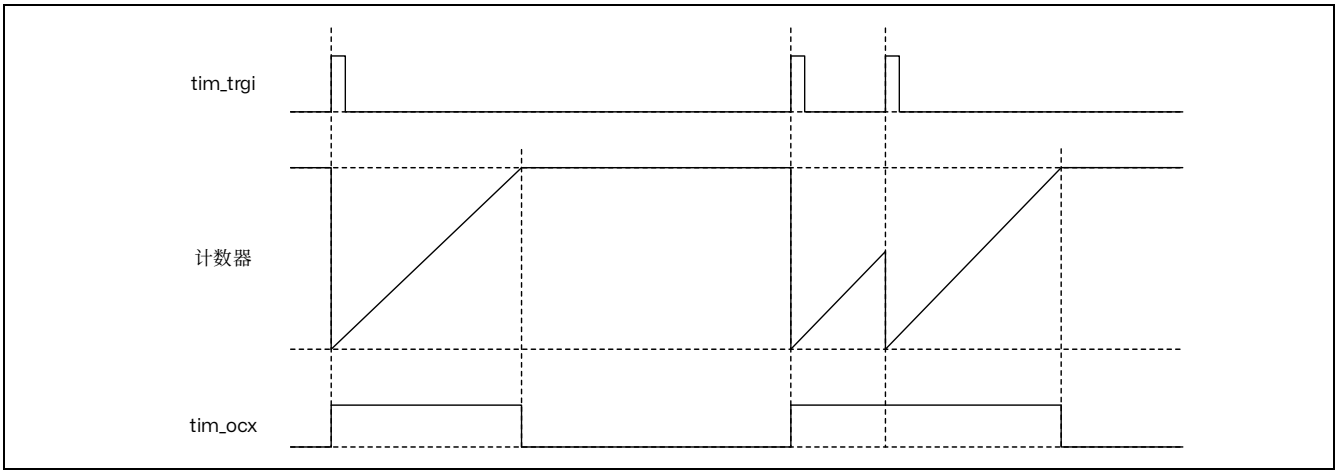


图 18.48 可重触发单脉冲模式

18.3.21 在比较模式下的脉冲

在比较匹配事件发生时，可以生成一个脉冲。当计数器的值等于给定的比较值时，可以生成一个具有可编程脉冲宽度的信号，用于调试或同步。

此模式适用于任何从机模式选择，包括边沿对齐和中心对齐计数模式。它仅适用于通道 3 和通道 4。脉冲发生器是唯一的，由两个通道共享，如下图所示。

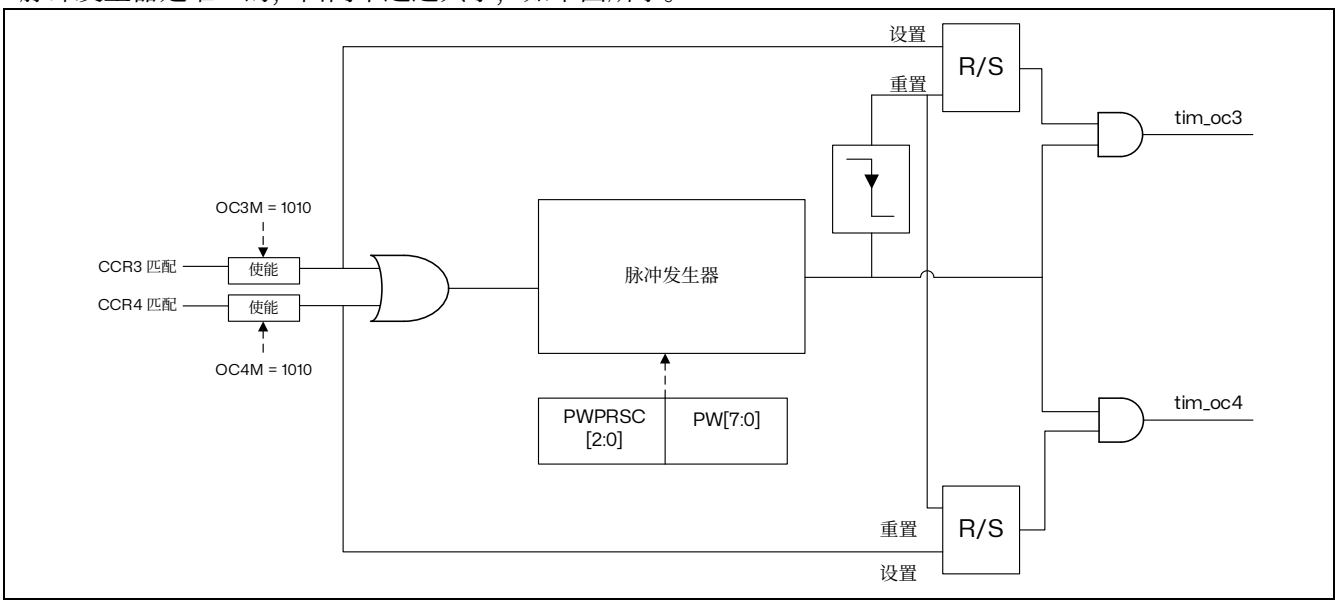


图 18.49 脉冲发生器电路

下图展示了如何在边沿对齐模式下生成脉冲。

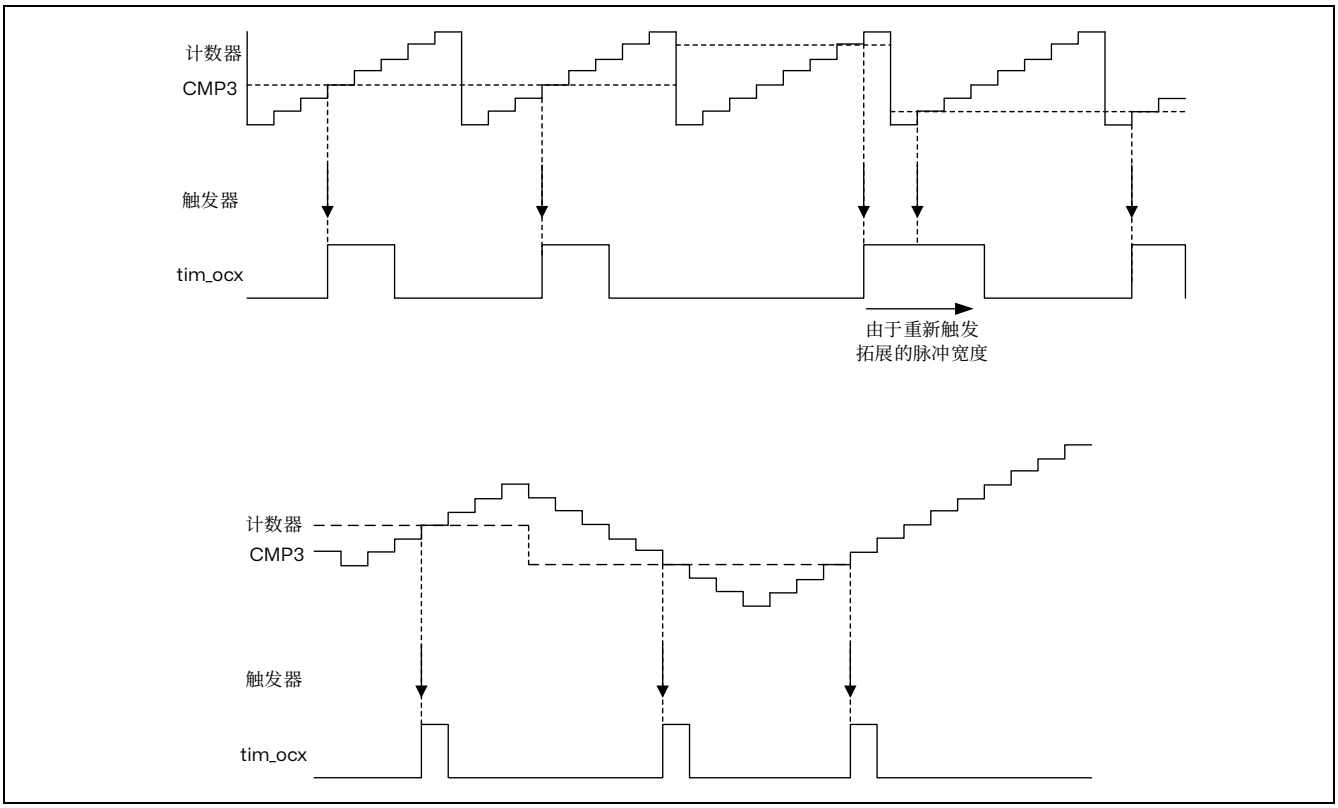


图 18.50 在比较事件上的脉冲生成，适用于边沿对齐模式

此输出比较模式是通过 TIMx_CCMR2 寄存器中的 OC3M[3:0]和 OC4M[3:0]位域进行选择的。

脉冲宽度通过寄存器中的 PW[7:0]位域进行编程，使用特定的时钟，并根据 PWPRSC[2:0]位的预设进行预分频，具体如下：

$t_{PW} = PW[7:0] \times t_{PWG}$

其中 $t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$

给出了分辨率和最大值，具体取决于预分频器的值。

脉冲可以重新触发：在脉冲正在进行时触发新的脉冲，会导致脉冲被延长。

注：如果同时使能两个通道，只要一个通道上的触发器不与并发输出上生成的脉冲重叠，脉冲就会独立发出。相反，如果两个触发器重叠，与第一个到达的触发器相关的脉冲宽度会被延长（由于重新触发），而最后一个到达的触发器的脉冲宽度是正确的（如下图所示）。

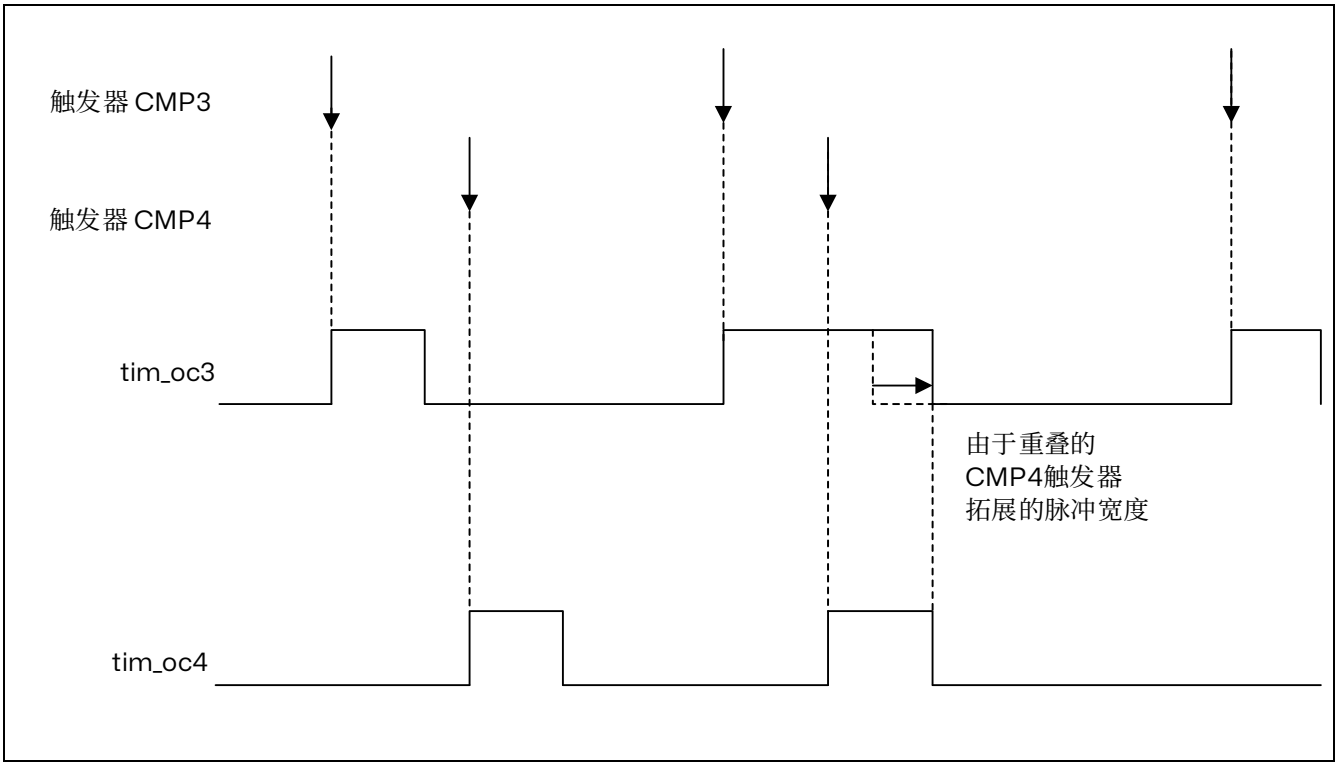


图 18.51 在同时触发的情况下延长脉冲宽度

18.3.22 方向位输出

可以将方向信号从定时器输出到 tim_oc3n 和 tim_oc4 输出信号（TIMx_CR1 寄存器中的 DIR 位的副本）。这是通过将 TIMx_CCMR2 寄存器中的 OC3M[3:0]或 OC4M[3:0]位域设置为 1011 来实现的。

18.3.23 UIF 位重映射

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许以原子方式读取计数器的值和一个潜在的由 UIFCPY 标志指示的翻转条件。在特定情况下，它可以避免由背景任务（读取计数器）和中断（更新中断）之间的处理引起的竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间没有延迟。

18.3.24 定时器同步

TIM1/8 定时器在内部连接在一起，用于定时器同步或链接。它们可以在几种模式下同步：复位模式，门控模式，触发模式，复位+触发和门控+复位模式。

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 TIM1/8_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIM1/8_ARR，TIM1/8_CCRx）都被更新了。

在以下的例子中，tim_itr1 内部触发输入端接收到 tim3_ trgo 触发信号导致向上计数器被清零：

- 置 TIM3_CR2 寄存器中 MMS=010，选择更新事件作为触发输出（tim3_ trgo）
- 置 TIM3_CR1 寄存器中 CEN=1，启动计数器。
- 置 TIM1/8_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM1/8_SMCR 寄存器中 TS=0001，选择 tim_itr1 作为输入源。
- 置 TIM1/8_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 tim_itr1 出现一个 tim3_ trgo 触发信号；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIM1/8_SR 寄存器中的 TIF 位）被设置，中断请求可以被发送（取决于 TIM1/8_DIER 寄存器中的 TIE 位）。

下图显示当自动重装载寄存器 TIM1/8_ARR=0x36 时的动作。

在 tim_itr1 触发信号和计数器的实际复位之间的延时取决于 tim_itr1 输入端的重同步电路。

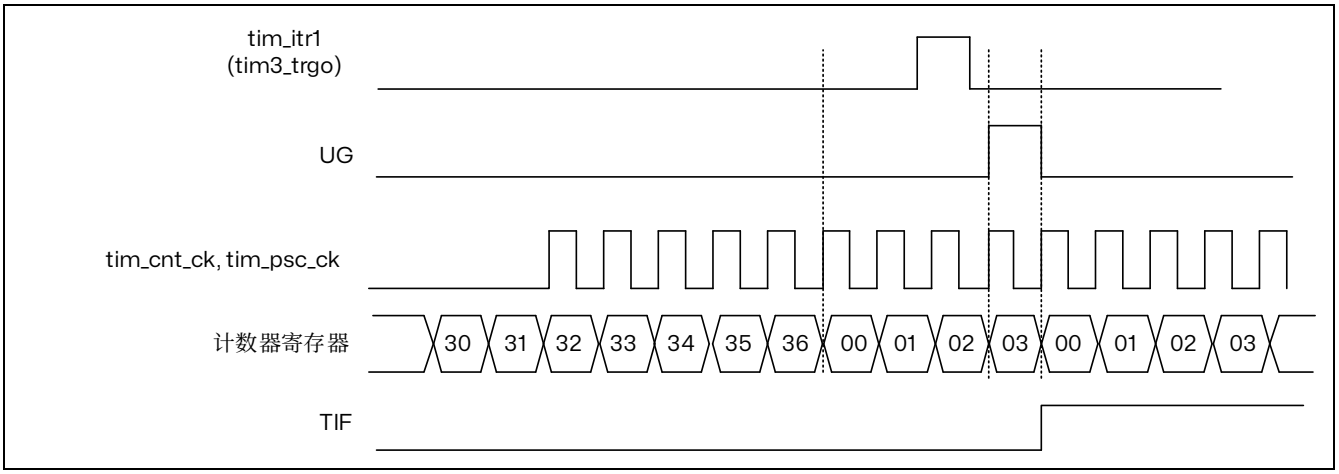


图 18.52 复位模式下的控制电路

从模式：门控模式

按照选中的**输入端有效电平**使能计数器。

在如下的例子中，计数器只在 tim_itr1（选 oc1refc 信号用作 tim3_trgo）电平有效时向上计数：

- 置 TIM3_CR2 寄存器中 MMS=100，选择 oc1refc 信号用作触发输出（tim_trgo）
- 置 TIM3_CCMR1 寄存器中 OC1M=0101，强制 tim_oc1ref 变为有效电平输出（tim_oc1refc）
- 置 TIM3_CCMR1 寄存器中 OC1M=0100，强制 tim_oc1ref 变为无效电平输出（tim_oc1refc）
- 置 TIM1/8_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM1/8_SMCR 寄存器中 TS=0001，选择 tim_itr1 作为输入源。
- 置 TIM1/8_CR1 寄存器中 CEN=1，启动计数器（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）。

只要 tim_itr1 **电平有效**，计数器开始依据内部时钟计数，一旦 tim_itr1 **电平无效**则停止计数。当计数器开始或停止时都设置 TIM1/8_SR 中的 TIF 标置。

tim_itr1 触发信号和计数器实际停止之间的延时取决于 tim_itr1 输入端的重同步电路。

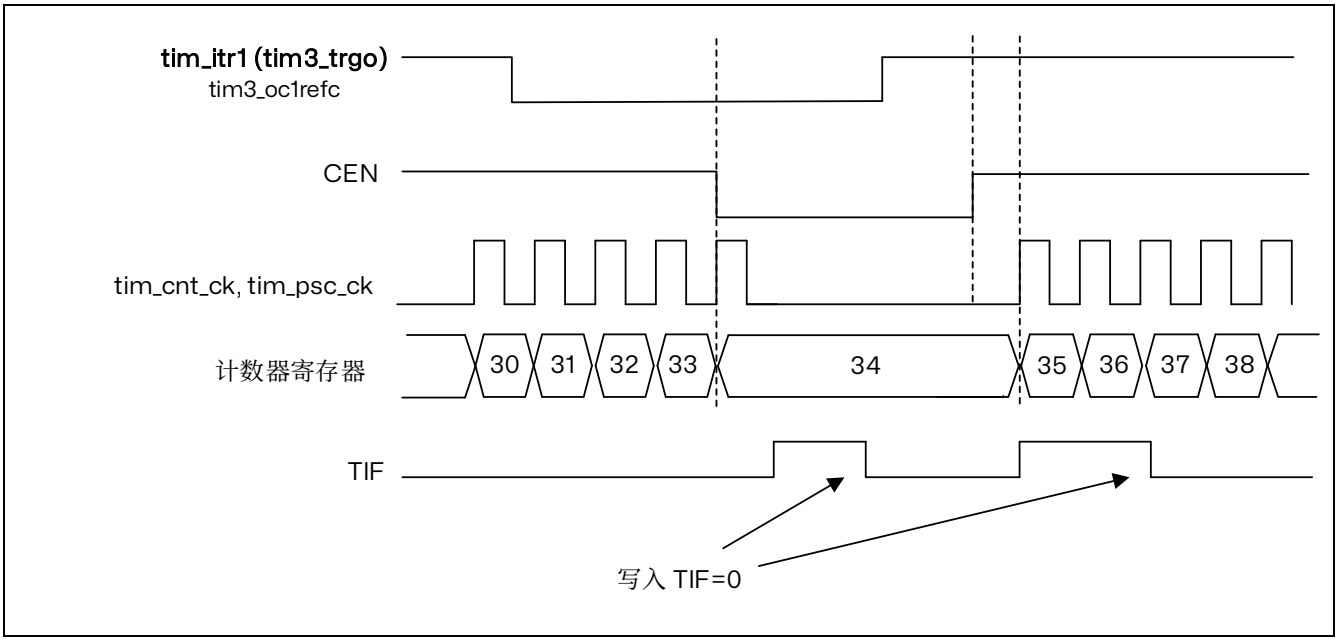


图 18.53 门控模式下的控制电路

从模式：触发模式

输入端上选中的事件使能计数器。

在以下的例子中，计数器在 tim_itr1 内部触发输入端接收到 tim3_ trgo 触发信号开始向上计数：

- 置 TIM3_CR2 寄存器中 MMS=001，选择计数器使能信号 CNT_EN 用作触发输出（tim3_trgo）
- 置 TIM1/8_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM1/8_SMCR 寄存器中 TS=0001，选择 tim_itr1 作为输入源。
- 置 TIM3_CR1 寄存器中 CEN=1，启动计数器，同步发出 tim3_trgo。

当 tim_itr1 出现 tim3_ trgo 触发信号时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

tim_itr1 触发信号和计数器启动计数之间的延时，取决于 tim_itr1 输入端的重同步电路。

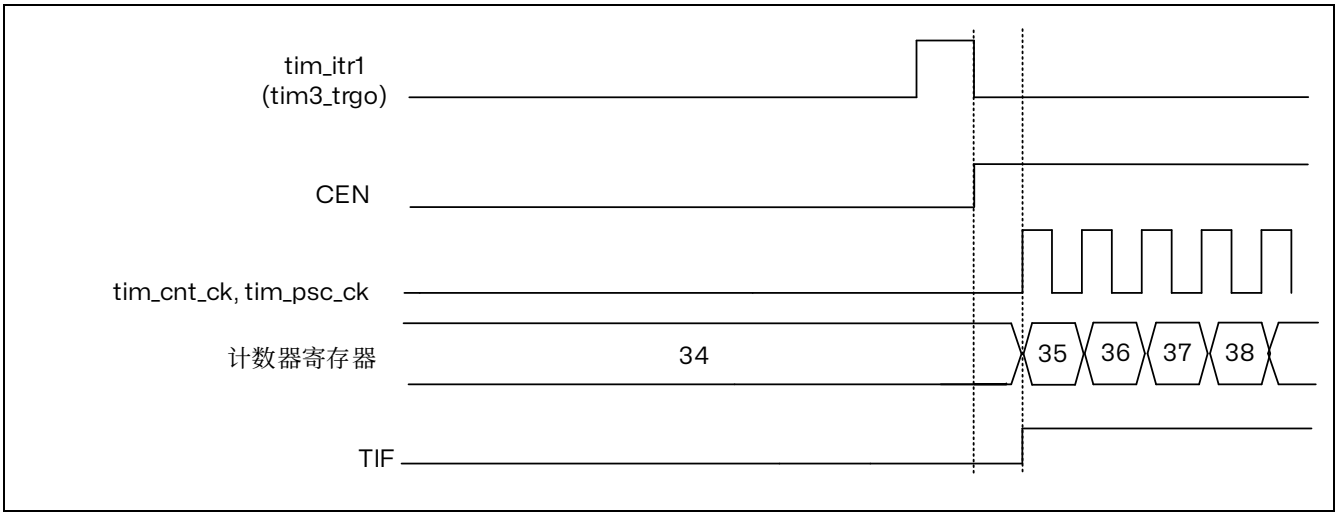


图 18.54 触发器模式下的控制电路

从模式：组合复位+触发模式

在这种情况下，当检测到选择的触发输入（tim_trgi）对应的触发信号就重新初始化计数器，生成寄存器的更新，并启动计数器。

从模式：组合门控+复位模式

触发输入（tim_trgi）电平有效时使能计数器。一旦触发电平无效，计数器停止并复位。计数器的开始和停止都是受控的。

18.3.25 ADC 触发器

定时器可以用各种内部信号（例如重置、使能或比较事件）生成 ADC 触发事件。也可以生成由内部边缘检测器发出的脉冲。例如：

- OC4ref 的上升沿和下降沿
- OC5ref 的上升沿或 OC6ref 的下降沿

触发器在 tim_trgo2 内部线路发出，该线路被重定向到 ADC。共有 16 个可能的事件，可以使用 TIMx_CR2 寄存器中的 MMS2[3:0]位选择。前章节：三相组合 PWM 模式中给出了一个 3 相马达驱动的应用示例。

注：接收 tim_trgo *或* tim_trgo2 *信号的从属外设（定时器、*ADC *等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能实时更改。*

*注：*ADC *的时钟必须在从主定时器接收事件之前使能，并且在接收到定时器的触发器时不能更改*

18.3.26 调试模式

当微控制器进入调试模式（内核停止）时，TIMx 计数器的行为取决于 DBG 模块中的 DBG_TIMx_STOP 配置位，它可以继续正常工作或停止。

在调试模式下，每个定时器的行为可以通过 Debug 支持（DBG）模块的专用配置位进行编程。

出于安全考虑，当计数器停止时，输出被禁用（就像将 MOE 位复位一样）。输出可以强制处于无效状态（OSSI 位=1），或者由 GPIO 控制器接管控制（OSSI 位=0）以强制其处于高阻态。

有关更多详细信息，请参阅调试部分。

18.4 TIM1/TIM8 低功耗模式

表 18.11 低功耗模式对 TIM1/TIM8 的影响

模式	描述
睡眠	无影响，外设是运行的。中断会导致设备退出睡眠模式。
停机	定时器操作被停止并保留其寄存器内容。无法生成中断。
待机	定时器被断电并且退出待机模式后必须重新初始化。

18.5 TIM1/TIM8 中断

TIM1/TIM8 可以生成多个中断，如下表所示。

表 18.12 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	从睡眠模式退出	从停机和待机退出
TIM_UP	更新	UIF	UIE	UIF 写 0	是	否
TIM_CC	捕获/比较 1	CC1IF	CC1IE	CC1IF 写 0	是	否
	捕获/比较 2	CC2IF	CC2IE	CC2IF 写 0	是	否
	捕获/比较 3	CC3IF	CC3IE	CC3IF 写 0	是	否
	捕获/比较 4	CC4IF	CC4IE	CC4IF 写 0	是	否
TIM_TRG_COM	换向 (COM)	COMIF	COMIE	COMIF 写 0	是	否
	触发	TIF	TIE	TIF 写 0	是	否
TIM_BRK	刹车	BIF	BIE	BIF 写 0	是	否
	刹车 2	B2IF		B2IF 写 0	是	否
	系统中断	SBIF		SBIF 写 0	是	否

18.6 TIM1/TIM8 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。

18.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 1，8）

偏移地址：0x000

复位值：0x0000

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

Bit 15:13 保留，必须保持为复位值。

Bit 12 **DITHEN**：使能抖动（Dithering enable）

0：抖动禁止

1：抖动使能

*注：*DITHEN *位只能在*CEN *位复位时修改*

Bit 11 **UIFREMAP**：UIF 状态位重映射（UIF status bit remapping）

0：无重映射。UIF 状态位未复制到 TIMx_CNT 寄存器位 31

1：重映射使能。UIF 状态位复制到 TIMx_CNT 寄存器位 31

Bit 10 保留，必须保持为复位值。

Bit 9:8 **CKD[1:0]**：时钟分频（Clock division）

此位域指示定时器时钟（tim_ker_ck）频率与死区发生器以及数字滤波器（tim_tix）所使用的死区及采样时钟（t_{DTS}）之间的分频比

00：$t_{DTS} = t_{tim_ker_ck}$

01：$t_{DTS} = 2 \times t_{tim_ker_ck}$

10：$t_{DTS} = 4 \times t_{tim_ker_ck}$

11：保留，不要设置成此值

Bit 7 **ARPE**：自动重载预装载使能（Auto-reload preload enable）

0：TIMx_ARR 寄存器不进行缓冲；

1：TIMx_ARR 寄存器进行缓冲。

Bit 6:5 **CMS[1:0]**：中心对齐模式选择（Center-aligned mode selection）

00：边沿对齐模式。计数器根据方向位（DIR）递增计数或递减计数。

01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志都会置 1。

注：只要计数器处于使能状态 (CEN=1)，*就不得从边沿对齐模式切换为中心对齐模式。*

Bit 4 **DIR**：方向（Direction）

0：计数器递增计数；

1：计数器递减计数。

注：当定时器配置为中心对齐模式或编码器模式时，该位为只读状态。

Bit 3 **OPM**：单脉冲模式（One pulse mode）

0：计数器在发生更新事件时不会停止计数；

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

Bit 2 **URS**：更新请求源（Update request source）

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断。此类事件包括：

- 计数器上溢/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断。

Bit 1 **UDIS**：更新禁止（Update disable）

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新（UEV）事件可通过以下事件之一生成：

- 计数器上出/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

Bit 0 **CEN**：计数器使能（Counter enable）

0：禁止计数器；

1：使能计数器。

*注：只有事先通过软件将*CEN *位置*1*，才可以使用外部时钟和门控模式。而触发模式可通过硬件自动将*CEN *位置*1*。*

18.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 1，8）

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						MMS[3]	Res	MMS2[3:0]				Res	OIS6	Res	OIS5
						rw		rw	rw	rw	rw		rw		rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OIS4N	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	Res	MMS[2:0]			Res	CCUS	Res	CCPC
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw		rw		rw

Bit 31:26 保留，必须保持为复位值。

Bit 24 保留，必须保持为复位值。

Bit 23:20 **MMS2[3:0]**：主模式选择 2（Master mode selection 2）

这些位允许将信息发送到 ADC 进行同步（tim_trgo2）以进行选择。组合如下：

0000：**重置** - 来自 TIMx_EGR 寄存器的 UG 位用作触发器输出（tim_trgo2）。如果重置是由触发器输入（配置为重置模式的从机模式控制器）生成的，则 tim_trgo2 上的信号与实际重置相比会有所延迟。

0001：**使能** - 计数器使能信号 CNT_EN 用作触发器输出（tim_trgo2）。它对于同时启动多个定时器或在一段时间内控制从机定时器的使能窗口是有用的。当在门控模式下配置时，计数器使能信号是由 CEN 控制位和触发器输入之间的逻辑 AND 生成的。当计数器使能信号由触发器输入控制时，除选择了主/从模式的情况外（请参见 TIMx_SMCR 寄存器中的 MSM 位描述），tim_trgo2 上会有延迟。

0010：**更新** - 选择更新事件作为触发器输出（tim_trgo2）。例如，主定时器可以用作从机定时器的预分频器。

0011：**比较脉冲** - 当 CC1IF 标志被设置时（即使它已经是高电平），当发生捕获或比较匹配时，触发输出会发送一个正脉冲（tim_trgo2）。

0100：**比较** - tim_oc1refc 信号用作触发器输出（tim_trgo2）

0101：**比较** - tim_oc2refc 信号用作触发器输出（tim_trgo2）

0110：**比较** - tim_oc3refc 信号用作触发器输出（tim_trgo2）

0111：**比较** - tim_oc4refc 信号用作触发器输出（tim_trgo2）

1000：**比较** - tim_oc5refc 信号用作触发器输出（tim_trgo2）

1001：**比较** - tim_oc6refc 信号用作触发器输出（tim_trgo2）

1010：**比较脉冲** - tim_oc4refc 上升或下降沿在 tim_trgo2 上生成脉冲

1011：**比较脉冲** - tim_oc6refc 上升或下降沿在 tim_trgo2 上生成脉冲

1100：**比较脉冲** - tim_oc4refc 或 tim_oc6refc 上升沿在 tim_trgo2 上生成脉冲

1101：**比较脉冲** - tim_oc4refc 上升或 tim_oc6refc 下降沿在 tim_trgo2 上生成脉冲

1110：**比较脉冲** - tim_oc5refc 或 tim_oc6refc 上升沿在 tim_trgo2 上生成脉冲

1111：**比较脉冲** - tim_oc5refc 上升或 tim_oc6refc 下降沿在 tim_trgo2 上生成脉冲

注：在从主定时器接收事件之前，必须使能从机定时器或 ADC 的时钟，并且在从主定时器接收触发器时不能随意更改。

注：若选 tim_ocxrefc 为 tim_trgo2 输出，当 tim_ocref_clr_int 清零信号生效时，tim_ocxrefc 信号将保持低电平，tim_trgo2 也不会有触发输出，直到发生下一更新事件（UEV）（特别注意，当 TIMx_RCR>0 时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。

Bit 19 保留，必须保持为复位值。

Bit 18 **OIS6**：输出空闲状态 6（OC6 输出）

参见 OIS1 位

Bit 17 保留，必须保持为复位值。

Bit 16 **OIS5**：输出空闲状态 5（OC5 输出）

参见 OIS1 位

Bit 15 **OIS4N**：输出空闲状态 4（OC4N 输出）

参见 OIS1 位

Bit 14 **OIS4**：输出空闲状态 4（OC4 输出）

参见 OIS1 位

Bit 13 **OIS3N**：输出空闲状态 3（OC3N 输出）

参见 OIS1 位

Bit 12 **OIS3**：输出空闲状态 3（OC3 输出）

参见 OIS1 位

Bit 11 **OIS2N**：输出空闲状态 2（OC2N 输出）

参见 OIS1 位

Bit 10 **OIS2**：输出空闲状态 2（OC2 输出）

参见 OIS1 位

Bit 9 **OIS1N**：输出空闲状态 1（OC1N 输出）

0：当 MOE=0 时，经过死区时间后 tim_oc1n=0

1：当 MOE=0 时，经过死区时间后 tim_oc1n=1

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别* 1、2 *或*3，*此位即无法修改。*

Bit 8 **OIS1**：输出空闲状态 1（OC1 输出）

0：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）tim_oc1=0

1：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）tim_oc1=1

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别* 1、2 *或*3，*此位即无法修改。*

Bit 7 保留，必须保持为复位值。

Bit 25, 6:4 **MMS[3:0]：**主模式选择

这些位可选择主模式下将要发送到从定时器以实现同步的信息（tim_trgo）。这些位的组合如下：

0000：**复位** – TIMx_EGR 寄存器中的 UG 位用作触发输出（tim_trgo）。如果复位由触发输入生成（从模式控制器配置为复位模式），则 tim_trgo 上的信号相比实际复位会有延迟。

0001：**使能** – 计数器使能信号 CNT_EN 用作触发输出（tim_trgo）。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号可由 CEN 控制位产生。当配置为门控模式时，也可由触发输入产生。当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明）。

0010：**更新** – 选择更新事件作为触发输出（tim_trgo）。例如，主定时器可用作从定时器的预分频器。

0011：**比较脉冲** – 一旦发生比较匹配事件，当 CC1IF 被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(tim_trgo)。

0100：**比较** – tim_oc1refc 信号用作触发输出（tim_trgo）

0101：**比较** – tim_oc2refc 信号用作触发输出（tim_trgo）

0110：**比较** – tim_oc3refc 信号用作触发输出（tim_trgo）

0111：**比较** – tim_oc4refc 信号用作触发输出（tim_trgo）

其他编码保留

注：在从主定时器接收事件之前，必须使能从机定时器或 ADC 的时钟，并且在从主定时器接收触发器时不能随意更改。

注：若选 tim_ocxrefc *为* tim_trgo *输出，当* tim_ocref_clr_int *清零信号生效时，* tim_ocxrefc *信号将保持低电平，* tim_trgo *也不会有触发输出，直到发生下一更新事件（*UEV*）（特别注意，当* TIMx_RCR>0 *时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。*

Bit 3 保留，必须保持为复位值。

Bit 2 **CCUS**：捕获/比较控制更新选择

0：如果捕获/比较控制位（CCPC=1）进行预装载，仅通过将 COMG 位置 1 来对这些位进行更新；

1：如果捕获/比较控制位（CCPC=1）进行预装载，可通过将 COMG 位置 1 或 tim_trgi 的上升沿对这些位进行更新。

注：此位仅对具有互补输出的通道有效。。

Bit 1 保留，必须保持为复位值。

Bit 0 **CCPC**：捕获/比较预装载控制

0：CCxE，CCxNE 和 OCxM 位未进行预装载；

1：CCxE，CCxNE 和 OCxM 位进行了预装载，写入这些位后，仅当发生换向事件（COM）（COMG 位置 1 或在 tim_trgi 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。

注：此位仅对具有互补输出的通道有效。

18.6.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = 1，8）

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SMSPS	SMSPE	Reserved			TS[3]	Reserved			SMS[3]
						rw	rw				rw				rw

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

Bit 31:26 保留，必须保持为复位值。

Bit 25 **SMSPS**：SMS 预加载源（SMS preload source）

此位选择触发 SMS[3:0]位域从预加载到活动的事件来源：

0：由定时器的更新事件触发

1：由索引事件触发

Bit 24 **SMSPE**：SMS 预加载使能（SMS preload enable）

此位选择是否使能 SMS[3:0]位域的预加载：

0：SMS[3:0]位域不进行预加载

1：使能 SMS[3:0]预加载

Bit 23:21 保留，必须保持为复位值。

Bit 20 **TS[3]**：触发器选项- 位 3（Trigger selection - bit 3）

参考 TS[2:0]的描述 - 位 6:4

Bit 19:17 保留，必须保持为复位值。

Bit 16 **SMS[3]**：参考 SMS[2:0]的描述

Bit 15:8 保留，必须保持为复位值。

Bit 7 **MSM**：主/从模式（Master/slave mode）

0：不执行任何操作；

1：当前定时器的触发输入事件（tim_trgi）的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

Bit 6:4 **TS[2:0]**：触发选择（Trigger selection）

此位域和位 TS[3]组合

此位域可选择将要用于同步计数器的触发输入。

0000：内部触发 0（tim_itr0）

0001：内部触发 1（tim_itr1）

0010：内部触发 2（tim_itr2）

0011：内部触发 3（tim_itr3）

0100：保留

0101：保留

0110：保留

0111：保留

1000：内部触发器 4 (tim_itr4)

1001：内部触发器 5 (tim_itr5)

其他：保留

有关定时器 ITRx 含义的详细信息，请参见章节 TIMx 引脚和内部信号中的表格：TIMx 内部触发连接。

*注：这些位只能在未使用的情况下（例如，*SMS=000 *时）进行更改，以避免转换时出现错误的边沿检测。*

Bit 3 保留，必须保持为复位值。

Bit 16, 2:0 **SMS[2:0]**：从模式选择（Slave mode selection）

当选择了外部信号，触发信号（tim_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）

0000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。

0001：保留

0010：保留

0011：保留

0100：复位模式 – 选中的触发输入（tim_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。

0101：门控模式 – 当触发输入（tim_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。

0110：触发模式 – 计数器在触发输入 tim_trgi 的上升沿启动（但不复位），只有计数器的启动是受控的。

0111：外部时钟模式 1– 选中的触发输入（tim_trgi）的上升沿驱动计数器。

1000：组合复位+触发模式 - 选定触发器输入（tim_trgi）的上升沿会重新初始化计数器，生成寄存器的更新并启动计数器。

1001：组合门控+复位模式 - 当触发器输入（tim_trgi）为高电平时，计数器时钟使能。一旦触发器变为低电平，计数器将停止并重置。同时控制计数器的启动和停止。

其他：保留

注：接收 tim_trgo *或* tim_trgo2 *信号的从机外设（定时器、*ADC *等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能随意更改。*

18.6.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x = 1，8）

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CC6IE	Res
														rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC5IE	Reserved							BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
rw								rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:18 保留，必须保持为复位值。

Bit 17 **CC6IE**：捕获/比较 6 中断使能（Capture/Compare 6 interrupt enable）

0：禁止 CC6 中断；

1：使能 CC6 中断。

Bit 16 保留，必须保持为复位值。

Bit 15 **CC5IE**：捕获/比较 5 中断使能（Capture/Compare 5 interrupt enable）

0：禁止 CC5 中断；

1：使能 CC5 中断。

Bit 14:8 保留，必须保持为复位值。

Bit 7 **BIE**：刹车中断使能（Break interrupt enable）

0：禁止刹车中断；

1：使能刹车中断。

Bit 6 **TIE**：触发信号（TGRI）中断使能（Trigger interrupt enable）

0：禁止触发信号（TGRI）中断；

1：使能触发信号（TGRI）中断。

Bit 5 **COMIE**：COM 中断使能（COM interrupt enable）

0：禁止 COM 中断；

1：使能 COM 中断。

Bit 4 **CC4IE**：捕获/比较 4 中断使能（Capture/Compare 4 interrupt enable）

0：禁止 CC4 中断；

1：使能 CC4 中断。

Bit 3 **CC3IE**：捕获/比较 3 中断使能（Capture/Compare 3 interrupt enable）

0：禁止 CC3 中断；

1：使能 CC3 中断。

Bit 2 **CC2IE**：捕获/比较 2 中断使能（Capture/Compare 2 interrupt enable）

0：禁止 CC2 中断；

1：使能 CC2 中断。

Bit 1 **CC1IE**：捕获/比较 1 中断使能（Capture/Compare 1 interrupt enable）

0：禁止 CC1 中断；

1：使能 CC1 中断。

Bit 0 **UIE**：更新中断使能（Update interrupt enable）

0：禁止更新中断；

1：使能更新中断。

18.6.5 TIMx 状态寄存器（TIMx_SR）（x = 1，8）

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														CC6IF	CC5IF
														rc_w0	rc_w0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SBIF	Reserved				B2IF	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
		rc_w0					rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

Bit 31:18 保留，必须保持为复位值。

Bit 17 **CC6IF**：比较 6 中断标志

参考 CC1IF 描述

Bit 16 **CC5IF**：比较 5 中断标志

参考 CC1IF 描述

Bit 15:14 保留，必须保持为复位值。

Bit 13 **SBIF**：系统刹车中断标志（System break interrupt flag）

该标志由硬件在系统刹车输入激活时立即设置。如果系统刹车输入不处于有效状态，则可以通过软件清除该标志。

要重新启动 PWM 操作，必须将此标志重置为 0。

0：未发生刹车事件。

1：在系统刹车输入上检测到有效电平。如果 TIMx_DIER 寄存器中的 BIE=1，则生成中断。

Bit 12:9 保留，必须保持为复位值。

Bit 8 **BI2F**：刹车 2 中断标志（Break 2 interrupt flag）

只要刹车 2 输入变为有效状态，此标志便由硬件置 1。刹车 2 输入无效后可通过软件对其清零。

0：无刹车事件产生；

1：刹车 2 输入上检测到有效电平。如果 TIMx_DIER 寄存器上 BIE=1 则产生一个中断。

Bit 7 **BIF**：刹车中断标志（Break interrupt flag）

只要刹车输入变为有效状态，此标志便由硬件置 1。刹车输入无效后可通过软件对其清零。

0：无刹车事件产生；

1：刹车输入上检测到有效电平。如果 TIMx_DIER 寄存器上 BIE=1 则产生一个中断。

Bit 6 **TIF**：触发中断标志（Trigger interrupt flag）

在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。

0：未发生触发事件。

1：触发中断挂起。

Bit 5 **COMIF**：COM 中断标志（COM interrupt flag）

此标志在发生 COM 事件时（捕获/比较控制位 CCxE、CCxNE 和 OCxM 已更新时）由硬件置 1。但需要通过软件清零。

0：未发生 COM 事件。

1：COM 中断挂起。

Bit 4 **CC4IF**：捕获/比较 4 中断标志（Capture/Compare 4 interrupt flag）

参考 CC1IF 描述。

Bit 3 **CC3IF**：捕获/比较 3 中断标志（Capture/Compare 3 interrupt flag）

参考 CC1IF 描述。

Bit 2 **CC2IF**：捕获/比较 2 中断标志（Capture/Compare 2 interrupt flag）

参考 CC1IF 描述。

Bit 1 **CC1IF**：捕获/比较 1 中断标志（Capture/Compare 1 interrupt flag）

此标志由硬件设置，它可以通过软件（输入捕获或输出比较模式）或读取 TIMx_CCR1 寄存器（仅输入捕获模式）来清除。

0：无比较匹配/无输入捕获发生

1：发生比较匹配或输入捕获

如果通道 CC1 配置为输出：当计数器 TIMx_CNT 的内容与 TIMx_CCR1 寄存器内容匹配时，会设置此标志。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和增减计数模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下有 3 种可能的标志设置选项，请参考 TIMx_CR1 寄存器中的 CMS 位以获取完整描述。

Bit 0 **UIF**：更新中断标志（Update interrupt flag）

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

- TIMx_CR1 寄存器中的 UDIS=0，并且重复计数器值上溢或下溢时（重复计数器= 0 时更新）。
- TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。
- TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（请参见章节：TIMx 从模式控制寄存器（TIMx_SMCR）（x=1，8））。

18.6.6 TIMx 事件产生寄存器（TIMx_EGR）（x = 1，8）

偏移地址：0x014

复位：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							B2G	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
							w	w	w	w	w	w	w	w	w

Bit 15:9 保留，必须保持为复位值。

Bit 8 **B2G**：刹车 2 生成（Break generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：生成刹车 2 事件。MOE 位清零且 BIF 标志置 1。使能后可发生相关中断事件。

Bit 7 **BG**：刹车生成（Break generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：生成刹车事件。MOE 位清零且 BIF 标志置 1。使能后可发生相关中断事件。

Bit 6 **TG**：生成触发信号（Trigger generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件。

Bit 5 **COMG**：捕获/比较控制更新生成（Capture/Compare control update generation）

该位可通过软件置 1，并由硬件自动清零

0：不执行任何操作；

1：CCPC 位置 1 时，可更新 CCxE、CCxNE 和 OCxM 位。

注：此位仅对具有互补输出的通道有效。

Bit 4 **CC4G**：捕获/比较 4 生成（Capture/Compare 4 generation）

参考 CC1G 描述。

Bit 3 **CC3G**：捕获/比较 3 生成（Capture/Compare 3 generation）

参考 CC1G 描述。

Bit 2 **CC2G**：捕获/比较 2 生成（Capture/Compare 2 generation）

参考 CC1G 描述。

Bit 1 **CC1G**：捕获/比较 1 生成（Capture/Compare 1 generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：通道 1 上生成捕获/比较事件：

若通道 CC1 配置为输出：

使能时，CC1IF 标志置 1 并发送相应的中断请求。

Bit 0 **UG**：更新生成（Update generation）

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作；

1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或 DIR=0（递增计数），计数器将清零；如果 DIR=1（递减计数），计数器将使用自动重载值（TIMx_ARR）。

18.6.7 TIMx 捕获/比较模式寄存器 1[复用]（TIMx_CCMR1）（x = 1，8）

偏移地址：0x018

复位值：0x0000 0000

这些通道可用于输出（比较模式）模式。OCxx 用于说明通道配置为输出时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							OC2M[3]	Reserved							OC1M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	Reserved		OC1CE	OC1M[2:0]			OC1PE	OC1FE	Reserved	
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		

输出比较模式

Bit 31:25 保留，必须保持为复位值。

Bit 23:17 保留，必须保持为复位值。

Bit 15 **OC2CE：**输出比较 2 清零使能（Output Compare 2 clear enable）

Bit 24, 14:12 **OC2M[3:0]：**输出比较 2 模式（Output Compare 2 mode）

Bit 11 **OC2PE：**输出比较 2 预装载使能（Output Compare 2 preload enable）

Bit 10 **OC2FE：**输出比较 2 快速使能（Output Compare 2 fast enable）

Bit 9:8 保留，必须保持为复位值。

Bit 7 **OC1CE：**输出比较 1 清零使能（Output Compare 1 clear enable）

0：tim_oc1ref 不受 tim_ocref_clr_int 信号影响；

1：tim_oc1ref 在 tim_ocref_clr_int 信号中被检测到高电平，tim_oc1ref 立即清零。（tim_ocref_clr 输入）

Bit 16, 6:4 **OC1M[3:0]：**输出比较 1 模式（Output Compare 1 mode）

这些位定义提供 tim_oc1 和 tim_oc1n 的输出参考信号 tim_oc1ref 的行为。tim_oc1ref 为高电平有效，而 tim_oc1 和 tim_oc1n 的有效电平则取决于 CC1P 位和 CC1NP 位。

0000：**冻结** - 输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1（TIMx_CCR1）匹配时，tim_oc1ref 信号强制变为高电平。

0010：将通道 1 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1（TIMx_CCR1）匹配时，tim_oc1ref 信号强制变为低电平。

0011：**翻转** - TIMx_CNT=TIMx_CCR1 时，tim_oc1ref 发生翻转。

0100：**强制变为无效电平** - tim_oc1ref 强制变为低电平。

0101：**强制变为有效电平** - tim_oc1ref 强制变为高电平。

0110：**PWM 模式 1**-在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为无效状态（tim_oc1ref=0），否则为有效状态（tim_oc1ref=1）。

0111：**PWM 模式 2** - 在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

1000：**可重触发 OPM 模式 1** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下

次更新时使通道再次处于有效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于无效状态。

1001：**可重触发 OPM 模式 2** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 2 那样执行比较，并在下次更新时使通道再次处于无效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。

1010：保留。

1011：保留。

1100：**组合 PWM 模式 1** - tim_oc1ref 的行为与 PWM 模式 1 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑 OR。

1101：**组合 PWM 模式 2** - tim_oc1ref 的行为与 PWM 模式 2 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑 AND。

1110：**非对称 PWM 模式 1** - tim_oc1ref 的行为与 PWM 模式 1 相同。当计数器向上计数时，tim_oc1refc 输出 tim_oc1ref；当计数器向下计数时，tim_oc1refc 输出 tim_oc2ref。

1111：**非对称 PWM 模式 2** - tim_oc1ref 的行为与 PWM 模式 2 相同。当计数器向上计数时，tim_oc1refc 输出 tim_oc1ref；当计数器向下计数时，tim_oc1refc 输出 tim_oc2ref。

注：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S= “00”（通道配置为输出），这些位即无法修改。

注：在 PWM 模式 1 或 PWM 模式 2 下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

注：此位域将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC1M 有效位才会从预装载位获取新值。

Bit 3 **OC1PE**：输出比较 1 预装载使能（Output Compare 1 preload enable）

0：禁止与 TIMx_CCR1 相关的预装载寄存器。可随时向 TIMx_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIMx_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIMx_CCR1 预装载值在每次生成更新事件时都会装载到活动寄存器中。

注：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S= “00”（通道配置为输出），这些位即无法修改。

Bit 2 **OC1FE**：输出比较 1 快速使能（Output Compare 1 fast enable）

此位减少了触发事件和定时器输出上的转换之间的延迟。它必须在单脉冲模式（在 TIMx_CR1 寄存器中设置 OPM 位）中使用，以便在启动触发后尽快开始输出脉冲。

0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。

1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。

Bit 1:0 保留，必须保持为复位值。

18.6.8 TIMx 捕获/比较模式寄存器 2 [复用]（TIMx_CCMR2）（x = 1，8）

偏移地址：0x01C
复位值：0x0000 0000
这些通道可用于输出（比较模式）模式。OCxx 用于说明通道配置为输出时该位对应的功能。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							OC4M[3]	Reserved							OC3M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	Reserved		OC3CE	OC3M[2:0]			OC3PE	OC3FE	Reserved	
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		

输出比较模式：

Bit 31:25 保留，必须保持为复位值。
Bit 23:17 保留，必须保持为复位值。
Bit 15 **OC4CE：**输出比较 4 清零使能（Output Compare 4 clear enable）
Bit 24, 14:12 **OC4M[2:0]：**输出比较 4 模式（Output Compare 4 mode）
参考 OC3M[3:0]位的描述
Bit 11 **OC4PE：**输出比较 4 预装载使能（Output Compare 4 preload enable）
Bit 10 **OC4FE：**输出比较 4 快速使能（Output Compare 4 fast enable）
Bit 9:8 保留，必须保持为复位值。
Bit 7 **OC3CE：**输出比较 3 清零使能（Output Compare 3 clear enable）
Bit 16, 6:4 **OC3M[3:0]：**输出比较 3 模式（Output Compare 3 mode）
这些位定义提供 tim_oc3 和 tim_oc3n 的输出参考信号 tim_oc3ref 的行为。tim_oc3ref 为高电平有效，而 tim_oc3 和 tim_oc3n 的有效电平则取决于 CC3P 位和 CC3NP 位。
0000：**冻结** - 输出比较寄存器 TIMx_CCR3 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。
0001：将通道 3 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 3（TIMx_CCR3）匹配时，tim_oc3ref 信号强制变为高电平。
0010：将通道 3 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 3（TIMx_CCR3）匹配时，tim_oc3ref 信号强制变为低电平。
0011：**翻转** - TIMx_CNT=TIMx_CCR3 时，tim_oc3ref 发生翻转。
0100：**强制变为无效电平** - tim_oc3ref 强制变为低电平。
0101：**强制变为有效电平** - tim_oc3ref 强制变为高电平。
0110：**PWM 模式 1**-在递增计数模式下，只要 TIMx_CNT<TIMx_CCR3，通道 3 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR3，通道 3 便为无效状态（tim_oc3ref=0），否则为有效状态（tim_oc3ref=1）。
0111：**PWM 模式 2** - 在递增计数模式下，只要 TIMx_CNT<TIMx_CCR3，通道 3 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR3，通道 3 便为有效状态，否则为无效状态。
1000：**可重触发 OPM 模式 1** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。在向下计数模式下，通道在检测到触发事件

（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于无效状态。

1001：**可重触发 OPM 模式 2** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 2 那样执行比较，并在下次更新时使通道再次处于无效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。

1010：**比较脉冲输出**：当 CCR3 匹配事件发生时，根据 TIMxECR 中的 PWPRSC[2:0]和 PW[7:0]位域编程，产生 tim_oc3ref 脉冲。

1011：**方向输出**。tim_oc3ref 信号被 DIR 位的副本覆盖。

1100：**组合 PWM 模式 1** - tim_oc3ref 的行为与 PWM 模式 1 相同。tim_oc3refc 是 tim_oc3ref 和 tim_oc4ref 之间的逻辑 OR。

1101：**组合 PWM 模式 2** - tim_oc3ref 的行为与 PWM 模式 2 相同。tim_oc3refc 是 tim_oc3ref 和 tim_oc4ref 之间的逻辑 AND。

1110：**非对称 PWM 模式 1** - tim_oc3ref 的行为与 PWM 模式 1 相同。当计数器向上计数时，tim_oc3refc 输出 tim_oc3ref；当计数器向下计数时，tim_oc3refc 输出 tim_oc4ref。

1111：**非对称 PWM 模式 2** - tim_oc3ref 的行为与 PWM 模式 2 相同。当计数器向上计数时，tim_oc3refc 输出 tim_oc3ref；当计数器向下计数时，tim_oc3refc 输出 tim_oc4ref。

注：只要编程了 LOCK（TIMx_BDTR 寄存器中的 LOCK 位）级别 3 且 CC1S= “00”（通道配置为输出），这些位即无法修改。

注：在 PWM 模式 1 或 PWM 模式 2 下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，OCREF 电平才会发生更改。

注：此位域将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的 CCPC 位置 1，则仅当生成 COM 事件时，OC3M 有效位才会从预装载位获取新值。

Bit 3 **OC3PE**：输出比较 3 预装载使能（Output Compare 3 preload enable）

Bit 2 **OC3FE**：输出比较 3 快速使能（Output Compare 3 fast enable）

Bit 1:0 保留，必须保持为复位值。

18.6.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = 1，8）

偏移地址：0x020

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										CC6P	CC6E	Reserved		CC5P	CC5E
										rw	rw			rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	CC4NE	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:22 保留，必须保持为复位值。

Bit 21 **CC6P**：输入/捕获 6 输出极性

参考 CC1P 的描述。

Bit 20 **CC6E**：输入/捕获 6 输出使能

参考 CC1E 的描述。

Bit 19:18 保留，必须保持为复位值。

Bit 17 **CC5P**：输入/捕获 5 输出极性

参考 CC1P 的描述。

Bit 16 **CC5E**：输入/捕获 5 输出使能

参考 CC1E 的描述。

Bit 15 **CC4NP**：输入/捕获 4 互补输出极性

参考 CC1NP 的描述。

Bit 14 **CC4NE**：输入/捕获 4 互补输出使能

参考 CC1NE 的描述。

Bit 13 **CC4P**：输入/捕获 4 输出极性

参考 CC1P 的描述。

Bit 12 **CC4E**：输入/捕获 4 输出使能

参考 CC1E 的描述。

Bit 11 **CC3NP**：输入/捕获 3 互补输出极性

参考 CC1NP 的描述。

Bit 10 **CC3NE**：输入/捕获 3 互补输出使能

参考 CC1NE 的描述。

Bit 9 **CC3P**：输入/捕获 3 输出极性

参考 CC1P 的描述。

Bit 8 **CC3E**：输入/捕获 3 输出使能

参考 CC1E 的描述。

Bit 7 **CC2NP**：输入/捕获 2 互补输出极性

参考 CC1NP 的描述。

Bit 6 **CC2NE**：输入/捕获 2 互补输出使能

参考 CC1NE 的描述。

Bit 5 **CC2P**：输入/捕获 2 输出极性

参考 CC1P 的描述。

Bit 4 **CC2E**：输入/捕获 2 输出使能

参考 CC1E 的描述。

Bit 3 **CC1NP**：输入/捕获 1 互补输出极性

CC1 通道配置为输出：

0：tim_oc1n 高电平有效；

1：tim_oc1n 低电平有效。

注：此位将在具有互补输出的通道上进行预装载。如果 TIMx_CR2 寄存器中的CCPC 位置 1，则仅当生成换向事件时，CC1NP 有效位才会从预装载位获取新值。

注：只要编程了LOCK（TIMx_BDTR 寄存器中的LOCK 位）级别 2 或 3 且 CC1S=00（通道配置为输出），此位立即变为不可写状态。

Bit 2 **CC1NE**：输入/捕获 1 互补输出使能

0：关闭—OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1E 位的值。

1：开启—OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1E 位的值。

注：此位将在具有互补输出的通道上进行预装载。如果TIMx_CR2 寄存器中的CCPC 位置 1，则仅当生成换向事件时，CC1NE 有效位才会从预装载位获取新值。

Bit 1 **CC1P**：输入/捕获 1 输出极性

0：OC1 高电平有效

1：OC1 低电平有效

注：只要编程了LOCK（TIMx_BDTR 寄存器中的LOCK 位）级别2 或3，此位立即变为不可写状态。

注：此位将在具有互补输出的通道上进行预装载。如果TIMx_CR2 寄存器中的CCPC 位置 1，则仅当生成换向事件时，CC1P 有效位才会从预装载位获取新值。

Bit 0 **CC1E**：输入/捕获 1 输出使能

0：禁止捕获模式/OC1 未激活（参见以下）

1：使能捕获模式/ OC1 信号输出到对应的输出引脚

当 CC1 通道配置为输出，OC1 的输出电平依赖于 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1NE 位的值，不论 CC1E 位状态。请参考下表获得更多细节

注：此位将在具有互补输出的通道上进行预装载。如果TIMx_CR2 寄存器中的CCPC 位置 1，则仅当生成换向事件时，CC1E 有效位才会从预装载位获取新值。

表 18.13 带刹车功能的互补通道 tim_ocx 和 tim_ocxn 的输出控制位

控制位						输出状态 (1)	
MOE 位	OSSI 位	OSSR 位	OSSD 位	CCxE 位	CCxNE 位	tim_ocx 输出状态	tim_ocxn 输出状态
1	X	X	0	0	0	禁止输出（不由定时器驱动：高阻态） tim_ocx = 0，tim_ocxn = 0	
			1			关闭状态（输出使能为无效状态） tim_ocx = CCxP，tim_ocxn = CCxNP	
		0	X	0	1	禁止输出（不由定时器驱动：高阻态）tim_ocx=0	tim_ocxref + 极性，tim_ocxn = tim_ocxref 异或 CCxNP
		0		1	0	tim_ocxref + 极性，tim_ocx = tim_ocxref 异或 CCxP	禁止输出（不由定时器驱动：高阻态）tim_ocxn = 0
		X		1	1	tim_ocxref + 极性 + 死区	tim_ocxref 互补项（对 tim_ocxref 进行“非”运算）+ 极性 + 死区
		1		0	1	关闭状态（输出使能为无效状态）tim_ocx = CCxP	tim_ocxref + 极性，tim_ocxn = tim_ocxref 异或 CCxNP
		1		1	0	tim_ocxref + 极性，tim_ocx = tim_ocxref 异或 CCxP	关闭状态（输出使能为无效状态）tim_ocxn = CCxNP
0	0	X	X	X	X	禁止输出（不再由定时器驱动）。	
	1		0	0	0	输出状态由 GPIO 控制器定义，可以是高电平、低电平或高阻态。	
			1	0	0	关闭状态（输出使能为无效状态）	
			X	0	1	异步：tim_ocx = CCxP、tim_ocxn = CCxNP（如果触发 tim_brk 或 tim_brk2）。 随后，如果存在时钟：在死区后 tim_ocx = OISx 且 tim_ocxn = OISxN，假定 OISx 和 OISxN 并没有都设置成 tim_ocx 及 tim_ocxn 的有效电平（否则在半桥配置下驱动开关时可能导致短路）。	
				1	0		
				1	1		

1. 如果一个通道的两个输出均未使用（由 GPIO 接管控制控制），则 OISx、OISxN、CCxP 和 CCxNP 位必须保持清零状态。

注： *与互补通道* tim_ocx *和* tim_ocxn *相连的外部* I/O *引脚的状态取决于通道* tim_ocx *和* tim_ocxn *的状态以及* GPIO *寄存器。*

18.6.10 TIMx 计数器（TIMx_CNT）（x = 1，8）

偏移地址：0x024

复位值：0x0000 0000

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

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

Bit 31 **UIFCPY**：UIF 副本（UIF copy）

该位是 TIMx_SR 寄存器中 UIF 位的一个只读副本。如果 TIMx_CR1 上的 UIFREMAP 位被清零，则位 31 被保留并读为 0。

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CNT[15:0]**：计数器的值（Counter value）

在非抖动模式下（DITHEN = 0）

该寄存器（CNT[15:0]）保存的是计数器的值。

在抖动模式下（DITHEN = 1）

该寄存器（CNT[15:0]）只保存计数器的不抖动部分。抖动部分不可用。

18.6.11 TIMx 预分频器（TIMx_PSC）（x = 1，8）

偏移地址：0x028

复位值：0x0000

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

Bit 15:0 **PSC[15:0]**：预分频器的值（Prescaler value）

计数器时钟频率（$f_{tim_cnt_ck}$）等于 $f_{tim_psc_ck}$ /（PSC[15:0] + 1）。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到活动预分频器寄存器的值。

18.6.12 TIMx 自动重装载寄存器（TIMx_ARR）（x = 1，8）

偏移地址：0x02C

复位：0x0000 FFFF

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

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **ARR[19:0]：**自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

有关 ARR 更新和行为的详细信息，请参考章节 TIMx 功能描述：时基单元。

当自动重载值为空时，计数器不工作。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是自动重新加载的值

在抖动模式下（DITHEN = 1）

该寄存器存储的是 ARR[19:4]中的整数部分，而 ARR[3:0]位域存储的是抖动的部分

18.6.13 TIMx 重复计数寄存器（TIMx_RCR）（x = 1，8）

偏移地址：0x030

复位值：0x0000

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

Bit 15:0 **REP[15:0]**：重复计数器的值（Repetition counter value）

使能预装载寄存器时，用户可通过这些位设置比较寄存器的更新频率（即从预装载寄存器向活动寄存器周期性传输数据）；使能更新中断时，也可设置更新中断的生成速率。

与 REP_CNT 相关的减计数器每次计数到 0 时，都将生成一个更新事件并且计数器从 REP 值重新开始计数。由于只有生成重复更新事件 U_RC 时，REP_CNT 才会重载 REP 值，因此在生成下一重复更新事件之前，无论向 TIMx_RCR 寄存器写入何值都无影响。

这意味着 PWM 模式下（REP+1）相当于：

– 边沿对齐模式下的 PWM 周期数；

– 中心对称模式下的 PWM 半周期数；

18.6.14 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x = 1，8）

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR1 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR1[19:0]：**捕获/比较 1 值（Capture/Compare 1 value）

若 CC1 通道配置为输出：

CCR1 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 1 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR1[15:0]中的比较值。CCR1[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR1[19:4]中的整数部分。CCR1[3:0]位域存储的是抖动的部分。

18.6.15 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x = 1，8）

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR2 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR2[19:0]：**捕获/比较 2 值（Capture/Compare 2 value）

若 CC2 通道配置为输出：

CCR2 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 2 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc2 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR2[15:0]中的比较值。CCR2[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR2[19:4]中的整数部分。CCR2[3:0]位域存储的是抖动的部分。

18.6.16 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x = 1，8）

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR3 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR3[19:0]：**捕获/比较 3 值（Capture/Compare 3 value）

若 CC3 通道配置为输出：

CCR3 为要装载到实际捕获/比较 3 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR2 寄存器中的 OC3PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 3 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc3 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR3[15:0]中的比较值。CCR3[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR3[19:4]中的整数部分。CCR3[3:0]位域存储的是抖动的部分。

18.6.17 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x = 1，8）

偏移地址：0x040

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR4 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR4[19:0]：**捕获/比较 4 值（Capture/Compare 4 value）

若 CC4 通道配置为输出：

CCR4 为要装载到实际捕获/比较 4 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR2 寄存器中的 OC4PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 4 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc4 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR4[15:0]中的比较值。CCR4[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR4[19:4]中的整数部分。CCR4[3:0]位域存储的是抖动的部分。

18.6.18 TIMx 刹车和死区寄存器（TIMx_BDTR）（x = 1，8）

偏移地址：0x044

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	OSSD	BK2BID	BKBID	BK2 DSRM	BK DSRM	BK2P	BK2E	BK2F[3:0]				BKF[3:0]			
	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
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

*注：由于可以根据*LOCK *配置锁定位*OSSD、BKBID/BK2BID/BK2P、BK2E、BK2F[3:0]、BKF[3:0]、AOE、BKP、BKE、OSSI、OSSR *和*DTG[7:0]*的写操作，因此必须在第一次对*TIMx_BDTR *寄存器执行写访问时对这些位进行配置。*

Bit 31 保留，必须保持为复位值。

Bit 30 **OSSD**：互补通道都禁用模式下关闭状态选择（**CCxE=CCxNE=0**）

此位在 MOE=1，**CCxE=CCxNE=0** 时，作用于配置为输出的通道使其输出无效状态。

此位在 MOE=0，OSSI=1，**CCxE=CCxNE=0** 时，作用于配置为输出的通道。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=1，8））。

0：处于无效状态时，当 **CCxE=CCxNE=0** 时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于有效状态时：

一旦 MOE=1，**CCxE=CCxNE=0** 时，便将 tim_ocx/tim_ocxn 输出设为无效状态。（输出仍由定时器控制）。

一旦 MOE=0，OSSI=1，**CCxE=CCxNE=0** 时，首先将 tim_ocx/tim_ocxn 输出强制为其无效电平，然后在死区后将其强制为空闲电平。定时器始终控制输出。

*注：只要编程了*LOCK *（*TIMx_BDTR *寄存器中的*LOCK *位）级别*2*，此位即无法修改。*

Bit 29 **BK2BID**：刹车 2 双向（Break2 bidirectional）

请参阅 BKBID 说明

Bit 28 **BKBID**：刹车双向（Break bidirectional）

0：刹车输入 tim_brk 处于输入模式

1：刹车输入 tim_brk 处于双向模式

在双向模式（BKBID 置 1）下，刹车输入同时配置为输入模式和开漏输出模式。任何有效的 break 事件会在刹车输入上输出低电平，以向外部设备指示内部 break 事件。

*注：只要编程了*LOCK *（*TIMx_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 27 **BK2DSRM：**刹车 2 解除（Break2 disarm）

请参阅 BKDSRM 说明

Bit 26 **BKDSRM：**刹车解除保护（Break disarm）

0：刹车输入 tim_brk 处于保护状态

1：刹车输入 tim_brk 解除保护

当没有刹车源处于有效状态时，该位由硬件清零。

软件需置位 BKDSRM 以释放双向输出控制（开漏输出处于高阻态），并轮询直至硬件将其清零，表明故障状态已消失。

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 25 **BK2P：**刹车 2 极性（Break 2 polarity）

0：刹车输入 tim_brk2 为低电平有效

1：刹车输入 tim_brk2 为高电平有效

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 24 **BK2E：**刹车 2 使能（Break 2 enable）

该位可使能完整的刹车 2 保护（参考图片：刹车和刹车 2 电路概述）。

0：禁用 Break2 功能

1：使能 Break2 功能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 23:20 **BK2F[3:0]：**刹车 2 滤波器（Break 2 filter）

此位域可定义 tim_brk2 输入的采样频率和适用于 tim_brk2 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，tim_brk2 异步工作

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/4$，N=8。

0101：$f_{SAMPLING}=f_{DTS}/8$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/16$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/32$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/64$，N=6。

1011：$f_{SAMPLING}=f_{DTS}/64$，N=8。

1100：$f_{SAMPLING}=f_{DTS}/128$，N=6。

1101：$f_{SAMPLING}=f_{DTS}/128$，N=8。

1110：$f_{SAMPLING}=f_{DTS}/256$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/256$，N=8。

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 19:16 **BKF[3:0]：**刹车滤波器（Break filter）

此位域可定义 tim_brk 输入的采样频率和适用于 tim_brk 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，tim_brk 异步工作

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/4$，N=8。

0101：$f_{SAMPLING}=f_{DTS}/8$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/16$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/32$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/64$，N=6。

1011：$f_{SAMPLING}=f_{DTS}/64$，N=8。

1100：$f_{SAMPLING}=f_{DTS}/128$，N=6。

1101：$f_{SAMPLING}=f_{DTS}/128$，N=8。

1110：$f_{SAMPLING}=f_{DTS}/256$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/256$，N=8。

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 15 **MOE**：主输出使能（Main output enable）

只要刹车输入（tim_brk 或 tim_brk2）为有效状态，此位便由硬件异步清零。此位由软件置 1，也可根据 AOE 位状态自动置 1。此位仅对配置为输出的通道有效。

0：响应刹车事件或向 MOE 写入 0 时：tim_ocx 和 tim_ocxn 输出被禁止或被强制为空闲状态，具体取决于 OSSI 位。

1：如果 tim_ocx 和 tim_ocxn 输出的相应使能位（TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位）均置 1，则使能 tim_ocx 和 tim_ocxn 输出。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=1，8））。

Bit 14 **AOE**：自动输出使能（Automatic output enable）

0：MOE 只能由软件置 1

1：MOE 可由软件置 1，也可在发生下一更新事件时自动置 1（如果刹车输入 tim_brk 和 tim_brk2 均无效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 13 **BKP**：刹车极性（Break polarity）

0：刹车输入 tim_brk 为低电平有效

1：刹车输入 tim_brk 为高电平有效

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 12 **BKE**：刹车使能（Break enable）

该位可使能完整的刹车保护（包括所有连接到 tim_sys_brk 和 BKIN 的源，参考图片：刹车和刹车 2 电路概述）。

0：禁用 Break 功能

1：使能 Break 功能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 11 **OSSR**：运行模式下的关闭状态选择（Off-state selection for Run mode）

此位在 MOE=1 时作用于配置为输出模式且具有互补输出的通道。如果定时器中没有互补输出，则不存在 OSSR。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=1，8））。

0：处于无效状态时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，便使能 tim_ocx/tim_ocxn 输出并将其设为无效电平（输出仍由定时器控制）。

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*2，*此位即无法修改。*

Bit 10 **OSSI**：空闲模式下的关闭状态选择（Off-state selection for Idle mode）

当由于刹车事件或软件写操作而使 MOE=0 时，此位作用于配置为输出的通道。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=1，8））。

0：处于无效状态时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于无效状态时，首先将 tim_ocx/tim_ocxn 输出强制为其无效电平，然后在死区后将其强制为空闲电平。定时器始终控制输出。

*注：只要编程了*LOCK *（*TIMx_BDTR *寄存器中的*LOCK *位）级别*2*，此位即无法修改。*

Bit 9:8 **LOCK[1:0]**：锁定设置（Lock configuration）

这些位用于针对软件错误提供写保护。

00：关闭锁定 – 不对任何位提供写保护。

01：锁定级别 1，此时无法对 TIMx_BDTR 寄存器中的 DTG 位、TIMx_CR2 寄存器中的 OISx 和 OISxN 位以及 TIMx_BDTR 寄存器中的 BKBID/BK2BID/BKE/BKP/AOE 位执行写操作。

10：锁定级别 2，此时无法对锁定级别 1 中适用的各位、CC 极性位（TIMx_CCER 寄存器中的 CCxP/CCxNP 位，只要通过 CCxS 位将相关通道配置为输出）以及 OSSD、OSSR 和 OSSI 位执行写操作。

11：锁定级别 3，此时无法对锁定级别 2 中适用的各位、CC 控制位（TIMx_CCMRx 寄存器中的 OCxM 和 OCxPE 位，只要通过 CCxS 位将相关通道配置为输出）执行写操作。

*注：复位后只能对*LOCK *位执行一次写操作。对*TIMx_BDTR *寄存器执行写操作后其中的内容将冻结，直到下一次复位。*

Bit 7:0 **DTG[7:0]**：配置死区发生器（Dead-time generator setup）

此位域定义插入到互补输出之间的死区持续时间。DT 与该持续时间相对应。

DTG[7:5]=0xx => DT= DTG[7:0] × T_{dtg}，其中 T_{dtg} = T_{DTS}；

DTG[7:5]=10x => DT=（64+DTG[5:0]）× T_{dtg}，其中 T_{dtg} = 2 × T_{DTS}；

DTG[7:5]=110 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 8 × T_{DTS}；

DTG[7:5]=111 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 16 × T_{DTS}；

例：若 T_{DTS} = 125ns（8MHZ），可能的死区时间为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

*注：只要编程了*LOCK *（*TIMx_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3*，此位域即无法修改。*

18.6.19 TIMx 捕获/比较寄存器 5（TIMx_CCR5）（x = 1，8）

偏移地址：0x048

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GC5C3	GC5C2	GC5C1	Reserved									CCR5 [19:16]			
rw	rw	rw										rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR5 [15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **GC5C3**：通道 5 和通道 3 组（Group channel 5 and channel 3）

通道 3 输出失真：

0：tim_oc5ref 对 tim_oc3refc 无影响

1：tim_oc3refc 是 tim_oc3ref 和 tim_oc5ref 的逻辑 AND 操作

此位要么立即生效，要么被预加载并在更新事件后被考虑（如果 TIMx_CCMR2 中选择了预加载功能）。

注：也可以在组合 PWM 信号上应用这种失真。

Bit 30 **GC5C2**：通道 5 和通道 2 组（Group channel 5 and channel 2）

通道 2 输出失真：

0：tim_oc5ref 对 tim_oc2refc 无影响

1：tim_oc2refc 是 tim_oc2ref 和 tim_oc5ref 的逻辑 AND 操作

此位要么立即生效，要么被预加载并在更新事件后被考虑（如果 TIMx_CCMR1 中选择了预加载功能）。

注：也可以在组合 PWM 信号上应用这种失真。

Bit 29 **GC5C1**：通道 5 和通道 1 组（Group channel 5 and channel 1）

通道 1 输出失真：

0：tim_oc5ref 对 tim_oc1refc 无影响

1：tim_oc1refc 是 tim_oc1ref 和 tim_oc5ref 的逻辑 AND 操作

此位要么立即生效，要么被预加载并在更新事件后被考虑（如果 TIMx_CCMR1 中选择了预加载功能）。

注：也可以在组合 PWM 信号上应用这种失真。

Bit 28:20 保留，必须保持为复位值。

Bit 19:0 **CCR5[15:0]**：捕获/比较 5 值（Capture/Compare 5 value）

CCR5 为要装载到实际捕获/比较 5 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR3 寄存器中的 OC5PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 5 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc5 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR5[15:0]中的比较值。CCR5[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR5[19:4]中的整数部分。CCR5[3:0]位域存储的是抖动的部分。

18.6.20 TIMx 捕获/比较寄存器 6（TIMx_CCR6）（x = 1，8）

偏移地址：0x04C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR6 [19:16]			
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR6 [15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR6[15:0]：**捕获/比较 6 值（Capture/Compare 6 value）

CCR6 为要装载到实际捕获/比较 6 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR3 寄存器中的 OC6PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 6 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc6 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR6[15:0]中的比较值。CCR6[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR6[19:4]中的整数部分。CCR6[3:0]位域存储的是抖动的部分。

18.6.21 TIMx 捕获/比较模式寄存器 3（TIMx_CCMR3）（x = 1，8）

偏移地址：0x050

复位值：0x0000 0000

参看以上 CCMR1 寄存器的描述，通道 5 和 6 只能配置为输出。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							OC6M[3]	Reserved							OC5M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC6CE	OC6M[2:0]			OC6PE	OC6FE	Reserved		OC5CE	OC5M[2:0]			OC5PE	OC5FE	Reserved	
rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw		

Bit 31:25 保留，必须保持为复位值。

Bit 23:17 保留，必须保持为复位值。

Bit 15 **OC6CE：**输出比较 6 清零使能（Output Compare 6 clear enable）

Bit 24, 14:12 **OC6M[2:0]：**输出比较 6 模式（Output Compare 6 mode）

Bit 11 **OC6PE：**输出比较 6 预装载使能（Output Compare 6 preload enable）

Bit 10 **OC6FE：**输出比较 6 快速使能（Output Compare 6 fast enable）

Bit 9:8 保留，必须保持为复位值。

Bit 7 **OC5CE：**输出比较 5 清零使能（Output Compare 5 clear enable）

Bit 16, 6:4 **OC5M[2:0]：**输出比较 5 模式（Output Compare5 mode）

Bit 3 **OC5PE：**输出比较 5 预装载使能（Output Compare 5 preload enable）

Bit 2 **OC5FE：**输出比较 5 快速使能（Output Compare 5 fast enable）

Bit 1:0 保留，必须保持为复位值。

18.6.22 TIMx 定时器死区寄存器 2（TIMx_DTR2）（x = 1，8）

偏移地址：0x054

复位值：0x0000 0000

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

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

Bit 31:18 保留，必须保持为复位值。

Bit 17 **DTPE：**死区时间预加载使能（Deadtime preload enable）

0：死区时间值未预加载

1：使能死区时间值预加载

*注：编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 16 **DTAE：**死区时间非对称使能（Deadtime asymmetric enable）

0：上升沿和下降沿的死区时间是相同的，并且使用 DTG[7:0]寄存器定义

1：上升沿的死区时间使用 DTG[7:0]寄存器定义，下降沿的死区时间使用 DTGF[7:0]位定义。

*注：编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 15:8 保留，必须保持为复位值。

Bit 7:0 **DTGF[7:0]：**死区下降沿发生器（Dead-time falling edge generator setup）

这个位域定义了在下降沿时，互补输出之间插入死区时间的长度。

DTG[7:5]=0xx => DT= DTG[7:0] × T_{dtg}，其中 T_{dtg} = T_{DTS}；

DTG[7:5]=10x => DT=（64+DTG[5:0]）× T_{dtg}，其中 T_{dtg} = 2 × T_{DTS}；

DTG[7:5]=110 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 8 × T_{DTS}；

DTG[7:5]=111 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 16 × T_{DTS}；

例：若 T_{DTS} = 125ns（8MHZ），可能的死区时间为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

*注：编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

18.6.23 TIMx 定时器编码器控制寄存器（TIMx_ECR）（x = 1，8）

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					PWPRSC[2:0]			PW[7:0]							
					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
Reserved															

Bit 31:27 保留，必须保持为复位值。

Bit 26:24 **PWPRSC[2:0]**：脉冲宽度预分频器（Pulse width prescaler）

这个位域为脉冲发生器设置时钟预分频器，具体如下：

$t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$

Bit 23:16 **PW[7:0]**：脉冲宽度（Pulse width）

这个位域定义脉冲的持续时间，具体如下：

$t_{PW} = PW[7:0] \times t_{PWG}$

Bit 15:0 保留，必须保持为复位值。

18.6.24 TIMx 复用功能选择寄存器 1（TIMx_AF1）（x = 1，8）

偏移地址：0x060

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	Reserved				BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
		rw	rw	rw	rw	rw					rw	rw	rw	rw	rw

Bit 31:14 保留，必须保持为复位值。

Bit 13 **BKCMP4P**：tim_brk_cmp4 输入极性（tim_brk_cmp4 input polarity）

此位选择 tim_brk_cmp4 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp4 输入极性未反转（如果 BKP=0，则低电平有效，如果 BKP=1，则高电平有效）

1：tim_brk_cmp4 输入极性反转（如果 BKP=0，则高电平有效，如果 BKP=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 12 **BKCMP3P**：tim_brk_cmp3 输入极性（tim_brk_cmp3 input polarity）

此位选择 tim_brk_cmp3 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp3 输入极性未反转（如果 BKP=0，则低电平有效，如果 BKP=1，则高电平有效）

1：tim_brk_cmp3 输入极性反转（如果 BKP=0，则高电平有效，如果 BKP=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 11 **BKCMP2P**：tim_brk_cmp2 输入极性（tim_brk_cmp2 input polarity）

此位选择 tim_brk_cmp2 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp2 输入极性未反转（如果 BKP=0，则低电平有效，如果 BKP=1，则高电平有效）

1：tim_brk_cmp2 输入极性反转（如果 BKP=0，则高电平有效，如果 BKP=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 10 **BKCMP1P**：tim_brk_cmp1 输入极性（tim_brk_cmp1 input polarity）

此位选择 tim_brk_cmp1 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp1 输入极性未反转（如果 BKP=0，则低电平有效，如果 BKP=1，则高电平有效）

1：tim_brk_cmp1 输入极性反转（如果 BKP=0，则高电平有效，如果 BKP=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 9 **BKINP**：TIMx_BKIN 输入极性（TIMx_BKIN input polarity）

该位选择 TIMx_BKIN 备用功能输入敏感度。必须对其和 BKP 极性位进行编程。

0：TIMx_BKIN 输入极性未反转（如果 BKP=0，则低电平有效，如果 BKP=1，则高电平有效）

1：TIMx_BKIN 输入极性反转（如果 BKP=0，则高电平有效，如果 BKP=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 8:5 保留，必须保持为复位值。

Bit 4 **BKCMP4E**：tim_brk_cmp4 使能（tim_brk_cmp4 enable）

该位用于使能 tim_brk 输入的 tim_brk_cmp4。tim_brk_cmp4 输出与其他 tim_brk 源进行“或”运算。

0：tim_brk_cmp4 输入禁用

1：tim_brk_cmp4 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 3 **BKCMP3E**：tim_brk_cmp3 使能（tim_brk_cmp3 enable）

该位用于使能 tim_brk 输入的 tim_brk_cmp3。tim_brk_cmp3 输出与其他 tim_brk 源进行“或”运算。

0：tim_brk_cmp3 输入禁用

1：tim_brk_cmp3 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 2 **BKCMP2E**：tim_brk_cmp2 使能（tim_brk_cmp2 enable）

该位用于使能 tim_brk 输入的 tim_brk_cmp2。tim_brk_cmp2 输出与其他 tim_brk 源进行“或”运算。

0：tim_brk_cmp2 输入禁用

1：tim_brk_cmp2 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 1 **BKCMP1E**：tim_brk_cmp1 使能（tim_brk_cmp1 enable）

该位用于使能 tim_brk 输入的 tim_brk_cmp1。tim_brk_cmp1 输出与其他 tim_brk 源进行“或”运算。

0：tim_brk_cmp1 输入禁用

1：tim_brk_cmp1 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 0 **BKINE**：TIMx_BKIN 输入使能（TIMx_BKIN input enable）

该位用于使能 TIMx_BKIN 复用功能输入作为定时器的 tim_brk 输入。TIMx_BKIN 输入与其他 tim_brk 源进行“或”运算。

0：TIMx_BKIN 输入禁用

1：TIMx_BKIN 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注意：有关产品特定实现的信息，请参考章节：*TIM1/TIM8 *引脚和内部信号。*

18.6.25 TIMx 复用功能选择寄存器 2（TIMx_AF2）（x = 1，8）

偏移地址：0x064

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BK2 CMP4P	BK2 CMP3P	BK2 CMP2P	BK2 CMP1P	BK2INP	Reserved				BK2 CMP4E	BK2 CMP3E	BK2 CMP2E	BK2 CMP1E	BK2INE
		rw	rw	rw	rw	rw					rw	rw	rw	rw	rw

Bit 31:19 保留，必须保持为复位值。

Bit 17:16 **OCRSEL[1:0]**：ocref_clr 源选择（ocref_clr source selection）

这些位选择 ocref_clr 输入源。

00：tim_ocref_clr0

01：tim_ocref_clr1

10：tim_ocref_clr2

11：tim_ocref_clr3

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 15:14 保留，必须保持为复位值。

Bit 13 **BK2CMP4P**：tim_brk2_cmp4 输入极性（tim_brk2_cmp4 input polarity）

此位选择 tim_brk2_cmp4 输入敏感度。它必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp4 输入极性未反转（如果 BK2P=0，则低电平有效，如果 BK2P=1，则高电平有效）

1：tim_brk2_cmp4 输入极性反转（如果 BK2P=0，则高电平有效，如果 BK2P=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 12 **BK2CMP3P**：tim_brk2_cmp3 输入极性（tim_brk2_cmp3 input polarity）

此位选择 tim_brk2_cmp3 输入敏感度。它必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp3 输入极性未反转（如果 BK2P=0，则低电平有效，如果 BK2P=1，则高电平有效）

1：tim_brk2_cmp3 输入极性反转（如果 BK2P=0，则高电平有效，如果 BK2P=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 11 **BK2CMP2P**：tim_brk2_cmp2 输入极性（tim_brk2_cmp2 input polarity）

此位选择 tim_brk2_cmp2 输入敏感度。它必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp2 输入极性未反转（如果 BK2P=0，则低电平有效，如果 BK2P=1，则高电平有效）

1：tim_brk2_cmp2 输入极性反转（如果 BK2P=0，则高电平有效，如果 BK2P=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 10 **BK2CMP1P**：tim_brk2_cmp1 输入极性（tim_brk2_cmp1 input polarity）

此位选择 tim_brk2_cmp1 输入敏感度。它必须与 BK2P 极性位一起编程。

0：tim_brk2_cmp1 输入极性未反转（如果 BK2P=0，则低电平有效，如果 BK2P=1，则高电平有效）

1：tim_brk2_cmp1 输入极性反转（如果 BK2P=0，则高电平有效，如果 BK2P=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 9 **BK2INP**：TIMx_BKIN2 输入极性（TIMx_BKIN2 input polarity）

该位选择 TIMx_BKIN2 复用功能输入敏感度。它必须与 BK2P 极性位一起编程。

0：TIMx_BKIN2 输入极性未反转（如果 BK2P=0，则低电平有效，如果 BK2P=1，则高电平有效）

1：TIMx_BKIN2 输入极性反转（如果 BK2P=0，则高电平有效，如果 BK2P=1，则低电平有效）

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 8:5 保留，必须保持为复位值。

Bit 4 **BK2CMP4E**：tim_brk2_cmp4 使能（tim_brk2_cmp4 enable）

该位用于使能 tim_brk2 输入的 tim_brk_cmp4。tim_brk2_cmp4 输出与其他 tim_brk2 源进行“或”运算。

0：tim_brk2_cmp4 输入禁用

1：tim_brk2_cmp4 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 3 **BK2CMP3E**：tim_brk2_cmp3 使能（tim_brk2_cmp3 enable）

该位用于使能 tim_brk2 输入的 tim_brk_cmp3。tim_brk2_cmp3 输出与其他 tim_brk2 源进行“或”运算。

0：tim_brk2_cmp3 输入禁用

1：tim_brk2_cmp3 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 2 **BK2CMP2E**：tim_brk2_cmp2 使能（tim_brk2_cmp2 enable）

该位用于使能 tim_brk2 输入的 tim_brk_cmp2。tim_brk2_cmp2 输出与其他 tim_brk2 源进行“或”运算。

0：tim_brk2_cmp2 输入禁用

1：tim_brk2_cmp2 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 1 **BK2CMP1E**：tim_brk2_cmp1 使能（tim_brk2_cmp1 enable）

该位用于使能 tim_brk2 输入的 tim_brk_cmp1。tim_brk2_cmp1 输出与其他 tim_brk2 源进行“或”运算。

0：tim_brk2_cmp1 输入禁用

1：tim_brk2_cmp1 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 0 **BK2INE**：TIMx_BKIN2 输入使能（TIMx_BKIN2 input enable）

该位用于使能 TIMx_BKIN2 复用功能输入作为定时器的 tim_brk2 输入。TIMx_BKIN2 输入与其他 tim_brk2 源进行“或”运算。

0：TIMx_BKIN2 输入禁用

1：TIMx_BKIN2 输入使能

*注：只要编程了*LOCK（TIMx_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注意：有关产品特定实现的信息，请参考章节：*TIM1/TIM8 *引脚和内部信号。*

18.6.26 TIMx 捕获/比较寄存器 1N（TIMx_CCR1N）（x = 1，8）

偏移地址：0x068

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1N[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **EN**：若 CC1 通道配置为输出：

CCR1N_EN=1：CNT 向下数比较的寄存器为 CCR1N

CCR1N_EN=0：CNT 向下数比较的寄存器为 CCR1

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CCR1N[15:0]**：捕获/比较通道 1N 的值（Capture/Compare 1N value）

若 CC1 通道配置为输出：

CCR1N 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。

如果在 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。CCR1N_EN 必须为 1，寄存器才有效果。

18.6.27 TIMx 捕获/比较寄存器 2N（TIMx_CCR2N）（x = 1，8）

偏移地址：0x06C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2N[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **EN**：若 CC2 通道配置为输出：

CCR2N_EN=1：CNT 向下数比较的寄存器为 CCR2N

CCR2N_EN=0：CNT 向下数比较的寄存器为 CCR2

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CCR2N[15:0]**：捕获/比较通道 2N 的值（Capture/Compare 2N value）

若 CC2 通道配置为输出：

CCR2N 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。

如果在 TIMx_CCMR1 寄存器（OC2PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。CCR2N_EN 必须为 1，寄存器才有效果。

18.6.28 TIMx 捕获/比较寄存器 3N（TIMx_CCR3N）（x = 1，8）

偏移地址：0x070

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3N[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **EN**：若 CC3 通道配置为输出：

CCR3N_EN=1：CNT 向下数比较的寄存器为 CCR3N

CCR3N_EN=0：CNT 向下数比较的寄存器为 CCR3

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CCR3N[15:0]**：捕获/比较通道 3N 的值（Capture/Compare 3N value）

若 CC3 通道配置为输出：

CCR3N 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。

如果在 TIMx_CCMR2 寄存器（OC3PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。CCR3N_EN 必须为 1，寄存器才有效果。

18.6.29 TIMx 捕获/比较寄存器 4N（TIMx_CCR4N）（x = 1，8）

偏移地址：0x074

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4N[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31 **EN**：若 CC4 通道配置为输出：

CCR4N_EN=1：CNT 向下数比较的寄存器为 CCR4N

CCR4N_EN=0：CNT 向下数比较的寄存器为 CCR4

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CCR4N[15:0]**：捕获/比较通道 4N 的值（Capture/Compare 4N value）

若 CC4 通道配置为输出：

CCR4N 包含了装入当前捕获/比较 4 寄存器的值（预装载值）。

如果在 TIMx_CCMR2 寄存器（OC4PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。CCR4N_EN 必须为 1，寄存器才有效果。

18.6.30 TIMx 刹车输入控制选择（TIMx_BICS）（x = 1，8）

偏移地址：0x078

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CH6	CH5	CH4	CH3	CH2	CH1	BICE
									rw	rw	rw	rw	rw	rw	rw

Bit 15:7 保留，必须保持为复位值。

Bit 6 **CH6**：刹车输入控制选择 CH6（Break Iutput Control Selection CH6）

当 BICE=1 时：

0：CH6 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH6 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 5 **CH5**：刹车输入控制选择 CH5（Break Iutput Control Selection CH5）

当 BICE=1 时：

0：CH5 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH5 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 4 **CH4**：刹车输入控制选择 CH4（Break Iutput Control Selection CH4）

当 BICE=1 时：

0：CH4 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH4 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 3 **CH3**：刹车输入控制选择 CH3（Break Iutput Control Selection CH3）

当 BICE=1 时：

0：CH3 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH3 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 2 **CH2**：刹车输入控制选择 CH2（Break Iutput Control Selection CH2）

当 BICE=1 时：

0：CH2 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH2 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 1 **CH1**：刹车输入控制选择 CH1（Break Iutput Control Selection CH1）

当 BICE=1 时：

0：CH1 刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

1：CH1 刹车无效，设置 MOE=0 无效，通道维持原有输出

Bit 0 **BICE**：刹车输入控制使能（Break Iutput Control Enable）

当刹车输入或设置 MOE=0 时，此位控制刹车或设置 MOE=0 是否对 CHx/CHyN(x=1,2,3,4,5,6)(y=1,2,3,4)有效：

BICE=0：刹车有效或设置 MOE=0 有效，通道根据极性配置输出对应电平

BICE=1：根据 BICS_CHx 位可选某些通道忽略刹车输入或忽略设置 MOE=0 操作维持原有输出

19 通用定时器（TIM2/TIM3）

19.1 TIM2/TIM3 简介

通用定时器包含一个 16 位或 28 位自动重载计数器，该计数器由可编程预分频器驱动。

它们可用于多种用途，包括测量输入信号的脉冲宽度（输入捕获）或生成输出波形（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，可将脉冲宽度和波形周期从几微秒调制到几毫秒。这些定时器彼此完全独立，不共享任何资源，它们可以同步操作。

19.2 TIM2/TIM3 主要功能

通用 TIMx 定时器功能包括：

- 16 位（TIM3）或 28 位（TIM2）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
- 2 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边沿或中间对齐模式）
  - 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请求：
  - 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
- 触发输入作为外部时钟或者按周期的电流管理

19.3 TIM2/TIM3 实现

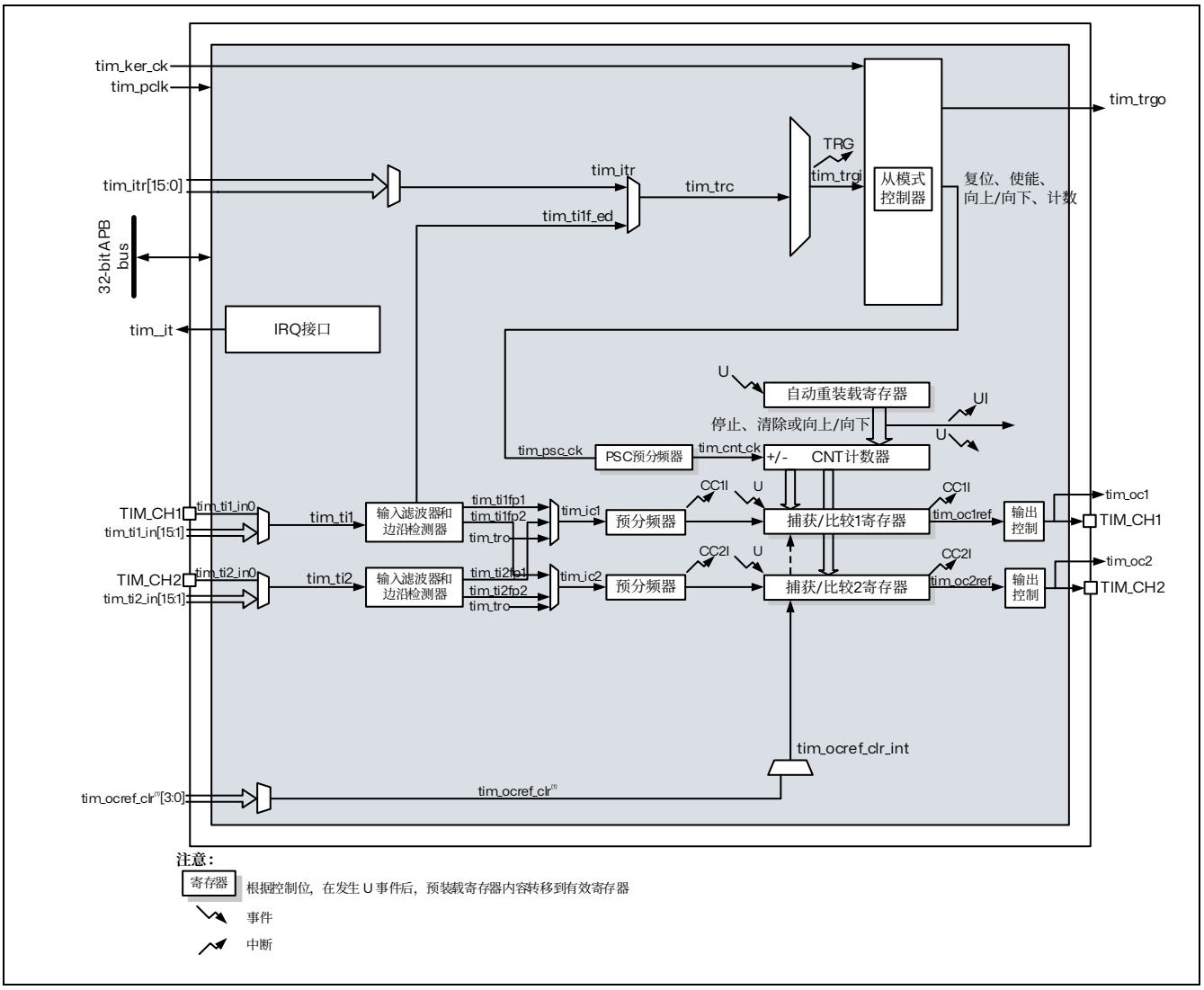
表 19.1 RX32H710 系列通用定时器

定时器实例	TIM2	TIM3
分辨率	28 位	16 位
OCREF 清除选择源	是 tim_ocref_clr[3:0]	是 tim_ocref_clr[3:0]

19.4 TIM2/TIM3 功能描述

19.4.1 框图

图 19.1 通用定时器框图



注 1. 此功能具体请参考章节：TIM2/TIM3 实现

19.4.2 TIM2/TIM3 引脚和内部信号

本节中的下两张表格总结了 TIM 的输入和输出。

表 19.2 TIM 输入/输出引脚

引脚名称	信号类型	描述
TIM_CH1 TIM_CH2	输入/输出	定时器多功能通道。 每个通道用于捕获、比较或 PWM。 TIM_CH1 和 TIM_CH2 也可用作外部时钟（tim_ker_ck 时钟的 1/4 以下）和外部触发器输入。

表 19.3 TIM 内部输入/输出信号

内部信号名称	信号类型	描述
tim_ti1_in[15:0] tim_ti2_in[15:0]	输入	内部定时器输入总线。 tim_ti1_in[15:0]和 tim_ti2_in[15:0]输入可被用作捕获或作为外部时钟（tim_ker_ck 时钟的 1/4 以下）。
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或作为输入时钟（tim_ker_ck 时钟的 1/4 以下）。
tim_trgo	输出	内部触发器输出。此触发器可以触发其他片上外设。
tim_ocref_clr[3:0]	输入	定时器 tim_ocref_clr 输入总线。这些输入可以用来清除 tim_ocxref 信号，通常用于硬件的逐周期脉冲宽度控制。
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟。
tim_it	输出	全局定时器中断，收集捕获/比较、更新和中断触发器请求。

如下两张表列出了连接到 tim_ti[2:1]输入多路复用器的源。

表 19.4 连接到 tim_ti1 的输入多路复用器

tim_ti1 输入	源	
	TIM2	TIM3
tim_ti1_in0	TIM2_CH1	TIM3_CH1
tim_ti1_in1	comp1_out	comp1_out
tim_ti1_in2	comp2_out	comp2_out
tim_ti1_in3	comp3_out	comp3_out
tim_ti1_in4	comp4_out	comp4_out
tim_ti1_in5	LSI	LSI

表 19.5 连接到 tim_ti2 的输入多路复用器

tim_ti2 输入	源	
	TIM2	TIM3
tim_ti2_in0	TIM2_CH2	TIM3_CH2
tim_ti2_in1	comp1_out	comp1_out
tim_ti2_in2	comp2_out	comp2_out
tim_ti2_in3	comp3_out	comp3_out
tim_ti2_in4	comp4_out	comp4_out
tim_ti2_in5	LSI	LSI

下表列出连接到 tim_itr 输入多路复用器的内部源

表 19.6 TIMx 内部触发连接

tim_itrx 输入	TIM2	TIM3
tim_itr0	tim3_trgo	tim2_trgo
tim_itr1	tim8_trgo	tim8_trgo
tim_itr2	tim15_trgo	tim15_trgo
tim_itr3	tim1_trgo	tim1_trgo
tim_itr4	timA_trgo	timA_trgo
tim_itr5	timB_trgo	timB_trgo
tim_itr[15:6]	保留	

下表列出连接到 tim_ocref_clr 输入多路复用器的内部源

表 19.7 连接到 ocref_clr 的输入多路复用器

定时器信号 tim_ocref_clr	定时器 tim_ocref_clr 信号分配	
	TIM2	TIM3
tim_ocref_clr0	comp1_out	comp1_out
tim_ocref_clr1	comp2_out	comp2_out
tim_ocref_clr2	comp3_out	comp3_out
tim_ocref_clr3	comp4_out	comp4_out

19.4.3 时基单元

可编程定时器的主要模块由一个 16 位/28 位计数器及其相关的自动重装寄存器组成。计数器可递增计数、递减计数或同时进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包含:

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件（UEV）时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE）。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生 进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位（CEN）置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

请注意，真正的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

预分频器描述

预分频器可对计数器时钟频率进行分频，分频系数介于 1 到 65536 之间。该预分频器基于 16 位/28 位寄存器（TIMx_PSC 寄存器）所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用.

下两张图给出了在预分频器运行时，更改计数器参数的例子。

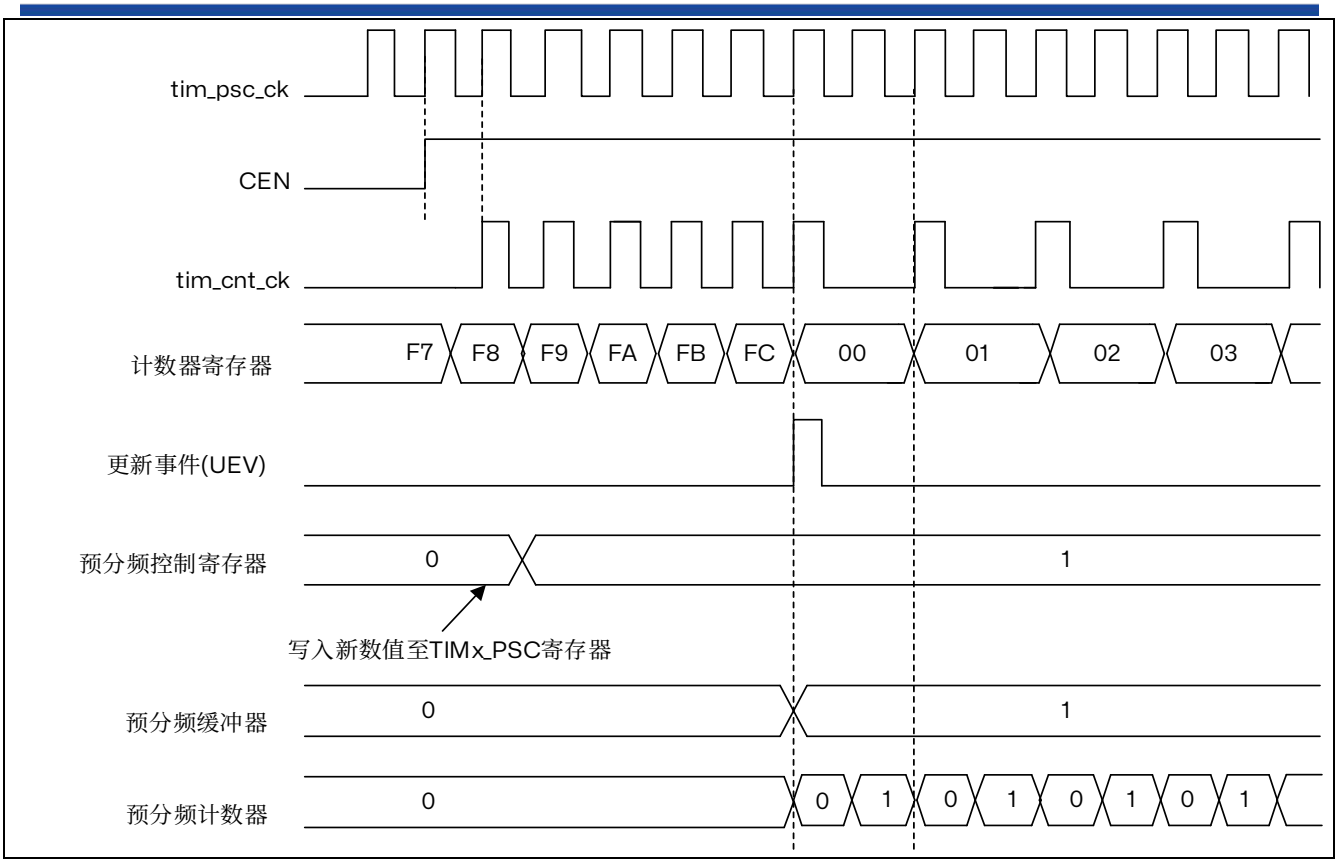


图 19.2 当预分频器的参数从 1 变到 2 时，计数器的时序图

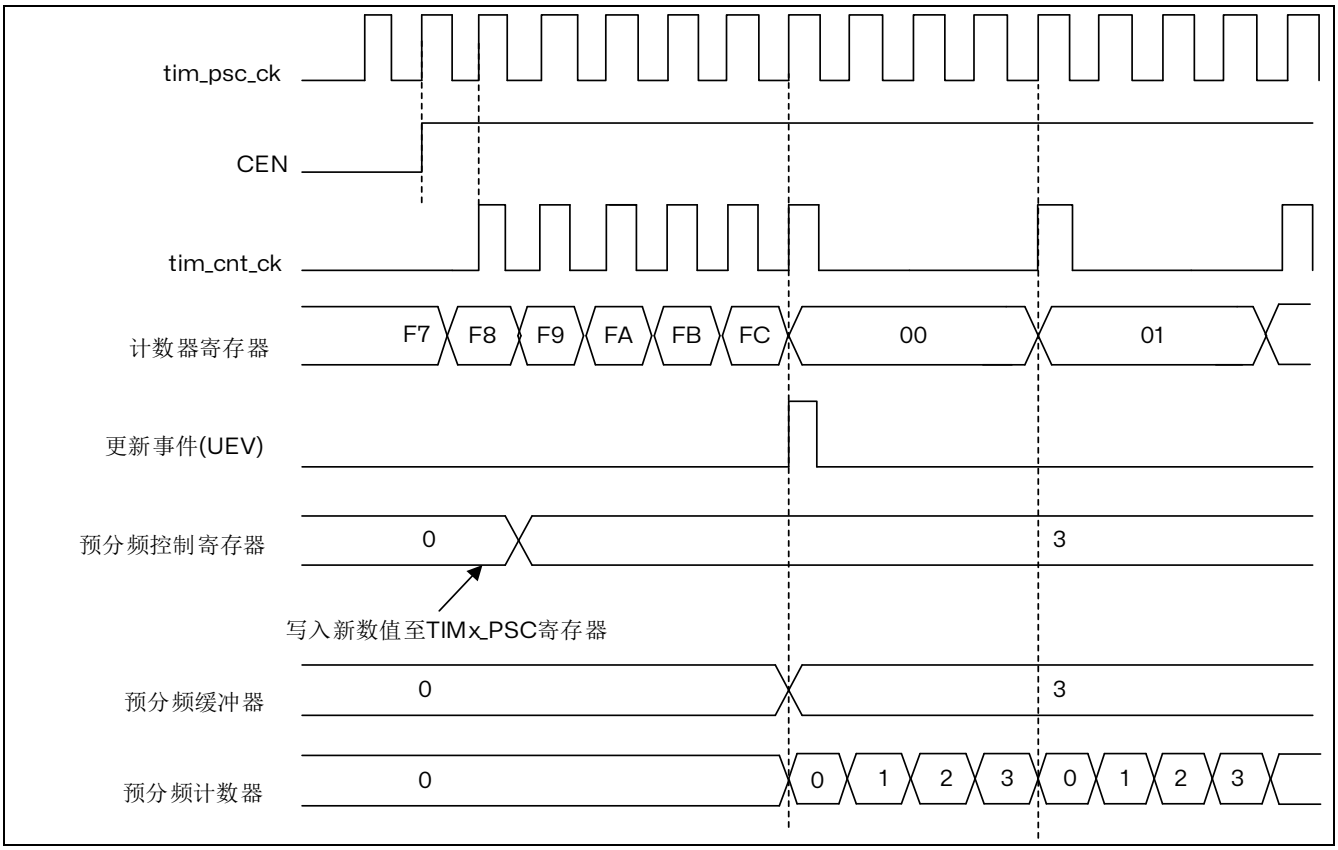


图 19.3 当预分频器的参数从 1 变到 4 时，计数器的时序图

19.4.4 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载影子寄存器将以预装载值进行更新

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

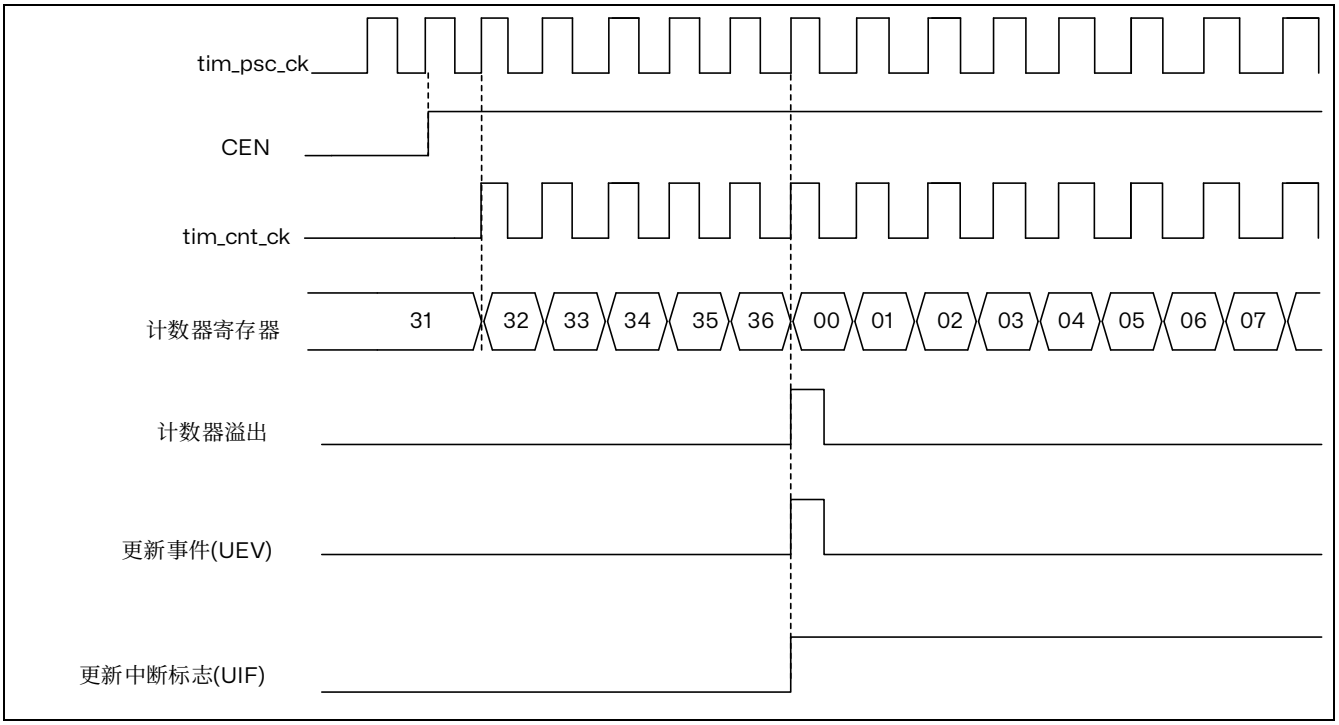


图 19.4 计数器时序图，内部时钟分频系数为 1

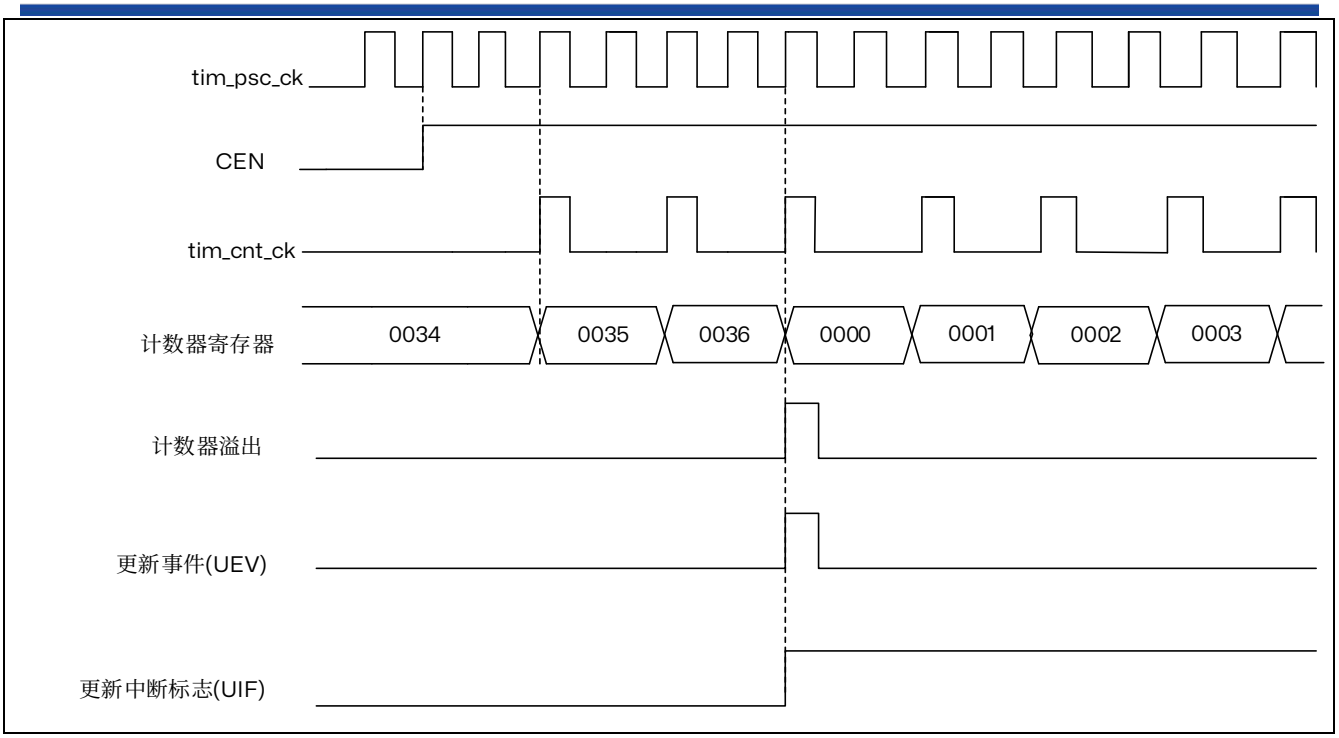


图 19.5 计数器时序图，内部时钟分频系数为 2

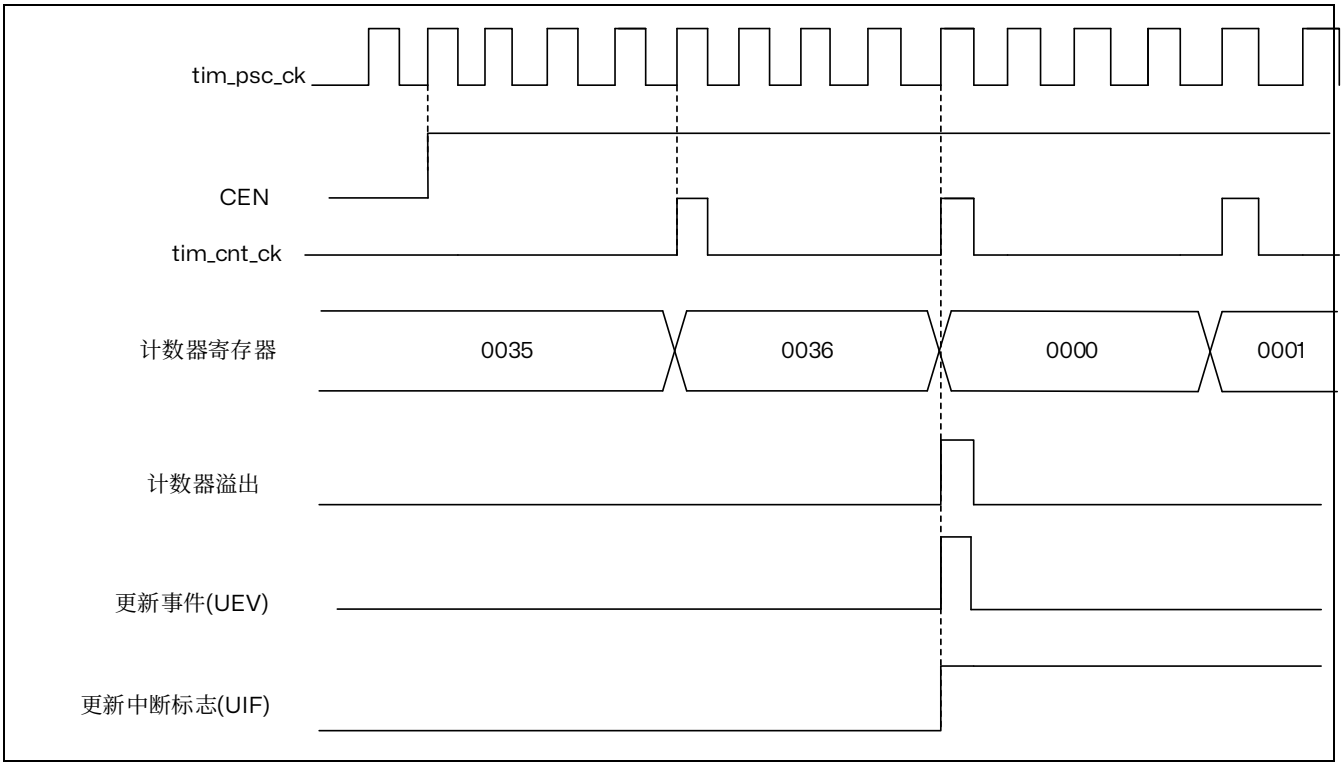


图 19.6 计数器时序图，内部时钟分频系数为 4

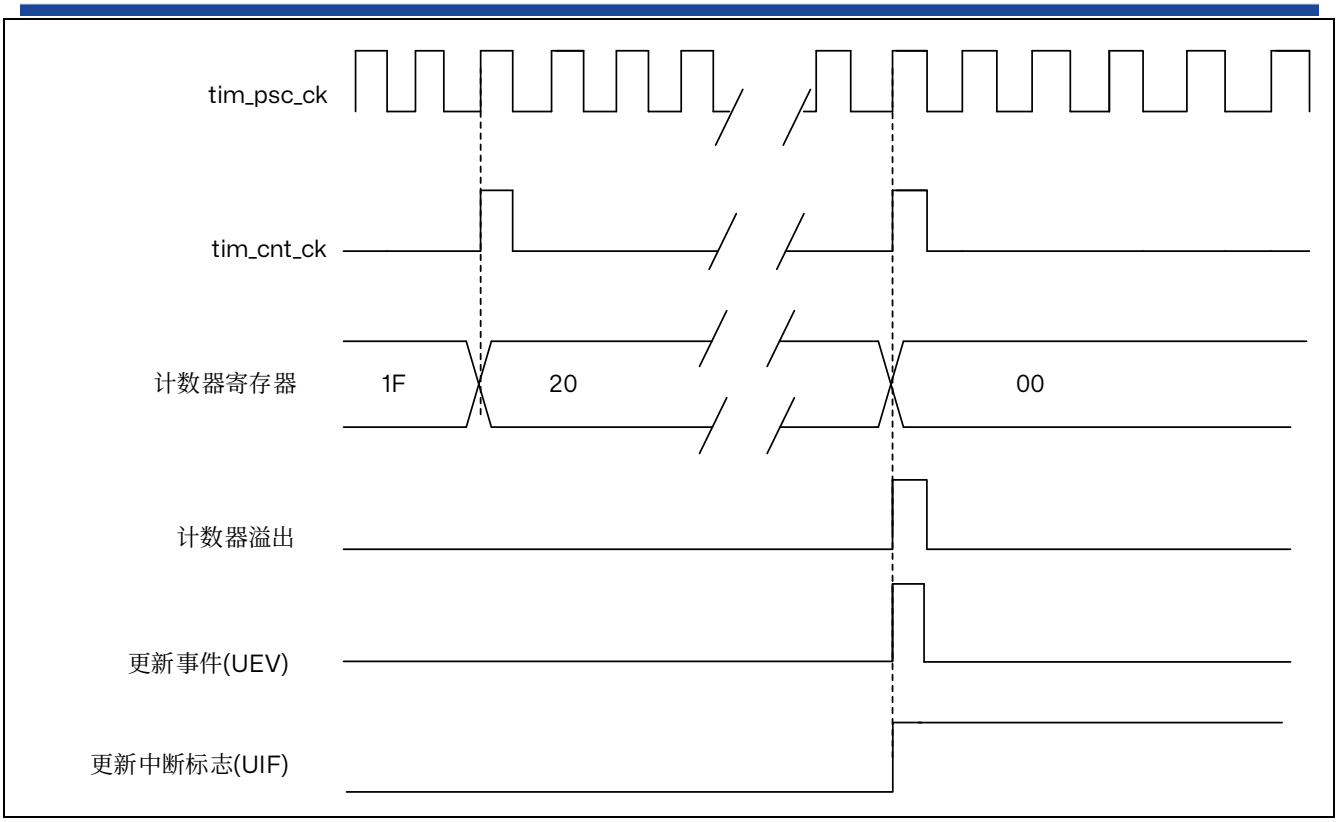


图 19.7 计数器时序图，内部时钟分频系数为 N

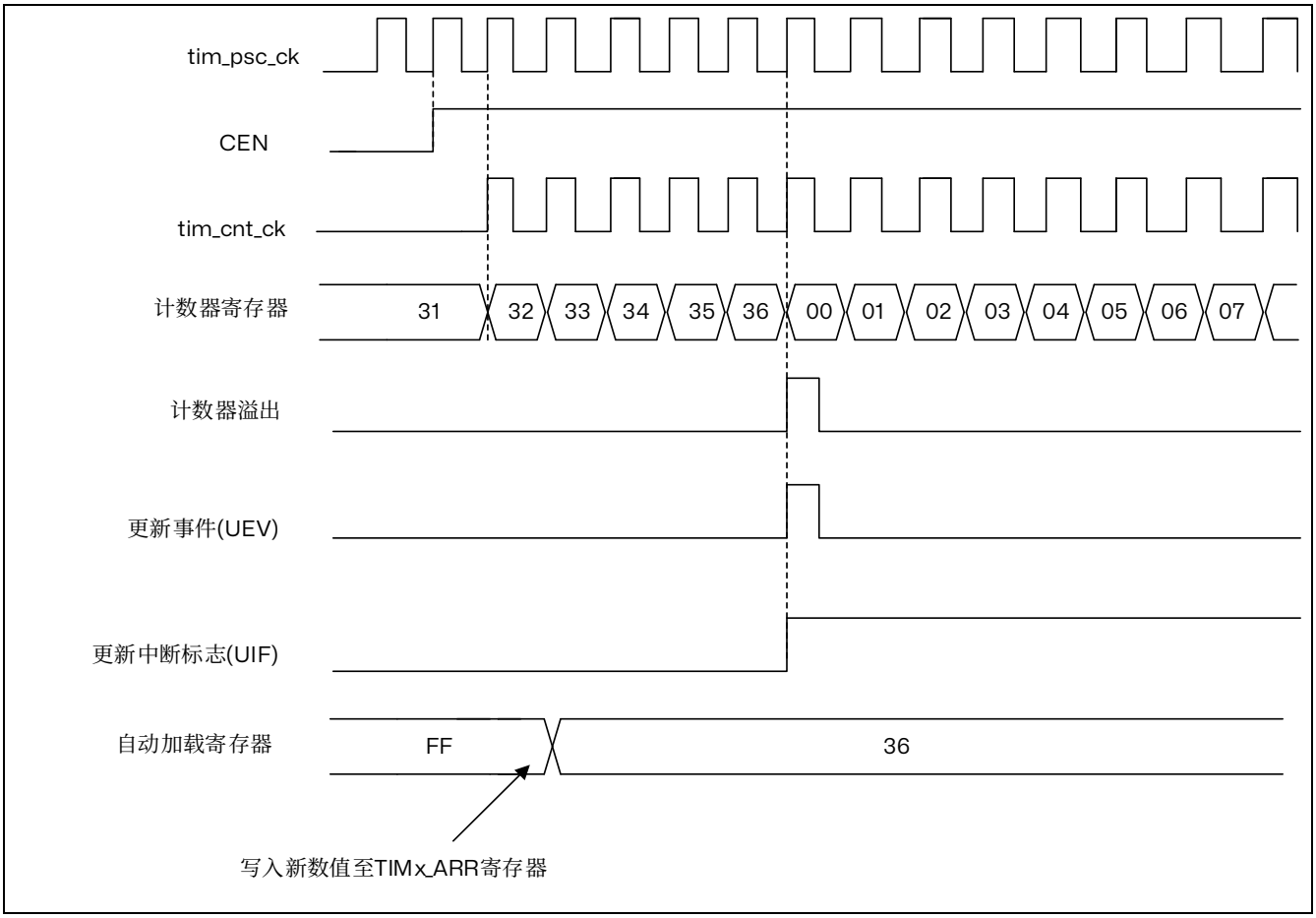


图 19.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

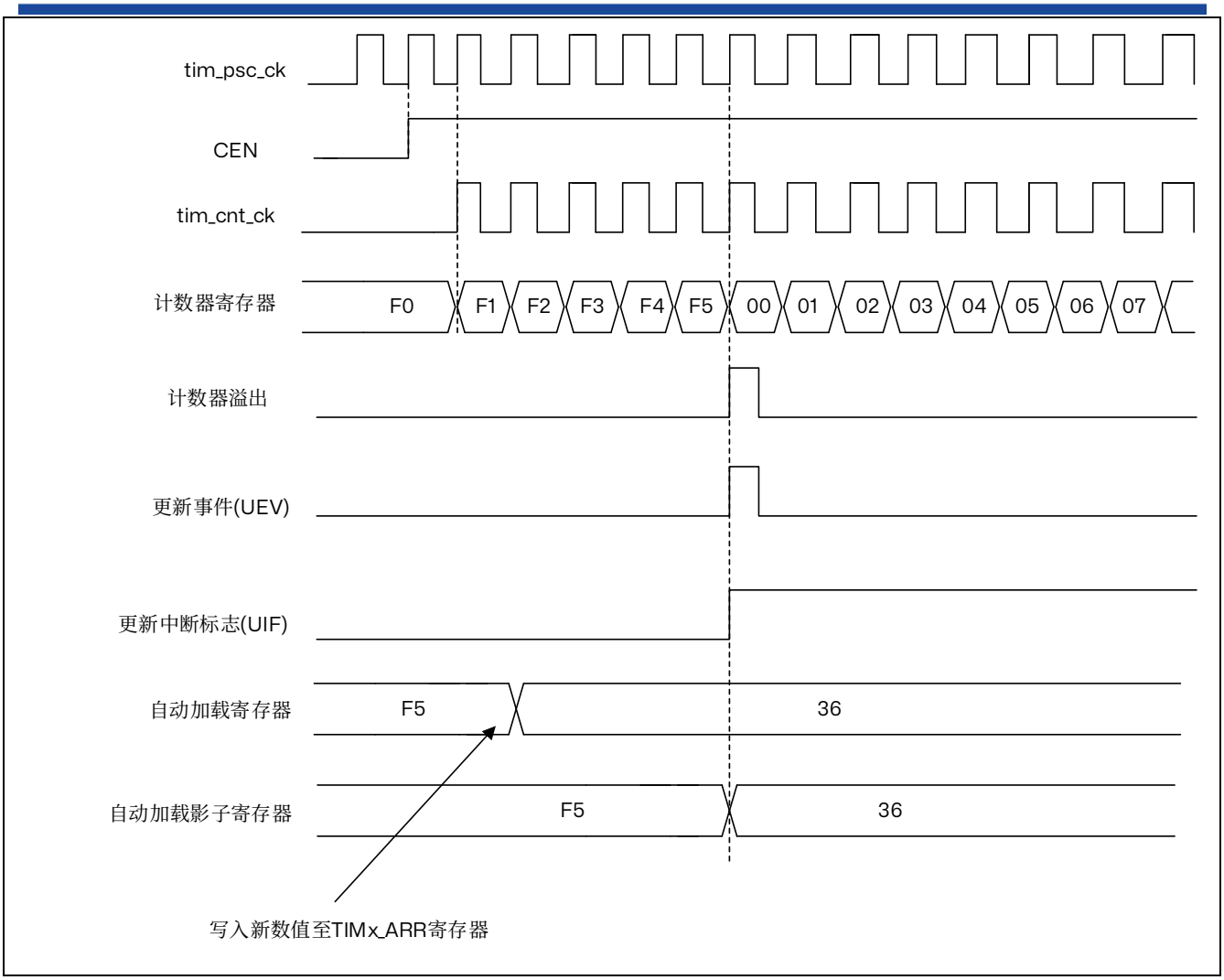


图 19.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

每次发生计数器下溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器会重新从当前自动重载值开始计数，而预分频器计数器则重新从 0 开始计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，自动重载寄存

器会在计数器重载之前得到更新，因此，下一个计数周期就是我们所希望的新的周期长度。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

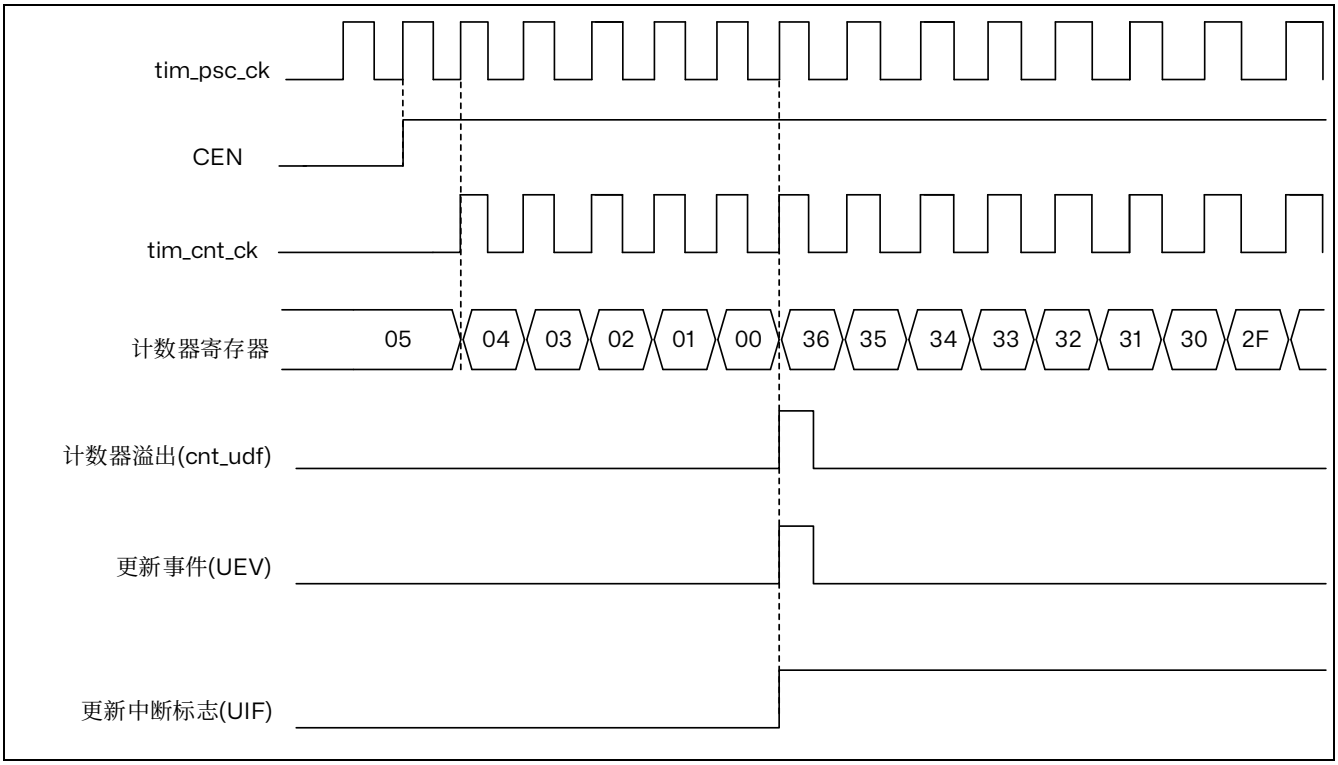


图 19.10 计数器时序图，内部时钟分频系数为 1

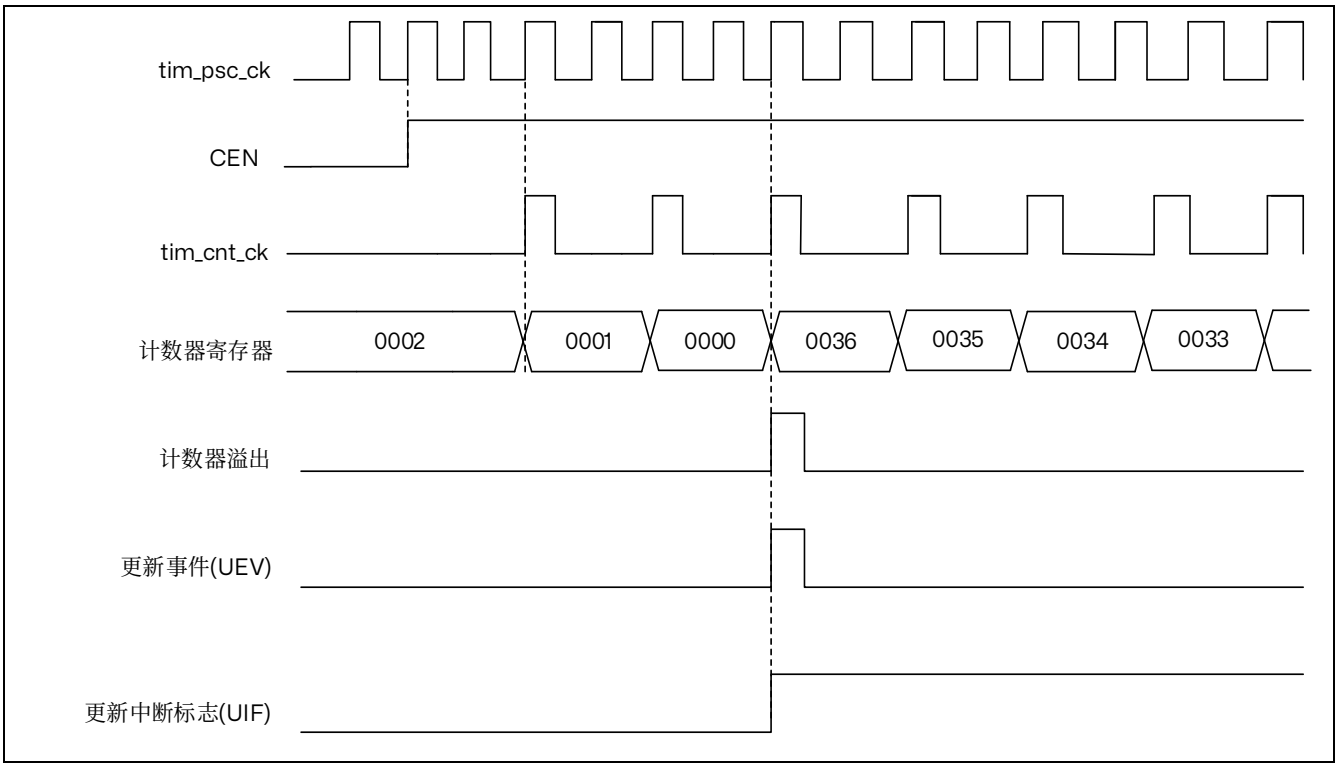


图 19.11 计数器时序图，内部时钟分频系数为 2

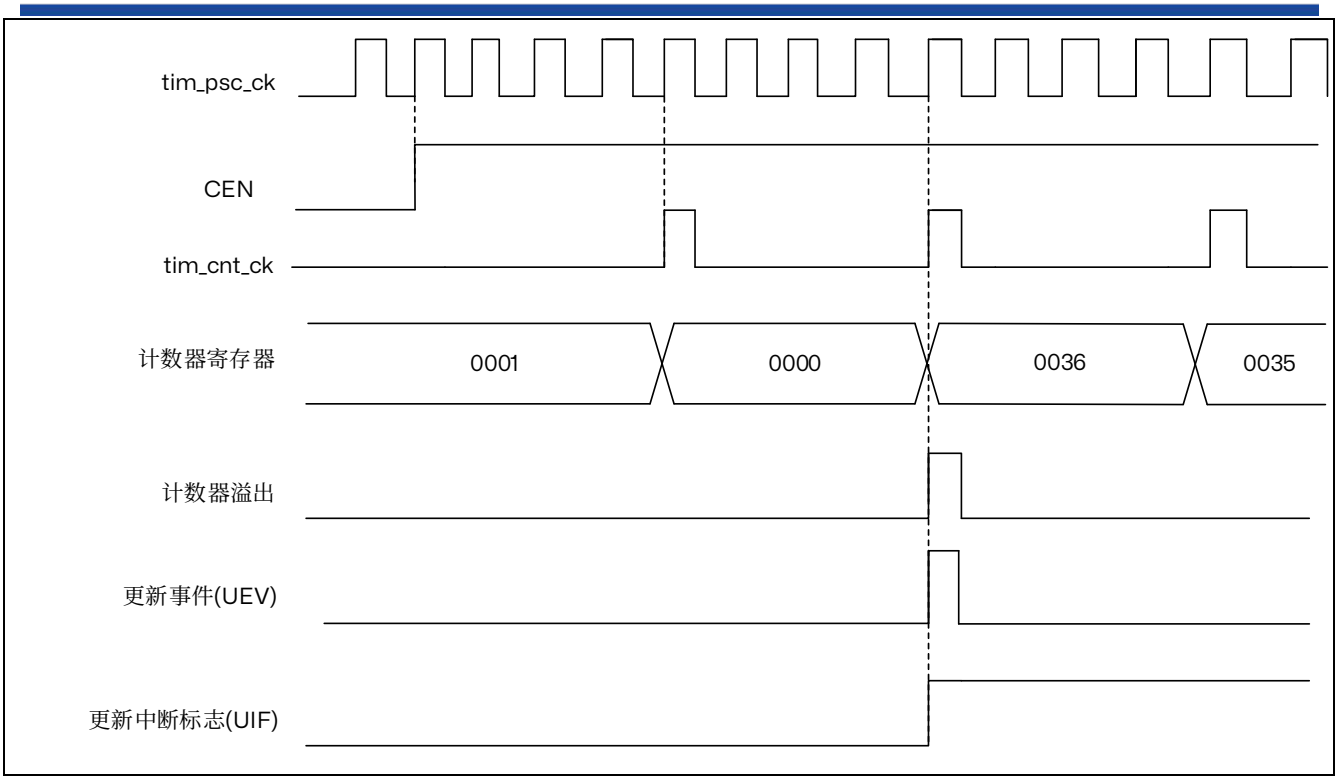


图 19.12 计数器时序图，内部时钟分频系数为 4

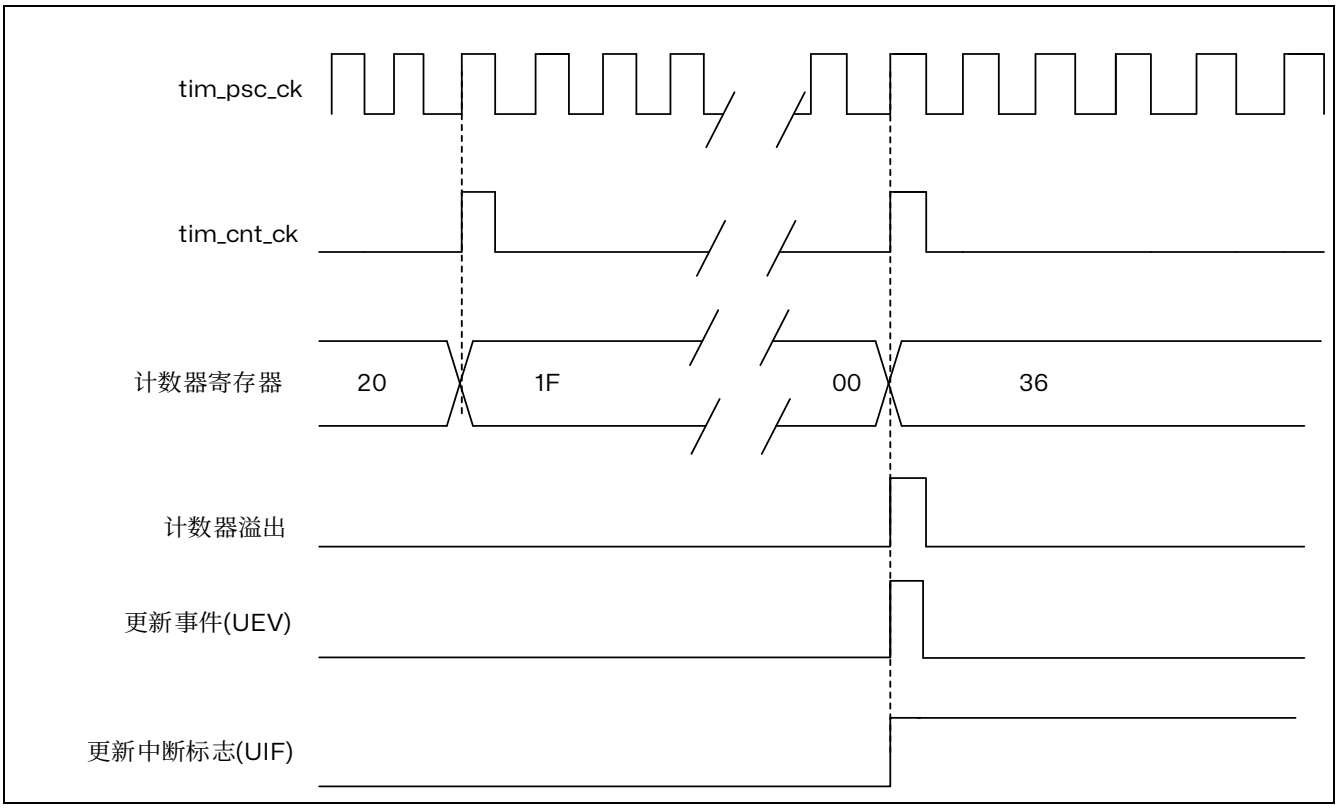


图 19.13 计数器时序图，内部时钟分频系数为 N

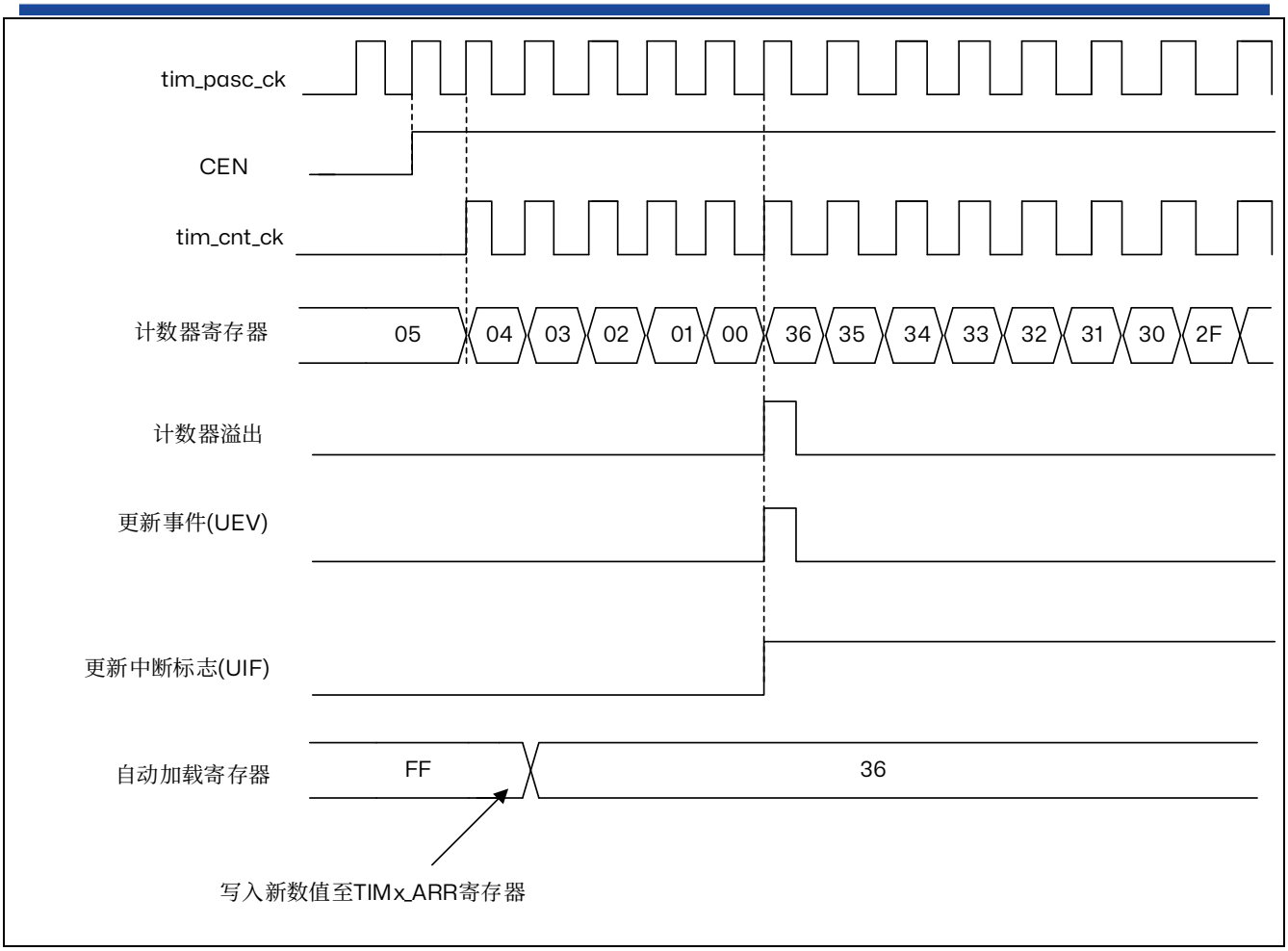


图 19.14 计数器时序图，更新事件

中央对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS = “01”）、计数器递增计数（中心对齐模式 2，CMS = “10”）以及计数器递增/递减计数（中心对齐模式 3，CMS = “11”）。

此模式下无法写入方向位（TIMx_CR1 寄存器中的 DIR 位）。而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

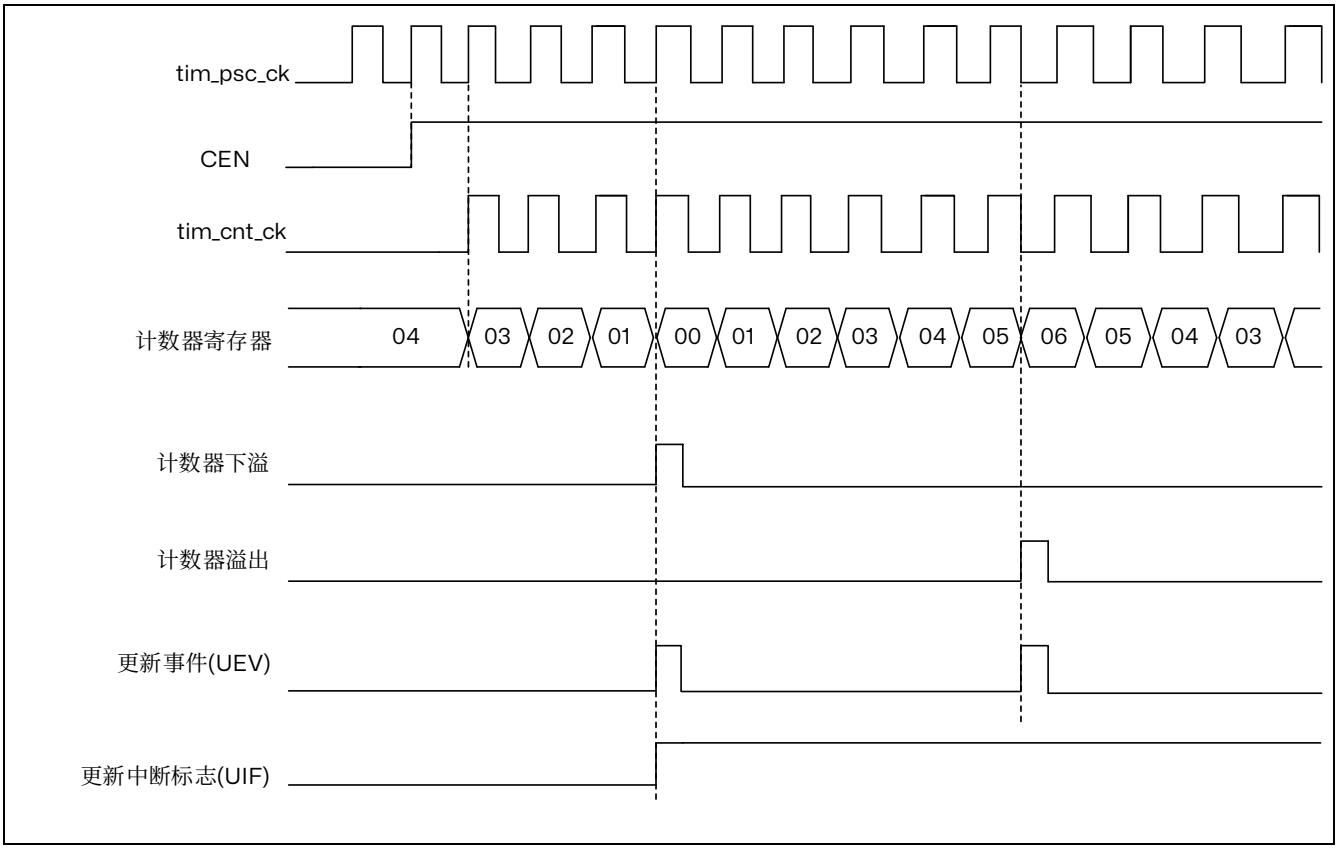
此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将

计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 当自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，如果更新操作是由计数器上溢触发的，则自动重载寄存器在重载计数器之前更新，因此，下一个计数周期就是我们所希望的新的周期长度（计数器被重载新的值）。

以下各图以一些示例说明不同时钟频率下计数器的行为。



1. 这里使用了中心对齐模式 1

图 19.15 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6

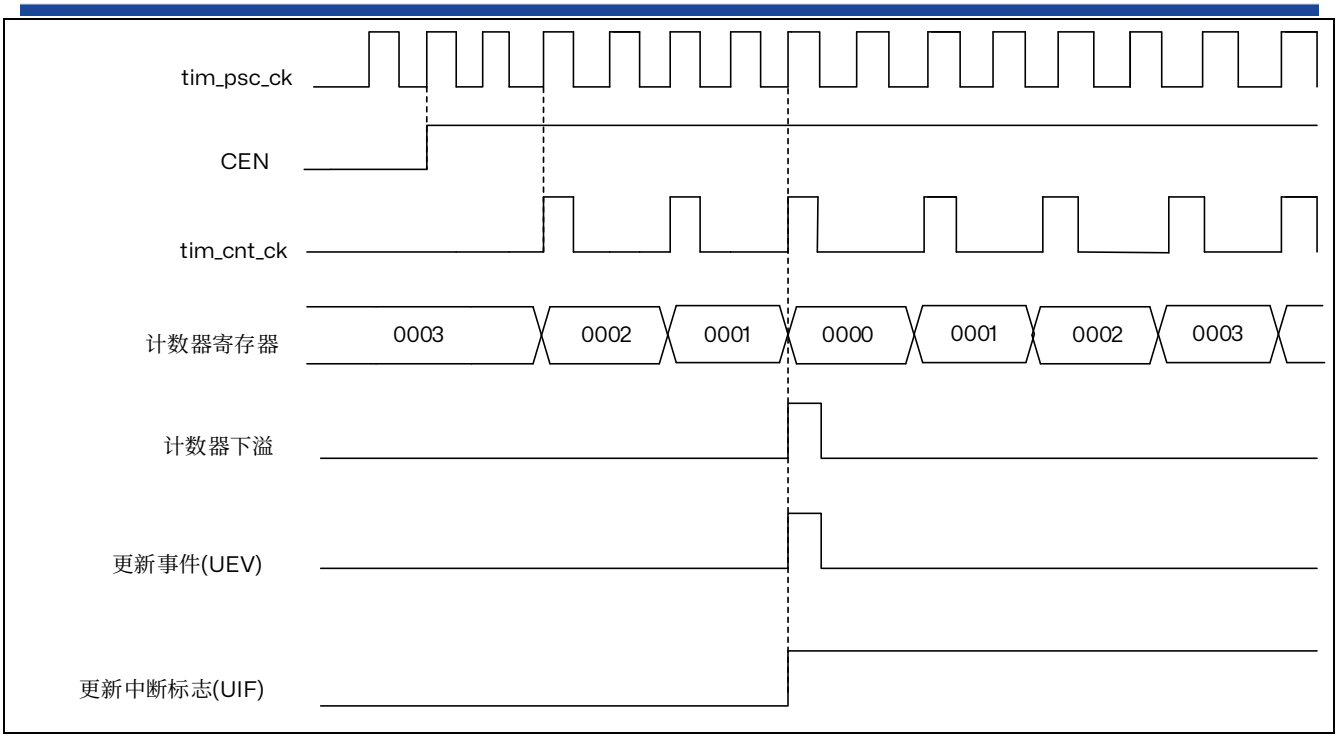


图 19.16 计数器时序图，内部时钟分频系数为 2

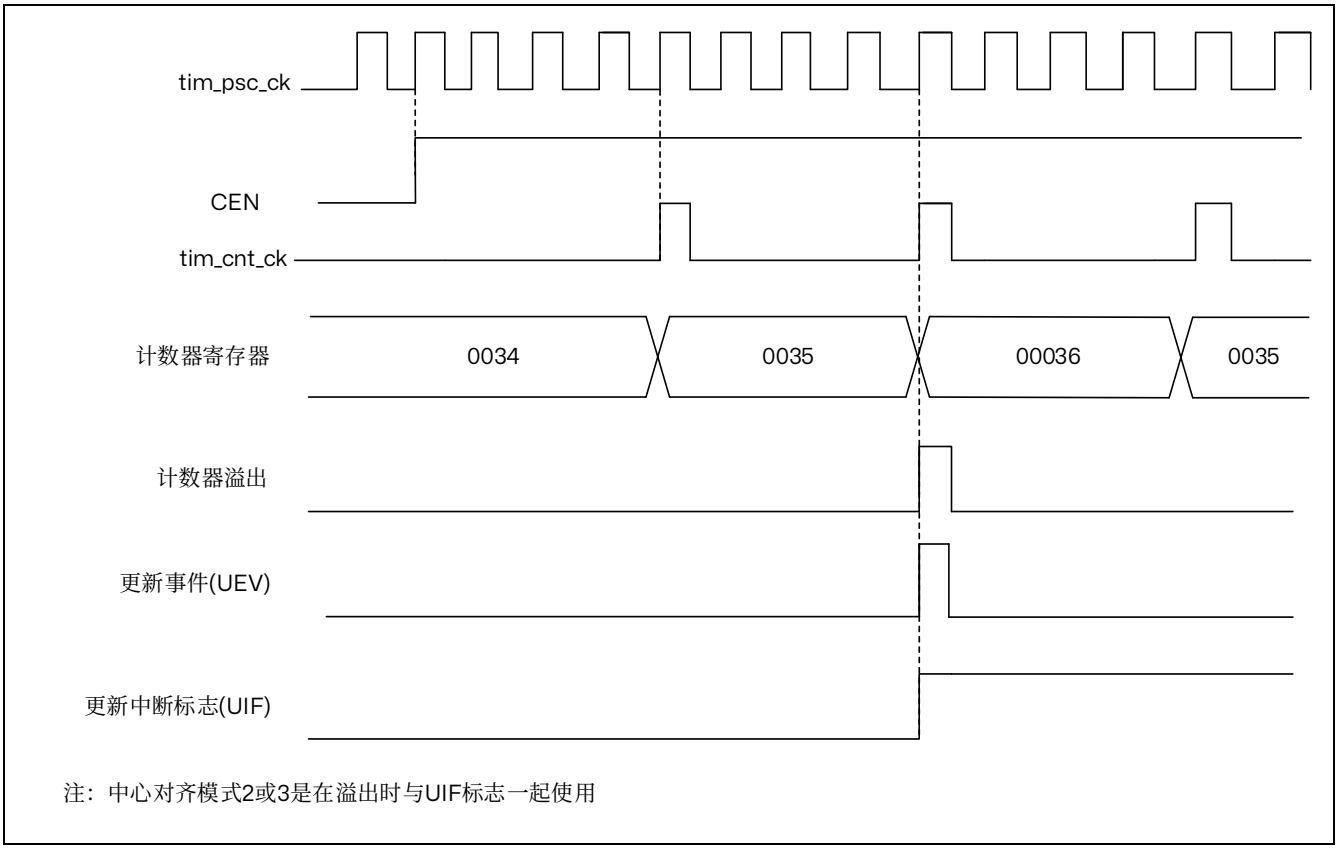


图 19.17 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36

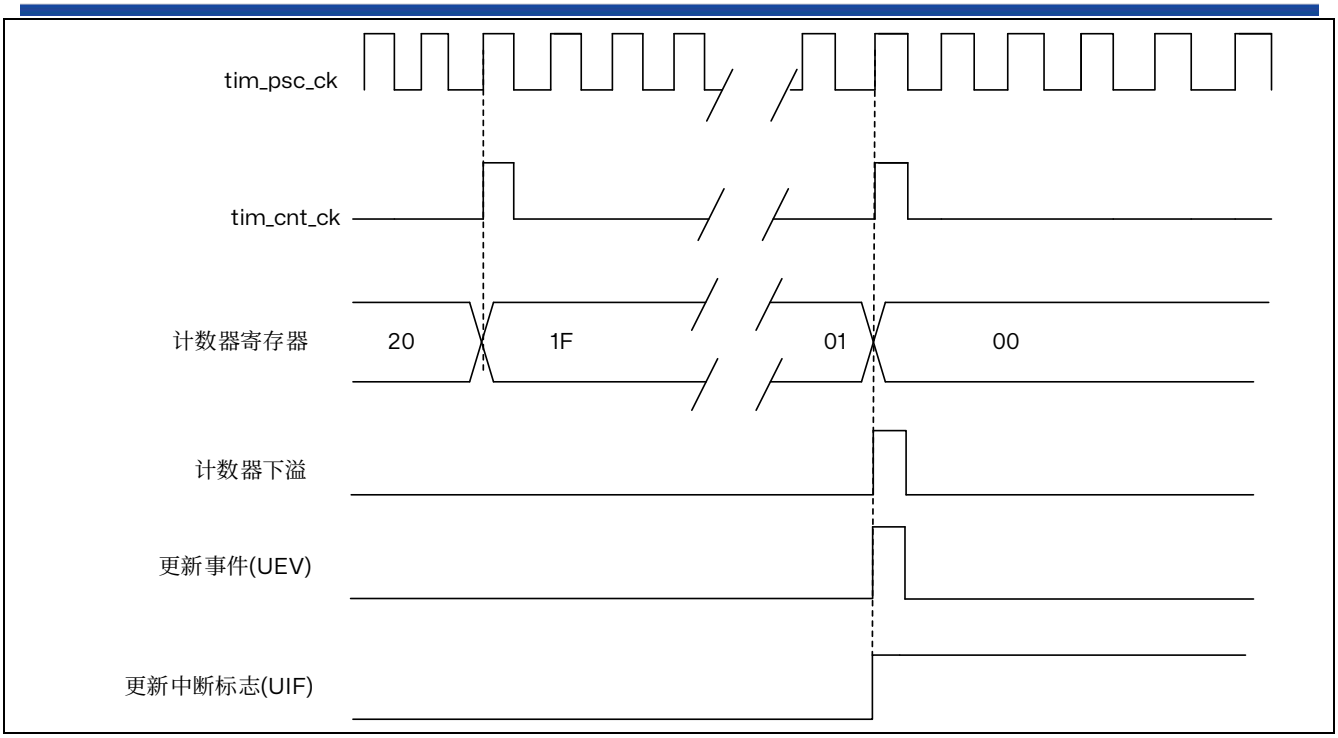


图 19.18 计数器时序图，内部时钟分频系数为 N

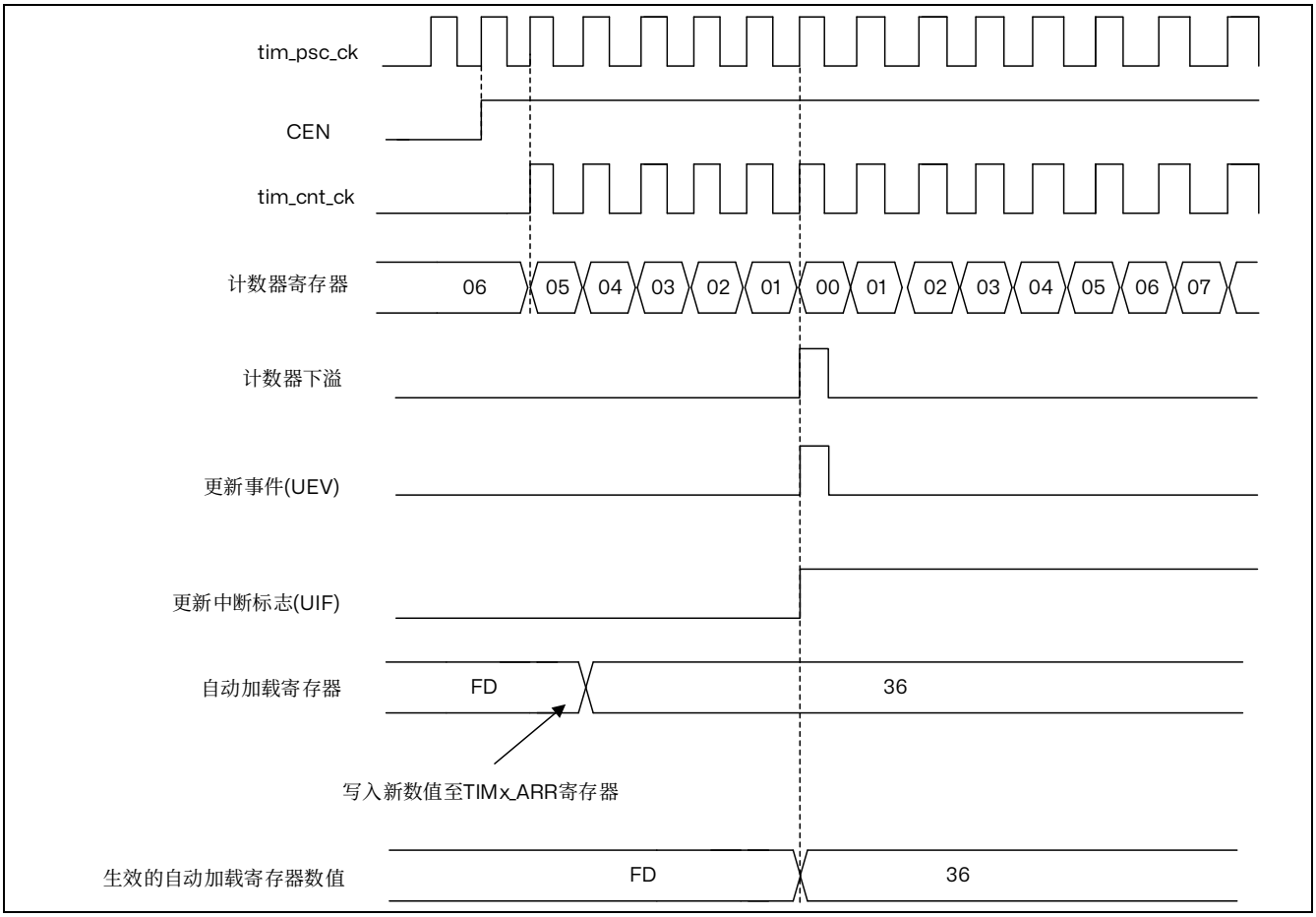


图 19.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

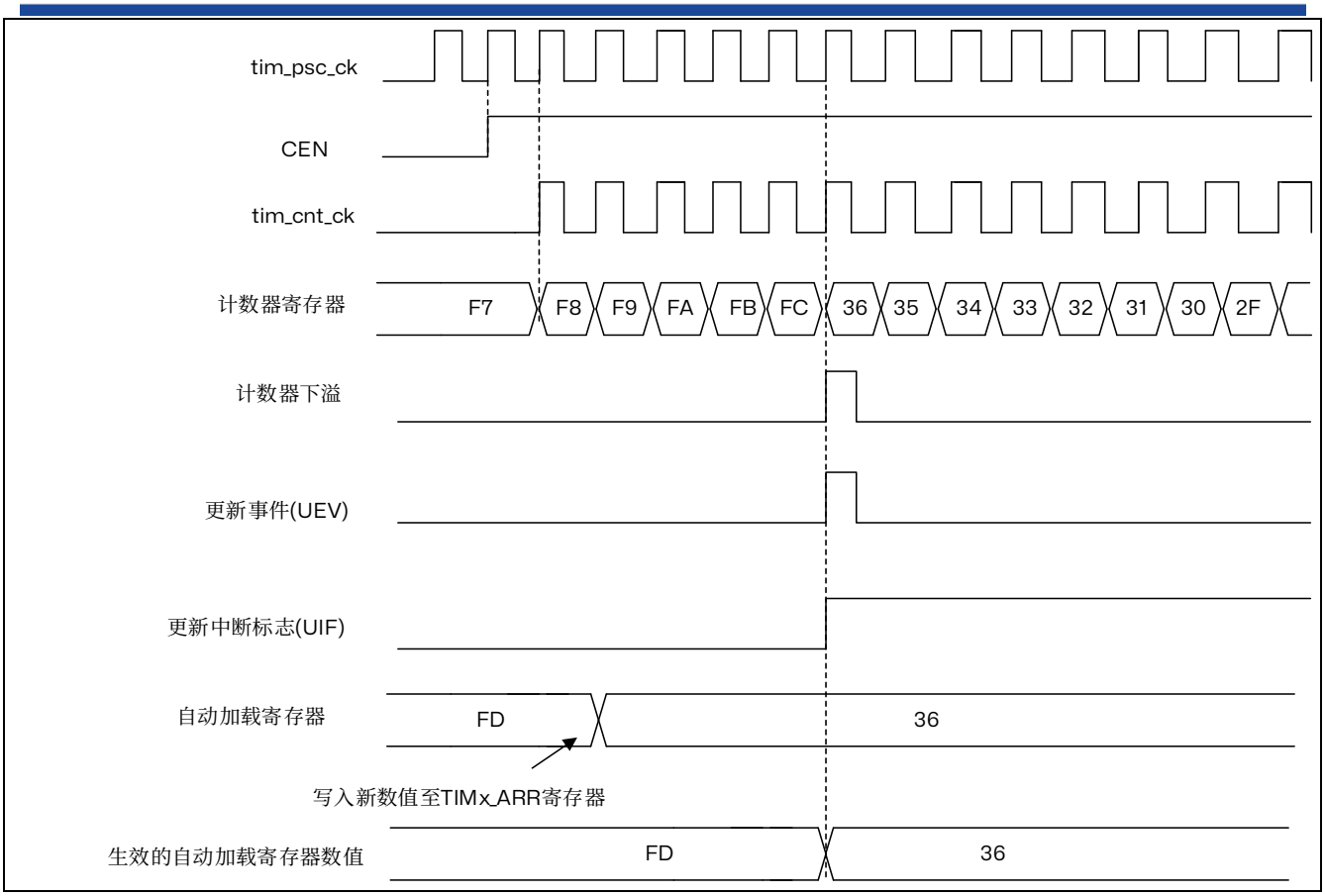


图 19.20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

19.4.5 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（tim_ker_ck）
- 外部时钟模式 1：外部输入引脚（tim_ti1 或 tim_ti2）
- 内部触发输入（tim_itr）：使用一个定时器作为另一个定时器的预分频器，例如可以将定时器 8 配置为定时器 2 的预分频器。

内部时钟源（tim_ker_ck）

如果禁止从模式控制器（TIMx_SMCR 寄存器中 SMS=000），则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

下图显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

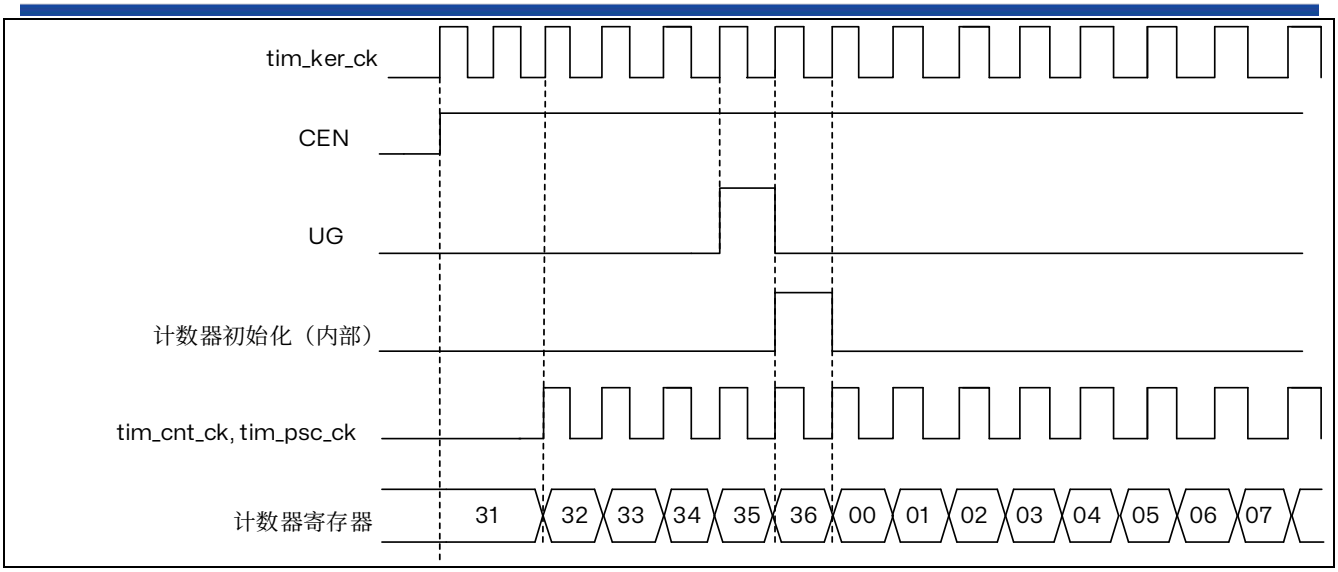


图 19.21 一般模式下的控制电路，内部时钟分频系数为 1

外部时钟源模式 1

当 TIMx_SMCR 寄存器中的 SMS=111 时，可选择此模式。计数器可在选定的输入信号上出现上升沿或下降沿时计数。

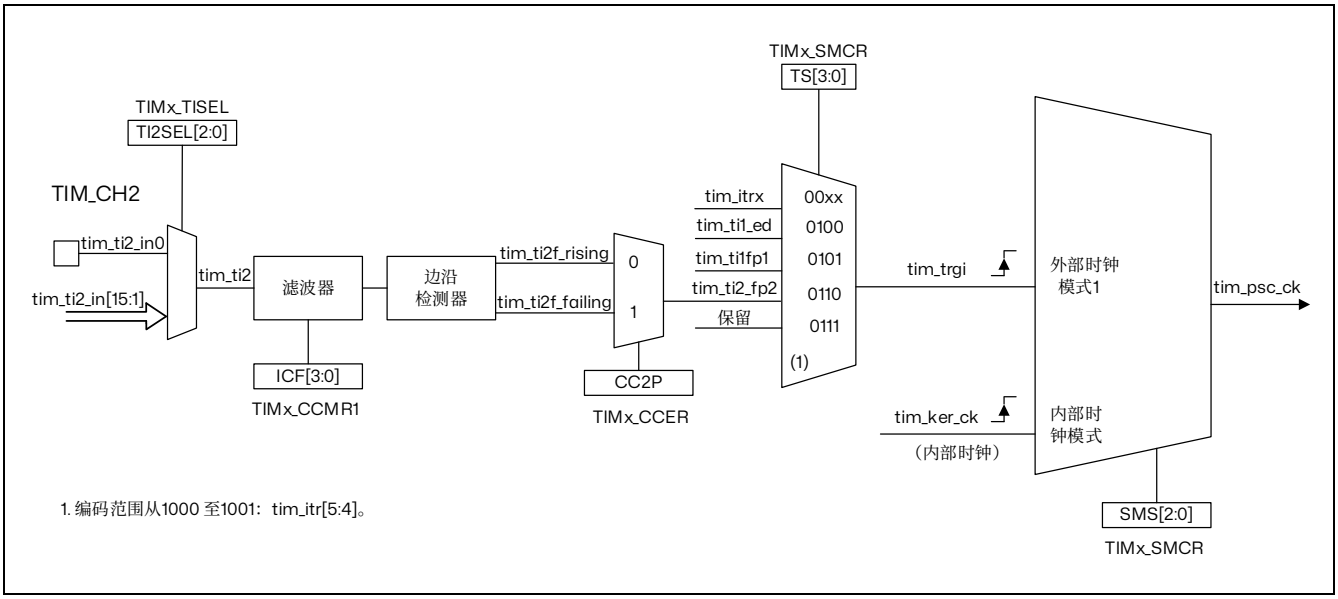


图 19.22 tim_ti2 外部时钟连接例子

例如，要使递增计数器在 tim_ti2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIMx_TISEL 寄存器中使用 TI2SEL[2:0]位选择合适的 tim_ti2_in[15:0]源（内部或外部）。
2. 通过在 TIMx_CCMR1 寄存器中写入 CC2S=“01”来配置通道 2，使其能够检测 tim_ti2 输入的上升沿。
3. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0]位来配置输入滤波时间（如果不需要任何滤波，请保持 IC2F=0000）。

注：由于捕获预分频器不用于触发操作，因此无需对其进行配置。

4. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。

5. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
6. 通过在 TIMx_SMCR 寄存器中写入 TS=110 来选择 tim_ti2 作为输入源。
7. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

当 tim_ti2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器时钟之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

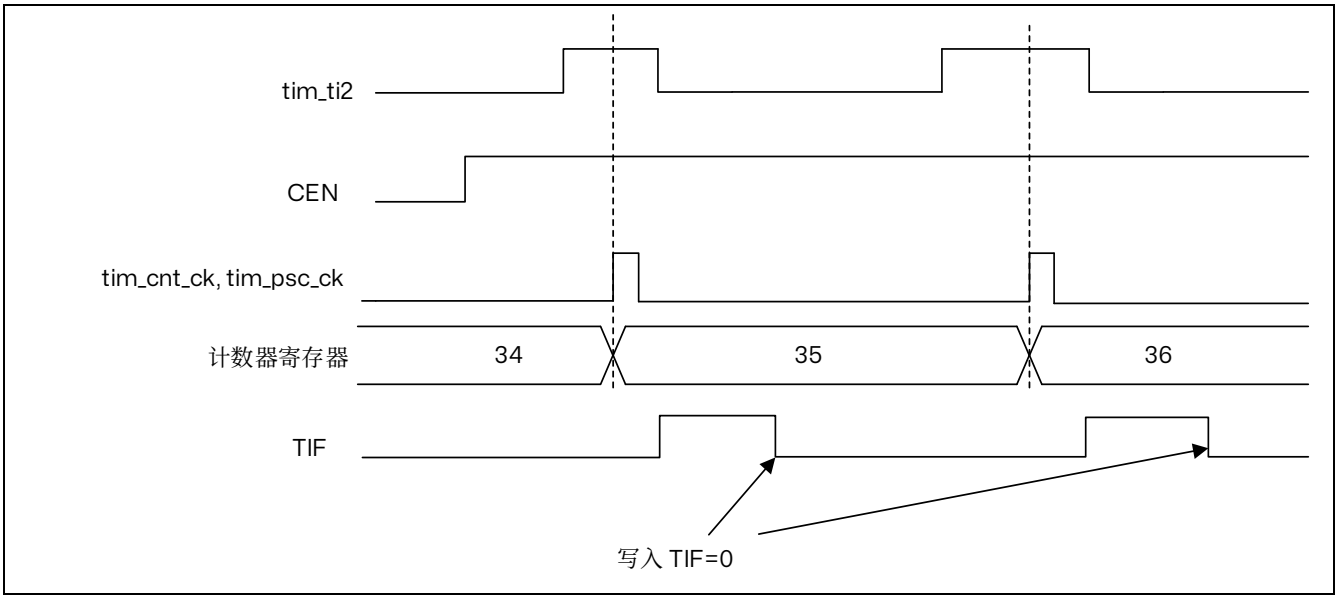


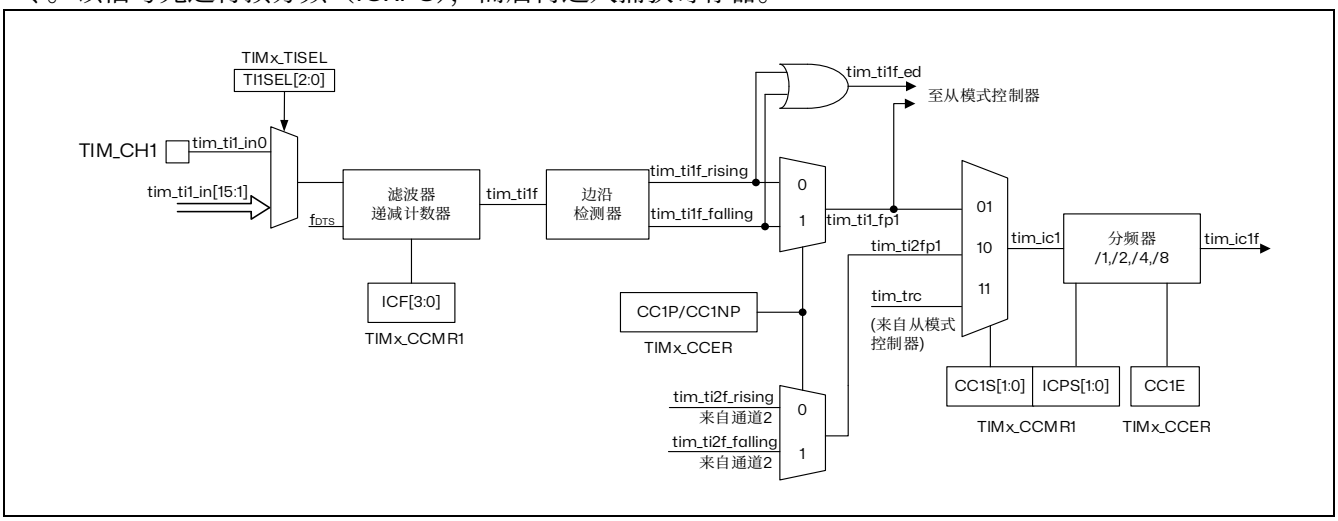
图 19.23 外部时钟模式 1 下的控制电路

19.4.6 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器）和一个输出阶段（比较器和输出控制）构建而成。

下图概括介绍了一个捕获/比较通道。

输入阶段对相应的 tim_tix 输入进行采样，生成一个滤波后的信号 tim_tixf。然后，带有极性选择功能的边沿检测器生成一个信号（tim_tixfpy），该信号可用作从模式控制器的触发输入，也可用作捕获命令。该信号先进行预分频（ICxPS），而后再进入捕获寄存器。



输出阶段生成一个中间波形作为基准：OCxRef（高电平有效）。链的末端决定最终输出信号的极性。

图 19.24 捕获/比较通道（如：通道 1 输入部分）

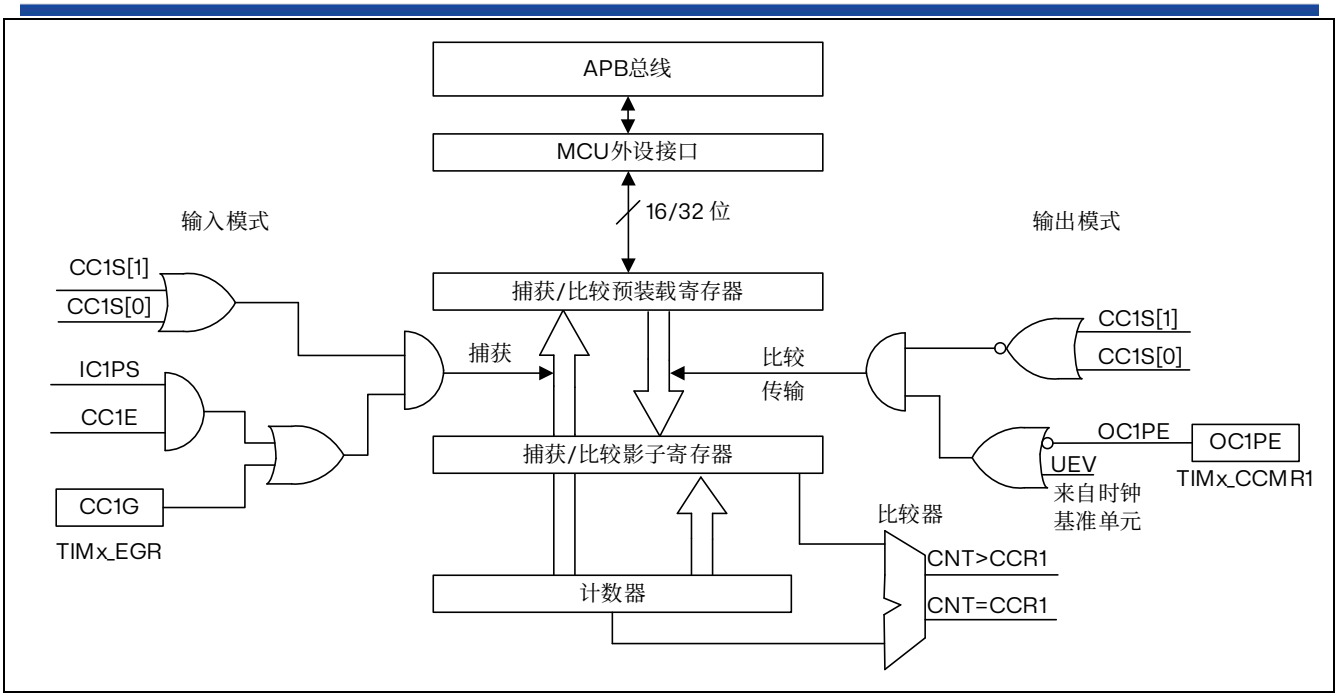


图 19.25 捕获/比较通道 1 的主电路

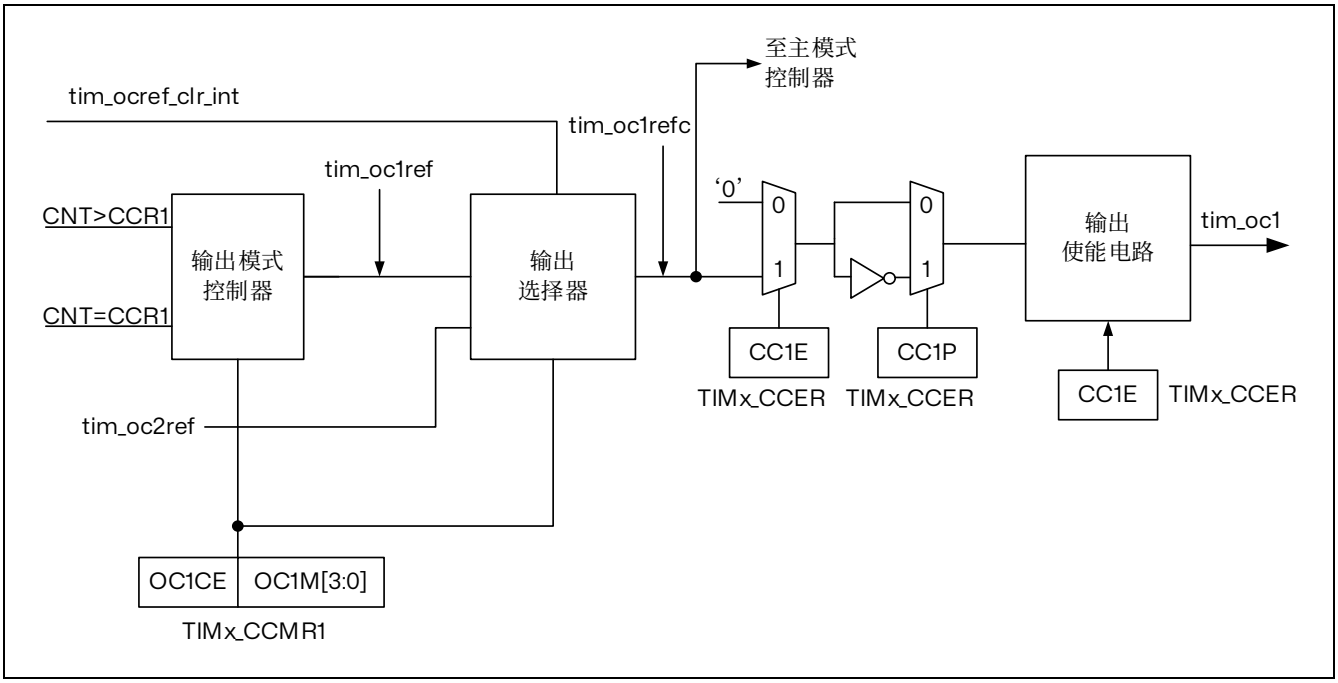


图 19.26 捕获/比较通道的输出阶段（通道 1，通道 2、3 和 4 同上）

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。始终可通过读写操作访问预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容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将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

19.4.7 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器(TIMx_CCRx) 来锁存计数器的值。发生捕获事件时，会将相应的 CCXIF 标志（TIMx_SR 寄存器）置 1，并可发送中断请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将重复捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件向 CCxIF 写入 0 来给 CCxIF 清零，或读取存储在 TIMx_CCRx 寄存器中的已捕获数据。向 CCxOF 写入 0 后会将其清零。

以下示例说明了如何在 tim_ti1 输入出现上升沿时将计数器的值捕获到 TIMx_CCR1 中。具体操作步骤如下：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:0]源（内部或外部）。
2. 选择有效输入：TIMx_CCR1 必须连接到 tim_ti1 输入，因此向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01。只要 CC1S 不等于 00，就会将通道配置为输入模式，并且 TIMx_CCR1 寄存器将处于只读状态。
3. 根据连接到定时器的信号，对所需的输入滤波时间进行编程（如果输入为 tim_tix 输入之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波时间设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 tim_ti1 上的跳变沿。然后向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。
4. 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 0，选择 tim_ti1 通道的有效转换边沿（本例中为上升沿）。
5. 对输入预分频器进行编程。在本例中，我们希望每次有效转换时都执行捕获操作，因此需要禁止预分频器（向 TIMx_CCMR1 寄存器中的 IC1PSC 位写入 00）。
6. 通过将 TIMx_CCER 寄存器中的 CC1E 位置 1，允许将计数器的值捕获到捕获寄存器中。
7. 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成一个中断。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

*注：通过软件将*TIMx_EGR *寄存器中的相应*CCxG *位置*1 *可生成*IC *中断请求。*

19.4.8 PWM 输入模式

此模式允许测量连接到单个 tim_tix 输入的 PWM 信号的周期和占空比：

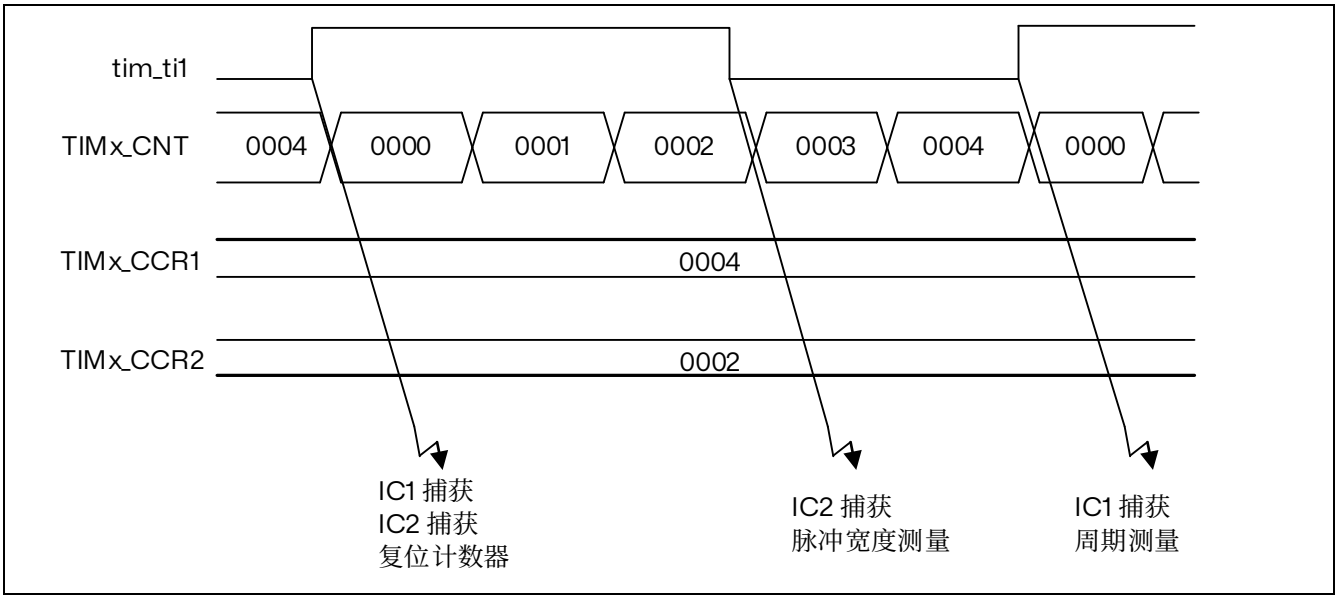
- TIMx_CCR1 寄存器保持周期值（两个连续上升沿之间的间隔）。
- TIMx_CCR2 寄存器保持脉冲宽度（两个连续上升沿和下降沿之间的间隔）。

此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 ICx 信号被映射至同一个 tim_tix 输入。
- 这两个 ICx 信号在边沿处有效，但极性相反。
- 选择两个 TIxFP 信号之一作为触发输入，并将从模式控制器配置为复位模式。

可通过以下步骤测量施加在 tim_ti1 上的 PWM 信号的周期和脉冲宽度：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:0]源（内部或外部）。
2. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 tim_ti1）。
3. 选择 tim_ti1fp1 的有效极性（用于 TIMx_CCR1 中的捕获和计数器清零）：向 CC1P 位和 CC1NP 位写入“0”（上升沿有效）。
4. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 tim_ti1）。
5. 选择 tim_ti1fp2 的有效极性（用于 TIMx_CCR2 中的捕获）：向 CC2P 位写入“1”和 CC2NP 位写入“0”（下降沿有效）。
6. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 101（选择 tim_ti1fp1）。
7. 将从模式控制器配置为复位模式：向 TIMx_SMCR 寄存器中的 SMS 位写入 100。
8. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。



PWM 输入模式只能与 TIMx_CH1/TIMx_CH2 信号一起使用，因为只有 tim_ti1fp1 和 tim_ti2fp2 连接到从模式控制器。

图 19.27 PWM 输入模式时序

19.4.9 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS 位= 00）下，可直接由软件将每个输出比较信号（tim_ocxref 和 tim_ocx）强制设置为有效电平或无效电平，而无需考虑输出比较寄存器和计数器之间的任何比较结果。

要将输出比较信号（tim_ocxref/tim_ocx）强制设置为有效电平，只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 101。tim_ocxref 进而强制设置为高电平（tim_ocxref 始终为高电平有效），同时 tim_ocx 获取 CCxP 极性位的相反值。

例如：CCxP=0（tim_ocx 高电平有效）=> tim_ocx 强制设置为高电平。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 100，可将 tim_ocxref 信号强制设置为低电平。

无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可发送相应的中断请求。输出比较模式一节对此进行了介绍。

19.4.10 输出比较模式

此功能用于控制输出波形，或指示已经过某一时间段。

当捕获/比较寄存器与计数器之间相匹配时，输出比较功能：

- 将为相应的输出引脚分配一个可编程值，该值由输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）定义。匹配时，输出引脚既可保持其电平（OCXM=000），也可设置为有效电平（OCXM=001）、无效电平（OCXM=010）或进行翻转（OCxM=011）。
- 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
- 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装载寄存器。A

在输出比较模式下，更新事件 UEV 对 tim_ocxref 和 tim_ocx 输出毫无影响。同步的精度可以达到计数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，将 CCxIE 位置 1。
4. 选择输出模式。例如，
   a) 写入 OCxM=0011 以在 CNT 匹配 CCRx 时翻转 tim_ocx 输出引脚
   b) 写入 OCxPE=0 以禁用预装载寄存器
   c) 写入 CCxP=0 以选择高电平有效
   d) 写入 CCxE=1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可随时通过软件更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预装载寄存器（OCxPE=0，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。

下图列出了相关示例。

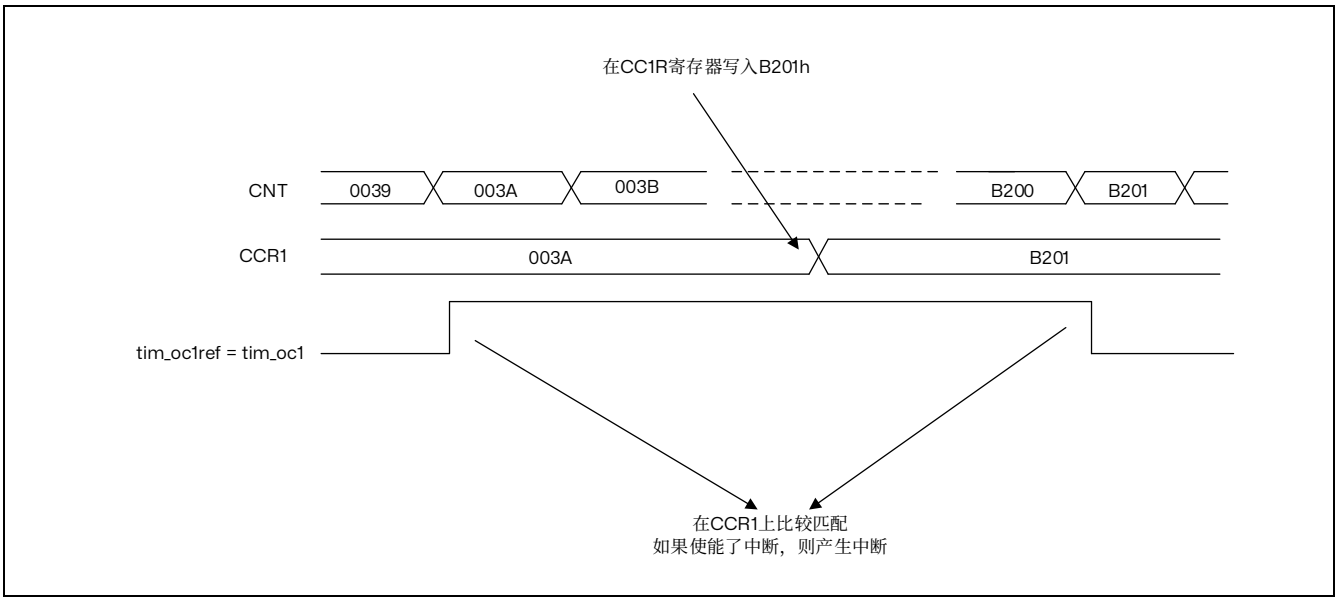


图 19.28 输出比较模式，翻转 tim_oc1

19.4.11 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），可以独立选择各通道（每个 tim_ocx 输出对应一个 PWM）的 PWM 模式。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器。

由于只有在发生更新事件时预装载寄存器才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

tim_ocx 极性可使用 TIMx_CCER 寄存器的 CCxP 位来编程。既可以设为高电平有效，也可以设为低电平有效。tim_ocx 输出通过将 TIMx_CCER 寄存器中的 CCxE 位置 1 来使能。有关详细信息，请参见 TIMx_CCERx 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 始终与 TIMx_CCRx 进行比较，以确定是 TIMx_CCRx ≤ TIMx_CNT 还是 TIMx_CNT ≤ TIMx_CCRx（取决于计数器计数方向）。外部事件可以通过 tim_etr_in 或 tim_oceref_clr 信号。在本例中仅声明了 tim_ocref_clr 信号

- 比较匹配事件之后
- 输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）从“冻结”配置（不进行比较，OCxM=“000”）切换为任一 PWM 模式（OCxM=“110”或“111”）。定时器运行期间，可以通过软件强制 PWM 输出。

根据 TIMx_CR1 寄存器 CMS 位状态，定时器能够产生边沿对齐模式或中心对齐模式的 PWM 信号。

PWM 边沿对齐模式

递增计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低时执行递增计数。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 tim_ocxref 便为高电平，否则为低电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。如果比较值为 0，则 tim_ocxref 保持为“0”。下图举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

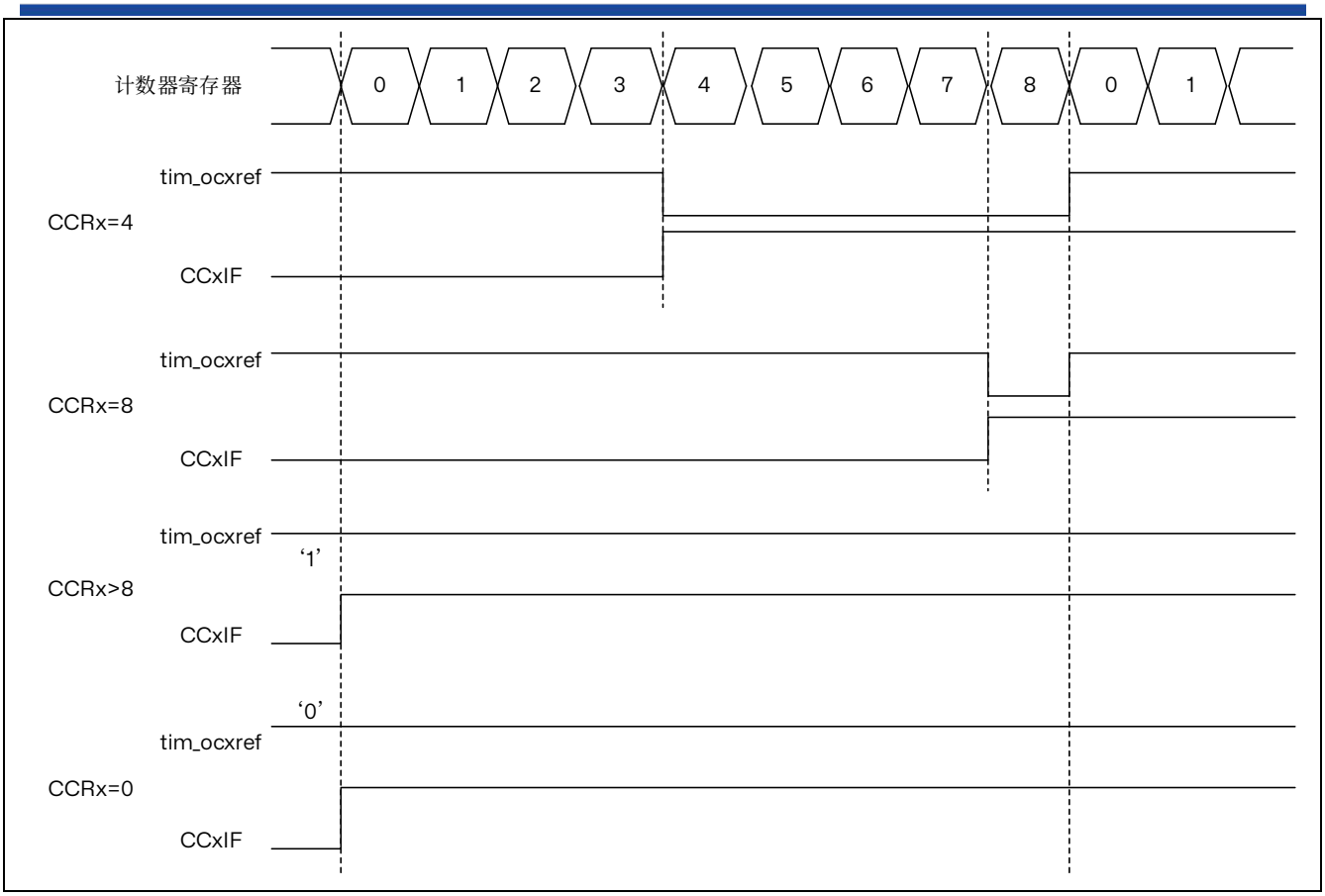


图 19.29 边沿对齐的 PWM 波形（ARR=8）

递减计数配置

当 TIMx_CR1 寄存器中的 DIR 位为高时执行递减计数。

在 PWM 模式 1 下，只要 TIMx_CNT>TIMx_CCRx，参考信号 tim_ocxref 便为低电平，否则为高电平。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重载值，则 tim_ocxref 保持为“1”。此模式下不可能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时（其余所有配置对 tim_ocxref/tim_ocx 信号具有相同的作用），中心对齐模式生效。根据 CMS 位的配置，可以在计数器递增计数、递减计数或同时递增和递减计数时将比较标志置 1。TIMx_CR1 寄存器中的方向位（DIR）由硬件更新，不得通过软件更改。

下图显示了中心对齐模式的 PWM 波形，在此例中：

- TIMx_ARR=8
- PWM 模式为 PWM 模式 1
- 在根据 TIMx_CR1 寄存器中 CMS=01 而选择的中心对齐模式 1 下，当计数器递减计数时，比较标志置 1。

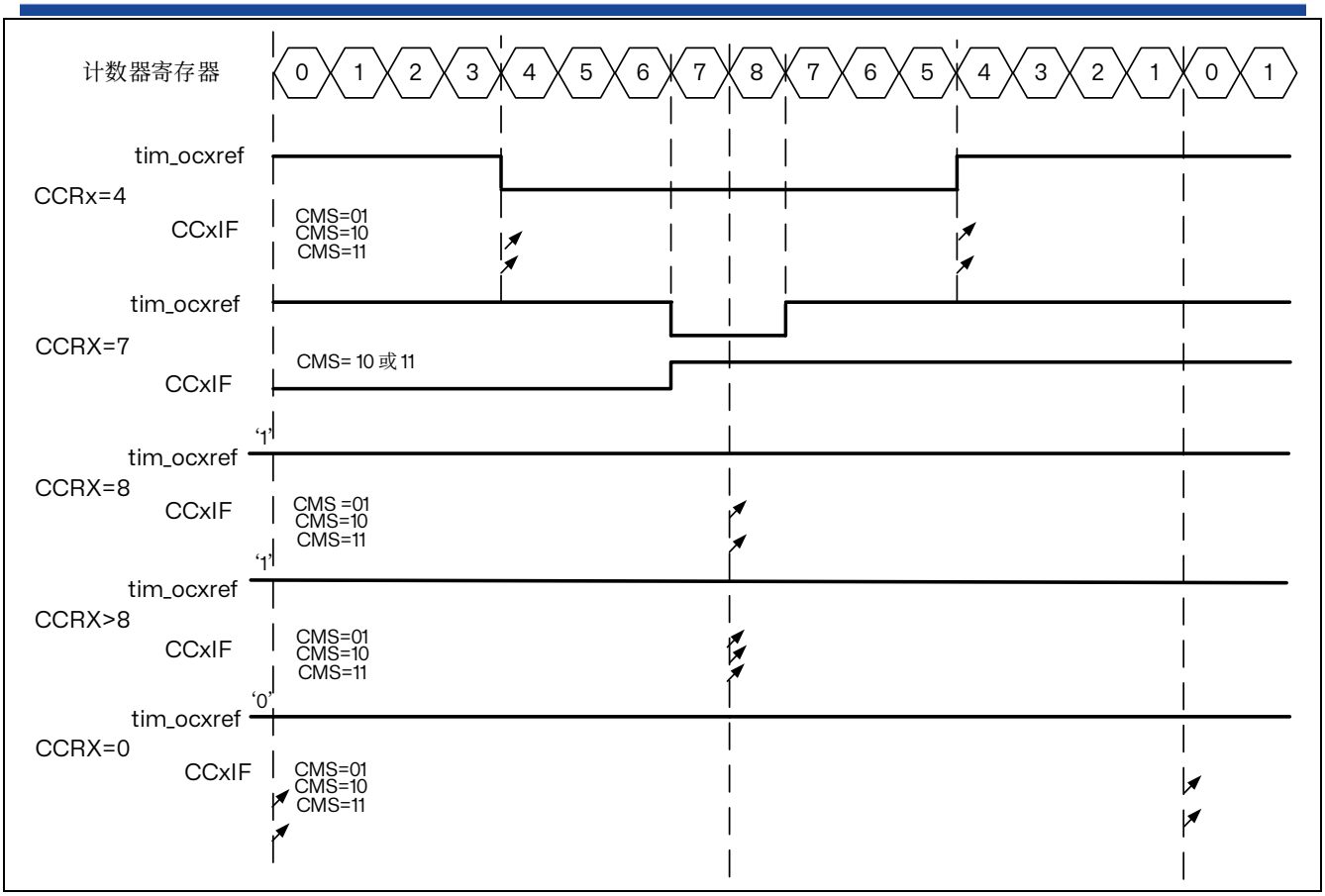


图 19.30 中央对齐的 PWM 波形（APR=8）

中心对齐模式使用建议：

- 启动中心对齐模式时将使用当前的递增/递减计数配置。这意味着计数器将根据写入 TIMx_CR1 寄存器中 DIR 位的值进行递增或递减计数。此外，不得同时通过软件修改 DIR 和 CMS 位。
- 不建议在运行中心对齐模式时对计数器执行写操作，否则将发生意想不到的结果。尤其是：
  — 如果写入计数器中的值大于自动重载值（TIMx_CNT>TIMx_ARR），计数方向不会更新。例如，如果计数器之前递增计数，则继续递增计数。
  — 如果向计数器写入 0 或 TIMx_ARR 的值，计数方向会更新，但不生成更新事件 UEV。
- 使用中心对齐模式最为保险的方法是：在启动计数器前通过软件生成更新（将 TIMx_EGR 寄存器中的 UG 位置 1），并且不要在计数器运行过程中对其执行写操作。

抖动模式

通过使能抖动模式，使用 TIMx_CR1 寄存器中的 DITHEN 位，可以提高 PWM 模式的有效分辨率。这既适用于 CCR（用于增加占空比分辨率），也适用于 ARR（用于增加 PWM 频率分辨率）。

工作原理是在 16 个连续的 PWM 周期中，以预定义的模式使实际的 CCR（或 ARR）值略有变化（增加或减少一个定时器时钟周期）。这使得占空比或 PWM 期间的平均值提高了 16 倍分辨率。下图展示了应用于 4 个连续 PWM 周期的抖动原理。

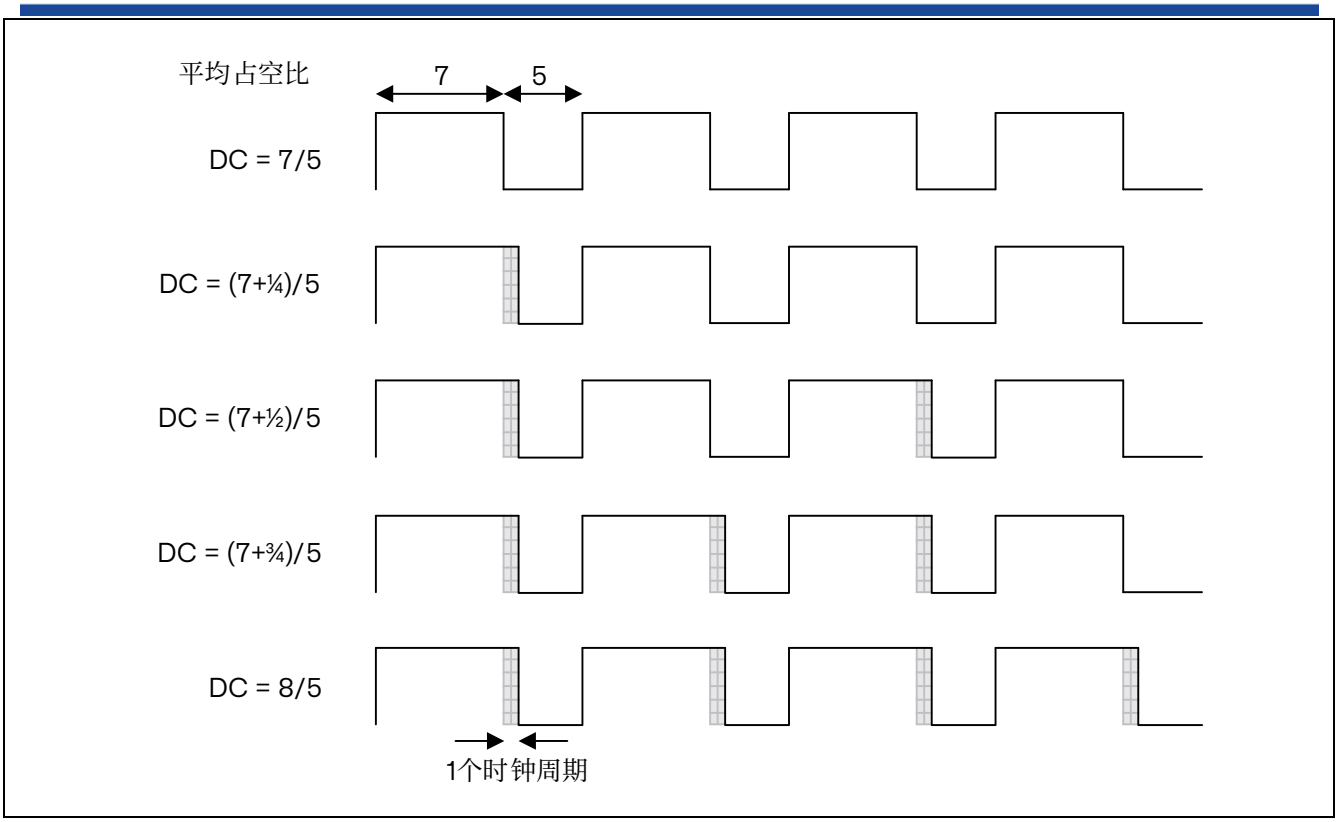


图 19.31 抖动原理

当抖动模式被使能时，寄存器编码将如下所示进行变化（参见下图示例）

- 4 个最低有效位（LSB）用于编码增强的分辨率部分（小数部分）
- 最高有效位（MSB）左移至 19:4 位，用于编码基本值

*注：如果设置清除*DITHEN *位，*ARR *和*CCR *值将自动更新。（例如，如果*ARR=0x05 *且*DITHEN=0，*它将更新为*ARR=0x50 *且*DITHEN=1*）。*

1. *必须重置*CEN *和*ARPE *位*
2. ARR[3:0]*位必须重置*
3. DITHEN *位必须重置*
4. CCIF *标志必须清除*
5. *可以设置*CEN *位（最终设置*ARPE=1*）。*

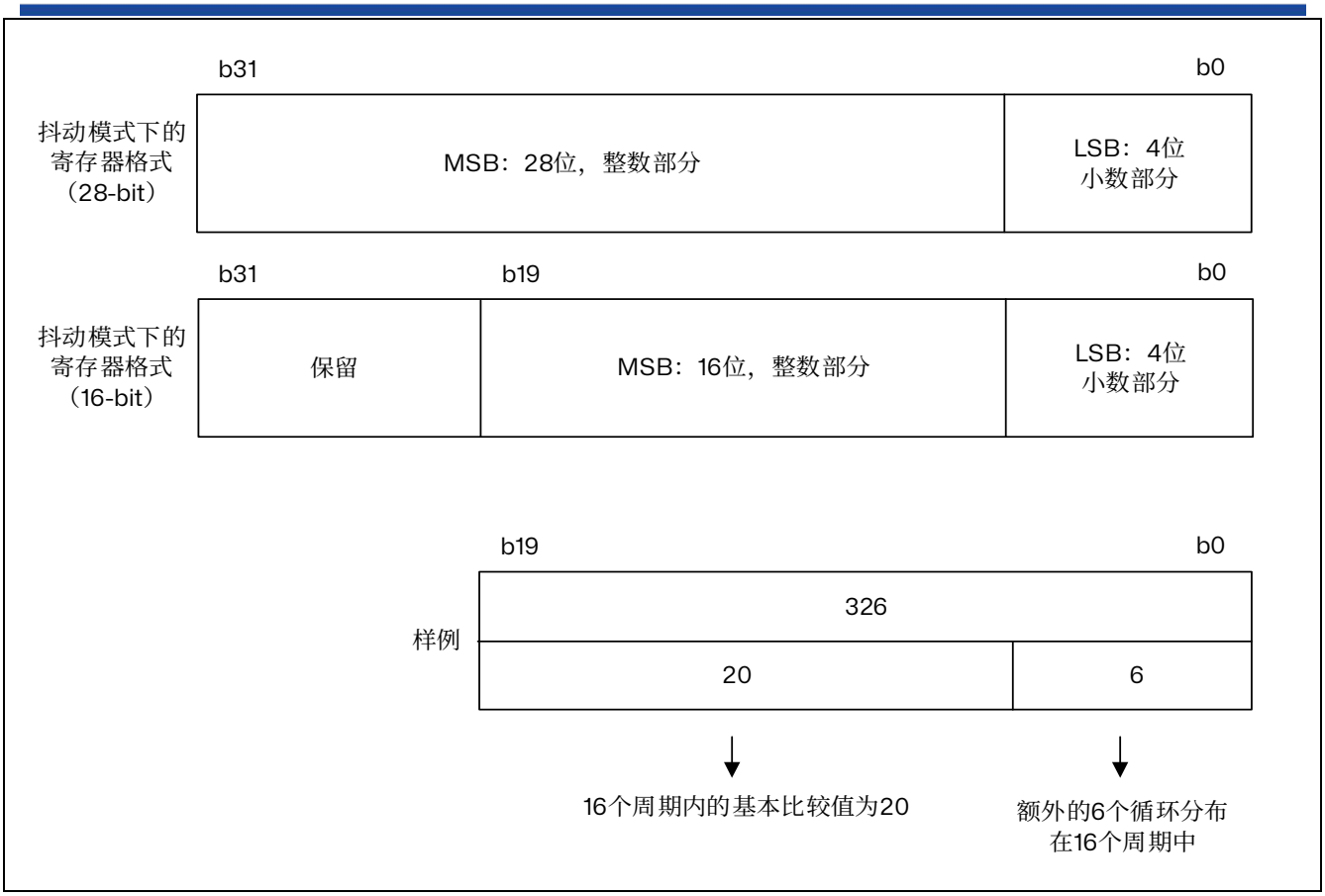


图 19.32 抖动模式下的数据格式和寄存器编码

最小频率由以下公式给出：

$$\text{分辨率} = \frac{\text{FTim}}{\text{Fpwm}} \Rightarrow \text{FpwmMin} = \frac{\text{FTim}}{\text{最大分辨率}}$$

$$\text{禁用抖动模式：FpwmMin} = \frac{\text{FTim}}{65536}$$

$$\text{使能抖动模式（16 位定时器）：FpwmMin} = \frac{\text{FTim}}{65535 + \frac{15}{16}}$$

$$\text{使能抖动模式（28 位定时器）：FpwmMin} = \frac{\text{FTim}}{268435454 + \frac{15}{16}}$$

*注意：对于16 位定时器，在抖动模式下，*TIMx_ARR *和*TIMx_CCRy *的最大值限制为*0xFFFEF *（对应于整数部分的*65534 *和抖动部分的*15*）。*

*对于28 位定时器，在抖动模式下，*TIMx_ARR *和*TIMx_CCRy *的最大值限制为*0xFFFFFFEF *（对应于整数部分的*264435454 *和抖动部分的*15*）。*

如下两张图所示，抖动模式可以增加 PWM 分辨率。

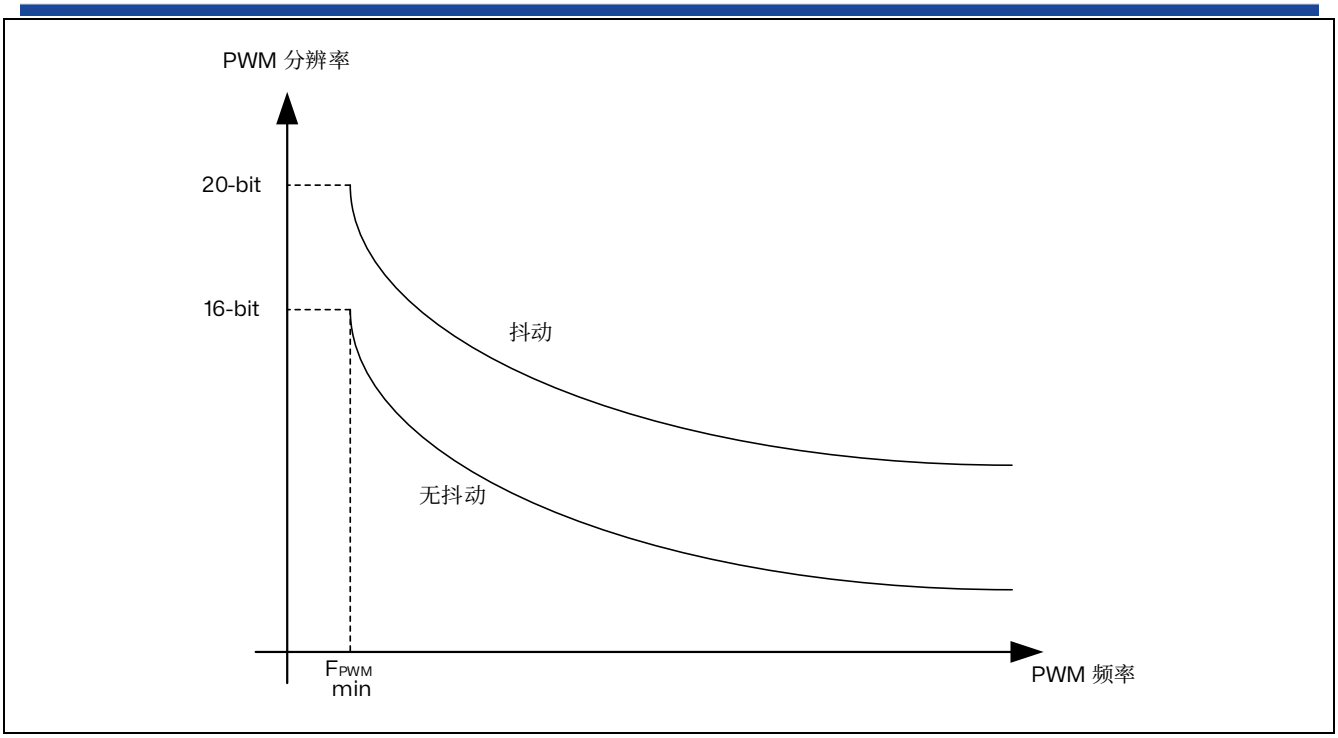


图 19.33 PWM 分辨率 vs 频率（16 位模式）

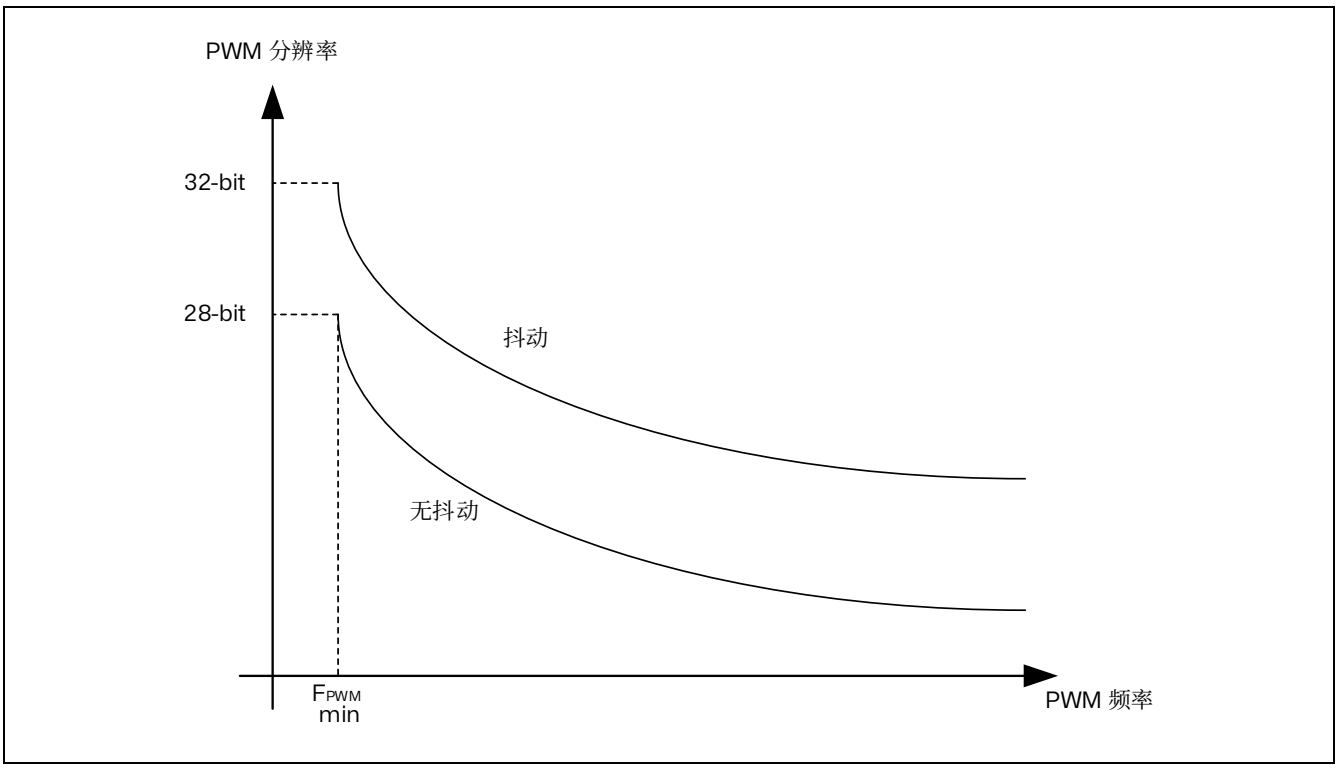


图 19.34 PWM 分辨率 vs 频率（28 位模式）

如下图所示，占空比和/或周期的变化分散在 16 个连续周期。

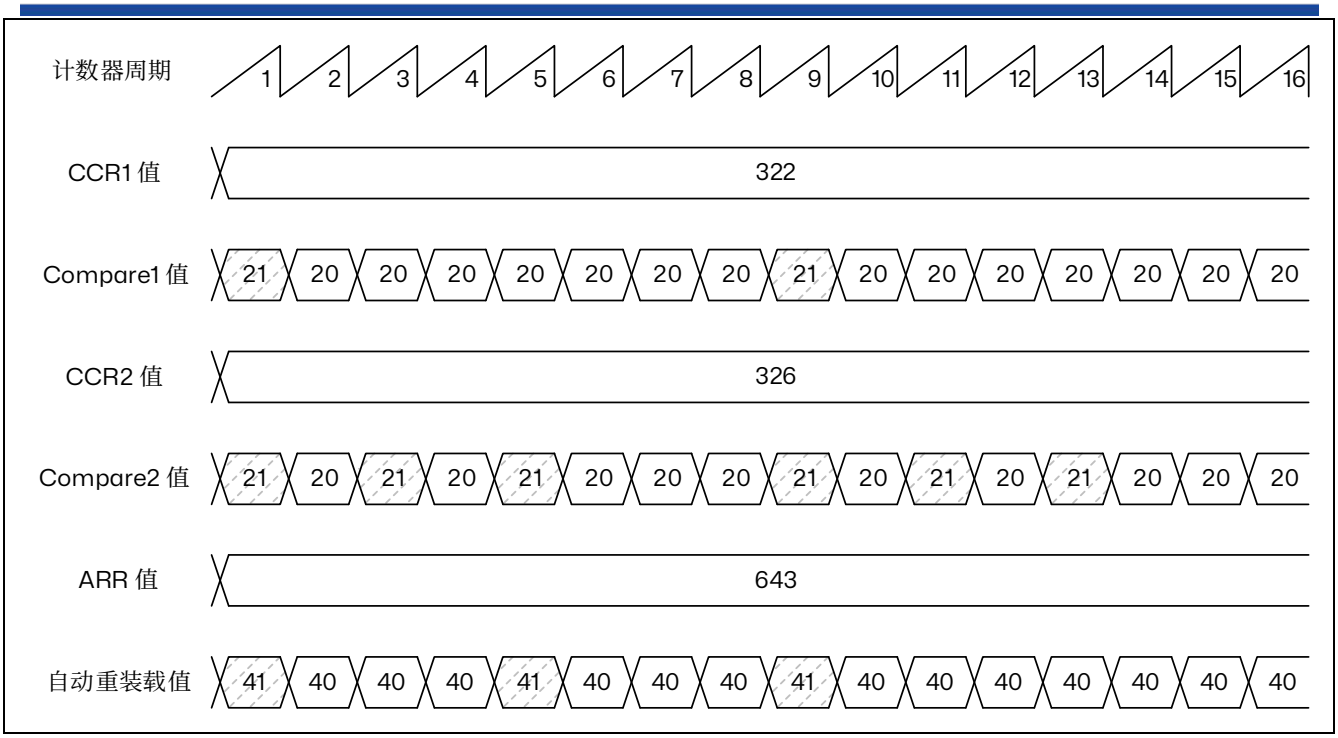


图 19.35 PWM 抖动模式

自动重新加载和比较值的增量按照下表中描述的特定模式分布。抖动序列是为了使增量分布尽可能均匀并最大限度地减少整体波动。

表 19.8 CCR 和 ARR 寄存器的变化抖动模式

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

在中心对齐 PWM 模式下（TIMx_CR1 寄存器中的 CMS 位不为“00”），也提供了抖动模式。在这种

模式下，抖动模式考虑到递增递减计数阶段，应用于 8 个连续的 PWM 周期，如下图所示。

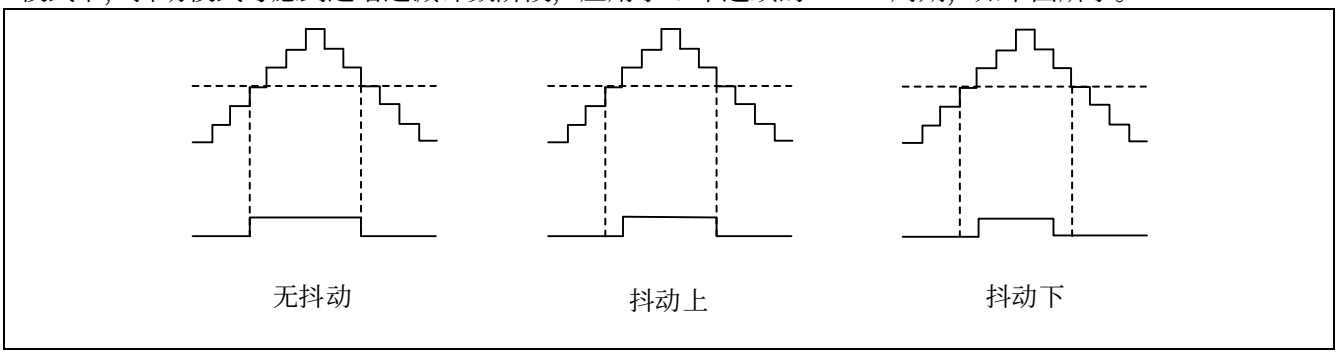


图 19.36 中心对齐 PWM 模式下抖动对占空比的影响

下表显示了如何在中心对齐 PWM 模式下添加抖动模式。

表 19.9 CCR 寄存器在中心对齐 PWM 模式下的变化抖动模式

LSB 值	PWM 周期															
	1		2		3		4		5		6		7		8	
	上	下	上	下	上	下	上	下	上	下	上	下	上	下	上	下
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

19.4.12 组合 PWM 模式

组合 PWM 模式允许生成两个边缘对齐或中心对齐的 PWM 信号，各自脉冲之间的可编程延迟和相位偏移。虽然频率由 TIMx_ARR 寄存器的值决定，但占空比和延迟由两个 TIMx_CCRx 寄存器确定。结果信号 tim_ocxrefc 由两个参考 PWM 的 OR 或 AND 逻辑组合而成：

— tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制

通过在 TIMx_CCMRx 寄存器的 OCxM 位中写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2），可以在两个通道上独立选组合 PWM 模式（每个 CCR 寄存器对一个 tim_ocx 输出）。

当某个通道用作组合 PWM 通道时，其互补通道必须配置为相反的 PWM 模式（例如，一个通道为组

合 PWM 模式 1，另一个通道为组合 PWM 模式 2）。

*注意：为了兼容性原因，*OCxM[3:0]*位域被分为两个部分，最高有效位与最低* 3 *位不连续。*

下图表示使用组合 PWM 模式生成信号的示例，该示例的配置如下：

— 通道 1 配置为组合 PWM 模式 2。

— 通道 2 配置为 PWM 模式 1。

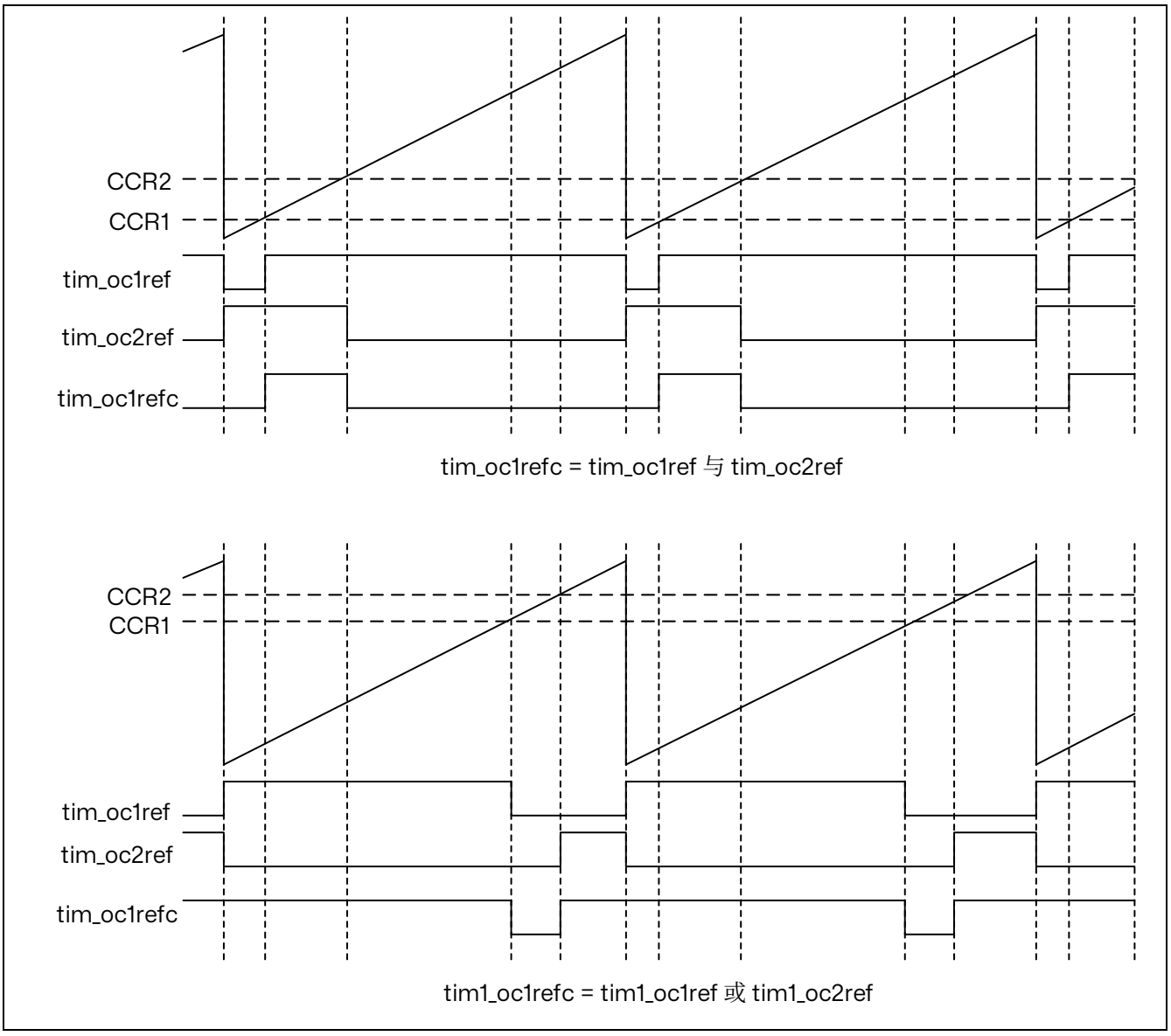


图 19.37 通道 1 和通道 3 的组合 PWM 模式

19.4.13 在外部事件时清除 tim_ocxref 信号

对于给定通道，在 tim_ocref_clr_int 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可使 tim_ocxref 信号变为低电平。tim_ocxref 信号将保持低电平，直到发生下一更新事件(UEV)。此功能仅能用于输出比较模式和 PWM 模式，而不适用于强制输出模式。

tim_ocref_clr_int 源取决于 OCREF 清除选择功能实施，请参考前序章节：TIM2/TIM3 实现。

如果实施了 OCREF 清除选择功能，则 tim_ocref_clr_int 为 tim_ocref_clr 输入。tim_ocref_clr 输入可以是使用中 OCRSEL[1:0]位域，从多个 tim_ocref_clr[3:0]输入中选出 TIMx_AF2 寄存器，如下图所示。

当配置组合 PWM 模式，对于给定通道，在 tim_ocref_clr_int 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可使组合结果 tim_ocxrefc 信号变为低电平。tim_ocxrefc 信号将保持低电平，直到发生下一更新事件（UEV）。

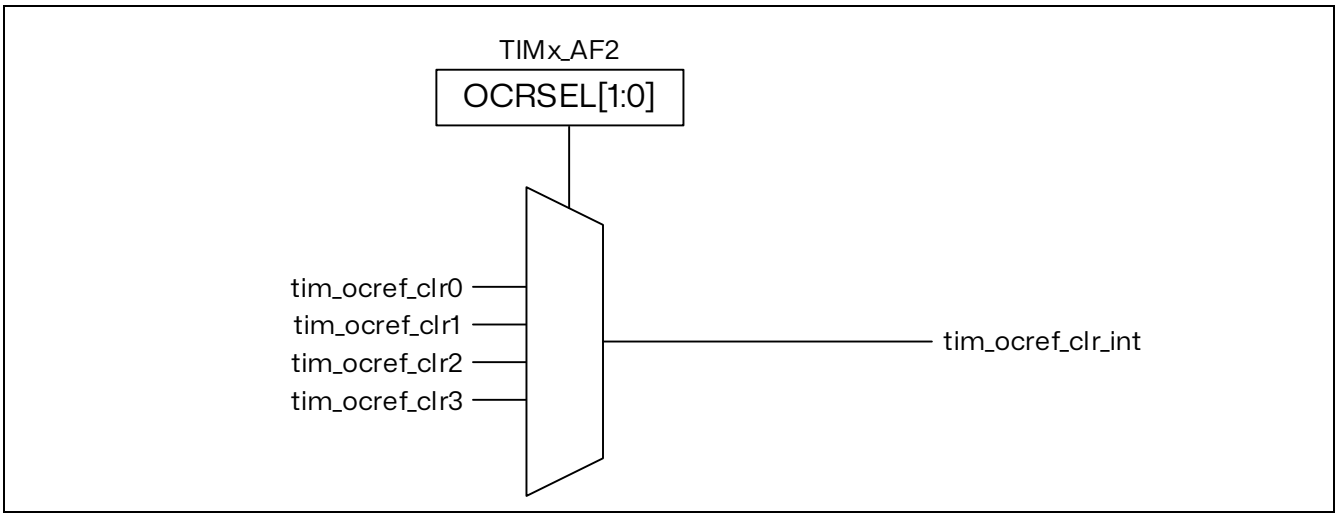


图 19.38 tim_ocref_clr 输入选择复用器

19.4.14 单脉冲模式

单脉冲模式（OPM）是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- CNT < CCRx ≤ ARR（特别注意，0<CCRx）

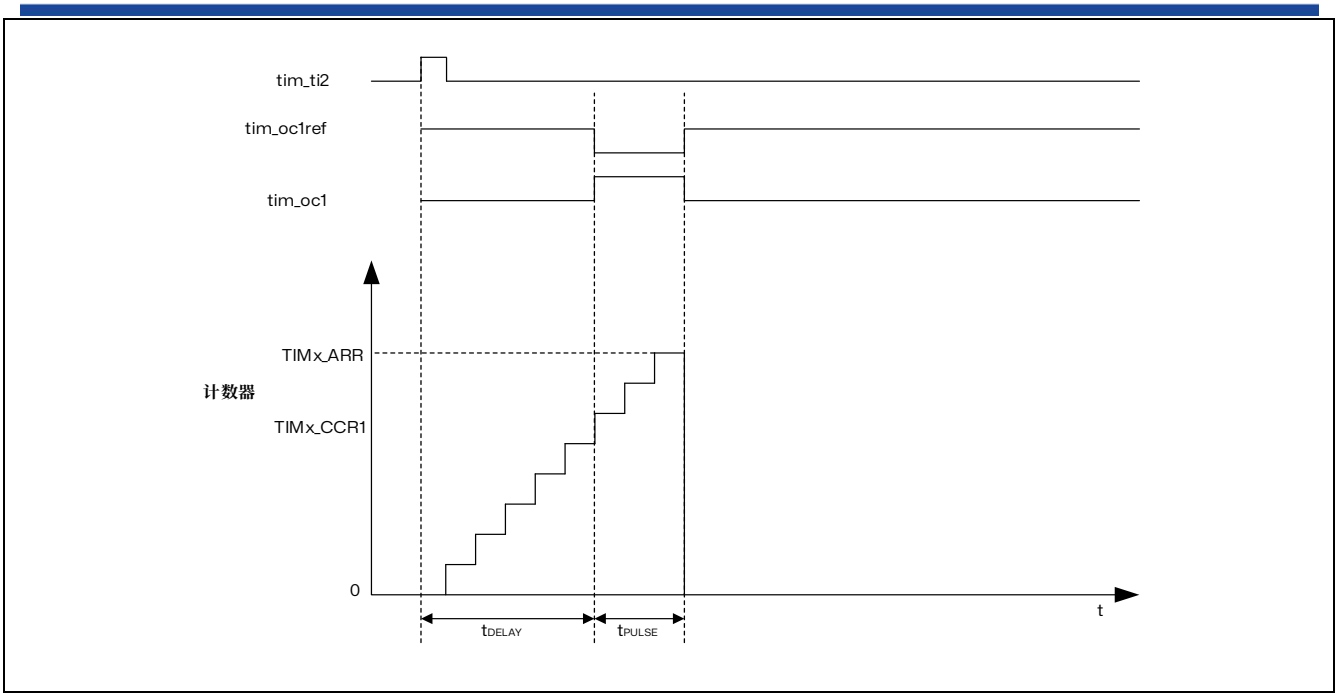


图 19.39 单脉冲模式的例子

例如，用户希望达到这样的效果：在 tim_ti2 输入引脚检测到正沿时，经过 t_{DELAY} 的延迟，在 tim_oc1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 tim_ti2fp2 作为触发 1:

1. 通过在 TIMx_TISEL 寄存器中使用 TI2SEL[2:0]位选择合适的 tim_ti2_in[15:0]源（内部或外部）。
2. 在 TIMx_CCMR1 寄存器中写入 CC2S=01，将 tim_ti2fp2 映射到 tim_ti2。
3. 在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=“0”，使 tim_ti2fp2 能够检测上升沿。
4. 在 TIMx_SMCR 寄存器中写入 TS=0110，将 tim_ti2fp2 配置为从模式控制器的触发（tim_trgi)。
5. 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），使用 tim_ti2fp2 启动计数器。

OPM 波形通过对比较寄存器执行写操作来定义（考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIMx_CCR1 寄存器的值定义。
- t_{PULSE} 由自动重载值与比较值（TIMx_ARR - TIMx_CCR1）之差来定义。
- 假设希望产生这样的波形：信号在发生比较匹配时从“0”变为“1”，在计数器达到自动重载值时由“1”变为“0”。为此，应在 TIMx_CCMR1 寄存器中写入 OC1M=111，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 tim_ti2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应为低。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特例：tim_ocx 快速使能：

在单脉冲模式下，tim_tix 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟

（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 tim_ocxref（和 tim_ocx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用

19.4.15 可重触发单脉冲模式

此模式允许计数器在接收到刺激后开始计数，并生成一个具有可编程长度的脉冲，但与上一节中描述的非重触发单脉冲模式存在以下差异：

- 只要触发器发生，脉冲就会立即开始（没有可编程的延迟）
- 如果在先前触发的脉冲完成之前发生新的触发，则脉冲会被延长

定时器必须处于从模式，TIMx_SMCR 寄存器的 SMS[3:0]位设置为“1000”（组合重置+触发模式），OCxM[3:0]位设置为“1000”或“1001”，用于可重触发 OPM 模式 1 或 2。

如果定时器配置为递增计数模式，则相应的 CCRx 必须设置为 0（ARR 寄存器设置脉冲长度）。如果定时器配置为递减计数模式，CCRx 必须大于或等于 ARR。

*注：在可重复触发单脉冲模式中，*CCxIF *标志不重要。*

OCxM[[3:0]*和*SMS[3:0]*位域由于兼容性的原因被分成两个部分，最高有效位与最低*3 *位不是连续的。*

OCxM *这种模式不能与中央对齐*PWM *模式一起使用。*TIMx_CR1 *寄存器中的*CMS[1:0]*必须设置为*00。

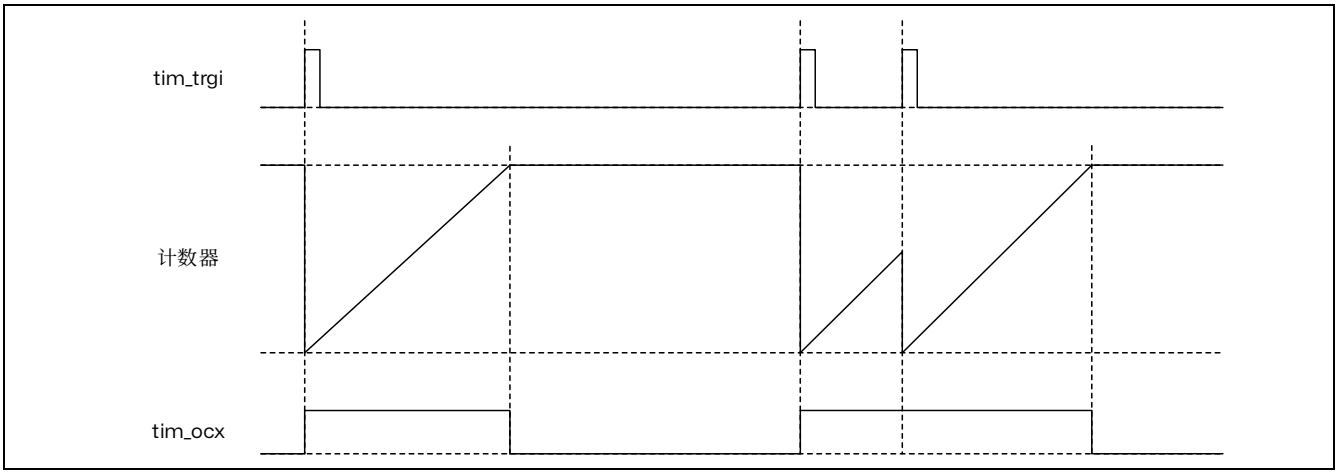


图 19.40 可重触发单脉冲模式

19.4.16 UIF 位重映射

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许以原子方式读取计数器的值和一个潜在的由 UIFCPY 标志指示的翻转条件。在特定情况下，它可以避免由背景任务（读取计数器）和中断（更新中断）之间的处理引起的竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间没有延迟。

19.4.17 定时器和外部触发的同步

TIMx 定时器以下列模式与外部触发实现同步：复位模式、门控模式、触发模式、复位+触发和门控+复位模式。

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 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIMx_ARR，TIMx_CCRx）都被更新了。

在以下的例子中，tim_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 tim_ti1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 tim_ti1 出现一个上升沿；当 tim_ti1 上升时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIMx_SR 寄存器中的 TIF 位）被设置，中断请求可以被发送（取决于 TIMx_DIER 寄存器中的 TIE 位）。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。

在 tim_ti1 上升沿和计数器的实际复位之间的延时取决于 tim_ti1 输入端的重同步电路。

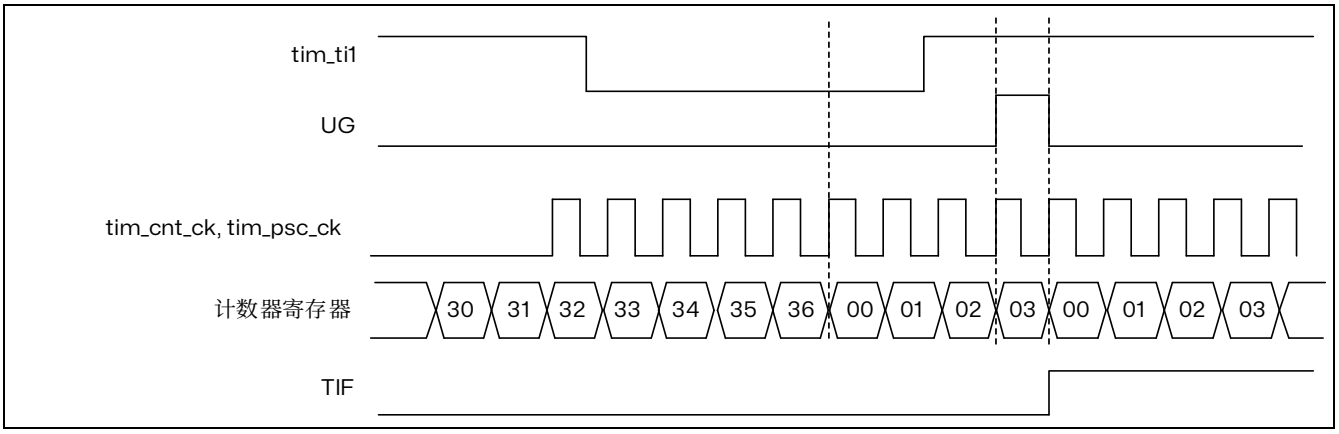


图 19.41 复位模式下的控制电路

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 tim_ti1 为低时向上计数：

- 配置通道 1 以检测 tim_ti1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）。

只要 tim_ti1 为低，计数器开始依据内部时钟计数，一旦 tim_ti1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

tim_ti1 上升沿和计数器实际停止之间的延时取决于 tim_ti1 输入端的重同步电路。

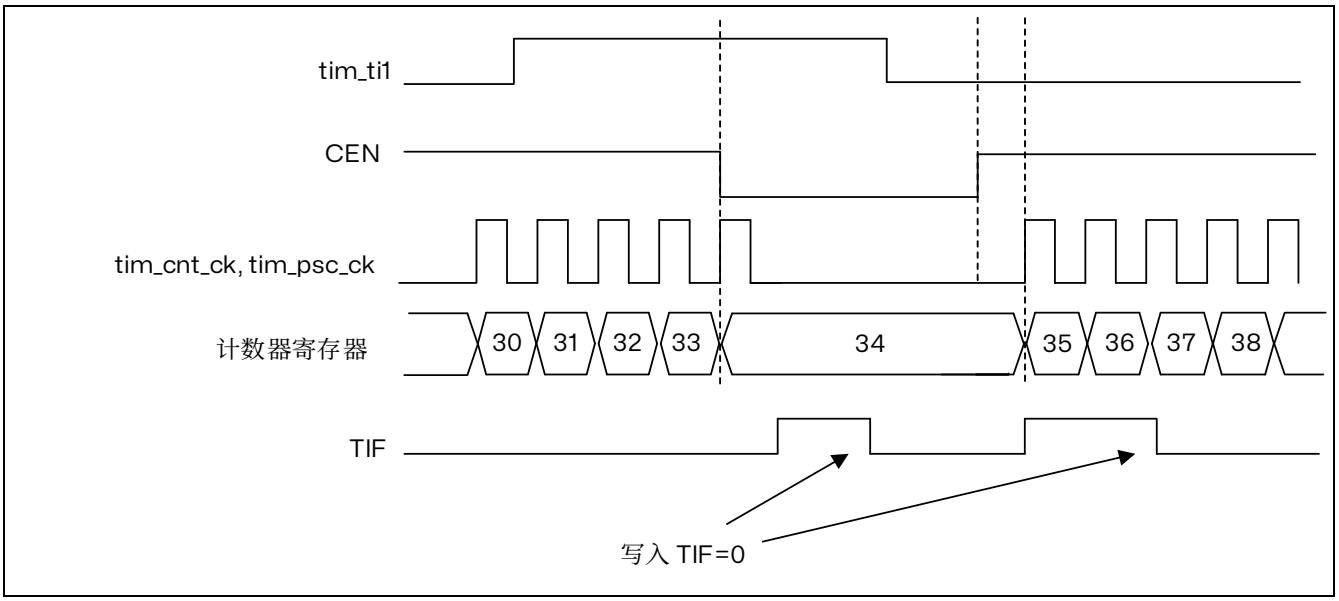


图 19.42 门控模式下的控制电路

注：由于门控模式作用于电平而非边沿，因此在门控模式下，“CCxP=CCxNP=1”（同时检测上升沿和下降沿）不发挥任何作用。

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 tim_ti2 输入的上升沿开始向上计数：

- 配置通道 2 检测 tim_ti2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 和 CC2NP=0 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=0110，选择 tim_ti2 作为输入源。

当 tim_ti2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

tim_ti2 上升沿和计数器启动计数之间的延时，取决于 tim_ti2 输入端的重同步电路。

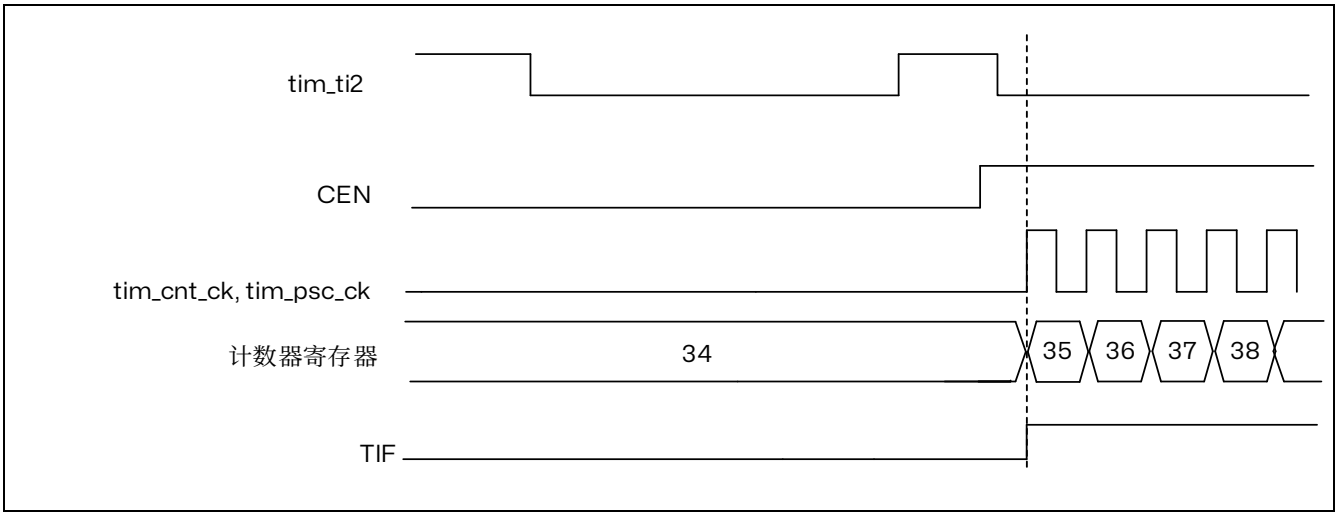


图 19.43 触发器模式下的控制电路

对实时更新编码器模式的从模式选择预装载

SMS[3:0]位可以预加载。这是通过在 TIMx_SMCR 寄存器中设置 SMSPE 使能位来实现的。从 SMS[3:0]预加载到有效值的触发器是计数器溢出时发生的更新事件（UEV）。

从模式 - 组合重置 + 触发模式

在这种情况下，选择的触发输入（tim_trgi）的上升沿重新初始化计数器，生成寄存器的更新，并启动计数器。

该模式用于单脉冲模式。

从模式 - 组合重置 + 门控模式

触发输入（tim_trgi）高时使能计数器。一旦触发变低，计数器停止并复位。计数器的开始和停止都是受控的。

该模式允许检测超出范围的 PWM 信号（占空比超过最大期望值）。

19.4.18 定时器同步

TIMx 定时器从内部连接在一起，以实现定时器同步或级联。当某个定时器配置为主模式时，可对另一个配置为从模式的定时器的计数器执行复位、启动、停止操作或为其提供时钟。下图显示了触发选择和主模式选择模块的概况。

下图简要介绍了触发选择和主模式选择框图。

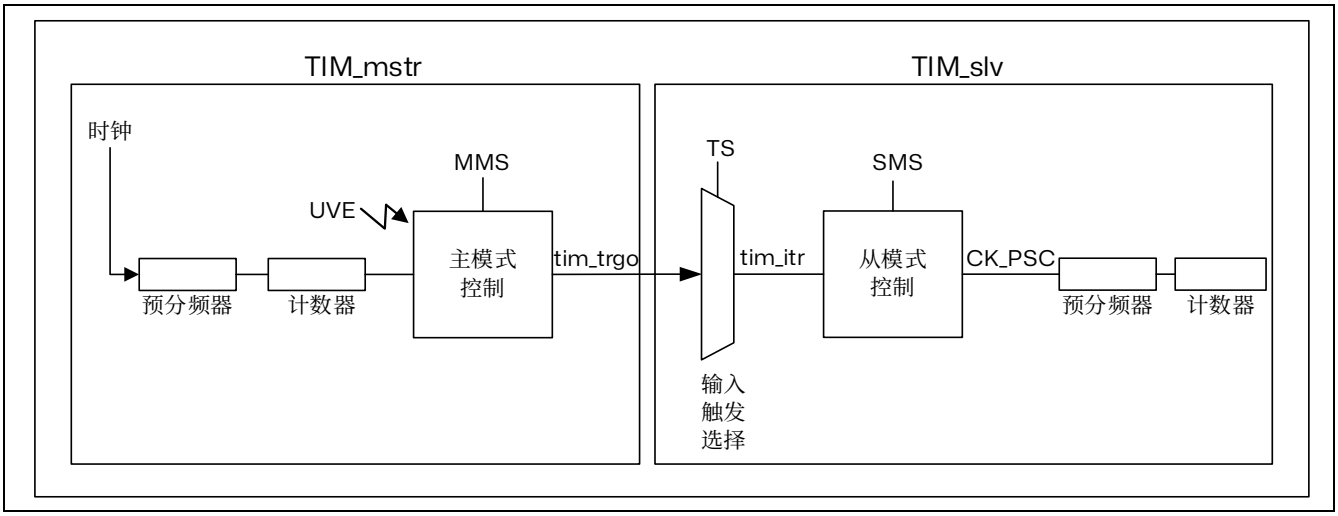


图 19.44 主/从定时器的例子

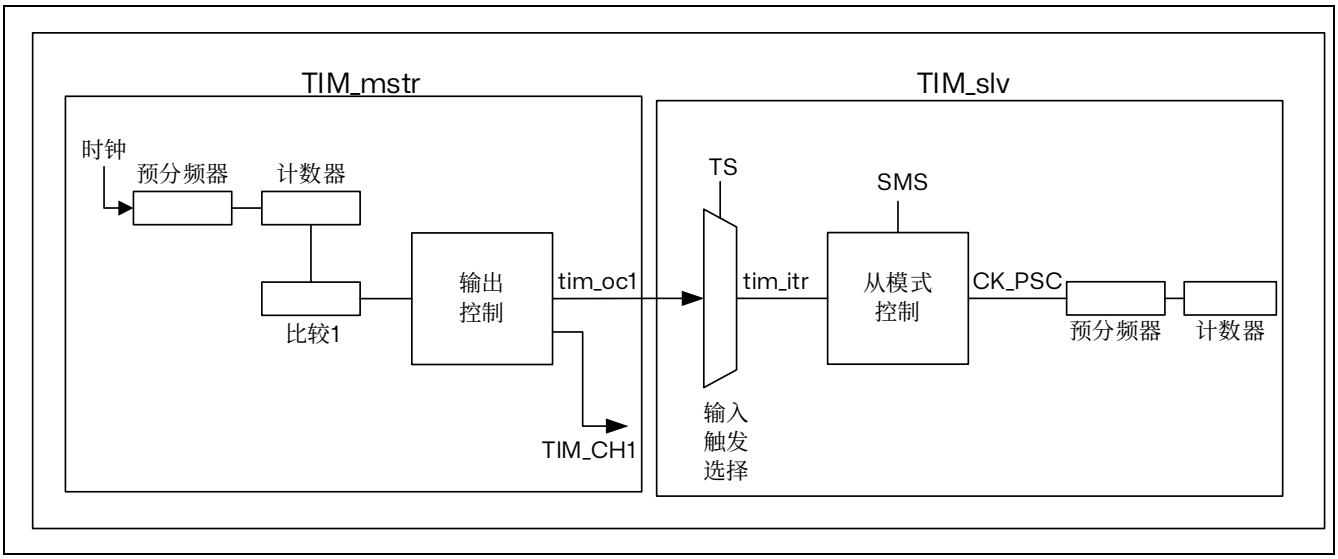


图 19.45 仅定时器通道 1 主/从连接示例

*注：只有一个通道的计时器（参见上图）不具有主模式。然而，*tim_oc1 *输出信号可以作为从属定时器的触发信号（参见*TIMx *内部触发连接表：*TIM2/TIM3 *引脚和内部信号）。*
*必须将*tim_oc1 *信号脉冲宽度编程为至少为目的地定时器的*2 *个时钟周期，以确保从属定时器检测到触发信号。例如，如果目的地定时器*tim_ker_ck *时钟比源定时器慢*4 *倍，则*OC1 *脉冲宽度必须是*8 *个时钟周期。*

使用一个定时器作为另一个定时器的预分频器

例如，可以将 TIM_mstr 配置为 TIM_slv 的预分频器。请参见上图。为此：

1. 将 TIM_mstr 配置为主模式，以便每次发生更新事件 UEV 时都输出一个周期性触发信号。如果在 TIM_mstr_CR2 寄存器中写入 MMS=010，则每次生成更新事件时，tim_trgo 都会输出一个上升沿。
2. 要将 TIM_mstr 的 tim_trgo 输出连接到 TIM_slv，必须将 TIM_slv 配置为从模式，使用 ITR0 作为内部触发。通过 TIM_slv_SMCR 寄存器中的 TS 位（写入 TS=000）可对此进行选择。
3. 然后将从模式控制器设为外部时钟模式 1（在 TIM_slv_SMCR 寄存器中写入 SMS=111）。这样一来，TIM_slv 的时钟将由 TIM_mstr 周期性触发信号的上升沿（与 TIM_mstr 的计数器上溢对应）提供。
4. 最后必须通过将这两个定时器的相应 CEN 位（TIMx_CR1 寄存器）置 1 同时使能二者。

注：如果选择 TIM_mstr 的 tim_ocx 信号作为触发输出（MMS=1xx），该信号的上升沿将用于驱动 TIM_slv 的计数器。

使用一个定时器使能另一个定时器

本例中通过 TIM_mstr 的输出比较 1 来使能 TIM_slv。相关连接图，参考下图。仅当 TIM_mstr 的 tim_oc1ref 为高电平时，TIM_slv 才根据分频后的内部时钟进行计数。两个计数器的时钟频率都基于 tim_ker_ck 通过预分频器执行 3 分频（$f_{tim_cnt_ck} = f_{tim_ker_ck}/3$）

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号（tim_oc1ref）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref 波形（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
6. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

注：计数器 2 的时钟与计数器 1 不同步，此模式仅影响 TIM_slv 的计数器使能信号。

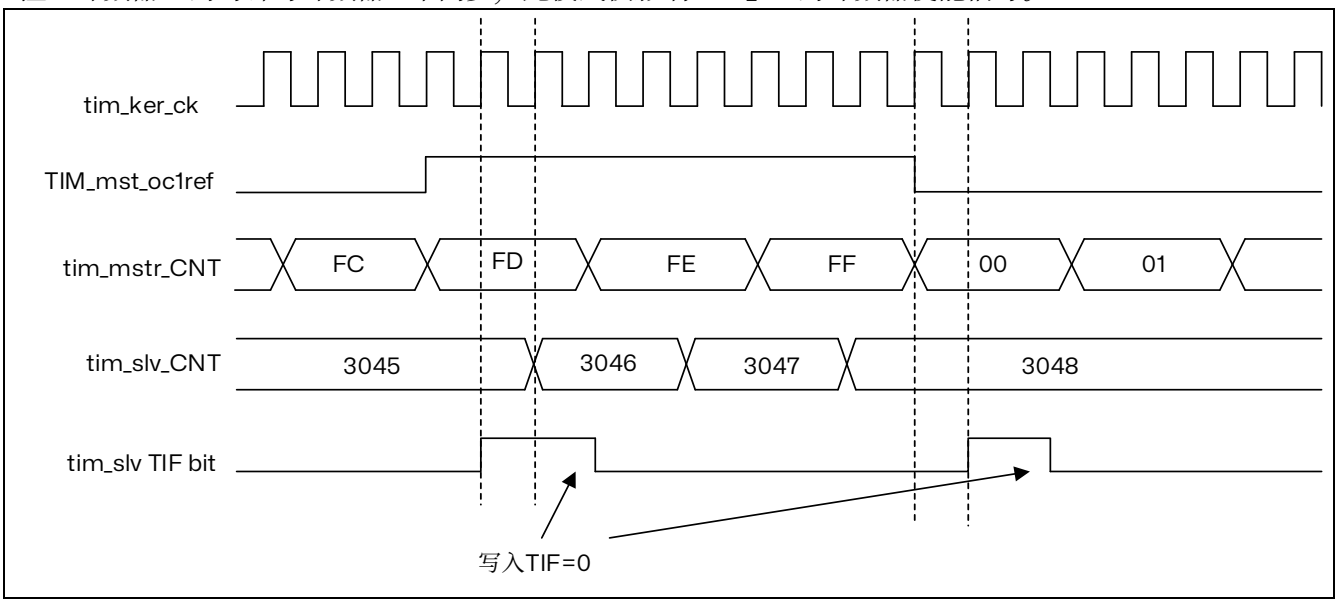


图 19.46 TIM_mstr 的 tim_oc1ref 控制 TIM_slv

在上图示例中，TIM_slv 的计数器和预分频器在启动前未进行初始化。因此从各自的当前值开始计数。启动 TIM_mstr 之前，通过复位这两个定时器可以从指定值开始计数。这样便可以在定时器计数器中写入所需的任意值。两个定时器都可通过软件使用 TIMx_EGR 寄存器中的 UG 位轻松复位。

在下一示例中，TIM_mstr 与 TIM_slv 同步。TIM_mstr 为主模式，从 0 开始计数。TIM_slv 为从模式，从 0xE7 开始计数。两个定时器的预分频比相同。在 TIM_mstr_CR1 寄存器中通过向 CEN 位写入“0”来禁止 TIM_mstr 时，TIM_slv 将停止：

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号（tim_oc1ref）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 UG 位（TIM_mstr_EGR 寄存器）写入“1”复位 TIM_mstr。
6. 通过向 UG 位（TIM_slv_EGR 寄存器）写入“1”复位 TIM_slv。
7. 通过在 TIM_slv 的计数器（TIM_slv_CNT）中写入“0xE7”使 TIM_slv 初始化为 0xE7。
8. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
9. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。
10. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“0”停止 TIM_mstr。

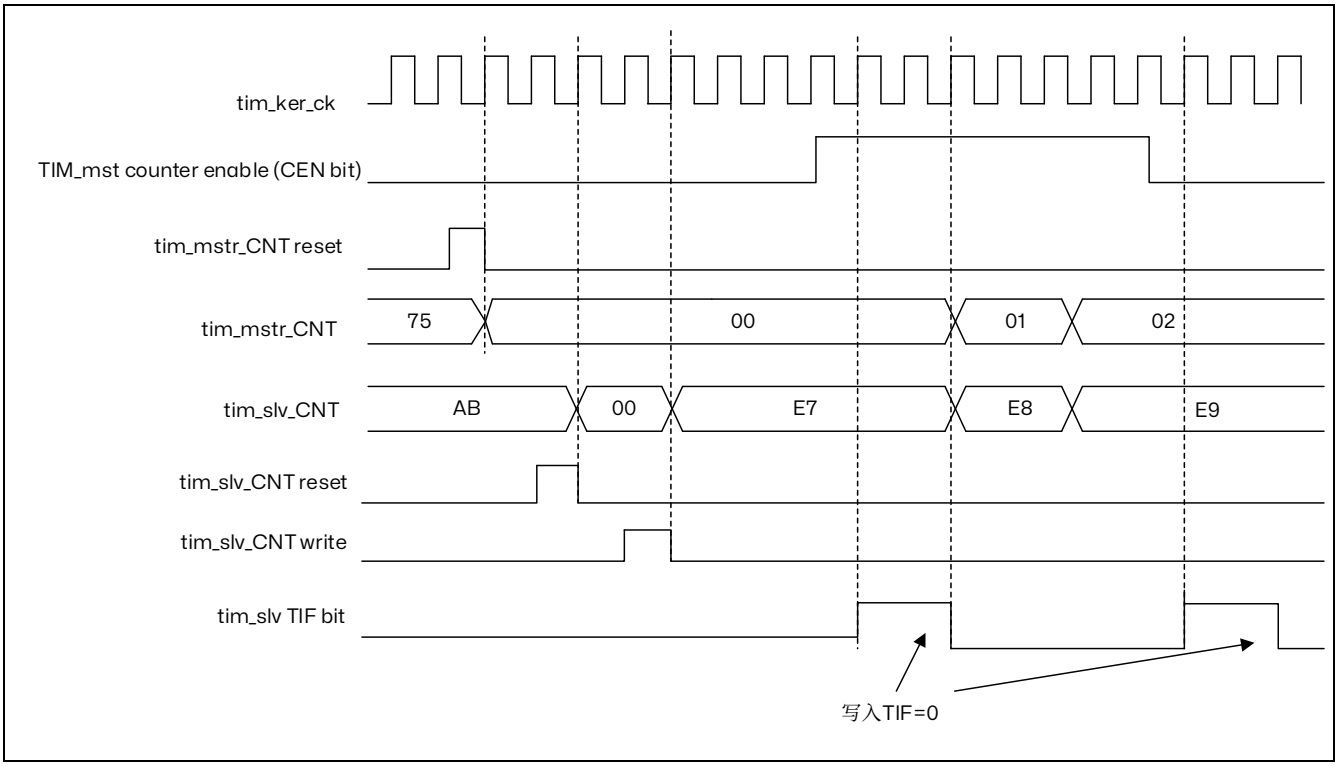


图 19.47 通过使能 TIM_mstr 可以控制 TIM_slv

使用一个定时器去启动另一个定时器

本例中使用 TIM_mstr 的更新事件使能 TIM_slv。相关连接图，请参见下图。只要 TIM_mstr 生成更新事件，TIM_slv 便根据分频后的内部时钟从当前值（可以不为 0）开始计数。TIM_slv 收到触发信号时，其 CEN 位自动置 1，并且计数器开始计数，直到向 TIM_slv_CR1 寄存器的 CEN 位写入“0”后停止计数。两个计数器的时钟频率都基于 CK_INT 通过预分频器执行 3 分频（$f_{tim_cnt_ck} = f_{tim_ker_ck}/3$）。

1. 将 TIM_mstr 配置为主模式，发送其更新事件（UEV）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=010）。
2. 配置 TIM_mstr 的周期（TIM_mstr_ARR 寄存器）。

3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。
4. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。
5. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

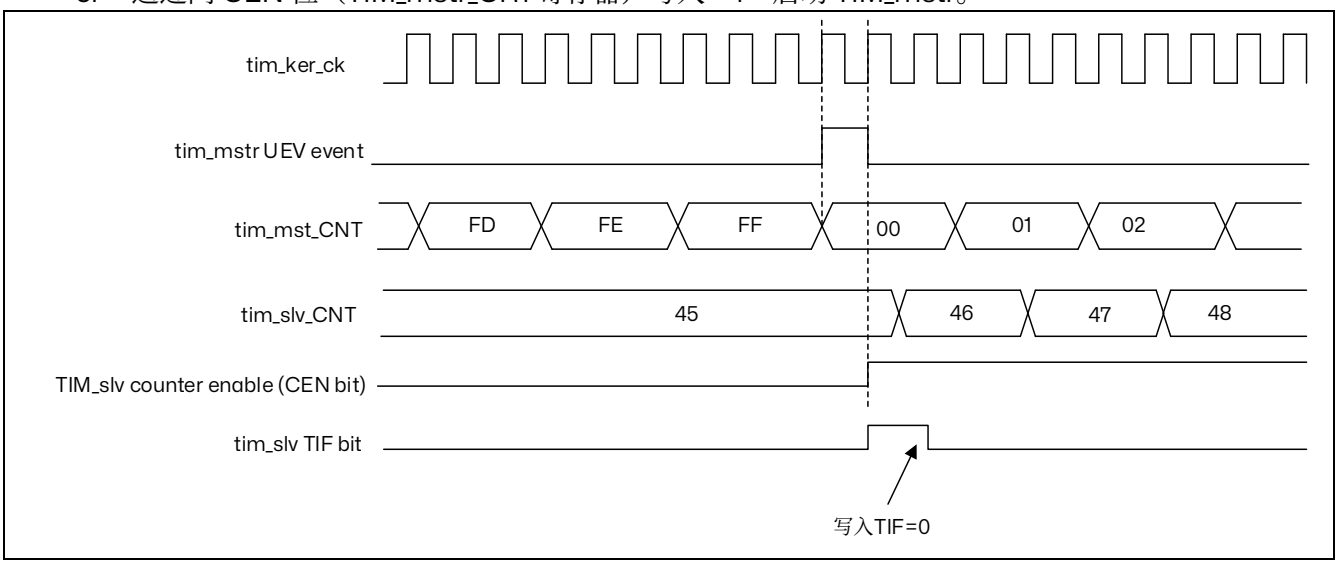


图 19.48 使用 TIM_mstr 的更新触发 TIM_slv

如上述示例所示，用户可以在开始计数之前初始化两个计数器。下图显示了与上图具有相同配置，只不过处于触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）而非门控模式的计数行为。

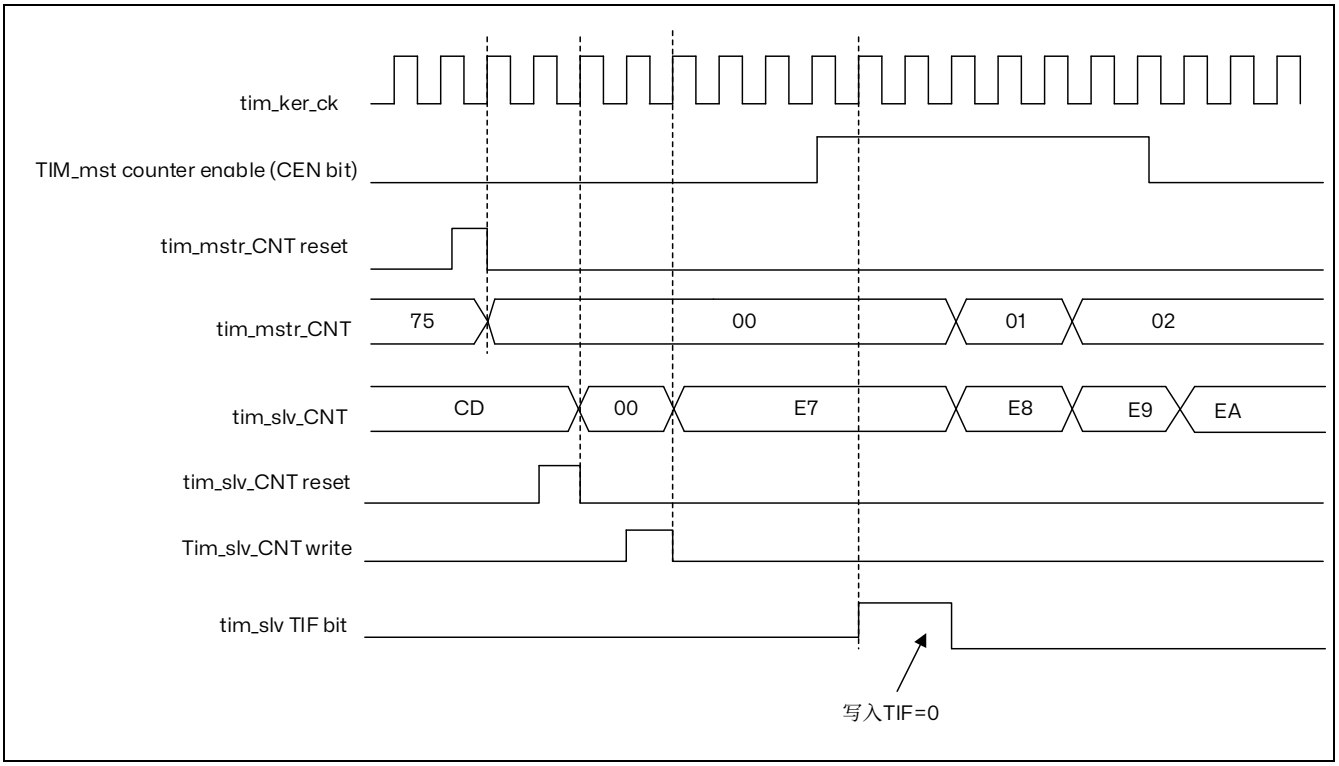


图 19.49 利用 TIM_mstr 的使能触发 TIM_slv

使用一个外部触发同步地启动 2 个定时器

本例中，TIM_mstr 的 tim_ti1 输入出现上升沿时使能 TIM_mstr，使能 TIM_mstr 的同时使能

TIM_slv。要确保两个计数器对齐，TIM_mstr 必须配置为主/从模式（对应的 tim_ti1 为从，对应 TIM_slv 为主）：

1. 将 TIM_mstr 配置为主模式，发送其使能信号作为触发输出（TIM_mstr_CR2 寄存器中 MMS=001）。
2. 将 TIM_mstr 配置为从模式以接收来自 tim_ti1 的输入触发（TIM_mstr_SMCR 寄存器中的 TS=0100）。
3. 将 TIM_mstr 配置为触发模式（TIM_mstrSMCR 寄存器中的 SMS=110）。
4. 通过写入 MSM=1（TIM_mstr_SMCR 寄存器）将 TIM_mstr 配置为主/从模式。
5. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0000）。
6. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。

当 tim_ti1（TIM_mstr）出现上升沿时，两个计数器开始根据内部时钟同步计数，并且两个 TIF 标志都置 1。

注：本例中，两个定时器都在启动之前进行了初始化（通过将各自的UG 位置1）。两个计数器都从 0 开始计数，但可以通过对任意一个计数器寄存器（TIMx_CNT）进行写操作，在二者之间轻松插入一个偏移量。可注意到主/从模式在TIM_mstr 的CNT_EN 与CK_PSC 之间产生了延迟。

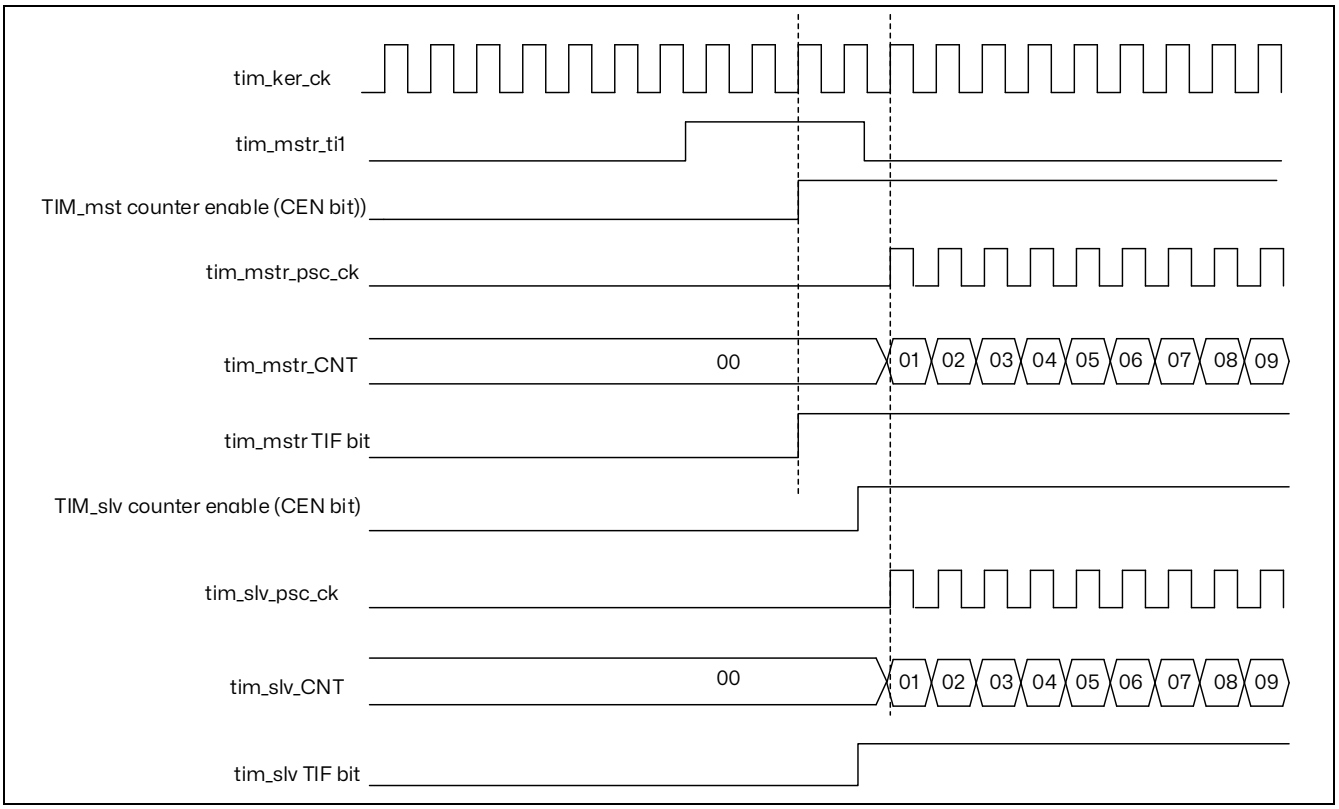


图 19.50 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv

注：接收tim_trgo 信号的从外设（定时器，ADC 等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能实时更改。

19.4.19 ADC 触发

定时器可以用各种内部信号（例如重置、使能或比较事件）生成 ADC 触发事件。

注意：接收tim_trgo 信号的从外设（例如定时器，ADC）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能实时更改。

19.4.20 调试模式

当微控制器进入调试模式（内核停止）时，TIMx 计数器可以继续正常工作或停止。
在调试模式下，每个定时器的行为可以通过 Debug 支持（DBG）模块的专用配置位进行编程。
有关更多详细信息，请参阅调试部分。

19.5 TIM2/TIM3 低功耗模式

表 19.10 低功耗模式对 TIM2/TIM3 的影响

模式	描述
睡眠	无影响，外设是运行的。中断会导致设备退出睡眠模式。
停机	定时器操作被停止并保留其寄存器内容。无法生成中断。
待机	定时器被断电并且退出待机模式后必须重新初始化。

19.6 TIM2/TIM3 中断

TIM2/TIM3 可以生成多个中断，如下表所示。

表 19.11 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	从睡眠模式退出	从停机和待机退出
TIM_UP	更新	UIF	UIE	UIF 写 0	是	否
TIM_CC	捕获/比较 1	CC1IF	CC1IE	CC1IF 写 0	是	否
	捕获/比较 2	CC2IF	CC2IE	CC2IF 写 0	是	否
TIM_TRG	触发	TIF	TIE	TIF 写 0	是	否

19.7 TIM2/TIM3 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
外设寄存器可以通过半字（16 位）或字（32 位）进行访问。

19.7.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 2，3）

偏移地址：0x000
复位值：0x0000

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

Bit 15:13 保留，必须保持为复位值。

Bit 12 **DITHEN**：使能抖动（Dithering enable）

0：抖动禁止

1：抖动使能

*注：*DITHEN *位只能在*CEN *位复位时修改*

Bit 11 **UIFREMAP**：UIF 状态位重映射（UIF status bit remapping）

0：无重映射。UIF 状态位未复制到 TIMx_CNT 寄存器位 31

1：重映射使能。UIF 状态位复制到 TIMx_CNT 寄存器位 31

Bit 10 保留，必须保持为复位值。

Bit 9:8 **CKD[1:0]**：时钟分频（Clock division）

此位域指示定时器时钟（tim_ker_ck）频率以及数字滤波器（tim_tix）所使用的采样时钟（t_{DTS}）之间的分频比

00：$t_{DTS} = t_{tim_ker_ck}$

01：$t_{DTS} = 2 \times t_{tim_ker_ck}$

10：$t_{DTS} = 4 \times t_{tim_ker_ck}$

11：保留，不要设置成此值

Bit 7 **ARPE**：自动重载预装载使能（Auto-reload preload enable）

0：TIMx_ARR 寄存器不进行缓冲；

1：TIMx_ARR 寄存器进行缓冲。

Bit 6:5 **CMS[1:0]**：中心对齐模式选择（Center-aligned mode selection）

00：边沿对齐模式。计数器根据方向位（DIR）递增计数或递减计数。

01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志都会置 1。

注：只要计数器处于使能状态 (CEN=1)，*就不得从边沿对齐模式切换为中心对齐模式。*

Bit 4 **DIR**：方向（Direction）

0：计数器递增计数；

1：计数器递减计数。

注：当定时器配置为中心对齐模式时，该位为只读状态。

Bit 3 **OPM**：单脉冲模式（One pulse mode）

0：计数器在发生更新事件时不会停止计数；

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

Bit 2 **URS**：更新请求源（Update request source）

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断。此类事件包括：

- 计数器上溢/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断。

Bit 1 **UDIS**：更新禁止（Update disable）

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新（UEV）事件可通过以下事件之一生成：

- 计数器上出/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

Bit 0 **CEN**：计数器使能（Counter enable）

0：禁止计数器；

1：使能计数器。

*注：只有事先通过软件将*CEN *位置*1，*才可以使用外部时钟和门控模式。而触发模式可通过硬件自动将*CEN *位置*1。

19.7.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 2，3）

偏移地址：0x004

复位值：0x0000 0000

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

Bit 31:7 保留，必须保持为复位值。

Bit 6:4 **MMS[2:0]：**主模式选择（Master mode selection）

这些位可选择主模式下将要发送到从定时器以实现同步的信息(tim_trgo)。这些位的组合如下：

000：**复位** – TIMx_EGR 寄存器中的 UG 位用作触发输出（tim_trgo)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。

001：**使能** – 计数器使能信号 CNT_EN 用作触发输出（tim_trgo)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号可由 CEN 控制位产生。当配置为门控模式时，也可由触发输入产生。当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明)。

010：**更新** – 选择更新事件作为触发输出（tim_trgo)。例如，主定时器可用作从定时器的预分频器。

011：**比较脉冲** – 一旦发生输入捕获或比较匹配事件，当 CC1IF 被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(tim_trgo)。

100：**比较** – tim_oc1refc 信号用作触发输出（tim_trgo）

101：**比较** – tim_oc2refc 信号用作触发输出（tim_trgo）

其他编码保留

注：在从主定时器接收事件之前，必须使能从机定时器或 ADC *的时钟，并且在从主定时器接收触发器时不能随意更改。*

注：若选 tim_ocxrefc *为* tim_trgo *输出，当* tim_ocref_clr_int *清零信号生效时，* tim_ocxrefc *信号将保持低电平，* tim_trgo *也不会有触发输出，直到发生下一更新事件（*UEV*）（特别注意，当* TIMx_RCR>0 *时，只有重复计数器达到零才会生成更新事件，波形才会恢复)。*

Bit 3:0 保留，必须保持为复位值。

19.7.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = 2，3）

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SMSPS	SMSPE	Reserved			TS[3]	Reserved			SMS[3]
						rw	rw				rw				rw

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

Bit 31:26 保留，必须保持为复位值。

Bit 25 **SMSPS**：SMS 预加载源（SMS preload source）

此位选择触发 SMS[3:0]位域从预加载到活动的事件来源：

0：由定时器的更新事件触发

1：由索引事件触发

Bit 24 **SMSPE**：SMS 预加载使能（SMS preload enable）

此位选择是否使能 SMS[3:0]位域的预加载：

0：SMS[3:0]位域不进行预加载

1：使能 SMS[3:0]预加载

Bit 23:21 保留，必须保持为复位值。

Bit 19:17 保留，必须保持为复位值。

Bit 15:8 保留，必须保持为复位值。

Bit 7 **MSM**：主/从模式（Master/slave mode）

0：不执行任何操作；

1：当前定时器的触发输入事件(tim_trgi)的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

Bit 20,6,5,4 **TS[3:0]**：触发选择（Trigger selection）

此位域可选择将要用于同步计数器的触发输入。

0000：内部触发 0（tim_itr0）

0001：内部触发 1（tim_itr1）

0010：内部触发 2（tim_itr2）

0011：内部触发 3（tim_itr3）

0100：tim_ti1 边沿检测器（tim_ti1f_ed）

0101：滤波后的定时器输入 1（tim_ti1fp1）

0110：滤波后的定时器输入 2（tim_ti2fp2）

0111：保留

1000：内部触发器 4 (tim_itr4)

1001：内部触发器 5 (tim_itr5)

其他：保留

有关定时器 ITRx 含义的详细信息，请参见章节：TIM2/TIM3 引脚和内部信号

注：这些位只能在未使用的情况下（例如，SMS=000 时）进行更改，以避免转换时出现错误的边沿检测。

Bit 3 保留，必须保持为复位值

Bit 16, 2,1,0 **SMS[3:0]**：从模式选择（Slave mode selection）

当选择了外部信号，触发信号（tim_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）

0000：**关闭从模式** – 如果 CEN=1，则预分频器直接由内部时钟驱动。

0001：保留

0010：保留

0011：保留

0100：**复位模式** – 选中的触发输入（tim_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。

0101：**门控模式** – 当触发输入（tim_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。

0110：**触发模式** – 计数器在触发输入 tim_trgi 的上升沿启动（但不复位），只有计数器的启动是受控的。

0111：**外部时钟模式 1** – 选中的触发输入（tim_trgi）的上升沿驱动计数器。

1000：**组合复位+触发模式** - 选定触发器输入（tim_trgi）的上升沿会重新初始化计数器，生成寄存器的更新并启动计数器。

1001：**组合门控+复位模式** - 当触发器输入（tim_trgi）为高电平时计数器时钟使能。一旦触发器变为低电平计数器将停止并重置。同时控制计数器的启动和停止。

其他：保留

注：如果选择 tim_ti1f_ed *作为触发输入（*TS=0100*），则不得使用门控模式。实际上，*tim_ti1f_ed *为*TI1F *上的每个过渡输出*1 *个脉冲，而门控模式检查触发信号的电平。*

注：接收 tim_trgo *或* tim_trgo2 *信号的从机外设（定时器、*ADC *等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能随意更改。*

19.7.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x = 2，3）

偏移地址：0x00C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TIE	Reserved			CC2IE	CC1IE	UIE
									rw				rw	rw	rw

Bit 31:7 保留，必须保持为复位值。

Bit 6 **TIE**：触发信号（TGRI）中断使能（Trigger interrupt enable）

0：禁止触发信号（TGRI）中断；

1：使能触发信号（TGRI）中断。

Bit 5:3 保留，必须保持为复位值。

Bit 2 **CC2IE**：捕获/比较 2 中断使能（Capture/Compare 2 interrupt enable）

0：禁止 CC2 中断；

1：使能 CC2 中断。

Bit 1 **CC1IE**：捕获/比较 1 中断使能（Capture/Compare 1 interrupt enable）

0：禁止 CC1 中断；

1：使能 CC1 中断。

Bit 0 **UIE**：更新中断使能（Update interrupt enable）

0：禁止更新中断；

1：使能更新中断。

19.7.5 TIMx 状态寄存器（TIMx_SR）（x = 2，3）

偏移地址：0x010

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2OF	CC1OF	Reserved		TIF	Reserved			CC2IF	CC1IF	UIF
					rc_w0	rc_w0			rc_w0				rc_w0	rc_w0	rc_w0

Bit 31:11 保留，必须保持为复位值。

Bit 10 **CC2OF**：捕获/比较 2 重复捕获标志

参考 CC1OF 的描述

Bit 9 **CC1OF**：捕获/比较 1 重复捕获标志

仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件写入“0”可将该位清零。

0：未检测到重复捕获

1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

Bit 8:7 保留，必须保持为复位值。

Bit 6 **TIF**：触发中断标志（Trigger interrupt flag）

在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。

但需要通过软件清零。

0：未发生触发事件。

1：触发中断挂起。

Bit 5:3 保留，必须保持为复位值。

Bit 2 **CC2IF**：捕获/比较 2 中断标志（Capture/Compare 2 interrupt flag）

参考 CC1IF 描述。

Bit 1 **CC1IF：**捕获/比较 1 中断标志（Capture/Compare 1 interrupt flag）

此标志由硬件设置，它可以通过软件（输入捕获或输出比较模式）或读取 TIMx_CCR1 寄存器（仅输入捕获模式）来清除。

0：无比较匹配/无输入捕获发生

1：发生比较匹配或输入捕获

如果通道 CC1 配置为输出：当计数器 TIMx_CNT 的内容与 TIMx_CCR1 寄存器内容匹配时，会设置此标志。当 TIMx_CCR1 的值大于 TIMx_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和增减计数模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下有 3 种可能的标志设置选项，请参考 TIMx_CR1 寄存器中的 CMS 位以获取完整描述。

如果通道 CC1 配置为输入：当在 TIMx_CCR1 寄存器中捕获到计数器值时（在 TIMx_CCER 中定义的 CC1P 和 CC1NP 位设置所确定的 IC1 上检测到边沿时），会设置此位。

Bit 0 **UIF**：更新中断标志（Update interrupt flag）

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

– TIMx_CR1 寄存器中的 UDIS=0，并且在上溢或下溢时。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参考同步控制寄存器描述）。

19.7.6 TIMx 事件产生寄存器（TIMx_EGR）（x = 2，3）

偏移地址：0x14

复位：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Reserved			CC2G	CC1G	UG
									w				w	w	w

Bit 15:7 保留，必须保持为复位值。

Bit 6 **TG**：生成触发信号（Trigger generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件。

Bit 5:3 保留，必须保持为复位值。

Bit 2 **CC2G**：捕获/比较 2 生成（Capture/Compare 2 generation）

参考 CC1G 描述。

Bit 1 **CC1G**：捕获/比较 1 生成（Capture/Compare 1 generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：通道 1 上生成捕获/比较事件：

若通道 CC1 配置为输出：

使能时，CC1IF 标志置 1 并发送相应的中断。

如果通道 CC1 配置为输入：

TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

Bit 0 **UG**：更新生成（Update generation）

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作；

1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或 DIR=0（递增计数），计数器将清零；如果 DIR=1（递减计数），计数器将使用自动重载值（TIMx_ARR）。

19.7.7 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = 2，3）

偏移地址：0x018

复位值：0x0000 0000

相同的寄存器可以用于输入捕获模式（本节）或输出比较模式（下节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

Bit 31:16 保留，必须保持为复位值。

Bit 15:12 **IC2F[3:0]：**输入捕获 2 滤波器（Input capture 2 filter）

Bit 11:10 **IC2PSC[1:0]：**输入/捕获 2 预分频器（input capture 2 prescaler）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择 （Capture/compare 2 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道被配置为输出；

01：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti2 上；

10：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti1 上；

11：CC2 通道被配置为输入，tim_ic2 映射在 tim_trc 上。此模此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：仅当通道关闭时（TIMx_CCER 中的 CC2E = 0），才可向 CC2S 位写入数据。

Bit 7:4 **IC1F[3:0]：**输入捕获 1 滤波器（Input capture 1 filter）

此位域可定义 tim_ti1 输入的采样频率和适用于 tim_ti1 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个事件才视为一个有效边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6。

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5。

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6。

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5。

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8。

Bit 3:2 **IC1PSC[1:0]：**输入/捕获 1 预分频器（Input capture 1 prescaler）

此位域定义 CC1 输入（tim_ic1）的预分频比。

只要 CC1E=0（TIMx_CCER 寄存器），预分频器便立即复位。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获；

10：每发生 4 个事件便执行一次捕获；

11：每发生 8 个事件便执行一次捕获。

Bit 1:0 **CC1S[1:0]：**捕获/比较 1 选择（Capture/Compare 1 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道被配置为输出；

01：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti1 上；

10：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti2 上；

11：CC1 通道被配置为输入，tim_ic1 映射在 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

19.7.8 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = 2，3）

偏移地址：0x018

复位值：0x0000 0000

相同的寄存器可以用于输出比较模式（本节）或输入捕获模式（上节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							OC2M[3]	Reserved							OC1M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

Bit 31:25 保留，必须保持为复位值。

Bit 23:17 保留，必须保持为复位值。

Bit 15 **OC2CE：**输出比较 2 清零使能（Output Compare 2 clear enable）

Bit 24, 14:12 **OC2M[2:0]：**输出比较 2 模式（Output Compare 2 mode）

参考 OC1M 描述

Bit 11 **OC2PE：**输出比较 2 预装载使能（Output Compare 2 preload enable）

Bit 10 **OC2FE：**输出比较 2 快速使能（Output Compare 2 fast enable）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择（Capture/Compare 2 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上

10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上

11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIMx_CCER *中的*CC2E = 0*），才可向*CC2S *位写入数据*

Bit 7 **OC1CE：**输出比较 1 清零使能（Output Compare 1 clear enable）

0：tim_oc1ref 不受 tim_ocref_clr_int 输入的影响；

1：一旦在 tim_ocref_clr_int 输入上检测到高电平，tim_oc1ref 立即清零。

Bit 16, 6:4 **OC1M[3:0]：**输出比较 1 模式（Output Compare 1 mode）

这些位定义提供 tim_oc1 和 tim_oc1n 的输出参考信号 tim_oc1ref 的行为。tim_oc1ref 为高电平有效，而 tim_oc1 和 tim_oc1n 的有效电平则取决于 CC1P 位和 CC1NP 位。

0000：**冻结** - 输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1（TIMx_CCR1）匹配时，tim_oc1ref 信号强制变为高电平。

0010：将通道 1 设置为匹配时输出无效电平。当计数器 TIMx_CNT 与捕获/比较寄存器 1（TIMx_CCR1）匹配时，tim_oc1ref 信号强制变为低电平。

0011：**翻转** - TIMx_CNT=TIMx_CCR1 时，tim_oc1ref 发生翻转。

0100：**强制变为无效电平** – tim_oc1ref 强制变为低电平。

0101：**强制变为有效电平** – tim_oc1ref 强制变为高电平。

0110：**PWM 模式 1**-在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为无效状态（tim_oc1ref=0），否则为有效状态（tim_oc1ref=1）。

0111：**PWM 模式 2** – 在递增计数模式下，只要 TIMx_CNT<TIMx_CCR1，通道 1 便为无效状态，否则为有效状态。在递减计数模式下，只要 TIMx_CNT>TIMx_CCR1，通道 1 便为有效状态，否则为无效状态。

1000：**可重触发 OPM 模式 1**- 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于无效状态。

1001：**可重触发 OPM 模式 2** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 2 那样执行比较，并在下次更新时使通道再次处于无效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。

1010：保留。

1011：保留。

1100：**组合 PWM 模式 1** - tim_oc1ref 的行为与 PWM 模式 1 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑 OR。

1101：**组合 PWM 模式 2** - tim_oc1ref 的行为与 PWM 模式 2 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑 AND。

1110：保留。

1111：保留。

注：在 PWM 模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“PWM”模式时，tim_ocref_clr 电平才会发生更改。

Bit 3 **OC1PE**：输出比较 1 预装载使能（Output Compare 1 preload enable）

0：禁止与 TIMx_CCR1 相关的预装载寄存器。可随时向 TIMx_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIMx_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIMx_CCR1 预装载值在每次生成更新事件时都会装载到活动寄存器中。

Bit 2 **OC1FE**：输出比较 1 快速使能（Output Compare 1 fast enable）

此位减少了触发事件和定时器输出上的转换之间的延迟。它必须在单脉冲模式（在 TIMx_CR1 寄存器中设置 OPM 位）中使用，以便在启动触发后尽快开始输出脉冲。

0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。

1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。

Bit 1:0 **CC1S[1:0]**：捕获/比较 1 选择（Capture/Compare 1 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上

10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上

11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：仅当通道关闭时（TIMx_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据

19.7.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = 2，3）

偏移地址：0x020

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CC2NP	Res	CC2P	CC2E	CC1NP	Res	CC1P	CC1E
								rw		rw	rw	rw		rw	rw

Bit 14:8 保留，必须保持为复位值。

Bit 7 **CC2NP**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1NP 的描述。

Bit 6 保留，必须保持为复位值。

Bit 5 **CC2P**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1P 的描述。

Bit 4 **CC2E**：输入/捕获 2 输出使能（Capture/Compare 2 output enable）

参考 CC1E 的描述。

Bit 3 **CC1NP**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

CC1 通道配置为输出：

在这种情况下， CC1NP 必须保持清零。

CC1 通道配置为输入：

此位与 CC1P 配合使用，用以定义 tim_ti1fp1/ tim_ti2fp1 的极性。请参见 CC1P 说明。

Bit 2 保留，必须保持为复位值。

Bit 1 **CC1P**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

0：OC1 高电平有效（输出模式）/边沿灵敏度选择（输入模式，参见以下）

1：OC1 低电平有效（输出模式）/边沿灵敏度选择（输入模式，参见以下）

CC1 通道配置为输入，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。

CC1NP=0，CC1P=0：非反相/上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。

CC1NP=0，CC1P=1：反相/下降沿触发。电路对 TIxFP1 下降沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式下执行触发操作）。

CC1NP=1，CC1P=1：非反相/上升沿和下降沿均触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。

CC1NP=1，CC1P=0：此配置是保留的，不应该使用此配置。

Bit 0 **CC1E**：输入/捕获 1 输出使能（Capture/Compare 1 output enable）

0：禁止捕获模式/OC1 未激活

1：使能捕获模式/ OC1 信号输出到对应的输出引脚

表 19.12 标准 tim_ocx 通道的输出控制位

CCxE 位	tim_ocx 输出状态
0	禁止输出（不由定时器驱动：高阻态）
1	输出使能（tim_ocx= tim_ocxref + 极性）

注：当 CCxE=0 *时，连接到* tim_ocx *通道的外部* IO *引脚状态仅取决于* GPIO *寄存器*

19.7.10 TIM3 计数器（TIM3_CNT）

偏移地址：0x024

复位值：0x0000 0000

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

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

Bit 31 **UIFCPY：** 值取决于 TIMx_CR1 的 UIFREMAP

如果 UIFREMAP = 0

保留

如果 UIFREMAP = 1

UIFCPY：复制 UIF

该位是 TIMx_SR 寄存器中 UIF 位的一个只读副本。

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CNT[15:0]：**计数器的值（Counter value）

在非抖动模式下（DITHEN = 0）

该寄存器（CNT[15:0]）保存的是计数器的值。

在抖动模式下（DITHEN = 1）

该寄存器（CNT[15:0]）只保存计数器的不抖动部分。抖动部分不可用。

19.7.11 TIM2 计数器（TIM2_CNT）

偏移地址：0x024

复位值：0x0000 0000

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

Bit 31 **UIFCPY：**值取决于 TIMx_CR1 的 UIFREMAP

如果 UIFREMAP = 0

保留

如果 UIFREMAP = 1

UIFCPY：复制 UIF

该位是 TIMx_SR 寄存器中 UIF 位的一个只读副本。

Bit 30:28 保留，必须保持复位值。

Bit 27:0 **CNT[27:0]：**计数器值的最小有效部分（Least significant part of counter value）

在非抖动模式下（DITHEN = 0）

该寄存器（CNT[27:0]）保存的是计数器的值。

在抖动模式下（DITHEN = 1）

该寄存器（CNT[27:0]）只保存计数器的不抖动部分。抖动部分不可用。

19.7.12 TIMx 预分频器（TIMx_PSC）（x = 2，3）

偏移地址：0x028

复位值：0x0000

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

Bit 15:0 **PSC[15:0]：**预分频器的值（Prescaler value）

计数器时钟频率（$f_{tim_cnt_ck}$）等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到活动预分频器寄存器的值。

19.7.13 TIM3 自动重装载寄存器（TIM3_ARR）

偏移地址：0x02C

复位：0x0000 FFFF

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

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **ARR[19:0]：**自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

当自动重载值为空时，计数器不工作。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 ARR[15:0]中的自动重新加载的值

在抖动模式下（DITHEN = 1）

该寄存器存储的是 ARR[19:4]中的整数部分，而 ARR[3:0]位域存储的是抖动的部分

19.7.14 TIM2 自动重装载寄存器（TIM2_ARR）

偏移地址：0x02C

复位：0xFFFF FFFF

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

Bit 31:0 **ARR[31:0]：**自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

当自动重载值为空时，计数器不工作。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 ARR[27:0]中的自动重新加载的值

在抖动模式下（DITHEN = 1）

该寄存器存储的是 ARR[31:4]中的整数部分，而 ARR[3:0]位域存储的是抖动的部分

19.7.15 TIM3 捕获/比较寄存器 1（TIM3_CCR1）

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR1 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR1[19:0]：**捕获/比较 1 值（Capture/Compare 1 value）

若 CC1 通道配置为输出：

CCR1 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 1 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR1[15:0]中的比较值。CCR1[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR1[19:4]中的整数部分。CCR1[3:0]位域存储的是抖动的部分。

若 CC1 通道配置为输入：

CCR1 为上一个输入捕获 1 事件（tim_ic1）发生时的计数器值。TIMx_CCR1 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR1[15:0]位存储了捕获值。CCR1[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR1[19:4]中。CCR1[3:0]位会被重置。

19.7.16 TIM2 捕获/比较寄存器 1（TIM2_CCR1）

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1 [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
CCR1 [15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **CCR1[31:0]：**捕获/比较 1 值（Capture/Compare 1 value）

若 CC1 通道配置为输出：

CCR1 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 1 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR1[27:0]中的比较值。CCR1[31:28]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR1[31:4]中的整数部分。CCR1[3:0]位域存储的是抖动的部分。

若 CC1 通道配置为输入：

CCR1 为上一个输入捕获 1 事件（tim_ic1）发生时的计数器值。TIMx_CCR1 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR1[27:0]位存储了捕获值。CCR1[31:28]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR1[31:4]中。CCR1[3:0]位会被重置。

19.7.17 TIM3 捕获/比较寄存器 2（TIM3_CCR2）

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR2 [19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR2[19:0]：**捕获/比较 2 值（Capture/Compare 2 value）

若 CC2 通道配置为输出：

CCR2 为要装载到实际捕获/比较 2 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 2 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc2 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR2[15:0]中的比较值。CCR2[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR2[19:4]中的整数部分。CCR2[3:0]位域存储的是抖动的部分。

若 CC2 通道配置为输入：

CCR2 为上一个输入捕获 2 事件（tim_ic2）发生时的计数器值。TIMx_CCR2 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR2[15:0]位存储了捕获值。CCR2[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR2[19:4]中。CCR2[3:0]位会被重置。

19.7.18 TIM2 捕获/比较寄存器 2（TIM2_CCR2）

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[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
CCR2[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:0 **CCR2[31:0]：**捕获/比较 2 值（Capture/Compare 2 value）

若 CC2 通道配置为输出：

CCR2 为要装载到实际捕获/比较 2 寄存器的值（预装载值）。

如果没有通过 TIMx_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 2 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIMx_CNT 进行比较并在 tim_oc1 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR2[27:0]中的比较值。CCR2[31:28]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR2[31:4]中的整数部分。CCR2[3:0]位域存储的是抖动的部分。

若 CC2 通道配置为输入：

CCR2 为上一个输入捕获 2 事件（tim_ic2）发生时的计数器值。TIMx_CCR2 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR2[27:0]位存储了捕获值。CCR2[31:28]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR2[31:4]中。CCR2[3:0]位会被重置。

19.7.19 TIMx 编码器控制寄存器（TIMx_ECR）（x = 2，3）

偏移地址：0x058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					PWPRSC[2:0]			PW[7:0]							
					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
Reserved															

Bit 31:27 保留，必须保持为复位值。

Bit 26:24 **PWPRSC[2:0]：**脉冲宽度预分频器（Pulse width prescaler）

该位域设置脉冲发生器的时钟预分频，如下所示：

$t_{PWG} = (2^{(PWPRSC[2:0])}) \times t_{tim_ker_ck}$

Bit 23:16 **PW[7:0]：**脉冲宽度（Pulse width）

该位域定义脉冲持续时间，如下所示：

$t_{PW} = PW[7:0] \times t_{PWG}$

Bit 15:0 保留，必须保持为复位值。

19.7.20 TIMx 定时器输入选择寄存器（TIMx_TISEL）（x = 2，3）

偏移地址：0x05C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					TI2SEL[2:0]			Reserved					TI1SEL[2:0]		
					rw	rw	rw						rw	rw	rw

Bit 31:11 保留，必须保持为复位值。

Bit 10:8 **TI2SEL[2:0]：**选择 tim_ti2[5:0]输入（Selects tim_ti2[5:0] input）

000：tim_ti2_in0: TIMx_CH2

001：tim_ti2_in1

010：tim_ti2_in2

011：tim_ti2_in3

100：tim_ti2_in4

101：tim_ti2_in5

111：保留

参考章节：TIM2/TIM3 引脚和内部信号获得关联列表。

Bit 7:3 保留，必须保持为复位值。

Bit 2:0 **TI1SEL[2:0]：**选择 tim_ti1[5:0]输入（Selects tim_ti1[5:0] input）

000：tim_ti1_in0: TIMx_CH1

001：tim_ti1_in1

010：tim_ti1_in2

011：tim_ti1_in3

100：tim_ti1_in4

101：tim_ti1_in5

111：保留

参考章节：TIM2/TIM3 引脚和内部信号获得关联列表。

19.7.21 TIMx 复用功能选择寄存器 2（TIMx_AF2）（x = 2，3）

偏移地址：0x064

复位值：0x0000 0000

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

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

Bit 31:18 保留，必须保持为复位值。

Bit 17:16 **OCRSEL[1:0]：**ocref_clr 的源选择（ocref_clr source selection）

这些位选择 ocref_clr 的输入源

00：tim_ocref_clr0

01：tim_ocref_clr1

10：tim_ocref_clr2

11：tim_ocref_clr3

请参考章节：TIM2/TIM3 引脚和内部信号，了解特定于产品的实现。

Bit 15:0 保留，必须保持为复位值。

20 通用定时器（TIM15）

20.1 TIM15 简介

TIM15 定时器包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

它可用于多种用途，包括测量输入信号的脉冲宽度（输入捕获）或生成输出波形（输出比较和 PWM 和带死区插入的互补 PWM）。 使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度和波形周期从几微秒调制到几毫秒。TIM15 定时器是完全独立的，不与其他定时器共享任何资源。

20.2 TIM15 主要功能

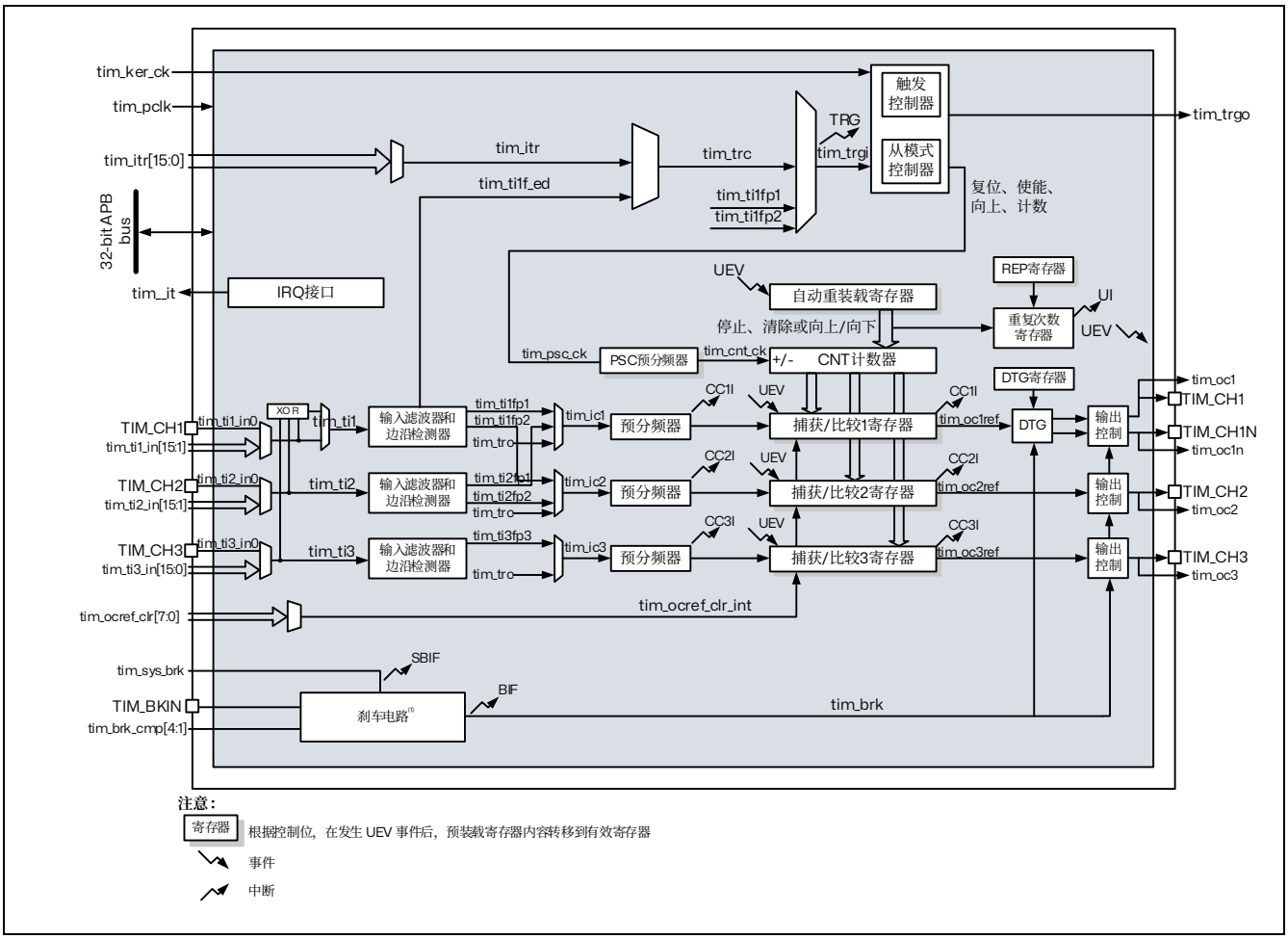
TIM15 定时器功能包括：

- 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 1～65536 之间的任意数值
- 3 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边缘模式）
  - 单脉冲模式输出
- 带可编程死区的互补输出（仅限通道 1）
- 使用外部信号控制定时器和定时器互连的同步电路
- 重复计数器，用于仅在给定数目的计数器周期后更新定时器寄存器。
- 用于将定时器的输出信号置于复位状态或已知状态的刹车输入。
- 如下事件发生时产生中断：
  - 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
  - 刹车输入（中断请求）

20.3 TIM15 功能描述

20.3.1 框图

图 20.1 TIM15 框图



注 1. 参考章节：使用刹车功能以获得细节

20.3.2 TIM15 引脚和内部信号

本节中的下两张表格总结了 TIM 的输入和输出。

表 20.1 TIM15 输入/输出引脚

引脚名称	信号类型	描述
TIM_CH1 TIM_CH2 TIM_CH2	输入/输出	定时器多功能通道。 每个通道用于捕获、比较或 PWM。 TIM_CH1 和 TIM_CH2 也可用作外部时钟（tim_ker_ck 时钟的 1/4 以下）和外部触发器输入。
TIM_CH1N	输出	由 TIM_CH1 输出派生的定时器的互补输出，具有插入死区时间的能力。
TIM_BKIN	输入/输出	刹车输入。此输入也可以配置为双向模式。

表 20.2 TIM15 内部输入/输出信号

内部信号名称	信号类型	描述
tim_ti1_in[15:0] tim_ti2_in[15:0] tim_ti3_in[15:0]	输入	内部定时器输入总线。 这些输入可被用作捕获或作为外部时钟 (tim_ker_ck 时钟的 1/4 以下)。
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或作为输入时钟(tim_ker_ck 时钟的 1/4 以下)。
tim_trgo	输出	内部触发器输出。此触发器可以触发其他片上外设。
tim_ocref_clr[7:0]	输入	定时器 tim_ocref_clr 输入总线。这些输入可以用来清除 tim_ocxref 信号，通常用于硬件的逐周期脉冲宽度控制。
tim_brk_cmp[4:1]	输入	内部信号的刹车输入。
tim_sys_brk[n:0]	输入	系统刹车输入。该输入收集 MCU 的系统级错误。
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟。此时钟必须与 tim_pclk 同步（来自同一源）。 时钟比率 tim_ker_ck/ tim_pclk 必须为整数： 1，2，3，...，16（最大值）。
tim_it	输出	全局定时器中断，收集捕获/比较、更新、中断触发和换向请求。

如下三张表列出了连接到 tim_ti[3:1]输入多路复用器的源。

表 20.1 连接到 tim_ti1 的输入多路复用器

tim_ti1 输入	源
	TIM15
tim_ti1_in0	TIM15_CH1
tim_ti1_in1	comp1_out
tim_ti1_in2	comp2_out
tim_ti1_in3	comp3_out
tim_ti1_in4	comp4_out

表 20.2 连接到 tim_ti2 的输入多路复用器

tim_ti2 输入	源
	TIM15
tim_ti2_in0	TIM15_CH2
tim_ti2_in1	comp1_out
tim_ti2_in2	comp2_out
tim_ti2_in3	comp3_out
tim_ti2_in4	comp4_out

表 20.3 连接到 tim_ti3 的输入多路复用器

tim_ti3 输入	源
	TIM15
tim_ti3_in0	TIM15_CH3
tim_ti3_in1	comp1_out
tim_ti3_in2	comp2_out
tim_ti3_in3	comp3_out
tim_ti3_in4	comp4_out
tim_ti3_in5	HSE/128
tim_ti3_in6	LSI

下表列出连接到 tim_itr 输入多路复用器的内部源

表 20.3 TIM15 内部触发连接

tim_itrx 输入	TIM15
tim_itr0	tim2_trgo
tim_itr1	tim3_trgo
tim_itr2	tim8_trgo
tim_itr3	tim1_trgo
tim_itr4	timA_trgo
tim_itr5	timB_trgo
tim_itr[15:6]	保留

下两张表列出了连接到 tim_brk 输入的源

表 20.4 定时器刹车互连

tim_brk 输入	TIM15
TIM_BKIN	TIM15_BKIN 引脚
tim_brk_cmp1	comp1_out
tim_brk_cmp2	comp2_out
tim_brk_cmp3	comp3_out
tim_brk_cmp4	comp4_out

表 20.5 系统中断互连

tim_sys_brk 输入	TIM15	SYSCFG_CFGR2 寄存器的使能位
tim_sys_brk0	内核锁住	CLL
tim_sys_brk1	可编程电压检测器（PVD）	PVDL
tim_sys_brk2	时钟安全系统(CSS)	无（始终使能）

下表列出连接到 tim_ocref_clr 输入多路复用器的内部源

表 20.6 连接到 ocref_clr 的输入多路复用器

定时器 OCREF 清除信号	定时器 tim_ocref_clr 信号分配
	TIM15
tim_ocref_clr0	comp1_out
tim_ocref_clr1	comp2_out
tim_ocref_clr2	comp3_out
tim_ocref_clr3	comp4_out

20.3.3 时基单元

可编程定时器的主要模块由一个 16 位递增计数器及其相关的自动重装寄存器组成。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包含:

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）
- 重复计数器寄存器 (TIMx_RCR)

自动重载寄存器是预装载的。对自动重载寄存器执行写入或读取操作时会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件(UEV)时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位 (ARPE)。当计数器达到溢出并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位（CEN）置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

注意，计数器将在 TIMx_CR1 寄存器的 CEN 位置 1 时刻的一个时钟周期后开始计数。

预分频器描述

预分频器可对计数器时钟频率进行分频，分频系数介于 1 到 65536 之间。该预分频器基于 16 位寄存器（TIMx_PSC 寄存器）所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用.

下图给出了在预分频器运行时，更改计数器参数的例子。

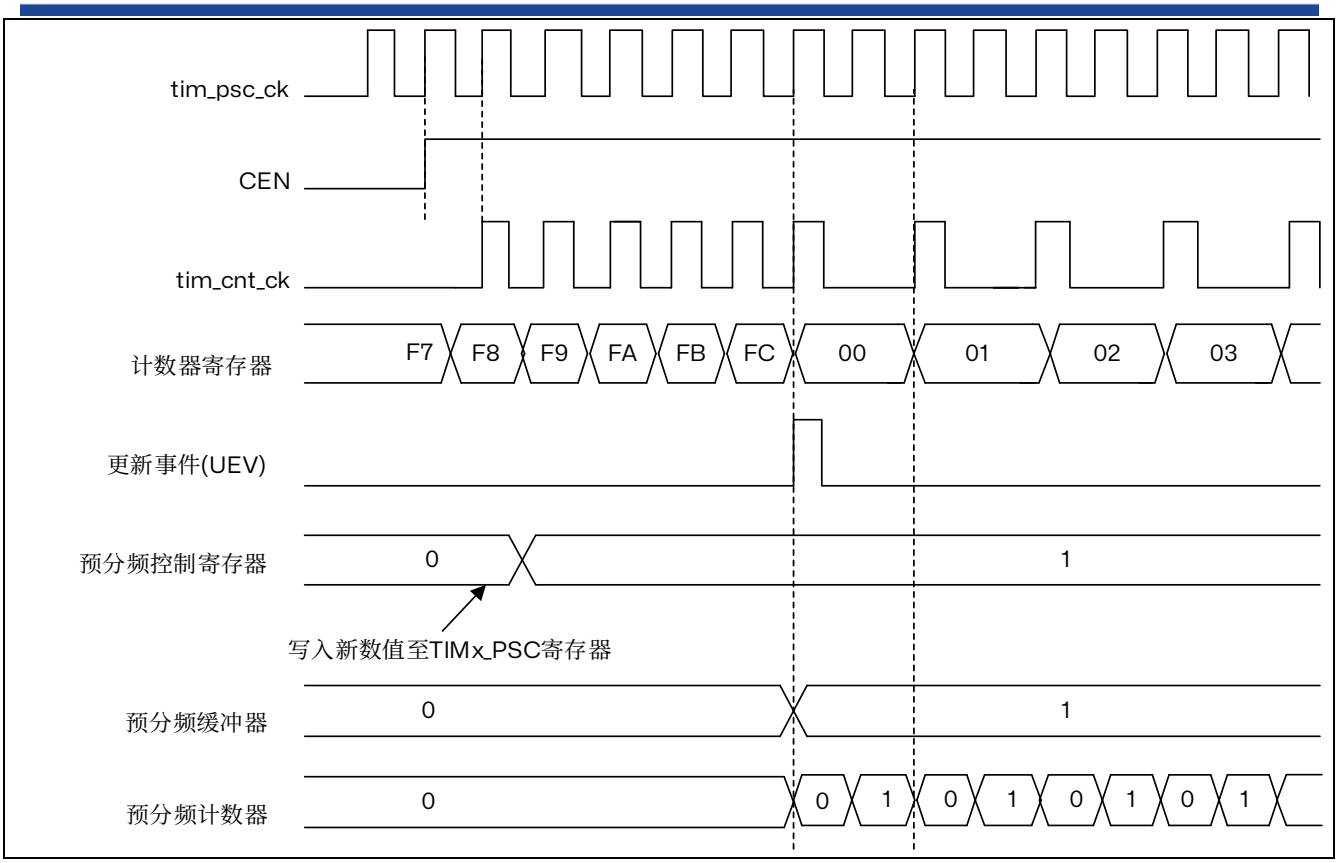


图 20.2 当预分频器的参数从 1 变到 2 时，计数器的时序图

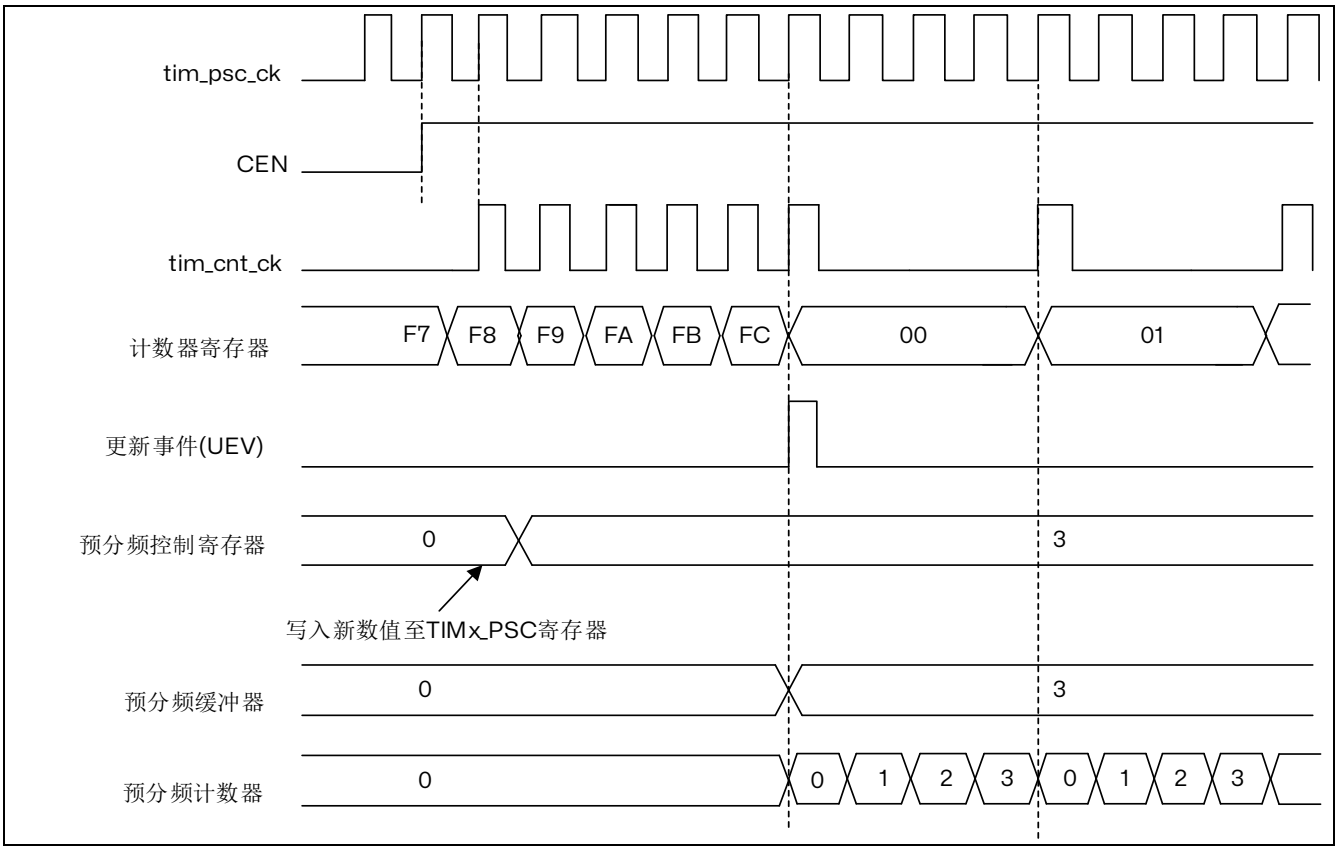


图 20.3 当预分频器的参数从 1 变到 4 时，计数器的时序图

20.3.4 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

如果使用重复计数器，则当递增计数的重复次数达到重复计数器寄存器中编程的次数加一次(TIMx_RCR+1)后，将生成更新事件（UEV)。否则，将在每次计数器上溢时产生更新事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变)。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断)。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位)：

- 重复计数器中将重新装载 TIMx_RCR 寄存器的内容
- 自动重载影子寄存器将以预装载值进行更新（TIMx_ARR）
- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

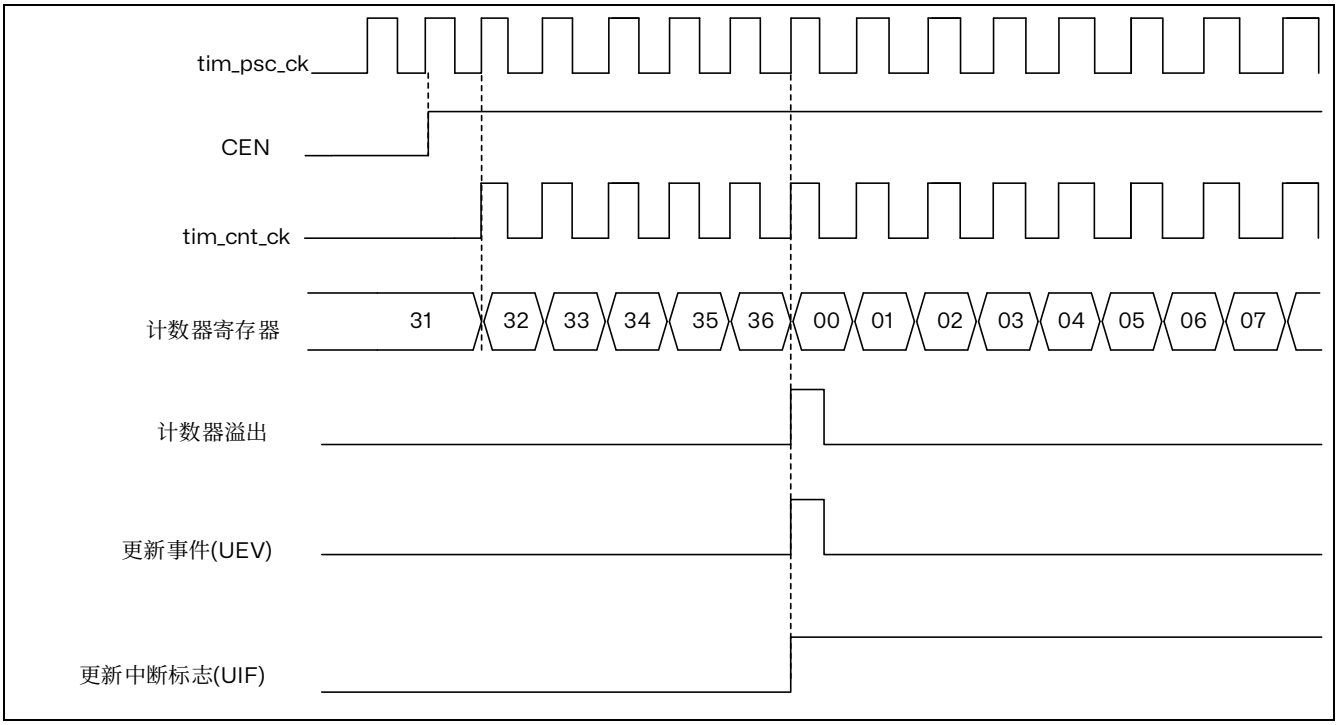


图 20.4 计数器时序图，内部时钟分频系数为 1

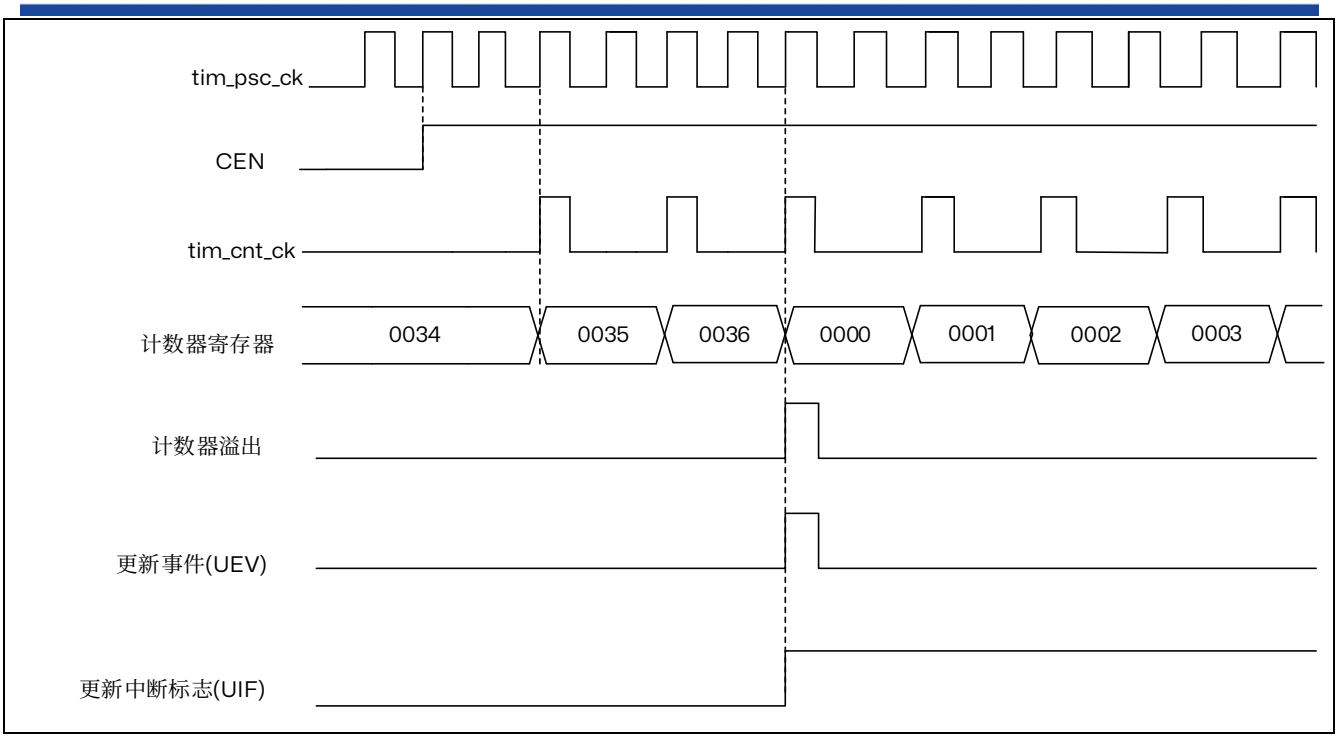


图 20.5 计数器时序图，内部时钟分频系数为 2

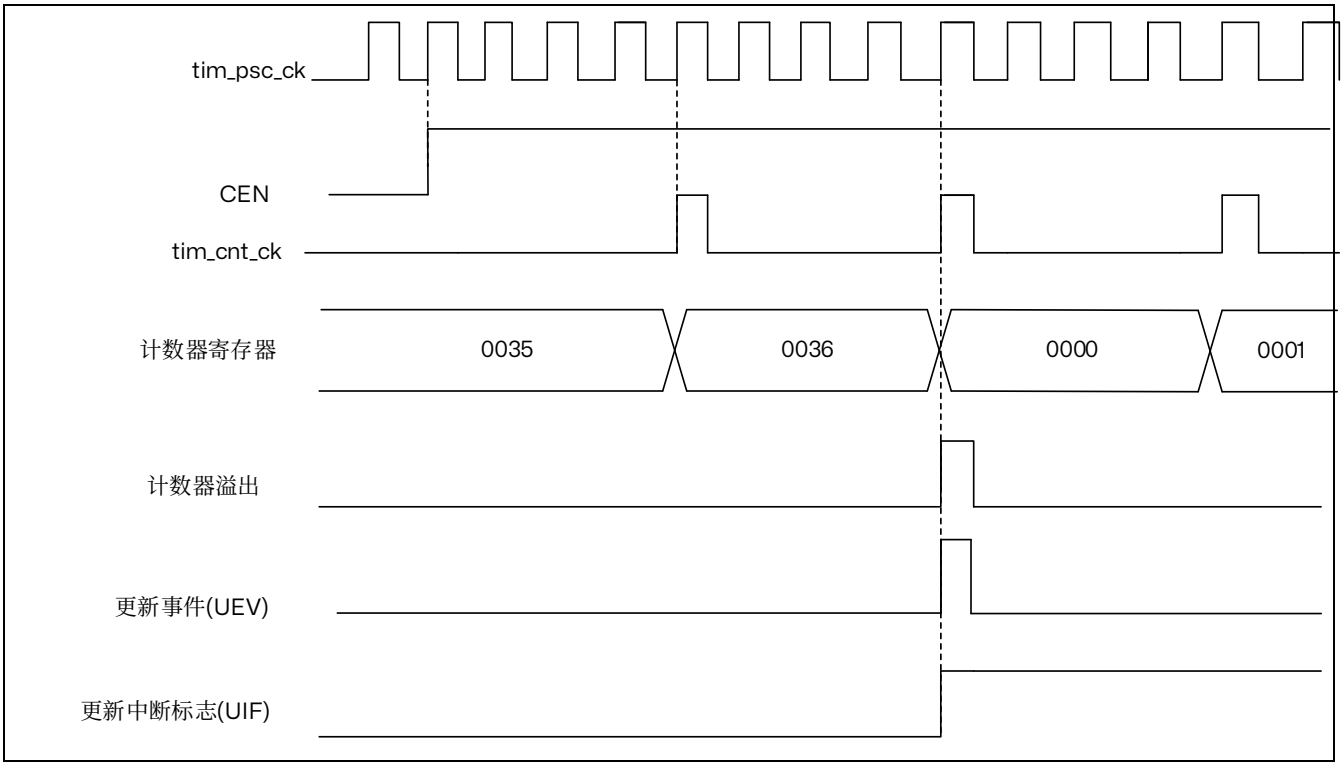


图 20.6 计数器时序图，内部时钟分频系数为 4

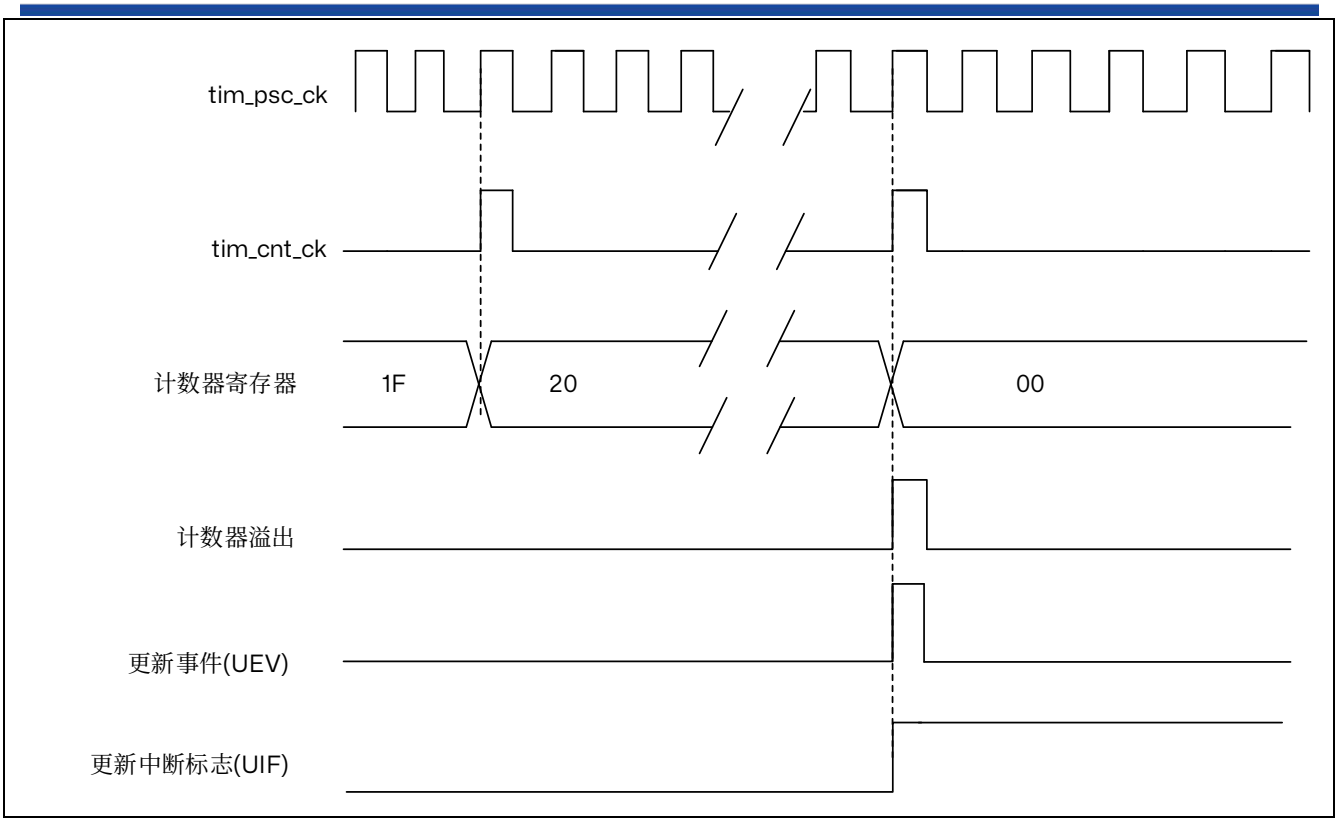


图 20.7 计数器时序图，内部时钟分频系数为 N

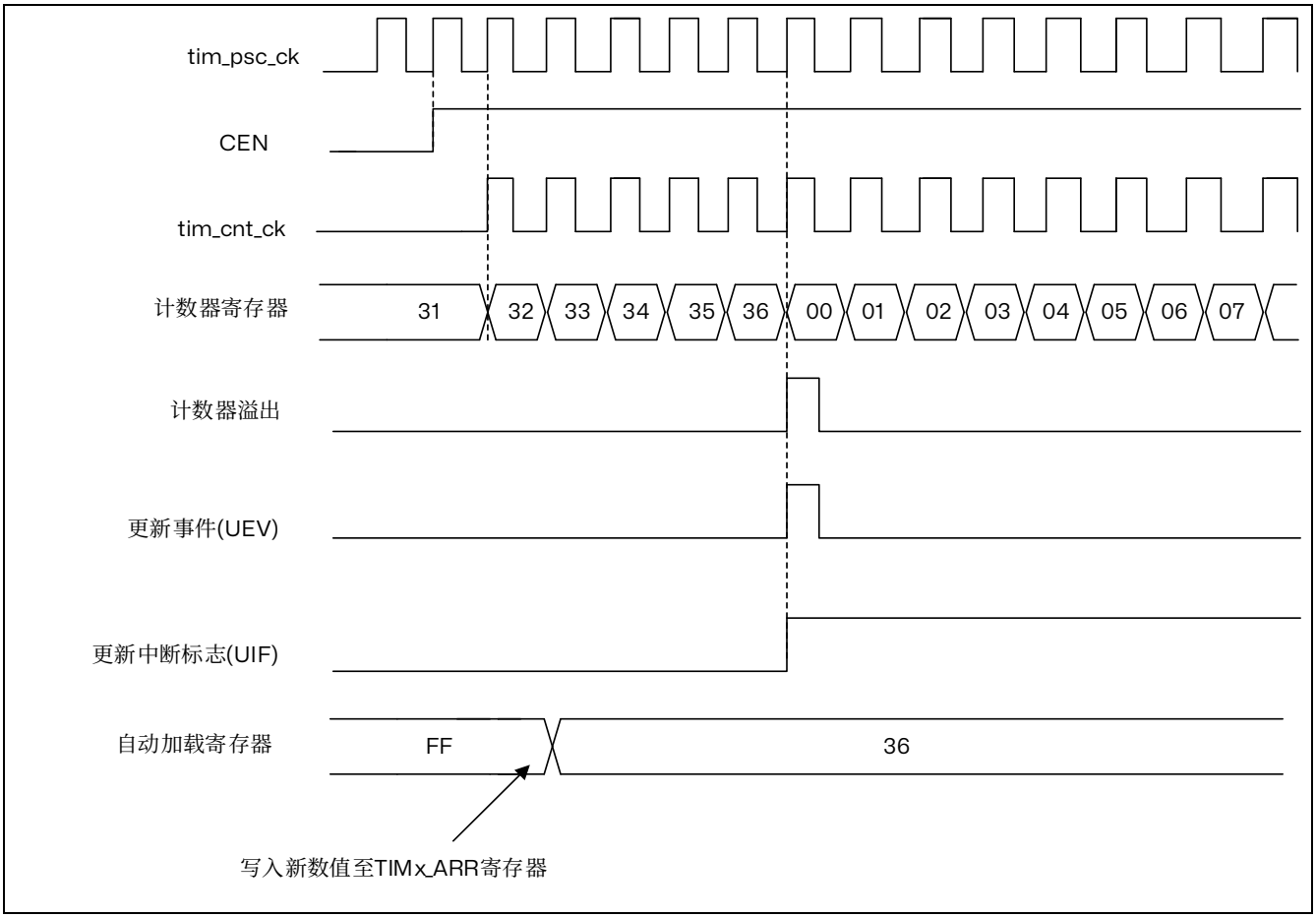


图 20.8 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

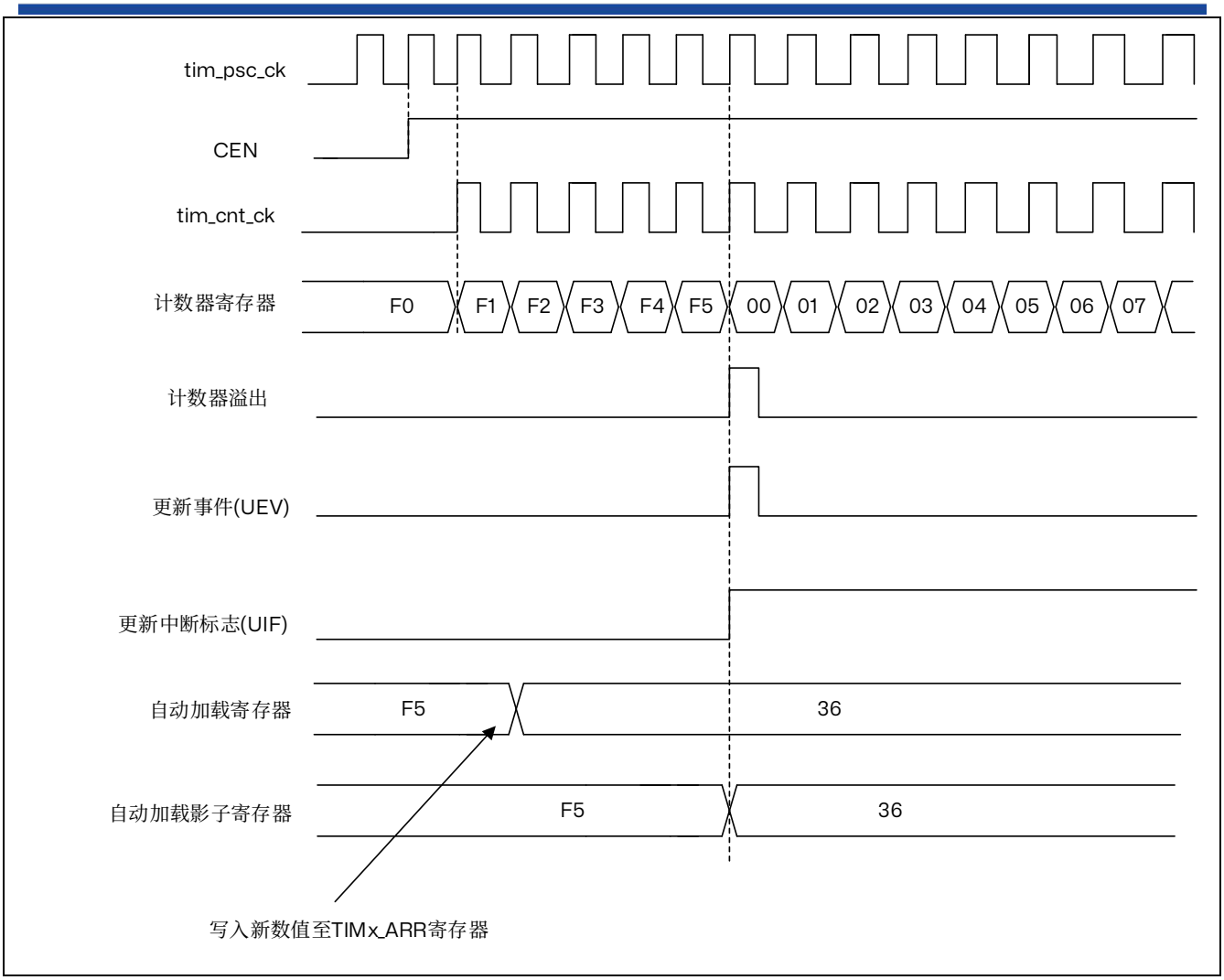


图 20.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

20.3.5 重复计数器

时基单元介绍如何因计数器溢出而生成更新事件(UEV)。实际上，只有当重复计数器达到零时，才会生成更新事件。这在生成 PWM 信号时很有用。

这意味着，每当发生 N 个计数器溢出（其中，N 是 TIMx_RCR 重复计数器寄存器中的值），数据就将从预装载寄存器转移到影子寄存器（TIMx_ARR 自动重载寄存器、TIMx_PSC 预分频器寄存器以及比较模式下的 TIMx_CCRx 捕获/比较寄存器）。

重复计数器在每次计数器溢出时递减：

重复计数器是自动重载类型；其重复率为 TIMx_RCR 寄存器所定义的值。当更新事件由软件（通过将 TIMx_EGR 寄存器的 UG 位置 1）或硬件（通过从模式控制器）生成时，无论重复计数器的值为多少，更新事件都将立即发生，并且在重复计数器中重新装载 TIMx_RCR 寄存器的内容。

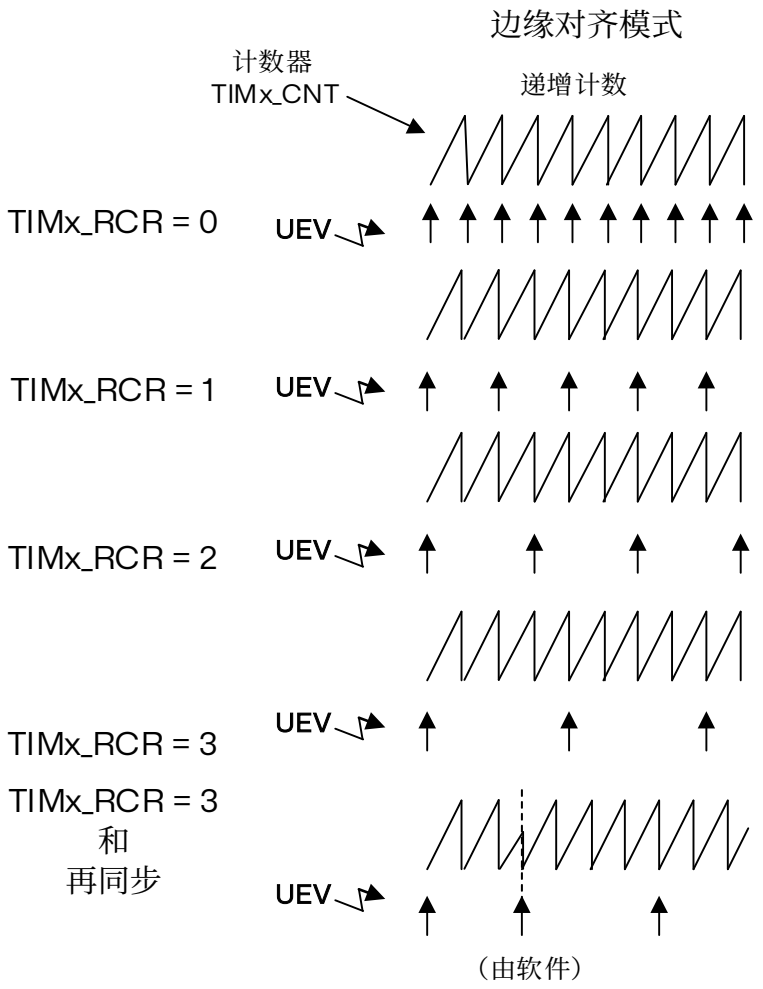


图 20.10 取决于模式和 TIMx_RCR 寄存器设置的更新频率示例

20.3.6 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（tim_ker_ck）
- 外部时钟模式 1：外部输入引脚（tim_ti1 or tim_ti2，如果可用）
- 内部触发输入(tim_itrx)：使用一个定时器作为另一个定时器的预分频器，例如可以将 TIM1 配置为 TIM15 的预分频器。

内部时钟源（tim_ker_ck）

如果禁止从模式控制器（TIMx_SMCR 寄存器中 SMS=000），则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

下图显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

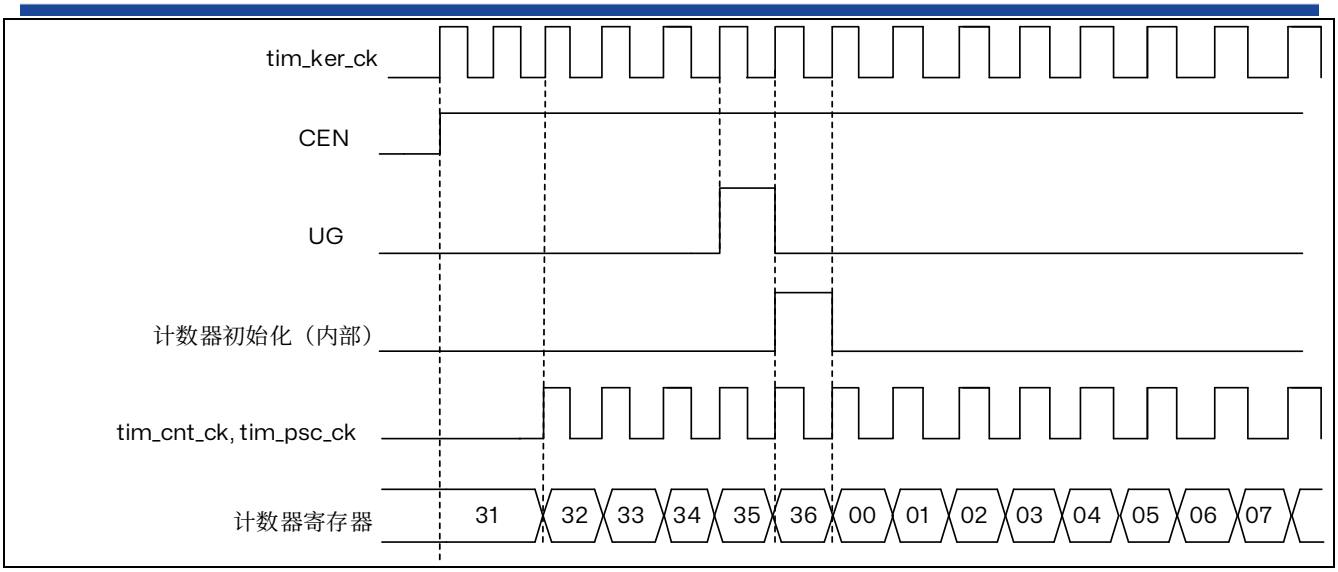


图 20.11 一般模式下的控制电路，内部时钟分频系数为 1

外部时钟源模式 1

当 TIMx_SMCR 寄存器中的 SMS=111 时，可选择此模式。计数器可在选定的输入信号上出现上升沿或下降沿时计数。

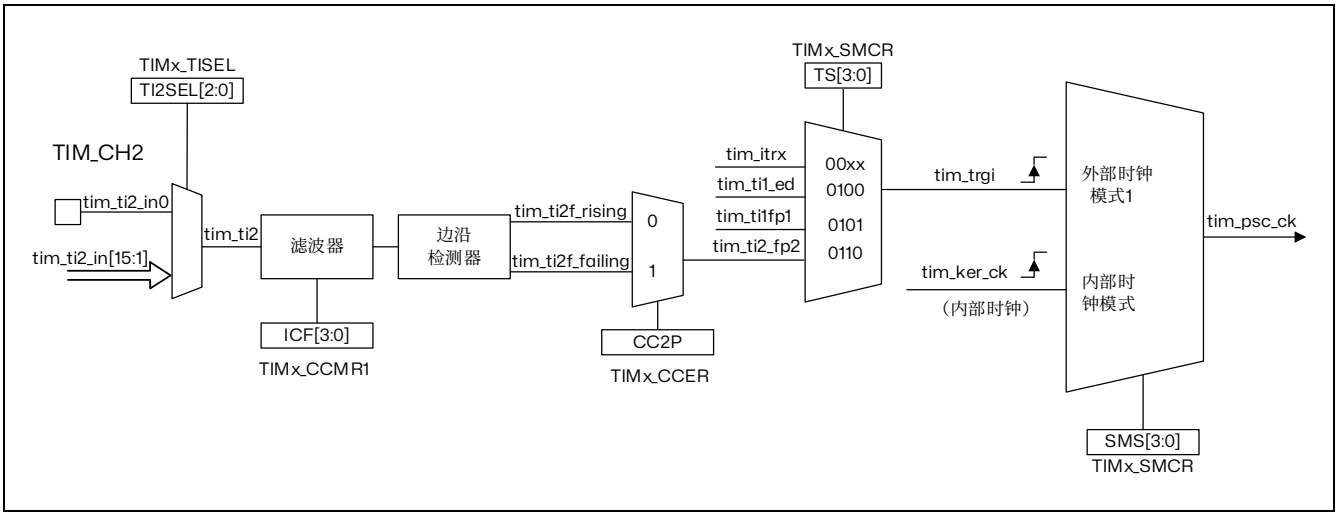


图 20.12 tim_ti2 外部时钟连接例子

例如，要使递增计数器在 tim_ti2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIMx_TISEL 寄存器中使用 TI2SEL[2:0]位选择合适的 tim_ti2_in[4:0]源（内部或外部）。
2. 通过在 TIMx_CCMR1 寄存器中写入 CC2S=“01”来配置通道 2，使其能够检测 tim_ti2 输入的上升沿。
3. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0]位来配置输入滤波时间（如果不需要任何滤波，请保持 IC2F=0000）。
4. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
5. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
6. 通过在 TIMx_SMCR 寄存器中写入 TS=110 来选择 tim_ti2 作为输入源。
7. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

注：由于捕获预分频器不用于触发操作，因此无需对其进行配置。

当 tim_ti2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器时钟之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

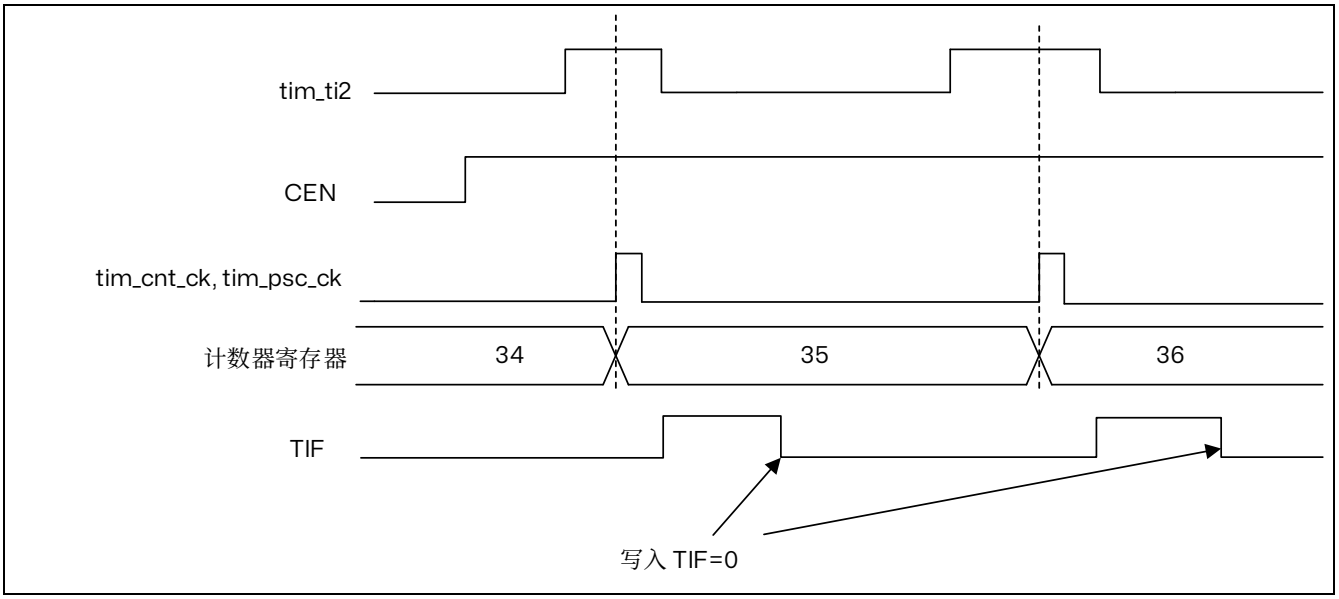


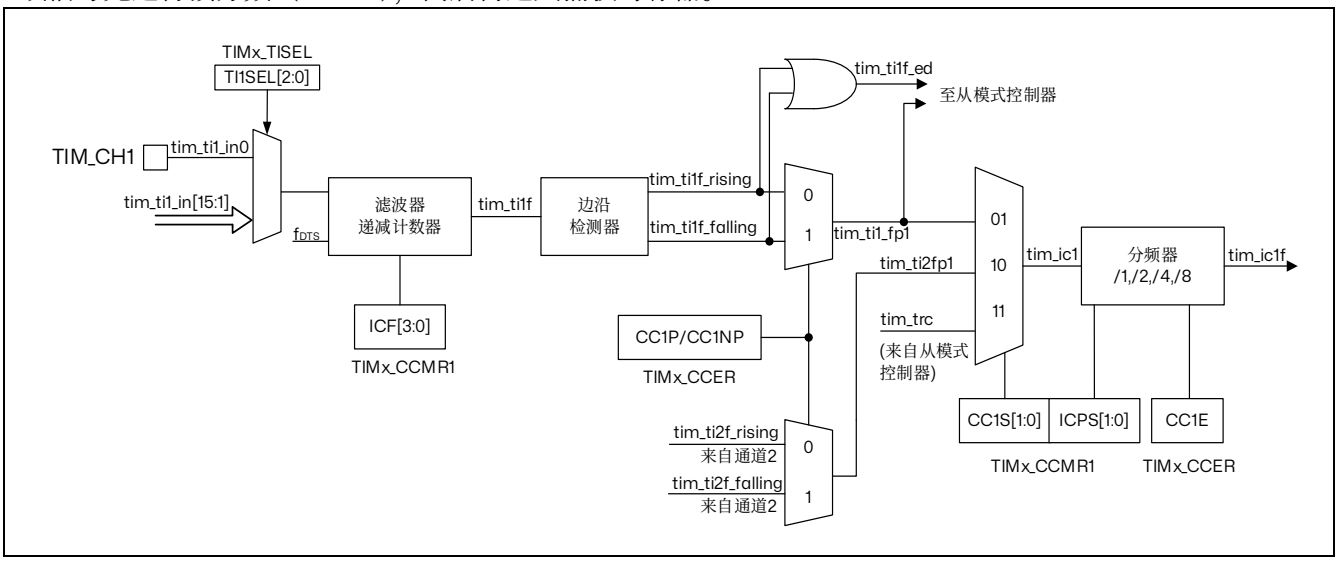
图 20.13 外部时钟模式 1 下的控制电路

20.3.7 捕获/比较通道

每个捕获/比较通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）、一个捕获输入阶段（数字滤波、多路复用和预分频器）和一个输出阶段（比较器和输出控制）构建而成。

下图概括介绍了一个捕获/比较通道。

输入阶段对相应的 tim_tix 输入进行采样，生成一个滤波后的信号 tim_tixf。然后，带有极性选择功能的边沿检测器生成一个信号（tim_tixfpy），该信号可用作从模式控制器的触发输入，也可用作捕获命令。该信号先进行预分频（ICxPS），而后再进入捕获寄存器。



输出阶段生成一个中间波形作为基准：OCxRef（高电平有效）。链的末端决定最终输出信号的极性。

图 20.14 捕获/比较通道（如：通道 1 输入部分）

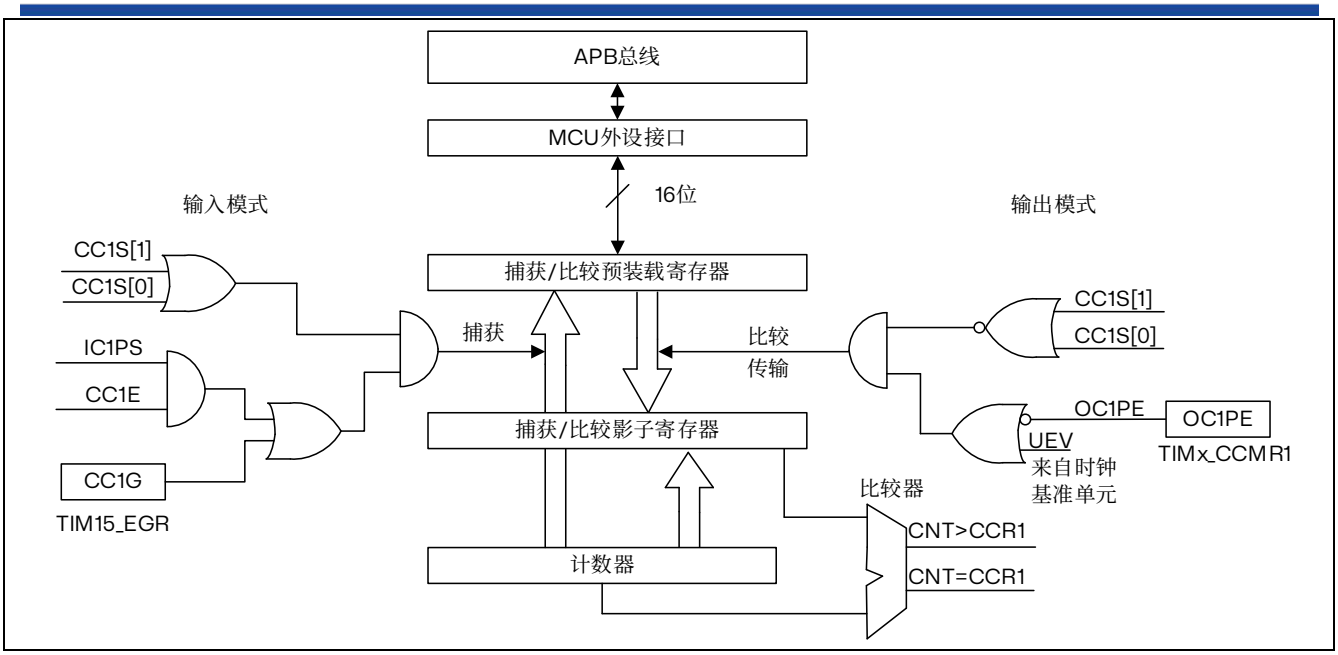


图 20.15 捕获/比较通道 1 的主电路

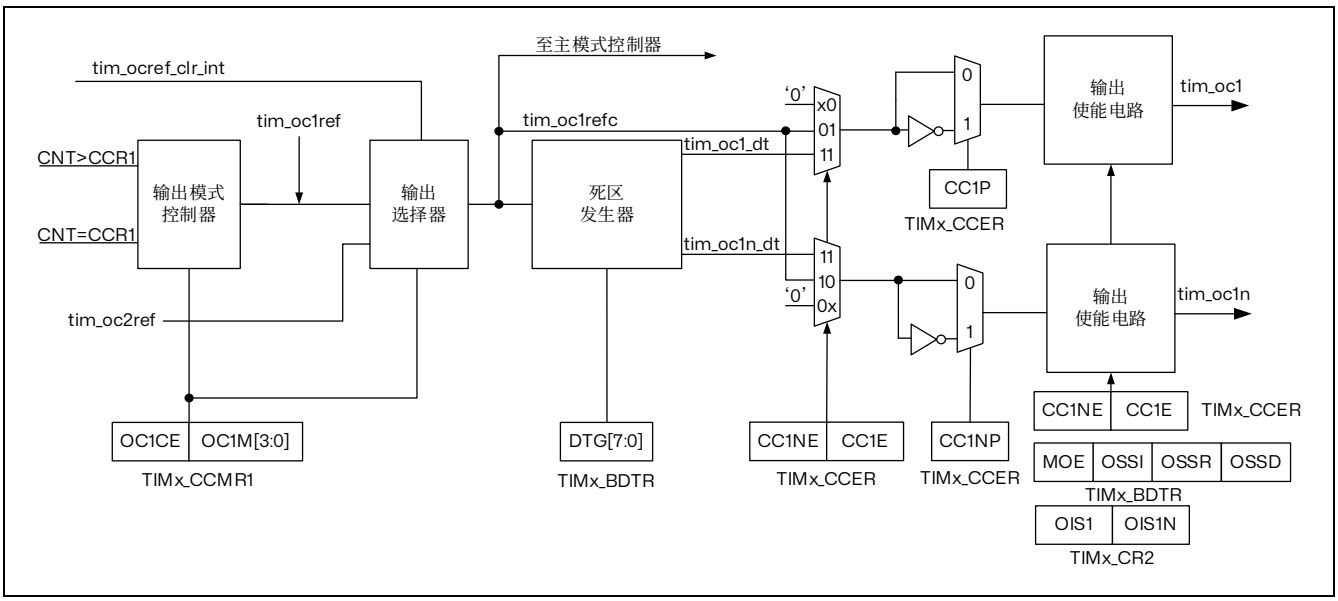


图 20.16 捕获/比较通道的输出阶段（通道 1）

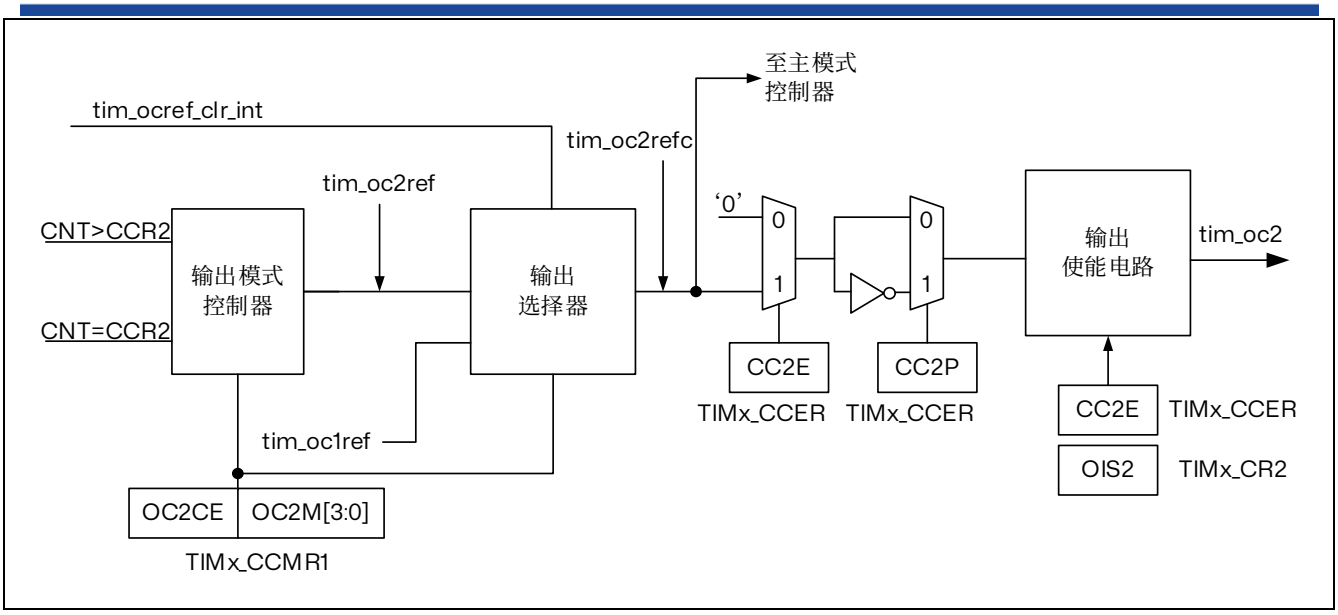


图 20.17 捕获比较通道的输出阶段（TIM15 的通道 2）

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。始终可通过读写操作访问预装载寄存器。

在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容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将复制到影子寄存器中，然后将影子寄存器的内容与计数器进行比较。

20.3.8 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器(TIMx_CCRx)来锁存计数器的值。发生捕获事件时，会将相应的 CCXIF 标志（TIMx_SR 寄存器）置 1，并可发送中断请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将重复捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件向 CCxIF 写入 0 来给 CCxIF 清零，或读取存储在 TIMx_CCRx 寄存器中的已捕获数据。向 CCxOF 写入 0 后会将其清零。

以下示例说明了如何在 tim_ti1 输入出现上升沿时将计数器的值捕获到 TIMx_CCR1 中。具体操作步骤如下：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:1]源（内部或外部）。
2. 选择有效输入：TIMx_CCR1 必须连接到 tim_ti1 输入，因此向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01。只要 CC1S 不等于 00，就会将通道配置为输入模式，并且 TIMx_CCR1 寄存器将处于只读状态。
3. 根据连接到定时器的信号，对所需的输入滤波时间进行编程（如果输入为 tim_tix 输入之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波时间设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 tim_ti1 上的跳变沿。然后向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。
4. 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 0，选择 tim_ti1 通道的有效转换边沿（本例中为上升沿）。
5. 对输入预分频器进行编程。在本例中，我们希望每次有效转换时都执行捕获操作，因此需要禁止预分频器（向 TIMx_CCMR1 寄存器中的 IC1PSC 位写入 00）。

6. 通过将 TIMx_CCER 寄存器中的 CC1E 位置 1，允许将计数器的值捕获到捕获寄存器中。
7. 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成中断。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

注：通过软件将 TIMx_EGR 寄存器中的相应 CCxG 位置 1 可生成 IC 中断请求。

20.3.9 PWM 输入模式

此模式允许测量连接到单个 tim_tix 输入的 PWM 信号的周期和占空比：

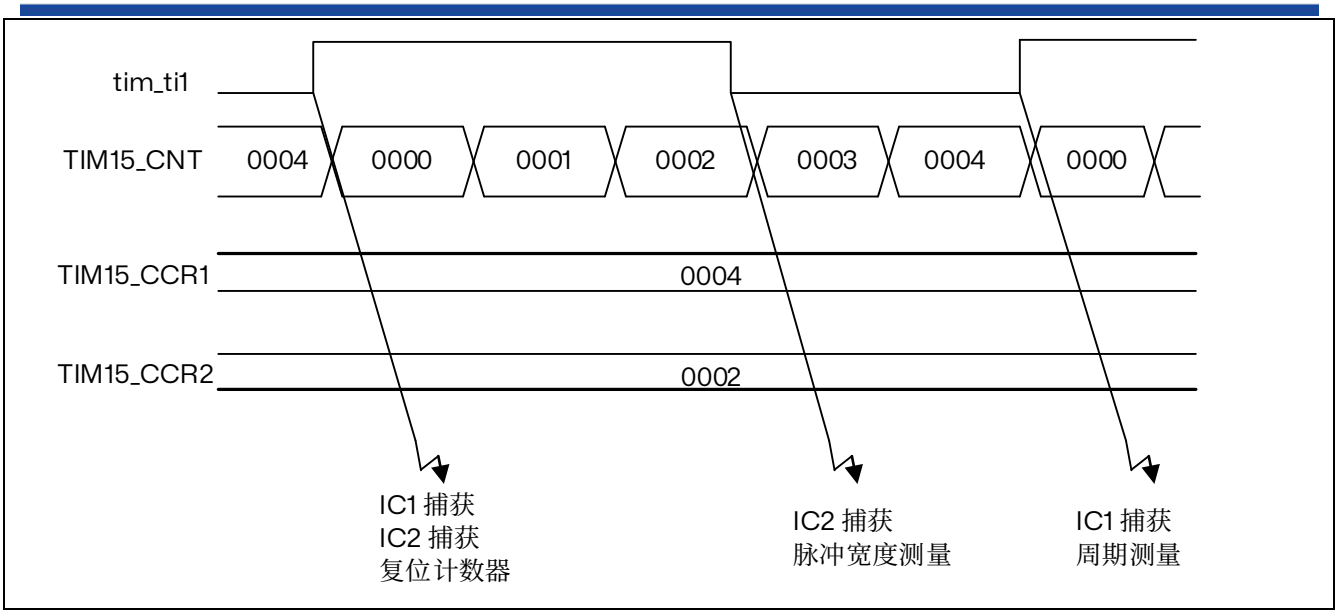
- TIMx_CCR1 寄存器保持周期值（两个连续上升沿之间的间隔）。
- TIMx_CCR2 寄存器保持脉冲宽度（两个连续上升沿和下降沿之间的间隔）。

此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 tim_icx 信号被映射至同一个 tim_tix 输入。
- 这两个 tim_icx 信号在边沿处有效，但极性相反。
- 选择两个 tim_tixfpy 信号之一作为触发输入，并将从模式控制器配置为复位模式。

例如，可通过以下步骤测量施加在 tim_ti1 上的 PWM 的周期（在 TIMx_CCR1 寄存器中）和占空比（在 TIMx_CCR2 寄存器中）（取决于 tim_ker_ck 频率和预分频器）：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:0]源（内部或外部）。
2. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 tim_ti1）。
3. 选择 tim_ti1fp1 的有效极性（用于 TIMx_CCR1 中的捕获和计数器清零）：向 CC1P 位和 CC1NP 位写入“0”（上升沿有效）。
4. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 tim_ti1）。
5. 选择 tim_ti1fp2 的有效极性（用于 TIMx_CCR2 中的捕获）：向 CC2P 位写入 1 和 CC2NP 位写入“0”（下降沿有效）。
6. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 101（选择 tim_ti1fp1）。
7. 将从模式控制器配置为复位模式：向 TIMx_SMCR 寄存器中的 SMS 位写入 100。
8. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。



PWM 输入模式只能与 TIM15_CH1/TIM15_CH2 信号一起使用，因为只有 tim_ti1fp1 和 tim_ti2fp2 连接到从模式控制器。

图 20.18 PWM 输入模式时序

20.3.10 强制输出模式

在输出模式（TIMx_CCMRx 寄存器中的 CCxS 位 = 00）下，可直接由软件将每个输出比较信号（tim_ocxref 和 tim_ocx）强制设置为有效电平或无效电平，而无需考虑输出比较寄存器和计数器之间的任何比较结果。

要将输出比较信号（tim_ocxref/tim_ocx）强制设置为有效电平，只需向相应 TIMx_CCMRx 寄存器中的 OCxM 位写入 101。tim_ocxref 进而强制设置为高电平（tim_ocxref 始终为高电平有效），同时 tim_ocx 获取 CCxP 极性位的相反值。

例如：CCxP=0（tim_ocx 高电平有效）=> tim_ocx 强制设置为高电平。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 100，可将 tim_ocxref 信号强制设置为低电平。

无论如何，TIMx_CCRx 影子寄存器与计数器之间的比较仍会执行，而且允许将标志置 1。因此可发送相应的中断请求。输出比较模式一节对此进行了介绍。

20.3.11 输出比较模式

此功能用于控制输出波形，或指示已经过某一时间段。

当捕获/比较寄存器与计数器之间相匹配时，输出比较功能：

- 将为相应的输出引脚分配一个可编程值，该值由输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）定义。匹配时，输出引脚既可保持其电平（OCXM=000），也可设置为有效电平（OCXM=001）、无效电平（OCXM=010）或进行翻转（OCxM=011）。
- 将中断状态寄存器中的标志置 1（TIMx_SR 寄存器中的 CCxIF 位）。
- 如果相应中断使能位（TIMx_DIER 寄存器中的 CCXIE 位）置 1，将生成中断。

使用 TIMx_CCMRx 寄存器中的 OCxPE 位，可将 TIMx_CCRx 寄存器配置为带或不带预装载寄存器。

在输出比较模式下，更新事件 UEV 对 tim_ocxref 和 tim_ocx 输出毫无影响。同步的精度可以达到计

数器的一个计数周期。输出比较模式也可用于输出单脉冲（在单脉冲模式下）。

步骤：

1. 选择计数器时钟（内部、外部、预分频器）。
2. 在 TIMx_ARR 和 TIMx_CCRx 寄存器中写入所需数据。
3. 如果要生成中断请求，将 CCxIE 位和/或 CCxDE 位置 1。
4. 选择输出模式。例如，
   a) 写入 OCxM=0011 以在 CNT 匹配 CCRx 时翻转 tim_ocx 输出引脚
   b) 写入 OCxPE=0 以禁用预装载寄存器
   c) 写入 CCxP=0 以选择高电平有效
   d) 写入 CCxE=1 以使能输出
5. 通过将 TIMx_CR1 寄存器中的 CEN 位置 1 来使能计数器。

可随时通过软件更新 TIMx_CCRx 寄存器以控制输出波形，前提是未使能预装载寄存器（OCxPE=0，否则仅当发生下一个更新事件 UEV 时，才会更新 TIMx_CCRx 影子寄存器）。

下图列出了相关示例。

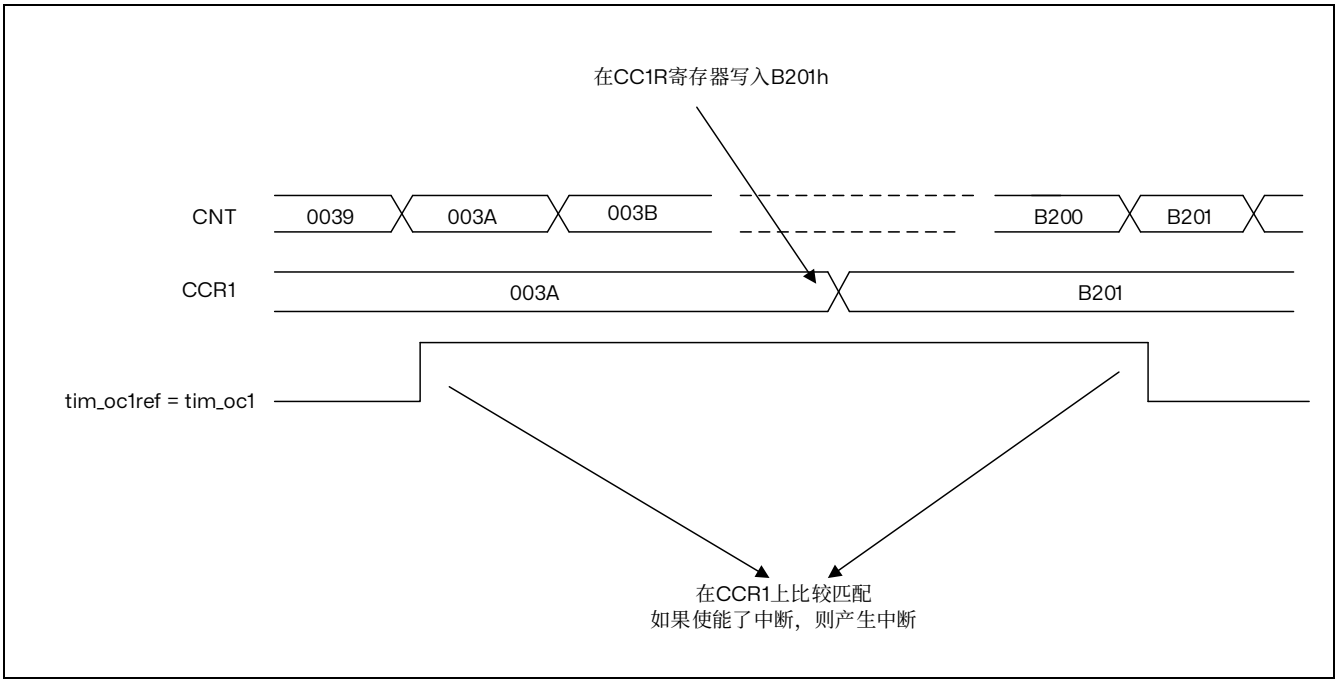


图 20.19 输出比较模式，翻转 tim_oc1

20.3.12 PWM 模式

脉冲宽度调制模式可以生成一个信号，该信号频率由 TIMx_ARR 寄存器值决定，其占空比则由 TIMx_CCRx 寄存器值决定。

通过向 TIMx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），可以独立选择各通道（每个 tim_ocx 输出对应一个 PWM）的 PWM 模式。必须通过将 TIMx_CCMRx 寄存器中的 OCxPE 位置 1 使能相应预装载寄存器，最后通过将 TIMx_CR1 寄存器中的 ARPE 位置 1 使能自动重载预装载寄存器。

由于只有在发生更新事件时预装载寄存器才会传送到影子寄存器，因此启动计数器之前，必须通过将 TIMx_EGR 寄存器中的 UG 位置 1 来初始化所有寄存器。

tim_ocx 极性可使用 TIMx_CCER 寄存器的 CCxP 位来编程。既可以设为高电平有效，也可以设为低电平有效。通过 CCxE、CCxNE、MOE、OSSI、OSSR 和 OSSD 位（TIMx_CCER 和 TIMx_BDTR 寄存器）的组合使能 tim_ocx 输出。有关详细信息，请参见 TIMx_CCER 寄存器说明。

在 PWM 模式（1 或 2）下，TIMx_CNT 始终与 TIMx_CCRx 进行比较，以确定是 TIMx_CCRx ≤ TIMx_CNT 还是 TIMx_CNT ≤ TIMx_CCRx（取决于计数器计数方向）。

TIM15 只能递增计数，具体请参考递增计数模式。

以下以 PWM 模式 1 为例。只要 TIMx_CNT < TIMx_CCRx，PWM 参考信号 tim_ocxref 便为高电平，否则为低电平。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR 中），则 tim_ocxref 保持为“1”。如果比较值为 0，则 tim_ocxref 保持为“0”。下图举例介绍边沿对齐模式的一些 PWM 波形 (TIMx_ARR=8)。

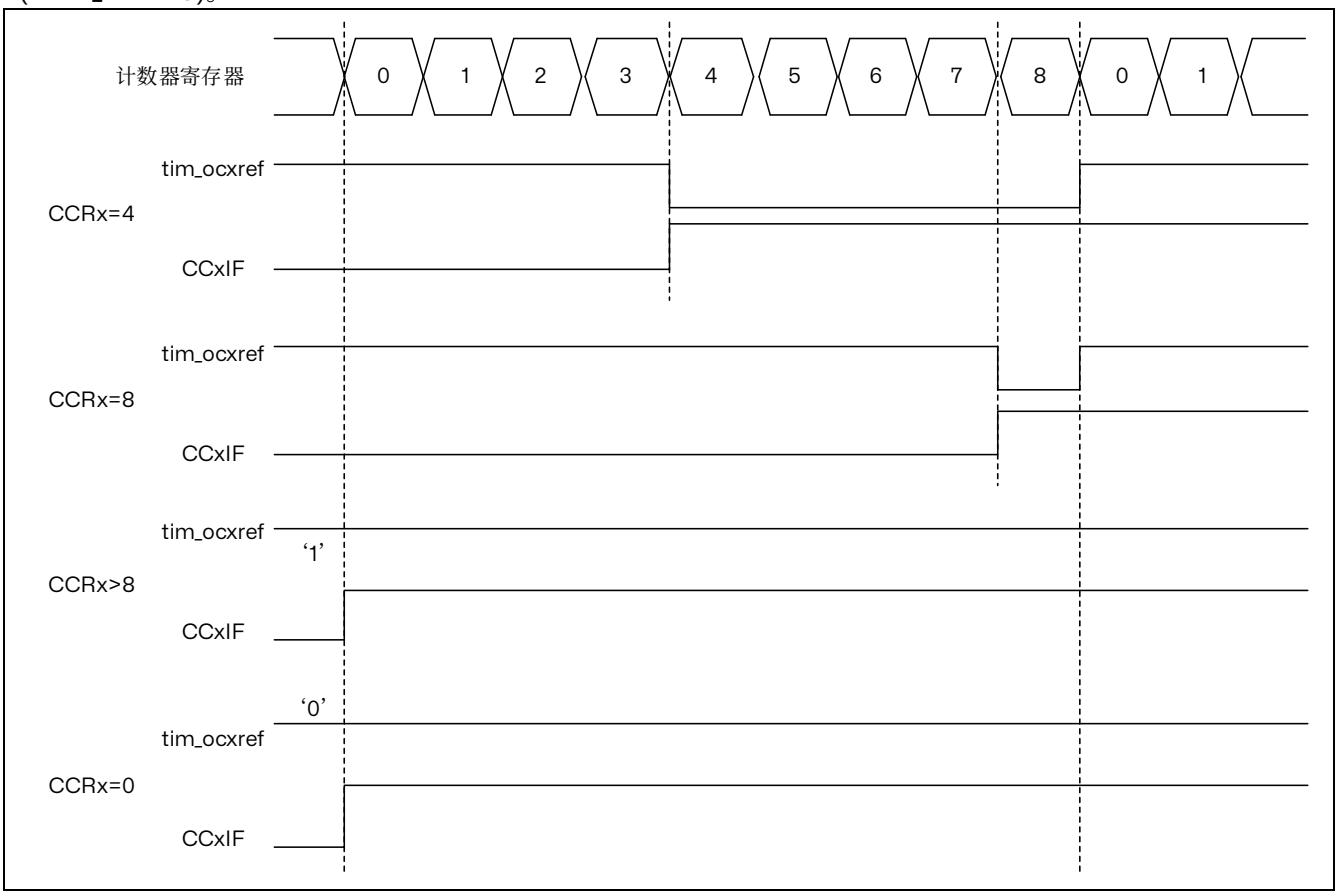


图 20.20 边沿对齐的 PWM 波形（ARR=8）

抖动模式

通过使能抖动模式，使用 TIMx_CR1 寄存器中的 DITHEN 位，可以提高 PWM 模式的有效分辨率。这既适用于 CCR（用于增加占空比分辨率），也适用于 ARR（用于增加 PWM 频率分辨率）。

工作原理是在 16 个连续的 PWM 周期中，以预定义的模式使实际的 CCR（或 ARR）值略有变化（增加或减少一个定时器时钟周期）。这使得占空比或 PWM 期间的平均值提高了 16 倍分辨率。下图展示了应用于 4 个连续 PWM 周期的抖动原理。

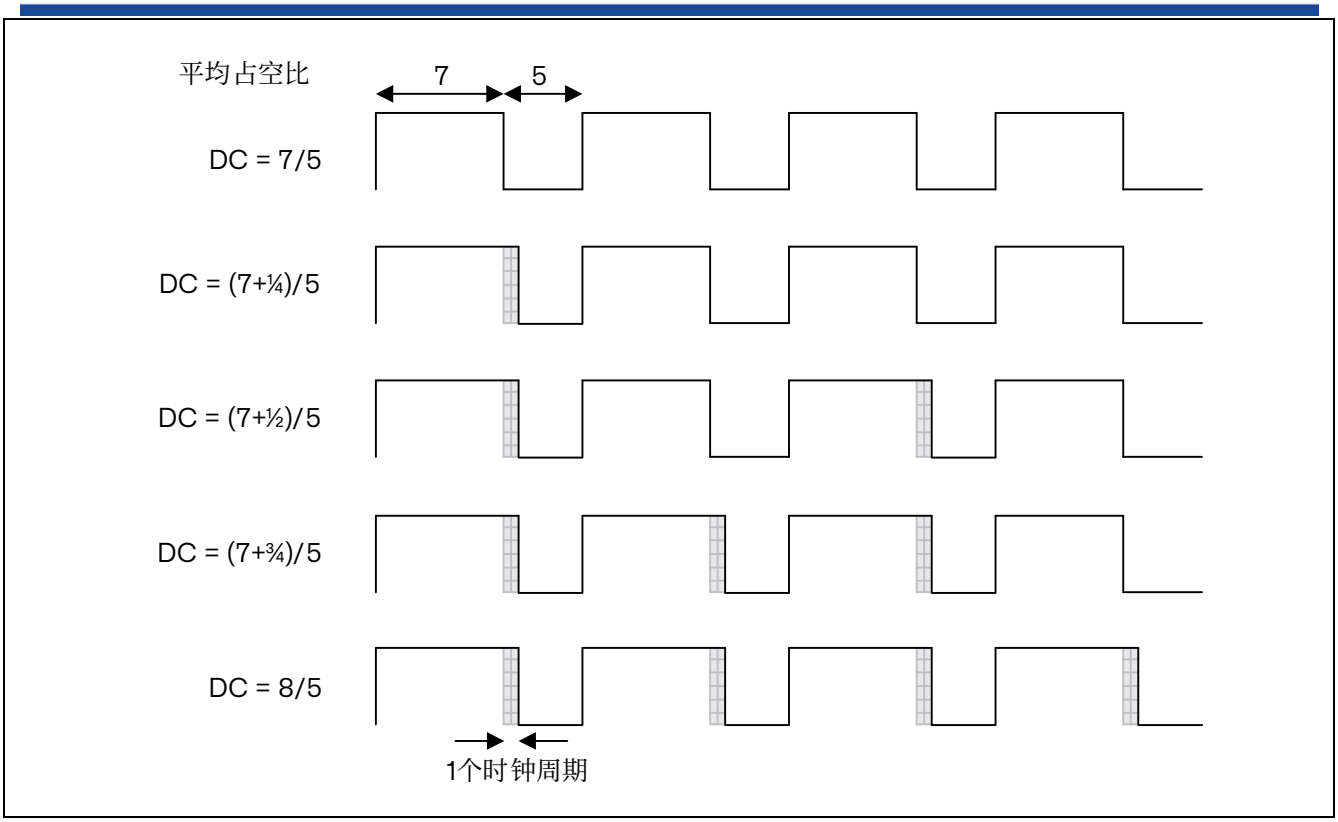


图 20.21 抖动原理

当抖动模式被使能时，寄存器编码将如下所示进行变化（参见下图示例）

- 4 个最低有效位（LSB）用于编码增强的分辨率部分（小数部分）
- 最高有效位（MSB）左移至 19:4 位，用于编码基本值

*注意：如果设置/清除*DITHEN *位，*ARR *和*CCR *值将自动更新。（例如，如果*ARR=0x05 *且*DITHEN=0，*它将更新为*ARR=0x50 *且*DITHEN=1*）。*

1. 必须重置 CEN 和 ARPE 位
2. ARR[3:0]位必须重置
3. DITHEN 位必须重置
4. CCIF 标志必须清除
5. 可以设置 CEN 位（最终设置 ARPE=1）。

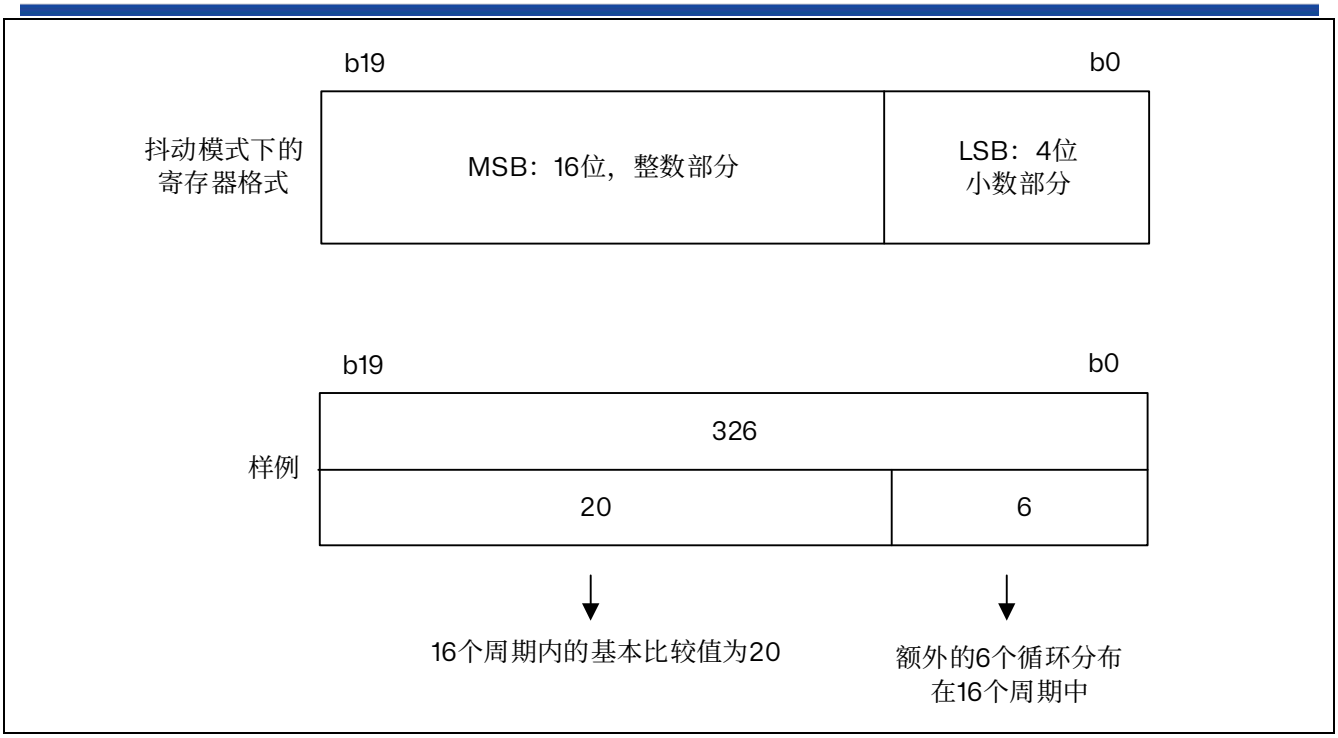


图 20.22 抖动模式下的数据格式和寄存器编码

最小频率由以下公式给出：

$$\text{分辨率} = \frac{\text{FTim}}{\text{Fpwm}} \Rightarrow \text{FpwmMin} = \frac{\text{FTim}}{\text{最大分辨率}}$$

$$\text{禁用抖动模式：FpwmMin} = \frac{\text{FTim}}{65536}$$

$$\text{使能抖动模式：FpwmMin} = \frac{\text{FTim}}{65535 + \frac{15}{16}}$$

*注意：在抖动模式下，*TIMx_ARR *和*TIMx_CCRy *的最大值限制为*0xFFFEF *（对应于整数部分的*65534 *和抖动部分的*15*）。*

如下图所示，无论 PWM 频率如何，抖动模式都可以增加 PWM 分辨率。

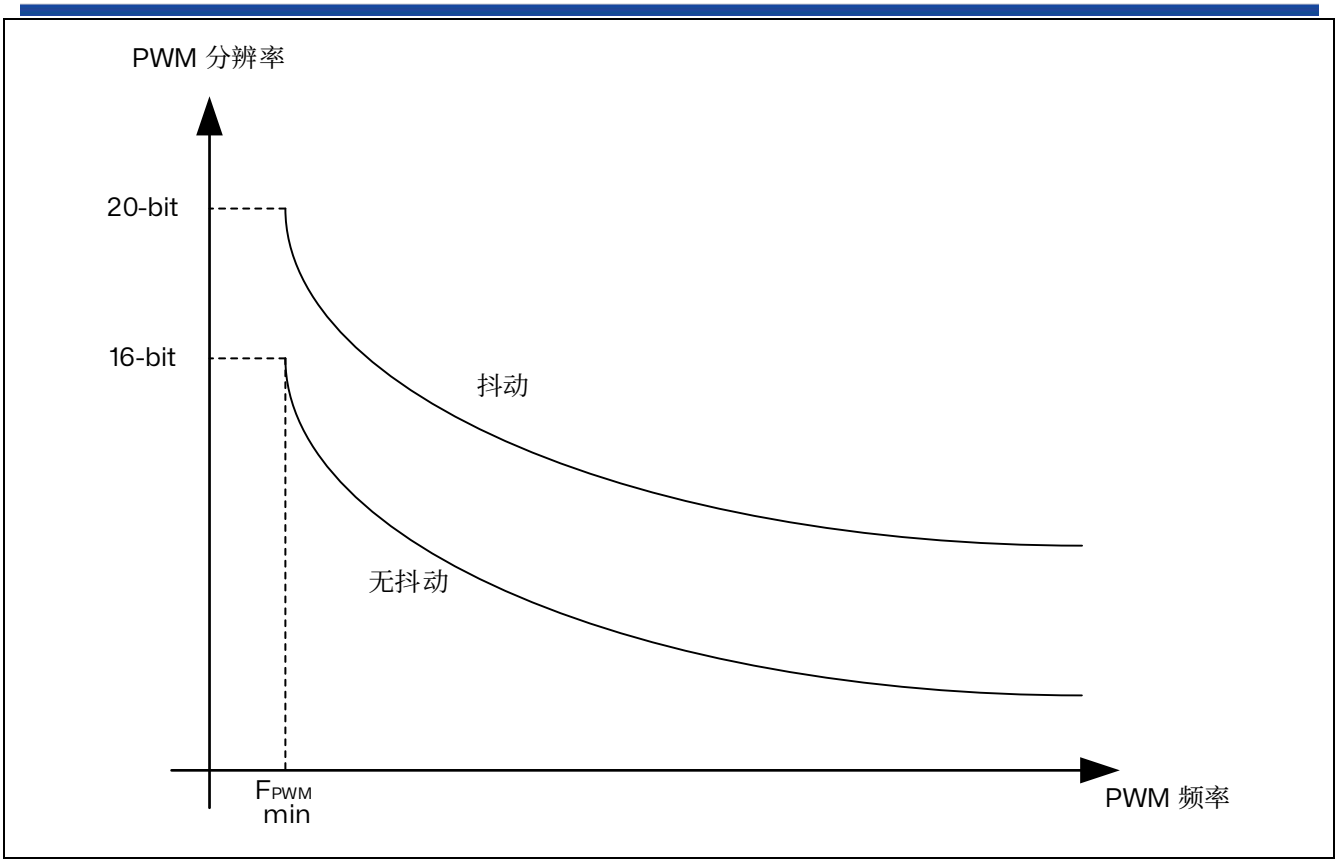


图 20.23 PWM 分辨率 vs 频率

如下图所示，占空比和/或周期的变化分散在 16 个连续周期。

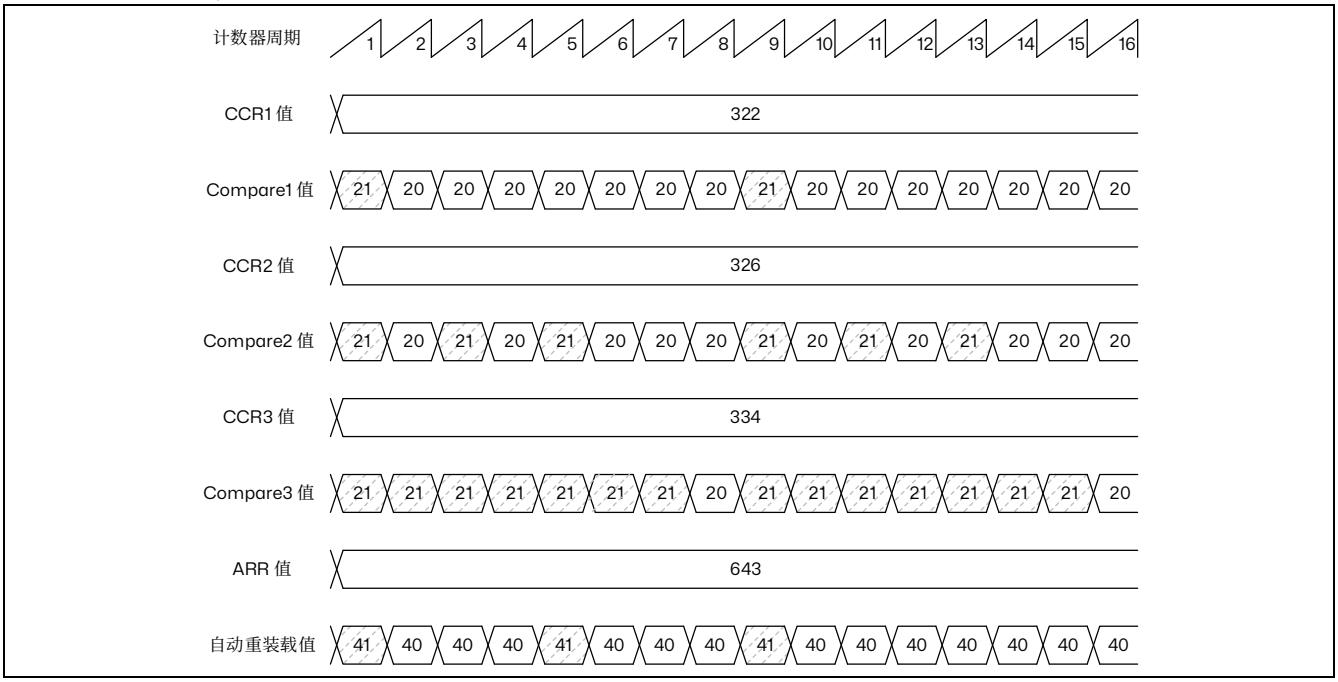


图 20.24 PWM 抖动模式

自动重新加载和比较值的增量按照下表中描述的特定模式分布。抖动序列是为了使增量分布尽可能均匀并最大限度地减少整体波动。

表 20.7 CCR 和 ARR 寄存器的变化抖动模式

LSB 值	PWM 周期															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0001	+1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0010	+1	-	-	-	-	-	-	-	+1	-	-	-	-	-	-	-
0011	+1	-	-	-	+1	-	-	-	+1	-	-	-	-	-	-	-
0100	+1	-	-	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0101	+1	-	+1	-	+1	-	-	-	+1	-	-	-	+1	-	-	-
0110	+1	-	+1	-	+1	-	-	-	+1	-	+1	-	+1	-	-	-
0111	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	-	-
1000	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1001	+1	+1	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-	+1	-
1010	+1	+1	+1	-	+1	-	+1	-	+1	+1	+1	-	+1	-	+1	-
1011	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	-	+1	-
1100	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1101	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	-	+1	+1	+1	-
1110	+1	+1	+1	+1	+1	+1	+1	-	+1	+1	+1	+1	+1	+1	+1	-
1111	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-

20.3.13 组合 PWM 模式

组合 PWM 模式允许生成两个边缘对齐或中心对齐的 PWM 信号，各自脉冲之间的可编程延迟和相位偏移。虽然频率由 TIMx_ARR 寄存器的值决定，但占空比和延迟由两个 TIMx_CCRx 寄存器确定。结果信号 tim_ocxrefc 由两个参考 PWM 的 OR 或 AND 逻辑组合而成：

— tim_oc1refc（或 tim_oc2refc）由 TIMx_CCR1 和 TIMx_CCR2 控制

通过在 TIMx_CCMRx 寄存器的 OCxM 位中写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2），可以在两个通道上独立选组合 PWM 模式（每个 CCR 寄存器对一个 tim_ocx 输出）。

当某个通道用作组合 PWM 通道时，其互补通道必须配置为相反的 PWM 模式（例如，一个通道为组合 PWM 模式 1，另一个通道为组合 PWM 模式 2）。

*注意：为了兼容性原因，*OCxM[3:0]*位域被分为两个部分，最高有效位与最低*3 *位不连续。*

下图表示使用组合 PWM 模式生成信号的示例，该示例的配置如下：

— 通道 1 配置为组合 PWM 模式 2。

— 通道 2 配置为 PWM 模式 1。

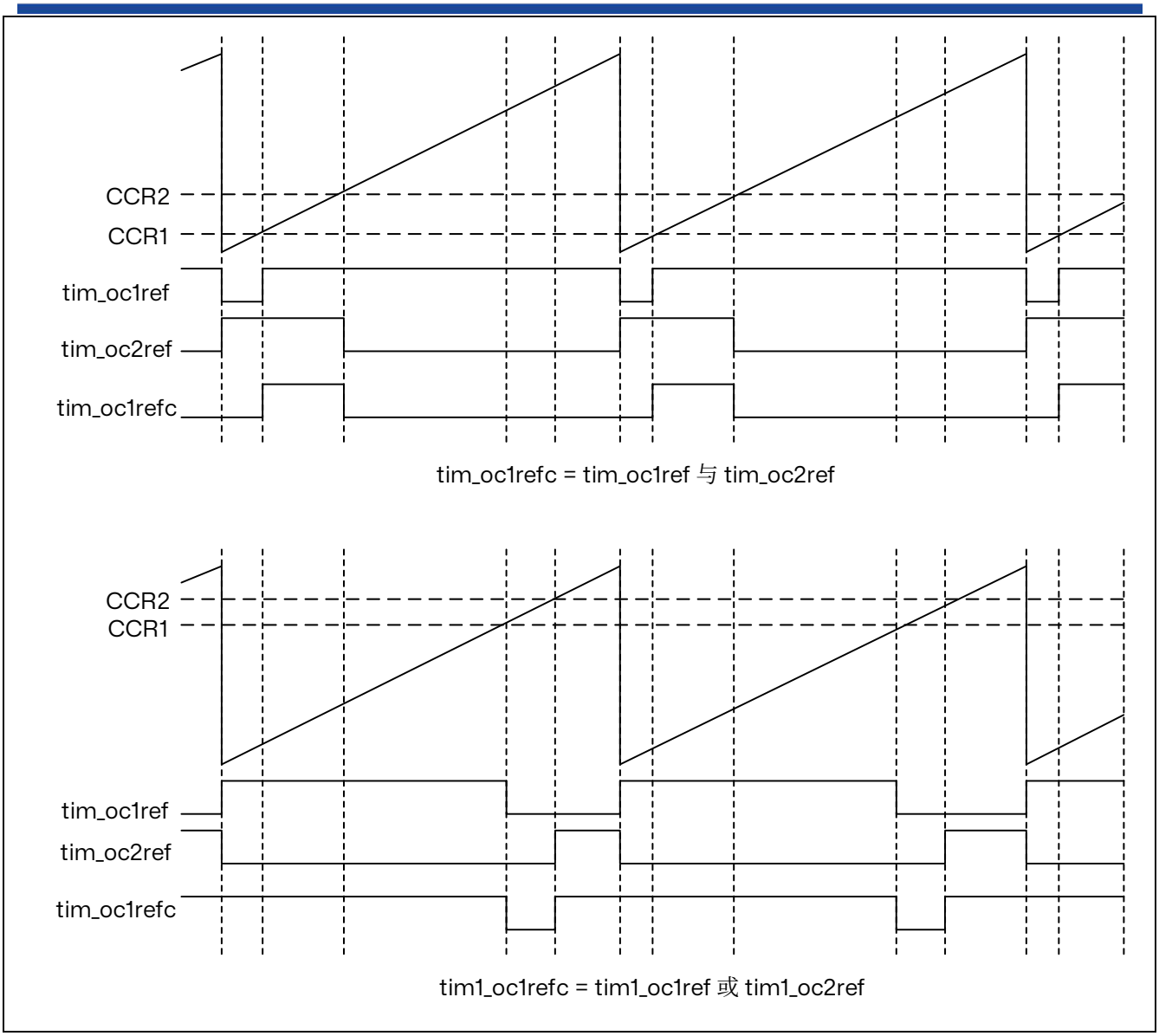


图 20.25 通道 1 和通道 2 的组合 PWM 模式

20.3.14 互补输出和死区插入

通用定时器（TIM15）可以输出一路互补信号，并管理输出的关断与接通瞬间。

这段时间通常称为死区，用户必须根据与输出相连接的器件及其特性（电平转换器的固有延迟、开关器件产生的延迟...）来调整死区时间

每路输出可以独立选择输出极性（主输出 tim_ocx 或互补输出 tim_ocxn）。可通过对 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位执行写操作来完成极性选择。

互补信号 tim_ocx 和 tim_ocxn 通过以下多个控制位的组合进行激活：TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位以及 TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI、OSSR 和 OSSD 位。更多详细信息，请参考表格：带刹车功能的互补输出通道 tim_oc1 和 tim_oc1n 的控制位。应当注意，切换至 idle（MOE 下降到 0）的时刻，死区仍然有效。

CCxE 和 CCxNE 位同时置 1 并且 MOE 位置 1（如果存在刹车）时，将使能死区插入。每个通道都有一个 10 位的死区时间生成器，将基于参考波形 tim_ocxref 生成 2 个输出 tim_ocx 和 tim_ocxn。如果

tim_ocx 和 tim_ocxn 为高电平有效：

- 输出信号 OCx 与参考信号相同，只是其上升沿相对参考上升沿存在延迟。
- 输出信号 OCxN 与参考信号相反，并且其上升沿相对参考下降沿存在延迟。

如果延迟时间大于有效输出（OCx 或 OCxN）的宽度，则不会产生相应的脉冲。

下图所示为死区发生器的输出信号与参考信号 OCxREF 之间的关系。（在这些示例中，假定 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

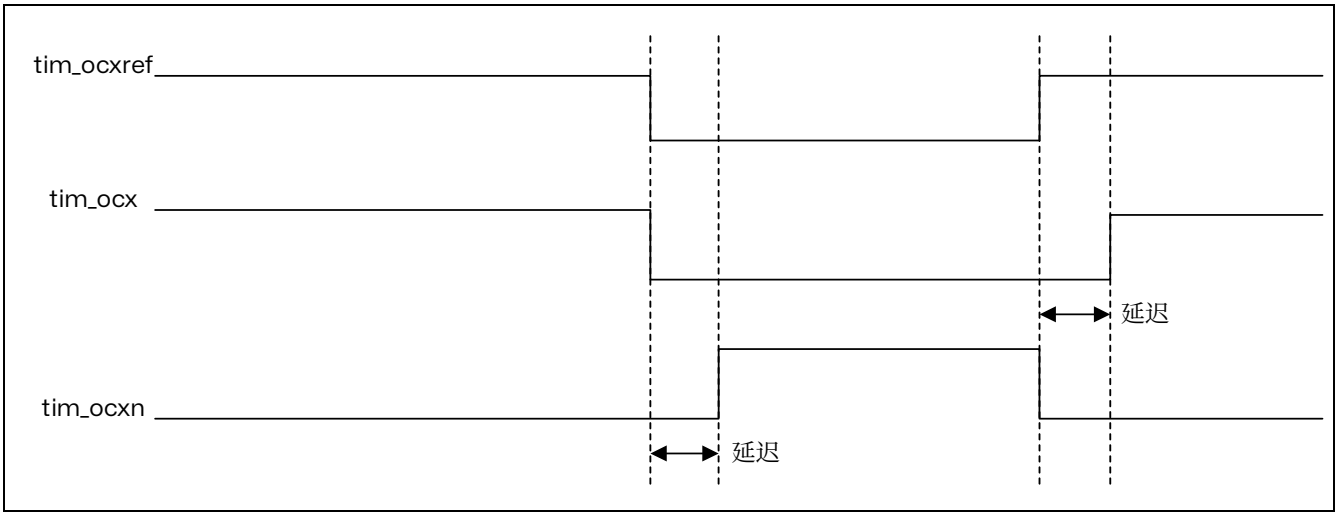


图 20.26 带死区插入的互补输出

TIMx_DTR2 中的 DTAE 位允许对参考信号上升沿和下降沿的死区值进行区分，如下图所示。

在非对称模式（DTAE=1）下，上升沿参考死区由 TIMx_BDTR 寄存器中的 DTG[7:0]位域定义，而下降沿参考死区由 TIMx_DTR2 寄存器中的 DTGF[7:0]位域定义。在使能计数器之前，必须写入 DTAE 位，并且在 CEN=1 时不得修改。

使用预加载机制可以在 PWM 操作期间动态更新死区值。当 TIMx_DTR2 寄存器中的 DTPE 位被设置时，死区位域 DTG[7:0]和 DTGF[7:0]将被预加载。预加载值将在下一个更新事件上加载到活动寄存器中。

注：如果 DTPE *位在计数器使能时被使能。自上次更新以来写入的新值将被丢弃，并使用先前的值。*

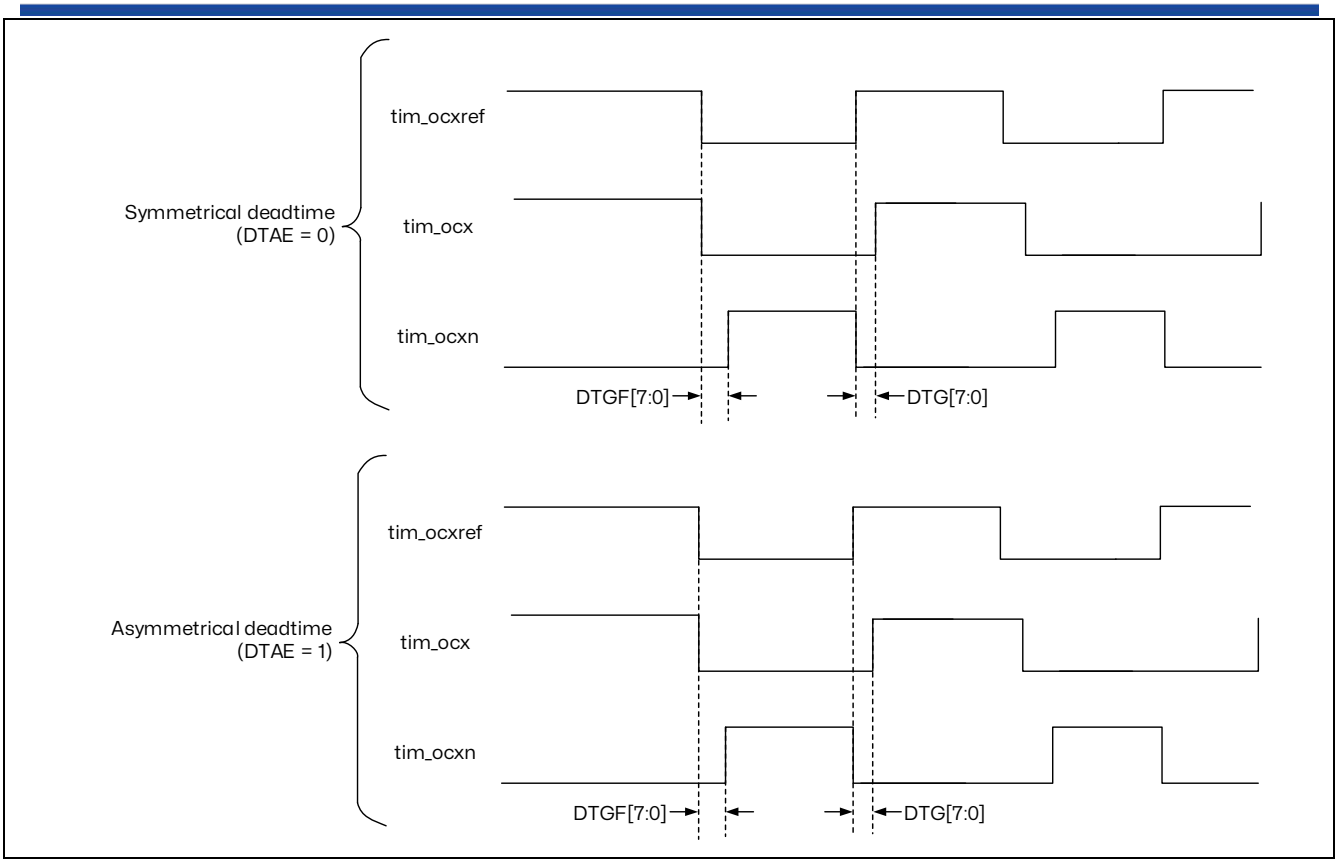


图 20.27 非对称死区时间

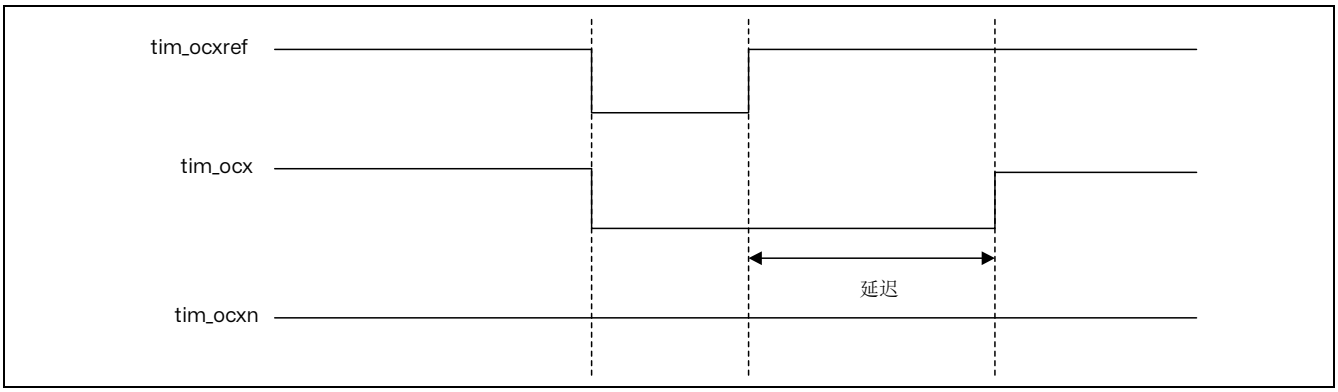


图 20.28 死区波形延迟大于负脉冲

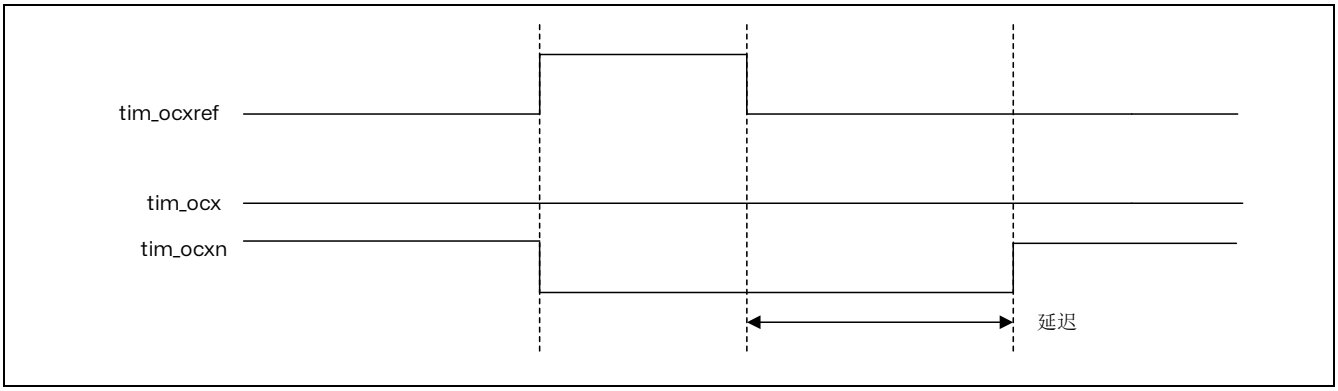


图 20.29 死区波形延迟大于正脉冲

死区延迟对于所有通道均相同，可通过 TIMx_BDTR 寄存器中的 DTG 位进行编程。有关延迟时间计算的信息，请参见后续章节：TIMx 刹车和死区寄存器(TIMx_BDTR)。

将 OCxREF 重定向到 OCx 或 OCxN

在输出模式（强制输出模式、输出比较模式或 PWM 模式）下，通过配置 TIMx_CCER 寄存器中的 CCxE 和 CCxNE 位，可将 tim_ocxref 重定向到 tim_ocx 输出或 tim_ocxn 输出。通过此功能，可以在一个输出上发送特定波形（如 PWM 或静态有效电平），而同时使互补输出保持其无效电平。或者，使两个输出同时保持无效电平，或者两个输出同时处于有效电平，两者互补并且带死区。

注：如果仅使能 tim_ocxn *（*CCxE=0, CCxNE=1*），两者不互补，一旦* tim_ocxref *为高电平，*tim_ocxn *即变为有效。例如，如果* CCxNP=0*，则*tim_ocxn= tim_ocxref*。另一方面，如果同时使能* tim_ocxn *和*tim_ocxref *（*CCxE=CCxNE=1*），*tim_ocx *在*tim_ocxref *为高电平时变为有效，而*tim_ocxn *则与之互补，在*tim_ocxref *为低电平时变为有效。*

20.3.15 使用刹车功能

刹车功能的目的是保护由定时器产生的 PWM 信号驱动的功率开关。刹车输入通常连接到功率级和三相逆变器的故障输出。当激活时，刹车电路会关闭 PWM 输出并强制它们进入预定义的安全状态。

刹车通道可以收集系统级故障（时钟故障……）和应用故障（来自输入引脚和内置比较器），并可以在死区时间后将输出强制到一个预定义的水平（无论是有效还是无效）。

刹车期间输出使能信号和输出电平取决于几个控制位：

- TIMx_BDTR 寄存器中的 MOE 位允许通过软件使能/禁用输出，并在发生刹车事件时重置。
- TIMx_BDTR 寄存器中的 OSSI 位定义了定时器是否在无效状态下控制输出或将其控制权释放给 GPIO 控制器（通常使其处于高阻模式）。
- TIMx_CR2 寄存器中的 OISx 和 OISxN 位用于设置输出关闭电平，可以是有效电平或无效电平。tim_oc1 和 tim_oc1n 输出不能同时设为有效电平。更多详细信息请参考后续表格：具有刹车功能的互补 tim_oc1 和 tim_oc1n 通道的输出控制位。

退出复位时，刹车电路被禁用，MOE 位为低电平。通过设置 TIMx_BDTR 寄存器中的 BKE 位，可以使能刹车功能。通过配置同一寄存器中的 BKP 位，可以选择刹车输入极性。BKE 和 BKP 可以同时修改。在写入 BKE 和 BKP 位后，需要等待 1 个 APB 时钟周期才能使写入生效。因此，在写入操作后，需要等待 1 个 APB 时钟周期才能正确地读取该位。

因为 MOE 下降沿可以是异步的，所以在实际的信号（作用于输出）和同步控制位（在 TIMx_BDTR 寄存器中访问）之间插入了一个重新同步电路。这导致异步信号和同步信号之间存在一些延迟。特别是当 MOE 被设置为 1 而之前它是低电平时，在正确读取之前必须插入一个延迟（伪指令）。这是因为写入作用于异步信号，而读取反映的是同步信号。

刹车是由 tim_brk 输入产生的，它具有：

- 可编程极性（TIMx_BDTR 寄存器中的 BKP 位）
- 可编程使能位（TIMx_BDTR 寄存器中的 BKE 位）
- 可编程滤波器（TIMx_BDTR 寄存器中的 BKF[3:0]位），以避免产生虚假事件。

可以使用 TIMx_AF1 寄存器，从多个源生成刹车，这些源可以单独使能，并具有可编程的边沿敏感度。

刹车（tim_brk）通道的源为：

- 连接到 TIMx_BKIN 引脚的外部源（根据 GPIO 备用功能选择寄存器中的选择而定），具有极性选择和可选的数字滤波

- 内部源：
  - 来自 tim_brk_cmpx 输入（请参考章节：TIM15 引脚和内部信号）
  - 来自 tim_sys_brk 输入的系统中断请求（请参考章节：TIM15 引脚和内部信号）

也可以通过软件使用 TIMx_EGR 寄存器中的 BG 位生成刹车事件。

所有源在进入定时器的 tim_brk 输入之前进行 OR 操作，如下图所示。

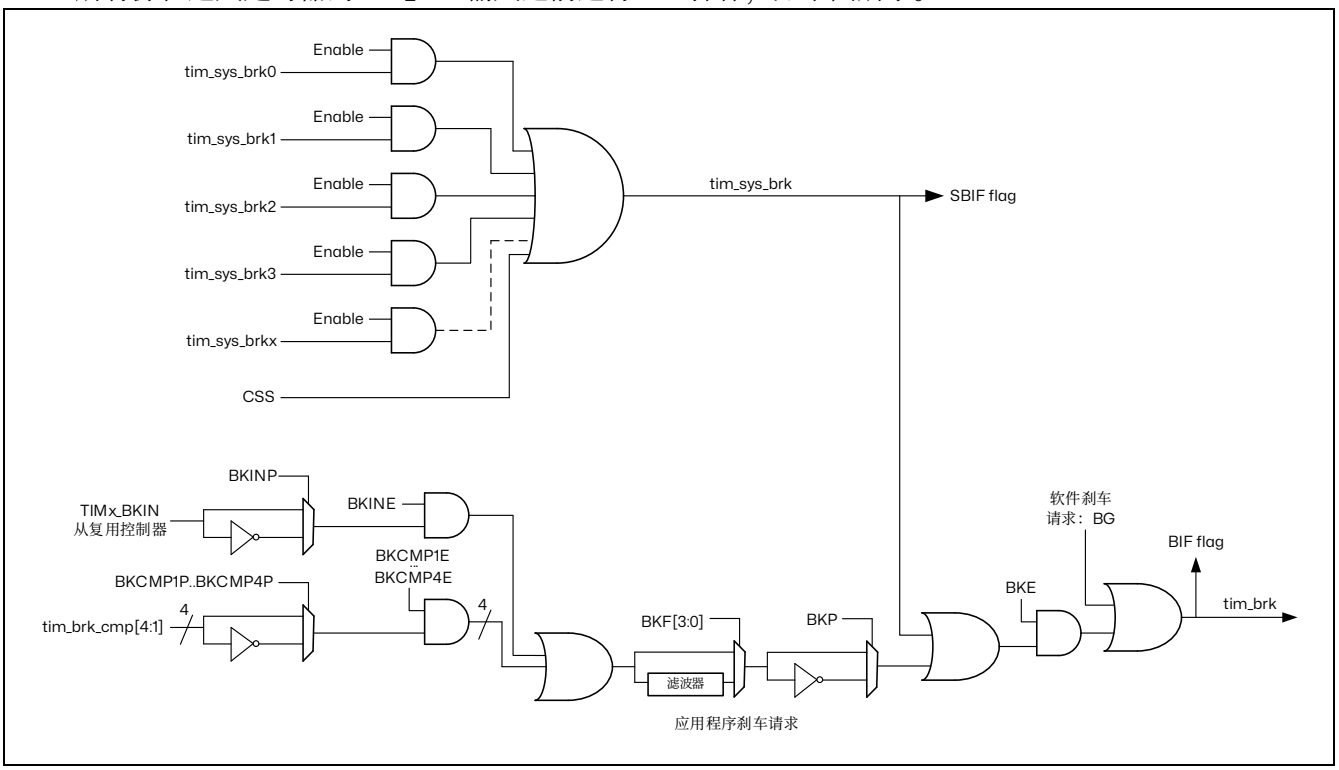


图 20.30 刹车电路概述

注：只有当可编程滤波器禁用时，才保证异步（无时钟）操作。如果使能，必须使用故障安全时钟模式（例如使用内部PLL 和/或CSS）来保证刹车事件得到处理。

当一个刹车发生时（在刹车输入上选择电平）：

- MOE 位被异步清除，使输出处于无效状态、空闲状态，甚至释放控制权给 GPIO（由 OSSI 位选择）。即使 MCU 振荡器关闭，此功能也可工作。
- 每当 MOE=0 时，每个输出通道都以 TIMx_CR2 寄存器中的 OISx 位中编程的电平驱动。如果 OSSI=0，定时器将释放输出控制（由 GPIO 接管），否则使能输出保持高电平。
- 当使用互补输出时：
  - 输出首先被置于复位状态（取决于极性），这是异步完成的，因此即使没有向定时器提供时钟也可以工作。
  - 如果定时器的时钟仍然存在，则重新激活死区时间发生器，以便在死区时间后驱动输出到编程在 OISx 和 OISxN 位中的电平。即使在这种情况下，tim_ocx 和 tim_ocxn 也不能同时驱动到有效电平。请注意，由于在 MOE 上进行了重新同步，死区时间持续时间比通常情况稍长（大约 2 个 tim_ker_ck 时钟周期）。
  - 如果 OSSI=0，则定时器释放使能输出（由 GPIO 接管，强制进入高阻态），否则使能输出在 CCxE 或 CCxNE 位中的任意一位变为高电平时立即变为高电平。
- 刹车状态标志（TIMx_SR 寄存器中的 BIF 位）被设置。如果 TIMx_DIER 寄存器中的 BIE 位被设置，则可以生成刹车。

- 如果 TIMx_BDTR 寄存器中的 AOE 位被设置，则 MOE 位将在下一个更新事件 UEV 处自动重新设置。这可以用于执行调节等操作。否则，MOE 保持低电平，直到再次写入 1。在这种情况下，它可以用于安全，刹车输入可以连接到来自功率驱动器、温度传感器或任何安全组件的警报。

注：如果 MOE *被* CPU *复位，而* AOE *位已设置，则输出处于空闲状态，并强制处于非活动电平或高阻态，具体取决于* OSSI *值。如果* MOE *和* AOE *位都被* CPU *复位，则输出处于禁用状态，并驱动在* TIMx_CR2 *寄存器中* OISx *位编程的电平。*

注：刹车输入在电平上有效。因此，当刹车输入处于有效状态时（无论是自动还是通过软件设置）， MOE *都无法设置。同时，状态标志* BIF *不能被清除。*

除刹车输入和输出管理外，刹车电路内部还实施了写保护，用以保护应用的安全。通过该功能，用户可冻结多个参数配置（死区持续时间、tim_ocx/tim_ocxn 极性和禁止时的状态、OCxM 配置、刹车使能和极性）。可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从 3 种保护级别中进行选择。请参考章节：TIMx 刹车和死区寄存器（TIMx_BDTR）。MCU 复位后只能对 LOCK 位执行一次写操作。

下图所示为输出对刹车响应行为的示例。

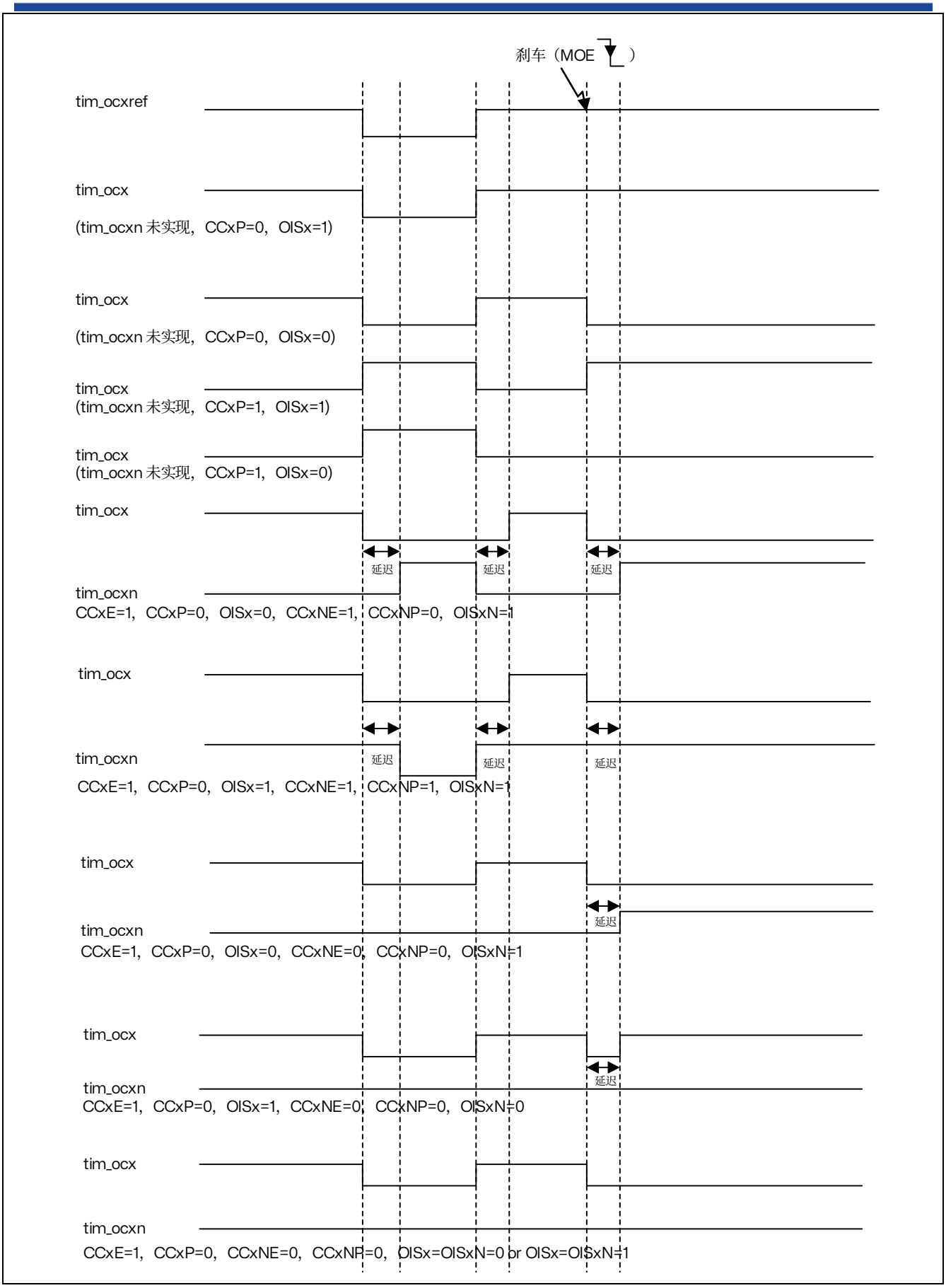


图 20.31 tim_brk 在发生刹车事件时的各种输出行为（OSSI=1）

20.3.16 双向刹车输入

TIM15 具有双向刹车 I/O，如下图所示。

这为以下方面提供支持：

- 板级全局刹车信号，用于向外部 MCU 或门驱动器发送故障信号，使用唯一的引脚作为输入和输出状态引脚
- 内部刹车源和多个外部开漏源 OR 连接在一起，以触发唯一的刹车事件，当多个内部和外部刹车源必须合并时

使用 TIMxBDTR 寄存器中的 BKBID 位配置 tim_brk 输入为双向模式。BKBID 编程位可以使用 TIMxBDTR 寄存器中的 LOCK 位锁定为只读模式（在 LOCK 级别 1 或更高）。

双向模式要求 I/O 配置为开漏模式，使用低电平有效极性（使用 BKINP、BKP 位）。来自系统（例如 CSS）、片上外设或刹车输入的任何刹车请求都会强制将刹车输入置于低电平，以指示故障事件。出于安全原因，如果极性位没有正确设置（高电平有效），则双向模式将被抑制。

刹车软件事件（通过设置 BG 位触发）也会强制将刹车 I/O 置为“0”，以指示外部组件定时器已进入刹车状态。然而，只有使能了刹车（BKE = 1）时，这才有效。当使用 BKE = 0 生成软件刹车事件时，输出将处于安全状态，并且会设置刹车标志，但 TIMx_BKIN I/O 没有影响。

安全解发机制可防止系统被彻底锁定（刹车输入低电平触发刹车，强制该输入为低电平）。

当 BKDSRM 位设置为 1 时，这将释放刹车输出以清除故障信号并允许重新给系统装载。

在任何时候都不能禁用刹车保护电路：

- 刹车输入路径始终处于有效状态：即使设置了 BKDSRM 位并且开放漏极控制被释放，刹车事件仍然处于有效状态。这可以防止 PWM 输出在刹车条件存在时重新启动。
- 只要输出使能（MOE 位被设置），BKDSRM 位就不能解除刹车保护（参见下表）

表 20.8 刹车保护解除条件

MOE	BKBID	BKDSRM	刹车保护状态
0	0	X	装载
0	1	0	装载
0	1	1	解除
1	X	X	装载

装载和重装载刹车电路

默认情况下（外设复位配置），刹车电路（输入或双向模式）处于装载状态。

在发生刹车事件后，必须按照以下步骤重新装载保护：

- BKDSRM 位必须设置为释放输出控制
- 软件必须等待系统刹车条件消失，并清除 SBIF 状态标志（或在重新装载之前系统地清除它）
- 软件必须轮询 BKDSRM 位，直到硬件清除该位（当应用程序刹车条件消失时）

从这一点开始，刹车电路处于装载和有效状态，并且可以设置 MOE 位以重新使能 PWM 输出。

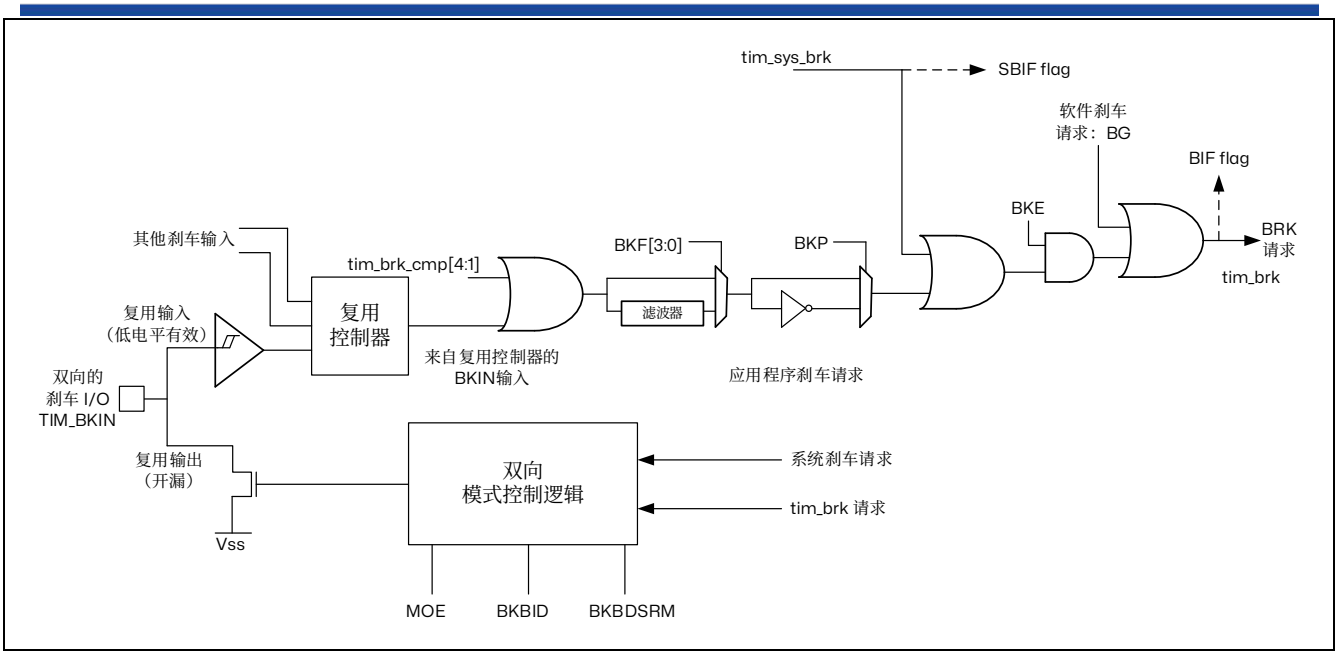


图 20.32 输出重定向

20.3.17 在外部事件时清除 tim_ocxref 信号

对于给定通道，在 tim_ocref_clr_int 输入施加高电平（相应 TIMx_CCMRx 寄存器中的 OCxCE 使能位置“1”），可使 tim_ocxref 信号变为低电平。tim_ocxref 信号将保持低电平，直到发生下一更新事件（UEV）（特别注意，当 TIMx_RCR>0 时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。此功能仅能用于输出比较模式和 PWM 模式，而不适用于强制输出模式。

当配置组合 PWM 模式，对于 TIM15_CH1，在 tim_ocref_clr_int 输入施加高电平（相应 TIM15_CCMR1 寄存器中的 OC1CE 使能位置“1”），可使组合结果 tim_oc1refc 信号变为低电平。tim_oc1refc 信号将保持低电平，直到发生下一更新事件（UEV）（特别注意，当 TIM15_RCR>0 时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。

tim_ocref_clr 输入可以在多个输入中进行选择，如下图所示。

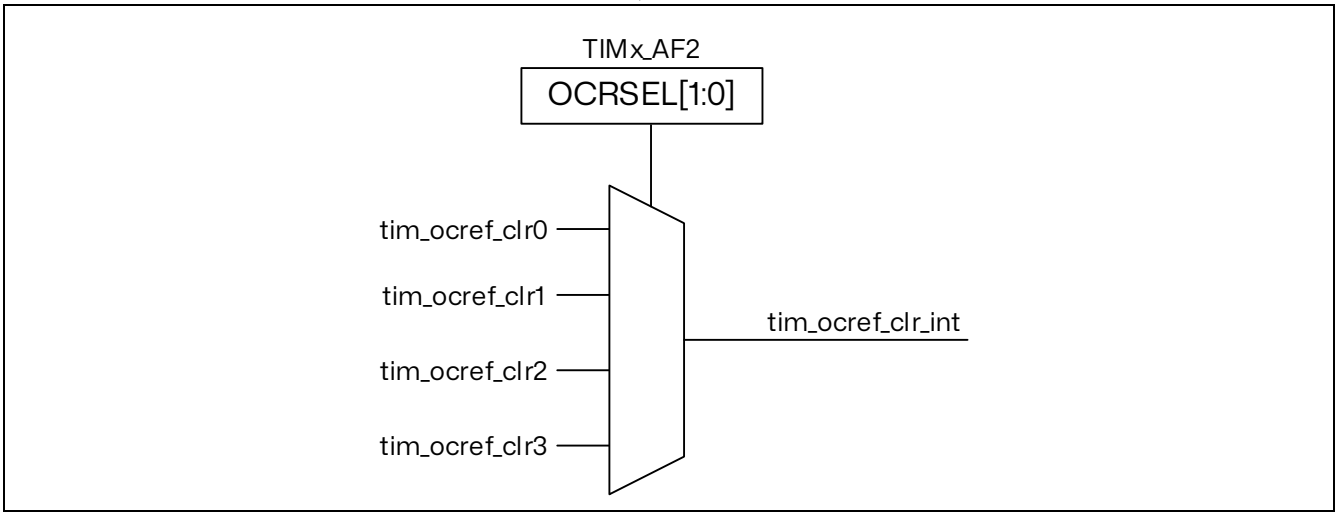


图 20.33 tim_ocref_clr 输入选择复用器

20.3.18 单脉冲模式

单脉冲模式（OPM）是上述模式的一个特例。在这种模式下，计数器可以在一个激励信号的触发下启动，并可在一段可编程的延时后产生一个脉宽可编程的脉冲。

可以通过从模式控制器启动计数器。可以在输出比较模式或 PWM 模式下生成波形。将 TIMx_CR1 寄存器中的 OPM 位置 1，即可选择单脉冲模式。这样，发生下一更新事件 UEV 时，计数器将自动停止。

只有当比较值与计数器初始值不同时，才能正确产生一个脉冲。启动前（定时器等待触发时），必须进行如下配置：

- CNT < CCRx ≤ ARR（特别注意，0<CCRx）

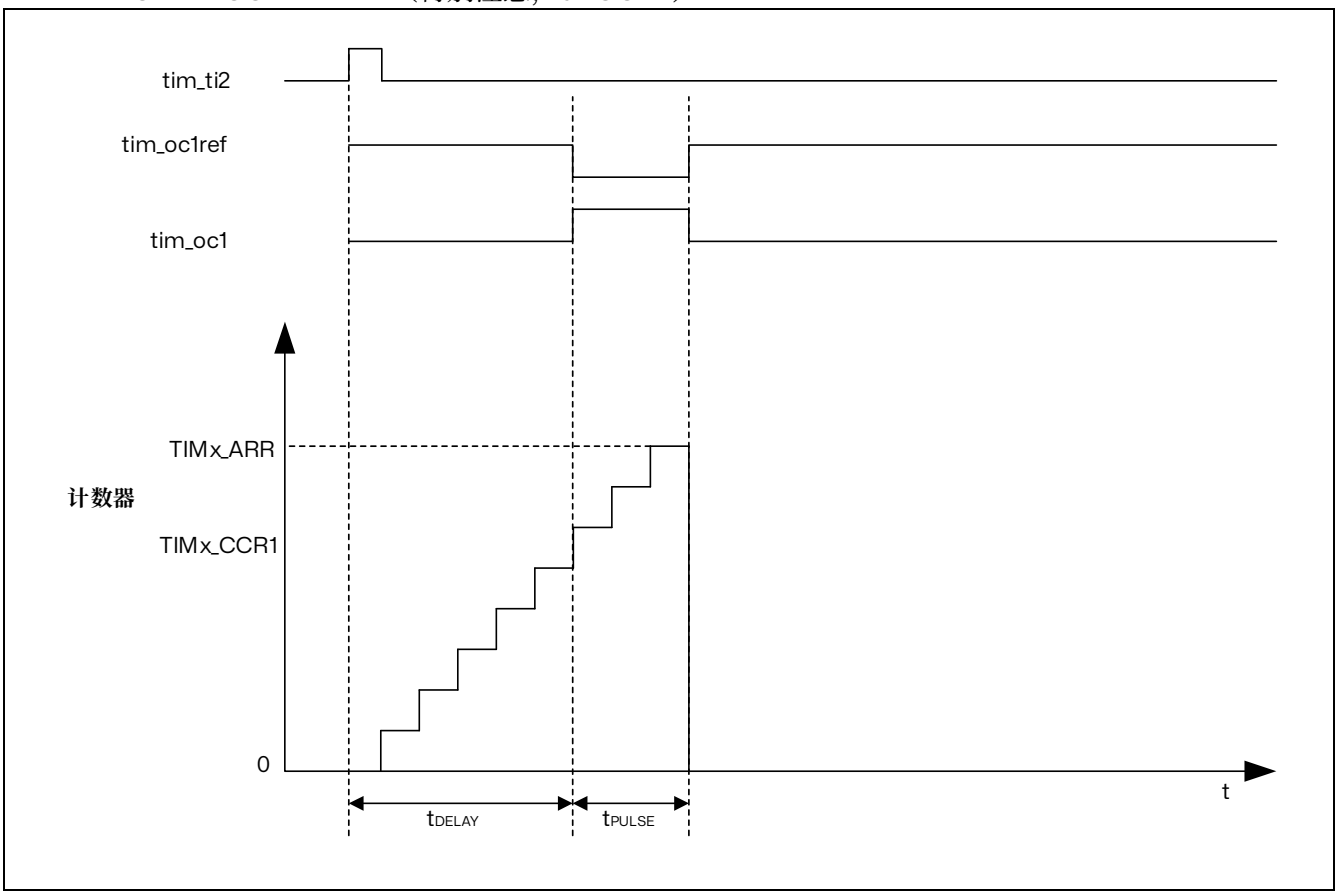


图 20.34 单脉冲模式的例子

例如，用户希望达到这样的效果：在 tim_ti2 输入引脚检测到正沿时，经过 t_{DELAY} 的延迟，在 tim_oc1 上产生一个长度为 t_{PULSE} 的正脉冲。

使用 tim_ti2fp2 作为触发 1:

1. 通过在 TIMx_TISEL 寄存器中使用 TI2SEL[2:0]位选择合适的 tim_ti2_in[15:0]源（内部或外部）。
2. 在 TIMx_CCMR1 寄存器中写入 CC2S=01，将 tim_ti2fp2 映射到 tim_ti2。
3. 在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=“0”，使 tim_ti2fp2 能够检测上升沿。
4. 在 TIMx_SMCR 寄存器中写入 TS=0110，将 tim_ti2fp2 配置为从模式控制器的触发(tim_trgi)。
5. 在 TIMx_SMCR 寄存器中写入 SMS=“110”（触发模式），使用 tim_ti2fp2 启动计数器。

OPM 波形通过对比较寄存器执行写操作来定义（考虑时钟频率和计数器预分频器）。

- t_{DELAY} 由写入 TIMx_CCR1 寄存器的值定义。
- t_{PULSE} 由自动重载值与比较值（TIMx_ARR - TIMx_CCR1）之差来定义。

- 假设希望产生这样的波形：信号在发生比较匹配时从“0”变为“1”，在计数器达到自动重载值时由“1”变为“0”。为此，应在 TIMx_CCMR1 寄存器中写入 OC1M=111，以使能 PWM 模式 2。如果需要，可选择在 TIMx_CCMR1 寄存器的 OC1PE 和 TIMx_CR1 寄存器的 ARPE 中写入“1”，以使能预装载寄存器。这种情况下，必须在 TIMx_CCR1 寄存器中写入比较值并在 TIMx_ARR 寄存器中写入自动重载值，通过将 UG 位置 1 来产生更新，然后等待 tim_ti2 上的外部触发事件。本例中，CC1P 的值为“0”。

在本例中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应为低。

由于仅需要 1 个脉冲（单脉冲模式），因此应向 TIMx_CR1 寄存器的 OPM 位写入“1”，以便在发生下一更新事件（计数器从自动重载值返回到 0）时使计数器停止计数。TIMx_CR1 寄存器中的 OPM 位置“0”时，即选择重复模式。

特例：OCx 快速使能：

在单脉冲模式下，TIx 输入的边沿检测会将 CEN 位置 1，表示使能计数器。然后，在计数器值与比较值之间发生比较时，将切换输出。但是，完成这些操作需要多个时钟周期，这会限制可能的最小延迟（t_{DELAY} 最小值）。

如果要输出延迟时间最短的波形，可以将 TIMx_CCMRx 寄存器中的 OCxFE 位置 1。这样会强制 tim_ocxref（和 tim_ocx）对激励信号做出响应，而不再考虑比较的结果。其新电平与发生比较匹配时相同。仅当通道配置为 PWM1 或 PWM2 模式时，OCxFE 才会起作用

20.3.19 可重触发单脉冲模式

此模式允许计数器在接收到刺激后开始计数，并生成一个具有可编程长度的脉冲，但与上一节中描述的非重触发单脉冲模式存在以下差异：

— 只要触发器发生，脉冲就会立即开始（没有可编程的延迟）

— 如果在先前触发的脉冲完成之前发生新的触发，则脉冲会被延长

定时器必须处于从模式，TIMx_SMCR 寄存器的 SMS[3:0]位设置为“1000”（组合重置+触发模式），OCxM[3:0]位设置为“1000”或“1001”，用于可重触发 OPM 模式 1 或 2。

如果定时器配置为递增计数模式，则相应的 CCRx 必须设置为 0（ARR 寄存器设置脉冲长度）。如果定时器配置为递减计数模式，CCRx 必须大于或等于 ARR。

注： OCxM[[3:0]*和* SMS[3:0]*位域由于兼容性的原因被分成两个部分，最高有效位与最低* 3 *位不是连续的。*OCxM *这种模式不能与中央对齐* PWM *模式一起使用。*TIMx_CR1 *寄存器中的* CMS[1:0]*必须设置为* 00*。*

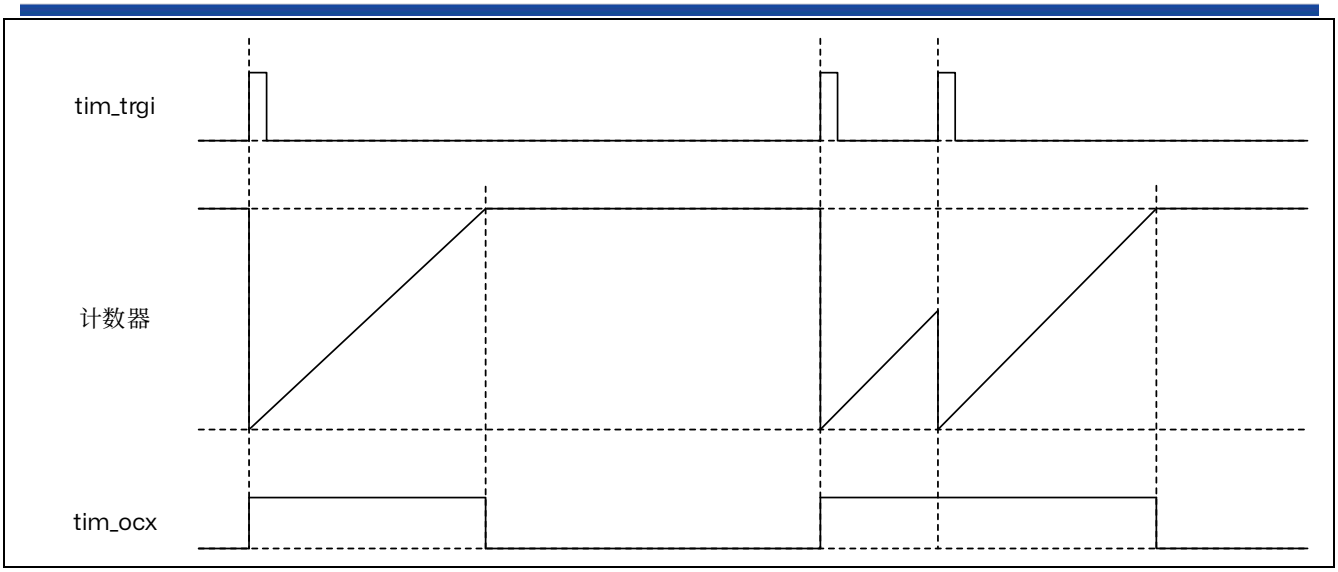


图 20.35 可重触发单脉冲模式

20.3.20 UIF 位重映射

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许以原子方式读取计数器的值和一个潜在的由 UIFCPY 标志指示的翻转条件。在特定情况下，它可以避免由背景任务（读取计数器）和中断（更新中断）之间的处理引起的竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间没有延迟。

20.3.21 定时器输入异或功能

TIM15_CR2 寄存器中的 TI1S 位允许将通道 1 的输入滤波器连接到异或门的输出，将两个输入引脚 tim_ti1 和 tim_ti2 组合起来。

异或输出可以与所有定时器输入功能一起使用，例如触发器或输入捕获。它对于测量两个输入信号之间的边沿间隔非常有用，如下图所示。

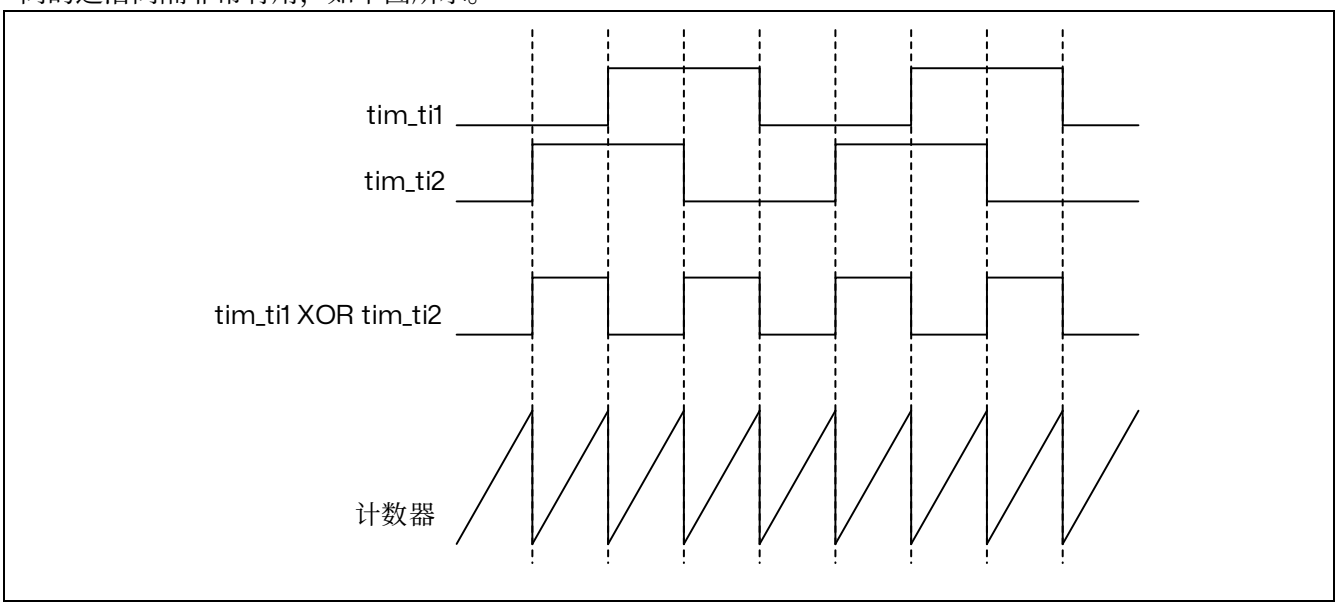


图 20.36 测量两个信号边沿之间的间隔时间

20.3.22 定时器和外部触发的同步

TIM 定时器在互相连接用于定时器的同步或链接。

TIM15 定时器以下列模式与外部触发实现同步：复位模式、门控模式、触发模式、复位+触发和门控+复位模式。

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 TIM15_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIM15_ARR，TIM15_CCRx）都被更新了。

在以下的例子中，tim_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 tim_ti1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIM15_CCMR1 寄存器中 CC1S=01。置 TIM15_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）。
- 置 TIM15_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM15_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIM15_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 tim_ti1 出现一个上升沿；当 tim_ti1 上升时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIM15_SR 寄存器中的 TIF 位）被设置，中断请求可以被发送（取决于 TIM15_DIER 寄存器中的 TIE 位）。

下图显示当自动重装载寄存器 TIM15_ARR=0x36 时的动作。

在 tim_ti1 上升沿和计数器的实际复位之间的延时取决于 tim_ti1 输入端的重同步电路。

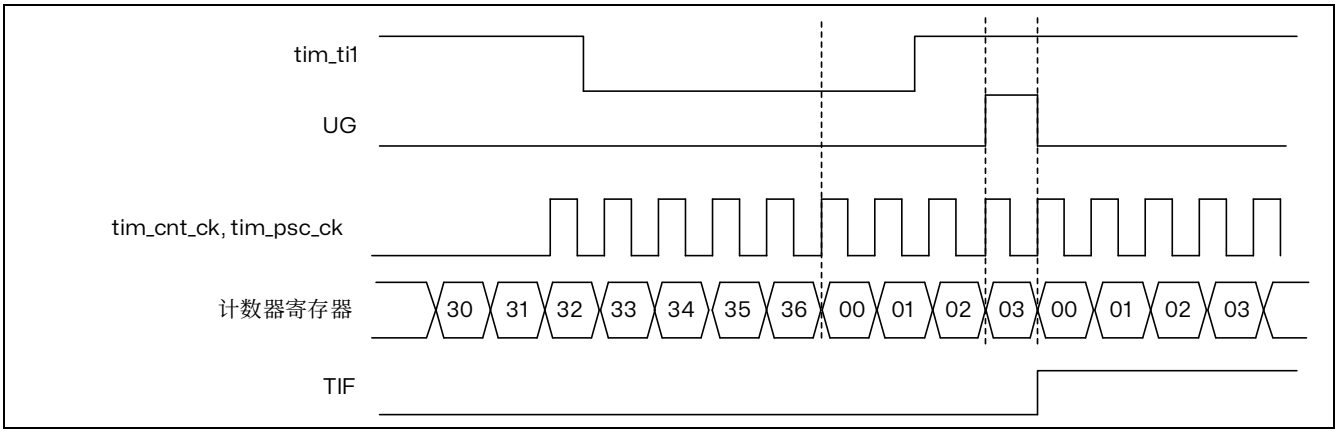


图 20.37 复位模式下的控制电路

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 tim_ti1 为低时向上计数：

- 配置通道 1 以检测 tim_ti1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIM15_CCMR1 寄存器中 CC1S=01。置 TIM15_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 置 TIM15_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM15_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIM15_CR1 寄存器中 CEN=1，启动计数器（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）。

只要 tim_ti1 为低，计数器开始依据内部时钟计数，一旦 tim_ti1 变高则停止计数。当计数器开始或停止时都设置 TIM15_SR 中的 TIF 标置。

tim_ti1 上升沿和计数器实际停止之间的延时取决于 tim_ti1 输入端的重同步电路。

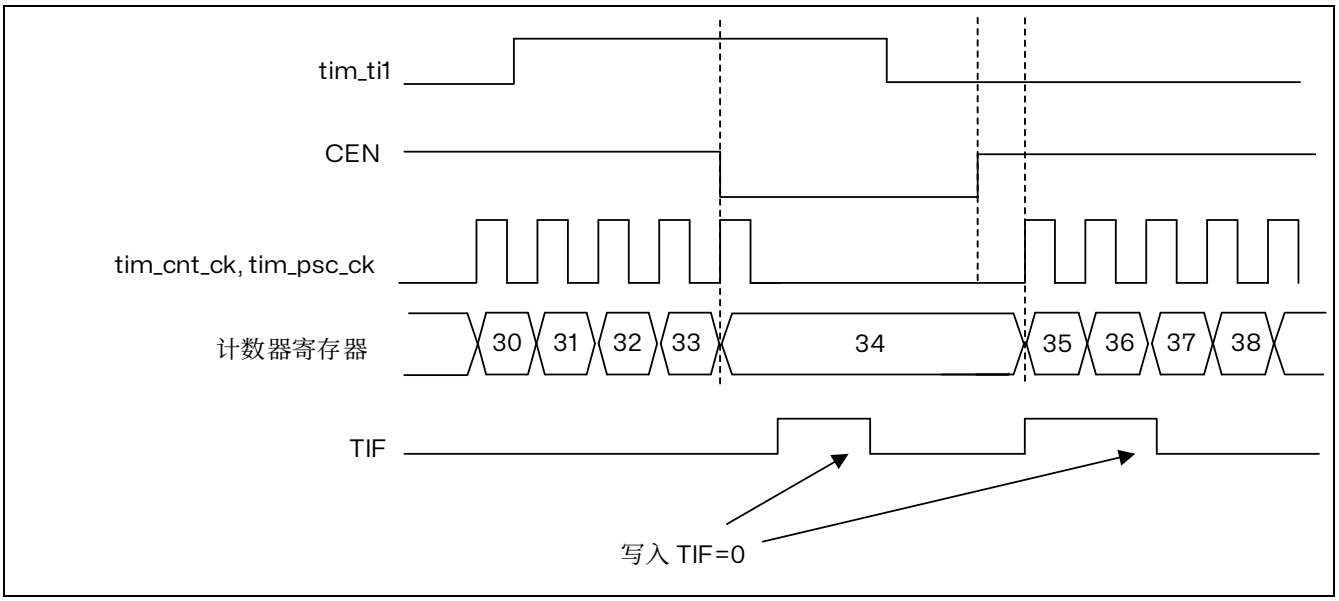


图 20.38 门控模式下的控制电路

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 tim_ti2 输入的上升沿开始向上计数：

- 配置通道 2 检测 tim_ti2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIM15_CCMR1 寄存器中 CC2S=01。置 TIM15_CCER 寄存器中 CC2P=1 和 CC2NP=0 以确定极性（只检测低电平）。
- 置 TIM15_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM15_SMCR 寄存器中 TS=0110，选择 tim_ti2 作为输入源。

当 tim_ti2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

tim_ti2 上升沿和计数器启动计数之间的延时，取决于 tim_ti2 输入端的重同步电路。

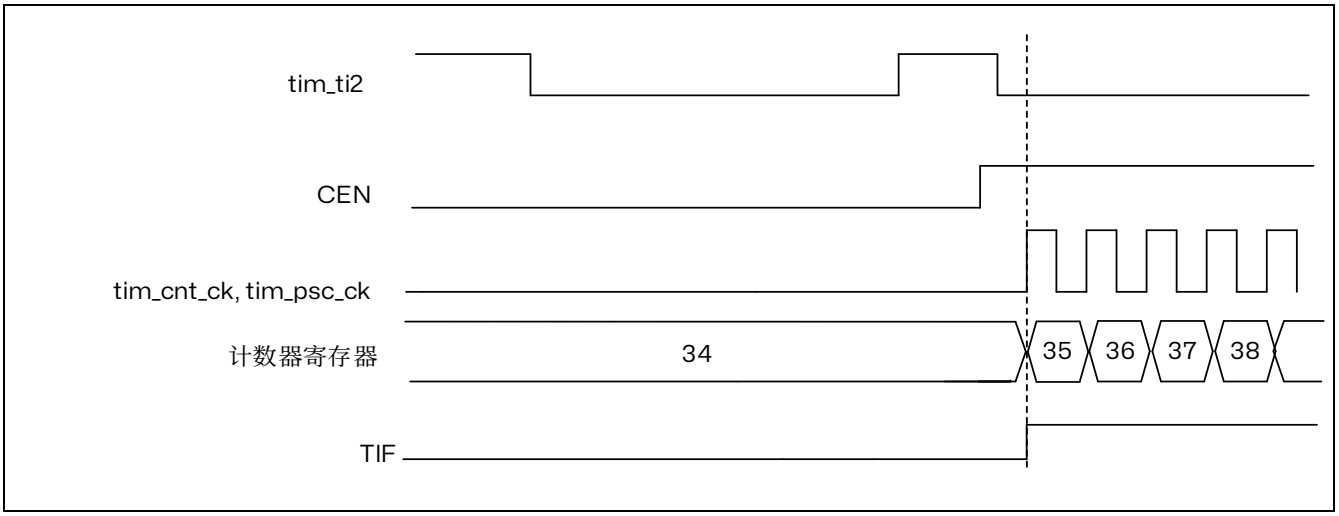


图 20.39 触发器模式下的控制电路

对实时更新编码器模式的从模式选择预装载

SMS[3:0]位可以预加载。这是通过在 TIM15_SMCR 寄存器中设置 SMSPE 使能位来实现的。从 SMS[3:0]预加载到有效值的触发器是计数器溢出时发生的更新事件（UEV）。

从模式 - 组合复位 + 触发模式

在这种情况下，选择的触发输入（tim_trgi）的上升沿重新初始化计数器，生成寄存器的更新，并启动计数器。

该模式用于单脉冲模式。

从模式 - 组合复位 + 门控模式

触发输入（tim_trgi）高时使能计数器。一旦触发变低，计数器停止并复位。计数器的开始和停止都是受控的。

该模式允许检测超出范围的 PWM 信号（占空比超过最大期望值）。

20.3.23 定时器同步

TIM15 定时器从内部连接在一起，以实现定时器同步或级联。

注：接收 tim_trgo 信号的从外设（定时器、ADC...）的时钟必须在主定时器接收事件前使能，并且在主定时器接收触发时，时钟频率（预分频器）不得动态更改。

20.3.24 ADC 触发

定时器可以使用各种内部信号（例如重置、使能或比较事件）生成 ADC 触发事件。

注意：接收 tim_trgo 信号的从属外设（定时器、ADC 等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能更改。

20.3.25 调试模式

当微控制器进入调试模式（内核停止）时，TIMx 计数器可以继续正常工作或停止。

在调试模式下，每个定时器的行为可以通过 Debug 支持（DBG）模块的专用配置位进行编程。

出于安全考虑，当计数器停止时，输出被禁用（就像将 MOE 位复位一样）。输出可以强制处于无效状态（OSSI 位=1），或者由 GPIO 控制器接管控制（OSSI 位=0）以强制其处于高阻态。

有关更多详细信息，请参阅调试部分。

20.4 TIM15 低功耗模式

表 20.9 低功耗模式对 TIM15 的影响

模式	描述
睡眠	无影响，外设是运行的。中断会导致设备退出睡眠模式。
停机	定时器操作被停止并保留其寄存器内容。无法生成中断。
待机	定时器被断电并且退出待机模式后必须重新初始化。

20.5 TIM15 中断

TIM15 可以生成多个中断，如下表所示。

表 20.10 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	从睡眠模式退出	从停机和待机退出
TIM	更新	UIF	UIE	UIF 写 0	是	否
	捕获/比较 1	CC1IF	CC1IE	CC1IF 写 0	是	否
	捕获/比较 2	CC2IF	CC2IE	CC2IF 写 0	是	否
	捕获/比较 3	CC3IF	CC3IE	CC3IF 写 0	是	否
	换向（COM）	COMIF	COMIE	COMIF 写 0	是	否
	触发	TIF	TIE	TIF 写 0	是	否
	刹车	BIF	BIE	BIF 写 0	是	否

20.6 TIM15 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。

20.6.1 TIM15 控制寄存器 1（TIM15_CR1）

偏移地址：0x000

复位值：0x0000

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

Bit 15:13 保留，必须保持为复位值。

Bit 12 **DITHEN**：使能抖动（Dithering enable）

0：抖动禁止

1：抖动使能

*注：*DITHEN *位只能在*CEN *位复位时修改*

Bit 11 **UIFREMAP**：UIF 状态位重映射（UIF status bit remapping）

0：无重映射。UIF 状态位未复制到 TIM15_CNT 寄存器位 31

1：重映射使能。UIF 状态位复制到 TIM15_CNT 寄存器位 31

Bit 10 保留，必须保持为复位值。

Bit 9:8 **CKD[1:0]**：时钟分频（Clock division）

此位域指示定时器时钟（tim_ker_ck）频率以及数字滤波器（tim_tix）所使用的采样时钟（t_{DTS}）之间的分频比

00：$t_{DTS} = t_{tim_ker_ck}$

01：$t_{DTS} = 2 \times t_{tim_ker_ck}$

10：$t_{DTS} = 4 \times t_{tim_ker_ck}$

11：保留，不要设置成此值

Bit 7 **ARPE**：自动重载预装载使能（Auto-reload preload enable）

0：TIM15_ARR 寄存器不进行缓冲；

1：TIM15_ARR 寄存器进行缓冲。

Bit 6:4 保留，必须保持为复位值。

Bit 3 **OPM**：单脉冲模式（One pulse mode）

0：计数器在发生更新事件时不会停止计数；

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

Bit 2 **URS**：更新请求源（Update request source）

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断。此类事件包括：

- 计数器上溢/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断。

Bit 1 **UDIS**：更新禁止（Update disable）

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新（UEV）事件可通过以下事件之一生成：

- 计数器上出/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

Bit 0 **CEN**：计数器使能（Counter enable）

0：禁止计数器；

1：使能计数器。

*注：只有事先通过软件将*CEN *位置*1*，才可以使用外部时钟、门控模式。而触发模式可通过硬件自动将*CEN *位置*1*。*

20.6.2 TIM15 控制寄存器 2（TIM15_CR2）

偏移地址：0x004

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			OIS3	Res	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			Res	CCUS	Res	CCPC
			rw		rw	rw	rw	rw	rw	rw	rw		rw		rw

Bit 15:13 保留，必须保持为复位值。

Bit 12 **OIS3**：输出空闲状态 3（OC3 输出）(Output idle state 3 (tim_oc3 output))

0：当 MOE=0 时，tim_oc3=0

1：当 MOE=0 时，tim_oc3=1

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 11 保留，必须保持为复位值。

Bit 10 **OIS2**：输出空闲状态 2（OC2 输出）(Output idle state 2 (tim_oc2 output))

0：当 MOE=0 时，tim_oc2=0

1：当 MOE=0 时，tim_oc2=1

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 9 **OIS1N**：输出空闲状态 1（OC1N 输出）(Output idle state 1 (tim_oc1n output))

0：当 MOE=0 时，经过死区时间后 tim_oc1n=0

1：当 MOE=0 时，经过死区时间后 tim_oc1n=1

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 8 **OIS1**：输出空闲状态 1（OC1 输出）(Output idle state 1 (tim_oc1 output))

0：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）tim_oc1=0

1：当 MOE=0 时，（如果 OC1N 有效，则经过死区时间之后）tim_oc1=1

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别* 1、2 *或*3，*此位即无法修改。*

Bit 7 **TI1S**：tim_ti1 选择（tim_ti1 selection）

0：tim_ti1_in[15:0]多路复用器输出连接到 tim_ti1_in 输入

1：tim_ti1_in[15:0]和 tim_ti2_in[15:0]多路复用器输出连接到 tim_ti1_in 输入（异或组合）

Bit 6:4 **MMS[2:0]：**主模式选择（Master mode selection）

这些位可选择主模式下将要发送到从定时器以实现同步的信息（tim_trgo）。这些位的组合如下：

000：**复位** – TIM15_EGR 寄存器中的 UG 位用作触发输出（tim_trgo）。如果复位由触发输入生成（从模式控制器配置为复位模式），则 tim_trgo 上的信号相比实际复位会有延迟。

001：**使能** – 计数器使能信号 CNT_EN 用作触发输出（tim_trgo）。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号可由 CEN 控制位产生。当配置为门控模式时，也可由触发输入产生。当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIM15_SMCR 寄存器中 MSM 位的说明）。

010：**更新** – 选择更新事件作为触发输出（tim_trgo）。例如，主定时器可用作从定时器的预分频器。

011：**比较脉冲** – 一旦发生输入捕获或比较匹配事件，当 CC1IF 被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(tim_trgo)。

100：**比较** – tim_oc1refc 信号用作触发输出（tim_trgo）

101：**比较** – tim_oc2refc 信号用作触发输出（tim_trgo）

注：若选 tim_ocxrefc *为* tim_trgo *输出，当* tim_ocref_clr_int *清零信号生效时，* tim_ocxrefc *信号将保持低电平，* tim_trgo *也不会有触发输出，直到发生下一更新事件（*UEV*）（特别注意，当* TIMx_RCR>0 *时，只有重复计数器达到零才会生成更新事件，波形才会恢复）。*

Bit 3 保留，必须保持为复位值。

Bit 2 **CCUS**：捕获/比较控制更新选择（Capture/compare control update selection）

0：如果捕获/比较控制位(CCPC=1)进行预装载，仅通过将 COMG 位置 1 来对这些位进行更新；

1：如果捕获/比较控制位(CCPC=1)进行预装载，可通过将 COMG 位置 1 或 TRGI 的上升沿对这些位进行更新。

注：此位仅对具有互补输出的通道有效。

Bit 1 保留，必须保持为复位值。

Bit 0 **CCPC**：捕获/比较预装载控制（Capture/compare preloaded control）

0：CCxE，CCxNE 和 OCxM 位未进行预装载；

1：CCxE，CCxNE 和 OCxM 位进行了预装载，写入这些位后，仅当发生换向事件(COM)（COMG 位置 1 或在 TRGI 上检测到上升沿，取决于 CCUS 位）时才会对这些位进行更新。

注：此位仅对具有互补输出的通道有效。

20.6.3 TIM15 从模式控制寄存器（TIM15_SMCR）

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											TS[3]	Reserved			SMS[3]
											rw				rw

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

Bit 31:21 保留，必须保持为复位值。

Bit 19:17 保留，必须保持为复位值。

Bit 15:8 保留，必须保持为复位值。

Bit 7 **MSM：**主/从模式（Master/slave mode）

0：不执行任何操作；

1：当前定时器的触发输入事件（tim_trgi）的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

Bit 20,6,5,4 **TS[3:0]：**触发选择（Trigger selection）

此位域可选择将要用于同步计数器的触发输入。

0000：内部触发 0（tim_itr0）

0001：内部触发 1（tim_itr1）

0010：内部触发 2（tim_itr2）

0011：内部触发 3（tim_itr3）

0100：tim_ti1 边沿检测器（tim_ti1f_ed）

0101：滤波后的定时器输入 1（tim_ti1fp1）

0110：滤波后的定时器输入 2（tim_ti1fp2）

0111：保留

1000：内部触发器 4 (tim_itr4)

1001：内部触发器 5 (tim_itr5)

其他：保留

有关定时器 ITRx 含义的详细信息，请参见章节：TIM15 引脚和内部信号。

*注：这些位只能在未使用的情况下（例如，*SMS=000 *时）进行更改，以避免转换时出现错误的边沿检测。*

Bit 3 保留，必须保持为复位值。

Bit 16, 2,1,0 **SMS[3:0]**：从模式选择（Slave mode selection）

当选择了外部信号，触发信号（tim_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）

0000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。

0001：保留

0010：保留

0011：保留

0100：**复位模式** – 选中的触发输入（tim_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。

0101：**门控模式** – 当触发输入（tim_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。

0110：**触发模式** – 计数器在触发输入 tim_trgi 的上升沿启动（但不复位），只有计数器的启动是受控的。

0111：**外部时钟模式 1** – 选中的触发输入（tim_trgi）的上升沿驱动计数器。

1000：**组合复位+触发模式** - 当选择的触发输入（tim_trgi）上升沿出现时，会重新初始化计数器，生成寄存器的更新并启动计数器。

1001：**组合门控+复位模式** - 当触发输入（tim_trgi）为高电平时，计数器时钟被使能。一旦触发变为低电平，计数器就会停止并重置。这样，计数器的启动和停止都得到了控制。

其他：保留

注：如果选择了 tim_ti1f_ed *作为触发输入（*TS=0100*），则不能使用门控模式。事实上，* tim_ti1f_ed *为每个* tim_ti1f *上的转换输出一个脉冲，而门控模式检查触发信号的电平。*

注：接收 tim_trgo *信号的从设备（定时器、*ADC *等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发时不能改变。*

20.6.4 TIM15 DMA/中断使能寄存器（TIM15_DIER）

偏移地址：0x00C

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BIE	TIE	COMIE	Res	CC3IE	CC2IE	CC1IE	UIE
								rw	rw	rw		rw	rw	rw	rw

Bit 15:8 保留，必须保持为复位值。

Bit 7 **BIE**：刹车中断使能（Break interrupt enable）

0：禁止刹车中断；

1：使能刹车中断。

Bit 6 **TIE**：触发信号（TGRI）中断使能（Trigger interrupt enable）

0：禁止触发信号（TGRI）中断；

1：使能触发信号（TGRI）中断。

Bit 5 **COMIE**：COM 中断使能（COM interrupt enable）

0：禁止 COM 中断；

1：使能 COM 中断。

Bit 4 保留，必须保持为复位值。

Bit 3 **CC3IE**：捕获/比较 3 中断使能（Capture/Compare 3 interrupt enable）

0：禁止 CC3 中断；

1：使能 CC3 中断。

Bit 2 **CC2IE**：捕获/比较 2 中断使能（Capture/Compare 2 interrupt enable）

0：禁止 CC2 中断；

1：使能 CC2 中断。

Bit 1 **CC1IE**：捕获/比较 1 中断使能（Capture/Compare 1 interrupt enable）

0：禁止 CC1 中断；

1：使能 CC1 中断。

Bit 0 **UIE**：更新中断使能（Update interrupt enable）

0：禁止更新中断；

1：使能更新中断。

20.6.5 TIM15 状态寄存器（TIM15_SR）

偏移地址：0x010

复位值：0x0000

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

Bit 15:12 保留，必须保持为复位值。

Bit 11 **CC3OF**：捕获/比较 3 重复捕获标志

参考 CC1OF 的描述

Bit 10 **CC2OF**：捕获/比较 2 重复捕获标志

参考 CC1OF 的描述

Bit 9 **CC1OF**：捕获/比较 1 重复捕获标志

仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件写入“0”可将该位清零。

0：未检测到重复捕获

1：TIM15_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

Bit 8 保留，必须保持为复位值。

Bit 7 **BIF**：刹车中断标志（Break interrupt flag）

只要 tim_brk 输入变为有效状态，此标志便由硬件置 1。刹车输入无效后可通过软件对其清零。

0：未发生刹车事件。

1：在刹车输入上检测到有效电平。

Bit 6 **TIF**：触发中断标志（Trigger interrupt flag）

在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。

0：未发生触发事件。

1：触发中断挂起。

Bit 5 **COMIF**：COM 中断标志（COM interrupt flag）

此标志在发生 COM 事件时（捕获/比较控制位 CCxE、CCxNE 和 OCxM 已更新时）由硬件置 1。但需要通过软件清零。

0：未发生 COM 事件。

1：COM 中断挂起。

Bit 4 保留，必须保持为复位值。

Bit 3 **CC3IF**：捕获/比较 3 中断标志（Capture/Compare 3 interrupt flag）

参考 CC1IF 描述。

Bit 2 **CC2IF**：捕获/比较 2 中断标志（Capture/Compare 2 interrupt flag）

参考 CC1IF 描述。

Bit 1 **CC1IF**：捕获/比较 1 中断标志（Capture/Compare 1 interrupt flag）

此标志由硬件设置，它可以通过软件（输入捕获或输出比较模式）或读取 TIM15_CCR1 寄存器（仅输入捕获模式）来清除。

0：未比较匹配/无输入捕获发生

1：发生比较匹配或输入捕获

如果通道 CC1 配置为输出：当计数器与比较值匹配时，此标志由硬件置 1，当 TIM15_CCR1 的值大于 TIM15_ARR 的值时，CC1IF 位将在计数器发生上溢（递增计数模式和增减计数模式下）或下溢（递减计数模式下）时变为高电平。在中心对齐模式下有 3 种可能的标志设置选项，请参考 TIM15_CR1 寄存器中的 CMS 位以获取完整描述。

如果通道 CC1 配置为输入：当在 TIM15_CCR1 寄存器中捕获到计数器值时（在 IC1 上检测到边沿，具体取决于在 TIM15_CCER 中通过 CC1P 和 CC1NP 位设置定义的边沿灵敏度），会设置此位。

Bit 0 **UIF**：更新中断标志（Update interrupt flag）

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

- TIM15_CR1 寄存器中的 UDIS=0，并且重复计数器值上溢或下溢时（重复计数器=0 时更新）。
- TIM15_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 TIM15_EGR 寄存器中的 UG 位重新初始化 CNT 时。
- TIM15_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参考 TIM15 从模式控制寄存器（TIM15_SMCR））。

20.6.6 TIM15 事件产生寄存器（TIM15_EGR）

偏移地址：0x014

复位：0x0000

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

Bit 15:8 保留，必须保持为复位值。

Bit 7 **BG**：刹车生成（Break generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：生成刹车事件。MOE 位清零且 BIF 标志置 1。使能后可发生相关中断传输事件。

Bit 6 **TG**：生成触发信号（Trigger generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件或 DMA 传输。

Bit 5 **COMG**：捕获/比较控制更新生成（Capture/Compare control update generation）

该位可通过软件置 1，并由硬件自动清零

0：不执行任何操作；

1：CCPC 位置 1 时，可更新 CCxE、CCxNE 和 OCxM 位。

注：此位仅对具有互补输出的通道有效。

Bit 4:3 保留，必须保持为复位值。

Bit 3 **CC3G**：捕获/比较 3 生成（Capture/Compare 3 generation）

参考 CC1G 描述。

Bit 2 **CC2G**：捕获/比较 2 生成（Capture/Compare 2 generation）

参考 CC1G 描述。

Bit 1 **CC1G**：捕获/比较 1 生成（Capture/Compare 1 generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：通道 1 上生成捕获/比较事件：

若通道 CC1 配置为输出：

使能时，CC1IF 标志置 1 并发送相应的中断请求。

如果通道 CC1 配置为输入：

TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

Bit 0 **UG**：更新生成（Update generation）

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作；

1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。

20.6.7 TIM15 捕获/比较模式寄存器 1 [复用]（TIM15_CCMR1）

偏移地址：0x018

复位值：0x0000 0000

相同的寄存器可以用于输入捕获模式（本节）或输出比较模式（下节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

Bit 31:16 保留，始终读为 0。

Bit 15:12 **IC2F[3:0]：**输入捕获 2 滤波器（Input capture 2 filter）

Bit 11:10 **IC2PSC[1:0]：**输入/捕获 2 预分频器（input capture 2 prescaler）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择（Capture/compare 2 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道被配置为输出；

01：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti2 上；

10：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti1 上；

11：CC2 通道被配置为输入，tim_ic2 映射在 tim_trc 上。此模此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIM15_CCER *中的*CC2E = 0*），才可向*CC2S *位写入数据。*

Bit 7:4 **IC1F[3:0]：**输入捕获 1 滤波器（Input capture 1 filter）

此位域可定义 tim_ti1 输入的采样频率和适用于 tim_ti1 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个事件才视为一个有效边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6。

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5。

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6。

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5。

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8。

Bit 3:2 **IC1PSC[1:0]：**输入/捕获 1 预分频器（Input capture 1 prescaler）

此位域定义 CC1 输入（tim_ic1）的预分频比。

只要 CC1E=0（TIM15_CCER 寄存器），预分频器便立即复位。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获；

10：每发生 4 个事件便执行一次捕获；

11：每发生 8 个事件便执行一次捕获。

Bit 1:0 **CC1S[1:0]：**捕获/比较 1 选择（Capture/Compare 1 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道被配置为输出；

01：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti1 上；

10：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti2 上；

11：CC1 通道被配置为输入，tim_ic1 映射在 tim_trc 上。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发输入时有效

注：仅当通道关闭时（TIM15_CCER 中的 CC1E = 0），才可向 CC1S 位写入数据。

20.6.8 TIM15 捕获/比较模式寄存器 1 [复用]（TIM15_CCMR1）

偏移地址：0x018

复位值：0x0000 0000

相同的寄存器可以用于输出比较模式（本节）或输入捕获模式（上节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							OC2M[3]	Reserved							OC1M[3]
							rw								rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

Bit 31:25 保留，必须保持为复位值。

Bit 23:17 保留，必须保持为复位值。

Bit 15 **OC2CE：**输出比较 2 清零使能（Output Compare 2 clear enable）

Bit 24, 14:12 **OC2M[2:0]：**输出比较 2 模式（Output Compare 2 mode）

Bit 11 **OC2PE：**输出比较 2 预装载使能（Output Compare 2 preload enable）

Bit 10 **OC2FE：**输出比较 2 快速使能（Output Compare 2 fast enable）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择（Capture/Compare 2 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC2 通道配置为输出

01：CC2 通道配置为输入，tim_ic2 映射到 tim_ti2 上

10：CC2 通道配置为输入，tim_ic2 映射到 tim_ti1 上

11：CC2 通道配置为输入，tim_ic2 映射到 tim_trc 上。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIM15_CCER *中的* CC2E = 0*），才可向* CC2S *位写入数据*

Bit 7 **OC1CE：**输出比较 1 清零使能（Output Compare 1 clear enable）

0：tim_oc1ref 不受 tim_ocref_clr_int 输入影响；

1：tim_oc1ref 在 tim_ocref_clr_int 输入中被检测到高电平，tim_oc1ref 立即清零。

Bit 16, 6:4 **OC1M[3:0]：**输出比较 1 模式（Output Compare 1 mode）

这些位定义提供 tim_oc1 和 tim_oc1n 的输出参考信号 tim_oc1ref 的行为。tim_oc1ref 为高电平有效，而 tim_oc1 和 tim_oc1n 的有效电平则取决于 CC1P 位和 CC1NP 位。

0000：**冻结** — 输出比较寄存器 TIM15_CCR1 与计数器 TIM15_CNT 进行比较不会对输出造成任何影响。（该模式用于生成时基）。

0001：将通道 1 设置为匹配时输出有效电平。当计数器 TIM15_CNT 与捕获/比较寄存器 1（TIM15_CCR1）匹配时，tim_oc1ref 信号强制变为高电平。

0010：将通道 1 设置为匹配时输出无效电平。当计数器 TIM15_CNT 与捕获/比较寄存器 1（TIM15_CCR1）匹配时，tim_oc1ref 信号强制变为低电平。

0011：**翻转** – TIM15_CNT=TIM15_CCR1 时，tim_oc1ref 发生翻转。

0100：**强制变为无效电平** – tim_oc1ref 强制变为低电平。

0101：**强制变为有效电平** – tim_oc1ref 强制变为高电平。

0110：**PWM 模式 1** – 在递增计数模式下，只要 TIM15_CNT<TIM15_CCR1，通道 1 便为有效状态，否则为无效状态。

0111：**PWM 模式 2** – 在递增计数模式下，只要 TIM15_CNT<TIM15_CCR1，通道 1 便为无效状态，否则为有效状态。

1000：**可重触发 OPM 模式 1** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于无效状态。

1001：**可重触发 OPM 模式 2** - 在向上计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于无效状态。然后，会像 PWM 模式 2 那样执行比较，并在下次更新时使通道再次处于无效状态。在向下计数模式下，通道在检测到触发事件（tim_trgi 信号）之前一直处于有效状态。然后，会像 PWM 模式 1 那样执行比较，并在下次更新时使通道再次处于有效状态。

1010：保留

1011：保留

1100：**组合 PWM 模式 1** - tim_oc1ref 的行为与 PWM 模式 1 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑或。

1101：**组合 PWM 模式 2** - tim_oc1ref 的行为与 PWM 模式 2 相同。tim_oc1refc 是 tim_oc1ref 和 tim_oc2ref 之间的逻辑与。

1110：保留

1111：保留

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*3 *且*CC1S= *“*00*”（通道配置为输出），这些位即无法修改。*

在 PWM *模式下，仅当比较结果发生改变或输出比较模式由“冻结”模式切换到“*PWM*”模式时，*tim_ocref_clr *电平才会发生更改。*

*对于具有互补输出的通道，此位域是预加载的。如果*TIM15_CR2 *寄存器中*CCPC *位被设置，则*OC1M *有效位仅在生成*COM *事件时从预加载位获取新值。*

Bit 3 **OC1PE**：输出比较 1 预装载使能（Output Compare 1 preload enable）

0：禁止与 TIM15_CCR1 相关的预装载寄存器。可随时向 TIM15_CCR1 写入数据，写入后将立即使用新值。

1：使能与 TIM15_CCR1 相关的预装载寄存器。可读/写访问预装载寄存器。TIM15_CCR1 预装载值在每次生成更新事件时都会装载到有效寄存器中。

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*3 *且* CC1S=00 *（通道配置为输出），这些位即无法修改。*

Bit 2 **OC1FE**：输出比较 1 快速使能（Output Compare 1 fast enable）

此位减少了触发事件和定时器输出上的转换之间的延迟。它必须在单脉冲模式（在 TIM15_CR1 寄存器中设置 OPM 位）中使用，以使输出脉冲在启动触发后尽快开始。

0：即使触发开启，CC1 也将根据计数器和 CCR1 值正常工作。触发输入出现边沿时，激活 CC1 输出的最短延迟时间为 5 个时钟周期。

1：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。

Bit 1:0 **CC1S[1:0]**：捕获/比较 1 选择（Capture/Compare 1 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：CC1 通道配置为输出

01：CC1 通道配置为输入，tim_ic1 映射到 tim_ti1 上

10：CC1 通道配置为输入，tim_ic1 映射到 tim_ti2 上

11：CC1 通道配置为输入，tim_ic1 映射到 tim_trc 上。此模式仅在通过 TS 位（TIM15_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIM15_CCER *中的* CC1E = 0*），才可向* CC1S *位写入数据*

20.6.9 TIM15 捕获/比较模式寄存器 2 [复用]（TIM15_CCMR2）

偏移地址：0x1C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

相同的寄存器可以用于输入捕获模式（本节）或输出比较模式（下节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
								rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

Bit 31:8 保留，必须保持为复位值。

Bit 7:4 **IC3F[3:0]：**输入捕获 3 滤波器（Input capture 3 filter）

参考 IC1F 描述

Bit 3:2 **IC3PSC[1:0]：**输入/捕获 1 预分频器（Input capture 3 prescaler）

参考 IC1PSC 描述

Bit 1:0 **CC3S[1:0]：**捕获/比较 3 选择（Capture/Compare 3 selection）

参考 CC1S 描述

20.6.10 TIM15 捕获/比较模式寄存器 2 [复用]（TIM15_CCMR2）

偏移地址：0x1C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

相同的寄存器可以用于输出比较模式（本节）或输入捕获模式（上节）。通道方向通过配置相应的 CCxS 位进行定义。此寄存器的所有其它位在输入模式和输出模式下的功能均不同。可以独立组合这两种模式（例如，通道 1 用作输入捕获模式而通道 2 用作输出比较模式）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															OC3M[3]
															rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
								rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式

Bit 31:17 保留，必须保持为复位值。

Bit 16 **OC3M[3]：**输出比较 3 模式（Output compare 3 mode）

Bit 15:8 保留，必须保持为复位值。

Bit 7 **OC3CE：**输出比较 3 清零使能（Output Compare 3 clear enable）

参考 OC1CE 描述

Bit 16, 6:4 **OC3M[3:0]：**输出比较 3 模式（Output Compare 1 mode）

参考 OC1M 描述

Bit 3 **OC3PE：**输出比较 3 预装载使能（Output Compare 3 preload enable）

参考 OC1PE 描述

Bit 2 **OC3FE：**输出比较 3 快速使能（Output Compare 3 fast enable）

参考 OC1FE 描述

Bit 1:0 **CC3S[1:0]：**捕获/比较 3 选择（Capture/Compare 3 selection）

参考 CC1S 描述

20.6.11 TIM15 捕获/比较使能寄存器（TIM15_CCER）

偏移地址：0x020

复位值：0x0000

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

Bit 15:12 保留，必须保持为复位值。

Bit 11 **CC3NP**：输入/捕获 3 输出极性（Capture/Compare 3 output polarity）

参考 CC1NP 的描述。

Bit 10 保留，必须保持为复位值。

Bit 9 **CC3P**：输入/捕获 3 输出极性（Capture/Compare 3 output polarity）

参考 CC1P 的描述。

Bit 8 **CC3E**：输入/捕获 3 输出使能（Capture/Compare 3 output enable）

参考 CC1E 的描述。

Bit 7 **CC2NP**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1NP 的描述。

Bit 6 保留，必须保持为复位值。

Bit 5 **CC2P**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1P 的描述。

Bit 4 **CC2E**：输入/捕获 2 输出使能（Capture/Compare 2 output enable）

参考 CC1E 的描述。

Bit 3 **CC1NP**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

CC1 通道配置为输出：

0：tim_oc1n 高电平有效；

1：tim_oc1n 低电平有效。

CC1 通道配置为输入：

此位与 CC1P 配合使用，用以定义 tim_ti1fp1/ tim_ti2fp1 的极性。请参见 CC1P 说明。

*注：只要编程了*LOCK *（*TIMx_BDTR *寄存器中的*LOCK *位）级别*2 *或*3 *且*CC1S=00 *（通道配置为输出），此位立即变为不可写状态*

注：此位将在具有互补输出的通道上进行预装载。如果 TIM15_CR2 *寄存器中的*CCPC *位置*1*，则仅当生成换向事件时，*CC1NP *有效位才会从预装载位获取新值。*

Bit 2 **CC1NE**：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 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1E 位的值。

1：开启－OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1E 位的值。

Bit 1 **CC1P**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

CC1 通道配置为输出：

0：OC1 高电平有效（输出模式）/边沿灵敏度选择（输入模式，参见以下）

1：OC1 低电平有效（输出模式）/边沿灵敏度选择（输入模式，参见以下）

CC1 通道配置为输入，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。

CC1NP=0，CC1P=0：非反相/上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=0，CC1P=1：反相/下降沿触发。电路对 TIxFP1 下降沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=1，CC1P=1：非反相/上升沿和下降沿均触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。

CC1NP=1，CC1P=0：此配置是保留的，不应该使用此配置。

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的* LOCK *位）级别*2 *或*3*，此位立即变为不可写状态。*

注：此位将在具有互补输出的通道上进行预装载。如果 TIM15_CR2 *寄存器中的*CCPC *位置*1*，则仅当生成换向事件时，*CC1P *有效位才会从预装载位获取新值。*

Bit 0 **CC1E**：输入/捕获 1 输出使能（Capture/Compare 1 output enable）

0：禁止捕获模式/OC1 未激活（参见以下）

1：使能捕获模式/ OC1 信号输出到对应的输出引脚

当 CC1 通道配置为输出，C1 的输出电平依赖于 MOE、OSSI、OSSR、OSSD、OIS1、OIS1N 和 CC1NE 位的值，不论 CC1E 位状态。请参考下表获得更多细节

表 20.11 带刹车功能的互补输出通道 tim_oc1 和 tim_oc1n 的控制位（TIM15）

控制位						输出状态 (1)	
MOE 位	OSSI 位	OSSR 位	OSSD 位	CCxE 位	CCxNE 位	tim_ocx 输出状态	tim_ocxn 输出状态
1	X	X	0	0	0	禁止输出（不由定时器驱动：高阻态） tim_ocx = 0，tim_ocxn = 0	
			1			关闭状态（输出使能为无效状态） tim_ocx = CCxP，tim_ocxn = CCxNP	
		0	X	0	1	禁止输出（不由定时器驱动：高阻态）tim_ocx = 0	tim_ocxref + 极性，tim_ocxn = tim_ocxref 异或 CCxNP
		0		1	0	tim_ocxref + 极性，tim_ocx = tim_ocxref 异或 CCxP	禁止输出（不由定时器驱动：高阻态）tim_ocxn = 0
		X		1	1	tim_ocxref + 极性 + 死区	tim_ocxref 互补项（对 tim_ocxref 进行“非”运算）+极性+死区
		1		0	1	关闭状态（输出使能为无效状态）tim_ocx = CCxP	tim_ocxref + 极性，tim_ocxn = tim_ocxref 异或 CCxNP
		1		1	0	tim_ocxref + 极性，tim_ocx = tim_ocxref 异或 CCxP	关闭状态（输出使能为无效状态）tim_ocxn = CCxNP
0	0	X	X	X	X	禁止输出（不再由定时器驱动）。	
	1		0	0	0	输出状态由 GPIO 控制器定义，可以是高电平、低电平或高阻态。	
			1	0	0	关闭状态（输出使能为无效状态）	
			X	0	1	异步：tim_ocx = CCxP、tim_ocxn = CCxNP 如果存在时钟：在死区后 tim_ocx = OISx 且 tim_ocxn = OISxN， 假定 OISx 和 OISxN 并没有都设置成 tim_ocx 及 tim_ocxn 的有效电平。	
				1	0		
				1	1		

1. 如果一个通道的两个输出均未使用（由 GPIO 接管控制控制），则 OISx、OISxN、CCxP 和 CCxNP 位必须保持清零状态。

注： *与互补通道* tim_ocx *和* tim_ocxn *相连的外部* I/O *引脚的状态取决于通道* tim_ocx *和* tim_ocxn *的状态以及* GPIO *寄存器。*

20.6.12 TIM15 计数器（TIM15_CNT）

偏移地址：0x024

复位值：0x0000 0000

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

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

Bit 31 **UIFCPY：**复制 UIF

该位是 TIM15_SR 寄存器中 UIF 位的一个只读副本。

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CNT[15:0]：**计数器的值（Counter value）

在非抖动模式下（DITHEN = 0）

该寄存器（CNT[15:0]）保存的是计数器的值。

在抖动模式下（DITHEN = 1）

该寄存器（CNT[15:0]）只保存计数器的不抖动部分。抖动部分不可用。

20.6.13 TIM15 预分频器（TIM15_PSC）

偏移地址：0x028

复位值：0x0000

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

Bit 15:0 **PSC[15:0]：**预分频器的值（Prescaler value）

计数器时钟频率（$f_{tim_cnt_ck}$）等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIM15_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到活动预分频器寄存器的值。

20.6.14 TIM15 自动重装载寄存器（TIM15_ARR）

偏移地址：0x02C

复位：0x0000 FFFF

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

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **ARR[19:0]**：自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

当自动重载值为空时，计数器不工作。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是自动重新加载的值

在抖动模式下（DITHEN = 1）

该寄存器存储的是 ARR[19:4]中的整数部分，而 ARR[3:0]位域存储的是抖动的部分

20.6.15 TIM15 重复计数寄存器（TIM15_RCR）

偏移地址：0x030

复位值：0x0000

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

Bit 15:8 保留，必须保持为复位值。

Bit 7:0 **REP[7:0]**：重复计数器的值（Repetition counter value）

使能预装载寄存器时，用户可通过这些位设置比较寄存器的更新频率（即从预装载寄存器向活动寄存器周期性传输数据）；使能更新中断时，也可设置更新中断的生成速率。与 REP_CNT 相关的减计数器每次计数到 0 时，都将生成一个更新事件并且计数器从 REP 值重新开始计数。由于只有生成重复更新事件 U_RC 时，REP_CNT 才会重载 REP 值，因此在生成下一重复更新事件之前，无论向 TIMx_RCR 寄存器写入何值都无影响。

这意味着 PWM 模式下（REP+1）相当于：

– 边沿对齐模式下的 PWM 周期数；

– 中心对称模式下的 PWM 半周期数；

20.6.16 TIM15 捕获/比较寄存器 1（TIM15_CCR1）

偏移地址：0x034

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR1[19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR1[19:0]：**捕获/比较 1 值（Capture/Compare 1 value）

若 CC1 通道配置为输出：

CCR1 为要装载到实际捕获/比较 1 寄存器的值（预装载值）。

如果没有通过 TIM15_CCMR1 寄存器中的 OC1PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 1 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIM15_CNT 进行比较并在 tim_oc1 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR1[15:0]中的比较值。CCR1[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR1[19:4]中的整数部分。CCR1[3:0]位域存储的是抖动的部分。

若 CC1 通道配置为输入：

CCR1 为上一个输入捕获 1 事件（tim_ic1）发生时的计数器值。TIM15_CCR1 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR1[15:0]位存储了捕获值。CCR1[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR1[19:4]中。CCR1[3:0]位会被重置。

20.6.17 TIM15 捕获/比较寄存器 2（TIM15_CCR2）

偏移地址：0x038

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR2[19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR2[19:0]：**捕获/比较 2 值（Capture/Compare 2 value）

若 CC2 通道配置为输出：

CCR2 为要装载到实际捕获/比较 2 寄存器的值（预装载值）。

如果没有通过 TIM15_CCMR1 寄存器中的 OC2PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 2 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIM15_CNT 进行比较并在 tim_oc2 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR2[15:0]中的比较值。CCR2[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR2[19:4]中的整数部分。CCR2[3:0]位域存储的是抖动的部分。

若 CC2 通道配置为输入：

CCR2 为上一个输入捕获 2 事件（tim_ic2）发生时的计数器值。TIM15_CCR2 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR2[15:0]位存储了捕获值。CCR2[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR2[19:4]中。CCR2[3:0]位会被重置。

20.6.18 TIM15 捕获/比较寄存器 3（TIM15_CCR3）

偏移地址：0x03C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												CCR3[19:16]			
												rw	rw	rw	rw

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:0 **CCR3[19:0]：**捕获/比较 3 值（Capture/Compare 3 value）

若 CC3 通道配置为输出：

CCR3 为要装载到实际捕获/比较 3 寄存器的值（预装载值）。

如果没有通过 TIM15_CCMR2 寄存器中的 OC3PE 位来使能预装载功能，写入的数值会被直接传输至当前寄存器中。否则只有发生更新事件时，预装载值才会复制到活动捕获/比较 3 寄存器中。实际捕获/比较寄存器中包含要与计数器 TIM15_CNT 进行比较并在 tim_oc3 输出上发出信号的值。

在非抖动模式下（DITHEN = 0）

该寄存器存储的是 CCR3[15:0]中的比较值。CCR3[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储的是 CCR3[19:4]中的整数部分。CCR3[3:0]位域存储的是抖动的部分。

若 CC3 通道配置为输入：

CCR3 为上一个输入捕获 3 事件（tim_ic3）发生时的计数器值。TIM15_CCR3 是只读的并且不可编程。

在非抖动模式下（DITHEN = 0）

CCR3[15:0]位存储了捕获值。CCR3[19:16]位会被重置。

在抖动模式下（DITHEN = 1）

该寄存器存储捕获在 CCR3[19:4]中。CCR3[3:0]位会被重置。

20.6.19 TIM15 刹车和死区寄存器（TIM15_BDTR）

偏移地址：0x044

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	OSSD	Res	BKBID	Res	BK DSRM	Reserved						BKF[3:0]			
	rw		rw		rw							rw	rw	rw	rw

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

*注：由于可以根据*LOCK *配置锁定位*OSSD、BKBID、BKF[3:0]、AOE、BKP、BKE、OSSI、OSSR *和*DTG[7:0]*的写操作，因此必须在第一次对*TIM15_BDTR *寄存器执行写访问时对这些位进行配置。*

Bit 31 保留，必须保持为复位值。

Bit 30 **OSSD**：互补通道都禁用模式下关闭状态选择（**CCxE=CCxNE=0**）

此位在 MOE=1，**CCxE=CCxNE=0** 时，作用于配置为输出的通道使其输出无效状态。

此位在 MOE=0，OSSI=1，**CCxE=CCxNE=0** 时，作用于配置为输出的通道。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIM15 捕获/比较使能寄存器（TIM15_CCER））。

0：处于无效状态时，当 **CCxE=CCxNE=0** 时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于有效状态时：

一旦 MOE=1，**CCxE=CCxNE=0** 时，便将 tim_ocx/tim_ocxn 输出设为无效状态。（输出仍由定时器控制）。

一旦 MOE=0，OSSI=1，**CCxE=CCxNE=0** 时，首先将 tim_ocx/tim_ocxn 输出强制为其无效电平，然后在死区后将其强制为空闲电平。定时器始终控制输出。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*2，*此位即无法修改。*

Bit 29 保留，必须保持为复位值。

Bit 28 **BKBID**：刹车双向（Break bidirectional）

0：刹车输入 tim_brk 处于输入模式

1：刹车输入 tim_brk 处于双向模式

在双向模式（BKBID 置 1）下，刹车输入同时配置为输入模式和开漏输出模式。任何有效的 break 事件会在刹车输入上输出低电平，以向外部设备指示内部 break 事件。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 27 保留，必须保持为复位值。

Bit 26 **BKDSRM：**刹车解除保护（Break disarm）

0：刹车输入 tim_brk 处于保护状态

1：刹车输入 tim_brk 解除保护

当没有刹车源处于有效状态时，该位由硬件清零。

软件需置位 BKDSRM 以释放双向输出控制（开漏输出处于高阻态），并轮询直至硬件将其清零，表明故障状态已消失。

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 25:20 保留，必须保持为复位值。

Bit 19:16 **BKF[3:0]：**刹车滤波器（Break filter）

此位域可定义 tim_brk 输入的采样频率和适用于 tim_brk 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个连续事件才视为一个有效输出边沿：

0000：无滤波器，tim_brk 异步工作

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/4$，N=8。

0101：$f_{SAMPLING}=f_{DTS}/8$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/16$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/32$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/64$，N=6。

1011：$f_{SAMPLING}=f_{DTS}/64$，N=8。

1100：$f_{SAMPLING}=f_{DTS}/128$，N=6。

1101：$f_{SAMPLING}=f_{DTS}/128$，N=8。

1110：$f_{SAMPLING}=f_{DTS}/256$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/256$，N=8。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 15 **MOE：**主输出使能（Main output enable）

只要刹车输入（tim_brk）为有效状态，此位便由硬件异步清零。此位由软件置 1，也可根据 AOE 位状态自动置 1。此位仅对配置为输出的通道有效。

0：响应刹车事件或向 MOE 写入 0 时：tim_ocx 和 tim_ocxn 输出被禁止或被强制为空闲状态，具体取决于 OSSI 位。

1：如果 tim_ocx 和 tim_ocxn 输出的相应使能位（TIM15_CCER 寄存器中的 CCxE 和 CCxNE 位）均置 1，则使能 tim_ocx 和 tim_ocxn 输出。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIM15 捕获/比较使能寄存器（TIM15_CCER））。

Bit 14 **AOE**：自动输出使能（Automatic output enable）

0：MOE 只能由软件置 1

1：MOE 可由软件置 1，也可在发生下一更新事件时自动置 1（如果刹车输入 tim_brk 无效）

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 13 **BKP**：刹车极性（Break polarity）

0：刹车输入 tim_brk 为低电平有效

1：刹车输入 tim_brk 为高电平有效

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 12 **BKE**：刹车使能（Break enable）

0：禁止刹车输入（tim_brk 及 tim_sys_brk 事件）；

1：使能刹车输入（tim_brk 及 tim_sys_brk 事件）。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

*注：对该位执行任何写操作后，都需要经过*1 *个*APB *时钟周期的延迟才生效。*

Bit 11 **OSSR**：运行模式下的关闭状态选择（Off-state selection for Run mode）

此位在 MOE=1 时作用于配置为输出模式且具有互补输出的通道。如果定时器中没有互补输出，则不存在 OSSR。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIM15 捕获/比较使能寄存器（TIM15_CCER））。

0：处于无效状态时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于无效状态时，一旦 CCxE=1 或 CCxNE=1，便使能 tim_ocx/tim_ocxn 输出并将其设为无效电平（输出仍由定时器控制）。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*2，*此位即无法修改。*

Bit 10 **OSSI**：空闲模式下的关闭状态选择（Off-state selection for Idle mode）

当由于刹车事件或软件写操作而使 MOE=0 时，此位作用于配置为输出的通道。

有关详细信息，请参见 tim_ocx/tim_ocxn 使能说明（TIM15 捕获/比较使能寄存器（TIM15_CCER））。

0：处于无效状态时，禁止 tim_ocx/tim_ocxn 输出（定时器释放输出控制，由强制高阻态的 GPIO 逻辑接管）。

1：处于无效状态时，首先将 tim_ocx/tim_ocxn 输出强制为其无效电平，然后在死区后将其强制为空闲电平。定时器始终控制输出。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*2，*此位即无法修改。*

Bit 9:8 **LOCK[1:0]：**锁定设置（Lock configuration）

这些位用于针对软件错误提供写保护。

00：关闭锁定 – 不对任何位提供写保护。

01：锁定级别 1，此时无法对 TIM15_BDTR 寄存器中的 DTG 位、TIM15_CR2 寄存器中的 OISx 和 OISxN 位以及 TIM15_BDTR 寄存器中的 BKBID/BKE/BKP/AOE 位执行写操作。

10：锁定级别 2，此时无法对锁定级别 1 中适用的各位、CC 极性位（TIM15_CCER 寄存器中的 CCxP/CCxNP 位，只要通过 CCxS 位将相关通道配置为输出）以及 OSSD、OSSR 和 OSSI 位执行写操作。

11：锁定级别 3，此时无法对锁定级别 2 中适用的各位、CC 控制位（TIM15_CCMRx 寄存器中的 OCxM 和 OCxPE 位，只要通过 CCxS 位将相关通道配置为输出）执行写操作。

注：复位后只能对 LOCK *位执行一次写操作。对* TIM15_BDTR *寄存器执行写操作后其中的内容将冻结，直到下一次复位。*

Bit 7:0 **DTG[7:0]：**配置死区发生器（Dead-time generator setup）

此位域定义插入到互补输出之间的死区持续时间。DT 与该持续时间相对应。

DTG[7:5]=0xx => DT= DTG[7:0] × T_{dtg}，其中 T_{dtg} = T_{DTS}；

DTG[7:5]=10x => DT=（64+DTG[5:0]）× T_{dtg}，其中 T_{dtg} = 2 × T_{DTS}；

DTG[7:5]=110 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 8 × T_{DTS}；

DTG[7:5]=111 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 16 × T_{DTS}；

例：若 T_{DTS} = 125ns（8MHZ），可能的死区时间为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

注：只要编程了 LOCK（TIM15_BDTR *寄存器中的* LOCK *位）级别*1、2 *或*3，*此位域即无法修改。*

20.6.20 TIM15 定时器死区寄存器 2（TIM15_DTR2）

偏移地址：0x054

复位值：0x0000 0000

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

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

Bit 31:18 保留，必须保持为复位值。

Bit 17 **DTPE：**死区时间预加载使能（Deadtime preload enable）

0：死区时间值未预加载

1：使能死区时间值预加载

*注：编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 16 **DTAE：**死区时间非对称使能（Deadtime asymmetric enable）

0：上升沿和下降沿的死区时间是相同的，并且使用 DTG[7:0]寄存器定义

1：上升沿的死区时间使用 DTG[7:0]寄存器定义，下降沿的死区时间使用 DTGF[7:0]位定义。

*注：编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

Bit 15:8 保留，必须保持为复位值。

Bit 7:0 **DTGF[7:0]：**死区下降沿发生器（Dead-time falling edge generator setup）

这个位域定义了在下降沿时，互补输出之间插入死区时间的长度。

DTG[7:5]=0xx => DT= DTG[7:0] × T_{dtg}，其中 T_{dtg} = T_{DTS}；

DTG[7:5]=10x => DT=（64+DTG[5:0]）× T_{dtg}，其中 T_{dtg} = 2 × T_{DTS}；

DTG[7:5]=110 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 8 × T_{DTS}；

DTG[7:5]=111 => DT=（32+DTG[4:0]）× T_{dtg}，其中 T_{dtg} = 16 × T_{DTS}；

例：若 T_{DTS} = 125ns（8MHZ），可能的死区时间为：

0 到 15875 ns（步长为 125 ns），

16 us 到 31750 ns（步长为 250 ns），

32 us 到 63us（步长为 1 us），

64 us 到 126 us（步长为 2 us）

*注：编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1、2 *或*3，*此位即无法修改。*

20.6.21 TIM15 定时器输入选择寄存器（TIM15_TISEL）

偏移地址：0x05C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													TI3SEL[2:0]		
													rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					TI2SEL[2:0]			Reserved					TI1SEL[2:0]		
					rw	rw	rw						rw	rw	rw

Bit 31:19 保留，必须保持为复位值。

Bit 18:16 **TI3SEL[2:0]：**选择 tim_ti3[6:0]输入（Selects tim_ti3[6:0] input）

0000：TIM15_CH3 输入（tim_ti3_in0）

0001：tim_ti3_in1

0010：tim_ti3_in2

0011：tim_ti3_in3

0100：tim_ti3_in4

0101：tim_ti3_in5

0110：tim_ti3_in6

参考章节：TIM15 引脚和内部信号获得关联列表。

Bit 15:11 保留，必须保持为复位值。

Bit 10:8 **TI2SEL[2:0]：**选择 tim_ti2[4:0]输入（Selects tim_ti2[4:0] input）

0000：TIM15_CH2 输入（tim_ti2_in0）

0001：tim_ti2_in1

0001：tim_ti2_in1

0010：tim_ti2_in2

0011：tim_ti2_in3

0100：tim_ti2_in4

参考章节：TIM15 引脚和内部信号获得关联列表。

Bit 7:3 保留，必须保持为复位值。

Bit 2:0 **TI1SEL[2:0]：**选择 tim_ti1[4:0]输入（Selects tim_ti1[4:0] input）

0000：TIM15_CH1 输入（tim_ti1_in0）

0001：tim_ti1_in1

0010：tim_ti1_in2

0011：tim_ti1_in3

0100：tim_ti1_in4

参考章节：TIM15 引脚和内部信号获得关联列表。

20.6.22 TIM15 复用功能选择寄存器 1（TIM15_AF1）

偏移地址：0x060

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BKCMP4P	BKCMP3P	BKCMP2P	BKCMP1P	BKINP	Reserved				BKCMP4E	BKCMP3E	BKCMP2E	BKCMP1E	BKINE
		rw	rw	rw	rw	rw					rw	rw	rw	rw	rw

*注：有关产品特定实现的信息，请参考章节：*TIM15 *引脚和内部信号。*

Bit 31:14 保留，必须保持为复位值。

Bit 13 **BKCMP4P**：tim_brk_cmp4 输入极性（tim_brk_cmp4 input polarity）

此位选择 tim_brk_cmp2 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp4 输入高电平有效

1：tim_brk_cmp4 输入低电平有效

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

Bit 12 **BKCMP3P**：tim_brk_cmp3 输入极性（tim_brk_cmp3 input polarity）

此位选择 tim_brk_cmp3 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp3 输入高电平有效

1：tim_brk_cmp3 输入低电平有效

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

Bit 11 **BKCMP2P**：tim_brk_cmp2 输入极性（tim_brk_cmp2 input polarity）

此位选择 tim_brk_cmp2 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp2 输入高电平有效

1：tim_brk_cmp2 输入低电平有效

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

Bit 10 **BKCMP1P**：tim_brk_cmp1 输入极性（tim_brk_cmp1 input polarity）

此位选择 tim_brk_cmp1 输入敏感度。它必须与 BKP 极性位一起编程。

0：tim_brk_cmp1 输入高电平有效

1：tim_brk_cmp1 输入低电平有效

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

Bit 9 **BKINP**：TIMx_BKIN 输入极性（TIMx_BKIN input polarity）

该位选择 TIMx_BKIN 备用功能输入敏感度。必须对其进行编程

0：TIMx_BKIN 输入电平有效

1：TIMx_BKIN 输入低电平有效

*注：只要编程了*LOCK *（*TIM15_BDTR *寄存器中的*LOCK *位）级别*1*，此位即无法修改。*

Bit 8:5 保留，必须保持为复位值。

Bit 4 **BKCMP4E**：tim_brk_cmp4 使能（tim_brk_cmp4 enable）

该位用于使能或禁用 tim_brk_cmp4 输入。如果该位设置为 1，则 tim_brk_cmp4 输入将被使能，否则将被禁用。

0：tim_brk_cmp4 输入禁用

1：tim_brk_cmp4 输入使能

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 4 **BKCMP3E**：tim_brk_cmp3 使能（tim_brk_cmp3 enable）

该位用于使能或禁用 tim_brk_cmp3 输入。如果该位设置为 1，则 tim_brk_cmp3 输入将被使能，否则将被禁用。

0：tim_brk_cmp3 输入禁用

1：tim_brk_cmp3 输入使能

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 2 **BKCMP2E**：tim_brk_cmp2 使能（tim_brk_cmp2 enable）

该位用于使能或禁用 tim_brk_cmp2 输入。如果该位设置为 1，则 tim_brk_cmp2 输入将被使能，否则将被禁用。

0：tim_brk_cmp2 输入禁用

1：tim_brk_cmp2 输入使能

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 1 **BKCMP1E**：tim_brk_cmp1 使能（tim_brk_cmp1 enable）

该位用于使能或禁用 tim_brk_cmp1 输入。如果该位设置为 1，则 tim_brk_cmp1 输入将被使能，否则将被禁用。

0：tim_brk_cmp1 输入禁用

1：tim_brk_cmp1 输入使能

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 0 **BKINE**：TIMx_BKIN 输入使能（TIMx_BKIN input enable）

该位用于使能或禁用 TIMx_BKIN alternate function 输入作为定时器的 tim_brk 输入。TIMx_BKIN 输入与其他 tim_brk 源进行“或”操作。

0：TIMx_BKIN 输入禁用

1：TIMx_BKIN 输入使能

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

20.6.23 TIM15 复用功能选择寄存器 2（TIM15_AF2）

偏移地址：0x064

复位值：0x0000 0000

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

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

Bit 31:18 保留，必须保持为复位值。

Bit 17:16 **OCRSEL[1:0]：**ocref_clr 源选择（ocref_clr source selection）

这些位选择 ocref_clr 输入源。

00：tim_ocref_clr0

01：tim_ocref_clr1

10：tim_ocref_clr2

11：tim_ocref_clr3

参考章节：TIM15 引脚和内部信号获取更多特定产品信息。

*注：只要编程了*LOCK（TIM15_BDTR *寄存器中的*LOCK *位）级别*1，*此位即无法修改。*

Bit 15:0 保留，必须保持为复位值。

21 捕获定时器（TIMA/TIMB）

21.1 TIMA/TIMB 简介

捕获定时器包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。

它们专用于捕获功能，测量输入信号的脉冲宽度（输入捕获）。使用定时器预分频器和 RCC 时钟控制器预分频器，可将脉冲宽度从几微秒调制到几毫秒。这些定时器彼此完全独立，不共享任何资源，它们可以同步操作。

21.2 TIMA/TIMB 主要功能

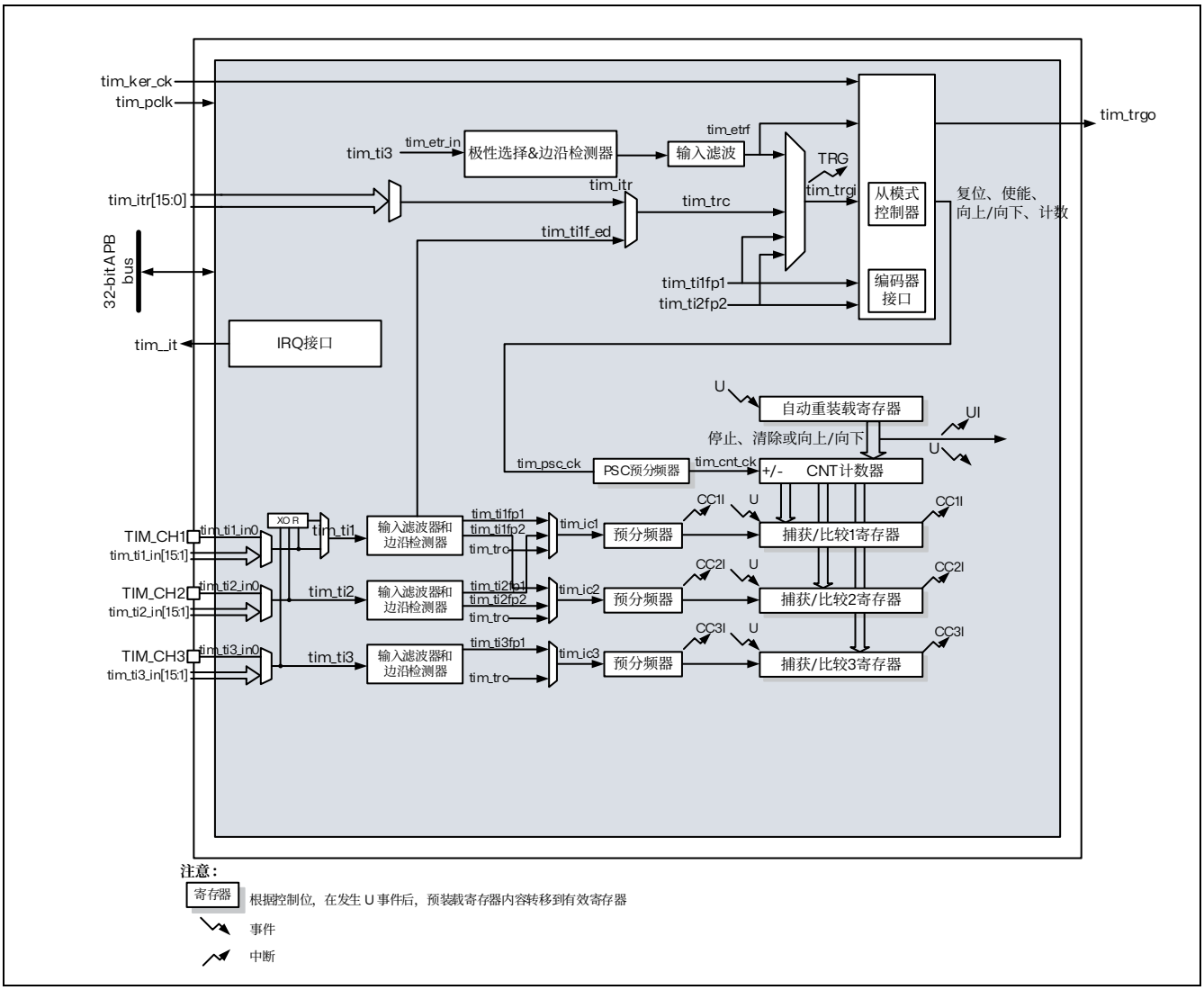
捕获 TIMx（TIMA、TIMB）定时器功能包括：

- 16 位（TIMA 和 TIMB）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
- 3 个独立通道：
  - 输入捕获
- 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/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
- 支持针对定位的增量（正交）编码器和霍尔传感器电路

21.3 TIMA/TIMB 功能描述

21.3.1 框图

图 21.1 捕获定时器框图



21.3.2 TIMA/TIMB 引脚和内部信号

本节中的下两张表格总结了 TIM 的输入和输出。

表 21.1 TIM 输入/输出引脚

引脚名称	信号类型	描述
TIM_CH1 TIM_CH2 TIM_CH3	输入/输出	定时器捕获通道。 每个通道专用于捕获。 TIM_CH1 和 TIM_CH2 也可用作外部时钟（tim_ker_ck 时钟的 1/4 以下），外部触发器和正交编码器输入。 TIM_CH1，TIM_CH2 和 TIM_CH3 可被用作数字霍尔效应传感器接口。

表 21.2 TIM 内部输入/输出信号

内部信号名称	信号类型	描述
tim_ti1_in[15:0] tim_ti2_in[15:0] tim_ti3_in[15:0]	输入	内部定时器输入总线。 tim_ti1_in[15:0]和 tim_ti2_in[15:0]输入可被用作捕获或作为外部时钟（tim_ker_ck 时钟的 1/4 以下）和正交编码器信号。
tim_etr_in	输入	外部触发器内部输入总线。此输入可被用作触发器、外部时钟。 在编码器接口模式下的索引输入功能通过 tim_ti3 连接 tim_etr_in。
tim_itr[15:0]	输入	内部触发输入总线。这些输入可用于从模式控制器或作为输入时钟(tim_ker_ck 时钟的 1/4 以下)。
tim_trgo	输出	内部触发器输出。此触发器可以触发其他片上外设。
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟。
tim_it	输出	全局定时器中断，收集捕获、更新和中断触发器请求。

如下三张表列出了连接到 tim_ti[3:1]输入多路复用器的源。

表 21.3 连接到 tim_ti1 的输入多路复用器

tim_ti1 输入	源	
	TIMA	TIMB
tim_ti1_in0	TIMA_CH1	TIMB_CH1
tim_ti1_in1	comp1_out	comp1_out
tim_ti1_in2	comp3_out	comp3_out
tim_ti1_in3	comp2_out	comp4_out
tim_ti1_in4	LSI	LSI

表 21.4 连接到 tim_ti2 的输入多路复用器

tim_ti2 输入	源	
	TIMA	TIMB
tim_ti2_in0	TIMA_CH2	TIMB_CH2
tim_ti2_in1	comp1_out	comp1_out
tim_ti2_in2	comp3_out	comp3_out
tim_ti2_in3	comp4_out	comp2_out

表 21.5 连接到 tim_ti3 的输入多路复用器

tim_ti3 输入	源	
	TIMA	TIMB
tim_ti3_in0	TIMA_CH3	TIMB_CH3
tim_ti3_in1	comp1_out	comp1_out
tim_ti3_in2	comp3_out	comp3_out
tim_ti3_in3	comp2_out	comp4_out

下表列出连接到 tim_itr 输入多路复用器的内部源

表 21.6 TIMx 内部触发连接

tim_itrx 输入	TIMA	TIMB
tim_itr0	tim2_trgo	tim2_trgo
tim_itr1	tim3_trgo	tim3_trgo
tim_itr2	tim8_trgo	tim8_trgo
tim_itr3	tim15_trgo	tim15_trgo
tim_itr4	tim1_trgo	tim1_trgo
tim_itr5	timB_trgo	timA_trgo
tim_itr[15:6]	保留	

下表列出连接到 tim_etr 输入多路复用器的内部源。

表 21.7 连接到 tim_etr 的输入多路复用器

定时器外部触发输入信号	定时器外部触发信号分配	
	TIMA	TIMB
tim_etr	tim_ti3(1)	tim_ti3(1)

1. 关于信号 tim_ti3 详情请参考上表：连接到 tim_ti3 的输入多路复用器

21.3.3 时基单元

可编程定时器的主要模块由一个 16 位计数器及其相关的自动重装寄存器组成。计数器可递增计数、递减计数或同时进行递增和递减计数。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包含：

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）

对自动重载寄存器执行写入或读取操作时会直接传送到影子寄存器。当计数器达到上溢值（或者在递减计数时达到下溢值）并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生 进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位（CEN）置 1 时，才会启动计数器（有关计数器使能的更多详细信息，另请参见从模式控制器的相关说明）。

请注意，真正的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

预分频器描述

预分频器可对计数器时钟频率进行分频，分频系数介于 1 到 65536 之间。该预分频器基于 16 位寄存器（TIMx_PSC 寄存器）所控制的 16 位计数器。由于该控制寄存器具有缓冲功能，因此预分频器可实现实时更改。而新的预分频比将在下一更新事件发生时被采用.

下两张图给出了在预分频器运行时，更改计数器参数的例子。

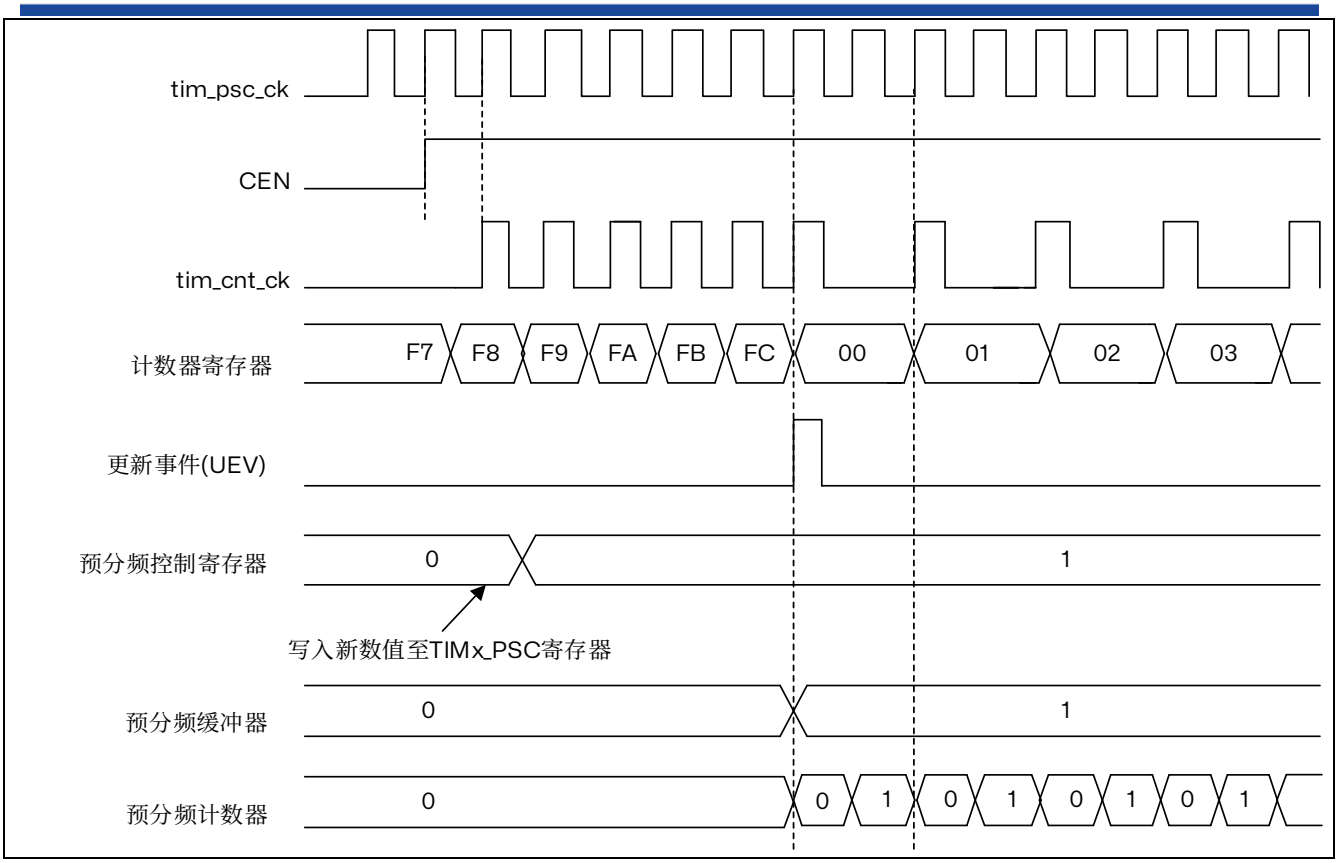


图 21.2 当预分频器的参数从 1 变到 2 时，计数器的时序图

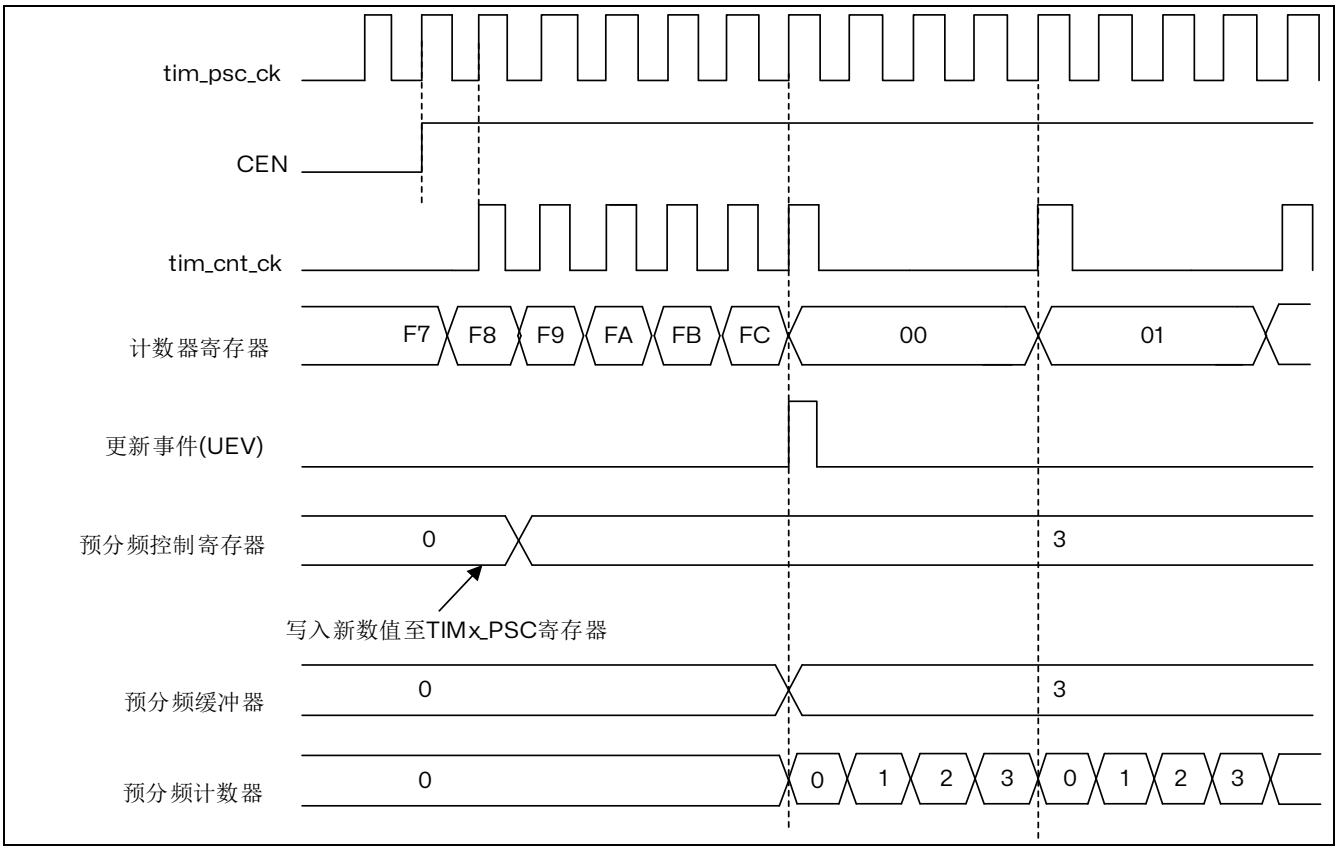


图 21.3 当预分频器的参数从 1 变到 4 时，计数器的时序图

21.3.4 计数器模式

递增计数模式

在递增计数模式下，计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）
- 自动重载影子寄存器将以预装载值进行更新

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

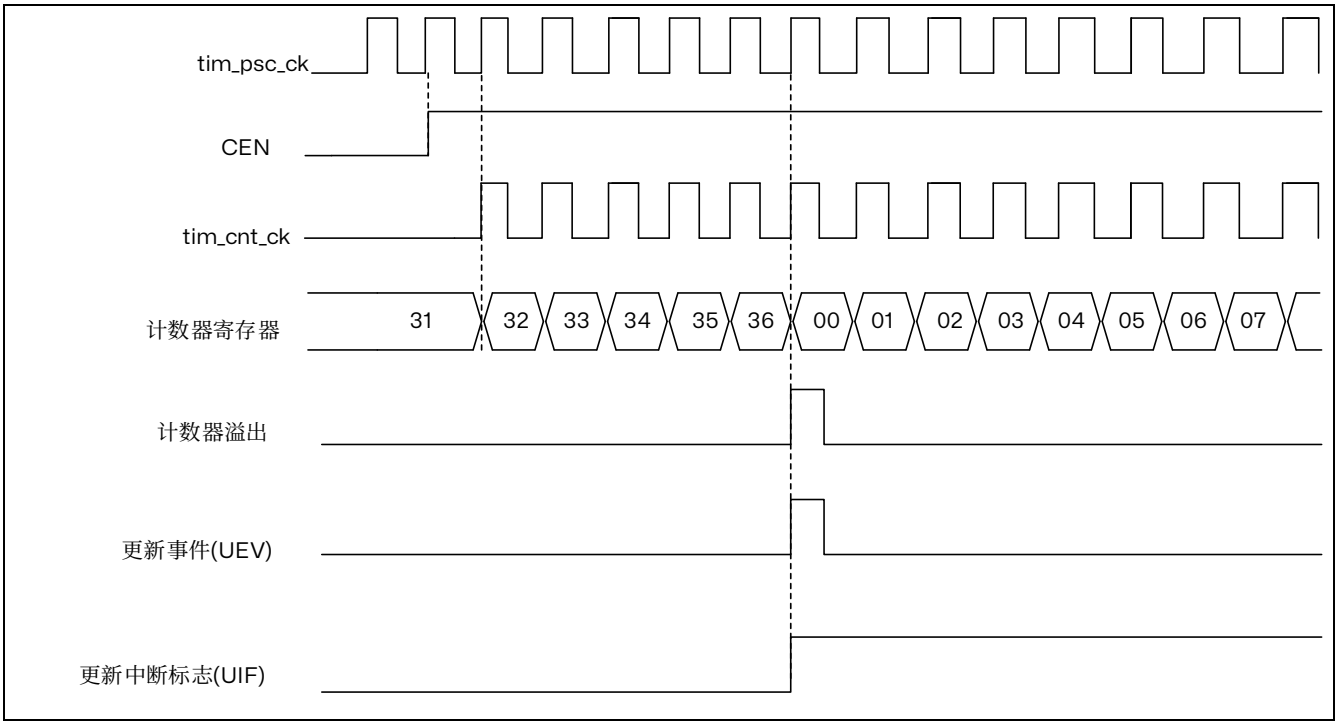


图 21.4 计数器时序图，内部时钟分频系数为 1

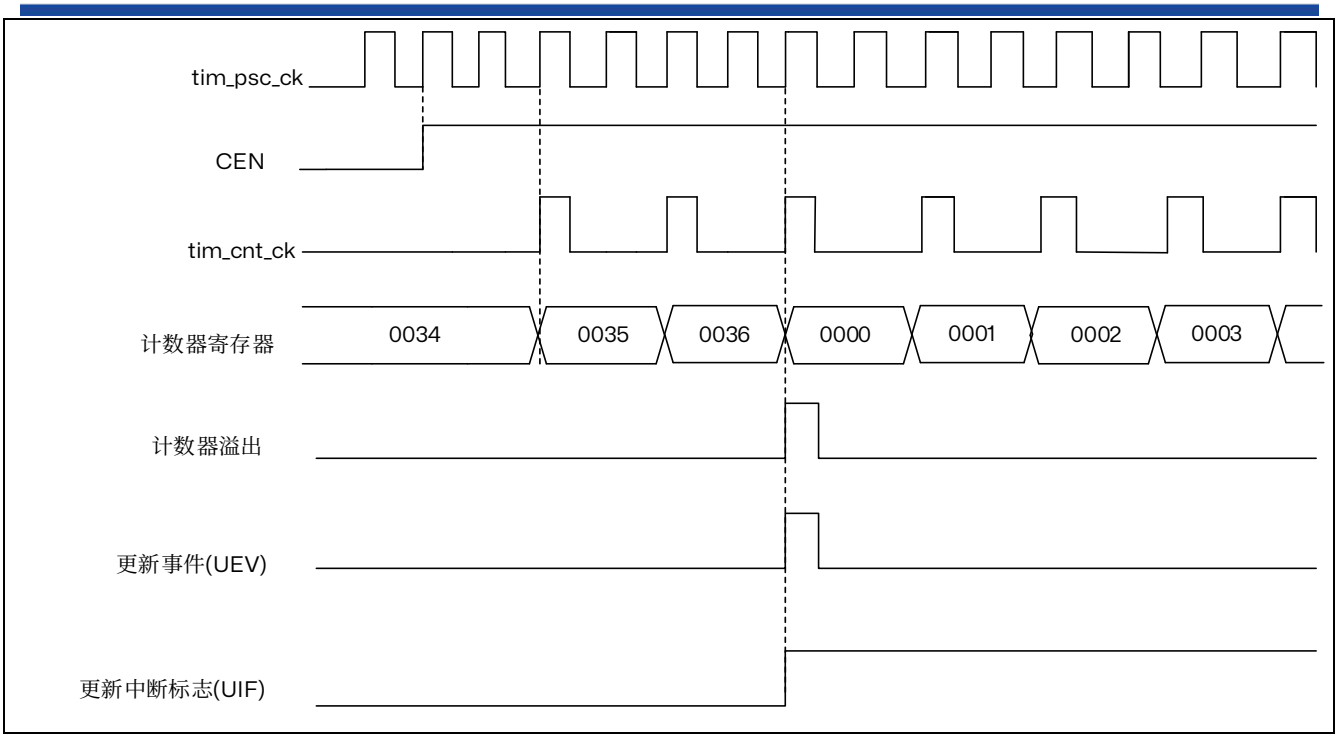


图 21.5 计数器时序图，内部时钟分频系数为 2

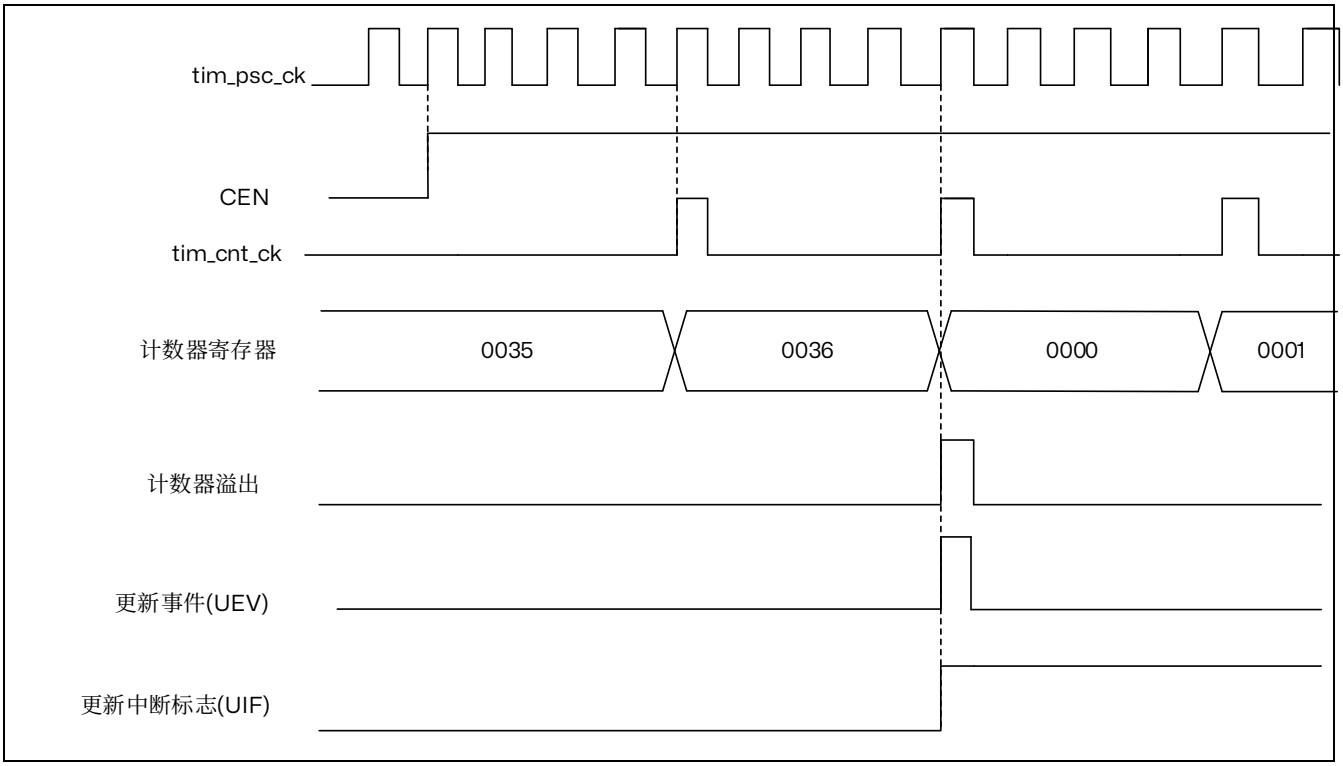


图 21.6 计数器时序图，内部时钟分频系数为 4

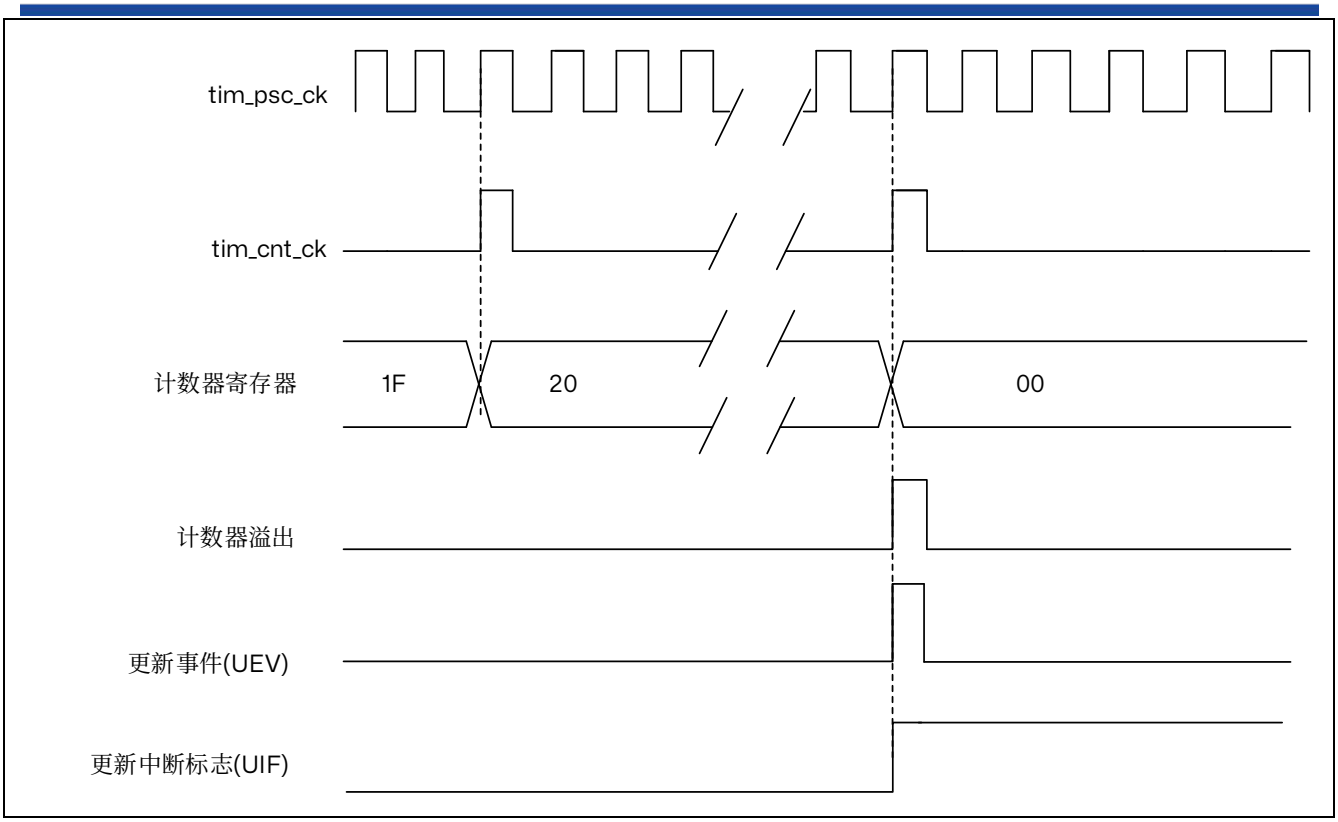


图 21.7 计数器时序图，内部时钟分频系数为 N

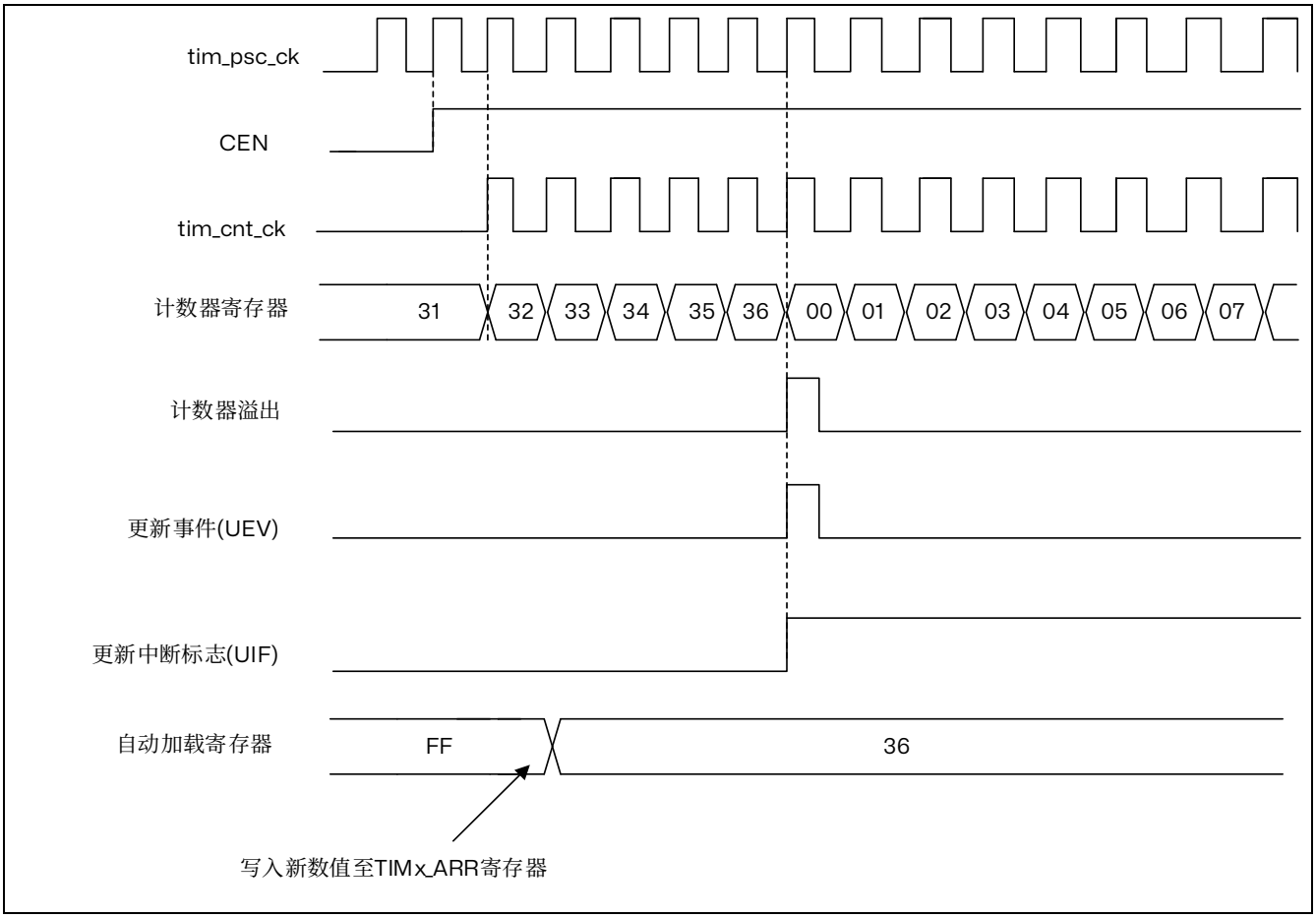


图 21.8 计数器时序图，更新事件

递减计数模式

在递减计数模式下，计数器从自动重载值（TIMx_ARR 寄存器的内容）开始递减计数到 0，然后重新从自动重载值开始计数并生成计数器下溢事件。

每次发生计数器下溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器会重新从当前自动重载值开始计数，而预分频器计数器则重新从 0 开始计数（但预分频比保持不变）。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，自动重载寄存器会在计数器重载之前得到更新，因此，下一个计数周期就是我们所希望的新的周期长度。

以下各图以一些示例说明当 TIMx_ARR=0x36 时不同时钟频率下计数器的行为。

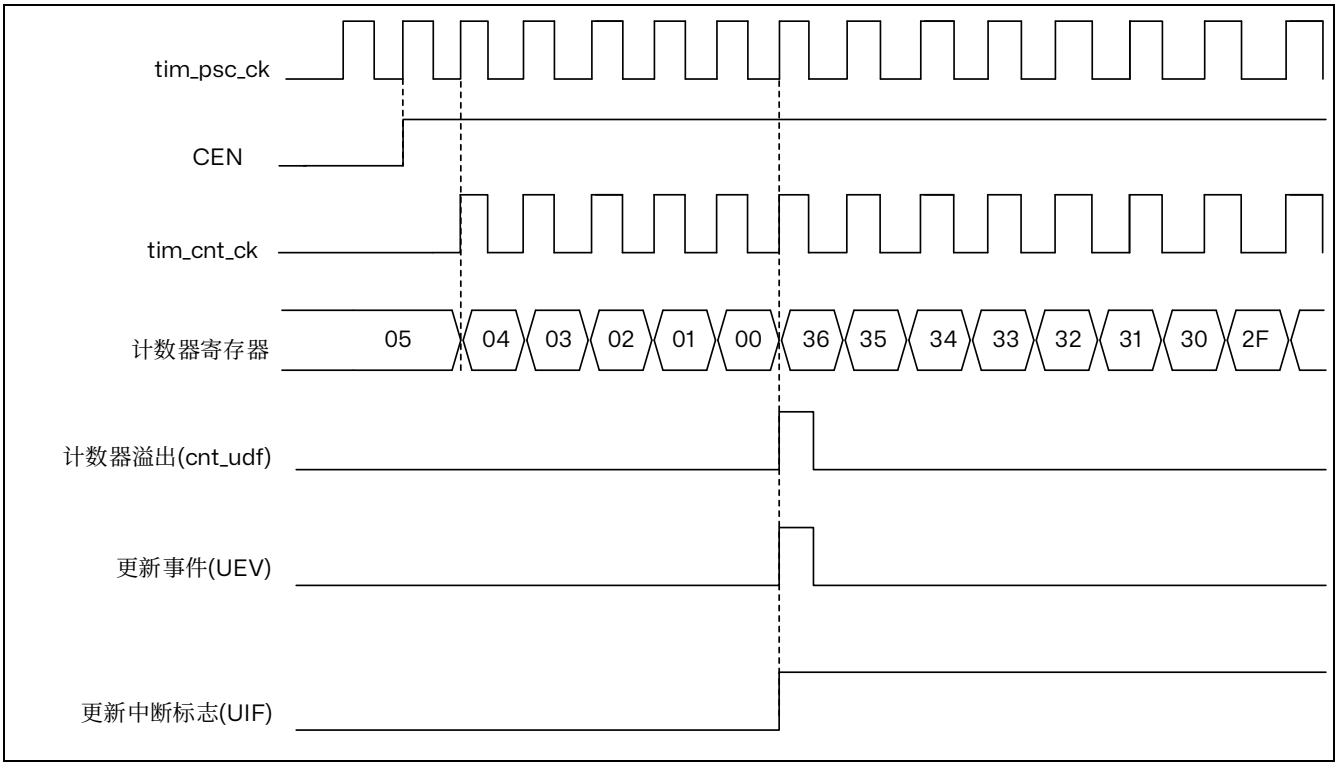


图 21.9 计数器时序图，内部时钟分频系数为 1

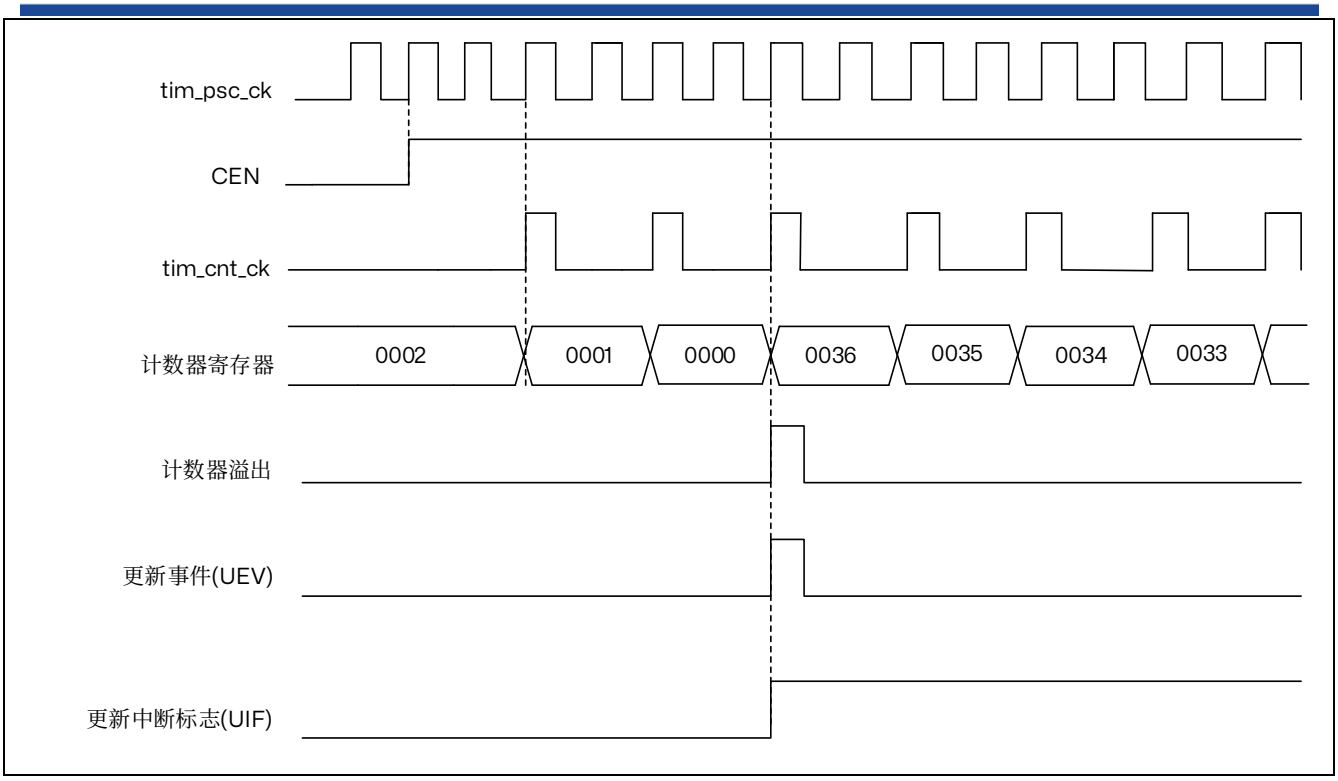


图 21.10 计数器时序图，内部时钟分频系数为 2

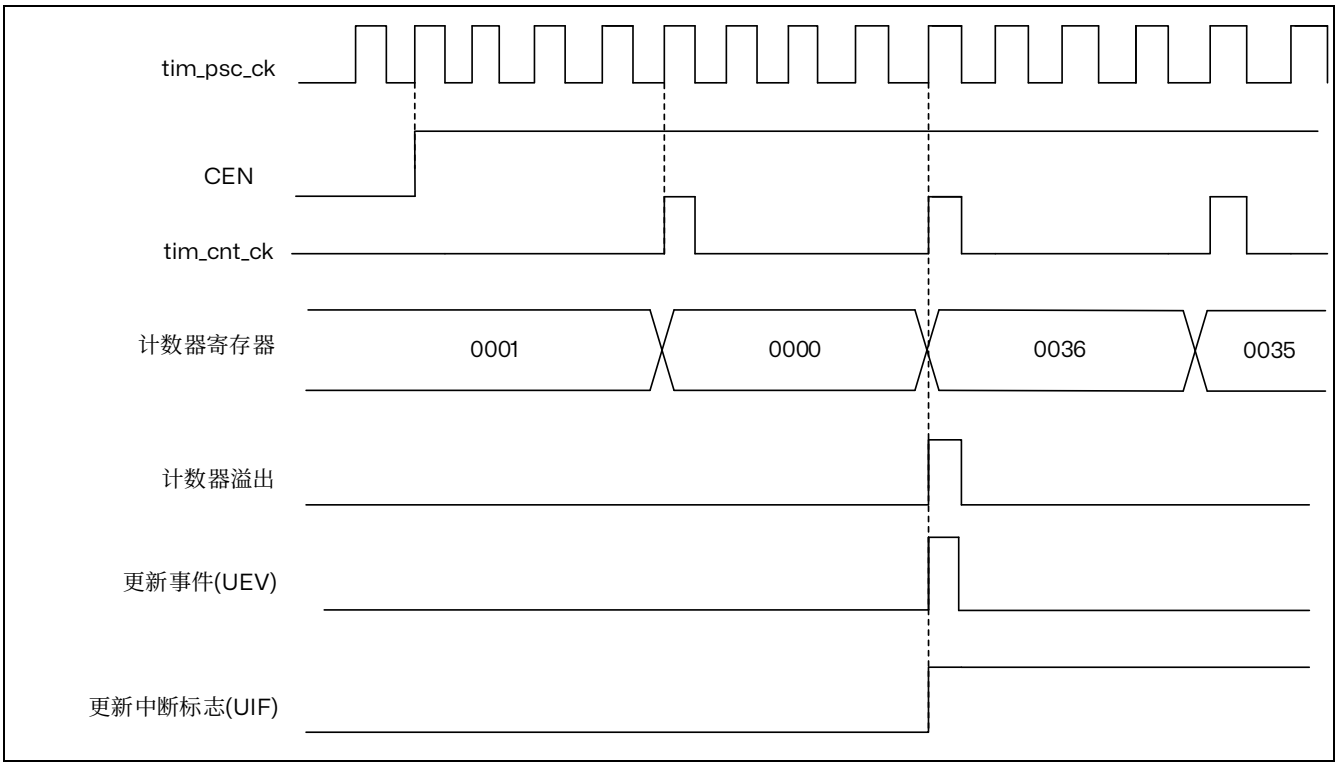


图 21.11 计数器时序图，内部时钟分频系数为 4

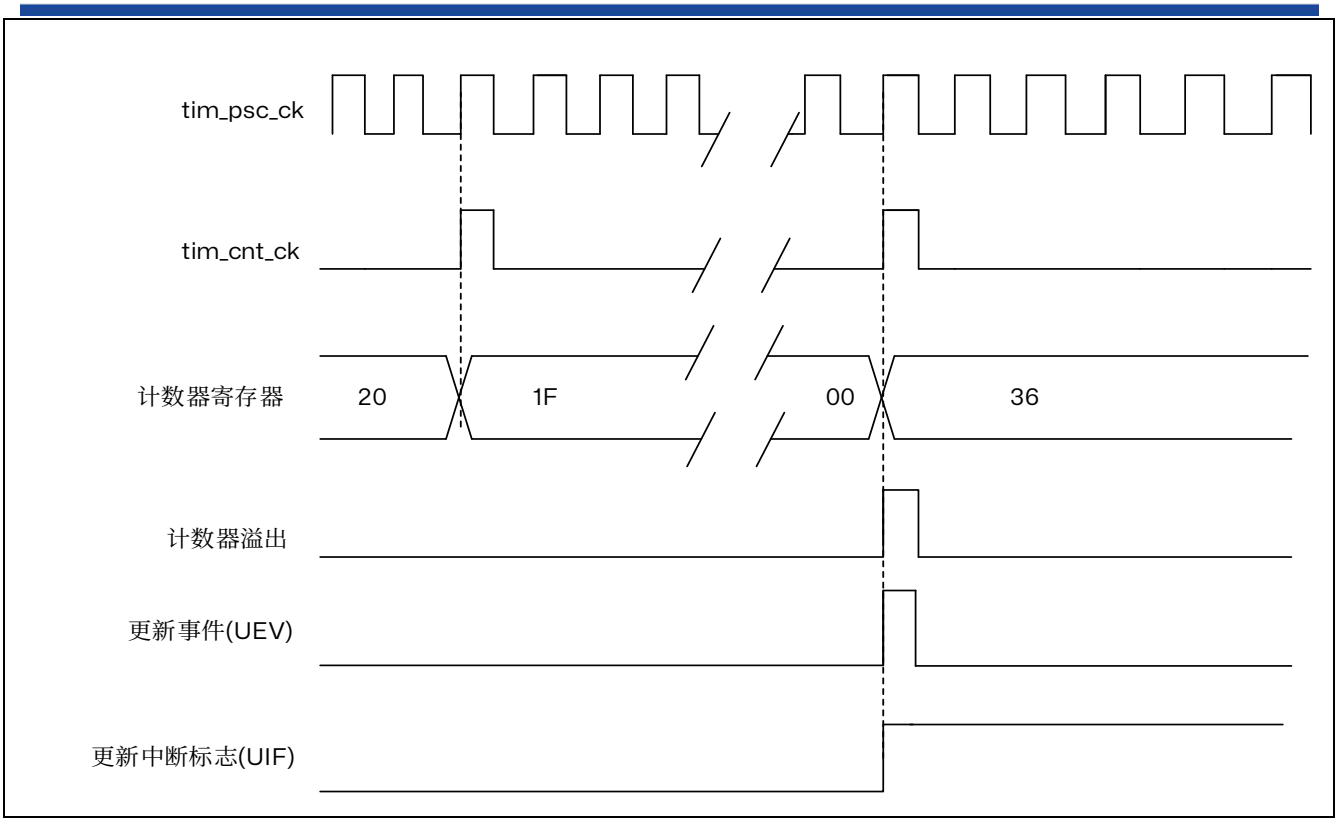


图 21.12 计数器时序图，内部时钟分频系数为 N

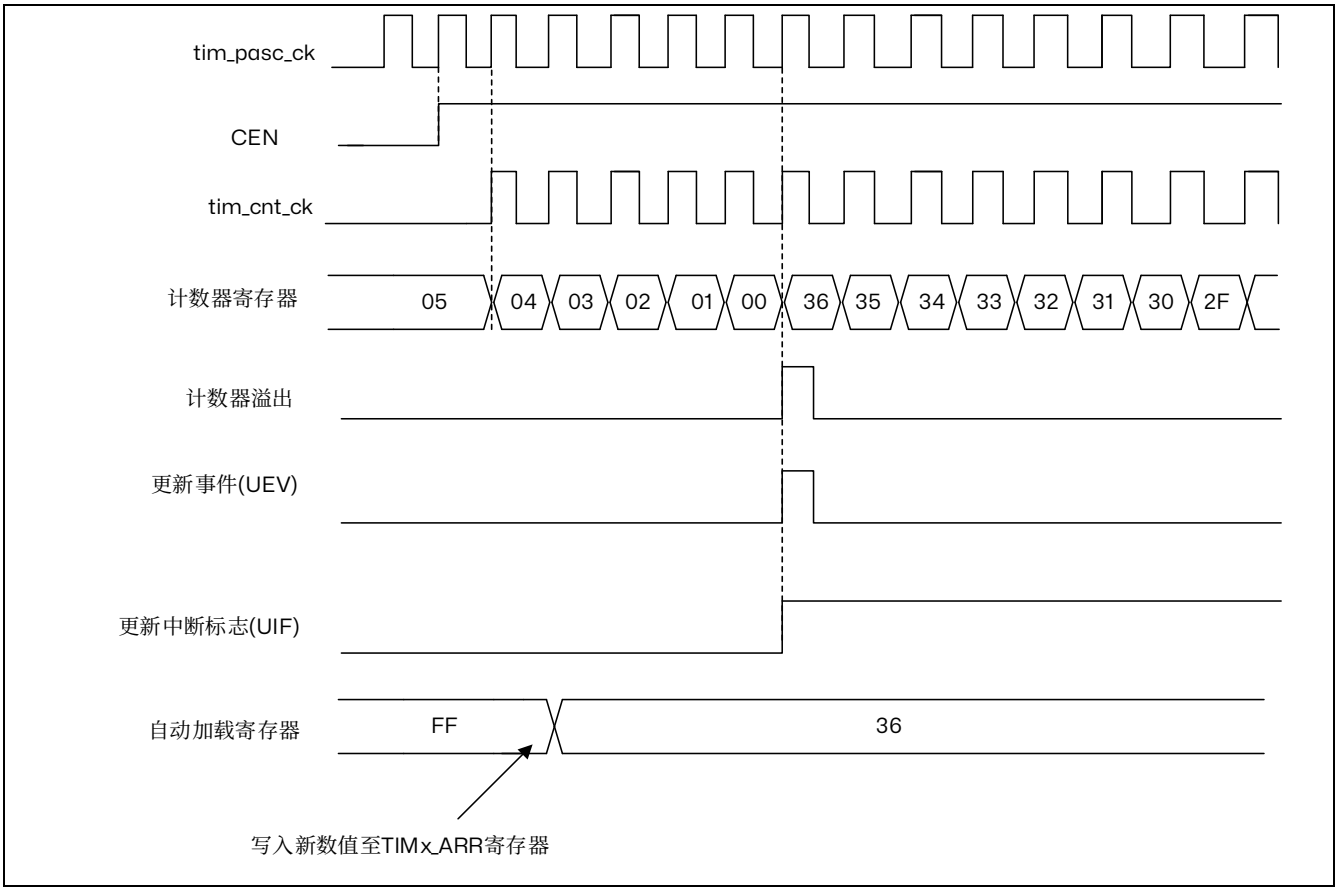


图 21.13 计数器时序图，更新事件

中央对齐模式（递增/递减计数）

在中心对齐模式下，计数器从 0 开始计数到自动重载值（TIMx_ARR 寄存器的内容）– 1，生成计数器上溢事件；然后从自动重载值开始向下计数到 1 并生成计数器下溢事件。之后从 0 开始重新计数。

当 TIMx_CR1 寄存器中的 CMS 位不为“00”时，中心对齐模式有效。将通道配置为输出模式时，其输出比较中断标志将在以下模式下置 1，即：计数器递减计数（中心对齐模式 1，CMS = “01”）、计数器递增计数（中心对齐模式 2，CMS = “10”）以及计数器递增/递减计数（中心对齐模式 3，CMS = “11”）。

此模式下无法写入方向位（TIMx_CR1 寄存器中的 DIR 位）。而是由硬件更新并指示当前计数器方向。

每次发生计数器上溢和下溢时都会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。这种情况下，计数器以及预分频器计数器将重新从 0 开始计数。

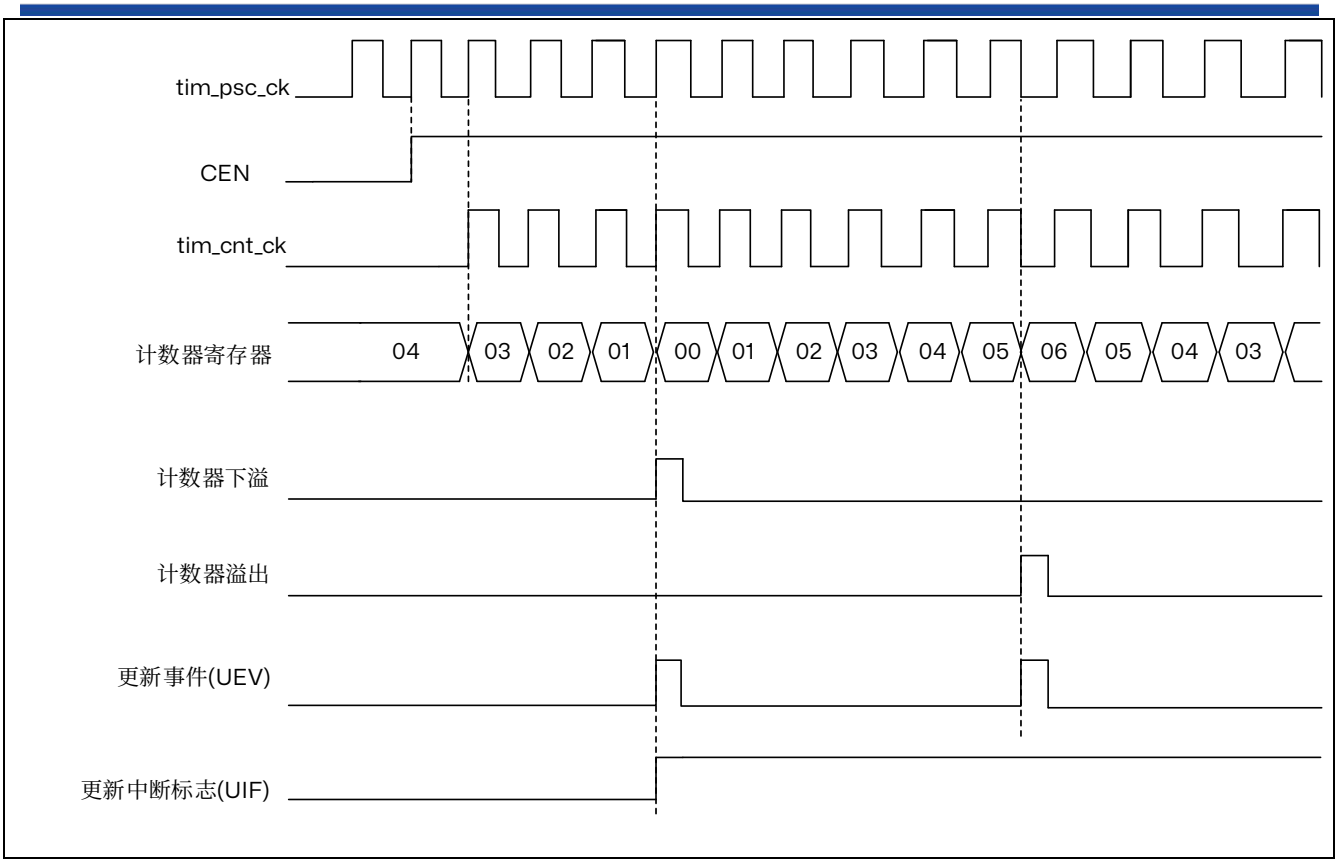
通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 更新事件。这可避免向预装载寄存器写入新值时更新影子寄存器。在 UDIS 位写入 0 之前不会产生任何更新事件。不过，计数器仍会根据当前自动重载值进行递增和递减计数。

此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求请求）。这样一来，如果在发生捕获事件时将计数器清零，将不会同时产生更新中断和捕获中断。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 预分频器的缓冲区中将重新装载预装载值（TIMx_PSC 寄存器的内容）。
- 当自动重载活动寄存器将以预装载值（TIMx_ARR 寄存器的内容）进行更新。注意，如果更新操作是由计数器上溢触发的，则自动重载寄存器在重载计数器之前更新，因此，下一个计数周期就是我们所希望的新的周期长度（计数器被重载新的值）。

以下各图以一些示例说明不同时钟频率下计数器的行为。



1. 这里使用了中心对齐模式 1

图 21.14 计数器时序图，内部时钟分频系数为 1，TIMx_ARR=0x6

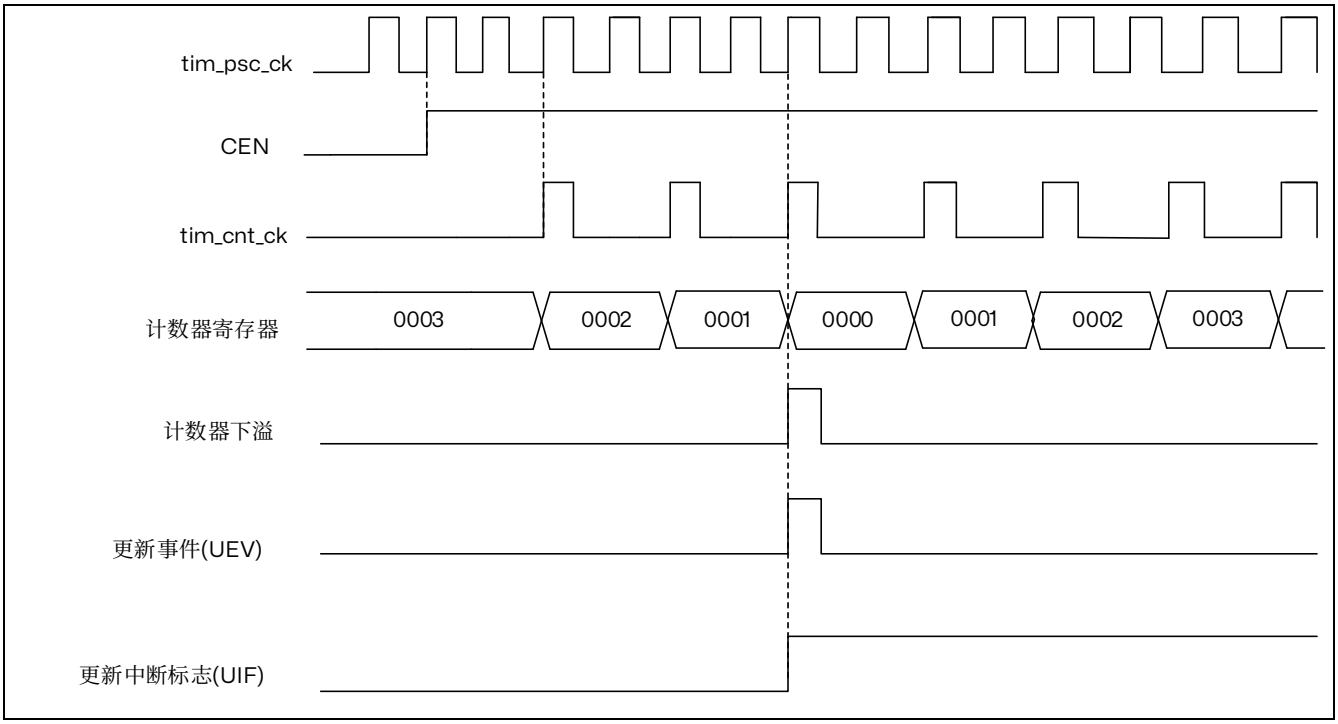


图 21.15 计数器时序图，内部时钟分频系数为 2

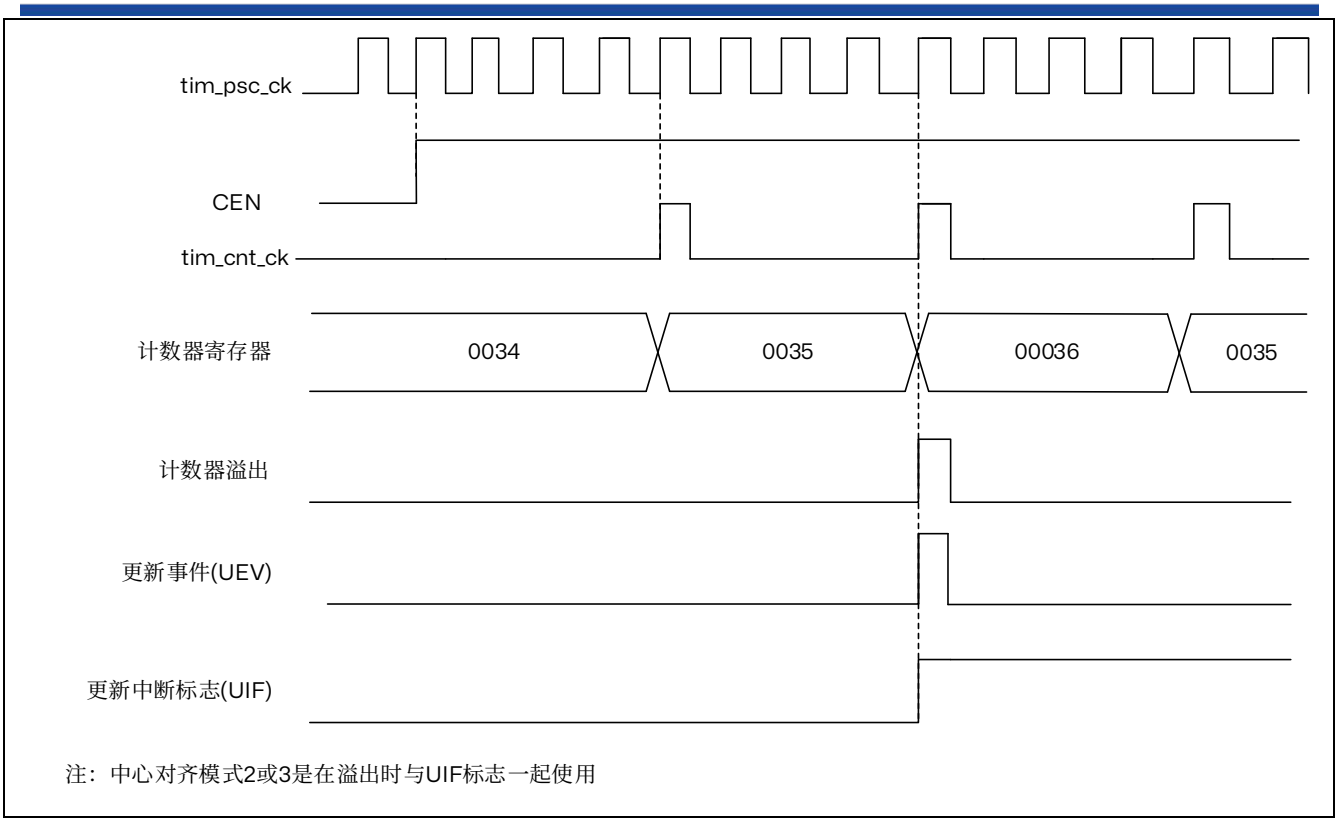


图 21.16 计数器时序图，内部时钟分频系数为 4，TIMx_ARR=0x36

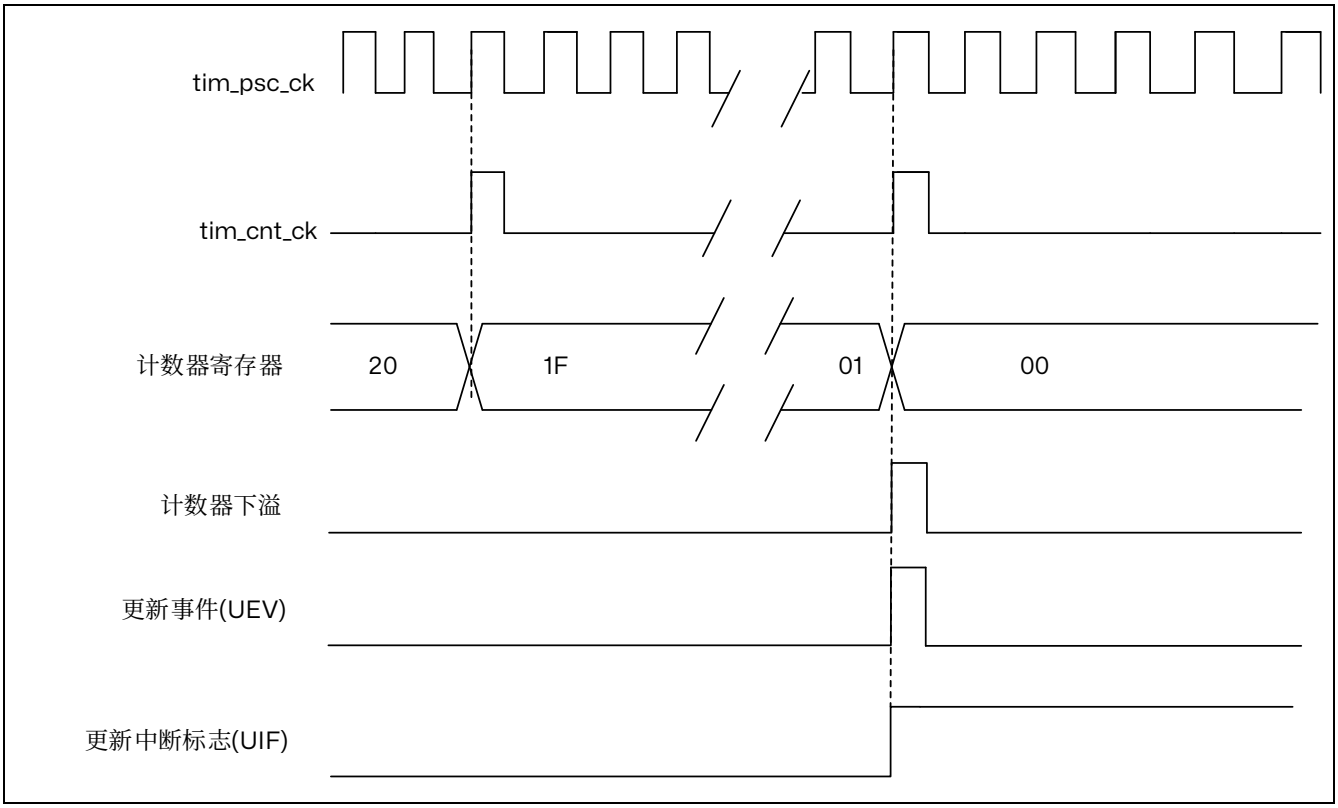


图 21.17 计数器时序图，内部时钟分频系数为 N

21.3.5 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（tim_ker_ck）
- 外部时钟模式 1：外部输入引脚（tim_ti1 或 tim_ti2）
- 外部时钟模式 2：外部触发输入（(tim_etr_in）
- 内部触发输入（tim_itr)：使用一个定时器作为另一个定时器的预分频器，例如可以将定时器 8 配置为定时器 2 的预分频器。

内部时钟源（tim_ker_ck）

如果禁止从模式控制器（TIMx_SMCR 寄存器中 SMS=000），则 CEN 位、DIR 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

下图显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

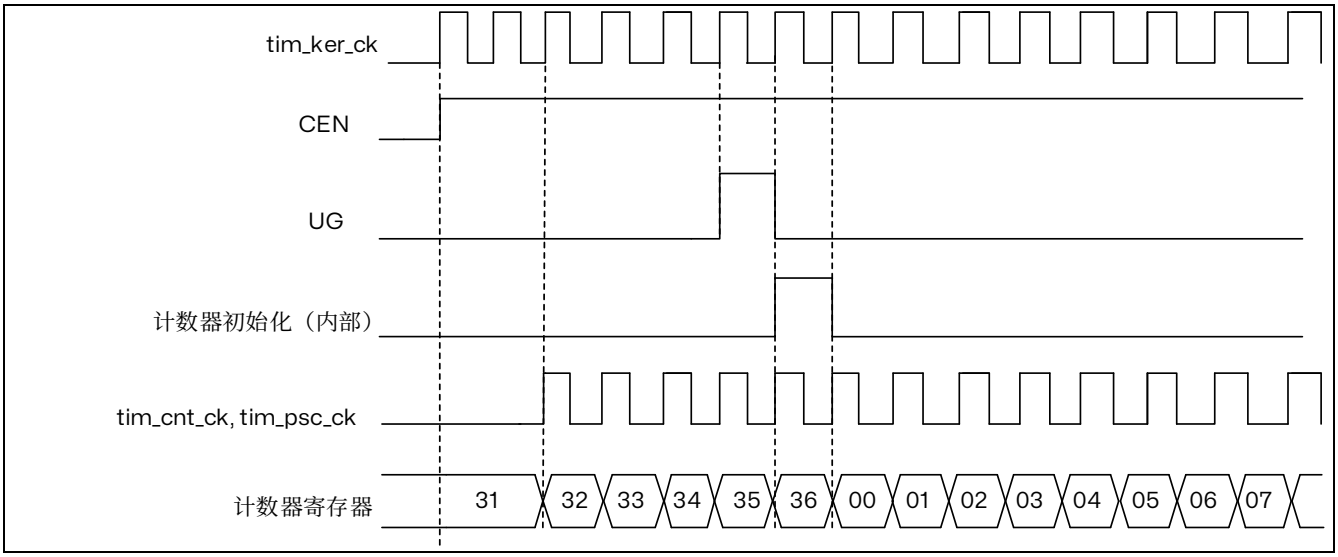


图 21.18 一般模式下的控制电路，内部时钟分频系数为 1

外部时钟源模式 1

当 TIMx_SMCR 寄存器中的 SMS=111 时，可选择此模式。计数器可在选定的输入信号上出现上升沿或下降沿时计数。

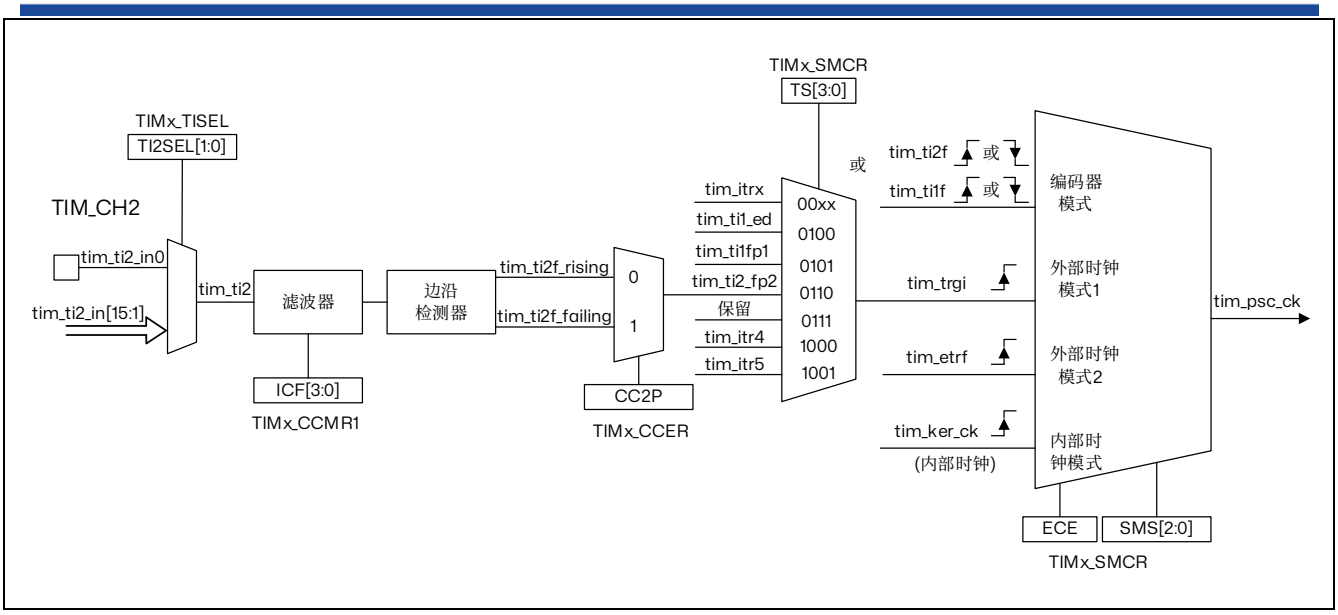


图 21.19 tim_ti2 外部时钟连接例子

例如，要使递增计数器在 tim_ti2 输入出现上升沿时计数，请执行以下步骤：

1. 通过在 TIMx_TISEL 寄存器中使用 TI2SEL[1:0]位选择合适的 tim_ti2_in[15:0]源（内部或外部）。
2. 通过在 TIMx_CCMR1 寄存器中写入 CC2S=“01”来配置通道 2，使其能够检测 tim_ti2 输入的上升沿。
3. 通过在 TIMx_CCMR1 寄存器中写入 IC2F[3:0]位来配置输入滤波时间（如果不需要任何滤波，请保持 IC2F=0000）。

注：由于捕获预分频器不用于触发操作，因此无需对其进行配置。

4. 通过在 TIMx_CCER 寄存器中写入 CC2P=0 和 CC2NP=0 来选择上升沿极性。
5. 通过在 TIMx_SMCR 寄存器中写入 SMS=111，使定时器在外部时钟模式 1 下工作。
6. 通过在 TIMx_SMCR 寄存器中写入 TS=110 来选择 tim_ti2 作为输入源。
7. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

当 tim_ti2 出现上升沿时，计数器便会计数一次并且 TIF 标志置 1。

tim_ti2 的上升沿与实际计数器时钟之间的延迟是由于 tim_ti2 输入的重新同步电路引起的。

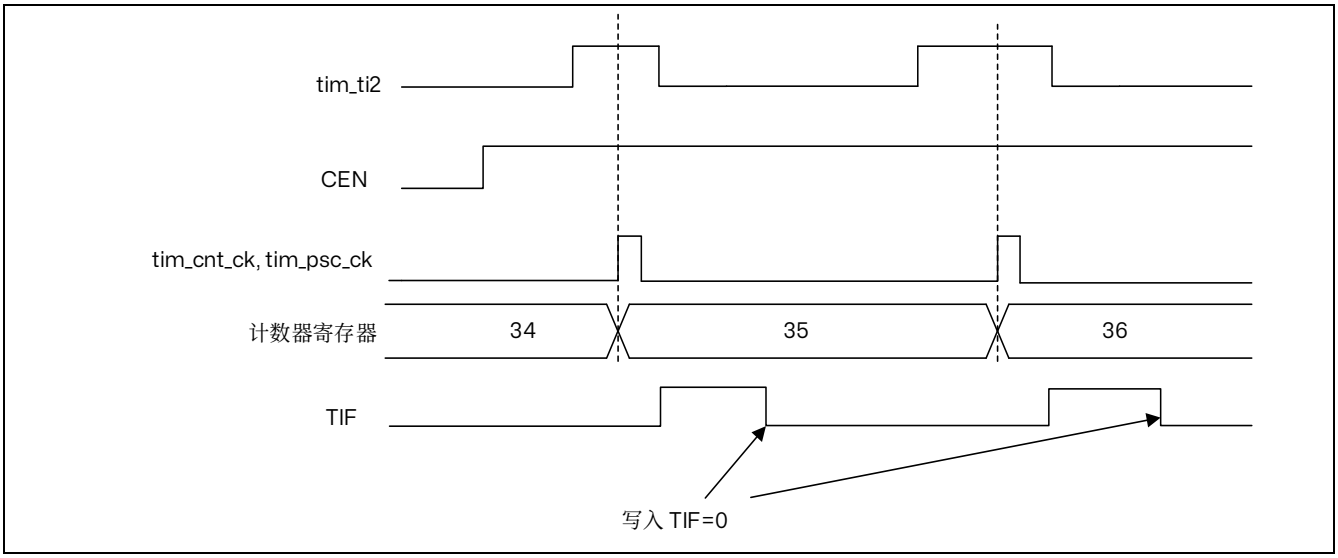


图 21.20 外部时钟模式 1 下的控制电路

外部时钟源模式 2

通过在 TIMx_SMCR 寄存器中写入 ECE=1 选择此模式。
计数器可在外部触发输入 ETR 出现上升沿或下降沿时计数。
下图是外部触发输入的框图

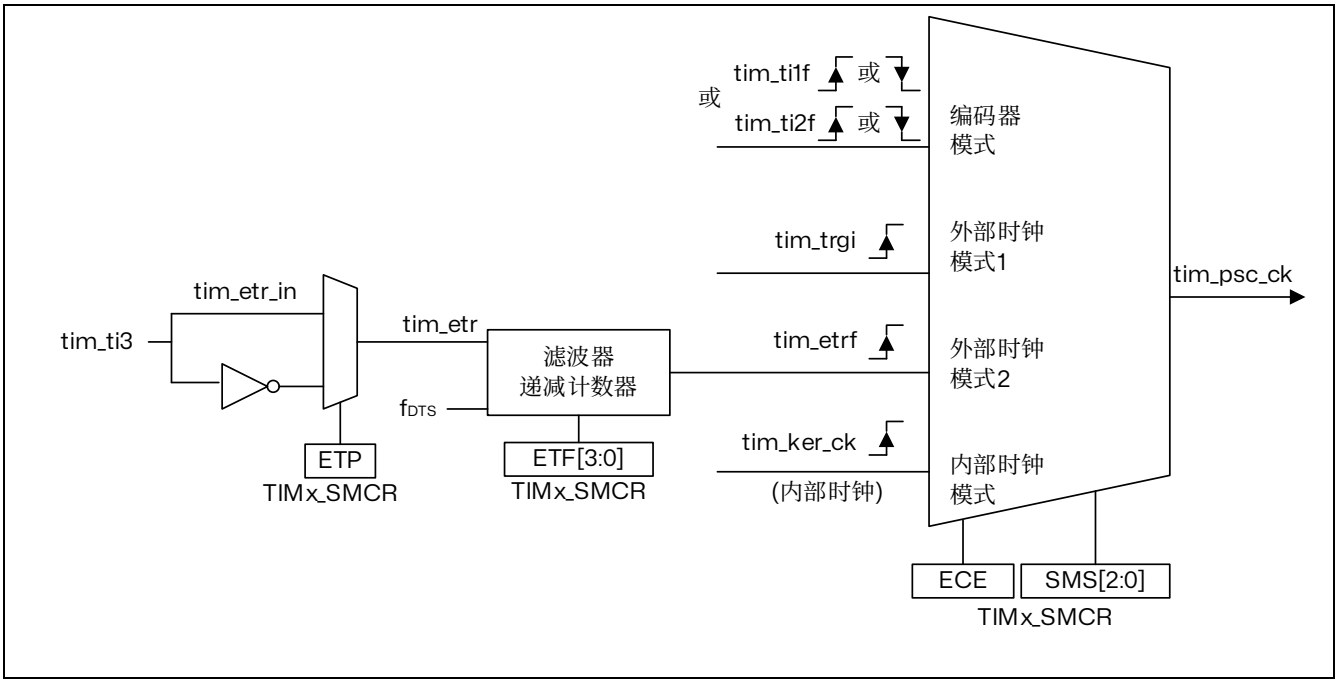


图 21.21 外部触发输入框图

例如，要使递增计数器在 ETR 每出现 2 个上升沿时计数，请执行以下步骤：

1. 由于此例中不需滤波器，因此在 TIMx_SMCR 寄存器中写入 ETF[3:0]=0000。
2. 通过在 TIMx_SMCR 寄存器中写入 ETP=0 来选择 tim_etr_in 引脚的上升沿检测。
3. 通过在 TIMx_SMCR 寄存器中写入 ECE=1 来使能外部时钟模式 2。
4. 通过在 TIMx_CR1 寄存器中写入 CEN=1 来使能计数器。

tim_etr_in 每出现 2 个上升沿，计数器计数一次。

在 tim_etr_in 的上升沿和计数器实际时钟之间的延时取决于在 tim_etrp 信号端的重新同步电路。因此，计数器可以正确捕获的最大频率最高为 TIMxCLK 频率的¼。

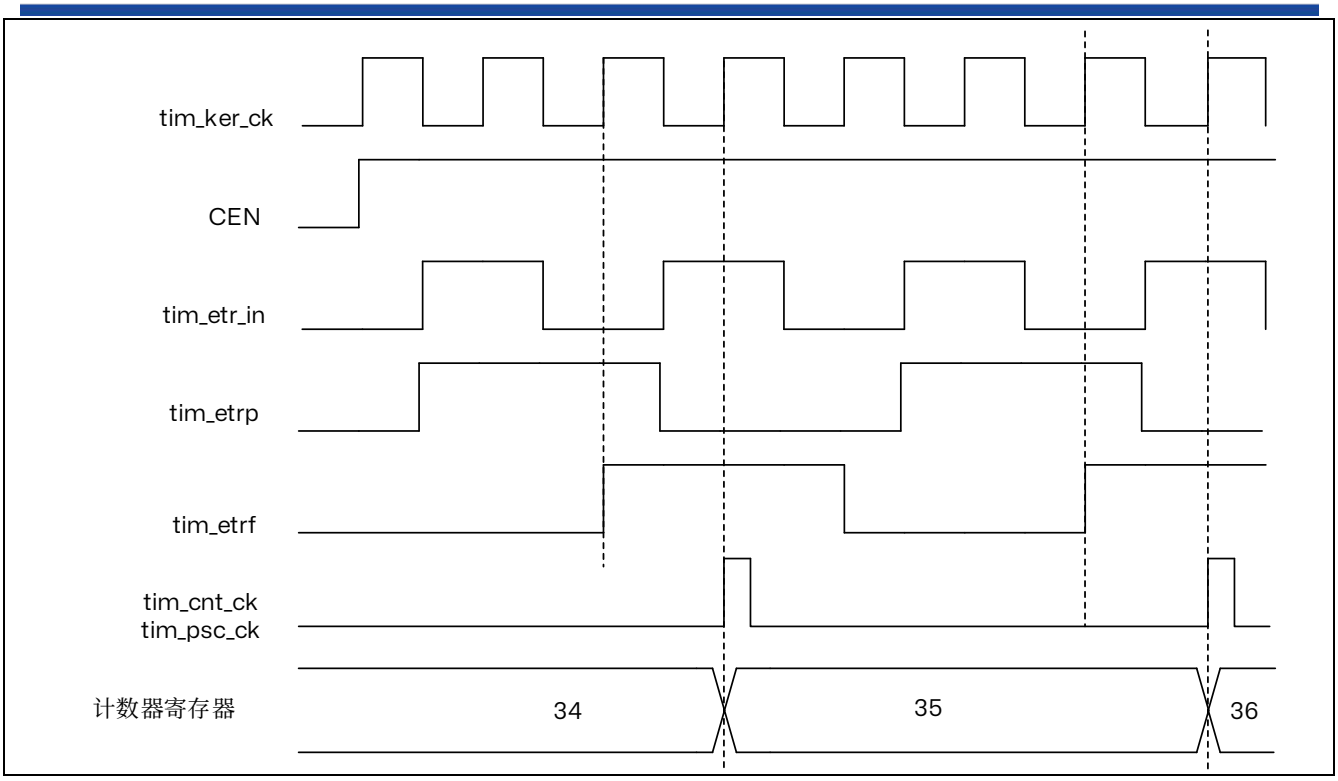


图 21.22 外部时钟模式 2 下的控制电路

21.3.6 输入捕获模式

在输入捕获模式下，当相应的 ICx 信号检测到跳变沿后，将使用捕获/比较寄存器(TIMx_CCRx) 来锁存计数器的值。发生捕获事件时，会将相应的 CCXIF 标志（TIMx_SR 寄存器）置 1，并可发送中断请求（如果已使能）。如果发生捕获事件时 CCxIF 标志已处于高位，则会将重复捕获标志 CCxOF（TIMx_SR 寄存器）置 1。可通过软件向 CCxIF 写入 0 来给 CCxIF 清零，或读取存储在 TIMx_CCRx 寄存器中的已捕获数据。向 CCxOF 写入 0 后会将其清零。

以下示例说明了如何在 tim_ti1 输入出现上升沿时将计数器的值捕获到 TIMx_CCR1 中。具体操作步骤如下：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:0]源（内部或外部）。
2. 选择有效输入：TIMx_CCR1 必须连接到 tim_ti1 输入，因此向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01。只要 CC1S 不等于 00，就会将通道配置为输入模式，并且 TIMx_CCR1 寄存器将处于只读状态。
3. 根据连接到定时器的信号，对所需的输入滤波时间进行编程（如果输入为 tim_tix 输入之一，则对 TIMx_CCMRx 寄存器中的 ICxF 位进行编程）。假设信号变化时，输入信号最多在 5 个内部时钟周期内发生抖动。因此，我们必须将滤波时间设置为大于 5 个内部时钟周期。在检测到 8 个具有新电平的连续采样（以 f_{DTS} 频率采样）后，可以确认 tim_ti1 上的跳变沿。然后向 TIMx_CCMR1 寄存器中的 IC1F 位写入 0011。
4. 通过向 TIMx_CCER 寄存器中的 CC1P 位和 CC1NP 位写入 0，选择 tim_ti1 通道的有效转换边沿（本例中为上升沿）。
5. 对输入预分频器进行编程。在本例中，我们希望每次有效转换时都执行捕获操作，因此需要禁止预分频器（向 TIMx_CCMR1 寄存器中的 IC1PSC 位写入 00）。
6. 通过将 TIMx_CCER 寄存器中的 CC1E 位置 1，允许将计数器的值捕获到捕获寄存器中。

7. 如果需要，可通过将 TIMx_DIER 寄存器中的 CC1IE 位置 1 来使能相关中断请求。

发生输入捕获时：

- 发生有效跳变沿时，TIMx_CCR1 寄存器会获取计数器的值。
- 将 CC1IF 标志置 1（中断标志）。如果至少发生了两次连续捕获，但 CC1IF 标志未被清零，这样 CC1OF 捕获溢出标志会被置 1。
- 根据 CC1IE 位生成一个中断。

要处理重复捕获，建议在读出捕获溢出标志之前读取数据。这样可避免丢失在读取捕获溢出标志之后与读取数据之前可能出现的重复捕获信息。

*注：通过软件将*TIMx_EGR *寄存器中的相应*CCxG *位置*1 *可生成*IC *中断请求。*

21.3.7 捕获通道

每个捕获通道均围绕一个捕获/比较寄存器（包括一个影子寄存器）和一个捕获输入阶段（数字滤波、多路复用和预分频器）构建而成。

下图概括介绍了一个捕获通道。

输入阶段对相应的 tim_tix 输入进行采样，生成一个滤波后的信号 tim_tixf。然后，带有极性选择功能的边沿检测器生成一个信号（tim_tixfpy），该信号可用作从模式控制器的触发输入，也可用作捕获命令。该信号先进行预分频（ICxPS），而后再进入捕获寄存器。

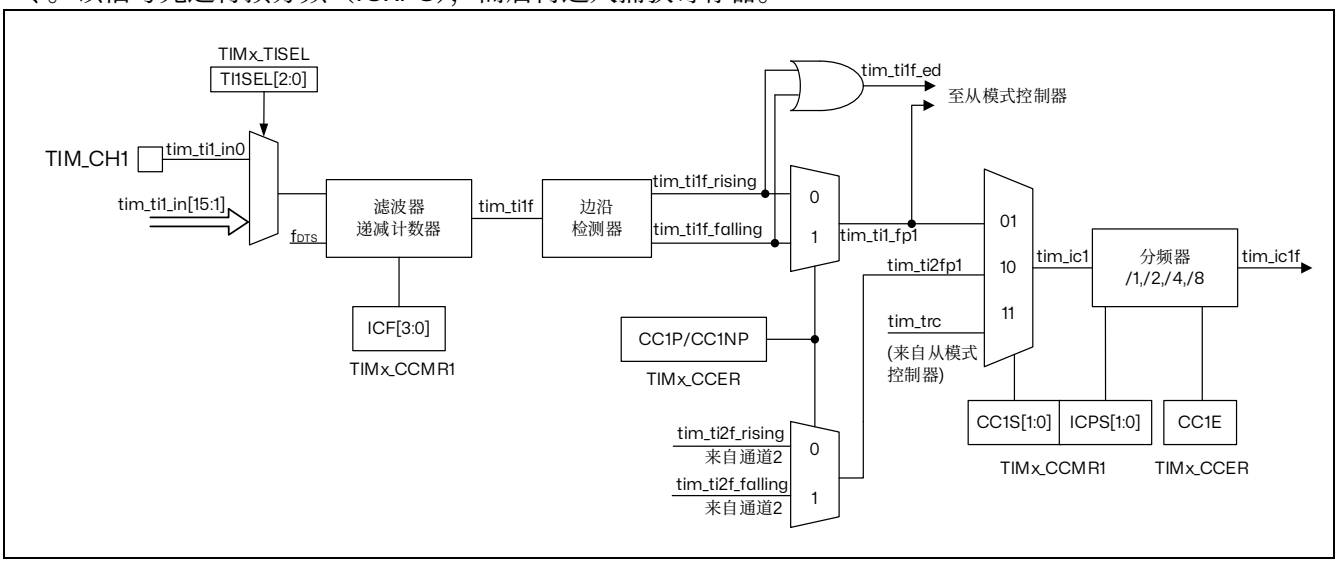


图 21.23 捕获通道（如：通道 1 输入部分）

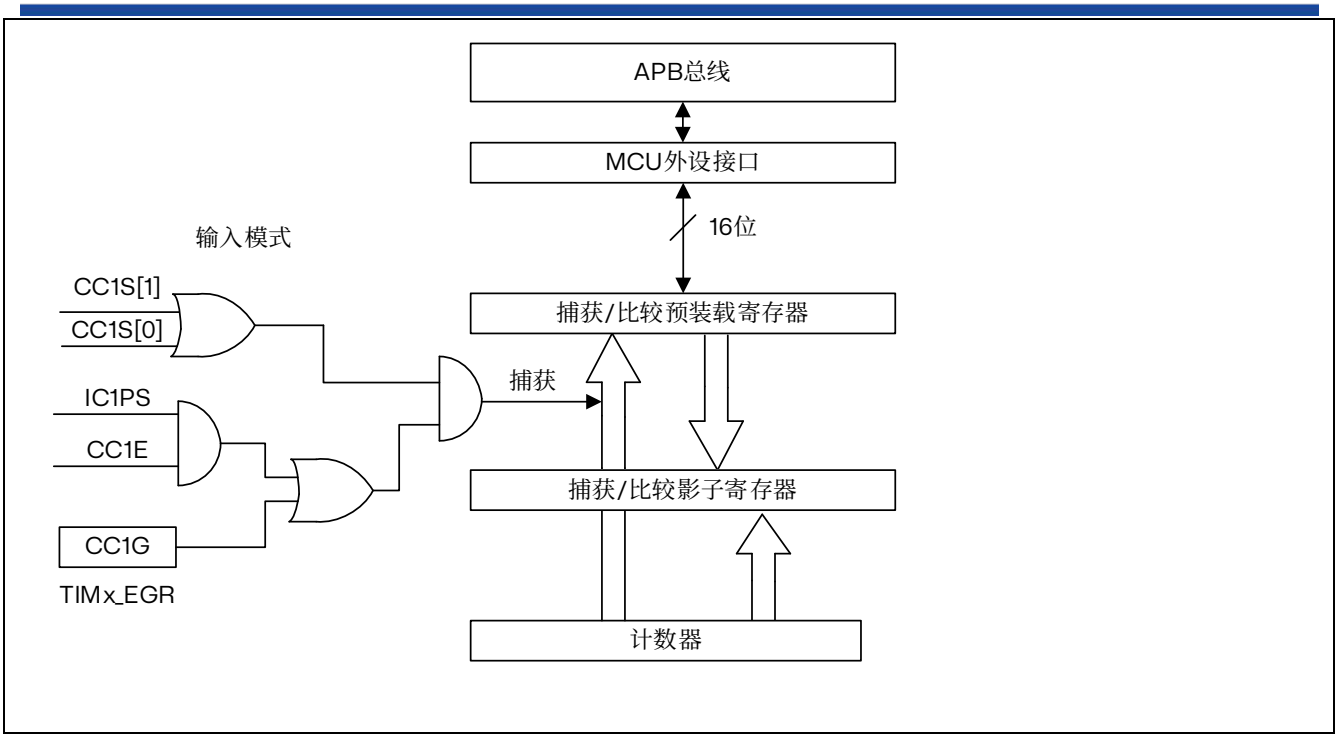


图 21.24 捕获通道 1 的主电路

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。始终可通过读写操作访问预装载寄存器。在捕获模式下，捕获实际发生在影子寄存器中，然后将影子寄存器的内容复制到预装载寄存器中。

21.3.8 PWM 输入模式

此模式允许测量连接到单个 tim_tix 输入的 PWM 信号的周期和占空比：

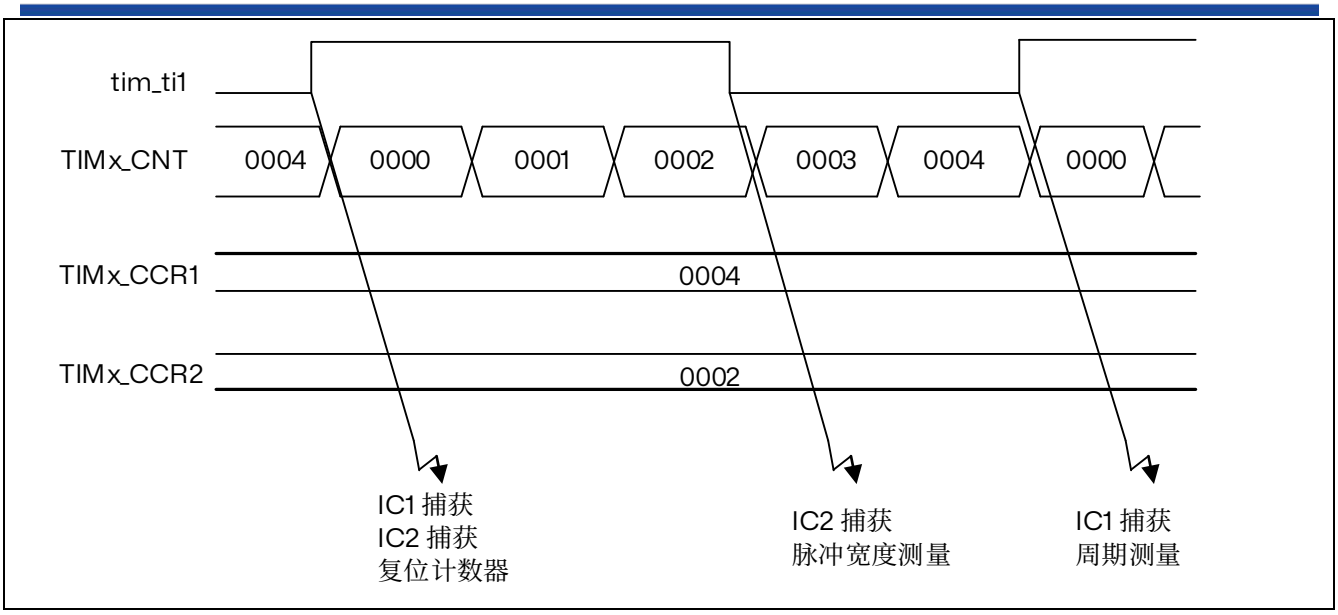
- TIMx_CCR1 寄存器保持周期值（两个连续上升沿之间的间隔）。
- TIMx_CCR2 寄存器保持脉冲宽度（两个连续上升沿和下降沿之间的间隔）。

此模式是输入捕获模式的一个特例。其实现步骤与输入捕获模式基本相同，仅存在以下不同之处：

- 两个 ICx 信号被映射至同一个 tim_tix 输入。
- 这两个 ICx 信号在边沿处有效，但极性相反。
- 选择两个 TIxFP 信号之一作为触发输入，并将从模式控制器配置为复位模式。

可通过以下步骤测量施加在 tim_ti1 上的 PWM 信号的周期和脉冲宽度：

1. 通过在 TIMx_TISEL 寄存器中使用 TI1SEL[2:0]位选择合适的 tim_tix_in[15:0]源（内部或外部）。
2. 选择 TIMx_CCR1 的有效输入：向 TIMx_CCMR1 寄存器中的 CC1S 位写入 01（选择 tim_ti1）。
3. 选择 tim_ti1fp1 的有效极性（用于 TIMx_CCR1 中的捕获和计数器清零）：向 CC1P 位和 CC1NP 位写入“0”（上升沿有效）。
4. 选择 TIMx_CCR2 的有效输入：向 TIMx_CCMR1 寄存器中的 CC2S 写入 10（选择 tim_ti1）。
5. 选择 tim_ti1fp2 的有效极性（用于 TIMx_CCR2 中的捕获）：向 CC2P 位写入“1”和 CC2NP 位写入“0”（下降沿有效）。
6. 选择有效触发输入：向 TIMx_SMCR 寄存器中的 TS 位写入 101（选择 tim_ti1fp1）。
7. 将从模式控制器配置为复位模式：向 TIMx_SMCR 寄存器中的 SMS 位写入 100。
8. 使能捕获：向 TIMx_CCER 寄存器中的 CC1E 位和 CC2E 位写入“1”。



PWM 输入模式只能与 TIMx_CH1/TIMx_CH2 信号一起使用，因为只有 tim_ti1fp1 和 tim_ti2fp2 连接到从模式控制器。

图 21.25 PWM 输入模式时序

21.3.9 编码器接口模式

正交编码器

选择编码器接口模式时，如果计数器仅在 tim_ti2 边沿处计数，在 TIMx_SMCR 寄存器中写入 SMS=0001；如果计数器仅在 tim_ti1 边沿处计数，写入 SMS=0010；如果计数器在 tim_ti1 和 tim_ti2 边沿处均计数，则写入 SMS=0011。

通过编程 TIMx_CCER 寄存器的 CC1P 和 CC2P 位，选择 tim_ti1 和 tim_ti2 极性。如果需要，还可对输入滤波器进行编程。

tim_ti1 和 tim_ti2 两个输入用于连接增量编码器。请参见下表。如果使能计数器（在 TIMx_CR1 寄存器的 CEN 位中写入“1”），则计数器的时钟由 tim_ti1fp1 或 tim_ti2fp2 上的每次有效信号转换提供。tim_ti1fp1 和 tim_ti2fp2 是进行输入滤波器和极性选择后 tim_ti1 和 tim_ti2 的信号，如果不进行滤波和反相，则 tim_ti1fp1=tim_ti1，tim_ti2fp2=tim_ti2。将根据两个输入的信号转换序列，产生计数脉冲和方向信号。根据该信号转换序列，计数器相应递增或递减计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应修改。任何输入（tim_ti1 或 tim_ti2）发生信号转换时，都会计算 DIR 位，无论计数器是仅在 tim_ti1 或 tim_ti2 边沿处计数，还是同时在 tim_ti1 和 tim_ti2 处计数。

编码器接口模式就相当于带有方向选择的外部时钟。这意味着，计数器仅在 0 到 TIMx_ARR 寄存器中的自动重载值之间进行连续计数（根据具体方向，从 0 递增计数到 ARR，或从 ARR 递减计数到 0）。因此，在启动前必须先配置 TIMx_ARR。同样，捕获、比较、预分频器、触发输出功能继续正常工作。

在此模式下，计数器会根据增量编码器的速度和方向自动进行修改，因此，其内容始终表示编码器的位置。计数方向对应于所连传感器的旋转方向。下表汇总了可能的组合（假设 tim_ti1 和 tim_ti2 不同时切换）。

表 21.8 计数方向与编码器信号的关系（CC1P = CC2P = 0）

有效边沿	SMS[3:0]	相对信号的电平（tim_ti1fp1 对应 tim_ti2, tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
仅 x1 模式在 tim_ti1 计数	1110	高	递减	递增	不计数	不计数
		低	不计数	不计数	不计数	不计数
仅 x1 模式在 tim_ti2 计数	1111	高	不计数	不计数	递增	递减
		低	不计数	不计数	不计数	不计数
仅 x2 模式在 tim_ti1 计数	0001	高	递减	递增	不计数	不计数
		低	递增	递减	不计数	不计数
仅 x2 模式在 tim_ti2 计数	0010	高	不计数	不计数	递增	递减
		低	不计数	不计数	递减	递增
x4 模式在 tim_ti1 和 tim_ti2 计数	0011	高	递减	递增	递增	递减
		低	递增	递减	递减	递增

外部增量编码器可直接与 MCU 相连，无需外部接口逻辑。不过，通常使用比较器将编码器的差分输出转换为数字信号。这样大幅提高了抗噪声性能。用于指示机械零位的第三个编码器输出可与外部中断输入相连，用以触发计数器复位。

下图以计数器工作为例，说明了计数信号的生成和方向控制。同时也说明了选择双边沿时如何对输入抖动进行补偿。将传感器靠近其中一个切换点放置时可能出现这种情况。本例中假设配置如下：

- CC1S=“01”（TIMx_CCMR1 寄存器，tim_ti1fp1 映射到 tim_ti1 上）。
- CC2S=“01”（TIMx_CCMR2 寄存器，tim_ti2fp2 映射到 tim_ti2 上）。
- CC1P=“0”，CC1NP=“0”（TIMx_CCER 寄存器，tim_ti1fp1 未反相，tim_ti1fp1= tim_ti1）。
- CC2P=“0”，CC2NP=“0” TIMx_CCER 寄存器，tim_ti2fp2 未反相，tim_ti2fp2= tim_ti2）。
- SMS=“011”（TIMx_SMCR 寄存器，两个输入在上升沿和下降沿都处于有效状态）。
- CEN=“1”（TIMx_CR1 寄存器，使能计数器）。

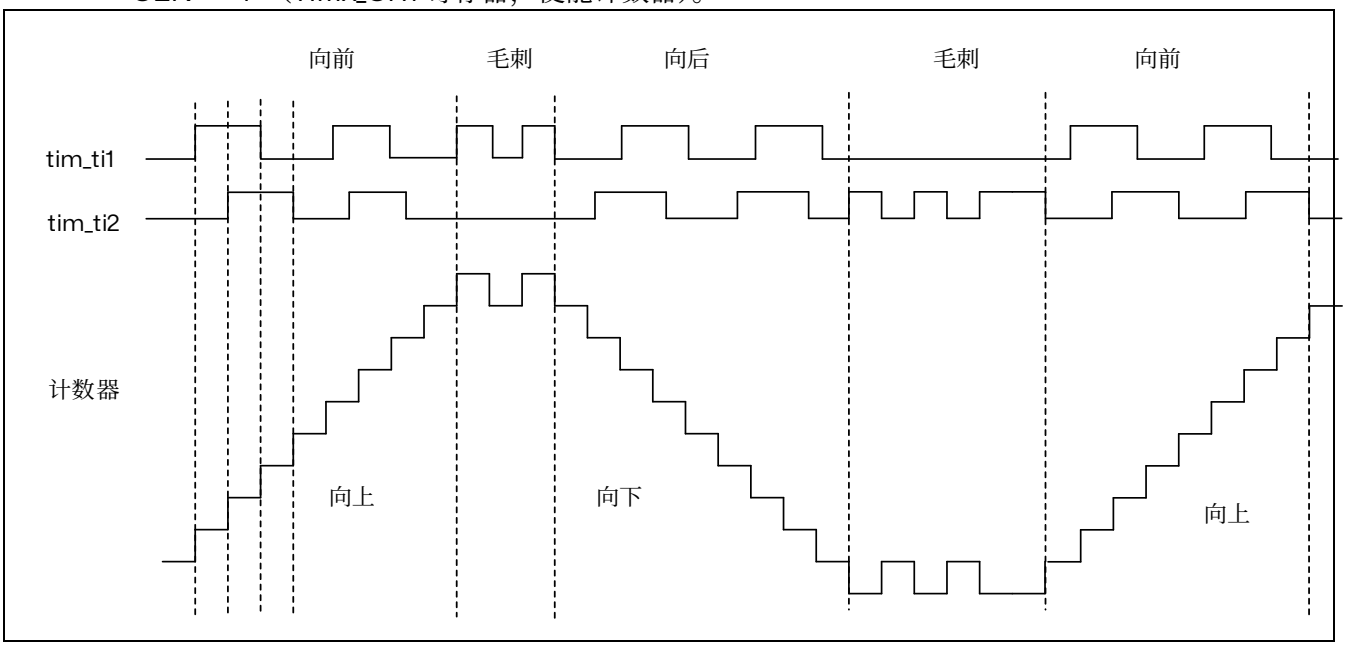


图 21.26 编码器接口模式下的计数器工作示例

下图举例说明 tim_ti1fp1 极性反相时计数器的行为（除 CC1P=“1”外，其它配置与上例相同）。

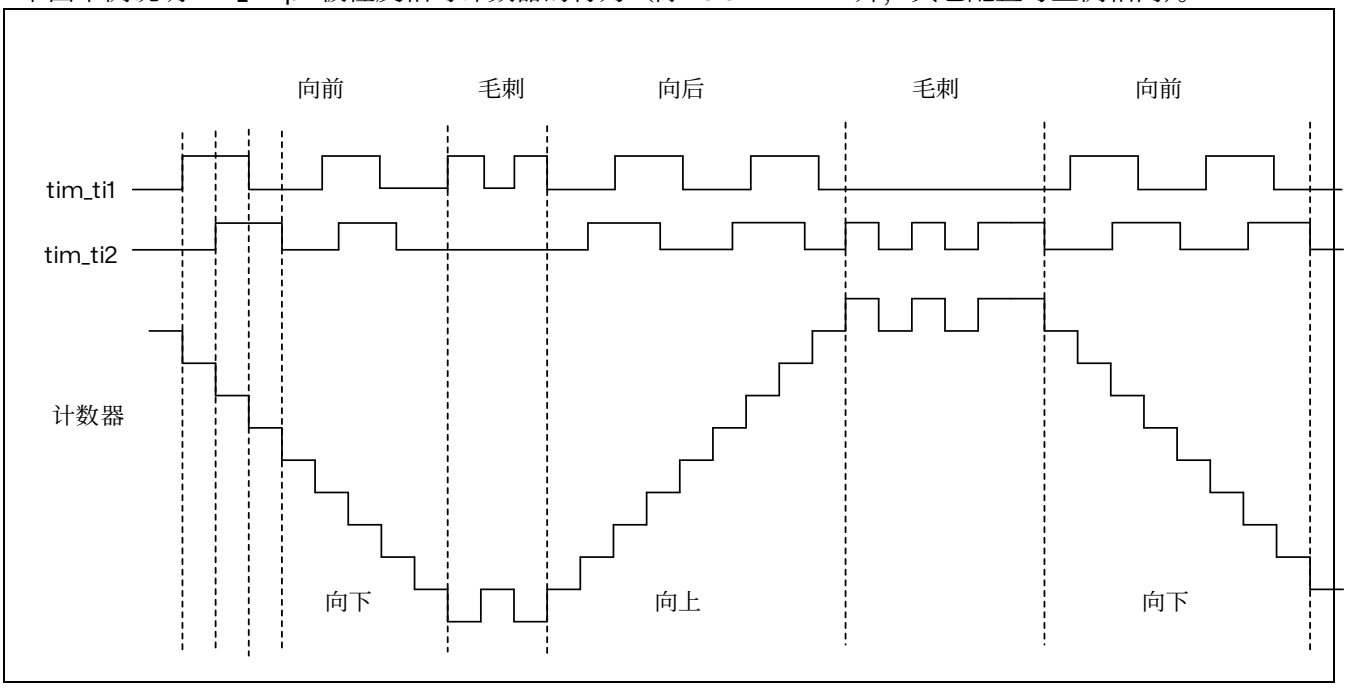


图 21.27 tim_ti1fp1 极性反相时的编码器接口模式示例

下图显示了各种计数模式下速度反转期间计时器计数器的值。

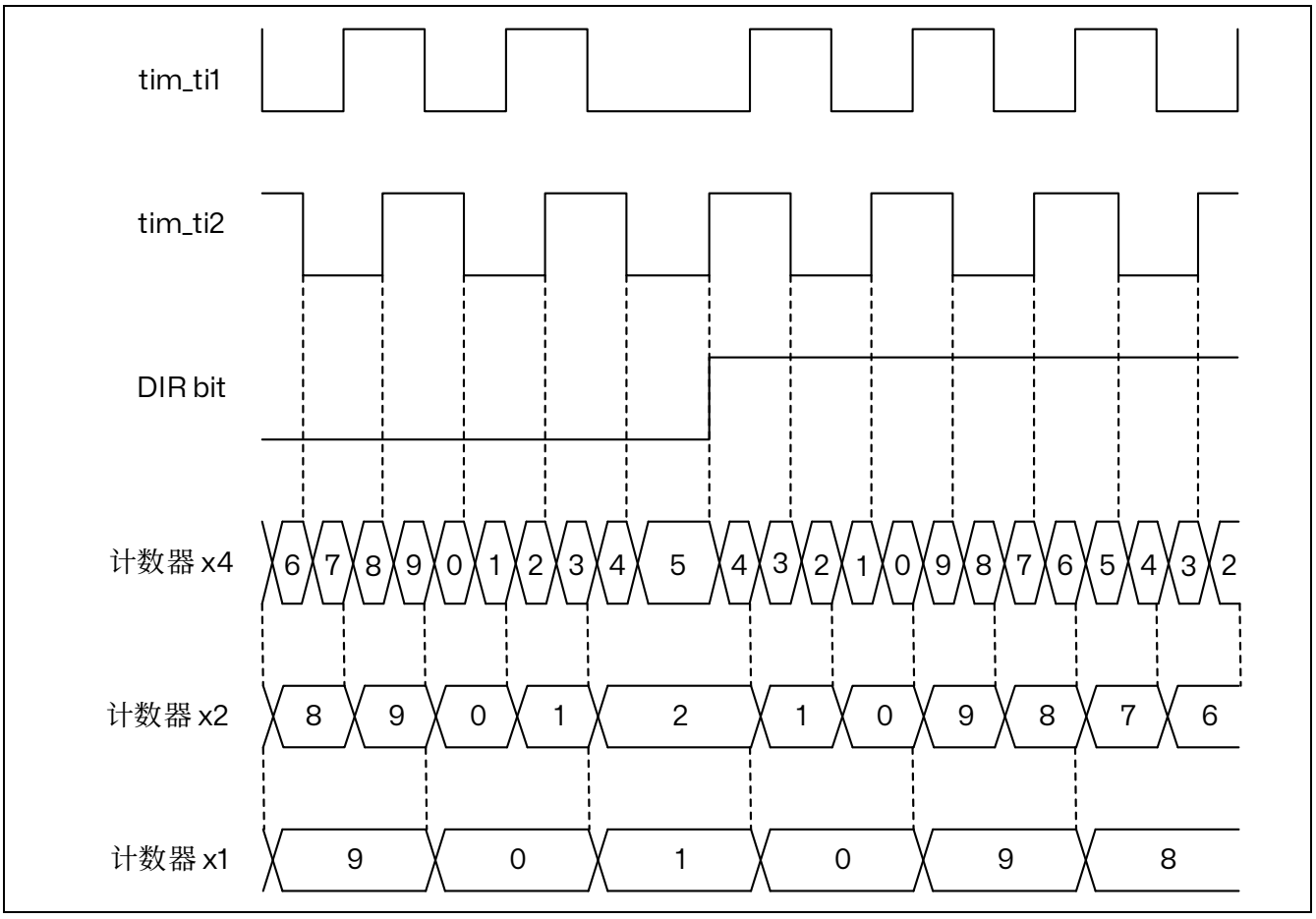


图 21.28 正交编码器的计数模式

当定时器配置为编码器接口模式时，可以提供传感器当前位置的信息。通过测量两个编码器事件之间的周期，使用配置为捕获模式的第二个定时器，可以获得动态信息（速度、加速度、减速度）。指示机械零的编码器输出可用于此目的。根据两个事件之间的时间，还可以定期读取计数器的值。如果可用，可以通过将计数器的值锁存到第三个输入捕获寄存器来实现（此时捕获信号必须是周期性的，并且可以由另一个定时器生成）。

TIMx_CR1 寄存器中的 IUFREMAP 位将更新中断标志（UIF）的连续副本强制复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许原子地读取计数器的值和由 UIFCPY 标志触发的潜在的翻转条件。这可以通过避免在后台任务（计数器读取）和中断（更新中断）之间共享处理来简化角速度的计算。

UIF 和 UIFCPY 标志之间没有延迟。

在 32 位定时器实现中，当 IUFREMAP 位被设置时，在读取访问时，计数器的第 31 位将被 UIFCPY 标志覆盖（计数器最高有效位仅可在写入模式下访问）。

时钟加方向编码器模式

除了正交编码器模式外，定时器还支持其他类型的编码器。

在下图中显示的“时钟加方向”模式下，时钟在 tim_ti2 上提供单线，而方向使用 tim_ti1 输入强制。

此模式通过 TIMx_SMCR 寄存器中的 SMS[3:0]位域使能，如下所示：

- 1010：x2 模式，计数器在时钟的上升沿和下降沿都更新
- 1011：x1 模式，计数器在单个时钟边沿更新，根据 CC2P 位的值而定：CC2P=0 对应上升沿敏感，CC2P=1 对应下降沿敏感。

tim_ti1 上的方向信号极性通过 CC1P 位设置：0 对应正极性（当 tim_ti1 为高电平时向上计数，当 tim_ti1 为低电平时向下计数），CC1P=1 对应负极性（当 tim_ti1 为低电平时向上计数）。

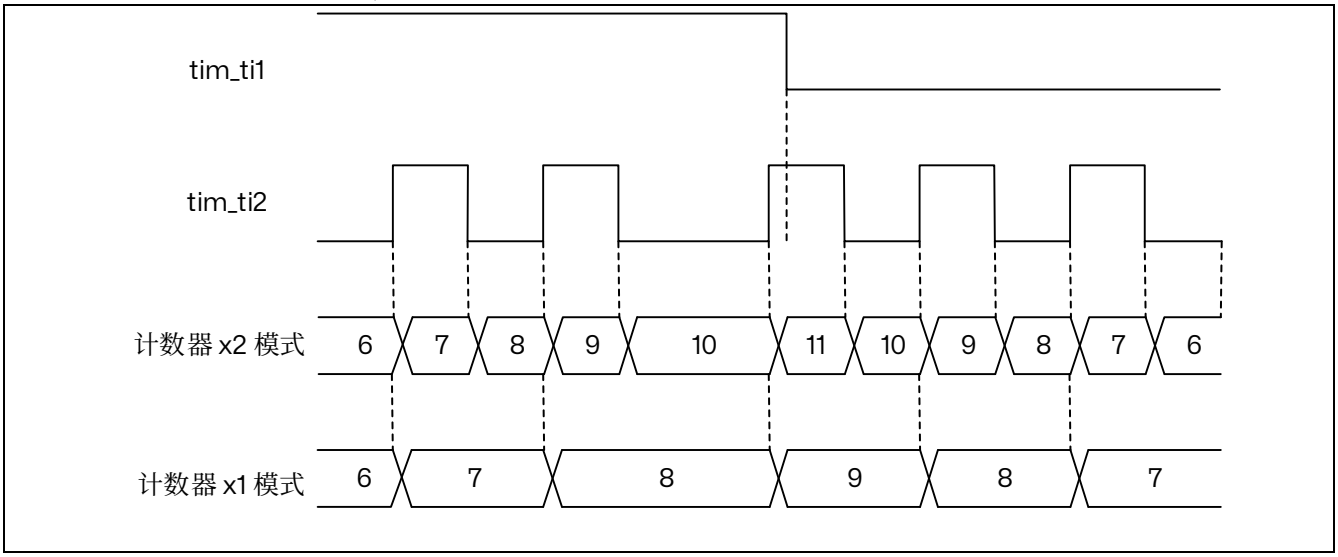


图 21.29 方向加时钟编码器模式

方向时钟编码器模式

在下图中的“方向时钟”模式下，时钟在两条线上提供，根据方向一次使用一条，以便有一条向上计数时钟线和一条向下计数时钟线。

此模式通过 TIMx_SMCR 寄存器中的 SMS[3:0]位域使能，如下所示：

- 1100：x2 模式，计数器在两条时钟线的上升沿和下降沿都更新。CC1P 和 CC2P 位编码时钟空闲状态。CCxP=0 对应高电平空闲状态（参考下图），CCxP=1 对应低电平空闲状态（参考下图）。
- 1101：x1 模式，计数器在单个时钟边沿更新，根据 CC1P 和 CC2P 位的值而定。CCxP=0 对应

下降沿敏感性和高电平空闲状态（参考下图），CCxP=1 对应上升沿敏感性和低电平空闲状态（参考下图）。

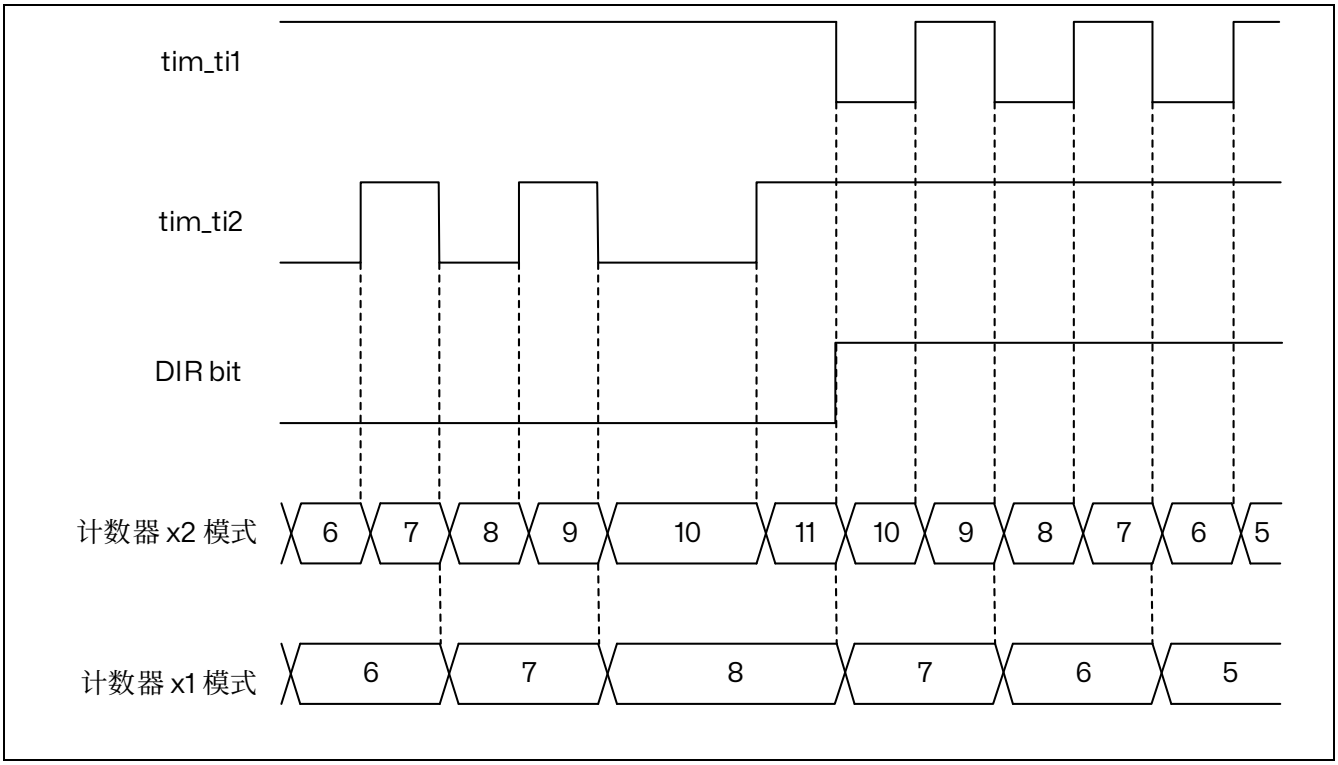


图 21.30 方向时钟编码器模式（CC1P=CC2P=0）

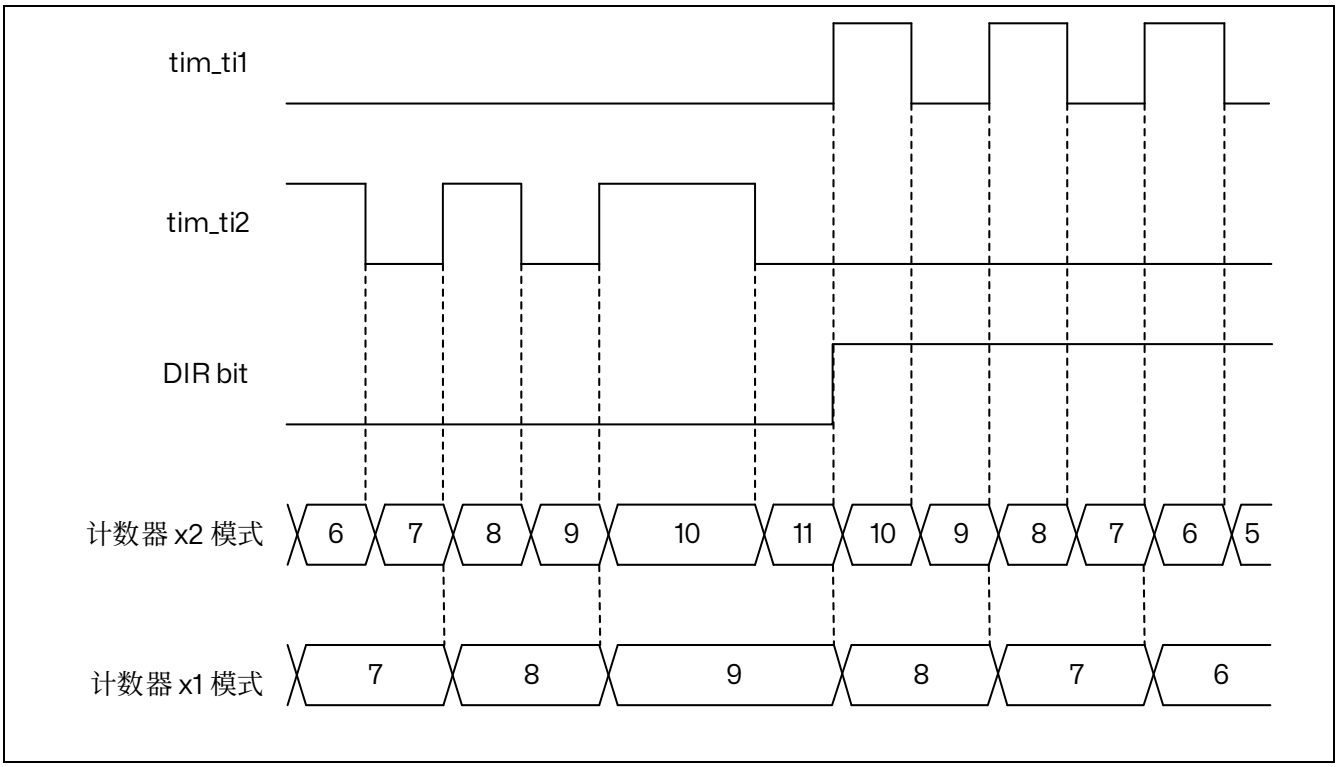


图 21.31 方向时钟编码器模式（CC1P=CC2P=1）

下表详细说明了方向时钟模式对任何输入转变的操作方式。

表 21.9 计数方向与编码器信号及极性设置的关系

方向时钟模式	SMS[3:0]	相对信号的电平（tim_ti1fp1 对应 tim_ti2, tim_ti2fp2 对应 tim_ti1）	tim_ti1fp1 信号		tim_ti2fp2 信号	
			上升	下降	上升	下降
x2 模式 CCxP=0	1100	高	递减	递减	递增	递增
		低	不计数	不计数	不计数	不计数
x2 模式 CCxP=1	1100	高	不计数	不计数	不计数	不计数
		低	递减	递减	递增	递增
x1 模式 CCxP=0	1101	高	不计数	递减	不计数	递增
		低	不计数	不计数	不计数	不计数
x1 模式 CCxP=1	1101	高	不计数	不计数	不计数	不计数
		低	递减	不计数	递增	不计数

索引输入

计数器可以通过来自编码器的索引信号进行复位，以指示绝对参考位置。索引信号必须连接到 tim_etr_in 输入。它可以使用数字输入滤波器进行滤波。

通过设置 TIMX_ECR 寄存器中的 IE 位，可以使能索引功能。只有在编码器模式下，当 SMS[3:0]位域具有以下值时，才可以设置 IE 位：0001、0010、0011、1010、1011、1100、1101、1110、1111。

根据下图所示，有几种可设置的编码器选项，用于索引脉冲的调节：

- 与 A 和 B 门控：脉冲宽度为 1/4 的通道周期，与 A 和 B 边缘对齐
- 与 A（或与 B）门控：脉冲宽度为 1/2 的通道周期，与通道 A（或通道 B）的两个边缘对齐
- 非门控：脉冲宽度可达一个通道周期，不与任何边缘对齐

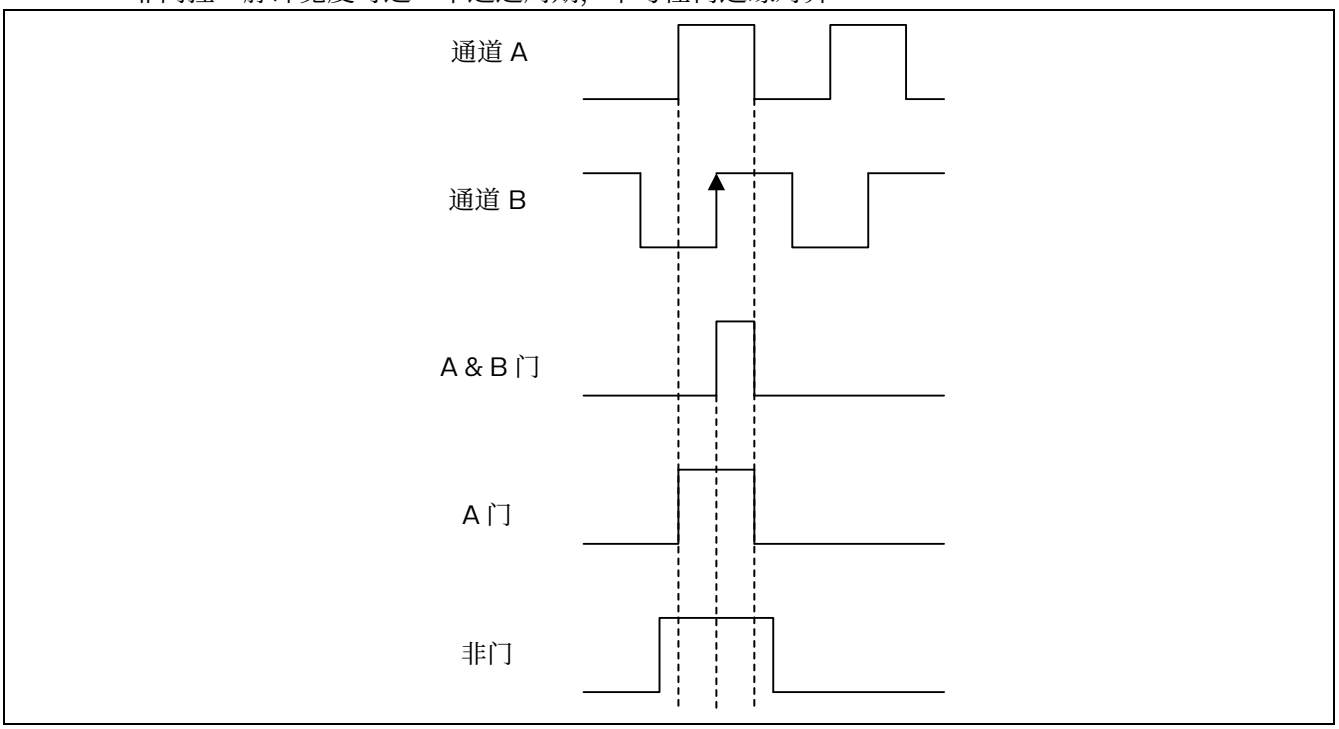


图 21.32 索引门控选项

电路可以容忍索引信号的抖动，无论门控模式如何，如下图所示。

在非门控模式下，信号必须严格低于 2 个编码器周期。如果脉冲宽度大于或等于 2 个编码器周期，则

计数器会多次复位。

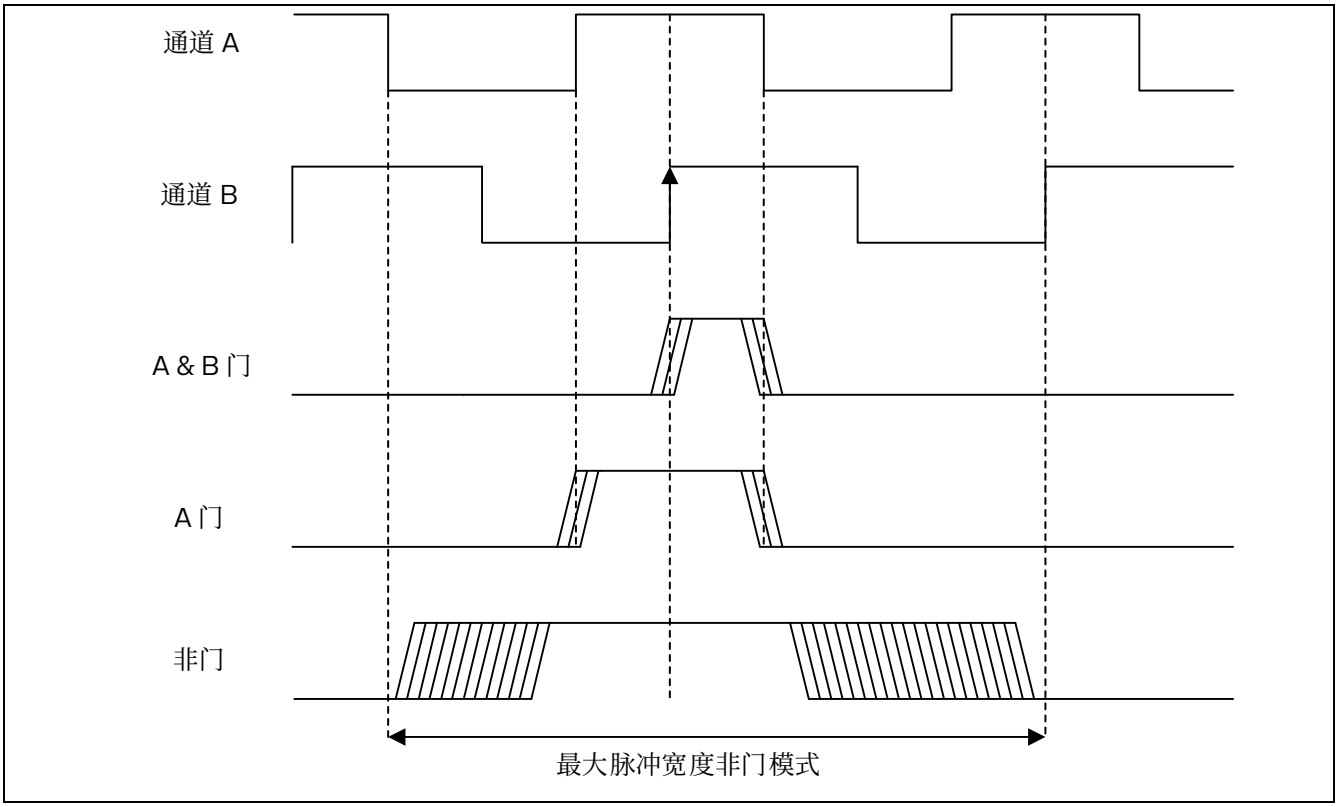


图 21.33 抖动的索引信号

定时器支持 3 种门控选项，不需要任何特定的编程。只需要定义在哪个编码器状态（即通道 A 和通道 B 状态组合）上，索引必须同步，使用 TIMx_ECR 寄存器中的 IPOS[1:0]位域。

索引检测事件根据计数方向的不同而有所不同，以确保速度反转时的对称操作：

- 递增计数时（DIR 位=0），计数器被复位。
- 递减计数时，计数器被设置为 TIMx_ARR。

这使得索引可以在相同的机械角度位置上生成，无论计数方向如何。下图显示了在一个简单的例子中（一个编码器每次机械旋转提供 4 个边缘），索引在哪个位置生成。

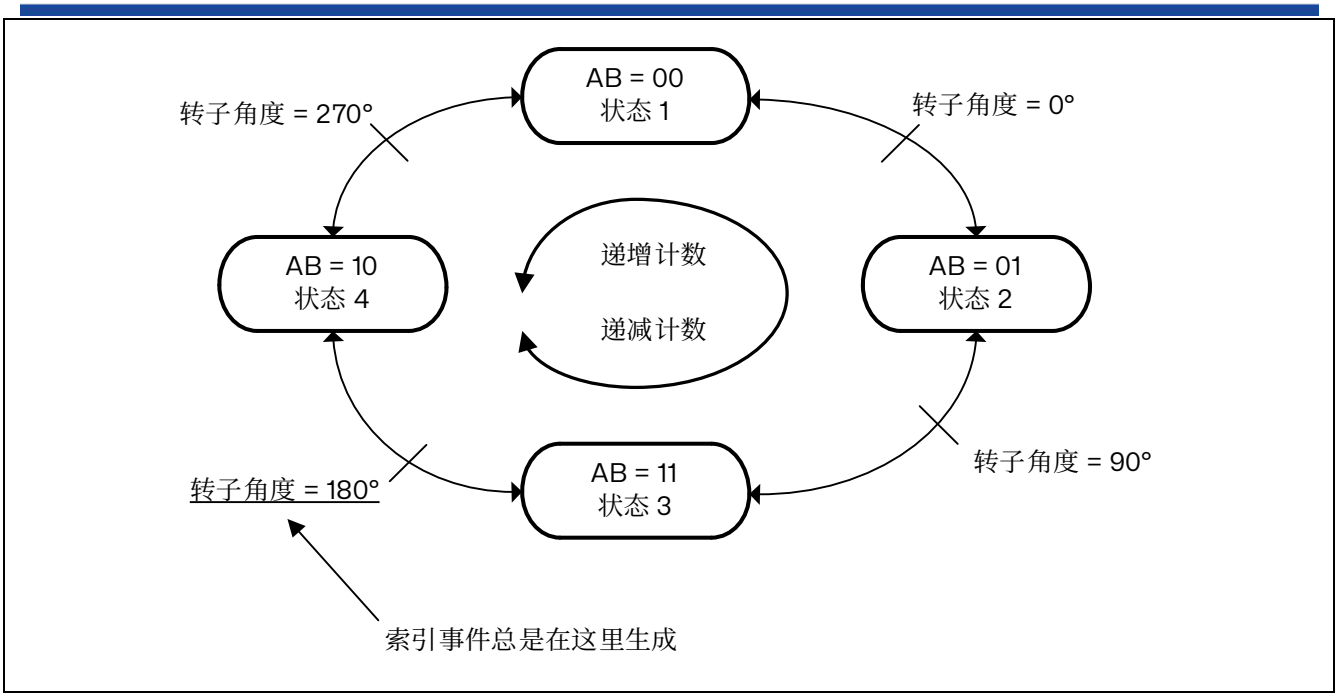


图 21.34 IPOS[1:0] = 11 时的索引生成

下图展示了 IPOS[1:0] = 11 时的波形和相应值。它表明，计数器值被强制的时刻会自动根据计数方向进行调整：

- 递增计数时（DIR 位=0），当编码器状态为“11”（ChA=1，ChB=1）时，计数器设为 0。
- 递减计数时（DIR 位=1），当退出“11”状态时，计数器设为 TIMx_ARR。

可以在索引检测事件发生时发出中断。

箭头指示在哪个转换上生成索引事件中断。

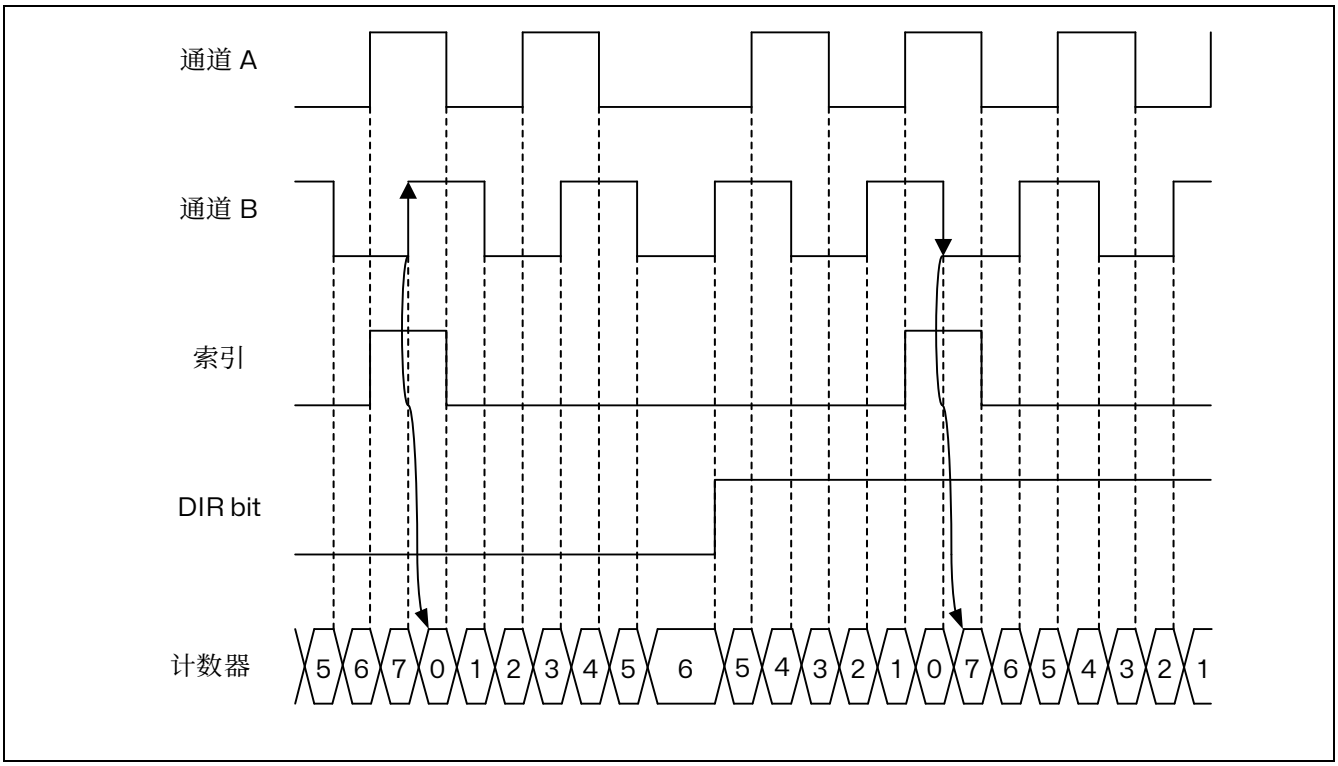


图 21.35 通道 A 上带索引门控的计数器读取（IPOS[1:0] = 11）

下图展示了非门控模式的波形和相应值。箭头指示在哪个转换上生成索引事件。

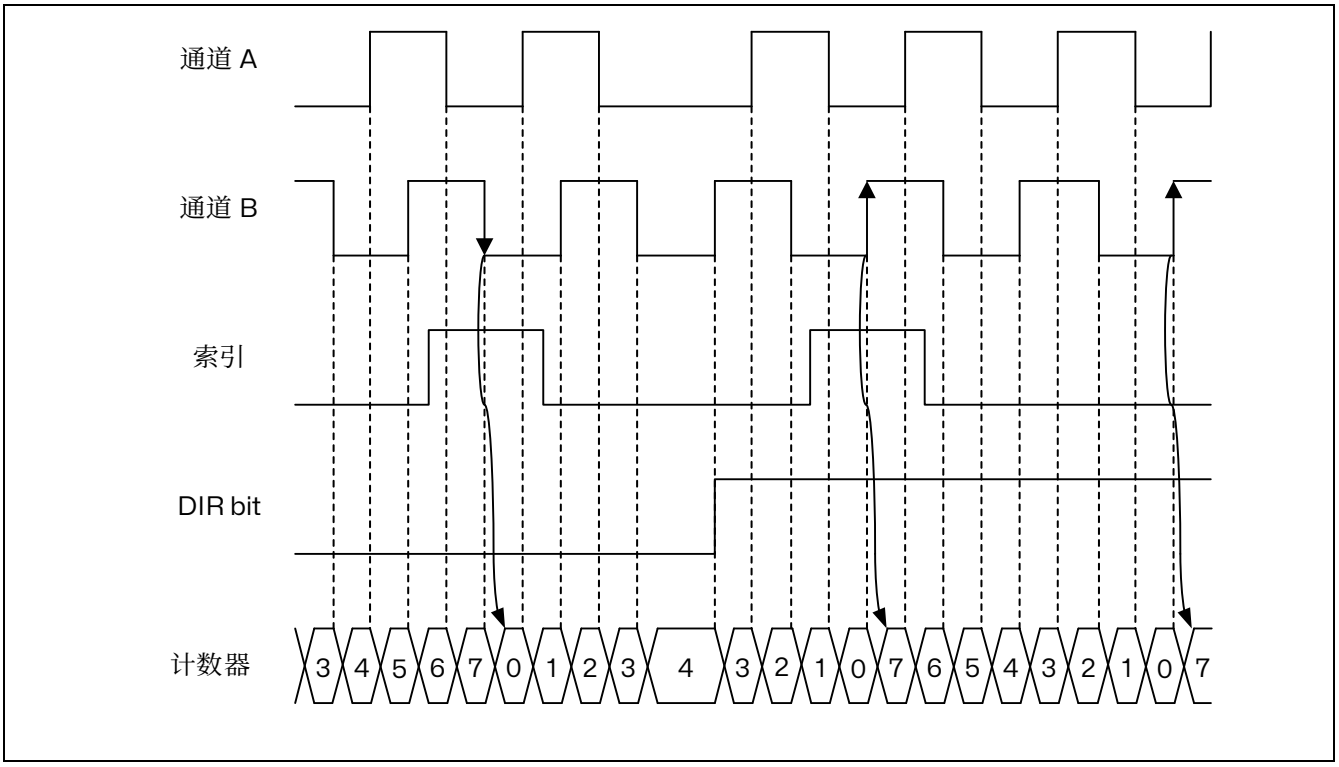


图 21.36 非门控模式下索引的计数器读取（IPOS[1:0] = 00）

下图展示了针对各种脉冲对齐情况下的“在 A 和 B 门控”模式处理方式。箭头指示在哪个转换上生成索引事件。

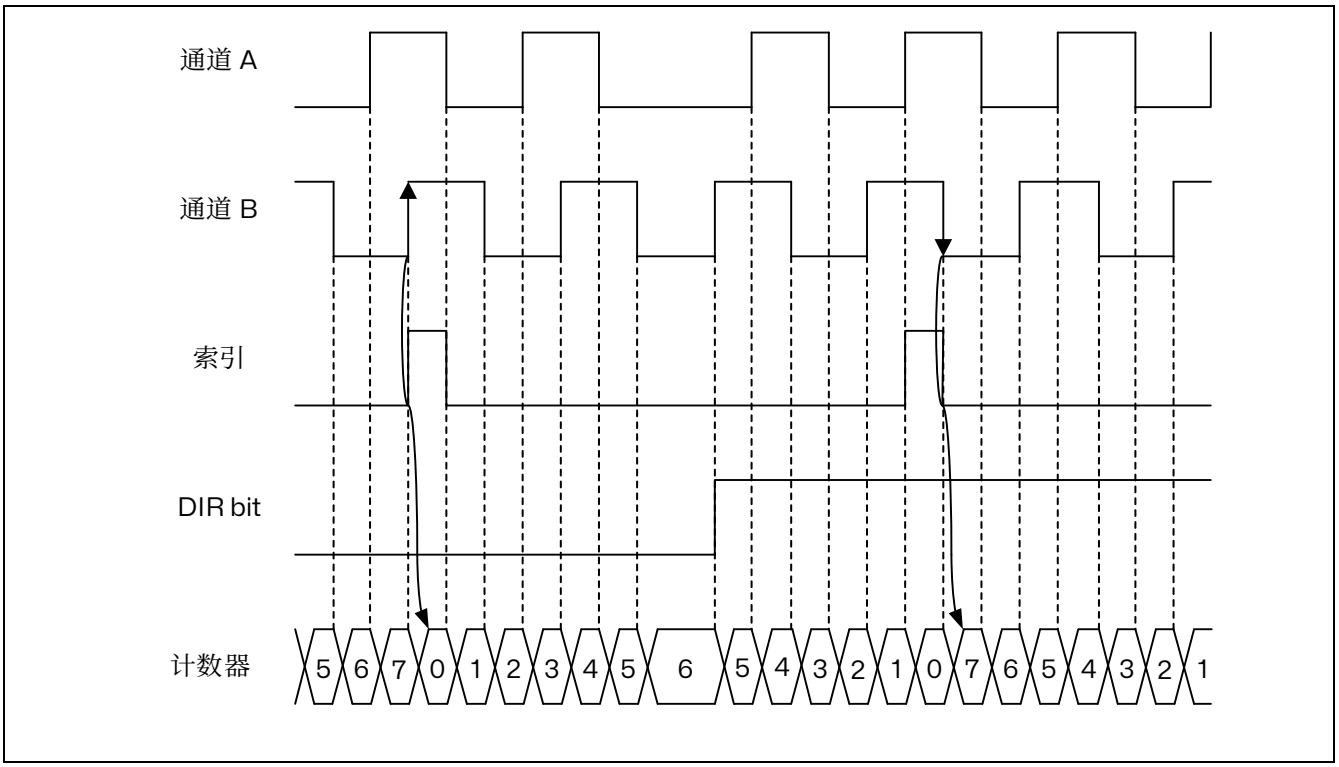
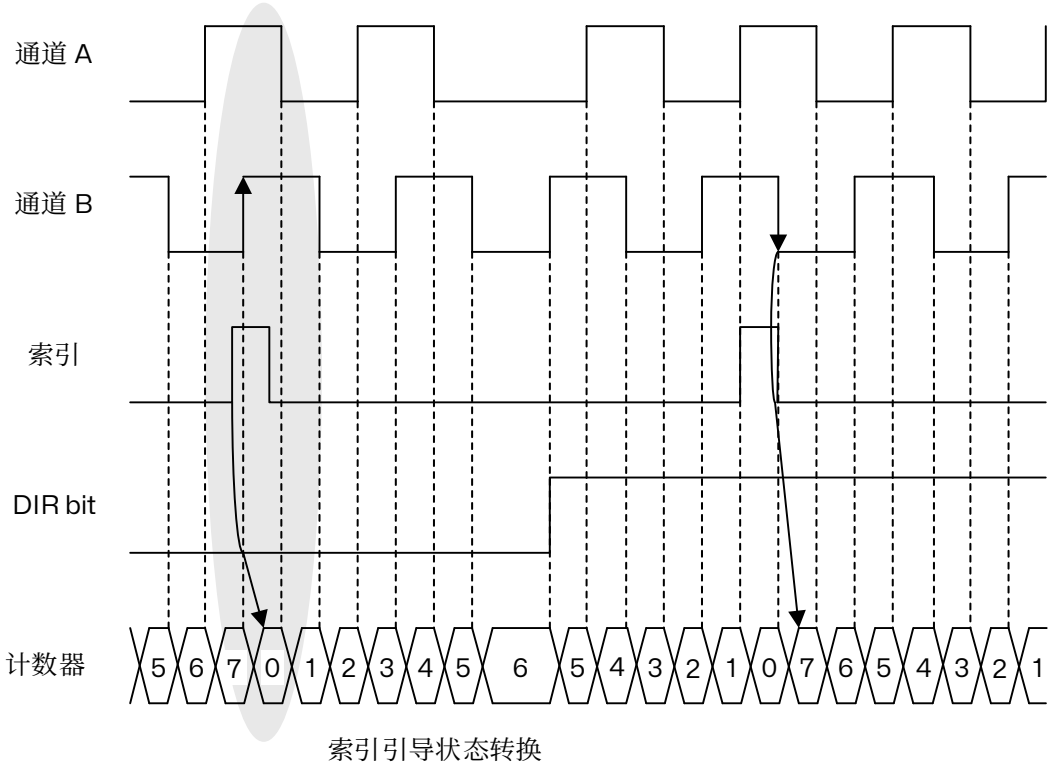
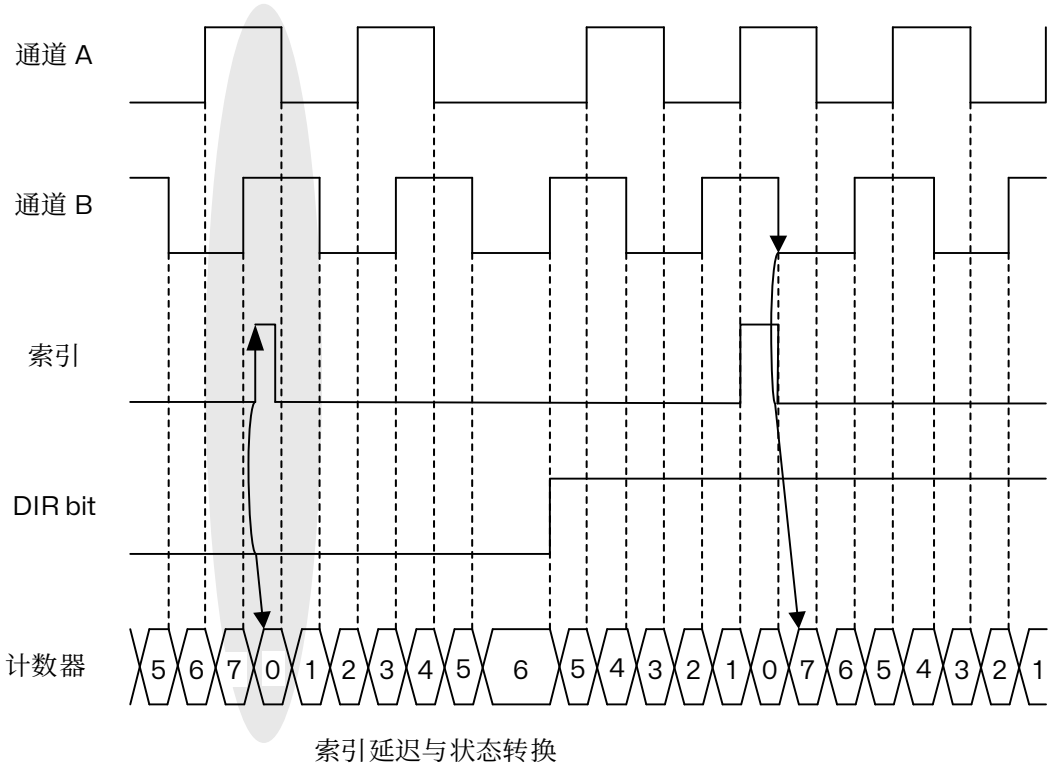


图 21.37 通道 A 和 B 上带索引门控的计数器读取

下两张图详细说明了后续索引脉冲可能比编码器时钟周期的四分之一窄的情况。



索引引导状态转换



索引延迟与状态转换

图 21.38 窄索引脉冲情况下的编码器模式行为（IPOS[1:0] = 11）

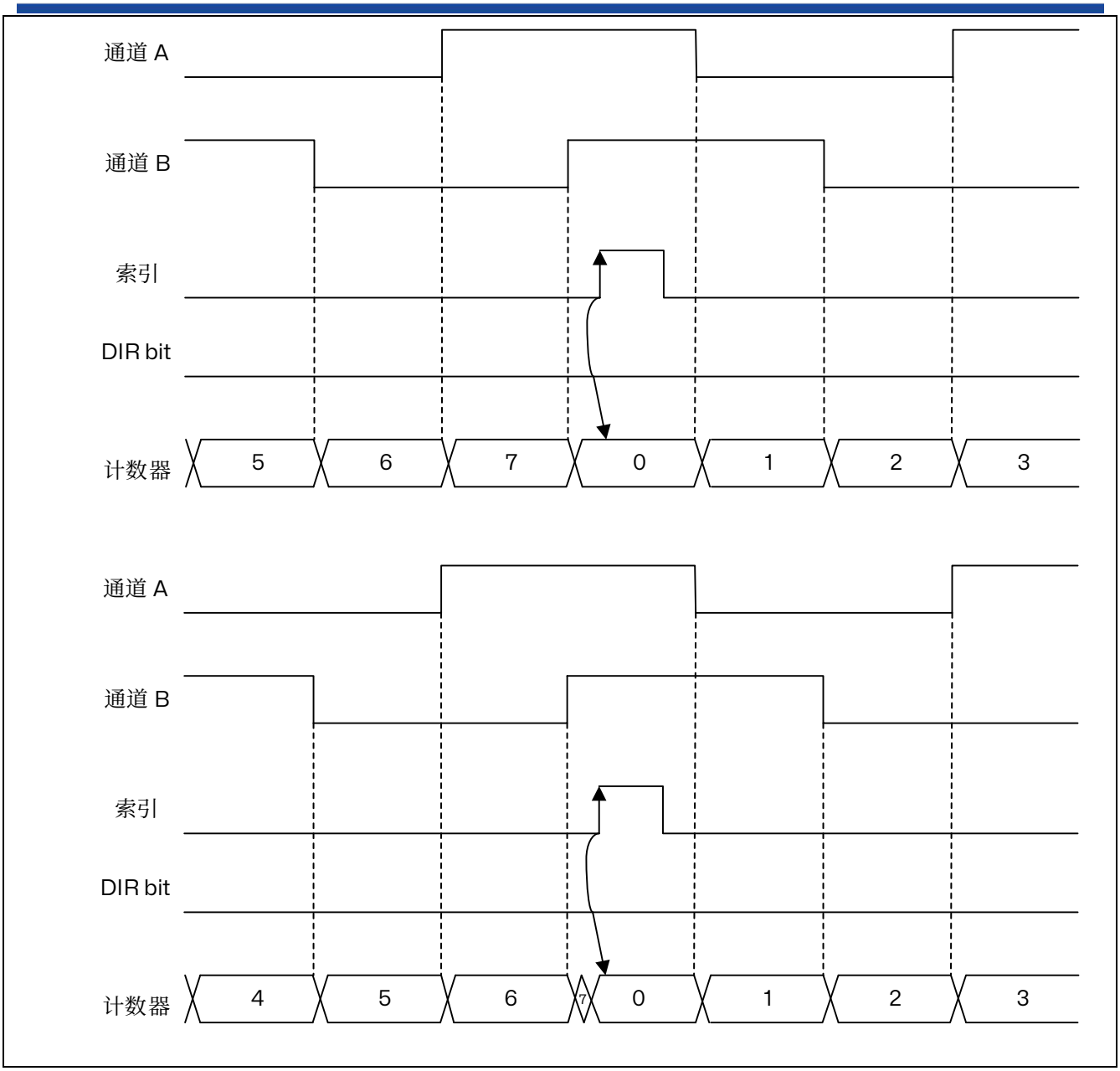


图 21.39 计数器重置窄索引脉冲（更近距离，ARR = 0x07）

下图说明了 x1 和 x2 模式下的索引管理方式。

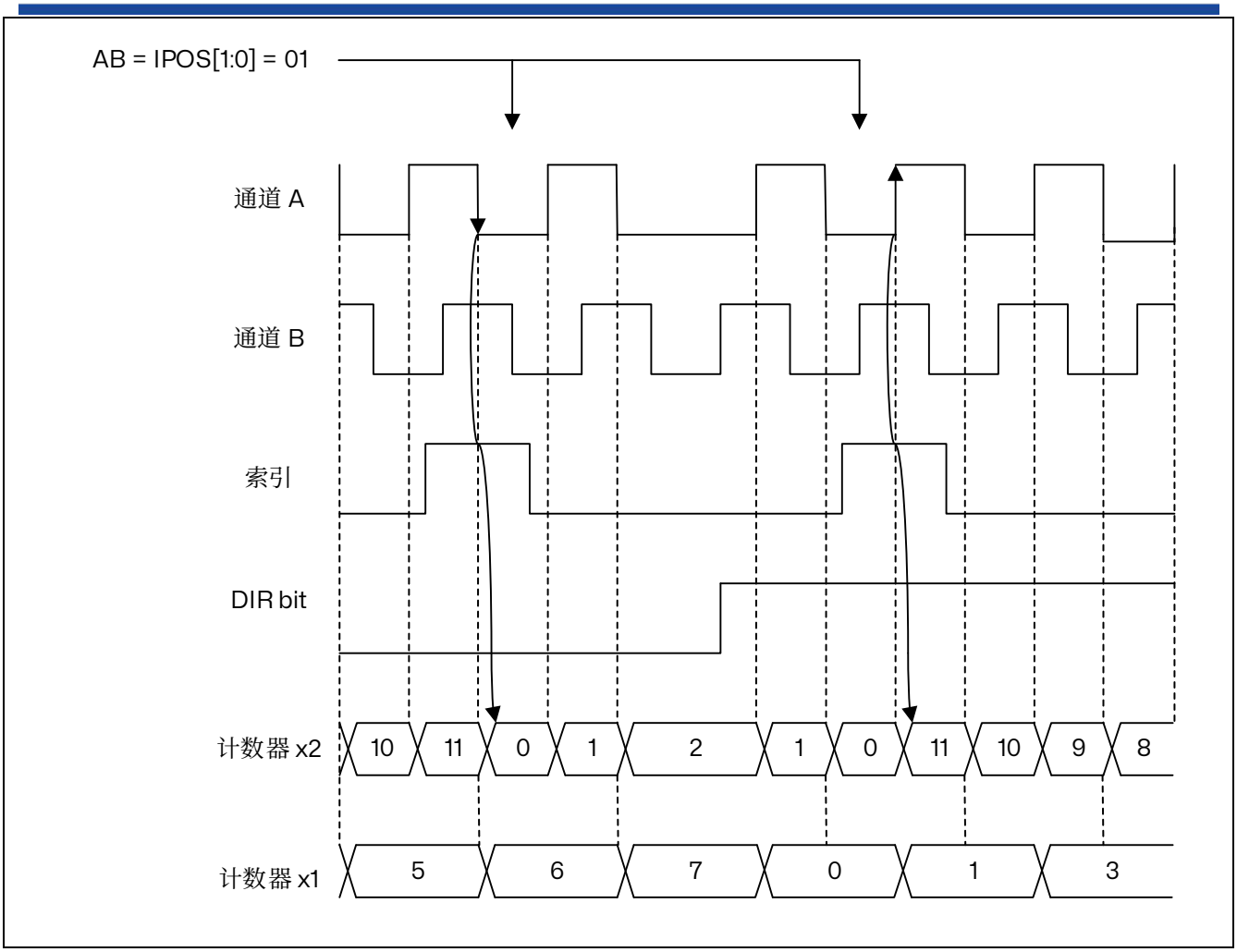


图 21.40 x1 和 x2 模式下的索引行为（IPOS[1:0] = 01）

方向索引灵敏度

TIMx_ECR 寄存器中的 IDIR[1:0]位域允许索引仅在选定计数方向上有效。

下图显示了根据 IDIR[1:0]值索引和计数器重置事件之间的关系。

注：当 IE *位被复位时（禁用索引模式），必须写入* IDR[1:0] *位域。*

注：在时钟+方向模式下不支持定向索引灵敏度。当 SMS[3:0] = 1010 *或* 1011 *时，* IDIR[1:0] *必须设置为* 00。

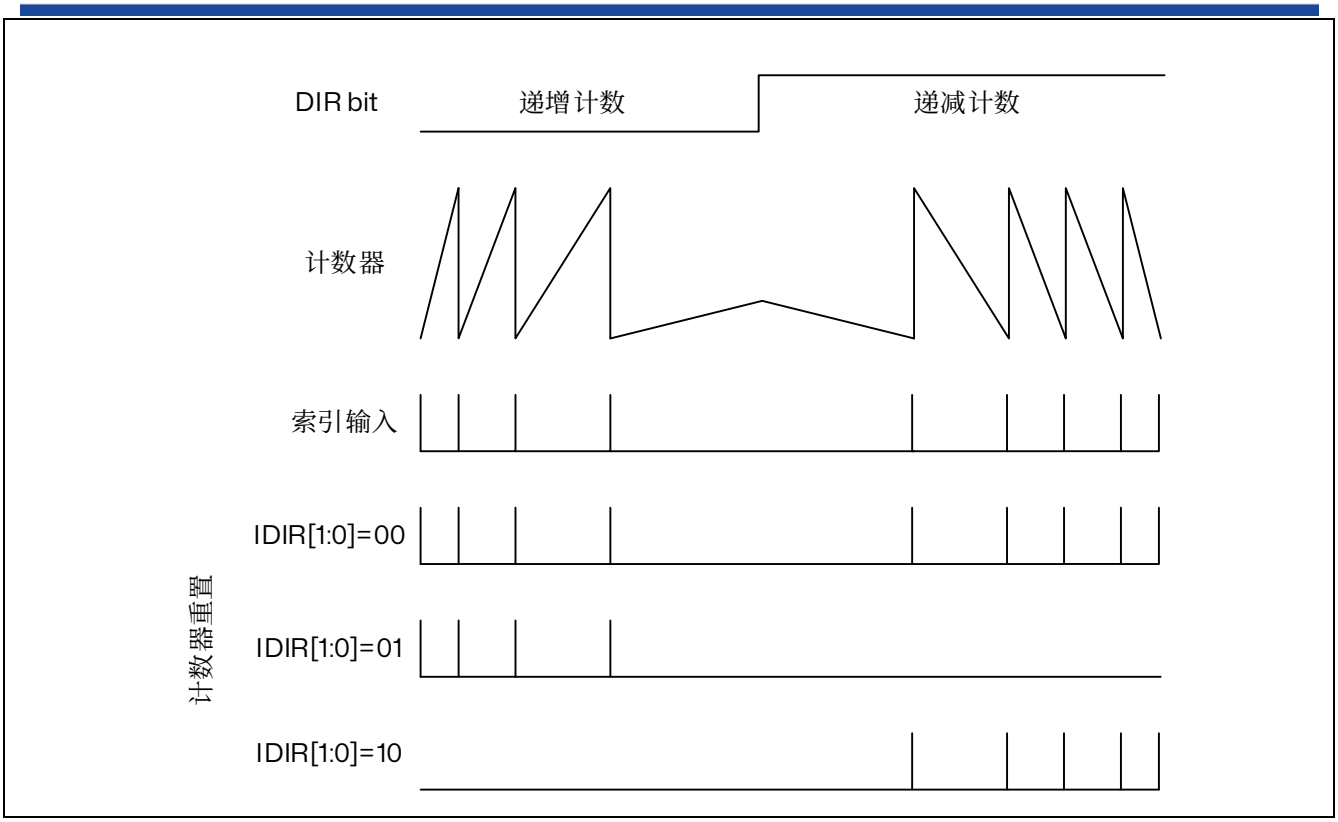


图 21.41 定向索引灵敏度

特殊的第一索引事件管理

TIMx_ECR 寄存器中的 FIDX 位允许仅捕获一次索引，如下图所示。一旦第一个索引到达，任何后续索引都会被忽略。如果需要，可以通过将 FIDX 位写入 0 并再次将其设置为 1 来重新启动电路。

注：当 FIDX = 1 时，如果方向在位置 0 处改变（索引激活），索引可以发布两次（IDXF 标志设置）。

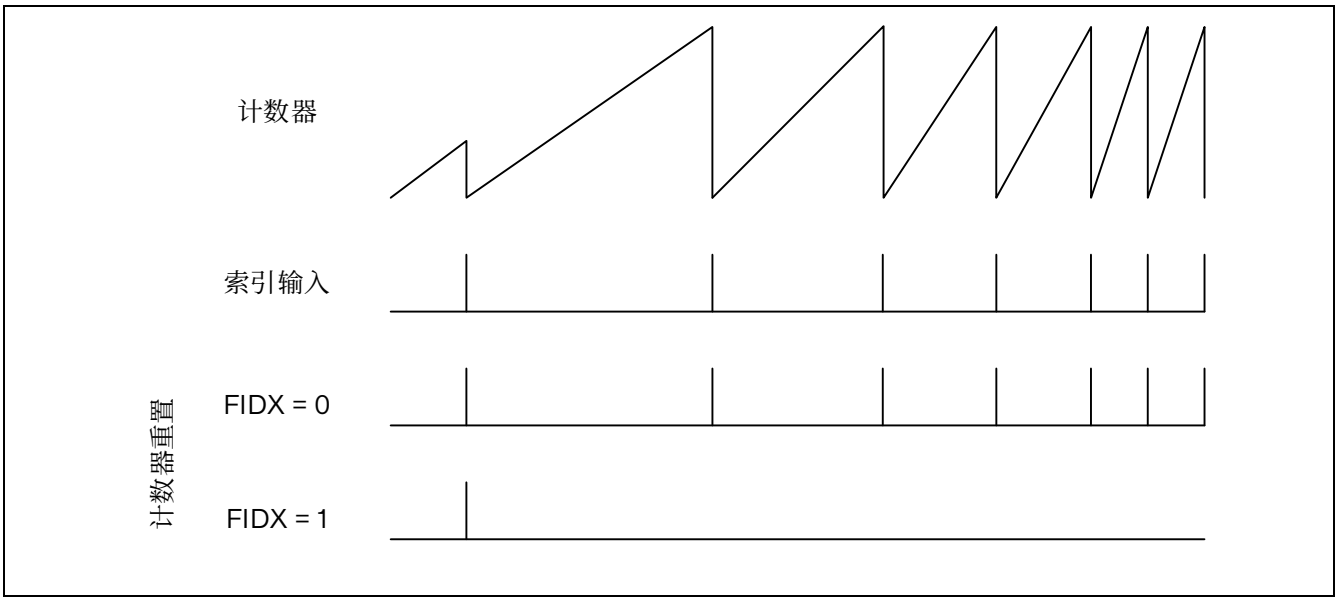


图 21.42 作为 FIDX 位设置的功能的计数器重置

非正交模式下的索引管理

下面两张图详细说明了当 SMS[3:0]位域等于 1010、1011、1100、1101 时，如何在方向时钟模式和时钟加方向模式下管理索引。

对于这两种模式，索引灵敏度通过 IPOS[0]位进行设置，具体如下：

- IPOS[0] = 0：在时钟低电平上检测索引
- IPOS[0] = 1：在时钟高电平上检测索引

IPOS[1]位不重要。

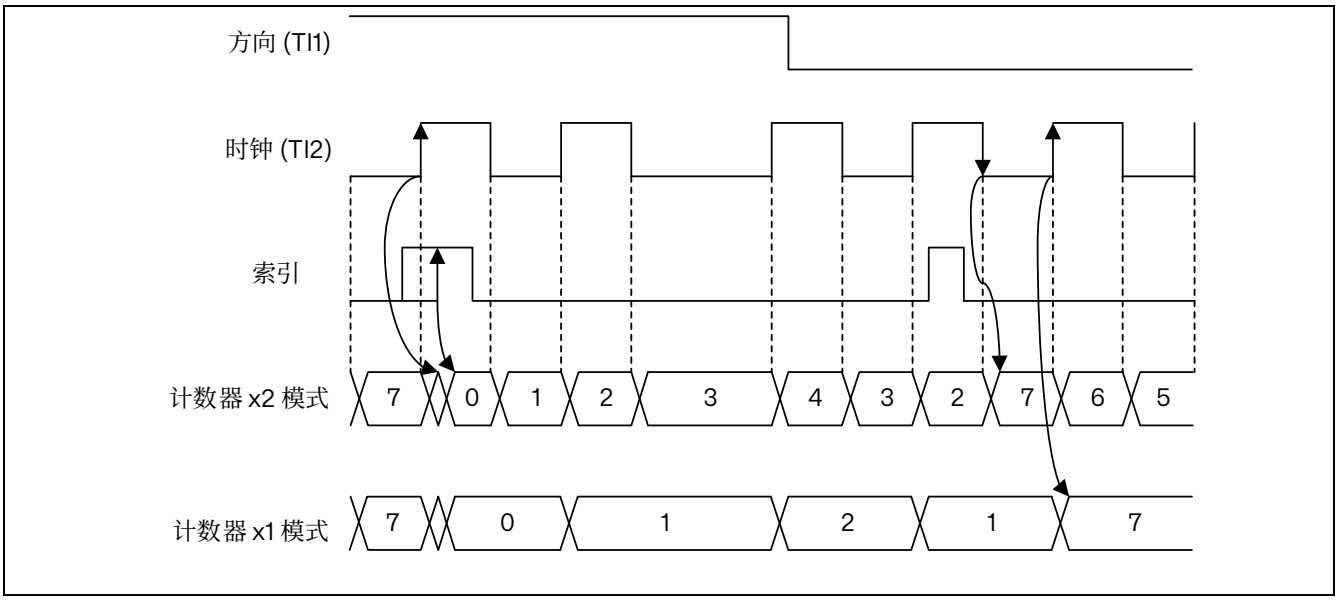


图 21.43 时钟+方向模式下的索引行为，IPOS[0] = 1

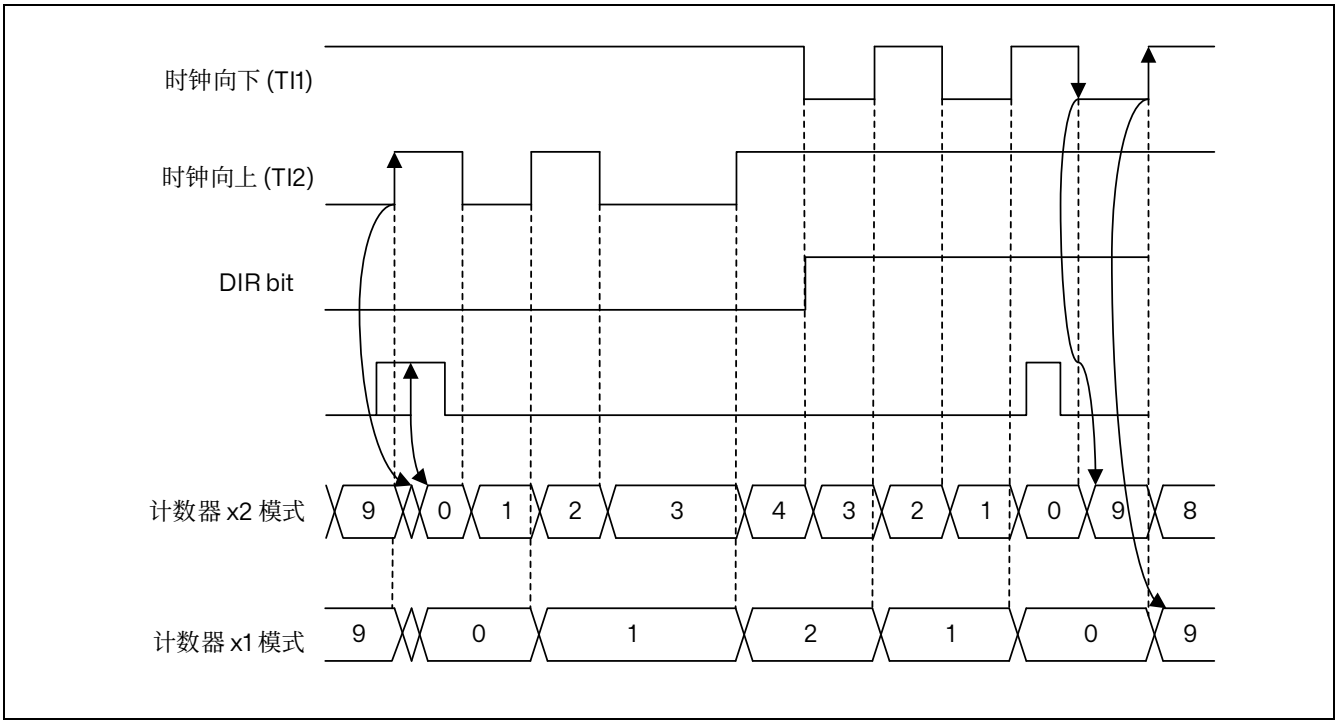


图 21.44 方向时钟模式下的索引行为，IPOS[0] = 1

编码器错误管理

对于具有 2 个正交信号的编码器配置，可以检测到转换错误。2 个输入的读数对应于一个 2 位灰度码，该灰度码可以表示为状态图，如以下图所示。期望单个比特立即更改。错误的转换会在 TIMx_SR 状态寄存器中设置 TERRF 中断标志。如果 TIMx_DIER 寄存器中的 TERRIE 位被设置，将产生转换错误中断。

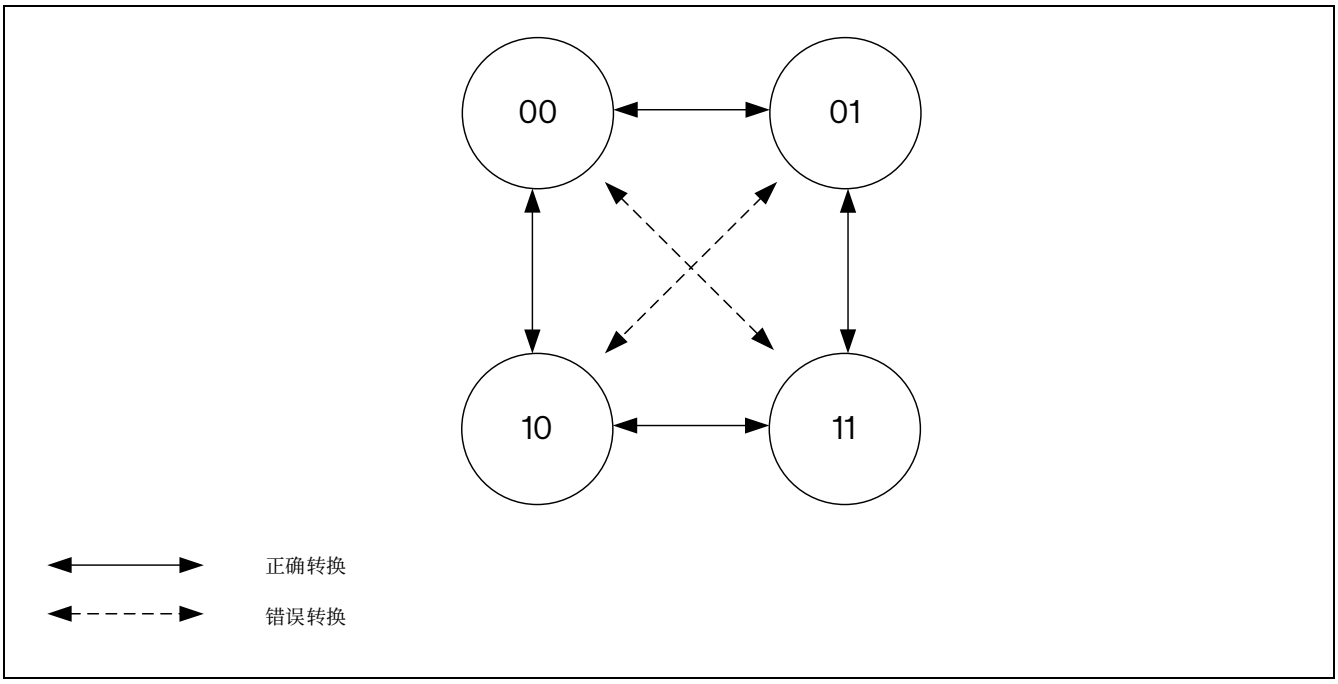


图 21.45 正交编码信号的状态图

对于具有索引信号的编码器，可以检测到导致每转脉冲过多的异常操作。一个编码器每转提供 N 个脉冲，则每转有 4xN 个计数。索引信号每 4xN 个时钟周期重置一次计数器。

如果计数器的值从 TIMx_ARR 增加到 0 或从 0 减少到 TIMxARR 的值而没有索引事件，则报告为索引位置错误。

可使用 TIMx_ARR 寄存器来编程溢出阈值。一个 1000 线的编码器导致计数器的值在 0 和 3999 之间（在 4 倍读取模式下）。必须通过设置 TIMx_ARR = 3999 + 1 = 4000 来编程溢出检测阈值。

错误断言被延迟到在递增计数时从 0 到 1 的转换。这是为了处理门控 A 和 B 模式中的窄索引脉冲，如下图所示。

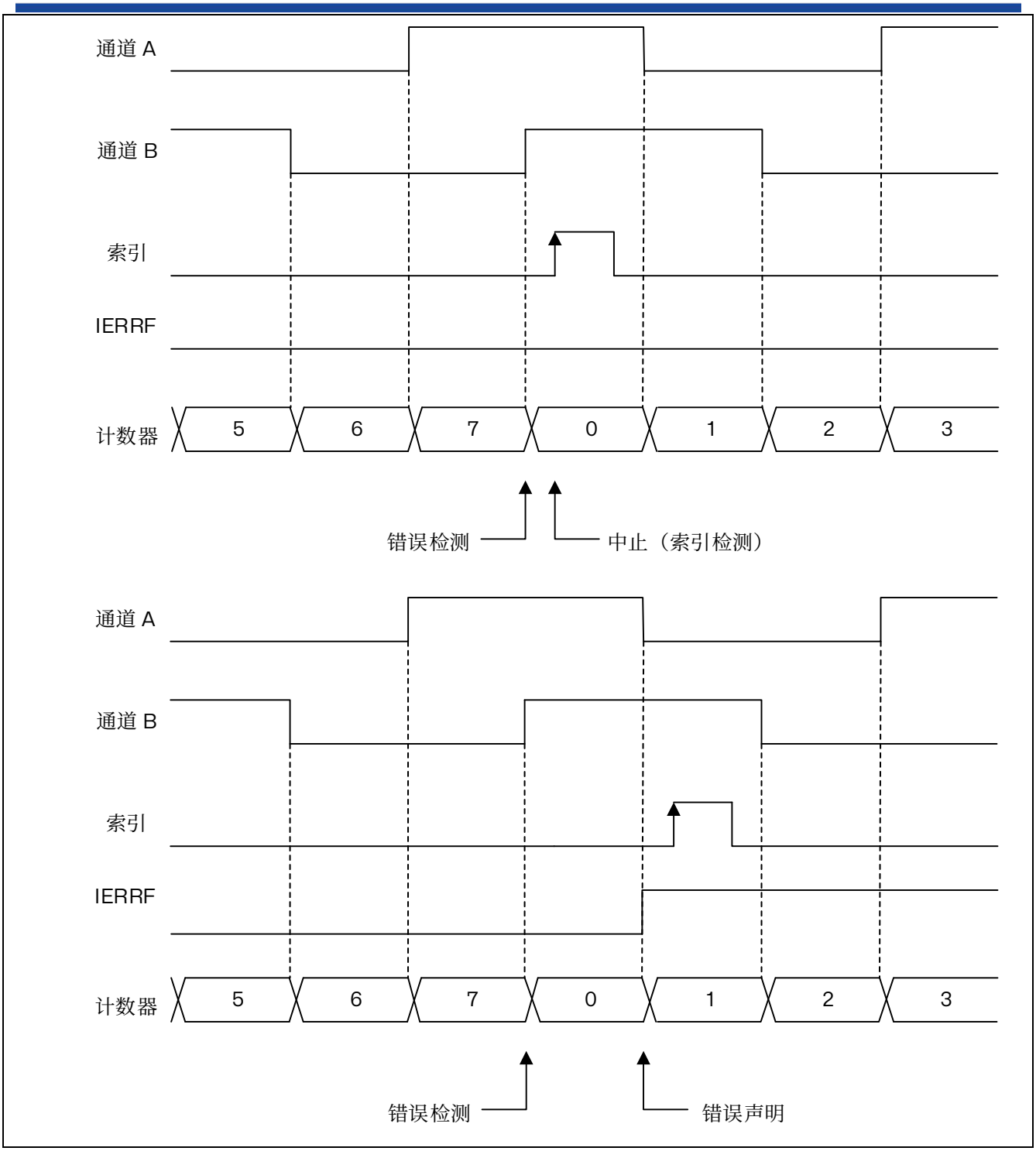


图 21.46 递增计数编码器错误检测

在递减计数模式下，检测是由从 1 到 0 的初步转换引起的。这是为了处理门控 A 和 B 模式中的窄索引脉冲，如下图所示，以避免编码器在索引检测后立即在 TIMx_ARR 和 0 之间抖动时出现任何错误检测。

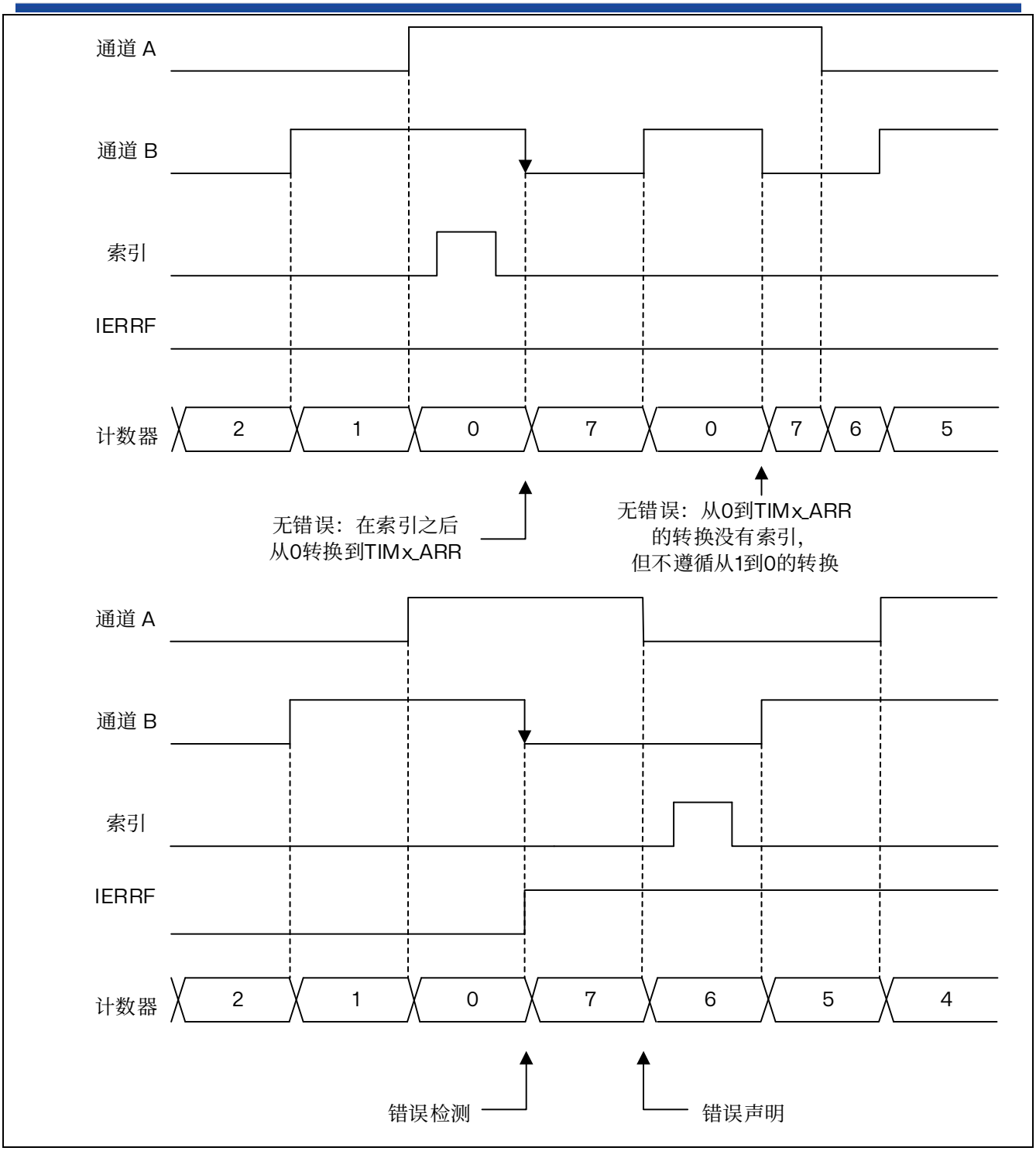


图 21.47 递减计数编码器错误检测

索引错误会在 TIMx_SR 状态寄存器中设置 IERRF 中断标志。如果 TIMx_DIER 寄存器中的 IERRIE 位被设置，将产生索引错误中断。

功能编码器中断

编码器模式下也可使用以下中断：

- 方向改变：在编码器模式下，任何计数方向的改变都会导致 TIMx_CR1 寄存器中的 DIR 位发生翻转。方向改变会设置 TIMx_SR 状态寄存器中的 DIRF 中断标志。如果 DIRIE 位在 TIMx_DIER 寄存器中被设置，则产生方向改变中断。
- 索引事件：索引事件会设置 TIMx_SR 状态寄存器中的 IDXF 中断标志。如果 IDXIE 位在 TIMx_DIER 寄存器中被设置，则产生索引中断。

运行时更新编码器模式时从机模式选择预加载

在运行过程中，可能需要从一个编码器模式切换到另一个编码器模式。这通常在高速度下完成，以降低更新中断频率，从 x4 模式切换到 x2 模式再到 x1 模式，如下图所示。

为此，可以预加载 SMS[3:0]位。通过设置 TIMx_SMCR 寄存器中的 SMSPE 使能位可以使能此功能。从 SMS[3:0]预加载值转换到有效值可以由 TIMx_SMCR 寄存器中的 SMSPS 位选择触发。

- SMSPS = 0：当计数器递增时溢出并在计数器递减时下溢时，由更新事件(UEV)触发的转换。只有在索引被禁用时（位 IE = 0）才可以使用此模式。
- SMSPS = 1：由索引事件触发的转换。

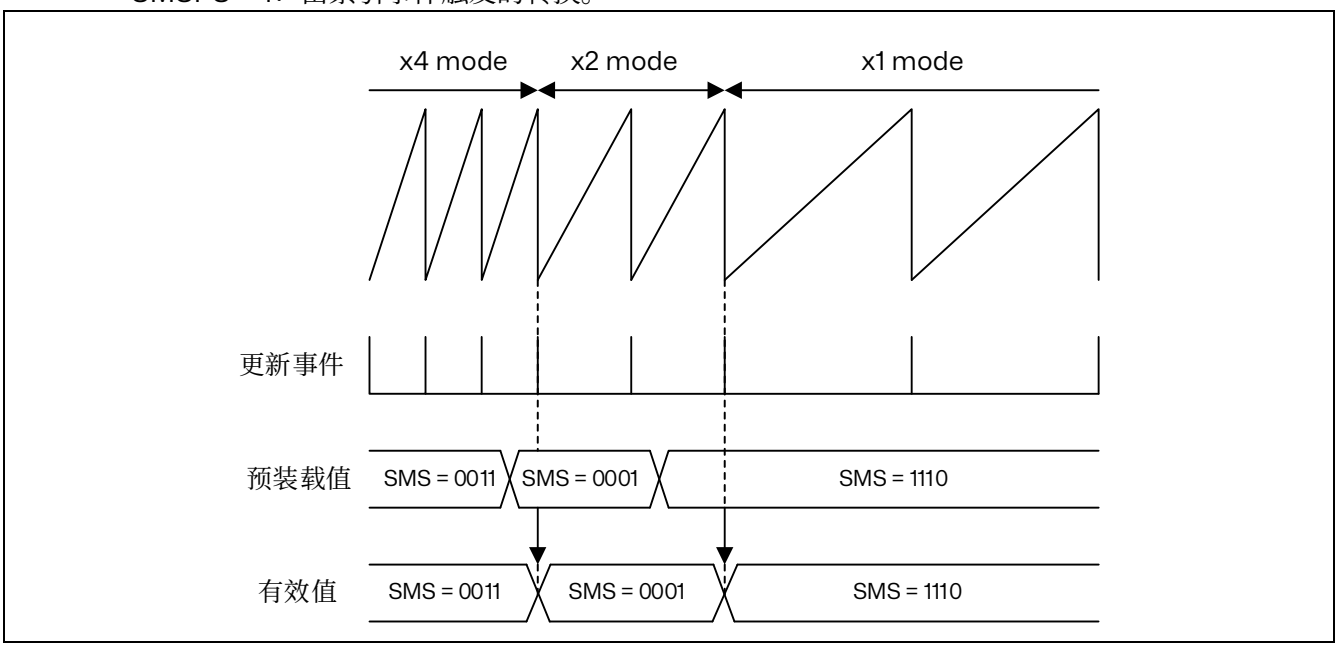


图 21.48 更新时带有预加载转移的编码器模式更改（SMSPS = 0）

编码器时钟输出

编码器模式的操作原理并不完全适合于高分辨率速度测量，在低速时，因为它需要相对较长的积分时间才能获得足够数量的时钟边缘和精确的测量。

在低速时，更好的解决方案是进行边到边的时钟周期测量。这可以通过使用从属定时器来实现。定时器可以在 tim_trgo 输出上输出编码器时钟信息。然后，从属定时器可以进行周期测量并为每个编码器时钟边缘提供速度信息。

通过将 TIMx_CR2 寄存器中的 MMS[3:0]位域设置为 1000 可以使能此模式。对于以下 SMS[3:0]值有效：0001、0010、0011、1010、1011、1100、1101、1110、1111。任何可能导致意外行为的其他 SMS[3:0]代码是不被允许的。

21.3.10 UIF 位重映射

TIMx_CR1 寄存器中的 UIFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许以原子方式读取计数器的值和一个潜在的由 UIFCPY 标志指示的翻转条件。在特定情况下，它可以避免由背景任务（读取计数器）和中断（更新中断）之间的处理引起的竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间没有延迟。

21.3.11 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，可将通道 1 的输入滤波器连接到异或门的输出，从而将 tim_ti1，tim_ti2，tim_ti3 这三个输入引脚组合在一起。

异或输出可与触发或输入捕获等所有定时器输入功能配合使用。

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。

21.3.12 与霍尔传感器接口

这是通过使用高级控制定时器生成 PWM 信号来驱动电机，并使用另一个被称为“接口定时器”的 TIMx 定时器来实现的。“接口定时器”捕获通过 XOR 连接到 tim_ti1 输入通道（通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择）的 3 个定时器输入引脚（tim_ti1、tim_ti2 和 tim_ti3）。

从属模式控制器配置为复位模式；从属输入是 tim_ti1f_ed。因此，每当 3 个输入中的任何一个切换时，计数器都会从 0 开始重新计数。这创建了一个由霍尔输入上的任何变化触发的时间基准。

在“接口定时器”上，捕获/比较通道 1 配置为捕获模式，捕获信号是 tim_trc。捕获的值对应于输入上两次变化之间的时间间隔，提供了有关电机速度的信息。

21.3.13 定时器和外部触发的同步

TIMx 定时器以下列模式与外部触发实现同步：复位模式、门控模式、触发模式、复位+触发和门控+复位模式。

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 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIMx_ARR，TIMx_CCRx）都被更新了。

在以下的例子中，tim_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 tim_ti1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性（只检测上升沿）。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 tim_ti1 出现一个上升沿；当 tim_ti1 上升时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIMx_SR 寄存器中的 TIF 位）被设置，中断请求或 DMA 请求可以被发送（取决于 TIMx_DIER 寄存器中的 TIE 和 TDE 位）。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。

在 tim_ti1 上升沿和计数器的实际复位之间的延时取决于 tim_ti1 输入端的重同步电路。

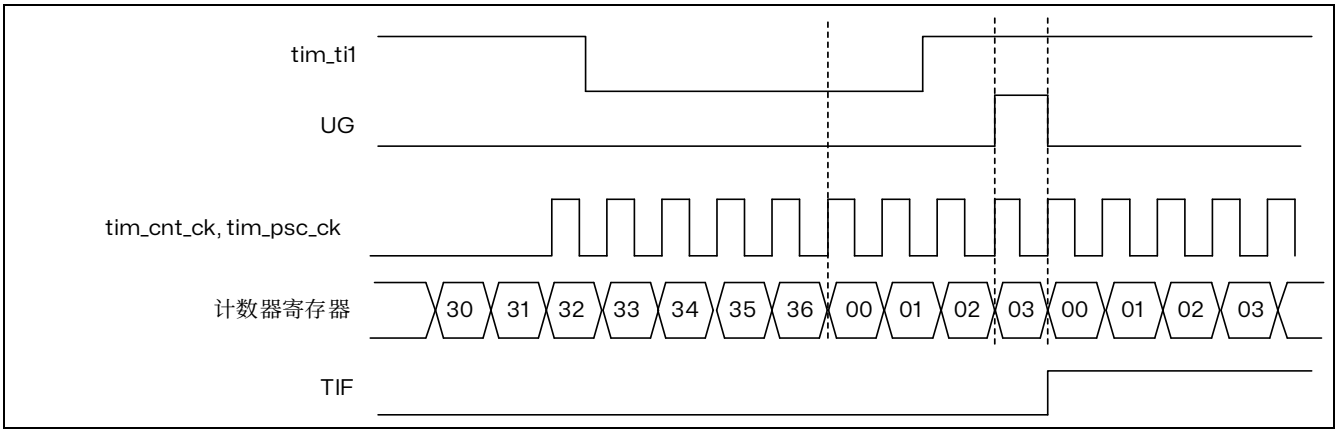


图 21.49 复位模式下的控制电路

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 tim_ti1 为低时向上计数：

- 配置通道 1 以检测 tim_ti1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=0101，选择 tim_ti1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）。

只要 tim_ti1 为低，计数器开始依据内部时钟计数，一旦 tim_ti1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

tim_ti1 上升沿和计数器实际停止之间的延时取决于 tim_ti1 输入端的重同步电路。

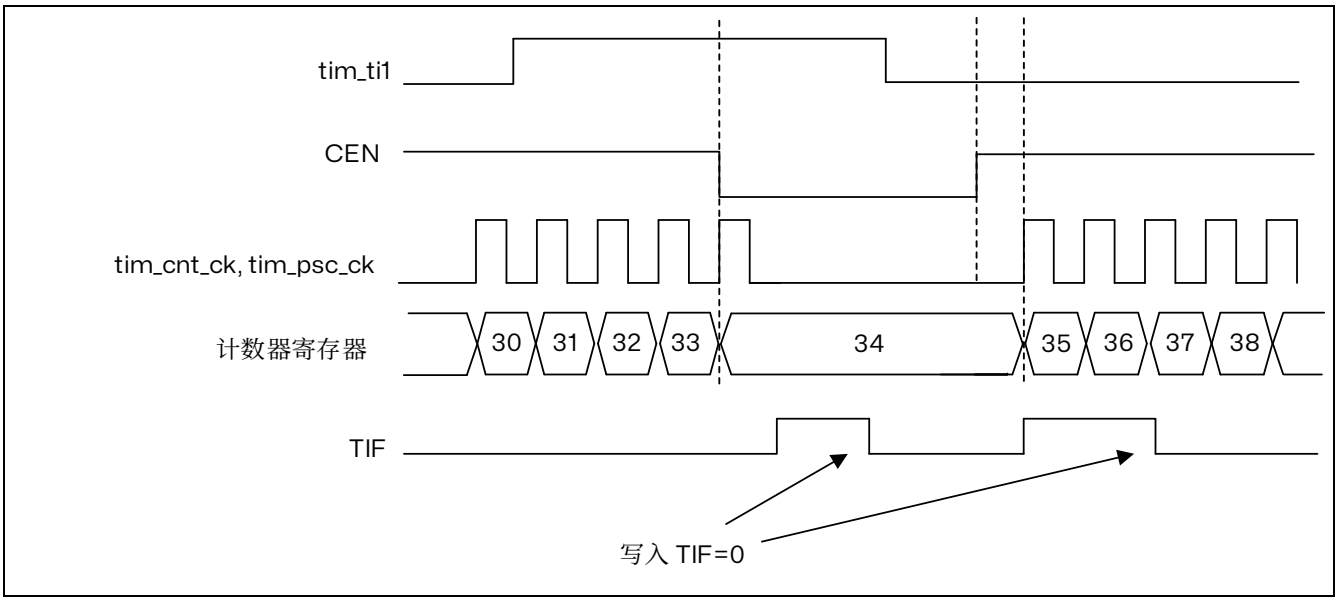


图 21.50 门控模式下的控制电路

注：由于门控模式作用于电平而非边沿，因此在门控模式下，“CCxP=CCxNP=1”（同时检测上升沿和下降沿）不发挥任何作用。

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 tim_ti2 输入的上升沿开始向上计数：

- 配置通道 2 检测 tim_ti2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 和 CC2NP=0 以确定极性（只检测低电平）。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=0110，选择 tim_ti2 作为输入源。

当 tim_ti2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

tim_ti2 上升沿和计数器启动计数之间的延时，取决于 tim_ti2 输入端的重同步电路。

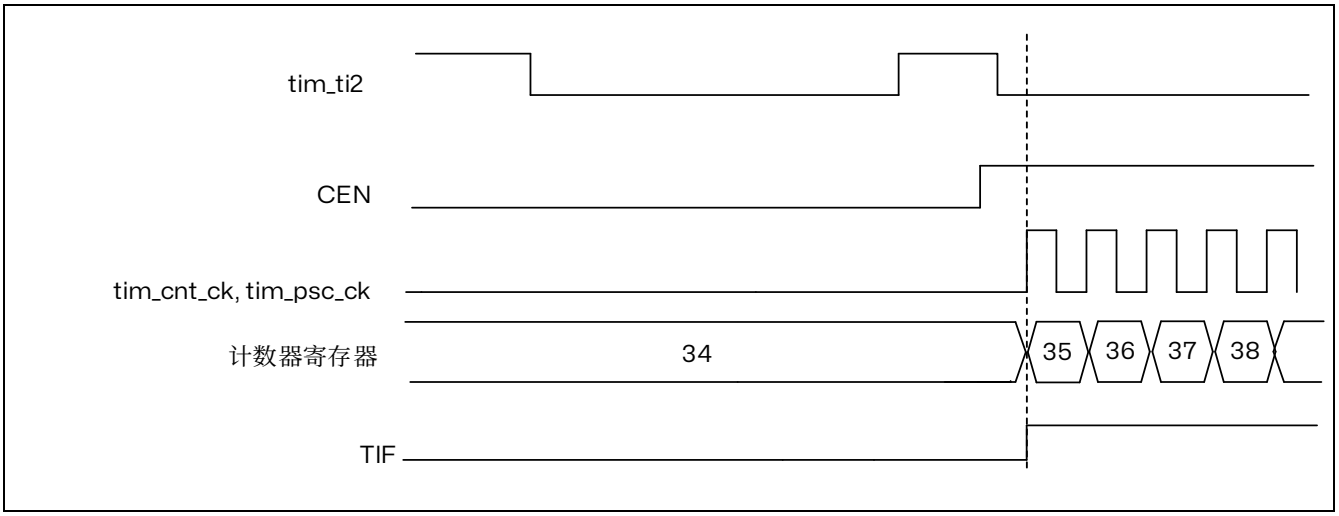


图 21.51 触发器模式下的控制电路

对实时更新编码器模式的从模式选择预装载

SMS[3:0]位可以预加载。这是通过在 TIMx_SMCR 寄存器中设置 SMSPE 使能位来实现的。从 SMS[3:0]预加载到有效值的触发器是计数器溢出时发生的更新事件（UEV）。

从模式：组合复位+触发模式

在这种情况下，选择的触发输入（tim_trgi）的上升沿重新初始化计数器，生成寄存器的更新，并启动计数器。

该模式用于单脉冲模式。

从模式：组合门控+复位模式

触发输入（tim_trgi）高时使能计数器。一旦触发变低，计数器停止并复位。计数器的开始和停止都是受控的。

该模式允许检测超出范围的 PWM 信号（占空比超过最大期望值）。

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，tim_etr_in 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（复位模式、门控模式或触发模式）。

在下面的例子中，一旦在 tim_ti1 上出现一个上升沿，计数器即在 tim_etr_in 的每一个上升沿向上计数一次：

1. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：
   - ETF = 0000：没有滤波
   - ETP = 0：检测 tim_etr_in 的上升沿，置 ECE=1 使能外部时钟模式 2。
2. 按如下配置通道 1，检测 TI 的上升沿：
   - IC1F = 0000：没有滤波。
   - 置 TIMx_CCMR1 寄存器中 CC1S = 01，选择输入捕获源
   - 置 TIMx_CCER 寄存器中 CC1P = 0 和 CC1NP = 0 以确定极性（只检测上升沿）
3. 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=00101，选择 tim_ti1 作为输入源。

当 tim_ti1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 tim_etr_in 的上升沿计数。

tim_etr_in 信号的上升沿和计数器实际复位间的延时，取决于 tim_etrp 输入端的重同步电路。

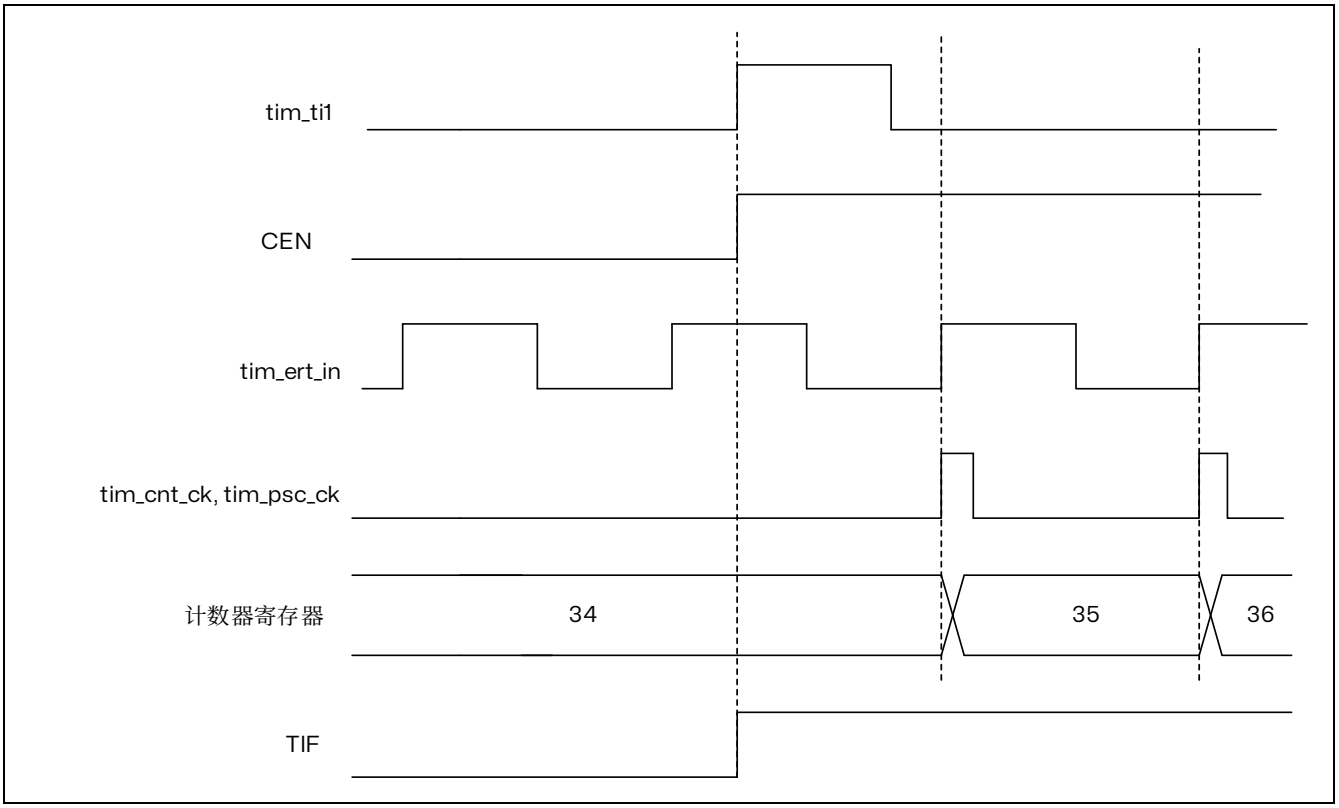


图 21.52 外部时钟模式 2+触发模式下的控制电路

21.3.14 定时器同步

TIMx 定时器从内部连接在一起，以实现定时器同步或级联。当某个定时器配置为主模式时，可对另一个配置为从模式的定时器的计数器执行复位、启动、停止操作或为其提供时钟。下图显示了触发选择和主模式选择模块的概况。

下图简要介绍了触发选择和主模式选择框图。

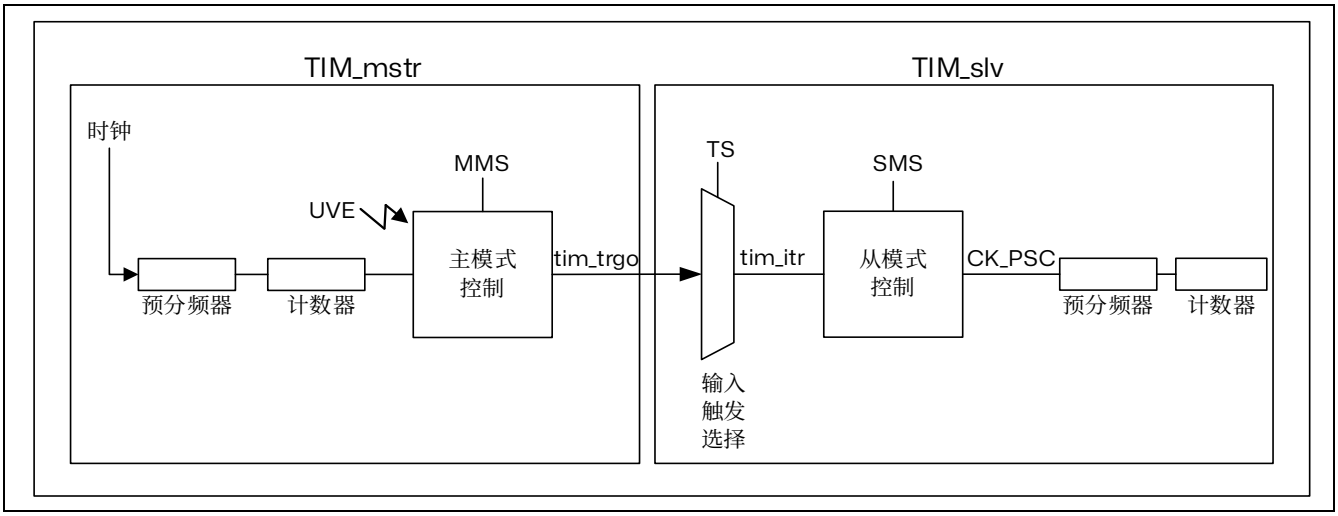


图 21.53 主/从定时器的例子

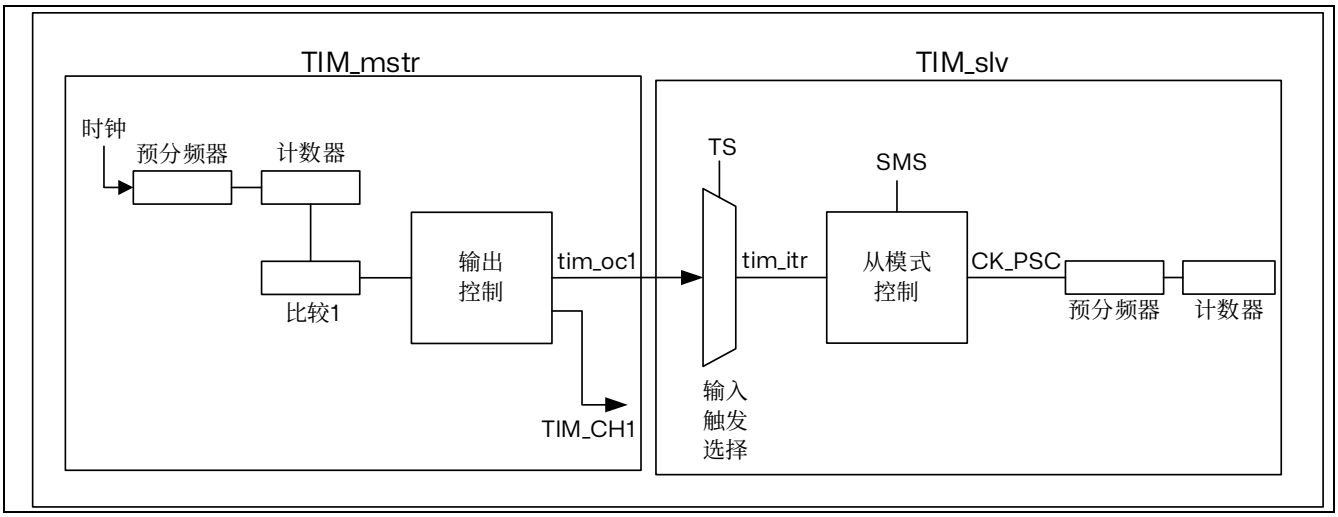


图 21.54 仅定时器通道 1 主/从连接示例

*注：只有一个通道的计时器（参见上图）不具有主模式。然而，*tim_oc1 *输出信号可以作为从属定时器的触发信号（参见*TIMx *内部触发连接表：*TIMA/TIMB *引脚和内部信号）。*
*必须将*tim_oc1 *信号脉冲宽度编程为至少为目的地定时器的*2 *个时钟周期，以确保从属定时器检测到触发信号。例如，如果目的地定时器*tim_ker_ck *时钟比源定时器慢*4 *倍，则*OC1 *脉冲宽度必须是*8 *个时钟周期。*

使用一个定时器作为另一个定时器的预分频器

例如，可以将 TIM_mstr 配置为 TIM_slv 的预分频器。请参见上图。为此：

1. 将 TIM_mstr 配置为主模式，以便每次发生更新事件 UEV 时都输出一个周期性触发信号。如果

在 TIM_mstr_CR2 寄存器中写入 MMS=010，则每次生成更新事件时，tim_trgo 都会输出一个上升沿。

2. 要将 TIM_mstr 的 tim_trgo 输出连接到 TIM_slv，必须将 TIM_slv 配置为从模式，使用 ITR0 作为内部触发。通过 TIM_slv_SMCR 寄存器中的 TS 位（写入 TS=000）可对此进行选择。
3. 然后将从模式控制器设为外部时钟模式 1（在 TIM_slv_SMCR 寄存器中写入 SMS=111）。这样一来，TIM_slv 的时钟将由 TIM_mstr 周期性触发信号的上升沿（与 TIM_mstr 的计数器上溢对应）提供。
4. 最后必须通过将这两个定时器的相应 CEN 位（TIMx_CR1 寄存器）置 1 同时使能二者。

注：如果选择 TIM_mstr 的 tim_ocx 信号作为触发输出（MMS=1xx），该信号的上升沿将用于驱动 TIM_slv 的计数器。

使用一个定时器使能另一个定时器

本例中通过 TIM_mstr 的输出比较 1 来使能 TIM_slv。相关连接图，参考下图。仅当 TIM_mstr 的 tim_oc1ref 为高电平时，TIM_slv 才根据分频后的内部时钟进行计数。两个计数器的时钟频率都基于 tim_ker_ck 通过预分频器执行 3 分频（$f_{tim_cnt_ck} = f_{tim_ker_ck}/3$）

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号（tim_oc1ref）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref 波形（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
6. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

注：计数器 2 的时钟与计数器 1 不同步，此模式仅影响 TIM_slv 的计数器使能信号。

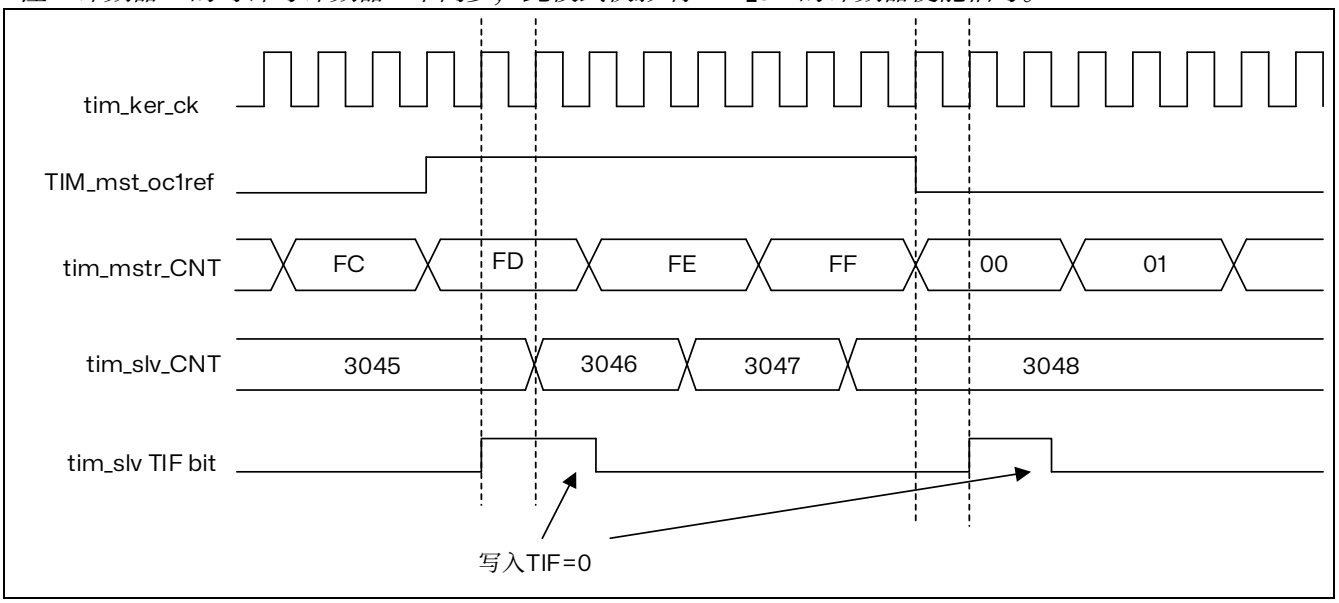


图 21.55 TIM_mstr 的 tim_oc1ref 控制 TIM_slv

在上图示例中，TIM_slv 的计数器和预分频器在启动前未进行初始化。因此从各自的当前值开始计数。启动 TIM_mstr 之前，通过复位这两个定时器可以从指定值开始计数。这样便可以在定时器计数器中写入所需的任意值。两个定时器都可通过软件使用 TIMx_EGR 寄存器中的 UG 位轻松复位。

在下一示例中，TIM_mstr 与 TIM_slv 同步。TIM_mstr 为主模式，从 0 开始计数。TIM_slv 为从模

式，从 0xE7 开始计数。两个定时器的预分频比相同。在 TIM_mstr_CR1 寄存器中通过向 CEN 位写入“0”来禁止 TIM_mstr 时，TIM_slv 将停止：

1. 将 TIM_mstr 配置为主模式，发送其输出比较 1 参考信号（tim_oc1ref）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=100）。
2. 配置 TIM_mstr 的 tim_oc1ref（TIM_mstr_CCMR1 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。
4. 将 TIM_slv 配置为门控模式（TIM_slv_SMCR 寄存器中的 SMS=101）。
5. 通过向 UG 位（TIM_mstr_EGR 寄存器）写入“1”复位 TIM_mstr。
6. 通过向 UG 位（TIM_slv_EGR 寄存器）写入“1”复位 TIM_slv。
7. 通过在 TIM_slv 的计数器（TIM_slv_CNT）中写入“0xE7”使 TIM_slv 初始化为 0xE7。
8. 通过向 CEN 位（TIM_slv_CR1 寄存器）写入“1”使能 TIM_slv。
9. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。
10. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“0”停止 TIM_mstr。

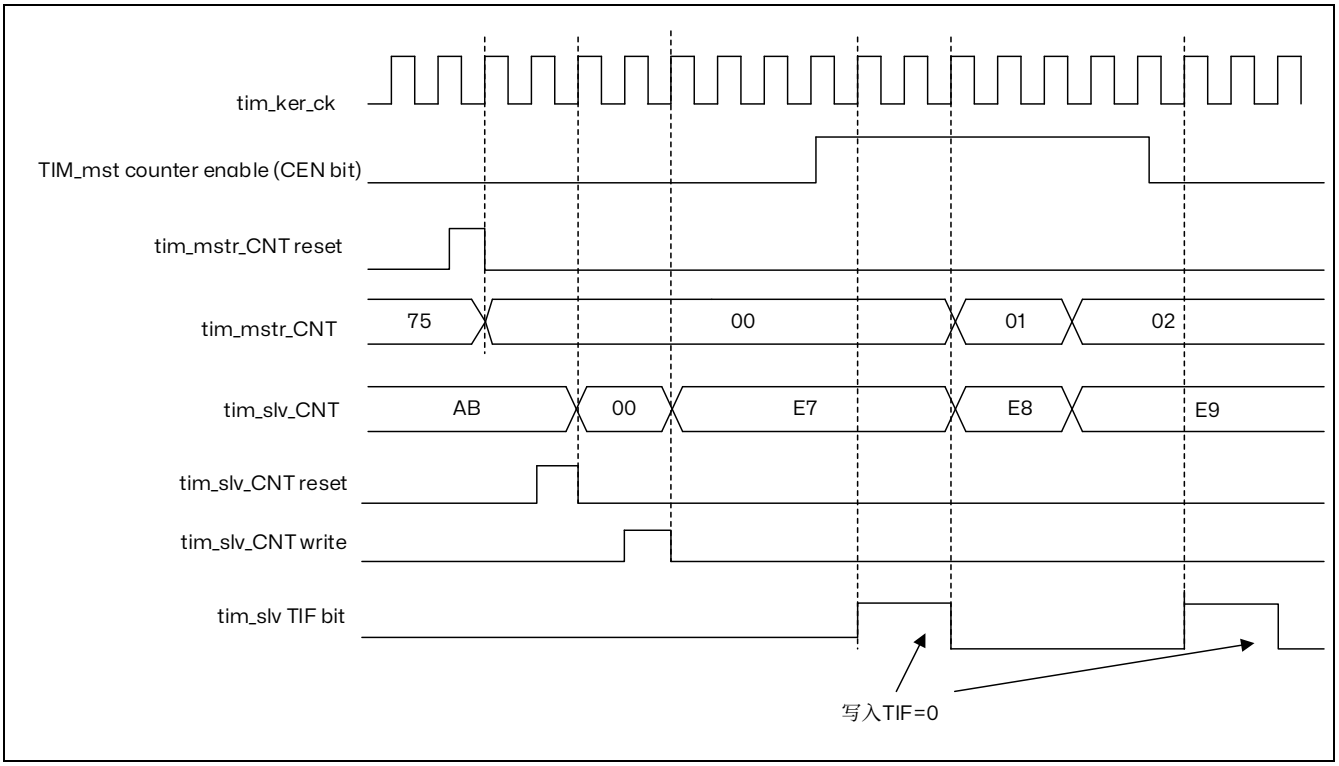


图 21.56 通过使能 TIM_mstr 可以控制 TIM_slv

使用一个定时器去启动另一个定时器

本例中使用 TIM_mstr 的更新事件使能 TIM_slv。相关连接图，请参见下图。只要 TIM_mstr 生成更新事件，TIM_slv 便根据分频后的内部时钟从当前值（可以不为 0）开始计数。TIM_slv 收到触发信号时，其 CEN 位自动置 1，并且计数器开始计数，直到向 TIM_slv_CR1 寄存器的 CEN 位写入“0”后停止计数。两个计数器的时钟频率都基于 CK_INT 通过预分频器执行 3 分频（$f_{tim_cnt_ck} = f_{tim_ker_ck}/3$）。

1. 将 TIM_mstr 配置为主模式，发送其更新事件（UEV）作为触发输出（TIM_mstr_CR2 寄存器中的 MMS=010）。
2. 配置 TIM_mstr 的周期（TIM_mstr_ARR 寄存器）。
3. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0010）。

4. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。
5. 通过向 CEN 位（TIM_mstr_CR1 寄存器）写入“1”启动 TIM_mstr。

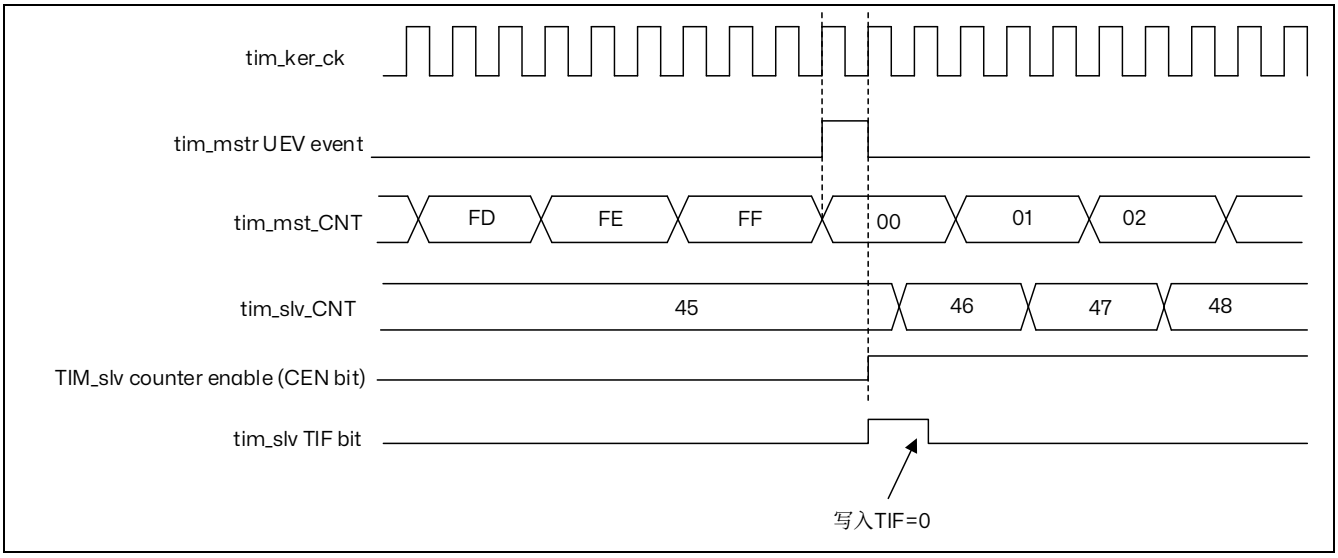


图 21.57 使用 TIM_mstr 的更新触发 TIM_slv

如上述示例所示，用户可以在开始计数之前初始化两个计数器。下图显示了与上图具有相同配置，只不过处于触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）而非门控模式的计数行为。

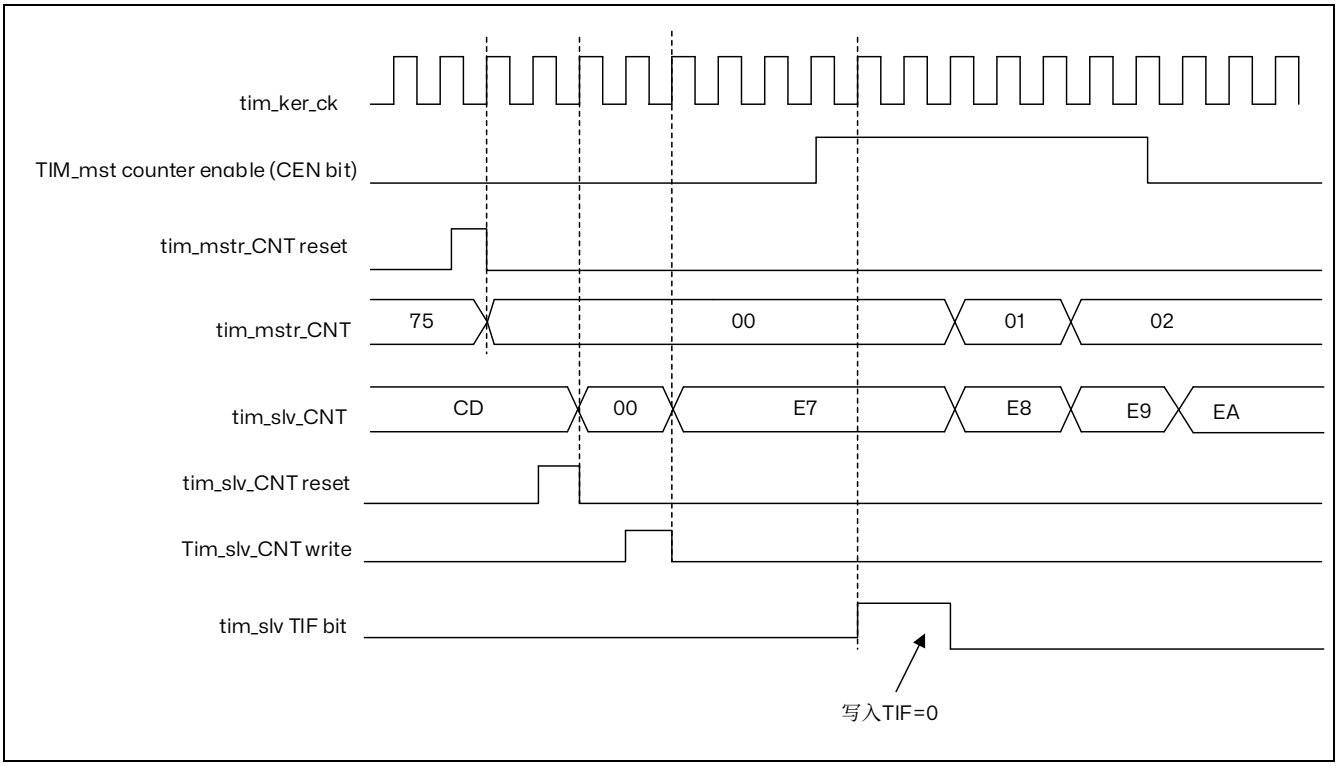


图 21.58 利用 TIM_mstr 的使能触发 TIM_slv

使用一个外部触发同步地启动 2 个定时器

本例中，TIM_mstr 的 tim_ti1 输入出现上升沿时使能 TIM_mstr，使能 TIM_mstr 的同时使能 TIM_slv。要确保两个计数器对齐，TIM_mstr 必须配置为主/从模式（对应的 tim_ti1 为从，对应 TIM_slv 为主）：

1. 将 TIM_mstr 配置为主模式，发送其使能信号作为触发输出（TIM_mstr_CR2 寄存器中 MMS=001）。
2. 将 TIM_mstr 配置为从模式以接收来自 tim_ti1 的输入触发（TIM_mstr_SMCR 寄存器中的 TS=0100）。
3. 将 TIM_mstr 配置为触发模式（TIM_mstrSMCR 寄存器中的 SMS=110）。
4. 通过写入 MSM=1（TIM_mstr_SMCR 寄存器）将 TIM_mstr 配置为主/从模式。
5. 配置 TIM_slv 以接收来自 TIM_mstr 的输入触发（TIM_slv_SMCR 寄存器中的 TS=0000）。
6. 将 TIM_slv 配置为触发模式（TIM_slv_SMCR 寄存器中的 SMS=110）。

当 tim_ti1（TIM_mstr）出现上升沿时，两个计数器开始根据内部时钟同步计数，并且两个 TIF 标志都置 1。

注：本例中，两个定时器都在启动之前进行了初始化（通过将各自的UG 位置1）。两个计数器都从 0 开始计数，但可以通过对任意一个计数器寄存器（TIMx_CNT）进行写操作，在二者之间轻松插入一个偏移量。可注意到主/从模式在TIM_mstr 的CNT_EN 与CK_PSC 之间产生了延迟。

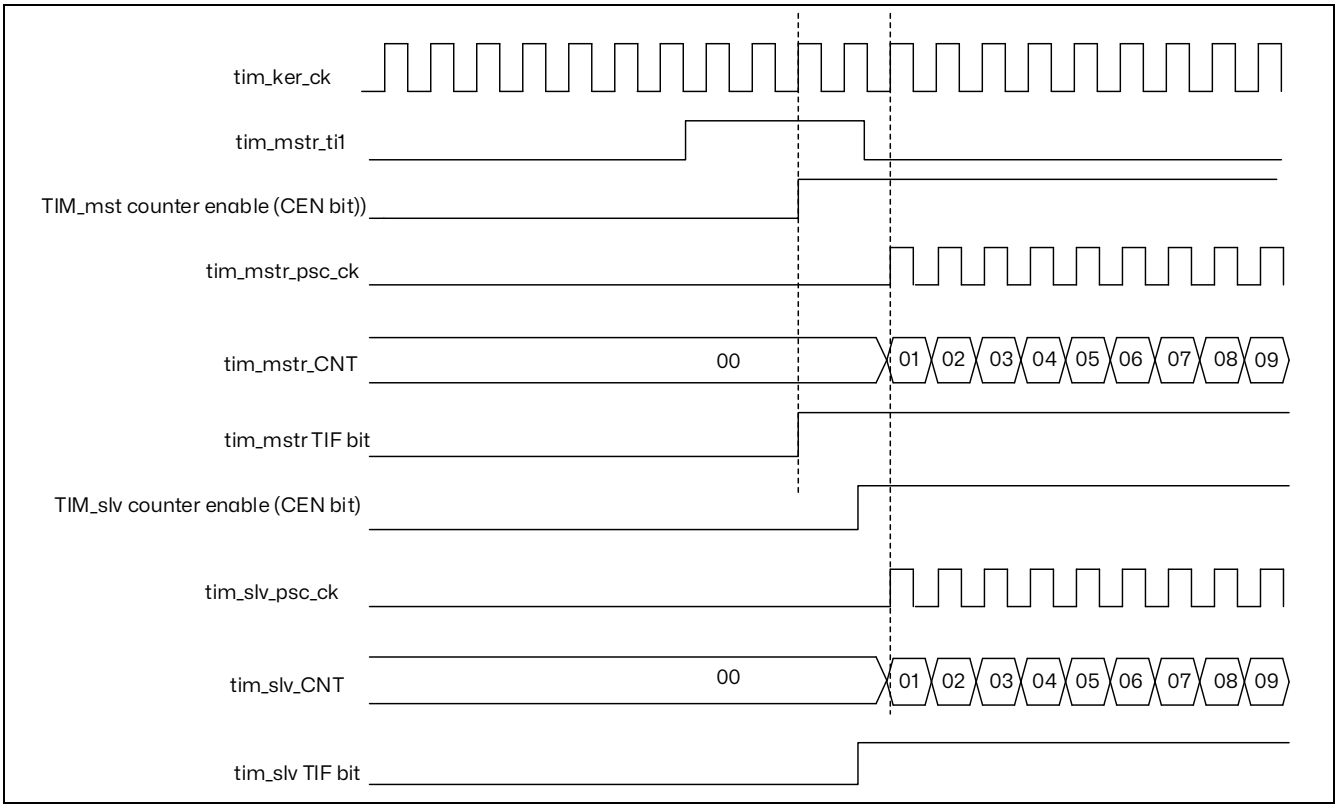


图 21.59 使用 TIM_mstr 的 tim_ti1 输入触发 TIM_mstr 和 TIM_slv

注：接收tim_trgo 信号的从定时器的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能实时更改。

21.3.15 调试模式

当微控制器进入调试模式（内核停止）时，TIMx 计数器可以继续正常工作或停止。
在调试模式下，每个定时器的行为可以通过 Debug 支持（DBG）模块的专用配置位进行编程。
有关更多详细信息，请参阅调试部分。

21.4 TIMA/TIMB 低功耗模式

表 21.10 低功耗模式对 TIMA/TIMB 的影响

模式	描述
睡眠	无影响，外设是运行的。中断会导致设备退出睡眠模式。
停机	定时器操作被停止并保留其寄存器内容。无法生成中断。
待机	定时器被断电并且退出待机模式后必须重新初始化。

21.5 TIMA/TIMB 中断

TIMA/TIMB 可以生成多个中断，如下表所示。

表 21.11 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	从睡眠模式退出	从停机和待机退出
TIM_UP	更新	UIF	UIE	UIF 写 0	是	否
TIM_CC	捕获/比较 1	CC1IF	CC1IE	CC1IF 写 0	是	否
	捕获/比较 2	CC2IF	CC2IE	CC2IF 写 0	是	否
	捕获/比较 3	CC3IF	CC3IE	CC3IF 写 0	是	否
TIM_TRG	触发	TIF	TIE	TIF 写 0	是	否
TIM_DIR_IDX	索引	IDXF	IDXIE	IDXF 写 0	是	否
	方向	DIRF	DIRIE	DIRF 写 0	是	否
TIM_IERR	索引错误	IERRF	IERRIE	IERRF 写 0	是	否
TIM_TER	转换错误	TERRF	TERRIE	TERRF 写 0	是	否

21.6 TIMA/TIMB 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
外设寄存器可以通过半字（16 位）或字（32 位）进行访问。

21.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = A，B）

偏移地址：0x000
复位值：0x0000

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

Bit 15:12 保留，必须保持为复位值。

Bit 11 **UIFREMAP**：UIF 状态位重映射（UIF status bit remapping）

0：无重映射。UIF 状态位未复制到 TIMx_CNT 寄存器位 31

1：重映射使能。UIF 状态位复制到 TIMx_CNT 寄存器位 31

Bit 10 保留，必须保持为复位值。

Bit 9:8 **CKD[1:0]**：时钟分频（Clock division）

此位域指示定时器时钟（tim_ker_ck）频率以及数字滤波器（tim_tix）所使用的采样时钟（t_{DTS}）之间的分频比

00：$t_{DTS} = t_{tim_ker_ck}$

01：$t_{DTS} = 2 \times t_{tim_ker_ck}$

10：$t_{DTS} = 4 \times t_{tim_ker_ck}$

11：保留，不要设置成此值

Bit 7 保留，必须保持为复位值。

Bit 6:5 **CMS[1:0]**：中心对齐模式选择（Center-aligned mode selection）

00：边沿对齐模式。计数器根据方向位（DIR）递增计数或递减计数。

01：中心对齐模式 1。计数器交替进行递增计数和递减计数。仅当计数器递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

10：中心对齐模式 2。计数器交替进行递增计数和递减计数。仅当计数器递增计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志才置 1。

11：中心对齐模式 3。计数器交替进行递增计数和递减计数。当计数器递增计数或递减计数时，配置为输出的通道（TIMx_CCMRx 寄存器中的 CxS=00）的输出比较中断标志都会置 1。

注：只要计数器处于使能状态 (CEN=1)，*就不得从边沿对齐模式切换为中心对齐模式。*

Bit 4 **DIR**：方向（Direction）

0：计数器递增计数；

1：计数器递减计数。

注：当定时器配置为编码器模式时，该位为只读状态。

Bit 3 保留，必须保持为复位值。

Bit 2 **URS**：更新请求源（Update request source）

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断。此类事件包括：

- 计数器上溢/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断。

Bit 1 **UDIS**：更新禁止（Update disable）

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新（UEV）事件可通过以下事件之一生成：

- 计数器上出/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC 和 CCRx）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

Bit 0 **CEN**：计数器使能（Counter enable）

0：禁止计数器；

1：使能计数器。

*注：只有事先通过软件将*CEN *位置*1*，才可以使用外部时钟、门控模式和编码器模式。而触发模式可通过硬件自动将*CEN *位置*1*。*

21.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = A，B）

偏移地址：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						MMS[3]	Reserved								
						rw									

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TI1S	MMS[2:0]			Reserved			
								rw	rw	rw	rw				

Bit 31:26 保留，必须保持为复位值。

Bit 24:8 保留，必须保持为复位值。

Bit 7 **TI1S**：tim_ti1 选择（tim_ti1 selection）

0：tim_ti1_in[15:0]引脚连接到 tim_ti1 输入

1：tim_ti1_in[15:0]，tim_ti2_in[15:0]和 tim_ti3_in[15:0]多路复用器输出被异或并连接至 tim_ti1 输入

Bit 25, 6:4 **MMS[3:0]：**主模式选择（Master mode selection）

这些位可选择主模式下将要发送到从定时器以实现同步的信息(tim_trgo)。这些位的组合如下：

0000：**复位** – TIMx_EGR 寄存器中的 UG 位用作触发输出（tim_trgo)。如果复位由触发输入生成（从模式控制器配置为复位模式），则 TRGO 上的信号相比实际复位会有延迟。

0001：**使能** – 计数器使能信号 CNT_EN 用作触发输出（tim_trgo)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号可由 CEN 控制位产生。当配置为门控模式时，也可由触发输入产生。当计数器使能信号由触发输入控制时，tim_trgo 上会存在延迟，选择主/从模式时除外（请参见 TIMx_SMCR 寄存器中 MSM 位的说明)。

0010：**更新** – 选择更新事件作为触发输出（tim_trgo)。例如，主定时器可用作从定时器的预分频器。

0011：**比较脉冲** – 一旦发生输入捕获或比较匹配事件，当 CC1IF 被置 1 时（即使已为高电平），触发输出都会发送一个正脉冲。(tim_trgo)。

0100：保留

0101：保留

0110：保留

0111：保留

1000：**编码器时钟输出** – 编码器时钟信号用作出发输出（tim_trgo)。此编码对以下 SMS[3:0]值有效：0001，0010，0011，1010，1011，1100，1101，1110，1111。任何其他 SMS[3:0]编码都是不允许的，可能会导致意外行为。

其他编码保留

注：在从主定时器接收事件之前，必须使能从机定时器或 ADC 的时钟，并且在从主定时器接收触发器时不能随意更改。

Bit 3:0 保留，必须保持为复位值。

21.6.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x = A，B）

偏移地址：0x008

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SMSPS	SMSPE	Reserved			TS[3]	Reserved			SMS[3]
						rw	rw				rw				rw

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

Bit 31:26 保留，必须保持为复位值。

Bit 25 **SMSPS**：SMS 预加载源（SMS preload source）

此位选择触发 SMS[3:0]位域从预加载到活动的事件来源：

0：由定时器的更新事件触发

1：由索引事件触发

Bit 24 **SMSPE**：SMS 预加载使能（SMS preload enable）

此位选择是否使能 SMS[3:0]位域的预加载：

0：SMS[3:0]位域不进行预加载

1：使能 SMS[3:0]预加载

Bit 23:21 保留，必须保持为复位值。

Bit 19:17 保留，必须保持为复位值。

Bit 15 **ETP**：外部触发极性（External trigger polarity）

此位可选择将 tim_etr_in 还是 tim_etr_in 的反相用于触发操作

0：tim_etr_in 未反相，高电平或上升沿有效。

1：tim_etr_in 反相，低电平或下降沿有效。

Bit 14 **ECE**：外部时钟使能位（External clock enable）

此位可使能外部时钟模式 2。

0：禁止外部时钟模式 2

1：使能外部时钟模式 2。计数器时钟由 tim_etrf 信号的任意有效边沿提供。

注：将 ECE *位置*1 *与选择外部时钟模式*1 *并将* tim_trgi *连接到* tim_etrf *（*SMS=111 *且* TS=0111*）具有相同效果。*

*外部时钟模式*2 *可以和以下从模式同时使用：复位模式、门控模式和触发模式。不过此类情况下* tim_trgi *不得连接* tim_etrf *（*TS *位不得为*0111*）。*

*如果同时使能外部时钟模式*1 *和外部时钟模式*2*，则外部时钟输入为* tim_etrf。

Bit 13:12 保留，必须保持为复位值。

Bit 11:8 **ETF[3:0]**：外部触发滤波器（External trigger filter）

此位域可定义 tim_etrp 信号的采样频率和适用于 tim_etrp 的数字滤波时间。数字滤波器由事件计数器组成，每 N 个事件才视为一个有效边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{CK_INT}$，N=2。

0010：$f_{SAMPLING}=f_{CK_INT}$，N=4。

0011：$f_{SAMPLING}=f_{CK_INT}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6。

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5。

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6。

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5。

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8。

Bit 7 **MSM：**主/从模式（Master/slave mode）

0：不执行任何操作；

1：当前定时器的触发输入事件(tim_trgi)的动作被推迟，以使当前定时器与其从定时器实现完美同步（通过 tim_trgo）。此设置适用于由单个外部事件对多个定时器进行同步的情况。

Bit 20,6,5,4 **TS[3:0]：**触发选择（Trigger selection）

此位域可选择将要用于同步计数器的触发输入。

0000：内部触发 0（tim_itr0）

0001：内部触发 1（tim_itr1）

0010：内部触发 2（tim_itr2）

0011：内部触发 3（tim_itr3）

0100：tim_ti1 边沿检测器（tim_ti1f_ed）

0101：滤波后的定时器输入 1（tim_ti1fp1）

0110：滤波后的定时器输入 2（tim_ti2fp2）

0111：保留

1000：内部触发器 4 (tim_itr4)

1001：内部触发器 5 (tim_itr5)

其他：保留

有关定时器 ITRx 含义的详细信息，请参见章节：TIMA/TIMB 引脚和内部信号

注：这些位只能在未使用的情况下（例如，SMS=000 时）进行更改，以避免转换时出现错误的边沿检测。

Bit 3 保留，必须保持为复位值。

Bit 16, 2,1,0 **SMS[3:0]**：从模式选择（Slave mode selection）

当选择了外部信号，触发信号（tim_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）

0000：**关闭从模式** – 如果 CEN=1，则预分频器直接由内部时钟驱动。

0001：**编码器模式 1** – 计数器根据 tim_ti1fp1 电平在 tim_ti2fp2 边沿递增/递减计数。

0010：**编码器模式 2** – 计数器根据 tim_ti2fp2 电平在 tim_ti1fp1 边沿递增/递减计数。

0011：**编码器模式 3** – 计数器在 tim_ti1fp1 和 tim_ti2fp2 的边沿计数，计数的方向取决于另外一个信号的电平。

0100：**复位模式** – 选中的触发输入（tim_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。

0101：**门控模式** – 当触发输入（tim_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。

0110：**触发模式** – 计数器在触发输入 tim_trgi 的上升沿启动（但不复位），只有计数器的启动是受控的。

0111：**外部时钟模式 1** – 选中的触发输入（tim_trgi）的上升沿驱动计数器。

1000：**组合复位+触发模式** - 选定触发器输入（tim_trgi）的上升沿会重新初始化计数器，生成寄存器的更新并启动计数器。

1001：**组合门控+复位模式** - 当触发器输入（tim_trgi）为高电平时计数器时钟使能。一旦触发器变为低电平计数器将停止并重置。同时控制计数器的启动和停止。

1010：**编码器模式**：时钟加方向，x2 模式。

1011：**编码器模式**：时钟加方向，x1 模式，CC2P 设置 tim_ti2fp2 边沿敏感度。

1100：**编码器模式**：方向时钟，x2 模式。

1101：**编码器模式**：方向时钟，x1 模式，CC1P 和 CC2P 设置 tim_ti1fp1 和 tim_ti2fp2 边沿敏感度.

1110：**四象限编码器模式**：x1 模式，仅计算 tim_ti1fp1 边沿，CC1P 设置边沿敏感度。

1111：**四象限编码器模式**：x1 模式，仅计算 tim_ti2fp2 边沿，CC2P 设置边沿敏感度。

注：如果选择 tim_ti1f_ed *作为触发输入（*TS=0100*），则不得使用门控模式。实际上，*tim_ti1f_ed *为* TI1F *上的每个过渡输出* 1 *个脉冲，而门控模式检查触发信号的电平。*

注：接收 tim_trgo *或* tim_trgo2 *信号的从机外设（定时器、*ADC *等）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能随意更改。*

21.6.4 TIMx 中断使能寄存器（TIMx_DIER）（x = A，B）

偏移地址：0x00C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TERRIE	IERRIE	DIRIE	IDXIE	Reserved			
								rw	rw	rw	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TIE	Reserved		CC3IE	CC2IE	CC1IE	UIE
									rw			rw	rw	rw	rw

Bit 31:24 保留，必须保持为复位值。

Bit 23 **TERRIE**：转换错误中断使能（Transition error interrupt enable）

0：禁止转换错误中断；

1：使能转换错误中断。

Bit 22 **IERRIE**：索引错误中断使能（Index error interrupt enable）

0：禁止索引错误中断；

1：使能索引错误中断。

Bit 21 **DIRIE**：方向改变中断使能（Direction change interrupt enable）

0：禁止方向改变中断；

1：使能方向改变中断。

Bit 20 **IDXIE**：索引中断使能（Index interrupt enable）

0：禁止索引中断；

1：使能索引中断。

Bit 19:7 保留，必须保持为复位值。

Bit 6 **TIE**：触发信号（TGRI）中断使能（Trigger interrupt enable）

0：禁止触发信号（TGRI）中断；

1：使能触发信号（TGRI）中断。

Bit 5:4 保留，必须保持为复位值。

Bit 3 **CC3IE**：捕获/比较 3 中断使能（Capture/Compare 3 interrupt enable）

0：禁止 CC3 中断；

1：使能 CC3 中断。

Bit 2 **CC2IE**：捕获/比较 2 中断使能（Capture/Compare 2 interrupt enable）

0：禁止 CC2 中断；

1：使能 CC2 中断。

Bit 1 **CC1IE**：捕获/比较 1 中断使能（Capture/Compare 1 interrupt enable）

0：禁止 CC1 中断；

1：使能 CC1 中断。

Bit 0 **UIE**：更新中断使能（Update interrupt enable）

0：禁止更新中断；

1：使能更新中断。

21.6.5 TIMx 状态寄存器（TIMx_SR）（x = A，B）

偏移地址：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TERRF	IERRF	DIRF	IDXF	Reserved			
								rc_w0	rc_w0	rc_w0	rc_w0				

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

Bit 31:24 保留，必须保持为复位值。

Bit 23 **TERRF**：转换错误中断标志（Transition error interrupt flag）

当在编码器模式下检测到转换错误时，硬件会设置此标志当软件将其写入“0”时，该标志将被清除

0：未检测到编码器转换错误

1：已检测到编码器转换错误

Bit 22 **IERRF**：索引错误中断标志（Index error interrupt flag）

当检测到索引错误时，硬件会设置此标志当软件将其写入“0”时，该标志将被清除

0：未检测到索引错误

1：已检测到索引错误

Bit 21 **DIRF**：方向改变中断标志（Direction change interrupt flag）

当编码器模式中的方向发生变化时（TIMx_CR 中的 DIR 位值正在改变），硬件会设置此标志当软件将其写入“0”时，该标志将被清除

0：方向未改变

1：方向已改变

Bit 20 **IDXF**：索引中断标志（Index interrupt flag）

当检测到索引事件时，硬件会设置此标志当软件将其写入“0”时，该标志将被清除

0：未发生索引事件

1：已发生索引事件

Bit 19:12 保留，必须保持为复位值。

Bit 11 **CC3OF**：捕获/比较 3 重复捕获标志

参考 CC1OF 的描述

Bit 10 **CC2OF**：捕获/比较 2 重复捕获标志

参考 CC1OF 的描述

Bit 9 **CC1OF**：捕获/比较 1 重复捕获标志

仅当将相应通道配置为输入捕获模式时，此标志位才会由硬件置 1。通过软件写入“0”可将该位清零。

0：未检测到重复捕获

1：TIMx_CCR1 寄存器中已捕获到计数器值且 CC1IF 标志已置 1。

Bit 8:7 保留，必须保持为复位值。

Bit 6 **TIF**：触发中断标志（Trigger interrupt flag）

在除门控模式以外的所有模式下，当使能从模式控制器后在 tim_trgi 输入上检测到有效边沿时，该标志将由硬件置 1。选择门控模式时，该标志将在计数器启动或停止时置 1。但需要通过软件清零。

0：未发生触发事件。

1：触发中断挂起。

Bit 5:4 保留，必须保持为复位值。

Bit 3 **CC3IF：**捕获/比较 3 中断标志（Capture/Compare 3 interrupt flag）

参考 CC1IF 描述。

Bit 2 **CC2IF**：捕获/比较 2 中断标志（Capture/Compare 2 interrupt flag）

参考 CC1IF 描述。

Bit 1 **CC1IF：**捕获/比较 1 中断标志（Capture/Compare 1 interrupt flag）

此标志由硬件设置，它可以通过软件（输入捕获）或读取 TIMx_CCR1 寄存器（仅输入捕获模式）来清除。

0：无输入捕获发生

1：发生输入捕获

如果通道 CC1 配置为输入：

当在 TIMx_CCR1 寄存器中捕获到计数器值时（在 TIMx_CCER 中定义的 CC1P 和 CC1NP 位设置所确定的 IC1 上检测到边沿时），会设置此位。

Bit 0 **UIF**：更新中断标志（Update interrupt flag）

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

– TIMx_CR1 寄存器中的 UDIS=0，并且在上溢或下溢时。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且 CNT 由触发事件重新初始化时（参考同步控制寄存器描述）。

21.6.6 TIMx 事件产生寄存器（TIMx_EGR）（x = A，B）

偏移地址：0x14

复位：0x0000

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

Bit 15:7 保留，必须保持为复位值。

Bit 6 **TG**：生成触发信号（Trigger generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：TIMx_SR 寄存器中的 TIF 标志置 1。使能后可发生相关中断事件。

Bit 5:4 保留，必须保持为复位值。

Bit 3 **CC3G**：捕获/比较 3 生成（Capture/Compare 3 generation）

参考 CC1G 描述。

Bit 2 **CC2G**：捕获/比较 2 生成（Capture/Compare 2 generation）

参考 CC1G 描述。

Bit 1 **CC1G**：捕获/比较 1 生成（Capture/Compare 1 generation）

此位由软件置 1 以生成事件，并由硬件自动清零。

0：不执行任何操作；

1：通道 1 上生成捕获事件：

如果通道 CC1 配置为输入：

TIMx_CCR1 寄存器中将捕获到计数器当前值。使能时，CC1IF 标志置 1 并发送相应的中断请求。如果 CC1IF 标志已为高电平，CC1OF 标志将置 1。

Bit 0 **UG**：更新生成（Update generation）

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作；

1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。如果选择中心对齐模式或 DIR=0（递增计数），计数器将清零；如果 DIR=1（递减计数），计数器将使用自动重载值（TIMx_ARR）。

21.6.7 TIMx 捕获/比较模式寄存器 1 [复用]（TIMx_CCMR1）（x = A，B）

偏移地址：0x018

复位值：0x0000 0000

该寄存器只能用于输入捕获模式。通道方向通过配置相应的 CCxS 位进行定义。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC2F[3:0]				IC2PSC[1:0]		CC2S[1:0]		IC1F[3:0]				IC1PSC[1:0]		CC1S[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

Bit 31:16 保留，必须保持为复位值。

Bit 15:12 **IC2F[3:0]：**输入捕获 2 滤波器（Input capture 2 filter）

Bit 11:10 **IC2PSC[1:0]：**输入/捕获 2 预分频器（input capture 2 prescaler）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择 （Capture/compare 2 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：保留

01：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti2 上；

10：CC2 通道被配置为输入，tim_ic2 映射在 tim_ti1 上；

11：CC2 通道被配置为输入，tim_ic2 映射在 tim_trc 上。此模此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIMx_CCER *中的*CC2E = 0*），才可向*CC2S *位写入数据。*

Bit 7:4 **IC1F[3:0]：**输入捕获 1 滤波器（Input capture 1 filter）

此位域可定义 tim_ti1 输入的采样频率和适用于 tim_ti1 的数字滤波器带宽。数字滤波器由事件计数器组成，每 N 个事件才视为一个有效边沿：

0000：无滤波器，按 f_{DTS} 频率进行采样

0001：$f_{SAMPLING}=f_{tim_ker_ck}$，N=2。

0010：$f_{SAMPLING}=f_{tim_ker_ck}$，N=4。

0011：$f_{SAMPLING}=f_{tim_ker_ck}$，N=8。

0100：$f_{SAMPLING}=f_{DTS}/2$，N=6。

0101：$f_{SAMPLING}=f_{DTS}/2$，N=8。

0110：$f_{SAMPLING}=f_{DTS}/4$，N=6。

0111：$f_{SAMPLING}=f_{DTS}/4$，N=8。

1000：$f_{SAMPLING}=f_{DTS}/8$，N=6。

1001：$f_{SAMPLING}=f_{DTS}/8$，N=8。

1010：$f_{SAMPLING}=f_{DTS}/16$，N=5。

1011：$f_{SAMPLING}=f_{DTS}/16$，N=6。

1100：$f_{SAMPLING}=f_{DTS}/16$，N=8。

1101：$f_{SAMPLING}=f_{DTS}/32$，N=5。

1110：$f_{SAMPLING}=f_{DTS}/32$，N=6。

1111：$f_{SAMPLING}=f_{DTS}/32$，N=8。

Bit 3:2 **IC1PSC[1:0]：**输入/捕获 1 预分频器（Input capture 1 prescaler）

此位域定义 CC1 输入（tim_ic1）的预分频比。

只要 CC1E=0（TIMx_CCER 寄存器），预分频器便立即复位。

00：无预分频器，捕获输入上每检测到一个边沿便执行捕获

01：每发生 2 个事件便执行一次捕获；

10：每发生 4 个事件便执行一次捕获；

11：每发生 8 个事件便执行一次捕获。

Bit 1:0 **CC1S[1:0]：**捕获/比较 1 选择（Capture/Compare 1 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：保留

01：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti1 上；

10：CC1 通道被配置为输入，tim_ic1 映射在 tim_ti2 上；

11：CC1 通道被配置为输入，tim_ic1 映射在 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

*注：仅当通道关闭时（*TIMx_CCER *中的* CC1E = 0*），才可向* CC1S *位写入数据。*

21.6.8 TIMx 捕获/比较模式寄存器 2 [复用]（TIMx_CCMR2）（x = A，B）

偏移地址：0x01C

复位值：0x0000 0000

该寄存器只能用于输入捕获模式。通道方向通过配置相应的 CCxS 位进行定义。

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IC3F[3:0]				IC3PSC[1:0]		CC3S[1:0]	
								rw	rw	rw	rw	rw	rw	rw	rw

输入捕获模式

Bit 31:8 保留，始终读为 0。

Bit 7:4 **IC3F[3:0]：**输入捕获 3 滤波器（Input capture 3 filter）

Bit 3:2 **IC3PSC[1:0]：**输入/捕获 1 预分频器（Input capture 3 prescaler）

Bit 1:0 **CC3S[1:0]：**捕获/比较 3 选择（Capture/Compare 3 selection）

此位域定义通道方向（输入/输出）以及所使用的输入。

00：保留

01：CC3 通道被配置为输入，tim_ic3 映射在 tim_ti3 上；

10：保留

11：CC3 通道被配置为输入，tim_ic3 映射在 tim_trc 上。此模式仅在通过 TS 位（TIMx_SMCR 寄存器）选择内部触发输入时有效

注：仅当通道关闭时（TIMx_CCER 中的 CC3E = 0），才可向 CC3S 位写入数据。

21.6.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x = A，B）

偏移地址：0x020

复位值：0x0000

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

Bit 15:12 保留，必须保持为复位值。

Bit 11 **CC3NP**：输入/捕获 3 输出极性（Capture/Compare 3 output polarity）

参考 CC1NP 的描述。

Bit 10 保留，必须保持为复位值。

Bit 9 **CC3P**：输入/捕获 3 输出极性（Capture/Compare 3 output polarity）

参考 CC1P 的描述。

Bit 8 **CC3E**：输入/捕获 3 输出使能（Capture/Compare 3 output enable）

参考 CC1E 的描述。

Bit 7 **CC2NP**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1NP 的描述。

Bit 6 保留，必须保持为复位值。

Bit 5 **CC2P**：输入/捕获 2 输出极性（Capture/Compare 2 output polarity）

参考 CC1P 的描述。

Bit 4 **CC2E**：输入/捕获 2 输出使能（Capture/Compare 2 output enable）

参考 CC1E 的描述。

Bit 3 **CC1NP**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

CC1 通道配置为输入：

此位与 CC1P 配合使用，用以定义 tim_ti1fp1/ tim_ti2fp1 的极性。请参见 CC1P 说明。

Bit 2 保留，必须保持为复位值。

Bit 1 **CC1P**：输入/捕获 1 输出极性（Capture/Compare 1 output polarity）

0：OC1 边沿灵敏度选择（输入模式，参见以下）

1：OC1 边沿灵敏度选择（输入模式，参见以下）

CC1 通道配置为输入，CC1NP/CC1P 位可针对触发或捕获操作选择 TI1FP1 和 TI2FP1 的极性。

CC1NP=0，CC1P=0：非反相/上升沿触发。电路对 TIxFP1 上升沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=0，CC1P=1：反相/下降沿触发。电路对 TIxFP1 下降沿敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 反相（在门控模式或编码器模式下执行触发操作）。

CC1NP=1，CC1P=1：非反相/上升沿和下降沿均触发。电路对 TIxFP1 上升沿和下降沿都敏感（在复位模式、外部时钟模式或触发模式下执行捕获或触发操作），TIxFP1 未反相（在门控模式下执行触发操作）。编码器模式下不得使用此配置。

CC1NP=1，CC1P=0：此配置是保留的，不应该使用此配置。

Bit 0 **CC1E**：输入/捕获 1 输出使能（Capture/Compare 1 output enable）

0：禁止捕获模式/OC1 未激活

1：使能捕获模式/ OC1 信号输出到对应的输出引脚

21.6.10 TIMx 计数器（TIMx_CNT）（x = A，B）

偏移地址：0x024

复位值：0x0000 0000

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

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

Bit 31 **UIFCPY：** 值取决于 TIMx_CR1 的 UIFREMAP

如果 UIFREMAP = 0

保留

如果 UIFREMAP = 1

UIFCPY：复制 UIF

该位是 TIMx_SR 寄存器中 UIF 位的一个只读副本。

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CNT[15:0]：**计数器的值（Counter value）

该寄存器（CNT[15:0]）保存的是计数器的值。

21.6.11 TIMx 预分频器（TIMx_PSC）（x = A，B）

偏移地址：0x028

复位值：0x0000

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

Bit 15:0 **PSC[15:0]：**预分频器的值（Prescaler value）

计数器时钟频率（$f_{tim_cnt_ck}$）等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零时，或在配置为“复位模式”时通过触发控制器清零时）时要装载到活动预分频器寄存器的值。

21.6.12 TIMx 自动重装载寄存器（TIMx_ARR）（x = A，B）

偏移地址：0x02C

复位：0x0000 FFFF

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

Bit 15:0 **ARR[15:0]：**自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

当自动重载值为空时，计数器不工作。

该寄存器存储的是自动重新加载的值

21.6.13 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x = A，B）

偏移地址：0x034

复位值：0x0000 0000

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

Bit 15:0 **CCR1[15:0]：**捕获/比较 1 值（Capture/Compare 1 value）

若 CC1 通道配置为输入：

CCR1 为上一个输入捕获 1 事件（tim_ic1）发生时的计数器值。TIMx_CCR1 是只读的并且不可编程。

CCR1[15:0]位存储了捕获值。

21.6.14 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x = A，B）

偏移地址：0x038

复位值：0x0000 0000

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

Bit 15:0 **CCR2[15:0]：**捕获/比较 2 值（Capture/Compare 2 value）

若 CC2 通道配置为输入：

CCR2 为上一个输入捕获 2 事件（tim_ic2）发生时的计数器值。TIMx_CCR2 是只读的并且不可编程。

CCR2[15:0]位存储了捕获值。

21.6.15 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x = A，B）

偏移地址：0x03C

复位值：0x0000 0000

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

Bit 15:0 **CCR3[15:0]：**捕获/比较 3 值（Capture/Compare 3 value）

若 CC3 通道配置为输入：

CCR3 为上一个输入捕获 3 事件（tim_ic3）发生时的计数器值。TIMx_CCR3 是只读的并且不可编程。

CCR3[15:0]位存储了捕获值。

21.6.16 TIMx 编码器控制寄存器（TIMx_ECR）（x = A，B）

偏移地址：0x058

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IPOS[1:0]		FIDX	Reserved		IDIR[1:0]		IE
								rw	rw	rw			rw	rw	rw

Bit 31:8 保留，必须保持为复位值。

Bit 7:6 **IPOS[1:0]：索引位置**（Index positioning）

在正交编码器模式（SMS[3:0] = 0001，0010，0011，1110，1111）下，该位指示了索引事件在哪个 AB 输入配置下会重置计数器。

00：当 AB=00 时，索引重置计数器

01：当 AB=01 时，索引重置计数器

10：当 AB=10 时，索引重置计数器

11：当 AB=11 时，索引重置计数器

在方向时钟模式或时钟加方向模式（SMS[3:0] = 1010，1011，1100，1101）下，此位指示了索引事件在哪个时钟水平上重置计数器。在双向时钟模式下，这适用于两个时钟输入。具体为：

x0：当时钟=0 时，索引重置计数器

x1：当时钟=1 时，索引重置计数器

注：IPOS[1]位不重要

Bit 5 **FIDX[1:0]：首位索引**（First index）

该位指示是否仅考虑首位索引

0：索引始终处于活跃状态

1：仅首个索引会重置计数器

Bit 4:3 保留，必须保持为复位值。

Bit 2:1 **IDIR[1:0]：索引方向**（Index direction）

该位指示索引事件在哪个方向上重置计数器。

00：无论方向如何，索引都会重置计数器

01：仅在递增计数时，索引会重置计数器

10：仅在递减计数时，索引会重置计数器

11：保留

注：当 IE 位被复位（禁止索引）时，必须写入 IDR[1:0]位域

Bit 0 **IE：索引使能**（Index enable）

该位指示索引事件是否能重置计数器

0：禁止索引

1：使能索引

21.6.17 TIMx 定时器输入选择寄存器（TIMx_TISEL）（x = A，B）

偏移地址：0x05C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														TI3SEL[1:0]	
														rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TI2SEL[1:0]		Reserved					TI1SEL[2:0]		
						rw	rw						rw	rw	rw

Bit 31:18 保留，必须保持为复位值。

Bit 17:16 **TI3SEL[1:0]：**选择 tim_ti3[3:0]输入（Selects tim_ti3[3:0] input）

00：tim_ti3_in0: TIMx_CH3

01：tim_ti3_in1

10：tim_ti3_in2

11：tim_ti3_in3

参考章节：TIMA/TIMB 引脚和内部信号获得关联列表。

Bit 15:10 保留，必须保持为复位值。

Bit 9:8 **TI2SEL[1:0]：**选择 tim_ti2[3:0]输入（Selects tim_ti2[3:0] input）

00：tim_ti2_in0: TIMx_CH2

01：tim_ti2_in1

10：tim_ti2_in2

11：tim_ti2_in3

参考章节：TIMA/TIMB 引脚和内部信号获得关联列表。

Bit 7:3 保留，必须保持为复位值。

Bit 2:0 **TI1SEL[2:0]：**选择 tim_ti1[4:0]输入（Selects tim_ti1[4:0] input）

000：tim_ti1_in0: TIMx_CH1

001：tim_ti1_in1

010：tim_ti1_in2

011：tim_ti1_in3

100：tim_ti1_in4

其他：保留

参考章节：TIMA/TIMB 引脚和内部信号获得关联列表。

22 基本定时器（TIM6/TIM7）

22.1 TIM6/TIM7 简介

基本定时器 TIM6/TIM7 包含一个 16 位自动重载计数器，该计数器由可编程预分频器驱动。
他们可以用作通用定时器以生成时基。
这些定时器彼此完全独立，不共享任何资源。

22.2 TIM6/TIM7 主要功能

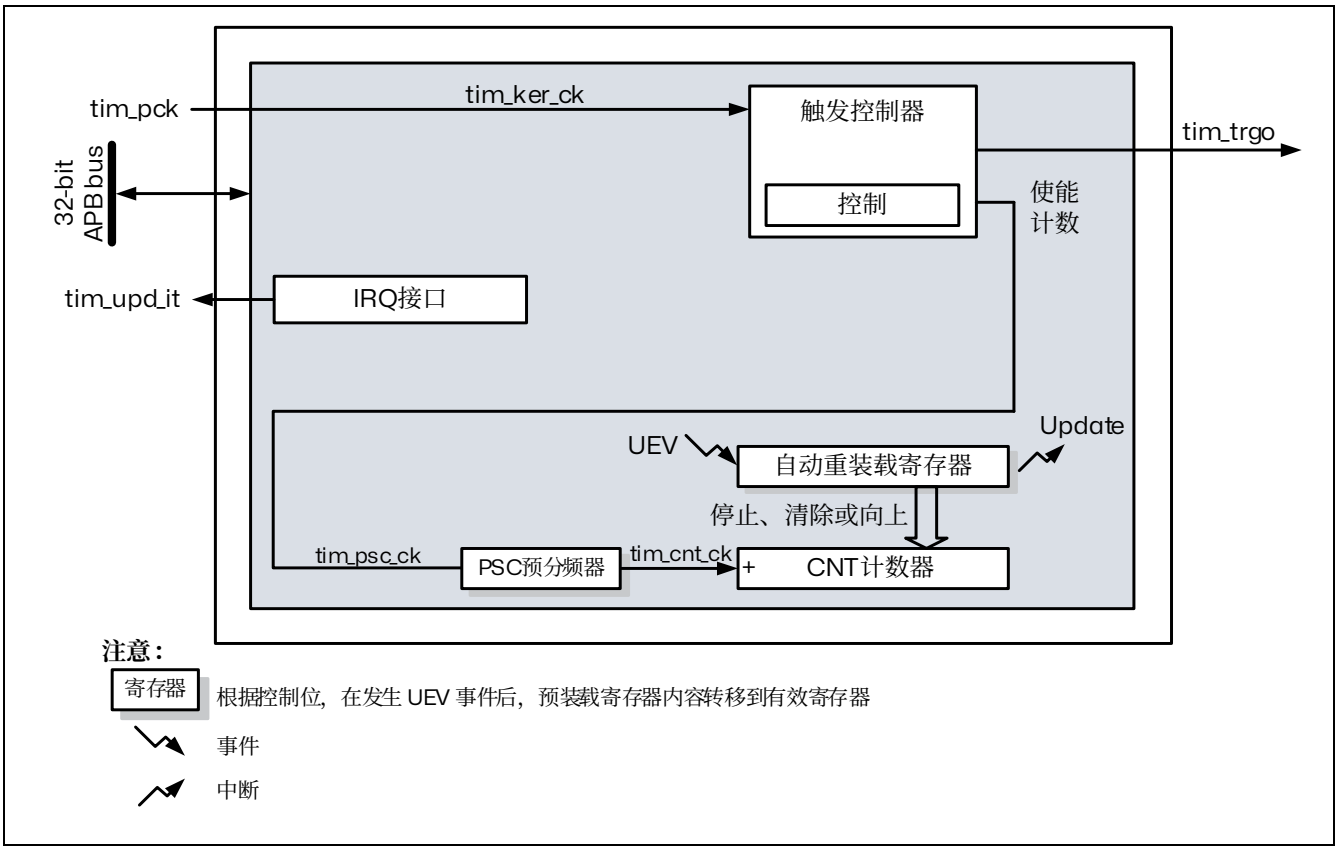
基本定时器（TIM6/TIM7）功能包括：

- 16 位自动重载递增计数器
- 16 位可编程预分频器，用于对计数器时钟频率进行分频（即运行时修改），分频系数介于 1 和 65536 之间
- 发生如下更新事件时会生成中断：计数器上溢

22.3 TIM6/TIM7 功能描述

22.3.1 框图

图 22.1 基本定时器框图



22.3.2 TIM6/TIM7 内部信号

本节中的下表格总结了 TIM 的输入和输出。

表 22.1 TIM 内部输入/输出信号

内部信号名称	信号类型	描述
tim_pclk	输入	定时器 APB 时钟。
tim_ker_ck	输入	定时器内核时钟。此时钟必须与 tim_plck 同步（来自同一源）。 时钟比率 tim_ker_ck/ tim_pclk 必须为整数： 1，2，3，...，16（最大值）。
tim_trgo	输出	内部触发器输出。此触发器可以触发其他片上外设（ADC）。
tim_upd_it	输出	定时器更新事件中断。

22.3.3 TIM6/TIM7 时钟

定时器总线接口由 tim_pclk APB 时钟驱动。

计数器时钟 tim_ker_ck 连接到 tim_pclk 输入。

CEN 位（TIMx_CR1 寄存器中）和 UG 位（TIMx_EGR 寄存器中）为实际控制位，并且只能通过软件进行更改（UG 除外，仍自动清零）。当对 CEN 位写入 1 时，预分频器的时钟就由内部时钟 tim_ker_ck 提供。

下图显示了正常模式下控制电路与递增计数器的行为（没有预分频的情况下）。

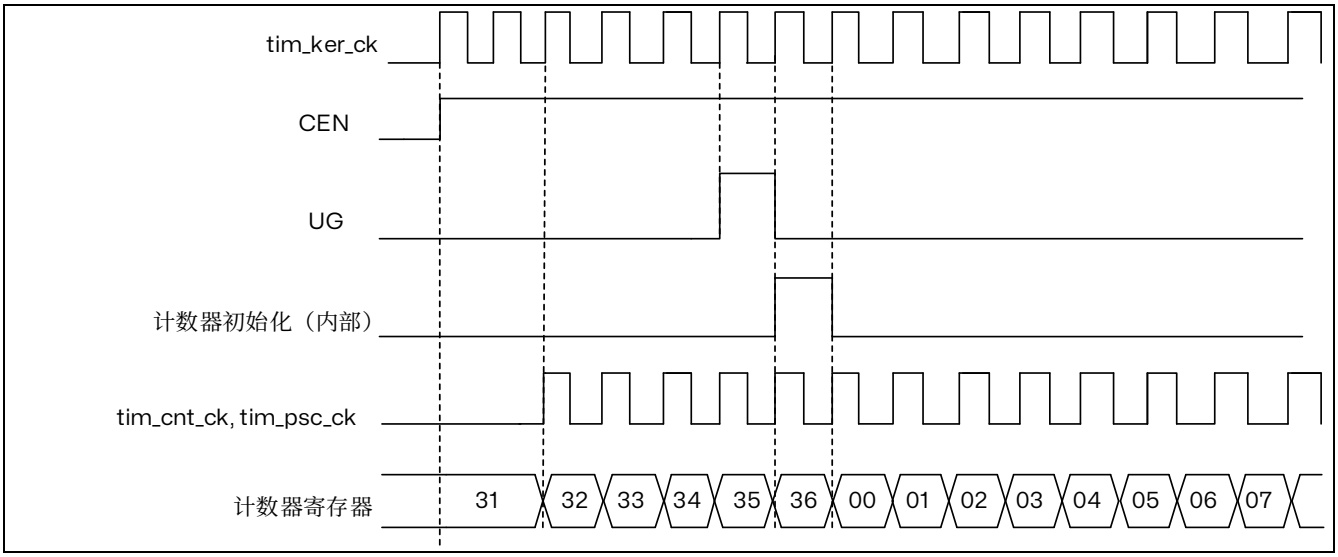


图 22.2 一般模式下的控制电路，内部时钟分频系数为 1

22.3.4 时基单元

可编程定时器的主要模块由一个 16 位递增计数器及其相关的自动重载寄存器组成。计数器的时钟可通过预分频器进行分频。

计数器、自动重载寄存器和预分频器寄存器可通过软件进行读写。即使在计数器运行时也可执行读写操作。

时基单元包含：

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）

自动重载寄存器是预装载的。每次尝试对自动重载寄存器执行读写操作时，都会访问预装载寄存器。预装载寄存器的内容既可以直接传送到影子寄存器，也可以在每次发生更新事件 UEV 时传送到影子寄存器，这取决于 TIMx_CR1 寄存器中的自动重载预装载使能位(ARPE)。当计数器达到上溢值并且 TIMx_CR1 寄存器中的 UDIS 位为 0 时，将发送更新事件。该更新事件也可由软件产生。下文将针对各配置的更新事件的产生进行详细介绍。

计数器由预分频器输出 CK_CNT 提供时钟，仅当 TIMx_CR1 寄存器中的计数器启动位（CEN）置 1 时，才会启动计数器。

请注意，实际的计数器使能信号 CNT_EN 在 CEN 置 1 的一个时钟周期后被置 1。

预分频器描述

预分频器可对计数器时钟频率进行分频，分频系数介于 1 和 65536 之间。该预分频器基于 TIMx_PSC 寄存器中的 16 位寄存器所控制的 16 位计数器。由于 TIMx_PSC 控制寄存器有缓冲，因此可对预分频器进行实时更改。而新的预分频比将在下一更新事件发生时被采用。

下图给出了在预分频比发生实时变化时一些计数器行为的示例。

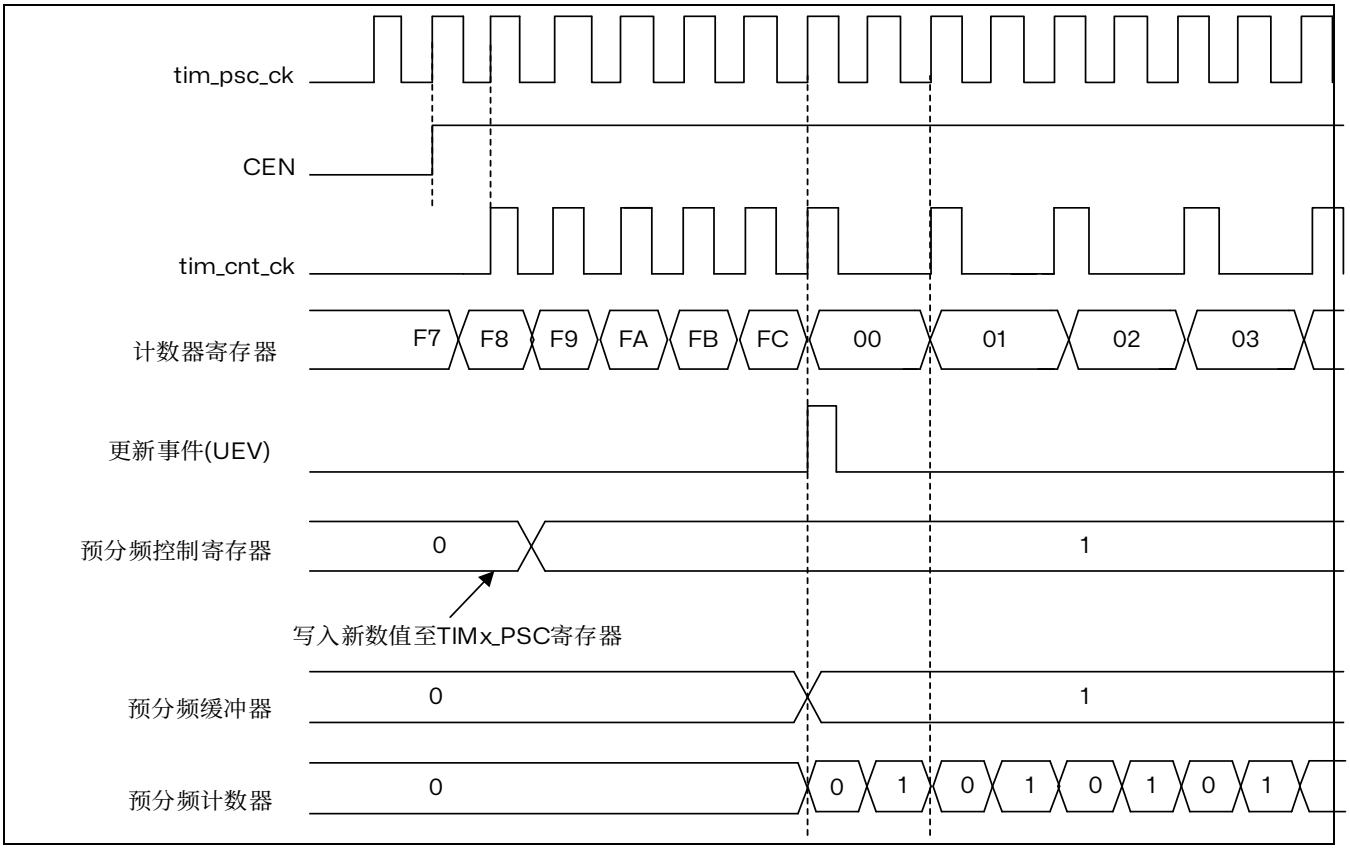


图 22.3 当预分频器的参数从 1 变到 2 时，计数器的时序图

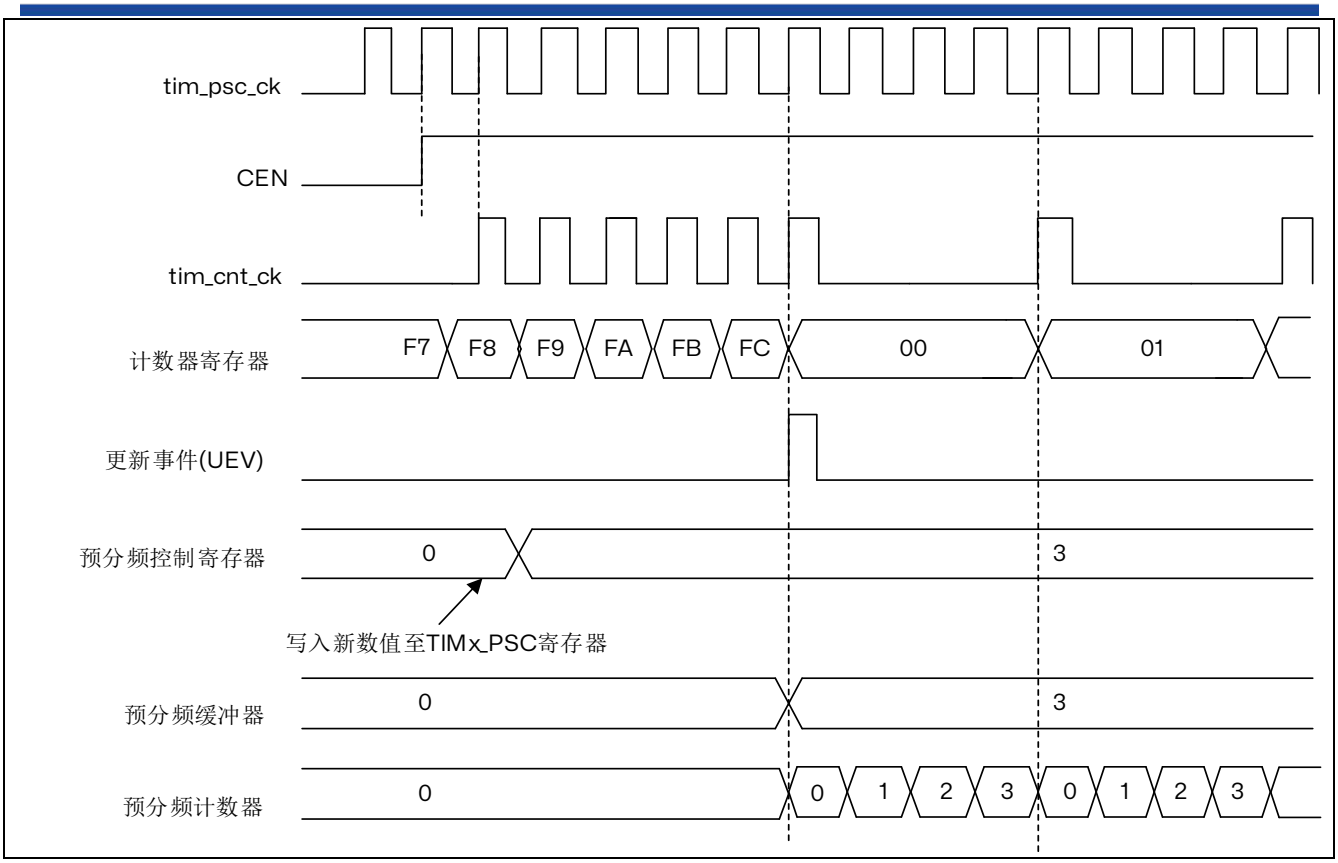


图 22.4 当预分频器的参数从 1 变到 4 时，计数器的时序图

22.3.5 计数器模式

计数器从 0 计数到自动重载值（TIMx_ARR 寄存器的内容），然后重新从 0 开始计数并生成计数器上溢事件。

每次发生计数器上溢时会生成更新事件，或将 TIMx_EGR 寄存器中的 UG 位置 1（通过软件或使用从模式控制器）也可以生成更新事件。

通过软件将 TIMx_CR1 寄存器中的 UDIS 位置 1 可禁止 UEV 事件。这可避免向预装载寄存器写入新值时更新影子寄存器。这样，直到 UDIS 位中写入 0 前便不会生成任何更新事件，但计数器和预分频器计数器都会重新从 0 开始计数（而预分频比保持不变）。此外，如果 TIMx_CR1 寄存器中的 URS 位（更新请求选择）已置 1，则将 UG 位置 1 会生成更新事件 UEV，但不会将 UIF 标志置 1（因此，不会发送任何中断请求）。

发生更新事件时，将更新所有寄存器且将更新标志（TIMx_SR 寄存器中的 UIF 位）置 1（取决于 URS 位）：

- 使用预装载值（TIMx_PSC 寄存器的内容）重新装载预分频器的缓冲区
- 使用预装载值（TIMx_ARR）更新自动重载影子寄存器

以下各图显示了当 TIMx_ARR=0x36 时不同时钟频率下计数器行为的示例。

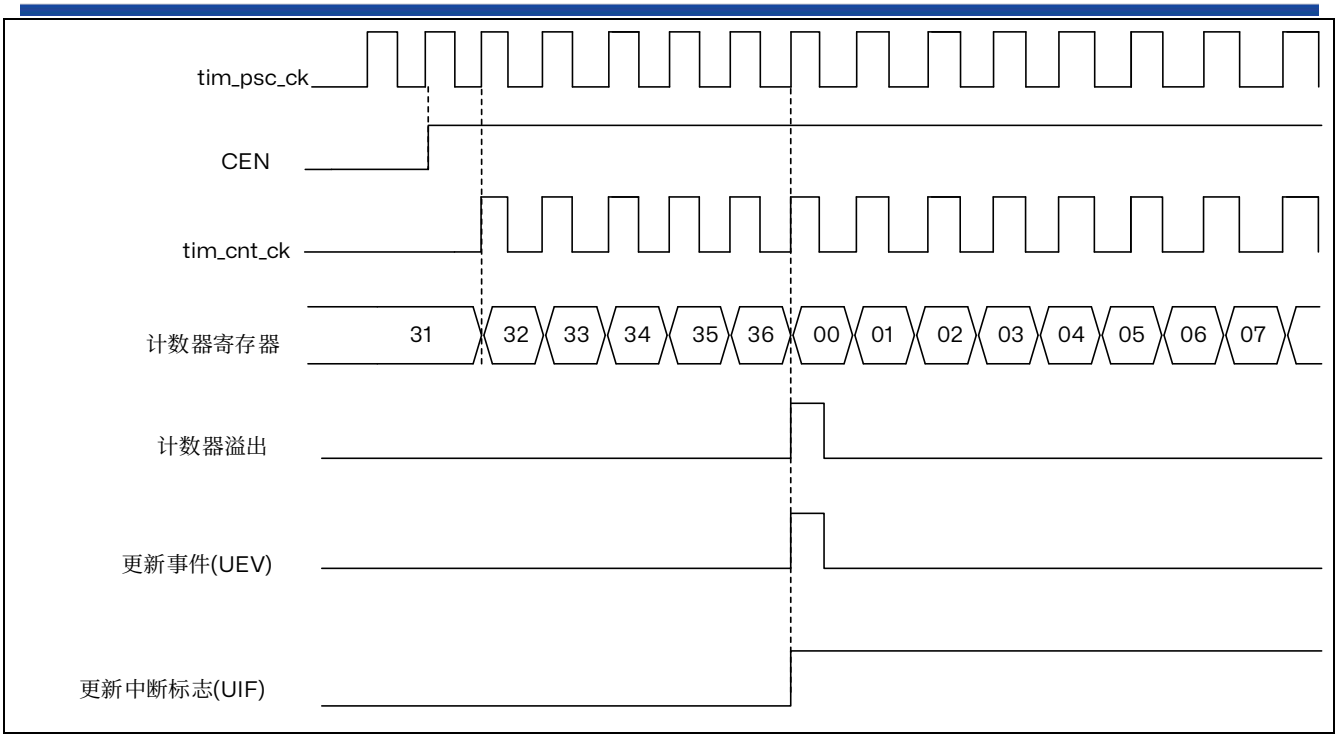


图 22.5 计数器时序图，内部时钟分频系数为 1

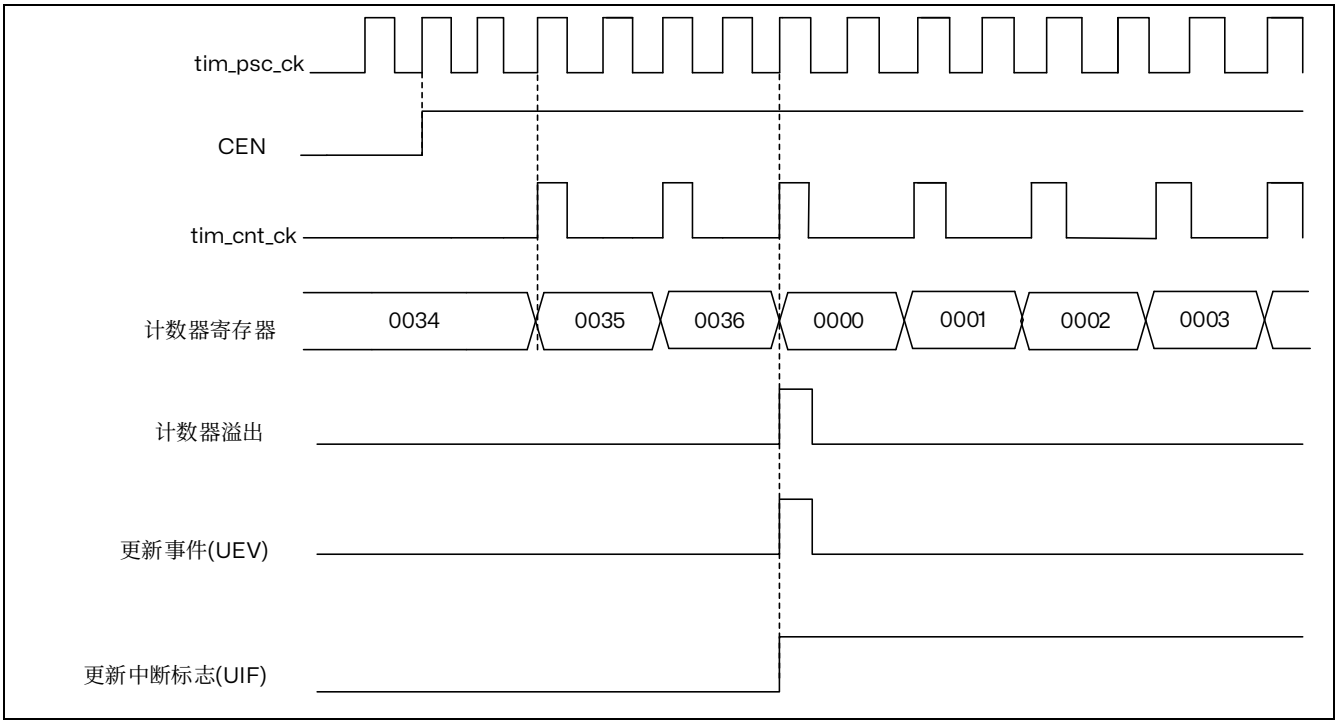


图 22.6 计数器时序图，内部时钟分频系数为 2

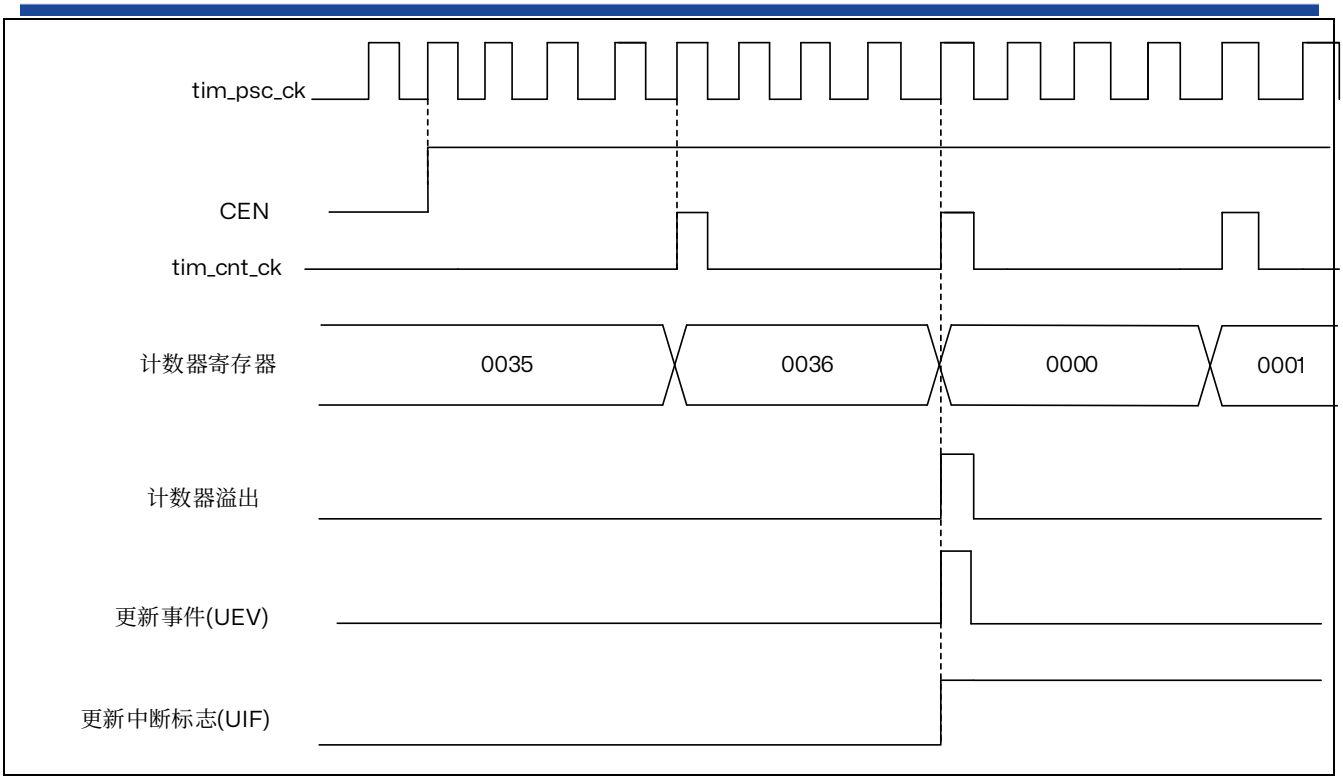


图 22.7 计数器时序图，内部时钟分频系数为 4

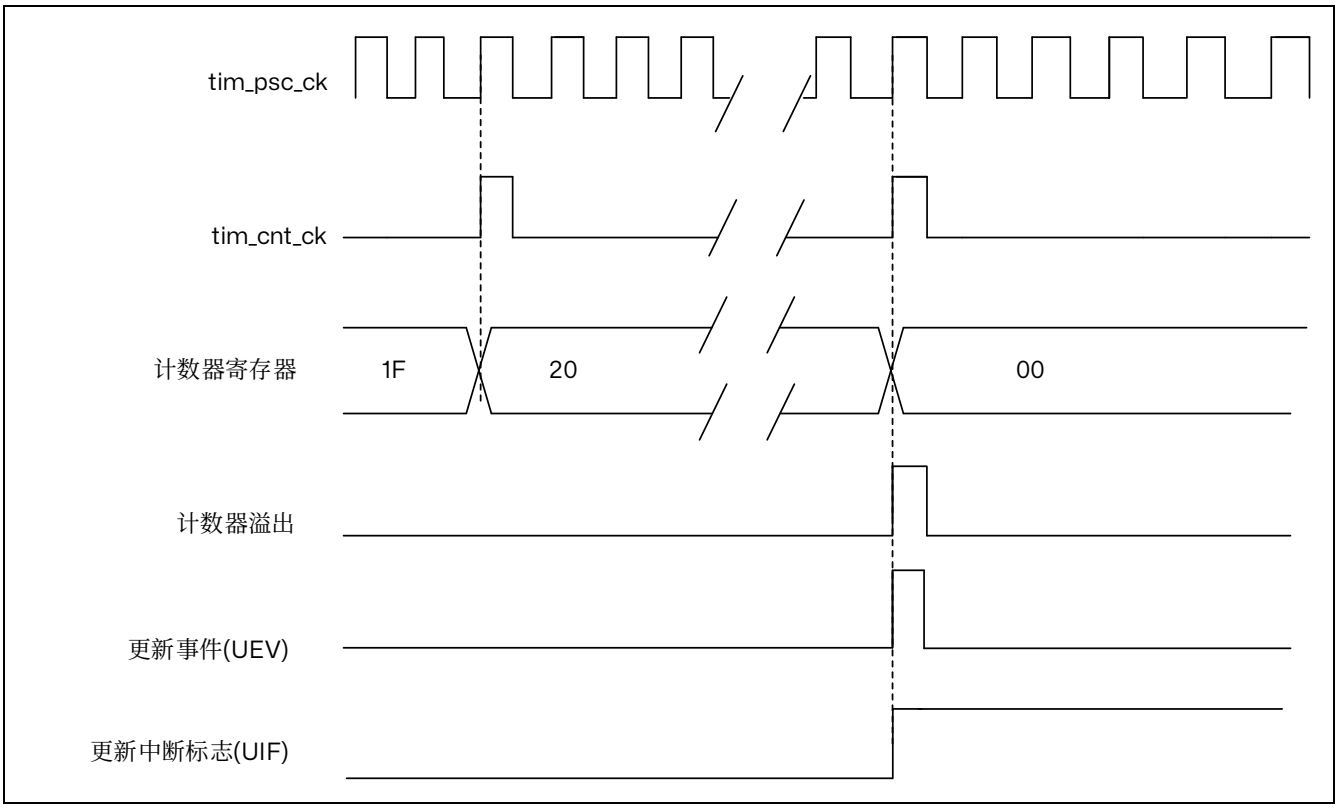


图 22.8 计数器时序图，内部时钟分频系数为 N

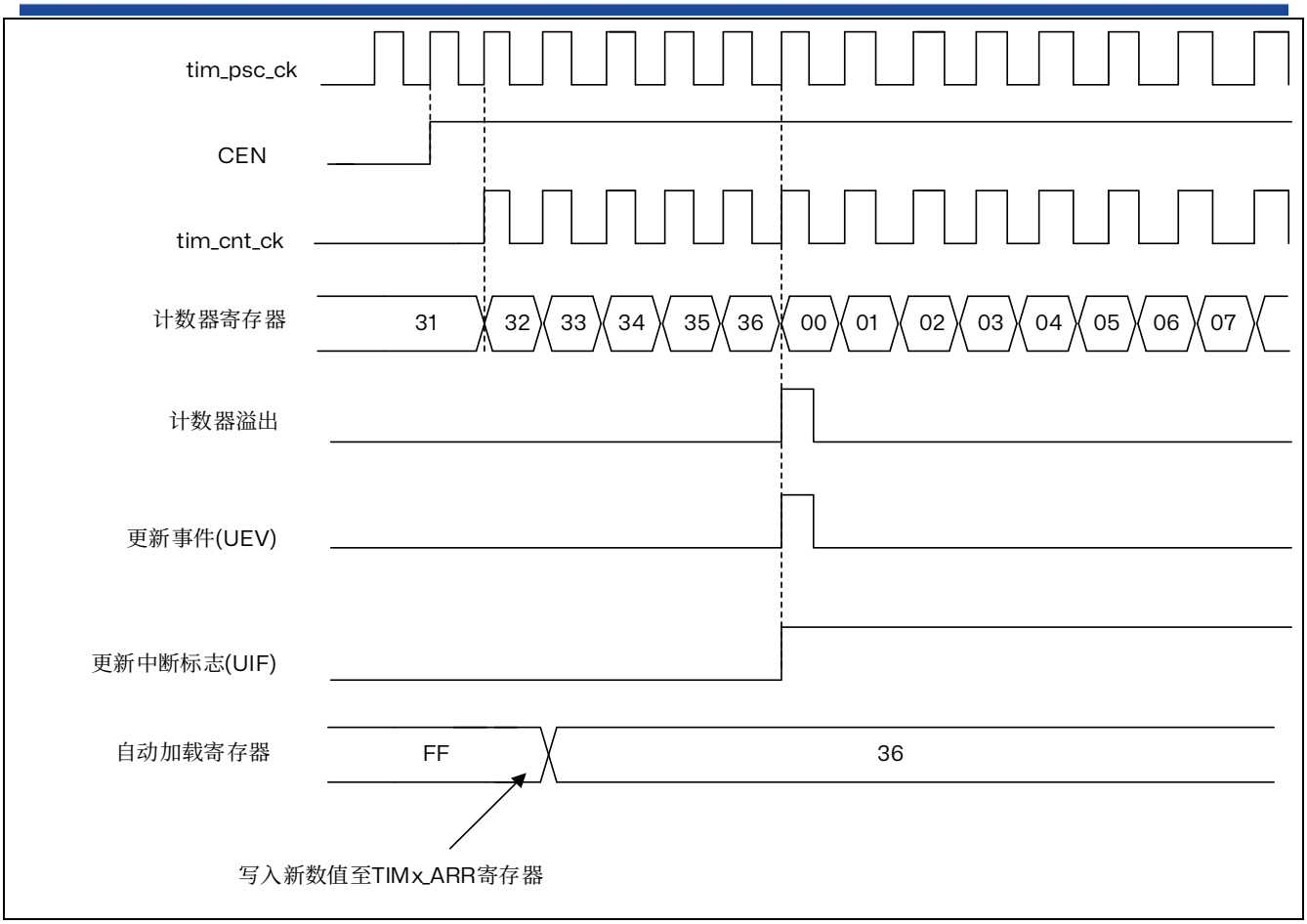


图 22.9 计数器时序图，当 ARPE=0 时的更新事件（TIMx_ARR 没有预装入）

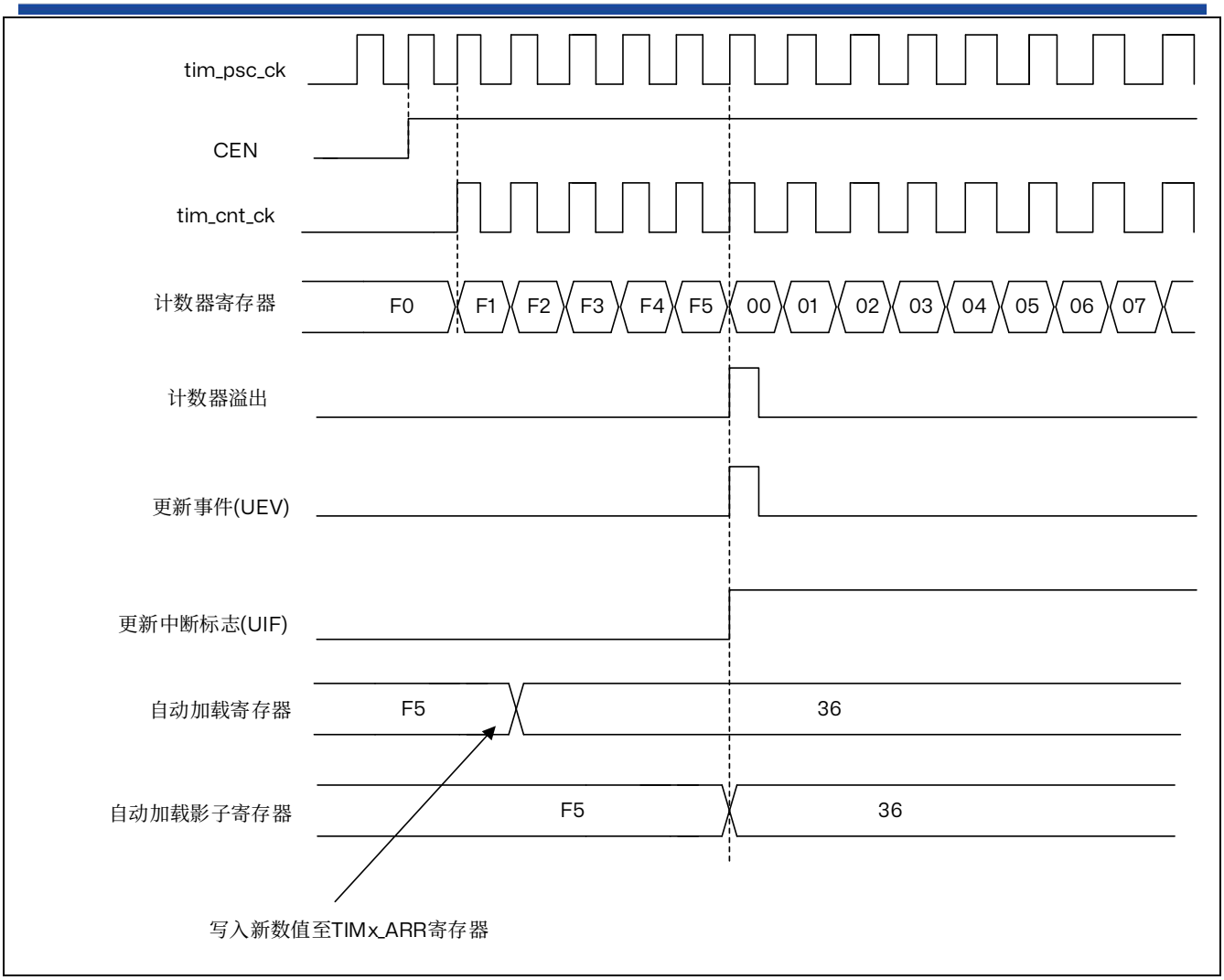


图 22.10 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIMx_ARR）

22.3.6 UIF 位重映射

TIMx_CR1 寄存器中的 IUFREMAP 位强制将更新中断标志 UIF 连续复制到定时器计数器寄存器的第 31 位（TIMxCNT[31]）。这允许以原子方式读取计数器的值和一个潜在的由 UIFCPY 标志指示的翻转条件。在特定情况下，它可以避免由背景任务（读取计数器）和中断（更新中断）之间的处理引起的竞争条件，从而简化计算。

UIF 和 UIFCPY 标志之间没有延迟。

22.3.7 ADC 触发

定时器可以使用各种内部信号（例如重置、使能或比较事件）生成 ADC 触发事件。

注意：接收 tim_trgo 信号的从属外设（例如定时器、ADC）的时钟必须在从主定时器接收事件之前使能，并且时钟频率（预分频器）在从主定时器接收触发器时不能更改。

22.3.8 调试模式

当微控制器进入调试模式（内核停止）时，TIMx 计数器可以继续正常工作或停止。
在调试模式下，每个定时器的行为可以通过 Debug 支持（DBG）模块的专用配置位进行编程。
有关更多详细信息，请参阅调试部分。

22.4 TIM6/TIM7 低功耗模式

表 22.2 低功耗模式对 TIM6/TIM7 的影响

模式	描述
睡眠	无影响，外设是运行的。中断会导致设备退出睡眠模式。
停机	定时器操作被停止并保留其寄存器内容。无法生成中断。
待机	定时器被断电并且退出待机模式后必须重新初始化。

22.5 TIM6/TIM7 中断

TIM6/TIM7 可以生成一个单一中断，如下表所示。

表 22.3 中断请求

中断缩写	中断事件	事件标志	使能控制位	中断清除方法	从睡眠模式退出	从停机和待机退出
TIM6 TIM7	更新	UIF	UIE	UIF 写 0	是	否

22.6 TIMx 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
外设寄存器可以通过半字（16 位）或字（32 位）进行访问。

22.6.1 TIMx 控制寄存器 1（TIMx_CR1）（x = 6/7）

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
Reserved				UIFREMAP	Reserved			ARPE	Reserved			OPM	URS	UDIS	CEN
				rw				rw				rw	rw	rw	rw

Bit 15:12 保留，必须保持为复位值。

Bit 11 **UIFREMAP**：UIF 状态位重映射（UIF status bit remapping）

0：无重映射。UIF 状态位未复制到 TIMx_CNT 寄存器位 31

1：重映射使能。UIF 状态位复制到 TIMx_CNT 寄存器位 31

Bit 10:8 保留，必须保持为复位值。

Bit 7 **ARPE**：自动重载预装载使能（Auto-reload preload enable）

0：TIMx_ARR 寄存器不进行缓冲；

1：TIMx_ARR 寄存器进行缓冲。

Bit 6:4 保留，必须保持为复位值。

Bit 3 **OPM**：单脉冲模式（One pulse mode）

0：计数器在发生更新事件时不会停止计数；

1：计数器在发生下一更新事件时停止计数（将 CEN 位清零）

Bit 2 **URS**：更新请求源（Update request source）

此位由软件置 1 和清零，用以选择 UEV 事件源。

0：使能时，所有以下事件都会生成更新中断。此类事件包括：

- 计数器上溢/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

1：使能时，只有计数器上溢/下溢会生成更新中断。

Bit 1 **UDIS**：更新禁止（Update disable）

此位由软件置 1 和清零，用以使能/禁止 UEV 事件生成。

0：使能 UEV。更新（UEV）事件可通过以下事件之一生成：

- 计数器上出/下溢
- 将 UG 位置 1
- 通过从模式控制器生成的更新事件

然后更新影子寄存器的值。

1：禁止 UEV。不会生成更新事件，各影子寄存器的值（ARR、PSC）保持不变。但如果将 UG 位置 1，或者从从模式控制器接收到硬件复位，则会重新初始化计数器和预分频器。

Bit 0 **CEN**：计数器使能（Counter enable）

0：禁止计数器；

1：使能计数器。

单脉冲模式下，当发生更新事件时会自动将 CEN 位清零。

22.6.2 TIMx 控制寄存器 2（TIMx_CR2）（x = 6/7）

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
Reserved									MMS[2:0]			Reserved			
									rw	rw	rw				

Bit 15:7 保留，必须保持为复位值。

Bit 6:4 **MMS[2:0]：**主模式选择（Master mode selection）

这些位可选择主模式下将要发送到从定时器以实现同步的信息(tim_trgo)。这些位的组合如下：

000：**复位** – TIMx_EGR 寄存器中的 UG 位用作触发输出（tim_trgo）。

001：**使能** – 计数器使能信号 CNT_EN 用作触发输出（tim_trgo）。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号可由 CEN 控制位产生。

010：**更新** – 选择更新事件作为触发输出（tim_trgo）。例如，主定时器可用作从定时器的预分频器。

注：从定时器或接收 tim_trgo *的外设时钟必须在从主定时器接收事件之前使能，在主定时器接收触发期间不能实时更改。*

Bit 3:0 保留，必须保持为复位值。

22.6.3 TIMx/DMA/中断使能寄存器（TIMx_DIER）（x = 6/7）

偏移地址：0x0C

复位值：0x0000 0000

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

Bit 15:1 保留，必须保持为复位值。

Bit 0 **UIE**：更新中断使能（Update interrupt enable）

0：禁止更新中断；

1：使能更新中断。

22.6.4 TIMx 状态寄存器（TIMx_SR）（x = 6/7）

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
Reserved															UIF
															rc_w0

Bit 15:1 保留，必须保持为复位值。

Bit 0 **UIF**：更新中断标志（Update interrupt flag）

该位在发生更新事件时通过硬件置 1。但需要通过软件清零。

0：未发生更新。

1：更新中断挂起。该位在以下情况下更新寄存器时由硬件置 1：

– TIMx_CR1 寄存器中的 UDIS=0，并且重复计数器值上溢或下溢时（重复计数器= 0 时更新）。

– TIMx_CR1 寄存器中的 URS=0 且 UDIS=0，并且由软件使用 TIMx_EGR 寄存器中的 UG 位重新初始化 CNT 时。

22.6.5 TIMx 事件产生寄存器（TIMx_EGR）（x = 6/7）

偏移地址：0x14

复位：0x0000

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

Bit 15:1 保留，必须保持为复位值。

Bit 0 **UG**：更新生成（Update generation）

该位可通过软件置 1，并由硬件自动清零。

0：不执行任何操作；

1：重新初始化计数器并生成一个寄存器更新事件。请注意，预分频器计数器也将清零（但预分频比不受影响）。

22.6.6 TIMx 计数器（TIMx_CNT）（x = 6/7）

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
UIF CPY	Reserved														
r															

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

Bit 31 **UIFCPY**：复制 UIF

该位是 TIMx_SR 寄存器中 UIF 位的一个只读副本。如果 TIMx_CR1 中的 UIFREMAP 位被复位，那位 31 被保留且读为 0

Bit 30:16 保留，必须保持为复位值。

Bit 15:0 **CNT[15:0]**：计数器的值（Counter value）

该寄存器（CNT[15:0]）保存的是计数器的值。

22.6.7 TIMx 预分频器（TIMx_PSC）（x = 6/7）

偏移地址：0x28

复位值：0x0000

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

Bit 15:0 **PSC[15:0]：**预分频器的值（Prescaler value）

计数器时钟频率（$f_{tim_cnt_ck}$）等于 $f_{tim_psc_ck}$ / (PSC[15:0] + 1)。

PSC 包含每次发生更新事件（包括计数器通过 TIMx_EGR 寄存器中的 UG 位清零）时要装载到活动预分频器寄存器的值。

22.6.8 TIMx 自动重装载寄存器（TIMx_ARR）（x = 6/7）

偏移地址：0x2C

复位：0x0000 FFFF

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **ARR[15:0]：**自动重装载的值（Auto-reload value）

ARR 为要装载到实际自动重载寄存器的值。

有关 ARR 更新和行为的更多详细信息，请参阅前序章节：时基单元。

当自动重载值为空时，计数器不工作。

该寄存器存储的是 ARR[15:0]中自动重装载值。

23 实时时钟（RTC）

23.1 RTC 简介

实时时钟是一个独立的定时器。RTC 模块拥有一组连续计数的计数器，在相应软件配置下，可提供时钟日历的功能。修改计数器的值可以重新设置系统当前的时间和日期。

RTC 模块和时钟配置系统（RCC_BDCR 寄存器）不在 V_{CORE} 域内，即在系统复位或从待机模式唤醒后，RTC 的设置和时间维持不变。

系统复位后，对 RTC 的访问被禁止。执行以下操作将使能对 RTC 的访问：

- 设置寄存器 RCC_APB1ENR 的 PWREN，使能电源。
- 设置寄存器 PWR_CR1 的 DBP 位，使能对 RTC 的访问。

23.2 主要特性

- 可编程的预分频系数：分频系数最高为 2^{20}。
- 32 位的可编程计数器，可用于较长时间段的测量。
- 2 个分离的时钟：用于 APB1 接口的 PCLK1 和 RTC 时钟（RTC 时钟的频率必须小于 PCLK1 时钟频率的四分之一以上）。
- 可以选择以下两种 RTC 的时钟源：
  - HSE 时钟除以 128；
  - LSI 振荡器时钟
- 2 个独立的复位类型：
  - APB1 接口由系统复位；
  - RTC 核心（预分频器、闹钟、计数器和分频器）只能由 RTC 域（RCC_BDCR）复位
- 3 个专门的可屏蔽中断：
  - 闹钟中断，用来产生一个软件可编程的闹钟中断。
  - 秒中断，用来产生一个可编程的周期性中断信号（最长可达 1 秒）。
  - 溢出中断，指示内部可编程计数器溢出并回转为 0 的状态。

23.3 功能描述

23.3.1 概述

RTC 由两个主要部分组成（参见下图）。第一部分（APB1 接口）用来和 APB1 总线相连。此单元还包含一组 16 位寄存器，可通过 APB1 总线对其进行读写操作。APB1 接口由 APB1 总线时钟驱动，用来与 APB1 总线接口。

另一部分（RTC 核心）由一组可编程计数器组成，分成两个主要模块。第一个模块是 RTC 的预分频模块，它可编程产生最长为 1 秒的 RTC 时间基准 TR_CLK。RTC 的预分频模块包含了一个 20 位的可编程分频器（RTC 预分频器）。如果在 RTC_CR 寄存器中设置了相应的允许位，则在每个 TR_CLK 周期中 RTC 产生一个中断（秒中断）。第二个模块是一个 32 位的可编程计数器，可被初始化为当前的系统时间。系统时间按 TR_CLK 周期累加并与存储在 RTC_ALR 寄存器中的可编程时间相比较，如果 RTC_CR 控制寄存器中设置了相应允许位，比较匹配时将产生一个闹钟中断。

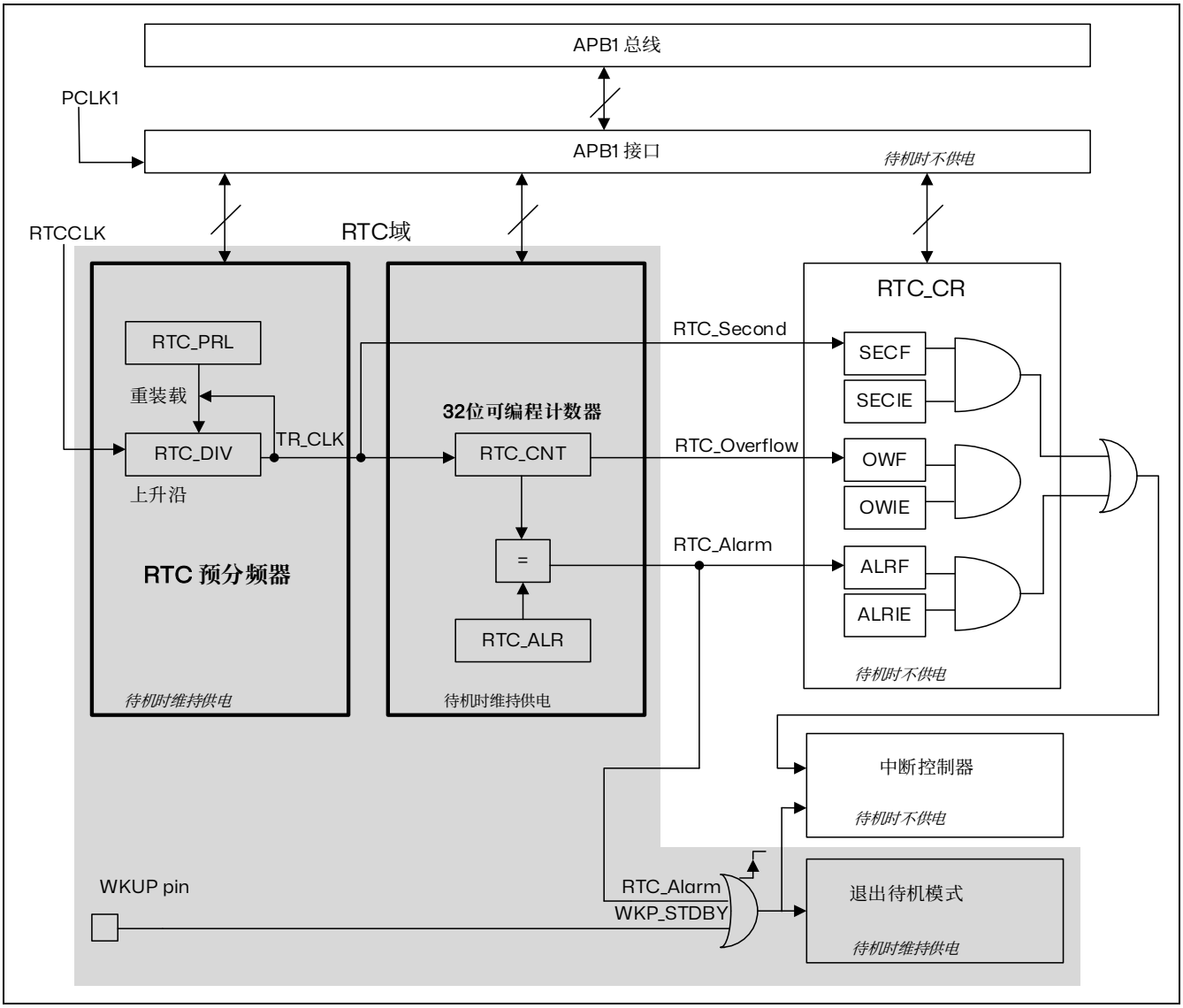


图 23.1 简化的 RTC 框图

23.3.2 复位过程

除了 RTC_PRL、RTC_ALR、RTC_CNT 和 RTC_DIV 寄存器外，所有的系统寄存器都由系统复位或电源复位进行异步复位。

RTC_PRL、RTC_ALR、RTC_CNT 和 RTC_DIV 寄存器仅能通过 RTC 域复位信号 BDRST 复位。

23.3.3 读 RTC 寄存器

RTC 核完全独立于 RTC APB1 接口。

软件通过 APB1 接口访问 RTC 的预分频值、计数器值和闹钟值，软件需要连续两次读取才能获取 RTC 的预分频值、计数器值和闹钟值。但是，相关的可读寄存器只在与 RTC APB1 时钟进行重新同步的 RTC 时钟的上升沿被更新。RTC 标志也是如此的。

这意味着，如果 APB1 接口曾经被关闭，而读操作又是在刚刚重新开启 APB1 之后，则在第一次的内部寄存器更新之前，从 APB1 上读出的 RTC 寄存器数值可能被破坏了（通常读到 0）。下述几种情况下能够发生这种情形：

- 发生系统复位或电源复位
- 系统刚从待机模式唤醒。
- 系统刚从停机模式唤醒。

所有以上情况中，APB1 接口被禁止时（复位、无时钟或断电）RTC 核仍保持运行状态。

因此，若在读取 RTC 寄存器时，RTC 的 APB1 接口曾经处于禁止状态，则软件首先必须等待 RTC_CRL 寄存器中的 RSF 位（寄存器同步标志）被硬件置“1”，再软件连续两次读取 RTC 寄存器以获取 RTC 的预分频值、计数器值和闹钟值。

注：RTC 的 APB1 接口不受低功耗模式的影响。

23.3.4 配置 RTC 寄存器

必须设置 RTC_CRL 寄存器中的 CNF 位，使 RTC 进入配置模式后，才能写入 RTC_PRL、RTC_CNT、RTC_ALR 寄存器。

另外，对 RTC 任何寄存器的写操作，都必须在前一次写操作结束后进行。可以通过查询 RTC_CR 寄存器中的 RTOFF 状态位，判断 RTC 寄存器是否处于更新中。仅当 RTOFF 状态位是“1”时，才可以写入 RTC 寄存器。

配置过程：

1. 查询 RTOFF 位，直到 RTOFF 的值变为“1”
2. 置 CNF 值为 1，进入配置模式
3. 对一个或多个 RTC 寄存器进行写操作
4. 清除 CNF 标志位，退出配置模式
5. 查询 RTOFF，直至 RTOFF 位变为“1”以确认写操作已经完成。

仅当 CNF 标志位被清除时，写操作才能进行，这个过程至少需要 3 个 RTCCLK 周期。

23.3.5 RTC 标志的设置

在每一个 RTC 核心的时钟周期中，更改 RTC 计数器之前设置 RTC 秒标志（SECF）。

在计数器到达 0x0000 之前的最后一个 RTC 时钟周期中，设置 RTC 溢出标志（OWF）。

在计数器的值到达闹钟寄存器的值加 1（RTC_ALR+1）之前的 RTC 时钟周期中，设置 RTC_Alarm 和 RTC 闹钟标志（ALRF）。对 RTC 闹钟的写操作必须使用下述过程之一与 RTC 秒标志同步：

- 使用 RTC 闹钟中断，并在中断处理程序中修改 RTC 闹钟和/或 RTC 计数器。
- 等待 RTC 控制寄存器中的 SECF 位被设置，再更改 RTC 闹钟和/或 RTC 计数器。

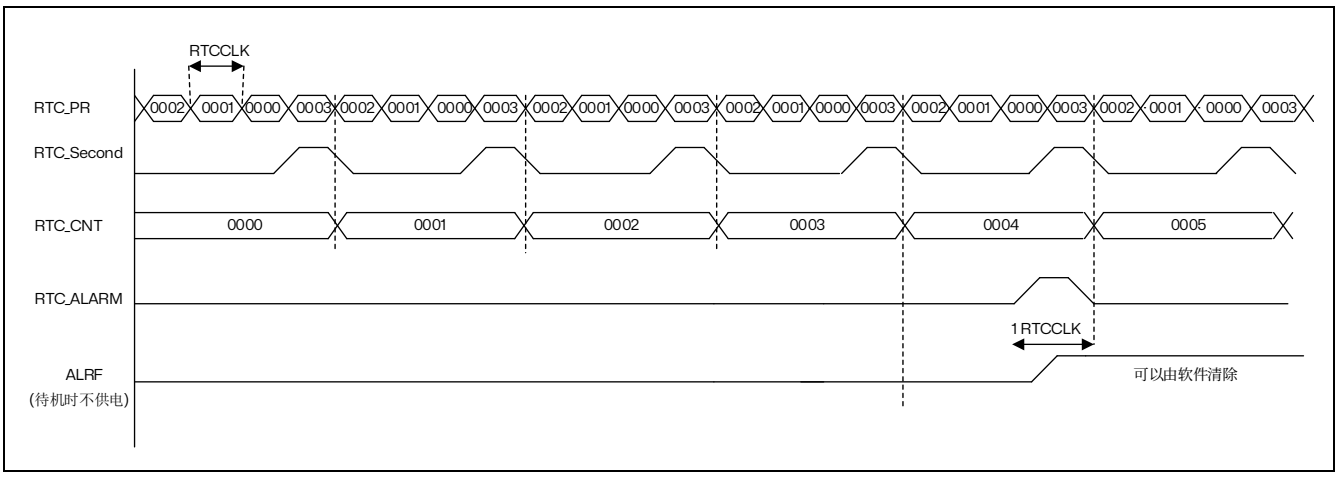


图 23.2 RTC 秒和闹钟波形图示例，PR=0003，ALARM=00004

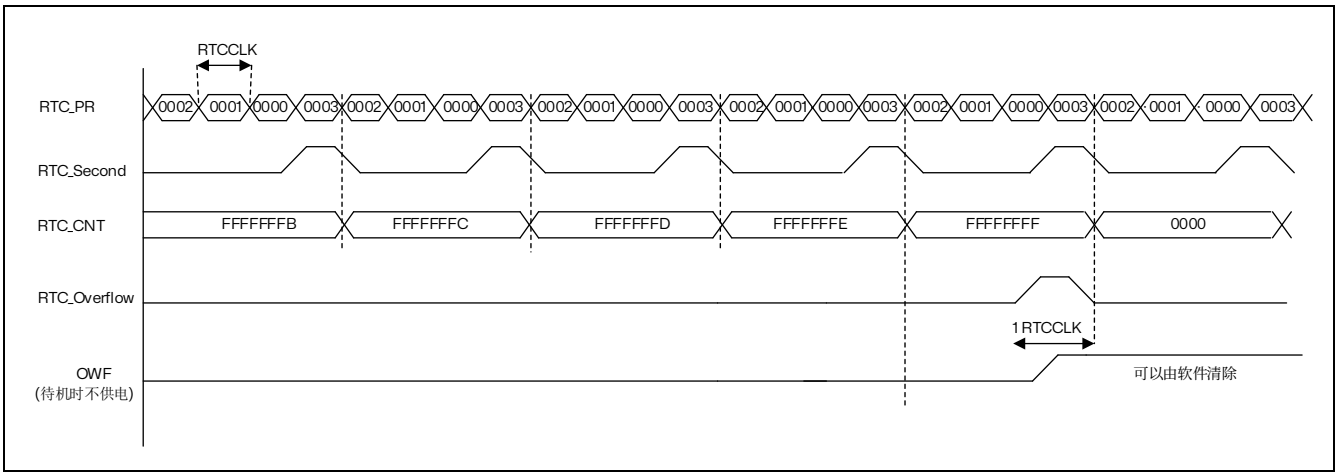


图 23.3 RTC 溢出波形图示例，PR=0003

23.4 RTC 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。
外设寄存器必须按半字（16 位）或字（32 位）进行访问。

23.4.1 RTC 控制寄存器高位（RTC_CRH）

地址偏移量：0x00
复位值：0x0000

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

Bit 15:3 保留，必须保持为复位值。

Bit 2 **OWIE**：允许溢出中断位（Overflow interrupt enable）
0：屏蔽（不允许）溢出中断
1：允许溢出中断

Bit 1 **ALRIE**：允许闹钟中断（Alarm interrupt enable）
0：屏蔽（不允许）闹钟中断
1：允许闹钟中断

Bit 0 **SECIE**：允许秒中断（Second interrupt enable）
0：屏蔽（不允许）秒中断
1：允许秒中断

这些位用来屏蔽中断请求。

注：系统复位后所有的中断被屏蔽，因此可通过写 RTC *寄存器来确保在初始化后没有挂起的中断请求。当外设正在完成前一次写操作时（标志位* RTOFF=0*），不能对* RTC_CRH *寄存器进行写操作。*

RTC 功能由这个控制寄存器控制。一些位的写操作必须经过一个特殊的配置过程来完成。

23.4.2 RTC 控制寄存器低位（RTC_CRL）

偏移地址：0x04
复位值：0x0020

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										RTOFF	CNF	RSF	OWF	ALRF	SECF
										r	rw	rc_w0	rc_w0	rc_w0	rc_w0

Bit 15:6 保留，必须保持为复位值。

Bit 5 **RTOFF**：RTC 操作关闭（RTC operation OFF）
RTC 模块利用这位来指示对其寄存器进行的最后一次操作的状态，指示操作是否完成。若此位为“0”，则表示无法对任何的 RTC 寄存器进行写操作。此位为只读位。
0：上一次对 RTC 寄存器的写操作仍在进行；
1：上一次对 RTC 寄存器的写操作已经完成。

Bit 4 **CNF**：配置标志（Configuration flag）
此位必须由软件置“1”以进入配置模式，从而允许向 RTC_CNT、RTC_ALR 或 RTC_PRL

寄存器写入数据。只有当此位在被置“1”并重新由软件清“0”后，才会执行写操作。
0：退出配置模式（开始更新 RTC 寄存器）；
1：进入配置模式。

Bit 3 **RSF**：寄存器同步标志（Registers synchronized flag）

每当 RTC_CNT 寄存器和 RTC_DIV 寄存器由软件更新或清“0”时，此位由硬件置“1”。在 APB1 复位后，或 APB1 时钟停止后，此位必须由软件清“0”。要进行任何的读操作之前，用户程序必须等待这位被硬件置“1”，以确保 RTC_CNT、RTC_ALR 或 RTC_PRL 已经被同步。
0：寄存器尚未被同步；
1：寄存器已经被同步。

Bit 2 **OWF**：溢出标志（Overflow flag）

当 32 位可编程计数器溢出时，此位由硬件置“1”。如果 RTC_CRH 寄存器中 OWIE=1，则产生中断。此位只能由软件清“0”。对此位写“1”是无效的。
0：无溢出；
1：32 位可编程计数器溢出。

Bit 1 **ALRF**：闹钟标志（Alarm flag）

当 32 位可编程计数器达到 RTC_ALR 寄存器所设置的预定值，此位由硬件置“1”。如果 RTC_CRH 寄存器中 ALRIE=1，则产生中断。此位只能由软件清“0”。对此位写“1”是无效的。
0：无闹钟；
1：有闹钟。

Bit 0 **SECF**：秒标志（Second flag）

当 32 位可编程预分频器溢出时，此位由硬件置“1”同时 RTC 计数器加 1。因此，此标志为分辨率可编程的 RTC 计数器提供一个周期性的信号（通常为 1 秒）。如果 RTC_CRH 寄存器中 SECIE=1，则产生中断。此位只能由软件清除。对此位写“1”是无效的。
0：秒标志条件不成立；
1：秒标志条件成立。

RTC 的功能由这个控制寄存器控制。当前一个写操作还未完成时，不能写 RTC_CR 寄存器。

*注：1. 任何标志位都将保持挂起状态，直到适当的*RTC_CR *请求位被软件复位，表示所请求的中断已经被接受。*

*2. 在复位时禁止所有中断，无挂起的中断请求，可以对*RTC *寄存器进行写操作。*

*3. 当*APB1 *时钟不运行时，*OWF、ALRF、SECF *和*RSF *位不被更新。*

4. OWF、ALRF、SECF *和*RSF *位只能由硬件置位，由软件来清零。*

*5 若*ALRF=1 *且*ALRIE=1，*则允许产生*RTC *全局中断。如果在*EXTI *控制器中允许产生*EXTI *线*17 *中断，则允许产生*RTC *全局中断和*RTC *闹钟中断。*

*6 若*ALRF=1，*如果在*EXTI *控制器中设置了*EXTI *线*17 *的中断模式，则允许产生*RTC *闹钟中断；如果在*EXTI *控制器中设置了*EXTI *线*17 *的事件模式，则这条线上会产生一个脉冲（不会产生*RTC *闹钟中断）。*

23.4.3 RTC 预分频装载寄存器（RTC_PRLH/RTC_PRLL）

预分频装载寄存器用来保存 RTC 预分频器的周期计数值。它们受 RTC_CR 寄存器的 RTOFF 位保护，仅当 RTOFF 值为“1”时允许进行写操作。

预分频装载寄存器需要连续两次读操作来获取 RTC 预分频器的周期计数值。

RTC 预分频装载寄存器高位（RTC_PRLH）

偏移地址：0x08

只写

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PRL[19:16]			
												w	w	w	w

Bit 15:4 保留，必须保持为复位值。

Bit 3:0 **PRL[19:16]：** RTC 预分频装载值高位（RTC prescaler reload value high）

根据以下公式，这些位用来定义计数器的时钟频率：

$f_{TR_CLK} = f_{RTCCLK}/（PRL[19:0]+1）$

注：不推荐使用 0 值，否则无法正确的产生 RTC 中断和标志位。

RTC 预分频装载寄存器低位（RTC_PRLL）

偏移地址：0x0C

只写

复位值：0x8000

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

Bit 15:0 **PRL[15:0]：** RTC 预分频装载值低位

根据以下公式，这些位用来定义计数器的时钟频率：

$f_{TR_CLK} = f_{RTCCLK}/（PRL[19:0]+1）$

注：如果输入时钟频率是 32.768kHz（f_{RTCCLK}），这个寄存器中写入 7FFFh 可获得周期为 1 秒钟的信号。

23.4.4 RTC 预分频器余数寄存器（RTC_DIVH / RTC_DIVL）

在 TR_CLK 的每个周期里，RTC 预分频器中计数器的值都会被重新设置为 RTC_PRL 寄存器的值。用户可通过读取 RTC_DIV 寄存器，以获得预分频计数器的当前值，而不停止分频计数器的工作，从而获得精确的时间测量。此寄存器是只读寄存器，其值在 RTC_PRL 或 RTC_CNT 寄存器中的值发生改变后，由硬件重新装载。

预分频器余数寄存器需要连续两次读操作来获取预分频计数器的当前值。

RTC 预分频器余数寄存器高位（RTC_DIVH）

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
Reserved												RTC_DIV[19:16]			
												r	r	r	r

Bit 15:4 保留，必须保持为复位值。

Bit 3:0 **RTC_DIV[19:16]：**RTC 时钟分频器余数高位（RTC clock divider high）

RTC 预分频器余数寄存器低位（RTC_DIVL）

偏移地址：0x14

复位值：0x8000

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

Bit 15:0 **RTC_DIV[15:0]：**RTC 时钟器余数低位（RTC clock divider low）

23.4.5 RTC 计数器寄存器（RTC_CNTH/RTC_CNTL）

RTC 核有一个 32 位可编程的计数器，可通过两个 16 位的寄存器访问。计数器以预分频器产生的 TR_CLK 时间基准为参考进行计数。RTC_CNT 寄存器用来存放计数器的计数值。他们受 RTC_CR 的位 RTOFF 写保护，仅当 RTOFF 值为“1”时，允许写操作。在高或低寄存器（RTC_CNTH 或 RTC_CNTL）上的写操作，能够直接装载到相应的可编程计数器，并且重新装载 RTC 预分频器。当进行读操作时，直接返回计数器内的计数值（系统时间）。

计数器寄存器需要连续两次读操作来获取计数器的计数值。

RTC 计数器寄存器高位（RTC_CNTH）

偏移地址：0x18

复位值：0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_CNT[31:16]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:0 **RTC_CNT[31:16]：**RTC 计数器高位（RTC counter high）

可通过读 RTC_CNTH 寄存器来获得 RTC 计数器当前值的高位部分。要对此寄存器进行写操作前，必须先进入配置模式。

RTC 计数器寄存器低位（RTC_CNTL）

偏移地址：0x1C
复位值：0x0000

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

Bit 15:0 **RTC_CNT[15:0]**：RTC 计数器低位。

可通过读 RTC_CNTL 寄存器来获得 RTC 计数器当前值的低位部分。要对此寄存器进行写操作，必须先进入配置模式。

23.4.6 RTC 闹钟寄存器（RTC_ALRH/RTC_ALRL）

当可编程计数器的值与 RTC_ALR 中的 32 位值相等时，即触发一个闹钟事件，并且产生 RTC 闹钟中断。此寄存器受 RTC_CR 寄存器里的 RTOFF 位写保护，仅当 RTOFF 值为“1”时，允许写操作。

闹钟寄存器需要连续两次读操作来获取 RTC 闹钟时间。

RTC 闹钟寄存器高位（RTC_ALRH）

偏移地址：0x20
只写
复位值：0xFFFF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTC_ALR[31:16]															
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit 15:0 **RTC_ALR[31:16]**：RTC 闹钟值高位（RTC alarm high）

此寄存器用来保存由软件写入的闹钟时间的高位部分。要对此寄存器进行写操作，必须先进入配置模式。

RTC 闹钟寄存器低位（RTC_ALRL）

偏移地址：0x24
只写
复位值：0xFFFF

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

Bit 15:0 **RTC_ALR[15:0]**：RTC 闹钟值低位（RTC alarm low）

此寄存器用来保存由软件写入的闹钟时间的低位部分。要对此寄存器进行写操作，必须先进入配置模式。

24 参考电压缓冲器（VREFBUF）

24.1 VREFBUF 简介

RX32H710 器件内置有参考电压缓冲器，既可用作 ADC 和 DAC 的参考电压，也可通过 VREF 引脚用作外部元件的参考电压。当 VREF 引脚与 VDD 引脚短接时，参考电压缓冲器将无法使用，并且必须保持禁用状态。

24.2 VREFBUF 功能描述

内部参考电压缓冲器支持三种电压[a]，这些电压通过 VREFBUF_CSR 寄存器中的 VRS 位进行配置：

- VRS = 00：大约为 2.0 V。
- VRS = 01：大约为 2.5 V。
- VRS = 10：大约为 2.8 V。

内部参考电压配置为四种不同模式，具体取决于 ENVR 和 HIZ 位的配置。下表列出了这些模式：

表 24.1 VREF 缓冲器模式

ENVR	HIZ	VREF 缓冲器配置
0	0	VREFBUF 缓冲器关闭 – V_{REF} 引脚下拉到 V_{SS}
0	1	外部参考电压模式（默认值）： – VREFBUF 缓冲器关闭 – V_{REF} 引脚输入模式
1	0	内部参考电压模式： – VREFBUF 缓冲器开启 – V_{REF} 引脚连接到 VREFBUF 缓冲器输出
1	1	保持模式： – VREF 引脚悬空。借助外部电容来保持电压 – 禁止 VRR 检测，VRR 位保持最后一个状态

通过将 VREFBUF_CSR 寄存器中的 ENVR 位置 1 并将 HIZ 位清零使能 VREFBUF 后，用户必须等待至 VRR 位置 1，指示参考电压输出已达到预期值。

a. 最小 V_{DD} 电压取决于 VRS 设置，请参阅产品数据手册。

24.3 VREFBUF 校准

VREFBUF 的输出电压由睿兴在出厂时进行校准。在复位时，以及每次更改 VRS 设置时，校准数据会自动加载到 TRIM 寄存器中。

用户还可以选择通过直接更改 TRIM 寄存器的位来微调输出电压。在这种情况下，直到设备复位，VRS 设置将不再对 TRIM 寄存器产生影响。

24.4 VREFBUF 寄存器描述

24.4.1 VREFBUF 控制和状态寄存器（VREFBUF_CSR）

地址偏移：0x00

复位值：0x0000 0012

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

Bit 31:7 保留，必须保持为复位值。

Bit 6 **VCM：** VCM 偏置电压（V_{REFBUF}/2）缓冲器使能

0：关闭 VCM 偏置电压（V_{REFBUF}/2）缓冲器

1：使能 VCM 偏置电压（V_{REFBUF}/2）缓冲器

注：当 OPAMP *偏置电压选择为* V_{REFBUF}/2，*需要置位* VCM *位来使能* V_{REFBUF}/2 *缓冲器。详见章节：*OPAMPx *控制寄存器*2（OPAMPx_CR2）。

Bit 5:4 **VRS[1:0]：** 参考电压调节

这些位选择由电压参考缓冲器生成的值。

00：参考电压设为 2.0 V

01：参考电压设为 2.5 V

10：参考电压设为 2.8 V

11：保留

注：仅当 VREFBUF *被禁用*（ENVR = 0）*时，软件才能编程该位域。*

Bit 3 **VRR：** 参考电压缓冲器就绪

0：参考电压缓冲器输出未就绪。

1：参考电压缓冲器输出已达到请求值。

Bit 2 保留，必须保持为复位值。

Bit 1 **HIZ：**高阻抗模式

此位控制模拟开关是否连接到 V_{REF+}引脚。

0：此位控制模拟开关是否连接到 V_{REF+}引脚。

1：V_{REF+}引脚为高阻态。

关于 ENVR 位配置及模式说明，请参见上述表格：VREF 缓冲器模式。

Bit 0 **ENVR：**参考电压缓冲器模式使能

此位用于使能参考电压缓冲器模式。

0：禁止内部参考电压模式（外部参考电压模式）。

1：使能内部参考电压模式（参考缓冲器使能或保持模式）。

24.4.2 VREFBUF 校准控制寄存器（VREFBUF_CCR）

地址偏移：0x04

复位值：0x0000 000X

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

Bit 31:4 保留，必须保持为复位值。

Bit 3:0 **TRIM[3:0]：**微调代码

TRIM 代码是一个 4 位无符号数据（最小值 0000，最大值 1111），它根据以下描述的机制进行设置和更新。

重置：

TRIM[3:0]在生产测试期间会自动初始化为存储在闪存中的 VRS=0 微调值。

VRS 更改：

TRIM[3:0]在生产测试期间会自动初始化为存储在闪存中的微调值（对应于 VRS 设置）。

TRIM[3:0]写入：

用户可以将 TRIM[3:0]修改为任意值。这将永久禁用 VRS 对微调值的控制（直到设备重置为止）。

注：如果用户应用程序执行微调，微调代码必须从 000 *开始，按升序到* 111。

25 独立看门狗（IWDG）

25.1 简介

RX32H710 设备内置一个看门狗，提供了更高的安全性、时间的精确性和使用的灵活性。此独立看门狗外设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。

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。

25.2 IWDG 主要特性

- 自由运行的递减计数器
- 时钟由独立的 RC 振荡器提供（可在停止和待机模式下工作）
- 看门狗被激活后，则在计数器计数至 0x000 时产生复位

25.3 IWDG 功能描述

下图为独立看门狗模块的功能框图。

在键寄存器（IWDG_KR）中写入 0xCCCC，开始使能独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号（IWDG_RESET）。

无论何时，只要在键寄存器 IWDG_KR 中写入 0xAAAA，IWDG_RLR 中的值就会被重新加载到计数器，从而避免产生看门狗复位。

25.3.1 硬件看门狗

如果用户在选项字节中使能了“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。

25.3.2 寄存器访问保护

IWDG_PR 和 IWDG_RLR 寄存器具有写保护功能。要修改这两个寄存器的值，必须先向 IWDG_KR 寄存器中写入 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。

25.3.3 调试模式

当微控制器进入调试模式时（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。

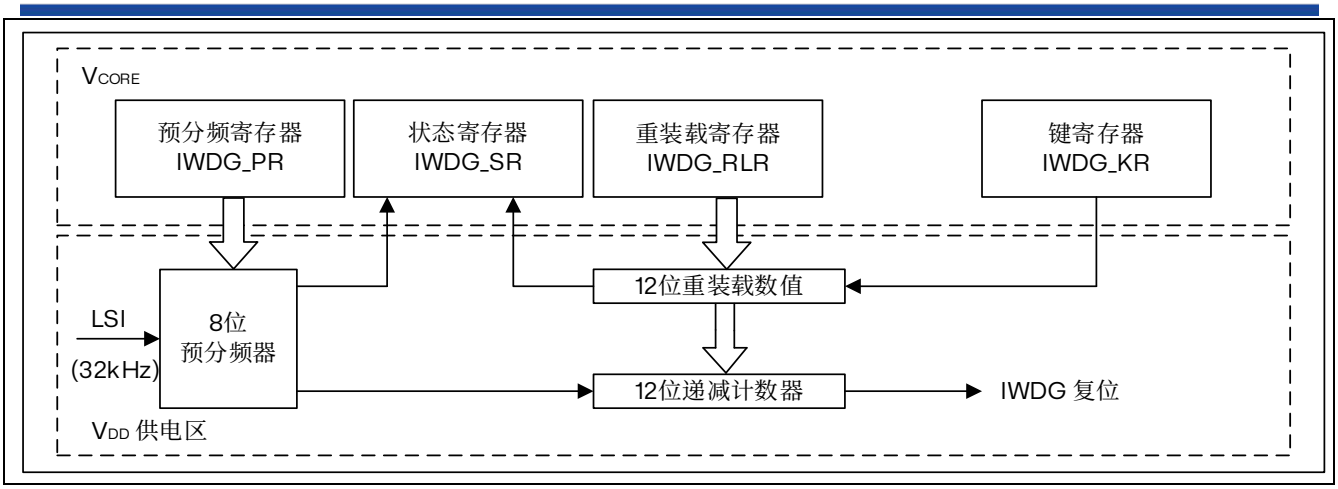


图 25.1 独立看门狗框图

注：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 25.1 看门狗超时时间（32kHz 的输入时钟（LSI））(1)

预分频系数	PR[2:0]位	最短时间（ms） RL[11:0] = 0x000	最短时间（ms） RL[11:0] = 0xFFF
/4	0	0.125	512
/8	1	0.25	1024
/16	2	0.5	2048
/32	3	1	4096
/64	4	2	8192
/128	5	4	16384
/256	（6 或 7）	8	32768

1. 这些时序参数是基于 32kHz 时钟给出的，但微控制器内部 RC 频率可能会有所变化。有关最大值和最小值，请参考器件数据手册中的 LSI 振荡器特性表。

25.4 IWDG 寄存器描述

有关寄存器说明中使用的缩写，请参见第 2 章节：文档约定中的第 1 节 。
外设寄存器必须按半字（16 位）或字（32 位）进行访问。

25.4.1 键寄存器（IWDG_KR）

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
Reserved															
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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **KEY[15:0]：**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_RLR 寄存器。
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）。

25.4.2 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
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													PR[2:0]		
													rw	rw	rw

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 PR[2:0]：预分频系数（Prescaler divider）
这些位具有写保护设置。通过设置这些位来选择计数器时钟的预分频系数。要改变预分频系数，IWDG_SR 寄存器的 PVU 位必须为 0。

000：预分频系数=4　　100：预分频系数=64
001：预分频系数=8　　101：预分频系数=128
010：预分频系数=16　　110：预分频系数=256
011：预分频系数=32　　111：预分频系数=256

注：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_SR 寄存器的 PVU 位为 0 时，读出的值才有效。

25.4.3 重装载寄存器（IWDG_RLR）

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
Reserved															

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

Bit 31:12 保留，必须保持为复位值。

Bit 11:0 RL[11:0]：看门狗计数器重装载值（Watchdog counter reload value）

这些位具有写保护功能。用于定义看门狗计数器的重装载值，每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中。随后计数器从这个值开始递减计数。看门狗超时周期可通过此重装载值和时钟预分频值来计算。

只有当 IWDG_SR 寄存器中的 RVU 位为 0 时，才能对此寄存器进行修改。

注：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_SR 寄存器的 RVU 位为 0 时，读出的值才有效。

25.4.4 状态寄存器（IWDG_SR）

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
Reserved															

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

Bit 31:2 保留，必须保持为复位值。

Bit 1 RVU：看门狗计数器重装载值更新（Watchdog counter reload value update）

此位由硬件置“1”用来指示重装载值的更新正在进行中。当在 VDD 域中的重装载更新结束后，此位由硬件清“0”（最多需 5 个 32kHz 的 RC 周期）。重装载值只有在 RVU 位被清“0”后才可更新。

Bit 0 PVU：看门狗预分频值更新（Watchdog prescaler value update）

此位由硬件置“1”用来指示预分频值的更新正在进行中。当在 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 32kHz 的 RC 周期）。预分频值只有在 PVU 位被清“0”后才可更新。

注：如果在应用程序中使用了多个重装载值或预分频值，则必须在 RVU 位被清除后才能重新改变预装载值，在 PVU 位被清除后才能重新改变预分频值。然而，在预分频和/或重装值更新后，不必等待 RVU 或 PVU 复位，可继续执行下面的代码。（即是在低功耗模式下，此写操作仍会被继续执行完成。）

26 FD 控制器局域网络（FDCAN）

26.1 简介

控制器局域网（CAN）子系统（见下图）由一个 CAN 模块、共享消息 RAM 和配置块组成。各部分的基地址请参考存储器映射表。

FDCAN 模块符合标准包括 ISO 11898-1:2015（CAN 协议规范版本 2.0，包括 A 部分和 B 部分）和 CAN FD 协议规范版本 1.0。

FDCAN 实例配备 0.8KB 的消息 RAM，用于实现以下功能：过滤器，接收 FIFO，发送事件 FIFO 以及发送 FIFO。

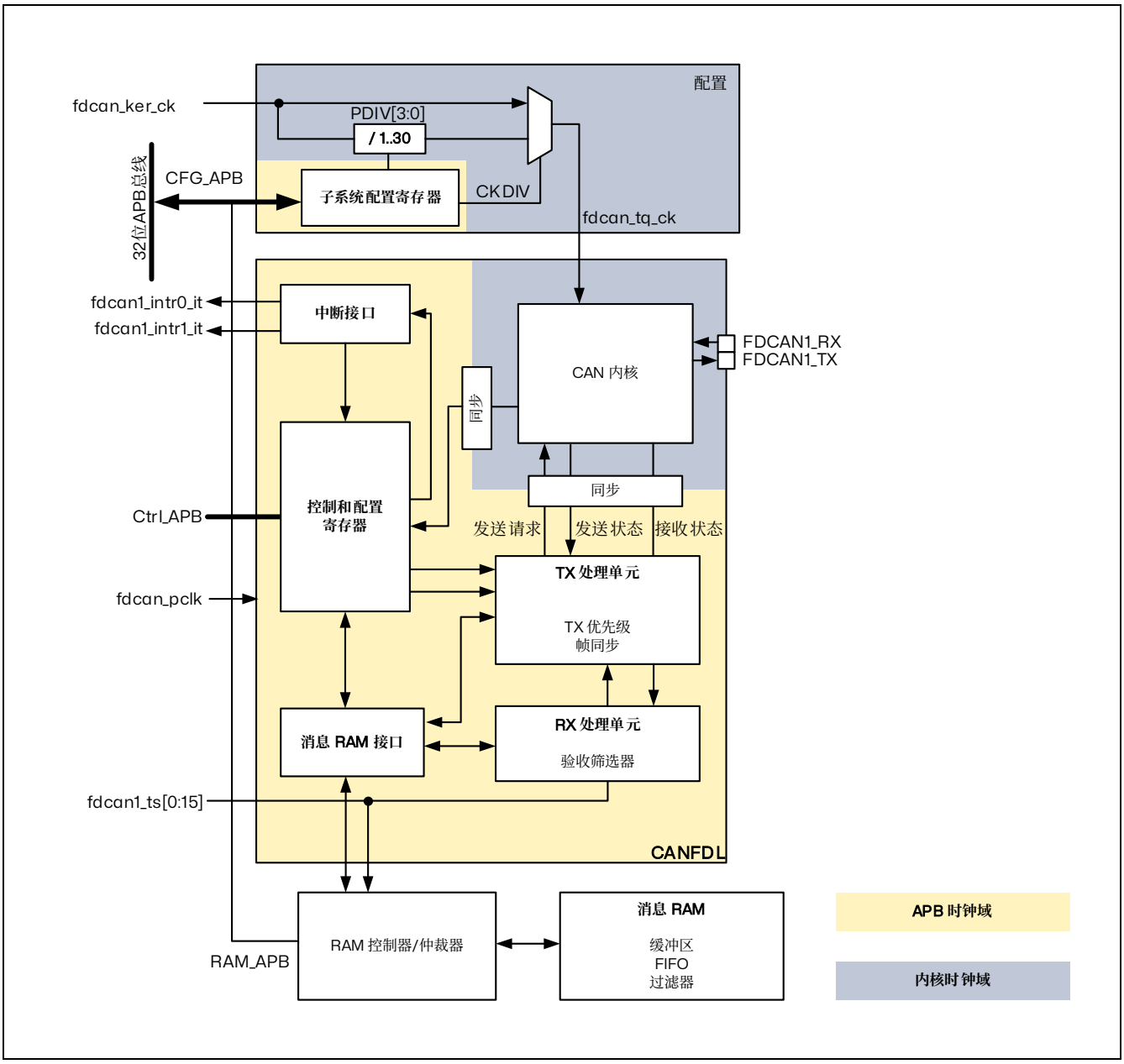


图 26.1 CAN 子系统

26.2 FDCAN 主要特性

- 符合 CAN 协议第 2.0 版 A、B 部分和 ISO 11898-1: 2015
- CAN FD 最多支持收发 64 个数据字节
- CAN 错误记录
- 支持 AUTOSAR 和 J1939
- 改进了接收过滤
- 两个接收 FIFO，每个 FIFO 包含三个有效载荷（每个有效载荷最大 64 字节）
- 接收到高优先级消息时单独发出信号指示
- 可配置的发送 FIFO/队列，包含三个有效载荷（每个有效载荷最大 64 字节）
- 发送事件 FIFO
- 可编程环回测试模式
- 可屏蔽模块中断
- 两个时钟域：APB 总线接口时钟和 CAN 内核时钟
- 支持掉电

26.3 FDCAN 功能描述

26.3.1 FDCAN 框图

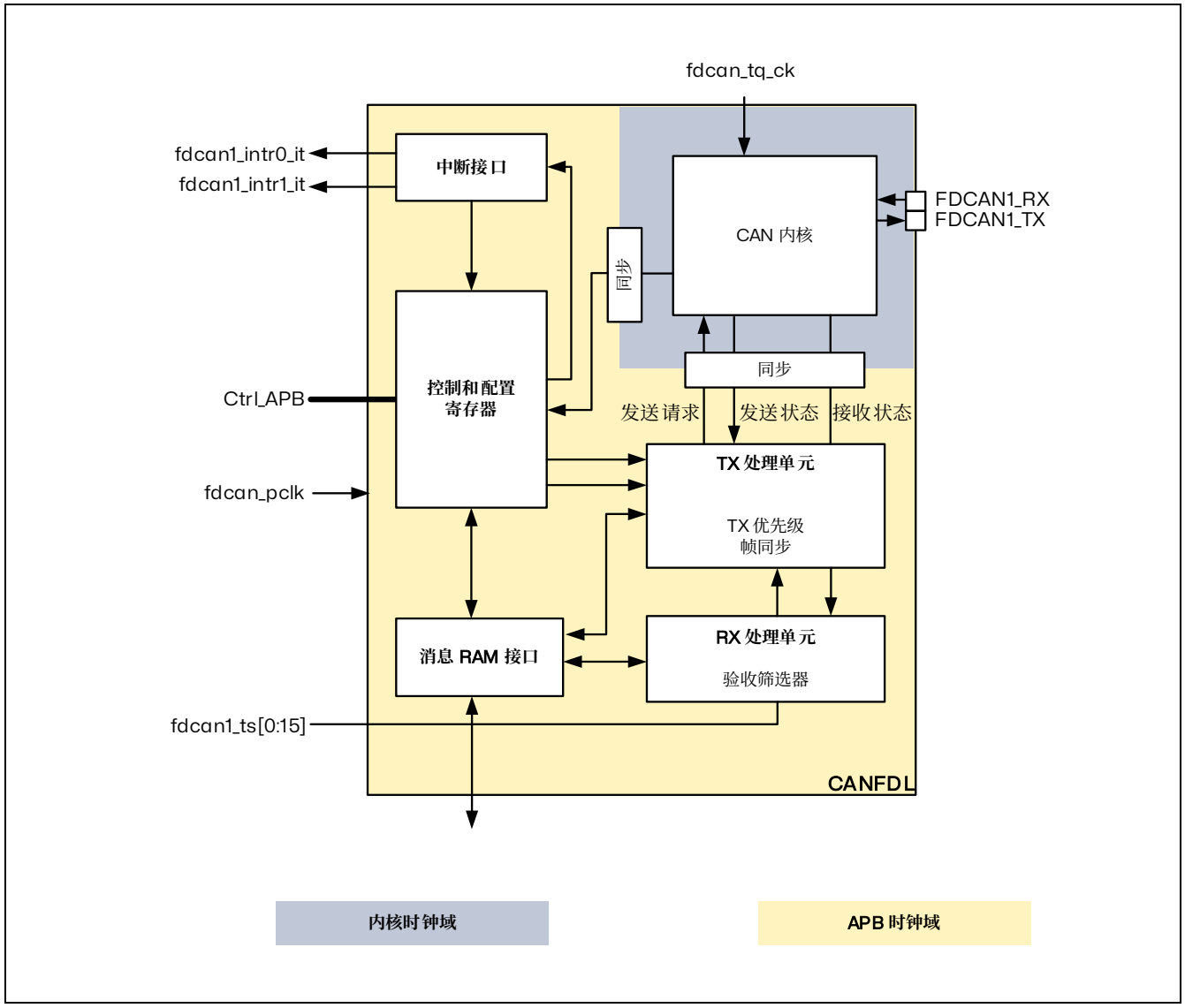


图 26.2 FDCAN 框图

双中断线

FDCAN 外设提供 fdcan_intr0_it 和 fdcan_intr1_it 两条中断线。
通过对 FDCAN_ILE 寄存器中的 EINT0 和 EINT1 位进行编程，可分别使能或禁止这两条中断线。

CAN 内核

CAN 内核包含协议控制器和接收/发送移位寄存器。它可处理所有 ISO 11898-1: 2015 协议功能，并支持 11 位和 29 位标识符。

同步

同步模块将来自 APB 时钟域的信号同步到 CAN 内核时钟域，反之亦然。

发送处理单元

该处理单元控制从消息 RAM 到 CAN 内核的消息传输。最多可配置 3 个发送缓冲区进行发送。发送缓冲区可用作发送 FIFO 或发送队列。发送事件 FIFO 会将发送时间戳与相应的消息 ID 存储在一起，另外还支持取消发送。

接收处理单元

该处理单元控制着从 CAN 内核接收的消息到外部消息 RAM 的传输。接收处理单元支持两个接收 FIFO，用于存储所有通过验收过滤的消息。每条消息均与其接收时间戳存储在一起。对于 11 位 ID，最多可定义 28 个过滤器；对于 29 位 ID，最多可定义 8 个过滤器。

APB 接口

APB 接口将 FDCAN 连接到 APB 总线，用于访问配置寄存器、控制器配置和 RAM。

消息 RAM 接口

通过 RAM 控制器/仲裁器将 FDCAN 访问连接到外部 1KB 消息 RAM。

26.3.2 FDCAN 引脚和内部信号

CAN 子系统的 I/O 信号和引脚分配分别详见下表 27.1 CAN 子系统 I/O 信号和表 27.2 CAN 子系统 I/O 引脚以及图 27.1 CAN 子系统。

表 26.1 CAN 子系统 I/O 信号

名称	类型	说明
fdcan_ker_ck	数字输入	CAN 子系统内核时钟输入
fdcan_pclk		CAN 子系统 APB 接口时钟输入
fdcan_tq_ck		CAN 子系统最小时间片时钟输入
fdcan1_intr0_it	数字输出	FDCAN 中断 0
fdcan1_intr1_it		FDCAN 中断 1
fdcan_ts[0:15]	-	外部时间戳向量
APB 接口	数字输入/输出	单一应用程序，支持多外设选择以进行配置、控制和 RAM 访问

表 26.2 CAN 子系统 I/O 引脚

名称	类型	说明
FDCAN1_RX	数字输入	FDCAN 接收引脚
FDCAN1_TX	数字输出	FDCAN 发送引脚

26.3.3 位定时

位定时逻辑监控串行总线线路，通过在起始位边沿同步并在后续边沿重新同步，执行采样和采样点调整。

如下图所示，其操作可通过将位时间分为三个段来简单解释：

- 同步段（SYNC_SEG）：预期在此时间段内发生位变化，固定长度为一个时间片（1 × tq）。
- 位段 1（BS1）：定义采样点位置。包括 CAN 标准中的 PROP_SEG 和 PHASE_SEG1。其持续时间可编程为 1 到 16 个时间片，但可自动延长以补偿网络中各节点频率差异导致的正相位漂移。
- 位段 2（BS2）：定义发送点位置。代表 CAN 标准中的 PHASE_SEG2，其持续时间可编程为 1 到 8 个时间片，但也可自动缩短以补偿负相位漂移。

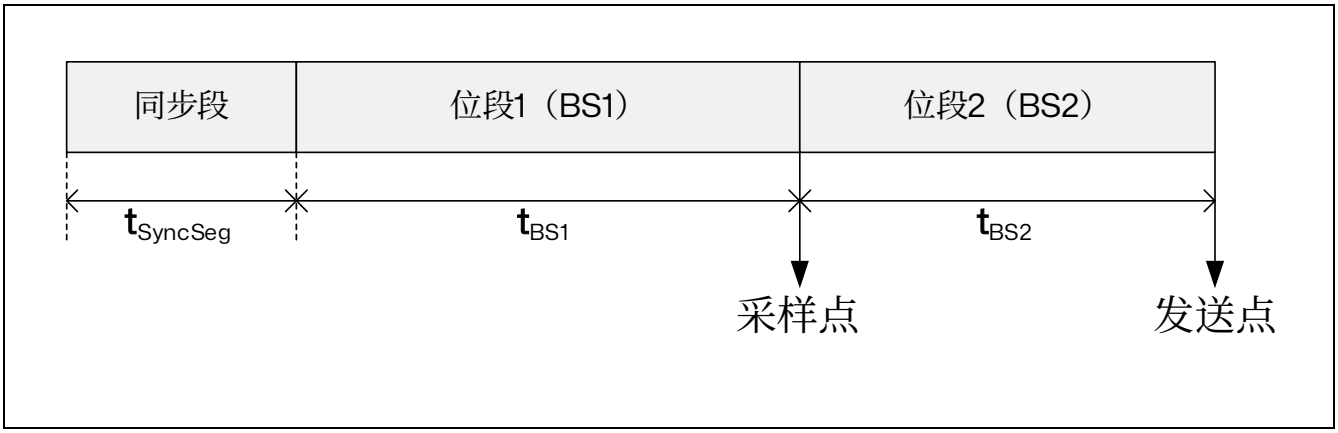


图 26.3 位定时

波特率是位时间的倒数（波特率 = 1/位时间），而位时间由三个分量组成（见上图）：

位时间 = $t_{SyncSeg} + t_{BS1} + t_{BS2}$

其中：

– 对于标称位时间

$t_q = (NBRP[8:0] + 1) \times t_{fdcan_tq_ck}$

$t_{SyncSeg} = 1 \times t_q$

$t_{BS1} = t_q \times (NTSEG1[7:0] + 1)$

$t_{BS2} = t_q \times (NTSEG2[6:0] + 1)$

其中，NBRP[8:0]、NTSEG1[7:0]和 NTSEG2[6:0]位域属于 FDCAN_NBTP 寄存器。

$t_{fdcan_tq_ck}$ 为 CAN 子系统最小时间片，由 CAN 时钟经 FDCAN_CKDIV 寄存器分频得到。

– 对于数据位时间

$t_q = (DBRP[4:0] + 1) \times t_{fdcan_tq_ck}$

$t_{SyncSeg} = 1 \times t_q$

$t_{BS1} = t_q \times (DTSEG1[4:0] + 1)$

$t_{BS2} = t_q \times (DTSEG2[3:0] + 1)$

其中，DBRP[4:0]、DTSEG1[4:0]和 DTSEG2[3:0]属于 FDCAN_DBTP 寄存器。

$t_{fdcan_tq_ck}$ 为 CAN 子系统最小时间片，由 CAN 时钟经 FDCAN_CKDIV 寄存器分频得到。

（重新）同步跳变宽度（SJW）定义了位段延长或缩短的上限，可编程为 1 到 4 个时间片。

有效边沿定义为位时间内从显性总线电平到隐性总线电平的首次跳变，前提是控制器本身未发送隐性位。

若在 BS1 而非 SYNC_SEG 中检测到有效边沿，BS1 将延长最多 SJW，从而延迟采样点。

反之，若在 BS2 而非 SYNC_SEG 中检测到有效边沿，BS2 将缩短最多 SJW，从而提前发送点。

为防止编程错误，位定时寄存器的配置仅在设备处于待机模式时才允许。FDCAN_DBTP 和 FDCAN_NBTP 寄存器（分别用于数据位定时和标称位定时）仅在 FDCAN_CCCR 寄存器的 CCE 和 INIT 位被置位时才可访问。

FDCAN 要求 CAN 时间片时钟始终小于或等于 APB 时钟（fdcan_tq_ck ≤ fdcan_pclk）。

注：有关 CAN 位定时和重新同步机制的详细描述，请参考 ISO 11898-1 标准。

26.3.4 工作模式

配置模式

可访问 IP 版本、硬件及输入时钟分频器配置。当时钟分频器设置为 0 时，直接使用主输入时钟。

软件初始化

要进行软件初始化，首先应通过软件或硬件复位、或者进入 Bus_Off 状态将 FDCAN_CCCR 寄存器中的 INIT 位置 1。当 FDCAN_CCCR 寄存器中的 INIT 位置 1 时，从 CAN 总线传入和传出的消息都将停止，并且 CAN 总线输出 FDCAN_TX 的状态为隐性（高）。错误管理逻辑（EML）的计数器保持不变。将 FDCAN_CCCR 中的 INIT 位置 1 不会更改任何配置寄存器。将 FDCAN_CCCR 中的 INIT 位清零会结束软件初始化。随后，位流处理单元（BSP）会等待出现 11 个连续隐性位（Bus_Idle）序列，以此将其自身与 CAN 总线上的数据传输进行同步，然后才能参与总线活动并开始消息传输。

仅当 FDCAN_CCCR 寄存器中的 INIT 位以及 FDCAN_CCCR 寄存器中的 CCE 位均置 1 时，才能访问 FDCAN 配置寄存器。

FDCAN_CCCR 寄存器中的 CCE 位只能在 FDCAN_CCCR 中的 INIT 位置 1 时才能置 1/清零。FDCAN_CCCR 寄存器中的 CCE 位会在 FDCAN_CCCR 中的 INIT 位清零时自动清零。

FDCAN_CCCR 寄存器中的 CCE 位置 1 后，以下寄存器会复位：

- FDCAN_HPMS：高优先级消息状态
- FDCAN_RXF0S：接收 FIFO 0 状态
- FDCAN_RXF1S：接收 FIFO 1 状态
- FDCAN_TXFQS：发送 FIFO/队列状态
- FDCAN_TXBRP：发送缓冲区请求挂起
- FDCAN_TXBTO：发送缓冲区发送已发生
- FDCAN_TXBCF：发送缓冲区取消完成
- FDCAN_TXEFS：发送事件 FIFO 状态

当 FDCAN_CCCR 的 CCE 位被置 1 时，超时计数器值（FDCAN_TOCV 寄存器的 TOC[15:0]位）被预设为 FDCAN_TOCC 寄存器的 TOP[15:0]配置的值。

此外，FDCAN_CCCR 中的 CCE 位置 1 时，发送处理单元和接收处理单元中的状态机会保持空闲状态。

仅当 FDCAN_CCCR 寄存器中的 CCE 位清零时，才能写入以下寄存器：

- FDCAN_TXBAR：发送缓冲区添加请求
- FDCAN_TXBCR：发送缓冲区取消请求

仅当 CCCR 寄存器中的 INIT 位和 CCE 位均已置 1 时，才能通过软件将 FDCAN_CCCR 中的 TEST 位和 MON 位置 1。这两个位可随时复位。仅当 FDCAN_CCCR 中的 INIT 位和 CCE 位均已置 1 时，才能将 FDCAN_CCCR 中的 DAR 位置 1/清零。

正常操作

FDCAN 在硬件复位后的默认工作模式为事件驱动型 CAN 通信，不支持时间触发（TT）操作模式。

FDCAN 完成初始化且 FDCAN_CCCR 寄存器中的 INIT 位清零后，FDCAN 会将其自身与 CAN 同步并准备好进行通信。

通过接收滤波后，接收到的包含消息 ID 和数据长度码 DLC 的消息会存储到专用的接收缓冲区中，或者存储到接收 FIFO 0 或接收 FIFO 1 中。

要发送消息，可初始化或更新发送缓冲区和/或发送 FIFO 或发送队列。不支持在接收到远程帧后自动发送。

CAN FD 操作

FDCAN 协议中有两种类型：

- 长帧模式（LFM）：CAN 帧的数据字段长度可超过 8 字节。
- 快速帧模式（FFM）：CAN 帧的控制字段、数据字段和 CRC 字段的发送比特率要高于帧开始和结束的发送比特率。

快速帧模式可与长帧模式结合使用。

具有 11 位标识符的 CAN 帧中之前保留的位和具有 29 位标识符的 CAN 帧中之前保留的第一个位现在可解码为 FDF 位。FDF 隐性表示 CAN FD 帧，而 FDF 显性则表示典型 CAN 帧。

在 CAN FD 帧中，FDF 之后的两个位 res 和 BRS 决定是否切换这一 CAN FD 帧中的比特率。CAN FD 比特率切换通过 res 显性和 BRS 隐性指示。res 隐性的编码保留，供将来协议扩展使用。如果 FDCAN 接收到的帧为 FDF 隐性和 res 隐性，则会通过将 FDCAN_PSR 寄存器的 PXE 位置 1 的方式发出协议异常事件。如果使能协议异常处理（FDCAN_CCCR 中的 PXHD = 0），下一采样点的工作状态会从接收器（FDCAN_PSR 中的 ACT[1:0] = 10）变为同步中（FDCAN_PSR 中的 ACT[1:0] = 00）。如果禁止协议异常处理（FDCAN_CCCR 中的 PXHD = 1），FDCAN 会将隐性 res 位当作格式错误处理，并将以错误帧进行响应。

CAN FD 操作通过编程 FDCAN_CCCR 寄存器的 FDOE 位来使能。

如果 FDOE = 1，则会使能 CAN FD 帧的发送和接收。始终可进行典型 CAN 帧的发送和接收。是否可发送 CAN FD 帧或典型 CAN 帧可通过相应发送缓冲区元素中的 FDF 位配置。如果 FDOE = 0，接收到的帧会被当作典型 CAN 帧，从而会在接收到 CAN FD 帧时发送错误帧。如果 CAN FD 操作已禁止，即使发送缓冲区元素的 FDF 位置 1，也不会发送 CAN FD 帧。FDCAN_CCCR 寄存器的 FDOE 和 BRSE 位仅可在 INIT 和 CCE 位均被置位时更改。

如果 FDOE = 0，则会忽略 FDF 和 BRS 位的设置，并会以典型 CAN 格式发送帧。如果 FDOE = 1 且 BRSE = 0，则仅会评估发送缓冲区元素的 FDF 位。如果 FDOE = 1 且 BRSE = 1，则会使能通过比特率切换发送 CAN FD 帧。所有 FDF 和 BRS 位置 1 的发送缓冲区元素均会通过比特率切换以 CAN FD 格式发送。

建议仅当满足以下条件下在 CAN 操作期间切换模式：

- CAN FD 数据阶段的故障率显著高于 CAN FD 仲裁阶段的故障率。在这种情况下，应禁止发送的 CAN FD 比特率切换选项。
- 系统启动期间，所有节点在被验证能够以 CAN FD 格式进行通信之前，都将发送典型 CAN 消息。如果通过验证，所有节点会切换为 CAN FD 操作。
- CAN 局部网络中的唤醒消息必须以典型 CAN 格式进行发送。
- 并非所有节点均能进行 CAN FD 操作时采用行结束编程。非 CAN FD 节点在编程完成之前会一直保持静默模式。随后，所有节点会切换回典型 CAN 通信。

在 FDCAN 格式中，数据长度码（DLC）的编码与标准 CAN 格式不同。
DLC 编码 0 至 8：与标准 CAN 中的编码相同。
DLC 编码 9 至 15（在标准 CAN 中，这些编码均表示 8 字节的数据字段）：根据下表进行编码。

表 26.3 FDCAN 中的 DLC 编码

DLC	9	10	11	12	13	14	15
数据字节数	12	16	20	24	32	48	64

在 CAN FD 快速帧中，如果该位为隐性，则在 BRS（比特率切换）位后，位时长将在帧内切换。在 BRS 位之前，在 FDCAN 仲裁阶段，标准 CAN 位时长按照标称位定时和预分频器寄存器 FDCAN_NBTP 定义的方式使用。在接下来的 FDCAN 数据阶段，快速 CAN 位时长按照数据位定时和预分频器寄存器 FDCAN_DBTP 定义的方式使用。位时长会在出现 CRC 分隔符或检测到错误时从快速时长切换回来，以先发生的事件为准。

CAN FD 数据阶段的最大可配置比特率取决于 FDCAN 内核时钟频率。举例来说，如果 FDCAN 内核时钟频率为 20 MHz，最短可配置位时间为四个时间片（t_q），则数据阶段的比特率为 5 Mbit/s。

在 CAN FD 长帧和 CAN FD 快速帧这两种数据帧格式中，位 ESI（错误状态指示符）的值是由发送开始时的发送器错误状态决定的。如果发送器为错误被动状态，则 ESI 会隐性传送，否则会显性传送。在 CAN FD 远程帧中，ESI 位始终显性传送，与发送器错误状态无关。CAN FD 远程帧的数据长度代码以 0 的形式发送。

为 DLC > 8 的 FDCAN 发送配置 FDCAN 发送缓冲区时，前 8 个字节会按照发送缓冲区中的配置发送，而数据字段的剩余部分会用 0xCC 填充。当 FDCAN 接收到 DLC > 8 的 FDCAN 帧时，该帧的前 8 个字节会存储到匹配的接收缓冲区或接收 FIFO 中。剩余字节会被丢弃。

收发器延迟补偿

在 FDCAN 发送的数据阶段，只有一个节点会进行发送，所有其他节点都是接收器。总线的长度没有影响。通过引脚 FDCAN_TX 进行发送时，协议控制器会通过引脚 FDCAN_RX 接收到其本地 CAN 收发器发送的数据。接收数据的延迟为 CAN 收发器环路延迟。如果该延迟大于 TSEG1（采样点之前的时间段），则会检测到位错误。如果不进行收发器延迟补偿，FDCAN 帧数据阶段的比特率会受到收发器环路延迟的限制。

FDCAN 会实施延迟补偿机制来补偿 CAN 收发器环路延迟，从而可在 FDCAN 数据阶段以较高的比特率进行发送，而不受特定 CAN 收发器延迟的影响。

为了检查发送节点的数据阶段是否存在位错误，会将延迟发送数据与在第二采样点（SSP）接收到的数据进行比较。如果检测到位错误，发送器将在下一常规采样点对这一位错误作出响应。在仲裁阶段，始终会禁止延迟补偿。

发送器延迟补偿会使能数据位时间短于发送器延迟的配置，新版 ISO11898-1 中进行了详细介绍。发送器延迟补偿是通过将 FDCAN_DBTP 寄存器的 TDC 位置 1 使能的。

接收到的位会与在 SSP 发送的位进行比较。SSP 位置定义为从 FDCAN 发送输出引脚 FDCAN_TX 通过收发器到接收输入引脚 FDCAN_RX 测得的延迟与由 FDCAN_TDCR 的 TDCO[6:0]配置的发送器延迟补偿偏移之和。发送器延迟补偿偏移用于调整已接收位中 SSP 的位置（例如数据阶段的半个位时间）。第二采样点的位置会向下舍入到下一 mt_q 整数（最小时间片，即一个 fdcan_tq_ck 时钟周期）。

FDCAN_PSR 寄存器的 TDCV[6:0]位显示了实际发送器延迟补偿值。当 FDCAN_CCCR 寄存器的 INIT 位置 1 时，TDCV[6:0]会清零；并且如果 FDCAN_DBTP 寄存器的 TDC 位置 1，每次发送 FD 帧时都会更新该值。

对于在 FDCAN 中实施的发送器延迟补偿，必须考虑以下边界条件：

- 从 FDCAN_Tx 到 FDCAN_Rx 的测量延迟与配置的发送方延迟补偿偏移量 TDCO[6:0]之和，在数据阶段必须小于 6 个位时间。
- 从 FDCAN_TX 到 FDCAN_RX 的测量延迟与配置的发送方延迟补偿偏移量 TDCO[6:0]之和，必须小于或等于 127 × mt_q。若该和超过此值，则发送方延迟补偿将使用最大值（127 × mt_q）。
- 数据阶段在 CRC 分隔符的采样点结束，此时会停止检查在 SSP 接收到的位。

若通过设置 FDCAN_DBTP 寄存器的 TDC 位来启用发送方延迟补偿，则测量将在每个发送的 CAN FD 帧中从位 FDF 到位 res 的下降沿开始。当在发送方的接收输入引脚 FDCAN_TX 处看到该下降沿时，测量停止。此测量的分辨率为一个 mt_q。

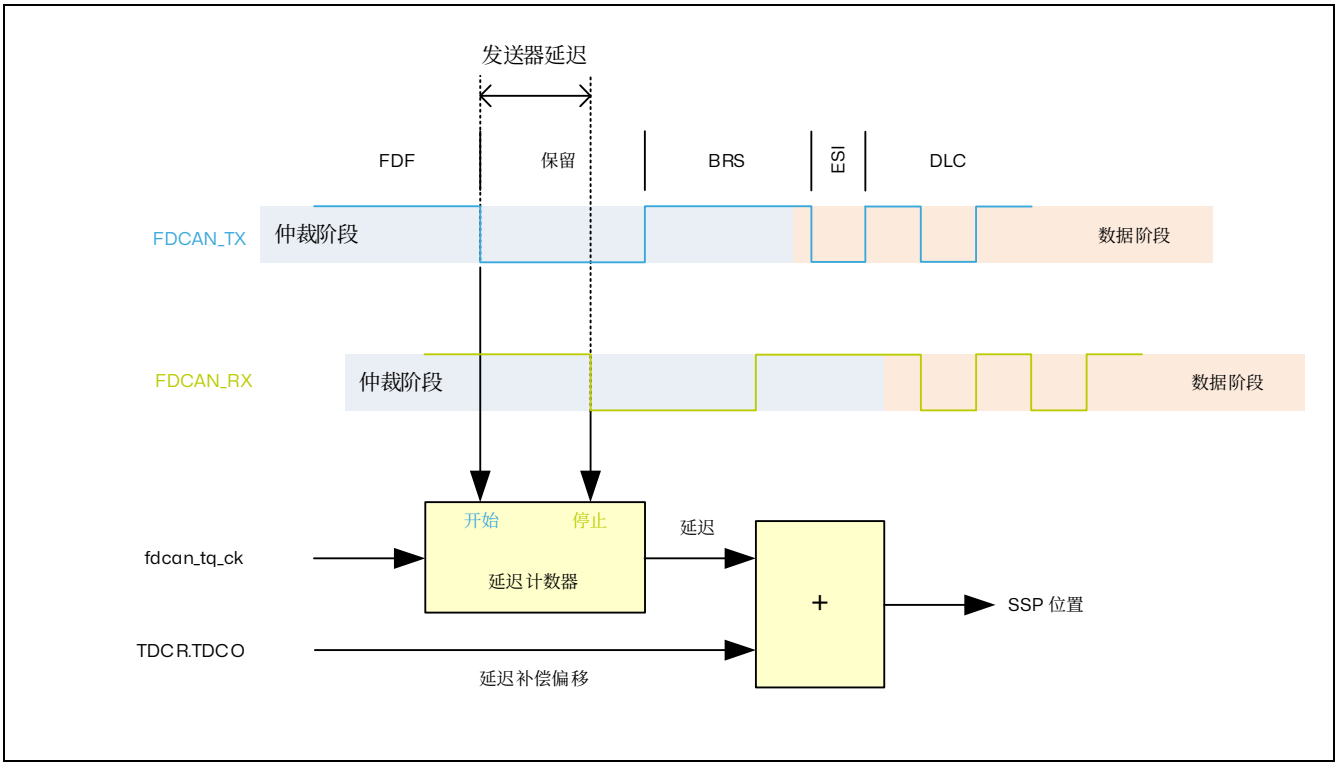


图 26.4 收发延迟测量

为了避免接收到的 FDF 位中的显性毛刺信号导致在接收到的 res 位下降沿之前结束延迟补偿测量（造成 SSP 位置过早），可通过编程 FDCAN_TDCR 寄存器的 TDCF[6:0]位域来使能发送器延迟补偿过滤器窗口，从而定义最小 SSP 位置值。进行发送器延迟测量时，会忽略可能导致 SSP 位置更加提前的 FDCAN_RX 上的显性边沿。当 SSP 位置至少为 TDCF[6:0]且 FDCAN_RX 为低电平时，会停止测量。

受限工作模式

在受限工作模式下，节点能够接收数据帧和远程帧，并能对有效帧进行确认，但不会发送数据帧、远程帧、有效错误帧或过载帧。如果存在错误条件或过载条件，则不会发送显性位，而会等待出现总线空闲条件，以便使自身与 CAN 通信重新同步。错误记录（CEL[7:0]）激活时，错误计数器（FDCAN_ECR 中的 REC[6:0]和 TEC[7:0]）会冻结。软件可通过将 FDCAN_CCCR 寄存器的 ASM 位置 1 的方式将 FDCAN 设置为进入受限工作模式。仅当 FDCAN_CCCR 寄存器中的 CCE 和 INIT 位均被置位时，才能通过软件将此位置 1。此位可随时通过软件清零。

当发送处理单元无法及时从消息 RAM 中读取数据时，会自动进入受限工作模式。要退出受限工作模式，软件必须复位 FDCAN_CCCR 寄存器的 ASM 位。

受限工作模式可用于根据不同 CAN 比特率对自身进行调整的应用中。在这种情况下，应用会测试不同比特率，并在收到有效帧后退出受限工作模式。

注：受限操作模式不得与环回模式（内部或外部）结合使用。

总线监控模式

将 FDCAN_CCCR 寄存器的 MON 位置 1，FDCAN 会被置于总线监控模式。在总线监控模式下（更多详细信息，请参见 ISO11898-1，10.12 总线监控），FDCAN 能够接收有效数据帧和有效远程帧，但不能启动发送过程。在该模式下，FDCAN 仅会在 CAN 总线上发送隐性位，如果 FDCAN 必须发送一个显性位（ACK 位、过载标志、活动错误标志），该位将在内部被改道发送，以便 FDCAN 可以监视该显性位，但 CAN 总线可以保持隐性状态。在总线监控模式下，FDCAN_TXBRP 寄存器会保持复位状态。

总线监控模式可用于分析 CAN 总线上的通信流量，而不会因发送显性位而对其产生影响。下图展示了在总线监控模式下，FDCAN_TX 和 FDCAN_RX 信号与 FDCAN 的连接方式。

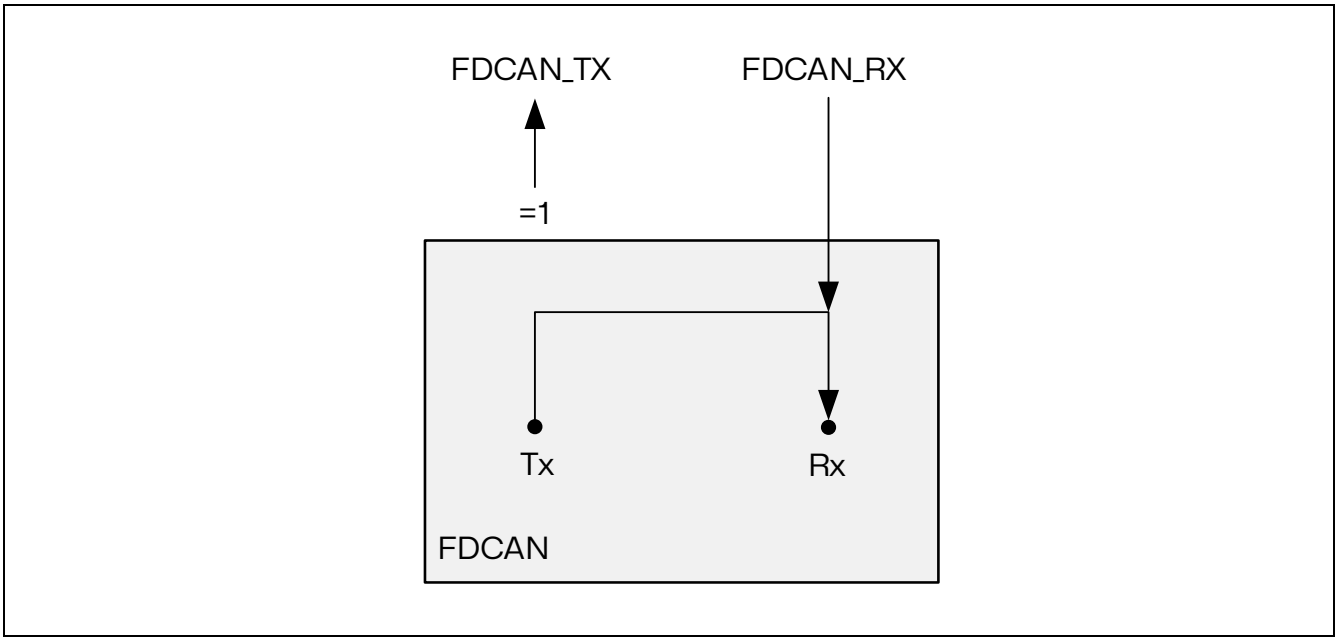


图 26.5 总线监控模式下的引脚控制

禁止自动重发送（DAR）模式

根据 CAN 规范（请参见 ISO11898-1，6.3.3 恢复管理），FDCAN 提供的方法可自动重发已丢失仲裁的帧或在发送期间受到错误干扰的帧。默认使能自动重发送。通过 FDCAN_CCCR 寄存器的 DAR 位，可以禁用 DAR 模式。

禁止自动重发送（DAR）模式下的帧发送

在 DAR 模式下，所有发送在 CAN 总线上启动后都会自动取消。发送缓冲区发送请求挂起位 FDCAN_TXBRP 中的 TRPx 会在成功发送后复位，如果取消时发送尚未开始、发送已因仲裁丢失而中止、或者帧发送期间出错，该位也会复位。

- 成功发送
  - 在 FDCAN_TXBTO 寄存器中，相应的发送缓冲区发送已发生位 TOx 置 1。
  - 在 FDCAN_TXBCF 寄存器中，相应的发送缓冲区取消完成位 CFx 被清除。
- 即使已取消发送也会成功进行发送：
  - 在 FDCAN_TXBTO 寄存器中，相应的发送缓冲区发送已发生位 TOx 置 1。
  - 在 FDCAN_TXBCF 寄存器中，相应的发送缓冲区取消完成位 CFx 置 1。
- 仲裁丢失或帧发送受到干扰
  - 在 FDCAN_TXBTO 寄存器中，相应的发送缓冲区发送已发生位 TOx 被清除。
  - 在 FDCAN_TXBCF 寄存器中，相应的发送缓冲区取消完成位 CFx 被置位。

对于帧发送成功的情况，如果使能了发送事件存储，则会写入发送事件 FIFO 元素，写入的事件类型 ET = 10（即使已取消发送也会进行发送）。

掉电（休眠模式）

进入掉电模式

通过设置 FDCAN_CCCR 寄存器的 CSR 位，可将 FDCAN 设置为掉电模式。只要时钟停止请求有效，读取 CSR 位将返回 1。

所有挂起发送请求完成后，FDCAN 会一直等待至检测到总线空闲状态。检测到总线空闲状态之后，FDCAN 会置 1，随后，FDCAN_CCCR 寄存器的 INIT 位会置 1，以免继续进行 CAN 传输。现在，FDCAN 会通过将 FDCAN_CCCR 寄存器的 CSA 位置 1 确认其准备好进入掉电状态。在该模式下，时钟关闭之前，可继续访问寄存器。对 INIT 位的写访问不起作用。现在，可以关闭模块时钟输入。

退出掉电模式

退出掉电模式之前，应用程序必须在清除 CSR 位之前打开模块时钟。FDCAN 将通过复位 CSA 位确认时钟开启。之后，应用可通过复位 INIT 位重新开始 CAN 通信。

测试模式

要使能对 FDCAN 测试寄存器（FDCAN_TEST）的写访问，必须将 FDCAN_CCCR 寄存器的 TEST 位置 1，从而能够配置测试模式和功能。

可通过编程 FDCAN_TEST 寄存器的 TX[1:0]位域将四种输出功能用于 CAN 发送引脚 FDCAN_TX。除了串行数据输出这一默认功能之外，还可驱动 CAN 采样点信号来监控 FDCAN 位时长，并可驱动常量显性或隐性值。引脚 FDCAN_RX 的实际值可从 FDCAN_TEST 寄存器的 RX 位读取。这两种功能均可用于检查 CAN 总线物理层。

由于 CAN 内核时钟与 APB 时钟域之间的同步机制，在 FDCAN_TX 输出引脚上显示新配置之前，对 TX[1:0]的写操作之间可能有几个 APB 时钟周期的延迟。这一点还适用于通过 RX 位读取 FDCAN_RX 输入引脚的情况。

注：测试模式仅限用于生产测试和自检。对 FDCAN_TX 引脚进行软件控制会干扰所有 CAN 协议功能。不建议为应用使用测试模式。

外部环回模式

通过设置 FDCAN_TEST 寄存器的 LBCK 位，可将 FDCAN 设置为外部环回模式。在环回模式下，FDCAN 将其自身发送的消息作为接收的消息来处理，并将消息存储（如果这些消息通过了验收过滤）在接收 FIFO 中。下图显示了外部环回模式下发送和接收信号 FDCAN_TX 和 FDCAN_RX 与 FDCAN 的连接。

该模式为硬件自检提供。为了不受外部仿真的影响，FDCAN 在环回模式下将忽略确认错误（在数据/远程帧的确认时隙对隐性位采样）。在此模式下，FDCAN 将执行从发送输出到其接收输入的内部反馈。FDCAN 将忽略 FDCAN_RX 输入引脚的实际值。从 FDCAN_TX 发送引脚可以监视发送的消息。

内部环回模式

将 FDCAN_TEST 寄存器的 LBCK 位和 FDCAN_CCCR 寄存器的 MON 位编程为 1 会进入内部环回模式。该模式可用于“热自检”，也就是说，FDCAN 可以进行检测，同时又不会影响与 FDCAN_TX 和 FDCAN_RX 引脚相连接的运行中的 CAN 系统。在此模式下，FDCAN_RX 引脚与 FDCAN 断开连接，FDCAN_TX 引脚则保持隐性。下图显示了内部环回模式下 FDCAN_TX 和 FDCAN_RX 引脚与 FDCAN 的连接。

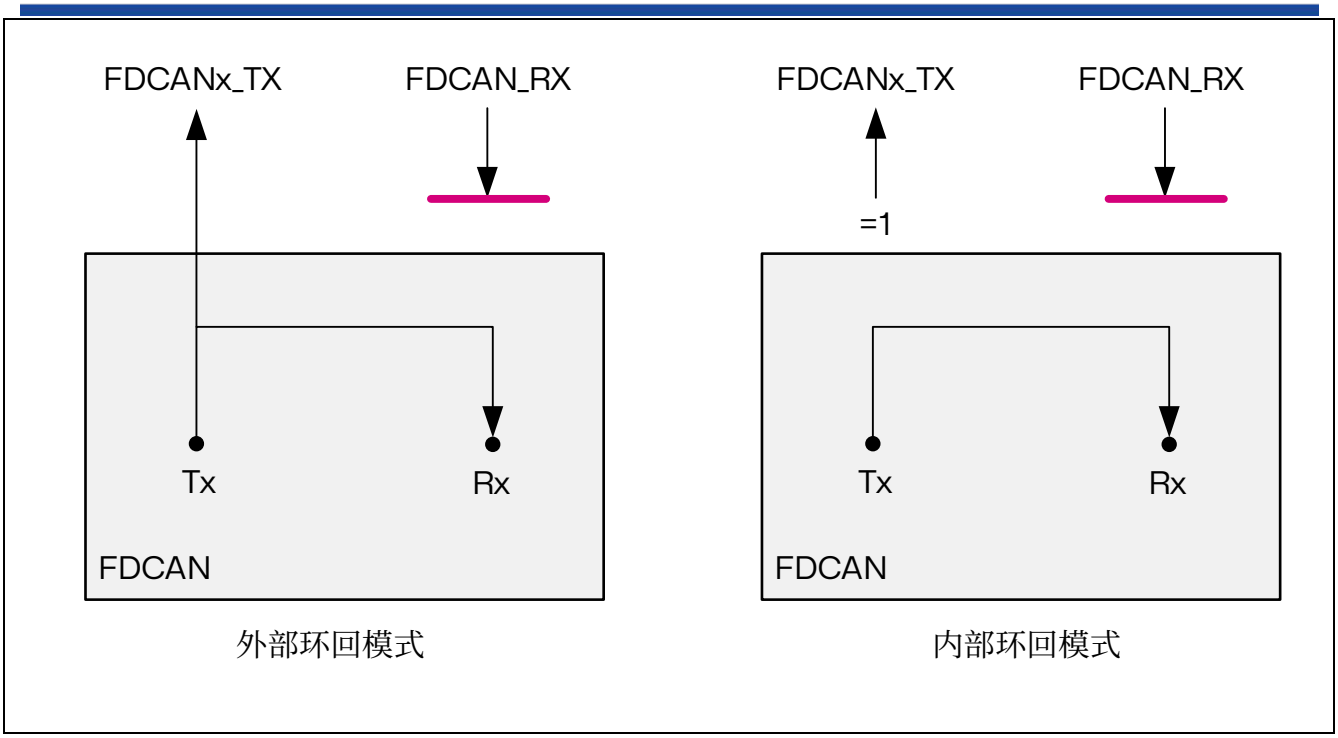


图 26.6 环回模式下的引脚控制

时间戳生成

FDCAN 提供一个 16 位回卷计数器来生成时间戳。预分频器（FDCAN_TSCC 寄存器的 TCP[3:0]位）可配置为以 CAN 位时间的倍数（1...16）为计数器提供时钟。计数器值可通过 FDCAN_TSCV 寄存器的 TCV[15:0]位域读取。对 TSCV[15:0]进行写访问会将计数器复位为 0。时间戳计数器回卷时，中断标志（FDCAN_ISR 寄存器的 TSW 位）会置 1。

开始接收/发送帧时，会捕获计数器值，并将该值存储在接收缓冲区/接收 FIFO（RXTS[15:0]）或发送事件 FIFO（TXTS[15:0]）元素的时间戳部分。

通过编程 FDCAN_TSCC 寄存器的 TSS[1:0]位，可以使用 16 位时间戳。

调试模式行为

在调试模式下，在调试器进行寄存器访问期间，读即置位/复位功能会自动禁用，而在 MCU 正常操作期间则启用。

超时计数器

FDCAN 提供 16 位超时计数器来指示接收 FIFO 0、接收 FIFO 1 和发送事件 FIFO 的超时条件。它作为递减计数器工作，并且与时间戳计数器一样，使用由 FDCAN_TSCC 寄存器的 TCP[3:0]位控制的预分频器。超时计数器通过寄存器 FDCAN_TOCC 配置。实际计数器值可从 FDCAN_TOCV 寄存器的 TOC[15:0]位域读取。仅当 FDCAN_CCCR 寄存器的 INIT 位被清除时，才会启动超时计数器。当 INIT 位被置位时，超时计数器停止计数，例如 FDCAN 进入 Bus_Off 状态。

工作模式是通过 FDCAN_TOCC 寄存器的 TOS[1:0]位选择的。在连续模式下工作时，计数器会在 INIT 位复位时开始计数。对 FDCAN_TOCV 寄存器进行写操作会将计数器预设为由 FDCAN_TOCC 寄存器中 TOP[15:0]位配置的值并继续递减计数。

超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为由 FDCAN_TOCC 寄存器中 TOP[15:0]位配置的值。存储了第一个 FIFO 元素后，会开始递减计数。对 FDCAN_TOCV 寄存器执行写入操作不起作用。

计数器达到 0 时，FDCAN_IR 寄存器中的 TOO 中断标志位会置 1。在连续模式下，计数器会在达到 TOP[15:0]位配置的值后立即重启。

注：超时计数器的时钟信号来自 CAN 内核采样点信号。因此，由于 CAN 内核同步/重新同步机制的原因，超时计数器递减的时间点可能有所不同。如果使用 FDCAN 中的波特率切换功能，则超时计数器在仲裁字段和数据字段中使用不同的时钟。

26.3.5 错误管理

如 CAN 协议所述，错误管理完全由硬件处理，使用发送错误计数器（FDCAN 错误计数器寄存器（FDCAN_ECR）中的 TEC[7:0]位域）和接收错误计数器（FDCAN 错误计数器寄存器（FDCAN_ECR）中的 REC[6:0]位域）。根据错误条件，这些值会递增或递减。有关 TEC[7:0]和 REC[6:0]管理的详细信息，请参考 CAN 标准。软件可读取这两个值以确定网络的稳定性。

当 TEC[7:0]大于 255 时，会进入 bus-off 总线关闭状态。该状态也由 FDCAN 协议状态寄存器（FDCAN_PSR）的 BO 标志位指示。在 bus-off 总线关闭状态下，CAN 无法再发送和接收消息。它必须至少等待 CAN 标准中规定的恢复序列持续时间（在 FDCAN_RX 输入端监测到 128 次连续 11 个隐性位）。

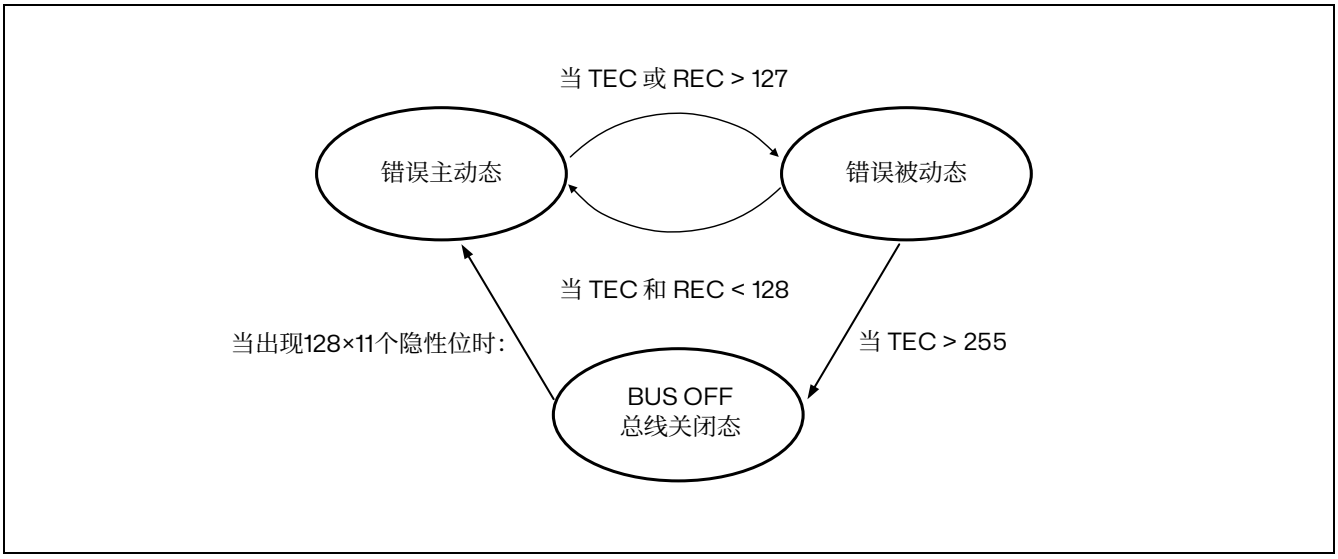


图 26.7 CAN 错误状态图

注：在初始化模式下，CAN 不监测 FDCAN_RX 信号，因此无法完成恢复序列。要从错误状态中恢复，CAN 必须在正常模式下运行。

26.3.6 消息 RAM

消息 RAM 的宽度为 32 位，FDCAN 模块被配置为可在其中分配最多 212 个字。无需对下图中所示的每个部分都进行配置。

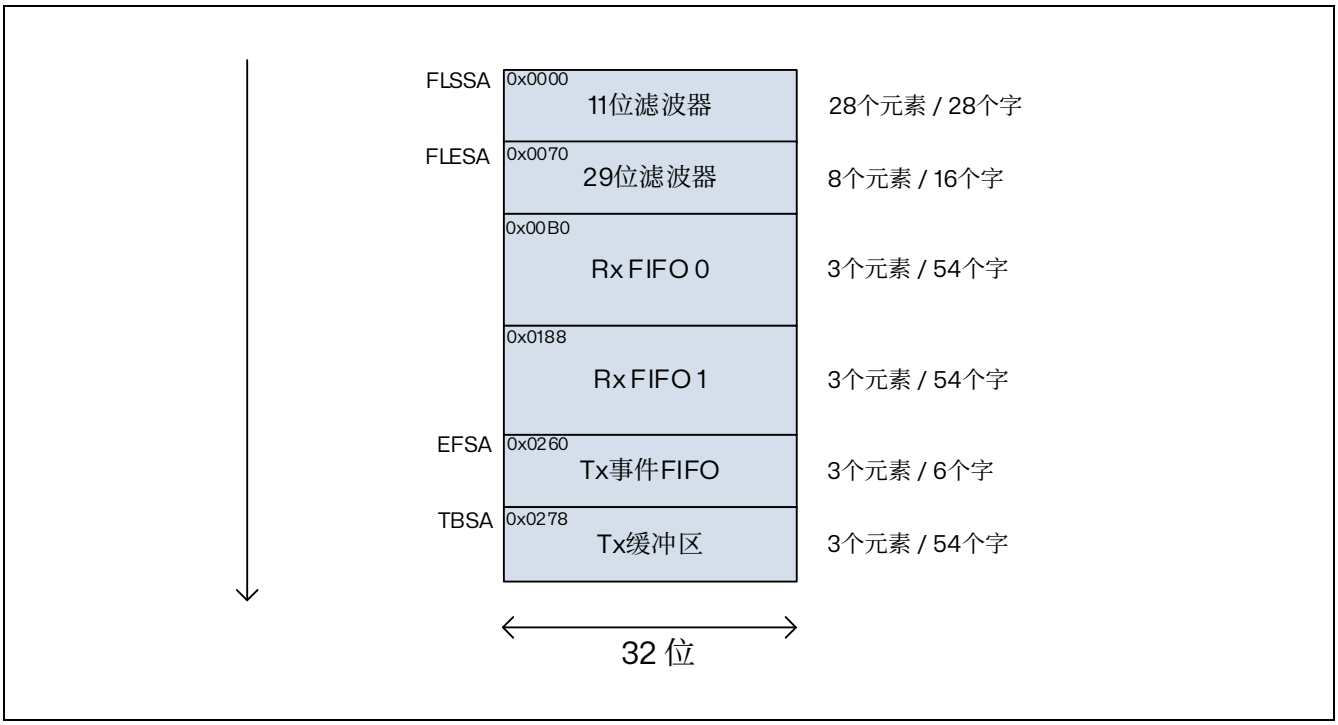


图 26.8 消息 RAM 配置

当 FDCAN 访问消息 RAM 时，它访问的是 32 位字（已对齐），而非单个字节。RAM 地址以 32 位字为单位，即仅评估第 15 位至第 2 位，最低两位被忽略。

接收处理

接收处理单元控制着验收过滤、已接收消息到接收缓冲区或其中一个接收 FIFO（共两个）的传输以及接收 FIFO 放入和获取索引。

验收滤波器

FDCAN 能够配置两组验收过滤器，一组供标准标识符使用，另一组供扩展标识符使用。这两组过滤器可分配给接收缓冲区、接收 FIFO 0 或接收 FIFO 1。进行验收过滤时，会从元素#0 开始执行每个过滤器列表，直至过滤出第一个匹配元素。过滤出第一个匹配元素后，验收过滤会停止。不会为该消息评估以下过滤器元素。

主要特性有：

- 每个过滤器元素可配置为：
  - 范围过滤器（起止范围）
  - 用于一个或两个专用 ID 的过滤器
  - 典型位屏蔽过滤器
- 每个过滤器元素均可配置为实现验收过滤或拒绝过滤
- 每个过滤器元素均可单独使能/禁止
- 会按顺序检查过滤器，过滤出第一个匹配的过滤器元素后会停止执行

相关配置寄存器位：

- 全局过滤器配置（RXGFC）
- 扩展 ID 与掩码（XIDAM）

根据过滤器元素的配置（SFEC[2:0]/EFEC[2:0]），发生匹配时会触发下列操作之一：

- 将接收到的帧存储在 FIFO 0 或 FIFO 1 中
- 拒绝接收到的帧
- 在 FDCAN_IR 中设置高优先级消息中断标志 HPM
- 在 FDCAN_IR 中设置高优先级消息中断标志 HPM，并将接收到的帧存储在 FIFO 0 或 FIFO 1 中

接收到完整的标识符后，开始验收过滤。验收过滤完成后，如果已找到匹配的接收缓冲区或接收 FIFO，消息处理单元便会开始将接收到的消息数据以 32 位的形式写入匹配的接收缓冲区或接收 FIFO 中。如果 CAN 协议控制器已检测到错误条件（例如 CRC 错误），则会丢弃此消息，丢弃后的影响如下：

- **接收 FIFO**
  匹配接收 FIFO 的放入索引不会更新，但相关接收 FIFO 元素会被接收到的数据覆盖。有关错误类型，请参见 FDCAN_PSR 寄存器的 LEC[2:0]和 DLEC[2:0]位域。如果匹配的接收 FIFO 在覆盖模式下工作，必须考虑接收 FIFO 覆盖模式中介绍的边界条件。

注：当接受的消息写入两个接收 FIFO 之一时，未修改的已接收标识符会独立于所用过滤器进行存储。验收过滤过程的结果很大程度上取决于所配置的过滤器元素的顺序。

范围过滤器

该过滤器会将所有接收到的帧与由 SF1ID/SF2ID 和 EF1ID/EF2ID 定义的范围内的消息 ID 进行匹配检查。

范围过滤与扩展帧一起使用时，有两种可能的情况：

- EFT[1:0] = 00：应用范围过滤器之前，会将已接收帧的消息 ID 与扩展 ID 与掩码（XIDAM）进行与运算
- EFT[1:0] = 11：不会为范围过滤使用扩展 ID 与掩码 (XIDAM)

专用 ID 过滤器

可将过滤器元素配置为过滤一个或两个特定的消息 ID。要过滤一个特定的消息 ID，过滤器元素必须配置为 SF1ID = SF2ID 且 EF1ID = EF2ID。

典型位屏蔽过滤器

典型位屏蔽过滤用于通过屏蔽已接收消息 ID 的单个位来过滤消息 ID 组。进行典型位屏蔽过滤时，SF1ID/EF1ID 用作消息 ID 过滤器，SF2ID/EF2ID 则用作过滤器屏蔽位。

过滤器屏蔽的 0 位将屏蔽掉已配置 ID 过滤器的相应位位置，例如，已接收消息 ID 在该位位置的值与验收过滤无关。只有相应屏蔽位为 1 的已接收消息 ID 的位才与验收过滤有关。

如果所有屏蔽位均为 1，则仅当接收到的消息 ID 和消息 ID 过滤器完全相同时，才会出现匹配。如果所有屏蔽位均为 0，则所有消息 ID 都匹配。

标准消息 ID 过滤

下图展示了标准消息 ID（11 位标识符）过滤流程。标准消息 ID 过滤器元素在第 8 节中进行了描述。

标准消息过滤由 FDCAN_RXGFC 寄存器控制。接收到的帧的标准消息 ID、远程传输请求位（RTR）以及标识符扩展位（IDE）会与配置的过滤器元素列表进行比较。

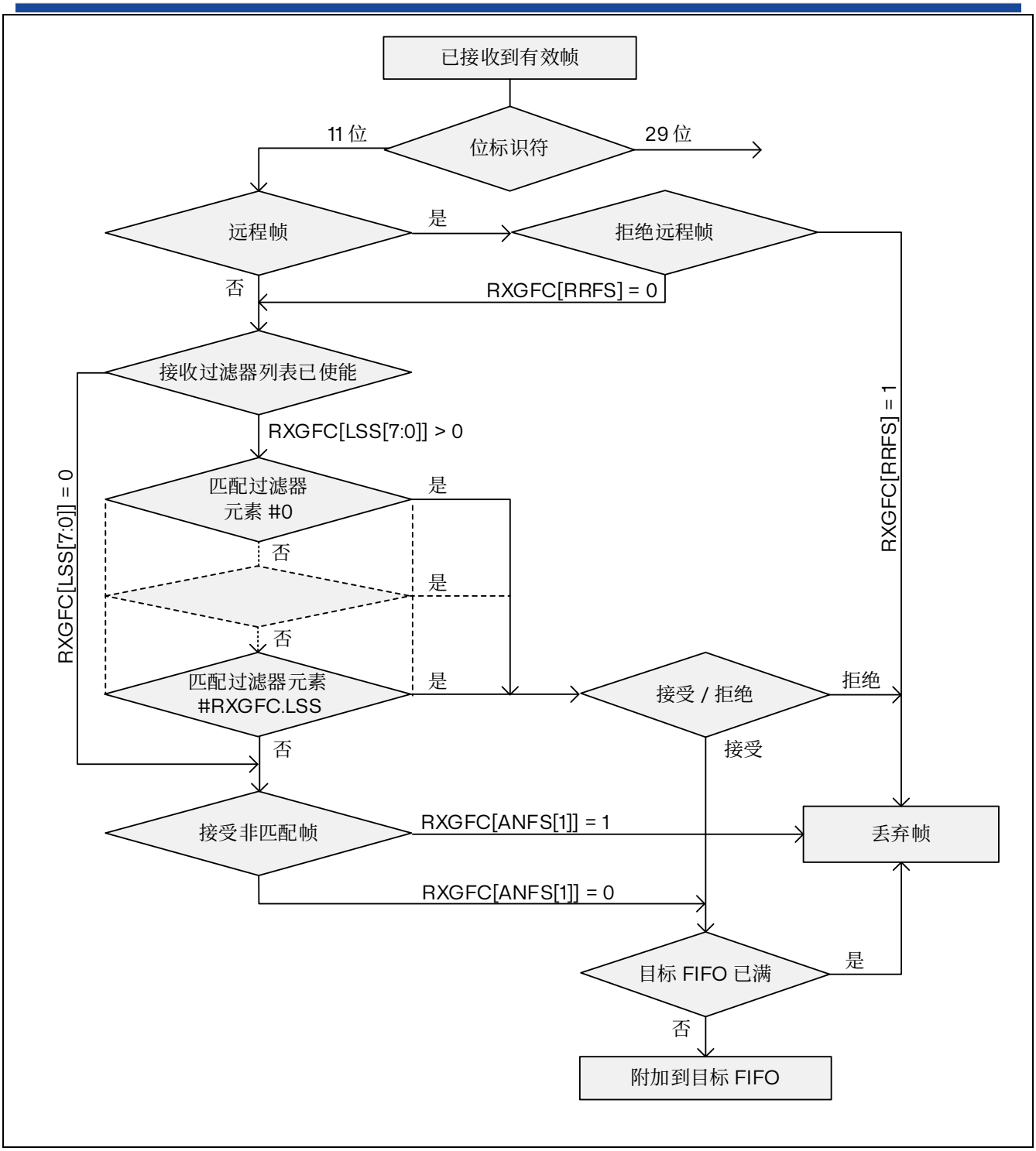


图 26.9 标准消息 ID 过滤器路径

扩展消息 ID 过滤

下图展示了扩展消息 ID（29 位标识符）过滤的流程。扩展消息 ID 过滤器元素在第 9 节中进行了描述。

扩展消息滤波由 FDCAN_RXGFC 寄存器控制。接收到的帧的扩展消息 ID、远程传输请求位（RTR）以及标识符扩展位（IDE）会与配置的滤波器元素列表进行比较。

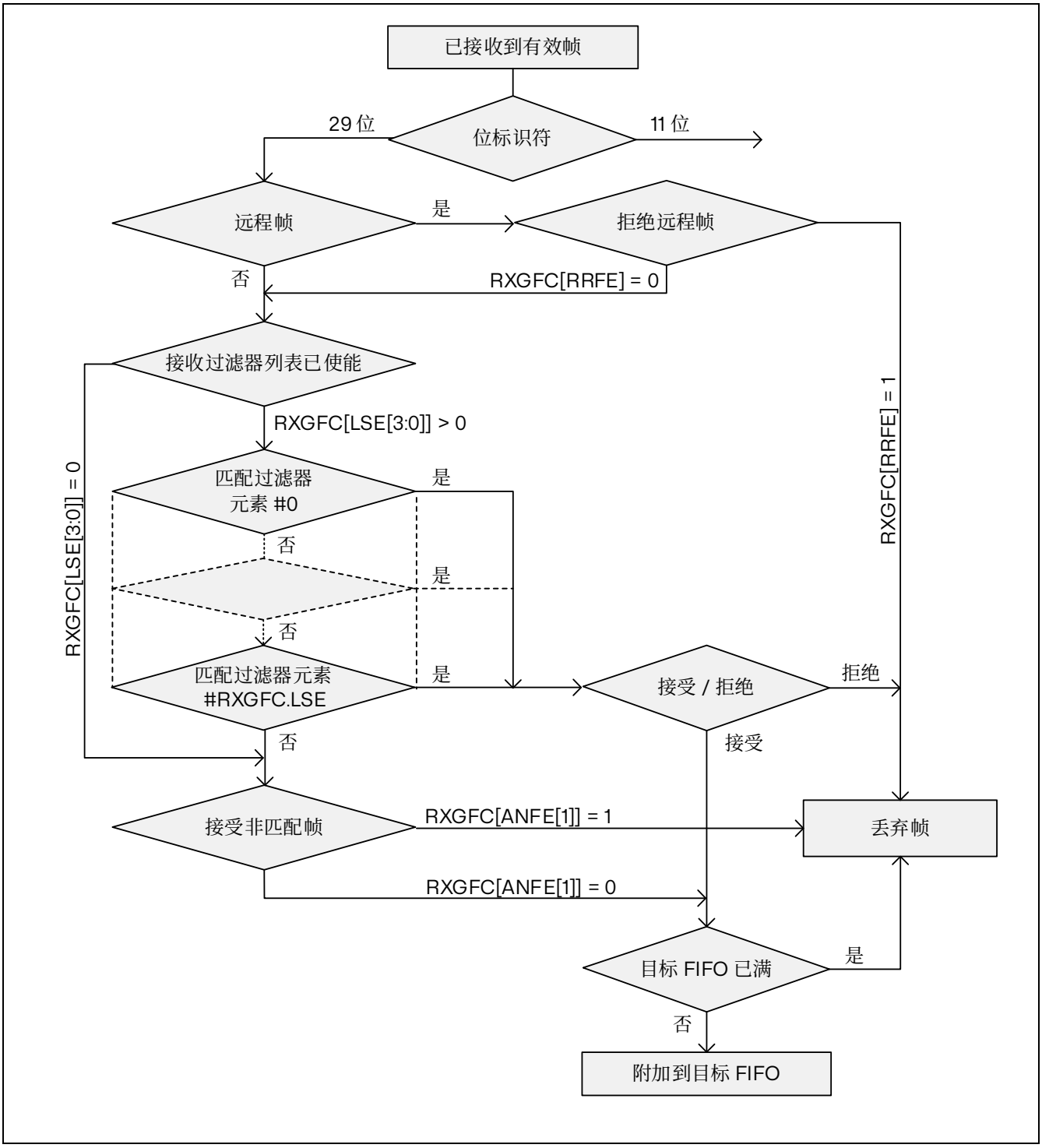


图 26.10 扩展消息 ID 过滤器路径

扩展 ID 与掩码（XIDAM）会在执行过滤器列表之前与接收到的标识符进行与运算。

接收 FIFO

接收 FIFO 0 和接收 FIFO 1 最多可分别保存三个元素。

通过验收过滤的已接收消息会传输到由匹配过滤器元素配置的接收 FIFO 中。有关可用于接收 FIFO 0 和接收 FIFO 1 的过滤器机制说明，请参见验收过滤器。接收 FIFO 元素在第 5 节中进行了描述。

当 FDCAN_IR 中的 RFnF（其中 n 为 FIFO 编号）指示接收 FIFO 已满条件时，在至少已读取一条消息且接收 FIFO 获取索引递增之前，不会继续向相应的接收 FIFO 写入消息。如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，FDCAN_IR 寄存器中的 RFnL 中断标志会置 1。

从接收 FIFO 中读取消息时，接收 FIFO 获取索引（FDCAN_RXFnS 的 FnGI）加上 FIFO 元素大小必须与相应的接收 FIFO 起始地址（FnSA）相加。

接收 FIFO 阻止模式

接收 FIFO 阻止模式是通过清除 FDCAN_RXGFC 寄存器中的 FnOM 位配置的。该模式是接收 FIFO 的默认工作模式。

达到接收 FIFO 已满条件时（FDCAN_RXFnS 中的 FnPI = FnGI），在至少已读取一条消息且接收 FIFO 获取索引递增之前，不会继续向相应的接收 FIFO 写入消息。接收 FIFO 已满条件是通过 FDCAN_RXFnS 中的 FnF = 1 指示的。此外，FDCAN_IR 中的 RFnF 中断标志也会置 1。

如果在相应的接收 FIFO 已满时收到消息，此消息会被丢弃，并会通过设置 FDCAN_RXFnS 中的 RFnL 位指示消息丢失条件。此外，FDCAN_IR 中的 RFnL 中断标志也会置 1。

接收 FIFO 覆盖模式

接收 FIFO 覆盖模式是通过设置 FDCAN_RXGFC 寄存器中的 FnOM 位配置的。

当 FDCAN_RXFnS 中的 FnF = 1 指示接收 FIFO 已满条件时（FDCAN_RXFnS 中的 FnPI = FnGI），FIFO 接收的下一条消息将覆盖最原始的 FIFO 消息。放入索引和获取索引都会加 1。

如果接收 FIFO 在覆盖模式下工作，并且指示了接收 FIFO 已满条件，则应至少从获取索引+ 1 处开始读取接收 FIFO 元素。这是因为 CPU 从消息 RAM（获取索引）读取消息时可能会将接收到的消息写入消息 RAM（放入索引）。在这种情况下，从相应的接收 FIFO 元素中读取的数据可能不一致。从接收 FIFO 读取数据时向获取索引中添加偏移可避免此问题。偏移取决于 CPU 访问接收 FIFO 的速度。

从接收 FIFO 读取数据后，读取的最后一个元素的编号必须写入接收 FIFO 确认索引（FDCAN_RXFnA 的 FnA）。这样会使获取索引增加到该元素编号。如果放入索引未增加到该接收 FIFO 元素，接收 FIFO 已满条件会复位（FDCAN_RXFnS 中的 FnF = 0）。

发送处理

发送处理单元负责处理发送 FIFO 和发送队列的发送请求。它控制发送消息到 CAN 内核的传输、放入和获取索引，以及发送事件 FIFO。最多可为消息发送设置 3 个发送缓冲区。CAN 消息的数据字段被配置为 64 字节。发送 FIFO 分配了十八个 32 位字用于存储一个发送元素。

表 26.4 帧发送的可能配置

CCCR		发送缓冲区元素		帧发送
BRSE	FDOE	FDF	BRS	
忽略	0	忽略	忽略	典型 CAN
0	1	0	忽略	典型 CAN
0	1	1	忽略	无比特率切换的 FD
1	1	0	忽略	典型 CAN
1	1	1	0	无比特率切换的 FD
1	1	1	1	有比特率切换的 FD

*注：*AUTOSAR *要求至少具备三个发送队列缓冲区，并支持发送取消功能。*

当发送缓冲区请求挂起寄存器（FDCAN_TXBRP）更新时，或者发送已开始时，发送处理单元会启动发送扫描来检查优先级最高的挂起发送请求（消息 ID 最小的发送缓冲区）。

发送暂停

发送暂停功能用于 CAN 消息标识符（永久性地）被指定为特定值、并且无法轻易更改的 CAN 系统中。这些消息标识符的 CAN 仲裁优先级可能高于其他已定义的消息，而在特定的应用中，其相对仲裁优先级应反转。这样可能会导致如下情况：一个电子控制单元（ECU）发送的 CAN 消息突发造成 CAN 仲裁优先级较低的另一个 ECU CAN 消息被延迟。

举例来说，如果 CAN ECU-1 启用了此功能，并且其应用软件请求发送四条消息，成功发送第一条消息后，将等待两个总线空闲的 CAN 位时间，然后才允许开始发送下一条请求的消息。如果其他 ECU 有挂起的消息，则会在空闲时间开始发送这些消息，不需要与 ECU-1 的下一条消息进行仲裁。接收到消息后，ECU-1 可在接收到的消息释放 CAN 总线后立即开始进行下一次发送。

此功能是通过 CCCR 寄存器中的 TXP 位控制的。如果此位置 1，则每次成功发送消息后，FDCAN 都将暂停两个 CAN 位时间，然后再开始进行下一次发送。这样一来，即使网络中其他节点的优先级标识符较低，也能发送消息。默认禁止此功能（FDCAN_CCCR 中的 TXP = 0）。

此功能可使来自单个节点的突发发送变得松散一些，避免出现应用程序错误地请求过多传输的“胡乱指示”情况。

发送 FIFO

发送 FIFO 操作是通过清除 FDCAN_TXBC 寄存器的 TFQM 位来配置的。

发送 FIFO 中存储的消息是先从获取索引（FDCAN_TXFQS 的 TFGI[1:0]位域）引用的消息开始发送的。每次发送后，获取索引会循环递增，直至发送 FIFO 为空。发送 FIFO 可按消息写入发送 FIFO 的顺序发送来自不同发送缓冲区但消息 ID 相同的消息。FDCAN 会计算获取索引和放入索引之差作为发送 FIFO 空闲级别（FDCAN_TXFQS 的 TFFL[2:0]位域），用于指示可用（空闲）的发送 FIFO 元素数。

新的发送消息必须写入以放入索引（FDCAN_TXFQS 的 TFQPI[1:0]位域）引用的发送缓冲区开始的发送 FIFO 中。添加请求会将放入索引增加到下一空闲发送 FIFO 元素。放入索引达到获取索引后，会指示发送 FIFO 已满（FDCAN_TXFQS 中的 TFQF = 1）。在这种情况下，下一条消息已发送且获取索引已递增之前，不应继续向发送 FIFO 写入消息。

如果有一条消息添加到发送 FIFO，则会通过向与发送 FIFO 放入索引引用的发送缓冲区相关的 FDCAN_TXBAR 位写入 1 来请求发送消息。

如果有多条（n 条）消息添加到发送 FIFO，则会写入以放入索引开始的 n 个连续发送缓冲区中。随后会通过 FDCAN_TXBAR 寄存器请求发送。放入索引随后会循环递增 n。请求的发送缓冲区数不应超过发送 FIFO 空闲级别指示的空闲发送缓冲区数。

如果获取索引引用的发送缓冲区的发送请求取消，获取索引会增加到下一个具有挂起发送请求的发送缓冲区，并会重新计算发送 FIFO 空闲级别。如果取消对其他任何发送缓冲区的发送，获取索引和 FIFO 空闲级别保持不变。

发送 FIFO 元素会在消息 RAM 中分配十八个 32 位字。因此下一可用（空闲）发送 FIFO 缓冲区的起始地址是通过将放入索引 TFQPI[1:0]（0 ... 2）乘以四再与发送缓冲区起始地址 TBSA 相加计算得出的。

发送队列

发送队列操作是通过将 FDCAN_TXBC 寄存器的 TFQM 位编程为 1 配置的。

发送队列中存储的消息是先从消息 ID 最小（优先级最高）的消息开始发送的。

如果多个队列缓冲区配置为使用同一消息 ID，则会先发送缓冲区编号最小的队列缓冲区。

新消息必须写入放入索引（FDCAN_TXFQS 中的 TFQPI[1:0]）引用的发送缓冲区中。添加请求会将放入索引循环增加到下一空闲发送缓冲区。如果发送队列已满（FDCAN_TXFQS 中的 TFQF = 1），则放入索引无效，并且在至少有一个请求的消息已发出或挂起的发送请求已取消之前，不应继续向发送队列写入消息。

应用可使用寄存器 FDCAN_TXBRP 来代替放入索引，并可将消息放入任何没有挂起传输请求的发送缓冲区中。

发送队列缓冲区会在消息 RAM 中分配十八个 32 位字。因此下一可用（空闲）发送队列缓冲区的起始地址是通过将发送队列放入索引 TFQPI[1:0]（0 ... 2）乘以四再与发送缓冲区起始地址 TBSA 相加计算得出的。

发送取消

FDCAN 支持发送取消功能。要取消专用发送缓冲区或发送队列缓冲区请求的发送，主机必须向寄存器 FDCAN_TXBCR 中相应的位位置（=发送缓冲区的数量）写入 1。发送取消不适用于发送 FIFO 操作。

成功取消后，寄存器 FDCAN_TXBCF 的相应位会设为 1。

如果已经在从发送缓冲区进行发送时请求发送取消，则只要发送正在进行中，相应的 FDCAN_TXBRP 位就会保持置 1 状态。如果发送成功，相应的 FDCAN_TXBTO 和 FDCAN_TXBCF 位会置 1。如果发送不成功，则不会重复发送，只会将相应的 FDCAN_TXBCF 位置 1。

注：如果在挂起的发送尚未开始之前立即取消此次发送，即使该节点中有另一消息挂起，取消后也会有一个短暂的时间窗，在此期间不会开始任何发送。这样一来，另一节点便可以发送优先级可能比该节点中第二条消息低的消息。

发送事件处理

为了支持发送事件处理，FDCAN 实现了发送事件 FIFO。FDCAN 在 CAN 总线上发送消息后，消息 ID 和时间戳会存储在发送事件 FIFO 元素中。为了将发送事件关联到发送事件 FIFO 元素，已发送的发送缓冲区中的消息标志会被复制到发送事件 FIFO 元素中。

发送事件 FIFO 最多可配置为三个元素。发送 FIFO 中介绍了发送事件 FIFO 元素。

发送事件 FIFO 的用途是将处理发送状态信息与处理发送消息分开，也就是让发送缓冲区仅保存要发送的消息，而将发送状态单独存储在发送事件 FIFO 中。这样做有很大的优势，尤其是在处理动态管理的发送队列时，发送缓冲区可在发送成功后立即用于新消息。覆盖发送缓冲区之前，不需要保存该发送缓冲区的发送状态信息。

当 FDCAN_IR 的 TEFF 位指示发送事件 FIFO 已满条件时，在至少已读取一个元素且发送事件 FIFO 获取索引递增之前，不会继续向发送事件 FIFO 写入元素。如果在发送事件 FIFO 已满时发生发送事件，此事件会被丢弃，FDCAN_IR 寄存器中的 TEFL 中断标志会置 1。

从发送事件 FIFO 读取数据时，必须将发送事件 FIFO 获取索引（FDCAN_TXEFS 的 EFGI[1:0]）乘以二再与发送事件 FIFO 起始地址址 EFSA 相加。

26.3.7 FIFO 确认处理

接收 FIFO 0、接收 FIFO 1 和发送事件 FIFO 是通过向相应的 FIFO 确认索引进行写操作控制的（参见第 27.4.23 节和第 27.4.25 节）。

对 FIFO 确认索引进行写操作会将 FIFO 获取索引设置为 FIFO 确认索引加 1，进而会更新 FIFO 填充级别。有两种用例：

- 如果只从 FIFO 读取了一个元素（获取索引指向的元素），则该获取索引值会写入 FIFO 确认索引中。
- 如果已从 FIFO 中读取一系列元素，则仅在该读取序列结束时写入一次 FIFO 确认索引（数值=读取的最后一个元素的索引）即可更新 FIFO 获取索引。

由于 CPU 可自由访问 FDCAN 消息 RAM，以任意顺序（不考虑获取索引）读取 FIFO 元素时需要特别留意。这一点在从两个接收 FIFO 之一读取高优先级消息时会很有用。在这种情况下，不应写入 FIFO 确认索引，否则会将获取索引设为错误的位置，还会改变 FIFO 填充级别。在这种情况下，一些时间较久的 FIFO 元素会丢失。

注：应用必须确保 FIFO *确认索引中已写入有效值。* FDCAN *不会检查数值是否正确。*

26.3.8 FDCAN 接收 FIFO 元素

在消息 RAM 中配置了 2 个接收 FIFO。每个接收 FIFO 部分均可配置为最多存储 3 个已接收消息。接收 FIFO 元素的结构如下方第一张表所示，相关说明请参见下方第二张表。

表 26.5 接收 FIFO 元素

Bit	31			24	23			16	15	8	7	0
R0	ESI	XTD	RTR	ID[28:0]								
R1	ANMF	FIDX[6:0]			Res.	FDF	BRS	DLC[3:0]	RXTS[15:0]			
R2	DB3[7:0]				DB2[7:0]				DB1[7:0]		D[7:0]	
R3	DB7[7:0]				DB6[7:0]				DB5[7:0]		DB4[7:0]	
⋮	⋮				⋮				⋮			
Rn	DBm[7:0]				DBm-1[7:0]				DBm-2[7:0]		DBm-3[7:0]	

为存储 CAN FD 消息而配置的元素大小被设置为 64 字节的数据字段。

表 26.6 接收 FIFO 元素说明

位域	说明
R0 Bit 31 ESI	错误状态指示符（Error State Indicator） – 0：发送节点为错误主动状态 – 1：发送节点为错误被动状态
R0 Bit 30 XTD	扩展标识符（Extended identifier） 向主机指示接收到的帧具有标准标识符还是扩展标识符。 – 0：11 位标准标识符 – 1：29 位扩展标识符
R0 Bit 29	远程发送请求（Remote Transmission Request）

位域	说明
RTR	向主机指示接收到的帧属于数据帧还是远程帧。 – 0：接收到的帧为数据帧 – 1：接收到的帧为远程帧
R0 Bits 28:0 ID[28:0]	标识符（Identifier） 标准标识符或扩展标识符取决于 XTD 位。标准标识符存储在 ID[28:18]中。
R1 Bit 31 ANMF	接受非匹配帧（Accepted Non-matching Frame） 可通过 FDCAN_RXGFC 寄存器的 ANFS[1:0]和 ANFE[1:0]位域使能接受非匹配帧。 – 0：接收到的帧与过滤器索引 FIDX 匹配 – 1：接收到的帧与任何接收过滤器元素都不匹配
R1 Bits 30:24 FIDX[6:0]	过滤器索引（Filter Index） 0-27 = 匹配接收验收过滤器元素的索引（ANMF = 1 时无效）。 范围：FDCAN_RXGFC 寄存器中 0 到 LSS[4:0] - 1 或 LSE[3:0] - 1。
R1 Bit 21 FDF	FD 格式（FD Format） – 0：标准帧格式 – 1：FDCAN 帧格式（新 DLC 编码和 CRC）
R1 Bit 20 BRS	比特率切换（Bit Rate Switch） – 0：接收帧时不切换比特率 – 1：接收帧时切换比特率
R1 Bits 19:16 DLC[3:0]	数据长度代码（Data Length Code） – 0-8：典型 CAN + CAN FD：接收到的帧包含 0-8 个数据字节 – 9-15：典型 CAN：接收到的帧包含 0-8 个数据字节 – 9-15：接收到的帧包含 12/16/20/24/32/48/64 个数据字节
R1 Bits 15:0 RXTS[15:0]	接收时间戳（Rx Timestamp） 开始接收帧时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 FDCAN_TSCC 寄存器 TCP[3:0]的配置。
R2 Bits 31:24 DB3[7:0]	数据字节 3（Data Byte 3）
R2 Bits 23:16 DB2[7:0]	数据字节 2（Data Byte 2）
R2 Bits 15:8 DB1[7:0]	数据字节 1（Data Byte 1）
R2 Bits 7:0 D[7:0]	数据字节 0（Data Byte 0）
R3 Bits 31:24 DB7[7:0]	数据字节 7（Data Byte 7）
R3 Bits 23:16 DB6[7:0]	数据字节 6（Data Byte 6）
R3 Bits 15:8 DB5[7:0]	数据字节 5（Data Byte 5）
R3 Bits 7:0 DB4[7:0]	数据字节 4（Data Byte 4）
⋮	⋮
Rn Bits 31:24	数据字节 m（Data Byte m）

位域	说明
DBm[7:0]	
Rn Bits 23:16 DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Rn Bits 15:8 DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Rn Bits 7:0 DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

26.3.9 FDCAN 发送缓冲区元素

发送缓冲区部分（包含三个元素）可配置为用于存储发送 FIFO 或发送队列。发送处理器通过 FDCAN_TXBC 寄存器中的发送缓冲区配置 TFQM 位来区分发送 FIFO 和发送队列。元素大小被配置为可存储数据长度最多达 64 字节的 CAN FD 消息。

表 26.7 发送缓冲区和 FIFO 元素

Bit	31 24	23 16	15 8	7 0
T0	ESI / XTD / RTR / ID[28:0]			
T1	MM[7:0]	EFC / Res. / FDF / BRS / DLC[3:0]	Res.	
T2	DB3[7:0]	DB2[7:0]	DB1[7:0]	D[7:0]
T3	DB7[7:0]	DB6[7:0]	DB5[7:0]	DB4[7:0]
⋮	⋮	⋮	⋮	
Tn	DBm[7:0]	DBm-1[7:0]	DBm-2[7:0]	DBm-3[7:0]

表 26.8 发送缓冲区元素说明

位域	说明
T0 Bit 31 ESI(1)	错误状态指示符（Error State Indicator） – 0：CAN FD 格式的 ESI 位仅取决于错误被动标志 – 1：CAN FD 格式的 ESI 位会进行隐性发送
T0 Bit 30 XTD	扩展标识符（Extended identifier） – 0：11 位标准标识符 – 1：29 位扩展标识符
T0 Bit 29 RTR(2)	远程发送请求（Remote Transmission Request） – 0：发送数据帧 – 1：发送远程帧
T0 Bits 28:0 ID[28:0]	标识符（Identifier） 标准标识符或扩展标识符取决于 XTD 位。标准标识符必须写入 ID[28:18]。
T1 Bits 31:24 MM[7:0]	消息标记（Message Marker） 在发送缓冲区配置期间由 CPU 写入。复制到发送事件 FIFO 元素中，用于标识发送消息状态。
T1 Bit 23 EFC	事件 FIFO 控制（Event FIFO Control） – 0：不存储发送事件 – 1：存储发送事件

位域	说明
T1 Bit 21 FDF	FD 格式（FD Format） – 0：以典型 CAN 格式发送帧 – 1：以 CAN FD 格式发送帧
T1 Bit 20 BRS(3)	比特率切换（Bit Rate Switching） – 0：发送 CAN FD 帧时不切换比特率 – 1：发送 CAN FD 帧时切换比特率
T1 Bits 19:16 DLC[3:0]	数据长度代码（Data Length Code） – 0 - 8：典型 CAN + CAN FD：接收到的帧包含 0-8 个数据字节 – 9 - 15：典型 CAN：接收到的帧包含 8 个数据字节 – 9 - 15：CAN FD：接收到的帧包含 12/16/20/24/32/48/64 个数据字节
T2 Bits 31:24 DB3[7:0]	数据字节 3（Data Byte 3）
T2 Bits 23:16 DB2[7:0]	数据字节 2（Data Byte 2）
T2 Bits 15:8 DB1[7:0]	数据字节 1（Data Byte 1）
T2 Bits 7:0 D[7:0]	数据字节 0（Data Byte 0）
T3 Bits 31:24 DB7[7:0]	数据字节 7（Data Byte 7）
T3 Bits 23:16 DB6[7:0]	数据字节 6（Data Byte 6）
T3 Bits 15:8 DB5[7:0]	数据字节 5（Data Byte 5）
T3 Bits 7:0 DB4[7:0]	数据字节 4（Data Byte 4）
⋮	⋮
Tn Bits 31:24 DBm[7:0]	数据字节 m（Data Byte m）
Tn Bits 23:16 DBm-1[7:0]	数据字节 m-1（Data Byte m-1）
Tn Bits 15:8 DBm-2[7:0]	数据字节 m-2（Data Byte m-2）
Tn Bits 7:0 DBm-3[7:0]	数据字节 m-3（Data Byte m-3）

1. 发送缓冲区的 ESI 位会与错误被动标志进行或运算，以确定已发送 FD 帧中 ESI 位的值。按照 CAN FD 协议规范的要求，错误主动节点可选择隐性发送 ESI 位，但错误被动节点将始终隐性发送 ESI 位。
2. 当 RTR = 1 时，即使 FDCAN_CCCR 寄存器的 FDOE 位使能以 CAN FD 格式进行发送，FDCAN 也会按照 ISO11898-1 的规定发送远程帧。
3. 仅当通过设置 FDCAN_CCCR 寄存器的 FDOE 位使能了 CAN FD 操作时，才会评估 ESI、FDF 和 BRS 位。仅当在 FDCAN_CCCR 寄存器中额外设置了 BRSE 位时，才会对 BRS 位进行评估。

26.3.10 FDCAN 发送事件 FIFO 元素

每个元素均存储与已发送消息相关的消息。通过读取发送事件 FIFO，主机 CPU 会按消息的发送顺序获取此类信息。关于发送事件 FIFO 的状态信息，可从寄存器 FDCAN_TXEFS 中获取。

表 26.9 发送事件 FIFO 元素

Bit	31			24	23				16	15		8	7		0
E0	ESI	XTD	RTR		ID[28:0]										
E1	MM[7:0]				ET[1:0]		EDL	BRS	DLC[3:0]	TXTS[15:0]					

表 26.10 发送事件 FIFO 元素说明

位域	说明
E0 Bit 31 ESI	错误状态指示符（Error State Indicator） – 0：发送节点为错误主动状态 – 1：发送节点为错误被动状态
E0 Bit 30 XTD	扩展标识符（Extended Identifier） – 0：11 位标准标识符 – 1：29 位扩展标识符
E0 Bit 29 RTR	远程发送请求（Remote Transmission Request） – 0：发送数据帧 – 1：发送远程帧
E0 Bits 28:0 ID[28:0]	标识符（Identifier） 标准标识符或扩展标识符取决于 XTD 位。标准标识符必须写入 ID[28:18]。
E1 Bits 31:24 MM[7:0]	消息标记（Message Marker） 从发送缓冲区复制到发送事件 FIFO 元素中，用于标识发送消息状态。
E1 Bits 23:22 EFC	事件类型（Event Type） – 00：保留 – 01：发送事件 – 10：取消后仍发送（在 DAR 模式下发送时始终设为该值） – 11：保留
E1 Bit 21 EDL	扩展数据长度（Extended Data Length） – 0：标准帧格式 – 1：FDCAN 帧格式（新 DLC 编码和 CRC）
E1 Bit 20 BRS	比特率切换（Bit Rate Switching） – 0：发送帧时不切换比特率 – 1：发送帧时切换比特率
T1 Bits 19:16 DLC[3:0]	数据长度代码（Data Length Code） 0 - 8：发送包含 0-8 个数据字节的帧 9 - 15：发送包含 8 个数据字节的帧
E1 Bits 15:0 TXTS[15:0]	发送时间戳（Tx Timestamp） 开始接收帧时捕获的时间戳计数器值。分辨率取决于时间戳计数器预分频器 FDCAN_TSCC 寄存器 TCP[3:0]的配置。

26.3.11 FDCAN 标准消息 ID 过滤器元素

最多可为 11 位标准 ID 配置 28 个过滤器元素。访问标准消息 ID 过滤器元素时，其地址为过滤器列表标准起始地址（FLSSA）加上过滤器元素的索引（0 ... 27）。

表 26.11 标准消息 ID 过滤器元素

Bit	31		24	23	16	15	8	7	0
S0	SFT[1:0]	SFEC[2:0]		SFID1[10:0]		Res.		SFID2[10:0]	

表 26.12 标准消息 ID 过滤器元素位域说明

位域	说明
Bit 31:30 SFT[1:0](1)	标准过滤器类型（Standard Filter Type） – 00：从 SFID1 到 SFID2 的范围过滤器 – 01：用于 SFID1 或 SFID2 的双 ID 过滤器 – 10：典型过滤器：SFID1 = 过滤器，SFID2 = 掩码 – 11：禁止过滤器元素
Bit 29:27 SFEC[2:0]	标准过滤器元素配置（Standard Filter Element Configuration） 所有使能的过滤器元素都用于对标准帧进行验收过滤。当发现第一个匹配的已使能过滤器元素时，或者到达过滤器列表结尾处时，会停止验收过滤。如果 SFEC[2:0] = 100，101 或 110，则出现匹配时，会将中断标志 IR.HPM 置 1，并会生成中断（若使能）。在这种情况下，寄存器 HPMS 会更新为优先级匹配的状态。 – 000：禁止过滤器元素 – 001：如果过滤器匹配，则存储在接收 FIFO 0 中 – 010：如果过滤器匹配，则存储在接收 FIFO 1 中 – 011：如果过滤器匹配，则拒绝 ID – 100：如果过滤器匹配，则设置优先级 – 101：如果过滤器匹配，则设置优先级并存储在 FIFO 0 中 – 110：如果过滤器匹配，则设置优先级并存储在 FIFO 1 中 – 111：未使用
Bits 26:16 SFID1[10:0]	标准过滤器 ID 1（Standard Filter ID 1） 标准 ID 过滤器元素的第一个 ID。
Bits 10:0 SFID2[10:0]	标准过滤器 ID 2（Standard Filter ID 2） 标准 ID 过滤器元素的第二个 ID。

1. 当 SFT[1:0] = 11 时，会禁止过滤器元素，验收过滤会继续进行（与 SFEC[2:0] = 000 时的行为相同）。

注：如果配置了保留值，则会将过滤器元素视为已禁止。

26.3.12 FDCAN 扩展消息 ID 过滤器元素

最多可为 29 位扩展 ID 配置 8 个过滤器元素。访问扩展消息 ID 过滤器元素时，其地址为过滤器列表扩展起始地址（FLESA）加上过滤器元素的索引（0...7）的二倍。

表 26.13 扩展消息 ID 过滤器元素

Bit	31		24	23 16	15 8	7 0
F0	EFEC[2:0]		EFID1[28:0]			
F1	EFTI[1:0]	Res.	EFID2[28:0]			

表 26.14 扩展消息 ID 过滤器元素位域说明

位域	说明
F0 Bits 31:29 EFEC[2:0]	扩展过滤器元素配置（Extended Filter Element Configuration） 所有使能的过滤器元素都用于对扩展帧进行验收过滤。当发现第一个匹配的已使能过滤器元素时，或者到达过滤器列表结尾处时，会停止验收过滤。如果 EFEC[2:0] = 100, 101 或 110，则出现匹配时，会将中断标志 IR[HPM]置 1，并会生成中断（若使能）。在这种情况下，寄存器 HPMS 会更新为优先级匹配的状态。 – 000：禁止过滤器元素 – 001：如果过滤器匹配，则存储在接收 FIFO 0 中 – 010：如果过滤器匹配，则存储在接收 FIFO 1 中 – 011：如果过滤器匹配，则拒绝 ID – 100：如果过滤器匹配，则设置优先级 – 101：如果过滤器匹配，则设置优先级并存储在 FIFO 0 中 – 110：如果过滤器匹配，则设置优先级并存储在 FIFO 1 中 – 111：未使用
F0 Bits 28:0 EFID1[28:0]	扩展过滤器 ID 1（Extended Filter ID 1） 扩展 ID 过滤器元素的第一个 ID。 过滤接收缓冲区或调试消息时，此位域定义要存储的扩展消息的 ID。接收的标识符必须完全匹配，仅使用 XIDAM 屏蔽机制。
F1 Bits 31:30 EFT[1:0]	扩展过滤器类型（Extended Filter Type） – 00：从 EF1ID 到 EF2ID 的范围过滤器（EF2ID ≥ EF1ID） – 01：用于 EF1ID 或 EF2ID 的双 ID 过滤器 – 10：典型过滤器：EF1ID = 过滤器，EF2ID = 掩码 – 11：从 EF1ID 到 EF2ID 的范围过滤器（EF2ID ≥ EF1ID），未应用 XIDAM 掩码
F1 Bit 29	未使用
F1 Bits 28:0 EFID2[28:0]	扩展过滤器 ID 2（Extended Filter ID 2） 扩展 ID 过滤器元素的第二个 ID。

26.4 FDCAN 寄存器

26.4.1 FDCAN 内核释放寄存器（FDCAN_CREL）

地址偏移：0x0000

复位值：0x3214 1218

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL[3:0]				STEP[3:0]				SUBSTEP[3:0]				YEAR[3:0]			
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
MON[7:0]								DAY[7:0]							
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:28 **REL[3:0]**：内核释放 3

Bit 27:24 **STEP[3:0]**：内核释放步 2

Bit 23:20 **SUBSTEP[3:0]** ：内核释放子步 1

Bit 19:16 **YEAR[3:0]**：时间戳年 4

Bit 15:8 **MON[7:0]**：时间戳月 12

Bit 7:0 **DAY[7:0]**：时间戳日 18

26.4.2 FDCAN 字节序寄存器（FDCAN_ENDN）

地址偏移：0x0004

复位值：0x8765 4321

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

Bit 31:0 **ETV[31:0]**：字节序测试值

字节序测试值为 0x8765 4321。

注：读取该寄存器时必须得到复位值，以确保不存在字节序问题。

26.4.3 FDCAN 数据位定时和预分频器寄存器（FDCAN_DBTP）

地址偏移：0x000C
复位值：0x0000 0A33

仅当 FDCAN_CCCR 寄存器中的 CCE 位和 INIT 位置 1 时，才可以向此寄存器写入数据。CAN 时间片的可编程范围为 1 到 32 个 FDCAN 时钟周期：

t_q =（DBRP[4:0] + 1）个 FDCAN 时钟周期。
DTSEG1[4:0]为 PROP_SEG 与 PHASE_SEG1 之和。DTSEG2[3:0]为 PHASE_SEG2。因此，位时间的长度为：
（编程值）× [DTSEG1[4:0] + DTSEG2[3:0] + 3] × t_q，或
（函数值）× [SYNC_SEG + PROP_SEG + PHASE_SEG1 + PHASE_SEG2] × t_q。

信息处理时间（IPT）为 0，这意味着下一位的数据在采样点后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TDC	Reserved		DBRP[4:0]				
								rw			rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DTSEG1[4:0]					DTSEG2[3:0]				DSJW[3:0]			
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:24 保留，必须保持为复位值。

Bit 23 **TDC**：收发器延迟补偿
0：禁用收发器延迟补偿
1：使能收发器延迟补偿

Bit 22:21 保留，必须保持为复位值。

Bit 20:16 **DBRP[4:0]**：数据比特率预分频器
生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。比特率预分频器的有效值为 0 到 31。硬件将该值解析为编程值加 1。

Bit 15:13 保留，必须保持为复位值。

Bit 12:8 **DTSEG1[4:0]**：采样点之前的数据时间段
有效值为 0 至 31。硬件使用的值为编程值加 1，即 t_{BS1} = (DTSEG1[4:0] + 1) × t_q。

Bit 7:4 **DTSEG2[3:0]**：采样点之后的数据时间段
有效值为 0 至 15。硬件使用的值为编程值加 1，即 t_{BS2} = (DTSEG2[3:0] + 1) × t_q。

Bit 3:0 **DSJW[3:0]**：同步跳转宽度
有效值为 0 至 15。硬件使用的值为编程值加 1：t_{SJW} = (DSJW[3:0] + 1) × tq。

注：如果 FDCAN 时钟为 8 MHz，复位值 0x00000A33 会将 FDCAN 的比特率配置为较快的 500 kbit/s。

数据相位位速率必须高于或等于标称位速率。

26.4.4 FDCAN 测试寄存器（FDCAN_TEST）

必须通过将 FDCAN_CCCR 寄存器中的 TEST 位置 1 的方式使能对测试寄存器的写访问。当该位被清零时，所有测试寄存器功能都会设为其复位值。

接收引脚 FDCANx_TX 的环回模式和软件控制属于硬件测试模式。
将 TX[1:0]编程为非 00 值，可能会干扰 CAN 总线上的消息传输。

地址偏移：0x0010
复位值：0x0000 0000

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

Bit 31:8 保留，必须保持为复位值。

Bit 7 **RX：**接收引脚

监控引脚 FDCANx_RX 的实际值
0：CAN 总线处于显性状态（FDCANx_RX = 0）
1：CAN 总线处于隐性状态（FDCANx_RX = 1）

Bit 6:5 **TX[1:0]：发**送引脚控制

00：复位值，FDCANx_TX 由 CAN 内核控制，在 CAN 位时间结束时更新
01：可在引脚 FDCANx_TX 处监控采样点
10：引脚 FDCANx_TX 处于显性（0）电平
11：引脚 FDCANx_TX 处于隐性（1）电平

Bit 4 **LBCK：**环回模式

0：复位值，环回模式禁用
1：使能环回模式（参见掉电（休眠模式））

Bit 3:0 保留，必须保持为复位值。

26.4.5 FDCAN RAM 看门狗寄存器（FDCAN_RWD）

RAM 看门狗监视消息 RAM 的 READY 输出。对消息 RAM 进行访问会启动消息 RAM 看门狗计数器，计数器的起始值由 FDCAN_RWD 寄存器的 WDC[7:0]位域配置。

当消息 RAM 通过激活其 READY 输出指示访问成功完成时，计数器会重新载入 WDC[7:0]的值。如果计数器递减为 0 之前消息 RAM 没有发出响应，则计数器会停止计数，中断标志 FDCAN_IR 寄存器中的 WDI 位置 1。RAM 看门狗计数器由 fdcan_pclk 时钟驱动。

地址偏移：0x0014
复位值：0x0000 0000

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:8 **WDV[7:0]：**看门狗值

实际的消息 RAM 看门狗计数器值。

Bit 7:0 **WDC[7:0]：**看门狗配置

消息 RAM 看门狗计数器的起始值。复位值为 00 时，计数器被禁用。

该位域受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置 1 时，才允许进行写访问。

26.4.6 FDCAN CC 控制寄存器（FDCAN_CCCR）

地址偏移：0x0018

复位值：0x0000 0001

有关各个位置 1 和复位的详细信息，请参见软件初始化。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NISO	TXP	EFBI	PXHD	Reserved		BRSE	FDOE	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
rw	rw	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:16 保留，必须保持为复位值。

Bit 15 **NISO：**非 ISO 模式操作

如果此位置 1，FDCAN 会使用 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式。

0：符合 ISO11898-1 规定的 CAN FD 帧格式

1：符合 Bosch CAN FD 规范 V1.0 规定的 CAN FD 帧格式

Bit 14 **TXP：**发送暂停使能

如果此位置 1，成功发送帧后，FDCAN 会先暂停两个 CAN 位时间，然后再开始进行下一次发送。

0：禁用

1：使能

Bit 13 **EFBI：**总线同步期间的边沿过滤

0：禁用边沿过滤

1：需要两个连续显性 t_q 才能检测硬同步边沿

Bit 12 **PXHD：**协议异常处理禁用

0：使能协议异常处理

1：禁用协议异常处理

Bit 11:10 保留，必须保持为复位值。

Bit 9 **BRSE**：FDCAN 比特率切换

0：禁用发送时进行比特率切换

1：使能发送时进行比特率切换

Bit 8 **FDOE**：FD 操作使能

0：禁用 FD 操作

1：使能 FD 操作

Bit 7 **TEST**：测试模式使能

0：正常操作，FDCAN_TEST 寄存器保持复位值

1：测试模式，允许对 FDCAN_TEST 寄存器进行写访问

Bit 6 **DAR**：禁用自动重发送

0：使能自动重发送未成功发送的消息

1：禁用自动重发送

Bit 5 **MON**：总线监控模式

此位仅当 CCE 和 INIT 同时置位时，才可通过软件设置。主机可随时清除该位。

0：禁用总线监控模式

1：使能总线监控模式

Bit 4 **CSR**：时钟停止请求

0：未请求时钟停止

1：请求时钟停止。当请求时钟停止时，在所有待处理传输请求完成且 CAN 总线空闲后，首先置位 INIT，然后置位 CSA。

Bit 3 **CSA**：时钟停止确认

0：未确认时钟停止

1：通过停止 APB 时钟和内核时钟，FDCAN 可进入掉电模式。

Bit 2 **ASM**：ASM 受限工作模式

受限工作模式可用于根据不同 CAN 比特率对自身进行调整的应用中。应用会测试不同比特率，并在收到有效帧后退出受限工作模式。在可选的受限工作模式下，节点能够发送和接收数据帧和远程帧，并能对有效帧进行确认，但不会发送有效错误帧或过载帧。如果存在错误条件或过载条件，则不会发送显性位，而会等待出现总线空闲条件，以便使自身与 CAN 通信重新同步。错误计数器不会递增。仅当 CCE 和 INIT 同时置位时，才能通过软件将 ASM 位置 1。此位可随时通过软件复位。

0：正常 CAN 操作

1：受限工作模式激活

Bit 1 **CCE**：配置更改使能

0：CPU 对受保护的配置寄存器无写访问权限。

1：CPU 对受保护的配置寄存器有写访问权限（当 FDCAN_CCCR 中的 INIT=1 时）。

Bit 0 **INIT**：初始化

0：正常工作

1：启动初始化

注：由于两个时钟域之间存在同步机制，可能会经过一定的延迟之后才能读取写入 INIT *的值。因此，编程人员必须确保之前写入* INIT *中的值已通过读取* INIT *的方式被接受，然后再将* INIT *设为新值。*

26.4.7 FDCAN 标称位定时和预分频器寄存器（FDCAN_NBTP）

地址偏移：0x001C
复位值：0x0600 0A03

仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位置 1 时，才可以向此寄存器写入数据。CAN 位时间的可编程范围为 4 到 81 个 t_q。CAN 时间片的可编程范围为 1 到 1024 个 FDCAN 内核时钟周期：
t_q = (BRP + 1) × FDCAN 时钟周期 fdcan_ker_ck。

NTSEG1[7:0]为 PROP_SEG 与 PHASE_SEG1 之和。NTSEG2[6:0]为 PHASE_SEG2。因此，位时间的长度为：
（编程值）× [NTSEG1[7:0] + NTSEG2[6:0] + 3] × t_q 或
（函数值）× [SYNC_SEG + PROP_SEG + PHASE_SEG1 + PHASE_SEG2] × t_q。
信息处理时间（IPT）为 0，这意味着下一位的数据在采样点后的第一个时钟边沿可用。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NSJW[6:0]							NBRP[8:0]								
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
NTSEG1[7:0]								Res	NTSEG2[6:0]						
rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Bit 31:25 **NSJW[6:0]：**标称（重新）同步跳转宽度
有效值为 0 到 127。实际上，硬件会将该值解析为编程值加 1。
该位域受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

Bit 24:16 **NBRP[8:0]：**比特率预分频器
生成位时间片时将振荡器频率除以的值。位时间为该时间片的倍数。有效值为 0 到 511。实际上，硬件会将该值解析为编程值加 1。
该位域受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

Bit 15:8 **NTSEG1[7:0]：**采样点之前的标称时间段
有效值为 1 到 255。实际上，硬件会将该值解析为编程值加 1。
该位域受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

Bit 7 保留，必须保持为复位值。

Bit 6:0 **NTSEG2[6:0]：**采样点之后的标称时间段
有效值为 0 到 127。实际上，硬件会将该值解析为编程值加 1。

注：如果 CAN *内核时钟为* 48MHz，*复位值* 0x0600 0A03 *会将* FDCAN *的比特率配置为* 3 Mbit/s。

26.4.8 FDCAN 时间戳计数器配置寄存器（FDCAN_TSCC）

地址偏移：0x0020

复位值：0x0000 0000

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

Bit 31:20 保留，必须保持为复位值。

Bit 19:16 **TCP[3:0]：**时间戳计数器预分频器

将时间戳和超时计数器时间单位配置为 CAN 位时间的倍数[1...16]。

实际上，硬件会将该值解析为编程值加 1。

在 CAN FD 模式下，由于仲裁阶段与数据阶段之间的 CAN 位时间不同，内部时间戳计数器 TCP 不会提供恒定的时基。因此，CAN FD 需要使用外部计数器生成时间戳（TSS[1:0] = 10）。

该位域受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

Bit 15:2 保留，必须保持为复位值。

Bit 1:0 **TSS[1:0]：**时间戳选择

00：时间戳计数器值始终为 0x0000

01：时间戳计数器值根据 TCP 递增

10：来自 TIM3 值（tim3_cnt[0:15]）的外部时间戳计数器

11：与 00 相同。

这些位受写保护（P）：仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

26.4.9 FDCAN 时间戳计数器值寄存器（FDCAN_TSCV）

地址偏移：0x0024

复位值：0x0000 0000

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **TSC[15:0]：**时间戳计数器

内部/外部时间戳计数器值是在（接收和发送）帧开始时捕获的。当 FDCAN_TSCC 中的 TSS[1:0]=01 时，时间戳计数器会以 CAN 为时间的倍数[1...16]递增，具体取决于 FDCAN_TSCC 中 TCP[3:0]的配置。回卷操作会将 FDCAN_IR 的中断标志 TSW 置 1。写访问会将计数器复位为 0。

当 TSS[1:0]=10 时，TSC[15:0]会反映外部时间戳计数器值。写访问没有影响。

注：“回卷”是指时间戳计数器值不是由于对 FDCAN_TSCV *进行写访问的原因而由非零值变为* 0。

26.4.10 FDCAN 超时计数器配置寄存器（FDCAN_TOCC）

地址偏移：0x0028

复位值：0xFFFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOP[15:0]															
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
Reserved													TOS[1:0]		ETOC
													rw	rw	rw

Bit 31:16 **TOP[15:0]：**超时时长

超时计数器（递减计数器）的起始值。用于配置超时时长。

Bit 15:3 保留，必须保持为复位值。

Bit 2:1 **TOS[1:0]：**超时选择

在连续模式下工作时，对 FDCAN_TOCV 进行写访问会将计数器预设为由 FDCAN_TOCC 中由 TOP[15:0]配置的值并继续递减计数。超时计数器由其中一个 FIFO 控制时，空 FIFO 会将计数器预设为由 TOP[15:0]配置的值。存储了第一个 FIFO 元素后，会开始递减计数。

00：连续工作

01：超时由发送事件 FIFO 控制

10：超时由接收 FIFO 0 控制

11：超时由接收 FIFO 1 控制

该位域受写保护（P），仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

Bit 0 **ETOC：**使能超时计数器

0：禁用超时计数器

1：使能超时计数器

该位受写保护（P），仅当 FDCAN_CCCR 寄存器的 CCE 位和 INIT 位同时置位时，才可进行写操作。

更多详情，请参阅超时计数器部分。

26.4.11 FDCAN 超时计数器值寄存器（FDCAN_TOCV）

地址偏移：0x002C

复位值：0x0000 FFFF

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:0 **TOC[15:0]：**超时计数器

超时计数器会以 CAN 位时间的倍数[1...16]递减，具体取决于 FDCAN_TSCC 寄存器中 TCP[3:0]位域的配置。当计数器递减至 0 时，FDCAN_IR 的中断标志 TOO 会置 1，超时计数器会停止计数。开始和复位/重启条件是通过 FDCAN_TOCC 中的 TOS[1:0]进行配置的。

26.4.12 FDCAN 错误计数器寄存器（FDCAN_ECR）

地址偏移：0x0040

复位值：0x0000 0000

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

Bit 31:24 保留，必须保持为复位值。

Bit 23:16 **CEL[7:0]：**CAN 错误记录

每当 CAN 协议错误导致发送错误计数器或接收错误计数器递增时，计数器都会递增。对 CEL[7:0]进行读访问时，计数器会复位。

计数器值达到 0xFF 时，会停止计数；TEC[7:0]或 REC[6:0]的下一次递增时，FDCAN_IR 的中断标志 ELO 会置 1。

访问类型为 rc_r：读取时清零。

Bit 15 **RP：**接收错误被动

0：接收错误计数器低于错误被动级别 128

1：接收错误计数器已达到错误被动级别 128

Bit 14:8 **REC[6:0]：**接收错误计数器

接收错误计数器的实际状态，取值范围为 0 至 127。

Bit 7:0 **TEC[7:0]：**发送错误计数器

发送错误计数器的实际状态，取值范围为 0 至 255。

当 FDCAN_CCCR 的 ASM 位置位时，CAN 协议控制器在检测到 CAN 协议错误时不会递增 TEC 和 REC，但 CEL[7:0]仍会递增。

26.4.13 FDCAN 协议状态寄存器（FDCAN_PSR）

地址偏移：0x0044

复位值：0x0000 0707

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									TDCV[6:0]						
									r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	PXE	REDL	RBRS	RESI	DLEC[2:0]			BO	EW	EP	ACT[1:0]		LEC[2:0]		
	rc_r	rc_r	rc_r	rc_r	rs	rs	rs	r	r	r	r	r	rs	rs	rs

Bit 31:23 保留，必须保持为复位值。

Bit 22:16 **TDCV[6:0]：**发送器延迟补偿值

第二采样点的位置，由从 FDCAN_TX 到 FDCAN_RX 之间测得的延迟与 FDCAN_TDCR 中的 TDCO[6:0]之和定义。在数据阶段，SSP 位置为已发送位起点与第二采样点之间的最小时间片（mt_q）数。有效值为 0 至 127×mt_q。

Bit 15 保留，必须保持为复位值。

Bit 14 **PXE：**协议异常事件

0：自上次读取访问以来未发生协议异常事件

1：已发生协议异常事件

Bit 13 **REDL：**已接收 FDCAN 消息

此位的设置与验收过滤无关。

0：由于此位是通过 CPU 复位的，因此尚未接收到 FDCAN 消息

1：已接收到 EDL 标志置 1 的 FDCAN 格式的消息

访问类型为 rc_r：读取时清零。

Bit 12 **RBRS：**上次接收的 FDCAN 消息的 BRS 标志

此位与 REDL 一起设置，与验收过滤无关。

0：上次接收的 FDCAN 消息的 BRS 标志未置 1

1：上次接收的 FDCAN 消息的 BRS 标志已置 1

访问类型为 rc_r：读取时清零。

Bit 11 **RESI：**上次接收的 FDCAN 消息的 ESI 标志

此位与 REDL 一起设置，与验收过滤无关。

0：上次接收的 FDCAN 消息的 ESI 标志未置 1

1：上次接收的 FDCAN 消息的 ESI 标志已置 1

访问类型为 rc_r：读取时清零。

Bit 10:8 **DLEC[2:0]：**数据上一错误代码

在 BRS 标志置 1 的 FDCAN 格式帧数据阶段发生的上一错误类型。代码与 LEC[2:0]相同。当 BRS 标志置 1 的 FDCAN 格式帧已无错传输（接收或发送）时，此位域将被清零。

访问类型为 rs：读取时置位。

Bit 7 **BO：**Bus_Off 状态

0：FDCAN 未处于 Bus_Off 状态。

1：FDCAN 处于 Bus_Off 状态。

Bit 6 **EW：**警告状态

0：两个错误计数器的值均小于 Error_Warning 限值 96
1：至少有一个错误计数器已达到 Error_Warning 限值 96

Bit 5 **EP**：错误被动
0：FDCAN 处于 Error_Active 状态。此位通常会参与总线通信，并会在检测到错误后发送主动错误标志
1：FDCAN 处于 Error_Passive 状态

Bit 4:3 **ACT[1:0]**：活动
监测模块的 CAN 通信状态。
00：同步中：节点在 CAN 通信时同步
01：空闲：节点既不是接收器，也不是发送器
10：接收器：节点作为接收器工作
11：发送器：节点作为发送器工作

Bit 2:0 **LEC[2:0]**：上一错误代码
LEC[2:0]指示 CAN 总线上发生的上一错误的类型。消息已无错传输后（接收或发送），此位域将被清零。
000：无错：自消息成功接收或发送而将 LEC[2:0]复位后，未发生任何错误。
001：内容错误：在已接收的消息中，有一部分连续出现 5 个以上的相等位，这种情况是不允许发生的。
010：格式错误：已接收帧的固定格式部分的格式不正确。
011：AckError。由 FDCAN 发送的消息未被另一节点确认。
100：Bit1Error。在消息发送过程中（仲裁字段例外），器件希望发送隐性电平（位逻辑值为 1），但受监控的总线值为显性。
101：Bit0Error。在消息发送过程中（或者确认位，或者主动错误标志，或者过载标志），器件希望发送显性电平（数据或标识符位逻辑值 0），但受监控的总线值为隐性。在 Bus_Off 恢复期间，每次监控到 11 个隐性位组成的序列时，此状态都会置 1。这样，CPU 便可监控 Bus_Off 恢复序列的进行（指示总线并未保持显性或持续受到干扰）。
110：CRCError。已接收消息的 CRC 校验和不正确。传入消息的 CRC 与通过已接收数据计算出的 CRC 不匹配。
111：NoChange。如果对协议状态寄存器进行读访问，则会将 LEC[2:0]重新初始化为 7。如果 LEC[2:0]显示的值为 7，则自 CPU 上一次对协议状态寄存器进行读访问起，未检测到任何 CAN 总线事件。
访问类型为 rs：读取时置位。

注：如果采用 FDCAN 格式的帧已到达 BRS 标志置 1 的数据阶段，下一 CAN 事件（错误或有效帧）将显示在 DLEC[2:0]中，而不显示在 LEC[2:0]中。FDCAN CRC 序列中的固定内容位中的错误将显示为格式错误，而不是内容错误。

Bus_Off 恢复序列（请参见 CAN 规范第 2.0 版或 ISO11898-1）不能通过将 FDCAN_CCCR 寄存器的 INIT 位置 1 或复位来缩短。如果器件进入 Bus_Off 状态，则会将其自身的 INIT 置 1，从而会停止所有总线活动。一旦 INIT 位由 CPU 清零，器件随后将等待总线空闲状态出现 129 次（129 × 11 个连续隐性位），然后才会恢复正常操作。Bus_Off 恢复序列结束时，错误管理计数器将复位。

在 INIT 位复位后的等待时间内，每次监控到由 11 个隐性位组成的序列时，Bit0 错误代码都会写入

FDCAN_PSR *的*LEC[2:0]*中，从而使*CPU *能够准备好检查*CAN *总线保持显性还是持续受到干扰，并能够监控*Bus_Off *恢复序列。*

FDCAN_ECR *寄存器的*REC[6:0]*位域用于对这些序列进行计数。*

26.4.14 FDCAN 发送器延迟补偿寄存器（FDCAN_TDCR）

地址偏移：0x0048

复位值：0x0000 0000

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

Bit 31:15 保留，必须保持为复位值。

Bit 14:8 **TDCO[6:0]：**发送器延迟补偿偏移量

定义从 FDCAN_TX 到 FDCAN_RX 测得的延迟与第二采样点之间距离的偏移值。有效值为 0 至 127 个 mt_q。

该位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均被置位时，才可进行写访问。

Bit 7 保留，必须保持为复位值。

Bit 6:0 **TDCF[6:0]：**发送器延迟补偿过滤器窗口长度

定义 SSP 位置的最小值，要测量发送器延迟，会忽略 FDCAN_RX 上会导致 SSP 位置提前的显性边沿。

该位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均被置位时，才可进行写访问。

26.4.15 FDCAN 中断寄存器（FDCAN_IR）

如果检测到其中一个列出的条件（边沿有效），标志会置 1。在主机将标志清零之前，标志保持置 1 状态。通过向对应位位置写入 1 将标志清零。

写入 0 则不会带来任何影响。硬复位会将寄存器清零。FDCAN_IE 的配置控制着是否生成中断。FDCAN_ILS 的配置控制着是哪条中断线发出中断。

地址偏移：0x0050

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ARA	PED	PEA	WDI	BO	EW	EP	ELO
								rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOO	MRAF	TSW	TEFL	TEFF	TEFN	TFE	TCF	TC	HPM	RF1L	RF1F	RF1N	RF0L	RF0F	RF0N
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bit 31:24 保留，必须保持为复位值。

Bit 23 **ARA**：访问保留地址

0：未对保留地址进行访问

1：已对保留地址进行访问

Bit 22 **PED**：数据阶段的协议错误（使用数据位时间）

0：数据阶段无协议错误

1：检测到数据阶段有协议错误（FDCAN_PSR 中的 DLEC[2:0]不为 0 和 7）

Bit 21 **PEA**：仲裁阶段的协议错误（使用标称位时间）

0：仲裁阶段无协议错误

1：检测到仲裁阶段有协议错误（FDCAN_PSR 中的 LEC[2:0]不为 0 和 7）

Bit 20 **WDI**：看门狗中断

0：未发生消息 RAM 看门狗事件

1：由于缺少 READY 信号而触发消息 RAM 看门狗事件

Bit 19 **BO**：Bus_Off 状态

0：Bus_Off 状态未改变

1：Bus_Off 状态已改变

Bit 18 **EW**：警告状态

0：Error_Warning 状态未改变

1：Error_Warning 状态已改变

Bit 17 **EP**：错误被动状态

0：Error_Passive 状态未改变

1：Error_Passive 状态已改变

Bit 16 **ELO**：错误记录溢出

0：CAN 错误记录计数器未溢出

1：CAN 错误记录计数器已溢出

Bit 15 **TOO**：超时发生

0：未超时

1：达到超时

Bit 14 **MRAF**：消息 RAM 访问失败

当接收处理器：

- 如果接收处理单元未在已接收以下消息的仲裁字段之前完成对已接收消息的验收过滤或存储时，该标志置位。在这种情况下，验收过滤或消息存储会中止，接收处理单元会开始处理以下消息。
- 无法将消息写入消息 RAM 时，该标志置位。在此情况下，消息存储被中止。

在这两种情况下，FIFO 放入索引不会更新，专用接收缓冲区的新数据标志也不会置 1。下一条消息存储到此位置时，部分存储的消息会被覆盖。

当发送处理器无法及时从消息 RAM 中读取消息时，该标志也会置位。在此情况下，消息传输被中止。若发送处理器访问失败，FDCAN 将切换至受限工作模式（见受限工作模式）。要退出受限操作模式，主机 CPU 必须清除 FDCAN_CCCR 寄存器的 ASM 位。

0：未发生消息 RAM 访问失败

1：发生了消息 RAM 访问失败

Bit 13 **TSW**：时间戳回卷
0：时间戳计数器未回卷
1：时间戳计数器已回卷
Bit 12 **TEFL**：发送事件 FIFO 元素丢失
0：发送事件 FIFO 元素未丢失
1：发送事件 FIFO 元素已丢失
Bit 11 **TEFF**：发送事件 FIFO 已满
0：发送事件 FIFO 未满
1：发送事件 FIFO 已满
Bit 10 **TEFN**：发送事件 FIFO 新条目
0：发送事件 FIFO 未改变
1：发送处理单元写入发送事件 FIFO 元素
Bit 9 **TFE**：发送 FIFO 为空
0：发送 FIFO 非空
1：发送 FIFO 为空
Bit 8 **TCF**：发送取消完成
0：发送取消未完成
1：发送取消已完成
Bit 7 **TC**：发送完成
0：发送未完成
1：发送已完成
Bit 6 **HPM**：高优先级消息
0：未接收到高优先级消息
1：接收到高优先级消息
Bit 5 **RF1L**：接收 FIFO 1 消息丢失
0：接收 FIFO 1 消息未丢失
1：接收 FIFO 1 消息已丢失
Bit 4 **RF1F**：接收 FIFO 1 已满
0：接收 FIFO 1 未满
1：接收 FIFO 1 已满
Bit 3 **RF1N**：接收 FIFO 1 新消息
0：未向接收 FIFO 1 写入新消息
1：向接收 FIFO 1 写入了新消息
Bit 2 **RF0L**：接收 FIFO 0 消息丢失
0：接收 FIFO 0 消息未丢失
1：接收 FIFO 0 消息已丢失
Bit 1 **RF0F**：接收 FIFO 0 已满
0：接收 FIFO 0 未满
1：接收 FIFO 0 已满
Bit 0 **RF0N**：接收 FIFO 0 新消息
0：未向接收 FIFO 0 写入新消息
1：向接收 FIFO 0 写入了新消息

26.4.16 FDCAN 中断使能寄存器（FDCAN_IE）

中断使能寄存器中的设置决定了将在中断线上指示中断寄存器中的哪些状态更改。

地址偏移：0x0054

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ARAE	PEDE	PEAE	WDIE	BOE	EWE	EPE	ELOE
								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOE	MRAFE	TSWE	TEFLE	TEFFE	TEFNE	TFEE	TCFE	TCE	HPME	RF1LE	RF1FE	RF1NE	RF0LE	RF0FE	RF0NE
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 31:24 保留，必须保持为复位值。

Bit 23 **ARAE**：对保留地址访问使能

Bit 22 **PEDE**：数据阶段协议错误使能

Bit 21 **PEAE**：仲裁阶段协议错误使能

Bit 20 **WDIE**：看门狗中断使能

0：禁用中断

1：使能中断

Bit 19 **BOE**：Bus_Off 状态

0：禁用中断

1：使能中断

Bit 18 **EWE**：警告状态中断使能

0：禁用中断

1：使能中断

Bit 17 **EPE**：错误被动中断使能

0：禁用中断

1：使能中断

Bit 16 **ELOE**：错误记录溢出中断使能

0：禁用中断

1：使能中断

Bit 15 **TOOE**：超时发生中断使能

0：禁用中断

1：使能中断

Bit 14 **MRAFE**：消息 RAM 访问失败中断使能

0：禁用中断

1：使能中断

Bit 13 **TSWE**：时间戳回卷中断使能

0：禁用中断

1：使能中断

Bit 12 **TEFLE**：发送事件 FIFO 元素丢失中断使能
0：禁用中断
1：使能中断

Bit 11 **TEFFE**：发送事件 FIFO 已满中断使能
0：禁用中断
1：使能中断

Bit 10 **TEFNE**：发送事件 FIFO 新条目中断使能
0：禁用中断
1：使能中断

Bit 9 **TFEE**：发送 FIFO 为空中断使能
0：禁用中断
1：使能中断

Bit 8 **TCFE**：发送取消完成中断使能
0：禁用中断
1：使能中断

Bit 7 **TCE**：发送完成中断使能
0：禁用中断
1：使能中断

Bit 6 **HPME**：高优先级消息中断使能
0：禁用中断
1：使能中断

Bit 5 **RF1LE**：接收 FIFO 1 消息丢失中断使能
0：禁用中断
1：使能中断

Bit 4 **RF1FE**：接收 FIFO 1 已满中断使能
0：禁用中断
1：使能中断

Bit 3 **RF1NE**：接收 FIFO 1 新消息中断使能
0：禁用中断
1：使能中断

Bit 2 **RF0LE**：接收 FIFO 0 消息丢失中断使能
0：禁用中断
1：使能中断

Bit 1 **RF0FE**：接收 FIFO 0 已满中断使能
0：禁用中断
1：使能中断

Bit 0 **RF0NE**：接收 FIFO 0 新消息中断使能
0：禁用中断
1：使能中断

26.4.17 FDCAN 中断线选择寄存器（FDCAN_ILS）

中断线选择寄存器将特定中断标志生成的中断从中断寄存器分配到两个模块中断线之一。要生成中断，必须通过 FDCAN_ILE 寄存器的 EINT0 和 EINT1 位使能相应的中断线。

地址偏移：0x0058

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									PERR	BERR	MISC	TFERR	SMSG	RXFIFO1	RXFIFO0
									rw	rw	rw	rw	rw	rw	rw

Bit 31:7 保留，必须保持为复位值。

Bit 6 **PERR**：协议错误分组（包含以下中断）

ARAL：访问保留地址线

PEDL：数据阶段的协议错误线

PEAL：仲裁阶段的协议错误线

WDIL：看门狗中断线

BOL：Bus_Off 状态

EWL：警告状态中断线

Bit 5 **BERR**：位和线路错误分组（包含以下中断）

EPL：错误被动中断线

ELOL：错误记录溢出中断线

Bit 4 **MISC**：中断重新分组（包含以下中断）

TOOL：超时已发生中断线

MRAFL：消息 RAM 访问失败中断线

TSWL：时间戳回卷中断线

Bit 3 **TFERR**：发送 FIFO 错误分组（包含以下中断）

TEFLL：发送事件 FIFO 元素丢失中断线

TEFFL：发送事件 FIFO 已满中断线

TEFNL：发送事件 FIFO 新条目中断线

TFEL：发送 FIFO 为空中断线

Bit 2 **SMSG**：状态消息位分组（包含以下中断）

TCFL：发送取消完成中断线

TCL：发送完成中断线

HPML：高优先级消息中断线

Bit 1 **RXFIFO1**：接收 FIFO1 位分组（包含以下中断）

RF1LL：接收 FIFO 1 消息丢失中断线

RF1FL：接收 FIFO 1 已满中断线

RF1NL：接收 FIFO 1 新消息中断线

Bit 0 **RXFIFO0：**接收 FIFO0 位分组（包含以下中断）

RF0LL：接收 FIFO 0 消息丢失中断线

RF0FL：接收 FIFO 0 已满中断线

RF0NL：接收 FIFO 0 新消息中断线

26.4.18 FDCAN 中断线使能寄存器（FDCAN_ILE）

通过对位 EINT0 和 EINT1 进行编程，可分别使能/禁止两条连接至 CPU 的中断线。

地址偏移：0x005C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														EINT1	EINT0
														rw	rw

Bit 31:2 保留，必须保持为复位值。

Bit 1 **EINT1：**使能中断线 1

0：禁用中断线 fdcan_intr0_it

1：使能中断线 fdcan_intr0_it

Bit 0 **EINT0：**使能中断线 0

0：禁用中断线 fdcan_intr1_it

1：使能中断线 fdcan_intr1_it

26.4.19 FDCAN 全局过滤器配置寄存器（FDCAN_RXGFC）

用于消息 ID 过滤的全局设置。全局过滤器配置控制着标准和扩展消息的过滤器路径，请具体如图 27.8 标准消息 ID 过滤路径和图 27.9 扩展消息 ID 过滤路径所述。

地址偏移：0x0080

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				LSE[3:0]				Reserved			LSS[4:0]				
				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
Reserved						F0OM	F1OM	Reserved		ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
						rw	rw			rw	rw	rw	rw	rw	rw

Bit 31:28 保留，必须保持为复位值。

Bit 27:24 **LSE[3:0]：**列表中扩展过滤器元素的数量

0：无扩展消息 ID 过滤器

1 至 8：扩展消息 ID 过滤器元素的数量

8：大于 8 的值视为 8

此位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 23:21 保留，必须保持为复位值

Bit 20:16 **LSS[4:0]：**列表中标准过滤器元素的数量

0：无标准消息 ID 过滤器

1 至 28：标准消息 ID 过滤器元素的数量

28：大于 28 的值视为 28

此位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可通过这些位进行写操作

Bit 15:10 保留，必须保持为复位值

Bit 9 **F0OM：**FIFO 0 操作模式（覆盖或阻止）

此位受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 8 **F1OM：**FIFO 1 操作模式（覆盖或阻止）

此位受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 7:6 保留，必须保持为复位值

Bit 5:4 **ANFS[1:0]：**接受非匹配标准帧

定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 11 位的已接收消息。

00：在接收 FIFO 0 中接受

01：在接收 FIFO 1 中接受

10：拒绝

11：拒绝

此位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 3:2 **ANFE[1:0]：**接受非匹配扩展帧

定义了如何处理与过滤器列表的任何元素都不匹配的 ID 为 29 位的已接收消息。

00：在 Rx FIFO 0 中接受

01：在 Rx FIFO 1 中接受

10：拒绝

11：拒绝

此位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 1 **RRFS：**拒绝标准远程帧

0：过滤采用 11 位标准 ID 的远程帧

1：拒绝所有采用 11 位标准 ID 的远程帧

此位受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均置 1 时，才可进行写操作

Bit 0 **RRFE：**拒绝扩展远程帧

0：过滤采用 29 位标准 ID 的远程帧

1：拒绝所有采用 29 位标准 ID 的远程帧

此位受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均被置位时，才可进行写操作

26.4.20 FDCAN 扩展 ID 和掩码寄存器（FDCAN_XIDAM）

地址偏移：0x0084

复位值：0x1FFF FFFF

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

Bit 31:29 保留，必须保持为复位值。

Bit 28:0 **EIDM[28:0]：**扩展 ID 掩码

要对扩展帧进行验收过滤，会将扩展 ID 与掩码与已接收帧的消息 ID 进行与运算。用于屏蔽 SAE J1939 中的 29 位 ID。如果所有位的复位值均设为 1，则掩码无效。

此位域受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均被置位时，才可进行写操作。

26.4.21 FDCAN 高优先级消息状态寄存器（FDCAN_HPMS）

每次消息 ID 过滤器元素配置为生成优先级事件匹配时，都会更新此寄存器。此寄存器可用于监控传入高优先级消息的状态，并可使能对这些消息的快速访问。

地址偏移：0x0088

复位值：0x0000 0000

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

Bit 31:16 保留，必须保持为复位值。

Bit 15 **FLST：**过滤器列表

指示匹配过滤器元素的过滤器列表。:

0：标准过滤器列表

1：扩展过滤器列表

Bit 14:13 保留，必须保持为复位值。

Bit 12:8 **FIDX[4:0]：**过滤器索引

匹配过滤器元素的索引。

范围：0 至 FDCAN_RXGFC 中 LSS[4:0] - 1 或 LSE[3:0] - 1。

Bit 7:6 **MSI[1:0]：**消息存储指示符

00：未选择 FIFO

01：FIFO 溢出

10：消息存储在 FIFO 0 中

11：消息存储在 FIFO 1 中

Bit 5:3 保留，必须保持为复位值。

Bit 2:0 **BIDX[2:0]：**缓冲区索引

消息存储到的接收 FIFO 元素的索引。仅当 MSI[1] = 1 时有效。

26.4.22 FDCAN 接收 FIFO 0 状态寄存器（FDCAN_RXF0S）

地址偏移：0x0090

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						RF0L	F0F	Reserved						F0PI[1:0]	
						r	r							r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						F0GI[1:0]		Reserved				F0FL[3:0]			
						r	r					r	r	r	r

Bit 31:26 保留，必须保持为复位值。

Bit 25 **RF0L：**接收 FIFO 0 消息丢失

此位是 FDCAN_IR 寄存器中 RF0L 中断标志的副本。当 RF0L 复位时，此位也会复位。

0：接收 FIFO 0 消息未丢失

1：接收 FIFO 0 消息已丢失，尝试向大小为零的接收 FIFO 0 进行写入后也会置 1

Bit 24 **F0F：**接收 FIFO 0 已满

0：接收 FIFO 0 未满

1：接收 FIFO 0 已满

Bit 23:18 保留，必须保持为复位值。

Bit 17:16 **F0PI[1:0]：**接收 FIFO 0 放入索引

接收 FIFO 0 写入索引指针。

范围：0 至 2。

Bit 15:10 保留，必须保持为复位值。

Bit 9:8 **F0GI[1:0]：**接收 FIFO 0 获取索引

接收 FIFO 0 的读取索引指针。

范围：0 至 2。

Bit 7:4 保留，必须保持为复位值。

Bit 3:0 **F0FL[3:0]：**接收 FIFO 0 填充级别

接收 FIFO 0 中存储的元素数量。

范围：0 至 3。

26.4.23 FDCAN 接收 FIFO 0 确认寄存器（FDCAN_RXF0A）

地址偏移：0x0094

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													F0AI[2:0]		
													rw	rw	rw

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **F0AI[2:0]：**接收 FIFO 0 确认索引

主机从接收 FIFO 0 读取消息或消息序列后，必须将从接收 FIFO 0 读取的最后一个元素的缓冲区索引写入 F0AI[2:0]中。此操作会将接收 FIFO 0 获取索引（FDCAN_RXF0S 中的 F0GI[1:0]）设为 F0AI[2:0] + 1，并会更新 FIFO 0 填充级别（FDCAN_RXF0S 中的 F0FL[3:0]）。

26.4.24 FDCAN 接收 FIFO 1 状态寄存器（FDCAN_RXF1S）

地址偏移：0x0098

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						RF1L	F1F	Reserved						F1PI[1:0]	
						r	r							r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						F1GI[1:0]		Reserved				F1FL[3:0]			
						r	r					r	r	r	r

Bit 31:26 保留，必须保持为复位值。

Bit 25 **RF1L：**接收 FIFO 1 消息丢失

此位是 FDCAN_IR 寄存器中 RF1L 中断标志的副本。当 RF1L 复位时，此位也会复位。

0：接收 FIFO 1 消息未丢失

1：接收 FIFO 1 消息已丢失，尝试向大小为零的接收 FIFO 1 进行写入后也会置 1

Bit 24 **F1F：**接收 FIFO 1 已满

0：接收 FIFO 1 未满

1：接收 FIFO 1 已满

Bit 23:18 保留，必须保持为复位值。

Bit 17:16 **F1PI[1:0]：**接收 FIFO 1 放入索引

接收 FIFO 1 写入索引指针。

范围：0 至 2。

Bit 15:10 保留，必须保持为复位值。

Bit 9:8 **F1GI[1:0]：**接收 FIFO 1 获取索引

接收 FIFO 1 读取索引指针。

范围：0 至 2。

Bit 7:4 保留，必须保持为复位值。

Bit 3:0 **F1FL[3:0]：**接收 FIFO 1 填充级别

接收 FIFO 1 中存储的元素数量。

范围：0 至 3。

26.4.25 FDCAN 接收 FIFO 1 确认寄存器（FDCAN_RXF1A）

地址偏移：0x009C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													F1AI[2:0]		
													rw	rw	rw

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **F1AI[2:0]：**接收 FIFO 1 确认索引

主机从接收 FIFO 1 读取消息或消息序列后，必须将从接收 FIFO 1 读取的最后一个元素的缓冲区索引写入 F1AI[2:0]中。此操作会将接收 FIFO 1 获取索引（FDCAN_RXF1S 中的 F1GI[1:0]）设为 F1AI[2:0] + 1，并会更新 FIFO 1 填充级别（FDCAN_RXF1S 中的 F1FL[3:0]）。

26.4.26 FDCAN 发送缓冲区配置寄存器（FDCAN_TXBC）

地址偏移：0x00C0

复位值：0x0000 0000

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

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

Bit 31:25 保留，必须保持为复位值。

Bit 24 **TFQM：**发送 FIFO/队列模式

0：发送 FIFO 操作

1：发送队列操作。

此位受写保护（P），即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位均被置位时，才可进行写操作。

Bit 23:0 保留，必须保持为复位值。

26.4.27 FDCAN 发送 FIFO/队列状态寄存器（FDCAN_TXFQS）

发送 FIFO/队列状态与寄存器 FDCAN_TXBRP 中列出的挂起发送请求相关。因此，添加/取消请求的作用可能因正在运行发送扫描而延迟（FDCAN_TXBRP 尚未更新）。

地址偏移：0x00C4

复位值：0x0000 0003

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										TFQF	Reserved			TFQPI[1:0]	
										r				r	r

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

Bit 31:22 保留，必须保持为复位值。

Bit 21 **TFQF**：发送 FIFO/队列已满

0：发送 FIFO/队列未满

1：发送 FIFO/队列已满

Bit 20:18 保留，必须保持为复位值。

Bit 17:16 **TFQPI[1:0]**：发送 FIFO/队列放入索引

发送 FIFO/队列写入索引指针，范围 0 至 3

Bit 15:10 保留，必须保持为复位值。

Bit 9:8 **TFGI[1:0]**：发送 FIFO 获取索引

发送 FIFO 读取索引指针，范围 0 至 3。如果已配置发送队列操作时（FDCAN_TXBC 中 TFQM = 1），则读为零。

Bit 7:3 保留，必须保持为复位值。

Bit 2:0 **TFFL[2:0]**：发送 FIFO 空闲级别

从 TFGI 开始的连续空闲发送 FIFO 元素数，范围为 0 到 3。如果已配置发送队列操作（FDCAN_TXBC 中 TFQM = 1），则读为零。

26.4.28 FDCAN 发送缓冲区请求待处理寄存器（FDCAN_TXBRP）

地址偏移：0x00C8

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **TRP[2:0]：**发送请求挂起

每个发送缓冲区都有自己的发送请求挂起位。这些位通过寄存器 FDCAN_TXBAR 置 1。请求的发送已完成或已通过 FDCAN_TXBCR 取消后，这些位会复位。

FDCAN_TXBRP 位置 1 后，会启动发送扫描以检查优先级最高（消息 ID 最小的发送缓冲区）的挂起发送请求。

取消请求会使寄存器 FDCAN_TXBRP 的相应发送请求挂起位复位。如果请求取消时发送已开始进行，则无论发送是否成功，都会在发送结束时执行复位操作。相应的 TXBRP 位复位后，取消请求位也会立即复位。

请求取消后，会在下列情况下通过 FDCAN_TXBCF 指示完成取消：

- 与相应的 TXBTO 位一起成功发送后
- 取消时发送尚未开始
- 发送因仲裁丢失而中止
- 帧发送过程中出错

在 DAR 模式下，所有发送在发送失败后都会自动取消。对于所有未成功的发送，相应的 FDCAN_TXBCF 位会置 1。

0：无发送请求挂起

1：发送请求挂起

注：如果 FDCAN_TXBRP 位在正在进行发送扫描时置 1，则进行该特殊发送扫描期间不会考虑此位。如果请求取消此类发送缓冲区，则该添加请求会立即取消，相应的 FDCAN_TXBRP 位会复位。

26.4.29 FDCAN 发送缓冲区添加请求寄存器（FDCAN_TXBAR）

地址偏移：0x00CC

复位值：0x0000 0000

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **AR[2:0]：**添加请求

每个发送缓冲区都有自己的添加请求位。写入 1 会将相应的添加请求位置 1；写入 0 没有影响。这样，主机通过对 FDCAN_TXBAR 进行一次写操作便可为多个发送缓冲区设置发送请求。如果没有运行发送扫描，这些位会立即复位，否则在发送扫描过程完成之前，这些位保持置 1 状态。

0：未添加发送请求

1：已添加发送请求。

*注：如果添加请求应用到具有挂起发送请求的发送缓冲区（对应的*FDCAN_TXBRP *位已置位），则会忽略请求。*

26.4.30 FDCAN 发送缓冲区取消请求寄存器（FDCAN_TXBCR）

地址偏移：0x00D0

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **CR[2:0]：**取消请求

每个发送缓冲区都有自己的取消请求位。写入 1 会将相应的取消请求位置 1；写入 0 没有影响。

这样，主机通过对 FDCAN_TXBCR 进行一次写操作便可为多个发送缓冲区设置取消请求。在相应的 FDCAN_TXBRP 位复位之前，这些位保持置 1 状态。

0：无取消挂起

1：取消挂起

26.4.31 FDCAN 发送缓冲区发送已发生寄存器（FDCAN_TXBTO）

地址偏移：0x00D4

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **TO[2:0]：**发送已发生

每个发送缓冲区都有自己的发送已发生位。当相应的 FDCAN_TXBRP 位在发送成功后清零时，这些位会置 1。通过向寄存器 FDCAN_TXBAR 的相应位写入 1 请求进行新发送时，这些位会复位。

0：未进行发送

1：已进行发送

26.4.32 FDCAN 发送缓冲区取消完成寄存器（FDCAN_TXBCF）

地址偏移：0x00D8

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **CF[2:0]：**取消完成

每个发送缓冲区都有自己的取消完成位。当相应的 FDCAN_TXBRP 位在通过 FDCAN_TXBCR 请求取消后清零时，这些位会置 1。

如果相应的 FDCAN_TXBRP 位在取消时未置 1，CF 会立即置 1。通过向寄存器 FDCAN_TXBAR 的相应位写入 1 请求进行新发送时，这些位会复位。

0：无发送缓冲区取消

1：发送缓冲区取消完成

26.4.33 FDCAN 发送缓冲区发送中断使能寄存器（FDCAN_TXBTIE）

地址偏移：0x00DC

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **TIE[2:0]：**发送中断使能

每个发送缓冲区都有自己的 TIE 发送中断使能位。

0：禁用发送中断

1：使能发送中断

26.4.34 FDCAN 发送缓冲区取消完成中断使能寄存器（FDCAN_TXBCIE）

地址偏移：0x00E0

复位值：0x0000 0000

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

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

Bit 31:3 保留，必须保持为复位值。

Bit 2:0 **CFIE[2:0]：**取消完成中断使能。

每个发送缓冲区都有自己的 CFIE 取消完成中断使能位。

0：禁用取消完成中断

1：使能取消完成中断

26.4.35 FDCAN 发送事件 FIFO 状态寄存器（FDCAN_TXEFS）

地址偏移：0x00E4

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						TEFL	EFF	Reserved						EFPI[1:0]	
						r	r							r	r

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

Bit 31:26 保留，必须保持为复位值。

Bit 25 **TEFL：**发送事件 FIFO 元素丢失

此位是 FDCAN_IR 中 TEFL 的副本。当 TEFL 复位时，此位也会复位。

0：没有发送事件 FIFO 元素丢失

1：发送事件 FIFO 元素已丢失，尝试向大小为零的发送事件 FIFO 进行写入时也会置 1。

Bit 24 **EFF：**事件 FIFO 已满

0：发送事件 FIFO 未满

1：发送事件 FIFO 已满

Bit 23:18 保留，必须保持为复位值。

Bit 17:16 **EFPI[1:0]：**事件 FIFO 放入索引

发送事件 FIFO 写入索引指针。

范围：0 至 3。

Bit 15:10 保留，必须保持为复位值。

Bit 9:8 **EFGI[1:0]：**事件 FIFO 获取索引

发送事件 FIFO 读取索引指针。

范围：0 至 3。

Bit 7:3 保留，必须保持为复位值。

Bit 2:0 **EFFL[2:0]：**事件 FIFO 填充级别

发送事件 FIFO 中存储的元素数量。

范围：0 至 3。

26.4.36 FDCAN 发送事件 FIFO 确认寄存器（FDCAN_TXEFA）

地址偏移：0x00E8

复位值：0x0000 0000

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

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

Bit 31:2 保留，必须保持为复位值。

Bit 1:0 **EFAI[1:0]：**事件 FIFO 确认索引

主机从发送事件 FIFO 读取元素或元素序列后，必须将从发送事件 FIFO 读取的最后一个元素的索引写入 EFAI[1:0]中。此操作会将发送事件 FIFO 获取索引（FDCAN_TXEFS 中的 EFGI[1:0]）设为 EFAI[1:0] + 1，并会更新 FIFO 0 填充级别（FDCAN_TXEFS 中的 EFFL[2:0]）。

26.4.37 FDCAN 配置时钟分频寄存器（FDCAN_CKDIV）

地址偏移：0x0100

复位值：0x0000 0000

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

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

Bit 31:4 保留，必须保持为复位值。

Bit 3:0 **PDIV[3:0]：**输入时钟分频器

CAN 内核时钟在提供 CAN 子系统使用前可进行分频。必须使用分频器输出时钟来计算速率。分频后得到 CAN 子系统最小时间片 fdcan_tq_ck 提供给 CAN 子系统时钟。

0000：分频系数为 1　　1000：分频系数为 16
0001：分频系数为 2　　1001：分频系数为 18
0010：分频系数为 4　　1010：分频系数为 20
0011：分频系数为 6　　1011：分频系数为 22
0100：分频系数为 8　　1100：分频系数为 24
0101：分频系数为 10　　1101：分频系数为 26
0110：分频系数为 12　　1110：分频系数为 28
0111：分频系数为 14　　1111：分频系数为 30

该位域受写保护（P）：即仅当 FDCAN_CCCR 寄存器的 CCE 和 INIT 位同时置位时，才允许进行写操作。

27 串行外设接口（SPI）

27.1 SPI 简介

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。

27.2 SPI 主要特性

- 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
- 8 个主模式波特率预分频系数（最大为 $f_{PCLK}/2$）
- 从模式频率（最大为 $f_{PCLK}/2$）
- 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 1 字节发送和接收缓冲器：产生发送和接受请求

27.3 SPI 功能描述

27.3.1 概述

SPI 的框图参见下图。

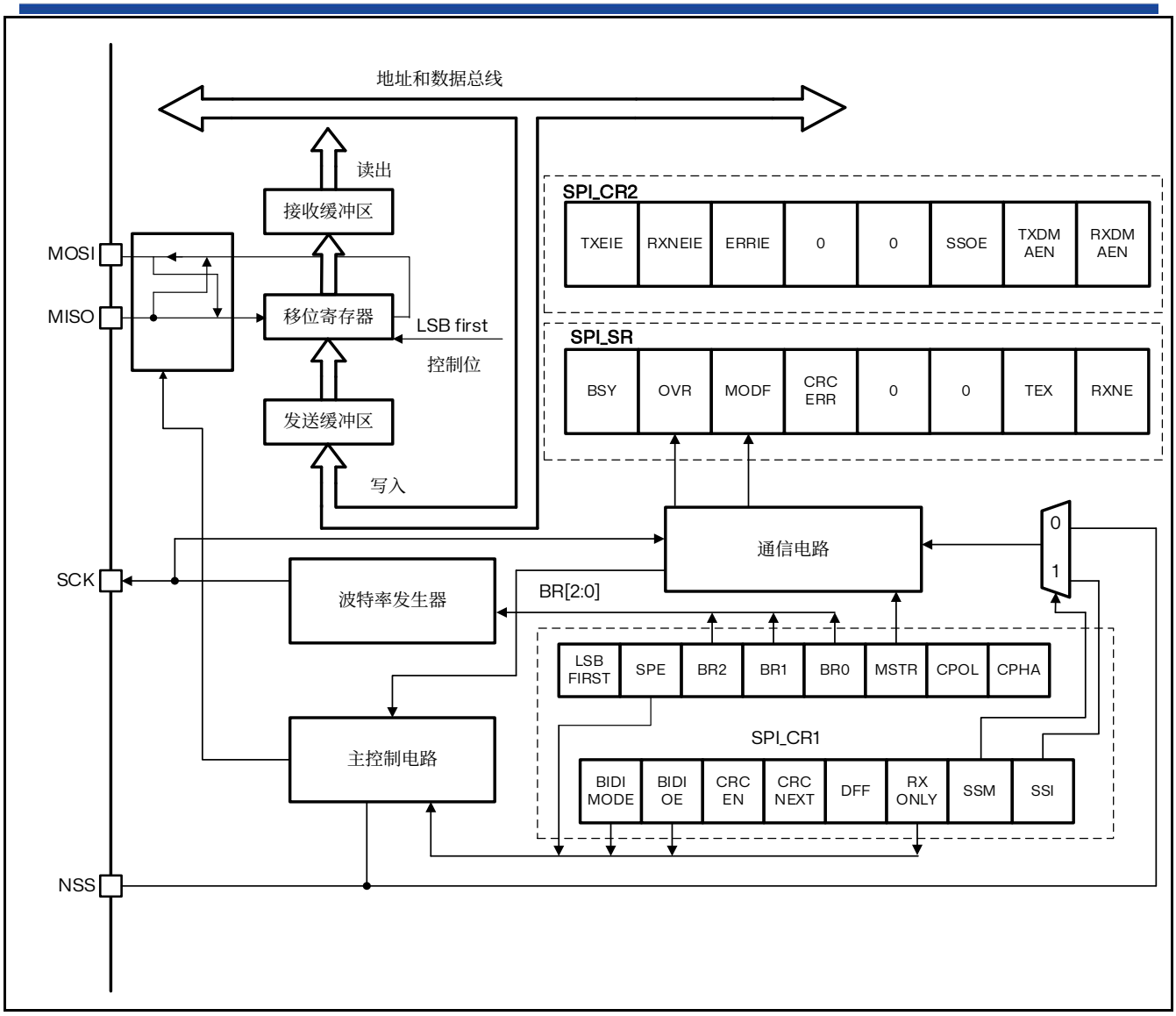
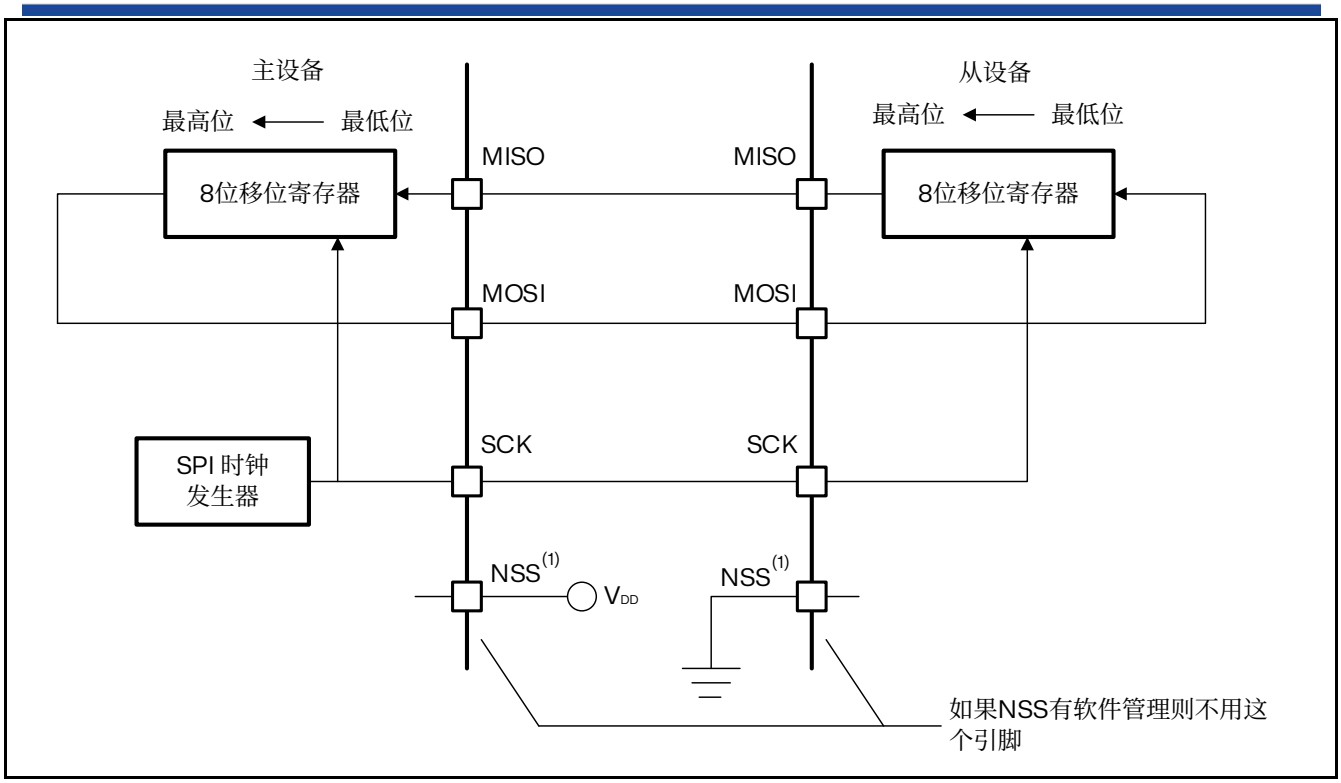


图 27.1 SPI 框图

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（SSOE 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（MSTR=1，SSOE=0）时，如果 NSS 被拉低，则这个 SPI 设备进入主模式失败状态：即 MSTR 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。



1. 这里 NSS 引脚设置为输入

图 27.2 单主和单从应用

MOSI 脚相互连接，MISO 脚相互连接。这样，数据在主和从之间串行地传输（MSB 位在前）。

通信总是由主设备发起。主设备通过 MOSI 脚把数据发送给从设备，从设备通过 MISO 引脚回传数据。这意味全双工通信的数据输出和数据输入是用同一个时钟信号同步的；时钟信号由主设备通过 SCK 脚提供。

从选择（NSS）脚管理

可以使用 SPI_CR1 寄存器中的 SSM 位设置硬件或软件管理从器件选择。

- 软件管理 NSS（SSM = 1）
  从器件选择信息在内部由 SPI_CR1 寄存器中的 SSI 位的值驱动。外部 NSS 引脚空闲，可供其它应用使用。
- 硬件 NSS 模式，分两种情况：硬件管理 NSS（SSM = 0）
  根据 NSS 输出配置（SPI_CR1 寄存器中的 SSOE 位），硬件管理 NSS 有两种模式。
  - NSS 输出使能（SSM = 0，SSOE = 1）
    仅当器件在主模式下工作时才使用此配置。当主器件开始通信时，NSS 信号驱动为低电平，并保持到 SPI 被关闭为止。
  - NSS 输出禁止（SSM = 0，SSOE = 0）
    对于在主模式下工作的器件，此配置允许多主模式功能。对于设置为从模式的器件，NSS 引脚用作传统 NSS 输入：在 NSS 为低电平时片选该从器件，在 NSS 为高电平时取消对它的片选。

时钟相位和时钟极性

通过 SPI_CR1 寄存器中的 CPOL 和 CPHA 位，可以用软件选择四种可能的时序关系。CPOL（时钟极性）位控制不传任何数据时的时钟电平状态。此位对主器件和从器件都有作用。如果复位 CPOL，SCK 引脚在空闲状态处于低电平。如果将 CPOL 置 1，SCK 引脚在空闲状态处于高电平。

如果将 CPHA（时钟相位）位置 1，则 SCK 引脚上的第二个边沿（如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）对 MSBit 采样。即，在第二个时钟边沿锁存数据。如果复位 CPHA 位，则 SCK 引脚上的第一个边沿（如果将 CPOL 位置 1，则为下降沿；如果复位 CPOL 位，则为上升沿）对 MSBit 采样。即，在第一个时钟边沿锁存数据。

CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

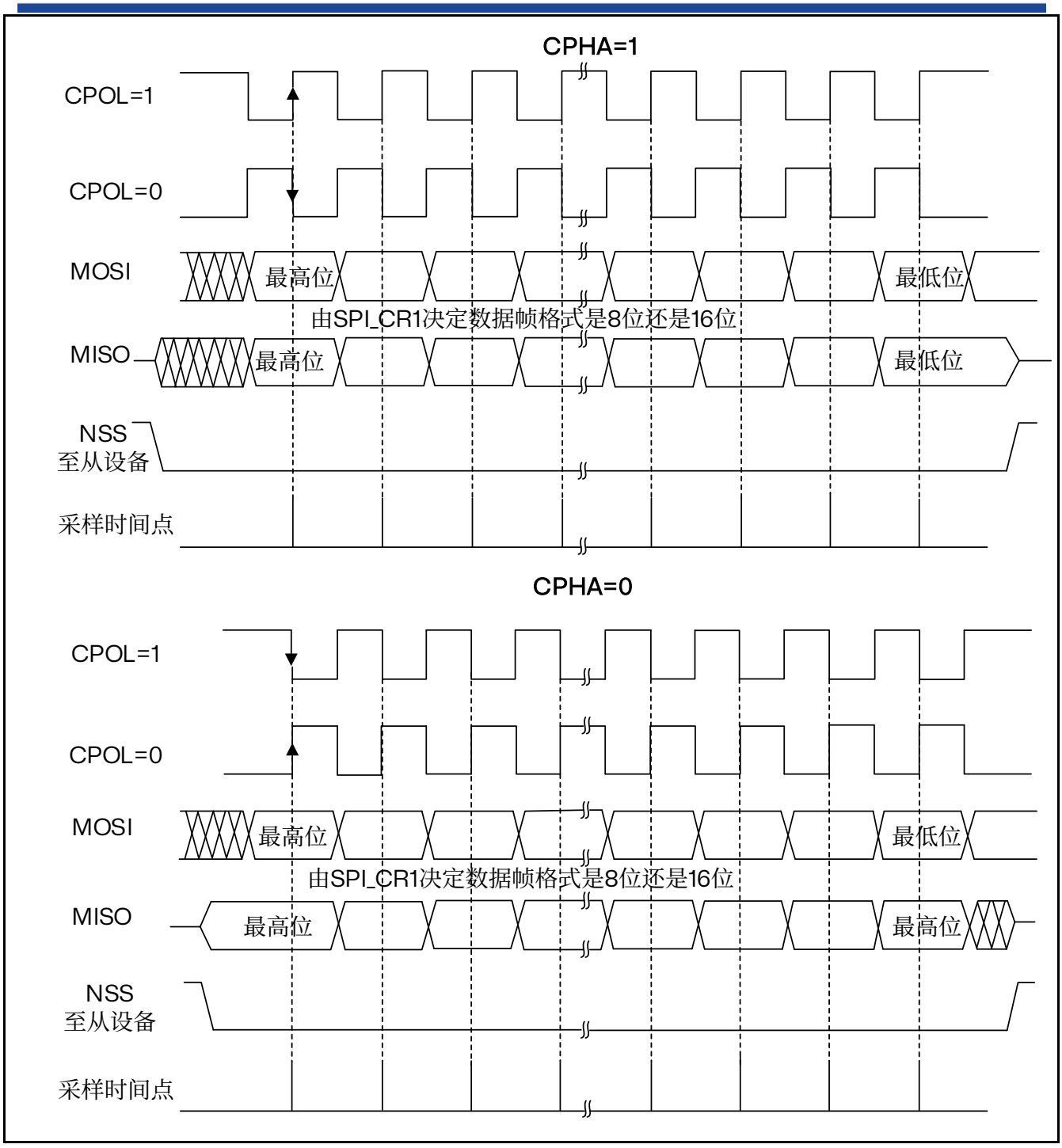
下图显示了在 CPHA 和 CPOL 位的四种组合下的 SPI 传输。可以将该图解释为主器件或从器件时序图，其中 SCK 引脚、MISO 引脚、MOSI 引脚直接连接在主器件和从器件之间。

注：在切换 CPOL/CPHA *位之前，必须通过复位* SPE *位来关闭* SPI*。*

必须以同一时序模式对主器件和从器件进行编程。

SCK *的空闲状态必须与* SPI_CR1 *寄存器中选择的极性相对应（如果* CPOL=1*，则上拉* SCK*；如果* CPOL=0*，则下拉* SCK*）。*

通过 SPI_CR1 *寄存器中的* DFF *位选择数据帧格式（*8 *或* 16 *位），该格式决定了发送/接收过程中的数据长度。*



1. 所示的时序为 SPI_CR1 寄存器中的 LSBFIRST 位复位时的时序。

图 27.3 数据时钟时序图

数据帧格式

根据 SPI_CR1 寄存器中的 LSBFIRST 位，输出数据位时可以 MSB 在先也可以 LSB 在先。

根据 SPI_CR1 寄存器的 DFF 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。

27.3.2 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_CR1 寄存器中 BR[2:0]的设置不影响数据传输速率。

注：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 DFF 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。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_CR1 寄存器中的 LSBFIRST 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_CR1 寄存器中的 SSM 位并清除 SSI 位。
5. 清除 MSTR 位、设置 SPE 位（SPI_CR1 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，数据字被并行地写入发送缓冲器。

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（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_SP 寄存器的 TXE 标志被设置，如果设置了 SPI_CR2 寄存器的 TXEIE 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_SR 寄存器中的 RXNE 标志被设置。
- 如果设置了 SPI_CR2 寄存器中的 RXNEIE 位，则产生中断。

在最后一个采样时钟边沿后，RXNE 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_DR 寄存器时，SPI 设备返回这个接收缓冲器的数值。

读 SPI_DR 寄存器时，RXNE 位被清除。

27.3.3 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_CR1 寄存器的 BR[2:0]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。
3. 设置 DFF 位来定义 8 位或 16 位数据帧格式。
4. 配置 SPI_CR1 寄存器的 LSBFIRST 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_CR1 寄存器的 SSM 位和 SSI 位。如果 NSS 引脚工作在输出模式，则只需设置 SSOE 位。
6. 必须设置 MSTR 位和 SPE 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_CR1 寄存器中的 LSBFIRST 位的设置。数据从发送缓冲器传输到移位寄存器时 TXE 标志将被置位，如果设置了 SPI_CR1 寄存器中的 TXEIE 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 RXNE 标志被置位。
- 如果设置了 SPI_CR2 寄存器中的 RXNEIE 位，则产生中断。

在最后采样时钟沿，RXNE 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_DR 寄存器时，SPI 设备返回接收缓冲器中的数据。

读 SPI_DR 寄存器将清除 RXNE 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 TXE 标志应该为“1”。

注：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。

27.3.4 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（BIDIMODE=1）

设置 SPI_CR1 寄存器中的 BIDIMODE 位而使能此模式。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_CR1 寄存器里的 BIDIOE 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（BIDIMODE=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（BIDIMODE=0，RXONLY=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_CR2 寄存器的 RXONLY 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 SPE 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，SSI 位为“0”）同时 SCK 有时钟脉冲，SPI 就一直在接收。

27.3.5 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_DR 寄存器的读操作，将返回接收缓冲器的内容；写入 SPI_DR 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（BIDIMODE=0 并且 RXONLY=0）
  - 当写入数据到 SPI_DR 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_DR 寄存器（接收缓冲器）中。
- 单向的只接收模式（BIDIMODE=0 并且 RXONLY=1）
  - SPE=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_DR 寄存器（接收缓冲器）中。

- 双向模式，发送时（BIDIMODE=1 并且 BIDIOE=1）
  - 当写入数据到 SPI_DR 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（BIDIMODE=1 并且 BIDIOE=0）
  - SPE=1 并且 BIDIOE=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_DR 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（BIDIMODE=0 并且 RXONLY=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（BIDIMODE=0 并且 RXONLY=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（BIDIMODE=1 并且 BIDIOE=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（BIDIMODE=1 并且 BIDIOE=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_DR 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 TXE 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_CR2 寄存器中设置了 TXEIE 位，则此时会产生一个中断；写入 SPI_DR 寄存器即可清除 TXE 位。

注：在写入发送缓冲器之前，软件必须确认 TXE 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 RXNE 标志（接收缓冲器非空）；它表示数据已经就绪，可以从 SPI_DR 寄存器读出；如果在 SPI_CR2 寄存器中设置了 RXNEIE 位，则此时会产生一个中断；读出 SPI_DR 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（BIDIMODE=0 并且 RXONLY=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：

1. 设置 SPE 位为“1”，使能 SPI 模块；
2. 在 SPI_DR 寄存器中写入第一个要发送的数据，这个操作会清除 TXE 标志；
3. 等待 TXE=1，然后写入第二个要发送的数据。等待 RXNE=1，然后读出 SPI_DR 寄存器并获得第一个接收到的数据，读 SPI_DR 的同时清除了 RXNE 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 RXNE=1，然后接收最后一个数据；
5. 等待 TXE=1，在 BSY=0 之后关闭 SPI 模块。

也可以在响应 RXNE 或 TXE 标志的上升沿产生的中断的处理程序中实现这个过程。

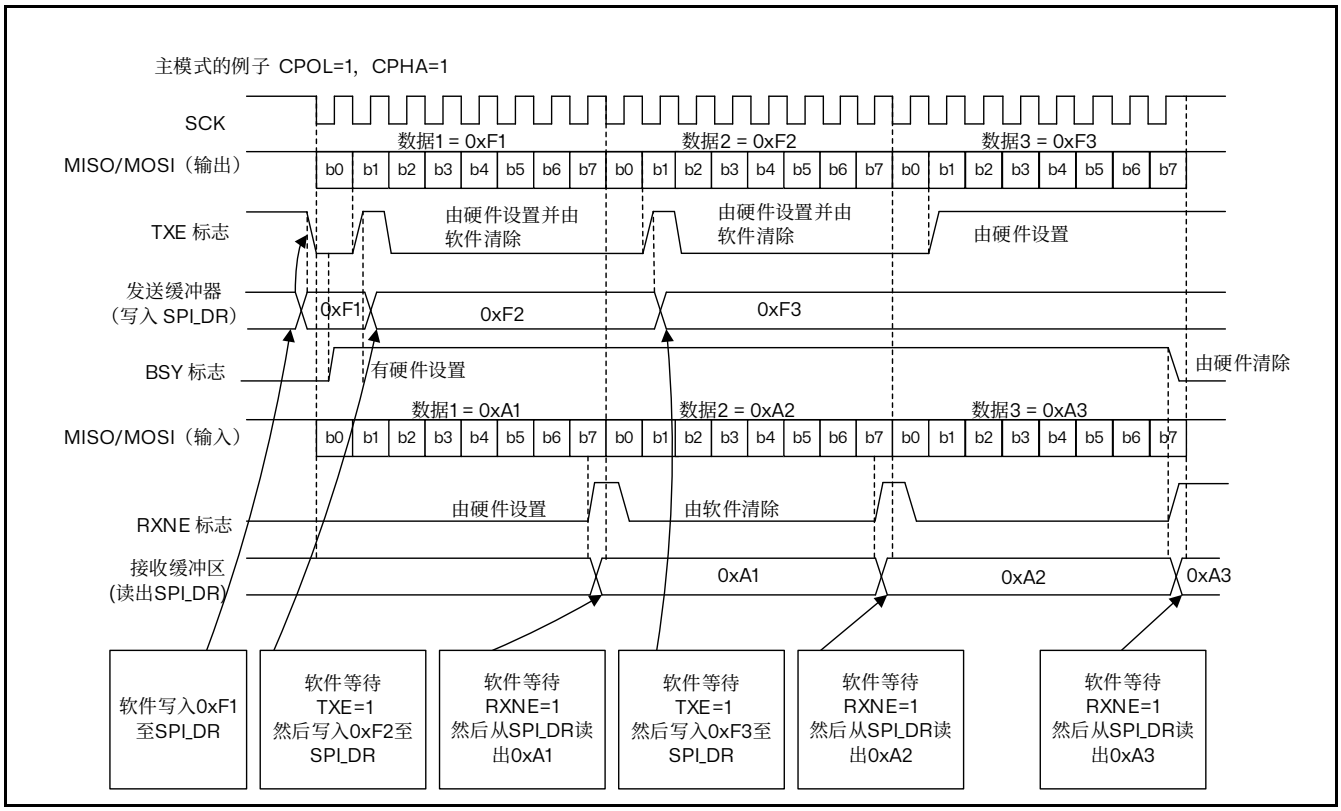


图 27.4 主/全双工模式下（BIDIMODE=0 且 RXONLY=0）连续传输时，TXE/RXNE/BSY 行为

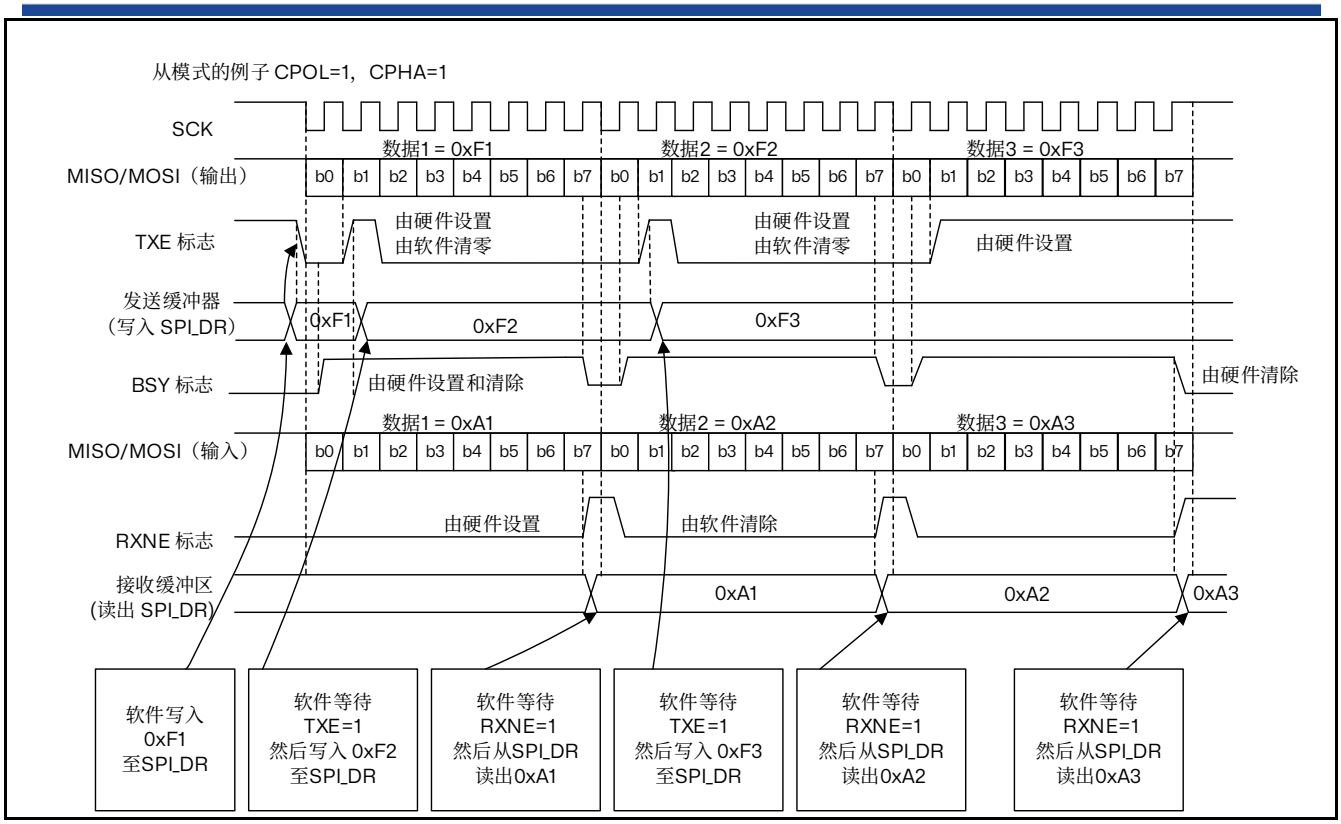


图 27.5 从/全双工模式下（BIDIMODE=0 且 RXONLY=0）连续传输时，TXE/RXNE/BSY 行为

只发送过程（BIDIMODE=0 并且 RXONLY=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：

1. 设置 SPE 位为“1”，使能 SPI 模块；
2. 在 SPI_DR 寄存器中写入第一个要发送的数据，这个操作会清除 TXE 标志；
3. 等待 TXE=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_DR 寄存器之后，等待 TXE=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 TXE 标志的上升沿产生的中断的处理程序中实现这个过程。

注：对于不连续的传输，在写入 SPI_DR 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 TXE=1，然后再等待 BSY=0。

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_SR 寄存器中的 OVR 位会变为“1”。（注：软件不必理会这个 OVR 标志位）

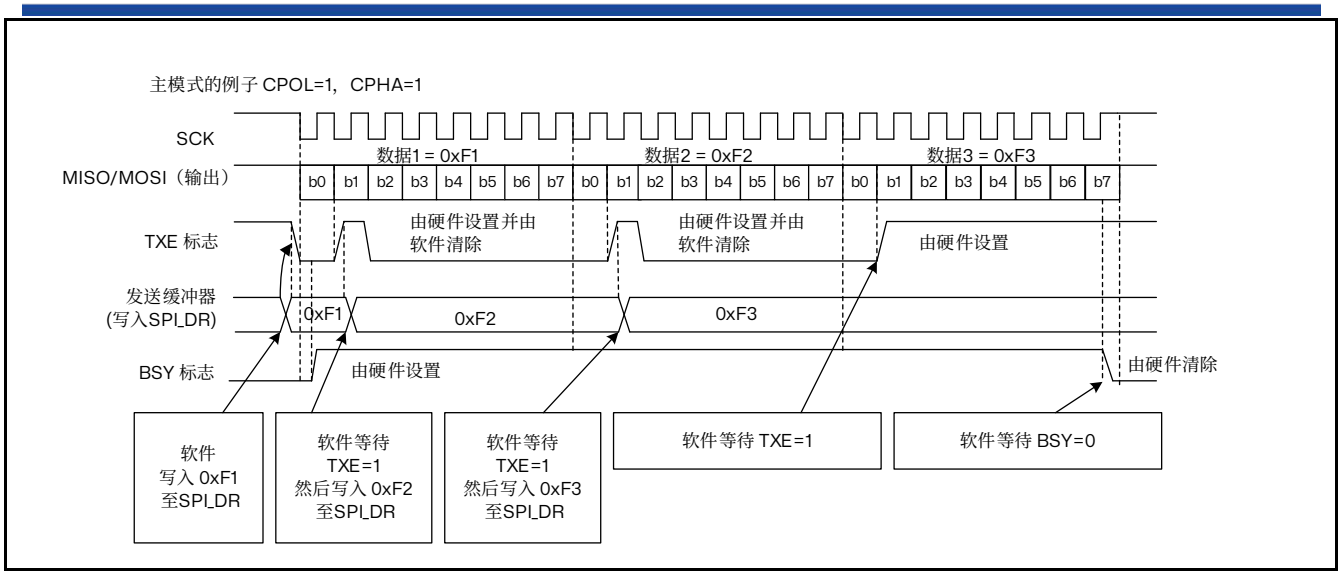


图 27.6 主设备只发送模式（BIDIMODE=0 并且 RXONLY=0）下连续传输时，TXE/BSY 行为

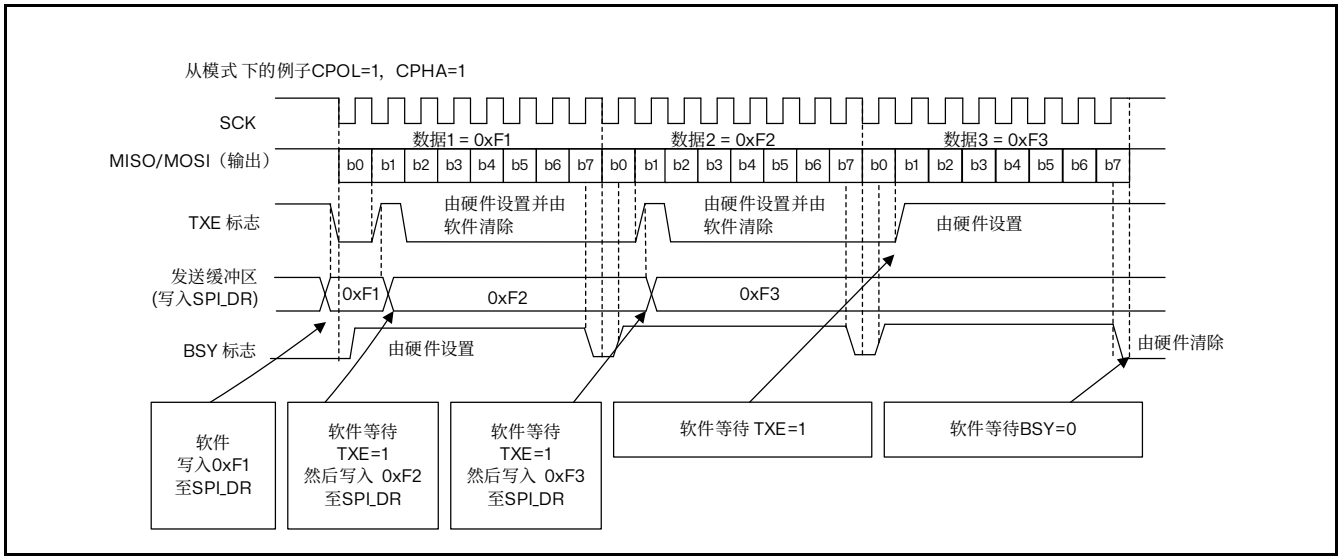


图 27.7 从设备只发送模式（BIDIMODE=0 并且 RXONLY=0）下连续传输时，TXE/BSY 行为

双向发送过程（BIDIMODE=1 并且 BIDIOE=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_CR1 寄存器中同时设置 BIDIMODE 和 BIDIOE 位为“1”。

单向只接收模式（BIDIMODE=0 并且 RXONLY=1）

在此模式下，传输过程可以简要说明如下：

1. 在 SPI_CR1 寄存器中，设置 RXONLY=1；
2. 设置 SPE=1，使能 SPI 模块：
   a）主模式下，立刻产生 SCK 时钟信号，在关闭 SPI（SPE=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 RXNE=1，然后读出 SPI_DR 寄存器以获得收到的数据（同时会清除 RXNE 位）。重复这个操作接收所有数据。

也可以在响应 RXNE 标志的上升沿产生的中断的处理程序中实现这个过程。

注：如果在最后一个数据传输结束后关闭 SPI 模块，请按照章节：关闭 SPI 中的建议操作。

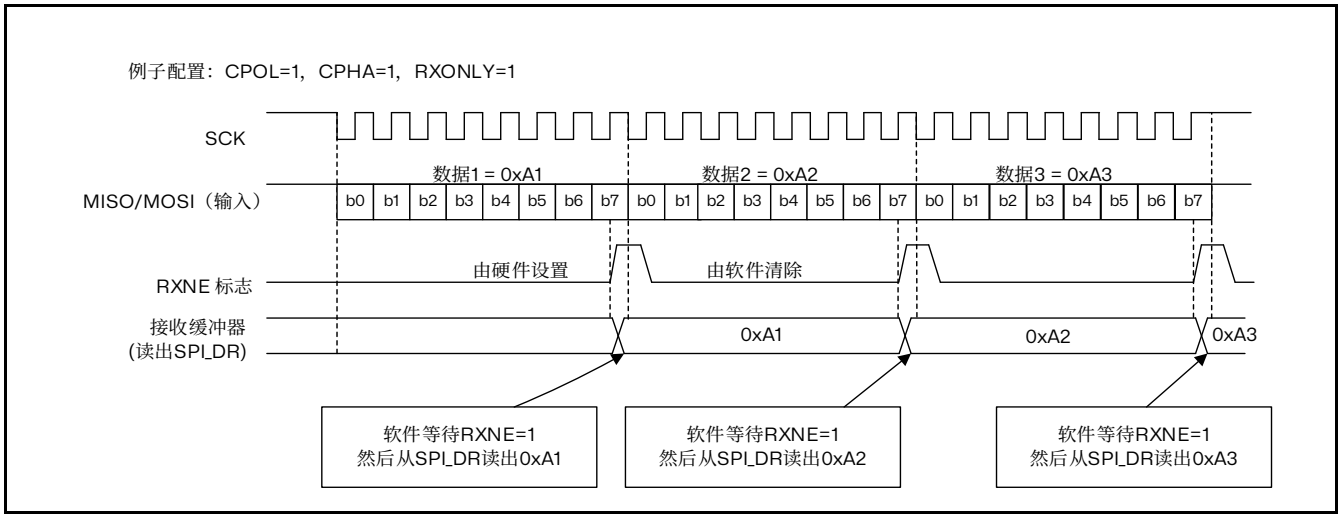


图 27.8 只接收模式（BIDIMODE=0 并且 RXONLY=1）下连续传输时，RXNE 行为

单向接收过程（BIDIMODE=1 并且 BIDIOE=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_CR1 寄存器中设置 BIDIMODE 为“1”并清除 BIDIOE 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 TXE 的上升沿（或 TXE 中断），并立即在正在进行的传输结束之前写入 SPI_DR 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（RXONLY=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。

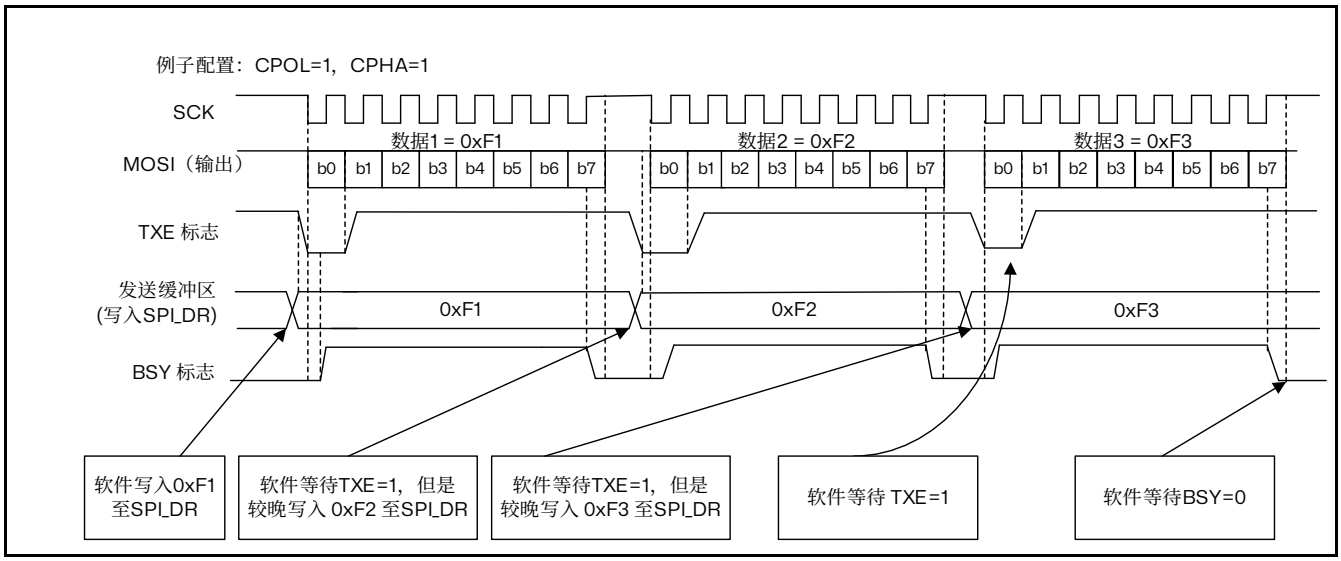


图 27.9 非连续传输发送（BIDIMODE=0 并且 RXONLY=0）时，TXE/BSY 行为

27.3.6 CRC 计算

为确保通信的可靠性，SPI 模块实现了硬件 CRC 功能。针对发送的数据和接收的数据分别实现 CRC 计算。使用可编程的多项式对每个位来计算 CRC。在由 SPI_CR1 寄存器中的 CPHA 位和 CPOL 位定义的采样时钟边沿采样每个位来进行计算。

注：该SPI 模块提供两种CRC 计算标准，具体取决于为发送和/或接收选择的数据帧格式：8 位数据采用 CRC8，*16 位数据采用* CRC16。

通过将 SPI_CR1 寄存器中的 CRCEN 位置 1 来使能 CRC 的计算。此操作将复位 CRC 寄存器（SPI_RXCRCR 和 SPI_TXCRCR）。在全双工或只发送模式下，如果传输由软件（CPU 模式）管理，则在将最后传输的数据写入 SPI_DR 后，必须立即对 CRCNEXT 位执行写操作。最后一次数据传输结束时，将发送 SPI_TXCRCR 值。

在只接收模式下，如果传输由软件（CPU 模式）管理，则在接收到倒数第二个数据后，必须对 CRCNEXT 位执行写操作。在收到最后一个数据后会收到 CRC，然后执行 CRC 校验。如果传输过程中出现数据损坏，则在数据和 CRC 传输结束时，SPI_SR 寄存器中的 CRCERR 标志将置 1。

如果发送缓冲区中存在数据，则只有在发送数据字节后才会发送 CRC 值。在 CRC 发送期间，CRC 计算器处于关闭状态且寄存器值保持不变。

可通过以下步骤使用 CRC 进行 SPI 通信：

1. 对 CPOL、CPHA、LSBFirst、BR、SSM、SSI 和 MSTR 值进行编程。
2. 对 SPI_CRCPR 寄存器中的多项式进行编程。
3. 通过将 SPI_CR1 寄存器中的 CRCEN 位置 1 来使能 CRC 计算。此操作还会将 SPI_RXCRCR 和 SPI_TXCRCR 寄存器清零。
4. 通过将 SPI_CR1 寄存器中的 SPE 位置 1 使能 SPI。
5. 启动并维持通信，直到只剩下一个字节或半字未发送或接收。
   - 在全双工或只发送模式下，如果传输由软件管理，则在向发送缓冲区写入最后一个字节或半字后，将 SPI_CR1 寄存器中的 CRCNEXT 位置 1，以表示在发送完最后一个字节后将发送 CRC。
   - 在只接收模式下，在接收倒数第二个数据后，立即将 CRCNEXT 位置 1，以便使 SPI 准备好在接收完最后一个数据后进入 CRC 阶段。在 CRC 传输期间，CRC 计算将冻结。
6. 传输完最后一个字节或半字后，SPI 进入 CRC 传输和校验阶段。在全双工模式或只接收模式下，将接收的 CRC 与 SPI_RXCRCR 值进行比较。如果两个值不匹配，则 SPI_SR 中的 CRCERR 标志将置 1，并且在 SPI_CR2 寄存器中的 ERRIE 位置 1 时会产生中断。

注：当SPI 处于从模式时，注意只能在时钟稳定（即，时钟处于空闲电平）时使能CRC 计算。 否则，可能导致CRC 计算错误。因为，只要CRCEN 位置1，无论SPE 位的值如何，只要有时钟输入，CRC 计算器就开始工作。

在SPI 通信时钟频率较高的情况下，发送CRC 时务必小心。由于在CRC 传输阶段CPU 应尽可能空闲，因此禁止在CRC 发送阶段调用函数，以便避免最后的数据和CRC 接收出错。实际上，在发送/接收最后的数据之前必须对CRCNEXT 位执行写操作。

SPI 通信时钟频率较高时，建议使用DMA 模式来避免由于CPU 访问影响SPI 带宽而导致SPI 速度性能下降。

*如果将器件配置为从器件，并且使用*NSS *硬件模式，则需要在数据阶段和*CRC *阶段之间将*NSS *引脚保持为低电平。*

当 SPI 配置为从模式并且 CRC 功能已使能时，即使 NSS 引脚上为高电平，也会进行 CRC 计算。例如，在多从模式环境下可能出现这种情况，此时通信主器件会交替寻址从器件。

在对从器件片选的切换期间内，应在主器件和从器件两端同时将 CRC 值清零，以重新同步主从双方的 CRC 计算。

要将 CRC 清零，请按以下步骤操作：

1. 关闭 SPI（SPE=0）
2. 将 CRCEN 位清零
3. 将 CRCEN 位置 1
4. 使能 SPI（SPE = 1）

27.3.7 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（TXE）

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_DR 时，TXE 标志被清除。

接收缓冲器非空（RXNE）

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（MSTR=1、BDM=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（MSTR=1、BDM=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：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*TXE *和*RXNE *标志。*

27.3.8 关闭 SPI

当通讯结束，可以通过关闭 SPI 模块来终止通讯。清除 SPE 位即可关闭 SPI。

在某些配置下，如果再传输还未完成时，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（BIDIMODE=0，RXONLY=0）

1. 等待 RXNE=1 并接收最后一个数据；
2. 等待 TXE=1；
3. 等待 BSY=0；
4. 关闭 SPI（SPE=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（BIDIMODE=0，RXONLY=0）或双向的发送模式（BIDIMODE=1，BIDIOE=1）

在 SPI_DR 寄存器中写入最后一个数据后：

1. 等待 TXE=1；
2. 等待 BSY=0；
3. 关闭 SPI（SPE=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接收模式（MSTR=1，BIDIMODE=0，RXONLY=1）或双向的接收模式（MSTR=1，BIDIMODE=1，BIDIOE=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）RXNE=1；
2. 在关闭 SPI（SPE=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 RXNE=1。

*注：在主模式下的单向只发送模式（*MSTR=1，BDM=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（MSTR=0，BIDIMODE=0，RXONLY=1）或双向的接收模式（MSTR=0，BIDIMODE=1，BIDIOE=0）

1. 可以在任何时候关闭 SPI（SPE=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。

27.3.9 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_CR2 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 TXE 被设置为“1”时发出 DMA 请求，DMA 控制器则写数据至 SPI_DR 寄存器，TXE 标志因此而被清除。
- 接收时，在每次 RXNE 被设置为“1”时发出 DMA 请求，DMA 控制器则从 SPI_DR 寄存器读出数据，RXNE 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”（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_ISR 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 或进入停止模式时，破坏最后一个数据的传输。因此软件需要先等待 TXE=1，然后等待 BSY=0。

*注：在不连续的通信中，在写数据到*SPI_DR *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*TXE=1 *再等待*BSY=0。

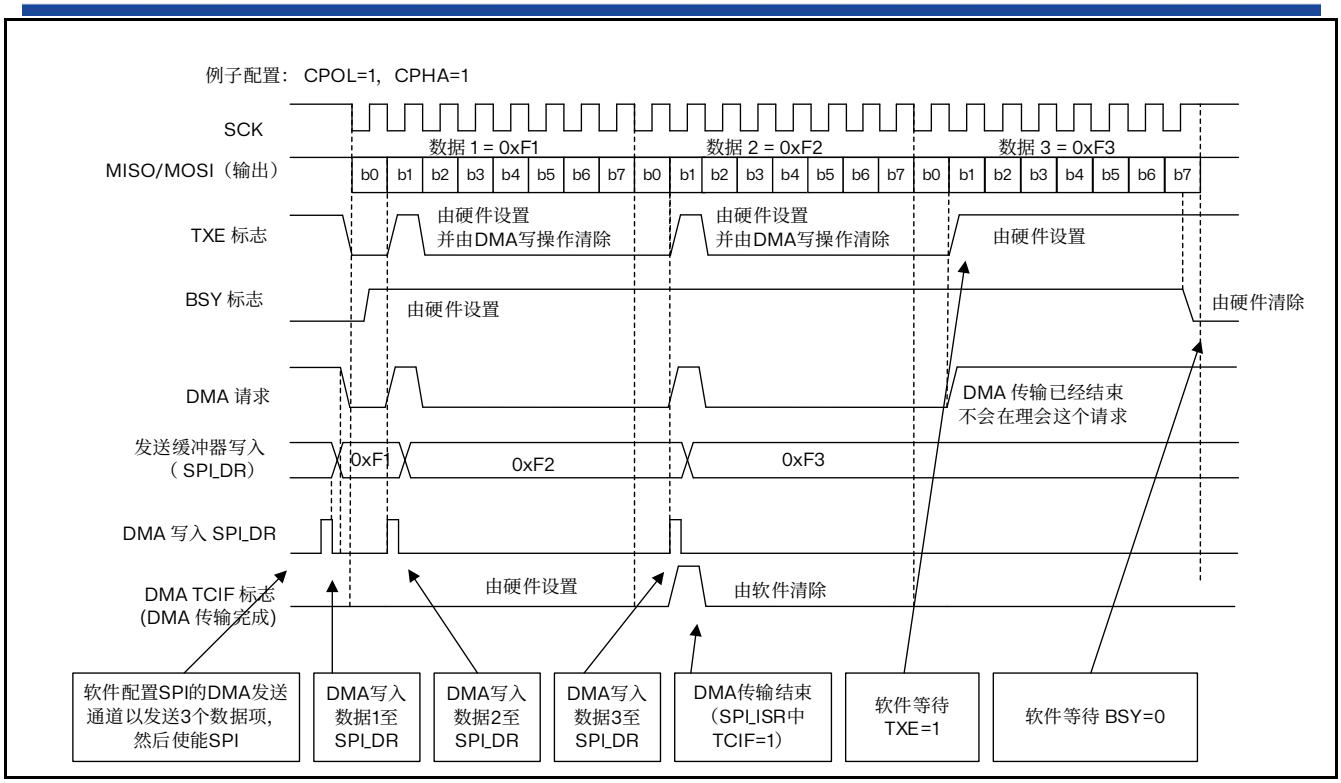


图 27.10 使用 DMA 发送

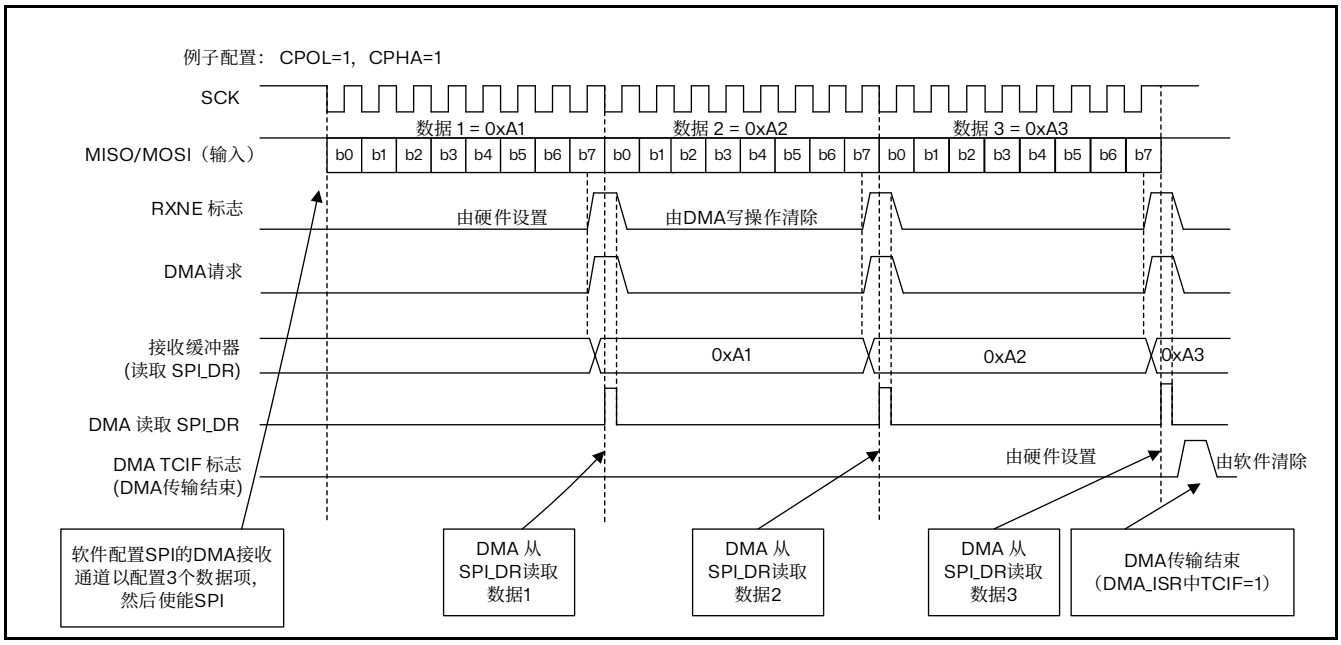


图 27.11 使用 DMA 接收

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_SR 寄存器的 CRCERR 标志为“1”表示在传输期间发生错误。

27.3.10 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，SSI 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；
- SPE 位被清为“0”。这将停止一切输出，并且关闭 SPI 接口；
- MSTR 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_SR 寄存器的读或写操作；
2. 然后写 SPI_CR1 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，SPE 和 MSTR 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 SPE 和 MSTR 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 RXNE 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_DR 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_DR 寄存器和 SPI_SR 寄存器可将 OVR 清除。

CRC 错误

当设置了 SPI_CR 寄存器上的 CRCEN 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_TXCRCR 数值）与接收方 SPI_RXCRCR 寄存器中的数值不匹配，则 SPI_SR 寄存器上的 CRCERR 标志被置位为“1”。

27.3.11 SPI 中断

表 27.1 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	TXE	TXEIE
接收缓冲器非空标志	RXNE	RXNEIE
主模式失效事件	MODF	ERRIE
溢出错误	OVR	
CRC 错误标志	CRCERR	

27.4 SPI 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

27.4.1 SPI 控制寄存器 1（SPI_CR1）

地址偏移：0x00
复位值：0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDI MODE	BIDIOE	CRCEN	CRC NEXT	DFF	RX ONLY	SSM	SSI	LSB FIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15 **BIDIMODE**：双向数据模式使能（Bidirectional data mode enable）
0：选择“双线双向”模式；
1：选择“单线双向”模式。

Bit 14 **BIDIOE**：双向模式下的输出使能（Output enable in bidirectional mode）
和 BIDIMODE 位一起决定在“单线双向”模式下数据的输出方向
0：输出禁止（只收模式）；
1：输出使能（只发模式）。

注：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。*

Bit 13 **CRCEN**：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（*SPE=0*），才能写该位，否则出错。*

该位只能在全双工模式下使用。

Bit 12 **CRCNEXT**：下一个发送 CRC（Transmit CRC next）
0：下一个发送的值来自发送缓冲区。
1：下一个发送的值来自发送 CRC 寄存器。

注：在 SPI_DR *寄存器写入最后一个数据后应马上设置该位。*

Bit 11 **DFF**：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（*SPE=0*）时，才能写该位，否则出错。*

Bit 10 **RXONLY**：只接收（Receive only）
该位和 BIDIMODE 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）。

Bit 9 **SSM**：软件从设备管理 （Software slave management）

当 SSM 被置位时，NSS 引脚上的电平由 SSI 位的值决定。

0：禁止软件从设备管理；

1：使能软件从设备管理。

Bit 8 **SSI**：内部从设备选择（Internal slave select）

该位只在 SSM 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。

Bit 7 **LSBFIRST**：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。

Bit 6 **SPE**：SPI 使能（SPI enable）

0：禁止 SPI 设备；

1：开启 SPI 设备。

注：当关闭SPI 设备时，请按照章节：关闭SPI 中描述的过程操作。

Bit 5:3 **BR[2:0]**：波特率控制（Baud rate control）

000：$f_{PCLK}/2$

001：$f_{PCLK}/4$

010：$f_{PCLK}/8$

011：$f_{PCLK}/16$

100：$f_{PCLK}/32$

101：$f_{PCLK}/64$

110：$f_{PCLK}/128$

111：$f_{PCLK}/256$

注：当通信正在进行的时候，不能修改这些位。

Bit 2 **MSTR**：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。

Bit 1 **CPOL**：时钟极性（Clock polarity）

0：空闲状态时，SCK 保持低电平；

1：空闲状态时，SCK 保持高电平。

注：当通信正在进行的时候，不能修改该位。

Bit 0 **CPHA**：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。

27.4.2 SPI 控制寄存器 2（SPI_CR2）

地址偏移：0x04

复位值：0x0000 0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TXEIE	RXNEIE	ERRIE	Reserved		SSOE	TXDMAEN	RXDMAEN
								rw	rw	rw			rw	rw	rw

Bit 15:8 保留，必须保持为复位值。

Bit 7 **TXEIE**：发送缓冲区空中断使能（Tx buffer empty interrupt enable）

0：禁止 TXE 中断；

1：允许 TXE 中断，当 TXE 标志置位为“1”时产生中断请求。

Bit 6 **RXNEIE**：接收缓冲区非空中断使能（RX buffer not empty interrupt enable）

0：禁止 RXNE 中断；

1：允许 RXNE 中断，当 RXNE 标志置位时产生中断请求。

Bit 5 **ERRIE**：错误中断使能（Error interrupt enable）

当错误（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。

Bit 4:3 保留，必须保持为复位值。

Bit 2 **SSOE**：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。

Bit 1 **TXDMAEN**：发送缓冲区 DMA 使能（Tx buffer DMA enable）

当该位被设置时，TXE 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。

Bit 0 **RXDMAEN**：接收缓冲区 DMA 使能（Rx buffer DMA enable）

当该位被设置时，RXNE 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。

27.4.3 SPI 状态寄存器（SPI_SR）

地址偏移：0x08

复位值：0x0000 0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BSY	OVR	MODF	CRC ERR	Reserved		TXE	RXNE
								r	r	r	rc_w0			r	r

Bit 15:8 保留，必须保持为复位值。

Bit 7 **BSY**：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，详见章节：状态标志和关闭*SPI。

Bit 6 **OVR**：溢出标志（Overrun 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。

Bit 5 **MODF**：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。

Bit 4 **CRCERR**：CRC 错误标志（CRC error flag）

0：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值匹配；

1：收到的 CRC 值和 SPI_RXCRCR 寄存器中的值不匹配。

该位由硬件置位，由软件写“0”而复位。

Bit 3:2 保留，必须保持为复位值。

Bit 1 **TXE**：发送缓冲为空（Transmit buffer empty）

0：发送缓冲非空；

1：发送缓冲为空。

Bit 0 **RXNE**：接收缓冲非空（Receive buffer not empty）

0：接收缓冲为空；

1：接收缓冲非空。

27.4.4 SPI 数据寄存器（SPI_DR）

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
DR[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:0 **DR[15:0]：**数据寄存器（Data register）

待发送或者已经收到的数据

数据寄存器对应两个缓冲区：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。

注：对 SPI 模式的注释：

根据 SPI_CR1 的 DFF 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，需要在使能 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_DR[7:0]。在接收时，SPI_DR[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_DR[15:0]。

27.4.5 SPI CRC 多项式寄存器（SPI_CRCPR）

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
CRCPOLY[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:0 **CRCPOLY[15:0]：**CRC 多项式寄存器（CRC polynomial register）

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。

27.4.6 SPI RX CRC 寄存器（SPI_RXCRCR）

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
RXCRC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:0 **RXCRC[15:0]：**接收 CRC 寄存器

在启用 CRC 计算时，RXCRC[15:0]中包含了依据收到的字节计算的 CRC 数值。当在 SPI_CR1 的 CRCEN 位写入“1”时，该寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。

当数据帧格式被设置为 8 位时，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 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。

27.4.7 SPI TX CRC 寄存器（SPI_TXCRCR）

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
TXCRC[15:0]															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:0 **TXCRC[15:0]：**发送 CRC 寄存器

在启用 CRC 计算时，TXCRC[15:0]中包含了依据将要发送的字节计算的 CRC 数值。当在 SPI_CR1 中的 CRCEN 位写入“1”时，该寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。

当数据帧格式被设置为 8 位时，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 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。

28 I²C 接口（I2C）

28.1 I²C 简介

I²C（芯片间）总线接口连接微控制器和串行 I²C 总线。它提供多主机功能，控制所有 I²C 总线特定的时序、协议、仲裁和定时。它支持标准模式（Sm，最高 100kHz）和快速模式（Fm，最高 400kHz）。

根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

28.2 I²C 主要特点

- 并行总线/I²C 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
- I²C 主设备功能
  - 产生时钟
  - 产生起始和停止信号
- I²C 从设备功能
  - 可编程的 I²C 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）
- 模拟噪声滤波器
- 状态标志：
  - 发送器/接收器模式标志
  - 字节发送结束标志
  - I²C 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
  - 禁止拉长时钟功能时的上溢或下溢
- 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
- 可选的拉长时钟功能
- 具单字节缓冲器的 DMA

注：不是所有产品中都包含上述所有特性。请参考相关的数据手册，确认该产品支持的 I²C 功能。

28.3 I²C 功能描述

I²C 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 I²C 总线。允许连接到标准（高达 100kHz）或快速（高达 400kHz）的 I²C 总线。

28.3.1 模式选择

接口可以下述 4 种模式中的一种运行：

- 从发送器模式
- 从接收器模式
- 主发送器模式
- 主接收器模式

该模块默认地工作于从模式。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，I²C 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，I²C 接口能识别它自己的地址（7 位或 10 位）和广播呼叫地址。软件能够控制开启或禁止广播呼叫地址的识别。

数据和地址按 8 位/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。

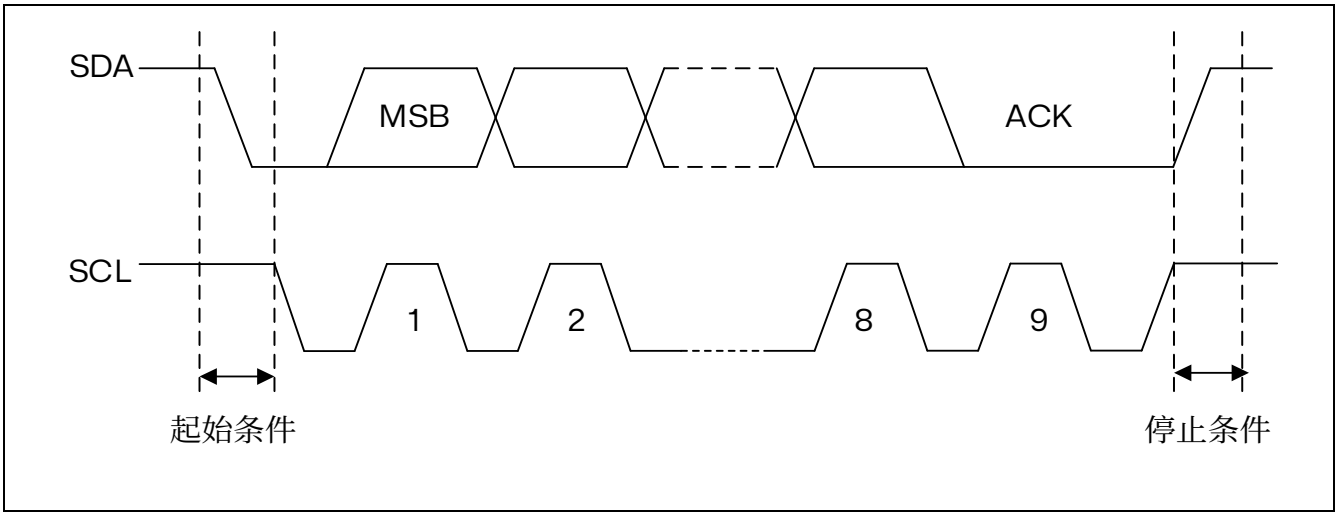


图 28.1 I²C 总线协议

软件可以开启或禁止应答（ACK），并可以设置 I²C 接口的地址（7 位、10 位地址或广播呼叫地址）。

I²C 接口的功能框图示于下图。

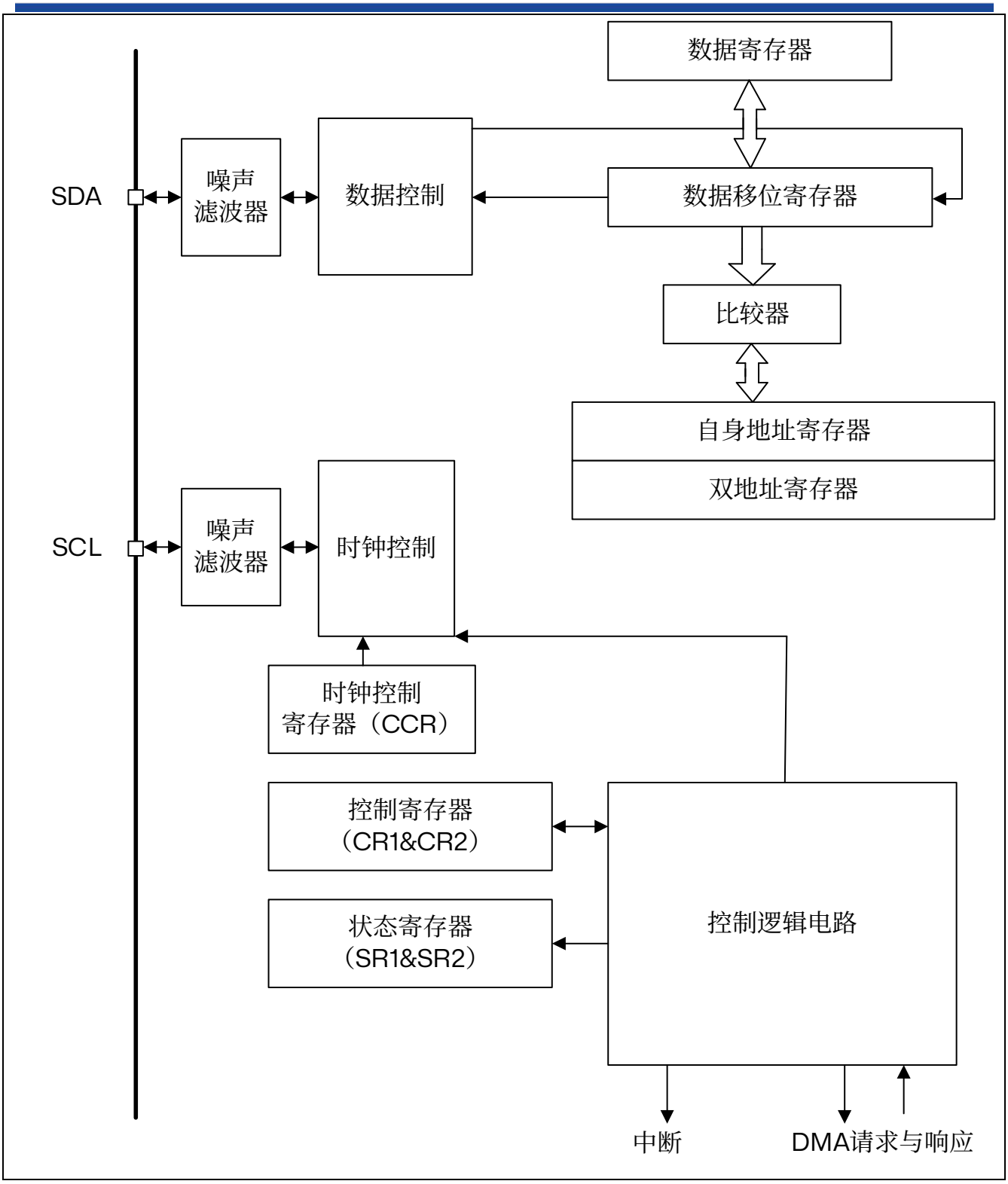


图 28.2 I²C 的功能框图

28.3.2 I²C 从模式

默认情况下，I²C 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。

为产生正确的时序，必须在 I2C_CR2 寄存器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 OAR1 和 OAR2（当 ENDUAL=1）或者广播呼叫地址（如果 ENGC=1）相比较。

注：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：I²C 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 ACK 位被置“1”，I²C 接口产生一个应答脉冲并等待 8 位从地址。

地址匹配：I²C 接口产生以下时序：

- 如果 ACK 被置‘1’，则产生一个应答脉冲
- 硬件设置 ADDR 位；如果设置了 ITEVFEN 位，则产生一个中断
- 如果 ENDUAL=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。在收到与地址匹配的头序列并且最低位为“1”（即 11110xx1）后，当接收到重复的起始条件时，将进入发送器模式。

在从模式下 TRA 位指示当前是处于接收器模式还是发送器模式。

从发送器

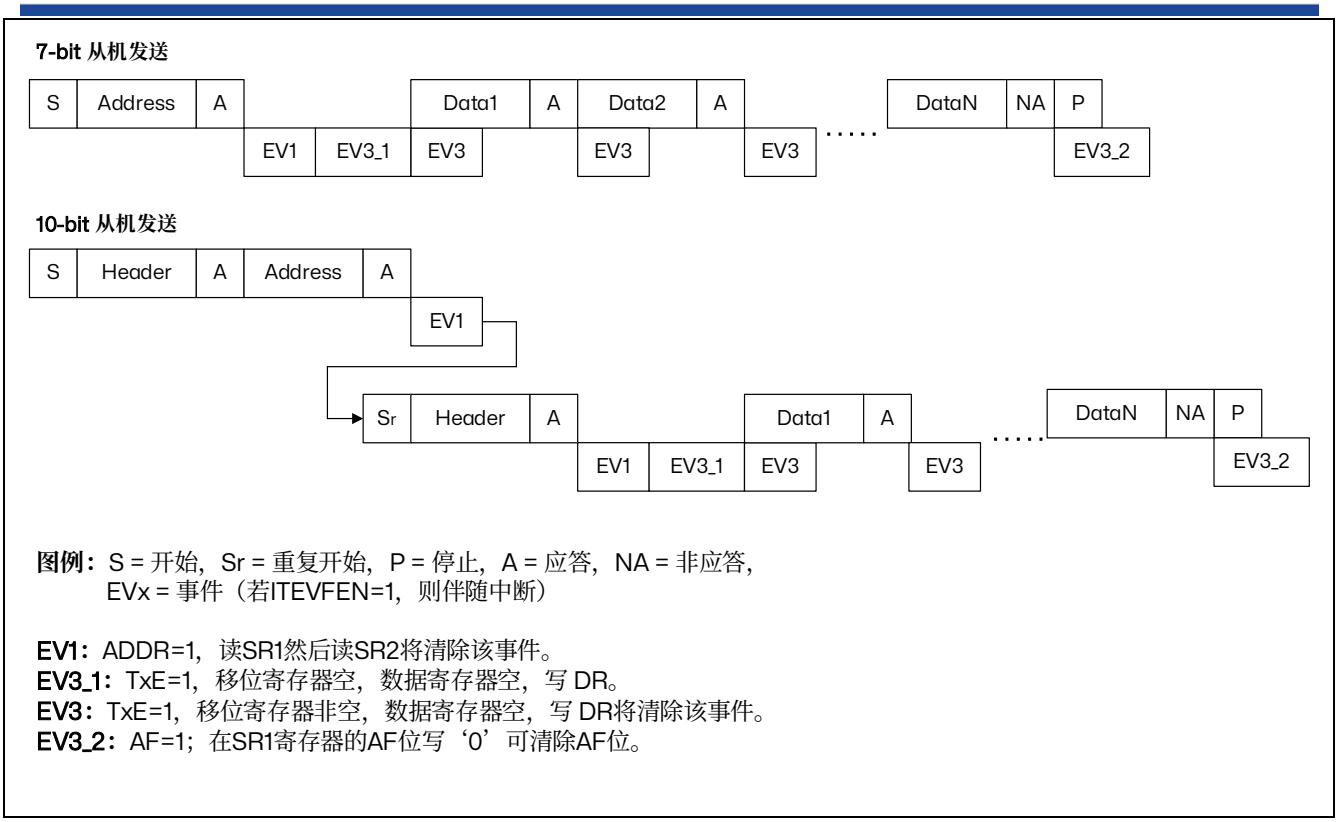
在接收到地址和清除 ADDR 位后，从发送器将字节从 DR 寄存器经由内部移位寄存器发送到 SDA 线上。

从设备保持 SCL 为低电平，直到 ADDR 位被清除并且待发送数据已写入 DR 寄存器。（见下图中的 EV1 和 EV3）。

当收到应答脉冲时：

- TxE 位被硬件置位，如果设置了 ITEVFEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 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_DR 寄存器，则 BTF 位被置位，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。



图例： S = 开始，Sr = 重复开始，P = 停止，A = 应答，NA = 非应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）

EV1： ADDR=1，读SR1然后读SR2将清除该事件。
EV3_1： TxE=1，移位寄存器空，数据寄存器空，写 DR。
EV3： TxE=1，移位寄存器非空，数据寄存器空，写 DR将清除该事件。
EV3_2： AF=1；在SR1寄存器的AF位写‘0’可清除AF位。

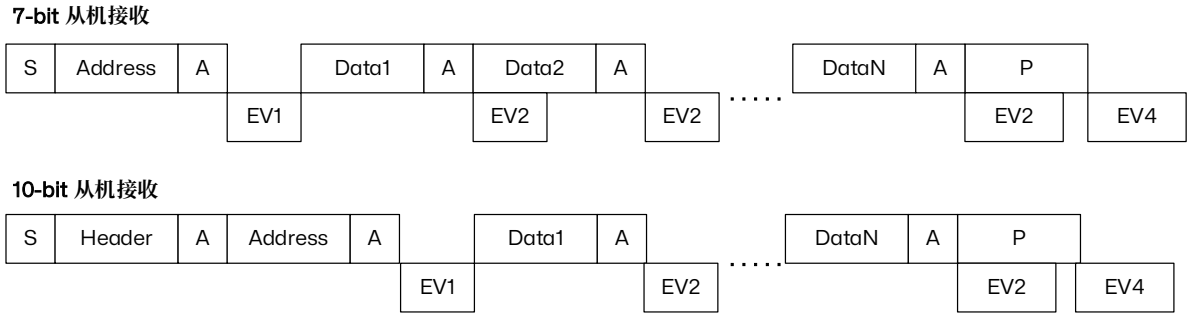
图 28.3 从发送器的传送序列图

从接收器

在接收到地址并清除 ADDR 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 DR 寄存器。I²C 接口在接收到每个字节后都执行下列操作：

- 如果设置了 ACK 位，则产生一个应答脉冲
- 硬件设置 RxNE=1。如果设置了 ITEVFEN 和 ITBUFEN 位，则产生一个中断。

如果 RxNE 被置位，并且在接收新的数据结束之前 DR 寄存器未被读出，BTF 位被置位，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。（见下图）。



图例： S = 开始，Sr = 重复开始，P = 停止，A = 应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）

EV1： ADDR=1，读SR1然后读SR2将清除该事件。
EV2： RxNE=1，读DR将清除该事件。
EV4： STOPF=1，读SR1然后写CR1寄存器将清除该事件。

图 28.4 从接收器的传送序列图

1. EV1 事件将 SCL 拉低，直到相应的软件序列结束。
2. EV2 软件序列必须在当前字节传输结束之前完成。
3. 在检查 SR1 寄存器内容后，用户应对每个已设置的标志执行完整的清除序列。
   因此，对于 ADDR 和 STOPF 标志，I2C 中断例程中需要以下序列：
   READ SR1
   if (ADDR == 1) {READ SR1; READ SR2}
   f (STOPF == 1) {READ SR1; WRITE CR1}
   目的是确保在发现 ADDR 和 STOPF 标志均被设置时，能够清除这两个标志。

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，I²C 接口检测到这一条件时：

- 设置 STOPF=1，如果设置了 ITEVFEN 位，则产生一个中断。

然后 I²C 接口等待读 SR1 寄存器，再写 CR1 寄存器。（见上图的 EV4）。

28.3.3 I²C 主模式

在主模式时，I²C 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 START 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_CR2 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_CR1 寄存器启动外设
- 置 I2C_CR1 寄存器中的 START 位为 1，产生起始条件

I²C 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

SCL 主机时钟生成

CCR 位用于生成 SCL 时钟的高电平和低电平，分别从上升沿和下降沿（分别）的生成开始。由于从设备可能会拉长 SCL 线，外设在上升沿生成后，会在 TRISE 位编程的时间结束时检查来自总线的 SCL 输入。

- 如果 SCL 线为低电平，表示有从设备正在拉长总线，高电平计数器停止计数，直到检测到 SCL 线为高电平。这可以确保 SCL 时钟参数的最小高电平周期。
- 如果 SCL 线为高电平，高电平计数器继续计数。

实际上，即使没有从设备拉长时钟，外设生成 SCL 上升沿到外设检测 SCL 上升沿的反馈循环也需要时间。这个反馈循环持续时间与 SCL 上升时间（影响 SCL VIH 输入检测）有关，加上 SCL 输入路径上噪声滤波器引起的延迟，以及 SCL 输入与 APB 时钟内部同步引起的延迟。反馈循环使用的最大时间在 TRISE 位中编程，以便无论 SCL 上升时间如何，SCL 频率都保持稳定。

起始条件

当 BUSY=0 时，设置 START=1，I²C 接口将产生一个开始条件并切换至主模式（MSL 位置位）。

*注：在主模式下，设置*START *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：

- SB 位被硬件置位，如果设置了 ITEVFEN 位，则会产生一个中断。

然后主设备等待读 SR1 寄存器，紧跟着将从地址写入 DR 寄存器。（见下两张图，传输序列 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：
  – ADD10 位被硬件置位，如果设置了 ITEVFEN 位，则产生一个中断。
  然后主设备等待读 SR1 寄存器，再将第二个地址字节写入 DR 寄存器（见下两张图，传输序列）。
  – ADDR 位被硬件置位，如果设置了 ITEVFEN 位，则产生一个中断。
  随后主设备等待一次读 SR1 寄存器，跟着读 SR2 寄存器（见下两张图，传输序列）。
- 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，
  – ADDR 位被硬件置位，如果设置了 ITEVFEN 位，则产生一个中断。
  随后主设备等待一次读 SR1 寄存器，跟着读 SR2 寄存器（见下两张图，传输序列）。

根据送出从地址的最低位（LSB），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（LSB）为‘0’。
  – 要进入接收器模式，主设备发送从地址时置最低位（LSB）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），然后送最低位为‘0’的从地址。（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），然后送最低位为‘1’的从地址。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。）

TRA 位指示主设备是在接收器模式还是发送器模式。

主发送器

在发送地址和清除 ADDR 位后，主设备通过内部移位寄存器将字节从 DR 寄存器发送到 SDA 线上。

主设备等待直到第一个数据字节被写入 I2C_DR（见下图，传输序列 EV8_1）。

当收到应答脉冲时，TxE 位被硬件置位，如果设置了 INEVFEN 和 ITBUFEN 位，则产生一个中断。

如果 TxE 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 DR 寄存器，则 BTF 被硬件置位，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。

关闭通信

在 DR 寄存器中写入最后一个字节后，通过设置 STOP 位产生一个停止条件（见下图，传输序列 EV8_2）。接口自动返回到从设备模式（MSL 位被清除）。

注：当 TxE 或 BTF 位置位时，停止条件应安排在出现 EV8_2 事件时。

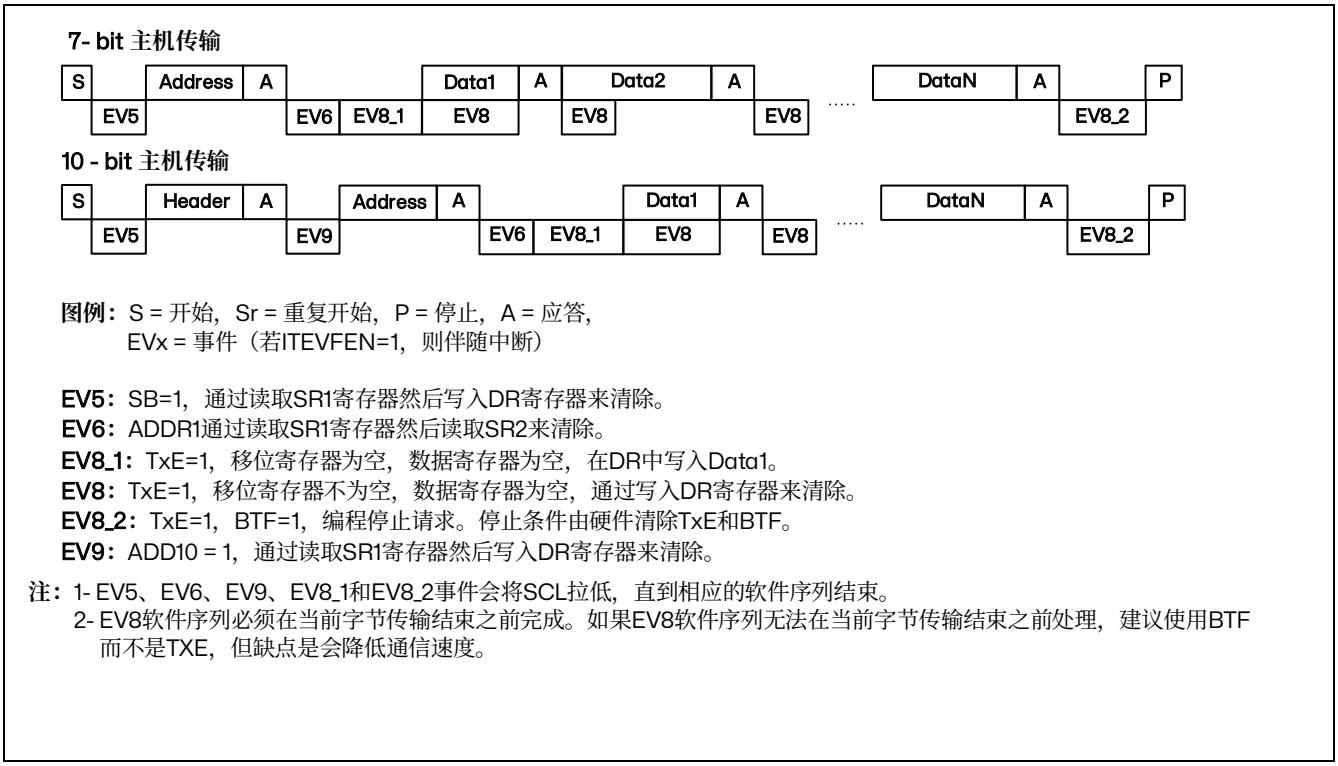


图 28.5 主发送器传送序列图

主接收器

在发送地址和清除 ADDR 之后，I²C 接口进入主接收器模式。在此模式下，I²C 接口从 SDA 线接收数据字节，并通过内部移位寄存器送至 DR 寄存器。在每个字节后，I²C 接口依次执行以下操作：

1. 如果 ACK 位被置位，发出一个应答脉冲。
2. 硬件设置 RxNE=1，如果设置了 INEVFEN 和 ITBUFEN 位，则会产生一个中断。（见下图，传输序列 EV7）。

如果 RxNE 位被置位，并且在接收新数据结束前，DR 寄存器中的数据没有被读走，硬件将设置 BTF=1，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再读出 I2C_DR 寄存器将清除 BTF 位。

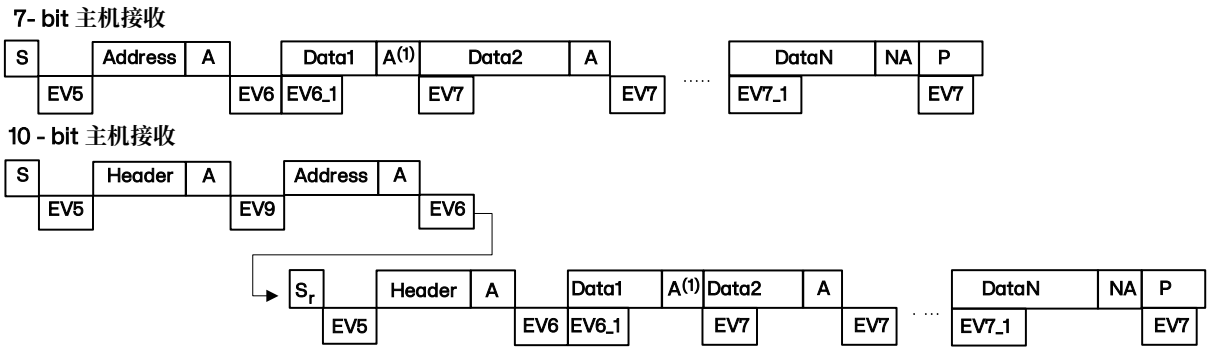
关闭通信

方法 1：此方法适用于 I2C 在应用中使用最高优先级中断的情况。

主设备在从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。

1. 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 RxNE 事件之后）必须清除 ACK 位。
2. 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 RxNE 事件之后）设置 STOP/START 位。
3. 只接收一个字节时，刚好在 EV6 之后（EV6_1 时，清除 ADDR 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，I²C 接口自动回到从模式（MSL 位被清除）。



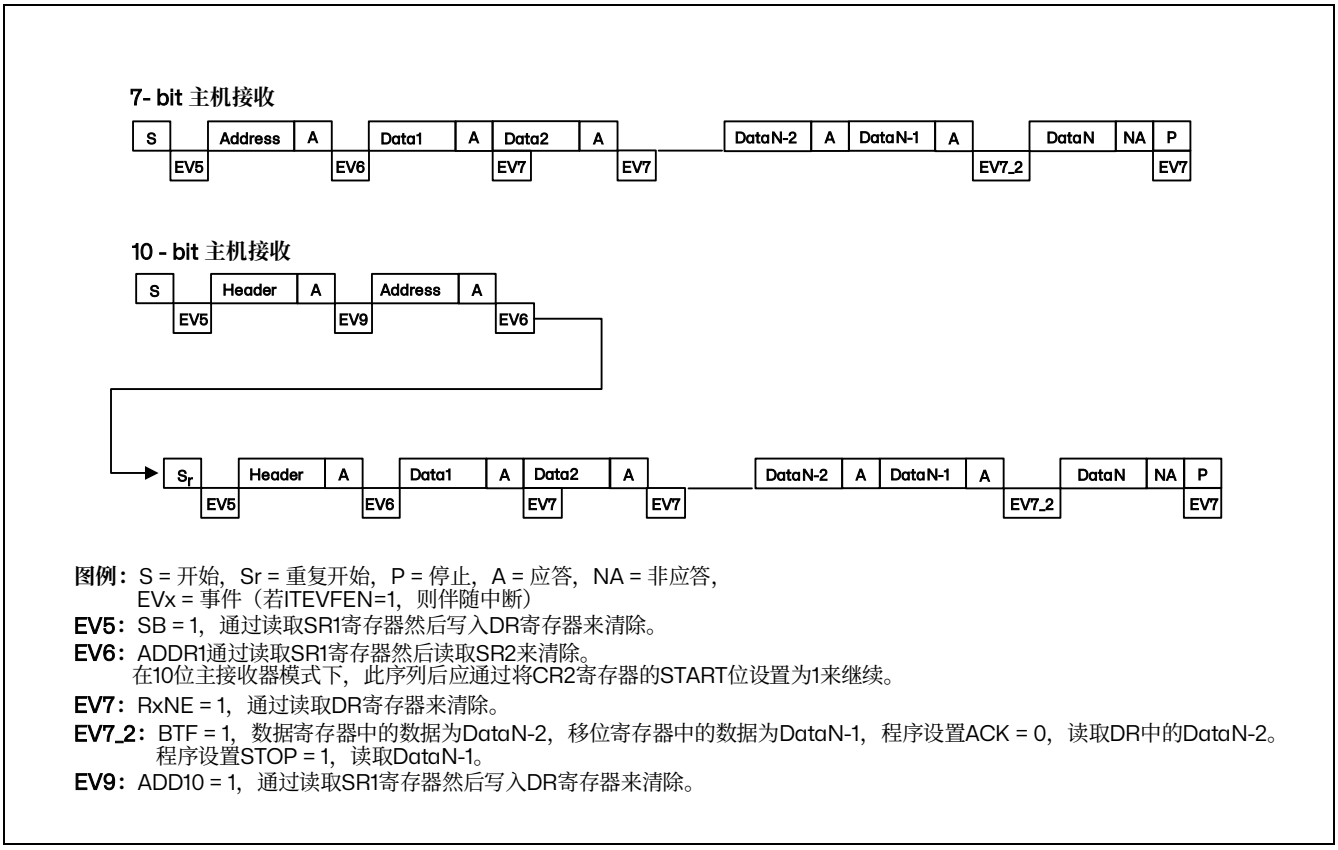
图例：S = 开始，Sr = 重复开始，P = 停止，A = 应答，NA = 非应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）
EV5：SB=1，通过读取SR1寄存器然后写入DR寄存器来清除。
EV6：ADDR1通过读取SR1寄存器然后读取SR2来清除。
在10位主接收器模式下，此序列后应通过将CR2寄存器的START位设置为1来继续。
EV6_1：无相关标志事件，仅用于接收1个字节。
在EV6之后，即ADDR被清除后，立即禁用应答并生成停止条件。
EV7：RxNE=1，通过读取DR寄存器来清除。
EV7_1：RxNE=1，通过读取DR寄存器来清除，同时编程ACK=0并请求STOP。
EV9：ADD10 = 1，通过读取SR1寄存器然后写入DR寄存器来清除。

图 28.6 方法 1：主接收器传送序列图

1. 如果收到一个单独的字节，则为 NA 非应答。
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。如果 EV7 软件序列无法在当前字节传输结束之前处理，建议使用 BTF 而不是 RXNE，但缺点是会降低通信速度。
4. EV6_1 或 EV7_1 的软件序列 必须在当前传输字节的 ACK 应答脉冲之前完成。

方法 2：此方法适用于 I2C 在应用中未使用最高优先级中断的情况或 I2C 采用轮询方式使用的情况。

在此方法中，不读取 DataN-2，因此在 DataN-1 之后，通信会被拉长（即 RxNE 和 BTF 均被置位）。然后，在读取数据寄存器（DR）中的 DataN-2 之前，清除应答（ACK）位，以确保在 DataN 应答脉冲之前清除该位。之后，紧接着读取 DataN-2 之后，设置停止（STOP）/开始（START）位，并读取 DataN-1。当 RxNE 被置位后，读取 DataN。以下是对此过程的图示说明：



图例：S = 开始，Sr = 重复开始，P = 停止，A = 应答，NA = 非应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）
EV5：SB = 1，通过读取SR1寄存器然后写入DR寄存器来清除。
EV6：ADDR1通过读取SR1寄存器然后读取SR2来清除。
在10位主接收器模式下，此序列后应通过将CR2寄存器的START位设置为1来继续。
EV7：RxNE = 1，通过读取DR寄存器来清除。
EV7_2：BTF = 1，数据寄存器中的数据为DataN-2，移位寄存器中的数据为DataN-1，程序设置ACK = 0，读取DR中的DataN-2。
程序设置STOP = 1，读取DataN-1。
EV9：ADD10 = 1，通过读取SR1寄存器然后写入DR寄存器来清除。

图 28.7 方法 2：N>2 时主接收器传送序列图

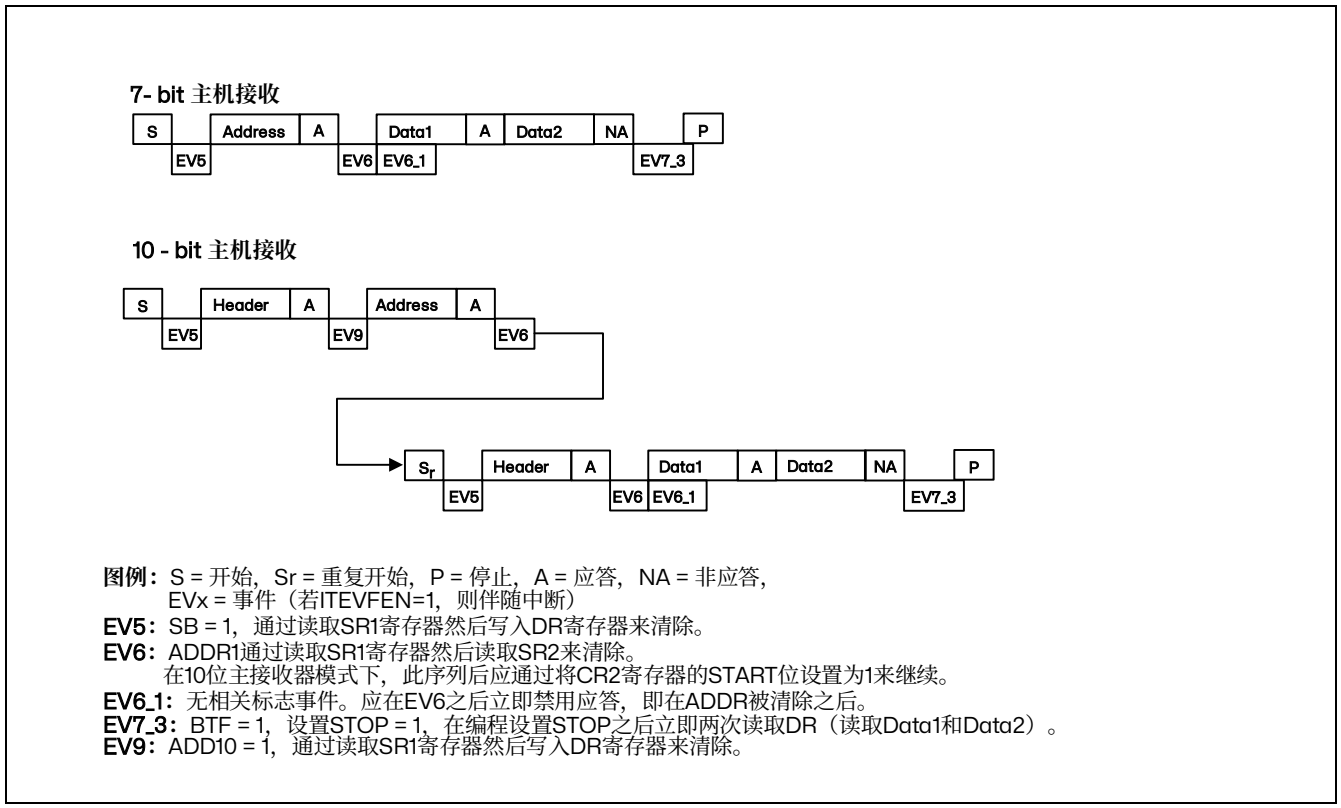
1. EV5、EV6 和 EV9 事件拉长 SCL 低电平，直到对应的软件序列结束。
2. EV7 的软件序列必须在当前字节传输结束前完成。如果 EV7 软件序列无法在当前字节传输结束之前处理，建议使用 BTF 而不是 RXNE，但缺点是会降低通信速度。

当剩余 3 个字节待读取时：

- RxNE = 1 => 无操作（DataN-2 尚未读取）。
- 已接收 DataN-1。
- BTF = 1，因为移位寄存器和数据寄存器均已满：DR（数据寄存器）中有 DataN-2，移位寄存器中有 DataN-1 => SCL 被拉低：总线上不会接收其他数据。
- 清除 ACK 位。
- 从 DR 中读取 DataN-2 => 这将启动移位寄存器中的 DataN 接收。
- 已接收 DataN（伴随 NACK）。
- 编程 START/STOP。
- 读取 DataN-1。
- RxNE = 1。
- 读取 DataN。

上述过程适用于 N>2 的情况。当需要接收单个字节或两个字节时，应按下述方式另行处理：

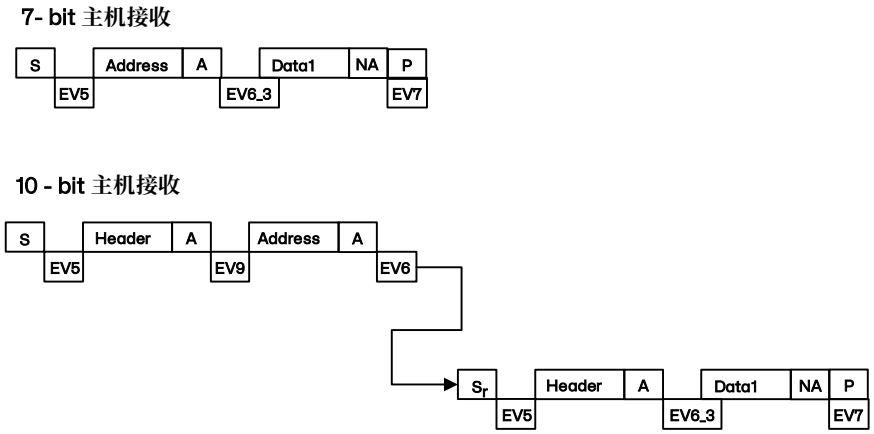
- 接收单个字节的情况：
  - 在 ADDR 事件中，清除 ACK 位
  - 清除 ADDR 标志
  - 编程设置 STOP/START 位
  - 在 RxNE 标志置位后读取数据
- 接收两个字节的情况：
  - 设置 POS 和 ACK 位
  - 等待 ADDR 标志置位
  - 清除 ADDR 标志
  - 清除 ACK 位
  - 等待 BTF 标志置位
  - 编程设置 STOP
  - 两次读取 DR



图例： S = 开始，Sr = 重复开始，P = 停止，A = 应答，NA = 非应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）
EV5： SB = 1，通过读取SR1寄存器然后写入DR寄存器来清除。
EV6： ADDR1通过读取SR1寄存器然后读取SR2来清除。
在10位主接收器模式下，此序列后应通过将CR2寄存器的START位设置为1来继续。
EV6_1： 无相关标志事件。应在EV6之后立即禁用应答，即在ADDR被清除之后。
EV7_3： BTF = 1，设置STOP = 1，在编程设置STOP之后立即两次读取DR（读取Data1和Data2）。
EV9： ADD10 = 1，通过读取SR1寄存器然后写入DR寄存器来清除。

图 28.8 方法 2：N=2 时主接收器传送序列图

1. EV5、EV6 和 EV9 事件拉长 SCL 低电平，直到对应的软件序列结束。
2. EV6_1 软件序列必须在当前字节传输的 ACK 脉冲之前完成。



图例： S = 开始，Sr = 重复开始，P = 停止，A = 应答，NA = 非应答，
EVx = 事件（若ITEVFEN=1，则伴随中断）
EV5： SB = 1，通过读取SR1寄存器然后写入DR寄存器来清除。
EV6： ADDR = 1，通过读取SR1寄存器然后读取SR2寄存器来清除。
EV6_3： ADDR = 1，设置ACK = 0，通过读取SR1寄存器然后读取SR2寄存器来清除ADDR，
在ADDR被清除后立即设置STOP = 1。
注： EV6_3软件序列必须在当前字节传输结束之前完成。
EV7： RxNE = 1，通过读取DR寄存器来清除。
EV9： ADD10 = 1，通过读取SR1寄存器然后写入DR寄存器来清除。

图 28.9 方法 2：N=1 时主接收器传送序列图

1. EV5、EV6 和 EV9 事件拉长 SCL 低电平，直到对应的软件序列结束。

28.3.4 错误条件

以下条件可能造成通讯失败。

总线错误（BERR）

在一个地址或数据字节传输期间，当 I²C 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：

- BERR 位被置位为“1”；如果设置了 ITERREN 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，从设备认为是一个重启动，并等待地址或停止条件。
  - 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（AF）

当接口检测到一个无应答位时，产生应答错误。此时：

- AF 位被置位，如果设置了 ITERREN 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重复起始条件。

仲裁丢失（ARLO）

当 I²C 接口检测到仲裁丢失时产生仲裁丢失错误，此时：

- ARLO 位被硬件置位，如果设置了 ITERREN 位，则产生一个中断；
- I²C 接口自动回到从模式（MSL 位被清除）。当 I²C 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（OVR）

在从模式下，如果禁止时钟延长，I²C 接口正在接收数据时，当它已经接收到一个字节（RxNE=1），但在 DR 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 RxNE 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，I²C 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 DR 寄存器（TxE=1），则发生欠载错误。此时：

- 在 DR 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 I²C 总线标准在规定的时间更新 DR 寄存器。

在发送第一个字节时，必须在清除 ADDR 之后并且第一个 SCL 上升沿之前写入 DR 寄存器；如果不能做到这点，则接收方应该丢弃第一个数据。

28.3.5 SDA/SCL 线控制

- 如果允许时钟延长：
  - 发送器模式：如果 TxE=1 且 BTF=1：I²C 接口在传输前保持时钟线为低，以等待软件读取 SR1，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 RxNE=1 且 BTF=1：I²C 接口在接收到数据字节后保持时钟线为低，以等待软件读 SR1，然后读数据寄存器 DR（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 RxNE=1，在接收到下个字节前 DR 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 TxE=1，在必须发送下个字节之前却没有新数据写进 DR，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。

28.3.6 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。必须在 ADDR 事件之前设置 I2C_CR2 寄存器中的 DMAEN 位。在主模式或从模式（且时钟延长启用时）下，也可以在 ADDR 事件期间、清除 ADDR 标志之前设置 DMAEN 位。必须在当前字节传输结束之前处理 DMA 请求。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 I2C 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 BTF 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_CR2 寄存器中设置了 LAST 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_CR2 寄存器中的 DMAEN 位可以激活 DMA 模式。只要 TxE 位被置位，数据将 DMA 从预置的存储区装载进 I2C_DR 寄存器。为 I²C 分配一个 DMA 通道，须执行以下步骤(x 是通道号):

1. 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器地址。数据将在每个 TxE 事件后从存储器传送至这个地址。
2. 在 DMA_CMARx 寄存器中设置存储器地址。数据在每个 TxE 事件后从这个存储区传送至 I2C_DR。
3. 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 TxE 事件后，此值将被递减。
4. 利用 DMA_CCRx 寄存器中的 PL[0:1]位配置通道优先级。
5. 设置 DMA_CCRx 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_CCRx 寄存器上的 EN 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 I2C 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：如果使用 DMA *进行发送时，不要设置* I2C_CR2 *寄存器的* ITBUFEN *位。*

利用 DMA 接收

通过设置 I2C_CR2 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_DR 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 I²C 接收，须执行以下步骤（x 是通道号）:

1. 在 DMA_CPARx 寄存器中设置 I2C_DR 寄存器的地址。数据将在每次 RxNE 事件后从此地址传送到存储区。
2. 在 DMA_CMARx 寄存器中设置存储区地址。数据将在每次 RxNE 事件后从 I2C_DR 寄存器传送到此存储区。
3. 在 DMA_CNDTRx 寄存器中设置所需的传输字节数。在每个 RxNE 事件后，此值将被递减。
4. 用 DMA_CCRx 寄存器中的 PL[0:1]配置通道优先级。
5. 清除 DMA_CCRx 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_CCRx 寄存器中的 EN 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 I²C 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：如果使用 DMA *进行接收时，不要设置* I2C_CR2 *寄存器的* ITBUFEN *位。*

28.4 I²C 中断请求

下表列出了所有的 I²C 中断请求

表 28.1 I²C 中断请求

中断事件	事件标志	开启控制位
起始位已发送（主）	SB	ITEVFEN
地址已发送（主）或地址匹配（从）	ADDR	
10 位头段已发送（主）	ADD10	
已收到停止（从）	STOPF	
数据字节传输完成	BTF	
接收缓冲区非空	RxNE	ITEVFEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失（主）	ARLO	
响应失败	AF	
过载/欠载	OVR	
超时/Tlow 错误	TIMEOUT	

注： SB、ADDR、ADD10、STOPF、BTF、RxNE *和* TxE *通过逻辑或汇到同一个中断通道中。*
BERR、ARLO、AF、OVR *和* TIMEOUT *通过逻辑或汇到同一个中断通道中。*

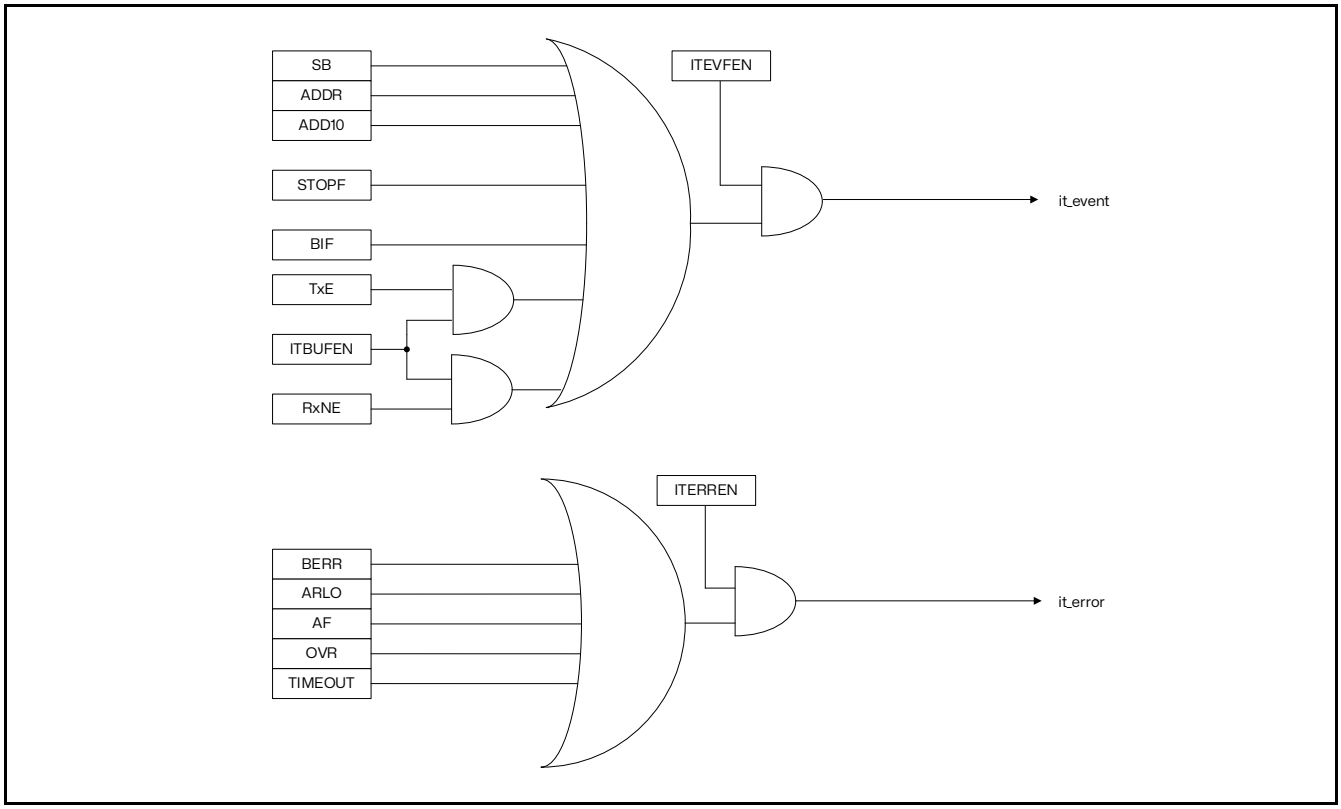


图 28.10 I²C 中断映射图

28.5 I²C 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
外设寄存器可以通过半字（16 位）或字（32 位）进行访问。

28.5.1 控制寄存器 1（I2C_CR1）

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
SWRST	Reserved			POS	ACK	STOP	START	NOSTRETCH	ENGC	Reserved					PE
rw				rw	rw	rw	rw	rw	rw						rw

Bit 15 **SWRST**：软件复位（Software reset）

当被置位时，I²C 处于复位状态。在复位该位前确信 I²C 的引脚被释放，总线是空的。

0：I²C 模块不处于复位状态；

1：I²C 模块处于复位状态。

注：当出现错误或锁定状态后，可使用此位重新初始化外设。例如，如果 BUSY *位已置*1 *但因母线干扰而不能复位，则可使用* SWRST *位退出此状态。*

Bit 14:12 保留，必须保持为复位值。

Bit 11 **POS**：应答（Acknowledge）

软件可以设置或清除该位，或当 PE=0 时，由硬件清除。

0：ACK 位控制当前移位寄存器内正在接收的字节的（N）ACK。

1：ACK 位控制在移位寄存器里接收的下一个字节的（N）ACK。

*注：当遵循接收*2 *个字节的程序时（见方法*2：N=2 *时主接收器传输时序图），会使用到* POS *位。*POS *位必须在数据接收开始之前配置好。在此情况下，若要对第*2 *个字节发送非应答（*NACK*），必须在* ADDR *（地址）位被清除后立即清除* ACK *（应答）位。*

Bit 10 **ACK**：应答使能（Acknowledge enable）

软件可以设置或清除该位，或当 PE=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）。

Bit 9 **STOP**：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：在当前字节传输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：在当前字节传输或释放 SCL 和 SDA 线。

Bit 8 **START**：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 PE=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。

Bit 7 **NOSTRETCH**：禁止时钟延长（从模式）（Clock stretching disable (Slave mode)）

该位用于当 ADDR 或 BTF 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。

Bit 6 **ENGC**：广播呼叫使能（General call enable）

0：禁止广播呼叫。以非应答响应地址 00h；

1：允许广播呼叫. 以应答响应地址 00h。

Bit 5:1 保留，必须保持为复位值。

Bit 0 **PE**：I²C 模块使能 （Peripheral enable）

0：禁用 I²C 模块；

1：使能 I²C 模块。

注：如果清除该位时通讯正在进行，在当前通讯结束后，I²C 模块被禁用并返回空闲状态。
由于在通讯结束后发生 PE =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： STOP *或* START *位置1 之后，在硬件将该位清零之前，软件不能对* I2C_CR1 *执行任何写操作。否则，可能会再次发出* STOP *或* START *置位请求。*

28.5.2 控制寄存器 2（I2C_CR2）

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
Reserved		I2CFLTSEL	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Reserved		FREQ[5:0]					
		rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw

Bit 15:14 保留，必须保持为复位值。

Bit 13 **I2CFLTSEL**：I2C 引脚滤波器选择（I2C pin filter select）

0：选择 I2C 50ns 滤波器；

1：禁用 I2C 50ns 滤波器。

Bit 12 **LAST**：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。

Bit 11 **DMAEN**：DMA 请求使能（DMA requests enable）

0：禁止 DMA 请求；

1：当 TxE=1 或 RxNE =1 时，允许 DMA 请求。

Bit 10 **ITBUFEN**：缓冲器中断使能（Buffer interrupt enable）

0：当 TxE=1 或 RxNE=1 时，不产生任何中断；

1：当 TxE=1 或 RxNE=1 时，产生事件中断（不管 DMAEN 是何种状态）。

Bit 9 **ITEVTEN**：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：

- SB = 1（主模式）
- ADDR = 1（主/从模式）
- ADD10 = 1（主模式）
- STOPF = 1（从模式）
- BTF = 1，但是没有 TxE 或 RxNE 事件
- 如果 ITBUFEN = 1，TxE 事件为 1
- 如果 ITBUFEN = 1，RxNE 事件为 1

Bit 8 **ITERREN**：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：

- BERR = 1
- ARLO = 1
- AF = 1
- OVR = 1
- TIMEOUT = 1

Bit 7:6 保留，必须保持为复位值。

Bit 5:0 **FREQ[5:0]**：外设时钟频率（Peripheral clock frequency）

外设利用 FREQ 位域生成符合 I2C 规范的数据建立时间和保持时间。最小运行频率为 2MHz，最大频率受 APB 频率限制，且不得超过 36MHz（外设固有最大限制）。

0b000000：禁用

0b000001：禁用

0b000010：2MHz

…

0b100100：36MHz

大于 100100：禁用。

28.5.3 自身地址寄存器 1（I2C_OAR1）

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
ADD MODE	Reserved					ADD[9:8]		ADD[7:1]							ADD0
rw						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15 **ADDMODE：**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）。

Bit 14 必须始终由软件保持为“1”。

Bit 13:10 保留，必须保持为复位值 0。

Bit 9:8 **ADD[9:8]：**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。

Bit 7:1 **ADD[7:1]：**接口地址（Interface address）

地址的 7~1 位。

Bit 0 **ADD0：**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。

28.5.4 自身地址寄存器 2（I2C_OAR2）

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
Reserved								ADD2[7:1]							ENDUAL
								rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:8 保留，必须保持为复位值。

Bit 7:1 **ADD2[7:1]：**接口地址（Interface address）

在双地址模式下地址的 7~1 位。

Bit 0 **ENDUAL：**双地址模式使能位（Dual addressing mode enable）

0：在 7 位地址模式下，只有 OAR1 被识别；

1：在 7 位地址模式下，OAR1 和 OAR2 都被识别。

28.5.5 数据寄存器（I2C_DR）

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
Reserved								DR[7:0]							
								rw	rw	rw	rw	rw	rw	rw	rw

Bit 15:8 保留，必须保持为复位值 0。

Bit 7:0 **DR[7:0]：** 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 DR 寄存器时，自动启动数据传输。一旦传输开始（TxE=1），如果能及时把下一个需传输的数据写入 DR 寄存器，I²C 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 DR 寄存器（RxNE=1）。在接收到下一个字节（RxNE=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 DR；

注：硬件不管理写冲突（如果 TxE=0，仍能写入数据寄存器）；

注：如果在处理 ACK 脉冲时发生 ARLO 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。

28.5.6 状态寄存器 1（I2C_SR1）

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
Res	TIME OUT	Reserved		OVR	AF	ARLO	BERR	TxE	RxNE	Res	STOPF	ADD10	BTF	ADDR	SB
	rc_w0			rc_w0	rc_w0	rc_w0	rc_w0	r	r		r	r	r	r	r

Bit 15 保留，必须保持为复位值。

Bit 14 **TIMEOUT：** 超时或 Tlow 错误（Timeout or Tlow error）

0：无超时错误；

1：SCL 处于低已达到 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 PE=0 时由硬件清除。

Bit 13:12 保留，必须保持为复位值。

Bit 11 **OVR：** 过载/欠载（Overrun/Underrun）

0：无过载/欠载；

1：出现过载/欠载。

– 当 NOSTRETCH=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 PE=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。

Bit 10 **AF**：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 PE=0 时由硬件清除。

Bit 9 **ARLO**：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，硬件将将置该位为“1”。

– 该位由软件写“0”清除，或在 PE=0 时由硬件清除。

在 ARLO 事件之后，I²C 接口自动切换回从模式（MSL=0）。

Bit 8 **BERR**：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 PE=0 时由硬件清除。

Bit 7 **TxE**：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 DR 寄存器可清除该位；或在发生一个起始或停止条件后，或当 PE=0 时由硬件自动清除。

如果收到一个 NACK，该位不被置位。

注：在写入第1 个要发送的数据后，或设置了BTF 时写入数据，都不能清除TxE 位，这是因为数据寄存器仍然为空。

Bit 6 **RxNE**：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 PE=0 时由硬件清除。

在发生 ARLO 事件时，RxNE 不被置位。

注：当设置了BTF 时，读取数据不能清除RxNE 位，因为数据寄存器仍然为满。

Bit 5 保留，必须保持为复位值。

Bit 4 **STOPF**：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 ACK=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 SR1 寄存器后，对 CR1 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。

注：在收到NACK 后，STOPF 位不被置位。
建议在STOPF 置1 后执行完整的清零序列（首先读取SR1，然后写入 CR1）。请参见图：从接收器的传输序列图。

Bit 3 **ADD10：**10 位头序列已发送（主模式）（10-bit header sent (Master mode)）

0：没有 ADD10 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 SR1 寄存器后，对 CR1 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。

注：收到一个NACK 后，ADD10 位不被置位。

Bit 2 **BTF：**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 NOSTRETCH=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（RxNE=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（TxE=1）。

– 在软件读取 SR1 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 PE=0 时，由硬件清除该位。

注：在收到一个NACK 后，BTF 位不会被置位。

Bit 1 **ADDR**：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 SR1 寄存器后，对 SR2 寄存器的读操作将清除该位，或当 PE=0 时，由硬件清除该位。

地址匹配（从模式）

0：地址不匹配或没有收到地址；

1：收到的地址匹配。

– 当收到的从地址与 OAR 寄存器中的内容相匹配、或发生广播呼叫，硬件就将该位置“1”（当对应的设置被使能时）。

注：在从模式下，建议在 ADDR *置*1 *后执行完整的清零序列（首先读取* SR1，*然后写入* SR2*）。请参见图：从接收器的传输序列图。*

地址已被发送（主模式）

0：地址发送没有结束；

1：地址发送结束。

– 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”。

– 7 位地址模式时，当收到地址的 ACK 后该位被置“1”。

注：在收到 NACK *后，* ADDR *位不会被置位。*

Bit 0 **SB**：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。

– 当发送出起始条件时该位被置“1”。

– 软件读取 SR1 寄存器后，写数据寄存器的操作将清除该位，或当 PE=0 时，硬件清除该位。

28.5.7 状态寄存器 2（I2C_SR2）

地址偏移：0x18

复位值：0x0000

注：读取 I2C_SR1 *后再读取* I2C_SR2 *可将* ADDR *标志清零，即使* ADDR *标志在读取* I2C_SR1 *之后置*1 *也如此。因此，必须仅在* I2C_SR1 *中的* ADDR *位已置*1 *或者* STOPF *位已清零后读取* I2C_SR2*。*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DUALF	Reserved		GEN CALL	Res	TRA	BUSY	MSL
								r			r		r	r	r

Bit 15:8 保留，必须保持为复位值。

Bit 7 **DUALF**：双标志（从模式）（Dual flag）

0：接收到的地址与 OAR1 内的内容相匹配；

1：接收到的地址与 OAR2 内的内容相匹配。

– 在产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件将该位清除。

Bit 6:5 保留，必须保持为复位值。

Bit 4 **GENCALL：**广播呼叫地址（从模式）（General call address）

0：未收到广播呼叫地址；

1：当 ENGC=1 时，收到广播呼叫的地址

– 在产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件将该位清除。

Bit 3 保留，必须保持为复位值。

Bit 2 **TRA：**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（ARLO=1）后，或当 PE=0 时，硬件将其清除。

Bit 1 **BUSY：**总线忙（Bus busy）

0：总线上无数据通讯；

1：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（PE=0）时该信息仍然被更新。

Bit 0 **MSL：**主从模式（Master/slave）

0：从模式；

1：主模式。

– 当接口处于主模式（SB=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（ARLO=1 时）、或当 PE=0 时，硬件清除该位。

注：读取 I2C_SR1 后再读取 I2C_SR2 可将 ADDR 标志清零，即使 ADDR 标志在读取 I2C_SR1 之后置 1 也如此。因此，必须仅在 I2C_SR1 中的 ADDR 位已置 1 或者 STOPF 位已清零后读取 I2C_SR2。

28.5.8 时钟控制寄存器（I2C_CCR）

地址偏移：0x1C

复位值：0x0000

注：f_{PCLK1} 必须至少为2MHz，才能达到标准模式 I²C 频率。必须至少为 4 MHz 才能达到快速模式 I²C 频率。必须为10MHz 的倍数才能实现最大 400 kHz 的 I²C 快速模式时钟。

CCR 寄存器只有在关闭 I²C 时（PE=0）才能设置。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F/S	DUTY	Reserved		CCR[11:0]											
rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 15 **F/S**：I²C 主模式选项（I2C master mode selection）

0：标准模式的 I²C；

1：快速模式的 I²C。

Bit 14 **DUTY**：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（参见 CCR）。

Bit 13:12 保留，必须保持为复位值。

Bit 11:0 **CCR[11:0]**：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。

在 I²C 标准模式下：

$T_{high} = CCR \times T_{PCLK1}$

$T_{low} = CCR \times T_{PCLK1}$

在 I²C 快速模式下：

如果 DUTY = 0：

$T_{high} = CCR \times T_{PCLK1}$

$T_{low} = 2 \times CCR \times T_{PCLK1}$

如果 DUTY = 1：

$T_{high} = 9 \times CCR \times T_{PCLK1}$

$T_{low} = 16 \times CCR \times T_{PCLK1}$

例如：在标准模式下，产生 100kHz 的 SCL 的频率：

如果 FREQ=08，T_{PCLK1} = 125ns，则 CCR 必须写入 0x28

（0x28 <=> 40d × 125ns = 5000 ns）。

注：*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；*

$t_{high} = t_{r(SCL)}+t_{w(SCLH)}$，*详见数据手册中对这些参数的定义；*

$t_{low} = t_{f(SCL)}+t_{w(SCLL)}$，*详见数据手册中对这些参数的定义；*

I2C *通信速率* $f_{SCL} \approx 1/(t_{high} + t_{low})$。*实际频率可能因模拟噪声滤波器输入延迟存在差异；*

只有在关闭 I²C 时（PE = 0）才能设置 CCR 寄存器；

28.5.9 TRISE 寄存器（I2C_TRISE）

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
Reserved										TRISE[5:0]					
										rw	rw	rw	rw	rw	rw

Bit 15:6 保留，必须保持为复位值。

Bit 5:0 **TRISE[5:0]：**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)）

这些位用于设置主模式中 SCL 反馈环路的最大持续时间，其目的是无论 SCL 上升沿持续时间如何变化，均能维持 SCL 频率的稳定性。

编程时需将 I2C 总线规范中规定的最大 SCL 上升时间值加 1 后写入该位域。

例如：标准模式中最大允许 SCL 上升时间为 1000ns。若 T_{PCLK1}=125ns，故 TRISE[5:0]中必须写入 09h（1000ns/125 ns = 8+1）。

滤波器的值也可以加到 TRISE[5:0]内。

如果结果不是一个整数，则将整数部分写入 TRISE[5:0]以确保 t_{HIGH} 参数。

注：只有当 I²C *被禁用（*PE=0*）时，才能设置* TRISE[5:0]。

29 通用异步收发器（UART）

29.1 UART 介绍

通用异步收发器（UART）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。UART 利用分数波特率发生器提供宽范围的波特率选择。它支持单向通信和半双工单线通信，也支持 LIN（局部互连网）。使用多缓冲器配置的 DMA 方式，可以实现高速数据通信。

29.2 UART 主要特性

- 全双工的，异步通信
- NRZ 标准格式
- 分数波特率发生器系统
- 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 主发送同步断开符的能力以及 LIN 从检测断开符的能力
  - 当 UART 硬件配置成 LIN 时，生成 13 位断开符；检测 10/11 位断开符
- 单线半双工通信
- 可配置的使用 DMA 的多缓冲器通信
  - 在 SRAM 里利用集中式 DMA 缓冲接收/发送字节
- 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
- 8 个带标志的中断源
  - 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
  - 溢出错误
  - 帧错误
  - 噪音错误
  - 校验错误

29.3 UART 功能概述

接口通过三个引脚与其他设备连接在一起。任何 UART 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：接收数据串行输入。通过过采样技术来区别数据和噪音，从而恢复数据。

TX：发送数据输出。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，并且未发送数据时，TX 引脚处于高电平；在发送完数据后，TX 引脚处于高阻态，需要配置内部上拉来保持 TX 引脚处于高电平。在单线模式里，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
- 1 或 2 个的停止位，由此表明数据帧的结束
- 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（UART_SR）
- 数据寄存器（UART_DR）
- 一个波特率寄存器（UART_BRR），12 位的整数和 4 位小数

关于以上寄存器中每个位的具体定义，请参考章节：UART 寄存器描述。

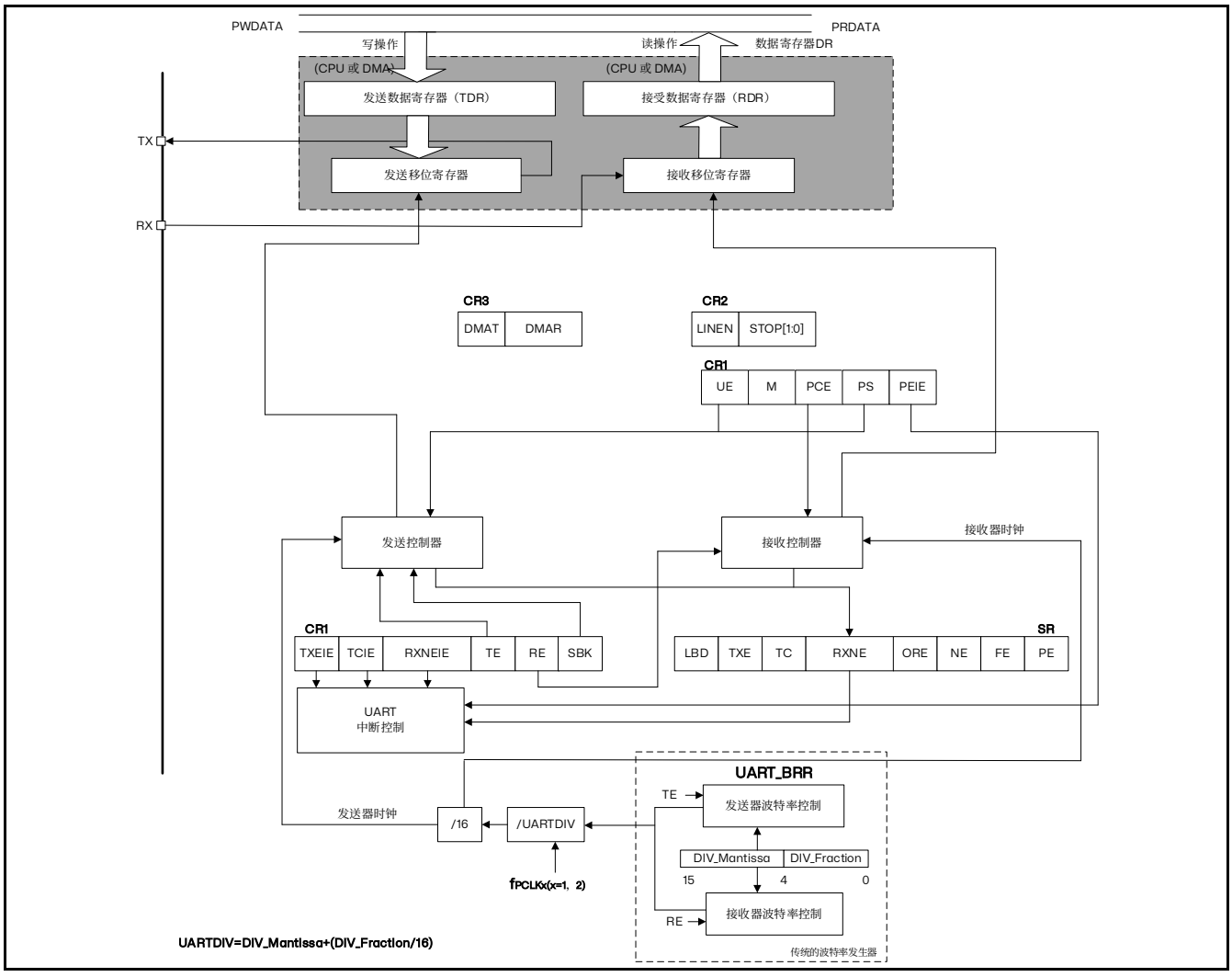


图 29.1 UART 框图

29.3.1 UART 特性描述

字长可以通过编程 UART_CR1 寄存器中的 M 位，选择成 8 或 9 位。在起始位期间，TX 脚处于低电平，在停止位期间处于高电平。

断开符号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。

发送和接收由一共用的波特率发生器驱动，当发送器和接收器的使能位分别置位时，分别为其产生时钟。

随后将有每个功能块的详细说明。

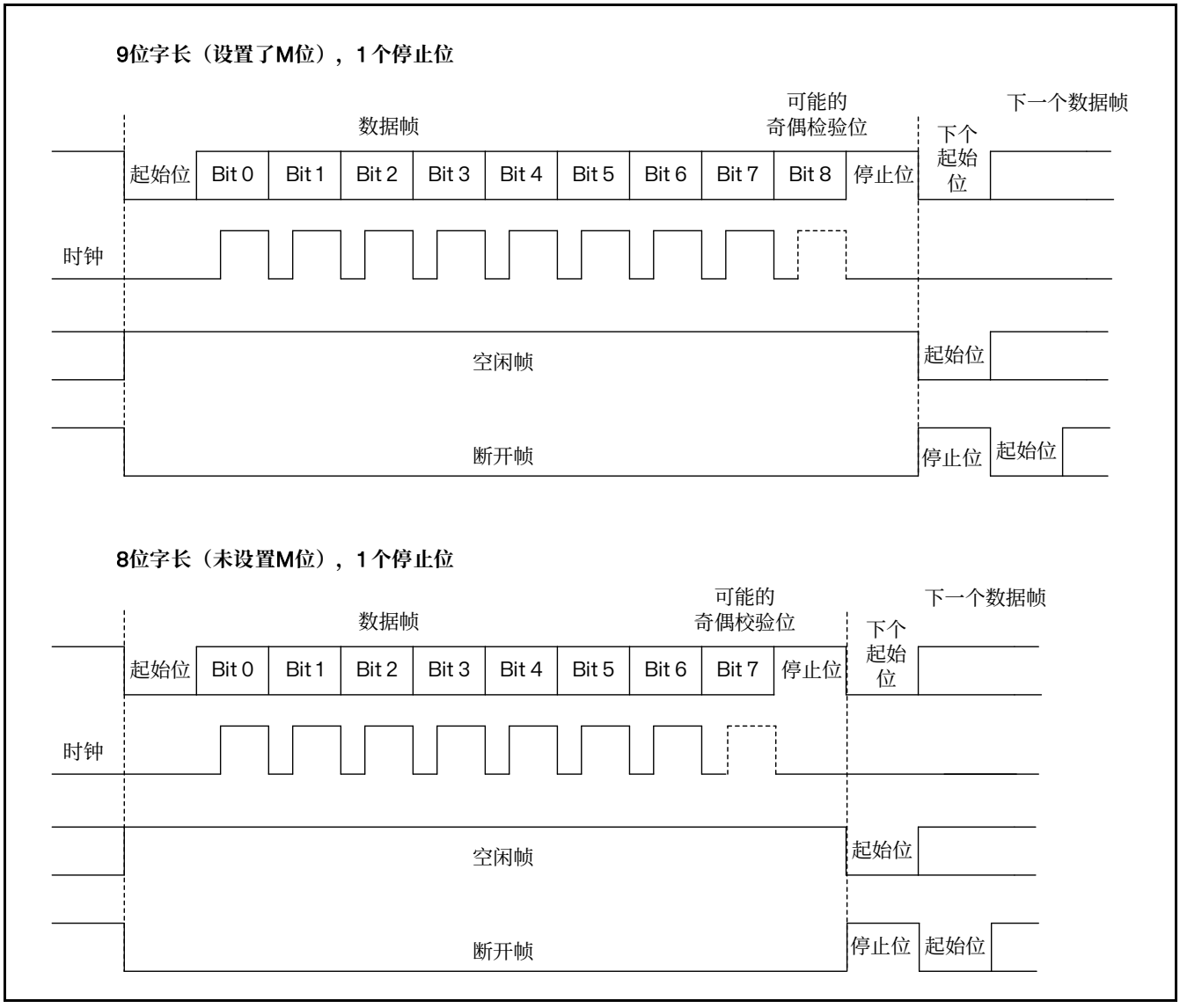


图 29.2 字长设置

29.3.2 发送器

发送器根据 M 位的状态发送 8 位或 9 位的数据字。当发送使能位（TE）被设置时，发送移位寄存器中的数据在 TX 脚上输出。

字符发送

在 UART 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，UART_DR 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。

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。

UART 支持多种停止位的配置：1 和 2 个停止位。

注：在数据传输期间不能复位 TE 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。

注：TE 位被激活后将发送一个空闲帧。

可配置的停止位

随每个字符发送的停止位的位数可以通过控制寄存器 2 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 UART 模式、单线模式。

空闲帧包括了停止位。

断开帧是 10 位低电平，后跟停止位（当 m=0 时）；或者 11 位低电平，后跟停止位（m=1 时）。不可能传输更长的断开帧（长度大于 10 或者 11 位）。

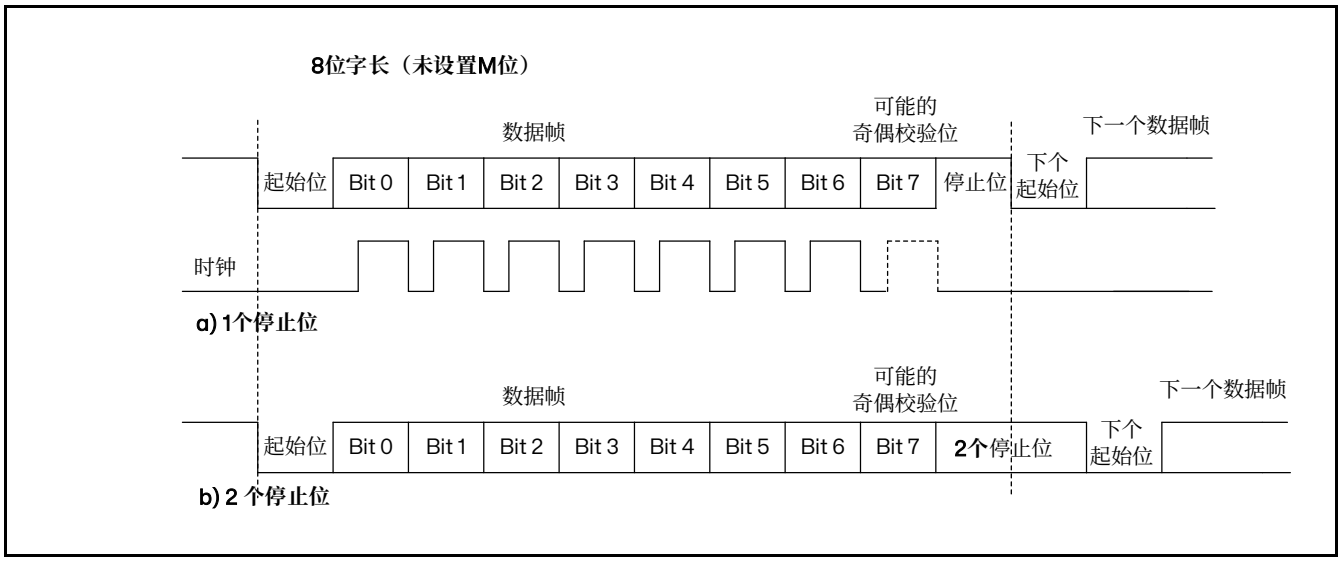


图 29.3 配置停止位

配置步骤：

1. 通过在 UART_CR1 寄存器上清除 UE 位来关闭 UART
2. 首先编程 UART_CR1 的 M 位来定义字长。
3. 通过在 UART_CR1 寄存器上置位 UE 位来激活 UART
4. 在 UART_CR2 中编程停止位的位数。
5. 如果采用多缓冲器通信，配置 UART_CR3 中的 DMA 使能位（DMAT）。按多缓冲器通信中的描述配置 DMA 寄存器。
6. 利用 UART_BRR 寄存器选择要求的波特率。
7. 设置 UART_CR1 中的 TE 位，发送一个空闲帧作为第一次数据发送。
8. 把要发送的数据写进 UART_DR 寄存器（此动作清除 TXE 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。
9. 在 UART_DR 寄存器中写入最后一个数据字后，要等待 TC=1，它表示最后一个数据帧的传输结束。当需要关闭 UART 或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次传输。

单字节通信

清零 TXE 位总是通过对数据寄存器的写操作来完成的。TXE 位由硬件来设置，它表明：

- 数据已经从 TDR 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 UART_DR 寄存器而不会覆盖先前的数据

如果 TXEIE 位被设置，此标志将产生一个中断。

如果此时 UART 正在发送数据，对 UART_DR 寄存器的写操作把数据存进 TDR 寄存器，并在当前传输结束时把该数据复制进移位寄存器。

如果此时 UART 没有在发送数据，处于空闲状态，对 UART_DR 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，TXE 位立即被置起。

当一帧发送完成时（停止位发送后）并且设置了 TXE 位，TC 位被置起，如果 UART_CR1 寄存器中的 TCIE 位被置起时，则会产生中断。

在 UART_DR 寄存器中写入了最后一个数据字后，在关闭 UART 模块之前或设置微控制器进入低功耗模式之前，必须先等待 TC=1。

使用下列软件过程清除 TC 位：

1. 读一次 UART_SR 寄存器；
2. 写一次 UART_DR 寄存器。

注：TC 位也可以通过软件对它写“0”来清除。此清零方式只推荐在多缓冲器通信模式下使用。

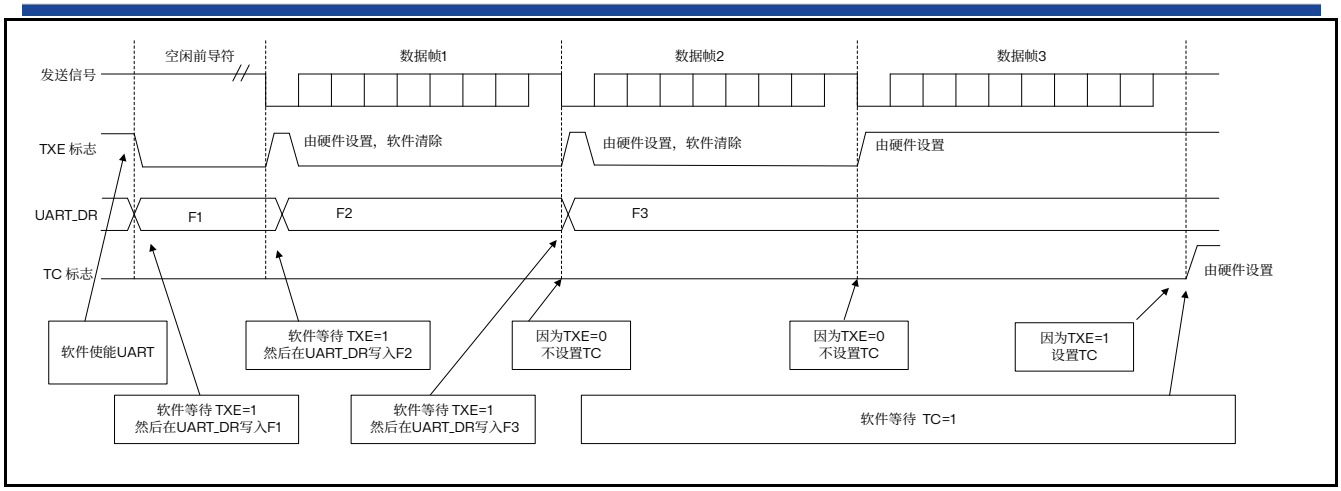


图 29.4 发送时 TC/TXE 的变化情况

断开符号

设置 SBK 可发送一个断开符号。断开帧长度取决 M 位。如果设置 SBK=1，在完成当前数据发送后，将在 TX 线上发送一个断开符号。断开字符发送完成时（在断开符号的停止位时）SBK 被硬件复位。UART 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：如果在开始发送断开帧之前，软件又复位了SBK 位，断开符号将不被发送。如果要发送两个连续的断开帧，SBK 位应该在前一个断开符号的停止位之后置起。

空闲符号

置位 TE 将使得 UART 在第一个数据帧前发送一空闲帧。

29.3.3 接收器

UART 可以根据 UART_CR1 的 M 位接收 8 位或 9 位的数据字。

起始位侦测

在 UART 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

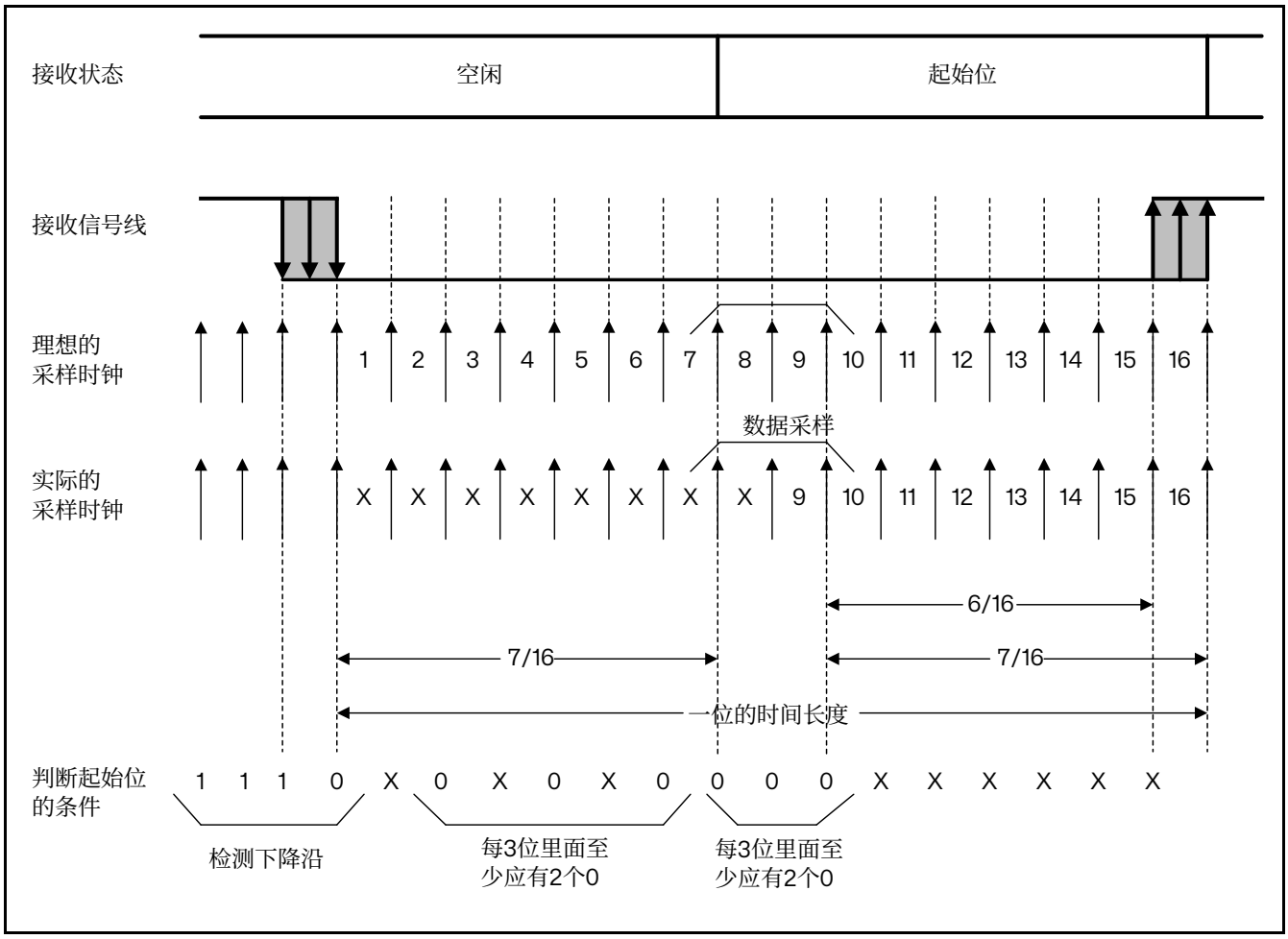


图 29.5 起始位侦测

注：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（不设置标志位）等待下降沿。

如果3 个采样点都为“0”（在第3、5、7 位的第一次采样，和在第8、9、10 的第二次采样都为“0”），则确认收到起始位，这时设置RXNE 标志位，如果RXNEIE=1，则产生中断。

如果两次3 个采样点上仅有2 个是“0”（第3、5、7 位的采样点和第8、9、10 位的采样点），那么起始位仍然是有效的，但是会设置NE 噪声标志位。如果不能满足这个条件，则中止起始位的侦测过程，接收器会回到空闲状态（不设置标志位）。

如果有一次3 个采样点上仅有2 个是“0”（第3、5、7 位的采样点或第8、9、10 位的采样点），那么起始位仍然是有效的，但是会设置NE 噪声标志位。

字符接收

在 UART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里，UART_DR 寄存器包含的缓冲器位于内部总线和接收移位寄存器之间。

配置步骤：

1. 通过在 UART_CR1 寄存器上清除 UE 位来关闭 UART
2. 首先编程 UART_CR1 的 M 位来定义字长。
3. 通过在 UART_CR1 寄存器上置位 UE 位来激活 UART
4. 在 UART_CR2 中编写停止位的个数
5. 如果需多缓冲器通信，选择 UART_CR3 中的 DMA 使能位（DMAR）。按多缓冲器通信所要求的配置 DMA 寄存器。
6. 利用波特率寄存器 UART_BRR 选择希望的波特率。
7. 设置 UART_CR1 的 RE 位。激活接收器，使它开始寻找起始位。

当一字符被接收到时

- RXNE 位被置位。它表明移位寄存器的内容被转移到 RDR。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 RXNEIE 位被设置，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，RXNE 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 UART_DR 寄存器完成对 RXNE 位清除。RXNE 标志也可以通过对它写 0 来清除。RXNE 位必须在下一字符接收结束前被清零，以避免溢出错误。

注：在接收数据时，RE 位不应该被复位。如果 RE 位在接收时被清零，当前字节的接收被丢失

断开符号

当接收到一个断开帧时，UART 像处理帧错误一样处理它。

溢出错误

如果 RXNE 还没有被复位，又接收到一个字符，则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。RXNE 标记是接收到每个字节后被置位的。如果下一个数据已被收到或先前 DMA 请求还没被服务时，RXNE 标志仍是置起的，溢出错误产生。

当溢出错误产生时：

- ORE 位被置位。
- RDR 内容将不会丢失。读 UART_DR 寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。随后接收到的数据都将丢失。
- 如果 RXNEIE 位被设置或 EIE 和 DMAR 位都被设置，中断产生。
- 顺序执行对 UART_SR 和 UART_DR 寄存器的读操作，可复位 ORE 位

注：当 ORE 位置位时，表明至少有 1 个数据已经丢失。有两种可能性：

- 如果 RXNE=1，上一个有效数据还在接收寄存器 RDR 上，可以被读出。
- 如果 RXNE=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 UART_SR 寄存器读访问和 UART_DR 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。

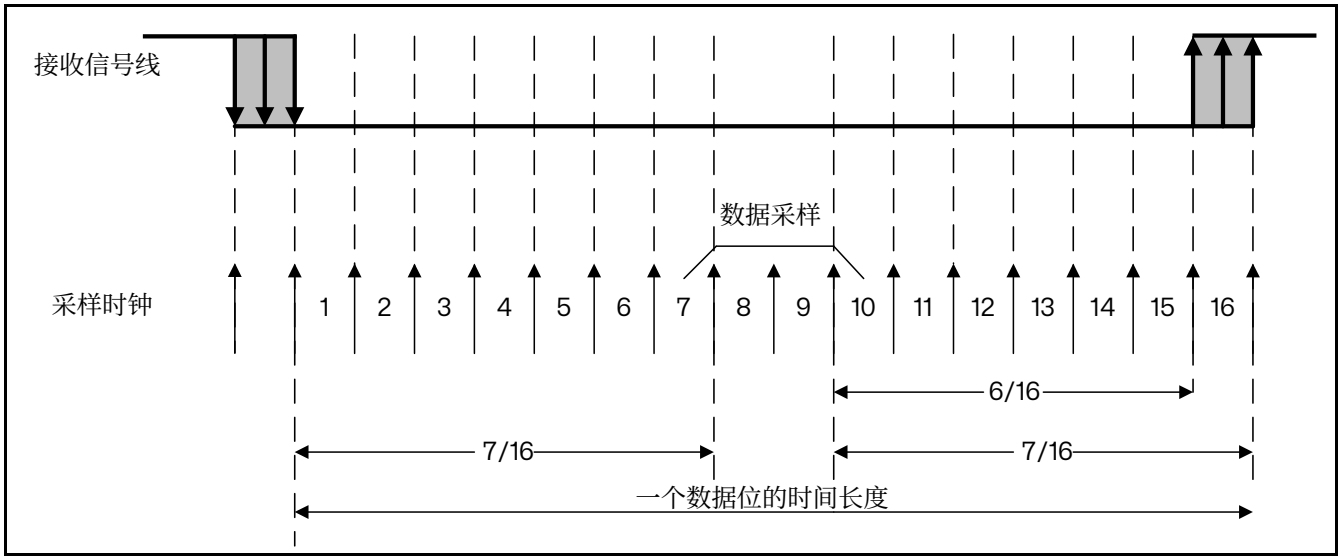


图 29.6 检测噪声的数据采样

表 29.1 检测噪声的数据采样

采样值	NE 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 RXNE 位的上升沿设置 NE 标志。
- 无效数据从移位寄存器传送到 UART_DR 寄存器。
- 在单个字节通信情况下，没有中断产生。然而，因为 NE 标志位和 RXNE 标志位是同时被设置，RXNE 将产生中断。在多缓冲器通信情况下，如果已经设置了 UART_CR3 寄存器中 EIE 位，将产生一个中断。

先读出 UART_SR，再读出 UART_DR 寄存器，将清除 NE 标志位

帧错误

当以下情况发生时检测到帧错误：

由于没有同步上或大量噪音的原因，停止位没有在预期的时间上接和收识别出来。

当帧错误被检测到时：

- FE 位被硬件置起
- 无效数据从移位寄存器传送到 UART_DR 寄存器。
- 在单字节通信时，没有中断产生。然而，这个位和 RXNE 位同时置起，后者将产生中断。在多缓

冲器通信情况下，如果 UART_CR3 寄存器中 EIE 位被置位的话，将产生中断。
顺序执行对 UART_SR 和 UART_DR 寄存器的读操作，可复位 FE 位。

接收期间的可配置的停止位

被接收的停止位的个数可以通过控制寄存器 2 的控制位来配置，在正常模式时，可以是 1 或 2 个。

1. **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。
2. **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 RXNE 标志将被设置。

29.3.4 分数波特率的产生

接收器和发送器的波特率在 UARTDIV 的整数和小数寄存器中的值应设置成相同。

$$\text{Tx / Rx 波特率} = \frac{f_{ck}}{(16*UARTDIV)}$$

这里的f_{ck} *是给外设的时钟（*PCLK1 *用于*UART2、3，PCLK2 *用于*UART1*）*

UARTDIV 是一个无符号的定点数。这 12 位的值设置在 UART_BRR 寄存器。

注：在写入 UART_BRR *之后，波特率计数器会被波特率寄存器的新值替换。因此，不要在通信进行中改变波特率寄存器的数值。*

如何从 UART_BRR 寄存器值得到 UARTDIV

示例1：

如果 DIV_Mantissa = 0d27，DIV_Fraction = 0d12（UART_BRR=0x1BC），于是
尾数（UARTDIV）= 0d27
小数（UARTDIV）= 12/16 = 0d0.75
所以 UARTDIV = 0d27.75

示例2：

要求 UARTDIV = 0d25.62,
就有：
DIV_Fraction = 16*0d0.62 = 0d9.92
最接近的整数是：0d10 = 0x0A
DIV_Mantissa = 尾数（0d25.620）= 0d25 = 0x19
于是，UART_BRR = 0x19A，因此 UARTDIV = 0d25.625

示例3：

要求 UARTDIV = 0d50.99
就有：
DIV_Fraction = 16*0d0.99 = 0d15.84
最接近的整数是：0d16 = 0x10 => DIV_frac[3:0]溢出 => 尾数必须添加进位
DIV_Mantissa = 尾数（0d50.990 + 进位）= 0d51 = 0x33
于是：UART_BRR = 0x330，UARTDIV = 0d51.000

表 29.2 设置波特率时的误差计算

波特率		f_{PCLK} = 76MHz			f_{PCLK} = 152MHz		
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	2.3999	1979.1875	0.0011%	2.4000	3958.3125	0.0005%
2	9.6	9.5995	494.8125	0.0042%	9.6002	989.5625	0.0021%
3	19.2	19.2016	247.375	0.0084%	19.1991	494.8125	0.0042%
4	38.4	38.4032	123.6875	0.0084%	38.4032	247.375	0.0084%
5	115.2	115.1515	41.25	0.0421%	115.2388	82.4375	0.0337%
6	230.4	230.3030	20.625	0.0421%	230.3030	41.25	0.0421%
7	460.8	460.6060	10.3125	0.0421%	460.6060	20.625	0.0421%
8	921.6	926.8292	5.125	0.5674%	921.2121	10.3125	0.0421%
9	4750	4750	1	0%	4750	2	0%
10	9500	不可能	不可能	不可能	9500	1	0%

注：CPU 时钟越低，特定波特率的精度就越低。可达到的波特率的上限可使用这些数据进行确定。

29.3.5 UART 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 UART 异步接收器能够容忍的范围，UART 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 < UART 接收器的容忍度

对于正常接收数据，UART 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 UART_CR1 寄存器的 M 位定义的 10 或 11 位字符长度
- 是否使用分数波特率产生

表 29.3 当 DIV_Fraction=0 时，UART 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 29.4 当 DIV_Fraction!=0 时，UART 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：在特殊的情况下，即当收到的帧包含一些在 M=0 时，正好是 10 位（M=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。

29.3.6 校验控制

当 UART_CR1 寄存器的 UE 使能位被清零时，通过设置 UART_CR1 寄存器上的 PCE 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）。

当 UART_CR1 寄存器的 UE 使能位被清零且 UART_CR1 寄存器的 PCE 校验控制使能位被置位时，通过设置 UART_CR1 寄存器上的 PS 位可选择是采用偶校验还是奇校验。

根据 M 位定义的帧长度，可能的 UART 帧格式列在下表中。

表 29.5 帧格式

M 位	PCE 位	UART 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：在用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 UART_CR1 中的 PS＝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 UART_CR1 中的 PS＝1），校验位将是“1”。

传输模式：如果 UART_CR1 的 PCE 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，UART_SR 寄存器中的 PE 标志被置“1”，并且如果 UART_CR1 寄存器的 PEIE 在被预先设置的话，中断产生。

29.3.7 LIN（局域互联网）模式

LIN 模式是通过设置 UART_CR2 寄存器的 LINEN 位选择。在 LIN 模式下，下列位必须保持为 0：

- UART_CR2 寄存器的 STOP[1:0]
- UART_CR3 寄存器的 HDSEL。

LIN 发送

UART 章节：发送器其中所描述的同样步骤适用于 LIN 主发送，但和正常 UART 发送有以下区别：

- 通过在 UART_CR1 寄存器上清除 UE 位来关闭 UART
- 清零 M 位以配置 8 位字长
- 通过在 UART_CR1 寄存器上置位 UE 位来激活 UART
- 置位 LINEN 位以进入 LIN 模式。这时，置位 SBK 将发送 13 位“0”作为断开符号。然后发一位“1”，以允许对下一个开始位的检测。

LIN 接收

当 LIN 模式被使能时，断开符号检测电路被激活。该检测完全独立于 UART 接收器。断开符号只要一出现就能检测到，不管是在总线空闲时还是在发送某数据帧其间，数据帧还未完成，又插入了断开符号的发送。

当接收器被激活时（UART_CR1 的 RE=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符号或数据是一样的。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 UART_CR2 的 LBDL = 0）或 11 个（当 UART_CR2 的 LBDL = 1）连续位都是“0”，并且又跟着一个定界符，UART_SR 的 LBD 标志被设置。如果 LBDIE 位=1，中断产生。在确认断开符号前，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，接收器继续如正常 UART 那样工作，不需要考虑检测断开符号。

如果 LIN 模式没有被激活（LINEN=0），接收器仍然正常工作于 UART 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，直到断开符号检测电路接收到一个“1”（这种情况发生于断开符号没有完整的发出来），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符号）。

下面两张图分别说明了断开符号检测器状态机的行为和断开符号标志的关系以及给出了一个断开帧的例子。

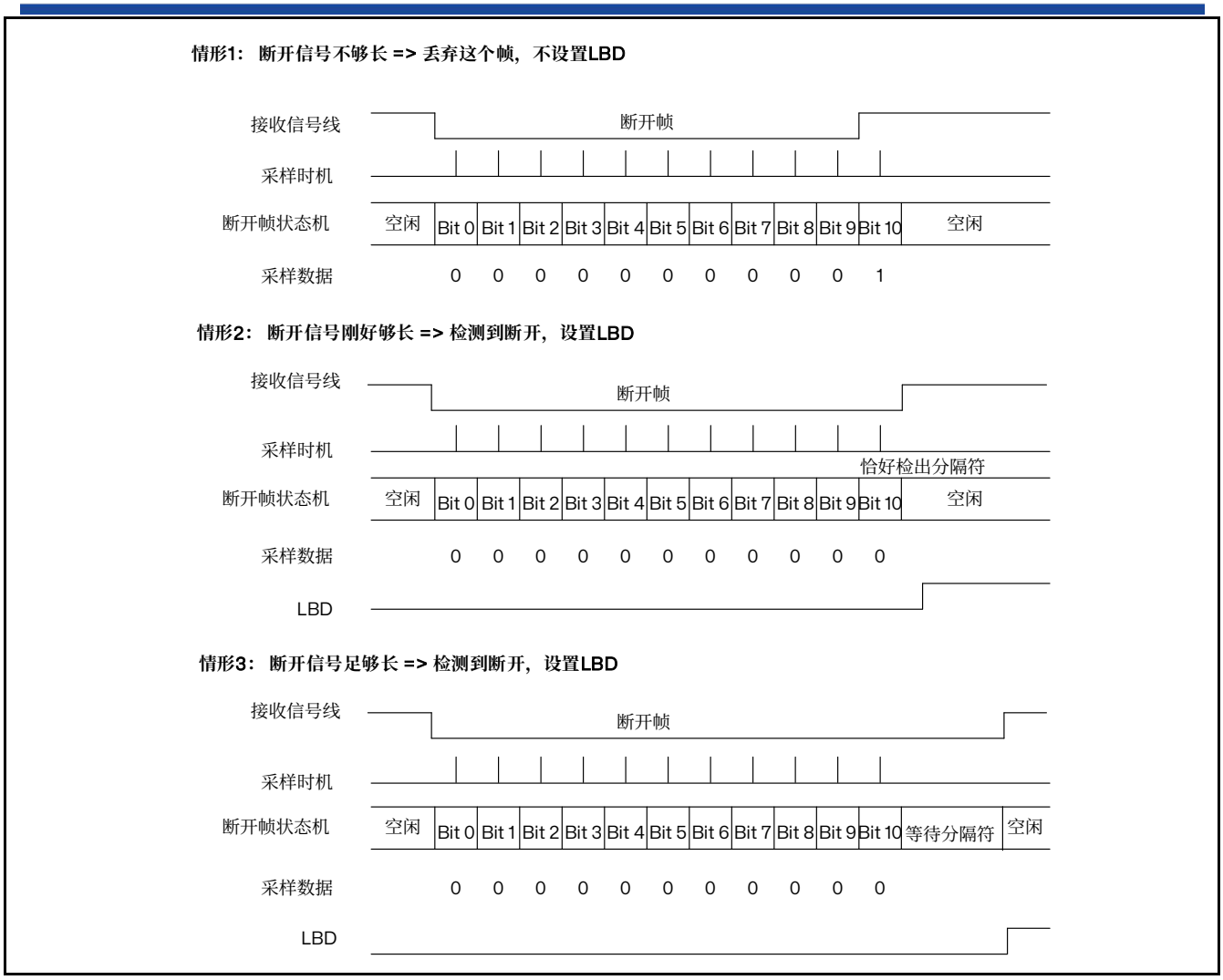


图 29.7 LIN 模式下的断开检测（11 位断开长度 – 设置了 LBDL 位）

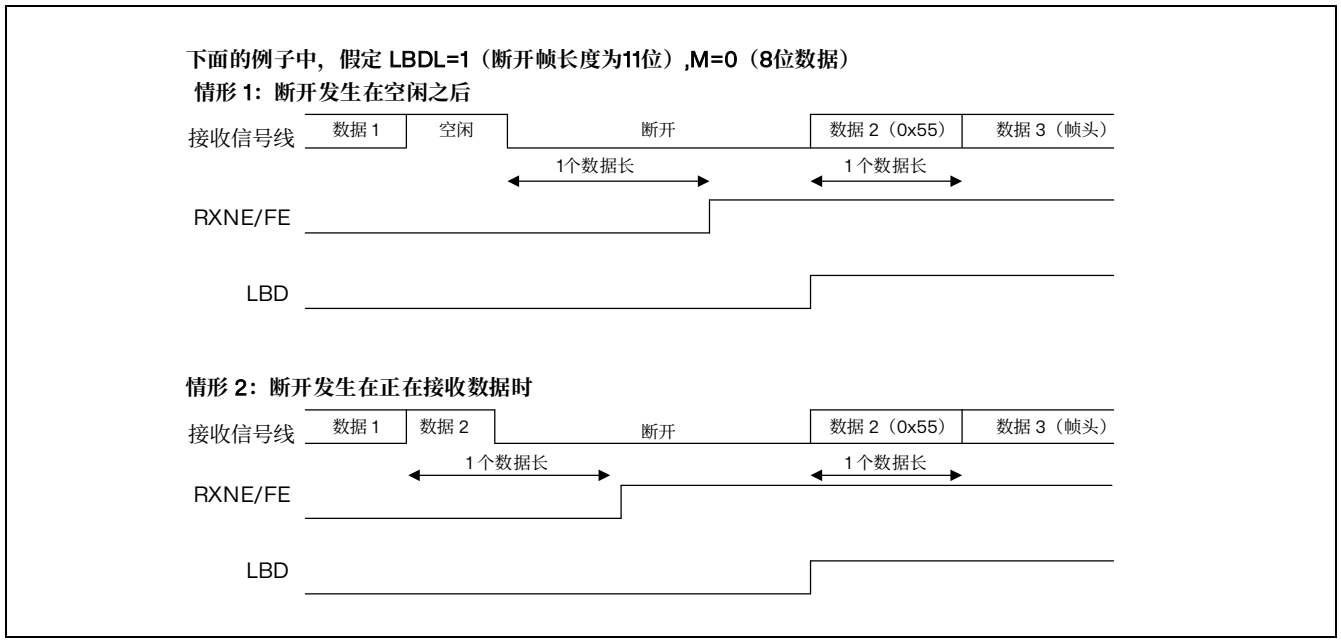


图 29.8 LIN 模式下的断开检测与帧错误的检测

29.3.8 单线半双工通信

单线半双工模式通过设置 UART_CR3 寄存器的 HDSEL 位选择。在这个模式里，下面的位必须保持清零状态：

- UART_CR2 寄存器的 LINEN 位

UART 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“HALF DUPLEX SEL”（UART_CR3 中的 HDSEL 位）选择半双工和全双工通信。当 HDSEL 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 UART 驱动时，必须配置成悬空输入（或开漏的输出高）。

除此以外，通信与正常 UART 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。当 TE 位被设置时，只要数据一写到数据寄存器上，发送就继续。

29.3.9 利用 DMA 连续通信

UART 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：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，应按照章节：发送器或者章节：接收器里所指定的方法是用* UART，*在* UART_SR *寄存器里，可以清零* TXE 或 RXNE 标志来实现连续通信。

利用 DMA 发送

用 DMA 进行发送，可以通过设置 UART_CR3 寄存器上的 DMAT 位激活。当 TXE 位被置为“1”时，DMA 就从指定的 SRAM 区传送数据到 UART_DR 寄存器。为 UART 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 UART_DR 寄存器地址配置成 DMA 传输的目的地址。在每个 TXE 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 TXE 事件后，将从此存储器区读出数据并传送到 UART_DR 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. 通过向 SR 寄存器中的 TC 位写入 0 来清除该位。
7. 在 DMA 寄存器上激活该通道。

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_ISR 寄存器的 TCIF 标志；监视 UART_SR 寄存器的 TC 标志可以确认 UART 通信是否结束，这样可以在关闭 UART 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 TXE=1，再等 TC=1。

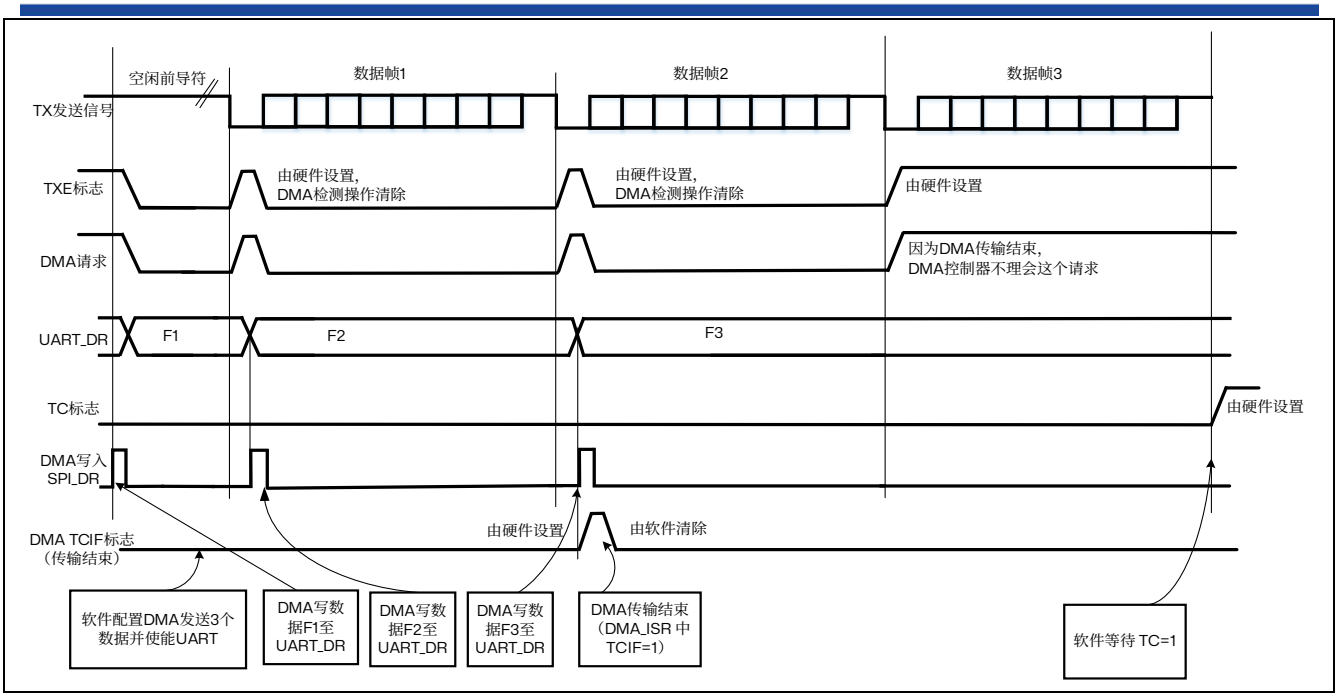


图 29.9 利用 DMA 发送

利用 DMA 接收

以通过设置 UART_CR3 寄存器的 DMAR 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 UART_DR 寄存器传送到指定的 SRAM 区（参考 DMA 相关说明）。为 UART 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 UART_DR 寄存器地址配置成传输的源地址。在每个 RXNE 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 RXNE 事件后，数据将从 UART_DR 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 通道的中断矢量上产生一中断。

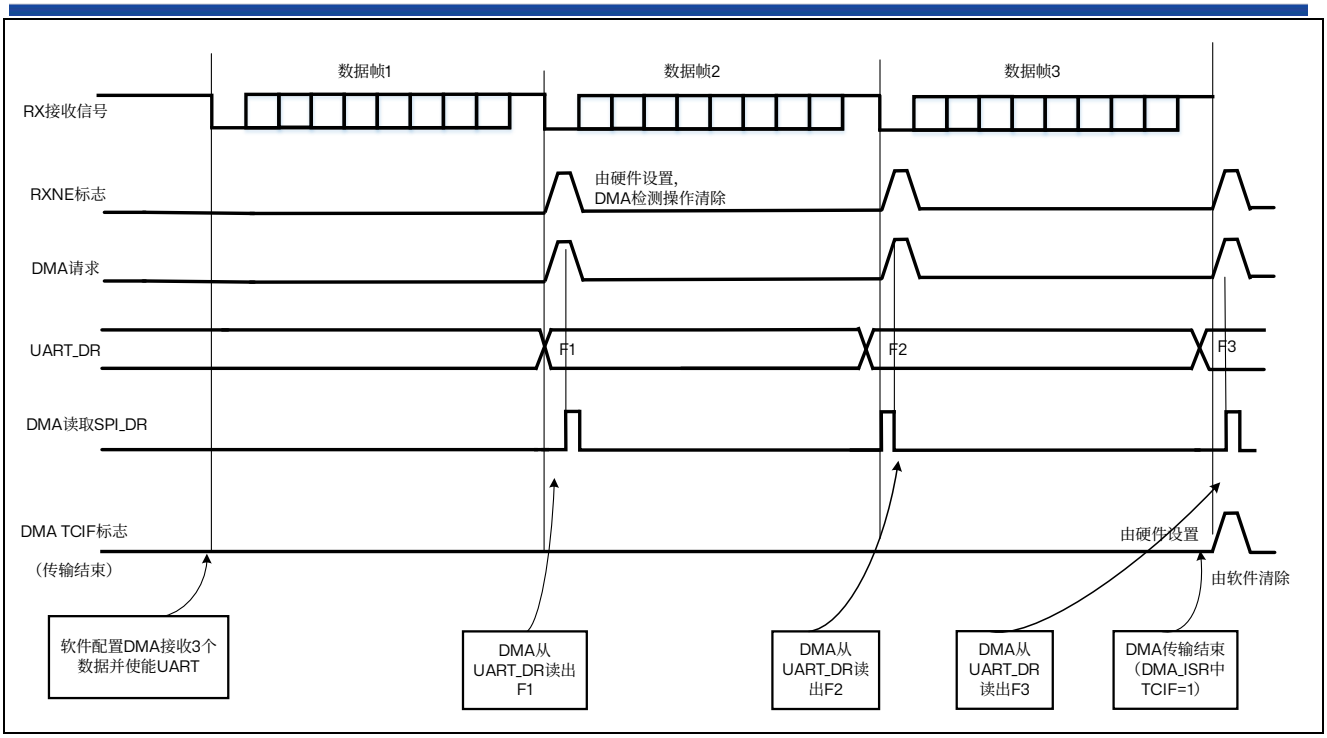


图 29.10 利用 DMA 接收

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 RXNE 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。

29.4 UART 中断请求

表 29.6 UART 中断请求

中断事件	事件标志	使能位
发送数据寄存器空	TXE	TXEIE
发送完成	TC	TCIE
接收数据就绪可读	RXNE	RXNEIE
检测到数据溢出	ORE	
奇偶检验错	PE	PEIE
断开标志	LBD	LBDIE
噪声标志，多缓冲通信中的溢出错误和帧错误	NE 或 ORT 或 FE	EIE(1)

1. 仅当使用 DMA 接收数据时，才使用这个标志位。

UART 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、清除发送、发送数据寄存器非空。
- 接收期间：溢出错误、接收数据寄存器非空、校验错误、LIN 断开符号检测、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。

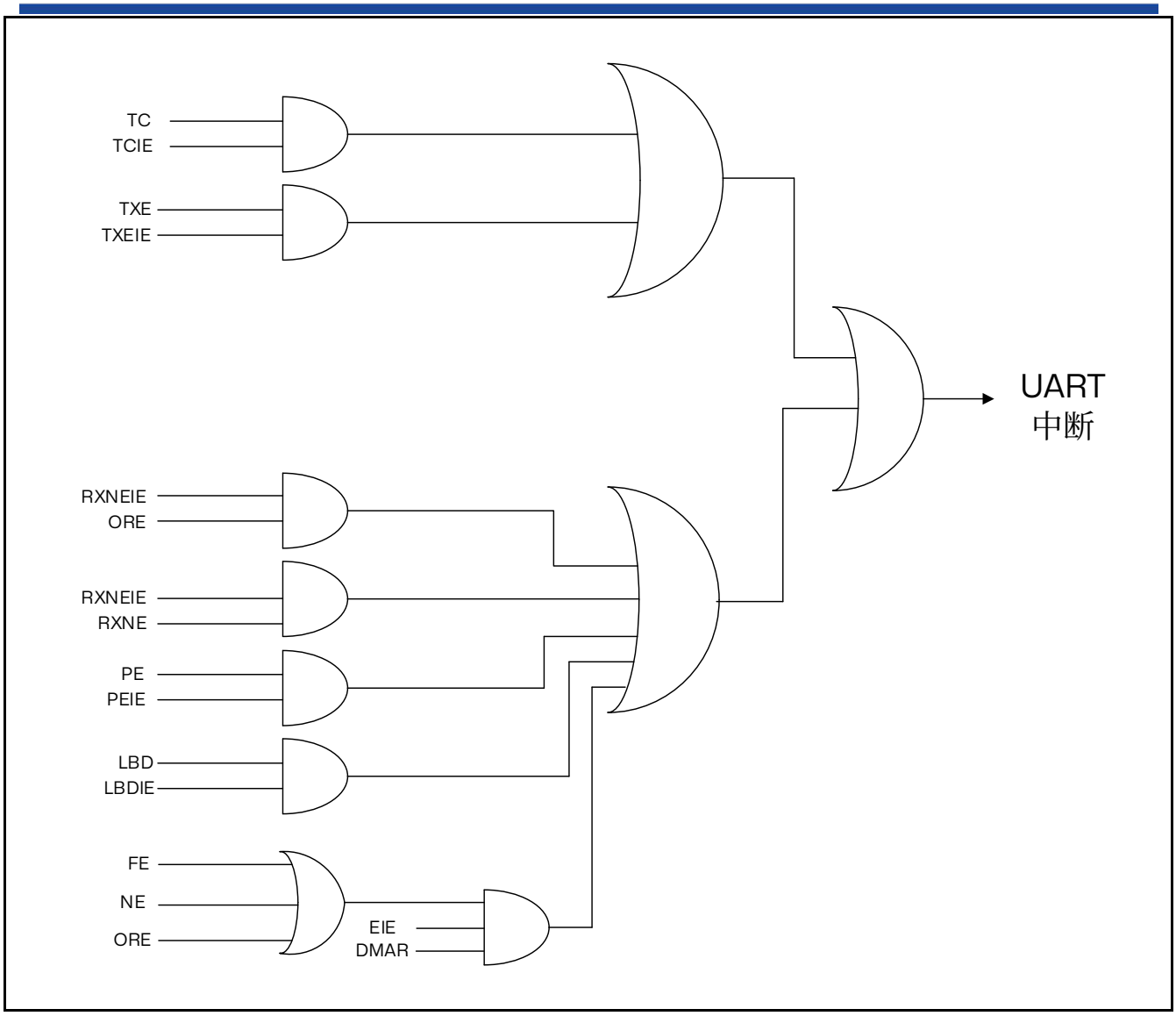


图 29.11 UART 中断映像图

29.5 UART 模式配置

表 29.7 UART 模式设置 (1)

UART 模式	UART1	UART2	UART3
异步模式	×	×	×
多缓存通讯（DMA）	×	×	×
半双工（单线模式）	×	×	×
LIN	×	×	×

1. ×=支持；NA = 不支持该应用

29.6 UART 寄存器描述

在寄存器描述中使用的缩写列表请参考章节 2.1。
可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

29.6.1 状态寄存器（UART_SR）

地址偏移：0x00
复位值：0x0000 00C0

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							LBD	TXE	TC	RXNE	Res	ORE	NE	FE	PE
							rc_w0	r	rc_w0	rc_w0		r	r	r	r

Bit 31:9 保留，必须保持为复位值。

Bit 8 **LBD**：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 UART_CR2 中的 LBDIE = 1，则产生中断。
0：没有检测到 LIN 断开；
1：检测到 LIN 断开。

注：若 LBDIE=1，当 LBD 为“1”时要产生中断。

Bit 7 **TXE**：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 UART_CR1 寄存器中的 TXEIE 为 1，则产生中断。对 UART_DR 的写操作，将该位清零。
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：单缓冲器传输中使用该位。

Bit 6 **TC**：发送完成（Transmission complete）
当包含有数据的一帧发送完成后，并且 TXE=1 时，由硬件将该位置“1”。如果 UART_CR1 中的 TCIE 为“1”，则产生中断。由软件序列清除该位（先读 UART_SR，然后写入 UART_DR）。TC 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。

Bit 5 **RXNE**：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 UART_DR 寄存器中，该位被硬件置位。如果 UART_CR1 寄存器中的 RXNEIE 为 1，则产生中断。对 UART_DR 的读操作可以将该位清零。RXNE 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。

Bit 4 保留，必须保持为复位值。

Bit 3 **ORE**：过载错误（Overrun error）

当 RXNE 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 UART_CR1 中的 RXNEIE 为“1”的话，则产生中断。由软件序列将其清零（先读 UART_SR，然后读 UART_CR）。

0：没有过载错误；

1：检测到过载错误。

注：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EIE 位，在多缓冲器通信模式下，ORE 标志置位会产生中断的。

Bit 2 **NE**：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 UART_SR，再读 UART_DR）。

0：没有检测到噪声；

1：检测到噪声。

注：该位不会产生中断，因为它和RXNE 一起出现，硬件会在设置RXNE 标志时产生中断。在多缓冲区通信模式下，如果设置了EIE 位，则设置NE 标志时会产生中断。

Bit 1 **FE**：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 UART_SR，再读 UART_DR）。

0：没有检测到帧错误；

1：检测到帧错误或者 break 符。

注：该位不会产生中断，因为它和RXNE 一起出现，硬件会在设置RXNE 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ORE 标志位。

注：在多缓冲区通信模式下，如果设置了EIE 位，则设置FE 标志时会产生中断。

Bit 0 **PE**：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 UART_SR 和 UART_DR）。在清除 PE 位前，软件必须等待 RXNE 标志位被置“1”。如果 UART_CR1 中的 PEIE 为“1”，则产生中断。

0：没有奇偶校验错误；

1：奇偶校验错误。

29.6.2 数据寄存器（UART_DR）

地址偏移：0x04

复位值：不确定

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

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

Bit 31:9 保留，必须保持为复位值。

Bit 8:0 **DR[8:0]：**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。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（UART_CR1 中 PCE 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）会被后来的校验位该取代。

当使能校验位进行接收时，读到的 MSB 位是接收到的校验位。

29.6.3 波特比率寄存器（UART_BRR）

*注：如果*TE *或*RE *被分别禁止，波特计数器停止计数*

地址偏移：0x08

复位值：0x0000 0000

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

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

Bit 31:16 保留，必须保持为复位值。

Bit 15:4 **DIV_Mantissa[11:0]：**UARTDIV 的整数部分

这 12 位定义了 UART 分频器除法系数（UARTDIV）的整数部分。

Bit 3:0 **DIV_Fraction[3:0]：**UARTDIV 的小数部分

这 4 位定义了 UART 分频器除法系数（UARTDIV）的小数部分。

29.6.4 控制寄存器 1（UART_CR1）

地址偏移：0x0C

复位值：0x0000 0000

31	31	30	29	28	27	26	25	24	23	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		UE	M	Res	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	Res	TE	RE	Res	SBK
		rw	rw		rw	rw	rw	rw	rw	rw		rw	rw		rw

Bit 31:14 保留，必须保持为复位值。

Bit 13 **UE**：UART 使能（UART enable）

当该位被清零，在当前字节传输完成后 UART 的分频器和输出停止工作，以减少功耗。该位由软件设置和清零。

0：UART 分频器和输出被禁止；

1：UART 模块使能。

注：当 UE 位使能时，Bit12 M，Bit 10 PCE，Bit9 PS 位不能修改。

Bit 12 **M**：字长（Word length）

该位定义了数据字的长度，由软件对其设置和清零

0：一个起始位，8 个数据位，n 个停止位；

1：一个起始位，9 个数据位，n 个停止位。

注：当 UE 位使能时，不能修改这个位。

注：在数据传输过程中（发送或者接收时），不能修改这个位。

注：当使能校验控制 PCE 时，校验位的插入会导致数据帧长度改变。数据帧长度根据 M 位定义，可能的 UART 帧格式请参考章节 30.3.6 校验控制中的帧格式表格。

Bit 11 保留，必须保持为复位值。

Bit 10 **PCE**：校验控制使能（Parity control enable）

用该位选择是否进行硬件校验控制（对于发送来说就是校验位的产生；对于接收来说就是校验位的检测）。当使能了该位，在发送数据的最高位（如果 M=1，最高位就是第 9 位；如果 M=0，最高位就是第 8 位）插入校验位；对接收到的数据检查其校验位。软件对它置“1”或清“0”。一旦设置了该位，当前字节传输完成后，校验控制才生效。

0：禁止校验控制；

1：使能校验控制。

注：当 UE 位使能时，不能修改这个位。

Bit 9 **PS**：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。

注：当 UE 位使能时，不能修改这个位。

Bit 8 **PEIE**：PE 中断使能（PE interrupt enable）

该位由软件设置或清除。

0：禁止产生中断；

1：当 UART_SR 中的 PE 为“1”时，产生 UART 中断。

Bit 7 **TXEIE**：发送缓冲区空中断使能（TXE interrupt enable）

该位由软件设置或清除。

0：禁止产生中断；

1：当 UART_SR 中的 TXE 为“1”时，产生 UART 中断。

Bit 6 **TCIE**：发送完成中断使能（Transmission complete interrupt enable）

该位由软件设置或清除。

0：禁止产生中断；

1：当 UART_SR 中的 TC 为“1”时，产生 UART 中断。

Bit 5 **RXNEIE**：接收缓冲区非空中断使能（RXNE interrupt enable）

该位由软件设置或清除。

0：禁止产生中断；

1：当 UART_SR 中的 ORE 或者 RXNE 为“1”时，产生 UART 中断。

Bit 4 保留，必须保持为复位值。

Bit 3 **TE**：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：在数据传输过程中，如果 TE 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）。

注：当 TE 被设置后，在真正发送开始之前，有一个比特时间的延迟。

Bit 2 **RE**：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。

Bit 1 保留，必须保持为复位值。

Bit 0 **SBK**：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位它，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。

29.6.5 控制寄存器 2（UART_CR2）

地址偏移：0x10

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	LINEN	STOP[1:0]		Reserved					LBDIE	LBDL	Reserved				
	rw	rw	rw						rw	rw					

Bit 31:15 保留，必须保持为复位值。

Bit 14 **LINEN**：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。

Bit 13:12 **STOP**：停止位（STOP bits）

这 2 位用来设置停止位的位数

00：1 个停止位；

10：2 个停止位；

其他：保留

Bit 11:7 保留，必须保持为复位值。

Bit 6 **LBDIE**：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 UART_SR 寄存器中的 LBD 为“1”就产生中断。

Bit 5 **LBDL**：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。

Bit 4:0 保留，必须保持为复位值。

29.6.6 控制寄存器 3（UART_CR3）

地址偏移：0x14

复位值：0x0000 0000

31	31	30	29	28	27	26	25	24	23	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DMAT	DMAR	Reserved		HDSEL	Reserved		EIE
								rw	rw			rw			rw

Bit 31:8 保留，必须保持为复位值。

Bit 7 **DMAT：**DMA 使能发送（DMA enable transmitter）

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；

Bit 6 **DMAR：**DMA 使能接收（DMA enable receiver）

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；

Bit 5:4 保留，必须保持为复位值。

Bit 3 **HDSEL：**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。

Bit 2:1 保留，必须保持为复位值。

Bit 0 **EIE：**错误中断使能（Error interrupt enable）

0：禁止中断；

1：只要 UART_CR3 中的 DMAR=1，并且 UART_SR 中的 FE=1，或者 ORE=1，或者 NE=1，则产生中断。

30 调试支持（DBG）

30.1.1 对定时器和看门狗的调试支持

断点期间，必须选择定时器和看门狗的计数器的行为方式：

- 在产生断点时，计数器继续计数。例如，当 PWM 控制电机时，通常需要这种方式。
- 在产生断点时，计数器停止计数。用于看门狗时需要这种方式。

30.1.2 MCU 设备 ID 代码

RX32H710 集成了一个 MCU ID 代码。该 ID 用于标识 RX MCU 的器件型号和晶圆版本。

调试器/编程器工具应仅使用 DEV_ID[11:0]进行器件识别。

DBGMCU_IDCODE

地址：0xE004 2000

访问：仅支持 32 位访问。只读

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REV_ID[15:0]															
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
Reserved				DEV_ID[11:0]											
				r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:16 **REV_ID[15:0]：**版本标识符

该位域表示器件版本：

0x2003：版本 A

Bit 15:12 保留，必须保持为复位值。

Bit 11:0 **DEV_ID[15:0]：**器件标识符

器件 ID 如下：

0x050：RX32H710

30.1.3 调试 MCU 控制寄存器（DBGMCU_CR）

地址：0xE004 2004
上电复位（POR）：0x0000 0000
系统复位：无影响
访问：仅支持 32 位访问

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					DBG_TIMB_STOP	DBG_TIMA_STOP	DBG_TIM15_STOP	Reserved			DBG_TIM7_STOP	DBG_TIM6_STOP	Res	DBG_TIM8_STOP	Res
					rw	rw	rw				rw	rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DBG_TIM3_STOP	DBG_TIM2_STOP	DBG_TIM1_STOP	Res	DBG_IWDG_STOP	Reserved					DBG_STANDBY	DBG_STOP	DBG_SLEEP
			rw	rw	rw		rw						rw	rw	rw

Bit 31:27 保留，必须保持为复位值。

Bit 26 **DBG_TIMB_STOP**：内核停止时 TIMB 计数器停止
0：即使内核停止，仍然馈送 TIMB 计数器的时钟
1：内核停止时停止 TIMB 计数器的时钟

Bit 25 **DBG_TIMA_STOP**：内核停止时 TIMA 计数器停止
0：即使内核停止，仍然馈送 TIMA 计数器的时钟
1：内核停止时停止 TIMA 计数器的时钟

Bit 24 **DBG_TIM15_STOP**：内核停止时 TIM15 计数器停止
0：即使内核停止，仍然馈送 TIM15 计数器的时钟
1：内核停止时停止 TIM15 计数器的时钟

Bit 23:21 保留，必须保持为复位值。

Bit 20 **DBG_TIM7_STOP**：内核停止时 TIM7 计数器停止
0：即使内核停止，仍然馈送 TIM7 计数器的时钟
1：内核停止时停止 TIM7 计数器的时钟

Bit 19 **DBG_TIM6_STOP**：内核停止时 TIM6 计数器停止
0：即使内核停止，仍然馈送 TIM6 计数器的时钟
1：内核停止时停止 TIM6 计数器的时钟

Bit 18 保留，必须保持为复位值。

Bit 17 **DBG_TIM8_STOP**：内核停止时 TIM8 计数器停止
0：即使内核停止，仍然馈送 TIM8 计数器的时钟
1：内核停止时停止 TIM8 计数器的时钟

Bit 16:13 保留，必须保持为复位值。

Bit 12 **DBG_TIM3_STOP**：内核停止时 TIM3 计数器停止
0：即使内核停止，仍然馈送 TIM3 计数器的时钟
1：内核停止时停止 TIM3 计数器的时钟

Bit 11 **DBG_TIM2_STOP**：内核停止时 TIM2 计数器停止
0：即使内核停止，仍然馈送 TIM2 计数器的时钟
1：内核停止时停止 TIM2 计数器的时钟

Bit 10 **DBG_TIM1_STOP**：内核停止时 TIM1 计数器停止

0：即使内核停止，仍然馈送 TIM1 计数器的时钟

1：内核停止时停止 TIM1 计数器的时钟

Bit 9 保留，必须保持为复位值。

Bit 8 **DBG_IWDG_STOP**：内核停止时停止独立看门狗

0：即使内核停止，独立看门狗计数器时钟仍继续工作

1：内核停止时，独立看门狗计数器时钟停止

Bit 7:3 保留，必须保持为复位值。

Bit 2 **DBG_STANDBY**：调试待机模式

0：（FCLK=关闭，HCLK=关闭）将整个数字部分断电。从软件角度来看，退出待机模式相当于复位（指示 MCU 从待机状态恢复的几个状态位除外）

1：（FCLK=开启，HCLK=开启）在这种情况下，数字部分不断电，FCLK 和 HCLK 由保持激活状态的内部 RC 振荡器提供。此外，MCU 会在待机模式期间产生系统复位，这样退出待机模式相当于复位

Bit 1 **DBG_STOP**：调试停止模式

0：（FCLK=关闭，HCLK=关闭）在停机模式下，时钟控制器禁止所有时钟（包括 HCLK 和 FCLK）。退出停机模式时，时钟配置与复位后的时钟配置相同（CPU 时钟由 16MHz 内部 RC 振荡器（HSI）提供）。因此，软件必须重新编程时钟控制器以使能 PLL 和晶振等。

1：（FCLK=开启，HCLK=开启）在这种情况下，进入停机模式时，FCLK 和 HCLK 由在停机模式下仍保持激活状态的内部 RC 振荡器提供。退出停机模式时，软件必须重新编程时钟控制器以使能 PLL 和晶振等（操作步骤与在 DBG_STOP=0 时相同）

Bit 0 **DBG_SLEEP**：调试睡眠模式

0：（FCLK=开启，HCLK=关闭）在睡眠模式下，FCLK 由原先配置好的系统时钟驱动。在睡眠模式下，时钟控制器配置不复位，保持先前设置的状态。因此，退出睡眠模式时，软件不需要重新配置时钟控制器。

1：（FCLK=开启，HCLK=开启）在这种情况下，进入睡眠模式时，将为 HCLK 和 FCLK 馈送相同的时钟（软件先前配置的系统时钟）。

31 器件电子签名

电子签名存放在闪存存储器模块的系统存储区域。它所包含的芯片识别信息在出厂时编写，用户固件或者外部设备可以读取电子签名，用以自动匹配不同配置的 RX32H710xx 微控制器。

31.1 闪存容量寄存器

基地址：0x1FFF 07E0
只读，它的内容在出厂时编写

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

Bit 15:0 **FLASH_SIZE[15:0]：**闪存存储器容量
以 K 字节为单位指示产品中闪存存储器容量。
例：0x0080 = 128K 字节

31.2 产品唯一身份标识寄存器（96 位）

产品唯一的身份标识非常适合：

- 用来作为序列号
- 用来作为密码，在编写闪存时，将此唯一标识与软件加解密算法结合使用，提高代码在闪存存储器内的安全性。
- 用来激活带安全机制的自举过程

96 位的产品唯一身份标识所提供的参考号码对任意一个 RX32 微控制器，在任何情况下都是唯一的。用户在何种情况下，都不能修改这个身份标识。

这个 96 位的产品唯一身份标识，按照用户不同的用法，可以以字节（8 位）为单位读取，也可以以半字（16 位）或者全字（32 位）读取。

基地址：0x1FFF 07E4
地址偏移：0x00
只读，其值在出厂时编写

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

Bit 31:0 **UID[31:0]：**唯一身份标志 31:0 位

地址偏移：0x04

只读，其值在出厂时编写

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID[63:48]															
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
UID[47:32]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:0 **UID[63:32]**：唯一身份标志 63:32 位

地址偏移：0x08

只读，其值在出厂时编写

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID[95:80]															
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
UID[79:64]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit 31:0 **UID[95:64]**：唯一身份标志 95:64 位

32 版本历史

表 32.1 版本历史

日期	版本	更改内容
2026/06/10	V2.0	新版
2026/07/24	V2.1	1. 删除 TIM1/8 章节图 18.25 信号 oc5/oc6 无法在外部获得的描述 2. 更新 TIM1/8 章节 18.6.2 寄存器 TIMx_CR2 中 MMS2 和 MMS 注释 3. 更新 TIM2/3 章节 19.7.2 寄存器 TIMx_CR2 中 MMS 注释 4. 更新 TIM15 章节 20.6.2 寄存器 TIMx_CR2 中 MMS 注释 5. 优化 TIM1/8 和 TIM15 章节表 18.13/20.11 内容排版格式 6. 更新 ADC 章节 13.7.3 寄存器 ADC_CR2 中 TSVREFE 说明 7. 更新 ADC 章节 13.3.4 ADC 通道连接关于 TSVREFE 注释 8. 更新 TIM2 定时器为 28 位定时器 9. 删除 UART 章节 IDLE 和 IDLE 中断相关描述 10. 新增 Flash 章节配置 RDP[7:0]=0xCC 来设置读保护等级为 Level2