Technology Innovator
Puya

PY32F410 系列参考手册

32 位 ARM® Cortex®-M4 微控制器

Puya Semiconductor (Shanghai) Co., Ltd.

目录

1. 文档约定 24

1.1 寄存器相关缩写词列表 24

1.2 外设可用性 24

2. 系统框图 25

3. 存储器和总线架构 26

3.1 系统架构 26

3.1.1 I_bus 26

3.1.2 D_bus 26

3.1.3 S_bus 26

3.1.4 DMA BUS 27

3.1.5 总线矩阵 27

3.1.6 AHB/APB 总线桥 27

3.2 存储器组织架构 27

3.2.1 简介 27

3.2.2 存储器映射 28

3.3 嵌入式 SRAM 32

3.4 嵌入式 FLASH 32

3.5 位段 32

3.6 启动配置 33

3.6.1 系统存储启动程序 34

3.6.2 物理重映射 34

4. 嵌入式 FLASH 接口(FMC) 35

4.1 FLASH 简介 35

4.2 FLASH 主要特征 35

4.3 FLASH 功能描述 35

4.3.1 闪存结构 35

4.3.2 读访问延迟 36

4.3.3 自适应实时存储器加速器 37

4.3.4 擦除和编程操作 37

4.3.5 闪存解锁 38

4.3.6 闪存擦除操作 38

4.3.7 闪存页擦除 38

4.3.8 闪存扇区擦除 38

4.3.9 闪存 Bank 擦除 39

4.3.10 闪存写操作 39
4.4 产品唯一身份标识码（UID） 40
4.5 FLASH 选项字节 41
4.5.1 Flash 选项字节描述 41
4.5.2 Flash 用户选项字节 0 42
4.5.3 Flash 用户选项字节 1 43
4.5.4 Flash 保护配置 2（BANK0_WRP） 44
4.5.5 Flash 保护配置 3（BANK1_WRP） 44
4.5.6 Flash 保护配置 4（PCROP0SR） 45
4.5.7 Flash 保护配置 5（PCROP0ER） 45
4.5.8 Flash 保护配置 6（PCROP1SR） 46
4.5.9 Flash 保护配置 7（PCROP1ER） 46
4.5.10 Flash 选项字节编程 47
4.6 FLASH 用户数据字节 49
4.7 FLASH 存储区保护 49
4.7.1 闪存读保护(RDP) 49
4.7.2 闪存写保护 (WRP) 50
4.7.3 专有代码读出保护(PCROP) 51
4.7.4 强制从 Main flash 启动 51
4.8 FLASH 中断 51
4.9 FLASH 寄存器 52
4.9.1 Flash 访问控制寄存器(FLASH_ACR) 52
4.9.2 Flash 密钥寄存器(FLASH_KEYR) 53
4.9.3 Flash 选项密钥寄存器 (FLASH_OPTKEYR) 54
4.9.4 Flash 状态寄存器(FLASH_SR) 54
4.9.5 Flash 控制寄存器(FLASH_CR) 55
4.9.6 Flash 选项 1 寄存器(FLASH_OPTR1) 57
4.9.7 Flash 选项 2 寄存器(FLASH_OPTR2) 58
4.9.8 Flash BANK0 WRP 地址寄存器 (FLASH_BANK0_WRPR) 59
4.9.9 Flash BANK1 WRP 地址寄存器 (FLASH_BANK1_WRPR) 60
4.9.10 Flash PCROP0 起始地址寄存器 (FLASH_PCROP0SR) 60
4.9.11 Flash PCROP0 结束地址寄存器 (FLASH_PCROP0ER) 61
4.9.12 Flash PCROP1 起始地址寄存器 (FLASH_PCROP1SR) 61
4.9.13 Flash PCROP1 结束地址寄存器 (FLASH_PCROP1ER) 61
5. 电源控制（PWR） 63
5.1 PWR 简介 63
5.2 PWR 电源 63
5.2.1 电源结构 63

5.2.2 调压器（VR） 64
5.2.3 动态电压调节 64
5.3 PWR 电源检测 64
5.3.1 上电复位(POR)/下电复位(PDR) 64
5.3.2 可编程电压检测器（PVD) 65
5.4 PWR 系统低功耗模式 65
5.4.1 正常运行模式 70
5.4.2 低功耗运行模式 70
5.4.3 低功耗模式进入和退出 70
5.4.4 睡眠模式 71
5.4.5 低功耗睡眠模式 71
5.4.6 停止模式 72
5.4.7 停止模式的调试 73
5.4.8 低功耗模式下的自动唤醒（AWU） 73
5.5 PWR 寄存器 73
5.5.1 PWR 控制寄存器 1(PWR_CR1) 73
5.5.2 PWR 控制寄存器 2(PWR_CR2) 74
5.5.3 PWR 状态寄存器(PWR_SR) 75
6. 复位与时钟控制（RCC） 76
6.1 RCC 简介 76
6.2 RCC 复位 76
6.2.1 电源复位 76
6.2.2 系统复位 76
6.2.3 NRST 引脚（外部复位） 76
6.2.3.1 看门狗复位 76
6.2.3.2 软件复位 77
6.2.3.3 低功耗管理复位 77
6.2.3.4 选项字节加载复位 77
6.2.4 备份域复位 77
6.2.5 复位的统一处理 77
6.3 RCC 时钟 78
6.3.1 时钟源 78
6.3.1.1 HSE 时钟 78
6.3.1.2 HSI 时钟 78
6.3.1.3 PLL 时钟 78
6.3.1.4 LSE 时钟 78
6.3.1.5 LSI 时钟 78
6.3.1.6 HSI10M 时钟 78

6.3.2 时钟结构 79
6.3.3 时钟安全系统（CSS） 79
6.3.4 时钟输出 80
6.3.5 外设时钟使能寄存器 80
6.4 RCC 寄存器 80
6.4.1 RCC 时钟控制寄存器（RCC_CR） 80
6.4.2 RCC 时钟配置寄存器（RCC_CFGR） 82
6.4.3 RCC 时钟中断寄存器（RCC_CIR） 84
6.4.4 RCC AHB 外设复位寄存器（RCC_AHBRSTR） 86
6.4.5 RCC APB1 外设复位寄存器 1（RCC_APB1RSTR1） 86
6.4.6 RCC APB1 外设复位寄存器 2（RCC_APB1RSTR2） 88
6.4.7 RCC APB2 外设复位寄存器（RCC_APB2RSTR） 88
6.4.8 RCC AHB 外设时钟使能寄存器（RCC_AHBENR） 89
6.4.9 RCC APB1 外设时钟使能寄存器 1（RCC_APB1ENR1） 90
6.4.10 RCC APB1 外设时钟使能寄存器 2（RCC_APB1ENR2） 92
6.4.11 RCC APB2 外设时钟使能寄存器（RCC_APB2ENR） 92
6.4.12 RCC 外设独立时钟配置寄存器（RCC_CCIPR） 93
6.4.13 RCC RTC 域控制寄存器（RCC_BDCR） 95
6.4.14 RCC 控制/状态寄存器（RCC_CSR） 96
6.4.15 RCC 时钟配置寄存器 1（RCC_CFGR1） 97
6.4.16 RCC 时钟配置寄存器 2（RCC_CFGR2） 97
6.4.17 RCC 时钟配置寄存器 3（RCC_CFGR3） 98

7. 通用 IO（GPIO） 99

7.1 GPIO 简介 99
7.2 GPIO 主要特征 99
7.3 GPIO 功能描述 99
7.3.1 通用 IO（GPIO） 101
7.3.2 IO 引脚复用器和映射 101
7.3.3 IO 端口控制寄存器 101
7.3.4 IO 端口数据寄存器 101
7.3.5 IO 数据位操作 102
7.3.6 GPIO 锁定机制 102
7.3.7 IO 复用功能输入/输出 102
7.3.8 外部中断线/唤醒线 102
7.3.9 输入配置 102
7.3.10 输出配置 103
7.3.11 复用功能配置 104
7.3.12 模拟配置 104

7.3.13 使用 HSE 或 LSE 振荡器引脚作为 GPIO....105
7.3.14 使用 PD3 作为 GPIO....105
7.3.15 使用 PD11 作为 GPIO....105
7.4 寄存器描述....105
7.4.1 GPIO 端口模式寄存器（GPIOx_MODER）(x= A..D)....106
7.4.2 GPIO 端口输出类型寄存器（GPIOx_OTYPER）(x= A..D)....106
7.4.3 GPIO 端口输出速度寄存器（GPIOx_OSPEEDR）(x= A..D)....106
7.4.4 GPIO 端口上拉/下拉寄存器（GPIOx_PUPDR）(x= A..D)....107
7.4.5 GPIO 端口输入数据寄存器（GPIOx_IDR）(x= A..D)....107
7.4.6 GPIO 端口输出数据寄存器（GPIOx_ODR）(x= A..D)....107
7.4.7 GPIO 端口置位/复位寄存器（GPIOx_BSRR）(x= A..D)....108
7.4.8 GPIO 端口配置锁定寄存器（GPIOx_LCKR）(x= A..D)....108
7.4.9 GPIO 复用功能低寄存器（GPIOx_AFRL）(x= A..D)....109
7.4.10 GPIO 复用功能高寄存器（GPIOx_AFRH）(x= A..D)....109
7.4.11 GPIO 端口位复位寄存器（GPIOx_BRR）(x= A..D)....110
8. 外设互联....111
8.1 简介....111
8.2 互联详情....111
8.2.1 定时器输入触发(ITR)....111
8.2.2 定时器外部触发(ETR)....111
8.2.3 清除定时器 OCxREF 信号....112
8.2.4 定时器输入捕获....112
8.2.5 定时器刹车....113
8.2.6 红外输出波形控制（IRTIM）....113
8.2.7 比较器消隐源....114
8.2.8 系统错误刹车....114
8.2.9 ADC 硬件触发输入....114
9. 系统配置控制器（SYSCFG）....116
9.1 SYSCFG 主要特性....116
9.2 SYSCFG 寄存器....116
9.2.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）....116
9.2.2 SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)....117
9.2.3 SYSCFG 配置寄存器 3（SYSCFG_CFGR3）....117
9.2.4 SYSCFG 配置寄存器 4（SYSCFG_CFGR4）....118
9.2.5 SYSCFG 外部中断配置寄存器 1（SYSCFG_EXTICR1）....118
9.2.6 SYSCFG 外部中断配置寄存器 2（SYSCFG_EXTICR2）....119
9.2.7 SYSCFG 外部中断配置寄存器 3（SYSCFG_EXTICR3）....120

9.2.8 SYSCFG 外部中断配置寄存器 4（SYSCFG_EXTICR4） 121
9.2.9 SYSCFG GPIOA 滤波使能寄存器（SYSCFG_PA_ENS） 122
9.2.10 SYSCFG GPIOB 滤波使能寄存器（PB_ENS） 122
9.2.11 SYSCFG GPIOC 滤波使能寄存器（SYSCFG_PC_ENS） 123
9.2.12 SYSCFG_GPIOD 滤波使能寄存器（SYSCFG_PD_ENS） 123
9.2.13 SYSCFG 电压比较器模拟通道 2 使能寄存器(SYSCFG_COMP_ANA2ENR) 123

10. DMA 控制器（DMA） 125

10.1 DMA 简介 125
10.2 DMA 主要特性 125
10.3 DMA 功能描述 126
10.3.1 DMA 顶层结构图 126
10.3.2 DMA 传输 126
10.3.3 DMA 仲裁器 126
10.3.4 DMA 通道 127
10.3.5 DMA 数据格式 128
10.3.6 DMA 错误处理 130
10.3.7 DMA 外设请求映射 130
10.4 DMA 中断 131
10.5 DMA 寄存器 131
10.5.1 DMA 中断状态寄存器（DMA_ISR） 131
10.5.2 DMA 中断标志清零寄存器（DMA_IFCR） 134
10.5.3 DMA 通道 x 控制寄存器（DMA_CCRx）(x=1~8) 137
10.5.4 DMA 通道 x 数据传输个数寄存器（DMA_CNDTRx）(x=1~8) 139
10.5.5 DMA 通道 x 外设地址寄存器（DMA_CPARx）(x=1~8) 140
10.5.6 DMA 通道 x 存储地址寄存器（DMA_CMARx）(x=1~8) 140
10.5.7 DMA 通道 x 块循环传输配置寄存器（DMA_CCCFGRx）(x=1~8) 141

11. 中断和事件 142

11.1 嵌套向量中断控制器(NVIC) 142
11.1.1 NVIC 主要特性 142
11.1.2 SysTick 校准值寄存器 142
11.1.3 中断和异常向量表 142
11.2 外部扩展中断/事件控制器（EXTI） 144
11.2.1 EXTI 主要特性 144
11.2.2 EXTI 功能说明 145
11.2.3 EXTI 唤醒事件管理 145
11.2.4 EXTI 外部中断/事件线映射 146
11.3 EXTI 寄存器 147

11.3.1 EXTI 中断屏蔽寄存器（EXTI_IMR） 147
11.3.2 EXTI 事件屏蔽寄存器（EXTI_EMR） 147
11.3.3 EXTI 上升沿触发选择寄存器（EXTI_RTSR） 148
11.3.4 EXTI 下降沿触发选择寄存器（EXTI_FTSR） 148
11.3.5 EXTI 软件中断事件寄存器（EXTI_SWIER） 148
11.3.6 EXTI 挂起寄存器（EXTI_PR） 149

12. CRC 计算单元（CRC） 150

12.1 CRC 简介 150
12.2 CRC 主要特性 150
12.3 CRC 功能描述 150
12.3.1 CRC 结构框图 150
12.3.2 CRC 操作 150
12.4 CRC 寄存器 151
12.4.1 CRC 数据寄存器（CRC_DR） 151
12.4.2 CRC 独立数据寄存器（CRC_IDR） 151
12.4.3 CRC 控制寄存器（CRC_CR） 151

13. 模拟/数字转换（ADC） 152

13.1 ADC 简介 152
13.2 ADC 主要特性 152
13.3 ADC 功能描述 153
13.3.1 ADC 框图 153
13.3.2 单端和差分输入通道 154
13.3.3 ADC 校准 156
13.3.4 ADC 开-关控制(ADEN,ADDIS,ADRDY) 157
13.3.5 转换启动(ADSTART，JADSTART) 158
13.3.6 停止正在进行的转换 (ADSTP, JADSTP) 158
13.3.7 ADC 时序(单次/连续模式，硬件/软件触发) 160
13.3.8 写 ADC 控制位的限制 161
13.3.9 ADC 时钟 162
13.3.10 ADC 通道选择(SQRx, JSQR) 162
13.3.11 动态低功耗特性 163
13.3.12 转换模式 166
13.3.13 注入通道管理 168
13.3.14 模拟看门狗 169
13.3.15 数据处理 172
13.3.16 非过载非 DMA 模式 175
13.3.17 DMA 模式 175

13.3.18 过采样器 176
13.3.19 可编程采样时间 181
13.3.20 外部触发转换 181
13.3.21 可配置分辨率 182
13.3.22 温度传感器和内部参考电压 182
13.3.23 电池监视 184
13.3.24 ADC 中断 184
13.4 ADC 寄存器 184
13.4.1 ADC 中断和状态寄存器 (ADC_ISR) 184
13.4.2 ADC 中断使能寄存器 (ADC_IER) 186
13.4.3 ADC 控制寄存器 (ADC_CR) 187
13.4.4 ADC 配置寄存器 (ADC_CFGR) 190
13.4.5 ADC 配置寄存器 2 (ADC_CFGR2) 193
13.4.6 ADC 采样时间寄存器 1(ADC_SMPR1) 195
13.4.7 ADC 采样时间寄存器 2(ADC_SMPR2) 195
13.4.8 DC 采样时间寄存器 2(ADC_SMPR3) 196
13.4.9 ADC 看门狗阈值寄存器 1 (ADC_TR1) 196
13.4.10 ADC 看门狗阈值寄存器 2 (ADC_TR2) 197
13.4.11 ADC 看门狗阈值寄存器 3 (ADC_TR3) 197
13.4.12 ADC 规则序列寄存器 1(ADC_SQR1) 198
13.4.13 ADC 规则序列寄存器 2(ADC_SQR2) 198
13.4.14 ADC 规则序列寄存器 3(ADC_SQR3) 199
13.4.15 ADC 规则序列寄存器 4(ADC_SQR4) 199
13.4.16 ADC 规则数据寄存器 (ADC_DR) 200
13.4.17 ADC 注入序列寄存器(ADC_JSQR) 200
13.4.18 ADC offset y 寄存器(ADC_OFRy) 201
13.4.19 ADC 注入数据寄存器 (ADC_JDRy) 202
13.4.20 ADC 模拟看门狗 2 配置寄存器 (ADC_AWD2CR) 202
13.4.21 ADC 模拟看门狗 3 配置寄存器 (ADC_AWD3CR) 203
13.4.22 ADC 校准因子寄存器 (ADC_CALFACT) 203
13.4.23 ADC 增益补偿寄存器 (ADC_GCOMP) 205
13.4.24 ADC 公共寄存器(ADC_CCR) 205
14. 电压基准缓冲器(V_{REFBUF}) 208
14.1 V_{REFBUF} 简介 208
14.2 V_{REFBUF} 功能描述 208
15. 比较器(COMP) 209
15.1 COMP 简介 209

15.2 COMP 主要特性 209
15.3 COMP 功能描述 210
15.3.1 COMP 模块框图 210
15.3.2 COMP 的输入和输出 210
15.3.3 COMP 的时钟和复位 211
15.3.4 COMP 的滤波功能 211
15.3.5 COMP 的锁定机制 212
15.3.6 COMP 的输出消隐 212
15.3.7 COMP 的窗口模式 213
15.3.8 COMP 的迟滞功能 213
15.3.9 COMP 的功耗模式 214
15.3.10 COMP 的低功耗模式 214
15.3.11 COMP 选择 V_{CC}/V_{REFBUF} 配置 214
15.4 COMP 的中断 214
15.5 COMP 寄存器 214
15.5.1 COMP1 控制和状态寄存器(COMP1_CSR) 214
15.5.2 COMP1 滤波寄存器 (COMP1_FR) 216
15.5.3 COMP2 控制和状态寄存器(COMP2_CSR) 217
15.5.4 COMP2 滤波寄存器(COMP2_FR) 218
16. 运算放大器(OPA) 219
16.1 OPA 简介 219
16.2 OPA 主要特性 219
16.3 OPA 功能描述 219
16.3.1 模块框图 219
16.3.2 OPA 输出重定向到内部 ADC 通道 220
16.3.3 OPA 复位和时钟 220
16.3.4 初始配置 220
16.3.5 信号连接 220
16.3.6 OPA 模式 220
16.3.7 OPA 低功耗模式 221
16.4 OPA 中断 221
16.5 OPA 寄存器 221
16.5.1 OPA1 控制/状态寄存器（OPA1_CSR） 221
16.5.2 OPA2 控制/状态寄存器（OPA2_CSR） 222
16.5.3 OPA2 滤波寄存器（OPA2_FR） 223
16.5.4 OPA2 中断寄存器（OPA2_INTR） 223
17. 高级定时器（TIM1） 224

17.1 TIM1 简介 224
17.2 TIM1 主要特征 224
17.2.1 TIM1 模块框图 225
17.3 TIM1 功能描述 225
17.3.1 时基单元 225
17.3.2 计数器模式 227
17.3.3 重复计数器 234
17.3.4 外部触发输入 235
17.3.5 时钟选择 235
17.3.6 捕获/比较通道 238
17.3.7 输入捕获模式 240
17.3.8 PWM 输入模式 240
17.3.9 强置输出模式 241
17.3.10 输出比较模式 241
17.3.11 PWM 模式 242
17.3.12 互补输出和死区插入 246
17.3.13 使用刹车功能 247
17.3.14 在外部事件时清除 OCxREF 信号 250
17.3.15 产生六步 PWM 输出 250
17.3.16 单脉冲模式 251
17.3.17 可重触发的单脉冲模式 253
17.3.18 编码器接口模式 253
17.3.19 定时器输入异或功能 255
17.3.20 与霍尔传感器的接口 255
17.3.21 定时器和外部触发的同步 257
17.3.22 DMA 突发模式 259
17.3.23 TIM1 DMA 请求 260
17.3.24 定时器同步 260
17.3.25 调试模式 265
17.4 TIM1 中断 265
17.5 TIM1 寄存器描述 265
17.5.1 TIMx 控制寄存器 1（TIMx_CR1)(x=1) 265
17.5.2 TIMx 控制寄存器 2（TIMx_CR2)(x=1) 267
17.5.3 TIMx 从模式控制寄存器（TIMx_SMCR)(x=1) 269
17.5.4 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=1) 272
17.5.5 TIMx 状态寄存器（TIMx_SR)(x=1) 273
17.5.6 TIMx 事件产生寄存器（TIMx_EGR)(x=1) 275
17.5.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1)(x=1) 277
17.5.8 TIMx 捕获/比较模式控制寄存器 2（TIMx_CCMR2)(x=1) 281

17.5.9 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=1)....282
17.5.10 TIMx 计数器（TIMx_CNT)(x=1)....285
17.5.11 TIMx 预分频器（TIMx_PSC)(x=1)....286
17.5.12 TIMx 自动重装载寄存器（TIMx_ARR)(x=1)....286
17.5.13 TIMx 重复计数寄存器（TIMx_RCR)(x=1)....286
17.5.14 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=1)....287
17.5.15 TIMx 捕获/比较寄存器 2（TIMx_CCR2)(x=1)....288
17.5.16 TIMx 捕获/比较寄存器 3（TIMx_CCR3)(x=1)....288
17.5.17 TIMx 捕获/比较寄存器 4（TIMx_CCR4)(x=1)....289
17.5.18 TIMx 刹车和死区寄存器（TIMx_BDTR)(x=1)....289
17.5.19 TIMx 捕获/比较寄存器 5（TIMx_CCR5)(x=1)....291
17.5.20 TIMx 捕获/比较寄存器 6（TIMx_CCR6)(x=1)....292
17.5.21 TIMx 捕获/比较模式控制寄存器 3（TIMx_CCMR3)(x=1)....292
17.5.22 TIMx 输入选择寄存器（TIMx_TISEL)(x=1)....294
17.5.23 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=1)....295
17.5.24 TIMx 备用功能选项寄存器 2（TIMx_AF2)(x=1)....297
17.5.25 TIMx DMA 控制寄存器（TIMx_DCR)(x=1)....297
17.5.26 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=1)....298

18. 通用定时器（TIM2/TIM3/TIM4）....299

18.1 TIM2/TIM3/TIM4 简介....299
18.2 TIM2/TIM3/TIM4 主要特征....299
18.2.1 TIM2/TIM3/TIM4 模块框图....300
18.3 TIM2/TIM3/TIM4 功能描述....300
18.3.1 时基单元....300
18.3.2 计数器模式....301
18.3.3 外部触发输入....308
18.3.4 时钟选择....309
18.3.5 捕获/比较通道....311
18.3.6 输入捕获模式....312
18.3.7 PWM 输入模式....313
18.3.8 强置输出模式....314
18.3.9 输出比较模式....314
18.3.10 PWM 模式....315
18.3.11 在外部事件时清除 OCxREF 信号....317
18.3.12 单脉冲模式....318
18.3.13 编码器接口模式....320
18.3.14 定时器输入异或功能....321
18.3.15 定时器和外部触发的同步....322

18.3.16 定时器同步 323
18.3.17 DMA 突发模式 323
18.3.18 TIM2/TIM3/TIM4 DMA 请求 324
18.3.19 调试模式 324
18.3.20 TIM2/TIM3/TIM4 低功耗模式 325
18.4 TIM2/TIM3/TIM4 中断 325
18.5 TIM2/TIM3/TIM4 寄存器描述 325
18.5.1 TIMx 控制寄存器 1（TIMx_CR1）（x=2/3/4） 325
18.5.2 TIMx 控制寄存器 2（TIMx_CR2）（x=2/3/4） 326
18.5.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x=2/3/4） 328
18.5.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x=2/3/4） 330
18.5.5 TIMx 状态寄存器（TIMx_SR）（x=2/3/4） 331
18.5.6 TIMx 事件产生寄存器（TIMx_EGR）（x=2/3/4） 333
18.5.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1）（x=2/3/4） 334
18.5.8 TIMx 捕获/比较模式控制寄存器 2（TIMx_CCMR2）（x=2/3/4） 337
18.5.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=2/3/4） 338
18.5.10 TIMx 计数器（TIMx_CNT）（x=2） 340
18.5.11 TIMx 计数器（TIMx_CNT）（x=3/4） 340
18.5.12 TIMx 预分频器（TIMx_PSC）（x=2/3/4） 340
18.5.13 TIMx 自动重装载寄存器（TIMx_ARR）（x=2） 341
18.5.14 TIMx 自动重装载寄存器（TIMx_ARR）（x=3/4） 341
18.5.15 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x=2） 342
18.5.16 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x=3/4） 342
18.5.17 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x=2） 342
18.5.18 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x=3/4） 343
18.5.19 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x=2） 343
18.5.20 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x=3/4） 344
18.5.21 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x=2） 344
18.5.22 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x=3/4） 345
18.5.23 TIMx 输入选择寄存器（TIMx_TISEL）（x=2/3/4） 345
18.5.24 TIMx 备用功能选项寄存器 1（TIMx_AF1）（x=2/3/4） 346
18.5.25 TIMx 备用功能选项寄存器 2（TIMx_AF2）（x=2/3/4） 346
18.5.26 TIMx DMA 控制寄存器（TIMx_DCR）（x=2/3/4） 347
18.5.27 TIMx 连续模式的 DMA 地址（TIMx_DMAR）（x=2/3/4） 348
19. 通用定时器（TIM15/TIM16/TIM17） 349
19.1 TIM15/TIM16/TIM17 简介 349
19.2 TIM15 主要特征 349
19.2.1 TIM15 模块框图 350

19.3 TIM16/TIM17 主要特征 350
19.3.1 TIM16/TIM17 模块框图 351
19.4 TIM15/TIM16/TIM17 功能描述 351
19.4.1 时基单元 351
19.4.2 计数器模式 352
19.4.3 重复计数器 355
19.4.4 时钟选择 356
19.4.5 捕获/比较通道 358
19.4.6 输入捕获模式 359
19.4.7 PWM 输入模式（仅 TIM15） 360
19.4.8 强置输出模式 361
19.4.9 输出比较模式 361
19.4.10 PWM 模式 362
19.4.11 互补输出和死区插入 363
19.4.12 使用刹车功能 364
19.4.13 单脉冲模式（仅 TIM15） 366
19.4.14 定时器和外部触发的同步 367
19.4.15 定时器同步（仅 TIM15） 369
19.4.16 DMA 突发模式 370
19.4.17 TIM15/TIM16/TIM17 DMA 请求 370
19.4.18 调试模式 371
19.5 TIM15/TIM16/TIM17 中断 371
19.6 TIM15 寄存器描述 371
19.6.1 TIMx 控制寄存器 1（TIMx_CR1)(x=15) 371
19.6.2 TIMx 控制寄存器 2（TIMx_CR2)(x=15) 372
19.6.3 TIMx 从模式控制寄存器（TIMx_SMCR)(x=15) 373
19.6.4 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=15) 375
19.6.5 TIMx 状态寄存器（TIMx_SR)(x=15) 376
19.6.6 TIMx 事件产生寄存器（TIMx_EGR)(x=15) 378
19.6.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1)(x=15) 379
19.6.8 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=15) 382
19.6.9 TIMx 计数器（TIMx_CNT)(x=15) 385
19.6.10 TIMx 预分频器（TIMx_PSC)(x=15) 385
19.6.11 TIMx 自动重装载寄存器（TIMx_ARR)(x=15) 385
19.6.12 TIMx 重复计数寄存器（TIMx_RCR)(x=15) 386
19.6.13 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=15) 386
19.6.14 TIMx 捕获/比较寄存器 2（TIMx_CCR2)(x=15) 386
19.6.15 TIMx 刹车和死区寄存器（TIMx_BDTR)(x=15) 387
19.6.16 TIMx 输入选择寄存器（TIMx_TISEL)(x=15) 389

19.6.17 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=15)....389
19.6.18 TIMx DMA 控制寄存器（TIMx_DCR)(x=15)....391
19.6.19 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=15)....392
19.7 TIM16/TIM17 寄存器描述....392
19.7.1 TIMx 控制寄存器 1（TIMx_CR1)(x=16/17)....392
19.7.2 TIMx 控制寄存器 2（TIMx_CR2)(x=16/17)....393
19.7.3 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=16/17)....394
19.7.4 TIMx 状态寄存器（TIMx_SR)(x=16/17)....395
19.7.5 TIMx 事件产生寄存器（TIMx_EGR)(x=16/17)....396
19.7.6 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1)(x=16/17)....397
19.7.7 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=16/17)....399
19.7.8 TIMx 计数器（TIMx_CNT)(x=16/17)....401
19.7.9 TIMx 预分频器（TIMx_PSC)(x=16/17)....402
19.7.10 TIMx 自动重装载寄存器（TIMx_ARR)(x=16/17)....402
19.7.11 TIMx 重复计数寄存器（TIMx_RCR)(x=16/17)....402
19.7.12 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=16/17)....403
19.7.13 TIMx 刹车和死区寄存器（TIMx_BDTR)(x=16/17)....403
19.7.14 TIMx 输入选择寄存器（TIMx_TISEL)(x=16/17)....405
19.7.15 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=16/17)....405
19.7.16 TIMx DMA 控制寄存器（TIMx_DCR)(x=16/17)....407
19.7.17 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=16/17)....408

20. 基本定时器（TIM6/TIM7）....409

20.1 TIM6/TIM7 简介....409
20.2 TIM6 和 TIM7 主要特征....409
20.3 TIM6/TIM7 功能描述....409
20.3.1 功能框图....409
20.3.2 时基单元....409
20.3.3 计数模式（递增）....410
20.4 TIM6/TIM7 调试模式....413
20.5 TIM6/TIM7 寄存器....413
20.5.1 TIM6/TIM7 控制寄存器 1（TIMx_CR1）....413
20.5.2 TIM6/TIM7 控制寄存器 2（TIMx_CR2）....414
20.5.3 TIM6/TIM7 DMA/中断使能寄存器（TIMx_DIER）....415
20.5.4 TIM6/TIM7 状态寄存器（TIMx_SR）....415
20.5.5 TIM6/TIM7 事件产生寄存器（TIMx_EGR）....415
20.5.6 TIM6/TIM7 计数寄存器（TIMx_CNT）....416
20.5.7 TIM6/TIM7 预分频寄存器（TIMx_PSC）....416
20.5.8 TIM6/TIM7 自动重载寄存器（TIMx_ARR）....416

21. 专用脉冲宽度调制(PWM) 418

21.1 PWM 简介 418
21.2 PWM1/PWM2/PWM3 主要特征 418
21.3 PWM4 主要特征 419
21.4 PWM 功能描述 419
21.4.1 PWM1/PWM2/PWM3 模块框图 419
21.4.2 PWM4 模块框图 420
21.4.3 时基单元 420
21.4.4 计数器模式 421
21.4.5 时钟选择（仅 PWM1/PWM2/PWM3 支持） 425
21.4.6 输出比较模式 427
21.4.7 PWM 模式 428
21.4.8 互补输出和死区插入（仅 PWM1/PWM2/PWM3 支持） 431
21.4.9 使用刹车功能（仅 PWM1/PWM2/PWM3 支持） 431
21.5 PWM1/PWM2/PWM3 寄存器描述 432
21.5.1 PWM 控制寄存器 1（PWM_CR1） 432
21.5.2 PWM 控制寄存器 2（PWM_CR2） 433
21.5.3 PWM 从模式控制寄存器（PWM_SMCR） 435
21.5.4 PWM DMA/中断使能寄存器（PWM_DIER) 436
21.5.5 PWM 状态寄存器（PWM_SR) 437
21.5.6 PWM 事件产生寄存器 1(PWM_EGR) 439
21.5.7 PWM 输出比较模式寄存器 1(PWM_CMR) 439
21.5.8 PWM 输出比较使能寄存器（PWM_CER) 440
21.5.9 PWM 计数寄存器(PWM_CNT) 441
21.5.10 PWM 预分频器（PWM_PSC） 442
21.5.11 PWM 自动重载寄存器（PWM_ARR) 442
21.5.12 PWM 比较寄存器 1(PWM_CCR1) 442
21.5.13 PWM 比较寄存器 2(PWM_CCR2) 443
21.5.14 PWM 比较寄存器 3(PWM_CCR3) 443
21.5.15 PWM 比较寄存器 4(PWM_CCR4) 443
21.5.16 PWM 刹车和死区寄存器(PWM_BDTR) 444
21.5.17 PWM DMA 控制寄存器（PWM_DCR） 445
21.5.18 PWM 连续模式的 DMA 地址（PWM_DMAR） 446
21.6 PWM4 寄存器描述 446
21.6.1 PWM 控制寄存器 1（PWM_CR1） 446
21.6.2 PWM DMA/中断使能寄存器（PWM_DIER) 448
21.6.3 PWM 状态寄存器（PWM_SR) 449
21.6.4 PWM 事件产生寄存器 1(PWM_EGR) 450

21.6.5 PWM 输出比较模式寄存器 1(PWM_CMR) 450
21.6.6 PWM 输出比较使能寄存器 (PWM_CER) 451
21.6.7 PWM 计数寄存器(PWM_CNT) 452
21.6.8 PWM 预分频器（PWM_PSC） 452
21.6.9 PWM 自动重载寄存器 (PWM_ARR) 452
21.6.10 PWM 比较寄存器 1(PWM_CCR1) 453
21.6.11 PWM 比较寄存器 2(PWM_CCR2) 453
21.6.12 PWM 比较寄存器 3(PWM_CCR3) 453
21.6.13 PWM 比较寄存器 4(PWM_CCR4) 454
21.6.14 PWM DMA 控制寄存器（PWM_DCR） 454
21.6.15 PWM 连续模式的 DMA 地址（PWM_DMAR） 455

22. 低功耗定时器(LPTIM) 456

22.1 LPTIM 简介 456
22.2 LPTIM 主要特性 456
22.3 LPTIM 功能描述 456
22.3.1 模块框图 456
22.3.2 复位和时钟 457
22.3.3 毛刺滤波器 457
22.3.4 预分频 457
22.3.5 触发选择 457
22.3.6 操作模式 458
22.3.7 波形产生 459
22.3.8 寄存器更新 460
22.3.9 计数器模式 460
22.3.10 定时器使能 460
22.3.11 定时器复位 461
22.3.12 编码器模式 461
22.3.13 调试模式 462
22.4 LPTIM 低功耗模式 462
22.5 LPTIM 中断 462
22.6 LPTIM 寄存器 463
22.6.1 LPTIM 中断和状态寄存器 (LPTIM_ISR) 463
22.6.2 LPTIM 中断清零寄存器 (LPTIM_ICR) 464
22.6.3 LPTIM 中断使能寄存器 (LPTIM_IER) 464
22.6.4 LPTIM 配置寄存器 (LPTIM_CFGR) 465
22.6.5 LPTIM 控制寄存器 (LPTIM_CR) 467
22.6.6 LPTIM 比较寄存器 (LPTIM_CMP) 468
22.6.7 LPTIM 自动重载寄存器 (LPTIM_ARR) 469

22.6.8 LPTIM 计数器寄存器（LPTIM_CNT）.....469
22.6.9 LPTIM 选择寄存器（LPTIM_OR）.....470
23. 红外接口(IRTIM) 471
24. 实时时钟（RTC） 472
24.1 RTC 简介.....472
24.2 RTC 主要特性 472
24.3 RTC 功能描述 473
24.3.1 概述 473
24.3.2 寄存器复位.....473
24.3.3 读 RTC 寄存器.....473
24.3.4 配置 RTC 寄存器 474
24.3.5 RTC 标志设置.....475
24.3.6 RTC 时序 475
24.3.7 侵入检测 476
24.3.8 RTC 校准 476
24.3.9 备份域寄存器擦除.....479
24.4 RTC 寄存器 479
24.4.1 RTC 控制寄存器高位(RTC_CRH) 479
24.4.2 RTC 控制寄存器低位(RTC_CRL).....480
24.4.3 RTC 预分频装载寄存器高位(RTC_PRLH).....481
24.4.4 RTC 预分频装载寄存器低位(RTC_PRLL) 482
24.4.5 RTC 预分频余数寄存器高位(RTC_DIVH).....482
24.4.6 RTC 预分频余数寄存器低位(RTC_DIVL) 483
24.4.7 RTC 计数寄存器高位(RTC_CNTH) 483
24.4.8 RTC 计数寄存器低位(RTC_CNTL).....483
24.4.9 RTC 闹钟寄存器高位(RTC_ALRH) 484
24.4.10 RTC 闹钟寄存器低位(RTC_ALRL) 484
24.4.11 RTC 入侵配置寄存器（RTC_TMPCFGR） 484
24.4.12 RTC 入侵控制/状态寄存器（RTC_TMPCSR） 485
24.4.13 RTC 时钟校准寄存器(RTC_CALIBR) 486
24.4.14 RTC 备份寄存器(RTC_BKPDR) 486
25. 独立看门狗（IWDG） 488
25.1 IWDG 简介 488
25.2 IWDG 主要特性 488
25.3 IWDG 功能说明 488
25.3.1 模块框图 488
25.3.2 硬件看门狗.....489

25.3.3 寄存器保护 489
25.3.4 调试模式 489
25.3.5 低功耗冻结 489
25.4 IWDG 寄存器 489
25.4.1 IWDG 键值寄存器（IWDG_KR） 489
25.4.2 IWDG 预分频寄存器（IWDG_PR） 490
25.4.3 IWDG 重装载寄存器（IWDG_RLR） 490
25.4.4 IWDG 状态寄存器（IWDG_SR） 491
26. 窗口看门狗（WWDG） 492
26.1 WWDG 简介 492
26.2 WWDG 主要特性 492
26.3 WWDG 功能描述 492
26.3.1 如何编写看门狗超时程序 493
26.3.2 调试模式 493
26.4 WWDG 寄存器描述 494
26.4.1 WWDG 控制寄存器（WWDG_CR） 494
26.4.2 WWDG 配置寄存器（WWDG_CFR） 494
26.4.3 WWDG 状态寄存器（WWDG_SR） 495
27. 串行外设接口/集成电路内置音频总线(SPI/I²S) 496
27.1 SPI/I²S 简介 496
27.2 SPI/I²S 主要特性 496
27.2.1 SPI 主要特性 496
27.2.2 I²S 主要特性 496
27.3 SPI 功能描述 497
27.3.1 SPI 框图 497
27.3.2 一个主机和一个从机的通信 498
27.3.3 标准多从机通信 500
27.3.4 多主机通信 500
27.3.5 从机选择接口（NSS） 501
27.3.6 通信格式 502
27.3.7 配置 SPI 503
27.3.8 使能 SPI 步骤 504
27.3.9 数据发送和接收 504
27.3.10 状态标志 508
27.3.11 错误标志 508
27.3.12 TI 模式 509
27.3.13 SPI CRC 510

27.3.14 SPI 中断 511
27.4 I²S 功能描述 512
27.4.1 I²S 框图 512
27.4.2 支持音频协议 513
27.4.3 时钟产生 521
27.4.4 I²S 传输 523
27.4.5 I²S 标志 525
27.4.6 I²S DMA 526
27.4.7 I²S 中断 526
27.5 SPI/I²S 寄存器 527
27.5.1 SPI 控制寄存器 1 (SPI_CR1) 527
27.5.2 SPI 控制寄存器 2 (SPI_CR2) 529
27.5.3 SPI 状态寄存器 (SPI_SR) 530
27.5.4 SPI 数据寄存器 (SPI_DR) 531
27.5.5 SPI CRC 多项式寄存器 (SPI_CRCPR) 532
27.5.6 SPI 接收 CRC 寄存器 (SPI_RXCRC) 532
27.5.7 SPI 发送 CRC 寄存器 (SPI_TXCRC) 533
27.5.8 SPI_I2S 配置寄存器 (SPI_I2SCFGR) 533
27.5.9 SPI_I2S 预分频器寄存器 (SPI_I2SPR) 535
28. 内部集成电路接口(I²C) 536
28.1 I²C 简介 536
28.2 I²C 功能描述 536
28.2.1 简介 536
28.2.2 时钟 537
28.2.3 包错误校验（PEC） 537
28.2.4 I²C 的从模式 537
28.2.5 I²C 的主模式 539
28.2.6 错误标志 545
28.2.7 DMA 功能 546
28.2.8 支持从 Stop 模式唤醒 546
28.2.9 系统管理总线 SMbus 547
28.3 I²C 中断 549
28.4 I²C 寄存器 550
28.4.1 I²C 控制寄存器 1 (I2C_CR1) 550
28.4.2 I2C 控制寄存器 2 (I2C_CR2) 553
28.4.3 I2C 自身地址寄存器 1 (I2C_OAR1) 554
28.4.4 I2C 自身地址寄存器 2（I2C_OAR2） 554
28.4.5 I2C 数据寄存器(I2C_DR) 555

28.4.6 I²C 状态寄存器 1 (I2C_SR1)....556
28.4.7 I²C 状态寄存器 2 (I2C_SR2)....559
28.4.8 I²C 时钟控制寄存器(I2C_CCR)....560
28.4.9 I²C 上升时间寄存器 (I2C_TRISE)....561
28.4.10 I²C 超时寄存器(I2C_TIMEOUTR)....562
29. 通用同步收发器(USART)....564
29.1 USART 简介....564
29.2 USART 主要特征....564
29.3 USART 功能描述....565
29.3.1 USART 功能框图....565
29.3.2 USART 输入/输出引脚说明....565
29.3.3 USART 时钟....566
29.3.4 USART 字符说明....566
29.3.5 USART FIFO 和阈值....567
29.3.6 USART 数据发送....568
29.3.7 USART 数据接收....571
29.3.8 USART 分数波特率的产生....575
29.3.9 USART 接收器容忍时钟的变化....576
29.3.10 USART 自动波特率检测....577
29.3.11 USART 多处理器通信....577
29.3.12 USART Modbus 通信....579
29.3.13 USART 校验控制....579
29.3.14 USART LIN(局域互联网)模式....579
29.3.15 USART 同步模式....581
29.3.16 USART 单线半双工通信....582
29.3.17 USART 接收超时....582
29.3.18 USART 智能卡....582
29.3.19 USART IrDA SIR ENDEC 功能模块....584
29.3.20 USART 利用 DMA 连续通信....586
29.3.21 USART 硬件流控制....588
29.4 中断请求....589
29.5 USART 寄存器....589
29.5.1 USART 状态寄存器（USART_SR）....589
29.5.2 USART 发送数据寄存器（USART_DR）....593
29.5.3 USART 波特率寄存器（USART_BRR）....593
29.5.4 USART 控制寄存器 1（USART_CR1）....593
29.5.5 USART 控制寄存器 2（USART_CR2）....596
29.5.6 USART 控制寄存器 3（USART_CR3）....598

29.5.7 USART 保护时间和预分频（USART_GTPR） 600
29.5.8 USART 接收超时寄存器（USART_RTOR） 601

30. 通用异步收发器(UART) 602

30.1 UART 简介 602
30.2 UART 主要特征 602
30.3 UART 功能描述 603
30.3.1 UART 框图 603
30.3.2 UART（RS232）串行协议 603
30.3.3 UART 9 位数据传输 605
30.3.4 UART 接收容忍度 608
30.3.5 UART DMA 通信 609
30.4 UART 中断 609
30.5 UART 寄存器 610
30.5.1 UART 数据寄存器（UART_DR） 610
30.5.2 UART 波特率寄存器（UART_BRR） 611
30.5.3 UART 状态寄存器（UART_SR） 611
30.5.4 UART 控制寄存器 1 (UART_CR1) 613
30.5.5 UART 控制寄存器 2 (UART_CR2) 615
30.5.6 UART 控制寄存器 3 (UART_CR3) 616
30.5.7 UART 接收地址寄存器（UART_RAR） 617
30.5.8 UART 发送地址寄存器（UART_TAR） 618
30.5.9 UART 波特率小数寄存器（UART_BRRF） 618

31. 低功耗通用异步收发器（LPUART） 619

31.1 LPUART 主要特性 619
31.2 LPUART 功能描述 620
31.2.1 LPUART 框图 620
31.2.2 LPUART 信号 620
31.2.3 LPUART 字符描述 620
31.2.4 LPUART 发送 623
31.2.5 LPUART 接收 623
31.2.6 LPUART 时钟源选择 624
31.2.7 LPUART 波特率产生 625
31.2.8 LPUART 接收容忍度 626
31.2.9 LPUART 多处理器通信 626
31.2.10 LPUART 奇偶校验 627
31.2.11 LPUART 单线半双工通信 628
31.2.12 DMA 连续通信 628

31.2.13 RS232 硬件流控制和 RS485 驱动器使能 630
31.2.14 LPUART 低功耗模式 632
31.3 LPUART 中断 633
31.4 LPUART 寄存器 633
31.4.1 LPUART 控制寄存器 1（LPUART_CR1） 633
31.4.2 LPUART 控制寄存器 2（LPUART_CR2） 636
31.4.3 LPUART 控制寄存器 3（LPUART_CR3） 637
31.4.4 LPUART 波特率寄存器（LPUART_BRR） 639
31.4.5 LPUART 请求寄存器（LPUART_RQR） 639
31.4.6 LPUART 中断和状态寄存器（LPUART_ISR） 640
31.4.7 LPUART 中断标志清零寄存器（LPUART_ICR） 642
31.4.8 LPUART 接收数据寄存器（LPUART_RDR） 643
31.4.9 LPUART 发送数据寄存器（LPUART_TDR） 643
31.4.10 LPUART 预分频器寄存器（LPUART_PRESC） 643
32. MCU 调试接口 645
32.1 DBGMCU 简介 645
32.2 DBGMCU 主要特性 645
32.3 DBGMCU 功能描述 645
32.3.1 支持调试低功耗模式 645
32.4 DBGMCU 寄存器描述 645
32.4.1 ID 编码（DBGMCU_IDCODE） 645
32.4.2 调试配置寄存器（DBGMCU_CR1） 645
32.4.3 调试配置寄存器（DBGMCU_CR2） 647
33. 版本历史 649

1.文档约定

1.1 寄存器相关缩写词列表

缩写	描述
读/写(RW)	软件可以读写此位
只读(R)	软件只能读取此位
只写(W)	软件只能写入此位，读此位将返回复位值
读取/写入 0 清零(RC_W0)	软件可以读取此位，也可以通过写 0 清除此位，写 1 对此位无影响
读取/写入 1 清零(RC_W1)	软件可以读取此位，也可以通过写 1 清除此位，写 0 对此位无影响
读取/写入清零(RC_W)	软件可以读取此位，也可以通过写入寄存器来清除该位，写入该位的值并不重要
读取/读取清零(RC_R)	软件可以读取此位。读取此位会将其自动清 0，写入此位对其值无影响
读取/读取置位(RS_R)	软件可以读取此位。读取此位会自动将其置 1，写入此位对其值无影响
读取/置位(RS)	软件可以读取此位，也可以将其置 1，写 0 对此位无影响
切换(T)	软件可以通过写入 1 来切换此位，写入 0 无影响
保留(Res.)	保留位，必须保持在复位值

1.2 外设可用性

有关各型号产品的外设可用性以及数量信息，请参见相关器件数据手册。

2. 系统框图

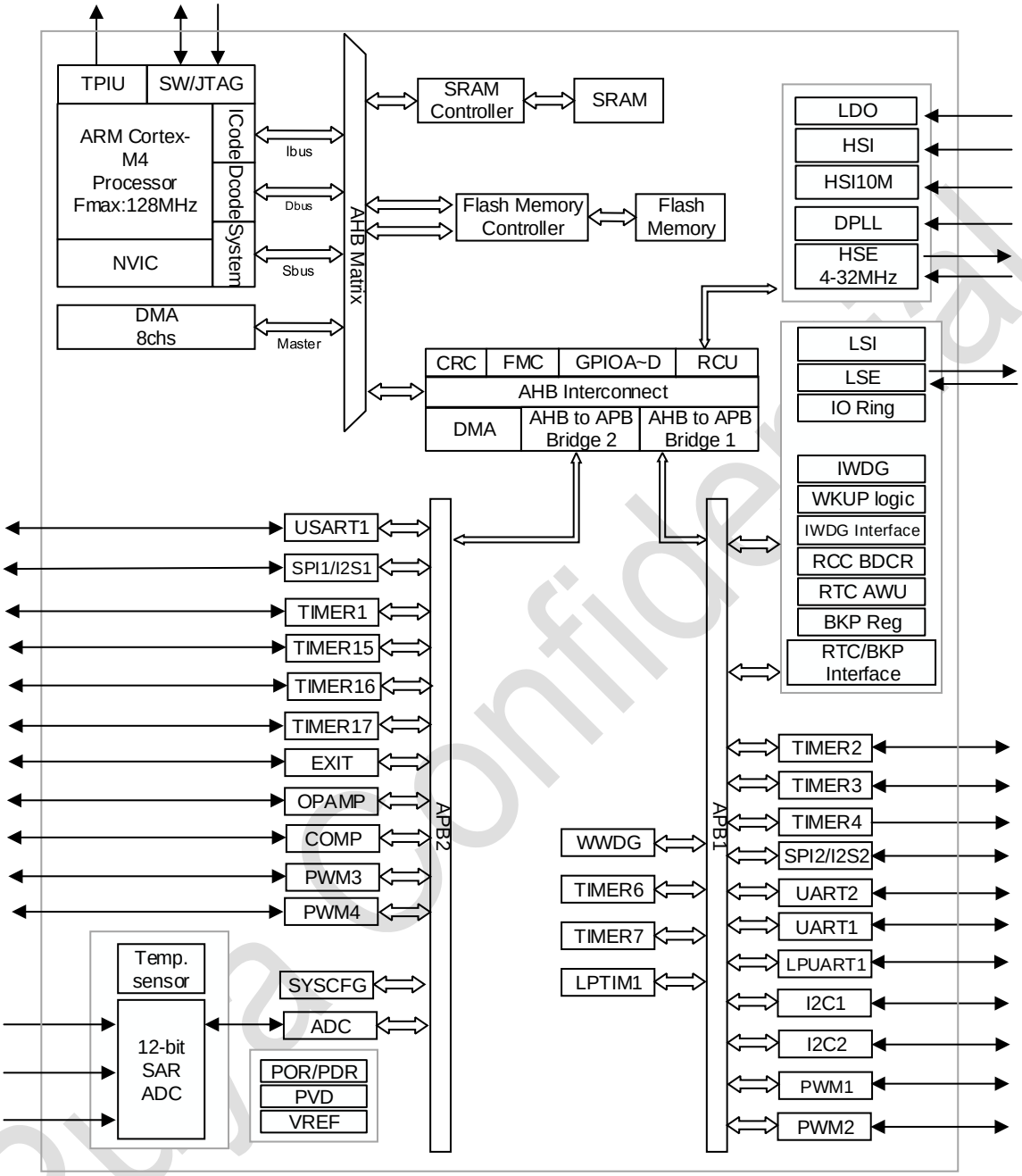


图 2-1 系统框图

3. 存储器和总线架构

3.1 系统架构

系统采用 32 位多层 AHB 总线矩阵，可以保证多主机和多从机之间的并行通信：

- 四条主控总线：
  - Cortex®-M4 内核 I -Code 总线、D-Code 总线和系统总线
  - DMA 总线
- 四条被控总线：
  - 内部 Flash I-Code 总线
  - 内部 Flash D-Code 总线
  - 内部 SRAM 总线
  - AHB 外设总线（包括 AHB 到 APB 的总线桥和 APB 外设）

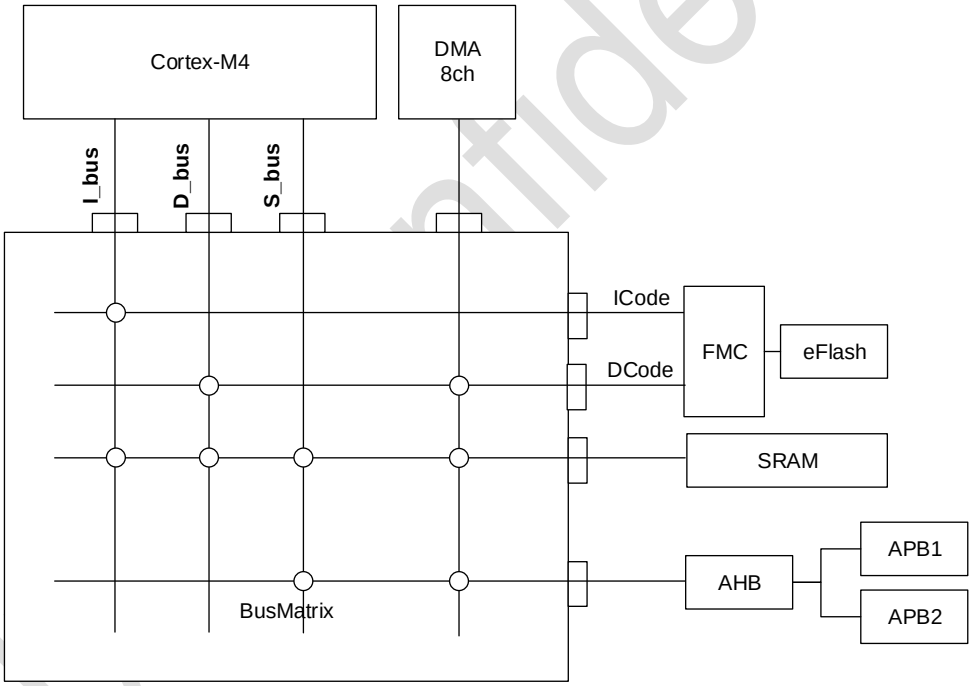


图 3-1 总线架构

3.1.1 I_bus

此总线用于将 Cortex®-M4 内核的指令总线连接到总线矩阵。内核通过此总线获取指令。此总线访问的对象是包含代码的存储器（内部 Flash/SRAM）。

3.1.2 D_bus

此总线用于将 Cortex®-M4 数据总线连接到总线矩阵。内核通过此总线进行立即数加载和调试访问。此总线访问的对象是包含代码或数据的存储器（内部 Flash/SRAM）。

3.1.3 S_bus

此总线用于将 Cortex®-M4 内核的系统总线连接到总线矩阵。此总线用于访问位于外设或 SRAM 中的数据。此总线访问的对象是 16 KB 的内部 SRAM、包括 AHB 外设在内的 APB1 外设、APB2 外设。

3.1.4 DMA BUS

此总线用于将 DMA 存储器总线主接口连接到总线矩阵。 DMA 通过此总线访问的对象是 16 KB 的内部 SRAM、包括 AHB 外设在内的 APB1 外设、 APB2 外设。

3.1.5 总线矩阵

总线矩阵用于主控总线之间的访问仲裁管理。仲裁采用循环调度算法。系统包含 4 个主设备 Master，分别为 ARM® Cortex®-M4 的 I_BUS、D_BUS、S_BUS、DMA；及 4 个从设备 Slave，分别为内部的 Flash 存储器、内部的 SRAM，AHB 外设（AHB2APB 桥 1 对应的外设和 AHB2APB 桥 2 对应的外设）。

3.1.6 AHB/APB 总线桥

借助两个 AHB/APB 总线桥 APB1 和 APB2，可在 AHB 总线与两个 APB 总线之间实现完全同步的连接，从而灵活选择外设频率。

每次芯片复位后，所有外设时钟都被关闭（SRAM 和 Flash 接口除外）。使用外设前，必须在 RCC_AHBENR 或 RCC_APBxENR 寄存器中使能其时钟。

有关 AHB 和 APB 外设地址映射信息，参考 3.2 章节。

3.2 存储器组织架构

3.2.1 简介

Arm® Cortex®-M4 处理器采用哈佛结构，可以使用相互独立的总线来读取指令和加载/存储数据。指令代码和数据都位于相同的存储器地址空间，但在不同的地址范围。程序存储器，数据存储器，寄存器和 I/O 端口都在同一个线性的 4 GB 的地址空间之内。

各字节按小端格式在存储器中编码。字中编号最低的字节被视为该字的最低有效字节，而编号最高的字节被视为最高有效字节。

有关外设寄存器映射的详细信息，请参见相关章节。

可寻址的存储空间分为 8 个主要块，每个块为 512 MB。

3.2.2 存储器映射

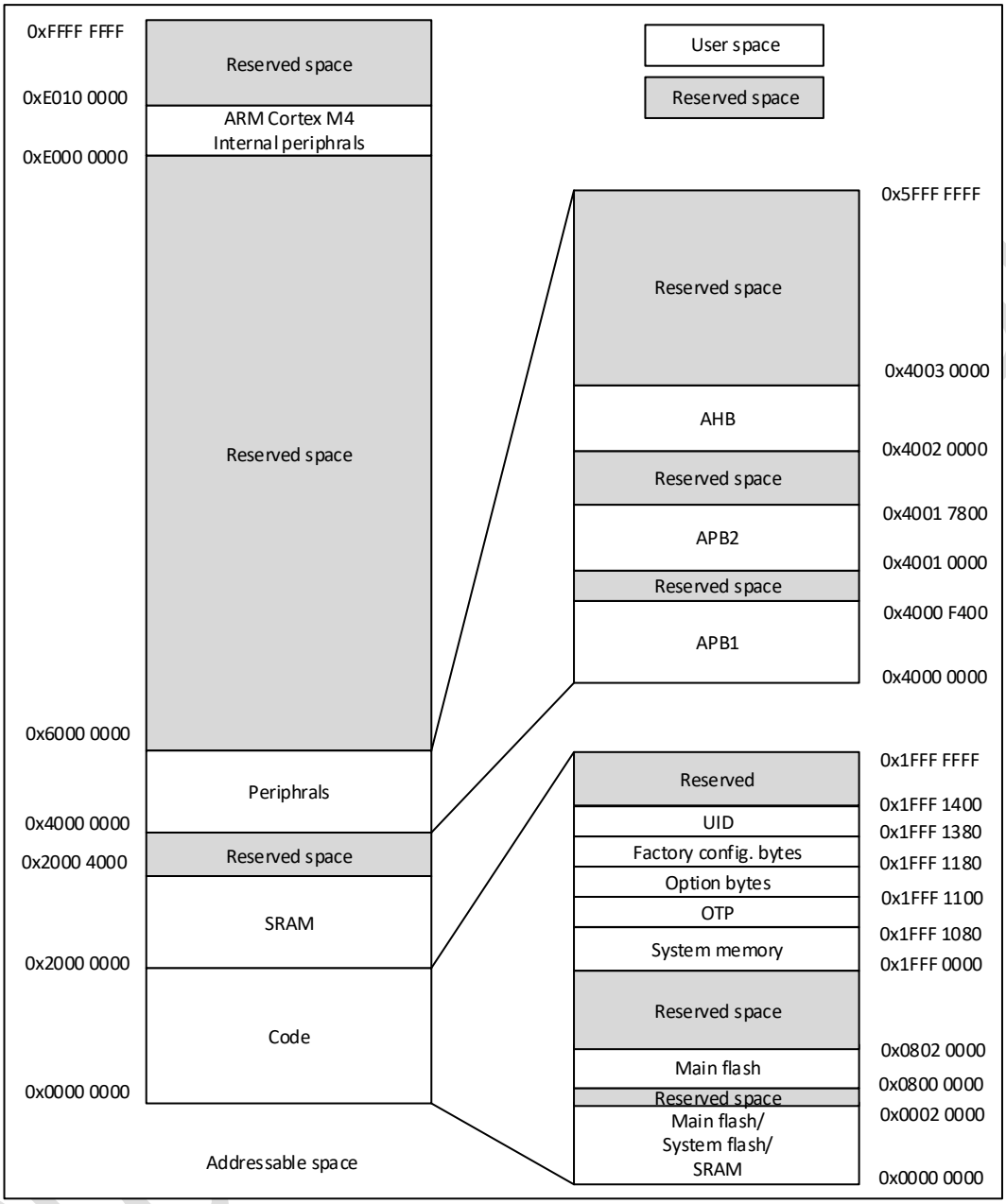


图 3-2 存储器映射

其他所有没有分配给片上存储器和外设的存储器空间都是保留的地址空间。详细的存储和寄存器空间映射参考下面的表格。

下表中给出了产品中可使用的外设及对应的地址边界。

表 3-1 存储器地址

类型	存储器地址	Size	Memory Area	描述
SRAM	0x2002 4000-0x3FFF FFFF	511 MB	保留	1.CPU 读写该空间时产生 Response error，进而进入 HardFault 异常； 2.DMA 访问时产生 TEIF 状态位；
	0x2000 0000-0x2000 3FFF	16 KB	SRAM	SRAM 为 16 KB，地址空间为 0x2000 0000-0x2000 3FFF
Code	0x1FFF 1380 - 0x1FFF 13FF	128 Bytes	UID bytes	Unique ID
	0x1FFF 1180 - 0x1FFF 137F	512 Bytes	Factory config. bytes	-
	0x1FFF 1100 - 0x1FFF 117F	128 Bytes	Option bytes	芯片软硬件 option bytes 信息
	0x1FFF 1080 - 0x1FFF 10FF	128 Bytes	OTP	-
	0x1FFF 0000 - 0x1FFF 107F	4.125 KB	System memory	存放 Boot loader
	0x0802 0000-0x1FFE FFFF	127 MB	保留	-
	0x0800 0000-0x0801 FFFF	128 KB	Main flash memory	-
	0x0002 0000-0x07FF FFFF	127 MB	保留	1.CPU 读写该空间时产生 Response error，进而进入 HardFault 异常 2.DMA 访问时产生 TEIF 状态位
	0x0000 0000-0x0001 FFFF	128 KB	根据 Boot 配置选择是： 1）Main flash memory 2）System memory 3）SRAM	-

1. 上述空间标注为**保留**的空间,无法进行写操作,读为 0，且产生 response error。

表 3-2 外设寄存器地址

存储器起止地址	外设	总线	最大频率
0x4002 5000 - 0x4002 FFFF	保留	AHB	128 MHz
0x4002 4C00 - 0x4002 4FFF	GPIOD		
0x4002 4800 - 0x4002 4BFF	GPIOC		
0x4002 4400 - 0x4002 47FF	GPIOB		
0x4002 4000 - 0x4002 43FF	GPIOA		
0x4002 3400 - 0x4002 3FFF	保留		
0x4002 3000 - 0x4002 33FF	CRC		
0x4002 2400 - 0x4002 2FFF	保留		
0x4002 2000 - 0x4002 23FF	FMC		
0x4002 1400 - 0x4002 1FFF	保留		
0x4002 1000 - 0x4002 13FF	RCC		
0x4002 0400 - 0x4002 0FFF	保留		
0x4002 0000 - 0x4002 03FF	DMA1		
0x4001 7C00 - 0x4001 FFFF	保留	APB2	128 MHz
0x4001 7800 - 0x4001 7BFF	PWM4		
0x4001 7400 - 0x4001 77FF	保留		
0x4001 7000 - 0x4001 73FF	OPAMP		
0x4001 6C00 - 0x4001 6FFF	COMP		
0x4001 6400 - 0x4001 6BFF	保留		
0x4001 6000 - 0x4001 63FF	TIMER17		
0x4001 5C00 - 0x4001 5FFF	TIMER16		
0x4001 5800 - 0x4001 5BFF	TIMER15		
0x4001 4400 - 0x4001 57FF	保留		
0x4001 4000 - 0x4001 43FF	PWM3		
0x4001 3C00 - 0x4001 3FFF	保留		
0x4001 3800 - 0x4001 3BFF	USART1		
0x4001 3400 - 0x4001 37FF	保留		
0x4001 3000 - 0x4001 33FF	SPI1		
0x4001 2C00 - 0x4001 2FFF	TIMER1		
0x4001 2800 - 0x4001 2BFF	保留		
0x4001 2400 - 0x4001 27FF	ADC1		
0x4001 0800 - 0x4001 23FF	保留		
0x4001 0400 - 0x4001 07FF	EXTI		
0x4001 0000 - 0x4001 03FF	SYSCFG		
0x4000 8400 - 0x4000 FFFF	保留	APB1	128 MHz
0x4000 8000 - 0x4000 83FF	LPUART1		

0x4000 7C00 - 0x4000 7FFF	LPTIM1		
0x4000 7400 - 0x4000 7BFF	保留		
0x4000 7000 - 0x4000 73FF	PWR		
0x4000 6000 - 0x4000 6FFF	保留		
0x4000 5800 - 0x4000 5FFF	I2C2		
0x4000 5400 - 0x4000 57FF	I2C1		
0x4000 5000 - 0x4000 53FF	UART2		
0x4000 4C00 - 0x4000 4FFF	UART1		
0x4000 4400 - 0x4000 4BFF	保留		
0x4000 4000 - 0x4000 43FF	PWM2		
0x4000 3C00 - 0x4000 3FFF	保留		
0x4000 3800 - 0x4000 3BFF	SPI2/I2S2		
0x4000 3400 - 0x4000 37FF	PWM1		
0x4000 3000 - 0x4000 33FF	IWDG		
0x4000 2C00 - 0x4000 2FFF	WWDG		
0x4000 2800 - 0x4000 2BFF	RTC		
0x4000 1800 - 0x4000 27FF	保留		
0x4000 1400 - 0x4000 17FF	TIMER7		
0x4000 1000 - 0x4000 13FF	TIMER6		
0x4000 0C00 - 0x4000 0FFF	保留		
0x4000 0800 - 0x4000 0BFF	TIMER4		
0x4000 0400 - 0x4000 07FF	TIMER3		
0x4000 0000 - 0x4000 03FF	TIMER2		

1. 上表 AHB 标注为保留的地址空间，无法写操作，读回为 0，且产生 HardFault。

3.3 嵌入式 SRAM

PY32F410 系列器件内置最大 16 KB 的静态 SRAM，根据不同产品定义，SRAM 容量会有差异，具体请参考数据手册。它可以以字节、半字(16 位)或全字(32 位)访问。SRAM 的起始地址是 0x2000 0000。

3.4 嵌入式 Flash

Flash 存储器有两个不同的物理区域组成：

■ Main flash 区域， 128 KB，有两个 Bank 组成，它包含应用程序和用户数据。用于存储用户程序和用户数据。软件对设定范围外空间的访问会产生 HardFault。

■ Information 区域，5 KB，它包括以下部分：

- Factory config. bytes：512 Bytes
- OTP：128 Bytes，用于存放芯片的 OTP
- UID：128 Bytes，用于存放芯片的 UID
- 选项字节：128 Bytes， 用于存放芯片硬件和存储保护的配置值
- System memory（系统存储器）：4.125 KB， 用于存放 Boot loader

Flash 接口实现基于 AHB 协议的指令读取和数据访问，它也通过寄存器实现了 Flash 的基本 program/erase 等操作。

3.5 位段

Cortex®-M4 存储器映射包括两个位段区域。这些区域将存储器别名区域中的每个字映射到存储器位段区域中的相应位。在别名区域写入字时，相当于对位段区域的目标位执行读-修改-写操作。

在 PY32F410 器件中，外设寄存器和 SRAM 均映射到一个位段区域，这样可实现单个位段的读写操作。这些操作仅适用于 Cortex®-M4 访问，对于其它总线主接口（如 DMA）无效。

可通过一个映射公式说明别名区域中的每个字与位段区域中各个位之间的对应关系。映射公式为：

bit_word_addr = bit_band_base + (byte_offset x 32) + (bit_number × 4)

其中：

— bit_word_addr 代表别名区域中将映射到目标位的字的地址

— bit_band_base 代表别名区域的起始地址

— byte_offset 代表目标位所在位段区域中的字节编号

— bit_number 代表目标位的位位置 (0-7)

示例

下例说明如何将 SRAM 地址 0x20000300 处字节的位 2 映射到别名区域：

0x22006008 = 0x22000000 + (0x300*32) + (2*4)

对地址 0x22006008 执行写操作相当于在 SRAM 地址 0x20000300 处字节的位 2 执行读-修改-写操作。

对地址 0x22006008 执行读操作将返回 SRAM 地址 0x20000300 处字节的位 2 的值（0x01 表示位置位， 0x00 表示位复位）。

有关位段的详细信息，请参见 Cortex®-M4 编程手册（DUI508B_cortex_m4_ugrm.pdf）。

3.6 启动配置

通过配置位 BOOT0 Pin 以及 Option bytes 的配置，可选择三种不同的启动模式，如下表所示：

表 3-3 Boot 配置

Boot mode configuration					Mode
BOOT_LOCK	nBOOT1 FLASH_OPTR2[8]	nBOOT0 FLASH_OPTR2[14]	BOOT0 Pin	nSWBOOT0 FLASH_OPTR2[13]	
1	X	X	X	X	选择 Main flash 作为启动区
0	X	X	0	1	选择 Main flash 作为启动区
0	X	1	X	0	选择 Main flash 作为启动区
0	0	X	1	1	选择 SRAM 作为启动区
0	0	0	X	0	选择 SRAM 作为启动区
0	1	X	1	1	选择 System flash 作为启动区
0	1	0	X	0	选择 System flash 作为启动区

在系统复位后，BOOT0 引脚或用户选项配置位（取决于 FLASH_OPTR 寄存器中的 nSWBOOT0 位值）的值和 nBOOT1 位将被锁存。

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

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

在启动后， CPU 从地址 0x0000 0000 获取堆栈顶的地址，并从启动存储器的 0x0000_0004 指示的地址开始执行代码。因为固定的存储器映像，代码区始终从地址 0x0000_0000 开始（通过 ICode 和 DCode 总线访问），而数据区（SRAM）始终从地址 0x2000_0000 开始（通过系统总线访问）。Cortex M4 的 CPU 始终从 ICode 总线获取复位向量，即启动仅适合于从代码区开始（典型地从 Flash 启动）。PY32F410 系列微控制器实现了一个特殊的机制，可以从片上 SRAM 启动。当从片上 SRAM 启动，在应用程序的初始化代码中，必须使用 NVIC 的异常表和偏移寄存器，重新映射向量表之 SRAM 中。

3.6.1 系统存储启动程序

系统存储器中包含内嵌的引导加载程序，可用于对闪存存储器编程。引导加载程序是通过 USART/UART 接口对闪存存储器进行重新编程。

3.6.2 物理重映射

选择启动引脚后，应用程序软件可以将某些存储器设定为从代码空间进行访问（这样，可通过 ICode 总线而非系统总线执行代码）。这样的修改通过在 SYSCFG 控制器中编程 SYSCFG.MEM_MODE 来实现。因此可重映射以下存储器：

- Main flash
- 系统存储器
- 嵌入式 SRAM

表 3-4 重映射地址

地址	Main flash 启动/重映射	嵌入式 SRAM 中的启动/重映射	系统存储器中的启动/重映射
0x2000 0000 - 0x2000 3FFF	SRAM	SRAM	SRAM
0x1FFF 0000 - 0x1FFF 1400	系统存储器	系统存储器	系统存储器
0x0802 0000 - 0x1FFE FFFF	保留	保留	保留
0x0800 0000 - 0x0801 FFFF	Flash	Flash	Flash
0x0002 0000 - 0x07FF FFFF	保留	保留	保留
0x0000 0000 - 0x0001 FFFF	Flash (使用别名)	SRAM (使用别名)	系统存储器(4.125 KB) (使用别名)

4.嵌入式 Flash 接口(FMC)

4.1 Flash 简介

Flash 接口可管理 CPU 通过 AHB I-Code 和 D-Code 总线对 Flash 进行访问。该接口可针对 Flash 执行擦除和编程操作，并实施读写保护机制。

Flash 接口通过指令预取和缓存机制加速代码执行。

4.2 Flash 主要特征

- 存储器组织结构：
  - Main memory：最大 128 KB
  - Information memory：5 KB
    - Page size：128 Bytes
    - Sector size：4 KB
    - 64-bit wide data read
    - 32-bit wide data write
- Flash 读操作
- Flash 编程操作（页编程，编程的最小单位是页）
- Flash 擦除操作（页擦/扇区擦/全擦，擦除的最小单位是页）
- 读/写保护
- I-Code 上的预取操作
- I-Code 上的分支缓存
- D-Code 上的数据缓存
- 选项字节加载功能
- Flash 编程只能写 0，不能写 1，擦除以后的值是 1

4.3 Flash 功能描述

4.3.1 闪存结构

Flash 存储器由 64 位宽的 cell 组成，具有支持边读边写功能（RWW）的双 Bank 架构，可以用作程序和数据的存储。Page 大小为 128 Bytes，Sector 大小为 4 KB。

从功能上， Flash 存储器分为 Main flash 和 information flash，前者容量最大是 128 KB，后者容量为 5 KB，如下：

表 4-1 闪存结构及边界地址

区域	空间			地址	大小
Main memory	Bank0	Sector 0	Page 0-31	0x0800 0000 - 0x0800 0FFF	4 KB
		Sector 1	Page 32-63	0x0800 1000 - 0x0800 1FFF	4 KB

		Sector 2	Page 64-95	0x0800 2000 - 0x0800 2FFF	4 KB
		Sector 3	Page 96-127	0x0800 3000 - 0x0800 3FFF	4 KB
		…	…	…	…
		Sector 15	Page 480- 511	0x0800 F000 - 0x0800 FFFF	4 KB
	Bank1	Sector 16	Page 512-543	0x0801 0000 - 0x0801 0FFF	4 KB
		Sector 17	Page 544-575	0x0801 1000 - 0x0801 1FFF	4 KB
		Sector 18	Page 576-607	0x0801 2000 - 0x0801 2FFF	4 KB
		…	…	…	…
		Sector 31	Page 992-1023	0x0801 F000 - 0x0801 FFFF	4 KB
Information block	Bank0	System memory		0x1FFF 0000 - 0x1FFF 107F	4.125 KB
		OTP		0x1FFF 1080 - 0x1FFF 10FF	128 bytes
		Option bytes		0x1FFF 1100 - 0x1FFF 117F	128 bytes
		Factory config. bytes		0x1FFF 1180 - 0x1FFF 137F	512 bytes
		UID		0x1FFF 1380 - 0x1FFF 13FF	128 bytes

4.3.2 读访问延迟

为了正确的从 Flash 存储器读取指令/数据，需要根据 CPU 时钟（HCLK)的频率，设置寄存器 FLASH_ACR 的等待周期数（Latency）。

在芯片复位后（包括上电复位，系统复位），HCLK 时钟频率为 8 MHz，Flash 读周期数为 0。

当改变CPU时钟频率或范围时,必须按照以下软件步骤来调整访问Flash存储器所需要的等待周期数。

提高 CPU 频率时的操作步骤

1. 将新的等待周期数编程到 FLASH_ACR 寄存器中的 LATENCY 位
2. 通过读取 FLASH_ACR 寄存器，检查新的等待周期是否设置成功
3. 通过改写 RCC_CFGR 寄存器中的 SW 位来修改 CPU 时钟源
4. 如有需要，可通过改写 RCC_CFGR 中的 HPRE 位来修改 CPU 时钟预分频器
5. 通过读取 RCC_CFGR 寄存器中相应的时钟源状态（SWS 位）和/或 AHB 预分频值（HPRE 位），检查新的 CPU 时钟源和/或新的 CPU 时钟预分频值是否设置成功

降低 CPU 频率时的操作步骤

1. 通过改写 RCC_CFGR 寄存器中的 SW 位来修改 CPU 时钟源
2. 如有需要，可通过改写 RCC_CFGR 中的 HPRE 位来修改 CPU 时钟预分频器
3. 通过读取 RCC_CFGR 寄存器中相应的时钟源状态（SWS 位）和/或 AHB 预分频值（HPRE 位），检查新的 CPU 时钟源和/或新的 CPU 时钟预分频值是否设置成功
4. 将新的等待周期数编程到 FLASH_ACR 中的 LATENCY 位
5. 通过读取 FLASH_ACR 寄存器，检查新的等待周期是否设置成功

4.3.3 自适应实时存储器加速器

专有的自适应实时(ART)存储器加速器面向 ARM® Cortex®-M4 处理器 进行了优化。该加速器很好的体现了 ARM Cortex M4 的固有性能优势，克服了通常条件下，高速的处理器在运行中需要经常等待 Flash 读取的情况。

为了发挥处理器的全部性能，该加速器将实施指令预取队列和分支缓存，从而提高了 64 位 Flash 的程序执行速度。根据 CoreMark 基准测试，凭借 ART 加速器所获得的性能相当于 Flash 在 CPU 频率高达 128 MHz 时以 0 个等待周期执行程序。

- **指令预取**

每个 Flash 读操作可读取 64 位，可以是 2 行 32 位指令，也可以是 4 行 16 位指令，具体取决于烧写在 Flash 中的程序。因此对于顺序执行的代码，至少需要 4 个 CPU 周期来执行前一次读取的 64 位指令行。在 CPU 请求当前指令行时，可使用 I-Code 总线的预取操作读取 Flash 中的下一个连续存放的 64 位指令行。

- **指令缓存存储器**

为了减少因指令跳转而产生的时间损耗，可将 64 行 64 位的指令保存到指令缓存存储器中。每当出现指令缺失（即请求的指令未存在于当前使用的指令行、预取指令行或指令缓存存储器中）时，系统会将新读取的行复制到指令缓存存储器中。如果 CPU 请求的指令已存在于指令缓存区中，则无需任何延时即可立即获取。指令缓存存储器存满后，可采用 LRU（最近最少使用）策略确定指令缓存存储器中待替换的指令行。此特性非常适用于包含循环的代码。

- **数据缓存存储器**

在 CPU 流水线执行阶段，将通过 D-Code 总线访问 Flash 中的数据缓冲区。CPU 流水线只有获得请求的数据后才会继续执行。为了减少因此而产生的时间损耗，通过 AHB 数据总线 D-Code 进行的访问优先于通过 AHB 指令总线 I-Code 进行的访问。此特性的工作原理与指令缓存存储器类似，但保留的数据大小限制在 16 行 64 位以内。

4.3.4 擦除和编程操作

通过 ICP (In-circuit programming) 或者 IAP (In-application programming) 可以对 Flash 进 program 操作。

ICP： 用来更新整个 Flash 存储器的内容，可以使用 JTAG/SWD 协议或者 boot loader，把用户应用装入 MCU 中。

IAP：可以使用芯片支持的通讯接口，下载要编程的数据到 Flash 中。 IAP 允许用户在应用运行时，再次 program flash 存储器。

如果在进行 Flash program 或 erase 操作时，发生了复位，则 Flash 存储器的内容是不被保护的。

在对一个 BANK 进行 program 或者 erase 操作期间， 可以对另外一个 BANK 进行读访问。对正在 program 或者 erase 操作中 BANK 再进行 program 或 erase 操作时，会产生 HardFault。对正在 program 或者 erase 操作中 BANK 再进行读操作时，会拉住总线。

对于 program 和 erase 操作，必须打开 HSI。

4.3.5 闪存解锁

在复位后，Flash 控制寄存器（FLASH_CR）不允许执行写操作，以防止因电气干扰等原因出现对 Flash 的意外操作。每次对 Flash 的 erase 和 program 操作，都必须通过写 FLASH_KEYR 来解锁，以启用对 FLASH_CR 寄存器的访问。

具体步骤如下：

步骤 1：向 FLASH_KEYR 寄存器写入 KEY1=0x4567 0123

步骤 2：向 FLASH_KEYR 寄存器写入 KEY2=0xCDEF 89AB

任何错误的时序都会锁定住 FLASH_CR 寄存器，直到下一次复位。在错误的 KEY 时序时，会产生总线 HardFault 中断。这样的错误包括第一个写周期的 KEY1 不匹配，或者 KEY1 匹配但第二个写周期的 KEY2 不匹配。

FLASH_CR 寄存器可以通过软件写 FLASH_CR 寄存器的 LOCK 位被再次锁定。

注意： 当 FLASH_SR 寄存器的 BSY 位被置位时， FLASH_CR 寄存器不能进行写操作。

4.3.6 闪存擦除操作

Flash 擦除操作支持 page,sector，bank erase，执行 bank0 erase 时不会对 information memory 起作用。

4.3.7 闪存页擦除

页擦除用来对 128 Bytes 的 Main flash 进行 erase 操作，但对 information 区不起作用。

当某个页被 WRP 保护，它是不会被擦除的，此时 WRPERR 位被置位。

页擦除操作的具体步骤如下：

1) 检查 FLASH_SR 寄存器 BSY 位，确认没有正在进行的 Flash 操作
2) 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护
3) 设置 FLASH_CR 寄存器的 PER 位和 EOPIE（如果需要产生 EOP 中断） 位
4) 向该 page 写任意数据（必须 32 位数据）
5) 等待 BSY 位被清零
6) 检查 EOP 标志位被置位
7) 清除 EOP 标志

4.3.8 闪存扇区擦除

扇区擦除用来对 4 KB 的 Main flash 进行擦除操作，但对 information 区不起作用。

当某个扇区被 WRP 保护，它是不会被擦除的，此时 WRPERR 位被置位。

扇区擦除操作的具体步骤如下：

1) 检查 BSY 位，确认是否没有正在进行的 Flash 操作
2) 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器保护
3) 设置 FLASH_CR 寄存器的 SER 位和 EOPIE（如果需要产生 EOP 中断）位
4) 向该 sector 写任意数据（必须 32 位数据）
5) 等待 BSY 位被清零

6) 检查 EOP 标志位被置位

7) 清除 EOP 标志

4.3.9 闪存 Bank 擦除

Bank erase 用来对 64 KB 的 Main flash 中进行 erase 操作，但对 information 区不起作用。

当 WRP 被使能， Bank erase 功能无效，不会产生 Bank erase 操作，并且 WEPERR 位被置位。

Bank erase 操作的具体步骤如下：

1) 检查 BSY 位，确认是否没有正在进行的 Flash 操作

2) 向 FLASH_KEYR 寄存器依次写 KEY1，KEY2，解除 FLASH_CR 寄存器保护

3) 设置 FLASH_CR 寄存器的 MER1 位或 MER0 位和 EOPIE（如果需要产生 EOP 中断）位

4) 向对应 Bank flash 的任意 Flash 空间写任意数据（必须 32 位数据）

5) 等待 BSY 位被清零

6) 检查 EOP 标志位被置位

7) 清除 EOP 标志

4.3.10 闪存写操作

Flash 存储器每次以 32 位(word)为单位（进行 half word 或者 byte 操作会产生 HardFault） 进行整个 page 的 program 操作。当 FLASH_CR 寄存器的 PG 位被置位， CPU 向 FLASH 存储器地址空间写 32 位数据时， program 操作开始启动。

如果要 program 的 Flash 地址空间，是被 FLASH_BANK0_WRPR 或 FLASH_BANK1_WRPR 寄存器设置为保护的区域，则 program 操作会被忽略掉，同时 FLASH_SR 寄存器 WRPERR 位会被置位。

Program 操作的结束， FLASH_CR 寄存器的 EOP 位会被置位。

具体 Flash program 的操作步骤如下所示：

1) 检查 FLASH_SR 寄存器的 BSY 位，判断是否当前没有正在继续的 Flash 操作

2) 如果没有正在进行的 Flash erase 或者 program 操作，则软件读出该 Page 的 32 个 word（如果需要保留该 page 已有的数据，则进行该步骤，否则跳过该步骤）

3) 向 FLASH_KEYR 寄存器依次写 KEY1 和 KEY2，解除 FLASH_CR 寄存器的保护

4) 设置 FLASH_CR 寄存器的 PG 位和 EOPIE（如果需要产生 EOP 中断）位

5) 向目标地址进行第 1 到第 31 个 word 的 program 操作（只接受 32 位的 program）

6) 设置 FLASH_CR 寄存器的 PGSTRT

7) 写第 32 个 word

8) 等待 FLASH_SR 寄存器的 BSY 位被清零

9) 检查 FLASH_SR 寄存器的 EOP 标志位（当 program 操作已经成功，该位被置位），然后软件清除该位

10) 如果不再有 program 操作，则软件清除 PG 位

当上述步骤 7）成功执行，则 program 操作自动启动，同时 BSY 位被硬件置位。

4.4 产品唯一身份标识码（UID）

唯一身份标识码典型应用场景：

- 用作序列号
- 对内部闪存编程时，将其用作密钥或加密原语以提高代码的安全性
- 激活安全自举过程等

产品唯一身份标识提供了一个对于任何设备都唯一的参考号码。

用户永远不能改变这些位。唯一身份标识符也可以以单字节/半字/字等不同方式进行读取，然后使用自定义的算法连接起来。

本产品 UID 内容如下表所示：

基址：0x1FFF 1380

表 4-2 UID 标识码

偏移地址	描述	UID Bits							
		7	6	5	4	3	2	1	0
0	Lot Numer	Lot Number ASCII 码							
1	Lot Numer	Lot Number ASCII 码							
2	Lot Numer	Lot Number ASCII 码							
3	Lot Numer	Lot Number ASCII 码							
4	Wafer Number	Wafer Number							
5	Lot Numer	Lot Number ASCII 码							
6	Lot Numer	Lot Number ASCII 码							
7	Lot Numer	Lot Number ASCII 码							
8	内部编码	内部编码							
9	Y 坐标低位	Y 坐标低位							
10	X 坐标低位	X 坐标低位							
11	X,Y 坐标高地址	Y 坐标高位				X 坐标高位			
12	CP pass ID	CP pass ID							
13	CRC8	0x1FFF 1380 - 0x1FFF 138C 字节地址的数据的 CRC8 校验值							
14	内部编码	内部编码							
15	内部编码	内部编码							

4.5 Flash 选项字节

4.5.1 Flash 选项字节描述

芯片内的 Flash 的 information 区域的部分区域作为选项字节使用，用来存放芯片或者用户针对应用需要对硬件进行的配置。比如，看门狗可以选择为硬件或者软件模式。

为了数据的安全性，选项字节以正文及反码形式分别存储。

表 4-3 字节格式

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

选项字节的内容可以从上表的选项字节结构所述的存储器地址读到，也可以从以下选项字节的相关寄存器读到：

- FLASH 用户选项寄存器 1（FLASH_OPTR1）
- FLASH 用户选项寄存器 2（FLASH_OPTR2）
- FLASH BANK0 写保护地址寄存器（FLASH_BANK0_WRPR）
- FLASH BANK1 写保护地址寄存器（FLASH_BANK1_WRPR）
- FLASH PCROP0SR 地址寄存器（FLASH_PCROP0SR）
- FLASH PCROP0ER 地址寄存器（FLASH_PCROP0ER）
- FLASH PCROP1SR 地址寄存器（FLASH_PCROP1SR）
- FLASH PCROP1ER 地址寄存器（FLASH_PCROP1ER）

表 4-4 选项字节结构

Word Address	Description
0x1FFF 1100	用户选项字节 1 及反码
0x1FFF 1104	用户选项字节 2 及反码
0x1FFF 1108	BANK0 WRP 地址选项字节及反码
0x1FFF 110C	BANK1 WRP 地址选项字节及反码
0x1FFF 1110	PCROP0SR 地址选项字节及反码
0x1FFF 1114	PCROP0ER 地址选项字节及反码
0x1FFF 1118	PCROP1SR 地址选项字节及反码
0x1FFF 111C	PCROP1ER 地址选项字节及反码
…	保留
0x1FFF 117C	保留

4.5.2 Flash 用户选项字节 0

Flash memory address: 0x1FFF 1100

Production value:0x0955 F6AA

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~WWDG_SW	Res	~nRST_STOP	~IWDG_SW	~BFB	Res	~IWDG_STOP	Res	~RDP[7:0]							
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
WWDG_SW	Res	nRST_STOP	IWDG_SW	BFB	Res	IWDG_STOP	Res	RDP[7:0]							
R		R	R	R		R		R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31	~WWDG_SW	R	WWDG_SW 的反码
30	Reserved	-	-
29	~nRST_STOP	R	NRST_STOP 的反码
28	~IWDG_SW	R	IWDG_SW 的反码
27	~BFB	R	BFB 的反码
26	Reserved	-	-
25	~IWDG_STOP	R	IWDG_STOP 的反码
24	Reserved	-	-
23：16	~RDP	R	RDP 的反码
15	WWDG_SW	R	0：硬件窗口看门狗 1：软件窗口看门狗
14	Reserved	-	-
13	nRST_STOP	R	0：进入 Stop 模式时产生 Reset 1：无 Reset 产生
12	IWDG_SW	R	0：硬件独立看门狗 1：软件独立看门狗
11	BFB	R	0：从 Bank0 启动(Bank0 映射在 0x0800 0000 Bank1 映射在 0x0801 0000) 1：从 Bank1 启动(Bank1 映射在 0x0800 0000 Bank0 映射在 0x0801 0000)
10	Reserved	-	-

9	IWDG_STOP	R	设置 IWDG 在 Stop 模式下定时器运行状态 0： 冻结定时器 1： 正常运行
8	Reserved	-	-
7：0	RDP	R	0xAA：level 0，读保护无效 非 0xAA：level 1，读保护有效

4.5.3 Flash 用户选项字节 1

Flash memory address: 0x1FFF 1104

Production value: 0x981F 67E0

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~ BOOT_LOCK	~ nBOOT0	~ nSWBOOT0	Res	Res	~ NRST_MODE		~ BOOT_LOCK	~ ACC_CTRL			Res				
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**
BOOT_LOCK	nBOOT0	nSWBOOT0	Res	Res	NRST_MODE		nBOOT1	ACC_CTRL			Res				
R	R	R			R	R	R	R	R	R					

Bit	Name	R/W	Function
31	~BOOT_LOCK	R	BOOT_LOCK 的反码
30	~nBOOT0	R	nBOOT0 的反码
29	~nSWBOOT0	R	nSWBOOT0 的反码
28:27	Reserved	-	-
26:25	~ NRST_MODE	R	NRST_MODE 的反码
24	~ nBOOT1	R	nBOOT1 的反码
23:21	~ ACC_CTRL	R	ACC_CTRL 的反码
20:16	Reserved	-	-
15	BOOT_LOCK	R	用于强制从用户 Flash 区域启动 0：基于 pad/选项位配置启动 1：从主闪存强制启动
14	nBOOT0	R	nBOOT0 选项位 0:nBOOT0=0 1:nBOOT0=1
13	nSWBOOT0	R	软件 BOOT0 0:BOOT0 取自选项位 nBOOT0 1:BOOT0 取自 PD3/BOOT0 引脚

12:11	Reserved	-	-
10:9	NRST_MODE	R	PD11 pad mode 00: Reset 输入/输出 01: Reset 输入 10: GPIO 11: Reset 输入/输出
8	nBOOT1	R	启动配置与 BOOT0 引脚决定从闪存主存储器、SRAM 或系统存储器中选择启动模式。 请参阅启动配置章节。
7:5	ACC_CTRL	R	ACC 上电初始值控制 ACC_CTRL[2]: 数据缓存使能控制 ACC_CTRL[1]: 指令缓存使能控制 ACC_CTRL[0]: 预取使能控制
4:0	Reserved	-	-

4.5.4 Flash 保护配置 2（BANK0_WRP）

Flash memory address: 0x1FFF 1108

Production value: 0x0000 FFFF

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~BANK0_WRP[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**
BANK0_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~BANK0_WRP	R	BANK0_WRP 的反码
15:0	BANK0_WRP	R	0：sector[y]被保护 1：sector[y]无保护 y=0~15

4.5.5 Flash 保护配置 3（BANK1_WRP）

Flash memory address: 0x1FFF 110C

Production value: 0x0000 FFFF

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
~BANK1_WRP[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
BANK1_WRP[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:16	~BANK1_WRP	R	BANK1_WRP 的反码
15:0	BANK1_WRP	R	0：sector[y]被保护 1：sector[y]无保护 y=0~15

4.5.6 Flash 保护配置 4（PCROP0SR）

Flash memory address: 0x1FFF 1110

Production value: 0xFE00 01FF

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	~ PCROP0SR[8:0]								
							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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0SR[8:0]								
							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	-	-
24:16	~PCROP0SR	R	PCROP0SR 的反码
15:9	Reserved	-	-
8:0	PCROP0SR	R	BANK0 PCROP 区域起始地址(以 Page 为单位)

4.5.7 Flash 保护配置 5（PCROP0ER）

Flash memory address: 0x1FFF 1114

Production value: 0xFFFF 0000

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	~ PCROP0ER[8:0]								
							R	R	R	R	R	R	R	R	R
14	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0ER[8:0]								
							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	-	-
24:16	~PCROP0ER	R	PCROP0ER 的反码
15:9	Reserved	-	-
8:0	PCROP0ER	R	BANK0 PCROP 区域结束地址(以 Page 为单位)

4.5.8 Flash 保护配置 6（PCROP1SR）

Flash memory address: 0x1FFF 1118

Production value: 0xFE00 01FF

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	~ PCROP1SR[8:0]								
							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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1SR[8:0]								
							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	-	-
24:16	~PCROP1SR	R	PCROP1SR 的反码
15:9	Reserved	-	-
8:0	PCROP1SR	R	BANK1 PCROP 区域起始地址(以 Page 为单位)

4.5.9 Flash 保护配置 7（PCROP1ER）

Flash memory address: 0x1FFF 111C

Production value: 0xFFFF 0000

在上电复位（POR/OBL_LAUNCH）释放后，从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	~ PCROP1ER[8:0]								
							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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1ER[8:0]								
							R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Function
31:25	Reserved	-	-
24:16	~PCROP1ER	R	PCROP1ER 的反码
15:9	Reserved	-	-
8:0	PCROP1ER	R	BANK1 PCROP 区域结束地址(以 Page 为单位)

4.5.10 Flash 选项字节编程

复位后，FLASH_CR 寄存器中与选项字节相关的位是被写保护的。当对选项字节进行相关操作前，FLASH_CR 寄存器中的 OPTLOCK 位必须被清零。

以下步骤用来解锁该寄存器：

1. 通过解锁流程，解除 FLASH_CR 寄存器的写保护
2. 向 FLASH_OPTKEYR 寄存器，写 OPTKEY1=0x0819 2A3B
3. 向 FLASH_OPTKEYR 寄存器，写 OPTKEY2=0x4C5D 6E7F

任何错误的时序都会锁住 FLASH_CR 寄存器，直到下一次复位。在错误的 KEY 时序时，会产生总线 HardFault 中断。

User option（information flash 的选项字节）可以通过软件写 FLASH_CR 寄存器的 OPTLOCK 位，被保护住，以防止不要的 erase/program 操作。

如果软件设置 Lock 位，则 OPTLOCK 位也被自动置位。

■ 修改选项字节

选项字节的 program 操作，和 Main flash 的操作不一样。为修改选项字节，需要进行如下步骤：

1. 用之前描述的步骤，清零 OPTLOCK 位
2. 检查 BSY 位，确认没有正在进行的 Flash 操作
3. 向选项字节寄存器 FLASH_OPTR1/FLASH_OPTR2/ FLASH_BANK0_WRPR/FLASH_BANK1_WRPR/FLASH_PCROP0SR/ FLASH_PCROP0ER/FLASH_PCROP1SR/FLASH_PCROR1ER 写期望的值（1~8 个 word）
4. 设置 OPTSTRT 位和 EOPIE 位
5. 向 0x1FFF 1100 地址写任意数据（必须 32bit 数据）
6. 等待 BSY 位被清零
7. 检查 EOP 标志位被置位

8. 清除 EOP 标志

任何对选项字节的改动，硬件都会先把选项字节对应的整个 page erase 掉，然后用 FLASH_OPTR1、FLASH_OPTR2 、 FLASH_BANK0_WRPR 、 FLASH_BANK1_WRPR 、 FLASH_PCROR0SR 、FLASH_PCROR0ER、FLASH_PCROR1SR、FLASH_PCROR1ER 寄存器的值，写入选项字节中。并且，硬件自动计算相应的补码，并把计算值写入选项字节的相应区域。

- **加载选址字节**

在 BSY 位被清零后，所有新的选项字节被写入了 Flash information 存储器中，但是未应用于芯片系统。对选项字节寄存器进行读操作，仍然返回上一次被装载的选项字节里的值。仅当他们 (新值) 被装载后，才对芯片系统起作用。

选项字节的装载，在以下两种情况下进行：

1. 当 FLASH_CR 寄存器中的 OBL_LAUNCH 位被置位
2. 在上电复位后（POR）

“装载选项字节”进行的操作是：对 information memory 区域的选项字节进行读操作，再把读出的数据存储在内部 option 寄存器中 （FLASH_OPTR1 、 FLASH_OPTR2 、 FLASH_BANK0_WRPR 、FLASH_BANK1_WRPR 、 FLASH_PCROR0SR 、 FLASH_PCROR0ER 、 FLASH_PCROR1SR 、FLASH_PCROR1ER）。

每个 option 位在它相同的双字地址 (下一个 half word) 有相应的补码。在选项字节装载期间，会对 option bit 和其反码进行验证，确保装载被正确进行。

如果正反码匹配，则选项字节被复制到 option 寄存器中。

如果正反码不匹配，则 FLASH_SR 寄存器的 OPTVERR 状态位被置位。不匹配的值被写入 option 寄存器：

- 对于 OPTR1 和 OPTR2
  - RDP 位写成 0xFF（即 level 1）
  - BFB 为写为 0（即不进行 bank 交换）
  - 其余不匹配的值都写成 1
- 对于 WRP option，不匹配的值是默认值“无保护”
- 对于 PCROP option，不匹配的值是默认值“全闪存保护”

在系统复位后，选项字节的内容被复制到下面的 option 寄存器（软件可读可写）：

- FLASH_OPTR1
- FLASH_OPTR2
- FLASH_BANK0_WRPR
- FLASH_BANK1_WRPR
- FLASH_PCROP0SR
- FLASH_PCROP0ER
- FLASH_PCROP1SR
- FLASH_PCROP1ER

这些寄存器也被用来修改选项字节。

4.6 Flash 用户数据字节

芯片内的 Flash 的 information 区域的部分区间作为 Flash USER OTP memory Bytes。

共计 1 Page。

表 4-5 USER OTP 结构

Page	Word	Address	Contents
1	0	0x1FFF 1080	Bit[31:16]:存放用户数据 Bit[15:0]: USER OTP MEMORY LOCK
	1	0x1FFF 1084	存放用户数据
	…	…	存放用户数据
	31	0x1FFF 10FC	存放用户数据

这 1 个 Page 配置在 information 区域，对本 Page 区域写和擦是按照 Main flash 的方法来处理。另外，Main flash 区域的 mass erase 对本区域无效。

设定 USER OTP MEMORY LOCK 内容不会立刻更新，直到上电复位（POR/PDR），从会起到保护功能。

对本 Page write 有如下保护。

表 4-6 Flash USER OTP 写保护

USER OTP MEMORY_LOCK	Write protection
0xAA55	读：可以 program 和擦操作：不可以
除(0xAA55)之外的任何值	读、program 和擦操作：可以

4.7 Flash 存储区保护

对 Main flash memory 的保护包括以下几种机制：

- Read protection(RDP)，防止来自外部的访问。
- Wrtie protection（WRP）控制，以防止不要的写操作（由于程序存储器指针的混乱）。写保护的粒度设计为 4 KB。
- 选项字节写保护，有专门的解锁设计。

4.7.1 闪存读保护(RDP)

通过设置 RDP 选项字节，并进行系统复位 (POR 或者 OBL 复位)装载新的 RDP 选项字节，可以激活读保护功能。RDP 保护 Main flash memory 和备份寄存器。

如果通过 SWD 的 debug 仍在连接时，读保护被设置，需要进行上电复位而不是系统复位。

当 RDP 选项字节和反码成对正确存在于选项字节时，Flash memory 会被保护。

表 4-7 闪存读保护状态

RDP 选项字节值	RDP 选项字节反码值	等保护等级
0xAA	0x55	Level 0
除[0xAA,0x55]组合的任何值		Level 1

无论任何保护级别，System memory 都是可读的，但不能进行 program 和 erase 操作。

Level 0: no protection

对 main flash 的读、编程和擦除操作是可以的，对选项字节和备份寄存器也是可以进行任何操作。

Level 1: Read protection

当选项字节里的 RDP 及其反码包含任何除了[0xAA,0x55]之外的组合，则 level 1 read protection 生效，Level 1 是默认的保护级别。

- User mode：在用户模式下执行的程序（boot from Main flash），可以对 Main flash、选项字节和备份寄存器进行所有操作。
- Debug,boot from SRAM, and boot from system memory modes：在 debug 模式，或者当从 SRAM 或者 system memory 启动，Main flash 和备份寄存器是不能被访问的。在这些模式下，对 Main flash 读或者写访问会产生总线 HardFault 中断。

读保护级别可以改变：

- 从 Level 0 到 Level 1，将 RDP 字节的值更改为除 0xAA 之外的任何值
- 从 Level 1 到 Level 0，将 RDP 字节的值更改为 0xAA

Level 1 变为 Level 0，会触发硬件 mass erase Main flash。

表 4-8 访问状态与保护级别和执行模式的关系

Area	Protection level	User execution (Boot From Flash)			Debug/ Boot From Ram/ Boot From Loader		
		Read	Write	Erase	Read	Write	Erase
Main memory	0	Yes	Yes	Yes	Yes	Yes	Yes
	1	Yes	Yes	Yes	No	No	No
System memory	x	Yes	No	No	Yes	No	No
选项字节	x	Yes	Yes	N/A	Yes	Yes	N/A
OTP	x	Yes	Yes	Yes	Yes	Yes	Yes
Backup registers	0	Yes	Yes	Yes	Yes	Yes	Yes
	1	Yes	Yes	Yes	No	No	No

注：

(1) Information 区域只读，和保护等级及执行模式无关。

(2) RDP 从 Level 1 修改为 Level 0，会触发硬件对 Main flash 和 Backup register 的 mass erase。

4.7.2 闪存写保护（WRP)

Flash 可以被设置成写保护，以应对不想要的写操作。定义 WRP 寄存器每 bit 的控制粒度为 4 KB 的写保护（WRP）区域，即 1 个 sector 大小。具体参见 WRP 寄存器的描述。

当 WRP 的区域被激活，则不允许进行 erase 或者 program 操作。相应的，即使只有一个区域被设定为写保护，则 bank erase 功能不起作用。此外，如果尝试对设为写保护的区域进行 erase 或者 program 操作，则 FLASH_SR 寄存器的写保护错误标识（WRPERR）会被置位。

注：写保护仅对 Main flash 起作用，对 system memory 不起作用。

4.7.3 专有代码读出保护(PCROP)

除了闪存之外，还可以防止来自第三方的读写。保护区仅执行：它只能由 CPU 作为指令代码访问，而严格禁止所有其他访问（DMA、调试和 CPU 数据读取、写入和擦除）。当 RDP 保护从级别 1 更改为级别 0 时， 包括 PCROP 区域也会被擦除（请参阅更改读取保护级别）。

每个 PCROP 区域由与物理闪存地址相关的起始页和结束页地址定义。这些地址定义在 PCROP 地址寄存器 FLASH PCROP 起始地址寄存器（FLASH_PCROP0SR，FLASH_PCROP1SR）、FLASH PCROP 结束地址寄存器（FLASH_PCROP0ER，FLASH_PCROP1ER）。

通过 D-code 总线对 PCROP 保护区执行的任何读取访问都会触发 RDERR 标志错误。

任何受 PCROP 保护的地址也是写保护的，对这些地址之一的任何写访问都会触发 WRPERR。

任何 PCROP 区域也受到擦除保护。因此，对该区域中的页面的任何擦除都是不可能的（包括包含该区域的开始地址和结束地址的页面）。此外，如果一个区域受到 PCROP 保护，则不能执行软件 Bank Mass 擦除。

只有当 RDP 从级别 1 更改为级别 0 时，才能禁用 PCROP。如果通过修改用户选项试图清除 PCROP 或缩小 PCROP 区域，可启动选项编程，但 PCROP 区域保持不变。相反，可以增加 PCROP 区域。

表 4-9 PCROP 保护

PCROPx 寄存器值(x = 0,1)	PCROP 保护区
PCROPxSR >PCROPxER	无 PCROP 区域。
PCROPxSR ≤PCROPxER	PCROPxSR 和 PCROPxER 之间的区域受到保护。

4.7.4 强制从 Main flash 启动

为了提高安全性，FLASH_OPTR2 寄存器的 BOOT_LOCK 选项位允许强制系统从主闪存启动，从而无视其他启动选项。重置 BOOT_LOCK 只有在以下情况：

- RDP 设置为级别 0
- RDP 级别 1 修改为级别 0

4.8 Flash 中断

表 4-10 Flash 中断请求

中断事件	事件标志	时间标志/中断清除方法	控制位使能
End of operation	EOP	Write EOP=1	EOPIE
Write protection	WRPERR	Write WRPERR=1	ERRIE
Read error	RDERR	Write RDERR=1	RDERRIE

以下事件没有单独的中断标识，但会产生 HardFault：

- 解锁 Flash memory 的 FLASH_CR 寄存器的流程错误
- 解锁 Flash 选项字节的写操作流程错误
- Flash program、erase 操作未进行 32 位数据的对齐
- 擦写过程中再次发起擦写请求

4.9 Flash 寄存器

4.9.1 Flash 访问控制寄存器(FLASH_ACR)

偏移地址： 0x00

复位值： 0x0000 0700

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

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	DCEN	RW	1	数据缓存使能 0：数据缓存禁止 1：数据缓存使能 注：该位的初始值，还受 FLASH_OPTR2.ACC_CTRL[2]位影响，如果 FLASH_OPTR2.ACC_CTRL[2]设为 0，则该位的默认值为 0
9	ICEN	RW	1	指令缓存使能 0：指令缓存禁止 1：指令缓存使能 注：该位的初始值，还受 FLASH_OPTR2.ACC_CTRL[1]位影响，如果 FLASH_OPTR2.ACC_CTRL[1]设为 0，则该位的默认值为 0
8	PRFTEN	RW	1	预取使能位 0：预取禁止 1：预取使能 注：该位的初始值，还受 FLASH_OPTR2.ACC_CTRL 位[0]影响，如果 FLASH_OPTR2.ACC_CTRL[0]设为 0，则该位的默认值为 0
7:4	Reserved	-	-	保留
3:0	LATENCY[3:0]	RW	4'h0	Flash 读操作对应的等待状态： HCLK ≤ 25 MHz

				0：Flash 读操作没有等待状态，即每次读 Flash 需要 1 个系统时钟周期 25 MHz < HCLK ≤ 50 MHz 1：Flash 读操作有 1 个等待状态，即每次读 Flash 需要 2 个系统时钟周期 50 MHz < HCLK ≤ 75 MHz 3：Flash 读操作有 3 个等待状态，即每次读 Flash 需要 4 个系统时钟周期 75 MHz < HCLK ≤ 100 MHz 4：Flash 读操作有 4 个等待状态，即每次读 Flash 需要 5 个系统时钟周期 100 MHz < HCLK ≤ 125 MHz 5：Flash 读操作有 5 个等待状态，即每次读 Flash 需要 6 个系统时钟周期 HCLK >125 MHz 6：Flash 读有 6 个等待状态，即每次读 Flash 需要 7 个系统时钟周期 ... Flash 可以配置比上表设置更高的延迟，为了降低功耗。比如，在 HCLK >125 MHz 的频率条件下，设置 7~15 个等待状态。 注：在配置 latecy 位时，该寄存器其他位不能写入任何值

4.9.2 Flash 密钥寄存器(FLASH_KEYR)

偏移地址： 0x08

复位值： 0x0000 0000

所有寄存器位是 write-only，读出返回 0。

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

Bit	Name	R/W	Reset Value	Function
31:0	KEY[31:0]	W	32’h0	下面的值必须被连续的写入，才能解锁 FLASH_CR 寄存器，并使能 Flash 的 program/erase 操作 KEY1: 0x4567 0123 KEY2: 0xCDEF 89AB

4.9.3 Flash 选项密钥寄存器 (FLASH_OPTKEYR)

偏移地址：0x0C

复位值：0x0000 0000

所有寄存器位是 write-only，读出返回 0。

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

Bit	Name	R/W	Reset Value	Function
31:0	OPTKEY[31:0]	W	32'h0	下面的值必须被连续的写入，才能解锁 Flash 的 option 寄存器，并使能选项字节的 program/erase 操作 KEY1: 0x0819 2A3B KEY2: 0x4C5D 6E7F

4.9.4 Flash 状态寄存器(FLASH_SR)

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BSY1	BSY0
														R	R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OPTV ERR	RD ERR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WRP ERR	Res.	Res.	Res.	EOP
RC_W1	RC_W1										RC_W1				RC_W1

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17	BSY1	R	0	Busy 位 该位表示 Flash bank1 的操作正在进行。该位在 Flash 操作的开始被硬件置位，当操作完成或者错误产生时由硬件清零。
16	BSY0	R	0	Busy 位 该位表示 Flash bank0 的操作正在进行。该位在 Flash 操作的开始被硬件置位，当操作完成或者错误产生时由硬件清零。
15	OPTVERR	RC_W1	0	选项字节加载错误

				当 option 及其反码不匹配时，硬件置位该位。装载不匹配的选项字节，被强制成安全值，参考 FLASH 选项字节。 软件写 1，清零。
14	RDERR	RC_W1	0	PCROP 读取错误 当要通过 D-code 总线读取的地址属于闪存的读取保护区域时由硬件置位（PCROP 保护）。 软件写 1，清零。
13:5	Reserved	-	-	保留
4	WRPERR	RC_W1	0	写保护错误 当要被 program/erase 的地址处于被写保护的 Flash 区域时（WRP），硬件置位该位。 软件写 1，清零。
3:1	Reserved	-	-	保留
0	EOP	RC_W1	0	当 Flash 的 program/erase 操作成功完成，硬件置位。该位仅当 FLASH_CR 寄存器的 EOPIE 位使能才会被置位。 软件写 1，清零。

4.9.5 Flash 控制寄存器(FLASH_CR)

偏移地址：0x14

复位值：0xC000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOCK	OPT LOCK	Res.	Res.	OBL_ LAUN CH	RD ERR IE	ERR IE	EOP IE	Res.	Res.	Res.	Res.	PG STRT	Res.	OPT STRT	Res.
RS	RS			RC_W1	RW	RW	RW					RW		RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	SER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MER1	MER0	PER	PG
				RW								RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Lock	RS	1	FLASH_CR Lock 位。 软件对该位只能置位。当置位后，FLASH_CR 寄存器被锁住。当成功给出解锁流程后，该位被硬件清零。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的解锁流程给出，该位仍然保持置位状态，直到下一次系统复位。
30	OPTLOCK	RS	1	选项字节 Lock 位。

				软件对该位只能置位。当置位后，FLASH_CR 寄存器中与选项字节有关的位被锁住。当成功给出解锁流程后，该位被硬件清零。 【软件要在 program/erase 操作完成后，置位该位】 当不成功的解锁时序给出，该位仍然保持置位状态，直到下一次系统复位。
29：28	Reserved	-	-	保留
27	OBL_LAUNCH	RC_W1	0	强制选项字节加载 当置位时，该位强制系统进行选项字节的重装载。该位仅当选项字节装载被完成后被硬件清零。如果 OPTLOCK 位被置位，该位不能被写. 0：选项字节加载完成 1：产生选项字节加载请求，产生系统复位，进行选项字节的重装载。
26	RDERRIE	RW	0	PCROP 读取错误中断使能 当 FLASH_SR 中的 RDERR 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
25	ERRIE	RW	0	错误中断使能 当 FLASH_SR 寄存器的 WRPERR 位被置位，如果该位使能，则产生中断请求。 0：无中断产生 1：有中断产生
24	EOPIE	RW	0	操作结束中断使能 当 FLASH_SR 寄存器的 EOP 位被置位，该位使能中断的产生。 0：EOP 中断关闭 1：EOP 中断使能
23:20	Reserved	-	-	保留
19	PGSTRT	RW	0	Flash main memory 的 program 操作的启动位。 该位启动了 Flash main memory 的 program 操作，软件置位，在 FLASH_SR 寄存器的 BSY 位被清零后，硬件清零该位。
18	Reserved	-	-	保留
17	OPTSTRT	RW	0	Flash 选项字节修改的启动位 该位启动了对选项字节的修改。软件置位，在 FLASH_SR 寄存器的 BSY 位被清零后，硬件清零该位。

				注意：当对 Flash 选项字节进行修改时，硬件自动把整个 128 Bytes 的 page 进行 erase 操作，再进行 program 操作，其中也包括自动进行反码的写入。
16:12	Reserved	-	-	保留
11	SER	RW	0	Sector erase 操作 0：未选择 Flash 的 sector erase 操作 1：选择 Flash 的 sector erases 操作
10:4	Reserved	-	-	保留
3	MER1	RW	0	Bank 1 erase 操作 0：未选择 Flash 的 bank1 erase 操作 1：选择 Flash 的 bank1 erases 操作
2	MER0	RW	0	Bank 0 erase 操作 0：未选择 Flash 的 bank0 erase 操作 1：选择 Flash 的 bank0 erases 操作
1	PER	RW	0	Page erase 操作 0：未选择 Flash 的 page erase 操作 1：选择 Flash 的 page erase 操作
0	PG	RW	0	Program 操作 0：未选择 Flash 的 program 操作 1：选择 Flash 的 program 操作

4.9.6 Flash 选项 1 寄存器(FLASH_OPTR1)

偏移地址：0x20

复位值： 0x0000 XXXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WWDG_SW.	Res.	nRSTSTOP	IWDG_SW	BFB	Res.	IWDG_STOP	Res.	RDP[7:0]							
RW		RW	RW	RW		RW		RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	WWDG_SW	RW	-	0：硬件窗口看门狗 1：软件窗口看门狗

14	Reserved	-	-	保留
13	nRST_STOP	RW	-	0：进入 Stop 模式时产生 Reset 1：无 Reset 产生
12	IWDG_SW	RW	-	0：硬件看门狗 1：软件看门狗
11	BFB	RW	-	0：从 Bank0 启动(Bank0 映射在 0x0800 0000 Bank1 映射在 0x0801 0000) 1：从 Bank1 启动(Bank1 映射在 0x0800 0000 Bank0 映射在 0x0801 0000)
10	Reserved	-	-	保留
9	IWDG_STOP	RW	-	设置 IWDG 在 Stop 模式下定时器运行状态 0：冻结定时器 1：正常运行
8	Reserved	-	-	保留
7:0	RDP	RW	-	0xAA：level 0，读保护无效 非 0xAA：level 1，读保护有效

4.9.7 Flash 选项 2 寄存器(FLASH_OPTR2)

偏移地址：0x24

复位值： 0x0000 XXXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的 option bit。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BOOT_LOCK	nBOOT0	nSWBOOT0	Res.	Res.	NRST_MODE		nBOOT1	ACC_CTRL			Res.				
RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	BOOT_LOCK	RW	-	用于强制从用户 Flash 区域启动 0：基于 pad/选项位配置启动 1：从主闪存强制启动
14	nBOOT0	RW	-	nBOOT0 选项位 0：nBOOT0=0

				1:nBOOT0=1
13	nSWBOOT0	RW	-	软件 BOOT0 0:BOOT0 取自选项位 nBOOT0 1:BOOT0 取自 PD3/BOOT0 引脚
12:11	Reserved	-	-	保留
10:9	NRST_MODE	RW	-	PD11 pad mode 00: Reset 输入/输出 01: Reset 输入 10: GPIO 11: Reset 输入/输出
8	nBOOT1	RW	-	启动配置与 BOOT0 引脚决定从闪存主存储器、SRAM 或系统存储器中选择启动模式。 请参阅启动配置该节。
7:5	acc_ctrl	RW	-	ACC 上电初始值控制 acc_ctrl [2]: 数据缓存使能控制 acc_ctrl [1]: 指令缓存使能控制 acc_ctrl [0]: 预取使能控制
4:0	Reserved	-	-	保留

4.9.8 Flash BANK0 WRP 地址寄存器 (FLASH_BANK0_WRPR)

偏移地址：0x2C

复位值： 0x0000 XXXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BANK0_WRP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BANK0_WRP [15:0]	RW	-	Write protection，每一位对应一个 sector 0：sector N，写保护有效 1：sector N，写保护无效 N=0-15

4.9.9 Flash BANK1 WRP 地址寄存器 (FLASH_BANK1_WRPR)

偏移地址： 0x30

复位值： 0x0000 XXXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的 option bytes 区域读出相应的值，写入到该寄存器相应的

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BANK1_WRP[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BANK1_WRP [15:0]	RW	-	写保护，每一位对应一个 sector 0：sector N，写保护有效 1：sector N，写保护无效 N=0-15

4.9.10 Flash PCROP0 起始地址寄存器 (FLASH_PCROP0SR)

偏移地址：0x34

复位值： 0x0000 0xXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0SR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	PCROP0SR	RW	-	BANK0 PCROP 区域起始地址(以 Page 为单位)

4.9.11 Flash PCROP0 结束地址寄存器 (FLASH_PCROP0ER)

偏移地址：0x38

复位值： 0x0000 0xXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP0ER[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	PCROP0ER	RW	-	BANK0 PCROP 区域结束地址(以 Page 为单位)

4.9.12 Flash PCROP1 起始地址寄存器 (FLASH_PCROP1SR)

偏移地址：0x3C

复位值： 0x0000 0xXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1SR[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	PCROP1SR	RW	-	BANK1 PCROP 区域起始地址(以 Page 为单位)

4.9.13 Flash PCROP1 结束地址寄存器 (FLASH_PCROP1ER)

偏移地址：0x40

复位值： 0x0000 0xXX。

在上电复位（POR/OBL_LAUNCH）释放后,从 Flash information memory 的选项字节区域读出相应的值，写入到该寄存器相应的。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	PCROP1ER[8:0]								
							RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	PCROP1ER	RW	-	BANK1 PCROP 区域结束地址(以 Page 为单位)

5. 电源控制（PWR）

5.1 PWR 简介

PWR 模块主要完成如下功能：

- 芯片上电和掉电流程控制
- 模拟 PMU 控制信号产生
- 存储器低功耗控制信号产生
- 低功耗模式进入和唤醒时序控制
- PVD 功能

5.2 PWR 电源

5.2.1 电源结构

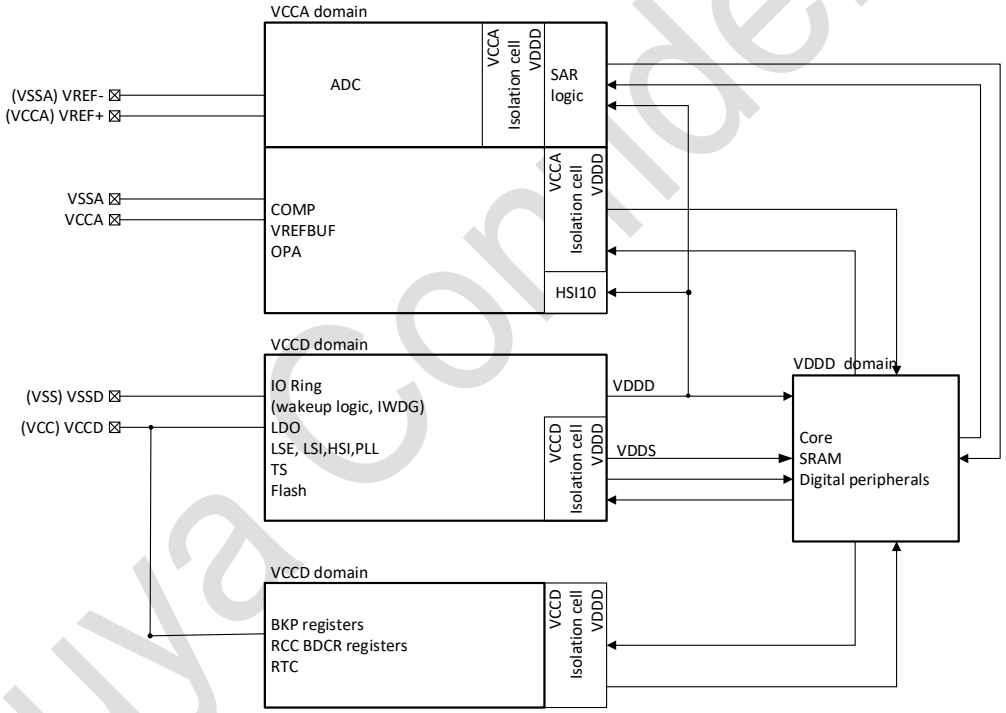


图 5-1 电源框图

表 5-1 电源框图

编号	电源	电源值	描述
1	V_{CC}	2.0 ~ 3.6 V	通过电源管脚为芯片提供电源。
2	V_{DDD}	1.2 V（默认）	来自于 VR 的输出，为芯片内部主要逻辑电路、SRAM 供电。MR、LPR、DLPR 三种模式可以选。
3	V_{CCA}	2.0 ~ 3.6 V	通过电源管脚为芯片模拟电路提供电源。

5.2.2 调压器（VR）

嵌入式线性 VR 为备份域和电路以外的所有数字电路供电（V_{DDD}）。

调压器在复位后始终处于使能状态。根据应用模式的不同，可采用两种不同的模式工作：

- 在运行模式中，VR 为 V_{DDD} 域（内核、存储器和数字外设）提供全功率。在该模式下，调压器输出电压可通过软件调节为不同的电压值。
- 在停止模式下，主调压器或低功耗调压器为 V_{DDD} 域提供低功率电压，从而保存寄存器和内部 SRAM 的内容。还可以将调压器置于主调压器模式（MR）或低功耗模式（LPR/DLPR）。

5.2.3 动态电压调节

该设备采用嵌入式线性稳压器——主稳压器（MR）为大部分数字电路供电。主稳压器在运行模式和睡眠模式下工作。

该设备支持动态电压调节功能，可在运行模式下优化功耗。为逻辑电路供电的主稳压器输出电压，可根据系统最高运行频率进行动态调整。

主调节器（MR）在以下范围内运行：

- 高电压（Range 1）模式，CPU 运行频率最高可达 128 MHz。
- 中电压（Range 2）模式，CPU 运行频率最高可达 96 MHz。
- 低电压（Range 3）模式，CPU 运行频率最高可达 64 MHz。

MR 模式从低电压切换到高电压：

MR 模式下，系统时钟频率从低频到高频，电压从低压到高压的切换步骤如下：

1）电源控制寄存器 1 (PWR_CR1) 中的 LPR 位配置为 0

2）根据支持的频率，参考 MR_VSEL[1:0]中支持的频率配置合适的 MR 电压模式

3）等待 7 μs

4）根据支持的频率配置 FLASH_ACR.LATENCY 的值

5）切换系统时钟频率

MR 模式从高电压切换到低电压：

MR 模式下，系统时钟频率从高频到低频，电压从高压到低压的切换步骤如下：

1）电源控制寄存器 1 (PWR_CR1) 中的 LPR 位配置为 0

2）切换系统频率

3）根据支持的频率配置 FLASH_ACR.LATENCY 的值

4）根据支持的频率，参考 MR_VSEL[1:0]中支持的频率配置合适的 MR 电压模式

5.3 PWR 电源检测

5.3.1 上电复位(POR)/下电复位(PDR)

芯片包含 Power-on reset（POR）、Power-off reset（PDR）模块，提供电源相关复位。其中 POR/PDR 在所有 power 模式下都默认工作。POR/PDR 输出复位低电平有效。

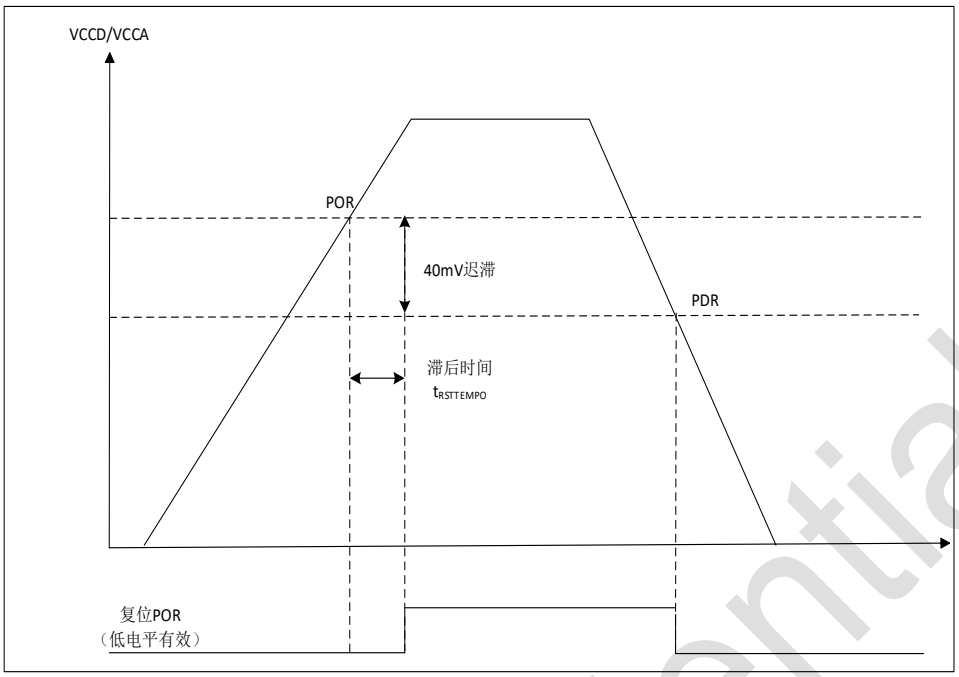


图 5-2 POR/PDR 阈值

5.3.2 可编程电压检测器 (PVD)

PVD 模块是检测 V_{CC} 是否低于 PWR_CR2.PLS 设置的阈值。

通过配置 PWR_CR2.PVDE 寄存器使能 PVD 功能。

电源控制/状态寄存器(PWR_CSR)中的 PVDO 标志用来表明 V_{CC} 是高于还是低于 PVD 的电压阀值。该事件在内部连接到 EXTI 的输入，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 V_{CC} 下降到 PVD 阀值以下或当 V_{CC} 上升到 PVD 阀值之上时，根据 EXTI 的上升/下降沿触发设置，就会产生 PVD 中断。实际应用中，这一特性可用于执行紧急关闭任务。

5.4 PWR 系统低功耗模式

默认情况下，系统复位或上电复位后，微控制器进入运行模式。在运行模式下， CPU 通过 HCLK 提供时钟，并执行程序代码。系统提供了多个低功耗模式，可在 CPU 不需要运行时（例如等待外部事件时）节省功耗。由用户根据应用选择具体的低功耗模式，在低功耗、短启动时间和可用唤醒源之间寻求最佳平衡。

器件有四种低功耗模式：

- 睡眠模式（Sleep）：Cortex®-M4 内核停止工作，外设包括 Cortex®-M4 的外设（NVIC，Sys Tick）等可以配置为运行
- 低功耗运行模式（Low-power run）：HCLK 配置为低于 2 MHz，V_{DDD} 由 LPR 供电
- 低功耗睡眠模式（Low-power sleep）：从 Low-power run 进入 Sleep 模式
- 停止模式（Stop）：
  - Stop0：MR 供电，HSI 可以打开
  - Stop1：LPR 供电，HSI 不可以打开
  - Stop2：DLPR 供电，HSI 不可以打开，V_{DDD} 电源域模块不能工作，V_{CCD} 电源域模块正常工作

此外，可通过下列方法之一降低运行模式的功耗：

- 降低系统时钟速度
- 不使用 APBx 和 AHBx 外设时，将对应的外设时钟关闭

注意：如果系统时钟源再进入低功耗模式之前是高速时钟源(PLL/HSE)，为了保证系统时钟切换成功，需要软件切换系统时钟源为 HSI。

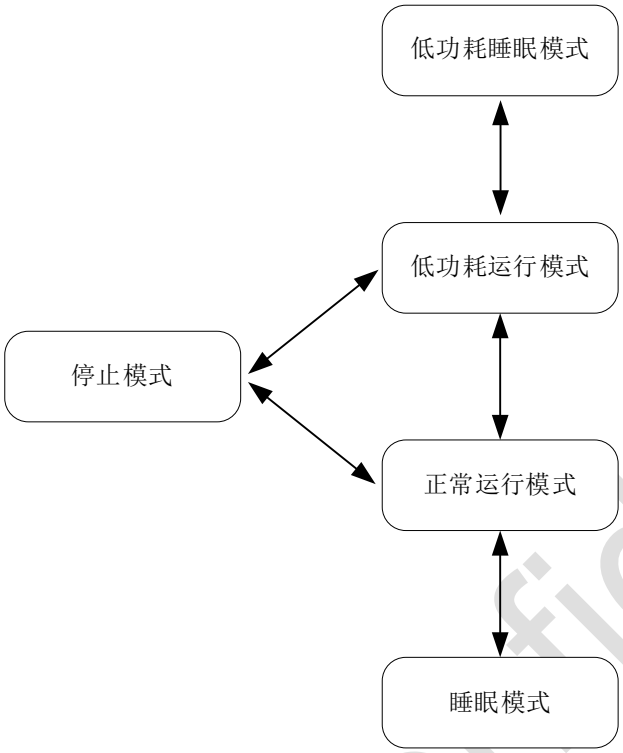


图 5-3 低功耗模式状态转移图

表 5-2 低功耗模式概要

大模式	模式	时钟状态	进入条件	退出条件	唤醒后系统时钟	MR	LPR	DLPR
						状态	状态	状态
Run 模式	Run	任意时钟	-	-	-	开	关	关
Sleep 模式	Sleep	CPU 时钟关闭	WFI 或者从中断返回 SLEEPDEEP = 0	任意中断	进入前时钟			
			WFE	唤醒事件			-	
LP 模式	Lower power run	系统时钟:HSI8M 4 分频	lpr=1	lpr=0		-	开	
	Lower power sleep	CPU 时钟关闭	WFI 或者从中断返回 SLEEPDEEP = 0 lpr=1	任意中断				
			WFE lpr=1	唤醒事件		-		
Stop 模式	Stop0	HSI/LSI/LSE 可以选择打开 其他时钟关闭	LPMS=00 SLEEPDEEP=1 WFI 或者从中断返回或者 WFE	任意 EXTI 特定 外设唤醒事件		开	关	
	Stop1	除 LSE,LSI 外其他时钟关闭	LPMS=01 SLEEPDEEP=1 WFI 或者从中断返回或者 WFE			关	开	
	Stop2	除 LSE,LSI 外其他时钟关闭 (VDD 域电路保持、不工作)	LPMS=10 SLEEPDEEP=1 WFI 或者从中断返回或者 WFE			关	关	开

下表为不同模式下，各个功能模块是否存在的情况：注 (1)

表 5-3 各工作模式下的功能

Peripheral	Run	Sleep	LP Run		LP Sleep	Stop0 (MR 供电)		Stop1 (LPR 供电)		Stop2 (DLPR 供电)	
						工作	唤醒	工作	唤醒	工作	唤醒
CPU Core	Y	-	Y		-	-	-	-	-	-	-
Flash memory	Y	Y(2)	Y		Y(2)	-	-	-	-	-	-
SRAM	Y	Y(3)	Y		Y(3)	Y	-	Y	-	Y	-
BKP 寄存器	Y	Y	Y		Y	Y	-	Y	-	Y	-
PVD	O	O	O		O	O	O	O	O	O	O
DMA	O	O	O		O	-	-	-	-	-	-
HSI8/16/24/48M	O	O	O		O	O(4)	-	-	-	-	-
HSE	O	O	O		O	-	-	-	-	-	-
LSI	O	O	O		O	O	-	O	-	O	-
LSE	O	O	O		O	O	-	O	-	O	-
PLL	O	O	-		-	-	-	-	-	-	-
HSE Clock Security System (CSS)	O	O	O		O	-	-	-	-	-	-
RTC/Auto Wakeup	O	O	O		O	O	O	O	O	O	O
USART1	O	O	O		O	O	-	O	-	-	-
UARTx（x=1,2）	O	O	O		O	-	-	-	-	-	-
LPUART1	O	O	O		O	O(5)	O(5)	O(5)	O(5)	-	-
I2Cx(x=1,2)	O	O	O		O	O(6)	O(6)	O(6)	O(6)	-	-
SPIx(x=1,2)	O	O	O		O	-	-	-	-	-	-
V_{REFBUF}	O	O	O		O	-	-	-	-	-	-
OPAx(x=1,2)	O	O	O		O	-	-	-	-	-	-
COMPx(x=1,2)	O	O	O		O	O	O	O	O	-	-
Temperature sensor	O	O	O		O	-	-	-	-	-	-
Timers	O	O	O		O	-	-	-	-	-	-
PWMx（x=1,2,3）	O	O	O		O	-	-	-	-	-	-

LPTIM	O	O	O		O	O	O	O	O	-	-
IWDG	O	O	O		O	O	O	O	O	O	O
WWDG	O	O	O		O	-	-	-	-	-	-
SysTick	O	O	O		O	-	-	-	-	-	-
CRC	O	O	O		O	-	-	-	-	-	-
GPIOs	O	O	O		O	O	O	O	O	O	O

注：

1. Y = YES(ENABLE)；O= Optional(默认关闭，可以软件使能)； - = not available。
2. Flash 在 Sleep 和 Lower power sleep 模式下，可由软件选择时钟是否打开。
3. SRAM 时钟可被门控开或者关。
4. 在 Stop0 模式下，HSI 可由软件控制开关。
5. LPUART 在 Stop 模式下可以接收，产生 START 和地址匹配中断唤醒。
6. I^2C 在 Stop 模式下提供地址匹配和超时中断唤醒。

5.4.1 正常运行模式

在正常运行模式下，可以通过降低系统时钟（SYSCLK，HCLK，PCLK）来降低功耗；在进入 Sleep 模式前，也可以通过分频来降低外设时钟频率；

在正常运行模式下，不同的外设和存储器，都可以通过门控单独关闭时钟，达到降低功耗作用；在进入 Sleep 前，可以关闭不是用作唤醒的外设时钟。

5.4.2 低功耗运行模式

为了进一步降低运行模式下的功耗，可以关闭 MR，V_{DDD} 域由 LPR 供电。同时切换系统时钟为 HSI 8M，降低 HCLK 的工作频率，不能超过 2 MHz。

低功耗运行模式下的 IO 状态

在低功耗运行模式下，所有 IO 状态保持和运行状态一致。

低功耗运行模式的进入

1. 可选：程序跳转到 SRAM 运行，同时配置 Flash 进入低功耗模式
2. 降低系统时钟，要低于 2 MHz
3. 设置 V_{DDD} 使用 LPR 供电，MR 关闭

低功耗运行模式的退出

1. 设置 V_{DDD} 使用 MR 供电
2. 等待 MR 上电完成
3. 恢复正常的系统时钟

表 5-4 低功耗运行模式

低功耗运行	描述
模式进入	降低系统时钟低于 2 MHz LPR=1
模式退出	LPR=0 等待 MR 上电完成 恢复正常时钟
唤醒延时	MR 开启到稳定时间

5.4.3 低功耗模式进入和退出

进入：

MCU 通过执行 WFI（等待中断）或 WFE（等待事件）指令进入低功耗模式，也可通过在从 ISR 返回时将 Cortex®-M4 系统控制寄存器中的 SLEEPONEXIT 位置 1 进入低功耗模式。

仅当没有中断或事件挂起时，才可通过 WFI 或 WFE 进入低功耗模式，即 CPU 在进入低功耗前需要清零挂起中断或者事件。

在进入低功耗模式的进程中，出现唤醒信号，系统会根据当前状态判断如何处理：

1. 在系统时钟关闭之前，退出进入流程，处理唤醒中断或事件；
2. 在系统时钟关闭之后，先保持住唤醒信号，再正常进入低功耗模式，进入低功耗模式后，再按正常唤醒流程唤醒。

退出：

MCU 根据进入睡眠和停止模式的方式退出这两种低功耗模式：

- 如果使用 WFI 指令或从 ISR 返回进入低功耗模式，通过 NVIC 确认的任何外设中断都能唤醒器件。

- 如果使用 WFE 指令进入低功耗模式，MCU 将在有事件发生时立即退出低功耗模式。唤醒事件可通过以下方式产生：
  - NVIC IRQ 中断：
  - SEVONPEND = 0 时，通过使能 NVIC 和对应的外设的中断使能控制位，可以从 WFE 唤醒。当系统唤醒后，必须清除 NVIC 和外设中的中断挂起位；
  - SEVONPEND = 1 时，只用使能外设的中断使能控制位，NVIC 中的中断使能可以选择是否使能，就可以从 WFE 唤醒。当系统唤醒后，必须清除外设中的中断挂起位，如果 NVIC 中的中断使能，也要同时清除；
  - 所有的 NVIC 中断都可以唤醒 CPU，NVIC 中没有使能的也可以。
  - 事件：

    配置一个外部或内部的 EXTI 线为事件模式。当 CPU 从 WFE 恢复时，因为对应事件线的挂起位没有被置 1，不必清零 EXTI 外设的中断挂起位或 NVIC IRQ 通道挂起位。可能需要清零外设中的中断标志。该模式唤醒所需的时间最短，因为没有时间花费在中断的进入或退出上。

5.4.4 睡眠模式

进入睡眠模式之前，必须清零所有的中断挂起位。

睡眠模式下的 IO 状态

在睡眠模式下，所有 IO 状态保持和运行状态一致。

表 5-5 睡眠模式

睡眠模式	描述
模式进入	WFI 或 WFE，并且： — SLEEPDEEP=0 — 没有中断挂起
	从最低优先级的 ISR 返回，并且： — SLEEPDEEP=0 — SLEEPONEXIT=1 — 没有中断挂起
模式退出	使用 WFI 或从最低优先级 ISR 返回进入：中断（使能） 使用 WFE 进入且 SEVONPEN=0：唤醒事件 使用 WFE 进入且 SEVONPEN=1：外设使能中断（即使 NVIC 中禁止）
唤醒延时	无

5.4.5 低功耗睡眠模式

在低功耗睡眠模式下，所有 IO 状态保持和运行状态一致。

表 5-6 低功耗睡眠模式

低功耗睡眠	描述
模式进入	低功耗睡眠模式只能从低功耗运行模式进入 WFI 或 WFE，并且： — SLEEPDEEP=0 — 没有中断挂起
	低功耗睡眠模式只能从低功耗运行模式进入 从最低优先级的 ISR 返回，并且： — SLEEPDEEP=0 — 没有中断挂起

模式退出	使用 WFI 或从最低优先级 ISR 返回进入：中断（使能） 使用 WFE 进入且 SEVONPEN=0：唤醒事件 使用 WFE 进入且 SEVONPEN=1：外设使能中断（即使 NVIC 中禁止） 从低功耗睡眠模式退出，系统进入低功耗运行模式。
唤醒延时	无

5.4.6 停止模式

停止模式基于内核 SLEEPDEEP 模式与外设时钟门控。V_{DDD} 域 LDO 既可以配置为正常模式（MR），也可以配置为低功耗模式（LPR/DLPR）。在停止模式下，V_{DDD} 域的所有高频时钟都会停止，停止的时钟包括 PLL、HSI 和 HSE，停止顺序为 PLL->HSE->HSI，其中 PLL 和 HSE 由软件控制关闭，HSI 由硬件关闭。内部 SRAM 和寄存器内容将保留。在进入 Stop 模式前，需要软件将系统时钟切换为 HSI

停止模式下的 IO 状态

在停止模式下，所有 IO 状态保持和运行状态一致。

停止模式的进入

如果正在执行 Flash 编程，停止模式的进入将延迟到存储器访问结束后执行（通过软件保证）。

进入停止模式前，如果正在访问 APB 域，停止模式的进入将延迟到 APB 访问结束。（通过软件保证）

在停止模式下，可以通过对各控制位进行编程来选择以下功能：

- 独立的看门狗（IWDG）：IWDG 通过写入其密钥寄存器或使用硬件选项来启动。而且一旦启动便无法停止，除非复位。
- 实时时钟（RTC）：通过 RCC 备份域控制寄存器（RCC_BDCR）中的 RTCEN 位进行配置。
- 内部 RC 振荡器（LSI RC）：通过 RCC 控制/状态寄存器（RCC_CSR)中的 LSION 位进行配置。
- 外部振荡器（LSE OSC）：通过 RCC 备份域控制寄存器(RCC_BDCR) 中的 LSEON 位进行配置。

在 Stop 模式下，一些低功耗外设也可以配置为唤醒源，例如 LPUART 和 I2C；

在进入停止模式前，一些模拟模块（ADC，OPA）也可以工作，为了进一步降低功耗，进入前请关闭。

停止模式的退出

退出停止模式，将选择 HSI 作为系统时钟。如果配置为退出停止模式后，进入低功耗运行模式，需要在进入停止模式前，将系统时钟配置为低于 2 MHz；同时选择 V_{DDD} 的供电方式 MR（Stop0）或者 LPR（Stop1）或者 DLPR(Stop2)。

在进入 Stop 时，根据配置的 V_{DDD} 供电模式不同，启动后需要的延时不同。

表 5-7 停止模式

停止模式	描述
模式进入	WFI 或 WFE，并且： — 没有中断（针对 WFI）或者事件（针对 WFE）挂起 — SLEEPDEEP=1 — 将 PWR_CR1 寄存器中的 LPMS 配置为 00、01、10
	从最低优先级的 ISR 返回，并且： — SLEEPDEEP=1 — 没有中断挂起 — 将 PWR_CR1 寄存器中的 LPMS 配置为 00、01、10
模式退出	使用 WFI 或从 ISR 返回进入：任何配置为中断模式的 EXTI line（同时使能 NVIC 中对应的 EXTI 中断）。中断源为外部中断或者具有唤醒功能的外设产生的中断。 使用 WFE 进入且 SEVONPEND=0：任何配置为事件模式的 EXTI line。 使用 WFE 进入且 SEVONPEND=1：

	— 任何配置为中断模式的 EXTI line（即使在 NVIC 中断对应的 EXTI 中断禁止）。中断源为外部中断或者具有唤醒功能的外设产生的中断。 — 唤醒事件 NRST 复位 IWDG 复位
唤醒延时	HSI 唤醒时间+（LDO 从低功耗模式退出时间（LDO 不同模式时间不同））+Flash 唤醒的时间

5.4.7 停止模式的调试

在停止模式，默认情况下，调试功能将中断，因为 CPU 核无时钟。但如果配置了 DBGMCU_CR 中相关寄存器，则即使 CPU 进入 SLEEPDEEP 模式，仍可进行调试。

5.4.8 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 MCU(自动唤醒模式)。RTC 提供一个可编程的时间基数，用于周期性从停止模式下唤醒。通过对备份区域控制寄存器（RCC_BDCR）的 RTCSEL[1:0]位的编程，三个 RTC 时钟源中的二个时钟源可以选作实现此功能：

- 低功耗外部晶振（LSE)
  该时钟源提供了一个低功耗且精确的时间基准。
- 低功耗内部 RC 振荡器（LSI)
  使用该时钟源，节省了一个晶振的成本。但是 RC 振荡器将少许增加电源消耗。

该外设的寄存器可以通过 half-word 或者 word 访问。

5.5 PWR 寄存器

可以用半字(16 位)或字(32 位)的方式操作这些外设寄存器。

5.5.1 PWR 控制寄存器 1(PWR_CR1)

偏移地址： 0x00

复位值： 0x0001 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLS_WUPT		Res.
													RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	LPR	MR_VSEL		Res.	Res.	Res.	DBP	Res.	Res.	Res.	Res.	Res.	Res.	LPMS	
	RW	RW	RW				RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18:17	FLS_WUPT	RW	2’b0	停止模式唤醒时序中，在 HSI 稳定后，在 Flash 操作前需要等待时间。 00：3μs；01：5 μs；10：2 μs；11：0 μs
16:15	Reserved	-	-	保留
14	LPR	RW	0	VR 工作模式选择； 0：工作在主模式（MR） 1：工作在低功耗模式（LPR）
13:12	MR_VSEL	RW	2’b0	配置寄存器 MR_VSEL 选择工作模式 00：高电压（Range 1）模式 01：低电压（Range 3）模式

				10：中电压（Range 2）模式 11：高电压（Range 1）模式
11:9	Reserved	-	-	保留
8	DBP	RW	0	备份域寄存器写保护。 0：不可写 RTC 和备份寄存器 1：可以写 RTC 和备份寄存器
7:2	Reserved	-	-	保留
1:0	LPMS	RW	2’b0	低功耗模式选择 00：Stop0（MR 工作） 01：Stop1（LPR 工作） 10：Stop2 (DLPR 工作) 11：保留

5.5.2 PWR 控制寄存器 2(PWR_CR2)

偏移地址： 0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	FLT_CTRL			FLTEN	Res.	Res.	Res.	Res.	PLS			PVDE
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:9	FLT_CTRL	RW	3’b0	PVD 滤波时间配置。 111：保留 110：滤波时间大约为 1024 个滤波时钟（LSI 或者 LSE 时，大约为 30.7ms） 101：滤波时间大约为 128 个滤波时钟（LSI 或者 LSE 时，大约为 3.8ms） 100：滤波时间大约为 64 个滤波时钟（LSI 或者 LSE 时，大约为 1.92ms） 011：滤波时间大约为 16 个滤波时钟（LSI 或者 LSE 时，大约为 480μs） 010：滤波时间大约为 4 个滤波时钟（LSI 或者 LSE 时，大约为 120μs） 001：滤波时间大约为 2 个滤波时钟（LSI 或者 LSE 时，大约为 60μs） 000：滤波时间大约为 1 个滤波时钟（LSI 或者 LSE 时，大约为 30μs）
8	FLTEN	RW	0	PVD 数字滤波使能。 0：PVD 数字滤波禁止 1：PVD 数字滤波使能
7:4	Reserved	-	-	保留
3:1	PLS[2:0]	RW	3’b0	PVD 电平选择。

				由软件写入，用于选择 PVD 的电压阈值。 000：保留 001：VPVD1 (around 2.0 V) 010：VPVD2 (around 2.2 V) 011：VPVD3 (around 2.4 V) 100：VPVD4 (around 2.6 V) 101：VPVD5 (around 2. 8 V) 110：VPVD6 (around 3.0 V) 111：保留
0	PVDE	RW	0	PVD 使能 0：禁止 PVD 1：使能 PVD

5.5.3 PWR 状态寄存器(PWR_SR)

偏移地址： 0x10

复位值： 0x0000 0200

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

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	PVDO	R	0	PVD 输出标志。 此位通过硬件置 1 和清零。仅当 PVD 工作时 此位才有效。 0：V_{CC} 高于 PLS[2:0] 位选择的 PVD 阈值。 1：V_{CC} 低于 PLS[2:0] 位选择的 PVD 阈值。 注：复位后，此位等于 0。等到 PVD 使能后，根据检测电压阈值输出对应值。
15:10	Reserved	-	-	保留
9	MR_RDY	R	1	用于指示 MR 工作状态 0：MR 关闭 1：MR 工作
8:0	Reserved	-	-	保留

6.复位与时钟控制（RCC）

6.1 RCC 简介

本模块完成如下功能：

- 产生系统时钟和系统复位
- 产生各个模块时钟和复位
- 产生时钟源控制信号

6.2 RCC 复位

6.2.1 电源复位

只要发生以下事件之一，就会产生电源复位：

- 上电/掉电复位（POR/PDR 复位）
- 电源复位会将所有寄存器复位。

复位入口向量在存储器映射中固定在地址 0x0000_0004。

6.2.2 系统复位

除了时钟控制寄存器 CSR 中的复位标志和备份域中的寄存器外，系统复位会将其他全部寄存器都复位为复位值。

只要发生以下事件之一，就会产生系统复位：

- NRST 引脚低电平（外部复位）
- 窗口看门狗计数结束（WWDG 复位）
- 独立看门狗计数结束（IWDG 复位）
- 软件复位（SYSRESETREQ 复位）
- 低功耗管理复位（NRST_STOP）
- 选项字节加载复位

6.2.3 NRST 引脚（外部复位）

通过特定的选项位（NRST_MODE），NRST 引脚可配置为：

- 复位输入/输出（设备交付时的默认值）

复位输入/输出（设备交付时默认）引脚上的任何有效复位信号都会传播到设备内部逻辑，所有内部复位源都会通过脉冲发生器驱动到此引脚。GPIO 功能（PD11）不可用。脉冲发生器保证每个内部复位源在 NRST 引脚上输出的最小复位脉冲持续时间为 20μs。该模式在每次设备上电复位期间（直到加载选项字节）始终处于有效状态（独立于选项字节设置）。

- 复位输入

在此模式下，NRST 引脚上的任何有效复位信号都会传播到设备内部逻辑，设备内部产生的复位。在此配置中，GPIO 功能（PD11）不可用。

- GPIO

在此模式下，引脚可以用作 PD11 标准 GPIO。引脚的复位功能不可用。复位只能从设备内部复位源进行，并且不会传播到此引脚。

6.2.3.1 看门狗复位

参见 “独立看门狗（IWDG）和“系统窗口看门狗（WWDG）章节。

6.2.3.2 软件复位

可通过查看 RCC 时钟控制和状态寄存器（RCC_CSR）中的复位标志确定。要对器件进行软件复位，必须将 CPU 中断和复位控制寄存器中的 SYSRESETREQ 位置 1。

6.2.3.3 低功耗管理复位

引发低功耗管理复位的方式有一种：

- 进入停止模式时产生复位：

此复位的使能方式是清零用户选项字节中的 nRST_STOP 位。且 PWR_CR1.LPMS 寄存器配置为停止模式，进入低功耗模式时，器件将复位，而非进入停止模式。

6.2.3.4 选项字节加载复位

当 FLASH_CR 寄存器中设置 OBL_LAUNCH 位（位 27）时，会生成选项字节加载复位。此位用于通过软件启动选项字节加载。

6.2.4 备份域复位

备份域复位会将所有备份域寄存器复位，备份域寄存器包括 RCC_BDCR,备份寄存器以及 RTC 部分寄存器。

只要发生以下事件之一，就会产生备份域复位：

- 软件复位：通过将 RCC 备份域控制寄存器（RCC_BDCR）中的 BDRST 位置 1 触发。
- 在电源 V_{CC} 已掉电后，又再上电。

6.2.5 复位的统一处理

NRST 引脚复位输入/输出模式下，其他源复位都作用于 NRST 引脚，该引脚在复位过程中时钟保持低电平。

电路如下图所示：

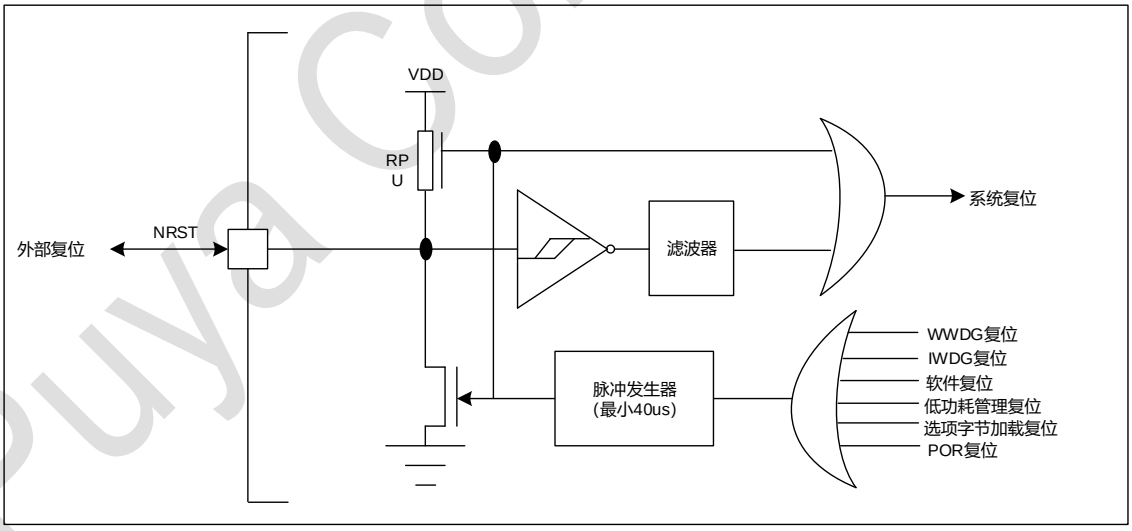


图 6-1 复位电路

当系统发生内部复位时（高电平有效），脉冲发生器开始产生脉冲信号，并保证该脉冲信号持续至少 40 μs。此脉冲信号使 N 管导通。N 管导通会不断拉低 NRST 引脚电压，当拉低到 VIL 时，NRST 引脚产生低电平信号。该信号经过施密特触发器和滤波后，产生系统复位（低电平有效）。

注 1：除外部 PIN 复位源之外的复位源，即脉冲发生器的源，不会经过 30 μs 的滤波后产生系统复位；

注 2：脉冲发生器的复位源，在产生系统复位后，除了本身复位标志置位外，外部引脚复位标志也会置位。

6.3 RCC 时钟

6.3.1 时钟源

系统存在如下时钟源：

6.3.1.1 HSE 时钟

外部高频 OSC. 频率范围 4 ~ 32 MHz

HSE 时钟的稳定时间由模拟 HSE 模块计数，模拟 HSE 模块在启动并且稳定后产生 HSE_RDY 信号。

HSE 时钟可以有两个工作模式：

- 外部 XTAL OSC+内部起振电路（HSEBYP=0）
- 通过 OSC_IN IO 输入的外部时钟（HSEBYP=1）

为保证 CLOCK 输出第一个时钟就为稳定时钟， HSE 内部包含一个计数器，计数完成后输出第一个时钟上升沿。HSEBYP=1 时计数周期是 HSEBYP=0 的一半。

6.3.1.2 HSI 时钟

内部 HSI RC 振荡器。相较于 HSE，功耗低，稳定时间短，但精度低。模拟 HSI 模块在启动并且稳定后产生 HSI_RDY 信号。

上电复位后，在加载阶段，HSI 的校准值加载到 RCC_CR. HSITRIM 寄存器。

HSI 上电后默认是 8 MHz，可以通过 RCC_CR.HSI_FS 寄存器选择 16 MHz，24 MHz 和 48 MHz。

从停止模式唤醒后，HSI 将作为系统时钟源。

6.3.1.3 PLL 时钟

PLL 模块是用来倍频 HSI 或者 HSE 时钟，PLL 输入时钟频率范围为 8 ~ 48 MHz。输出时钟频率范围为 48 ~ 144 MHz。

注意：如果 HSE 作为 PLL 参考时钟，开启顺序：先开启 HSE 等 HSERDY 变高后再开 PLL，关闭顺序：先关闭 PLL 等 PLLRDY 变低后再关闭 HSE。

6.3.1.4 LSE 时钟

外部 32.768 kHz OSC，用作低功耗时钟。

可以通过配置 LSEDRV 在稳定时间和功耗之间做平衡。LSE 稳定时间为 0.5 s（典型值）。LSE 稳定信号由 LSE 模拟模块产生。

与 HSE 来源类似，LSE 也有两个来源：

- 32.768 kHz XTAL+内部起振电路（LSEBYP=0）
- 通过 OSC32_IN 输入的外部时钟（LSEBYP=1）

为保证 CLOCK 输出第一个时钟就为稳定时钟， LSE 内部包含一个计数器， 计数完成后输出第一个时钟上升沿。LSEBYP=1 时计数周期是 LSEBYP=0 的一半。

在 LSEBYP=1 情况下，MCO 输出的时钟为原始外灌时钟，且不会受 LSE_ON 信号的控制。但使用 LSE 时钟的逻辑会受 LSE_ON 信号控制。

6.3.1.5 LSI 时钟

内部低频 40 kHz 时钟。用作 IWDG、RTC 时钟。

6.3.1.6 HSI10M 时钟

该时钟作为低精度时钟，用作 NRST pin 的滤波计数。每次产生系统复位，经过脉冲发生器后，都通过 NRST pin 返回，所以每次复位产生时都会使能 HSI10M 时钟。

6.3.2 时钟结构

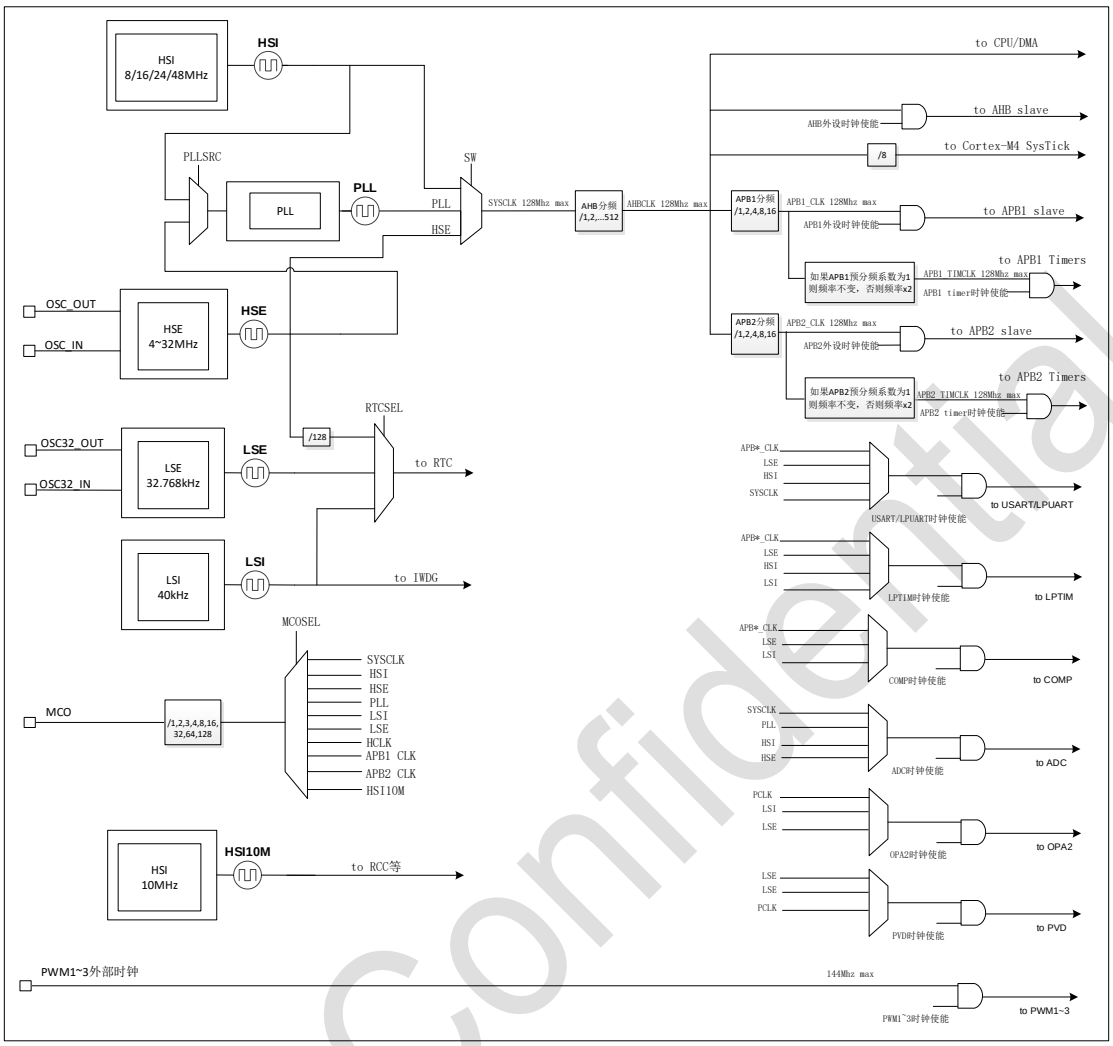


图 6-2 系统时钟结构图

6.3.3 时钟安全系统（CSS）

时钟安全系统可以通过软件激活。在这种情况下，时钟检测器在 HSE 振荡器启动延迟后启用，并在该振荡器停止时禁用。

如果在 HSE 时钟上检测到故障，HSE 振荡器将被自动禁用，时钟故障事件将发送到定时器（TIM1 和 TIM15/16/17）的中断输入，并生成一个中断以通知软件故障（时钟安全系统中断 CSSI），允许 MCU 执行救援操作。CSSI 通过 NMI（不可屏蔽中断）异常向量链接到 Cortex®-M4。

注：一旦启用 CSS，如果 HSE 时钟出现故障，则会发生 CSS 中断，并自动生成 NMI。除非 CSS 中断挂起位被清除，否则 NMI 将无限期执行。因此，在 NMI ISR 中，用户必须通过在时钟中断寄存器（RCC_CIR）中设置 CSSC 位来清除 CSS 中断。

如果 HSE 振荡器直接或间接用作系统时钟（间接表示：它用作 PLL 输入时钟，PLL 时钟用作系统时钟），则检测到的故障会导致系统时钟切换到 HSI 振荡器，并禁用 HSE 振荡器。如果 HSE 时钟是用作系统时钟的 PLL 的时钟源，则 PLL 也被禁用。

6.3.4 时钟输出

为了方便板级应用，节省 BOM 成本，以及 debug 等的需求，需要芯片提供时钟输出功能。即把下表的 MCO 信号（并分频）通过 GPIO 的复用功能实现时钟输出功能。

表 6-1 输出时钟选择

时钟源/内部时钟	MCO 可输出的时钟源
HSI	√
HSE	√
PLL	√
LSE	√
LSI	√
HSI_10M	√
SYSCLK	√
HCLK	√
PCLK	√

注意：当对 MCO 时钟源进行切换，以及选择 GPIO AF 功能为 MCO 的起始阶段，MCO 可能会产生毛刺，需要避开该段时间。

6.3.5 外设时钟使能寄存器

每个外围时钟可以由 RCC_AHBxENR、RCC_APBxENRy 寄存器的 xxxxEN 位启用。

当外围时钟未激活时，不支持外围寄存器的读或写访问。

使能位具有同步机制，在使能位被设置之后，在时钟被激活之前有 2 个时钟周期的延迟。

注意：在为外围设备启用时钟后，软件必须等待一段时间后才能访问外围寄存器。

6.4 RCC 寄存器

该模块寄存器可以按字节（8 位）[RCC_BDCR 除外]、半字（16 位）或字（32 位）访问。

6.4.1 RCC 时钟控制寄存器（RCC_CR）

偏移地址： 0x00

复位值： 0x0000 4203

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	HSIFS[1:0]		PLLRDY	PLLON	Res.	Res.	Res.	Res.	CSSON	HSEBYP	HSERDY	HSEON
				RW	RW	R	RW					RW	RW	R	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	HSITRIM[10:0]											Res.	HSIKERON	HSIRDY	HSION
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:26	HSIFS[1:0]	RW	2'h0	HSI 频率选择： 00：8 MHz 01：16 MHz 10：24 MHz 11：48 MHz
25	PLLRDY	R	0	PLL 时钟 ready 标志。

				硬件置位，表明 PLL 时钟稳定。 0：PLL 没有 ready 1：PLL ready
24	PLLON	RW	0	PLL 使能。 当 PLL 时钟被用作（SWS 为 PLL）或将要被用作系统时钟（SW 选择 PLL）时，该位不能被清零。 0：PLL OFF 1：PLL ON 注意：PLL 使能前需先开启参考时钟，PLL 关闭前不能关闭参考时钟。
23:20	Reserved	-	-	保留
19	CSSON	RW	0	HSE 时钟安全系统使能。 当该位为 1 时，如果 HSE OSC ready 则硬件会使能时钟检测模块；如果 HSE 检测失败，则关闭时钟检测模块。(该位只能写 1，系统复位清零) 0：时钟安全系统关闭（时钟检测关闭） 1：时钟安全系统开启（如果 HSE 时钟稳定则时钟检测开启，否则关闭时钟检测）
18	HSEBYP	RW	0	HSE 屏蔽晶振，选择管脚输入时钟。 该位只有当 HSEON=0 时才能写. 0：HSE 晶振不屏蔽，外部高速时钟选择晶振 1：HSE 晶振屏蔽，外部高速时钟选择外部管脚输入时钟源
17	HSERDY	R	0	HSE 晶振时钟 ready 标志。 该位由硬件置位表明 HSE 晶振稳定。 0：HSE 晶振没有 ready 1：HSE 晶振 ready
16	HSEON	RW	0	当 HSE 直接或者间接作为系统时钟源时，该位不能被置 0. 0：HSE 晶振 OFF 1：HSE 晶振 ON
15	Reserved	-	-	保留
14:4	HSITRIM[10:0]	RW	11’h420	HSI 时钟频率校准值。 上电后硬件用 HSI 8MHz 的默认校准值，在上电加载 Trimming 时会把出厂信息写入该寄存器中。 软件通过读出存放在 information 区相应地址的数据，写入该寄存器，实现 HSI 特定输出频率下的校准。 保存在 Flash 的如下地址内： 8 MHz 校准值存放地址：0x1FFF 1200 16 MHz 校准值存放地址：0x1FFF 1204 24 MHz 校准值存放地址：0x1FFF 1208 48 MHz 校准值存放地址：0x1FFF 120C 系统复位会复位该寄存器。
3	Reserved	-	-	保留
2	HSIKERON	RW	0	HSI 振荡器在停止模式时钟使能。

				HSI 作为内核时钟配置提供给外设 USART 和 I2C 等。保持 HSI 在停止模式下打开可避免由于 HSI 启动时间而降低通信速度。此位对 HSION 值没有影响。 0：对 HSI 振荡器没有影响。 1：HSI 振荡器即使在停止模式下也被强制使能。
1	HSIRDY	R	1	HSI 时钟 ready 标志。 硬件置位表明 HSI OSC 稳定。该位只有当 HSION=1 时才有效。 0：HSI OSC 没有 ready 1：HSI OSC ready 当 HSION 清零后，HSIRDY 将在 6 个 HSI 时钟之后拉低。
0	HSION	RW	1	HSI 时钟使能。 硬件在进入停止模式时，会根据需要清零该寄存器停止 HSI. 当 HSI 直接或者间接作为系统时钟时，该寄存器不能为 0. 0：HSI OSC OFF 1：HSI OSC ON 硬件在如下情况会使能 HSI： 1） 硬件从停止模式唤醒后 2） HSE 直接/间接作为系统时钟但出现 HSE CSS fail

注意：HSIFS、HSITRIM 需要同时配置。

6.4.2 RCC 时钟配置寄存器（RCC_CFGR)

偏移地址： 0x04

复位值： 0x0000 0000

当时钟源切换时，访问该寄存器有 1 或者 2 个时钟的等待周期。

当 APB 或者 AHB 分频值更新时，访问该寄存器可能有 0~15 个时钟的等待周期。

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

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	MCO [3:0]	RW	4’h0	MCO 输出时钟选择。 0000：MCO 不输出 0001：LSE 0010：LSI 0100：SYSCLK 0101：HSI 0110：HSE 0111：PLL 1000：HCLK 1001：APB1 时钟 1010：APB2 时钟 1011：HSI10M

				其它：MCO 不输出 注 1：MCO 输出选择系统时钟时，需要保证输出时频率不超过 IO 的最大允许频率。
23:14	Reserved	-	-	保留
13:11	PPRE2[2:0]	RW	3'h0	高速 APB（APB2）时钟分频系数。 PCLK 基于 HCLK 的分频系数。 0xx：不分频 100：2 分频 101：4 分频 110：8 分频 111：16 分频
10:8	PPRE1[2:0]	RW	3'h0	高速 APB（APB1）时钟分频系数。 PCLK 基于 HCLK 的分频系数。 0xx：不分频 100：2 分频 101：4 分频 110：8 分频 111：16 分频
7:4	HPRE[3:0]	RW	4'h0	AHB 时钟 HCLK 基于 SYSCLK 的分频系数。 0xxx：不分频 1000：2 分频 1001：4 分频 1010：8 分频 1011：16 分频 1100：64 分频 1101：128 分频 1110：256 分频 1111：512 分频 为了保证系统正常工作，需要根据 VR 电源情况配置合适频率。 注：建议逐级切换分频系数。
3:2	SWS[1:0]	R	2'h0	系统时钟源选择。 该位由硬件控制，表明系统时钟源的选择。 00：HSI 01：HSE 10：PLL CLK 其它：保留
1:0	SW[1:0]	RW	2'h0	系统时钟源选择。 该位由硬件和软件控制，表明系统时钟源的选择。 00：HSI 01：HSE 10：PLL CLK 其它：保留 硬件配置为 HSI 的情况：

				1.MCU 从停止模式退出 2.软件配置为 01(HSE)，但 HSE 检测失败 3.软件配置为 10，并且 PLL 源选择 HSE，但 HSE 检测失败

6.4.3 RCC 时钟中断寄存器（RCC_CIR）

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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSSC	Res.	Res.	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
								W			W	W	W	W	W
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE	CSSF	Res.	Res.	PLLRDYF	HSERDYF	HSIRDYIF	LSERDYF	LSIRDYF
			RW	RW	RW	RW	RW	R			R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	CSSC	W	0	HSE 时钟安全系统（CSS）中断标志清零。 0：没有影响 1：清零 CSSF 标志
22:21	Reserved	-	-	保留
20	PLLRDYC	W	0	PLL ready 中断标志清零。 0：没有影响 1：清零 PLLRDYF 标志
19	HSERDYC	W	0	HSEready 中断标志清零。 0：没有影响 1：清零 HSERDYF 标志
18	HSIRDYC	W	0	HSI ready 中断标志清零。 0：没有影响 1：清零 HSIRDYF 标志
17	LSERDYC	W	0	LSE ready 中断标志清零。 0：没有影响 1：清零 LSERDYF 标志
16	LSIRDYC	W	0	LSI ready 中断标志清零。 0：没有影响 1：清零 LSIRDYF 标志
15:13	Reserved	-	-	保留
12	PLLRDYIE	RW	0	PLL ready 中断使能。 0：禁止 1：使能
11	HSERDYIE	RW	0	HSE 时钟 ready 中断使能。 0：禁止 1：使能
10	HSIRDYIE	RW	0	HSI 时钟 ready 中断使能。 0：禁止

				1：使能
9	LSERDYIE	RW	0	LSE 时钟 ready 中断使能。 0：禁止 1：使能
8	LSIRDYIE	RW	0	LSI 时钟 ready 中断使能。 0：禁止 1：使能
7	CSSF	R	0	时钟安全系统中断标志。 当 HSE 时钟出现故障时，由硬件置 1。 0：HSE 时钟安全系统中断未产生 1：HSE 时钟安全系统中断产生 写 CSSC 寄存器 1 清零该位。
6:5	Reserved	-	-	保留
4	PLLRDYF	R	0	PLL 时钟 ready 中断标志。 PLL 时钟稳定并且 PLLRDYDIE=1 时，硬件置位该寄存器。 0：PLL 时钟 ready 中断未产生 1：PLL 时钟 ready 中断产生 写 PLLRDYC 寄存器 1 清零该位。
3	HSERDYF	R	0	HSE 时钟 ready 中断标志。 当 HSE 时钟稳定并且 HSERDYIE=1 时，硬件置位该寄存器。 0：HSE 时钟 ready 中断未产生 1：HSE 时钟 ready 中断产生 写 HSERDYC 寄存器 1 清零该位。
2	HSIRDYF	R	0	HSI 时钟 ready 中断标志。 HSI 时钟稳定并且 HSIRDYIE=1 时，硬件置位该寄存器。 0：HSI 时钟 ready 中断未产生 1：HSI 时钟 ready 中断产生 写 HSIRDYC 寄存器 1 清零该位。
1	LSERDYF	R	0	LSERDY 时钟 ready 中断标志。 LSE 时钟稳定并且 LSERDYDIE=1 时，硬件置位该寄存器。 0：LSERDY 时钟 ready 中断未产生 1：LSERDY 时钟 ready 中断产生 写 LSERDYC 寄存器 1 清零该位。
0	LSIRDYF	R	0	LSIRDY 时钟 ready 中断标志。 LSI 时钟稳定并且 LSIRDYIE=1 时，硬件置位该寄存器。 0：LSIRDY 时钟 ready 中断未产生 1：LSIRDY 时钟 ready 中断产生 写 LSIRDYC 寄存器 1 清零该位。

6.4.4 RCC AHB 外设复位寄存器（RCC_AHBRSTR）

偏移地址：0x28

复位值： 0x0000 0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	IOPDRST	IOPCRST	IOPBRST	IOPARST	Res.	CRCRST	Res.	Res.	Res.	Res.	Res.	DMA1RST
				RW	RW	RW	RW		RW						RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	IOPDRST	RW	0	IOPD 模块复位。 0：没有影响 1：该模块复位
10	IOPCRST	RW	0	IOPC 模块复位。 0：没有影响 1：该模块复位
9	IOPBRST	RW	0	IOPB 模块复位。 0：没有影响 1：该模块复位
8	IOPARST	RW	0	IOPA 模块复位。 0：没有影响 1：该模块复位
7	Reserved	-	-	保留
6	CRCRST	RW	0	CRC 模块复位。 0：没有影响 1：该模块复位
5:1	Reserved	-	-	保留
0	DMA1RST	RW	0	DMA1 模块复位。 0：没有影响 1：该模块复位

6.4.5 RCC APB1 外设复位寄存器 1（RCC_APB1RSTR1）

偏移地址：0x38

复位值： 0x0000 0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	PWRRST	Res.	Res.	Res.	Res.	Res.	I2C2RST	I2C1RST	UART2RST	UART1RST	Res.	Res.	Res.
			RW						RW	RW	RW	RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	SPI2RST	PWM2RST	PWM1RST	WWDGRST	Res.	Res.	Res.	Res.	Res.	TIM7RST	TIM6RST	Res.	TIM4RST	TIM3RST	TIM2RST
	RW	RW	RW	RW						RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	PWRRST	RW	0	Power 接口模块复位。 0：没有影响 1：该模块复位
27:23	Reserved	-	-	保留
22	I2C2RST	RW	0	I2C2 模块复位。 0：没有影响 1：该模块复位
21	I2C1RST	RW	0	I2C1 模块复位。 0：没有影响 1：该模块复位
20	UART2RST	RW	0	UART2 模块复位。 0：没有影响 1：该模块复位
19	UART1RST	RW	0	UART1 模块复位。 0：没有影响 1：该模块复位
18:15	Reserved	-	-	保留
14	SPI2RST	RW	0	SPI2 模块复位。 0：没有影响 1：该模块复位
13	PWM2RST	RW	0	PWM2 模块复位。 0：没有影响 1：该模块复位
12	PWM1RST	RW	0	PWM1 模块复位。 0：没有影响 1：该模块复位
11	WWDGRST	RW	0	WWDG 模块复位。 0：没有影响 1：该模块复位
10:6	Reserved	-	-	保留
5	TIM7RST	RW	0	TIM7 模块复位。 0：没有影响; 1：该模块复位;
4	TIM6RST	RW	0	TIM6 模块复位。 0：没有影响 1：该模块复位
3	Reserved	-	-	保留
2	TIM4RST	RW	0	TIM4 模块复位。 0：没有影响 1：该模块复位
1	TIM3RST	RW	0	TIM3 模块复位。 0：没有影响

				1：该模块复位
0	TIM2RST	RW	0	TIM2 模块复位。 0：没有影响 1：该模块复位

6.4.6 RCC APB1 外设复位寄存器 2（RCC_APB1RSTR2）

偏移地址：0x3C

复位值： 0x0000_0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPUART1RST	LPTIM1RST
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	LPUART1RST	RW	0	LPUART1 模块复位。 0：没有影响 1：该模块复位
0	LPTIM1RST	RW	0	LPTIM1 模块复位。 0：没有影响 1：该模块复位

6.4.7 RCC APB2 外设复位寄存器（RCC_APB2RSTR）

偏移地址： 0x40

复位值： 0x0000_0000

该寄存器由软件置位和清零，软件置位后，该模块维持复位直到软件清零。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	OPARST	COMPRST	Res.	Res.	TIM17RST	TIM16RST	TIM15RST	Res.	Res.	Res.	Res.	PWM4RST	PWM3RST
			RW	RW			RW	RW	RW					RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	USART1RST	Res.	SPI1RST	TIM1RST	Res.	ADC1RST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGRST
	RW		RW	RW		RW									RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	OPARST	RW	0	OPA 模块复位。 0：没有影响 1：该模块复位
27	COMPRST	RW	0	COMP 模块复位。 0：没有影响 1：该模块复位
26:25	Reserved	-	-	保留
24	TIM17RST	RW	0	TIM17 模块复位。

				0：没有影响 1：该模块复位
23	TIM16RST	RW	0	TIM16 模块复位。 0：没有影响 1：该模块复位
22	TIM15RST	RW	0	TIM15 模块复位。 0：没有影响 1：该模块复位
21:18	Reserved	-	-	保留
17	PWM4RST	RW	0	PWM4 模块复位。 0：没有影响 1：该模块复位
16	PWM3RST	RW	0	PWM3 模块复位。 0：没有影响 1：该模块复位
15	Reserved	-	-	保留
14	USART1RST	RW	0	USART1 模块复位。 0：没有影响 1：该模块复位
13	Reserved	-	-	保留
12	SPI1RST	RW	0	SPI1 模块复位。 0：没有影响 1：该模块复位
11	TIM1RST	RW	0	TIM1 模块复位。 0：没有影响 1：该模块复位
10	Reserved	-	-	保留
9	ADC1RST	RW	0	ADC1 模块复位。 0：没有影响 1：该模块复位
8:1	Reserved	-	-	保留
0	SYSCFGRST	RW	0	SYSCFG 模块复位。 0：没有影响 1：该模块复位

6.4.8 RCC AHB 外设时钟使能寄存器（RCC_AHBENR）

偏移地址： 0x48

复位值： 0x0000_0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	IOPDEN	IOPCEN	IOPBEN	IOPAEN	Res.	CRCEN	Res.	FMCEN	Res.	SRAMEN	Res.	DMA1EN
				RW	RW	RW	RW		RW		RW		RW		RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	IOPDEN	RW	0	IOPD 模块时钟使能。 0：禁止 1：使能
10	IOPCEN	RW	0	IOPC 模块时钟使能。 0：禁止 1：使能
9	IOPBEN	RW	0	IOPB 模块时钟使能。 0：禁止 1：使能
8	IOPAEN	RW	0	IOPA 模块时钟使能。 0：禁止 1：使能
7	Reserved	-	-	保留
6	CRCEN	RW	0	CRC 模块时钟使能。 0：禁止 1：使能
5	Reserved	-	-	保留
4	FMCEN	RW	1	Flash 接口模块时钟使能，针对 Sleep 模式。 0：禁止 1：使能
3	Reserved	-	-	保留
2	SRAMEN	RW	1	SRAM 存储区时钟使能，针对 Sleep 模式。 0：禁止 1：使能
1	Reserved	-	-	保留
0	DMA1EN	RW	0	DMA1 模块时钟使能。 0：禁止 1：使能

6.4.9 RCC APB1 外设时钟使能寄存器 1（RCC_APB1ENR1）

偏移地址： 0x58

复位值： 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	PWREN	Res.	Res.	Res.	Res.	Res.	I2C2EN	I2C1EN	UART2EN	UART1EN	Res.	Res.	Res.
			RW						RW	RW	RW	RW			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	SPI2EN	PWM2EN	PWM1EN	WWDGEN	Res.	Res.	Res.	Res.	Res.	TIM7EN	TIM6EN	Res.	TIM4EN	TIM3EN	TIM2EN
	RW	RW	RW	RW						RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	PWREN	RW	0	PWR 模块时钟使能。 0：禁止 1：使能

27:23	Reserved	-	-	保留
22	I2C2EN	RW	0	I2C2 模块时钟使能。 0：禁止 1：使能
21	I2C1EN	RW	0	I2C1 模块时钟使能。 0：禁止 1：使能
20	UART2EN	RW	0	UART2 模块时钟使能。 0：禁止 1：使能
19	UART1EN	RW	0	UART1 模块时钟使能。 0：禁止 1：使能
18:15	Reserved	-	-	保留
14	SPI2EN	RW	0	SPI2 模块时钟使能。 0：禁止 1：使能
13	PWM2EN	RW	0	PWM2 模块时钟使能。 0：禁止 1：使能
12	PWM1EN	RW	0	PWM1 模块时钟使能。 0：禁止 1：使能
11	WWDGEN	RW	0	WWDG 模块时钟使能。 0：禁止 1：使能
10:6	Reserved	-	-	保留
5	TIM7EN	RW	0	TIM7 模块时钟使能。 0：禁止 1：使能
4	TIM6EN	RW	0	TIM6 模块时钟使能。 0：禁止 1：使能
3	Reserved	-	-	保留
2	TIM4EN	RW	0	TIM4 模块时钟使能。 0：禁止 1：使能
1	TIM3EN	RW	0	TIM3 模块时钟使能。 0：禁止 1：使能
0	TIM2EN	RW	0	TIM2 模块时钟使能。 0：禁止 1：使能

6.4.10 RCC APB1 外设时钟使能寄存器 2（RCC_APB1ENR2）

偏移地址： 0x5C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPUART1EN	LPTIM1EN
														RW	RW

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	LPUART1EN	RW	0	LPUART1 模块时钟使能。 0：禁止 1：使能
0	LPTIM1EN	RW	0	LPTIM1 模块时钟使能。 0：禁止 1：使能

6.4.11 RCC APB2 外设时钟使能寄存器（RCC_APB2ENR）

偏移地址： 0x60

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	OPAEN	COMPEN	Res.	Res.	TIM17EN	TIM16EN	TIM15EN	Res.	Res.	Res.	Res.	PWM4EN	PWM3EN
			RW	RW			RW	RW	RW					RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART1EN	Res.	SPI1EN	TIM1EN	Res.	ADC1EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYSCFGEN
	RW		RW	RW		RW									RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28	OPAEN	RW	0	OPA 模块时钟使能。 0：禁止 1：使能
27	COMPEN	RW	0	COMP 模块时钟使能。 0：禁止 1：使能
26:25	Reserved	-	-	保留
24	TIM17EN	RW	0	TIM17 模块时钟使能。 0：禁止 1：使能
23	TIM16EN	RW	0	TIM16 模块时钟使能。 0：禁止 1：使能
22	TIM15EN	RW	0	TIM15 模块时钟使能。

				0：禁止 1：使能
21:18	Reserved	-	-	保留
17	PWM4EN	RW	0	PWM4 模块时钟使能。 0：禁止 1：使能
16	PWM3EN	RW	0	PWM3 模块时钟使能。 0：禁止 1：使能
15	Reserved	-	-	保留
14	USART1EN	RW	0	USART1 模块时钟使能。 0：禁止 1：使能
13	Reserved	-	-	保留
12	SPI1EN	RW	0	SPI1 模块时钟使能。 0：禁止 1：使能
11	TIM1EN	RW	0	TIM1 模块时钟使能。 0：禁止 1：使能
10	Reserved	-	-	保留
9	ADC1EN	RW	0	ADC1 模块时钟使能。 0：禁止 1：使能
8:1	Reserved	-	-	保留
0	SYSCFGEN	RW	0	SYSCFG 模块时钟使能。 0：禁止 1：使能

6.4.12 RCC 外设独立时钟配置寄存器（RCC_CCIPR)

偏移地址： 0x68

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPA2SEL[1:0]		Res.	Res.	Res.	Res.	ADC1SEL[1:0]		Res.	Res.	Res.	Res.	COMP2SEL[1:0]		COMP1SEL[1:0]	
RW	RW					RW	RW					RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LPTIM1SEL[1:0]		LPUART1SEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	USART1SEL[1:0]	
RW	RW	RW	RW											RW	RW

Bit	Name	R/W	Reset Value	Function
31:30	OPA2SEL[1:0]	RW	2’h0	OPA2(作为 COMP 使用时)时钟源选择。 00：PCLK 01：LSI 10：LSE 11：保留

				注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
29:26	Reserved	-	-	保留
25:24	ADC1SEL[1:0]	RW	2'h0	ADC1 时钟源选择。 00：SYSCLK 01：PLL 10：HSI 11：HSE 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
23:20	Reserved	-	-	保留
19:18	COMP2SEL[1:0]	RW	2'h0	COMP2 时钟源选择。 00：PCLK 01：LSI 10：LSE 11：保留 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
17:16	COMP1SEL[1:0]	RW	2'h0	COMP1 时钟源选择。 00：PCLK 01：LSI 10：LSE 11：保留 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
15:14	LPTIM1SEL[1:0]	RW	2'h0	LPTIM1 时钟源选择。 00：PCLK 01：LSI 10：HSI 11：LSE 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
13:12	LPUART1SEL[1:0]	RW	2'h0	LPUART1 时钟源选择。 00：PCLK 01：SYSCLK 10：HSI 11：LSE 注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。
11:2	Reserved	-	-	保留
1:0	USART1SEL[1:0]	RW	2'h0	USART1 时钟源选择。 00：PCLK 01：SYSCLK 10：HSI 11：LSE

				注：切换时钟源前，需要先关闭时钟使能，切换完成后再开启时钟使能。

6.4.13 RCC RTC 域控制寄存器（RCC_BDCR）

偏移地址： 0x70

复位值： 0x0000 0000

该寄存器中 LSEON、LSEBYP、LSEDRV、RTCSEL、RTCEN 位在 V_{CC} 域中，仅 V_{CC} 域复位能被复位，复位包括 V_{CC} 域上电复位(POR)和备份域软复位(BDRST)。

当连续访问该寄存器时，0 ≤ wait state ≤ 3。

该寄存器不属于 Vcore 域，而属于 RTC 域（V_{CC} 域）。在复位后，该寄存器需要处于写保护状态，即需要设置 PWR_CR1.DBP 位为 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDRST
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTCEN	Res.	Res.	Res.	Res.	Res.	RTCSEL[1:0]		Res.	Res.	Res.	LSEDRV[1:0]		LSEBYP	LSERDY	LSEON
RW						RW	RW				RW	RW	RW	R	RW

Bit	Name	R/W	Reset Value	Function
31:17	Reserved	-	-	保留
16	BDRST	RW	0	备份域软复位。 0：没有影响 1：复位
15	RTCEN	RW	0	RTC，TAMP 和 BKPREG 时钟使能。 0：禁止 1：使能
14:10	Reserved	-	-	保留
9:8	RTCSEL[1:0]	RW	0	RTC 时钟源选择。 00: No clock 01: LSE 10: LSI 11: HSE /128 一旦 RTC 时钟源选择后不能再改变，除非备份域复位被清零。
7：5	Reserved	-	-	保留
4:3	LSEDRV[1:0]	RW	0	LSE 驱动能力设置 00：gm 2.5 uA/V 01：gm 3.75 uA/V 10：gm 8.5 uA/V 11：gm 13.5 uA/V
2	LSEBYP	RW	0	LSE OSC bypass 0:低速外部时钟选择晶振； 1:低速外部时钟选择外部接口输入时钟； 注：只有当外部 32.768 kHz OSC 禁止（LSEON=0 并且 LSERDY=0）时才能写该位。
1	LSERDY	R	0	LSE OSC ready. 硬件配置该位为 1 表明 LSE 时钟 ready。

				在 LSEON 清零后，该位需要 6 个 LSE 时钟后再清零。
0	LSEON	RW	0	LSE OSC 使能。 0：禁止 1：使能

6.4.14 RCC 控制/状态寄存器（RCC_CSR）

偏移地址： 0x74

复位值： 0x0C00 0000

该寄存器中复位标志位只能由 power reset 复位，其他由 system reset 复位。

当连续访问该寄存器时，0 ≤ wait state ≤ 3。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PWRRSTF	PINRSTF	OBLRSTF	RMVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSIRDY	LSION
														R	RW

Bit	Name	R/W	Reset Value	Function
31	LPWRRSTF	R	0	低功耗复位标志。 当进入停止低功耗模式时，硬件置位该寄存器。 只有当 nRST_STOP, nRST_STDBY 选项字节为清零（有效）状态时才能操作该寄存器。 RMVF 置 1 会清零该位。
30	WWDGRSTF	R	0	Window WDG 复位标志。 RMVF 置 1 会清零该位。
29	IWDGRSTF	R	0	IWDG 复位标志。 RMVF 置 1 会清零该位。
28	SFTRSTF	R	0	软复位标志。 RMVF 置 1 会清零该位。
27	PWRRSTF	R	1	POR/PDR 复位标志。 RMVF 置 1 会清零该位。
26	PINRSTF	R	1	外部 NRST 管脚复位标志。 RMVF 置 1 会清零该位。
25	OBLRSTF	R	0	选项字节 loader 复位标志。 RMVF 置 1 会清零该位。
24	RMVF	RW	0	软件置位清零复位标志。 写该位为 1 的操作会清零 bit25 开始的复位标志，软件写 1 后该位会维持为 1，硬件不会清零。
23:2	Reserved	-	-	保留
1	LSIRDY	R	0	LSI OSC 稳定标志。 0：LSI OSC 没有 ready 1：LSI OSC ready
0	LSION	RW	0	LSI OSC 使能。

				0：禁止 1：使能 硬件开启模拟 LSI 的情况： 1） 硬件 IWDG 使能

6.4.15 RCC 时钟配置寄存器 1（RCC_CFGR1）

偏移地址： 0x78

复位值： 0x0000 0000

1	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	PVDSEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	RW	RW													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCOPRE[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:29	PVDSEL[1:0]	RW	2’h0	PVD 滤波时钟选择。 00：PVD 滤波时钟选择 LSI 01：PVD 滤波时钟选择 LSE 10：PVD 滤波时钟选择 PWR PCLK 11：保留 注：在配置 PWR_CR2.PVDE=1 使能 PVD 功能之前需要先配置该寄存器选择 PVD 滤波时钟，并在整个过程中时钟选择不会变化。
28:4	Reserved	-	-	保留
3:0	MCOPRE[3:0]	RW	4’h0	MCO（microcontroller clock output）分频系数。 软件控制这些位，设置 MCO 输出的分频系数： 0000：1 0001：2 0010：4 0011：8 0100：16 0101：32 0110：64 0111：128 1xxx：3 在 MCO 输出使能前，设置这些位。

6.4.16 RCC 时钟配置寄存器 2（RCC_CFGR2）

偏移地址： 0x7C

复位值： 0x0000 0000

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

						RW	RW								

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	HSE_DRV[1:0]	RW	2'h0	HSE 驱动能力配置 00：4 ~ 8 MHz 01：8 ~ 16 MHz 10：16 ~ 32 MHz 11：保留
7:0	Reserved	-	-	保留

6.4.17 RCC 时钟配置寄存器 3（RCC_CFGR3）

偏移地址： 0x80

复位值： 0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:8	PLLMUL[2:0]	RW	3'b0	PLL 倍频系数 000：x2 001：x3 010：x4 011：x5 100：x6 101：x7 110：x8 111：x9
7:1	Reserved	-	-	保留
0	PLLSRC	RW	0	PLL 时钟源选择。 只能在关闭 PLL 时才能写入此位。 0：HSI 1：HSE

7. 通用 IO（GPIO）

7.1 GPIO 简介

每个通用 IO 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)、一个 32 位复位寄存器(GPIOx_BRR)

1 个 32 位锁定寄存器 (GPIOx_LCKR)

2 个 32 位复用功能选择寄存器（GPIOx_AFRH 和 GPIOx_AFRL）。

7.2 GPIO 主要特征

- 受控 IO 最多达 60 个
- 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）输出到引脚
- 可为每个 IO 选择不同的速度
- 输入状态：浮空、上拉/下拉、模拟
- 将引脚输入到数据寄存器 (GPIOx_IDR) 或外设（复用功能输入）
- 置位/复位寄存器 (GPIOx_BSRR)，对 GPIOx_ODR 具有按位写权限
- 复位寄存器 (GPIOx_BRR)，对 GPIOx_ODR 具有按位写权限
- 锁定寄存器 (GPIOx_LCKR)，可冻结 IO 配置
- 复用功能输入/输出选择寄存器（一个 IO 最多可具有 16 个复用功能）
- 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 IO 引脚用作 GPIO 或多种外设功能中的一种

7.3 GPIO 功能描述

根据数据手册中列出的每个 IO 端口的特性，可通过软件将通用 IO (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
- 模拟功能
- 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 IO 端口位均可自由编程，但 IO 端口寄存器必须按字、半字或字节进行访问。GPIOx_BSRR 和 GPIOx_BRR 寄存器允许对任何 GPIO 寄存器的读/更改的独立访问。这样，在读和更改访问之间产生 IRQ 时不会发生危险。

图 7-1 和图 7-2 显示了 5 V 兼容和基本 IO 端口位的基本结构。表 7-1 给出了可能的端口位配置方案。

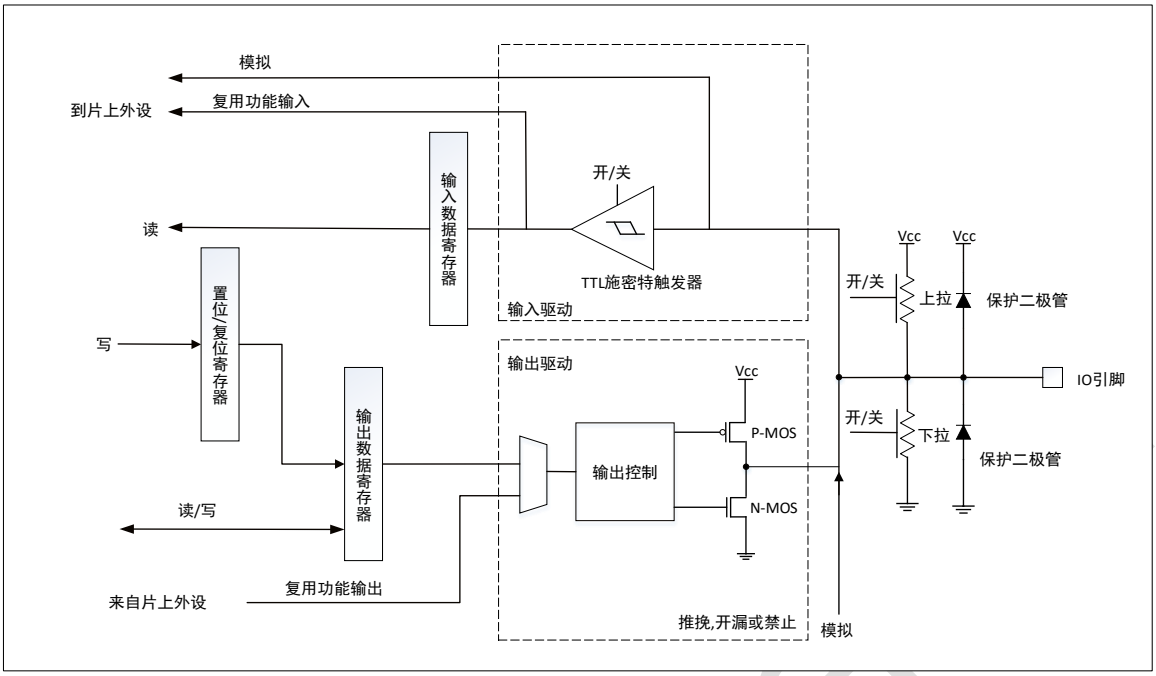


图 7-1 IO 端口基本结构

表 7-1 端口位配置表

MODE(i)[1:0]	OTYPE(i)	OSPEED(i)[1:0]		PUPD(i)[1:0]		IO configuration	
01	0	SPEED[1:0]		0	0	GP output	PP
	0			0	1	GP output	PP+PU
	0			1	0	GP output	PP+PD
	0			1	1	Reserved	
	1			0	0	GP output	OD
	1			0	1	GP output	OD+PU
	1			1	0	GP output	OD+PD
	1			1	1	Reserved(GP output 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	Reserved	
	1			0	0	AF	OD
	1			0	1	AF	OD+PU
	1			1	0	AF	OD+PD
	1			1	1	Reserved	
00	x	x	x	0	0	Input	Floating
	x	x	x	0	1	Input	PU
	x	x	x	1	0	Input	PD
	x	x	x	1	1	Reserved(input floating)	
11	x	x	x	0	0	Input/Output	Analog
	x	x	x	0	1	Reserved	
	x	x	x	1	0	Input/Output	Analog,PD
	x	x	x	1	1	Reserved	

GP = general-purpose, PP = push-pull, PU = pull-up, PD = pull-down, OD = open-drain, AF = alternate function.

7.3.1 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，大部分 IO 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，BOOT0 引脚处于输入模式下拉状态：

- PA15：JTDI 处于上拉状态
- PA14：JTCK/SWCLK 处于下拉状态
- PA13：JTMS/SWDAT 处于上拉状态
- PB4：NJTRST 处于上拉状态
- PD3：BOOT0 处于下拉状态

输入数据寄存器（GPIOx_IDR）每隔 1 个 AHB 时钟周期捕获一次 IO 引脚的数据。

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

7.3.2 IO 引脚复用器和映射

IO 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 IO 引脚。这可以确保共用同一个 IO 引脚的外设之间不会发生冲突。

每个 IO 引脚都有一个复用器，该复用器采用 16 路复用功能输入（AF0 到 AF15），可通过 GPIOx_AFRL（用于引脚 0 到 7）和 GPIOx_AFRH（用于引脚 8 到 15）寄存器对这些输入进行配置。

要将 IO 配置成所需功能，请按照以下步骤操作：

调试功能

系统复位后，调试相关的引脚复用为调试功能，供系统调试接口使用。

GPIO

在 GPIOx_MODER 寄存器中将所需 IO 配置为输出或输入。

外设复用功能

- 在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中，设置 IO 的复用功能
- 通过 GPIOx_OTYPER、 GPIOx_PUPDR 和 GPIOx_OSPEEDR 寄存器，分别选择输出类型、上拉/下拉以及输出速度
- 在 GPIOx_MODER 寄存器中将所需 IO 配置为复用功能

额外功能

- 对于 ADC，OPA 和 COMP，需要通过寄存器 GPIOx_MODER 配置相应的 IO 为模拟模式，并在 ADC，OPA 和 COMP 中配置相应功能；需要特别注意，对于模拟附加功能(如 OPA)的输出，需要先配置相应 IO 为模拟模式，再使能外设的控制寄存器。
- 对于额外功能 TAMP_RTC，WKUPx，OSC 和 OSC32，需要在 RTC，PWR 和 RCC 等模块配置相应功能，这些功能优先于 GPIO 设定。

7.3.3 IO 端口控制寄存器

每个 GPIO 有 4 个 32 位控制寄存器（GPIOx_MODER、GPIOx_OTYPER、GPIOx_OSPEEDR、GPIOx_PUPDR），可配置多达 16 个 IO。GPIOx_MODER 寄存器用于选择 IO 工作模式（输入、输出、复用、模拟）。GPIOx_OTYPER 用于选择输出类型（推挽或开漏）。GPIOx_OSPEEDR 寄存器用于设定 IO 速度。GPIOx_PUPDR 寄存器用于选择上拉/下拉。

7.3.4 IO 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。IO 输入数据存储到 GPIOx_IDR 中，它是一个只读寄存器。

7.3.5 IO 数据位操作

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

GPIOx_ODR 中的每个数据位对应于 GPIOx_BSRR 中的两个控制位：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)位会清零 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清零操作，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。

复位寄存器（GPIOx_BRR）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复位操作。复位寄存器的作用与置位复位寄存器相似，但仅能复位输出数据寄存器的一个或多个位。

7.3.6 GPIO 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，必须用特定的写/读序列，软件以正确的锁定序列操作此寄存器的 LCKR[16]后，会使用 LCKR[15:0]的值来锁定 IO 的配置（在写序列期间，LCKR[15:0]的值必须保持不变）。某个或多个端口被锁定后，无法解锁，直到发生系统复位，或 GPIO 被复位。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。

锁定序列只能通过字访问对 GPIOx_LCKR 寄存器操作。

具体操作参考 GPIOx_LCKR 寄存器描述。

7.3.7 IO 复用功能输入/输出

有两个寄存器可用来从每个 IO 可用的 16 个复用功能输入/输出中进行选择。

不同的复用信号有不同的输入/输出方向，具体情况由各自的外设模块决定。

7.3.8 外部中断线/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，请参见中断和事件章节。

7.3.9 输入配置

当 IO 口配置为输入：

- 输出缓冲器不使能
- 施密特触发器输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输入数据寄存器可得知引脚电平

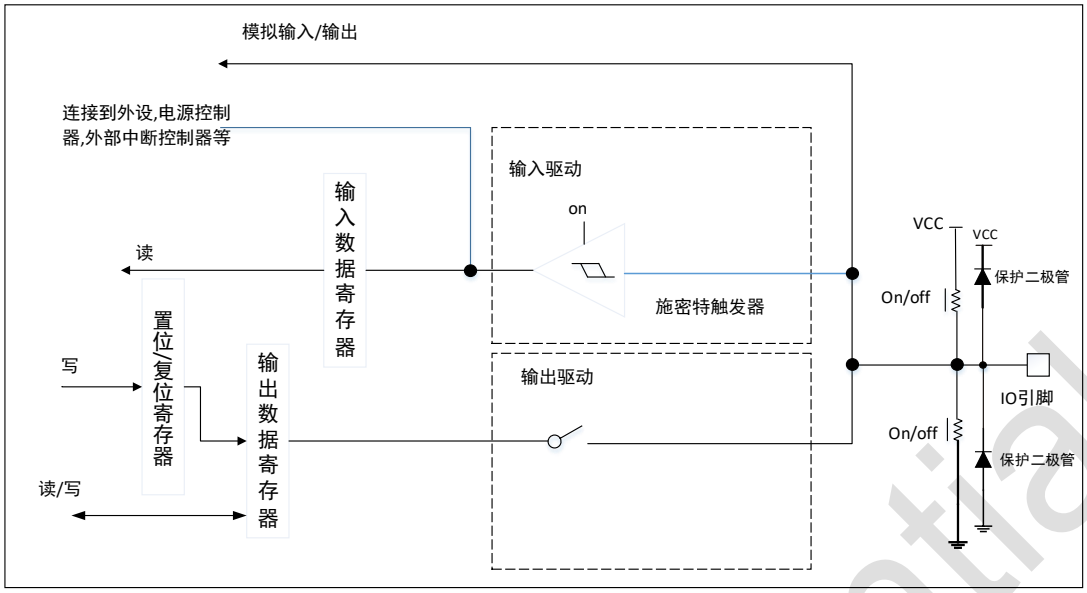


图 7-2 输入浮空/上拉/下拉配置

7.3.10 输出配置

当 IO 端口被配置为输出时：

- 输出缓冲器使能
  - 开漏模式：输出寄存器上的’0’导通 N-MOS，而输出寄存器上的’1’将端口置于高阻状态(PMOS 不导通)。
  - 推挽模式：输出寄存器上的’0’导通 N-MOS，而输出寄存器上的’1’将导通 P-MOS。
- 施密特触发输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输入数据寄存器可得知引脚电平
- 读取输出数据寄存器可得知上一次写 GPIO 的值

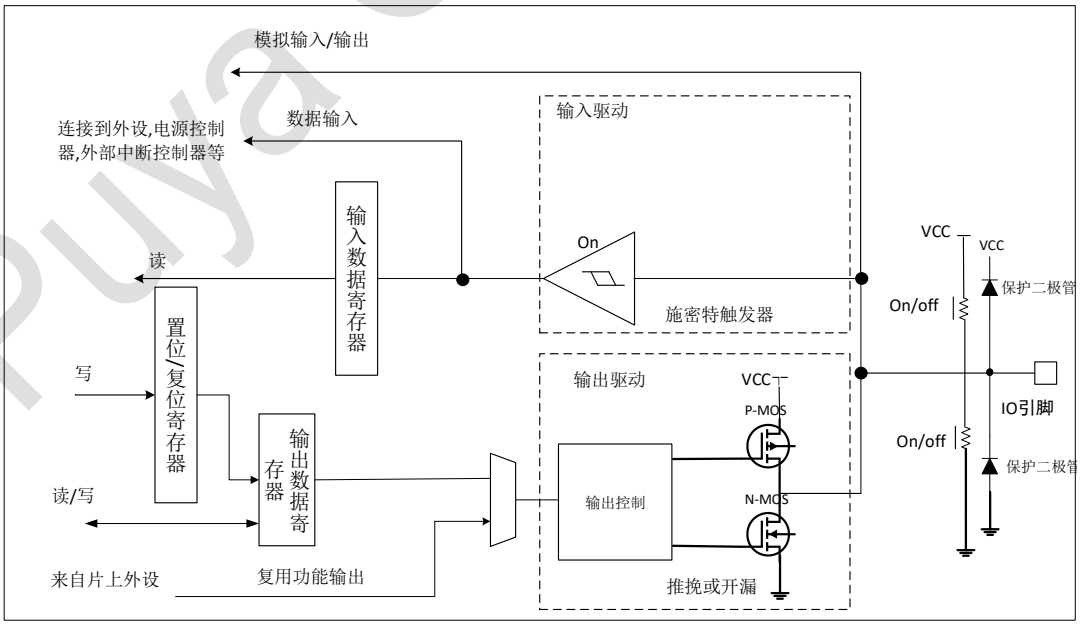


图 7-3 输出配置

7.3.11 复用功能配置

当 IO 端口被配置为复用功能时：

- 输出缓冲器使能
- 输出缓冲器的输出数据和输出使能由外设模块驱动
- 施密特触发输入使能
- 根据寄存器 GPIOx_PUPDR 配置可使能/不使能上下拉电阻
- 每个 AHB 时钟采样引脚电平，并存放于输入数据寄存器
- 读取输出数据寄存器可得知上一次写 GPIO 的值

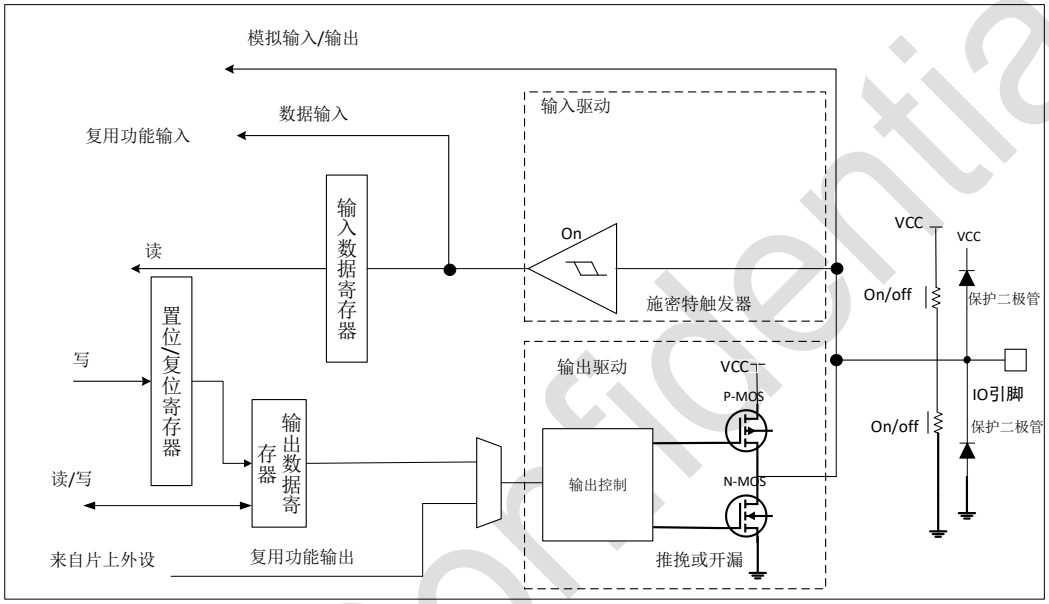


图 7-4 复用功能配置

7.3.12 模拟配置

当 IO 端口被配置为模拟模式时：

- 输出缓冲器不使能；
- 施密特触发器输入不使能，实现了每个模拟 IO 引脚上的零功耗。施密特触发输出值被强置为’0’；
- 弱上拉和下拉电阻被禁止（需要软件设定）；
- 读取输入数据寄存器时数值为’0’。

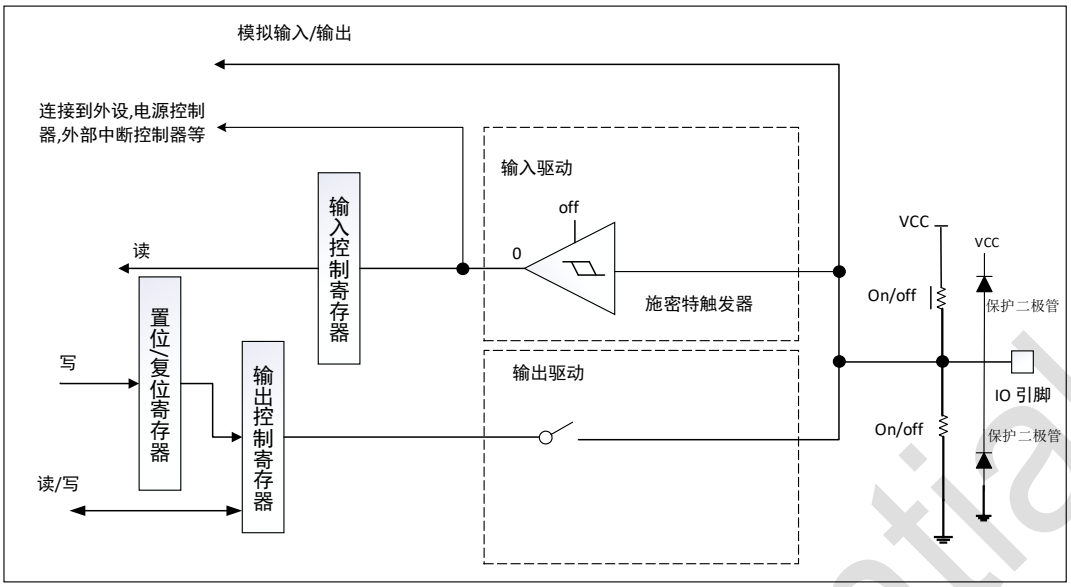


图 7-5 高阻抗模拟配置

7.3.13 使用 HSE 或 LSE 振荡器引脚作为 GPIO

当 HSE 或 LSE 功能不使能（复位后的默认），相应的管脚可以当作正常的 GPIO。

当 HSE 或 LSE 功能使能（RCC 寄存器中置位 HSEON 或 LSEON），相关引脚的功能由振荡器控制，不受这些 GPIO 寄存器的设定影响。

当晶振配置为用户外部时钟模式，只有 OSC_IN 或 OSC32_IN 用于接收外部时钟输入，而 OSC_OUT 或 OSC32_OUT 脚仍然可以用作正常 GPIO。

7.3.14 使用 PD3 作为 GPIO

选项字节中的 nSWBOOT0 位选择：

- nSWBOOT0=1，复位中为 BOOT0 引脚。
- nSWBOOT0=0，复位中为 GPIO 引脚。

注意事项:

在复位后，PD3 为 GPIO 引脚，和 nSWBOOT0 无关。

7.3.15 使用 PD11 作为 GPIO

PD11 可以用作复位引脚（NRST）或 GPIO。根据用户选项字节中的 NRST_MODE 位，它切换到以下模式：

- 复位输入/输出： NRST_MODE=2’b11(默认值)或 NRST_MODE=2’b00
- 仅复位输入： NRST_MODE=2’b01
- GPIO PD11 模式: NRST_MODE=2’b10

7.4 寄存器描述

该章节详细描述 GPIO 的寄存器功能

有关寄存器位、寄存器偏移地址和复位值的汇总，请参见 GPIO 寄存器映射表；

可通过字节（8 位）、半字（16 位）或字（32 位）对 GPIO 寄存器进行访问。

GPIOA 基地址: 0x4002 4000

GPIOB 基地址: 0x4002 4400

GPIOC 基地址: 0x4002 4800

GPIOD 基地址: 0x4002 4C00

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

偏移地址:0x00

复位值: 0xABFF_FFFF（端口 A）、0xFFFF_FEBF（端口 B）、0xFFFF FFFF（端口 C）
0xFFFF FF3F（端口 D）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:0	MODEy[1:0]	RW	PA13 ~ PA15, PB3 ~ PB4：2'b10 PD3：2'b00 其他：2'b11	y = 15..0 软件通过这些位配置相应的 IO 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)

7.4.2 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Res.			
15:0	OTy	RW	0	y = 15..0 软件配置 IO 的输出类型 0: 推挽输出 (复位状态) 1: 开漏输出

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

偏移地址:0x08

复位值: 0x6400 0000（端口 A）、0x0000 0100（端口 B）、0x0000 0000（端口 C）、
0x0000 0080（端口 D）

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	Name	R/W	Reset Value	Function
31:0	OSPEEDy[1:0]	RW	PA13:2'b11 其他:2'b00	y = 15..0 软件配置 IO 口的输出速度 00：非常低 01：低速 10：高速 11：非常高

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

偏移地址:0x0C

复位值: 0x6400 0000（端口 A）、0x0000 0100（端口 B）、0x0000 0000（端口 C）、0x0000 0080（端口 D）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPD15[1:0]		PUPD14[1:0]		PUPD13[1:0]		PUPD12[1:0]		PUPD11[1:0]		PUPD10[1:0]		PUPD9[1:0]		PUPD8[1:0]	
RW	RW	RW	RW	RW	RW	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	Name	R/W	Reset Value	Function
31:0	PUPDy [1:0]	RW	PA13,PA15,PB4：2'b01 PA14,PD3：2'b10 其他：2'b00	y = 15..0 软件配置 IO 口上拉或者下拉 00：无上下拉 01：上拉 10：下拉 11：保留

7.4.5 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Res.			
15:0	IDy	R		y = 15..0 这是只读的，读出值位对应 IO 口的状态

7.4.6 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Res.			
15:0	ODy	RW	0	y = 15..0 对 GPIOx_BSRR 或 GPIOx_BRR 寄存器 (x=A,B,C,D,E,F)，可以分别对各个 ODR 位进行独立的设置/清除。

7.4.7 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	Name	R/W	Reset Value	Function
31:16	BRy	W	0	y = 15..0 读返回值是 0 0：对应的 ODRy 位不受影响 1：清除对应的 ODRy 位 注：如果同时写 BSy 和 BRy 的对应位，BSy 位起作用
15: 0	BSy	W	0	y = 15..0 读返回值是 0 0：对应的 ODRy 位不受影响 1：设置对应的 ODRy 位

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

当执行正确的写序列置位 LCKK 时，该寄存器用来锁定端口位的配置。LCKR[15:0]用于锁定 GPIO 端口的配置。在规定的写入操作期间，不能改变 LCKR[15:0]。对相应的端口执行锁定序列后，在下次系统复位前将不能再更改端口位的配置。

注：特殊写时序用来写 GPIOx_LCKR 寄存器。在锁定时序中只允许字访问。

每个锁定位冻结一种特定的配置寄存器（控制和复用功能寄存器）

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LCKK
															RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:17	Res.	-	-	保留
16	LCKK	RW	**0**	该位可随时读出，它只能通过锁键写入序列修改 0: 端口配置锁键位未激活 1: 端口配置锁键位被激活，下次系统复位前 GPIOx_LCKR 寄存器被锁定 锁键值写序列: 写 1->写 0->写 1->读 0->读 1，最后一个读可省略，但可以用来确认锁键已被激活。 注：在操作锁键的写入序列时，不能改变 LCK[15:0]的值。锁键时序的任何错误都会终止锁键被激活。对端口的任何一位首次锁键时序之后，读 LCKK 位都是返回 1,直到系统复位或者 GPIO 复位。
15: 0	LCKy	RW	**0**	y = 15..0 只能在 LCKK 位为 0 时写入。 0: 不锁定端口的配置 1: 锁定端口配置

7.4.9 GPIO 复用功能低寄存器（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	Name	R/W	Reset Value	Function
31:0	AFSELy[3:0]	RW	**4'h0**	y = 7..0 配置复用功能 IO AFSELy 选择: 0000: AF0　1000: AF8 0001: AF1　1001: AF9 0010: AF2　1010: AF10 0011: AF3　1011: AF11 0100: AF4　1100: AF12 0101: AF5　1101: AF13 0110: AF6　1110: AF14 0111: AF7　1111: AF15

7.4.10 GPIO 复用功能高寄存器（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	Name	R/W	Reset Value	Function
31:0	AFSELy[3:0]	RW	**4'h0**	y = 15..8 配置复用功能 IO AFSELy 选择: 0000: AF0 1000: AF8 0001: AF1 1001: AF9 0010: AF2 1010: AF10 0011: AF3 1011: AF11 0100: AF4 1100: AF12 0101: AF5 1101: AF13 0110: AF6 1110: AF14 0111: AF7 1111: AF15

7.4.11 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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**2**	**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	Name	R/W	Reset Value	Function
31:16	Res.			
15:0	BRy	W	**0**	y = 15..0 读返回值是 0 0: 对应的 ODy 位受影响 1: 清除对应的 ODy 位

8.外设互联

8.1 简介

多个外设间有直接连接。

这可以在外设间实现自动通信和/或同步，节省了 CPU 资源，进而降低功耗。

此外，这些硬件连接消除了软件延迟，允许设计可预测系统。

这些互连可以在运行、睡眠、低功耗运行、低功耗睡眠、停止模式下工作，具体取决于外设。

8.2 互联详情

8.2.1 定时器输入触发(ITR)

目的

一些 TIMx 定时器在内部连接在一起，用于定时器的同步或者连接。当一个定时器为主模式时，它可以重置、启动、停止从模式定时器。

触发信号

在可配置定时器事件发生后，输出（来自主设备）出现在 TIMx_TRGO 信号上。

对于没有触发输出的 TIM16 和 TIM17 定时器，将使用输出比较 1 代替。

输入（到从设备）位于 TIMx_ITR0/ITR1/ITR2/ITR3/ITR4/ITR5/ITR6 信号上。

表 8-1 定时器输入触发

TIMx 内部触发	定时器输入触发源（ITR）				
	TIM1	TIM2	TIM3	TIM4	TIM15
timx_itr0	-	tim1_trgo	tim1_trgo	tim1_trgo	tim1_trgo
timx_itr1	tim2_trgo	-	tim2_trgo	tim2_trgo	tim2_trgo
timx_itr2	tim3_trgo	tim3_trgo	-	tim3_trgo	tim3_trgo
timx_itr3	tim4_trgo	tim4_trgo	tim4_trgo	-	tim4_trgo
timx_itr4	tim15_trgo	tim15_trgo	tim15_trgo	tim15_trgo	-
timx_itr5	tim16_oc1	tim16_oc1	tim16_oc1	tim16_oc1	tim16_oc1
timx_itr6	tim17_oc1	tim17_oc1	tim17_oc1	tim17_oc1	tim17_oc1

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.2 定时器外部触发(ETR)

目的

ADC 模拟看门狗信号输出、比较器的输出，以及定时器 ETR 引脚可以连接到定时器的外部触发 ETR。

触发信号

ADC 模拟看门狗输出 ADC_AWD、比较器的输出 COMP1_OUT,COMP2_OUT.

表 8-2 定时器外部触发

TIMx 外部触发	定时器外部触发源（ETR）				
	TIM1	TIM2	TIM3	TIM4	TIM15
timx_etr0	TIM1_ETR	TIM2_ETR	TIM3_ETR	TIM4_ETR	TIM15_ETR
timx_etr1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out

timx_etr2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out
timx_etr3	adc_awd1	TIM3_ETR	TIM2_ETR	TIM2_ETR	adc_awd1
timx_etr4	adc_awd2	TIM4_ETR	TIM4_ETR	TIM3_ETR	adc_awd2
timx_etr5	adc_awd3	-	-	-	adc_awd3

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.3 清除定时器 OCxREF 信号

目的

比较器的输出，可以用于清除高级定时器 TIM1、通用定时器 TIM2、TIM3 的 OCxREF 信号。

触发信号

比较器的输出 COMP1_OUT,COMP2_OUT.

表 8-3 清除定时器 OCxREF 信号

Timx OCREF CLR	清除定时器 OCxREF（OCxREF_CLR）源			
	TIM1	TIM2	TIM3	TIM4
timx_ocref_clr0	comp1_out	comp1_out	comp1_out	comp1_out
timx_ocref_clr1	comp2_out	comp2_out	comp2_out	comp2_out

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.4 定时器输入捕获

目的

比较器的输出，可以用作高级定时器 TIM1，通用定时器 TIM2/TIM3/TIM15 的输入捕获通道。

通用定时器 TIM15/TIM16 可以使用输入捕获通道 1 实现时钟测量。

表 8-4 输入捕获通道 1(TI1)源

TIMx TI1	输入捕获通道 1(TI1)源						
	TIM1	TIM2	TIM3	TIM4	TIM15	TIM16	TIM17
TI1_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM4_CH1	TIM5_CH1	TIM16_CH1	TIM17_CH1
TI1_IN1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	MCO	MCO
TI1_IN2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	HSE_Div32	HSE_Div32
TI1_IN3	-	-	-	-	-	-	-
TI1_IN4	-	-	-	-	-	LSI	LSI

表 8-5 输入捕获通道 2(TI2)源

TIMx TI2	输入捕获通道 2(TI2)源				
	TIM1	TIM2	TIM3	TIM4	TIM15
TI2_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM4_CH1	TIM5_CH1
TI2_IN1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out
TI2_IN2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out

表 8-6 输入捕获通道 3(TI3)源

TIMx TI3	输入捕获通道 3(TI3)源			
	TIM1	TIM2	TIM3	TIM4
TI3_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM4_CH1
TI3_IN1	comp1_out	comp1_out	comp1_out	comp1_out
TI3_IN2	comp2_out	comp2_out	comp2_out	comp2_out

表 8-7 输入捕获通道 4(TI4)源

TIMx TI4	输入捕获通道 4(TI4)源			
	TIM1	TIM2	TIM3	TIM4
TI3_IN0	TIM1_CH1	TIM2_CH1	TIM3_CH1	TIM4_CH1
TI3_IN1	comp1_out	comp1_out	comp1_out	comp1_out
TI3_IN2	comp2_out	comp2_out	comp2_out	comp2_out

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.5 定时器刹车

目的

比较器的输出，可以用作高级定时器 TIM1，通用定时器 TIM15/TIM16/TIM17，以及 PWM 的刹车。

触发信号

比较器的输出 COMP1_OUT、COMP2_OUT，以及刹车引脚输入 TIMx_BRK。

表 8-8 定时器刹车输入

TIMx/PWM 刹车	定时器刹车输入						
	TIM1	TIM15	TIM16	TIM17	PWM1	PWM2	PWM3
BRK0	TIM1_BRK	TIM15_BRK	TIM16_BRK	TIM17_BRK	PWM1_BRK	PWM2_BRK	PWM3_BRK
BRK1	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out	comp1_out
BRK2	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out	comp2_out

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.6 红外输出波形控制（IRTIM）

目的

TIM16 或 TIM17 定时器的 TIMx_OC1 输出通道可生成红外输出波形。

表 8-9 IRTIM 控制信号分配

IRTIM 控制信号	IRTIM 控制信号分配
调制包络信号	Tim16_oc1
载波信号	Tim17_oc1

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.7 比较器消隐源

目的

高级控制定时器 TIM1 和通用定时器 TIM2、TIM3、TIM15 和 PWM 可用作 COMP1 和 COMP2 的消隐窗口输入。

触发信号

定时器输出信号 TIMx_OCx/PWM_OC3 是 COMP1/COMP2 的消隐源输入。

表 8-10 比较器消隐输入

COMPx_CSR.BLANKSEL[2:0]	比较器消隐输入	
	COMP1	COMP2
1	tim1_oc5	tim1_oc5
2	tim2_oc3	tim2_oc3
3	tim3_oc3	tim3_oc3
4	tim4_oc3	tim4_oc3
5	tim15_oc1	tim15_oc1

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.8 系统错误刹车

目的

HSE 和 LSE CSS、CPU LOCKUP、SRAM 奇偶校验错误，PVD 报警可面向 TIM1、TIM15、TIM16、TIM17 和 PWM 模块以定时器刹车形式生成系统错误。

刹车功能的目的是保护由这些定时器生成的 PWM 信号所驱动的功率器件。

表 8-11 系统错误刹车

系统错误源	系统错误触发 TIM			
	TIM1	TIM15	TIM16	TIM17
PVD output	tim1_bk	tim15_bk	tim16_bk	tim17_bk
Cortex®-M4 Lockup (HardFault)				
Clock security system (CSS)				

相关功耗模式

这些互连在运行、睡眠、低功耗运行和低功耗睡眠功耗模式下工作。

8.2.9 ADC 硬件触发输入

目的

通用定时器 TIM2、TIM3 和 TIM15、TIM16、基本定时器 TIM6 和 TIM7，高级定时器 TIM1 ，LPTIM 和 EXTI 可用于生成 ADC 触发事件。

触发信号

输出（来自定时器）位于 TIMx_TRGO、或 TIMx_OCx 信号事件上，以及 EXTI Line11、EXTI Line15 事件。

输入（到 ADC）位于 EXT[15:0] 和 JEXT[15:0] 信号上，选择分别由寄存器 ADC_CFGR.EXTSEL[3:0]和 ADC_JSQR.JEXTSEL[3:0]选择。

表 8-12 ADC 硬件触发源

EXTSEL[4:0] or JEXTSEL[4:0]	ADC 硬件触发源	
	规则触发	注入触发
0	tim1_cc1	tim1_trgo
1	tim1_cc2	tim1_cc4
2	tim1_cc3	tim2_trgo
3	tim2_cc2	tim2_cc1
4	tim3_trgo	tim3_cc4
5	tim4_cc4	tim4_trgo
6	exti11	exti15
7	tim1_trgo	tim3_cc3
8	tim2_trgo	tim3_trgo
9	tim4_trgo	tim3_cc1
10	tim6_trgo	tim6_trgo
11	tim15_trgo	tim15_trgo
12	tim3_cc4	tim2_cc3
13	tim7_trgo	tim16_cc1
14	lptim_out	tim7_trgo
15	-	lptim_out

9.系统配置控制器（SYSCFG）

9.1 SYSCFG 主要特性

SYSCFG 模块主要完成如下功能：

- I²C IO 噪声滤波器控制
- I²C Fm+模式控制
- 所有 IO 噪声滤波器控制
- EXTI IO 选择
- 具有 COMP 模拟功能的引脚的模拟通道 2 控制
- 根据不同 boot 模式，映射初始程序区
- DMA 外设通道选择
- 定时器以及 PWM 的刹车输入控制

9.2 SYSCFG 寄存器

9.2.1 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）

偏移地址：0x00

复位值：0x0000 000x(x 取决于 BOOT0 引脚以及选项字节里启动模式配置)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	I2C2_FLTEN	I2C1_FLTEN	I2C2_FMP	I2C1_FMP	MEM_MODE[1:0]	
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	I2C2_FLTEN	RW	0	I2C2 滤波使能 0：I2C2 滤波关闭 1：I2C2 滤波开启
4	I2C1_FLTEN	RW	0	I2C1 滤波使能 0：I2C1 滤波关闭 1：I2C1 滤波开启
3	I2C2_FMP	RW	0	I2C2 Fm+模式驱动能力(Fast-mode Plus driving capability)激活 0: I2C2 被 AF 通道所选择的引脚的 Fm+驱动模式不使能 1: I2C2 被 AF 通道所选择的引脚的 Fm+驱动模式使能 适用 IO: PA1,PA3,PA7,PA8,PA11,PA12, PB10,PB11,PB13,PB14, PC0,PC1,PC9, PD6,PD7
2	I2C1_FMP	RW	0	I2C1 Fm+模式驱动能力(Fast-mode Plus driving capability)激活 0: I2C1 被 AF 通道所选择的引脚的 Fm+驱动模式不使能

				1: I2C1 被 AF 通道所选择的引脚的 Fm+驱动模式使能 适用 IO: PA4,PA9,PA10,PA13, PB4,PB6,PB7,PB8,PB9, PC1,PC6,PC7,PC9, PD4,PD5
1:0	MEM_MODE[1:0]	RW	X	启动模式选择位 控制存储器的 0x0000 0000 地址的映射。在复位后，这些位采用实际实际启动模式配置值。 x0：Main flash，映射到 0x0000 0000 01：System flash，映射到 0x0000 0000 11：SRAM，映射到 0x0000 0000

9.2.2 SYSCFG 配置寄存器 2 (SYSCFG_CFGR2)

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	PVDL	RS	0	PVD 锁定使能位 软件置位，系统复位清零。它可以被用作使能和锁定 PVD 连接给 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入，也锁定 PWR_CR2 寄存器的 PVDE。 软件写 0 到该位不会改变该位的值 0：PVD 中断不与 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入连接。PVDE 位可以被软件写入。 1：PVD 中断与 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入连接。PVDE 位只读。
1	Res.			
0	CLL	RS	0	Cortex-M4 LOCKUP 使能位 软件置位，系统复位清零。它可以使能和锁定 Cortex-M4 的 LOCKUP(hardfault)输出给 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入。 软件写 0 到该位不会改变该位的值 0：Cortex-M4 的 LOCKUP 输出不与 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入连接； 1：Cortex-M4 的 LOCKUP 输出与 TIM1/15/16/17 以及 PWM1/2/3 的刹车输入连接；

9.2.3 SYSCFG 配置寄存器 3（SYSCFG_CFGR3）

偏移地址：0x8

复位值：0x7F7F 7F7F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	DMA1_MAP4[6:0]							Res.	DMA1_MAP3[6:0]						

	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**
Res.	DMA1_MAP2[6:0]							Res.	DMA1_MAP1[6:0]						
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:24	DMA1_MAP4[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
23	Reserved	-	-	保留
22:16	DMA1_MAP3[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
15	Reserved	-	-	保留
14:8	DMA1_MAP2[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
7	Reserved	-	-	保留
6:0	DMA1_MAP1[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节

9.2.4 SYSCFG 配置寄存器 4（SYSCFG_CFGR4）

偏移地址：0xC

复位值：0x7F7F 7F7F

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	DMA1_MAP8[6:0]							Res.	DMA1_MAP7[6:0]						
	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**
Res.	DMA1_MAP6[6:0]							Res.	DMA1_MAP5[6:0]						
	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:24	DMA1_MAP8[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
23	Reserved	-	-	保留
22:16	DMA1_MAP7[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
15	Reserved	-	-	保留
14:8	DMA1_MAP6[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节
7	Reserved	-	-	保留
6:0	DMA1_MAP5[6:0]	RW	7'h7F	DMA1 通道请求映射。详见 DMA 章节

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

偏移地址：0x14

复位值：0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	EXTI3[1:0]		Res.	Res.	EXTI2[1:0]		Res.	Res.	EXTI1[1:0]		Res.	Res.	EXTI0[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留

13:12	EXTI3[1:0]	RW	2’b0	EXTI3 配置(EXTI 3 configuration) 这些位可由软件读写，用于选择 EXTI3 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[3]引脚 01：PB[3]引脚 10：PC[3]引脚 11：PD[3]引脚
11:10	Reserved	-	-	保留
9:8	EXTI2[1:0]	RW	2’b0	EXTI2 配置(EXTI 2 configuration) 这些位可由软件读写，用于选择 EXTI2 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[2]引脚 01：PB[2]引脚 10：PC[2]引脚 11：PD[2]引脚
7:6	Reserved	-	-	保留
5:4	EXTI1[1:0]	RW	2’b0	EXTI1 配置(EXTI 1 configuration) 这些位可由软件读写，用于选择 EXTI1 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[1]引脚 01：PB[1]引脚 10：PC[1]引脚 11：PD[1]引脚
3:2	Reserved	-	-	保留
1:0	EXTI0[1:0]	RW	2’b0	EXTI0 配置(EXTI 0 configuration) 这些位可由软件读写，用于选择 EXTI0 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[0]引脚 01：PB[0]引脚 10：PC[0]引脚 11：PD[0]引脚

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

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	EXTI7[1:0]		Res.	Res.	EXTI6[1:0]		Res.	Res.	EXTI5[1:0]		Res.	Res.	EXTI4[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:12	EXTI7[1:0]	RW	2’b0	EXTI7 配置(EXTI7 configuration) 这些位可由软件读写，用于选择 EXTI7 外部中断的输入源。参考 EXTI 外部中断事件映射。

				00：PA[7]引脚 01：PB[7]引脚 10：PC[7]引脚 11：PD[7]引脚
11:10	Reserved	-	-	保留
9:8	EXTI6[1:0]	RW	2’b0	EXTI6 配置(EXTI6 configuration) 这些位可由软件读写，用于选择 EXTI6 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[6]引脚 01：PB[6]引脚 10：PC[6]引脚 11：PD[6]引脚
7:6	Reserved	-	-	保留
5:4	EXTI5[1:0]	RW	2’b0	EXTI5 配置(EXTI5 configuration) 这些位可由软件读写，用于选择 EXTI5 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[5]引脚 01：PB[5]引脚 10：PC[5]引脚 11：PD[5]引脚
3:2	Reserved	-	-	保留
1:0	EXTI4[1:0]	RW	2’b0	EXTI4 配置(EXTI4 configuration) 这些位可由软件读写，用于选择 EXTI4 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[4]引脚 01：PB[4]引脚 10：PC[4]引脚 11：PD[4]引脚

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

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	EXTI11[1:0]		Res.	Res.	EXTI10[1:0]		Res.	Res.	EXTI9[1:0]		Res.	Res.	EXTI8[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:12	EXTI11[1:0]	RW	2’b0	EXTI11 配置(EXTI11 configuration) 这些位可由软件读写，用于选择 EXTI11 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[11]引脚 01：PB[11]引脚 10：PC[11]引脚

				11：PD[11]引脚
11:10	Reserved	-	-	保留
9:8	EXTI10[1:0]	RW	2’b0	EXTI10 配置(EXTI10 configuration) 这些位可由软件读写，用于选择 EXTI10 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[10]引脚 01：PB[10]引脚 10：PC[10]引脚 11：PD[10]引脚
7:6	Reserved	-	-	保留
5:4	EXTI9[1:0]	RW	2’b0	EXTI9 配置(EXTI9 configuration) 这些位可由软件读写，用于选择 EXTI9 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[9]引脚 01：PB[9]引脚 10：PC[9]引脚 11：PD[9]引脚
3:2	Reserved	-	-	保留
1:0	EXTI8[1:0]	RW	2’b0	EXTI8 配置(EXTI8 configuration) 这些位可由软件读写，用于选择 EXTI8 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[8]引脚 01：PB[8]引脚 10：PC[8]引脚 11：PD[8]引脚

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

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	EXTI15[1:0]		Res.	Res.	EXTI14[1:0]		Res.	Res.	EXTI13[1:0]		Res.	Res.	EXTI12[1:0]	
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:12	EXTI15[1:0]	RW	2’b0	EXTI15 配置(EXTI15 configuration) 这些位可由软件读写，用于选择 EXTI15 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[15]引脚 01：PB[15]引脚 10：PC[15]引脚 11：保留
11:10	Reserved	-	-	保留

9:8	EXTI14[1:0]	RW	2’b0	EXTI14 配置(EXTI14 configuration) 这些位可由软件读写，用于选择 EXTI14 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[14]引脚 01：PB[14]引脚 10：PC[14]引脚 11：保留
7:6	Reserved	-	-	保留
5:4	EXTI13[1:0]	RW	2’b0	EXTI13 配置(EXTI13 configuration) 这些位可由软件读写，用于选择 EXTI13 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[13]引脚 01：PB[13]引脚 10：PC[13]引脚 11：保留
3:2	Reserved	-	-	保留
1:0	EXTI12[1:0]	RW	2’b0	EXTI12 配置(EXTI12 configuration) 这些位可由软件读写，用于选择 EXTI12 外部中断的输入源。参考 EXTI 外部中断事件映射。 00：PA[12]引脚 01：PB[12]引脚 10：PC[12]引脚 11：保留

9.2.9 SYSCFG GPIOA 滤波使能寄存器（SYSCFG_PA_ENS）

偏移地址：0x24

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Res.			
15:0	PA_ENS[15:0]	RW	16’h0	GPIOA 0 ~ 15 的噪声滤波使能 0: 关闭噪声滤波功能 1: 开启噪声滤波功能

9.2.10 SYSCFG GPIOB 滤波使能寄存器（PB_ENS）

偏移地址：0x28

复位值：0x0000 0000

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

PB_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PB_ENS[15:0]	RW	16'h0	GPIOB 0 ~ 15 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能

9.2.11 SYSCFG GPIOC 滤波使能寄存器（SYSCFG_PC_ENS）

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC_ENS[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PC_ENS[15:0]	RW	16'h0	GPIOC0~15 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能

9.2.12 SYSCFG_GPIOD 滤波使能寄存器（SYSCFG_PD_ENS）

偏移地址：0x30

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	PD_ENS[11:0]	RW	16'h0	GPIOD0~11 的噪声滤波使能 0：关闭噪声滤波功能 1：开启噪声滤波功能

9.2.13 SYSCFG 电压比较器模拟通道 2 使能寄存器(SYSCFG_COMP_ANA2ENR)

偏移地址：0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PB1_ENA2	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Res.	Res.	Res.	Res.	Res.	Res.	PA9_ENA2	Res.	Res.	Res.	PA5_ENA2	PA4_ENA2	PA3_ENA2	PA2_ENA2	PA1_ENA2	PA0_ENA2
						RW				RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17	PB1_ENA2	RW	1'b0	PB1 的 ANA2 使能，COMP1_INP 专用 0:不使能 1:使能
16:10	Reserved	-	-	保留
9	PA9_ENA2	RW	1'b0	PA9 的 ANA2 使能，COMP2_INP 专用 0:不使能 1:使能
8:6	Reserved	-	-	保留
5	PA5_ENA2	RW	1'b0	PA5 的 ANA2 使能，COMP2_INM 专用 0:不使能 1:使能
4	PA4_ENA2	RW	1'b0	PA4 的 ANA2 使能，COMP1_INM 专用 0:不使能 1:使能
3	PA3_ENA2	RW	1'b0	PA3 的 ANA2 使能，COMP2_INP 专用 0:不使能 1:使能
2	PA2_ENA2	RW	1'b0	PA2 的 ANA2 使能，COMP2_INM 专用 0:不使能 1:使能
1	PA1_ENA2	RW	1'b0	PA1 的 ANA2 使能，COMP1_INP 专用 0:不使能 1:使能

10. DMA 控制器（DMA）

10.1 DMA 简介

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

DMA 控制器有 8 条 DMA 通道，每条通道负责管理来自 1 个或者多个外设对存储器访问的请求。DMA 控制器包括处理 DMA 请求的仲裁器，用于处理各个 DMA 请求的优先级。

10.2 DMA 主要特性

- 8 个独立可配置的通道
- 每个通道通过配置可以连接任一外设的硬件 DMA 请求，每个通道都同样支持软件触发。
- 在同一个 DMA 模块上，多个请求间的优先权可以通过软件编程设置，优先权设置相等时由硬件决定（通道号越低优先级越高）
- 独立数据源和目标数据区的传输宽度（字节、半字、全字），模拟打包和拆包的过程。源和目标地址必须按数据传输宽度对齐
- 可编程的源和目标地址，地址可选递增，递减或不变
- 每个通道都有 4 个事件标志（传输完成、块传输完成、半块传输完成，传输错误），这 4 个事件标志进行“逻辑或”，成为一个单独的中断请求
- 支持存储器和存储器间、外设和存储器、存储器和外设、外设和外设的数据传输
- SRAM、APB 和 AHB 外设均可作为访问的源和目标，Flash 只能作为源不能作为目标
- 支持单次触发模式和四种循环模式
  - 外设地址保持，存储器地址保持
  - 外设地址重新加载，存储器地址保持
  - 外设地址保持，存储器地址重新加载
  - 外设和存储器地址都重新加载
- 单次模式可编程传输数量 0~65535
- 循环模式支持无限循环和有限循环(1~255)
- 支持单一传输和批量传输
  - 单一传输：搬运 1 次数据回复 1 次 ACK；
  - 批量传输：搬运配置的数据量后回复 1 次 ACK(所有数据搬运结束后释放总线)；
- 支持存储器到储存器模式的两种传输方式
  - 快速模式：获得仲裁后始终占据总线，直到传输完所有数据后释放总线；
  - 轮换模式：传输 1 次数据后释放总线重新仲裁；
- 支持在循环模式下进入块传输完成中断后暂停传输

10.3 DMA 功能描述

10.3.1 DMA 顶层结构图

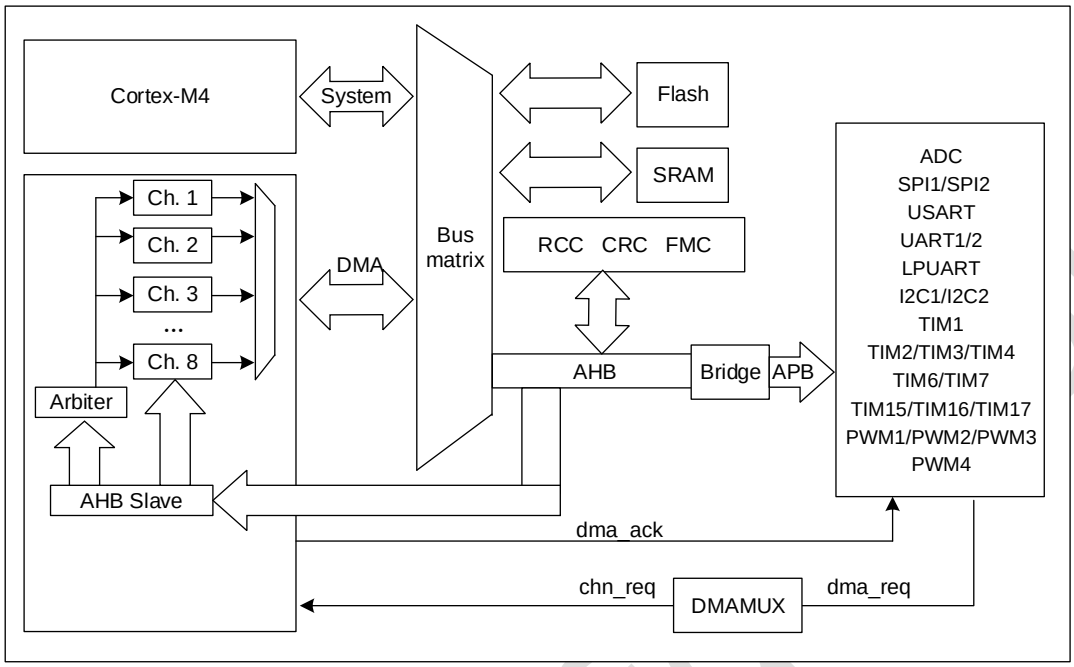


表 10-1 DMA 结构图

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

10.3.2 DMA 传输

在完成一个事件后，外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道的优先权处理请求。当 DMA 控制器开始访问发出请求的外设时，DMA 控制器在传输结束后发送给它一个应答信号。当从 DMA 控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，DMA 控制器撤销应答信号。如果有更多的请求时，外设可以启动下一个传送。

总之，每次 DMA 传送由三个操作组成：

- 从外设数据寄存器或者从当前外设/存储器地址寄存器取数，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设基地址或存储器单元。
- 存数据到外设寄存器或者当前外设/存储器地址寄存器指示的存储器地址，第一次传输时的开始地址是 DMA_CPARx 或 DMA_CMARx 寄存器指定的外设及地址或存储器单元。
- 执行一次 DMA_CNDTRx 寄存器的递减操作，该寄存器表明未完成的操作数目。

10.3.3 DMA 仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。

优先权管理分 2 个阶段：

- 软件：每个通道的优先权可以在 DMA_CCRx 寄存器中设置，有 8 个等级：
  - 0
  - 1
  - 2
  - 3
  - 4

- 5
- 6
- 7

注：优先级 7>6>5>4>3>2>1>0。

- 硬件: 如果 2 个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有更高的优先权。比如，通道 2 优先于通道 3。

10.3.4 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 的传输数据量是可编程的，最大为 65535。包含要传输的数据数量的寄存器，在每次传输后递减。

可编程数据大小

外设和存储器的传输数据宽度可以通过 DMA_CCRx 寄存器中的 MSIZE 和 PSIZE 位编程。

可编程地址增量方式

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的完成自动增量。当设置位增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决于所选的数据宽度为 1，2，4。

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，这些寄存器保持他们的初始值，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）。

当通道配置为单块传输时，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。

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/存储器地址寄存器根据 MNORLD 和 PNORLD 的配置保持地址或者重新加载地址。

循环模式

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

循环模式地址重产生方式有如下 4 种情况：

- 外设地址保持，存储器地址保持
- 外设地址重新加载，存储器地址保持
- 外设地址保持，存储器地址重新加载
- 外设和存储器地址都重新加载

循环模式分为以下两种循环次数：

- 无限循环：通道使能，且 DMA_CCRx.CIRC 为 1，且 DMA_CCCFGRx.NBT 为 0，单块传输数据量为 DMA CNDTRx.NDT；
- 有限循环：通道使能，且 DMA_CCRx.CIRC 为 1，且 DMA_CCCFGRx.NBT 不为 0。
  - DMA_CCCFGRx.NBT 大于 0 时，单块传输数据量为 DMA_CNDTRx.NDT；
  - DMA_CCCFGRx.NBT 等于 0 时，若 DMACCCEGRX.NDTL 等于 0 则传输结束；若 DMACCCFGRx.NDTL 大于 0 则继续传输一次 DMA CCCFGRx.NDTL 个数的数据后传输结束；

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。

存储器到存储器模式分为以下两种模式

- 快速模式：获得仲裁后始终占据总线，直到传输完所有数据后释放总线，该模式不能与循环模式同时使用；
- 轮换模式：传输 1 次数据后释放总线重新仲裁，该模式下其他通道也有获得仲裁的机会；

传输类型

根据外设的传输需求，支持以下两种数据传输方式

- 单一传输：传输 1 次数据回复 1 次 dma_ack
- 批量传输：传输 ndt 次数据回复 1 次 dma_ack，使用该方式传输数据的通道会在传输完所有数据后再释放总线

块传输完成中断后暂停

在循环模式下，可以设置 DMA_CCRx.BTCSUSP 为 1 使通道在进入到块传输完成中断后暂停传输数据，在清除块传输完成中断后恢复数据传输。

通道配置步骤

按如下步骤配置 DMA 通道：

- 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。
- 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。
- 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。
- 在 DMA_CCRx 寄存器的 PL 位中设置通道的优先级。
- 在 DMA_CCRx 寄存器中设置数据传输的方向、循环模式、外设和存储器的增量模式、外设和存储器的数据宽度、传输一半产生中断或传输完成产生中断。
- 在 DMA_CCCFGRx 寄存器中设置循环传输次数。在有限循环时，每个块传输后，这个数值递减。
- 设置 DMA_CCRx 寄存器的 ENABLE 位，启动该通道。

一旦启动了 DMA 通道，即可响应连到该通道上的外设的 DMA 请求。

当传输一半的数据后，半块传输标志(HBTIF)被置 1，当设置了允许半块传输中断位(HBTIE)时，将产生一个中断请求。单块数据传输结束后，块传输完成标志(BTCIF)被置 1，当设置了允许块传输完成中断位(BTCIE)时，将产生一个中断请求。多块数据全部传输结束后，传输完成标志(TCIF)被置 1，当设置了允许传输完成中断位(TCIE)时，将产生一个中断请求。

10.3.5 DMA 数据格式

当存储器数据宽度 MSIZE 和外设数据宽度 PSIZE 不同时，DMA 按照下表进行数据对齐：

表 10-2 数据宽度和大小端（PINC=MINC=1）

源宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
8	8	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1:在 0x0 读 B0[7:0],在 0x0 写 B0[7:0] 2:在 0x1 读 B1[7:0],在 0x1 写 B1[7:0] 3:在 0x2 读 B2[7:0],在 0x2 写 B2[7:0] 4:在 0x3 读 B3[7:0],在 0x3 写 B3[7:0]	0x0/B0 0x1/B1 0x2/B2 0x3/B3
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1:在 0x0 读 B0[7:0],在 0x0 写 00B0[7:0] 2:在 0x1 读 B1[7:0],在 0x2 写 00B1[7:0] 3:在 0x2 读 B2[7:0],在 0x4 写 00B2[7:0] 4:在 0x3 读 B3[7:0],在 0x6 写 00B3[7:0]	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[7:0],在 0x0 写 0000B1B0[31:0] 2:在 0x2 读 B3B2[7:0],在 0x4 写 0000B3B2[31:0] 3:在 0x4 读 B5B4[7:0],在 0x8 写 0000B5B4[31:0] 4:在 0x6 读 B7B6[7:0],在 0xC 写 0000B7B6[31:0]	0x0/0000B1B0 0x4/000B3B2 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[7:0] 2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B5B4[7:0] 3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 B9B8[7:0] 4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BDBC[7:0]	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[7:0] 2:在 0x4 读 B7B6B5B4 [31:0],在 0x2 写 B7B6B5B4[7:0] 3:在 0x8 读 BBBAB9B8 [31:0],在 0x4 写 BBBAB9B8[7:0] 4:在 0xc 读 BFBEBDBC [31:0],在 0x6 写 BFBEBDBC[7:0]	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC

不支持字节/半字访问

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。

10.3.6 DMA 错误处理

读写一个保留的地址区域，将会产生 DMA 传输错误。在 DMA 读写操作时，发生 DMA 传输错误时，硬件会自动清除发生错误的通道所对应的通道配置寄存器（DMA_CCRx）的 EN 位，该通道操作被停止。此时，在 DMA_IFR 寄存器中对应该通道的传输错误中断标志位（TEIF）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。

10.3.7 DMA 外设请求映射

DMA 外设请求映射到 DMA（8 通道）的各个通道，由 SYSCFG 的 DMAx_MAP 寄存器控制，每个外设请求通过配置可以映射到 8 个通道中的任意一个。

序号与外设请求的关系如下表所示：

表 10-2 DMA 外设请求映射

请求 MUX 输入序号	源	请求 MUX 输入序号	源	请求 MUX 输入序号	源
0	ADC1	24	TIM2_CH1	48	TIM16_CH1
1	SPI1_RX	25	TIM2_CH2	49	TIM16_UP
2	SPI1_TX	26	TIM2_CH3	50	TIM17_CH1
3	SPI2_RX	27	TIM2_CH4	51	TIM17_UP
4	SPI2_TX	28	TIM2_UP	52	PWM1_CH1
5	USART_RX	29	TIM2_TRIG	53	PWM1_CH2
6	USART_TX	30	TIM3_CH1	54	PWM1_CH3
7	UART1_RX	31	TIM3_CH2	55	PWM1_CH4
8	UART1_TX	32	TIM3_CH3	56	PWM1_UP
9	UART2_RX	33	TIM3_CH4	57	PWM2_CH1
10	UART2_TX	34	TIM3_UP	58	PWM2_CH2

11	LPUART_RX	35	TIM3_TRIG	59	PWM2_CH3
12	LPUART_TX	36	TIM4_CH1	60	PWM2_CH4
13	I2C1_RX	37	TIM4_CH2	61	PWM2_UP
14	I2C1_TX	38	TIM4_CH3	62	PWM3_CH1
15	I2C2_RX	39	TIM4_CH4	63	PWM3_CH2
16	I2C2_TX	40	TIM4_UP	64	PWM3_CH3
17	TIM1_CH1	41	TIM4_TRIG	65	PWM3_CH4
18	TIM1_CH2	42	TIM6_UP	66	PWM3_UP
19	TIM1_CH3	43	TIM7_UP	67	PWM4_CH1
20	TIM1_CH4	44	TIM15_CH1	68	PWM4_CH2
21	TIM1_COM	45	TIM15_UP	69	PWM4_CH3
22	TIM1_TRIG	46	TIM15_TRIG	70	PWM4_CH4
23	TIM1_UP	47	TIM15_COM	71	PWM4_UP

10.4 DMA 中断

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

表 10-3 DMA 中断请求

中断事件	事件标志位	使能控制位
块传输过半	HBTIF	HBTIE
块传输完成	BTCIF	BTCIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

注：当 DMA_CNDTRx 寄存器为 1 时，不会置 HBTIFx 位

10.5 DMA 寄存器

10.5.1 DMA 中断状态寄存器（DMA_ISR)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEIF8	HBTIF8	BTCIF8	TCIF8	TEIF7	HBTIF7	BTCIF7	TCIF7	TEIF6	HBTIF6	BTCIF6	TCIF6	TEIF5	HBTIF5	BTCIF5	TCIF5
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**
TEIF4	HBTIF4	BTCIF4	TCIF4	TEIF3	HBTIF3	BTCIF3	TCIF3	TEIF2	HBTIF2	BTCIF2	TCIF2	TEIF1	HBTIF1	BTCIF1	TCIF1
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31	TEIF8	R	0	通道 8 传输错误（TE）标志。 硬件置位，软件写 DMA_TEIF8=1 清零。 0：无 TE 事件 1：发生 TE 事件
30	HBTIF8	R	0	通道 8 半块传输完成（HBT）标志。

				硬件置位，软件写 DMA_CHBTIF8=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
29	BTCIF8	R	0	通道 8 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF8=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
28	TCIF8	R	0	通道 8 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF8=1 清零。 0：无 TC 事件 1：发生 TC 事件
27	TEIF7	R	0	通道 7 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF7=1 清零。 0：无 TE 事件 1：发生 TE 事件
26	HBTIF7	R	0	通道 7 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF7=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
25	BTCIF7	R	0	通道 7 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF7=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
24	TCIF7	R	0	通道 7 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF7=1 清零。 0：无 TC 事件 1：发生 TC 事件
23	TEIF6	R	0	通道 6 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF6=1 清零。 0：无 TE 事件 1：发生 TE 事件
22	HBTIF6	R	0	通道 6 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF6=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
21	BTCIF6	R	0	通道 6 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF6=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
20	TCIF6	R	0	通道 6 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF6=1 清零。 0：无 TC 事件 1：发生 TC 事件
19	TEIF5	R	0	通道 5 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF5=1 清零。

				0：无 TE 事件 1：发生 TE 事件
18	HBTIF5	R	0	通道 5 块半传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF5=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
17	BTCIF5	R	0	通道 5 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF5=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
16	TCIF5	R	0	通道 5 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF5=1 清零。 0：无 TC 事件 1：发生 TC 事件
15	TEIF4	R	0	通道 4 传输错误（TE）标志。 硬件置位，软件写 DMA_TEIF4=1 清零。 0：无 TE 事件 1：发生 TE 事件
14	HBTIF4	R	0	通道 4 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF4=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
13	BTCIF4	R	0	通道 4 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF4=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
12	TCIF4	R	0	通道 4 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF4=1 清零。 0：无 TC 事件 1：发生 TC 事件
11	TEIF3	R	0	通道 3 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF3=1 清零。 0：无 TE 事件 1：发生 TE 事件
10	HBTIF3	R	0	通道 3 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF3=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
9	BTCIF3	R	0	通道 3 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF3=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
8	TCIF3	R	0	通道 3 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF3=1 清零。 0：无 TC 事件；

				1：发生 TC 事件；
7	TEIF2	R	0	通道 2 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF2=1 清零。 0：无 TE 事件 1：发生 TE 事件
6	HBTIF2	R	0	通道 2 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF2=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
5	BTCIF2	R	0	通道 2 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF2=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
4	TCIF2	R	0	通道 2 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF2=1 清零。 0：无 TC 事件 1：发生 TC 事件
3	TEIF1	R	0	通道 1 传输错误（TE）标志。 硬件置位，软件写 DMA_CTEIF1=1 清零。 0：无 TE 事件 1：发生 TE 事件
2	HBTIF1	R	0	通道 1 半块传输完成（HBT）标志。 硬件置位，软件写 DMA_CHBTIF1=1 清零。 0：无 HBT 事件 1：发生 HBT 事件
1	BTCIF1	R	0	通道 1 块传输完成（BTC）标志。 硬件置位，软件写 DMA_CBTCIF1=1 清零。 0：无 BTC 事件 1：发生 BTC 事件
0	TCIF1	R	0	通道 1 所有块传输完成（TC）中断标志。 硬件置位，软件写 DMA_CTCIF1=1 清零。 0：无 TC 事件 1：发生 TC 事件

10.5.2 DMA 中断标志清零寄存器（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
CTEIF8	CHBTIF8	CBTCIF8	CTCIF8	CTEIF7	CHBTIF7	CBTCIF7	CTCIF7	CTEIF6	CHBTIF6	CBTCIF6	CTCIF6	CTEIF5	CHBTIF5	CBTCIF5	CTCIF5
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**
CTEIF4	CHBTIF4	CBTCIF4	CTCIF4	CTEIF3	CHBTIF3	CBTCIF3	CTCIF3	CTEIF2	CHBTIF2	CBTCIF2	CTCIF2	CTEIF1	CHBTIF1	CBTCIF1	CTCIF1
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31	CTEIF8	W	0	通道 8 传输错误标志清零。 0：无效果

				1：清零 TEIF8
30	CHBTIF8	W	0	通道 8 半块传输标志清零。 0：无效果 1：清零 HBTIF8
29	CBTCIF8	W	0	通道 8 块传输完成标志清零。 0：无效果 1：清零 BTCIF8
28	CTCIF8	W	0	通道 8 所有块传输完成标志清零。 0：无效果 1：清零 TCIF8
27	CTEIF7	W	0	通道 7 传输错误标志清零。 0：无效果 1：清零 TEIF7
26	CHBTIF7	W	0	通道 7 半块传输标志清零。 0：无效果 1：清零 HBTIF7
25	CBTCIF7	W	0	通道 7 块传输完成标志清零。 0：无效果 1：清零 BTCIF7
24	CTCIF7	W	0	通道 7 所有块传输完成标志清零。 0：无效果 1：清零 TCIF7
23	CTEIF6	W	0	通道 6 传输错误标志清零。 0：无效果 1：清零 TEIF6
22	CHBTIF6	W	0	通道 6 块半传输标志清零。 0：无效果 1：清零 HBTIF6
21	CBTCIF6	W	0	通道 6 块传输完成标志清零。 0：无效果 1：清零 BTCIF6
20	CTCIF6	W	0	通道 6 所有块传输完成标志清零。 0：无效果 1：清零 TCIF6；
19	CTEIF5	W	0	通道 5 传输错误标志清零。 0：无效果； 1：清零 TEIF5
18	CHBTIF5	W	0	通道 5 块半传输标志清零。 0：无效果 1：清零 HBTIF5
17	CBTCIF5	W	0	通道 5 块传输完成标志清零。 0：无效果 1：清零 BTCIF5
16	CTCIF5	W	0	通道 5 所有块传输完成标志清零。 0：无效果；

				1：清零 TCIF5
15	CTEIF4	W	0	通道 4 传输错误标志清零。 0：无效果 1：清零 TEIF4
14	CHBTIF4	W	0	通道 4 块半传输标志清零。 0：无效果 1：清零 HBTIF4
13	CBTCIF4	W	0	通道 4 块传输完成标志清零。 0：无效果 1：清零 BTCIF4
12	CTCIF4	W	0	通道 4 所有块传输完成标志清零。 0：无效果 1：清零 TCIF4
11	CTEIF3	W	0	通道 3 传输错误标志清零。 0：无效果； 1：清零 TEIF3；
10	CHBTIF3	W	0	通道 3 块半传输标志清零。 0：无效果 1：清零 HBTIF3
9	CBTCIF3	W	0	通道 3 块传输完成标志清零。 0：无效果 1：清零 BTCIF3
8	CTCIF3	W	0	通道 3 所有块传输完成标志清零。 0：无效果 1：清零 TCIF3
7	CTEIF2	W	0	通道 2 传输错误标志清零。 0：无效果 1：清零 TEIF2
6	CHBTIF2	W	0	通道 2 块半传输标志清零。 0：无效果； 1：清零 HBTIF2
5	CBTCIF2	W	0	通道 2 块传输完成标志清零。 0：无效果 1：清零 BTCIF2
4	CTCIF2	W	0	通道 2 所有块传输完成标志清零。 0：无效果 1：清零 TCIF2
3	CTEIF1	W	0	通道 1 传输错误标志清零。 0：无效果； 1：清零 TEIF1
2	CHBTIF1	W	0	通道 1 块半传输标志清零。 0：无效果 1：清零 HBTIF1
1	CBTCIF1	W	0	通道 1 块传输完成标志清零。 0：无效果

				1：清零 BTCIF1
0	CTCIF1	W	0	通道 1 所有块传输完成标志清零。 0：无效果 1：清零 TCIF1

10.5.3 DMA 通道 x 控制寄存器（DMA_CCRx）(x=1~8)

偏移地址：0x08+x*0x14(x=1~8)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	M2MARB	BTCSUSP	MNORLD	PNORLD	MEM2MEM	PL[2:0]		
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MSIZE[1:0]		PSIZE[1:0]		MINC[1:0]		PINC[1:0]		TRANST	CIRC	DIR	TEIE	HBTIE	BTCIE	TCIE	EN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	M2MARB	RW	0	通道 x 存储器到存储器模式下仲裁方式 （mem2mem=1 时有效） 0：传输完 1 次数据后不释放总线；（不支持循环模式） 1：传输完 1 次数据后释放总线； EN=0 时软件可读可写； EN=1 时软件只可读；
22	BTCSUSP	RW	0	通道 x 循环模式下进入 BTC 中断后处理方式 0：不暂停传输； 1：暂停传输； 该位配置为 1 时，进入 BTC 中断后软件清除 BTC 中断后恢复传输； EN=0 时软件可读可写； EN=1 时软件只可读；
21	MNORLD	RW	0	通道 x 的存储器地址循环模式（CICR=1 时有效） 0：存储器地址重新加载使能； 1：存储器地址重新加载禁止； EN=0 时软件可读可写； EN=1 时软件只可读；
20	PNORLD	RW	0	通道 x 的外设地址循环模式（CICR=1 时有效） 0：外设地址重新加载使能； 1：外设地址重新加载禁止； EN=0 时软件可读可写； EN=1 时软件只可读；
19	MEM2MEM	RW	0	通道 x 存储器到存储器模式。 0：禁止； 1：存储器到存储器模式使能； EN=0 时软件可读可写； EN=1 时软件只可读；

18:16	PL	RW	3'h0	通道 x 优先级配置。 000：通道优先级为 0 001：通道优先级为 1 010：通道优先级为 2 011：通道优先级为 3 100：通道优先级为 4 101：通道优先级为 5 110：通道优先级为 6 111：通道优先级为 7 EN=0 时软件可读可写 EN=1 时软件只可读
15:14	MSIZE	RW	2'h0	通道 x 存储器数据宽度。 00：8 位 01：16 位 10：32 位 11：保留 EN=0 时软件可读可写 EN=1 时软件只可读
13:12	PSIZE	RW	2'h0	通道 x 外设数据宽度。 00：8 位 01：16 位 10：32 位 11：保留 EN=0 时软件可读可写 EN=1 时软件只可读
11:10	MINC	RW	2'h0	通道 x 存储器地址增量模式。 00：不变 01：递增 10：递减 11：保留 EN=0 时软件可读可写 EN=1 时软件只可读
9:8	PINC	RW	2'h0	通道 x 外设地址增量模式。 00：不变 01：递增 10：递减 11：保留 EN=0 时软件可读可写 EN=1 时软件只可读
7	TRANST	RW	0	通道 x 的数据传输方式（mem2mem=0 时有效） 0：单一传输 1：批量传输 EN=0 时软件可读可写 EN=1 时软件只可读
6	CIRC	RW	0	循环模式

				0：禁止 1：使能 EN=0 时软件可读可写 EN=1 时软件只可读
5	DIR	RW	0	通道 x 数据传输方向。 0：从外设读 1：从存储器读 EN=0 时软件可读可写 EN=1 时软件只可读
4	TEIE	RW	0	通道 x 传输错误中断（TE）使能。 0：禁止 1：TE 中断使能
3	HBTIE	RW	0	通道 x 半块传输中断（HBT）使能。 0：禁止 1：HBT 中断使能
2	BTCIE	RW	0	通道 x 块传输完成中断（BTC）使能。 0：禁止 1：BTC 中断使能
1	TCIE	RW	0	通道 x 所有块传输完成中断（TC）使能。 0：禁止 1：TC 中断使能
0	EN	RW	0	通道 x 使能. 0：禁止 1：通道 x 使能 当所有 block 传输完成或者出现传输错误时硬件清零； 软件可读可写； 注 1：在无效配置，如配置 NDT=0，但配置 EN=1 时，硬件会清零 EN，以保证后续传输正常。 注 2：配置 DMA 寄存器遵循先配置后控制的原则，即在最后配置 EN=1。

10.5.4 DMA 通道 x 数据传输个数寄存器（DMA_CNDTRx）(x=1~8)

偏移地址： 0x0C+x*0x14(x=1~8)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	NDT	RW	16’h0	通道 x 数据传输数量。 数据传输数量为 0~65535。该寄存器只在通道不工作（DMA_CCRx.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输字节数。该寄存器值在每次 DMA 传输后递减。

				数据传输结束后，寄存器的内容或者变为 0，或者当该通道配置为循环模式时，寄存器的内容将被自动重新加载为之前配置时的数值。 当该寄存器值为 0 时，即使 DMA 通道开始，都不会传输数据。 EN=0 时软件可读可写 EN=1 时软件只可读

10.5.5 DMA 通道 x 外设地址寄存器（DMA_CPARx）(x=1~8)

偏移地址：0x10+x*0x14(x=1~8)

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	Name	R/W	Reset Value	Function
31:0	PA	RW	32’h0	通道 x 外设地址。 通道 x 外设数据寄存器的基址，作为数据传输的源或目标。 当 PSIZE=2’b01，不使用 PA[0]位。操作自动与半字地址对齐。 当 PSIZE=2’b10，不使用 PA[1:0]位。操作自动与字地址对齐。 EN=0 时软件可读可写 EN=1 时软件只可读

10.5.6 DMA 通道 x 存储地址寄存器（DMA_CMARx）(x=1~8)

偏移地址：0x14+x*0x14(x=1~8)

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	Name	R/W	Reset Value	Function
31:0	MA	RW	32’h0	通道 x 存储器地址。 通道 x 存储器地址，作为数据传输的源或目标。 当 MSIZE=2’b01，不使用 MA[0]位。操作自动与半字地址对齐。 当 MSIZE=2’b10，不使用 MA[1:0]位。操作自动与字地址对齐。 EN=0 时软件可读可写 EN=1 时软件只可读

10.5.7 DMA 通道 x 块循环传输配置寄存器（DMA_CCCFGRx）(x=1~8)

偏移地址：0x18+x*0x14(x=1~8)

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDTL[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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NBT[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	NDTL	RW	16’h0	有限循环模式下通道 x 最后一次 block 数据传输数量。 数据传输数量为 0~65535。 该寄存器只在通道不工作(DMA CCRx.EN=0)时写入。通道使能后该寄存器为只读，表明循环模式最后一次剩余传输数量。 该寄存器值在循环模式最后一个 block 传输时随每笔 传输递减。 EN=0 时软件可读可写 EN=1 时软件只可读
15:8	Reserved	-	-	保留
7:0	NBT	RW	8’h0	通道 x block 循环传输数量。 在循环模式下，该寄存器初始值为 0 时 block 循环传输数量为无穷，该寄存器初始值大于 0 时 block 循环传输数量为该寄存器值(1~255)。该寄存器只在通道不工作（DMA_CCRx.EN=0）时写入。通道使能后该寄存器为只读，表明剩余传输 block 数。该寄存器值在每个 block 传输完成后递减。 EN=0 时软件可读可写 EN=1 时软件只可读 注:有限循环传输数据个数=NDT*NBT+NDTL

11. 中断和事件

11.1 嵌套向量中断控制器(NVIC)

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

嵌套向量中断控制器管理着包括内核异常等中断。更多关于异常和 NVIC 编程的说明请参考 Cortex-M4 编程手册。

11.1.1 NVIC 主要特性

- 53 个可屏蔽中断通道(不包含 16 个 Cortex®-M4 的中断线)
- 8 个可编程的优先等级(使用了 3 位中断优先级)
- 低延迟的异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

11.1.2 SysTick 校准值寄存器

系统嘀嗒校准值固定为 9000，当系统嘀嗒时钟设定为 9 MHz，产生 1 ms 时间基准。

11.1.3 中断和异常向量表

表 11-1 终端和异常向量表

向量编号	优先级	优先级类型	名称	地址	说明
-	-	-	-	0x0000_0000	保留
-	-3	固定	Reset	0x0000_0004	复位
-	-2	NMI	RCC HSE 时钟安全（HSE CSS)	0x0000_0008	不可屏蔽中断
-	-1	固定	硬件失效(HardFault)	0x0000_000C	所有类型的失效
-	0	可编程	存储管理(MemManage)	0x0000_0010	存储器管理
-	1	可编程	总线错误(BusFault)	0x0000_0014	预取指失败，存储器访问失败
-	2	可编程	错误应用(UsageFault)	0x0000_0018	未定义的指令或非法状态
-	-	-	-	0x0000_001C	保留
-	-	-	-	0x0000_0020	保留
-	-	-	-	0x0000_0024	保留
-	-	-	-	0x0000_0028	保留
-	3	可编程	SVCall	0x0000_002C	通过 SWI 指令的系统服务调用
-	4	可编程	调试监控(DebugMonitor)	0x0000_0030	调试监控器
-	-	-	-	0x0000_0034	保留
-	5	可编程	PendSV	0x0000_0038	可挂起的系统服务
-	6	可编程	SysTick	0x0000_003C	系统嘀嗒定时器
0	7	可编程	WWDG	0x0000_0040	窗口看门狗定时器中断
1	8	可编程	PVD	0x0000_0044	连到 EXTI 的电源电压检测(PVD)中断

2	9	可编程	TAMPER	0x0000_0048	侵入检测中断
3	10	可编程	RTC	0x0000_004C	实时时钟(RTC)全局中断
4	11	可编程	FMC	0x0000_0050	闪存 FMC 全局中断
5	12	可编程	RCC	0x0000_0054	复位和时钟控制(RCC)中断
6	13	可编程	EXTI0	0x0000_0058	EXTI 线 0 中断
7	14	可编程	EXTI1	0x0000_005C	EXTI 线 1 中断
8	15	可编程	EXTI2	0x0000_0060	EXTI 线 2 中断
9	16	可编程	EXTI3	0x0000_0064	EXTI 线 3 中断
10	17	可编程	EXTI4	0x0000_0068	EXTI 线 4 中断
11	18	可编程	EXTI9_5	0x0000_006C	EXTI 线[9:5]中断
12	19	可编程	EXTI15_10	0x0000_0070	EXTI 线[15:10]中断
13	20	可编程	DMA1 通道 1	0x0000_0074	DMA1 通道 1 全局中断
14	21	可编程	DMA1 通道 2	0x0000_0078	DMA1 通道 2 全局中断
15	22	可编程	DMA1 通道 3	0x0000_007C	DMA1 通道 3 全局中断
16	23	可编程	DMA1 通道 4	0x0000_0080	DMA1 通道 4 全局中断
17	24	可编程	DMA1 通道 5	0x0000_0084	DMA1 通道 5 全局中断
18	25	可编程	DMA1 通道 6	0x0000_0088	DMA1 通道 6 全局中断
19	26	可编程	DMA1 通道 7	0x0000_008C	DMA1 通道 7 全局中断
20	27	可编程	DMA1 通道 8	0x0000_0090	DMA1 通道 8 全局中断
21	28	可编程	ADC	0x0000_0094	ADC 全局中断
22	29	可编程	TIM1_BRK	0x0000_0098	TIMER1 中止中断
23	30	可编程	TIM1_UP	0x0000_009C	TIMER1 更新中断
24	31	可编程	TIM1_TRG_COM	0x0000_00A0	TIMER1 触发和通信中断
25	32	可编程	TIM1_CC	0x0000_00A4	TIMER1 捕获比较中断
26	33	可编程	TIM2	0x0000_00A8	TIMER2 全局中断
27	34	可编程	TIM3	0x0000_00AC	TIMER3 全局中断
28	35	可编程	TIM4	0x0000_00B0	TIMER4 全局中断
29	36	可编程	TIM6	0x0000_00B4	TIM6 全局中断
30	37	可编程	TIM7	0x0000_00B8	TIM7 全局中断
31	38	可编程	TIM15	0x0000_00BC	TIMER15 全局中断
32	39	可编程	TIM16	0x0000_00C0	TIMER16 全局中断
33	40	可编程	TIM17	0x0000_00C4	TIMER17 全局中断
34	41	可编程	PWM1	0x0000_00C8	PWM1 全局中断
35	42	可编程	PWM2	0x0000_00CC	PWM2 全局中断
36	43	可编程	PWM3	0x0000_00D0	PWM3 全局中断
37	44	可编程	PWM4	0x0000_00D4	PWM4 全局中断
38	45	可编程	LPTIM	0x0000_00D8	LPTIM 全局中断
39	46	可编程	SPI1	0x0000_00DC	SPI1 全局中断

40	47	可编程	SPI2	0x0000_00E0	SPI2 全局中断
41	48	可编程	USART	0x0000_00E4	USART 全局中断
42	49	可编程	UART1	0x0000_00E8	UART1 全局中断
43	50	可编程	UART2	0x0000_00EC	UART2 全局中断
44	51	可编程	LPUART	0x0000_00F0	LPUART 全局和唤醒中断
45	52	可编程	RTCAlarm	0x0000_00F4	连到 EXTI 的 RTC 闹钟中断
46	53	可编程	I2C1_EV	0x0000_00F8	I2C1 事件中断
47	54	可编程	I2C1_ER	0x0000_00FC	I2C1 错误中断
48	55	可编程	I2C2_EV	0x0000_0100	I2C2 事件中断
49	56	可编程	I2C2_ER	0x0000_0104	I2C2 错误中断
50	57	可编程	COMP1	0x0000_0108	COMP1 全局中断
51	58	可编程	COMP2	0x0000_010C	COMP2 全局中断
52	59	可编程	OPA2	0x0000_0110	OPA2 作为 COMP 时的全局中断

11.2 外部扩展中断/事件控制器（EXTI）

扩展中断和事件控制器，通过可配置线和直接输入线，管理着 CPU 和系统唤醒功能，并输出下述请求信号：

- 事件请求，送给 CPU 的事件输入
- 唤醒请求，送给功耗管理控制模块

EXTI 唤醒请求允许系统从停止模式唤醒，事件请求也可以在运行模式使用。

EXTI 允许管理多达 24 个可配置线和直接输入线（20 个可配置线和 4 个直接输入线）。

有 24 个能产生事件/中断请求的边沿检测器。其中 20 个可配置线，每个输入线可以独立地配置输入触发方式(上升沿或下降沿或者双边沿触发)。24 个可配置线和直接输入线，每个输入线都可以独立地被屏蔽。挂起寄存器保持着可配置线的中断请求。

11.2.1 EXTI 主要特性

EXTI 控制器的主要特性如下：

- 可通过 GPIO 和指定模块(如 PVD/RTC 等) 唤醒低功耗模式
- 可配置线
  - 可选有效触发沿(上升沿/下降沿)
  - 中断挂起标志位
  - 独立中断和事件屏蔽位
  - 可软件触发
- 直接输入线
  - 固定的上升沿触发
  - 无中断挂起标志位
  - 独立中断和事件屏蔽位
  - 无软件触发

11.2.2 EXTI 功能说明

要产生中断，必须先配置好并使能中断线。20 个可配置线根据需要的边沿检测设置 2 个触发寄存器，同时在中断屏蔽寄存器的相应位写'1'允许中断请求。当外部中断线上发生了期待的边沿时，将产生一个中断请求，对应的挂起位也随之被置'1'。在挂起寄存器的对应位写'1'，将清除该中断请求。如果需要产生事件，必须先配置好并使能事件线。根据需要的边沿检测通过设置 2 个触发寄存器，同时在事件屏蔽寄存器的相应位写'1'允许事件请求。当事件线上发生了需要的边沿时，将产生一个事件请求脉冲，对应的挂起位不被置'1'。通过在软件中断事件寄存器写'1'，也可以通过软件产生中断/事件请求。

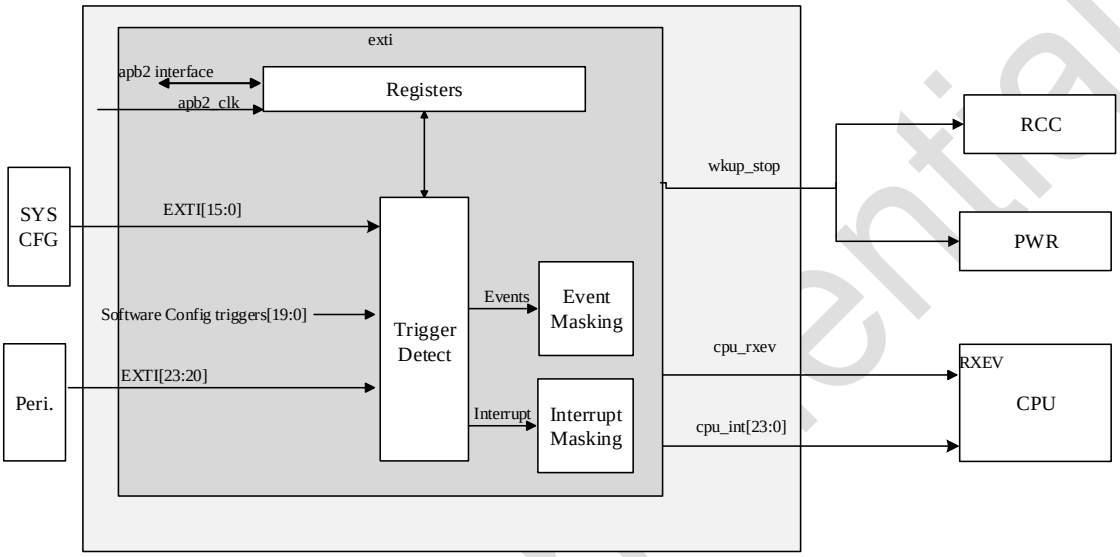


图 11-1 EXTI 框图

硬件中断选择

- 配置中断线的屏蔽位(EXTI_IMR)
- 配置所选中断线的触发选择位(EXTI_RTSR 和 EXTI_FTSR);

硬件事件选择

- 配置事件线的屏蔽位(EXTI_EMR)
- 配置事件线的触发选择位(EXTI_RTSR 和 EXTI_FTSR)

软件中断/事件的选择

- 配置中断/事件线屏蔽位(EXTI_IMR, EXTI_EMR)
- 设置软件中断寄存器的请求位(EXTI_SWIER)

11.2.3 EXTI 唤醒事件管理

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

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

11.2.4 EXTI 外部中断/事件线映射

最多允许 60 个通用 I/O 端口以下图的方式连接到 16 个外部中断/事件线上：

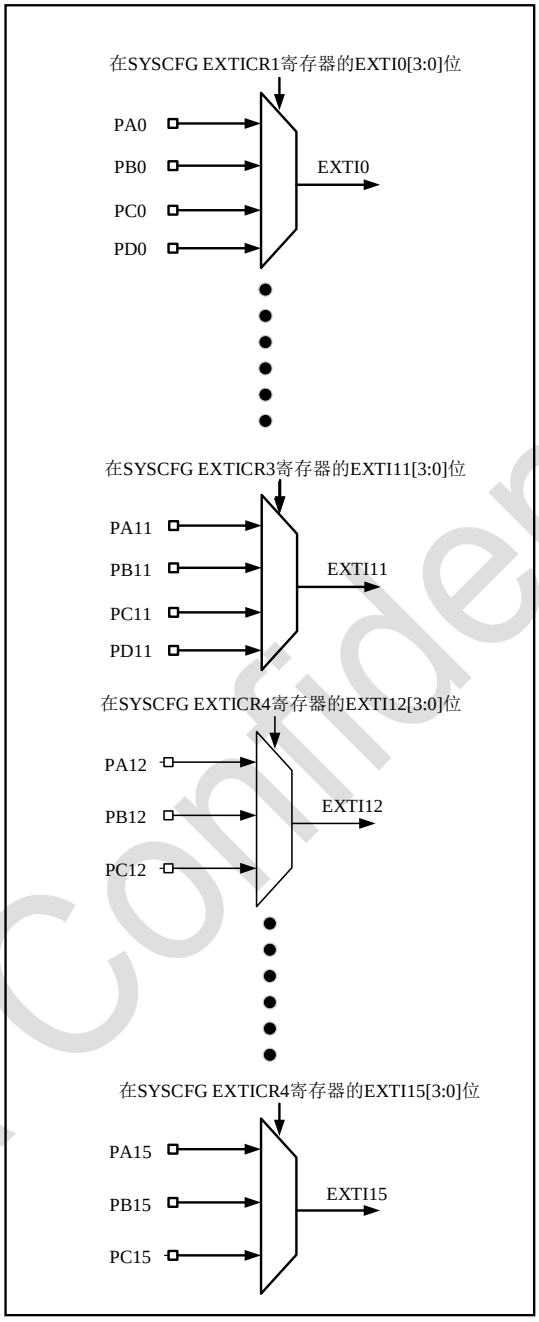


图 11-2 EXTI 线连接

注：通过 SYSCFG_EXTICRx 配置 GPIO 线上的外部中断/事件，必须先使能 SYSCFG 时钟。

表 11-2 EXTI 线连接

EXTI	Line 源	Line 类型
0-15	GPIO	可配置线
16	PVD	可配置线
17	RTC alarm 事件	可配置线
18	COMP1 输出	可配置线

19	COMP2 输出	可配置线
20	I2C1 EV 唤醒	直接输入线
21	I2C2 EV 唤醒	直接输入线
22	LPTIM 唤醒	直接输入线
23	LPUART 唤醒	直接输入线

11.3 EXTI 寄存器

必须以 32 位的方式对这些寄存器进行访问。

11.3.1 EXTI 中断屏蔽寄存器（EXTI_IMR）

偏移地址：0x00

复位值：0x00F0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IMR23	IMR22	IMR21	IMR20	IMR19	IMR18	IMR17	IMR16
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23	IMR23	RW	1	**IMR23**：线 23 上的中断屏蔽 (Interrupt Mask on line 23) 0：屏蔽来自线 23 上的中断请求 1：开放来自线 23 上的中断请求
22	IMR22	RW	1	**IMR22**：线 22 上的中断屏蔽 (Interrupt Mask on line 22) 0：屏蔽来自线 22 上的中断请求 1：开放来自线 22 上的中断请求
21	IMR21	RW	1	**IMR21**：线 21 上的中断屏蔽 (Interrupt Mask on line 21) 0：屏蔽来自线 21 上的中断请求 1：开放来自线 21 上的中断请求
20	IMR20	RW	1	**IMR20**：线 20 上的中断屏蔽 (Interrupt Mask on line 20) 0：屏蔽来自线 20 上的中断请求 1：开放来自线 20 上的中断请求
19:0	IMRx(x=0-19)	RW	20'h0	**IMRx**：线 x 上的中断屏蔽 (Interrupt Mask on line x) 0：屏蔽来自线 x 上的中断请求 1：开放来自线 x 上的中断请求

11.3.2 EXTI 事件屏蔽寄存器（EXTI_EMR）

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EMR23	EMR22	EMR21	EMR20	EMR19	EMR18	EMR17	EMR16
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EMR15	EMR14	EMR13	EMR12	EMR11	EMR10	EMR9	EMR8	EMR7	EMR6	EMR5	EMR4	EMR3	EMR2	EMR1	EMR0

RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:0	EMRx(x=0-23)	RW	24'h0	**EMRx**: 线 x 上的事件屏蔽 (Event Mask on line x) 0：屏蔽来自线 x 上的事件请求 1：开放来自线 x 上的事件请求

11.3.3 EXTI 上升沿触发选择寄存器（EXTI_RTSR）

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTSR 19	RTSR 18	RTSR 17	RTSR 16
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RTSR 15	RTSR 14	RTSR 13	RTSR 12	RTSR 11	RTSR 10	RTS R9	RTS R8	RTS R7	RTS R6	RTS R5	RTS R4	RTSR 3	RTSR 2	RTSR 1	RTSR 0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	RTSRx(x=0-19)	RW	20'h0	**RTSRx**: 线 x 上的上升沿触发事件配置位 0：禁止输入线 x 上的上升沿触发(中断和事件) 1：允许输入线 x 上的上升沿触发(中断和事件)

11.3.4 EXTI 下降沿触发选择寄存器（EXTI_FTSR）

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FTS R 19	FTS R 18	FTS R 17	FTS R 16
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FTSR 15	FTSR 14	FTFR 13	FTSR 12	FTSR 11	FTSR 10	FTSR 9	FTS R8	FTS R7	FTS R6	FTS R5	FTS R4	FTS R3	FTS R2	FTS R1	FTS R0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	FTSRx(x=0-19)	RW	20'h0	RTSRx: 线 x 上的下降沿触发事件配置位 0：禁止输入线 x 上的下降沿触发(中断和事件) 1：允许输入线 x 上的下降沿触发(中断和事件)

11.3.5 EXTI 软件中断事件寄存器（EXTI_SWIER）

偏移地址：0x10

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SWIE R19	SWIE R18	SWIE R17	SWIE R16
												RW	RW	RW	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWIER15	SWIER14	SWIER13	SWIER12	SWIER11	SWIER10	SWIER9	SWIER8	SWIER7	SWIER6	SWIER5	SWIER4	SWIER3	SWIER2	SWIER1	SWIER0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	SWIERx(x=0-19)	RW	20'h0	SWIERx: 线 x 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。 注：通过清除 EXTI_PR 的对应位(写入'1')，可以清除该位为'0'。

11.3.6 EXTI 挂起寄存器（EXTI_PR）

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PR19	PR18	PR17	PR16
												RC_W1	RC_W1	RC_W1	RC_W1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
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	Name	R/W	Reset Value	Function
31: 20	Reserved	-	-	保留
19: 0	PRx(x=0-19)	RC_W1	20'h0	PRx(x=0-19): 挂起位 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置'1'。在该位中写入'1'可以清除它，也可以通过改变边沿检测的极性清除。

12. CRC 计算单元（CRC）

12.1 CRC 简介

循环冗余校验(CRC)单元可以利用 32 位的以太网多项式计算出 CRC 码。

在其他的应用中，基于 CRC 的技术用于验证数据传输或存储的完整性。在功能安全标准的领域内，基于 CRC 的技术提供了一种验证闪存完整性的方法。CRC 计算单元可帮助软件签名的计算。

12.2 CRC 主要特性

$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

- 仅支持对 32 位的数据的计算
- CRC 初始值为 0xFFFFFFFF
- 具有一个通用 8 位寄存器(可用于存放临时数据)
- 具备输入缓存功能
- 具有一个 32 位数据寄存器用于输入/输出
- CRC 计算时间：4 个 AHB 时钟周期(32 位数据)

12.3 CRC 功能描述

12.3.1 CRC 结构框图

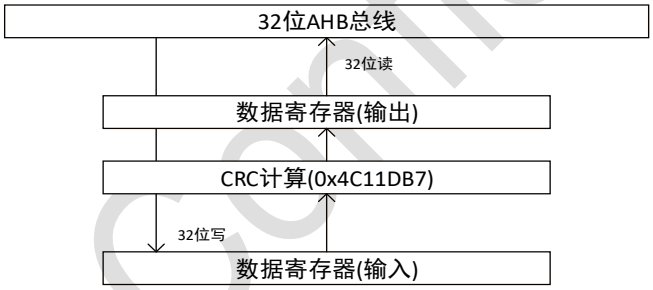


图 12-1 CRC 模块的结构

12.3.2 CRC 操作

CRC 计算单元含有 1 个 32 位数据寄存器(CRC_DR)。

对于写操作，它用于保存输入数据；对于读操作，它用于保存之前的 CRC 计算结果。

每一次写入数据寄存器，其计算结果是前一次 CRC 计算结果和新计算结果的组合(对整个 32 位字进行 CRC 计算，而不是逐字节地计算)。

输入缓存允许用户在写入第一个数据之后立刻写入第二个数据，无需等待之前的数据计算完毕。

可以通过向寄存器 CRC_CR 的 RESET 位写 1 来重置寄存器 CRC_DR 为 0xFFFFFFFF。该操作不影响寄存器 CRC_IDR 内的数据。

CRC_IDR 寄存器可以用来保存与 CRC 计算相关的临时值。它不受 CRC_CR 寄存器中 RESET 位的影响。

12.4 CRC 寄存器

12.4.1 CRC 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	Name	R/W	Reset Value	Function
31:0	DR[31:0]	RW	32'hFFFFFFFF	数据寄存器位。 写入 CRC 计数器的新数据时，作为输入寄存器。 读取时返回之前的 CRC 计算的结果。

12.4.2 CRC 独立数据寄存器（CRC_IDR）

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	IDR[7:0]	RW	8'h0	通用 8 位数据寄存器位。可临时存放数据。 寄存器 CRC_CR 的 RESET 位产生的 CRC 复位对本寄存器无影响。 注：此寄存器不参与 CRC 计算，可以存放任何数据。

12.4.3 CRC 控制寄存器（CRC_CR）

偏移地址：0x08

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	RESET	W	0	软件往此位写 1 将会复位 CRC 模块。 软件只能写 1，由硬件清零。

13. 模拟/数字转换（ADC）

13.1 ADC 简介

芯片具有 1 个 12 位的 SARADC（Successive approximation analog-to-digital converter）。该模块共有 21 个要被测量的通道，包括 16 个外部通道和 5 个内部通道的模拟信号，外部通道包含 5 对差分通道。

各通道的转换模式可以设定为单次、连续、非连续模式。转换结果存储在左对齐或者右对齐的 16 位数据寄存器中。

模拟看门狗允许应用检测是否输入电压超出了用户定义的高或者低阈值。

ADC 实现了在低频率下运行，可获得很低的功耗。

13.2 ADC 主要特性

- 高性能:
  - 12-bit、10-bit、8-bit 和 6-bit 分辨率可配置；
  - ADC 转换时间：0.3 us@12-bit(3 Msps，2.5 T SMP，f_{ADC} = 48 MHz)
  - 自校准（软件启动）
  - 支持规则转换和注入转换
  - 可编程采样时间
  - 可配置数据对齐模式(左对齐或者右对齐)
  - 规则通道转换支持 DMA 请求。
  - 配置支持 16 个规则序列转换
  - 配置支持 4 个注入序列转换
  - 配置支持 4 个注入通道转换数据寄存器
- 过采样器
  - 16 位数据寄存器
  - 过采样率可在 2x 到 256x 之间进行调整
  - 可编程数据最多可移位 8 位
- 数据预处理
  - 增益补偿
  - 偏移补偿
- 动态低功耗特性:
  - 自动延迟转换模式：防止以低频 PCLK 运行产生溢出
  - 为低功耗操作，降低工作频率，而仍然维持合适的 ADC 性能
- 模拟输入通道:
  - ADC 连接 16 个外部通道，5 个内部通道
  - 5 对差分通道
- 启动转换方式:
  - 软件启动
  - 可配置极性的硬件触发
- 转换模式:
  - 单次模式：1 次触发对所有选中的通道执行一次转换

- 连续模式: 1 次触发连续转换被选择的通道，直到软件终止
- 间断模式: 1 次触发连续转换子序列通道，多次触发直到转换所有通道

- 模拟看门狗
- 中断的产生:
  - 在单个通道采样结束
  - 在单个通道转换结束(规则/注入通道)
  - 在规则序列或注入序列转换结束
  - 模拟看门狗 1，2，3 事件
  - 过载事件
  - 校准完成
  - AD 上电完成

13.3 ADC 功能描述

13.3.1 ADC 框图

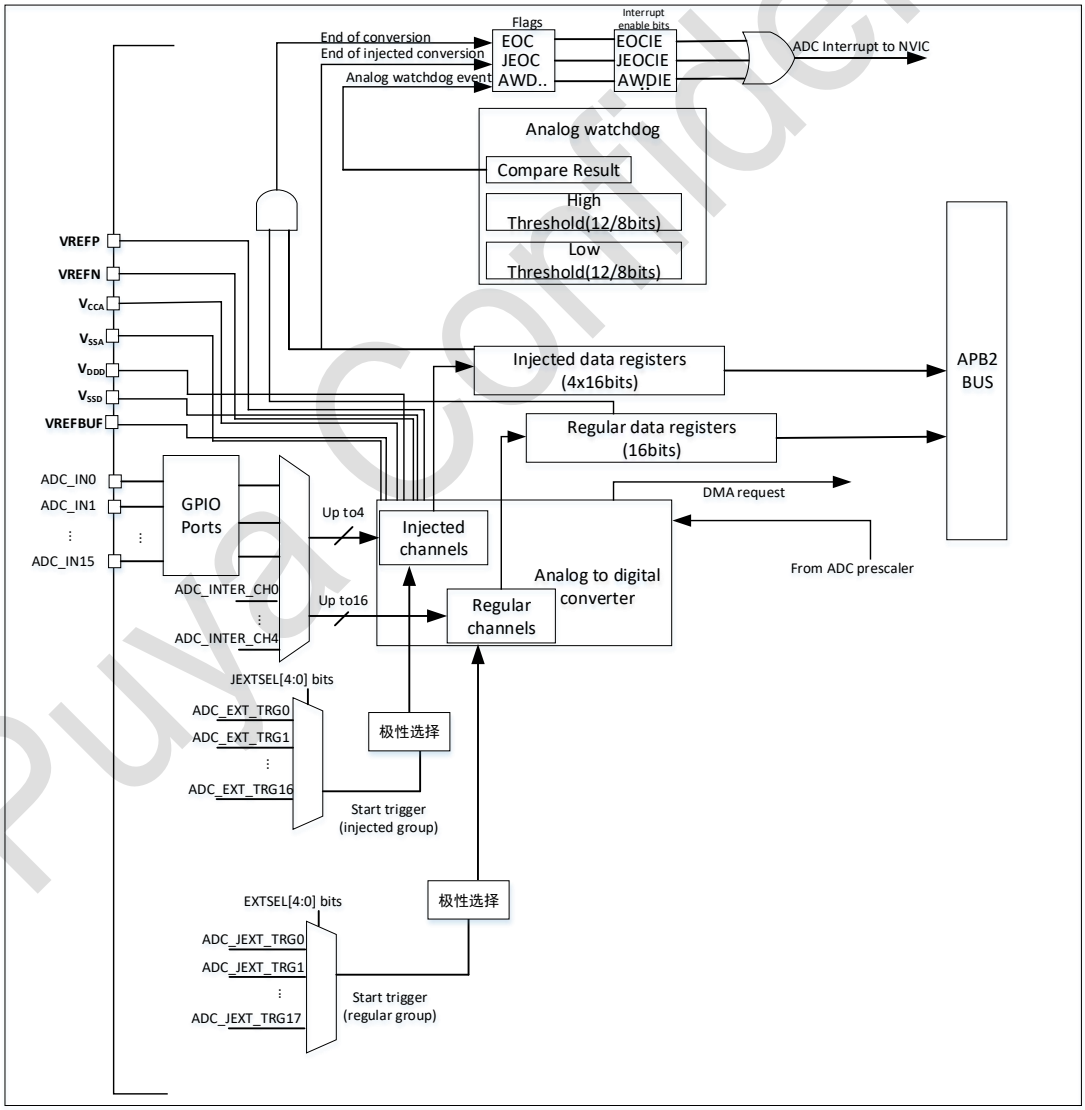


图 13-1 ADC结构图

13.3.2 单端和差分输入通道

ADC 部分通道可配置为单端输入或差分输入，通过 ADC_CCR.DIFF_EN 配置。

当 DIFF_EN 为 1 时， ADC_CH0-CH4 通道为差分输入模式，否则为单端输入模式。

必须当 ADC 被禁用时（ADEN=0）配置。

在单端模式下，通道 i 的模拟电压为 VINP[i]和 VREFN 之差。

在差分模式下，通道 i 的模拟电压为 VINP[i]和 VINN[i]电压差。

差分模式下，输入电压范围为 V_{REFN} 到 V_{REFP}，达到 2xVREFP 的全刻度。当 VINP[i]等于 VREFN,VINN[i]等于 VREFP 则最大负端输入差分电压(V_{REFN})对应 0x000 输出。当 VINP[i]等于 VREFP,VINN[i]等于 VREFN 则最大正端输入差分电压(VREFP)对应 0xFFF 输出。当 VINP[i]和 VINN[i]连在一起，零输入差分电压对应 0x800 输出。

差分模式下的 ADC 灵敏度比单端模式下小两倍。

当 ADC 配置为差分模式时，V_{CMIN} 电压应该(V_{REFP})/2 电压内。

内部通道只采用单端模式。

下图为单端/差分通道的示意图：

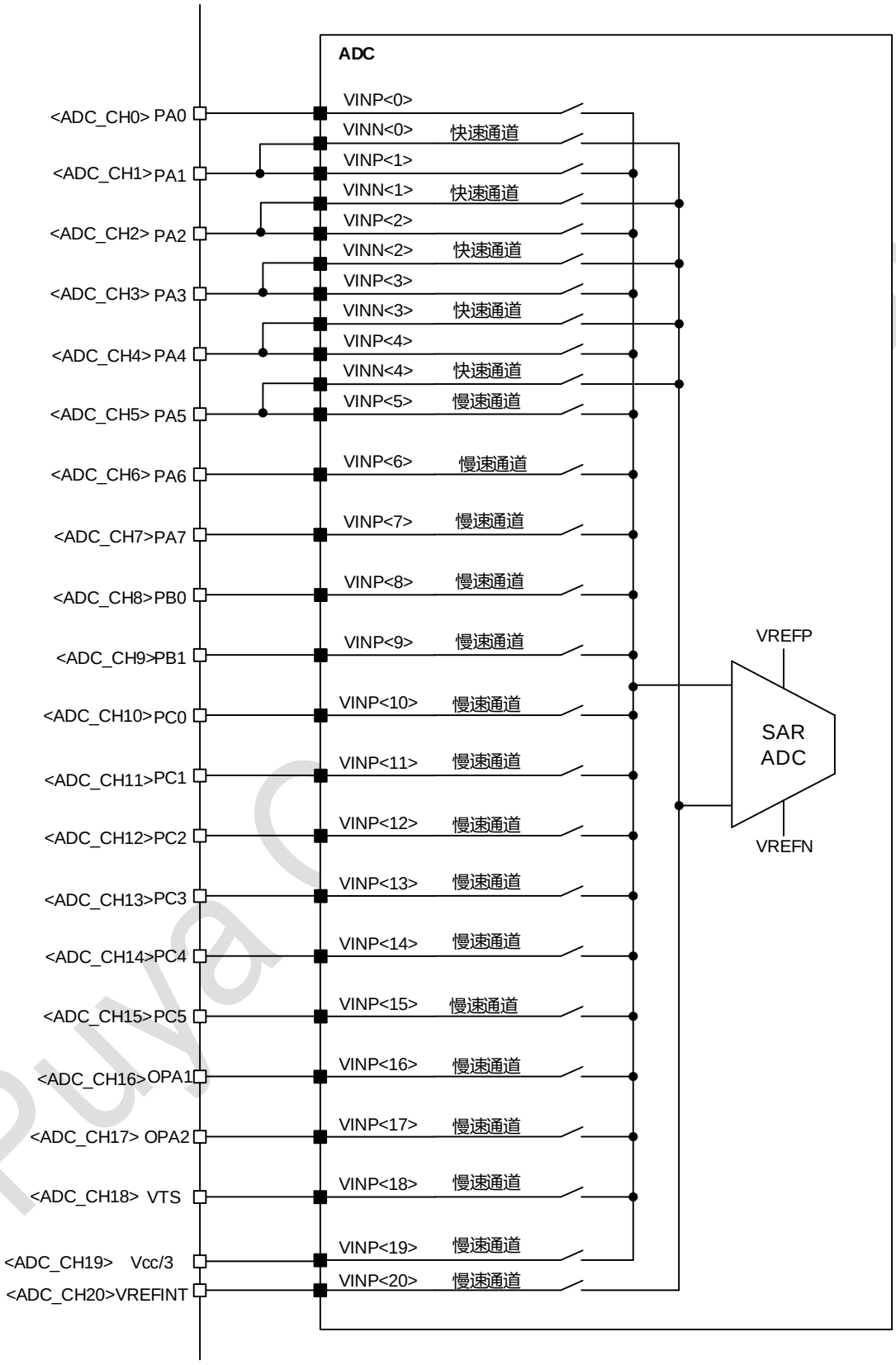


图 13-2 ADC 通道连接关系

注：通道配置通过 ADC_SQRx 或 ADC_JSQR 设置。

支持 5 对差分对：

- PA0<->PA1, ADC_CCR.DIFF_EN=1 且通道选择 PA0 时，PA1 自动作为 INN 端
- PA1<->PA2，ADC_CCR.DIFF_EN=1 且通道选择 PA1 时，PA2 自动作为 INN 端
- PA2<->PA3，ADC_CCR.DIFF_EN=1 且通道选择 PA2 时，PA3 自动作为 INN 端
- PA3<->PA4，ADC_CCR.DIFF_EN=1 且通道选择 PA3 时，PA4 自动作为 INN 端
- PA4<->PA5，ADC_CCR.DIFF_EN=1 且通道选择 PA4 时，PA5 自动作为 INN 端

警告：

在差分输入模式下配置通道“i”时，其负端输入电压 VINN[i]已连接到另一个通道，因此，此通道不再可用于单端模式或差分模式，不能配置转换。

13.3.3 ADC 校准

该 ADC 具有校准功能(软件启动)，用户可通过软件使能 ADCAL 进行校准。在校准期间， ADC 计算一个用于 ADC 内部的校准因子。在 ADC 校准期间、未完成校准前，应用不能使用 ADC 模块。在使用 ADC 转换前，建议用户进行校准操作。校准用于消除芯片和芯片之间的，由于工艺变化引起的失调误差和失配误差。

ADC 软件校准

软件设置 ADCAL=1 可启动校准，校准只能在 ADC 未使能时（ADEN=0)启动，且仅支持选择 PCLK 作为 ADC 的时钟。

当校准完成后，ADCAL 被硬件清 0。

当 ADC 的工作条件发生改变时（V_{CC} 改变是失调误差和失配误差的主要因素，温度改变次之），推荐进行再次校准操作。

通过设置 ADC_CFGR2 寄存器中 CALNUM[2:0]来配置校准过程的重复次数，并对结果进行平均以得到更精确的校准结果。

内部校准寄存器可通过设置 ADC_CR 的 RSTCAL 复位，该位由硬件清除置 0。在校准寄存器被初始化后(即 RSTCAL 置 1 后)，该位即被清除。

校准的软件操作过程：

1. 确认 ADC_CR.ADEN=0
2. 设置 ADC_CR.RSTCAL（该步骤是可选的）；
3. 设置校准次数 ADC_CFGR2.CALNUM（该步骤是可选的）；
4. 设置 ADC_CR.ADCAL=1
5. 等待 ADC_SR.EOCAL=1

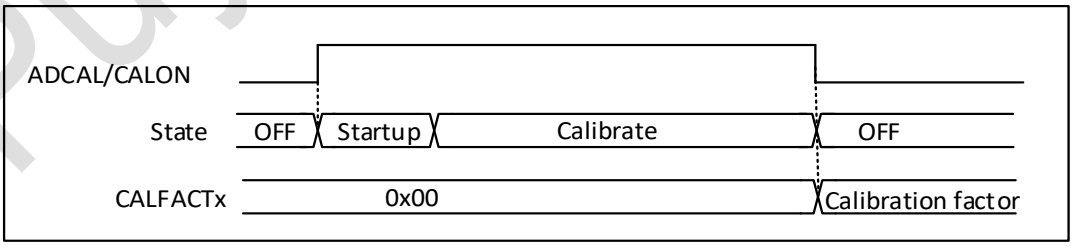


图 13-3 ADC 校准

- 软件写校准因子

1. 确保 ADEN=1，且没有转换正在进行，同时 ADCAL=0。
2. 使能 WRVLD，FACTSEL[4:0]选择写入哪个电容，设置 WCALFACT [8:0]写入校准因子。
3. ADEN 使能，进行 AD 转换时，校准因子自动注入模拟 ADC。

- 软件读校准因子

1. 确保 ADEN=1，且没有转换正在进行，同时 ADCAL=0。
2. 使能 RDVLD，FACTSEL[4:0]选择写入哪个电容。
3. 读 RCALFACT [8:0]校准因子。

13.3.4 ADC 开-关控制(ADEN,ADDIS,ADRDY)

芯片上电复位后，ADC 模块不使能，处于不工作状态(ADEN=0)。

ADEN 位用于控制位开启或关闭 ADC。ADC 启用后，在开始准确转换之前，ADC 需要一个稳定时间 t_{STAB}，如下图。有两个控制位用于启用或禁用 ADC：

- ADEN=1 时，启用 ADC。当 ADC 准备好进行操作时，标志 ADRDY 将被设置。
- ADDIS=1 时，禁用 ADC。一旦模拟 ADC 实际上被禁用，硬件会自动清除 ADEN 和 ADDIS。

ADC 规则转换由设置 ADSTART 来启动，根据 EXTEN 的配置，可以立即开始转换（软件启动）或者等待硬件触发后开始转换。

ADC 注入转换由设置 JADSTART 来启动，根据 JEXTEN 的配置，可以立即开始转换（软件启动）或者等待硬件触发后开始转换。

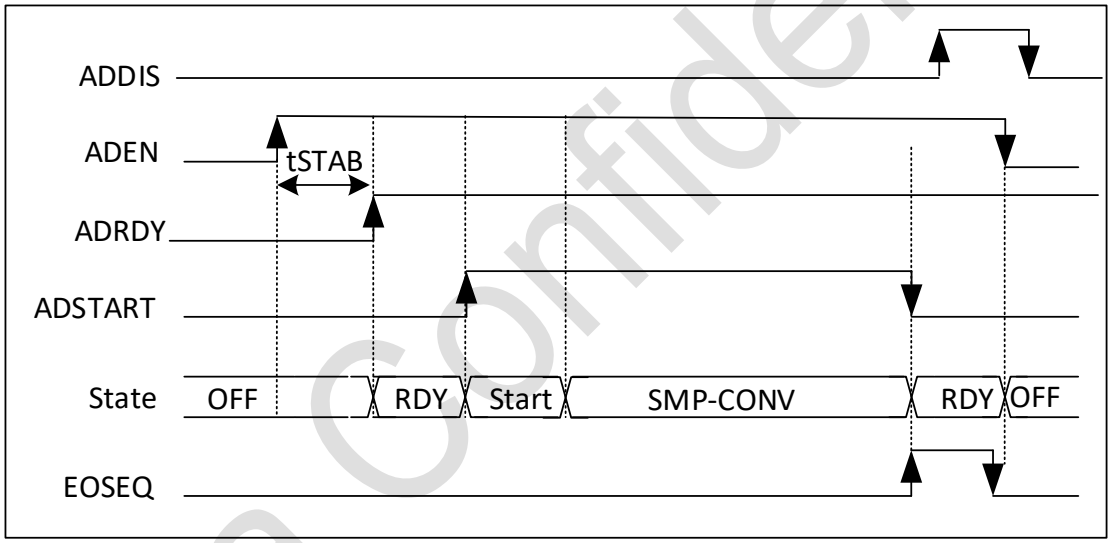


图 13-4 使能/禁止 ADC

- 软件启用 ADC 的步骤

步骤：

1. 通过写入'1'清除 ADC_ISR 寄存器中的 ADRDY 位。
2. 设置 ADEN=1 或 ADCAL 以启用 ADC。(ADCAL 启用 ADC 进行校准)
3. 等待直到 ADRDY=1（在 ADC 上电结束后，ADRDY 将被设置）。这可以通过关联的中断来完成（设置 ADRDYIE=1）。(ADCAL 启用 ADC 进行校准后，硬件自清零)
4. 通过写入'1'清除 ADC_ISR 寄存器中的 ADRDY 位（可选步骤）。

注意事项：

1. ADEN 或 ADCAL 从 0 置 1 后，SARADC 模拟需要等待 3us 的时间。

- 软件禁用 ADC 的步骤

1. 检查 ADSTART 和 JADSTART 是否均为 0，确保没有正在进行的转换。如有需要，通过设置 ADSTP=1 和 JADSTP=1 停止任何正在进行的规则和注入转换，然后等待直到 ADSTP=0 和 JADSTP=0。

2. 设置 ADDIS=1 以禁用 ADC。
3. 等待直至 ADEN=0，即模拟 ADC 实际上被禁用（一旦 ADEN=0，ADDIS 将自动重置）。

13.3.5 转换启动(ADSTART，JADSTART)

软件通过设置 ADSTART=1 启动 ADC 规则转换。

当 ADSTART 设置，则：

- 当 EXTEN=0x0(软件触发) 时，立即开始
- 当 EXTEN 不等于 00 时，在选中的规则硬件触发源的下一个有效边沿开始

软件通过设置 JADSTART=1 启动 ADC 注入转换。

设置 JADSTART 后，转换开始:

- 当 JEXTEN[1:0] = 00(软件触发)时，立即开始
- 如果 JEXTEN[1:0]不等于 00 时，在选中的注入硬件触发源的下一个活动边沿开始

注意:在自动注入模式下(JAUTO=1)，使用 ADSTART 位启动规则转换，然后是自动注入转换(JADSTART 必须保持清除)。

ADSTART 和 JADSTART 位也用于说明目前 ADC 转换操作是否正在进行。

ADSTART 位由硬件清除：

- 当软件触发（CONT=0，EXTSEL=0x0）：

1. 在规则转换序列结束时（EOS 信号产生），硬件清零 ADSTART。
2. 若 DISCEN=1，在子序列转换结束后，硬件清零 ADSTTART。

- 在所有的情况下(CONT=X，EXTSEL=X),在软件设置 ADSTP 后，硬件清零 ADSTART。

注：在连续模式（CONT=1）下，ADSTART 位不能在 EOS 产生时硬件清除，因为序列会自动重新开始转换。当硬件触发时(CONT=0 且 EXTSEL ≠0x00)，则当 EOS 标志设置后，ADSTART 不会被硬件清 0，避免软件重新设置 ADSTART 位且要确保无硬件触发事件错过。

JADSTART 由硬件清除:

1. 当软件触发（JEXTSEL = 0x0）：
   1) 在注入转换序列结束(JEOS 产生)时，硬件清零 JADSTART。
   2) 若 JDISCEN = 1，子序列转换结束，硬件清零 JADSTART。
2. 在所有情况下(JEXTSEL=x)，软件设置 JADSTP 后，硬件清零 JADSTART。

注意：当选择软件触发时，如果 EOC 标志未清零，则不应设置 ADSTART 位。

13.3.6 停止正在进行的转换（ADSTP, JADSTP)

通过设置 ADC_CR 寄存器中的 ADSTP=1 可以停止当前正在进行的规则转换,通过设置 ADC_CR 寄存器中的 JADSTP=1 可以停止当前正在进行的注入转换。

停止转换将重置正在进行的 ADC 操作。然后可以重新配置 ADC(例如:改变通道选择或触发方式)，为新的 AD 转换做好准备。

请注意，在规则转换仍在运行时可以停止注入转换，反之亦然。例如，这允许在规则转换仍在运行时重新配置注入的转换序列和触发方式(反之亦然)。

当 ADSTP 位由软件设置时，任何正在进行的规则转换被中止，转换结果被丢弃(ADC_DR 寄存器不更新当前转换)。

当 JADSTP 位由软件设置时，任何正在进行的注入转换将被中止，转换结果将被丢弃(ADC_JDRy 寄存器不更新当前转换)。

扫描序列也被中止和重置(这意味着重新启动 ADC 将启动一个新序列)。

一旦 ADC 重置完成，ADSTP/ADSTART(在规则转换的情况下)，或 JADSTP/JADSTART(在注入转换的情况下)被硬件清除。若软件需要检测 ADC 是否重置，可以通过轮询 ADSTART(或 JADSTART)是否被重置。

注意:在自动注入模式下(JAUTO=1)，设置 ADSTP 位会同时终止规则转换和注入转换(不能使用 JADSTP)。

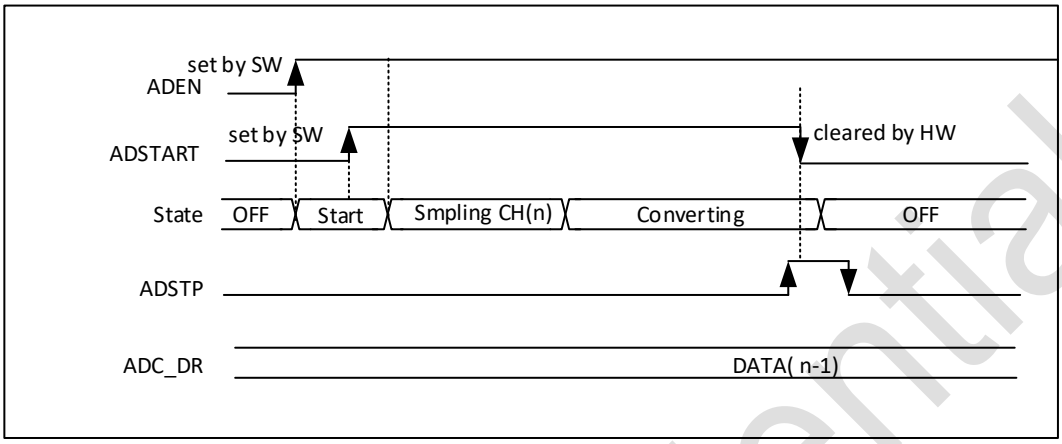


图 13-5 ADSTP 停止规则转换

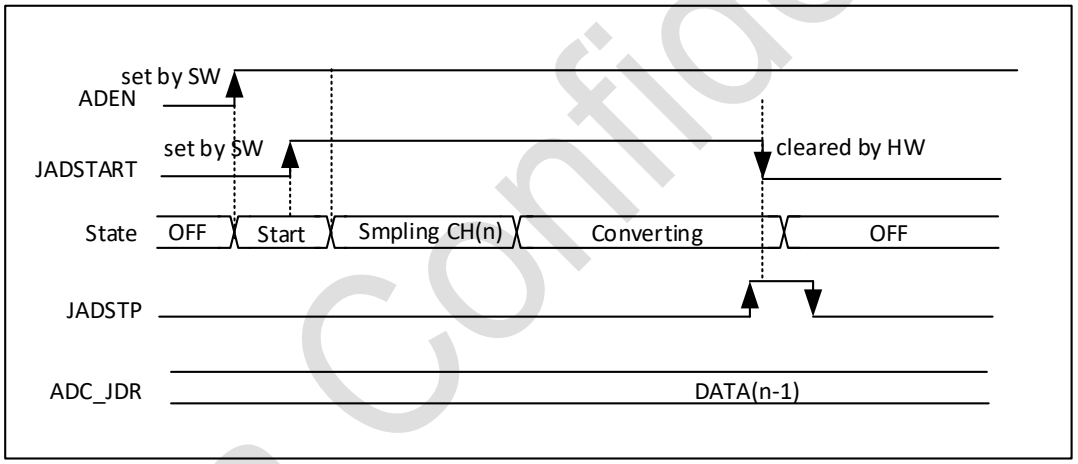


图 13-6 JADSTP 停止注入转换

13.3.7 ADC 时序(单次/连续模式，硬件/软件触发)

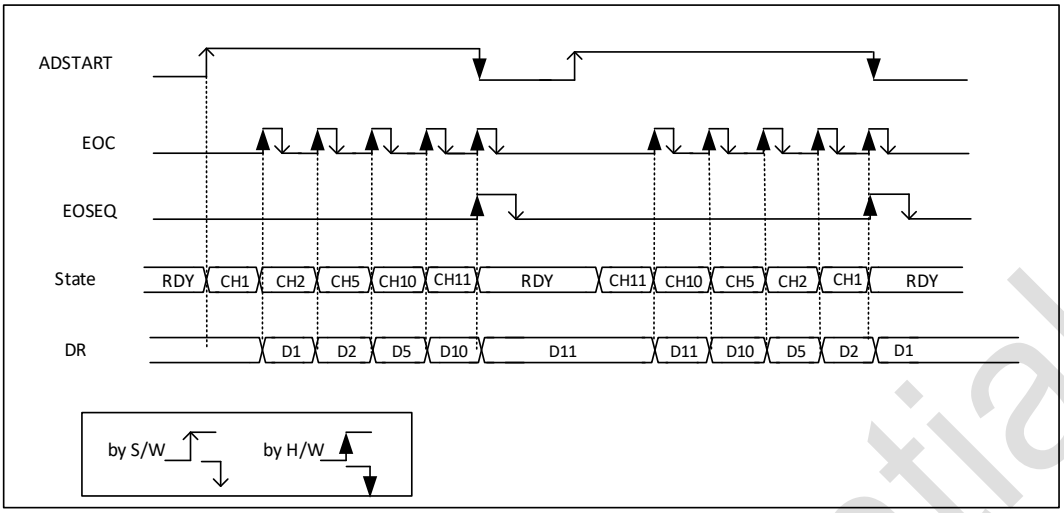


图 13-7 单次模式，软件触发

1. EXTEN=0x0，CONT=0
2. SQ1~5 分别配置为通道 1，2，5，10，11, AUTDLY=0

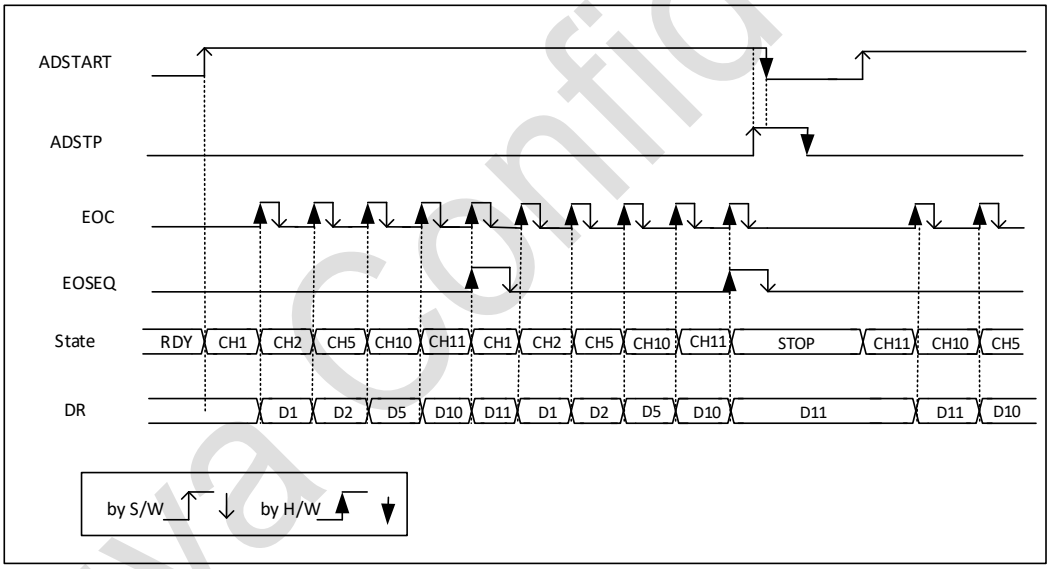


图 13-8 连续模式，软件触发

1. EXTEN=0x0，CONT=1,
2. SQ1~5 分别配置为通道 1，2，5，10，11，AUTDLY=0

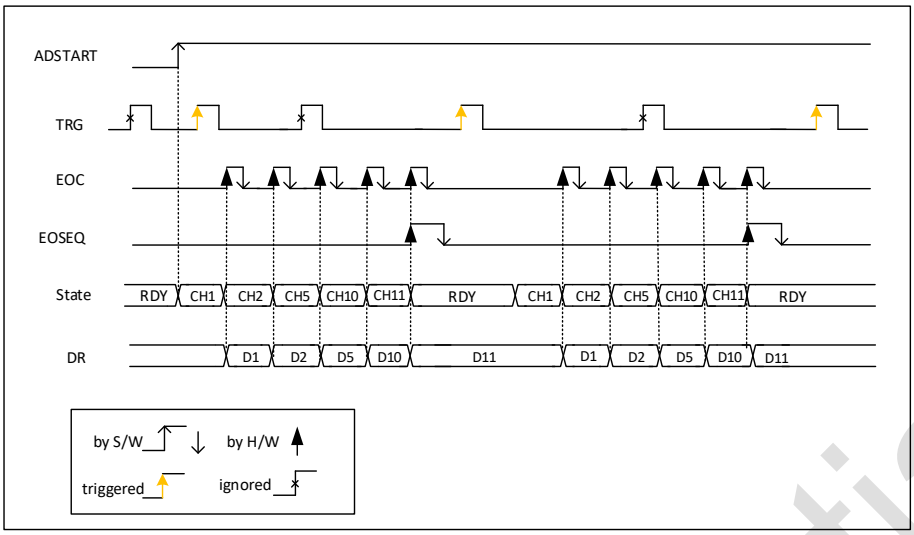


图 13-9 单次转换一个序列，硬件触发

1. EXTSEL=TRGx，EXTEN=0x1（上升沿），CONT=0
2. SQ1~5 分别配置为通道 1，2，5，10，11，AUTDLY=0

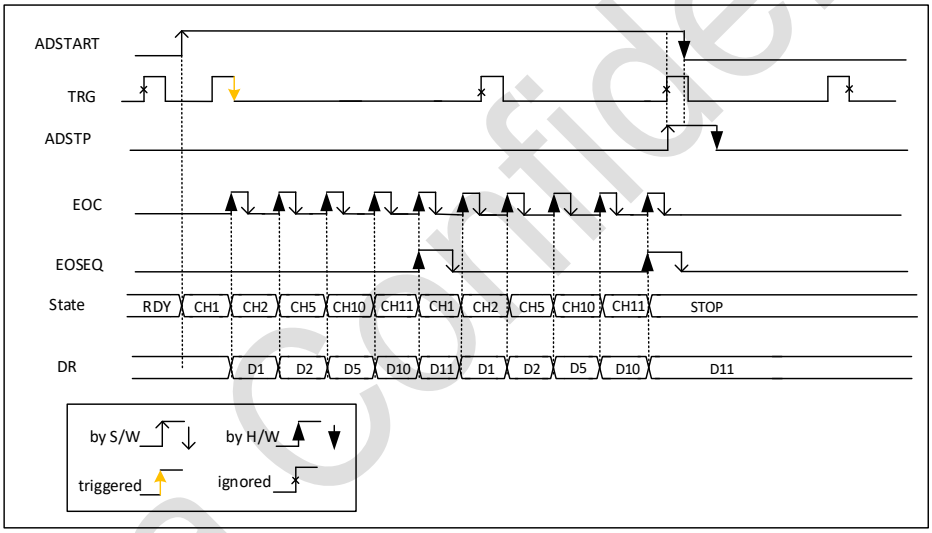


图 13-10 连续转换一个序列，硬件触发

1. EXTSEL=TRGx, EXTEN=0x2（下降沿），CONT=1
2. SQ1~5 分别配置为通道 1，2，5，10，11，AUTDLY =0

13.3.8 写 ADC 控制位的限制

软件写 ADC 寄存器控制位，需要遵循以下原则：

1. ADC 已启用且没有待处理的禁能 ADC 请求（即 ADEN 必须为 1 且 ADDIS 为 0），软件才被允许写入 ADC_CR 寄存器中的 ADSTART、JADSTART 和 ADDIS 控制位。
2. 当 ADC 已启用，可能正在转换，并且没有待处理的禁能 ADC 请求（即 ADSTART 或 JADSTART 必须等于 1 且 ADDIS 为 0）时，软件才被允许写入 ADC_CR 寄存器中的 ADSTP 或 JADSTP 控制位。
3. 在 ADEN=1 且 ADSTART 为 0 时，软件才能配置 ADDIS=1；
4. 软件写 ADC_CFGR、ADC_SMPRx、ADC_SQRy、ADC_JDRy、ADC_OFRy、ADC_IER、ADC_TR、ADC_JSQR 等寄存器，需要遵循以下原则：

1. 规则序列相关寄存器，在 ADEN=1 且 ADSTART=0 时配置；

2. 注入序列相关寄存器，在 ADEN=1 且 JADSTART=0 时配置；

注意：硬件并未提供写操作保护措施，若违反上述原则配置寄存器，ADC 的行为可能会进入未知状态。为了从这种状况中恢复，需要将 ADC 禁用（将 ADEN 清零，并同时清零 ADC_CR 寄存器中的所有位）。

13.3.9 ADC 时钟

ADC 具有双时钟域架构，ADC 时钟（ADC_CLK)独立于 APB 时钟（PCLK）。ADC_CLK 可由两种时钟源产生，如下图所示：

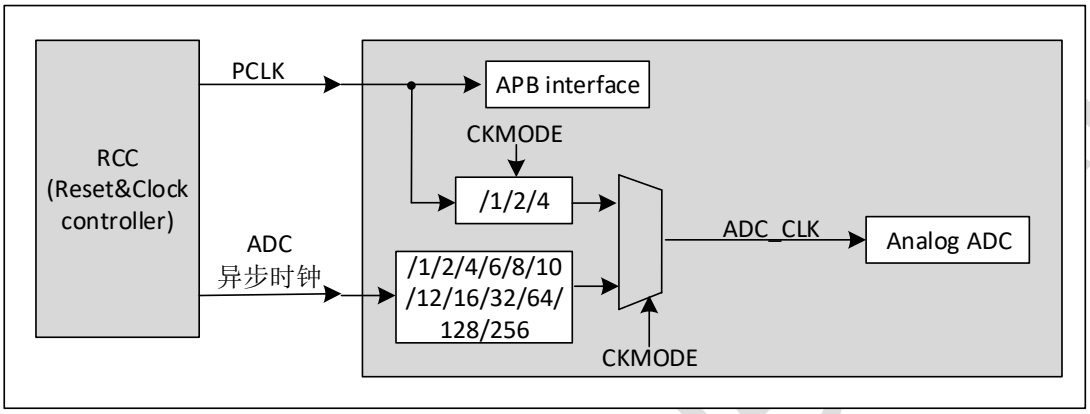


图 13-11 ADC 时钟结构

模拟 ADC 的输入时钟(ADC_CLK)有两个可选时钟源：

1. ADC_CLK 可选 “ADC 异步时钟”时钟源，该时钟源独立于 APB 时钟，与 APB 时钟异步。要选择此时钟方案，必须将 ADC_CCR 寄存器的 CKMODE[1:0] 位复位。
2. ADC 时钟可选 PCLK(ADC 总线接口时钟)或其分频时钟源，该时钟源与 APB 时钟同步，可选时钟分频因子（1、 2 或 4，具体根据 CKMODE[1:0] 位而定）。要选择此时钟方案， ADC_CCR 寄存器的 CKMODE[1:0] 位不得为“00”。

在选项 a) 中，对 ADC_CCR 寄存器中的 PRESC[3:0] 位进行编程时，生成的 ADC 时钟最后会除以预分频系数（1、2、4、6、8、10、12、16、32、64、128、256）。

选项 a) 的优点在于无论选择哪种 APB 时钟方案，都可以达到最大 ADC 时钟频率。

注意： 选择 CKMODE[1:0]=11（PCLK 1 分频）时，用户必须确保 PCLK 的占空比为 50 %。可通过选择占空比为 50 % 的系统时钟并在 RCC 不配置 APB 预分频器来实现（请参见复位和时钟控制器部分）。

13.3.10 ADC 通道选择(SQRx, JSQR)

ADC 有 16 个外部通道和 5 个内部通道，其中内部通道为：

- ADC_CH16 连接 OPA1
- ADC_CH17 连接 OPA2
- ADC_CH18 连接 VTS(温度传感器),
- ADC_CH19 连接 Vcc/3
- ADC_CH20 连接 V_{REFINT}

表 13-1 ADC 内部通道

ADC		
通道	**Name**	**Source**
ADC_CH16	ADC_INTER_CH0	OPA1_VOUT
ADC_CH17	ADC_INTER_CH1	OPA2_VOUT

ADC_CH18	ADC_INTER_CH2	VTS(温度传感器)
ADC_CH19	ADC_INTER_CH3	Vcc/3
ADC_CH20	ADC_INTER_CH4	V_{REFINT}

ADC 通道选择可通过 ADC_SQRx 或 ADC_JSQR 配置，配置值和通道对应关系见 ADC 连接关系。

ADC 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。所有通道都可以按注入或规则通道组进行转换。

- 一个规则转换组最多由 16 个转换构成。转换序列的规则通道及其转换顺序在 ADC_SQRx 寄存器中选择。规则转换组中的序列长度必须写入 L[3:0] 。
- 一个注入转换组最多由 4 个转换构成。注入通道及其转换顺序在 ADC_JSQR 寄存器中选择。注入转换组中的序列长度必须写入 JL[1:0] 位。

在规则转换期间，不得修改 ADC_SQRx 寄存器，必须先通过写入 ADSTP=1 来停止 ADC 的规则转换，再修改 ADC_SQRx 寄存器。

13.3.11 动态低功耗特性

自动延迟转换模式(AUTDLY)

自动延迟转换对于简化软件以及优化应用程序的性能非常有用，因为程序在低频运行时可能会遇到 ADC 过载的风险。

当在 ADC_CFGR 寄存器中设置 AUTDLY 为 1 时，只有在当前 ADC 转换数据处理完成后才开始新的转换：

- 对于规则转换：一旦 ADC_DR 寄存器被读取或 EOC 位被清除，才开始新的转换
- 对于注入转换：当 JEOS 位被清除时，才开始新的转换

这是一种自适应 ADC 速度和自适应系统读取 ADC 数据速度的方法。

在每个规则转换(无论 DISCEN=0 或 1)后，ADC 进行等待状态，等待 AD 转换数据处理完成。

在每个注入转换序列(无论 JDISCEN=0 或 1)后，ADC 进行等待状态，等待 AD 转换数据处理完成。

注：在注入序列的每次转换结束后不会进行等待状态，而是在整个注入序列结束后进入等待状态。

在等待状态时，将忽略在此期间发生的硬件触发事件(针对同一组转换)。

注：对于软件触发，在等待状态期间，仍然可以通过置位 ADSTART 或 JADSTART 来重新启动转换：在启动新的转换之前，由软件读取数据以避免数据覆盖丢失。

在不同组的转换之间不插入等待状态(规则转换后跟注入转换，反之亦然):

- 如果在规则转换的等待状态期间发生注入触发，则注入转换立即开始。
- 一旦注入序列完成，ADC 在启动新的规则转换之前等待前一个规则转换的等待状态结束。

在自动注入模式(JAUTO=1)中，行为略有不同，只有在注入转换序列的等待状态结束时(当 JEOS 被清除时)，新的规则转换才能开始。这是为了确保软件可以在开始一个新序列之前读取给定序列的所有数据。

要在 JAUTO=1，CONT=1，AUTDLY =1 下停止转换，请执行以下步骤:

1. 等待 JEOS=1(不再重新启动转换)
2. 清零 JEOS,
3. 设置 ADSTP = 1
4. 读取规则数据。

如果不遵守此过程，在上例的 CONT=1 的情况下，如在设置 ADSTP 后清零 JEOS (步骤 2 和 3 颠倒)，可以重新启动新的规则序列。

在 AUTDLY 模式中，如果硬件规则触发事件发生在已经在进行的规则序列期间，或者发生在序列最后一次规则转换之后的等待状态期间，则忽略该事件。如果发生在此等待状态之后，则认为该触发事件是挂起的。

在 AUTDLY 模式中，如果硬件注入的触发事件发生在已经在进行的注入序列期间，或者发生在序列的最后一次注入转换之后的等待状态期间，则忽略该事件。

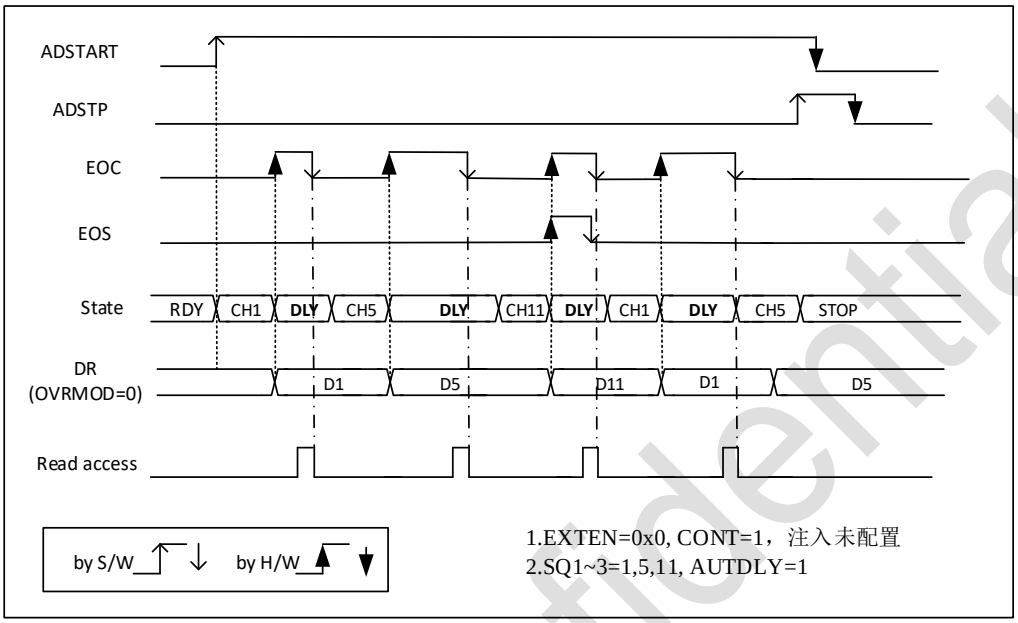


图 13-12 AUTDLY 模式

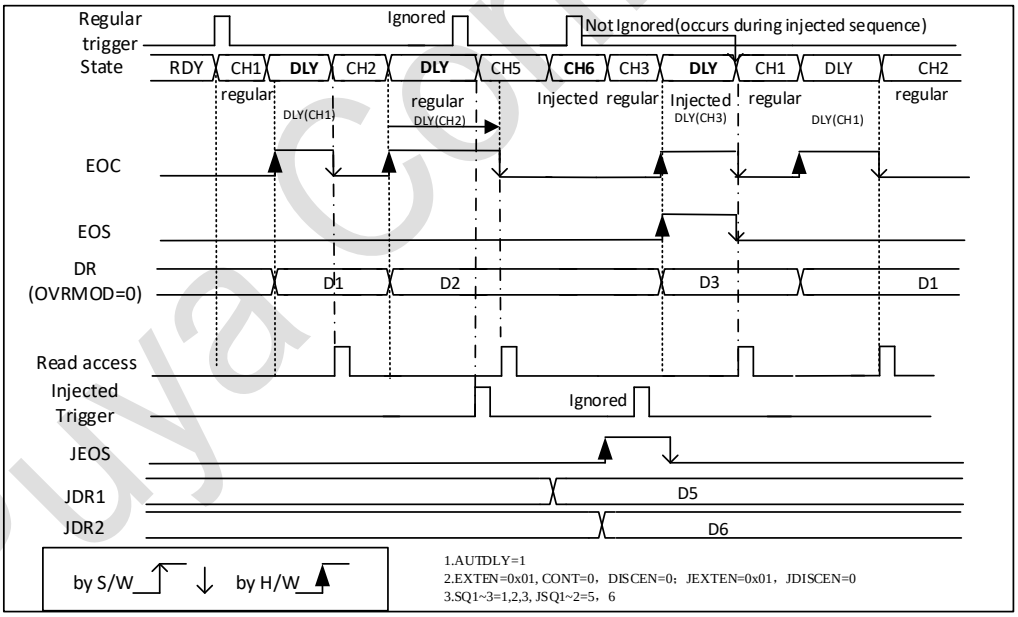


图 13-13 规则序列被注入触发中断

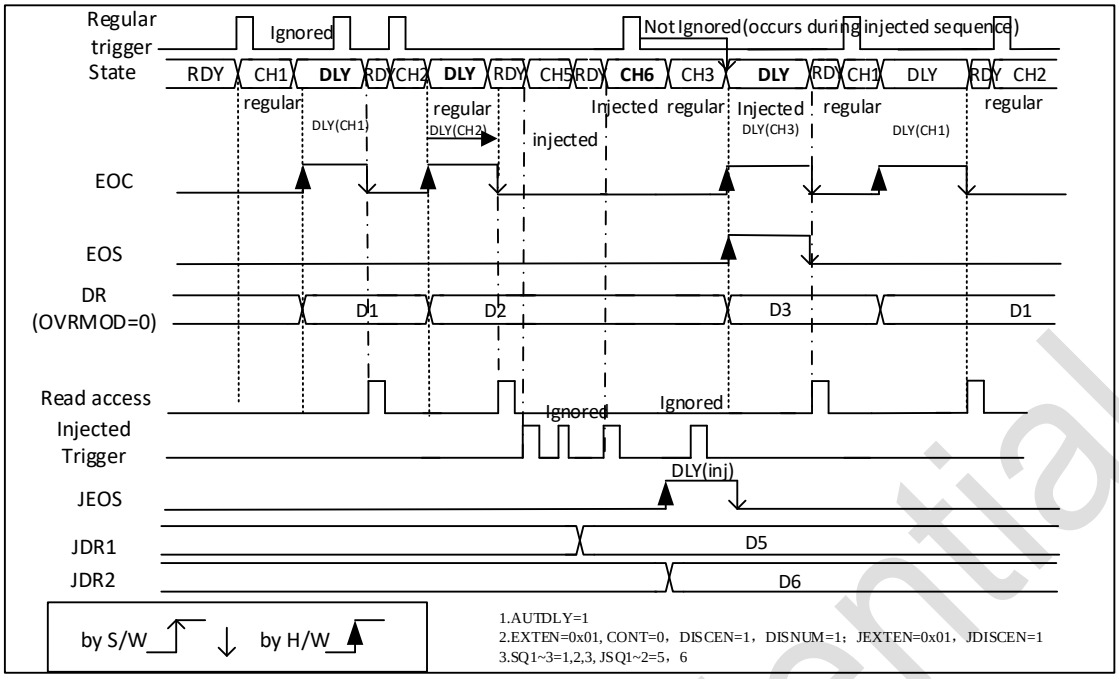


图 13-14 规则序列被注入触发中断

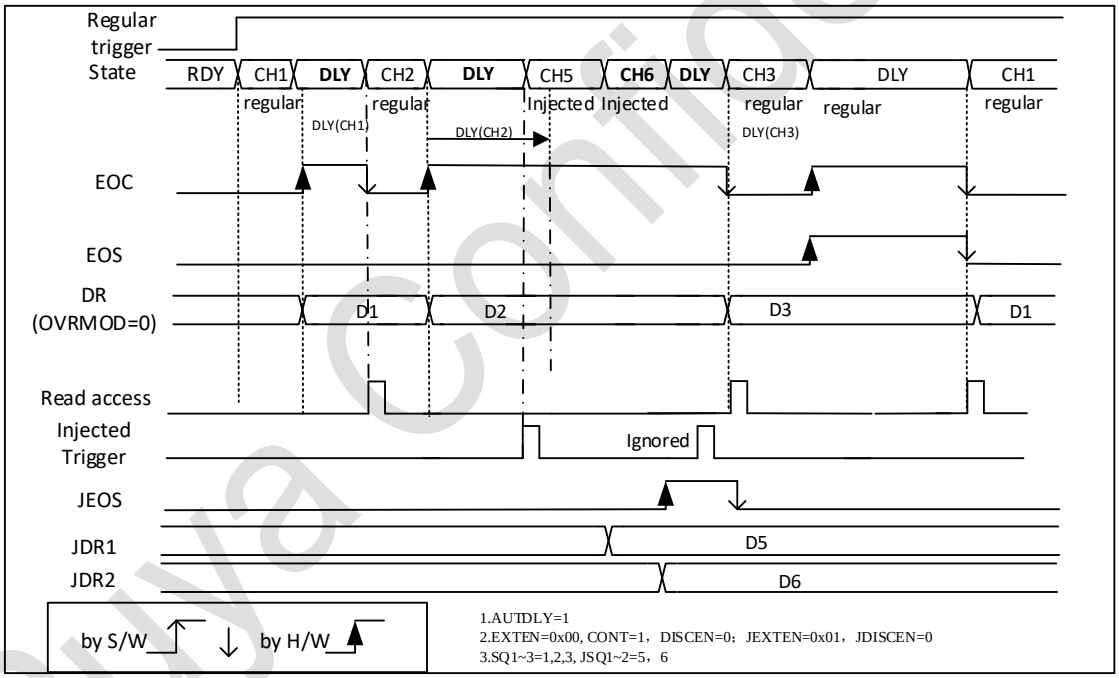


图 13-15 规则序列被注入触发中断

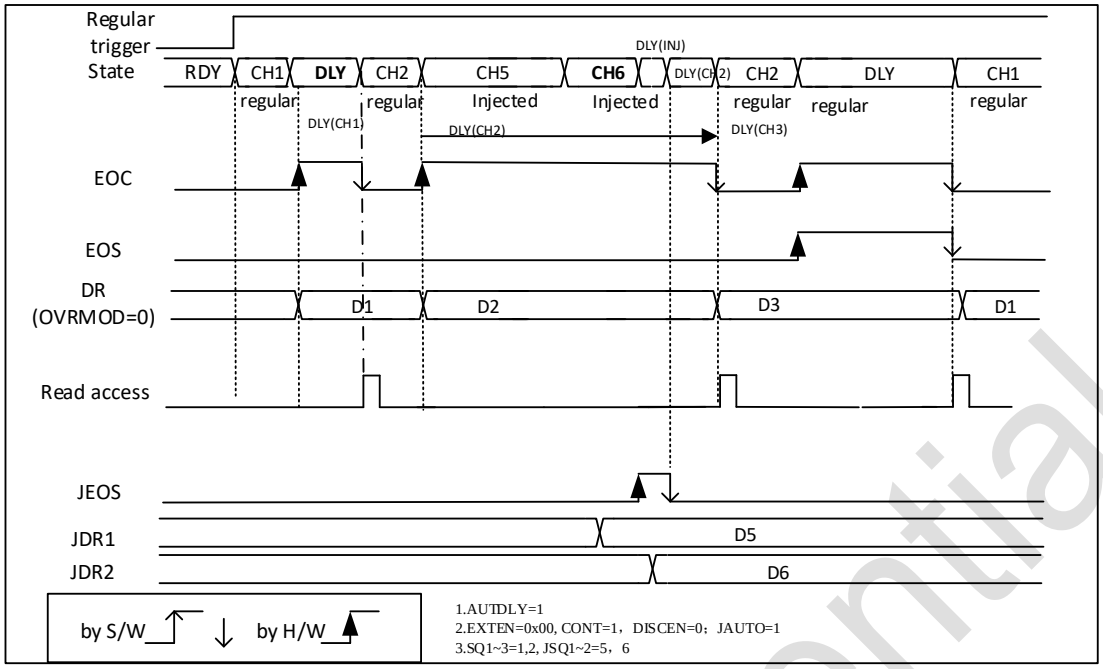


图 13-16 规则序列被注入触发中断

13.3.12 转换模式

单次转换模式(CONT=0)

单次转换模式下，ADC 执行一次序列转换，转换所有被选的通道。当 ADC_CFGR 寄存器中的 CONT=0，DISCEN=0 时，ADC 为单次转换模式。

该模式可由下述方法启动：

- 在 ADC_CR 寄存器中设置 ADSTART 位（规则转换）
- 在 ADC_CR 寄存器中设置 JADSTART 位（注入转换）
- ADSTART=1 且硬件触发事件（规则转换）
- JADSTART=1 且硬件触发事件（注入转换）

规则转换，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_DR 中。
- EOC(转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断。

规则转换，所有通道序列转换完成后：

- EOS(序列结束) 标志置位
- 若 EOSIE 位置位则产生一个中断

注入转换，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_JDRx 中
- JEOC(转换结束标志)标志置位
- 若 JEOCIE 位置位则产生一个中断。

注入转换，所有通道序列转换完成后：

- JEOS(序列结束) 标志置位
- 若 JEOSIE 位置位则产生一个中断

转换结束后，ADC 转换停止直到新的触发事件或 ADSTART/JADSTART 重新置位。

若转换单一通道，则可配置序列长度为 1。

连续转换模式(CONT=1)

该模式仅针对规则转换。

在连续转换模式中，当软件或硬件触发事件产生，ADC 执行一个序列转换。转换所有的通道且自动重新开始执行相同的序列转换。

当寄存器 ADC_CFGR 中的 CONT=1 时，ADC 选择为连续转换模式。

ADC 转换可由下述两种方法启动：

- 在 ADC_CR 寄存器中设置 ADSTART 位
- ADSTART=1 且硬件触发事件硬件触发事件

在序列通道的转换期间，每次转换完成后：

- 转换的数据结果存放到 16 位寄存器 ADC_DR 中
- EOC (转换结束标志)标志置位
- 若 EOCIE 位置位则产生一个中断

通道序列转换完成后：

- EOS (序列结束)标志置位
- 若 EOSIE 位置位则产生一个中断

一次序列转换结束后，ADC 立即重新转换相同的序列通道。

注：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

ADC 不能同时处于间断转换模式和连续转换模式（DISCEN=1，CONT=1)

注入通道不支持连续转换模式，唯一的例外是当连续自动注入时（JAUTO=1，CONT=1）。

间断转换模式(DISCEN=1/JDISCEN=1)

间断转换模式时，ADC 将选择的一个序列分成子序列(序列长度 1-8)，通过多次外部触发子序列，完成整个序列转换，该模式能够运行在规则组和注入组。DISCNUM 规定了规则通道子序列长度，ADC_SQRx 规定了一个规则序列所有转换通道，其中 L[3:0]位规定了该序列长度。 ADC_JSQR 寄存器规定了一个注入组序列所有转换通道，其中 JL[1:0]位规定了该序列长度。采样结束，则产生 EOSMP，如设置了 EOSMPIE，则产生中断。

注意：注入通道组的子序列为 1.该模式下禁止 DISCEN 与 JDISCEN 同时使能.

- 规则组

一个外部触发信号启动 ADC_SQRx 描述通道 n(n≤8)次转换，直到此序列所有转换完成为止。总序列长度由 L[3:0]定义。

例如：

n=3，被转换的通道 = 0, 1, 2, 3, 6, 7, 9, 10

第一次触发：转换的序列为 0，1，2，每个通道转换结束产生 EOC 事件。

第二次触发：转换的序列为 3，6，7，每个通道转换结束产生 EOC 事件。

第三次触发：转换的序列为 9，10，每个通道转换结束产生 EOC 事件。序列结束产生 EOS 事件。

第四次触发：转换的序列 0，1，2，每个通道转换结束产生 EOC 事件。

注：当一规则组以间断模式转换时，转换序列结束后不自动从头开始。当所有子组被转换完成，下一次触发启动第一个子组的转换。在上面的例子中，第四次触发重新转换第一子组的通道 0，1 和 2。

在软件同时使能连续转换模式和间断转换模式时，硬件会屏蔽这种情况(DISCEN =1，CONT=1)

- 注入组

一个外部触发信号启动 ADC_JSQR 描述通道 1 次转换，直到序列中所有转换完成为止。总序列长度 JL[1:0]位定义。

例如：

n=1，被转换的通道 = 1，2，3

第一次触发：通道 1 被转换，每个通道转换结束产生 JEOC 事件。

第二次触发：通道 2 被转换，每个通道转换结束产生 JEOC 事件。

第三次触发：通道 3 被转换，每个通道转换结束产生 JEOC 事件。序列结束产生 JEOS 事件。

第四次触发：通道 1 被转换，每个通道转换结束产生 JEOC 事件。

注：1 当完成所有注入通道转换，下个触发启动第 1 个注入通道的转换。在上述例子中，第四个触发重新转换第 1 个注入通道 1。

2 不能同时使用自动注入和间断模式。

3 必须避免同时为规则和注入组设置间断模式。间断模式只能作用于一组转换。

13.3.13 注入通道管理

注入通道外部触发优先级高于规则通道外部触发，即注入通道外部触发可以中断正在进行中的规则通道转换。注入通道有两种中断方式：触发注入和自动注入。

触发注入

触发注入模式，JAUTO 必须为 0。

- 通过外部触发或通过设置 ADC_CR 寄存器中的 ADSTART 位启动一组规则通道的转换。
- 如果发生外部注入触发，或者在转换一组规则通道期间设置了 ADC_CR 寄存器中的 JADSTART 位，则当前转换重置并启动注入通道序列（所有注入通道都转换一次)。
- 然后，从上次中断的规则转换中恢复规则组通道的规则转换。
- 如果在注入转换期间发生规则触发事件，则不会中断注入转换，而是在注入序列的末尾执行规则序列。

注意:当使用触发注入时，必须确保触发事件之间的间隔长于注入序列。例如，如果序列长度为 30 个 ADC 时钟周期，则触发器之间的最小间隔必须为 31 个 ADC 时钟周期。

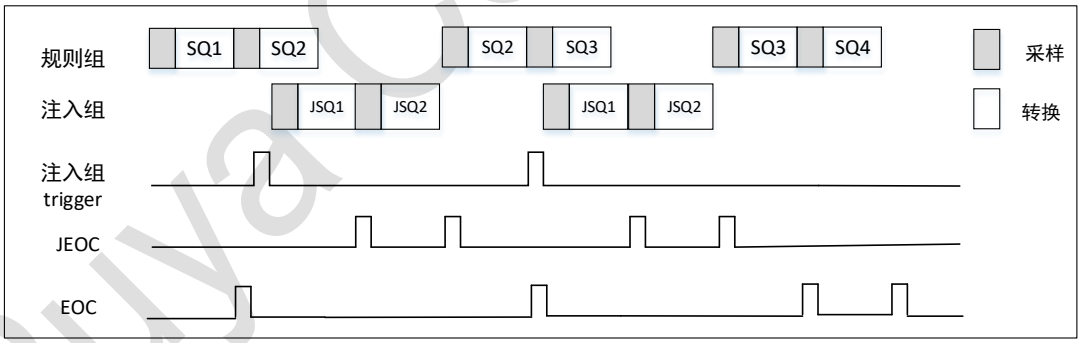


图 13-17 ADC 触发注入示意图

自动注入

如果设置了 ADC_CFGR 寄存器中的 JAUTO 位，则注入组中的通道在规则通道组之后自动转换。这可以用来转换在 ADC_SQRy 和 ADC_JSQR 中编程的最多 20 个转换通道（规则通道最多 16 个注入最多 4 个）。

在这种模式下，将 ADC_CR 寄存器中的 ADSTART 位设置为 1 启动规则转换，然后是注入转换(JADSTART 必须保持清除)。设置 ADSTP 位会终止规则转换和注入转换(不能使用 JADSTP 位)。

在此模式下，必须禁用注入通道上的外部触发。

如果除了 JAUTO 位之外还设置了 CONT 位，则连续转换规则通道，然后是注入通道。

注意:不可能同时使用自动注入和不连续模式。

当 DMA 在 JAUTO 模式下用于读取规则转换序列数据时，需要将其编程为循环模式(在 DMA_CCRx 寄存器中设置 CIRC 位)。如果 CIRC 位复位(单发模式)，在 DMA 传输完成事件发生时，JAUTO 序列将停止。

13.3.14 模拟看门狗

三个 AWD 模拟看门狗监测某些通道是否保持在配置的电压范围内（窗口）

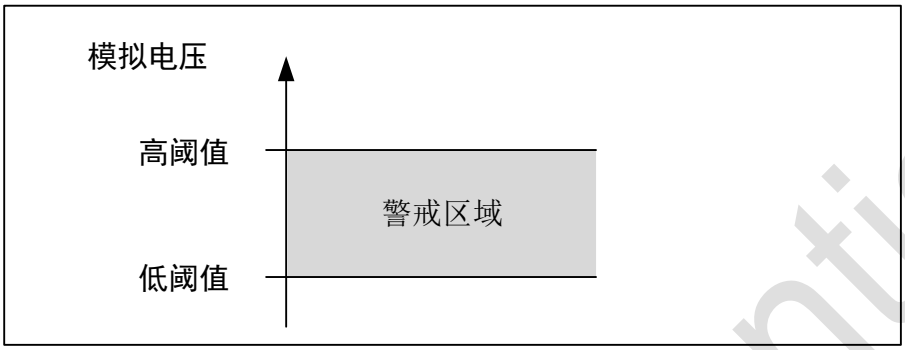


图 13-18 模拟看门狗保护区域

AWDx 标志和中断

通过在 ADC_IER 寄存器（x=1,2,3）中设置 AWDxIE，可以为 3 个模拟看门狗中的每一个启用中断。AWDx（x=1,2,3）标志由软件写 1 来清除。在数据对齐之前将 ADC 转换结果与低阈值和高阈值进行比较。

模拟看门狗 1 描述

通过设置的 AWD1EN 位启用 AWD 模拟看门狗 1。该看门狗监视一个选定的通道或所有启用的通道是否保持在配置的电压范围（窗口）内。

下表显示了应如何配置 ADC_CFGR 寄存器以启用一个或多个通道上的模拟看门狗。

表 13-2 模拟看门狗通道选择

模拟看门狗保护通道	AWD1SGL bit	AWD1EN bit	JAWD1EN bit
None	x	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有规则和注入通道	0	1	1
单个注入通道	1	0	1
单个规则通道	1	1	0
单个规则/注入通道	1	1	1

通过 AWD1CH[4:0]选择看门狗保护通道，这些被保护通道必须在规则或注入序列进行转换。

如果 ADC 转换的模拟电压低于较低阈值或高于较高阈值，则设置 AWD1 模拟看门狗状态位。这些阈值被写在模拟看门狗 1 的 ADC_TR1 寄存器的位 HT1[11:0]和 LT1[11:0]中。当转换分辨率小于 12 位的数据时（根据位 RES[1:0]），阈值的 LSB 必须保持清除，因为内部比较总是在完整的 12 位原始转换数据上执行（左对齐）。

下表描述了如何针对模拟看门狗 1 的所有可能的分辨率执行比较。

表 13-3 模拟看门狗 1 比较

分辨率(RES[1:0])	模拟看门狗比较：		描述
	原始转换左对齐	阈值	
00：12bit	DATA[11:0]	LT1[11:0]和 HT1[11:0]	-
01：10bit	DATA[11:2]，00	LT1[11:0]和 HT1[11:0]	用户必须将 LT1[1:0] 和 HT1[1:0]配置为 00
10：8bit	DATA[11:4]，0000	LT1[11:0]和 HT1[11:0]	用户必须将 LT1[3:0] 和 HT1[3:0]配置为 00
11：6bit	DATA[11:6]，000000	LT1[11:0]和 HT1[11:0]	用户必须将 LT1[5:0] 和 HT1[5:0]配置为 00

有关模拟看门狗比较的更多详细信息，请参阅“增益补偿”一节。

看门狗 1 的模拟看门狗滤波器

当 ADC 仅配置一个输入通道（不允许在扫描模式下选择多个通道）时，可以通过 ADC_TR1 寄存器配置有效的 ADC 转换数据间隔：

- 当转换后的数据属于 ADC_TR1 中定义的间隔时，会生成 DMA 请求。
- 否则，RDATA 寄存器在每次转换时都会更新。如果数据超出范围，比 ADC_TR1 的 AWDFILT 位中指定的值高出若干倍，则设置 AWDx 标志，并发出相应的中断。

看门狗 2 和 3 的描述

第 2 和第 3 模拟看门狗更灵活，可以通过对 AWDxCH[20:0]（x=2，3）中的相应位进行编程来保护几个选定的通道。当 AWDxCH[20:0]（x=2,3）的任何位被设置时，相应的看门狗被启用。它们被限制为 8 位的分辨率，并且只有阈值的 8 个 MSB 可以被编程为 HTx[7:0]和 LTx[7:0]。下表描述了如何针对所有可能的分辨率执行比较。

表 13-4 模拟看门狗 2 和 3 比较

分辨率(RES[1:0])	模拟看门狗比较：		描述
	原始转换左对齐	阈值	
00：12bit	DATA[11:4]	LTx[7:0]和 HTx[7:0]	数据[3:0]与比较无关
01：10bit	DATA[11:4]	LTx[7:0]和 HTx[7:0]	数据[3:2]与比较无关
10：8bit	DATA[11:4]	LTx[7:0]和 HTx[7:0]	-
11：6bit	DATA[11:6]，00	LTx[7:0]和 HTx[7:0]	用户必须将 LTx[1:0] 和 HTx[1:0]配置为 00

注：有关模拟看门狗比较的更多详细信息，请参阅“增益补偿”一节。

ADC_AWDx_OUT 信号输出产生

每个模拟看门狗都与内部硬件信号 ADC_AWDx_OUT（x=看门狗编号）相关联，该信号直接连接到一些片上 timer 的 ETR 输入（外部触发）。请参阅片上 timer 部分，了解如何选择 ADC_AWDx_OUT 信号作为 ETR。

启用相关的模拟看门狗时，ADC_AWDx_OUT 被激活：

- 当被保护转换超出编程阈值时，设置 ADC_AWDx_OUT。
- 当接下来的受保护转换结束且值在编程设定的阈值范围内时，ADC_AWDx_OUT 会复位（但若后续受保护转换超出仍设定阈值，则该信号保持为 1）。
- 当禁用 ADC 时（当设置 ADDIS=0 时），ADC_AWDx_OUT 被复位。请注意，停止规则或注入的转换（设置 ADSTP=1 或 JADSTP=1）对 ADC_AWDx_OUT 的生成没有影响。

注意：AWDx 标志由硬件设置，由软件复位：AWDx 标志对 ADC_AWDx_OUT 的生成没有影响（例如：如果软件未清除 AWDx，AWDx 标志保持为 1，ADC_AWDx_OUT 可以切换）。

ADC_AWDx_OUT 信号由 PCLK 域生成。AWD 比较在每次 ADC 转换结束时执行。规则、注入 ADC_AWDx_OUT 同理。

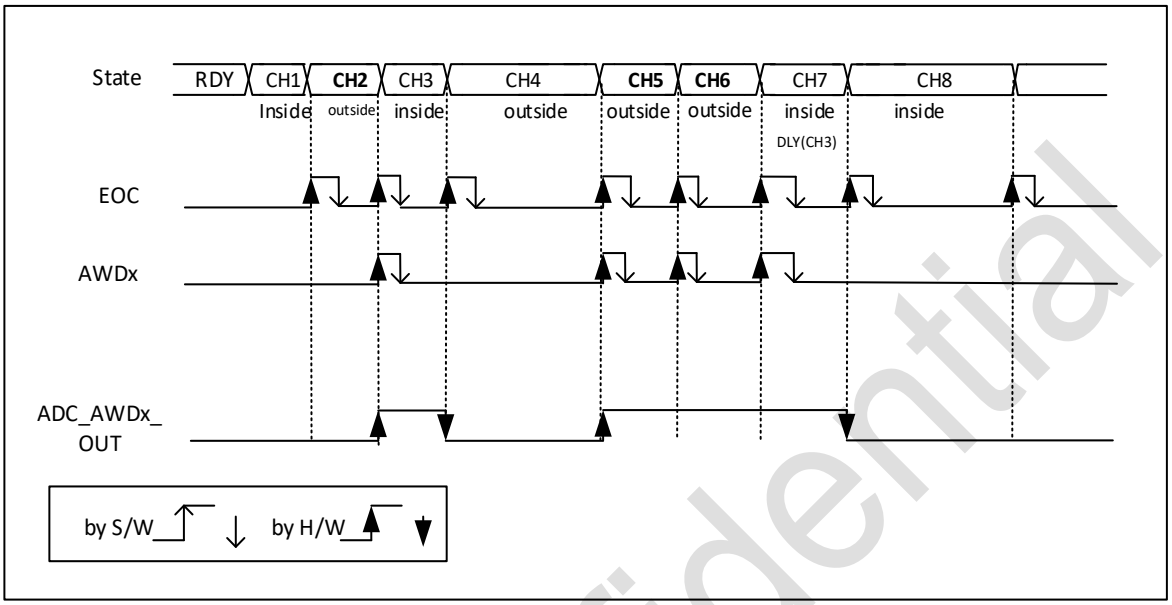


图 13-19 选择所有通道模拟看门狗

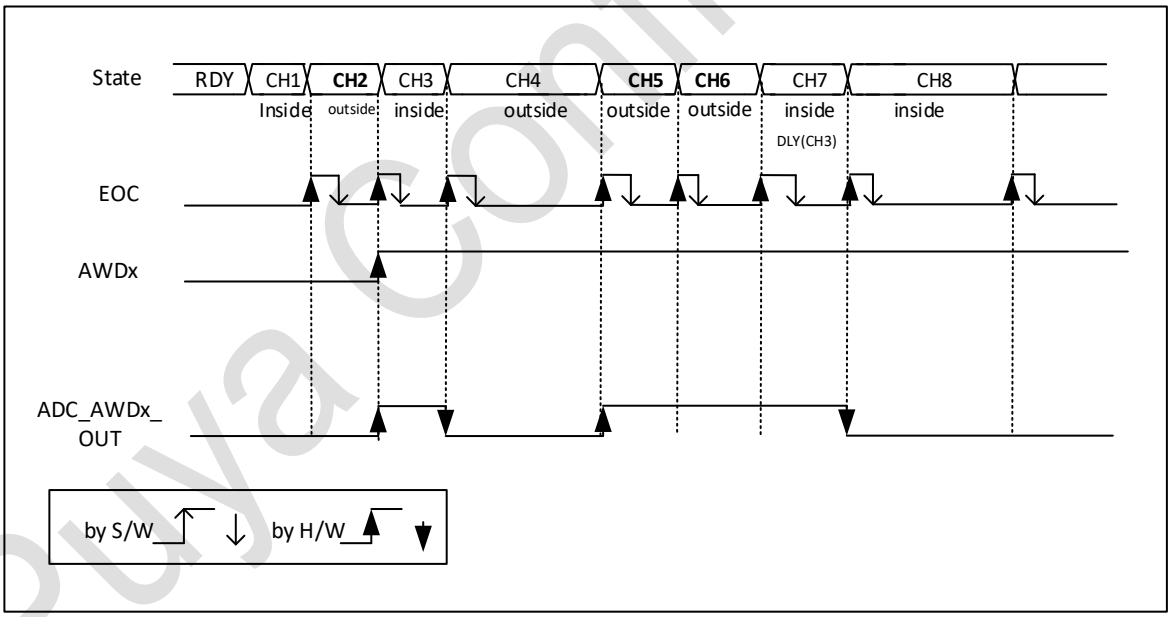


图 13-20 软件不清除 AWDx 的情况

带增益和偏移补偿的模拟看门狗

当增益和偏移补偿被启用时，模拟看门狗在补偿数据之后比较阈值。

注意：启用偏移补偿时（ADC_OFRy 寄存器中的 OFFSETy_EN 设置为 1），数据过载或欠载可能导致错误的看门狗结果。启用过载保护时（在 ADC_OFRy 中将 SATEN 设置为 1），看门狗会提供正确的结果。但是这会阻止使用有符号数据格式。

13.3.15 数据处理

数据对齐

ADC_CFGR 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式，可选择左对齐和右对齐两种方式。转换的数据值已经减去了 ADC_OFRy 寄存器中自定义的偏移量，因此结果可以是一个负值。SEXT 位表示扩展的符号值。

特殊情况：左对齐时，除分辨率为 6 位外，其他分辨率数据以半字为基础对齐。分辨率 6 位时，数据以字节为基础进行对齐。

注意：在过采样模式下不支持左对齐。当 ROVSE 和/或 JOVSE 位被设置时，ALIGN 位值被忽略，ADC 仅提供右对齐数据。

表 13-5 数据对齐和分辨率

ALIGN	RES	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0x0	0x0				DATA[11:0]											
	0x1	0x0						DATA[9:0]									
	0x2	0x0								DATA[7:0]							
	0x3	0x0										DATA[5:0]					
1	0x0	DATA[11:0]												0x0			
	0x1	DATA[9:0]										0x0					
	0x2	DATA[7:0]								0x0							
	0x3	0x0								DATA[5:0]						0x0	

偏移(offset)

通过设置 ADC_OFRy 寄存器中 OFFSETy_EN = 1，可以将偏移 y（y = 1、2、3、4）施加到通道。应用可通过 ADC_OFRy 寄存器的位 OFFSETy_CH[4:0]选择偏移的通道。在这种情况下，转换结果减去 OFFSETy[11:0]中的用户定义的偏移量，结果可能是负值，因此读取的数据是有符号的，并且 SEXT 位表示扩展的符号值。

注意：过采样模式下不支持偏移校正。设置 ROVSE 和/或 JOVSE 位时，ADC_OFRy 寄存器中 OFFSETy_EN 位的值被忽略（视为复位）。

下表描述了如何针对模拟看门狗 1 的所有可能的分辨率执行比较。

表 13-6 Offset 计算值 VS 分辨率

分辨率 (RES[1:0])	原始转换结果和 offset 减法		Result	描述
	原始转换结果 左对齐	Offset		
00:12bit	DATA[11:0]	OFFSET[11:0]	有符号 12bit 数	-
01:10bit	{DATA[11:2],00}	OFFSET[11:0]	有符号 10bit 数	用户必须配置 OFFSET[1:0]=00
10:8bit	{DATA[11:4],0000}	OFFSET[11:0]	有符号 8bit 数	用户必须配置 OFFSET[3:0]=0000
11:6bit	{DATA[11:6],000000}	OFFSET[11:0]	有符号 8bit 数	用户必须配置 OFFSET[5:0]=000000

从与通道“i”对应的 ADC_DR（规则通道）或 ADC_JDRy（注入通道，y=1,2,3,4）读数据时：

- 如果对应通道的偏移之一被使能（位 OFFSETy_EN = 1），则读数据时有符号的。
- 如果此通道的四个偏移均未启用，则读数据无符号。

下图所示为有符号数和无符号数数据对齐。

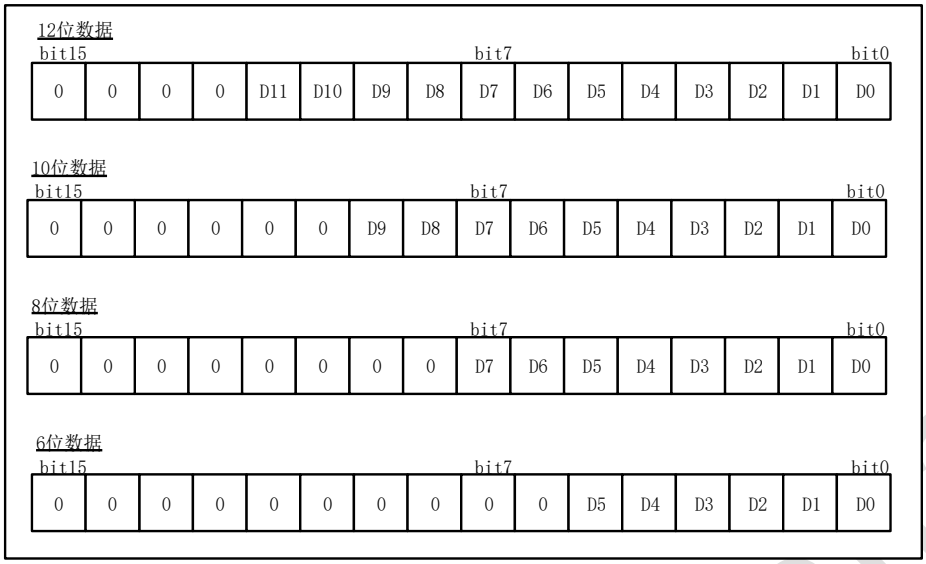


图 13-21 右对齐（offset 未使能，无符号数）

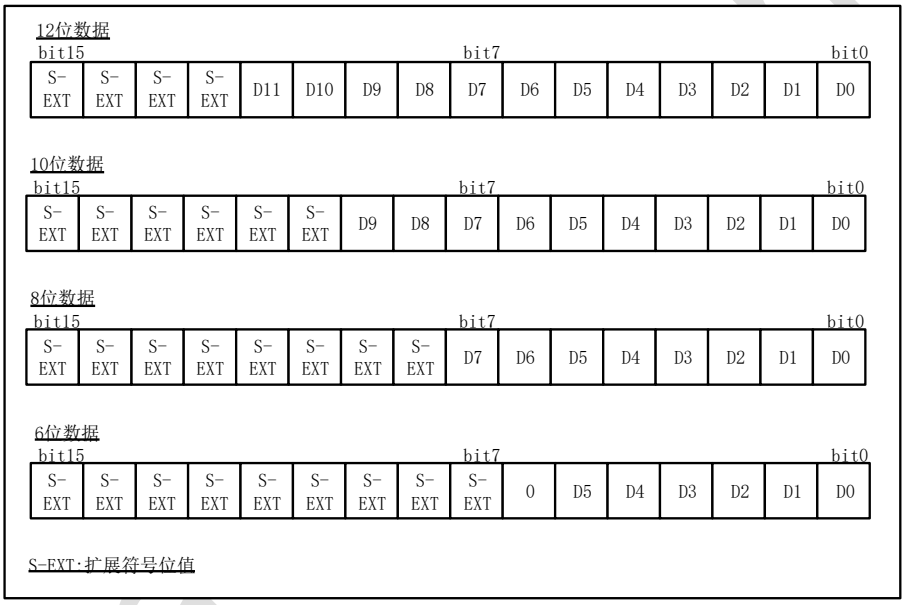


图 13-22 右对齐（offset 使能，有符号数）

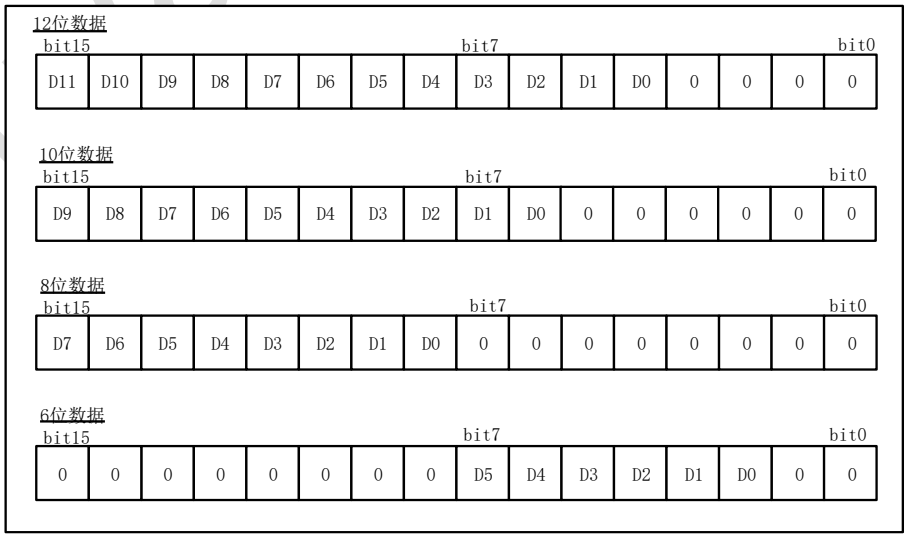


图 13-23 左对齐（offset 未使能，无符号数）

图 13-24 左对齐（offset 使能，有符号数）

增益补偿

当设置 ADC_CFGR2 寄存器 GCOMP 位时，对所有转换后的数据进行增益补偿。每次转换后，数据将使用以下公式进行计算。

DATA = (DATA(adc result) × GCOMPCOEFF)/4095

由于 GCOMPCOEFF 范围为 0-16383，因此实际增益补偿因子范围 0-3.999756。

在将结果数据存储在 RDATA 或 JDATAx 寄存器中之前，将计算 LSB−1 值将四舍五入，并将误差降至最低。

增益补偿对于过采样也是有效的。当增益补偿用于过采样模式时，在累加和右移操作之后执行增益计算，以最小化功耗（增益计算仅进行一次，而不是每次转换）。

偏移(offset)补偿

当在偏移操作期间在 ADC_OFRy 寄存器中设置 SATEN 位时，数据是无符号的。所有偏移数据在 0x000 处饱和（在 12 位分辨率）。当 OFFSETPOS 位被设置时，偏移方向是正的，并且数据在 0xFFF 饱和（在 12 位分辨率）。在 8 位分辨率中，数据分别在 0x00 和 0xFF 处饱和。

在偏移和增益补偿之后，对无符号数执行模拟看门狗比较。为了正确的看门狗操作，偏移补偿后的数据必须为无符号格式（ADC_OFRy 寄存器中的 SATEN 位设置为 1）。

ADC 过载(OVR, OVRMOD)

ADC 过载标志(OVR) 是指一个缓冲区过载事件，当转换好的数据未被 CPU 或 DMA 及时读取时，另一个转换数据已经有效时，就发生了 ADC 过载。

若 EOC 还为‘1’的情况下，这时一个新的转换已经完成，那么 ADC_ISR 寄存器中的 OVR 标志会被置位，表明 ADC 过载。当 ADC_IER 寄存器中的 OVRIE 置位时，产生一个 ADC 过载中断。

当过载事件发生时，ADC 会继续转换除非软件停止并复位这个序列转换，设置 ADC_CR 寄存器中的 ADSTP 为 1 来停止 ADC 转换。OVR 标志可用软件写 1 清除。

当发生过载事件时，可通过对 ADC_CFGR 寄存器中的 OVRMOD 位设置 ADC 数据寄存器中的数据是被保持还是被覆盖：

- OVRMOD=0

一个过载事件发生时，数据寄存器的不被覆盖：之前的数据被保持，新的转换数据丢弃。若 OVR 保持为 1，则后续的转换会被执行但结果都被丢弃。

- OVRMOD=1

数据寄存器用最新转换结果覆盖，先前未读的数据丢失。若 OVR 保持为 1，则后续的转换被执行且 ADC_DR 寄存器存放着最新转换的结果值。

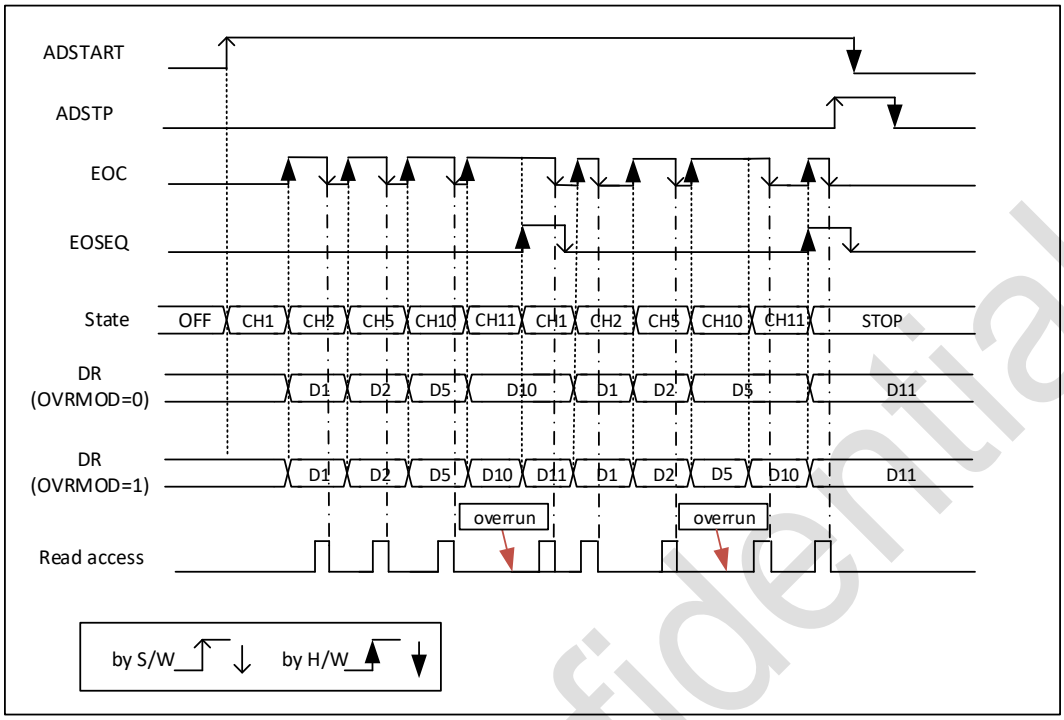


图 13-25 过载示意图

非 DMA 模式

若 ADC 的转换足够慢，转换序列可由软件来控制。这种情况下，软件应用 EOC 标志及其关联的中断去处理每个转换数据。当每次转换结束时，在 ADC_ISR 寄存器中的 EOC 位置位，此时可读 ADC_DR 寄存器的转换值。ADC_CFGR 寄存器中的 OVRMOD 位可配为 0 来管理过载事件。

13.3.16 非过载非 DMA 模式

如需要转换一个或多个通道且不用每次转换结果都要读取，这种情况下， OVRMOD 位必须置为 1 且软件应忽略 OVR 标志。当 OVRMOD=1 时，过载事件不能阻止 ADC 继续转换且 ADC_DR 寄存器中的数据一直为最新转换数据。

13.3.17 DMA 模式

因为所有通道的转换结果数据存放到一个单一的数据寄存器中，故当转换通道超过 1 个时用 DMA 方式会更有效。 这样可以避免丢失存在 ADC_DR 寄存器中的转换结果。当 DMA 模式开启时 (ADC_CFGR 寄存器中的 DMAEN =1)，每次转换结束时都会产生一个 DMA 请求。这样就允许把在 ADC_DR 寄存器中的转换数据传送到软件指定的目标地址中。

尽管如此，因 DMA 不能够及时为 DMA 请求服务而产生的过载(OVR=1) 时，ADC 就会停止产生 DMA 请求且新的转换数据也不会再由 DMA 进行传输(当 OVR=0 时，会继续传输)。这也可以认为所有传输到 RAM 中的数据都是有效的(因无效的数据再也不传输了)。

根据 OVRMOD 位的配置，ADC_DR 寄存器中的数据可选择为：保持或覆盖。

DMA 传输请求会被阻止直到软件清除 OVR 位。

有两种不同的 DMA 模式，其取决于 ADC_CFGR 寄存器中的 DMACFG 位的配置：

- DMA 一次模式 (one shot mode)(DMACFG=0)

  当 DMA 编程用于传输固定长度的数据时，可选用该模式。

- DMA 循环模式 (circular mode)(DMACFG=1)

  当 DMA 编程为循环模式时，可选用该模式。

DMA 单次模式(DMACFG=0)

在这种模式下，ADC 在每次转换的数据有效时产生一次 DMA 请求。一旦 DMA 已达到最后一个 DMA 传输时，即使 ADC 转换已再次启动，ADC 停止产生 DMA 请求。(产生 DMA_EOT 中断时，下一次的 ADC 转换有可能已开始)

当 DMA 传输完成 (配置在 DMA 控制器中的所有传输已经完成):

- ADC 数据寄存器的内容冻结
- 任何进行中的转换终止，且结果丢弃
- 不给 DMA 控制器发出新的 DMA 请求。假如仍有 ADC 转换启动，这种方式可避免产生一个 ADC 过载错误
- ADC 扫描序列停止并复位
- DMA 停止

DMA 连续模式(DMACFG=1)

在这种模式下，即使 DMA 完成最后一个数据传输，ADC 也会在每次转换的数据有效时产生一次 DMA 请求。这允许 DMA 配置为循环模式来处理连续模拟输入数据流。

13.3.18 过采样器

过采样单元执行数据预处理以卸载 CPU。它能够处理多个转换，并将它们平均为单个数据(增大数据宽度，最高可达 16 位)。

过采样器提供了以下公式，其中 N 和 M 可以调整:

$$Result = \frac{1}{M} \times \sum_{n=0}^{n=N-1} Conversion(t_n)$$

过采样器允许通过硬件执行以下功能：平均、降低数据速率、提高信噪比、基本滤波。

过采样率 N 是通过 ADC_CFGR2 寄存器中的 OVSR[2:0]位定义的，可在 2x 到 256x 的范围内调节。除法系数 M 由高达 8 位的右位移位组成，并且使用 ADC_CFGR2 寄存器中的 OVSS[3:0]位来定义。

求和单元可以产生高达 20 位的结果 (256x 12 位结果) , 该结果首先右移。然后，在最终传输到 ADC_DR 数据寄存器之前被截断为 16 个最低有效位，将移位留下的最低有效位四舍五入到最接近的值。

注意：如果移位后的中间结果超过 16 位，则该结果被原样截断而不饱和。

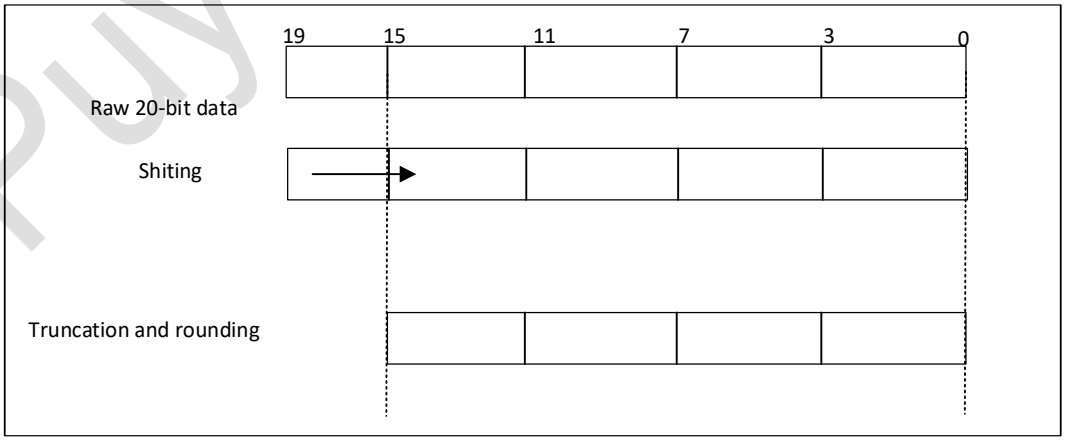


图 13-26 20bit 截断为 16bit

下图给出了处理的数字示例，从原始的 20 位累加数据到最终的 16 位结果。

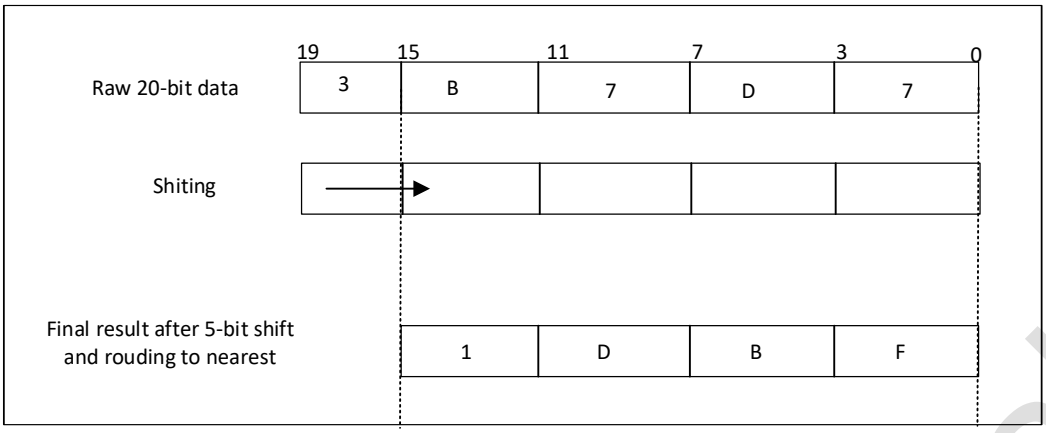


图 13-27 具有 5 位移位和舍入的数值示例

下表给出了原始转换数据为 0xFFF 的各种 N 和 M 组合的数据格式。

表 13-7 最大输出结果与 N 和 M 的关系（灰色单元格表示截断）

Over sampling ratio	Max Raw data	No-shift OVSS = 0000	1-bit shift OVSS = 0001	2-bit shift OVSS = 0010	3-bit shift OVSS = 0011	4-bit shift OVSS = 0100	5-bit shift OVSS = 0101	6-bit shift OVSS = 0110	7-bit shift OVSS = 0111	8-bit shift OVSS = 1000
2x	0x1FFE	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040	0x020
4x	0x3FFC	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080	0x0040
8x	0x7FF8	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100	0x0080
16x	0xFFF0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200	0x0100
32x	0x1FFE0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400	0x0200
64x	0x3FFC0	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800	0x0400
128x	0x7FF80	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF	0x0800
256x	0xFFF00	0xFF00	0xFF80	0xFFC0	0xFFE0	0xFFF0	0x7FF8	0x3FFC	0x1FFE	0x0FFF

过采样模式下的转换时序没有变化：在整个过采样序列中，采样时间保持相等。每 N 个转换提供一个新数据，通过 N x TCONV = N x（tSMPL+tSAR）等效延迟。标志设置如下：

- 采样阶段结束（EOSMP）设置在每个采样阶段之后
- 当过采样结果可用时，每 N 次转换发生一次转换结束（EOC）
- 序列结束（EOS）发生在过采样数据序列完成之后（即在 N x 总序列长度转换之后）

过采样时支持的 ADC 操作模式（单个 ADC 模式）

在过采样模式下，大多数 ADC 操作模式都保持不变：

- Single 模式/连续模式转换
- 通过软件或外部触发启动 ADC 转换
- 转换期间停止 ADC(中止)
- 通过 CPU/DMA 读数据
- 低功耗模式(AUTDLY)

可编程分辨率：在这种情况下，减少的转换值（根据 ADC_CFGR 寄存器中的 RES [1:0]位）以与 12 位转换相同的方式进行累加、截断、取整和移位。

注意：使用过采样数据时，数据对齐不可用。ADC_CFGR 中的 ALIGN 位被忽略，并且数据总是以右对齐的方式提供。过采样模式下不支持偏移校正。设置 ROVSE 和/或 JOVSE 位时，ADC_OFRy 寄存器中 OFFSETy_EN 位的值被忽略（视为重置）。

模拟看门狗

保持模拟看门狗功能（AWDSGL 和 AWDEN 位），但有以下区别：

- 忽略 RES[1:0]位，始终使用完整的 12 位值 HT[11:0]和 LT[11:0]进行比较
- 对 16 位过采样结果 ADC_DR[15:4]中的最高有效 12 位执行比较

注意：使用高移位值时必须小心，这会减小比较范围。例如，如果过采样结果被移位 4 位，从而产生 12 位的数据右对齐，则有效的模拟看门狗比较只能在 8 位上执行。在 ADC_DR[11:4]和 HTx[0:7]，LTx[0:7]进行比较，并且 HTx[11:8]，LTx[11:8]必须复位。

触发模式

平均器也可以用于基本的过滤。虽然不是一个非常强大的滤波器（缓慢的滚降和有限的阻带衰减），但它可以用作陷波滤波器来抑制恒定的寄生频率（通常来自电源或开关模式电源）。为此，可以使用 ADC_CFGR2 中的 TROVS 位启用特定的间断模式，以便能够具有由用户定义的过采样频率，并且独立于转换时间本身。

下图显示了在间断模式下如何响应触发启动转换。如果设置了 TROVS 位，则忽略 DISCEN 位的内容并将其视为 1。

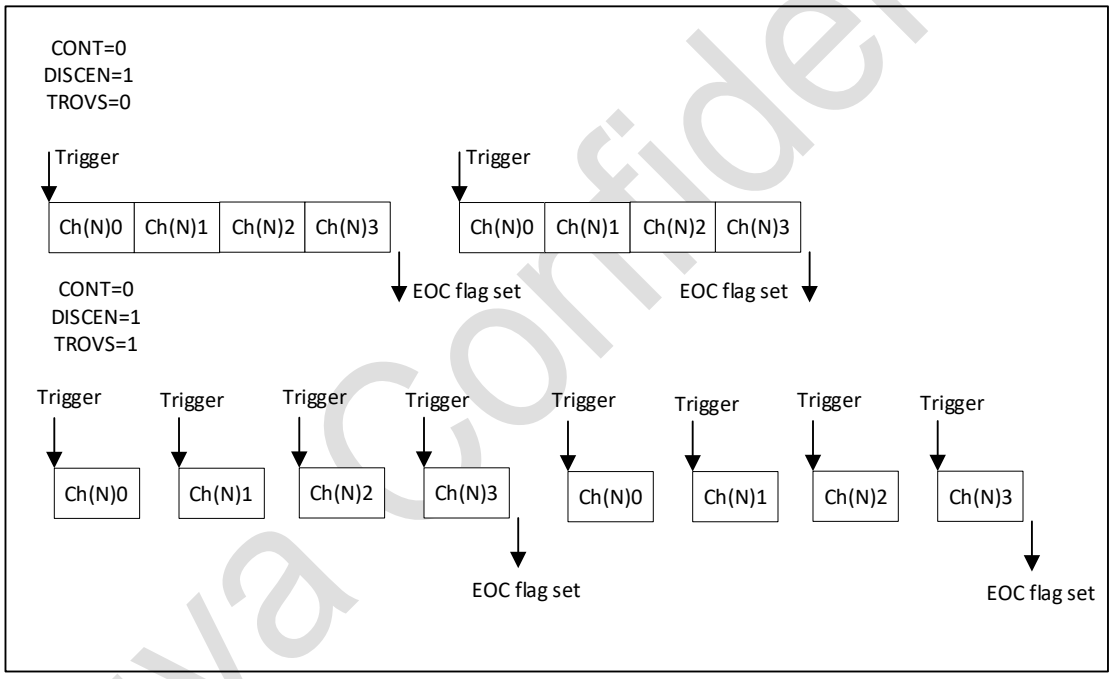


图 13-28 规则的过采样模式触发（TROVS 位=1）

过采样时注入和规则序列管理

在过采样模式下，注入序列和规则序列行为可能有所区别。如果两个序列必须同时使用,可以对它们启用过采样，但有一些限制（这与唯一的累加单元有关）。

仅对规则通道进行过采样

规则过采样模式位 ROVSM 定义了如果规则过采样序列被注入的转换中断，则如何恢复该序列：

- 在连续模式中，累加从最后一个有效数据重新开始（注入的触发而导致之前转换中止）。这确保了无论注入频率如何都将完成过采样（提供至少完成一次规则转换在触发之间）；
- 在恢复模式下，累加从 0 重新开始（忽略以前的转换结果）。该模式允许保证用于过采样的所有数据在单个时隙内背靠背地转换。必须注意使注入触发周期高于过采样周期长度。如果不遵守此条件，则过采样将无法完成，规则序列将被阻塞。

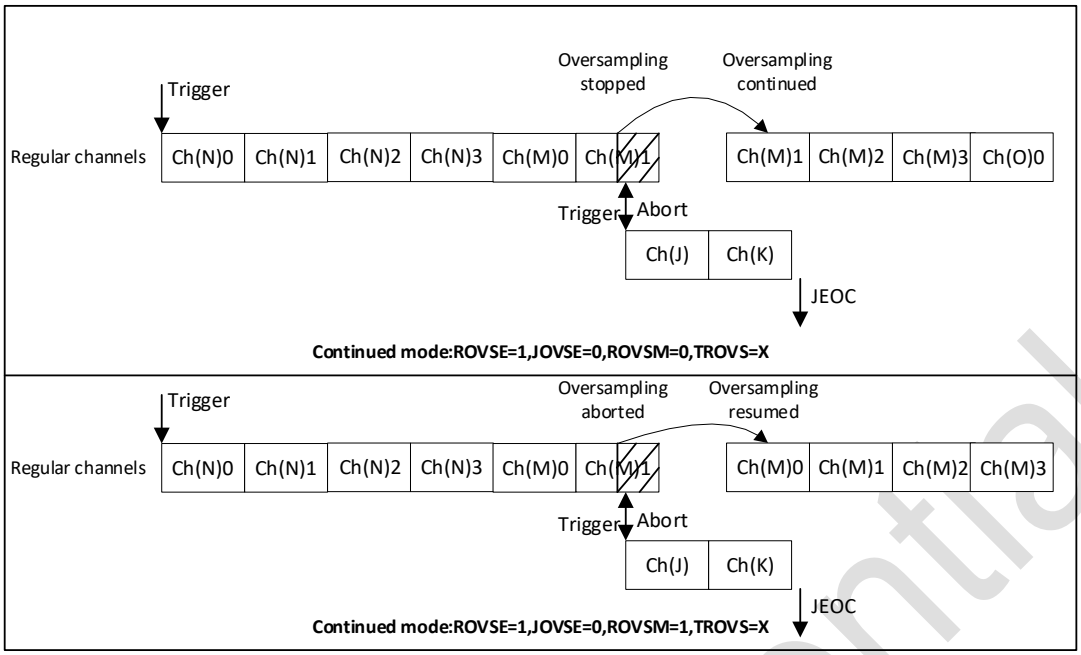


图 13-29 4 倍过采样率模式

仅对注入通道进行过采样

注入过采样模式位 JOVSE 仅为注入序列中的转换启用过采样。

规则通道和注入通道过采样

可以同时设置 ROVSE 和 JOVSE 位。在这种情况下，规则过采样模式被强制为恢复模式 (忽略 ROVSM 位），如下图所示

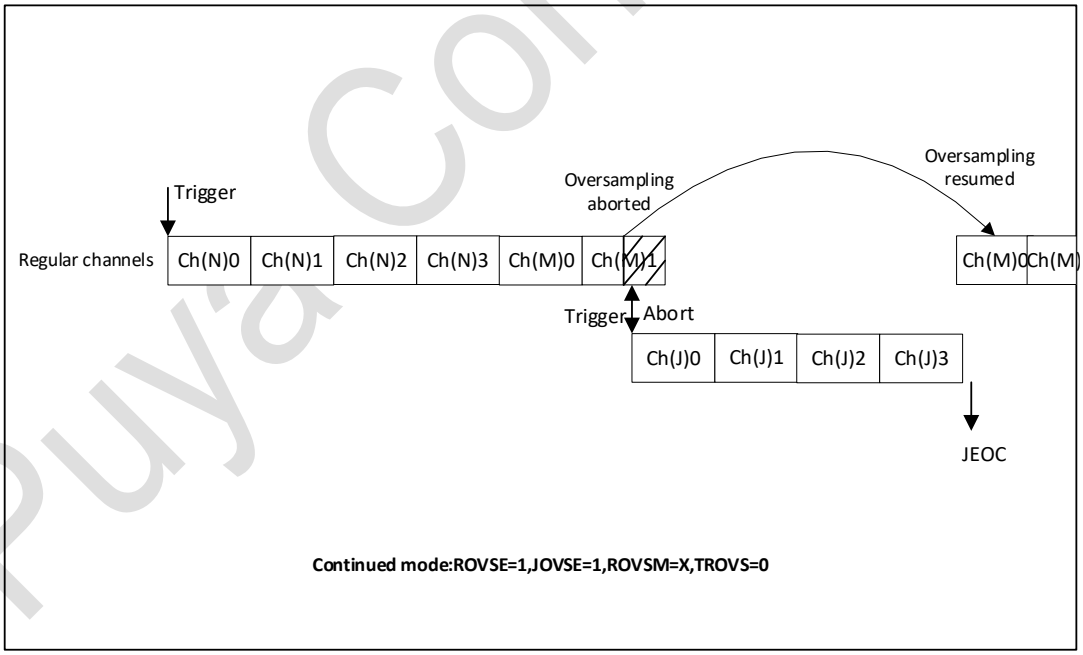


图 13-30 同时使用规则和注入过采样模式

TROVS 规则过采样被注入中断

已经触发转换规则模式被注入转换中断。在这种情况下，必须禁用注入模式过采样模式，并忽略 ROVSM 位（强制恢复模式）。JOVSE 位必须复位。该行为如下图所示。

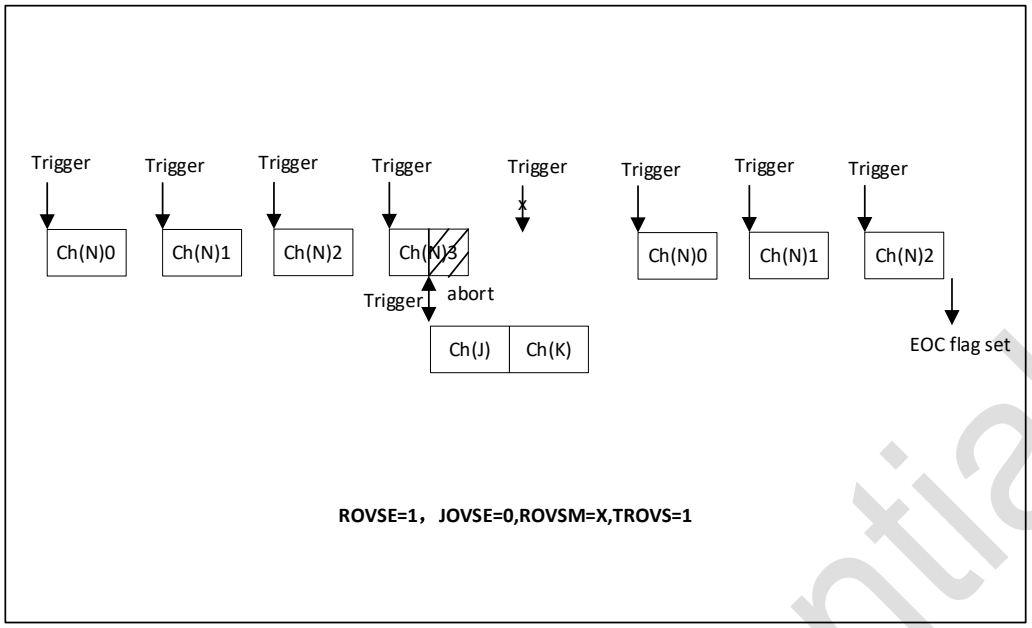


图 13-31 规则过采样被注入触发

自动注入模式

可以对自动注入的序列进行过采样，并将所有转换结果存储在寄存器中，以节省 DMA 资源。此模式仅在规则和注入过采样均使能时可用：JAUTO=1、ROVSE=1 和 JOVSE=1，不支持其他组合。ROVSM 位在自动注入模式下被忽略。下图显示了转换的顺序。

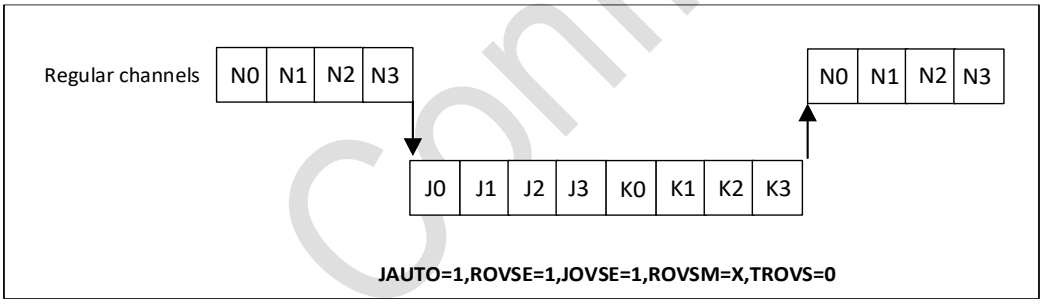


图 13-32 自动注入模式过采样

也可以使用 TROVS 位启用触发模式。在这种情况下，ADC 必须配置如下：JAUTO=1，DISCEN=0，JDISCEN=0，ROVSE=1，JOVSE=1 和 TROVSE=1。（该配置也可与 SCAN 组合，但禁止与 CONT 组合(使能 CONT 会导致功能异常)）。

模式组合小结

下表总结了所有组合，包括不支持的模式。

表 13-8 过采样器模式小结

规则过采样 ROVSE	注入过采样 JOVSE	过采样模式 ROVSM;0=continued 1=resumed	触发规则模式 TROVS	描述
1	0	0	0	Regular continued mode
1	0	0	1	不支持
1	0	1	0	Regular resumed mode
1	0	1	1	Triggered regular resumed mode

1	1	0	X	不支持
1	1	1	0	Injected and regular resumed mode
1	1	1	1	不支持
0	1	X	X	注入过采样

13.3.19 可编程采样时间

ADC 会在若干个 ADC_CLK 周期内对输入电压进行采样，采样周期数通过 ADC_SMPRx 寄存器中的 SMP[2:0] 位配置。每个通道均可以使用不同的采样时间进行采样。

总转换时间计算如下：

t_{CONV} = （采样时间 + (转换分辨率+0.5) ） x ADC 时钟周期

例如：当 f_{ADC} = 16 MHz，分辨率为 12 位，且采样时间为 3.5 个 ADC 时钟周期：

t_{CONV} = (3.5 + (12+0.5)) x ADC 时钟周期= 16 x ADC 时钟周期 = 1 μs

EOSMP 标志位用来表明采样阶段的结束。

对采样时间的限制：

- 对于每个通道，SMP[2:0]位必须按照数据手册中 ADC 特性部分所指定的最小采样时间进行编程。

13.3.20 外部触发转换

转换或转换序列可以由软件或外部事件（例如定时器捕获、输入引脚）触发。如果 EXTEN[1:0]位（用于规则转换）或 JEXTEN[1:0]bit（用于注入转换）不等于 00，则外部事件能够触发具有所选极性的转换。

一旦软件设置了位 ADSTART = 1，规则软件触发选择就有效，而一旦软件设置位 JADSTART = 1，注入的软件触发选择就生效。在转换过程中发生的任何软硬件触发都将被忽略。

- 如果位 ADSTART=0，任何发生的规则硬件触发都会被忽略。
- 如果位 JADSTART=0，任何注入的硬件触发将被忽略。当软件设置 ADSTART=1 时，触发选择有效。

下表提供了 EXTEN[1:0]和 JEXTEN[1:0]值与触发极性之间的对应关系。

表 13-9 规则外部触发配置触发极性

EXTEN[1:0]	触发方式
00	硬件触发检测禁用，软件触发检测启用(上升沿)
01	上升沿检测的硬件触发
10	下降沿检测的硬件触发
11	硬件触发，在上升沿和下降沿都有检测

注意：规则和注入触发的极性不能随时更改

表 13-10 注入外部触发配置触发极性

JEXTEN[1:0]	触发方式
00	硬件触发检测禁用，软件触发检测启用(上升沿)
01	上升沿检测的硬件触发
10	下降沿检测的硬件触发
11	硬件触发，在上升沿和下降沿都有检测

表 13-11 ADC 外部触发事件选择

ADC trigger selection EXTSEL[4:0] or JEXTSEL[4:0]	Regular		Injected	
	Name	Source	Name	Source
00000	ADC_EXT_TRG0	tim1_cc1	ADC_JEXT_TRG0	tim1_trgo
00001	ADC_EXT_TRG1	tim1_cc2	ADC_JEXT_TRG1	tim1_cc4
00010	ADC_EXT_TRG2	tim1_cc3	ADC_JEXT_TRG2	tim2_trgo
00011	ADC_EXT_TRG3	tim2_cc2	ADC_JEXT_TRG3	tim2_cc1
00100	ADC_EXT_TRG4	tim3_trgo	ADC_JEXT_TRG4	tim3_cc4
00101	ADC_EXT_TRG5	tim4_cc4	ADC_JEXT_TRG5	tim4_trgo
00110	ADC_EXT_TRG6	exti11	ADC_JEXT_TRG6	exti15
00111	ADC_EXT_TRG7	tim1_trgo	ADC_JEXT_TRG7	tim3_cc3
01000	ADC_EXT_TRG8	tim2_trgo	ADC_JEXT_TRG8	tim3_trgo
01001	ADC_EXT_TRG9	tim4_trgo	ADC_JEXT_TRG9	tim3_cc1
01010	ADC_EXT_TRG10	tim6_trgo	ADC_JEXT_TRG10	tim6_trgo
01011	ADC_EXT_TRG11	tim15_trgo	ADC_JEXT_TRG11	tim15_trgo
01100	ADC_EXT_TRG12	tim3_cc4	ADC_JEXT_TRG12	tim2_cc3
01101	ADC_EXT_TRG13	tim7_trgo	ADC_JEXT_TRG13	tim16_cc1
01110	ADC_EXT_TRG14	lptim_out	ADC_JEXT_TRG14	tim7_trgo
01111	ADC_EXT_TRG15	tim1_cc5	ADC_JEXT_TRG15	lptim_out
10000	ADC_EXT_TRG16	tim1_cc6	ADC_JEXT_TRG16	tim1_cc5
10001	ADC_EXT_TRG17	-	ADC_JEXT_TRG17	tim1_cc6
其他	-	-	-	-

13.3.21 可配置分辨率

可通过降低 ADC 分辨率来执行快速转换。ADC_CFGR 寄存器的 RES 位用于选择数据寄存器中可用的位数。每种分辨率的最小转换时间如下：

- 12 位：3.5 + 12.5 = 16 ADCCLK 周期
- 10 位：3.5 + 10.5 = 14 ADCCLK 周期
- 8 位：3.5 + 8.5 = 12 ADCCLK 周期
- 6 位：3.5 + 6.5 = 10 ADCCLK 周期

13.3.22 温度传感器和内部参考电压

温度传感器可用于测量器件结点温度（TJ)。温度传感器内部连接到 ADC_CH18 通道，此通道把传感器输出的电压转换成数字值。温度传感器不使用时，可将传感器置于掉电模式。

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。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。

注意：必须设置 VSENSESEL ，VREFEN 激活内部通道：ADC_CH18(温度传感器)和 ADC_CH20(VREFINT)的转换。

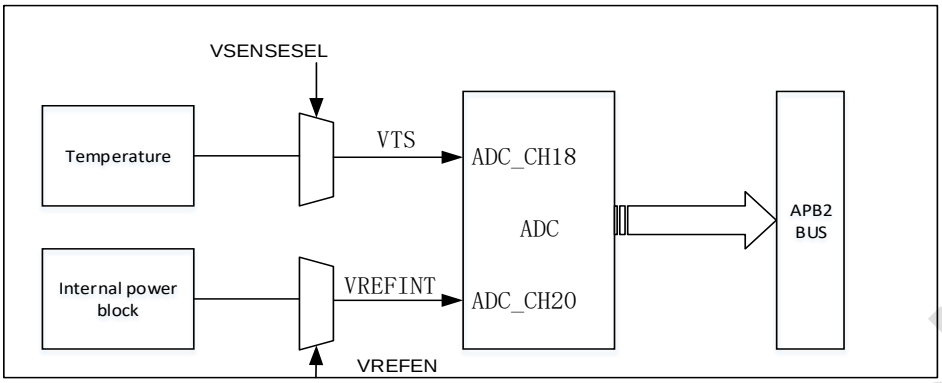


图 13-33 V_{TS} 和 V_{REFINT} 通道

读温度

要使用传感器，请执行以下操作：

1. 选择 ADC_CH18
2. 选择一个采样时间，该采样时间要大于数据手册中所指定的最低采样时间。
3. 在 ADC_CCR 寄存器中 VSENSESEL 位置 1，以便将温度传感器从掉电模式中唤醒。
4. 通过将 ADEN 位置 1，然后启动 ADC 转换
5. 读取 ADC 数据寄存器中生成的 VTS 数据
6. 使用以下公式计算温度：

$$Temperature(in\ ℃) = \frac{105℃ - 30℃}{TS_{CAL2} - TS_{CAL1}} \times (TS_{DATA} - TS_{CAL1}) + 30℃$$

TS_{CAL2} 代表 105 ℃温度传感器的校准值

TS_{CAL1} 代表 30℃温度传感器的校准值

TS_{DATA} 是 ADC 转换的实际输出值

注： 传感器从断电模式下唤醒时到能正确输出 VTS 要有一个启动时间，ADC 从上电后启动也有一个启动时间， 若要减少这个延时， 则需要在同一时间的设置 ADEN 和 VSENSESEL 位。

利用内部的参考电压计算实际的 V_{CC} 电压

下面公式可以给出真实 V_{CC} 的电压值：

$$VREFINT = 1.2V = \frac{ADC\_DATA}{4095} \times VCC$$

V_{REFINT} 固定值为 1.2 V。

ADC_DATA 是 V_{REFINT} 通道的转换数据。

把一个 ADC 测量到的通道电压相对值转化为绝对电压值

ADC 是根据模拟电源输入和转换通道上的电压比例给出一个数字值。大部分应用是需要把这个比例转换成一个电压值。对于 V_{CC} 可知的情况下并且 ADC 转换值是右对齐的，可用下面公式得到这个绝对电压值：

$$VCHANNEL = \frac{ADC\_DATAx}{4095} \times VCC$$

$V_{CHANNEL}$ 是通道电压；

ADC_DATAx 是 ADC_DR 里面的转换数据；

4095 表示为 12 位。

13.3.23 电池监视

由于 V_{CC} 电压可能不准，因此 V_{CC} 引脚需要内部连接到桥接分配器，以确保 ADC 正确运行。

桥接器自动使能将 $V_{CC}/3$ 连接到 ADC_CH19 输入通道。

13.3.24 ADC 中断

ADC 中断可由以下任一事件产生：

- 任何一次的规则转换结束(EOC 标志)
- 规则序列转换结束 (EOS 标志)
- 任何一次的注入转换结束 (JEOC 标志)
- 注入序列转换结束 (JEOS 标志)
- 当模拟看门狗检测发生 (AWD 标志)
- 当采样阶段结束发生 (EOSMP 标志)
- 当数据过载发生 (OVR 标志)
- 上电结束(ADRDY 标志)
- 校准结束（EOCAL 标志）

独自的中断使能位用于灵活设置 ADC 中断。

表 13-12 ADC 中断

中断事件	事件标志	使能控制
规则转换结束	EOC	EOCIE
规则序列转换结束	EOS	EOSIE
注入转换结束	JEOC	JEOCIE
注入序列转换结束	JEOS	JEOSIE
模拟看门狗状态置位	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
过载	OVR	OVRIE
AD 上电结束	ADRDY	ADRDYIE
校准结束	EOCAL	EOCALIE

13.4 ADC 寄存器

13.4.1 ADC 中断和状态寄存器 (ADC_ISR)

偏移地址： 0x00

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	EOCAL	Res.	AWD3	AWD2	AWD1	JEOS	JEOC	OVR	EOS	EOC	EOSMP	ADRDY
				RC_W1		RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1	RC_W1

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	EOCAL	RC_W1	0	校准完成标志 0：校准未启动或者校准未完成

				1：校准完成
10	Reserved	-	-	保留
9	AWD3	RC_W1	0	模拟看门狗 3 标志。当转换的电压超过/低于 ADC_TR3 寄存器的字段 LT3[7:0]和 HT3[7:0]中值时，该位由硬件设置。它通过软件写入 1 而被清除。 0：未发生模拟看门狗 3 事件（或软件已确认并清除标志事件） 1：发生模拟看门狗 3 事件
8	AWD2	RC_W1	0	模拟看门狗 2 标志。当转换的电压超过/低于 ADC_TR2 寄存器的字段 LT2[7:0]和 HT2[7:0]中值时，该位由硬件设置。它通过软件写入 1 而被清除。 0：未发生模拟看门狗 2 事件（或软件已确认并清除标志事件） 1：发生模拟看门狗 2 事件
7	AWD1	RC_W1	0	模拟看门狗 1 标志。当转换的电压超过/低于 ADC_TR1 寄存器的字段 LT1[7:0]和 HT1[7:0]中值时，该位由硬件设置。它通过软件写入 1 而被清除。 0：未发生模拟看门狗 1 事件（或软件已确认并清除标志事件） 1：发生模拟看门狗 1 事件
6	JEOS	RC_W1	0	注入序列转换结束标志。当注入序列的所有通道转换结束时，该位由硬件设置。它通过软件写入 1 而被清除。 0：注入序列转换未完成（或标志事件已被软件确认并清除） 1：注入序列转换已完成
5	JEOC	RC_W1	0	注入通道转换结束标志。当每次注入通道转换结束时，新数据可通过相应的 ADC_JDRy 寄存器中访问时，该位由硬件设置。通过软件写入 1 或通过读取相应的 ADC_JDRy 寄存器来清除。 0：注入通道转换未完成（或标志事件已被软件确认并清除） 1：注入通道转换已完成
4	OVR	RC_W1	0	规则通道过载 当过载发生时，硬件置位该位。当 EOC 标志已置起表明一次新的转换已完成。该位写 1 清 0 0：无过载发生（或软件已应答和清除该位） 1：过载已发生
3	EOS	RC_W1	0	规则序列结束标志 规则序列转换结束时硬件置位该位。软件写 1 清 0 0：转换序列没有完成（或者软件已经应答和清除该标志） 1：转换序列完成
2	EOC	RC_W1	0	转换结束标志 当每个规则通道每次转换结束后，新的数据结果可以从

				ADC_DR 寄存器读到时，硬件置位该位。软件写 1 清 0 或读 ADC_DR 寄存器清 0 0：通道转换没有完成（或者软件已经应答和清除该标志） 1：通道转换已完成
1	EOSMP	RC_W1	0	采样结束标志，在每次规则通道转换的采样阶段结束时，硬件置位该位，软件写 1 清 0 0：不处在采样阶段结束时（或者软件已经应答和清除该标志） 1：采样阶段结束
0	ADRDY	RC_W1	0	ADC 就绪 ADC 使能后（位 ADEN=1）以及 ADC 达到准备好接收转换请求的状态时，会通过硬件将该位置 1。 通过软件写入 1 可将该位清零。 0：ADC 未准备好开始转换（或标志事件已通过软件确认并清零） 1：ADC 已准备好开始转换

13.4.2 ADC 中断使能寄存器（ADC_IER)

偏移地址： 0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	EOCALIE	Res.	AWD3IE	AWD2IE	AWD1IE	JEOSIE	JEOCIE	OVRIE	EOSIE	EOCIE	EOSMPIE	ADRDYIE
				RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	EOCALIE	RW	0	校准完成中断使能 0：校准完成中断禁止 1：校准完成中断使能
10	Reserved	-	-	保留
9	AWD3IE	RW	0	模拟看门狗 3 标志中断使能。该位由软件设置和清除以使能/禁止模拟看门狗 3 标志中断。 0：禁止模拟看门狗 3 事件中断 1：开启模拟看门狗 3 事件中断
8	AWD2IE	RW	0	模拟看门狗 2 标志中断使能。该位由软件设置和清除以使能/禁止模拟看门狗 2 标志中断。 0：禁止模拟看门狗 2 事件中断 1：开启模拟看门狗 2 事件中断
7	AWD1IE	RW	0	模拟看门狗 1 标志中断使能。该位由软件设置和清除以使能/禁止模拟看门狗 1 标志中断。 0：禁止模拟看门狗 1 事件中断 1：开启模拟看门狗 1 事件中断

6	JEOSIE	RW	0	注入序列转换结束中断使能。该位由软件设置和清除以使能/禁止注入序列转换结束中断。 0：禁止 JEOS 中断 1：使能 JEOS 中断。当 JEOS 置位时，产生一个中断。
5	JEOCIE	RW	0	注入通道转换结束中断使能。该位由软件设置和清除以使能/禁止注入通道转换结束中断。 0：禁止 JEOC 中断 1：使能 JEOC 中断。当 JEOC 置位时，产生一个中断。 注意：只有当 JADSTART = 0 时，软件才被允许写入该位（这确保了没有注入转换正在进行）。
4	OVRIE	RW	0	ADC 规则通道过载中断使能位 软件清除或置起过载中断使能 0：ADC 过载中断不使能 1：ADC 过载中断使能 当 ADSTART=0 时(确保没有任何转换正在进行)软件可以写该位
3	EOSIE	RW	0	规则序列结束中断使能 该位由软件设置和清除以使能/禁止规则序列结束中断 0：禁止 EOS 中断 1：使能 EOS 中断。当 EOS 置位时，产生一个中断。 注意：只有当 ADSTART = 0 时，软件才被允许写入该位（这确保了没有规则转换正在进行）。
2	EOCIE	RW	0	规则转换结束中断使能 该位由软件设置和清除以使能/禁止规则转换结束中断 0：禁止 EOC 中断 1：使能 EOC 中断。当 EOC 置位时，产生一个中断。 注意：只有当 ADSTART = 0 时，软件才被允许写入该位（这确保了没有规则转换正在进行）。
1	EOSMPIE	RW	0	采样结束中断使能 该位由软件设置和清除以使能/禁止采样结束中断 0：禁止 EOSMP 中断 1：使能 EOSMP 中断。当 EOSMP 置位时，产生一个中断。 注意：只有当 ADSTART = 0 时，软件才被允许写入该位（这确保了没有规则转换正在进行）。
0	ADRDYIE	RW	0	中断使能位 0：ADC 就绪中断不使能 1：ADC 就绪中断使能 当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)软件可以写该位

13.4.3 ADC 控制寄存器（ADC_CR)

偏移地址： 0x08

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAL	Res.	Res.	Res.	RSTCAL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RS				RS											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	JADSTP	ADSTP	JADSTART	ADSTART	ADDIS	ADEN
										RS	RS	RS	RS	RS	RS

Bit	Name	R/W	Reset Value	Function
31	ADCAL	RS	0	ADC 校准启动。 软件设置启动 ADC 校准，校正完成后硬件自动清 0。在校准失败或者校准成功时由硬件清除 0: 校准完成 1: 写 1 校正 ADC，读为 1 表明校准正在进行 注意：仅当 ADEN=0 时，才允许软件通过设置 ADCAL 来启动校准。仅当 ADEN 无效且无外部触发时（未启用 ADC 且无转换正在进行），才允许软件通过写入 ADC_CALFACT 来更新校准因子
30:28	Reserved	-	-	保留
27	RSTCAL	RS	0	校准复位使能位 该位由软件置位，由硬件清 0。在校准寄存器被初始化后(即 RSTCAL 置 1 后)，该位即被清除。 0：校准寄存器已初始化 1：初始化校准寄存器 注：当正在进行转换时，如果设置 RSTCAL，清除校准寄存器需要额外的周期。
26：6	Reserved	-	-	保留
5	JADSTP	RS	0	ADC 停止注入转换命令。 软件置位停止和丢弃正在进行的注入转换（JADSTP 命令）。 当转换被丢弃并且准备接受新的转换命令时硬件会清除该位。 0: 没有正在进行的 ADC 停止注入转换命令 1: 写 1 停止 ADC 注入转换，读为 1 表明一个 JADSTP 命令正在进行中。 软件写该位为 0 为无效操作。
4	ADSTP	RS	0	ADC 停止规则转换命令。 软件置位停止和丢弃正在进行的转换（ADSTP 命令）。 当转换被丢弃并且准备接受新的转换命令时硬件会清除该位。 0: 没有正在进行的 ADC 停止转换命令

				1: 写 1 停止 ADC 规则转换，读为 1 表明一个 ADSTP 命令正在进行中。 软件写该位为 0 为无效操作。
3	JADSTART	RS	0	ADC 触发启动注入转换 该位由软件设置，用于启动注入通道的 ADC 转换。根据 EXTEN[1：0]的配置来决定转换是软件立即启动，还是由硬件触发事件来启动。 0：复位状态 1：开始转换注入通道 该位由硬件清除的情况： – 在单次转换模式，选择软件驱动时(JEXTEN=00)：序列转换完成时（JEOS 标志置位） – 其他情况下：执行 JADSTP 命令之后，同时 JADSTP 标志又被硬件清 0 之时; 0: 没有正在进行的 ADC 转换 1: 写 1 启动 ADC 注入转换，读为 1 表明 ADC 正在操作，可能正在转换。 注：软件只有当 ADEN=1 且 ADDIS=0 时才能配置 JADSTART=1. 软件写该位为 0 为无效操作。
2	ADSTART	RS	0	ADC 启动命令。 软件置位该位启动 ADC 转换。根据 EXTEN[1：0]的配置来决定转换是软件立即启动，还是由硬件触发事件来启动。 该位由硬件清除的情况： – 在单次转换模式 (CONT=0, DISCEN=0)，选择软件驱动时(EXTEN=00)：序列转换完成时（EOSEQ 标志置位） – 在非连续转换模式 (CONT=0, DISCEN=1),当软件驱动时(EXTEN=00)：转换结束标志（EOC 标志置位） – 其他情况下：执行 ADSTP 命令之后，同时 ADSTP 标志又被硬件清 0 之时 0: 没有正在进行的 ADC 转换 1: 写 1 启动 ADC 规则转换，读为 1 表明 ADC 正在操作，可能正在转换。 注：软件只有当 ADEN=1 且 ADDIS=0 时才能配置 ADSTART=1. 软件写该位为 0 为无效操作。
1	ADDIS	RS	0	ADEN 禁止使能。

				软件置位禁止 ADC。当 ADC 被禁止（ADEN 被硬件清零同时），硬件清除该位。软件写该位为 0 为无效操作。 0：没有 ADDIS 1: 写 1 禁止 ADC，读 1 表示 ADDIS 指令正在执行 注：设置 ADDIS 为 1 有效只能在 ADEN=1 并且 ADSTART=0 时（确保没有转换进行）。
0	ADEN	RS	0	开/关 A/D 转换器 ADC 转换器工作使能，当置 1 后 ADC 转换器唤醒，在转换器上电到开始转换有一个延迟 t_{STAB}。当置 0 后 ADC 转换器处于掉电状态。 0：禁止 ADC 转换，ADC 转换器进入断电模式 1：使能 ADC 转换器。

13.4.4 ADC 配置寄存器 (ADC_CFGR)

偏移地址： 0x0C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	AWD1CH[4:0]					JAUTO	JAWD1EN	AWD1EN	AWD1SGL	Res.	JDISCEN	DISCNUM[2:0]			DISCEN
	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
ALIGN	AUTDLY	CONT	OVRMOD	EXTEN[1:0]		EXTSEL[4:0]					RES[1:0]		Res.	DMACFG	DMAEN
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:26	AWD1CH[4:0]	RW	5'h0	模拟看门狗 1 通道选择位 该位由软件写设置，用于选择模拟看门狗的输入通道。 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 …… 01111：ADC 模拟输入通道 15 10000：ADC 模拟输入通道 16 10001：ADC 模拟输入通道 17 10010: ADC 模拟输入通道 18 10011: ADC 模拟输入通道 19 10100: ADC 模拟输入通道 20 保留所有其他数值。 注：模拟看门狗 1 通道选择对应的 ADC 模拟输入通道与 ADC 通道选择一致
25	JAUTO	RW	0	自动注入使能 该位用于开启或关闭规则通道组转换结束后自动进行注入通道组转换

				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
24	JAWD1EN	RW	0	注入通道模拟看门狗 1 使能。在注入通道上开启模拟看门狗。 0：在注入通道上禁用模拟看门狗 1 1：在注入通道上使用模拟看门狗 1
23	AWD1EN	RW	0	规则通道模拟看门狗 1 使能。在规则通道上开启模拟看门狗。 0：在规则通道上禁用模拟看门狗 1 1：在规则通道上使用模拟看门狗 1
22	AWD1SGL	RW	0	单一通道看门狗 1 使能。 该位用于开启或关闭由 AWD1CH[4:0]定义的通道上的模拟看门狗 1。 0：在所有的通道上使用模拟看门狗 1 1：在单一通道上使用模拟看门狗 1
21	Reserved	-	-	保留
20	JDISCEN	RW	0	注入通道间断模式使能。 该位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
19:17	DISCNUM[2:0]	RW	3’h0	间断模式通道计数 这些位用于定义在接收到外部触发后，在间断模式下要转换的规则通道的数量。 000：1 通道 001：2 通道 … 111：8 通道
16	DISCEN	RW	0	规则通道的间断模式使能 该位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
15	ALIGN	RW	0	数据对齐控制位 0：右对齐 1：左对齐
14	AUTDLY	RW	0	自动延迟转换模式 此位由软件设置和清除，以启用/禁用自动延迟转换模式。 0：自动延迟转换模式关闭 1：自动延迟转换模式打开
13	CONT	RW	0	连续转换使能 如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式 1：连续转换模式
12	OVRMOD	RW	0	过载管理模式。

				软件可设置和清除该位，配置数据过载管理的方式。 0：当过载发生时，ADC_DR 寄存器保留旧值 1：当过载发生时，ADC_DR 寄存器会被上一次转换结果覆盖掉 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
11:10	EXTEN[1:0]	RW	2'h0	规则通道的外部触发启用和极性选择 这些位由软件设置和清除，以选择外部触发极性并启用规则组的触发。 00：禁用硬件触发检测（可以通过软件启动转换） 01：上升沿的硬件触发检测 10：下降沿的硬件触发器检测 11：上升沿和下降沿都有硬件触发检测 注意：只有确保没有规则转换正在进行，软件才允许写入这些位。
9:5	EXTSEL[4:0]	RW	5'h0	规则通道外部触发事件选择位。选择启动规则通道组转换的外部事件 ADC 外部触发事件如下：（详情见上述章节） 00000：事件 1 00001：事件 2 00010：事件 3 00011：事件 4 … 10000：事件 16 10001：事件 17 其它：预留
4:3	RES[1:0]	RW	2'h0	分辨率 (Resolution)控制位。通过软件写入这些位可选择转换的分辨率。 00： 12 位 01： 10 位 10： 8 位 11： 6 位
2	Reserved	-	-	保留
1	DMACFG	RW	0	DMA 访问配置。 软件可设置和清除该位，在两种 DMA 模式操作中选择并在 DMAEN = 1 时有效。 0：DMA 单次模式选择 1：DMA 循环模式选择 注：仅当 ADSTART=0 时（确保没有正在进行的转换）允许软件写这些位。
0	DMAEN	RW	0	DMA 使能位 0：禁止 DMA 模式 1：使能 DMA 模式

13.4.5 ADC 配置寄存器 2 (ADC_CFGR2)

偏移地址： 0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	CALNUM[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GCOMP
	RW	RW	RW												RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	ROVSM	TROVS	OVSS[3:0]				OVSR[2:0]			JOVSE	ROVSE
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30:28	CALNUM[2:0]	RW	3'h0	校准平均次数 000：1 次校准和平均 001：2 次校准和平均 010：4 次校准和平均 011：8 次校准和平均 100：16 次校准和平均 101：32 次校准和平均 其它：预留
27:17	Reserved	-	-	保留
16	GCOMP	RW	0	增益补偿模式 此位由软件设置和清除，以启用增益补偿模式。 0：增益补偿未启用在 ADC 工作模式 1：增益补偿已启用并应用于所有通道 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位
15:11	Reserved	-	-	保留
10	ROVSM	RW	0	规则过采样模式 该位由软件设置和清除，以选择规则过采样模式。 0：继续模式：触发注入转换时，过采样暂时停止，并在注入序列后继续（注入序列期间保持过采样缓冲） 1：恢复模式：触发注射转换时，当前过采样被中止，并在注入序列后从开始恢复（过采样缓冲区在注入序列开始转换时清零） 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位
9	TROVS	RW	0	触发规则过采样 此位由软件设置并清除，以启用触发过采样 0：通道的所有过采样转换在触发后连续完成 1：通道的每个过采样转换都需要一个新的触发

				注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位
8:5	OVSS[3:0]	RW	4’b0	过采样移位 该位字段由软件设置和清除，以定义应用于原始过采样结果的右移。 0000：无移位 0001：移位 1 位 0010：移位 2 位 0011：移位 3 位 0100：移位 4 位 0101：移位 5 位 0110：移位 6 位 0111：移位 7 位 1000: 移位 8 位 其他：预留 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位。
4:2	OVSR[2:0]	RW	3’h0	过采样率 此位字段由软件设置并清除，以定义过采样率。 000：2x 001：4x 010：8x 011：16x 100：32x 101：64x 110：128x 111：256x 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位
1	JOVSE	RW	0	注入过采样使能 该位由软件设置和清除，以启用注入过采样。 0：禁用注入过采样 1：启用注入过采样 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位
0	ROVSE	RW	0	规则过采样使能 启用此位由软件设置并清除，以启用规则过采样。 0：禁用规则过采样 1：启用规则过采样

				注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位

13.4.6 ADC 采样时间寄存器 1(ADC_SMPR1)

偏移地址： 0x14

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:0]	
		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[2:0]	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	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:0	SMP[9:0][2:0]	RW	30'h0	通道 x 采样时间选择 这些位由软件编写，用于为每个通道单独选择采样时间。 在采样周期期间，通道选择位必须保持不变。 000：2.5 ADC 时钟周期 001：6.5 ADC 时钟周期 010：12.5 ADC 时钟周期 011：24.5 ADC 时钟周期 100：47.5 ADC 时钟周期 101：92.5 ADC 时钟周期 110：247.5 ADC 时钟周期 111：640.5 ADC 时钟周期

13.4.7 ADC 采样时间寄存器 2(ADC_SMPR2)

偏移地址： 0x18

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	SMP19[2:0]			SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:0]	
		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[2:0]	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	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29:0	SMP[19:10][2:0]	RW	30'h0	通道 x 采样时间选择 这些位由软件编写，用于为每个通道单独选择采样时间。 在采样周期期间，通道选择位必须保持不变。 000：2.5 ADC 时钟周期 001：6.5 ADC 时钟周期 010：12.5 ADC 时钟周期 011：24.5 ADC 时钟周期 100：47.5 ADC 时钟周期 101：92.5 ADC 时钟周期 110：247.5 ADC 时钟周期 111：640.5 ADC 时钟周期

13.4.8 DC 采样时间寄存器 2(ADC_SMPR3)

偏移地址： 0x20

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMP20[2:0]		
													RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:0	SMP20 [2:0]	RW	3'h0	通道 20 采样时间选择 这些位由软件编写，用于为每个通道单独选择采样时间。 在采样周期期间，通道选择位必须保持不变。 000：2.5 ADC 时钟周期 001：6.5 ADC 时钟周期 010：12.5 ADC 时钟周期 011：24.5 ADC 时钟周期 100：47.5 ADC 时钟周期 101：92.5 ADC 时钟周期 110：247.5 ADC 时钟周期 111：640.5 ADC 时钟周期

13.4.9 ADC 看门狗阈值寄存器 1 (ADC_TR1)

偏移地址： 0x24

复位值： 0x0FFF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	HT1[11:0]											
				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**
Res.	AWDFILT[2:0]			LT1[11:0]											
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:16	HT1[11:0]	RW	12'hFFF	模拟看门狗 1 高阈值 软件可配，定义模拟看门狗 1 高阈值： 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。
15	Reserved	-	-	保留
14:12	AWDFILT	RW	3'h0	模拟看门狗滤波参数 该位由软件设置和清除。 000：无过滤 001：连续两次检测生成 AWD 标志或中断 … 111：八个连续检测生成 AWD 标志或中断 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入此位（这确保没有转换正在进行）。
11:0	LT1[11:0]	RW	12'h0	模拟看门狗 1 低阈值

				软件可配，定义模拟看门狗 1 低阈值 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。

13.4.10 ADC 看门狗阈值寄存器 2 (ADC_TR2)

偏移地址：0x28

复位值：0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT2[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT2[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:16	HT2[7:0]	RW	8’hFF	模拟看门狗 2 高阈值 软件可配，定义模拟看门狗 2 高阈值 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。
15:8	Reserved	-	-	保留
7:0	LT2[7:0]	RW	8’h0	模拟看门狗 2 低阈值 软件可配，定义模拟看门狗 2 低阈值 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。

13.4.11 ADC 看门狗阈值寄存器 3 (ADC_TR3)

偏移地址： 0x30

复位值： 0x00FF 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HT3[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LT3[7:0]							
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:24	Reserved	-	-	保留
23:16	HT3[7:0]	RW	8’hFF	模拟看门狗 3 高阈值 软件可配，定义模拟看门狗 3 高阈值 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。
15:8	Reserved	-	-	保留
7:0	LT3[7:0]	RW	8’h0	模拟看门狗 3 低阈值 软件可配，定义模拟看门狗 3 低阈值 注意：只有当 ADSTART 且 JADSTART=0 时，软件才允许写入这些位（这确保了没有正在进行的转换）。

13.4.12 ADC 规则序列寄存器 1(ADC_SQR1)

偏移地址：0x34

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SQ4[4:0]					Res.	SQ3[4:0]					Res.	SQ2[4]
			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**
SQ2[3:0]				Res.	SQ1[4:0]					Res.	Res.	LT3[7:0]			
RW	RW	RW	RW		RW	RW	RW	RW	RW			RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28:24	SQ4[4:0]	RW	5'h0	规则序列第 4 次转换 这些位由软件写入,任一 ADC 通道被分配为规则转换序列中的第 4 次。
23	Reserved	-	-	保留
22:18	SQ3[4:0]	RW	5'h0	规则序列第 3 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 3 次。
17	Reserved	-	-	保留
16:12	SQ2[4:0]	RW	5'h0	规则序列第 2 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 2 次。
11	Reserved	-	-	保留
10:6	SQ1[4:0]	RW	5'h0	规则序列第 1 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 1 次。
5:4	Reserved	-	-	保留
3:0	L[3:0]	RW	4'h0	规则通道序列长度 软件配置这些位的值。这些位定义了在规则通道转换序列中通道数目。 0000：1 个转换 0001：2 个转换 …… 1111：16 个转换

注：某些通道未进行物理连接，因此不得选择进行转换。

13.4.13 ADC 规则序列寄存器 2(ADC_SQR2)

偏移地址：0x38

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	SQ9[4:0]					Res.	SQ8[4:0]					Res.	SQ7[4]
			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**
SQ7[3:0]				Res.	SQ6[4:0]					Res.	SQ5[4:0]				
RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28:24	SQ9[4:0]	RW	5'h0	规则序列第 9 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 9 次。
23	Reserved	-	-	保留
22:18	SQ8[4:0]	RW	5'h0	规则序列第 8 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 8 次。

17	Reserved	-	-	保留
16:12	SQ7[4:0]	RW	5'h0	规则序列第 7 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 7 次。
11	Reserved	-	-	保留
10:6	SQ6[4:0]	RW	5'h0	规则序列第 6 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 6 次。
5	Reserved	-	-	保留
4:0	SQ5[4:0]	RW	5'h0	规则序列第 5 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 5 次。

某些通道未进行物理连接，因此不得选择进行转换。

13.4.14 ADC 规则序列寄存器 3(ADC_SQR3)

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
Res.	Res.	Res.	SQ14[4:0]					Res.	SQ13[4:0]					Res.	SQ12[4]
			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**
SQ12[3:0]				Res.	SQ11[4:0]					Res.	SQ10[4:0]				
RW	RW	RW	RW		RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:29	Reserved	-	-	保留
28:24	SQ14[4:0]	RW	5'h0	规则序列第 14 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 14 次。
23	Reserved	-	-	保留
22:18	SQ13[4:0]	RW	5'h0	规则序列第 13 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 13 次。
17	Reserved	-	-	保留
16:12	SQ12[4:0]	RW	5'h0	规则序列第 12 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 12 次。
11	Reserved	-	-	保留
10:6	SQ11[4:0]	RW	5'h0	规则序列第 11 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 11 次。
5	Reserved	-	-	保留
4:0	SQ10[4:0]	RW	5'h0	规则序列第 10 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 10 次。

某些通道未进行物理连接，因此不得选择进行转换。

13.4.15 ADC 规则序列寄存器 4(ADC_SQR4)

偏移地址：0x40

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	SQ16[4:0]					Res.	SQ15[4:0]				
					RW	RW	RW	RW	RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:6	SQ16[4:0]	RW	5'b0	规则序列第 16 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 16 次。
5	Reserved	-	-	保留
4:0	SQ15[4:0]	RW	5'h0	规则序列第 15 次转换 这些位由软件写入，任一 ADC 通道被分配为规则转换序列中的第 15 次。

注：某些通道未进行物理连接，因此不得选择进行转换。

13.4.16 ADC 规则数据寄存器（ADC_DR)

偏移地址： 0x4C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	RDATA[15:0]	R	16'h0	规则通道转换结果 软件可读这些位的值。这些位为只读，包含了规则通道的转换结果。数据是左或右对齐

13.4.17 ADC 注入序列寄存器(ADC_JSQR)

偏移地址：0x50

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
JSQ4[4:0]					Res.	JSQ3[4:0]					Res.	JSQ2[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**
JSQ2[0]	Res	JSQ1[4:0]					JEXTEN[1:0]		JEXTSEL[4:0]					JL[1:0]	
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:27	JSQ4[4:0]	RW	5'h0	注入序列第 4 次转换 这些位由软件写入，任一 ADC 通道被分配为注入转换序列中的第 4 次。
26	Reserved	-	-	保留
25:21	JSQ3[4:0]	RW	5'h0	注入序列第 3 次转换 这些位由软件写入，任一 ADC 通道被分配为注入转换序列中的第 3 次。
20	Reserved	-	-	保留
19:15	JSQ2[4:0]	RW	5'h0	注入序列第 2 次转换 这些位由软件写入，任一 ADC 通道被分配为注入转换序列中的第 2 次。
14	Reserved	-	-	保留
13:9	JSQ1[4:0]	RW	5'h0	注入序列第 1 次转换 这些位由软件写入，任一 ADC 通道被分配为注入转换序列中的第 1 次。
8:7	JEXTEN[1:0]	RW	2'h0	注入通道的外部触发启用和极性选择

				这些位由软件设置和清除，以选择外部触发极性并启用注入组的触发。 00:禁用硬件触发检测（可以通过软件启动转换） 01:上升沿的硬件触发检测 10:下降沿的软件触发检测 11:上升沿和下降沿都有硬件触发检测 注意：只有当 ADSTART=0 且 JADSTART=0 时(确保没有任何转换正在进行)，软件才允许写入此位。
6:2	JEXTSEL[4:0]	RW	5'h0	注入通道组外部触发事件选择位 ADC 的外部触发事件如下：（详情见上述章节） 00000：事件 1 00001：事件 2 00010：事件 3 00011：事件 4 … 10000：事件 16 10001：事件 17 10010：事件 18 其它：保留
1:0	JL[1:0]	RW	2'h0	注入通道序列长度 这些位由软件编写，用于定义注入通道转换序列中的转换总数。 00： 1 个转换 01： 2 个转换 10： 3 个转换 11： 4 个转换

注：某些通道未进行物理连接，因此不得选择进行转换。

13.4.18 ADC offset y 寄存器(ADC_OFRy)

偏移地址： 0x60+0x04*(y-1), (y=1 to 4)

复位值：0x0000_0000

(SATEN、OFFSETPOS 只有 ADC_OFR1 有，并且 ADC_OFR1-4 共用)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSETy_EN	OFFSETy_CH[4:0]					SATEN	OFFSETPOS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	OFFSETy[11:0]											
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	OFFSETy_EN	RW	0	OFFSET y 使能 该位由软件写入，以启用或禁用 offset 值 OFFSETy[11:0]。 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。
30:26	OFFSETy_CH[4:0]	RW	5'h0	数据偏移 y 的通道选择 这些位由软件写入，以定义 OFFSETy[11:0]的偏移将应用的通道。 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。 某些通道没有物理连接，因此不能为数据偏移量 y 选择。
25	SATEN	RW	0	饱和使能

				此位由软件设置和清除，以使能偏移在 0x000 和 0xFFF 饱和。 0：无饱和控制，偏移结果可以有符号 1：启用饱和，偏移结果无符号且在 0x000 和 0xFFF 处饱和 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。
24	OFFSETPOS	RW	0	正偏移 此位由软件设置和清除，以使能正偏移。 0：负偏移 1：正偏移 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。
23:12	Reserved	-	-	保留
11:0	OFFSETy[11:0]	RW	12'h0	OFFSETy_CH[4:0]通道的数据偏移 y。 这些位由软件写入，以定义在转换通道（可以是规则的或注入的）时从原始转换数据中减去的偏移 y。应用数据偏移 y 的通道必须在位 OFFSETy_CH[4:0]中编程。转换结果可以从 ADC_DR（规则转换）或 ADC_JDRy 寄存器（注入转换）中读取。 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。 如果多个偏移（OFFSETy）指向同一通道，则仅考虑具有最低 x 值的偏移进行减法。 例如：如果 OFFSET1_CH[4:0]=4 且 OFFSET2_CH[4:0]=4，则这是在转换通道 4 时减去的 OFFSET1[11:0]。

13.4.19 ADC 注入数据寄存器（ADC_JDRy)

偏移地址： 0x80+0x04*(y-1)（y=1 to 4）

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15:0	JDATA[15:0]	R	16'h0	注入转换数据 这些位只读。注入通道 y 的转换结果放于此寄存器。数据是左对齐或者右对齐的。

13.4.20 ADC 模拟看门狗 2 配置寄存器（ADC_AWD2CR)

偏移地址： 0xA0

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD2CH[20:16]				
											RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AWD2CH[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20:0	AWD2CH[20:0]	RW	21'h0	模拟看门狗 2 通道选择

				这些位由软件设置和清除。它们启用并选择要由模拟看门狗 2 保护的输入通道。 AWD2CH[i]=0：AWD2 未监控 ADC 模拟输入通道 i AWD2CH[i]=1：AWD2 监控 ADC 模拟输出通道 i 当 AWD2CH[20:0]=000..0 时，模拟看门狗 2 被禁用 注意：AWD2CH 选择的通道也必须选择到 SQRi 或 JSQRi 寄存器中。 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。 某些通道未进行物理连接，不得为模拟看门狗选择。

13.4.21 ADC 模拟看门狗 3 配置寄存器（ADC_AWD3CR)

偏移地址： 0xA4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AWD3CH[20:16]				
											RW	RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AWD3CH[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20:0	AWD3CH[20:0]	RW	21’h0	模拟看门狗 3 通道选择 这些位由软件设置和清除。它们启用并选择要由模拟看门狗 3 保护的输入通道。 AWD3CH[i]=0：AWD3 未监控 ADC 模拟输入通道 i AWD3CH[i]=1：AWD3 监控 ADC 模拟输出通道 i 当 AWD3CH[20:0]=000..0 时，模拟看门狗 3 被禁用 注意：AWD3CH 选择的通道也必须选择到 SQRi 或 JSQRi 寄存器中。 注意：只有当确保没有正在进行的转换时，软件才允许写入这些位。 某些通道未进行物理连接，不得为模拟看门狗选择。

13.4.22 ADC 校准因子寄存器（ADC_CALFACT)

偏移地址： 0xB4

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCALFACT [8:0]									CALON	CAPSUC	OFFSUC	Res.	Res.	Res.	Res.
R	R	R	R	R	R	R	R	R	R	RC_W1	RC_W1				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDVLD	WRVLD	FACTSEL[4:0]					WCALFACT [8:0]								
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:23	RCALFACT [8:0]	R	9’h00	溢出标志位读寄存器； 也是读校准结果寄存器（补码表示）。 当 RDVLD 有效时，读取 ADC 校准结果。只有 ADCAL 为 0 时，软件才能读取。

				当 RDVLD 有效时，读取溢出标志位。只有 ADCAL 为 0 时，软件才能读取。
22	CALON	R	0	校准状态位。 1：ADC 正在进行校准； 0：ADC 校准已结束或未启动。
21	CAPSUC	RC_W1	0	电容校准状态位。 表示 ADC 电容校准是否成功。硬件置 1；软件写 1 置 0；
20	OFFSUC	RC_W1	0	Offset 校准状态位。 表示 ADC offset 校准是否成功。硬件置 1；软件写 1 置 0；
19:16	Reserved	-	-	保留
15	RDVLD	RW	0	溢出标志位读使能。 1：使能读溢出标志 0：复位状态 也是校准因子读使能。 1：使能读校准因子 0：复位状态
14	WRVLD	RW	0	校准因子写使能。 1：使能写校准因子 0：复位状态
13:9	FACTSEL[4:0]	RW	5’h0	写校准值,溢出标志选择位 当 RDVLD/WRVLD 有效时，选择需要读的校准结果/写入的校准因子加载到哪个电容。 {cov_error,cal_error}由硬件置位，软件清零。读{cov_error,cal_error}时需使能 RDVLD。写{cov_error,cal_error}时需使能 WDVLD，通过写 WCALFACT[1]清零 cov_error，通过写 WCALFACT[0]清零 cal_error。 5’d1: dos 5’d2: dcpm5 5’d3: dcpm4 5’d4: dcpm3 5’d5: dcpm2 5’d6: dcpm1 5’d7: dcpn4 5’d8: dcpn3 5’d9: dcpn2 5’d10: dcpn1 5’d11: dcpk3 5’d12: dcm5 5’d13: dcm4 5’d14: dcm3 5’d15: dcm2 5’d16: dcm1 5’d17: dcnm0 5’d18: {cov_error,cal_error}: cov_error: 当 AD 转换时，将校准结果补偿到 AD 转换时溢出，硬件置位该位，软件写 1 清零。 cal_error: 当校准时，校准过程中产生溢出，硬件置位该位 ，软件写 1 清零。
8:0	WCALFACT [8:0]	RW	9’h0	写校准因子寄存器（补码表示）。 当 WRVLD 有效时，WCALFACT 值加载到 ADC 中。

				注意(ADCAL 为 0 时，软件才能写入)。

13.4.23 ADC 增益补偿寄存器 (ADC_GCOMP)

偏移地址： 0xC0

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13:0	GCOMPCOEFF[13:0]	RW	14'h0	增益补偿系数 这些位由软件设置和清除，以编程增益补偿系数。 00 1000 0000 0000：增益因子为 0.5 … 01 0000 0000 0000：增益因子为 1 10 0000 0000 0000：增益因子为 2 11 0000 0000 0000：增益因子为 3 … 该系数除以 4096，得到范围从 0 到 3.999756 的增益因子。 注意：此增益补偿仅在 ADC_CFGR2 寄存器的 GCOMP 位为 1 时应用。

13.4.24 ADC 公共寄存器(ADC_CCR)

偏移地址： 0x308

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	DIFF_EN	VREFSEL	PWR_MODE[2:0]			Res.	VSENSESEL	VREFEN	PRESC[3:0]				CKMODE[1:0]	
		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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	VREFBUFSEL[2:0]			VREFBUFEN	Res.	Res.	Res.	Res.	Res.
							RW	RW	RW	RW					

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	DIFF_EN	RW	0	差分输入使能 0：单端输入 1：差分输入
28	VREFSEL	RW	0	ADC 参考电压选择 0：V_{REFP} 作为参考电压 (V_{REFP} 连接到 V_{CCA}) 1：V_{REFBUF} 作为参考电压
27:25	PWR_MODE	RW	3'h0	ADC 内比较器的功耗模式配置 000: typical 6 μA (default) 001: typical 7 μA 010: typical 8 μA

				011: typical 9 μA 100: typical 2 μA 101: typical 3 μA 110: typical 4 μA
24	Reserved	-	-	保留
23	VSENSESEL	RW	0	VTS 选择 此位由软件控制 VTS 的设置和清除。 0:温度传感器通道禁用 1:温度传感器通道启用
22	VREFEN	RW	0	V_{REFINT} 使能 该位由软件设置和清除，以启用/禁用 VREFINT 通道。 0:V_{REFINT} 通道禁用 1:V_{REFINT} 通道启用
21:18	PRESC[3:0]	RW	4'h0	ADC 预分频系数，针对异步时钟模式 由软件置 1 和清零，以选择 ADC 的时钟频率。 0000：输入 ADC 时钟未分频 0001：输入 ADC 时钟 2 分频 0010：输入 ADC 时钟 4 分频 0011：输入 ADC 时钟 6 分频 0100：输入 ADC 时钟 8 分频 0101：输入 ADC 时钟 10 分频 0110：输入 ADC 时钟 12 分频 0111：输入 ADC 时钟 16 分频 1000：输入 ADC 时钟 32 分频 1001：输入 ADC 时钟 64 分频 1010：输入 ADC 时钟 128 分频 1011：输入 ADC 时钟 256 分频 其他值：保留
17:16	CKMODE[1:0]	RW	2'h0	ADC 时钟模式 此位由软件置 1 和清零，用于定义为模拟 ADC 提供时钟的方式： 00： ADCCLK（异步时钟模式），由 RCC 配置源 01： PCLK/2（同步时钟模式） 10： PCLK/4（同步时钟模式） 11： PCLK（同步时钟模式）。使能此配置时， PCLK 的时钟占空比必须为 50%（在所有同步时钟模式下，从定时器触发到转换开始的延迟过程中，不存在抖动。 注： 仅当 ADC 已禁止时（ADCAL=0、ADSTART=0、ADSTP=0、 ADDIS=0 且 ADEN=0），才允许通过软件对这些位执行写操作。
15:9	Reserved	-	-	保留
8:6	VREFBUFSEL	RW	3'h0	V_{REFBUF} 电压档位选择 000: 2.048 V 001: 2.5 V 010: 2.9 V 011: 1.024 V 100:1.5 V

				其它:保留
5	VREFBUFEN	RW	0	V_{REFBUF} 使能，当 VREFBUFEN==0 时，电压输出高阻 软件写 0 置 0，写 1 置 1， 0：禁用 V_{REFBUF} 1：使能 V_{REFBUF}
4:0	Reserved	-	-	保留

14. 电压基准缓冲器(V_{REFBUF})

14.1 V_{REFBUF} 简介

内嵌的 V_{REFBUF} 被用作 ADC 参考电压，同时也可以通过 PD7 输出该电压。

14.2 V_{REFBUF} 功能描述

内嵌的参考电压支持五档电压，可通过 ADC_CCR 的 VREFBUFSEL 配置:

- VREFBUFSEL = 000: 2.048 V
- VREFBUFSEL = 001: 2.5 V
- VREFBUFSEL = 010: 2.9 V
- VREFBUFSEL = 011: 1.024 V
- VREFBUFSEL = 100: 1.5 V

通过置位 ADC_CCR 的 VREFBUFEN 使能 V_{REFBUF}，用户必须设置了 VREFBUFSEL，V_{REFBUF} 输出即达到期望值。

15. 比较器(COMP)

15.1 COMP 简介

芯片内集成 2 个通用比较器（General purpose comparators）COMP，分别是 COMP1 和 COMP2。这 2 个模块可以作为单独的模块，也可以与 timer 组合在一起使用。

用作比较器模拟功能的 I/O，必须在 GPIO 寄存器中被配置成模拟模式，且需要使能 SYSCFG.COMP_ANA2ENR 寄存器中的相关位。

比较器可以使用在：

- 低功耗模式 唤醒
- 模拟信号调节
- 和来自 timer 的 PWM 输出连接时，Cycle by cycle 的电流控制

15.2 COMP 主要特性

- 支持电压比较功能，每个比较器有可配置的正极或者负极输入，以实现灵活的电压选择
  - 多路 I/O pin
  - V_{CCA}/V_{REFBUF} 的 256 档分压（即 V_{REFCMP}）
  - 温度传感器输出
- 可编程速度和功耗
- 可编程迟滞功能
- 配置寄存器写保护(LOCK 功能)
- 输出可以被连接到 I/O 或者 timer 的输入
  - OCREF_CLR 事件（cycle by cycle 的电流控制）
  - 为快速 PWM shutdown 的刹车
- 每个 COMP 具有中断产生能力，用作芯片从低功耗模式（睡眠/低功耗睡眠/停止模式）唤醒（通过 EXTI)
- 提供软件可配置数字滤波时间以增强芯片抗干扰能力
- 支持输出消隐以降低开关噪声
- 支持窗口比较功能

15.3 COMP 功能描述

15.3.1 COMP 模块框图

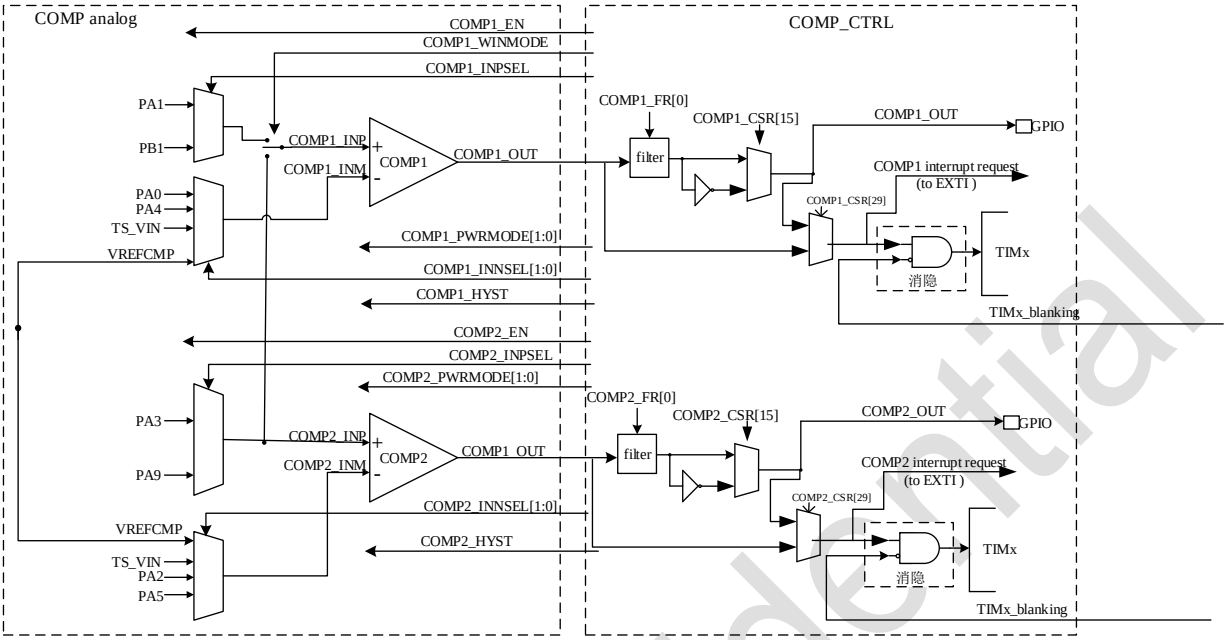


图 15-1 COMP 结构框图

15.3.2 COMP 的输入和输出

比较器输出可以通过在 GPIO 的可选功能（alternate function）连接到 I/O 引脚。

输出也可以在内部连接到各种 timer 的输入，达到以下目的:

- 连接刹车输入时，PWM 信号的紧急 shut-down 刹车
- 使用 OCREF_CLR 输入的 Cycle-by-cycle 电流控制
- 时序测量的输入捕获

比较器输出到其他模块的连接关系，见下表。

表 15-1 COMP 模块数字输出

COMP1_OUT	COMP2_OUT
PA0	PA2
PA6	PA7
PA11	PA12
PB8	PB9
EXTI18	EXTI19
lptim1_ext_trig2	lptim1_ext_trig3
tim1_bk	tim1_bk
tim15_bk	tim15_bk
tim16_bk	tim16_bk
tim17_bk	tim17_bk
tim2_ti4_in1	tim2_ti4_in2
tim2_ti2_in1	tim2_ti2_in2

tim3_ti2_in1	tim3_ti2_in2
tim4_ti2_in1	tim4_ti2_in2
tim1_ti1_in1	tim15_ti2_in1
tim2_ti1_in1	tim1_ti1_in2
tim3_ti1_in1	tim2_ti1_in2
tim4_ti1_in1	tim3_ti1_in2
tim15_ti1_in2	tim4_ti1_in2
tim1_ocref_clr0	tim15_ti1_in3
tim2_ocref_clr0	tim1_ocref_clr1
tim3_ocref_clr0	tim2_ocref_clr1
tim4_ocref_clr0	tim3_ocref_clr1
tim15_ocref_clr0	tim4_ocref_clr1
tim16_ocref_clr0	tim15_ocref_clr1
tim17_ocref_clr0	tim16_ocref_clr1
tim1_etr1	tim17_ocref_clr1
tim2_etr1	tim1_etr2
tim3_etr1	tim2_etr2
tim4_etr1	tim3_etr2
-	tim4_etr2

15.3.3 COMP 的时钟和复位

COMP 模块有两个时钟源：

1. PCLK（APB clock），用于配置寄存器

2. COMP 时钟，用于模拟比较器输出后的电路（模拟输出的锁存电路、滤毛刺电路等）的时钟，可选择为 PCLK、LSE 或者 LSI。当需要在 Stop 模式下工作时，选择 LSE 或者 LSI（通过 RCC_CCIPR.COMPxSEL 进行选择)。

注意：

1．PCLK 和 COMP CLK 由 RCC_APB2ENR.COMPEN 位同时使能，该使能关闭则 PCLK 和 COMP CLK 关闭，若使能则 PCLK 和 COMP CLK 同时开启。

2．进入 Stop 模式前，建议先配置 COMP CLK 为 LSI 或 LSE；然后进入 stop 模式。

COMP 模块的复位信号包含 APB 复位源和 COMP 模块软件复位源

1．APB 复位，用于 COMP 寄存器的复位

2．COMP 软件复位，用于模拟比较器输出后电路（模拟输出的锁存电路、滤毛刺电路等）的复位

注意：当 RCC_APB2RSTR 中的复位信号使能后，COMP 整个模块都会被复位，包括寄存器和用于模拟比较器输出后的电路。

15.3.4 COMP 的滤波功能

如果芯片的工作环境恶劣，比较器的输出会出现噪声信号。使能数字滤波模块，则比较器的输出波形中脉宽小于 COMPx_FR.FLTCNT[15:0]设定时间的噪声信号都可以被滤除。禁止数字滤波模块，则数字滤波模块的输出信号与输入信号相同。

注意：设置 COMP 滤波时间，开启滤波使能应在 COMPx_CSR.EN 使能前完成。

滤波示意如下所示：

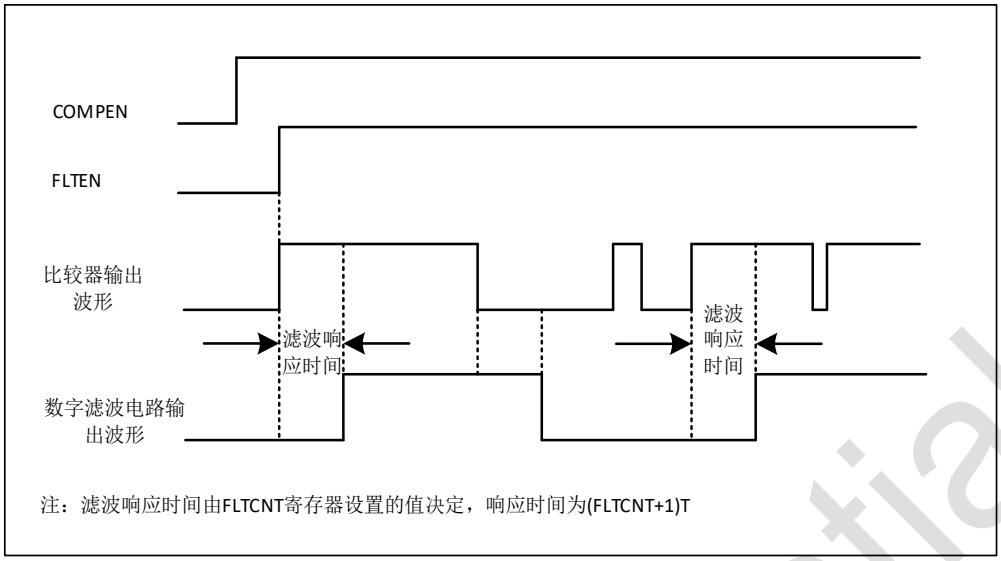


图 15-2 COMP 滤波示意图

15.3.5 COMP 的锁定机制

比较器可以用于安全目的，例如过电流或热保护。对于具有特定功能安全要求的应用程序，有必要确保比较器编程在伪寄存器访问或程序反损坏的情况下不会被更改。为此，比较器控制寄存器和状态寄存器可以被写保护（只读）。一旦编程完成，就可以设置 COMPx_CSR.LOCK 位。这导致整个寄存器变为只读，包括 COMPx_CSR.LOCK 位。写入保护只能通过 MCU 复位来移除。

15.3.6 COMP 的输出消隐

消隐功能的目的是防止电流在调节 PWM 周期开始时因短电流尖峰而跳闸（通常是电源开关反并联二极管中的恢复电流）。通过设置定时器的输出比较信号来实现消隐功能。每个比较器通道的消隐源由软件通过相应 COMPx_CSR 寄存器的 BLANKSEL[2:0]位单独选择。反相消隐信号与比较器输出进行逻辑“与”运算，以产生比较器通道的输出。

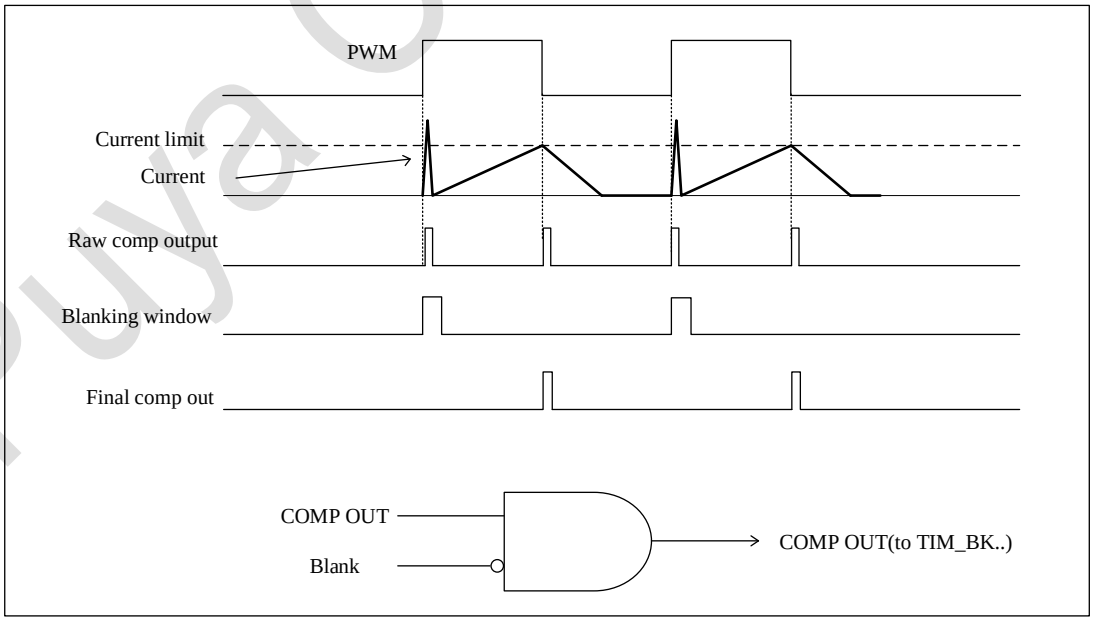


图 15-3 比较器消隐

15.3.7 COMP 的窗口模式

处于窗口模式的比较器简称为窗口比较器，窗口比较器的作用是监测模拟电压是否在低和高阈值范围内。可以使用 COMP1 和 COMP 2 创建窗口比较器。被监测的模拟电压同时连接到两个比较器的正输入端，高阈值和低阈值分别连接到两个比较器的负输入端。

通过使能 COMP1_CSR.WINMODE 位，可以将 COMP1 和 COMP2 的正输入端连接到一起，起到节省一个 I/O 引脚的作用。

例如当比较器 1 的 WINMODE 被置 1 时，比较器 1 的正输入端将接入 COMP2_INP，当 WINMODE 被清 0 时，比较器的正输入端将接入 COMP1_INP。这样做可以省掉一个 IO 的引脚。

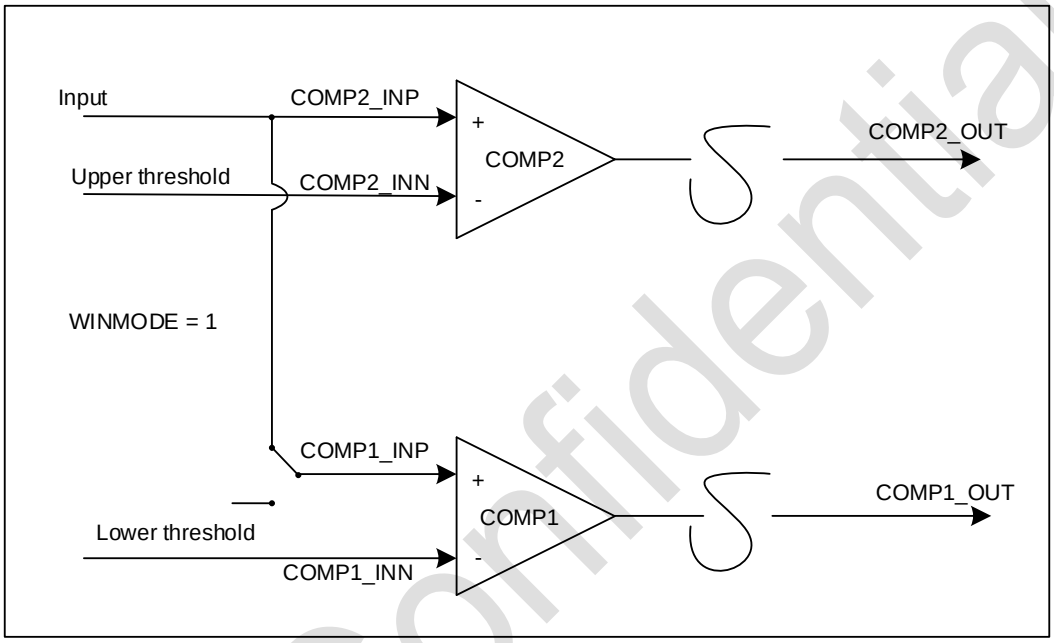


图 15-4 窗口比较器

15.3.8 COMP 的迟滞功能

比较器具有可编程的迟滞，可避免在有噪声信号的情况下出现虚假的输出跳变。 如果不需要迟滞模式（例如，从低功耗模式退出时）则可以将其禁用，以便能够使用外部组件强制迟滞值。COMP1 和 COMP2 有独立的迟滞功能使能信号 COMP1_HYST 和 COMP2_HYST。

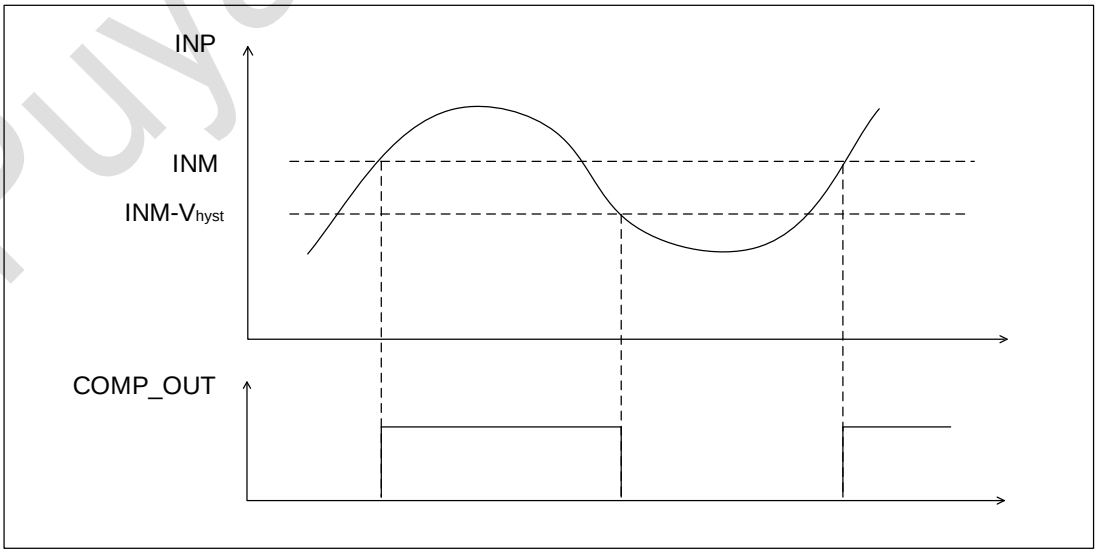


图 15-5 比较器的迟滞示意图

15.3.9 COMP 的功耗模式

比较器的功耗和传输延迟可以被调整，实现在特定应用的最适合的 trade-off。COMPx_CSR 寄存器的 PWRMODE[1:0]位可以被用作该功能的选择。注意，当进入 Stop 之前，如果选择 PWR_CR2 寄存器 LPR=1（即选择用 Low power regulator 供电），则需要首先设定 COMP 在 Medium speed (PWRMODE=01)。

此外，为降低功耗，APB 时钟和 COMP 时钟被 RCC_APBENR2.COMP1EN 和 RCC_APBENR2.COMP2EN 控制，软件可在使用 COMP 模块时，才使能该寄存器。

15.3.10 COMP 的低功耗模式

表 15-2 低功耗模式下的比较器特性

模式	描述
睡眠模式	对比较器无影响。 比较器中断可以使设备退出睡眠模式
低功耗运行模式	无影响
低功耗睡眠模式	无影响。 比较器中断可以使设备退出低功耗睡眠模式
停止模式 0	对比较器无影响。 比较器中断可以使设备退出停止模式
停止模式 1	对比较器无影响。 比较器中断可以使设备退出停止模式
停止模式 2	比较器不工作

15.3.11 COMP 选择 V_{CC}/V_{REFBUF} 配置

比较器选择 V_{CC}/V_{REFBUF} 作为比较器分压源时，软件配置流程

- 选择 V_{REFBUF} 作为参考源时
  - 配置 ADC 模块的 ADC_CCR 中 VREFEN
  - 配置 ADC 模块的 ADC_CCR 中 VREFBUFSEL [2:0]，选择 V_{REFBUF} 档位
  - 配置 ADC 模块的 ADC_CCR 中 VREFBUFEN，使能 V_{REFBUF} 电压
  - 配置 COMP1_CSR 中的 COMP_VCSEL 为 0，选择 V_{REFBUF}
  - 配置 COMP1_CSR 中的 COMP_VCDIV[7:0]，选择分压档位
- 选择 V_{CC} 作为参考源时
  - 配置 COMP1_CSR 中的 COMP_VCSEL 为 1，选择 V_{CC}
  - 配置 COMP1_CSR 中的 COMP_VCDIV[7:0]，选择分压档位

15.4 COMP 的中断

比较器的输出在芯片内部连接到 EXTI 控制器（extended interrupts and events）。每个比较器有它自己的 EXTI 线，并能够产生中断或者事件。相同的机制被用作低功耗唤醒。

15.5 COMP 寄存器

15.5.1 COMP1 控制和状态寄存器(COMP1_CSR)

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
LOCK	COMP1_OUT	TIM_EXTI_OUT_SEL	COMP_VCSEL	COMP_VCDIV[7:0]								PWRMODE[1:0]		COMP_VCDIV_EN	HYST
RW	R	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
POLARITY	BLANKSEL[2:0]			WINMODE	Res.	INPSEL	Res.	Res.	Res.	Res.	INNSEL[1:0]		Res.	Res.	EN
RW	RW	RW	RW	RW		RW					RW	RW			RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP1_CSR 寄存器 lock 该 bit 软件设置，硬件系统复位或模块软复位清除。锁住整个比较器 1 的 COMP1_CSR，当锁住时，所有 bit 和 flag 只能读不能写，当解锁后，软件可以写控制位。 0：未锁定 1：锁定
30	COMP1_OUT	R	0	COMP1 输出状态 该位只读，它反映了 COMP1 在经过极性选择的输出电平。
29	TIM_EXTI_OUT_SEL	RW	0	COMP1 输出到 TIM/EXTI 选择。 0：不经过滤波，模拟端比较结果（可以消隐）直接输出到 TIM/EXTI 1：比较器内部数字输出选择与外部输出一致（可以滤波，可以消隐）
28	COMP_VCSEL	RW	0	V_{REFCMP} 电压源选择信号 0：分压源来自 V_{REFBUF} 1：分压源来自 V_{CC}
27:20	COMP_VCDIV	RW	8’h0	V_{REFCMP} 输出电压选择信号 00000000：1/256 的分压源(V_{REFBUF}/V_{CC}) 00000001：2/256 的分压源(V_{REFBUF}/V_{CC}) 00000010：3/256 的分压源(V_{REFBUF}/V_{CC}) … 11111110：255/256 的分压源(V_{REFBUF}/V_{CC}) 11111111：256/256 的分压源(V_{REFBUF}/V_{CC})
19:18	PWRMODE	RW	2’h0	COMP1 功耗模式选择 00：高速 01：中速 10：保留 11：保留
17	COMP_VCDIV_EN	RW	0	V_{REFCMP} 使能信号 0：关闭 V_{REFCMP} 1：使能 V_{REFCMP}
16	HYST	RW	0	COMP1 迟滞功能使能控制 0：关闭迟滞功能 1：使能迟滞功能
15	POLARITY	RW	0	COMP1 输出极性选择

				软件可读可写 0：不反向 1：反向
14:12	BLANKSEL	RW	3’h0	比较器 1 消隐信号选择 该位由软件选择比较器 1 的消隐源。 001：tim1_oc5 010：tim2_oc3 011：tim3_oc3 100：tim4_oc3 101：tim15_oc1 其它：保留
11	WINMODE	RW	0	COMP1 窗口模式使能 0：关闭窗口模式； 1：使能窗口模式，COMP1 的正极输入连接 COMP2 的正极输入
10	Reserved	-	-	保留
9	INPSEL	RW	0	COMP1 正极输入选择 0：PA1 1：PB1
8:5	Reserved	-	-	保留
4:3	INNSEL	RW	2’h0	COMP1 负极输入选择 00：PA0 01：PA4 10：V_{REFCMP} 11：TS_VIN
2:1	Reserved	-	-	保留
0	EN	RW	0	COMP1 使能位 软件可读可写（如果没有被锁定） 0：关闭 COMP1 1：使能 COMP1

15.5.2 COMP1 滤波寄存器 (COMP1_FR)

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
FLTCNT1[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**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN1
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT1	RW	16’h0	比较器 1 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved	-	-	保留

0	FLTEN1	RW	0	比较器 1 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注意：该位必须在 COMP1 的 EN 为 0 时置位

15.5.3 COMP2 控制和状态寄存器(COMP2_CSR)

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
LOCK	COMP2_OUT	TIM_EXTI_OUT_SEL.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWRMODE[1:0]		Res.	Res.
RW	R	RW										RW	RW		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POLARITY	BLANKSEL[2:0]			Res.	Res.	INPSEL	Res.	Res.	Res.	Res.	INNSEL[1:0]		Res.	HYST	EN
RW	RW	RW	RW			RW					RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	COMP2_CSR 寄存器 lock 软件置位，系统复位清零。当被置位，则会锁定 COMP2_CSR 寄存器的所有 32 位 0：未锁定，可读写整个寄存器 1：锁定，整个寄存器只读
30	COMP2_OUT	R	0	COMP2 输出状态 该位只读，它反映了 COMP2 在经过极性选择的输出电平。
29	TIM_EXTI_OUT_SEL	RW	0	COMP2 输出到 TIM/EXTI 选择。 0：不经过滤波，ana_out（可以消隐）直接输出到 TIM/EXTI 1：选择输出到 PAD 上的 comp_out（可以滤波，可以消隐）输出到 TIM/EXTI
28:20	Reserved	-	-	保留
19:18	PWRMODE	RW	2’h0	COMP2 功耗模式选择 选择了功耗和由此而来的 COMP2 的速度 00：高速 01：中速 10：保留 11：保留
17:16	Reserved	-	-	保留
15	POLARITY	RW	0	COMP2 极性选择 0：不反向 1：反向
14:12	BLANKSEL	RW	3’h0	比较器 2 消隐信号选择 该位由软件选择比较器 2 的消隐源。 001:tim1_oc5 010:tim2_oc3 011:tim3_oc3 100:tim4_oc3 101:tim15_oc1 其它:保留

11:10	Reserved	-	-	保留
9	INPSEL	RW	0	COMP2 非反向输入的信号选择 0：PA3 1：PA9
8:5	Reserved	-	-	保留
4:3	INNSEL	RW	2'h0	COMP2 反向输入的信号选择 00:PA2 01:PA5 10:V_{REFCMP} 11:TS_VIN
2	Reserved	-	-	保留
1	HYST	RW	0	COMP2 迟滞功能使能控制 0：迟滞功能关闭 1：迟滞功能使能
0	EN	RW	0	COMP2 使能位 0：关闭 COMP2 1：使能 COMP2

15.5.4 COMP2 滤波寄存器(COMP2_FR)

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
FLTCNT2[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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN2
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT2	RW	16'h0	比较器 2 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved	-	-	保留
0	FLTEN2	RW	0	比较器 2 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注: 该位必须在 COMP2 的 EN 为 0 时置位

16. 运算放大器(OPA)

16.1 OPA 简介

内嵌 2 个运算放大器，每个运算放大器有两个输入和一个输出。三个 I/O 可以连接到外部引脚，从而实现任何类型的外部互连，也可以将输出连接到内部 ADC。其中 OPA2 可以当作 COMP 使用，通过 OPA2_CSR. OPA_CMP_CR 位控制。

16.2 OPA 主要特性

- 轨对轨输入和输出电压范围
- 低输入偏置电流
- 低输入偏移电压
- 高频增益带宽

16.3 OPA 功能描述

OPA 只有独立模式；禁用时输出为高阻抗。

16.3.1 模块框图

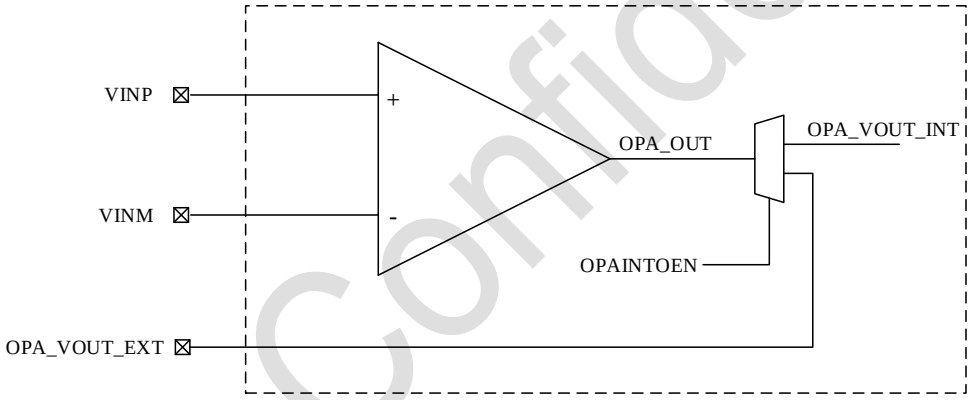


图 16-1 OPA 模块运算放大器模式框图

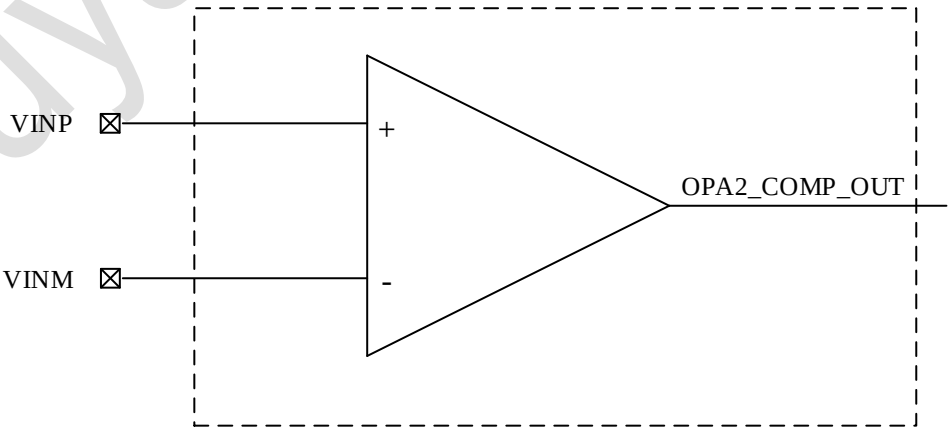


图 16-2 OPA 模块比较器模式框图

16.3.2 OPA 输出重定向到内部 ADC 通道

通过设置 OPAx_CSR 寄存器中的 OPAINTOEN 位，运算放大器输出可以在内部重定向到 ADC 通道。在这种情况下，映射到 OPAx_VOUT 的 GPIO 被释放可以用于其他目的。ADC 可以在内部测量 OPAx_VOUT 电压。

16.3.3 OPA 复位和时钟

RCC 提供的 OPA 时钟与 PCLK2（APB2 时钟）同步。在 RCC 控制器中提供时钟使能控制位。OPAx_CSR.OPAEN 位用来启用和禁用 OPA 模块。如果需要更改 OPA 寄存器配置，必须在使能 OPAEN 位之前，以避免对输出产生影响。当不再需要运算放大器的输出时，可以禁用 OPA 以节省功耗。禁用 OPA 时，将保留先前设置的所有配置。当 OPA2 需要用作 COMP 功能时，RCC 提供了可以用于滤波的时钟，该时钟可以通过 RCC_CCIPR.OPA2SEL 位来选择是 APB 时钟、LSI 时钟还是 LSE 时钟，该时钟只有在 OPA2_CSR. OPA_CMP_CR 位置位时，才可以通过 RCC_APB2ENR.OPAEN 位来使能。

16.3.4 初始配置

运算放大器的默认配置是一种功能模式，其中三个输入/输出连接到外部引脚。

一旦设置了 OPAx_CSR 寄存器中的 OPAEN 位，运算放大器即可工作。两个输入引脚和输出引脚按照下表中的定义进行连接。

注意：输入和输出引脚必须在相应的 GPIOx_MODER 寄存器配置为模拟模式（默认状态）。

16.3.5 信号连接

运算放大器引脚的连接由 OPAx_CSR 寄存器决定。下表中描述了 2 个运算放大器（OPAx，x=1…2）的连接。

表 16-1 运算放大器可能的连接

Signal	IO pin	Internal	Comment
OPA1_VINM	PA3	-	-
OPA1_VINP	PA1	-	-
OPA1_VOUT	PA2	ADC1_CH16	由 OPA1_CSR 寄存器中的 OPAINTOEN 位控制
OPA2_VINM	PC5	-	-
OPA2_VINP	PA7	-	-
OPA2_VOUT	PA6	ADC1_CH17	由 OPA2_CSR 寄存器中的 OPAINTOEN 位控制

16.3.6 OPA 模式

运算放大器的输入和输出都可以通过 GPIO 访问。运算放大器可用于多种配置环境：

- 独立模式(外部增益设置模式)
- 比较器模式

独立模式(外部增益设置模式)

首先从 OPAx_CSR 默认值和 GPIOx_MODER 默认状态开始，一旦设置了 OPAx_CSR.OPAEN 位，两个输入引脚和输出引脚就连接到运算放大器。

此默认配置使 OPA 在正常模式下运行。

比较器模式

下文介绍在比较器模式下使用 OPA 流程。

注意只有 OPA2 支持比较器模式。

1) 配置 OPA2_CSR. OPA_CMP_CR 为 1

2) 如果需要对比较结果进行滤波，则根据滤波需求配置 RCC_CCIPR.OPA2SEL，选择滤波用的时钟，配置 OPA2_FR. FLTCNT，选择需要滤波的长度，最后使能 OPA2_FR. FLTEN

3) 配置 OPA2_CSR. OPAEN 为 1

配置完成后，比较器模式下 OPA2 的比较输出结果可以通过寄存器 OPA2_CSR. OPA_CMP_OUT 位查看，也可以通过 IO 引脚输出。

表 16-2 OPA2 模块比较结果 IO 引脚输出映射

OPA2_COMP_OUT
PA1
PA5
PA13
PB10

16.3.7 OPA 低功耗模式

表 16-3 低功耗模式对 OPA 的影响

模式	描述
睡眠模式	无影响
低功耗运行模式	无影响
低功耗睡眠模式	无影响
停止模式	OPA 不工作

16.4 OPA 中断

OPA 工作在运算放大器模式时，是没有中断产生的，只有 OPA2 工作在 COMP 模式时，当置位了 OPA2 的中断使能信号后，在经过滤波(如果置位了滤波使能信号)后的比较结果由低电平变为高电平时，中断标志位置位，此时产生 OPA2 的中断，写 OPA2_INTR.CINTIF 位可以清除中断标志位。

16.5 OPA 寄存器

16.5.1 OPA1 控制/状态寄存器（OPA1_CSR）

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
LOCK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPAINTOEN	Res	Res.	Res.	Res.	Res.	Res.	Res.	OPAEN
							RW								RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	该 bit 写一次。它是由软件设置的。它只能通过系统复位或模块软复位来清除。 此位用于将 OPA1_CSR 寄存器配置为只读。 0：OPA1_CSR 可读写 1：OPA1_CSR 只读
30:9	Reserved	-	-	保留
8	OPAINTOEN	RW	0	OPA1 内部输出使能 。

				0：OPA1 输出连接到输出 pin 1：OPA1 输出连接到一个 ADC 通道，同时连接到输出引脚。
7:1	Reserved	-	-	保留
0	OPAEN	RW	0	OPA1 使能 0：禁止 OPA1 1：使能 OPA1

16.5.2 OPA2 控制/状态寄存器（OPA2_CSR）

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
LOCK	OPA_CMP_OUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
RW	R														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OPAINTOEN	Res.	Res.	Res.	Res.	Res.	Res.	OPA_CMP_CR	OPAEN
							RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31	LOCK	RW	0	该 bit 写一次。它是由软件设置的。它只能通过系统复位或模块软复位来清除。 此位用于将 OPA2_CSR 寄存器配置为只读。 0:OPA2_CSR 可读写 1:OPA2_CSR 只读
30	OPA_CMP_OUT	R	0	OPA2 输出状态 该位只读，它反映了 OPA2 切换为 COMP 功能后 OPA2 的比较结果输出电平。
29:9	Reserved	-	-	保留
8	OPAINTOEN	RW	0	OPA2 内部输出使能 。 0：OPA2 输出连接到输出 pin 1：OPA2 输出连接到一个 ADC 通道，同时连接到输出引脚。
7:2	Reserved	-	-	保留
1	OPA_CMP_CR	RW	0	OPA 与 COMP 功能控制使能： 0：OPA 功能 1：COMP 功能
0	OPAEN	RW	0	OPA2 使能 0：禁止 OPA2 1：使能 OPA2

16.5.3 OPA2 滤波寄存器（OPA2_FR）

偏移地址 0x20

复位值 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLTCNT[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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FLTEN
															RW

Bit	Name	R/W	Reset Value	Function
31:16	FLTCNT	RW	16'h0	OPA2 采样滤波计数器 采样时钟为 APB 或 LSI 或 LSE。 滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期=FLTCNT[15:0]
15:1	Reserved	-	-	保留
0	FLTEN	RW	0	OPA2 数字滤波功能配置 0：禁止数字滤波功能 1：使能数字滤波功能 注: 该位必须在 OPA_CMP_CR 位为 1 且 OPA2 的 EN 位为 0 时置位

16.5.4 OPA2 中断寄存器（OPA2_INTR）

偏移地址 0x24

复位值 0x00000000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	CINTIF	W	0	OPA2 中断标志清零 0：无影响 1：清零 OPA2 中断标志
1	INTIF	R	0	OPA2 中断标志位
0	INTEN	RW	0	OPA2 中断使能位 0：屏蔽 OPA2 中断 1：使能 OPA2 中断

17. 高级定时器（TIM1）

17.1 TIM1 简介

高级控制定时器（TIM1）由一个 16 位的自动装载计数器组成，计数器由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。高级控制定时器（TIM1）和通用定时器（TIMx）是完全独立的，它们不共享任何资源。它们可以同步操作。

17.2 TIM1 主要特征

TIM1 定时器的功能包括：

- 16 位向上、向下、向上/下的自动装载计数器
- 16 位可编程(可以实时修改)的预分频器，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：
  - 输入捕获（5/6 通道不支持）
  - 输出比较
  - PWM 生成(边缘或中间对齐模式)
  - PWM 移相（CH1/CH2/CH3 支持此功能）
  - 单脉冲模式输出
- 带有可编程死区时间的互补输出
- 通过外部信号来控制定时器与定时器之间互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
- 支持针对定位的增量(正交)编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

17.2.1 TIM1 模块框图

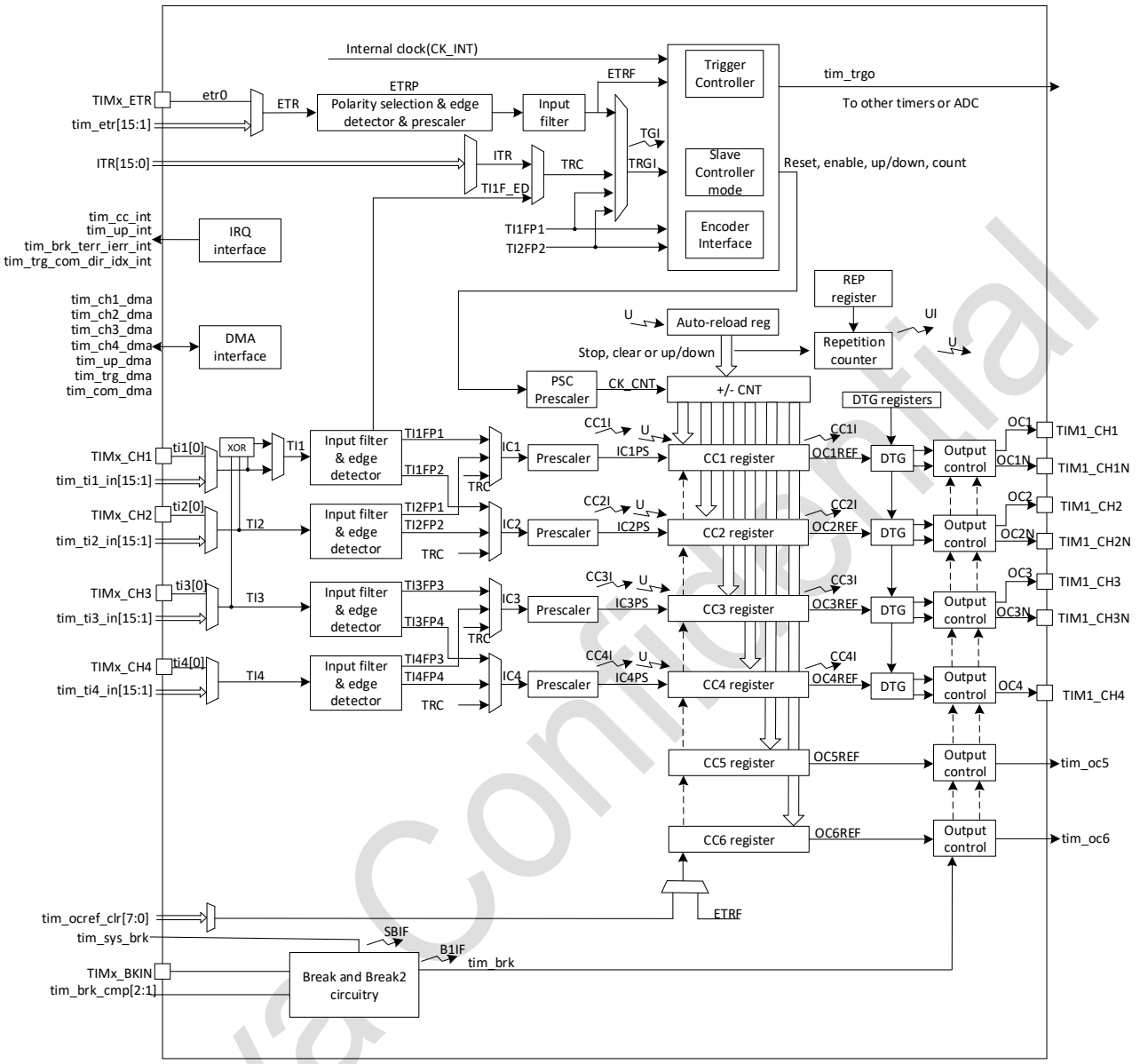


图 17-1 TIM 功能框图

17.3 TIM1 功能描述

17.3.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。计数器的时钟可以被预分频器分频。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器（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）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问它的预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件（UEV)时传送到影子寄存器。当计数器达到溢出条件(上溢或者下溢)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，会产生更新事件。更新事件也可以由软件及其他条件产生。后续会详细描述每一种配置下更新事件的产生。

计数器由预分频器分频后的时钟输出 CK_CNT 驱动，仅当设置了 TIMx_CR1 寄存器中的计数器使能位(CEN)，CK_CNT 才对计数器有效。(更多有关使能计数器的细节，请参见从模式控制器的描述)。

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

预分频器描述

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

下面几张图给出了在预分频器运行时，更改计数器参数的例子。

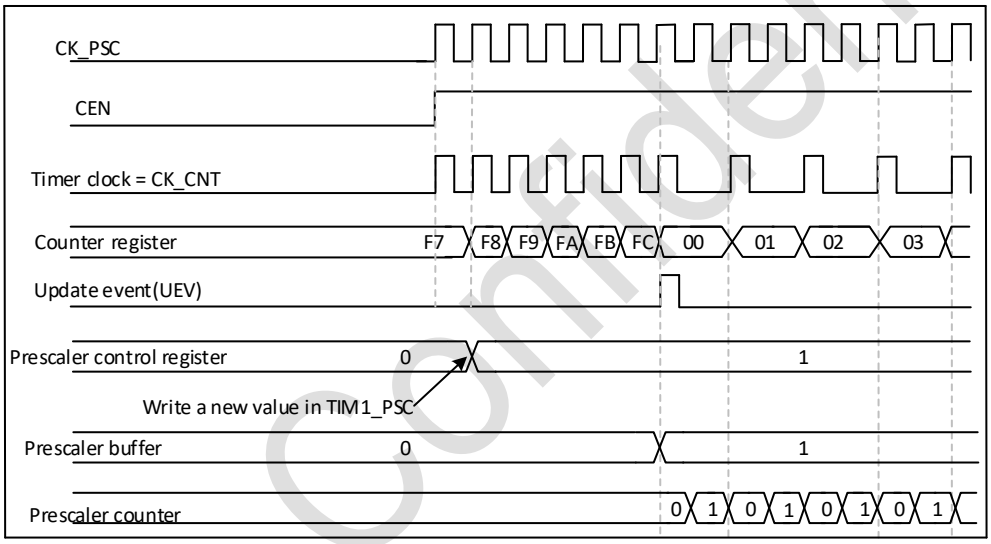


图 17-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

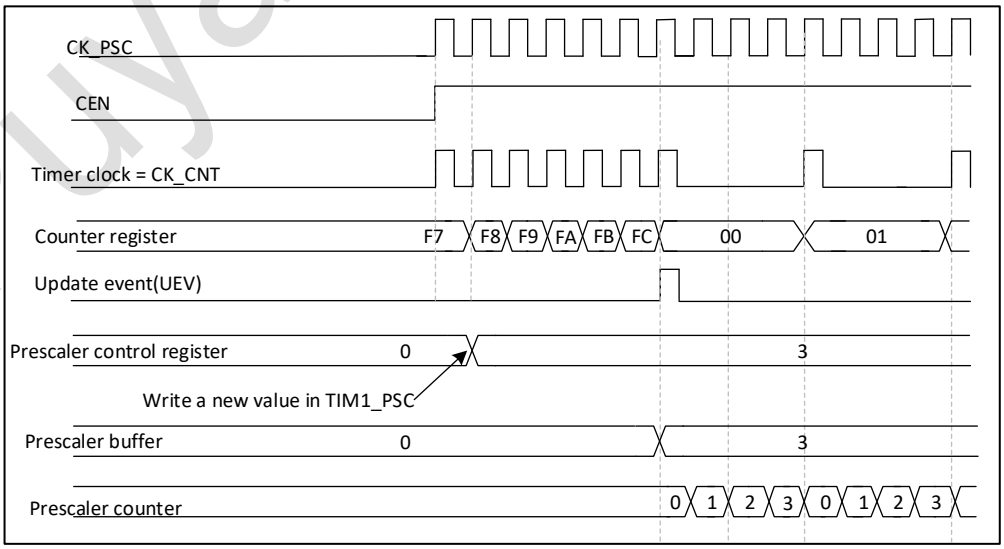


图 17-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

17.3.2 计数器模式

向上计数模式

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

如果使用了重复计数器，那么更新事件(UEV)需要在上溢次数达到所配置的重复计数寄存器的值加一（即 TIMx_RCR+1）时才会产生；如果没有使用重复计数器（即 TIMx_RCR=0），那么每次计数上溢都会产生更新事件。

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

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

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

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

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

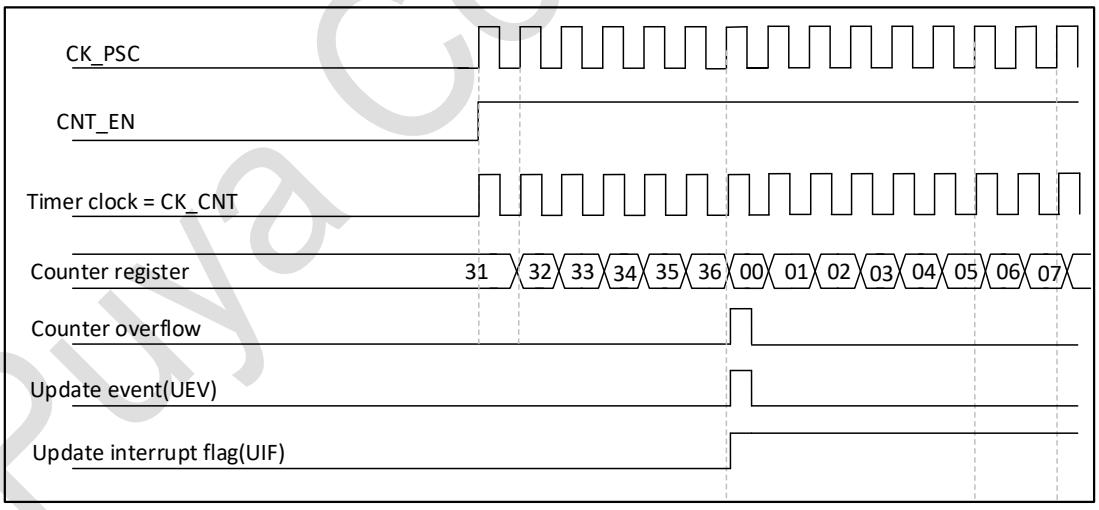


图 17-4 计数器时序图，内部时钟分频因子为 1

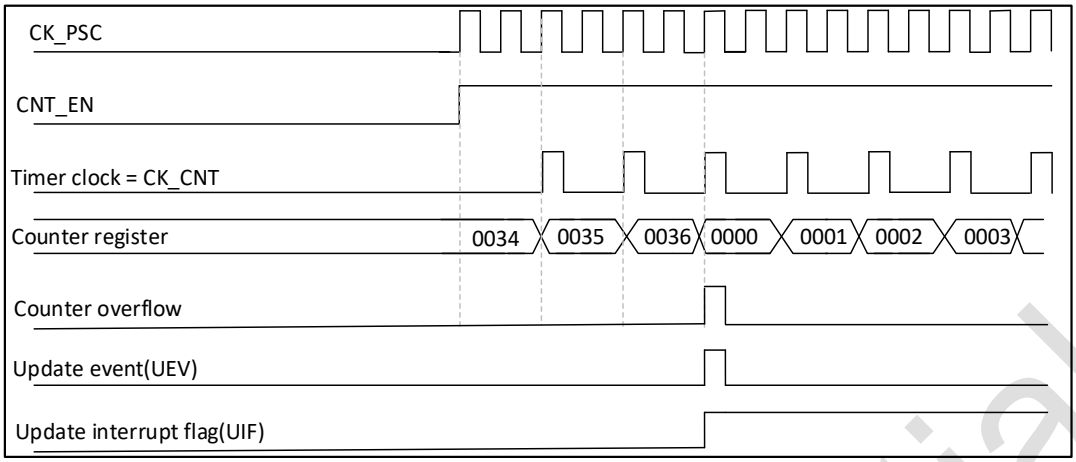


图 17-5 计数器时序图，内部时钟分频因子为 2

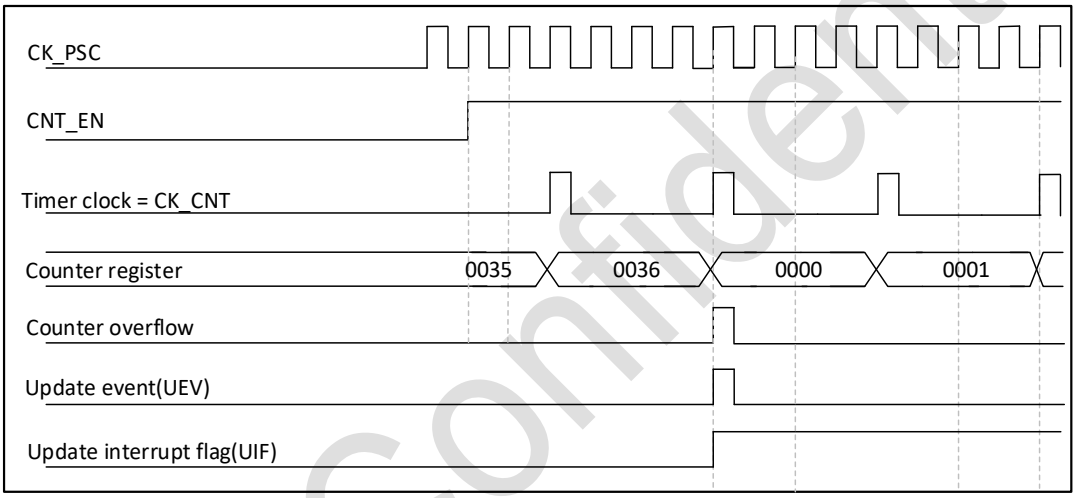


图 17-6 计数器时序图，内部时钟分频因子为 4

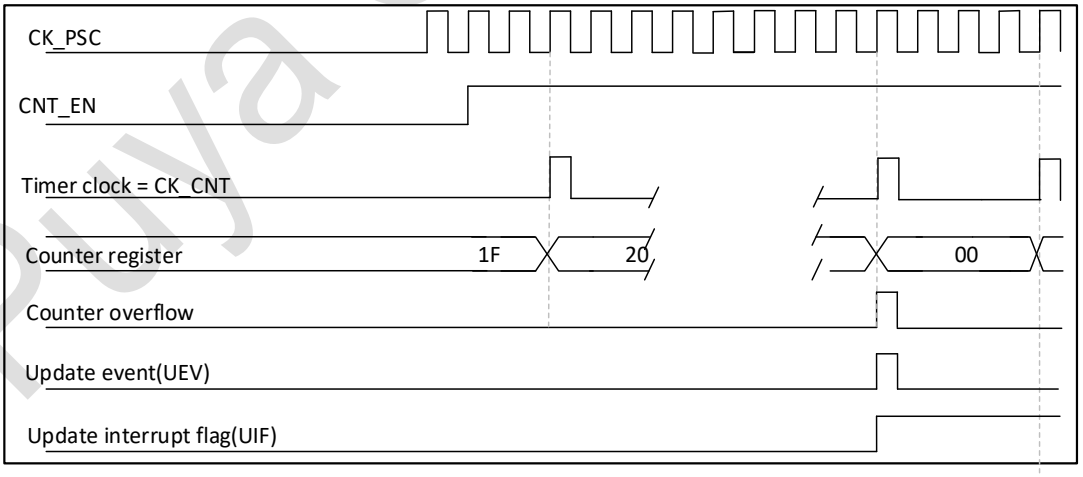


图 17-7 计数器时序图，内部时钟分频因子为 N

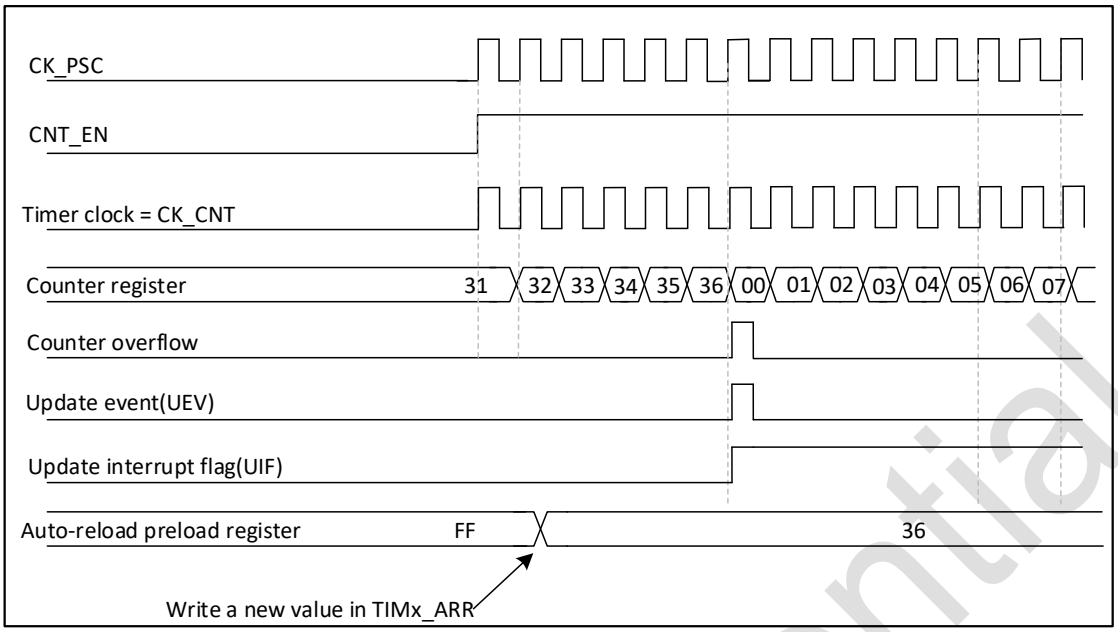


图 17-8 计数器时序图，当 ARPE=0 时的更新事件（没有预装 TIMx_ARR）

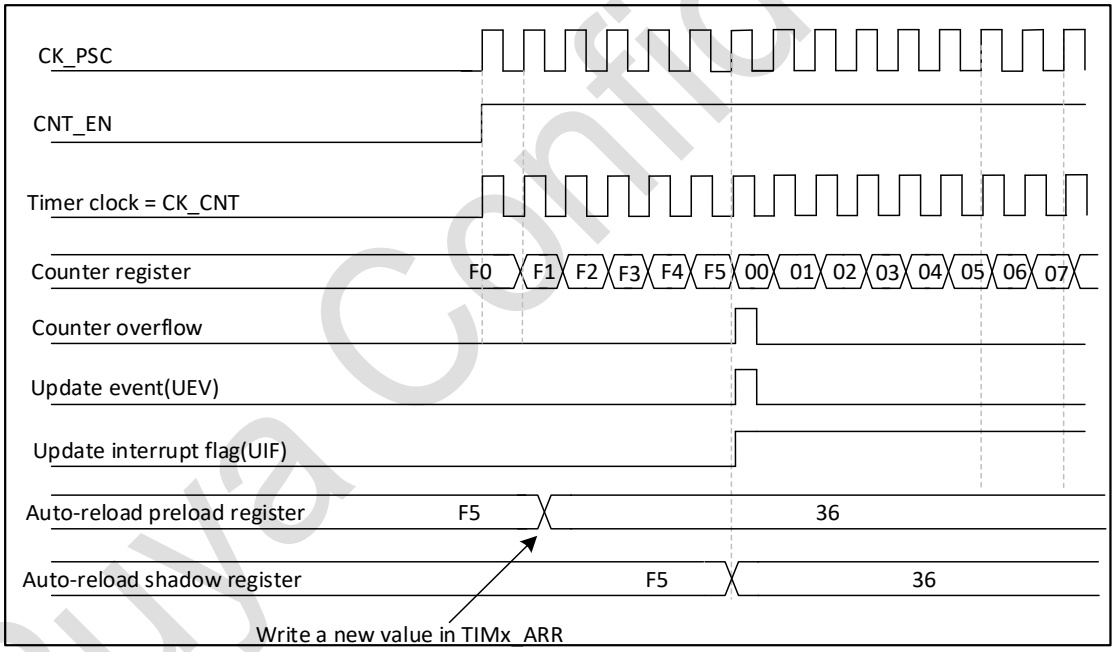


图 17-9 计数器时序图，当 ARPE=1 时的更新事件（预装了 TIMx_ARR）

向下计数模式

在向下计数模式中，计数器从自动加载值（TIMx_ARR 的内容）开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数下溢事件。

如果使用了重复计数器，那么更新事件（UEV)需要在下溢次数达到所配置的重复计数寄存器的值加一（即 TIMx_RCR+1）时才会产生；如果没有使用重复计数器（即 TIMx_RCR=0），那么每次计数下溢都会产生更新事件。

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

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会从当前自动加载值重新开始计数，同时预分频器内部的计数器被清‘0’（但预分频系数不变）。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源），通过设置 UG 位可以产生一个更新事件 UEV，但不置起 UIF 标志位(即不会产生中断或 DMA 请求)，这是为了避免在发生捕获事件时清除计数器，同时产生更新和捕获中断。

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

- 重复计数器被重新加载为 TIMx_RCR 寄存器中的内容。
- 预分频器的缓冲区被置入预装载寄存器的值（TIMx_PSC 寄存器的内容）。
- 当前的自动加载寄存器被更新为预装载值（TIMx_ARR 寄存器中的内容）。

注：自动加载值在计数器重载入之前被更新，因此下一个周期将是预期的值。

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

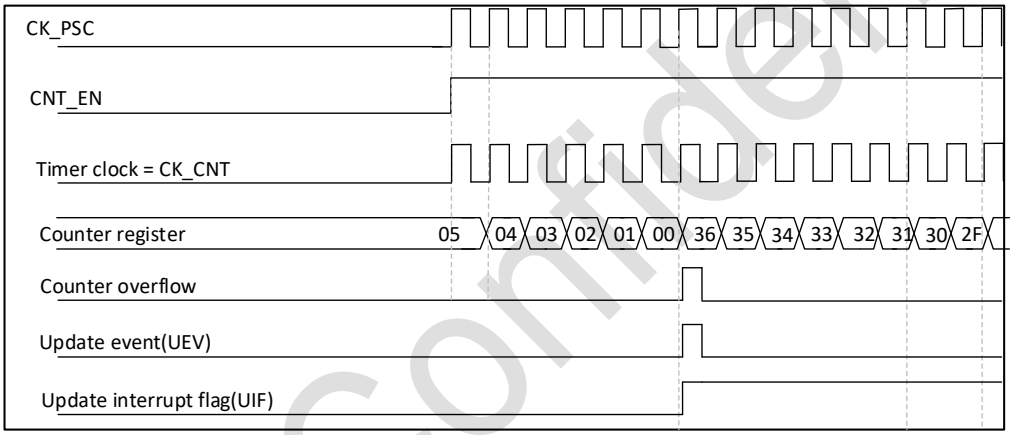


图 17-10 计数器时序图，内部时钟分频因子为 1

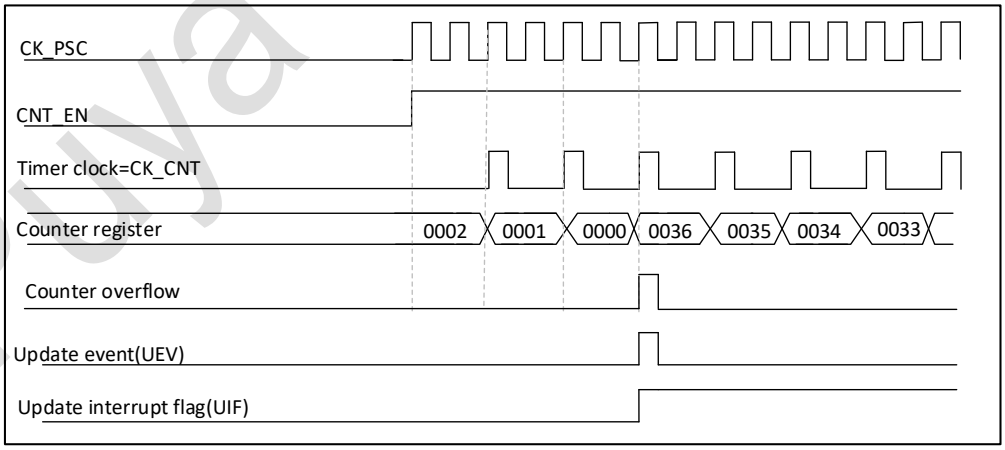


图 17-11 计数器时序图，内部时钟分频因子为 2

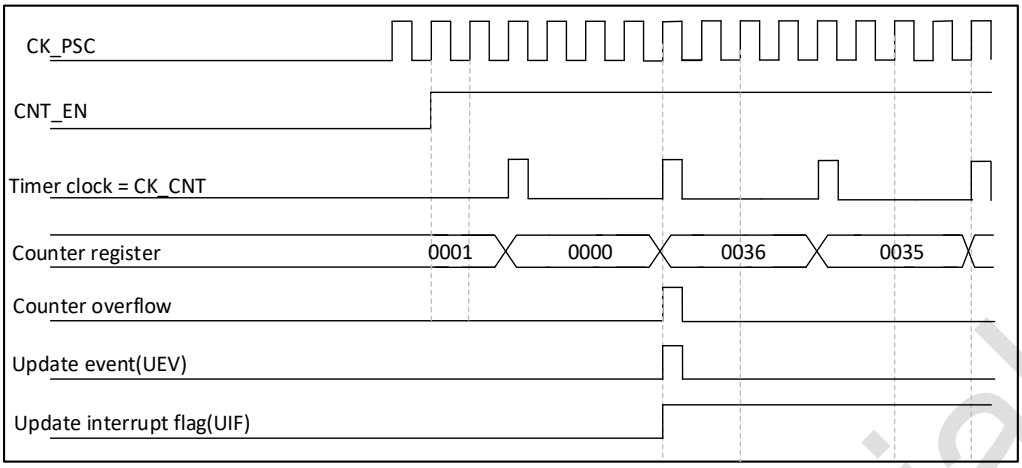


图 17-12 计数器时序图，内部时钟分频因子为 4

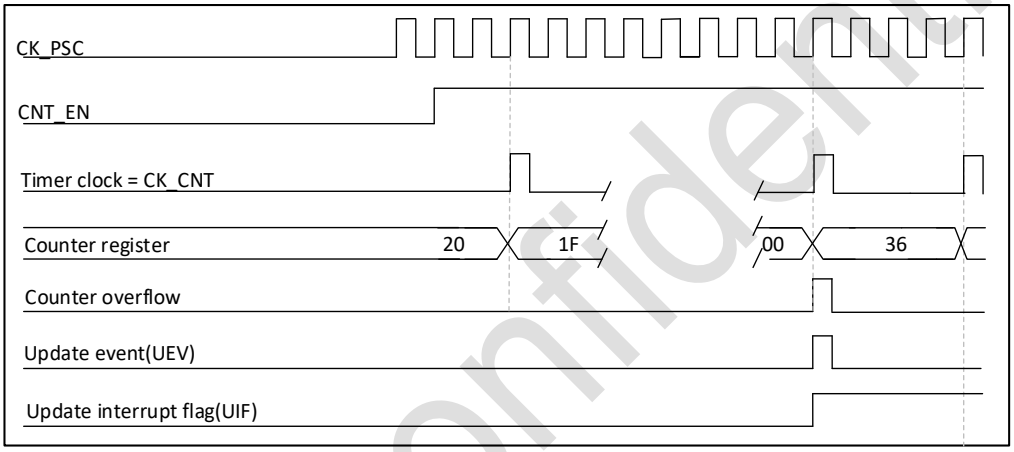


图 17-13 计数器时序图，内部时钟分频因子为 N

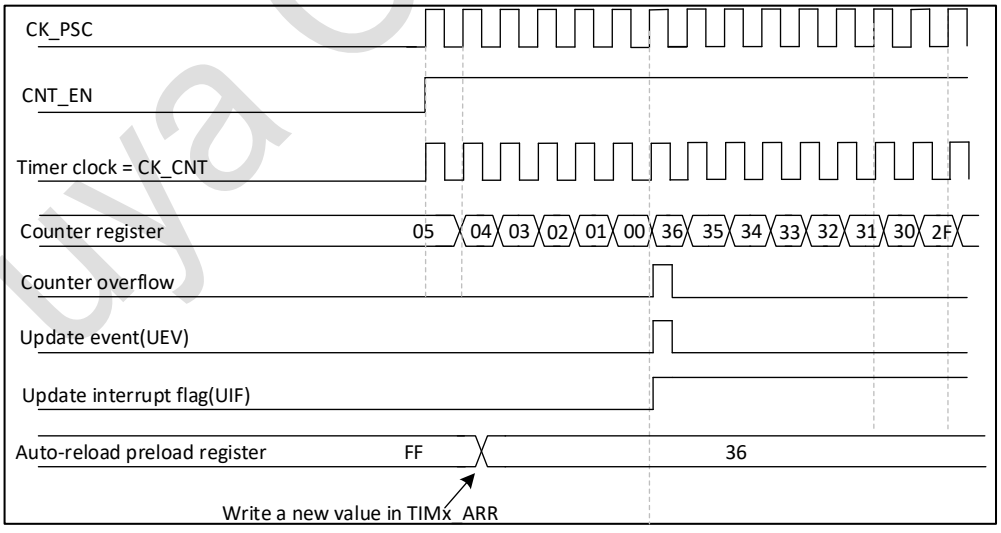


图 17-14 计数器时序图，当没有使用重复计数器时的更新事件

中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)减 1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

通过配置 TIMx_CR1 寄存器中的 CMS 位不为‘00’可以得到中央对齐模式。在通道配置为输出模式下的输出比较标志位在以下几种计数过程中会被置起：向下计数时（中央对齐模式 1，CMS=‘01’）、向上计数时（中央对齐模式 2，CMS=‘10’）、向上和向下计数时（中央对齐模式 3，CMS=‘11’）。

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

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

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

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

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

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)。

以下是一些计数器在不同时钟频率下的操作的例子：

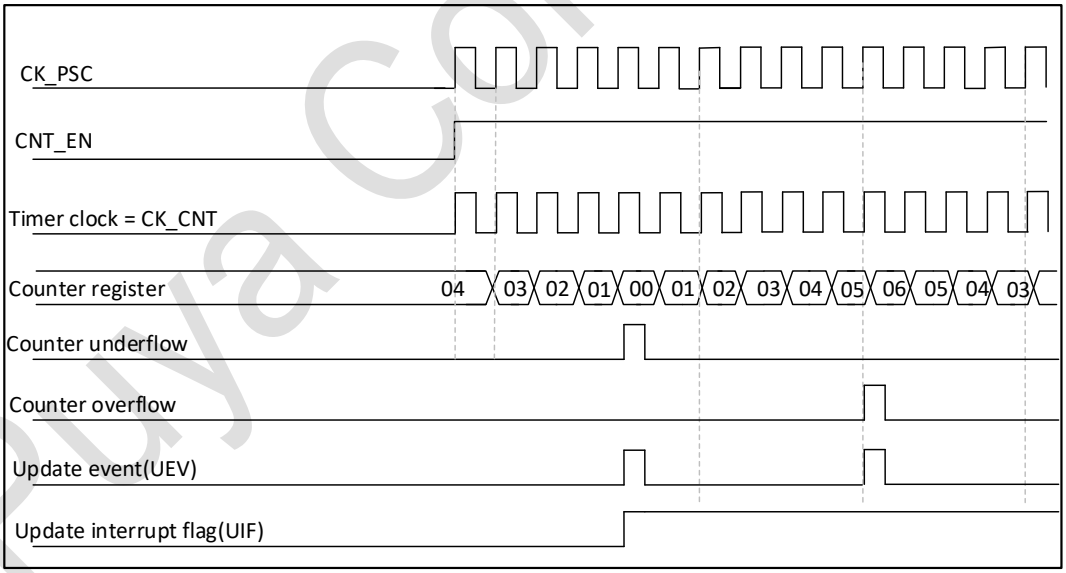


图 17-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

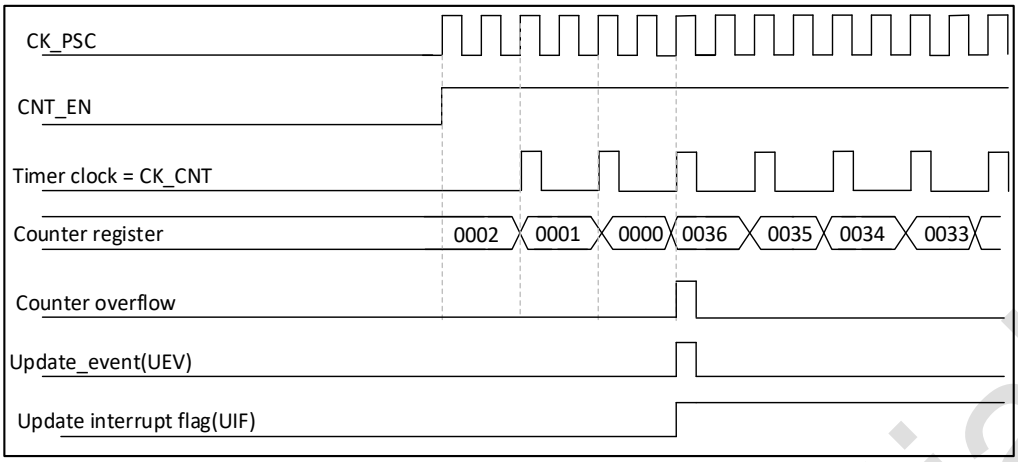


图 17-16 计数器时序图，内部时钟分频因子为 2

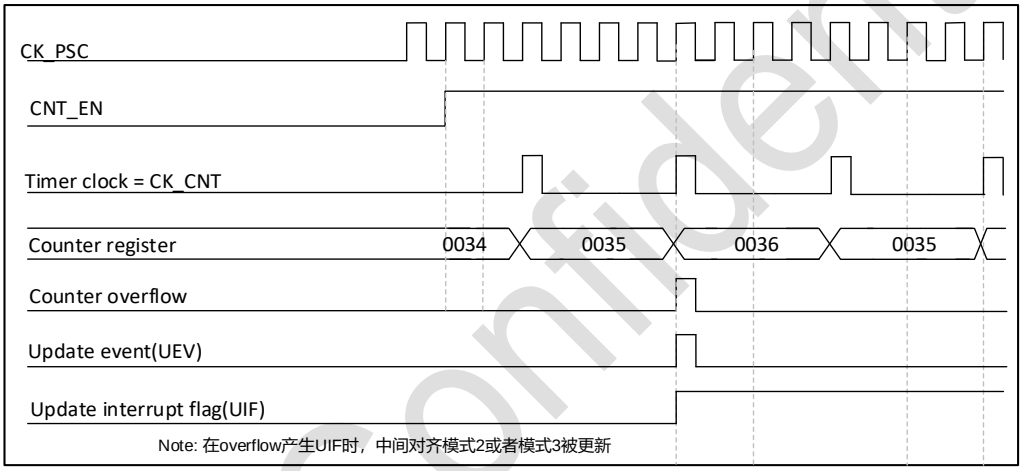


图 17-17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36

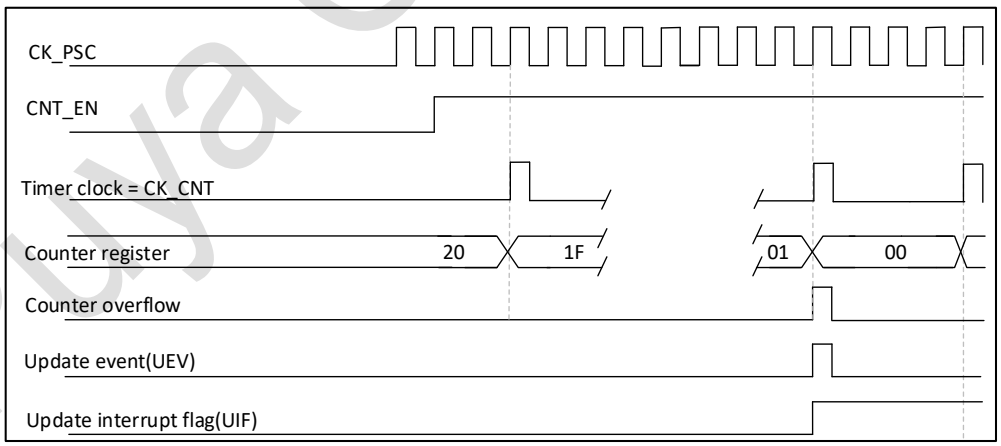


图 17-18 计数器时序图，内部时钟分频因子为 N

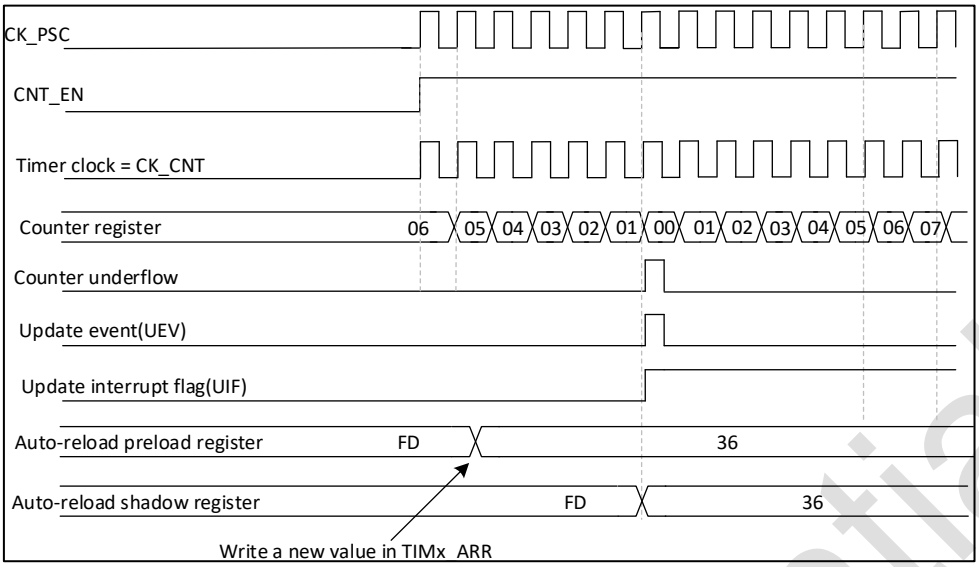


图 17-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

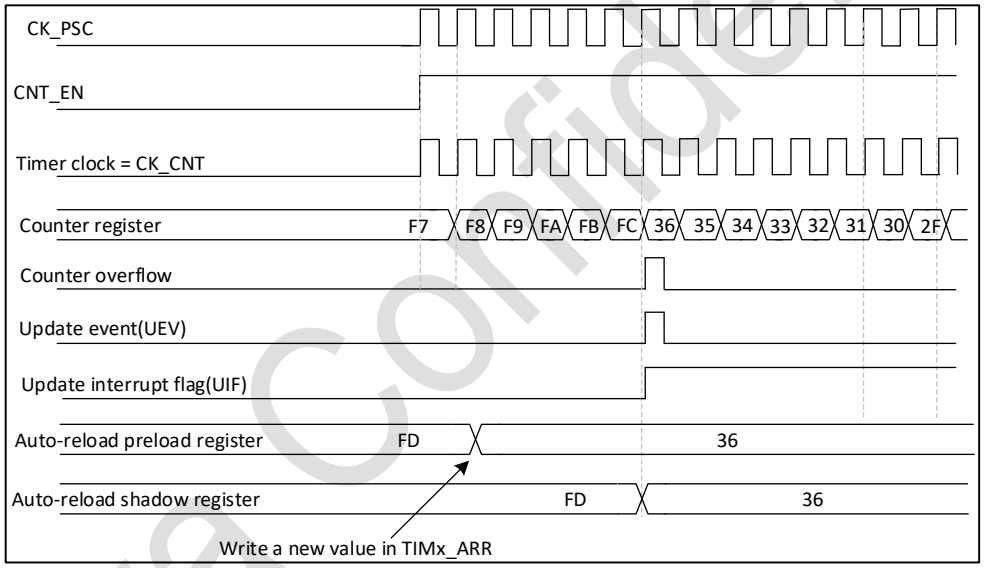


图 17-20 计数器时序图，ARPE=1 时的更新事件(计数器溢出)

17.3.3 重复计数器

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

这意味着在每 N+1 个计数上溢或下溢时，数据才会从预装载寄存器传输到影子寄存器(TIMx_ARR 自动重载寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器上溢时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。

中央对齐模式限制了 PWM 的最大循环周期为 128 (TIMx_RCR 为 8 位计数器)，但它能够在每个 PWM 周期 2 次更新占空比。

重复计数器是自动重载的，重复速率由 TIMx_RCR 寄存器的值定义。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

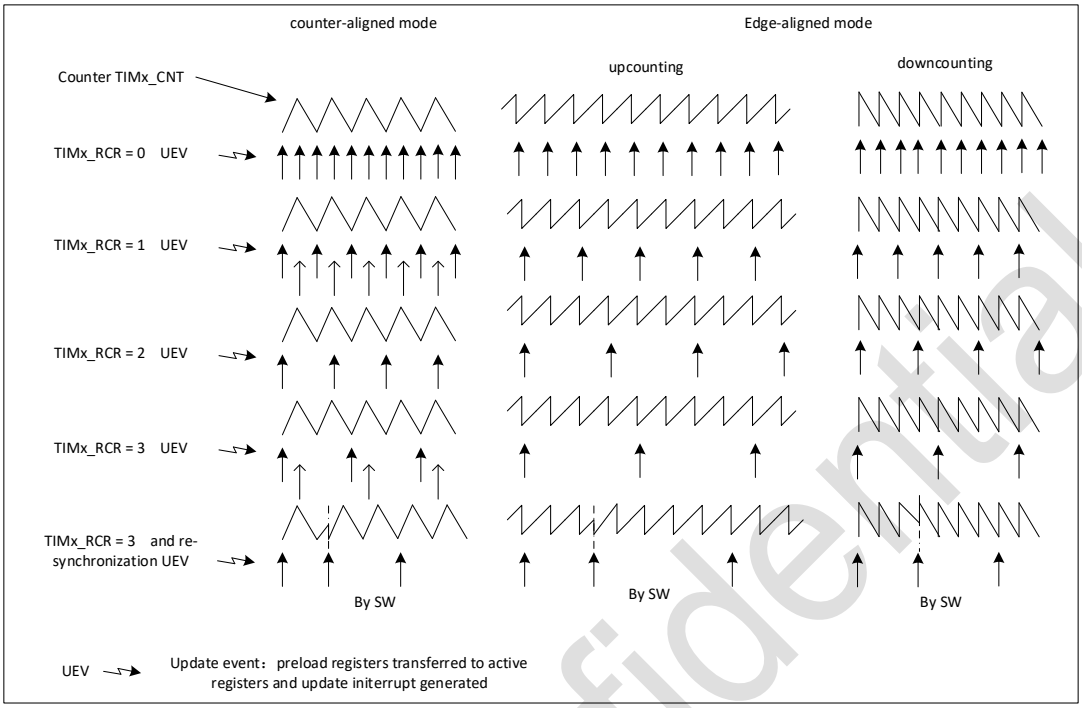


图 17-21 不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置

17.3.4 外部触发输入

定时器外部触发输入 tim_etr，可用于：

- 外部时钟
- 从模式的触发
- 在周期电流调节中作为 PWM 复位输入

17.3.5 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚(TI1 和 TI2)
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器而作为另一个定时器的预分频器。
- 编码器模式

内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=0000)，则 CEN、DIR(TIMx_CR1 寄存器)和 UG 位(TIMx_EGR 寄存器)是事实上的控制位，并且只能被软件修改(UG 位仍被自动清除)。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

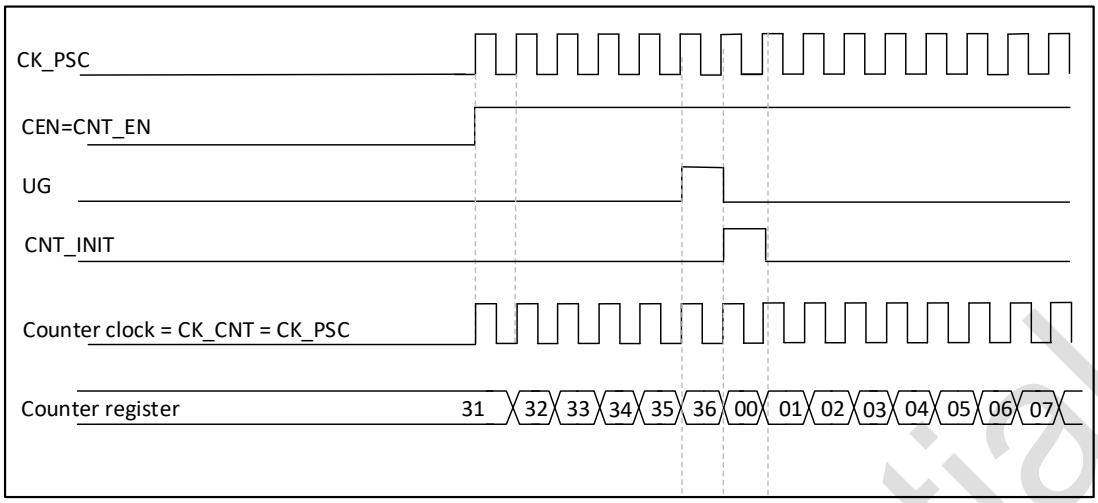


图 17-22 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

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

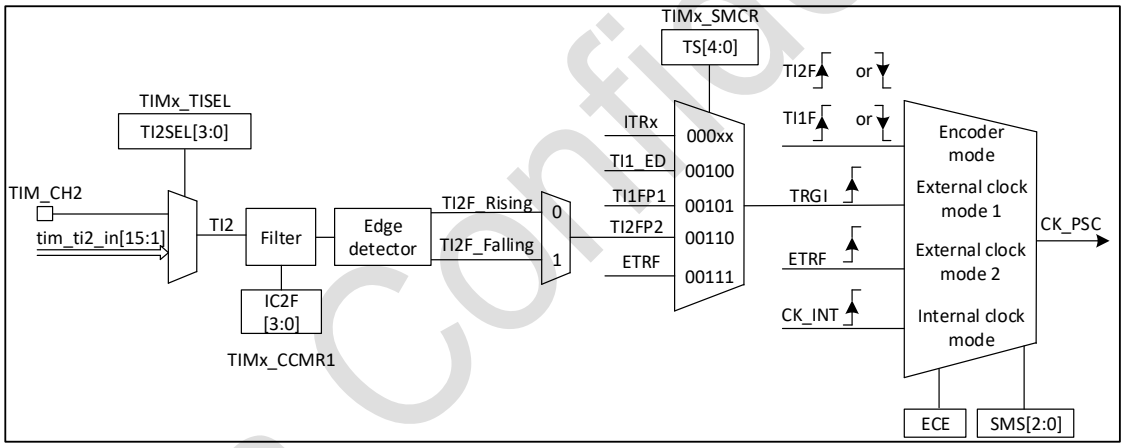


图 17-23 TI2 外部时钟连接例子

例如，要配置计数器在 TI2 输入端的上升沿向上计数，使用下列步骤：

1.配置 TIMx_CCMR1 寄存器 CC2S=01，使得通道 2 检测 TI2 输入端的上升沿；

2.配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽 (如果不需要滤波器，保持 IC2F=0000)；

3.配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；

4.配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器为外部时钟模式 1；

5.配置 TIMx_SMCR 寄存器中的 TS=00110，选定 TI2 作为触发输入源；

6.设置 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

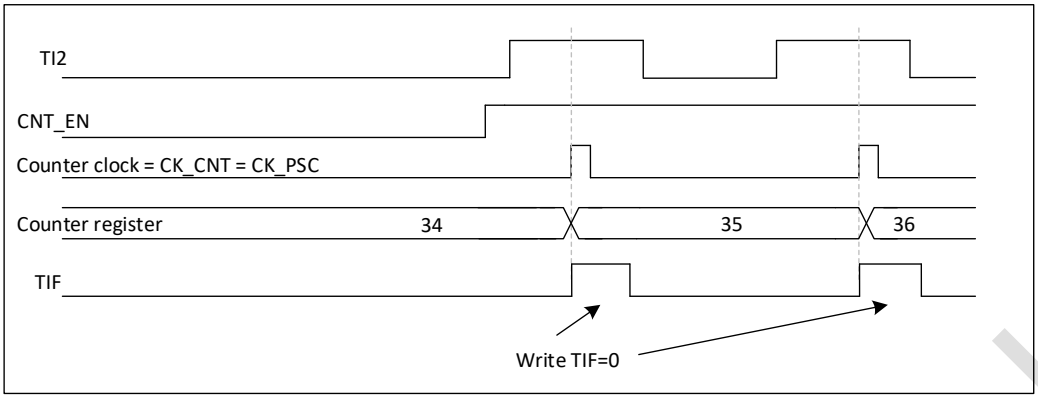


图 17-24 外部时钟模式 1 下的控制电路

外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1，计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图：

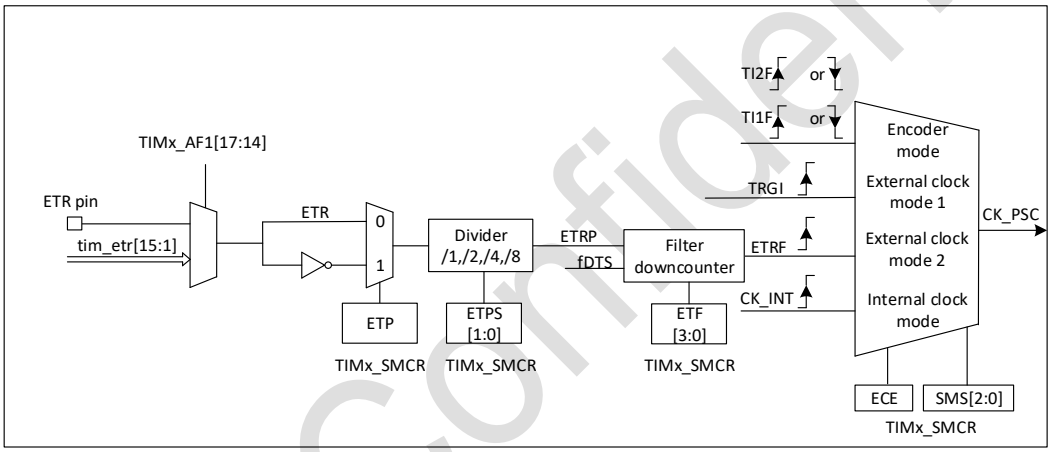


图 17-25 外部触发输入框图

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

1.本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000；

2.设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0]=01；

3.选择 ETR 输入端的上升沿，置 TIMx_SMCR 寄存器中的 ETP=0；

4.开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1；

5.启动计数器，写 TIMx_CR1 寄存器中的 CEN=1；

计数器在每 2 个 ETR 上升沿计数一次。

在 ETR 的上升沿和计数器实际时钟之间的延时取决于 ETRP 信号的同步电路。

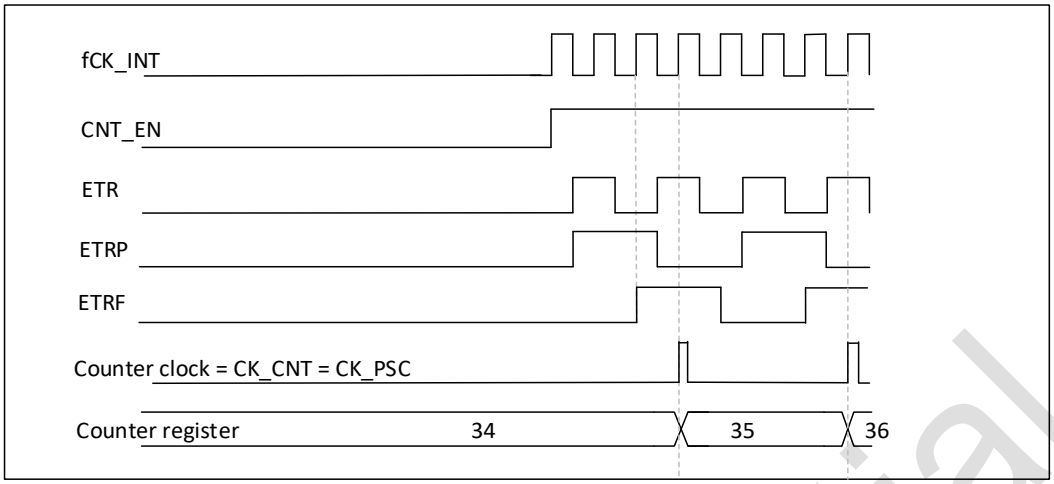


图 17-26 外部时钟模式 2 下的控制电路

17.3.6 捕获/比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器，通道 5 和 6 除外)，和输出部分(比较器和输出控制)。

下面几张图是捕获/比较通道概览。

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

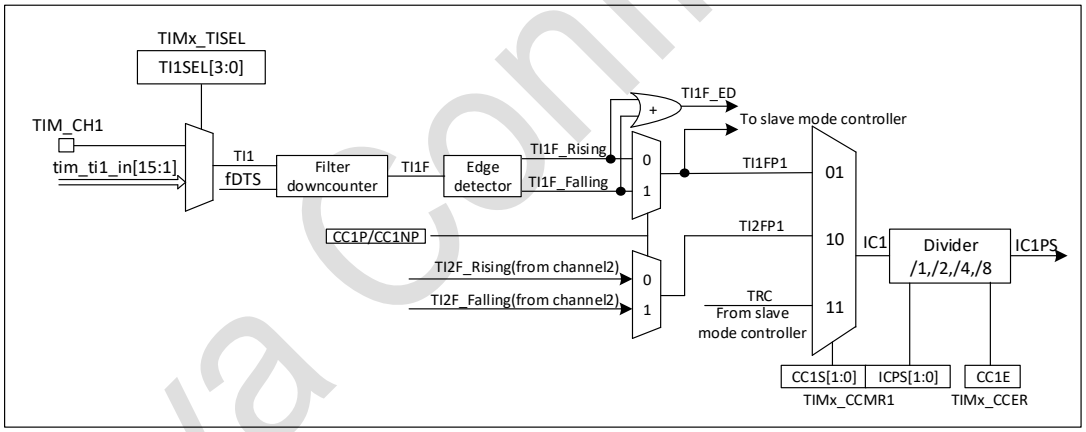


图 17-27 捕获/比较通道(如：通道 1 输入部分)

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

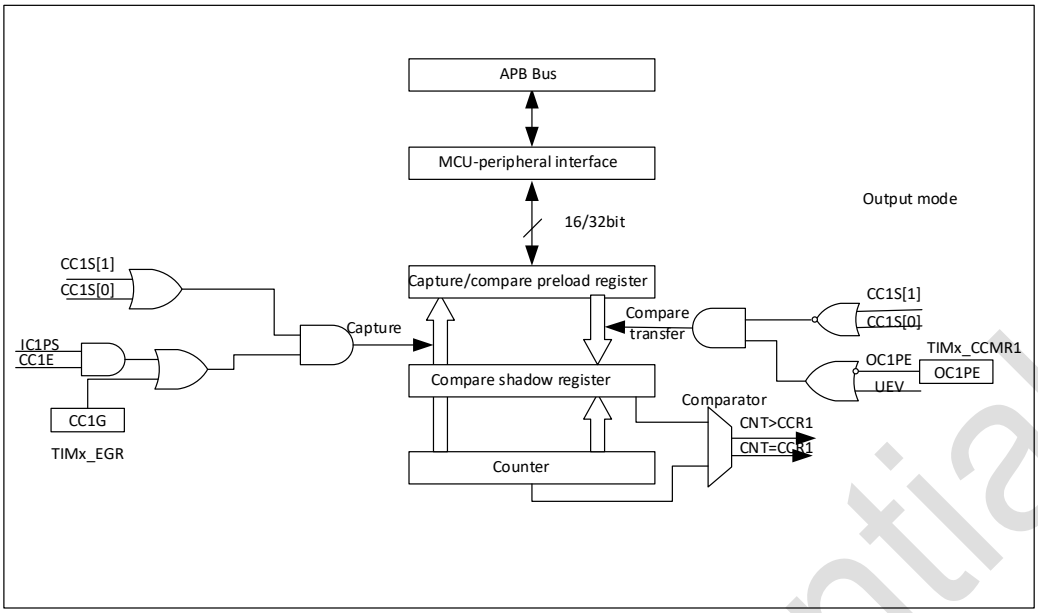


图 17-28 捕获/比较通道 1 的主电路

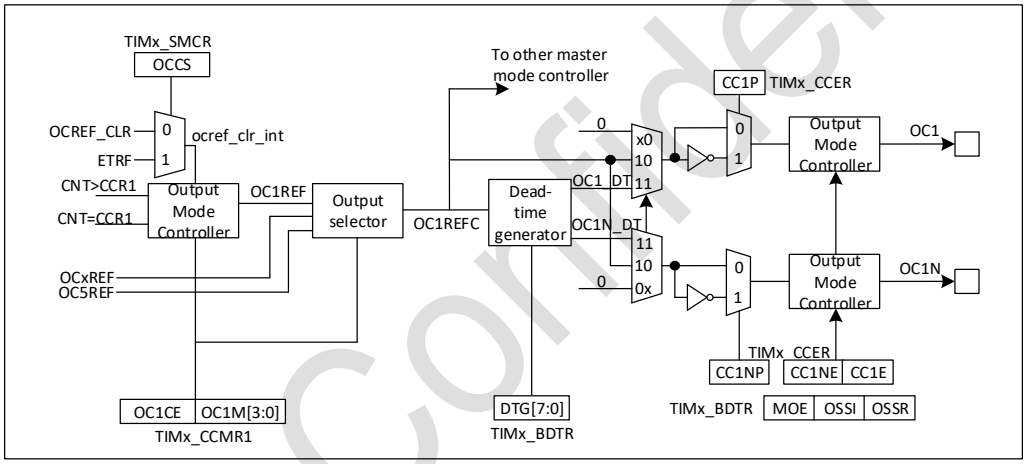


图 17-29 捕获/比较通道的输出部分(通道 1 至 4)

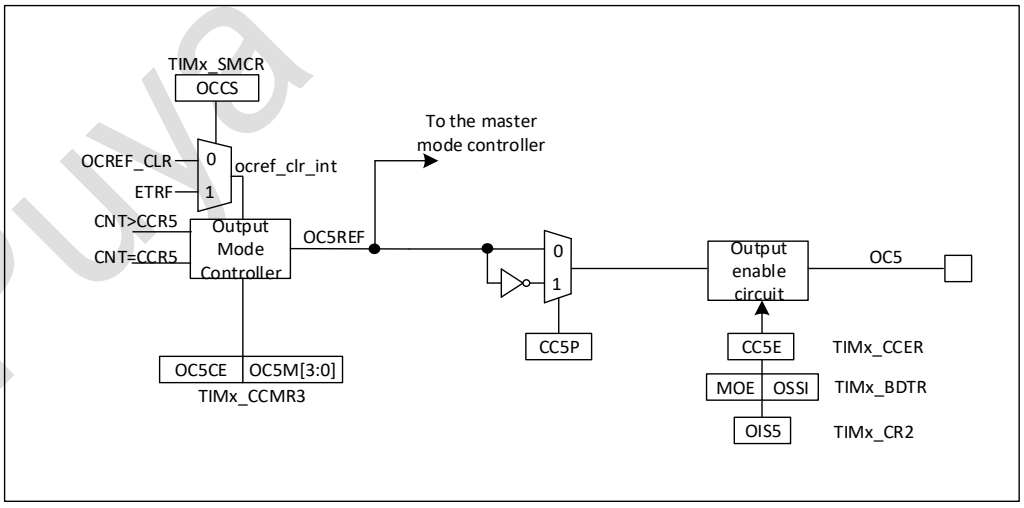


图 17-30 捕获/比较通道的输出部分(通道 5,6)

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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

17.3.7 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果使能中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么过捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。通过写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMx_CCMR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为‘00’，通道就被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(设置为上升沿)。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，可以通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，也可通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

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

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

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

17.3.8 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，其余操作与输入捕获模式相同：

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

例如，用户可以测量输入到 TI1 上的 PWM 信号的周期（TIMx_CCR1 寄存器）和占空比（TIMx_CCR2 寄存器），具体步骤如下

- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性（用来捕获数据到 TIMx_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10（选中 TI1)。
- 选择 TI1FP2 的有效极性（捕获数据到 TIMx_CCR2）：置 CC2P=1（下降沿有效)。

- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

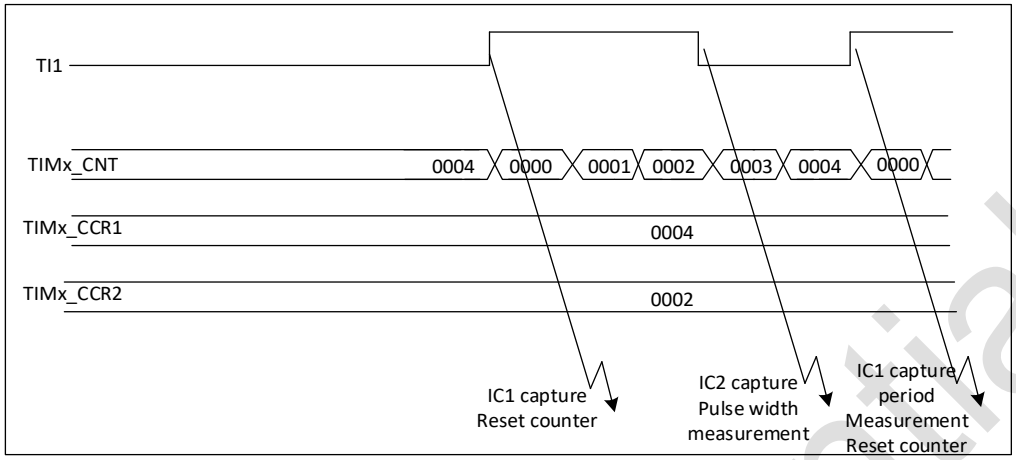


图 17-31 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIMx_CH1/TIMx_CH2 信号。

17.3.9 强置输出模式

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

置 TIMx_CCMRx 寄存器中相应的 OCxM=0101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性相反的信号。

例如： CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

置 TIMx_CCMRx 寄存器中的 OCxM=0100，可强置 OCxREF 信号为低。

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

17.3.10 输出比较模式

此项功能是用来控制一个输出波形，或者表明一段给定的的时间已经到时。通道 1 到 4 可以输出，但通道 5 和 6 仅在 MCU 内部使用（例如用于产生混合波形或用于 ADC 触发）。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=0000)、被设置成有效电平(OCxM=0001)、被设置成无效电平(OCxM=0010)或进行翻转(OCxM=0011)。
- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位， TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

可以通过配置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

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

输出比较模式的配置步骤：

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=0011
   - 置 OCxPE = 0 禁用预装载寄存器
   - 置 CCxP = 0 选择极性为高电平有效
   - 置 CCxE = 1 使能输出
5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器

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

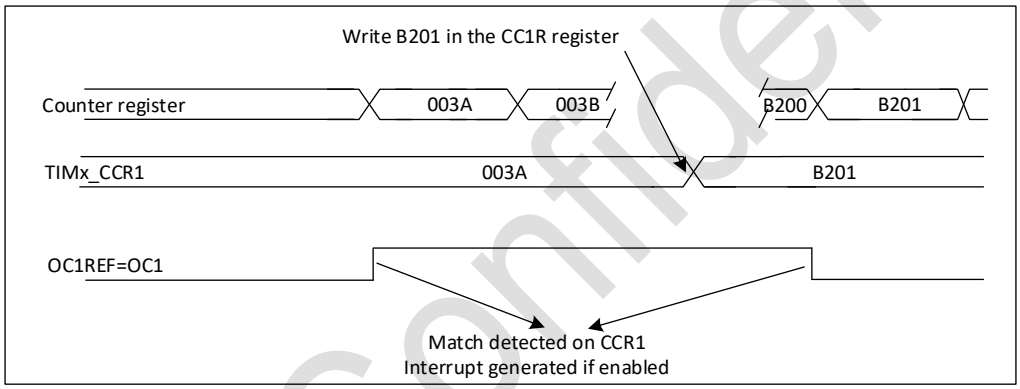


图 17-32 输出比较模式，翻转 OC1

17.3.11 PWM 模式

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

在 TIMx_CCMRx 寄存器中的 OCxM 位写入'0110' (PWM 模式 1)或'0111'(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位， (在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

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

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

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

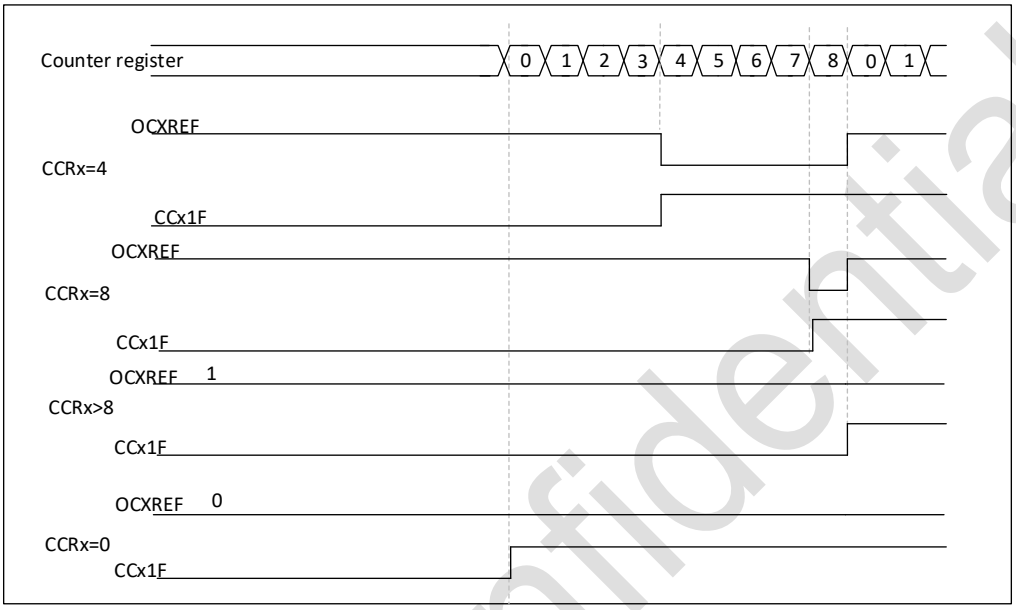


图 17-33 边沿对齐的 PWM 波形(ARR=8)

- 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

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

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为‘00’时为中央对齐模式(CMS 位的所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。 TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

下图给出了一些中央对齐的 PWM 波形的例子

- TIMx_ARR=8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

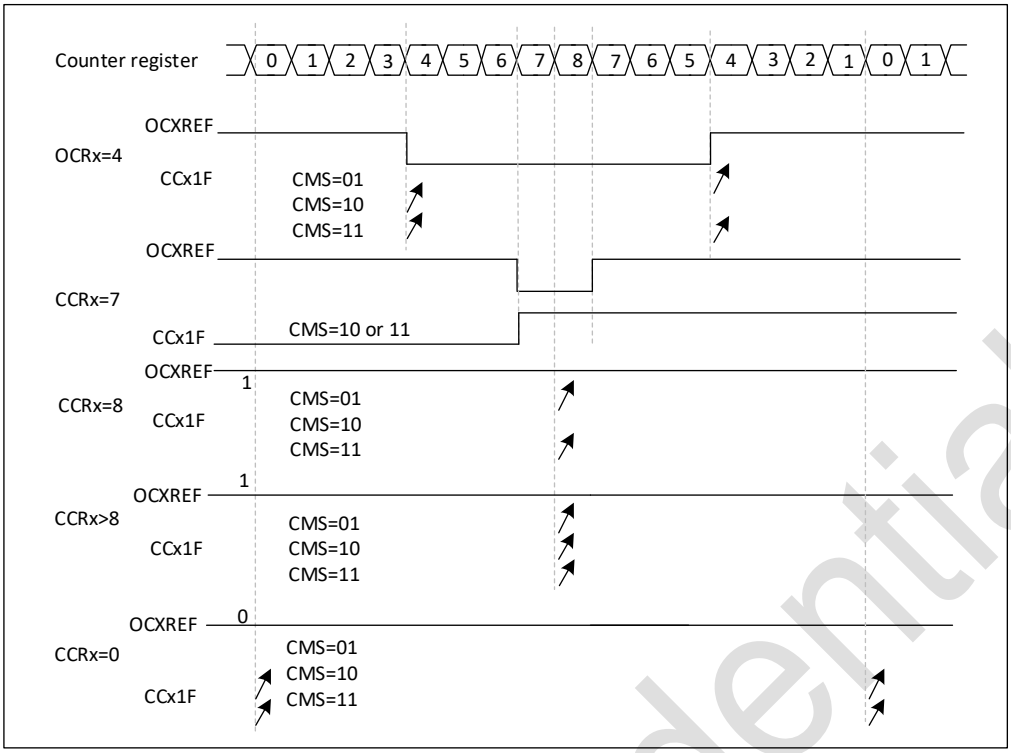


图 17-34 中央对齐的 PWM 波形(APR=8)

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，不要通过软件同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不会产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

PWM 移相模式

PWM 移相模式是指将脉冲信号的相位进行调整。在 PWM 中央对齐模式中，可以通过调整移相功能，将输出信号的相位相对于输入信号进行偏移。这一功能可以用于控制电子设备的输出波形，实现相位差的调整。

在 PWM 中央对齐模式（CMS 不为 00）下使能 PWM 移相模式，通过在各 PWM 周期产生一次中断（根据 TIM1_AF1 中的 INTR_SEL 选择中断位置），更新寄存器输出比较值 CCR1、CCR2、CCR3（其中 CCRx[15:0]为向上计数模式的比较值，CCRx[31:16]为向下计数模式的比较值）产生 PWM 移相功能。PWM 移相结构图如下图。

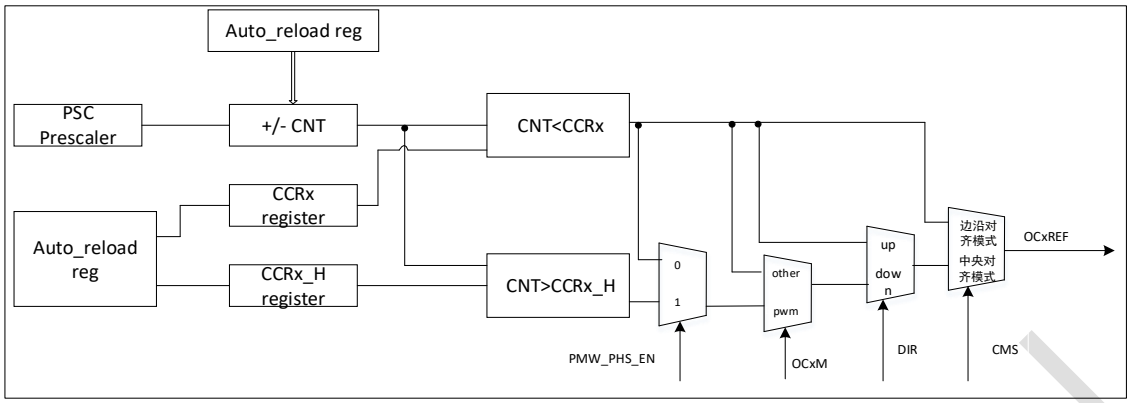


图 17-35 PWM 移相模式结构概览

下图为 PWM2 输出模式，使能预装载功能(OCxPE=1)，tim1_intr 在下溢时产生中断（上溢/下溢中断可用 INTR_SEL 选择），将预先设置的 CCRx/CCRx_H 的预装载值在更新事件到来时更新到寄存器 CCRx_SHA/CCRx_H_SHA 中。通过每次更新中断后设置 CCRx_SHA/CCRx_H_SHA 寄存器的值来实现 PWM 移相的功能。

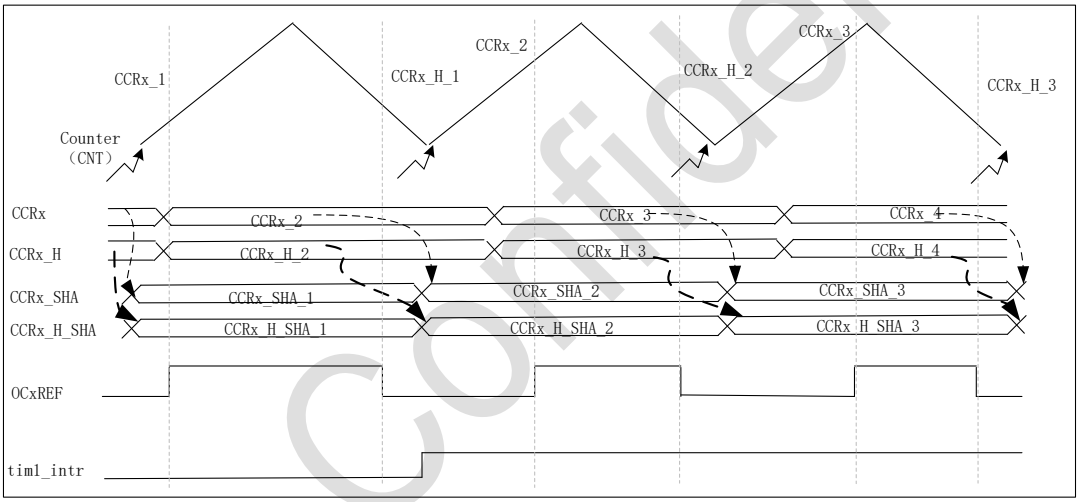


图 17-36 PWM2 输出模式，使能预装载功能(OCxPE=1)

注：PWM 移相模式更新 CCRx_SHA/CCRx_H_SHA 寄存器，有两种方法可以选择：

预装载功能使能时(OCxPE=1)，只有在更新事件到来时才会将预先设置的预装载值更新到 CCRx_SHA /CCRx_H_SHA 寄存器中。

预装载功能无效（OCxPE=0），CCRx_SHA /CCRx_H_SHA 的值可以实时进行更新（使用此方式时，向上计数模式时不能更改 CCRx_SHA（向下计数模式时不能更改 CCRx_H_SHA），否则，会不满足 PWM 移相功能。如下图为 PWM2 输出模式下，预装载功能无效(OCxPE=0)，tim1_intr 在下溢时产生中断，向上计数时实时更新寄存器 CCRx，此种情况不满足 PWM 移相模式。

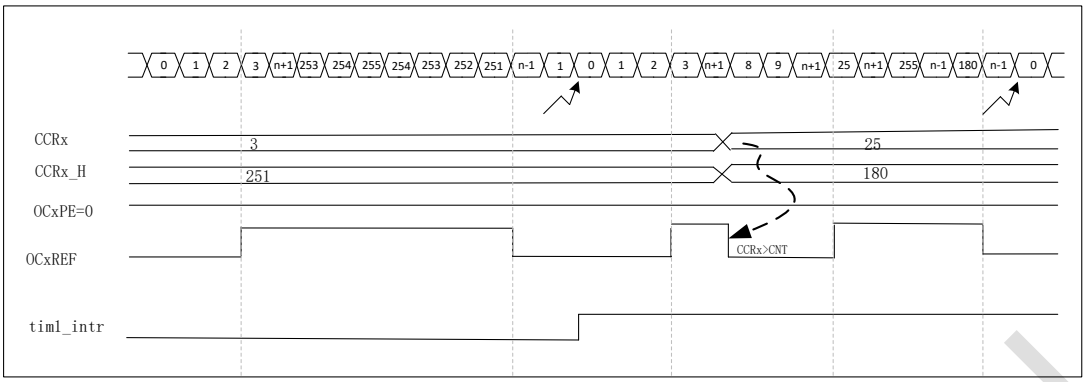


图 17-37 PWM2 输出模式下，预装载功能无效(OCxPE=0)

PWM 移相配置步骤：

假定计数器选择内部时钟源:

- 选择中央对齐模式：根据 TIMx_CR1 寄存器中的 CMS 选择中央对齐模式
- 选择预装载使能：置 TIMx_CCMRx 寄存器中的 OCxPE=1 使能输出比较 1、2、3 预装载使能。
- 配置比较寄存器 TIMx_CCRx[31:0]，置 UG 位产生一个更新事件，更新比较寄存器值。
- 置 TIMx_AF1 寄存器中的 PWM_PHS_EN=1，PWM 移相使能，此例中选择 ISR_SEL=10 下溢中断。
- 配置 TIMx_ARR 自动重装载寄存器。
- 选择 PWM2 输出模式：置 TIMx_CCMRx 中的 OCxM=4’b0111。
- 置 TIMx_CR1 寄存器中的 CEN=1 使能计数器，如果设置了 UIE 位，则会产生更新中断。
- 预先配置 TIMx_CCRx，出现更新事件后，预装载值更新到本地寄存器中。

17.3.12 互补输出和死区插入

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

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

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

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。通过配置 TIMx_BDTR 寄存器中的 DTG[7:0]位，可以控制所有通道的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

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

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

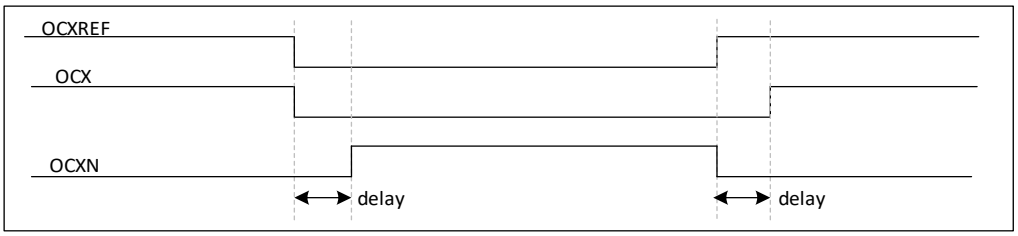


图 17-38 带死区插入的互补输出

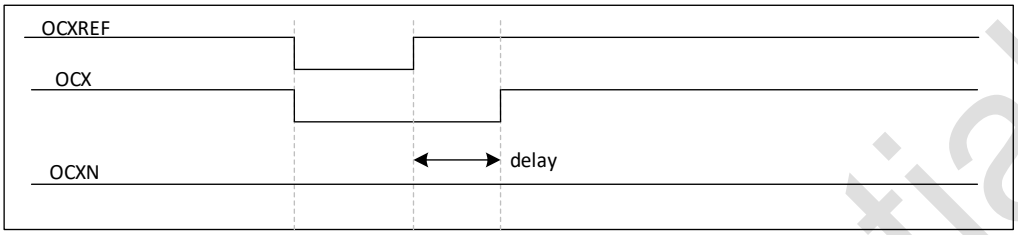


图 17-39 死区波形延迟大于负脉冲

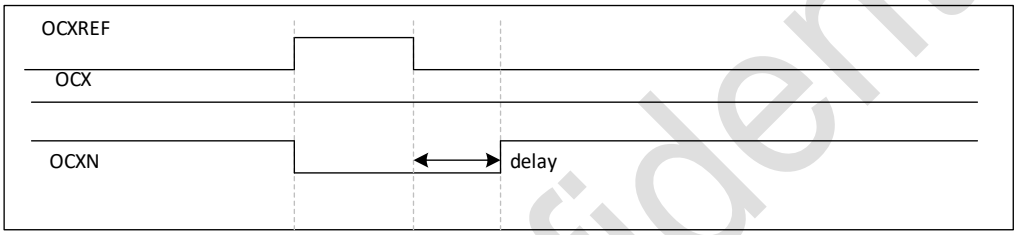


图 17-40 死区波形延迟大于正脉冲

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置、输出比较或 PWM),通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位,OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注：当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0,则 OCxN=OCxREF。另一方面,当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1),当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

17.3.13 使用刹车功能

刹车功能的目的是保护由 TIMER 输出的 PWM 驱动的电源开关。两路刹车的输入通常是连接到三相逆变器和功率级芯片的故障输出。当刹车信号有效时，会立即关断 PWM 的所有输出，并将它们强制到预先设置好的一个安全的电平状态。部分 MCU 内部的事件也可以作为触发信号来关闭输出。

刹车事件有两路通道，通道 1 包括了系统级的故障(clock failure、奇偶校验等)和应用层的故障（通过输入引脚或者内置的比较器），并且可以在一个死区区间后强制输出到预先设置好的电平（无论工作还是空闲）。

在刹车过程中输出使能信号和输出电平由以下多个 bit 控制

- TIMx_BDTR 的 MOE 位可以通过软件或 2 路刹车事件复位来实现使能和关闭输出
- TIMx_BDTR 的 OSSI 位用于定义当处于空闲状态时 timer 是否还控制输出，或者释放对 GPIO 控制器的控制（高阻状态）

TIMx_CR2 的 OISx 和 OISxN 位用于无论工作还是空闲状态下都将输出设置为无效电平,OCx 和 OCxN 输出在同一时刻不能同时设置为有效电平，无论 OISx 和 OISxN 为何值。

当从复位中退出时，刹车功能被关闭并且 MOE 为 0。刹车功能的开启可通过使能 TIMxBDTR 的 BKE 位。刹车输入的极性可通过 BKP 配置。BKEx 和 BKPx 可同时修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

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

刹车通道 1 的源；

- 外部源可连接到其中一个 TIMx_BKIN 引脚上（选择 GPIO 和配置寄存器），再加上极性选择和滤波
- 内部源包含 2 部分
  - 来自刹车比较器信号（tim_brk_cmpx）
  - 来自系统刹车请求

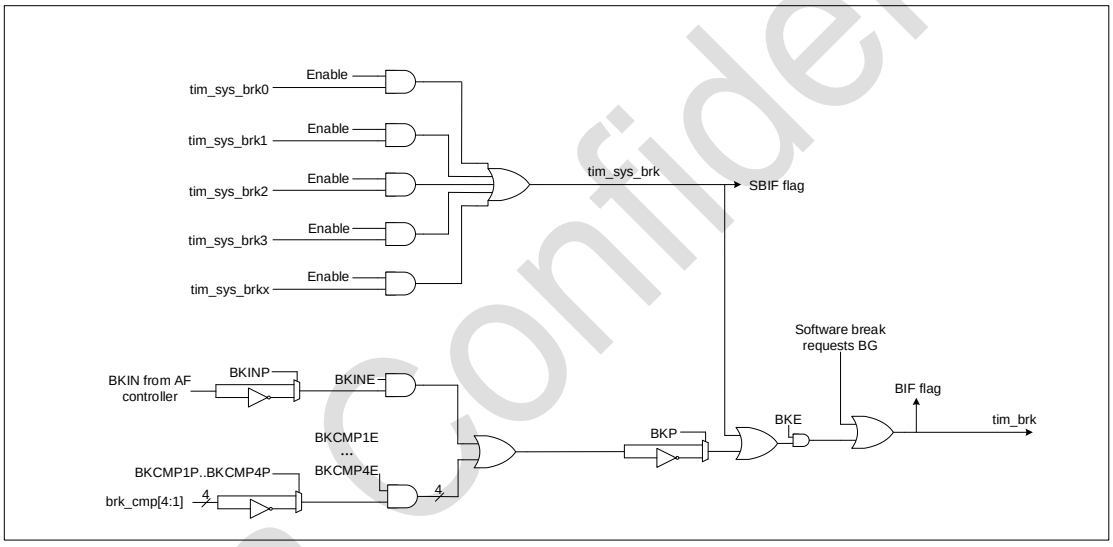


图 17-41 刹车电路

注意：只有没有滤波的时候才能保证是异步的操作，如果滤波开启了，必须使用一个故障保护时钟模式来保证刹车中断事件得到处理（例如使用内部 PLL 或 CSS，无滤波）。

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者释放对 GPIO 控制器的控制(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放输出控制，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下， OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个时钟周期)。
  - 如果 OSSI=0，定时器释放输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，

使能输出变为高。

- 刹车状态标志会被拉高（TIMx_SR 的 SBIF,BIF）。如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为‘1’时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 TDE 位，则产生一个 DMA 请求。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置‘1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注：当 AOE 为 1 时如果 MOE 被 CPU 复位了，输出会进入空闲状态，并被强制到无效电平或者高阻（取决于 OSSI 的值），如果 MOE 和 AOE 均被 CPU 复位，输出将进入无效状态并且由 OISx 位来驱动输出电平。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区持续时间， OCx/OCxN 极性和被禁止的状态， OCxM 配置，刹车使能和极性)。用户可以通过设置 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

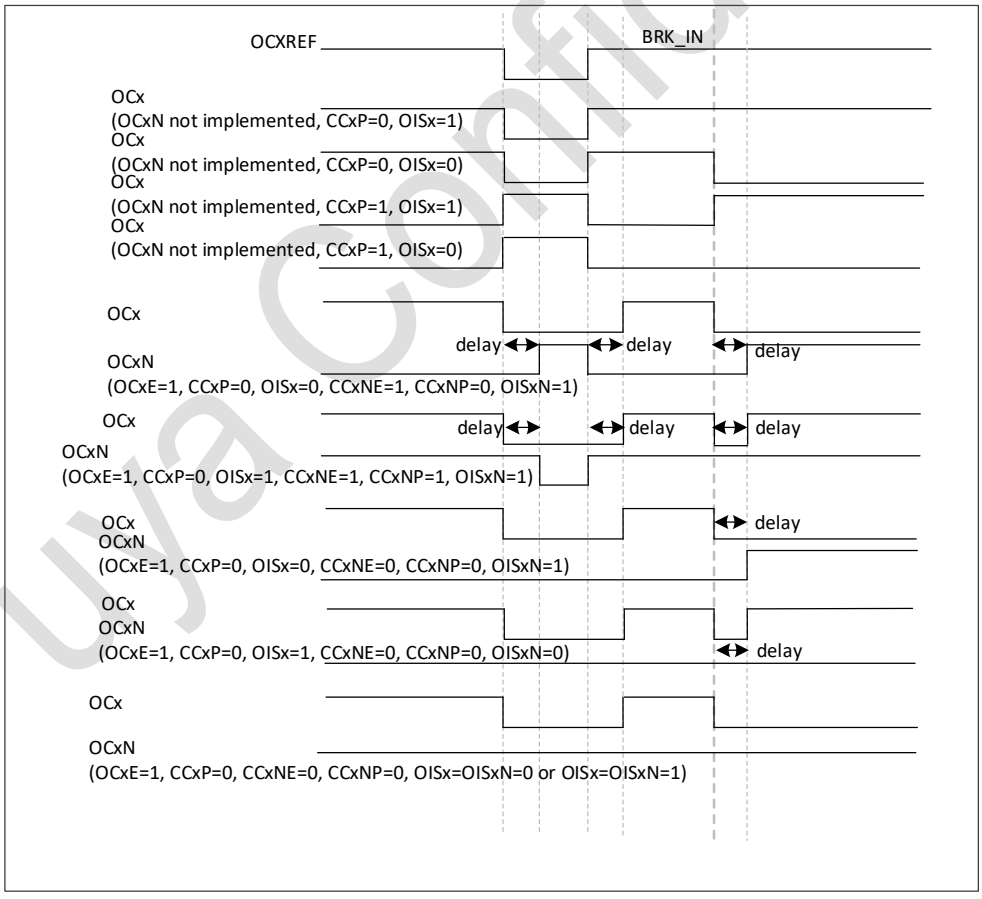


图 17-42 响应刹车的输出

17.3.14 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为‘1’，能够用 OCREF_CLR_INT 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次计数溢出所产生的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

而 OCREF_CLR_INT 可以通过配置 TIMx_SMCR 寄存器中的 OCCS 位，在 OCREF_CLR 和 ETRF(ETR 滤波后)之间选择。

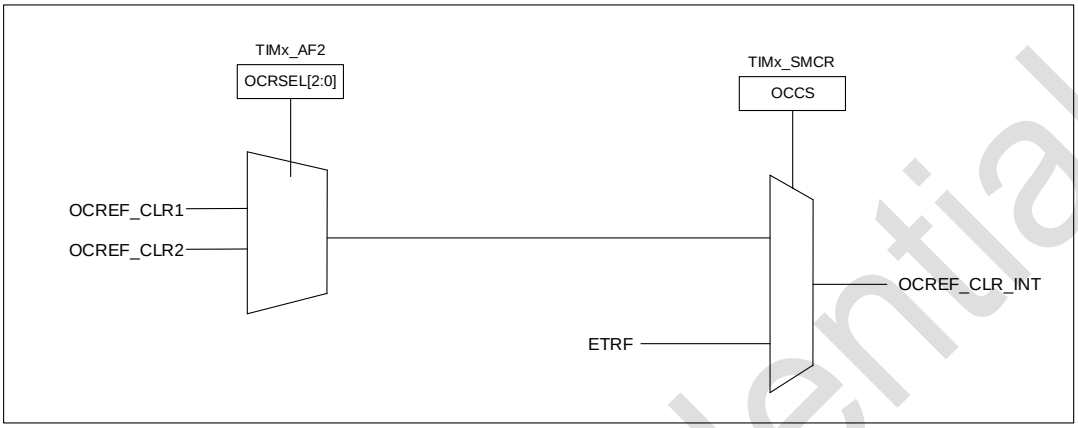


图 17-43 OCREF_CLR 输入选择

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

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

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值， OCxREF 信号的动作。在这个例子中，定时器 TIMx 被置于 PWM1 模式。

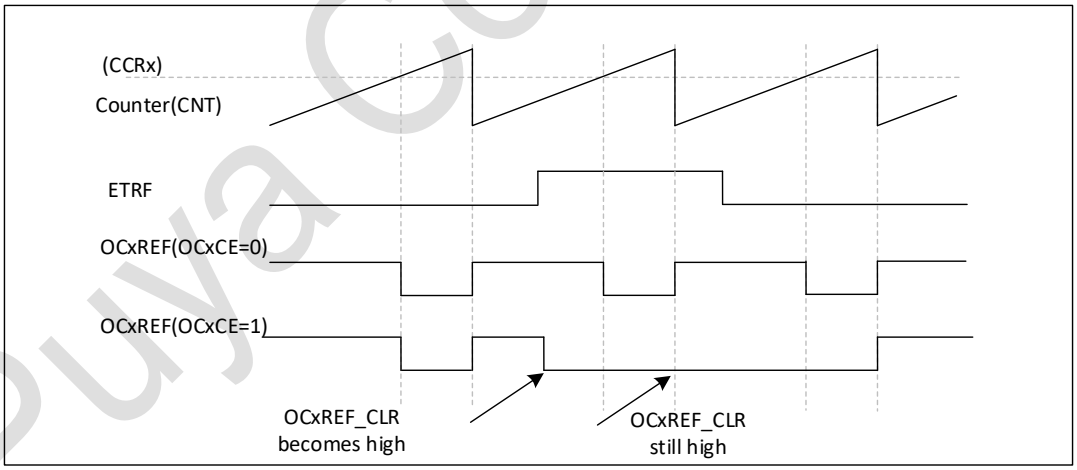


图 17-44 清除 TIMx 的 OCxREF

17.3.15 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时更改所有通道的配置。COM 可以通过设置 TIMx_EGR 寄存器的 COMG 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位(TIMx_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

下图显示当发生 COM 事件时，三种不同配置下 OCx 和 OCxN 输出。

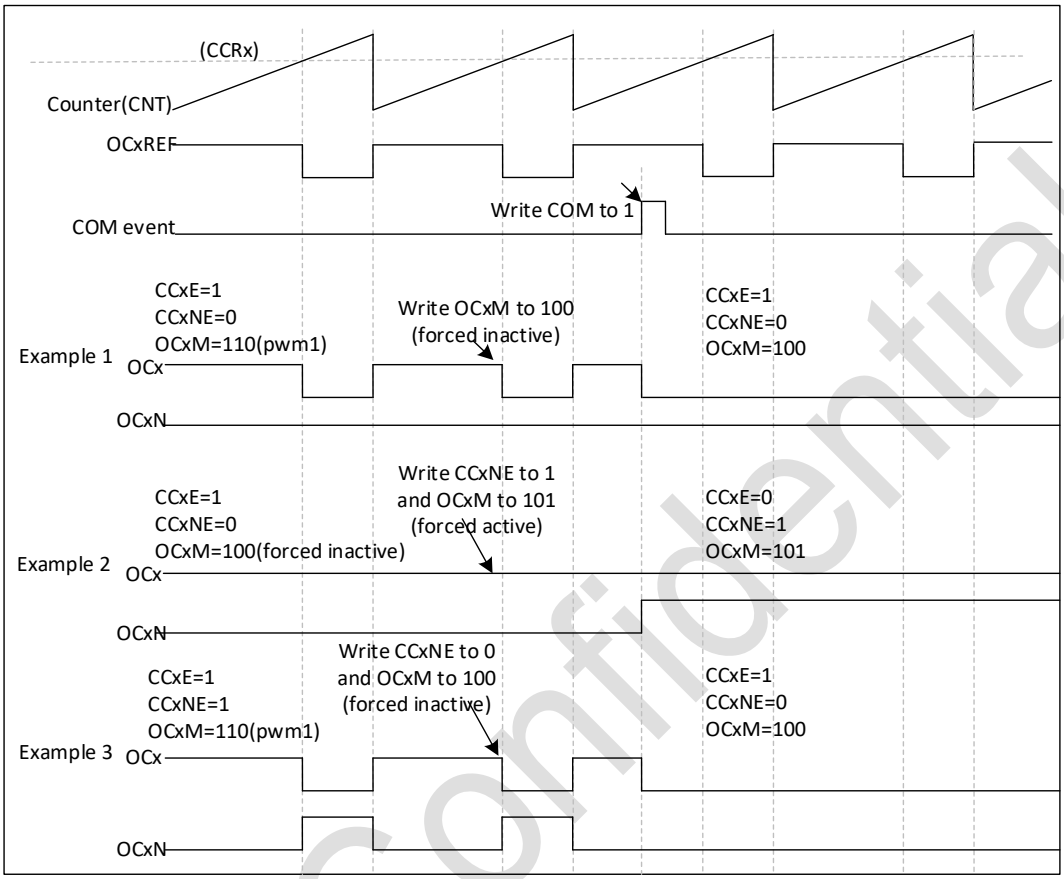


图 17-45 产生六步 PWM，使用 COM 的例子(OSSR=1)

17.3.16 单脉冲模式

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

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地，0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

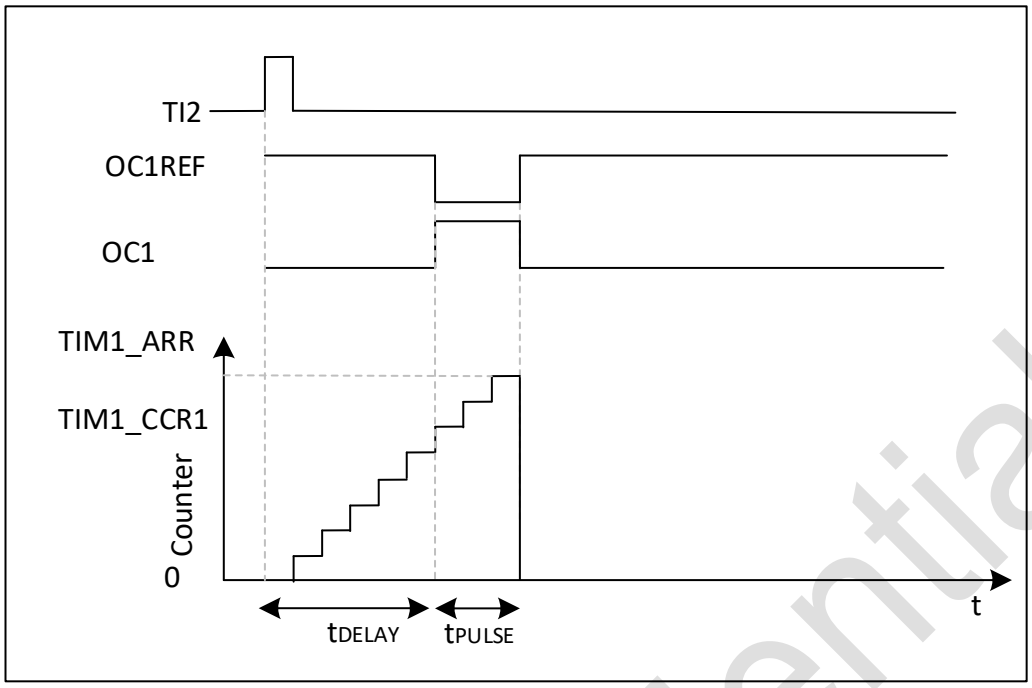


图 17-46 单脉冲模式的例子

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

假定 TI2FP2 作为触发:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110， TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)， TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中， CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 OPM=0 时，重复模式被选中。

特殊情况：OCx 快速使能：

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

17.3.17 可重触发的单脉冲模式

该模式允许计数器响应激励启动并产生长度可编程的脉冲，与上节所述的不可再触发单脉冲模式有以下区别：

- 触发一发生，脉冲就开始（无可编程延迟）。
- 如果在前一个触发所产生的脉冲完成之前发生新的触发，则延长脉冲。

要使用可重触发的单脉冲模式，计时器必须处于从模式，TIMx_SMCR 寄存器中的位 SMS[3:0]=“1000”（组合模式-复位+触发），OCxM[3:0]位设置为“1000”或“1001”（可重新触发 OPM 模式 1 或 2）。

如果此时计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）。

如果计时器配置为递减计数模式，CCRx 必须大于或等于 ARR。

注：出于兼容性原因，OCxM[3:0]和 SMS[3:0]位字段被分成两部分，最高有效位与 3 个最低有效位位置不连续。

该模式不得与中央对齐计数模式一起使用。TIMx_CR1 中必须为 CMS[1:0]=00。

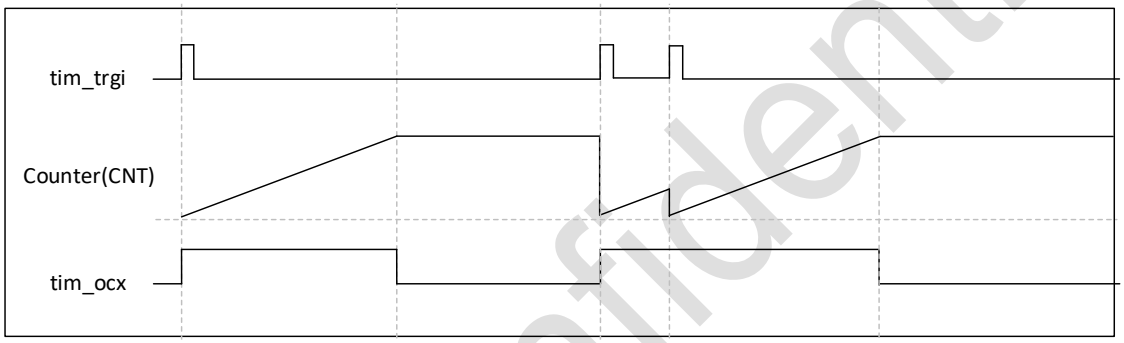


图 17-47 可重触发的单脉冲模式的例子

17.3.18 编码器接口模式

正交编码器

选择编码器接口模式的方法是：如果计数器只在 TI1 的边沿计数，则置 TIMx_SMCR 寄存器中的 SMS=0001；如果只在 TI2 边沿计数，则置 SMS=0010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=0011。

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

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看下表，假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 17-1 计数方向与编码器信号的关系（CC1P=CC2P=0）

有效沿	SMS[3:0]	反向信号电平 (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 信号		TI2FP2 信号	
			Rising	Falling	Rising	Falling
仅在 TI1 双沿计数 x2 模式	0001	高	减	加	-	-
		低	加	减	-	-
仅在 TI2 双沿计数 x2 模式	0010	高	-	-	加	减
		低	-	-	减	加
在 TI1 和 TI2 双沿计数 x4 模式	0011	高	减	加	加	减
		低	加	减	减	加

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

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

- CC1S='01'(TIMx_CCMR1 寄存器，TI1FP1 映射到 IC1)
- CC2S='01'(TIMx_CCMR2 寄存器，TI2FP2 映射到 IC2)
- CC1P='0' (TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1)
- CC2P='0' (TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2=TI2)
- SMS='011'(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效).
- CEN='1'(TIMx_CR1 寄存器，计数器使能)

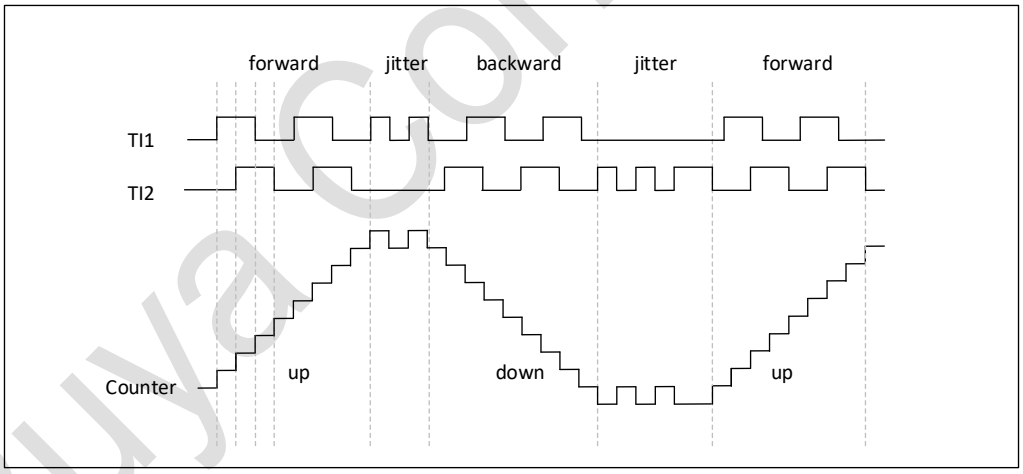


图 17-48 编码器模式下的计数器操作实例

下图为当 TI1FP1 极性反相时计数器的操作实例(CC1P='1'，其他配置与上例相同)

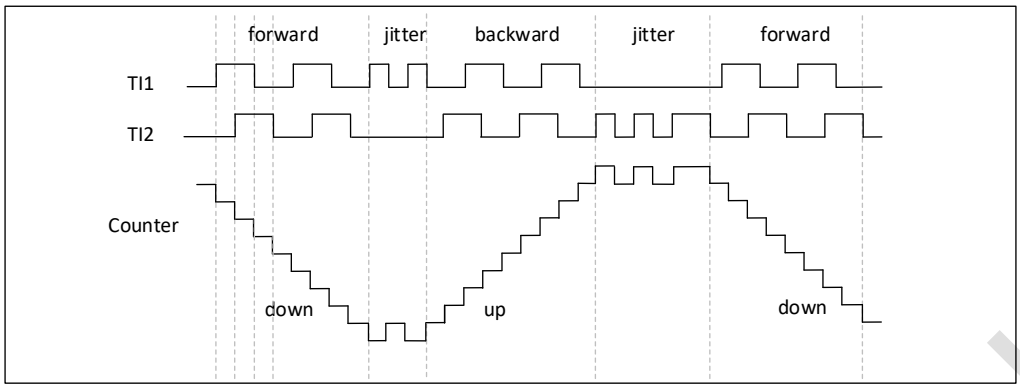


图 17-49 TI1FP1 反相的编码器接口模式实例

下图表示的是不同模式下当转向翻转时的计数值情况：

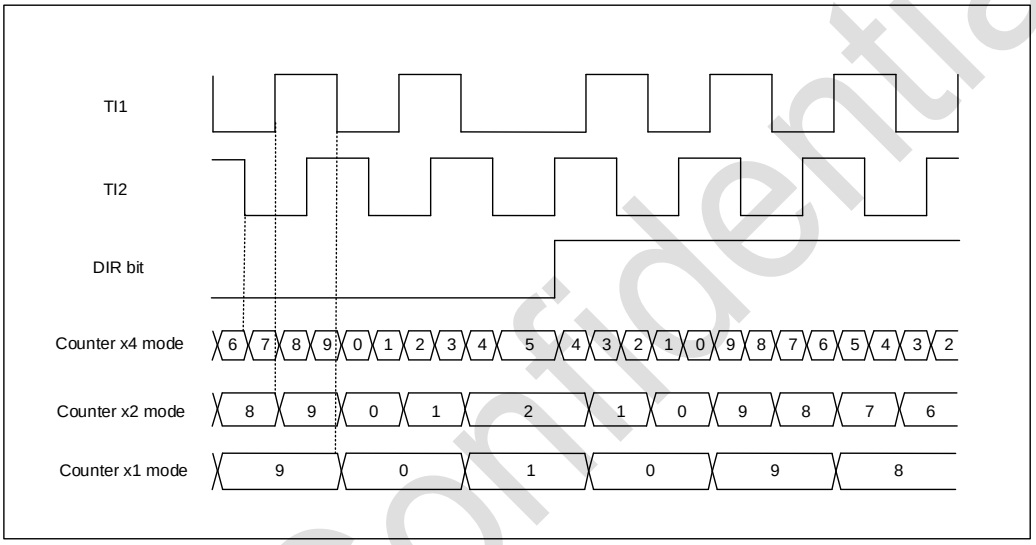


图 17-50 正交编码器计数模式

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

17.3.19 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下节给出了此特性用于连接霍尔传感器的例子。

17.3.20 与霍尔传感器的接口

使用高级控制定时器(TIM1)产生 PWM 信号驱动马达时，可以用另一个通用 TIMx(TIM2、TIM3、 TIM4)定时器作为“接口定时器”来连接霍尔传感器，见下图，3 个定时器输入引脚(TI1、TI2、TI3)通过一个异或门连接到 TI1 输入通道(通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，此时“接口定时器”被用来捕获这个信号。

从模式控制器被配置成复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器重新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC(如下图)。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以(通过触发一个 COM 事件)用于改变高级定时器 TIM1 各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。

因此“接口定时器”通道必须编程为在一个指定的延时(输出比较或 PWM 模式)之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器 TIM1。

举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。

- 置 TIMx_CR2 寄存器的 TI1S 位为‘1’，配置三个定时器输入逻辑异或到 TI1 输入；
- 时基编程：置 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔；
- 设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器；
- 设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00；
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的(TIMx_CR2 寄存器中 CCPC=1)，同时触发输入控制 COM 事件(TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入下一步的 PWM 控制位(CCxE、OCxM)，这可以在处理 OC2REF 上升沿的中断子程序里实现。

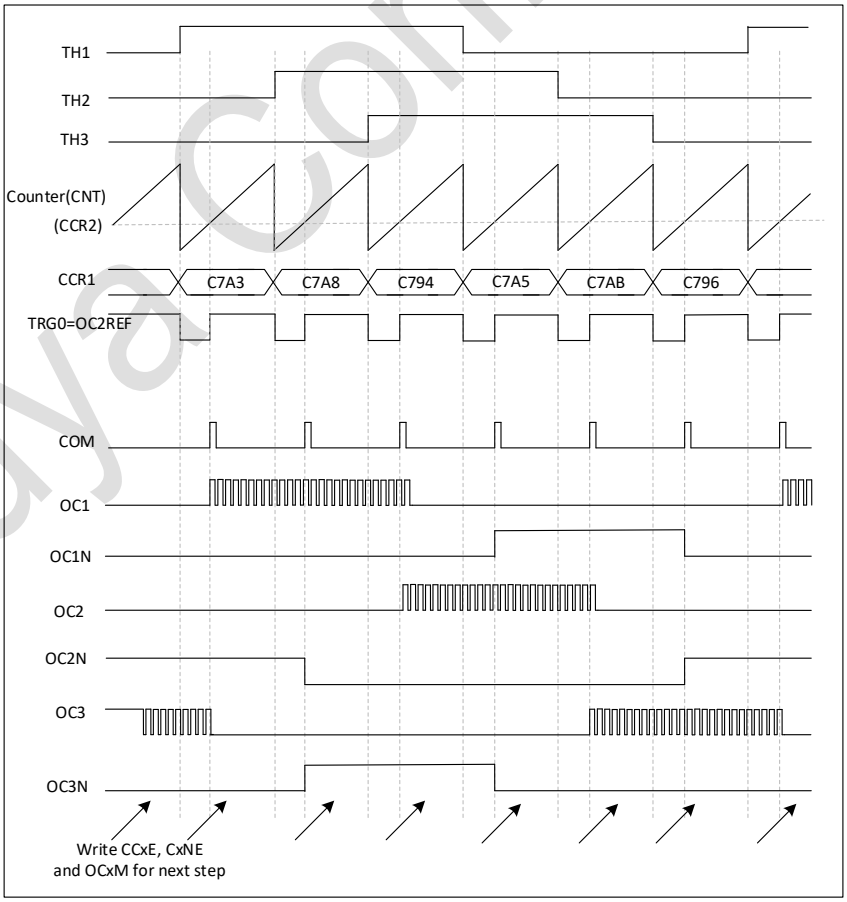


图 17-51 霍尔传感器接口的实例

17.3.21 定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式、触发、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 UDIS 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，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=0100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

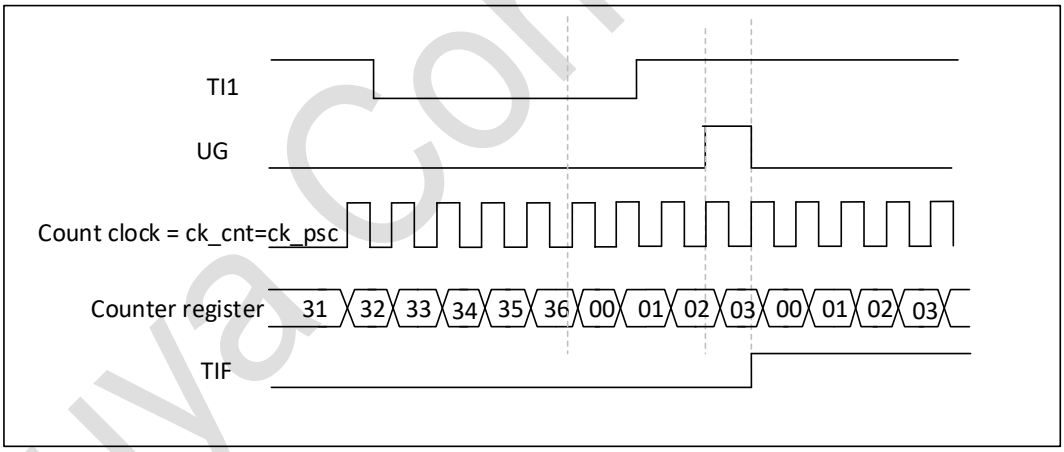


图 17-52 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，

不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 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。

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

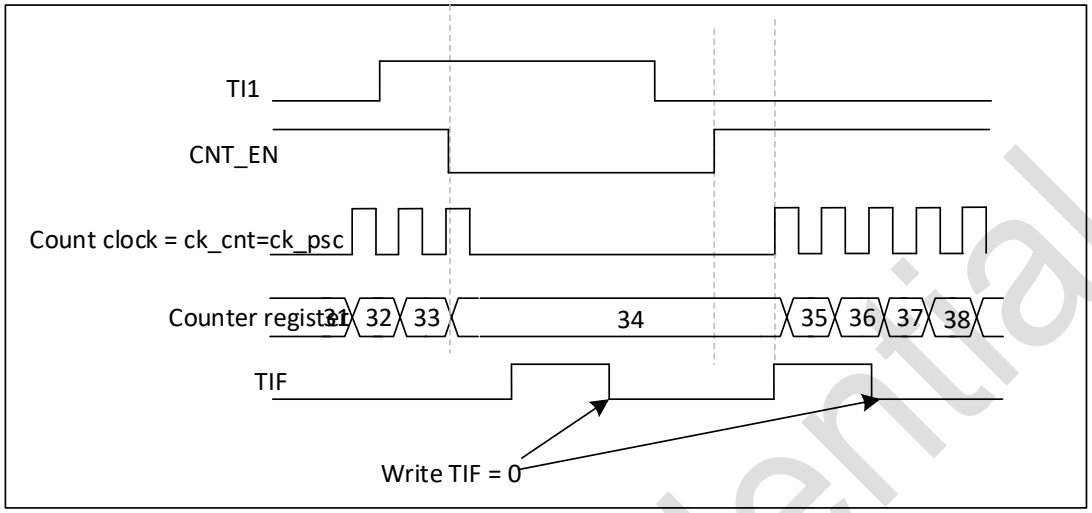


图 17-53 门控模式下的控制电路

从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

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

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

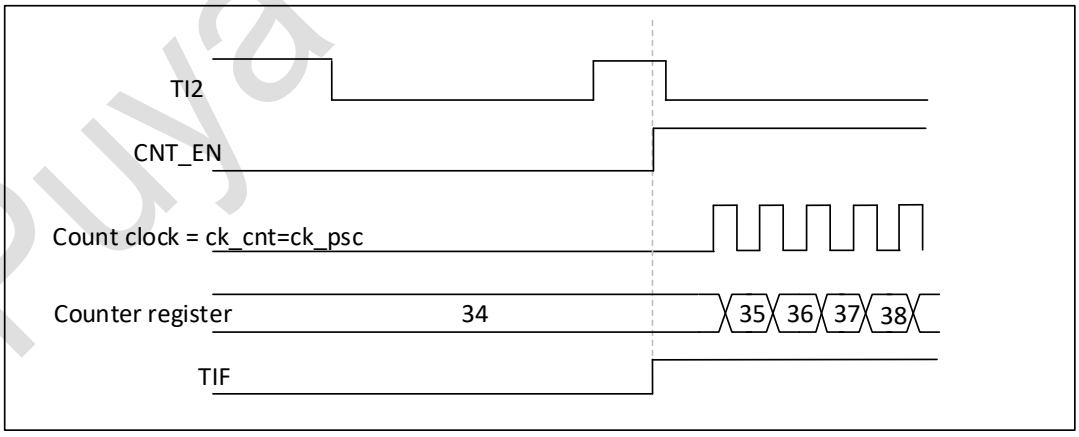


图 17-54 触发器模式下的控制电路

从模式：外部时钟模式 2 + 触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1. 通过 TIMx_SMCR 寄存器配置外部触发输入电路：
   1) ETF=0000：没有滤波
   2) ETPS=00：不用预分频器
   3) ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2
2. 按如下配置通道 1，检测 TI1 的上升沿：
   1) IC1F=0000：没有滤波
   2) 触发操作中不使用捕获预分频器，不需要配置
   3) 置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   4) 置 TIMx_CCER 寄存器中 CC1P=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=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时， TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

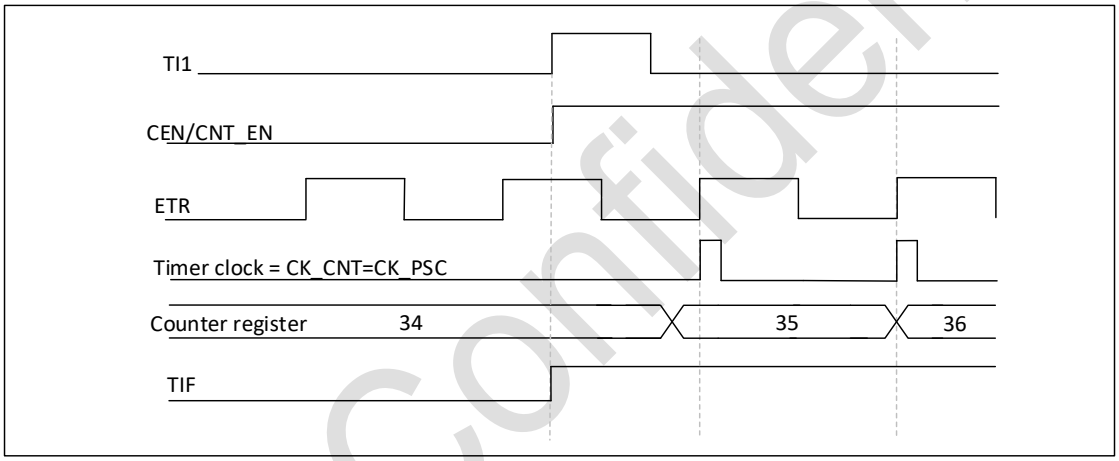


图 17-55 外部时钟模式 2 + 触发模式下的控制电路

17.3.22 DMA 突发模式

TIMx 定时器当发生单个事件时有生成多个 DMA 请求的能力。主要目的是能够多次重新编程定时器的部分功能而无需软件的开销。但是也可以在固定的间隔时间内读取一行中的几个寄存器。

DMA 控制器的目的地是唯一的并且必须指向虚拟寄存器 TIMx_DMAR.当给定的定时器事件发生，定时器发起一系列 DMA 请求。每个写入 TIMx_DMAR 寄存器的操作实际上是重定向一个定时器寄存器的。

TIMx_DCR 寄存器中的 DBL[4:0]位设置 DMA 突发长度。当对 TIMx_DMR 地址进行读取或写入访问时，定时器会识别突发传输，即传输次数（以半个字或字节为单位)

TIMx_DCR 的 DBA[4:0]位定义了 DMA 传输的 DMA 基地址(当通过 TIMx_DMAR 地址完成读写操作)。DBA 定义为从 TIMx_CR1 寄存器地址开始的偏移量。

例子：

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

作为一个例子，定时器 DMA 突发特性是用于发生更新事件时更新 CCRx 寄存器中的内容，通过 DMA 传输半字到 CCRx。

通过以下步骤来完成：

1. 如下配置对应的 DMA 通道：
    1) DMA 通道外设地址是 DMAR 寄存器的地址
    2) DMA 通道存储地址是 RAM 中的 BUFF 地址，包含了 DMA 传给 CCRx 的数据
    3) 传输的数据个数=3（见注释）
    4) 轮询模式关闭
2. 通过配置 DBA 和 DBL 实现配置 DCR 寄存器：DBL=3，DBA=0XE
3. 使能定时器更新 DMA 请求（UDE=1）
4. 使能定时器
5. 使能 DMA 通道

这此示例适用于每个 CCRx 寄存器更新一次的情况。如果每个 CCRx 都更新两次，数据的传输个数就需要设置为 6。举例假设 RAM 的 buffer 中包含了 data1, data2, data3, data4, data5 and data6。数据依次传输到 CCRx 中：第一次更新 DMA 请求，data1 传输给 CCR2，data2 传输给 CCR3，data3 传输给 CCR4；第二次更新 DMA 请求，data4 传输给 CCR2，data5 传输给 CCR3，data6 传输给 CCR4。

注：空值可以写入到保留寄存器。

17.3.23 TIM1 DMA 请求

TIM1 可以生成 DMA 请求，如下表所示：

表 17-2 DMA 请求

DMA 请求源	DMA 使能控制位
更新	UDE
比较/捕获 1	CC1DE
比较/捕获 2	CC2DE
比较/捕获 3	CC3DE
比较/捕获 4	CC4DE
COM	COMDE
触发	TDE

17.3.24 定时器同步

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

下图显示了触发选择和主模式选择模块的概况。

使用一个定时器作为另一个定时器的预分频器

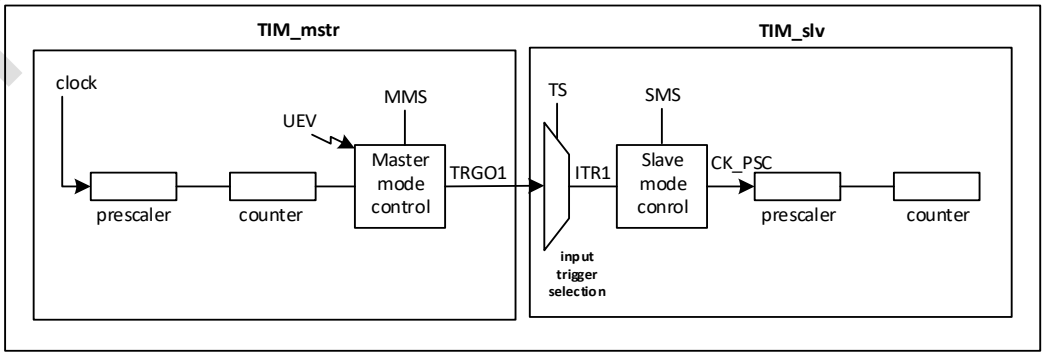


图 17-56 主/从定时器的例子

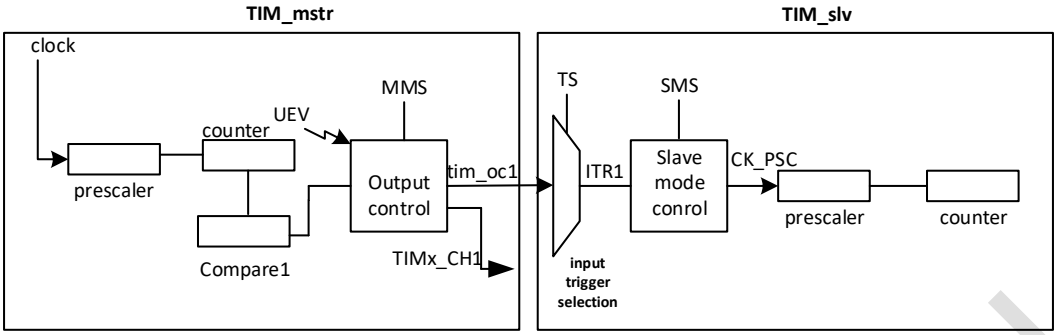


图 17-57 仅有 1 个通道定时器的主从连接示例

仅具有一个通道的计时器（见上图）不具有主模式。但是，tim_oc1 输出信号可以作为从定时器的触发器。tim_oc1 信号脉冲宽度必须编程为目标定时器的至少 2 个时钟周期，以确保从属定时器检测到触发。例如，如果目标定时器时钟比源定时器慢 4 倍，则 OC1 脉冲宽度必须为 8 个时钟周期。

如：可以配置 TIM_mstr 作为 TIM_slv 的预分频器。参考上图，进行下述操作：

- 配置 TIM_mstr 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM_mst_CR2 寄存器的 MMS='010'时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。
- 连接 TIM_mstr 的 TRGO1 输出至 TIM_slv，设置 TIM_slv_SMCR 寄存器的 TS='00000'，配置 TIM_slv 为使用 ITR1 作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式 1(TIM_slv_SMCR 寄存器的 SMS=111)；这样 TIM_slv 即可由 TIM_mstr 周期性的上升沿(即 TIM_mstr 的计数器溢出)信号驱动。
- 最后，必须设置相应 CEN 位分别启动两个定时器。

注：如果 OCx 已被选中为 TIM_mstr 的触发输出(MMS=1xx)，它的上升沿用于驱动 TIM_slv 的计数器。

使用一个定时器使能另一个定时器

在这个例子中，TIM_slv 的使能由 TIM_mstr 的输出比较控制。只当 TIM_mstr 的 OC1REF 为高时，TIM_slv 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/3)得到。

- 配置 TIM_mstr 为主模式，送出它的输出比较参考信号(OC1REF)为触发输出(TIM_mstr_CR2 寄存器的 MMS=100)
- 配置 TIM_mstr 的 OC1REF 波形(TIM_mst_CCMR1 寄存器)
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为门控模式(TIM_slv_SMCR 寄存器的 SMS=101)
- 置 TIM_slv_CR1 寄存器的 CEN=1 以使能 TIM_slv
- 置 TIM_mst_CR1 寄存器的 CEN=1 以启动 TIM_mstr

注： TIM_slv 的时钟不与 TIM_mstr 的时钟同步，这个模式只影响 TIM_slv 计数器的使能信号。

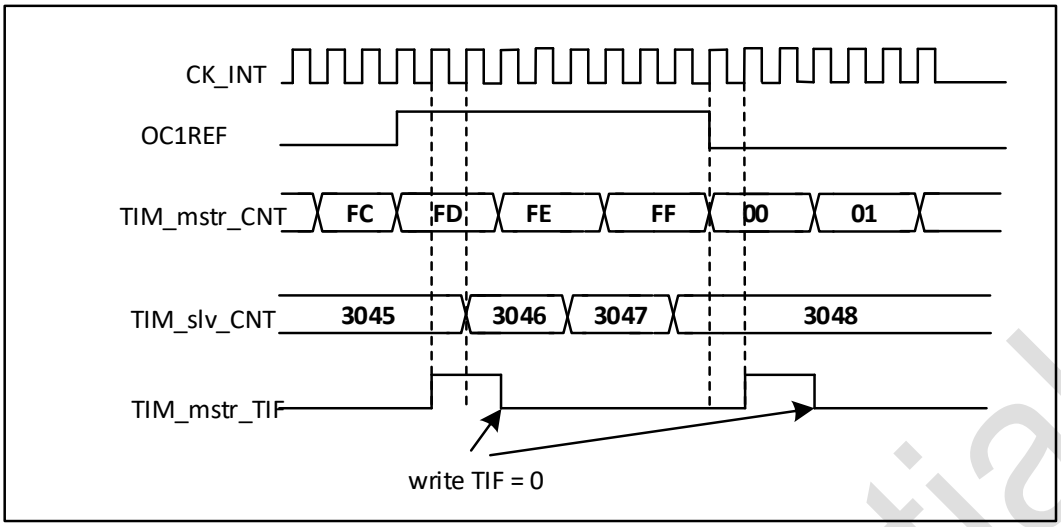


图 17-58 TIM_mstr 的 OC1REF 控制 TIM_slv

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

在下一个例子中，需要同步 TIM_mstr 和 TIM_slv。TIM_mstr 是主模式并从 0 开始，TIM_slv 是从模式并从 0xE7 开始;2 个定时器的预分频器系数相同。写‘0’到 TIM_mstr_CR1 的 CEN 位将禁止 TIM_mstr，TIM_slv 随即停止。

- 配置 TIM_mstr 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM_mst_CR2 寄存器的 MMS=100)。
- 配置 TIM_mstr 的 OC1REF 波形(TIM_mstr_CCMR1 寄存器)。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为门控模式(TIM_slv_SMCR 寄存器的 SMS=101)
- 置 TIM_mstr_EGR 寄存器的 UG=‘1’，复位 TIM_mstr。
- 置 TIM_slv_EGR 寄存器的 UG=‘1’，复位 TIM_slv。
- 写‘0XE7’至 TIM_slv 的计数器(TIM_slv_CNT)，初始化它为 0xE7。
- 置 TIM_slv_CR1 寄存器的 CEN=‘1’以使能 TIM_slv。
- 置 TIM_mstr_CR1 寄存器的 CEN=‘1’以启动 TIM_mstr。
- 置 TIM_mstr_CR1 寄存器的 CEN=‘0’以停止 TIM_mstr。

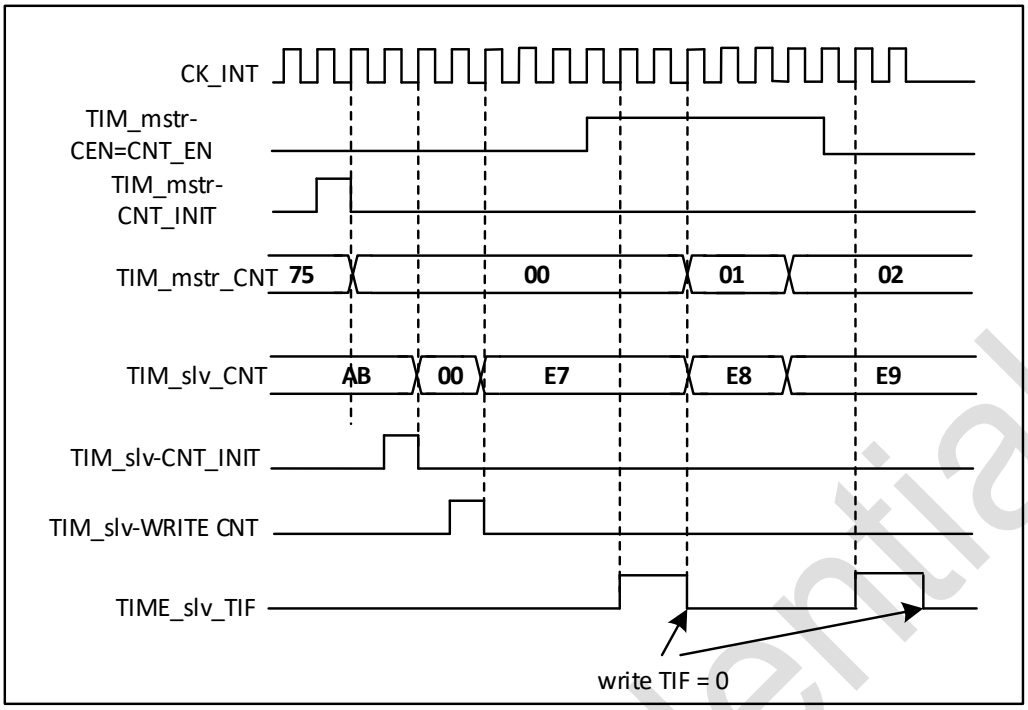


图 17-59 通过使能 TIM_mstr 可以控制 TIM_slv

使用一个定时器去启动另一个定时器

在这个例子中，使用 TIM_mstr 的更新事件使能 TIM_slv。一旦 TIM_mstr 产生更新事件，TIM_slv 即从它当前的数值(可以是非 0)按照分频的内部时钟开始计数。在收到触发信号时，TIM_slv 的 CEN 位被自动地置‘1’，同时计数器开始计数直到写‘0’到 TIM_slv_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/3)。

- 配置 TIM_mstr 为主模式，送出它的更新事件(UEV)做为触发输出(TIM_mstr_CR2 寄存器的 MMS=010)。
- 配置 TIM_mstr 的周期(TIM_mstr_ARR 寄存器)。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为触发模式(TIM_slv_SMCR 寄存器的 SMS=110)
- 置 TIM_mstr_CR1 寄存器的 CEN=1 以启动 TIM_mstr。

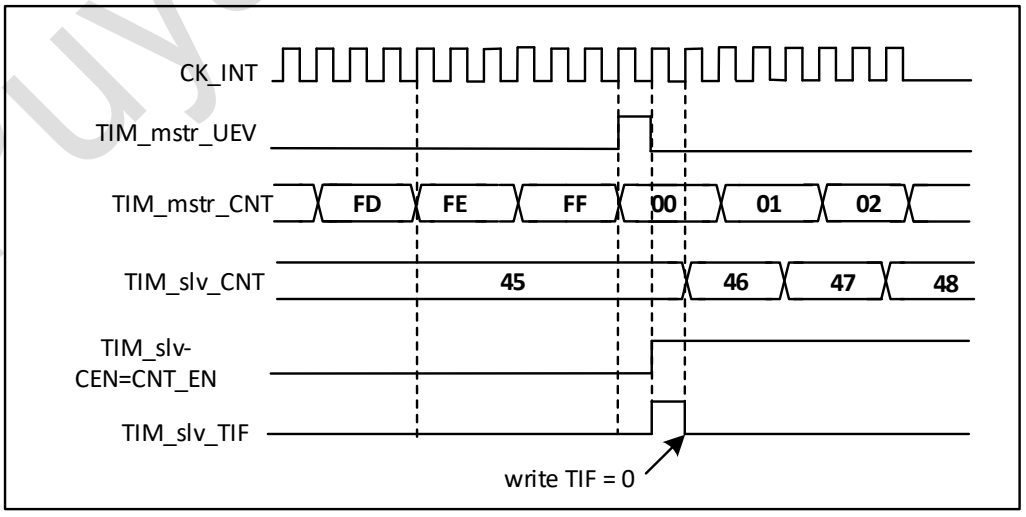


图 17-60 使用 TIM_mstr 的更新触发 TIM_slv

在上一个例子中，可以在启动计数之前初始化两个计数器。下图显示在与 0 相同配置情况下，使用触发模式而不是门控模式(TIM_slv_SMCR 寄存器的 SMS=110)的动作。

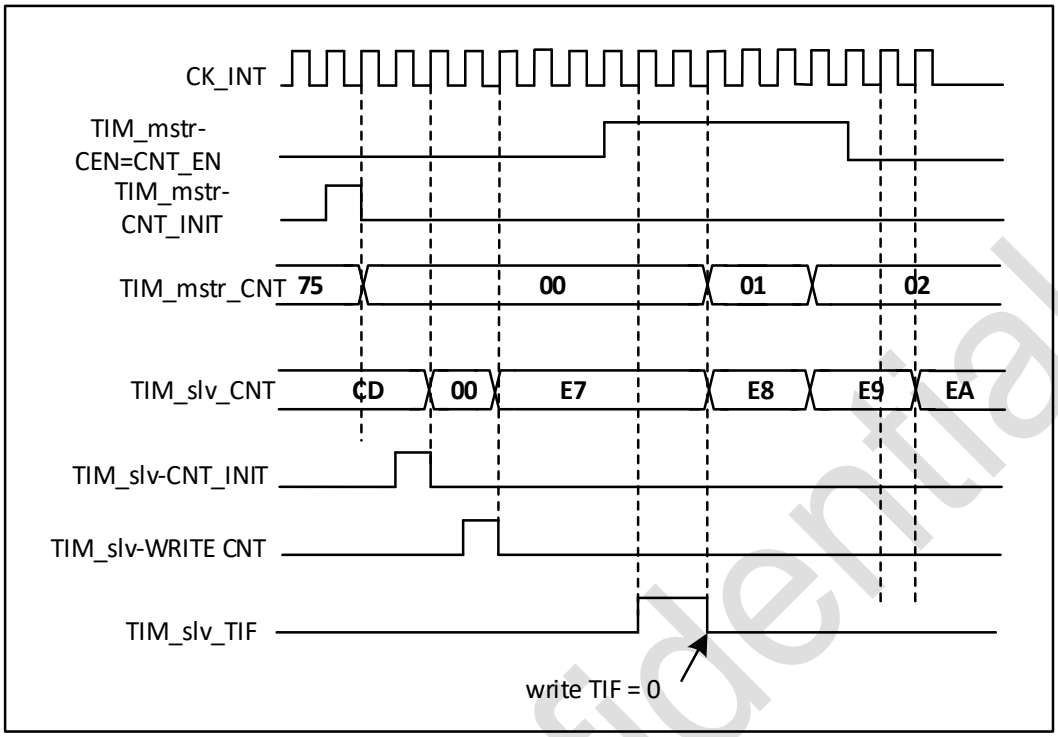


图 17-61 利用 TIM_mstr 的使能触发 TIM_slv

使用一个外部触发同步地启动 2 个定时器

这个例子中当 TIM_mstr 的 TI1 输入上升时使能 TIM_mstr，使能 TIM_mstr 的同时使能 TIM_slv。保证计数器的对齐，TIM_mstr 必须配置为主/从模式(对应 TI1 为从，对应 TIM_slv 为主)：

- 配置 TIM_mstr 为主模式，送出它的使能做为触发输出(TIM_mstr_CR2 寄存器的 MMS=001)。
- 配置 TIM_mstr 为从模式，从 TI1 获得输入触发(TIM_mstr_SMCR 寄存器的 TS=00100)。
- 配置 TIM_mstr 为触发模式(TIM_mstr_SMCR 寄存器的 SMS=110)。
- 配置 TIM_mstr 为主/从模式，TIM_mstr_SMCR 寄存器的 MSM=1。
- 配置 TIM_slv 从 TIM_mstr 获得输入触发(TIM_slv_SMCR 寄存器的 TS=00000)
- 配置 TIM_slv 为触发模式(TIM_slv_SMCR 寄存器的 SMS=110)。

当 TIM_mstr 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被设置。

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

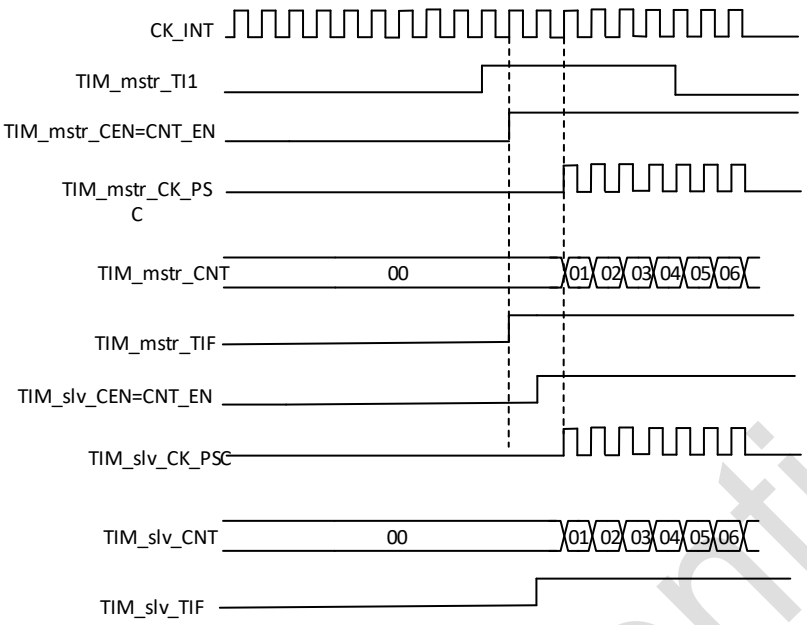


图 17-62 使用 TIM_mstr 的 TI1 输入触发 TIM_mstr 和 TIM_slv

17.3.25 调试模式

当微控制器进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，

TIMx 计数器可以或者继续正常操作，或者停止。详见随后的 DBG 节。

调试模式下的行为可以使用调试支持模块（DBG）中每个定时器的专用配置进行编程

为了安全的考虑，当计数器停止时，输出被禁止（如果 MOE 被复位了）。输出即可被强制到一个无效电平（OSSI=1），或者由 GPIO 控制器接管（OSSI=0），典型的是强制为高阻。

更多的细节，见 DBG 模块描述

17.4 TIM1 中断

表 17-3 中断说明

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
比较/捕获 3	CC3IF	CC3IE
比较/捕获 4	CC4IF	CC4IE
COM	COM	COMIE
触发	TIF	TIE
刹车	BIF	BIE
系统刹车	SBIF	/

17.5 TIM1 寄存器描述

17.5.1 TIMx 控制寄存器 1（TIMx_CR1)(x=1)

偏移地址：0x00

复位值：0x0000 0000

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

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

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD	RW	2'h0	时钟分频因子 (Clock division) 这 2 位定义在定时器时钟(CK_INT)频率和死区时间的分频比和死区发生器与数字滤波器(ETR,TIx)所用的采样时钟（t_{DTS}）之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 (Auto-reload preload enable) 0： TIMx_ARR 寄存器没有缓冲； 1： TIMx_ARR 寄存器被装入缓冲器。
6:5	CMS	RW	2'h0	选择中央对齐模式 (Center-aligned mode selection) 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 **注**：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	方向 (Direction) 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0:如果使能了更新中断或 DMA 请求,则下述任一事件产生更新中断或 DMA 请求: - 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生

				0：允许 UEV。更新(UEV)事件由下述任一事件产生： - 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、 CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	使能计数器 (Counter enable) 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

17.5.2 TIMx 控制寄存器 2（TIMx_CR2)(x=1)

偏移地址：0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	OIS6	Res.	OIS5
-	-	-	-	-	-	RW	-	-	-	-	-	-	RW	-	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	MMS[3]	RW	0	详见 MMS[2:0]描述
24:19	Reserved	-	-	保留
18	OIS6	RW	0	输出空闲状态 6(OC6 输出)。参见 OIS1 位。
17	Reserved	-	-	保留
16	OIS5	RW	0	输出空闲状态 5(OC5 输出)。参见 OIS1 位。
15	Reserved	-	-	保留
14	OIS4	RW	0	输出空闲状态 4(OC4 输出)。参见 OIS1 位。
13	OIS3N	RW	0	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位。
12	OIS3	RW	0	输出空闲状态 3(OC3 输出)。参见 OIS1 位。
11	OIS2N	RW	0	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。参见 OIS1 位。
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。

				注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7	TI1S	RW	0	TI1 选择 (TI1 selection) 0： TIMx_CH1 引脚连到 TI1 输入； 1： TIMx_CH1、 TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
6:4	MMS	RW	3'h0	主模式选择 (Master mode selection) 用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 0000： 复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 0001： 使能 – 计数器使能信号 CNT_EN 可被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 0010: 更新 – 更新事件被选为触发输出(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 0011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 0100：比较 – OC1REFC 信号被用于作为触发输出(TRGO)。 0101：比较 – OC2REFC 信号被用于作为触发输出(TRGO)。 0110：比较 – OC3REFC 信号被用于作为触发输出(TRGO)。 0111：比较 – OC4REFC 信号被用于作为触发输出(TRGO)。 1000：编码器时钟输出-编码器时钟信号作为触发输出（TRGO）。其他的 SMS 值可能导致预期之外的效果。 1001-1111：保留 注意： 1.从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。 2.若主从定时器不在同一总线上，主模式应该配置为能被从定时器采到的宽度。
3	CCDS	RW	0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0	捕获/比较控制更新选择 (Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。

1	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位 (Capture/compare preloaded control) 0： CCxE，CCxNE 和 OCxM 位不是预装载的； 1： CCxE，CCxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新 (COMG 位设置或 tim_trgi 上检测到的上升沿，具体取决于 CCUS 位）。 注：该位只对具有互补输出的通道起作用。

17.5.3 TIMx 从模式控制寄存器（TIMx_SMCR)(x=1)

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

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[4:3]	RW	2'h0	详见 TS[2:0]描述
19:17	Reserved	-	-	保留
16	SMS[3]	RW	0	详见 SMS[2:0]描述
15	ETP	RW	0	外部触发极性 (External trigger polarity) 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
14	ECE	RW	0	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1：设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=00111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是'00111')。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS	RW	2'h0	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4；

				11：ETRP 频率除以 8。
11:8	ETF	RW	4’h0	外部触发滤波（External trigger filter) 这些位定义了对ETRP 信号采样的频率和对ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	0	主/从模式（Master/slave mode) 0：无作用; 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS	RW	3’h0	触发选择（Trigger selection) 选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测（TI1F_ED） 00101：滤波后的定时器输入 1（TI1FP1） 00110：滤波后的定时器输入 2（TI1FP2） 00111：外部触发输入（ETRF） 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 01010：内部触发 6(ITR6) 01011：内部触发 7(ITR7) 01100：内部触发 8(ITR8) 01101：内部触发 9(ITR9) 01110：内部触发 10(ITR10) 01111：内部触发 11(ITR11) 10000：内部触发 12(ITR12)

				10001：内部触发 13(ITR13) 10010：内部触发 14(ITR14) 10011：内部触发 15(ITR15) 10100：内部触发 16(ITR16) 10101：内部触发 17(ITR17) 10110：内部触发 18(ITR18) 其它：保留 更多有关 ITRx 的细节，参见系统级联表 注：这些位只能在未用到(如 SMS=0000)时被改变，以避免在改变时产生错误的边沿检测。
3	OCCS	RW	0	OCREF 清除选择位（OCREF clear selection） 这位选择了 OCREF 的清除源。 0：OCREF_CLR_INT 连接 OCREF_CLR 输入 1：OCREF_CLR_INT 连接 ETRF
2:0	SMS	RW	3’h0	从模式选择（Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 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：正交编码器模式 1，x2 模式– 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 0010：正交编码器模式 2，x2 模式– 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 0011：正交编码器模式 3，x4 模式–根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 0100: 复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 0101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 0110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 0111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 1000：复位+触发组合模式：-触发输入（TRGI）的上升沿初始化计数器，生成寄存器更新并启动计数器 注：如果 TI1F_ED 被选为触发输入(TS=00100)时，不要使用门控模式。这是因为， TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。 注：必须在接收主定时器事件之前，使能从设备（定时器，ADC）来接收 TRGO，并且时钟频率不能再接收过程中改变 注：在编码器模式下，不要使用 UEV 作为 tim_trgo 输出信号，（即 mms 不能配置为 010）

17.5.4 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=1)

偏移地址：0x0C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	TDE	RW	0	允许触发 DMA 请求（Trigger DMA request enable) 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	允许 COM 的 DMA 请求（COM DMA request enable) 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
12	CC4DE	RW	0	允许捕获/比较 4 的 DMA 请求（Capture/Compare 4 DMA request enable) 0：禁止捕获/比较 4 的 DMA 请求 1：允许捕获/比较 4 的 DMA 请求
11	CC3DE	RW	0	允许捕获/比较 3 的 DMA 请求（Capture/Compare 3 DMA request enable) 0：禁止捕获/比较 3 的 DMA 请求 1：允许捕获/比较 3 的 DMA 请求
10	CC2DE	RW	0	允许捕获/比较 2 的 DMA 请求（Capture/Compare 2 DMA request enable) 0：禁止捕获/比较 2 的 DMA 请求 1：允许捕获/比较 2 的 DMA 请求
9	CC1DE	RW	0	允许捕获/比较 1 的 DMA 请求（Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	允许更新的 DMA 请求（Update DMA request enable) 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	允许刹车中断（Break interrupt enable) 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	触发中断使能（Trigger interrupt enable) 0：禁止触发中断 1：使能触发中断
5	COMIE	RW	0	允许 COM 中断（COM interrupt enable) 0：禁止 COM 中断 1：允许 COM 中断
4	CC4IE	RW	0	允许捕获/比较 4 中断（Capture/Compare 4 interrupt enable) 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断

3	CC3IE	RW	0	允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	允许更新中断 (Update interrupt enable) 0：禁止更新中断 1：允许更新中断

17.5.5 TIMx 状态寄存器（TIMx_SR)(x=1)

偏移地址：0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	TERRF	IERRF	DIRF	IDXF	IC2IR	IC1IR	CC6IF	CC5IF
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	SBIF	CC4OF	CC3OF	CC2OF	CC1OF	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	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
28	IC3IF	RC_W0	0	下降沿捕获 3 标志 参见 IC1IF 描述。
27	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
26	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生下降沿捕获事件。
25	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
24	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
23:20	Reserved	-	-	保留
19	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。

18	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生上升沿捕获事件。
17	CC6IF	RC_W0	0	捕获/比较 6 中断标记 (Capture/Compare 6 interrupt flag) 参考 CC1IF 描述。 注：通道 6 只能配置为输出
16	CC5IF	RC_W0	0	捕获/比较 5 中断标记 (Capture/Compare 5 interrupt flag) 参考 CC1IF 描述。 注：通道 5 只能配置为输出
15:14	Reserved	-	-	保留
13	SBIF	RC_W0	0	系统刹车中断标志（System break interrupt flag） 当系统刹车输入一旦有效，该标志信号会硬件置位。当系统刹车输入无效时可通过软件清零 该标志必须复位来重启 PWM 操作 0：无刹车事件发生 1：系统刹车输入处于有效。如果 TIMx_DIER 寄存器的 BIE 位=1 则会产生中断
12	CC4OF	RC_W0	0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。
11	CC3OF	RC_W0	0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。
10	CC2OF	RC_W0	0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1: 计数器的值被捕获到 TIMx_CCR1 寄存器时， CC1IF 的状态已经为‘1’。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标记 (Break interrupt flag) 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，则该位可由软件清‘0’。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。如果 TIM_DIER 的 BIE=1 则产生中断
6	TIF	RC_W0	0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置‘1’。它由软件清‘0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	COMIF	RC_W0	0	COM 中断标记 (COM interrupt flag)

				一旦产生 COM 事件(当捕获/比较控制位： CCxE、 CCxNE、OCxM 已被更新)该位由硬件置‘1’。它由软件清‘0’。 0：无 COM 事件产生； 1： COM 中断等待响应。
4	CC4IF	RC_W0	0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
3	CC3IF	RC_W0	0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
2	CC2IF	RC_W0	0	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 0：无匹配发生； 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下， CC1IF 位变高 如果通道 TI1 配置为输入模式： 当捕获事件发生时该位由硬件置‘1’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： − 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当计数器 CNT 被触发事件重新初始化时。 (参考从模式控制寄存器(TIMx_SMCR))。

17.5.6 TIMx 事件产生寄存器（TIMx_EGR)(x=1)

偏移地址：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
								W	W	W	W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留

7	BG	W	0	产生刹车事件 (Break generation) 该位由软件置'1'，用于产生一个刹车事件，由硬件自动清'0'。 0：无动作 1：产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
6	TG	W	0	产生触发事件 (Trigger generation) 该位由软件置'1'，用于产生一个触发事件，由硬件自动清'0'。 0：无动作 1： TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
5	COMG	W	0	捕获/比较事件，产生控制更新 (Capture/Compare control update generation) 该位由软件置'1'，由硬件自动清'0'。 0：无动作 1：当 CCPC=1，允许更新 CCxE、 CCxNE、 OCxM 位 注： 该位只对拥有互补输出的通道有效。
4	CC4G	W	0	产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 CC1G 描述。
3	CC3G	W	0	产生捕获/比较 3 事件 (Capture/Compare 3 generation) 参考 CC1G 描述。
2	CC2G	W	0	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。
1	CC1G	W	0	产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1'，用于产生一个捕获/比较事件，由硬件自动清'0'。 0：无动作 1：在通道 1 上产生一个捕获/比较事件： 若通道 1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 (Update generation) 该位由软件置'1'，由硬件自动清'0'。 0：无动作 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清'0'(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

17.5.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1)(x=1)

偏移地址：0x18

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC2M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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]	
IC2F[3:0]				IC2PSC[1:0]				ICIF[3:0]				ICIPSC[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	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC2M[3]	RW	0	详见 OC2M[2:0]描述
23:17	Reserved	-	-	保留
16	OC1M[3]	RW	0	详见 OC1M[2:0]描述
15	OC2CE	RW	0	输出比较 2 清零使能 (Output Compare 2 clear enable)
14:12	OC2M	RW	3'h0	输出比较 2 模式 (Output Compare 2 mode)(支持移相模式)
11	OC2PE	RW	0	输出比较 2 预装载使能 (Output Compare 2 preload enable)
10	OC2FE	RW	0	输出比较 2 快速使能 (Output Compare 2 fast enable)
9:8	CC2S	RW	2'h0	捕获/比较 2 选择。 (Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0	输出比较 1 清'0'使能 (Output Compare 1 clear enable) 0： OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
6:4	OC1M	RW	3'h0	输出比较 1 模式 (Output Compare 1 mode) (支持移相模式) 这些位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、 OC1N 的值。 OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、 CC1NP 位。 0000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 0001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。

				0010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 0011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 0100：强制为无效电平。强制 OC1REF 为低。 0101：强制为有效电平。强制 OC1REF 为高。 0110： PWM 模式 1 移相模式： -在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1_H 时通道 1 为无效电平，TIMx_CNT≤TIMx_CCR1_H 时通道 1 为有效电平。 非移相模式： - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 0111： PWM 模式 2 移相模式： -在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1_H 时通道 1 为有效电平，TIMx_CNT≤TIMx_CCR1_H 时通道 1 为无效电平。 非移相模式： - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 1000: 可重触发 OPM 模式 1-在递增计数模式下，通道处于有效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且通道在下一次更新时再次有效。在递减计数模式下，通道处于无效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且信道在下一次更新时再次变为无效 1001: 可重触发 OPM 模式 2-在递增计数模式下，信道处于无效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且通道在下一次更新时再次变为无效。在递减计数模式下，信道处于有效状态，直到检测到触发事件（tim_trgi 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且信道在下一次更新时再次变为有效 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。

				注 2：在 PWM 模式中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。 注 3：当通道有互补输出时，该位段可以预加载。如果 TIMx_CR2 的 CCPC 位置 1，然后仅当 COM 事件生成时 OC1M 有效位从预加载中更新新的值
3	OC1PE	RW	0	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快输出对触发输入事件的响应。必须在单脉冲模式（TIMx_CR1 寄存器的 OPM 位置 1）中使用，当触发到来时实现脉冲快速输出。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。仅能用于 PWM 模式 1 和 PWM 模式 2
1:0	CC1S	RW	2’h0	捕获/比较 1 选择。 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC1 通道被配置为输出； 01： CC1 通道被配置为输入， IC1 映射在 TI1 上； 10： CC1 通道被配置为输入， IC1 映射在 TI2 上； 11： CC1 通道被配置为输入， IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F	RW	4’h0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	RW	2’h0	输入/捕获 2 预分频器 (Input capture 2 prescaler)
9:8	CC2S	RW	2’h0	捕获/比较 2 选择 (Capture/Compare 2 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择：

				00： CC2 通道被配置为输出； 01： CC2 通道被配置为输入，IC2 映射在 TI2 上； 10： CC2 通道被配置为输入，IC2 映射在 TI1 上； 11： CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注 CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F	RW	4’h0	输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 0010：采样频率 $f_{SAMPLING}= f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}= f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC	RW	2’h0	输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1S=00，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	2’h0	捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。

				注: CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

17.5.8 TIMx 捕获/比较模式控制寄存器 2（TIMx_CCMR2)(x=1)

偏移地址：0x1C

复位值： 0x0000 0000

参看以上 CCMR1 寄存器的描述

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

■ 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC4M[3]	RW	0	详见 OC4M[2:0]描述
23:17	Reserved	-	-	保留
16	OC3M[3]	RW	0	详见 OC3M[2:0]描述
15	OC4CE	RW	0	输出比较 4 清零使能 (Output Compare 4 clear enable)
14:12	OC4M	RW	3'h0	输出比较 4 模式 (Output Compare 4 mode) (不支持移相模式)
11	OC4PE	RW	0	输出比较 4 预装载使能 (Output Compare 4 preload enable)
10	OC4FE	RW	0	输出比较 4 快速使能 (Output Compare 4 fast enable)
9:8	CC4S	RW	2'h0	捕获/比较 4 选择。 (Capture/Compare 4 selection) 该 2 位定义通道的方向(输入/输出)，及输入脚的选择: 00：CC4 通道被配置为输出; 01：CC4 通道被配置为输入，IC4 映射在 TI4 上; 10：CC4 通道被配置为输入，IC4 映射在 TI3 上; 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注 CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7	OC3CE	RW	0	输出比较 3 清'0'使能 (Output Compare 1clear enable)
6:4	OC3M	RW	3'h0	输出比较 3 模式 (Output Compare 3 mode)参考 OC1M
3	OC3PE	RW	0	输出比较 3 预装载使能 (Output Compare 3 preload enable)
2	OC3FE	RW	0	输出比较 3 快速使能 (Output Compare 3 fast enable)
1:0	CC3S	RW	2'h0	捕获/比较 3 选择。(Capture/Compare 3 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择: 00：CC3 通道被配置为输出; 01：CC3 通道被配置为输入，IC3 映射在 TI3 上; 10： CC3 通道被配置为输入，IC3 映射在 TI4 上; 11： CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注 CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC4F	RW	4'h0	输入捕获 4 滤波器 (Input capture 4 filter)
11:10	IC4PSC	RW	2'h0	输入/捕获 4 预分频器 (Input capture 4 prescaler)
9:8	CC4S	RW	2'h0	捕获/比较 4 选择 (Capture/Compare 4 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入， IC4 映射在 TI4 上； 10：CC4 通道被配置为输入， IC4 映射在 TI3 上； 11：CC4 通道被配置为输入， IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F	RW	4'h0	输入捕获 3 滤波器 (Input capture 3 filter)
3:2	IC3PSC	RW	2'h0	输入/捕获 3 预分频器 (Input capture 3 prescaler)
1:0	CC3S	RW	2'h0	捕获/比较 3 选择 (Capture/Compare 3 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入， IC3 映射在 TI3 上； 10：CC3 通道被配置为输入， IC3 映射在 TI4 上； 11：CC3 通道被配置为输入， IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

17.5.9 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=1)

偏移地址：0x20

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC6P	CC6E	Res.	Res.	CC5P	CC5E
										RW	RW			RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21	CC6P	RW	0	输入/捕获 6 输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。
20	CC6E	RW	0	输入/捕获 6 输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。

19:18	Reserved	-	-	保留
17	CC5P	RW	0	输入/捕获 5 输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。
16	CC5E	RW	0	输入/捕获 5 输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。
15：14	Reserved	-	-	保留
13	CC4P	RW	0	输入/捕获 4 输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。
12	CC4E	RW	0	输入/捕获 4 输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。
11	CC3NP	RW	0	输入/捕获 3 互补输出极性 (Capture/Compare 3 output polarity) 参考 CC1NP 的描述。
10	CC3NE	RW	0	输入/捕获 3 互补输出使能 (Capture/Compare 3 output enable) 参考 CC1NE 的描述。
9	CC3P	RW	0	输入/捕获 3 输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。
7	CC2NP	RW	0	输入/捕获 2 互补输出极性 (Capture/Compare 2 output polarity) 参考 CC1NP 的描述。
6	CC2NE	RW	0	输入/捕获 2 互补输出使能 (Capture/Compare 2 output enable) 参考 CC1NE 的描述。
5	CC2P	RW	0	输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) 0： OC1N 高电平有效; 1： OC1N 低电平有效。 注: 一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	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、 OIS1、 OIS1N 和 CC1E 位的值。 1: 开启 - OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1E 位的值。
1	CC1P	RW	0	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; 1： OC1 低电平有效。

				CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效 (捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效 (捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效 (捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相 (门控模式) 。这个配置不能应用于编码器模式下。 注： 1.对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1P 的实际有效位只有在 com 事件发生时才会加载预载值。 2.一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0	输入/捕获 1 输出使能 (Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、 OSSI、 OSSR、 OIS1、 OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止 1：捕获使能 注： 对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

表 17-4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE位	OSSI位	OSSR位	CCxE位	CCxNE位	OCx输出状态	OCxN输出状态
1	x	0	0	0	输出禁止（与定时器断开） OCx=0, OCx_EN=0	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	0	1	输出禁止（与定时器断开） OCx=0, OCx_EN=0	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP, OCx=0 OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP, OCx=0 OCxN_EN=0
		1	0	1	关闭状态（输出使能且为无效电平） OCx=CCxP, OCx_EN=1	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平） OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
0	0	x	x	x	输出禁止（与定时器断开）	
	1		0	0		
	1		0	1	关闭状态（输出使能但为无效电平） 异步的：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCx_EN=1;（如果刹车或刹车2触发了） 然后若时钟存在（仅刹车1触发时有效，不需要刹车2）：经过一个死区时间后OCx=OISx，OCxN=OISxN，假设OISx与OISxN并不都对应OCx和OCxN的有效电平 注：刹车2仅用于OSSI=OSSR=1	
	1		1	0		
	1		1	1		

特别的，当死区有效时，输出总是按照死区内输出的规则输出，尽管此时 moe 可能已经被软件或硬件的方式置为 1。

如果一个通道的 2 个输出都没有使用(CCxE = CCxNE = 0)，那么 OISx, OISxN, CCxP 和 CCxNP 都必须清零。

17.5.10 TIMx 计数器（TIMx_CNT)(x=1)

偏移地址：0x24

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT	RW	16'h0	计数器的值 (Counter value)

17.5.11 TIMx 预分频器 (TIMx_PSC)(x=1)

偏移地址：0x28

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16'h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清'0'或被工作在复位模式的从控制器清'0'。

17.5.12 TIMx 自动重装载寄存器 (TIMx_ARR)(x=1)

偏移地址：0x2C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	ARR	RW	20'hFFFF	自动重装载的值 (Prescaler value) 自动重装载的值为空时，计数器不工作。 无抖动模式（DITHEN=0）： 该寄存器保持自动加载值 抖动模式（DITHEN=1）： 该寄存器保持整数部分 ARR[19:4], ARR[3:0]包含抖动部分

17.5.13 TIMx 重复计数寄存器 (TIMx_RCR)(x=1)

偏移地址：0x30

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	REP	RW	16'h0	重复计数器的值 (Repetition counter value) 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器达到 0，会产生一个更新事件并且重复计数器重新从 REP 值开始计数。对 TIMx_RCR 寄存器写入的新值只在下次更新事件发生时才起作用。 这意味着在 PWM 模式中， (REP+1)对应着： - 在边沿对齐模式下， PWM 周期的数目； - 在中心对称模式下， PWM 半周期的数目；

17.5.14 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=1)

偏移地址：0x34

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1_H[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	CCR1_H[31:16]	RW	16'h0	输出比较 1 寄存器的值 若 CC1 通道配置为中央对齐 pwm 模式输出： CCR1_H 包含了装入向下计数比较 1 寄存器的值（预装载值）。
15:0	CCR1	RW/R	16'h0	捕获/比较通道 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出： CCR1 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。TIMx_CCR1 寄存器只读

17.5.15 TIMx 捕获/比较寄存器 2（TIMx_CCR2)(x=1)

偏移地址：0x38

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2_H[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	CCR2_H[31:16]	RW	16'h0	输出比较 2 寄存器的值 若 CC2 通道配置为中央对齐 pwm 模式输出： CCR2_H 包含了装入向下计数比较 2 寄存器的值（预装载值）。
15:0	CCR2	RW/R	16'h0	捕获/比较通道 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出： CCR2 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。TIMx_CCR2 寄存器只读

17.5.16 TIMx 捕获/比较寄存器 3（TIMx_CCR3)(x=1)

偏移地址：0x3C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3_H[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	CCR3_H[31:16]	RW	16'h0	输出比较 3 寄存器的值 若 CC3 通道配置为中央对齐 pwm 模式输出： CCR3_H 包含了装入向下计数比较 3 寄存器的值（预装载值）。
15:0	CCR3	RW/R	16'h0	捕获/比较通道 3 的值 (Capture/Compare 3 value) 若 CC3 通道配置为输出：

				CCR3 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 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。 TIMx_CCR3 寄存器只读

17.5.17 TIMx 捕获/比较寄存器 4（TIMx_CCR4)(x=1)

偏移地址：0x40

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4	RW/R	16'h0	捕获/比较通道 4 的值 (Capture/Compare 4 value) 若 CC4 通道配置为输出： CCR4 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 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。 TIMx_CCR4 寄存器只读

17.5.18 TIMx 刹车和死区寄存器（TIMx_BDTR)(x=1)

偏移地址：0x44

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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：根据锁定设置，BKBID/BK2BID/BK2P/BK2E/BK2F/AOE/BKP/BKE/OSSI、OSSR 和 DTG[7:0]位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1: 如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。 有关 OC/OCN 使能的细节，参见 TIMx 捕获/比较使能寄存(TIMx_CCER)。
14	AOE	RW	0	自动输出使能 (Automatic output enable) 0： MOE 只能被软件置'1'； 1： MOE 能被软件置'1'或在下一个更新事件被自动置'1'(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 (Break polarity) 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 (Break enable) 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下"关闭状态"选择 (Off-state selection for Run mode) 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下"关闭状态"选择 (Off-state selection for Idle mode) 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1， OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。

				注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	2'h0	锁定设置 (Lock configuration) 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01: 锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKBID、BK2BID、BKE、 BKP、 AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出， CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCxNP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出， CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	8'h0	死区发生器设置 (Dead-time generator setup) 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg， Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16μs 到 31750ns，若步长时间为 250ns； 32μs 到 63μs，若步长时间为 1μs； 64μs 到 126μs，若步长时间为 2μs； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、 2 或 3，则不能修改这些位。

17.5.19 TIMx 捕获/比较寄存器 5 (TIMx_CCR5)(x=1)

偏移地址：0x48

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR5[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR5	RW/R	16'h0	捕获/比较通道 5 的值(Capture/Compare 5 value) 若 CC5 通道配置为输出： CCR5 包含了装入当前捕获/比较 5 寄存器的值(预装载值)。

				如果在 TIMx_CCMR3 寄存器(OC5PE 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 5 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC5 端口上产生输出信号。 无抖动模式（DITHEN=0）： 该寄存器保持 CCR5[15:0]的比较值，CCR5[19:16]位被复位 抖动模式（DITHEN=1）： 该寄存器保持整数部分 CCR5[19:4]，CCR5[3:0]包含抖动部分

17.5.20 TIMx 捕获/比较寄存器 6（TIMx_CCR6)(x=1)

偏移地址：0x4C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR6[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR6	RW/R	16’h0	捕获/比较通道 6 的值(Capture/Compare 6 value) 若 CC6 通道配置为输出： CCR6 包含了装入当前捕获/比较 6 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC6PE 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 6 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC6 端口上产生输出信号。

17.5.21 TIMx 捕获/比较模式控制寄存器 3（TIMx_CCMR3)(x=1)

偏移地址：0x50

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC6M[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	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	Res.	Res.	OC5CE	OC5M[2:0]			OC5PE	OC5FE	Res.	Res.
RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW		

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	OC6M[3]	RW	0	详见 OC6M[2:0]描述
23:17	Reserved	-	-	保留

16	OC5M[3]	RW	0	详见 OC5M[2:0]描述
15	OC6CE	RW	0	输出比较 6 清零使能 (Output Compare 6 clear enable)
14:12	OC6M	RW	3'h0	输出比较 6 模式 (Output Compare 6 mode) 参照 OC5M[3:0]描述
11	OC6PE	RW	0	输出比较 6 预装载使能 (Output Compare 6 preload enable)
10	OC6FE	RW	0	输出比较 6 快速使能 (Output Compare 6 fast enable)
9:8	Reserved	-	-	Reserved
7	OC5CE	RW	0	输出比较 5 清'0'使能 (Output Compare 5 clear enable)
6:4	OC5M	RW	3'h0	输出比较 5 模式 (Output Compare 5 mode) 这些位定义了输出参考信号 OC5REF 的动作，而 OC5REF 决定了 OC5、 OC5N 的值。 OC5REF 是高电平有效，而 OC5、 OC5N 的有效电平取决于 CC5P、 CC5NP 位。 0000：冻结。输出比较寄存器 TIMx_CCR5 与计数器 TIMx_CNT 间的比较对 OC5REF 不起作用； 0001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR5)相同时，强制 OC5REF 为高。 0010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR5)相同时，强制 OC5REF 为低。 0011：翻转。当 TIMx_CCR5=TIMx_CNT 时，翻转 OC5REF 的电平。 0100：强制为无效电平。强制 OC5REF 为低。 0101：强制为有效电平。强制 OC5REF 为高。 0110：PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR5 时通道 5 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR5 时通道 5 为无效电平(OC5REF=0)，否则为有效电平(OC5REF=1)。 0111：PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR5 时通道 5 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR5 时通道 5 为有效电平，否则为无效电平。 1000：可重触发 OPM 模式 1-在递增计数模式下，通道处于有效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且通道在下一次更新时再次有效。在递减计数模式下，通道处于无效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 1 中那样执行比较，并且信道在下一次更新时再次变为不活动 1001：可重触发 OPM 模式 2-在递增计数模式下，信道处于无效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且通道在下一次更新时再次变为无效。在递减计数模式下，信道处于有效状态，直到检测到触发事件（TRGI 信号）。然后，如在 PWM 模式 2 中那样执行比较，并且信道在下一次更新时再次变为有效

				其它：保留 注 1: 一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 在 PWM 模式中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC5REF 电平才改变。 注 3：当通道有互补输出时，该位段可以预加载。如果 TIMx_CR2 的 CCPC 位置 1，然后仅当 COM 事件生成时 OC5M 有效位从预加载中更新新的值 注 4：使用可重复 OPM 模式时，计数模式不要配置为中央计数模式
3	OC5PE	RW	0	输出比较 5 预装载使能 (Output Compare 5 preload enable)
2	OC5FE	RW	0	输出比较 5 快速使能 (Output Compare 5 fast enable)
1:0	Reserved	-	-	保留

17.5.22 TIMx 输入选择寄存器（TIMx_TISEL)(x=1)

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	TI4SEL	RW	4’h0	TI4 输入选择. 0000：TIMx_CH4 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
23:20	Reserved	-	-	保留
19:16	TI3SEL	RW	4’h0	TI3 输入选择. 0000：TIMx_CH3 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
15:12	Reserved	-	-	保留
11:8	TI2SEL	RW	4’h0	TI3 输入选择. 0000：TIMx_CH3 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

7:4	Reserved	-	-	保留
3:0	TI1SEL	RW	4'h0	TI1 输入选择. 0000：TIMx_CH1 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

17.5.23 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=1)

偏移地址：0x60

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL [1:0]		BK CMP4P	BK CMP3P	BK CMP2P	BK CMP1P	BKINP	Res.	Res.	Res.	Res.	BK CMP4E	BK CMP3E	BK CMP2E	BK CMP1E	BKINE
RW	RW	RW	RW	RW	RW	RW					RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20	PWM_PHS_EN	RW	0	PWM 移相使能 0：关 1：开
19:18	INTR_SEL	RW	0	00:上溢或下溢中断 01:上溢中断 10:下溢中断 11:保留 注：不使用移相功能时，此位配置为 0
17:14	ETRSEL	RW	4'h0	外部触发源选择（etr_in source selection） 该位段用于选择 ETR 输入源 0000：TIMx_ETR 0001：COMP1_OUT 0010：COMP2_OUT 0011：ADC_AWD1 0100：ADC_AWD2 0101：ADC_AWD3 0110：保留 0111：保留 其它：保留 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
13：12	Reserved	-	-	保留
11	BKCMP2P	RW	0	tim_brk_cmp2 输入极性 该位选择 brk_cmp2 输入极性，必须与 BKP 极性位同时配置

				0：tim_brk_cmp2 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp2 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	tim_brk_cmp1 输入极性 该位选择 brk_cmp1 输入极性，必须与 BKP 极性位同时配置 0：tim_brk_cmp1 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp1 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性（TIMx_BKIN input polarity） 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	tim_brk_cmp2 输入使能（tim_brk_cmp2 enable） 该位用于刹车输入时使能 tim_brk_cmp2。 tim_brk_cmp2 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp2 输入关闭 1：tim_brk_cmp2 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	tim_brk_cmp1 输入使能（tim_brk_cmp1 enable） 该位用于刹车输入时使能 tim_brk_cmp1。 tim_brk_cmp1 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp1 输入关闭 1：tim_brk_cmp1 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能（TIMx_BKIN input enable） 该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭

				1：BKIN 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

17.5.24 TIMx 备用功能选项寄存器 2（TIMx_AF2)(x=1)

偏移地址：0x64

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18:16	OCRSEL	RW	3’h0	OCREF 复位源选择（ocref_clr source selection） 该位段用于选择 ocref_clr 输入源 000：COMP1_OUT 001：COMP2_OUT 其它：保留 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
15:0	Reserved	-	-	保留

17.5.25 TIMx DMA 控制寄存器（TIMx_DCR)(x=1)

偏移地址：0x3DC

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5’h0	DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000： 1 次传输 00001： 2 次传输 00010： 3 次传输 .... 10001： 24 次传输
7:5	Reserved	-	-	保留
4:0	DBA	RW	5’h0	DMA 基地址 (DMA base address)

				这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， …….

17.5.26 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=1)

偏移地址：0x3E0

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	32’h0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： TIMx_CR1 地址 + (DBA + DMA 索引)x4 其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

18. 通用定时器（TIM2/TIM3/TIM4）

18.1 TIM2/TIM3/TIM4 简介

通用控制定时器(TIMx)由一个 32 位或 16 位的自动装载计数器组成，计数器由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。通用控制定时器(TIMx)和高级定时器(TIMx)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 TIM2/TIM3/TIM4 主要特征

主要功能包括：

- 32 位（TIM2）或 16 位（TIM3/TIM4）
- 向上、向下、向上/下的自动装载计数器
- 16 位可编程(可以实时修改)的预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘或中央对齐模式)
  - 单脉冲模式输出
- 通过外部信号来控制定时器与定时器之间互联的同步电路
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
- 支持针对定位的增量(正交)编码器
- 触发输入作为外部时钟或者按周期的电流管理

18.2.1 TIM2/TIM3/TIM4 模块框图

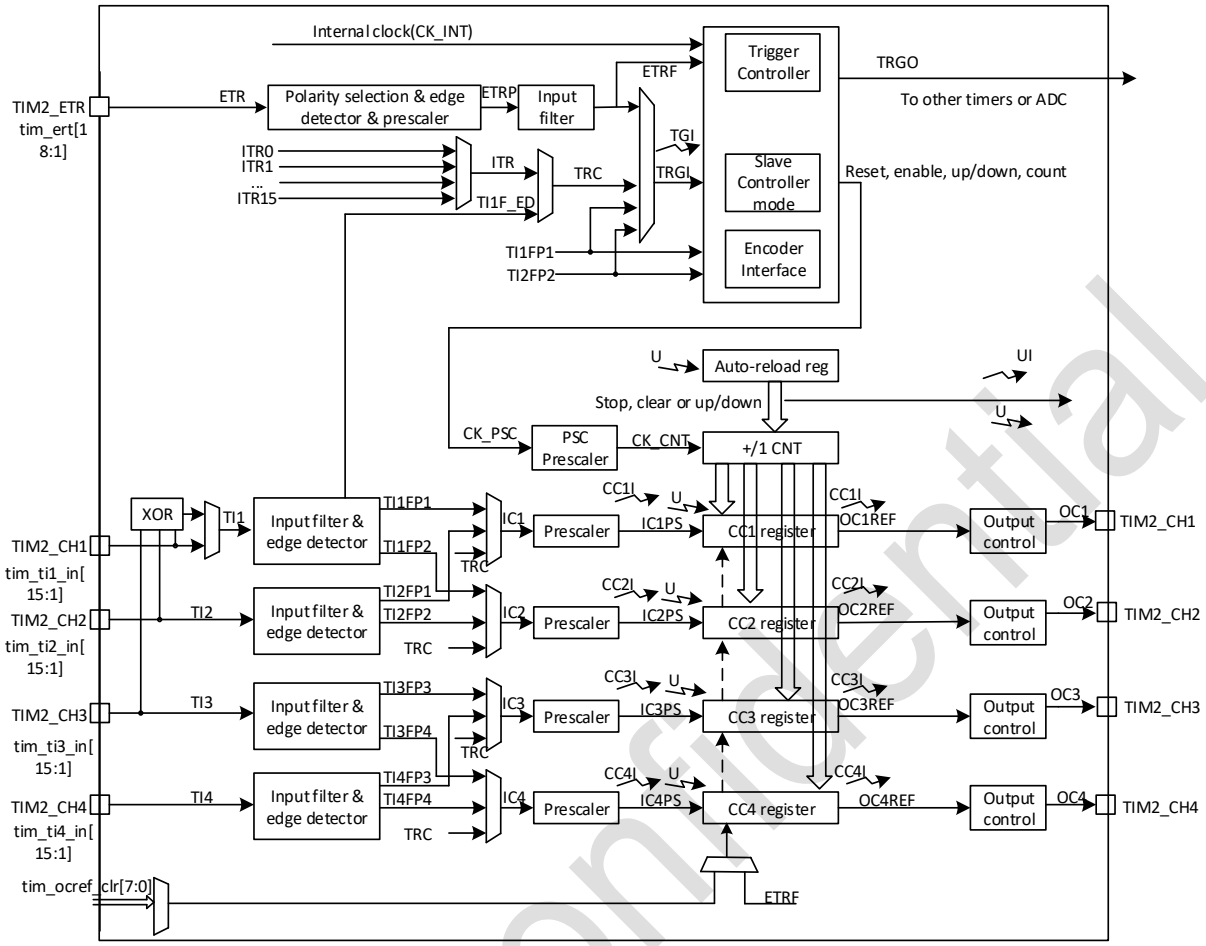


图 18-1 TIM 功能框图

18.3 TIM2/TIM3/TIM4 功能描述

18.3.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。计数器的时钟可以被预分频器分频。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器（TIMx_CNT）
- 预分频器寄存器（TIMx_PSC）
- 自动装载寄存器（TIMx_ARR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问它的预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件（UEV)时传送到影子寄存器。当计数器达到溢出条件(上溢或者下溢)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，会产生更新事件。更新事件也可以由软件及其他条件产生。后续会详细描述每一种配置下更新事件的产生。

计数器由预分频器分频后的时钟输出 CK_CNT 驱动，仅当设置了 TIMx_CR1 寄存器中的计数器使能位(CEN)，CK_CNT 才对计数器有效。(更多有关使能计数器的细节，请参见从模式控制器的描述)。

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

预分频器描述

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

下边几张图给出了在预分频器运行时，更改计数器参数的例子。

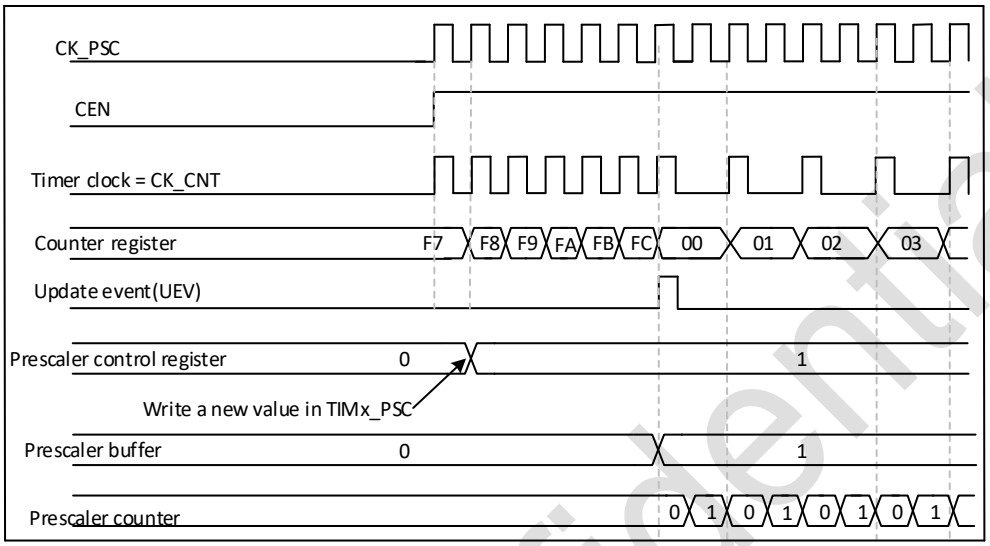


图 18-1 当预分频器的参数从 1 变到 2 时，计数器的时序图

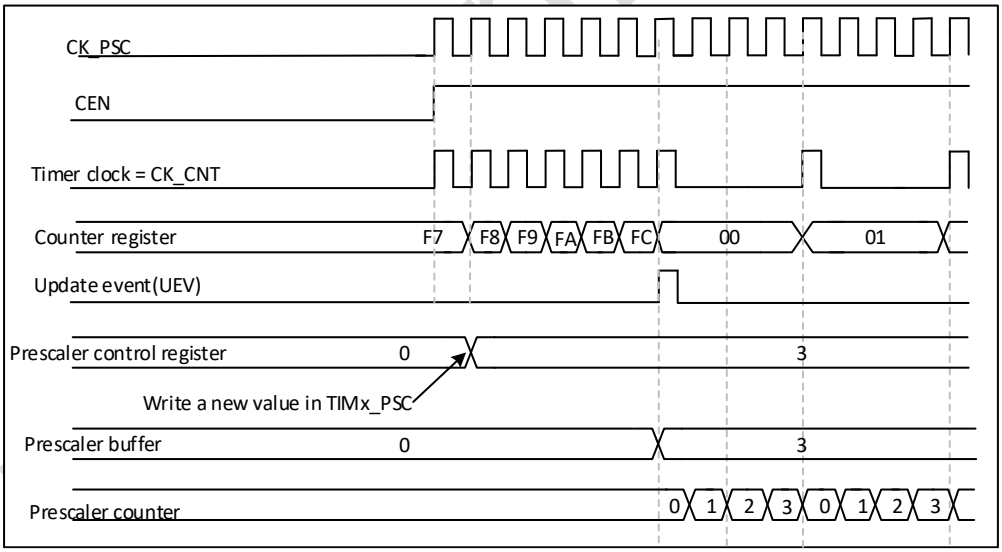


图 18-2 当预分频器的参数从 1 变到 4 时，计数器的时序图

18.3.2 计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值（TIMx_ARR 的内容），然后重新从 0 开始计数并且产生一个计数上溢事件。而在 TIMx_EGR 寄存器中（通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

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

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

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

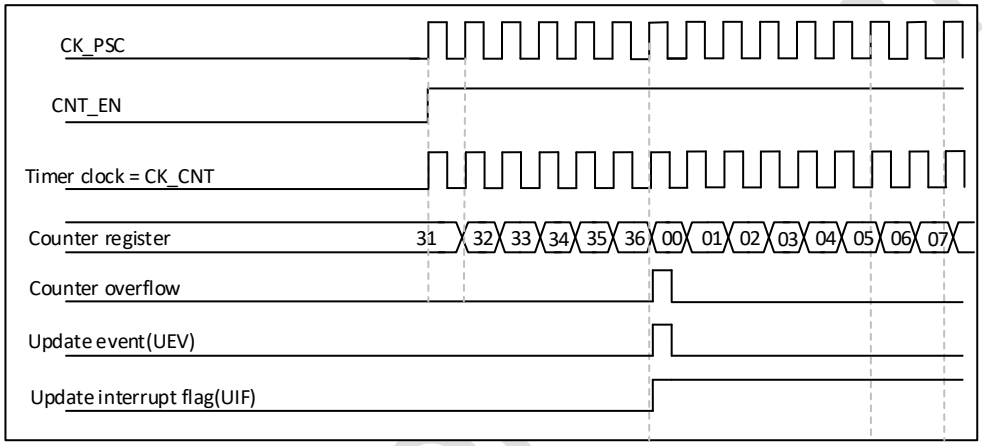


图 18-3 计数器时序图，内部时钟分频因子为 1

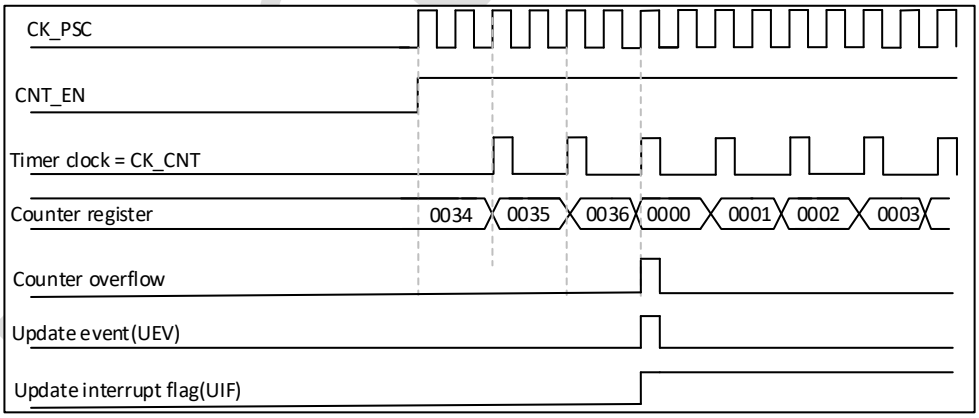


图 18-4 计数器时序图，内部时钟分频因子为 2

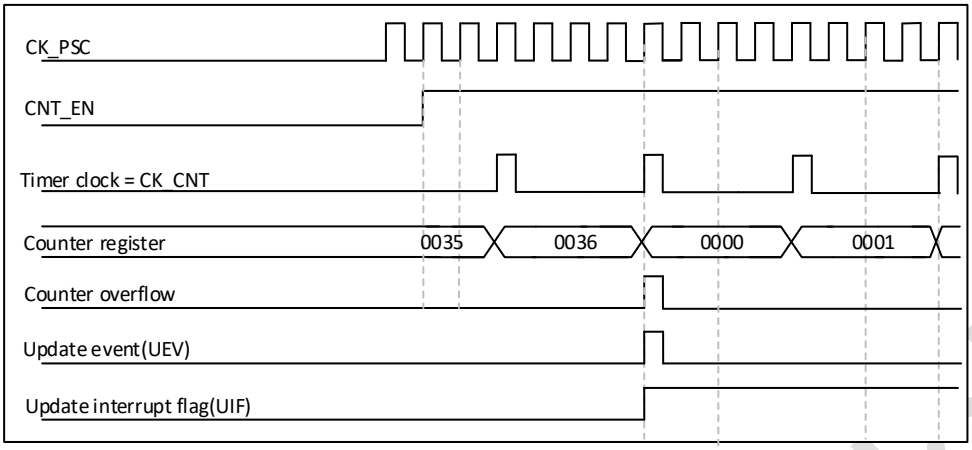


图 18-5 计数器时序图，内部时钟分频因子为 4

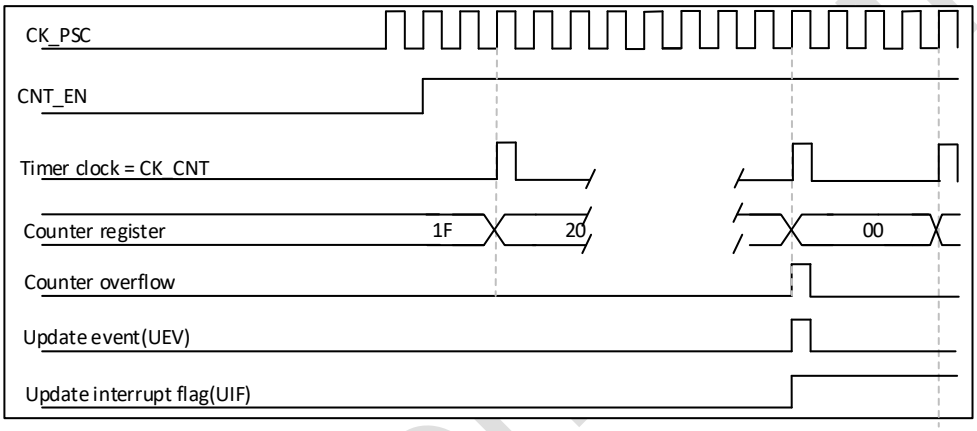


图 18-6 计数器时序图，内部时钟分频因子为 N

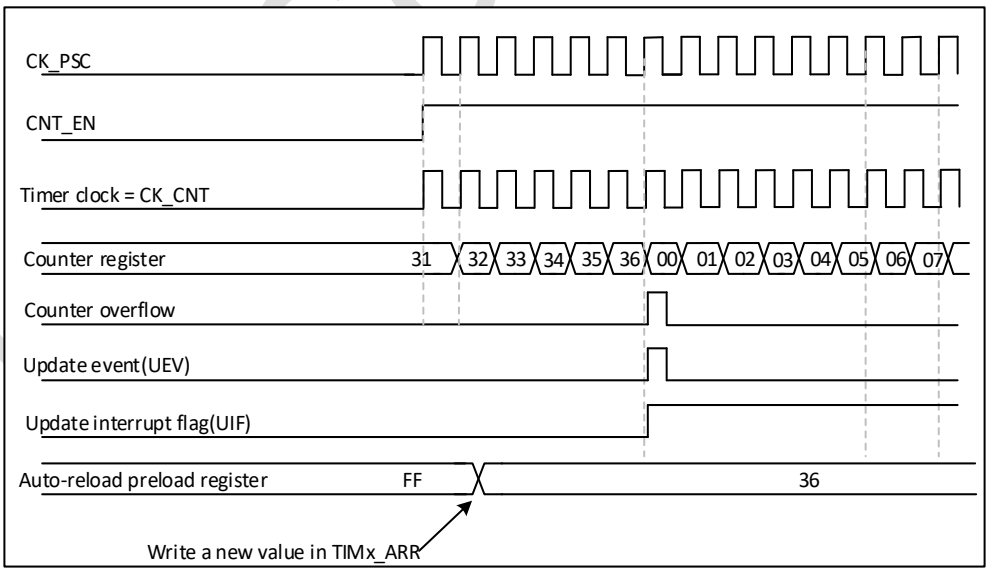


图 18-7 计数器时序图，当 ARPE=0 时的更新事件(没有预装 TIMx_ARR)

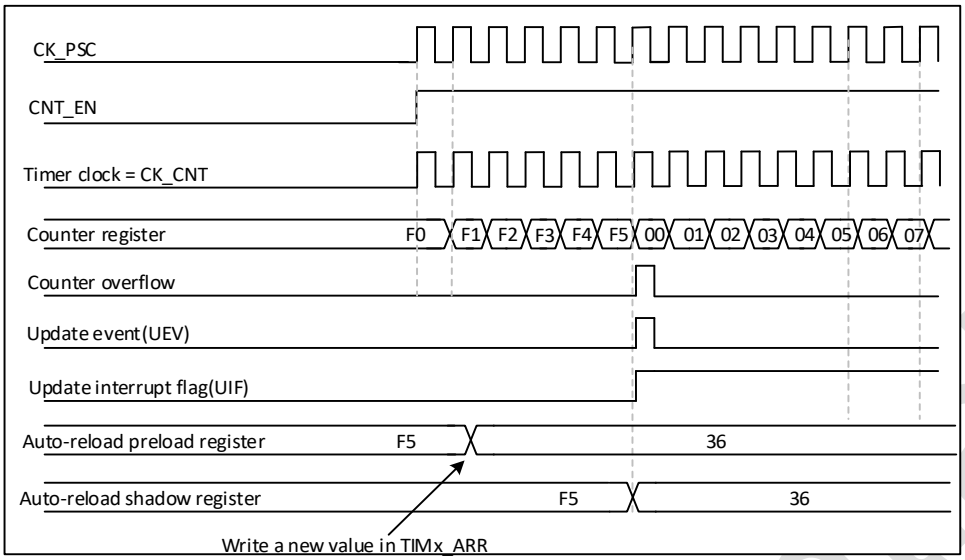


图 18-8 计数器时序图，当 ARPE=1 时的更新事件(预装了 TIMx_ARR)

向下计数模式

在向下计数模式中，计数器从自动加载值(TIMx_ARR 的内容)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数下溢事件。而在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会从当前自动加载值重新开始计数，同时预分频器内部的计数器被清'0'(但预分频系数不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源) ，通过设置 UG 位可以产生一个更新事件 UEV，但不置起 UIF 标志位(即不会产生中断或 DMA 请求)，这是为了避免在发生捕获事件时清除计数器，同时产生更新和捕获中断。

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

- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动加载值在计数器重载入之前被更新，因此下一个周期将是预期的值。

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

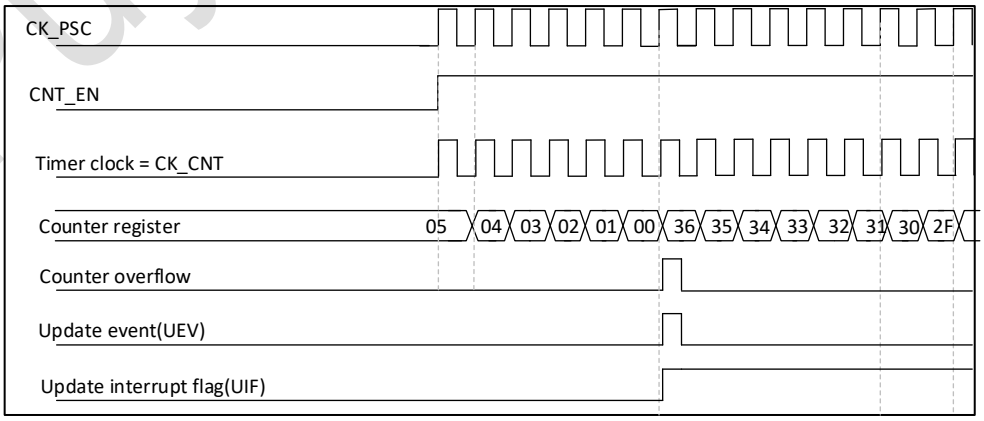


图 18-9 计数器时序图，内部时钟分频因子为 1

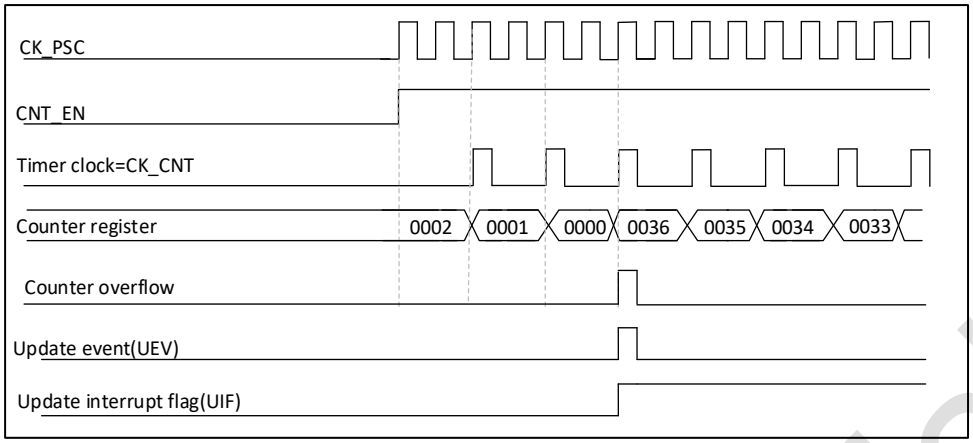


图 18-10 计数器时序图，内部时钟分频因子为 2

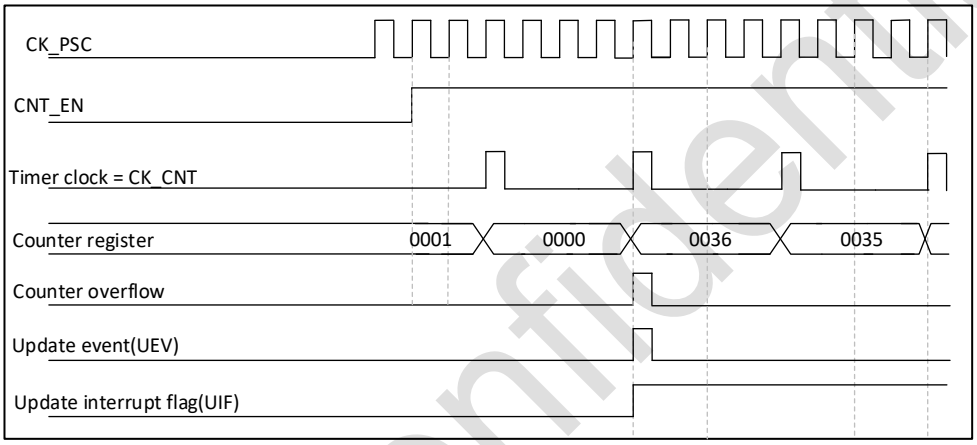


图 18-11 计数器时序图，内部时钟分频因子为 4

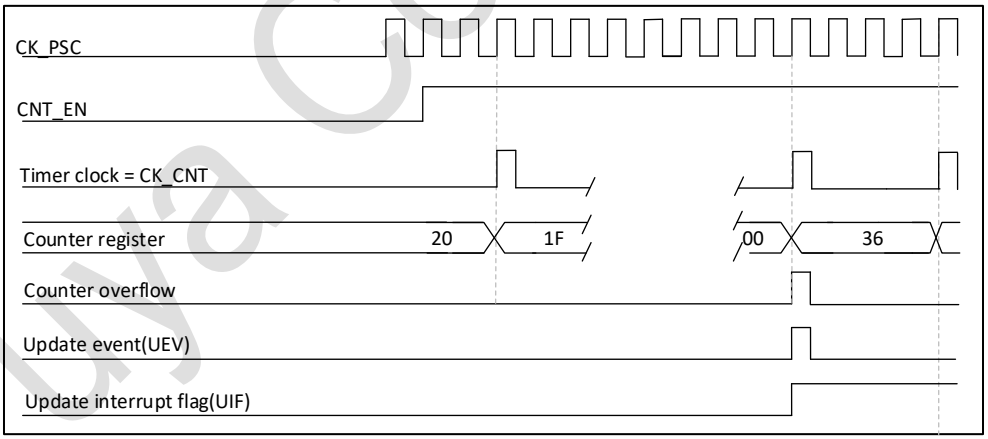


图 18-12 计数器时序图，内部时钟分频因子为 N

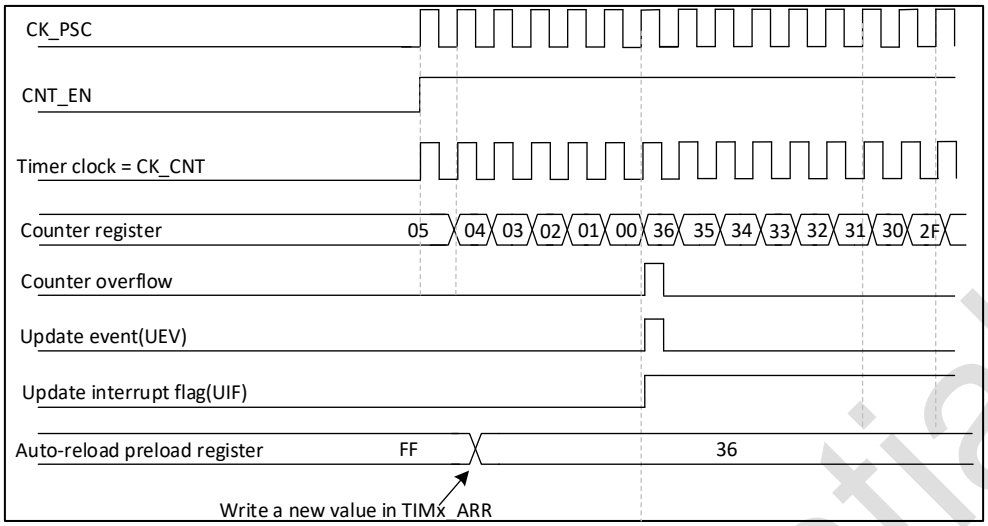


图 18-13 计数器时序图，当没有使用重复计数器时的更新事件

中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)减 1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

通过配置 TIMx_CR1 寄存器中的 CMS 位不为‘00’可以得到中央对齐模式。在通道配置为输出模式下的输出比较标志位在以下几种计数过程中会被置起：向下计数时（中央对齐模式 1，CMS=‘01’）、向上计数时（中央对齐模式 2，CMS=‘10’）、向上和向下计数时（中央对齐模式 3，CMS=‘11’）。

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

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

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

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

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

- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。
- 当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)。

以下是一些计数器在不同时钟频率下的操作的例子：

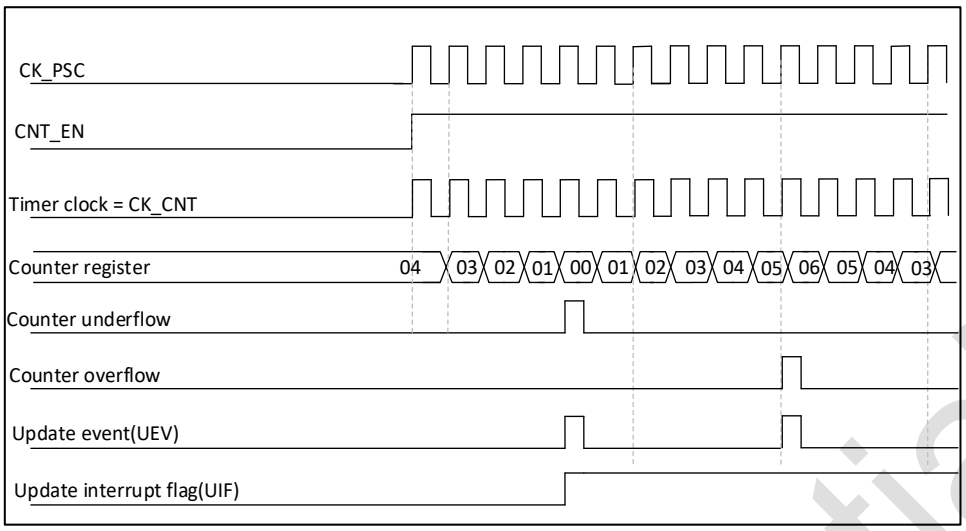


图 18-14 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

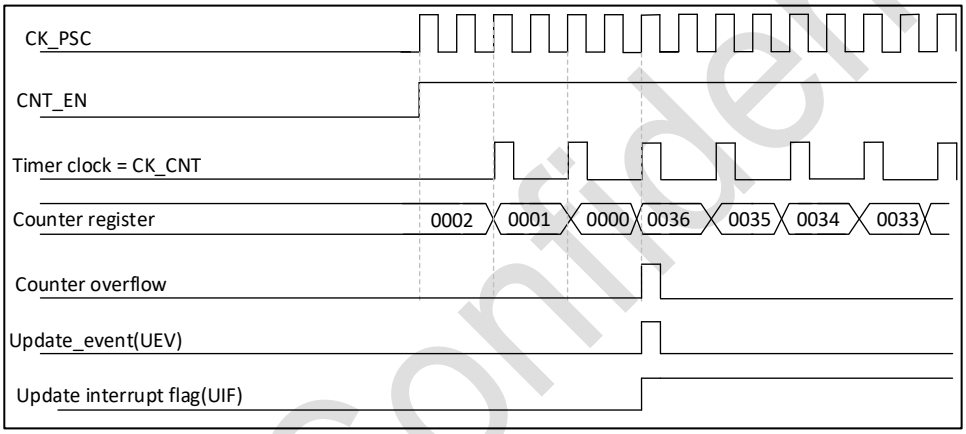


图 18-15 计数器时序图，内部时钟分频因子为 2

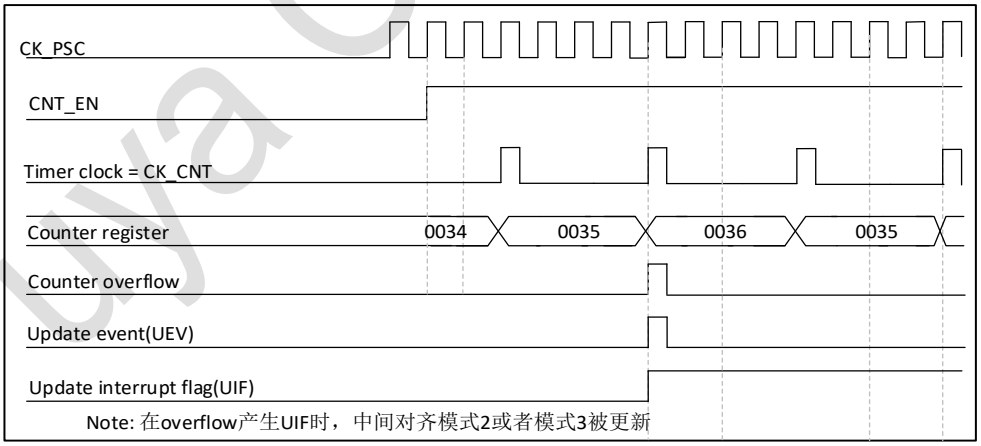


图 18-16 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36

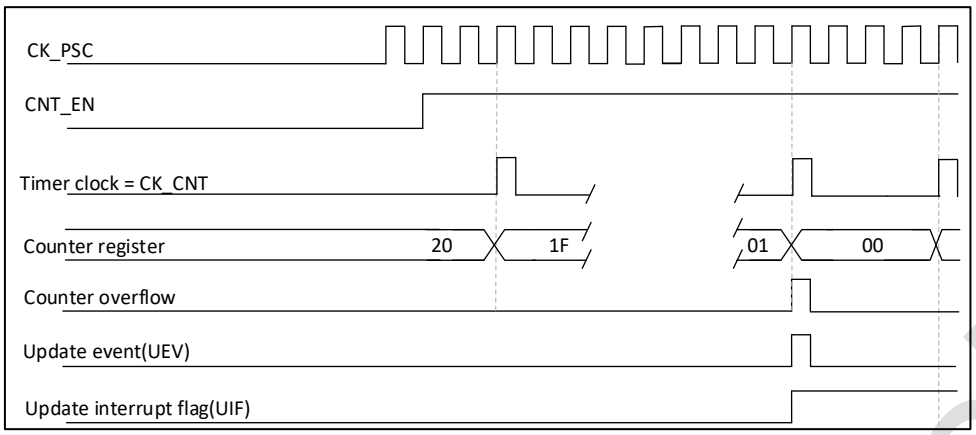


图 18-17 计数器时序图，内部时钟分频因子为 N

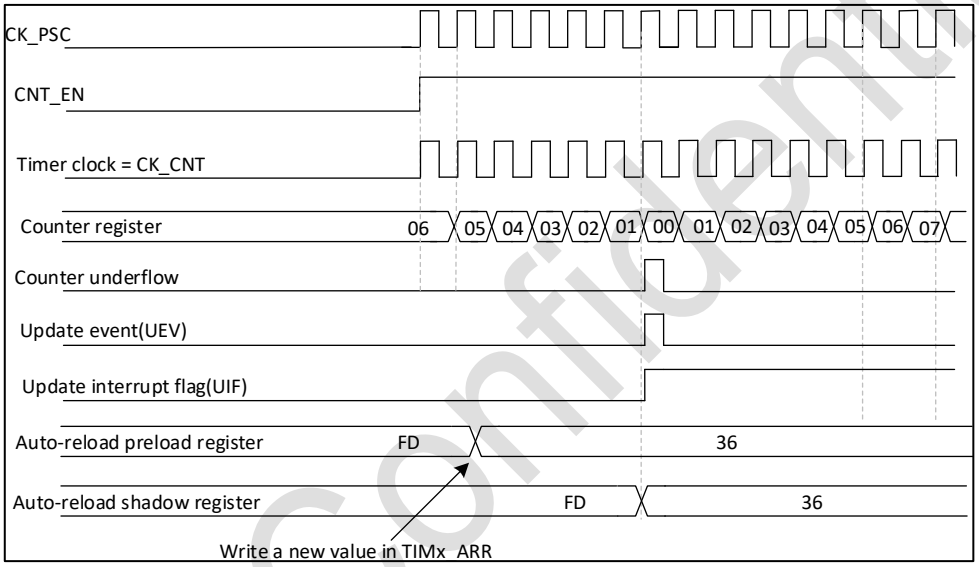


图 18-18 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

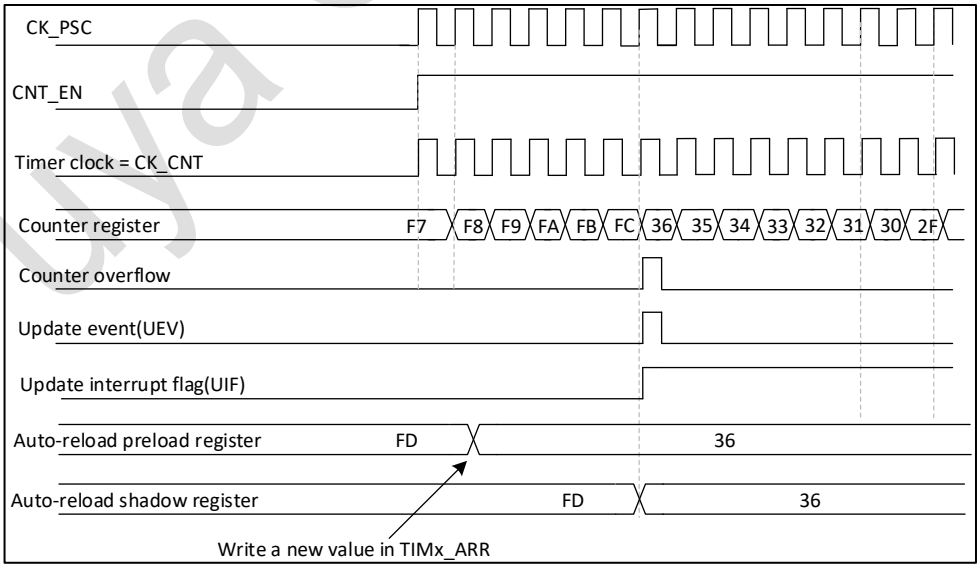


图 18-19 计数器时序图，ARPE=1 时的更新事件(计数器溢出)

18.3.3 外部触发输入

定时器外部触发输入 tim_etr，可用于：

- 外部时钟
- 从模式的触发
- 在周期电流调节中作为 PWM 复位输入

18.3.4 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚(TI1 和 TI2)
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。
- 编码器模式

内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=0000)，则 CEN、DIR(TIMx_CR1 寄存器)和 UG 位(TIMx_EGR 寄存器)是事实上的控制位，并且只能被软件修改(UG 位仍被自动清除)。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

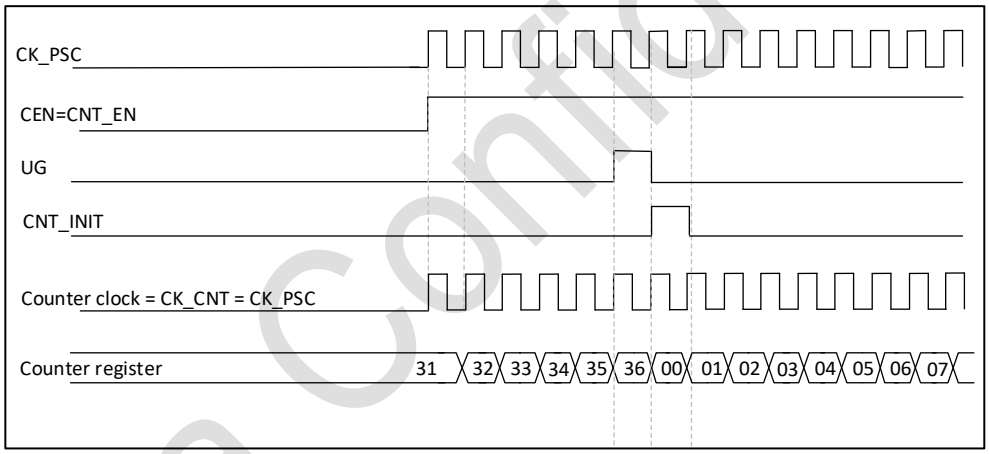


图 18-20 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

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

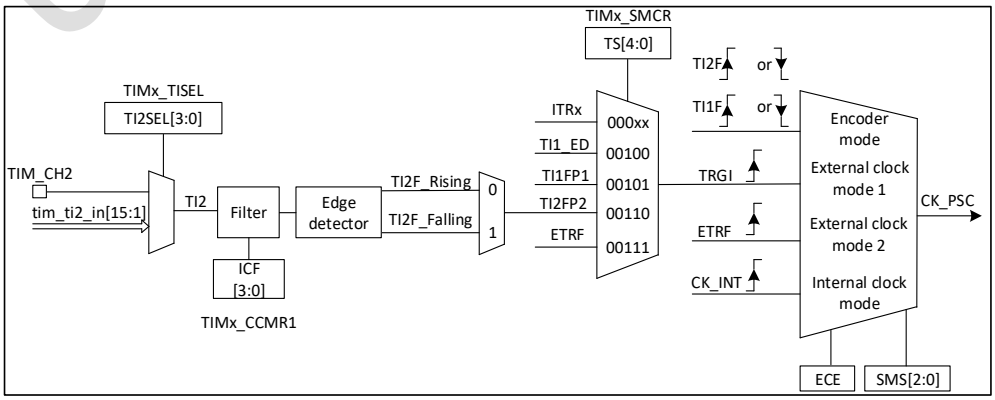


图 18-21 TI2 外部时钟连接例子

例如，要配置计数器在 TI2 输入端的上升沿向上计数，使用下列步骤：

1.配置 TIMx_CCMR1 寄存器 CC2S=01，使得通道 2 检测 TI2 输入端的上升沿；

2.配置 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽 (如果不需要滤波器，保持 IC2F=0000)；

3.配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；

4.配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器为外部时钟模式 1；

5.配置 TIMx_SMCR 寄存器中的 TS=00110，选定 TI2 作为触发输入源；

6.设置 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

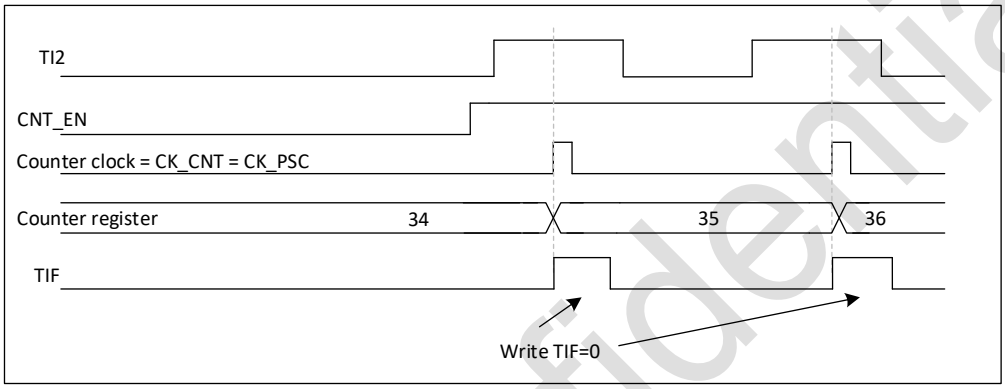


图 18-22 外部时钟模式 1 下的控制电路

外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1，计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。

下图是外部触发输入的框图：

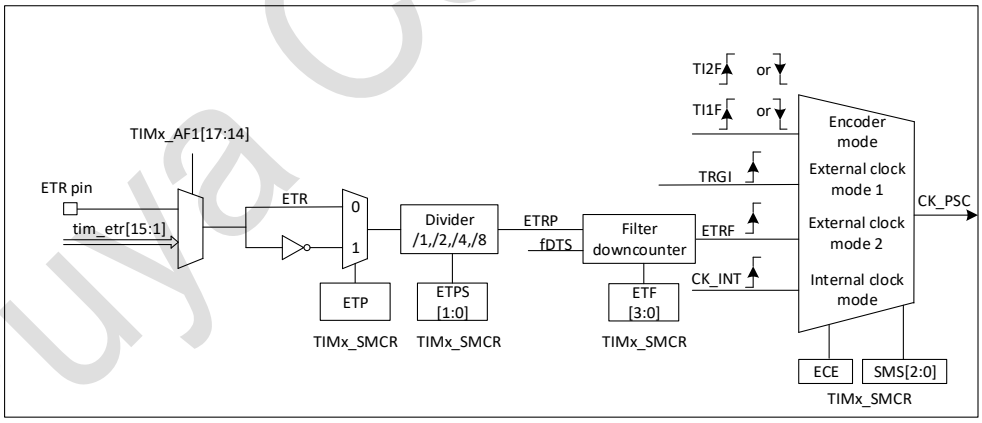


图 18-23 外部触发输入框图

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

1.本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000；

2.设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0]=01；

3.选择 ETR 输入端的上升沿，置 TIMx_SMCR 寄存器中的 ETP=0；

4.开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1；

5.启动计数器，写 TIMx_CR1 寄存器中的 CEN=1；

计数器在每 2 个 ETR 上升沿计数一次。

在 ETR 的上升沿和计数器实际时钟之间的延时取决于 ETRP 信号的同步电路。

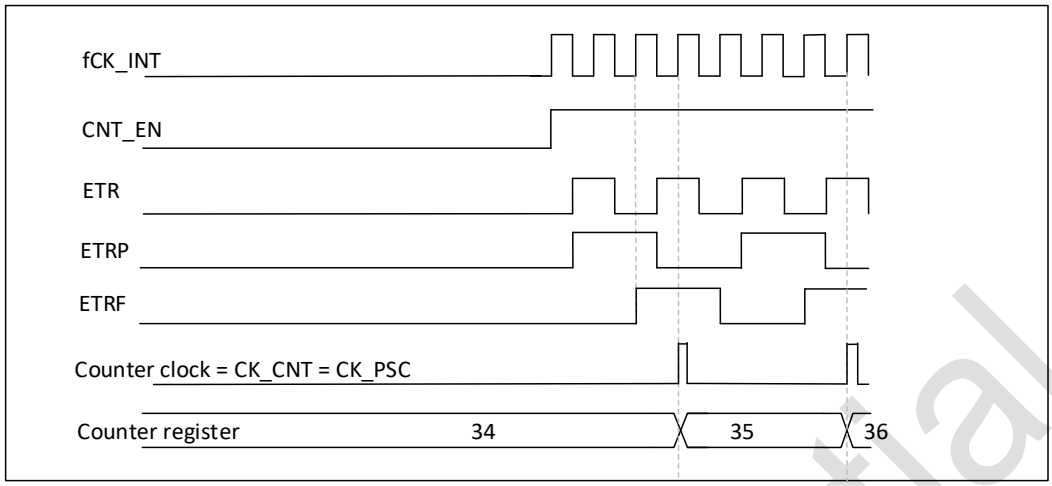


图 18-24 外部时钟模式 2 下的控制电路

18.3.5 捕获/比较通道

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

下面几张图是捕获/比较通道概览。

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

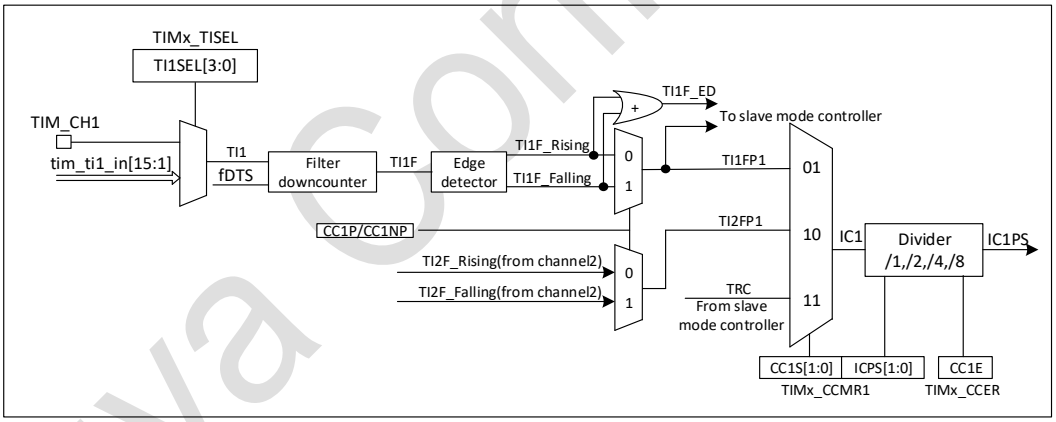


图 18-25 捕获/比较通道(如：通道 1 输入部分)

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

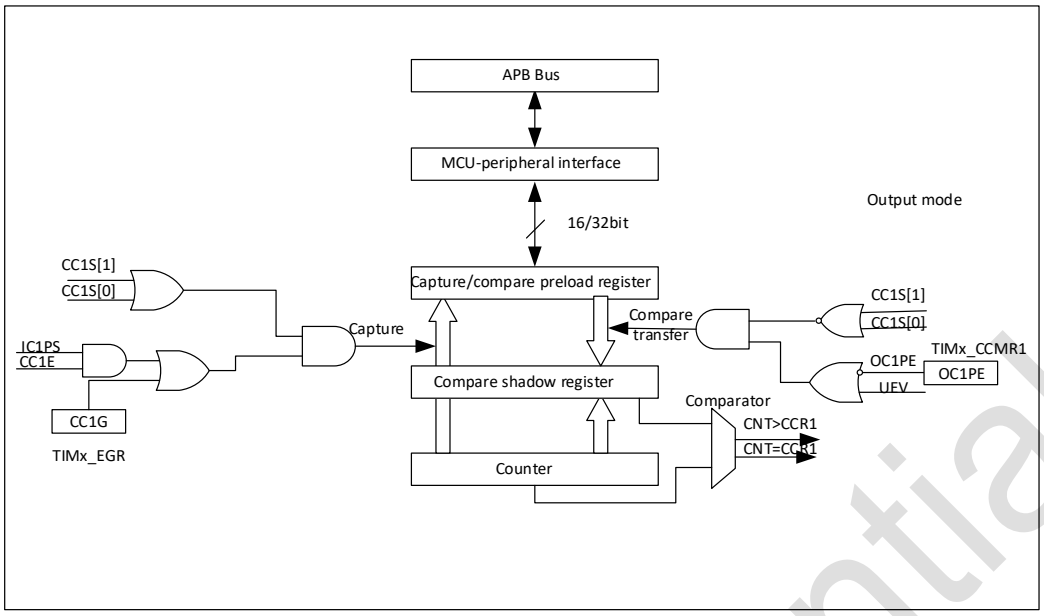


图 18-26 捕获/比较通道 1 的主电路

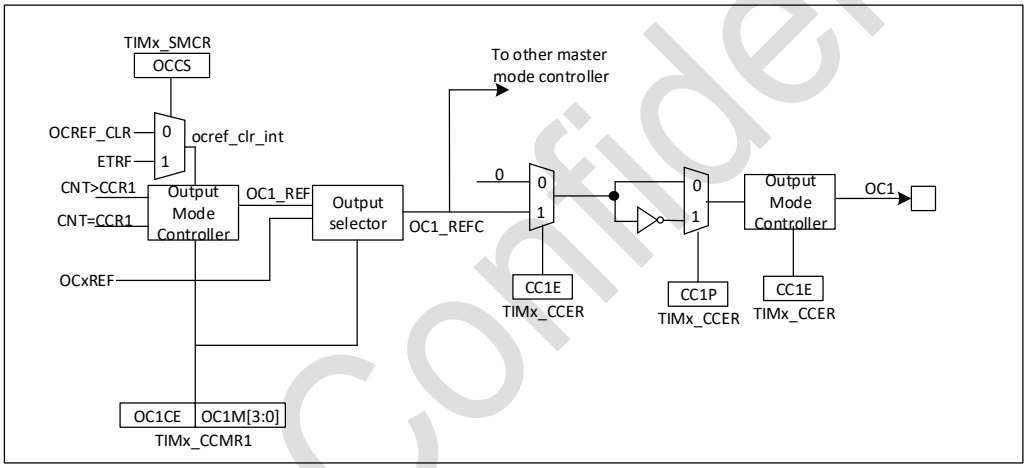


图 18-27 捕获/比较通道的输出部分(通道 1 至 4)

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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

18.3.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果使能中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么过捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。通过写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端： TIMx_CCMR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为‘00’，通道就被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1

上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(设置为上升沿)。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，可以通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，也可通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

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

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

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

18.3.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，其余操作与输入捕获模式相同：

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

例如，用户可以测量输入到 TI1 上的 PWM 信号的周期(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下

- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

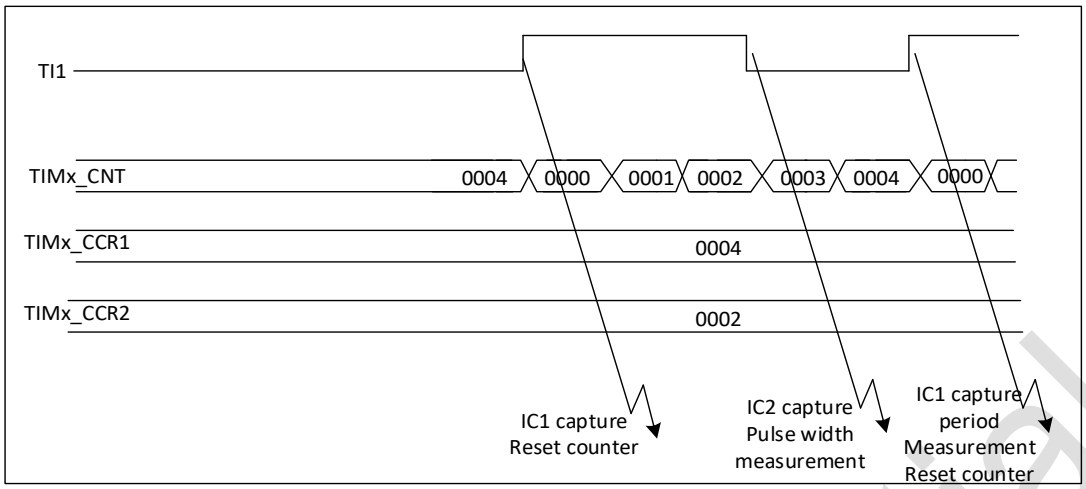


图 18-28 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器,所以 PWM 输入模式只能使用 TIMx_CH1/TIMx_CH2 信号。

18.3.8 强置输出模式

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

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如： CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

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

18.3.9 输出比较模式

此项功能是用来控制一个输出波形，或者表明一段给定的的时间已经到时。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作:

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

可以通过配置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

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

输出比较模式的配置步骤:

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。

4. 选择输出模式，例如：
   1) 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   2) 置 OCxPE = 0 禁用预装载寄存器
   3) 置 CCxP = 0 选择极性为高电平有效
   4) 置 CCxE = 1 使能输出
5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器

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

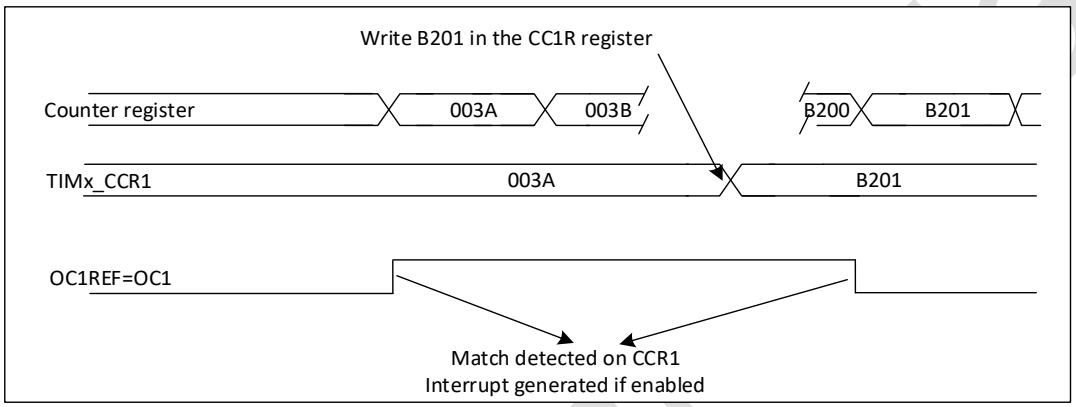


图 18-29 输出比较模式，翻转 OC1

18.3.10 PWM 模式

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

在 TIMx_CCMRx 寄存器中的 OCxM 位写入'110' (PWM 模式 1)或'111'(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位， (在向上计数或中心对称模式中)使能自动重装载的预装载寄存器。

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

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER)CCxE、CCxNE 控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

■ 向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

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

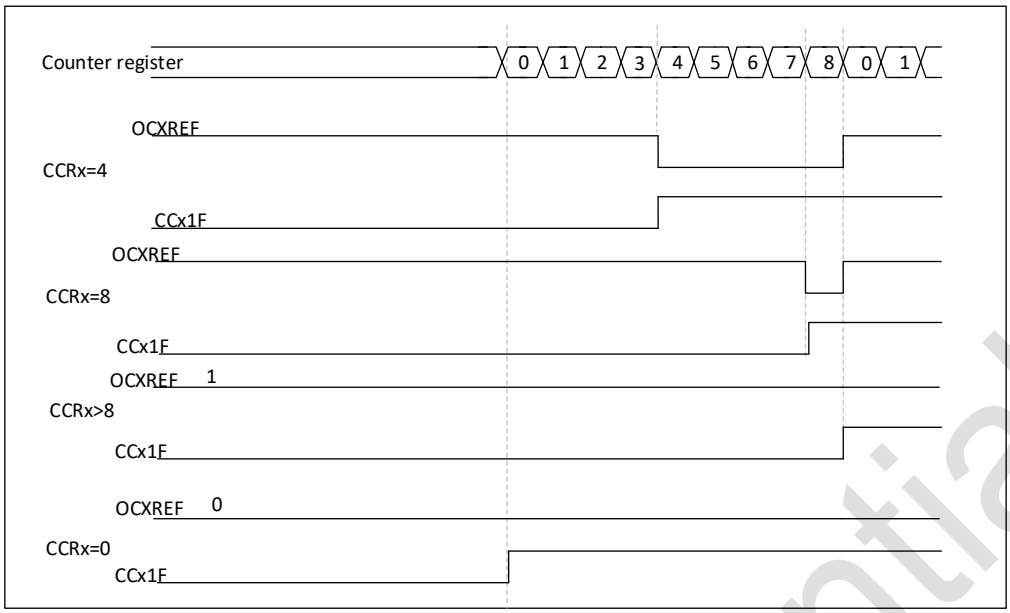


图 18-30 边沿对齐的 PWM 波形(ARR=8)

- 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

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

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为‘00’时为中央对齐模式(CMS 位的所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。 TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

下图给出了一些中央对齐的 PWM 波形的例子

- TIMx_ARR=8
- PWM 模式 1
- TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

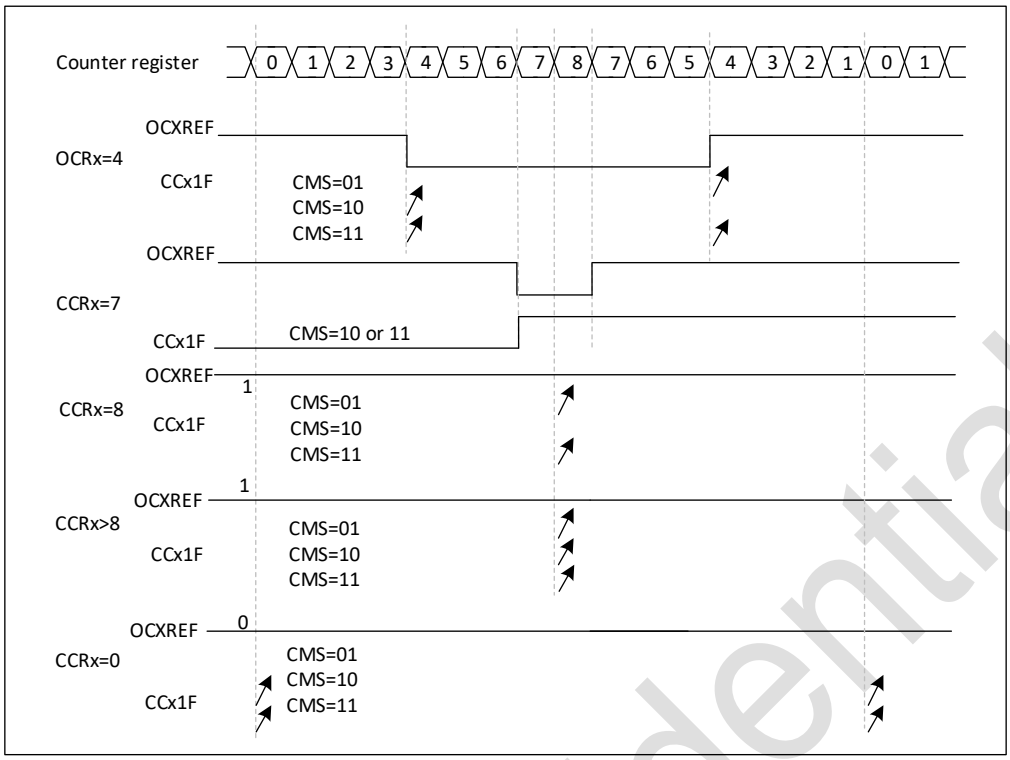


图 18-31 中央对齐的 PWM 波形(APR=8)

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，不要通过软件同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不会产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

18.3.11 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为‘1’，能够用 OCREF_CLR_INT 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次计数溢出所产生的更新事件 UEV。该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。

而 OCREF_CLR_INT 可以通过配置 TIMx_SMCR 寄存器中的 OCCS 位，在 OCREF_CLR 和 ETRF(ETR 滤波后)之间选择。

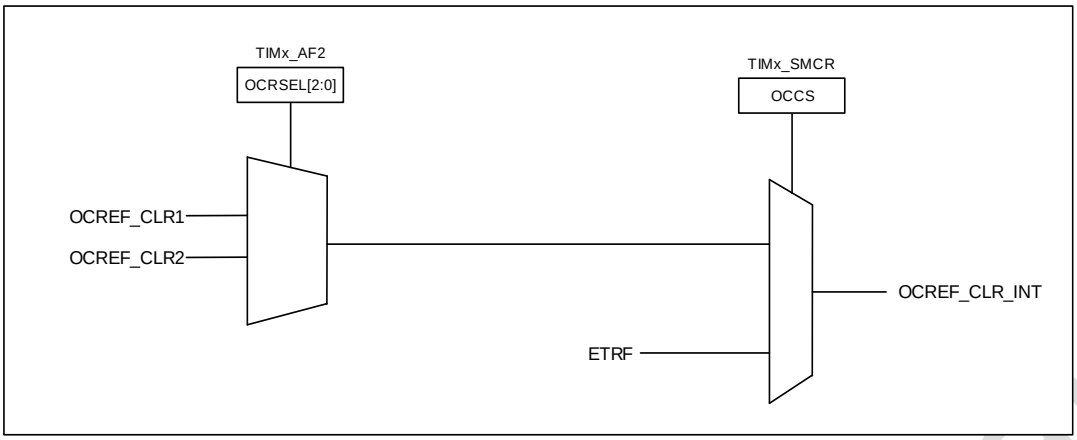


图 18-32 OCREF_CLR 输入选择

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

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

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值， OCxREF 信号的动作。在这个例子中，定时器 TIMx 被置于 PWM1 模式。

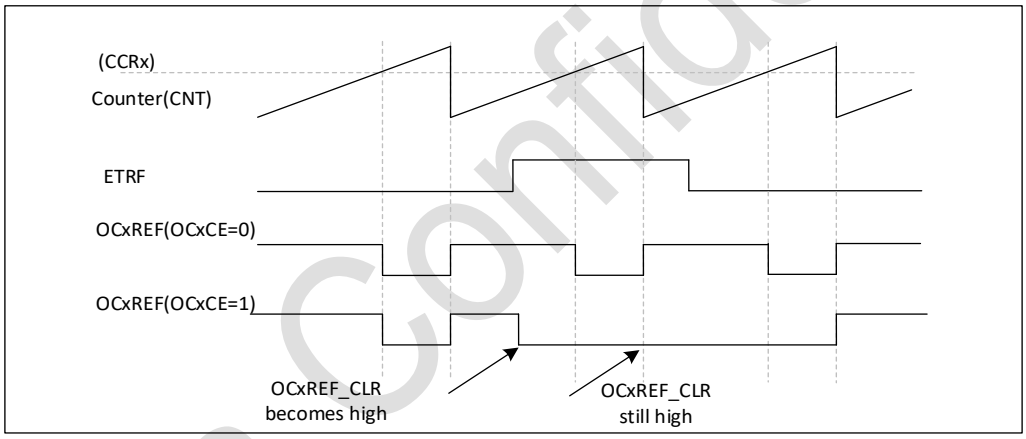


图 18-33 清除 TIMx 的 OCxREF

18.3.12 单脉冲模式

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

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

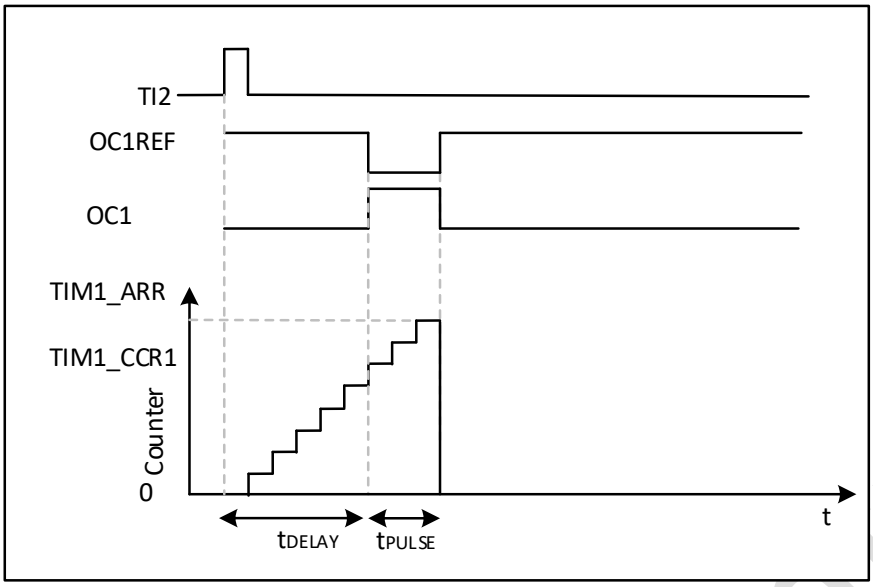


图 18-34 单脉冲模式的例子

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

假定 TI2FP2 作为触发:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110， TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)， TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中， CC1P=0。

在这个例子中， TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

特殊情况： OCx 快速使能：

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

18.3.13 编码器接口模式

正交编码器

选择编码器接口模式的方法是：如果计数器只在 TI1 的边沿计数，则置 TIMx_SMCR 寄存器中的 SMS=0001；如果只在 TI2 边沿计数，则置 SMS=0010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=0011。

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

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看下表，假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 18-1 计数方向与编码器信号的关系（CC1P=CC2P=0）

有效沿	SMS[3:0]	反向信号电平 (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 信号		TI2FP2 信号	
			Rising	Falling	Rising	Falling
仅在 TI1 双沿计数 x2 模式	001	高	减	加	-	-
		低	加	减	-	-
仅在 TI2 双沿计数 x2 模式	010	高	-	-	加	减
		低	-	-	减	加
在 TI1 和 TI2 双沿计数 x4 模式	011	高	减	加	加	减
		低	加	减	减	加

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

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

- CC1S='01'(TIMx_CCMR1 寄存器，TI1FP1 映射到 IC1)
- CC2S='01'(TIMx_CCMR2 寄存器，TI2FP2 映射到 IC2)
- CC1P='0' (TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1)
- CC2P='0' (TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2=TI2)
- SMS='011'(TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效).
- CEN='1'(TIMx_CR1 寄存器，计数器使能)

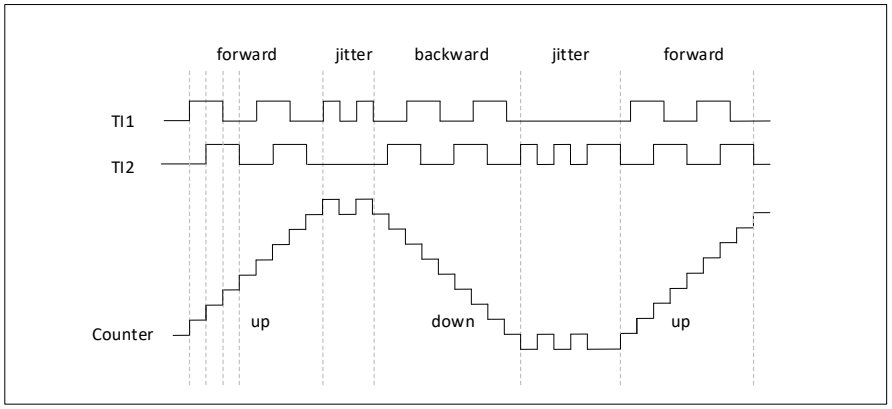


图 18-35 编码器模式下的计数器操作实例

下图为当 TI1FP1 极性反相时计数器的操作实例(CC1P='1'，其他配置与上例相同)

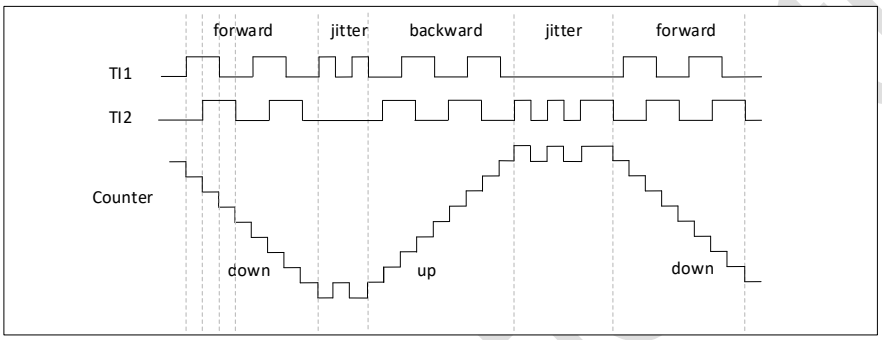


图 18-36 TI1FP1 反相的编码器接口模式实例

下图表示的是不同模式下当转向翻转时的计数值情况：

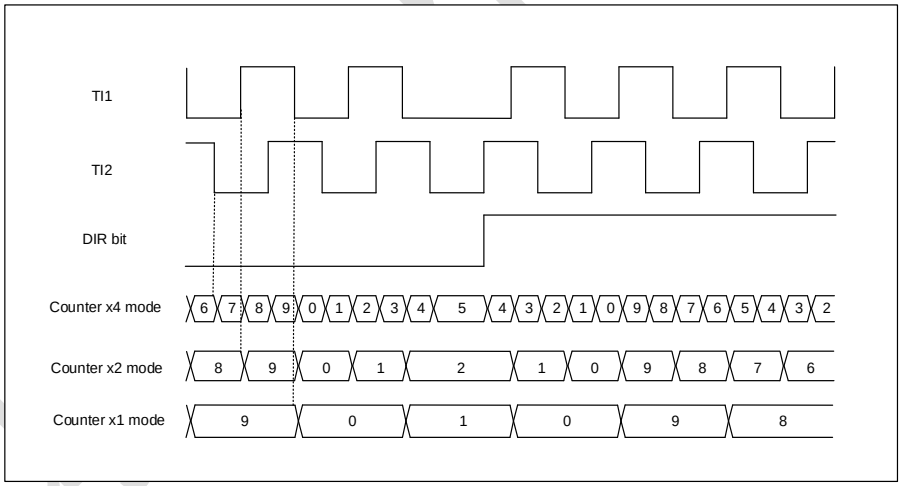


图 18-37 正交编码器计数模式

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

18.3.14 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、 TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

18.3.15 定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式、触发、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 UDIS 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， TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

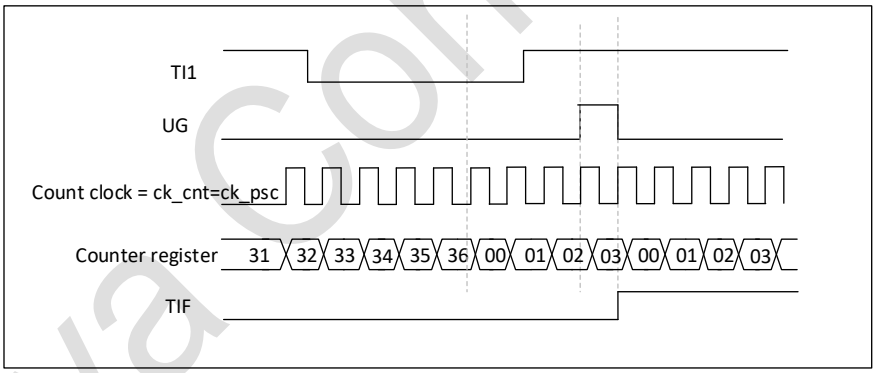


图 18-38 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 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。

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

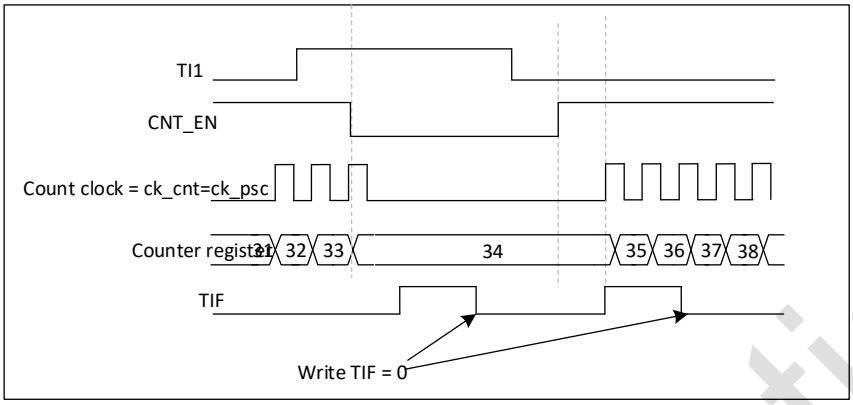


图 18-39 门控模式下的控制电路

从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

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

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

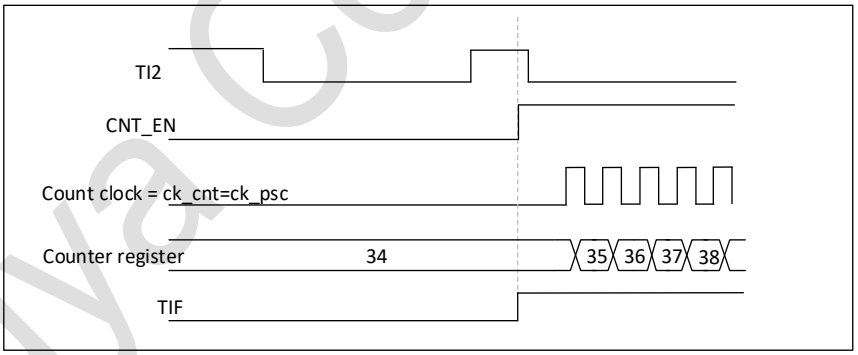


图 18-40 触发器模式下的控制电路

18.3.16 定时器同步

所有 TIMx 定时器在内部相连，用于定时器同步或链接。详见定时器同步章节。

18.3.17 DMA 突发模式

TIMx 定时器当发生单个事件时有生成多个 DMA 请求的能力。主要目的是能够多次重新编程定时器的部分功能而无需软件的开销。但是也可以在固定的间隔时间内读取一行中的几个寄存器。

DMA 控制器的目的地是唯一的并且必须指向虚拟寄存器 TIMx_DMAR.当给定的定时器事件发生，定时器发起一系列 DMA 请求。每个写入 TIMx_DMAR 寄存器的操作实际上是重定向一个定时器寄存器的。TIMx_DCR 寄存器中的 DBL[4:0]位设置 DMA 突发长度。当对 TIMx_DMR 地址进行读取或写入访问时，定时器会识别突发传输，即传输次数（以半个字或字节为单位）

TIMx_DCR 的 DBA[4:0]位定义了 DMA 传输的 DMA 基地址(当通过 TIMx_DMAR 地址完成读写操作)。DBA 定义为从 TIMx_CR1 寄存器地址开始的偏移量。

例子:

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

作为一个例子，定时器 DMA 突发特性是用于发生更新事件时更新 CCRx 寄存器中的内容，通过 DMA 传输半字到 CCRx。

通过以下步骤来完成:

1. 如下配置对应的 DMA 通道:

1) DMA 通道外设地址是 DMAR 寄存器的地址

2) DMA 通道存储地址是 RAM 中的 BUFF 地址，包含了 DMA 传给 CCRx 的数据

3) 传输的数据个数=3

4) 轮询模式关闭

2. 通过配置 DBA 和 DBL 实现配置 DCR 寄存器：DBL=3，DBA=0XE

3. 使能定时器更新 DMA 请求（UDE=1）

4. 使能定时器

5. 使能 DMA 通道

这个例子此示例适用于每个 CCRx 寄存器更新一次的情况。如果每个 CCRx 都更新两次，数据的传输个数就需要设置为 6。举例假设 RAM 的 buffer 中包含了 data1, data2, data3, data4, data5 and data6。数据依次传输到 CCRx 中：第一次更新 DMA 请求，data1 传输给 CCR2，data2 传输给 CCR3，data3 传输给 CCR4;第二次更新 DMA 请求,data4 传输给 CCR2,data5 传输给 CCR3,data6 传输给 CCR4。

注：空值可以写入到保留寄存器。

18.3.18 TIM2/TIM3/TIM4 DMA 请求

TIM2/TIM3/TIM4 可以生成 DMA 请求，如下表所示:

表 18-2 DMA 请求

DMA 请求源	DMA 使能控制位
更新	UDE
比较/捕获 1	CC1DE
比较/捕获 2	CC2DE
比较/捕获 3	CC3DE
比较/捕获 4	CC4DE
触发	TDE

18.3.19 调试模式

当微控制器进入调试模式时，根据 DBG 模块中 DBG_TIMx_STOP 的设置，

TIMx 计数器可以或者继续正常操作，或者停止。详见随后的 DBG 节。

调试模式下的行为可以使用调试支持模块（DBG）中每个定时器的专用配置进行编程

为了安全的考虑，当计数器停止时，输出被禁止（如果 MOE 被复位了）。输出即可被强制到一个无效电平（OSSI=1），或者由 GPIO 控制器接管（OSSI=0），典型的是强制为高阻。

更多的细节，见 DBG 模块描述。

18.3.20 TIM2/TIM3/TIM4 低功耗模式

表 18-3 低功耗模式下的模块特性

模式	说明
Sleep	无效果，外围设备处于活动状态。中断可能导致设备退出睡眠状态
Stop	定时器操作停止，寄存器内容保留。不能产生中断

18.4 TIM2/TIM3/TIM4 中断

表 18-4 18.4 TIM2/TIM3/TIM4 中断请求

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
比较/捕获 3	CC3IF	CC3IE
比较/捕获 4	CC4IF	CC4IE
触发	TIF	TIE

18.5 TIM2/TIM3/TIM4 寄存器描述

18.5.1 TIMx 控制寄存器 1（TIMx_CR1）（x=2/3/4）

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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**
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD	RW	2'h0	时钟分频因子 (Clock division) 这 2 位定义在定时器时钟(CK_INT)频率和死区时间和死区发生器与数字滤波器(ETR,TIx)所用的采样时钟（t_{DTS}）之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 \times t_{CK\_INT}$ 10：$t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 (Auto-reload preload enable) 0： TIMx_ARR 寄存器没有缓冲 1： TIMx_ARR 寄存器被装入缓冲器
6:5	CMS	RW	2'h0	选择中央对齐模式 (Center-aligned mode selection) 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。

				10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 **注：**在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	方向 (Direction) 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	OPM	RW	0	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止
2	URS	RW	0	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1: 禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、 CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	使能计数器 (Counter enable) 0：禁止计数器 1：使能计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

18.5.2 TIMx 控制寄存器 2（TIMx_CR2） (x=2/3/4)

偏移地址：0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	MMS[3]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
						RW									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1S	MMS[2:0]			CCDS	Res.	Res.	Res.
								RW	RW	RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31:26	Reserved	-	-	保留
25	MMS[3]	RW	0	详见 MMS[2:0]描述
24:8	Reserved	-	-	保留
7	TI1S	RW	0	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、 TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
6:4	MMS	RW	3'h0	主模式选择 (Master mode selection) 用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 0000：复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 0001：使能 – 计数器使能信号 CNT_EN 可被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 0010: 更新 – 更新事件被选为触发输出(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 0011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 0100：比较 – OC1REFC 信号被用于作为触发输出(TRGO)。 0101：比较 – OC2REFC 信号被用于作为触发输出(TRGO)。 0110：比较 – OC3REFC 信号被用于作为触发输出(TRGO)。 0111：比较 – OC4REFC 信号被用于作为触发输出(TRGO)。 1000：编码器时钟输出-编码器时钟信号作为触发输出（TRGO）。 1001-1111：保留 注意： 1.从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。 2.若主从定时器不在同一总线上，主模式应该配置为能被从定时器采到的宽度。
3	CCDS	RW	0	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求；

				1：当发生更新事件时，送出 CCx 的 DMA 请求。
2:0	Reserved	-	-	保留

18.5.3 TIMx 从模式控制寄存器（TIMx_SMCR）（x=2/3/4）

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

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[4:3]	RW	2’h0	详见 TS[2:0]描述
19:16	Reserved	-	-	Reserved
15	ETP	RW	0	外部触发极性 (External trigger polarity) 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
14	ECE	RW	0	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1: 设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=00111)具有相同功效。 注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是‘00111’)。 注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
13:12	ETPS	RW	2’h0	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
11:8	ETF	RW	4’h0	外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6

				0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
7	MSM	RW	0	主/从模式 (Master/slave mode) 0：无作用； 1: 触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS	RW	3’h0	触发选择 (Trigger selection) 选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) (TIM3/TIM4 支持) 00010：内部触发 2(ITR2) (TIM2/TIM4 支持) 00011：内部触发 3(ITR3) (TIM2/TIM3 支持) 00100：TI1 边沿检测（TI1F_ED） 00101：滤波后的定时器输入 1（TI1FP1） 00110：滤波后的定时器输入 2（TI1FP2） 00111：外部触发输入（ETRF） 01000：内部触发 4(ITR4) 01001：内部触发 5(ITR5) 01010：内部触发 6(ITR6) 其它：保留 更多有关 ITRx 的细节，参见系统级联表 注：这些位只能在未用到(如 SMS=0000)时被改变，以避免在改变时产生错误的边沿检测。
3	OCCS	RW	0	OCREF 清除选择位（OCREF clear selection） 这位选择了 OCREF 的清除源。 0：OCREF_CLR_INT 连接 OCREF_CLR 输入 1：OCREF_CLR_INT 连接 ETRF
2:0	SMS	RW	3’h0	从模式选择 (Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001: 正交编码器模式 1，x2 模式– 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。

				010: 正交编码器模式 2，x2 模式– 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 011：正交编码器模式 3，x4 模式–根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_ED 被选为触发输入(TS=00100)时，不要使用门控模式。这是因为， TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。 注：在编码器模式下，不要使用 UEV 作为 tim_trgo 输出信号，（即 mms 不能配置为 010。

18.5.4 TIMx DMA/中断使能寄存器（TIMx_DIER）（x=2/3/4）

偏移地址：0x0C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDE	Res.	CC4DE	CC3DE	CC2DE	CC1DE	UDE	Res.	TIE	Res.	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	RW		RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	TDE	RW	0	允许触发 DMA 请求 (Trigger DMA request enable) 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	Reserved	-	-	保留
12	CC4DE	RW	0	允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0：禁止捕获/比较 4 的 DMA 请求 1：允许捕获/比较 4 的 DMA 请求
11	CC3DE	RW	0	允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0：禁止捕获/比较 3 的 DMA 请求 1：允许捕获/比较 3 的 DMA 请求。。
10	CC2DE	RW	0	允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0：禁止捕获/比较 2 的 DMA 请求 1：允许捕获/比较 2 的 DMA 请求

9	CC1DE	RW	0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	允许更新的 DMA 请求 (Update DMA request enable) 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	Reserved	-	-	保留
6	TIE	RW	0	触发中断使能 (Trigger interrupt enable) 0：禁止触发中断 1：使能触发中断
5	Reserved	-	-	保留
4	CC4IE	RW	0	允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	RW	0	允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	RW	0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	允许更新中断 (Update interrupt enable) 0：禁止更新中断 1：允许更新中断

18.5.5 TIMx 状态寄存器（TIMx_SR）（x=2/3/4）

偏移地址：0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	IC4IF	IC3IF	IC2IF	IC1IF	IC4IR	IC3IR	Res.	Res.	Res.	Res.	IC2IR	IC1IR	Res.	Res.
		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0					RC_W0	RC_W0		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	CC4OF	CC3OF	CC2OF	CC1OF	Res.	Res.	TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			RC_W0	RC_W0	RC_W0	RC_W0			RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:30	Reserved	-	-	保留
29	IC4IF	RC_W0	0	下降沿捕获 4 标志 参见 IC1IF 描述。
28	IC3IF	RC_W0	0	下降沿捕获 3 标志

				参见 IC1IF 描述。
27	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
26	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生下降沿捕获事件。
25	IC4IR	RC_W0	0	上升沿捕获 4 标志 参见 IC1IR 描述。
24	IC3IR	RC_W0	0	上升沿捕获 3 标志 参见 IC1IR 描述。
23:20	Reserved	-	-	保留
19	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
18	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生； 1：发生上升沿捕获事件。
17:13	Reserved	-	-	保留
12	CC4OF	RC_W0	0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。
11	CC3OF	RC_W0	0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。
10	CC2OF	RC_W0	0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1: 计数器的值被捕获到 TIMx_CCR1 寄存器时， CC1IF 的状态已经为‘1’。
8:7	Reserved	-	-	保留
6	TIF	RC_W0	0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置‘1’。它由软件清‘0’。 0：无触发器事件产生； 1：触发中断等待响应。
5	Reserved	-	-	保留
4	CC4IF	RC_W0	0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag)

				参考 CC1IF 描述。
3	CC3IF	RC_W0	0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
2	CC2IF	RC_W0	0	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式: 0：无匹配发生; 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下， CC1IF 位变高 如果通道 TI1 配置为输入模式: 当捕获事件发生时该位由硬件置‘1’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生; 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
0	UIF	RC_W0	0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生; 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’: − 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当计数器 CNT 被触发事件重新初始化时。(参考从模式控制寄存器(TIMx_SMCR))。

18.5.6 TIMx 事件产生寄存器（TIMx_EGR）（x=2/3/4）

偏移地址：0x14

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	TG	W	0	产生触发事件 (Trigger generation) 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作; 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	Reserved	-	-	保留

4	CC4G	W	0	产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 CC1G 描述。
3	CC3G	W	0	产生捕获/比较 3 事件 (Capture/Compare 3 generation) 参考 CC1G 描述。
2	CC2G	W	0	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。
1	CC1G	W	0	产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1'，用于产生一个捕获/比较事件，由硬件自动清'0'。 0：无动作； 1：在通道 1 上产生一个捕获/比较事件： 若通道 1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 (Update generation) 该位由软件置'1'，由硬件自动清'0'。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清'0'(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

18.5.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1）（x=2/3/4）

偏移地址：0x18

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				ICIF[3:0]				ICIPSC[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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	OC2CE	RW	0	输出比较 2 清零使能 (Output Compare 2 clear enable)
14:12	OC2M	RW	3'h0	输出比较 2 模式 (Output Compare 2 mode)
11	OC2PE	RW	0	输出比较 2 预装载使能 (Output Compare 2 preload enable)
10	OC2FE	RW	0	输出比较 2 快速使能 (Output Compare 2 fast enable)
9:8	CC2S	RW	2'h0	捕获/比较 2 选择。 (Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入， IC2 映射在 TI2 上； 10：CC2 通道被配置为输入， IC2 映射在 TI1 上；

				11：CC2 通道被配置为输入， IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	OC1CE	RW	0	输出比较 1 清'0'使能 (Output Compare 1 clear enable) 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
6:4	OC1M	RW	3'h0	输出比较 1 模式 (Output Compare 1 mode) 这些位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。
3	OC1PE	RW	0	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。
2	OC1FE	RW	0	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快输出对触发输入事件的响应。必须在单脉冲模式（TIMx_CR1 寄存器的 OPM 位置 1）中使用，当触发到来时实现脉冲快速输出。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。仅能用于 PWM 模式 1 和 PWM 模式 2

1:0	CC1S	RW	2'h0	捕获/比较 1 选择。 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F	RW	4'h0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	RW	2'h0	输入/捕获 2 预分频器 (Input capture 2 prescaler)
9:8	CC2S	RW	2'h0	捕获/比较 2 选择 (Capture/Compare 2 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC2 通道被配置为输出； 01： CC2 通道被配置为输入， IC2 映射在 TI2 上； 10： CC2 通道被配置为输入， IC2 映射在 TI1 上； 11： CC2 通道被配置为输入， IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F	RW	4'h0	输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC	RW	2'h0	输入/捕获 1 预分频器 (Input capture 1 prescaler)

				这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1S=00，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	2'h0	捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

18.5.8 TIMx 捕获/比较模式控制寄存器 2（TIMx_CCMR2）（x=2/3/4）

偏移地址：0x1C

复位值： 0x0000 0000

参看以上 CCMR1 寄存器的描述

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]		OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]				IC3PSC[1:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

■ 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	OC4CE	RW	0	输出比较 4 清零使能 (Output Compare 4 clear enable)
14:12	OC4M	RW	3'h0	输出比较 4 模式 (Output Compare 4 mode)
11	OC4PE	RW	0	输出比较 4 预装载使能 (Output Compare 4 preload enable)
10	OC4FE	RW	0	输出比较 4 快速使能 (Output Compare 4 fast enable)
9:8	CC4S	RW	2'h0	捕获/比较 4 选择。 (Capture/Compare 4 selection) 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC4 通道被配置为输出； 01： CC4 通道被配置为输入， IC4 映射在 TI4 上； 10： CC4 通道被配置为输入， IC4 映射在 TI3 上； 11： CC4 通道被配置为输入， IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。

7	OC3CE	RW	0	输出比较 3 清'0'使能 (Output Compare 1clear enable)
6:4	OC3M	RW	3'h0	输出比较 3 模式
3	OC3PE	RW	0	输出比较 3 预装载使能 (Output Compare 3 preload enable)
2	OC3FE	RW	0	输出比较 3 快速使能 (Output Compare 3 fast enable)
1:0	CC3S	RW	2'h0	捕获/比较 3 选择。 (Capture/Compare 3 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC3 通道被配置为输出； 01： CC3 通道被配置为输入， IC3 映射在 TI3 上； 10： CC3 通道被配置为输入， IC3 映射在 TI4 上； 11： CC3 通道被配置为输入， IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC4F	RW	4'h0	输入捕获 4 滤波器 (Input capture 4 filter)
11:10	IC4PSC	RW	2'h0	输入/捕获 4 预分频器 (Input capture 4 prescaler)
9:8	CC4S	RW	2'h0	捕获/比较 4 选择 (Capture/Compare 4 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC4 通道被配置为输出； 01： CC4 通道被配置为输入， IC4 映射在 TI4 上； 10： CC4 通道被配置为输入， IC4 映射在 TI3 上； 11： CC4 通道被配置为输入， IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
7:4	IC3F	RW	4'h0	输入捕获 3 滤波器 (Input capture 3 filter)
3:2	IC3PSC	RW	2'h0	输入/捕获 3 预分频器 (Input capture 3 prescaler)
1:0	CC3S	RW	2'h0	捕获/比较 3 选择 (Capture/Compare 3 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC3 通道被配置为输出； 01： CC3 通道被配置为输入， IC3 映射在 TI3 上； 10： CC3 通道被配置为输入， IC3 映射在 TI4 上； 11： CC3 通道被配置为输入， IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

18.5.9 TIMx 捕获/比较使能寄存器（TIMx_CCER）（x=2/3/4）

偏移地址：0x20

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	CC4P	CC4E	Res.	Res.	CC3P	CC3E	Res.	Res.	CC2P	CC2E	Res.	Res.	CC1P	CC1E
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	CC4P	RW	0	输入/捕获 4 输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。
12	CC4E	RW	0	输入/捕获 4 输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。
11:10	Reserved	-	-	保留
9	CC3P	RW	0	输入/捕获 3 输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。
8	CC3E	RW	0	输入/捕获 3 输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。
7:6	Reserved	-	-	保留
5	CC2P	RW	0	输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3:2	Reserved	-	-	保留
1	CC1P	RW	0	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; 1: OC1 低电平有效。 CC1 通道配置为输入: CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿: TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿: TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。

0	CC1E	RW	0	输入/捕获 1 输出使能 (Capture/Compare 1 output enable) CC1 通道配置为输出： 0: 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、 OSSR、 OIS1、 OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。 注： 对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

18.5.10 TIMx 计数器（TIMx_CNT） (x=2)

偏移地址：0x24

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT	RW	32’h0	计数器的值 (Counter value)

18.5.11 TIMx 计数器（TIMx_CNT） (x=3/4)

偏移地址：0x24

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT	RW	16’h0	计数器的值 (Counter value)

18.5.12 TIMx 预分频器（TIMx_PSC） (x=2/3/4)

偏移地址：0x28

复位值： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16'h0	预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清'0'或被工作在复位模式的从控制器清'0'。

18.5.13 TIMx 自动重装载寄存器（TIMx_ARR）（x=2）

偏移地址：0x2C

复位值：0xFFFF FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARR[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	Name	R/W	Reset Value	Function
31:0	ARR	RW	32'hFFFFFFFF	自动重装载的值 (Prescaler value) 自动重装载的值为空时，计数器不工作。 无抖动模式（DITHEN=0）： 该寄存器保持自动加载值 抖动模式（DITHEN=1）： 该寄存器保持整数部分 ARR[31:4]，ARR[3:0]包含抖动部分

18.5.14 TIMx 自动重装载寄存器（TIMx_ARR）（x=3/4）

偏移地址：0x2C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR	RW	16'hFFFF	自动重装载的值 (Prescaler value) 自动重装载的值为空时，计数器不工作。 无抖动模式（DITHEN=0）： 该寄存器保持自动加载值 抖动模式（DITHEN=1）： 该寄存器保持整数部分 ARR[19:4]，ARR[3:0]包含抖动部分

18.5.15 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x=2）

偏移地址：0x34

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR1	RW/R	32’h0	捕获/比较通道 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出： CCR1 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。

18.5.16 TIMx 捕获/比较寄存器 1（TIMx_CCR1）（x=3/4）

偏移地址：0x34

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1	RW/R	16’h0	捕获/比较通道 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出： CCR1 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。

18.5.17 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x=2）

偏移地址：0x38

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

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

Bit	Name	R/W	Reset Value	Function
31:0	CCR2	RW/R	32'h0	捕获/比较通道 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出： CCR2 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_CCMR2 寄存器(OC2PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。

18.5.18 TIMx 捕获/比较寄存器 2（TIMx_CCR2）（x=3/4）

偏移地址：0x38

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2	RW/R	16'h0	捕获/比较通道 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出： CCR2 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_CCMR2 寄存器(OC2PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。

18.5.19 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x=2）

偏移地址：0x3C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR3	RW/R	32'h0	捕获/比较通道 3 的值 (Capture/Compare 3 value)

				若 CC3 通道配置为输出： CCR3 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_CCMR3 寄存器(OC3PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。

18.5.20 TIMx 捕获/比较寄存器 3（TIMx_CCR3）（x=3/4）

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR3[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR3	RW/R	16’h0	捕获/比较通道 3 的值 (Capture/Compare 3 value) 若 CC3 通道配置为输出： CCR3 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_CCMR3 寄存器(OC3PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。

18.5.21 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x=2）

偏移地址：0x40

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:0	CCR4	RW/R	32’h0	捕获/比较通道 4 的值 (Capture/Compare 4 value) 若 CC4 通道配置为输出： CCR4 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_CCMR4 寄存器(OC4PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。

				当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。

18.5.22 TIMx 捕获/比较寄存器 4（TIMx_CCR4）（x=3/4）

偏移地址：0x40

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR4[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4	RW/R	16’h0	捕获/比较通道 4 的值 (Capture/Compare 4 value) 若 CC4 通道配置为输出： CCR4 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_CCMR4 寄存器(OC4PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。

18.5.23 TIMx 输入选择寄存器（TIMx_TISEL）（x=2/3/4）

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TI4SEL[3:0]				Res.	Res.	Res.	Res.	TI3SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27:24	TI4SEL	RW	4’h0	TI4 输入选择. 0000：TIMx_CH4 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
23:20	Reserved	-	-	保留
19:16	TI3SEL	RW	4’h0	TI3 输入选择. 0000：TIMx_CH3 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

15:12	Reserved	-	-	保留
11:8	TI2SEL	RW	4'h0	TI2 输入选择. 0000：TIMx_CH2 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL	RW	4'h0	TI1 输入选择. 0000：TIMx_CH1 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

18.5.24 TIMx 备用功能选项寄存器 1（TIMx_AF1）（x=2/3/4）

偏移地址：0x60

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:14	ETRSEL	RW	4'h0	外部触发源选择（etr_in source selection） 该位段用于选择 ETR 输入源 0000：TIMx_ETR 0001：COMP1_OUT 0010：COMP2_OUT 0011：TIM2_ETR for TIM3_ETR_IO,TIM3_ETR for TIM2_ETR_IO，TIM4_ETR for TIM2_ETR_IO 0100：TIM2_ETR for TIM4_ ETR_IO,TIM3_ETR for TIM4_ETR_IO，TIM4_ETR for TIM3_ETR_IO 其它：保留
13:0	Reserved	-	-	保留

18.5.25 TIMx 备用功能选项寄存器 2（TIMx_AF2）（x=2/3/4）

偏移地址：0x64

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:19	Reserved	-	-	保留
18:16	OCRSEL	RW	3'h0	OCREF 复位源选择（ocref_clr source selection） 该位段用于选择 ocref_clr 输入源 000：COMP1_OUT 001：COMP2_OUT 其它：保留
15:0	Reserved	-	-	保留

18.5.26 TIMx DMA 控制寄存器（TIMx_DCR）（x=2/3/4）

偏移地址：0x3DC

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5'h0	DMA 连续传送长度（DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000： 1 次传输 00001： 2 次传输 00010： 3 次传输 .... 例：我们考虑这样的传输： DBL=7 字节， DBA=TIMx_CR1 - 如果 DBL=7 字节， DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址,这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	-	-	保留
4:0	DBA	RW	5'h0	DMA 基地址（DMA base address) 这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量：

				00000： TIMx_CR1， 00001： TIMx_CR2， 00010： TIMx_SMCR， ……..

18.5.27 TIMx 连续模式的 DMA 地址（TIMx_DMAR）（x=2/3/4）

偏移地址：0x3E0

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	32’h0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： TIMx_CR1 地址 + (DBA + DMA 索引)x4 其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

19. 通用定时器（TIM15/TIM16/TIM17）

19.1 TIM15/TIM16/TIM17 简介

通用控制定时器(TIM15/TIM16/TIM17)由一个 16 位的自动装载计数器组成，计数器由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。通用控制定时器(TIM15_16_17)和通用定时器(TIMx)是完全独立的，它们不共享任何资源。它们可以同步操作。

19.2 TIM15 主要特征

TIM15 定时器的功能包括：

- 16 位向上的自动装载计数器
- 16 位可编程(可以实时修改)的预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 2 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘对齐模式)
  - 单脉冲模式输出
- 带有可编程死区时间的互补输出(只有通道 1)
- 通过外部信号来控制定时器与定时器之间互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出，计数器初始化(通过软件或者内部/外部触发)
  - 触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
  - 输入捕获
  - 输出比较
  - 刹车信号输入

19.2.1 TIM15 模块框图

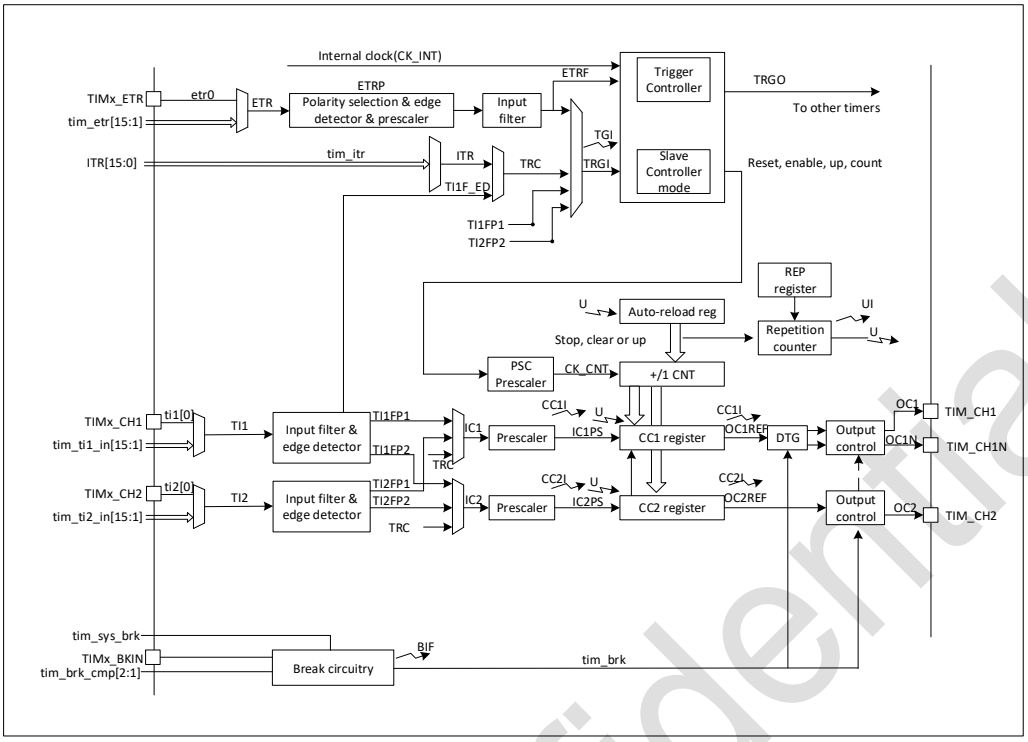


图 19-1 TIM15 模块框图

19.3 TIM16/TIM17 主要特征

TIM16/TIM17 定时器的功能包括：

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

19.3.1 TIM16/TIM17 模块框图

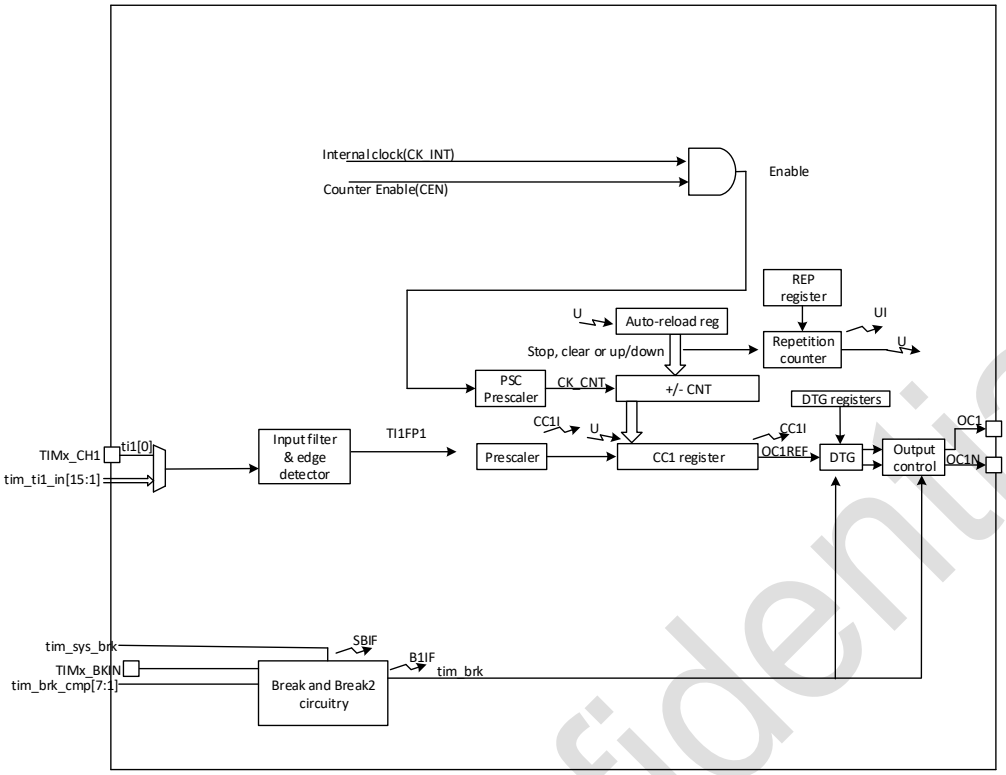


图 19-2 TIM16/TIM17 模块框图

19.4 TIM15/TIM16/TIM17 功能描述

19.4.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。计数器的时钟可以被预分频器分频。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器(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)

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问它的预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件（UEV)时传送到影子寄存器。当计数器达到溢出条件(上溢或者下溢)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，会产生更新事件。更新事件也可以由软件及其他条件产生。后续会详细描述每一种配置下更新事件的产生。

计数器由预分频器分频后的时钟输出 CK_CNT 驱动，仅当设置了 TIMx_CR1 寄存器中的计数器使能位(CEN)，CK_CNT 才对计数器有效。(更多有关使能计数器的细节，请参见从模式控制器的描述)。

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

预分频器描述

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

下面几张图给出了在预分频器运行时，更改计数器参数的例子。

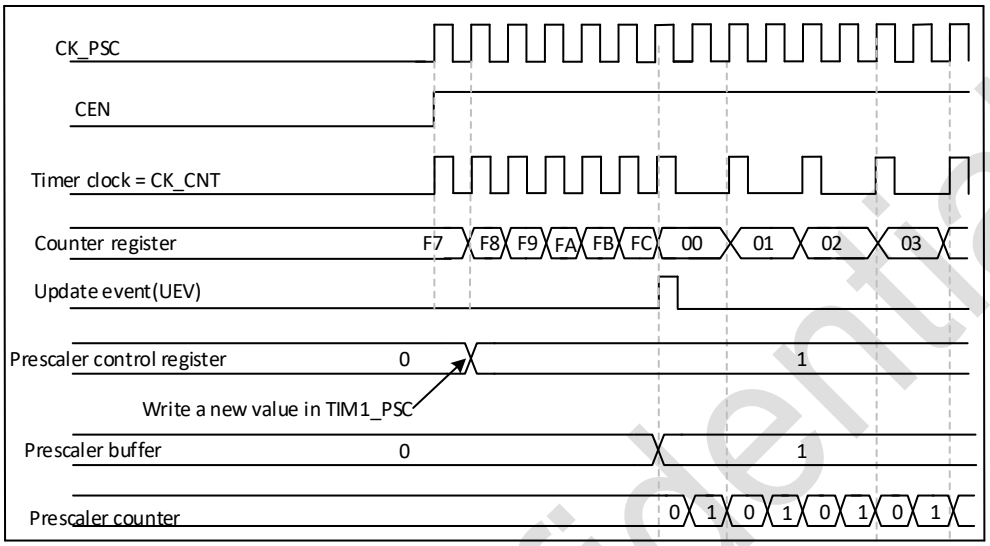


图 19-3 当预分频器的参数从 1 变到 2 时，计数器的时序图

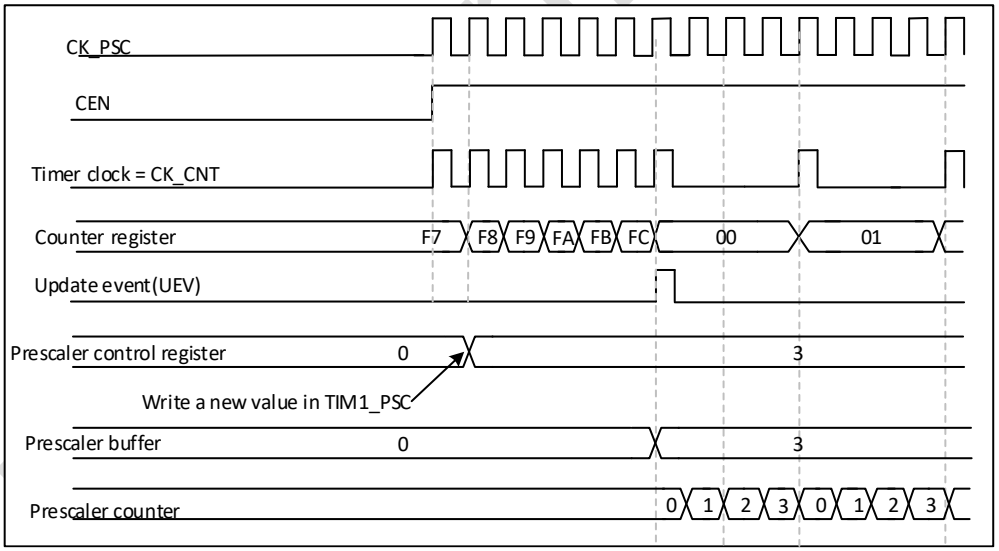


图 19-4 当预分频器的参数从 1 变到 4 时，计数器的时序图

19.4.2 计数器模式

向上计数模式

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

如果使用了重复计数器，那么更新事件(UEV)需要在上溢次数达到所配置的重复计数寄存器的值加一（即 TIMx_RCR+1）时才会产生；如果没有使用重复计数器（即 TIMx_RCR=0），那么每次计数上溢都会产生更新事件。

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

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

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

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

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

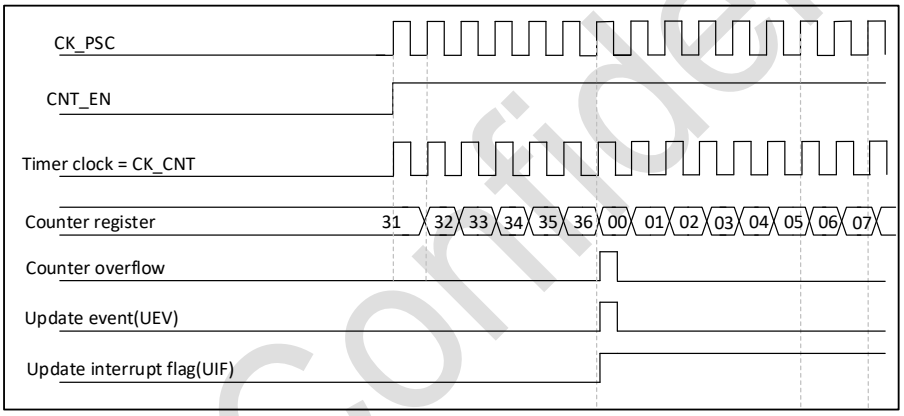


图 19-5 计数器时序图，内部时钟分频因子为 1

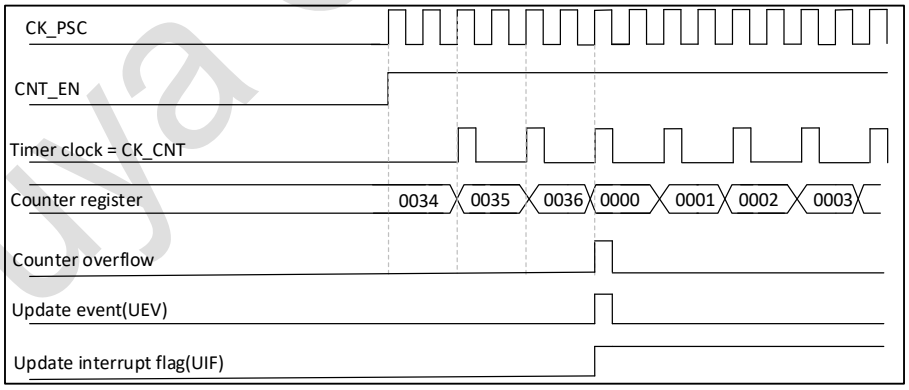


图 19-6 计数器时序图，内部时钟分频因子为 2

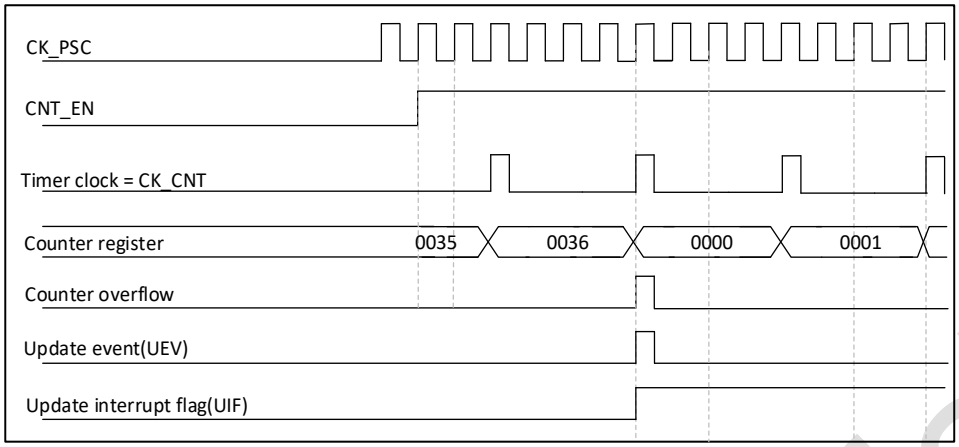


图 19-7 计数器时序图，内部时钟分频因子为 4

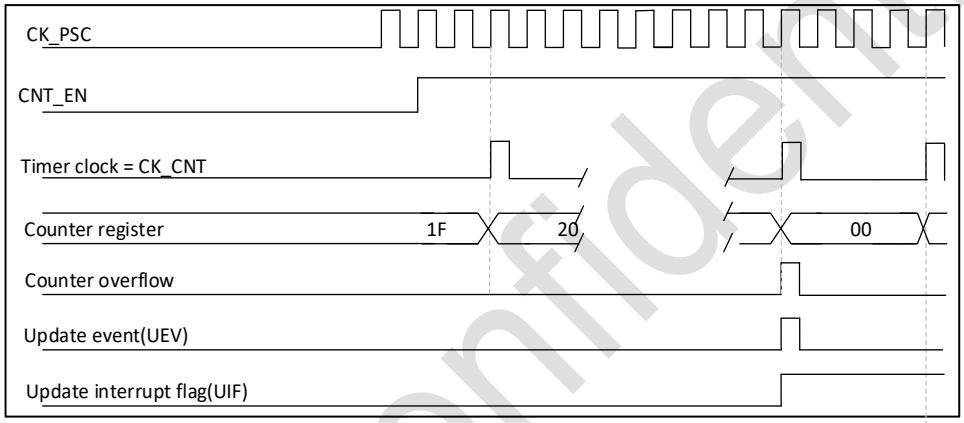


图 19-8 计数器时序图，内部时钟分频因子为 N

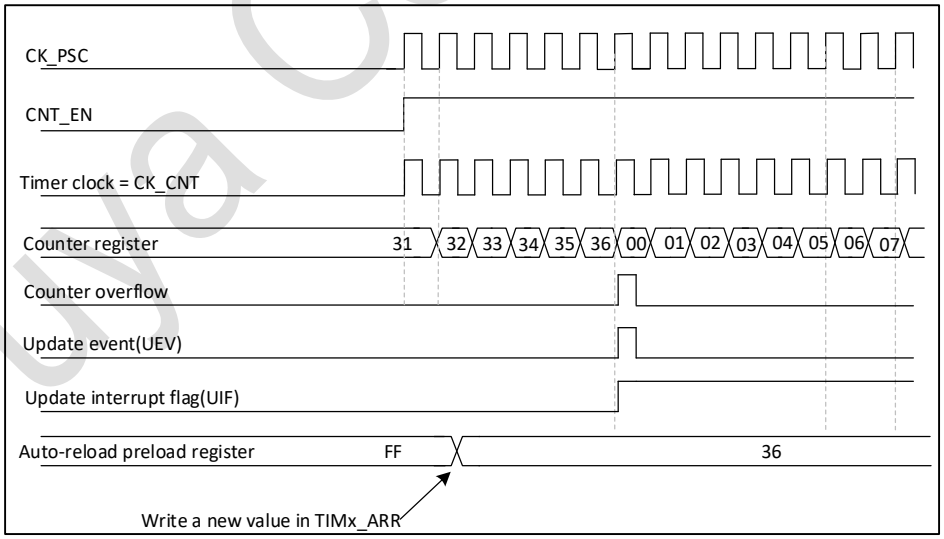


图 19-9 计数器时序图，当 ARPE=0 时的更新事件(没有预装 TIMx_ARR)

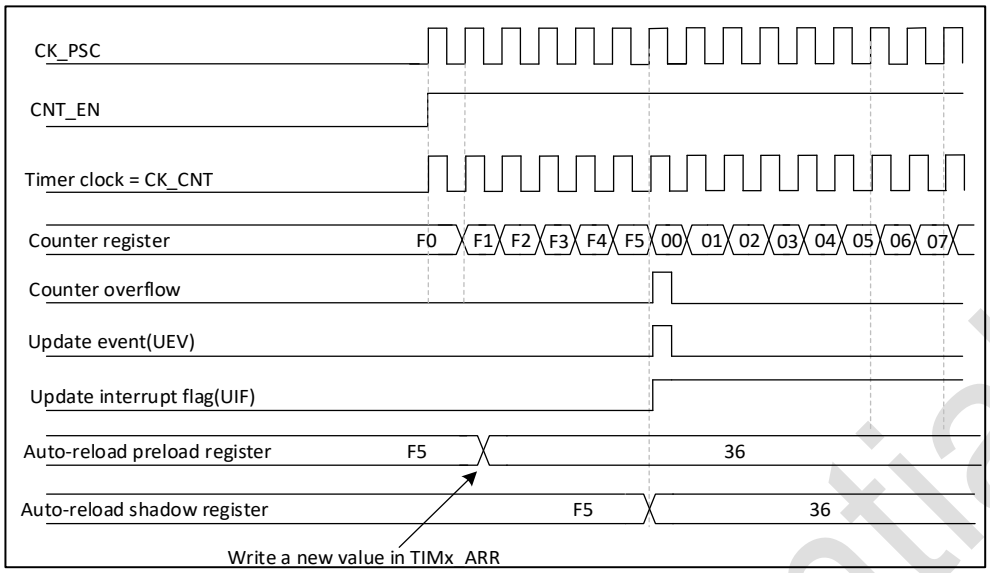


图 19-10 计数器时序图，当 ARPE=1 时的更新事件(预装了 TIMx_ARR)

19.4.3 重复计数器

“时基单元”解释了计数器溢出时更新事件(UEV)是如何产生的，事实上它只能在重复计数器计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N+1 个计数溢出时，数据才会从预装载寄存器传输到影子寄存器(TIMx_ARR 自动重载寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在计数器溢出时递减。

重复计数器是自动重载的，重复速率由 TIMx_RCR 寄存器的值定义。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

下图不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置

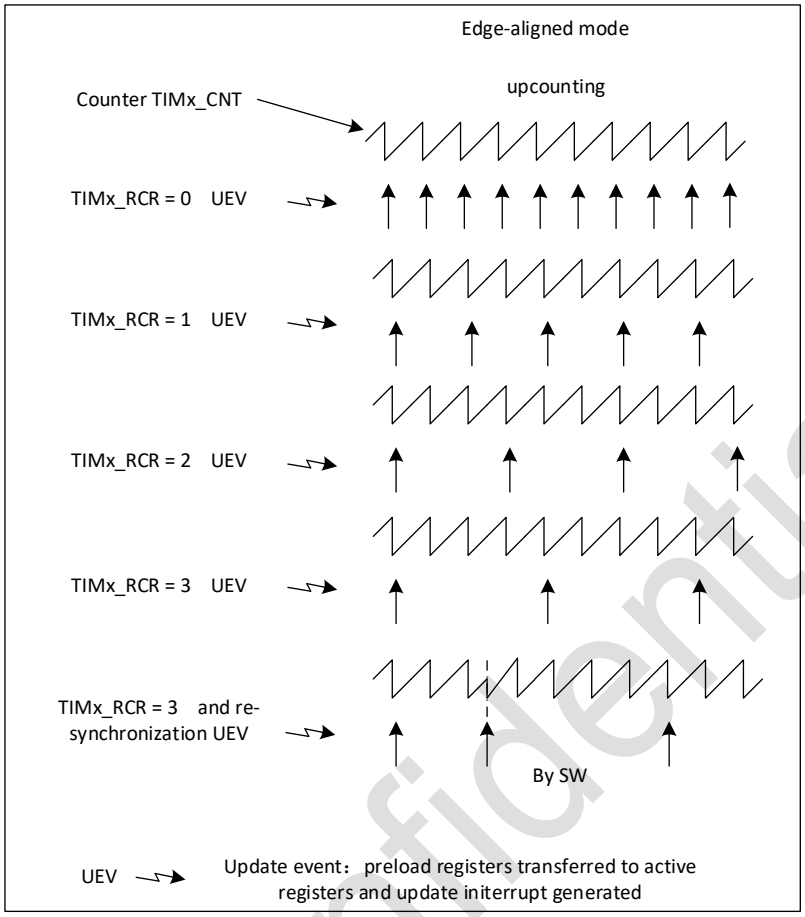


图 19-11 向上计数示例

19.4.4 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟(CK_INT)
- 外部时钟模式 1：外部输入引脚(TI1 和 TI2) （仅 TIM15）
- 内部触发输入(ITRx)(仅 tim15)：使用一个定时器作为另一个定时器的预分频器。

内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=0000)，则 CEN 和 UG 位(TIMx_EGR 寄存器)是事实上的控制位，并且只能被软件修改(UG 位仍被自动清除)。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

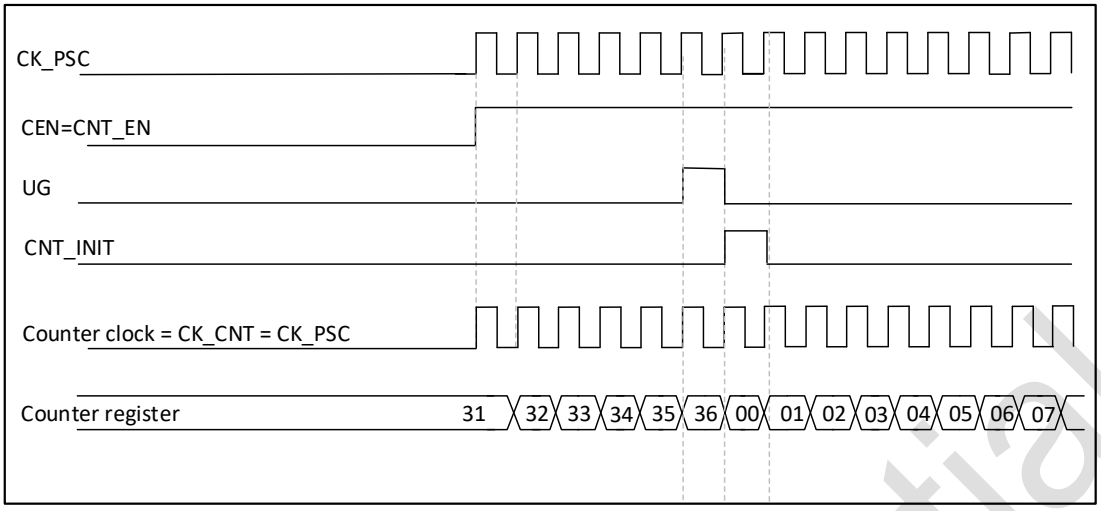


图 19-12 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式 1

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

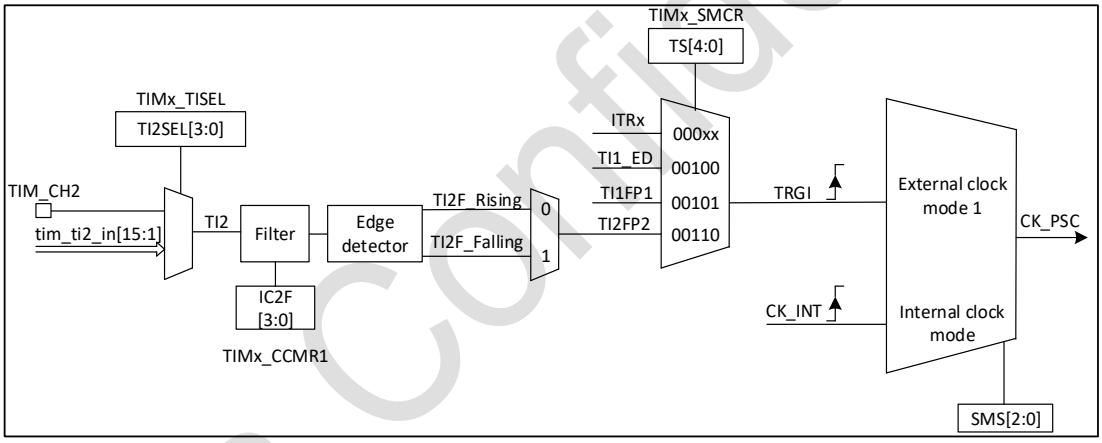


图 19-13 TI2 外部时钟连接例子

例如，要配置计数器在 TI2 输入端的上升沿向上计数，使用下列步骤：

1.配置 TIMx_CCMR1 寄存器 CC2S=01，使得通道 2 检测 TI2 输入端的上升沿；

2.配置 TIMx_CCMR1 寄存器的 IC2F[3:0],选择输入滤波器带宽 (如果不需要滤波器，保持 IC2F=0000)；

3.配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；

4.配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器为外部时钟模式 1；

5.配置 TIMx_SMCR 寄存器中的 TS=00110，选定 TI2 作为触发输入源；

6.设置 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

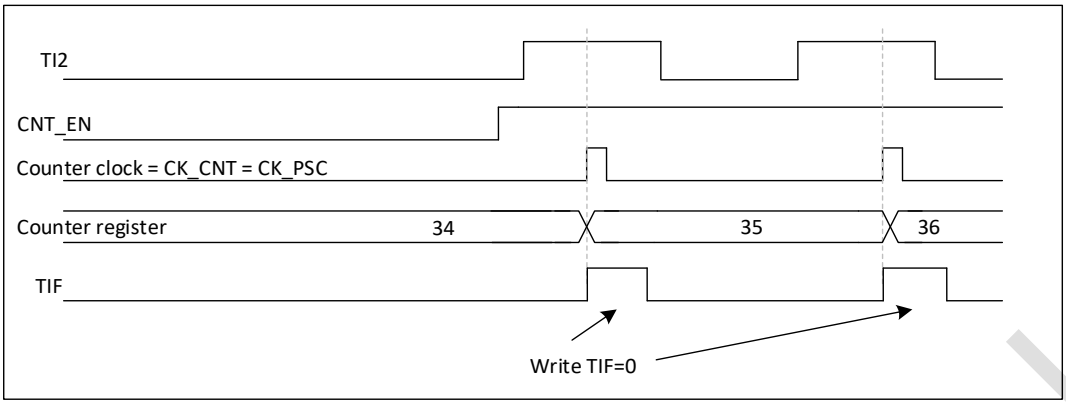


图 19-14 外部时钟模式 1 下的控制电路

19.4.5 捕获/比较通道

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

下边几张图是一个捕获/比较通道概览。

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

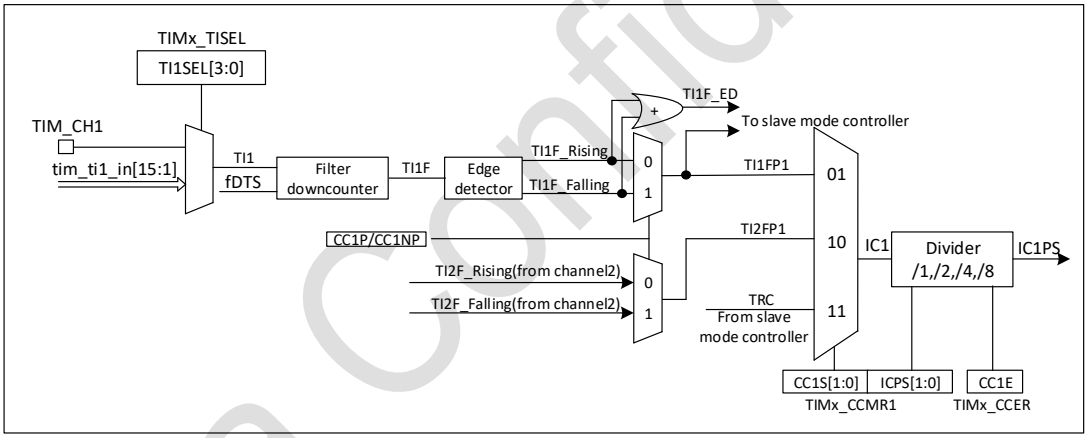


图 19-15 捕获/比较通道(如：通道 1 输入部分)

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

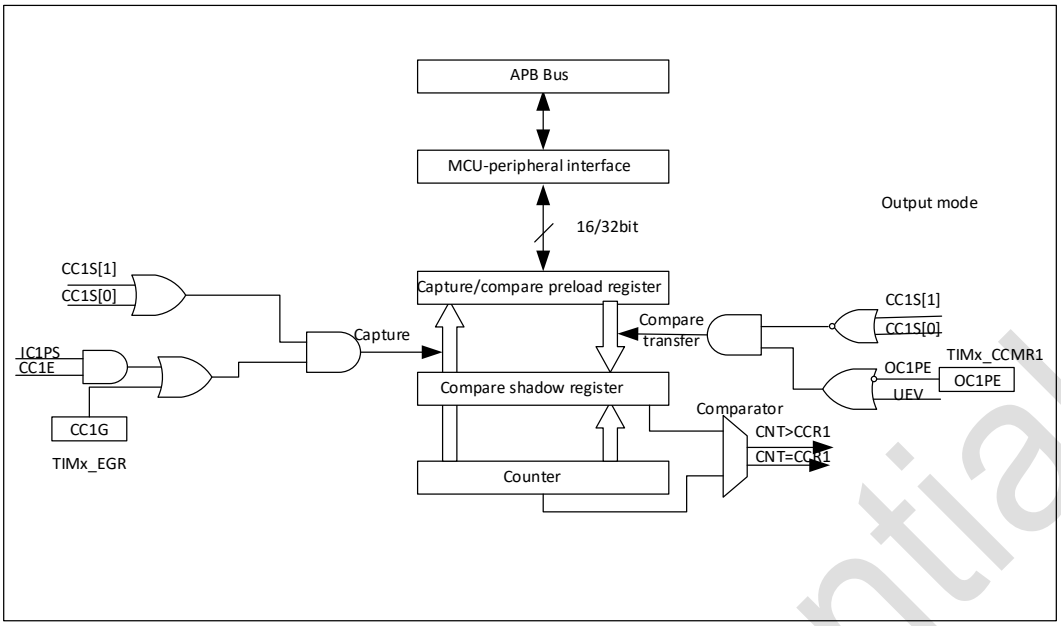


图 19-16 捕获/比较通道 1 的主电路

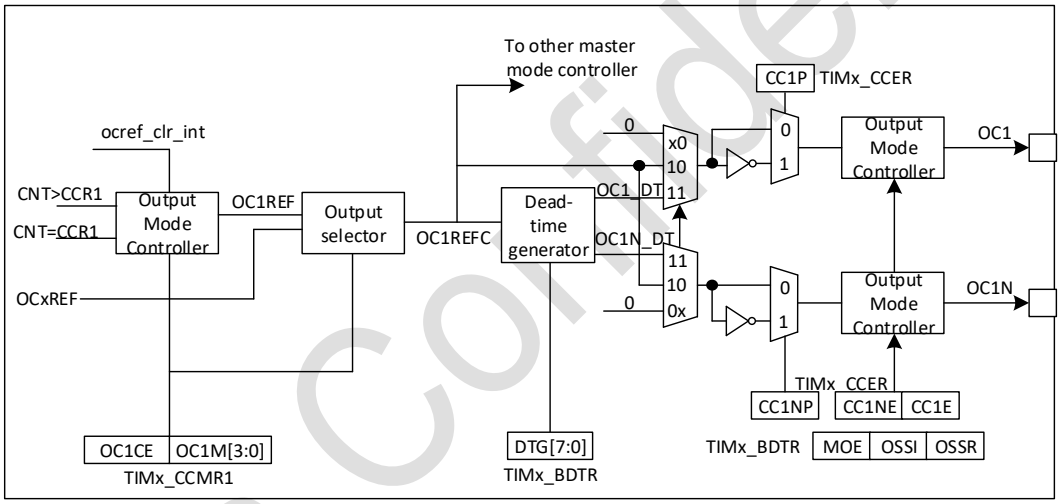


图 19-17 捕获/比较通道的输出部分(通道 1)

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

在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。

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

19.4.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果使能中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么过捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。通过写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMx_CCMR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为‘00’，通道就被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们

须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。

- 选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(设置为上升沿)。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，可以通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，也可通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

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

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

注： 设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

19.4.7 PWM 输入模式（仅 TIM15）

该模式是输入捕获模式的一个特例，除下列区别外，其余操作与输入捕获模式相同：

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

例如，用户可以测量输入到 TI1 上的 PWM 信号的周期(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下

- 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
- 选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
- 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

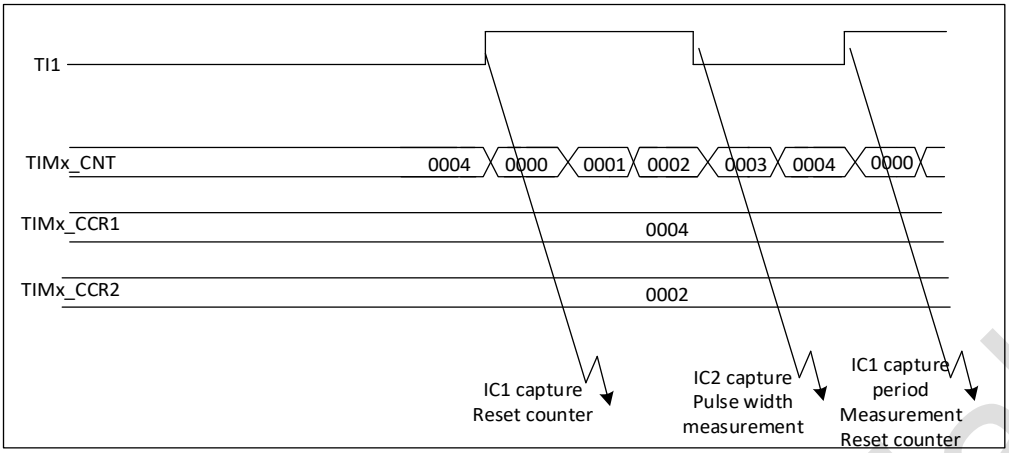


图 19-18 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器,所以 PWM 输入模式只能使用 TIMx_CH1/TIMx_CH2 信号。

19.4.8 强置输出模式

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

置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如： CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

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

19.4.9 输出比较模式

此项功能是用来控制一个输出波形，或者表明一段给定的的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=0000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
- 设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
- 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
- 若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位， TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

可以通过配置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。

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

输出比较模式的配置步骤：

1. 选择计数器时钟(内部，外部，预分频器)。
2. 将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：

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

5. 设置 TIMx_CR1 寄存器的 CEN 位启动计数器

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

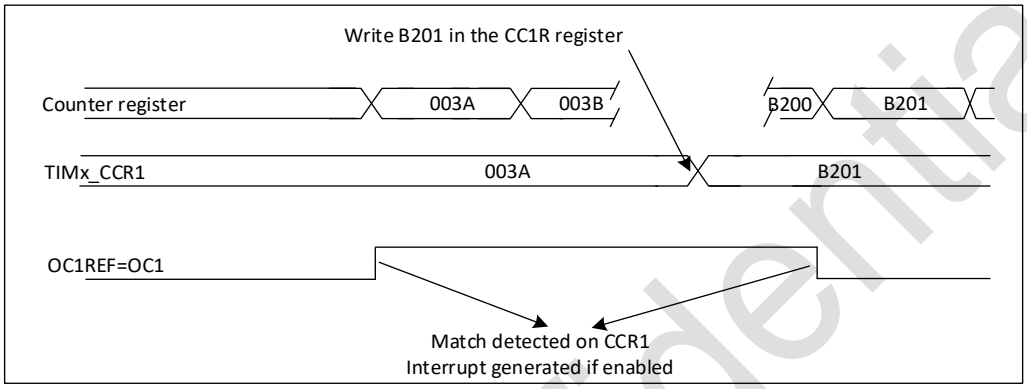


图 19-19 输出比较模式，翻转 OC1

19.4.10 PWM 模式

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

在 TIMx_CCMRx 寄存器中的 OCxM 位写入'110' (PWM 模式 1)或'111'(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，使能自动重装载的预装载寄存器。

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

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

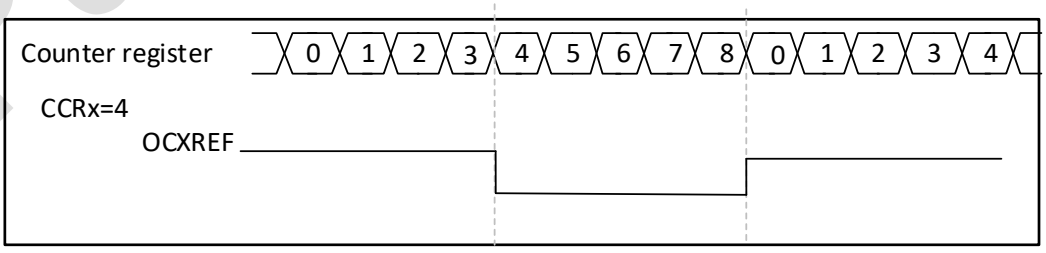


图 19-20 边沿对齐的 PWM 波形(APR=8)

19.4.11 互补输出和死区插入

通用控制定时器(TIM15_16_17)能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。

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

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

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。通过配置 TIMx_BDTR 寄存器中的 DTG[7:0]位，可以控制所有通道的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

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

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

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

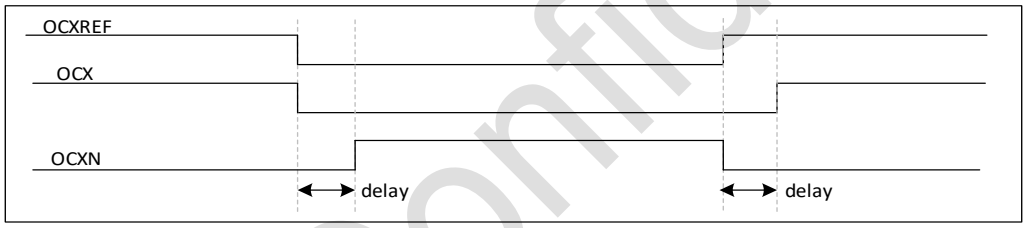


图 19-21 带死区插入的互补输出

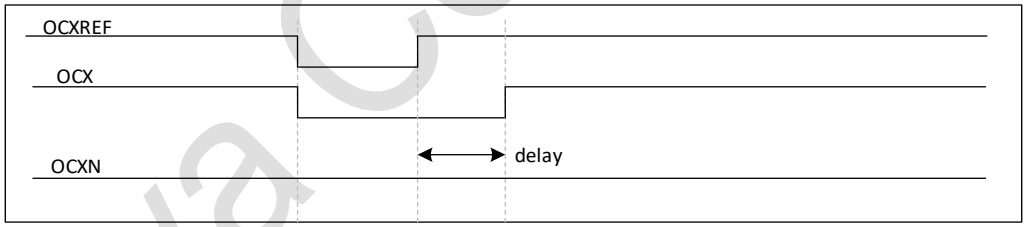


图 19-22 死区波形延迟大于负脉冲

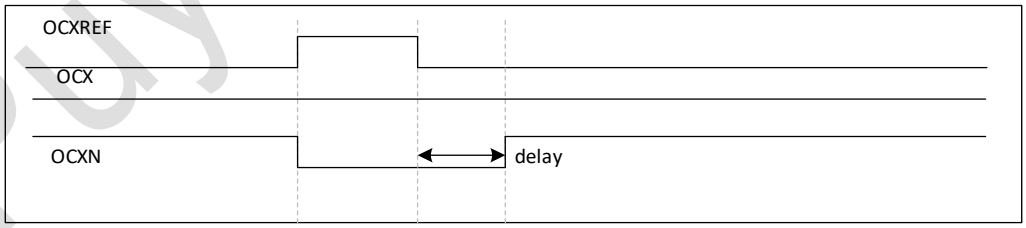


图 19-23 死区波形延迟大于正脉冲

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注：当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1)，当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

19.4.12 使用刹车功能

刹车功能的目的是保护由 TIMER 输出的 PWM 驱动的电源开关。两路刹车的输入通常是连接到三相逆变器和功率级芯片的故障输出。当刹车信号有效时，会立即关断 PWM 的所有输出，并将它们强制到预先设置好的一个安全的电平状态。部分 MCU 内部的事件也可以作为触发信号来关闭输出。

刹车事件有两路通道，通道 1 包括了系统级的故障(clock failure、奇偶校验等)和应用层的故障（通过输入引脚或者内置的比较器），并且可以在一个死区区间后强制输出到预先设置好的电平（无论工作还是空闲）。通道 2 仅包含一些应用层的故障，并且强制输出为无效状态。

在刹车过程中输出使能信号和输出电平由以下多个 bit 控制

- TIMx_BDTR 的 MOE 位可以通过软件或 2 路刹车事件复位来实现使能和关闭输出
- TIMx_BDTR 的 OSSI 位用于定义当处于空闲状态时 timer 是否还控制输出，或者释放对 GPIO 控制器的控制（高阻状态）
- TIMx_CR2 的 OISx 和 OISxN 位用于无论工作还是空闲状态下都将输出设置为无效电平，OCx 和 OCxN 输出在同一时刻不能同时设置为有效电平，无论 OISx 和 OISxN 为何值。

当从复位中退出时，刹车功能被关闭并且 MOE 为 0。刹车功能的开启可通过使能 TIMxBDTR 的 BKE 位。刹车输入的极性可通过 BKP 配置。BKEx 和 BKPx 可同时修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

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

刹车通道的源；

- 外部源可连接到其中一个 TIMx_BKIN 引脚上（选择 GPIO 和配置寄存器），再加上极性选择和滤波
- 内部源包含 2 部分
  - 来自刹车比较器信号（tim_brk_cmpx）
  - 来自系统刹车请求

刹车事件也可以由软件生成，可置位 TIMx_EGR 的 BG 位，所有刹车源均或起来作为 tim_brk 刹车输入。如下图：

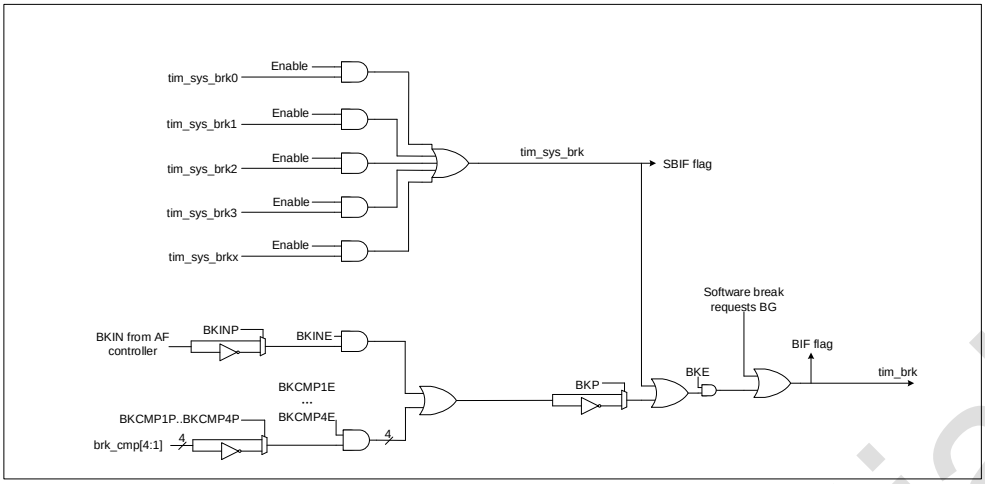


图 19-24 刹车电路

注意：只有没有滤波的时候才能保证是异步的操作，如果滤波开启了，必须使用一个故障保护时钟模式来保证刹车中断事件得到处理（例如使用内部 PLL 或 CSS，无滤波）。

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者释放对 GPIO 控制器的控制(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放输出控制，否则使能输出始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下， OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个时钟周期)。
  - 如果 OSSI=0，定时器释放输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 刹车状态标志会被拉高（TIMx_SR 的 BIF）。如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为‘1’时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 TDE 位，则产生一个 DMA 请求。
- 如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则， MOE 始终保持低直到被再次置‘1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注：当 AOE 为 1 时如果 MOE 被 CPU 复位了，输出会进入空闲状态，并被强制到无效电平或者高阻（取决于 OSSI 的值），如果 MOE 和 AOE 均被 CPU 复位，输出将进入无效状态并且由 OISx 位来驱动输出电平。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区持续时间， OCx/OCxN 极性和被禁止的状态， OCxM 配置，刹车使能和极性)。用户可以

通过设置 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

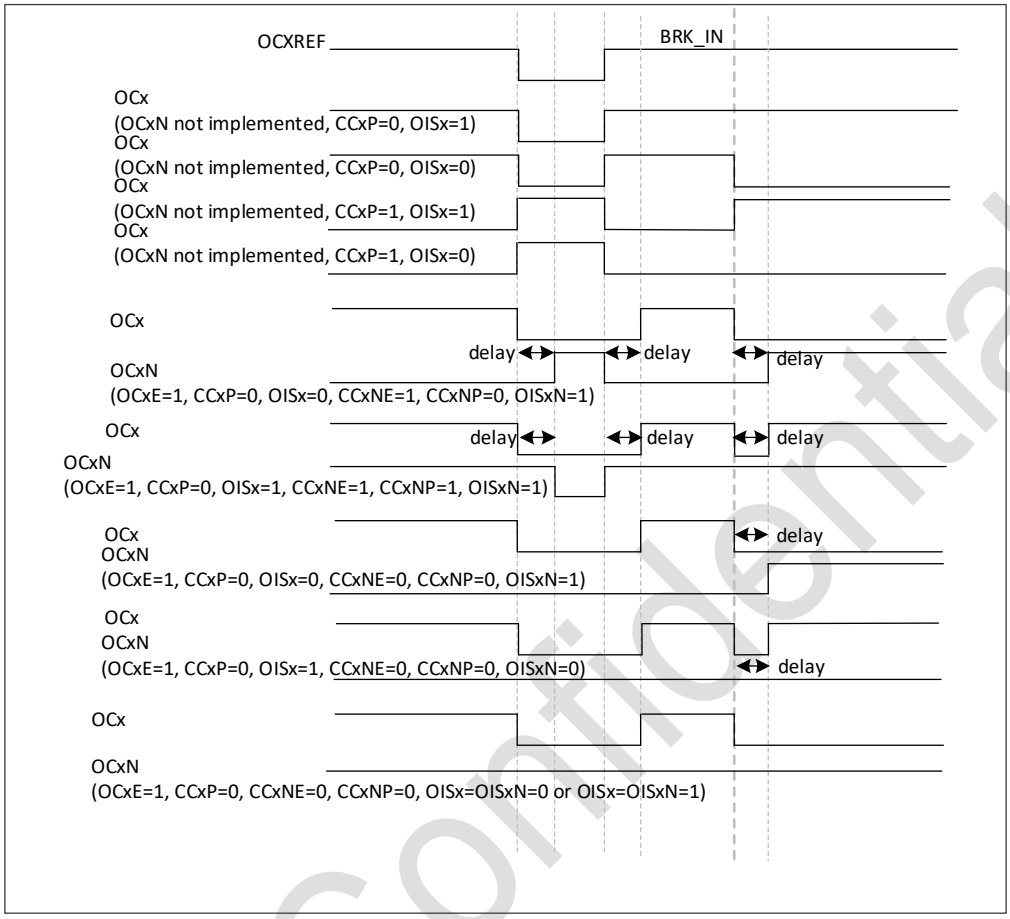


图 19-25 响应刹车的输出

19.4.13 单脉冲模式（仅 TIM15）

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

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，
- 向下计数方式：计数器 CNT > CCRx。

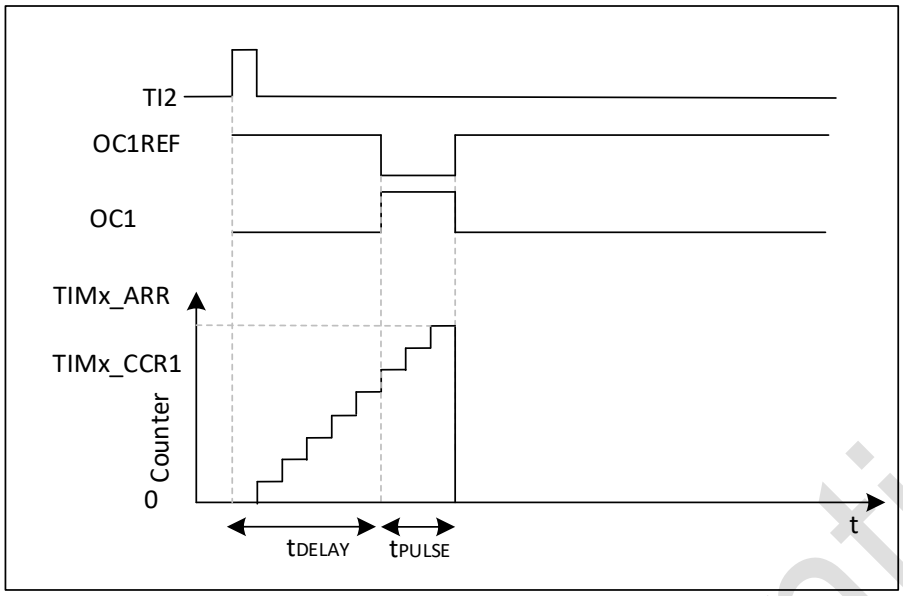


图 19-26 单脉冲模式的例子

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

假定 TI2FP2 作为触发:

- 置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映射到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110， TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)， TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形；首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中， CC1P=0。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 OPM=0 时，重复模式被选中。

特殊情况： OCx 快速使能：

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 t_{DELAY}。

如果要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。 OCxFE 只在通道配置为 PWM1 和 PWM2 模式时起作用。

19.4.14 定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式、触发、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 UDIS 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， TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=0100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=00101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

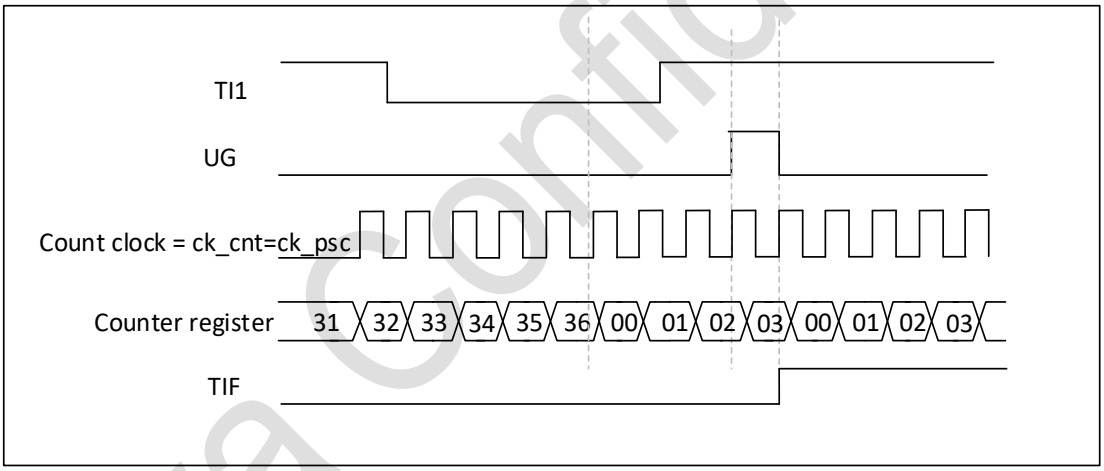


图 19-27 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。 CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 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。

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

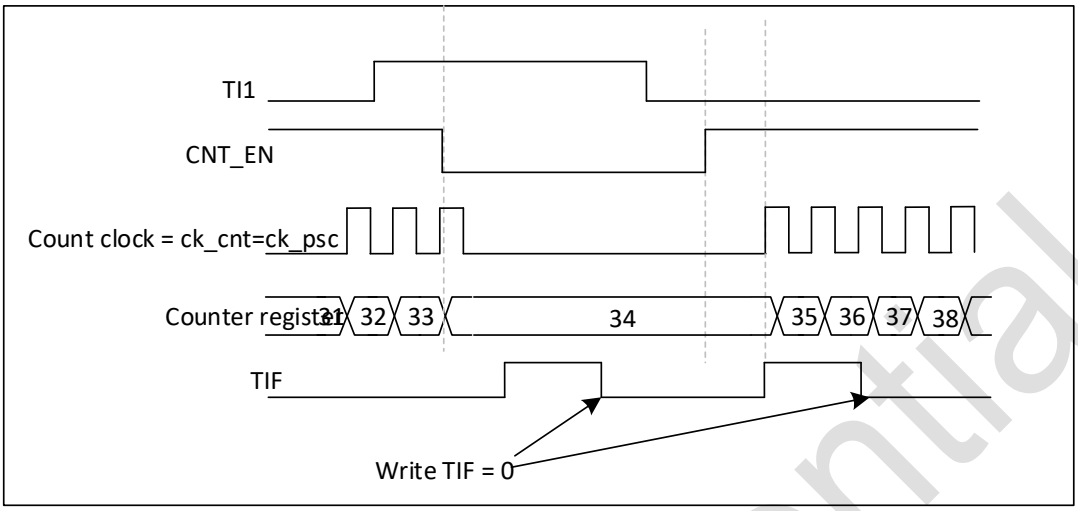


图 19-28 门控模式下的控制电路

从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

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

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

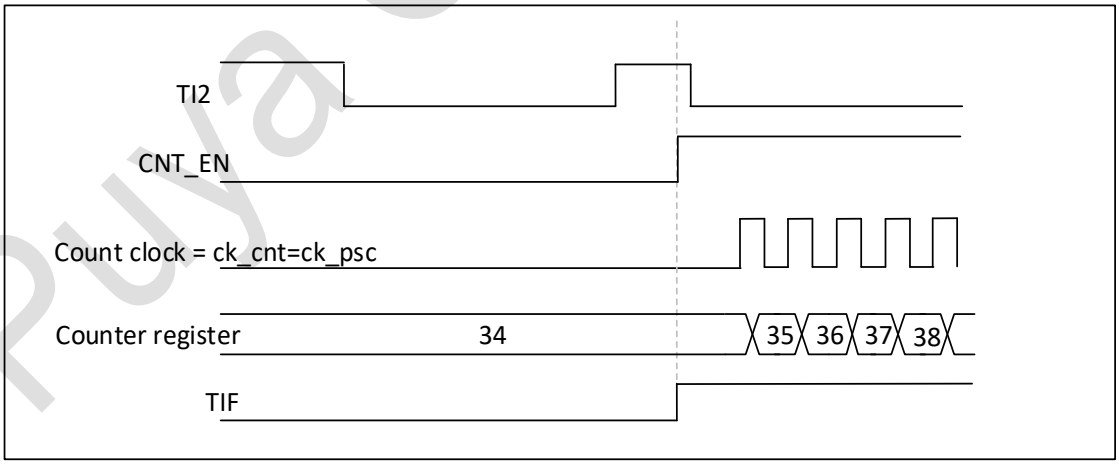


图 19-29 触发器模式下的控制电路

19.4.15 定时器同步（仅 TIM15）

所有 TIMx 定时器在内部相连，用于定时器同步或链接。

19.4.16 DMA 突发模式

TIMx 定时器当发生单个事件时有生成多个 DMA 请求的能力。主要目的是能够多次重新编程定时器的部分功能而无需软件的开销。但是也可以在固定的间隔时间内读取一行中的几个寄存器。

DMA 控制器的目的地是唯一的并且必须指向虚拟寄存器 TIMx_DMAR.当给定的定时器事件发生，定时器发起一系列 DMA 请求。每个写入 TIMx_DMAR 寄存器的操作实际上是重定向一个定时器寄存器的。

TIMx_DCR 寄存器中的 DBL[4:0]位设置 DMA 突发长度。当对 TIMx_DMR 地址进行读取或写入访问时，定时器会识别突发传输，即传输次数（以半个字或字节为单位）

TIMx_DCR 的 DBA[4:0]位定义了 DMA 传输的 DMA 基地址(当通过 TIMx_DMAR 地址完成读写操作)。DBA 定义为从 TIMx_CR1 寄存器地址开始的偏移量。

例子：

00000：TIMx_CR1

00001：TIMx_CR2

00010：TIMx_SMCR

作为一个例子，定时器 DMA 突发特性是用于发生更新事件时更新 CCRx 寄存器中的内容，通过 DMA 传输半字到 CCRx。

通过以下步骤来完成：

1. 如下配置对应的 DMA 通道：
   1) DMA 通道外设地址是 DMAR 寄存器的地址
   2) DMA 通道存储地址是 RAM 中的 BUFF 地址，包含了 DMA 传给 CCRx 的数据
   3) 传输的数据个数=3（见注释）
   4) 轮询模式关闭
2. 通过配置 DBA 和 DBL 实现配置 DCR 寄存器：DBL=3，DBA=0XE
3. 使能定时器更新 DMA 请求（UDE=1）
4. 使能定时器
5. 使能 DMA 通道

这个例子此示例适用于每个 CCRx 寄存器更新一次的情况。如果每个 CCRx 都更新两次，数据的传输个数就需要设置为 4。举例假设 RAM 的 buffer 中包含了 data1，data2，data3, data4。数据依次传输到 CCRx 中：第一次更新 DMA 请求，data1 传输给 CCR1，data2 传输给 CCR2；第二次更新 DMA 请求，data3 传输给 CCR1，data4 传输给 CCR2。

注：空值可以写入到保留寄存器

19.4.17 TIM15/TIM16/TIM17 DMA 请求

TIM15/TIM16/TIM17 可以生成 DMA 请求，如下表所示：

表 19-1 DMA 请求

DMA 请求源	DMA 使能控制位
更新	UDE
比较/捕获 1	CC1DE
COM(1)	COMDE
触发(1)	TDE

其中 1 仅用于 TIM15。

19.4.18 调试模式

当微控制器进入调试模式时(Cortex-M4 核心停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。详见随后的 DBG 节。

19.5 TIM15/TIM16/TIM17 中断

表 19-2 TIM15/TIM16/TIM17 中断请求

中断事件	事件标志	中断使能控制位
更新	UIF	UIE
比较/捕获 1	CC1IF	CC1IE
比较/捕获 2	CC2IF	CC2IE
COM	COMIF	COMIE
触发	TIF	TIE
刹车	BIF	BIE

19.6 TIM15 寄存器描述

19.6.1 TIMx 控制寄存器 1（TIMx_CR1)(x=15)

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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**
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	Res.	URS	UDIS	CEN
						RW	RW	RW					RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD	RW	2'h0	时钟分频因子 (Clock division) 这 2 位定义在定时器时钟(CK_INT)频率和死区时间和死区发生器与数字滤波器(ETR,TIx)所用的采样时钟（tDTS）之间的分频比例。 00： $t_{DTS} = t_{CK\_INT}$ 01： $t_{DTS} = 2 \times t_{CK\_INT}$ 10： $t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 (Auto-reload preload enable) 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
6:3	Reserved	-	-	保留
2	URS	RW	0	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢

				- 设置 UG 位 - 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： - 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注: 更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、 CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	使能计数器 (Counter enable) 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

19.6.2 TIMx 控制寄存器 2（TIMx_CR2)(x=15)

偏移地址：0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Res.	CCPC
					RW	RW	RW	RW	RW	RW	RW	RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10	OIS2	RW	0	输出空闲状态 2(OC2 输出)。参见 OIS1 位。
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7	Reserved	-	-	保留

6:4	MMS	RW	3'h0	主模式选择（Master mode selection) 用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下: 000： 复位 – TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001： 使能 – 计数器使能信号 CNT_EN 可被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时， TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010： 更新 – 更新事件被选为触发输出(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011： 比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100： 比较 – OC1REFC 信号被用于作为触发输出(TRGO)。 101： 比较 – OC2REFC 信号被用于作为触发输出(TRGO)。 其他：保留 注意： 1.从定时器和 ADC 的时钟必须先被使能以接收主定时器的信号，并在接收时不要改变。 2.若主从定时器不在同一总线上，主模式应该配置为能被从定时器采到的宽度。
3	CCDS	RW	0	捕获/比较的 DMA 选择（Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
2	CCUS	RW	0	捕获/比较控制更新选择（Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位（Capture/compare preloaded control) 0： CCxE， CCxNE 和 OCxM 位不是预装载的； 1： CCxE， CCxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新（COMG 位设置或 tim_trgi 上检测到的上升沿，具体取决于 CCUS 位）。 注：该位只对具有互补输出的通道起作用。

19.6.3 TIMx 从模式控制寄存器（TIMx_SMCR)(x=15)

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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TS[4:3]		Res.	Res.	Res.	Res.
										RW	RW				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Res.	SMS[2:0]		
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21:20	TS[4:3]	RW	2'h0	详见 TS[2:0]描述
19:16	Reserved	-	-	保留
15	ETP	RW	0	外部触发极性。该位选择是否 ETR 或者 ETR 的反向被用作触发操作。 0：ETR 不进行反向，高电平或者上升沿有效 1：ETR 反向，低电平或者下降沿有效
14	ECE	RW	0	外部时钟使能。这位使能外部时钟模式 2 0：外部时钟模式 2 不使能 1：外部时钟模式 2 使能，计数器工作在 ETRF 信号的有效沿
13:12	ETPS[1:0]	RW	00	外部触发预分频器。外部触发信号 ETRP 频率必须至多 TIMx CLK 频率的 1/4。一个预分频器可以被使能，以降低 ETRP 的频率。当输入快速外部时钟是有效的。 00：预分频器关闭 01：ETRP 频率的 2 分频 10：ETRP 频率的 4 分频 11：ETRP 频率的 8 分频
11:8	ETF[3:0]	RW	0000	外部触发滤波。这些位定义采样 ETRP 信号的频率和应用在 ETRP 的数字滤波长度。这个数字滤波由一个事件计数器组成，在改计数器里，N 个连续的事件被需要使输出的边沿有效。 0000：没有滤波器，在 f_{DTS} 下采样 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：$f_{SAMPLING}=f_{CK\_INT}/2$, N=6 0101：$f_{SAMPLING}=f_{CK\_INT}/2$, N=8 0110：$f_{SAMPLING}=f_{CK\_INT}/4$, N=6 0111：$f_{SAMPLING}=f_{CK\_INT}/4$, N=8 1000：$f_{SAMPLING}=f_{CK\_INT}/8$, N=6 1001：$f_{SAMPLING}=f_{CK\_INT}/8$, N=8 1010：$f_{SAMPLING}=f_{CK\_INT}/16$, N=5 1011：$f_{SAMPLING}=f_{CK\_INT}/16$, N=6 1100：$f_{SAMPLING}=f_{CK\_INT}/16$, N=8 1101：$f_{SAMPLING}=f_{CK\_INT}/32$, N=5 1110：$f_{SAMPLING}=f_{CK\_INT}/32$, N=6 1111：$f_{SAMPLING}=f_{CK\_INT}/32$, N=8
7	MSM	RW	0	主/从模式 (Master/slave mode) 0：无作用

				1: 触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
6:4	TS[2:0]	RW	3'h0	触发选择 (Trigger selection) 选择用于同步计数器的触发输入。 00000：内部触发 0(ITR0) 00001：内部触发 1(ITR1) 00010：内部触发 2(ITR2) 00011：内部触发 3(ITR3) 00100：TI1 边沿检测（TI1F_ED） 01001：内部触发 5(ITR5) 01010：内部触发 6(ITR6) 更多有关 ITRx 的细节，参见系统级联表 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
3	Reserved	-	-	保留
2:0	SMS	RW	3'h0	从模式选择 (Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：保留 010：保留 011：保留 100: 复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101: 门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：必须在接收主定时器事件之前，使能从设备（定时器，ADC）来接收 TRGO，并且时钟频率不能再接收过程中改变

19.6.4 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=15)

偏移地址：0x0C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	TDE	COMDE	Res.	Res.	CC2DE	CC1DE	UDE	BIE	TIE	COMIE	Res.	Res.	CC2IE	CC1IE	UIE
	RW	RW			RW	RW	RW	RW	RW	RW	Res.		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	TDE	RW	0	允许触发 DMA 请求 (Trigger DMA request enable) 0：禁止触发 DMA 请求 1：允许触发 DMA 请求
13	COMDE	RW	0	允许 COM 的 DMA 请求 (COM DMA request enable) 0：禁止 COM 的 DMA 请求 1：允许 COM 的 DMA 请求
12:11	Reserved	-	-	保留
10	CC2DE	RW	0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
9	CC1DE	RW	0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	允许更新的 DMA 请求 (Update DMA request enable) 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	允许刹车中断 (Break interrupt enable) 0：禁止刹车中断 1：允许刹车中断
6	TIE	RW	0	触发中断使能 (Trigger interrupt enable) 0：禁止触发中断 1：使能触发中断
5	COMIE	RW	0	允许 COM 中断 (COM interrupt enable) 0：禁止 COM 中断 1：允许 COM 中断
4:3	Reserved	-	-	保留
2	CC2IE	RW	0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	RW	0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	允许更新中断 (Update interrupt enable) 0：禁止更新中断 1：允许更新中断

19.6.5 TIMx 状态寄存器（TIMx_SR)(x=15)

偏移地址：0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IC2IF	IC1IF	Res.	Res.	IC2IR	IC1IR
										RC_W0	RC_W0			RC_W0	RC_W0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	CC2OF	CC1OF	Res.	BIF	TIF	COMIF	Res.	Res.	CC2IF	CC1IF	UIF
					RC_W0	RC_W0		RC_W0	RC_W0	RC_W0			RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:22	Reserved	-	-	保留
21	IC2IF	RC_W0	0	下降沿捕获 2 标志 参见 IC1IF 描述。
20	IC1IF	RC_W0	0	下降沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由下降沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生 1：发生下降沿捕获事件
19:18	Reserved	-	-	保留
17	IC2IR	RC_W0	0	上升沿捕获 2 标志 参见 IC1IR 描述。
16	IC1IR	RC_W0	0	上升沿捕获 1 标志 仅当相应的通道被配置为输入捕获时且由上升沿触发捕获事件，该标记可由硬件置 1。它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无重复捕获产生 1：发生上升沿捕获事件
15:11	Reserved	-	-	保留
10	CC2OF	RC_W0	0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时， CC1IF 的状态已经为‘1’。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标记 (Break interrupt flag) 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，则该位可由软件清‘0’。 0：无刹车事件产生 1：刹车输入上检测到有效电平。如果 TIM_DIER 的 BIE=1 则产生中断
6	TIF	RC_W0	0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置‘1’。它由软件清‘0’。 0：无触发器事件产生 1：触发中断等待响应
5	COMIF	RC_W0	0	COM 中断标记 (COM interrupt flag)

				一旦产生 COM 事件(当捕获/比较控制位： CCxE、 CCxNE、 OCxM 已被更新)该位由硬件置‘1’。它由软件清‘0’。 0：无 COM 事件产生 1： COM 中断等待响应
4:3	Reserved	-	-	保留
2	CC2IF	RC_W0	0	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 0：无匹配发生 1： TIMx_CNT 的值与 TIMx_CCR1 的值匹配 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下， CC1IF 位变高 如果通道 TI1 配置为输入模式： 当捕获事件发生时该位由硬件置‘1’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生 1: 计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)
0	UIF	RC_W0	0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若 TIMx_CR1 寄存器的 UDIS=0，重复计数器=0 时产生更新事件。 – 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 – 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当计数器 CNT 被触发事件重新初始化时。 (参考从模式控制寄存器(TIMx_SMCR))。

19.6.6 TIMx 事件产生寄存器（TIMx_EGR)(x=15)

偏移地址：0x14

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	TG	COMG	Res.	Res.	CC2G	CC1G	UG
								W	W	W			W	W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件 (Break generation) 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作 1: 产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
6	TG	W	0	产生触发事件 (Trigger generation)

				该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’ 0：无动作 1： TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
5	COMG	W	0	捕获/比较事件，产生控制更新 (Capture/Compare control update generation) 该位由软件置’1’，由硬件自动清’0’。 0：无动作 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位 注： 该位只对拥有互补输出的通道有效。
4:3	Reserved	-	-	保留
2	CC2G	W	0	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。
1	CC1G	W	0	产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作 1：在通道 1 上产生一个捕获/比较事件： 若通道 1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 (Update generation) 该位由软件置’1’，由硬件自动清’0’。 0：无动作 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清’0’(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

19.6.7 TIMx 捕获/比较模式控制寄存器 1（TIMx_CCMR1)(x=15)

偏移地址：0x18

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		Res.	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				ICIF[3:0]				ICIPSC[1:0]			
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

■ 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14:12	OC2M	RW	3’h0	输出比较 2 模式 (Output Compare 2 mode)
11	OC2PE	RW	0	输出比较 2 预装载使能 (Output Compare 2 preload enable)

10	OC2FE	RW	0	输出比较 2 快速使能 (Output Compare 2 fast enable)
9:8	CC2S	RW	2'h0	捕获/比较 2 选择。 (Capture/Compare 2 selection) 该位定义通道的方向(输入/输出)，及输入脚的选择: 00：CC2 通道被配置为输出 01：CC2 通道被配置为输入， IC2 映射在 TI2 上 10：CC2 通道被配置为输入， IC2 映射在 TI1 上 11：CC2 通道被配置为输入， IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7	Reserved	-	-	保留
6:4	OC1M	RW	3'h0	输出比较 1 模式 (Output Compare 1 mode) 这些位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、 OC1N 的值。 OC1REF 是高电平有效，而 OC1、 OC1N 的有效电平取决于 CC1P、 CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用; 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。 注 3：当通道有互补输出时，该位段可以预加载。如果 TIMx_CR2 的 CCPC 位置 1，然后仅当 COM 事件生成时 OC1M 有效位从预加载中更新新的值
3	OC1PE	RW	0	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。

				注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快输出对触发输入事件的响应。必须在单脉冲模式（TIMx_CR1 寄存器的 OPM 位置 1）中使用，当触发到来时实现脉冲快速输出。 0: 根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。仅能用于 PWM 模式 1 和 PWM 模式 2
1:0	CC1S	RW	2’h0	捕获/比较 1 选择。 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC1 通道被配置为输出； 01： CC1 通道被配置为输入， IC1 映射在 TI1 上； 10： CC1 通道被配置为输入， IC1 映射在 TI2 上； 11： CC1 通道被配置为输入， IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:12	IC2F	RW	4’h0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	RW	2’h0	输入/捕获 2 预分频器 (Input capture 2 prescaler)
9:8	CC2S	RW	2’h0	捕获/比较 2 选择 (Capture/Compare 2 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC2 通道被配置为输出； 01： CC2 通道被配置为输入，IC2 映射在 TI2 上； 10： CC2 通道被配置为输入，IC2 映射在 TI1 上； 11： CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
7:4	IC1F	RW	4’h0	输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2

				0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8
3:2	IC1PSC	RW	2'h0	输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1S=00，则预分频器复位。 00: 无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	2'h0	捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC1 通道被配置为输出； 01： CC1 通道被配置为输入， IC1 映射在 TI1 上； 10： CC1 通道被配置为输入， IC1 映射在 TI2 上； 11： CC1 通道被配置为输入， IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

19.6.8 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=15)

偏移地址：0x20

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC2NP	Res.	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
								RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留

7	CC2NP	RW	0	输入捕获 2 极性 (Capture/Compare 2 output polarity) 参考 CC1NP 的描述。
6	Reserved	-	-	保留
5	CC2P	RW	0	输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	RW	0	输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3	CC1NP	RW	0	输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	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、 OIS1、 OIS1N 和 CC1E 位的值。 1: 开启 - OC1N 信号输出到对应的输出引脚,其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1E 位的值。
1	CC1P	RW	0	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 1： OC1 低电平有效 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效 (捕获、复位模式下触发、外部时钟或触发模式下)； TIxFP1 不反相（门控模式）。 01：反相/下降沿： TIxFP1 下降沿有效 (捕获、复位模式下触发、外部时钟或触发模式下)； TIxFP1 反相（门控模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。 注： 1.对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1P 的实际有效位只有在 com 事件发生时才会加载预载值。 2.一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0	输入/捕获 1 输出使能 (Capture/Compare 1 output enable) CC1 通道配置为输出：

				0: 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、 OIS1、 OIS1N 和 CC1NE 位的值。 1: 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、 OSSR、OIS1、OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。 注： 对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

表 19-3 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE位	**OSSI位**	**OSSR位**	**CCxE位**	**CCxNE位**	**OCx输出状态**	**OCxN输出状态**
1	x	0	0	0	输出禁止（与定时器断开） OCx=0, OCx_EN=0	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	0	1	输出禁止（与定时器断开） OCx=0, OCx_EN=0	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP, OCx=0 OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP, OCx=0 OCxN_EN=0
		1	0	1	关闭状态（输出使能且为无效电平） OCx=CCxP, OCx_EN=1	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平） OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
0	0	x	x	x	输出禁止（与定时器断开）	
	1		0	0		
	1	x	0	1	关闭状态（输出使能但为无效电平） 异步的：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCx_EN=1;（如果刹车或刹车2触发了） 然后若时钟存在（仅刹车1触发时有效，不需要刹车2）：经过一个死区时间后OCx=OISx, OCxN=OISxN，假设OISx与OISxN并不都对应OCx和OCxN的有效电平 注：刹车2仅用于OSSI=OSSR=1	
	1		1	0		
	1		1	1		

特别的，当死区有效时，输出总是按照死区内输出的规则输出，尽管此时 moe 可能已经被软件或硬件的方式置为 1。

如果一个通道的 2 个输出都没有使用(CCxE = CCxNE = 0)，那么 OISx, OISxN, CCxP 和 CCxNP 都必须清零。

19.6.9 TIMx 计数器（TIMx_CNT)(x=15)

偏移地址：0x24

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT	RW	16’h0	计数器的值 (Counter value)

19.6.10 TIMx 预分频器（TIMx_PSC)(x=15)

偏移地址：0x28

复位值： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16’h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’。

19.6.11 TIMx 自动重装载寄存器（TIMx_ARR)(x=15)

偏移地址：0x2C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR	RW	16’hFFFF	自动重装载的值 (Prescaler value) 自动重装载的值为空时，计数器不工作。

19.6.12 TIMx 重复计数寄存器（TIMx_RCR)(x=15)

偏移地址：0x30

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	REP	RW	8’h0	重复计数器的值 (Repetition counter value) 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器达到 0，会产生一个更新事件并且重复计数器重新从 REP 值开始计数。对 TIMx_RCR 寄存器写入的新值只在下次更新事件发生时才起作用。

19.6.13 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=15)

偏移地址：0x34

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1	RW/R	16’h0	捕获/比较通道 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出： CCR1 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较,并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。 TIMx_CCR1 寄存器只读

19.6.14 TIMx 捕获/比较寄存器 2（TIMx_CCR2)(x=15)

偏移地址：0x38

复位值： 0x0000 0000

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

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2	RW/R	16'h0	捕获/比较通道 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出: CCR2 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较, 并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入: CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。 TIMx_CCR2 寄存器只读

19.6.15 TIMx 刹车和死区寄存器 (TIMx_BDTR)(x=15)

偏移地址：0x44

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态; 1: 如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、 CCxNE 位)，则开启 OC 和 OCN 输出。 有关 OC/OCN 使能的细节，参见 TIMx 捕获/比较使能寄存(TIMx_CCER)。
14	AOE	RW	0	自动输出使能 (Automatic output enable) 0： MOE 只能被软件置'1'; 1: MOE 能被软件置'1'或在下一个更新事件被自动置'1'(如果刹车输入无效)。 注: 一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 (Break polarity) 0：刹车输入低电平有效; 1：刹车输入高电平有效。 注: 一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。

				注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 (Break enable) 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 (Off-state selection for Idle mode) 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1， OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	2’h0	锁定设置 (Lock configuration) 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKBID、BK2BID、BKE、 BKP、 AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出， CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCxNP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出， CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	8’h0	死区发生器设置 (Dead-time generator setup) 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间: DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg， Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns；

				16μs 到 31750ns，若步长时间为 250ns； 32μs 到 63μs，若步长时间为 1μs； 64μs 到 126μs，若步长时间为 2μs； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、 2 或 3，则不能修改这些位。

19.6.16 TIMx 输入选择寄存器（TIMx_TISEL)(x=15)

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	TI2SEL[3:0]				Res.	Res.	Res.	Res.	TI1SEL[3:0]			
				RW	RW	RW	RW					RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:8	TI2SEL	RW	4’h0	TI2 输入选择（tim_ti2[15:0] input） 0000：TIMx_CH2 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留
7:4	Reserved	-	-	保留
3:0	TI1SEL	RW	4’h0	TI1 输入选择（tim_ti1[15:0] input） 0000：TIMx_CH1 0001：COMP1_OUT 0010：COMP2_OUT 其他：保留

19.6.17 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=15)

偏移地址：0x60

复位值：0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ETRSEL[3:2]	
														RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ETRSEL[1:0]		Res.	Res.	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW			RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:18	Reserved	-	-	保留
17:14	ETRSEL	RW	0	外部触发源选择（etr_in source selection）

				该位段用于选择 ETR 输入源 0000：TIMx_ETR 0001：COMP1_OUT 0010：COMP2_OUT 0011：ADC_AWD1 0100：ADC_AWD2 0101：ADC_AWD3 0110：保留 0111：保留 其它：保留 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
13:12	Reserved	-	-	保留
11	BKCMP2P	RW	0	tim_brk_cmp2 输入极性 该位选择 brk_cmp2 输入极性，必须与 BKP 极性位同时配置 0：tim_brk_cmp2 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp2 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	tim_brk_cmp1 输入极性 该位选择 brk_cmp1 输入极性，必须与 BKP 极性位同时配置 0：tim_brk_cmp1 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp1 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性（TIMx_BKIN input polarity） 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	tim_brk_cmp2 输入使能（tim_brk_cmp2 enable） 该位用于刹车输入时使能 tim_brk_cmp2。tim_brk_cmp2 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp2 输入关闭 1：tim_brk_cmp2 输入开启 注: 一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改

1	BKCMP1E	RW	0	tim_brk_cmp1 输入使能（tim_brk_cmp1 enable） 该位用于刹车输入时使能 tim_brk_cmp1。tim_brk_cmp1 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp1 输入关闭 1：tim_brk_cmp1 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能（TIMx_BKIN input enable） 该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

19.6.18 TIMx DMA 控制寄存器（TIMx_DCR)(x=15)

偏移地址：0x3DC

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5’h0	DMA 连续传送长度（DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 例：我们考虑这样的传输： DBL=7 字节， DBA=TIMx_CR1 - 如果 DBL=7 字节， DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。

7:5	Reserved	-	-	保留
4:0	DBA	RW	5'h0	DMA 基地址 (DMA base address) 这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量: 00000： TIMx_CR1， 00001： TIMx_CR2， 00010： TIMx_SMCR， ……

19.6.19 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=15)

偏移地址：0x3E0

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作: TIMx_CR1 地址 + (DBA + DMA 索引)x4 其中: "TIMx_CR1 地址"是控制寄存器 1(TIMx_CR1)所在的地址; "DBA"是 TIMx_DCR 寄存器中定义的基地址; "DMA 索引"是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

19.7 TIM16/TIM17 寄存器描述

19.7.1 TIMx 控制寄存器 1（TIMx_CR1)(x=16/17)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
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**
Res.	Res.	Res.	Res.	Res.	Res.	CKD[1:0]		ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
						RW	RW	RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9:8	CKD	RW	2'h0	时钟分频因子 (Clock division) 这 2 位定义在定时器时钟(CK_INT)频率和死区时间和死区发生器与数字滤波器(ETR,TIx)所用的采样时钟（tDTS）之间的分频比例。 00： $t_{DTS} = t_{CK\_INT}$

				01： $t_{DTS} = 2 \times t_{CK\_INT}$ 10： $t_{DTS} = 4 \times t_{CK\_INT}$ 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 (Auto-reload preload enable) 0： TIMx_ARR 寄存器没有缓冲； 1： TIMx_ARR 寄存器被装入缓冲器。
6:4	Reserved	-	-	保留
3	OPM	RW	0	单脉冲模式 (One pulse mode) 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： − 计数器溢出/下溢 − 设置 UG 位 − 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 (Update disable) 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： − 计数器溢出/下溢 − 设置 UG 位 − 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、 CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	使能计数器 (Counter enable) 0：禁止计数器 1：使能计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

19.7.2 TIMx 控制寄存器 2（TIMx_CR2)(x=16/17)

偏移地址：0x04

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	OIS1N	OIS1	TI1S	Res.	Res.	Res.	CCDS	CCUS	Res.	CCPC
						RW	RW	RW				RW	RW		RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	OIS1N	RW	0	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
8	OIS1	RW	0	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=0； 1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、 2 或 3 后，该位不能被修改。
7:4	Reserved	-	-	保留
3	CCDS	RW	0	捕获/比较的 DMA 选择（Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求 1：当发生更新事件时，送出 CCx 的 DMA 请求
2	CCUS	RW	0	捕获/比较控制更新选择（Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 注：该位只对具有互补输出的通道起作用。
1	Reserved	-	-	保留
0	CCPC	RW	0	捕获/比较预装载控制位（Capture/compare preloaded control) 0： CCxE， CCxNE 和 OCxM 位不是预装载的； 1： CCxE， CCxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新（COMG 位设置或 tim_trgi 上检测到的上升沿，具体取决于 CCUS 位）。 注：该位只对具有互补输出的通道起作用。

19.7.3 TIMx DMA/中断使能寄存器（TIMx_DIER)(x=16/17)

偏移地址：0x0C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1DE	UDE	BIE	Res.	COMIE	Res.	Res.	Res.	CC1IE	UIE
						RW	RW	RW		RW				RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	CC1DE	RW	0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0：禁止捕获/比较 1 的 DMA 请求 1：允许捕获/比较 1 的 DMA 请求
8	UDE	RW	0	允许更新的 DMA 请求 (Update DMA request enable)

				0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	允许刹车中断 (Break interrupt enable) 0：禁止刹车中断 1：允许刹车中断
6	Reserved	-	-	保留
5	COMIE	RW	0	允许 COM 中断 (COM interrupt enable) 0：禁止 COM 中断 1：允许 COM 中断
4:2	Reserved	-	-	保留
1	CC1IE	RW	0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	RW	0	允许更新中断 (Update interrupt enable) 0：禁止更新中断 1：允许更新中断

19.7.4 TIMx 状态寄存器（TIMx_SR)(x=16/17)

偏移地址：0x10

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	CC1OF	Res.	BIF	Res.	COMIF	Res.	Res.	Res.	CC1IF	UIF
						RC_W0		RC_W0		RC_W0				RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	CC1OF	RC_W0	0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无重复捕获产生 1: 计数器的值被捕获到 TIMx_CCR1 寄存器时， CC1IF 的状态已经为‘1’
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标记 (Break interrupt flag) 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，则该位可由软件清‘0’。 0：无刹车事件产生 1: 刹车输入上检测到有效电平。如果 TIM_DIER 的 BIE=1 则产生中断
6	Reserved	-	-	保留
5	COMIF	RC_W0	0	COM 中断标记 (COM interrupt flag)

				一旦产生 COM 事件(当捕获/比较控制位： CCxE、 CCxNE、 OCxM 已被更新)该位由硬件置‘1’。它由软件清‘0’。 0：无 COM 事件产生 1： COM 中断等待响应
4:2	Reserved	-	-	保留
1	CC1IF	RC_W0	0	捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag) 如果通道 CC1 配置为输出模式： 0：无匹配发生 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在溢出时，CC1IF 位变高 如果通道 TI1 配置为输入模式： 当捕获事件发生时该位由硬件置‘1’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)
0	UIF	RC_W0	0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： − 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器=0 时产生更新事件)。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 − 若 TIMx_CR1 寄存器的 URS=0、 UDIS=0，当计数器 CNT 被触发事件重新初始化时。(参考从模式控制寄存器(TIMx_SMCR))。

19.7.5 TIMx 事件产生寄存器（TIMx_EGR)(x=16/17)

偏移地址：0x14

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BG	Res.	COMG	Res.	Res.	Res.	CC1G	UG
								W		W				W	W

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	BG	W	0	产生刹车事件 (Break generation) 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作 1: 产生一个刹车事件。此时 MOE=0、 BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
6	Reserved	-	-	保留

5	COMG	W	0	捕获/比较事件，产生控制更新 (Capture/Compare control update generation) 该位由软件置'1'，由硬件自动清'0'。 0：无动作 1：当 CCPC=1，允许更新 CCxE、 CCxNE、 OCxM 位 注： 该位只对拥有互补输出的通道有效。
4:2	Reserved	-	-	保留
1	CC1G	W	0	产生捕获/比较 1 事件 (Capture/Compare 1 generation) 该位由软件置'1'，用于产生一个捕获/比较事件，由硬件自动清'0'。 0：无动作 1：在通道 1 上产生一个捕获/比较事件： 若通道 1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
0	UG	W	0	产生更新事件 (Update generation) 该位由软件置'1'，由硬件自动清'0'。 0：无动作 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清'0'(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

19.7.6 TIMx 捕获/比较模式控制寄存器 1 (TIMx_CCMR1)(x=16/17)

偏移地址：0x18

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ICIF[3:0]				ICIPSC[1:0]			
								RW	RW	RW	RW	RW	RW	RW	RW

■ 输出比较模式

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6:4	OC1M	RW	3'h0	输出比较 1 模式 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。 OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、 CC1NP 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。

				010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110： PWM 模式 1 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111： PWM 模式 2 - 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时， OC1REF 电平才改变。
3	OC1PE	RW	0	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作， TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
2	OC1FE	RW	0	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快输出对触发输入事件的响应。必须在单脉冲模式（TIMx_CR1 寄存器的 OPM 位置 1）中使用，当触发到来时实现脉冲快速输出。 0：根据计数器与 CCR1 的值， CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此， OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。仅能用于 PWM 模式 1 和 PWM 模式 2
1:0	CC1S	RW	2’h0	捕获/比较 1 选择。 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00： CC1 通道被配置为输出； 01： CC1 通道被配置为输入， IC1 映射在 TI1 上； 其他：保留 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

■ 输入捕获模式

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:4	IC1F	RW	4’h0	输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变：

				0000：无滤波器，以 f_{DTS} 采样 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=2 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=4 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$， N=8 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$， N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$， N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$， N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$， N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$， N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$， N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$， N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$， N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$， N=8
3:2	IC1PSC	RW	2’h0	输入/捕获 1 预分频器 (Input capture 1 prescaler) 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1S=00，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
1:0	CC1S	RW	2’h0	捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注： CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

19.7.7 TIMx 捕获/比较使能寄存器（TIMx_CCER)(x=16/17)

偏移地址：0x20

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CC1NP	CC1NE	CC1P	CC1E
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留

3	CC1NP	RW	0	输入/捕获 1 互补输出极性 (Capture/Compare 1 complementary output polarity) 0： OC1N 高电平有效; 1： OC1N 低电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
2	CC1NE	RW	0	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、 OIS1、 OIS1N 和 CC1E 位的值。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1E 位的值。
1	CC1P	RW	0	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; 1： OC1 低电平有效。 CC1 通道配置为输入： CC1NP/CC1P 位选择作为触发或捕获信号的 TI1FP1 和 TI2FP1 的极性。 00：不反相/上升沿： TIxFP1 上升沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式、编码器模式）。 01：反相/下降沿： TIxFP1 下降沿有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 反相（门控模式、编码器模式）。 10：保留，不要使用这个配置。 11：不反相/双沿 TIxFP1 上升和下降沿都有效（捕获、复位模式下触发、外部时钟或触发模式下）； TIxFP1 不反相（门控模式）。这个配置不能应用于编码器模式下。 注： 1.对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1P 的实际有效位只有在 com 事件发生时才会加载预载值。 2.一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
0	CC1E	RW	0	输入/捕获 1 输出使能 (Capture/Compare 1 output enable) CC1 通道配置为输出： 0： 关闭 - OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、 OSSI、 OSSR、 OIS1、 OIS1N 和 CC1NE 位的值。 1： 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、 OSSI、 OSSR、OIS1、 OIS1N 和 CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止; 1：捕获使能。 注：

				对于互补输出通道，这一位是预载的。如果 TIMx_CR2 寄存器中的 CCPC 位被设置，那么 CC1E 的实际有效位只有在 com 事件发生时才会加载预载值。

表 19-4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE 位	**OSSI 位**	**OSSR 位**	**CCxE 位**	**CCxNE 位**	**OCx输出状态**	**OCxN输出状态**
1	x	0	0	0	输出禁止（与定时器断开） OCx=0, OCx_EN=0	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	0	1	输出禁止（与定时器断开） OCx=0, OCx_EN=0	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		0	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	输出禁止（与定时器断开） OCxN=0, OCxN_EN=0
		0	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
		1	0	0	输出禁止（与定时器断开） OCx=CCxP, OCx=0 OCx_EN=0	输出禁止（与定时器断开） OCxN=CCxNP, OCx=0 OCxN_EN=0
		1	0	1	关闭状态（输出使能且为无效电平） OCx=CCxP, OCx_EN=1	OCxREF+极性， OCxN=OCREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF+极性， OCx=OCREF xor CCxP, OCx_EN=1	关闭状态（输出使能且为无效电平） OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCxREF+极性+死区，， OCx_EN=1	OCxREF（取反）+极性+死区， OCxN_EN=1
0	0	x	x	x	输出禁止（与定时器断开）	
	1		0	0		
	1		0	1	关闭状态（输出使能但为无效电平） 异步的：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCx_EN=1;（如果刹车触发了）然后若时钟存在（仅刹车1触发时有效）：经过一个死区时间后OCx=OISx, OCxN=OISxN，假设OISx与OISxN并不都对应OCx和OCxN的有效电平	
	1		1	0		
	1		1	1		

特别的，当死区有效时，输出总是按照死区内输出的规则输出，尽管此时 moe 可能已经被软件或硬件的方式置为 1。

如果一个通道的 2 个输出都没有使用(CCxE = CCxNE = 0)，那么 OISx, OISxN, CCxP 和 CCxNP 都必须清零。

19.7.8 TIMx 计数器（TIMx_CNT)(x=16/17)

偏移地址：0x24

复位值： 0x0000 0000

31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留

15:0	CNT	RW	16'h0	计数器的值 (Counter value)

19.7.9 TIMx 预分频器 (TIMx_PSC)(x=16/17)

偏移地址：0x28

复位值： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16'h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清'0'或被工作在复位模式的从控制器清'0'。

19.7.10 TIMx 自动重装载寄存器 (TIMx_ARR)(x=16/17)

偏移地址：0x2C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR	RW	16'hFFFF	自动重装载的值 (Prescaler value) 自动重装载的值为空时，计数器不工作。

19.7.11 TIMx 重复计数寄存器 (TIMx_RCR)(x=16/17)

偏移地址：0x30

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	REP	RW	8'h0	重复计数器的值 (Repetition counter value) 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。

				每次向下计数器达到 0，会产生一个更新事件并且重复计数器重新从 REP 值开始计数。对 TIMx_RCR 寄存器写入的新值只在下次更新事件发生时才起作用。

19.7.12 TIMx 捕获/比较寄存器 1（TIMx_CCR1)(x=16/17)

偏移地址：0x34

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															
RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R	RW/R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1	RW/RO	16'h0	捕获/比较通道 1 的值 (Capture/Compare 1 value) 若 CC1 通道配置为输出： CCR1 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。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。 TIMx_CCR1 寄存器只读

19.7.13 TIMx 刹车和死区寄存器（TIMx_BDTR)(x=16/17)

偏移地址：0x44

复位值： 0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1: 如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、 CCxNE 位)，则开启 OC 和 OCN 输出。 有关 OC/OCN 使能的细节，参见 TIMx 捕获/比较使能寄存(TIMx_CCER)。

14	AOE	RW	0	自动输出使能 (Automatic output enable) 0： MOE 只能被软件置’1’; 1： MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 (Break polarity) 0：刹车输入低电平有效; 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
12	BKE	RW	0	刹车功能使能 (Break enable) 0：禁止刹车输入(BRK 及 CCS 时钟失效事件); 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
11	OSSR	RW	0	运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0); 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
10	OSSI	RW	0	空闲模式下“关闭状态”选择 (Off-state selection for Idle mode) 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明(TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0); 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1， OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
9:8	LOCK	RW	2’h0	锁定设置 (Lock configuration) 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护; 01: 锁定级别 1, 不能写入 TIMx_BDTR 寄存器的 DTG、BKBID、BK2BID、BKE、 BKP、 AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位; 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出， CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCxNP 位)以及 OSSR/OSSI 位;

				11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出， CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG	RW	8'h0	死区发生器设置（Dead-time generator setup) 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg，Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg，Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16μs 到 31750ns，若步长时间为 250ns； 32μs 到 63μs，若步长时间为 1μs； 64μs 到 126μs，若步长时间为 2μs； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位。

19.7.14 TIMx 输入选择寄存器（TIMx_TISEL)(x=16/17)

偏移地址：0x5C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TI1SEL[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	TI1SEL	RW	4'h0	TI1 输入选择（tim_ti1[15:0] input） 0000：TIMx_CH1 0001：MCO 0010：HSE_DIV32 0011：保留 0100：LSI 其他：保留

19.7.15 TIMx 备用功能选项寄存器 1（TIMx_AF1)(x=16/17)

偏移地址：0x60

复位值：0x0000 0001

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

15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	BK CMP2P	BK CMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BK CMP2E	BK CMP1E	BKINE
				RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
11	BKCMP2P	RW	0	tim_brk_cmp2 输入极性 该位选择 brk_cmp2 输入极性，必须与 BKP 极性位同时配置 0：tim_brk_cmp2 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp2 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	tim_brk_cmp1 输入极性 该位选择 brk_cmp1 输入极性，必须与 BKP 极性位同时配置 0：tim_brk_cmp1 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：tim_brk_cmp1 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性（TIMx_BKIN input polarity） 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	tim_brk_cmp2 输入使能（tim_brk_cmp2 enable） 该位用于刹车输入时使能 tim_brk_cmp2。tim_brk_cmp2 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp2 输入关闭 1：tim_brk_cmp2 输入开启 注: 一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
1	BKCMP1E	RW	0	tim_brk_cmp1 输入使能（tim_brk_cmp1 enable） 该位用于刹车输入时使能 tim_brk_cmp1。tim_brk_cmp1 输入与其它刹车源进行或逻辑作为刹车输入。 0：tim_brk_cmp1 输入关闭 1：tim_brk_cmp1 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能（TIMx_BKIN input enable）

				该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

19.7.16 TIMx DMA 控制寄存器（TIMx_DCR)(x=16/17)

偏移地址：0x3DC

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5’h0	DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 例：我们考虑这样的传输： DBL=7 字节， DBA=TIMx_CR1 - 如果 DBL=7 字节， DBA=TIMx_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	-	-	保留
4:0	DBA	RW	5’h0	DMA 基地址 (DMA base address) 这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR，

				…….

19.7.17 TIMx 连续模式的 DMA 地址（TIMx_DMAR)(x=16/17)

偏移地址：0x3E0

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	32'h0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： TIMx_CR1 地址 + (DBA + DMA 索引)x4 其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

20. 基本定时器（TIM6/TIM7）

20.1 TIM6/TIM7 简介

基本定时器 TIM6 和 TIM7 各包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。

TIM6 和 TIM7 是完全独立的，它们不共享任何资源。

20.2 TIM6 和 TIM7 主要特征

TIM6 和 TIM7 定时器的主要功能包括：

- 16 位自动装载计数器
- 16 位可编程(可以实时修改)的预分频器，计数器时钟频率的分频系数为 1～65535 之间的任意数值
  在更新事件（计数器溢出）发生时产生中断/DMA

20.3 TIM6/TIM7 功能描述

20.3.1 功能框图

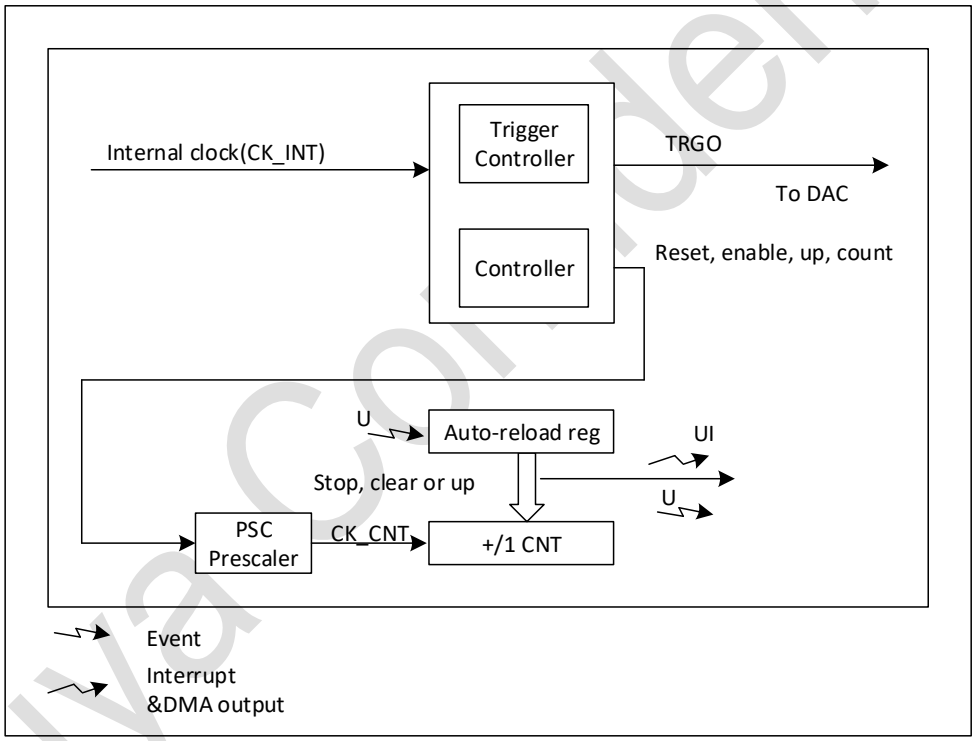


图 20-1 TIM6/TIM7 通用框图

20.3.2 时基单元

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。

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。

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

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

下图给出了在预分频器运行时，更改计数器参数的例子。

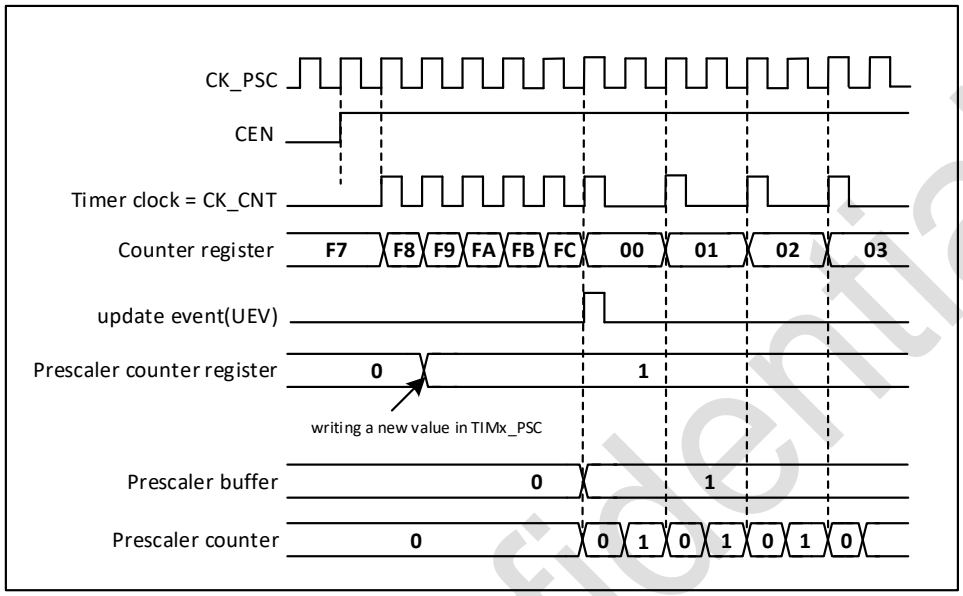


图 20-2 预分频器分频由 1 变为 2 时的计数器时序图

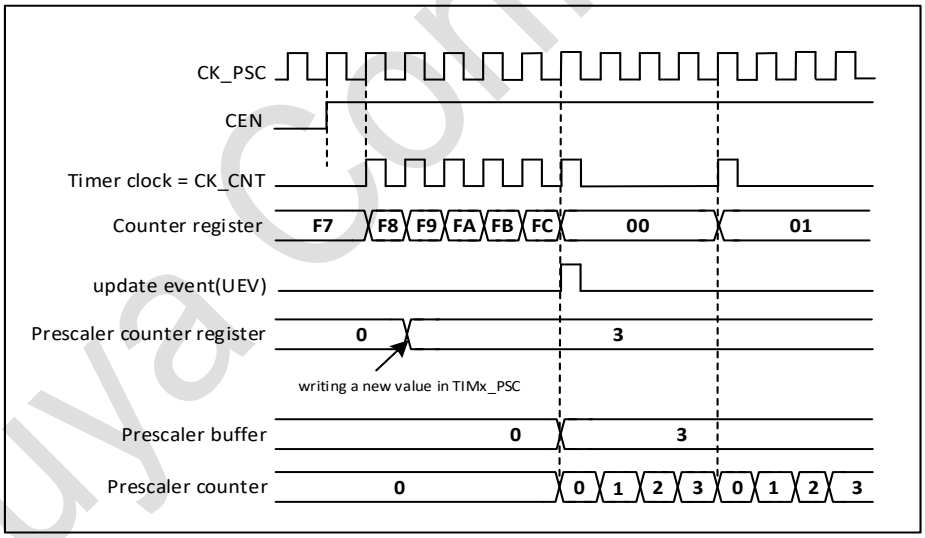


图 20-3 预分频器分频由 1 变为 4 时的计数器时序图

20.3.3 计数模式（递增）

计数器从 0 计数到自动加载值(TIMx_ARR 的内容)，然后重新从 0 开始计数并且产生一个计数上溢事件。

每次计数上溢都会产生更新事件。而在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，如下寄存器都被更新，硬件同时设置更新标志位(TIMx_SR 寄存器中的 UIF 位)置位 (依据 URS 位)：

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

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

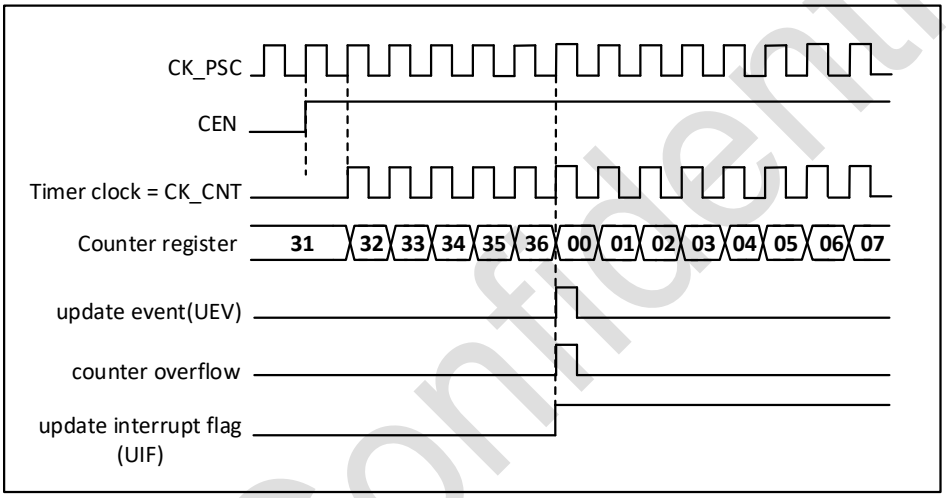


图 20-4 计数器时序图，内部时钟 1 分频

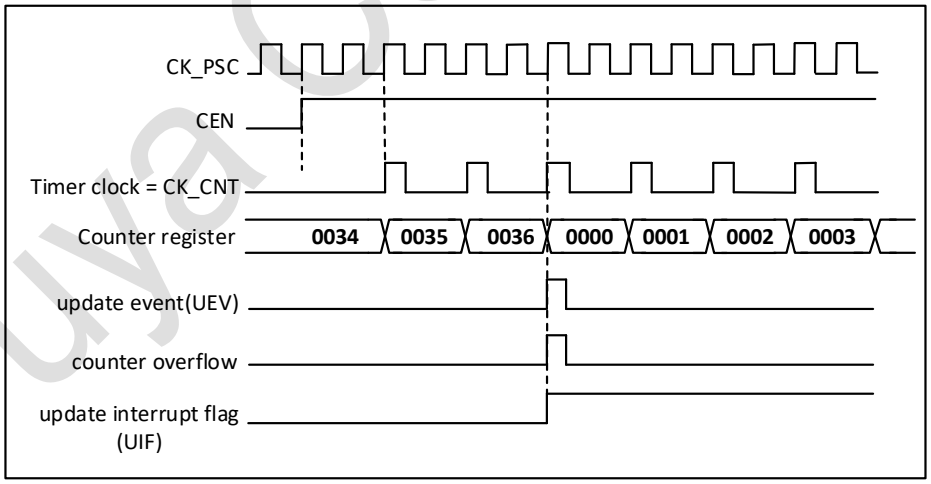


图 20-5 计数器时序图，内部时钟 2 分频

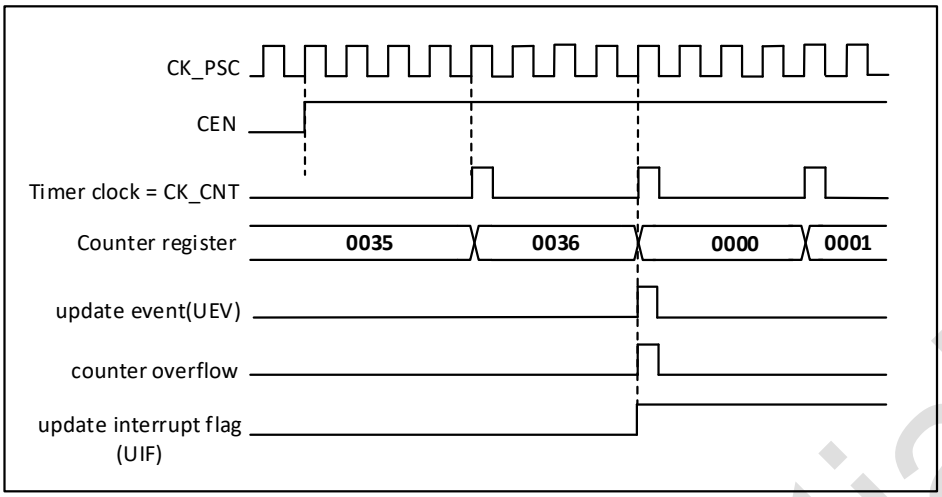


图 20-6 计数器时序图，内部时钟 4 分频

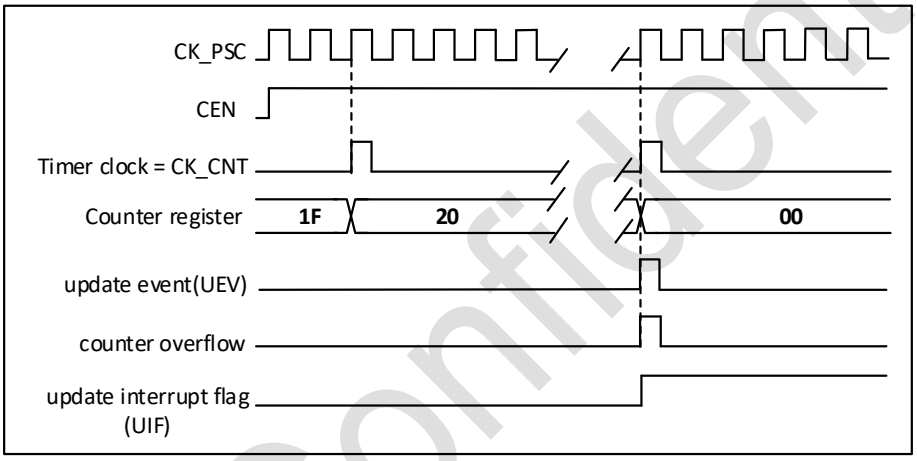


图 20-7 计数器时序图，内部时钟 N 分频

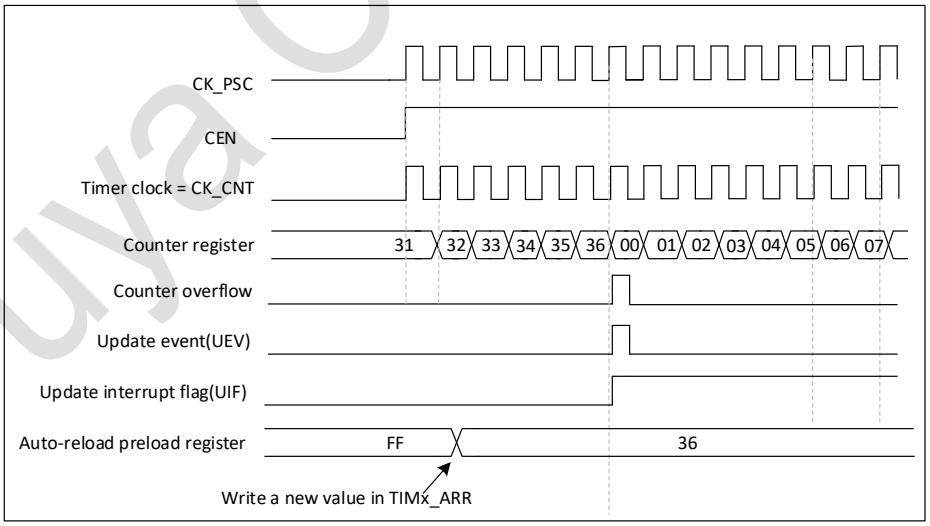


图 20-8 计数器时序图，ARPE=0 时的更新事件(TIMx_ARR 未预装载)

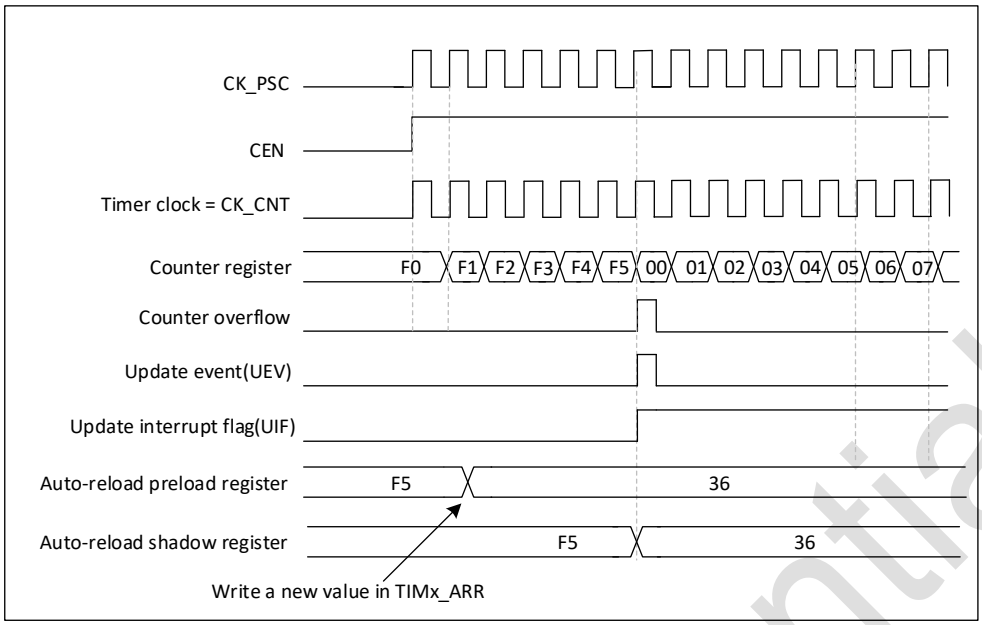


图 20-9 计数器时序图，ARPE=1 时的更新事件(TIMx_ARR 预装载)

20.4 TIM6/TIM7 调试模式

当微控制器进入调试模式时(Cortex®-M0+核停止)，根据 DBGMCU 模块中 DBG_TIMx_STOP 的设置，TIM6、TIM7 计数器可以或者继续正常操作，或者停止。

20.5 TIM6/TIM7 寄存器

20.5.1 TIM6/TIM7 控制寄存器 1（TIMx_CR1）

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARPE	Res.	Res.	Res.	OPM	URS	UDIS	CEN
								RW				RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	ARPE	RW	0	自动重装载预装载允许位 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
6:4	Reserved	-	-	保留
3	OPM	RW	0	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源。 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求：

				– 计数器上溢/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器上溢/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	更新禁止 软件通过该位允许/禁止 UEV 事件的产生。 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 影子寄存器装入预装载值。 1：禁止 UEV。不产生更新事件，影子寄存器(ARR,PSC,CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则重新初始化计数器和预分频器。
0	CEN	RW	0	使能计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，门控模式才能工作。触发模式可以自动地通过硬件设置 CEN 位为 1。 在单脉冲模式下，当产生更新事件时 CEN 被自动清除。

20.5.2 TIM6/TIM7 控制寄存器 2（TIMx_CR2）

偏移地址:0x04

复位值:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6:4	MMS[2:0]	RW	3’b0	主模式选择 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位---TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能---计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。该触发输出可用于在同一时间启动多个定时器或在一段时间内控制从定时器。计数器使能信号由 CEN 控制位与门控模式下的触发输入的逻辑或运算组合而成。 010：更新---更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。
3:0	Reserved	-	-	保留

20.5.3 TIM6/TIM7 DMA/中断使能寄存器（TIMx_DIER）

偏移地址:0x0C

复位值:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8	UDE	RW	0	更新的 DMA 请求使能 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7:1	Reserved	-	-	保留
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：允许更新中断

20.5.4 TIM6/TIM7 状态寄存器（TIMx_SR）

偏移地址:0x10

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	UIF	RC_W0	0	当产生更新事件时该位由硬件置 1。它由软件清 0。 0：无更新事件产生； 1：更新事件发生。当寄存器被更新时该位由硬件置 1： − 若 TIMx_CR1 寄存器的 UDIS=0，当计数器上溢时产生更新事件； − 若 TIMx_CR1 寄存器的 UDIS=0、URS=0，当 TIMx_EGR 寄存器的 UG=1 时产生更新事件(软件对 CNT 重新初始化)；

20.5.5 TIM6/TIM7 事件产生寄存器（TIMx_EGR）

偏移地址:0x14

复位值:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留

0	UG	W	0	产生更新事件 该位由软件置‘1’，由硬件自动清‘0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清‘0’(但是预分频系数不变)。

20.5.6 TIM6/TIM7 计数寄存器（TIMx_CNT）

偏移地址:0x24

复位值:0x0000 0000

TIM6：

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	RW	32’b0	计数器值

TIM7：

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	16’b0	计数器值

20.5.7 TIM6/TIM7 预分频寄存器（TIMx_PSC）

偏移地址:0x28

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC[15:0]	RW	16’b0	预分频器的值。 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC}/(PSC[15:0]+1)。 PSC 包含每次发生更新事件时要装载到有效预分频器寄存器的值（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）。

20.5.8 TIM6/TIM7 自动重载寄存器（TIMx_ARR）

偏移地址:0x2C

复位值:0xFFFF FFFF

TIM6:

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	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'hFFFF FFFF	自动重装载的值。 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

TIM7:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'hFFFF	自动重装载的值 (Prescaler value) ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

21. 专用脉冲宽度调制(PWM)

21.1 PWM 简介

PWM 模块通过可编程的周期寄存器、占空比寄存器和一个 16 位主计数器来实现 4 路 PWM 的输出，其中两路可互补输出(仅 PWM1/PWM2/PWM3 支持)。计数器由一个可编程的预分频器驱动。

21.2 PWM1/PWM2/PWM3 主要特征

PWM1/PWM2/PWM3 的主要功能包括：

- 在使用 PLL 作为时钟源时，最高可运行在 144 MHz（f_{HCLK} = 72 MHz）
- 16bit 向上、向下或者向上向下的自动重装载计数器
- 可编程周期和占空比
- 支持通过外部输入时钟为系统时钟的倍频的情况下计数
- 可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频
- 支持重要寄存器写保护
- 多达 4 个独立的通道
- 支持 2 个通道死区时间可编程的互补输出
- 输出极性可配置
- 支持边沿对齐和中心对齐
- 支持移相功能
- 刹车输入可以将定时器的输出信号置为复位状态和已知状态
- 支持一路刹车输入
- 支持外部时钟计数
- 支持 DMA
- 中断事件
  - 计数器向上、向下溢出
  - 输出比较
  - 刹车输入

21.3 PWM4 主要特征

PWM4 的主要功能包括：

- 16bit 向上、向下或者向上向下的自动重装载计数器
- 可编程周期和占空比
- 可编程分频器，允许对计数器的时钟频率进行 1 到 65535 的分频
- 多达 4 个独立的通道
- 输出极性可配置
- 支持边沿对齐和中心对齐
- 支持 DMA
- 中断事件
  - 计数器向上、向下溢出

21.4 PWM 功能描述

21.4.1 PWM1/PWM2/PWM3 模块框图

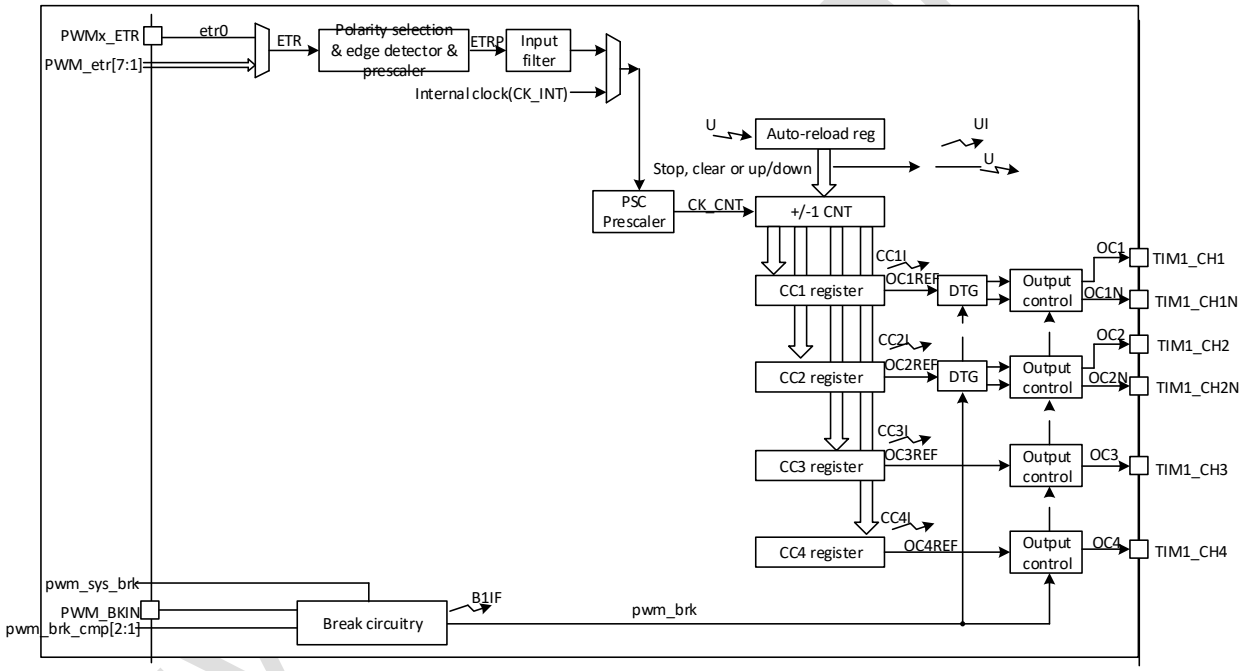


图 21-1 PWM1/PWM2/PWM3 模块框图

21.4.2 PWM4 模块框图

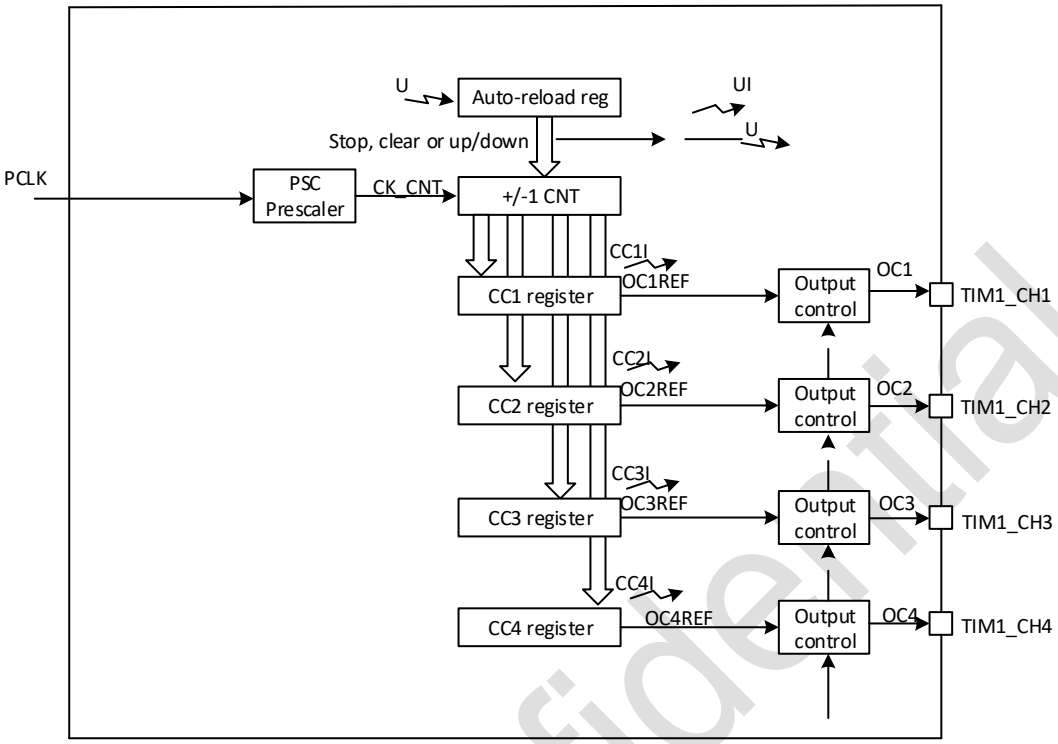


图 21-2 PWM4 模块框图

21.4.3 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。计数器的时钟可以被预分频器分频。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器（PWMx_CNT）
- 预分频器寄存器（PWMx_PSC）
- 自动装载寄存器（PWMx_ARR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问它的预装载寄存器。根据在 PWMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件（UEV）时传送到影子寄存器。当计数器达到溢出条件(上溢或者下溢)并当 PWMx_CR1 寄存器中的 UDIS 位等于 0 时，会产生更新事件。更新事件也可以由软件及其他条件产生。后续会详细描述每一种配置下更新事件的产生。

计数器由预分频器分频后的时钟输出 CK_CNT 驱动，仅当设置了 PWMx_CR1 寄存器中的计数器使能位(CEN)，CK_CNT 才对计数器有效。(更多有关使能计数器的细节，请参见从模式控制器的描述)。

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

预分频器描述

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

下面几张图给出了在预分频器运行时，更改计数器参数的例子。

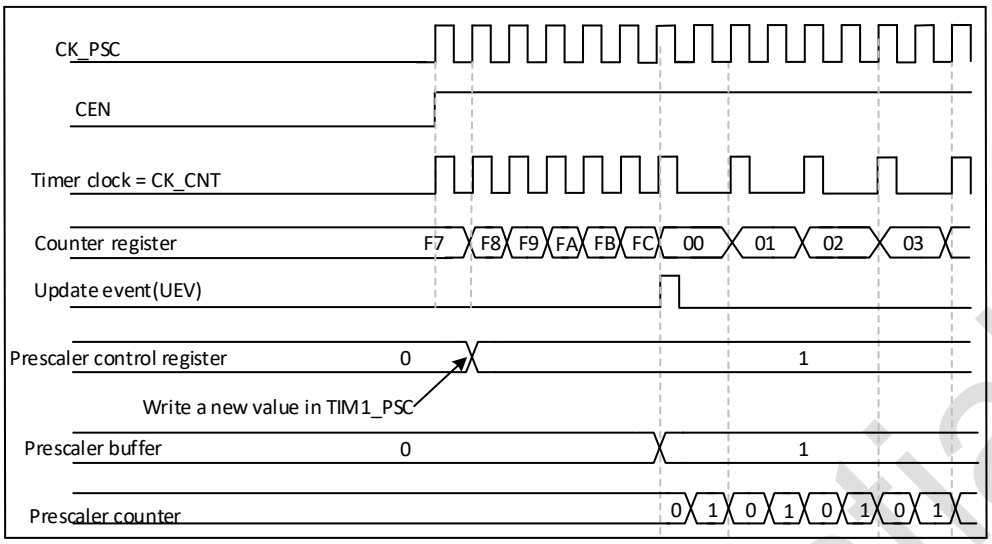


图 21-3 当预分频器的参数从 1 变到 2 时，计数器的时序图

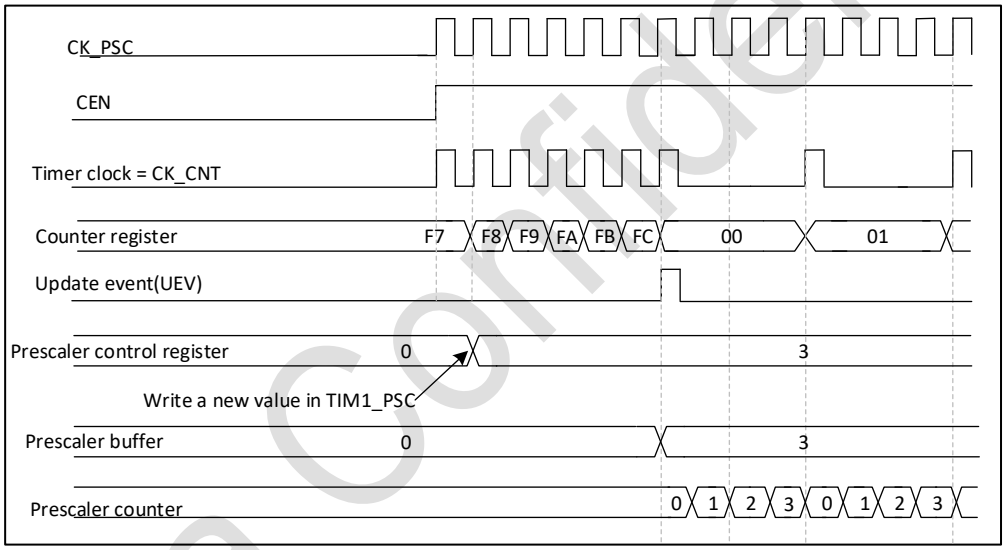


图 21-4 当预分频器的参数从 1 变到 4 时，计数器的时序图

21.4.4 计数器模式

向上计数模式

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

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

通过设置 PWMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清‘0’之前，将不会产生更新事件。即使这样，在应该产生更新事件时，计数器仍会被清‘0’，同时预分频器内部的计数器也被清‘0’(但预分频器的数值不变)。

此外，如果设置了 PWMx_CR1 寄存器中的 URS 位(选择更新请求源)，通过设置 UG 位可以产生一个更新事件 UEV，但不会置起 UIF 标志位(即不会产生中断或 DMA 请求)。这是为了避免在捕获事件时清除计数器，同时产生更新和捕获中断。

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

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

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

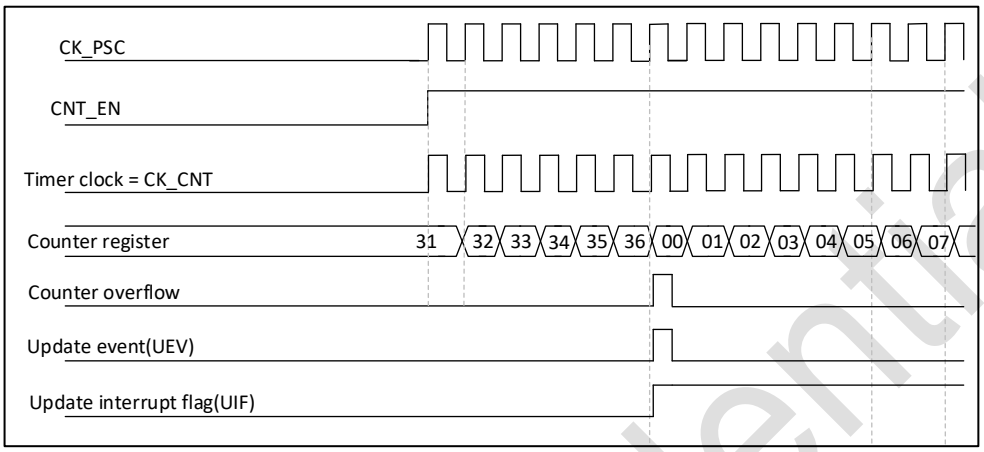


图 21-5 计数器时序图，内部时钟分频因子为 1

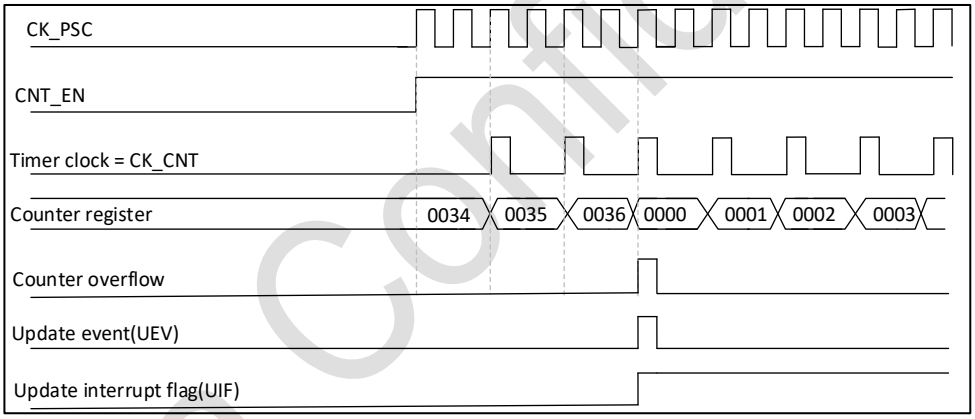


图 21-6 计数器时序图，内部时钟分频因子为 2

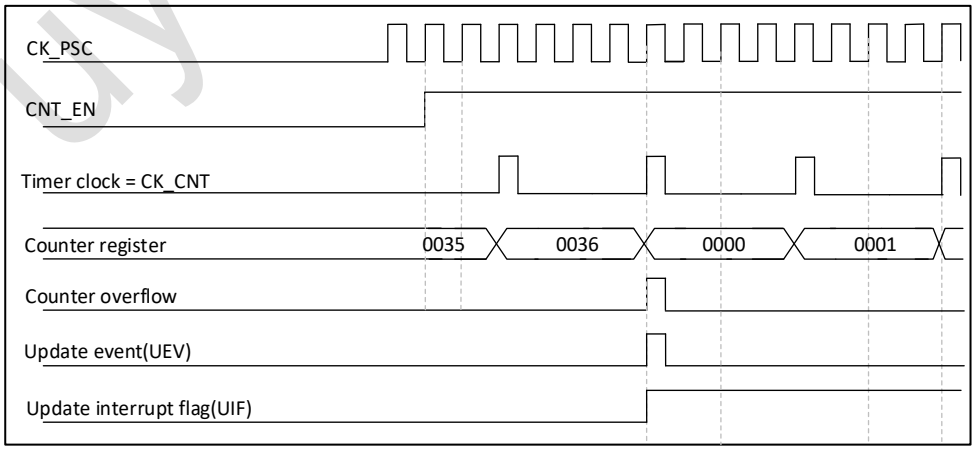


图 21-7 计数器时序图，内部时钟分频因子为 4

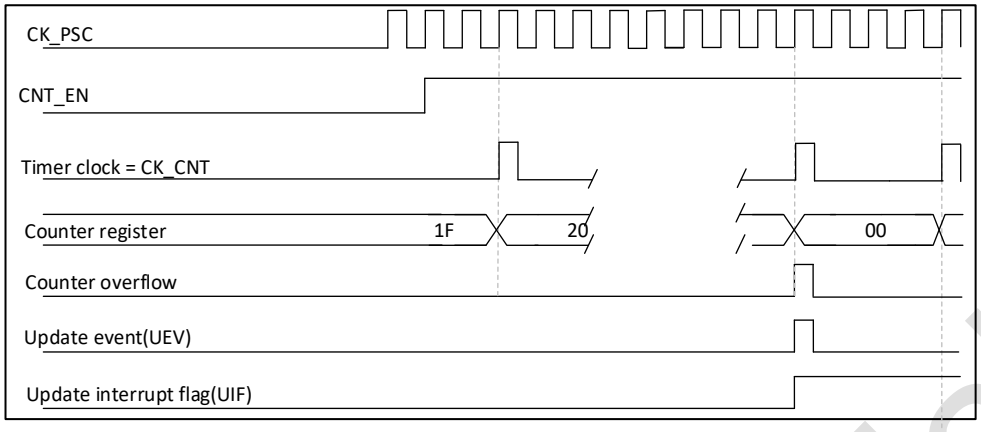


图 21-8 计数器时序图，内部时钟分频因子为 N

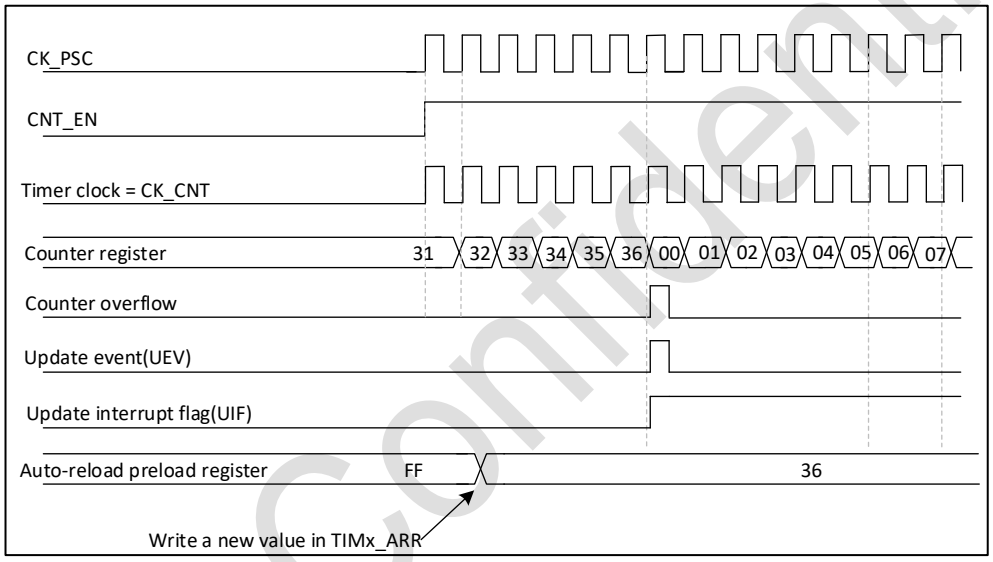


图 21-9 计数器时序图，当 ARPE=0 时的更新事件(没有预装 PWMx_ARR)

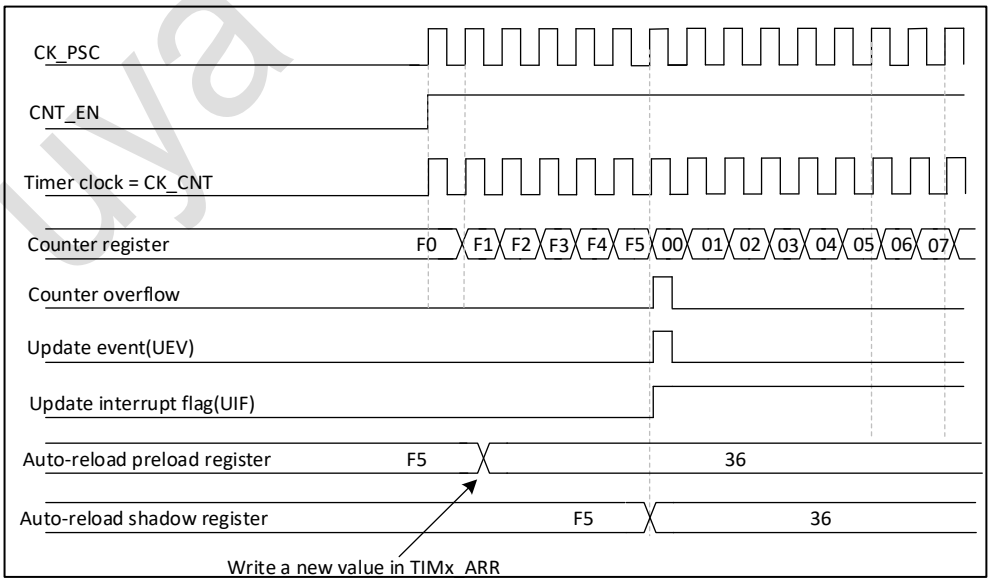


图 21-10 计数器时序图，当 ARPE=1 时的更新事件(预装了 PWMx_ARR)

向下计数模式

在向下计数模式中，计数器从自动加载值(PWM_ARR 的内容)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数下溢事件。每次计数下溢都会产生更新事件。

当发生一个更新事件时，所有以下的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

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

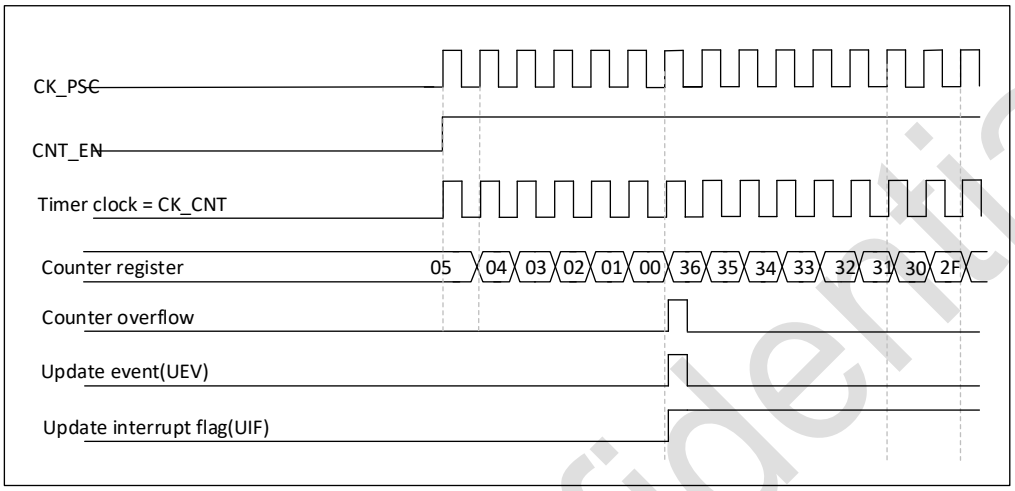


图 21-11 计数器时序图，内部时钟分频因子为 1

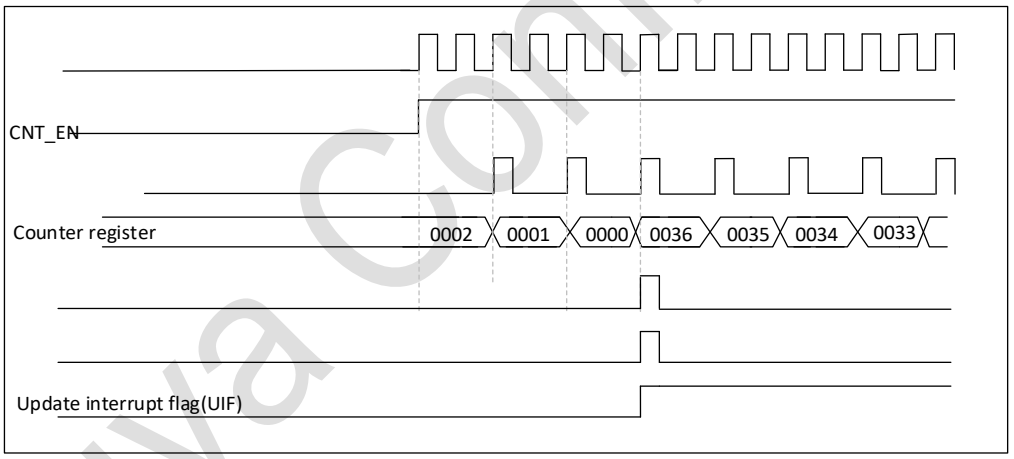


图 21-12 计数器时序图，内部时钟分频因子为 2

中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(PWM_ARR 寄存器)−1，产生一个计数器上溢事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

通过配置 PWM_CR1 寄存器中的 CMS 位不为‘00’可以得到中央对齐模式。

在此模式下，不能写入 PWM_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件。然后，计数器重新从 0 开始计数，预分频器内部计数器也重新从 0 开始计数。

当发生更新事件时，所有的寄存器都被更新：

- 周期影子寄存器被重新置入周期寄存器的值(PWM_ARR 寄存器的内容)。
- 预分频器的缓冲区被置入预分频寄存器的值(PWM_PSC 寄存器的内容)。

以下是一些计数器在不同时钟频率下的操作的例子：

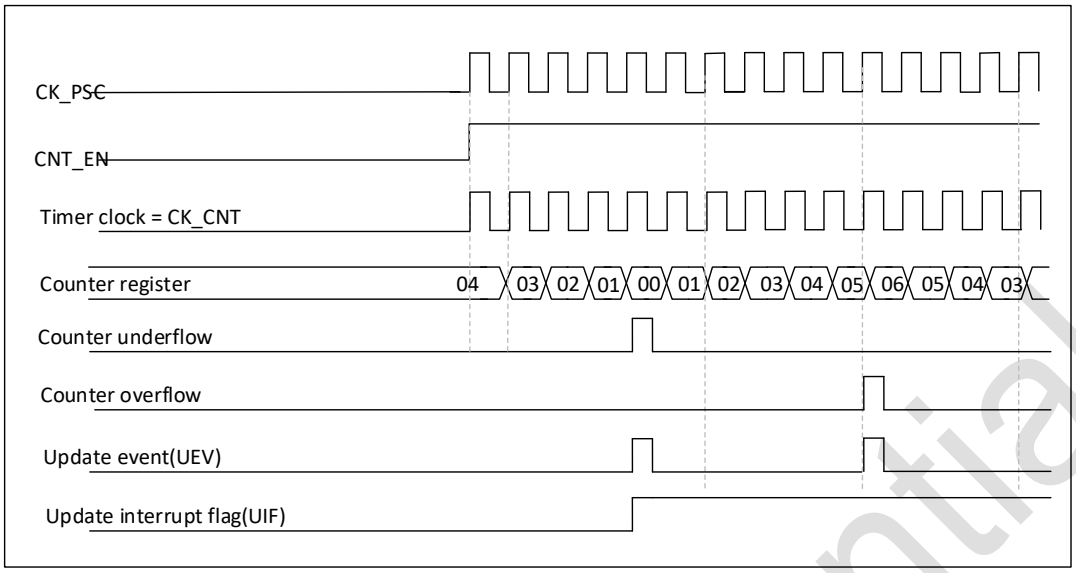


图 21-13 计数器时序图，内部时钟分频因子为 1， PWM_ARR=0x6

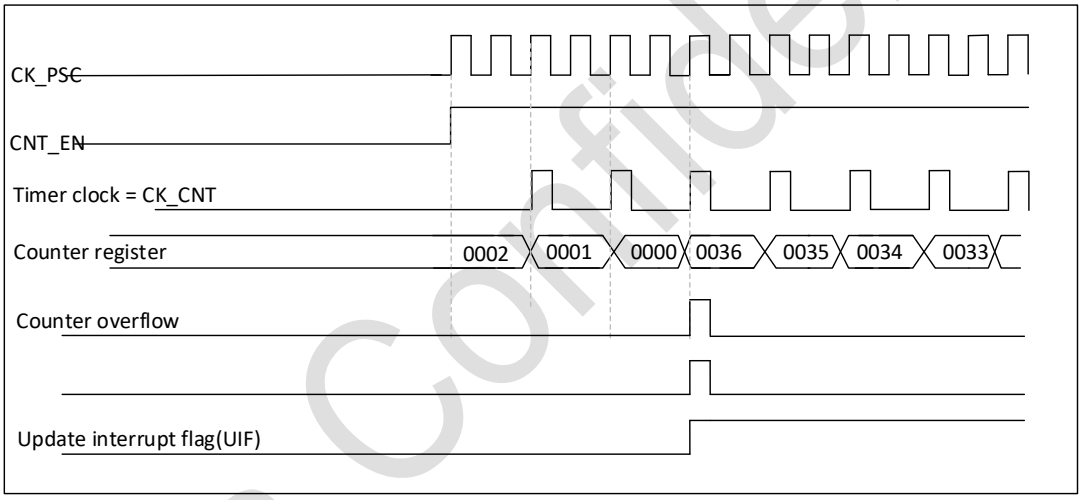


图 21-14 计数器时序图，内部时钟分频因子为 2

21.4.5 时钟选择（仅 PWM1/PWM2/PWM3 支持）

计数器时钟可由下列时钟源提供：

- 内部时钟(pwm_ker_ck)
- 外部时钟模式：外部触发输入 ETR

内部时钟源

只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 pwm_ker_ck 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

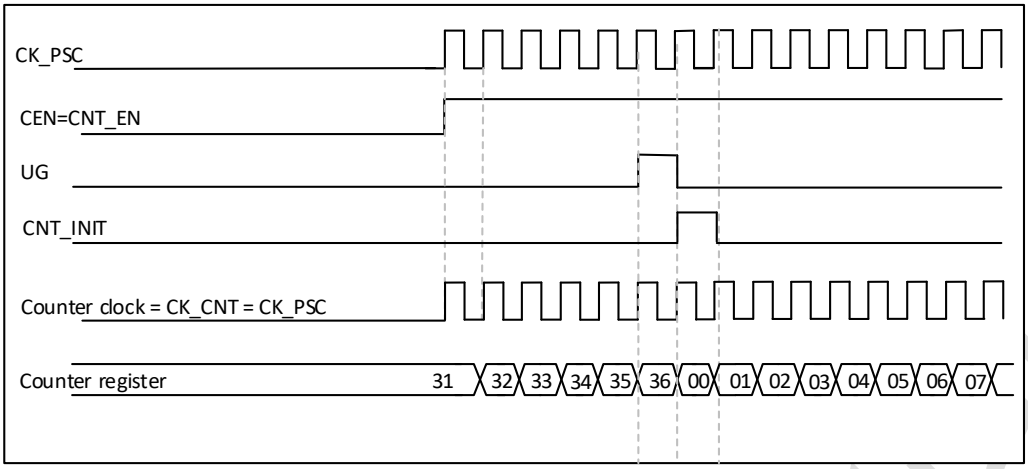


图 21-15 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式

选定此模式的方法为：令 PWM_SMCR 寄存器中的 ECE=1，计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。当 ETRSEL 选择 100/101 时，外部时钟可高于 pwm_ker_ck，因此不能进行分频和滤波操作。

下图是外部触发输入的框图：

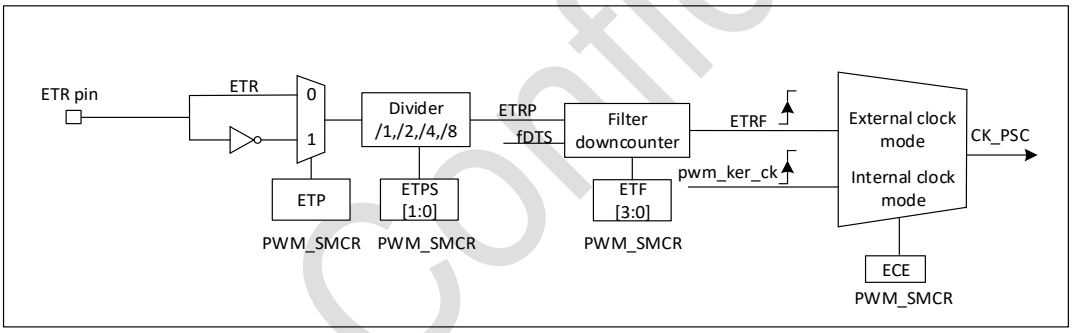


图 21-16 外部触发输入框图

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

1.本例中不需要滤波器，置 PWM_SMCR 寄存器中的 ETF[3:0]=0000；

2.设置预分频器，置 PWM_SMCR 寄存器中的 ETPS[1:0]=01；

3.选择 ETR 输入端的上升沿，置 PWM_SMCR 寄存器中的 ETP=0；

4.开启外部时钟模式，写 PWM_SMCR 寄存器中的 ECE=1；

5.启动计数器，写 PWM_CR1 寄存器中的 CEN=1；

计数器在每 2 个 ETR 上升沿计数一次。

在 ETR 的上升沿和计数器实际时钟之间的延时取决于 ETRP 信号的重新同步电路。如果 ETRSEL=100/101 时，ETR 的上升沿和计数器实际时钟之间没有延时。

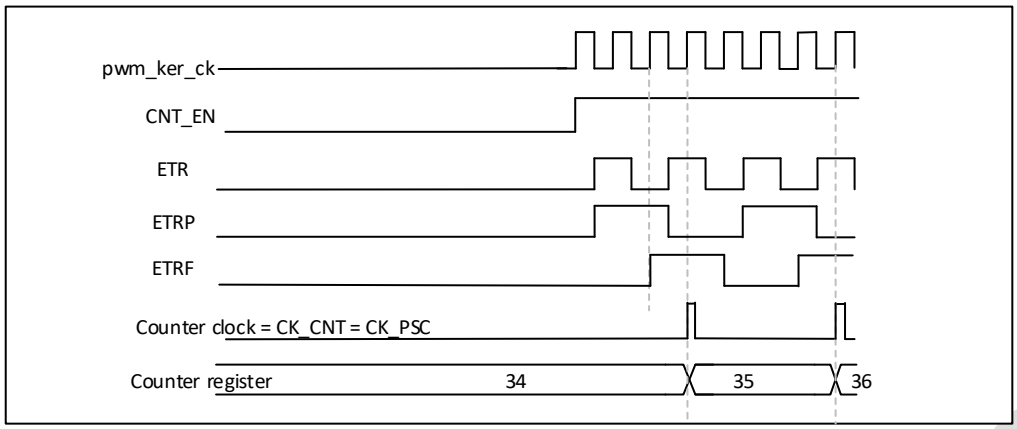


图 21-17 外部时钟模式下的控制电路

21.4.6 输出比较模式

每一个比较通道都是围绕着一个比较寄存器(包含影子寄存器)，包括输出部分(比较器和输出控制)。下面两张图是一个比较通道概览。

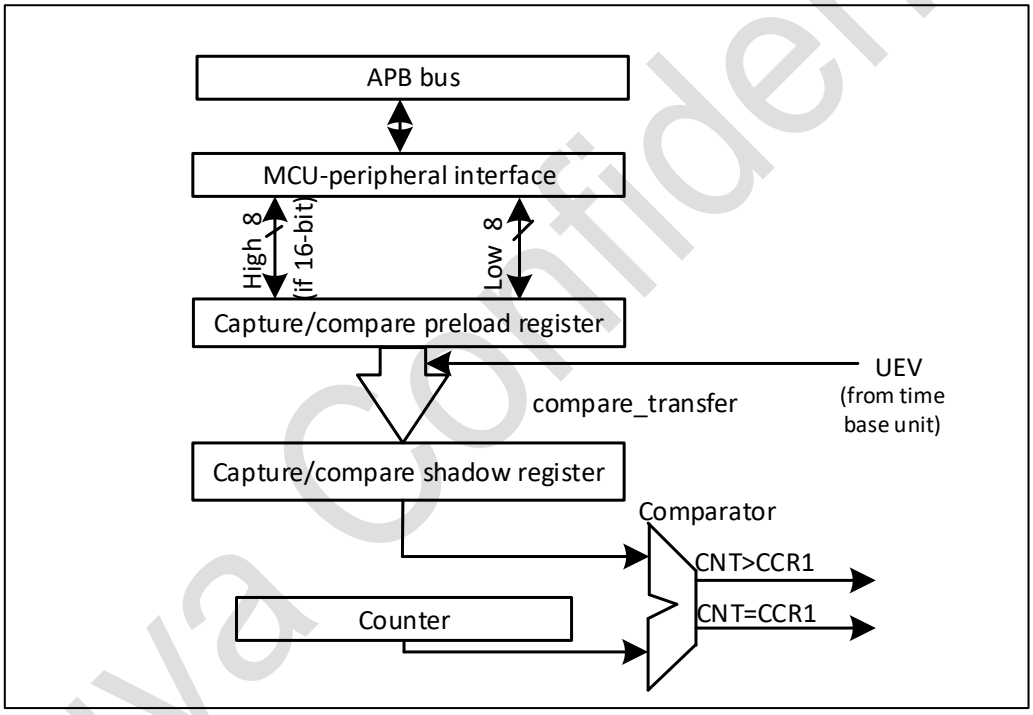


图 21-18 比较通道 1 的主电路

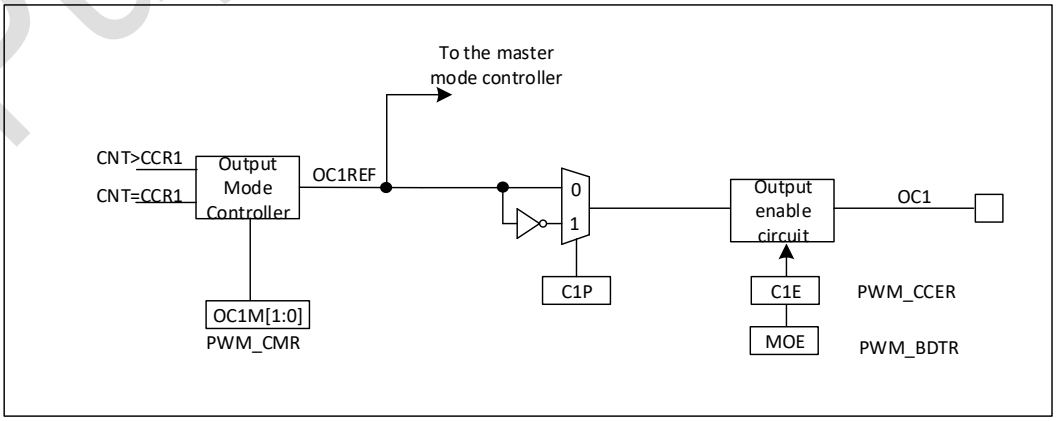


图 21-19 比较通道的输出部分(通道 1 至 4)

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

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

21.4.7 PWM 模式

该模块能产生一个由 PWM_ARR 寄存器确定频率、由 PWM_CCRx 寄存器确定占空比的信号。

在 PWM_CMR 寄存器中的 OCxM 位写入“10”（PWM 模式 1）或“11”（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。

如果要改变 PWM 的周期或者占空比，修改 PWM_ARR 和 PWM_CCRx 寄存器后，在下一个周期新值才会生效。

OCx 的极性可以通过软件在 PWM_CER 寄存器中的 CxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过 PWM_CER 中 CxE 控制。

在 PWM 模式(模式 1 或模式 2)下，PWM_CNT 和 PWM_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 PWM_CCRx≤PWM_CNT 或者 PWM_CNT≤PWM_CCRx。

根据 PWM_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- 向上计数配置

当 PWM_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参看下面是一个 PWM 模式 1 的例子。当 PWM_CNT<PWM_CCRx 时，PWM 为 CxP 定义的有效电平，否则为无效电平。

如果 PWM_CCRx 中的比较值大于自动重装载值(PWM_ARR)，则 PWM 输出保持为’1’。如果比较值为 0，则输出保持为’0’。

下图为 PWM_ARR=8 时边沿对齐的 PWM 波形实例。

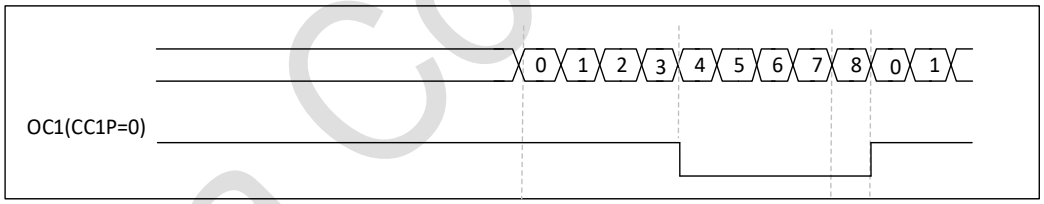


图 21-20 PWM 波形实例

- 向下计数的配置

当 PWM_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 PWM_CNT>PWM_CCRx 时输出为无效电平，否则为有效电平。如果 PWM_CCRx 中的比较值大于 PWM_ARR 中的自动重装载值，则输出保持为’1’。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 PWM_CR1 寄存器中的 CMS 位不为 0 时为中央对齐模式。此时，PWM_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。

下图给出一些中央对齐的 PWM 波形的例子：

- PWM_ARR = 8
- PWM 模式 1
- PWM_CR1 寄存器的 CMS=1

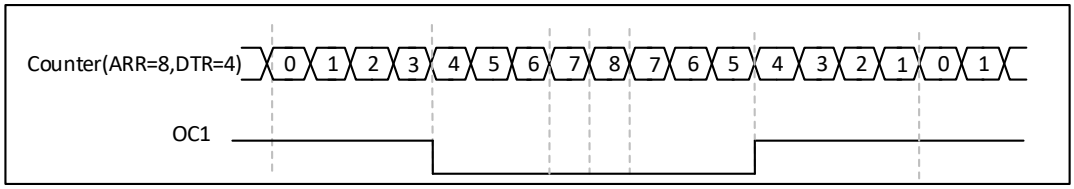


图 21-21 PWM 波形实例

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 PWM_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
  - 如果写入计数器的值大于自动重加载的值(PWM_CNT>PWM_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将 0 或者 PWM_ARR 的值写入计数器，方向被更新。

PWM 移相模式（PWM1_2_3 支持）

PWM 移相模式是指将脉冲信号的相位进行调整。在 PWM 中央对齐模式中，可以通过调整移相功能，将输出信号的相位相对于输入信号进行偏移。这一功能可以用于控制电子设备的输出波形，实现相位差的调整。

在 PWM 中央对齐模式（CMS 不为 00）下使能 PWM 移相模式，通过在各 PWM 周期产生一次中断（根据 PWM_CR1 中的 INTR_SEL 选择中断位置），更新寄存器输出比较值 CCR1、CCR2、CCR3（其中 CCRx[15:0]为向上计数模式的比较值，CCRx[31:16]为向下计数模式的比较值）产生 PWM 移相功能。PWM 移相结构图如下图。

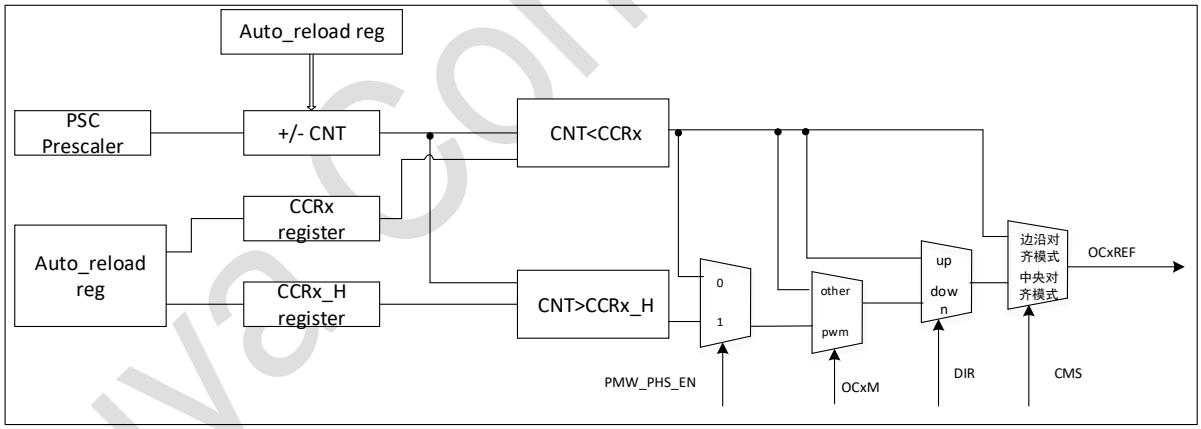


图 21-22 PWM 移相模式

下图为 PWM2 输出模式，使能预装载功能(OCxPE=1)，pwm_intr 在下溢时产生中断（上溢/下溢中断可用 INTR_SEL 选择），将预先设置的 CCRx/CCRx_H 的预装载值在更新事件到来时更新到寄存器 CCRx_SHA/CCRx_H_SHA 中。通过每次更新中断后设置 CCRx_SHA/CCRx_H_SHA 寄存器的值来实现 PWM 移相的功能。

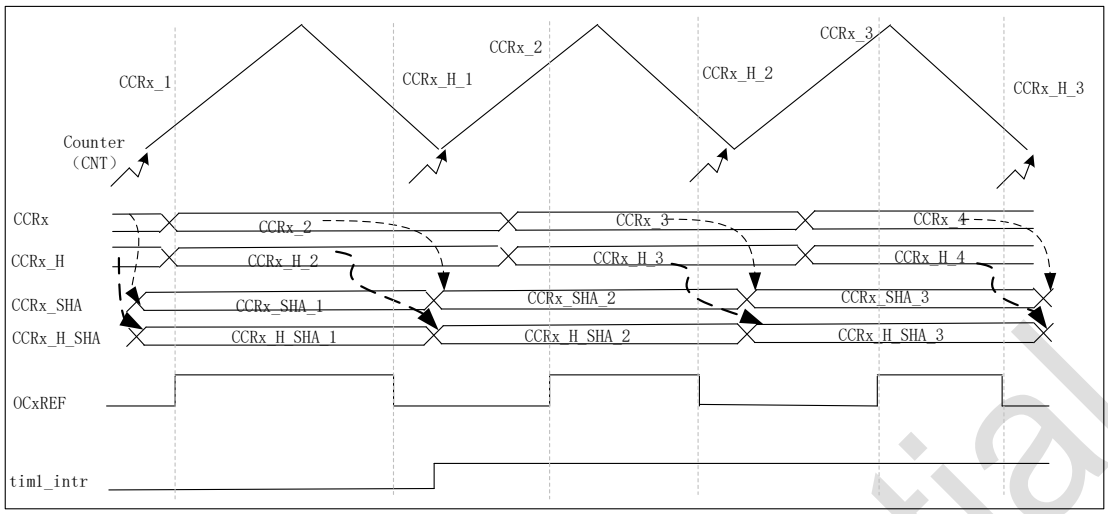


图 21-23 PWM2 输出模式，使能预装载功能(OCxPE=1)

注：PWM 移相模式更新 CCRx_SHA/CCRx_H_SHA 寄存器，有两种方法可以选择：

预装载功能使能时(OCxPE=1)，只有在更新事件到来时才会将预先设置的预装载值更新到 CCRx_SHA /CCRx_H_SHA 寄存器中，如上图。

预装载功能无效（OCxPE=0），CCRx_SHA /CCRx_H_SHA 的值可以实时进行更新（使用此方式时，向上计数模式时不能更改 CCRx_SHA（向下计数模式时不能更改 CCRx_H_SHA），否则，会不满足 PWM 移相功能。如下图为 PWM2 输出模式下，预装载功能无效(OCxPE=0)，pwm_intr 在下溢时产生中断，向上计数时实时更新寄存器 CCRx，此种情况不满足 PWM 移相模式。

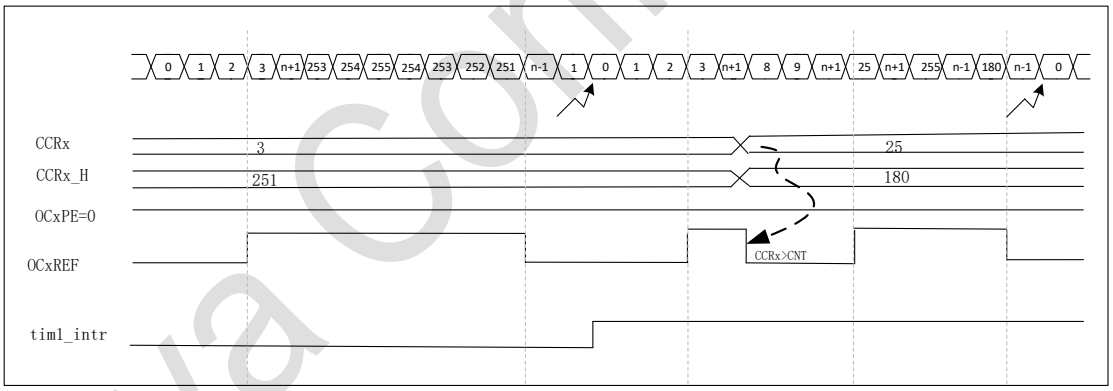


图 21-24 PWM2 输出模式，预装载功能无效(OCxPE=0)

PWM 移相配置步骤：

假定计数器选择内部时钟源:

- 选择中央对齐模式：根据 PWMx_CR1 寄存器中的 CMS 选择中央对齐模式
- 选择预装载使能：置 PWMx_CMRx 寄存器中的 OCxPE=1 使能输出比较 1、2、3 预装载使能。
- 配置比较寄存器 PWMx_CCRx[31:0]，置 UG 位产生一个更新事件，更新比较寄存器值。
- 置 PWMx_CR1 寄存器中的 PWM_PHS_EN=1,PWM 移相使能，此例中选择 ISR_SEL=10 下溢中断。
- 配置 PWMx_ARR 自动重装载寄存器。
- 选择 PWM2 输出模式：置 PWMx_CMRx 中的 OCxM=2’b11。
- 置 PWMx_CR1 寄存器中的 CEN=1 使能计数器，如果设置了 UIE 位，则会产生更新中断。
- 预先配置 PWMx_CCRx，出现更新事件后，预装载值更新到本地寄存器中。

21.4.8 互补输出和死区插入（仅 PWM1/PWM2/PWM3 支持）

PWM 能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

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

同时设置 CxE 和 CxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 8 位的死区发生器 DTG[7:0]。

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

下图显示了插入死区时 OCx 和 OCxN 的输出。(假设 CxP=0、CxNP=0、MOE=1、CxE=1 并且 CxNE=1)

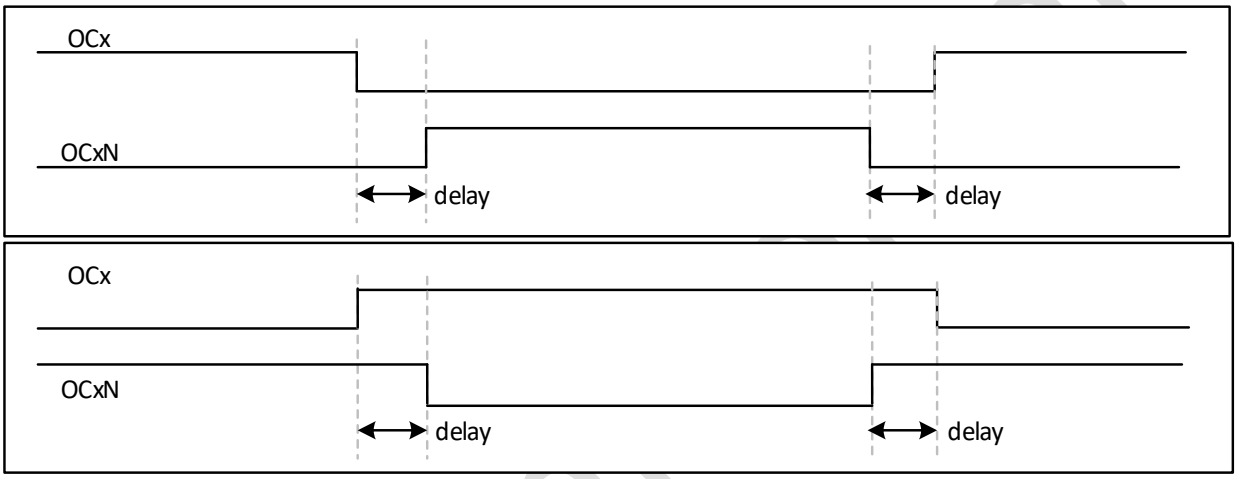


图 21-25 带死区插入的互补输出

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

21.4.9 使用刹车功能（仅 PWM1/PWM2/PWM3 支持）

当使用刹车功能时，依据额外的控制位，输出使能信号和无效电平信号都会被修改。无论什么情况下，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源既可以是刹车输入引脚，或者以下内部源：

- CPU LOCKUP 输出
- PVD 输出
- 由 CSS 监测产生的时钟 failure 事件

系统复位后，刹车电路被禁止，MOE 位为低。设置 PWM_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。

当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

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

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- MOE 位被异步地清除，将输出禁止。这个特性在 MCU 的振荡器关闭时依然有效。
- 如果设置了 PWM_DIER 寄存器中的 BIE 位，当刹车状态标志(PWM_SR 寄存器中的 BIF 位)为'1'时，则产生一个中断。

注： 刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

刹车可以由 BRK 输入产生，它的有效极性是可编程的，且由 PWM_BDTR 寄存器中的 BKE 位开启。

21.5 PWM1/PWM2/PWM3 寄存器描述

21.5.1 PWM 控制寄存器 1（PWM_CR1）

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	PWM_PHS_EN	INTR_SEL[1:0]		Res.	Res.	CKD[1:0]		ARPE	CMS[1:0]		DIR	Res.	URS	UDIS	CEN
	RW	RW	RW			RW	RW	RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:15	Reserved	-	-	保留
14	PWM_PHS_EN	RW	0	PWM 移相使能 0：关 1：开
13:12	INTR_SEL	RW	0	00:上溢或下溢中断 01:上溢中断 10:下溢中断 11:保留
11:10	Reserved	-	-	保留
9:8	CKD[1:0]	RW	2’h0	时钟分频因子 这 2 位定义在定时器时钟(pwm_ker_ck)频率和死区时间和死区发生器与数字滤波器(ETR)所用的采样时钟 (t_{DTS}) 之间的分频比例。 00： t_{DTS} = tpwm_ker_ck 01： t_{DTS} = 2 x tpwm_ker_ck 10： t_{DTS} = 4 x tpwm_ker_ck 11：保留，不要使用这个配置
7	ARPE	RW	0	自动重装载预装载允许位 0： PWM_ARR 寄存器没有缓冲 1： PWM_ARR 寄存器被装入缓冲器
6:5	CMS[1:0]	RW	2’h0	计数模式。 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位,只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位,只在计数器向上计数时被设置。

				11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	计数方向。 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式时，该位为只读。
3	Reserved	-	-	保留
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： - 计数器溢出/下溢 - 设置 UG 位 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： - 计数器溢出/下溢 - 设置 UG 位 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、CCRx)保持它们的值。如果设置了 UG 位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0：禁止计数器 1：开启计数器 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

21.5.2 PWM 控制寄存器 2（PWM_CR2）

偏移地址：0x04

复位值：0x0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETRSEL[2:0]			Res	BKCMP2P	BKCMP1P	BKINP	Res.	Res.	Res.	Res.	Res.	Res.	BKCMP2E	BKCMP1E	BKINE
RW	RW	RW		RW	RW	RW							RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:13	ETRSEL	RW	3'h0	外部时钟源选择（etr_in source selection） 该位段用于选择 ETR 输入源 000：PWM_ETR_LS（低速） 001：comp1 010：comp2 011：保留 100：PLL 101：pwm_etr_io_hs(高速) 其它：保留 注意： （当配置为 100/101 时，触发频率可以超过 PWM PCLK，故无法使能分频和滤波功能，需要保证无毛刺输入） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 1，则这些位不能被修改。
12	Reserved	-	-	保留
11	BKCMP2P	RW	0	brk_cmp2 输入极性。 该位选择 brk_cmp2 输入极性，必须与 BKP 极性位同时配置 0：brk_cmp2 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：brk_cmp2 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
10	BKCMP1P	RW	0	brk_cmp1 输入极性。 该位选择 brk_cmp1 输入极性，必须与 BKP 极性位同时配置 0：brk_cmp1 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：brk_cmp1 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
9	BKINP	RW	0	BKIN 输入极性。 该位选择 BKIN 输入极性的备用功能，必须与 BKP 极性位同时配置 0：BKIN 输入极性不翻转（BKP=0 时低有效，BKP=1 时高有效） 1：BKIN 输入极性翻转（BKP=0 时高有效，BKP=1 时低有效） 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
8:3	Reserved	-	-	保留
2	BKCMP2E	RW	0	brk_cmp2 输入使能。 该位用于刹车输入时使能 brk_cmp2。brk_cmp2 输入与其它刹车源进行或逻辑作为刹车输入。 0：brk_cmp2 输入关闭 1：brk_cmp2 输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改

1	BKCMP1E	RW	0	brk_cmp1 输入使能。 该位用于刹车输入时使能 brk_cmp1。brk_cmp1 输入与其它刹车源进行或逻辑作为刹车输入。 0：brk_cmp1 输入关闭 1：brk_cmp1 输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改
0	BKINE	RW	1	BKIN 输入使能。 该位用于刹车输入时使能 BKIN 的备用功能。BKIN 输入与其它刹车源进行或逻辑作为刹车输入。 0：BKIN 输入关闭 1：BKIN 输入开启 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)被设置为 1，则该位不能被修改。

21.5.3 PWM 从模式控制寄存器（PWM_SMCR）

Address:0x08

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ETP	RW	0	外部触发极性 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
14	ECE	RW	0	外部时钟使能位 该位启用外部时钟模式 0：禁止外部时钟模式； 1：使能外部时钟模式。 计数器由 ETRF 信号上的任意有效边沿驱动。
13:12	ETPS	RW	2’h0	外部触发预分频 外部触发信号 ETRP 的频率必须最多是 CNTCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。 注意：当 ETF 选择 0000 时，不支持预分频。
11:8	ETF	RW	4’h0	外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，异步

				0001：采样频率 $f_{SAMPLING}$=fpwm_ker_ck， N=2 0010：采样频率 $f_{SAMPLING}$=fpwm_ker_ck， N=4 0011：采样频率 $f_{SAMPLING}$=fpwm_ker_ck， N=8 0100：采样频率 $f_{SAMPLING}$=f_{DTS}/2，N=6 0101：采样频率 $f_{SAMPLING}$=f_{DTS}/2，N=8 0110：采样频率 $f_{SAMPLING}$=f_{DTS}/4，N=6 0111：采样频率 $f_{SAMPLING}$=f_{DTS}/4，N=8 1000：采样频率 $f_{SAMPLING}$=f_{DTS}/8，N=6 1001：采样频率 $f_{SAMPLING}$=f_{DTS}/8, N=8 1010：采样频率 $f_{SAMPLING}$=f_{DTS}/16, N=5 1011：采样频率 $f_{SAMPLING}$=f_{DTS}/16，N=6 1100：采样频率 $f_{SAMPLING}$=f_{DTS}/16，N=8 1101：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=5 1110：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=6 1111：采样频率 $f_{SAMPLING}$=f_{DTS}/32，N=8
7:0	Reserved	-	-	保留

21.5.4 PWM DMA/中断使能寄存器 (PWM_DIER)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	OC4DE	OC3DE	OC2DE	OC1DE	UDE	BIE	Res.	Res.	OC4IE	OC3IE	OC2IE	OC1IE	UIE
			RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	OC4DE	RW	0	OC4DE：允许输出比较 4 的 DMA 请求 0：禁止输出比较 4 的 DMA 请求 1：允许输出比较 4 的 DMA 请求
11	OC3DE	RW	0	OC3DE：允许输出比较 3 的 DMA 请求 0：禁止输出比较 3 的 DMA 请求 1：允许输出比较 3 的 DMA 请求
10	OC2DE	RW	0	OC2DE：允许输出比较 2 的 DMA 请求 0：禁止输出比较 2 的 DMA 请求 1：允许输出比较 2 的 DMA 请求
9	OC1DE	RW	0	OC1DE：允许输出比较 1 的 DMA 请求 0：禁止输出比较 1 的 DMA 请求 1：允许输出比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7	BIE	RW	0	BIE：允许刹车中断 0：禁止刹车中断 1：允许刹车中断

6:5	Reserved	-	-	保留
4	OC4IE	RW	0	OC4IE：允许比较输出 4 中断 0：禁止比较输出 4 中断 1：允许比较输出 4 中断
3	OC3IE	RW	0	OC3IE：允许比较输出 3 中断 0：禁止比较输出 3 中断 1：允许比较输出 3 中断
2	OC2IE	RW	0	OC2IE：允许比较输出 1 中断 0：禁止比较输出 2 中断 1：允许比较输出 2 中断
1	OC1IE	RW	0	OC1IE：允许比较输出 1 中断 0：禁止比较输出 1 中断 1：允许比较输出 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断 1：使能更新中断

21.5.5 PWM 状态寄存器 (PWM_SR)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ARROK	PSCOK	CNTOK	CCR4OK	CCR3OK	CCR2OK	CCR1OK	Res.	BIF	Res.	Res.	OC4IF	OC3IF	OC2IF	OC1F	UIF
RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	RC_W0		RC_W0			RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ARROK	RC_W0	0	PWM_ARR 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 ARROK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_ARR 的写操作已成功完成。 该寄存器由软件清零。
14	PSCOK	RC_W0	0	PWM_PSC 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 PSCOK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_PSC 的写操作已成功完成。 该寄存器由软件清零。
13	CNTOK	RC_W0	0	PWM_CNT 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CNTOK 由硬件置位，在配置为异步计数时通知应用程序 APB 总线对 PWM_CNT 的写操作已成功完成。 该寄存器由软件清零。

12	CCR4OK	RC_W0	0	PWM_CCR4 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR4OK 由硬件置位,在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR4 的写操作已成功完成。 该寄存器由软件清零。
11	CCR3OK	RC_W0	0	PWM_CCR3 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR3OK 由硬件置位,在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR3 的写操作已成功完成。 该寄存器由软件清零。
10	CCR2OK	RC_W0	0	PWM_CCR2 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR2OK 由硬件置位,在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR2 的写操作已成功完成。 该寄存器由软件清零。
9	CCR1OK	RC_W0	0	PWM_CCR1 寄存器更新完成标志，仅在配置为异步计数时有效（ETRSEL 配置为 100/101）。 CCR1OK 由硬件置位,在配置为异步计数时通知应用程序 APB 总线对 PWM_CCR1 的写操作已成功完成。 该寄存器由软件清零。
8	Reserved	-	-	保留
7	BIF	RC_W0	0	刹车中断标记 一旦刹车输入有效，由硬件对该位置 1。如果刹车输入无效，则该位可由软件清 0。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
6:5	Reserved	-	-	保留
4	OC4IF	RC_W0	0	输出比较 4 中断标记 参考 OC1IF 描述
3	OC3IF	RC_W0	0	输出比较 3 中断标记 参考 OC1IF 描述
2	OC2IF	RC_W0	0	输出比较 2 中断标记 参考 OC1IF 描述
1	OC1IF	RC_W0	0	输出比较 1 中断标记 当计数器值与比较值匹配时该位由硬件置 1，它由软件清 0。 0：无匹配发生； 移相无效时： 当 PWM1_CCR1 大于 PWM1_ARR 的内容，OC1IF 位在计数器溢出（在递增计数和递增/递减计数模式下）或下溢（在下降计数模式）。在中心对齐模式下，有 3 种可能的标志设置选项，有关完整描述，请参阅 PWM1_CR1 寄存器中的 CMS 位. 移相有效时：

				中央对齐模式向上计数 PWM1_CNT 的值与 PWM1_CCR1 的值匹配，中央对齐模式向下计数 PWM1_CNT 的值与 PWM1_CCR1_H 的值匹配。 当 PWM1_CCR1 大于 PWM1_ARR 的内容，OC1IF 位在计数器溢出（在递增计数）或下溢（在下降计数模式）。 或者中央对齐模式下当 PWM1_CCR1 的值大于 PWM1_ARR 向上计数上溢时 OC1IF 置 1(CMS=2,CMS=3)，中央对齐模式下当 PWM1_CCR1_H 的值大于 PWM1_ARR 向下计数下溢时 OC1IF 置 1(CMS=1,CMS=3)。
0	UIF	RC_W0	0	更新中断标记 当产生更新事件时该位由硬件置 1。它由软件清 0。 0：无更新事件产生 1：更新事件产生(计数器上溢或者下溢或者置位 UG)。该位由硬件置 1

21.5.6 PWM 事件产生寄存器 1(PWM_EGR)

偏移地址：0x14

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	Reserved
0	UG	W	0	产生更新事件 该位由软件置‘1’，由硬件自动清‘0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清‘0’(但是预分频系数不变)。

21.5.7 PWM 输出比较模式寄存器 1(PWM_CMR)

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	OC4PE	OC3PE	OC2PE	OC1PE	OC4M[1:0]		OC3M[1:0]		OC2M[1:0]		OC1M[1:0]	
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	OC4PE	RW	0	输出比较 4 预装载使能
10	OC3PE	RW	0	输出比较 3 预装载使能
9	OC2PE	RW	0	输出比较 2 预装载使能
8	OC1PE	RW	0	输出比较 1 预装载使能

				0：禁止 PWM_CCR1 寄存器的预装载功能，可随时写入 PWM1_CCR1 寄存器，且新值马上起作用。 1: 开启 PWM_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，PWM_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1: 一旦 LOCK 级别设为 3(PWM_BDTR 寄存器中的 LOCK 位)则该位不能被修改。
7:6	OC4M[1:0]	RW	2'h0	OC4 PWM 模式。参考 OC1M 的描述。（不支持移相模式）
5:4	OC3M[1:0]	RW	2'h0	OC3 PWM 模式。参考 OC1M 的描述。（不支持移相模式）
3:2	OC2M[1:0]	RW	2'h0	OC2 PWM 模式。参考 OC1M 的描述。（支持移相模式）
1:0	OC1M[1:0]	RW	2'h0	OC1 PWM 模式 10： PWM 模式 1 移相模式： - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1_H 通道 1 为无效电平，PWM_CNT≤PWM_CCR1_H 通道 1 为有效电平 非移相模式 - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为无效电平，否则为有效电平。 11： PWM 模式 2 移相模式： - 在向上计数时，一旦 PWM_CNT<PWM _CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 PWM_CNT>PWM_ CCR1_H 时通道 1 为有效电平，PWM_CNT≤PWM_CCR1_H 通道 1 为无效电平。 非移相模式 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为有效电平，否则为无效电平。 其它：保留 注：有效电平由 PWM_CER 寄存器配置，无效电平为取反。

21.5.8 PWM 输出比较使能寄存器 (PWM_CER)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	C4P	C4E	Res.	Res.	C3P	C3E	C2NP	C2NE	C2P	C2E	C1NP	C1NE	C1P	C1E
		RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	C4P	RW	0	OC4 输出极性。参考 C1P 的描述。

12	C4E	RW	0	OC4 输出使能。参考 C1E 的描述。
11:10	Reserved	-	-	保留
9	C3P	RW	0	OC3 输出极性。参考 C1P 的描述。
8	C3E	RW	0	OC3 输出使能。参考 C1E 的描述。
7	C2NP	RW	0	OC2 互补输出极性。参考 C1NP 的描述。
6	C2NE	RW	0	OC2 互补输出使能。参考 C1NE 的描述。
5	C2P	RW	0	OC2 输出极性。参考 C1P 的描述。
4	C2E	RW	0	OC2 输出使能。参考 C1E 的描述。
3	C1NP	RW	0	OC1 互补输出极性 0：OC1N 高电平有效 1：OC1N 低电平有效 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 则该位不能被修改。
2	C1NE	RW	0	OC1 互补输出使能 0：关闭 - OC1N 禁止输出，输出电平为高阻。 1：开启 - OC1N 信号输出到对应的输出引脚，其输出电平为 OC1 的互补。
1	C1P	RW	0	OC1 输出极性 0：OC1 高电平有效 1：OC1 低电平有效 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 则该位不能被修改。
0	C1E	RW	0	OC1 输出使能 0：关闭 - OC1 禁止输出，输出电平为高阻。 1: 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 C1P 的值。

MOE	CCxE	CCxNE	OCx output state	OCxN output state
1	0	0	输出禁止(与定时器断开)，OCx=0， OCx_EN=0	输出禁止(与定时器断开)， OCxN=0, OCxN_EN=0
	0	1	输出禁止(与定时器断开)，OCx=0， OCx_EN=0	OCxREF + Polarity OCxN=OCxREF 异或 CCxNP， OCxN_EN=1
	1	0	OCxREF + Polarity OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(与定时器断开)， OCxN=0, OCxN_EN=0
	1	1	OCREF + Polarity + dead-time OCx_EN=1	OCREF 的互补（not OCREF）+ Polarity + dead-time OCxN_EN=1
0	X	X	输出禁止(与定时器断开)	

21.5.9 PWM 计数寄存器(PWM_CNT)

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	16'h0	计数器的值

21.5.10 PWM 预分频器（PWM_PSC）

偏移地址：0x28

复位值： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16'h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 tpwm_psc_ck/(PSC[15:0]+1)。 PSC 包含每次当更新事件产生时，装入当前预分频器寄存器的值

21.5.11 PWM 自动重载寄存器 (PWM_ARR)

偏移地址：0x2c

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'hFFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

21.5.12 PWM 比较寄存器 1(PWM_CCR1)

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1_H[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	Name	R/W	Reset Value	Function
31:16	CCR1_H[31:16]	RW	16'h0	输出比较 1 寄存器的值 若 CC1 通道配置为中央对齐 pwm 模式输出： CCR1_H 包含了装入向下计数比较 1 寄存器的值 (预装载值)。
15:0	CCR1[15:0]	RW	16'h0	比较 1 的值 CCR1 包含了装入当前比较 1 寄存器的值（预装载值）。

				只有当更新事件发生时，此预装载值才装入当前比较 1 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC1 端口上输出 PWM 信号。

21.5.13 PWM 比较寄存器 2(PWM_CCR2)

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR2_H[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	Name	R/W	Reset Value	Function
31:16	CCR2_H[31:16]	RW	16'h0	输出比较 2 寄存器的值 若 CC2 通道配置为中央对齐 pwm 模式输出： CCR2_H 包含了装入向下计数比较 2 寄存器的值（预装载值）。
15:0	CCR2[15:0]	RW	16'h0	比较 2 的值 CCR2 包含了装入当前比较 2 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 1 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC2 端口上输出 PWM 信号。。

21.5.14 PWM 比较寄存器 3(PWM_CCR3)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR3[15:0]	RW	16'h0	比较 3 的值 CCR3 包含了装入当前比较 3 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 3 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC3 端口上输出 PWM 信号。

21.5.15 PWM 比较寄存器 4(PWM_CCR4)

偏移地址：0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4[15:0]	RW	16'h0	比较 4 的值 CCR4 包含了装入当前比较 4 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 3 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC4 端口上输出 PWM 信号。

21.5.16 PWM 刹车和死区寄存器(PWM_BDTR)

偏移地址：0x44

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	Res.	Res.	LOCK[1:0]		DTG[7:0]							
RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	MOE	RW	0	刹车主输出使能 一旦刹车输入有效，该位被硬件异步清 0。 0：禁止 OCx 和 OCxN 输出或强制为无效状态； 1: 如果设置了相应的使能位 (PWM_CER 寄存器的 CxE、CxNE 位），则开启 OCx 和 OCxN 输出。
14	AOE	RW	0	自动输出使能 0： MOE 只能被软件置 1； 1： MOE 能被软件置 1 或在下一个更新事件自动置 1（如果刹车输入无效）。 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。
13	BKP	RW	0	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。
12	BKE	RW	0	刹车功能使能 0：禁止刹车输入； 1：开启刹车输入。
11:10	Reserved	-	-	保留
9:8	LOCK[1:0]	RW	2'h0	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 PWM_BDTR 寄存器的 DTG/BKE/BKP/AOE 位 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CxP/CNxP 极性位 11: 锁定级别 3,不能写入锁定级别 2 中的各位,也不能写入 CxM 控制位

				注: 在系统复位后,只能写一次 LOCK 位,一旦写入 PWM_BDTR 寄存器，则其内容冻结直至复位。
7:0	DTG[7:0]	RW	8'h0	死区时间设置。 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间: DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg， Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS; DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0]) × Tdtg， Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为: 0 到 15875ns，若步长时间为 125ns； 16μs 到 31750ns，若步长时间为 250ns； 32μs 到 63μs，若步长时间为 1μs； 64μs 到 126μs，若步长时间为 2μs； 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则这些位不能被修改。

21.5.17 PWM DMA 控制寄存器（PWM_DCR）

偏移地址：0x48

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5'h0	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度(当对 PWM_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节: 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 例：我们考虑这样的传输： DBL=7 字节， DBA=PWM_CR1 - 如果 DBL=7 字节， DBA=PWM_CR1 表示待传输数据的地址，那么传输的地址由下式给出: (PWM_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(PWM_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(PWM_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况:

				- 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	-	-	保留
4:0	DBA	RW	5'h0	DMA 基地址 这些位定义了 DMA 在连续模式下的基地址(当对 PWM_DMAR 寄存器进行读或写时)， DBA 定义为从 PWM_CR1 寄存器所在地址开始的偏移量： 00000：PWM_CR1， 00001：PWM_CR2， 00010：PWM_SMCR， 00011：PWM_DIER， 00100：PWM_SR， ……..

21.5.18 PWM 连续模式的 DMA 地址（PWM_DMAR）

偏移地址：0x4C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	32'h0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 PWM_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： PWM_CR 地址 + (DBA + DMA 索引)x4 其中： "PWM_CR 地址"是控制寄存器 1(PWM_CR)所在的地址； "DBA"是 PWM_DCR 寄存器中定义的基地址； "DMA 索引"是由 DMA 自动控制的偏移量，它取决于 PWM_DCR 寄存器中定义的 DBL。

21.6 PWM4 寄存器描述

21.6.1 PWM 控制寄存器 1（PWM_CR1）

偏移地址：0x00

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ARPE	CMS[1:0]		DIR	Res.	URS	UDIS.	CEN
								RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	ARPE	RW	0	自动重装载预装载允许位 0： PWM_ARR 寄存器没有缓冲 1： PWM_ARR 寄存器被装入缓冲器
6:5	CMS[1:0]	RW	2'h0	计数模式。 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	DIR	RW	0	计数方向。 0：计数器向上计数 1：计数器向下计数 注：当计数器配置为中央对齐模式时，该位为只读。
3	Reserved	-	-	保留
2	URS	RW	0	更新请求源 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： − 计数器溢出/下溢 − 设置 UG 位 1: 如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
1	UDIS	RW	0	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： − 计数器溢出/下溢 − 设置 UG 位 具有缓存的寄存器被装入它们的预装载值。 (译注：更新影子寄存器) 1: 禁止 UEV。不产生更新事件，影子寄存器(ARR、 PSC、CCRx)保持它们的值。如果设置了 UG 位，则计数器和预分频器被重新初始化。
0	CEN	RW	0	允许计数器 0： 禁止计数器 1： 开启计数器

				注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。

21.6.2 PWM DMA/中断使能寄存器 (PWM_DIER)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	OC4DE	OC3DE	OC2DE	OC1DE	UDE	Res.	Res.	Res.	OC4IE	OC3IE	OC2IE	OC1IE	UIE
			RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	OC4DE	RW	0	OC4DE：允许输出比较 4 的 DMA 请求 0：禁止输出比较 4 的 DMA 请求 1：允许输出比较 4 的 DMA 请求
11	OC3DE	RW	0	OC3DE：允许输出比较 3 的 DMA 请求 0：禁止输出比较 3 的 DMA 请求 1：允许输出比较 3 的 DMA 请求
10	OC2DE	RW	0	OC2DE：允许输出比较 2 的 DMA 请求 0：禁止输出比较 2 的 DMA 请求 1：允许输出比较 2 的 DMA 请求
9	OC1DE	RW	0	OC1DE：允许输出比较 1 的 DMA 请求 0：禁止输出比较 1 的 DMA 请求 1：允许输出比较 1 的 DMA 请求
8	UDE	RW	0	UDE：允许更新的 DMA 请求 0：禁止更新的 DMA 请求 1：允许更新的 DMA 请求
7:5	Reserved	-	-	保留
4	OC4IE	RW	0	OC4IE：允许比较输出 4 中断 0：禁止比较输出 4 中断 1：允许比较输出 4 中断
3	OC3IE	RW	0	OC3IE：允许比较输出 3 中断 0：禁止比较输出 3 中断 1：允许比较输出 3 中断
2	OC2IE	RW	0	OC2IE：允许比较输出 1 中断 0：禁止比较输出 2 中断 1：允许比较输出 2 中断
1	OC1IE	RW	0	OC1IE：允许比较输出 1 中断 0：禁止比较输出 1 中断 1：允许比较输出 1 中断
0	UIE	RW	0	更新中断使能 0：禁止更新中断

				1：使能更新中断

21.6.3 PWM 状态寄存器 (PWM_SR)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC4IF	OC3IF	OC2IF	OC1F	UIF
											RC_W0	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	OC4IF	RC_W0	0	输出比较 4 中断标记 参考 OC1IF 描述
3	OC3IF	RC_W0	0	输出比较 3 中断标记 参考 OC1IF 描述
2	OC2IF	RC_W0	0	输出比较 2 中断标记 参考 OC1IF 描述
1	OC1IF	RC_W0	0	输出比较 1 中断标记 当计数器值与比较值匹配时该位由硬件置 1，它由软件清 0。 0：无匹配发生； 移相无效时： 当 PWM1_CCR1 大于 PWM1_ARR 的内容，OC1IF 位在计数器溢出（在递增计数和递增/递减计数模式下）或下溢（在下降计数模式）。在中心对齐模式下，有 3 种可能的标志设置选项，有关完整描述，请参阅 PWM1_CR1 寄存器中的 CMS 位。 移相有效时： 中央对齐模式向上计数 PWM1_CNT 的值与 PWM1_CCR1 的值匹配，中央对齐模式向下计数 PWM1_CNT 的值与 PWM1_CCR1_H 的值匹配。 当 PWM1_CCR1 大于 PWM1_ARR 的内容，OC1IF 位在计数器溢出（在递增计数）或下溢（在下降计数模式）。 或者中央对齐模式下当 PWM1_CCR1 的值大于 PWM1_ARR 向上计数上溢时 OC1IF 置 1(CMS=2,CMS=3)，中央对齐模式下当 PWM1_CCR1_H 的值大于 PWM1_ARR 向下计数下溢时 OC1IF 置 1(CMS=1,CMS=3)。
0	UIF	RC_W0	0	更新中断标记 当产生更新事件时该位由硬件置 1。它由软件清 0。 0：无更新事件产生； 1：更新事件产生(计数器上溢或者下溢或者置位 UG)。该位由硬件置 1：

21.6.4 PWM 事件产生寄存器 1(PWM_EGR)

偏移地址：0x14

复位值：0x0000

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

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	UG	W	0	产生更新事件 该位由软件置‘1’，由硬件自动清‘0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清‘0’(但是预分频系数不变)。

21.6.5 PWM 输出比较模式寄存器 1(PWM_CMR)

偏移地址：0x18

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	OC4PE	OC3PE	OC2PE	OC1PE	OC4M[1:0]		OC3M[1:0]		OC2M[1:0]		OC1M[1:0]	
				RW	RW	RW	RW	RW		RW		RW		RW	

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	OC4PE	RW	0	输出比较 4 预装载使能
10	OC3PE	RW	0	输出比较 3 预装载使能
9	OC2PE	RW	0	输出比较 2 预装载使能
8	OC1PE	RW	0	输出比较 1 预装载使能 0：禁止 PWM_CCR1 寄存器的预装载功能，可随时写入 PWM1_CCR1 寄存器，且新值马上起作用。 1: 开启 PWM_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，PWM_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 注 1: 一旦 LOCK 级别设为 3(PWM_BDTR 寄存器中的 LOCK 位)则该位不能被修改。
7:6	OC4M[1:0]	RW	2’h0	OC4 PWM 模式。参考 OC1M 的描述。（不支持移相模式）
5:4	OC3M[1:0]	RW	2’h0	OC3 PWM 模式。参考 OC1M 的描述。（不支持移相模式）
3:2	OC2M[1:0]	RW	2’h0	OC2 PWM 模式。参考 OC1M 的描述。（支持移相模式）
1:0	OC1M[1:0]	RW	2’h0	OC1PWM 模式 10：PWM 模式 1 移相模式： - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦

				PWM_CNT>PWM_CCR1_H 通道 1 为无效电平，PWM_CNT≤PWM_CCR1_H 通道 1 为有效电平 非移相模式 - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为无效电平，否则为有效电平。 11：PWM 模式 2 移相模式： - 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1_H 时通道 1 为有效电平，PWM_CNT≤PWM_CCR1_H 通道 1 为无效电平。 非移相模式 在向上计数时，一旦 PWM_CNT<PWM_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 PWM_CNT>PWM_CCR1 时通道 1 为有效电平，否则为无效电平。 其它:保留 注：有效电平由 PWM_CER 寄存器配置，无效电平为取反。

21.6.6 PWM 输出比较使能寄存器 (PWM_CER)

偏移地址：0x20

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	C4P	C4E	Res.	Res.	C3P	C3E	Res.	Res.	C2P	C2E	Res.	Res.	C1P	C1E
		RW	RW			RW	RW			RW	RW			RW	RW

Bit	Name	R/W	Reset Value	Function
31:14	Reserved	-	-	保留
13	C4P	RW	0	OC4 输出极性。参考 C1P 的描述。
12	C4E	RW	0	OC4 输出使能。参考 C1E 的描述。
11:10	Reserved	-	-	保留
9	C3P	RW	0	OC3 输出极性。参考 C1P 的描述。
8	C3E	RW	0	OC3 输出使能。参考 C1E 的描述。
7:6	Reserved	-	-	保留
5	C2P	RW	0	OC2 输出极性。参考 C1P 的描述。
4	C2E	RW	0	OC2 输出使能。参考 C1E 的描述。
3:2	Reserved	-	-	保留
1	C1P	RW	0	OC1 输出极性 0：OC1 高电平有效 1：OC1 低电平有效 注：一旦 LOCK 级别(PWM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 则该位不能被修改。
0	C1E	RW	0	OC1 输出使能

				0：关闭 - OC1 禁止输出，输出电平为高阻。 1: 开启 - OC1 信号输出到对应的输出引脚，其输出电平依赖于 C1P 的值。

MOE	CCxE	CCxNE	OCx output state	OCxN output state
1	0	0	输出禁止(与定时器断开)，OCx=0， OCx_EN=0	输出禁止(与定时器断开)， OCxN=0, OCxN_EN=0
	0	1	输出禁止(与定时器断开)，OCx=0， OCx_EN=0	OCxREF + Polarity OCxN=OCxREF 异或 CCxNP， OCxN_EN=1
	1	0	OCxREF + Polarity OCx=OCREF 异或 CCxP, OCx_EN=1	输出禁止(与定时器断开)， OCxN=0, OCxN_EN=0
	1	1	OCREF + Polarity + dead-time OCx_EN=1	OCREF 的互补（not OCREF）+ Polarity + dead-time OCxN_EN=1
0	X	X	输出禁止(与定时器断开)	

21.6.7 PWM 计数寄存器(PWM_CNT)

偏移地址：0x24

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	16’h0	计数器的值

21.6.8 PWM 预分频器（PWM_PSC）

偏移地址：0x28

复位值： 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PSC	RW	16’h0	预分频器的值 (Prescaler value) 计数器的时钟频率(CK_CNT)等于 tpwm_psc_ck/(PSC[15:0]+1)。 PSC 包含每次当更新事件产生时，装入当前预分频器寄存器的值

21.6.9 PWM 自动重载寄存器 (PWM_ARR)

偏移地址：0x2C

复位值：0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'hFFFF	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

21.6.10 PWM 比较寄存器 1(PWM_CCR1)

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR1[15:0]	RW	16'h0	比较 1 的值 CCR1 包含了装入当前比较 1 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 1 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC1 端口上输出 PWM 信号。

21.6.11 PWM 比较寄存器 2(PWM_CCR2)

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR2[15:0]	RW	16'h0	比较 2 的值 CCR2 包含了装入当前比较 2 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 1 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC2 端口上输出 PWM 信号。。

21.6.12 PWM 比较寄存器 3(PWM_CCR3)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR3[15:0]	RW	16'h0	比较 3 的值 CCR3 包含了装入当前比较 3 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 3 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC3 端口上输出 PWM 信号。

21.6.13 PWM 比较寄存器 4(PWM_CCR4)

偏移地址：0x40

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CCR4[15:0]	RW	16'h0	比较 4 的值 CCR4 包含了装入当前比较 4 寄存器的值（预装载值）。 只有当更新事件发生时，此预装载值才装入当前比较 3 寄存器中。 当前比较寄存器包含了与计数器 PWM_CNT 比较的值，并且在 OC4 端口上输出 PWM 信号。

21.6.14 PWM DMA 控制寄存器（PWM_DCR)

偏移地址：0x48

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:8	DBL	RW	5'h0	DMA 连续传送长度 这些位定义了 DMA 在连续模式下的传送长度(当对 PWM_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 例：我们考虑这样的传输： DBL=7 字节， DBA=PWM_CR1

				- 如果 DBL=7 字节， DBA=PWM_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (PWM_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(PWM_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传输将发生在从地址(PWM_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传输的数据宽度。
7:5	Reserved	-	-	保留
4:0	DBA	RW	5'h0	DMA 基地址 这些位定义了 DMA 在连续模式下的基地址(当对 PWM_DMAR 寄存器进行读或写时)， DBA 定义为从 PWM_CR1 寄存器所在地址开始的偏移量： 00000：PWM_CR1， 00001：PWM_CR2， 00010：PWM_SMCR， 00011：PWM_DIER， 00100：PWM_SR， ……..

21.6.15 PWM 连续模式的 DMA 地址（PWM_DMAR）

偏移地址：0x4C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	DMAB	RW	32'h0	DMA 连续传送寄存器 (DMA register for burst accesses) 对 PWM_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： PWM_CR 地址 + (DBA + DMA 索引)x4 其中： "PWM_CR 地址"是控制寄存器 1(PWM_CR)所在的地址； "DBA"是 PWM_DCR 寄存器中定义的基地址； "DMA 索引"是由 DMA 自动控制的偏移量，它取决于 PWM_DCR 寄存器中定义的 DBL。

22. 低功耗定时器(LPTIM)

22.1 LPTIM 简介

LPTIM 是一个 16 位定时器。由于其时钟源的多样性，LPTIM 能够在所有功耗模式下保持运行。考虑到 LPTIM 即使没有内部时钟源也能运行，可以用作“脉冲计数器”，这在某些应用中很有用。LPTIM 引入了一种灵活的时钟方案，可提供所需的功能和性能，同时将功耗降至最低。

22.2 LPTIM 主要特性

- 16 位向上计数器
- 3 位预分频器，具有 8 个可能的分频因子（1，2，4，8，16，32，64，128）
- 可选时钟
  - 内部时钟源: APB 时钟或其他任何振荡器（见 RCC 章节）
  - 通过 LPTIM 输入的外部时钟源（在没有运行嵌入式振荡器的情况下工作，由脉冲计数器应用程序使用）。
- 16 位 ARR 可重载寄存器
- 16 位比较寄存器
- 连续/单次模式
- 可选软件/硬件输入触发
- 可配置数字毛刺滤波器
- 可配置输出：脉冲,PWM
- 可配置 IO 极性
- 编码器模式

22.3 LPTIM 功能描述

22.3.1 模块框图

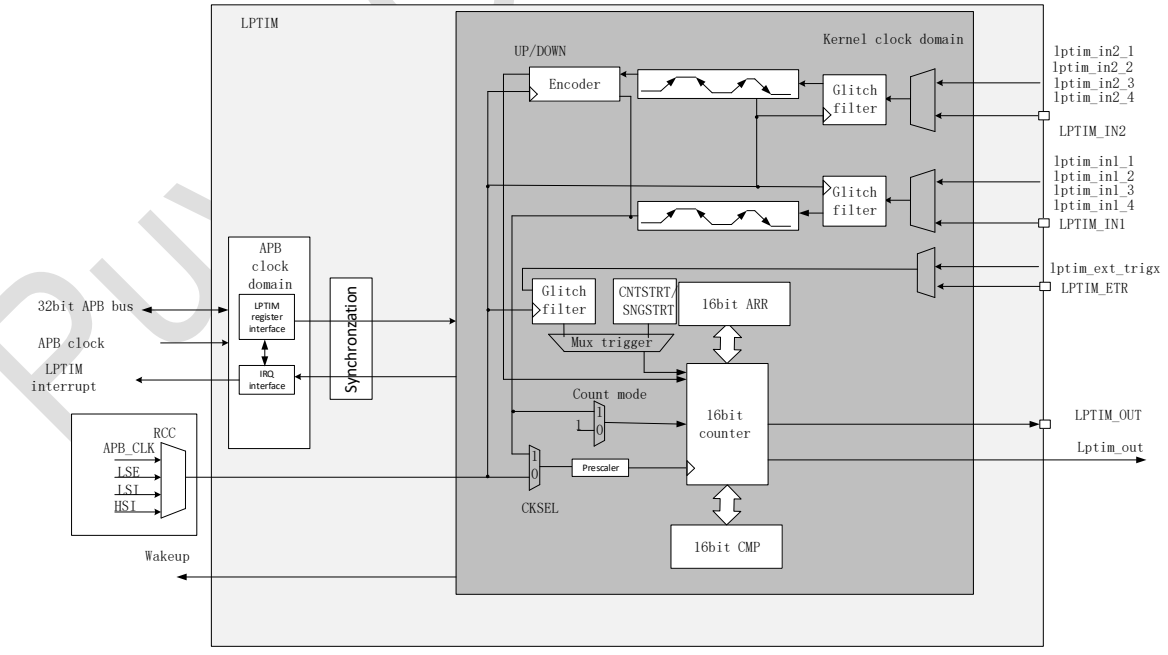


图 22-1 LPTIM 框图

22.3.2 复位和时钟

通过 RCC 模块，可以使用内部时钟信号对 LPTIM 进行时钟控制（该时钟信号可以在 APB、LSI、LSE 中进行选择）。

22.3.3 毛刺滤波器

LPTIM 输入，无论是外部（映射到 GPIO）还是内部（在芯片级映射到其他嵌入式外围设备，如比较器）触发，都受到数字滤波器的保护，可防止任何毛刺和噪声扰动在 LPTIM 内部传播。这是为了防止意外计数或触发。在激活数字滤波器之前，应首先向 LPTIM 提供一个内部时钟源。这是保证滤波器正常运行所必需的。LPTIM 包含如下数字滤波器：

- 数字滤波器保护 LPTIM 内部触发输入。数字滤波器灵敏度由 TRGFLT 位控制。

注意：数字滤波器灵敏度由组控制。不可能在同一组内单独配置每个数字滤波器灵敏度。

滤波器灵敏度作用于在 LPTIM 某一输入上检测到的连续相等样本的数量，以将信号电平变化视为有效转换。下图为在编程 2 个连续样本的情况下毛刺滤波器行为的示例。

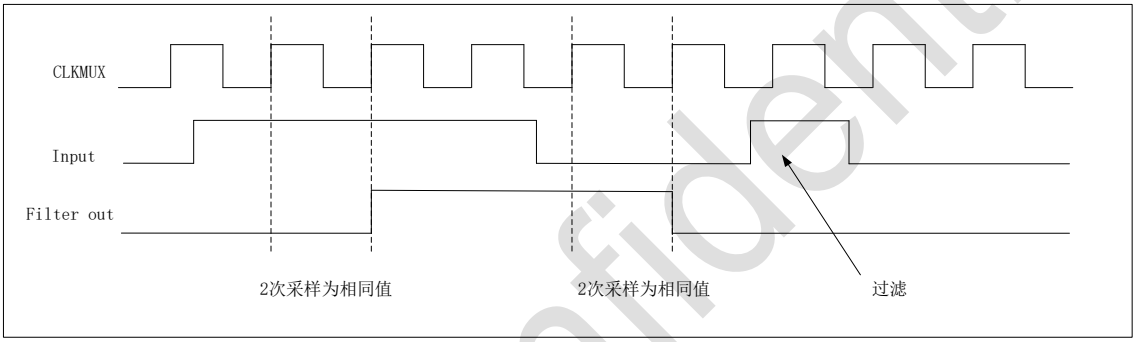


图 22-2 毛刺滤波器行为示例

注意：如果没有提供内部时钟信号，则必须通过将 TRGFLT 位设置为“0”来停用数字滤波器。在这种情况下，可以使用外部模拟滤波器（IO 模块实现）来保护 LPTIM 外部输入免受毛刺干扰。

22.3.4 预分频

LPTIM 16 位计数器，由一个可配置的 2 次方预分频器控制驱动。预分频器分频比由 LPTIM_CFGR.PRESC[2:0]控制。

下表列出了所有情况：

表 22-1 预分频器的分频比

Programming	Dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

22.3.5 触发选择

LPTIM 计数器可以通过软件启动，也可以在检测到 8 个（可选最大输入个数）触发输入之一的激活边沿后启动。

TRIGEN[1:0]用于确定 LPTIM 触发源：

- 当 TRIGEN[1:0]等于“00”时，一旦 CNTSTRT 或 SNGSTRT 位之一由软件设置，LPTIM 计数器就会启动。TRIGEN[1:0]的其他值可用于配置触发输入使用的有效边沿，支持上升沿和下降沿。一旦检测到有效边沿，LPTIM 计数器就会启动。
- 当 TRIGEN[1:0]不是‘00’时，TRIGSEL[2:0]用于选择 8 个触发输入中的哪一个启动计数器。

外部触发器被视为 LPTIM 的异步信号。因此，在触发检测之后，由于同步，在定时器开始运行之前需要两个计数器时钟周期的延迟。

如果在定时器已经启动时发生新的触发事件，它将被忽略。

在设置 SNGSTRT/CNTSTRT 位之前必须使能定时器。当定时器被禁用时，对这些位的任何写入都将被硬件丢弃。

注意：当通过软件启动计数器时（TRIGEN[1:0] = 00），LPTIM_CR 寄存器更新（设置 SNGSTRT 或 CNTSTRT 位之一）和计数器有效启动之间有 3 个计数时钟周期的延迟。

22.3.6 操作模式

LPTIM 具有两种如下工作模式：

- 连续计数模式：计时器自由运行，从触发事件开始运行，直到计时器被 disable 掉才停止。
- 单次计数模式：定时器从一个触发事件开始，当达到 ARR 值时停止。

22.3.6.1 单次模式

要使能单次计数，LPTIM_CR.SNGSTRT 寄存器位必须置 1。

设置 SNGSTRT 将启动计数器进行单次计数。

一个新的触发事件将重新启动计时器。在计数器启动之后，到达 ARR 之前，任何触发事件都将被忽略。

如果选择外部触发，则在设置 SNGSTRT 位后以及计数器寄存器停止（包含零值）后，再产生外部触发事件，LPTIM 将启动计数器进行新的单次计数周期，如下图所示。

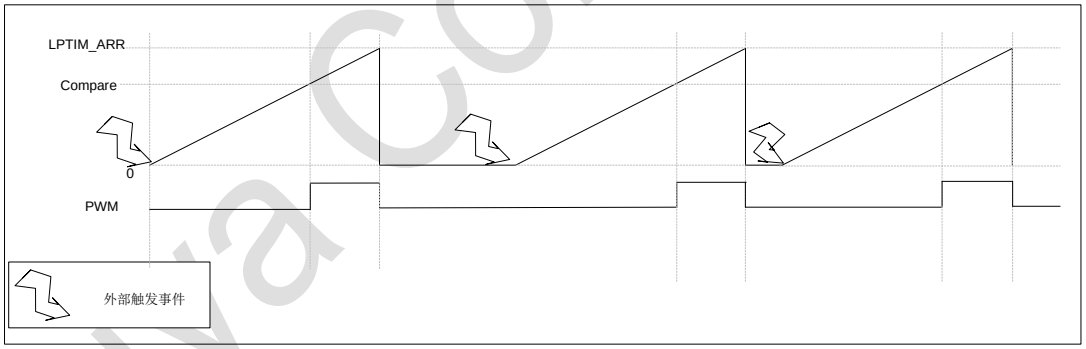


图 22-3 LPTIM 输出,单次计数模式

当 LPTIM_CFGR.WAVE 位置 1 时，一次设置模式被激活。这种情况下，计数器仅在第一次触发后启动一次，随后的任何触发事件均被忽略。如下图所示：

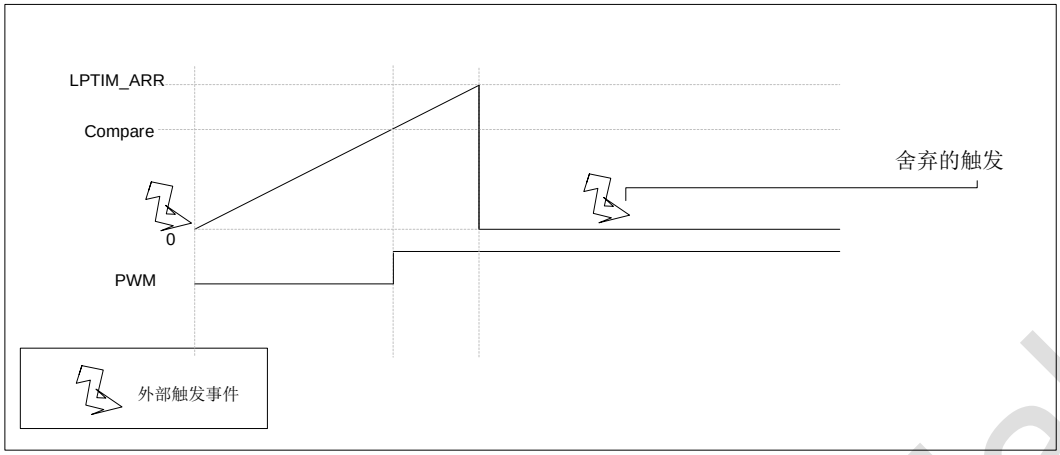


图 22-4 LPTIM 输出,单次计数模式（WAVE=1）

在软件启动(TRIGEN[1:0]=‘00’)的情况下，SNGSTRT 置位将启动计数器进行单次计数。

22.3.6.2 连续计数模式

要使能连续计数，LPTIM_CR.CNTSTRT 位必须置 1。

如果选择外部触发，则在 CNTSTRT 置位后到达的外部触发事件将启动计数器进行连续计数。任何后续的外部触发事件都将被丢弃，如下图所示。

在软件启动(TRIGEN[1:0]=‘00’)的情况下，设置 CNTSTRT 将启动计数器以进行连续计数。

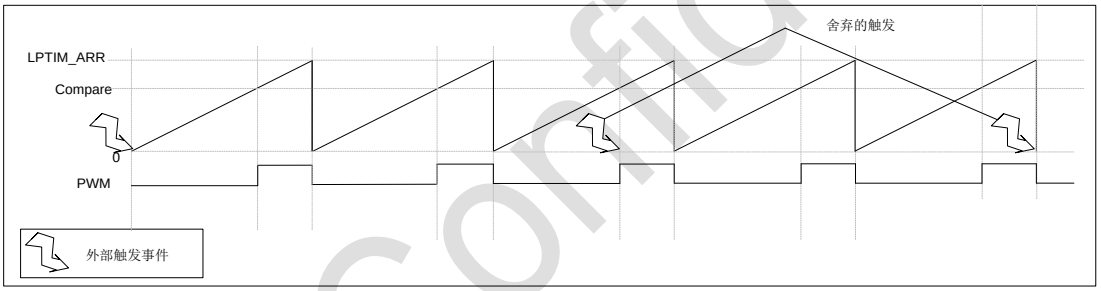


图 22-5 LPTIM 输出,连续计数模式

LPTIM_CR.SNGSTRT 和 LPTIM_CR.CNTSTRT 位只能在定时器使能时置位 (LPTIM_CR.ENABLE 为“1”）。可以快速的从单次模式转变为连续模式：

- 如果之前选择了连续模式，则置位 LPTIM_CR.SNGSTRT 会将 LPTIM 切换到单次模式。计数器（如果开始计数）一到达 ARR 就会停止。
- 如果先前选择了单次模式，则置位 LPTIM_CR.CNTSTRT 会将 LPTIM 切换到连续模式。计数器（如果开始计数）一到达 ARR 就会重新启动。

22.3.7 波形产生

两个 16 位寄存器，LPTIM_ARR（自动重载寄存器）和 LPTIM_CMP（比较寄存器），用于在 LPTIM 输出上生成几种不同的波形。

定时器可以生成以下波形：

- PWM 模式：LPTIM 输出置位条件为 LPTIM_CNT 中的计数器值超过了 LPTIM_CMP 中的比较值。一旦 LPTIM_ARR 和 LPTIM_CNT 寄存器之间发生匹配，LPTIM 输出就会复位。
- 单脉冲模式：输出波形类似于第一个脉冲的 PWM 模式，然后输出永久复位。
- 一次设置模式：输出波形类似于单脉冲模式，不同点在于输出会保持在最后一个信号电平（取决于输出配置的极性）。

上述模式要求 LPTIM_ARR 寄存器值严格大于 LPTIM_CMP 寄存器值。

LPTIM 输出波形可通过 WAVE 位进行如下配置：

- 将 WAVE 位重置为“0”会强制 LPTIM 生成 PWM 波形或单脉冲波形，具体取决于设置的位：CNTSTRT 或 SNGSTRT。
- 将 WAVE 位设置为“1”会强制 LPTIM 生成一次设置模式波形。

WAVPOL 位控制 LPTIM 输出极性。更改立即生效，因此，在重新配置极性后，甚至在启用定时器之前，输出默认值将立即改变。

可以产生频率高达 LPTIM 时钟频率除以 2 的信号。下图显示了可在 LPTIM 输出上生成的三种可能波形。此外，它还显示了使用 WAVPOL 位改变极性的效果。

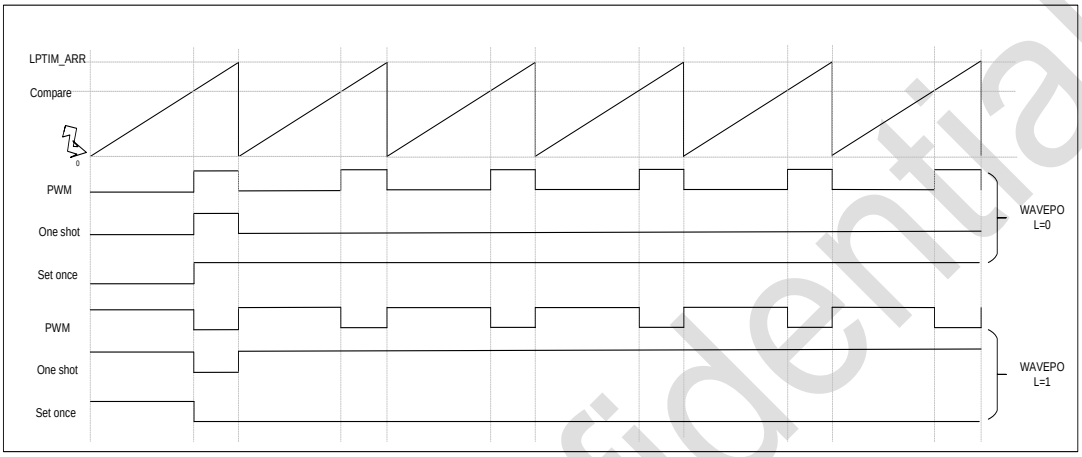


图 22-6 LPTIM 波形产生

22.3.8 寄存器更新

LPTIM_ARR 寄存器在 APB 总线写操作后立即更新，如果定时器已经启动，则与下一个 LPTIM 更新事件同步进行更新。

PRELOAD 位控制 LPTIM_ARR 寄存器的更新方式：

- 当 PRELOAD 位被复位为“0”：LPTIM_ARR 寄存器在任何写访问后立即更新。
- 当 PRELOAD 位被设为“1”时：如果定时器已经启动，则 LPTIM_ARR 寄存器将在当前周期结束时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用不同的时钟，因此在 APB 写入,和写入的值被应用到计数器比较器时，存在一定的延迟。在此延迟周期内，必须避免对这些寄存器进行任何额外的写操作。

LPTIM_ISR 寄存器中的 ARROK 标志和 CMPOK，指示对 LPTIM_ARR 和 LPTIM_CMP 寄存器的写操作是否完成。

对 LPTIM_ARR 和 LPTIM_CMP 寄存器进行写操作后，只有在前一次写操作完成后，才能对同一寄存器执行新的写操作。在 ARROK 标志位或者 CMPOK 标志位置位之前的任何连续写入都将导致不可预测的结果。

22.3.9 计数器模式

可选择以下计数模式：

- LPTIM 由内部时钟源计时

LPTIM 时钟由内部时钟源提供，LPTIM 计数器配置为在每个内部时钟脉冲后更新。

22.3.10 定时器使能

LPTIM_CR.ENABLE 位用于使能/禁用 LPTIM 内核逻辑。置位 ENABLE 位后，需要延迟两个计数器时钟才能使能 LPTIM。

仅当 LPTIM 禁用时，才能修改 LPTIM_CFGR 和 LPTIM_IER 寄存器。

22.3.11 定时器复位

为了将 LPTIM_CNT 寄存器的内容复位，提供如下所述两种复位机制。

注意：异步复位和同步复位应确保互斥使用。

22.3.11.1 异步复位

异步复位由 LPTIM_CR 寄存器的 RSTARE 位控制。当该位被置为 1 时，任何读 LPTIM_CNT 寄存器的访问都将其内容复位为零。异步复位应在未提供 LPTIM 内核时钟的时间范围内触发。

应注意，为了可靠地读取 LPTIM_CNT 寄存器，必须进行 2 次读访问并比较其结果，结果一致，被则认为是读出值是可靠的。

需要注意的是：

- 使能异步复位时，读取两次 LPTIM_CNT 寄存器的结果是不同的（第一次读会复位 LPTIM_CNT）。
- 在 LPTIM 计数时钟选择 PCLK 时，连续访问 2 次也不能保证读出值可靠。

22.3.11.2 同步复位

同步复位由 LPTIM_CR.COUNTRST 位控制。将 COUNTRST 位置 1 后，复位信号送给 LPTIM 的内核时钟域。所以需要注意在复位起作用前，LPTIM 内核时钟域的逻辑电路已过去几个时钟周期了。这将使从复位触发到复位生效，LPTIM 计数器多计了额外几个数。

由于 COUNTRST 是 APB 时钟域的，而 LPTIM 计数器位于 LPTIM 内核时钟域，所以当向 COUNTRST 位写入 1 时，需要内核时钟的 3 个时钟周期的延迟来同步来自 APB 时钟域的复位信号。

22.3.12 编码器模式

该模式允许处理来自用于检测旋转元件角位置的正交编码器的信号。编码器接口模式简单地充当具有方向选择的外部时钟。这意味着计数器只是在 0 和 ARR 值之间连续计数（0 到 ARR 或 ARR 到 0，具体取决于方向）。因此必须在启动前配置 LPTIM_ARR。从两个外部输入信号 Input1 和 Input2 生成一个时钟信号来为 LPTIM 计数器提供时钟。这两个信号之间的相位决定了计数方向。

编码器模式仅在 LPTIM 由内部时钟源计数时可用。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟的 1/4。这是强制性要求以保证 LPTIM 的正常运行。

方向更改由 LPTIM_ISR 寄存器中的两个向下和向上标志指示。此外，如果启用 DOWNIE/UPIE 位，则可以为两个方向更改事件生成中断。

要激活编码器模式，必须将 ENC 位设置为“1”。LPTIM 必须首先配置为连续模式。

当编码器模式处于活动状态时，LPTIM 计数器会根据增量编码器的速度和方向自动修改。因此，它的内容总是代表编码器的位置。由 UP 和 DOWN 标志指示的计数方向对应于编码器转子的旋转方向。

根据 CKPOL[1:0]位配置的边沿灵敏度，可能会出现不同的计数情况。下表总结了可能的组合，假设 Input1 和 Input2 不同时切换。

表 22-2 计数方式

有效沿	反向信号电平 (input1 for input2 input2 for input1)	input1 信号		input2 信号	
		上升沿	下降沿	上升沿	下降沿
上升沿	高	递减	不计数	递增	不计数
	低	递增	不计数	递减	不计数

下降沿	高	不计数	递增	不计数	递减
	低	不计数	递减	不计数	递增

下图显示了配置双沿灵敏度的编码器模式的计数序列。

在此模式下，预分频器分频比必须等于其复位值 1（PRESC[2:0]位必须为“000”）。

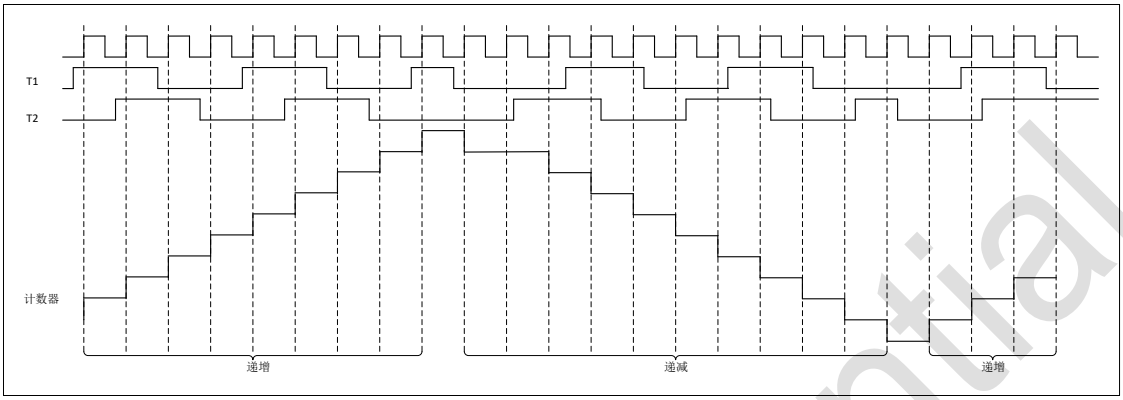


图 22-7 LPTIM 计数模式

22.3.13 调试模式

当芯片进入 debug 模式，取决于 DBG 模块的 DBG_LPTIM_STOP 位的设定，LPTIM 或者继续正常工作，或者停止工作。

22.4 LPTIM 低功耗模式

表 22-3 LPTIM 低功耗模式

Mode	Description
Sleep	无影响。 LPTIM 中断（使能后）会退出 Sleep 模式.
Low-power run	无影响。
Low-power sleep	无影响。LPTIM 中断导致设备退出低功耗睡眠模式。
Stop0/Stop1	当 LPTIM 由 LSE 或 LSI 计时时没有影响。 LPTIM 中断导致设备退出 Stop 0 和 Stop 1。

注：在使用 lptim 唤醒时，需要确保两次唤醒的间隔时间大于 6 个 PCLK 周期，否则将导致不可预知的结果。

22.5 LPTIM 中断

如果下列事件在 LPTIM_IER 寄存器内使能，则这些事件将生成中断/唤醒事件：

- 比较匹配
- 自动重新加载匹配（如果是编码器模式，则无论方向如何）
- 外部触发事件
- 自动重载寄存器写入完成
- 比较寄存器写入完成
- 方向改变（编码器模式），可编程（上/下/两者）。

注意:如果在 LPTIM_ISR 寄存器（状态寄存器）中的相应标志置 1 后，LPTIM_IER 寄存器（中断使能寄存器）中的相应位才被置 1，则不会产生中断。

表 22-4 LPTIM 中断事件

中断事件	描述
比较匹配	当计数器寄存器 (LPTIM_CNT) 的内容与比较寄存器 (LPTIM_CMP) 的内容匹配时，会产生中断标志。
自动重载匹配	当计数器寄存器的内容(LPTIM_CNT)与自动重新加载寄存器的内容匹配 (LPTIM_ARR)，中断标志置位
外部触发事件	当检测到外部触发事件时，中断标志被置位
自动重载寄存器写入完成	当对 LPTIM_ARR 寄存器的写操作完成时，会产生中断标志。
比较寄存器写入完成	当对 LPTIM_CMP 寄存器的写操作完成时，会产生中断标志。
方向改变	在编码器模式下使用。 嵌入了两个中断标志以指示方向改变： — UP 标志指示递增计数方向改变 — DOWN 标志表示减计数方向改变

22.6 LPTIM 寄存器

22.6.1 LPTIM 中断和状态寄存器 (LPTIM_ISR)

地址偏移：0x000

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWM	UP	ARROK	CMPOK	Res.	ARRM	CMPM
									R	R	R	R		R	R

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWN	R	0	计数器由递增计数变为递减计数 在编码器模式下，DOWN 位由硬件设置用于通知应用程序计数器已从递增计数变为递减计数。可以通过向 LPTIM_ICR 寄存器中的 DOWNCF 位写入 1 来清除 DOWN 标志。 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UP	R	0	计数器由递减计数变为递增计数 在编码器模式下，UP 位由硬件设置用于通知应用程序计数器已从递减计数变为递增计数。可以通过向 LPTIM_ICR 寄存器中的 UPCF 位写入 1 来清除 UP 标志。注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROK	R	0	自动重载寄存器更新 OK。 ARROK 由硬件设置，以通知应用程序 APB 总线对 LPTIM_ARR 的写操作已成功完成。向 LPTIM_ICR.ARROKCF 写入 1 可清除 ARROK 标志。
3	CMPOK	R	0	比较寄存器更新 OK CMPOK 由硬件置位，用于通知应用程序 APB 总线对 LPTIM_CMP 寄存器的写操作已成功完成。向 LPTIM_ICR.CMPOKCF 写入 1 可清除 ARROK 标志。
2	Reserved	-	-	保留
1	ARRM	R	0	自动重载匹配

				ARRM 由硬件设置，通知应用程序 LPTIM_CNT 寄存器值匹配 LPTIM_ARR 寄存器的值。向 LPTIM_ICR 寄存器的 ARRMCF 位写入 1 可清除 ARRM 标志。
0	CMPM	R	0	比较匹配 CMPM 位由硬件置位，用于通知应用程序 LPTIM_CNT 寄存器值达到 LPTIM_CMP 寄存器的值。向 LPTIM_ICR 寄存器的 CMPMCF 位写入 1 可清除 CMPM 标志。

22.6.2 LPTIM 中断清零寄存器（LPTIM_ICR)

地址偏移：0x004

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWMCF	UPCF	ARROKCF	CMPOKCF	Res.	ARRM CF	CMPMCF
									W	W	W	W		W	W

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWNCF	W	0	计数方向改变为向下清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 DOWN 标志。 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UPCF	W	0	方向改变为向上清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 UP 标志。 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROKCF	W	0	自动重载寄存器更新 OK 清除标志。 向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARROK 标志。
3	CMPOKCF	W	0	比较寄存器更新 OK 清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 CMPOK 标志。
2	Reserved	-	-	保留
1	ARRMCF	W	0	自动重载匹配清除标志 向该位写入 1 可清除 LPTIM_ISR 寄存器中的 ARRM 标志。
0	CMPMCF	W	0	比较匹配清除标志 向该位写入 1 会清除 LPTIM_ISR 寄存器中的 CMPM 标志。

22.6.3 LPTIM 中断使能寄存器（LPTIM_IER)

地址偏移：0x008

复位值： 0x0000 0000

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

Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWNIE	UPIE	ARROKIE	CMPOKIE	Res.	ARRMIE	CMPMIE
									RW	RW	RW	RW		RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DOWNIE	RW	0	计数器由递增计数变为递减计数中断使能 0：禁用 DOWN 中断 1：使能 DOWN 中断 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
5	UPIE	RW	0	计数器由递减计数变为递增计数中断使能 0：禁用 UP 中断 1：使能 UP 中断 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
4	ARROKIE	RW	0	自动重载寄存器更新 OK 中断使能。 0：禁用 ARROK 中断 1： 使能 ARROK 中断
3	CMPOKIE	RW	0	比较寄存器更新 OK 中断使能 0：禁用 CMPOK 中断 1：使能 CMPOK 中断
2	Reserved	-	-	保留
1	ARRMIE	RW	0	自动重载匹配中断使能 0：禁用 ARRM 中断 1： 使能 ARRM 中断
0	CMPMIE	RW	0	比较匹配中断使能 0：禁用 CMPM 中断 1：使能 CMPM 中断

22.6.4 LPTIM 配置寄存器 (LPTIM_CFGR)

地址偏移：0x00C

复位值： 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENC	Res.	PRELOAD	WAVPOL	WAVE	Res.	TRIGEN[1:0]		Res.
							RW		RW	RW	RW		RW	RW	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TRIGSEL[2:0]			Res.	PRESC[2:0]			Res.	TRGFLT[1:0]		Res.	Res.	Res.	CKPOL[1:0]		Res.
RW	RW	RW		RW	RW	RW		RW	RW				RW	RW	

Bit	Name	R/W	Reset Value	Function
31:25	Reserved	-	-	保留
24	ENC	RW	0	编码器模式使能 ENC 位控制编码器模式 0：禁用编码器模式 1：启用编码器模式 注意：如果 LPTIM 不支持编码器模式功能，则该位保留。
23	Reserved	-	-	保留
22	PRELOAD	RW	0	寄存器更新模式。

				预加载位控制 LPTIM_ARR 寄存器更新模式。 0:每次 APB 总线写访问后更新寄存器； 1:寄存器在当前 LPTIM 计数周期结束时更新； 注：本寄存器仅针对 LPTIM_ARR 寄存器。
21	WAVPOL	RW	0	波形极性 WAVPOL 位控制输出极性 0：LPTIM 输出反映了 LPTIM_CNT 和 LPTIM_CMP 寄存器之间的比较结果 1：LPTIM 输出反映了 LPTIM_CNT 和 LPTIM_CMP 寄存器之间比较结果相反的值
20	WAVE	RW	0	波形形状. WAVE 位控制输出形状 0：停用一次设置模式，PWM 还是单脉冲波形具体取决于定时器的启动方式（用于 PWM 的 CNTSTRT 或用于单脉冲波形的 SNGSTRT）。 1：激活一次设置模式
19	Reserved	-	-	保留
18:17	TRIGEN[1:0]	RW	2’h0	触发使能和极性 TRIGEN 位控制 LPTIM 计数器是否由外部触发启动。如果选择了外部触发选项，则触发有效边沿可以采用 2 种配置： 00：软件触发（计数启动由软件发起） 01：上升沿为有效沿 10：下降沿为有效沿 11：保留
16	Reserved	-	-	保留
15:13	TRIGSEL[2:0]	RW	3’h0	触发选择器 TRIGSEL 位在以下 4 个可用源中选择将作为 LPTIM 触发事件的触发源： 000：lptim_ext_trig0（GPIO） 001：lptim_ext_trig1（RTC_ALARM） 010：lptim_ext_trig2（COMP1_OUT） 011：lptim_ext_trig3（COMP2_OUT） 其他：保留
12	Reserved	-	-	保留
11:9	PRESC[2:0]	RW	3’h0	时钟预分频器。 PRESC 位配置预分频器分频系数。分频系数可从以下分频系数中选择: 000:/1； 001:/2； 010:/4； 011:/8； 100:/16； 101:/32； 110:/64； 111:/128；

8	Reserved	-	-	保留
7:6	TRGFLT[1:0]	RW	2'h0	触发的可配置数字滤波器 TRGFLT 值设置当内部触发器上发生电平变化时应检测到的连续相等采样值的数量，然后才将其视为有效电平转换。必须存在内部时钟源才能使用此功能。 00：任何触发活动电平变化都被视为有效触发 01：触发有效电平变化必须稳定至少 2 个时钟周期才被认为是有效触发。 10：触发有效电平变化必须稳定至少 4 个时钟周期才被认为是有效触发。 11：触发有效电平变化必须稳定至少 8 个时钟周期才被认为是有效触发。
5:3	Reserved	-	-	保留
2:1	CKPOL[1:0]	RW	2'h0	处于活动状态编码器模式选择。 00：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 1 处于活动状态。 01：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 2 处于活动状态。 10：如果 LPTIM 配置为编码器模式(置位 ENC 位)，则编码器子模式 3 处于活动状态。 11：保留 注意：如果 LPTIM 不支持编码器模式功能，则该位域保留。
0	Reserved	-	-	保留

注：配置 CFGR 寄存器时建议一次写所有配置位，若分开配置时，两次配置之间需要间隔 3 个内核时钟。

22.6.5 LPTIM 控制寄存器（LPTIM_CR)

地址偏移：0x010

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RSTARE	COUNTRST	CNTSTRT	SNGSTRT	ENABLE
											rw	rw	rw	rw	rw

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	RSTARE	RW	0	读取后复位使能 此位由软件置 1 和清 0。当 RSTARE 设置为"1"时，对 LPTIM_CNT 的任何读取访问寄存器将异步重置 LPTIM_CNT 寄存器内容。
3	COUNTRST	RW	0	计数器复位。 该位由软件置 1，硬件清 0。设置为"1"时，此位将触发 LPTIM_CNT 计数寄存器同步复位。由于此复位的同步特性，它只需要在同步延迟 3 个 LPTIM 内核时钟周期之后释放（LPTIM 内核时钟可能是与 APB 时钟不同）。

				注：在 COUNTRST 已被硬件清除为“0”之前，软件绝不能将其设置为“1”.因此，软件在尝试将其设置为“1”之前，应检查 COUNTRST 位是否已清零为“0”
2	CNTSTRT	RW	0	定时器启动连续模式。 该位由软件置位。 如果在进行单次计数模式计数时该位被置 1，则定时器不会在下一个 LPTIM_ARR 和 LPTIM_CNT 寄存器匹配的脉冲模式计数时停止。LPTIM 计数器保持在连续模式下计数。 注：仅当 LPTIM 使能时，此位才能置 1。该位在同步到内核时钟后由硬件清零。
1	SNGSTRT	RW	0	LPTIM 启动单次模式。 该位由软件置位，由硬件清零。该位置 1 将以单脉冲模式启动 LPTIM。 注：仅当 LPTIM 使能时，此位才能置 1。它将由硬件自动复位。
0	ENABLE	RW	0	LPTIM 使能位,由软件设置和清零 0:LPTIM 禁用 1:LPTIM 使能

22.6.6 LPTIM 比较寄存器 (LPTIM_CMP)

地址偏移：0x014

复位值： 0x0000 0000

LPTIM1:

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

Bit	Name	R/W	Reset Value	Function
31:0	CMP	RW	32’h0	比较值。 CMP 是 LPTIM 的比较值 当 LPTIM 使能后才能更新该寄存器。

LPTIM2:

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CMP	RW	16’h0	比较值。 CMP 是 LPTIM 的比较值 当 LPTIM 使能后才能更新该寄存器。

22.6.7 LPTIM 自动重载寄存器（LPTIM_ARR)

地址偏移：0x018

复位值： 0x0000 0001

LPTIM1:

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	Name	R/W	Reset Value	Function
31:0	ARR[31:0]	RW	32'h1	自动重新加载值 ARR 是 LPTIM 的自动重载值。 当 LPTIM 使能后才能更新该寄存器。

LPTIM2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ARR[15:0]	RW	16'h1	自动重新加载值 ARR 是 LPTIM 的自动重载值。 当 LPTIM 使能后才能更新该寄存器。

22.6.8 LPTIM 计数器寄存器（LPTIM_CNT)

地址偏移：0x01C

复位值： 0x0000 0000

LPTIM1:

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

Bit	Name	R/W	Reset Value	Function
31:0	CNT[31:0]	R	32'h0	计数器值 当 LPTIM 以异步时钟运行时，读取 LPTIM_CNT 寄存器可能返回不可靠的值。因此在这种情况下，有必要执行两次连续的读访问并验证返回的两个值是否相同。需要注意的是，对于可靠的 LPTIM_CNT 寄存器读访问，需要两次连续的读操作并比较，相等时，可以认为读取访问是可靠的。

LPTIM2:

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

R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	R	16'h0	计数器值 当 LPTIM 以异步时钟运行时，读取 LPTIM_CNT 寄存器可能返回不可靠的值。因此在这种情况下，有必要执行两次连续的读访问并验证返回的两个值是否相同。需要注意的是，对于可靠的 LPTIM_CNT 寄存器读访问，需要两次连续的读操作并比较，相等时，可以认为读取访问是可靠的。

22.6.9 LPTIM 选择寄存器（LPTIM_OR)

地址偏移：0x01C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IN1[2:1]		IN1[2:1]		IN2[0]	IN1[0]
										RW	RW	RW	RW	RW	RW

Bit	Name	R/W	ResetValue	Function
31:16	Reserved	-	-	保留
5:4	IN2[2:1]	RW	2'h0	IN2SEL[1:0]：LPTIM 输入 2 选择 IN2SEL 位控制 LPTIM 输入 2 多路复用器，它将 LPTIM 输入 2 连接到以下可用输入之一。 00：COMP1 01：COMP2 其它：保留
3:2	IN1[2:1]	RW	2'h 0	IN2SEL[1:0]：LPTIM 输入 1 选择 IN2SEL 位控制 LPTIM 输入 1 多路复用器，它将 LPTIM 输入 1 连接到以下可用输入之一。 00：COMP1 01：COMP2 其它：保留
1	IN2[0]	RW	0	LPTIM 输入 2 重映射 1：连接到 IN2[2:1]指示的 COMP 输出 0：连接到 GPIO
0	IN1[0]	RW	0	LPTIM 输入 1 重映射 1：连接到 IN1[2:1]指示的 COMP 输出 0：连接到 GPIO

23. 红外接口(IRTIM)

芯片内集成为遥控而用的红外接口（IRTIM）。它可以与一个红外 LED 一起实现遥控的功能。它使用 TIM16 和 TIM17 的内部连接，如下图所示。

为产生红外遥控信号，必须打开 Infrared interface（红外接口），并且 TIM16 的 channel 1(TIM16_OC1) 和 TIM17 channel 1（TIM17_OC1）要被适当的配置以产生正确的波形。

红外接收器可以很容易通过一个基本输入捕获模式实现。

所有标准的红外脉冲调制模式可以通过对两个定时器的输出比较通道进行编程而获得。

TIM17 被用来产生高频载波信号，而 TIM16 可以产生调制包络。

红外功能输出到 IR_OUT pin，这个功能的激活是通过使能 GPIO_AFRx 寄存器的相关复用功能位实现的。

LED 需要大的灌电流驱动能力（仅可以在 PB9 引脚），这可以通过 SYSCFG_CFGR1 寄存器的 I2C1_FMP 位被激活，这样就足够支撑直接控制红外 LED 大的灌电流而用。

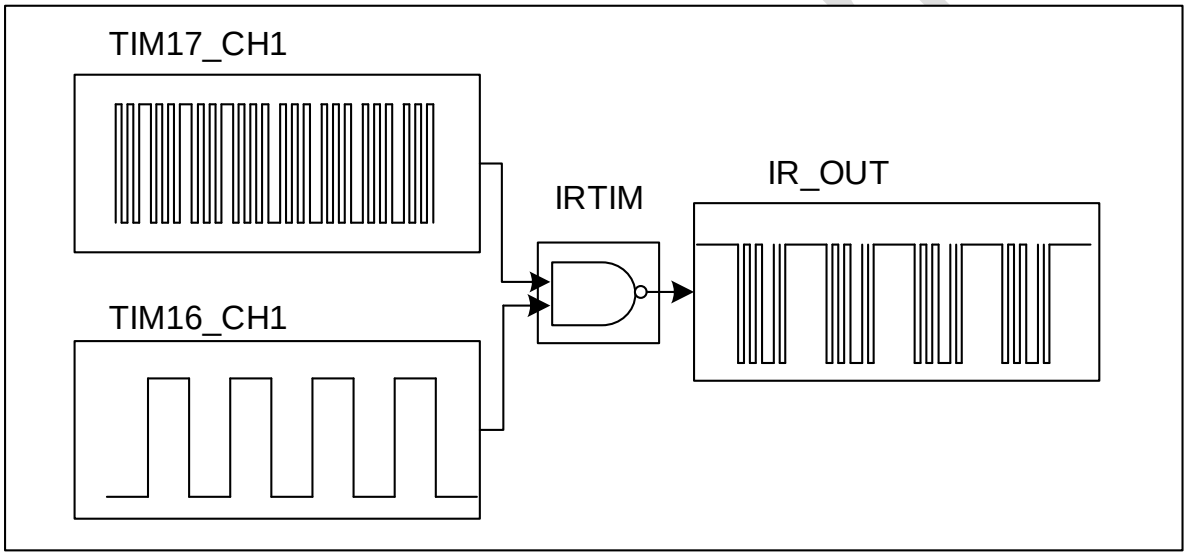


图 23-1 IRTIM 和 TIM16/TIM17 的内部连接

24. 实时时钟（RTC）

24.1 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)处于备份域，即在系统复位后，RTC 的设置和时间维持不变。系统复位后，对备份寄存器和 RTC 的访问被禁止，这是为了防止对备份域(BKP)的意外写操作。

执行以下操作将使能对备份寄存器和 RTC 的访问：

- 设置寄存器 RCC_APB1ENR 的 PWREN 位，使能电源时钟
- 设置寄存器 PWR_CR1 的 DBP 位，使能对备份寄存器和 RTC 的访问。

RTC 内还包含了 5 个 32 位的备份寄存器，可用来存储 20 个字节的应用程序数据。备份寄存器处在备份域里。当系统复位时，它们也不会被擦除。

备份寄存器由上电复位（POR）或软件复位（BDRST）所复位，且在检测到侵入事件时，擦除备份寄存器内容。

备份寄存器受 RDP 保护，当 RDP 等级为 1 时，不可写入或读取备份寄存器，当 RDP 等级从 1 变为 0 时，擦除备份寄存器的所有数据。

此外，RTC 内还包含了用来管理侵入检测的寄存器和 RTC 校准功能控制寄存器

24.2 RTC 主要特性

主要功能如下：

- 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)。
- 可以选择以下三种 RTC 的时钟源：
  - HSE 时钟的 128 分频
  - LSE 振荡器时钟
  - 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、分频器、侵入检测电路、校准电路、备份寄存器)只能由备份域复位。
- 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。
  - RTC 全局中断，包含秒中断、闹钟中断、溢出中断。
  - 侵入事件中断
- 支持 20 字节数据备份寄存器
- 拥有 1 个用来存储 RTC 时钟的校验值的校验寄存器
- PC13 引脚可用于侵入检测
- 可在 PC13 引脚（当该引脚不用于侵入检测时)上输出 RTC 校准时钟，RTC 闹钟脉冲或者秒脉冲
- 备份寄存器受 Flash 的 RDP 保护

24.3 RTC 功能描述

24.3.1 概述

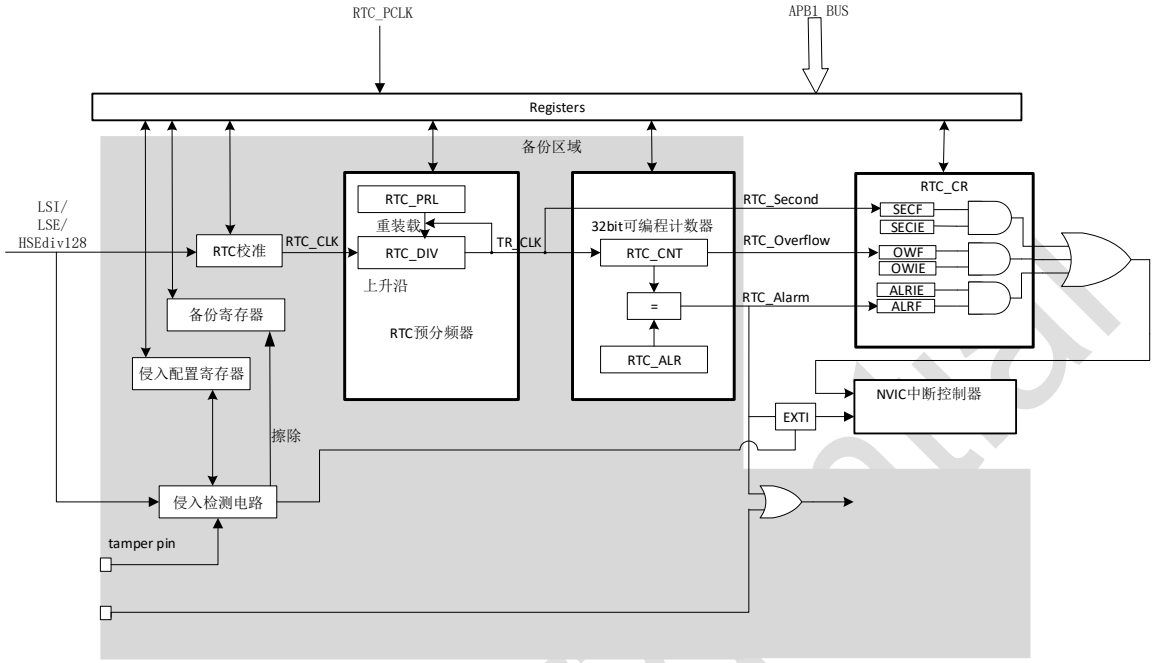


图 24-1 RTC 功能框图

RTC 由两个主要部分组成(参见上图)。第一部分(APB1 接口)用来和 APB1 总线相连。此单元还包含一组 32 位寄存器（RTC_CR，实际分散在两个地址的寄存器），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，用来连接 APB1 总线。

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 的预分频器模块，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)（second）。

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（alarm）。

侵入检测电路、RTC 时钟校准电路、备份寄存器也包含在了 RTC 核心里，在停止状态下也能正常工作。闹钟中断作为停止模式的唤醒信号。在停止模式，RTC 闹钟作为唤醒需要配置 EXTI line17 对应寄存器。

注意：RTC 全局中断和侵入中断并不能唤醒停止模式。

24.3.2 寄存器复位

RTC_PRL，RTC_ALR，RTC_CNT，RTC_DIV，RTC_TMPCFGR，RTC_TMPCSR，RTC_CALIBR，RTC_BKPDRx 寄存器仅能通过备份域复位信号复位，RTC_CR 则由系统复位或电源复位进行异步复位。

24.3.3 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。但是，相关的可读寄存器只在与 RTC 时钟的上升沿同步到 RTC APB1 时钟后的信号有效时更新。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
- 系统刚从停机模式唤醒(这种情况计数值只是不会更新，因为 CPU 不工作，系统时钟停止，但 RTC 正常计数，计数值不会同步到 V_{DDD} 区)

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_PRL，RTC_CNT，RTC_DIV，RTC_ALR 时，如果 RTC 的 APB1 接口曾经处于禁止状态，则软件首先必须等待 RTC_CR 寄存器中的 RSF 位(寄存器同步标志)被硬件置'1'。

说明：

RTC_CR 寄存器为 RTC_PCLK 域，CPU 任何时候读能读到稳定值

RTC_CNT 和 RTC_DIV 来源于 RTC_CLK 域。在 RTC 工作后，RTC_DIV 寄存器在每个 RTC_CLK 的上升沿更新；RTC_CNT 和来源于 RTC_CLK 时钟域的标志位同样采用跟 RTC_DIV 寄存器同样的更新信号，虽然这样不是每次更新时 RTC_CNT 的值都改变；

RSF 实现在 RTC_PCLK 域，在 RTC_CLK 同步到 RTC_PCLK 后的脉冲信号有效时置位；

RSF 仅控制 RTC_CNT 和 RTC_DIV 的读取时机（硬件不会控制）

读侵入检测、时钟校准、备份寄存器(RTC_TMPCFGR、RTC_TMPCSR、RTC_CALIBR、RTC_BKPDRx)时，无需考虑 RSF 位的值。

24.3.4 配置 RTC 寄存器

必须设置 RTC_CR 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_PRL、RTC_CNT、RTC_ALR 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。

写侵入检测、时钟校准、备份寄存器(RTC_TMPCFGR、RTC_TMPCSR、RTC_CALIBR、RTC_BKPDRx)时，无需考虑 CNF 位的值。

配置过程：

- 查询 RTOFF 位，直到 RTOFF 的值变为'1'；(表明前面配置已经完成)
- 置 CNF 值为 1，进入配置模式；(此时 RTOFF 位仍为 1，保证 CPU 在写寄存器时 RTOFF=1)
- 对一个或多个 RTC 寄存器进行写操作；(RTOFF 在此过程仍为 1，RTC_CLK 域寄存器此步骤操作写入 buffer 寄存器)
- 清除 CNF 标志位，退出配置模式；(硬件检测到 CNF 清零后，开始执行上一步骤中的写寄存器操作，开始写入 RTC_CLK 域的寄存器；同时 RTOFF 被清零)
- 查询 RTOFF，直至 RTOFF 位变为'1'以确认写操作已经完成。(buffer 寄存器写入 RTC_CLK 域后置位 RTOFF)

注：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 RTC_CLK 周期。(在清除 CNF 标志位后 3 个 RTC_CLK 不能重新启动配置，否则会出现配置错误（此时通过 RTOFF=0 控制）)

说明：

1. 在此过程中，CPU 写寄存器时 RTOFF=1；
2. CPU 的写周期为从 CNF=1 到 CNF=0，配置其他寄存器在这两个操作之间；

3. 先将 CNF 写 1 后将 CNF 清零，这个操作清零 RTOFF；只写 CNF=0 或者写 CNF=1 后不清零不会清零 RTOFF；
4. 先将 CNF 写 1 后将 CNF 清零，这个操作启动 buffer 寄存器写 RTC_CLK 域寄存器的过程；
5. RTOFF 实现在 RTC_PCLK 域。

24.3.5 RTC 标志设置

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)。（达到 0 之前即为检测到 RTC_CNT 的最大值置位 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)。（检测到 RTC_CNT 计数到 RTC_ALR，且在最后一个 RTC_CLK 置位 ALRF）

对 RTC 闹钟寄存器（RTC_ALR）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 闹钟寄存器（RTC_ALR）和/或 RTC 计数器寄存器（RTC_CNT）。
- 等待 RTC 控制寄存器中的 SECF 位被设置，再更改 RTC 闹钟寄存器（RTC_ALR）和/或 RTC 计数器寄存器（RTC_CNT）。

24.3.6 RTC 时序

RTC 秒和闹钟时序

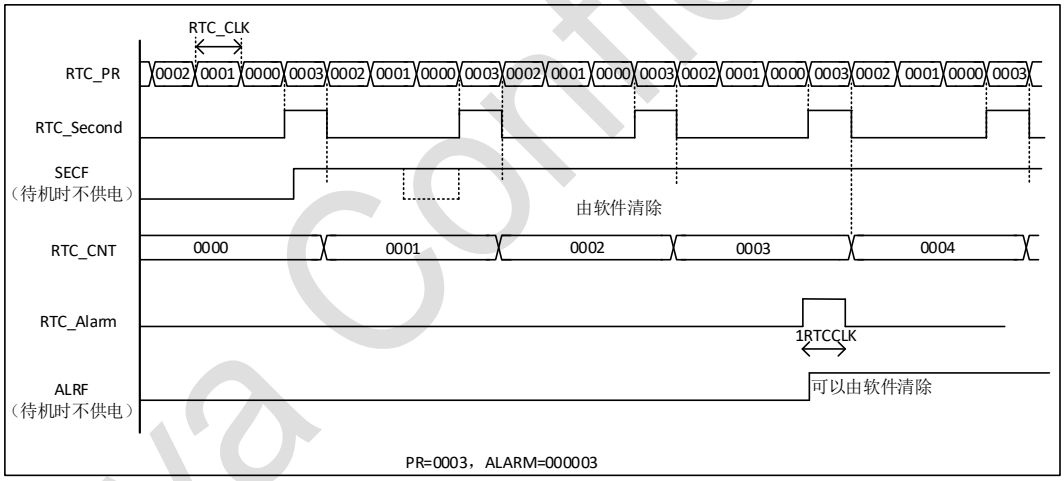


图 24-2 RTC 秒和闹钟时序图

SECF 和 ALRF 为 RTC_PCLK 域信号，分别采样 RTC_CLK 域 RTC_Sectond 和 RTC_Alarm 信号产生。

RTC 溢出时序

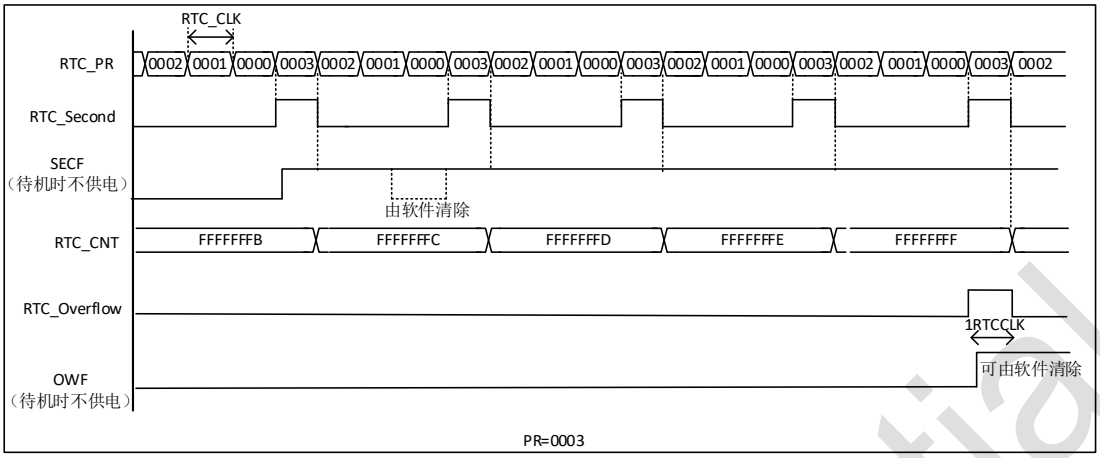


图 24-3 RTC 溢出时序图

SECF 和 OWF 为 RTC_PCLK 域信号，分别采样 RTC_CLK 域 RTC_Sectond 和 RTC_Overflow 号产生。

24.3.7 侵入检测

当侵入检测引脚(PC13)上的信号从 0 变成 1 或者从 1 变成 0(取决于 RTC 入侵配置寄存器 RTC_TMPCFGR 的 TPAL 位)，会产生一个侵入检测事件。侵入检测事件将所有数据备份寄存器内容清除。然而为了避免丢失侵入事件，侵入检测信号是边沿检测的信号与侵入检测允许位的逻辑‘与’，从而在侵入检测引脚被允许前发生的侵入事件也可以被检测到。

- 当 TPAL=0 时：如果在使能侵入检测前(通过设置 TPE 位)，PC13 已经为高电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件(尽管在 TPE 位置‘1’后并没有出现上升沿)。
- 当 TPAL=1 时：如果在使能侵入检测前(通过设置 TPE 位)，PC13 已经为低电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件(尽管在 TPE 位置‘1’后并没有出现下降沿)。

设置 RTC_TMPCSR 寄存器的 TPIE 位为‘1’，当检测到侵入事件时就会产生一个中断。在一个侵入事件被检测到并被清除后，应关闭侵入检测使能，然后在再次写入备份数据寄存器前重新用 TPE 位启动侵入检测功能。这样，可以阻止软件在侵入检测引脚上仍然有侵入事件时对备份数据寄存器进行写操作。这相当于对侵入引脚进行电平检测。

注：当 V_{CC} 电源断开时，侵入检测功能仍然有效。为了避免不必要的复位数据备份寄存器， 侵入检测引脚应该在片外连接到正确的电平。

24.3.8 RTC 校准

为方便测量，RTC 时钟可以经 64 分频输出到侵入检测引脚(PC13)上。通过设置 RTC_CALIBR 的 CCO 位来开启这一功能。

通过配置 CAL[6:0]位，此时钟可以最多减慢 121 ppm。

概述

实时时钟(RTC)精度是大多数嵌入式应用程序的要求，但由于外部环境温度变化，时钟的晶体频率变化引起 RTC 精度可能不像预期的那样准确。使用数字时钟校准电路，允许应用程序补偿晶体和温度变化。

RTC 校准方式

RTC 时钟可选择一个额定频率为 32.768 kHz 的石英晶体控制振荡器驱动。晶体振荡器是提供固定频率的最精确电路之一。时钟误差有两个原因：

1. 温度变化
2. 晶体变化

如前所述，大多数时钟芯片通过使用繁琐的微调电容器来补偿晶体频率和温度变化。设计采用周期性计数器校正。数字校准电路每 220 个时钟周期消除 0 至 127 个周期。脉冲被消隐的个数取决于加载到 BKP 的 RTC 时钟校准寄存器 BKP_RTCCR.CAL 的值。

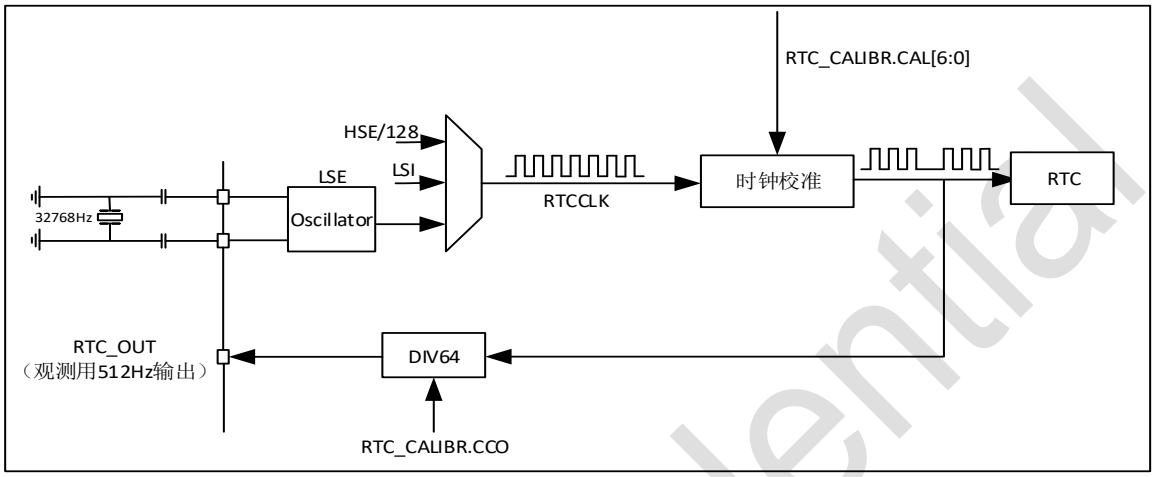


图 24-4 RTC 校准示意图

每个校准步骤的效果是每 1 048 576(220)实际振荡器周期减去 1 个振荡器周期。即校准寄存器中每个校准步骤的调整量为 0.954(1000000/220)ppm。因此，振荡器时钟可以从 0 放慢到 121 ppm。

下表显示了每个位在每个月(30 天)中实时表示的 ppm 和秒数。

表 24-1 ppm 计算对照表

校准值	四舍五入到最接近的 ppm	值以秒为单位，每月(30 天)四舍五入到最近的秒	校准值	四舍五入到最接近的 ppm	值以秒为单位，每月(30 天)四舍五入到最近的秒
0	0	0	64	61	158
1	1	2	65	62	161
2	2	5	66	63	163
3	3	7	67	64	166
4	4	10	68	65	168
5	5	12	69	66	171
6	6	15	70	67	173
7	7	17	71	68	176
8	8	20	72	69	178
9	9	22	73	70	180
10	10	25	74	71	183
11	10	27	75	72	185
12	11	30	76	72	188
13	12	32	77	73	190
14	13	35	78	74	193
15	14	37	79	75	195
16	15	40	80	76	198
17	16	42	81	77	200
18	17	44	82	78	203
19	18	47	83	79	205

20	19	49	84	80	208
21	20	52	85	81	210
22	21	54	86	82	213
23	22	57	87	83	215
24	23	59	88	84	218
25	24	62	89	85	220
26	25	64	90	86	222
27	26	67	91	87	225
28	27	69	92	88	227
29	28	72	93	89	230
30	29	74	94	90	232
31	30	77	95	91	235
32	31	79	96	92	237
33	31	82	97	93	240
34	32	84	98	93	242
35	33	87	99	94	245
36	34	89	100	95	247
37	35	91	101	96	250
38	36	94	102	97	252
39	37	96	103	98	255
40	38	99	104	99	257
41	39	101	105	100	260
42	40	104	106	101	262
43	41	106	107	102	264
44	42	109	108	103	267
45	43	111	109	104	269
46	44	114	110	105	272
47	45	116	111	106	274
48	46	119	112	107	277
49	47	121	113	108	279
50	48	124	114	109	282
51	49	126	115	110	284
52	50	129	116	111	287
53	51	131	117	112	289
54	51	133	118	113	292
55	52	136	119	113	294
56	53	138	120	114	297
57	54	141	121	115	299
58	55	143	122	116	302
59	56	146	123	117	304
60	57	148	124	118	307
61	58	151	125	119	309
62	59	153	126	120	311
63	60	156	127	121	314

如上所述，RTC 时钟校准电路只从晶体时钟中减去周期。基于 RTC 预分频器值默认设置为 32768，更快的晶体频率(>32768 Hz)可以被校准，而较慢的晶体频率(<32768 Hz)不能被补偿（只校准变快的晶体，变慢的会加剧变慢的情况）。所以只有晶体频率量程[32 772,32 768]可以校准。

由于晶体频率可能在 32.768 kHz 附近变化，比如可以通过设置 RTC 预分频为 32766(而不是 32768)来调整晶体频率由 32 768 为 32 766。这样，可以补偿晶体频率为[32770,32766]的范围。

计算所需的校准量

为了确定在给定的应用程序中需要多少校准，需要根据具体情况，借助 RTC 时钟的输出功能。RTC 时钟输出可以用来测量晶体振荡器的精度，如果晶体振荡器的频率是精确的 32768Hz，则 RTC 时钟输出的频率正好是 512 Hz。

该方法可分为以下步骤:

1.启用低速外部振荡器(LSE)，选择 LSE 作为 RTC 时钟源，然后启用 RTC 时钟。

2.在 RTC_OUT 引脚(PC13)上输出 RTC 时钟的 64 分频率，用于晶体频率测量。通过将 BKP_RTCCR 中的 CCO 位置 1 来实现。

3.计算晶体频率偏差(ppm)。假设 RTC 预分频器值为 32766，通过将 511.968 Hz(32766/64≈511.968)的测量偏差除以 511.968，并将结果乘以 100 万，就可以快速计算出 ppm 中的偏差。使用表 1 找到最近的校准值。此表是基于 ppm 表示的变化值的校准值的直接查找表。

4.向 RTC 校准寄存器中写入校准值以补偿晶体偏差。

注意:要将 RTC 预分频器设置为 32766，请将 32765 写入 RTC 预分频器寄存器。

例如，如果在实测中测量到的频率为 511.982 Hz，则 δ 为 0.014。除以 511.968，再乘以 100 万，结果是 27.35 ppm。在这种情况下，最接近的补偿值是 28。误差将从 27.35 ppm(每月~71 秒)减少到 0.65 ppm(每月~1.7 秒)。

注意:由于 RTC 校准的原理是忽略部分 RTC 时钟脉冲，它不能改善短时间内的计数，它只改善长时间内的计数。例如，在没有校准的情况下，使用 RTC 计数 0.01 秒会比有校准的情况下更准确。由于校准周期在考虑的时间范围内可能发生或不发生，因此结果值可能会发生显著变化。因此，根据应用情况，最好不要使用校准。

24.3.9 备份域寄存器擦除

以下方式可以使得备份寄存器被擦除：

①上电复位

②侵入检测电路成功捕捉到侵入事件

③寄存器 RCC_BDCR 的 BDRST（备份域软复位）被写入 1

④RDP level 从 1 变为 0

⑤TEF 标志位为 1

24.4 RTC 寄存器

寄存器宽度为 32 位，只允许字操作。

24.4.1 RTC 控制寄存器高位(RTC_CRH)

偏移地址：0x00

复位值：0x0000 0000

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

该寄存器由系统复位复位。

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2	OWIE	RW	0	溢出中断允许位 0：不允许溢出中断 1：允许溢出中断
1	ALRIE	RW	0	闹钟中断允许位 0：不允许闹钟中断 1：允许闹钟中断
0	SECIE	RW	0	秒中断允许位 0：不允许秒中断 1：允许秒中断

这些位用于屏蔽中断请求。注意：在复位后，所有中断是未使能的，所以在初始化后，写 RTC 寄存器以确保没有正在挂起的中断请求是可能的。但是当外设正在完成前一次的写操作（RTOFF=0）时，是不能写 RTC_CRH 寄存器的。

该控制寄存器控制着 RTC 的功能。某些位必须使用专门的配置流程才能进行写操作。

24.4.2 RTC 控制寄存器低位(RTC_CRL)

偏移地址：0x04

复位值：0x0000 0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOFF	CNF	RSF	OWF	ALRF	SECF
										R	RW	RC_W0	RC_W0	RC_W0	RC_W0

Bit	Name	R/W	Reset Value	Function
31:6	Reserved	-	-	保留
5	RTOFF	R	1	RTC 操作关闭 (RTC operation OFF)，该位只读。 RTC 模块利用该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。 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
4	CNF	RW	0	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 寄存器写入新值。 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
3	RSF	RC_W0	0	寄存器同步标志 (Registers synchronized flag).该寄存器由硬件置位，软件清零。当 RTC_CNT 寄存器和 RTC_DIV 寄存器更新后，硬件置'1'该位。 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，用户程序必须等待该位被硬件置'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
2	OWF	RC_W0	0	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'，写'1'无效。 0：无溢出； 1：32 位可编程计数器溢出
1	ALRF	RC_W0	0	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'，写'1'无效。 0：无闹钟； 1：有闹钟。
0	SECF	RC_W0	0	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，写'1'无效。 0：秒标志条件不成立 1：秒标志条件成立

RTC 的功能是被该控制寄存器控制的。当外设正在继续上一次写操作时（RTOFF=0），是不能写 RTC_CRL 寄存器的。

注：

- 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。
- 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
- 当 APB1 时钟不运行时， OWF、 ALRF、 SECF 和 RSF 位不被更新（无法同步）。
- OWF、 ALRF、 SECF 和 RSF 位只能由硬件置位，由软件来清零。
- 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。
- 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)。

24.4.3 RTC 预分频装载寄存器高位(RTC_PRLH)

PRL 寄存器用来保存 RTC 预分频器周期性的计数值。该寄存器是被 RTC_CR 寄存器的 RTOFF 位写保护的，只有 RTOFF=1，才允许 CPU 进行写操作。

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.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRL[19:16]			
												W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	PRL[19:16]	W	4'h0	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

24.4.4 RTC 预分频装载寄存器低位(RTC_PRLL)

偏移地址：0x0C

复位值：0x0000 8000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	PRL[15:0]	W	16'h8000	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。

24.4.5 RTC 预分频余数寄存器高位(RTC_DIVH)

在每个 TR_CLK 周期，RTC_PRL 寄存器的值被重装载到 RTC 预分频计数器里。用户可以通过读取 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。

该寄存器只读属性，当 RTC_PRL 或者 RTC_CNT 寄存器的值发生任何变化，该寄存器值将由硬件重新装载。

偏移地址：0x10

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	DIV[19:16]	R	4'h0	RTC 时钟分频器余数高位。

24.4.6 RTC 预分频余数寄存器低位(RTC_DIVL)

偏移地址：0x14

复位值：0x0000 8000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	DIV[15:0]	R	16'h8000	RTC 时钟分频器

24.4.7 RTC 计数寄存器高位(RTC_CNTH)

RTC 模块有个 32bit 可编程的计数器，该寄存器通过两个寄存器访问，计数基于预分频器产生的 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。寄存器是被写保护的，仅当 RTOFF=1 时 CPU 才能进行写操作。对高 16bit 的 RTC_CNTH 或者低 16bit 的 RTC_CNTL 寄存器进行写操作，直接装载到相应的可编程计数器里，并重装载 RTC 预分频器。当读操作发生，返回计数器的当前值（系统时间）。

偏移地址：0x18

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[31:16]	RW	16'h0	RTC core counter 的高 16bit. 当读 RTC_CNTH 寄存器时，返回 RTC 计数器寄存器的当前值的高 16bit。只有进入配置模式才能对该寄存器进行写操作。

24.4.8 RTC 计数寄存器低位(RTC_CNTL)

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CNT[15:0]	RW	16'h0	RTC core counter 低 16bit 当读 RTC_CNTL 寄存器时，返回 RTC 计数器寄存器当前值的低 16bit。只有进入配置模式才能对该寄存器进行写操作。

24.4.9 RTC 闹钟寄存器高位(RTC_ALRH)

当可编程计数器（计数）达到存储在 RTC_ALR 寄存器的 32bit 值时，并产生 alarm 中断请求。该寄存器是被 RTOFF 位写保护的，只有 RTOFF=1，才允许写访问。

偏移地址：0x20

复位值：0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ALR[31:16]	RW	16'hFFFF	RTC 闹钟值高 16bit. 该寄存器用来保存由软件写入的闹钟时间的高 16bit。写该寄存器必须进入配置模式。

24.4.10 RTC 闹钟寄存器低位(RTC_ALRL)

偏移地址：0x24

复位值：0x0000 FFFF

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	ALR[15:0]	RW	16'hFFFF	RTC 闹钟值低 16bit. 该寄存器用来保存由软件写入的闹钟时间的高 16bit。写该寄存器必须进入配置模式。

24.4.11 RTC 入侵配置寄存器（RTC_TMPCFGR)

偏移地址：0x30

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	TPAL	RW	0	侵入检测引脚(PC13)有效电平。 0：侵入检测引脚上的高电平会清除所有数据备份寄存器（TPE=1 时）； 1：侵入检测引脚上的低电平会清除所有数据备份寄存器（TPE=1 时）； 注：当 TPE=1 时，无法更改此位。
0	TPE	RW	0	启动侵入检测引脚。 0：侵入检测引脚作为通用 IO 口使用；

				1：侵入检测引脚作为侵入检测使用；

24.4.12 RTC 入侵控制/状态寄存器（RTC_TMPCSR）

偏移地址：0x34

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	TIF	TEF	Res.	Res.	Res.	Res.	Res.	TPIE	CTI	CTE
						R	R						RW	W	W

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	TIF	R	0	侵入中断标志。 当检测到侵入事件且 TPIE 位为 1 时，此位由硬件置 1. 通过向 CTI 位写 1 来清除此标志位（同时也清除中断）。如果 TPIE 位被清除，此位也会被清除。 0：无侵入中断 1：产生侵入中断 注： 此标志能被系统复位清零,但不能被备份域复位清零。
8	TEF	R	0	侵入事件标志。 当检测到侵入事件时此位由硬件置 1.通过向 CTE 位写 1 可清除此标志位。 0：无侵入事件 1：检测到侵入事件 注： 侵入事件会复位所有的 RTC_BKPDRx 寄存器。只要 TEF 为 1，所有的 RTC_BKPDRx 寄存器就一直保持复位状态。当此位被置 1 时，若对 RTC_BKPDRx 进行写操作，写入的值不会被保存
7:3	Reserved	-	-	保留
2	TPIE	RW	0	允许侵入检测引脚中断。 0：禁止侵入检测中断 1：允许侵入监测中断（RTC_TMPCFGR 寄存器的 TPE 位也必须被置 1） 注: 此标志能被系统复位清零,但不能被备份域复位清零.
1	CTI	W	0	清除侵入检测中断。 此位只能写入，读出值为 0. 0：无效 1：清除侵入检测中断和 TIF 侵入检测中断标志
0	CTE	W	0	清除侵入检测事件。 此位只能写入，读出值为 0.

				0：无效 1：清除 TEF 侵入检测事件标志（并复位侵入检测器）

24.4.13 RTC 时钟校准寄存器(RTC_CALIBR)

偏移地址：0x38

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	ASOS	ASOE	CCO	CAL[6:0]						
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	ASOS	RW	0	闹钟或者秒输出选择。 当设置了 ASOE 位，ASOS 位可以用于选择在侵入检测引脚上输出的是 RTC 秒脉冲还是闹钟脉冲信号。 0：输出 RTC 闹钟脉冲 1：输出秒脉冲 注：该位只能被备份区复位清零。
8	ASOE	RW	0	允许输出闹钟或者秒脉冲。 根据 ASOS 位配置，该位允许 RTC 闹钟或秒脉冲输出到侵入检测引脚上。 在 ASOE 置 1 之前，应先将 TPE 置 0，否则可能会错误地产生侵入事件。 注：该位只能被备份区复位清零。
7	CCO	RW	0	校准时钟输出： 0：无影响； 1：在侵入检测引脚输出校准后的 RTCCLK 64 分频时钟。 在 CCO 置 1 之前，应先将 TPE 置 0，否则可能会错误地产生侵入事件
6:0	CAL[6:0]	RW	0	校准值。 校准值表示在每 2^{20} 个时钟脉冲内将有多少个时钟脉冲被跳过。这可以用来对 RTC 进行校准，以 1000000/2^{20}ppm 的比例减慢时钟。 RTC 时钟可以被减慢 0 ~ 121ppm。

注：1.当 CCO 和 ASOE 同时配置时，ASOE 优先级高。

2.该寄存器的数据需要从 APB 同步到 RTC 时钟域，如果使用者想多次写该寄存器，那么在 n 次写入之前，应该先读取该寄存器，确认第 n-1 次的值以及成功地被写入，再对该寄存器进行写操作，否则第 n-1 次的数据会丢失。

24.4.14 RTC 备份寄存器(RTC_BKPDR)

偏移地址：0x40, 0x44, 0x48, 0x4C, 0x50

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:0	D[31:0]	RW	32'h0	备份数据。这些位可以被用来写入用户数据。 注：RTC_BKPDRx 寄存器不会被系统复位。可以由备份域复位来复位或（如果侵入检测功能被开启时）由侵入事件复位。

25. 独立看门狗（IWDG）

25.1 IWDG 简介

独立看门狗寄存器（简称 IWDG），该模块具有高安全级别、时序精确及灵活使用的特点。IWDG 可用来检测和解决由软件错误引起的故障，并在计数器达到指定的超时值时（TIMEOUT）触发系统复位。

独立看门狗(IWDG)由专用的低速内部时钟(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(可在停止模式下工作)
- 看门狗被激活后，则在计数器递减至 0x000 时产生复位
- 具有硬件看门狗功能：在用户选项字节里将 IWDG 配置成硬件启动模式后，IWDG 会在复位后自动使能
- 可自动使能 LSI：IWDG 使能后，自动使能 LSI，并作为 IWDG 的内核时钟

25.3 IWDG 功能说明

25.3.1 模块框图

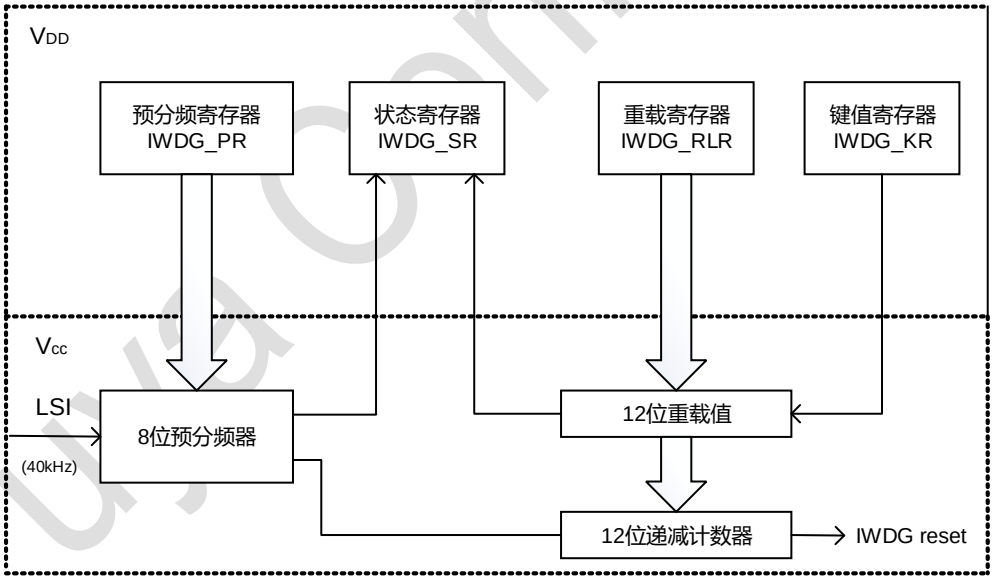


图 25-1 IWDG 模块框图

注： 看门狗功能处于 Vcc 供电区，即在停模式时仍能正常工作。

在键寄存器(IWDG_KR)中写入 0xCCCC，开始启用独立看门狗；此时计数器开始从其复位值 0xFFF 递减计数。当计数器计数到末尾 0x000 时，会产生一个复位信号(IWDG_RESET)。

无论何时，只要键寄存器 IWDG_KR 中被写入 0xAAAA， IWDG_RLR 中的值就会被重新加载到计数器中从而避免产生 IWDG 复位 。

表 25-1 看门狗超时时间 40 kHz 的输入时钟(LSI)

预分频系数	PR[2:0]位	最短时间(ms) PR[11:0] = 0x000	最长时间(ms) PR[11:0] = 0xFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	6	6.4	26214.4

注：这些时间是按照 40 kHz 时钟给出。实际上，微控制器内部的 RC 频率会在 30 kHz 到 60 kHz 之间变化。此外，即使 RC 振荡器的频率是精确的，确切的时序仍然依赖于 APB 接口时钟与 RC 振荡器时钟之间的相位差，因此总会有一个完整的 RC 周期是不确定的。通过对 LSI 进行校准可获得相对精确的看门狗超时时间。

25.3.2 硬件看门狗

如果用户在选择字节中启用了“硬件看门狗”功能，在系统上电复位后，看门狗会自动开始运行；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

25.3.3 寄存器保护

对 IWDG_PR 和 IWDG_RLR 寄存器的写访问是被保护的，要修改这两个寄存器的值，IWDG_KR 必须先被写入 0x5555。对这些寄存器的写其他数将重新开启写保护，比如写入 0xAAAA，IWDG_PR 和 IWDG_RLR 将被再次保护。

IWDG_SR 指示 IWDG_PR 或 IWDG_RLR 的值是否正在更新。

25.3.4 调试模式

如果 CPU 进入调试模式，根据调试模块中的 DBG 模块中 DBG_IWDG_STOP 配置位的状态，IWDG 的计数器继续工作或停止

25.3.5 低功耗冻结

取决于选项字节里的 IWDG_STOP 位，决定了 IWDG 停止模式下是否继续保持计数。如果 IWDG 在停止模式下保持工作，IWDG 能从当前的低功耗模式下唤醒微控制器。

25.4 IWDG 寄存器

只能字的方式操作以下寄存器。

25.4.1 IWDG 键值寄存器（IWDG_KR）

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	KEY[15:0]	W	16’h0	键值。

				软件必须以一定的时间间隔向该寄存器写入 0xAAAA，否则，当计数器计数到 0 时，看门狗会产生复位。 写入 0x5555：允许访问 IWDG_PR 和 IWDG_RLR； 写入 0xCCCC：激活 IWDG（如果选择了硬件看门狗则不受此命令控制）。

25.4.2 IWDG 预分频寄存器（IWDG_PR）

偏移地址：0x04

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:3	Reserved	-	-	保留
2:0	PR[2:0]	RW	3'h0	预分频值。 通过配置该寄存器选择计数器时钟的预分频值。 要改变该寄存器，IWDG_SR 寄存器的 PVU 必须为 0。 此寄存器受写保护。向 IWDG_KR 写入 0x5555 可解除写保护。 000：4 分频 001：8 分频 010：16 分频 011：32 分频 100：64 分频 101：128 分频 110：256 分频 111：256 分频

25.4.3 IWDG 重装载寄存器（IWDG_RLR）

偏移地址：0x08

复位值：0x0000 0FFF

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

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11:0	RL[11:0]	RW	12'h FFF	IWDG 计数器重装载值。 当向 IWDG_KR 寄存器写入 0xAAAA 时，RL 值会传送到计数器中。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 RL 值和时钟预分频值来计算。 此寄存器受写保护。向 IWDG_KR 写入 0x5555 可解除写保护。只有当 IWDG_SR.RVU=0 时，才能对寄存器进行修改。

25.4.4 IWDG 状态寄存器（IWDG_SR）

偏移地址：0x0C

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:2	Reserved	-	-	保留
1	RVU	R	0	看门狗计数器重装值更新。 此位由硬件置 1，表明重装载值正在更新。当重装载值更新结束后，此位由硬件清零。
0	PVU	R	0	看门狗预分频值更新。 此位由硬件置 1，表明预分频值正在更新。当预分频值更新结束后，此位由硬件清零。

注：在更新 IWDG_PR 和 IWDG_SR.RLR 前，要分别等待 IWDG_PVU 和 IWDG_SR.RVU 为 0.但在更新 IWDG_PR、IWDG_RLR 后，不必再等待 IWDG_SR.PVU、IWDG_SR.RVU 为 0，可继续执行后续的代码。

26. 窗口看门狗（WWDG）

26.1 WWDG 简介

窗口看门狗通常用来监测由外部干扰或不可预见的逻辑条件造成的应用程序跑飞而产生的软件故障。除非递减计数器的值在 T[6]位变成 0 前被刷新，看门狗电路在达到预置的时间周期时，会产生一个复位。在递减计数器达到窗口寄存器数值之前，如果 7 位的递减计数器数值(在控制寄存器中)被刷新， 那么也将产生一个复位。这表明递减计数器需要在一个有限的时间窗口中被刷新。

26.2 WWDG 主要特性

- 可编程的自由运行递减计数器
- 若看门狗被启动，满足以下任一条件时，产生系统复位:
  - 递减计数器的值小于 0x40
  - 递减计数器在窗口外被重新装载
- 如果启动了看门狗并且开启早期唤醒中断(EWI)，则当递减计数器等于 0x40 时产生 EWI，软件可在 EWI 中断程序内重装载计数器以避免 WWDG 复位。
- 具有硬件看门狗功能：在用户选项字节里将 WWDG 配置成硬件启动模式后，WWDG 会在复位后自动使能

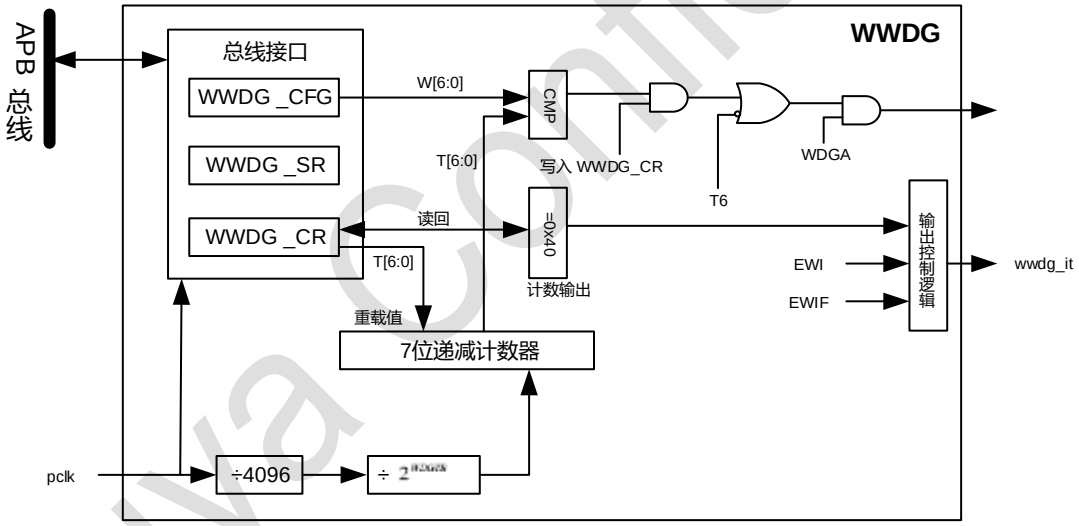


图 26-1 WWDG 模块框图

26.3 WWDG 功能描述

如果看门狗被启动(WWDG_CR 寄存器中的 WDGA 位被置”1”)，并且当 7 位(T[6:0])递减计数器从 0x40 翻转到 0x3F(T[6]位清零)时，产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重新装载计数器，将产生一个复位。

软件在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止微控制器发生复位。只有当计数器值小于窗口寄存器的值时，才能进行写操作。储存在 WWDG_CR 寄存器中的数值必须在 0xFF 和 0xC0 之间:

- 启动看门狗

当选项字节里的 WWDG_SW 选择了软件启动 WWDG 时(WWDG_SW=1),系统复位之后 WWDG 处于关闭状态，将 WWDG_CR 里的 WDGA 为设为 1 即可使能 WWDG。WWDG 一旦被软件启动就无法被关闭，除非发生了系统复位。

当选项字节里的 WWDG_SW 选择了硬件启动 WWDG 时(WWDG_SW=0),系统复位之后 WWDG 时钟自动启动，且 WWDG 处于开启状态，且无法被关闭。

■ 控制递减计数器

递减计数器处于自由运行状态，即使看门狗被禁止，递减计数器仍继续递减计数。当看门狗被启用时，T[6]位必须被置 1，以防止立即产生一个复位。

T[5:0]位包含了看门狗产生复位之前的计时数目；复位前的延时时间在一个最小值和一个最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频值是未知的。

配置寄存器(WWDG_CFR) 中包含窗口的上限值: 要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载， 图 4-1 描述了窗口寄存器的工作过程。

另一个重装载计数器的方法是利用早期唤醒中断(EWI)。设置 WWDG_CFR 寄存器中的 EWI 位开启该中断。当递减计数器到达 0x40 时产生此中断，相应的中断服务程序(ISR)可以用来加载计数器以防止 WWDG 复位。在 WWDG_SR 寄存器中写'0'可以清除该中断。

注： 可以用 T[6]位产生一个软件复位(设置 WDGA 位为"1"， T[6]位为"0")。

26.3.1 如何编写看门狗超时程序

可以使用下图中提供的公式计算窗口看门狗的超时时间。

警告：当写入 WWDG_CR 寄存器时，始终置 T[6] 位为'1'以避免立即产生一个复位。

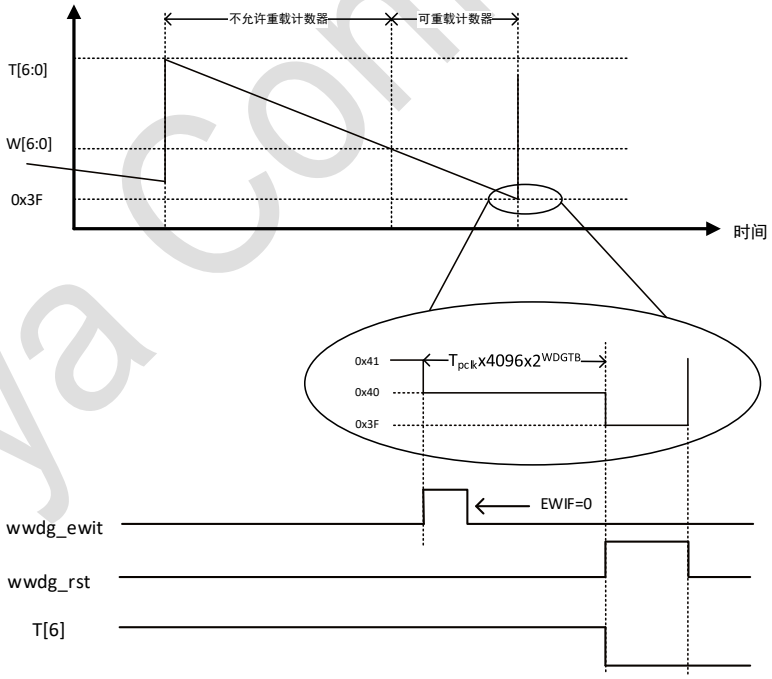


图 26-2 窗口看门狗时序图

计算 WWDG timeout 值的公式如下:

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WWDGTB[1:0]} \times (T[5:0] + 1) \quad (ms)$$

26.3.2 调试模式

当微控制器进入调试模式时(Cortex-M4 核心停止)，根据调试模块中的 DBG_WWDG_STOP 配置位的状态， 决定 WWDG 的计数器继续工作或停止。详见调试模式章节。

26.4 WWDG 寄存器描述

26.4.1 WWDG 控制寄存器（WWDG_CR)

偏移地址：0x00

复位值：0x0000 007F

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7	WDGA	RS	0	激活位 (Activation bit) 此位由软件置”1”,但仅能由硬件在复位后清”0”。当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗 1：启用看门狗
6:0	T[6:0]	RW	7’h7F	7 位计数器(MSB 至 LSB) (7-bit counter) 这些位用来存储看门狗的计数器值。每(4096x2^WDGTB)个 PCLK1 周期减 1。当计数器值从 0x40 变 为 0x3F 时(T[6]变成 0)，产生看门狗复位。

26.4.2 WWDG 配置寄存器（WWDG_CFR)

偏移地址：0x04

复位值：0x0000 007F

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
9	EWI	RS	0	提前唤醒中断 (Early wakeup interrupt) 此位若置’1’，则当计数器值达到 0x40，即产生中断。 此中断只能由硬件在复位后清除。
8:7	WDGTB[1:0]	RW	2’h0	时基 (Timer base) 预分频器的时基可以设置如下： 00: CK 计时器时钟(PCLK1 除以 4096)除以 1 01: CK 计时器时钟(PCLK1 除以 4096)除以 2 10: CK 计时器时钟(PCLK1 除以 4096)除以 4 11: CK 计时器时钟(PCLK1 除以 4096)除以 8

6:0	W[7:0]	RW	7'h7F	7 位窗口值 (7-bit window value) 这些位包含了用来与递减计数器进行比较用的窗口值。

26.4.3 WWDG 状态寄存器（WWDG_SR）

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res	Res	Res	Res	Res	Res	Res	Res	Res	EWIF
															RC_W0

Bit	Name	R/W	Reset Value	Function
31:1	Reserved	-	-	保留
0	EWIF	RC_W0	0	提前唤醒中断标志 (Early wakeup interrupt flag) 当计数器值达到 0x40 时，此位由硬件置"1"。它必须通过软件写"0"来清除。对此位写"1"无效。若中断未被使能，此位也会被置"1"。

27. 串行外设接口/集成电路内置音频总线(SPI/I²S)

27.1 SPI/I²S 简介

SPI/I²S 接口可以配置为支持 SPI 协议或者支持 I²S 音频协议。SPI 接口默认工作在 SPI 方式，可以通过软件把功能从 SPI 模式切换到 I²S 模式。

串行外设接口(SPI)允许芯片与外部设备以半双工、全双工、单工同步的串行方式通信。此接口可以被配置成主模式，并为外部从机提供通信时钟(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。

I²S 也是同步串行接口通讯协议。它支持四种音频标准，包括飞利浦 I²S 标准，MSB 和 LSB 对齐标准，以及 PCM 标准。它在半双工通讯中，可以工作在主和从两种模式下。当它作为主机时，通过接口向外部的从机提供时钟信号。

27.2 SPI/I²S 主要特性

27.2.1 SPI 主要特性

- 支持主机或者从机模式
- 3 线全双工同步传输
- 2 线半双工同步传输（有双向数据线）
- 2 线单工同步传输（无双向数据线）
- 8 位或者 16 位传输帧选择
- 支持多主模式
- 8 个主模式波特率预分频系数
- 从模式频率
- 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
  - CRC 错误标志
- 支持 Motorola 和 TI 模式
- 可触发引起中断的主模式故障、过载标志
- 2 个具备 DMA 能力的深度为 4，宽度为 16bit（当数据帧设置为 8bit 时，宽度为 8bit）的嵌入式 Rx 和 Tx FIFOs

27.2.2 I²S 主要特性

- 半双工通信(仅发送或接收)
- 主或者从操作

- 8 位线性可编程预分频器，获得精确的音频采样频率(8 kHz~ 192 kHz)
- 数据格式可以是 16 位，24 位或者 32 位
- 音频信道固定数据包帧为 16 位(16 位数据帧)或 32 位(16、24 或 32 位数据帧)
- 可编程的时钟极性(稳定态)
- 从发送模式下的下溢标志位、接收模式下的上溢标志位（主或从），以及接收和发送模式下的帧错误标志（仅适用于从机）
- 16 位数据寄存器用来发送和接收，在通道两端各有一个寄存器
- 支持的 I²S 协议：
  - I²S 飞利浦标准
  - MSB 对齐标准(左对齐)
  - LSB 对齐标准(右对齐)
  - PCM 标准(16 位通道帧上带长或短帧同步或者 16 位数据帧扩展为 32 位通道帧)
- 数据方向总是 MSB 在先
- 发送和接收都具有 DMA 能力
- 主时钟可以输出到外部音频设备，比率固定为 256xFs(Fs 为音频采样频率)

27.3 SPI 功能描述

27.3.1 SPI 框图

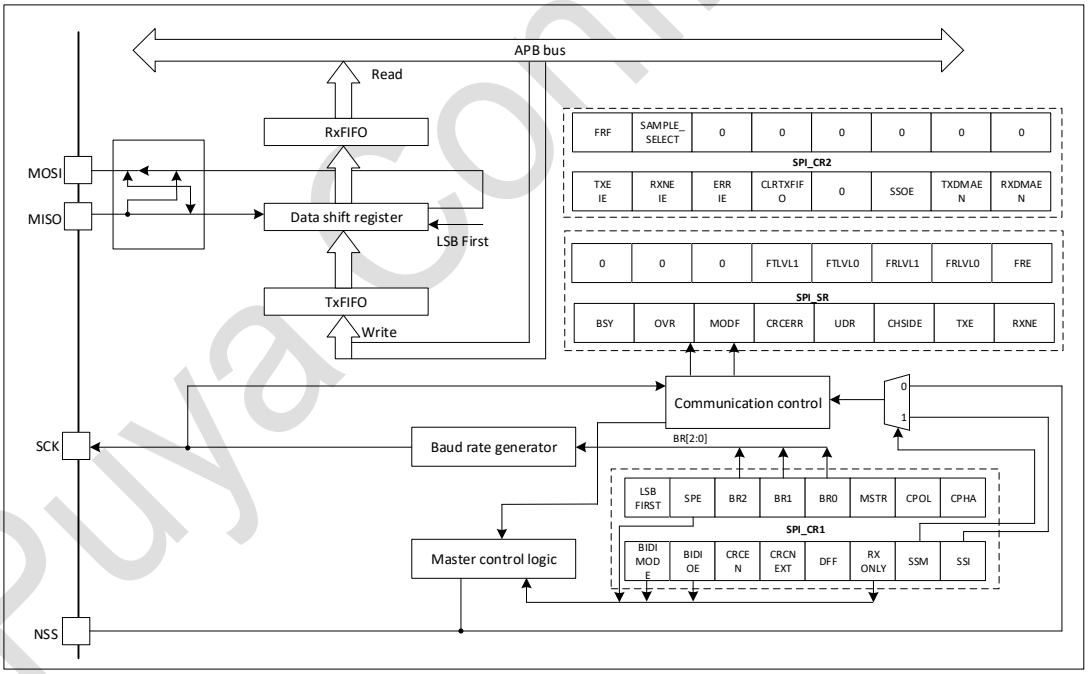


图 27-1 SPI 框图

SPI 通过 4 个引脚与外部器件相连：

MISO：主机输入/从机输出引脚。该引脚在从模式下发送数据，在主模式下接收数据。

MOSI：主机输出/从机输入引脚。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，作为主机的输出，从机的输入。

NSS：从机选择。取决于 SPI 和 NSS 的设定，该引脚可以用作：

— 选择要通讯的从机

– 同步数据帧 或者

– 检测多主机间的冲突

SPI 总线允许在一个主机和一个或者多个从机之间的通讯。总线由至少两根线组成：一个是时钟，另一个是用于同步传输的数据。根据应用场景，可以选择增加另外一根数据线和从机选择信号。

27.3.2 一个主机和一个从机的通信

针对不同的应用场景，SPI 可以使用几种不同的配置进行通讯。这些配置使用 2 线、3 线 (软件 NSS)，或者 3 线、4 线（硬件 NSS）。通讯始终由主机启动。

27.3.2.1 全双工通信

默认情况下，SPI 被配置成全双工通讯。在这种配置下，主机和从机的移位寄存器，在 MOSI 和 MISO 引脚之间，使用两个单向的线连到一起。在 SPI 通讯期间，数据在主机提供的时钟沿同步的被移位。主机通过 MOSI 线发送数据，从 MISO 线接收来自从机的数据。当数据帧传输完成（所有位完成移位），在主机和从机之间即完成信息交换。

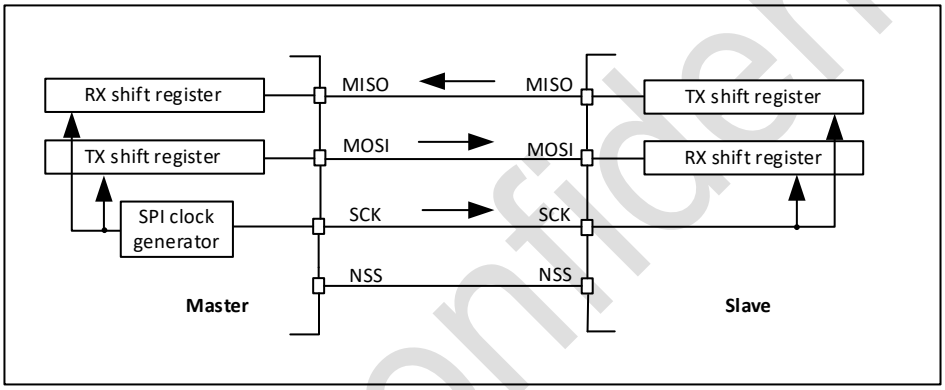


图 27-2 全双工单主/单从机应用

27.3.2.2 半双工通信

通过设定 BIDIMODE 位（SPI_CR1 寄存器），SPI 可以工作在半双工模式。在这种配置下，用 1 根数据线完成主机和从机移位寄存器的连接。在通讯过程中，在 SCK 的时钟沿，数据在两个移位寄存器之间以 BIDIOE（SPI_CR1 寄存器）选择的方向，同步移位。在该配置下，主机的 MISO 引脚和从机的 MOSI 引脚被释放作为 GPIO 给其他应用使用。

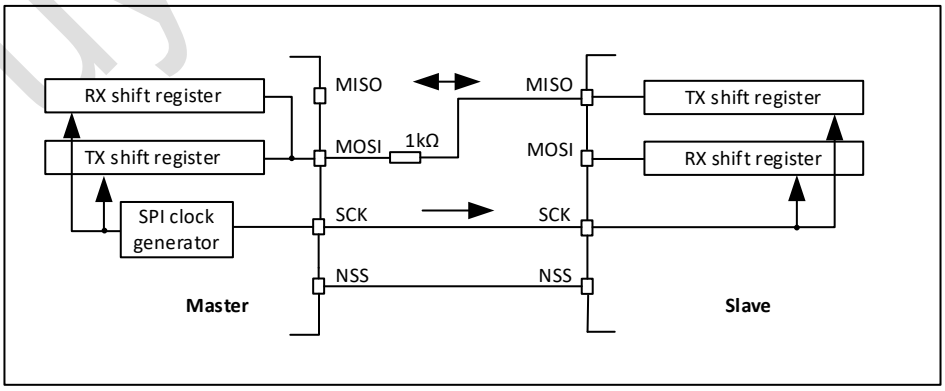


图 27-3 半双工单主/从机应用

（1）NSS 引脚可以被使用在主机和从机之间进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

（2）在该配置下，主机的 MISO 引脚和从机的 MOSI 引脚可以用作 GPIO。

（3）当以双向模式工作的两个节点间的通信方向不是同步变化时，会出现临界情况，新发送器访问共用数据线，而前一个发送器仍保持线路上的相反值（值取决于 SPI 配置和通信数据）。两个节点会出现冲突，在共用线上短暂提供相反的输出电平，直到下一个节点也相应地改变其方向设置。建议此模式下在 MISO 和 MOSI 引脚之间插入串行电阻以在这种情况下保护输出并限制电流在二者之间流过。

27.3.2.3 单工通信

通过使用 RXONLY（SPI_CR1 寄存器）位将 SPI 设置为只发送模式或只接收模式，可使 SPI 以单工模式进行通信。在这个配置下，在主机和从机的移位寄存器之间只使用 1 根线。另一对 MISO 和 MOSI 引脚不被使用，可以被释放成 GPIO。

- 只发送模式（RXONLY=0）：配置与全双工模式相同。应用忽略在未使用的输入引脚上的信息。这个引脚可以被用作标准的 GPIO。
- 只接收模式（RXONLY=1）：通过置位 RXONLY，应用可以禁止 SPI 输出功能。
  - 在从机模式配置下，MISO 输出被禁止，该引脚被用作 GPIO。在从机选择信号有效时，从机继续从 MOSI 引脚接收数据。接收到的数据事件是否发生取决于数据缓冲区的配置。
  - 在主机模式配置下，MOSI 输出被禁止，引脚可以用作 GPIO。只要 SPI 处于使能状态，便不断生成时钟信号。停止时钟的唯一方式是将 RXONLY 位或 SPE 位清零，直至来自 MISO 引脚的传入模式结束，然后基于相应配置填充数据缓冲区结构。

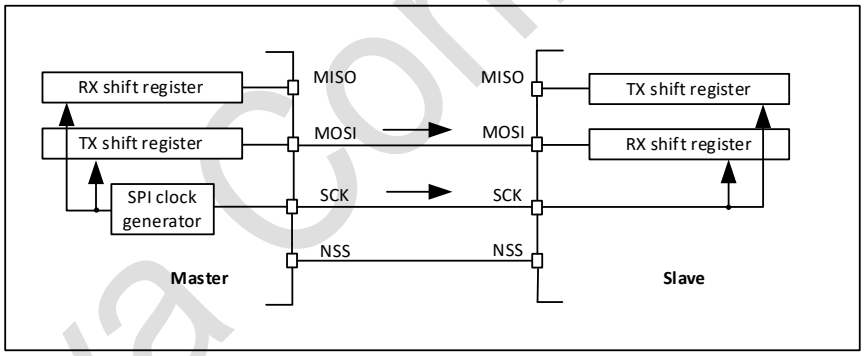


图 27-4 单工单主/从机应用(主机只发送/从机只接收模式)

（1）在主机和从机之间可以使用 NSS 引脚进行硬件控制流。可选的，NSS 也可以不使用。然后该流程就要内部处理。

（2）在 Rx 移位寄存器的输入上捕获意外的输入信息。在标准的只发送模式下，所有与传输接收相关的事件都被必须被忽略（如 OVF 标志）。

（3）在该配置下，两边的 MISO 引脚都被用作 GPIO。

注：任何单工通信都可以通过固定半双工模式中的方向设置来实现（使能双向模式，同时 BDIO 位保持不变）。

27.3.3 标准多从机通信

在一个有两个或者更多个独立从机的配置里，主机使用 GPIO 引脚为每个从机，来管理片选线。主机必须通过拉低与从机 NSS 输入相连的 GPIO 电平来选择某个从机。执行该操作后，便建立了标准主机和专门的从机之间的通讯。

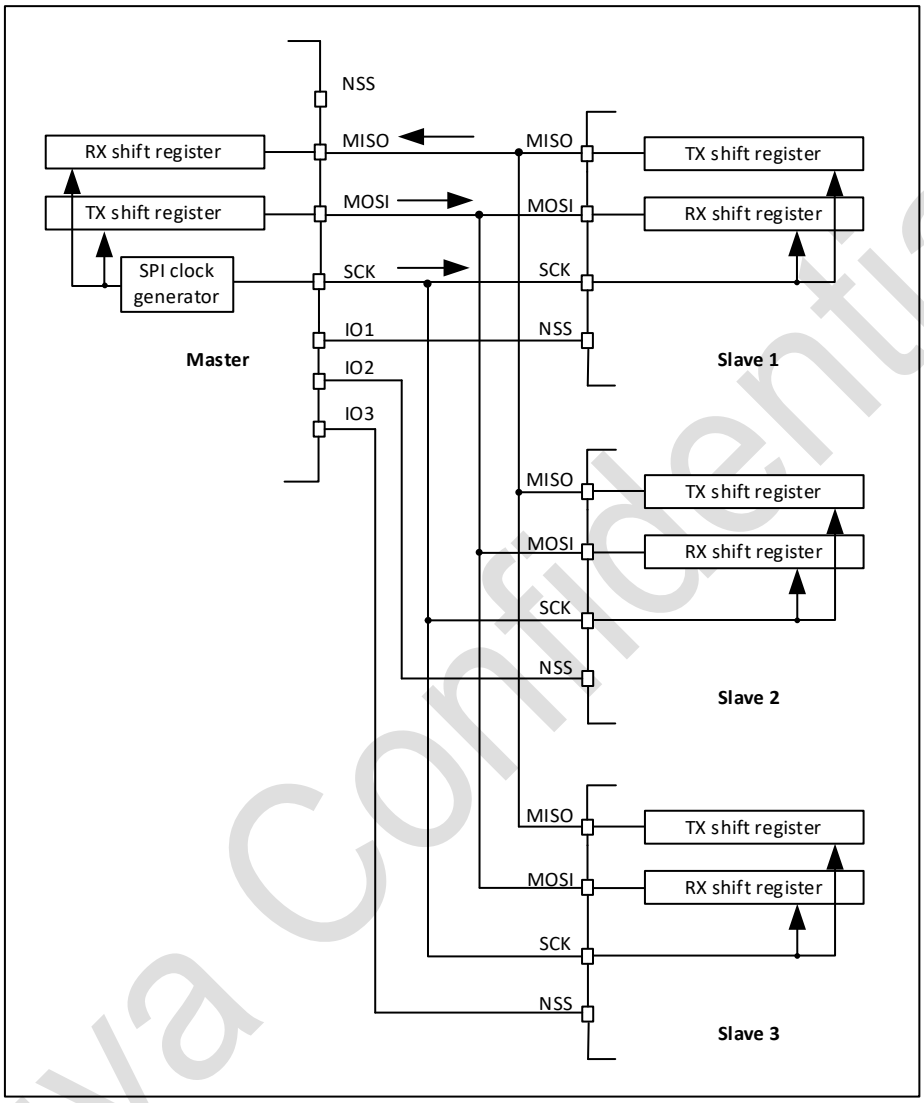


图 27-5 主机选择 3 个从机

在这种配置下不使用 NSS 引脚。必须通过 SSM=1, SSI=1 来防止任何 MODF 错误。

由于从机的 MISO 引脚连接到一起，所有从机 MISO 引脚的 GPIO 必须配置为开漏复用功能。

27.3.4 多主机通信

如果 SPI 总线未用于多主机功能，用户可使用内置功能来检测试图同时控制总线的两个节点间是否存在潜在冲突。对于该检测，NSS 引脚配置为硬件输入模式。

由于此时只有一个结点可将其输出施加到公用数据线上，因此该模式连接的 SPI 节点不能超过两个。当节点无效时，默认情况下均保持从机模式。一旦一个节点要接管对总线的控制，它会将自身切换到主机模式，然后通过专用 GPIO 引脚向其他节点的从机选择输入施加有效电平。会话完成后，有效的从机选择信号将被释放，控制总线的节点会短暂切换回被动从模式，等待下一个会话开始。

如果两个节点同时发出各自的控制请求，则会出现总线冲突（模式故障 MODF 事件）。随后，用户可应用某个简单的仲裁过程（例如，在两个节点上施加不同的预定义超时来推迟下一次尝试） 。

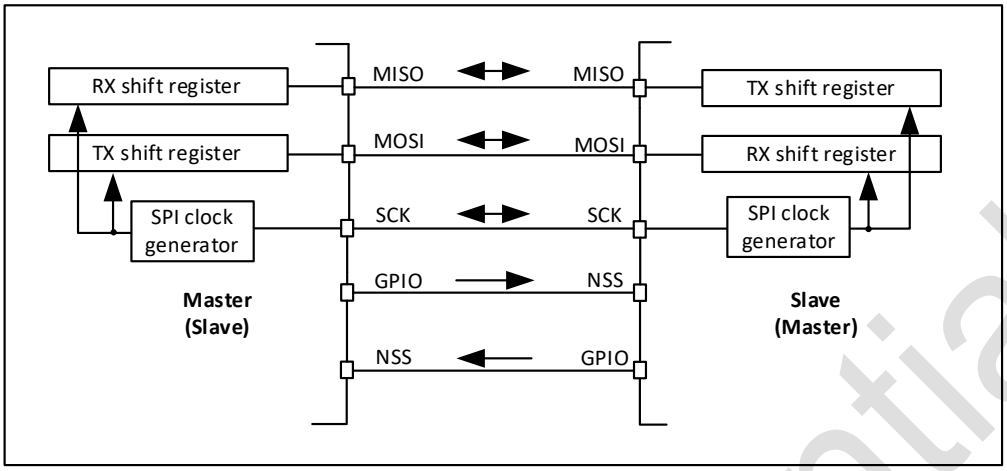


图 27-6 多主机应用

NSS 引脚在两个节点都被配置成硬件输入模式。当被动节点被配置成从机时，其有效电平将使能 MISO 输出控制。

27.3.5 从机选择接口 (NSS)

在从机模式，NSS 作为标准的片选输入，使从机能与主机通讯。在主机模式，NSS 既可以作为输出又可以作为输入。当作为输入时，它可以防止多主机的总线冲突，当作为输出时，它可以驱动单个从机的从机选择信号。

可以使用 SPI_CR1 寄存器的 SSM 位设置硬件或者软件从机选择管理：

- 软件 NSS 管理（SSM=1）：在这个配置下，由 SPI_CR1 寄存器中的 SSI 位的值内部驱动从机选择信息。外部 NSS 引脚空闲，可供其他应用使用。
- 硬件 NSS 管理（SSM=0）：在这个情况下，有几个可能的配置：
  - NSS 输出使能（SSM=0, SSOE=1）：这个配置仅在作为主机时使用。硬件管理 NSS 引脚。只要在主机模式下使能 SPI (SPE=1)， NSS 信号便会被驱动为低电平，并且会一直保持低电平状态，直至关闭 SPI (SPE =0)。在多主机应用中，SPI 不能进行这种 NSS 配置。
  - NSS 输出禁止（SSM=0, SSOE=0）：如果 MCU 在总线上作为主机，此配置可实现多主模式功能。如果在该模式下将 NSS 引脚拉至低电平，SPI 将进入主模式故障状态，器件将在从模式下自动进行重新配置。在从机模式，NSS 引脚作为标准的片选输入，当 NSS 为低时选择从机。

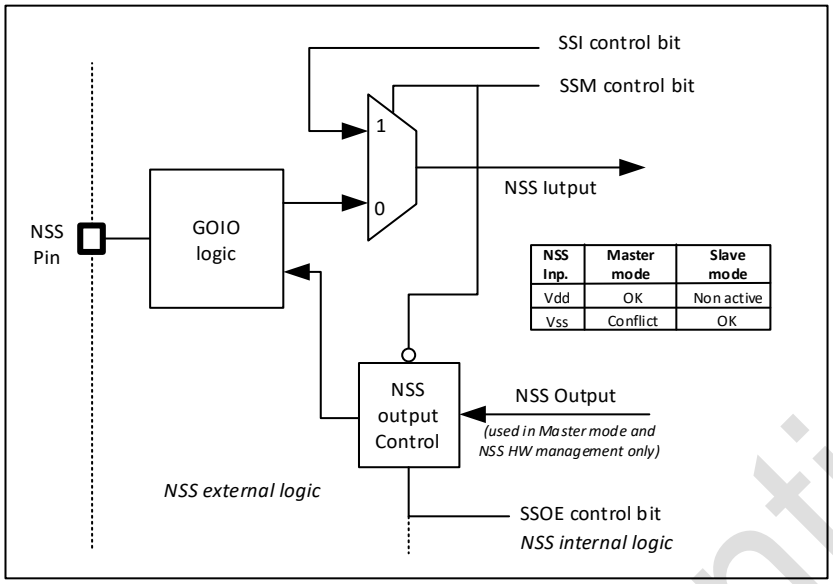


图 27-7 硬件/软件从模式选择管理

27.3.6 通信格式

在 SPI 通讯期间，接收和发送操作同时进行。串行时钟（SCK）将数据线上的信息移位和采样操作同步。通讯格式取决于时钟相位、时钟极性和数据帧格式。为了能够进行通讯，主机和从机必须遵循相同的通讯格式。

27.3.6.1 时钟相位和极性控制

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）位控制不传输任何数据时的时钟空闲状态值。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 位置 1，则会在 SCK 引脚的第二个边沿捕获传输的第一个数据位（如果复位 CPOL 位，则为下降沿；如果将 CPOL 位置 1，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。如果将 CPHA 位复位，则会在 SCK 引脚的第一个边沿捕获传输的第一个数据位（如果将 CPOL 位置 1，则为下降沿；如果将 CPOL 位复位，则为上升沿）。即，在每次出现该时钟边沿时锁存数据。

CPOL（时钟极性）和 CPHA（时钟相位）位的组合用于选择数据捕获时钟边沿。

在 CPOL/CPHA 改变之前，SPI 必须被禁止（SPE=0）。

SCK 的空闲状态必须与 SPI_CR1 寄存器选择的极性对应。

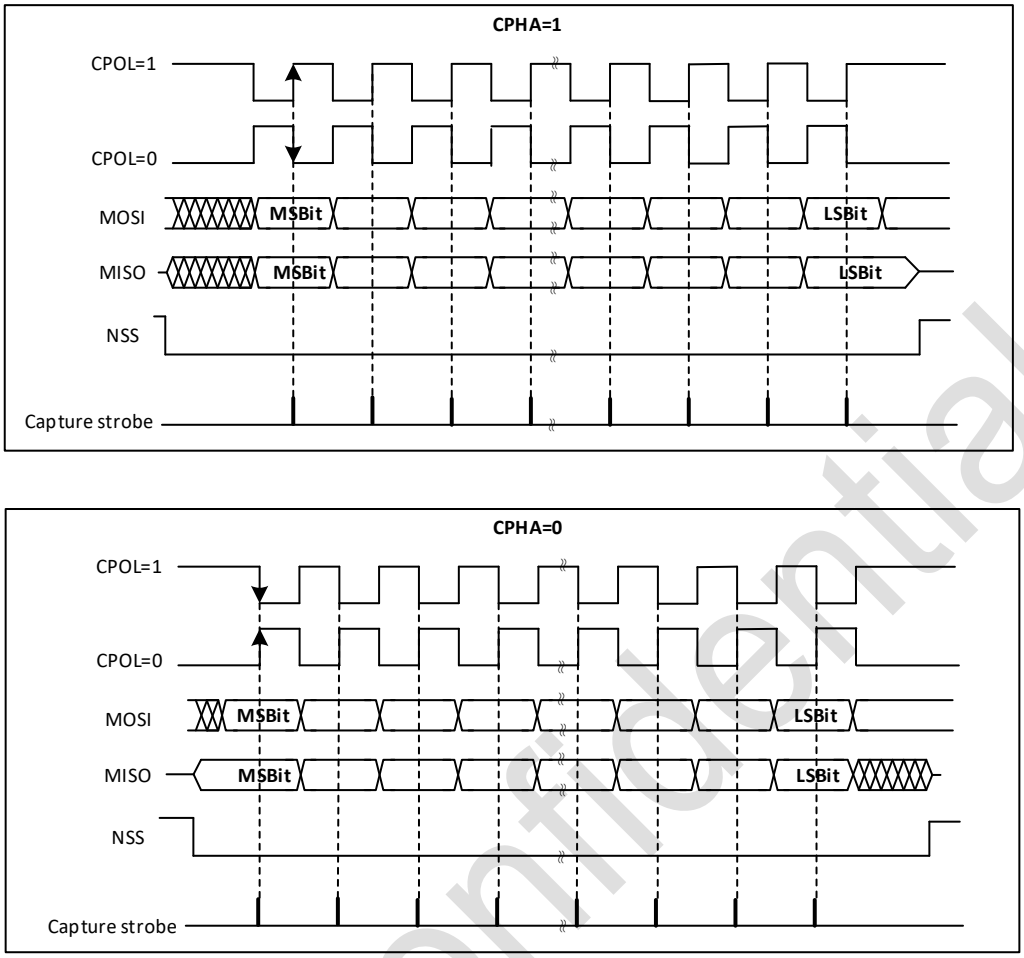


图 27-8 时钟/数据时序图

数据位的顺序取决于 LSBFIRST 位的设置。

注：在作为 SPI 主机高速时，外部迟延大时。可以使能 SAMPLE_SELECT 位（SPI_CR2 寄存器），来选择采样时间点。默认主机接收数据为 1/2 个 cycle 采样，使能后将转变为 1 个 cylce 采样。

27.3.6.2 数据帧格式

SPI 移位寄存器可设置为以 MSB 在前或 LSB 在前的方式移出数据,具体取决于 LSBFIRST 位的值。通过使用 DFF 位(SPI_CR1 寄存器)，选择数据帧的位数。可选择为 8 位或者 16 位长度，该设置对于发送和接收都适用。

27.3.7 配置 SPI

对于主机和从机，SPI 的配置流程几乎一样。对于具体的模式建立，遵循专门的章节介绍。当进行标准的通讯，进行以下步骤：

1. 写相关的 GPIO 寄存器：配置 MOSI、MISO 和 SCK 引脚
2. 写 SPI_CR1 寄存器
   1) 通过 BR[2:0]配置时钟波特率（从机模式不需要）
   2) 配置 CPOL 和 CPHA，定义数据传输和串行时钟之间的关系
   3) 通过 RXONLY 或者 BIDIMODE 和 BIDIOE（RXONLY 和 BIDIMODE 不能同时有效），选择单工或者半双工模式
   4) 配置 LSBFIRST 定义帧格式

5) 如果需要 CRC，则配置 CRCL 和 CRCEN 位（ SCK 时钟信号处于空闲状态时）
6) 配置 SSM 和 SSI
7) 配置 MSTR 位 (在多主机 NSS 配置中，如果主机被配置防止 MODF 错误，要避免 NSS 的冲突状态)
8) 配置 DFF 位，选择数据帧位数

3. 写 SPI_CR2 寄存器
   1) 配置 SSOE（从机模式不需要）
4. 写 SPI_CRCPR 寄存器：需要时配置 CRC 多项式
5. 写相应的 DMA 寄存器（包括 SYSCFG_CFGR3 和 SYSCFG_CFGR4 配置 DMA 通道）：配置 DMA 的 SPI TX 和 Rx 通道

27.3.8 使能 SPI 步骤

建议在主机发送时钟前使能 SPI 从机。否则，数据传输可能会不正常。从机的数据寄存器必须包含待发送的数据才能开始与主机通信（在通信时钟的第一个边沿；如果时钟信号连续，则是在正在进行的通信结束前）。使能 SPI 从机前，SCK 信号必须稳定为所选极性对应的空闲状态电平。

当 SPI 被使能并且 TXFIFO 不空，或者向 TXFIFO 进行下一个写操作，全双工模式（或者只发送）的主机开始通讯。

在任何主机只接收模式（RXONLY=1，或者 BIDIMODE=1 且 BIDIOE=0），SPI 使能后，主机开始通讯且主机立即开始提供时钟。

对于 DMA 处理，遵从具体的章节内容。

27.3.9 数据发送和接收

27.3.9.1 RXFIFO 和 TXFIFO

SPI 所有数据通讯都通过深度为 4，宽度为 16bit（当数据帧设置为 8bit 时，宽度为 8bit）的 FIFO。该特性使 SPI 能够以连续数据流进行工作，并防止由于 CPU 来不及处理数据导致的通讯问题。发送和接收有独立的 FIFO，叫做 TXFIFO 和 RXFIFO。这些 FIFO 被用在所有的 SPI 模式。

FIFO 的处理取决于多种参数，包括：数据交换模式（全双工、半双工）、数据帧格式。

读 SPI_DR 寄存器会得到最早存放在 RXFIFO 中还未被读走的数据结果。写 SPI_DR 寄存器，会在 FIFO 发送队列的最后位置，存入被写的数据。读写访问必须与数据宽度对齐。FTLVL[1:0]和 FRLVL[1:0]位显示了两个 FIFO 当前的占用级别。

对 SPI_DR 寄存器的读访问必须通过 RXNE 事件管理。当 RXFIFO 数据非空，该事件被触发。当 RXNE 被清零，RXFIFO 就被认为是空的。

相似地，写要发送的数据帧，通过 TXE 事件管理。当 TXFIFO 的占用级别小于或者等于总容量的一半时，该事件就会被触发。否则，TXE 被清零，并且 TXFIFO 被认为是满的。

用这样的方式，RXFIFO 和 TXFIFO 都可以存 4 个数据帧。

TXE 和 RXNE 事件都都可以通过查询或者中断方式处理。

另一种管理数据交换的方式是使用 DMA。

如果在 RXFIFO 已满时收到下一个数据，将发生上溢事件（OVR 标志）。上溢事件可通过轮询或中断方式来处理。

BSY 位被置 1 表示当前正在处理数据帧。当时钟信号连续运行时，在主机中，BSY 标志在数据帧之间保持置 1 状态，但在从机中，BSY 标志在数据帧传输之间变为低电平并持续最短的一段时间（一个 SPI 时钟）。

某些应用场景下，当向 TXFIFO 中写入数据，可以通过设置 CLRTXFIFO 位，清空 TXFIFO 数据，从而往 TXFIFO 里写新的数据重新进行通信。

27.3.9.2 序列处理

一些数据帧可以通过单序列传递来完成一条信息。使能发送后，当主机的 TXFIFO 里有任何数据，序列开始，只要主器件的 TXFIFO 中存在数据便一直继续。时钟信号由主机连续提供，直到 TXFIFO 为空，然后时钟信号停止，等待其他的数据。

在只接收模式，即半双工（BIDIMODE=1, BIDIOE=0）或者单工模式（BIDIMODE=0,RXONLY=1），当使能 SPI 并且只激活接收模式时，主机就立即开始接收。主机一直会提供时钟并连续地接收数据，直到主机停止 SPI 或者关闭只接收模式。

当主机能够以连续的模式（SCK 信号是连续的）提供所有通讯，主机必须要考虑从机处理数据流的能力。当有必要时，主机必须降低通讯速度，并提供更慢的时钟，或者或带有足够延时的单独帧或数据段。要注意的是，对于主机或者从机来说，没有下溢错误信号，来自从机的数据始终由主机处理，即使从机无法及时正确地准备数据也是如此。对于从机，更好的方法是使用 DMA，尤其当数据帧较短且总线速率较高的情况。

在多从器件系统中，每个序列必须通过 NSS 脉冲进行控制，从而只选择其中一个从器件进行通信。在单个从器件系统中，无需通过 NSS 来控制从器件，但此时提供此脉冲通常会更好，以在每个数据序列开始时同步从器件。NSS 可以由软件和硬件两种方式管理。

当 BSY 位置 1 时，表示正在处理数据帧事务。当所进行的帧交互完成时，RXNE 标志将置 1。最后一位采样后，整个数据帧会存储到 RXFIFO 中。

27.3.9.3 关闭 SPI 步骤

当关闭 SPI 时，必须按照本段中介绍的关闭步骤进行操作。当外设时钟停止时，在系统进入低功耗模式前做到这一点是十分重要的。否则会损坏正在进行的交互。在某些模式下，禁止步骤是停止所进行的连续通信的唯一方式。

全双工或者只发送模式下，主机可以在停止提供要发送的数据时完成交互。在这种情况下，在最后的数据交互后，时钟被停止。在这些模式下，SPI 被禁止之前，用户必须使用标准的禁止流程。当 SPI 在主机发送时，如果此时一个帧交互正在进行，或者下一个数据帧存在 TXFIFO 中，此时关闭 SPI，则 SPI 的的状态是不可预测的。

当主机处在只接收模式，停止连续时钟的唯一方法是关闭外设（SPE=0）。这必须在最后一个数据帧传输内的特定时间段，即第一位采样与最后一位传输开始之间完成（以便接收全部数量的预期数据帧并防止在最后一个有效数据帧后读取任何其他的“空”数据）。在该模式下关闭 SPI 时必须遵循特定步骤。

关闭 SPI 后，已接收但未读取的数据始终存储在 RXFIFO 中，这些数据必须在下次使能 SPI 后进行处理，然后才能启动新序列。为防止存在未读取的数据，需确保关闭 SPI 时 RXFIFO 为空，可通过正确的关闭步骤来关闭 SPI，也可以通过控制外设复位专用的特定寄存器以软件复位的方式来初始化所有 SPI 寄存器从而关闭 SPI（RCC_APBRSTx 寄存器）。

标准的禁止流程是基于 BSY 状态，并查看 FTLVL[1:0]，以确保传输彻底完成。还可以在必须识别正在处理的传输是否结束的特定情况下完成这种检查，例如：

- 当 NSS 信号被软件管理，主机要向从机提供正确的 NSS 脉冲。 或者
- 最后的数据帧或者 CRC 帧仍在传输过程中，来自 DMA 或者 FIFO 的交互数据流完成。

正确的禁止流程是（只接收模式除外）：

1. 等待 FTLVL[1:0]=00（没有数据要发送）

2. 等待 BSY=0（最后的数据被处理完成）
3. 关闭 SPI（SPE=0）
4. 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

对于特定只接收模式，正确的禁止流程是：

1. 在最后一个数据帧传输过程中，通过禁止 SPI（SPE=0），打断接收流程
2. 等待 BSY=0（最后的数据帧已被处理）
3. 读数据，直到 FRLVL[1:0]=00（读所有接收到的数据）

27.3.9.4 DMA 通信

为了以最大速度工作并且方便避免上溢所需的数据寄存器读/写过程，SPI 提供了 DMA 功能，该功能采用了简单的请求/应答协议。

当 TXE 或者 RXNE 置位，会产生 DMA 请求。Tx 和 Rx 缓冲区有独立的请求。

- 发送时，每次 TXE 置为 1，则产生 DMA 请求。然后 DMA 会向 SPI_DR 寄存器写入数据。
- 接收时，每次 RXNE 置为 1，则产生 DMA 请求。然后 DMA 读 SPI_DR 寄存器的数据。

当 SPI 被仅用作发送数据，可以只使能 SPI Tx DMA 通道。在这个情况下，因为被接收到的数据没有被读走，OVR 标志位置位。当 SPI 被仅用作接收数据，可以使能 SPI Rx DMA 通道。

在发送时，当 DMA 已经写入了所有要发送的数据（DMA_ISR 寄存器的 TCIF 标志位被置位），可以通过监测 BSY 标志来确保 SPI 通讯已完成。这是用来避免当禁止 SPI 或者进入停止模式时，最后的传输被破坏。软件必须先等待 FTLVL[1:0]=00，然后再等待 BSY=0。

通过 DMA 开始通讯时，为防止 DMA 通道管理引发错误事件发生，必须遵从以下步骤：

1. 使能 DMA Rx 缓冲区（SPI_CR2 的 RXDMAEN 位）（如果 Rx DMA 被使用）
2. 配置 SYSCFG_CFGR3.DMAx_MAP 寄存器，选择 SPI 使用的 DMA 通道；并配置 DMA 的寄存器（参考 DMA 模块的描述）
3. 使能 DMA Tx 缓冲区（在 SPI_CR2 寄存器的 TXDMAEN 位）（如果 Tx DMA 被使用）
4. 配置 SPE=1 使能 SPI

强制用以下步骤关闭通讯：

1. 如果使能了 DMA，通过操作 DMA 寄存器来关闭 DMA 模块（参考 DMA 模块的描述）
2. 通过 SPI 关闭流程关闭 SPI
3. 通过清除 TXDMAEN 和 RXDMAEN（SPI_CR2 寄存器），关闭 DMA Tx 和 Rx 缓冲区（如果 DMA Tx 和 Rx 被使用）

27.3.9.5 时序

本节介绍一些典型的时序，这些时序对于查询、中断或者 DMA 都是有效的。为了简化，假定 LSBFIRST=0，CPOL=0，CPHA=1。也不提供完整的 DMA 操作配置。

1. 激活 NSS 并使能 SPI 后，从机开始控制 MISO；当 NSS 被释放或者 SPI 关闭时从机失去对 MISO 的控制。必须为从机提供充足的时间，以便在传输开始前准备好主机专用的数据。
2. 在主机端，仅在 SPI 使能后，SPI 外设才会控制 MOSI 和 SCK 信号（也包括 NSS 信号）。如果 SPI 被关闭，SPI 外设就从 GPIO 断开，在这些线上的电平值取决于 GPIO 的设定。
3. 在主机端，如果通讯是连续的，则 BSY 在帧之间保持有效。在从机端，BSY 信号在数据帧之间通常会保持至少一个时钟周期的低电平状态。
4. 只有当 TXFIFO 是满的，TXE 信号才被清零。

5. 在 TXDMAEN 位一被置位，DMA 仲裁过程即开始。在 TXEIE 被置位后，产生 TXE 中断。当 TXE 信号有效时，开始向 TxFIFO 传输数据，直到 TxFIFO 变满，或者 DMA 传输完成。
6. 如果所有要被发送的数据装进了 TxFIFO，在 SPI 总线传输前 DMA Tx TCIF 标志就会被拉高。这个标志在 SPI 交互完成之前，一直都为高。
7. 在数据封装模式，TxE 和 RxNE 事件是成对出现的，每个读/写 FIFO 的访问是 16bit 宽，直至数据帧数为偶数。如果 TxFIFO 是 3/4 满，FTLVL 状态停在 FIFO 全满级别。这就是为什么最后一个奇数帧不能在 TxFIFO 变成 1/2 满之前存储。该数据帧以 8-bit 的访问方式（由 DMA 内部信号在 LDMA_TX 置 1 时自动实现）存储在 TxFIFO 中。
8. 为了接收数据封装模式的最后一个奇数数据帧，当最后一个数据帧被处理时，Rx 阈值必须被改变成 8-bit（由 DMA 内部信号在 LDMA_RX 置 1 时自动实现）。

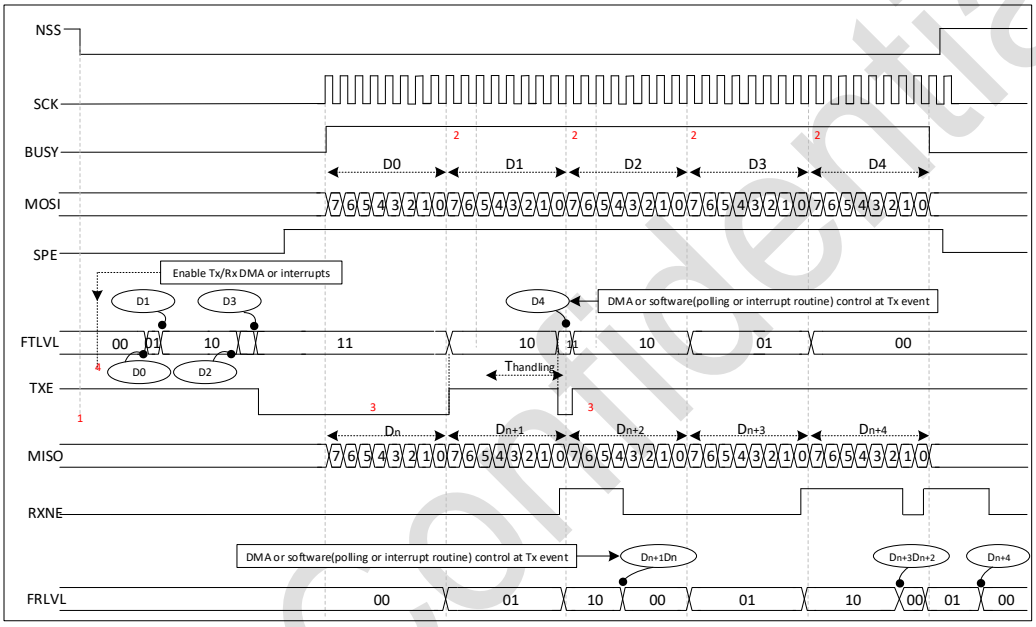


图 27-9 主模式全双工通讯(帧长=8)

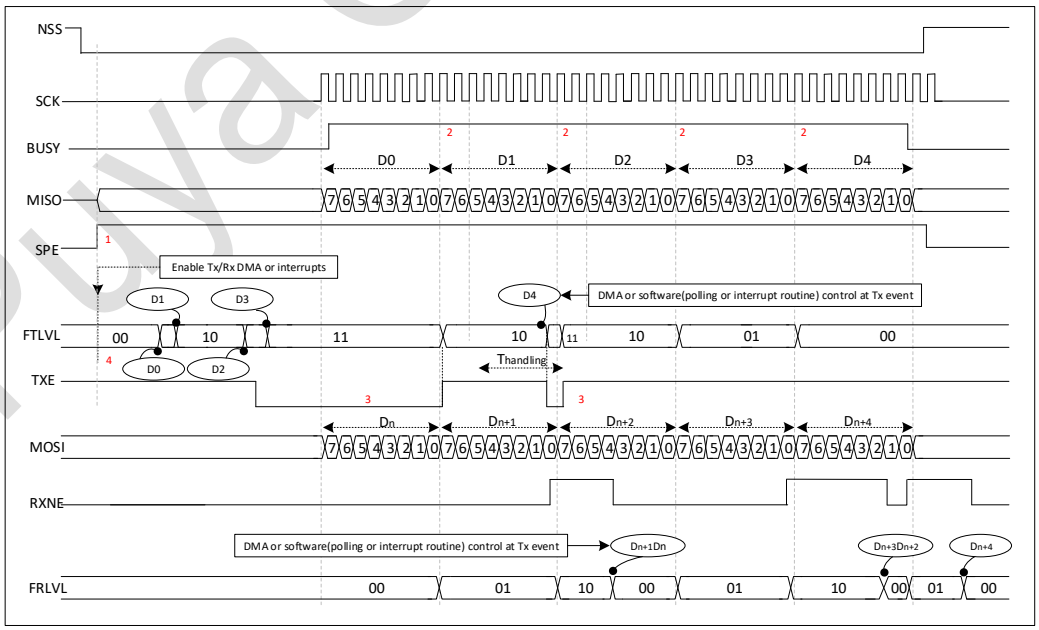


图 27-10 从模式全双工通讯(帧长=8)

27.3.10 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。

27.3.10.1 发送缓存空标志(TXE)

当 TXFIFO 有足够的空间存放要发送的数据时，TXE 标志位置位。TXE 标志位与 TXFIFO 占用级别有关。该标志位变高并保持高电平，直到 TXFIFO 占用级别小于等于 1/2 FIFO 深度。如果 SPI_CR2 寄存器的 TXEIE 置位，则会产生中断请求。当 TXFIFO 占用级别大于 1/2，该位被自动清零。

27.3.10.2 接收缓存非空(RXNE)

当接收到数据，RXNE 置位。

如果 SPI_CR2 寄存器的 RXNEIE 位置位，则产生中断。

当上述条件不再成立，则 RXNE 被硬件自动清零。

27.3.10.3 忙标志(BSY)

BSY 标志由硬件设置与清除(写入此位无作用)，此标志表明 SPI 正在进行数据传输（ SPI 总线繁忙）。当它被设置为'1'时，表明 SPI 正忙于通信，但有一个例外：主模式的双向接收模式下(MSTR=1、IDIMODE=1 并且 BIDIOE=0)，在接收期间 BSY 标志保持为低（接收 CRC 期间除外）。

在软件要关闭 SPI 模块并进入停止低功耗模式(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。

BSY 标志还可以用于在多主机系统中避免写冲突。

除了主模式的双向接收模式(MSTR=1、IDIMODE=1 并且 BIDIOE=0)，当传输开始时，BSY 标志被置'1'。以下情况该标志将被清除为'0'：

- 正确关闭 SPI
- 主机模式，当检测到故障时（MODF=1）
- 主机模式，完成了数据发送并且不准备发送任何新数据时
- 从机模式，在每个数据传输之间，BSY 标志置为 0，并保持至少一个 SPI 时钟周期

注: 不要使用 BSY 标志处理每个数据发送和接收。使用 TXE 和 RXNE 更合适。

27.3.11 错误标志

27.3.11.1 主模式故障（MODF）

主模式故障（MODF）仅发生在：当 NSS 作为输入信号（SSOE=0），NSS 引脚硬件管理模式下，主机的 NSS 脚被拉低；或者在 NSS 引脚软件管理模式下，SSI 位被置为'0'时，MODF 位被自动置位。主模式故障对 SPI 设备有以下影响：

- MODF 位置为'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机的冲突，必须先拉高该主机的 NSS 引脚，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，从机的 MODF 位不能被置为'1'。然而，在多主机配置里，一个设备可以在设置了 MODF 位的情况下，处于从机模式；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。

27.3.11.2 上溢（OVF）

当主机或者从机接收了数据，但 RXFIFO 没有足够的空间存储接收到的数据时，产生上溢情况。如果软件或者 DMA 没有足够的时间读走之前接收到的数据（RXFIFO 中存放），该情况就会发生。

当上溢情况发生，新收到的数据不会覆盖以前存放在 RXFIFO 的数据。接收到的新数据被忽略，并且丢弃所有接下来发送的数据。

依次读出 SPI_DR 寄存器和 SPI_SR 寄存器可将 OVR 清除。

27.3.11.3 CRC 错误（CRCERR）

当 SPI_CR1 寄存器中的 CRCEN 位置 1 时，此标志用于验证接收数据的有效性。如果移位寄存器中接收的值与 SPI_RXCRC 的值不匹配， SPI_SR 寄存器中的 CRCERR 标志将置 1。该标志由软件清零。

27.3.11.4 TI 模式帧格式错误（FRE）

如果 SPI 在从模式下工作，并配置为符合 TI 模式协议，则在通信进行期间出现 NSS 脉冲时，将检测到 TI 模式帧格式错误。出现此错误时，SPI_SR 寄存器中的 FRE 标志将置 1。发生错误时不会关闭 SPI，但会忽略 NSS 脉冲，并且 SPI 会等待下一个 NSS 脉冲，然后再开始新的传输。由于错误检测可能导致丢失两个数据字节，因此数据可能会损坏。

读取 SPI_SR 寄存器时，将清零 FRE 标志。如果 ERRIE 位置 1，则检测到 NSS 错误时将生成中断。在这种情况下，由于无法保证数据的一致性，应关闭 SPI，并在重新使能从 SPI 后，由主器件重新发起通信。

27.3.12 TI 模式

27.3.12.1 主模式下的 TI 协议

SPI 接口与 TI 协议兼容。可以使用 SPI_CR2 寄存器的 FRF 位来配置 SPI，以兼容此协议。

时钟极性和相位都被强制为遵循 TI 协议，和 SPI_CR1 中的设置无关。NSS 管理也特定于 TI 协议，在这种情况下，无法通过 SPI_CR1 和 SPI_CR2 寄存器（SSM、SSI 和 SSOE）来对 NSS 管理进行配置。

在从机模式下，SPI 波特率预分频器用于控制在当前传输完成时 MISO 引脚切换为高阻态的时刻。可以使用任意波特率，因此可以非常灵活地确定此时刻。但是，波特率通常设置为外部主时钟波特率。MISO 信号变为高阻态的延时（trelease）取决于内部重新同步以及通过 SPI_CR1 寄存器的 BR[2:0] 位设置的波特率值。具体公式如下：

$$\frac{t_{baud\_rate}}{2} + 4 \times t_{pclk} < t_{release} < \frac{t_{baud\_rate}}{2} + 6 \times t_{pclk}$$

如果从机在数据帧传输期间检测到错位的 NSS 脉冲，FRE 标志将置 1。

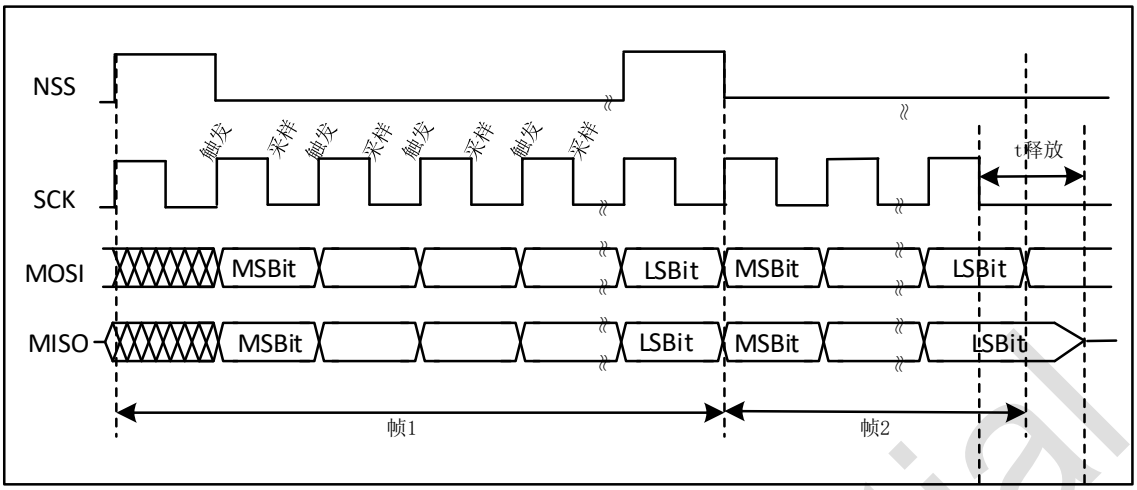


图 27-11 TI 模式传输

27.3.13 SPI CRC

为了检查发送和接收数据的可靠性，利用两个独立的 CRC 计算器。 SPI 提供 CRC8 或 CRC16 计算，且数据帧长度为 8 时选择 CRC8，数据帧长度为 16 时选择 CRC16。

27.3.13.1 CRC 原理

在 SPI 启用（SPE = 1）之前，通过设置 SPI_CR1 寄存器中的 CRCEN 位来启用 CRC 计算。CRC 值是使用每个位上的奇数可编程多项式计算的。该计算在由 SPI_CR1 寄存器中的 CPHA 和 CPOL 位定义的采样时钟边沿上进行处理。计算出的 CRC 值在数据块结束时自动校验，并且由 CPU 或 DMA 管理其传输。当检测到根据接收数据内部计算的 CRC 与发送器发送的 CRC 不匹配时，设置 CRCERR 标志以指示数据错误。处理 CRC 计算的正确程序取决于 SPI 配置和选择的传输管理方式。

27.3.13.2 CPU 管理 CRC 传输

通信开始后将一直持续到必须发送或接收 SPIx_DR 寄存器中的最后一个数据帧。之后，SPIx_CR1 寄存器中的 CRCNEXT 位必须置 1，以指示当前处理的数据帧传输后将处理 CRC 帧传输。 CRCNEXT 位必须在最后一个数据帧传输结束前置 1。在 CRC 传输期间，CRC 计算将冻结。

所接收的 CRC 以数据字节或数据字的形式存储在 RXFIFO 中。因此仅在 CRC 模式下，接收缓冲区才必须被视为一个 16 位缓冲区且一次仅接收一个数据帧。

CRC 格式的传输通常在数据传输结束时需要多出一个数据帧。但是，当设置一个通过 16 位 CRC 校验的 8 位数据帧时，需要多两帧才能发送完整的 CRC 值。

接收最后一个 CRC 数据后，将执行自动校验，将接收的值与 SPI_RXCRC 寄存器中的值进行比较。软件必须校验 SPI_SR 寄存器中的 CRCERR 标志，以确定数据传输是否损坏。软件通过向 CRCERR 标志写入“0”来将其清零。

接收 CRC 后，CRC 值存储到 RXFIFO 中，且必须在 SPI_DR 寄存器中进行读取，以将 RXNE 标志清零。

27.3.13.3 DMA 管理 CRC 通信

当使能 SPI，并开启带 CRC 的 DMA 模式进行通信时，在通信结束时自动发送和接收 CRC（在仅接收模式下读取 CRC 数据除外），CRCNEXT 位并非一定要通过软件来处理。 SPI 发送 DMA 通道计数器必须设置为要发送的数据帧数，其中不包括 CRC 帧。在接收器侧，接收的 CRC 值在传输结束时通过 DMA 自动处理，但用户必须注意刷新 RXFIFO 中接收的 CRC 信息(因为该信息始终加载到其中)。在全双工模式下，接收 DMA 通道计数器可设置为要接收的数据帧数，其中包括 CRC。

在只接收模式下，DMA 接收通道计数器应仅包含已传输的数据量，而不包括 CRC 计算。之后，基于通过 DMA 实现的完整传输，必须通过软件从 FIFO 中读回所有 CRC 值，因为 FIFO 在该模式下用作单个缓冲区。

如果传输过程中出现故障，则在数据和 CRC 传输结束时，将 SPI_SR 寄存器中的 CRCERR 标志位置 1。

27.3.13.4 CRC 使用流程

使能 CRC 的流程如下：

- 配置 CPOL、CPHA、LSBFIRST、BR、SSM、SSI 和 MSTR 寄存器
- 在 SPI_CRCPR 寄存器输入多项式
- 通过设置 SPI_CR1 寄存器 CRCEN 位使能 CRC 计算，该操作也会清除寄存器 SPI_RXCRCR 和 SPI_TXCRC
- 设置 SPI_CR1 寄存器的 SPE 位启动 SPI 功能
- 启动通信并且维持通信，直到只剩最后一个字节或者半字
- 在把最后一个字节或半字写进发送缓冲器时，设置 SPI_CR1 的 CRCNEXT 位，指示硬件在发送完成最后一个数据之后，发送 CRC 的数值。在发送 CRC 数值期间，停止 CRC 计算
- 当最后一个字节或半字被发送后，SPI 发送 CRC 数值，CRCNEXT 位被清除。同样，接收到的 CRC 与 SPI_RXCRCR 值进行比较，如果比较不匹配，则设置 SPI_SR 寄存器的 CRCERR 标志位，当设置了 SPI_CR2 寄存器的 ERRIE 时，则产生中断。

注意：在使能 CRC 后，接收或者发送数据会一直做 CRC 计算。所以，如果数据不再需要 CRC 计算，要及时关闭 CRC。

如果在通信期间关闭了 SPI，再使能 SPI 及 CRC，则必须遵循以下顺序：

关闭 SPI

- 将 CRCEN 清零
- 使能 CECEN
- 使能 SPI

27.3.14 SPI 中断

表 27-1 SPI 中断请求

中断事件	事件标志	使能控制位
TXFIFO 等待被装载	TXE	TXEIE
数据接收到 RXFIFO 中	RXNE	RXNEIE
主模式故障事件	MODF	ERRIE
上溢错误	OVR	
TI 帧格式错误	FRE	
CRC 校验错误	CRCERR	

27.4 I²S 功能描述

27.4.1 I²S 框图

I²S 模块框图如下：

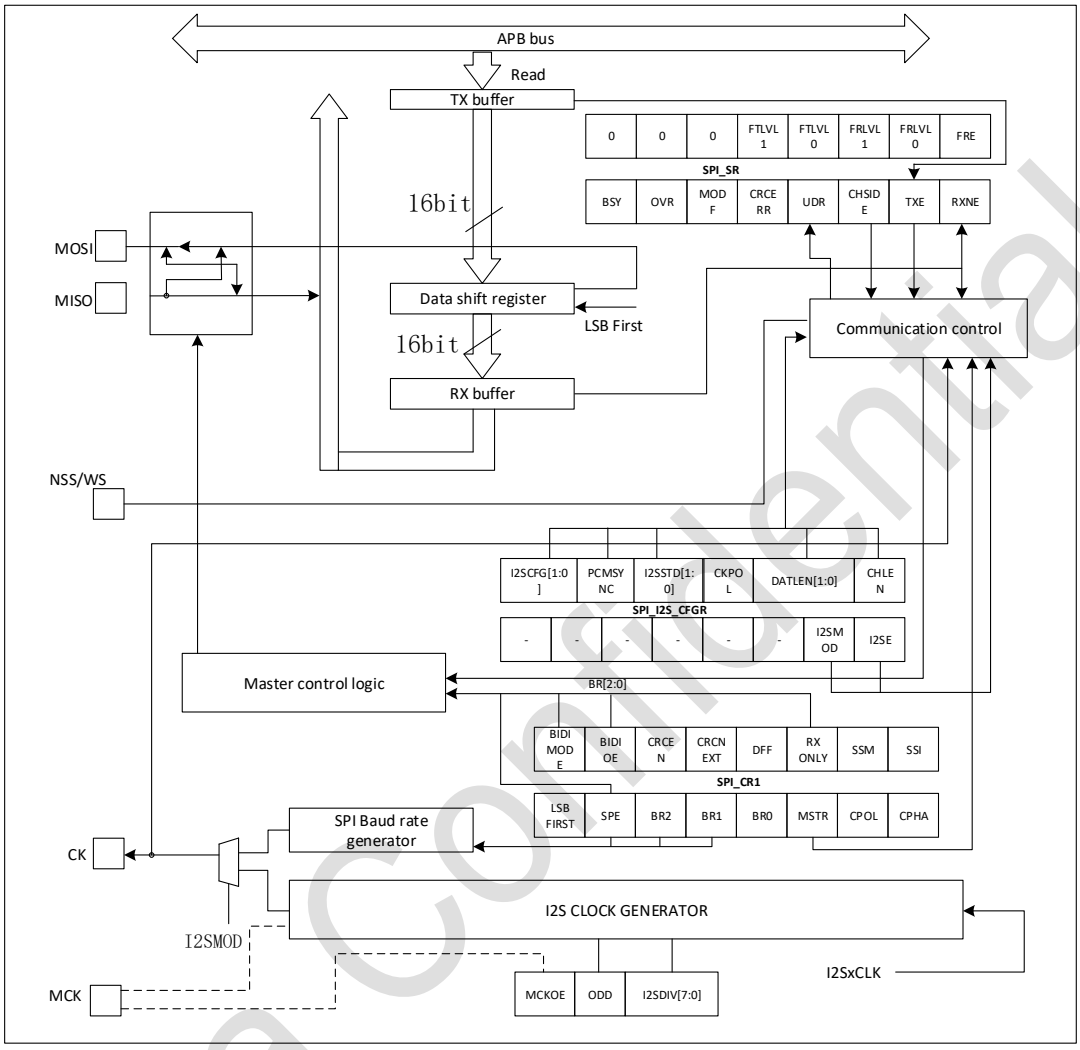


图 27-12 I²S 框图

通过将寄存器 SPI_I2SCFGR 的 I2SMOD 位置为‘1’，即可使能 I²S 功能。此时，可以把 SPI 模块用作 I²S 音频接口。I²S 接口与 SPI 接口使用大致相同的引脚、标志和中断。

I²S 与 SPI 共用 3 个引脚：

SD：串行数据(映射至 MOSI 引脚)，用来发送和接收 2 路时分复用通道的数据（仅半双工模式）；

WS：字选(映射至 NSS 引脚)，主模式下作为数据控制信号输出，从模式下作为数据控制信号输入；

CK：串行时钟(映射至 SCK 引脚)，主模式下作为时钟信号输出，从模式下作为时钟信号输入。

在某些外部音频设备需要主时钟时，可以另有一个附加引脚输出时钟：

MCK：主时钟(映射至 MISO 引脚)，在 I2S 配置为主模式，寄存器 SPI_I2SPR 的 MCKOE 位为‘1’时，作为输出额外的时钟信号引脚使用。输出时钟信号的频率预先设置为 256 × Fs，其中 Fs 是音频信号的采样频率。

设置成主模式时，I²S 使用自身的时钟发生器来产生通信用的时钟信号。这个时钟发生器也是主时钟输出的时钟源。I²S 模式下有 2 个额外的寄存器，一个是与时钟发生器配置相关的寄存器 SPI_I2SPR，另

一个是 I²S 通用配置寄存器 SPI_I2SCFGR(可设置音频标准、从/主模式、数据格式、数据包帧、时钟极性等参数)。

在 I²S 模式下不使用寄存器 SPI_CR1 和所有的 CRC 寄存器。同样，I²S 模式下也不使用寄存器 SPI_CR2 的 SSOE 位，和寄存器 SPI_SR 的 MODF 位和 CRCERR 位。

I²S 使用与 SPI 相同的寄存器 SPI_DR 用作 16 位宽模式数据传输。

27.4.2 支持音频协议

三线总线支持 2 个声道上音频数据的时分复用：左声道和右声道，但是只有一个 16 位寄存器用作发送或接收。因此，软件在对数据寄存器写入数据时，必须根据当前传输中的声道写入；同样，在读取寄存器数据时，必须通过检查寄存器 SPI_SR 的 CHSIDE 位来判明接收到的数据属于哪个声道。左声道总是先于右声道发送数据(CHSIDE 位在 PCM 协议下无意义)。

有四种可用的数据和包帧组合。可以通过以下四种数据格式发送数据：

- 16 位数据打包进 16 位帧
- 16 位数据打包进 32 位帧
- 24 位数据打包进 32 位帧
- 32 位数据打包进 32 位帧

在使用 16 位数据扩展到 32 位帧时，前 16 位（MSB）是有意义的数据，后 16 位（LSB)被强制为 0，该操作不需要软件干预，也不需要有 DMA 请求（仅需要一次读/写操作)。

24 位和 32 位数据帧需要 CPU 对寄存器 SPI_DR 进行 2 次读或写操作，在使用 DMA 时，需要 2 次 DMA 传输。对于 24 位数据，扩展到 32 位后，最低 8 位由硬件置 0。

对于所有的数据格式和通讯标准，总是先发送最高位（MSB)。

I²S 接口支持四种音频标准，可以通过设置寄存器 SPI_I2SCFGR 的 I2SSTD[1:0]位和 PCMSYNC 位来选择。

27.4.2.1 I²S 飞利浦标准

在此标准下，引脚 WS 用来指示正在发送的数据属于哪个声道。在发送第一位数据（MSB）前该引脚有效 1 个时钟周期。

16/32 位全精度时序图如下图所示，发送方在时钟信号（CK）的下降沿改变数据，接收方在上升沿读取数据。WS 信号也在 CK 的下降沿变化。

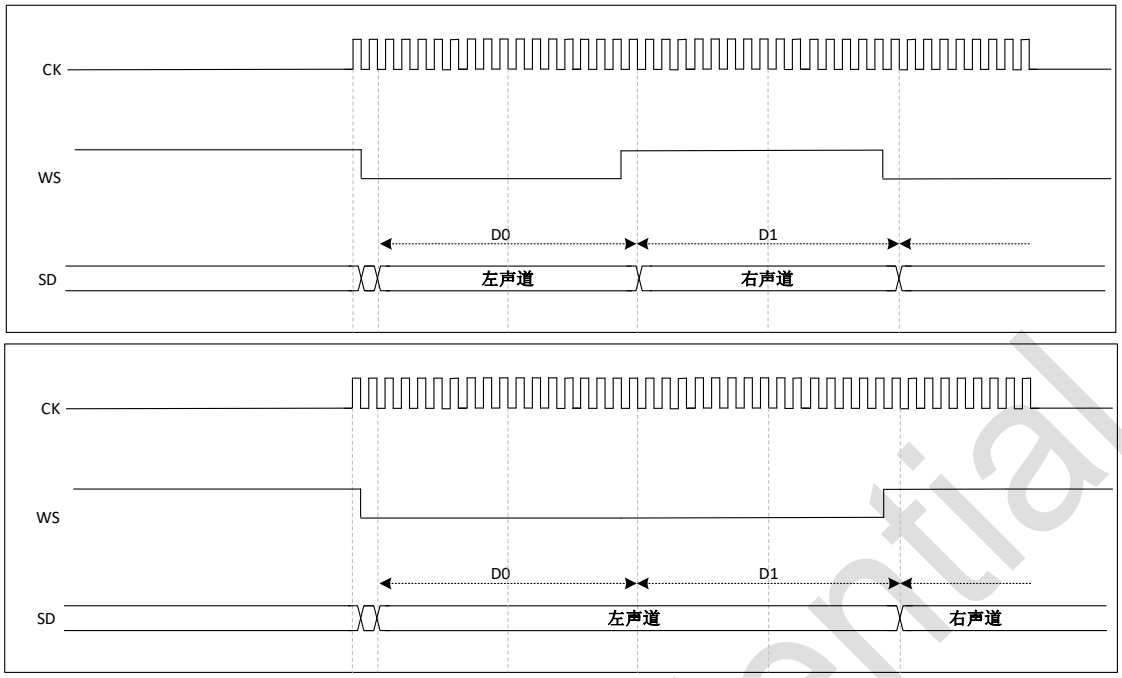


图 27-13 I²S 飞利浦协议波形 (16/32 位全精度, CKPOL=0)

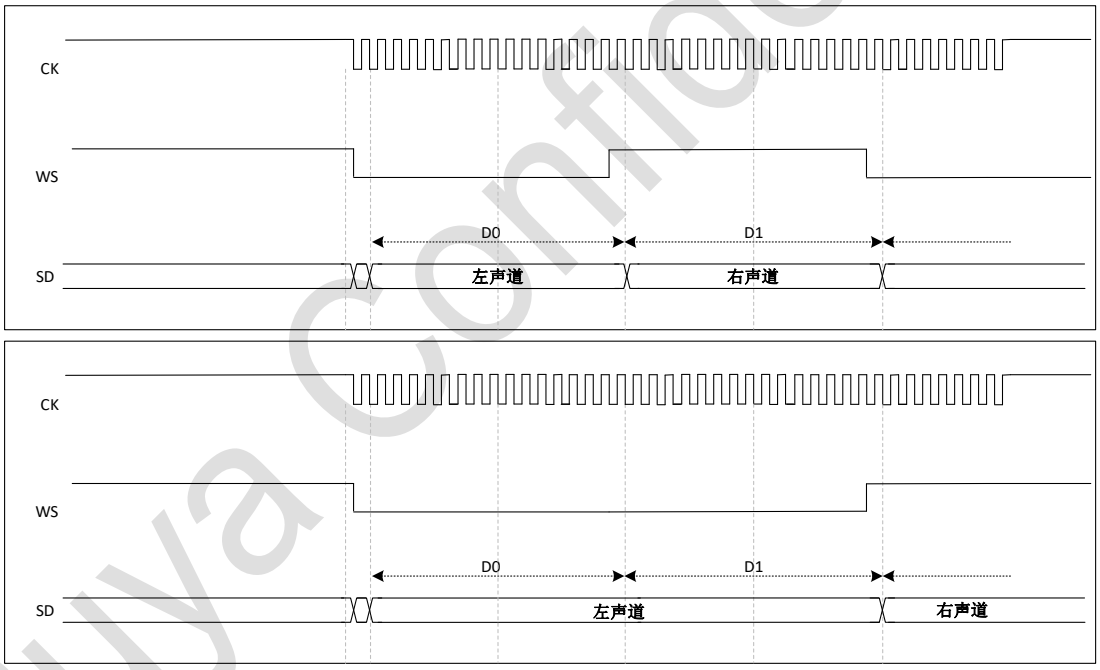


图 27-14 I²S 飞利浦协议波形 (16/32 位全精度, CKPOL=1)

24 位帧时序图如下:

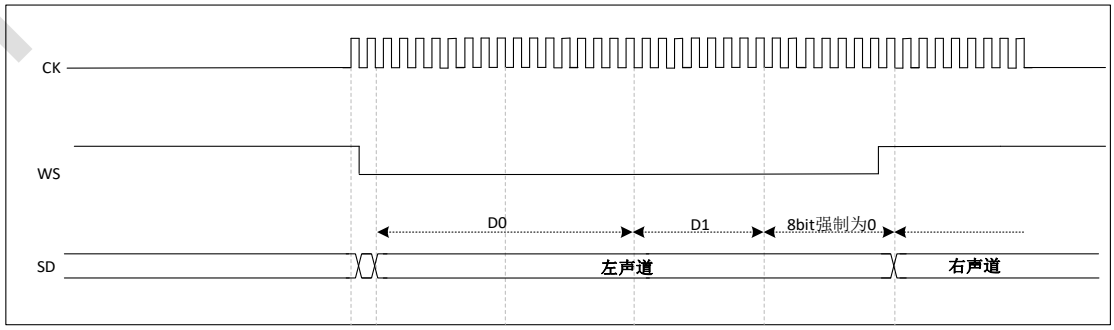


图 27-15 I²S 飞利浦协议波形(24 位帧, CKPOL=0)

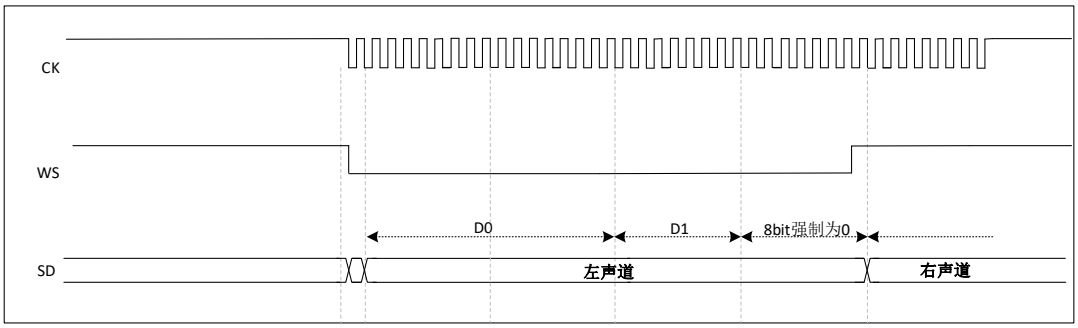


图 27-16 I²S 飞利浦协议波形(24 位帧, CKPOL=1)

此模式需要对寄存器 SPI_DR 进行 2 次读或写操作。

- 在发送模式下：如果需要发送 0x8EAA33(24 位)，第一次写入 0x8EAA，第二次写入 0x33xx，低 8 位无意义
- 在接收模式下：如果接收 0x8EAA33(24 位)，第一次接收 0x8EAA，第二次读出 0x3300，只高 8 位为有意义的数据，低 8 位始终为 0

下图为 16 位扩展到 32 位声道帧的时序：

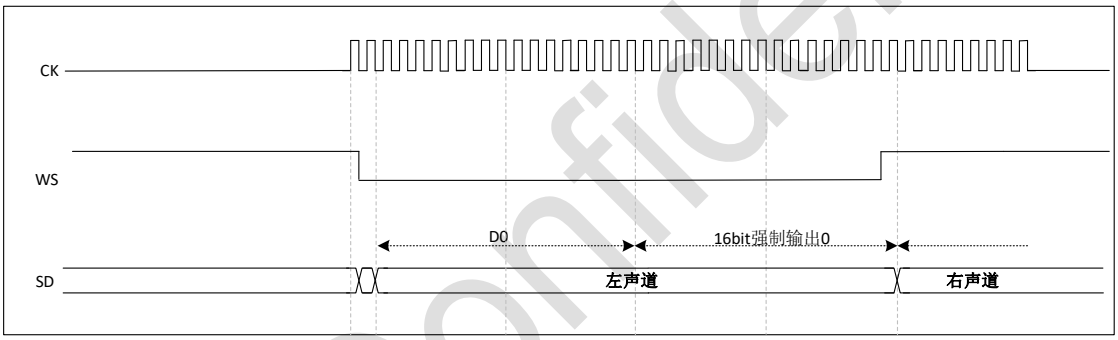


图 27-17 I²S 飞利浦协议波形(16 位扩展到 32 位)

在 I²S 配置阶段，如果选择将 16 位数据扩展到 32 位声道帧，只需要访问一次寄存器 SPI_DR。用来扩展到 32 位的低 16 位被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x76A30000)，只需要操作一次 SPI_DR，写入数据 0x76A3.

发送时，需要将 MSB 写入寄存器 SPI_DR：标志位 TXE 为‘1’表示可以写入新的数据，如果允许了相应的中断，则可以产生中断。发送是由硬件完成的，即使还未发送出后 16 位的 0x0000，也会设置 TXE 并产生相应的中断。

接收时，每次收到高 16 位半字(MSB)后，标志位 RXNE 置‘1’，如果允许了相应的中断，则可以产生中断。

这样，就延长了两个写入或读取操作之间的时间间隔，从而可防止出现下溢或上溢情况（具体取决于数据传输方向）。

27.4.2.2 MSB 对齐标准

在此标准下，WS 信号和第一个数据位，即最高位(MSB)同时产生。

16/32 位全精度时序图如下图所示,发送方在时钟信号的下降沿改变数据;接收方是在上升沿读取数据。

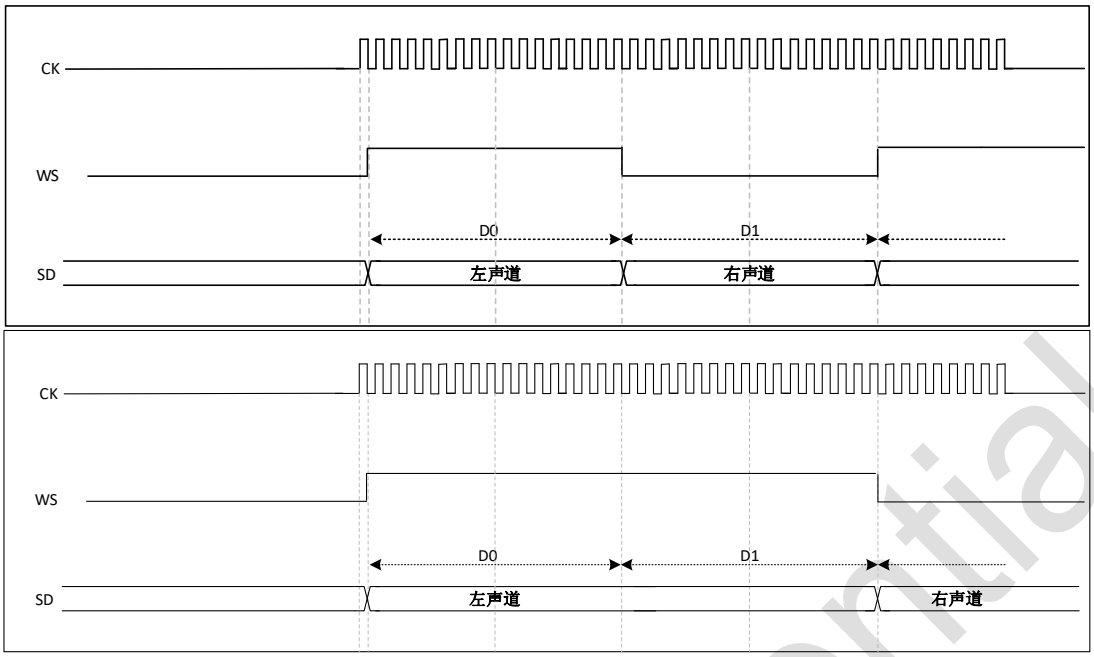


图 27-18 I²S MSB 对齐标准 (16/32 全精度, CKPOL=0)

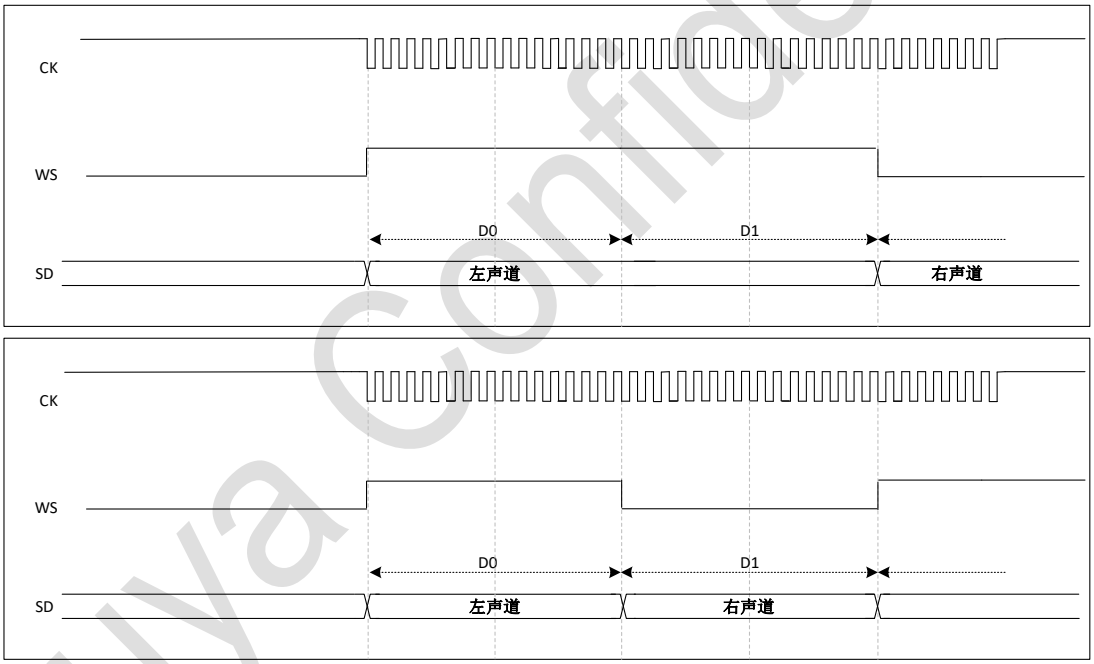


图 27-19 I²S MSB 对齐标准 (16/32 全精度, CKPOL=1)

24 位帧时序图如下:

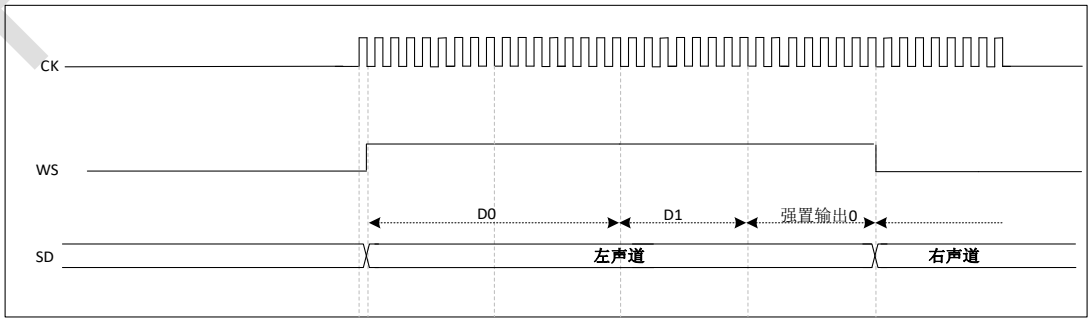


图 27-20 I²S MSB 对齐标准(24 位帧，CKPOL=0)

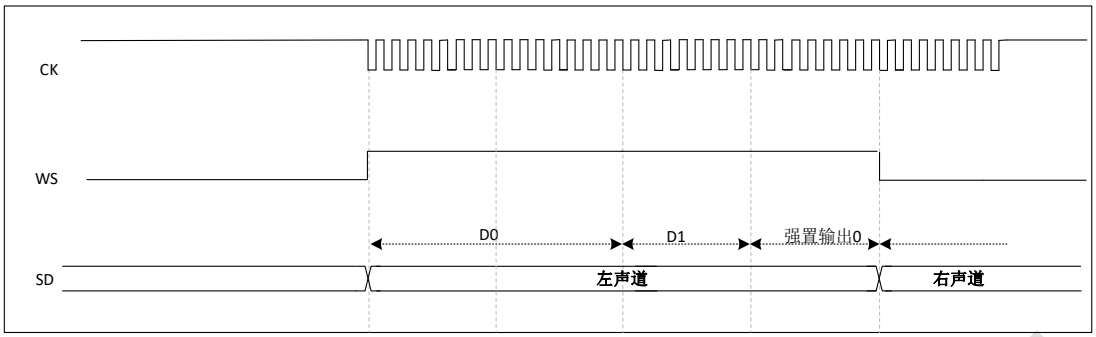


图 27-21 I²S MSB 对齐标准(24 位帧, CKPOL=1)

下图为 16 位扩展到 32 位声道帧的时序:

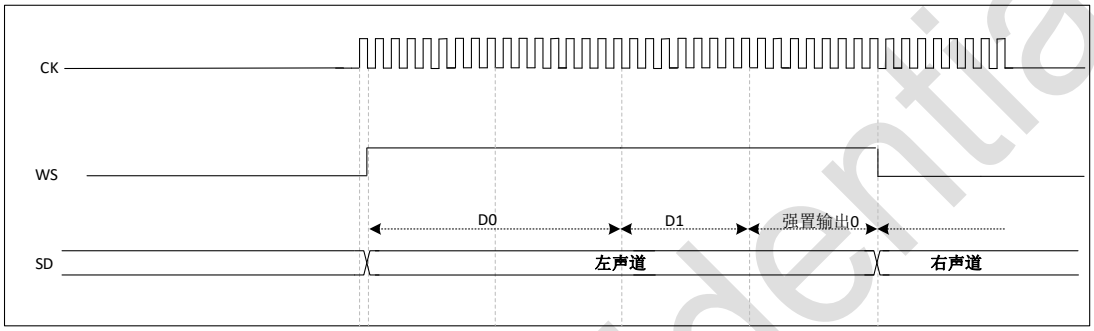


图 27-22 I²S MSB 对齐标准(16 位扩展到 32 位，CKPOL=0)

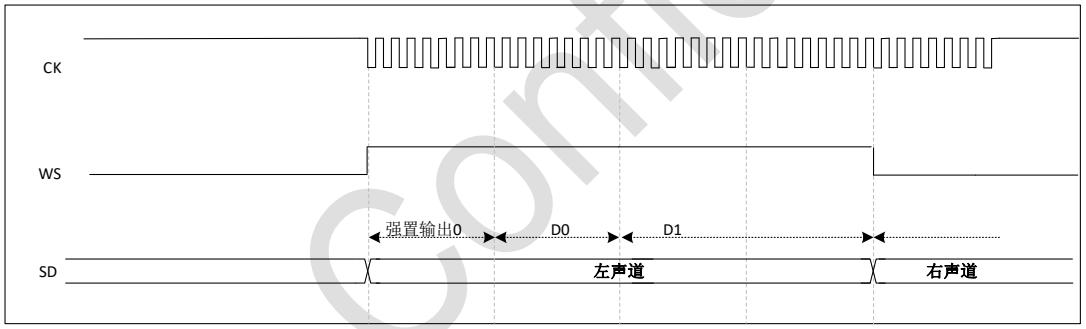


图 27-23 I²S MSB 对齐标准(16 位扩展到 32 位，CKPOL=1)

一旦有效数据开始从 SD 引脚送出，就发生下一次 TXE 事件。在接收时，一旦接收到有效数据(而不是 0x0000 部分)，就发生 RXNE 事件。

27.4.2.3 LSB 对齐标准

此标准与 MSB 对齐标准类似(在 16 位或 32 位全精度帧格式下无区别)。

16/32 位全精度时序图如下图所示,发送方在时钟信号的下降沿改变数据;接收方是在上升沿读取数据。

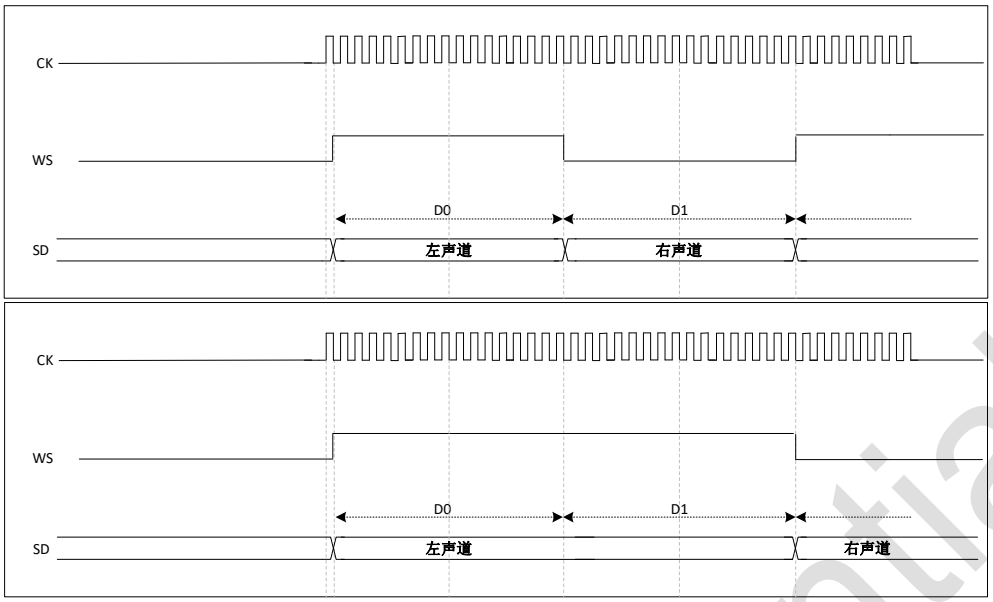


图 27-24 I²S LSB 对齐标准 (16/32 位全精度, CKPOL=0)

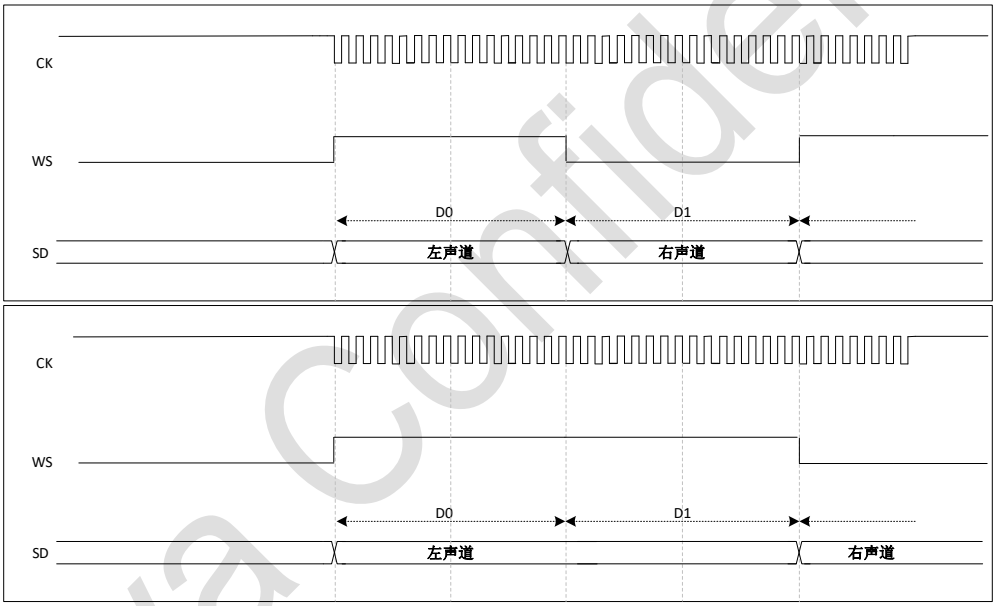


图 27-25 I²S LSB 对齐标准 (16/32 位全精度, CKPOL=1)

24 位帧时序图如下:

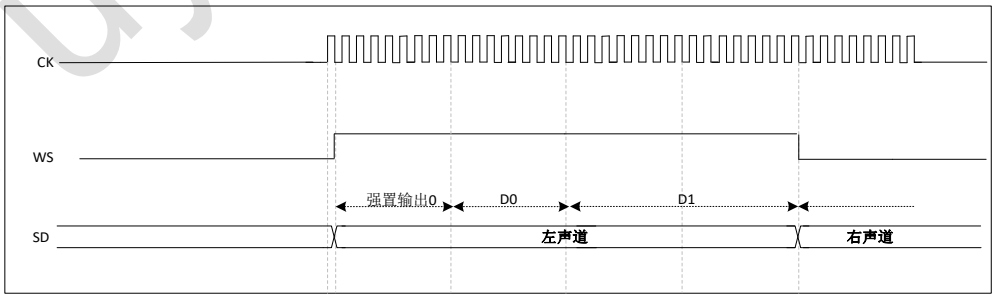


图 27-26 I²S LSB 对齐标准（24 位帧, CKPOL=0）

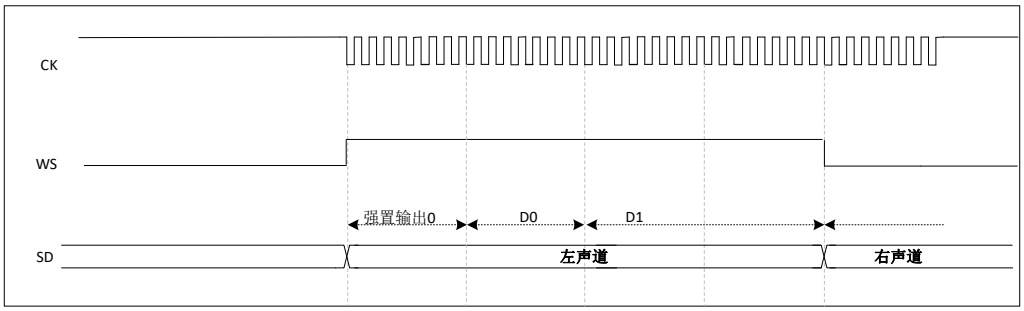


图 27-27 I²S LSB 对齐标准（24 位帧, CKPOL=1）

■ 在发送模式下

如果要发送数据 0x3478AE，需要通过软件或者 DMA 对寄存器 SPI_DR 进行 2 次写操作。第一次写入数据寄存器 0xXX34，第二次写入数据寄存器 0x78AE。

■ 在接收模式下

如果要接收数据 0x3478AE，需要在 2 个连续的 RXNE 事件发生时，分别对寄存器 SPI_DR 进行 1 次读操作。第一次读出 0x0034，只有低 8 位有意义；第二次读出 0x78AE。

下图为 16 位扩展到 32 位声道帧的时序：

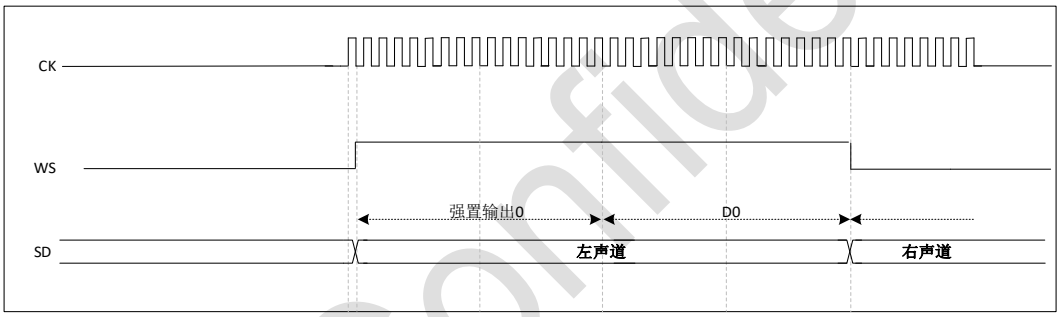


图 27-28 I²S LSB 对齐标准（16 位扩展到 32 位，CKPOL=0）

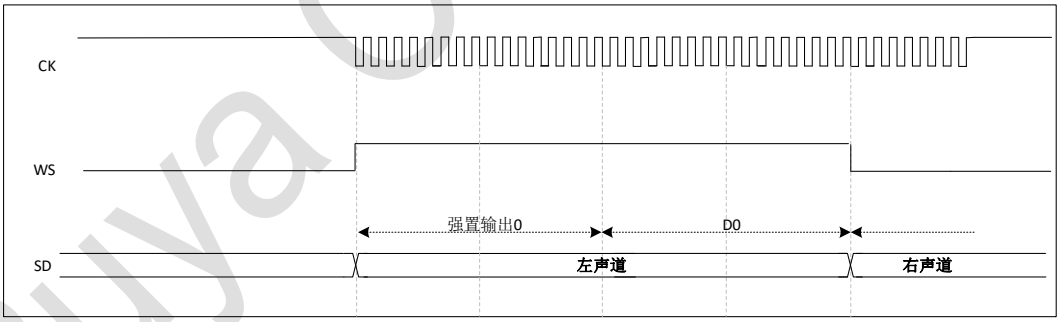


图 27-29 I²S LSB 对齐标准（16 位扩展到 32 位，CKPOL=1）

在 I²S 配置阶段，如果选择将 16 位数据扩展到 32 声道帧，只需要访问一次寄存器 SPI_DR。此时，扩展到 32 位后的高半字（16 位 MSB）被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3（扩展到 32 位是 0x0000 76A3），只需要操作一次 SPI_DR 寄存器，写入 0x76A3.

发送时，如果 TXE 为‘1’，用户需要写入待发送的数据（即 0x76A3）。用来扩展到 32 位的 0x0000，部分由硬件首先发送出去，一旦有效数据开始从 SD 引脚送出，就发生下一次 TXE 事件。

接收时，一旦接收到有效数据（而不是 0x0000 部分），就发生 RXNE 事件。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

27.4.2.4 PCM 标准

在 PCM 标准下，不存在声道选择的信息。PCM 标准有 2 种可用的帧结构，短帧或者长帧，可以通过设置寄存器 SPI_I2SCFGR 的 PCMSYNC 位来选择。

在 PCM 模式下，输出信号（WS 和 SD）在 CK 信号的上升沿采样。输入信号（WS 和 SD）在 CK 的下降沿捕获。

请注意，CK 和 WS 在主模式下配置为输出。

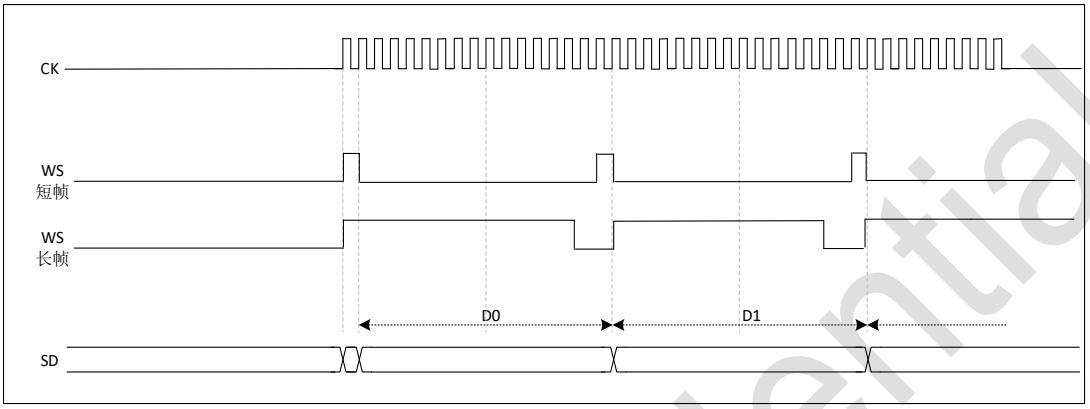


图 27-30 I²S PCM 标准(16 位帧，CKPOL=0)

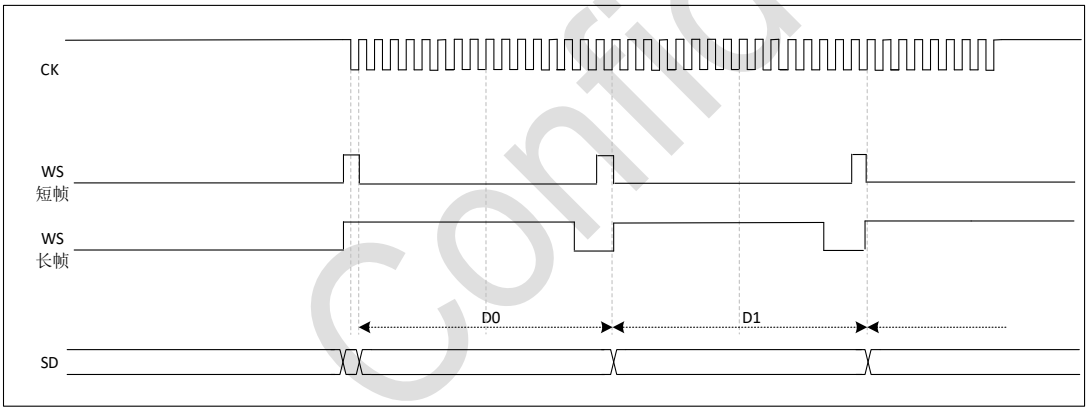


图 27-31 I²S PCM 标准 (16 位帧，CKPOL=1)

对于长帧，主模式下，用来同步的 WS 信号有效的时间固定为 13 位。

对于短帧，用来同步的 WS 信号长度只有 1 位。

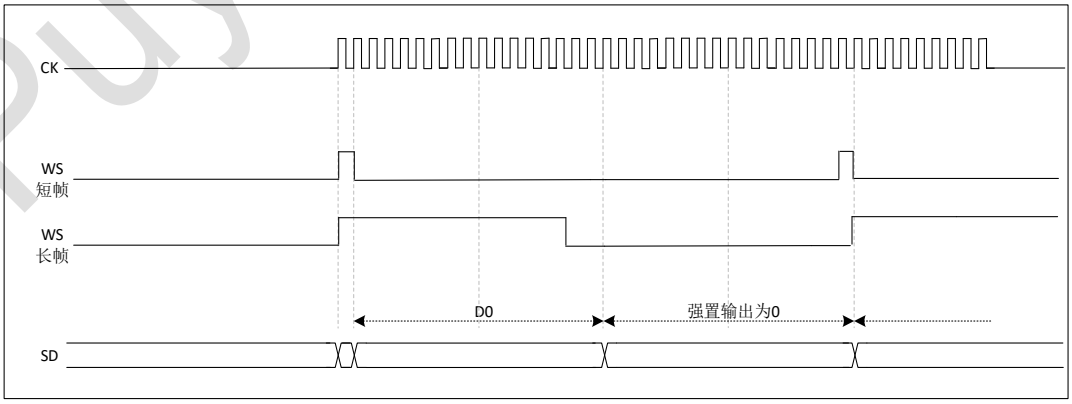


图 27-32I²S PCM 标准(16 位扩展到 32 位, CKPOL=0)

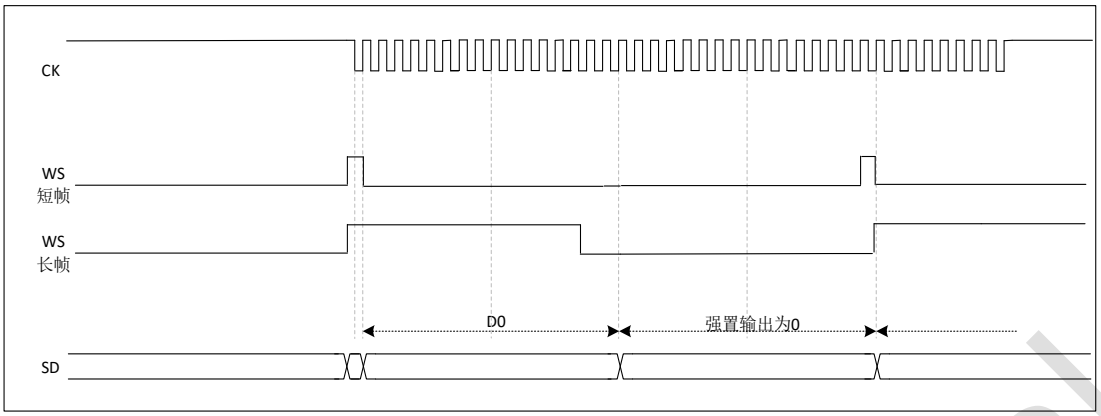


图 27-33 I²S PCM 标准（16 位扩展到 32 位，CKPOL=1）

无论哪种模式(主或从)、哪种同步方式(短帧或长帧)，即使是从模式，需要通过设置 SPI_I2SCFGR 寄存器的 DATLEN 位和 CHLEN 位来指定连续的 2 帧数据以及 2 个同步信号之间的时间差。

27.4.3 时钟产生

I²S 的比特率即确定了在 I²S 数据线上的数据流和 I²S 的时钟信号频率。即

I²S 比特率 = 每个声道的比特数 x 声道数目 x 音频采样频率.

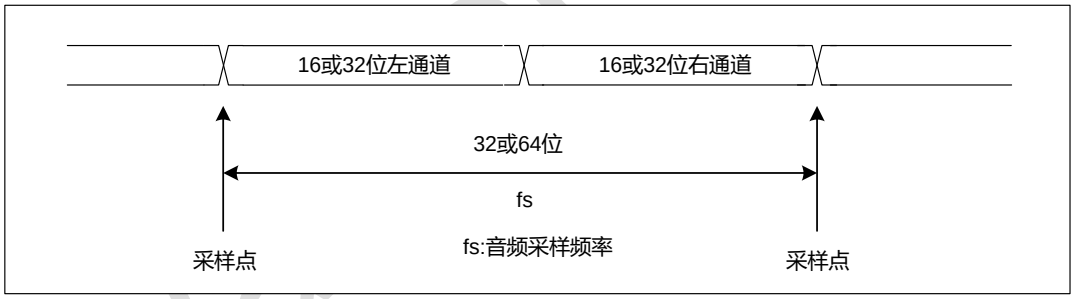
对于一个具有左右声道和 16 位音频信号，I²S 比特率计算如下：

I²S 比特率 = 16×2×fs

如果包长为 32 位，则有：

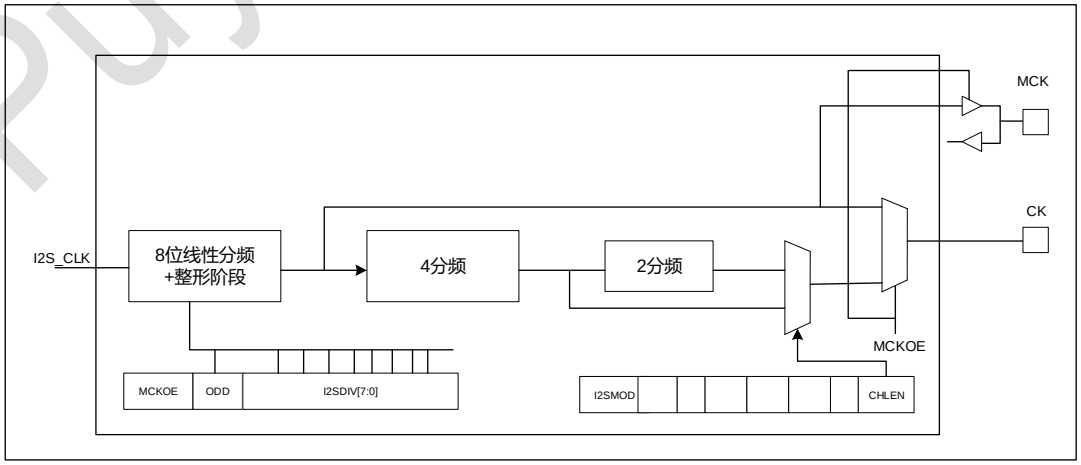
I²S 比特率=32×2×fs

音频采样定义频率定义如下图所示：



在主模式下，为了获得需要的音频频率，需要正确地对线性分频器进行设置。

I²S 时钟发生器结构如下图所示：



I2S_CLK 由 RCC 模块产生，且与 PCLK 同步。

音频的采样频率可以是 192 kHz、96 kHz、48 kHz、44.1 kHz、32 kHz、22.05 kHz、16 kHz、11.025 kHz 或者 8 kHz(或任何此范围内的数值)。为了获得需要的频率，需按照以下公式设置线性分频器而获得：

27.4.3.1 I²S 模式时钟产生

输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为‘1’)：

$fs=f_{I2S\_CLK}/[256x((2xI2SDIV)+ODD)]$

禁止输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为‘0’)：

声道的帧长为 16 位时，$fs=f_{I2S\_CLK}/[32x((2xI2SDIV)+ODD)]$

声道的帧长为 32 位时，$fs=f_{I2S\_CLK}/[64x((2xI2SDIV)+ODD)]$

27.4.3.2 PCM 模式时钟产生

输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为‘1’)：

$fs=f_{I2S\_CLK}/[128x((2xI2SDIV)+ODD)]$

禁止输出主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为‘0’)：

声道的帧长为 16 位时，$fs=f_{I2S\_CLK}/[16x((2xI2SDIV)+ODD)]$

声道的帧长为 32 位时，$fs=f_{I2S\_CLK}/[32x((2xI2SDIV)+ODD)]$

使用标准的 8 MHz HSE 时钟得到精确的音频频率，见下表。

表 27-2 使用 8 MHz HSE 的音频频率精度

SYSCLK (MHz)	I2SDIV		ODD		MCLK	期望 fs(Hz)	实际 fs(Hz)		误差	
	16 位	32 位	16 位	32 位			16 位	32 位	16 位	32 位
72	11	6	1	0	无	96000	97826.09	93750	1.90%	2.34%
72	23	11	1	1	无	48000	47872.34	48913.04	0.27%	1.90%
72	25	13	1	0	无	44100	44117.65	43269.23	0.04%	1.88%
72	35	17	0	1	无	32000	32142.86	32142.86	0.45%	0.45%
72	51	25	0	1	无	22050	22058.82	22058.82	0.04%	0.04%
72	70	35	1	0	无	16000	15957.45	16071.43	0.27%	0.45%
72	102	51	0	0	无	11025	11029.41	11029.41	0.04%	0.04%
72	140	70	1	1	无	8000	8007.117	7978.723	0.09%	0.27%
72	2	2	0	0	有	96000	70312.5	70312.5	26.76%	26.76%
72	3	3	0	0	有	48000	46875	46875	2.34%	2.34%
72	3	3	0	0	有	44100	46875	46875	6.29%	6.29%
72	4	4	1	1	有	32000	31250	31250	2.34%	2.34%
72	6	6	1	1	有	22050	21634.62	21634.62	1.88%	1.88%
72	9	9	0	0	有	16000	15625	15625	2.34%	2.34%
72	13	13	0	0	有	11025	10817.31	10817.31	1.88%	1.88%
72	17	17	1	1	有	8000	8035.714	8035.714	0.45%	0.45%

27.4.4 I²S 传输

27.4.4.1 主模式

I²S 可配置为主模式。这意味着将在 CK 引脚输出串行时钟，在 WS 引脚生成字选信号。主时钟（MCK)可以输出，也可以不输出，具体由 SPI_I2SPR 寄存器中的 MCKOE 位控制。

配置流程如下：

1. 设置寄存器 SPI_I2SPR 的 I2SDIV[7:0]定义与音频采样频率相符的串行时钟波特率。同时也要定义寄存器 SPI_I2SPR 的 ODD 位。
2. 设置 CKPOL 位定义通信用时钟在空闲时的电平状态。如果需要向外部的 DAC/ADC 音频器件提供主时钟 MCK，则将寄存器 SPI_I2SPR 的 MCKOE 位置为‘1’，并按照不同的 MCK 输出状态，计算 I2SDIV 和 ODD 的值。
3. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位为‘1’激活 I2S 功能，设置 I2SSTD[1:0]和 PCMSYNC 位选择所用的 I²S 标准，设置 CHLEN 选择每个声道的数据位数。还要设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I²S 主模式和方向（发送器或者接收器）。
4. 如果需要，可以通过设置寄存器 SPI_CR2 来打开所需的中断功能和 DMA 功能。
5. 必须将寄存器 SPI_I2SCFGR 的 I2SE 位置为‘1’。
6. 引脚 WS 和 CK 需要配置为输出模式。如果寄存器 SPI_I2SPR 的 MCKOE 位为‘1’，引脚 MCK 也要配置成输出模式。

■ 发送流程

当将 1 个半字(16 位)的数据写入发送缓存区后，发送流程开始。

假设第一个写入发送缓存的数据对应的是左声道数据。当数据从发送缓存区移到移位寄存器时，标志位 TXE 置‘1’，这时，要把对应右声道的数据写入发送缓存区。标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。TXE 标志置 1 时，CHSIDE 标志有意义，因为该标志在 TXE 变为高电平时进行更新。在先左声道后右声道的数据都传输完成后，才能被认为是一个完整的数据帧。不可以只传输部分数据帧，如仅有左声道的数据。

当发出第一位数据的同时，半字数据被并行地传送至 16 位移位寄存器，然后后面的位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓存移至移位寄存器时，标志位 TXE 置为‘1’，如果寄存器 SPI_CR2 的 TXEIE 位为‘1’，则产生中断。

写入数据的操作取决于所选择的 I2S 标准。为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。

要通过将 I2SE 清零来关闭 I2S，必须等待 TXE = 1 且 BSY = 0。

■ 接收流程

接收流程与发送流程类似，区别在第 3 步中选择接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收。即每次填满接收缓存后，标志位 RXNE 置‘1’，如果寄存器 SPI_CR2 的 RXNEIE 位为‘1’，则产生中断。根据配置的数据和声道长度，收到左声道或右声道的数据会需要 1 次或者 2 次把数据传送到接收缓存的过程。

对寄存器 SPI_DR 进行读操作即可清除 RXNE 标志位。

每次接收以后即更新 CHSIDE。它的值取决于 I2S 单元产生的 WS 信号。读取数据的操作取决于所选择的 I²S 标准。

如果前一个接收到的数据还没有被读取，又接收到新数据，即发生上溢，标志位 OVR 被置为‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，表示发生了错误。

■ **关闭 I²S**

若要关闭 I²S 功能，需要执行特别的操作，以保证 I²S 模块可以正常地完成传输周期而不会开始新的数据传输。

具体操作流程：

若要关闭 I²S 功能，需要执行特别的操作，以保证 I²S 模块可以正常地完成传输周期而不会开始新的数据传输。操作过程与数据配置和通道长度、以及音频协议的模式相关：

- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 LSB(低位)对齐模式(I2SSTD=10)
  - 等待倒数第二个(n-1)RXNE=1
  - 等待 17 个 I²S 时钟周期(使用软件延迟),
  - 关闭 I²S(I2SE=0)
- 16 位数据扩展到 32 位通道长度(DATLEN=00 并且 CHLEN=1)，使用 MSB(高位)对齐，I²S 或 PCM 模式(分别为 I2SSTD=00，I2SSTD=01 或 I2SSTD=11)
  - 等待最后一个 RXNE=1
  - 等待 1 个 I²S 时钟周期(使用软件延迟)
  - 关闭 I2S(I2SE=0)
- 所有其它 DATLEN 和 CHLEN 的组合，无论通过 I2SSTD 选择何种音频模式，使用下述方式关闭 I²S：
  - 等待倒数第二个(n-1)RXNE=1
  - 等待一个 I2S 时钟周期(使用软件延迟)
  - 关闭 I2S(I2SE=0)

27.4.4.2 从模式

在从模式下，I²S 可以设置成发送和接收模式。从模式的配置方式基本遵循和配置主模式一样的流程。在从模式下，不需要 I²S 接口提供时钟。时钟信号和 WS 信号都由外部主 I2S 设备提供，连接到相应的引脚上。因此用户无需配置时钟。

配置步骤如下：

1. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位激活 I2S 功能；设置 I2SSTD[1:0]来选择所用的 I2S 标准；设置 DATLEN[1:0]选择数据的比特数；设置 CHLEN 选择每个声道的数据位数。设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0]选择 I2S 从模式的数据方向。
2. 根据需要，设置寄存器 SPI_CR2 打开所需的中断功能和 DMA 功能。
3. 必须设置寄存器 SPI_I2SCFGR 的 I2SE 位为‘1’。

■ 发送流程

当外部主机发送时钟信号，并且当 NSS_WS 信号请求传输数据时，发送流程开始。必须先使能从机，并且写入 I²S 数据寄存器之后，外部主机才能开始通信。

对于 I²S 的 MSB 对齐和 LSB 对齐模式，第一个写入数据寄存器的数据项对应左声道的数据。当通信开始时，数据从发送缓冲器传送到移位寄存器，然后标志位 TXE 置为‘1’；这时，要把对应右声道的数据项写入 I2S 数据寄存器。

标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。与主模式的发送流程相比，在从模式中，CHSIDE 取决于来自外部主 I2S 的 WS 信号。这意味着从机在接收到主机生成的时钟信号之前，就要准备好第一个要发送的数据。 WS 信号为‘1’表示先发送左声道。

注：在主器件发出的第一个时钟出现在 CK 线上时，必须至少提前 2 个 PCLK 周期置位 I2SE。

当发出第一位数据的时候，半字数据并行地通过 I2S 内部总线传输至 16 位移位寄存器，然后其它位依次按高位在先的顺序从引脚 MOSI/SD 发出。每次数据从发送缓冲器传送至移位寄存器时，标志位 TXE 置‘1’，如果寄存器 SPI_CR2 的 TXEIE 位为‘1’，则产生中断。注意，在对发送缓冲器写入数据前，要确认标志位 TXE 为‘1’。写入数据的操作取决于所选中的 I2S 标准，详见“支持音频协议”章节。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。如果在数据尚未写入 SPI_DR 寄存器时下一个数据通信的首个时钟边沿到来，下溢标志将置‘1’并可能产生中断。通过这种方式，软件可以获知所传输的数据不正确。如果 SPI_CR2 寄存器的 ERRIE 位置 ‘1’，则当 SPI_SR 寄存器中的 UDR 标志变为‘1’时，将产生中断。这种情况下，必须关闭 I²S 并从左通道开始重新启动数据传输。

要通过清除 I2SE 位关闭 I²S，必须先等待 TXE=1 并且 BSY=0。

- 接收流程

接收流程与发送流程类似，区别在“配置流程”的第 1 步中通过寄存器 I2SCFG[1:0]选择接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收，即每次填满接收缓存，标志位 RXNE 置‘1’，如果寄存器 SPI_CR2 的 RXNEIE 位为‘1’，则产生中断。按照不同的数据和声道长度设置，收到左声道或者右声道数据会需要 1 次或者 2 次传输数据至接收缓冲器的过程。

每次接收到数据(将要从 SPI_DR 读出)以后即更新 CHSIDE，它对应 I²S 单元产生的 WS 信号。读取 SPI_DR 寄存器，将清除 RXNE 位。读取数据的操作取决于所选中的 I²S 标准。

在还没有读出前一个接收到的数据，又接收到新数据时，即产生上溢，并设置标志位 OVR 为‘1’；如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断，指示发生了错误。

要关闭 I²S 功能时，需要在接收到最后一次 RXNE=1 时将 I2SE 位清‘0’。

注：外部主机器件需要有通过音频声道发送/接收 16 位或 32 位数据包的功能。

27.4.5 I²S 标志

27.4.5.1 状态标志

有 3 个状态标志位供用户监控 I²S 总线的状态。

- 忙标志位(BSY)

BSY 标志由硬件设置与清除(写入此位无效)，该标志位指示 I²S 通信层的状态。

该位为‘1’时表明 I²S 通讯正在进行中，但有一个例外：主接收模式(I2SCFG=11)下，在接收期间 BSY 标志始终为低。

在软件要关闭 I²S 模块之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输。

当传输开始时，BSY 标志被置为‘1’，除非 I²S 模块处于主接收模式。

下述情况时，该标志位被清除：

- 当传输结束时(除了主发送模式，这种模式下通信是连续的)；
- 当关闭 I²S 模块时。

当通信是连续的时候：

- 在主发送模式时，整个传输期间，BSY 标志始终为高；
- 在从模式时，每个数据项传输之间，BSY 标志在变低并持续 1 个 I2S 时钟周期内。
- 注：请勿使用 BSY 标志处理每次数据发送或接收，最好改用 TXE 标志和 RXNE 标志。
- 发送缓存空标志位(TXE)

该标志位为‘1’表示发送缓冲器为空，可以对发送缓冲器写入新的待发送数据。在发送缓冲器中已有数据时，标志位清‘0’。在 I²S 被关闭时(I2SE 位为‘0’)，该标志位也为‘0’。

- 接收缓存非空标志位(RXNE)

该标志位置‘1’表示在接收缓存里有接收到的有效数据。在读取寄存器 SPI_DR 时，该位清‘0’。

- 声道标志位(CHSIDE)

在发送模式下，该标志位在 TXE 为高时刷新，指示从 SD 引脚上发送的数据所在的声道。如果在从发送模式下发生了下溢错误，该标志位的值无效，在重新开始通讯前需要把 I²S 关闭再打开。

在接收模式下，该标志位在寄存器 SPI_DR 接收到数据时刷新，指示接收到的数据所在的声道。如果发生错误(如上溢 OVR)，该标志位无意义，需要将 I²S 关闭再打开(同时，如果必要修改 I2S 的配置)。

在 PCM 标准下，无论短帧格式还是长帧格式，这个标志位都没有意义。

如果寄存器 SPI_SR 的标志位 OVR 或 UDR 为‘1’，且寄存器 SPI_CR2 的 ERRIE 位为‘1’，则会产生中断，而后可以通过读寄存器 SPI_SR 来清除中断标志。

27.4.5.2 错误标志

I²S 单元有 2 个错误标志位。

下溢标志位(UDR)

在从发送模式下，如果数据传输的第一个时钟边沿到达时，新的数据仍然没有写入 SPI_DR 寄存器，该标志位会被置‘1’。在寄存器 SPI_I2SCFGR 的 I2SMOD 位置‘1’后，该标志位才有效。如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，就会产生中断。

通过对寄存器 SPI_SR 进行读操作来清除该标志位。

上溢标志位(OVR)

如果还没有读出前一个接收到的数据时，又接收到新的数据，即产生上溢，该标志位置‘1’，如果寄存器 SPI_CR2 的 ERRIE 位为‘1’，则产生中断指示发生了错误。这时，接收缓存的内容，不会刷新为从发送设备送来的新数据。对寄存器 SPI_DR 的读操作返回最后一个正确接收到的数据。其他所有在上溢发生后由发送设备发出的 16 位数据都会丢失。

通过先读寄存器 SPI_SR 再读寄存器 SPI_DR，来清除该标志位。

27.4.6 I²S DMA

在 I²S 模式下，DMA 的工作方式与在 SPI 模式下完全相同。除了由于不存在数据传输保护机制，I²S 模式下没有 CRC 功能外，没有其他差别。

27.4.7 I²S 中断

I²S 中断请求如下表：

表 27-3 I²S 中断请求表

中断事件	事件标志位	使能标志位
发送缓冲器空标志位	TXE	TXEIE
接收缓冲器非空标志位	RXNE	RXNEIE
下溢标志位	OVR	ERRIE
上溢标志位	UDR	

27.5 SPI/I²S 寄存器

27.5.1 SPI 控制寄存器 1 (SPI_CR1)

偏移地址:0x00

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	BIDIMODE	RW	0	双向数据模式使能。 0：“双线单向”模式 1：“单线双向”模式 注：该位不适用于 I²S 模式。
14	BIDIOE	RW	0	双向模式输出使能。 与 BIDIMODE 位一起配置“单线双向”模式下数据的输出方向。 0：输出禁止（只接收模式） 1：输出使能（只发送模式） 在主模式下，使用 MOSI 引脚；在从模式下，使用 MISO 引脚。 注：该位不适用于 I²S 模式。
13	CRCEN	RW	0	硬件 CRC 校验使能。 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。 注：该位不适用于 I²S 模式。
12	CRCNEXT	RW	0	下一个发送 CRC。 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。 注：该位不适用于 I²S 模式。
11	DFF	RW	0	数据帧格式 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。 注：该位不适用于 I²S 模式。
10	RXONLY	RW	0	仅接收控制。

				该位和 BIDIMODE 位一起决定在“双线单向”模式下的传输方向。在多个从机的配置中，在未被访问的从机上该位置 1，使得只有被访问的从机才有输出，因而不会造成数据线上有数据冲突。 0：全双工（发送和接收） 1：禁止输出（只接收模式） 注：该位不适用于 I^2S 模式。
9	SSM	RW	0	软件从机管理。 当 SSM 置位，NSS 引脚上的电平由 SSI 位的值决定。 0：禁止软件从机管理 1：使能软件从机管理 注：该位不适用于 I^2S 模式和 SPI TI 模式。
8	SSI	RW	0	内部从机选择。 该寄存器只有当 SSM=1 时才有效。该寄存器决定了 NSS 上的电平，在 NSS 引脚上的 I/O 操作无效。 注：该位不适用于 I^2S 模式和 SPI TI 模式。
7	LSBFIRST	RW	0	帧格式。 0：先发送 MSB 1：先发送 LSB 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I^2S 模式和 SPI TI 模式。
6	SPE	RW	0	SPI 使能。 0：禁止 SPI 1：使能 SPI 注：该位不适用于 I^2S 模式。
5:3	BR[2:0]	RW	3’h0	波特率控制。 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：$f_{PCLK}/256$ 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I^2S 模式。
2	MSTR	RW	0	主机选择。 0：配置为从机 1：配置为主机 通讯进行时不能改变该寄存器的值。 注：该位不适用于 I^2S 模式。
1	CPOL	RW	0	时钟极性。 0：空闲状态时，SCK 保持低电平 1：空闲状态时，SCK 保持高电平

				通讯进行时不能改变该寄存器的值。 注：除了在 TI 模式下应用 CRC 的情况外，该位不适用于 I²S 模式和 SPI TI 模式。
0	CPHA	RW	0	时钟相位。 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 通讯进行时不能改变该寄存器的值。 注：除了在 TI 模式下应用 CRC 的情况外，该位不适用于 I²S 模式和 SPI TI 模式。

27.5.2 SPI 控制寄存器 2 (SPI_CR2)

偏移地址:0x04

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRF	SAMPLE_SELECT	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	RXNEIE	ERRIE	CLRTXFIFO	Res.	SSOE	TXDMAEN	RXDMAEN
RW	RW							RW	RW	RW	RW		RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	FRF	RW	0	帧格式： 0：SPI Motorola 模式 1：SPI TI 模式 只有在禁止 SPI(SPE=0)后才能对此位进行操作
14	SAMPLE_SELECT	RW	0	采样方式： 0：半个 cycle 采样 1：1 个 cycle 采样
13:8	Reserved	-	-	保留
7	TXEIE	RW	0	发送缓冲区空中断使能 0：禁止 TXE 中断 1：使能 TXE 中断。TXE=1 时产生中断请求。
6	RXNEIE	RW	0	接收缓冲区非空中断使能 0：禁止 RXNE 中断 1：使能 RXNE 中断。RXNE=1 时产生中断请求。
5	ERRIE	RW	0	错误中断使能。 0：禁止错误中断 1：使能错误中断。当 CRCERR、OVR 或 MODF 为 1 时，产生中断请求。
4	CLRTXFIFO	RW	0	清空 TXFIFO 软件置位，硬件复位 0：没作用 1：清空 TXFIFO 注：只有当 SPI 禁止(SPE=0)时，才能写该位，否则无效。

3	Reserved	-	-	保留
2	SSOE	RW	0	SS 输出使能。 0：禁止在主模式下 SS 输出，该设备可以工作在多主机模式 1：开启主模式下 SS 输出，该设备不能工作在多主机模式。 注：I²S 模式下不使用。
1	TXDMAEN	RW	0	发送缓冲区 DMA 使能。 0：禁止发送缓冲区 DMA 1：使能发送缓冲区 DMA。当 TXE=1，则发出 DMA 请求。
0	RXDMAEN	RW	0	接收缓冲区 DMA 使能。 0：禁止接收缓冲区 DMA 1：使能接收缓冲区 DMA。当 TXE=1，则发出 DMA 请求。

27.5.3 SPI 状态寄存器 (SPI_SR)

偏移地址:0x08

复位值:0x0000 0002

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	FTLVL [1:0]		FRLVL [1:0]		FRE	BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
			R	R	R	R	R	R	R	R	RC_W0	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12:11	FTLVL	R	2'h0	FIFO 发送级别。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满(当 FIFO 阈值大于 1/2，即认为是满) 注：I²S 模式下不使用。
10:9	FRLVL	R	2'h0	FIFO 接收级别。硬件置位，硬件清零 00：FIFO 空 01：1/4 FIFO 10：1/2 FIFO 11：FIFO 满 注：I²S 模式和带 CRC 校验的 SPI 仅接收模式下不使用。
8	FRE	R	0	FRE：帧格式错误。 该标志用于 SPI TI 从模式。 此标志由硬件置 1，在读取 SPI_SR 时由软件复位。

				0：未发生帧格式错误 1：发生帧格式错误
7	BSY	R	0	忙标志。 0：SPI 不忙； 1：SPI 处于通讯，或者发送缓冲非空。
6	OVR	R	0	溢出标志。 0：无溢出错误 1：产生溢出错误 该寄存器由硬件置位，或者软件序列复位（上溢和下溢序列不同）。
5	MODF	R	0	模式故障。 0：无模式故障 1：发生模式故障 该寄存器由硬件置位，或者软件序列复位。 注：I^2S 模式下不适用。
4	CRCERR	R	0	CRC 错误标志。 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。 注：I^2S 模式下不适用。
3	UDR	R	0	下溢标志位。 0：未发生下溢； 1：发生下溢错误。 硬件置位，软件序列清零。 注： SPI 模式下不适用。
2	CHSIDE	R	0	声道控制。 0：发送或者接收左声道； 1：发送或者接收右声道。 注：在 SPI 和 I^2S PCM 模式下不适用。
1	TXE	R	1	发送缓冲空。 0：发送缓冲非空 1：发送缓冲为空
0	RXNE	R	0	接收缓冲非空。 0：接收缓冲非空 1：接收缓冲为空

27.5.4 SPI 数据寄存器 (SPI_DR)

偏移地址:0x0C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	DR[15:0]	RW	16'h0	数据寄存器。 要发送或者接收到的数据。 数据寄存器作为 RxFIFO 和 TxFIFO 的接口。当要读数据，实际访问 RxFIFO，而要写数据，实际访问 TxFIFO。 注：取决于 DFF 位（数据帧宽度选择），数据发送或者接收是 8 位或者 16 位。 对于 8 位数据帧，数据寄存器是基于右对齐的 8 位数据进行发送和接收的。当在接收模式，DR[15:8]硬件置为 0。 对于 16 位数据帧，数据寄存器是 16 位的，整个 DR[15:0]都用作发送和接收。

27.5.5 SPI CRC 多项式寄存器 (SPI_CRCPR)

偏移地址:0x10

复位值:0x0000 0007

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
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	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	CRCPOLY[15:0]	RW	16'h7	CRC 多项式寄存器。 该寄存器包含了 CRC 计算时用到的多项式。 其复位值为 0x0007，根据应用可以设置其他数值。 注：在 I^2S 模式下不适用。 注：多项式值只能是奇数，不支持偶数值。

27.5.6 SPI 接收 CRC 寄存器 (SPI_RXCRC)

偏移地址:0x14

复位值:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	RXCRC[15:0]	R	16'h0	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 的标准。 注：当 BSY 标志为‘1’时读该寄存器，将可能读到不正确的数值。 注：在 I²S 模式下不适用。

27.5.7 SPI 发送 CRC 寄存器 (SPI_TXCRC)

偏移地址:0x18

复位值:0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	TXCRC[15:0]	R	16’h0	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 的标准。 注：当 BSY 标志为‘1’时读该寄存器，将可能读到不正确的数值。 注：在 I²S 模式下不适用。

27.5.8 SPI_I2S 配置寄存器 (SPI_I2SCFGR)

偏移地址:0x1C

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	I2SMOD	I2SE	I2SCFG[1:0]		PCMSYNC	Res.	I2SSTD[1:0]		CKPOL	DTALEN[1:0]		CHLEN
				RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:12	Reserved	-	-	保留
11	I2SMOD	RW	0	I²S 模式选择。 0: 选择 SPI 模式 1: 选择 I²S 模式 注：该位只有在关闭了 SPI 或者 I²S 时才能设置。 注：该寄存器如果不支持 I²S 功能，则固定为 0.

10	I2SE	RW	0	I^2S 使能。 0：关闭 I^2S 1：I^2S 使能 注：在 SPI 模式下不适用。 注：该寄存器如果不支持 I^2S 功能，则固定为 0.
9:8	I2SCFG	RW	0	I^2S 模式设置。 00: 从机发送 01: 从机接收 10: 主机发送 11: 主机接收 注：该位只有在关闭了 I^2S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I^2S 功能，则固定为 0.
7	PCMSYNC	RW	0	PCM 帧同步。 0: 短帧同步 1: 长帧同步 注: 该位只在 I2SSTD = 11 (使用 PCM 标准)时有意义。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I^2S 功能，则固定为 0.
6	Reserved	-	-	保留
5:4	I2SSTD	RW	0	I^2S 标准选择。 00: I^2S 飞利浦标准 01: 高字节对齐标准 (左对齐) 10: 低字节对齐标准(右对齐) 11: PCM 标准 注：为了正确操作，只有在关闭了 I^2S 时才能设置该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I^2S 功能，则固定为 0.
3	CKPOL	RW	0	无效状态时钟极性。 0: I^2S 时钟无效状态为低电平 1: I^2S 时钟无效状态为高电平 注：为了正确操作，该位只有在关闭了 I^2S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I^2S 功能，则固定为 0.
2:1	DATLEN	RW	0	待传输数据长度。 00: 16 位数据长度 01: 24 位数据长度 10: 32 位数据长度 11: 不允许 注：为了正确操作，该位只有在关闭了 I^2S 时才能设置。 在 SPI 模式下不适用。

				注：该寄存器如果不支持 I²S 功能，则固定为 0.
0	CHLEN	RW	0	声道长度 (每个音频声道的数据位数)。 0: 16 位宽 1: 32 位宽 只有在 DATLEN = 00 时该位的写操作才有意义，否则声道长度都由硬件固定为 32 位。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.

27.5.9 SPI_I2S 预分频器寄存器 (SPI_I2SPR)

偏移地址:0x20

复位值:0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	MCKOE	ODD	I2SDIV[7:0]							
						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	MCKOE	RW	0	主机时钟输出使能 0: 关闭主机时钟输出 1: 主机时钟输出使能 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.
8	ODD	RW	0	预分频器的奇数因子。 0: 实际分频系数 = I2SDIV *2 1: 实际分频系数 = (I2SDIV * 2)+1 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.
7:0	I2SDIV	RW	8'h0	I²S 线性预分频器。 禁止设置 I2SDIV [7:0] = 0 或者 I2SDIV [7:0] = 1。 注：为了正确操作，该位只有在关闭了 I²S 时才能设置。仅在 I²S 主机模式下使用该位。 在 SPI 模式下不适用。 注：该寄存器如果不支持 I²S 功能，则固定为 0.

28. 内部集成电路接口(I²C)

28.1 I²C 简介

I²C(Inter-integrated circuit)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 总线特定的顺序、协议、仲裁和时序。支持标准（Sm）、快速（Fm）、快速增强模式（Fm+）。根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

28.2 I²C 功能描述

28.2.1 简介

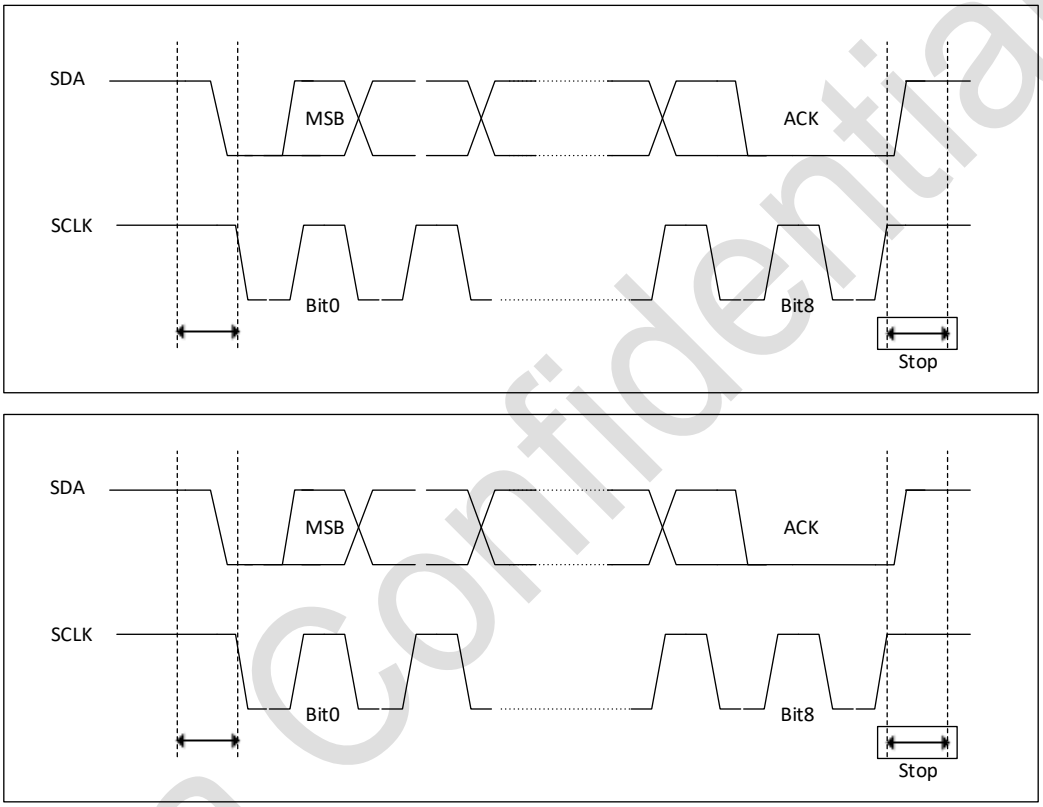


图 28-1 I²C 总线协议

I²C 支持以下四种模式：

- 从发送器模式 （Slave transmitter）
- 从接收器模式 （Slave receiver）
- 主发送器模式（Master transmitter）
- 主接收器模式（Master receiver）

默认情况下，I²C 接口总是工作在从模式。接口在生成起始条件后，自动地从从模式到主模式；在允许多主机功能的情况下，当仲裁丢失或产生停止信号时，则从主模式切换到从模式

作为主机，I²C 接口启动数据传输，并产生时钟信号。串行数据的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机模式下由软件控制产生。

作为从机，I²C 接口能识别自己的地址(7 位或 10 位)和广播呼叫地址。软件能够控制开启或禁止对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 master 模式发送。

在一个字节传输后的第一个始终时钟内，接收方必须回送一个应答位（ACK）给发送方。

28.2.2 时钟

I²C 模块的输入时钟是 APB 时钟。模块内部生成 SCL 时钟再输出到总线上给从机。

SCL 产生机制如下：有一个计数器根据 CCR 寄存器的值来计数高低电平（CCR 寄存器决定高低电平的长度）。此外，CCR 寄存器还控制标准模式，快速模式和快速增强模式的选择，以及 SCL 高低电平的占空比。

28.2.3 包错误校验（PEC）

包错误校验(PEC)计算器是用于提高通信的可靠性，这个计算器使用一个可编程的多项式对每一位串行数据进行计算。

- PEC 计算由 I2C_CR1 寄存器的 ENPEC 位使能。PEC 使用 CRC-8 算法对所有信息字节进行计算，包括地址和读/写位在内。
  - 在发送时：在最后一个 TxE 事件时设置 I2C_CR1 寄存器中的 PEC 位，PEC 将在这个字节后被发送。
  - 在接收时：在最后一个 RxNE 事件时设置 I2C_CR1 寄存器中的 PEC 位，如果下个接收到的字节不等于内部计算的 PEC，接收器会发送一个 NACK。如果是主接收器，不管校对的结果如何，PEC 后都将发送 NACK。PEC 位必须在接收当前字节的 ACK 脉冲之前设置。
- 在 I2C_SR1 寄存器中可获得 PECERR 错误标记/中断。
- 如果 DMA 和 PEC 计算器都被激活：
  - 在发送时：当 I²C 接口从 DMA 控制器处接收到 EOT(对应 DMA 传输字节数) 信号时，它在最后一个字节后自动发送 PEC。
  - 接收时：当 I²C 接口从 DMA 处接收到一个 EOT_1(对应 DMA 传输字节数-1) 信号时，它将自动把下一个字节作为 PEC，并且将检查它。在接收到 PEC 后产生一个 DMA 请求。
- 为了允许中间 PEC 传输，在 I2C_CR2 寄存器中有一个控制位(LAST 位)用于判别是否真是最后一个 DMA 传输。如果确实是最后一个主接收器的 DMA 请求，在接收到最后一个字节后自动发送 NACK。
- 仲裁丢失时 PEC 计算失效。

28.2.4 I²C 的从模式

默认情况下，I²C 接口总是工作在 slave 模式。从 slave 模式切换到 master 模式，需要产生一个起始条件。为了产生正确的时序，必须在 I2C_CR2 寄存器中设定该模块的输入时钟。输入时钟的频率必须至少是：

标准模式下为：4 MHz

快速模式下为：8 MHz

快速增强模式下为：16 MHz

设置从机地址及地址屏蔽:

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，并与芯片的地址 OAR1 或者广播呼叫地址（如果 ENGC=1)相比较。如果启用了额外地址（ENDUAL=1）,地址会和芯片的 OAR2 地址进行比较，并且 OAR2 寄存器中的地址可以通过 OA2MSK[2:0]进行屏蔽，屏蔽从 OA2 高第 7 位开始，

当 OA2MSK 配置为 1 到 6 时，将分别只有 OA2[7:2]、OA2[7:3]、OA2[7:4]、OA2[7:5]、OA2[7:6] 或 OA2[7] 与接收到的地址作比较。如果 OA2MSK=7，接收到的所有 7 位地址（保留地址除外）均得到应答。

地址不匹配（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。

地址匹配（地址匹配）：I2C 接口产生以下时序：

- 如果 ACK 被软件置'1'，则产生一个应答脉冲
- 硬件置位 ADDR 位，如果设置了 ITEVTEN 位，则产生中断
- 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

从机发送模式

在接收到地址并清除 ADDR 位后，（如果地址字节的最低位是 1）Slave 将数据（字节）从 DR 寄存器，经由内部移位寄存器发送到 SDA 上。

Slave 拉低 SCL，直到 ADDR 位被清除，并且待发送数据已写入 DR 寄存器（参考 EV1、EV3）。

当收到应答脉冲时：TxE 位被硬件置位，如果设置了 ITEVTEN 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。Slave 拉低 SCL，直到 BTF 位被软件清零（读 I2C_SR1 之后，再写入 I2C_DR 寄存器）。

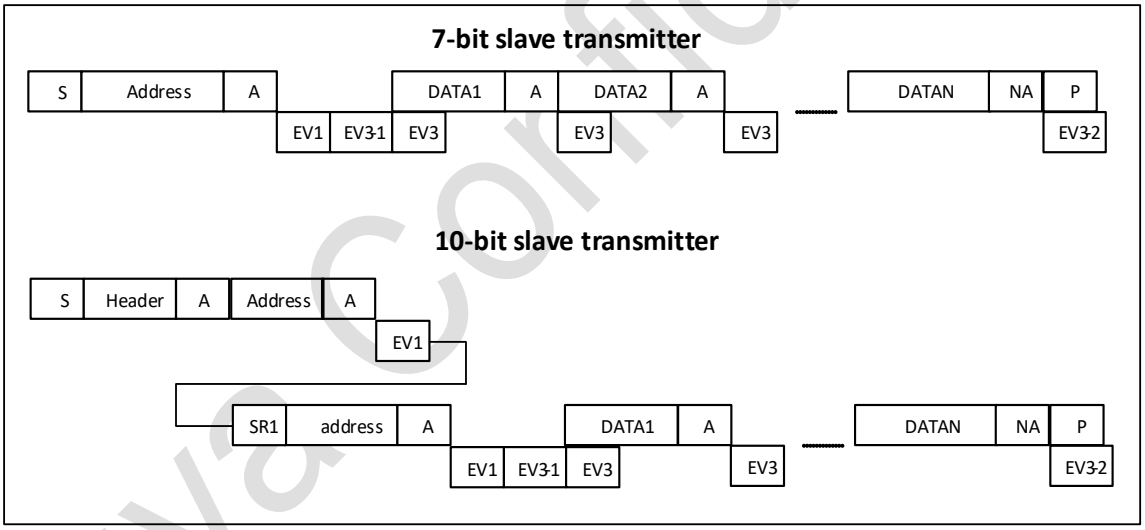


图 28-2 从发送器的传送序列图

说明：S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV1: ADDR=1，通过先读 SR1 寄存器，再读 SR2 寄存器，清零 ADDR 位

EV3-1：TxE=1，移位寄存器为空，数据寄存器为空，向 DR 寄存器写 Data1

EV3：TxE=1，移位寄存器不为空，数据寄存器为空，向 DR 寄存器写（Data2）清零 TxE

EV3-2: AF=1；软件向 AF 位写 0 清零该位

注:

1 ）EV1 和 EV3-1 事件拉低 SCL，直到对应的软件序列结束。

2 ）EV3 的软件序列必须在当前字节传输结束之前完成

从机接收模式

在接收到地址并清除 ADDR 后，（如果地址字节的最低位是 0）slave 将通过内部移位寄存器把从 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 ITEVTEN 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，则 BTF 位被置位，在清除 BTF（读出 I2C_SR1 之后再读 I2C_DR 寄存器）之前，slave 一直拉低 SCL。(见下图)。

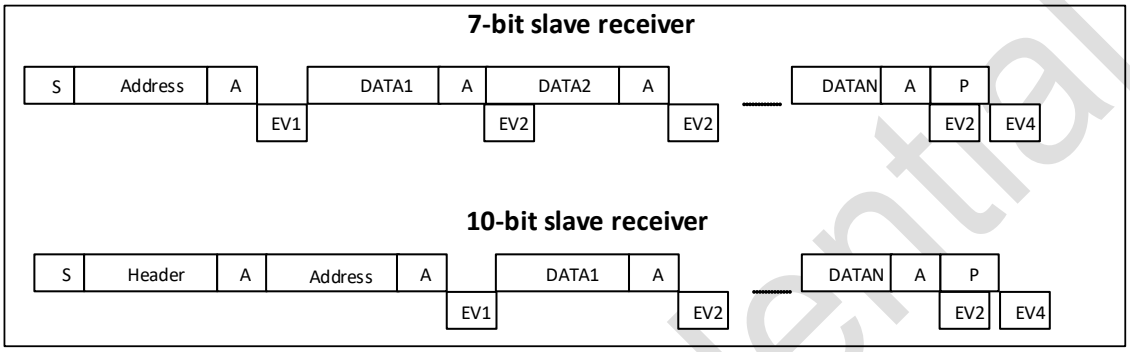


图 28-3 从接收器的传送序列图

说明：S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV1：ADDR=1，通过先读 SR1，后读 SR2 实现 ADDR 的清零

EV2：RxNE=1，读 DR 寄存器清零该位

EV4：STOPF=1，通过先读 SR1 寄存器，后写 CR1 寄存器实现对该位的清零。

注:

1） EV1 事件拉低 SCL，直到相应软件序列结束。

2） EV2 软件序列必须在当前字节传输完成之前完成。

3） 当用户检查 SR1 寄存器内容后，应该对每个置位的标志位，进行完整的清除序列。比如 ADDR 和 STOPF 标志位，需要用以下序列：

如果 ADDR=1，先读 SR1，再读 SR2；如果 STOPF=1，先读 SR1，再写 CR1。

这样做的目的是确保如果 ADDR 和 STOPF 都被置位，都能被发现并且清除掉。

关闭通信

在传输完最后一个数据字节后，主机产生一个停止条件，从机检测到该条件时：

- 硬件置位 STOPF，如果设置了 ITEVTEN 位，则产生一个中断。

通过先读 SR1，后写 CR1，实现对 STOPF 位的清零。

28.2.5 I²C 的主模式

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，设备就进入了主机模式。

以下是主机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 I2C_CCR 寄存器

- 配置 I2C_TRISE 寄存器
- 配置 I2C_CR1 寄存器中的 PE 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：4 MHz
- 快速模式下为：8 MHz
- 快速增强模式下为：16 MHz

主机产生时钟

CCR 寄存器以输入时钟上升沿计数，产生 SCL 的高低电平。由于从机可能拉长 SCL 信号，在 SCL 上升沿产生后，主机在 I2C_TRISE 寄存器所设置的时间到达时，检测来自总线的 SCL 信号。

- 如果 SCL 是低电平，意味着从机正在拉长 SCL 总线，高电平计数器停止计数，直到 SCL 被检测到高电平。这是为了确保 SCL 参数的最小高电平时间。
- 如果 SCL 是高电平，高电平计数器保持计数。

实际上，即使从机不拉长 SCL，从 SCL 上升沿产生，到 SCL 上升沿被发现，这样的反馈回路也是要花费些时间的。这个回路的时间与 SCL 的上升时间有关系，再加上 SCL 输入路径的模拟噪声滤波延时，以及芯片内部由于用 APB 时钟进行的 SCL 同步时间。反馈回路的最大时间在 TRISE 寄存器中设置，所以无论 SCL 上升时间如何，SCL 的频率保持稳定。

起始条件

当 BUSY=0 时，设置 START=1，I²C 接口将产生一个起始条件，并切换至主机模式(MSL 被置位)。

注：在主机模式下设置 START 位，将在当前字节传输完成后，由硬件产生一个重开始条件。

一旦发出起始条件：

- SB 位被硬件置位，如果设置了 ITEVTEN 位，则会产生一个中断。
- 主机读 I2C_SR1 寄存器，再把从机地址写入 I2C_DR 寄存器。

从机地址发送

主机将从机的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。
  - 然后主机读 SR1 寄存器，并且将第二个地址字节写入 DR 寄存器。
  - ADDR 位被硬件置位，如果设置了 ITEVFEN 位，则产生一个中断。
  - 随后主机读 SR1 寄存器，然后读 SR2 寄存器。
- 在 7 位地址模式时，将送出一个地址字节。

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。

根据送出从机地址的最低位，主设备决定进入是发送器模式，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让最低位等于 0。
  - 要进入接收器模式，主设备发送从地址时让最低位等于 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)，然后送最低位等于 0 的从地址。(头段字节中的 xx 是 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)，然后送最低位等于 1 的从地址。然后再重新发送一个起始条件，后面跟着头字节(11110xx1)。(头字节中的 xx 是 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

主机发送模式

在发送了地址和清除了 ADDR 位后,主机通过内部移位寄存器将数据字节从 DR 寄存器发送到 SDA 线上。

主机等待，直到第一个数据字节被写入 DR 寄存器（参见 EV8_1）。

当收到 ACK 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，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（读 I2C_SR1 之后，再写 I2C_DR 寄存器）之前，I2C 接口将保持 SCL 为低电平。

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(见图的 EV8_2)，然后 I²C 接口将自动回到从模式(MSL 位清除)。

注：当 TxE 或 BTF 位置位时，在出现 EV8_2 事件时产生停止条件。

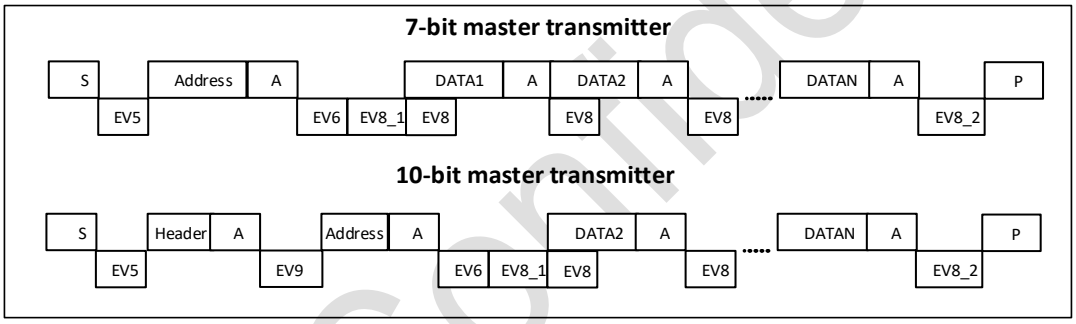


图 28-4 主发送器传送序列图

说明: S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledg 响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV5: SB=1，读 SR1 然后将地址写入 DR 寄存器将清除该事件

EV6: ADDR=1,读 SR1，再读 SR2，清除该事件

EV8_1: TXE=1，移位寄存器空

EV8: TXE=1，写 DR 寄存器清除该事件

EV8_2: TXE=1，BTF=1，产生停止条件时由硬件清除

EV9: ADDR10=1,读 SR1 然后写 DR 寄存器清除该事件

注:

1．EV5, EV6, EV8_1 和 EV8_2 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

2．EV8 软件序列必须在当前字节发送完成前执行完毕。若 EV8 的软件序列不能在当前传输的字节结束前完成，则推荐使用 BTF 代替 TxE

主机接收模式

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。在接收到每个字节后，I²C 接口依次执行以下操作：

■ 如果 ACK 位被置位，则发出一个应答脉冲。

■ 硬件设置 RxNE=1，如果设置了 INEVFEN 和 ITBUFEN 位，则会产生一个中断。

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：该方法的应用场景是，当 I²C 的中断被设成应用程序中最高优先级

主机在接收到从机最后一个字节后，发送一个 NACK。接收到 NACK 后，从机释放对 SCL 和 SDA 线的控制。主机就可以发送一个停止/重开始条件。

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/重开始条件，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)设置停止/开始位。
3. 当接收单个字节时，清除 ACK 和停止条件的产生位要刚好在 EV6 之后(EV6_1 时，清除 ADDR 之后)。在产生了停止条件后，I²C 接口自动回到从模式(MSL 位被清除)。

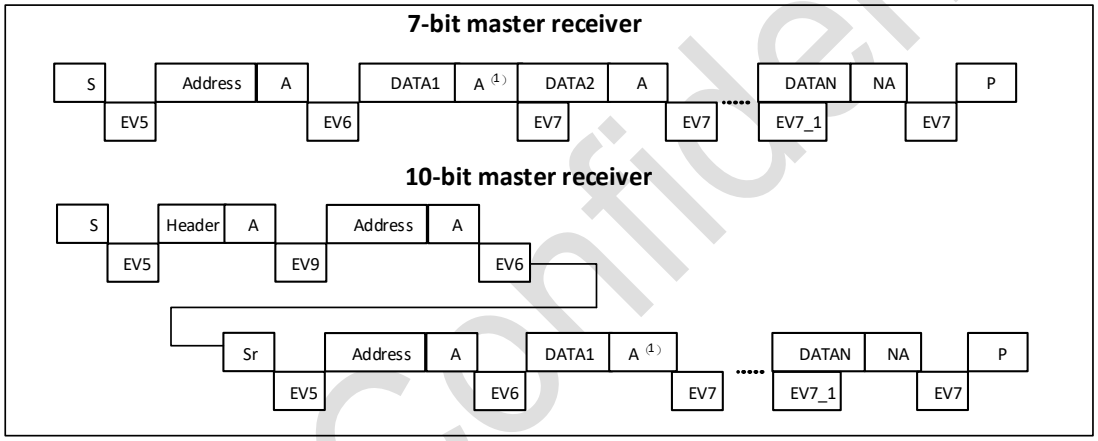


图 28-5 方法 1：主模式发送时的时序

说明: S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV5：SB=1，读 SR1，再写 DR 寄存器，该位被清零

EV6：ADDR=1，读 SR1，再读 SR2，该位被清零

EV6_1：无相关的标志事件，仅用作 1 个字节的接收。

EV7：RxNE=1，读 DR 寄存器，该位被清零

EV7_1：RxNE=1，读 DR 寄存器，写 ACK=0 并置位 STOP

EV9： ADD10=1,读 SR1 然后写 DR 寄存器清除该事件

注:

如果是单个字节接收，则上述标注为（1）的地方会是 NA

EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

EV7 软件序列必须在当前字节发送完成前执行完毕。在 EV7，软件序列不能在当前传输的字节传输完成前执行完毕，则推荐使用 BTF 代替 RXNE。

EV6_1 或者 EV7_1 的软件序列必须在当前字节传输的 ACK 位发送之前完成。

方法 2: 这个方法的应用场景是，I2C 的中断在应用中不是最高优先级，或者使用查询方式
用这个方法，如果 DataN-2 没有被读，在 DataN-1 之后，通讯会被拉长（RxNE 和 BTF 都被置位）。然后，在读 DR 寄存器的 DataN-2 前，清 ACK 位，以确保 ACK 位在 DataN ACK 之前被清掉。在读 DataN-2 之后，置位 STOP/START 位，并读 DataN-1。在 RxNE 置位后，读 DataN。

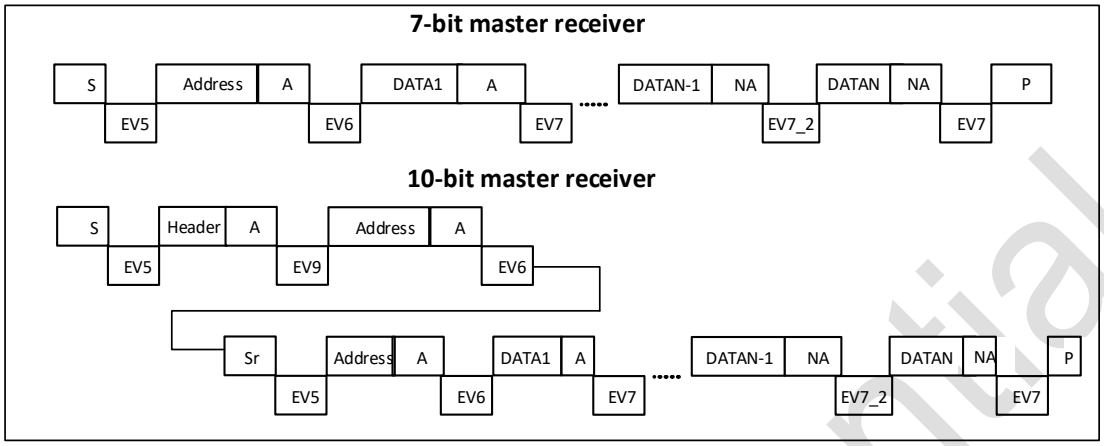


图 28-6 方法 2：N>2 时主模式发送时的时序

说明: S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)
EV5：SB=1，先读 SR1 寄存器先读 SR2 寄存器，再写 DR 寄存器，清零该位
EV6：ADDR，先读 SR1，再读 SR2，清零该位
EV7：RxNE=1，读 DR 寄存器清零该位
EV7_2：BTF=1, DataN-2 存在 DR 寄存器中，DataN-1 存在移位寄存器中，写 ACK=0，读 DR 寄存器中的 DataN-2。置位 STOP，读 DataN-1
EV9：ADD10=1,读 SR1 然后写 DR 寄存器清除该事件
注：

1. EV5, EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束
2. EV7 软件序列必须在当前字节发送完成前执行完毕。在 EV7，软件序列不能在当前传输的字节传输完成前执行完毕，则推荐使用 BTF 代替 RXNE。

- 当 3 个字节要被读走：
  - RXNE=1，DataN-2 没有读。
  - DataN-1 接收
  - BTF=1，移位和 DR 寄存器满。DR 寄存器存放了 DataN-2，移位寄存器存放了 DataN-1。此时，SCL 拉低，总线上没有其他要被接收的数据
  - 清零 ACK 位
  - 读 DR 寄存器中的 DataN-2，这将启动移位寄存器对 DataN 的接收
  - DataN 接收完成（发送 NACK 条件）
  - 写 START 或者 STOP 位
  - 读 DataN-1
  - RxNE=1
  - 读 DataN

以上流程是针对 N > 2 的描述。1 个字节和 2 个字节的接收，要用不同的处理方式，参见以下描述：

- 2 个字节接收的情况

- 置位 POS 和 ACK 位
- 等待 ADDR 置位
- 清零 ADDR 位
- 清零 ACK 位
- 等待 BTF 被置位
- 写 STOP 位
- 读 DR 两次

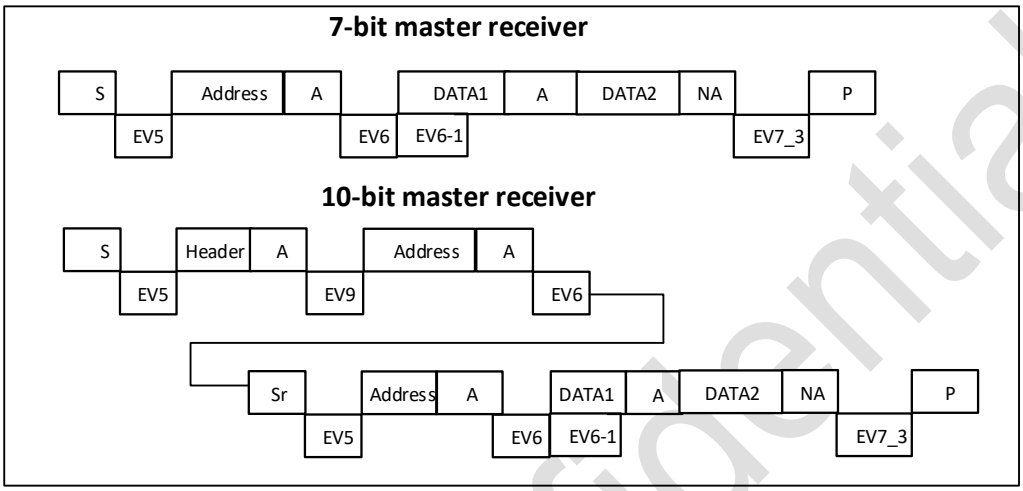


图 28-7 方法 2：N=2 时主模式发送时的时序

说明: S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV5：SB=1，先读 SR1 寄存器，再写 DR 寄存器，清零该位

EV6：ADDR=1，先读 SR1 寄存器，后读 SR2 寄存器，清零 ADDR 位

EV6_1：无相关的标志位事件。在 EV6 后，也就是地址被清零后，ACK 应该被清零

EV7_3：BTF=1，写 STOP=1，之后读两次 DR（Data1 和 Data2）

EV9: ADD10=1,读 SR1 然后写 DR 寄存器清除该事件

注:

1.EV5，EV6 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束

2.EV6_1 的软件序列必须在当前字节传输的 ACK 位发送之前完成。

■ 单个字节接收的情况

- 在 ADDR 事件里，清零 ACK 位
- 清零 ADDR
- 写 STOP 或者 START 位
- 在 RxNE 标志置位后，读数据

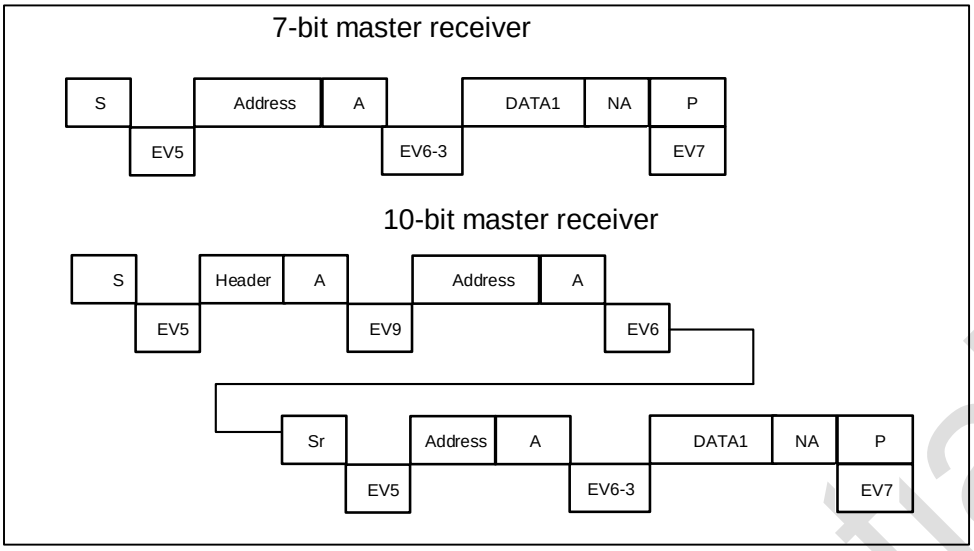


图 28-8 方法 2: N=1 时主模式发送时的时序

说明: S= Start（起始条件），SR= Repeated Start（重复的起始条件），P= Stop（停止条件），A= Acknowledge（响应），NA= Non-acknowledge（不响应），EVx= Event(ITEVFEN= 1 时产生中断)

EV5：SB=1，先读 SR1 寄存器，再写 DR 寄存器，清零该位

EV6_3：ADDR=1，写 ACK=0。先读 SR1 寄存器，后读 SR2 寄存器，清零 ADDR 位。在 ADDR 被清零后，写 STOP=1

EV7：RxNE=1，读 DR 寄存器清零该位

EV9：ADD10=1,读 SR1 然后写 DR 寄存器清除该事件

注:

EV5、EV6、EV8_1 和 EV8_2 事件，拉长 SCL 的低电平，直到相应的软件序列执行结束。

28.2.6 错误标志

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；
- 在从机模式：数据被丢弃，硬件释放总线。
  - 如果是错误的起始条件，从机认为是一个重开始，并等待地址或停止条件
  - 如果是错误的停止条件，从机按正常的停止条件操作，同时硬件释放总线
- 在主机模式：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。

应答失败（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。
  - 如果是处于从机模式，硬件释放总线
  - 如果是处于主机模式，软件必须生成一个停止条件或者重开始

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机发送重开始条件之后响应
- 硬件释放总线

过载/欠载错误 (OVR)

在从机模式下，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

在从机模式下，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运行错误时，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 之后且在第一个 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。

28.2.7 DMA 功能

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 必须在当前字节传输结束之前被初始化和使能。DMAEN 位（I2C_CR2 寄存器中）必须在 ADDR 事件发生前使能。

在 master 或者 slave 模式，当时钟延长功能使能，DMAEN 可以在清零 ADDR 之前的 ADDR 事件期间置位。DMA 请求必须在当前字节传输完成之前响应。当 DMA 传输数据长度达到 DMA 设定的值时，DMA 控制器向 I²C 发送 EOT（End of transfer），并产生传输完成中断（如果中断使能位有效）：

- 主机发送模式：在 EOT 中断服务程序中，需禁止 DMA 请求，然后在等到 BTF 事件后，置位 STOP 条件。
- 主机接受模式：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(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。

28.2.8 支持从 Stop 模式唤醒

在 MCU 处于 Stop 模式下，可以通过 I²C 接收地址，若地址匹配则可以唤醒 MCU，反之则保持 Stop 状态。

1）实现以上功能需要在进入 Stop 模式之前使能 I2C_CR1 的 PE 和 WUPEN，并且在 RCC 中使能 I2C 的时钟。

2）Stop 模式唤醒过程还支持可配置的 timeout 时间，可通过设置 I2C_CR1 的 WKUP_CNT 和 WKUP_DIV 进行配置，若使能 ITERREN，则在产生 TIMEOUT 之后唤醒 MCU，若不使能 ITERREN，则在产生 timeout 之后保持 Stop 状态。

3)进入 Stop 模式前,需要根据当前的通信速度合理配置 I2C_CR1. WKUP_CNT 和 I2C_CR1.WKUP_DIV，否则会导致 timeout，地址无法匹配。

28.2.9 系统管理总线 SMbus

系统管理总线(SMBus)是一个双线接口。通过它，各设备之间以及设备与系统的其他部分之间可以互相通信。它基于 I²C 操作原理。SMBus 为系统和电源管理相关的任务提供一条控制总线。一个系统利用 SMBus 可以和多个设备互传信息，而不需使用独立的控制线路。

系统管理总线(SMBus)标准涉及三类设备。从设备：接收或响应命令的设备。主设备：用来发布命令，产生时钟和终止发送的设备。主机：是一种专用的主设备，它提供与系统 CPU 的主接口。主机必须具有主-从机功能，并且必须支持 SMBus 提醒协议。在一个系统里只允许有一个主机。

利用系统管理总线，设备可提供制造商信息，告诉系统它的型号/部件号，保存暂停事件的状态，报告不同类型的错误，接收控制参数，和返回它的状态。SMBus 为系统和电源管理相关的任务提供控制总线。

本产品的 I²C 模块支持 SMbus/PMbus 功能。

SMbus 与 I²C 比较的不同点：

表 28-1 SMbus 与 I²C 不同点

SMBus	I²C
最大传输速度 100 kHz	最大传输速度 400 kHz
最小传输速度 10 kHz	无最小传输速度
35ms 时钟低超时	无时钟超时
固定的逻辑电平	逻辑电平由 V_{DD} 决定
不同的地址类型(保留、动态等)	7 位、 10 位和广播呼叫从地址类型
不同的总线协议（快速命令、处理呼叫等)	无总线协议

SMBus 和 I²C 之间的相似点：

- 两条线的总线协议(1 个时钟，1 个数据) + 可选的 SMBus 提醒线
- 主-从通信，主设备提供时钟
- 多主机功能
- SMBus 数据格式类似于 I2C 的 7 位地址格式

地址解析协议(ARP)

SMBus 的从地址冲突可以通过给每个从设备动态地分配一个新的唯一地址来解决。

ARP 有以下的属性：

- 地址分配利用标准 SMBus 物理层仲裁机制
- 当设备维持供电期间，分配的地址仍保持不变，允许设备在断电时保留其地址。
- 在地址分配后，没有额外的 SMBus 的打包开销(也就是说访问分配地址的设备与访问固定地址的设备所用时间是一样的）。
- 任何一个 SMBus 主设备可以遍历总线。

SMBus 提醒模式（ALERT）

SMBus 提醒是一个带中断线的可选信号，用于那些希望扩展它们的控制能力而牺牲一个引脚的设备。SMBALERT 和 SCL、SDA 信号一样，是一种线与信号。SMBALERT 通常和 SMBus 广播呼叫地址一起使用。与 SMBus 有关的消息为 2 字节。

一个只具有从功能的设备，可以通过 SMBALERT 发信号给主机表示它希望进行通信，这可通过设置 I2C_CR1 寄存器上的 ALERT 位实现。主机处理该中断并通过提醒响应地址 ARA(Alert Response Address，地址值为 0001100x)访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应

答 ARA。此状态是由 I2C_SR1 寄存器中的 SMBALERT 状态标记来标识的。主机执行一个修改过的接收字节操作。由从发送设备提供的 7 位设备地址被放在字节的 7 个最高位上，第八个位可以是 0 或 1。如果多个设备把 SMBALERT 拉低，最高优先级设备(最小的地址)将在地址传输期间通过标准仲裁赢得通信权。在确认从地址后，此设备不得再拉低它的 SMBALERT，如果当信息传输完成后，主机仍看到 SMBALERT 低，就知道需要再次读 ARA。

没有实现 SMBALERT 信号的主机可以定期访问 ARA。

超时错误（TIMEOUT）

在定时规范上 I²C 和 SMBus 之间有很多差别。

SMBus 定义一个时钟低超时，35ms 的超时。SMBus 规定 $T_{LOW:SEXT}$ 为从设备的累积时钟低扩展时间。SMBus 规定 $T_{LOW:MEXT}$ 为主设备的累积时钟低扩展时间。更多超时细节请参考 2.0 版的 SMBus 规范 (http://smbus.org/specs/)。I2C_SR1 中的状态标志 TIMEOUT 错误表明了这个特征的状态。

使用 SMBus 模式的接口

将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 和 TEXTEN 位置 1 来使能超时检测。定时器必须按如下方式编程：即在 SMBus 规范规定的时间最大值之前检测出超时情况。

■ **$t_{TIMEOUT}$ 检查**

要使能 $t_{TIMEOUT}$ 检查，必须将 12 位 TIMEOUTA[11:0]位编程为定时器重载值，以检查 $t_{TIMEOUT}$ 参数。必须将 TIDLE 位配置为“0”，以检测 SCL 低电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 的低电平持续时间超过 (TIMEOUTA+1) x 2048 x t_{PCLK}，I2C_SR1 寄存器的 TIMEOUT 标志将置 1。

（最大 $t_{TIMEOUT}$ = 25 ms）。

注： TIMEOUTEN 位置 1 时，不允许更改 TIMEOUTA[11:0] 位和 TIDLE 位的配置。

■ **$t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 检查**

必须根据外设配置为主设备还是从设备来配置 TIMEOUTB 定时器，以便为从设备校验 $t_{LOW:SEXT}$，为主设备校验 $t_{LOW:MEXT}$。由于标准只规定了最大值，用户可以为这两个参数选择相同的值。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TEXTEN 位置 1 来使能定时器。

如果 SMBus 外设延展 SCL 的累积时间超过 (TIMEOUTB+1) x 2048 x t_{PCLK}，则 I2C_SR1 寄存器中的 TIMEOUT 标志将置 1。

请参见下：不同 PCLK 频率下的 TIMEOUTB 设置示例。

注： TEXTEN 位置 1 时，不允许更改 TIMEOUTB 配置。

总线空闲检测

要使能 t_{IDLE} 检查，必须将 12 位 TIMEOUTA[11:0] 字段编程为定时器重载值，以获取 t_{IDLE} 参数。必须将 TIDLE 位配置为“1”，以检测 SCL 和 SDA 高电平超时。

然后，通过将 I2C_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1 来使能定时器。

如果 SCL 和 SDA 线的高电平持续时间超过 (TIMEOUTA+1) x 4 x t_{PCLK}，I2C_SR1 寄存器中的 TIMEOUT 标志将置 1。

请参见表:不同 PCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs）。

注： TIMEOUTEN 置 1 时，不允许更改 TIMEOUTA 和 TIDLE 配置。

PMbus

PMbus 是基于 SMbus 而来的，传输逻辑与 SMbus 完全一致，区别在于 PMbus 定义了一些与电源管理相关的功能（由软件完成）。

SMBus: I2C_TIMEOUTR 寄存器配置示例

仅当支持 SMBus 功能时，才涉及本节内容。

将 $t_{TIMEOUT}$ 的最大持续时间配置为 25 ms：

表 28-2 不同 PCLK 频率下的 TIMEOUTA 设置示例（最大 $t_{TIMEOUT}$ = 25 ms）

f_{PCLK}	TIMEOUTA[11:0]	TIDLE	TIMEOUTEN	$t_{TIMEOUT}$
8 MHz	0x61	0	1	98 x 2048 x 125 ns = 25 ms
16 MHz	0xC3	0	1	196 x 2048 x 62.5 ns = 25 ms
32 MHz	0x186	0	1	391 x 2048 x 31.25 ns = 25 ms
48 MHz	0x249	0	1	586 x 2048 x 20.08 ns = 25 ms

将 $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 的最大持续时间配置为 8 ms：

表 28-3 不同 PCLK 频率下的 TIMEOUTB 设置示例

f_{PCLK}	TIMEOUTB[11:0]	TEXTEN	$t_{LOW:EXT}$
8 MHz	0x1F	1	32 x 2048 x 125 ns = 8 ms
16 MHz	0x3F	1	64 x 2048 x 62.5 ns = 8 ms
48 MHz	0xBB	1	188 x 2048 x 20.08 ns = 8 ms

将 t_{IDLE} 的最大持续时间配置为 50 µs

表 28-4 不同 PCLK 频率下的 TIMEOUTA 设置示例（最大 t_{IDLE} = 50 µs）

f_{PCLK}	TIMEOUTB[11:0]	TIDLE	TIMEOUTEN	T_{TIDLE}
8 MHz	0x63	1	1	100 x 4 x 125 ns = 50 us
16 MHz	0xC7	1	1	64 x 4 x 62.5 ns = 50 us
48 MHz	0x257	1	1	600 x 4 x 20.08 ns = 50 us

表 28-5 SMBus 超时规范

符号	参数	限值		单位
		最小值	最大值	
$t_{TIMEOUT}$	检测时钟低电平超时	25	35	ms
$t_{LOW:SEXT}$(1)	累积时钟低电平延长时间（从设备）	-	25	ms
$t_{LOW:MEXT}$(2)	累积时钟低电平延长时间（主设备）	-	10	ms
t_{IDLE}	SCL 和 SDA 高电平超时	-	50	us

1. $t_{LOW:SEXT}$ 是一段累积时间，即给定从设备在一条消息的最初起始到停止期间时钟信号可延展的时间。其他从设备或主设备也可能延长时钟，进而导致时钟低电平总延长时间超过 $t_{LOW:SEXT}$。因此，测量该参数时该器件应该是全速主设备寻址的唯一器件。
2. $t_{LOW:MEXT}$ 是一段累积时间，即主设备在消息的每个字节（定义为 START 到 ACK、 ACK 到 ACK 或 ACK 到STOP）内时钟信号可延展的时间。从设备或其他主设备也可能延长时钟，进而导致时钟低电平总时间超过$t_{LOW:MEXT}$（针对给定字节）。因此，测量该参数时该全速主设备只寻址一个从设备。

28.3 I²C 中断

下表列出了所有的 I²C 中断请求

表 28-6 I²C 中断请求

中断事件	事件标志	开启控制位
起始位已发送(Master)	SB	ITEVTEN
地址已发送(Master) 或 地址匹配(Slave)	ADDR	
ADD10	10 位头段已发送(主)	
已收到停止(Slave)	STOPF	

数据字节传输完成	BTF	
接收缓冲区非空	RxNE	ITEVTEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失(Master)	ARLO	
响应失败	AF	
过载/欠载	OVR	
PEC 错误	PECERR	
超时/Tlow 错误	TIMEOUT	
SMBus 提醒	SMBALERT	

28.4 I²C 寄存器

寄存器可以 half-word 或者 word 访问。

28.4.1 I²C 控制寄存器 1 (I2C_CR1)

偏移地址：0x00

复位值：0x00080000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WKUP_CNT[1:0]		WKUP_DIV[1:0]	
												RW	RW	RW	RW
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SWRST	Res.	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	WUPEN	SMBUS	PE
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:18	WKUP_CNT[1:0]	RW	2'h2	低功耗唤醒 timeout 的计数个数 2'b00：2 2'b01：8 2'b10：32 2'b11：128 注：timout 时间的计算公式为 (WKUP_DIV/FREQ)*WKUP_CNT μs 例：WKUP_DIV 选择 11 即 1024，FREQ 为 8M，CNT 选择 00 即 2，则 timeout 时间为(1024/8)* 2 = 256 μs 注：PCLK 为 72M 时，timeout 时间为 3.5μs-1817.6μs PCLK 为 4M 时，timeout 时间为 64 μs - 32768 μs
17:16	WKUP_DIV[1:0]	RW	2'h0	PCLK 的分频系数，分频后的时钟用于计数 Stop 模式唤醒时的 timeout 时间 2'b00：128 2'b01：256 2'b10：512 2'b11：1024
15	SWRST	RW	0	软件复位。

				当被置位时，I²C 处于复位状态。在复位释放前，要确保 I²C 的引脚被释放，总线是空闲状态。 0：I²C 模块不处于复位状态 1：I²C 模块处于复位状态 注：该位可以用于错误或锁住状态时重新初始化 I²C。如 BUSY 位为 1，在总线上又没有检测到停止条件时。
14	Reserved	-	-	保留
13	ALERT	RW	0	SMBus 提醒。 0：释放 SMBAlert 引脚使其变高。提醒响应地址头紧跟在 NACK 信号后面 1：驱动 SMBAlert 引脚使其变低。提醒响应地址头紧跟在 ACK 信号后面 PE=0 时，由硬件清除。
12	PEC	RW	0	数据包出错检查 软件可置位和清零该位，硬件可以在以下情况下清零该位：当传送 PEC 后，起始条件、或者停止条件、或者当 PE=0 时； 0：无 PEC 传输 1：PEC 传输（在发送或接收模式） 注：仲裁丢失时，PEC 的计算失效。
11	POS	RW	0	ACK/PEC 位置（用于数据接收），软件可置位/清零该寄存器，或 PE=0 时由硬件清零。 0：ACK 位控制当前移位寄存器内正在接收的字节的(N)ACK。PEC 位表明当前移位寄存器内的字节是 PEC 1：ACK 位控制在移位寄存器里接收的下一个字节的(N)ACK。PEC 位表明在移位寄存器里接收的下一个字节是 PEC 注：POS 位只能用在 2 字节的接收配置中，必须在接收数据之前配置。为了 NACK 第 2 个字节，必须在清除 ADDR 之后清除 ACK 位。为了检测第 2 个字节的 PEC，必须在配置了 POS 位之后，ADDR 延长事件时设置 PEC 位。
10	ACK	RW	0	应答使能。软件可置位/清零该寄存器，或 PE = 0 时由硬件清零。 0：无应答返回 1：在接收到一个字节后返回一个应答。（匹配地址或数据时）
9	STOP	RW	0	停止条件产生，软件可以置位/清零该寄存器，或者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：在当前字节传输后释放 SCL 和 SDA 线
8	START	RW	0	起始条件产生。 软件可置位/清零该寄存器，或当起始条件发出后或 PE=0 时由硬件清零。 主模式： 0：无起始条件产生 1：重复产生起始条件 从模式： 0：无起始条件产生 1：当总线空闲时，产生起始条件（并由硬件自动切换到主机模式）
7	NOSTRETCH	RW	0	禁止时钟延长（从机）。 当 ADDR 或 BTF 标志被置位时，该位用于从机禁止时钟延长，直到被软件复位。 0：允许时钟延长 1：禁止时钟延长
6	ENGC	RW	0	广播呼叫使能。 0：禁止广播呼叫。以 NACK 响应地址 00h 1：允许广播呼叫。以 ACK 响应地址 00h
5	ENPEC	RW	0	PEC 使能。 0：禁止 PEC 计算 1：开启 PEC 计算
4	ENARP	RW	0	ARP 使能。 0：禁止 ARP 1：使能 ARP 如果 SMBTYPE=0，使用 SMBus 设备的默认地址； 如果 SMBTYPE=1，使用 SMBus 的主地址。
3	SMBTYPE	RW	0	SMBus 类型。 0：SMBus 设备 1：SMBus 主机
2	WUPEN	RW	0	从模式唤醒 Stop 模式使能 0：禁止从模式唤醒 Stop 模式 1：使能从模式唤醒 Stop 模式
1	SMBUS	RW	0	SMBus 模式使能 0：I^2C 模式 1：SMBus 模式
0	PE	RW	0	I^2C 模块使能。 0：禁止 1：I^2C 使能 注：如果清除该位时通讯正在进行，在当前通讯结束后，I^2C 模块被禁用并返回空闲状态。

				由于在通讯结束后 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。

28.4.2 I2C 控制寄存器 2 (I2C_CR2)

偏移地址：0x04

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Res.	FREQ[6:0]						
			RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	LAST	RW	0	DMA 最后一次传输。 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，使在最后一次接收数据时可以产生一个 NACK。
11	DMAEN	RW	0	DMA 请求使能 0：禁止 DMA 请求 1：使能 DMA 请求
10	ITBUFEN	RW	0	缓冲器中断使能。 0：当 TxE=1 或 RxNE=1 时，不产生中断 1：当 TxE=1 或 RxNE=1 时，产生事件中断（不管 DMAEN 是何值）
9	ITEVTEN	RW	0	事件中断使能。 0：禁止 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，但没有 TxE 或 RxNE 事件 ■ 如果 ITBUFFEN=1，TxE 事件为 1 ■ 如果 ITBUFEN=1，RxNE 事件为 1
8	ITERREN	RW	0	错误中断使能。 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 ■ PECERR=1 ■ TIMEOUT=1 ■ SMBAlert=1

7	Reserved	-	-	保留
6:0	FREQ[6:0]	RW	7’h0	I²C 模块时钟频率。 必须用 APB 时钟频率的值配置该寄存器，以产生与 I²C 协议兼容的数据 setup 和 hold 时间。运行时钟为偶数时钟。 最小允许可设定的频率分别是 100k 模式下 4MHz 及以上、400k 模式下大于 8MHz、1 MHz 模式下大于 16 MHz。 0000000：禁止 0000001：禁止 ..... 0000011：禁止 0000100：4 MHz ….. …..（可配的最大频率详见 APB 的最大时钟频率）

28.4.3 I2C 自身地址寄存器 1 (I2C_OAR1)

偏移地址：0x08

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDMODE	Res.	Res.	Res.	Res.	Res.	ADD[9:8]		ADD[7:1]							ADD0
RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	ADDMODE	RW	0	寻址模式（从模式）。 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）；
14:10	Reserved	-	-	保留
9:8	ADD[9:8]	RW	2’h0	接口地址。 7 位地址模式该寄存器无关。 10 位地址模式位地址的 9~8 位。
7:1	ADD[7:1]	RW	7’h0	接口地址的 7~1 位。
0	ADD0	RW	0	接口地址。 7 位地址模式时该寄存器无效。 10 位地址模式时为地址的 0 位。

28.4.4 I2C 自身地址寄存器 2（I2C_OAR2）

偏移地址：0x0C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	OA2MSK[2:0]			ADD2[7:1]							ENDUAL
					RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留
10:8	OA2MSK[2:0]	RW	3’h0	设备自身地址 2 屏蔽位

				000：无屏蔽 001：OA2[1] 被屏蔽，为无关位。仅比较 OA2[7:2]。 010：OA2[2:1] 被屏蔽，为无关位。仅比较 OA2[7:3]。 011：OA2[3:1] 被屏蔽，为无关位。仅比较 OA2[7:4]。 100：OA2[4:1] 被屏蔽，为无关位。仅比较 OA2[7:5]。 101：OA2[5:1] 被屏蔽，为无关位。仅比较 OA2[7:6]。 110：OA2[6:1] 被屏蔽，为无关位。仅比较 OA2[7]。 111：OA2[7:1] 被屏蔽，为无关位。不进行比较，对接收到的全部 7 位地址（保留位除外）应答。
7:1	ADD2[7:1]	RW	7’h0	接口地址的 7~1 位。 双地址模式下地址的 7~1 位。
0	ENDUAL	RW	0	双地址模式使能位。 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。

28.4.5 I2C 数据寄存器(I2C_DR)

偏移地址：0x10

复位值：0x00000000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	DR[7:0]	RW	8’h0	8 位数据寄存器，芯片内部实际是两个独立的缓存共用一个地址，分别用于存放接收到的数据（RX_DR）、放置要发送到总线的数据（TX_DR）。 发送器模式： 当写一个字节至 DR 寄存器时（实际写到 TX_DR），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（实际是 RX_DR）（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接收。 注： 1) 在从机模式下，地址不会被拷贝进数据寄存器 2) 硬件不处理写冲突 (如果 TxE=0，仍能写入数据寄存器) 3) 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，因此不能被读到

28.4.6 I²C 状态寄存器 1 (I2C_SR1)

偏移地址：0x14

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMBALERT	TIMEOUT	Res.	PECERR	OVR	AF	ARLO	BERR	TxE	RxNE	Res.	STOPF	ADD10	BTF	ADDR	SB
RC_W0	RC_W0		RC_W0	RC_W0	RC_W0	RC_W0	RC_W0	R	R		R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	SMBALERT	RC_W0	0	SMBus 提醒状态。 主机模式： 0：无 SMBus 提醒 1：在引脚产生 SMBAlert 提醒事件 从机模式： 0：没有 SMBAlert 响应地址头序列 1：收到 SMBAlert 响应地址头序列直到 SMBAlert 变低 该位由软件写 0 或 PE=0 时由硬件清除。
14	TIMEOUT	RC_W0	0	超时或 T_{low} 错误。 0：无超时错误； 1：SCL 为低的持续时间已达到 25ms（超时）；或者主机低电平累计时钟扩展时间超过 10ms（$T_{low:\ next}$）；或从设备低电平累积时钟扩展时间超过 25ms（$T_{low:sext}$）；或者 T_{idle} 的时间超时；或者 I²C 模式从 STOP 模式唤醒时可通过设置 I2C_CR1 设置 timeout 时间。 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，或当 PE=0 时由硬件清除。 注: 该功能仅在 SMBUS 模式和 I²C 模式下从 STOP 模式唤醒过程下有作用。
13	Reserved	-	-	保留
12	PECERR	RC_W0	0	在接收时发生 PEC 错误。 0：无 PEC 错误，接收到 PEC 后返回 ACK（如果 ACK=1） 1：有 PEC 错误，接收到 PEC 后返回 NACK（不管 ACK 为何值） 该位由将软件写 0 清除，或当 PE=0 时由硬件清除。
11	OVR	RC_W0	0	过载/欠载标志。 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； 该位由软件写 0 清除，或当 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。
10	AF	RC_W0	0	应答失败标志。 0：没有应答失败 1：应答失败 当没有返回应答时，硬件将置位该寄存器。 该位由软件写 0 清除，或当 PE=0 时由硬件清除。
9	ARLO	RC_W0	0	仲裁丢失（主模式）。 0：没有检测到仲裁丢失 1：检测到仲裁丢失 当接口失去对总线的控制给另一个主机时，硬件将置位该寄存器。 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，I^2C 接口自动切换回从模式（MSL=0）。 注：在 SMBUS 模式下，在从模式下的仲裁仅发生在数据解读，或应答传输区间（不包括地址的应答）。
8	BERR	RC_W0	0	总线出错标志。 0：无起始或者停止条件出错； 1：起始或者停止条件出错。 当接口检测到错误的起始或者停止条件，硬件将该位置 1。 该位由软件写 0 清除，或者在 PE=0 时由硬件清除。
7	TxE	R	0	数据寄存器为空（发送时）标志。 0：数据寄存器非空； 1：数据寄存器为空。 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，或下一个要发送的字节是 PEC（PEC=1），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，因为此时数据寄存器为空。
6	RxNE	R	0	数据寄存器非空（接收时）标志。 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，置位该寄存器。在接收地址阶段，该寄存器不置位。 软件对数据寄存器的读写操作会清除该寄存器，或当 PE=0 时由硬件清除。 注：当设置了 BTF 时，读取数据不能清除 RxNE 位，因为此时数据寄存器仍为满。
5	Reserved	-	-	保留
4	STOPF	R	0	停止条件检测位（从模式）。 0：没有检测到停止位

				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。 将软件读取 I2C_SR1 寄存器后，对 I2C_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 位不会置位。
3	ADD10	R	0	10 位地址头序列已发送（主模式）。 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，当主设备将第一个字节发送出去时，硬件将该位置 1. 软件读取 I2C_SR1 寄存器后，对 I2C_DR 寄存器的写操作将清除该位，或当 PE=0 时，硬件清除该位。 主：收到 NACK 后，该寄存器不被置位。
2	BTF	R	0	字节传输发送结束标志位。 0：字节传输发送未完成 1：字节传输发送成功结束 当 NOSTRETCH=0 时，在下列情况下硬件将置位该寄存器（从机模式，NOSTRETCH=0 时；主机模式，与 NOSTRETCH 无关）： — 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）。 — 发送时，当一个新数据应该被发送，且数据寄存器还未被写入新的数据（TxE=1）。 软件读取 I2C_SR1 寄存器后，对数据寄存器的读或写操作将清除该位 或发送一个起始或停止条件后，或当 PE=0 时，由硬件清除。 注： 在收到一个 NACK 后，BTF 位不会被置位。 如果下一个要传输的字节是 PEC（I2C_SR1.TRA=1,I2C_CR1.PEC=1）,BTF 位不会被置位。
1	ADDR	R	0	地址已被发送（主模式）/地址匹配（从模式）。 读 I2C_SR2 寄存器后，再读 I2C_SR1 寄存器将清除该位；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。 **地址匹配（从机）：** 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 寄存器或广播呼叫地址，或 SMBus 设备默认地址、或 SMBus 主机识别出 SMBus 提醒地址匹配时，硬件将置位该位。 **注:** 在从机模式下，推荐进行完整的清零序列，即在 ADDR 被置位后，先读 SR1 寄存器，再读 SR2 寄存器。 **地址已发送（主机）：**

				0：地址发送没有结束 1：地址发送结束 ● 10 位地址时，当收到地址的第二个字节的 ACK 后置位 — 7 位地址时，当收到 ACK byte 后置位。 **注:** 在收到 NACK 后，该寄存器不会被置位。
0	SB	R	0	起始位标志（主模式）。 0：未发送起始条件； 1：起始条件已发送； —当发送起始条件时，置位该寄存器。 —软件读取 I2C_SR1 寄存器后，对数据寄存器的读或写操作将清除该位；或当 PE=0 时，由硬件清除。

28.4.7 I²C 状态寄存器 2 (I2C_SR2)

偏移地址：0x18

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PEC[7:0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Res.	TRA	BUSY	MSL
R	R	R	R	R	R	R	R	R	R	R	R		R	R	R

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:8	PEC[7:0]	R	8’h0	数据包出错检测寄存器。 当 ENPEC=1 时，该寄存器存放内部的 PEC 的值。
7	DUALF	R	0	双地址标志（从模式）. 0：接收到的地址与 OAR1 匹配 1：接收到的地址与 OAR2 匹配 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
6	SMBHOST	R	0	接收到 SMBus 主机头序列（从模式）标志 0：未收到 SMBus 主机的地址 1：当 SMBTYPE=1 且 ENARP=1 时，收到 SMBus 主机地址 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
5	SMBDEFAULT	R	0	SMBus 从设备默认地址（从模式） 0：未收到 SMBus 设备的默认地址 1：当 ENARP=1 时，收到 SMBus 设备的默认地址 当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
4	GENCALL	R	0	广播呼叫地址（从模式） 0：未收到广播呼叫地址 1：当 ENGC=1 时，收到广播呼叫的地址

				当产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件清除该寄存器。
3	Reserved	-	-	保留
2	TRA	R	0	发送/接收标志 0：接收到数据 1：数据已发送 在整个地址传输阶段的结尾，该寄存器根据地址字节的 R/W 位来设定。 当检测到停止条件（STOPF=1），或者重复的起始条件、或者总线仲裁丢失（ARLO=1），或当 PE=0 时，硬件清除该寄存器。
1	BUSY	R	0	总线忙标志 0：在总线上无数据通讯 1：在总线上正在进行数据通讯 当检测到 SDA 或 SCL 为低电平时，硬件置位。 当检测到一个停止条件时，硬件清零。 该寄存器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。
0	MSL	R	0	主从模式标志 0：从机模式 1：主机模式 —当接口处于主模式（SB=1）时，硬件置位； —当总线上检测到一个停止条件（STOPF=1）、仲裁丢失（ARLO=1）、或当 PE=0 时，硬件清零。

28.4.8 I²C 时钟控制寄存器(I2C_CCR)

偏移地址：0x1c

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
F/S	DUTY	F+	Res.	CCR[11:0]											
RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15	F/S	RW	0	I²C 主模式选择。 0：标准模式 1：快速模式
14	DUTY	RW	0	快速模式时的占空比。 0：快速模式下：$T_{low}/T_{high}=2$ 1：快速模式下：$T_{low}/T_{high}=16/9$
13	F+	RW	0	I²C 快速增强模式选择。 0：标准模式或快速模式，具体选择哪种由 bit15 决定； 1：快速增强模式；
12	Reserved	-	-	保留

11:0	CCR[11:0]	RW	12'h0	快速/标准模式下的时钟控制分频系数（主模式）。 该分频系数用于设置主模式下的 SCL 时钟。 ■ 标准模式或者 SMBus 模式： T_{high}=CCR x Tpclk T_{low} =CCR x Tpclk ■ 快速模式： — DUTY=0: T_{high}=CCR x Tpclk T_{low} =2 x CCR x Tpclk — DUTY=1(为达到 400 kHz): T_{high}=9 x CCR x Tpclk T_{low} =16 x CCR x Tpclk ■ 快速增强版模式： — DUTY=0: T_{high} =3 x CCR x Tpclk T_{low} =5 x CCR x Tpclk — DUTY=1: T_{high} =2 x CCR x Tpclk T_{low} =3x CCR x Tpclk 注： 1. 允许设定的最小值为 0x04，在快速模式下允许的最小值为 0x01 2. $T_{high}=t_{r(SCL)}+t_{w(SCLH)}$ 3. $T_{low}=t_{r(SCL)}+t_{w(SCLL)}$ 4. 这些延时没有过滤器 5. 只有当 PE=0 时才能配置该寄存器

28.4.9 I²C 上升时间寄存器 (I2C_TRISE)

偏移地址：0x20

复位值：0x00000082

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

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	THOLDDATA_SEL	RW	0	数据保持时间选择 0：默认硬件计算数据保持时间 1：通过 THLDDATA 配置数据保持时间
11:7	THOLDDATA[4:0]	RW	5'h1	在快速/标准/快速增强模式下的最大数据保持时间(发送模式) 这些位是用于数据发送模式下，保证数据保持时间的最小时间。 例如：标准模式允许的最大 SDA 下降时间为 300ns。如果 I2C_CR2.FREQ[6:0]的值等于 0x08，T_{pclk} =125ns，则 TRISE 中配置为 0x03 （300ns/125ns = 2.4 +1 =3.4） 如果结果不为整数，则将整数部分写入 THOLDDATA，以确保配置。

6:0	TRISE[6:0]	RW	7'h2	在快速/标准模式下的最大上升时间（主模式）。 这些位应该提供在主机模式下， SCL 反馈回路的最大持续时间。这样做的目的是无论 SCL 上升沿持续时间多少，SCL 都能保持一个稳定的频率。 这些位必须设置为 I²C 总线规范里给出的最大的 SCL 上升时间，增长步幅为 1。 例如：标准模式中最大允许 SCL 上升时间为 1000ns。如果 I2C_CR2.FREQ[5:0]的值等于 0x08，T_{pclk}=125ns，则 TRISE 中配置为 0x09（1000ns/125ns = 8 + 1 = 9）。 如果结果不为整数，则将整数部分写入 TRISE，以确保 t_{HIGH} 参数。 注：当 PE=0 时才能设置该寄存器。

28.4.10 I²C 超时寄存器(I2C_TIMEOUTR)

偏移地址：0x24

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEXTEN	Res.	Res.	Res.	TIMEOUTB[11:0]											
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**
TIMOUTEN	Res.	Res.	TIDLE	TIMEOUTA[11:0]											
RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	TEXTEN	RW	0	时钟信号延展超时使能 0：禁止时钟信号延展超时检测。 1：使能时钟信号延展超时检测 当 I²C 接口执行 SCL 延展的累积时间超过 $t_{LOW:EXT}$ 时，将检测到超时错误（TIMEOUT = 1)。
30:28	Reserved	-	-	保留
27:16	TIMEOUTB[11:0]	RW	12'h0	总线超时 B 该字段用于配置累积时钟延展超时： 在主模式下，将检测主设备的累积时钟低电平延展时间($t_{LOW:MEXT}$) 在从模式下，将检测从设备的累积时钟低电平延展时间($t_{LOW:SEXT}$) $t_{LOW:EXT}$ = (TIMEOUTB+1) x 2048 x t_{PCLK} 注： 仅可在 TEXTEN=0 时写入这些位
15	TIMEOUTEN	RW	0	时钟超时使能 0：禁止 SCL 超时检测。 1：使能 SCL 超时检测：当 SCL 的低电平时间超过 $t_{TIMEOUT}$ (TIDLE=0)，或 SCL 的高电平 时间超过 t_{IDLE} (TIDLE=1) 时，将检测到超时错误(TIMEOUT=1)。
14:13	Reserved	-	-	保留
12	TIDLE	RW	0	空闲时钟超时检测 0：TIMEOUTA 用于检测 SCL 低电平超时 1：TIMEOUTA 用于检测 SCL 和 SDA 高电平超时（总线空闲条件）

				注： 仅可在 TIMOUTEN=0 时写入该位。
11: 0	TIMEOUTA[11:0]	RW	12’h0	总线超时 A 该字段用于配置： – SCL 低电平超时条件 $t_{TIMEOUT}$（当 TIDLE=0 时） $t_{TIMEOUT}$= (TIMEOUTA+1) x 2048 x t_{PCLK} –总线空闲条件，即 SCL 和 SDA 高电平（当 TIDLE=1 时） t_{IDLE}= (TIMEOUTA+1) x 4 x t_{PCLK} 注： 仅可在 TIMOUTEN=0 时写入这些位。

29. 通用同步收发器(USART)

29.1 USART 简介

通用同步异步收发器（USART）能够灵活地与外部设备进行全双工数据交换，满足外部设备对工业标准 NRZ 异步串行数据格式的要求。USART 利用分数波特率发生器提供宽范围的波特率选择。

USART 支持同步单向通信和半双工单线通信，也支持 LIN(局部互联网)，智能卡协议和 IrDA(红外数据组织)规范，以及调制解调器（CTS/RTS）操作。它还允许多处理器通信。

USART 也可以使用多缓冲器配置的 DMA 方式，实现高速数据通信。

29.2 USART 主要特征

- 全双工异步通信
- 可配置 16 或 8 的过采样方法，以实现速度和时钟容忍度之间的最佳选择
- 两个内置的 FIFO 用于发送和接收
- 两个 FIFO 都可以通过软件来使能和提供状态位
- 双时钟域：除了 PCLK 以外，还存在的一个 USART_CLK 时钟
- 自动波特率检测功能
- 可配置的数据字长度（7 位，8 位或者 9 位）
- 可配置的数据传输方向：MSB，LSB
- 可配置的停止位—支持 0.5，1，1.5 或 2 个停止位
- 提供同步模式的输入输出功能
- 使用 DMA（直接存储器访问）实现可配置的多缓冲区通信
- 单线半双工通信
- 单独的发送器和接收器使能位
- 可配置的 TX/RX 引脚信号
- 支持的硬件流控：modem 和 RS485
- 通信控制和错误检测信号
- 奇偶校验控制
  - 发送校验位
  - 对接收数据进行校验
- 多处理器通信。如果地址不匹配，则进入静默模式
  - 静默模式中唤醒（通过空闲总线检测或地址标志检测）
  - 唤醒接收器的方式：地址位（MSB,第 9 位），总线空闲
- Lin 主机发送同步断开帧的能力以及 Lin 从机检测断开帧的能力
  - 当 USART 硬件配置成 LIN 时，能够生成 13 位断开帧并检测断开帧
- IRDA SIR 编码器解码器
- 智能卡模拟功能
  - 智能卡接口支持 ISO7816-3 标准里定义的异步智能卡协议
  - 智能卡用到 0.5 和 1.5 个停止位
- MODBUS 通信
  - TIMOUT 功能
  - CR/LF 字符识别功能

29.3 USART 功能描述

29.3.1 USART 功能框图

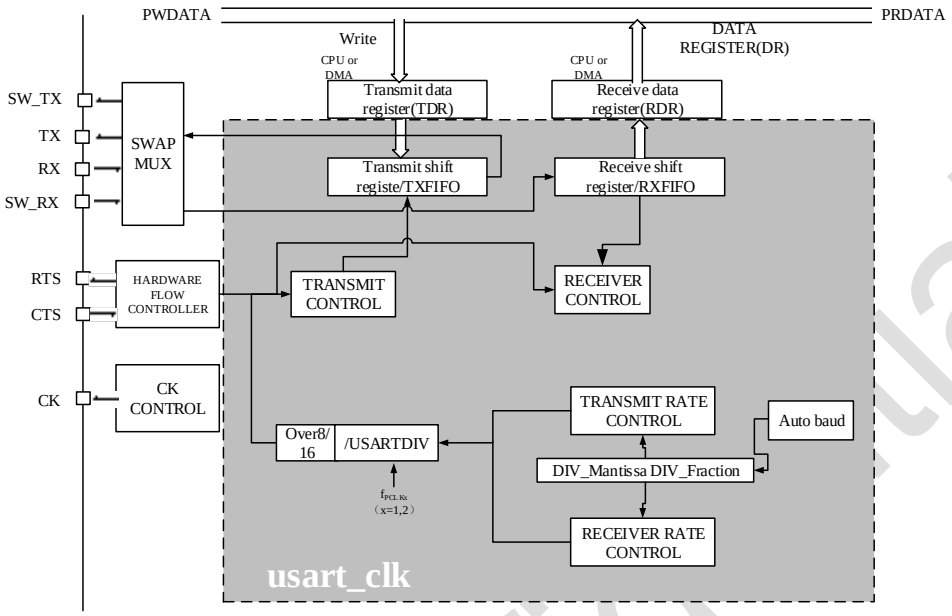


图 29-1 USART 框图

29.3.2 USART 输入/输出引脚说明

USART 接口通过三个引脚与其他设备连接在一起。任何 USART 双向通信至少需要两个脚：接收数据输入(RX)和发送数据输出(TX)。

RX：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：发送数据串行输出。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，并且不发送数据时，TX 引脚处于高电平。在单线和智能卡模式里，此 I/O 口被同时用于数据的发送和接收。

在 RS232 硬件流控制模式中需要以下引脚：

CTS（clear to send）：清除发送，若是高电平，在当前数据传输结束时阻断下一次的数据发送。

RTS（request to send)：发送请求，若是低电平，表明 USART 准备好接收数据。

在 RS485 硬件控制模式下需要以下引脚：

DE(Driver Enable)：驱动使能，这个信号用控制外部接收者的传输

在同步主模式和智能卡模式下需要以下引脚：

CK：该引脚在同步主模式和智能卡模式下用作时钟输出。

在同步主模式下，该引脚用于输出发送器数据时钟，以便按照 SPI 主器件模式进行同步发送(起始位和结束位上无时钟脉冲，可通过软件向最后一个数据位发送时钟脉冲)。同时， RX 引脚可同步接收数据。该机制可用于控制带移位寄存器的外设(例如 LCD 驱动器)。时钟相位和极性可通过软件编程。

在智能卡模式下， CK 输出向智能卡提供时钟。

USART 输入/输出引脚的列表如下表所示。

表 29-1 USART 输入/输出引脚

引脚名称	信号类型	说明
USART_RX	输入	串行数据接收输入
USART_TX	输出	发送数据输出引脚
USART_CTS	输入	清除以发送
USART_RTS	输出	请求以发送
USART_DE	输出	驱动器使能
USART_CK	输出	同步主模式和智能卡模式下的时钟输出

29.3.3 USART 时钟

USART 有两个完全独立的时钟域：

- usart_pclk 时钟域

usart_pclk 时钟信号为外设总线接口提供时钟。需要访问 USART 寄存器时，该信号必须有效。

- usart_clk 内核时钟域

usart_clk 是 USART 时钟源。它独立于 usart_pclk，由 RCC 提供。

注意：访问 USART 寄存器时，usart_clk 也是需要使能的。

不支持双时钟域功能时，usart_clk 时钟与 usart_pclk 时钟相同。

usart_pclk 和 usartr_clk 之间无任何约束：usart_clk 可快于/慢于 usart_pclk。唯一的限制就是软件以足够快的速度管理通信的能力。

29.3.4 USART 字符说明

可通过对 USART_CR1 寄存器中的 M 位(M0：位 12，M1：14 位)进行编程来将字长设置为 7 位、8 位或 9 位。

- 7 位字符长度：M[1:0] = 10
- 8 位字符长度：M[1:0] = 00
- 9 位字符长度：M[1:0] = 01

在默认情况下，信号(TX 或 RX)在起始位工作期间处于低电平状态，在停止位工作期间处于高电平。

通过极性配置控制，可以单独针对每个信号对这些值取反。

IDLE character(空闲帧) 可理解为整个帧周期内电平均为“1”(停止位的电平也是“1”)，该字符后是下一个数据帧的起始位。

Break character(断开帧) 可理解为在一个帧周期内接收到的电平均为“0”。发送器在断开帧的末尾插入 1 或 2 个停止位(逻辑“1”位)以确认起始位。

发送和接收操作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。

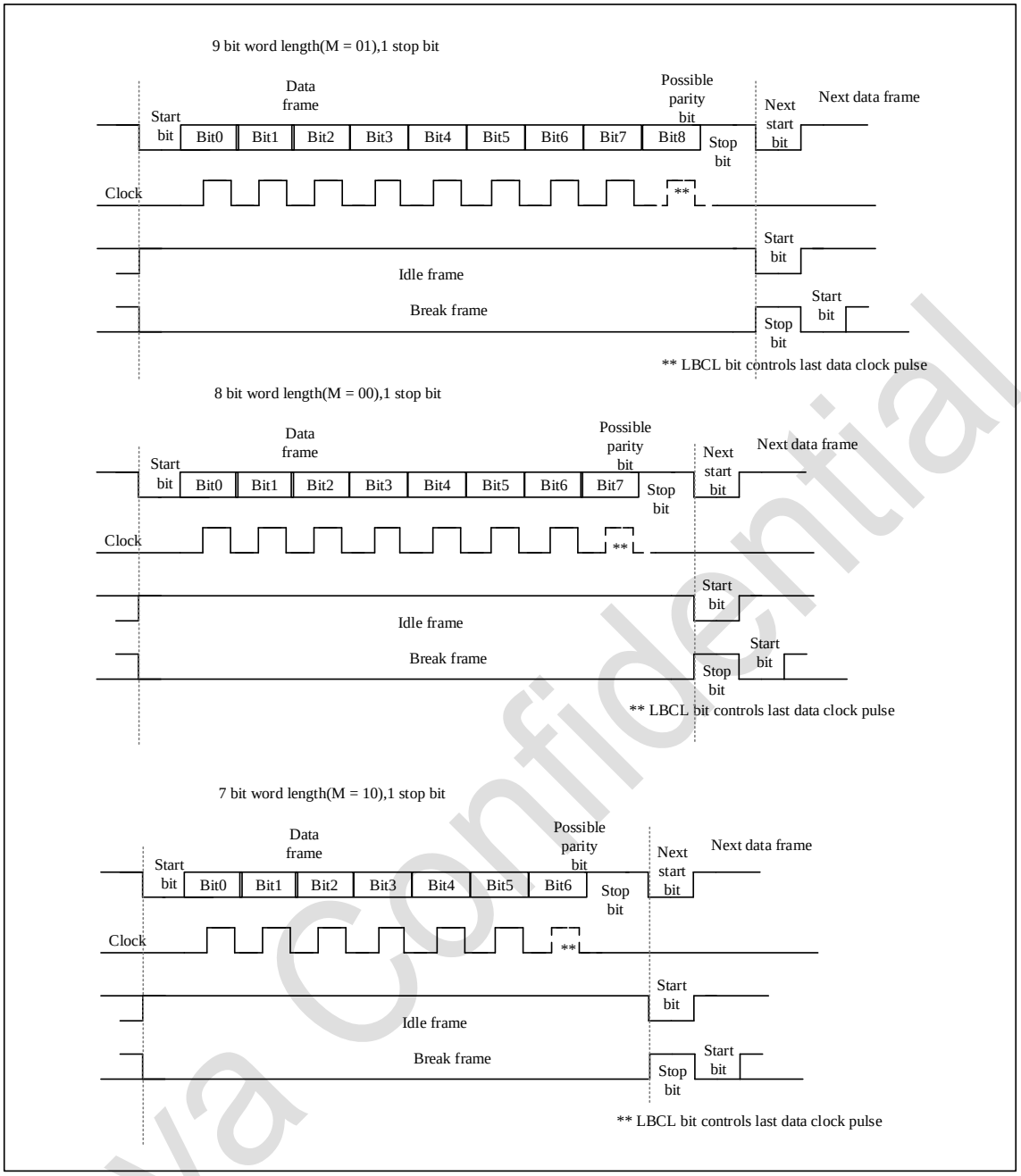


图 29-2 字长编程

29.3.5 USART FIFO 和阈值

USART 可工作在 FIFO 模式下。

USART 具有一个发送 FIFO (TXFIFO) 和一个接收 FIFO (RXFIFO)。可通过将 USART_CR1 寄存器中的 FIFOEN(位 31)置 1 使能 FIFO 模式。仅 UART、SPI 和智能卡模式下支持该模式。

最大数据字长度为 9 位，因此 TXFIFO 为 9 位宽。不过， RXFIFO 的默认宽度为 12 位。这是因为接收器不仅在 FIFO 中存储数据，而且还存储与每个字符相关的错误标志(奇偶校验错误、噪声错误和帧错误标志)。

注意：接收的数据与相应的标志一起存储在 RXFIFO 中，但读取 USART_DR 时仅读取数据。

状态标志位于 USART_SR 寄存器中。

可以配置触发 TX 和 RX 中断的 TXFIFO 和 RXFIFO 阈值。这些阈值通过 USART_CR1 控制寄存器中的 RXFTCFG 和 TXFTCFG 位进行编程。

在这种情况下：

- 当 RXFIFO 中接收到的数据量达到 RXFTCFG 中编程的阈值时， USART_SR 寄存器中的 RXFT 标志置 1，如果使能了 RXFT 的中断使能(RXFTIE)，则会产生相应的中断。
- 当 TXFIFO 中的空位置数达到在 TXFTCFG 中编程的阈值时， USART_SR 寄存器中的 TXFT 标志置 1，如果使能了 TXFT 的中断使能(TXFTIE)，则会产生相应的中断。
- 在多处理器通信模式下:
- 禁用 FIFO 模式时，RXNE 标志在每个字节接收后被设置。当 DMA 读取 USART_DR 寄存器时，RXNE 标志被清除。
- 使能 FIFO 模式时，当 RXFIFO 不为空时，RXFNE 标志置位。在每个 DMA 请求之后，从 RXFIFO 接收 1 个数据。当 RXFIFO 不为空时触发 DMA 请求(有数据要从 RXFIFO 读取)。

在单缓冲区模式下:

- 禁用 FIFO 模式时，清除 RXNE 标志是通过从 USART_DR 寄存器执行软件读取来完成的。RXNE 标志也可以通过在 USART_CR1 将 RXFRQ 位置为“1”来清除。RXNE 标志必须在接收下一个字符结束之前清除，以避免溢出错误。
- 使能 FIFO 模式时，当 RXFIFO 不为空时，设置 RXFNE。在每次从 USART_DR 读取操作之后，都会从 RXFIFO 检索一个数据。当 RXFIFO 为空时，RXFNE 标志被清除。当 RXFIFO 已满时，必须在接收下一个字符结束之前读取 RXFIFO 中的第一个数据，以避免溢出错误。如果设置了 RXFNEIE 位，RXFNE 标志会产生中断。

29.3.6 USART 数据发送

发送器根据 M 位的状态发送 7 位、8 位或 9 位的数据。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，相应的时钟脉冲在 CK 脚上输出。

29.3.6.1 字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式下，USART_DR 寄存器的缓冲区(TDR)位于内部总线和发送移位寄存器之间。

使能 FIFO 模式时，写入到发送数据寄存器(USART_DR)中的数据会在 TXFIFO 中排队。

每个字符之前都有一个低电平的起始位；之后跟着的停止位，其数目可配置。

USART 支持多种停止位的配置：0.5、1、1.5 或者 2 个停止位。

注意：

1. 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。
2. TE 位被激活后将发送一个空闲帧。

29.3.6.2 配置停止位

随每个字符发送的停止位的位数可以通过 USART_CR2 的位 13、12 进行编程。

- 1 个停止位：停止位位数的默认值。
- 2 个停止位：可用于常规 USART 模式、单线模式以及调制解调器模式。
- 0.5 个停止位：在智能卡模式下接收数据时使用。
- 1.5 个停止位：在智能卡模式下发送和接收数据时使用。

空闲帧包括了停止位。

断开帧是 10 位低电平(M[1:0] = 00 时)，或者 11 位低电平(M[1:0] = 01 时)，或者 9 位低电平(M[1:0] = 10 时)；后面都是已配置数量的停止位。不可能传输更长的断开帧(长度大于 9/10/11 位低电平)。

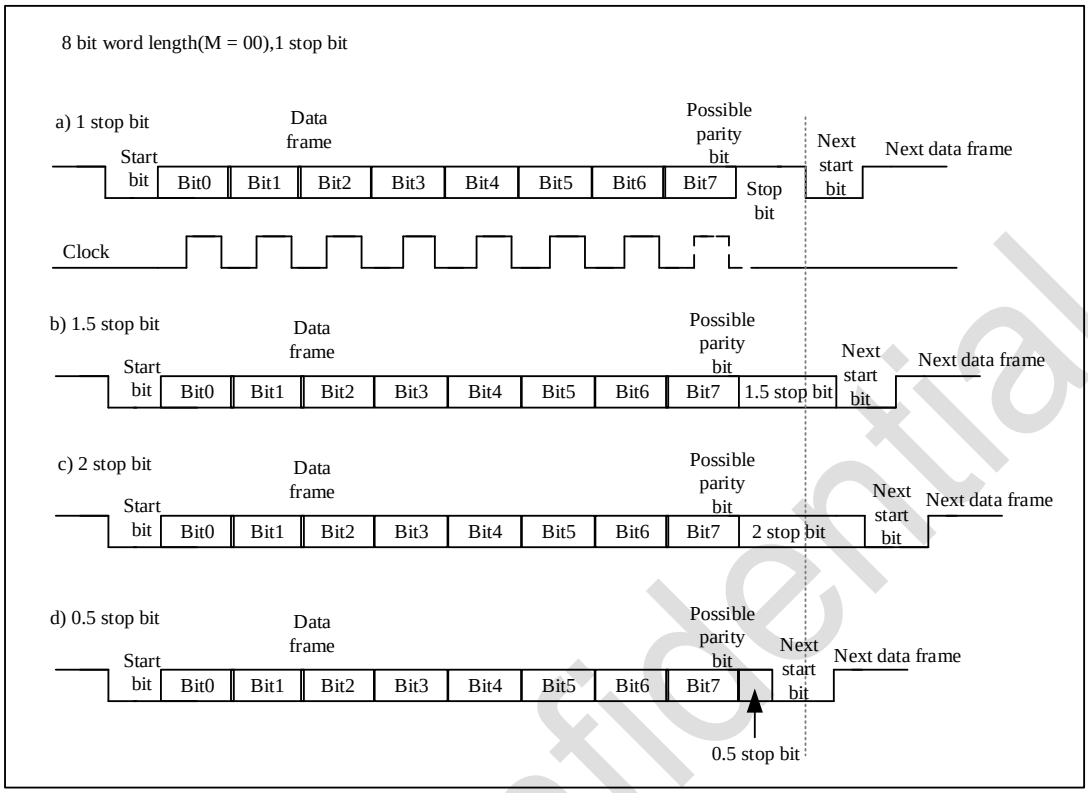


图 29-3 配置停止位

配置步骤：

1. 编程 USART_CR1 的 M 位来定义字长。
2. 在 USART_CR2 中编程停止位的位数。
3. 如果采用多缓冲器通信，配置 USART_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。
4. 利用 USART_BRR 寄存器选择要求的波特率。
5. 通过置位 USART_CR1 寄存器的 UE 位来激活 USART。
6. 设置 USART_CR1 中的 TE 位，发送一个空闲帧作为第一次数据发送。
7. 把要发送的数据写进 USART_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。
   1) 禁止 FIFO 模式时，向 USART_DR 写入数据会将 TXE 标志清零。
   2) 使能 FIFO 模式时，向 USART_DR 写入数据会为 TXFIFO 中增添一个数据。当 TXFNF 标志置 1 时，对 USART_DR 寄存器的写操作可以执行。该标志会保持置 1，直到 TXFIFO 已满。
8. 在 USART_DR 寄存器中写入最后一个数据字后，要等待 TC=1。
   1) 禁止 FIFO 模式时，这表示最后一个帧的发送已经完成。
   2) 使能 FIFO 模式时，这表示 TXFIFO 和移位寄存器均为空。
   3) 当需要关闭 USART 或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次传输。

29.3.6.3 单字节通讯

- 禁止 FIFO 模式时

对发送数据寄存器进行写操作始终会清零 TXE 位。TXE 标志由硬件置 1。他表示：

- 数据已从 USART_DR 寄存器传到移位寄存器中且数据发送已开始；
- USART_DR 寄存器为空；
- USART_DR 寄存器中可写入下一个数据，而且不会覆盖前一个数据。

TXEIE 位置 1 时，该标志位会发生中断。

如果此时 USART 正在发送数据，对 USART_DR 寄存器的写操作把数据存进内部的 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 USART 没有在发送数据，处于空闲状态，对 USART_DR 寄存器的写操作直接把数据放进移位寄存器，数据传输开始时，TXE 位立即被置起。

- 使能 FIFO 模式时，TXFNF(TXFIFO 未满)标志由硬件置 1，以指示：
  - TXFIFO 未满；
  - USART_DR 寄存器为空；
  - USART_DR 寄存器中可写入一个数据，而不会覆盖前一个数据。发送时，写 USART_DR 寄存器，会将数据存储在 TXFIFO。该数据随后会在当前发送结束时从 TXFIFO 复制到移位寄存器中。

TXFIFO 未满时，即使在对 USART_DR 寄存器执行完写操作后，TXFNF 标志也保持为 1。该标志在 TXFIFO 已满时清零。TXFNFIE 位置 1 时该标志位会生成中断。

或者当达到 TXFIFO 阈值时，会生成中断并将数据写入 FIFO。这种情况下，CPU 可写入由编程的触发阈值定义的数据块。

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 USART_CR1 寄存器中的 TCIE 位被置起时，则会产生中断。

在 USART_DR 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式(详见下图)之前，必须先等待 TC=1。

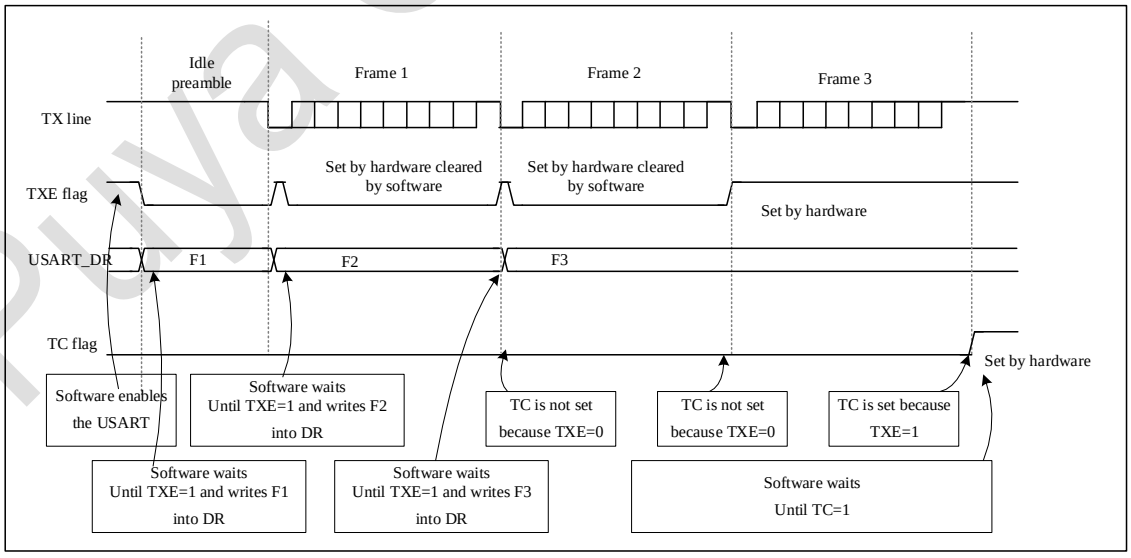


图 29-4 传输期间 TC/TXE 的行为

注意：使能 FIFO 管理时，TXFNF 标志将用于数据发送。

29.3.6.4 断开字符

设置 SBK 可发送一个断开字符。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 线上发送一个断开字符。断开字符发送完成时(发送断开字符后的停止位期间)SBK 被硬件复位。

USART 在最后一个断开帧的结束处插入一逻辑'1'，以保证能识别下一帧的起始位。

如果 SBK 位置 1，则在当前发送结束时，会发送一个断开字符。

使能 FIFO 模式时，即使 TXFIFO 已满，发送断开字符的优先级也仍高于发送数据的优先级。

注意：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。

29.3.6.5 空闲字符

置位 TE 将使得 USART 在第一个数据帧前发送一空闲帧。

29.3.7 USART 数据接收

USART 可以根据 USART_CR1 的 M 位接收 7 位、8 位或 9 位的数据。

29.3.7.1 起始位检测

在 USART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。该序列为： 1 1 1 0 X 0 X 0 X 0 0 0 0

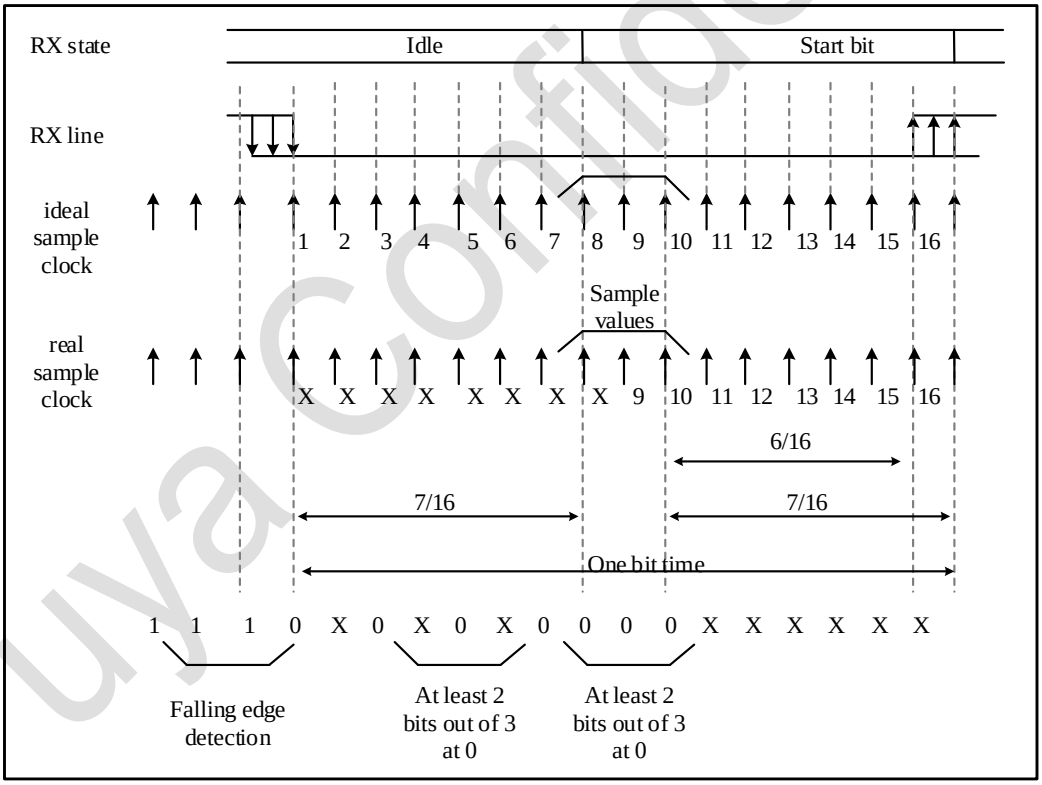


图 29-5 起始位的检测

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。

29.3.7.2 字符接收

在 USART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里， USART_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. 编程 USART_CR1 的 M 位定义字长
2. 在 USART_CR2 中编写停止位的个数
3. 如果需多缓冲器通信，选择 USART_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。
4. 利用波特率寄存器 USART_BRR 选择希望的波特率。
5. 将 USART_CR1 寄存器的 UE 置 1 来激活 USART。
6. 设置 USART_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：

- 如果已禁止 FIFO 模式,则 RXNE 位置 1。它表明移位寄存器的内容被转移到内部 RDR 寄存器中。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)。
- 如果已使能 FIFO 模式，则 RXFNE 位置 1，这表明 RXFIFO 非空。读取 USART_DR 寄存器会返回输入到 RXFIFO 中的最早数据。接收到数据时，数据以及相应的错误位将一起被存储在 RXFIFO 中。
- 如果 RXNEIE(使能 FIFO 模式时为 RXFNEIE)位被设置，则产生中断。
- 在接收期间如果检测到帧错误、噪音错误、奇偶校验错误或上溢错误，错误标志将被置 1。
- 在多缓冲器通信模式下：
  - 如果禁止 FIFO 模式，则 RXNE 标志会在每次接收到字节后置 1。该标志在 DMA 读取数据寄存器时被清零。
  - 如果使能 FIFO 模式，则 RXFNE 标志在 RXFIFO 非空时置 1.每次收到 DMA 请求后，都会从 RXFIFO 取回数据。当 RXFIFO 非空时(即当存在要从 RXFIFO 中读取的数据时)，会触发 DMA 请求。
- 在单缓冲器模式下：
  - 如果禁止 FIFO 模式，则通过软件对 USART_DR 寄存器进行读操作来将 RXNE 标志位清零。RXNE 位必须在下一字符接收结束前被清零，以避免上溢错误。
  - 如果使能 FIFO 模式，则 RXFNE 在 RXFIFO 非空时置 1。每次对 USART_DR 执行完读操作后，都会从 RXFIFO 中取回数据。RXFIFO 为空时，RXFNE 标志将清零。RXFIFO 已满时，必须在结束接收下一个字符前读取 RXFIFO 中的第一个条目，以避免发生上溢错误。当 RXFNEIE 位置 1 时，RXFNE 标志会生成中断。或者，当达到 RXFIFO 阈值时，会生成中断并从 RXFIFO 中读取数据。在这种情况下，CPU 可读取由编程的阈值定义的数据块。

注意： 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。

29.3.7.3 断开字符

当接收到一个断开字符时，USART 像处理帧错误一样处理它。

29.3.7.4 空闲字符

当一空闲帧被检测到时，其处理步骤和接收到普通数据帧一样，但如果 IDLEIE 位被设置将产生一个中断。

29.3.7.5 上溢错误

- 禁止 FIFO 模式

如果 RXNE 未复位时接收到字符，则会发生上溢错误。

RXNE 位清零前，数据无法从移位寄存器传送到内部的 RDR 寄存器。每接收到一个字节后，RXNE 标志都将置 1。

当 RXNE 标志位是 1 时，如果在接收到下一个数据或尚未处理上一个 DMA 请求，则也会发生上溢错误。

发生上溢错误时：

1. ORE 位被置位。
2. RDR 内容将不会丢失。读 USART_DR 寄存器仍能得到先前的数据。
3. 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。
4. 如果 RXNEIE 位置 1 或 EIE 位置 1，则会生成中断。
5. 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 ORE 位

- 使能 FIFO 模式

移位寄存器准备好传送并且接收 FIFO 已满时，会发生上溢错误

在 RXFIFO 中出现一个空闲位置之前，数据无法从移位寄存器传送到 USART_DR 寄存器。当 RXFIFO 非空时， RXFNE 标志置 1。如果 RXFIFO 已满且移位寄存器已准备好传送，则也会发生上溢错误。

发生上溢错误时：

1. ORE 位置 1。
2. RXFIFO 中的第一个条目不会丢失。通过读取 USART_DR 寄存器可获得此条目。
3. 移位寄存器会被覆盖。之后，上溢期间接收到的任何数据都将丢失。
4. 如果 RXFNEIE 位置 1 或 EIE 位置 1，则会生成中断。

注意：当 ORE 位置位时，表明至少有 1 个数据已经丢失。

禁止 FIFO 模式时，有以下两种可能

- 如果 RXNE=1，上一个有效数据还在内部寄存器 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 USART_SR 寄存器读访问和 USART_DR 读访问之间)接收到新的数据，此种情况也可能发生。

29.3.7.6 选择时钟源和合适的过采样方案

时钟源还决定通信速度范围（尤其是最大通信速度）。

接收器采用不同的用户可配置过采样技术（除了同步模式下），可以从噪声中提取有效数据。这可在最大通信速度与抗噪声/时钟误差性能之间实现最佳平衡。

可通过编程 USART_CR3 寄存器中的 OVER8 位来选择采样方法，且采样时钟可以是波特率时钟的 16 倍或 8 倍。

根据应用：

- 选择 8 倍过采样(OVER8 = 1)以获得更高的速度(高达 usart_clk/8)。这种情况下接收器对时钟偏差的最大容差将会降低。
- 选择 16 倍过采样(OVER8 = 0)以增加接收器对时钟偏差的容差。在这种情况下，最大速度被限制为最大 usart_clk/16。

帧中检测到噪声时：

- 在 RXNE 位(使能 FIFO 模式时为 RXFNE 位)的上升沿时 NE 位置 1。
- 无效数据从移位寄存器传送到 USART_DR 寄存器。
- 单字节通信时无中断产生。然而，在 RXNE 位(使能 FIFO 模式时为 RXFNE 位)生成中断时，该位出现上升沿。多缓冲区通信时，USART_CR3 寄存器中的 EIE 位置 1 时会发出中断。
- 先读出 USART_SR，再读出 USART_DR 寄存器，将清除 NE 标志位

注意： SPI 模式不支持噪声错误。

智能卡、 IrDA 和 LIN 模式下不可采用 8 倍过采样。在这些模式下，OVER8 位由硬件强制清零。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。

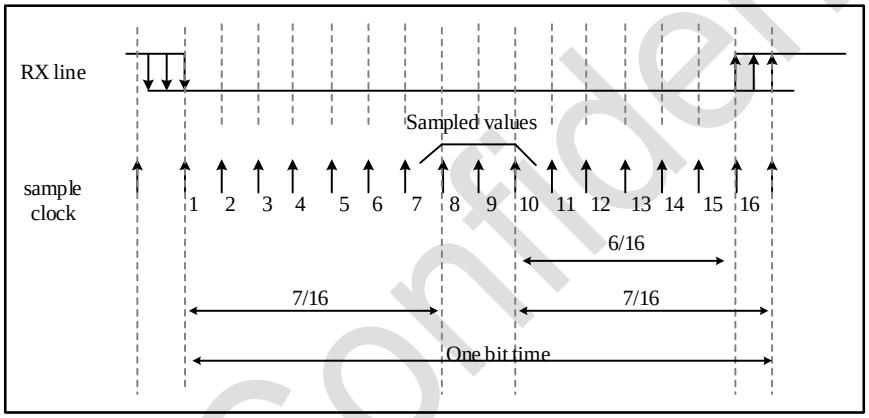


图 29-6 16 位过采样时噪声检测时数据采样

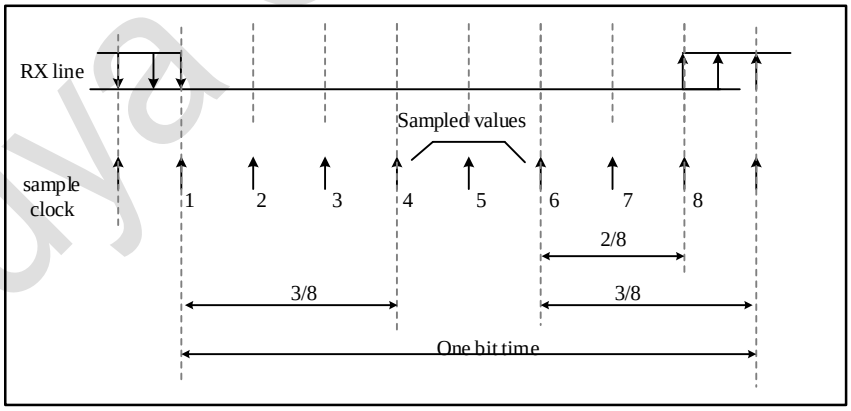


图 29-7 8 位过采样时噪声检测时数据采样

表 29-2 检测噪声的数据采样

采样值	NE 状态	接收的值	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

29.3.7.7 帧错误

如果接收数据时未在预期时间内识别出停止位，从而出现同步失效或过度的噪声，则会检测到帧错误。检测到帧错误被时：

- FE 位由硬件置 1
- 无效数据从移位寄存器传送到 USART_DR 寄存器(使能 FIFO 模式时为 RXFIFO)。
- 在单字节通信时，FE 位无法中断产生。然而，在 RXNE 位(使能 FIFO 模式时为 RXFN E 位)生成中断时，改位出现上升沿。多缓冲区通信时， USART_CR3 寄存器中的 EIE 位置 1 时会发出中断。
- 顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 FE 位。

注意：SPI 模式不支持帧错误。

29.3.7.8 接收期间可配置的停止位

可通过 USART_CR2 的控制位配置要接收的停止位的数量：可以是 1 或 2 个(正常模式下)，也可以是 0.5 或 1.5 个(智能卡模式下)。

0.5 个停止位(在智能卡模式下接收时) ：不会对 0.5 个停止位进行采样。结果，选择 0.5 个停止位时，无法检测到帧错误和断开帧。

1 个停止位：将在第 8、第 9 和第 10 次采样时对 1 个停止位进行采样。

1.5 个停止位(在智能卡模式下)：

在智能卡模式下发送时，设备必须检查数据是否正确发送。因此，必须使能接收器块(USART_CR1 中的 RE = 1)并检查停止位，以测试智能卡是否已检测到奇偶校验错误。发生奇偶校验错误时，智能卡会在采样时将数据信号强制为低电平(即 NACK 信号)，该信号被标记为帧错误。之后， FE 标志在 1.5 个停止位的末尾由 RXNE 标志(使能 FIFO 模式时为 RXFNE)置 1。在第 16、第 17 和第 18 次采样时对 1.5 个停止位进行采样(停止位采样开始后维持 1 个波特率时钟周期)。 1.5 个停止位可分为 2 个部分： 0.5 个波特时钟周期(未发生任何动作)，然后是 1 个正常的停止位周期(一半时间处进行采样)。

2 个停止位：

采样 2 个停止位时在第 8、第 9 和第 10 次采样时对第一个停止位进行采样。如果在第一个停止位期间检测到帧错误，则帧错误标志会置 1。发生帧错误时不检测第 2 个停止位。 RXNE 标志(使能 FIFO 模式时为 RXFNE)将在第一个停止位结束时置 1。

29.3.8 USART 分数波特率的产生

接收器和发送器的波特率 USARTDIV 由 USART_BRR 寄存器中的的整数和小数部分设置，且接收波特率和发送波特率相等。

1:16 位过采样模式

$$\text{TX/RX 波特率} = \frac{fck}{(16 * USARTDIV)}$$

这里的 f_{CK} 是给外设的时钟。

注： 在写入 USART_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。

表 29-3 设置波特率时的误差计算

波特率		f_{ck} = 8 MHz			f_{ck} = 36 MHz			f_{ck} = 100 MHz		
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%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.400	208.375	0.02%	2.400	937.5	0.00%	2.39998	2604.1875	0.00%
2	9.6	9.604	52.0625	0.04%	9.600	234.375	0.00%	9.59417	651.4375	0.06%
3	19.2	19.198	26.04375	0.01%	19.200	117.1875	0.00%	19.2012	325.5000	0.01%
4	57.6	57.554	8.6875	0.08%	57.600	39.0625	0.00%	57.6037	108.5000	0.01%
5	115.2	115.942	4.3125	0.64%	115.385	19.5	0.16%	115.207	54.2500	0.01%
6	230.4	228.571	2.1875	0.80%	230.769	9.75	0.16%	230.415	27.1250	0.01%
7	460.8	470.588	1.0625	2.08%	461.538	4.875	0.16%	460.829	13.5625	0.01%
8	921.6	-	不可能	-	923.077	2.4375	0.16%	925.926	6.7500	0.47%
9	2250	-	不可能	-	2250.000	1	0.00%	2272.73	2.7500	1.00%
10	4500	-	不可能	不可能	不可能	不可能	不可能	4545.45	1.3750	1.00%

注:

1. CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

2. 8 位过采样模式

$$\text{TX/RX 波特率} = \frac{fck}{(8*USARTDIV)}$$

3. 不得使用小于 1 分频系数

29.3.9 USART 接收器容忍时钟的变化

只有当整体的时钟系统地变化小于 USART 异步接收器能够容忍的范围， USART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的变化(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足： DTRA + DQUANT + DREC + DTCL < USART 接收器的容忍度

表 29-4 DIV_Fraction 为 0 时 USART 接收容忍度

M 位	OVR8=0		OVR8=1	
	认为 NF 是错误	不认为 NF 是错误	认为 NF 是错误	不认为 NF 是错误
00	3.38%	3.79%	1.05%	1.05%
01	3.31%	4.17%	1.05%	1.05%
10	3.65%	4.38%	1.05%	1.05%

表 29-5 DIV_Fraction 不为 0 时 USART 接收容忍度

M 位	OVR8=0		OVR8=1	
	认为 NF 是错误	不认为 NF 是错误	认为 NF 是错误	不认为 NF 是错误
00	2.08%	3.78%	0.89%	1.02%
01	3.71%	3.76%	0.88%	0.88%
10	2.14%	3.83%	0.88%	0.89%

29.3.10 USART 自动波特率检测

自动波特率检测

USART 能够基于接收到的字符自动检测波特率，并配置波特率寄存器 USART_BRR.

自动波特率检测适用于如下情况：

系统通讯速度预先未知；

系统使用精度较低的时钟源，这种机制允许在不测量时钟偏差的情况下获得正确的波特率；

时钟源频率必须与期望的通讯速度兼容:

1.16 位过采样的波特率必须在 f_{CK}/65535 和 f_{CK}/16 之间。

2.8 位过采样的波特率必须在 f_{CK}/65535 和 f_{CK}/8 之间。

通过 USART_CR3.ABRMOD 配置自动波特率检测模式。基于不同的字符模式有 2 种模式。在这些自动波特率模式中，在同步数据接收过程中测量波特率数次，并将每次测量结果与前一次进行比较。

自动波特率检测模式

Mode0：此模式针对以 1 开头为起始的情况。这种情况下，是测量起始位到 1（下降沿到上升沿）。

Mode1：此模式针对以 10xx 位模式开头的任何字符。在这种情况下，USART 测量开始位和第一个数据位的持续时间。测量是沿下降沿到下降沿进行的，确保在慢信号倾斜的情况下有更好的精度。

并行数据模式：对 RX 线的每个中间转换执行检测。如果 RX 上的转换没有与接收器充分同步(接收器基于 0 位计算的波特率)，就会产生一个错误。

注：在自动波特率检测模式时，如果波特率时钟接近于 fck 时钟的一分频，会导致误差较大。

29.3.11 USART 多处理器通信

通过 USART 可以实现多处理器通信(将几个 USART 连在一个网络里)。例如某个 USART 设备可以是主设备，它的 TX 输出和其他 USART 从设备的 RX 输入相连接； USART 从设备各自的 TX 输出逻辑与在一起，并且和主设备的 RX 输入相连接。

在多处理器配置中，我们通常希望只有被寻址的接收者才被激活，来接收随后的数据，这样就可以减少由未被寻址的接收器的参与带来的多余的 USART 服务开销。未被寻址的设备配置为静默模式。

通过静默功能，可以将非寻址设备置于静默模式。要使用静默模式特性，必须在 USART_CR1 寄存器中设置 RWU 位。

在静默模式里：

- 任何接收状态位都不会被设置。
- 所有接收中断被禁止。
- USART_CR1 寄存器中的 RWU 位被置 1。 RWU 可以被硬件自动控制或在某个条件下由软件写入。
- 根据 USART_CR1 寄存器中的 WAKE 位状态， USART 可以用二种方法进入或退出静默模式。
- 如果 WAKE 位被设置为 0：进行空闲总线检测。
- 如果 WAKE 位被设置为 1：进行地址标记检测。

空闲总线检测(WAKE=0)

当 RWU 位被写 1 时， USART 进入静默模式。当检测到一空闲帧时，它被唤醒。RWU 还可以被软件写 0。下图给出利用空闲总线检测来唤醒和进入静默模式的一个例子

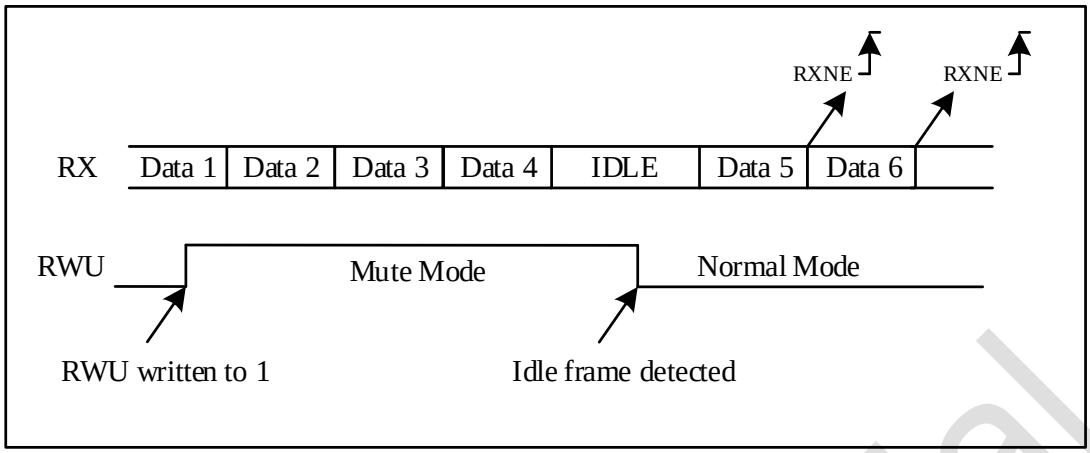


图 29-8 利用空闲总线检测的静默模式

地址标记(address mark)检测(WAKE=1)（4 位或者 7 位地址检测）

在这个模式里，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。在一个地址字节中，目标接收器的地址被放在 4 位或者 7 位 LSB 中。这个地址被接收器同它自己地址做比较，接收器的地址被编程在 USART_CR2 寄存器的 ADD。选择 7 位或 4 位地址检测是使用 ADDM7 位完成的。

如果接收到的字节与它的编程地址不匹配时， USART 进入静默模式。此时，硬件设置 RWU 位。接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 USART 已经在静默模式。

当接收到的字节与接收器内编程地址匹配时， USART 退出静默模式。然后 RWU 位被清零，随后的字节被正常接收。收到这个匹配的地址字节时将设置 RXNE 位，因为 RWU 位已被清零。当接收缓冲器不包含数据时(USART_SR 的 RXNE=0)， RWU 位可以被写 0 或 1。否则，该次写操作被忽略。下图给出利用地址标记检测来唤醒和进入静默模式的例子。

注：在 7 位和 9 位数据模式下，分别对 6 位和 8 位地址(ADD[5:0]和 ADD[7:0])进行地址检测。这个地址字节不会设置 RXNE 标志，当 USART 进入静默模式时，不会发出中断或 DMA 请求。当启用 FIFO 管理时，软件在进入静默模式之前应确保 RXFIFO 中至少有一个空位置。当 RWU 写为 1 时，USART 也进入静默模式。在这种情况下，RWU 位也会自动设置。当接收到与编程地址匹配的地址字符时，USART 退出静默模式。清除 RWU 位，正常接收后续字节。当 RWU 位已被清除，地址字符可以检测 RXNE/RXFNE 位。

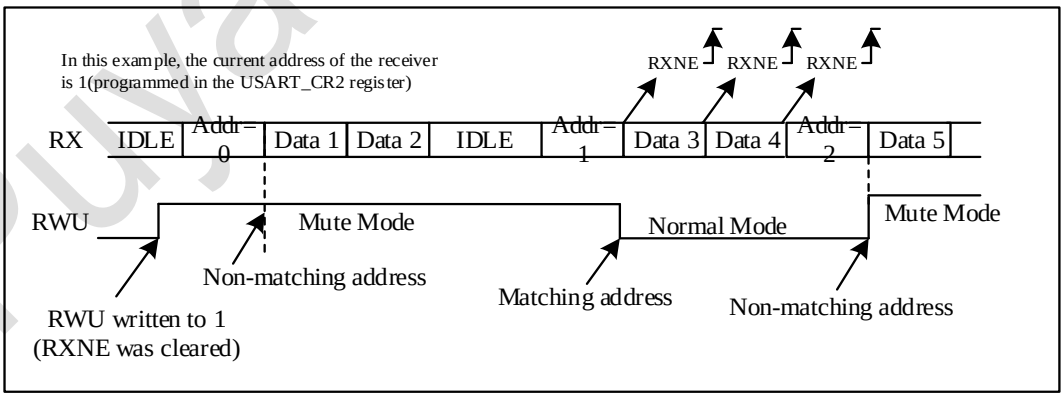


图 29-9 利用地址标记检测的静默模式

29.3.12 USART Modbus 通信

USART 为 Modbus/RTU 和 Modbus/ASCII 协议的实现提供基本支持。Modbus/RTU 是一种半双工的块传输协议。协议的控制部分(地址识别、块完整性控制和命令解释)必须在软件中实现。USART 提供了基本的支持：结束块检测，没有软件开销或其他资源。

Modbus/RTU

在这种模式下，一个块的结束被“沉默”(空闲行)识别超过 2 个字符。该功能通过可编程超时功能实现。超时功能和中断必须通过 USART_CR2 寄存器中的 RTOEN 位和 USART_CR1 寄存器中的 RTOIE 来激活。

对应于 2 个字符时间(例如 22 个位时间)的超时值必须在 RTO 寄存器中编程。当接收线在这段时间内空闲时，在接收到最后一个停止位之后，产生一个中断，通知软件当前块接收完成。

Modbus/ASCII

在这种模式下，块的结束由特定的(CR/LF)字符序列识别。USART 使用字符匹配功能来管理这个机制。通过在 ADD[7:0]字段中编写对应字符，当收到对应字符时退出静默模式。

29.3.13 USART 校验控制

设置 USART_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)。

偶校验：校验位使得一帧中的 6 个、7 个或 8 个 LSB 数据以及校验位中 1 的个数为偶数。

例如：数据=00110101，有 4 个 1，如果选择偶校验(在 USART_CR1 中的 PS = 0)，校验位将是 0。

奇校验：此校验位使得一帧中的 6 个、7 个或 8 个 LSB 数据以及校验位中 1 的个数为奇数。

例如：数据=00110101，有 4 个 1，如果选择奇校验(在 USART_CR1 中的 PS = 1)，校验位将是 1。

传输模式：如果 USART_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,则 1 的个数为偶数，如果选择奇校验，则 1 的个数为奇数。如果奇偶校验失败，USART_SR 寄存器中的 PE 标志被置 1，并且如果 USART_CR1 寄存器的 PEIE 在被预先设置的话，产生中断)。

29.3.14 USART LIN(局域互联网)模式

配置 USART_CR2.LINEN=1 选择 LIN 模式。在 LIN 模式下，下列位必须保持为 0：

1.USART_CR2 寄存器的 STOP/CLKEN;

2.USART_CR3 寄存器的 SCEN/HDSEL/IREN.

发送

与一般 USART 发送相比，LIN 发送有如下区别：

M=0，数据长度为 8 位;

需要配置 USART_CR2.LINEN=1。置位 SBK 后将发送 13 位 0 作为断开帧。

接收

当 LIN 模式被使能时，断开帧检测电路被激活。该检测完全独立于 USART 接收器。断开帧只要一出现就能检测到，不管是在总线空闲时还是在发送某数据帧期间，数据帧还未完成，又插入了断开帧的发送。当接收器被激活时(USART_CR1 的 RE=1)，电路检测 RX 上的起始信号。检测起始位的方法同检测断开帧或数据是一样的。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 USART_CR2 的 LBDL = 0) 或 11 个(当 USART_CR2 的 LBDL = 1)连续位都是 0，并且又跟着一个定界符，USART_SR 的 LBD 标志被置位。如果 LBDIE 位=1，中断产生。在确认断开帧前，要检查定界符，因为它意味 RX 线已经回到高电平。

如果在第 10 或 11 个采样点之前采样到了 1，检测电路取消当前检测并重新寻找起始位。如果 LIN 模式被禁止，接收器继续如正常 USART 那样工作，不需要考虑检测断开帧。

如果 LIN 模式没有被激活(LINEN=0)，接收器仍然正常工作于 USART 模式，不会进行断开检测。

如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误(也就是停止位检测到'0'，这种情况出现在断开帧)，接收器就停止，直到断开帧检测电路接收到一个 1(这种情况发生于断开帧没有完整的发出来)，或一个定界符(这种情况发生于已经检测到一个完整的断开帧)。

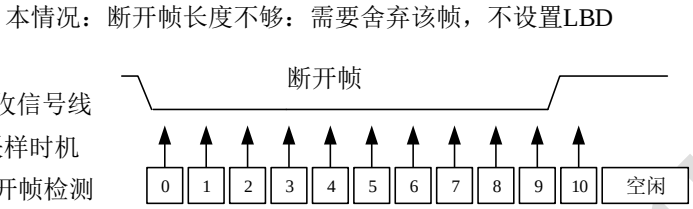


图 29-10 LIN 模式下的断开帧长度不够的情况

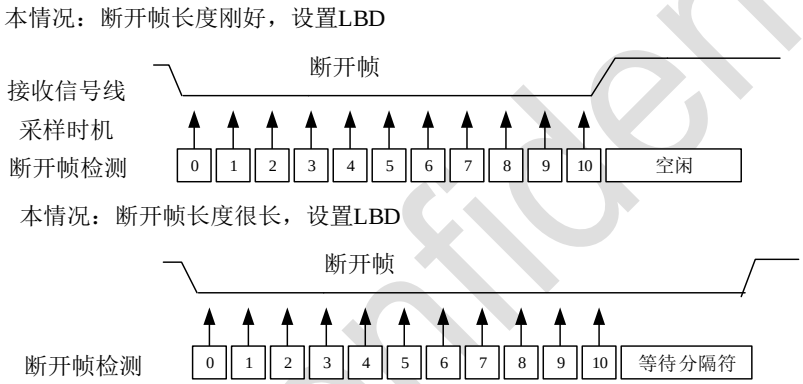


图 29-10 LIN 模式下的断开帧长度够的情况

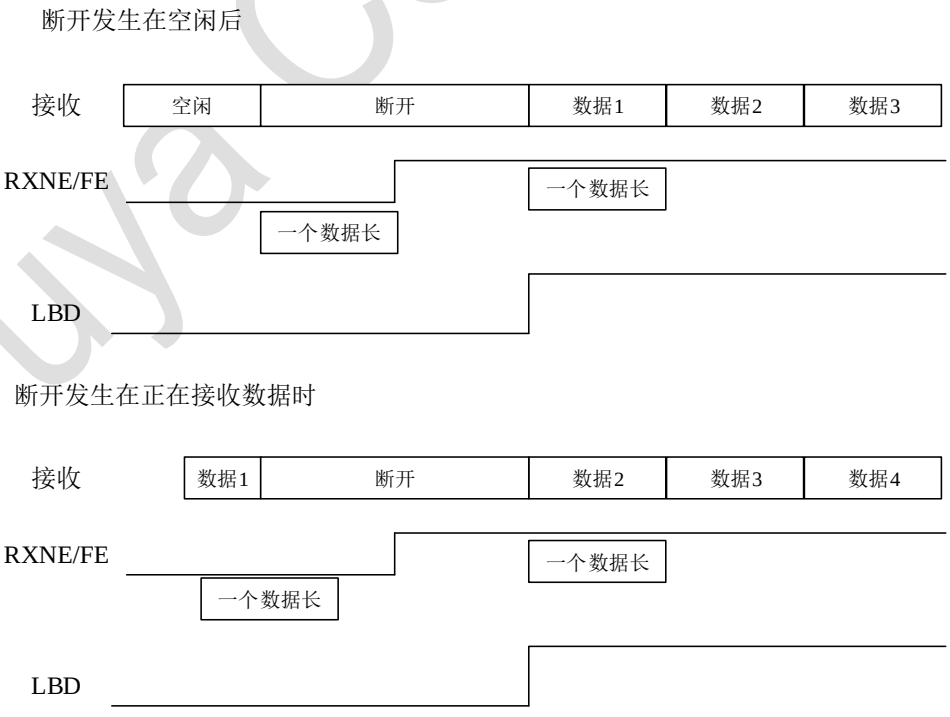


图 29-11 LIN 模式下的断开检测与帧错误的检测

29.3.15 USART 同步模式

通过在 USART_CR2 寄存器上写 CLKEN 位选择同步模式

主模式

在同步模式里，下列位必须保持清零状态：

- USART_CR2 寄存器中的 LINEN 位
- USART_CR3 寄存器中的 SCEN,HDSEL 和 IREN 位

USART 允许用户以主模式方式控制双向同步串行通信。 CK 脚是 USART 发送器时钟的输出。在起始位和停止位期间， CK 脚上没有时钟脉冲。根据 USART_CR2 寄存器中 LBCL 位的状态，决定在最后一个有效数据位期间产生或不产生时钟脉冲。 USART_CR2 寄存器的 CPOL 位允许用户选择时钟极性， USART_CR2 寄存器上的 CPHA 位允许用户选择外部时钟的相位。

在总线空闲期间，实际数据到来之前以及发送断开帧的时候，外部 CK 时钟不被激活。同步模式时，USART 发送器和异步模式里工作一模一样。但是因为 CK 是与 TX 同步的(根据 CPOL 和 CPHA)，所以 TX 上的数据是随 CK 同步发出的。

同步模式的 USART 接收器工作方式与异步模式不同。如果 RE=1，数据在 CK 上采样(根据 CPOL 和 CPHA 决定在上升沿还是下降沿)，不需要任何的过采样。但必须考虑建立时间和保持时间(取决于波特率，1/16 位时间)。

注：

1．CK 脚同 TX 脚一起联合工作。因而，只有在使能了发送器(TE = 1)，并且发送数据时(写入数据至 USART_DR 寄存器)才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。

2．应该在发送器和接收器都被禁止时，配置 LBCL,CPOL 和 CPHA；当使能了发送器或接收器时，这些位不能被改变

3．建议在同一条指令中设置 TE 和 RE，以减少接收器的建立时间和保持时间。

4．USART 只支持主模式：它不能用来自其他设备的输入时钟接收或发送数据(CK 永远是输出)。

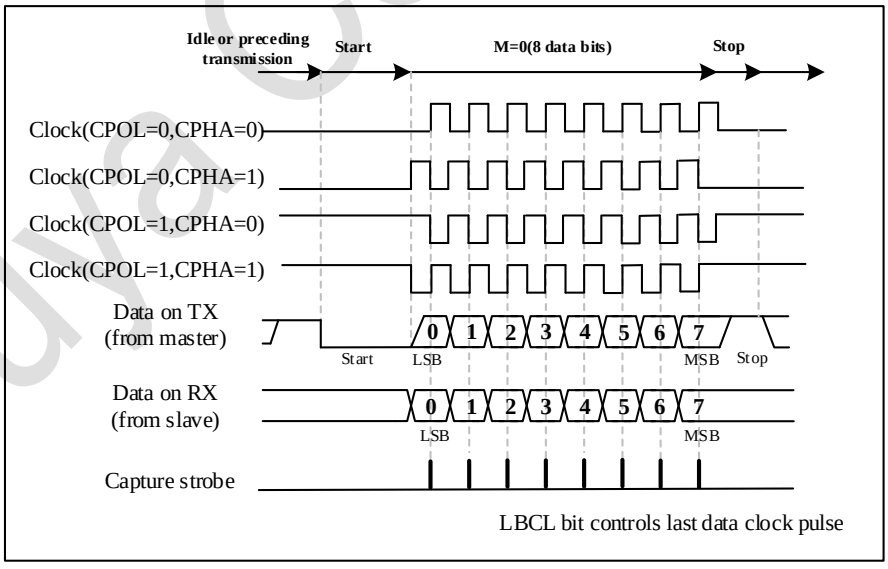


图 29-12 USART 数据时钟时序示例（M=00）

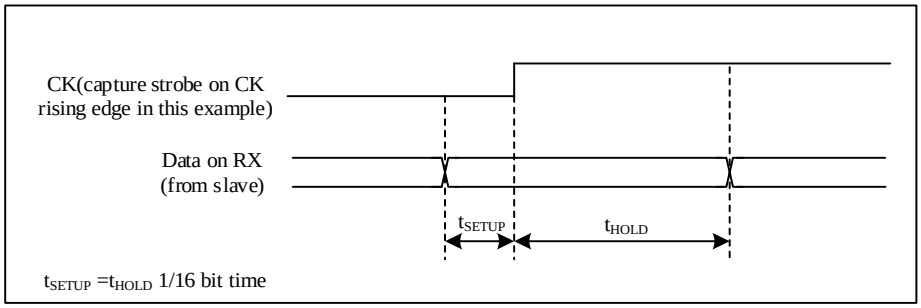


图 29-13 RX 数据建立/保持时间

29.3.16 USART 单线半双工通信

单线半双工模式通过设置 USART_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：

- USART_CR2 寄存器的 LINEN 和 CLKEN 位
- USART_CR3 寄存器的 SCEN 和 IREN 位

USART 可以配置成遵循单线半双工协议。在单线半双工模式下， TX 和 RX 引脚在芯片内部互连。使用 USART_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，它在空闲状态或接收状态时表现为一个标准 I/O 口。这就意味该 I/O 在不被 USART 驱动时，必须配置成开漏。

除此以外，通信与正常 USART 模式类似。由软件来管理线上的冲突(例如通过使用一个中央仲裁器)。特别的是，发送从不会被硬件所阻碍。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。

29.3.17 USART 接收超时

通过在 USART_CR2 控制寄存器中设置 RTOEN 位来启用接收器超时特性。

超时时间是使用 USART_RTOR 寄存器中的 RTO 位域进行设置的。

当超过设置的超时时间时，USART_SR 寄存器中的 RTOF 位被设置为 1。如果设置了 CR1 的 RTOIE，会产生对应的中断。

29.3.18 USART 智能卡

设置 USART_CR3 寄存器的 SCEN 位选择智能卡模式。在智能卡模式下，下列位必须保持清零：

- USART_CR2 寄存器的 LINEN 位
- USART_CR3 寄存器的 HDSEL 位和 IREN 位

此外， CLKEN 位可以被设置，以提供时钟给智能卡。

该接口符合 ISO7816-3 标准，支持智能卡异步协议。T = 0 (character mode) 和 T = 1 (block mode)都支持。

T = 0 (character mode)

USART 应该被设置为：

- 8 位数据位加校验位： 此时 USART_CR1 寄存器中 M=01、 PCE=1
- 发送和接收时为 1.5 个停止位：即 USART_CR2 寄存器的 STOP=11

注： 也可以在接收时选择 0.5 个停止位，但为了避免在 2 种配置间转换，建议在发送和接收时使用 1.5 个停止位。

在 T = 0(字符)模式下，在保护时间段内每个字符的末尾表示奇偶校验错误。

下图给出的例子说明了数据线上，在有校验错误和没校验错误两种情况下的信号。

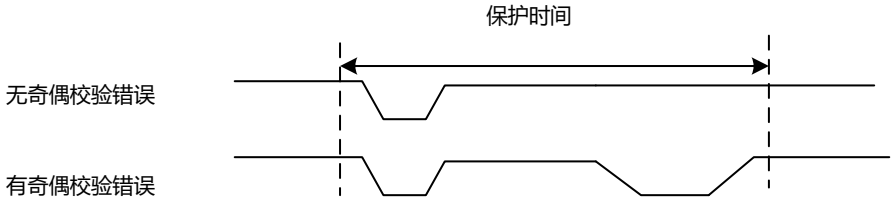


图 29-14 ISO7816-3 异步协议

当与智能卡相连接时， USART 的 TX 输出会驱动一条双向线(它也由智能卡也驱动)。必须将 TX 引脚配置成开漏引脚。

智能卡是一个单线半双工通信协议

- 从发送移位寄存器把数据发送出去，要被延时最小 1/2 个波特时钟周期。在正常操作时，一个满的发送移位寄存器将在下一个波特时钟沿开始向外移出数据。在智能卡模式里，此发送被延迟 1/2 个波特时钟周期。
- 如果在接收一个设置为 0.5 或 1.5 个停止位的数据帧期间，检测到奇偶校验错误，在完成接收该帧后(即停止位结束时)，发送线被拉低一个波特时钟周期。这是告诉智能卡发送到 USART 的数据没有被正确地接收到。此 NACK 信号(拉低发送线一个波特时钟周期)在发送端将产生一个帧错误(发送端被配置成 1.5 个停止位)。应用可以根据协议处理重新发送数据。如果设置了 NACK 控制位，发生校验错误时接收器会给出一个 NACK 信号；否则就不会发送 NACK。TXE 位(在启用 FIFO 模式的情况下 TXFNF 位)可以使用寄存器中的 TXFRQ 位进行设置。如果接收到的字符错误，RXNE(在启用 FIFO 模式的情况下为 RXFNE)/接收 DMA 请求不被激活。
- 智能卡传输中的自动重试:在 USART 检测 NACK 和重复字符的起始位之间插入 2.5 个波特率时钟周期的延迟。TC 位在接收到最后一个重复字符后立即设置(无保护时间)。如果软件想要再次重复它，它必须确保标准要求的最小 2 个波特时钟周期。
- 在传输中，USART 在两个连续字符之间插入保护时间(按照保护时间寄存器)。由于保护时间是在前一个字符的停止位之后测量的，因此必须将 GT[7:0]寄存器编程为所需的 CGT(字符保护时间，由 7816-3 规范定义)减去 12(一个字符的持续时间)。
- TC 标志的置起可以通过编程保护时间寄存器得以延时。在正常操作时，当发送移位寄存器变空并且没有新的发送请求出现时， TC 被置起。在智能卡模式里，空的发送移位寄存器将触发保护时间计数器开始向上计数，TC 在这段时间被强制拉低，当保护时间计数器达到保护时间寄存器中的值时， TC 被置高。
- TC 标志的撤销不受智能卡模式的影响。
- 如果发送器检测到一个帧错误(收到接收器的 NACK 信号)，发送器的接收功能模块不会把 NACK 当作起始位检测。根据 ISO 协议，接收到的 NACK 的持续时间可以是 1 或 2 个波特时钟周期。
- 在接收器这边，如果一个校验错误被检测到，并且 NACK 被发送，接收器不会把 NACK 检测成起始位。

注意：

1. 断开帧在智能卡模式里没有意义。一个带帧错误的 0x00 数据将被当成数据而不是断开帧。
2. 当来回切换 TE 位时，没有 IDLE 帧被发送。 ISO 协议没有定义 IDLE 帧。

下图详述了 USART 是如何采样 NACK 信号的。在这个例子里， USART 正在发送数据，并且被配置成 1.5 个停止位。为了检查数据的完整性和 NACK 信号， USART 的接收功能块被激活。

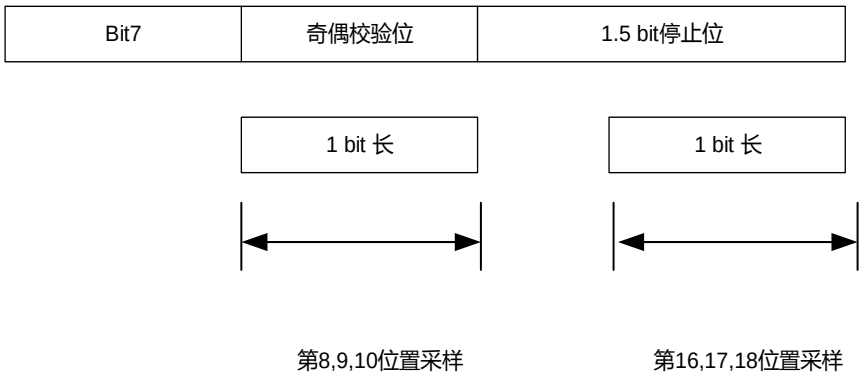


图 29-15 使用 1.5 停止位检测奇偶检验错

USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式里， CK 不和通信直接关联，而是先通过一个 5 位预分频器简单地用内部的外设输入时钟来驱动智能卡的时钟。分频率在预分频寄存器 USART_GTPR 中配置。 CK 频率可以从 fCK/2 到 fCK/62，这里的 fCK 是外设输入时钟。

T = 1 (block mode)

在块模式下，向智能卡发出读取请求时，软件必须将 RTOR 寄存器编程为 BWT(块等待时间)- 11 值。如果在此期限到期之前没有收到卡的答复，则会产生超时中断。如果第一个字符在期限到期之前被接收，RXNE/RXFNE 中断将发出信号

注：RXNE/RXFNE 中断必须启用，即使在 DMA 模式下使用 USART 以块模式从智能卡读取。同时，DMA 必须只在接收到第一个字节之后才启用。

在智能卡协议定义中，BWT/CWT 值应该从最后一个字符的开始(起始位)定义。考虑到最后一个字符本身的长度，RTO 寄存器必须分别编程为 BWT -11 或 CWT -11。

块长度计数器用于计算 USART 接收到的所有字符。当 USART 发送时，这个计数器被重置。块的长度由智能卡在块的第三个字节(序言字节)中进行通信。这个值必须编程到 USART_RTOR 寄存器的 BLEN 字段中。当使用 DMA 模式时，在块开始之前，必须将该寄存器字段编程为最小值。根据这个值，在接收到的第 4 个字符之后产生中断。软件必须读取 LEN 字段(第三个字节)，其值必须从接收缓冲区读取。在中断驱动接收模式下，块的长度可以通过软件或编程 BLEN 值来检查。但是，在块开始之前，可以编程 BLEN 的最大值(0xFF)。真正的值是在接收到第三个字符后编程的。

如果块使用 LRC 纵向冗余检查(尾声字节)，则 BLEN = LEN。如果块使用 CRC 机制(2 字节)，则必须编程 BLEN = LEN+1。总块长度(包括序言、尾声和信息字节)等于 BLEN+4。块的结束通过 EOBF 标志和中断(当设置 EOBIE 位时)向软件发出信号。

29.3.19 USART IrDA SIR ENDEC 功能模块

通过设置 USART_CR3 寄存器的 IREN 位选择 IrDA 模式。在 IRDA 模式里，下列位必须保持为零：

- USART_CR2 寄存器的 LINEN,STOP 和 CLKEN 位
- USART_CR3 寄存器的 SCEN 和 HDSEL 位。

IrDA 正常模式

IrDA SIR 物理层规定使用反相归零调制方案(RZI)，该方案用一个红外光脉冲代表逻辑'0'(见下图)。 SIR 发送编码器对从 USART 输出的 NRZ(非归零)比特流进行调制。输出脉冲流被传送到一个外部输出驱动器和红外 LED。 USART 为 SIR ENDEC 最高只支持到 115.2Kbps 速率。在正常模式里，脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 USART。在空闲状态里，解码器输入通常是高(标记状态)。发送编码器输出的极性和解码器的输入相反。当解码器输入低时，检测到一个起始位。

- IrDA 是一个半双工通信协议。如果发送器忙(也就是 USART 正在送数据给 IrDA 编码器)，IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙(也就是 USART 正在接收从 IrDA 解码器来的解码数据)，从 USART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- SIR 发送逻辑把'0'作为高脉冲发送，把'1'作为低电平发送。脉冲的宽度规定为正常模式时位周期的 3/16。
- SIR 接收逻辑把高电平状态解释为'1'，把低脉冲解释为'0'。
- 发送编码器输出与解码器输入有着相反的极性。当空闲时， SIR 输出处于低状态。
- SIR 解码器把 IrDA 兼容的接收信号转变成给 USART 的比特流。
- IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收器端的脉冲检测逻辑滤除宽度小于 2 个 PSC 周期的脉冲(PSC 是在 IrDA 低功耗波特率寄存器 USART_GTPR 中编程的预分频值)。宽度小于 1 个 PSC 周期的脉冲一定被滤除掉，但是那些宽度大于 1 个而小于 2 个 PSC 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。当 PSC=0 时， IrDA 编码器/解码器不工作。
- 接收器可以与一低功耗发送器通信。
- 在 IrDA 模式里， USART_CR2 寄存器上的 STOP 位必须配置成 1 个停止位。

IrDA 低功耗模式

发送器：

在低功耗模式，脉冲宽度不再持续 3/16 个位周期。取而代之，脉冲的宽度是低功耗波特率的 3 倍，它最小可以是 1.42 MHz。通常这个值是 1.8432MHz(1.42 MHz < PSC < 2.12 MHz)。一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。

接收器

低功耗模式的接收类似于正常模式的接收。为了滤除尖峰干扰脉冲， USART 应该滤除宽度小于 1 个 PSC 的脉冲。只有持续时间大于 2 个周期的 IrDA 低功耗波特率时钟(USART_GTPR 中的 PSC)的低电平信号才被接受为有效的信号。

注：

1. 宽度小于 2 个大于 1 个 PSC 周期的脉冲可能会也可能不会被滤除。
2. 接收器的建立时间应该由软件管理。 IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时(IrDA 是一个半双工协议)。

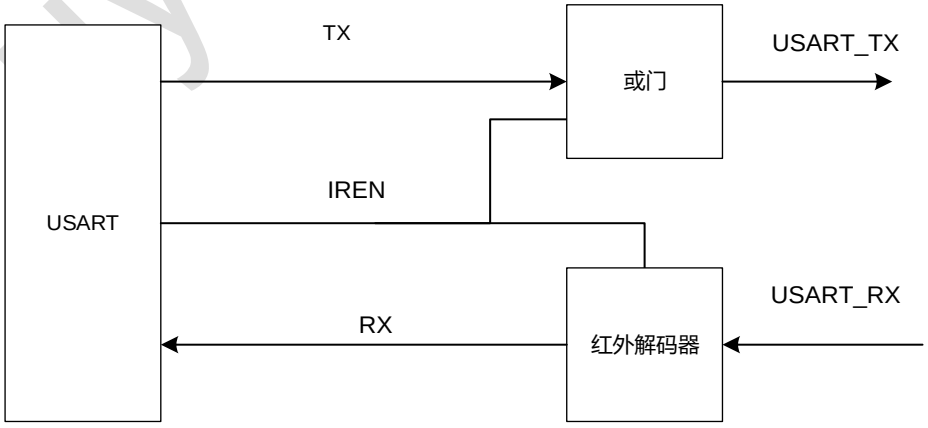


图 29-16 IrDA SIR ENDEC 框图

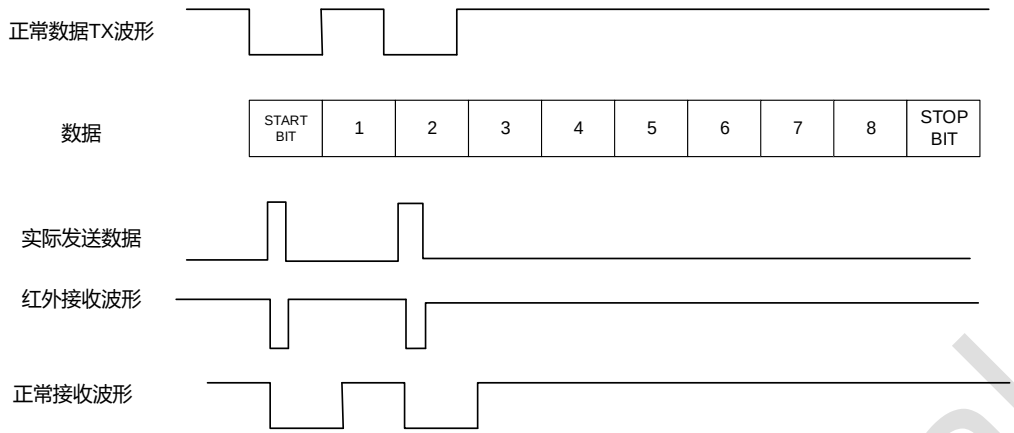


图 29-17 IrDA 数据调制（3/16）—普通模式

29.3.20 USART 利用 DMA 连续通信

USART 可以利用 DMA 连续通信。 RX 缓冲器和 TX 缓冲器的 DMA 请求是分别产生的。

注意：在 USART_SR 寄存器里，可以清零 TXE/RXNE 标志来实现连续通信。

利用 DMA 发送：

使用 DMA 进行发送，可以通过设置 USART_CR3 寄存器上的 DMAT 位激活。当 TXE 位被置为'1'时，DMA 就从指定的存储器区传送数据到 USART_DR 寄存器。为 USART 的发送分配一个 DMA 通道的步骤如下：

1、将 DMA 的目的地址配置为 USART_DR 寄存器的地址。在每次 TXE(如果启用 FIFO 模式 TXFNF)之后，数据会从存储器中移动到这个地址

2、将 DMA 的源地址配置为存放数据的存储器地址。在每次 TXE(或 TXFNF，如果启用 FIFO 模式)事件之后，数据从这个存储器加载到 USART_DR 寄存器中

3、在 DMA 控制寄存器中配置要传输的总的字节数。

4、在 DMA 寄存器上配置通道优先级。

5、根据应用程序的要求，配置在传输完成一半还是全部完成时产生 DMA 中断。

6、在 DMA 寄存器上激活该通道。

当传输完成 DMA 控制器指定的数据量时， DMA 控制器在该 DMA 通道的中断向量上产生一中断。检测 USART_SR 寄存器的 TC 标志可以确认 USART 通信是否结束，这样可以在关闭 USART 或进入停机模式之前避免破坏最后一次传输的数据；软件需要先等待 TXE=1，再等待 TC=1。

利用 DMA 接收：

可以通过设置 USART_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 控制器就把数据从 USART_DR 寄存器传送到指定的 SRAM 区。为 USART 的接收分配一个 DMA 通道的步骤如下：

1、将 DMA 的源地址配置为 USART_DR 寄存器的地址。在每个 RXNE 事件后，将从此地址读出数据并传输到存储器中。

2、将 DMA 的目的地址配置为想要存放数据的存储器地址。在每个 RXNE 事件后，数据将从 USART_DR 传输到此存储器。

3、在 DMA 控制寄存器中配置要传输的总的字节数。

4、在 DMA 寄存器上配置通道优先级。

5、根据应用程序的要求配置在传输完成一半还是全部完成时产生 DMA 中断。

6、在 DMA 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。

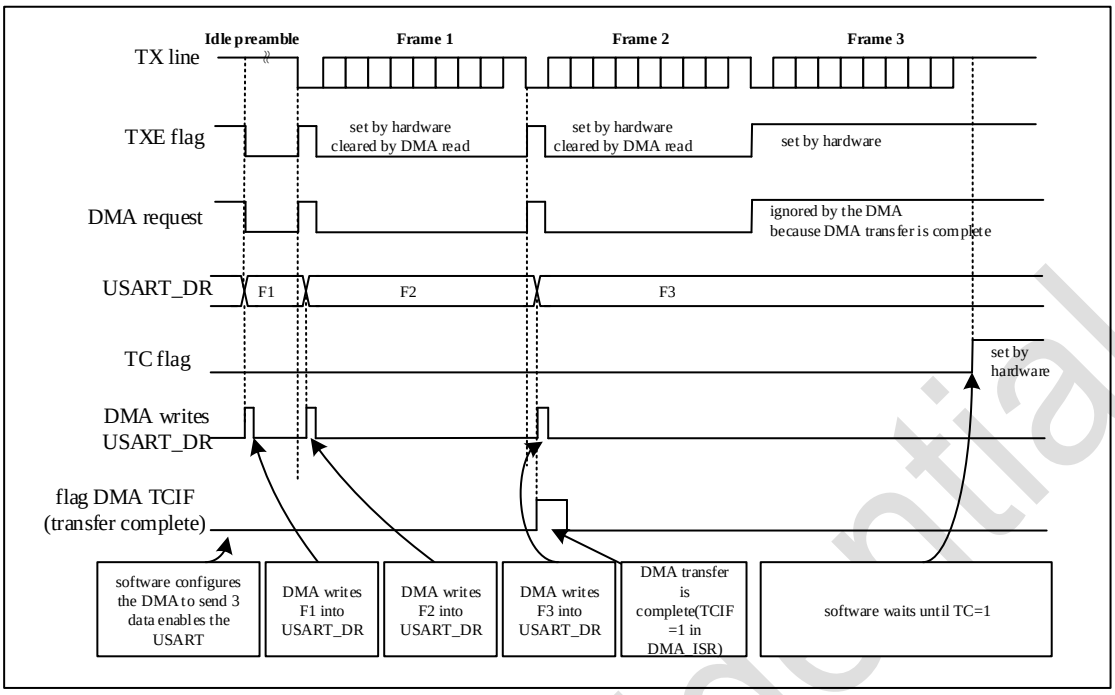


图 29-18 利用 DMA 发送

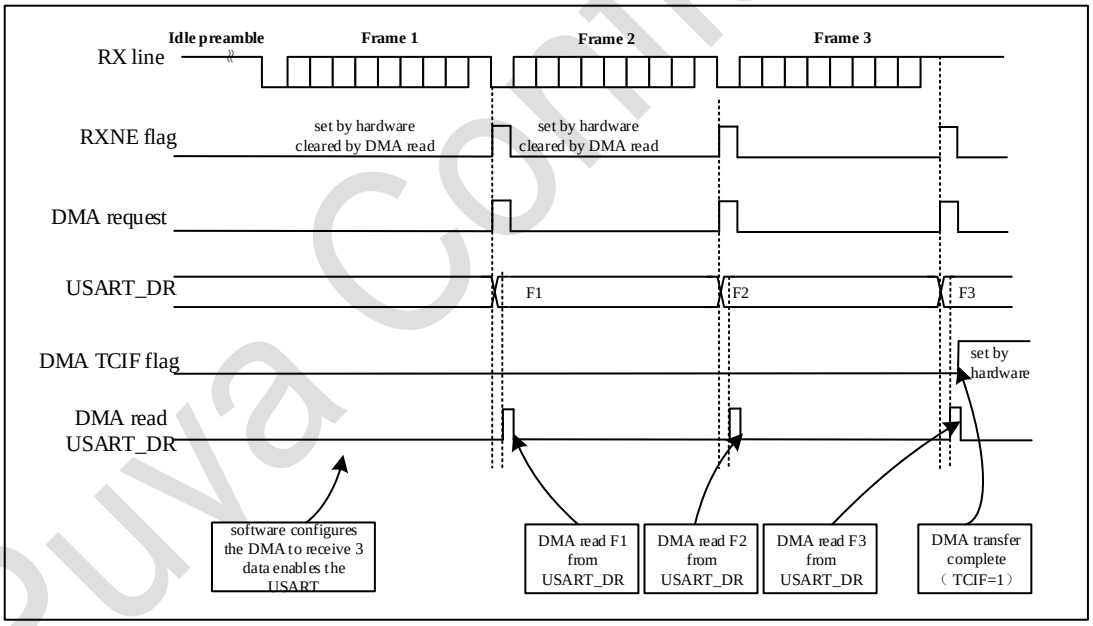


图 29-19 利用 DMA 接收

多缓冲区通信中的错误标志和中断产生

在多缓冲区通信的情况下，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。

29.3.21 USART 硬件流控制

RS232 硬件流控

利用 CTS 输入和 RTS 输出可以控制 2 个设备间的串行数据流。下图表明在这个模式里如何连接 2 个设备。

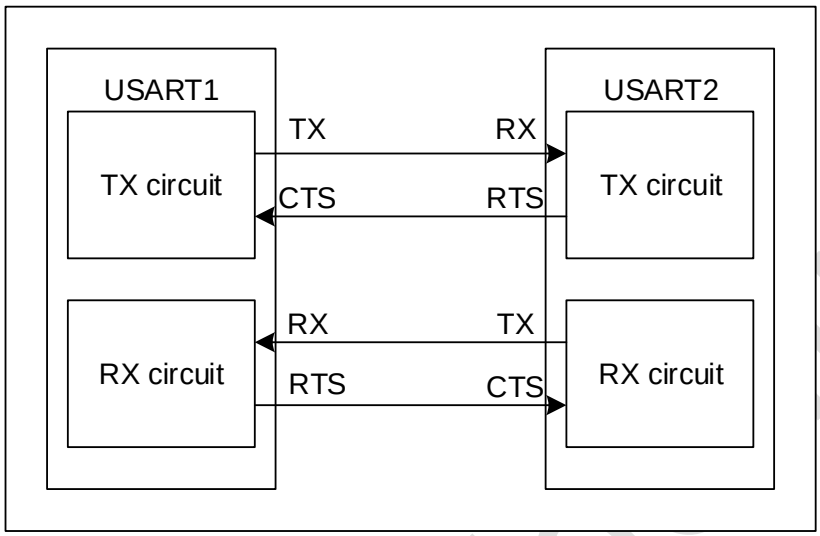


图 29-20 两个 USART 间的硬件流控制

通过将 USART_CR3 中的 RTSE 和 CTSE 置位，可以分别独立地使能 RTS 和 CTS 流控制。

RTS 流控制：

如果 RTS 流控制被使能(RTSE=1)，只要 USART 接收器准备好接收新的数据， nRTS 就变成有效(接低电平)。当接收寄存器内有数据到达时， nRTS 被释放，由此表明希望在当前帧结束时停止数据传输。下图是一个启用 RTS 流控制的通信的例子。

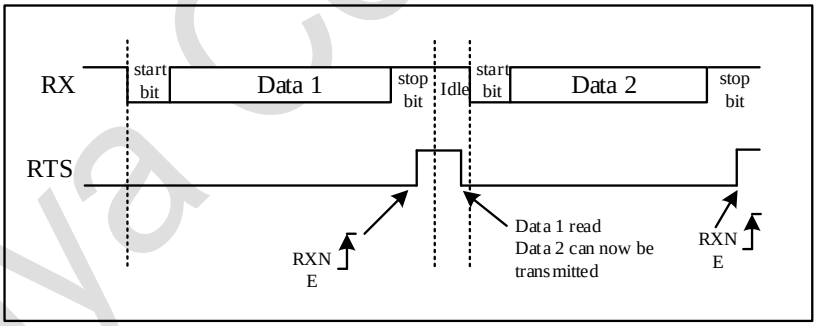


图 29-21 RTS 流控制

CTS 流控制：

如果 CTS 流控制被使能(CTSE=1)，发送器在发送下一帧前检查 CTS 输入。如果 CTS 有效(被拉成低电平)，则下一个数据被发送(假设那个数据是准备发送的，也就是 TXE=0)，否则下一帧数据不被发出去。若 CTS 在传输期间被变成无效，当前的传输完成后停止发送。

当 CTSE=1 时，只要 CTS 输入一变换状态，硬件就自动设置 CTSIF 状态位。它表明接收器是否准备好进行通信。如果设置了 USART_CT3 寄存器的 CTSIE 位，则产生中断。下图是一个启用 CTS 流控制通信的例子。

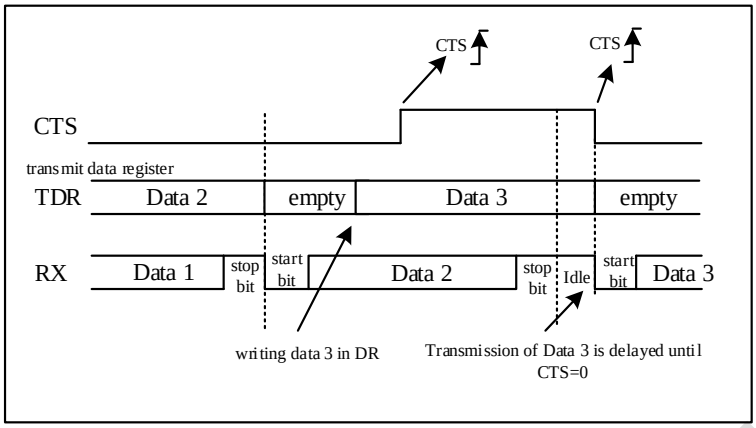


图 29-22 CTS 流控制

RS485 硬件流控

通过在 USART_CR3 控制寄存器中设置位 DEM 来启用，通过 DE(驱动使能)信号使用户能够激活外部收发器控制。断言时间是 DE 信号激活到起始位开始的时间。它是使用 USART_CR3 控制寄存器中的 DEAT[4:0]位域编程的。解除激活时间是传输数据中最后一个停止位结束到 DE 信号解除激活的时间。它是使用 USART_CR1 控制寄存器中的 DEDT[4:0]位域编程的。DE 信号的极性可以使用 USART_CR3 控制寄存器中的 DEP 位进行配置。

在 USART 中，DEAT 和 DEDT 以采样时间单位表示(1/8 或 1/16 位时间，取决于过采样率)。

29.4 中断请求

表 29-6 USART 中断请求

中断向量	标志位	使能位
USART	txe	txeie
	cts	ctsie
	tc	tcie
	rxne	rxneie
	idle	idleie
	pe	peie
	lbd	lbdie
	ne/ore/fe	eie
	txft	txftie
	rxft	rxftie
	rxff	rxffie
	txfe	txfeie
	rtof	rtofie
	eobf	eobfie

29.5 USART 寄存器

29.5.1 USART 状态寄存器（USART_SR)

地址偏移：0x00

复位值：0x0000 00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTOF	EOBF	Res.	TXFE
												RC_W0	RC__W0		R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXFF	RXFT	TXFT	ABRRQ	ABRE	ABRF	CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
R	R	R	W	R	R	RC_W0	RC__W0	R	RC_W0	RC_W0	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19	RTOF	RC_W0	0	已经超过在 RTOR 寄存器中编程的超时值后，如果无任何通信，此位由硬件置 1。 如果 USART_CR2 寄存器中 RTOIE=1，则会生成中断。 在智能卡模式下，该超时对应于 CWT 或 BWT 时间。 0：未达到超值值 1：已达到超时值，未接收到任何数据 注： 如果 RTOR 寄存器中编程的时间值将 2 个字符隔开，则 RTOF 不置 1。如果此时间大于该值 + 2 个采样时间（2/16 或 2/8，具体取决于过采样方法），则 RTOF 标志置 1。 即使 RE = 0，计数器仍会计数，但 RTOF 仅在 RE = 1 时置 1。如果 RE 置 1 时已经超时，则 RTOF 将置 1。 如果 USART 不支持接收器超时功能，该位保留且保持复位值。
18	EOBF	RC_W0	0	块结束标志 接收到完整块后，此位由硬件置 1（例如 T=1 智能卡模式）。 接收到的字节数（从块的起始处，包括起始字段）等于或大于 BLEN + 4 时执行检测。 0：未达到块结束 1：已达到块结束（字符数） 注： 如果不支持智能卡模式，该位保留且保持复位值。
17	Reserved	-	-	保留
16	TXFE	R	0	当 TXFIFO 为空时，该位由硬件置 1。当 TXFIFO 包含至少一个数据时，该标志被清零。也可以通过向 TXFRQ 写入 1 将 TXFE 标志置 1。 0： TXFIFO 非空。 1： TXFIFO 为空。
15	RXFF	R	0	当接收到的数据量对应于 RXFIFO 大小 + 1（RXFIFO 已满 + USART_RDR 寄存器中的 1 个数据）时，该位由硬件置 1。
14	RXFT	R	0	RXFT： RXFIFO 阈值标志 达到在 RXFTCFG 中编程的阈值时，该位由硬件置 1。这意味着，接收 FIFO 中有 (RXFTCFG - 1) 个数据， USART_RDR 寄存器中有一个数据。如果 USART_CR3 寄存器中的 RXFTIE 位 = 1（位 27），则会生成中断。 0：接收 FIFO 未达到编程的阈值。 1：接收 FIFO 已达到编程的阈值。
13	TXFT	R	0	TXFIFO 阈值标志 当 TXFIFO 达到在 TXFTCFG 中编程的阈值时（即 TXFIFO 包含 TXFTCFG 个位置），该位由硬件置 1。 0： TXFIFO 未达到编程的阈值。 1： TXFIFO 已达到编程的阈值。

12	ABRRQ	W	0	自动波特率请求。 该位写 1 会复位 ABRF 标志位，并且请求下一帧的自动波特率检测。
11	ABRE	R	0	自动波特率错误标志。 当自动波特率检测出错（波特率超出范围或者字符比较错误）时，硬件置位该寄存器。 软件通过写 1 到 ABRRQ 寄存器清零该位。
10	ABRF	R	0	自动波特率检测标志。 当自动波特率设置（同时设置 RXNE=1，当中断使能后产生中断），或者自动波特率检测操作出错（ABRE=1，RXNE=1，FE=1）时该位由硬件置 1. 软件通过写 1 到 ABRRQ 位清零该位。
9	CTS	RC_W0	0	CTS: CTS 标志 如果设置了 CTSE 位，当 CTS 输入变化状态时，该位被硬件置高。由软件将其清零。如果 USART_CR3 中的 CTSIE 为'1'，则产生中断。 0： CTS 状态线上没有变化； 1： CTS 状态线上发生变化。
8	LBD	RC_W0	0	LBD: LIN 断开检测标志 当探测到 LIN 断开帧时，该位由硬件置'1'，由软件清'0'(向该位写 0)。如果 USART_CR3 中的 LBDIE = 1，则产生中断。 0：没有检测到 LIN 断开帧； 1：检测到 LIN 断开帧。 注意：若 LBDIE=1，当 LBD 为'1'时要产生中断
7	TXE	R	1	TXE:发送数据寄存器空(FIFO 禁用) TXE:TXFIFO 未满(FIFO 使能) 当 DR 寄存器中的数据被硬件转移到移位寄存器的时候，该位被硬件置位。如果 USART_CR1 寄存器中的 TXEIE 为 1，则产生中断。对 USART_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。
6	TC	RC_W0	1	发送完成标志 当包含有数据的一帧发送完成后，并且 TXE=1，由硬件将该位置为 1。如果 USART_CR1 寄存器中的 TXEIE 为 1，则产生中断。先读 SR,再对 USART_DR 的写操作，将该位清零。 0：发送还未完成； 1：发送完成。
5	RXNE	RC_W0	0	读数据寄存器非空标志 当 RDR 移位寄存器中的数据被转移到 USART_DR 寄存器中，该位被硬件置位。如果 USART_CR1 寄存器中的 RXNEIE 为 1，则产生中断。对 USART_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方式。 0：数据也没有收到 1：收到数据，可以读出

4	IDLE	R	0	监测到总线空闲标志 当监测到总线空闲时，该位被硬件置位。如果 USART_CR1 中 IDLEIE 为 1，则产生中断。由于软件序列清除该位（先读 USART_SR;然后读 USART_DR）。 0：没有检测到空闲总线； 1：检测到空闲总线 注意： IDLE 位不会再次被置高直到 RXNE 位被置起(即又检测到一次空闲总线)
3	ORE	R	0	过载错误标志 当 RXNE=1 时，当前被接收在移位寄存器中的数据，需要传送到 DR 寄存器时，硬件将该位置位。如果 USART_CR1 中的 RXNEIE 则产生中断。由软件序列将其清零(先读 USART_SR，然后读 USART_DR)。 0：没有过载错误； 1：检测到过载错误。 注意：该位被置位时，DR 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，在多缓冲区通信模式下，ORE 标志置位会产生中断。
2	NE	R	0	NE: 噪声错误标志 在接收到的帧检测到噪音时，由硬件对该位置位。由软件序列对其清零(先读 USART_SR，再读 USART_DR)。 0：没有检测到噪声； 1：检测到噪声。 注意：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
1	FE	R	0	FE:帧错误标志 当检测到同步错位，过多的噪声或者检测到断开帧，该位被硬件置位。由软件序列将其清零(先读 USART_SR,再读 USART_DR)。 0:没有检测到帧错误； 1：检测到帧错误或者断开帧。 注意：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。 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。
0	PE	R	0	PE:校验错误标志 在接收模式下，如果出现奇偶校验错误，硬件对该位置位。由软件序列对其清零（依次读 USART_SR 和 USART_DR）。在清除 PE 位前，软件必须等待 RXNE 标志位被指 1，如果 USART_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。

29.5.2 USART 发送数据寄存器（USART_DR）

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	DR	RW	9’h0	包含要传输的数据字符。当启用奇偶校验(在 USART_CR1 寄存器中 PCE 位设置为 1)时，写在 MSB 中的值(根据数据长度，第 7 位或第 8 位)没有影响，因为它被奇偶校验所取代

29.5.3 USART 波特率寄存器（USART_BRR）

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV_Mantissa[11:0]												DIV_Faction[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:4	DIV_Mantissa	RW	12’h0	12bit 整数 这 12 个位用于定义 USART 除数 (USARTDIV) 的整数
3:0	DIV_Fraction	RW	4’h0	4bit 小数 这 4 个位用于定义 USART 除数 (USARTDIV) 的小数。

29.5.4 USART 控制寄存器 1（USART_CR1）

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
FIFOEN	TXFRQ	RXFRQ	Res.	DATAINV	TX_INV	RX_INV	RXFTCFG[2:0]			TXFTCFG[2:0]			SWAP	ADDM7	Res.
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**
MSB	M1	UE	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	FIFOEN	RW	0	FIFO 模式使能： 0：FIFO 模式关闭 1：FIFO 模式打开 FIFO 模式可用于标准 UART 通信，SPI 主模式和智能卡模式。它不能在 IrDA 和 LIN 模式下启用.

				该位只能在 USART 未使能时配置（UE=0）
30	TXFRQ	RW	0	TXFRQ：发送数据刷新请求 禁止 FIFO 模式时，向该位写入“1”会将 TXE 标志置 1。这可能会丢弃据。由于错误(NACK) 而未发送数据且 USART_SR 寄存器中的 FE 标志有效时，只能在智能卡模式下使用此位。如果 USART 不支持智能卡模式，该位保留且必须保持复位值。 使能 FIFO 时， TXFRQ 位置 1 以清空整个 FIFO。这会将 TXFE 标志置位。在 UART 模式和智能卡模式下都支持清空发送 FIFO。
29	RXFRQ	RW	0	接收数据刷新请求 写 1 来清除整个接收的 FIFO。 这个使能是不读数据但是会丢弃所有 FIFO 接收数据，避免出现过载情况
28	Reserved	-	-	保留
27	DATAINV	RW	0	二进制数据反向。 0: 按正/正向逻辑发送/接收数据寄存器中的逻辑数据（1=H，0=L） 1: 按负/反向逻辑发送/接收数据寄存器中的逻辑数据（1=L，0=H） 注：奇偶校验位同样按此定义翻转。
26	TX_INV	RW	0	TX 引脚电平反转 0：TX 引脚信号工作使用标准逻辑电平 1：TX 引脚信号值取反 这使得在 TX 线路上使用外部逆变器成为可能。 该位只能在 USART 未使能时配置（UE=0）
25	RX_INV	RW	0	RX 引脚电平反转 该位只能在 USART 被禁用(UE = 0)时写入。 该位由软件设置和清除 0：RX 引脚信号使用标准逻辑电平工作 1：RX 引脚信号值取反 这使得在 RX 线路上使用外部逆变器成为可能。 该位只能在 USART 未使能时配置（UE=0）。
22:24	RXFTCFG[2:0]	RW	3’h0	RXFIFO 阈值配置 000：接收 FIFO 到达深度的 1/8 001：接收 FIFO 到达深度的 1/4 010：接收 FIFO 到达深度的 1/2 011：接收 FIFO 到达深度的 3/4 100：接收 FIFO 到达深度的 7/8 101：接收 FIFO 满 其他:保留
19:21	TXFTCFG[2:0]	RW	3’h0	TXFIFO 阈值配置 000：TXFIFO 达到其深度的 1/8 001：TXFIFO 达到其深度的 1/4 010：TXFIFO 达到其深度的 1/2 011：TXFIFO 达到其深度的 3/4 100：TXFIFO 达到其深度的 7/8 101：TXFIFO 变空

				其他：保留
18	SWAP	RW	0	交换 TX/RX 引脚 0：按标准引脚排列定义使用 TX/RX 引脚 1：交换 TX 和 RX 引脚功能。这使得可以在与另一个 UART 交叉连接的情况下工作。 该位只能在 USART 被禁用(UE = 0)时写入。
17	ADDM7	RW	0	7 位/4 位地址检测 0: 4 位地址检测 1:7 位地址检测(8 位数据模式下) 该位只能在 USART 未使能时配置（UE=0）
16	Reserved	-	-	保留
15	MSB	RW	0	高位有效在前 0：在起始位之后，首先使用数据位 0 发送/接收数据。 1：在起始位之后，首先使用 MSB(位 7/8)发送/接收，数据。 该位只能在 USART 被禁用(UE = 0)时写入。
14	M1	RW	0	该位和第 12 位 M0 组成 M 决定数据大小, M： 00:8 位 01:9 位 10:7 位 该位只能在 USART 未使能时配置（UE=0） 在 7 位数据长度模式下，不支持智能卡模式、LIN 主模式。
13	UE	RW	0	USART 使能 当该位被清零，在当前字节传输完成后 USART 的分频器和输出停止工作，以此减少功耗。该位由软件置位和清零。 0：USART 分频器和输出被禁止； 1：USART 模块使能。
12	M0	RW	0	M0：字长 该位定义了数据字的长度，由软件对其设置和清零。 该位只能在 USART 未使能时配置（UE=0）
11	WAKE	RW	0	唤醒的方法 0：被空闲总线唤醒 1：被地址标记唤醒 该位只能在 USART 未使能时配置（UE=0）
10	PCE	RW	0	PCE:校验控制使能 0：禁止校验控制 1：使能校验控制 该位只能在 USART 未使能时配置（UE=0）
9	PS	RW	0	奇偶校验选择 0：偶校验 1：奇校验 该位只能在 USART 未使能时配置（UE=0）
8	PEIE	RW	0	PE 中断使能 0：禁止 PE 中断

				1：使能 PE 中断
7	TXEIE	RW	0	TXE 中断使能 0：禁止 TXE 中断 1：使能 TXE 中断
6	TCIE	RW	0	TC 中断使能 0：禁止 TC 中断 1：使能 TC 中断
5	RXNEIE	RW	0	RXNE 中断使能 0：禁止 RXNE 中断 1：当 USART_SR 中的 ORE 或者 RXNE 为 1 时，产生 USART 中断
4	IDLEIE	RW	0	IDLE 中断使能 0：禁止 IDLE 中断 1：使能 IDLE 中断
3	TE	RW	0	发送使能 0：禁止发送 1：使能发送 在数据传输过程中，除了在智能卡模式下，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"(空闲帧)。当 TE 被设置后，在真正发送开始之前，有一位时间的延迟。
2	RE	RW	0	接收使能 0：禁止接收 1：使能接收
1	RWU	RW	0	接收唤醒 该位用来决定是否把 USART 置于静默模式。该位由软件设置或清除。当唤醒序列到来时，硬件也会将其清零。 0：接收器处于正常工作模式； 1：接收器处于静默模式。 在把 USART 置于静默模式(设置 RWU 位)之前，USART 已经先接收了一个数据字节。否则在静默模式下，不能被空闲总线检测唤醒。当配置成地址标记检测唤醒(WAKE 位=1)，在 RXNE 位被置位时，不能用软件修改 RWU 位
0	SBK	RW	0	发送断开帧 软件置位该寄存器，发送断开帧。断开帧的停止位发送后，硬件清零该寄存器。 0：不发送断开帧 1：发送断开帧

29.5.5 USART 控制寄存器 2（USART_CR2）

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
Res.	Res.	Res.	Res.	Res.	Res.	RTOEN	DEP	DEM	ADD[7:1]						
						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
ADD[0]	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	Res	LBDIE	LBDL	Res.	Res.	Res.	Res.	Res.
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW					

Bit	Name	R/W	Reset Value	Function
31：26	Reserved	-	-	保留
25	RTOEN	RW	0	接收器超时使能 此位由软件置 1 和清零。 0：禁止接收器超时功能 1：使能接收器超时功能 使能此功能后，如果 RX 线路在 RTOR（接收器超时寄存器）中编程的持续时间内处于空闲状态（无接收），则 USART_SR 寄存器中的 RTOF 标志置 1。
24	DEP	RW	0	驱动使能极性选择 0：DE 信号为高 1：DE 信号为低 该位只能在 USART 未使能时配置（UE=0）。
23	DEM	RW	0	驱动使能模式： 0:DE 禁止 1:DE 使能 该位只能在 USART 未使能时配置（UE=0）
22:15	ADD	RW	8’h0	USART 地址 地址标记检测时的地址。 在 4 位地址检测时，只有 ADD[3:0]被使用 该位只能在 USART 未使能时配置（UE=0）
14	LINEN	RW	0	LIN 模式使能。 0：LIN 模式关闭 1：LIN 模式使能 LIN 模式下，通过使能 SBK 位，发送 LIN 同步断开帧（13 位），以及检测断开帧。 该位只能在 USART 未使能时配置（UE=0）。
13：12	STOP	RW	2’h0	Stop 位配置。 00：1 个停止位 01：0.5 个停止位 10：2 个停止位 11：1.5 个停止位 该位只能在 USART 未使能时配置（UE=0）
11	CLKEN	RW	0	时钟使能，该位用来使能 CK 引脚。 0：CK 引脚禁止 1：CK 引脚使能 该位只能在 USART 未使能时配置（UE=0）
10	CPOL	RW	0	时钟极性 0：总线空闲时 CK 引脚上保持低电平 1：总线空闲时 CK 引脚上保持高电平 该位只能在 USART 未使能时配置（UE=0）

9	CPHA	RW	0	时钟相位 0：在时钟的第一个边沿进行数据捕获 1：在时钟的第二个边沿进行数据捕获 该位只能在 USART 未使能时配置（UE=0）
8	LBCL	RW	0	最后一位时钟脉冲 在同步模式下，使用该位来控制是否在 CK 引脚上输出最后发送的那个字节（MSB）对应的时钟脉冲。 0：最后一位数据的时钟脉冲不在 CK pin 输出； 1：最后一位数据的时钟脉冲在 CK pin 输出。 注：最后一位数据位就是第八个或者第九个发送的位（由 USART_CR1 中的 M 位决定） 该位只能在 USART 未使能时配置（UE=0）
7	Reserved	-	-	保留
6	LBDIE	RW	0	LIN 断开帧中断使能。 0：禁止 1：产生
5	LBDL	RW	0	LIN 断开帧检测长度。 0：检测 10 位断开帧 1：检测 11 位断开帧
4:0	Reserved	-	-	保留

29.5.6 USART 控制寄存器 3（USART_CR3)

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
Res.	EOBIE	RTOIE	DEAT[4:0]					DEDT[4:0]					TXTFIE	RXFTIE	TXFEIE
	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**
RXFFIE	ABRMOD	Res.	ABREN	OVER8	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
RW	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	Reserved	-	-	保留
30	EOBIE	RW	0	EOB 中断使能。 0：禁止 1：中断使能
29	RTOIE	RW	0	RTO 中断使能。 0：禁止 1：中断使能
28:24	DEAT	RW	5’h0	起始位到 DE 拉高信号的时间。 以采样时间表示（1/8 位或 1/16 位时间，这个取决于过采样配置） 该位只能在 USART 未使能时配置（UE=0）
23:19	DEDT	RW	5’h0	最后一个数据的停止位到 DE 的时间。

				以采样时间表示（1/8 位或 1/16 位时间，这个取决于过采样配置） 该位只能在 USART 未使能时配置（UE=0）
18	TXFTIE	RW	0	TXFIFO 阈值中断使能 0：禁止中断 1：使能中断
17	RXFTIE	RW	0	RXFTIE：RXFIFO 阈值中断使能 0：禁止中断 1：使能中断
16	TXFEIE	RW	0	TXFEIE: TXFIFO 空中断使能 0：禁止中断 1：使能中断
15	RXFFIE	RW	0	RXFFIE:RXFIFO 满中断使能 0：禁止中断 1：使能中断
14	ABRMOD	RW	0	自动波特率检测模式。 0：从起始位开始测量波特率 1：下降沿到下降沿测量
13	Reserved	-	-	保留
12	ABREN	RW	0	自动波特率使能。 0：禁止 1：自动波特率使能 该位只能在 USART 未使能时配置（UE=0）
11	OVER8	RW	0	过采样模式。 0：16 位过采样 1：8 位过采样 该位只能在 USART 未使能时配置（UE=0）
10	CTSIE	RW	0	CTS 中断使能。 0：禁止中断 1：使能中断
9	CTSE	RW	0	CTS 使能。 0：CTS 硬件流控制禁止 1：CTS 模式使能 当 CTS 输入为 0 时，才会传输数据。此时，当数据写入数据寄存器后，要等待 CTS 有效后才会启动传输。 该位只能在 USART 未使能时配置（UE=0）
8	RTSE	RW	0	RTS 使能。 0：RTS 硬件流控制禁止 1：RTS 输出使能 只有当接收 buffer 未满时才会请求下一个数据。当前数据发送完成后，发送操作暂停。如果可以接收数据，将 RTS 置为有效（0）。 该位只能在 USART 未使能时配置（UE=0）
7	DMAT	RW	0	DMA 使能发送。

				0：禁止发送 DMA 1：使能发送 DMA
6	DMAR	RW	0	DMA 使能接收。 0：禁止接收 DMA 1：使能接收 DMA
5	SCEN	RW	0	智能卡模式使能。 0：禁止 1：使能 该位只能在 USART 未使能时配置（UE=0）
4	NACK	RW	0	智能卡 NACK 使能。 0：奇偶校验错误时发送 NACK 禁止 1：奇偶校验错误时发送 NACK 使能
3	HDSEL	RW	0	半双工选择。 0：非半双工模式 1：半双工模式 该位只能在 USART 未使能时配置（UE=0）
2	IRLP	RW	0	IrDA 低功耗。 0：正常模式 1：低功耗模式 该位只能在 USART 未使能时配置（UE=0）
1	IREN	RW	0	IrDA 模式使能。 软件使能和清零该寄存器。 0：IrDA 禁止 1：IrDA 使能 该位只能在 USART 未使能时配置（UE=0）
0	EIE	RW	0	错误中断使能。 0：禁止 1：帧错误(FE)、overrun 错误(ORE)、噪声(NE)中断使能

29.5.7 USART 保护时间和预分频（USART_GTPR）

地址偏移：0x18

复位值：0x0000 0000

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

Bit	Name	R/W	Reset Value	Function
31：16	Reserved	-	-	保留
15：8	GT	RW	8’h0	保护时间值 以波特时钟为单位的保护时间。 在智能卡模式，需要此功能。 注：该位只能在 USART 未使能时配置（UE=0）
7：0	PSC	RW	8’h0	预分频器值 在红外(IrDA)低功耗模式下： PSC[7:0] =红外低功耗波特率。

				对系统时钟分频以获得低功耗模式下的频率： 源时钟被寄存器中的值(8 位有效)分频 00000000：保留 00000001：对源时钟 1 分频 00000010：对源时钟 2 分频 …… 1. 在红外(IrDA)正常模式下： PSC 只能设置为 00000001。 2. 在智能卡模式下： PSC[4:0]：预分频值 对系统时钟进行分频，给智能卡提供时钟。 寄存器中给出的值(低 5 位有效)乘以 2 后，作为对源时钟的分频因子。 00000：保留 00001：对源时钟进行 2 分频 00010：对源时钟进行 4 分频 00011：对源时钟进行 6 分频 …… 注：位[7:5]在智能卡模式下没有意义。 注：该位只能在 USART 未使能时配置（UE=0）

29.5.8 USART 接收超时寄存器（USART_RTOR）

地址偏移：0x1C

复位值：0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLEN[7:0]								RTO[23:16]							
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**
RTO[15:0]															
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31：24	BLEN	RW	8’h0	块长度： 这是智能卡模式下 T=1 接收的块长度
23:0	RTO	RW	24’h0	RX 接收超时时间 在标准模式下，在最后接收达到的字符后在设定时间内没有检测到起始位，RTOF 标志位置起。 在智能卡模式下，这个值用来执行 CWT 和 BWT。 在这个情况下，CWT 和 BWT 是从最后接收到的字符起始位开始计算的

RTOR 可以动态编写。如果新值小于或等于计数器，则设置 RTOF 标志。

30. 通用异步收发器(UART)

30.1 UART 简介

UART 是一种可编程通用异步收发器。

30.2 UART 主要特征

- AMBA APB 接口
- 支持 5/6/7/8/9 位串行数据
- 支持 1/2 位 STOP 位（5 位数据时：1/1.5 位 STOP）
- 支持发送地址/数据
- 支持固定奇偶校验
- 支持断开帧
- 起始位错误检测
- 支持可编程分数波特率：可编程串行数据波特率，计算如下：波特率=（串行时钟频率）/（16*除数）
- 支持 4 位小数波特率
- 支持 Tx/Rx 引脚互换功能
- 支持大小端切换 MSB FIRST 功能
- 支持 DMA 传输

30.3 UART 功能描述

30.3.1 UART 框图

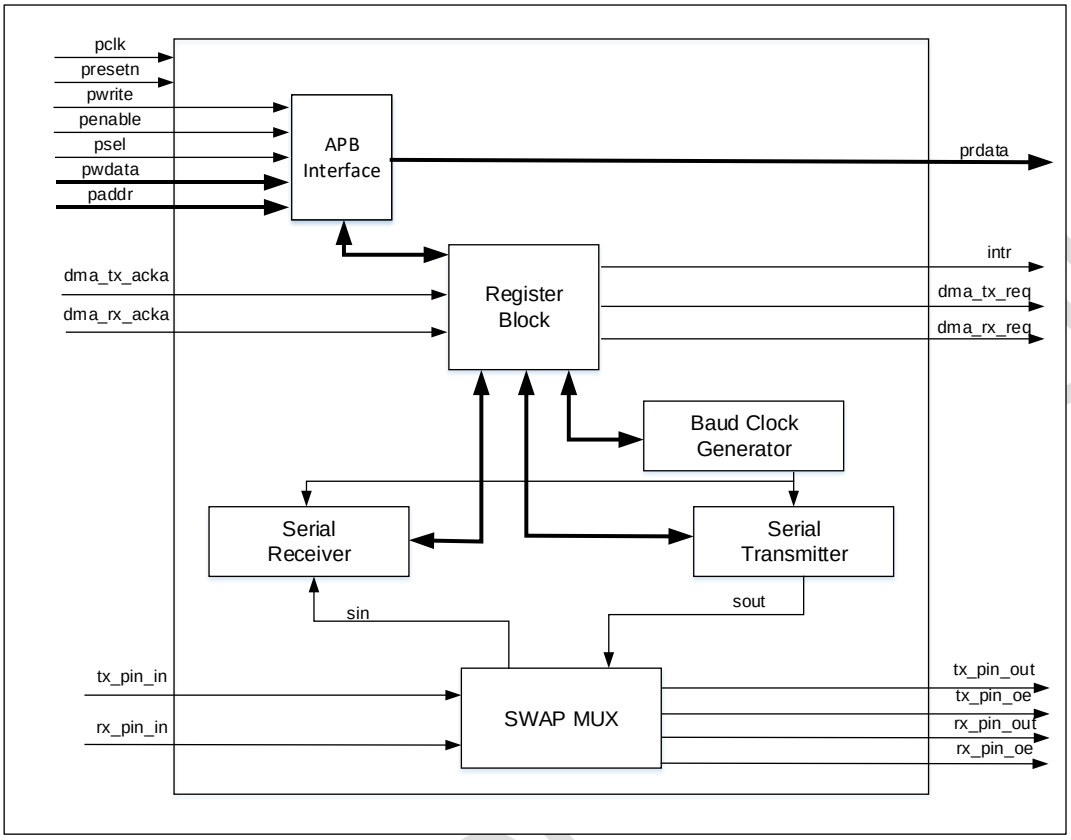


图 30-1 UART 框图

注：1.粗体线表示存在且为总线。

主要模块：

- APB 从接口（APB interface）---APB 总线接口。
- 寄存器块（Register Block）---负责 UART 的主要功能，包括控制、状态和中断生成。
- 波特时钟生成器（Baud Clock Generator）---生成发送器和接收器波特时钟以及输出参考时钟信号。
- 串行发送器（Serial Transmitter）---将写入 UART 的并行数据转换为串行形式，并添加控制寄存器指定的所有附加位，用于传输。
- 串行接收器（Serial Receiver）---将 UART 中接收的控制寄存器指定的串行数据字符转换为并行形式。此块控制：
  - 奇偶校验错误检测
  - 帧错误检测
  - 断开帧检测
- 引脚互换处理（SWAP MUX）---SWAP 使能时的引脚处理

30.3.2 UART（RS232）串行协议

因为 UART 和所选设备之间的串行通信是异步的，所以在串行数据中添加了额外的位（开始和停止）来指示开始和结束。利用这些比特可以使两个设备同步。这种由起始位和停止位组成的串行数据结构被称为一个字符，如下图所示。

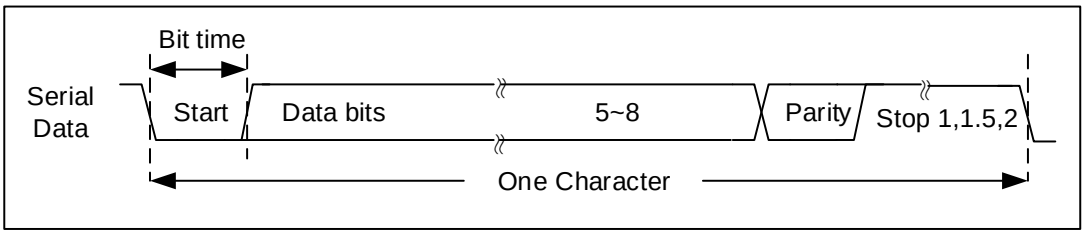


图 30-2 串行数据格式

一个额外的奇偶校验位可以添加到串行字符。该位出现在字符结构中的最后一个数据位之后和停止位之前，以便为 UART 提供对接收到的数据执行简单错误检查的能力。

UART 线路控制寄存器用于控制串行字符特性。数据字的各个位在起始位之后发送，从最低有效位 (LSB) 开始。它们后面是可选的奇偶校验位，后面是停止位，可以是 1、1.5 或 2。

注：UART 实现的停止位持续时间可能会更长，原因如下：

在某些配置的字符之间插入的空闲时间

传输中的所有比特都是在完全相同的持续时间内传输的；这方面的例外是当使用 1.5 个停止位时的半停止位。该持续时间被称为比特周期或比特时间；一位时间等于十六个波特时钟。

为了确保线路的稳定性，一旦检测到起始位，接收器就在位时间的大约中点对串行输入数据进行采样。因为每个比特传输的波特时钟的确切数量是已知的，所以计算采样的中点并不困难；即在起始位的中点采样之后每十六个波特时钟。

与串行输入去抖动一起，这种采样有助于避免错误起始位的检测。短故障通过去抖动被过滤掉，并且在线路上没有检测到转换。如果故障足够宽，可以通过去抖动来避免滤波，则会检测到下降沿。然而，只有当线路在半个采样周期后再次被采样为低时，才检测到起始位。

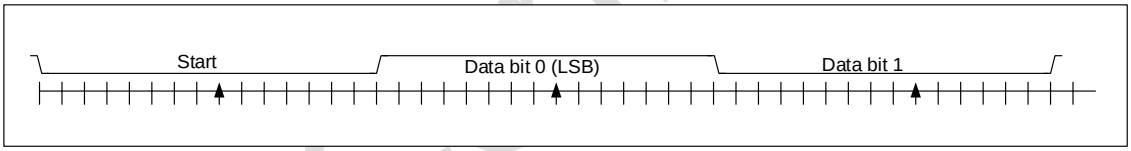


图 30-3 接收器串行数据采样点

作为 16550 标准的一部分，一个可选的波特时钟参考输出信号（baudout_n）为需要它的接收设备提供定时信息。UART 的波特率由单时钟实现中的串行时钟 pclk 以及分频锁存寄存器（BRR）控制。

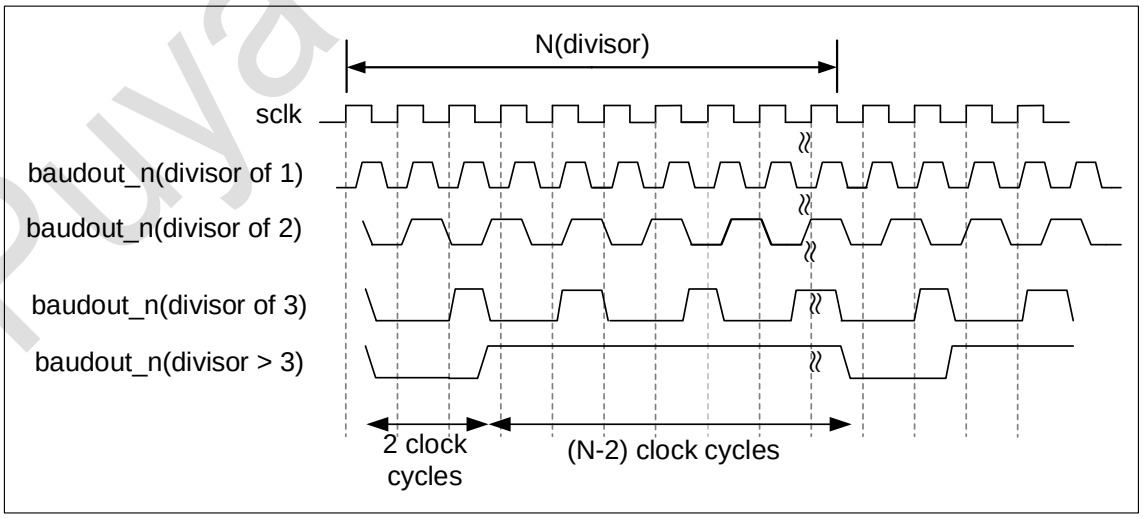


图 30-4 不同除数值的baudout_n输出的时序图

30.3.3 UART 9 位数据传输

UART 在发送和接收模式下都可以被配置为具有 9 位数据传输。字符中的第 9 位出现在字符的第 8 位之后和奇偶校验位之前。下图显示了字符的串行传输，其中 D8 表示第 9 位，还显示了 9 位模式下的一般串行传输。（此为正常小端模式，如果是大端模式则调转 9 位数据位的顺序）

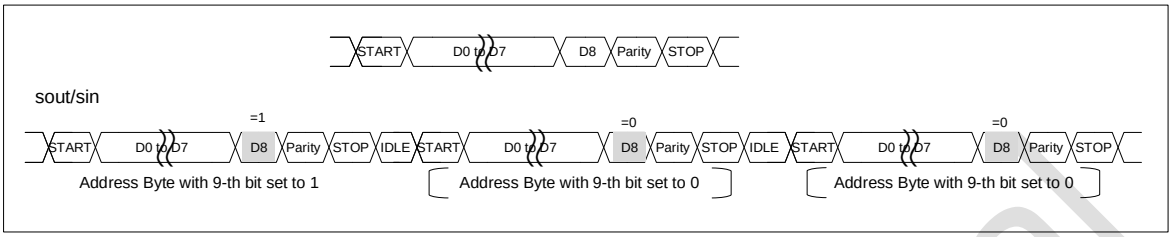


图 30-5 9 位字符

通过启用 9 位数据传输模式，UART 可以用于多点系统，其中一个主机连接到系统中的多个从机。主机与其中一个从机交流。当主机想要将数据块传输到从机时，它首先发送一个地址字节来识别目标从机。地址/数据字节之间的区分是基于输入字符中的第 9 位来完成的。如果第 9 位设置为 0，则该字符表示一个数据字节。如果第 9 位设置为 1，则该字符表示地址字节。所有从系统将地址字节与它们自己的地址进行比较，并且只有目标从机（其中地址匹配）能够从主机接收数据。主机开始向目标从机发送数据字节。未寻址的从机忽略传入的数据，直到接收到新的地址字节。

在上图中，请注意一个地址后面跟着 2 个数据字节。地址字节在第 9 位（D8）设置为 1 的情况下输出，而数据字节在第九位（D8）设置为 0 的情况下发出。奇偶校验位是一个可选字段。

用于 9 位数据传输的 UART 的配置执行以下操作：

- UART_CR3.M_E 位用于启用或禁用 9 位数据传输。
- 在接收的情况下，UART_CR3.ADDR_MATCH 用于在基于硬件和软件的地址匹配之间进行选择。
- UART_CR3.SEND_ADDR 位用于在发送的情况下使能发送地址。
- UART_CR3.TX_MODE 位用于在基于硬件和软件的地址传输之间进行选择。
- UART_TAR 和 UART_RAR 寄存器分别用于发送地址和匹配接收到的地址。
- UART_DR（TDR/RDR）寄存器为 9 位寄存器，用于在 9 位模式下进行数据传输。
- UART_SR.ADDR_RCVD 位用于指示地址接收中断。

30.3.3.1 发送模式

UART 支持两种类型的传输模式：

- 传输模式 0（当 UART_CR3.TX_MODE 设置为 0 时）
- 传输模式 1（当 UART_CR3.TX_MODE 设置为 1 时）

1. 模式 0：

在传输模式 0 中，地址被编程在传输地址寄存器（TAR）中，数据被写入传输保持寄存器（TDR）。TDR 寄存器的第 9 位在此模式中不适用。

下图说明了基于 SEND_ADDR（CR3[2]）、TDR 空条件的地址和数据传输。

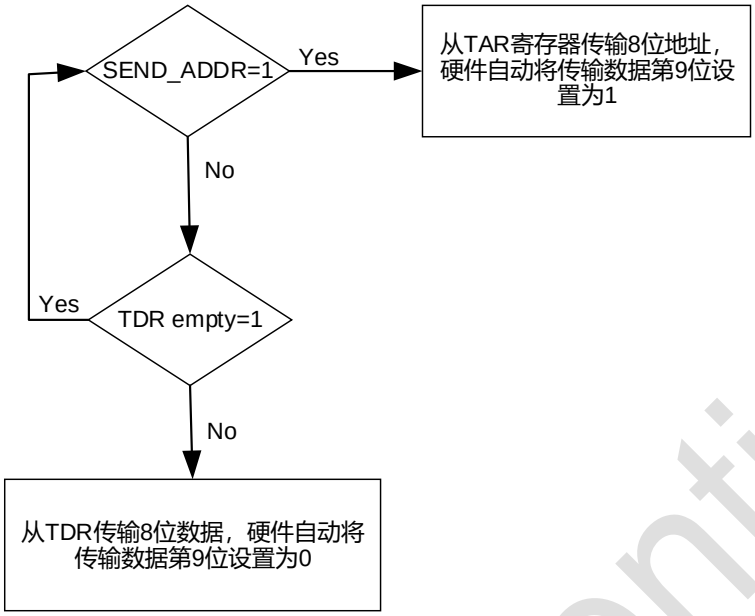


图 30-6 自动地址传输流程图

要向其传输数据的目标从机地址被编程在 TAR 寄存器中。必须启用 SEND_ADDR（CR3[2]）位来传输串行 UART 线上 TAR 寄存器中的目标从机地址，其中第 9 个数据位设置为 1，表示正在发送地址到从机。地址字符开始在线上传输后，UART 清除 SEND_ADDR 位。

传输到目标从机所需的数据通过传输保持寄存器（TDR）进行编程。数据在 UART 线上传输，第 9 个数据位设置为 0，表示正在发送数据到从机。

2. 模式 1：

在传输模式 1 中，TDR 寄存器为 9 位宽，地址和数据都通过 TDR 寄存器编程。UART 不区分地址和数据。SEND_ADDR（UART_CR3[2]）位和传输地址寄存器（UART_TAR）不适用于此模式。根据是否必须发送地址/数据，软件必须用 1/0 写入第 9 位。

表 30-1 发送配置

M_E	TX_MODE	SEND_ADDR	使用场景
0	X	X	发 8 位 TDR 发数据
1	0	0	发“0+8 位 TDR 数据“
1	0	1	发“1+8 位 TAR 地址”
1	1	X	发 “9 位 TDR 数据”

注：该表说明作为发送方可以使用的几种发送场景和对应的配置。

30.3.3.2 接收模式

UART 支持两种接收模式：

- 硬件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 1 时）
- 软件地址匹配接收模式（当 ADDR_MATCH（UART_CR3[1]）设置为 0 时）

1. 硬件地址匹配接收模式：

在硬件地址匹配接收模式中，如果接收字符的第 9 位设置为 1，则 UART 将接收字符与在接收地址寄存器（UART_RAR）中编程的地址进行匹配：

如果接收到的地址与 UART_RAR 寄存器中的编程地址匹配成功，则随后接收数据字节。

如果地址匹配失败，则 UART 控制器丢弃数据字符，直到接收到匹配的地址为止。

下图显示了基于地址匹配功能的数据字节接收流程图。

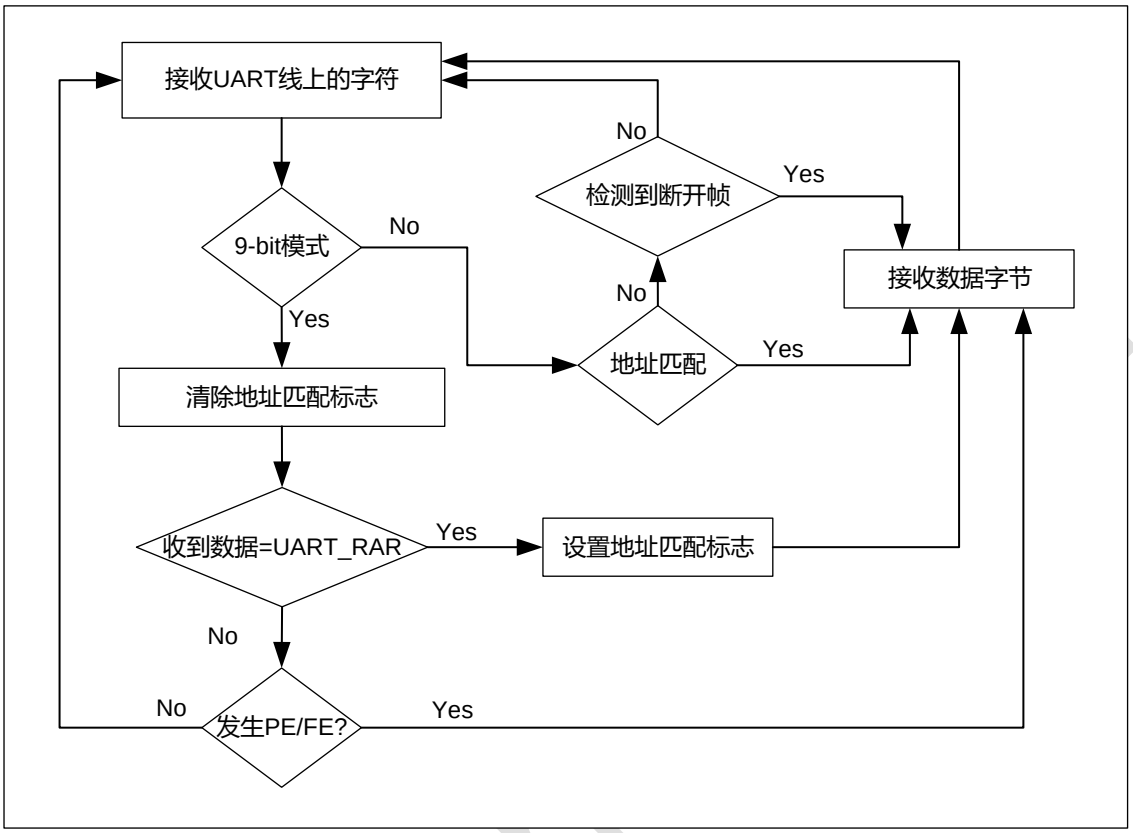


图 30-7 硬件地址匹配接收模式

UART 接收字符，而不管第 9 位数据是否设置为 1。如果接收到的字符的第 9 位被设置为 1，则它清除内部地址匹配标志，然后将接收到的 8 位字符信息与 UART_RAR 寄存器中编程的地址进行比较。

如果接收到的地址字符与 UART_RAR 寄存器中编程的地址匹配成功，则地址匹配标志被设置为 1，并且接收到的字符推送到 UART_RDR 寄存器，并且 UART_SR 寄存器中的 ADDR_RCVD 位被设置为指示地址已被接收，随后的数据字节（接收字符的第 9 位设置为 0）被推送到 UART_RDR。

如果地址与UART_RAR寄存器匹配失败并且(奇偶校验或者接收的地址字符中发现帧错误)的情况下，则接收的地址字符串仍然被推送到 UART_RDR 寄存器，ADDR_RCVD 和 PE/FE 错误位被设置为 1。

如果接收到任何中断字符，UART 将其视为一个特殊字符，并将其推送到 UART_RDR 寄存器，而不考虑地址匹配标志。

2. 软件地址匹配接收模式：

在这种操作模式中，UART 不执行与 UART_RAR 寄存器的接收地址字符（第 9 位数据设置为 1）的地址匹配。UART 始终接收 9 位数据，移位进 UART_RDR 寄存器。每当接收到地址字节并通过状态寄存器中的 ADDR_RCVD 位指示时，必须由用户自行比较地址。

30.3.3.3 UART 波特率

UART 的波特率由 PCLK 和分频锁存寄存器（UART_BRR）和小数波特率寄存器 UART_BRRF 控制。

波特率由以下因素决定：

- 串行时钟工作频率（PCLK）
- 波特率生成器除数值 divisor（由 UART_BRR 寄存器组成）
- 可接受的波特率误差

计算波特率的方程式如下：

波特率=串行时钟工作频率/（16* DIVISOR）——（1）

其中，DIVISOR–用于对 BRR 进行编程的数字（十六进制）。

串行时钟频率–UART PCLK 引脚处的频率。

根据等式（1），DIVISOR 可以计算为：DIVISOR=串行时钟频率/（16*波特率）

（等式算出的小数部分放在 UART_BRRF）

同样从等式（1）中，还可以示出：串行时钟频率= 波特率* 16* 除数

波特率和选定波特率定之间的误差如下所示：

百分比错误=（ |波特率-选定波特率| ）/波特率* 100%

表 30-2 设置波特率时的误差计算

波特率		fpclk = 4 MHz			fpclk = 24 MHz			fpclk = 48 MHz		
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%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.404	104	0.16%	2.4	625	0.00%	2.4	1250	0.00%
2	9.6	9.615	26	0.16%	9.615	156	0.16%	9.615	312	0.16%
3	19.2	19.231	13	0.16%	19.231	78	0.16%	19.231	156	0.16%
4	57.6	62.5	4	8.51%	57.692	26	0.16%	57.692	52	0.16%
5	115.2	125	2	8.51%	115.385	13	0.16%	115.385	26	0.16%
6	230.4	250	1	8.51%	250	6	8.51%	230.769	13	0.16%
7	460.8	不可能	不可能	不可能	500	3	8.51%	500	6	8.51%
8	921.6	不可能	不可能	不可能	1500	1	62.76%	1000	3	8.51%
9	2250	不可能	不可能	不可能	不可能	不可能	不可能	3000	1	33.33%
10	4500	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能	不可能

注:

1. CPU 的时钟频率越低，则某一特定波特率的误差也越低。可以达到的波特率上限可以由这组数据得到。

2.在配置时钟的时候，由于波特率时钟是在系统时钟上进行分频的，但得到的时钟频率与实际时钟频率会有误差。配置的时钟误差需要在 2%以内，才可以正常工作。

30.3.4 UART 接收容忍度

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

表 30-3 UART 接收容忍度

M 位	整数		小数	
	带帧错误	不带帧错误	带帧错误	不带帧错误
5	5.66%	6.67%	4.72%	5.56%
6	4.92%	5.66%	4.13%	4.72%
7	4.35%	4.92%	3.65%	4.10%
8	3.59%	3.90%	4.53%	4.58%

30.3.5 UART DMA 通信

UART 可以利用 DMA 连续通讯。Rx 缓冲器和 Tx 缓冲器的 DMA 请求分别产生。

30.3.5.1 DMA 发送

通过配置 UART_CR3.DMAT 使能利用 DMA 发送。当发送数据寄存器为空，DMA 就从指定的 SRAM 区传送数据到 UART_DR 寄存器。

DMA 发送的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 UART 使用的 DMA 通道
- 配置 UART_TDR 寄存器地址作为 DMA 传输的目标地址。在每个发送数据寄存器为空事件后，数据将被传送到这个地址
- 配置存储器（如 SRAM）地址作为 DMA 传输的源地址。在每次发送数据寄存器为空事件后，将从此存储器读数据加载到 UART_TDR 寄存器
- 写 DMA 控制寄存器配置传输字节数
- 写 DMA 寄存器配置通道优先级
- 使能 DMA 通道

当传输达到配置的传输字节数，DMA 控制器产生中断请求。

30.3.5.2 DMA 接收

通过配置 UART_CR3.DMAR 使能利用 DMA 接收。每收到一个字节，DMA 就把数据从 UART_DR 寄存器传送到指定的 SRAM 区。

DMA 接收的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 UART 使用的 DMA 通道
- 配置 UART_RDR 寄存器地址作为 DMA 传输的源地址。在每个接收数据寄存器非空事件后，数据将从此 UART_RDR 寄存器地址读出数据传输到存储器
- 配置 SRAM 存储器地址作为 DMA 传输的目标地址。在每次接收数据寄存器非空事件后，数据将从 UART_RDR 寄存器传送到此地址
- 写 DMA 控制寄存器配置传输字节数
- 写 DMA 寄存器配置通道优先级
- 使能 DMA 通道

当传输达到配置的传输字节数，DMA 控制器产生中断请求。

30.4 UART 中断

UART 中断输出信号（intr）的断言——只要几个优先中断类型中的一个被启用并激活，就会发生中断。

以下中断类型可以通过 UART_CR2 寄存器启用：

- 接收器数据可用（RXNEIE）
- 发送寄存器空（TXEIE）
- 发送保持寄存器为空（在可编程 TDR 中断模式下）（TDREIE）

- 忙错误检测指示（BUSYERRIE）
- 接收线路状态（LSIE）

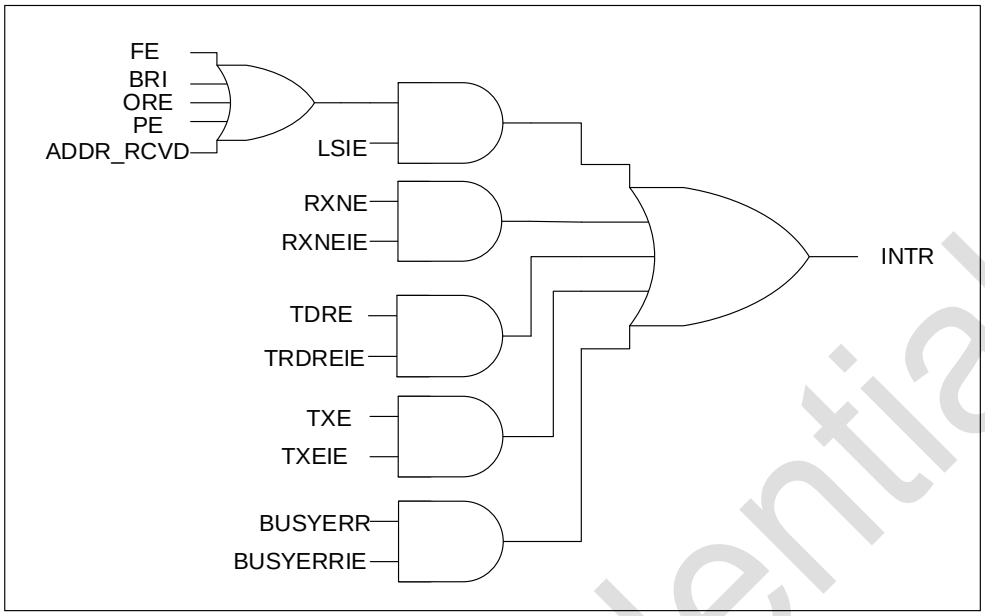


图 30-8 中断映射图

这些中断类型在下表中有详细说明。

表 30-4 中断控制功能

中断置位和清零功能		
中断类型	**中断源**	**中断清零控制**
接收线路状态	溢出/奇偶校验/帧错误中断或地址接收中断	对应位写 1 清 0
接收数据可用 RXNE	接收器数据可用	读取接收器缓冲寄存器
发送保持寄存器为空 TDRE	发送保持寄存器为空	写入 TDR
发送寄存器为空 TXE	发送寄存器为空	写入 TDR
忙错误检测 BUSYERR	UART 繁忙时（BUSY[0]设置为 1），主机尝试写入 CR1 寄存器。	对应位写 1 清 0

30.5 UART 寄存器

术语：设置=设置为 1；清除=清除为 0。

30.5.1 UART 数据寄存器 (UART_DR)

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留

8:0	DR[8:0]	RW	9'b0	接收/发送数据寄存器。 一共是由两个寄存器组成（一个是发送的 TDR,一个是接收的 RDR），所以 DR 寄存器实现了读和写的两个功能。 RDR 寄存器是 UART 模式下串行输入端口上接收到的数据字节。只有状态寄存器（SR）中的接收非空（RXNE）位被设置时，该寄存器中的数据才有效。必须在下一个数据到达之前读取 RDR 中的数据，否则它将被覆盖，从而导致溢出错误。 TDR 寄存器是在 UART 模式下串行输出端口上传输的数据。软件保证只有当 TDR 空（UART_SR.TDRE）位被设置时，数据才能再次写入 TDR。 如果 TDRE 被设置，则向 TDR 写入单个字符将清除 TDRE。在再次设置 TDRE 之前对 TDR 的任何额外写入都会导致 TDR 数据被重写。 仅当 UART_CR3.TX_MODE = 1 时，第 9 位才适用。

30.5.2 UART 波特率寄存器（UART_BRR)

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

Bit	Name	R/W	Reset Value	Function
31:16	Reserved	-	-	保留
15:0	BRR	RW	16'b0	该寄存器构成 16 位 divisor，divisor 包含 UART 的波特率除数。输出波特率等于串行时钟（PCLK）频率除以波特率除数值的十六倍，如下所示：波特率=（串行时钟频率）/（16*divisor）。请注意，当除数锁存寄存器（BRR）设置为零时，波特时钟被禁用，不会发生串行通信。 此外，一旦设置了 BRR，在发送或接收数据之前，应等待至少 8 个时钟周期。

30.5.3 UART 状态寄存器（UART_SR)

地址偏移：0x10

复位值：0x0000_0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	BUSY_ERR	BUSY	ADDR_RCVD	Res.	TXE	TDRE	BRI	FE	PE	ORE	RXNE
					RC_W1	R	RC_W1		R	O	RC_W1	RC_W1	RC_W1	RC_W1	R

Bit	Name	R/W	Reset Value	Function
31:11	Reserved	-	-	保留

10	BUSY_ERR	RC_W1	1’b0	忙检测错误，检测到忙时的误操作。 0：无忙误操作错误。 1：UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 该位写 1 清零。
9	BUSY	R	1'b0	UART 忙 这表示串行传输正在进行中，清除时表示 UART 处于空闲或非活动状态。 在以下任何一种情况下，此位都将设置为 1（忙）： ---串行接口上正在进行发送 ---串行接口上正在进行接收 ---传输 TDR 中存在的数据，波特除数为非零（BRR 不等于 0） ---接收 RDR 中存在数据 注意：即使可能从另一个设备发送了新字符，UART 忙位也可能被清除。也就是说，如果 UART 在 TDR 和 RDR 中没有数据，并且没有正在进行的传输，并且新字符的起始位刚刚到达 UART。这是由于直到位周期的中间才看到有效的开始，并且该持续时间取决于已编程的波特除数。 0（IDLE）：UART 处于空闲或非活动状态 1（忙）：UART 忙（主动传输数据）
8	ADDR_RCVD	RC_W1	1'b0	地址接收。 如果启用 9 位数据模式（UART_CR3.M_E = 1），则该位用于指示接收数据的第 9 位被设置为 1。该位还可以用于指示输入字符是地址还是数据。 0：表示字符为数据。 1：表示字符为地址。 注意：用户需要确保在下一个地址字节到达之前清除中断（该位写 1 清 0）。 如果清除中断有延迟，则软件将无法区分多个地址相关的中断。
7	Reserved	-	-	保留
6	TXE	R	1'b1	发送为空。 每当当前没有发送数据且 TDR 为空的时候，就会设置此位。如果 TXEIE 使能，则会产生 TXE 中断 0：发送不为空 1：发送为空
5	TDRE	R	1'b1	发送保持寄存器空。 该位指示 TDR 为空。 每当数据从 TDR 传输到发送器移位寄存器且没有新数据写入 TDR 时，该位被设置。如果 TDRIE 使能，则会产生中断。 0：TDR 非空 1：TDR 为空

4	BRI	RC_W1	1'b0	断开中断位 这用于指示在串行输入数据上检测到中断序列。 如果在 UART 模式下,每当串行输入保持在逻辑“0”状态的时间超过起始时间+数据位+奇偶校验位+停止位的总和时，就会设置它。对该位写 1 将清除 BRI 位。 0：未检测到断开序列 1：检测到断开序列
3	FE	RC_W1	1'b0	帧错误位。 这用于指示接收机中出现帧错误。当接收机在接收的数据中没有检测到有效的停止位时，就会发生帧错误。 应该注意的是，如果发生断开，也将设置帧错误（UART_SR.FE）位，如断开中断（UART_SR.BRI）位所示。之所以会发生这种情况，是因为断开字符通过将输入保持到逻辑 0 的时间长于字符的持续时间而隐式地生成帧错误。对该位写 1 将清除 FE 位。 0：无帧错误 1：帧错误
2	PE	RC_W1	1'b0	奇偶校验错误位。 如果设置了奇偶校验使能（UART_CR1.PEN）位，则该寄存器用于指示接收器发生奇偶校验错误。 应该注意的是，如果发生了断开，在这种情况下，如果奇偶校验生成和检测被启用（UART_CR1.PCE = 1）并且奇偶校验被设置为奇数（UART_CR1.PS = 0），则 PE 也会被设置。对该位写 1 将清除 PE 位。 0：无奇偶校验错误 1：奇偶校验错误
1	ORE	RC_W1	1'b0	溢出错误位。 这用于指示溢出错误的发生。 如果在读取以前的数据之前接收到新的数据字符，则会发生这种情况。 当新字符在从 RDR 读取前一个字符之前到达接收器时，设置 ORE 位。当这种情况发生时，RDR 中的数据将被覆盖。 对该位写 1 将清除 ORE 位。 0：无溢出错误 1：溢出错误
0	RXNE	R	1'b0	数据就绪位。 这用于指示接收器在 RDR 中至少包含一个字符。 读取 RDR 时，此位被清除。 0：数据未就绪 1：数据就绪

30.5.4 UART 控制寄存器 1 (UART_CR1)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	MSBFIRST	SWAP	Res.	SBK	SP	PS	PCE	STOP	M[1:0]	
						RW	RW		RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:10	Reserved	-	-	保留
9	MSBFIRST	RW	1'b0	最高有效位在前。 0：起始位后，收发第 0 位数据； 1：起始位后，收发第 5/6/7/8/9 位数据； 在数据传输过程中，不能修改这个位。
8	SWAP	RW	1'b0	TX/RX 引脚互换。 0：TX/RX 引脚按照标准引脚映射定义； 1: TX/RX 引脚互换。此时用作交叉连接其他 UART 时。
7	Reserved	-	-	保留
6	SBK	RW	1'b0	断开控制位。 这用于将断开发送到接收设备。如果设置为 1，串行输出将强制进入间隔（逻辑 0）状态。输出线被强制为低电平，直到 SBK 位被清除。 0：释放串行输出以进行数据传输 1：串行输出被迫进入间隔状态
5	SP	RW	1'b0	固定奇偶校验。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。此位用于强制固定奇偶校验值。当 PCE、PS 和 SP 设置为 1 时，奇偶校验位被发送并检查为逻辑 0。如果 PCE 和 SP 被设置为 1，并且 PS 是逻辑 0，则奇偶校验位被发送并被检查为逻辑 1。如果此位设置为 0，则 SP（固定奇偶校验）被禁用。 0：固定奇偶校验已禁用 1：固定奇偶校验已启用
4	PS	RW	1'b0	偶数奇偶校验选择。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当奇偶校验被启用（PCE 设置为 1）时，这用于在偶数和奇数奇偶校验之间进行选择。如果设置为 1，则传输或检查偶数个逻辑“1”。如果设置为零，则传输或检查奇数个逻辑“1”。 0：奇校验 1：偶校验
3	PCE	RW	1'b0	奇偶校验启用。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。该位用于控制发送和接收的串行字符中的奇偶校验生成和检测。 0：禁用奇偶校验 1：启用奇偶校验
2	STOP	RW	1'b0	停止位的数量。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。这

				用于选择外围设备将发送和接收的每个字符的停止位数。如果设置为零，则在串行数据中传输一个停止位。 如果设置为 1 并且数据位设置为 5（UART_CR1.M 设置为 0），则发送一个半停止位。否则，传输两个停止位。 注意，无论所选择的停止位的数量如何，接收器将仅检查第一个停止位。 注：由于某些配置的字符之间插入的空闲时间和传输方向上的波特时钟除数值，UART 实现的停止位持续时间可能会更长； 0：1 个停止位 1：1.5/2 个停止位
1:0	M[1:0]	RW	2'b0	数据长度选择。 仅当 UART 不忙时可写（UART_SR.BUSY 为 0）。当 UART_CR3 中的 M_E 设置为 0 时，此寄存器用于选择外围设备将发送和接收的每个字符的数据位数。 0x0：每个字符 5 个数据位 0x1：每个字符 6 个数据位 0x2：每个字符 7 个数据位 0x3：每个字符 8 个数据位

30.5.5 UART 控制寄存器 2 (UART_CR2)

地址偏移：0x18

复位值：0x0000_0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXEIE	BUSYERRIE	LSIE	TDREIE	RXNEIE
											RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	TXEIE	RW	1'b0	启用发送空中断。这用于启用/禁用传输发送空中断的生成。 0：禁用发送空中断 1：使能发送空中断
3	BUSYERRIE	RW	1’b1	启用 BUSYERR 状态中断。这用于启用/禁用 BUSYERR 状态中断生成。UART 繁忙时（UART_SR.BUSY 设置为 1），主机尝试写入 UART_CR1 寄存器。 0：禁用 BUSYERR 状态中断 1：使能 BUSYERR 状态中断
2	LSIE	RW	1'b0	启用接收器线路状态中断。这用于启用/禁用接收器线路状态中断的生成。 该中断标志是 PE、FE、ORE、BRI、ADDR_RCVD 中断标志的组合。 0：禁用接收器线路状态中断

				1：使能接收器线路状态中断
1	TDREIE	RW	1'b0	启用传输保持寄存器空中断。这用于启用/禁用传输保持寄存器空中断的生成。 0：禁用传输保持寄存器空中断 1：使能传输保持寄存器空中断
0	RXNEIE	RW	1'b0	启用接收数据可用中断。这用于启用/禁用接收数据可用中断。 0：禁用接收数据中断 1：使能接收数据中断

30.5.6 UART 控制寄存器 3 (UART_CR3)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMA_SOFT_ACK	DMAT	DMAR	TX_MODE	SEND_ADDR	ADDR_MATCH	M_E
									RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:7	Reserved	-	-	保留
6	DMA_SOFT_ACK	RW	1'b0	软件清除 DMA 发送请求和 DMA 接收请求。 0：不影响 1：清除 DMA 发送请求和 DMA 接收请求
5	DMAT	RW	1'b0	传送时使能 DMA。 0：禁止 1：使能发送 DMA
4	DMAR	RW	1'b0	接收时使能 DMA。 0：禁止 1：使能接收 DMA
3	TX_MODE	RW	1'b0	传输模式控制位。该比特用于控制 9-bit 数据传输期间的传输模式的类型。 数值： 1：在这种操作模式下，传输保持寄存器（TDR）为 9 位宽。 用户需要确保 TDR 寄存器的地址/数据写入正确。 地址：第 9 位设置为 1 数据：第 9 位数设置为 0 注意：传输地址寄存器（UART_TAR）不适用于此操作模式。 0：在此操作模式中，传输保持寄存器（TDR）的宽度为 8 位。 用户需要将地址编程到传输地址寄存器（UART_TAR）中，并将数据编程到 TDR 寄存器中。

2	SEND_ADDR	RW	1'b0	发送地址控制位。此位用作控制旋钮，供用户在传输模式下确定何时发送地址。 0：9 位字符内容来自 TDR 寄存器 1：9 位字符内容：第 9 位设置为 1，其余 8 位将与“传输地址寄存器”中正在编程的内容相匹配。 注： 1.在发出地址字符后，该位由硬件自动清除。用户不应将此位编程为 0。 2.此字段仅在 M_E 位设置为 1 且 TX_MODE 设置为 0 时适用。
1	ADDR_MATCH	RW	1'b0	地址匹配模式。此位用于在接收期间启用地址匹配功能。 ■ 在地址匹配模式下，UART 将等待第 9 位设置为 1 的传入字符。并进一步检查地址是否与“接收地址匹配寄存器”中编程的地址匹配。如果匹配，则后续字符将被视为有效数据，UART 开始接收数据。 ■ 在正常模式下，UART 将开始接收数据，9 位字符将形成。用户负责读取数据并区分 b/n 地址和数据。 0：正常模式 1：地址匹配模式 注：仅当 M_E 设置为 1 时，此字段才适用。
0	M_E	RW	1'b0	9 数据使能 此位用于启用用于发送和接收传输的 9 位数据 0：禁止 9 位数据 1：使能 9 位数据

30.5.7 UART 接收地址寄存器（UART_RAR)

地址偏移：0x20

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	RAR[7:0]	RW	8'b0	接收模式下的地址匹配寄存器。 如果在输入字符中第 9 位被设置为 1，则将对照该寄存器值检查剩余的 8 位。如果匹配成功，则第 9 位设置为 0 的后续字符将被视为数据字节，直到接收到下一个地址字节。 注： 1. 仅当“ADDR_MATCH”（UAR_CR3[1]）和“M_E”（UART_CR3[0]）位设置为 1 时，此寄存器才适用。 2. RAR 可以在任何时间点被编程。但是，当任何接收

				正在进行时，用户不得更改此寄存器值。

30.5.8 UART 发送地址寄存器（UART_TAR)

地址偏移：0x24

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:8	Reserved	-	-	保留
7:0	TAR[7:0]	RW	8'b0	这是传输模式下的地址匹配寄存器。 如果 M_E 位被启用，并且“SEND_ADDR”（UART_CR3[2]）位被设置为 1，则 UART 将发送第 9 位设置为 1 的 9 位字符，剩余的 8 位地址将从该寄存器发送。 注： 1. 此寄存器仅用于发送地址。正常数据应通过编程 TDR 寄存器发送。 2. 在 UART 串行通道上开始发送地址后，硬件将自动清除“SEND_ADDR”位

30.5.9 UART 波特率小数寄存器（UART_BRRF)

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BRRF[3:0]			
												RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	BRRF[3:0]	RW	4'b0	波特率小数部分。 小数值由（BRRF）/（2^4）确定。

31. 低功耗通用异步收发器（LPUART）

LPUART 是一种 UART，允许在有限功耗下双向 UART 通信。仅需 32.768 kHz LSE 时钟即可进行高达 9600 波特/s 的 UART 通信。当 LPUART 由与 LSE 时钟不同的时钟源驱动时，可以达到更高的波特率。即使当微控制器处于低功耗模式，能耗极低时，LPUART 也会等待 UART 帧的到来。LPUART 包含所有必要的硬件支持，使在最小功耗下可以进行异步串行通信。

它支持半双工单线通信和调制解调器操作(CTS/RTS)，还支持多处理器通信。

DMA（直接存储器访问）可用于数据发送/接收。

31.1 LPUART 主要特性

- AMBA APB 接口
- 全双工异步通讯
- NRZ 标准模式（标记/空格）
- 波特率可编程
- 32.768 kHz 时钟，波特率范围 300 ~ 9600 波特/s.更高波特率需要更高时钟频率支持
- 双时钟域：PCLK 及专用内核时钟
- 数据字长度可配置（7 位/8 位/9 位）
- 可编程的数据顺序，最先移位 MSB 或 LSB
- 停止位数可配置（1/2 位）
- 单线半双工通讯
- 支持 DMA 连续传输
- 传送和接收独立使能
- 独立发送/接收信号极性控制
- Tx/Rx 引脚可以互换
- 支持硬件 RS-485/modem 流控制
- 发送检测标志
  - 忙标志
  - 传输结束
- 奇偶校验控制
  - 发送时产生奇偶校验位
  - 接收时奇偶校验
- 4 个错误标志
  - 上溢错误
  - 噪声检测
  - 帧结构错误
  - 奇偶校验错误
- 中断源标志
  - CTS 改变
  - 发送寄存器空
  - 发送完成
  - 接收数据寄存器非空
  - 检测到总线空闲

- 溢出错误
- 帧错误
- 噪音操作
- 校验错误
- 地址字节匹配

■ 多处理器通讯：当地址不匹配时 LPUART 进入静默模式

■ 支持静默模式唤醒（空闲帧/地址标记检测）

31.2 LPUART 功能描述

31.2.1 LPUART 框图

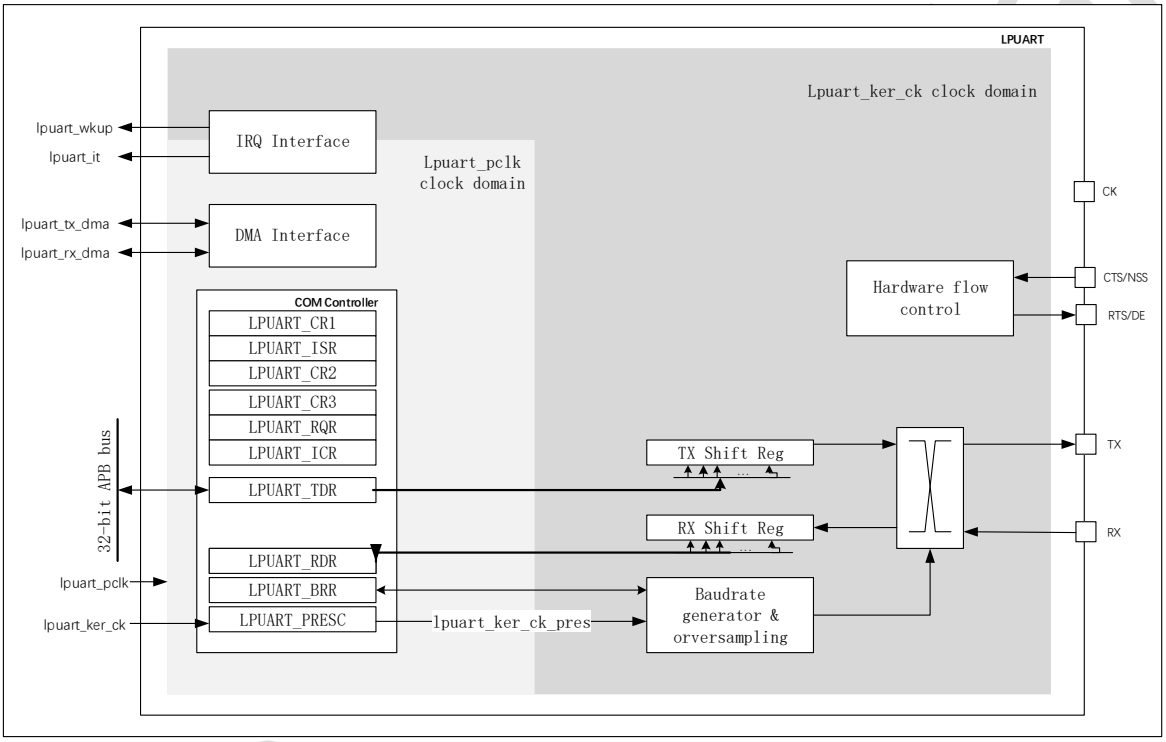


图 31-1 LPUART 框图

31.2.2 LPUART 信号

任何 LPUART 双向通信至少需要两个引脚：接收数据输入(RX)和发送数据输出(TX)。

■ RX：接收数据串行输入。

■ 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。当发送器使能，但不发送数据时，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。

硬件流控制模式

■ CTS：低电平开始发送，发送结束变为高电平

■ RTS：低电平表明 LPUART 准备好接收数据

RS485 硬件流控制模式

■ DE：驱动器使能，激活外部收发器的发送

注：DE 和 RTS 复用同一引脚

31.2.3 LPUART 字符描述

根据{M1，M0}配置，分为 7 位/8 位/9 位三种数据长度。

M[1:0]= 2’b10，7 位；

M[1:0]= 2’b00：8 位；

M[1:0]= 2’b01：9 位；

在默认情况下，信号（TX 或 RX）在起始位工作期间处于低电平状态。在停止位工作期间处于高电平状态。

通过极性配置控制，可以单独针对每个信号对这些值取反。

空闲字符定义：在 7 位/8 位/9 位数据长度+停止位时间内为全 1；

断开字符定义：在 7 位/8 位/9 位数据长度时间内为全 0；断开帧结尾，传送器插入 2 个停止位。即 8 位模式，断开帧长度为 10 位 0；9 位模式，断开帧长度为 11 位 0.

发送和接收操作由通用波特率发生器驱动。发送器和接收器的使能位置 1 时，将分别生成发送时钟和接收时钟。

下面是各个不同长度帧的波形。

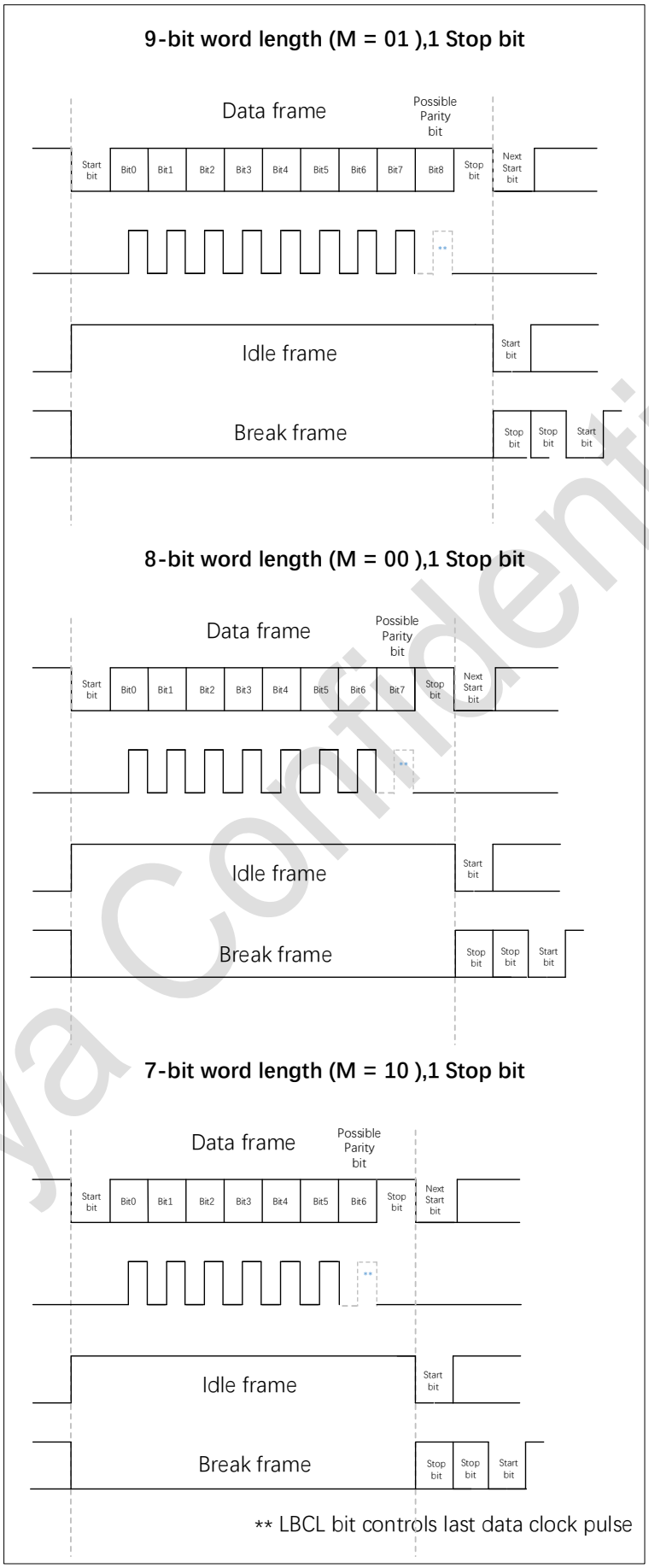


图 31-2 LPUART 字长编程

31.2.4 LPUART 发送

LPUART 发送步骤如下所述。

- 配置 LPUART_CR1.{M1,M0}，定义传输数据宽度（7/8/9 位）；
- 配置 LPUART_BRR 寄存器，选择波特率；
- 配置 LPUART_CR2.STOP 寄存器，定义停止位个数(1/2 位);
- 配置 LPUART_CR1.UE=1,使能 LPUART;
- 多处理器传输时，配置 LPUART_CR3.DMAT=1，使能 DMA;
- 配置 TE=1，硬件插入空闲帧时序;
- 写传输数据到 LPUART_TDR 寄存器（该操作清零 TXE 位）。该项操作可能重复多次。
- 写最后一个数据到 LPUART_TDR 后，软件等待 TC=1，表明最后一帧发送完成。
- 硬件产生发送时序：
  - 产生起始位，开始传送
  - 传送移位寄存器输出数据（默认 LSB）到 TX 引脚
  - 停止位（根据配置发送个数）
- 传送完最后 1 帧数据后，产生 TC=1（TXE=1 时）。当 TCIE=1，产生 TC 中断。

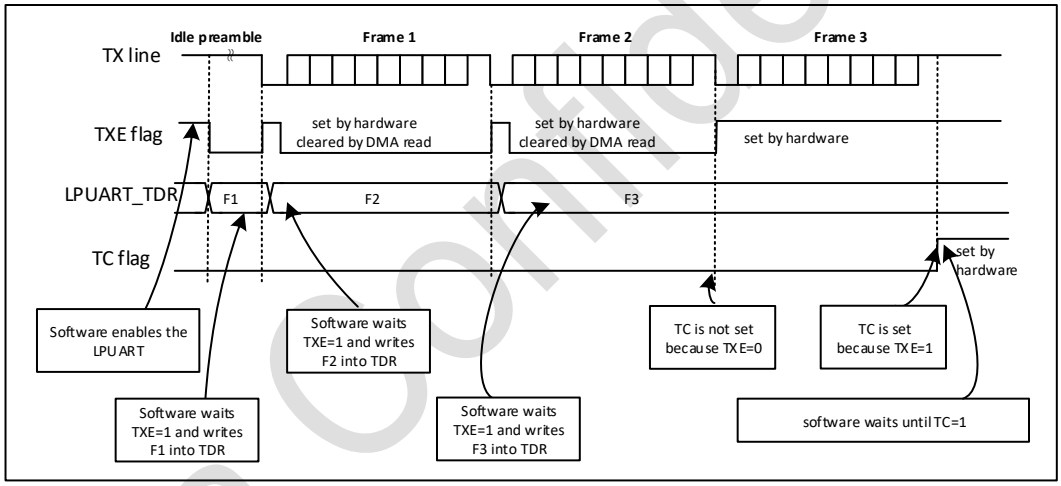


图 31-3 发送时的 TC/TXE 行为 TC/TXE 时序

特殊帧处理

- 断开帧

设置 SBKRQ=1 将会在当前传输完成后传送一个断开帧。断开帧的长度取决于{M1，M0}寄存器的配置。断开帧发送完毕后，硬件会清零 SBKRQ。

- 空闲帧

设置 TE=1，在第一个数据帧之前先发送一个空闲帧。

31.2.5 LPUART 接收

起始位检测

在 RX 上检测到下降沿后，硬件开始检测起始位。在该位的中间检测到 0，则判定该位为起始位。如果检测到的是 1，则置位噪声错误标志（NE），当前起始位无效，重新检测起始位。

数据字符接收

字符接收配置步骤：

- 配置 LPUART_CR1.{M1，M0}，选择接收长度；

- 配置 LPUART_BRR 寄存器，选择波特率；
- 配置 USAR_CR2.STOP，选择 1 位或者 2 位停止位；
- 配置 LPUART_CR1.UE=1，使能 LPUART；
- 多处理器传输时，配置 LPUART_CR3.DMAR=1，使能 DMA 接收;
- 配置 LPUART_CR1.RE，使能接收，开始检测起始位；

当接收到字符后：

- 非 FIFO 模式，置位 RXNE。表明移位寄存器的内容已被转移到 RDR，即数据可以被读出；
- 当 RXNEIE=1 时，产生中断；
- 在接收时，如果检测到帧错误、噪声错误或溢出错误，则设置相应错误标志位；
- 多处理器通讯：
  - 非 FIFO 模式，RXNE 在每个字节接收后被置位，并由 DMA 对接收数据寄存器的读操作而清零；
- 在单缓冲器模式：
  - 非 FIFO 模式，软件读 LPUART_RDR 寄存器对 RXNE 位清零。RXNE 标志也可以通过写 LPUART_RQR.RXFRQ=1 清零。RXNE 位必须在下一字符接收结束前被清零，以免溢出错误；

特殊帧及错误处理

- 断开字符

当接收到一个断开帧时，LPUART 按错误帧处理。

- 空闲字符

当检测到空闲帧时，按正常帧处理，如果 IDLEIE=1，则产生中断。

- 上溢错误

如果 RXNE 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。每接收到 1 个字节后置位 RXNE 标志。如果下一个数据已收到或先前 DMA 请求还没执行时，RXNE 标志仍为置起状态，溢出错误产生。

溢出错误产生时，执行如下错误处理：

- 置位 ORE；
- RDR 内容不会丢失。读 LPUART_DR 寄存器仍可正常读出先前数据；
- 移位寄存器内容被覆盖，随后接收到的数据丢失；
- 如果 RXNEIE=1，或者 EIE=1，则产生中断；
- 写 ORECF=1 清零 ORE 位；

ORE 置位，表明至少 1 个数据已经丢失。有如下两种情况：

- 如果 RXNE=1，上一个有效数据还在接收寄存器 RDR 中，可以被读出；
- 如果 RXNE=0，上一个有效数据已经被读走，RDR 无有效数据可读。当上一个有效数据在 RDR 中被读取的同时又接收到新的（被丢失）的数据时，此种情况可能发生。

31.2.6 LPUART 时钟源选择

LPUART 的时钟源由 RCC_CCIPR.LPUARTxSEL 寄存器配置。在配置 LPUART_CR1.UE=1 前，必须配置时钟源。

时钟源的选择决定了通讯速度范围。

LPUART 双时钟域，一个为内核时钟（lpuart_ker_ck），一个为系统时钟（lpuart_pclk）。当用作从低功耗唤醒时，内核时钟源由 RCC_CCIPR.LPUARTxSEL 寄存器配置。

当不支持双时钟域时，内核时钟与系统时钟相同。

当 MCU 进入低功耗模式，内核时钟源选择低频时，LPUART 可以接收数据并唤醒 MCU，然后软件或者 DMA 可以读取 LPUART_RDR 中数据。

帧错误

接收时没有检测到停止位，从而出现同步失效或大量噪声时，会产生帧错误。

帧错误时做如下处理：

- 硬件设置 FE 位；
- 无效数据从移位寄存器传送到 LPUART_RDR 寄存器；
- 在单字节通讯情况下，不产生中断。但因为 NE 标志位和 RXNE 标志位时同时被设置，FE 位出现上升沿。在多处理器通讯情况下，如果 LPUART_CR3.EIE=1，将产生一个中断。
- 写 LPUART_ICR.FECF=1 可以清零 FE 标志。

配置停止位个数

可配置位 1/2 个停止位：

- 1 停止位：在第 8/9/10 采样点采样停止位；
- 2 停止位：在第一个停止位采样后采样第 2 个停止位。在检测到第 2 个停止位结束后设置 RXNE 和 FE 标志。第 1 个停止位不会检测帧错误。

31.2.7 LPUART 波特率产生

发送和接收配置相同波特率。

$$\text{Tx/Rx 波特率} = \frac{256 \ X \ \text{lpuartckpres}}{\text{LPUARTDIV}}$$

LPUARTDIV 在 LPUART_BRR 寄存器配置。在通讯进行时不要改变波特率的值。LPUART_BRR 寄存器值不能低于 0x300.

fCK 的频率范围为 3 ~ 4096 倍*波特率。

LPUART 时钟源为 LSE 时，最大波特率为 9600 波特率。当 LPUART 由不同于 LSE 时钟的时钟源进行时钟时，可以达到更高的波特率。例如，LPUART 时钟源频率为 100 MHz，可达到的最大波特率约为 33M 波特。

表 31-1 lpuart_ker_ck_pres = 32.768 kHz 时编程波特率的误差计算

Baud rate	**lpuart_ker_ck_pres = 32,768 kHz**		
期望值	**实际值**	**在波特率寄存器中编程的值**	**误差%**
0.3 KBps	0.3 KBps	0x6D3A	0
0.6 KBps	0.6 KBps	0x369D	0
1200 Bps	1200.087 Bps	0x1B4E	0.007
2400 Bps	2400.17 Bps	0xDA7	0.007
4800 Bps	4801.72 Bps	0x6D3	0.035
9600 Bps	9608.94 Bps	0x369	0.035

表 31-2 lpuart_ker_ck_pres = 100 MHz 时编程波特率的误差计算

Baud rate	**lpuart_ker_ck_pres = 100 MHz**		
期望值	**实际值**	**在波特率寄存器中编程的值**	**误差%**
38400 波特	38400.04 波特	A2C2A	0.0001
57600 波特	38400.04 波特	6C81C	0.0001
115200 波特	115200.12 波特	3640E	0.0001
230400 波特	230400.23 波特	1B207	0.0001
460800 波特	460804.61 波特	D903	0.001
921600 波特	921625.81 波特	6C81	0.0028

4000000 波特	4000000.00 波特	1900	0
10000000 波特	10000000.00 波特	A00	0
20000000 波特	20000000.00 波特	500	0
33000000 波特	33032258.06 波特	307	0.1

31.2.8 LPUART 接收容忍度

只有当整体的时钟系统地变化小于 LPUART 异步接收器能够容忍的范围，LPUART 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+DWU < LPUART 接收器的容忍度

其中 DWU 是使用低功耗模式唤醒时由于采样点偏差引起的误差。

LPUART 接收器可以在指定的最大容忍偏差下正确接收数据

表 31-3 LPUART 接收容忍度

{M1,M0}位	768 < BRR < 1024	1024 < BRR < 2048	2048 < BRR < 4096	4096 ≤ BRR
7 位({M1,M0} =00), 1 停止位	2.3%	4.5%	4.9%	5.4%
8 位({M1,M0} =01), 1 停止位	2.2%	3.9%	4.9%	5%
9 位({M1,M0} =10), 1 停止位	1.4%	4%	4.2%	4.2%
7 位({M1,M0} =00), 2 停止位	2.3%	3.7%	4.9%	5.4%
8 位({M1,M0} =01), 2 停止位	1.3%	4.%	4.2%	4.2%
9 位({M1,M0} =10), 2 停止位	2.3%	3.6%	3.6%	4%

注：当接收到的帧中包含恰好是 10/11/9 位的空闲帧（对应 M='b00/11/10'）时，表中指定的数据在特殊情况下可能会略有不同。

31.2.9 LPUART 多处理器通信

通过 LPUART 可以实现多处理器通讯（将几个 LPUART 连在一个网络里）。例如某个 LPUART 设备可以是主，其 TX 输出和其他 LPUART 从机的 RX 输入相连接；LPUART 从机各自的 TX 输出逻辑与在一起，并且和主机的 RX 输入相连接。

未被寻址的设备可以配置 MME 寄存器置于静默模式。在静默模式：

- 不会设置任何接收状态位；
- 禁止所有接收中断；
- LPUART_ISR.RWU=1.通过配置 LPUART_RQR.MMRQ，硬件和软件可以控制 LPUART 进入静默模式；

根据 LPUART_CR1.WAKE 的值，LPUART 可以用两种方式进入或退出静默模式：

- WAKE=0：空闲总线检测；
- WAKE=1：地址标志检测；

空闲总线检测

当 MMRQ 写 1，RMU 自动置 1 后，LPUART 进入静默模式。

当检测到空闲帧时，LPUART 被唤醒。然后 RWU 被硬件清零，但是 LPUART_ISR.IDLE 并不置起。

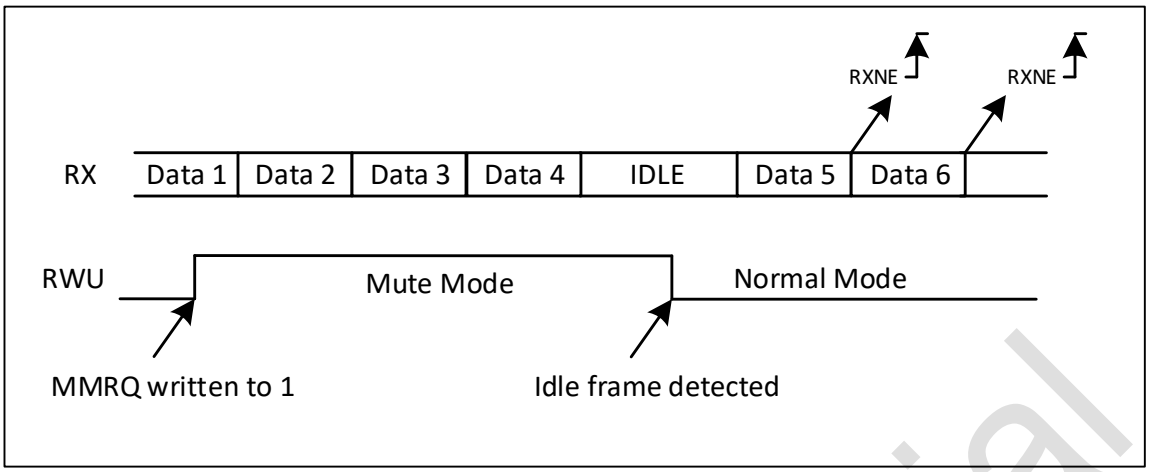


图 31-4 用总线帧检测的静默模式

注:

如果在 IDLE 字符已经过去时将 MMRQ 位置 1，将不会进入静默模式（RWU 未置 1）。

如果在线路处于空闲状态时激活 LPUART，在一个 IDLE 帧持续时间后（不只在接收一个字符帧后）会检测到空闲状态。

4 位/7 位地址标志检测

在这个模式，如果 MSB=1，该字节被认为是地址，否则被认为是数据。在一个地址字节中，目标接收器的地址被放在 4 个或者 7 个 LSB 中（由 ADDM7 寄存器选择几个）。这个 4 位/7 位地址被接收器同 LPUART_CR2 寄存器中 ADD 中配置的地址比较。

如果接收到的字节与配置地址不匹配，LPUART 进入静默模式。此时，硬件设置 RWU。接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 LPUART 已经在静默模式。

当 MMRQ 配置为 1 时，LPUART 也会进入静默模式，并设置 RWU=1.

如果接收到的字节与配置地址匹配，LPUART 退出静默模式。此时，清零 RWU，随后正常接收数据。因为 RWU 已被清零，此种情况会设置 RXNE/RXFNE 位。

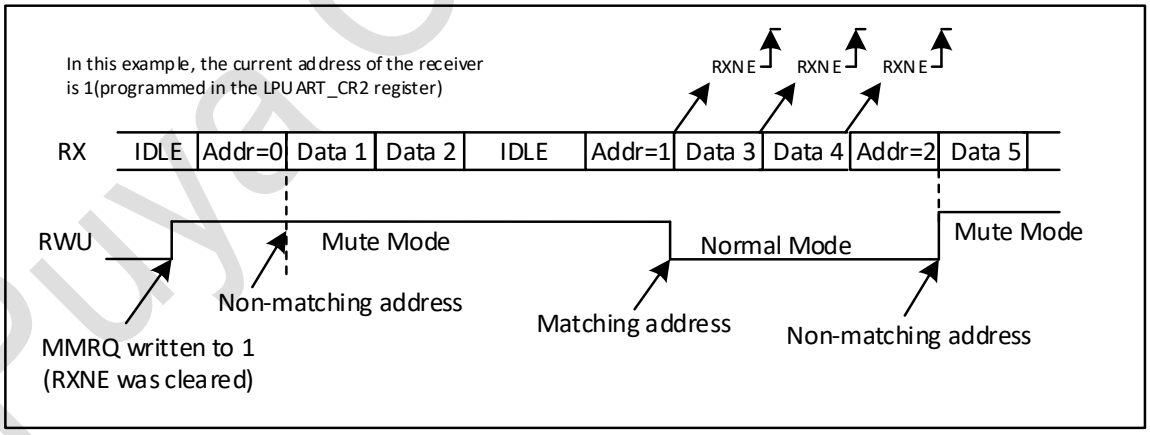


图 31-5 用地址标志检测的静默模式

31.2.10 LPUART 奇偶校验

设置 LPUART_CR1.PCE=1，可以使能奇偶控制（发送时生成一个奇偶位，接收时进行奇偶校验）。根据{M1,M0}位定义的帧长度，支持的 LPUART 帧格式如下表所示:

表 31-4 支持的 LPUART 帧格式

{M1,M0}位	PCE 位	LPUART 帧
00	0	| 开始位 | 8 位数据 | 停止位 |
00	1	|开始位| 7 位数据| 奇偶校验位 |停止位|
01	0	|开始位| 9 位数据|停止位|
01	1	|开始位| 8 位数据|奇偶校验位|停止位|
10	0	|开始位| 7 位数据|停止位|
10	1	|开始位| 6 位数据|奇偶校验位|停止位|

偶校验

偶校验位使得一帧中的 6 或 7 或 8 个 LSB 数据以及校验位中“1”的个数为偶数。

奇校验

奇校验位使得一帧中的 6 或 7 或 8 个 LSB 数据以及校验位中“1”的个数为奇数。

接收校验

奇偶校验错误时设置 PE=1，当 PEIE=1 时产生中断。软件写 PECF=1 清零 PE。

发送校验位

发送时 MSB 由奇偶校验位代替。

31.2.11 LPUART 单线半双工通信

当配置 LPUART_CR3.HDSEL=1 时，LPUART 进入单线半双工通讯模式。

在此模式下：

- TX 和 RX 在模块内部连接；
- RX 引脚不用；
- 当无数据传输时释放 TX 引脚作为标准 I/O

31.2.12 DMA 连续通信

LPUART 可以利用 DMA 连续通讯。接收缓冲器和发送缓冲器的 DMA 请求分别产生。

利用 DMA 发送

通过配置 LPUART_CR3.DMAT 使能利用 DMA 发送。当 TXE=1，DMA 就从指定的 SRAM 区传送数据到 LPUART_DR 寄存器。

DMA 发送的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 LPUART 使用的 DMA 通道
- 配置 LPUART_TDR 寄存器地址作为 DMA 传输的目标地址。在每个 TXE 事件后，数据将被传送到这个地址；
- 配置存储器（如 SRAM）地址作为 DMA 传输的源地址。在每次 TXE 事件后，将从此存储器读数据加载到 LPUART_TDR 寄存器；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 根据应用，配置在传输完成一半还是全部完成时产生 DMA 中断；
- 写 TCCF=1 清零 LPUART_ISR.TC 标志；
- 使能 DMA 通道；

  当传输达到配置的传输字节数，DMA 控制器产生中断请求。

  当 DMA 写完所有要传输数据后，设置 TCIF=1.然后软件需要等待 TC=1，表明 LPUART 已经发送完最后一帧数据。

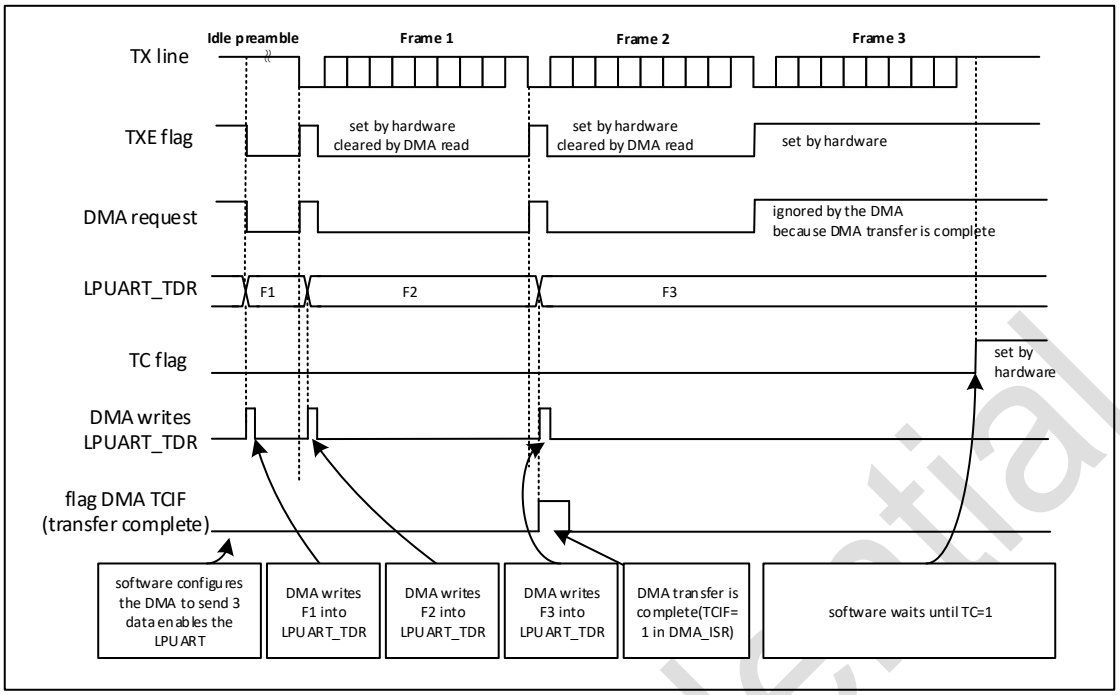


图 31-6 LPUART 用 DMA 发送

利用 DMA 接收

通过配置 LPUART_CR3.DMAR 使能利用 DMA 接收。每收到一个字节，DMA 就把数据从 LPUART_RDR 寄存器传送到指定的 SRAM 区。

DMA 接收的配置步骤如下：

- 配置 SYSCFG_CFGR3 或者 SYSCFG_CFGR4 选择 LPUART 使用的 DMA 通道
- 配置 LPUART_RDR 寄存器地址作为 DMA 传输的源地址。在每个 RXNE 事件后，数据将从此 LPUART_RDR 寄存器地址读出数据传输到存储器；
- 配置 SRAM 存储器地址作为 DMA 传输的目标地址。在每次 RXNE 事件后，数据将从 LPUART_RDR 寄存器传送到此地址；
- 写 DMA 控制寄存器配置传输字节数；
- 写 DMA 寄存器配置通道优先级；
- 根据应用，配置在传输完成一半还是全部完成时产生 DMA 中断；
- 使能 DMA 通道；

当传输达到配置的传输字节数，DMA 控制器产生中断请求。

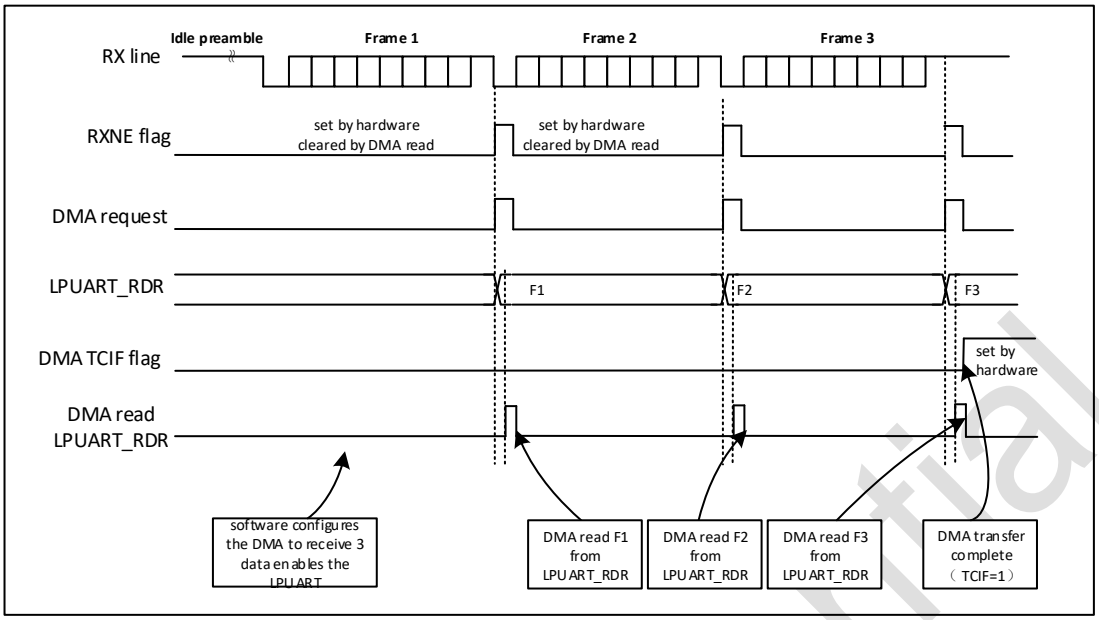


图 31-7 LPUART 用 DMA 接收

错误标志和中断

在多处理器通讯模式，在传输时产生任何错误都会在当前字节传送完成后产生错误标志，如果对应中断使能，则产生中断。

31.2.13 RS232 硬件流控制和 RS485 驱动器使能

通过 CTS 和 RTS 可以控制两个 LPUART 模块的串行数据流。

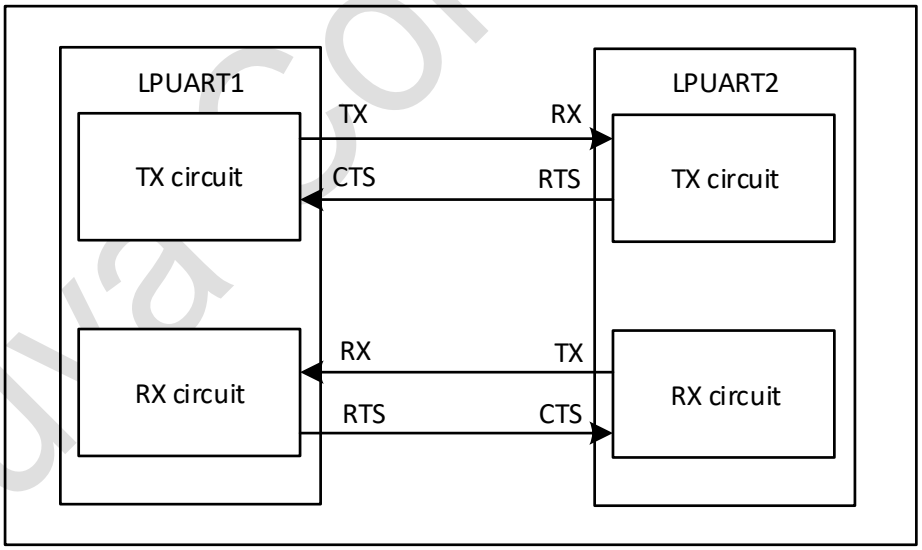


图 31-8 硬件流控制

RS232 RTS 流控制

通过配置 LPUART_CR3.RTSE=1，使能 RTS 流控制。当 LPUART 准备好接收数据时，将 RTS 拉低。当接收寄存器满时，RTS 拉高（无效），表明当前帧是最后一帧。

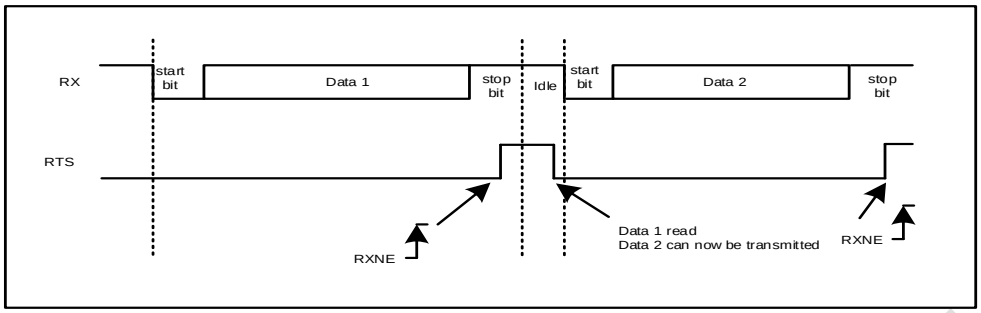


图 31-9 RTS 流控制

RS232 CTS 流控制

通过配置 LPUART_CR3.CTSE=1，使能 CTS 流控制。使能后，在传送下一帧前检测 CTS 信号。如果 CTS=0（有效），传送下一笔数据（假设有数据需要传输）。如果 CTS=1（无效），则当前传输完成（在 stop bit 位前完成）。

当 CTSE=1，只要 CTS 信号反转，硬件就置位 CTSIF。表明接收器准备好通讯或未准备好通讯。当 CTSIE=1 时，产生中断。

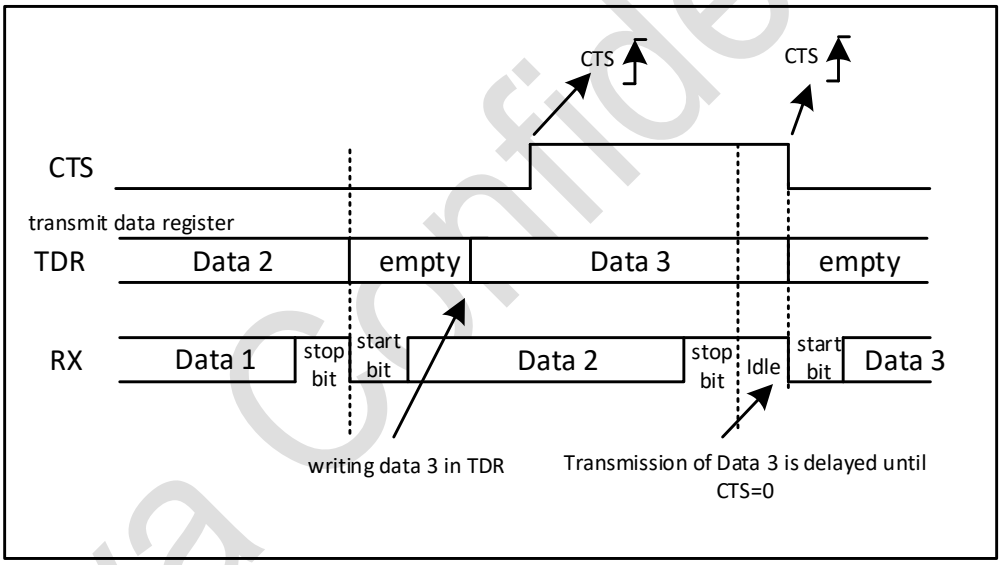


图 31-10 CTS 流控制

注：CTS 在当前帧结束前至少维持 3 个 LPUART 源时钟的高电平。CTSCF 也需要维持至少 2xPCLK 周期。

RS485 驱动使能

配置 LPUART_CR3.DEM=1，使能 DE 功能。通过 DE 信号，用户可以激活外部传输控制。DE 信号的有效极性通过 LPUART_CR3.DEP 寄存器配置。

有效时间由 LPUART_CR1.DEAT 配置，为 DE 信号有效沿与起始位开始之间的时间。

释放时间由 LPUART_CR1.DEDT 配置，为最后一个停止位与 DE 无效之间的时间。

具体时间计算如下所示，其中 P=BRR[20:11]:

- DE 插入时间：
  - （1+（DEAT x P））x f_{CK}，P≠0
  - （1+DEAT）x f_{CK}，P=0
- DE 释放时间：

— （1+（DEDT x P））x f_{CK}，P≠0

— （1+DEDT）x f_{CK}，P=0

31.2.14 LPUART 低功耗模式

LPUART 支持低功耗模式，使得在 LPUART PCLK 不存在时仍能传输数据。

当 UESM=1，LPUART 能在低功耗模式下唤醒 MCU。

根据 WUS 产生中断唤醒请求

当检测到唤醒事件，硬件产生 WUF 标志。如果 WUFIE=1 时，产生 WUF 中断。在此情况下， WUF 标志置 1 便足以将 MCU 从低功耗模式唤醒。

注：

- 进入低功耗模式前，保证没有 LPUART 传输；
- 不管 MCU 是在低功耗模式还是工作模式，只要检测到唤醒事件，就会置位 WUF 标志；
- 当初始化和使能 LPUART 接收后，必须要通过 REACK 位确认 LPUART 是否真正使能，然后再配置进入低功耗模式；
- 当使能 DMA 接收，在进入低功耗模式前必须关闭 DMA 接收；然后从低功耗唤醒后再重新使能 DMA；

从静默模式进入低功耗模式

如果 LPUART 在进入低功耗模式前处于静默模式：

- 不能用空闲帧唤醒静默模式，因为在低功耗模式下不支持空闲帧检测；
- 用地址匹配的方式唤醒静默模式，则从低功耗模式唤醒也要用地址匹配的方法。当进入低功耗时，如果 RXNE=1，则 LPUART 将不会进入低功耗模式，而维持在静默模式；

低功耗模式时内核时钟关闭

如果低功耗模式下，内核时钟也关闭，则 LPUART 在 RX 线上检测到下降沿时，会产生 lpuart_ker_ck_req 信号去使能内核时钟。

如果唤醒事件 LPUART_CR3.WUS 配置为地址匹配方式：

- 内核时钟使能后，接收完一帧后如果地址匹配，则 MCU 唤醒，开始正常数据接收；

下图为唤醒事件后地址验证匹配的情况：

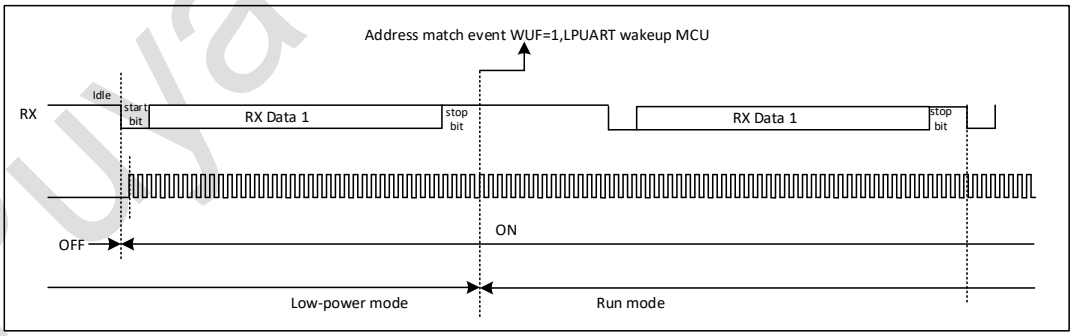


图 31-11 LPUART 地址匹配唤醒

如果唤醒事件后地址验证不匹配，则内核时钟再次关闭，MCU 不会被唤醒，系统仍在低功耗模式。

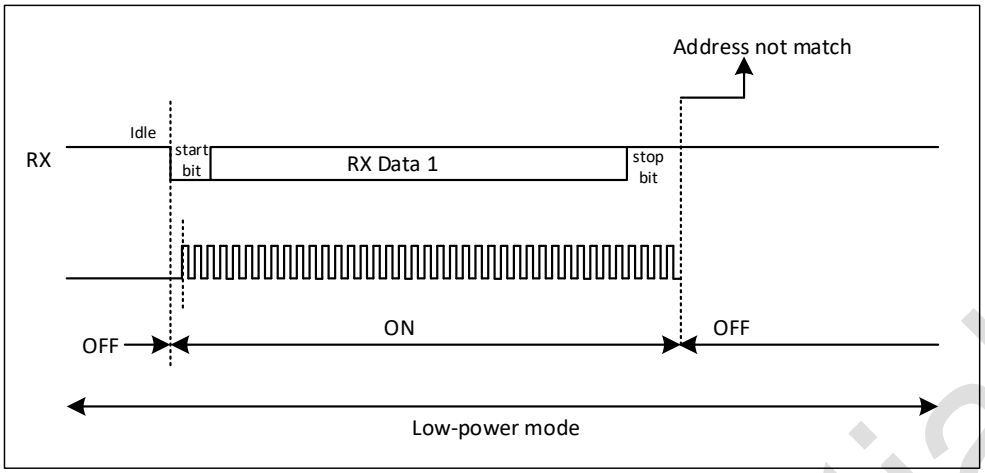


图 31-12 LPUART 地址不匹配无法唤醒

31.3 LPUART 中断

表 31-5 LPUART 中断请求

中断事件	事件标志	使能位	中断清零	中断信号是否有效	
				中断	唤醒
发送数据寄存器空	TXE	TXEIE	写数据到 TDR	YES	NO
CTS 中断	CTSIF	CTSIE	软件配置 CTSCF=1	YES	NO
传送完成	TC	TCIE	写数据到 TDR 或 TCCF=1	YES	NO
接收寄存器未空（读数据准备好）	RXNE（对应标志位 RXFNE）	RXNEIE（对应中断使能 RXFNEIE）	读 RDR 寄存器或 RXFRQ=1	YES	YES
溢出错误	ORE	RXNEIE/RXFNEIE	ORECF=1	YES	NO
空闲帧	IDLE	IDLEIE	IDLECF=1	YES	NO
奇偶校验错误	PE	PEIE	PECF=1	YES	NO
多处理器通讯时，噪声、overrun 和帧错误	NR/ORE/FE	EIE	NECF=1 清零 NE； ORECF=1 清零 ORE； FECF=1 清零 FE；	YES	NO
地址字节匹配	CMF	CMIE	CMCF=1	YES	NO
从低功耗模式唤醒	WUF	WUFIE	WUCF=1	YES	YES

31.4 LPUART 寄存器

31.4.1 LPUART 控制寄存器 1（LPUART_CR1）

偏移地址：0x00

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFFIE	TXFEIE	FIFOEN	M1	Res.	Res.	DEAT[4:0]					DEDT[4:0]				
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**
Res.	CMIE	MME	M0	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	R/W	Reset Value	Function
31	RXFFIE	RW	0	RXFIFO 满中断使能。 0：中断禁止

				1：RXFF 中断使能 只能在 FIFO 使能时（FIFOEN=1）才能写入该位
30	TXFEIE	RW	0	TXFIFO 空中断使能。 0：中断禁止 1：TXFE 中断使能 只能在 FIFO 使能时（FIFOEN=1）才能写入该位
29	FIFOEN	RW	0	FIFO 模式使能。 0：FIFO 模式禁止 1：FIFO 模式使能 只能在 LPUART 未使能时（UE=0）才能写入该位
28	M1	RW	0	{M1,M0}的值配置长度，由软件置位或者清零。 2’b00: 1 起始位，8 数据位，n 停止位 2’b01: 1 起始位，9 数据位，n 停止位 2’b10: 1 起始位，7 数据位，n 停止位 只能在 LPUART 未使能时（UE=0）才能写入该位。
27:26	Reserved	-	-	保留
25:21	DEAT[4:0]	RW	5’h0	该寄存器配置驱动使能（DE）信号有效和起始位之间的时间。用采样时间作为单位。 如果不支持 DE 功能，则该位保留。 只能在 LPUART 未使能时（UE=0）才能写入该位
20:16	DEDT[4:0]	RW	5’h0	该寄存器配置发送帧停止位与 DE 信号无效之间的时间。用采样时间作为单位（1/8 或者 1/16 位周期，取决于过采样率）。如果 DEDT 时间内有写 LPUART_TDR 寄存器请求发送，则数据只有当 DEDT 和 DEAT 都结束后才发送。 如果不支持 DE 功能，则该位保留。 只能在 LPUART 未使能时（UE=0）才能写入该位
15	Reserved	-	-	保留
14	CMIE	RW	0	字符匹配中断使能。该位由软件置位或者清零。 0：中断禁止 1：CMF 中断使能
13	MME	RW	0	静音模式使能。 0：接收器工作在工作模式 1：接收器在工作模式和静默模式之间切换
12	M0	RW	0	与 M1 结合配置字长度。 只能在 LPUART 未使能时（UE=0）才能写入该位
11	WAKE	RW	0	接收唤醒方式。 从静默模式唤醒方式。由软件置位或者清零。 0：空闲帧唤醒 1：地址标志唤醒 只能在 LPUART 未使能时（UE=0）才能写入该位
10	PCE	RW	0	奇偶校验控制。 0：奇偶校验禁止 1：奇偶校验使能 奇偶校验位：9bit 数据的第 9 位；8bit 数据的第 8 位；7bit 数据的第 7 位。

				只能在 LPUART 未使能时（UE=0）才能写入该位
9	PS	RW	0	奇偶校验选择。由软件置位和清零。 0：偶校验 1：奇校验 只能在 LPUART 未使能时（UE=0）才能写入该位
8	PEIE	RW	0	PE 中断使能。由软件置位和清零。 0：禁止 1：PE 中断使能
7	TXEIE	RW	0	传输寄存器空中断使能。由软件置位和清零。 0：禁止 1：TXE 中断使能
6	TCIE	RW	0	传送结束中断使能。由软件置位和清零。 0：禁止 1：TC 中断使能
5	RXNEIE	RW	0	接收寄存器非空中断使能；由软件置位和清零。 0：禁止 1：ORE 或者 RXNE/RXFNE 中断使能
4	IDLEIE	RW	0	IDLE 中断使能。由软件置位和清零。 0：禁止 1：IDLE 中断使能
3	TE	RW	0	传送使能。 0：传送禁止 1：传送使能
2	RE	RW	0	接收使能。 0：接收禁止 1：接收使能，开始检测 start 位
1	UESM	RW	0	停止低功耗模式下 LPUART 唤醒使能。 该寄存器为 1，如果 LPUART 时钟选择 LSI 或 LSE，LPUART 能唤醒 MCU。 0：LPUART 不能唤醒 MCU 1：LPUART 模式唤醒 MCU。使能后，LPUART 时钟源选择 LSI 或 LSE 从低功耗模式唤醒后，该位清零。
0	UE	RW	0	LPUART 使能。 当该位清零后，LPUART 模块会立即停止当前操作。LPUART 的配置会保持，但所有状态标志位，LPUART_ISR 寄存器值会变为默认值。该位由软件置位和清零。 0：LPUART 预分频器和输出禁止，低功耗模式 1：LPUART 使能 软件需要等待 LPUART_ISR.TC 置位后，才能清零 UE 位，进入低功耗模式； 同时，在清零 UE 位之前 DMA 通道需要禁止。 该位清零时，LPUART 配置寄存器值维持，但 LPUART_ISR 状态寄存器全部复位。

31.4.2 LPUART 控制寄存器 2（LPUART_CR2）

偏移地址：0x04

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD[7:0]								Res.	Res.	Res.	TXOE_ALWAYS_ON	MSBFIRST	DATAINV	TXINV	RXINV
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**
SWAP	Res.	STOP[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	ADDM7	Res.	Res.	Res.	Res.
RW		RW	RW								RW				

Bit	Name	R/W	Reset Value	Function
31:24	ADD[7:0]	RW	8’b0	LPUART 地址。 该寄存器用于多处理器静默或者停止模式，用作地址唤醒时的地址。传送器发送的 MSB 位为 1. 在正常接收模式时，该寄存器用作字符检测。此时，比对接收内容与 ADD[7:0]相比较，如果匹配，则置位 CMF 标志位。 只有在 RE=0 或者 UE=0 时，该寄存器可以被写。
23:21	Reserved	-	-	保留
20	TXOE_ALWAYS_ON	RW	1	该寄存器用于控制 LPUART 输出使能。 0：硬件自主控制输出使能开/关 1：软件控制输出使能常开 只能在 LPUART 未使能时（UE=0）才能写入该位 注：在半双工模式，需要将此位关闭。
19	MSBFIRST	RW	0	最高有效位在前。 0：起始位后，收发第 0 位数据； 1：起始位后，收发第 7/8/9 位数据； 只能在 LPUART 未使能时（UE=0）才能写入该位
18	DATAINV	RW	0	二进制数据逆处理。 0：以正/正向逻辑收发数据寄存器数据。（1=H，0=L） 1：以反/反向逻辑收发数据寄存器数据。奇偶位也会逆处理。（1=L，0=H） 只能在 LPUART 未使能时（UE=0）才能写入该位
17	TXINV	RW	0	TX 引脚有效电平反相。 0：TX 引脚为标准逻辑电平（V_{DD} =1/空闲, Gnd=0/标记） 1：TX 引脚信号值取反（V_{DD} =0/标记, Gnd=1/空闲） 只能在 LPUART 未使能时（UE=0）才能写入该位
16	RXINV	RW	0	RX 引脚有效电平反相。 0：RX 引脚为标准逻辑电平（V_{DD} =1/空闲, Gnd=0/标记） 1：RX 引脚信号值取反（V_{DD} =0/标记, Gnd=1/空闲） 只能在 LPUART 未使能时（UE=0）才能写入该位

15	SWAP	RW	0	TX/RX 引脚互换。 0：TX/RX 引脚按照标准引脚排列定义 1：TX/RX 引脚互换。此时用作交叉连接其他 LPUART 时 只能在 LPUART 未使能时（UE=0）才能写入该位
14	Reserved	-	-	保留
13:12	STOP[1:0]	RW	2’b0	停止位配置。 00：1 停止位 01，11：保留 10：2 停止位 只能在 LPUART 未使能时（UE=0）才能写入该位
11:5	Reserved	-	-	保留
4	ADDM7	RW	0	7 位/4 位地址检测配置。 0：4 位地址检测 1：7 位地址检测（8 位数据模式） 只能在 LPUART 未使能时（UE=0）才能写入该位
3:0	Reserved	-	-	保留

31.4.3 LPUART 控制寄存器 3（LPUART_CR3）

偏移地址：0x08

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXFTCFG[2:0]			RXFTIE	RXTFCFG			Res.	TXFTIE	WUFIE	WUS[1:0]		Res.	Res.	Res.	Res.
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**
DEP	DEM	DDRE	OVRDIS	Res.	CTSIE	CTSE	RTSE	DMAT	DMAR	Res.	Res.	HDSEL	Res.	Res.	EIE
RW	RW	RW	RW		RW	RW	RW	RW	RW			RW			RW

Bit	Name	R/W	Reset Value	Function
31:29	TXFTCFG[2:0]	RW	3’h0	TXFIFO 阈值配置。 000：TXFIFO 达到总空间 1/8 001：TXFIFO 达到总空间 1/4 110：TXFIFO 达到总空间 1/2 011：TXFIFO 达到总空间 3/4 100：TXFIFO 达到总空间 7/8 101：TXFIFO 空 其他：保留
28	RXFTIE	RW	0	RXFIFO 阈值中断使能。 0：中断禁止 1：FIFO 达到 RXFTCFG 中配置阈值中断使能
27:25	RXFTCFG[2:0]	RW	3’h0	RXFIFO 满阈值配置。 000：RXFIFO 达到总空间 1/8 001：RXFIFO 达到总空间 1/4 110：RXFIFO 达到总空间 1/2 011：RXFIFO 达到总空间 3/4 100：RXFIFO 达到总空间 7/8 101：RXFIFO 满 其他：保留

24	Reserved	-	-	保留
23	TXFTIE	RW	0	TXFIFO 阈值中断使能。 0：中断禁止 1：TXFIFO 达到 TXFTCFG 配置阈值中断使能
22	WUFIE	RW	0	低功耗唤醒中断使能。 0：中断禁止 1：WUF 中断使能
21:20	WUS[1:0]	RW	0	低功耗唤醒选择。 00：地址匹配产生 WUF 01：保留 10：起始位检测产生 WUF 11：RXNE 产生 WUF
19:16	Reserved	-	-	保留
15	DEP	RW	0	DE 极性选择。 0：DE 高电平有效 1：DE 低电平有效 只能在 LPUART 未使能时（UE=0）才能写入该位
14	DEM	RW	0	DE 模式使能。 0：禁止 1：DE 模式使能，DE 信号在 RTS 引脚输出 用户通过使能该位来控制外部传输。 只能在 LPUART 未使能时（UE=0）才能写入该位
13	DDRE	RW	0	接收错误时是否禁止 DMA。 0：接收错误时不禁止 DMA。相应出错标志置位，但 RXNE=0. 不会产生 DMA 请求，所以出错数据不会输出。 1：接收错误时禁止 DMA。RXNE 和出错标志位都会置位。在出错标志位清零前，屏蔽 DMA 请求。 只能在 LPUART 未使能时（UE=0）才能写入该位
12	OVRDIS	RW	0	溢出禁止。 0：当接收到新数据前已收数据没读时置位溢出出错标志（PRE） 1：溢出功能禁止。接收到新数据后，RXNE=1，ORE=0，新数据会覆盖 LPUART_RDR 寄存器中数据 只能在 LPUART 未使能时（UE=0）才能写入该位
11	Reserved	-	-	保留
10	CTSIE	RW	0	CTS 中断使能。 0：禁止 1：CTSIF 中断使能
9	CTSE	RW	0	CTS 使能。 0：CTS 硬件流控制禁止 1：CTS 模式使能。当有当 CTS 输入为 0 时，才会传输数据。此时，当数据写入数据寄存器后，要等待 CTS 有效后才会启动传输 只能在 LPUART 未使能时（UE=0）才能写入该位

8	RTSE	RW	0	RTS 使能。 0：RTS 硬件流控制禁止 1：RTS 输出使能，只有当接收缓冲区未满时才会请求下一个数据。当前数据发送完成后，发送操作暂停。如果可以接收数据了，将 RTS 置为有效（0） 只能在 LPUART 未使能时（UE=0）才能写入该位
7	DMAT	RW	0	传送时使能 DMA。 0：禁止 1：传送时使能 DMA
6	DMAR	RW	0	接收时使能 DMA。 0：禁止 1：接收时使能 DMA
5:4	Reserved	-	-	保留
3	HDSEL	RW	0	半双工选择。 0：非半双工模式 1：半双工模式选择 只能在 LPUART 未使能时（UE=0）才能写入该位
2:1	Reserved	-	-	保留
0	EIE	RW	0	错误中断使能。 0：禁止 1：帧错误 FE、溢出错误 ORE、噪声 NF 中断产生；

31.4.4 LPUART 波特率寄存器（LPUART_BRR）

偏移地址：0x0C

复位值：0x0000_0300

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

Bit	Name	R/W	Reset Value	Function
31:20	Reserved	-	-	保留
19:0	BBR[19:0]	RW	20’h300	LPUART 波特率。 禁止在 LPUART_BRR 寄存器中写入小于 0x300 的值

31.4.5 LPUART 请求寄存器（LPUART_RQR）

偏移地址：0x18

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFRQ	RXFRQ	MMRQ	SBKRQ	Res.
											RW	RW	RW	RW	

Bit	Name	R/W	Reset Value	Function
31:5	Reserved	-	-	保留
4	TXFRQ	W	0	传送数据刷新请求。 写 1 刷新整个 FIFO，同时会置位 TXFE（表明 TXFIFO 空）。

3	RXFRQ	W	0	接收数据刷新请求。 该位写 1 会清零 RXNE 标志。 通过该位能不读并且丢弃接受到的数据，避免 overrun 情况。
2	MMRQ	W	0	静音模式请求。 该位写 1 使 LPUART 进入静默模式，并置位 RWU 标志。
1	SBKRQ	W	0	发送中止请求。 该位写 1 会置位 SBKF，并发送 BREAK 请求。
0	Reserved	-	-	保留

31.4.6 LPUART 中断和状态寄存器（LPUART_ISR）

偏移地址：0x1C

复位值：0x0080_00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	TXFT	RXFT	Res	RXFF	TXFE	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
-				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**
Res.	Res.	Res.	Res.	Res.	CTS	CTSIF	Res.	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
-					R	R	-	R	R	R	R	R	R	R	R

Bit	Name	R/W	Reset Value	Function
31:28	Reserved	-	-	保留
27	TXFT	R	0	TXFIFO 阈值标志。 0：TXFIFO 未达到 TXFTCFG 阈值 1：TXFIFO 达到 TXFTCFG 阈值
26	RXFT	R	0	RCFIFO 阈值标志。 0：RXFIFO 未达到 RXFTCFG 阈值 1：RXFIFO 达到 RXFTCFG 阈值
25	Reserved	-	-	保留
24	RXFF	R	0	RXFIFO 满标志。表明接收到的数据 RXFIFO size+1。 0：RXFIFO 未满 1：RXFIFO 满
23	TXFE	R	1 开启 FIFO_EN 默认 1，否则默认 0	TXFIFO 空。 0：TXFIFO 未空 1：TXFIFO 空 注：FIFO_EN 关闭的时候该位为 0
22	REACK	R	0	接收使能确认标志。 硬件置位/复位该寄存器。表明硬件在进入低功耗模式前准备好接收。 如果 LPUART 不支持从停止低功耗模式唤醒，则该位保留。
21	TEACK	R	0	发送使能确认标志。 硬件置位/复位该寄存器。 当在 LPUART_CR1 寄存器中写入 TE=0 生成空闲帧请求，然后写入 TE=1 以满足 TE=0 的最小周期时，可以使用该寄存器。
20	WUF	R	0	从低功耗模式唤醒标志。

				当检测到唤醒事件时硬件写该位为 1. 事件通过 WUS 位域定义。通过向 LPUART_ICR 寄存器中的 WUCF 写入 1，此位由软件清零。如果 LPUART_CR3 寄存器中 WUFIE=1，则会生成中断。 注：当 UESM 清零时，WUF 标志也清零。
19	RWU	R	0	接收到静默模式唤醒。 当接收到静默模式序列，该寄存器置位；如果接收到唤醒序列，该寄存器清零。具体哪种唤醒序列（地址标记或者空闲帧）由寄存器 LPUART_CR1.WAKE 位控制。 0：接收器为工作模式； 1：接收器为静默模式；
18	SBKF	R	0	发送中止标志。 该寄存器表明请求发送断开字符。由软件写 1 到 LPUART_RQR.SBKRQ 寄存器置位该寄存器。硬件在发送完中止字符的停止位后清零该寄存器。 0：不发送断开字符； 1：发送断开字符；
17	CMF	R	0	地址匹配标志。 当接收到 ADD[7:0]值匹配的字符时置位该寄存器。软件写 1 到 CMCF 寄存器会清零该位。
16	BUSY	R	0	忙标志。 当 RX 线接收数据时（正确接收到起始位），硬件该寄存器置位。接收结束，硬件清零该寄存器。 0：LPUART 空闲 1：正在进行接收
15:11	Reserved	-	-	保留
10	CTS	R	0	CTS 标志。 0：CTS 线为 1 1：CTS 线为 0
9	CTSIF	R	0	CTS 中断标志。 0：CTS 状态线未改变 1：CTS 状态线值改变
8	Reserved	-	-	保留
7	TXE	R	1	传输寄存器空标志，可以写数据到 LPUART_TDR。 0：数据寄存器满 1：数据寄存器空
6	TC	R	1	传送完成标志。 传送数据帧完成后，且 TXE=1，则硬件置位该寄存器。TCIE=1 时产生中断。 软件写 1 到 TCCF 寄存器清零该位。 0：传送未完成 1：传送完成
5	RXNE	R	0	读数据寄存器不空标志。 当移位寄存器值传送到 LPUART_RDR 寄存器，硬件置位该寄存器。

				软件读 LPUART_RDR 寄存器则清零该位。 当 RXNEIE=1 时，产生中断。 0：未收到数据 1：接收数据准备好读出
4	IDLE	R	0	空闲标志。 检测到空闲帧，硬件置位该寄存器。当 IDLEIE=1 时产生中断。 软件写 1 到 IDLECF 清零该位。 0：未检测到 IDLE 帧 1：检测到 IDLE 帧
3	ORE	R	0	溢出错误标志。 当 RXNE=1 时，在移位寄存器中接收到的数据正准备转移到 RDR 寄存器时，硬件设置该位。 软件写 1 到 ORECF 寄存器清零该位。 当 RXNEIE=1 或者 EIE=1 时，产生中断。 注：该寄存器置位时，RDR 寄存器内容不会丢失，但移位寄存器内容被覆盖。 当 EIE=1 时，产生 ORE 中断。
2	NE	R	0	START 位噪声标志。 在数据帧接收到噪声时，硬件置位该寄存器。 软件写 NFCF=1 会清零该位。
1	FE	R	0	帧错误标志。 当检测到不同步、过多的噪声或中断字符时，由硬件设置此位。 软件写 FECF=1 清零该位。 当 EIE=1 时，产生中断。
0	PE	R	0	校验值错误。 当接收时校验值错误时，硬件置位该寄存器。 软件写 PECF=1 清零该位。 当 PEIE=1 时，产生中断。

31.4.7 LPUART 中断标志清零寄存器（LPUART_ICR）

偏移地址：0x20

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WUCF	Res.	Res.	CMCF	Res.
											W			W	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	Res.	Res.	Res.	Res.	Res.	CTSCF	Res.	Res.	TCCF	Res.	IDLECF	ORECF	NECF	FECF	PECF
									W		W	W	W	W	W

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20	WUCF	W	0	低功耗唤醒唤醒标志清零。 软件写 1 会清零 WUF 寄存器。
19:18	Reserved	-	-	保留
17	CMCF	W	0	地址匹配标志清零。 软件写 1 会清零 CMF 寄存器。
16:10	Res			

9	CTSCF	W	0	CTS 标志清零。 软件写 1 清零 CTSIF 寄存器。
8:7	Reserved	-	-	保留
6	TCCF	W	0	传送完成标志清零。 软件写 1 清零 TC 寄存器。
5	Reserved	-	-	保留
4	IDLECF	W	0	空闲标志清零。 软件写 1 清零 IDLEF 寄存器。
3	ORECF	W	0	溢出错误标志清零。 软件写 1 清零 ORE 寄存器。
2	NECF	W	0	噪声标志清零。 软件写 1 会清零 NE 寄存器。
1	FECF	W	0	帧错误标志清零。 软件写 1 清零 FE 寄存器。
0	PECF	W	0	校验值错误标志清零。 软件写 1 清零 PE 寄存器。

31.4.8 LPUART 接收数据寄存器（LPUART_RDR）

偏移地址：0x24

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Res			
8:0	RDR[8:0]	R	9’h0	接收数据寄存器。

31.4.9 LPUART 发送数据寄存器（LPUART_TDR）

偏移地址：0x28

复位值：0x0000_0000

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

Bit	Name	R/W	Reset Value	Function
31:9	Reserved	-	-	保留
8:0	TDR[8:0]	RW	9’h0	发送数据寄存器。 当校验使能时，该寄存器的 bit7 或者 bit8 在发送时被校验位代替。 注意：建议先开启 TE 和 UE 之后，再写 TDR 数据。

31.4.10 LPUART 预分频器寄存器（LPUART_PRESC）

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
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

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

Bit	Name	R/W	Reset Value	Function
31:4	Reserved	-	-	保留
3:0	PRESCALER	RW	4’h0	输入时钟预分频寄存器。 0000: 不分频 0001: 2 分频 0010: 4 分频 0011: 6 分频 0100: 8 分频 0101: 10 分频 0110: 12 分频 0111: 16 分频 1000: 32 分频 1001: 64 分频 1010: 128 分频 1011: 256 分频 其它: 256 分频

32. MCU 调试接口

32.1 DBGMCU 简介

DBGMCU 模块给调试者提供以下支持：

- 低功耗模式
- 定时器、看门狗、I²C 的时钟控制
- DBGMCU 寄存器还提供芯片 ID 的编码。使用 JTAG 或者 SW 调试接口，或者用户程序都可以访问此 ID 编码。

32.2 DBGMCU 主要特性

- 支持睡眠模式，停止模式模式的调试
- CPU 进入 HALT 时，控制定时器、看门狗停止计数或者继续计数
- CPU 进入 HALT 时，阻止 I²C 的 SMBUS 超时

32.3 DBGMCU 功能描述

32.3.1 支持调试低功耗模式

用户可以通过执行 WFI 和 WFE 指令进入低功耗模式。

MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的功耗。

内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。

为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

- 在睡眠模式下，调试器必须先置位 DBG_SLEEP 位。这将为 HCLK 提供与 FCLK 相同的时钟。
- 在停止模式下，调试器必须先置位 DBG_STOP 位。这将激活 HSI 时钟，在停止模式下为 FCLK 和 HCLK 提供时钟。

32.4 DBGMCU 寄存器描述

DBGMCU 寄存器映射在外部 PPB 总线。

32.4.1 ID 编码（DBGMCU_IDCODE）

该寄存器由上电复位，系统复位不会复位该寄存器。当内核处于复位状态下时，调试器可以访问该寄存器。

偏移地址：0xE0042000

复位值：0x4444_BBBB

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

Bit	Name	R/W	Reset Value	Function
31:0	REV_ID[31:0]	R	32'h4444BBBB	MCU 产品 ID。该值保存在 Flash information 区。

32.4.2 调试配置寄存器（DBGMCU_CR1）

该寄存器由上电复位，系统复位不会复位该寄存器。当内核处于复位状态下时，调试器可以访问该寄存器。

偏移地址：0xE0042004

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res											DBG_TIM7_STOP	DBG_TIM6_STOP	Res		DBG_I2C2_SMBUS_TIMEOUT
											RW	RW			RW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG_I2C1_SMBUS_TIMEOUT	Res	DBG_TIM4_STOP	DBG_TIM3_STOP	DBG_TIM2_STOP	DBG_TIM1_STOP	DBG_WWDG_STOP	DBG_IWDG_STOP	Res						DBG_STOP	DBG_SLEEP
RW		RW	RW	RW	RW	RW	RW							RW	RW

Bit	Name	R/W	Reset Value	Function
31:21	Reserved	-	-	保留
20	DBG_TIM7_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM7 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
19	DBG_TIM6_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM6 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
18:17	Reserved	-	-	保留
16	DBG_I2C2_SMBUS_TIMEOUT	RW	0	当 CPU 核处于 halt 状态时，控制 I2C2 SMBUS 超时模式停止。 0：CPU halt 时，I2C2 与正常模式相同； 1：CPU halt 时，I2C2 SMBUS 超时功能禁止；
15	DBG_I2C1_SMBUS_TIMEOUT	RW	0	当 CPU 核处于 halt 状态时，控制 I2C1 SMBUS 超时模式停止。 0：CPU halt 时，I2C1 与正常模式相同； 1：CPU halt 时，I2C21 SMBUS 超时功能禁止；
14	Reserved	-	-	保留
13	DBG_TIM4_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM4 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；
12	DBG_TIM3_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM3 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；
11	DBG_TIM2_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM2 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；
10	DBG_TIM1_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM1 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；
9	DBG_WWDG_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 WWDG 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；

8	DBG_IWDG_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 IWDG 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数；
7:2	Reserved	-	-	保留
1	DBG_STOP	RW	0	调试停止模式。 0：（FCLK 关，HCLK 关）在停止模式，HCLK 和 FCLK 都会关闭。当从停止模式退出时，时钟配置与复位后相同（系统时钟为 HSI）。随后，软件需要重新配置时钟控制器以使能 PLL 和 HSE 等。 1：（FCLK 开，HCLK 开）。当进入停止模式，系统内部时钟源不会关闭，FCLK 和 HCLK 存在。当退出停止模式，如果需要改变时钟控制，软件需要重新配置。
0	DBG_SLEEP	RW	0	调试睡眠模式。 0：（FCLK 开，HCLK 关）。在睡眠模式，FCLK 由原先配置好的系统时钟提供，HCLK 关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出后，软件不需要重新配置时钟。 1：（FCLK 开，HCLK 开）。在睡眠模式，FCLK 和 HCLK 时钟都由原先配置好的系统时钟提供。

32.4.3 调试配置寄存器（DBGMCU_CR2）

该寄存器由上电复位，系统复位不会复位该寄存器。当内核处于复位状态下时，调试器可以访问该寄存器。

偏移地址：0xE0042008

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res	Res	Res	DBG_PWM4_STOP	DBG_PWM3_STOP	DBG_PWM2_STOP	DBG_PWM1_STOP	DBG_LPTIM_STOP	Res	Res	DBG_TIM17_STOP	DBG_TIM16_STOP	DBG_TIM15_STOP	Res	Res	Res
			RW	RW	RW	RW	RW			RW	RW	RW			

Bit	Name	R/W	Reset Value	Function
31:13	Reserved	-	-	保留
12	DBG_PWM4_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 PWM4 计数停止。 0：CPU halt 时，PWM4 时钟使能，正常计数； 1：CPU halt 时，PWM4 时钟禁止，停止计数，输出禁止；
11	DBG_PWM3_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 PWM3 计数停止。 0：CPU halt 时，PWM3 时钟使能，正常计数； 1：CPU halt 时，PWM3 时钟禁止，停止计数，输出禁止；

10	DBG_PWM2_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 PWM2 计数停止。 0：CPU halt 时，PWM2 时钟使能，正常计数； 1：CPU halt 时，PWM2 时钟禁止，停止计数，输出禁止；
9	DBG_PWM1_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 PWM1 计数停止。 0：CPU halt 时，PWM1 时钟使能，正常计数； 1：CPU halt 时，PWM1 时钟禁止，停止计数，输出禁止；
8	DBG_LPTIM_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 LPTIM 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
7:6	Reserved	-	-	保留
5	DBG_TIM17_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM17 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
4	DBG_TIM16_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM16 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
3	DBG_TIM15_STOP	RW	0	当 CPU 核处于 halt 状态时，控制 TIM15 计数停止。 0：CPU halt 时，计数器时钟使能，正常计数； 1：CPU halt 时，计数器时钟禁止，停止计数，输出禁止；
2:0	Reserved	-	-	保留

33. 版本历史

版本	日期	更新记录
V1.0	2025.09.12	初版
V1.1	2025.11.28	1. 更新UART/USART/LPUART容忍度参数 2. 补充TIM6/7章节内容

Puya Semiconductor Co., Ltd.