



N32G430 系列

**32-bit ARM® Cortex®-M4 微控制器**

用户手册 V1.2.1

## 目录

<b>1 文中的缩写 .....</b>	<b>27</b>
1.1 寄存器位操作缩写列表 .....	27
1.2 可用外设 .....	27
<b>2 存储器和总线架构 .....</b>	<b>28</b>
2.1 系统架构 .....	28
2.1.1 总线架构 .....	28
2.1.2 总线地址映射 .....	29
2.1.3 启动管理 .....	33
2.2 存储系统 (Memory system) .....	34
2.2.1 FLASH 规格 .....	34
2.2.2 iCache .....	45
2.2.3 SRAM .....	46
2.2.4 FLASH 寄存器描述 .....	47
<b>3 电源控制 (PWR) .....</b>	<b>56</b>
3.1 通用描述 .....	56
3.1.1 电源 .....	56
3.1.2 电源监控 .....	57
3.1.3 NRST .....	58
3.2 功耗模式 .....	59
3.2.1 SLEEP 模式 .....	61
3.2.2 STOP0 模式 .....	62
3.2.3 STOP2 模式 .....	62
3.2.4 STANDBY 模式 .....	63
3.3 PWR 寄存器 .....	64
3.3.1 PWR 寄存器总览 .....	64
3.3.2 电源控制寄存器(PWR_CTRL) .....	64
3.3.3 电源控制状态寄存器 (PWR_CTRLSTS) .....	66
3.3.4 电源控制寄存器 2 (PWR_CTRL2) .....	68
<b>4 复位和时钟控制(RCC) .....</b>	<b>69</b>
4.1 概述 .....	69
4.2 复位控制单元 .....	69
4.2.1 电源复位 .....	69
4.2.2 系统复位 .....	69
4.2.3 备份域复位 .....	70
4.3 时钟控制单元 .....	71
4.3.1 时钟树 .....	72
4.3.2 HSE 时钟 .....	72
4.3.3 HSI 时钟 .....	73

4.3.4 PLL 时钟 .....	73
4.3.5 LSE 时钟 .....	74
4.3.6 LSI 时钟 .....	74
4.3.7 系统时钟(SYSLCK)选择 .....	74
4.3.8 时钟安全系统(CLKSS) .....	75
4.3.9 LSE 时钟安全系统(LSECSS) .....	75
4.3.10 RTC 时钟 .....	75
4.3.11 看门狗时钟 .....	76
4.3.12 时钟输出(MCO) .....	76
<b>4.4 RCC 寄存器 .....</b>	<b>76</b>
4.4.1 寄存器总览 .....	76
4.4.2 时钟控制寄存器(RCC_CTRL) .....	78
4.4.3 时钟配置寄存器(RCC_CFG) .....	79
4.4.4 时钟中断寄存器(RCC_CLKINT) .....	82
4.4.5 APB2 外设复位寄存器(RCC_APB2PRST) .....	85
4.4.6 APB1 外设复位寄存器(RCC_APB1PRST) .....	86
4.4.7 AHB 外设时钟使能寄存器(RCC_AHBCLKEN) .....	88
4.4.8 APB2 外设时钟使能寄存器 (RCC_APB2PCLKEN) .....	89
4.4.9 APB1 外设时钟使能寄存器(RCC_APB1PCLKEN) .....	90
4.4.10 备份域控制寄存器(RCC_BDCTRL) .....	92
4.4.11 控制/状态寄存器(RCC_CTRLSTS) .....	93
4.4.12 AHB 外设复位寄存器(RCC_AHBPRST) .....	95
4.4.13 时钟配置寄存器 2 (RCC_CFG2) .....	96
4.4.14 保持域控制寄存器(RCC_RDCTRL) .....	97
4.4.15 PLL HSI 配置寄存器(RCC_PLLHSIPRE) .....	98
4.4.16 AHB 外设时钟使能寄存器 1 (RCC_AHB1CLKEN) .....	98
<b>5 GPIO 和 AFIO .....</b>	<b>100</b>
5.1 概述 .....	100
5.2 功能描述 .....	101
5.2.1 模式配置 .....	101
5.2.2 复位后状态 .....	106
5.2.3 单独的位设置和位清除 .....	107
5.2.4 外部中断/唤醒线 .....	107
5.2.5 复用功能 .....	108
5.2.6 外设的 I/O 配置 .....	119
5.2.7 GPIO 锁定机制 .....	121
5.2.8 GPIO 使用注意事项 .....	122
5.3 GPIO 寄存器 .....	122
5.3.1 GPIOA 寄存器总览 .....	122
5.3.2 GPIOB 寄存器总览 .....	124
5.3.3 GPIOC 寄存器总览 .....	126
5.3.4 GPIOD 寄存器总览 .....	127
5.3.5 GPIO 端口模式寄存器 (GPIOx_PMODE) .....	129

5.3.6 GPIO 端口类型定义寄存器 (GPIOx_POTYPE) .....	130
5.3.7 GPIO 翻转率配置寄存器 (GPIOx_SR) .....	130
5.3.8 GPIO 端口上下拉寄存器 (GPIOx_PUPD) .....	131
5.3.9 GPIO 端口输入数据寄存器 (GPIOx_PID) .....	131
5.3.10 GPIO 端口输出数据寄存器 (GPIOx_POD) .....	132
5.3.11 GPIO 端口位设置/清除寄存器 (GPIOx_PBSC) .....	132
5.3.12 GPIO 端口锁定配置寄存器 (GPIOx_PLOCK) .....	133
5.3.13 GPIO 复用功能低配置寄存器 (GPIOx_AFL) .....	134
5.3.14 GPIO 复用功能高配置寄存器 (GPIOx_AFH) .....	135
5.3.15 GPIO 端口位清除寄存器 (GPIOx_PBC) .....	136
5.3.16 GPIO 驱动能力配置寄存器 (GPIOx_DS) .....	136
<b>5.4 AFIO 寄存器 .....</b>	<b>137</b>
5.4.1 AFIO 寄存器总览 .....	137
5.4.2 AFIO 复用重映射配置寄存器 (AFIO_RMP_CFG) .....	139
5.4.3 外部中断配置寄存器 1 (AFIO_EXTI_CFG1) .....	139
5.4.4 外部中断配置寄存器 2 (AFIO_EXTI_CFG2) .....	146
5.4.5 外部中断配置寄存器 3 (AFIO_EXTI_CFG3) .....	153
5.4.6 外部中断配置寄存器 4 (AFIO_EXTI_CFG4) .....	159
5.4.7 5V 耐压配置寄存器 (AFIO_TOL5V_CFG) .....	166
5.4.8 模拟滤波器配置寄存器 1 (AFIO_EFT_CFG1) .....	168
5.4.9 模拟滤波器配置寄存器 2 (AFIO_EFT_CFG2) .....	169
5.4.10 数字毛刺滤波级 (毛刺宽度) 配置寄存器 (AFIO_FILT_CFG) .....	170
5.4.11 数字毛刺滤波器配置寄存器 1 (AFIO_DIGEFT_CFG1) .....	170
5.4.12 数字毛刺滤波器配置寄存器 2 (AFIO_DIGEFT_CFG2) .....	171
<b>6 中断和事件 .....</b>	<b>172</b>
6.1 嵌套向量中断控制器 .....	172
6.1.1 SysTick 校准值寄存器 .....	172
6.1.2 中断和异常向量 .....	172
6.2 外部中断/事件控制器 (EXTI) .....	175
6.2.1 EXTI 简介 .....	175
6.2.2 EXTI 主要特征 .....	175
6.2.3 功能描述 .....	176
6.2.4 EXTI 线映射 .....	178
6.3 EXTI 寄存器 .....	179
6.3.1 EXTI 寄存器总览 .....	179
6.3.2 EXTI 中断屏蔽寄存器 (EXTI_IMASK) .....	179
6.3.3 EXTI 事件屏蔽寄存器 (EXTI_EMASK) .....	180
6.3.4 EXTI 上升沿触发配置寄存器 (EXTI_RT_CFG) .....	180
6.3.5 EXTI 下降沿触发配置寄存器 (EXTI_FT_CFG) .....	181
6.3.6 EXTI 软件中断事件寄存器 (EXTI_SWIE) .....	181
6.3.7 EXTI 中断请求挂起寄存器 (EXTI_PEND) .....	182
6.3.8 EXTI 时间戳触发源选择寄存器 (EXTI_TS_SEL) .....	182

<b>7 DMA 控制器.....</b>	<b>183</b>
7.1 简介 .....	183
7.2 主要特性 .....	183
7.3 功能框图 .....	184
7.4 功能描述 .....	184
7.4.1 DMA 操作 .....	184
7.4.2 通道优先级和仲裁器 .....	185
7.4.3 DMA 通道和传输数量 .....	185
7.4.4 可编程的数据位宽 .....	185
7.4.5 外设/内存地址递增 .....	187
7.4.6 通道配置流程 .....	187
7.4.7 流量控制 .....	187
7.4.8 循环模式 .....	188
7.4.9 错误管理 .....	188
7.4.10 中断 .....	188
7.4.11 DMA 请求映射 .....	189
7.5 DMA 寄存器 .....	191
7.5.1 DMA 寄存器总览 .....	191
7.5.2 DMA 中断状态寄存器 (DMA_INTSTS) .....	193
7.5.3 DMA 中断标志清除寄存器 (DMA_INTCLR) .....	194
7.5.4 DMA 通道 x 配置寄存器 (DMA_CHCFGx) .....	194
7.5.5 DMA 通道 x 传输数量寄存器 (DMA_TXNUMx) .....	196
7.5.6 DMA 通道 x 外设基址寄存器 (DMA_PADDRx) .....	197
7.5.7 DMA 通道 x 存储器基址寄存器 (DMA_MADDRx) .....	197
7.5.8 DMA 通道 x 请求选择寄存器 (DMA_CHSELx) .....	198
<b>8 CRC 计算单元.....</b>	<b>200</b>
8.1 简介 .....	200
8.2 主要特性 .....	200
8.2.1 CRC32 .....	200
8.2.2 CRC16 .....	200
8.3 CRC 功能描述 .....	201
8.3.1 CRC32 .....	201
8.3.2 CRC16 .....	201
8.4 CRC 寄存器 .....	202
8.4.1 CRC 寄存器总览 .....	202
8.4.2 CRC32 数据寄存器 (CRC_CRC32DAT) .....	202
8.4.3 CRC32 独立数据寄存器 (CRC_CRC32IDAT) .....	202
8.4.4 CRC32 控制寄存器 (CRC_CRC32CTRL) .....	203
8.4.5 CRC16 控制寄存器 (CRC_CRC16CTRL) .....	203
8.4.6 CRC16 待校验寄存器 (CRC_CRC16DAT) .....	204
8.4.7 CRC 循环冗余校验码寄存器 (CRC_CRC16D) .....	204
8.4.8 LRC 校验值寄存器 (CRC_LRC) .....	205

9 高级控制定时器 (TIM1 和 TIM8) .....	206
9.1 TIM1 和 TIM8 简介.....	206
9.2 TIM1 和 TIM8 主要特性.....	206
9.3 TIM1 和 TIM8 功能描述.....	208
9.3.1 时基单元 .....	208
9.3.2 计数器模式.....	209
9.3.3 重复计数器 .....	218
9.3.4 时钟选择 .....	220
9.3.5 捕获/比较通道.....	224
9.3.6 输入捕获模式.....	226
9.3.7 PWM 输入模式 .....	227
9.3.8 强制输出模式.....	228
9.3.9 输出比较模式 .....	228
9.3.10 PWM 模式 .....	230
9.3.11 单脉冲模式 .....	232
9.3.12 在外部事件上清除 OCxREF 信号 .....	234
9.3.13 互补输出和死区插入 .....	234
9.3.14 刹车功能 .....	236
9.3.15 调试模式 .....	238
9.3.16 TIMx 定时器和外部触发的同步 .....	238
9.3.17 定时器同步 .....	241
9.3.18 产生六步 PWM 输出.....	241
9.3.19 编码器接口模式 .....	242
9.3.20 与霍尔传感器的接口.....	244
9.4 TIMx 寄存器描述 (x=1,8) .....	246
9.4.1 寄存器总览 .....	246
9.4.2 控制寄存器 1 (TIMx_CTRL1) .....	249
9.4.3 控制寄存器 2 (TIMx_CTRL2) .....	253
9.4.4 从模式控制寄存器 (TIMx_SMCTRL) .....	256
9.4.5 DMA/中断使能寄存器 (TIMx_DINTEN) .....	259
9.4.6 状态寄存器 (TIMx_STS) .....	261
9.4.7 事件产生寄存器 (TIMx_EVTGEN) .....	264
9.4.8 捕获/比较模式寄存器 1 (TIMx_CCMOD1) .....	265
9.4.9 捕获/比较模式寄存器 2 (TIMx_CCMOD2) .....	269
9.4.10 捕获/比较使能寄存器 (TIMx_CCEN) .....	272
9.4.11 计数器 (TIMx_CNT) .....	276
9.4.12 预分频器 (TIMx_PSC) .....	276
9.4.13 自动重装载寄存器 (TIMx_AR) .....	277
9.4.14 重复计数寄存器 (TIMx_REPCNT) .....	277
9.4.15 捕获/比较寄存器 1 (TIMx_CCDAT1) .....	278
9.4.16 捕获/比较寄存器 2 (TIMx_CCDAT2) .....	279
9.4.17 捕获/比较寄存器 3 (TIMx_CCDAT3) .....	280
9.4.18 捕获/比较寄存器 4 (TIMx_CCDAT4) .....	280

9.4.19 刹车和死区寄存器 (TIMx_BKDT) .....	281
9.4.20 DMA 控制寄存器 (TIMx_DCTRL) .....	284
9.4.21 连续模式的DMA 地址 (TIMx_DADDR) .....	284
9.4.22 捕获/比较寄存器 (TIMx_CCMOD3) .....	285
9.4.23 捕获/比较寄存器 5 (TIMx_CC DAT5) .....	286
9.4.24 捕获/比较寄存器 6 (TIMx_CC DAT6) .....	286
9.4.25 捕获/比较寄存器 7 (TIMx_CC DAT7) .....	287
9.4.26 捕获/比较寄存器 8 (TIMx_CC DAT8) .....	287
9.4.27 捕获/比较寄存器 9 (TIMx_CC DAT9) .....	288
9.4.28 刹车滤波 (TIMx_BRKFILT) .....	288
<b>10 通用定时器 (TIM2、TIM3、TIM4 和 TIM5) .....</b>	<b>290</b>
10.1 TIM2、TIM3、TIM4 和 TIM5 简介 .....	290
10.2 TIM2、TIM3、TIM4 和 TIM5 主要特性 .....	290
10.3 TIM2、TIM3、TIM4、TIM5 功能描述 .....	291
10.3.1 时基单元 .....	291
10.3.2 计数器模式 .....	292
10.3.3 时钟选择 .....	297
10.3.4 捕获/比较通道 .....	301
10.3.5 输入捕获模式 .....	304
10.3.6 PWM 输入模式 .....	305
10.3.7 强制输出模式 .....	306
10.3.8 输出比较模式 .....	306
10.3.9 PWM 模式 .....	308
10.3.10 单脉冲模式 .....	310
10.3.11 在外部事件上清除 OCxREF 信号 .....	312
10.3.12 调试模式 .....	312
10.3.13 TIMx 定时器和外部触发的同步 .....	312
10.3.14 定时器同步 .....	316
10.3.15 编码器接口模式 .....	320
10.3.16 与霍尔传感器的接口 .....	322
10.4 TIMx 寄存器描述 (x=2, 3, 4 and 5) .....	323
10.4.1 寄存器总览 .....	323
10.4.2 控制寄存器 1 (TIMx_CTRL1) .....	325
10.4.3 控制寄存器 2 (TIMx_CTRL2) .....	328
10.4.4 从模式控制寄存器 (TIMx_SMCTRL) .....	329
10.4.5 DMA/中断使能寄存器 (TIMx_DINTEN) .....	332
10.4.6 状态寄存器 (TIMx_STS) .....	334
10.4.7 事件产生寄存器 (TIMx_EVTGEN) .....	336
10.4.8 捕获/比较模式寄存器 1 (TIMx_CCMOD1) .....	337
10.4.9 捕获/比较模式寄存器 2 (TIMx_CCMOD2) .....	341
10.4.10 捕获/比较使能寄存器 (TIMx_CCEN) .....	343
10.4.11 计数器 (TIMx_CNT) .....	345
10.4.12 预分频器 (TIMx_PSC) .....	345

10.4.13 自动重装载寄存器 (TIMx_AR) .....	345
10.4.14 捕获/比较寄存器 1 (TIMx_CC DAT1) .....	346
10.4.15 捕获/比较寄存器 2 (TIMx_CC DAT2) .....	346
10.4.16 捕获/比较寄存器 3 (TIMx_CC DAT3) .....	347
10.4.17 捕获/比较寄存器 4 (TIMx_CC DAT4) .....	347
10.4.18 DMA 控制寄存器 (TIMx_DCTRL) .....	348
10.4.19 连续模式的 DMA 地址 (TIMx_DADDR) .....	349
10.4.20 通道1 滤波寄存器(TIMx_C1FILT) .....	350
10.4.21 通道2 滤波寄存器(TIMx_C2FILT) .....	351
10.4.22 通道3 滤波寄存器(TIMx_C3FILT) .....	352
10.4.23 通道4 滤波寄存器(TIMx_C4FILT) .....	353
10.4.24 通道滤波输出寄存器 (TIMx_FILTO) .....	354
<b>11 基本定时器 (TIM6).....</b>	<b>355</b>
11.1 基本定时器简介 .....	355
11.2 基本定时器主要特性 .....	355
11.3 基础定时器描述 .....	355
11.3.1 时基单元 .....	355
11.3.2 计数模式 .....	356
11.3.3 时钟选择 .....	359
11.3.4 调试模式 .....	359
<b>11.4 TIMx 寄存器描述(x=6) .....</b>	<b>359</b>
11.4.1 寄存器总览 .....	360
11.4.2 控制寄存器 1 (TIMx_CTRL1) .....	360
11.4.3 DMA/中断使能寄存器 (TIMx_DINTEN) .....	361
11.4.4 状态寄存器 (TIMx_STS) .....	362
11.4.5 事件产生寄存器 (TIMx_EVTGEN) .....	362
11.4.6 计数器 (TIMx_CNT) .....	362
11.4.7 预分频器 (TIMx_PSC) .....	363
11.4.8 自动重装载寄存器 (TIMx_AR) .....	363
<b>12 低功耗定时器 (LPTIM).....</b>	<b>364</b>
12.1 简介 .....	364
12.2 主要特性 .....	364
12.3 功能框图 .....	365
12.4 功能描述 .....	365
12.4.1 LPTIM 复位和时钟 .....	365
12.4.2 分频系数 .....	366
12.4.3 毛刺滤波器 .....	366
12.4.4 开启定时器 .....	367
12.4.5 多路触发器 .....	367
12.4.6 工作模式 .....	368
12.4.7 波形发生器 .....	370
12.4.8 寄存器更新 .....	371

12.4.9 计数器模式 .....	372
12.4.10 编码器模式 .....	372
12.4.11 非正交编码器模式 .....	374
12.4.12 超时功能 .....	375
12.4.13 LPTIM 中断 .....	375
<b>12.5 LPTIM 寄存器 .....</b>	<b>377</b>
12.5.1 LPTIM 寄存器总览 .....	377
12.5.2 LPTIM 中断状态寄存器(LPTIM_INTSTS) .....	377
12.5.3 LPTIM 中断清除寄存器(LPTIM_INTCLR) .....	378
12.5.4 LPTIM 中断使能寄存器(LPTIM_INTEN) .....	379
12.5.5 LPTIM 配置寄存器(LPTIM_CFG) .....	380
12.5.6 LPTIM 控制寄存器(LPTIM_CTRL) .....	382
12.5.7 LPTIM 比较寄存器(LPTIM_COMP) .....	383
12.5.8 LPTIM 自动重载寄存器(LPTIM_ARR) .....	383
12.5.9 LPTIM 计数寄存器(LPTIM_CNT) .....	384
12.5.10 LPTIM 选项寄存器(LPTIM_OPT) .....	384
<b>13 独立看门狗 (IWDG) .....</b>	<b>386</b>
13.1 简介 .....	386
13.2 主要特征 .....	386
13.3 功能描述 .....	387
13.3.1 寄存器访问保护 .....	387
13.3.2 调试模式 .....	388
13.3.3 IWDG 冻结 .....	388
13.4 用户界面 .....	388
13.4.1 操作流程 .....	388
13.4.2 IWDG 配置流程 .....	389
13.5 IWDG 寄存器 .....	389
13.5.1 IWDG 寄存器总览 .....	389
13.5.2 IWDG 密钥寄存器 (IWDG_KEY) .....	390
13.5.3 IWDG 预分频寄存器 (IWDG_PREDIV) .....	390
13.5.4 IWDG 重装载寄存器 (IWDG_RELV) .....	391
13.5.5 IWDG 状态寄存器 (IWDG_STS) .....	391
<b>14 窗口看门狗 (WWDG) .....</b>	<b>393</b>
14.1 简介 .....	393
14.2 主要特征 .....	393
14.3 功能描述 .....	393
14.4 刷新看门狗和中断产生的时许 .....	394
14.5 调试模式 .....	395
14.6 用户界面 .....	395
14.6.1 WWDG 配置流程 .....	395
14.7 WWDG 寄存器 .....	396
14.7.1 WWDG 寄存器总览 .....	396

14.7.2 WWDG 控制寄存器 (WWDG_CTRL) .....	396
14.7.3 WWDG 配置寄存器 (WWDG_CFG) .....	397
14.7.4 WWDG 状态寄存器 (WWDG_STS) .....	397
<b>15 模数转换 (ADC).....</b>	<b>398</b>
15.1 简述 .....	398
15.2 主要特征.....	398
15.3 功能描述 .....	399
15.3.1 ADC 时钟 .....	400
15.3.2 ADC 开关控制.....	400
15.3.3 通道选择 .....	401
15.3.4 内部通道 .....	402
15.3.5 单次转换模式 .....	403
15.3.6 连续转换模式 .....	403
15.3.7 时序图 .....	403
15.3.8 模拟看门狗 .....	404
15.3.9 扫描模式 .....	405
15.3.10 注入通道管理 .....	405
15.3.11 间断模式 .....	406
15.4 校准 .....	407
15.5 数据对齐 .....	407
15.6 通道采样时间.....	409
15.7 外部触发转换 .....	409
15.8 DMA 请求.....	410
15.9 温度传感器 .....	410
15.9.1 测量温度值 .....	411
15.10 中断 .....	412
15.11 ADC 寄存器 .....	412
15.11.1 ADC 寄存器总览 .....	412
15.11.2 ADC 状态寄存器(ADC_STS) .....	413
15.11.3 ADC 控制寄存器 1(ADC_CTRL1) .....	414
15.11.4 ADC 控制寄存器 2(ADC_CTRL2) .....	416
15.11.5 ADC 采样时间寄存器 1(ADC_SAMPT1) .....	418
15.11.6 ADC 采样时间寄存器 2(ADC_SAMPT2) .....	419
15.11.7 ADC 注入通道数据偏移寄存器 x(ADC_JOFFSETx) (x=1...4) .....	419
15.11.8 ADC 看门狗高阈值寄存器(ADC_WDGHIGH) .....	420
15.11.9 ADC 看门狗低阈值寄存器(ADC_WDGLOW) .....	420
15.11.10 ADC 规则序列寄存器 1(ADC_RSEQ1) .....	420
15.11.11 ADC 规则序列寄存器 2(ADC_RSEQ2) .....	421
15.11.12 ADC 规则序列寄存器 3(ADC_RSEQ3) .....	421
15.11.13 ADC 注入序列寄存器(ADC_JSEQ) .....	422
15.11.14 ADC 注入数据寄存器 x(ADC_JDATx) (x= 1...4) .....	423
15.11.15 ADC 规则数据寄存器(ADC_DAT) .....	423
15.11.16 ADC 差分模式选择寄存器(ADC_DIFSEL) .....	423

15.11.17 ADC 校正因子(ADC_CALFACT) .....	424
15.11.18 ADC 控制寄存器 3(ADC_CTRL3) .....	424
15.11.19 ADC 采样时间寄存器 3(ADC_SAMPT3).....	426
<b>16 比较器 (COMP) .....</b>	<b>427</b>
16.1 COMP 系统连接框图 .....	427
16.2 COMP 特性 .....	428
16.3 COMP 配置流程 .....	428
16.4 COMP 工作模式 .....	429
16.4.1 窗口模式 .....	429
16.4.2 独立比较器 .....	429
16.5 比较器互联关系 .....	429
16.6 中断 .....	430
16.7 COMP 寄存器 .....	431
16.7.1 COMP 寄存器总览 .....	431
16.7.2 COMP 中断使能寄存器 (COMP_INTEN) .....	432
16.7.3 COMP 窗口比较寄存器 (COMP_WINMODE).....	432
16.7.4 COMP 锁寄存器(COMP_LOCK) .....	433
16.7.5 COMP1 控制寄存器 (COMP1_CTRL).....	434
16.7.6 COMP1 滤波控制寄存器 (COMP1_FILC) .....	435
16.7.7 COMP1 滤波器分频寄存器 (COMP1_FILP).....	436
16.7.8 COMP2 控制寄存器 (COMP2_CTRL).....	436
16.7.9 COMP2 滤波控制寄存器 (COMP2_FILC) .....	438
16.7.10 COMP2 滤波器分频寄存器 (COMP2_FILP).....	439
16.7.11 COMP2 输出选择寄存器 (COMP2_OSEL) .....	439
16.7.12 COMP3 控制寄存器 (COMP3_CTRL) .....	440
16.7.13 COMP3 滤波控制寄存器 (COMP3_FILC) .....	442
16.7.14 COMP3 滤波器分频寄存器 (COMP3_FILP).....	442
16.7.15 COMP 参考电压寄存器(COMP_VREFSCL).....	443
16.7.16 COMP 测试寄存器 (COMP_TEST) .....	443
16.7.17 COMP 中断状态寄存器 (COMP_INTSTS).....	444
<b>17 I<sup>2</sup>C 接口 .....</b>	<b>445</b>
17.1 简介 .....	445
17.2 特性 .....	445
17.3 功能描述 .....	445
17.3.1 SDA/SCL 控制 .....	445
17.3.2 软件通信流程 .....	446
17.3.3 错误条件 .....	455
17.3.4 DMA 应用 .....	456
17.3.5 包错误校验 (PEC) .....	457
17.3.6 噪声滤波 .....	458
17.3.7 SMBus .....	458
17.4 调试模式 .....	460

17.5 中断请求 .....	460
17.6 I <sup>2</sup> C 寄存器描述 .....	462
17.6.1 I <sup>2</sup> C 寄存器总览 .....	462
17.6.2 I <sup>2</sup> C 控制寄存器 1 (I2C_CTRL1) .....	462
17.6.3 I <sup>2</sup> C 控制寄存器 2 (I2C_CTRL2) .....	464
17.6.4 I <sup>2</sup> C 自身地址寄存器 1 (I2C_OADDR1) .....	466
17.6.5 I <sup>2</sup> C 自身地址寄存器 2 (I2C_OADDR2) .....	466
17.6.6 I <sup>2</sup> C 数据寄存器 (I2C_DAT) .....	467
17.6.7 I <sup>2</sup> C 状态寄存器 1 (I2C_STS1) .....	467
17.6.8 I <sup>2</sup> C 状态寄存器 2 (I2C_STS2) .....	470
17.6.9 时钟控制寄存器(I2C_CLKCTRL) .....	471
17.6.10 上升时间寄存器 (I2C_TMRISE) .....	472
<b>18 通用同步异步收发器 (USART) .....</b>	<b>474</b>
18.1 简介 .....	474
18.2 主要特性 .....	474
18.3 功能框图 .....	475
18.4 功能描述 .....	475
18.4.1 USART 帧格式 .....	476
18.4.2 发送器 .....	477
18.4.3 接收器 .....	479
18.4.4 分数波特率计算 .....	483
18.4.5 USART 接收器容忍时钟的变化 .....	484
18.4.6 校验控制 .....	485
18.4.7 DMA 通信 .....	485
18.4.8 硬件流控 .....	487
18.4.9 多处理器通信 .....	489
18.4.10 同步模式 .....	491
18.4.11 单线半双工模式 .....	493
18.4.12 串行 IrDA 红外编解码模式 .....	494
18.4.13 LIN 模式 .....	495
18.4.14 智能卡模式 (ISO7816) .....	498
18.5 中断请求 .....	500
18.6 模式配置 .....	501
18.7 USART 寄存器 .....	502
18.7.1 USART 寄存器总览 .....	502
18.7.2 USART 状态寄存器 (USART_STS) .....	503
18.7.3 USART 数据寄存器(USART_DAT) .....	505
18.7.4 USART 波特率配置寄存器 (USART_BRCF) .....	505
18.7.5 USART 控制寄存器 1 (USART_CTRL1) .....	506
18.7.6 USART 控制寄存器 2 (USART_CTRL2) .....	507
18.7.7 USART 控制寄存器 3 (USART_CTRL3) .....	509
18.7.8 USART 保护时间和预分频寄存器(USART_GTP) .....	510

<b>19 串行外设接口/内置音频总线 (SPI/I<sup>2</sup>S).....</b>	<b>512</b>
19.1 SPI/I <sup>2</sup> S 介绍.....	512
19.2 SPI 和 I <sup>2</sup> S 主要特征.....	512
19.2.1 SPI 特征 .....	512
19.2.2 I <sup>2</sup> S 特征 .....	512
19.3 SPI 功能描述 .....	513
19.3.1 通用描述 .....	513
19.3.2 SPI 工作模式 .....	516
19.3.3 状态标志 .....	522
19.3.4 关闭 SPI .....	523
19.3.5 使用 DMA 进行 SPI 通讯.....	523
19.3.6 CRC 计算 .....	524
19.3.7 错误标志位 .....	525
19.3.8 SPI 中断 .....	526
19.4 I <sup>2</sup> S 功能描述 .....	527
19.4.1 支持的音频协议 .....	528
19.4.2 时钟发生器 .....	534
19.4.3 I <sup>2</sup> S 发送和接收流程 .....	535
19.4.4 状态标识 .....	537
19.4.5 错误标志位 .....	538
19.4.6 I <sup>2</sup> S 中断 .....	538
19.4.7 DMA 功能.....	539
19.5 SPI 和 I <sup>2</sup> S 寄存器描述.....	539
19.5.1 SPI 寄存器总览 .....	539
19.5.2 SPI 控制寄存器 1 (SPI_CTRL1) (不能用于 I <sup>2</sup> S 模式).....	539
19.5.3 SPI 控制寄存器 2 (SPI_CTRL2).....	541
19.5.4 SPI 状态寄存器(SPI_STS) .....	542
19.5.5 SPI 数据寄存器(SPI_DAT) .....	543
19.5.6 SPI CRC 多项式寄存器 (SPI_CRCPOLY) (不能用于 I <sup>2</sup> S 模式).....	544
19.5.7 SPI 接收 CRC 寄存器 (SPI_CRCRDAT) (不能用于 I <sup>2</sup> S 模式) .....	544
19.5.8 SPI 发送 CRC 寄存器 (SPI_CRCTDAT) .....	545
19.5.9 SPI_I <sup>2</sup> S 配置寄存器 (SPI_I2SCFG) .....	545
19.5.10 SPI_I <sup>2</sup> S 预分频寄存器 (SPI_I2SPREDIV) .....	547
<b>20 实时时钟(RTC) .....</b>	<b>548</b>
20.1 简介 .....	548
20.1.1 主要特性 .....	549
20.2 RTC 功能描述.....	550
20.2.1 RTC 框图 .....	550
20.2.2 RTC 控制的 GPIO .....	551
20.2.3 RTC 寄存器写保护 .....	551
20.2.4 RTC 时钟和预分频 .....	551
20.2.5 RTC 日历 .....	552

20.2.6 日历初始化和配置 .....	552
20.2.7 日历读取 .....	553
20.2.8 校准时钟输出 .....	553
20.2.9 可编程闹钟 .....	554
20.2.10 闹钟配置 .....	554
20.2.11 闹钟输出 .....	554
20.2.12 周期性自动唤醒 .....	554
20.2.13 唤醒定时器配置 .....	555
20.2.14 时间戳功能 .....	555
20.2.15 入侵检测 .....	555
20.2.16 夏时令功能配置 .....	556
20.2.17 RTC 亚秒寄存器位移操作 .....	556
20.2.18 RTC 数字时钟精密校准 .....	556
20.2.19 RTC 低功耗模式 .....	558
<b>20.3 RTC 寄存器 .....</b>	<b>558</b>
20.3.1 RTC 寄存器总览 .....	558
20.3.2 RTC 日历时间寄存器 (RTC_TSH) .....	559
20.3.3 RTC 日历日期寄存器 (RTC_DATE) .....	560
20.3.4 RTC 控制寄存器(RTC_CTRL) .....	561
20.3.5 RTC 初始状态寄存器 (RTC_INITSTS) .....	563
20.3.6 RTC 预分频寄存器(RTC_PRE) .....	565
20.3.7 RTC 唤醒定时器寄存器(RTC_WKUPT) .....	565
20.3.8 RTC 闹钟 A 寄存器(RTC_ALARMA) .....	566
20.3.9 RTC 闹钟 B 寄存器 (RTC_ALARMB) .....	567
20.3.10 RTC 写保护寄存器(RTC_WRP) .....	568
20.3.11 RTC 亚秒寄存器(RTC_SUBS) .....	568
20.3.12 RTC 平移控制寄存器(RTC_SCTRL) .....	569
20.3.13 RTC 时间戳时间寄存器 (RTC_TST) .....	569
20.3.14 RTC 时间戳日期寄存器 (RTC_TSD) .....	570
20.3.15 RTC 时间戳亚秒寄存器(RTC_TSSS) .....	570
20.3.16 RTC 校准寄存器(RTC_CALIB) .....	571
20.3.17 RTC 入侵配置寄存器 (RTC_TMPCFG) .....	572
20.3.18 RTC 闹钟 A 亚秒寄存器(RTC_ALRMASS) .....	574
20.3.19 RTC 闹钟 B 亚秒寄存器 (RTC_ALRMBSS) .....	575
20.3.20 RTC 选项寄存器 (RTC_OPT) .....	576
20.3.21 RTC 备份寄存器 (RTC_BKP(1~20)) .....	576
<b>21 蜂鸣器 .....</b>	<b>577</b>
21.1 介绍 .....	577
21.2 功能描述 .....	577
21.2.1 主要特点 .....	577
21.2.2 设定蜂鸣器频率预分频因子 .....	577
21.2.3 旁路时钟 .....	577
21.2.4 输出占空比 .....	578

21.3 蜂鸣器寄存器 .....	578
21.3.1 BEEPER 寄存器总览 .....	578
21.3.2 蜂鸣器控制寄存器(BEEPER_CTRL).....	578
<b>22 控制器局域网(CAN).....</b>	<b>580</b>
22.1 CAN 简介 .....	580
22.2 CAN 的主要特性.....	580
22.3 CAN 整体介绍.....	580
22.3.1 CAN 模块.....	581
22.3.2 CAN 工作模式.....	581
22.3.3 发送邮箱 .....	583
22.3.4 接收过滤器 .....	583
22.3.5 接收 FIFO .....	583
22.3.6 CAN 测试模式.....	584
22.3.7 CAN 调试模式.....	587
22.4 CAN 功能描述.....	587
22.4.1 发送处理 .....	587
22.4.2 时间触发通信模式 .....	588
22.4.3 非自动重传模式 .....	588
22.4.4 接收管理 .....	588
22.4.5 标识符过滤 .....	589
22.4.6 消息存储 .....	593
22.4.7 位时序 .....	594
22.5 CAN 中断.....	597
22.5.1 错误管理 .....	598
22.5.2 总线关闭恢复 .....	598
22.6 CAN 配置流程.....	598
22.7 CAN 寄存器 .....	599
22.7.1 寄存器描述 .....	600
22.7.2 CAN 寄存器总览.....	600
22.7.3 CAN 控制和状态寄存器.....	604
22.7.4 CAN 邮箱寄存器.....	613
22.7.5 CAN 过滤器寄存器.....	618
<b>23 调试支持 (DBG).....</b>	<b>622</b>
23.1 概述 .....	622
23.2 JTAG/SW 功能.....	623
23.2.1 切换 JTAG/SW 接口 .....	623
23.2.2 引脚分配 .....	623
23.3 MCU 调试功能 .....	624
23.3.1 低功耗模式支持 .....	624
23.3.2 外设调试支持 .....	624
23.4 寄存器 .....	624
23.4.1 DBG 寄存器总览 .....	624

23.4.2 ID 寄存器 (DBG_ID).....	625
23.4.3 调试控制寄存器 (DBG_CTRL).....	625
<b>24 唯一设备序列号 (UID) .....</b>	<b>628</b>
24.1 简介 .....	628
24.2 UID 寄存器 .....	628
24.3 UCID 寄存器 .....	628
<b>25 版本历史 .....</b>	<b>629</b>
<b>26 声明 .....</b>	<b>632</b>

## 表目录

表 2-1 外设寄存器地址列表 .....	31
表 2-2 启动模式列表 .....	34
表 2-3 存储总线地址列表 .....	35
表 2-4 选项字节列表 .....	38
表 2-5 读保护配置列表 .....	40
表 2-6 存储区读写擦 <sup>(1)</sup> 权限控制表 .....	42
表 2-7 FLASH 寄存器总览 .....	47
表 3-1 功耗模式 .....	59
表 3-2 模块运行状态 .....	60
表 3-3 PWR 寄存器总览 .....	64
表 4-1 RCC 寄存器总览 .....	76
表 5-1 I/O 口配置表 .....	101
表 5-2 I/O 管脚功能特性列表 .....	102
表 5-3 复位后的特殊引脚 .....	106
表 5-4 调试接口信号 .....	110
表 5-5 调试端口映像 .....	110
表 5-6 ADC 外部触发注入转换复用功能重映射 .....	110
表 5-7 ADC 外部触发规则转换复用功能重映射 .....	111
表 5-8 TIM1 复用功能重映射 .....	111
表 5-9 TIM2 复用功能重映射 .....	112
表 5-10 TIM3 复用功能重映射 .....	112
表 5-11 TIM4 复用功能重映射 .....	112
表 5-12 TIM5 复用功能重映射 .....	112
表 5-13 TIM8 复用功能重映射 .....	113
表 5-14 LPTIM 复用功能重映射 .....	113
表 5-15 CAN 复用功能重映射 .....	114
表 5-16 USART1 复用功能重映射 .....	114
表 5-17 USART2 复用功能重映射 .....	114
表 5-18 UART3 复用功能重映射 .....	115
表 5-19 UART4 复用功能重映射 .....	115

表 5-20 I2C1 管脚重映射 .....	116
表 5-21 I2C2 管脚重映射 .....	116
表 5-22 SPI1/I2S1 管脚重映射 .....	117
表 5-23 SPI2/I2S2 管脚重映射 .....	117
表 5-24 COMP1 管脚重映射 .....	118
表 5-25 COMP2 管脚重映射 .....	118
表 5-26 COMP3 管脚重映射 .....	118
表 5-27 EVENTOUT 管脚重映射 .....	118
表 5-28 BEEPER 管脚重映射 .....	118
表 5-29 JTAG/SWD 管脚重映射 .....	119
表 5-30 TIMESTAMP 管脚重映射 .....	119
表 5-31 RTC_REFIN 管脚重映射 .....	119
表 5-32 MCO 管脚重映射 .....	119
表 5-33 ADC .....	119
表 5-34 TIM1/TIM8 .....	119
表 5-35 TIM2/3/4/5 .....	120
表 5-36 LPTIM .....	120
表 5-37 CAN .....	120
表 5-38 USART .....	120
表 5-39 UART .....	120
表 5-40 I2C .....	120
表 5-41 SPI-I2S .....	121
表 5-42 JTAG/SWD .....	121
表 5-43 RTC .....	121
表 5-44 COMP .....	121
表 5-45 EVENT_OUT .....	121
表 5-46 其他 .....	121
表 5-47 GPIOA 寄存器总览 .....	122
表 5-48 GPIOB 寄存器总览 .....	124
表 5-49 GPIOC 寄存器总览 .....	126
表 5-50 GPIOD 寄存器总览 .....	127
表 5-51 AFIO 寄存器总览 .....	137

表 6-1 向量表.....	172
表 6-2 EXTI 寄存器总览.....	179
表 7-1 可编程的数据宽度和大小端操作(当 PINC = MINC = 1).....	185
表 7-2 流量控制表.....	188
表 7-3 DMA 中断请求.....	189
表 7-4 DMA 请求映射.....	189
表 7-5 DMA 寄存器总览.....	191
表 8-1 CRC 寄存器总览.....	202
表 9-1 计数方向与编码器信号的关系 .....	243
表 9-2 TIM1 和 TIM8 寄存器总览 .....	246
表 9-3 TIMx 内部触发连接.....	259
表 9-4 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位 .....	275
表 10-1 计数方向与编码器信号的关系 .....	321
表 10-2 TIM2、TIM3、TIM4 and TIM5 寄存器总览 .....	323
表 10-3 TIMx 内部触发连接.....	332
表 10-4 标准 OCx 的输出控制位.....	344
表 11-1 寄存器总览.....	360
表 12-1 预分频因子.....	366
表 12-2 LPTIM_CFG.TRGSEL[3:0]对应的 9 个触发输入.....	367
表 12-3 编码器计数的场景 .....	373
表 12-4 中断事件 .....	376
表 12-5 LPTIM 寄存器总览 .....	377
表 13-1 IWDG 计数最大和最小复位时间 .....	388
表 13-2 IWDG 寄存器总览 .....	389
表 14-1 WWDG 的最大和最小计数时间 .....	395
表 14-2 WWDG 寄存器总览 .....	396
表 15-1 ADC 引脚说明 .....	399
表 15-2 模拟看门狗通道选择 .....	404
表 15-3 数据右对齐 .....	407
表 15-4 数据左对齐 .....	408
表 15-5 ADC 用于规则通道的外部触发 .....	409
表 15-6 ADC 用于注入通道的外部触发 .....	409

表 15-7 ADC 中断.....	412
表 15-8 ADC 寄存器总览.....	412
表 16-1 COMP 寄存器总览.....	431
表 17-1 SMBus 与 I <sup>2</sup> C 的比较.....	459
表 17-2 I <sup>2</sup> C 中断请求.....	461
表 17-3 I <sup>2</sup> C 寄存器总览.....	462
表 18-1 停止位配置.....	477
表 18-2 噪声检测的数据采样.....	482
表 18-3 设置波特率时的误差计算.....	484
表 18-4 当 DIV_Decimal =0 时, USART 接收器的容忍度.....	484
表 18-5 当 DIV_Decimal !=0 时, USART 接收器的容忍度.....	485
表 18-6 帧格式.....	485
表 18-7 USART 中断请求 .....	500
表 18-8 USART 模式设置 <sup>(1)</sup> .....	501
表 18-9 USART 寄存器总览 .....	502
表 19-1 SPI 中断请求 .....	526
表 19-2 使用标准的 8MHz HSE 时钟得到精确的音频频率.....	535
表 19-3 I <sup>2</sup> S 中断请求 .....	538
表 19-4 SPI 寄存器总览 .....	539
表 20-1 RTC 功能支持 .....	549
表 20-2 RTC 寄存器总览 .....	558
表 21-1 蜂鸣器支持的最大和最小频率以及相应的配置.....	577
表 21-2 BEEPER 寄存器总览 .....	578
表 22-1 过滤器编号示例 .....	592
表 22-2 发送邮箱寄存器列表 .....	593
表 22-3 接收邮箱寄存器列表 .....	594
表 22-4 CAN 寄存器总览.....	600
表 23-1 调试端口引脚 .....	623
表 23-2 DBG 寄存器总览.....	625

## 图目录

图 2-1 总线架构图 .....	28
图 2-2 总线地址映射图 .....	30
图 3-1 电源框图 .....	57
图 3-2 上电复位/掉电复位波形图 .....	58
图 3-3 PVD 阈值图 .....	58
图 4-1 复位电路 .....	70
图 4-2 时钟树 .....	72
图 4-3 HSE 时钟源 .....	73
图 5-1 5V 耐压 I/O 结构 .....	101
图 5-2 复用功能模式 .....	103
图 5-3 浮空输入/上拉/下拉配置模式 .....	104
图 5-4 输出模式 .....	105
图 5-5 高阻抗模拟输入配置 .....	106
图 5-6 EXTI 全局中断 .....	108
图 6-1 EXTI 功能框图 .....	176
图 6-2 外部中断 GPIO 映射 .....	178
图 7-1 DMA 框图 .....	184
图 8-1 CRC 计算单元框图 .....	201
图 9-1 TIM1 框图 .....	207
图 9-2 TIM8 框图 .....	207
图 9-3 当预分频的参数从 1 到 4, 计数器的时序图 .....	209
图 9-4 当内部时钟分频因子 = 2/N 时, 向上计数的时序图 .....	210
图 9-5 当 ARPEN=0/1 产生更新事件时, 向上计数的时序图 .....	211
图 9-6 内部时钟分频因子 = 2/N 时, 向下计数时序图 .....	212
图 9-7 内部时钟分频因子 = 2/N, 中央对齐时序图 .....	213
图 9-8 包含计数器上溢和下溢的中央对齐时序图(ARPEN=1) .....	214
图 9-9 非对称模式对应的输出波形 .....	215
图 9-10 CC DATx(x=4,7,8,9), 当 DIR = 0 时触发 ADC .....	216
图 9-11 CC DATx(x=4,7,8,9), 当 DIR = 1 时触发 ADC .....	217
图 9-12 CC DATx(x=4,7,8,9), 当 DIR = 1 或 DIR = 0 时触发 ADC .....	218

图 9-13 向下计数模式下的重复计数时序图 .....	219
图 9-14 向上计数模式下的重复计数时序列图 .....	219
图 9-15 中央对齐模式下的重复计数时序图 .....	220
图 9-16 正常模式下的控制电路, 内部时钟除以 1 .....	221
图 9-17 TI2 外部时钟连接示例 .....	221
图 9-18 外部时钟模式 1 的控制电路 .....	222
图 9-19 外部触发输入框图 .....	223
图 9-20 外部时钟模式 2 的控制电路 .....	223
图 9-21 捕获/比较通道 (例如: 通道 1 输入级) .....	224
图 9-22 捕获/比较通道 1 主电路 .....	225
图 9-23 通道 x 的输出部分 (TIM1 的 x=1,2,3,4; TIM8 的 x=1,2,3。以通道 1 为例子) .....	225
图 9-24 通道 x 的输出部分 (对于 TIM8, x=4) .....	226
图 9-25 PWM 输入模式时序 .....	228
图 9-26 输出比较模式, 开启 OC1.....	229
图 9-27 中央对齐的 PWM 波形 (AR=8) .....	231
图 9-28 边沿对齐 PWM 波形 (AR=8).....	232
图 9-29 单脉冲模式示例 .....	233
图 9-30 清除 TIMx 的 OCxREF.....	234
图 9-31 带死区插入的互补输出 .....	235
图 9-32 响应刹车的输出行为 .....	237
图 9-33 滑动滤波 .....	238
图 9-34 复位模式下的控制电路 .....	239
图 9-35 触发器模式下的控制电路 .....	239
图 9-36 门控模式下的控制电路 .....	240
图 9-37 外部时钟模式 2+触发模式下的控制电路 .....	241
图 9-38 产生六步 PWM, 使用 COM 的例子 (OSSR=1) .....	242
图 9-39 编码器模式下的计数器操作实例 .....	243
图 9-40 IC1FP1 反相的编码器接口模式实例 .....	244
图 9-41 霍尔传感器接口的实例 .....	245
图 10-1 TIMx (x=2,3,4 and 5) 框图 .....	291
图 10-2 当预分频的参数从 1 到 4, 计数器的时序图 .....	292
图 10-3 当内部时钟分频因子 = 2/N 时, 向上计数的时序图 .....	293

图 10-4 当 ARPEN=0/1 产生更新事件时，向上计数的时序图 .....	294
图 10-5 内部时钟分频因子 = 2/N 时，向下计数时序图 .....	295
图 10-6 内部时钟分频因子 = 2/N，中央对齐时序图 .....	296
图 10-7 包合计数器上溢和下溢的中央对齐时序图(ARPEN=1) .....	297
图 10-8 正常模式下的控制电路，内部时钟除以 1 .....	298
图 10-9 TI2 外部时钟连接示例 .....	299
图 10-10 外部时钟模式 1 的控制电路 .....	300
图 10-11 外部触发输入框图 .....	300
图 10-12 外部时钟模式 2 的控制电路 .....	301
图 10-13 捕获/比较通道（例如：通道 1 输入级） .....	302
图 10-14 捕获/比较通道 1 主电路 .....	303
图 10-15 通道 x 的输出部分（x=1/2/3/4，以通道 4 为例子） .....	304
图 10-16 滑动滤波 .....	305
图 10-17 PWM 输入模式时序 .....	306
图 10-18 输出比较模式，开启 OC1.....	307
图 10-19 中央对齐的 PWM 波形 (AR=8) .....	309
图 10-20 边沿对齐 PWM 波形 (AR=8) .....	310
图 10-21 单脉冲模式示例 .....	311
图 10-22 清除 TIMx 的 OCxREF.....	312
图 10-23 复位模式下的控制电路 .....	313
图 10-24 触发器模式下的控制电路 .....	314
图 10-25 门控模式下的控制电路 .....	315
图 10-26 外部时钟模式 2+触发模式下的控制电路 .....	316
图 10-27 主/从定时器的例子 .....	317
图 10-28 定时器 2 由定时器 1 的 OC1REF 门控.....	318
图 10-29 定时器 2 由定时器 1 的使能门控 .....	319
图 10-30 使用定时器 1 的更新触发定时器 2 .....	319
图 10-31 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2.....	320
图 10-32 编码器模式下的计数器操作实例 .....	322
图 10-33 IC1FP1 反相的编码器接口模式实例 .....	322
图 11-1 TIMx 的框图 (x = 6) .....	355
图 11-2 预分频器分频从 1 到 4 的计数器时序图 .....	356

图 11-3 向上计数时序图, 内部时钟分频因子 = 2/N .....	357
图 11-4 ARPEN=0/1 时向上计数、更新事件的时序图 .....	358
图 11-5 正常模式下的控制电路, 内部时钟分频系数为 1 .....	359
图 12-1 LPTIM 主框图 .....	365
图 12-2 毛刺滤波器时序图 .....	367
图 12-3 LPTIM 输出波形, 连续计数模式配置 .....	368
图 12-4 LPTIM 输出波形, 单触发计数模式配置 .....	369
图 12-5 LPTIM 输出波形, 一次模式 .....	370
图 12-6 波形发生器 .....	371
图 12-7 编码器模式计数序列 .....	374
图 12-8 非正交编码器正常工作 Input1、Input2 波形 .....	375
图 12-9 非正交编码器非正常工作 Input1、Input2 波形 .....	375
图 13-1 独立看门功能框图 .....	387
图 14-1 窗口看门狗功能框图 .....	393
图 14-2 WWDG 的刷新窗口和中断时序 .....	394
图 15-1 ADC 框图 .....	399
图 15-2 ADC 时钟 .....	400
图 15-3 ADC 通道引脚连接 .....	402
图 15-4 时序图 .....	404
图 15-5 注入转换延时 .....	406
图 15-6 校准时序图 .....	407
图 15-7 温度传感器和 V <sub>REFINT</sub> 通道框图 .....	411
图 16-1 比较器系统连接图 .....	427
图 17-1 I <sup>2</sup> C 功能框图 .....	447
图 17-2 I <sup>2</sup> C 总线协议 .....	447
图 17-3 从发送器传送序列 .....	449
图 17-4 从机接收器传送序列 .....	451
图 17-5 主发送器传送序列 .....	453
图 17-6 主接收器传送序列图 .....	455
图 18-1 USART 框图 .....	475
图 18-2 字长=8 设置 .....	476
图 18-3 字长=9 设置 .....	477

图 18-4 停止位配置.....	478
图 18-5 发送时 TXC/TXDE 的变化情况 .....	479
图 18-6 起始位检测.....	480
图 18-7 DMA 发送.....	486
图 18-8 DMA 接收.....	487
图 18-9 两个 USART 间的硬件流控制 .....	488
图 18-10 RTS 流控制.....	488
图 18-11 CTS 流控制.....	489
图 18-12 静默模式下的空闲总线检测 .....	490
图 18-13 静默模式下的地址标识检测 .....	491
图 18-14 USART 同步传输示例 .....	492
图 18-15 USART 数据时钟时序示例（WL=0） .....	492
图 18-16 USART 数据时钟时序示例（WL=1） .....	493
图 18-17 RX 数据采样/保持时间 .....	493
图 18-18 IrDA SIR ENDEC–框图 .....	495
图 18-19 IrDA 数据调制 (3/16)-正常模式.....	495
图 18-20 LIN 模式下的断开检测（11 位断开帧长度–设置了 LINBDL 位） .....	497
图 18-21 LIN 模式下的断开检测与帧错误的检测.....	498
图 18-22 ISO7816-3 异步协议 .....	499
图 18-23 使用 1.5 停止位检测奇偶检验错误 .....	500
图 19-1 SPI 框图 .....	513
图 19-2 硬件/软件的从选择管理 .....	514
图 19-3 单主和单从应用 .....	514
图 19-4 数据时钟时序图 .....	516
图 19-5 主机全双工模式下连续传输时，SPI_STS.TE/RNE/BUSY 的变化示意图 .....	517
图 19-6 主机单向只发送模式下连续传输时，SPI_STS.TE/BUSY 变化示意图 .....	518
图 19-7 只接收模式（BIDIRMODE=0 且 RONLY=1）下连续传输时，RNE 变化示意图 .....	519
图 19-8 从机全双工模式下连续传输时，SPI_STS.TE/RNE/BUSY 的变化示意图 .....	520
图 19-9 从机单向只发送模式下连续传输时，SPI_STS.TE/BUSY 变化示意图 .....	520
图 19-10 BIDIRMODE=0，RONLY=0 非连续传输发送时，SPI_STS.TE/BUSY 变化示意图 .....	522
图 19-11 使用 DMA 发送.....	524
图 19-12 使用 DMA 接收.....	524

图 19-13 I <sup>2</sup> S 框图 .....	527
图 19-14 I <sup>2</sup> S 飞利浦协议波形（16/32 位全精度，CLKPOL=0） .....	529
图 19-15 I <sup>2</sup> S 飞利浦协议标准波形（24 位帧，CLKPOL=0） .....	529
图 19-16 I <sup>2</sup> S 飞利浦协议标准波形（16 位扩展至 32 位包帧，CLKPOL=0） .....	530
图 19-17 MSB 对齐 16 位或 32 位全精度，CLKPOL=0 .....	530
图 19-18 MSB 对齐 24 位数据，CLKPOL=0 .....	531
图 19-19 MSB 对齐 16 位数据扩展到 32 位包帧，CLKPOL=0 .....	531
图 19-20 LSB 对齐 16 位或 32 位全精度，CLKPOL=0 .....	532
图 19-21 LSB 对齐 24 位数据，CLKPOL=0 .....	532
图 19-22 LSB 对齐 16 位数据扩展到 32 位包帧，CLKPOL=0 .....	533
图 19-23 PCM 标准波形（16 位） .....	533
图 19-24 PCM 标准波形（16 位扩展到 32 位包帧） .....	534
图 19-25 I <sup>2</sup> S 时钟发生器结构 .....	534
图 19-26 音频采样频率定义 .....	535
图 20-1 RTC 功能框图 .....	550
图 22-1 CAN 网络拓扑 .....	581
图 22-2 CAN 工作模式 .....	583
图 22-3 单 CAN 框图 .....	584
图 22-4 回环模式 .....	585
图 22-5 静默模式 .....	586
图 22-6 回环静默模式 .....	586
图 22-7 发送邮箱状态 .....	588
图 22-8 接收邮箱状态 .....	589
图 22-9 过滤器位宽设置-寄存器组织 .....	591
图 22-10 过滤机制示例 .....	593
图 22-11 位时序 .....	595
图 22-12 各类帧格式 .....	596
图 22-13 事件标志和中断产生 .....	597
图 22-14 CAN 错误状态框图 .....	598
图 23-1 N32G430 级和 Cortex®-M4 级调试框图 .....	622

# 1 文中的缩写

## 1.1 寄存器位操作缩写列表

以下缩写用于寄存器描述：

read/write(rw)	支持软件读写该位
read-only(r)	支持软件只读该位
write-only(w)	支持软件只写该位，软件读返回复位值
read/clear(rc_w1)	支持软件读该位，写1清除该位，写0无效
read/clear(rc_w0)	支持软件读该位，写0清除该位，写1无效
read/clear by read(rc_r)	支持软件读该位，读操作将清除该位，写0无效
read/set(rs)	支持软件读或者设置该位，写0无效
read-only write trigger(rt_w)	支持软件读该位，写0或1触发一个事件但不改变该位数值
toggle(t)	支持软件写1翻转该位，写0无效
Reserved(Res.)	保留位，必须保持默认值不变

## 1.2 可用外设

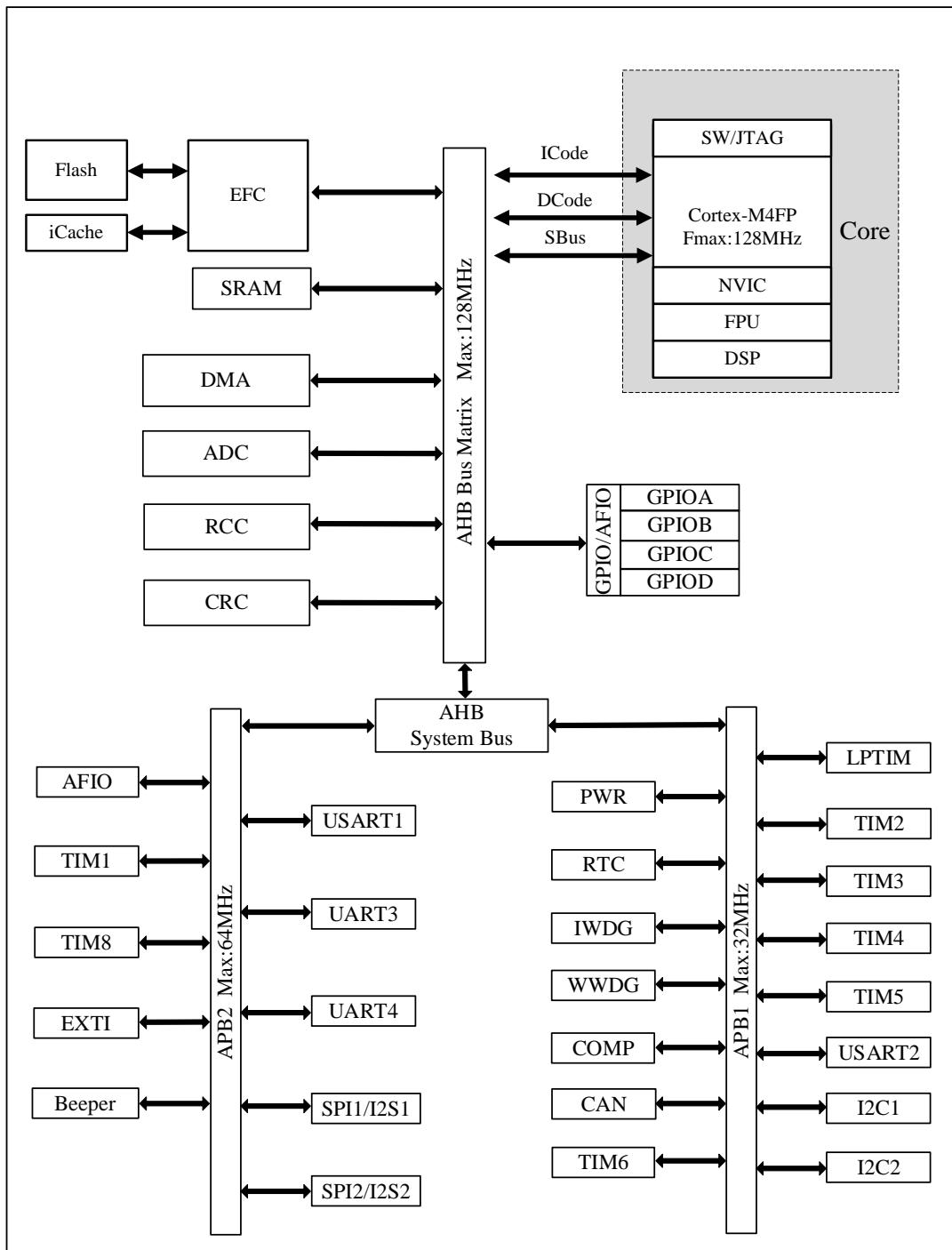
有关 N32G430 微控制器系列全部型号，某外设存在与否及其数目，请查阅相应型号的数据手册。

## 2 存储器和总线架构

## 2.1 系统架构

### 2.1.1 总线架构

图 2-1 总线架构图



- ICode 总线：将 Cortex™-M4FP 内核的 ICode 总线与闪存指令接口相连接。指令预取在此总线上完成。

- DCode 总线将 Cortex™-M4FP 内核的 DCode 总线与闪存存储器的数据接口相连接（常量加载和调试访问）。
- SBus 总线连接 Cortex™-M4FP 内核的 SBus 总线（外设总线）到总线矩阵，总线矩阵协调着内核和 DMA 间的访问。
- CRC 没有 DMA 硬件握手，但设计了矩阵互联，支持软件触发的方式进行 DMA 传输。
- 系统包含 2 个 AHB2APB 桥，即 AHB2APB1 和 AHB2APB2。其中 APB1 包含 15 个低速 APB 外设，PCLK 的最高速度为 32MHz；APB2 包含 10 个高速 APB 外设，PCLK 最高速度等于 64MHz。

## 2.1.2 总线地址映射

总线地址映射包括所有 AHB 和 APB 外设：AHB 外设、APB1 外设、APB2 外设、Flash、SRAM、SystemMemory 等。SRAM 的地址空间位于 SRAM 的 bit-band 区，可以通过 bit-band 别名进行原子访问，以完成对 bit-band 区域的目标位的读-改-写操作。所有 APB 和 AHB 外设的地址空间均位于外设的 bit-band 区，可以通过 bit-band 别名进行原子访问，以完成对 bit-band 区域的目标位的读-改-写操作。具体映射如下

图 2-2 总线地址映射图

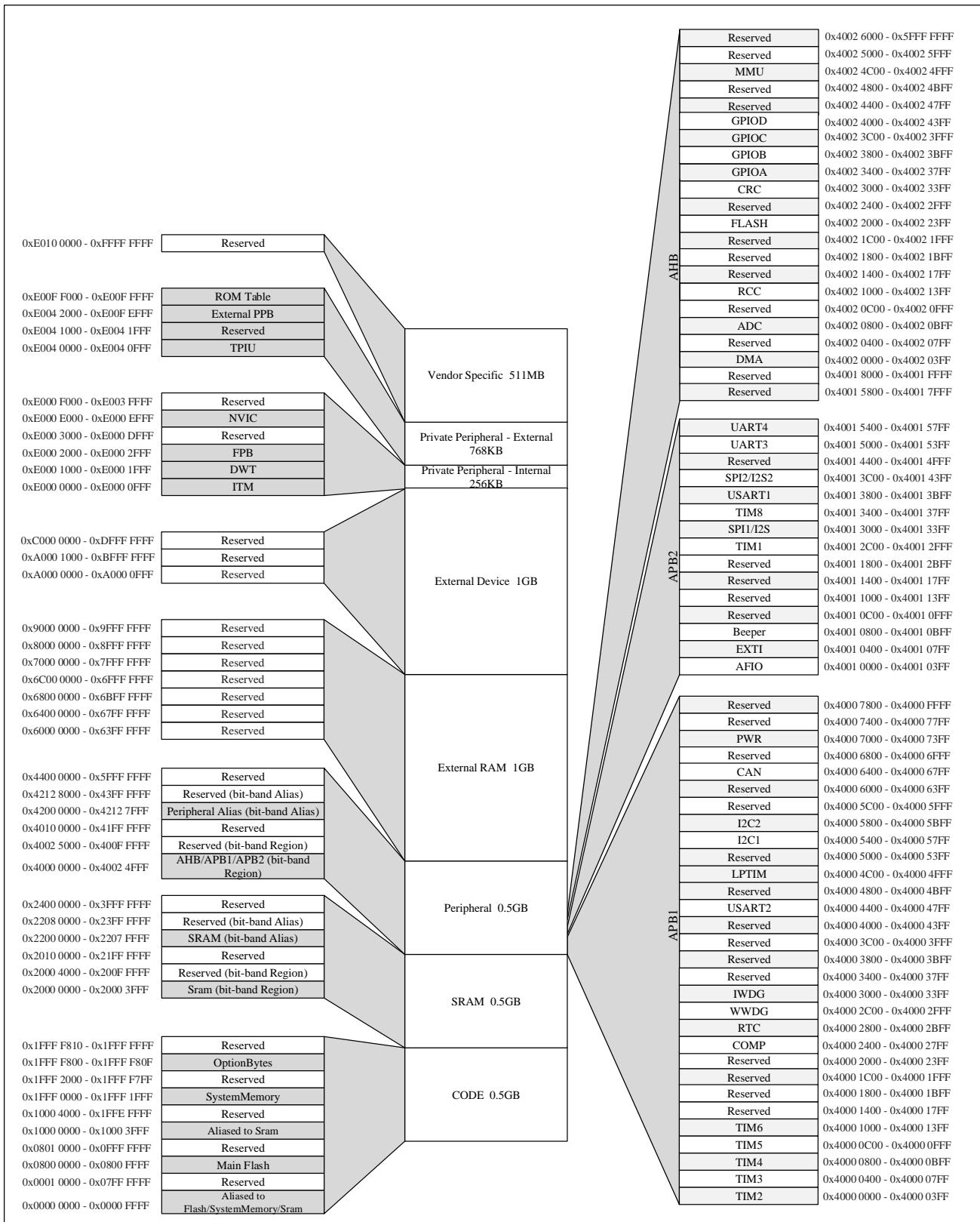


表 2-1 外设寄存器地址列表

地址范围	外设	总线
0x4002_6000 – 0x5FFF_FFFF	Reserved	AHB
0x4002_5000 – 0x4002_5FFF	Reserved	
0x4002_4C00 – 0x4002_4FFF	MMU	
0x4002_4800 – 0x4002_4BFF	Reserved	
0x4002_4400 – 0x4002_47FF	Reserved	
0x4002_4000 – 0x4002_43FF	GPIOD	
0x4002_3C00 – 0x4002_3FFF	GPIOC	
0x4002_3800 – 0x4002_3BFF	GPIOB	
0x4002_3400 – 0x4002_37FF	GPIOA	
0x4002_3000 – 0x4002_33FF	CRC	
0x4002_2400 – 0x4002_2FFF	Reserved	
0x4002_2000 – 0x4002_23FF	FLASH	
0x4002_1C00 – 0x4002_1FFF	Reserved	
0x4002_1800 – 0x4002_1BFF	Reserved	
0x4002_1400 – 0x4002_1FFF	Reserved	
0x4002_1000 – 0x4002_13FF	RCC	
0x4002_0C00 – 0x4002_0FFF	Reserved	
0x4002_0800 – 0x4002_0BFF	ADC	
0x4002_0400 – 0x4002_07FF	Reserved	
0x4002_0000 – 0x4002_03FF	DMA	
0x4001_8000 – 0x4001_FFFF	Reserved	
0x4001_5800 – 0x4001_7FFF	Reserved	
0x4001_5400 – 0x4001_57FF	UART4	APB2
0x4001_5000 – 0x4001_53FF	UART3	
0x4001_4400 – 0x4001_4FFF	Reserved	
0x4001_3C00 – 0x4001_43FF	SPI2/I2S2	
0x4001_3800 – 0x4001_3BFF	USART1	
0x4001_3400 – 0x4001_37FF	TIM8	
0x4001_3000 – 0x4001_33FF	SPI1/I2S	
0x4001_2C00 – 0x4001_2FFF	TIM1	
0x4001_1800 – 0x4001_2BFF	Reserved	
0x4001_1400 – 0x4001_17FF	Reserved	
0x4001_1000 – 0x4001_13FF	Reserved	
0x4001_0C00 – 0x4001_0FFF	Reserved	
0x4001_0800 – 0x4001_0BFF	Beeper	
0x4001_0400 – 0x4001_07FF	EXTI	
0x4001_0000 – 0x4001_03FF	AFIO	

地址范围	外设	总线
0x4000_7800 – 0x4000_FFFF	Reserved	APB1
0x4000_7400 – 0x4000_77FF	Reserved	
0x4000_7000 – 0x4000_73FF	PWR	
0x4000_6800 – 0x4000_6FFF	Reserved	
0x4000_6400 – 0x4000_67FF	CAN	
0x4000_6000 – 0x4000_63FF	Reserved	
0x4000_5C00 – 0x4000_5FFF	Reserved	
0x4000_5800 – 0x4000_5BFF	I2C2	
0x4000_5400 – 0x4000_57FF	I2C1	
0x4000_5000 – 0x4000_53FF	Reserved	
0x4000_4C00 – 0x4000_4FFF	LPTIM	
0x4000_4800 – 0x4000_4BFF	Reserved	
0x4000_4400 – 0x4000_47FF	USART2	
0x4000_4000 – 0x4000_43FF	Reserved	
0x4000_3C00 – 0x4000_3FFF	Reserved	
0x4000_3800 – 0x4000_3BFF	Reserved	
0x4000_3400 – 0x4000_37FF	Reserved	
0x4000_3000 – 0x4000_33FF	IWDG	
0x4000_2C00 – 0x4000_2FFF	WWDG	
0x4000_2800 – 0x4000_2BFF	RTC	
0x4000_2400 – 0x4000_27FF	COMP	
0x4000_2000 – 0x4000_23FF	Reserved	
0x4000_1C00 – 0x4000_1FFF	Reserved	
0x4000_1800 – 0x4000_1BFF	Reserved	
0x4000_1400 – 0x4000_17FF	Reserved	
0x4000_1000 – 0x4000_13FF	TIM6	
0x4000_0C00 – 0x4000_0FFF	TIM5	
0x4000_0800 – 0x4000_0BFF	TIM4	
0x4000_0400 – 0x4000_07FF	TIM3	
0x4000_0000 – 0x4000_03FF	TIM2	

### 2.1.2.1 Bit banding

Cortex™-M4FP 存储器映像包括两个位段（bit-band）区。这两个位段区将别名存储器区中的每个字映射到位段存储器区的一个位，在别名存储区写入一个字具有对位段区目标位执行读-改-写操作的相同效果。

外设寄存器和 SRAM 都被映射到一个位段区里，这允许执行单一的位段区写和读操作。

下面的映射公式给出了别名区中的每个字节是如何对应位段区的相应位的：

$$\text{bitband\_byte\_addr} = \text{bitband\_base} + (\text{byte\_offset} \times 32) + (\text{bit\_number} \times 4)$$

其中：

bitband\_byte\_addr 是别名存储器区中字节的地址，它映射到某个目标位；

bitband\_base 是别名区的起始地址；

byte\_offset 是包含目标位的字节在位段里的序号；

bit\_number 是目标位所在位置（0-7）。

举个例子：

下面的例子说明如何映射别名区中 SRAM 地址为 0x20000400 的字节中的位 4：

$$0x22008010 = 0x22000000 + (0x400 \times 32) + (4 \times 4).$$

对 0x22008010 地址的写操作与对 SRAM 中地址 0x20000400 字节的位 4 执行读-改-写操作有着相同的效果。

读 0x22008010 地址返回 SRAM 中地址 0x20000400 字节的位 4 的值（0x01 或 0x00）。请参考《Cortex™-M4 技术参考手册》以了解更多有关位段的信息。

## 2.1.3 启动管理

### 2.1.3.1 启动地址

在系统启动时，可以通过 BOOT0 (PD0) 引脚和选项字节 boot 配置 (USER2) 来选择在复位后的启动模式。在系统复位后或从待机模式退出时，BOOT 引脚的值将被重新锁存，选项字节 boot 配置 (USER2) 会重新被读取。经过启动延迟之后，CPU 从地址 0x0000\_0000 获取堆栈顶的地址，并从地址 0x0000\_0004 指示的复位向量地址开始执行代码。由于 Cortex-M4FP 始终通过 ICode 总线从地址 0x0000\_0000 和 0x0000\_0004 获取堆栈顶指针和复位向量，所以启动仅适合于从 CODE 代码区开始，设计上需要对启动空间进行地址重映射。有三种启动模式可选：

- 从主闪存存储器 (Main Flash) 启动：
  - ◆ 主闪存存储器被映射到启动空间 (0x0000\_0000)；
  - ◆ 主闪存存储器可在两个地址区域访问，0x0000\_0000 或 0x0800\_0000 (ICode/DCode/DMA)；
- 从系统存储器 (System Memory) 启动：
  - ◆ 系统存储器被映射到启动空间 (0x0000\_0000)；
  - ◆ 系统存储器可在两个地址区域访问，0x0000\_0000 或 0x1FFF\_0000 (ICode/DCode/DMA)；
- 从内置 SRAM 启动：
  - ◆ 内置 SRAM 被映射到启动空间 (0x0000\_0000)；
  - ◆ 内置 SRAM 可在两个地址区域访问，0x0000\_0000 或 0x2000\_0000 (ICode/DCode/SBus/DMA)；

### 2.1.3.2 启动配置

另外，SRAM 还可以通过虚拟地址段 0x1000\_0000 进行存取访问，这使得 CPU 从 Main Flash 或 System Memory 启动后，可跳转到 SRAM 通过 ICode/DCode 运行程序（注意不是从 SRAM 启动程序，不属于启动模式）。除了 BOOT 引脚配置启动程序外，还有两种方式可以在 SRAM 运行程序：

- 直接跳转到 SRAM 的物理地址段 0x2000\_0000 运行程序，此时将通过 SBus 运行程序；

- 跳转到 SRAM 的虚拟地址段 0x1000\_0000，内部重映射到物理地址段 0x2000\_0000 运行程序，此时将通过 ICode/DCode 高效运行程序。

表 2-2 启动模式列表

启动模式引脚选择				启动模式	对应启动模式下，访问内存空间的起始地址		
nBOOT1	nBOOT0	BOOT0 pin	nSWBOOT0		Main Flash	System Memory	SRAM
X	X	0	1	Main Flash start	0x0000_0000	0x1FFF_0000	0x2000 0000
X	1	X	0		0x0800_0000		0x1000 0000
1	X	1	1	System Memory Start	0x08000000	0x0000_0000	0x2000 0000
1	0	X	0			0x1FFF_0000	0x1000 0000
0	X	1	1	SRAM start	0x08000000	0x1FFF_0000	0x0000_0000
0	0	X	0				0x2000 0000 0x1000 0000

注：BOOT0 (PD0) 和 GPIO 复用，上电时默认使用输入下拉。

### 2.1.3.3 内嵌启动程序

内嵌的自举程序存放在系统存储器 System Memory 内，用于通过 USART1 对闪存存储器进行重新编程。而 USART1 接口除了可以依靠外部时钟 (HSE) 外，还可以依靠内部 8MHz 振荡器 (HSI) 运行。进一步的细节请查询自举程序手册。

## 2.2 存储系统 (Memory system)

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

### 2.2.1 FLASH 规格

Flash 由主存储区、信息区组成，以下分别进行说明：

- 主存储区最大为 64KB，也称作主闪存存储器，包含 32 个 Page，用于用户程序的存放和运行，以及数据存储。
- 信息区为 12KB，包含 6 个 Page，由系统存储区 (8KB)、系统配置区 (2KB)、选项字节区 (2KB) 组成：
  - ◆ 系统存储区为 8KB，包含 4 个 Page，也称作 System Memory，用于引导程序 (BOOT) 的存放和运行。
  - ◆ 系统配置区为 2KB，包含 1 个 Page。
  - ◆ 选项字节区为 2KB，包含 1 个 Page，也称作 OptionByte，有效空间为 16B，BOOT 程序、用户程序均可以读写擦。

### 2.2.1.1 存储地址

主存储区、信息区都分配了总线地址空间。

表 2-3 存储总线地址列表

存储区	页编号	地址范围	大小
主存储区	Page 0	0x0800_0000 – 0x0800_07FF	2KB
	Page 1	0x0800_0800 – 0x0800_0FFF	2KB
	Page 2	0x0800_1000 – 0x0800_17FF	2KB
	:	:	:
	Page 31	0x0800_F800 – 0x0800_FFFF	2KB
信息区	系统存储区	0x1FFF_0000 – 0x1FFF_1FFF	8KB
	系统配置区	0x1FFF_F000 – 0x1FFF_F7FF	2KB
	选项字节区	0x1FFF_F800 – 0x1FFF_F80F	16B
Flash 接口寄存器	FLASH_AC	0x4002_2000 – 0x4002_2003	4B
	FLASH_KEY	0x4002_2004 – 0x4002_2007	4B
	FLASH_OPTKEY	0x4002_2008 – 0x4002_200B	4B
	FLASH_STS	0x4002_200C – 0x4002_200F	4B
	FLASH_CTRL	0x4002_2010 – 0x4002_2013	4B
	FLASH_ADD	0x4002_2014 – 0x4002_2017	4B
	FLASH_OB2	0x4002_2018 – 0x4002_201B	4B
	FLASH_OB	0x4002_201C – 0x4002_201F	4B
	FLASH_WRP	0x4002_2020 – 0x4002_2023	4B
	Reserved	0x4002_2024 – 0x4002_202F	12B
	FLASH_CAHR	0x4002_2030 – 0x4002_2033	4B

闪存存储器被组织成 32 位宽的存储器单元，可以存放代码和数据常数。

信息区分为三个部分：

- 系统存储区是用于存放在系统存储器自举模式下的启动程序，启动程序使用 USART1 接口实现对闪存存储器的编程。
- 系统配置区，包含芯片基本信息。
- 选项字节区，对主存储器和信息块的写入由内嵌的闪存编程/擦除控制器管理。

闪存存储器有两种保护方式防止非法的访问（读、写、擦除）：

- 页写入保护（WRP）
- 读出保护（RDP）

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

进行闪存编程操作时（写或擦除），必须打开内部的 RC 振荡器（HSI）。

注： 在低功耗模式下，所有闪存存储器的操作都被中止。

### 2.2.1.2 读写操作

Flash 写操作仅支持 32 位操作，写操作之前先擦除 Flash，擦除最小块大小是一个页 2KB。写操作分为编程和擦除阶段。

读 Flash 时，读的等待周期数可以通过寄存器配置。使用时，需要结合 AHB 接口时钟频率进行计算，等待时间必须不小于 25ns。比如：当 HCLK<=32MHz 时，等待周期数最小为 0；当 32MHz<HCLK<=64MHz 时，等待周期数最小为 1；当 64MHz<HCLK<=96MHz，等待周期数最小为 2；当 96MHz<HCLK，等待周期数最小为 3。

注意：无论等待周期数是否为零，启用预取缓冲功能可以提高整体读代码的效率。

### 2.2.1.3 Flash 解锁操作

复位后，Flash 模块是被保护的，不能写入 FLASH\_CTRL 寄存器，以防因电气干扰等原因产生对 Flash 的意外操作。通过写入特定的键值序列到 FLASH\_KEY 寄存器，可以开启对 FLASH\_CTRL 寄存器的操作权限，这个特定的序列是：第一次在 Flash 密钥寄存器（FLASH\_KEY）中写入 KEY1 = 0x45670123，第二次则在 Flash 密钥寄存器（FLASH\_KEY）中写入 KEY2 = 0xCDEF89AB。

如果顺序出现错误或键值出现错误，将返回总线错误并锁定 FLASH\_CTRL 寄存器，直到下一次复位，软件可以通过查看 FLASH\_CTRL.LOCK 位来确认 Flash 是否已解锁。若需要进行正常的锁定设置，可以通过软件将 FLASH\_CTRL.LOCK 位置 1 来实现，此后可以通过在 FLASH\_KEY 中写入正确的键值系列来对 Flash 解锁。

### 2.2.1.4 擦除和编程

#### 2.2.1.4.1 主存储区擦除

主存储区可以按页擦除或者整片擦除

##### 页擦除

页擦除流程：

- 通过检查 FLASH\_STS.BUSY 位来确保没有正在进行闪存操作；
- 设置 FLASH\_CTRL.PER 为'1'；
- 将要擦除的页起始地址写入 FLASH\_ADD 寄存器；
- 设置 FLASH\_CTRL.START 为'1'；
- 等待 FLASH\_STS.BUSY 变为'0'；
- 读出被擦除页的内容检查是否被擦除。

##### 片擦除

片擦除流程：

- 通过检查 FLASH\_STS.BUSY 位来确保没有正在进行闪存操作；
- 设置 FLASH\_CTRL.MER 为'1'；
- 设置 FLASH\_CTRL.START 为'1'；
- 等待 FLASH\_STS.BUSY 位变为'0'；

- 读出所有被擦除页的内容检查是否被擦除。

#### 2.2.1.4.2 主存储区编程

对主存储区编程每次可以写入 32 位。当 FLASH\_CTRL.PG 为'1'时，在一个闪存地址写入一个字将启动一次编程；写入任何半字的数据，都会产生总线错误。在编程过程中(FLASH\_STS.BUSY 为'1')，任何读写闪存的操作都会使 CPU 暂停，直到此次闪存编程结束。

主存储区编程流程：

- 通过检查 FLASH\_STS.BUSY 位来确保没有正在进行闪存操作；
- 设置 FLASH\_CTRL.PG 为'1'；
- 在指定的地址写入要编程的字；
- 等待 FLASH\_STS.BUSY 变为'0'；
- 读出写入地址的数据检查是否正确。

注意：当 FLASH\_STS.BUSY 为'1'时，不能对任何 Flash 寄存器执行写操作。

#### 2.2.1.4.3 选项字节区擦除和编程

对选项字节区的编程与主存储区不同。选项字节的数目只有 8 个字节(2 个字节作为写保护，2 个字节作为读保护，2 个字节为配置选项，2 个字节存储用户数据)。对 Flash 解锁后，必须分别写入 KEY1 和 KEY2(见 2.2.1.3)到 FLASH\_OPTKEY 寄存器，再设置 FLASH\_CTRL.OPTWE 为'1'；此时可以对选项字节区进行编程：设置 FLASH\_CTRL.OPTPG 为'1'后写入字到指定的地址。

对选项字节区字编程时，使用半字中的低字节并自动地计算出高字节(高字节为低字节的补码)，并开始编程操作，这将保证选项字节和它的补码始终是正确的。

选项字节区擦除过程：

- 通过检查 FLASH\_STS.BUSY 位来确保没有正在进行闪存操作；
- 解锁 FLASH\_CTRL.OPTWE；
- 设置 FLASH\_CTRL.OPTER 为'1'；
- 设置 FLASH\_CTRL.START 为'1'；
- 等待 FLASH\_STS.BUSY 变为'0'；
- 读出被擦除选项字节的内容检查是否被擦除。

选项字节区编程流程：

- 通过检查 FLASH\_STS.BUSY 位来确保没有正在进行闪存操作；
- 解锁 FLASH\_CTRL.OPTWE；
- 设置 FLASH\_CTRL.OPTPG 为'1'；
- 在指定的地址写入要编程的字；
- 等待 FLASH\_STS.BUSY 变为'0'；
- 读出写入地址的数据检查是否正确。

## 2.2.1.5 指令预取

Flash 模块的指令预取功能，支持 16B 的预取 Buffer。通过指令预取操作，可提高 CPU 的指令执行效率。指令预取功能可以通过寄存器配置为使能或除能，默认使能。

## 2.2.1.6 选项字节

选项字节块主要用于配置读写保护、软件/硬件看门狗以及系统处于 standby/stop0/stop2 模式下的复位选项，并分配了总线地址空间，可以进行读写访问。它们由有 8 个选项字节组成：2 个字节作为写保护，2 个字节作为读保护，2 个字节作为配置选项，2 个字节由用户定义，这 8 个字节需要通过总线写入。选项字节块同时还包含与这 8 个选项字节相对应的补码，这些补码需要在总线写入选项字节时，由硬件自动计算出来，一起写入 Flash，并用于选项字节读取时的验证。

默认状态下，选项字节块始终是可以读且被写保护。要想对选项字节块进行写操作（编程/擦除），首先要解锁 Flash，然后解锁选项字节：在 FLASH\_OPTKEY 中写入正确的键值序列（KEY1 = 0x45670123，KEY2 = 0xCDEF89AB），随后对选项字节块的写操作将被允许。如果顺序出现错误或键值出现错误，将返回总线错误并锁定选项字节，直到下一次复位。若需要正常进行锁定设置，可以通过软件将 FLASH\_CTRL.OPTWE 位写 0 来实现，此后可以通过在 FLASH\_OPTKEY 中写入正确的键值系列来对选项字节解锁。

每次系统复位后，从 Flash 的选项字节块中读出选项字节数据，并保存在具有只读属性的选项字节寄存器（FLASH\_OB/FLASH\_WRP）中；同时一起读出来的选项字节补码数据，将用于验证选项字节数据是否正确，如果不匹配，将产生一个选项字节错误标志（FLASH\_OB.OBERR）。当发生选项字节错误时，对应的选项字节被强置为 0xFF。当选项字节和它的补码均为 0xFF 时（擦除后的状态），则略过上述验证步骤，无需进行验证。

表 2-4 选项字节列表

Address	[31:24] 补码	[23:16] 选项字节	[15:8] 补码	[7:0] 选项字节
0xFFFF_F800	nUSER	USER	nRDP1	RDP1
0xFFFF_F804	nData1	Data1	nData0	Data0
0xFFFF_F808	nWRP1	WRP1	nWRP0	WRP0
0xFFFF_F80C	nUSER2	USER2	nRDP2	RDP2

- 读保护 L1 等级：RDP1
  - ◆ 保护存储在闪存中的代码；
  - ◆ 当写入正确的是数值时，将禁止读出闪存存储器；
  - ◆ RDP1 是否开启的结果，可通过 FLASH\_OB[1]查询；
- 用户配置选项：USER
  - ◆ USER[7]: 保留
  - ◆ USER[6]: IWDGSLEEPFRZ，可通过 FLASH\_OB[8]查询；
    - 0: iwdg 在 sleep 模式下冻结
    - 1: iwdg 在 sleep 模式下非冻结
  - ◆ USER[5]: IWDGSTDBYFRZ，可通过 FLASH\_OB[7]查询；

0: iwdg 在 standby 模式下冻结

1: iwdg 在 standby 模式下非冻结

- ◆ USER[4]: IWDGSTOP2FRZ, 可通过 FLASH\_OB[6]查询;
  - 0: iwdg 在 stop2 模式下冻结
  - 1: iwdg 在 stop2 模式下非冻结
- ◆ USER[3]: IWDGSTOP0FRZ, 可通过 FLASH\_OB[5]查询;
  - 0: iwdg 在 stop0 模式下冻结
  - 1: iwdg 在 stop0 模式下非冻结
- ◆ USER[2]: nRST\_STDBY 配置选项, 可通过 FLASH\_OB[4]查询;
  - 0: 当进入 standby 模式时产生复位
  - 1: 进入 standby 模式时不产生复位
- ◆ USER[1]: nRST\_STOP 配置选项, 可通过 FLASH\_OB[3]查询;
  - 0: 当进入 stop0/stop2 模式时产生复位
  - 1: 进入 stop0/stop2 模式时不产生复位
- ◆ USER[0]: WDG\_SW 配置选项, 可通过 FLASH\_OB[2]查询;
  - 0: 硬件看门狗
  - 1: 软件看门狗

- 2 字节用户数据: Datax

- ◆ Data1 (FLASH\_OB[25:18])
- ◆ Data0 (FLASH\_OB [17:10])

- 写保护选项字节: WRP0 ~ 1, 可通过寄存器 FLASH\_WRP[15:0]查询

- ◆ WRP0: 第 0 ~ 15 页的写保护, bit[0]对应 Page0/1, ..... , bit[7]对应 Page14/15;
- ◆ WRP1: 第 16~31 页的写保护, bit[0]对应 Page16/17, ..... , bit[7]对应 Page30/31;

- 读保护 L2 等级: RDP2

- ◆ 在 L1 的基础上增加保护功能, 具体见 2.2.1.8 读保护的详细描述;
- ◆ RDP2 是否开启的结果, 可通过 FLASH\_OB[31]查询;

- 用户配置选项 2: **USER2**

- ◆ USER2[7:3]: 保留
- ◆ USER2[2] : nSWBOOT0, 可通过 FLASH\_OB2[26]查询, 默认值为 1
- ◆ USER2[1]: nBOOT1, 可通过 FLASH\_OB2[23]查询, 默认值为 1
- ◆ USER2[0]: nBOOT0, 可通过 FLASH\_OB2[27]查询, 默认值为 1

## 2.2.1.7 写保护

可以对 Flash 主存储区（最大 64KB）的所有 Page 配置写保护，以防在程序跑飞或电气干扰等原因导致的意外写操作，写保护的基本单位是：对于 Page0~31，每 2 页为一个基本保护单元。写保护可以通过设置选项字节块中的 WRP0~1 来进行配置；每次进行配置后，需要进行一次系统复位，配置的值才能生效。如果对一个受保护的页面进行编程或擦除操作，FLASH\_STS 中将会返回一个保护错误标志。

系统信息区中的系统存储块（8KB），存放了 BOOT 程序，不可更改。

系统信息区中的系统配置块（2KB），存放了芯片基本信息，不可更改。

系统信息区中的选项字节块（2KB），存放了用户可配置选项字节信息，将 FLASH\_CTRL.OPTWE 写 0 使能选项字节块的写保护，之后通过在 FLASH\_OPTKEY 中写入正确的键值序列，来对选项字节解除写保护。

## 2.2.1.8 读保护

Flash 中的用户代码可以通过设置读保护来防止被非法读取。读保护通过配置选项字节块中的 RDP 字节进行设置，可以配置 3 种不同的读保护级别，如下列表

表 2-5 读保护配置列表

读保护等级	RDP1	nRDP1	nRDP2	RDP2
L1 level	0xFF	0xFF	RDP2!=0xCC    nRDP2!=0x33	
未保护	0xA5	0x5A	RDP2!=0xCC    nRDP2!=0x33	
L2 level	0xXX	0xXX	0x33	0xCC
L1 level	非以上三种配置			

- L0 等级：
  - ◆ 处于未保护状态，(RDP1 == 0xA5 & nRDP1 == 0x5A) && (RDP2!=0xCC | nRDP2!=0x33)
  - ◆ 主存储区和选项字节可以被任意读取
  - ◆ 主存储区和选项字节可以进行编程和擦除，可配置读写保护
- L1 等级：
  - ◆ ~ (((RDP1 == 0xA5 & nRDP1 == 0x5A) && (RDP2!=0xCC | nRDP2!=0x33)) | (RDP2 == 0xCC & nRDP2 == 0x33))
  - ◆ 只允许从用户代码中对主存储区的读操作，即以非调试方式从主闪存存储器启动程序的情况才允许对主存储区的读操作
  - ◆ 全部主存储区页可以通过在主闪存存储器中执行的代码进行编程（实现 IAP 或数据存储等功能）
  - ◆ 全部主存储区页不允许在调试模式下或从内部 SRAM 启动后执行写或擦除操作（整片擦除除外）
  - ◆ 所有通过 JTAG/SWD 向内置 SRAM 装载代码并执行代码的功能依然有效，亦可以通过 JTAG/SWD 从内置 SRAM 启动，这个功能可以用来解除读保护；
  - ◆ 当读保护的选项字节被改写为未保护的 L0 级别时，将会自动擦除全部主存储区，执行的过程如下：（擦除选项字节块不会导致自动的整片擦除操作，因为擦除的结果是 0xFF，相当于仍然处于 L1 级别的保护状态）
    - 在 FLASH\_OPTKEY 中写入正确的键值序列解锁选项字节区；

- 总线发起命令擦除整个选项字节区（Page 擦）；
  - 总线写入读保护选项字节 0xA5；
  - 内部自动擦除全部主存储区；
  - 内部自动写入 0xA5 到读保护选项字节；
  - 进行系统复位（如软件复位等），选项字节块（包括新的 RDP 值 0xA5）将被重新加载到系统中，读保护被解除；
- ◆ 以下对闪存的访问操作都将被禁止：
- 通过从内置 SRAM 启动（包括使用 DMA）访问主闪存存储器；
  - 通过 JTAG、SWV（串行线观察器）、SWD（串行线调试）和边界扫描方式访问主闪存存储器；
- L2 等级：除了 SRAM 启动被禁止、调试模式被禁止、选项字节写/页擦除被禁止、保护级别不可修改（不可逆）之外，其余特性同 L1 级别。L2 级别通过配置另一个选项字节 RDP2 来实现，不管 RDP1 为何值，只要满足（RDP2=0xCC & nRDP2=0x33）即为 L2 级别

表 2-6 存储区读写擦<sup>(1)</sup>权限控制表

保护 等级	启动模式	Main Flash			修改读保护等级
	执行用户 访问区域	JTAG/ SWD	Main Flash	System Memory	
L0	Flash主存储区	读写擦	读写擦	读写擦	读写擦
	Flash主存储区 片擦除 <sup>(2)</sup>	允许	允许	允许	允许
	选项字节区	读写擦	读写擦	读写擦	读写擦
	系统存储区	禁止	禁止	读写擦	禁止
	系统配置区	仅id可读	仅id可读	id可读, 分区信息可读	仅id可读
	SRAM (All)	读写	读写	读写	读写
L1 level	Flash主存储区	禁止	读写擦	读写擦	读写擦
	Flash主存储区 片擦除 <sup>(2)</sup>	允许	允许	允许	允许
	选项字节区	读写擦	读写擦	读写擦	读写擦
	系统存储区	禁止	禁止	读写擦	禁止
	系统配置区	禁止	仅id可读	id可读, 分区信息可读	仅id可读
	SRAM (All)	读写	读写	读写	读写
L2 level	Flash主存储区	JTAG/SWD 接口被禁止	读写擦	读写擦	读写擦
	Flash主存储区 片擦除 <sup>(2)</sup>		允许	允许	允许
	选项字节区		只读	只读	只读
	系统存储区		禁止	读写擦	禁止

	系统配置区		仅id可读	id可读, 分区信息可读	仅id可读	
	SRAM (All)		读写	读写	读写	
<hr/>						
保护 等级	启动模式	SRAM			修改读保护等级	
	执行用户 访问区域	JTAG/ SWD	Main Flash	System Memory	SRAM	
L0 level	Flash主存储区	读写擦	读写擦	读写擦	读写擦	允许改为 L1 或 L2
	Flash主存储区 片擦除 <sup>(2)</sup>	允许	允许	允许	允许	
	选项字节区	读写擦	读写擦	读写擦	读写擦	
	系统存储区	禁止	禁止	读写擦	禁止	
	系统配置区	仅id可读	仅id可读	id可读, 分区信息可读	仅id可读	
	SRAM (All)	读写	读写	读写	读写	
L1 level	Flash主存储区	禁止	读写擦	读写擦	禁止	允许改为 L0 或 L2。 改为 L0 时，主存储区 将被自动擦除
	Flash主存储区 片擦除 <sup>(2)</sup>	允许	允许	允许	允许	
	选项字节区	读写擦	读写擦	读写擦	读写擦	
	系统存储区	禁止	禁止	禁止	禁止	
	系统配置区	禁止	仅id可读	id可读, 分区信息可读	禁止	
	SRAM (All)	读写	读写	读写	读写	
L2 level	Flash主存储区	L2 保护等级，无法从 SRAM 启动				不允许修改， JTAG/SWD 接口被禁
	Flash主存储区					

	片擦除 <sup>(2)</sup>					止
	选项字节区					
	系统存储区					
	系统配置区					
	SRAM (All)					
保护 等级	启动模式	System Memory				修改读保护等级
	执行用户 访问区域	JTAG/ SWD	Main Flash	System Memory	SRAM	
L0 level	Flash主存储区	读写擦	读写擦	读写擦	读写擦	允许改为L1或L2
	Flash主存储区 片擦除 <sup>(2)</sup>	允许	允许	允许	允许	
	选项字节区	读写擦	读写擦	读写擦	读写擦	
	系统存储区	禁止	禁止	读写擦	禁止	
	系统配置区	仅id可读	仅id可读	id可读, 分区信息可读	仅id可读	
	SRAM (All)	读写	读写	读写	读写	
L1 level	Flash主存储区	禁止	读写擦	读写擦	读写擦	允许改为 L0 或 L2。 改为L0时，主存储区 将被自动擦除
	Flash主存储区 片擦除 <sup>(2)</sup>	禁止	允许	允许	允许	
	选项字节区	读写擦	读写擦	读写擦	读写擦	
	系统存储区	禁止	禁止	读写擦	禁止	
	系统配置区	禁止	仅id可读	id可读, 分区信息可读	仅id可读	

	SRAM (All)	读写	读写	读写	读写	
L2 level	Flash主存储区	JTAG/SWD 接口被禁止	读写擦	读写擦	读写擦	不允许修改
	Flash主存储区 片擦除 <sup>(2)</sup>		允许	允许	允许	
	选项字节区		只读	只读	只读	
	系统存储区		禁止	读写擦	禁止	
	系统配置区		仅id可读, 分区信息可读		仅id可读	
	SRAM (All)		读写	读写	读写	

注：1.这里的擦是指 Flash 页擦除；

2. Flash 主存储区片擦除是指 mass erase；

## 2.2.2 iCache

为了达到更高的系统性能，高速 CPU 与低速 Flash 之间需要增加指令缓存器，以提高指令执行效率。由于指令缓存器的存在，CPU 将可以工作在更高的主频。当 CPU 请求的指令在指令缓存器里面时，CPU 将可以无延时地获得指令、实现零等待执行。当前指令序列、指令预取序列、指令缓存器均未命中时，将重新读取 Flash，并回填更新 Cache 缓存；依此，相当于 Cache 中只存储了程序的跳转头。

指令缓存器的主要特性如下：

- 1KB iCache
- 支持相联方式：4WAY

### 2.2.2.1 软件接口

- 使能
  - ◆ 提供软件使能/关闭 Icache 的配置。开关切换条件无限制（见 FLASH\_AC.ICAHEN 位）。
- 复位
  - ◆ 提供软件清空 iCache 接口，必须在 iCache 关闭时才发起。复位与切换不可同时切换，先关闭 FLASH\_AC.ICAHEN，然后 FLASH\_AC.ICAHRST 写 1，然后就可打开 FLASH\_AC.ICAHEN。
- 锁定
  - ◆ 支持 iCache 锁定机制，软件配置将程序放入其指定的 way 中。当所有 way 均锁定完成后，新的数据将不会写入 cache 中。软件复位 cache 后，锁定状态自动清除。
- 补充说明

- ◆ 不支持 iCache 替换算法选择。
- ◆ 使用 iCache 时，CPU 写操作时不存在 WB/WT 选择。

### 2.2.2.2 寄存器描述

FLASH\_AC.ICAHEN 及 FLASH\_AC.ICAHRST，其分别为 iCache 使能开关以及 iCache 数据清零开关。

FLASH\_CAH.R.LOCKSTRT 及 FLASH\_CAH.R.LOCKSTOP，其分别为 iCache 对应方式锁定的开始锁存和停止锁存。iCache 复位后，FLASH\_CAH.R 寄存器自动恢复为复位值。iCache 锁定的详细使用方法见 2.2.2.3.3 ICache 锁定。

### 2.2.2.3 操作流程

#### 2.2.2.3.1 iCache 启用与禁用

用户软件可以随时开关 iCache。如果用户程序需要在主存储区和其它存储区之间跳转时，必须关闭 iCache 并且将 iCache 数据清零，否者会产生指令获取错误。

#### 2.2.2.3.2 iCache 数据刷新

iCache 设计为指令 Cache，当指令被应用软件更新或者指令在主存储区和其它存储区之间跳转时，软件必须将 FLASH\_AC.ICAHRST 位置 1 来清除指令 Cache 内的数据。

注意：FLASH\_AC.ICAHRST 位是只写位，读该位时返回为 0。

#### 2.2.2.3.3 iCache 锁定

用户软件控制 FLASH\_CAH.R 寄存器来将一些重复使用的代码锁存在 iCache 中来提高代码执行的效率。iCache 模块有 4 个锁存通道，每个通道的大小为整个 Cache 的 1/4。在使用单个通道时，必须确保需要锁存的代码量小于每个通道的大小。否者需要用更多的通道来锁存代码。可以按照下面的控制流程来使用锁存功能：

1. 将 FLASH\_CAH.R.LOCKSTRT[0]置 1；
2. 执行需要锁存在通道 0 的函数 1（函数 1 的代码量应该小于单个通道的大小）；
3. 函数 1 执行完成后，将 FLASH\_CAH.R.LOCKSTOP[0]置 1；
4. 接着将 FLASH\_CAH.R.LOCKSTRT[1]置 1；
5. 执行需要锁存在通道 1 的函数 2（函数 2 的代码量应该小于单个通道的大小）；
6. 函数 2 执行完成后，将 FLASH\_CAH.R.LOCKSTOP[1]置 1；

注意：1. 通道锁存时寄存器操作必须要按照固定的流程 – 先设置 FLASH\_CAH.R.LOCKSTRT – 再设置 FLASH\_CAH.R.LOCKSTOP；

2. 通道锁存的顺序必须要按照 0~3，否者会降低执行效率。

### 2.2.3 SRAM

SRAM 主要用于代码运行，存放程序执行过程中的变量和数据或堆栈，容量最大为 16KB。

SRAM 支持字节、半字、字的读写访问。

SRAM 支持代码运行（支持 SBus 和 ICode、DCode 的访问），可以在 SRAM 全速运行程序。SRAM 的最大

地址范围是 0x2000\_0000~0x2000\_3FFF。

SRAM 在 Standby 模式下数据可选保持；其他工作模式（Run/Sleep/Stop0/Stop2）数据可以正常保持。

主要特性如下：

- 容量最大总共为 16KB
- 支持字节/半字/字读写
- I/D/S/DMA 均可访问
- I/D BUS 可以 Remap 到 SRAM 全速运行程序

## 2.2.4 FLASH 寄存器描述

必须以字（32 位）的方式操作寄存器。

### 2.2.4.1 FLASH 寄存器总览

表 2-7 FLASH 寄存器总览

Offset	Register																																
000h	FLASH_AC	Reserved																															
		Reset Value																															
004h	FLASH_KEY	FKEY																															
		Reset Value																															
008h	FLASH_OPTKEY	OPTKEY																															
		Reset Value																															
00Ch	FLASH_STS	Reserved																															
		Reset Value																															
010h	FLASH_CTRL	Reserved																															
		Reset Value																															
014h	FLASH_ADD	FADD																															
		Reset Value																															
018h	FLASH_OB2	Reserved																															
		Reset Value																															
01Ch	FLASH_OB	Data1																															
		Data0																															
020h	FLASH_WRP	WRPT																															
		Reset Value																															
024h		Reserved																															
028h		Reserved																															

02Ch		Reserved
030h	FLASH_CAHRS	Reserved
	Reset Value	LOCKSTOP LOCKSTART

#### 2.2.4.2 FLASH 控制和状态寄存器

有关寄存器说明中的缩写，请见 1.1 节

#### 2.2.4.2.1 FLASH 访问控制寄存器 (FLASH\_AC)

偏移地址: 0x00

复位值: 0x0000 0030

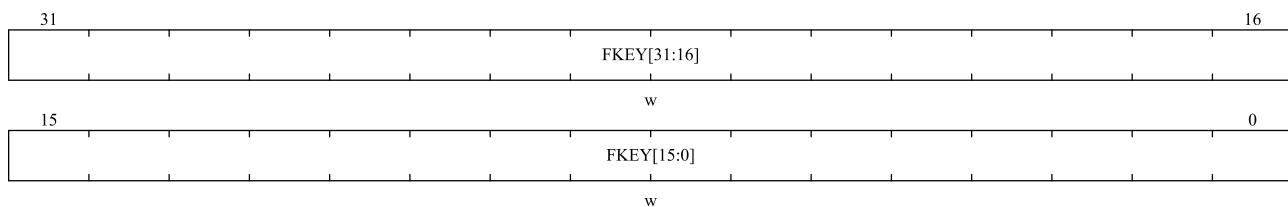
位域	名称	描述
31:8	Reserved	保留, 必须保持复位值
7	ICAHEN	iCache 使能 0: 关闭 iCache 1: 启用 iCache
6	ICAHRST	iCache 复位 0: 写'0'无效 1: 写'1'复位
5	PRFTBFS	预取缓冲区状态 该位指示预取缓冲区的状态 0: 预取缓冲区关闭 1: 预取缓冲区开启
4	PRFTBFE	预取缓冲区使能 0: 关闭预取缓冲区 1: 启用预取缓冲区
3	Reserved	保留, 必须保持复位值
2:0	LATENCY	时延 这些位表示 HCLK 周期与闪存访问时间的比例 000: 零周期时延, 当 $0 < \text{HCLK} \leq 32\text{MHz}$ 001: 一个周期时延, 当 $32\text{MHz} < \text{HCLK} \leq 64\text{MHz}$ 010: 两个周期时延, 当 $64\text{MHz} < \text{HCLK} \leq 96\text{MHz}$ 011: 三个周期时延, 当 $96\text{MHz} < \text{HCLK}$ 其他值: 保留

位域	名称	描述
		以上适用于 C 版及 C 版一下版本; 000: 零周期时延, 当 $0 < \text{HCLK} \leq 39\text{MHz}$ 001: 一个周期时延, 当 $39\text{MHz} < \text{HCLK} \leq 78\text{MHz}$ 010: 两个周期时延, 当 $78\text{MHz} < \text{HCLK} \leq 117\text{MHz}$ 011: 三个周期时延, 当 $117\text{MHz} < \text{HCLK}$ 其他值: 保留 以上适用于 D 版及 D 版以上版本;

#### 2.2.4.2.2 FLASH 键寄存器 (FLASH\_KEY)

偏移地址: 0x04

复位值: 0x0000 0000

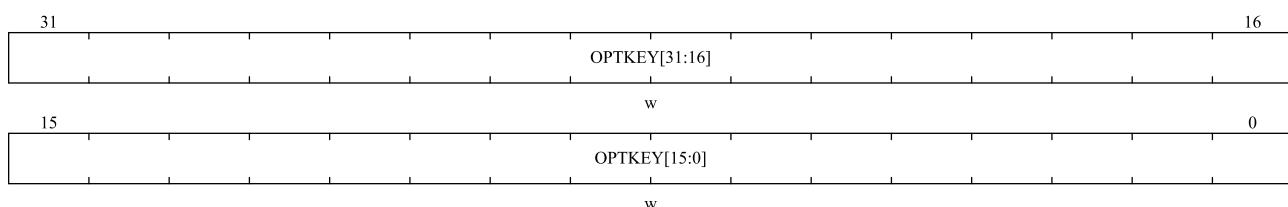


位域	名称	描述
31:0	FKEY	用于解锁 FLASH_CTRL.LOCK 位

#### 2.2.4.2.3 FLASH OPTKEY 寄存器 (FLASH\_OPTKEY)

偏移地址: 0x08

复位值: 0x0000 0000



位域	名称	描述
31:0	OPTKEY	用于解锁 FLASH_CTRL.OPTWE 位

#### 2.2.4.2.4 FLASH 状态寄存器 (FLASH\_STS)

偏移地址: 0x0C

复位值: 0x0000 0000

31	Reserved															16	
15	Reserved						6	5	4	3	2	1	0	rc_w1	rc_w1	rc_w1	r
							EOP	WRPERR	Reserved	PGERR	Reserved	BUSY					

位域	名称	描述
31:6	Reserved	保留，必须保持复位值
5	EOP	操作结束 当闪存操作（编程/擦除）完成时，硬件设置这位为'1'，写入'1'可以清除这位状态。 注：每次成功的编程或擦除都会设置 EOP 状态。
4	WRPERR	写保护错误 试图对写保护的闪存地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。
3	Reserved	保留，必须保持复位值
2	PGERR	编程错误 试图对内容不是'0xFFFF_FFFF'的地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。 注：进行编程操作之前，必须先清除 FLASH_CTRL.START 位。
1	Reserved	保留，必须保持复位值
0	BUSY	忙 该位指示闪存操作正在进行。在闪存操作开始时，该位被设置为'1'；在操作结束或发生错误时该位被清除为'0'。

#### 2.2.4.2.5 FLASH 控制寄存器 (FLASH\_CTRL)

偏移地址: 0x10

复位值: 0x0000 0080

31	Reserved															16	
15	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reserved	EOPITE	Reserved	ERRITE	OPTWE	Reserved	LOCK	START	OPTER	OPTPG	Reserved	MER	PER	PG	rw	rw	rw	rw

位域	名称	描述
31:13	Reserved	保留，必须保持复位值
12	EOPITE	允许操作完成中断 该位允许在 FLASH_STS.EOP 位变为'1'时产生中断。 0: 禁止产生中断 1: 允许产生中断
11	Reserved	保留，必须保持复位值
10	ERRITE	允许错误状态中断

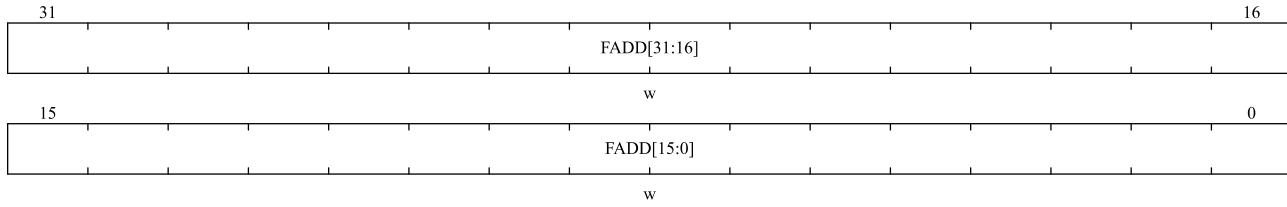
位域	名称	描述
		该位允许在发生 Flash 错误时产生中断（当 FLASH_STS.PGERR/WRPERR 置为'1'时）。 0: 禁止产生中断 1: 允许产生中断
9	OPTWE	允许写选项字节 当该位为'1'时，允许对选项字节进行编程操作。当在 FLASH_OPTKEY 寄存器写入正确的键序列后，该位被置为'1'。 软件可清除此位。
8	Reserved	保留，必须保持复位值
7	LOCK	锁定 只能写'1'。当该位为'1'时表示 Flash 和 FLASH_CTRL 被锁住。在检测到正确的解锁序列后，硬件清除此位为'0'。 在一次不成功的解锁操作后，下次系统复位前，该位不能再被改变。
6	START	开始 当该位为'1'时将触发一次擦除操作。该位只可由软件置为'1'并在 FLASH_STS.BUSY 变为'1'时清除为'0'。
5	OPTER	擦除选项字节 0: 不开启选项字节擦除模式 1: 开启选项字节擦除模式
4	OPTPG	编程选项字节 0: 不开启选项字节编程模式 1: 开启选项字节编程模式
3	Reserved	保留，必须保持复位值
2	MER	片擦除 0: 不开启片擦除模式 1: 开启片擦除模式
1	PER	页擦除 0: 不开启页擦除模式 1: 开启页擦除模式
0	PG	编程 0: 不开启编程模式 1: 开启编程模式

注:关于编程及擦除请参考 2.2.1.4 节。

#### 2.2.4.2.6 FLASH 地址寄存器 (FLASH\_ADD)

偏移地址: 0x14

复位值: 0x0000 0000

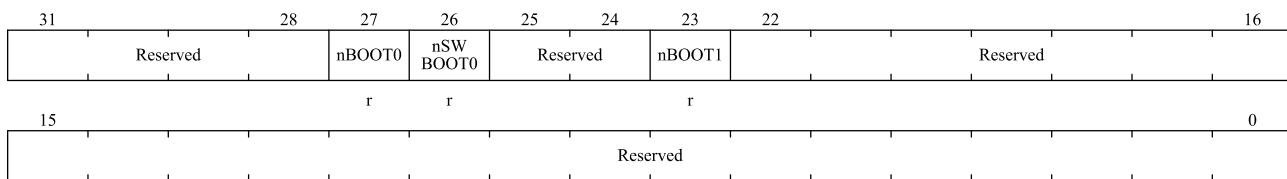


位域	名称	描述
31:0	FADD	闪存地址 当进行编程时选择要编程的地址，当进行页擦除时选择要擦除的页。 注意：当FLASH_STS.BUSY位为'1'时，不能写这个寄存器。

#### 2.2.4.2.7 选项字节寄存器 2 (FLASH\_OB2)

偏移地址: 0x18

复位值: 0x0C80 0000



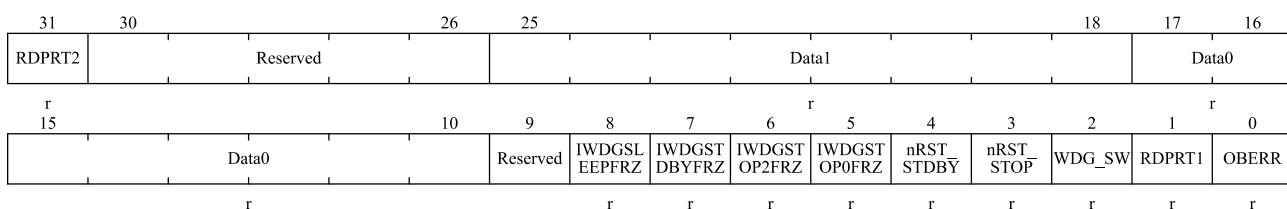
位域	名称	描述
31:28	Reserved	保留，必须保持复位值
27	nBOOT0	nBOOT0 注：只读位。
26	nSWBOOT0	nSWBOOT0 注：只读位。
25:24	Reserved	保留，必须保持复位值
23	nBOOT1	nBOOT1 注：只读位。
22:0	Reserved	保留，必须保持复位值

注：nBOOT0、nSWBOOT0、nBOOT1的具体组合功能见2.1.3启动管理章节。

#### 2.2.4.2.8 选项字节寄存器 (FLASH\_OB)

偏移地址: 0x1C

复位值: 0x03FF FFFF



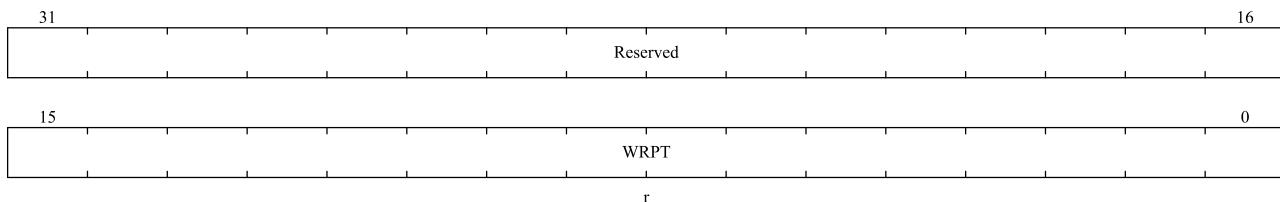
位域	名称	描述
31	RDPRT2	读保护 L2 级别 0: 读保护 L2 级别未使能 1: 读保护 L2 级别使能 注: 只读位。
30:26	Reserved	保留, 必须保持复位值
25:18	Data1[7:0]	Data1 注: 只读位。
17:10	Data0[7:0]	Data0 注: 只读位。
9	Reserved	未使用 (如果对应选项字节位被写入, 则可以从该寄存器中读到, 对芯片没有影响)
8	IWDGSLEEPFRZ	IWDG 在 sleep 模式下冻结使能 0: iwdg 在 sleep 模式下冻结 1: iwdg 在 sleep 模式下非冻结 注: 只读位。
7	IWDGSTDBYFRZ	IWDG 在 standby 模式下冻结使能 0: iwdg 在 standby 模式下冻结 1: iwdg 在 standby 模式下非冻结 注: 只读位。
6	IWDGSTOP2FRZ	IWDG 在 stop2 模式下冻结使能 0: iwdg 在 stop2 模式下冻结 1: iwdg 在 stop2 模式下非冻结 注: 只读位。
5	IWDGSTOP0FRZ	IWDG 在 stop0 模式下冻结使能 0: iwdg 在 stop0 模式下冻结 1: iwdg 在 stop0 模式下非冻结 注: 只读位。
4	nRST_STDBY	进入 Standby 模式复位配置 0: 进入 Standby 模式后立即产生复位 1: 进入 Standby 模式后不产生复位 注: 只读位。
3	nRST_STOP	进入 STOP0/STOP2 模式复位配置 0: 进入 STOP0/STOP2 模式后立即产生复位 1: 进入 STOP0/STOP2 模式后不产生复位 注: 只读位。
2	WDG_SW	看门狗设置 0: 硬件看门狗 1: 软件看门狗 注: 只读位。
1	RDPRT1	读保护 L1 级别 0: 读保护 L1 级别未使能 1: 读保护 L1 级别使能

位域	名称	描述
		注：只读位。
0	OBERR	选项字节错误 当该位为'1'时表示选项字节和它的补码不匹配 注：只读位。

#### 2.2.4.2.9 写保护寄存器 (FLASH\_WRP)

偏移地址: 0x20

复位值: 0x0000 FFFF

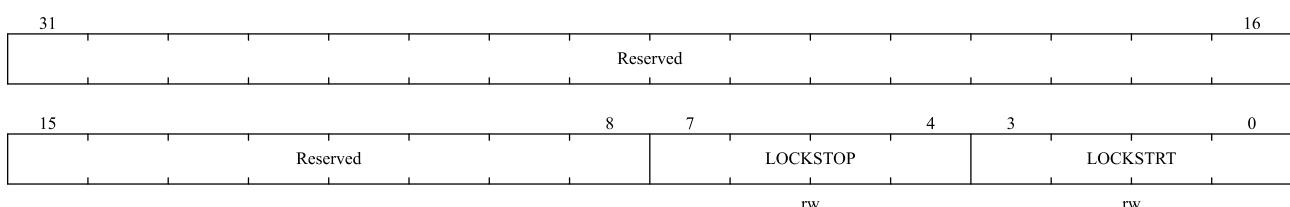


位域	名称	描述
31:16	Reserved	保留，必须保持复位值
15:0	WRPT	写保护 该寄存器包含由选项字节区加载的写保护选项字节。 0：写保护生效； 1：写保护失效。 注：只读位。

#### 2.2.4.2.10 CAHR 寄存器 (FLASH\_CAHRS)

偏移地址: 0x30

复位值: 0x0000 0000



位域	名称	描述
31:8	Reserved	保留，必须保持复位值
7:4	LOCKSTOP[3:0]	iCache 锁定停止（详细操作说明见 2.2.2.3.3 iCache 锁定章节） 0：不使能 1：使能
3:0	LOCKSTRT[3:0]	iCache 锁定开始 0：不使能 1：使能



## 3 电源控制 (PWR)

### 3.1 通用描述

PWR 是用于控制不同模块在不同功耗模式下的状态的电源管理单元。它的主要功能是控制 MCU 进入不同的功耗模式，并在事件或中断发生时唤醒。MCU 支持 RUN、SLEEP、STOP0、STOP2 和 STANDBY 模式。PWR 控制不同功耗模式下的 LDO、时钟源、复位和 Flash/SRAM/GPIO 状态。

#### 3.1.1 电源

MCU 需要 2.4V 至 3.6V 的工作电压(VDD)。VDD 为外部电源。

◆ 独立 A/D 转换器电源。为了提高转换精度，ADC 使用单独的电源来过滤和屏蔽来自印刷电路板上毛刺的干扰。

- ADC 电源引脚为 VDDA。
- 独立的电源地 VSSA。
- 没有 VREF+和 VREF-引脚，它们连接到芯片内部的 ADC 电源（VDDA）和地（VSSA）。

◆ 电压调节器

- 根据电源管理要求，电压调节器有 MR（主调节器）和 BKR（备用域调节器）。
- 电压调节器（MR、BKR）在复位后始终启用。
- 电压调节器有几种不同的工作模式，具体取决于应用：
  - RUN 模式：调节器在正常功耗模式下提供 VDDD 电源（内核、内存和外设）。
  - STOP0 模式：调节器在低功耗模式下提供 VDDDRRET 电源以保存寄存器和 SRAM 的内容。
  - STOP2 模式：关闭 MR 并切换到 BKR 供电以保存寄存器和 SRAM 内容。
  - STANDBY 模式：MR 调节器停止供电，BKR 可根据需要开启或关闭。

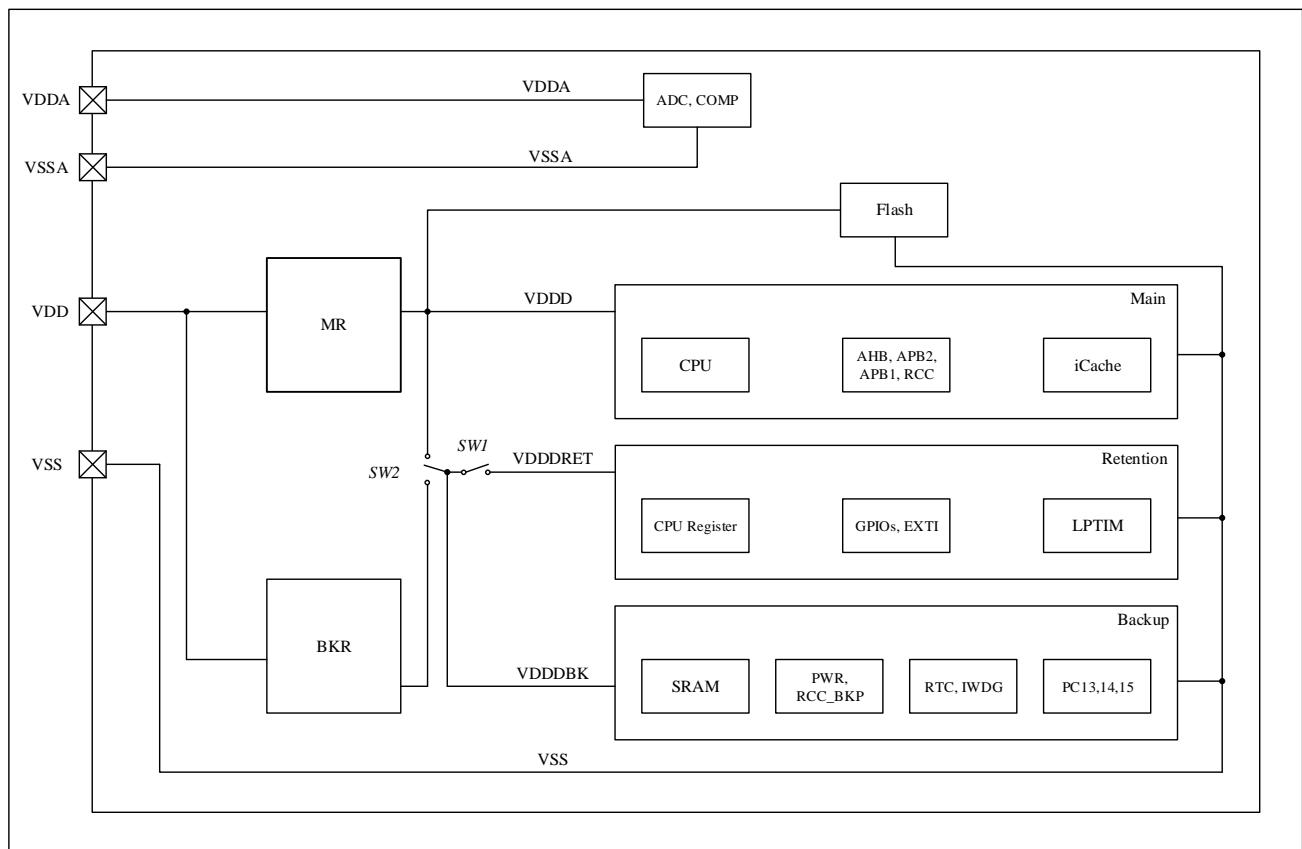
#### ■ MR

它是 VDDD 电源的主调节器。主要用于 MCU 的 RUN 模式、SLEEP 模式和 STOP0 模式。在 STOP0 模式下 MR 可选处于低功耗模式，在其他两种模式下处于正常模式。

#### ■ BKR

该备用调节器用于 STOP2 和 STANDBY 模式。

图 3-1 电源框图

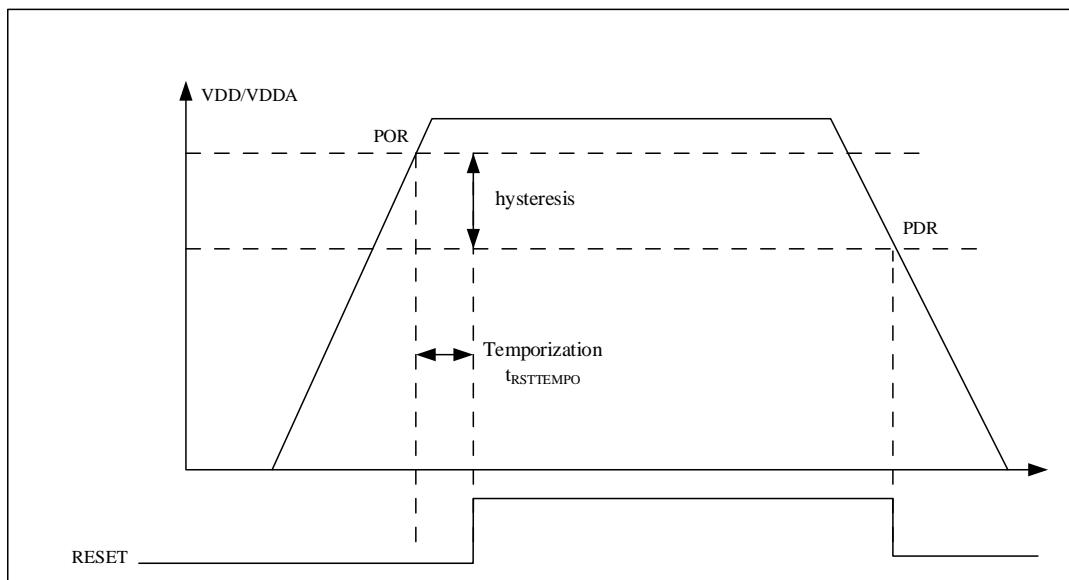


### 3.1.2 电源监控

#### 3.1.2.1 上电复位 (POR) 和掉电复位 (PDR)

上电复位(POR)和掉电复位(PDR)电路集成在芯片内部。当VDD/VDDA低于规定的限制电压VPOR/VPDR时，系统保持在复位状态，无需外部复位电路。有关上电和掉电复位的详细信息，请参阅数据手册的电气特性部分。

图 3-2 上电复位/掉电复位波形图

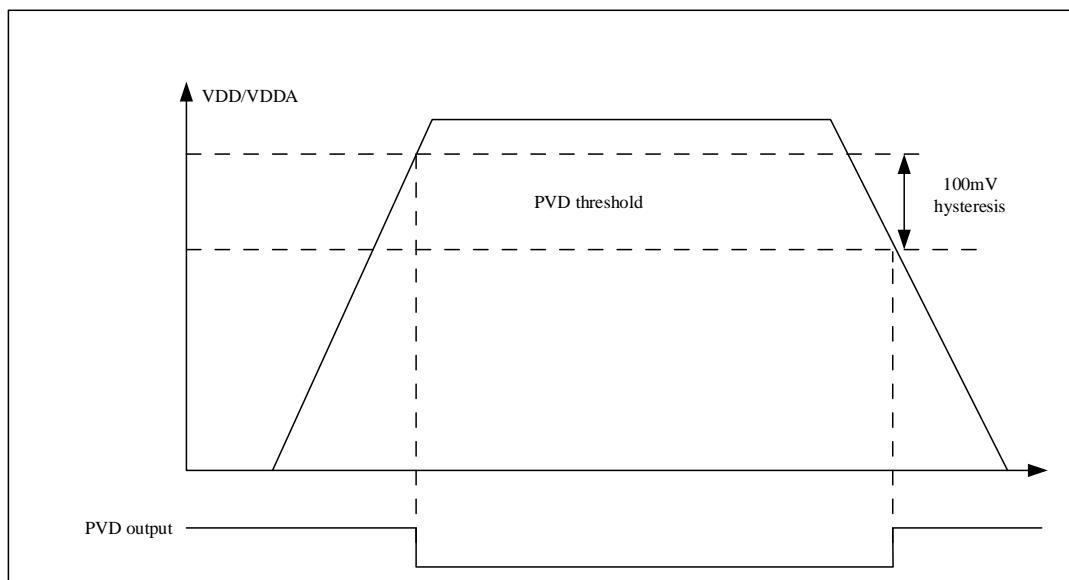


### 3.1.2.2 可编程电压监测器 (PVD)

PVD 通过将 VDD 电压与电源控制寄存器(PWR\_CTRL)相关 bits 进行比较来监控电源。PWR\_CTRL.MSB 和 PWR\_CTRL.PLS 选择监控电压的阈值。通过设置 PWR\_CTRL.PVDE 启用 PVD。

PWR\_CTRLSTS.PVDO 标志用于指示 VDD 是否高于/低于 PVD 电压阈值。该事件在内部连接到外部中断的中断线 16，如果在外部中断寄存器中使能了中断，则会产生中断。根据外部中断线 16 的上升/下降沿触发设置，当 VDD 下降到 PVD 阈值以下和/或 VDD 上升到 PVD 阈值以上时，会发生 PVD 中断。例如，此功能可用于执行紧急关机任务。

图 3-3 PVD 阈值图



### 3.1.3 NRST

NRST 是一个模拟 PAD。整个电路在 VDDBK/VDDA 域产生系统复位。

## 3.2 功耗模式

MCU 总共有五种电源模式：RUN、SLEEP、STOP0、STOP2 和 STANDBY。不同的模式有不同的性能和功耗。MCU 功耗模式总结如下所示。

表 3-1 功耗模式

项 模式	状态	进入	退出
RUN	CPU 运行，外设可配置。	上电、系统复位或从其他电源模式唤醒。	进入 SLEEP, STOP0, STOP2, STANDBY.
SLEEP	CPU 休眠，所有外设可配置，调节器开启，所有数字外设保持供电。中断和事件可以唤醒 CPU。	WFI  CPU 从 ISR 返回  WFE	任何中断唤醒事件。
STOP0	CPU 进入 SLEEPDEEP，对外设时钟进行门控。MR 可选处于 LP 模式下。HSE/HSI/PLL 禁用。LSE/LSI 可配置。RTC 可选。SRAM/所有寄存器保持。所有 IO 保持。唤醒后，默认启用 HSI。	WFI/WFE:  1) SCB_SCR.SLEEPDEEP = 1, 无挂起的中断/事件。  2) PWR_CTRL.LPS = 0/1, PWR_CTRL.PDS = 0	任何通过 EXTI、NRST、IWDG 的中断唤醒事件。
STOP2	CPU 进入 SLEEPDEEP，CPU 寄存器保持。MR 关闭，LSE/LSI 可配置，HSE/HSI/PLL 禁用，所有 IO 保持，SRAM 16KB 保持。RTC、LSE/LSI、LPTIMER、PVD 可选运行。唤醒后，默认启用 HSI。	WFI/WFE:  1) SCB_SCR.SLEEPDEEP = 1, 无挂起的中断/事件。  2) PWR_CTRL.LPS = 0, PWR_CTRL.PDS = 0 和 PWR_CTRL2.STOP2S = 1	NRST、IWDG、PVD、RTC 定时器或时间戳或入侵事件、LPTIMER，所有可配置为 EXTI 的 GPIO。
STANDBY	MR 关闭，LSE/LSI 开启，所有其他时钟关闭，所有 IO 保持，但是唤醒后会复位成默认状态，除了 NRST、PA0_WKUP、PA8、PC13、PC14_OSC32_IN、PC15_OSC32_OUT。可选：16KB SRAM 保持、RTC 工作于 LSE/LSI、IWDG。	WFI/WFE:  1) SCB_SCR.SLEEPDEEP = 1, 无挂起的中断/事件。  2) PWR_CTRL.LPS = 0, PWR_CTRL.PDS = 1	NRST、3个WKUP_IO唤醒引脚、RTC定时器或闹钟或时间戳或入侵事件、IWDG唤醒。  唤醒后复位，所有都需要重新配置。

注意：

1. STOP0 模式，唤醒后，代码可以从停止位置恢复并继续。不需要任何初始化。
2. STOP2 模式，唤醒后，如果堆栈和全局变量在 SRAM 内。代码可以从停止的位置恢复和继续。需要重

新初始化外设和 PWR。

不同模块在不同功耗模式下的运行使能情况如下表所示：

表 3-2 模块运行状态

Modules	Run	Sleep	Stop 0		Stop 2		Standby	
			-	Wakeup capability	-	Wakeup capability	-	Wakeup capability
CPU <sup>(1)</sup>	Y	-	-	-	-	-	-	-
Flash memory	O	O	-	-	-	-	-	-
SRAM	Y	Y <sup>(2)</sup>	Y	-	Y	-	O	-
Backup Registers	Y	Y	Y	-	Y	-	Y	-
POR(VDDD)	Y	Y	Y	Y	-	-	-	-
POR(VDDA)	Y	Y	Y	Y	Y	Y	Y	Y
POR(VDDBK)	Y	Y	Y	Y	Y	Y	Y	Y
PVD	O	O	O	O	O	O	-	-
DMA	O	O	-	-	-	-	-	-
HSI	O	O	(3)	-	(3)	-	-	-
HSE	O	O	-	-	-	-	-	-
LSI	O	O	O	-	O	-	O	-
LSE	O	O	O	-	O	-	O	-
Clock Security System (CSS)	O	O	-	-	-	-	-	-
LSE Clock Security System (LSE-CSS)	O	O	O	Y	O	Y	O	Y
RTC / Auto wakeup	O	O	O	O	O	O	O	O
Number of RTC Tamper pins	3 <sup>(4)</sup>	3 <sup>(4)</sup>	3 <sup>(4)</sup>	O <sup>(4)</sup>	3 <sup>(4)</sup>	O <sup>(4)</sup>	3 <sup>(4)</sup>	O <sup>(4)</sup>
USARTx (x=1, 2)	O	O	-	-	-	-	-	-
UARTx(x=3, 4)	O	O	-	-	-	-	-	-
I2Cx (x=1, 2)	O	O	-	-	-	-	-	-
SPIx (x=1, 2)	O	O	-	-	-	-	-	-
CANx(x=1)	O	O	-	-	-	-	-	-
ADCx (x=1)	O	O	-	-	-	-	-	-
COMPx(x=1, 2, 3)	O	O	O	O	-	-	-	-
Temperature sensor	O	O	-	-	-	-	-	-
TIMx(x=1~6, 8)	O	O	-	-	-	-	-	-

LPTIM	O	O	O	O	O	O	-	-
IWDG	O	O	O	O	O	O	O	O
WWDG	O	O	-	-	-	-	-	-
SysTick timer	O	O	-	-	-	-	-	-
CRC	O	O	-	-	-	-	-	-
GPIOS	O	O	O	O	O	O	3 pins <sup>(5)</sup>	3 pins <sup>(5)</sup>

注意：

1. Y: 是 (使能), O: 可选 (默认禁能, 可以软件使能), -: 不可用。
2. SRAM 时钟可以开启或关闭。
3. 一些可以从停止模式中唤醒的外设可以要求 HSI 使能。在这种情况下, HSI 由外设使能并且仅提供给外设。
4. 三个引脚支持入侵功能, 即 PA0(Tamper2)、PA8(Tamper3)和 PC13(Tamper1)
5. 可以从 STANDBY 唤醒的引脚有 PA0(WKUP2)、PA8(WKUP1)、PC13(WKUP3)、NRST。

### 3.2.1 SLEEP 模式

CPU 停止, 包括 Cortex®-M4 内核周围的外设 (如 NVIC、SysTick 等) 在内的所有外设都可以运行并在发生中断或事件时唤醒 CPU。

#### 3.2.1.1 进入 SLEEP 模式

通过执行 WFI (等待中断) 或 WFE (等待事件) 指令和 SCB\_SCR.SLEEPDEEP=0 进入 SLEEP 模式。根据 SCB\_SCR.SLEEPONEXIT, 进入 SLEEP 模式有两种方式:

- SLEEP-NOW: 如果 SCB\_SCR.SLEEPONEXIT=0, 则立即执行 WFI 或 WFE 指令, 系统立即进入 SLEEP 模式。
- SLEEP-ON-EXIT: 如果 SCB\_SCR.SLEEPONEXIT=1, 系统从最低优先级 ISR 退出时立即进入 SLEEP 模式。

#### 3.2.1.2 退出 SLEEP 模式

如果使用 WFI 指令进入 SLEEP 模式, 任何 NVIC 中断都可以将设备从 SLEEP 模式唤醒。

如果使用 WFE 指令进入 SLEEP 模式, MCU 将在事件发生时立即退出 SLEEP 模式。唤醒事件可以通过以下方式生成:

- 在外设控制寄存器而不是 NVIC 中使能中断, 并使能 SCB\_SCR.SEVONPEND。当 MCU 被 WFE 唤醒时, 外设中断挂起位和外设 NVIC 中断通道挂起位 (在 NVIC 中断清除挂起寄存器中) 必须清零。
- 配置外部或内部 EXTI 事件模式。当 MCU 唤醒时, 不需要清除外设中断挂起位和外设 NVIC 中断通道挂起位 (在 NVIC 中断清除挂起寄存器中), 因为没有设置事件线对应的挂起位。该模式提供最短的唤醒时间, 因为没有时间花在中断进入或退出上。

### 3.2.2 STOP0 模式

STOP0 模式是基于 Cortex®-M4 的 Deep-Sleep 模式（外部时钟门控）和相应的电源管理。工作在 VDDD 的所有时钟停止、PLL、HSI、HSE 禁用、LSI 和 LSE 运行。RTC 可以保持运行，一些具有唤醒功能的外设可以在检测到唤醒条件时启用 HSI。16KB SRAM 和所有寄存器保持。所有 I/O 状态必须与运行状态一致。

#### 3.2.2.1 进入 STOP0 模式

进入 STOP0 模式时，主要区别在于设置 SCB\_SCR.SLEEPDEEP = 1 和 PWR\_CTRL.PDS = 0。另一个区别是 MR 可以处于正常模式或低功耗模式，具体取决于 PWR\_CTRL.LPS。当 PWR\_CTRL.LPS = 1 时，MR 处于低功耗模式。当 PWR\_CTRL.LPS = 0 时，MR 处于正常模式。

如果正在进行 FLASH 操作，进入 STOP0 模式的时间将延迟到存储器访问完成。

如果正在访问 APB 区域，则进入 STOP0 模式的时间将延迟到 APB 访问完成。

在 STOP0 模式下，可以使用以下外设：

- 独立看门狗(IWDG)可选：一旦启用，它将一直计数，直到产生复位。
- RTC 可选：可以通过 RCC\_BDCTRL.RTCEN 启用。
- 内部 RC 振荡器（LSI RC）可选：可以通过 RCC\_CTRLSTS.LSIEN 启用。
- 外部 32.768kHz 晶振（LSE OSC）可选：可通过 RCC\_BDCTRL.LSEEN 启用。

进入 STOP0 模式时应禁用 ADC，以避免不必要的功耗。

**注意：**如果应用程序需要在进入停止模式之前禁用外部时钟，必须先将系统时钟切换到 HSI，然后禁用 RCC\_CTRL.HSEEN 位。如果在进入停止模式时 RCC\_CTRL.HSEEN 位保持有效并且移除了外部时钟（外部振荡器），则必须启用时钟安全系统(CSS)功能以检测任何外部振荡器故障。

#### 3.2.2.2 退出 STOP0 模式

当中断或唤醒事件唤醒 STOP0 模式时，HSI RC 振荡器作为系统时钟。如果 MR 处于低功耗模式，则需要额外的启动时间才能从 STOP0 模式唤醒。进入 STOP0 模式时保持 MR 处于正常模式可以减少启动时间，但功耗较高。

### 3.2.3 STOP2 模式

STOP2 模式是基于 Cortex®-M4 的 Deep-Sleep 模式（外部时钟门控）和相应的电源门控。工作在 VDDD 的所有时钟停止、PLL、HSI、HSE 禁用、LSI 和 LSE 运行。RTC 可以保持运行，一些具有唤醒功能的外设可以在检测到唤醒条件时启用 HSI。CPU 寄存器、80 字节备份寄存器保持。16KB SRAM 保持。所有 I/O 状态与运行状态一致。从 STOP2 模式唤醒后，代码继续从休眠处运行，需要重新配置使用的外设接口和 PWR。

#### 3.2.3.1 进入 STOP2 模式

要进入 STOP2 模式，寄存器位应配置为：SCB\_SCR.SLEEPDEEP = 1，PWR\_CTRL2.STOP2S = 1，PWR\_CTRL.PDS = 0，PWR\_CTRL.LPS = 0。

在 STOP2 模式下，如果 FLASH 正在操作，进入 STOP2 模式的时间将延迟到内存访问完成。

如果正在访问 APB 区域，则进入 STOP2 模式的时间将延迟到 APB 访问完成。

在 STOP2 模式下，可以使用以下外设：

- 独立看门狗(IWDG)可选：一旦启用，它将一直计数，直到产生复位。
- RTC 可选：可以通过 RCC\_BDCTRL.RTCEN 开启。
- 内部 RC 振荡器（LSI RC）可选：可以通过 RCC\_CTRLSTS.LSIEN 开启。
- 外部 32.768kHz 晶振（LSE OSC）可选：可通过 RCC\_BDCTRL.LSEEN 开启。

进入 STOP2 模式时应禁用 ADC，以避免不必要的功耗。

注意：在 STOP2 模式下，如果数据（全局变量、堆栈等）需要保持，则应将它们存储在 SRAM 中。

### 3.2.3.2 退出 STOP2 模式

当 STOP2 模式被中断或唤醒事件唤醒时，HSI RC 振荡器被设置为系统时钟。退出 STOP2 后，代码将从其停止位置恢复执行。唤醒后需要进行外设初始化。如果有访问需要，应在 RCC 中重新开启 PWR 和其他模块的时钟。

### 3.2.4 STANDBY 模式

STANDBY 模式是基于 Cortex®-M4 的 Deep-Sleep 模式，电压调节器关闭，PLL、HSI、HSE 关闭，LSI 和 LSE 运行。SRAM 保持、RTC 和 IWDG 是可选的。所有 IO 保持，但是唤醒后会复位成默认状态，除了 NRST、PA0\_WKUP、PA8、PC13、PC14\_OSC32\_IN、PC15\_OSC32\_OUT。

进入 STANDBY 模式之前，如果 PA13 和 PA14 用做非调试管脚，且引脚配置成输入模式，需要在 PA13 和 PA14 引脚外部加强上下拉，上下拉电阻建议 10KΩ 以内。

#### 3.2.4.1 进入 STANDBY 模式

通过执行 WFI/WFE，并设置 SCB\_SCR.SLEEPDEEP = 1 和 PWR\_CTRL.PDS = 1 进入 STANDBY 模式。

如果 FLASH 正在操作，进入 STANDBY 模式的时间将延迟到内存访问完成。

如果正在访问 APB 区域，则进入 STANDBY 模式的时间将延迟到 APB 访问完成。

在 STANDBY 模式下，可以使用以下外设：

- 独立看门狗(IWDG)可选：一旦启用，它将一直计数，直到产生复位。
- RTC 可选：可以通过 RCC\_BDCTRL.RTCEN 开启。
- 内部 RC 振荡器（LSI RC）可选：可以通过 RCC\_CTRLSTS.LSIEN 开启。
- 外部 32.768kHz 晶振（LSE OSC）可选：可通过 RCC\_BDCTRL.LSEEN 开启。
- SRAM 数据保持可选：可以通过 PWR\_CTRL2.SRSTBRET 开启。

进入 STANDBY 模式时应禁用 ADC，以避免不必要的功耗。

#### 3.2.4.2 退出 STANDBY 模式

当外部复位(NRST 引脚)、IWDG 复位、WKUP 引脚的上升/下降沿或 RTC 事件发生时，MCU 退出 STANDBY 模式。除电源控制状态寄存器 (PWR\_CTRLSTS) 外，所有寄存器在从 STANDBY 状态唤醒后都会被复位。

从 STANDBY 模式唤醒后，代码执行与上电相同（检测到 BOOT 引脚、复位向量初始化等）。PWR\_CTRLSTS.STBYF 标志表示 MCU 退出 STANDBY 模式。

## 3.3 PWR 寄存器

### 3.3.1 PWR 寄存器总览

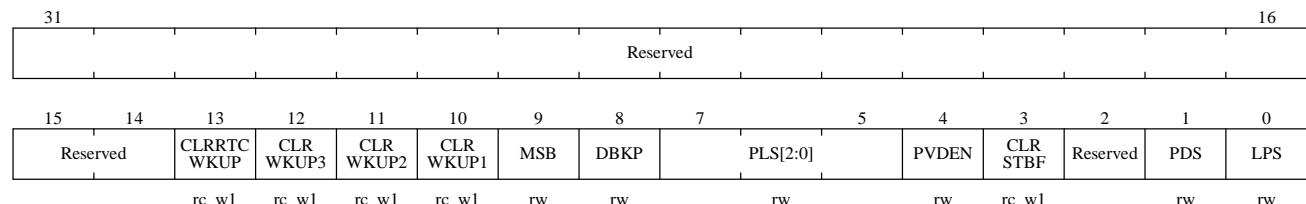
表 3-3 PWR 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	5	4	3	2	1	0	
000h	PWR_CTRL1																																
	Reset Value																																
004h	PWR_CTRLSTS																																
	Reset Value																																
008h	PWR_CTRL2																																
	Reset Value																																

### 3.3.2 电源控制寄存器(PWR\_CTRL)

偏移地址: 0x00

复位值: 0x0000 0000 (从 STANDBY 模式唤醒后复位)



位域	名称	描述
31:14	Reserved	保留，必须保持复位值。
13	CLRRTCWKUP	清除 RTC 唤醒标志。该位总是读为 0。 0: 无作用 1: 在 2 个系统时钟周期后清除唤醒标志。
12	CLRWKUP3	清除 PC13 唤醒标志。该位总是读为 0。 0: 无作用 1: 在 2 个系统时钟周期后清除唤醒标志。
11	CLRWKUP2	清除 PA0 唤醒标志。该位总是读为 0。 0: 无作用 1: 在 2 个系统时钟周期后清除唤醒标志。
10	CLRWKUP1	清除 PA8 唤醒标志。该位总是读为 0。 0: 无作用

位域	名称	描述																																				
		1: 在 2 个系统时钟周期后清除唤醒标志。																																				
9	MSB	<p>4 位 PVD 阈值设置中的最高位。</p> <p>PVD 阈值控制如下:</p> <p>当 MSB = 0 时, PVD 阈值为:</p> <table border="1"> <thead> <tr> <th>[MSB, PWR_CTRL.PLS] (4 bits)</th><th>Voltage</th></tr> </thead> <tbody> <tr><td>0000</td><td>2.18v</td></tr> <tr><td>0001</td><td>2.28v</td></tr> <tr><td>0010</td><td>2.38v</td></tr> <tr><td>0011</td><td>2.48v</td></tr> <tr><td>0100</td><td>2.58v</td></tr> <tr><td>0101</td><td>2.68v</td></tr> <tr><td>0110</td><td>2.78v</td></tr> <tr><td>0111</td><td>2.88v</td></tr> </tbody> </table> <p>当 MSB = 1 时, PVD 阈值为:</p> <table border="1"> <thead> <tr> <th>[MSB, PWR_CTRL.PLS](4 bits)</th><th>Voltage</th></tr> </thead> <tbody> <tr><td>1000</td><td>1.78v</td></tr> <tr><td>1001</td><td>1.88v</td></tr> <tr><td>1010</td><td>1.98v</td></tr> <tr><td>1011</td><td>2.08v</td></tr> <tr><td>1100</td><td>3.28v</td></tr> <tr><td>1101</td><td>3.38v</td></tr> <tr><td>1110</td><td>3.48v</td></tr> <tr><td>1111</td><td>3.58v</td></tr> </tbody> </table>	[MSB, PWR_CTRL.PLS] (4 bits)	Voltage	0000	2.18v	0001	2.28v	0010	2.38v	0011	2.48v	0100	2.58v	0101	2.68v	0110	2.78v	0111	2.88v	[MSB, PWR_CTRL.PLS](4 bits)	Voltage	1000	1.78v	1001	1.88v	1010	1.98v	1011	2.08v	1100	3.28v	1101	3.38v	1110	3.48v	1111	3.58v
[MSB, PWR_CTRL.PLS] (4 bits)	Voltage																																					
0000	2.18v																																					
0001	2.28v																																					
0010	2.38v																																					
0011	2.48v																																					
0100	2.58v																																					
0101	2.68v																																					
0110	2.78v																																					
0111	2.88v																																					
[MSB, PWR_CTRL.PLS](4 bits)	Voltage																																					
1000	1.78v																																					
1001	1.88v																																					
1010	1.98v																																					
1011	2.08v																																					
1100	3.28v																																					
1101	3.38v																																					
1110	3.48v																																					
1111	3.58v																																					
8	DBKP	<p>禁用备份域写保护。</p> <p>在复位状态下, RTC 和备份寄存器受到保护以防止发生非法访问。必须设置该位以启用对这些寄存器的写保护。</p> <p>0: 禁止访问 RTC 和备份寄存器 1: 启用对 RTC 和备份寄存器的访问</p> <p>注意: 如果 HSE 除以 128 作为 RTC 时钟, 该位必须保持为 1。</p>																																				
7:5	PLS[2:0]	<p>PVD 等级选择。</p> <p>当需要设置 PVD 阈值时, 请与 PWR_CTRL.MSB 一起配置这些位。</p> <p>注: 详见数据手册的电气特性部分。</p>																																				
4	PVDEN	<p>电源电压检测器 (PVD) 使能。软件控制。</p> <p>0: 禁用 PVD 1: 启用 PVD</p>																																				
3	CLRSTBF	<p>清除 STANDBY 标志。该位总是读为 0。</p> <p>0: 无效果。 1: 清除 STANDBY.STBYF 标志。</p>																																				
2	Reserved	保留, 必须保持复位值。																																				
1	PDS	<p>掉电深度睡眠。</p> <p>由软件设置和清除该位并与 PWR_CTRL.LPS 一起配置使用。</p>																																				

位域	名称	描述
		0: 当 CPU 进入深度睡眠时进入停止模式。调节器状态取决于 PWR_CTRL.LPS。 1: 当 CPU 进入深度睡眠模式时进入待机模式。
0	LPS	低功耗深度睡眠。 由软件设置和清除该位并与 PWR_CTRL.PDS 一起配置使用。 0: 停止模式下开启调节器。 1: 停止模式下调节器处于低功耗模式。

### 3.3.3 电源控制状态寄存器 (PWR\_CTRLSTS)

偏移地址: 0x04

复位值: 0x0000 0000 (从 STANDBY 模式唤醒后不会被复位)

与标准 APB 读操作相比, 读此寄存器需要额外的 APB 周期。

31	Reserved															16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved	WKUP3 PS	WKUP2 PS	WKUP1 PS	WKUP RTCCEN	WKUP3 EN	WKUP2 EN	WKUP1 EN	Reserved	WKUP RTCF	WKUPF3	WKUPF2	WKUPF1	PVDO	STBYF	Reserved	
	rw	rw	rw	rw	rw	rw	rw		r	r	r	r	r	r		

位域	名称	描述
31:15	Reserved	保留, 必须保持复位值。
14	WKUP3PS	PC13 唤醒引脚极性选择位。 使用上升沿或下降沿唤醒 STANDBY 模式。确保在更改极性之前对应的唤醒引脚是禁用的。 0: 上升沿 1: 下降沿
13	WKUP2PS	PA0 唤醒引脚极性选择位。 使用上升沿或下降沿唤醒 STANDBY 模式。确保在更改极性之前对应的唤醒引脚是禁用的。 0: 上升沿 1: 下降沿
12	WKUP1PS	PA8 唤醒引脚极性选择位。 使用上升沿或下降沿唤醒 STANDBY 模式。确保在更改极性之前对应的唤醒引脚是禁用的。 0: 上升沿 1: 下降沿
11	WKUPRTCCEN	RTC 内部唤醒使能 0: 禁用唤醒 1: 启用唤醒
10	WKUP3EN	启用 PC13_WKUP 引脚 软件可以设置和清除该位。

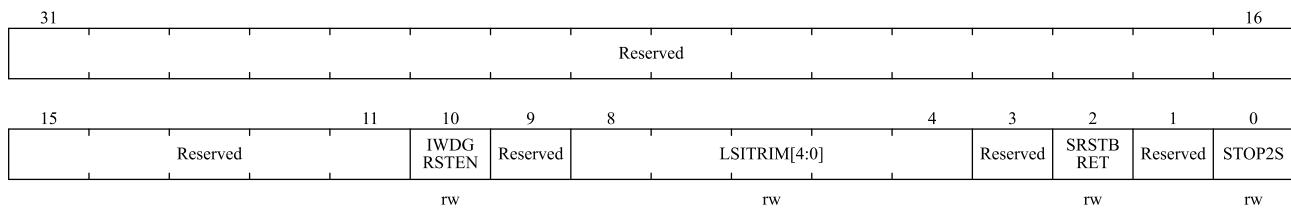
位域	名称	描述
		<p>0: WKUP 引脚用于通用 I/O。WKUP 引脚上的事件不会将器件从 STANDBY 模式唤醒。</p> <p>1: WKUP 引脚用于从 STANDBY 模式唤醒。</p> <p>注意: 该位还可以由系统复位清零。</p>
9	WKUP2EN	<p>启用 PA0_WKUP 引脚 软件可以设置和清除该位。</p> <p>0: WKUP 引脚用于通用 I/O。WKUP 引脚上的事件不会将器件从 STANDBY 模式唤醒。</p> <p>1: WKUP 引脚用于从 STANDBY 模式唤醒。</p> <p>注意: 该位还可以由系统复位清零。</p>
8	WKUP1EN	<p>启用 PA8_WKUP 引脚 软件可以设置和清除该位。</p> <p>0: WKUP 引脚用于通用 I/O。WKUP 引脚上的事件不会将器件从 STANDBY 模式唤醒。</p> <p>1: WKUP 引脚用于从 STANDBY 模式唤醒。</p> <p>注意: 该位还可以由系统复位清零。</p>
7	Reserved	保留, 必须保持复位值。
6	WKUPRTCF	<p>RTC 唤醒标志 该位由硬件设置。可以通过硬件、系统复位或设置 PWR_CTRL.CLRRTCWKUP 清除。</p> <p>0: 未发生唤醒事件 1: 从 RTC 接收到唤醒事件</p>
5	WKUPF3	<p>PC13_WKUP 引脚唤醒标志 该位由硬件设置。可以通过硬件、系统复位或设置 PWR_CTRL.CLRWKUP3 清除。</p> <p>0: 未发生唤醒事件 1: 从 WKUP 引脚接收到唤醒事件</p> <p>注意: 如果在 WKUP 引脚电平已经为高电平时启用 WKUP 引脚 (通过设置 PWR_CTRLSTS.WKUP3EN), 则会检测到额外的唤醒事件。</p>
4	WKUPF2	<p>PA0_WKUP 引脚唤醒标志 该位由硬件设置。可以通过硬件、系统复位或设置 PWR_CTRL.CLRWKUP2 清除。</p> <p>0: 未发生唤醒事件 1: 从 WKUP 引脚接收到唤醒事件</p> <p>注意: 如果在 WKUP 引脚电平已经为高电平时启用 WKUP 引脚 (通过设置 PWR_CTRLSTS.WKUP2EN), 则会检测到额外的唤醒事件。</p>
3	WKUPF1	<p>PA8_WKUP 引脚唤醒标志 该位由硬件设置。可以通过硬件、系统复位或设置 PWR_CTRL.CLRWKUP1 清除。</p> <p>0: 未发生唤醒事件 1: 从 WKUP 引脚接收到唤醒事件</p> <p>注意: 如果在 WKUP 引脚电平已经为高电平时启用 WKUP 引脚 (通过设置</p>

位域	名称	描述
		PWR_CTRLSTS.WKUP1EN), 则会检测到额外的唤醒事件。
2	PVDO	<p>PVD 输出。 该位由硬件设置和清除。仅当 PWR_CTRL.PVDEN = 1 时才有效。 0: VDD/VDDA 高于使用 PWR_CTRL.MSB 和 PWR_CTRL.PLS[2:0]选择的 PVD 阈值。 1: VDD/VDDA 低于使用 PWR_CTRL.MSB 和 PWR_CTRL.PLS[2:0]选择的 PVD 阈值。 注意: PVD 在 STANDBY 模式下停止。因此, 该位在 STANDBY 或复位后等于 0, 直到 PWR_CTRL.PVDEN 置位。</p>
1	STBYF	<p>STANDBY 模式标志。 该位由硬件设置。只能通过 POR/PDR (上电复位/掉电复位) 或设置 PWR_CTRL.CLRSTBF 来清除。 0: 设备未进入过 STANDBY 模式 1: 设备进入过 STANDBY 模式</p>
0	Reserved	保留, 必须保持复位值。

### 3.3.4 电源控制寄存器 2 (PWR\_CTRL2)

偏移地址: 0x08

复位值: 0x0000 04E4 (从 STANDBY 模式唤醒后复位)



位域	名称	描述
31:11	Reserved	保留, 必须保持复位值。
10	IWDGRSTEN	<p>独立看门狗复位使能。 0: IWDG 不能对 RCC 产生复位。 1: IWDG 可以对 RCC 产生复位。</p>
9	Reserved	保留, 必须保持复位值。
8:4	LSITRIM[4:0]	LSI 修正值。
3	Reserved	保留, 必须保持复位值。
2	SRSTBRET	<p>SRAM STANDBY 模式保持。 0: SRAM 在 STANDBY 模式下不保持。 1: SRAM 在 STANDBY 模式下保持。</p>
1	Reserved	保留, 必须保持复位值。
0	STOP2S	<p>STOP2 模式控制。 0: 无效果 1: 芯片进入 STOP2 模式。</p>

## 4 复位和时钟控制(RCC)

### 4.1 概述

本系统中的时钟和复位控制模块负责以下功能：

- CGU：位于 VDDD 电源域，负责控制系统时钟选择、外设时钟使能控制和时钟分频功能。
- RCU：位于 VDDD 电源域，负责系统复位控制和外设软件复位。
- RCC\_BKP：位于 VDDDBK 电源域，负责备份域复位和时钟控制以及 RTC 的时钟源选择。
- RCC\_RET：位于 VDDDRRET 电源域，负责 LPTIMER 时钟的 Retention 域复位控制和时钟源选择。

### 4.2 复位控制单元

N32G430 支持以下三种复位方式：

- 电源复位
- 系统复位
- 备份域复位

#### 4.2.1 电源复位

在以下情况下会发生电源重置：

- 上电/掉电复位 (POR/PDR 复位)。
- 退出 STANDBY/STOP2 模式时复位。

电源复位会将所有寄存器设置为其复位值，但备份域中的寄存器除外（参见图 3-1）。

#### 4.2.2 系统复位

除了控制/状态寄存器(RCC\_CTRLSTS)中的复位标志和备份域中的寄存器（参见图 3-1），系统复位会将所有寄存器设置为其复位值。

发生以下事件之一时会产生系统复位：

- NRST管脚上的低电平（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 (SW复位)
- 低功耗管理复位
- 电源复位
- MMU保护复位

- 备份域EMC复位

可以通过检查控制/状态寄存器(RCC\_CTRLSTS)中的复位标志来识别复位源。

#### 4.2.2.1 软件复位

可以通过设置 Cortex™-M4 应用中断和复位控制寄存器中的 SYSRESETREQ 位来产生软件复位。有关详细信息，请参阅 Cortex™-M4 技术参考手册。

#### 4.2.2.2 低功耗管理复位

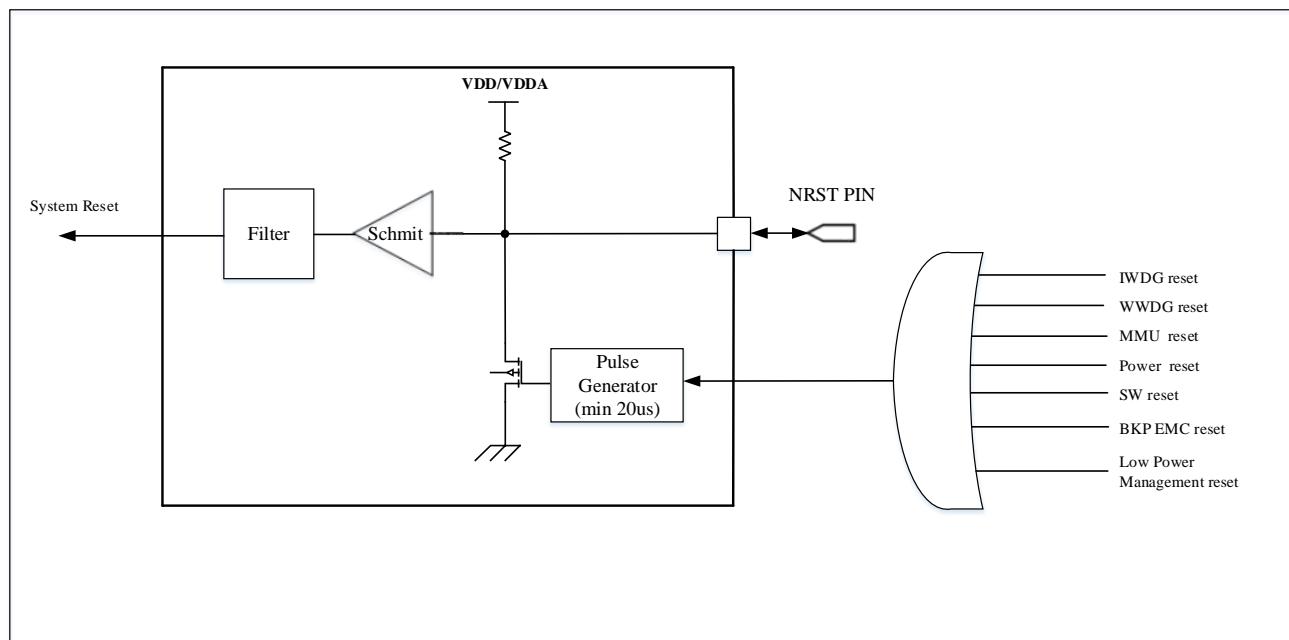
可以通过以下方式产生低功耗管理复位：

- 进入 STANDBY 模式时产生的低功耗管理复位：使能用户选项字节中的 nRST\_STDBY 位将使能该复位。在这种情况下，即使执行了进入 STANDBY 模式的流程，系统会复位而不是进入 STANDBY 模式。
- 进入 STOP0/STOP2 模式时产生的低功耗管理复位：使能用户选项字节中的 nRST\_STOP 位将使能该复位。在这种情况下，即使执行了进入 STOP0/STOP2 模式的流程，系统就会复位而不是进入 STOP0/STOP2 模式。

提供给芯片的系统复位信号会在 NRST 引脚上输出。脉冲发生器保证每个复位源（外部或内部）的复位脉冲至少持续时间 20μs。对于外部复位，当 NRST 引脚置为低电平时会产生复位脉冲。

下图展示了复位电路：

图 4-1 复位电路



#### 4.2.3 备份域复位

备份域有两个仅影响备份域的专用复位（参见图 3-1）。

发生以下事件之一时会产生备份域复位：

- 软件复位：通过设置 RCC\_BDCTRL.BDSFTRST 位可以产生备份域复位。

- 电源复位：在 VDDDBK 初始上电阶段产生备份域复位。

## 4.3 时钟控制单元

可以使用三种不同的时钟源来驱动系统时钟(SYSCLK)：

- HSI 振荡器时钟
- HSE 振荡器时钟
- PLL 时钟

有两种二级时钟源：

- LSI: 40kHz 低速内部 RC，可用于驱动独立看门狗 (IWDG) 或通过软件选择驱动 RTC。RTC 可用于从 SLEEP/STOP0/STOP2/STANDBY 模式自动唤醒。
- LSE: 32.768kHz 低速外部晶体，可通过软件选择驱动 RTC。

每个时钟源可以在不被使用时独立打开或关闭，以此优化系统功耗。

多个预分频器可用于配置 AHB、高速 APB(APB2)和低速 APB(APB1)的频率。AHB、APB2 和 APB1 的最大频率分别为 128MHz、64MHz 和 32MHz。

RCC 为 Cortex 系统定时器(SysTick)提供外部时钟：AHB 时钟(HCLK)8 分频。可以通过编程 SysTick 控制和状态寄存器来选择上述时钟或 Cortex 时钟(HCLK)来驱动 SysTick。ADC 时钟由 AHB 时钟或 PLL 时钟分频产生。

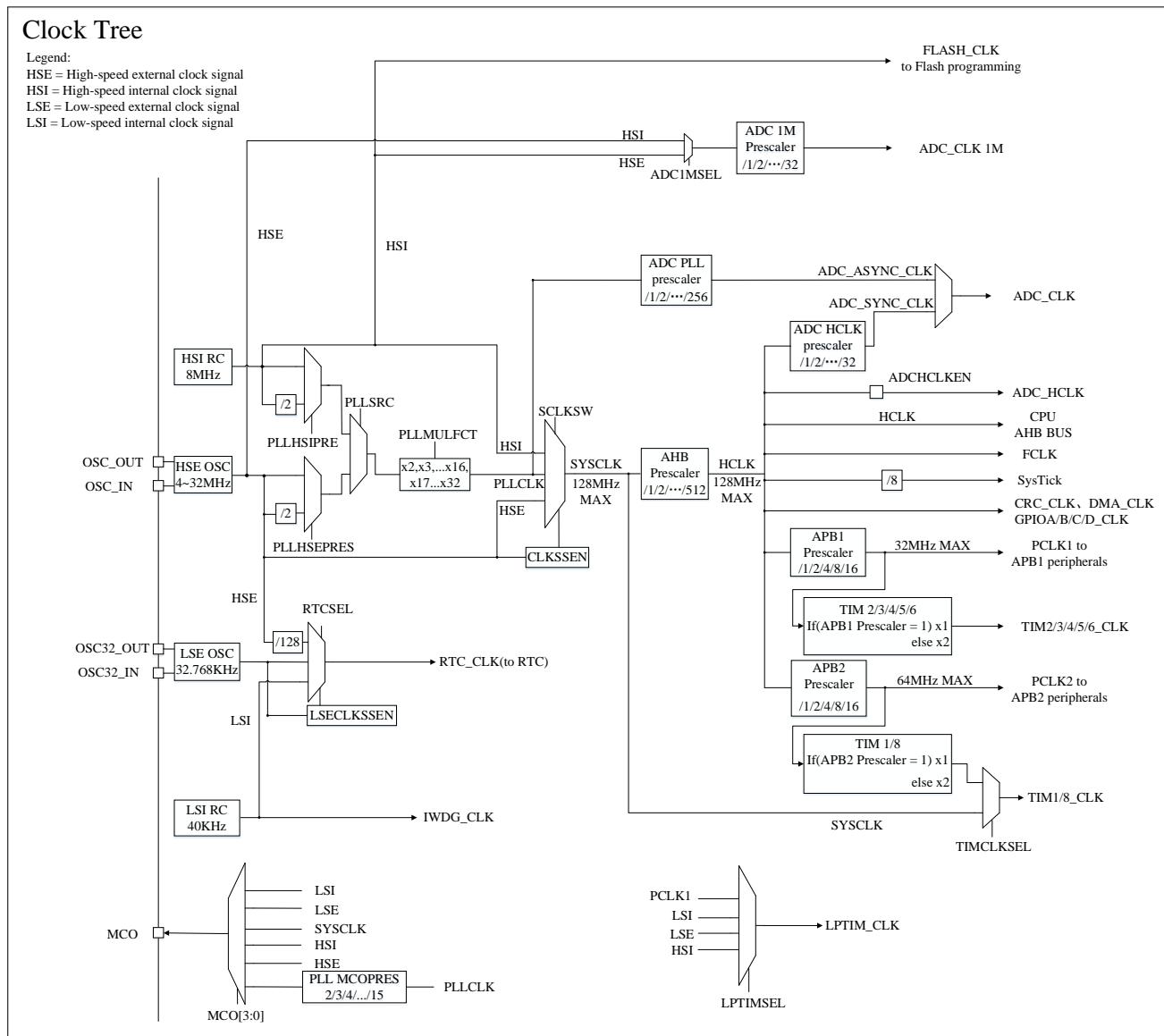
定时器的时钟频率由硬件自动设置，分以下两种情况：

- 如果 APB 预分频为 1，则定时器时钟频率与定时器所在的 APB 频率相同。
- 如果 APB 预分频不为 1，则定时器时钟频率是定时器所在的 APB 频率的 2 倍。

FCLK 是 Cortex<sup>TM</sup>-M4F 的自由运行时钟。有关更多详细信息，请参阅 ARM Cortex<sup>TM</sup>-M4 技术参考手册。

### 4.3.1 时钟树

图 4-2 时钟树



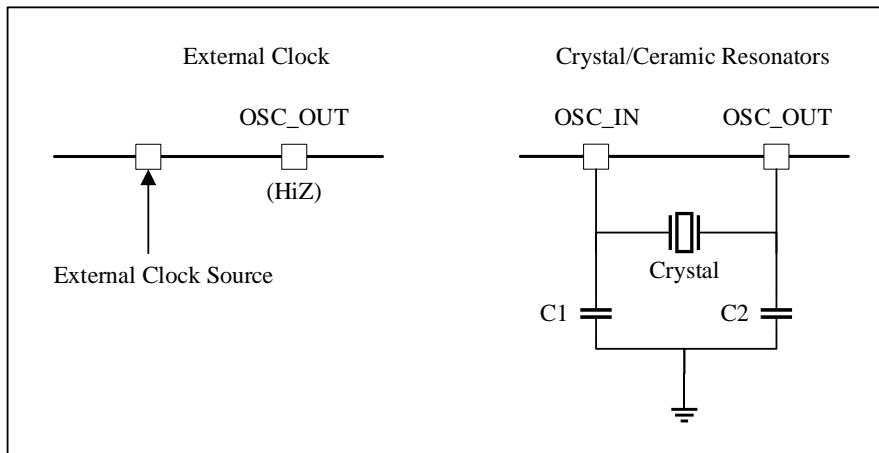
1. 系统时钟的最大频率为 128MHz。
2. 有关内部和外部时钟源特性的详细信息，请参阅产品数据手册中的“电气特性”部分。
3. PLL 作为系统时钟源时，最小系统时钟输出为 32MHz。

### 4.3.2 HSE 时钟

高速外部时钟信号（HSE）可以由以下两个时钟源产生：

- HSE外部晶体/陶瓷谐振器
- HSE用户外部时钟

图 4-3 HSE 时钟源



#### 4.3.2.1 外部时钟源(HSE 旁路模式)

在这种模式下，用户必须提供外部时钟源。它的频率最高可达 32MHz。用户可以通过设置 RCC\_CTRL.HSEBP 和 RCC\_CTRL.HSEEN 位来选择该模式。外部时钟信号必须连接到 OSC\_IN 引脚，而 OSC\_OUT 引脚必须悬空(Hi-Z)。见图 4-3。

#### 4.3.2.2 晶体/陶瓷谐振器(HSE 晶体模式)

4~32MHz 外部振荡器具有为系统产生更准确的主时钟的优势。相关的硬件配置如图 4-3 所示。更多详细信息，请参阅数据手册的电气特性部分。

RCC\_CTRL.HSERDF 位指示高速外部振荡器是否稳定。在启动时，直到该位被硬件设置，时钟才会被释放。如果在时钟中断寄存器(RCC\_CLKINT)中使能对应位，则可以产生中断。

通过设置 RCC\_CTRL.HSEEN 位可以打开和关闭 HSE 时钟。

### 4.3.3 HSI 时钟

HSI (高速内部) 时钟信号由内部 8MHz RC 振荡器产生，可直接用作系统时钟。HSI RC 振荡器无需任何外部设备即可提供时钟源。它启动时间比 HSE 晶体振荡器更短。然而，即使经过校准它的频率精度仍较差。

由于制造工艺的变化，RC 振荡器频率精度可能因芯片而异。系统复位后，出厂校准值会加载到 RCC\_CTRL.HSICAL[7:0]里。

由于用户的应用场景会受到电压或温度变化的影响，这也会影响 RC 振荡器的频率精度。用户可以使用 RCC\_CTRL.HSITRIM[4:0]位调整 HSI 频率。

RCC\_CTRL.HSIRDF 位指示内部 RC 振荡器是否稳定。在启动时，直到该位被硬件设置，HSI 输出时钟才会被释放。可以通过设置 RCC\_CTRL.HSIEN 位打开和关闭 HSI 时钟。

如果 HSE 晶振出现故障，HSI 时钟可以作为备用源。请参阅 4.3.8 时钟安全系统(CLKSS)。

### 4.3.4 PLL 时钟

内部 PLL 可用于倍频 HSI 或 HSE 时钟频率，参考图 4-2 时钟树。必须在使能 PLL 之前完成配置（选择 PLL 输入时钟（HSI/HSE 及分频）和配置倍频因子）。一旦 PLL 被使能，这些参数就不能改变。通过 RCC\_CTRL 和 RCC\_CFG 寄存器中的控制位来配置 PLL。

如果在时钟中断寄存器(RCC\_CLKINT)中使能 PLL 中断，则可以在 PLL 就绪时产生中断。

### 4.3.5 LSE 时钟

以下两个时钟源可生成低速外部时钟信号(LSE):

- LSE 晶体/陶瓷谐振器
- LSE 外部时钟(旁路)

#### 4.3.5.1 LSE 晶体时钟源

在这种模式下，LSE 晶体是一个 32.768kHz 低速外部晶体/陶瓷谐振器。它为 RTC 或其他计时功能提供一个功耗低且精确的时钟源。

通过设置 RCC\_BDCTRL.LSEEN 位可以打开和关闭 LSE 时钟。

RCC\_BDCTRL.LSERD 位指示 LSE 时钟是否稳定。启动时，直到该位被硬件设置，LSE 输出时钟才会被释放。如果在时钟中断寄存器(RCC\_CLKINT)中使能对应位，则可以产生中断。

#### 4.3.5.2 LSE 外部时钟源

在这种模式下，可以提供频率高达 1MHz 的外部时钟源。用户可以通过设置 RCC\_BDCTRL.LSEBP 和 RCC\_BDCTRL.LSEEN 位来选择该模式。占空比为 50% 的外部时钟信号必须连接到 OSC32\_IN 引脚，而 OSC32\_OUT 引脚必须悬空 (Hi-Z)。

### 4.3.6 LSI 时钟

LSI RC 振荡器提供频率约为 40kHz 的低速内部(LSI)时钟。通过 RCC\_CTRLSTS.LSIEN 位打开或关闭 LSI 时钟。

RCC\_CTRLSTS.LSIRD 位指示 LSI 时钟是否稳定。在启动时，直到该位被硬件设置，时钟才会被释放。如果在时钟中断寄存器(RCC\_CLKINT)中使能对应位，则可以产生中断。

#### 4.3.6.1 LSI 校准

可以通过校准补偿内部低速振荡器 LSI 频率误差，以获得更高精度的 RTC 时基和 IWDG 超时（当这些外设由 LSI 提供时钟时）。

校准可以通过使用 TIM2 的时钟(TIM2\_CLK)测量 LSI 时钟频率来实现。测量由 HSE 的准确性保证，软件可以通过调整 RTC 的预分频器来获得准确的 RTC 时钟基准，以及通过计算获得准确的 IWDG 超时时间。

LSI 校准步骤如下：

1. 打开 TIM2，设置通道 3 为输入捕捉模式；
2. 设置 TIM2\_CTRL1.C3SEL 位为 1，将 LSI 内部连接到 TIM2 的通道 3；
3. 通过 TIM2 捕获/比较 3 事件或中断测量 LSI 时钟频率；
4. 根据测量结果和所需的 RTC 时基和独立的看门狗超时，设置预分频器。

### 4.3.7 系统时钟(SYSCLK)选择

系统复位后，选择 HSI 振荡器作为系统时钟。当时钟源直接或通过 PLL 间接用作系统时钟时，HSI 无法被

停止。

仅当目标时钟源准备好（在启动延迟或 PLL 锁定之后）时，才能从一个时钟源切换到另一个时钟源。当所选时钟源未准备好时，不会发生系统时钟的切换。

RCC\_CFG.SCLKSW[1:0]用于选择系统时钟源。时钟配置寄存器(RCC\_CFG)和时钟控制寄存器(RCC\_CTRL)中的状态位指示哪个时钟已准备好以及哪个时钟当前用作系统时钟。

### 4.3.8 时钟安全系统(CLKSS)

时钟安全系统可以通过软件通过设置 RCC\_CTRL.CLKSSEN 位来激活。一旦被激活，时钟检测器在 HSE 振荡器的启动延时后被启用，并在 HSE 时钟关闭时被禁用。

如果 HSE 时钟出现故障，HSE 振荡器将自动关闭，时钟失效事件将发送到高级定时器（TIM1/TIM8）的刹车输入，并产生时钟安全系统中断 CLKSSIF，允许软件执行营救措施。CLKSSIF 中断连接到 Cortex<sup>TM</sup>-M4 的 NMI（不可屏蔽）中断。

NMI 将连续执行，直到 CLKSSIF 中断挂起位被清除。因此，需要通过在 NMI 处理程序中设置 RCC\_CLKINT.CLKSSICLR 位来清除中断。

如果 HSE 振荡器直接或间接用作系统时钟（间接的意思是：HSE 用作 PLL 输入时钟，PLL 时钟用作系统时钟），时钟失效会导致系统时钟切换到 HSI 振荡器，并且外部 HSE 振荡器被禁用。如果选择 HSE 时钟（分频或不分频）作为 PLL 输入时钟，那么当 HSE 时钟故障时，PLL 将被关闭。

### 4.3.9 LSE 时钟安全系统(LSECSS)

LSE 时钟安全系统通过启用 RCC\_BDCTRL.LSECLKSSEN 位来激活。RCC\_BDCTRL.LSECLKSSEN 位可以通过硬件复位或 RTC 软件复位或在检测到 LSE 故障后清零。当 LSE 和 LSI 使能并准备就绪时，必须在配置 RCC\_BDCTRL.RTCSEL 选择 RTC 时钟源后再使能 RCC\_BDCTRL.LSECLKSSEN 位。

如果检测到 LSE 故障，将不再向 RTC 提供 LSE，但硬件不会修改 RCC\_BDCTRL.RTCSEL 位来切换 RTC 时钟源。

在 LSE 时钟故障时，如果设置了 LSECSS 中断使能位 RCC\_CLKINT.LSESSIEN，则设置中断标志 RCC\_CLKINT.LSESSIF。LSE 故障检测信号还用作 EXTI 中断输入和 PWR 唤醒事件，以使系统能够在 LSE 故障的情况下从低功耗模式唤醒。然后软件可以进行清零 RCC\_BDCTRL.LSECLKSSEN 位，关闭 LSE，改变 RTC 时钟源等措施，保证应用的安全。

LSE 振荡器的频率必须高于 30KHz，以避免误检测 LSECSS。

### 4.3.10 RTC 时钟

通过对 RCC\_BDCTRL.RTCSEL[1:0]位进行编程，RTCCLK 时钟源可以是 HSE/128、LSE 或 LSI 时钟。除非重置备份域，否则无法修改此选择。

在配置 RTC 时钟源之前，必须将 PWR\_CTRL.DBPK 位设置为 1，以取消写保护。

LSE 和 LSI 时钟在备份域中，而 HSE 时钟不在备份域中：

- 如果选择 LSE 或 LSI 作为 RTC 时钟：
  - ◆ 如果 V<sub>DD</sub> 电源被关闭，RTC 将无法继续工作

- 如果 HSE 时钟 128 分频作为 RTC 时钟:
    - ◆ 仅工作在 RUN/SLEEP 模式下

#### 4.3.11 看门狗时钟

如果 IWDG 由硬件选项或软件访问启动，LSI 振荡器将被强制开启并且不能被禁用。LSI 振荡器稳定后，时钟被提供给 IWDG。

#### 4.3.12 时钟输出(MCO)

微控制器时钟输出(MCO)功能允许将时钟信号输出到外部 MCO 引脚。

对应的 GPIO 口寄存器必须配置为对应的功能。可以选择以下 6 个时钟信号作为 MCO 时钟：

- SYSCLK
  - HSI
  - HSE
  - PLL 时钟分频
  - LSI
  - LSE

时钟选择由 RCC\_CFG.MCO[3:0]位控制。

#### 4.4 RCC 寄存器

RCC 寄存器可通过 AHB 总线访问，寄存器说明如下。

#### 4.4.1 寄存器总览

表 4-1 RCC 寄存器总览

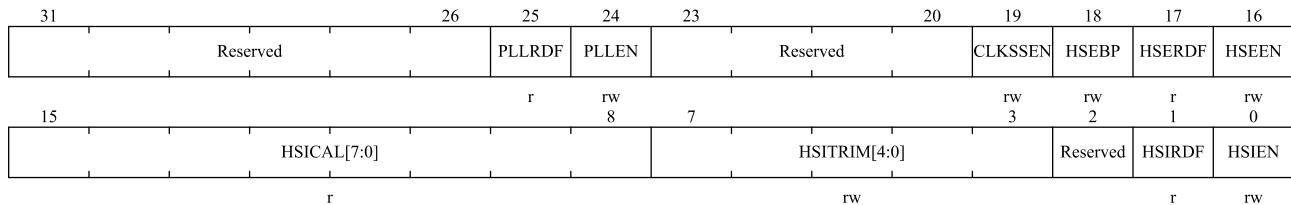
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	RCC_CTRL	Reserved				Reserved				Reserved				HSICAL[7:0]				HSITRIM[4:0]				Reserved												
	Reset Value																																	
	RCC_CFG	MCOPRES[3:0]				MCO[3:0]				PLLMULFCI[4]				PLLMULFCI[3:0]				CLKSSEN				HSERP				HSERDF				HSEEN				
000h		Reserved				Reserved				Reserved				Reserved				HSICAL[7:0]				HSITRIM[4:0]				Reserved								
004h		Reset Value				0	0	1	0	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0	0	PLLIDCLR	0
008h	RCC_CLKINT	Reserved				Reserved				Reserved				Reserved				APB2PRES[2:0]				APB1PRES[2:0]				AHBPRES[3:0]				SCLKSTS[1:0]				
	Reset Value																													Reserved				

00Ch	RCC_APB2PRST	Reserved									
	Reset Value										
010h	RCC_APB1PRST	Reserved									
	Reset Value	0	PWRRST	Reserved	0	CANRST	0	SPI2RST	0	UART4RST	
014h	RCC_AHBCLKEN	Reserved									
	Reset Value	0	PWREN	Reserved	0	I2C2RST	0	UART1RST	0	USART1RST	
018h	RCC_APB2PCLKEN	Reserved									
	Reset Value	0	CANEN	Reserved	0	I2C1EN	0	USART2RST	0	UART3RST	
01Ch	RCC_APB1PCLKEN	Reserved									
	Reset Value	0	I2C2EN	Reserved	0	UART4EN	0	WWDRST	0	TIM1RST	
020h	RCC_BDCTRL	Reserved									
	Reset Value	0	BDSRST	Reserved	0	USART2EN	0	WWDRST	0	TIM8RST	
024h	RCC_CTRLSTS	Reserved									
	Reset Value	0	BKPEMCF	Reserved	0	USART1EN	0	WWDRST	0	SPI1RST	
028h	RCC_AHBP_RST	Reserved									
	Reset Value	0	ADCRST	Reserved	0	TIM9EN	0	WWDRST	0	TIM1RST	
02Ch	RCC_CFG2	Reserved									
	Reset Value	0	ADCIMSEL	Reserved	0	WWDGGEN	0	WWDRST	0	TIM1RST	
034h	RCC_RDCTRL	Reserved									
	Reset Value	0	LPTIMRST	Reserved	0	COMPFLTEN	0	WWDRST	0	TIM1RST	
040h	RCC_PLLHSIPRE	Reserved									
	Reset Value	0	LPTIMSEL[1:0]	Reserved	0	COMPEN	0	WWDRST	0	TIM1RST	
080h	RCC_AHB1CLKEN	Reserved									
	Reset Value	0	LPTIMSEL[2:0]	Reserved	0	CRcen	0	WWDRST	0	AFIORST	

## 4.4.2 时钟控制寄存器(RCC\_CTRL)

偏移地址: 0x00

复位值: 0x0000 7783



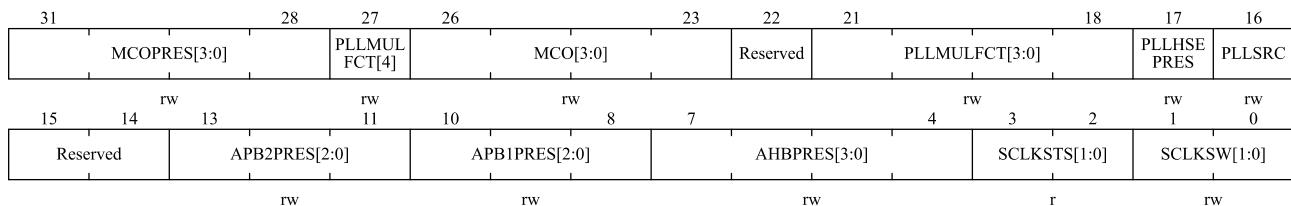
位域	名称	描述
31:26	Reserved	保留, 必须保持复位值
25	PLL RDF	PLL 时钟就绪标志位 PLL 时钟就绪后由硬件置位。 0:PLL 未就绪 1:PLL 已就绪
24	PLLEN	PLL 使能位 由软件置位和清零。进入 STOP0/STOP2 或 STANDBY 模式时, 由硬件清零。 当 PLL 用作系统时钟时, 该位不能清零。 0: 禁用 PLL 1: 使能 PLL
23:20	Reserved	保留, 必须保持复位值
19	CLKSSEN	时钟安全系统使能位 由软件置位和清零。 0: 禁用时钟检测器 1: 如果 HSE 振荡器就绪, 则使能时钟检测器
18	HSEBP	外部高速时钟旁路使能位 由软件置位和清零。该位只能在 HSE 振荡器被禁止时写入。 0: 禁止 HSE 振荡器的旁路功能 1: 使能 HSE 振荡器的旁路功能
17	HSERDF	外部高速时钟就绪标志位 HSE 就绪后由硬件置位。该位在 HSEEN 位清零后 6 个 HSE 时钟周期来清零。 0: HSE 未就绪 1: HSE 就绪
16	HSEEN	外部高速时钟使能位 由软件置位和清零。进入 STOP0/STOP2 或 STANDBY 模式时, 由硬件清零。 当 HSE 直接或间接用作系统时钟时, 该位不能被清零。 0: 禁止 HSE 振荡器 1: 使能 HSE 振荡器
15:8	HSICAL[7:0]	内部高速时钟校准值 这些位在上电启动时自动初始化。

位域	名称	描述
7:3	HSITRIM[4:0]	内部高速时钟修正值 由软件写入。这些位的值将被添加到 HSICAL[7:0]位，以形成校准内部 HSI RC 振荡器频率的最终值。两个连续 HSICAL 步进之间的微调步进约为 10kHz，默认值为 16，可将 HSI 调整为 8MHz±1%。
2	Reserved	保留，必须保持复位值
1	HSIRDF	内部高速时钟就绪标志位 HSI 就绪后由硬件置位。HSIEN 位清零后，该位需要 6 个内部 8MHz 振荡器时钟周期才能清零。 0:HSI 未就绪 1:HSI 就绪
0	HSIEN	内部高速时钟使能 由软件置位和清零。当 HSI 用作系统时钟时，该位不能清零。当从 STOP0/STOP2 或 STANDBY 模式返回或发生 HSE 故障时，由硬件置位以启用 HSI 振荡器。如果 HSI 直接或间接用作系统时钟，则该位不能复位。 0: 禁止 HSI 振荡器 1: 使能 HSI 振荡器

#### 4.4.3 时钟配置寄存器(RCC\_CFG)

偏移地址: 0x04

复位值: 0x2000 0000



位域	名称	描述
31:28	MCOPRES[3:0]	MCO 预分频 由软件置位和清零。配置 PLL 时钟的分频作为 MCO 时钟。 0010:PLL 时钟 2 分频 0011:PLL 时钟 3 分频 0100:PLL 时钟 4 分频 0101:PLL 时钟 5 分频 0110:PLL 时钟 6 分频 0111:PLL 时钟 7 分频 1000:PLL 时钟 8 分频 1001:PLL 时钟 9 分频 1010:PLL 时钟 10 分频 1011:PLL 时钟 11 分频

位域	名称	描述
		1100:PLL 时钟 12 分频 1101:PLL 时钟 13 分频 1110:PLL 时钟 14 分频 1111:PLL 时钟 15 分频 其他值：不允许设置
27	PLLMULFCT[4]	该位与位[21:18]组合形成 PLL 倍频因子。描述请参考 PLLMULFCT[3:0]。
26:23	MCO[3:0]	微控制器时钟输出选择 由软件置位和清零。 0xxx: 无时钟输出 1000: 选择系统时钟(SYSCLK) 1001: 选择 HSI 时钟 1010: 选择 HSE 时钟 1011: 选择 PLL 分频时钟 (MCOPRES 定义分频系数) 1100: 选择 LSI 时钟 1101: 选择 LSE 时钟 <p style="margin-left: 2em;">注意：此时钟输出可能在启动时或 MCO 时钟源切换期间被截断。当系统时钟输出到 MCO 引脚时，应保证输出时钟频率不超过 I/O 口的最大频率 (I/O 口的最大频率详见数据手册)</p>
22	Reserved	保留，必须保持复位值
21:18	PLLMULFCT[3:0]	PLL 倍频系数(包括 bit27) 倍频系数由软件写入。这些位只能在 PLL 被禁用时写入。PLL 输出频率不得超过 128MHz。 00000:PLL 输入时钟 ×2 00001:PLL 输入时钟 ×3 00010:PLL 输入时钟 ×4 00011:PLL 输入时钟 ×5 00100:PLL 输入时钟 ×6 00101:PLL 输入时钟 ×7 00110:PLL 输入时钟 ×8 00111:PLL 输入时钟 ×9 01000:PLL 输入时钟 ×10 01001:PLL 输入时钟 ×11 01010:PLL 输入时钟 ×12 01011:PLL 输入时钟 ×13 01100:PLL 输入时钟 ×14 01101:PLL 输入时钟 ×15 01110:PLL 输入时钟 ×16 01111:PLL 输入时钟 ×16 10000:PLL 输入时钟 ×17 10001:PLL 输入时钟 ×18 10010:PLL 输入时钟 ×19 10011:PLL 输入时钟 ×20

位域	名称	描述
		10100:PLL 输入时钟 ×21 10101:PLL 输入时钟 ×22 10110:PLL 输入时钟 ×23 10111:PLL 输入时钟 ×24 11000:PLL 输入时钟 ×25 11001:PLL 输入时钟 ×26 11010:PLL 输入时钟 ×27 11011:PLL 输入时钟 ×28 11100:PLL 输入时钟 ×29 11101:PLL 输入时钟 ×30 11110:PLL 输入时钟 ×31 11111:PLL 输入时钟 ×32
17	PLLHSEPRE	PLL 输入的 HSE 预分频器 由软件置位和清零，配置进入 PLL 之前 HSE 的分频。该位只能在 PLL 禁用时写入。 0: HSE 时钟不分频 1: HSE 时钟 2 分频
16	PLLSRC	PLL 时钟源 由软件置位和清零，配置选择 PLL 时钟源。该位只能在 PLL 禁用时写入。 0: 选择 HSI 时钟（分频或不分频）作为 PLL 输入时钟 1: 选择 HSE 时钟（分频或不分频）作为 PLL 输入时钟
15:14	Reserved	保留，必须保持复位值
13:11	APB2PRES[2:0]	APB 高速(APB2)预分频器 由软件置位和清零，配置 APB2 时钟(PCLK2)的分频系数。需确保 PCLK2 不超过 64MHz。 0xx:HCLK 不分频 100:HCLK 2 分频 101:HCLK 4 分频 110:HCLK 8 分频 111:HCLK 16 分频
10:8	APB1PRES[2:0]	APB 低速(APB1)预分频器 由软件置位和清零，配置 APB1 时钟(PCLK1)的分频系数。需确保 PCLK1 不超过 32MHz。 0xx:HCLK 不分频 100:HCLK 2 分频 101:HCLK 4 分频 110:HCLK 8 分频 111:HCLK 16 分频
7:4	AHBPRES[3:0]	AHB 预分频器 由软件置位和清零，配置 AHB 时钟(HCLK)的分频系数。 0xxx:SYSCLK 不分频 1000:SYSCLK 2 分频

位域	名称	描述
		1001:SYSCLK 4 分频 1010:SYSCLK 8 分频 1011:SYSCLK 16 分频 1100:SYSCLK 64 分频 1101:SYSCLK 128 分频 1110:SYSCLK 256 分频 1111:SYSCLK 512 分频
3:2	SCLKSTS[1:0]	系统时钟切换状态 由硬件置位和清零以指示使用哪个时钟源作为系统时钟 00: HSI 振荡器作为系统时钟 01: HSE 振荡器作为系统时钟 10: PLL 作为系统时钟 11: 不可用
1:0	SCLKSW[1:0]	系统时钟切换 由软件置位和清零以选择系统时钟源。 当退出 STOP0/STOP2 或 STANDBY 模式或 HSE 振荡器发生故障 (RCC_CTRL.CLKSSEN 启用) 时, 由硬件设置以强制选择 HSI。 00: 选择 HSI 作为系统时钟 01: 选择 HSE 作为系统时钟。 10: 选择 PLL 作为系统时钟 11: 不允许

#### 4.4.4 时钟中断寄存器(RCC\_CLKINT)

偏移地址: 0x08

复位值: 0x0000 0000

31	Reserved	27	LSESS ICLR	26	LSESS IEN	25	LSESSIF	24	CLKSS ICLR	23	Reserved	22	21	20	PLL RD ICLR	HSE RD ICLR	HSI RD ICLR	LSER RD ICLR	LSI RD ICLR	16
15	13	12	11	w 10	rw 9	r 8	w 7	w 6	w 5	w 4	w 3	w 2	w 1	w 0						
	Reserved	PLL RD IEN	HSE RD IEN	HSI RD IEN	LSER RD IEN	LSI RD IEN	CLKSSIF	Reserved	PLL RDIF	HSE RDIF	HSI RDIF	LSER RDIF	LSI RDIF							
		rw	rw	rw	rw	rw	r		r	r	r	r	r						r	

位域	名称	描述
31:27	Reserved	保留, 必须保持复位值
26	LSESSICLR	LSE 时钟安全系统中断清除 该位由软件设置以清除 LSESSIF 标志。 0: 无效 1: 清除 LSESSIF 标志

位域	名称	描述
25	LSESSIEN	LSE 时钟安全系统中断使能 由软件设置和清除以启用/禁用由 LSECSS 检测引起的中断。 0: 禁用 LSECSS 中断 1: 启用 LSECSS 中断
24	LSESSIF	LSE 时钟安全系统中断标志 当在 LSE 中检测到故障时由硬件设置。通过软件设置 LSESSICLR 位清零。 0: 无 LSE 时钟故障引起的时钟安全中断 1: LSE 时钟故障引起的时钟安全中断
23	CLKSSICLR	时钟安全系统中断清除 由软件置位以清除 CLKSSIF 标志。 0: 无效果 1: 清除 CLKSSIF 标志
22:21	Reserved	保留, 必须保持复位值
20	PLLRDICLR	PLL 就绪中断清除 由软件置位以清除 PLLRDIF 标志。 0: 无效果 1: 清除 PLLRDIF 标志
19	HSERDICLR	HSE 就绪中断清除 由软件置位以清除 HSERDIF 标志。 0: 无效果 1: 清除 HSERDIF 标志
18	HSIRDICLR	HSI 就绪中断清除 由软件置位以清除 HSIRDIF 标志。 0: 无效果 1: 清除 HSIRDIF 标志
17	LSERDICLR	LSE 就绪中断清除 由软件置位以清除 LSERDIF 标志。 0: 无效果 1: 清除 LSERDIF 标志
16	LSIRDICLR	LSI 就绪中断清除 由软件置位以清除 LSIRDIF 标志。 0: 无效果 1: 清除 LSIRDIF 标志
15:13	Reserved	保留, 必须保持复位值
12	PLLRDIEN	PLL 就绪中断使能 由软件置位和清零以启用和禁用 PLL 就绪中断 0: 禁用 PLL 就绪中断 1: 使能 PLL 就绪中断
11	HSERDIEN	HSE 就绪中断使能 由软件置位和清零以启用和禁用 HSE 就绪中断。 0: 禁用 HSE 就绪中断 1: 使能 HSE 就绪中断

位域	名称	描述
10	HSIRDIEN	HSI 就绪中断使能 由软件置位和清零以启用和禁用 HSI 就绪中断。 0: 禁止 HSI 就绪中断 1: 使能 HSI 就绪中断
9	LSERDIE	LSE 就绪中断使能 由软件置位和清零以启用和禁用 LSE 就绪中断。 0: 禁用 LSE 就绪中断 1: 使能 LSE 就绪中断
8	LSIRDIE	LSI 就绪中断使能 由软件置位和清零以启用和禁用 LSI 就绪中断。 0: 禁用 LSI 就绪中断 1: 使能 LSI 就绪中断
7	CLKSSIF	时钟安全系统中断标志 当在外部 HSE 振荡器中检测到故障时由硬件置位。 0: 无 HSE 时钟故障引起的时钟安全系统中断 1: HSE 时钟故障引起的时钟安全系统中断
6:5	Reserved	保留, 必须保持复位值
4	PLL RDIF	PLL 就绪中断标志 当 PLLRDIE 置位且 PLL 时钟准备好时, 该位由硬件置位。 该位由软件通过设置 PLLRDICLR 位来清除。 0: 无由 PLL 锁定引起的时钟就绪中断 1: 由 PLL 锁定引起的时钟就绪中断
3	HSE RDIF	HSE 就绪中断标志 当 HSERDIE 置位且 HSE 时钟准备好时由硬件置位。 该位由软件通过设置 HSERDICLR 位来清除。 0: 无由 HSE 振荡器引起的时钟就绪中断 1: HSE 振荡器引起的时钟就绪中断
2	HSIRDIF	HSI 就绪中断标志 当 HSIRDIE 置位且 HSI 时钟准备好时由硬件置位。 该位由软件通过设置 HSERDICLR 位来清除。 0: 无由 HSI 振荡器引起的时钟就绪中断 1: 由 HSI 振荡器引起的时钟就绪中断
1	LSERDIF	LSE 就绪中断标志 当 LSERDIE 置位且 LSE 时钟准备好时由硬件置位。 该位由软件通过设置 LSERDICLR 位来清除。 0: 无由 LSE 振荡器引起的时钟就绪中断 1: 由 LSE 振荡器引起的时钟就绪中断
0	LSIRDIF	LSI 就绪中断标志 当 LSIRDIE 置位且 LSI 时钟就绪时由硬件置位。 该位由软件通过设置 LSIRDICLR 位来清除。 0: 无由 LSI 振荡器引起的时钟就绪中断 1: LSI 振荡器引起的时钟就绪中断

#### 4.4.5 APB2 外设复位寄存器(RCC\_APB2PRST)

偏移地址: 0x0c

复位值: 0x0000 0000

31	Reserved										20	19	18	17	16
											SPI2RST	UART4 RST	UART3 RST	Reserved	
15	14	13	12	11	10						rw	rw	rw	rw	0
Reserved	USART1 RST	TIM8RST	SPI1RST	TIM1RST							BEEPRST	AFIORST			
	rw	rw	rw	rw							rw	rw			

位域	名称	描述
31:20	Reserved	保留, 必须保持复位值
19	SPI2RST	SPI2 复位 软件置位和清零. 0: 清除复位 1: 复位 SPI2
18	UART4RST	UART4 复位 软件置位和清零. 0: 清除复位 1: 复位 UART4
17	UART3RST	UART3 复位 软件置位和清零. 0: 清除复位 1: 复位 UART3
16:15	Reserved	保留, 必须保持复位值
14	USART1RST	USART1 复位 软件置位和清零. 0: 清除复位 1: 复位 USART1
13	TIM8RST	TIM8 复位 软件置位和清零. 0: 清除复位 1: 复位 TIM8
12	SPI1RST	SPI1 复位 软件置位和清零. 0: 清除复位 1: 复位 SPI1
11	TIM1RST	TIM1 复位 软件置位和清零.

位域	名称	描述
		0: 清除复位 1: 复位 TIM1
10:2	Reserved	保留, 必须保持复位值
1	BEEPRST	BEEPER 复位 软件置位和清零. 0: 清除复位 1: 复位 BEEPER
0	AFIORST	AFIO 复位 软件置位和清零. 0: 清除复位 1: 复位 AFIO

#### 4.4.6 APB1 外设复位寄存器(RCC\_APB1PRST)

偏移地址: 0x10

复位值: 0x0000 0000

31	29	28	27	26	25	24	23	22	21	20	18	17	16
		PWRRST	Reserved	CANRST	Reserved	I2C2RST	I2C1RST		Reserved		USART2 RST	Reserved	
15		rw	12	11	10	rw	7	rw	6	rw	5	4	rw
		Reserved	WWDG RST		Reserved		COMP RST	Reserved	TIM6RST	TIM5RST	TIM4RST	TIM3RST	TIM2RST
		rw				rw		rw		rw		rw	rw

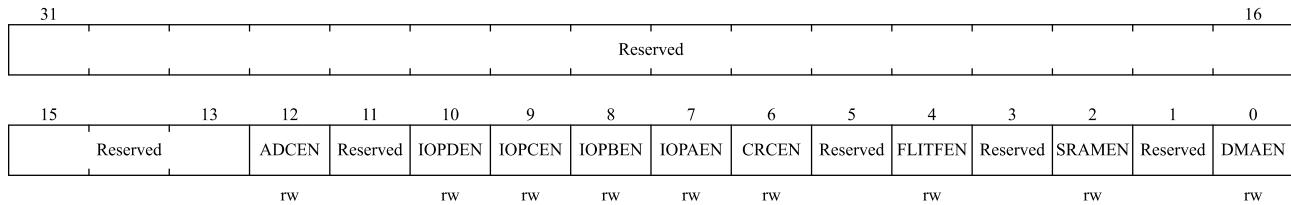
位域	名称	描述
31:29	Reserved	保留, 必须保持复位值
28	PWRRST	电源接口复位 软件置位和清零. 0:清除复位 1:复位电源接口
27:26	Reserved	Reserved
25	CANRST	CAN 复位 软件置位和清零. 0:清除复位 1:复位 CAN
24:23	Reserved	保留, 必须保持复位值
22	I2C2RST	I2C2 复位 软件置位和清零. 0:清除复位 1:复位 I2C2
21	I2C1RST	I2C1 复位 软件置位和清零.

位域	名称	描述
		0:清除复位 1:复位 I2C1
20:18	Reserved	保留, 必须保持复位值
17	USART2RST	USART2 复位 软件置位和清零. 0:清除复位 1:复位 USART2
16:12	Reserved	保留, 必须保持复位值
11	WWDGRST	窗口看门狗复位 软件置位和清零. 0: 清除复位 1: 复位窗口看门狗
10:7	Reserved	保留, 必须保持复位值
6	COMPRST	COMP 复位 软件置位和清零. 0: 清除复位 1: 复位 COMP
5	Reserved	保留, 必须保持复位值
4	TIM6RST	TIM6 复位 软件置位和清零. 0:清除复位 1:复位 TIM6
3	TIM5RST	TIM5 复位 软件置位和清零. 0:清除复位 1:复位 TIM5
2	TIM4RST	TIM4 复位 软件置位和清零. 0:清除复位 1:复位 TIM4
1	TIM3RST	TIM3 复位 软件置位和清零. 0:清除复位 1:复位 TIM3
0	TIM2RST	TIM2 复位 软件置位和清零. 0:清除复位 1:复位 TIM2

#### 4.4.7 AHB 外设时钟使能寄存器(RCC\_AHPCLKEN)

偏移地址: 0x14

复位值: 0x0000 0014



位域	名称	描述
31:13	Reserved	保留, 必须保持复位值
12	ADCEN	ADC 时钟使能 软件置位和清零. 0: ADC 时钟禁能 1: ADC 时钟使能
11	Reserved	保留, 必须保持复位值
10	IOPDEN	IO 端口 D 时钟使能 软件置位和清零. 0: IO 端口 D 时钟禁能 1: IO 端口 D 时钟使能
9	IOPCEN	IO 端口 C 时钟使能 软件置位和清零. 0: IO 端口 C 时钟禁能 1: IO 端口 C 时钟使能
8	IOPBEN	IO 端口 B 时钟使能 软件置位和清零. 0: IO 端口 B 时钟禁能 1: IO 端口 B 时钟使能
7	IOPAEN	IO 端口 A 时钟使能 软件置位和清零. 0: IO 端口 A 时钟禁能 1: IO 端口 A 时钟使能
6	CRCEN	CRC 时钟使能 软件置位和清零. 0: CRC 时钟禁能 1: CRC 时钟使能
5	Reserved	保留, 必须保持复位值
4	FLITFEN	FLITF 时钟使能 在 SLEEP 模式下, 软件置位和清零以启用/禁用 FLITF 时钟.

位域	名称	描述
		0: 在 SLEEP 模式下 FLITF 时钟禁能 1: 在 SLEEP 模式下 FLITF 时钟使能
3	Reserved	保留, 必须保持复位值
2	SRAMEN	SRAM 时钟使能 在 SLEEP 模式下, 软件置位和清零以启用/禁用 SRAM 时钟。 0: 在 SLEEP 模式下 SRAM 时钟禁能 1: 在 SLEEP 模式下 SRAM 时钟使能
1	Reserved	保留, 必须保持复位值
0	DMAEN	DMA 时钟使能 软件置位和清零. 0: DMA 时钟禁能 1: DMA 时钟使能

#### 4.4.8 APB2 外设时钟使能寄存器 (RCC\_APB2PCLKEN)

偏移地址: 0x18

复位值: 0x0000 0000

31	Reserved										20	19	18	17	16
15	14	13	12	11	10						rw	rw	rw		
Reserved	USART1EN	TIM8EN	SPI1EN	TIM1EN										BEEPEN	AFIOEN

位域	名称	描述
31:20	Reserved	保留, 必须保持复位值
19	SPI2EN	SPI2 时钟使能 软件置位和清零. 0: SPI2 时钟禁能 1: SPI2 时钟使能
18	UART4EN	UART4 时钟使能 软件置位和清零. 0: UART4 时钟禁能 1: UART4 时钟使能
17	UART3EN	UART3 时钟使能 软件置位和清零. 0: UART3 时钟禁能 1: UART3 时钟使能
16:15	Reserved	保留, 必须保持复位值
14	USART1EN	USART1 时钟使能 软件置位和清零.

位域	名称	描述
		0: USART1 时钟禁能 1: USART1 时钟使能
13	TIM8EN	TIM8 时钟使能 软件置位和清零. 0: TIM8 时钟禁能 1: TIM8 时钟使能
12	SPI1EN	SPI1 时钟使能 软件置位和清零. 0: SPI1 时钟禁能 1: SPI1 时钟使能
11	TIM1EN	TIM1 时钟使能 软件置位和清零. 0: TIM1 时钟禁能 1: TIM1 时钟使能
10:2	Reserved	保留, 必须保持复位值
1	BEEPEN	BEEPER 时钟使能 软件置位和清零. 0: BEEPER 时钟禁能 1: BEEPER 时钟使能
0	AFIOEN	AFIO 时钟使能 软件置位和清零. 0: AFIO 时钟禁能 1: AFIO 时钟使能

#### 4.4.9 APB1 外设时钟使能寄存器(RCC\_APB1PCLKEN)

偏移地址: 0x1c

复位值: 0x0000 0000

31	29	28	27	26	25	24	23	22	21	20	18	17	16
Reserved	PWREN	Reserved	CANEN	Reserved	I2C2EN	I2C1EN	Reserved	USART2EN	Reserved				
	rw 12	rw 11	rw 10		rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1	rw 0
15	Reserved	WWDGREN	Reserved	COMP FILTEN	COMPEN	Reserved	TIM6EN	TIM5EN	TIM4EN	TIM3EN	TIM2EN		
	rw		rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:29	Reserved	保留，必须保持复位值
28	PWREN	电源接口时钟使能 软件置位和清零。 0：电源接口时钟禁能 1：电源接口时钟使能

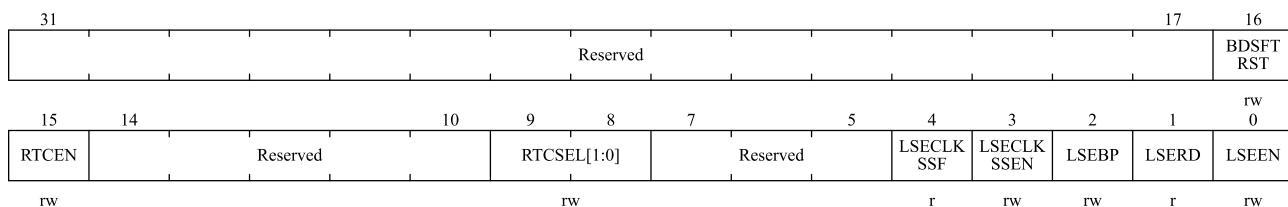
位域	名称	描述
27:26	Reserved	保留, 必须保持复位值
25	CANEN	CAN 时钟使能 软件置位和清零. 0: CAN 时钟禁能 1: CAN 时钟使能
24:23	Reserved	保留, 必须保持复位值
22	I2C2EN	I2C2 时钟使能 软件置位和清零. 0: I2C2 时钟禁能 1: I2C2 时钟使能
21	I2C1EN	I2C1 时钟使能 软件置位和清零. 0: I2C1 时钟禁能 1: I2C1 时钟使能
20:18	Reserved	保留, 必须保持复位值
17	USART2EN	USART2 时钟使能 软件置位和清零. 0: USART2 时钟禁能 1: USART2 时钟使能
16:12	Reserved	保留, 必须保持复位值
11	WWDGEN	窗口看门狗时钟使能 软件置位和清零. 0: 窗口看门狗时钟禁能 1: 窗口看门狗时钟使能
10:8	Reserved	保留, 必须保持复位值
7	COMPFILTEN	COMP 滤波器时钟使能 软件置位和清零. 0: COMP 滤波器时钟禁能 1: COMP 滤波器时钟使能
6	COMPEN	COMP 时钟使能 软件置位和清零. 0: COMP 时钟禁能 1: COMP 时钟使能
5	Reserved	保留, 必须保持复位值
4	TIM6EN	TIM6 时钟使能 软件置位和清零. 0: TIM6 时钟禁能 1: TIM6 时钟使能
3	TIM5EN	TIM5 时钟使能 软件置位和清零. 0: TIM5 时钟禁能 1: TIM5 时钟使能

位域	名称	描述
2	TIM4EN	TIM4 时钟使能 软件置位和清零. 0: TIM4 时钟禁能 1: TIM4 时钟使能
1	TIM3EN	TIM3 时钟使能 软件置位和清零. 0: TIM3 时钟禁能 1: TIM3 时钟使能
0	TIM2EN	TIM2 时钟使能 软件置位和清零. 0: TIM2 时钟禁能 1: TIM2 时钟使能

#### 4.4.10 备份域控制寄存器(RCC\_BDCTRL)

偏移地址: 0x20

复位值: 0x0000 0000



位域	名称	描述
31:17	Reserved	保留, 必须保持复位值
16	BDSFTRST	备份域软件复位 软件置位和清零. 0:清除复位 1:复位备份域
15	RTCEN	RTC 时钟使能 软件置位和清零. 0:禁能 RTC 时钟 1:使能 RTC 时钟
14:10	Reserved	保留, 必须保持复位值
9:8	RTCSEL[1:0]	RTC 时钟源选择 由软件设置以选择 RTC 时钟源。一旦选择了 RTC 时钟源, 在下次备份域复位之前无法更改。这些位可以通过设置 BDSFTRST 位来复位。 00: 无时钟 01: 选择 LSE 振荡器作为 RTC 时钟 10: 选择 LSI 振荡器作为 RTC 时钟

位域	名称	描述
		11: 选择 HSE 振荡器 128 分频为 RTC 时钟
7:5	Reserved	保留, 必须保持复位值
4	LSECLKSSF	LSE 故障检测时的 CSS 标志 由硬件设置以指示 LSE 上的时钟安全系统是否检测到故障 0: 在 LSE 上未检测到故障 1: 在 LSE 上检测到故障
3	LSECLKSSEN	LSE 故障检测使能 0: 禁用 LSE 故障检测 1: 启用 LSE 故障检测
2	LSEBP	外部低速振荡器旁路 在调试模式下, 软件置位和清零旁路振荡器。该位只能在外部低速振荡器禁用时写入。 0: LSE 振荡器未旁路 1: LSE 振荡器旁路
1	LSERD	外部低速时钟振荡器就绪 由硬件置位和清零以指示 LSE 振荡器是否就绪。LSEEN 位清零后, LSERD 在 LSE 时钟的 6 个周期后清零。 0: 外部低速振荡器未就绪 1: 外部低速振荡器就绪
0	LSEEN	外部低速时钟振荡器使能 软件置位和清零。 0: 禁止外部低速振荡器 1: 使能外部低速振荡器。

注意: 备份域控制寄存器(RCC\_BDCTRL)中的LSEEN、LSEBP、LSECLKSSEN、RTCSEL 和 RTCEN 位在备份域中。因此, 这些位在复位后是写保护的, 只有在PWR\_CTRL.DBKP 位置位后才能更改。这些位只能通过备份域复位来清除。任何内部或外部复位都不会影响这些位。

#### 4.4.11 控制/状态寄存器(RCC\_CTRLSTS)

偏移地址: 0x24

复位值: 0x0c000003

31	30	29	28	27	26	25	24	23	22	21	20	16
LPWR RSTF	WWDG RSTF	IWDG RSTF	SFT RSTF	POR RSTF	PINRSTF	MMU RSTF	RMRSTF	Reserved	BKP EMCF		Reserved	
r 15	r	r	r	r	r	r	rw		r		2	1 0
								Reserved			LSIRD	LSIEN
										r		rw

位域	名称	描述
31	LPWRRSTF	低功耗复位标志 当发生低功耗管理复位时由硬件设置。 软件通过写入 RMRSTF 位清零。

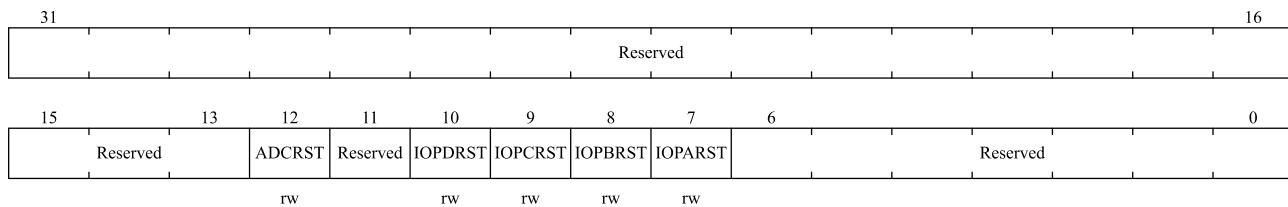
位域	名称	描述
		0: 未发生低功耗管理复位 1: 发生低功耗管理复位
30	WWDGRSTF	窗口看门狗复位标志 发生窗口看门狗复位时由硬件置位。 软件通过写入 RMRSTF 位清零。 0: 未发生窗口看门狗复位 1: 发生窗口看门狗复位
29	IWDGRSTF	独立看门狗复位标志 发生独立看门狗复位时由硬件置位 软件通过写入 RMRSTF 位清零。 0: 未发生独立看门狗复位 1: 发生独立看门狗复位
28	SFTRSTF	软件复位标志 发生软件复位时由硬件置位。 软件通过写入 RMRSTF 位清零。 0: 未发生软件复位 1: 发生软件复位
27	PORRSTF	上电/掉电复位标志 发生上电/掉电复位时由硬件置位 软件通过写入 RMRSTF 位清零。 0: 未发生上电/断电复位 1: 发生上电/掉电复位
26	PINRSTF	外部引脚复位标志 当 NRST 引脚发生复位时由硬件置位。 软件通过写入 RMRSTF 位清零。 0: 未发生 NRST 引脚复位 1: 发生 NRST 引脚复位
25	MMURSTF	MMU 复位标志 发生 MMU 复位时由硬件置位。 软件通过写入 RMRSTF 位清零。 0: 未发生 MMU 复位 1: 发生 MMU 复位
24	RMRSTF	清除复位标志 软件通过置 1 该位来清除所有复位标志。 0:无作用 1:清除复位标志
23:22	Reserved	保留, 必须保持复位值
21	BKPEMCF	备份域 EMC 复位标志 发生备份域 EMC 复位时由硬件设置。 软件通过写入 RMRSTF 位清零。 0: 未发生备份域 EMC 复位 1: 发生备份域 EMC 复位

位域	名称	描述
20:2	Reserved	保留, 必须保持复位值
1	LSIRD	内部低速振荡器就绪 由硬件置位和清零以指示内部 RC 40KHz 振荡器是否就绪。LSIEN 清零后, LSIRD 在 3 个内部 RC 40KHz 振荡器时钟周期后清零。 0: 内部 40KHz RC 振荡器时钟未就绪 1: 内部 40KHz RC 振荡器时钟就绪
0	LSIEN	内部低速振荡器使能 软件置位和清零。 0: 禁用内部 RC 40kHz 振荡器 1: 使能内部 RC 40kHz 振荡器

#### 4.4.12 AHB 外设复位寄存器(RCC\_AHBPRST)

偏移地址: 0x28

复位值: 0x0000 0000



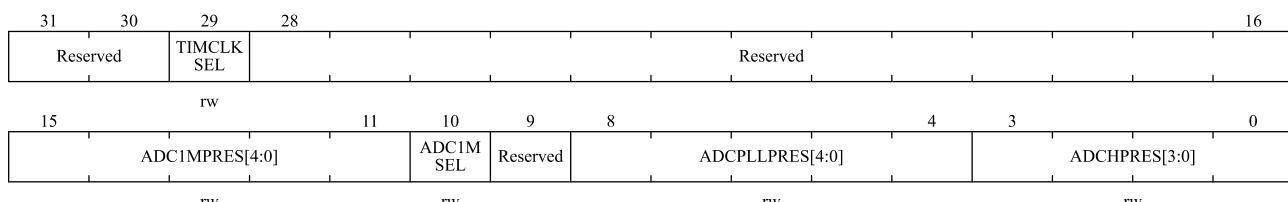
位域	名称	描述
31:13	Reserved	保留, 必须保持复位值
12	ADCRST	ADC 复位 软件置位和清零. 0: 清除复位 1: 复位 ADC
11	Reserved	保留, 必须保持复位值
10	IOPDRST	IO 端口 D 复位 软件置位和清零. 0: 清除复位 1: 复位 IO 端口 D
9	IOPCRST	IO 端口 C 复位 软件置位和清零. 0: 清除复位 1: 复位 IO 端口 C
8	IOPBRST	IO 端口 B 复位 软件置位和清零. 0: 清除复位 1: 复位 IO 端口 B

位域	名称	描述
7	IOPARST	IO 端口 A 复位 软件置位和清零。 0: 清除复位 1: 复位 IO 端口 A
6:0	Reserved	保留, 必须保持复位值

#### 4.4.13 时钟配置寄存器 2 (RCC\_CFG2)

偏移地址: 0x2c

复位值: 0x0000 3800



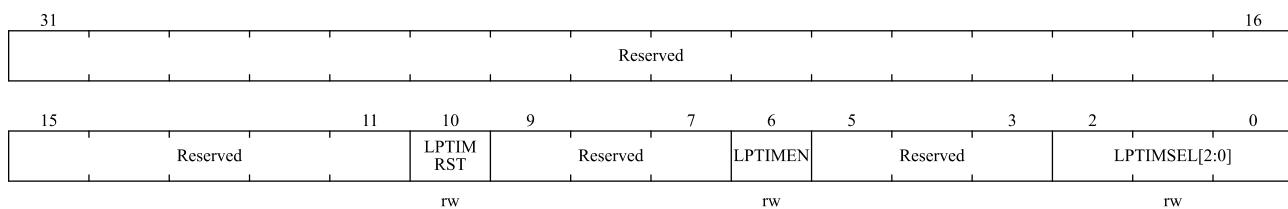
位域	名称	描述
31:30	Reserved	保留, 必须保持复位值
29	TIMCLKSEL	TIM1/8 时钟源选择 软件置位和清零。 0: 若 APB2 分频为 1, 则选择 PCLK2 作为 TIM1/8 时钟源; 否则, 选择 PCLK2×2 作为 TIM1/8 时钟源。 1: 选择 SYSCLK 输入时钟作为 TIM1/8 时钟源。
28:16	Reserved	保留, 必须保持复位值
15:11	ADC1MPRES[4:0]	ADC1M 时钟分频 软件置位和清零这些位来配置 ADC1M 时钟源的预分频系数。 00000: ADC 1M 时钟源不分频 00001: ADC 1M 时钟源 2 分频 00010: ADC 1M 时钟源 3 分频 ... 11110: ADC 1M 时钟源 31 分频 11111: ADC 1M 时钟源 32 分频
10	ADC1MSEL	ADC 1M 时钟源选择 软件置位和清零。 0: 选择 HSI 振荡器时钟作为 ADC1M 的输入时钟 1: 选择 HSE 振荡器时钟作为 ADC1M 的输入时钟 注意: 切换 ADC1M 时钟源时, 需要保证 HSI 时钟开启
9	Reserved	保留, 必须保持复位值
8:4	ADCPLL PRES[4:0]	ADC PLL 预分频 软件置位和清零这些位以配置 PLL 时钟到 ADC 的分频系数。

位域	名称	描述
		0xxxx: ADC PLL 时钟被禁用 10000: PLL 时钟不分频 10001: PLL 时钟 2 分频 10010: PLL 时钟 4 分频 10011: PLL 时钟 6 分频 10100: PLL 时钟 8 分频 10101: PLL 时钟 10 分频 10110: PLL 时钟 12 分频 10111: PLL 时钟 16 分频 11000: PLL 时钟 32 分频 11001: PLL 时钟 64 分频 11010: PLL 时钟 128 分频 11011: PLL 时钟 256 分频 其他: PLL 时钟 256 分频
3:0	ADCHCPRES[3:0]	ADC HCLK 预分频 软件置位和清零这些位以配置 HCLK 时钟到 ADC 的分频系数. 0000: HCLK 时钟不分频 0001: HCLK 时钟 2 分频 0010: HCLK 时钟 4 分频 0011: HCLK 时钟 6 分频 0100: HCLK 时钟 8 分频 0101: HCLK 时钟 10 分频 0110: HCLK 时钟 12 分频 0111: HCLK 时钟 16 分频 1000: HCLK 时钟 32 分频 其他: HCLK 时钟 32 分频

#### 4.4.14 保持域控制寄存器(RCC\_RDCTRL)

偏移地址: 0x34

复位值: 0x0000 0000



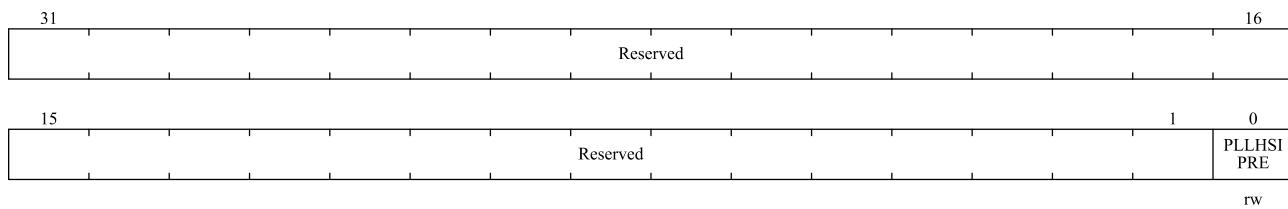
位域	名称	描述
31:11	Reserved	保留，必须保持复位值
10	LPTIMERRST	LPTIMER 复位 软件置位和清零.

位域	名称	描述
		0: 清除复位 1: 复位 LPTIMER
9:7	Reserved	保留, 必须保持复位值
6	LPTIMEN	LPTIMER 时钟使能 软件置位和清零. 0: LPTIMER 时钟禁能 1: LPTIMER 时钟使能
5:3	Reserved	保留, 必须保持复位值
2:0	LPTIMSEL[2:0]	LPTIMER 时钟源选择. 软件置位和清零 000: 选择 APB 时钟作为 LPTIMER 时钟 001: 选择 LSI 时钟作为 LPTIMER 时钟 010: 选择 HSI 时钟作为 LPTIMER 时钟 011: 选择 LSE 时钟作为 LPTIMER 时钟

#### 4.4.15 PLL HSI 配置寄存器(RCC\_PLLHSIPRE)

偏移地址: 0x40

复位值: 0x0000 0003

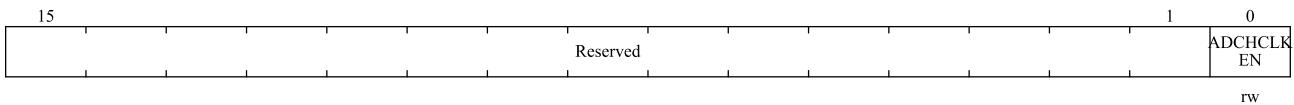
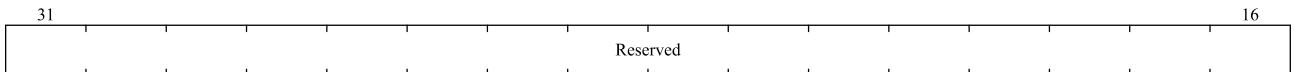


位域	名称	描述
31:1	Reserved	保留, 必须保持复位值
0	PLLHSIPRE	PLL 输入的 HSI 预分频器 由软件置位和清零, 配置进入 PLL 之前 HSI 的分频。该位只能在 PLL 禁用时写入。 0: HSI 时钟不分频 1: HSI 时钟 2 分频

#### 4.4.16 AHB 外设时钟使能寄存器 1 (RCC\_AHB1CLKEN)

偏移地址: 0x80

复位值: 0x0000 003E



位域	名称	描述
31:1	Reserved	保留, 必须保持复位值
0	ADCHCLKEN	ADC_HCLK 使能 软件置位和清零. 0: ADC_HCLK 禁能 1: ADC_HCLK 使能

## 5 GPIO 和 AFIO

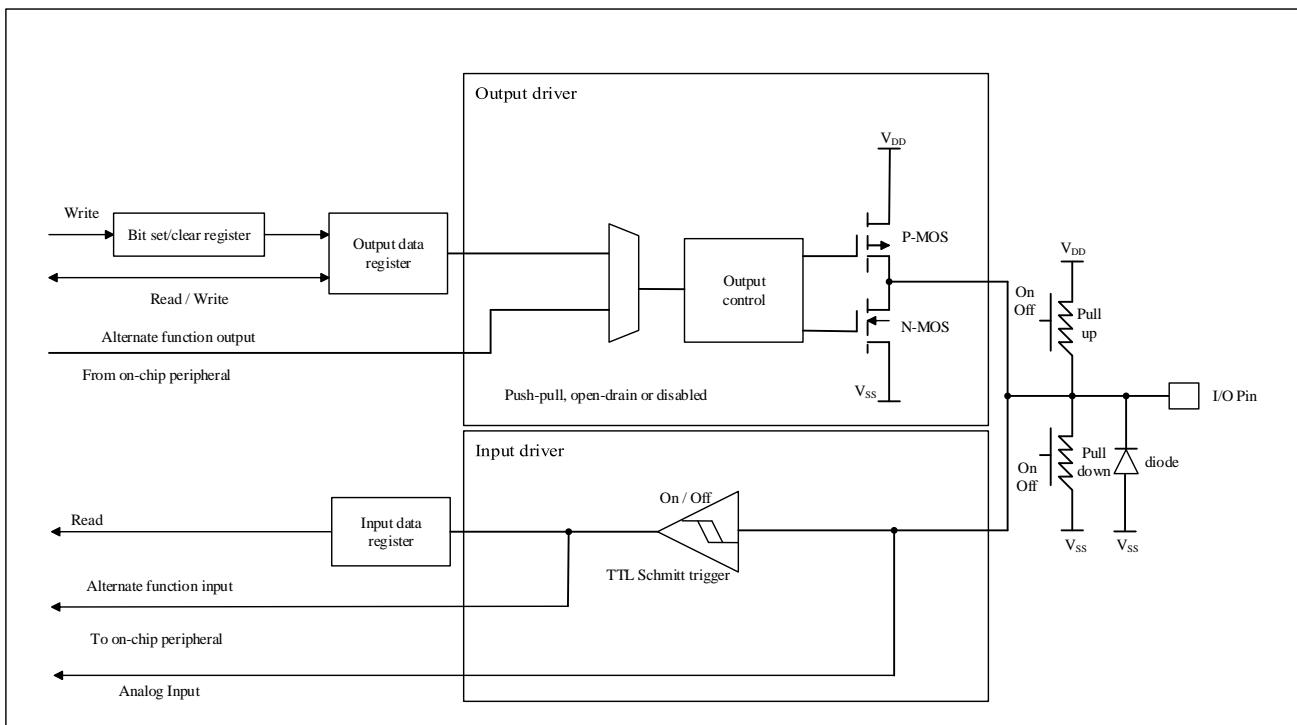
### 5.1 概述

本设计支持 40 个 GPIO，分为 4 组（GPIOA/GPIOB/GPIOC/GPIOD），GPIOA 和 GPIOB 各有 16 个引脚，GPIOC 有 3 个引脚，GPIOD 有 5 个引脚。每个 GPIO 引脚都可以通过软件配置为输出（推挽或开漏）、输入（带或不带上拉或下拉）或复用的外设功能端口（输出/输入），大多数 GPIO 引脚与数字或模拟复用外设共享，一些 I/O 引脚也与时钟引脚复用。除具有模拟输入功能的端口外，所有 GPIO 引脚都具有通过大电流的能力。

- 每个 GPIO 口都可以通过软件单独配置成多种模式
  - ◆ 输入浮空
  - ◆ 输入上拉（弱上拉，典型 50K）
  - ◆ 输入下拉（弱下拉，典型 50K）
  - ◆ 模拟功能
  - ◆ 开漏输出
  - ◆ 推挽输出
  - ◆ 推挽复用功能
  - ◆ 开漏复用功能
- 单独的位设置或位清除功能
- 所有 I/O 支持外部中断功能
- 支持软件重新映射 I/O 复用功能
- 支持 GPIO 锁定机制，复位锁定状态清除

每个 I/O 端口位都可以任意编程，并且 I/O 端口寄存器可以作为 32 位字、半字或字节访问，因为它是一个 AHB 接口。下图显示了 I/O 端口的基本结构。GPIOx\_PBSC 和 GPIOx\_PBC 寄存器的用途是允许对任何的 GPIOx\_POD 寄存器进行单独读取/修改访问。这样，在读取和修改访问之间产生 IRQ 时不会有风险。

图 5-1 5V 耐压 I/O 结构



## 5.2 功能描述

### 5.2.1 模式配置

I/O 口模式可以通过寄存器 GPIOx\_PMODE(x=A~D)、GPIOx\_POTYPE(x=A~D) 和 GPIOx\_PUPD(x=A~D) 进行配置。不同操作模式下的 I/O 配置如下表所示：

表 5-1 I/O 口配置表

PMODE[1:0]	POTYPE	PUPD[1:0]		I/O 配置
01	0	0	0	通用输出推挽 (Push-Pull)
	0	0	1	通用输出推挽 (Push-Pull) + 上拉
	0	1	0	通用输出推挽 (Push-Pull) + 下拉
	0	1	1	保留
	1	0	0	通用输出开漏 (Open-Drain)
	1	0	1	通用输出开漏 (Open-Drain) + 上拉
	1	1	0	通用输出开漏 (Open-Drain) + 下拉
	1	1	1	保留
10	0	0	0	复用功能+推挽 (Push-Pull)
	0	0	1	复用功能+推挽 (Push-Pull) + 上拉
	0	1	0	复用功能+推挽 (Push-Pull) + 下拉
	0	1	1	保留
	1	0	0	复用功能+开漏 (Open-Drain)

P MODE[1:0]	P OTYPE	P UPD[1:0]		I/O 配置
	1	0	1	复用功能+开漏(Open-Drain)+上拉
	1	1	0	复用功能+开漏(Open-Drain)+下拉
	1	1	1	保留
00	x	0	0	浮空输入
	x	0	1	上拉输入
	x	1	0	下拉输入
	x	1	1	保留
11	x	0	0	模拟模式
	x	0	1	保留
	x	1	0	
	x	1	1	

不同配置下 I/O 的输入输出特性如下表所示：

表 5-2 I/O 管脚功能特性列表

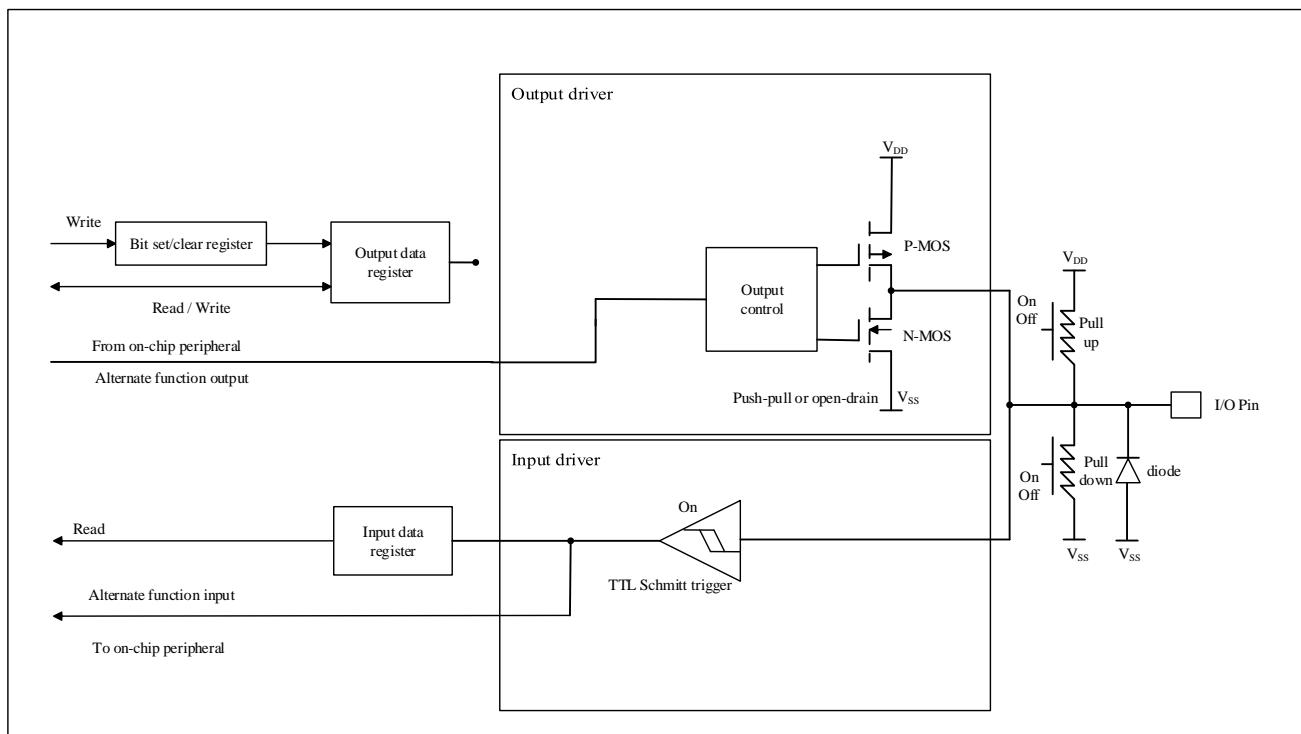
特性	GPIO输入	GPIO输出	模拟	复用功能
输出缓冲器	失能	使能	失能	根据外设功能从CPU自动配置
施密特触发器	使能	使能	失能, 输出强制为0(ToCore)	根据外设功能从CPU自动配置
上拉/下拉/浮空	可配	失能	失能	根据外设功能自动匹配
开漏模式	失能	可配置, 输出数据为“0”时GPIO输出0, “1”时GPIO为高阻	失能	可配置, 输出数据为“0”时GPIO输出0, “1”时GPIO为高阻
推挽模式	失能	可配置, 输出数据为“0”时GPIO输出0, 输出数据为“1”时GPIO输出1	失能	可配置, 输出数据为“0”时GPIO输出0, 输出数据为“1”时GPIO输出1
输入数据寄存器(I/O状态)	可读I/O状态	可读I/O状态	读出0(SCHMTOFF)	可读I/O状态
输出数据寄存器(输出值)	上次写入无效	可读可写	最后写入值无效	最后写入值无效

### 5.2.1.1 复用模式

当 I/O 端口配置为复用功能模式时：

- 开漏或推挽配置中，输出缓冲器打开
- 输出由来自外设的信号驱动
- 施密特触发器输入被激活
- 弱上拉和下拉电阻使能/失能
- 在每个 AHB 时钟周期，出现在 I/O 引脚上的数据被采样到输入数据寄存器中
- 在开漏模式下，I/O 状态可以通过读取输入数据寄存器来获得
- 在推挽模式下，对输出数据寄存器的读访问获取最后写入的值

图 5-2 复用功能模式

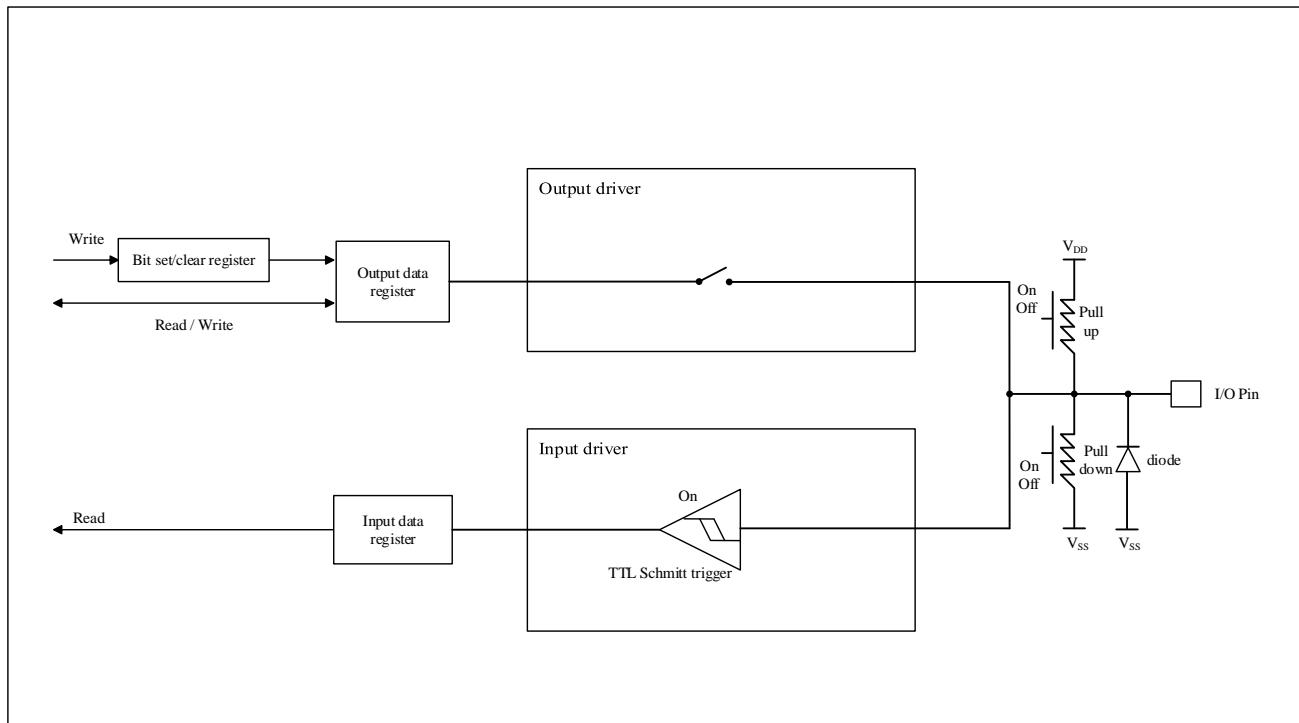


### 5.2.1.2 输入模式

当 I/O 口配置为输入模式时：

- 输出缓冲器被禁用
- 施密特触发器输入被激活
- 根据输入配置（上拉、下拉或浮空），连接弱上拉和下拉电阻
- 出现在 I/O 引脚上的数据在每个 AHB 时钟采样到输入数据寄存器
- 对输入数据寄存器的读访问可以得到 I/O 状态

图 5-3 浮空输入/上拉/下拉配置模式

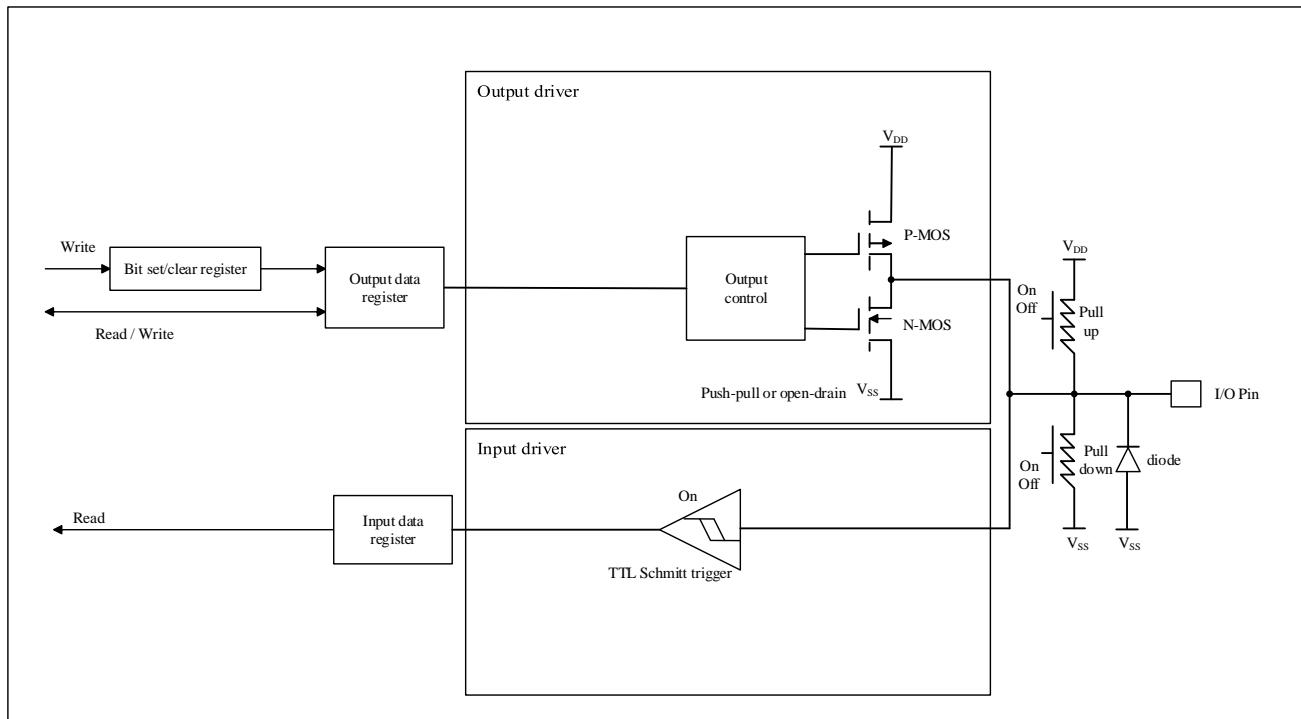


### 5.2.1.3 输出模式

当 I/O 口配置为输出模式时：

- 输出缓冲器被激活
  - ◆ 开漏模式：输出寄存器上的 0 激活 NMOS，而输出寄存器上的 1 使端口处于高阻状态（PMOS 永远不会被激活）
  - ◆ 推挽模式：输出寄存器为 0 激活 NMOS，输出寄存器为 1 激活 PMOS
- 施密特触发器输入被激活
- 弱上拉和下拉电阻使能/失能
- 出现在 I/O 引脚上的数据在每个 AHB 时钟采样到输入数据寄存器
- 在开漏模式下，I/O 状态可以通过读取输入数据寄存器来获得
- 在推挽模式下，对输出数据寄存器的读访问获取最后写入的值

图 5-4 输出模式

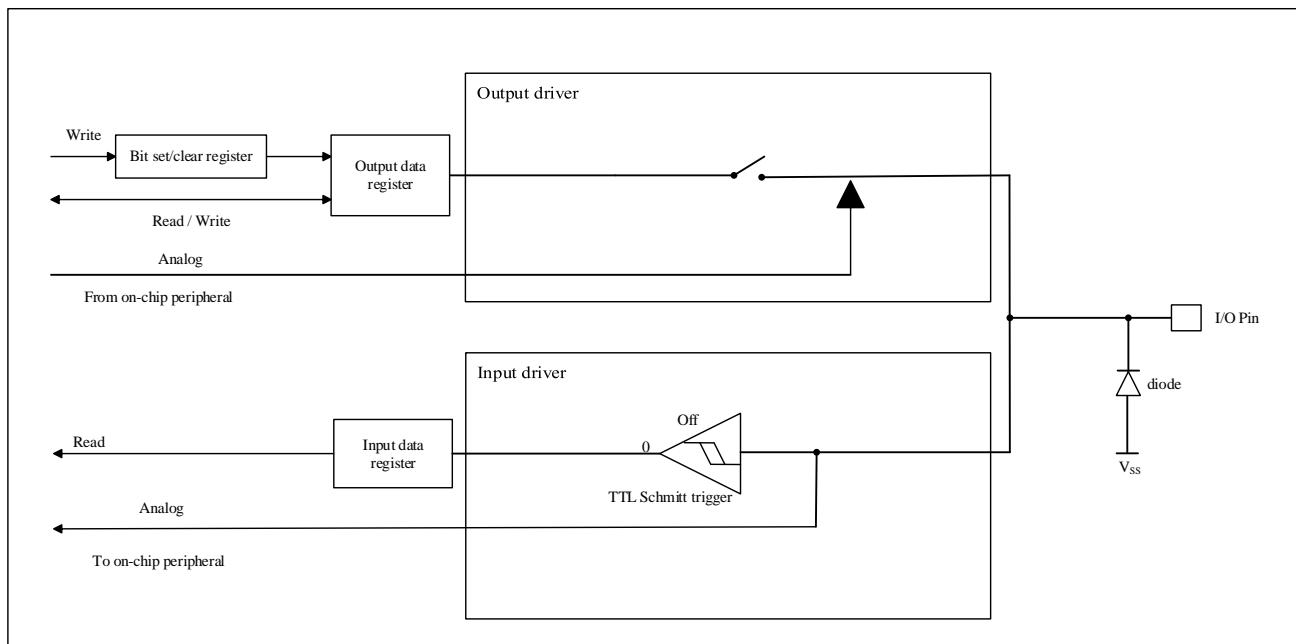


#### 5.2.1.4 模拟模式

当 I/O 端口编程为模拟模式时：

- 输出缓冲器被禁用
- 施密特触发器输入被禁用，为 I/O 引脚的每个模拟值提供零消耗。施密特触发器的输出被强制为恒定值 (0)
- 弱上拉和弱下拉电阻被禁用
- 对输入数据寄存器的读访问得到值“0”

图 5-5 高阻抗模拟输入配置



## 5.2.2 复位后状态

复位期间和刚复位后，复用功能不开启，I/O 端口被配置为模拟模式（GPIOx\_PMODE.PMODY[1:0]=11）。但是，以下 I/O 端口除外。

表 5-3 复位后的特殊引脚

TSSOP20	QFN20	QFN28	LQFP32	LQFP48	引脚名称	类型	模式	注释
19	16	21	23	34	PA13	I/O	输入上拉	SWDIO-JTMS
20	17	22	24	37	PA14	I/O	输入下拉	SWCLK-JTCK
-	-	23	25	38	PA15	I/O	输入上拉	JTDI
-	-	24	26	39	PB3	I/O	输出	JTDO
1	20	1	31	44	BOOT0/PD0	I/O	输入下拉	BOOT0
-	-	25	27	40	PB4	I/O	输入上拉	NJTRST

- BOOT0/PD0 默认输入下拉
- NRST 默认无 GPIO 功能，模拟引脚，无数字控制
- 复位后，与调试系统相关的引脚默认状态为使能 SWD-JTAG， JTAG 管脚置于输入上拉或下拉模式
  - ◆ PA15: JTDI 输入上拉
  - ◆ PA14: JTCK 输入下拉
  - ◆ PA13: JTMS 输入上拉

- ◆ PB4: NJTRST 输入上拉
- PC13、PC14、PC15:
  - ◆ PC13、PC14 和 PC15 是 BKP 电源域中的三个 I/O 引脚
  - ◆ 后续复用状态取决于 BKP 域寄存器配置。GPIO 模式只需要和 M4 一起操作，所以 GPIO 逻辑和内核在主电源域，复用功能在 BKP 域

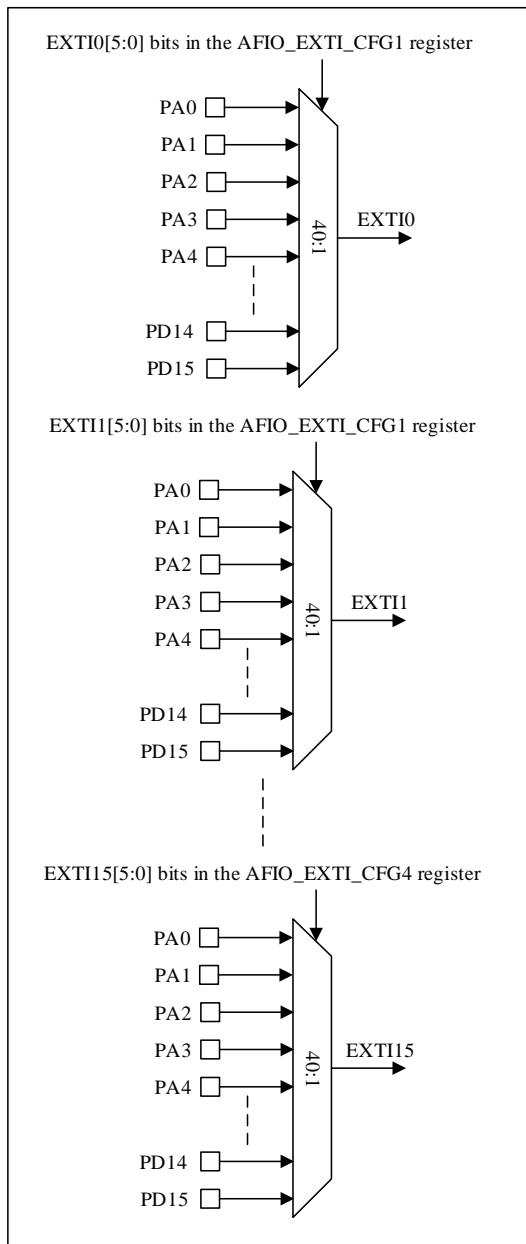
### 5.2.3 单独的位设置和位清除

通过将置位寄存器（GPIOx\_PBSC）和复位寄存器（GPIOx\_PBC）中要改变的位写“1”，可以实现数据寄存器（GPIOx\_POD）的单独位操作，可以设置/复位一个或多个位，写入'1'的位相应置位或清除，未写入'1'的位不会改变。该软件不需要禁用中断，并且在单个 AHB 写操作中完成。

### 5.2.4 外部中断/唤醒线

- 所有端口都具有外部中断能力，可以在 EXTI 模块中进行配置
- 为了使用外部中断线，端口必须配置为输入模式
- 所有端口均可配置为通过上升沿或下降沿触发中断的 SLEEP/STOP2 和 STOP0 模式唤醒功能
- 所有 I/O 口都连接到 16 条外部中断/事件线，如图 5-6 所示，相应的控制由寄存器 AFIO\_EXTI\_CFGx 配置

图 5-6 EXTI 全局中断



## 5.2.5 复用功能

当 I/O 端口被配置为复用功能模式时，端口位配置寄存器（GPIOx\_AFL/GPIOx\_AFH）和输出类型寄存器（GPIOx\_POTYPE 配置推挽或开漏）使用前必须编程，复用输入或输出由外设决定。

### 5.2.5.1 时钟输出 MCO

微控制器允许将时钟信号输出到外部 MCO 引脚(PA8/PA9)。相应的 GPIO 端口寄存器必须配置为复用功能模式。可以选择以下六个时钟信号作为 MCO 时钟，时钟选择由时钟配置寄存器 RCC\_CFG.MCO[26:23]位控制：

- LSI
- LSE

- SYSCLK
- HSI
- HSE
- PLL divided clock

### 5.2.5.2 RTC\_IOM 接口

RTC 和 IOM 接口支持以下功能:

- Tamper1: PC13
- Tamper2: PA0
- Tamper3: PA8
- Time-stamp—任何配置为 EXTI 线路的端口都可以配置为 RTC-Timestamp

注意:

- 1) 除 PC13、PC14 和 PC15 之外的引脚中的时间戳事件只能通过 EXTI 输入进行配置。
- 2) 但是, 可以将引脚 PC13、PC14 和 PC15 配置为在待机模式下支持时间戳。
- 3) 在待机模式下, 引脚 PC13、PC14 和 PC15 处于活动状态, 并通过这些引脚支持时间戳 (除了 EXTI 输入线), 通过 GPIOC\_AFH.AFSELy 位为这些引脚添加配置。

### 5.2.5.3 PC13-PC15 (BKP) 功能说明

PC13、PC14和PC15引脚位于BKP电源域, 可用于GPIO模式或功能模式。以下功能在功能模式下分配给这些引脚:

- GPIO PC14和PC15可用作GPIO、LSE振荡器引脚和时间戳输入
- PC13可用作GPIO、入侵输入、时间戳输入、RTC输出和待机唤醒源

### 5.2.5.4 HSE 引脚用作 GPIO 端口

HSE 的 OSC\_IN 和 OSC\_OUT 分别映射到 PD14 和 PD15。如果 HSE 关闭, 则对应的引脚可以用作 GPIO。如果 HSE 打开, 则相应的引脚进入模拟模式并绕过 GPIO 配置。

晶振配置为用户外部时钟模式, 引脚保持为时钟输入, OSC\_OUT 仍可作为普通 GPIO 使用。

### 5.2.5.5 LSE 引脚用作 GPIO 端口

LSE 的 OSC32\_IN 和 OSC32\_OUT 分别映射到 PC14 和 PC15。如果 LSE 关闭, 则对应的引脚可以用作 GPIO。如果 LSE 打开, 则相应的引脚进入模拟模式并绕过 GPIO 配置。

晶振配置为用户外部时钟模式, 引脚保持为时钟输入, OSC32\_OUT 不可作为普通 GPIO 使用。

### 5.2.5.6 JTAG/SWD 复用功能重映射

芯片上电默认开启 SWD-JTAG 调试接口, 接口信号映射到 GPIO 口, 如下表所示。

表 5-4 调试接口信号

调试信号	GPIO 引脚
JTMS/SWDIO	PA13
JTCK/SWCLK	PA14
JTDI	PA15
JTDO	PB3
NJTRST	PB4

如果在调试时需要使用它的 GPIO 功能，可以通过设置复用的重映射和调试 I/O 配置寄存器（GPIOx\_AFL 或 GPIOx\_AFH）来改变上述重映射配置。请参见下表。

表 5-5 调试端口映像

可能的调试端口	SWD-JTAG I/O 引脚分配				
	PA13/JTMS /SWDIO	PA14/JTCK/SW CLK	PA15/JTDI	PB3/JTDO	PB4/NJTRST
完全SWJ (JTAG-DP + SW-DP) (复位状态)	I/O不可用	I/O不可用	I/O不可用	I/O不可用	I/O不可用
完全SWJ (JTAG-DP + SW-DP) 但没有NJTRST	I/O不可用	I/O不可用	I/O不可用	I/O不可用	I/O可用
关闭JTAG-DP, 启用SW-DP	I/O不可用	I/O不可用	I/O可用	I/O可用	I/O可用
关闭JTAG-DP, 关闭SW-DP	I/O可用	I/O可用	I/O可用	I/O可用	I/O可用

JTAG 引脚上的内部上拉和下拉：

有必要确保 JTAG 输入引脚不悬空，因为它们直接连接到触发器以控制调试模式功能。必须特别小心。JTCK/SWCLK 引脚直接连接到其中一些触发器的时钟。

为避免任何不受控制的 I/O 电平，该器件在 JTAG 输入引脚上嵌入了内部上拉和下拉：

- NJTRST：内部上拉
- JTDI：内部上拉
- JTMS/SWDIO：内部上拉
- JTCK/SWCLK：内部下拉

### 5.2.5.7 ADC 外部触发复用功能重映射

ADC 的注入转换和规则转换的外部触发源支持重映射，参阅复用重映射和调试 I/O 配置寄存器 (AFIO\_RMP\_CFG)。

表 5-6 ADC 外部触发注入转换复用功能重映射

复用功能	ADC_ETRI=0	ADC_ETRI=1
ADC外部触发注入转换	ADC外部触发注入转换与EXTI (0-15) 相连	ADC外部触发注入转换与TIM8_Channel4相连

表 5-7 ADC 外部触发规则转换复用功能重映射

复用功能	ADC_ETRR=0	ADC_ETRR=1
ADC外部触发规则转换	ADC外部触发规则转换与EXTI (0-15)相连	ADC外部触发规则转换与TIM8_TRGO相连

### 5.2.5.8 TIMx 复用功能重映射

#### 5.2.5.8.1 TIM1 复用功能重映射

表 5-8 TIM1 复用功能重映射

复用功能	管脚	重映射
TIM1_ETR	PA12	AF3
TIM1_CH1	PA8	AF3
	PA0	AF10
	PB14	AF10
	PA9	AF3
TIM1_CH2	PA8	AF8
	PA10	AF3
TIM1_CH4	PA7	AF2
	PA11	AF3
	PB6	AF7
	PB9	AF10
TIM1_BKIN	PA10	AF2
	PA6	AF6
	PA4	AF6
	PB12	AF6
TIM1_CH1N	PA13	AF3
	PA1	AF2
	PA7	AF6
	PC13	AF2
TIM1_CH2N	PB14	AF3
	PB0	AF6
	PA6	AF2
	PB6	AF8
	PB15	AF10
	PB15	AF8
TIM1_CH3N	PB15	AF3
	PB1	AF6
	PA9	AF2
	PB14	AF2
TIM1_CH4N	PB2	AF6
	PB7	AF7

### 5.2.5.8.2 TIM2 复用功能重映射

表 5-9 TIM2 复用功能重映射

复用功能	管脚	重映射
TIM2_ETR	PA0	AF6
	PA15	AF3
TIM2_CH1	PA0	AF3
	PA15	AF6
TIM2_CH2	PA1	AF3
	PB3	AF3
TIM2_CH3	PA2	AF3
	PB10	AF3
TIM2_CH4	PB11	AF3

### 5.2.5.8.3 TIM3 复用功能重映射

表 5-10 TIM3 复用功能重映射

复用功能	管脚	重映射
TIM3_ETR	PB2	AF13
TIM3_CH1	PA6	AF3
	PB4	AF3
TIM3_CH2	PA7	AF3
	PB5	AF5
TIM3_CH3	PB0	AF3
TIM3_CH4	PB1	AF3

### 5.2.5.8.4 TIM4 复用功能重映射

表 5-11 TIM4 复用功能重映射

复用功能	管脚	重映射
TIM4_ETR	PA11	AF9
	PB10	AF13
TIM4_CH1	PB6	AF3
	PA7	AF9
TIM4_CH2	PB7	AF3
	PB1	AF10
TIM4_CH3	PA4	AF6
	PB8	AF3
TIM4_CH4	PA5	AF6
	PB9	AF3

### 5.2.5.8.5 TIM5 复用功能重映射

表 5-12 TIM5 复用功能重映射

复用功能	管脚	重映射

TIM5_CH1	PB12	AF13
	PA0	AF2
TIM5_CH2	PB13	AF13
	PA1	AF8
TIM5_CH3	PA2	AF7
TIM5_CH4	PA3	AF8

### 5.2.5.8.6 TIM8 复用功能重映射

表 5-13 TIM8 复用功能重映射

复用功能	管脚	重映射
TIM8_ETR	PA0	AF8
	PB7	AF13
TIM8_CH1	PA2	AF9
	PB14	AF13
TIM8_CH2	PA3	AF10
	PB15	AF13
TIM8_CH3	PA4	AF9
	PA6	AF9
	PB4	AF10
	PB8	AF10
	PA5	AF9
TIM8_CH4	PB5	AF10
	PA6	AF7
TIM8_BKIN	PA2	AF10
	PA9	AF10
	PB5	AF13
	PA7	AF7
TIM8_CH1N	PA15	AF11
	PB15	AF8
	PB0	AF8
TIM8_CH2N	PB3	AF8
	PB1	AF1
TIM8_CH3N	PB6	AF13

### 5.2.5.8.7 LPTIM 复用功能重映射

表 5-14 LPTIM 复用功能重映射

复用功能	管脚	重映射
LPTIM_IN1	PA5	AF12
	PB5	AF3
LPTIM_IN2	PA9	AF12
	PB7	AF6
LPTIM_OUT	PA4	AF12

复用功能	管脚	重映射
	PB2	AF3
LPTIM_ETR	PA10	AF12
	PB6	AF9

### 5.2.5.9 CAN 复用功能重映射

CAN 信号可以被映射到端口 A、端口 B 上，如下表所示。

表 5-15 CAN 复用功能重映射

复用功能	管脚	重映射
CAN_RX	PA11	AF2
	PA4	AF11
	PB8	AF6
	PB6	AF11
CAN_TX	PA12	AF2
	PA5	AF11
	PB9	AF6
	PB7	AF11

### 5.2.5.10 USARTx 复用功能重映射

#### 5.2.5.10.1 USART1 管脚重映射

表 5-16 USART1 复用功能重映射

复用功能	管脚	重映射
USART1_CTS	PA11	AF5
USART1_RTS	PA12	AF5
USART1_TX	PA4	AF2
	PA9	AF5
	PB6	AF1
	PB8	AF1
USART1_RX	PA5	AF5
	PA10	AF5
	PB7	AF1
USART1_CK	PA8	AF5

#### 5.2.5.10.2 USART2 管脚重映射

表 5-17 USART2 复用功能重映射

复用功能	管脚	重映射
USART2_CTS	PA0	AF5
	PA15	AF7
USART2_RTS	PA1	AF5
	PB3	AF5
USART2_TX	PA2	AF5

	PA6	AF11
	PB4	AF5
	PD14	AF4
USART2_RX	PA3	AF5
	PA7	AF11
	PB5	AF7
	PD15	AF4
USART2_CK	PA4	AF5
	PA14	AF5

### 5.2.5.11 USARTx 复用功能重映射

#### 5.2.5.11.1 UART3 管脚重映射

表 5-18 UART3 复用功能重映射

复用功能	管脚	重映射
UART3_TX	PB4	AF7
	PB8	AF7
	PB10	AF10
UART3_RX	PB5	AF8
	PB9	AF7
	PB11	AF10

#### 5.2.5.11.2 UART4 管脚重映射

表 5-19 UART4 复用功能重映射

复用功能	管脚	重映射
UART4_TX	PA0	AF11
	PB0	AF7
	PB14	AF7
	PD13	AF6
UART4_RX	PB1	AF7
	PB15	AF7
	PD12	AF6

### 5.2.5.12 I<sup>2</sup>C 复用功能重映射

#### 5.2.5.12.1 I<sup>2</sup>C1 管脚重映射

表 5-20 I<sup>2</sup>C1 管脚重映射

复用功能	管脚	重映射
I <sup>2</sup> C1_SCL	PA4	AF8
	PA15	AF8
	PB6	AF2
	PB8	AF5
	PD13	AF7
I <sup>2</sup> C1_SDA	PA5	AF8
	PA14	AF8
	PB7	AF2
	PB9	AF5
	PD12	AF7
I <sup>2</sup> C1_SMBA	PB5	AF2

#### 5.2.5.12.2 I<sup>2</sup>C2 管脚重映射

表 5-21 I<sup>2</sup>C2 管脚重映射

复用功能	管脚	重映射
I <sup>2</sup> C2_SCL	PA3	AF6
	PA9	AF7
	PB10	AF7
	PB13	AF6
	PD15	AF6
I <sup>2</sup> C2_SDA	PA2	AF6
	PA10	AF7
	PA8	AF7
	PB11	AF7
	PB14	AF6
	PD14	AF6
I <sup>2</sup> C2_SMBA	PB12	AF9
	PA8	AF2

### 5.2.5.13 SPI/I<sup>2</sup>S 复用功能重映射

#### 5.2.5.13.1 SPI1/I<sup>2</sup>S1 管脚重映射

表 5-22 SPI1/I<sup>2</sup>S1 管脚重映射

复用功能	管脚	重映射
SPI1_I2S1_NSS_WS	PA1	AF1
	PA4	AF1
	PA8	AF6
	PA15	AF1
	PB6	AF5
SPI1_I2S1_SCK_CK	PA5	AF1
	PA10	AF1
	PB3	AF2
	PB1	AF5
SPI1_I2S1_MISO_MCK	PA0	AF1
	PA6	AF1
	PB4	AF2
SPI1_I2S1_MOSI_SD	PA7	AF1
	PB5	AF1

#### 5.2.5.13.2 SPI2/I<sup>2</sup>S2 管脚重映射

表 5-23 SPI2/I<sup>2</sup>S2 管脚重映射

复用功能	管脚	重映射
SPI2_I2S2_NSS_WS	PA4	AF3
	PA13	AF6
	PA15	AF2
	PB12	AF1
SPI2_I2S2_SCK_CK	PA10	AF6
	PB6	AF6
	PB13	AF1
	PD12	AF5
SPI2_I2S2_MISO_MCK	PA9	AF1
	PA11	AF1
	PA13	AF1
	PB7	AF5
	PB14	AF1
SPI2_I2S2_MOSI_SD	PA12	AF1
	PA14	AF1
	PB1	AF2
	PB15	AF1

### 5.2.5.14 COMP 复用功能重映射

#### 5.2.5.14.1 COMP1 管脚重映射

表 5-24 COMP1 管脚重映射

复用功能	管脚	重映射
COMP1_OUT	PA0	AF9
	PA11	AF8
	PB6	AF10
	PB8	AF8

#### 5.2.5.14.2 COMP2 管脚重映射

表 5-25 COMP2 管脚重映射

复用功能	管脚	重映射
COMP2_OUT	PA2	AF8
	PA6	AF8
	PA7	AF8
	PA12	AF8
	PA14	AF9
	PB9	AF8

#### 5.2.5.14.3 COMP3 管脚重映射

表 5-26 COMP3 管脚重映射

复用功能	管脚	重映射
COMP3_OUT	PA6	AF10
	PA8	AF10
	PA10	AF10
	PB4	AF11
	PB10	AF11

#### 5.2.5.15 EVENTOUT 复用功能重映射

表 5-27 EVENTOUT 管脚重映射

复用功能	管脚	重映射
EVENTOUT	PA0~PA15	AF4
	PB0~PB15	AF4
	PC13	AF4
	PD12~PD13	AF4

#### 5.2.5.16 BEEPER 复用功能重映射

表 5-28 BEEPER 管脚重映射

复用功能	管脚	重映射
BEEPER_OUT_P	PA6	AF12
	PB6	AF12

复用功能	管脚	重映射
BEEPER_OUT_N	PA7	AF12
	PB7	AF12

### 5.2.5.17 JTAG/SWD 复用功能重映射

表 5-29 JTAG/SWD 管脚重映射

复用功能	管脚	重映射
JTMS/SWDIO	PA13	AF0
JTCK/SWCLK	PA14	AF0
JTDI	PA15	AF0
JTDO	PB3	AF0
NJTRST	PB4	AF0

### 5.2.5.18 TIMESTAMP 复用功能重映射

表 5-30 TIMESTAMP 管脚重映射

复用功能	管脚	重映射
TIMESTAMP	PC13	AF9
	PC14	AF9
	PC15	AF9

### 5.2.5.19 RTC\_REFIN 复用功能重映射

表 5-31 RTC\_REFIN 管脚重映射

复用功能	管脚	重映射
RTC_REFIN	PB15	AF9
	PA10	AF9

### 5.2.5.20 MCO 复用功能重映射

表 5-32 MCO 管脚重映射

复用功能	管脚	重映射
MCO	PA8	AF9
	PA9	AF9

## 5.2.6 外设的 I/O 配置

表 5-33 ADC

ADC引脚	GPIO配置
ADC	模拟模式

表 5-34 TIM1/TIM8

TIM1/TIM8引脚	配置	GPIO配置模式
TIM1/8_CHx	输入捕获通道x	浮空输入
	输出比较通道x	推挽复用输出

TIM1/TIM8引脚	配置	GPIO配置模式
TIM1/8_CHxN	互补输出通道x	推挽复用输出
TIM1/8_BKIN	刹车输入	浮空输入
TIM1/8_ETR	外部触发时钟输入	浮空输入

表 5-35 TIM2/3/4/5

TIM2/3/4/5引脚	配置	GPIO配置模式
TIM2/3/4/5_CHx	输入捕获通道x	推挽复用输出
	输出比较通道x	推挽复用输出
TIM2/3/4/5_ETR	外部触发时钟输入	推挽复用输出

表 5-36 LPTIM

LPTIM引脚	配置	GPIO配置模式
LPTIM_INx	模拟输入	推挽复用输出
LPTIM_OUT	模拟输入	推挽复用输出
LPTIM_ETR	模拟输入	推挽复用输出

表 5-37 CAN

CAN引脚	配置	GPIO配置模式
CAN_TX	模拟输入	推挽复用输出
CAN_RX	模拟输出	推挽复用输出

表 5-38 USART

USART引脚	配置	GPIO配置
USARTx_TX	全双工模式	推挽复用输出
	半双工同步模式	推挽复用输出
USARTx_RX	全双工模式	推挽复用输出
	半双工同步模式	未用，可作为通用I/O
USARTx_CK	同步模式	推挽复用输出
USARTx_RTS	硬件流量控制	推挽复用输出
USARTx_CTS	硬件流量控制	推挽复用输出

表 5-39 UART

USART引脚	配置	GPIO配置
UARTx_TX	全双工模式	推挽复用输出
	半双工同步模式	推挽复用输出
UARTx_RX	全双工模式	推挽复用输出
	半双工同步模式	未用，可作为通用I/O

表 5-40 I2C

I2C引脚	配置	GPIO配置
I2Cx_SCL	I2C时钟	开漏复用输出
I2Cx_SDA	I2C数据	开漏复用输出
I2Cx_SMBA	SMBA数据	开漏复用输出

表 5-41 SPI-I2S

SPI-I2S引脚	配置	GPIO配置
SPIx_I2Sx_MOSI_SD	主模式	推挽复用输出
	从模式	推挽复用输出
SPIx_I2Sx_MISO_MCK	主模式	推挽复用输出
	从模式	推挽复用输出
SPIx_I2Sx_NSS_WS	主模式	推挽复用输出
	从模式	推挽复用输出
SPIx_I2Sx_SCK_CK	主模式	推挽复用输出
	从模式	未用，可作为通用I/O

表 5-42 JTAG/SWD

JTAG/SWD引脚	配置	GPIO配置
JTMS/SWDIO	输入上拉	推挽复用输出+上拉
JTCK/SWCLK	输入下拉	推挽复用输出+下拉
JTDI	输入上拉	推挽复用输出+上拉
JTDO	输出	推挽复用输出
NJTRST	输入上拉	推挽复用输出+上拉

表 5-43 RTC

RTC引脚	配置	GPIO配置
RTC_REFIN	模拟输入	推挽复用输出

表 5-44 COMP

COMP引脚	配置	GPIO配置
COMPx_OUT	模拟输出	推挽复用输出

表 5-45 EVENT\_OUT

EVENT_OUT引脚	配置	GPIO配置
EVENT_OUT	模拟输出	推挽复用输出

表 5-46 其他

Other引脚	配置	GPIO配置
MCO	模拟输出	推挽复用输出
EXTI lines	模拟输入	浮空输入/上拉/下拉

## 5.2.7 GPIO 锁定机制

- 锁定机制允许冻结 I/O 配置(GPIOx\_PMODE、GPIOx\_POTYPE、GPIOx\_PUPD、GPIOx\_DS 和 GPIOx\_SR) 和复用功能寄存器 (GPIOx\_AFL 和 GPIOx\_AFH) 的内容。当锁定程序在一个端口位上执行时，该端口位的配置将不再改变，直到下一次复位，参考端口配置锁定寄存器 GPIOx\_PLOCK；
- PLOCKKK，即 GPIOx\_PLOCK[16]，只有在正确的顺序 w1->w0->w1->r0 后才变为 1（这里 r0 也是必须的）。之后，只有在执行系统复位时才变为 0。GPIOx\_PLOCK.PLOCK[15:0] 只能在

GPIOx\_PLOCK.PLOCKK=0 时修改;

- 设置 GPIOx\_PLOCK.PLOCKK 位的锁定顺序, w1->w0->w1->r0 仅当 GPIOx\_PLOCK.PLOCK[15:0]中的值（1 或 0）在此顺序中不改变时才有效。如果 GPIOx\_PLOCK.PLOCK[15:0]中的值在此序列期间发生变化，则不会设置 GPIOx\_PLOCK.PLOCKK 位；
- 只要 GPIOx\_PLOCK.PLOCKK=0 和 GPIOx\_PLOCK.PLOCKx=0 或 1，所有配置位和复用功能位都可以修改。当 GPIOx\_PLOCK.PLOCKK=1 但 GPIOx\_PLOCK.PLOCK[x]=0 时，可以修改 GPIOx\_PLOCK.PLOCK[x]=0 对应的配置和复用功能位；
- 只有当 GPIOx\_PLOCK.PLOCKK=1 且 GPIOx\_PLOCK.PLOCK[x]=1 时，GPIOx\_PLOCK.PLOCK[x]=1 对应的配置才会被锁定，不能修改；
- 如果锁序列操作错误，则必须重新启动锁（w1->w0->w1->r0）操作。

## 5.2.8 GPIO 使用注意事项

- 当 VDD 和 VDDA 没有上电时，GPIO 施加电压不得超过 3.6V；
- 当 GPIO 上施加的电压为 5.5V 时，VDD 和 VDDA 不得低于 2.4V；
- 如果不希望 MCU 出现漏电情况，需要保证 GPIO 上施加的电压小于等于 VDD 和 VDDA；
- 当 GPIO 上施加的电压大于 VDD 和 VDDA 时，如果希望 MCU 漏电电流减小，需要在 GPIO 上串联电阻。

## 5.3 GPIO 寄存器

GPIO 寄存器可通过 AHB 总线访问。寄存器描述如下。

GPIO 端口地址如下：

GPIOA 基地址: 0x40023400

GPIOB 基地址: 0x40023800

GPIOC 基地址: 0x40023C00

GPIOD 基地址: 0x40024000

### 5.3.1 GPIOA 寄存器总览

表 5-47 GPIOA 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	GPIOA_PMODE		P MODE15		P MODE14		P MODE13		P MODE12		P MODE11		P MODE10		P MODE9		P MODE8		P MODE7		P MODE6		P MODE5		P MODE4		P MODE3		P MODE2		P MODE1		P MODE0
	Reset value	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

Offset	Register	GPIOA_POTYPE		31
		Reset value		30
0x04	GPIOA_SR	GPIOA_SR		29
		Reset value		28
0x08	GPIOA_PUPD	GPIOA_PUPD		27
		Reset value		26
0x0C	GPIOA_PID	GPIOA_PID		25
		Reset value		24
0x10	GPIOA_POD	GPIOA_POD		23
		Reset value		22
0x14	GPIOA_PBSC	GPIOA_PBSC		21
		Reset value		20
0x18	GPIOA_PBC	GPIOA_PBC		19
		Reset value		18
0x28	GPIOA_PLOCK	GPIOA_PLOCK		17
		Reset value		16
0x1C	GPIOA_PLOCK	PLOCK		
0	PLOCK5	PBC15	PBS15	PID15
0	PLOCK4	PBC14	PBS14	PID14
0	PLOCK3	PBC13	PBS13	PID13
0	PLOCK2	PBC12	PBS12	PID12
0	PLOCK11	PBC11	PBS11	PID11
0	PLOCK10	PBC10	PBS10	PID10
0	PLOCK9	PBC9	PBS9	PID9
0	PLOCK8	PBC8	PBS8	PID8
0	PLOCK7	PBC7	PBS7	PID7
0	PLOCK6	PBC6	PBS6	PID6
0	PLOCK5	PBC5	PBS5	PID5
0	PLOCK4	PBC4	PBS4	PID4
0	PLOCK3	PBC3	PBS3	PID3
0	PLOCK2	PBC2	PBS2	PID2
0	PLOCK1	PBC1	PBS1	PID1
0	PLOCK0	PBC0	PBS0	PID0
				SR0
				SR1
				SR2
				SR3
				SR4
				SR5
				SR6
				SR7
				SR8
				SR9
				SR10
				SR11
				SR12
				SR13
				SR14
				SR15
				POT15
				POT14
				POT13
				POT12
				POT11
				POT10
				POT9
				POT8
				POT7
				POT6
				POT5
				POT4
				POT3
				POT2
				POT1
				POT0

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0x20	GPIOA_AFL	0	0	AFSEL7																															
		Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0x24	GPIOA_AFH	0	0	AFSEL15																															
		Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0x2C	GPIOA_DS	0	1	DS15																															
		Reset value	0	1	0	1	DS14																												

### 5.3.2 GPIOB 寄存器总览

表 5-48 GPIOB 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
0x00	GPIOB_PMODE	1	PMODE15	31	PMODE15	30	PMODE15	29	PMODE15	28	PMODE15	27	PMODE15	26	PMODE15	25	PMODE15	24	PMODE15	23	PMODE15	22	PMODE15	21	PMODE15	20	PMODE15	19	PMODE15	18	PMODE15	17	PMODE15	16		
		Reset value	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0		
0x04	GPIOB_POTYPE	Reserved																																		
		Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x08	GPIOB_SR	Reserved																																		
		Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
0x0C	GPIOB_PUPD	0	PUPD15	1	SROUT15	0	POT15	1	POT15	1																										
		Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

Offset	Register	Reset value	GPIOB_PID	Reset value	GPIOB_POD	Reset value	GPIOB_PBSC	Reset value	GPIOB_PBC	Reset value	GPIOB_PLOCK	Reset value	GPIOB_AFL	Reset value	GPIOB_AFH	Reset value	GPIOB_DS	Reset value	GPIOB_DS	Reset value
0x10																				
0x14																				
0x18																				
0x28																				
0x1C																				
0x20																				
0x24																				
0x2C																				

### 5.3.3 GPIOC 寄存器总览

表 5-49 GPIOC 寄存器总览

Offset	Register	Reset value	PMODE15	PMODE15	PMODE14	PMODE14	PMODE13	PMODE13	PMODE12	PMODE12	PMODE11	PMODE11	PMODE10	PMODE10	PMODE9	PMODE9	PMODE8	PMODE8	PMODE7	PMODE7	PMODE6	PMODE6	PMODE5	PMODE5	PMODE4	PMODE4	PMODE3	PMODE3	PMODE2	PMODE2	PMODE1	PMODE1	PMODE0	PMODE0
0x00	GPIOC_PMODE		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
0x04	GPIOC_POTYPE	Reset value																																
0x08	GPIOC_SR	Reset value																																
0x0C	GPIOC_PUPD	Reset value	PUPD15	0	0	0	0	0	PUPD14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0x10	GPIOC_PID	Reset value																																
0x14	GPIOC_POD	Reset value																																
0x18	GPIOC_PBSC	Reset value	PBC15	0	0	0	PBC14	0	PBC13	0																								

Offset	Register		31
0x28	GPIOC_PBC		30
0x1C	GPIOC_PLOCK		29
0x24	GPIOC_AFH		28
0x2C	GPIOC_DS		27
	Reset value	0 0 0 0 0 0	26
	Reset value	0 1 0 1 0 1	25
0x00	GPIOC_PMODE	DS15 DS14 DS13	24
0x04	GPIOC_POTYPE	PMODE15 PMODE14 PMODE13	23
0x08	GPIOC_SR	PMODE12	22
	Reset value	1 1 1 1 1 1	21
	Reset value	1 1 1 1 1 1	20
	Reset value	1 1 1 1 1 1	19
	Reset value	1 1 1 1 1 1	18
	Reset value	1 1 1 1 1 1	17
	Reset value	1 1 1 1 1 1	16
	Reset value	1 1 1 1 1 1	15
	Reset value	1 1 1 1 1 1	14
	Reset value	1 1 1 1 1 1	13
	Reset value	1 1 1 1 1 1	12
	Reset value	1 1 1 1 1 1	11
	Reset value	1 1 1 1 1 1	10
	Reset value	1 1 1 1 1 1	9
	Reset value	1 1 1 1 1 1	8
	Reset value	1 1 1 1 1 1	7
	Reset value	1 1 1 1 1 1	6
	Reset value	1 1 1 1 1 1	5
	Reset value	1 1 1 1 1 1	4
	Reset value	1 1 1 1 1 1	3
	Reset value	1 1 1 1 1 1	2
	Reset value	1 1 1 1 1 1	1
	Reset value	1 1 1 1 1 1	0

### 5.3.4 GPIOD 寄存器总览

表 5-50 GPIOD 寄存器总览

Offset	Register		31
0x00	GPIOD_PMODE	PMODE15 PMODE14 PMODE13	30
0x04	GPIOD_POTYPE	PMODE12	29
0x08	GPIOD_SR	SR15 SR14 SR13 SR12	28
	Reset value	1 1 1 1	27
	Reset value	1 1 1 1	26
	Reset value	1 1 1 1	25
	Reset value	1 1 1 1	24
	Reset value	1 1 1 1	23
	Reset value	1 1 1 1	22
	Reset value	1 1 1 1	21
	Reset value	1 1 1 1	20
	Reset value	1 1 1 1	19
	Reset value	1 1 1 1	18
	Reset value	1 1 1 1	17
	Reset value	1 1 1 1	16
	Reset value	1 1 1 1	15
	Reset value	1 1 1 1	14
	Reset value	1 1 1 1	13
	Reset value	1 1 1 1	12
	Reset value	1 1 1 1	11
	Reset value	1 1 1 1	10
	Reset value	1 1 1 1	9
	Reset value	1 1 1 1	8
	Reset value	1 1 1 1	7
	Reset value	1 1 1 1	6
	Reset value	1 1 1 1	5
	Reset value	1 1 1 1	4
	Reset value	1 1 1 1	3
	Reset value	1 1 1 1	2
	Reset value	1 1 1 1	1
	Reset value	1 1 1 1	0



Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	GPIOD_DS	DS15		DS14		DS13		DS12																								DS0	
	Reset value	0	1	0	1	0	1	0	1																					0	1		

### 5.3.5 GPIO 端口模式寄存器 (GPIOx\_PMODE)

偏移地址: 0x00

复位值 (端口 A): 0xABFFFFFF

复位值 (端口 B): 0xFFFFFEBF

复位值 (端口 C): 0xFC000000

复位值 (端口 D): 0xFF000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																
PMODE15[1:0]	PMODE14[1:0]	PMODE13[1:0]	PMODE12[1:0]	PMODE11[1:0]	PMODE10[1:0]	PMODE9[1:0]	PMODE8[1:0]																								
15	rw	14	rw	13	rw	12	rw	11	rw	10	rw	9	rw	8	rw	7	rw	6	rw	5	rw	4	rw	3	rw	2	rw	1	rw	0	
PMODE7[1:0]	PMODE6[1:0]	PMODE5[1:0]	PMODE4[1:0]	PMODE3[1:0]	PMODE2[1:0]	PMODE1[1:0]	PMODE0[1:0]																								
	rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw		rw

位域	名称	描述
31:30	PMODEy[1:0]	端口 x 的模式位 (y=0...15) 00: 输入模式 01: 通用输出模式 10: 复用功能模式 11: 模拟功能模式
29:28		
27:26		
25:24		
23:22		
21:20		
19:18		
17:16		
15:14		
13:12		
11:10		
9:8		
7:6		
5:4		
3:2		
1:0		

注意: 对于端口 C, 仅定义了 PMODE13、PMODE14 和 PMODE15 位字段, 其他字段保留。对于端口 D, 仅定义了 PMODE0、PMODE12、PMODE13、PMODE14 和 PMODE15 位域, 其他域保留。

### 5.3.6 GPIO 端口类型定义寄存器 (GPIOx\_POTYPE)

偏移地址: 0x04

复位值 (端口 A): 0x00000000

复位值 (端口 B): 0x00000000

复位值 (端口 C): 0x00000000

复位值 (端口 D): 0x00000000

Reserved																
31	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
POT15	POT14	POT13	POT12	POT11	POT10	POT9	POT8	POT7	POT6	POT5	POT4	POT3	POT2	POT1	POT0	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位域	名称	描述
31:16	保留	保留, 必须保持复位值。
15:0	POTy	端口 x 的输出模式位 (y=0...15) 0: 输出推挽模式 1: 输出开漏模式

注意: 对于端口 C, 仅定义了 POT13、POT14 和 POT15 位字段, 其他字段保留。对于端口 D, 仅定义了 POT0、POT12、POT13、POT14 和 POT15 位域, 其他域保留。

### 5.3.7 GPIO 翻转率配置寄存器 (GPIOx\_SR)

偏移地址: 0x08

复位值 (端口 A): 0x0000FFFF

复位值 (端口 B): 0x0000FFFF

复位值 (端口 C): 0x0000E000

复位值 (端口 D): 0x0000F001

Reserved																
31	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SR15	SR14	SR13	SR12	SR11	SR10	SR9	SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1	SR0	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位域	名称	描述
31:16	保留	保留, 必须保持复位值。
15:0	SRy	端口 GPIOx 的翻转率配置位 y (y=0...15) 0: 快速翻转 1: 慢速翻转

注意: 对于端口 C, 仅定义了 SR13、SR14 和 SR15 位字段, 其他字段保留。对于端口 D, 仅定义了 SR0、

*SR12、SR13、SR14 和 SR15 位字段，其他字段保留。*

### 5.3.8 GPIO 端口上下拉寄存器 (GPIOx\_PUPD)

偏移地址: 0x0C

复位值 (端口 A): 0x64000000

复位值 (端口 B): 0x000000100

复位值 (端口 C): 0x000000000

复位值 (端口 D): 0x00000002

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

位域	名称	描述
31:30	PUPDy[1:0]	端口 x 的模式位 (y=0...15)
29:28		00: 无上/下拉
27:26		01: 上拉
25:24		10: 下拉
23:22		11: 保留
21:20		
19:18		
17:16		
15:14		
13:12		
11:10		
9:8		
7:6		
5:4		
3:2		
1:0		

注意: 对于端口 C, 仅定义了 PUPD13、PUPD14 和 PUPD15 位字段, 其他字段保留。对于端口 D, 仅定义了 PUPD0、PUPD12、PUPD13、PUPD14 和 PUPD15 位字段, 其他字段保留。

### 5.3.9 GPIO 端口输入数据寄存器 (GPIOx\_PID)

偏移地址: 0x10

复位值 (端口 A): 0x00000000

复位值 (端口 B): 0x00000000

复位值 (端口 C): 0x00000000

复位值（端口 D）: 0x00000000

Reserved															
31	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
位域		名称		描述											
31:16		保留		保留，必须保持复位值。											
15:0		PIDy		端口输入数据 (y=0...15) 这些位为只读，它们包含对应的 I/O 端口的输入值。											

注意：对于端口 C，仅定义了 PID13、PID14 和 PID15 位字段，其他字段保留。对于端口 D，仅定义了 PID0、PID12、PID13、PID14 和 PID15 位字段，其他字段保留。

### 5.3.10 GPIO 端口输出数据寄存器 (GPIOx\_POD)

偏移地址：0x14

复位值（端口 A）: 0x00000000

复位值（端口 B）: 0x00000000

复位值（端口 C）: 0x00000000

复位值（端口 D）: 0x00000000

Reserved															
31	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
位域		名称		描述											
31:16		保留		保留，必须保持复位值。											
15:0		PODy		端口输出数据 (y=0...15) 这些位可以通过软件读和写。端口输出数据，通过 GPIOx_PBSC (x=A...D) 寄存器，可以对相应的 POD 位进行独立的设置/清除。											

注意：对于端口 C，仅定义了 POD13、POD14 和 POD15 位字段，其他字段保留。对于端口 D，仅定义了 POD0、POD12、POD13、POD14 和 POD15 位字段，其他字段保留。

### 5.3.11 GPIO 端口位设置/清除寄存器 (GPIOx\_PBSC)

偏移地址：0x18

复位值（端口 A）: 0x00000000

复位值（端口 B）: 0x00000000

复位值（端口 C）: 0x00000000

复位值（端口 D）: 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PBC15	PBC14	PBC13	PBC12	PBC11	PBC10	PBC9	PBC8	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
w 15	w 14	w 13	w 12	w 11	w 10	w 9	w 8	w 7	w 6	w 5	w 4	w 3	w 2	w 1	w 0
PBS15	PBS14	PBS13	PBS12	PBS11	PBS10	PBS9	PBS8	PBS7	PBS6	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0

位域	名称	描述
31:16	PBCy	<p>清除端口 GPIOx 的位 y (y=0...15)</p> <p>这些位是只写的，只能以 word 模式访问。</p> <p>0: 对相应的 PODy 位不产生影响</p> <p>1: 清除对应的 PODy 位为 0</p> <p>注：如果同时设置了 PBSy 和 PBCy 的对应位，PBSy 位起作用。</p>
15:0	PBSy	<p>设置端口 GPIOx 的位 y (y=0...15)</p> <p>这些位是只写的，只能以 word 模式访问。</p> <p>0: 对相应的 PODy 位不产生影响</p> <p>1: 设置对应的 PODy 位为 1</p>

注意：对于端口 C，仅定义了 PBC13、PBC14、PBC15、PBS13、PBS14 和 PBS15 位字段，其他字段保留。  
 对于端口 D，仅定义了 PBC0、PBC12、PBC13、PBC14、PBC15、PBS0、PBS12、PBS13、PBS14 和 PBS15 位域，其他域保留。

### 5.3.12 GPIO 端口锁定配置寄存器 (GPIOx\_PLOCK)

偏移地址: 0x1C

复位值（端口 A）: 0x00000000

复位值（端口 B）: 0x00000000

复位值（端口 C）: 0x00000000

复位值（端口 D）: 0x00000000

31	Reserved															17	16
PLOCK15	PLOCK14	PLOCK13	PLOCK12	PLOCK11	PLOCK10	PLOCK9	PLOCK8	PLOCK7	PLOCK6	PLOCK5	PLOCK4	PLOCK3	PLOCK2	PLOCK1	PLOCK0	RW 0	
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	

位域	名称	描述
31:17	保留	保留，必须保持复位值。
16	PLOCKK	<p>锁键</p> <p>可以随时读取该位，它只能使用锁键写入序列进行修改。</p> <p>0: 端口配置锁键位未激活</p> <p>1: 端口配置锁键位被激活，GPIOx_PLOCK 被锁定，直到 MCU 复位。</p> <p>锁键的写入序列：</p> <p>写 1-&gt;写 0-&gt;写 1-&gt;读 0-&gt;读 1</p>

位域	名称	描述
		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 PLOCK[15:0]的值。操作锁键写入序列中的任何错误将不能激活锁键。
15:0	PLOCKy	端口 GPIOx 的配置锁定位 y (y=0...15) 这些位可读可写但只能在 PLOCKK 位为 0 时写入。 0: 不锁定端口的配置 1: 锁定端口的配置

注意：对于端口 C，仅定义了 PLOCKK、PLOCK13、PLOCK14 和 PLOCK15 位字段，其他字段保留。对于端口 D，仅定义了 PLOCKK、PLOCK0、PLOCK12、PLOCK13、PLOCK14 和 PLOCK15 位字段，其他字段保留。

### 5.3.13 GPIO 复用功能低配置寄存器 (GPIOx\_AFL)

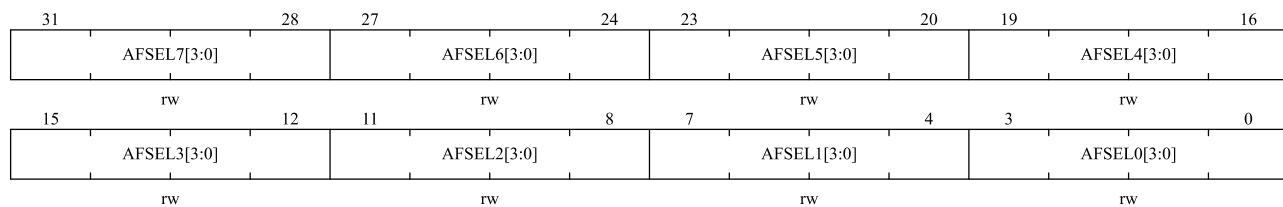
偏移地址：0x20

复位值（端口 A）：0x00000000

复位值（端口 B）：0x00000000

复位值（端口 C）：0x00000000

复位值（端口 D）：0x00000000



位域	名称	描述
位域	名称	描述
31:28	AFSELy[3:0]	端口 GPIOx 的复用功能配置位 y (y=0...7) 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12
27:24		
23:20		
19:16		
15:12		
11:8		
7:4		
3:0		

位域	名称	描述
		1101: AF13 1110: AF14 1111: AF15

注意：该寄存器没有为端口 C 定义。对于端口 D，仅定义了 AFSEL0 位域，所有其他域都保留。

### 5.3.14 GPIO 复用功能高配置寄存器 (GPIOx\_AFH)

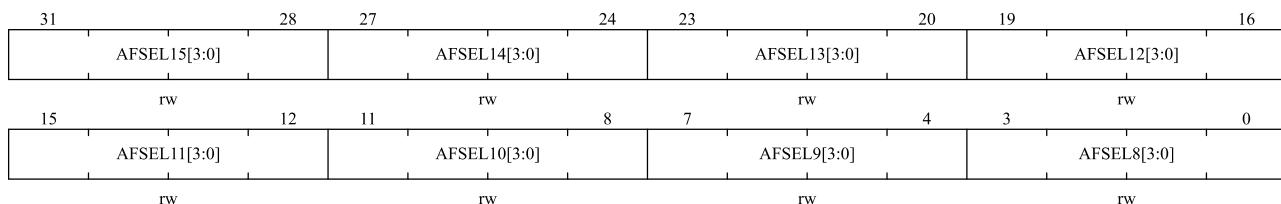
偏移地址：0x24

复位值（端口 A）：0x00000000

复位值（端口 B）：0x00000000

复位值（端口 C）：0x00000000

复位值（端口 D）：0x00000000



位域	名称	描述
位域	名称	描述
31:28	AFSELy[3:0]	端口 GPIOx 的复用功能配置位 y (y=8...15)
27:24		0000: AF0
23:20		0001: AF1
19:16		0010: AF2
15:12		0011: AF3
11:8		0100: AF4
7:4		0101: AF5
3:0		0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

注意：对于端口 C，仅定义了 AFSEL13、AFSEL14 和 AFSEL15 位字段，其他字段保留。对于端口 D，仅定义了 AFSEL12、AFSEL13、AFSEL14 和 AFSEL15 位域，其他域保留。

### 5.3.15 GPIO 端口位清除寄存器 (GPIOx\_PBC)

偏移地址: 0x28

复位值 (端口 A): 0x00000000

复位值 (端口 B): 0x00000000

复位值 (端口 C): 0x00000000

复位值 (端口 D): 0x00000000

31	Reserved															16
15	PBC15	PBC14	PBC13	PBC12	PBC11	PBC10	PBC9	PBC8	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位域	名称	描述
31:16	保留	保留, 必须保持复位值。
15:0	PBCy	清除端口 GPIOx 的位 y (y=0...15) 这些位是只写的, 只能以 word 模式访问。 0: 对相应的 PODy 位不产生影响 1: 清除对应的 PODy 位为 0

注意: 对于端口 C, 仅定义了 PBC13、PBC14 和 PBC15 位字段, 其他字段保留。对于端口 D, 仅定义了 PBC0、PBC12、PBC13、PBC14 和 PBC15 位域, 其他域保留。

### 5.3.16 GPIO 驱动能力配置寄存器 (GPIOx\_DS)

偏移地址: 0x2C

复位值 (端口 A): 0x55555555

复位值 (端口 B): 0x55555555

复位值 (端口 C): 0x54000000

复位值 (端口 D): 0x55000001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
DS15[1:0]		DS14[1:0]		DS13[1:0]		DS12[1:0]		DS11[1:0]		DS10[1:0]		DS9[1:0]		DS8[1:0]		
rw	15	rw	14	rw	13	rw	12	rw	11	rw	10	rw	9	rw	8	rw
DS7[1:0]	DS6[1:0]		DS5[1:0]		DS4[1:0]		DS3[1:0]		DS2[1:0]		DS1[1:0]		DS0[1:0]		rw	0
rw	rw		rw		rw		rw	rw								

位域	名称	描述
位域	名称	描述
31:30	DSy[1:0]	端口 GPIOx 的驱动能力配置位 y (y=0...15)
29:28		00: 2mA 驱动强度
27:26		01: 8mA 驱动强度

位域	名称	描述
25:24		10: 4mA 驱动强度
23:22		11: 12mA 驱动强度
21:20		
19:18		
17:16		
15:14		
13:12		
11:10		
9:8		
7:6		
5:4		
3:2		
1:0		

注意：对于端口 C，仅定义了 DS13、DS14 和 DS15 位字段，其他字段保留。对于端口 D，仅定义了 DS0、DS12、DS13、DS14 和 DS15 位字段，其他字段保留。

5.4 AFIO 寄存器

AFIO 寄存器可通过 APB 总线访问。寄存器说明如下。

#### 5.4.1 AFIO 寄存器总览

表 5-51 AFIO 寄存器总览

Offset	Register	Reset value	AFIO_RMP_CFG	31
				30
				29
0x00				28
				27
				26
				25
				24
				23
				22
				21
				20
				19
				18
				17
				16
				15
				14
				13
				12
				11
				10
				9
				8
				7
				6
				5
				4
				3
				2
				1
				0



## 5.4.2 AFIO 复用重映射配置寄存器 (AFIO\_RMP\_CFG)

偏移地址: 0x00

复位值: 0x00000000

31	Reserved										16
15	Reserved	SPI1_NSS	SPI2_NSS	ADC_ETRI	ADC_ETRR	EXTI_ETRI[3:0]				EXTI_ETRR[3:0]	0
	rw	rw	rw	rw	rw	rw				rw	

位域	名称	描述
31:12	Reserved	保留, 必须保持复位值。
11	SPI1_NSS	SPI1 的 NSS 模式选择位 (NSS 配置为 AFIO 推挽模式)。 0: NSS 空闲时为高阻态; 1: NSS 空闲时为高电平。
10	SPI2_NSS	SPI2 的 NSS 模式选择位 (NSS 配置为 AFIO 推挽模式)。 0: NSS 空闲时为高阻态; 1: NSS 空闲时为高电平。
9	ADC_ETRI	ADC 注入转换外部触发重映射 该位可由软件置'1'或置'0'。该位控制连接到 ADC 外部触发注入转换的触发输入。 0: ADC 注入转换外部触发与 EXTI (0-15) 相连 1: ADC 注入转换外部触发与 TIM8_Channel4 相连。
8	ADC_ETRR	ADC 规则转换外部触发重映射 该位可由软件置'1'或置'0'。该位控制连接到 ADC 外部触发规则转换的触发输入。 0: ADC 规则转换外部触发与 EXTI (0-15) 相连 1: ADC 规则转换外部触发与 TIM8_TRGO 相连
7:4	EXTI_ETRI[3:0]	选择中断线注入转换外部触发重映射
3:0	EXTI_ETRR[3:0]	选择中断线规则转换外部触发重映射

## 5.4.3 外部中断配置寄存器 1 (AFIO\_EXTI\_CFG1)

偏移地址: 0x04

复位值: 0x00000000

31	30	29	EXTI3[5:0]				24	23	22	21	16
Reserved			rw		Reserved					EXTI2[5:0]	
15	14	13		8	7	6	5			rw	0
Reserved			EXTI1[5:0]		Reserved					EXTI0[5:0]	
			rw							rw	

位域	名称	描述
31:30	保留	保留，必须保持复位值。
29:24	EXTI3[5:0]	<p>EXTI3配置 这些位由软件写入以选择EXTI3外部中断的源输入</p> <p>000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0</p>

位域	名称	描述
		100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
23:22	保留	保留，必须保持复位值。
21:16	EXTI2[5:0]	EXTI2配置 这些位由软件写入以选择EXTI2外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3]

位域	名称	描述
		001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13]

位域	名称	描述
		110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
15:14	保留	保留，必须保持复位值。
13:8	EXTI1[5:0]	<b>EXTI1配置</b> 这些位由软件写入以选择EXTI1外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7]

位域	名称	描述
		011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
7:6	保留	保留，必须保持复位值。
5:0	EXTIO[5:0]	<b>EXTIO配置</b> 这些位由软件写入以选择EXTIO外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1]

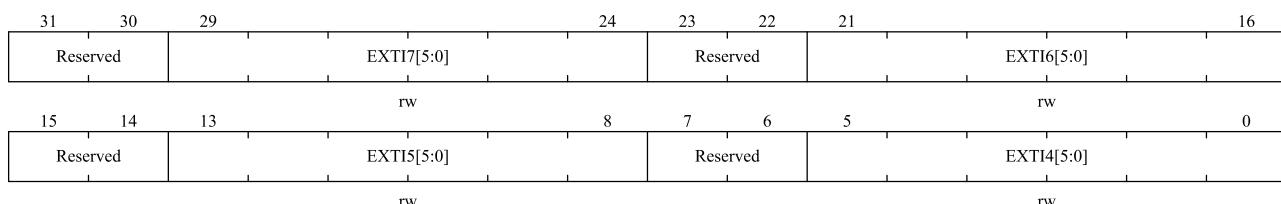
位域	名称	描述
		000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0

位域	名称	描述
		101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]

#### 5.4.4 外部中断配置寄存器 2 (AFIO\_EXTI\_CFG2)

偏移地址: 0x08

复位值: 0x00000000



位域	名称	描述
31:30	保留	保留, 必须保持复位值。
29:24	EXTI7[5:0]	EXTI7配置 这些位由软件写入以选择EXTI7外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2]

位域	名称	描述
		001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0

位域	名称	描述
		110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
23:22	保留	保留，必须保持复位值。
21:16	EXTI6[5:0]	<b>EXTI6配置</b> 这些位由软件写入以选择EXTI6外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6]

位域	名称	描述
		011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
15:14	保留	保留，必须保持复位值。
13:8	EXTI5[5:0]	EXTI5配置 这些位由软件写入以选择EXTI5外部中断的源输入 000000: PA[0]

位域	名称	描述
		000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0

位域	名称	描述
		101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
7:6	保留	保留，必须保持复位值。
5:0	EXTI4[5:0]	<b>EXTI4配置</b> 这些位由软件写入以选择EXTI4外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4]

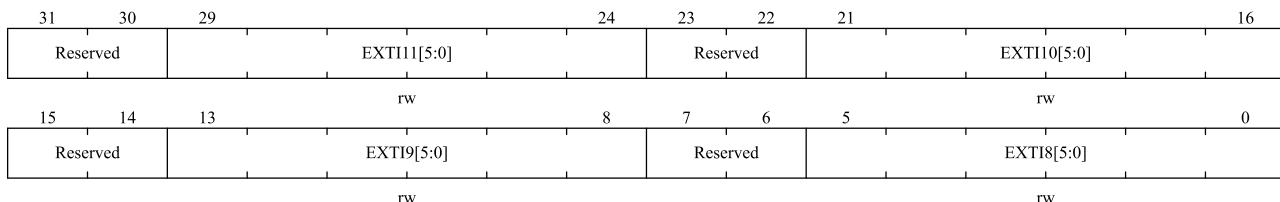
位域	名称	描述
		010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14]

位域	名称	描述
		111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]

### 5.4.5 外部中断配置寄存器 3 (AFIO\_EXTI\_CFG3)

偏移地址: 0x0C

复位值: 0x00000000



位域	名称	描述
31:30	保留	保留, 必须保持复位值。
29:24	EXTI11[5:0]	<b>EXTI11配置</b> 这些位由软件写入以选择EXTI11外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5]

位域	名称	描述
		010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15]

位域	名称	描述
		111111: PD[15]
23:22	保留	保留，必须保持复位值。
21:16	EXTI10[5:0]	<p>EXTI10配置 这些位由软件写入以选择EXTI10外部中断的源输入</p> <p>000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9]</p>

位域	名称	描述
		100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
15:14	保留	保留，必须保持复位值。
13:8	EXTI9[5:0]	<b>EXTI9配置</b> 这些位由软件写入以选择EXTI9外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3]

位域	名称	描述
		001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13]

位域	名称	描述
		110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
7:6	保留	保留，必须保持复位值。
5:0	EXTI8[5:0]	<b>EXTI8配置</b> 这些位由软件写入以选择EXTI8外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7]

位域	名称	描述
		011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]

#### 5.4.6 外部中断配置寄存器 4 (AFIO\_EXTI\_CFG4)

偏移地址: 0x10

复位值: 0x00000000

31	30	29		24	23	22	21		16
Reserved	EXTI15[5:0]					Reserved	EXTI14[5:0]		
15	14	13		8	7	6	5		0
Reserved	EXTI13[5:0]					Reserved	EXTI12[5:0]		
			rw				rw		rw

位域	名称	描述
31:30	保留	保留，必须保持复位值。
29:24	EXTI15[5:0]	<p>EXTI15配置 这些位由软件写入以选择EXTI15外部中断的源输入</p> <p>000000: PA[0]      000001: PB[0]      000010: 1'b0      000011: PD[0]      000100: PA[1]      000101: PB[1]      000110: 1'b0      000111: 1'b0      001000: PA[2]      001001: PB[2]      001010: 1'b0      001011: 1'b0      001100: PA[3]      001101: PB[3]      001110: 1'b0      001111: 1'b0      010000: PA[4]      010001: PB[4]      010010: 1'b0      010011: 1'b0      010100: PA[5]      010101: PB[5]      010110: 1'b0      010111: 1'b0      011000: PA[6]      011001: PB[6]      011010: 1'b0      011011: 1'b0      011100: PA[7]      011101: PB[7]      011110: 1'b0      011111: 1'b0      100000: PA[8]</p>

位域	名称	描述
		100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
23:22	保留	保留，必须保持复位值。
21:16	EXTI14[5:0]	<b>EXTI14配置</b> 这些位由软件写入以选择EXTI14外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0

位域	名称	描述
		000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12]

位域	名称	描述
		110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
15:14	保留	保留，必须保持复位值。
13:8	EXTI13[5:0]	<b>EXTI13配置</b> 这些位由软件写入以选择EXTI13外部中断的源输入 000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0

位域	名称	描述
		010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]
7:6	保留	保留，必须保持复位值。

位域	名称	描述
5:0	EXTI12[5:0]	<p>EXTI12配置 这些位由软件写入以选择EXTI12外部中断的源输入</p> <p>000000: PA[0] 000001: PB[0] 000010: 1'b0 000011: PD[0] 000100: PA[1] 000101: PB[1] 000110: 1'b0 000111: 1'b0 001000: PA[2] 001001: PB[2] 001010: 1'b0 001011: 1'b0 001100: PA[3] 001101: PB[3] 001110: 1'b0 001111: 1'b0 010000: PA[4] 010001: PB[4] 010010: 1'b0 010011: 1'b0 010100: PA[5] 010101: PB[5] 010110: 1'b0 010111: 1'b0 011000: PA[6] 011001: PB[6] 011010: 1'b0 011011: 1'b0 011100: PA[7] 011101: PB[7] 011110: 1'b0 011111: 1'b0 100000: PA[8] 100001: PB[8] 100010: 1'b0 100011: 1'b0 100100: PA[9] 100101: PB[9] 100110: 1'b0 </p>

位域	名称	描述
		100111: 1'b0 101000: PA[10] 101001: PB[10] 101010: 1'b0 101011: 1'b0 101100: PA[11] 101101: PB[11] 101110: 1'b0 101111: 1'b0 110000: PA[12] 110001: PB[12] 110010: 1'b0 110011: PD[12] 110100: PA[13] 110101: PB[13] 110110: PC[13] 110111: PD[13] 111000: PA[14] 111001: PB[14] 111010: PC[14] 111011: PD[14] 111100: PA[15] 111101: PB[15] 111110: PC[15] 111111: PD[15]

#### 5.4.7 5V 耐压配置寄存器 (AFIO\_TOL5V\_CFG)

偏移地址: 0x14

复位值: 0x00000000

31									25	24	23	22	21	20	19	18	17	16	
									Reserved		PB15 TOLENN	PB14 TOLENN	PB13 TOLENN	PB12 TOLENN	PB11 TOLENN	PB10 TOLENN	PB7 TOLENN	PB5 TOLENN	PB4 TOLENN
15	14	13	12	11	10	9			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
PB3 TOLENN	PB2 TOLENN	PB1 TOLENN	PB0 TOLENN	PA15 TOLENN	PA12 TOLENN	PA11 TOLENN	PA8 TOLENN	PA7 TOLENN	PA6 TOLENN	PA5 TOLENN	PA4 TOLENN	PA3 TOLENN	PA2 TOLENN	PA1 TOLENN	PA0 TOLENN				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

位域	名称	描述
31:25	保留	保留, 必须保持复位值。
24	PB15TOLENN	引脚 PB15 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压

位域	名称	描述
23	PB14TOLENN	引脚 PB14 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
22	PB13TOLENN	引脚 PB13 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
21	PB12TOLENN	引脚 PB12 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
20	PB11TOLENN	引脚 PB11 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
19	PB10TOLENN	引脚 PB10 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
18	PB7TOLENN	引脚 PB7 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
17	PB5TOLENN	引脚 PB5 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
16	PB4TOLENN	引脚 PB4 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
15	PB3TOLENN	引脚 PB3 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
14	PB2TOLENN	引脚 PB2 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
13	PB1TOLENN	引脚 PB1 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
12	PB0TOLENN	引脚 PB0 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
11	PA15TOLENN	引脚 PA15 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
10	PA12TOLENN	引脚 PA12 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压

位域	名称	描述
9	PA11TOLENN	引脚 PA11 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
8	PA8TOLENN	引脚 PA8 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
7	PA7TOLENN	引脚 PA7 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
6	PA6TOLENN	引脚 PA6 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
5	PA5TOLENN	引脚 PA5 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
4	PA4TOLENN	引脚 PA4 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
3	PA3TOLENN	引脚 PA3 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
2	PA2TOLENN	引脚 PA2 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
1	PA1TOLENN	引脚 PA1 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压
0	PA0TOLENN	引脚 PA0 5V 耐压使能 0: 使能 5V 耐压 1: 失能 5V 耐压

#### 5.4.8 模拟滤波器配置寄存器 1 (AFIO\_EFT\_CFG1)

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
PB15 EFTEN	PB14 EFTEN	PB13 EFTEN	PB12 EFTEN	PB11 EFTEN	PB10 EFTEN	PB9 EFTEN	PB8 EFTEN	PB7 EFTEN	PB6 EFTEN	PB5 EFTEN	PB4 EFTEN	PB3 EFTEN	PB2 EFTEN	PB1 EFTEN	PB0 EFTEN
rw 15	rw 14	rw 13	rw 12	rw 11	rw 10	rw 9	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1	rw 0
PA15 EFTEN	PA14 EFTEN	PA13 EFTEN	PA12 EFTEN	PA11 EFTEN	PA10 EFTEN	PA9 EFTEN	PA8 EFTEN	PA7 EFTEN	PA6 EFTEN	PA5 EFTEN	PA4 EFTEN	PA3 EFTEN	PA2 EFTEN	PA1 EFTEN	PA0 EFTEN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:16	PByEFTEN	引脚 PBy 模拟滤波使能位(y = 0...15) 0: 失能 EFT 1: 使能 EFT
15:0	PAyEFTEN	引脚 PAy 模拟滤波使能位(y = 0...15) 0: 失能 EFT 1: 使能 EFT

### 5.4.9 模拟滤波器配置寄存器 2 (AFIO\_EFT\_CFG2)

偏移地址: 0x1C

复位值: 0x00000000

31	30	29	28	27						17	16
PD15 EFTEN	PD14 EFTEN	PD13 EFTEN	PD12 EFTEN					Reserved			PD0 EFTEN
rw 15	rw 14	rw 13	rw 12								rw 0
PC15 EFTEN	PC14 EFTEN	PC13 EFTEN					Reserved				
rw	rw	rw									

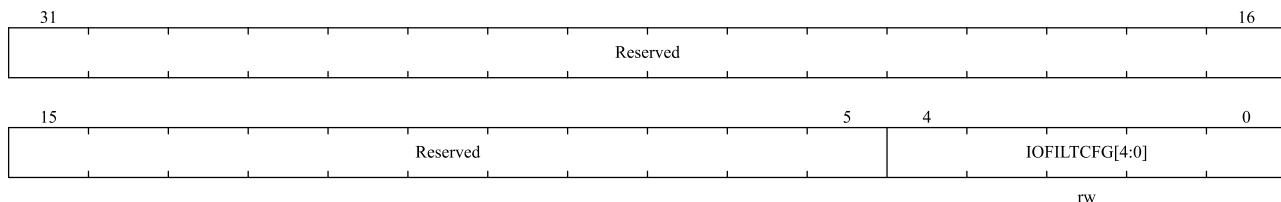
位域	名称	描述
31	PD15EFTEN	引脚 PD15 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
30	PD14EFTEN	引脚 PD14 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
29	PD13EFTEN	引脚 PD13 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
28	PD12EFTEN	引脚 PD12 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
27:17	保留	保留, 必须保持复位值。
16	PD0EFTEN	引脚 PD0 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
15	PC15EFTEN	引脚 PC15 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
14	PC14EFTEN	引脚 PC14 模拟滤波使能位 0: 失能 EFT 1: 使能 EFT
13	PC13EFTEN	引脚 PC13 模拟滤波使能位

位域	名称	描述
		0: 失能 EFT 1: 使能 EFT
12:0	保留	保留, 必须保持复位值。

### 5.4.10 数字毛刺滤波级（毛刺宽度）配置寄存器 (AFIO\_FILT\_CFG)

偏移地址: 0x20

复位值: 0x00000000

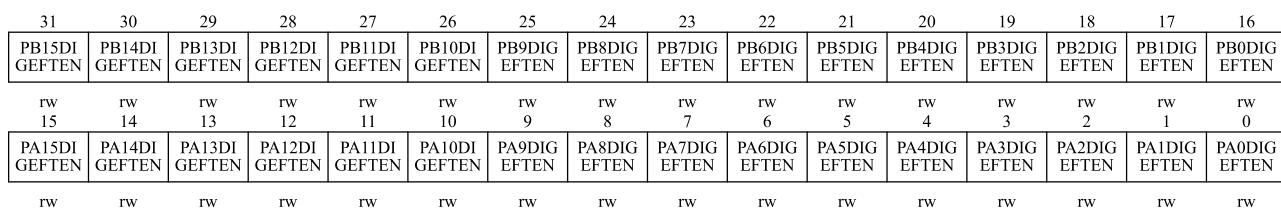


位域	名称	描述
31:5	保留	保留, 必须保持复位值。
4:0	IOFLITCFG[4:0]	滤波器级控制 00000:旁路滤波器 其他 : 以 AHB 时钟周期表示最小脉冲宽度的计数器值

### 5.4.11 数字毛刺滤波器配置寄存器 1 (AFIO\_DIGEFT\_CFG1)

偏移地址: 0x24

复位值: 0x00000000



位域	名称	描述
31:16	PByDGEFTEN	引脚 PBy 数字滤波使能位(y=0...15) 0: 失能 EFT 1: 使能 EFT
15:0	PAyDGEFTEN	引脚 PAy 数字滤波使能位(y=0...15) 0: 失能 EFT 1: 使能 EFT

## 5.4.12 数字毛刺滤波器配置寄存器 2 (AFIO\_DIGEFT\_CFG2)

偏移地址: 0x28

复位值: 0x00000000

31	30	29	28	27					17	16
PD15DI GEFTEN	PD14DI GEFTEN	PD13DI GEFTEN	PD12DI GEFTEN				Reserved		PD0DIG EFTEN	
RW 15	RW 14	RW 13	RW 12						RW 0	
PC15DI GEFTEN	PC14DI GEFTEN	PC13DI GEFTEN				Reserved				
RW	RW	RW								

位域	名称	描述
31	PD15DIEFETEN	引脚 PD15 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
30	PD14DIEFETEN	引脚 PD14 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
29	PD13DIEFETEN	引脚 PD13 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
28	PD12DIEFETEN	引脚 PD12 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
27:17	保留	保留, 必须保持复位值。
16	PD0DIEFETEN	引脚 PDO 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
15	PC15DIEFETEN	引脚 PC15 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
14	PC14DIEFETEN	引脚 PC14 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
13	PC13DIEFETEN	引脚 PC13 数字滤波使能位 0: 失能 EFT 1: 使能 EFT
12:0	保留	保留, 必须保持复位值。

## 6 中断和事件

### 6.1 嵌套向量中断控制器

#### 特征

- 53 个可屏蔽中断通道（不包括 Cortex-M4 的 16 个中断线）
- 16 个可编程优先级（使用 4 位中断优先级）
- 低延迟异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

嵌套向量中断控制器 (NVIC) 与处理器内核接口紧密相连，可实现低延迟中断处理，高效处理晚到的中断。嵌套向量中断控制器管理包括内核异常在内的中断。

#### 6.1.1 SysTick 校准值寄存器

系统节拍校准值固定为 16000。当系统节拍时钟设置为 16MHz (HCLK/8 的最大值) 时，会生成 1ms 的时间参考。

#### 6.1.2 中断和异常向量

表 6-1 向量表

位置	优先级	优先级类型	名称	说明	地址
-	-	-	-	保留	0x0000_0000
-	-3	固定	Reset	复位	0x0000_0004
-	-2	固定	NMI	不可屏蔽中断RCC时钟安全系统 (CSS) 连接到NMI向量	0x0000_0008
-	-1	固定	HardFault	所有类型的错误	0x0000_000C
-	0	可设置	MemManage	存储器管理	0x0000_0010
-	1	可设置	BusFault	预取指失败，存储器访问失败	0x0000_0014
-	2	可设置	UsageFault	未定义的指令或非法状态	0x0000_0018
-	-	-	-	保留	0x0000_001C- 0x0000_002B
-	3	可设置	SVCALL	通过SWI指令调用的系统服务	0x0000_002C

-	4	可设置	Debug Monitor	调试监控器	0x0000_0030
-	-	-	-	保留	0x0000_0034
-	5	可设置	PendSV	可挂起的系统服务	0x0000_0038
-	6	可设置	SysTick	系统嘀嗒定时器	0x0000_003C
0	7	可设置	WWDG	窗口看门狗中断	0x0000_0040
1	8	可设置	PVD	连到EXTI线16的电源电压检测(PVD)中断	0x0000_0044
2	9	可设置	RTC_TAMPER_STAMP	连到EXTI线18的入侵、时间戳和LSECSS中断	0x0000_0048
3	10	可设置	RTC_WKUP	连到EXTI线19的实时时钟(RTC)唤醒中断	0x0000_004C
4	11	可设置	FLASH	闪存全局中断	0x0000_0050
5	12	可设置	RCC	RCC全局中断	0x0000_0054
6	13	可设置	EXTI0	EXTI线0中断	0x0000_0058
7	14	可设置	EXTI1	EXTI线1中断	0x0000_005C
8	15	可设置	EXTI2	EXTI线2中断	0x0000_0060
9	16	可设置	EXTI3	EXTI线3中断	0x0000_0064
10	17	可设置	EXTI4	EXTI线4中断	0x0000_0068
11	18	可设置	DMA1_Channel1	DMA通道1全局中断	0x0000_006C
12	19	可设置	DMA1_Channel2	DMA通道2全局中断	0x0000_0070
13	20	可设置	DMA1_Channel3	DMA通道3全局中断	0x0000_0074
14	21	可设置	DMA1_Channel4	DMA通道4全局中断	0x0000_0078
15	22	可设置	DMA1_Channel5	DMA通道5全局中断	0x0000_007C
16	23	可设置	DMA1_Channel6	DMA通道6全局中断	0x0000_0080
17	24	可设置	DMA1_Channel7	DMA通道7全局中断	0x0000_0084
18	25	可设置	DMA1_Channel8	DMA通道8全局中断	0x0000_0088
19	26	可设置	ADC	ADC全局中断	0x0000_008C
20	27	可设置	MMU	MMU全局中断	0x0000_0090
21	28	可设置	COMP	连到EXTI线21/22/23的COMP1/COMP2	0x0000_0094

				/COMP3全局中断	
22	29	可设置	EXTI9_5	EXTI线[9:5]中断	0x0000_0098
23	30	可设置	TIM1_BRK	TIM1刹车中断	0x0000_009C
24	31	可设置	TIM1_UP	TIM1更新中断	0x0000_00A0
25	32	可设置	TIM1_TRG_COM	TIM1触发和通信中断	0x0000_00A4
26	33	可设置	TIM1_CC	TIM1捕获比较中断	0x0000_00A8
27	34	可设置	TIM2	TIM2全局中断	0x0000_00AC
28	35	可设置	TIM3	TIM3全局中断	0x0000_00B0
29	36	可设置	TIM4	TIM4全局中断	0x0000_00B4
30	37	可设置	I2C1_EV	I2C1事件中断	0x0000_00B8
31	38	可设置	I2C1_ER	I2C1错误中断	0x0000_00BC
32	39	可设置	I2C2_EV	I2C2事件中断	0x0000_00C0
33	40	可设置	I2C2_ER	I2C2错误中断	0x0000_00C4
34	41	可设置	SPI1	SPI1全局中断	0x0000_00C8
35	42	可设置	SPI2	SPI2全局中断	0x0000_00CC
36	43	可设置	USART1	USART1全局中断	0x0000_00D0
37	44	可设置	USART2	USART2全局中断	0x0000_00D4
38	45	可设置	UART3	UART3全局中断	0x0000_00D8
39	46	可设置	EXTI15_10	EXTI线[15:10]中断	0x0000_00DC
40	47	可设置	RTCAlarm	连到EXTI线17的RTC闹钟中断	0x0000_00E0
41	48	可设置	LPTIM_WKUP	连到EXTI线20的LPTIM唤醒中断	0x0000_00E4
42	49	可设置	TIM8_BRK	TIM8刹车中断	0x0000_00E8
43	50	可设置	TIM8_UP	TIM8更新中断	0x0000_00EC
44	51	可设置	TIM8_TRG_COM	TIM8触发和通信中断	0x0000_00F0
45	52	可设置	TIM8_CC	TIM8捕获比较中断	0x0000_00F4
46	53	可设置	UART4	UART4全局中断	0x0000_00F8
47	54	可设置	TIM5	TIM5全局中断	0x0000_00FC
48	55	可设置	TIM6	TIM6全局中断	0x0000_0100

49	56	可设置	CAN_TX	CAN发送中断	0x0000_0104
50	57	可设置	CAN_RX0	CAN接收0中断	0x0000_0108
51	58	可设置	CAN_RX1	CAN接收1中断	0x0000_010C
52	59	可设置	CAN_SCE	CAN的SCE中断	0x0000_0110

## 6.2 外部中断/事件控制器 (EXTI)

### 6.2.1 EXTI 简介

EXTI 控制器 IP 捕获中断/事件触发并将其转换为中断请求和事件脉冲。中断请求被发送到 NVIC 中断控制器。IP 有一个 APB 接口，通过它可以访问以下寄存器。

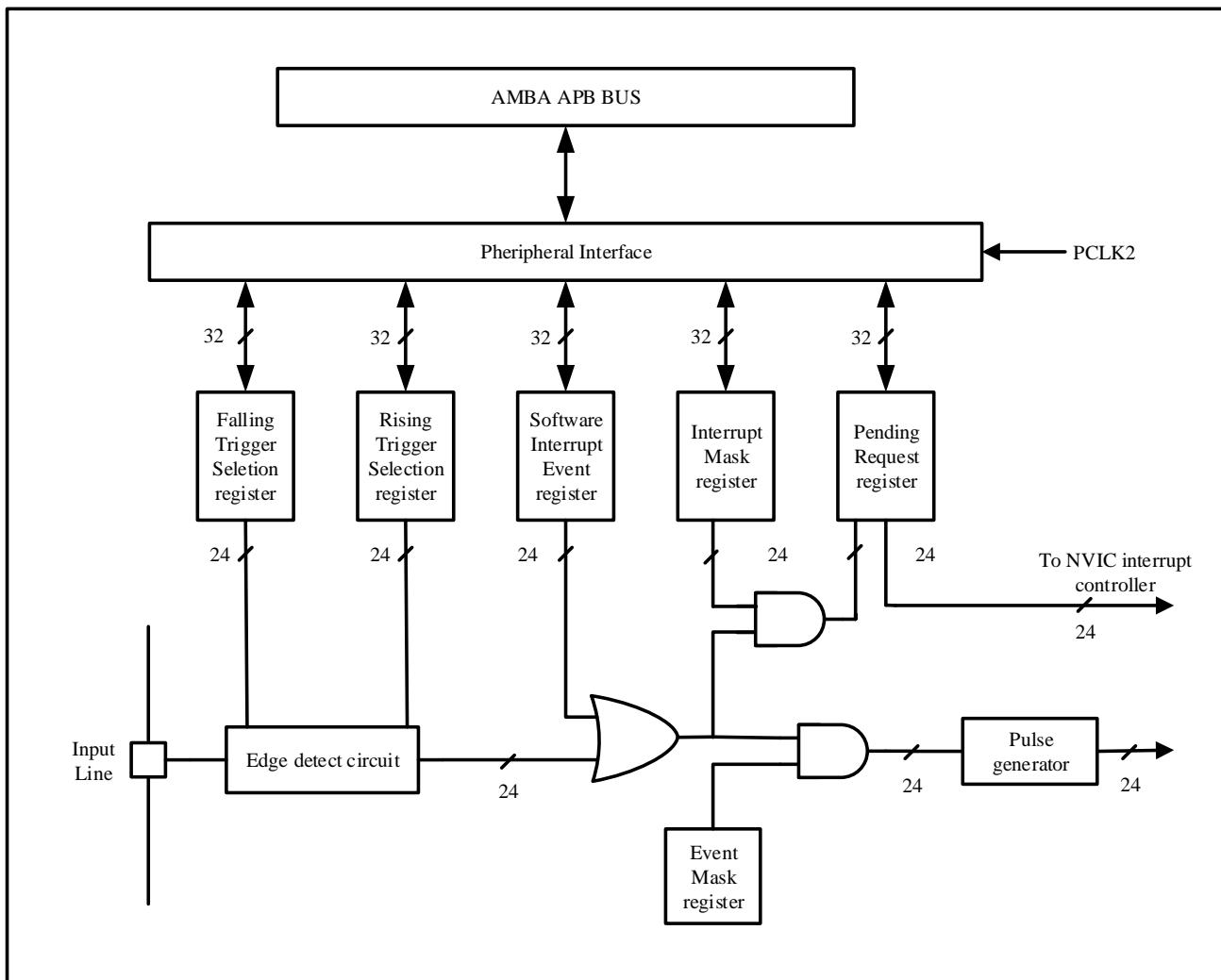
- 上升/下降沿触发选择寄存器
- 每个中断/事件线的专用屏蔽位寄存器
- 软件中断/事件配置寄存器
- 每个中断的中断挂起请求状态寄存器
- 时间戳选择寄存器

### 6.2.2 EXTI 主要特征

EXTI 控制器的主要特点如下：

- 支持 24 个软件中断/事件请求
- 每条输入线对应的中断/事件可独立配置触发或屏蔽
- 每条中断线都有一个独立的状态位
- 支持脉冲或挂起输入类型
- 支持三种触发事件：上升沿、下降沿或双边沿
- 唤醒退出低功耗模式

图 6-1 EXTI 功能框图



### 6.2.3 功能描述

EXTI 包含 24 条中断线，其中 16 条来自 I/O 引脚，8 条来自内部模块。要产生中断，必须配置外部中断控制器的 NVIC 中断通道去使能相应的中断线。通过边沿触发配置寄存器 EXTI\_RT\_CFG 和 EXTI\_FT\_CFG 选择上升沿、下降沿或双边沿触发事件的类型，并将'1'写入中断屏蔽寄存器 EXTI\_IMASK 的对应位，打开中断请求。当在外部中断线上检测到预设的边沿触发极性时，将产生中断请求并将相应的挂起位设置为'1'。在挂起寄存器的相应位中写入“1”将清除中断请求。

要生成事件，必须配置并启用相应的事件线。根据需要的边沿检测极性，设置上升沿/下降沿触发配置寄存器，并在事件屏蔽寄存器的相应位写入'1'，允许中断请求。当事件线上出现预设边沿时，将产生一个事件请求脉冲，并且相应的挂起位不会设置为“1”。

此外，通过在软件中断/事件寄存器中写入'1'，也可以由软件产生中断/事件请求。

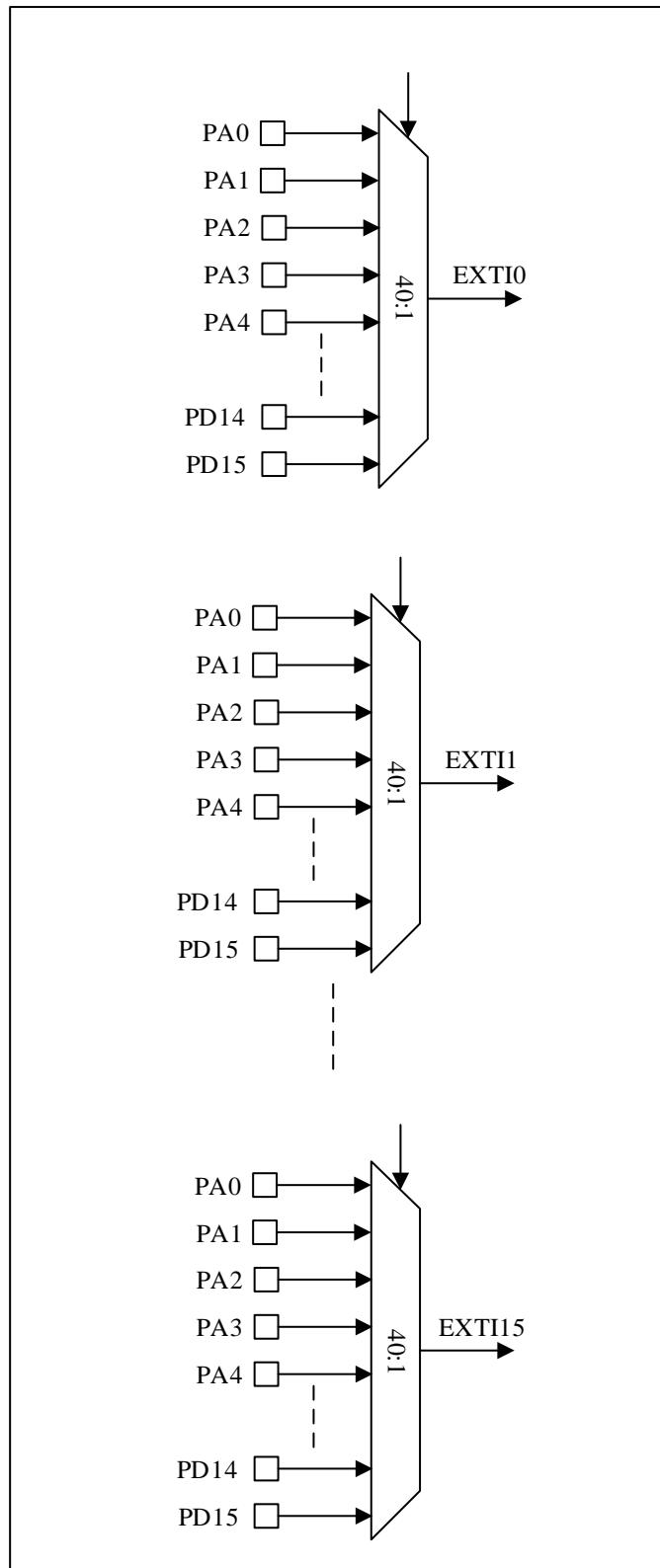
- 硬件中断配置，根据需要选择配置 24 条线作为中断源：

- ◆ 配置 24 条中断线的屏蔽位（EXTI\_IMASK）

- ◆ 配置所选中断线的触发配置位（EXTI\_RT\_CFG 和 EXTI\_FT\_CFG）
- ◆ 外部中断控制器对应的 NVIC 中断通道的使能和屏蔽位使 24 条中断线中的请求得到正确响应
- 硬件配置，根据需要选择配置 24 条线路作为事件源：
  - ◆ 配置 24 条事件线的屏蔽位（EXTI\_EMASK）
  - ◆ 配置所选事件线的触发配置位（EXTI\_RT\_CFG 和 EXTI\_FT\_CFG）
- 软件中断/事件配置，根据需要选择配置 24 条线作为软件中断/事件线：
  - ◆ 配置 24 个中断/事件线屏蔽位（EXTI\_IMASK,EXTI\_EMASK）
  - ◆ 配置软件中断事件寄存器（EXTI\_SWIE）的请求位

## 6.2.4 EXTI 线映射

图 6-2 外部中断 GPIO 映射



要通过 AFIO\_EXTI\_CFG 在 GPIO 线上配置外部中断/事件，必须首先启用 AFIO 时钟。通用 I/O 端口以上述方式连接到 16 条外部中断/事件线。其他 8 条 EXTI 线连接如下：

- EXTI 线 16 连接到 PVD 输出
- EXTI 线 17 连接到 RTC 铃声事件
- EXTI 线 18 连接到 RTC 时间戳事件、RTC 入侵事件和 LSECSS 事件
- EXTI 线 19 连接到 RTC 唤醒事件
- EXTI 线 20 连接到 LPTIM 唤醒中断
- EXTI 线 21 连接到 COMP1 输出
- EXTI 线 22 连接到 COMP2 输出
- EXTI 线 23 连接到 COMP3 输出

## 6.3 EXTI 寄存器

EXTI 基地址：0x40010400

### 6.3.1 EXTI 寄存器总览

表 6-2 EXTI 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	EXTI_IMASK	Reserved					IMASK[23:0]																										
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
004h	EXTI_EMASK	Reserved					EMASK[23:0]																										
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
008h	EXTI_RT_CFG	Reserved					RT_CFG[23:0]																										
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
00Ch	EXTI_FT_CFG	Reserved					FT_CFG[23:0]																										
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
010h	EXTI_SWIE	Reserved					SWIE[23:0]																										
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
014h	EXTI_PEND	Reserved					PEND23	PEND22	PEND21	PEND20	PEND19	PEND18	PEND17	PEND16	PEND15	PEND14	PEND13	PEND12	PEND11	PEND10	PEND9	PEND8	PEND7	PEND6	PEND5	PEND4	PEND3	PEND2	PEND1	PEND0			
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
018h	EXTI_TS_SEL	Reserved																											TSSEL[3:0]				
	Reset Value	0	0	0	0																												

### 6.3.2 EXTI 中断屏蔽寄存器 (EXTI\_IMASK)

偏移地址：0x00

复位值: 0x00000000

31	Reserved												24	23	22	21	20	19	18	17	16		
15	14	13	12	11	10	9	8	rw	7	rw	6	rw	5	rw	4	rw	3	rw	2	rw	1	rw	0
IMASK15	IMASK14	IMASK13	IMASK12	IMASK11	IMASK10	IMASK9	IMASK8	IMASK7	IMASK6	IMASK5	IMASK4	IMASK3	IMASK2	IMASK1	IMASK0								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:24	保留	保留, 必须保持复位值。
23:0	IMASKx	线 x 上的中断屏蔽位 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

### 6.3.3 EXTI 事件屏蔽寄存器 (EXTI\_EMASK)

偏移地址: 0x04

复位值: 0x00000000

31	Reserved												24	23	22	21	20	19	18	17	16		
15	14	13	12	11	10	9	8	rw	7	rw	6	rw	5	rw	4	rw	3	rw	2	rw	1	rw	0
EMASK15	EMASK14	EMASK13	EMASK12	EMASK11	EMASK10	EMASK9	EMASK8	EMASK7	EMASK6	EMASK5	EMASK4	EMASK3	EMASK2	EMASK1	EMASK0								
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:24	保留	保留, 必须保持复位值。
23:0	EMASKx	线 x 上的事件屏蔽位 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

### 6.3.4 EXTI 上升沿触发配置寄存器 (EXTI\_RT\_CFG)

偏移地址: 0x08

复位值: 0x00000000

31	Reserved												24	23	22	21	20	19	18	17	16		
15	14	13	12	11	10	9	8	rw	7	rw	6	rw	5	rw	4	rw	3	rw	2	rw	1	rw	0
RT_CFG15	RT_CFG14	RT_CFG13	RT_CFG12	RT_CFG11	RT_CFG10	RT_CFG9	RT_CFG8	RT_CFG7	RT_CFG6	RT_CFG5	RT_CFG4	RT_CFG3	RT_CFG2	RT_CFG1	RT_CFG0								

位域	名称	描述
31:24	保留	保留，必须保持复位值。
23:0	RT_CFGx	线 x 上的上升沿触发配置位 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）

### 6.3.5 EXTI 下降沿触发配置寄存器 (EXTI\_FT\_CFG)

偏移地址: 0x0C

复位值: 0x00000000

31								24	23	22	21	20	19	18	17	16
								FT_CFG23	FT_CFG22	FT_CFG21	FT_CFG20	FT_CFG19	FT_CFG18	FT_CFG17	FT_CFG16	
15	14	13	12	11	10	9	8	rw	rw							
FT_CFG15	FT_CFG14	FT_CFG13	FT_CFG12	FT_CFG11	FT_CFG10	FT_CFG9	FT_CFG8	FT_CFG7	FT_CFG6	FT_CFG5	FT_CFG4	FT_CFG3	FT_CFG2	FT_CFG1	FT_CFG0	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:24	保留	保留，必须保持复位值。
23:0	FT_CFGx	线 x 上的下降沿触发配置位 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）

### 6.3.6 EXTI 软件中断事件寄存器 (EXTI\_SWIE)

偏移地址: 0x10

复位值: 0x00000000

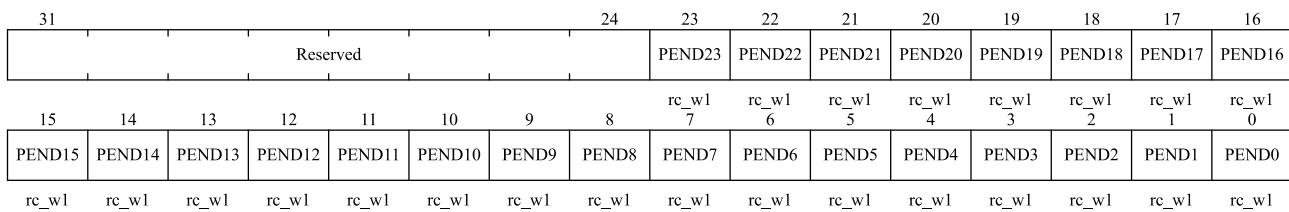
31								24	23	22	21	20	19	18	17	16
								SWIE23	SWIE22	SWIE21	SWIE20	SWIE19	SWIE18	SWIE17	SWIE16	
15	14	13	12	11	10	9	8	rw	rw							
SWIE15	SWIE14	SWIE13	SWIE12	SWIE11	SWIE10	SWIE9	SWIE8	SWIE7	SWIE6	SWIE5	SWIE4	SWIE3	SWIE2	SWIE1	SWIE0	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:24	保留	保留，必须保持复位值。
23:0	SWIEx	线 x 上的软件中断 当该位为'0'时，写'1'将设置 EXTI_PEND 中相应的挂起位。如果在 EXTI_IMASK 和 EXTI_EMASK 中允许产生该中断，此时将产生一个中断。 注：通过写入'1'清除 EXTI_PEND 的对应位，可以清除该位为'0'。

### 6.3.7 EXTI 中断请求挂起寄存器 (EXTI\_PEND)

偏移地址: 0x14

复位值: 0x00000000

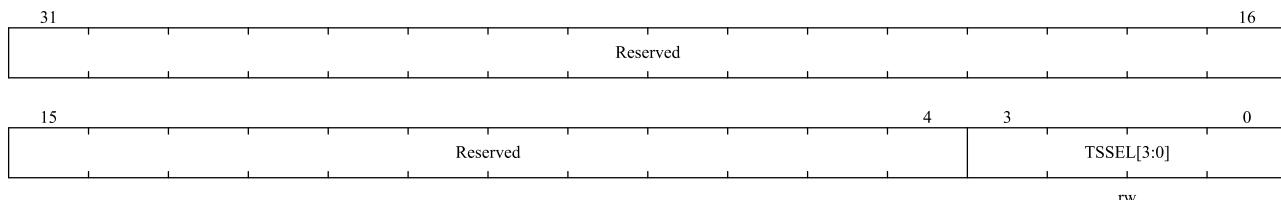


位域	名称	描述
31:24	保留	保留, 必须保持复位值。
23:0	PENDx	线 x 上的挂起位 0: 未产生挂起请求。 1: 产生了挂起的触发请求。 当外部中断线上发生选定的边沿触发事件时, 该位设置为 1。向该位写入“1”将其清零, 或改变边沿检测的极性以清零该位。

### 6.3.8 EXTI 时间戳触发源选择寄存器 (EXTI\_TS\_SEL)

偏移地址: 0x18

复位值: 0x00000000



位域	名称	描述
31:4	保留	保留, 必须保持复位值。
3:0	TSSEL[3:0]	选择外部中断输入作为时间戳事件的触发源 0: 选择 EXTI0 作为时间戳事件的触发源 1: 选择 EXTI1 作为时间戳事件的触发源 ..... 15: 选择 EXTI15 作为时间戳事件的触发源

## 7 DMA 控制器

### 7.1 简介

DMA 控制器总共可以访问 5 个 AHB 从机：Flash、SRAM、ADC、APB1 和 APB2。DMA 控制器由 CPU 控制以执行从源到目的的快速数据移动。配置完成后，无需 CPU 干预即可传输数据。因此，可以释放 CPU 用于其他计算/控制任务或节省整体系统功耗。

MCU 的主要架构是具有循环仲裁方案的多层次 AHB-Lite 总线结构。DMA 和 CPU 内核可以并行访问不同的从站，也可以顺序访问相同的从站。

DMA 控制器有 8 个逻辑通道。每个逻辑通道用于服务来自单个或多个外设的内存访问请求。内部仲裁器控制不同 DMA 通道的优先级。

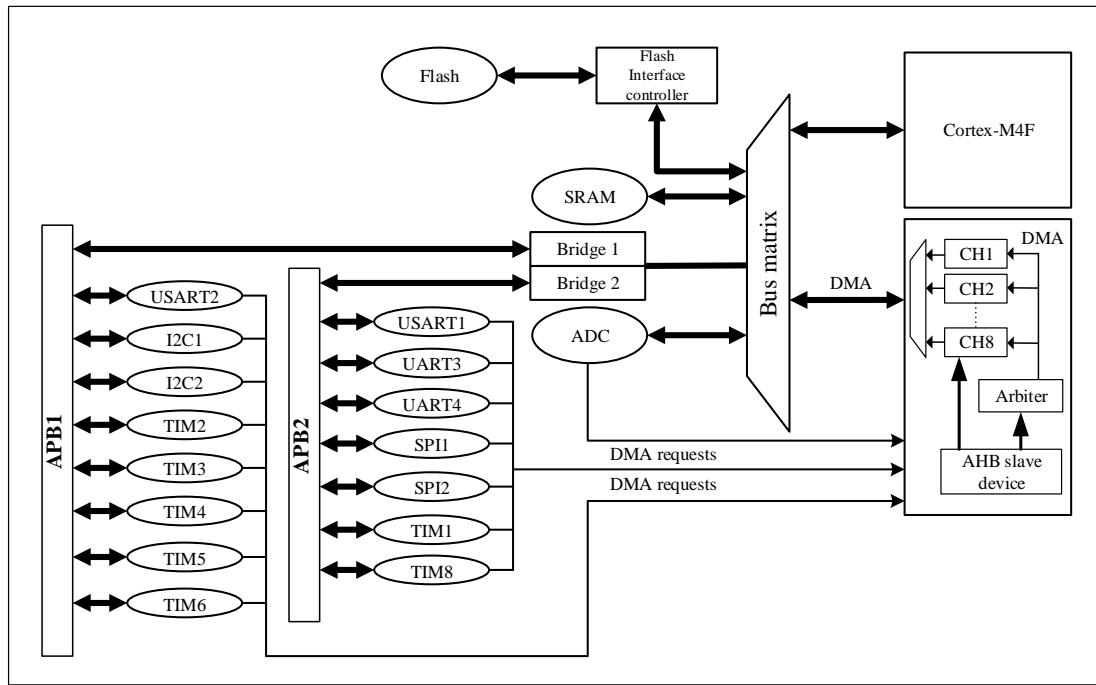
### 7.2 主要特性

DMA 主要特性：

- 8 个可独立配置的 DMA 通道。
- 每个 DMA 通道支持硬件请求和软件触发来启动传输，并由软件配置。
- 每个 DMA 通道都有专用的软件优先级（DMA\_CHCFGx.PRIOLVL[1:0]位，对应 4 个优先级），可以单独配置。具有相同软件优先级的通道将进一步比较硬件索引以确定最终优先级（索引号越低的通道优先级越高）。
- 可配置的源和目标大小。地址设置应与数据大小相对应。
- 每个通道可配置循环传输模式。
- 每个通道有 3 个独立的事件标志和中断（传输完成、半传输、传输错误）和 1 个全局中断标志（由 3 个事件的逻辑或设置）。
- 支持内存到内存、内存到外设和外设到内存三种传输类型。
- 共访问 5 个 AHB 从机：Flash、SRAM、ADC、APB1 和 APB2。
- 可配置数据传输数（0~65535）。

## 7.3 功能框图

图 7-1 DMA 框图



## 7.4 功能描述

DMA 控制器和 Cortex<sup>TM</sup>-M4F 内核共享相同的系统数据总线。当 CPU 和 DMA 同时访问同一个目标（RAM 或外设）时，DMA 请求会暂停 CPU 访问系统总线几个周期，由总线仲裁器进行循环调度。这允许 CPU 获得至少一半的系统总线（内存或外围设备）带宽。

### 7.4.1 DMA 操作

DMA 请求可以由硬件外设或软件触发，DMA 控制器根据通道的优先级处理请求。根据配置的传输地址和位宽从源地址读取数据，然后将读取的数据存储在目的地址空间中。一次操作后，控制器计算剩余传输次数，并更新下一次传输的源地址和目的地址（只有内部源地址和目的地址会被更新，用户不需要知道，这不会显示在寄存器中）。

每个 DMA 数据传输包括三个操作：

- 数据访问：根据传输方向确定源地址（DMA\_PADDRx 或 DMA\_MADDRx），从源地址读取数据。
- 数据存储：根据传输方向确定目的地址（DMA\_PADDRx 或 DMA\_MADDRx），将读取的数据存储到目的地址空间。
- 计算未完成操作的数量，对 DMA\_TXNUMx 寄存器进行减量操作，更新下一个操作的源地址和目的地址。

## 7.4.2 通道优先级和仲裁器

DMA 使用仲裁策略来处理来自不同通道的多个请求。每个通道的优先级可在通道控制寄存器(DMA\_CHCFGx)中进行编程。

4 个优先级：

- ◆ 非常高优先级
- ◆ 高优先级
- ◆ 中优先级
- ◆ 低优先级

默认情况下，如果编程的优先级相同，则索引较低的通道具有较高的优先级。

对于内存到内存的传输，在 4 次传输操作后进行重新仲裁。

对于与外设相关的传输，每次传输操作后都会进行重新仲裁。

## 7.4.3 DMA 通道和传输数量

每个通道都可以在指定地址的外设寄存器和内存地址之间进行 DMA 传输。DMA 传输的数据数量是可编程的，最大支持值为 65535。DMA\_TXNUM 寄存器在每次传输后递减。

## 7.4.4 可编程的数据位宽

外设和内存传输数据位宽支持字节、半字和字，可以通过 DMA\_CHCFGx.PSIZE 和 DMA\_CHCFGx.MSIZE 进行编程。

当 DMA\_CHCFGx.PSIZE 和 DMA\_CHCFGx.MSIZE 不同时，DMA 模块根据表 7-1 对齐数据。

表 7-1 可编程的数据宽度和大小端操作(当 PINC = MINC = 1)

Source width (bit)	Destina-tion width (bit)	Number of transfer (bit)	Source: Address / data	Transfer operations (R: Read,W: Write)	Destination: Address / data
8	8	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: R B0 [7:0] @0x0, W B0 [7:0] @0x0 2: R B1 [7:0] @0x1, W B1 [7:0] @0x1 3: R B2 [7:0] @0x2, W B2 [7:0] @0x2 4: R B3 [7:0] @0x3, W B3 [7:0] @0x3	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3
8	16	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: R B0 [7:0] @0x0, W 00B0 [15:0] @0x0 2: R B1 [7:0] @0x1, W 00B1 [15:0] @0x2 3: R B2 [7:0] @0x2, W 00B2 [15:0] @0x4 4: R B3 [7:0] @0x3, W 00B3 [15:0] @0x6	0x0 / 00B0 0x2 / 00B1 0x4 / 00B2 0x6 / 00B3
8	32	4	0x0 / B0 0x1 / B1 0x2 / B2	1: R B0 [7:0] @0x0, W 000000B0 [31:0] @0x0 2: R B1 [7:0] @0x1, W 000000B1 [31:0] @0x4 3: R B2 [7:0] @0x2, W 000000B2 [31:0] @0x8	0x0 / 000000B0 0x4 / 000000B1 0x8 / 000000B2

Source width (bit)	Destina- tion width (bit)	Number of transfer (bit)	Source: Address / data	Transfer operations (R: Read,W: Write)	Destination: Address / data
			0x3 / B3	4: R B3 [7:0] @0x3, W 000000B3 [31:0] @0xC	0xC / 000000B3
16	8	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W B0 [7:0] @0x0 2: R B3B2 [15:0] @0x2, W B2 [7:0] @0x1 3: R B5B4 [15:0] @0x4, W B4 [7:0] @0x2 4: R B7B6 [15:0] @0x6, W B6 [7:0] @0x3	0x0 / B0 0x1 / B2 0x2 / B4 0x3 / B6
16	16	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W B1B0 [15:0] @0x0 2: R B3B2 [15:0] @0x2, W B3B2 [15:0] @0x2 3: R B5B4 [15:0] @0x4, W B5B4 [15:0] @0x4 4: R B7B6 [15:0] @0x6, W B7B6 [15:0] @0x6	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6
16	32	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: R B1B0 [15:0] @0x0, W 0000B1B0 [31:0] @0x0 2: R B3B2 [15:0] @0x2, W 0000B3B2 [31:0] @0x4 3: R B5B4 [15:0] @0x4, W 0000B5B4 [31:0] @0x8 4: R B7B6 [15:0] @0x6, W 0000B7B6 [31:0] @0xC	0x0 / 0000B1B0 0x4 / 0000B3B2 0x8 / 0000B5B4 0xC / 0000B7B6
32	8	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDDBC	1: R B3B2B1B0 [31:0] @0x0, W B0 [7:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B4 [7:0] @0x1 3: R BBBAB9B8 [31:0] @0x8, W B8 [7:0] @0x2 4: R BFBEBDDBC [31:0] @0xC, W BC [7:0] @0x3	0x0 / B0 0x1 / B4 0x2 / B8 0x3 / BC
32	16	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDDBC	1: R B3B2B1B0 [31:0] @0x0, W B1B0 [15:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B5B4 [15:0] @0x2 3: R BBBAB9B8 [31:0] @0x8, W B9B8 [15:0] @0x4 4: R BFBEBDDBC [31:0] @0xC, W BDBC [15:0] @0x6	0x0 / B1B0 0x2 / B5B4 0x4 / B9B8 0x6 / BDBC
32	32	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDDBC	1: R B3B2B1B0 [31:0] @0x0, W B3B2B1B0 [31:0] @0x0 2: R B7B6B5B4 [31:0] @0x4, W B7B6B5B4 [31:0] @0x4 3: R BBBAB9B8 [31:0] @0x8, W BBBAB9B8 [31:0] @0x8 4: R BFBEBDDBC [31:0] @0xC, W BFBEBDDBC [31:0] @0xC	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDDBC

注意：

DMA 总是向 HWDATA[31:0]提供完整的 32 位数据，无论它是什么目标大小（HSIZE 仍然遵循设备支持字节/半字操作的目标大小设置）。它提供的 HWDATA[31:0]遵循以下规则：

- 当源大小小于目标大小时，DMA 用 0 填充 MSB，直到它们的大小匹配并将其复制为 32 位。例如，源是 8 位数据 0x55，目标大小是 16 位。DMA 用 0 填充源数据使其成为 16 位 0x0055，然后将其复制为 32 位数据 0x0055\_0055 并提供给 HWDATA[31:0]；（如果目标大小为 32 位，则 DMA 只会用 0 填充源数据）。
- 当源大小大于或等于目标大小且小于 32 位时，DMA 将源数据复制到 32 位数据。例如，源数据为 8 位数据 0x1F，HWDATA[31:0] = 0x1F1F\_1F1F。如果源数据是 16 位数据 0x2345，则 HWDATA[31:0] = 0x2345\_2345。

这保证了仅支持字操作的外设不会产生总线错误，并且所需的数据仍然可以通过额外的位（即 0 填充）移动到我们想要的位置。如果用户想要配置一个 8 位寄存器但与 32 位地址边界对齐，则源大小应设置为 8 位，目标大小应设置为 32 位，因此额外的位将用 0 填充。

### 7.4.5 外设/内存地址递增

DMA\_CHCFGx.PINC 和 DMA\_CHCFGx.MINC 分别控制外设地址和内存地址是否使能自动递增模式。软件在传输过程中不能（可以读）写地址寄存器。

- 在自动递增模式下，下一个要传输的地址在每次传输后根据数据位宽（1、2 或 4）自动增加。第一次传输的地址存储在 DMA\_PADDRx 或 DMA\_MADDRx 寄存器中。
- 在固定模式下，地址始终固定为初始地址。

在传输结束时（即传输计数变为 0），将根据当前是否工作于循环模式进行不同的处理。

- 在非循环模式下，DMA 在传输完成后停止。要开始新的 DMA 传输，需要在禁用 DMA 通道的情况下重写 DMA\_TXNUMx 寄存器中的传输数量。
- 在循环模式下，在传输结束时，DMA\_TXNUMx 寄存器的内容会自动重新加载其初始值，并且当前内部外设或内存地址也会重新加载 DMA\_PADDRx 或 DMA\_MADDRx 寄存器设置的初始基址。

### 7.4.6 通道配置流程

详细配置流程如下：

1. 配置中断屏蔽位，1：启用中断，0：禁用中断。
2. 配置传输方向。
3. 配置通道外设地址和内存地址。
4. 配置通道优先级，0：最低，3：最高。
5. 配置外设和内存地址增量。
6. 配置通道传输块大小。
7. 如有必要，配置循环模式。
8. 如果是存储器到存储器，配置 MEM2MEM 模式（注：要配置 DMA 工作在 M2M 模式，用户需要将相应的通道选择值设置为保留值，例如 53）。
9. 在通道 1~8 上重复第 1~8 步。
10. 最后使能相应通道。

如果使用软件提供中断服务，则软件必须查询中断状态寄存器以检查发生了哪个中断（软件需要向中断标志清除位写 1 来清除相应的中断）。在使能通道之前，应清除该通道对应的所有中断。

如果中断是传输完成中断，软件可以配置下一次传输，或者向用户报告该通道传输完成。

### 7.4.7 流量控制

支持三种主要的流量控制：

- 存储器到存储器
- 存储器到外设
- 外设到存储器

流控制由每个 DMA 通道配置寄存器中的两个寄存器位控制。流控制用于控制 DMA 通道的源/目标和方向。

表 7-2 流量控制表

DMA_CHCFGx.MEM2MEM	DMA_CHCFGx.DIR	Source	Destination	Transfer
1	x	Memory	Memory	AHB read to AHB write, can do back2back transfer
0	1	Memory	AHB Peripheral	AHB read to AHB write, single transfer
			APB Peripheral	AHB read to APB write, single transfer
0	0	AHB Peripheral	Memory	AHB read to AHB write, single transfer
		APB Peripheral		APB read to AHB write, single transfer

#### 7.4.8 循环模式

循环模式用于处理循环缓冲区和连续数据传输(如 ADC 扫描模式)。DMA\_CHCFGx.CIRC 用于启用此功能。激活循环模式时, 如果要传输的数据数变为 0, 则在配置通道时会自动恢复到初始值, 继续进行 DMA 操作。

如果用户想关闭循环模式, 用户需要向 DMA\_CHCFGx.CHEN 写入 0 以禁用 DMA 通道, 然后向 DMA\_CHCFGx.CIRC 写入 0 (当 DMA\_CHCFGx.CHEN 为 1 时, DMA\_CHCFGx 寄存器中的其他位不能被改写)。

#### 7.4.9 错误管理

对保留地址区域的 DMA 访问会导致 DMA 传输错误。发生错误时, 设置传输错误标志, 硬件自动清除当前 DMA 通道使能位 (DMA\_CHCFGx.CHEN), 通道操作停止。如果在 DMA\_CHCFGx 寄存器中设置了传输错误中断使能位, 则会产生中断。

#### 7.4.10 中断

- 传输完成中断:

通道数据传输完成时会产生中断。中断是一个电平信号。每个通道都有其专用的中断、中断屏蔽控制和中断状态位。当中断标志清除位被设置时, 中断状态位被清除。

- 半传输中断：

当传输了一半的通道数据时会产生中断。中断是一个电平信号。每个通道都有其专用的中断、中断屏蔽控制和中断状态位。当中断标志清除位被设置时，中断状态位被清除。

- 传输错误中断：

总线返回错误时产生中断。中断是一个电平信号。每个通道都有其专用的中断、中断屏蔽控制和中断状态位。当中断标志清除位被设置时，中断状态位被清除。

表 7-3 DMA 中断请求

中断事件	事件标志位	使能控制位
半传输	HTXF	HTXIE
传输完成	TXCF	TXCIE
传输错误	ERRF	ERRIE

### 7.4.11 DMA 请求映射

总共有来自所有外设的 52 个 DMA 请求。为了获得更好的支持和完全的灵活性，可以使用寄存器位来选择将哪个 DMA 请求映射到哪个 DMA 通道。下表显示了外设的 DMA 请求到 DMA 控制器的 DMA 通道的映射方案。

表 7-4 DMA 请求映射

DMA channel select	Peripheral DMA request
sel = 0	ADC
sel = 1	Usart1_tx
sel = 2	Usart1_rx
sel = 3	Usart2_tx
sel = 4	Usart2_rx
sel = 5	Uart3_tx
sel = 6	Uart3_rx
sel = 7	Uart4_tx
sel = 8	Uart4_rx
sel = 9	Spi1_tx
sel = 10	Spi1_rx

sel = 11	Spi2_tx
sel = 12	Spi2_rx
sel = 13	I2c1_tx
sel = 14	I2c1_rx
sel = 15	I2c2_tx
sel = 16	I2c2_rx
sel = 17	Tim1_ch1
sel = 18	Tim1_ch2
sel = 19	Tim1_ch3
sel = 20	Tim1_ch4
sel = 21	Tim1_com
sel = 22	Tim1_up
sel = 23	Tim1_trig
sel = 24	Tim2_ch1
sel = 25	Tim2_ch2
sel = 26	Tim2_ch3
sel = 27	Tim2_ch4
sel = 28	Tim2_up
sel = 29	Tim3_ch1
sel = 30	Tim3_ch3
sel = 31	Tim3_ch4
sel = 32	Tim3_up
sel = 33	Tim3_trig
sel = 34	Tim4_ch1

sel = 35	Tim4_ch2
sel = 36	Tim4_ch3
sel = 37	Tim4_up
sel = 38	Tim5_ch1
sel = 39	Tim5_ch2
sel = 40	Tim5_ch3
sel = 41	Tim5_ch4
sel = 42	Tim5_up
sel = 43	Tim5_trg
sel = 44	Tim6
sel = 45	Tim8_ch1
sel = 46	Tim8_ch2
sel = 47	Tim8_ch3
sel = 48	Tim8_ch4
sel = 49	Tim8_com
sel = 50	Tim8_up
sel = 51	Tim8_trg

## 7.5 DMA 寄存器

### 7.5.1 DMA 寄存器总览

表 7-5 DMA 寄存器总览

Offset	Register	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
000h	DMA_INTSTS	CERRF8	0	HTXF8	30	CTXCF8	0	TXCF8	29	CGLBF8	0	GLBF8	28	CERRF7	0	ERRF7	27	CHTF7	0	HTXF7	26	CTXCF7	0	TXCF7	25	CGLBF7	0	GLBF7	24	CTXCF6	0	TXCF6	21
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
004h	DMA_INTCLR	CERRF6	0	HTXF6	23	CTXCF6	0	TXCF6	20	CGLBF6	0	GLBF6	19	CERRF5	0	ERRF5	19	CHTF5	0	HTXF5	18	CTXCF5	0	TXCF5	17	CGLBF5	0	GLBF5	16	CERRF4	0	ERRF4	15
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	MEM2MEM	PRIOLVL[1]:	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN
008h	DMA_CHCFG1																	0	0	0	0	0	0	0	0	0	0	0	0	
	Reset Value																													
00Ch	DMA_TXNUM1																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
010h	DMA_PADDR1																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
014h	DMA_MADDR1																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
018h	DMA_CHSEL1																													
	Reset Value																													
01Ch	DMA_CHCFG2																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
020h	DMA_TXNUM2																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
024h	DMA_PADDR2																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
028h	DMA_MADDR2																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
02Ch	DMA_CHSEL2																													
	Reset Value																													
030h	DMA_CHCFG3																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
034h	DMA_TXNUM3																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
038h	DMA_PADDR3																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
03Ch	DMA_MADDR3																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
040h	DMA_CHSEL3																													
	Reset Value																													
044h	DMA_CHCFG4																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
048h	DMA_TXNUM4																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
04Ch	DMA_PADDR4																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
050h	DMA_MADDR4																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
054h	DMA_CHSEL4																													
	Reset Value																													
058h	DMA_CHCFG5																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
05Ch	DMA_TXNUM5																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
060h	DMA_PADDR5																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
064h	DMA_MADDR5																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
068h	DMA_CHSEL5																													
	Reset Value																													
06Ch	DMA_CHCFG6																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
070h	DMA_TXNUM6																													
	Reset Value																		0	0	0	0	0	0	0	0	0	0	0	0
074h	DMA_PADDR6																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
078h	DMA_MADDR6																													
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
07Ch	DMA_CHSEL6																													
	Reset Value																													

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
080h	DMA_CHCFG7	Reserved															MEM2MEM	PRIOLVL[1:0]	MSIZE[1:0]	PSIZE[1:0]	MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN	0				
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			
084h	DMA_TXNUM7	Reserved															NDTX[15:0]																
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			
088h	DMA_PADDR7	ADDR[31:0]																															
	Reset Value	0 0																															
08Ch	DMA_MADDR7	ADDR[31:0]																															
	Reset Value	0 0																															
090h	DMA_CHSEL7	Reserved															CH_SEL[5:0]																
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			
094h	DMA_CHCFG8	Reserved															CH_SEL[5:0]																
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			
098h	DMA_TXNUM8	Reserved															NDTX[15:0]																
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			
09Ch	DMA_PADDR8	ADDR[31:0]																															
	Reset Value	0 0																															
0A0h	DMA_MADDR8	ADDR[31:0]																															
	Reset Value	0 0																															
0A4h	DMA_CHSEL8	Reserved															CH_SEL[5:0]																
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0			

## 7.5.2 DMA 中断状态寄存器 (DMA\_INTSTS)

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
ERRF8	HTXF8	TXCF8	GLBF8	ERRF7	HTXF7	TXCF7	GLBF7	ERRF6	HTXF6	TXCF6	GLBF6	ERRF5	HTXF5	TXCF5	GLBF5
r 15	r 14	r 13	r 12	r 11	r 10	r 9	r 8	r 7	r 6	r 5	r 4	r 3	r 2	r 1	r 0
ERRF4	HTXF4	TXCF4	GLBF4	ERRF3	HTXF3	TXCF3	GLBF3	ERRF2	HTXF2	TXCF2	GLBF2	ERRF1	HTXF1	TXCF1	GLBF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

位域	名称	描述
31/27/23/19/15/11/7/3	ERRFx	通道 x(x = 1...8)的传输错误标志。 发生传输错误时，硬件设置该位。该位由软件通过向 DMA_INTCLR.CERRFx 位写 1 清零。 0: 通道 x 上没有发生传输错误。 1: 通道 x 上发生传输错误。
30/26/22/18/14/10/6/2	HTXFx	通道 x(x = 1...8)的半传输标志。 当半传输完成时，硬件设置该位。该位由软件通过向 DMA_INTCLR.CHTXFx 位写 1 清零。 0: 通道 x 上的半传输尚未完成。 1: 通道 x 上的半传输已完成。
29/25/21/17/13/9/5/1	TXCFx	通道 x(x = 1...8)的传输完成标志。 当传输完成时，硬件设置该位。该位由软件通过向 DMA_INTCLR.CTXCFx 位写 1 清零。 0: 通道 x 上的传输尚未完成。 1: 通道 x 上的传输已完成。
28/24/20/16/12/8/4/0	GLBFx	通道 x(x = 1...8)的全局标志。

位域	名称	描述
		<p>当该通道中发生任何中断事件时，硬件设置该位。该位由软件通过向 DMA_INTCLR.CGLBFx 位写 1 清零。</p> <p>0：通道 x 上没有发生传输错误、半传输或传输完成事件。</p> <p>1：通道 x 上发生传输错误、半传输或传输完成事件之一。</p>

### 7.5.3 DMA 中断标志清除寄存器 (DMA\_INTCLR)

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
CERRF8	CHTF8	CTXCF8	CGLBF8	CERRF7	CHTF7	CTXCF7	CGLBF7	CERRF6	CHTF6	CTXCF6	CGLBF6	CERRF5	CHTF5	CTXCF5	CGLBF5
w 15	w 14	w 13	w 12	w 11	w 10	w 9	w 8	w 7	w 6	w 5	w 4	w 3	w 2	w 1	w 0
CERRF4	CHTF4	CTXCF4	CGLBF4	CERRF3	CHTF3	CTXCF3	CGLBF3	CERRF2	CHTF2	CTXCF2	CGLBF2	CERRF1	CHTF1	CTXCF1	CGLBF1
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

位域	名称	描述
31/27/23/19/15/11/7/3	CERRFx	<p>清除通道 x(x = 1...8) 的传输错误标志。</p> <p>软件可以设置该位来清除相应通道的 ERRF。</p> <p>0：无动作。</p> <p>1：复位相应通道的 DMA_INTSTS.ERRF 位。</p>
30/26/22/18/14/10/6/2	CHTFx	<p>清除通道 x(x = 1...8) 的半传输标志。</p> <p>软件可以设置该位来清除相应通道的 HTXF。</p> <p>0：无动作。</p> <p>1：复位相应通道的 DMA_INTSTS.HTXF 位。</p>
29/25/21/17/13/9/5/1	CTXCFx	<p>清除通道 x(x = 1...8) 的传输完成标志。</p> <p>软件可以设置该位来清除相应通道的 TXCF。</p> <p>0：无动作。</p> <p>1：复位相应通道的 DMA_INTSTS.TXCF 位。</p>
28/24/20/16/12/8/4/0	CGLBFx	<p>清除通道 x(x = 1...8) 的全局事件标志。</p> <p>软件可以设置该位来清除相应通道的 GLBF。</p> <p>0：无动作。</p> <p>1：复位相应通道的 DMA_INTSTS.GLBF 位。</p>

### 7.5.4 DMA 通道 x 配置寄存器 (DMA\_CHCFGx)

注: x 为通道号, x = 1...8

偏移地址: 0x08+20 \* (x-1)

复位值: 0x0000 0000

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MEM2MEM	PRIOLVL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	ERRIE	HTXIE	TXCIE	CHEN
	rw	rw		rw		rw		rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:15	Reserved	保留，必须保持复位值。
14	MEM2MEM	存储器到存储器模式。 当通道尚未使能时，软件可以将此通道配置为存储器到存储器传输。 0: 存储器和外设之间的通道传输。 1: 通道设置为存储器到存储器间的传输。
13:12	PRIOLVL[1:0]	通道优先级。 当通道未使能时，软件可以编程通道优先级。 00: 低 01: 中 10: 高 11: 非常高
11:10	MSIZE[1:0]	存储器数据大小。 软件可以配置从/向存储器地址读取/写入的数据大小。 00: 8 位 01: 16 位 10: 32 位 11: 保留
9:8	PSIZE[1:0]	外设数据大小。 软件可以配置从/向外设地址读取/写入的数据大小。 00: 8 位 01: 16 位 10: 32 位 11: 保留
7	MINC	存储器地址递增模式。 软件可以使能/禁能存储器地址递增模式。 0: 内存地址不会随着每次传输而递增。 1: 内存地址随着每次传输而递增。
6	PINC	外设地址增量模式。 软件可以使能/禁能外设地址递增模式。 0: 外设地址不会随着每次传输而递增。 1: 外设地址随每次传输而递增。
5	CIRC	循环模式。 软件可以设置/清除该位。 0: 经过一轮传输后通道停止。 1: 通道配置为循环模式。
4	DIR	数据传输方向

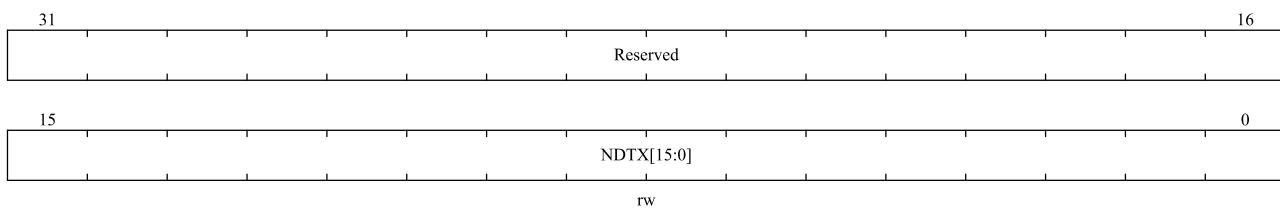
位域	名称	描述
		软件可以设置/清除该位。 0: 从外设到存储器的数据传输 1: 从存储器到外设的数据传输。
3	ERRIE	传输错误中断使能。 软件可以使能/禁能传输错误中断。 0: 禁止通道 x 的传输错误中断。 1: 使能通道 x 的传输错误中断。
2	HTXIE	半传输中断使能。 软件可以使能/禁能半传输中断。 0: 禁止通道 x 的半传输中断。 1: 使能通道 x 的半传输中断。
1	TXCIE	传输完成中断使能。 软件可以使能/禁能传输完成中断。 0: 禁止通道 x 的传输完成中断。 1: 使能通道 x 的传输完成中断。
0	CHEN	通道使能。 软件可以设置/复位该位。 0: 禁用通道。 1: 使能通道。

### 7.5.5 DMA 通道 x 传输数量寄存器 (DMA\_TXNUMx)

注: x 为通道号, x = 1...8

偏移地址: 0x0c+20 \* (x-1)

复位值: 0x0000 0000



位域	名称	描述
31:16	Reserved	保留, 必须保持复位值。
15:0	NDTX	数据传输数量。 要传输的数据数量 (0~65535)。软件可以在通道禁用时写入/读出传输数量, 并且通道使能后该位为只读。相应的 DMA 通道每次成功传输后, 该寄存器就会减 1。如果使能循环模式, 它会在达到零时自动重新加载预设值。否则它将保持为零并复位通道使能位。

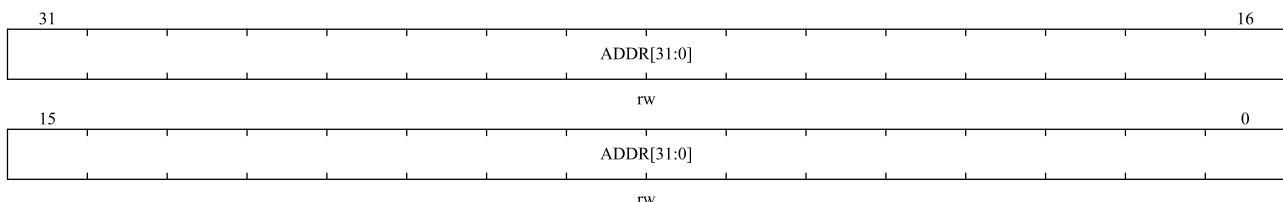
## 7.5.6 DMA 通道 x 外设基址寄存器 (DMA\_PADDRx)

注：x 为通道号，x = 1...8

偏移地址：0x10+20 \* (x-1)

复位值：0x0000 0000

只有在禁用通道(DMA\_CHCFGx.CHEN = 0)时才能写该寄存器。



位域	名称	描述
31:0	ADDR	外设基址。 DMA 读取/写入的外设起始地址。 地址的递增由 DMA_CHCFGx.PSIZE 决定。DMA_CHCFGx.PSIZE 等于‘01’， DMA 忽略 PADDR 的第 0 位，如果 DMA_CHCFGx.PSIZE 等于‘10’，DMA 将 忽略 PADDR 的第 0 位和第 1 位。

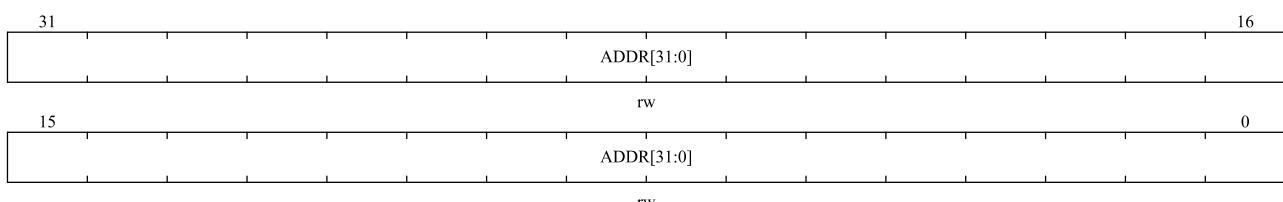
## 7.5.7 DMA 通道 x 存储器基址寄存器 (DMA\_MADDRx)

注：x 为通道号，x = 1...8

偏移地址：0x14+20 \* (x-1)

复位值：0x0000 0000

只有在禁用通道(DMA\_CHCFGx.CHEN = 0)时才能写该寄存器。



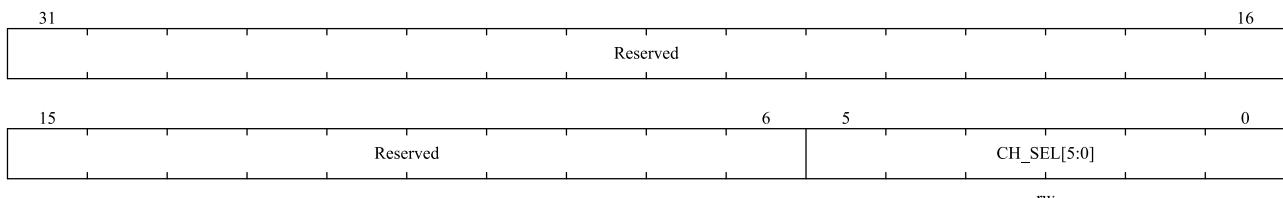
位域	名称	描述
31:0	ADDR	存储器基址。 DMA 读取/写入的存储器起始地址。 地址的递增由 DMA_CHCFGx.MSIZE 决定。DMA_CHCFGx.MSIZE 等于‘01’， DMA 忽略 MADDR 的第 0 位，如果 DMA_CHCFGx.MSIZE 等于‘10’，DMA 将 忽略 MADDR 的第 0 位和第 1 位。

## 7.5.8 DMA 通道 x 请求选择寄存器 (DMA\_CHSELx)

注: x 为通道号, x = 1...8

偏移地址: 0x18+20 \* (x-1)

复位值: 0x0000 0000



位域	名称	描述
31:6	Reserved	保留, 必须保持复位值。
5:0	CH_SEL[5:0]	DMA 通道请求选择 0x00: ADC 0x01: USART1_TX 0x02: USART1_RX 0x03: USART2_TX 0x04: USART2_RX 0x05: UART3_TX 0x06: UART3_RX 0x07: UART4_TX 0x08: UART4_RX 0x09: SPI1_TX 0x0A: SPI1_RX 0x0B: SPI2_TX 0x0C: SPI2_RX 0x0D: I2C1_TX 0x0E: I2C1_RX 0x0F: I2C2_TX 0x10: I2C2_RX 0x11: TIM1_CH1 0x12: TIM1_CH2 0x13: TIM1_CH3 0x14: TIM1_CH4 0x15: TIM1_COM 0x16: TIM1_UP 0x17: TIM1_TRIG 0x18: TIM2_CH1 0x19: TIM2_CH2 0x1A: TIM2_CH3

位域	名称	描述
		0x1B: TIM2_CH4 0x1C: TIM2_UP 0x1D: TIM3_CH1 0x1E: TIM3_CH3 0x1F: TIM3_CH4 0x20: TIM3_UP 0x21: TIM3_TRIG 0x22: TIM4_CH1 0x23: TIM4_CH2 0x24: TIM4_CH3 0x25: TIM4_UP 0x26: TIM5_CH1 0x27: TIM5_CH2 0x28: TIM5_CH3 0x29: TIM5_CH4 0x2A: TIM5_UP 0x2B: TIM5_TRIG 0x2C: TIM6 0x2D: TIM8_CH1 0x2E: TIM8_CH2 0x2F: TIM8_CH3 0x30: TIM8_CH4 0x31: TIM8_COM 0x32: TIM8_UP 0x33: TIM8_TRIG

## 8 CRC 计算单元

### 8.1 简介

该模块集成了 CRC32 和 CRC16 的功能，循环冗余校验（CRC）计算单元根据固定的生成多项式得到任意 CRC 计算结果。在其他应用中，CRC 技术主要用于验证数据传输或数据存储的正确性和完整性。EN/IEC 60335-1 提供了一种验证闪存完整性的方法。CRC 计算单元可以在程序运行时计算出软件的标识符，然后与连接时产生的参考标识符进行比较，然后存储在指定的内存空间中。

### 8.2 主要特性

#### 8.2.1 CRC32

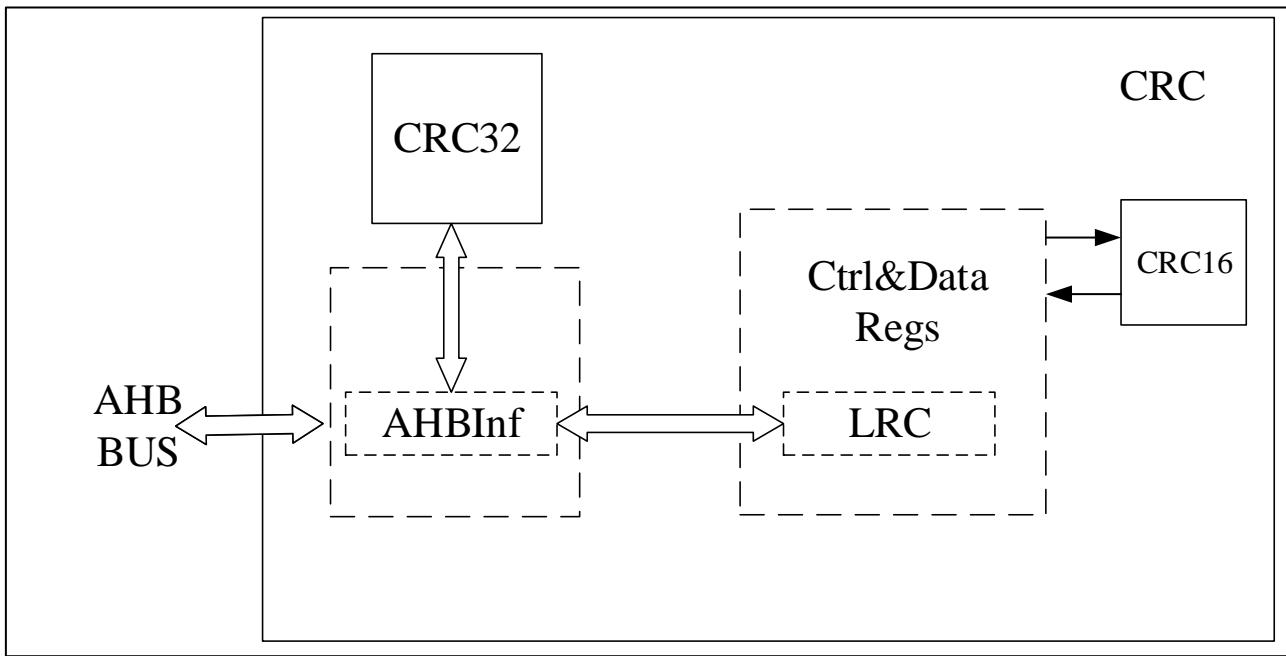
- CRC32( $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$ )
- 32 位待校验数据和 32 位输出校验码。
- CRC 计算时间：1 个 AHB 时钟周期（HCLK）
- 通用 8 位寄存器（可用于存储临时数据）

#### 8.2.2 CRC16

- CRC16( $X^{16}+X^{15}+X^2+1$ )
- 8 位待校验数据和 16 位输出校验码。
- CRC 计算时间：1 个 AHB 时钟周期（HCLK）
- 可配置校验初始值，可配置待校验数据的大小端。
- 支持 8bitLRC 校验值生成

下图为 CRC 计算单元框图：

图 8-1 CRC 计算单元框图



## 8.3 CRC 功能描述

### 8.3.1 CRC32

CRC 计算单元含有 1 个 32 位数据寄存器:

- 写该寄存器，作为 CRC 计算的输入。
- 读该寄存器，作为 CRC 计算的结果。

对该数据寄存器的每一次写操作都会触发用前一个计算结果来计算这个新数据 (CRC 计算是针对整个 32 位字而不是逐字节进行的)。

CRC 计算时要保持 BUS 写操作，因此支持回写或写读。

`CRC_CRC32DAT` 可以通过设置 `CRC_CRC32CTRL.RESET` 重新初始化为 `0xFFFFFFFF`。该操作不影响寄存器 `CRC_CRC32IDAT` 中的数据。

### 8.3.2 CRC16

通过 `CRC_CRC16CTRL.ENDHL` 位来控制校验数据的小端或大端。

要清除最后一次 CRC 操作的结果，请将 `CRC_CRC16CTRL.CLR` 设置为 1 或 `CRC_CRC16D` 设置为 0。

CRC 计算的初始值可以通过写 `CRC_CRC16D` 寄存器来配置。默认情况下，初始值是上一次计算的结果。

LRC 计算与 CRC 计算相同。两者同时进行。可根据需要读出 CRC 或 LRC。如果需要设置初始值，首先要配置 LRC 寄存器。

## 8.4 CRC 寄存器

### 8.4.1 CRC 寄存器总览

下表列出了 CRC 的寄存器映射和复位值

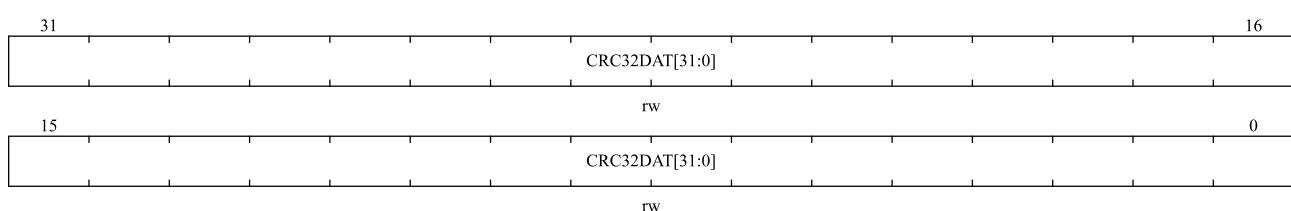
表 8-1 CRC 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	CRC32DAT																																
	Reset Value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
004h	CRC32IDAT																																
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
008h	CRC32CTRL																															RESET	0
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
00Ch	CRC16CTRL																														CLR	ENDHDL	Reserved
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
010h	CRC16DAT																														CRC16DAT[7:0]	0	0
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
014h	CRC16D																														CRC16D[15:0]	0	0
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
018h	LRC																														LRCDAT[7:0]	0	0
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### 8.4.2 CRC32 数据寄存器 (CRC\_CRC32DAT)

偏移地址: 0x00

复位值: 0xFFFF FFFF

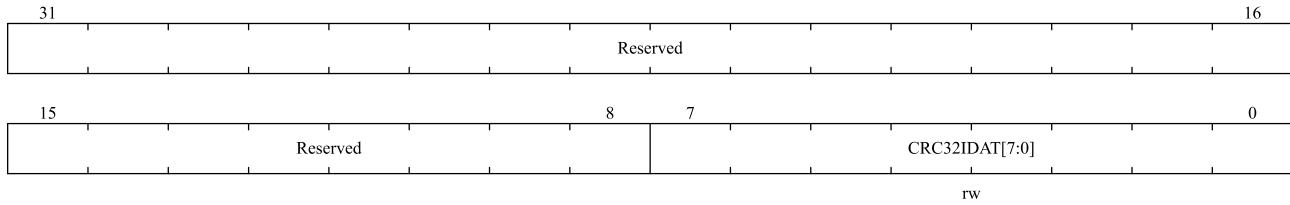


位域	名称	描述
31:0	CRC32DAT[31:0]	CRC32 数据寄存器。 写入的数据为 CRC 待校验值。读取的数据为 CRC 计算结果。仅支持 32 位操作。

### 8.4.3 CRC32 独立数据寄存器 (CRC\_CRC32IDAT)

偏移地址: 0x04

复位值: 0x0000 0000



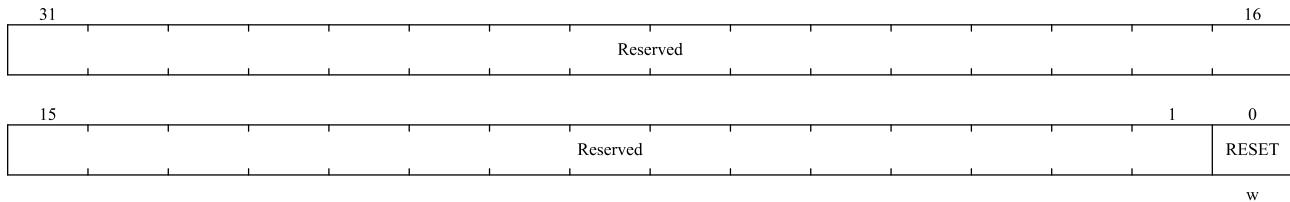
位域	名称	描述
31:8	Reserved	保留，必须保持复位值。
7:0	CRC32IDAT[7:0]	8位独立数据寄存器。 通用8位数据寄存器。它用于临时存储1字节数据。CRC_CRC32CTRL.RESET复位信号不会影响该寄存器。

注：该寄存器不是CRC计算的一部分，可用于存储任何数据。

#### 8.4.4 CRC32 控制寄存器 (CRC\_CRC32CTRL)

偏移地址: 0x08

复位值: 0x0000 0000

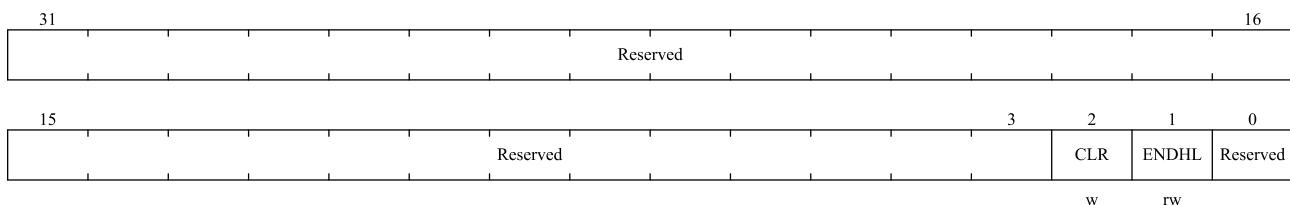


位域	名称	描述
31:1	Reserved	保留，必须保持复位值。
0	RESET	RESET信号。 它可以复位CRC模块并将数据寄存器设置为0xFFFF_FFFF。这个复位只能写1，硬件会自动清0。

#### 8.4.5 CRC16 控制寄存器 (CRC\_CRC16CTRL)

Address offset: 0x0C

Reset value: 0x0000 0000



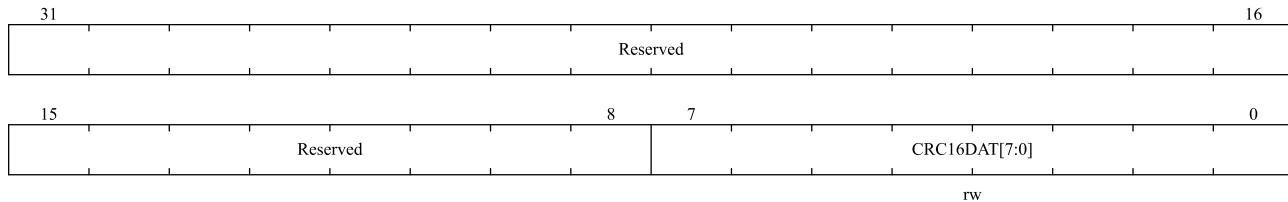
位域	名称	描述
31:3	Reserved	保留，必须保持复位值。
2	CLR	清除 CRC16 结果。 0: 不清除 1: 清除为默认值 0x0000。将此位设置为 1 只会维持 1 时钟周期，硬件会自动清零。(软件读取始终为 0)。
1	ENDHL	要验证的数据从 MSB 或 LSB 开始计算（配置大小端）。 0: 从 MSB 到 LSB 1: 从 LSB 到 MSB 该位仅用于要验证的数据。
0	Reserved	保留，必须保持复位值。

注：支持 8 位、16 位、32 位操作

#### 8.4.6 CRC16 待校验寄存器 (CRC\_CRC16DAT)

偏移地址：0x10

复位值：0x0000 0000



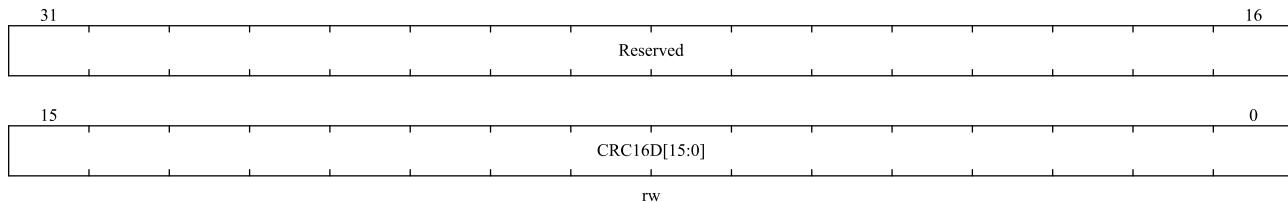
位域	名称	描述
31:8	Reserved	保留，必须保持复位值。
7:0	CRC16DAT[7:0]	待校验的数据。

注：支持 8 位、16 位、32 位操作

#### 8.4.7 CRC 循环冗余校验码寄存器 (CRC\_CRC16D)

偏移地址：0x14

复位值：0x0000 0000



位域	名称	描述
31:16	Reserved	保留，必须保持复位值。

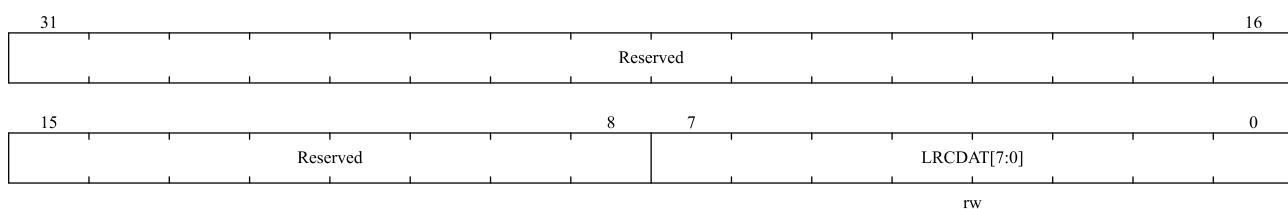
位域	名称	描述
15:0	CRC16D[15:0]	16 位循环冗余结果值。 每次软件写入 CRC16DAT 寄存器时，来自 CRC16 的 16 位计算数据都会在该寄存器中更新。

注：支持 8 位、16 位和 32 位操作（8 位操作必须连续执行两次才能保证 16 位初始值配置正确）

#### 8.4.8 LRC 校验值寄存器 (CRC\_LRC)

偏移地址：0x18

复位值：0x0000 0000



位域	名称	描述
31:8	Reserved	保留，必须保持复位值。
7:0	LRCDAT[7:0]	LRC 校验值。 软件使用前需要写入初始值。然后每次写入 CRC_CRC16DAT 的数据都会与 CRC_LCR 寄存器的值进行异或。结果将存储在 CRC_LCR 中。软件读取结果，下次使用前应清除。

## 9 高级控制定时器（TIM1 和 TIM8）

### 9.1 TIM1 和 TIM8 简介

高级控制定时器（TIM1 和 TIM8）主要用于以下场合：对输入信号进行计数、测量输入信号的脉冲宽度和产生输出波形等。

高级定时器具有互补输出功能、死区插入和刹车功能。适用于电机控制。

### 9.2 TIM1 和 TIM8 主要特性

- 16 位自动装载计数器。（可实现向上计数、向下计数、向上/下计数）。TIM1 支持中央对齐非对称模式
- 16 位可编程预分频器。（分频系数可配置为 1 到 65536 之间的任意值）
- 可编程重复计数器
- TIM1 最多 9 个通道，TIM8 最多 6 个通道
- 4 个捕获/比较通道，工作模式为：PWM 输出、输出比较、单脉冲模式输出、输入捕获
- 如下事件发生时产生中断/DMA：
  - ◆ 更新事件
  - ◆ 触发事件
  - ◆ 输入捕获
  - ◆ 输出比较
  - ◆ 刹车信号输入
- 死区时间可编程的互补输出
  - 对于 TIM1，通道 1、2、3、4 支持此功能
  - 对于 TIM8，通道 1、2、3 支持此功能
- 可通过外部信号控制定时器
- 多个定时器从内部连接在一起，以实现定时器同步或链接
- TIM1\_CC5 和 TIM8\_CC5 用于比较器消隐
- TIM1 的通道 4/7/8/9 可输出的脉冲信号，可通过配置上升沿和下降沿来触发 ADC
- 增量（正交）编码器接口：用于追踪运行轨迹和解析旋转方位
- 霍尔传感器接口：用于三相电机控制

图 9-1 TIM1 框图

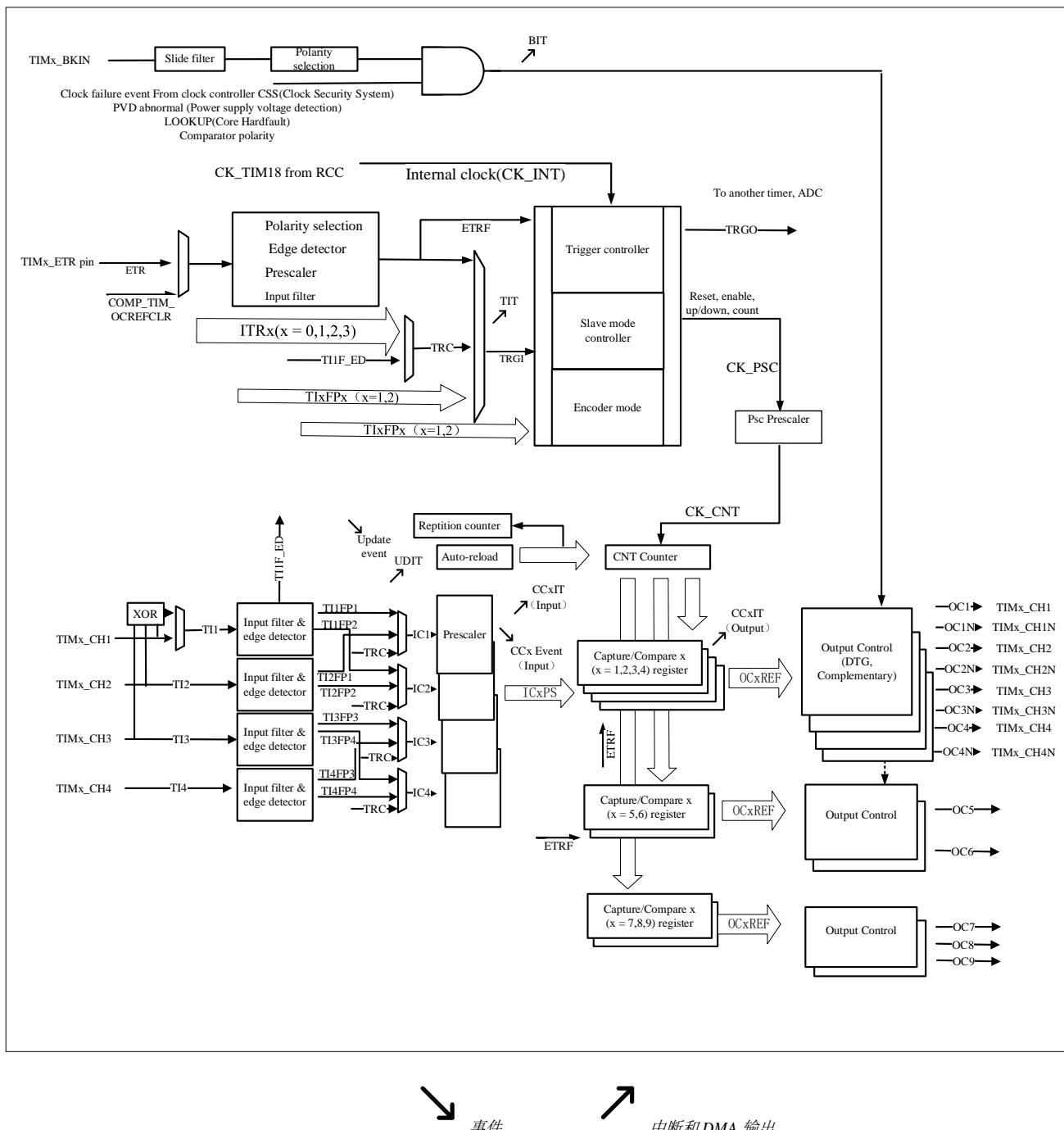
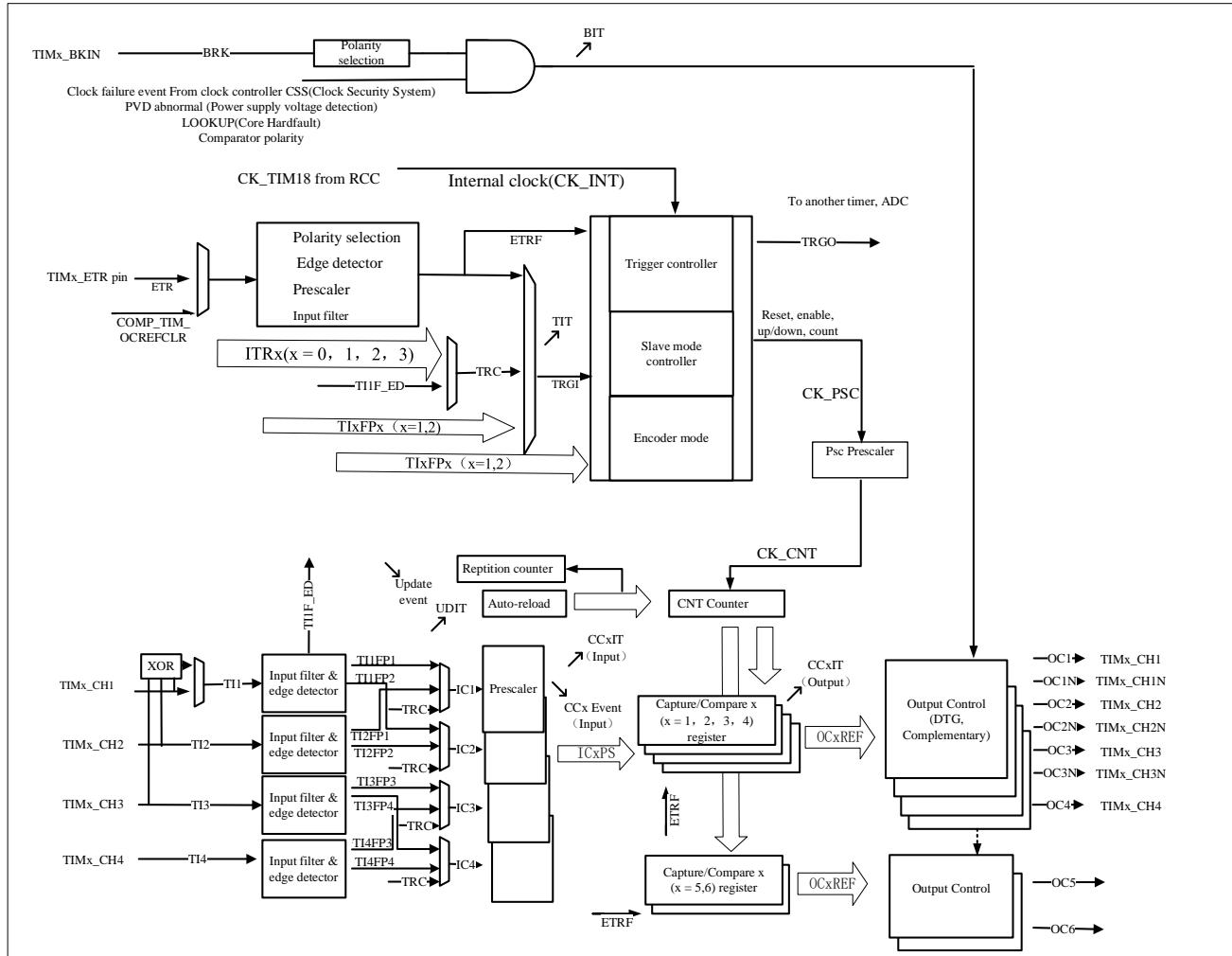


图 9-2 TIM8 框图



事件

中断和DMA 输出

捕获通道 1 输入可以来自 IOM 或比较器输出

## 9.3 TIM1 和 TIM8 功能描述

### 9.3.1 时基单元

高级控制器的时基单元主要包括：预分频器、计数器、自动重装载寄存器和重复计数器。当时基单元工作时，软件可以随时读取和写入相应的寄存器（TIMx\_PSC、TIMx\_CNT、TIMx\_AR 和 TIMx\_REPCNT）。

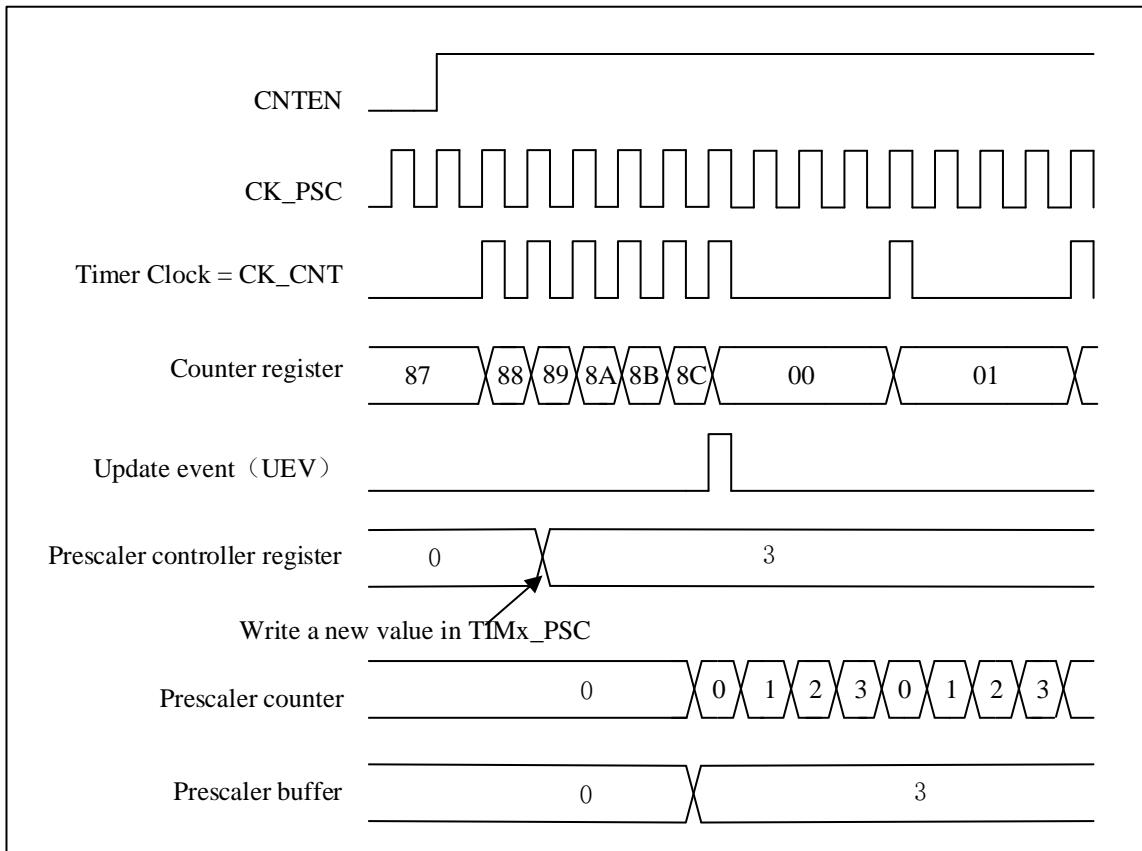
根据自动重装载预装载使能位（TIMx\_CTRL1.AR PEN）的设置，预装载寄存器的值会立即或在每次更新事件 UEV 时传输到影子寄存器。TIMx\_CTRL1.UPDIS=0 时，当计数器达到上溢/下溢条件或由软件设置 TIMx\_EVTGEN.UDGN，将生成更新事件。计数器 CK\_CNT 仅在 TIMx\_CTRL1.CNTEN 位被设置时有效。计数器在 TIMx\_CTRL.CNTEN 位被设置后一个时钟周期之后开始计数。

#### 9.3.1.1 预分频器描述

TIMx\_PSC 寄存器由一个 16 位计数器组成，可用于计数器时钟频率按 1 和 65536 之间的任意分频。因为这

个控制器带有缓冲器，可以在运行时动态改变。新的预分频器值只有在下次更新事件中才会被采用。

图 9-3 当预分频的参数从 1 到 4，计数器的时序图



## 9.3.2 计数器模式

### 9.3.2.1 向上计数模式

使用向上计数模式，计数器将从 0 计数到寄存器 TIMx\_AR 的值，然后重置为 0。并产生一个计数器溢出事件。

如果设置 TIMx\_CTRL1.UPRS 位(选择更新请求)和 TIMx\_EVTGEN.UDGN 位，将产生一个更新事件(UEV)。但是 TIMx\_STS.UDITF 不会被硬件置起，因此不会产生更新中断或 DMA 更新请求。这是为了避免清除计数器时产生更新中断。

取决于 TIMx\_CTRL1.UPRS 的配置，发生更新事件时，所有寄存器都会更新，TIMx\_STS.UDITF 被设置

- 重复计数器被重新加载为 TIMx\_REPCNT 的内容
- 当 TIMx\_CTRL1.ARREN = 1，预装载寄存器(TIMx\_AR)的值被更新到自动装载影子寄存器
- 预加载值 (TIMx\_PSC) 被重新加载到预分频器影子寄存器中

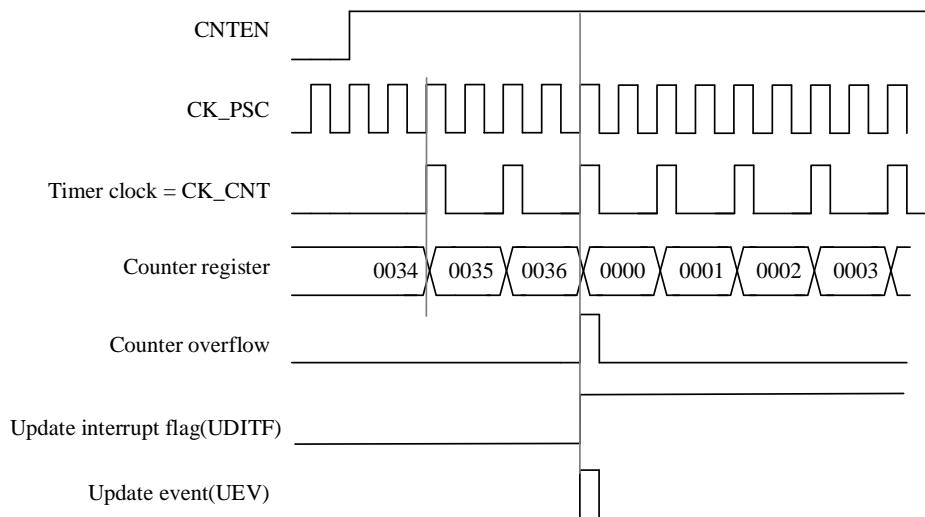
为了避免在向预装载寄存器中写入新值时更新影子寄存器，可以通过设置 TIMx\_CTRL1.UPDIS=1 来禁止更新事件。

当产生一个更新事件时,计数器仍将被清除,预分频器计数器也将被设置为0(但预分频器值将保持不变)。

下图给出一些示例,展示了向上计数模式计数器在不同分频因子下的动作。

图 9-4 当内部时钟分频因子 =  $2/N$  时, 向上计数的时序图

### Internal clock divided by 2



### Internal clock divided by N

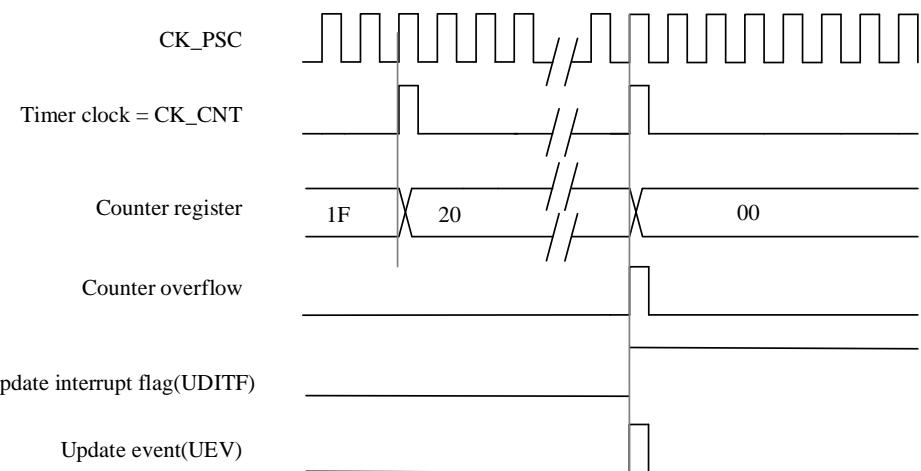
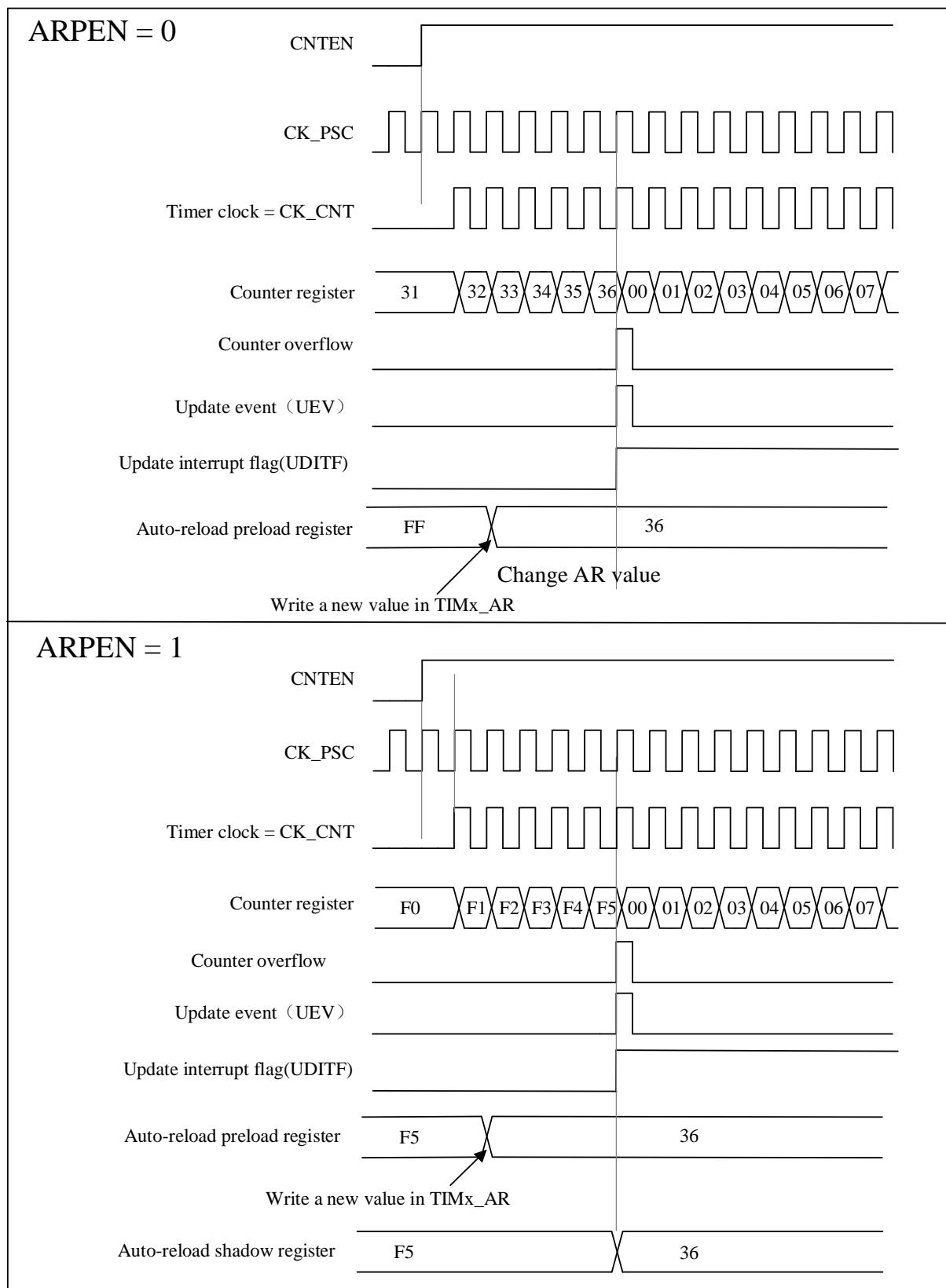


图 9-5 当 ARPEN=0/1 产生更新事件时，向上计数的时序图



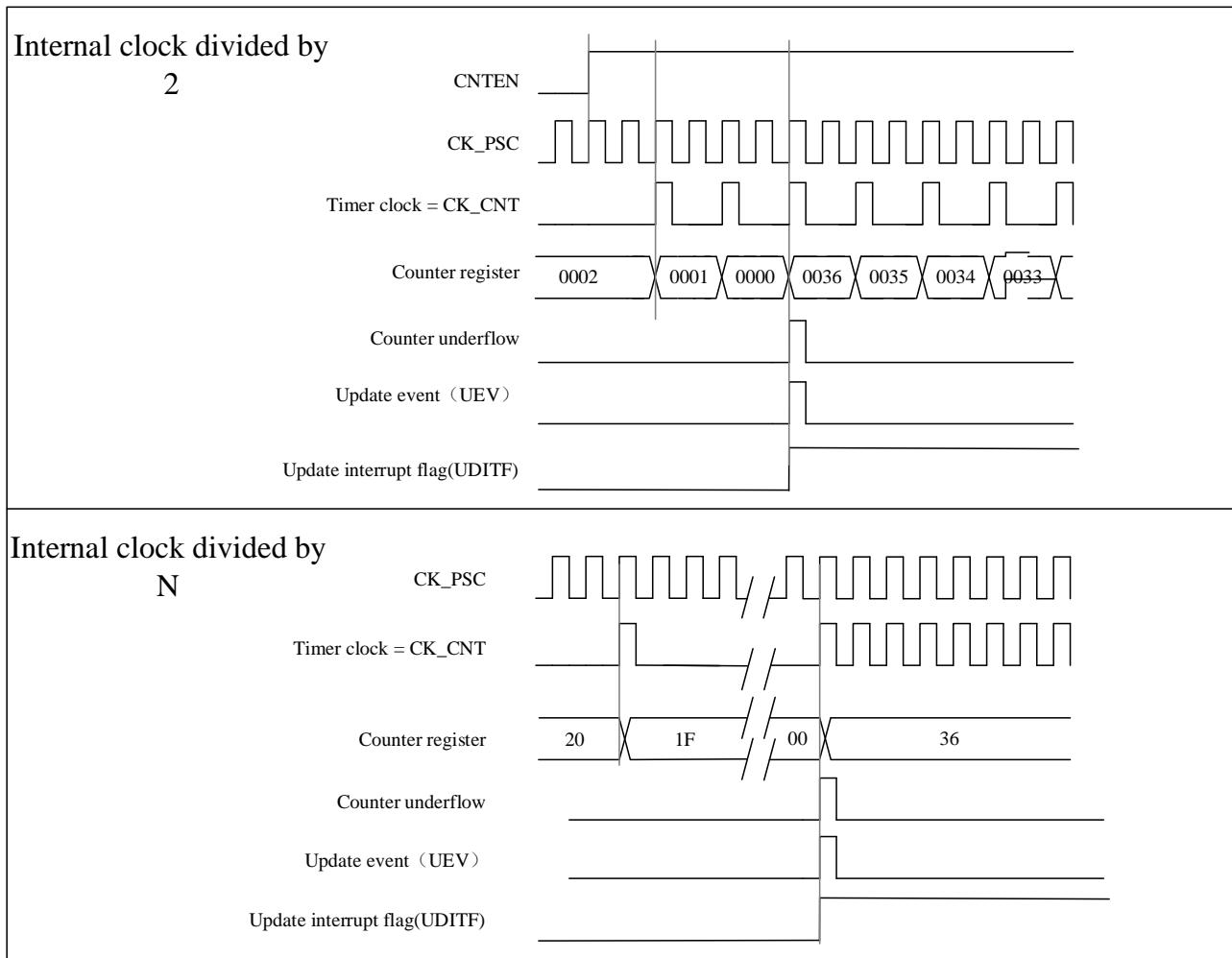
### 9.3.2.2 向下计数模式

向下计数模式，计数器将从寄存器 TIMx\_AR 的值减至 0，然后从自动重装载值重新开始，并产生计数器向下溢出事件。

向下计数模式和向上计数模式配置更新事件和更新寄存器的过程相同，请查阅 9.3.2.1 章节。

下图给出一些示例，展示了向下计数模式计数器在不同分频因子下的动作。

图 9-6 内部时钟分频因子 = 2/N 时，向下计数时序图



### 9.3.2.3 中央对齐模式

#### 9.3.2.3.1 中央对齐对称模式

在中央对齐对称模式下，计数器从 0 增加到值 (TIMx\_AR) -1，产生计数器溢出事件。然后，它从自动重装载值 (TIMx\_AR) 向下计数到 1，并生成一个计数器向下溢出事件。然后计数器重置为 0 并再次开始计数。

在这种模式下，TIMx\_CTRL1.DIR 方向位无效，由硬件更新和指定当前计数方向。当 TIMx\_CTRL1.CAMSEL 位不等于“00”时，中央对齐模式有效。

每次计数上溢和计数下溢时都会生成更新事件。或者，也可以通过设置 TIMx\_EVTGEN.UDGN 位（通过软件或使用从模式控制器）来生成更新事件。在这种情况下，计数器从 0 重新开始计数，预分频器的计数器也

从 0 重新开始计数。

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新载入之前被更新。

图 9-7 内部时钟分频因子 = 2/N，中央对齐时序图

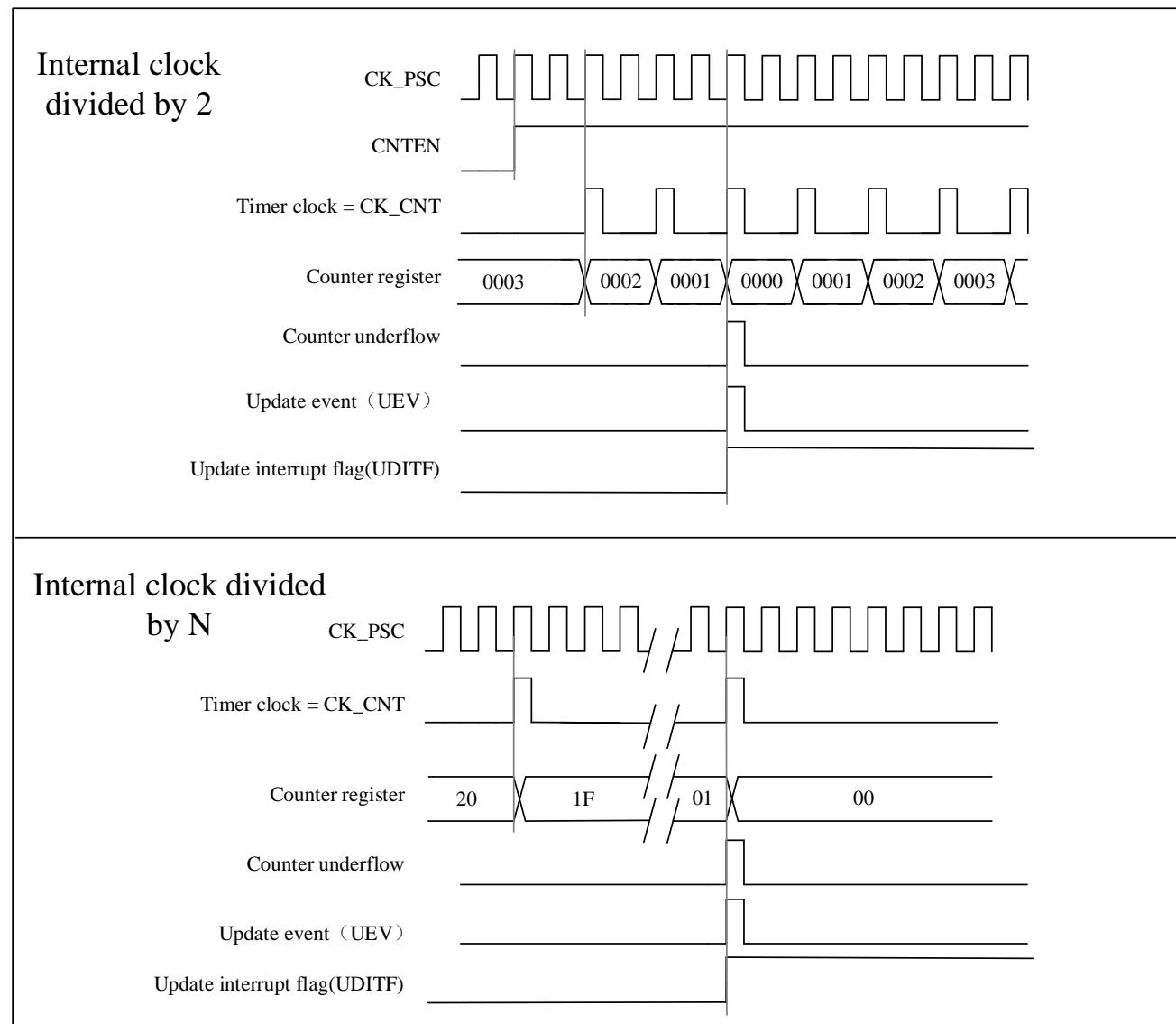
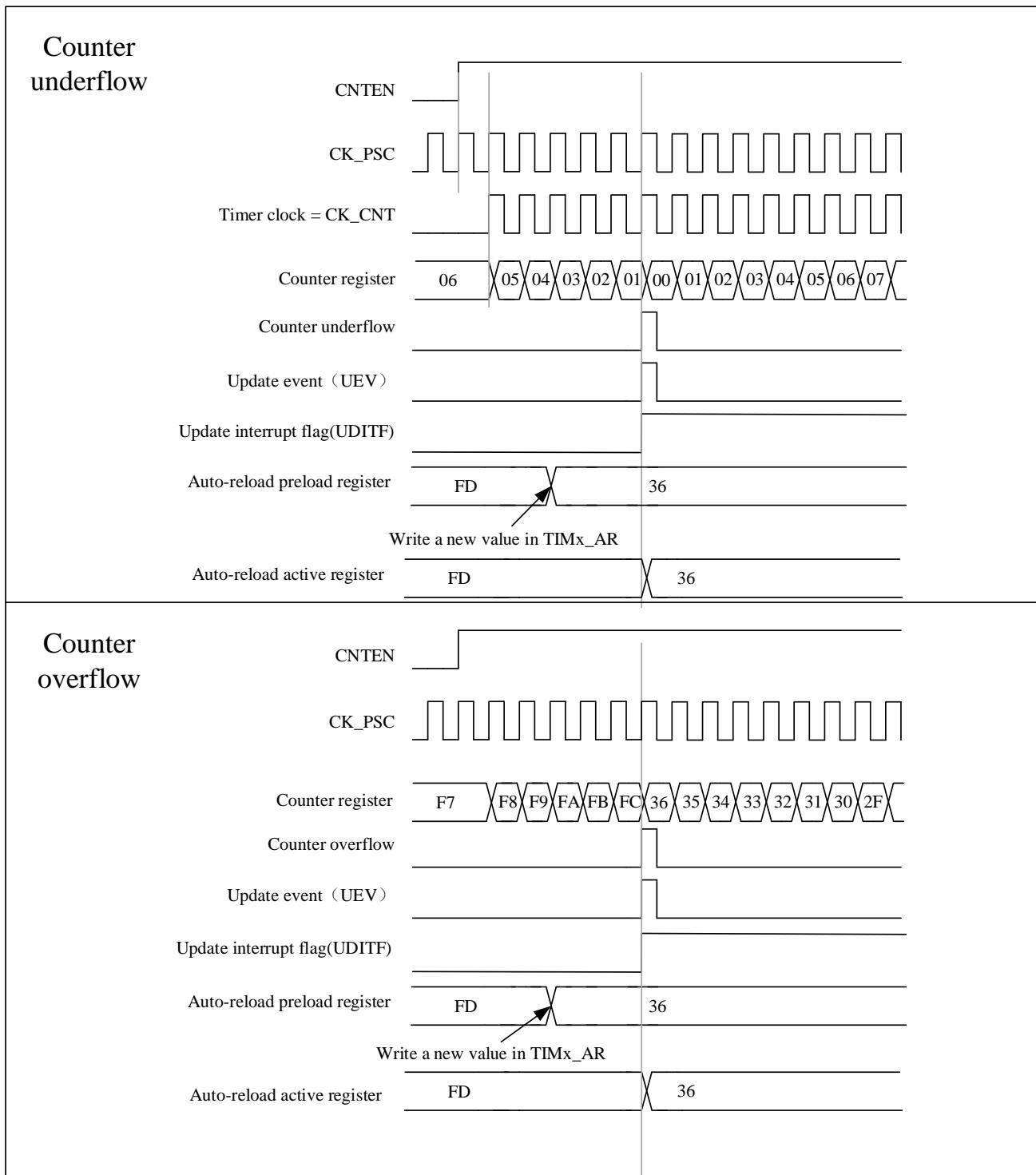


图 9-8 包含计数器上溢和下溢的中央对齐时序图(ARPEN=1)



### 9.3.2.3.2 中央对齐非对称模式

中央对齐非对称模式仅用于 TIM1

在中央对齐非对称模式下 (TIMx\_CTRL1.ASYMMETRIC 为 1, TIMx\_CTRL1.CAMSEL[1:0] 为非零), 计数器从 0 计数到自动重载值 (TIMx\_AR) -1, 并产生计数器溢出事件, 然后从自动重载值计数到 1, 并产生计数器向下溢出事件, 然后从 0 重新开始计数。

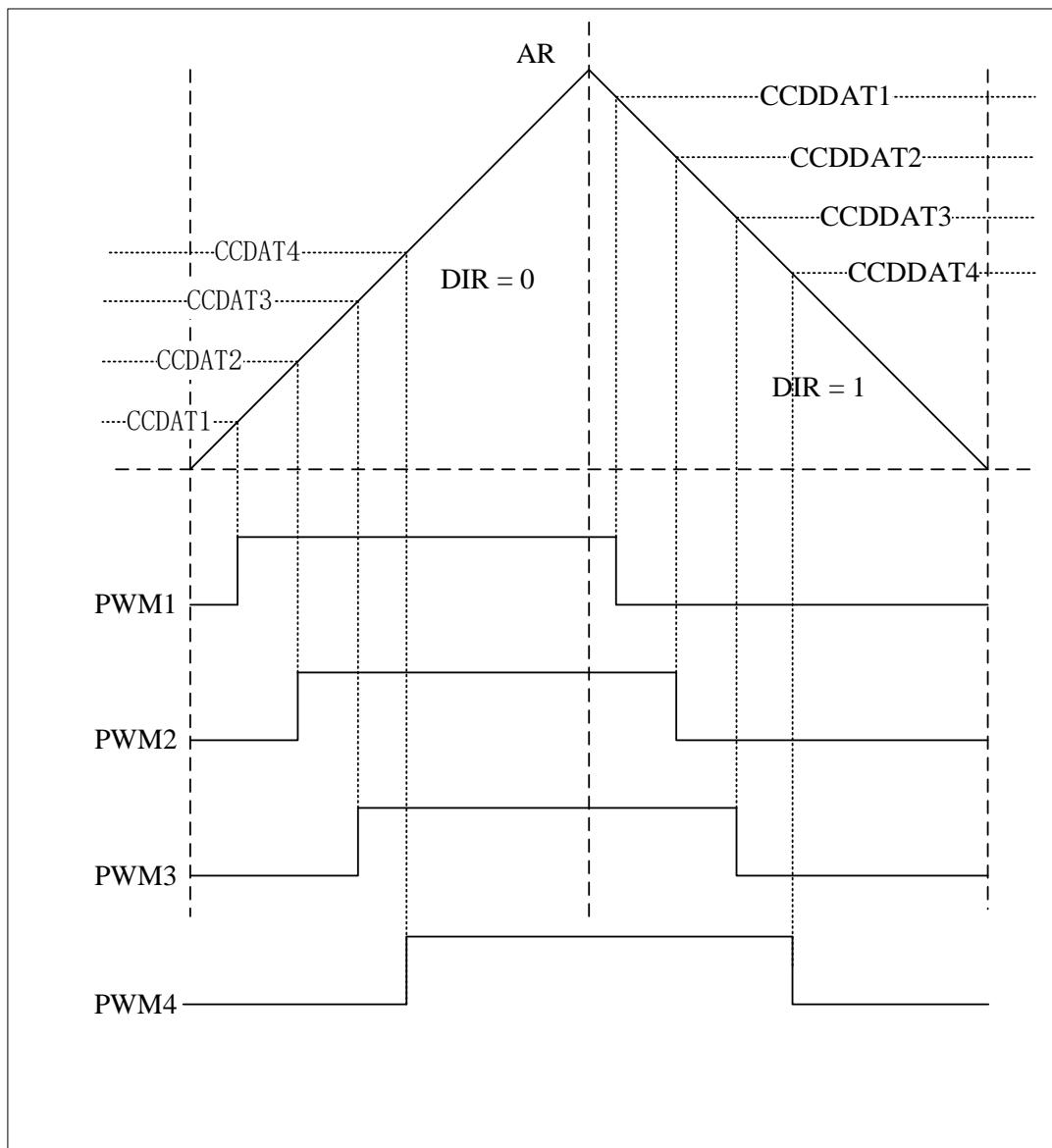
TIMx\_CTRL1.DIR 值不能在此模式下写入，它由硬件更新和指定当前计数方向。

当通道不是 1,2,3,4 时，比较值是 CCDDATx。当死区时间发生器打开时，请注意，当 DIR=0 时，死区时间插入点是计数器值等于 CCDDATx ( $x=1,2,3,4$ )，当 DIR=1 时，死区时间插入点是计数器值等于 CCDDATx ( $x=1,2,3,4$ )。

每次计数上溢和计数下溢时都会产生更新事件。或者，也可以通过设置 TIMx\_EVTGEN.UDGN 位（通过软件或使用从模式控制器）产生更新事件。在这种情况下，在这种情况下，计数器从 0 重新开始计数，预分频器的计数器也从 0 重新开始计数。

**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新载入之前被更新。

图 9-9 非对称模式对应的输出波形

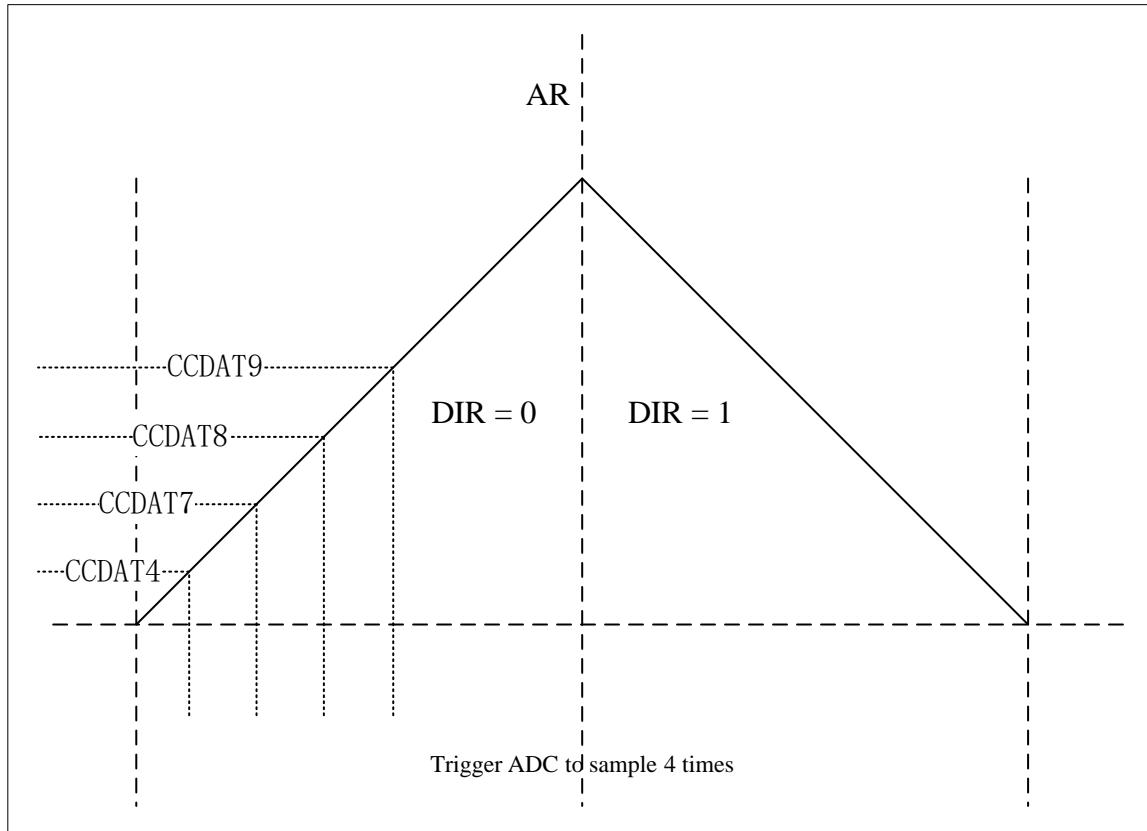


由于增加了 CC7/CC8/CC9 三个通道的触发功能，并对 CC4 的触发功能进行了修改，现在描述 CC4/CC7/CC8/CC9 通道对 ADC 触发的描述。

当定时器在中央对齐非对称模式下工作时，每个通道（CC4/CC7/CC8/CC9）都可以单独触发 ADC。如果

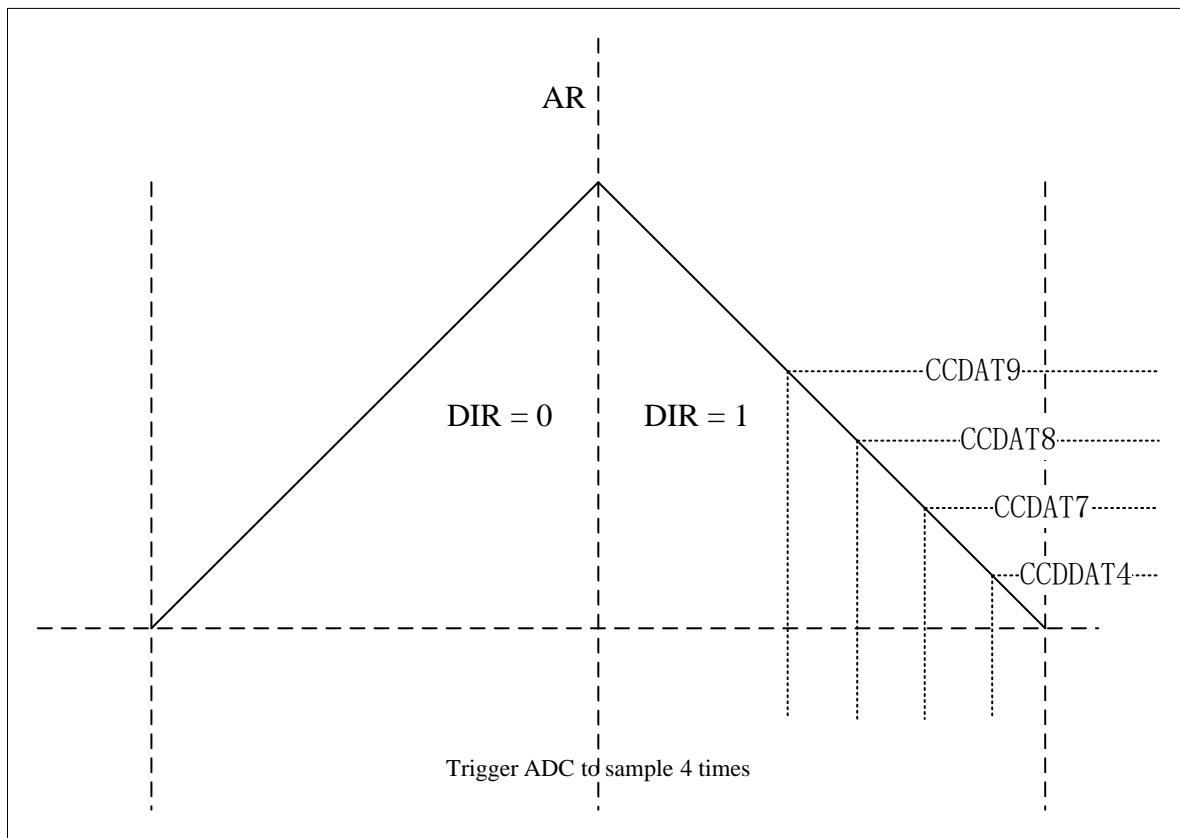
TIMx\_CTRL1.CMODE[1:0]=00, 在 CCDDATx(x=4,7,8,9) 中, CCDDATx 的 CCDDAT 值仅在 DIR=0 时触发 ADC。

图 9-10 CCDDATx(x=4,7,8,9), 当 DIR = 0 时触发 ADC



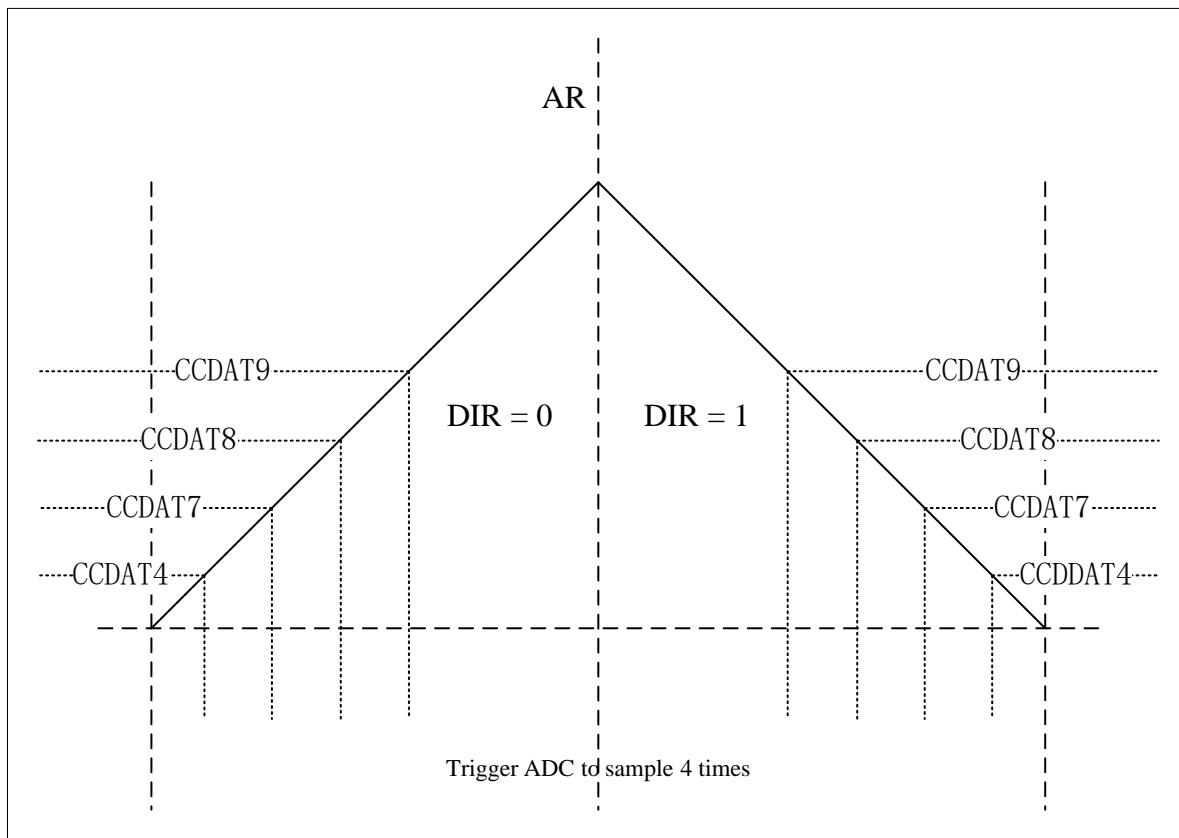
如果 TIMx\_CTRL1.CMODE [1:0]=01, 在 CCDDATx (x=4,7,8,9) 中, CCDDATx 的 CCDDAT/CCDDAT 值仅在 DIR=1 时触发 ADC。

图 9-11 CCDDATx(x=4,7,8,9), 当 DIR = 1 时触发 ADC



如果 TIMx\_CTRL1.CMODE[1:0]=1x，在 CCDDATx ( $x=4,7,8,9$ ) 中，当 DIR=0 或 DIR=1 时，CCDDATx 的 CCDDAT/CCDDAT 值将触发 ADC.

图 9-12 CCDDATx(x=4,7,8,9), 当 DIR = 1 或 DIR = 0 时触发 ADC



在上图中，通道 4 向上计数至 CCDDAT4 或向下计数至 CCDDAT4，通道 7/8/9 向上计数或向下计数至 CCDDAT7/8/9，触发有效。

### 9.3.3 重复计数器

第 9.3.1 章节的基本单元描述了生成更新事件（UEV）的条件。更新事件（UEV）实际上仅在重复计数器达到零时生成，这对于生成 PWM 信号非常有用。

这意味着每  $N+1$  计数器溢出或下溢一次，数据就会从预加载寄存器传输到影子寄存器，其中  $N$  是 TIMx\_REPCNT 中的值。

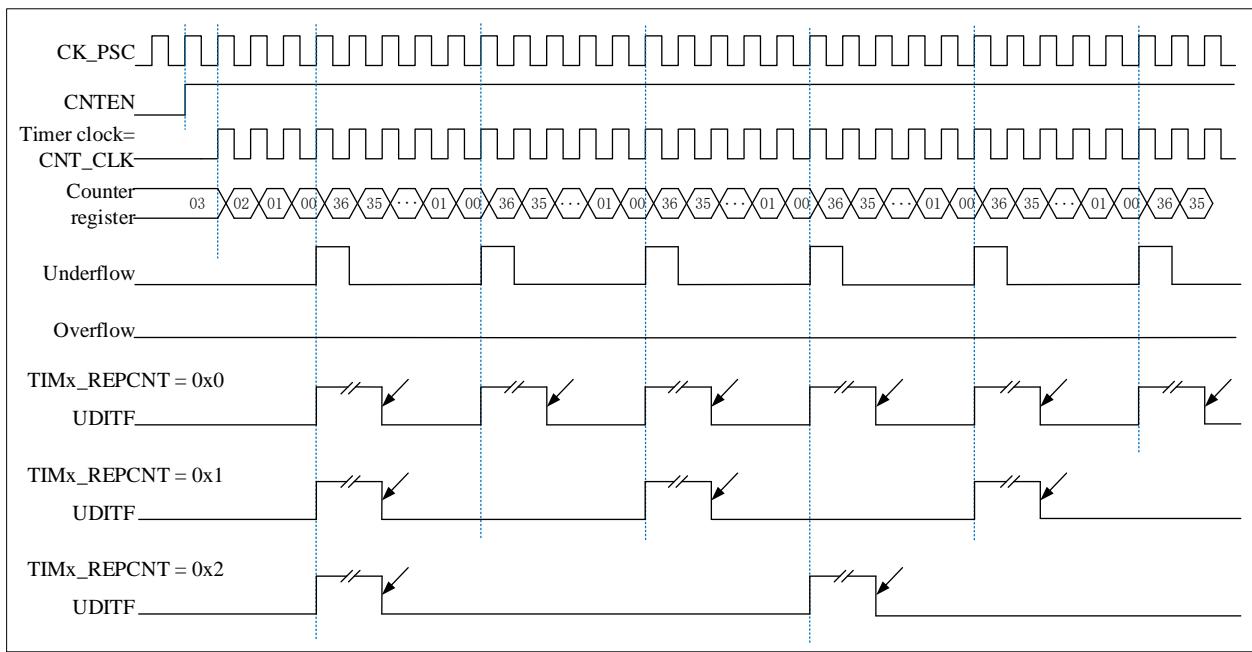
重复计数器递减：

- 在向上计数模式下，每次计数器达到最大值时，都会发生溢出
- 在向下计数模式下，每次计数器减至最小值时，都会发生下溢
- 在中央对齐模式下，每次计数上溢或下溢时

其重复率由 TIMx\_REPCNT 寄存器的值定。

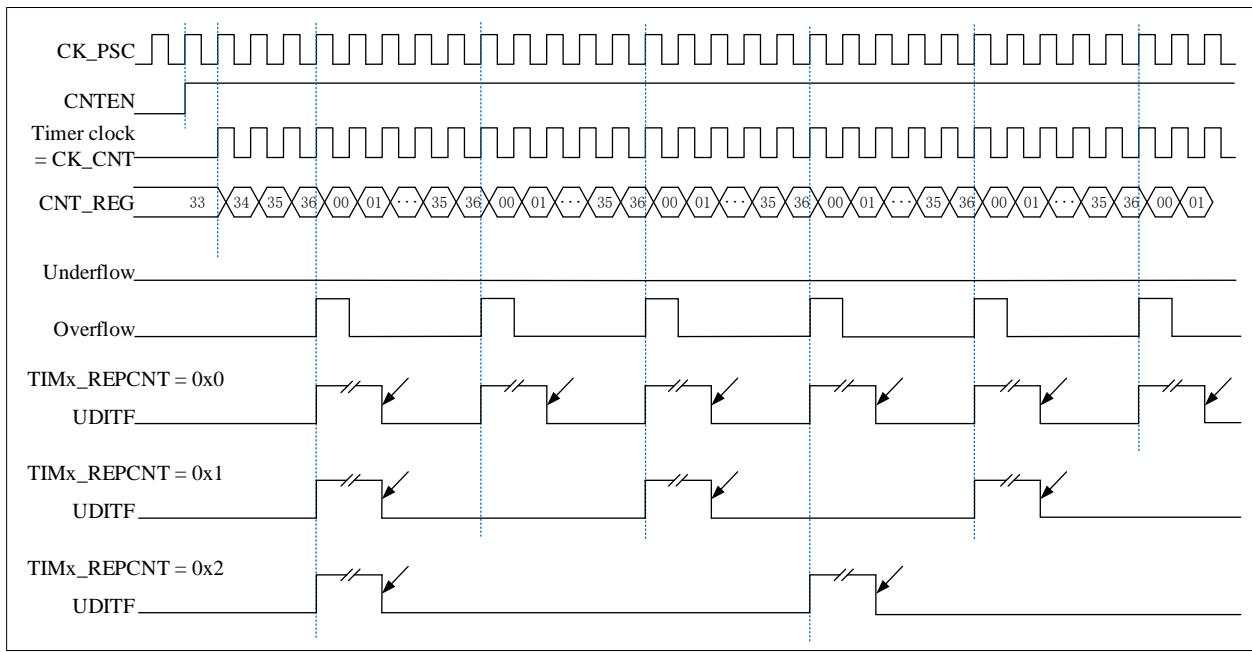
重复计数器具有自动重新加载功能。无论重复计数器的值如何，更新事件（通过从模式控制器设置 TIMx\_EVTGEN.UDGN 或硬件生成）都会立即发生。

图 9-13 向下计数模式下的重复计数时序图



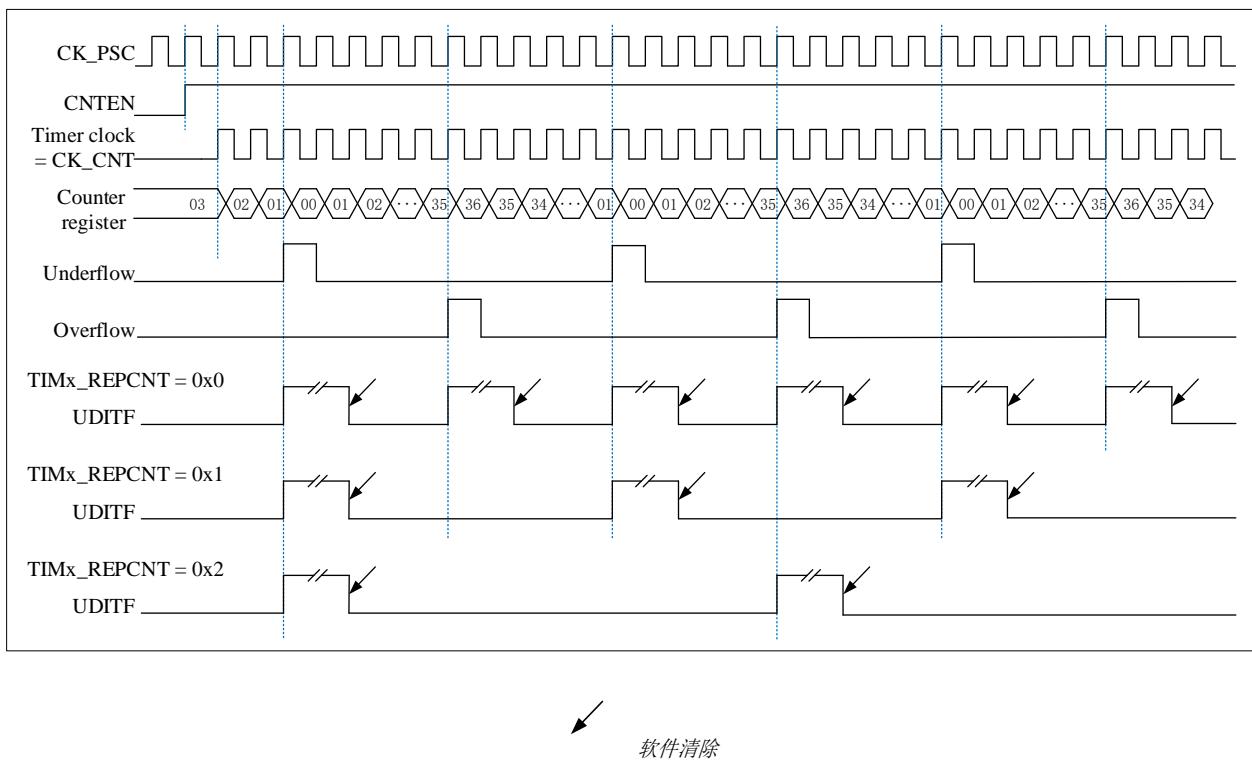
↓  
软件清除

图 9-14 向上计数模式下的重复计数时序图



↓  
软件清除

图 9-15 中央对齐模式下的重复计数时序图



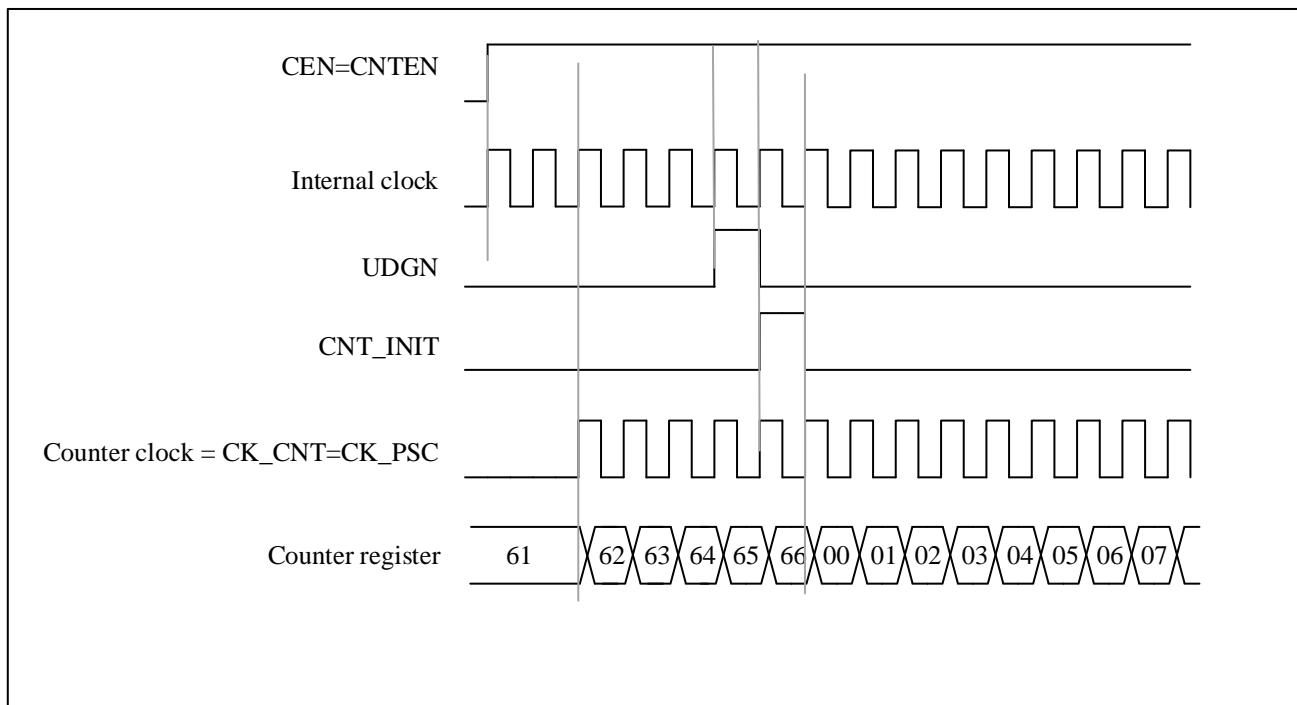
### 9.3.4 时钟选择

- 高级控制定时器的内部时钟: CK\_INT;
- 两种外部时钟模式:
  - 外部输入引脚
  - 外部触发输入 ETR
- 内部触发输入 (ITRx): 一个定时器用作另一个定时器的预分频器

#### 9.3.4.1 内部时钟源(CK\_INT)

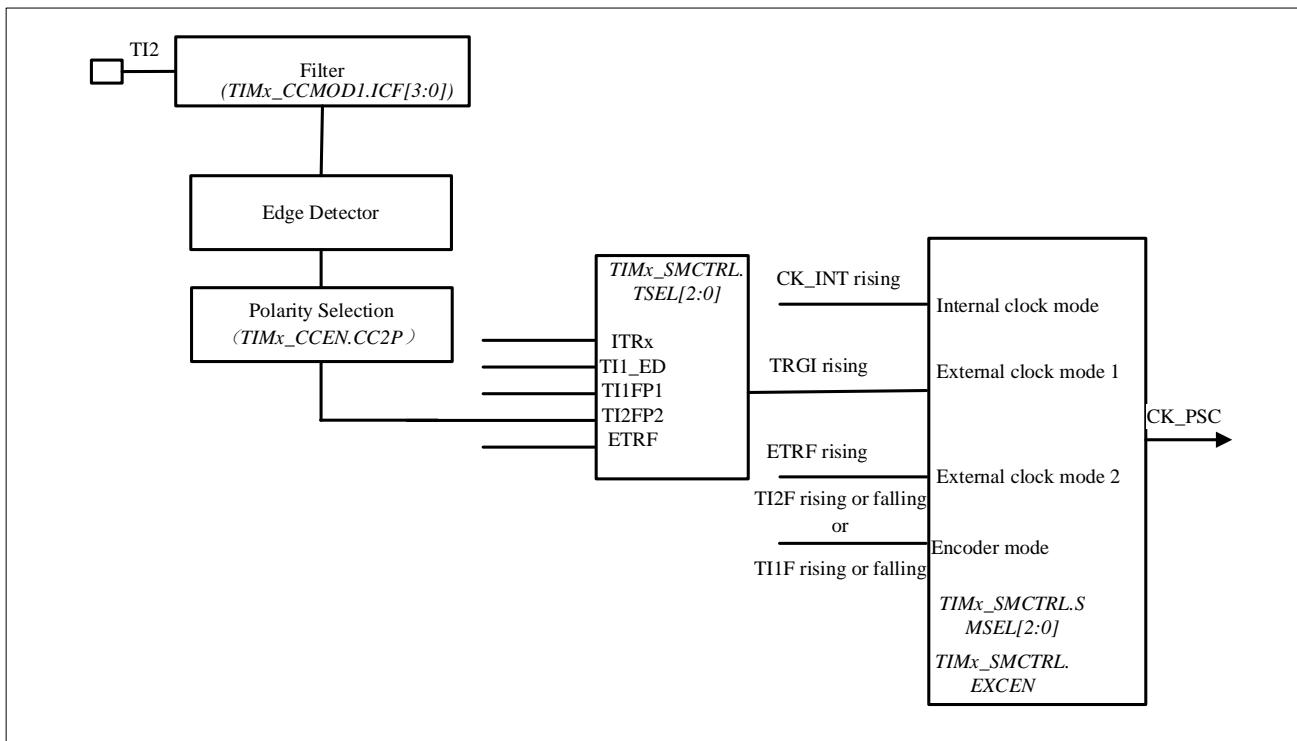
当 TIMx\_SMCTRL.SMSEL 等于“000”时，从模式控制器被禁用。这三个控制位 (TIMx\_CTRL1.CNTEN、TIMx\_CTRL1.DIR、TIMx\_EVTGEN.UDGN) 只能由软件改变 (TIMx\_EVTGEN.UDGN 除外，它保持自动清零)。前提是 TIMx\_CTRL1.CNTEN 位被软写为'1'，预分频器的时钟源由内部时钟 CK\_INT 提供。

图 9-16 正常模式下的控制电路，内部时钟除以 1



### 9.3.4.2 外部时钟源模式 1

图 9-17 TI2 外部时钟连接示例



通过配置 **TIMx\_SMCTRL.SMSEL=111** 选择该模式。计数器可以配置为在所选输入的时钟上升沿或下降沿进行计数。

例如，配置向上计数模式在 TI2 输入的时钟上升沿计数，配置步骤如下：

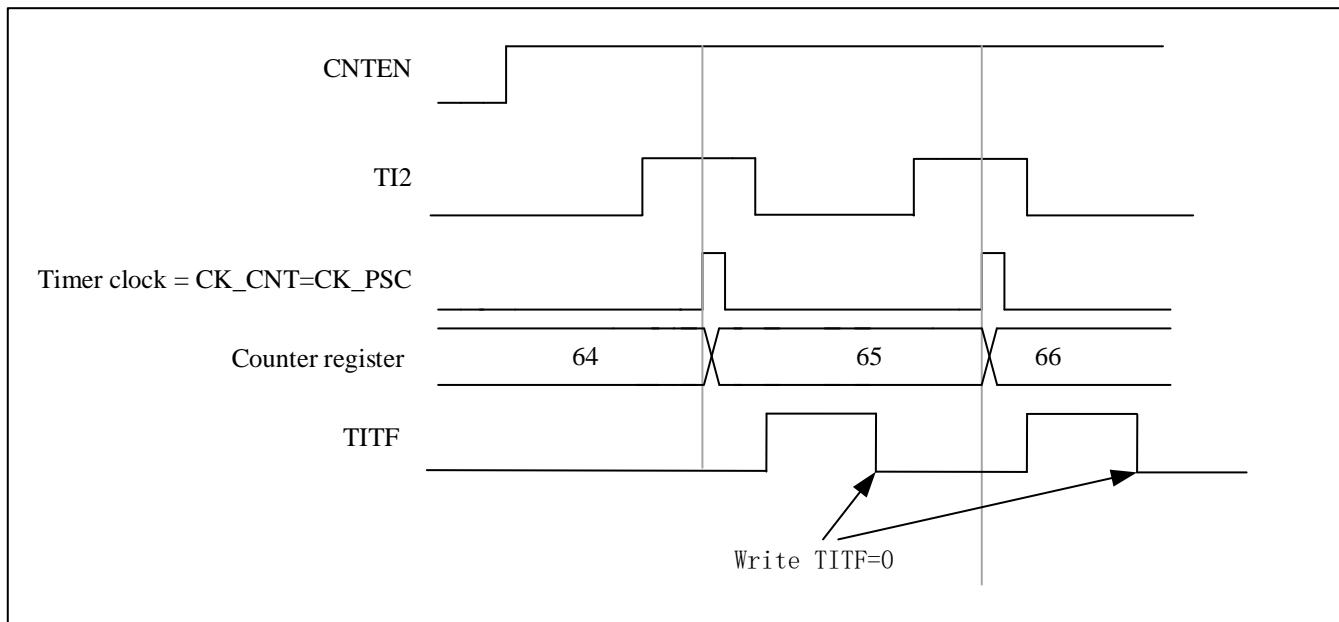
- 配置 TIMx\_CCMOD1.CC2SEL 等于‘01’，CC2 通道配置为输入，IC2 映射到 TI2
- 配置 TIMx\_CCEN.CC2P 等于‘0’，选择时钟上升沿极性
- 通过配置 TIMx\_CCMOD1.IC2F[3:0] 选择输入滤波器带宽（如果不需要滤波器，保持 IC2F 位为‘0000’）
- 配置 TIMx\_SMCTRL.SMSEL 等于‘111’，选择定时器外部时钟模式 1
- 配置 TIMx\_SMCTRL.TSEL 等于‘110’，选择 TI2 作为触发输入源
- 配置 TIMx\_CTRL1.CNTEN 等于 ‘1’ 以启动计数器

**注意：捕获预分频器不用于触发，所以不需要配置**

当定时器时钟的上升沿出现在 TI2=1 时，计数器计数一次并且 TIMx\_STS .TITF 标志被拉高。

TI2 的上升沿与计数器实际时钟之间的延迟取决于 TI2 输入端的再同步电路。

图 9-18 外部时钟模式 1 的控制电路

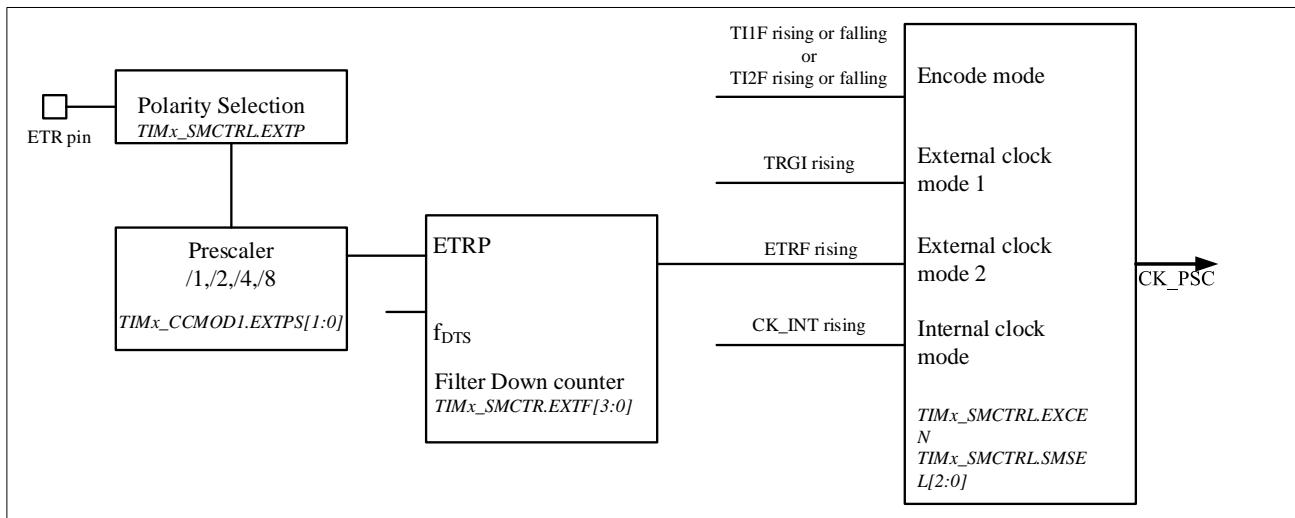


### 9.3.4.3 外部时钟源模式 2

此模式由 TIMx\_SMCTRL.EXCEN 选择等于 1。计数器可以在外部触发输入 ETR 的每个上升沿或下降沿计数。

下图为外部时钟源模式 2 的外部触发输入模块示意图。

图 9-19 外部触发输入框图

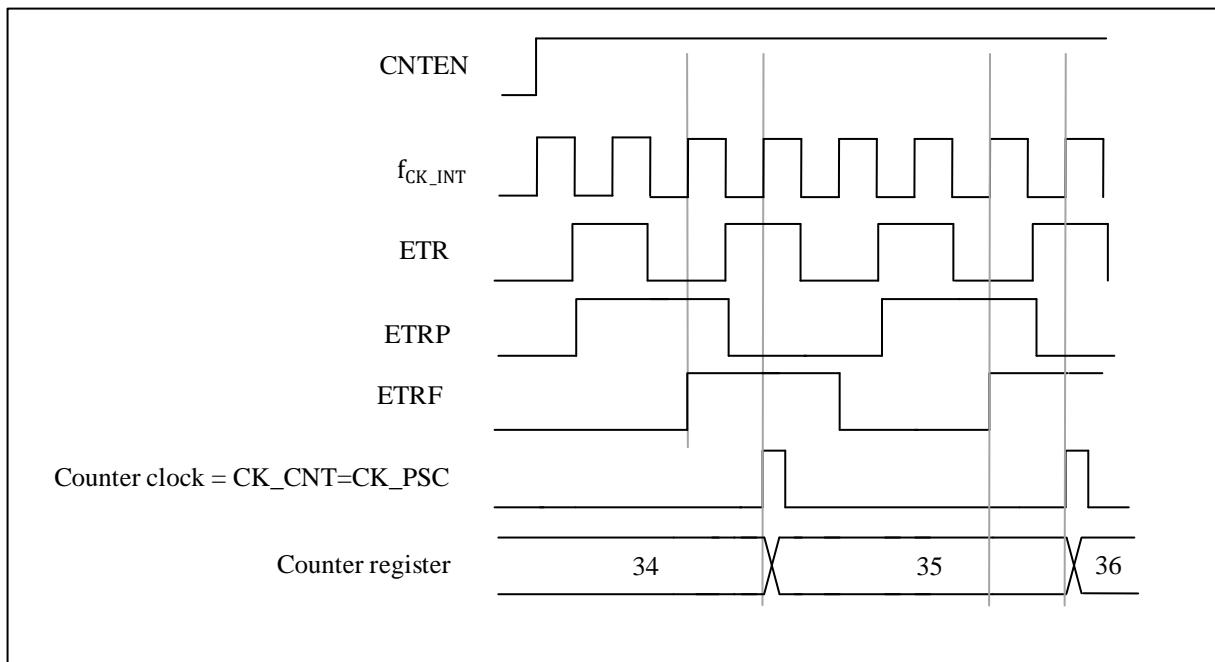


例如，使用以下配置步骤使向上计数器在 ETR 上每 2 个上升沿计数一次。

- 由于在这种情况下不需要过滤器，因此使 `TIMx_SMCTRL.EXTF[3:0]` 等于‘0000’
- 通过使 `TIMx_SMCTRL.EXTPS[1:0]` 等于 ‘01’ 来配置预分频器
- 通过设置 `TIMx_SMCTRL.EXTP` 等于‘0’来选择 ETR 引脚的极性，ETR 的上升沿有效
- 外部时钟模式 2 通过设置 `TIMx_SMCTRL.EXCEN` 等于‘1’来选择
- 通过设置 `TIMx_CTRL1.CNTEN` 等于“1”启动计数器。

计数器每 2 个 ETR 上升沿计数一次。ETR 的上升沿与计数器的实际时钟之间的延迟是由于 ETRP 信号上的再同步电路造成的。

图 9-20 外部时钟模式 2 的控制电路

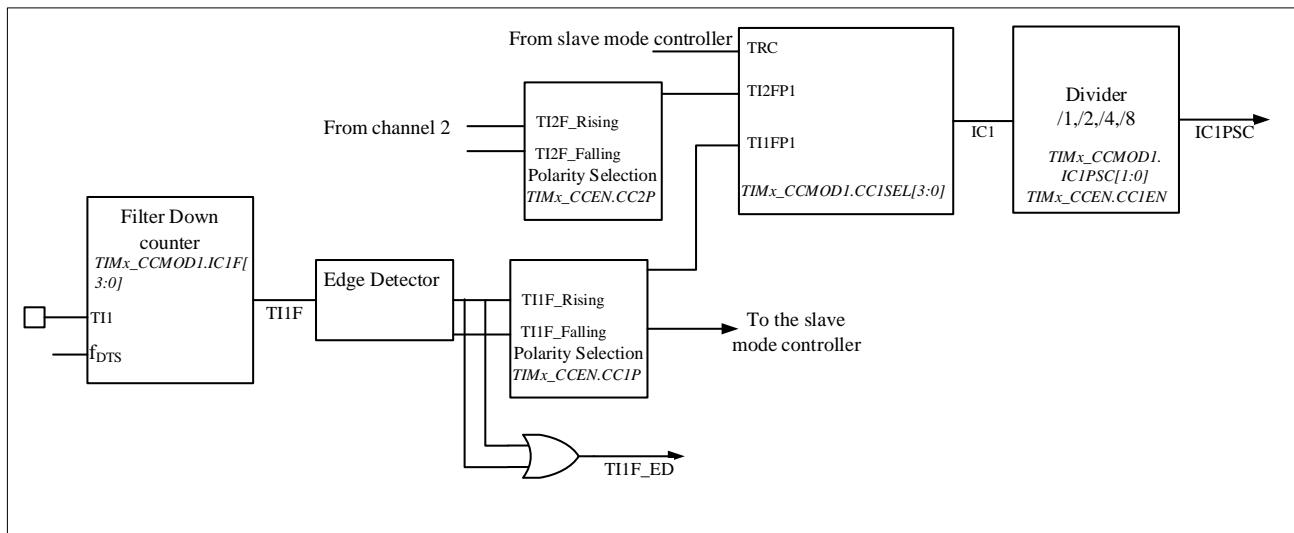


### 9.3.5 捕获/比较通道

捕获/比较通道包括捕获/比较寄存器和影子寄存器。输入部分由数字滤波器、多路复用器和预分频器组成。输出部分包括比较器和输出控制。

输入信号 TIx 被采样和滤波以产生信号 TIxF。然后由极性选择功能的边沿检测器生成信号 (TIxF\_rising 或 TIxF\_falling)，其极性由 TIMx\_CCEN.CCxFP 位选择。该信号可用作从模式控制器的触发输入。同时，信号 ICx 经过分频后送入捕获寄存器。下图显示了捕获/比较通道的框图。

图 9-21 捕获/比较通道（例如：通道 1 输入级）



输出部分生成一个中间波形 OCxRef (高电平有效) 作为参考。极性作用在链的末端。

图 9-22 捕获/比较通道 1 主电路

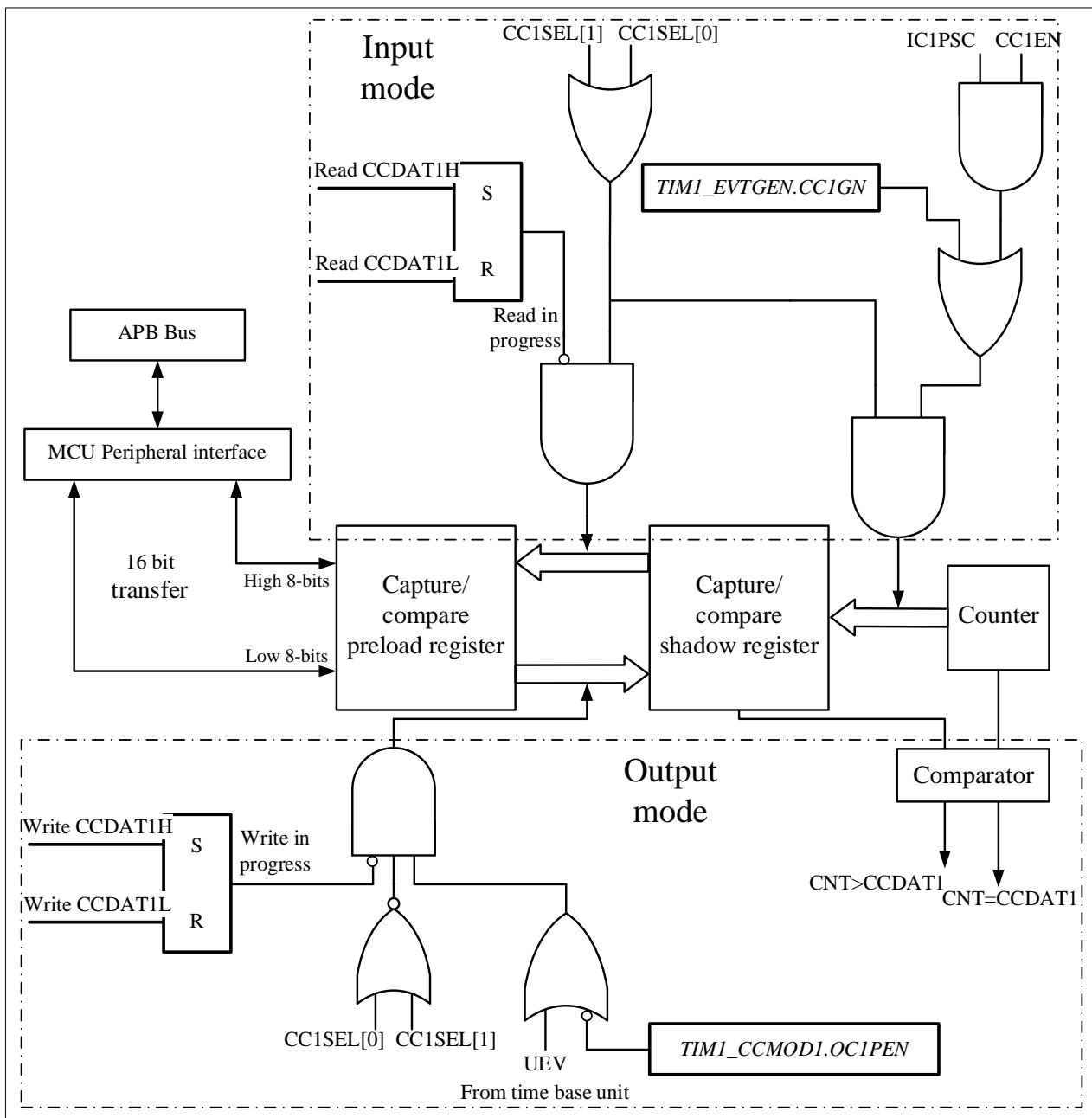


图 9-23 通道 x 的输出部分 (TIM1 的 x=1,2,3,4; TIM8 的 x=1,2,3。以通道 1 为例子)

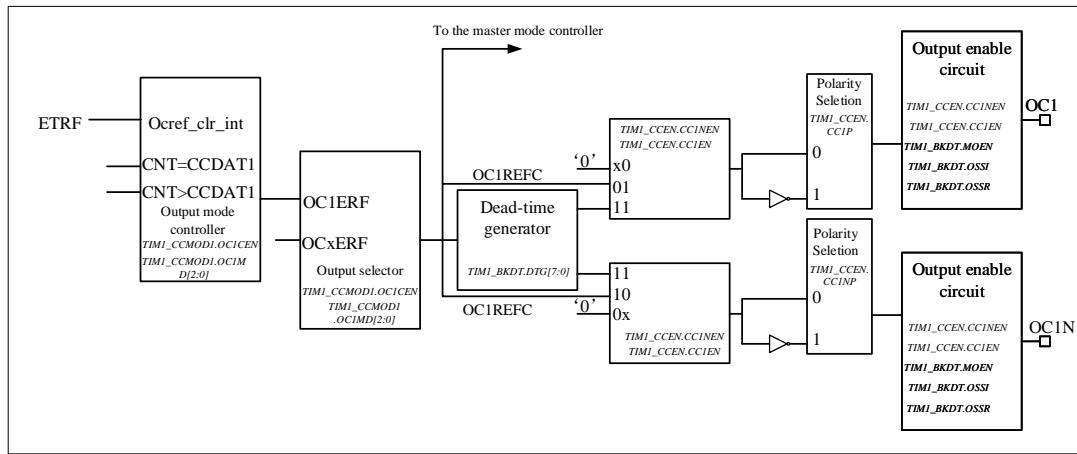
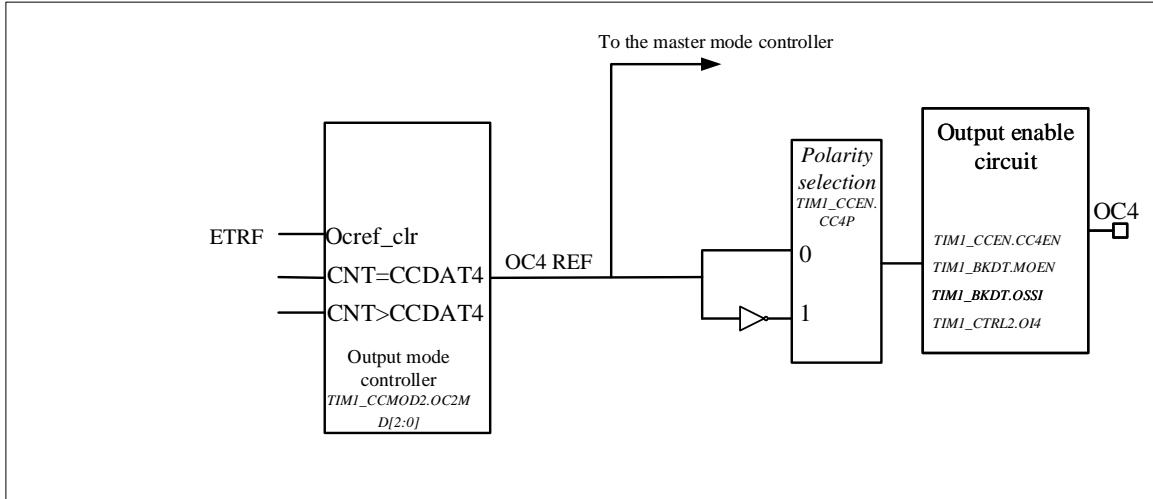


图 9-24 通道 x 的输出部分（对于 TIM8, x=4）



在捕获/比较时，读取和写入始终访问预加载的寄存器。两个具体工作流程如下：

在捕获模式下，捕获实际上是在影子寄存器中完成的，然后将影子寄存器中的值复制到预加载寄存器中。

在比较模式下，与捕获模式相反，预加载寄存器的值被复制到影子寄存器中，并与计数器进行比较。

### 9.3.6 输入捕获模式

在捕获模式下，`TIMx_CCDATx` 寄存器用于在检测到 `ICx` 信号后锁存计数器值。

有一个捕获中断标志 `TIMx_STS.CCxITF`，如果相应的中断使能被拉高，它可以发出中断或 DMA 请求。

`TIMx_STS.CCxITF` 位在发生捕获事件时由硬件设置，并由软件或读取 `TIMx_CCDATx` 寄存器清零。

当 `TIMx_CCDATx` 寄存器中的计数器值被捕获并且 `TIMx_STS.CC1ITF` 被拉高时，重复捕获标志 `TIMx_STS.CCxOCF` 设置为 1。与前者不同，`TIMx_STS.CCxOCF` 通过向其写入 0 来清除。

为实现 TI1 输入的上升沿将计数器值捕获到 `TIMx_CCDAT1` 寄存器中，配置流程如下：

- 选择有效输入：

将 TIMx\_CCMOD1.CC1SEL 配置为“01”。此时输入为 CC1 通道，IC1 映射到 TI1。

■ 编程所需的输入滤波器持续时间：

通过配置 TIMx\_CCMODx.ICxF 位来定义 TI1 输入的采样频率和数字滤波器的长度。示例：如果输入信号抖动多达 5 个内部时钟周期，我们必须选择比这 5 个时钟周期更长的滤波器持续时间。当检测到具有新电平的 8 个连续样本（以 f<sub>DTS</sub> 频率采样）时，我们可以验证 TI1 上的转换。然后配置 TIMx\_CCMOD1.IC1F 到“0011”

■ 通过配置 TIMx\_CCEN.CC1P=0，选择上升沿作为 TI1 通道的有效跳变极性

■ 配置输入预分频器。在本例中，配置 TIMx\_CCMOD1.IC1PSC=‘00’以禁用预分频器，因为我们想要捕获每个有效转换

■ 通过配置 TIMx\_CCEN.CC1EN = ‘1’启用捕获。

如果要使能 DMA 请求，可以配置 TIMx\_DINTEN.CC1DEN=1。如果要使能相关中断请求，可以配置 TIMx\_DINTEN.CC1IEN =1。

### 9.3.7 PWM 输入模式

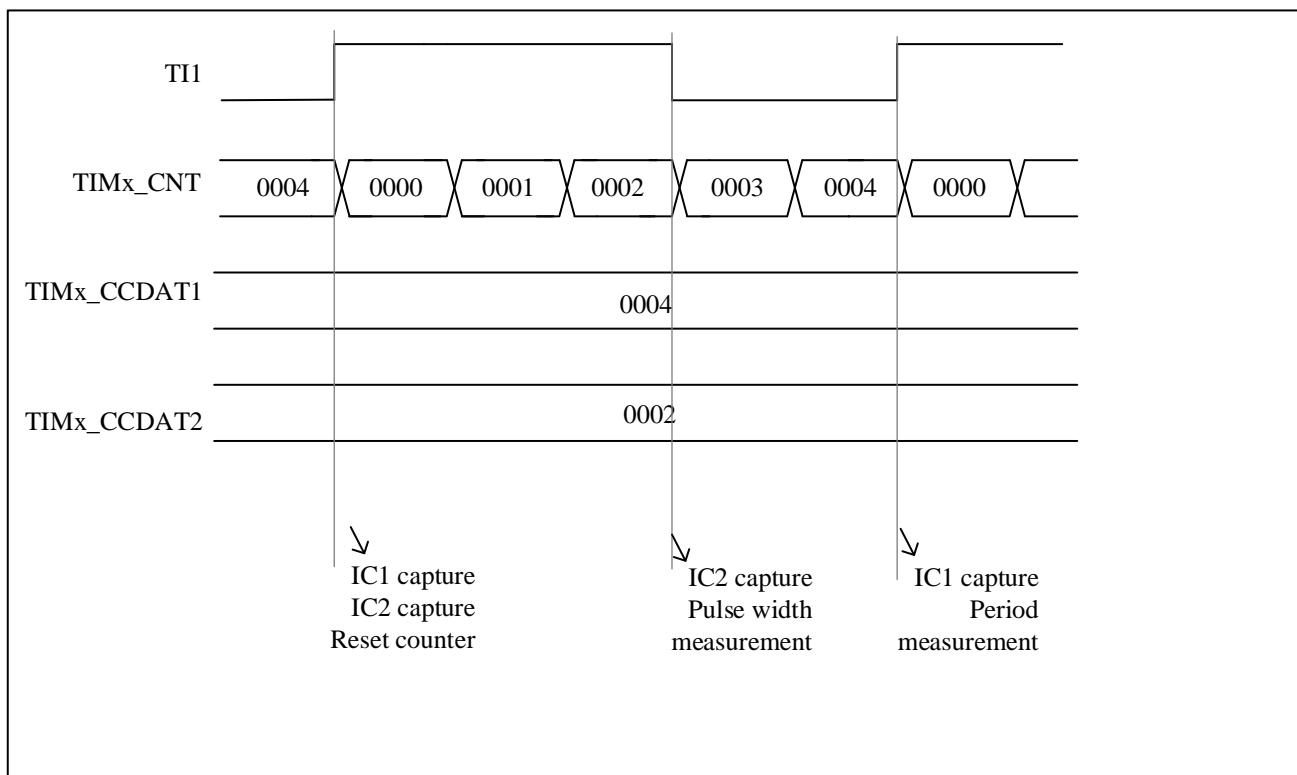
PWM 输入模式和普通输入捕获模式有一些区别，包括：

- 两个 IC<sub>x</sub> 信号映射到同一个 TI<sub>x</sub> 输入
- 两个 IC<sub>x</sub> 信号在极性相反的边沿有效
- 选择两个 TI<sub>x</sub>FP 信号之一作为触发输入
- 从机模式控制器配置为复位模式

例如，下面的配置流程可以用来知道 TI1 上 PWM 信号的周期和占空比（这取决于 CK\_INT 的频率和预分频器的值）。

- 配置 TIMx\_CCMOD1.CC1SEL 等于 ‘01’ 以选择 TI1 作为 TIMx\_CCDAT1 的有效输入
- 配置 TIMx\_CCEN.CC1P 等于 ‘0’ 选择滤波定时器输入 1(TI1FP1) 的有效极性，在上升沿有效
- 配置 TIMx\_CCMOD1.CC2SEL 等于 ‘10’ 选择 TI1 作为 TIMx\_CCDAT2 的有效输入
- 配置 TIMx\_CCEN.CC2P 等于 1 选择滤波定时器输入 2(TI1FP2) 的有效极性，下降沿有效
- 配置 TIMx\_SMCTRL.TSEL=101 选择 Filtered timer input 1 (TI1FP1) 作为有效触发输入
- 配置 TIMx\_SMCTRL.SMSEL=100 配置从模式控制器为复位模式
- 配置 TIMx\_CCEN.CC1EN=1 和 TIMx\_CCEN.CC2EN=1 以启用捕获

图 9-25 PWM 输入模式时序



由于只有滤波器定时器输入 1(TI1FP1) 和滤波器定时器输入 2(TI2FP2) 连接到从模式控制器，因此 PWM 输入模式只能与 TIMx\_CH1/TIMx\_CH2 信号一起使用。

### 9.3.8 强制输出模式

在输出模式 (TIMx\_CCMODx.CCxSEL=00) 下，软件可以直接将输出比较信号强制为有效或无效电平。

用户可以设置 TIMx\_CCMODx.OCxMD=101 强制输出比较信号为有效电平。OCxREF 将被强制为高电平，OCx 得到与 CCxP 极性位相反的值。另一方面，用户可以设置 TIMx\_CCMODx.OCxMD=100 强制输出比较信号为无效电平，即 OCxREF 被强制为低电平。

在此模式下，TIMx\_CC DATx 影子寄存器和计数器的值仍然相互比较。

输出比较寄存器 TIMx\_CC DATx 和计数器 TIMx\_CNT 之间的比较对 OCxREF 没有影响。并且仍然可以设置标志。因此，仍然可以发送中断和 DMA 请求。

### 9.3.9 输出比较模式

用户可以使用此模式来控制输出波形，或指示一段时间已过。

当捕获/比较寄存器和计数器的值相同时，输出比较函数的操作如下：

- TIMx\_CCMODx.OCxMD 为输出比较模式，TIMx\_CCEN.CCxP 为输出极性。当比较匹配时，如果设置 TIMx\_CCMODx.OCxMD=000，则输出管脚将保持其电平；如果设置 TIMx\_CCMODx.OCxMD=001，则设置输出管脚有效；如果设置 TIMx\_CCMODx.OCxMD=010，则输出管脚将为设置为无效；如果设置 TIMx\_CCMODx.OCxMD=011，则输出引脚将设置为翻转。

- 设置 TIMx\_STS.CCxITF
- 如果用户设置了 TIMx\_DINTEN.CCxIEN，将产生相应的中断
- 如果用户设置 TIMx\_DINTEN.CCxDEN 并设置 TIMx\_CTRL2.CCDSEL 选择 DMA 请求，将发送 DMA 请求

用户可以设置 TIMx\_CCMODx.OCxPEN 来选择是否使用捕获/比较预加载寄存器 (TIMx\_CCDATx) 来选择捕获/比较影子寄存器。

时间分辨率是计数器的一个计数周期。

在单脉冲模式下，输出比较模式也可用于输出单脉冲。

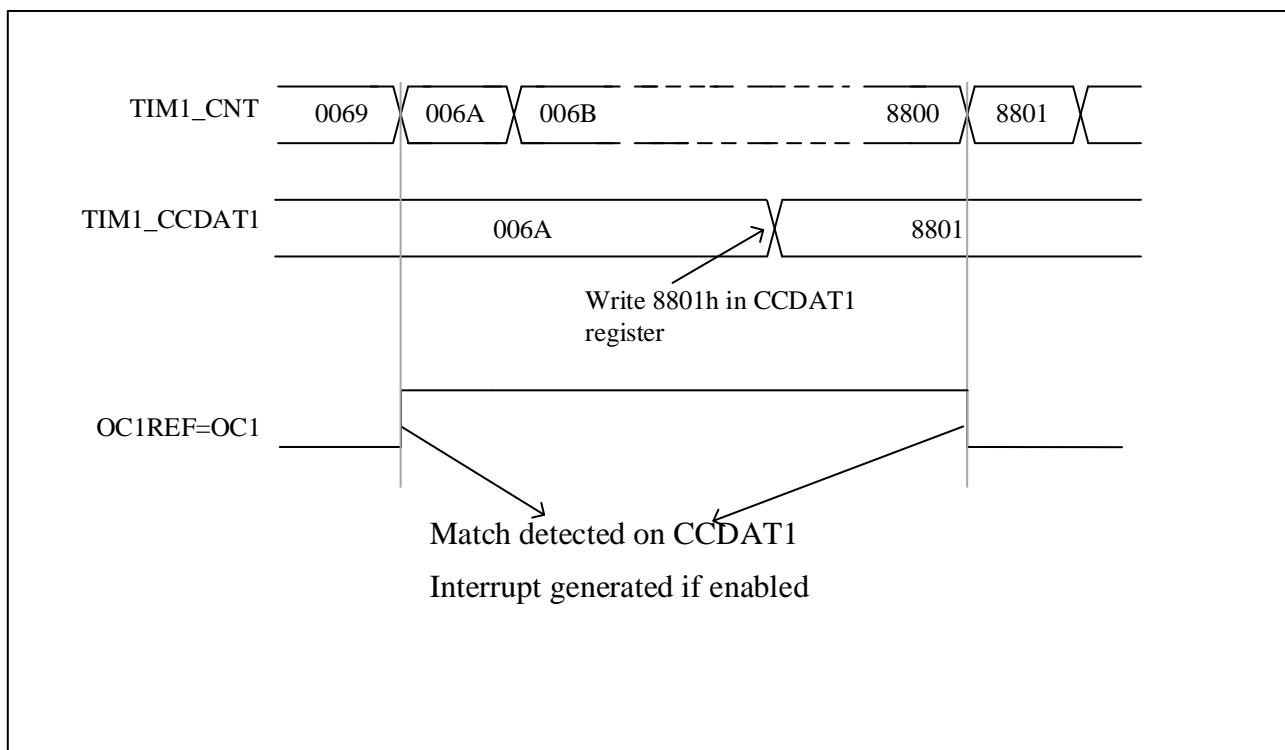
以下是输出比较模式的配置步骤：

- 首先，用户应该选择计数器时钟
- 其次，用所需数据设置 TIMx\_AR 和 TIMx\_CCDATx
- 如果用户需要产生中断，设置 TIMx\_DINTEN.CCxIEN
- 然后通过设置 TIMx\_CCEN.CCxP、TIMx\_CCMODx.OCxMD、TIMx\_CCEN.CCxEN 等选择输出模式
- 最后，设置 TIMx\_CTRL1.CNTEN 启用计数器

用户可以随时通过设置 TIMx\_CCDATx 来更新输出波形，只要不启用预加载寄存器。否则，TIMx\_CCDATx 影子寄存器将在下一次更新事件中更新。

例如：

图 9-26 输出比较模式，开启 OC1



### 9.3.10 PWM 模式

用户可以使用 PWM 模式产生一个信号，其占空比由 TIMx\_CCDATx 寄存器的值决定，其频率由 TIMx\_AR 寄存器的值决定。并且取决于 TIMx\_CTRL1.CAMSEL 的值，TIM 可以在边沿对齐模式或中央对齐模式下产生 PWM 信号。

用户可以通过设置 TIMx\_CCMODx. OCxMD=110 或设置 TIMx\_CCMODx.OCxMD=111 来设置 PWM 模式 1 或 PWM 模式 2。要使能预加载寄存器，用户必须设置相应的 TIMx\_CCMODx.OCxPEN。然后设置 TIMx\_CTRL1.AR PEN 自动重加载预加载寄存器。

用户可以通过设置 TIMx\_CCEN.CCxP 来设置 OCx 的极性。另一方面，要使能 OCx 的输出，用户需要在 TIMx\_CCEN 和 TIMx\_BKDT 中设置 CCxEN、CCxNEN、MOEN、OSSI 和 OSSR 的值的组合。

当 TIM 处于 PWM 模式时，TIMx\_CNT 和 TIMx\_CCDATx 的值总是相互比较。

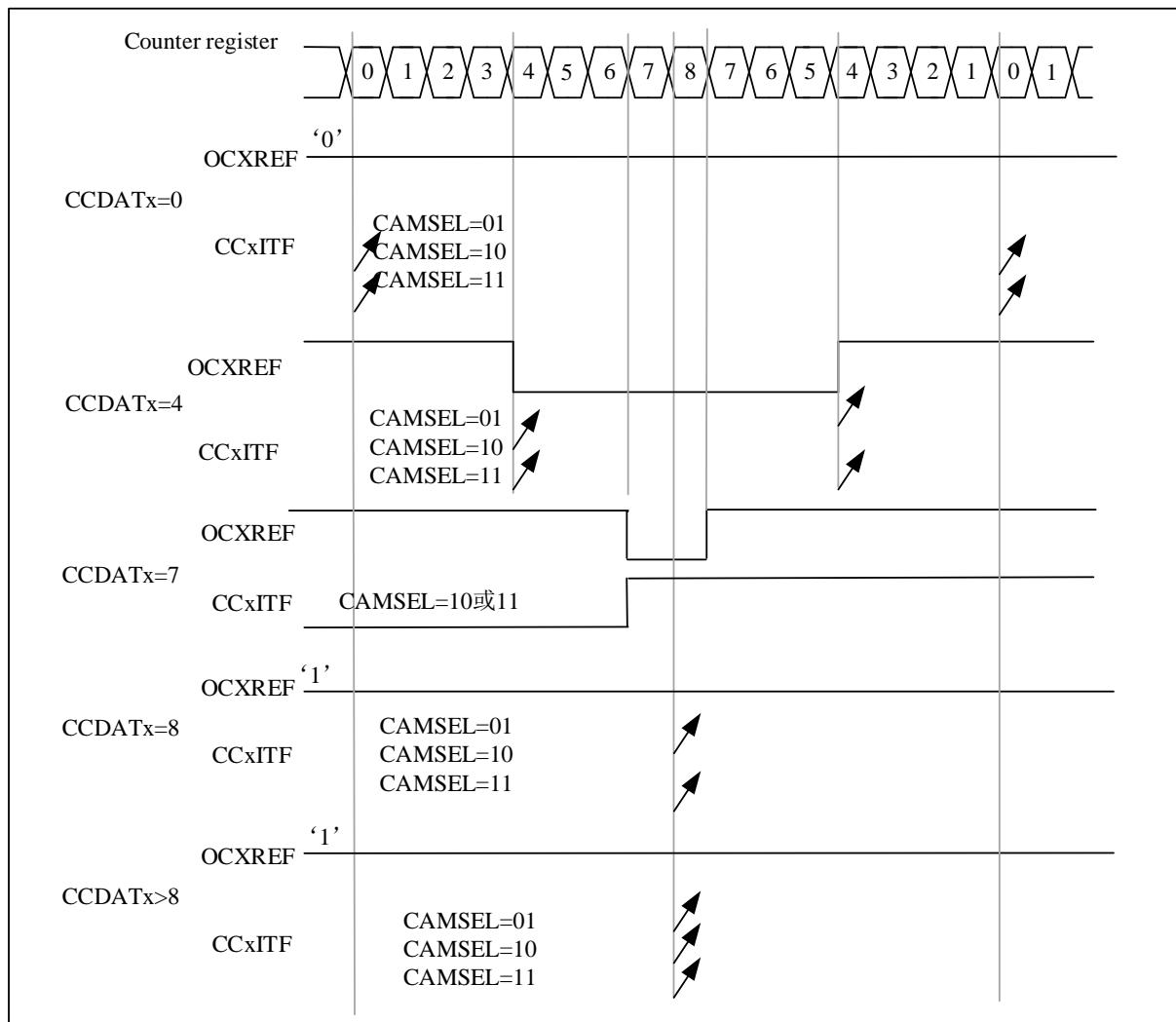
只有当更新事件发生时，预加载寄存器才会转移到影子寄存器。因此，用户必须在计数器开始计数之前通过设置 TIMx\_EVTGEN.UDGN 来复位所有寄存器。

#### 9.3.10.1 PWM 中央对齐模式

如果用户设置 TIMx\_CTRL1.CAMSEL 等于 01、10 或 11，PWM 中央对齐模式将被激活。比较标志的设置取决于 TIMx\_CTRL1.CAMSEL 的值。设置比较标志的情况有 3 种，仅当计数器向上计数时，仅当计数器向下计数时，或当计数器向上计数和向下计数时。用户不应通过软件修改 TIMx\_CTRL1.DIR，它是由硬件更新的。

中央对齐 PWM 波形示例如下，波形设置为：TIMx\_AR=8，PWM 模式 1，当计数器向下计数对应 TIMx\_CTRL1.CAMSEL=01 时设置比较标志。

图 9-27 中央对齐的 PWM 波形 (AR=8)



使用中央对齐模式时用户应注意的事项如下：

- 计数器向上或向下计数取决于 TIMx\_CTRL1.DIR 的值。注意不要同时更改 DIR 和 CAMSEL 位
- 用户在中央对齐模式下不要写计数器，否则会导致意想不到的结果。例如：
  - ◆ 如果写入计数器的值为 0 或者是 TIMx\_AR 的值，则方向会被更新，但不会产生更新事件
  - ◆ 如果写入计数器的值大于自动重载的值，则方向不会更新
- 为了安全起见，建议用户在启动计数器之前设置 TIMx\_EVTGEN.UDGN 以通过软件生成更新，并且在计数器运行时不要写入计数器

### 9.3.10.2 PWM 中央对齐非对称模式

关于 PWM 中央对齐非对称模式请查阅9.3.2.3.2。

### 9.3.10.3 PWM 边沿对齐模式

边沿对齐模式有两种配置，向上计数和向下计数。

- 向上计数

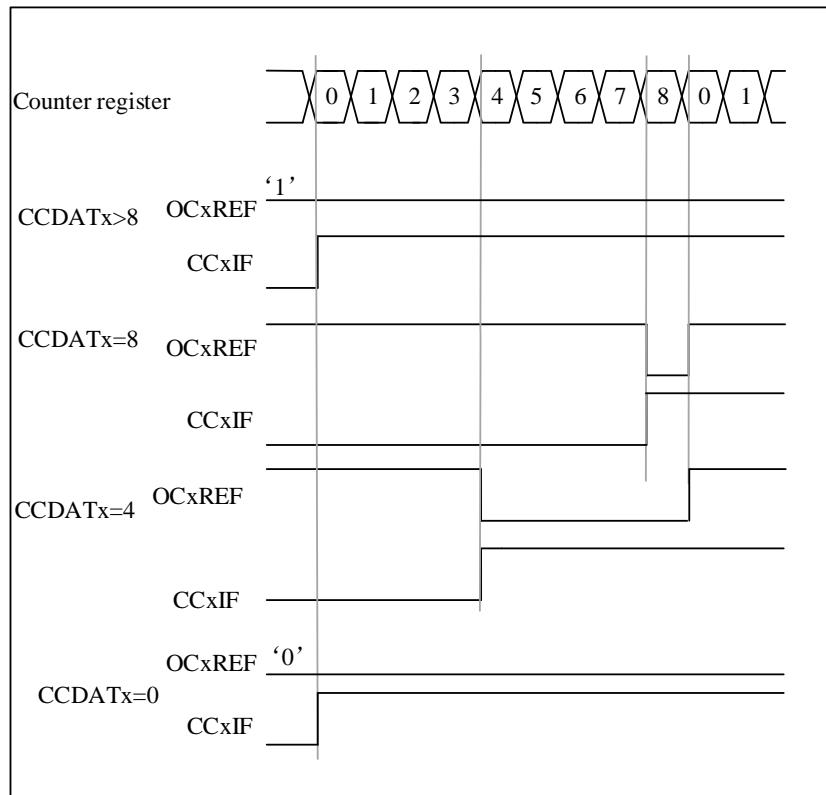
用户可以设置  $\text{TIMx\_CTRL1.DIR}=0$  使计数器向上计数。

PWM 模式 1 的示例：

当  $\text{TIMx_CNT} < \text{TIMx_CCDATx}$  时， $\text{OCxREF}$  为高电平，否则为低电平。如果  $\text{TIMx_CCDATx}$  中的比较值大于自动重载值，则  $\text{OCxREF}$  将保持为 1。相反，如果比较值为 0，则  $\text{OCxREF}$  将保持为 0。

当  $\text{TIMx_AR}=8$  时，PWM 波形如下：

图 9-28 边沿对齐 PWM 波形 ( $\text{AR}=8$ )



### ● 向下计数

用户可以设置  $\text{TIMx_CTRL1.DIR}=1$  使计数器向下计数。

PWM 模式 1 的示例：

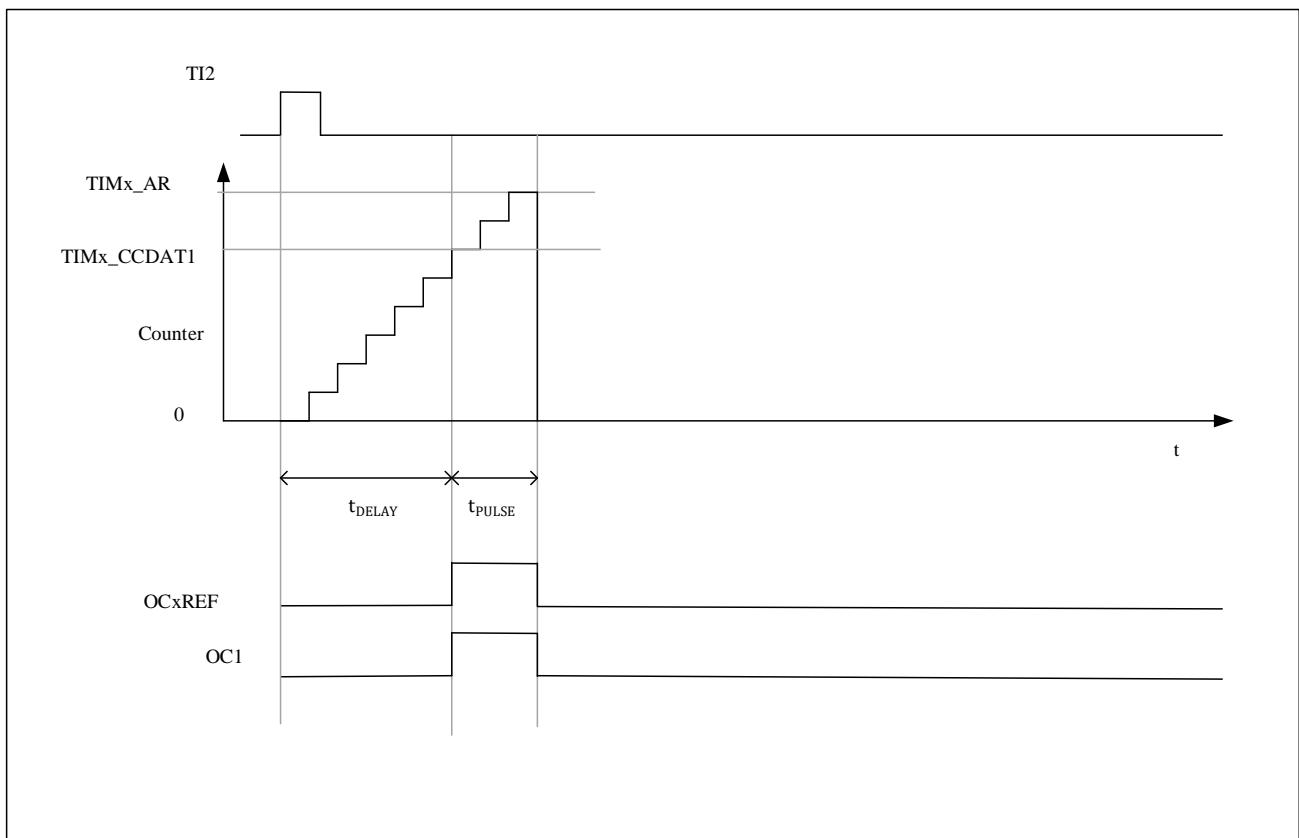
当  $\text{TIMx_CNT} > \text{TIMx_CCDATx}$  时， $\text{OCxREF}$  为低电平，否则为高电平。如果  $\text{TIMx_CCDATx}$  中的比较值大于自动重载值，则  $\text{OCxREF}$  将保持为 1。

**注：**若第  $n$  个 PWM 周期  $\text{CCDATx}$  影子寄存器  $\geq \text{AR}$  值，第  $(n+1)$  个 PWM 周期  $\text{CCDATx}$  的影子寄存器值是 0。第  $(n+1)$  个 PWM 周期的计数器为 0 的时刻，虽然计数器 =  $\text{CCDATx}$  影子寄存器的值 = 0， $\text{OCxREF} = '0'$ ，但不会产生比较事件。

### 9.3.11 单脉冲模式

在单脉冲模式(ONEPM)中，接收到触发信号，经过可控延迟  $t_{\text{DELAY}}$  后产生脉宽可控的脉冲  $t_{\text{PULSE}}$ 。输出模式需要配置为输出比较模式或 PWM 模式。选择单脉冲模式后，计数器会在更新事件 UEV 产生后停止计数。

图 9-29 单脉冲模式示例



以下是单脉冲模式的示例：

从 TI2 输入检测到上升沿触发，延迟  $t_{DELAY}$  后在 OC1 上产生宽度为  $t_{PULSE}$  的脉冲。

1. 计数器配置：向上计数，计数器  $TIMx\_CNT < TIMx\_CCDAT1 \leq TIMx\_AR$ ；
2. TI2FP2 映射到 TI2,  $TIMx\_CCMOD1.CC2SEL = '01'$ ; TI2FP2 配置为上升沿检测,  $TIMx\_CCEN.CC2P = '0'$ ;
3. TI2FP2 充当从模式控制器的触发器 (TRGI) 并启动计数器,  $TIMx\_SMCTRL.TSEL = '110'$ ,  $TIMx\_SMCTRL.SMSEL = '110'$  (触发模式);
4.  $TIMx\_CCDAT1$  写入要延迟的计数值 ( $t_{DELAY}$ ),  $TIMx\_AR - TIMx\_CCDAT1$  为脉宽  $t_{PULSE}$  的计数值;
5. 配置  $TIMx\_CTRL1.ONEPM=1$  使能单脉冲模式, 配置  $TIMx\_CCMOD1.OC1MD = '111'$  选择 PWM2 模式;
6. 等待 TI2 有外部触发事件, OC1 输出一个单脉冲波形;

### 9.3.11.1 特殊情况：OCx 快速使能：

在单脉冲模式下，通过  $TIx$  输入检测到一个边沿，并触发计数器开始计数到比较值，然后输出一个脉冲。这些操作限制了可以达到的最小延迟  $t_{DELAY}$ 。

您可以设置  $TIMx\_CCMODx.OCxFEN=1$  开启 OCx 快速使能，在触发上升沿后，OCxREF 信号将被强制转换为与比较匹配立即发生的电平相同的电平，而不管比较结果如何。OCxFEN 快速使能仅在通道模式配置为 PWM1 和 PWM2 模式时生效。

### 9.3.12 在外部事件上清除 OCxREF 信号

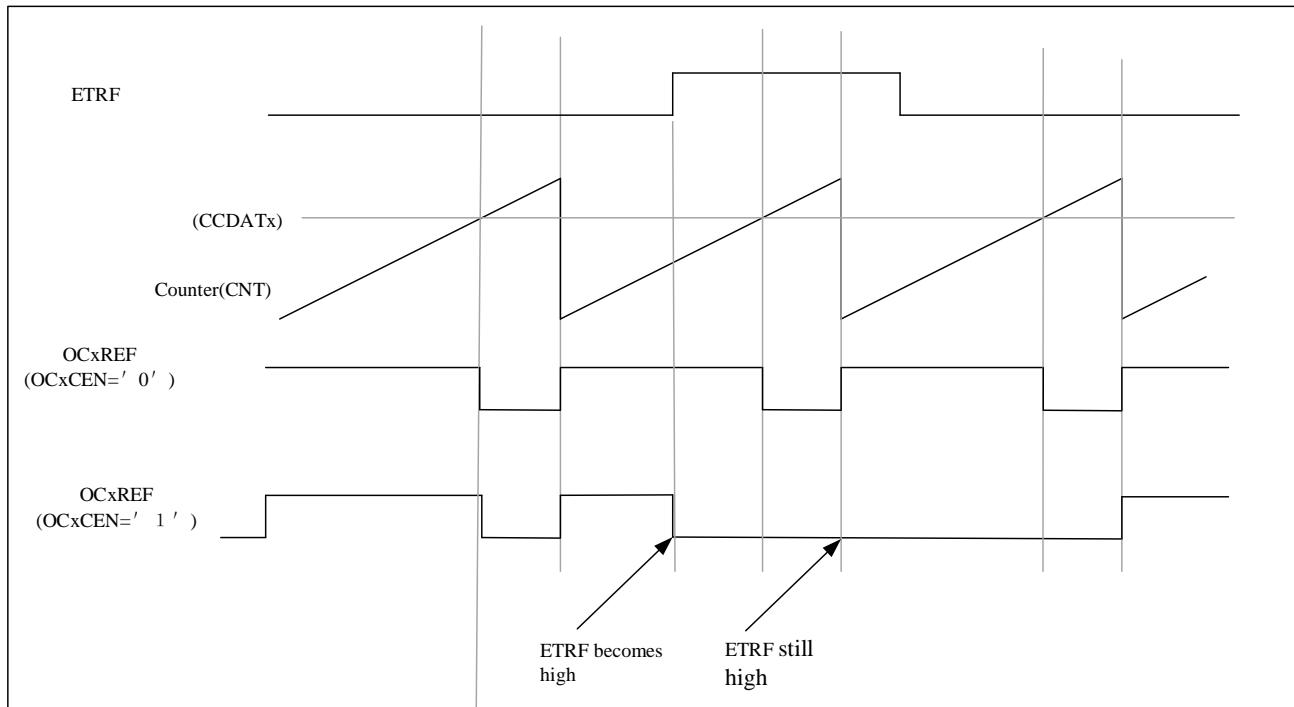
如果用户设置 `TIMx_CCMODx.OCxCEN=1`, `ETRF` 输入的高电平可用于驱动 `OCxREF` 信号为低电平, `OCxREF` 信号将保持低电平, 直到下一次 UEV 发生。只有输出比较和 PWM 模式可以使用该功能。在强制模式下不能使用。

例如: 为了控制电流, 用户可以将 `ETR` 信号连接到比较器的输出端, `ETR` 的操作如下:

- 设置 `TIMx_SMCTRL.EXTPS=00` 禁用外部触发预分频器。
- 设置 `TIMx_SMCTRL.EXCEN=0` 禁用外部时钟模式 2。
- 设置 `TIMx_SMCTRL.EXTP` 和 `TIMx_SMCTRL.EXTF`, 根据需要配置外触发极性和外触发滤波器。

例: 当 `ETRF` 输入变高时, `OCxREF` 信号对于不同的 `OCxCEN` 值的行为。在这种情况下, 定时器设置为 PWM 模式。

图 9-30 清除 TIMx 的 `OCxREF`



### 9.3.13 互补输出和死区插入

高级控制定时器可以输出两个互补信号, 并管理输出的关闭和打开。这称为死区时间。用户应根据连接到输出的设备及其特性调整死区时间。

用户可以通过设置 `TIMx_CCEN.CCxP` 和 `TIMx_CCEN.CCxNP` 来选择输出的极性。并且此选择对于每个输出都是独立的。

用户可以通过设置几个控制位的组合来控制互补信号 `OCx` 和 `OCxN`, 它们分别是 `TIMx_CCEN.CCxEN`、

`TIMx_CCEN.CCxNEN`、`TIMx_BKDT.MOEN`、`TIMx_CTRL2.OIx`、`TIMx_CTRL2.OIxN`、`TIMx_BKDT.OSSI` 和 `TIMx_BKDT.OSSR`。当切换到空闲状态时，死区时间将被激活。

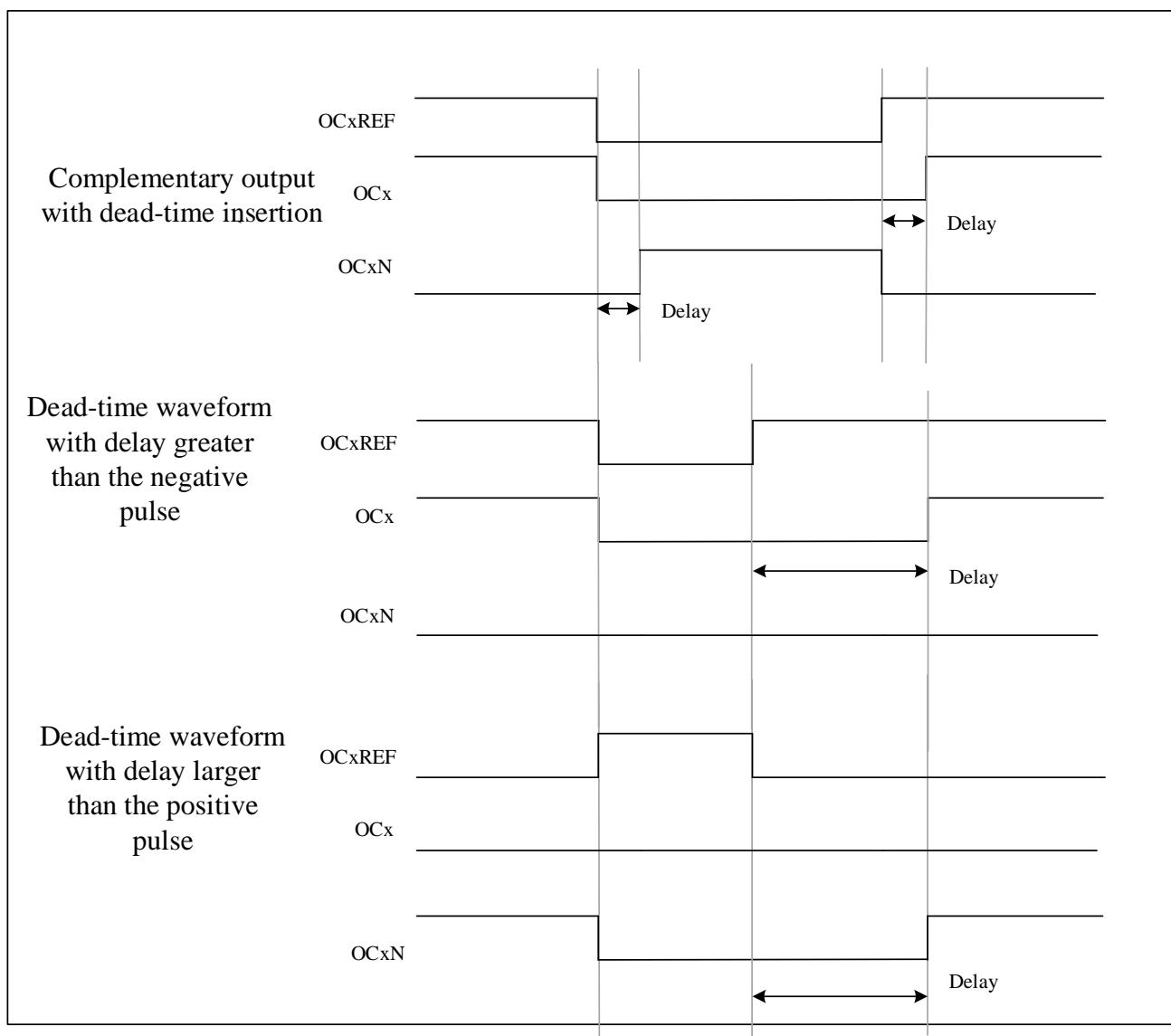
如果用户同时设置 `TIMx_CCEN.CCxEN` 和 `TIMx_CCEN.CCxNEN`，则会插入死区时间。如果有刹车，还要设置 `TIMx_BKDT.MOEN`。每个通道都有 10 位死区时间发生器。

参考波形 `OCxREF` 可以生成 2 个输出 `OCx` 和 `OCxN`。如果 `OCx` 和 `OCxN` 为高电平有效，则 `OCx` 输出信号与参考信号相同，而 `OCxN` 输出信号与参考信号相反。但是，`OCx` 输出信号将相对于参考上升沿延迟，而 `OCxN` 输出信号将相对于参考下降沿延迟。如果延迟大于有效 `OCx` 或 `OCxN` 输出的宽度，则不会产生相应的脉冲。

死区时间发生器的输出信号与参考信号 `OCxREF` 之间的关系如下。

假设 `TIMx_CCEN.CCxP=0`, `TIMx_CCEN.CCxNP=0`, `TIMx_BKDT.MOEN=1`, `TIMx_CCEN.CCxEN=1`, `TIMx_CCEN.CCxNEN=1`。

图 9-31 带死区插入的互补输出



用户可以设置 `TIMx_BKDT.DTGN` 来编程每个通道的死区时间延迟。

### 9.3.13.1 重定向 OCxREF 到 OCx 或 OCxN

在输出模式下，用户可以设置 `TIMx_CCEN.CCxEN` 和 `TIMx_CCEN.CCxNEN` 以将 `OCxREF` 重定向到 `OCx` 输出或 `OCxN` 输出。

这里有两种使用这个功能的方法。当互补保持在其无效电平时，用户可以使用此功能发送特定波形，例如 `PWM` 或静态有效电平。用户还可以使用此功能将两个输出设置为无效电平，或将两个输出都设置为有效并与死区时间互补。

如果用户设置 `TIMx_CCEN.CCxEN=0` 和 `TIMx_CCEN.CCxNEN=1`，两者不互补，当 `OCxREF` 为高电平时 `OCxN` 将变为有效。另一方面，如果用户设置 `TIMx_CCEN.CCxEN=1` 和 `TIMx_CCEN.CCxNEN=1`，当 `OCxREF` 为高电平时，`OCx` 将变为有效。相反，当 `OCxREF` 为低电平时，`OCxN` 将变为有效。

### 9.3.14 刹车功能

使用刹车功能时，设置相应的控制位时会修改输出使能信号和无效电平。但是，无论何时，`OCx` 和 `OCxN` 的输出都不能同时处于有效电平，即需要满足  $(CCxP \wedge OIx) \wedge (CCxNP \wedge OIxN) != 0$ 。

当启用多个刹车信号时，每个刹车信号构成一个 `OR` 逻辑。这里有一些信号可能是刹车的来源。

- 刹车输入引脚
- 时钟失效事件，由时 RCC 中的时钟安全系统（CSS）生成。
- PVD 事件。
- 内核 Hardfault 事件。
- 比较器的输出信号（在比较器模块中配置，高电平刹车）。
- 软件设置 `TIMx_EVTGEN.BGN`。

复位后刹车电路将被禁用。`MOEN` 位将为低电平。用户可以设置 `TIMx_BKDT.BKEN` 来启用刹车功能。通过设置 `TIMx_BKDT.BKP` 可以选择刹车输入信号的极性。用户可以同时修改 `TIMx_BKDT.BKEN` 和 `TIMx_BKDT.BKP`。用户设置 `TIMx_BKDT.BKEN` 和 `TIMx_BKDT.BKP` 后，生效前有 1 个 APB 时钟周期延迟。因此，用户需要等待 1 个 APB 时钟周期才能读回写入位的值。

`MOEN` 的下降沿可以是异步的，所以在实际信号和同步控制位之间设置了一个再同步电路。该电路将导致异步和同步信号之间的延迟。当用户设置 `TIMx_BKDT.MOEN` 为低电平时，用户需要在读取该值之前插入一个延迟。因为写入了异步信号，但用户读取了同步信号。

刹车发生后的行为如下：

- `TIMx_BKDT.MOEN` 将被异步清除，然后输出将进入无效状态、空闲状态或复位状态。通过设置 `TIMx_BKDT.OSSI` 选择输出状态。即使 MCU 振荡器关闭，这也会生效。
- 一旦 `TIMx_BKDT.MOEN=0`，每个输出通道的输出将使用 `TIMx_CTRL2.OIx` 中编程的电平驱动。如果 `TIMx_BKDT.OSSI=0`，定时器将释放使能输出（由 GPIO 控制器接管），否则将保持高电平。
- 如果用户选择使用互补输出，TIM 的行为如下
  - 取决于极性，输出将首先设置为复位状态。它是一个异步选项，因此即使没有为计时器提供时钟，它仍然可以工作。
  - 如果仍然提供定时器时钟，死区发生器将重新激活，当  $(CCxP \wedge OIx) \wedge (CCxNP \wedge OIxN) != 0$ ，

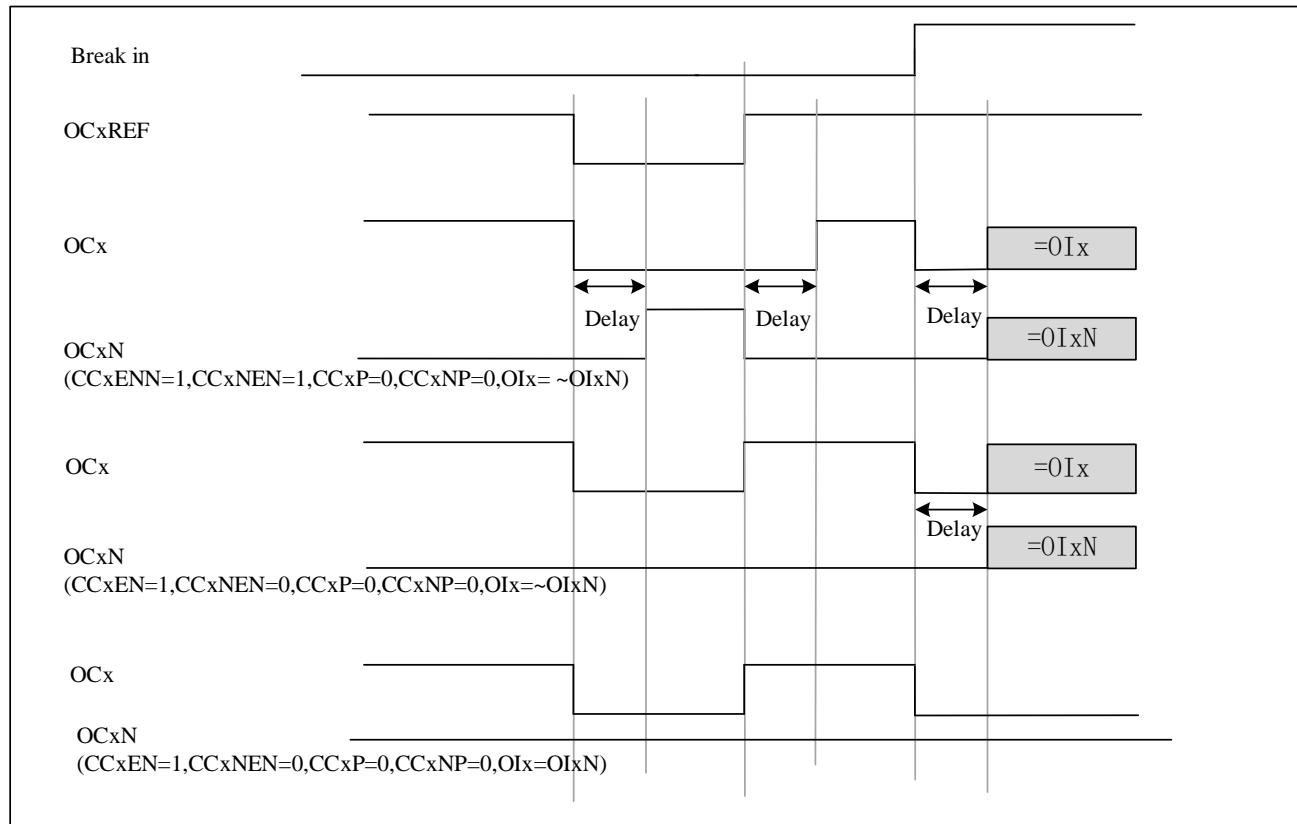
即 OC<sub>x</sub> 和 OC<sub>xN</sub> 仍然不能同时被驱动到有效电平，在死区时间后时根据 TIM<sub>x</sub>\_CTRL2.OIx 和 TIM<sub>x</sub>\_CTRL2.OIxN 的值驱动输出。请注意，由于 MOEN 上的重新同步（大概 2 个 ck\_tim 周期），死区时间将比平时长。

- 如果 TIM<sub>x</sub>\_BKDT.OSSI=0，定时器将释放输出控制。否则，如果使能输出为高电平，它将保持为高电平。如果为低电平，则在 TIM<sub>x</sub>\_CCEN.CCxEN 或 TIM<sub>x</sub>\_CCEN.CCxNEN 为高电平时变为高电平。
- 如果 TIM<sub>x</sub>\_DINTEN.BIEN=1，当 TIM<sub>x</sub>\_STS.BITF=1 时，会产生中断。
- 如果用户设置了 TIM<sub>x</sub>\_BKDT.AOEN，TIM<sub>x</sub>\_BKDT.MOEN 将在下一次 UEV 发生时自动设置。用户可以使用它来调节。如果用户未设置 TIM<sub>x</sub>\_BKDT.AOEN，则 TIM<sub>x</sub>\_BKDT.MOEN 将保持低电平，直到再次设置为 1。在这种情况下，用户可以使用它来保证安全。用户可以将刹车输入连接到热传感器、电源驱动器警报或其他安全组件。
- 刹车输入有效时，TIM<sub>x</sub>\_BKDT.MOEN 不能自动置位或软件同时置位，TIM<sub>x</sub>\_STS.BITF 也不能清零。因为刹车输入在电平上处于有效状态。

为保证应用安全，刹车电路具有写保护功能，并有刹车输入输出管理。它允许用户冻结一些参数，例如死区持续时间、OC<sub>x</sub>/OC<sub>xN</sub> 极性和禁用时的状态、OC<sub>x</sub>MD 配置、刹车启用和极性。用户可以通过设置 TIM<sub>x</sub>\_BKDT.LCKCFG 选择使用 3 种保护级别之一。但是，TIM<sub>x</sub>\_BKDT.LCKCFG 只能在 MCU 复位后写入一次。

响应刹车的输出行为示例如下

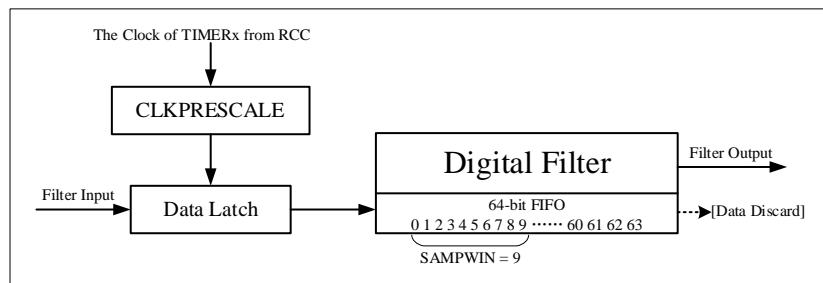
图 9-32 响应刹车的输出行为



### 9.3.14.1 刹车滤波

寄存器 TIM1\_BRKFILT 描述如下：

图 9-33 滑动滤波



- 数字滤波器通过 RCC 的 TIM1 时钟采样刹车信号，在 64 位 FIFO 中累积采样。仅在 TIM1\_BRKFILT.WSIZE [5:0] 中定义的窗口大小内采样数据，最大大小为 64。
- 过滤器输出采样窗口内的多数值，该值由 TIM1\_BRKFILT.THRESH [5:0] 中的阈值定义，最大阈值为 63。此值应等于或大于窗口大小的一半。如果采样窗口内的逻辑 1 和逻辑 0 计数均不大于阈值，则数字滤波器保持先前的输出值。
- TIM1\_BRKFILT.PSC 寄存器决定相应数字滤波器的采样率。过滤器 FIFO 在每个采样时钟从输入中捕获一个采样值。
- 如果数字滤波器关闭，滤波器输入将像电线一样绕过输出。

### 9.3.15 调试模式

当微控制器处于调试模式 (Cortex-M4 内核停止) 时，根据 DBG 模块中的 DBG\_TIMx\_STOP 配置，TIMx 计数器可以继续正常工作或停止。详见 23.4.3。

### 9.3.16 TIMx 定时器和外部触发的同步

TIMx 定时器可以通过从模式 (复位、触发和门控) 中的触发器进行同步。

#### 9.3.16.1 从模式：复位模式

在复位模式下，触发事件可以复位计数器和预分频器。更新预加载寄存器 TIMx\_AR、TIMx\_CC DATx，并产生更新事件 UEV (TIMx\_CTRL1.UPRS=0)。

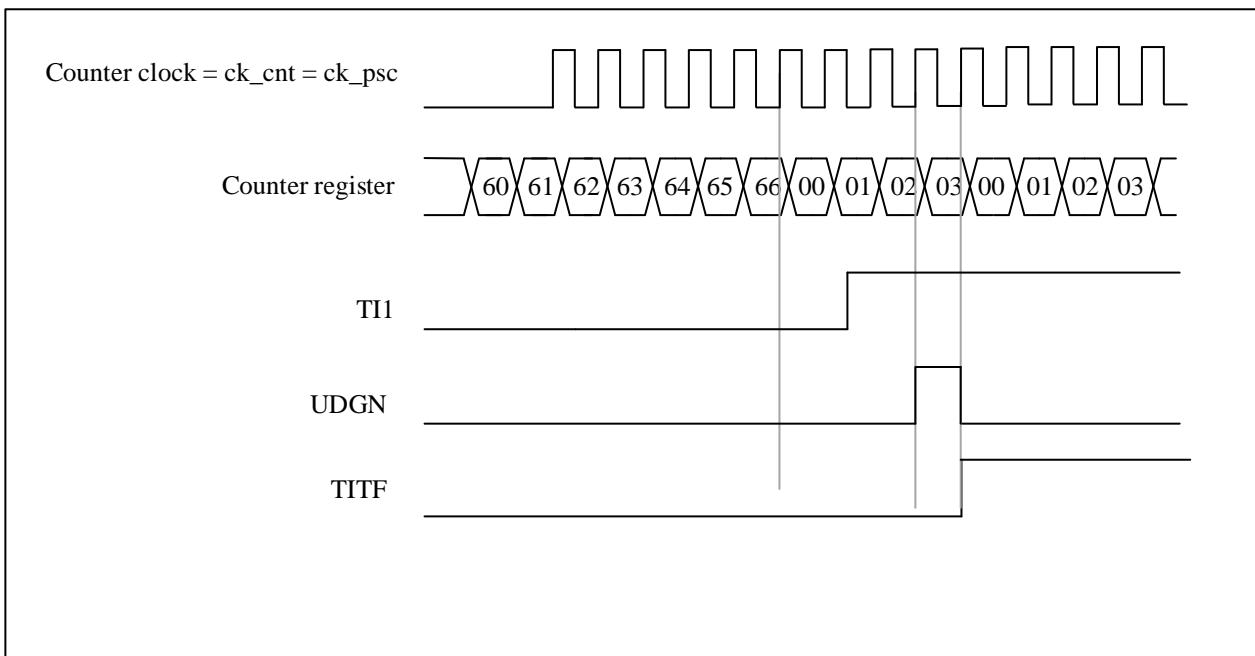
以下是复位模式的示例：

1. 通道 1 配置为输入检测 TI1 的上升沿 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=0);
2. 从模式选择为复位模式 (TIMx\_SMCTRL.SMSEL=100)，触发输入选择为 TI1 (TIMx\_SMCTRL.TSEL=101);
3. 设置 TIMx\_CTRL1.CNTEN = 1 来启动定时器

启动定时器后，当 TI1 检测到上升沿时，计数器复位并重新开始计数，并设置触发标志 (TIMx\_STS.TITF=1)；

TI1 的上升沿与实际计数器复位之间的延迟是由于 T11 输入的重新同步电路引起的。

图 9-34 复位模式下的控制电路



### 9.3.16.2 从模式：触发模式

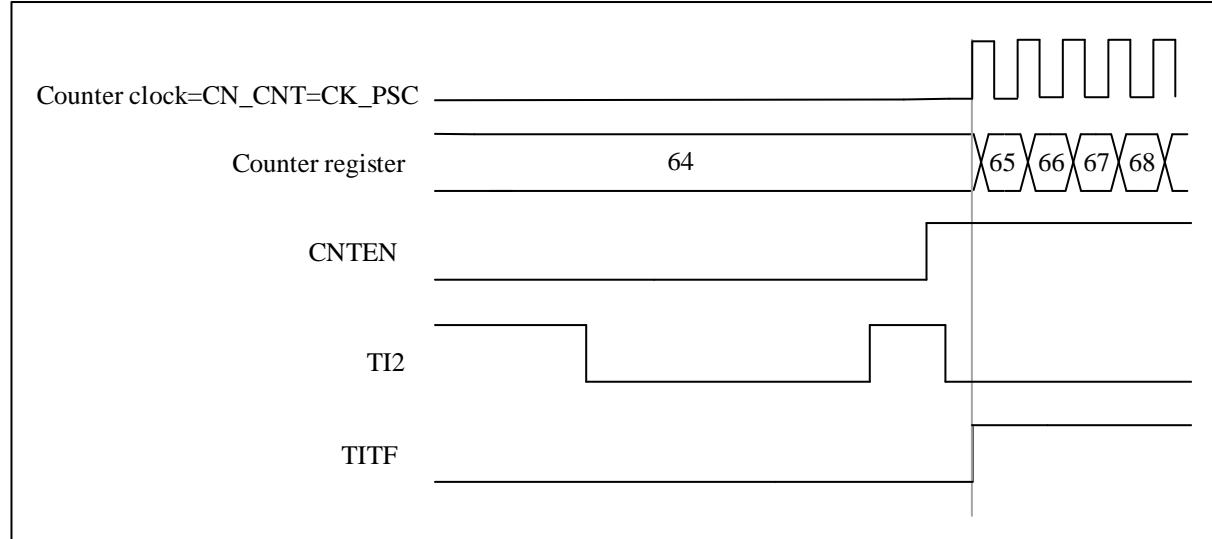
在触发模式下，输入端口的触发事件（上升沿/下降沿）可以触发计数器开始计数。

以下是触发模式的示例：

1. 通道 2 配置为输入，检测 TI2 的上升沿 (TIMx\_CCMOD1.CC2SEL=01, TIMx\_CCEN.CC2P=0);
  2. 选择从模式为触发模式 (TIMx\_SMCTRL.SMSEL=110)，触发输入选择 TI2 (TIMx\_SMCTRL.TSEL=110);
- 当 TI2 检测到上升沿时，计数器开始计数，触发标志置位 (TIMx\_STS.TITF=1)；

TI2 的上升沿与实际计数器启动之间的延迟是由于 T12 输入的重新同步电路引起的。

图 9-35 触发器模式下的控制电路



### 9.3.16.3 从模式：门控模式

在门控模式下，输入端口的电平极性可以控制计数器是否计数。

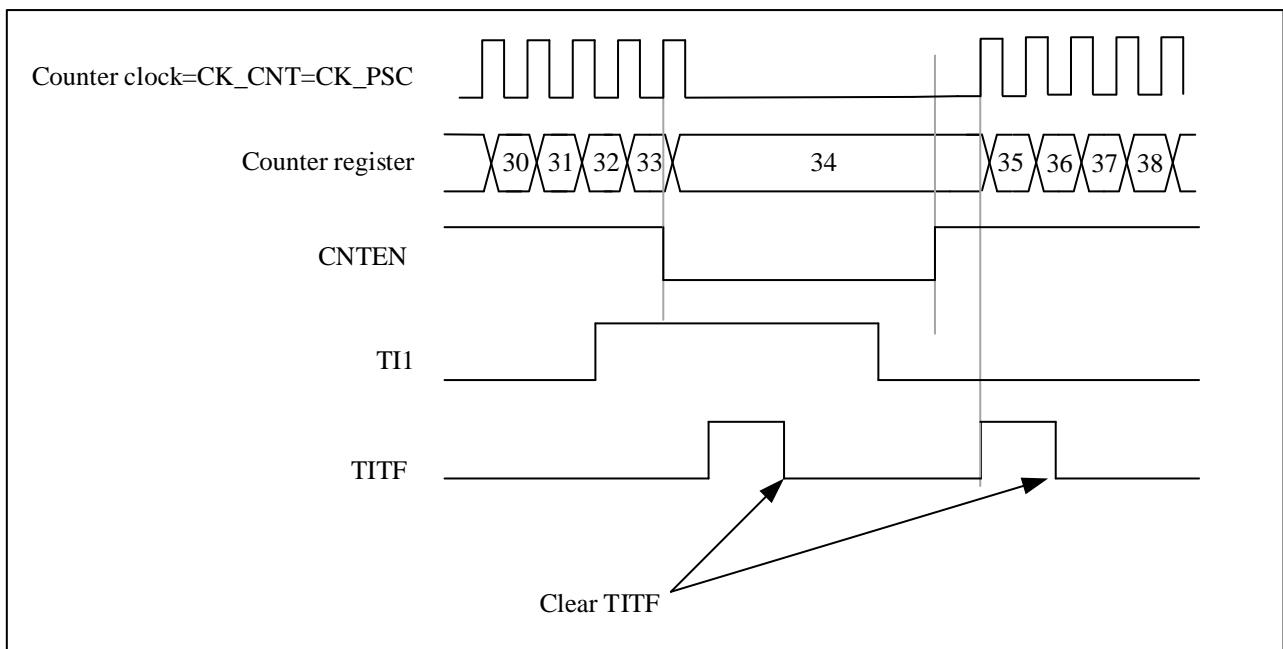
以下是门控模式的示例：

1. 通道 1 配置为 TI1 上的输入检测低电平有效 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=1);
2. 选择从模式为门控模式 (TIMx\_SMCTRL.SMSEL=101)，选择 TI1 作为 TRGI (TIMx\_SMCTRL.TSEL=101);
3. 设置 TIMx\_CTRL1.CNTEN = 1 来启动定时器

当 TI1 检测到电平由低变高时，计数器停止计数，当 TI1 检测到电平由高变低时，计数器开始计数，开始或停止计数时触发标志置位 (TIMx\_STS.TITF=1);

TI1 的上升沿与实际计数器停止之间的延迟是由于 T11 输入的重新同步电路引起的。

图 9-36 门控模式下的控制电路



### 9.3.16.4 从模式：触发模式 + 外部时钟模式 2

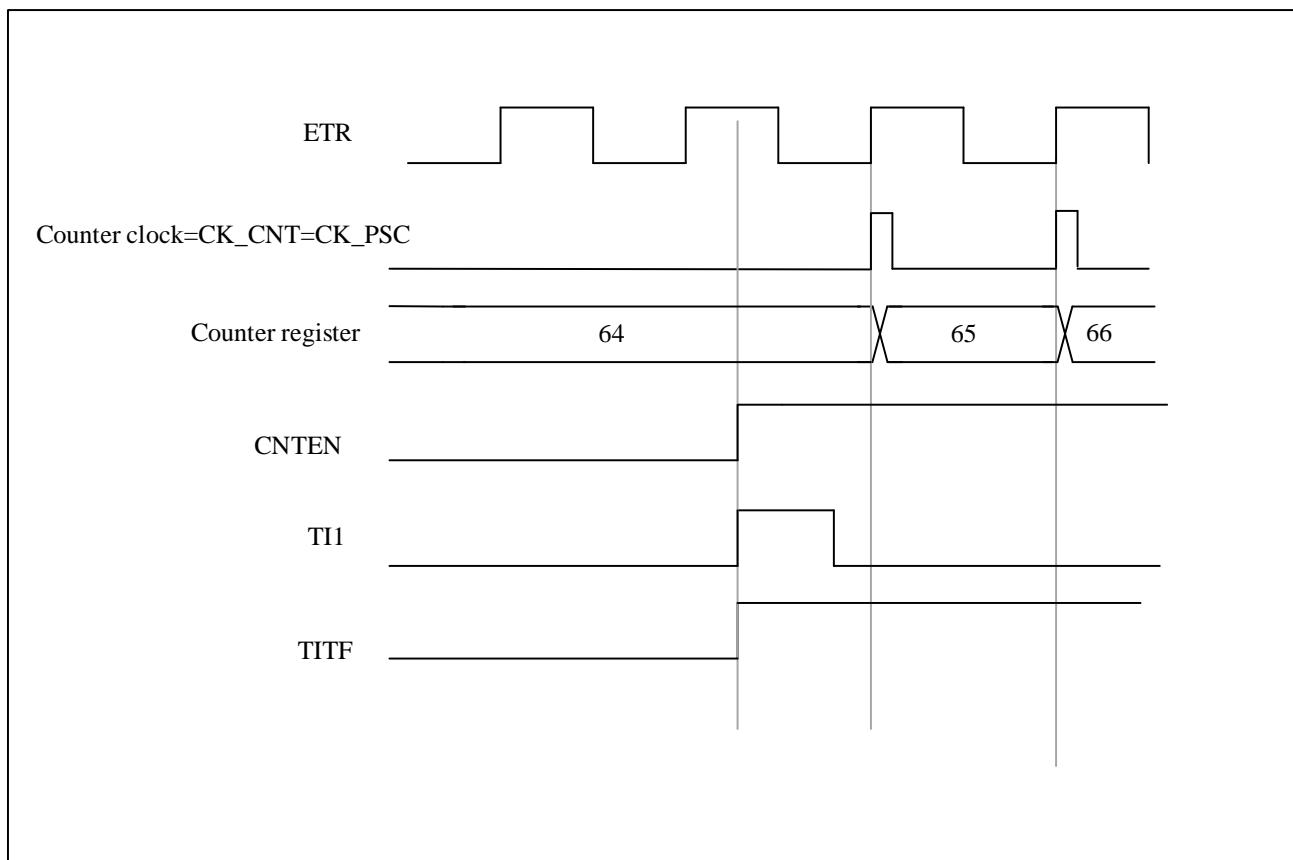
在复位模式、触发模式和门控模式下，计数器时钟可选择为外部时钟模式 2，ETR 信号作为外部时钟源输入。这时候触发选择需要选择非 ETRF (TIMx\_SMCTRL.TSEL=111)。

这是一个例子：

1. 通道 1 配置为输入检测 TI1 的上升沿 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=0);
2. 使能外部时钟模式 2 (TIMx\_SMCTRL.EXCEN=1), 外部触发极性选择上升沿 (TIMx\_SMCTRL.EXTP=0), 触发模式作为从模式 (TIMx\_SMCTRL.SMSEL=110), TRGI 选择 TI1 (TIMx\_SMCTRL.TSEL=101);

当 TI1 检测到上升沿时，计数器在 ETR 的上升沿开始计数，并设置触发标志 (TIMx\_STS.TITF=1);

图 9-37 外部时钟模式 2+触发模式下的控制电路



### 9.3.17 定时器同步

所有 TIM 定时器在内部相连，用于定时器同步或链接。详见 10.3.14 章节。

### 9.3.18 产生六步 PWM 输出

为了同时修改所有通道的配置，可以提前设置下一步的配置（预加载位为 OCxMD、CCxEN 和 CCxNEN）。当发生 COM 换相事件时，OCxMD、CCxEN 和 CCxNEN 预加载位被传送到影子寄存器位。

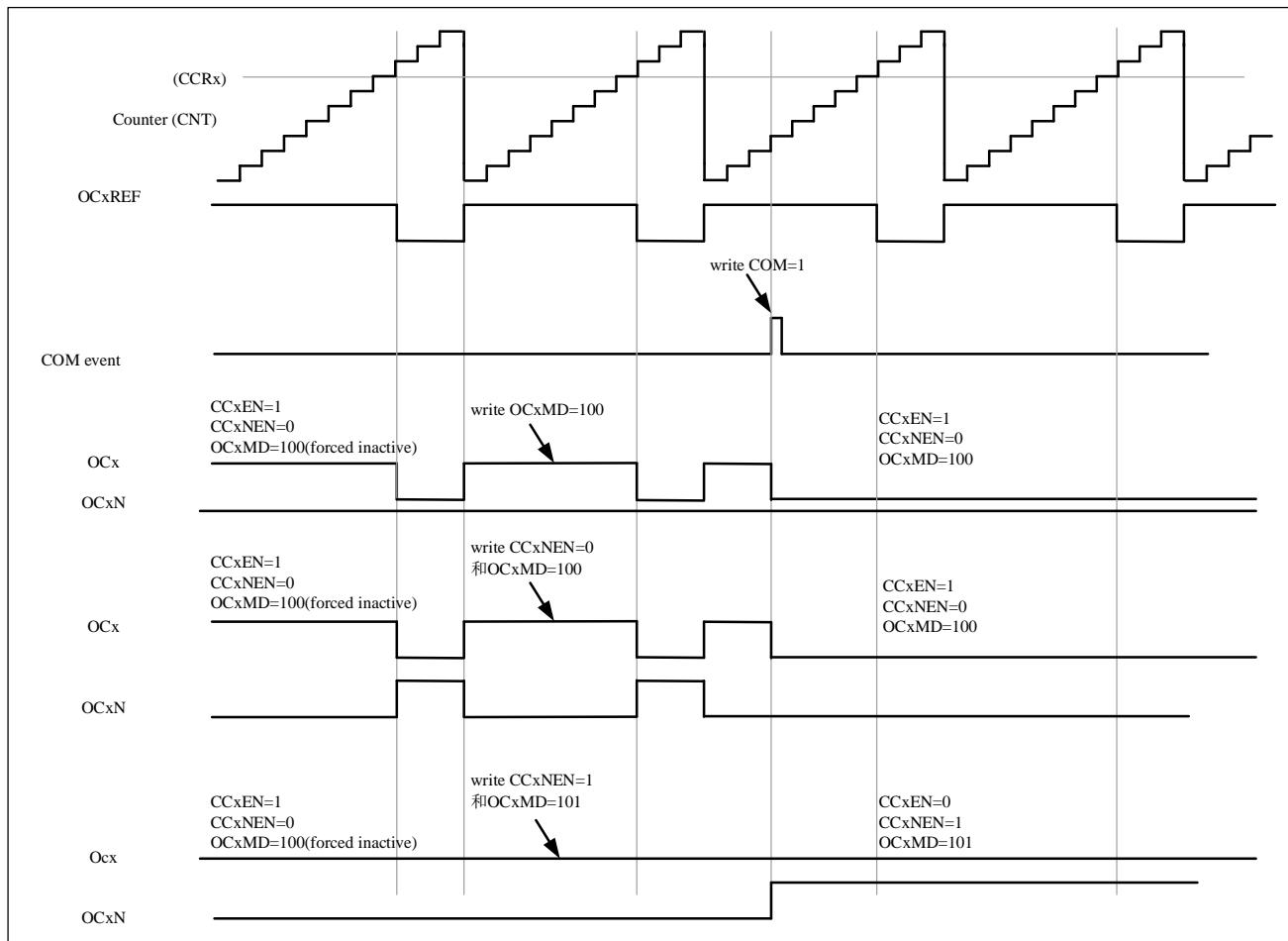
COM 换相事件生成方法：

1. 软件设置 TIMx\_EVTGEN.CCUDGN；
2. 在 TRGI 的上升沿由硬件产生；

当 COM 换相事件发生时，TIMx\_STS.COMITF 标志将被设置，启用中断 (TIMx\_DINTEN.COMIEN) 将产生中断，启用 DMA 请求 (TIMx\_DINTEN.COMDEN) 将产生 DMA 请求。

下图显示了三种不同配置下发生 COM 换向事件时 OCx 和 OCxN 的输出时序图：

图 9-38 产生六步 PWM，使用 COM 的例子（OSSR=1）



### 9.3.19 编码器接口模式

编码器使用两个输入 TI1 和 TI2 作为接口，计数器对 TI1FP1 或 TI2FP2 上的每个边沿变化进行计数。计数方向由硬件 TIM<sub>x</sub>\_CTRL1.DIR 自动控制。编码器计数模式共有三种：

1. 计数器只在 TI1 的边沿计数，TIM<sub>x</sub>\_SMCTRL.SMSEL = ‘001’；
2. 计数器只在 TI2 的边沿计数，TIM<sub>x</sub>\_SMCTRL.SMSEL = ‘010’；
3. 计数器同时在 TI1 和 TI2 的边沿计数，TIM<sub>x</sub>\_SMCTRL.SMSEL = ‘011’；

编码器接口相当于使用带方向选择的外部时钟，计数器只在 0 和自动重载值 (TIM<sub>x</sub>\_AR.AR [15:0]) 之间连续计数。因此，需要提前配置自动重载寄存器 TIM<sub>x</sub>\_AR。

**注意：**编码器模式和外部时钟模式 2 不兼容，不能同时选择。

计数方向与编码器信号的关系如下表：

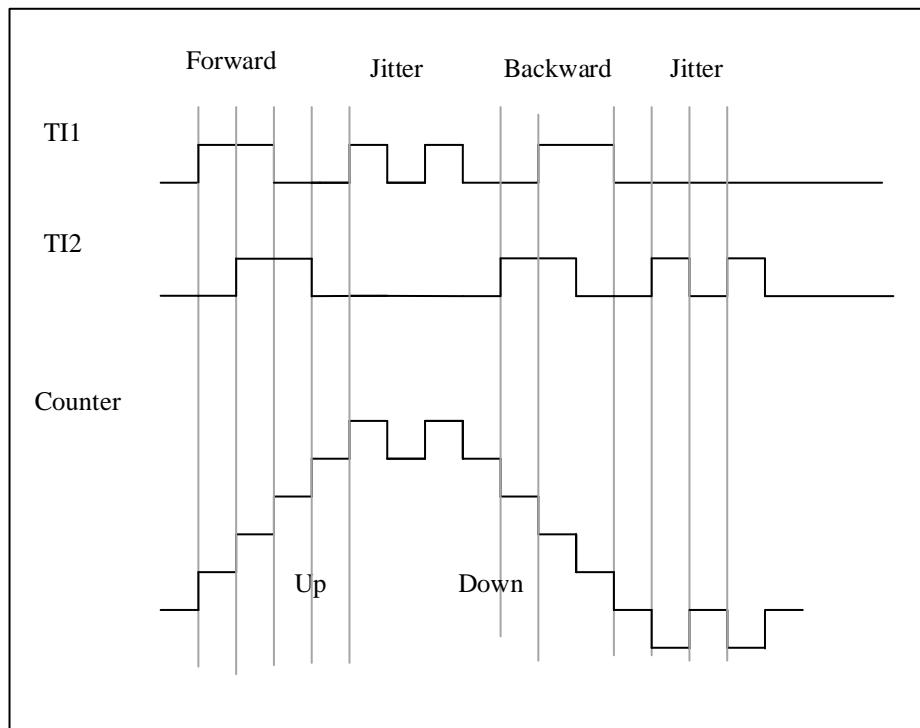
表 9-1 计数方向与编码器信号的关系

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

以下是选择了双边沿触发以抑制输入抖动的编码器示例：

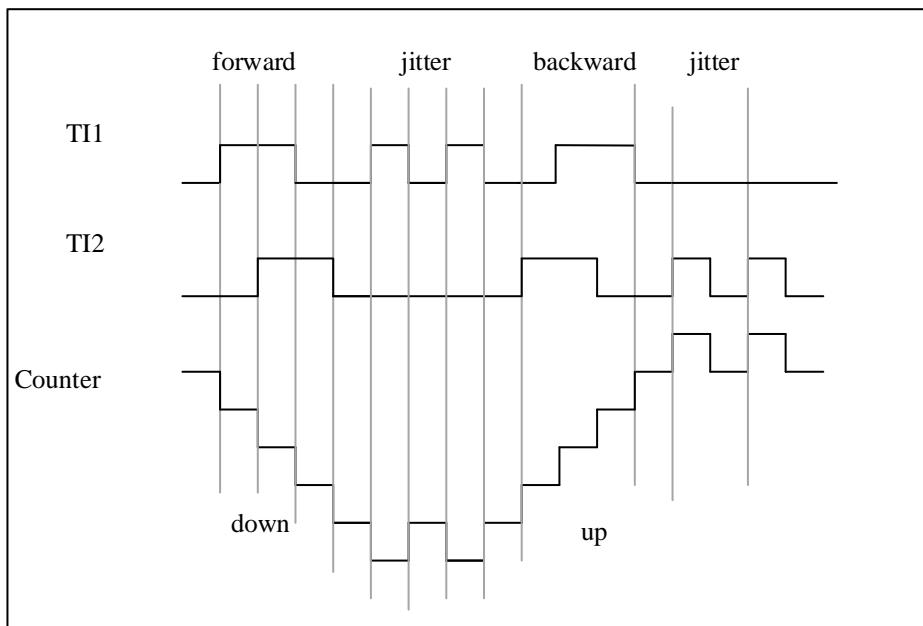
1. IC1FP1 映射到 TI1 (TIMx\_CCMOD1.CC1SEL='01'), IC1FP1 不反相 (TIMx\_CCEN.CC1P='0');
2. IC1FP2 映射到 TI2 (TIMx\_CCMOD2.CC2SEL='01'), IC2FP2 不反相 (TIMx\_CCEN.CC2P='0');
3. 输入在上升沿和下降沿均有效 (TIMx\_SMCTRL.SMSEL='011');
4. 启用计数器 TIMx\_CTRL1.CNTEN='1';

图 9-39 编码器模式下的计数器操作实例



下图为 IC1FP1 极性反转时的计数器行为示例（CC1P='1'，其他配置同上）

图 9-40 IC1FP1 反相的编码器接口模式实例



### 9.3.20 与霍尔传感器的接口

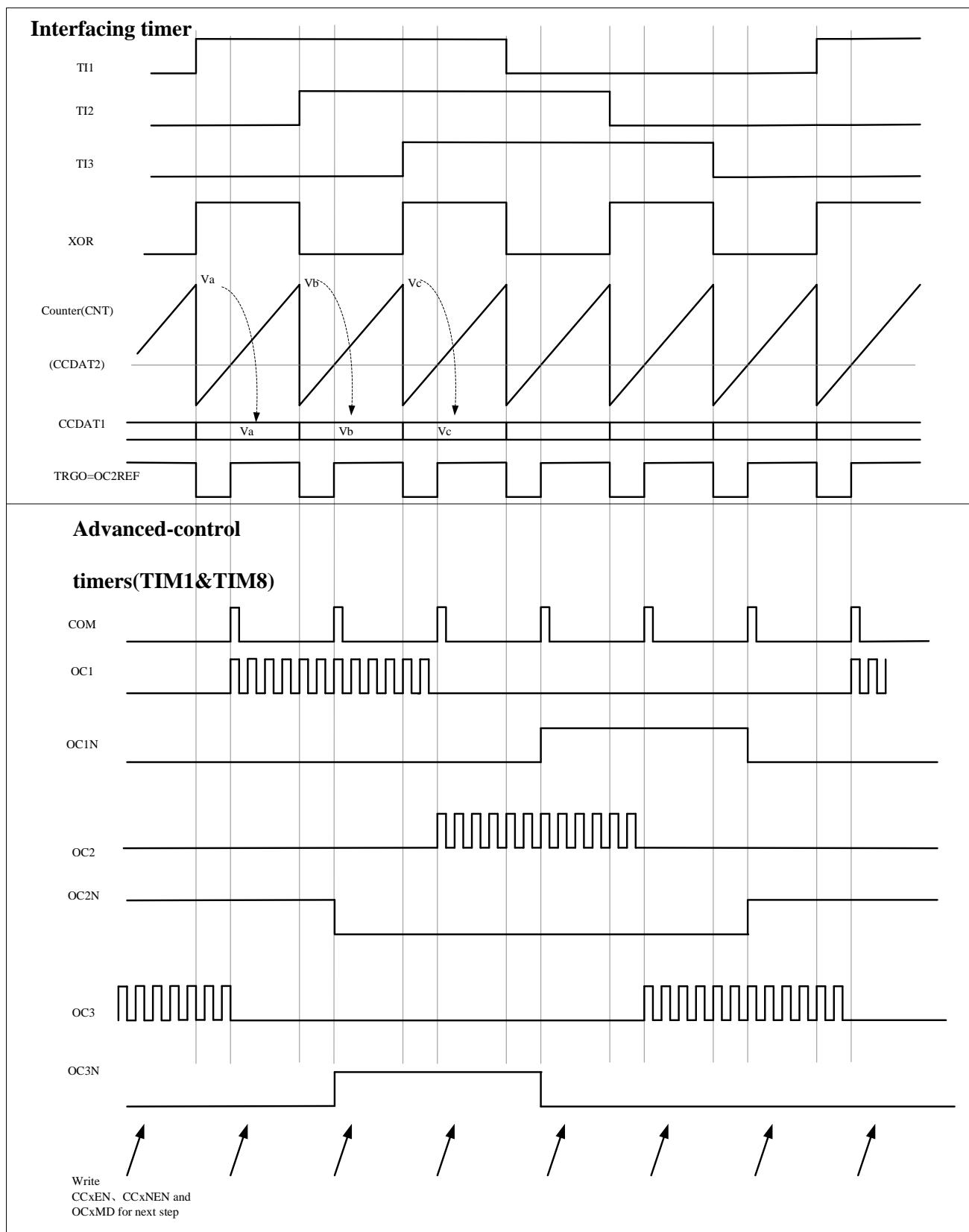
将霍尔传感器连接到定时器的三个输入引脚（CC1、CC2 和 CC3），然后选择异或功能将 TIMx\_CH1、TIMx\_CH2 和 TIMx\_CH3 的输入通过异或门作为 TI1 的输出到通道 1 进行捕捉信号。

定时器需要配置为从模式下的复位模式（TIMx\_SMCTRL.SMSEL='100'）；触发选择 TI1 的边沿触发 TI1F\_ED (TIMx\_SMCTRL.TSEL='100')，霍尔 3 输入的任何变化都会触发计数器重新计数，因此用作时间参考；捕获/比较通道 1 配置为捕获模式下的 TRC 信号 (TIMx\_CCMOD1.CC1SEL='11')，用于计算两个输入时间间隔，从而反映电机速度。

选择定时器通道 2 向高级定时器输出脉冲，触发高级定时器的 COM 事件，更新输出 PWM 的控制位。高级定时器的触发选择需要选择对应的内部触发信号 (TIMx\_SMCTRL.TSEL="ITRx")，捕获/比较预加载控制位需要配置为支持预加载 (TIMx\_CTRL2.CCPCTL=1) 并支持上升沿 TRGI 边沿触发更新 (TIMx\_CTRL2.CCUSEL=1)。

此示例如下图所示。

图 9-41 霍尔传感器接口的实例



## 9.4 TIMx 寄存器描述 (x=1,8)

关于在寄存器描述里面所用到的缩写，详见第 1.1 节。

可以用半字（16位）或字（32位）的方式操作这些外设寄存器。

#### 9.4.1 寄存器总览

表 9-2 TIM1 和 TIM8 寄存器总览

Offset	Register	Reset value	0x000	TIMx_CTRL1	31
			0x004	TIMx_CTRL2	30
			0x008	TIMx_CTRL	29
0x10	TIMx_STS	Reset value	0x0C	TIMx_DIN	28
0x14	TIMx_EVGEN	Reset value		Reserved	27
0x18				Reserved	26
0x1C					25
0x20					24
0x24					23
0x28					22
0x2C					21
0x30				CMD0[1:0]	20
0x34				0	19
0x38				TRIG7	19
0x3C				Reserved	19
0x40				0	18
0x44				OI6	18
0x48				0	ASYMMETRIC
0x4C				0	17
0x50				OI4N	17
0x54				0	PBKPN
0x58				OI5	16
0x5C				0	LBKPN
0x60				TRIG4	15
0x64				0	CLRSEL
0x68					15
0x70					14
0x74					13
0x78					12
0x7C					11
0x80					10
0x84					9
0x88					8
0x90					7
0x94					6
0x98					5
0xA0					4
0xA4					3
0xA8					2
0xB0					1
0xB4					0

0x18	TIMx_CC MOD1 Output compare mode	Reserved																0	OC2CEN
	Reset value																		
0x18	TIMx_CC MOD1 Input capture mode	Reserved																0	OC2FEN
	Reset value																		
0x1C	TIMx_CC MOD2 Output comparison mode	Reserved																0	CC2SEL[1:0]
	Reset value																		
0x1C	TIMx_CC MOD2 Input capture mode	Reserved																0	IC2PSC[1:0]
	Reset value																		
0x20	TIMx_CC EN	Reserved																0	IC2PSC[1:0]
	Reset value																		
0x24	TIMx_CNT	Reserved																0	CC2SEL[1:0]
	Reset value																		
0x28	TIMx_PSC	Reserved																0	IC1PSC[1:0]
	Reset value																		
0x2C	TIMx_AR	Reserved																0	CC1SEL[1:0]
	Reset value																		



#### 9.4.2 控制寄存器 1 (TIMx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000 0000

31	Reserved										22	21	20	19	18	17	16		
15	CLRSEL	Reserved	14	C1SEL	IOMBKPN	12	CLKD[1:0]	10	ARPEN	9	CAMSEL[1:0]	8	DIR	7	ONEPM	6	UPRS	5	rw
																		rw	

位域	名称	描述
31:22	Reserved	保留，必须保持复位值。
21:20	CMODE	<p>在中央对齐非对称模式，通道 4/7/8/9 触发模式下，只有当 TIMx_CTRL2.MMSEL3 = 1 时，TRGO 输出才会有效。</p> <p>00: 向上计数到CCDAT4/7/8/9，触发有效</p> <p>01: 向下计数，通道4计数到CCDDAT4，通道7/8/9计数到CCDAT7/8/9，触发有效</p> <p>1x: 通道4向上计数到CCDAT4或向下计数到CCDDAT4，通道7/8/9向上计数或向下计数到CCDAT7/8/9，触发有效</p> <p>在中央对齐对称模式，通道 4/7/8/9 触发模式下，只有当 TIMx_CTRL2.MMSEL3 = 1 时，TRGO 输出才会有效。</p> <p>00: 向上计数到CCDAT4/7/8/9，触发有效</p> <p>01: 向下计数到CCDAT4/7/8/9，触发有效</p> <p>1x: 向上计数或向下计数到CCDAT4/7/8/9，触发有效</p> <p><i>注意：此功能仅适用于 TIM1</i></p>
19	Reserved	保留，必须保持复位值。
18	ASYMMETRIC	<p>中央对齐非对称模式使能</p> <p>0: 禁用</p> <p>1: 使能（当TIMx_CTRL1.CAMSEL[1:0]非零时有效，每个通道向上计数时与CCDATx 比较，向下计数时与CCDDATx比较）</p> <p><i>注意：此功能仅适用于 TIM1</i></p>
17	PBKPN	<p>PVD作为BRK启用（PVD as brk Enable）</p> <p>0: 禁止</p> <p>1: 使能</p>
16	LBKPN	<p>锁存作为BRK使能（LockUp as brk Enable）（Core Hardfault）</p> <p>0: 禁止</p> <p>1: 使能</p>

位域	名称	描述
15	CLRSEL	OCxRef选择 (OCxRef selection) 0: 选择外部Ocxclr (ETR) 信号 1: 选择内部OCxclr (来自COMP) 信号
14:12	Reserved	保留, 必须保持复位值。
11	C1SEL	通道1选择 (Channel 1 selection) 0: 选择外部CH1 (来自IOM) 信号 1: 选择内部CH1 (来自COMP) 信号
10	IOMBKPEN	IOM作为BRK使能 (IOM as brk Enable) 0: 使能, 刹车信号来自外部IOM 1: 禁止, 刹车信号来自内部比较器
9:8	CLKD[1:0]	时钟分频因子 (Clock division) CLKD[1:0] 表示 CK_INT (定时器时钟) 和 DTS (用于死区时间发生器和数字滤波器 (ETR、TIx) 的时钟) 之间的分频比。 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	ARPEN	自动重装载预装载允许位 (Auto-reload preload enable) 0: TIMx_AR 寄存器的影子寄存器禁用 1: TIMx_AR 寄存器的影子寄存器使能
6:5	CAMSEL[1:0]	选择中央对齐模式 (Center-aligned mode selection) 00: 边缘对齐模式。 TIMx_CTRL1.DIR 指定向上计数或向下计数。 01: 中央对齐模式1。 计数器在中央对齐模式下计数, 向下计数时输出比较中断标志位设置为 1。 10: 中央对齐模式2。 计数器在中央对齐模式下计数, 向上计数时输出比较中断标志位设置为1。 11: 中央对齐模式3。 计数器在中央对齐模式下计数, 向上计数或向下计数时输出比较中断标志位设置为 1。  注意: 当计数器仍然启用时 (TIMx_CTRL1.CNTEN = 1), 不允许从边缘对齐模式切换

位域	名称	描述
		到中央对齐模式。
4	DIR	<p>方向 (Direction)</p> <p>0: 计数器向上计数;</p> <p>1: 计数器向下计数。</p> <p>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。</p>
3	ONEPM	<p>单脉冲模式 (One pulse mode)</p> <p>0: 禁用单脉冲模式, 发生更新事件时不影响计数器计数。</p> <p>1: 使能单脉冲模式, 下次更新事件发生时计数器停止计数</p>
2	UPRS	<p>更新请求源 (Update request source)</p> <p>该位用于通过软件选择 UEV 事件源。</p> <p>0: 如果更新中断或 DMA 请求使能, 以下任何事件都会产生更新中断或 DMA 请求:</p> <ul style="list-style-type: none"> <li>- 计数器上溢/下溢</li> <li>- TIMx_EVTGEN.UDGN 位被设置</li> <li>- 从模式控制器的更新生成</li> </ul> <p>1: 如果更新中断或 DMA 请求使能, 只有计数器上溢/下溢会产生更新中断或 DMA 请求。</p>
1	UPDIS	<p>更新禁用 (Update disable)</p> <p>该位用于启用/禁用软件生成的更新事件 (UEV) 事件。</p> <p>0: 启用。 如果满足以下条件之一, 将生成 UEV:</p> <ul style="list-style-type: none"> <li>- 计数器上溢/下溢</li> <li>- TIMx_EVTGEN.UDGN 位被设置</li> <li>- 从模式控制器的更新生成</li> </ul> <p>影子寄存器将使用预加载值进行更新。</p> <p>1: UEV 禁用。 不生成更新事件, 影子寄存器 (AR、PSC 和 CCDAFx) 保持它们的</p>

位域	名称	描述
		值。如果 TIMx_EVTGEN.UDGN 位置位或从模式控制器发出硬件复位，则重新初始化计数器和预分频器。
0	CNTEN	<p>使能计数器 (Counter enable)</p> <p>0: 禁止计数器;</p> <p>1: 使能计数器。</p> <p>注: 在软件设置了CNTEN位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置CNTEN位。</p>

### 9.4.3 控制寄存器 2 (TIMx\_CTRL2)

偏移地址: 0x04

复位值: 0x0000 0000

31								25	24	23	22	21	20	19	18	17	16	
								Reserved		MMSEL3	TRIG9	Reserved	TRIG8	Reserved	TRIG7	OI6	OI4N	OI5
15	14	13	12	11	10	9		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
TRIG4	OI4	OI3N	OI3	OI2N	OI2	OI1N	OII	TI1SEL		MMSEL[2:0]		CCDSEL	CCUSEL	Reserved	CCPCTL			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

位域	名称	描述
31:25	Reserved	保留, 必须保持复位值。
24	MMSEL3	<p>主模式选择的Bit3位.</p> <p>注意: 仅用于 TIM1</p>
23	TRIG9	<p>OC9REF 触发 ADC 使能</p> <p>0: 触发禁用</p> <p>1: 触发使能</p> <p>注意: 此功能仅适用于 TIM1</p>
22	Reserved	保留, 必须保持复位值。

位域	名称	描述
21	TRIG8	OC8REF 触发 ADC 使能 0: 触发禁用 1: 触发使能 <i>注意: 此功能仅适用于 TIM1</i>
20	Reserved	保留, 必须保持复位值。
19	TRIG7	OC7REF 触发 ADC 使能 0: 触发禁用 1: 触发使能 <i>注意: 此功能仅适用于 TIM1</i>
18	OI6	输出空闲状态6 (OC6输出)。参见OI1位。
17	OI4N	输出空闲状态4 (OC4N输出), 参见OI1N位。 <i>注意: 此功能仅适用于 TIM1</i>
16	OI5	输出空闲状态5 (OC5输出)。参见OI1位。
15	TRIG4	OC4REF 触发 ADC 使能 0: 触发禁用 1: 触发使能 <i>注意: 此功能仅适用于 TIM1</i>
14	OI4	输出空闲状态4 (OC4输出)。参见OI1位。
13	OI3N	输出空闲状态3 (OC3N输出)。参见OI1N位。
12	OI3	输出空闲状态3 (OC3输出)。参见OI1位。
11	OI2N	输出空闲状态2 (OC2N输出)。参见OI1N位。
10	OI2	输出空闲状态2 (OC2输出)。参见OI1位。
9	OI1N	输出空闲状态1 (OC1N输出) (Output Idle state 1) 0: 当MOEN=0时, 死区后OC1N=0; 1: 当MOEN=0时, 死区后OC1N=1。

位域	名称	描述
8	OI1	<p>输出空闲状态1（OC1输出）（Output Idle state 1）</p> <p>0: 当MOEN=0时，如果实现了OC1N，则死区后OC1=0；</p> <p>1: 当MOEN=0时，如果实现了OC1N，则死区后OC1=1。</p>
7	TI1SEL	<p>TI1选择（TI1 selection）</p> <p>0: TIMx_CH1引脚连到TI1输入；</p> <p>1: TIMx_CH1、TIMx_CH2和TIMx_CH3引脚经异或后连到TI1输入。</p>
6:4	MMSEL[2:0]	<p>主模式选择</p> <p>这 4 位 (TIMx_CTRL2.MMSEL3 和 TIMx_CTRL2.MMSEL [2:0]) 用于选择在主模式下发送到从定时器的同步信息 (TRGO)。可能的组合如下：</p> <p>000: 复位 - 当 TIMx_EVTGEN.UDGN 置位或从模式控制器产生复位时，将出现 TRGO 脉冲。在后一种情况下，TRGO 上的信号与实际复位相比有所延迟。</p> <p>001: 使能 - TIMx_CTRL1.CNTEN 位用作触发输出 (TRGO)。有时需要同时启动多个定时器或者在一段时间内开启从定时器。</p> <p>当 TIMx_CTRL1.CNTEN 位置位或门控模式下的触发输入为高电平时，计数器使能信号置位。</p> <p>当计数器使能信号由触发输入控制时，TRGO 上有一个延迟，除非选择了主/从模式（参见 TIMx_SMCTRL.MSMD 位的说明）。</p> <p>010: 更新 - 选择更新事件作为触发输出 (TRGO)。例如，主定时器时钟可用作从定时器预分频器。</p> <p>011: 比较脉冲 - 当 TIMx_STS.CC1ITF 被设置时（即使它已经是高电平），即捕获或比较成功时，触发输出发送一个正脉冲 (TRGO)。</p> <p>100: 比较 - OC1REF 信号用作触发输出 (TRGO)。</p> <p>101: 比较 - OC2REF 信号用作触发输出 (TRGO)。</p> <p>110: 比较 - OC3REF 信号用作触发输出 (TRGO)。</p> <p>111: 比较 - OC4REF 信号用作触发输出 (TRGO)。</p> <p>1xxx: 比较-如果计数器为中央对齐模式：OC4REF/OC7REF/OC8REF/OC9REF的相应边沿信号作为触发输出 (TRGO)，向上/向下计数可配置，具体参考TIMx_CTRL1.CMODE 中。</p> <p>如果计数器为边沿对齐模式：OC4REF 信号用作触发输出 (TRGO)。</p>
3	CCDSEL	<p>捕获/比较的DMA选择（Capture/compare DMA selection）</p> <p>0: 当发生CCx事件时，送出CCx的DMA请求；</p>

位域	名称	描述
		1: 当发生更新事件时，送出CCx的DMA请求。
2	CCUSEL	<p>捕获/比较控制更新选择 (Capture/compare control update selection)</p> <p>0: 如果捕获/比较控制位是预装载的 (CCPCTL = 1)，只能通过设置CCUDGN位更新它们；</p> <p>1: 如果捕获/比较控制位是预装载的 (CCPCTL = 1)，可以通过设置CCUDGN位或TRGI上的一个上升沿更新它们。</p> <p>注：该位只对具有互补输出的通道起作用。</p>
1	Reserved	保留，必须保持复位值。
0	CCPCTL	<p>捕获/比较预装载控制位 (Capture/compare preloaded control)</p> <p>0: CCxEN, CCxNEN和OCxMD位不是预装载的；</p> <p>1: CCxEN, CCxNEN和OCxMD位是预装载的；设置该位后，它们只在设置了CCUDGN位后被更新。</p> <p>注：该位只对具有互补输出的通道起作用。</p>

#### 9.4.4 从模式控制寄存器 (TIMx\_SMCTRL)

偏移地址: 0x08

复位值: 0x0000

15	14	13	12	11	8	7	6	4	3	2	0
EXTP	EXCEN	EXTPS[1:0]		EXTF[3:0]	MSMD	TSEL[2:0]	Reserved	SMSEL[2:0]			

rw            rw            rw            rw            rw            rw            rw            rw            rw

位域	名称	描述
15	EXTP	<p>外部触发极性 (External trigger polarity)</p> <p>该位选择是用ETR还是ETR的反相来作为触发操作</p> <p>0: ETR高电平或上升沿有效；</p> <p>1: ETR低电平或下降沿有效。</p>

位域	名称	描述
14	EXCEN	<p>外部时钟使能位（External clock enable）该位启用外部时钟模式2。启用后，计数器由ETRF信号上的任意有效边沿驱动。</p> <p>0：禁止外部时钟模式2； 1：使能外部时钟模式2。</p> <p><i>注 1：当同时使能外部时钟模式 1 和外部时钟模式 2 时，外部时钟的输入为 ETRF。</i></p> <p><i>注2：以下从机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，TRGI 无法连接到 ETRF (TIMx_SMCTRL.TSEL ≠ '111')。</i></p> <p><i>注 3：设置 TIMx_SMCTRL.EXCEN 位与选择外部时钟模式 1 并将 TRGI 连接到 ETRF (TIMx_SMCTRL.SMSEL = 111 和 TIMx_SMCTRL.TSEL = 111) 的效果相同</i></p>
13:12	EXTPS[1:0]	<p>外部触发预分频（External trigger prescaler）</p> <p>外部触发信号 ETRP 的频率必须最多为 TIMxCLK 频率的 1/4。当输入更快的外部时钟时，可以使用预分频器来降低 ETRP 的频率。</p> <p>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。</p>
11:8	EXTF[3:0]	<p>外部触发滤波（External trigger filter）</p> <p>这些位用于定义 ETRP 信号的采样频率和 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，在记录连续 N 个事件后生成验证输出。</p> <p>0000：无滤波器，以fDTS采样 1000：采样频率fSAMPLING=fDTS/8, N=6 0001：采样频率fSAMPLING=fCK_INT, N=2 1001：采样频率fSAMPLING=fDTS/8, N=8 0010：采样频率fSAMPLING=fCK_INT, N=4 1010：采样频率fSAMPLING=fDTS/16, N=5 0011：采样频率fSAMPLING=fCK_INT, N=8 1011：采样频率fSAMPLING=fDTS/16, N=6 0100：采样频率fSAMPLING=fDTS/2, N=6 1100：采样频率fSAMPLING=fDTS/16, N=8 0101：采样频率fSAMPLING=fDTS/2, N=8 1101：采样频率fSAMPLING=fDTS/32, N=5 0110：采样频率fSAMPLING=fDTS/4, N=6 1110：采样频率fSAMPLING=fDTS/32, N=6</p>

位域	名称	描述
		0111: 采样频率fSAMPLING=fDTS/4, N=8      1111: 采样频率fSAMPLING=fDTS/32, N=8
7	MSMD	<p>主/从模式 (Master/slave mode)</p> <p>0: 无作用;</p> <p>1: 触发输入 (TRGI) 上的事件被延迟了, 以允许在当前定时器 (通过TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。</p>
6:4	TSEL[2:0]	<p>触发选择 (Trigger selection)</p> <p>这3位选择用于同步计数器的触发输入。</p> <p>000: 内部触发0 (ITR0)  100: TI1的边沿检测器 (TI1F_ED)</p> <p>001: 内部触发1 (ITR1)  101: 滤波后的定时器输入1 (TI1FP1)</p> <p>010: 内部触发2 (ITR2)  110: 滤波后的定时器输入2 (TI2FP2)</p> <p>011: 内部触发3 (ITR3)  111: 外部触发输入 (ETRF)</p> <p>更多有关ITRx的细节, 参见表9-3。</p> <p>注: 这些位只能在未用到 (如SMSEL=000) 时被改变, 以避免在改变时产生错误的边沿检测。</p>
3	Reserved	保留, 必须保持复位值。

位域	名称	描述
2:0	SMSEL[2:0]	<p>从模式选择 (Slave mode selection)</p> <p>当选择了外部信号，触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关 (见输入控制寄存器和控制寄存器的说明)</p> <p>000: 关闭从模式 – 如果CNTEN=1，则预分频器直接由内部时钟驱动。</p> <p>001: 编码器模式1 – 根据TI2FP2的电平，计数器在TI1FP1的边沿向上/下计数。</p> <p>010: 编码器模式2 – 根据TI1FP1的电平，计数器在TI2FP2的边沿向上/下计数。</p> <p>011: 编码器模式3 – 根据另一个信号的输入电平，计数器在TI1FP1和TI2FP2的边沿向上/下计数。</p> <p>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，并更新影子寄存器。</p> <p>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。</p> <p>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。</p> <p>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。</p> <p>注: 如果TIIF_ED 被选为触发输入 (TSEL=100) 时，不要使用门控模式。这是因为，TIIF_ED 在每次TIIF 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。</p>

表 9-3 TIMx 内部触发连接

Slave timer	ITR0 (TSEL = 000)	ITR1 (TSEL = 001)	ITR2 (TSEL = 010)	ITR3 (TSEL = 011)
<b>TIM1</b>	TIM5	TIM2	TIM3	TIM4
<b>TIM8</b>	TIM1	TIM2	TIM4	TIM5

#### 9.4.5 DMA/中断使能寄存器 (TIMx\_DINTEN)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TDEN	COMDEN	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	BIEN	TIEN	COMIEN	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN

位域	名称	描述
15	Reserved	保留，必须保持复位值。

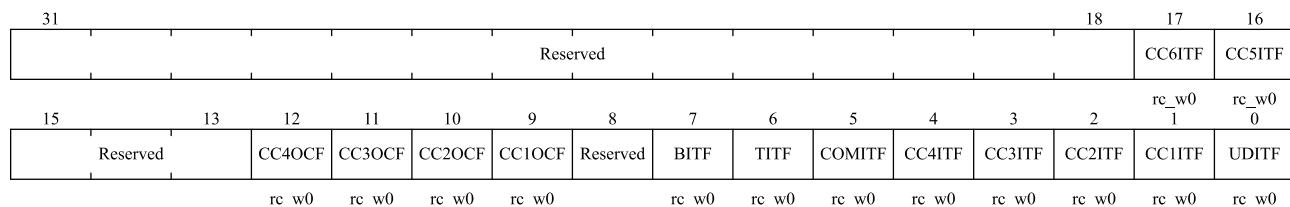
位域	名称	描述
14	TDEN	允许触发DMA请求 (Trigger DMA request enable) 0: 禁止触发DMA请求; 1: 允许触发DMA请求。
13	COMDEN	允许COM的DMA请求 (COM DMA request enable) 0: 禁止COM的DMA请求; 1: 允许COM的DMA请求。
12	CC4DEN	允许捕获/比较4的DMA请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较4的DMA请求; 1: 允许捕获/比较4的DMA请求。
11	CC3DEN	允许捕获/比较3的DMA请求 (Capture/Compare 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	CC2DEN	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	CC1DEN	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	UDEN	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	BIEN	允许刹车中断 (Break interrupt enable) 0: 禁止刹车中断; 1: 允许刹车中断。
6	TIEN	触发中断使能 (Trigger interrupt enable) 0: 禁止触发中断; 1: 使能触发中断。

位域	名称	描述
5	COMIEN	允许COM中断 (COM interrupt enable) 0: 禁止COM中断; 1: 允许COM中断。
4	CC4IEN	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	CC3IEN	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	CC2IEN	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	CC1IEN	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	UIEN	允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

#### 9.4.6 状态寄存器 (TIMx\_STS)

偏移地址: 0x10

复位值: 0x0000 0000



位域	名称	描述
31:18	Reserved	保留，必须保持复位值。
17	CC6ITF	捕获/比较6中断标记 (Capture/Compare 6 interrupt flag) 参考CC1ITF描述。
16	CC5ITF	捕获/比较5中断标记 (Capture/Compare 5 interrupt flag) 参考CC1ITF描述。
15:13	Reserved	保留，必须保持复位值。
12	CC4OCF	捕获/比较4重复捕获标记 (Capture/Compare 4 overcapture flag) 参见CC1OCF描述。
11	CC3OCF	捕获/比较3重复捕获标记 (Capture/Compare 3 overcapture flag) 参见CC1OCF描述。
10	CC2OCF	捕获/比较2重复捕获标记 (Capture/Compare 2 overcapture flag) 参见CC1OCF描述。
9	CC1OCF	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_CCDAT1寄存器时，CC1ITF的状态已经为‘1’。
8	Reserved	保留，必须保持复位值。
7	BITF	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。
6	TITF	触发器中断标记 (Trigger interrupt flag) 当发生触发事件（当从模式控制器处于除门控模式外的其它模式时，在TRGI输入端检测到有效边沿，或门控模式下的任一边沿）时由硬件对该位置‘1’。它由软件清‘0’。 0: 无触发器事件产生； 1: 触发中断等待响应。

位域	名称	描述
5	COMITF	<p>COM中断标记 (COM interrupt flag)</p> <p>一旦产生COM事件（当捕获/比较控制位：CCxEN、CCxNEN、OCxMD已被更新）该位由硬件置'1'。它由软件清'0'。</p> <p>0: 无COM事件产生； 1: COM中断等待响应。</p>
4	CC4ITF	<p>捕获/比较4中断标记 (Capture/Compare 4 interrupt flag)</p> <p>参考CC1ITF描述。</p>
3	CC3ITF	<p>捕获/比较3中断标记 (Capture/Compare 3 interrupt flag)</p> <p>参考CC1ITF描述。</p>
2	CC2ITF	<p>捕获/比较2中断标记 (Capture/Compare 2 interrupt flag)</p> <p>参考CC1ITF描述。</p>
1	CC1ITF	<p>捕获/比较1中断标记 (Capture/Compare 1 interrupt flag)</p> <p><b>如果通道CC1配置为输出模式：</b></p> <p>除中央对齐模式外，当计数器值与比较值相同时，该位由硬件设置（参见 TIMx_CTRL1.CAMSEL 位描述）。该位由软件清零。</p> <p>0: 未发生匹配。 1: TIMx_CNT 的值与 TIMx_CCDAT1 的值相同。</p> <p>当 TIMx_CCDAT1 的值大于 TIMx_AR 的值时，如果计数器溢出（在向上计数和向上/向下计数模式下）和向下计数模式下溢，则 TIMx_STS.CC1ITF 位将变为高电平。</p> <p><b>如果通道CC1配置为输入模式：</b></p> <p>当捕捉事件发生时，该位由硬件设置。该位由软件或读取 TIMx_CCDAT1 清零。</p> <p>0: 未发生输入捕捉。 1: 发生输入捕捉。计数器值已在 TIMx_CCDAT1 中捕获。在 IC1 上检测到与所选极性相同的边沿。</p>
0	UDITF	<p>更新中断标志 (Update interrupt flag)</p> <p>当在以下条件下发生更新事件时，该位由硬件设置：</p> <ul style="list-style-type: none"> <li>- 当 TIMx_CTRL1.UPDIS = 0 时，并且重复计数器值上溢或下溢（当重复计数器等于 0 时生成更新事件UEV）。</li> <li>- 当 TIMx_CTRL1.UPRS = 0 时，TIMx_CTRL1.UPDIS = 0，并通过软件设置 TIMx_EVTGEN.UDGN 位以重新初始化 CNT。</li> </ul>

位域	名称	描述
		<ul style="list-style-type: none"> <li>- 当 TIMx_CTRL1.UPRS = 0 时, TIMx_CTRL1.UPDIS = 0, 并且计数器 CNT 由触发事件重新初始化。 (参见 TIMx_SMCTRL 寄存器说明)</li> </ul> <p>该位由软件清零。</p> <p>0: 未发生更新事件 1: 发生更新中断</p>

#### 9.4.7 事件产生寄存器 (TIMx\_EVTGEN)

偏移地址:0x14

复位值:0x0000

15	Reserved	8	7	6	5	4	3	2	1	0
			BGN	TGN	CCUDGN	CC4GN	CC3GN	CC2GN	CC1GN	UDGN

位域	名称	描述
15:8	Reserved	保留, 必须保持复位值。
7	BGN	<p>产生刹车事件 (Break generation)</p> <p>当由软件设置时, 该位可以产生一个刹车事件。而此时TIMx_BKDT.MOEN = 0, TIMx_STS.BITF = 1, 如果相应的中断和DMA被使能, 就会产生相应的中断和DMA。该位由硬件自动清零。</p> <p>0: 无动作 1: 产生刹车事件</p>
6	TGN	<p>产生触发事件 (Trigger generation)</p> <p>当由软件置位时, 该位可以产生一个触发事件。而此时TIMx_STS.TITF = 1, 如果相应的中断和DMA被使能, 就会产生相应的中断和DMA。该位由硬件自动清零。</p> <p>0: 无动作 1: 产生触发事件</p>
5	CCUDGN	<p>捕获/比较事件, 产生控制更新 (Capture/Compare control update generation)</p> <p>该位由软件设置。如果此时 TIMx_CTRL2.CCPCTL = 1, 则允许更新 CCxEN、CCxNEN 和 OCxMD 位。该位由硬件自动清零。</p> <p>0: 无动作 1: 产生一个COM事件</p>

位域	名称	描述
		注意：该位仅对具有互补输出的通道有效。
4	CC4GN	产生捕获/比较4事件 (Capture/Compare 4 generation) 参考CC1GN描述。
3	CC3GN	产生捕获/比较3事件 (Capture/Compare 3 generation) 参考CC1GN描述。
2	CC2GN	产生捕获/比较2事件 (Capture/Compare 2 generation) 参考CC1GN描述。
1	CC1GN	产生捕获/比较1事件 (Capture/Compare 1 generation) 当由软件设置时，该位可以产生一个捕获/比较事件。该位由硬件自动清零。 <b>CC1对应通道为输出模式时：</b> TIMx_STS.CC1ITF 标志将被拉高，如果相应的中断和 DMA 被使能，就会产生相应的中断和 DMA。 <b>CC1对应通道为输入模式时：</b> TIMx_CCDAT1 将捕获当前计数器值，并将 TIMx_STS.CC1ITF 标志拉高，如果相应的中断和 DMA 被使能，则会产生相应的中断和 DMA。如果 TIMx_STS.CC1ITF 已经拉高，则拉高 TIMx_STS.CC1OCF。 0：无动作 1：生成 CC1 捕获/比较事件
0	UDGN	产生更新事件 (Update generation) 该位由软件置'1'，由硬件自动清'0'。 当由软件设置时，该位可以生成更新事件。而此时计数器会重新初始化，预分频计数器会被清零，计数器在中央对齐或向上计数模式下会被清零，但在向下计数模式下取 TIMx_AR 寄存器的值。该位由硬件自动清零。 0：无动作 1：生成更新事件

#### 9.4.8 捕获/比较模式寄存器 1 (TIMx\_CCMOD1)

偏移地址：0x18

复位值：0x0000

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

输出比较模式：

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC2CEN	OC2MD[2:0]	OC2PEN	OC2FEN	CC2SEL[1:0]	OC1CEN	OC1MD[2:0]	OC1PEN	OC1FEN	CC1SEL[1:0]				

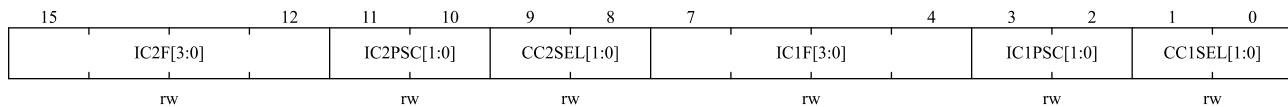
rw                    rw

位域	名称	描述
15	OC2CEN	输出比较2清0使能 （Output Compare 2 clear enable）
14:12	OC2MD[2:0]	输出比较2模式 （Output Compare 2 mode）
11	OC2PEN	输出比较2预装载使能 （Output Compare 2 preload enable）
10	OC2FEN	输出比较2快速使能 （Output Compare 2 fast enable）
9:8	CC2SEL[1:0]	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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。 <i>注：CC2SEL 仅在通道关闭时 (TIMx_CCEN 寄存器的CC2EN=0) 才是可写的。</i>
7	OC1CEN	输出比较1清'0'使能 （Output Compare 1 clear enable） 0: OC1REF 不受ETRF输入的影响； 1: 一旦检测到ETRF输入高电平，清除OC1REF=0。

位域	名称	描述
6:4	OC1MD[2:0]	<p>输出比较1模式 (Output Compare 1 mode)</p> <p>这些位用于管理输出参考信号 OC1REF，它决定了 OC1 和 OC1N 的值，在高电平有效，而 OC1 和 OC1N 的有效电平取决于 TIMx_CCEN.CC1P 和 TIMx_CCEN.CC1NP 位。</p> <p>000: 冻结。 TIMx_CCDAT1 寄存器和计数器 TIMx_CNT 之间的比较对 OC1REF 信号没有影响。</p> <p>001: 将通道 1 设置为匹配时的有效电平。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被强制为高电平。</p> <p>010: 将通道 1 设置为匹配时的无效电平。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被强制为低电平。</p> <p>011: 翻转。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被翻转。</p> <p>100: 强制无效电平。 OC1REF 信号被强制为低电平。</p> <p>101: 强制有效电平。 OC1REF 信号被强制为高电平。</p> <p>110: PWM 模式 1 - 在向上计数模式下，如果 TIMx_CNT &lt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为高电平，否则为低电平。在向下计数模式下，如果 TIMx_CNT &gt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为低电平，否则为高电平。</p> <p>111: PWM 模式 2 - 在向上计数模式下，如果 TIMx_CNT &lt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为低电平，否则为高电平。在向下计数模式下，如果 TIMx_CNT &gt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为高电平，否则为低电平。</p> <p><i>注 1: 在 PWM 模式 1 或 PWM 模式 2 中，OC1REF 电平仅在比较结果改变或输出比较模式从冻结模式切换到 PWM 模式时才会改变。</i></p>
3	OC1PEN	<p>输出比较 1 预加载使能 (Output Compare 1 preload enable)</p> <p>0: 禁用 TIMx_CCDAT1 寄存器的预加载功能。 支持随时对 TIMx_CCDAT1 寄存器进行写操作，写入的值立即生效。</p> <p>1: 使能 TIMx_CCDAT1 寄存器的预加载功能。 仅对预加载寄存器进行读写操作。 当更新事件发生时，TIMx_CCDAT1 的值被加载到影子寄存器中。</p> <p><i>注 1: 只有当 TIMx_CTRL1.ONEPM = 1 (在单脉冲模式下) 时，才能使用 PWM 模式而不验证预加载寄存器，否则无法预测其他行为。</i></p>
2	OC1FEN	<p>输出比较1 快速使能 (Output Compare 1 fast enable)</p> <p>该位用于加快CC输出对触发输入事件的响应。</p> <p>0: 根据计数器与CCDAT1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。</p> <p>1: 输入到触发器的有效沿的作用就像发生了一次比较匹配。因此，OC1被设置为比较</p>

位域	名称	描述
		<p>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。</p> <p>OCxFEN只在通道被配置成PWM1或PWM2模式时起作用。</p>
1:0	CC1SEL[1:0]	<p>捕获/比较1 选择。 (Capture/Compare 1 selection)</p> <p>这2位定义通道的方向（输入/输出），及输入脚的选择：</p> <ul style="list-style-type: none"> <li>00: CC1通道被配置为输出；</li> <li>01: CC1通道被配置为输入， IC1映射在TI1上；</li> <li>10: CC1通道被配置为输入， IC1映射在TI2上；</li> <li>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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。</li> </ul> <p>注：CC1SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC1EN=0) 才是可写的。</p>

输入捕获模式：



位域	名称	描述
15:12	IC2F[3:0]	输入捕获2滤波器 (Input capture 2 filter)
11:10	IC2PSC[1:0]	输入/捕获2预分频器 (Input capture 2 prescaler)
9:8	CC2SEL[1:0]	<p>捕获/比较2选择 (Capture/Compare 2 selection)</p> <p>这2位定义通道的方向（输入/输出），及输入脚的选择：</p> <ul style="list-style-type: none"> <li>00: CC2通道被配置为输出；</li> <li>01: CC2通道被配置为输入， IC2映射在TI2上；</li> <li>10: CC2通道被配置为输入， IC2映射在TI1上；</li> <li>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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。</li> </ul> <p>注：CC2SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC2EN=0) 才是可写的。</p>

位域	名称	描述																
7:4	IC1F[3:0]	<p>输入捕获1滤波器 (Input capture 1 filter)</p> <p>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组成，它记录到N个事件后会产生一个输出的跳变：</p> <table> <tbody> <tr> <td>0000: 无滤波器，以<math>f_{DTS}</math>采样</td> <td>1000: 采样频率<math>f_{SAMPLING}=f_{DTS}/8</math>, N=6</td> </tr> <tr> <td>0001: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=2</td> <td>1001: 采样频率<math>f_{SAMPLING}=f_{DTS}/8</math>, N=8</td> </tr> <tr> <td>0010: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=4</td> <td>1010: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=5</td> </tr> <tr> <td>0011: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=8</td> <td>1011: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=6</td> </tr> <tr> <td>0100: 采样频率<math>f_{SAMPLING}=f_{DTS}/2</math>, N=6</td> <td>1100: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=8</td> </tr> <tr> <td>0101: 采样频率<math>f_{SAMPLING}=f_{DTS}/2</math>, N=8</td> <td>1101: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=5</td> </tr> <tr> <td>0110: 采样频率<math>f_{SAMPLING}=f_{DTS}/4</math>, N=6</td> <td>1110: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=6</td> </tr> <tr> <td>0111: 采样频率<math>f_{SAMPLING}=f_{DTS}/4</math>, N=8</td> <td>1111: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=8</td> </tr> </tbody> </table>	0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=6	0001: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=8	0010: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=5	0011: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=6	0100: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=8	0101: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=5	0110: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=6	0111: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=8
0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=6																	
0001: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=8																	
0010: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=5																	
0011: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=6																	
0100: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=8																	
0101: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=5																	
0110: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=6																	
0111: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=8																	
3:2	IC1PSC[1:0]	<p>输入/捕获1预分频器 (Input capture 1 prescaler)</p> <p>这2位定义了CC1输入 (IC1) 的预分频系数。</p> <p>一旦<math>TIMx\_CCEN.CC1EN=0</math>，则预分频器复位。</p> <p>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；</p> <p>01: 每2个事件触发一次捕获；</p> <p>10: 每4个事件触发一次捕获；</p> <p>11: 每8个事件触发一次捕获。</p>																
1:0	CC1SEL[1:0]	<p>捕获/比较1选择 (Capture/Compare 1 Selection)</p> <p>这2位定义通道的方向 (输入/输出)，及输入脚的选择：</p> <p>00: CC1通道被配置为输出；</p> <p>01: CC1通道被配置为输入，IC1映射在TI1上；</p> <p>10: CC1通道被配置为输入，IC1映射在TI2上；</p> <p>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输入被选中时 (由<math>TIMx\_SMCTRL</math>寄存器的TSEL位选择)。</p> <p>注：CC1SEL仅在通道关闭时 (<math>TIMx\_CCEN</math>寄存器的<math>CC1EN=0</math>) 才是可写的。</p>																

#### 9.4.9 捕获/比较模式寄存器 2 (TIMx\_CCMOD2)

偏移地址：0x1C

复位值：0x0000 0000

参看以上 CCMOD1 寄存器的描述

输出比较模式:

31	Reserved	27	26	25	Reserved	22	21	20	17	16
		OC9PEN			OC8PEN			Reserved		OC7PEN
15	14	12	11	rw 10	9	8	7	rw 6	4	rw 0
OC4CEN	OC4MD[2:0]	OC4PEN	OC4FEN	CC4SEL[1:0]	OC3CEN	OC3MD[2:0]	OC3PEN	OC3FEN	CC3SEL[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:27	Reserved	保留，必须保持复位值。
26	OC9PEN	输出比较9预装载使能 (Output compare 4 preload enable) <i>注意：此功能仅适用于TIM1</i>
25:22	Reserved	保留，必须保持复位值。
21	OC8PEN	输出比较8预装载使能 (Output compare 4 preload enable) <i>注意：此功能仅适用于TIM1</i>
20:17	Reserved	保留，必须保持复位值。
16	OC7PEN	输出比较7预装载使能 (Output compare 4 preload enable) <i>注意：此功能仅适用于TIM1</i>
15	OC4CEN	输出比较4清0使能 (Output compare 4 clear enable)
14:12	OC4MD[2:0]	输出比较4模式 (Output compare 4 mode)
11	OC4PEN	输出比较4预装载使能 (Output compare 4 preload enable)
10	OC4FEN	输出比较4快速使能 (Output compare 4 fast enable)
9:8	CC4SEL[1:0]	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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。

位域	名称	描述
		注: CC4SEL 仅在通道关闭时 (TIMx_CCEN 寄存器的CC4EN=0) 才是可写的。
7	OC3CEN	输出比较3清0使能 (Output compare 3 clear enable)
6:4	OC3MD[2:0]	输出比较3模式 (Output compare 3 mode)
3	OC3PEN	输出比较3预装载使能 (Output compare 3 preload enable)
2	OC3FEN	输出比较3快速使能 (Output compare 3 fast enable)
1:0	CC3SEL[1:0]	<p>捕获/比较3选择 (Capture/Compare 3 selection)</p> <p>这2位定义通道的方向 (输入/输出), 及输入脚的选择:</p> <ul style="list-style-type: none"> <li>00: CC3通道被配置为输出;</li> <li>01: CC3通道被配置为输入, IC3映射在TI3上;</li> <li>10: CC3通道被配置为输入, IC3映射在TI4上;</li> <li>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</li> </ul> <p>注: CC3SEL 仅在通道关闭时 (TIMx_CCEN 寄存器的CC3EN=0) 才是可写的。</p>

输入捕获模式：

15	12	11	10	9	8	7	4	3	2	1	0
IC4F[3:0]	IC4PSC[1:0]	CC4SEL[1:0]			IC3F[3:0]		IC3PSC[1:0]	CC3SEL[1:0]			
rw	rw	rw			rw		rw	rw			

位域	名称	描述
15:12	IC4F[3:0]	输入捕获4滤波器 (Input capture 4 filter)
11:10	IC4PSC[1:0]	输入/捕获4预分频器 (Input capture 4 prescaler)
9:8	CC4SEL[1:0]	<p>捕获/比较4选择 (Capture/Compare 4 selection)</p> <p>这2位定义通道的方向(输入/输出), 及输入脚的选择:</p> <p>00: CC4通道被配置为输出;</p> <p>01: CC4通道被配置为输入, IC4映射在TI4上;</p>

位域	名称	描述
		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输入被选中时(由TIMx_SMCTRL寄存器的TSEL位选择)。 <i>注: CC4SEL仅在通道关闭时(TIMx_CCEN寄存器的CC4EN=0)才是可写的。</i>
7:4	IC3F[3:0]	输入捕获3滤波器 (Input capture 3 filter)
3:2	IC3PSC[1:0]	输入/捕获3预分频器 (Input capture 3 prescaler)
1:0	CC3SEL[1:0]	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输入被选中时(由TIMx_SMCTRL寄存器的TSEL位选择)。 <i>注: CC3SEL仅在通道关闭时(TIMx_CCEN寄存器的CC3EN=0)才是可写的。</i>

#### 9.4.10 捕获/比较使能寄存器 (TIMx\_CCEN)

偏移地址: 0x20

复位值: 0x0000 0000

31	Reserved												22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	rw	5	rw	4	3	2	rw	1	rw	0
CC4NP	CC4NEN	CC4P	CC4EN	CC3NP	CC3NEN	CC3P	CC3EN	CC2NP	CC2NEN	CC2P	CC2EN	CC1NP	CC1NEN	CC1P	CC1EN				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

位域	名称	描述
31:22	Reserved	保留, 必须保持复位值。
21	CC6P	捕获/比较6输出极性 (Capture/Compare 6 output polarity) 参考TIMx_CCEN.CC1P的描述。
20	CC6EN	捕获/比较6输出使能 (Capture/Compare 6 output enable)

位域	名称	描述
		参考TIMx_CCEN.CC1EN 的描述
19:18	Reserved	保留，必须保持复位值。
17	CC5P	捕获/比较5输出极性 (Capture/Compare 5 output polarity) 参考TIMx_CCEN.CC1P的描述。
16	CC5EN	捕获/比较5输出使能 (Capture/Compare 5 output enable) 参考TIMx_CCEN.CC1EN 的描述
15	CC4NP	捕获/比较4互补输出极性 (Capture/Compare 4 complementary output polarity) 参考TIMx_CCEN.CC1NP的描述。 <i>注意：此功能仅适用于 TIM1</i>
14	CC4NEN	捕获/比较4互补输出使能 (Capture/Compare 4 complementary output enable) 参考TIMx_CCEN.CC1NEN的描述。 <i>注意：此功能仅适用于 TIM1</i>
13	CC4P	捕获/比较4输出极性 (Capture/Compare 4 output polarity) 参考TIMx_CCEN.CC1P的描述。
12	CC4EN	捕获/比较4输出使能 (Capture/Compare 4 output enable) 参考TIMx_CCEN.CC1EN 的描述。
11	CC3NP	捕获/比较3互补输出极性 (Capture/Compare 3 complementary output polarity) 参考TIMx_CCEN.CC1NP的描述。
10	CC3NEN	捕获/比较3互补输出使能 (Capture/Compare 3 complementary output enable) 参考TIMx_CCEN.CC1NEN的描述。
9	CC3P	捕获/比较3输出极性 (Capture/Compare 3 output polarity) 参考TIMx_CCEN.CC1P的描述。
8	CC3EN	捕获/比较3输出使能 (Capture/Compare 3 output enable) 参考TIMx_CCEN.CC1E 的描述。
7	CC2NP	捕获/比较2互补输出极性 (Capture/Compare 2 complementary output polarity) 参考TIMx_CCEN.CC1NP的描述。

位域	名称	描述
6	CC2NEN	捕获/比较2互补输出使能 (Capture/Compare 2 complementary output enable) 参考TIMx_CCEN.CC1NEN的描述。
5	CC2P	捕获/比较2输出极性 (Capture/Compare 2 output polarity) 参考TIMx_CCEN.CC1P的描述。
4	CC2EN	捕获/比较2输出使能 (Capture/Compare 2 output enable) 参考TIMx_CCEN.CC1EN的描述。
3	CC1NP	捕获/比较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。
2	CC1NEN	捕获/比较1互补输出使能 (Capture/Compare 1 complementary output enable) 0: 禁用 - 禁用输出 OC1N 信号。 OC1N 的电平取决于TIMx_BKDT.MOEN、TIMx_BKDT.OSSI、TIMx_BKDT.OSSR、TIMx_CTRL2.OI1、TIMx_CTRL2.OI1N 和 TIMx_CCEN.CC1EN 的值。 1: 使能 - 使能输出 OC1N 信号。 OC1N 的电平取决于TIMx_BKDT.MOEN、TIMx_BKDT.OSSI、TIMx_BKDT.OSSR、TIMx_CTRL2.OI1、TIMx_CTRL2.OI1N 和 TIMx_CCEN.CC1EN 的值。
1	CC1P	捕获/比较1输出极性 (Capture/Compare 1 output polarity) <b>CC1对应通道为输出模式时:</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1对应通道为输入模式时:</b> 此时, 该位用于选择是使用IC1还是IC1的反相信号作为触发信号或捕捉信号。 0: 非反相: 当 IC1 产生上升沿时发生捕获动作。 当用作外部触发时, IC1 是非反相的。 1: 反相: 当 IC1 产生下降沿时发生捕获动作。 当用作外部触发时, IC1 被反相。
0	CC1EN	捕获/比较1输出使能 (Capture/Compare 1 output enable) <b>CC1通道配置为输出:</b> 0: 关闭 - OC1禁止输出, 因此OC1的输出电平依赖于MOEN、OSSI、OSSR、OI1、OI1N和CC1NEN位的值。 1: 开启 - OC1信号输出到对应的输出引脚, 其输出电平依赖于MOEN、OSSI、OSSR、

位域	名称	描述
		<p>OI1、OI1N和CC1NEN位的值。</p> <p><b>CC1通道配置为输入：</b></p> <p>该位决定了计数器的值是否能捕获入TIMx_CCDAT1寄存器。</p> <p>0: 捕获禁止;</p> <p>1: 捕获使能。</p>

表 9-4 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 <sup>(1)</sup>	
MOEN	OSSI	OSSR	CCxEN	CCxNEN	OCx 输出状态	OCxN 输出状态
1 X		0	0	0	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	输出禁止 (与定时器断开) OCxN=0, OCxN_EN=0
		0	0	1	输出禁止 (与定时器断开) OCx=0, OCx_EN=0	OCxREF + 极性, OCxN= OCxREF xor CCxNP, OCxN_EN=1
		0	1	0	OCxREF + 极性, OCx= OCxREF 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 反相 + 极性 + 死区, OCxN_EN=1
		1	0	0	输出禁止 (与定时器断开) OCx=CCxP, OCx_EN=0	输出禁止 (与定时器断开) OCxN=CCxNP, 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= OCxREF xor CCxNP, OCxN_EN=1
		1	1	0	OCxREF + 极性,	关闭状态 (输出使能且为无效电平)

					OCx= OCxREF xor CCxP, OCx_EN=1	OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCxREF + 极性 + 死区, OCx_EN=1	OCxREF 反相 + 极性 + 死区, OCxN_EN=1
0	0	X	0	0	输出禁止 (与定时器断开)	
	0		0	1	异步: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0;	
	0		1	0	若时钟存在: 假设 (CCxP ^ OIx) ^ (CCxNP^OIxN)! = 0,	
	0		1	1	经过一个死区时间后 OCx=OIx, OCxN=OIxN	
	1		0	0	关闭状态 (输出使能且为无效电平)	
	1		0	1	异步: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1;	
	1		1	0	若时钟存在: 假设 (CCxP ^ OIx) ^ (CCxNP^OIxN)! = 0,	
	1		1	1	经过一个死区时间后 OCx=OIx, OCxN=OIxN,	

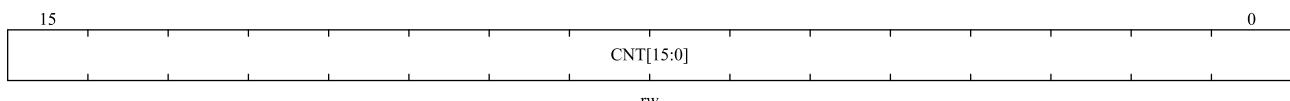
- 如果一个通道的 2 个输出都没有使用 (CCxEN = CCxNEN = 0), 那么 OIx, OIxN, CCxP 和 CCxNP 都必须清零。

注: 引脚连接到互补的 OCx 和 OCxN 通道的外部 I/O 引脚的状态, 取决于 OCx 和 OCxN 通道状态和 GPIO 以及 AFIO 寄存器。

#### 9.4.11 计数器 (TIMx\_CNT)

偏移地址: 0x24

复位值: 0x0000

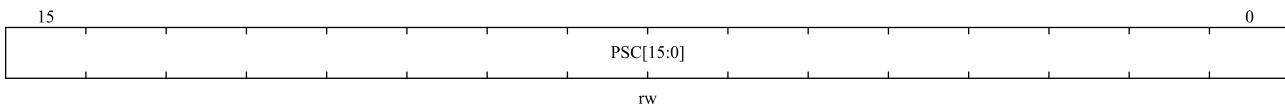


位域	名称	描述
15:0	CNT[15:0]	计数器的值 (Counter value)

#### 9.4.12 预分频器 (TIMx\_PSC)

偏移地址: 0x28

复位值: 0x0000

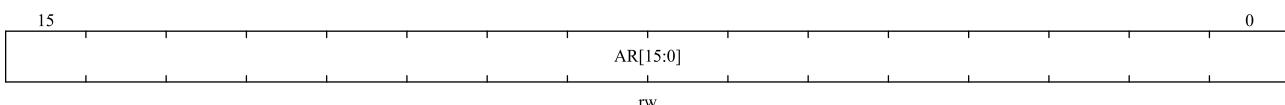


位域	名称	描述
15:0	PSC[15:0]	<p>预分频器的值 (Prescaler value)</p> <p>计数器时钟 <math>f_{CK\_CNT} = f_{CK\_PSC} / (PSC [15:0] + 1)</math>。</p> <p>每次发生更新事件时，PSC 值都会加载到预分频器的影子寄存器中。</p>

#### 9.4.13 自动重装载寄存器 (TIMx\_AR)

偏移地址: 0x2C

复位值: 0xFFFF

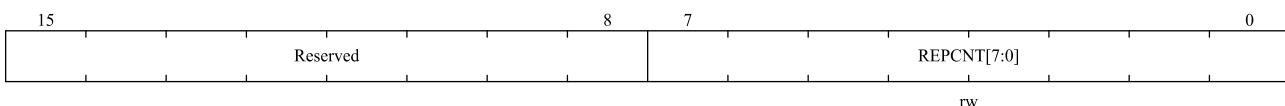


位域	名称	描述
15:0	AR[15:0]	<p>自动重装载的值 (Auto-reload value)</p> <p>AR包含了将要装载入实际的自动重装载寄存器的值。详细参考9.3.1节：有关AR的更新和动作。</p> <p>当自动重装载的值为空时，计数器不工作。</p>

#### 9.4.14 重复计数寄存器 (TIMx\_REPCNT)

偏移地址: 0x30

复位值: 0x0000



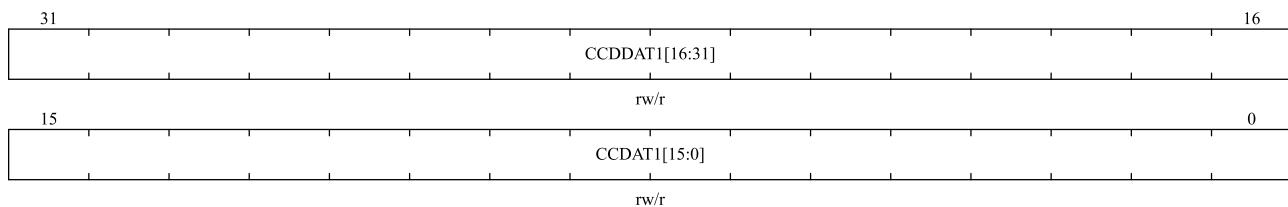
位域	名称	描述
15:8	Reserved	保留，必须保持复位值。

位域	名称	描述
7:0	REPCNT[7:0]	<p>重复计数器的值 (Repetition counter value)</p> <p>重复计数器仅在给定数量 (<math>N+1</math>) 个计数器周期后用于生成更新事件或更新定时器寄存器，其中 <math>N</math> 是 TIMx_REPCNT.REPCNT 的值。在向上计数模式下，每次计数器溢出，向下计数模式下每次计数器下溢或中央对齐模式下每次计数器溢出和每次计数器下溢时，重复计数器都会递减。设置 TIMx_EVTGEN.UDGN 位将重新加载 TIMx_REPCNT.REPCNT 的内容并生成更新事件。</p>

#### 9.4.15 捕获/比较寄存器 1 (TIMx\_CC DAT1)

偏移地址: 0x34

复位值: 0x0000 0000



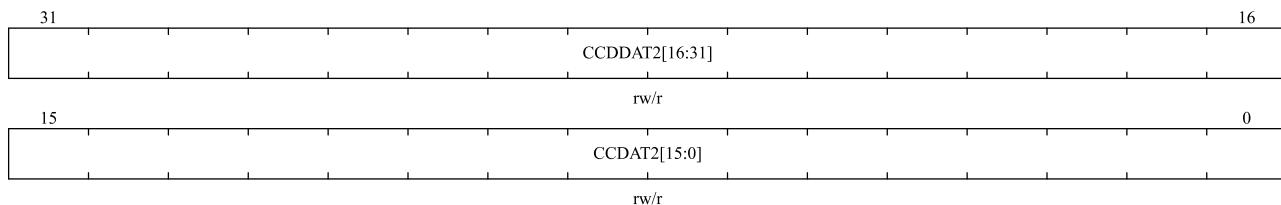
位域	名称	描述
31:16	CCDDAT1[15:0]	<p>捕获/比较通道1向下计数值(Capture/Compare 1 down-counting value)，专用于中央对齐非对称模式。仅适用于 TIM1。</p> <ul style="list-style-type: none"> <li>■ CC1 通道只能配置为输出：</li> </ul> <p>CCDDAT1 包含要与计数器 TIMx_CNT 比较的值（仅当 TIMx_CTRL1.DIR = 1 且处于非对称模式时），在 OC1 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC1PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p>
15:0	CCDAT1[15:0]	<p>捕获/比较通道1的值 (Capture/Compare 1 value)</p> <ul style="list-style-type: none"> <li>■ CC1 通道配置为输出：</li> </ul> <p>CCDAT1 包含要与计数器 TIMx_CNT 比较的值（当 TIMx_CTRL1.DIR = 1 且处于非对称模式时除外），在 OC1 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC1PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC1 通道配置为输入：</li> </ul> <p>CCDAT1 包含由最后一个输入捕获 1 事件 (IC1) 传输的计数器值。</p>

位域	名称	描述
		当配置为输入模式时，寄存器 CCDDAT1 和 CCDDAT2 只能读取。 当配置为输出模式时，寄存器 CCDDAT1 和 CCDDAT2 是可读写的。

#### 9.4.16 捕获/比较寄存器 2 (TIMx\_CC DAT2)

偏移地址: 0x38

复位值: 0x0000 0000

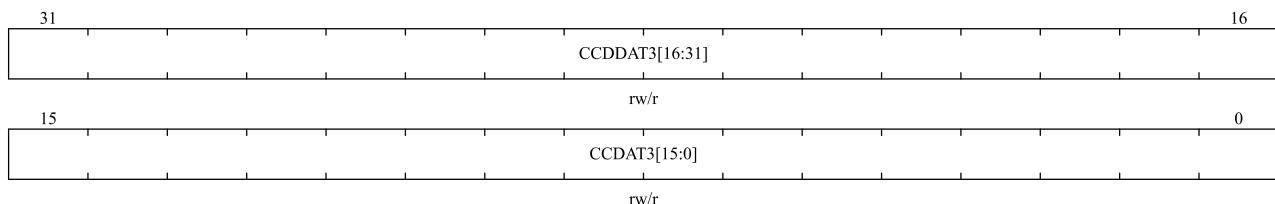


位域	名称	描述
31:16	CCDDAT2[15:0]	<p>捕获/比较通道2向下计数值(Capture/Compare 2 down-counting value)，专用于中央对齐非对称模式。仅适用于 TIM1。</p> <ul style="list-style-type: none"> <li>■ CC2 通道只能配置为输出：</li> </ul> <p>CCDDAT2 包含要与计数器 TIMx_CNT 比较的值（仅当 TIMx_CTRL1.DIR = 1 且处于非对称模式时），在 OC2 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC2PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p>
15:0	CCDAT2[15:0]	<p>捕获/比较通道2的值 (Capture/Compare 2 value)</p> <ul style="list-style-type: none"> <li>■ CC2 通道配置为输出：</li> </ul> <p>CCDAT2 包含要与计数器 TIMx_CNT 比较的值（当 TIMx_CTRL1.DIR = 1 且处于非对称模式时除外），在 OC2 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC2PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC2 通道配置为输入：</li> </ul> <p>CCDAT2 包含由最后一个输入捕获 2 事件 (IC2) 传输的计数器值。</p> <p>当配置为输入模式时，寄存器 CCDDAT2 和 CCDDAT2 只能读取。</p> <p>当配置为输出模式时，寄存器 CCDDAT2 和 CCDDAT2 是可读写的。</p>

### 9.4.17 捕获/比较寄存器 3 (TIMx\_CC DAT3)

偏移地址: 0x3C

复位值: 0x0000 0000

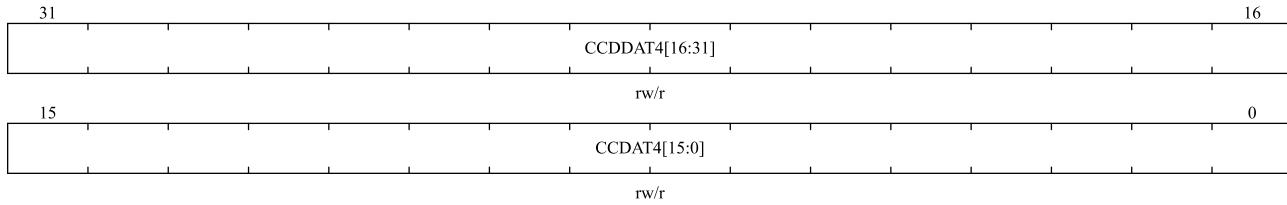


位域	名称	描述
31:16	CCDDAT3[15:0]	<p>捕获/比较通道3向下计数值(Capture/Compare 3 down-counting value), 专用于中央对齐非对称模式。仅适用于 TIM1。</p> <ul style="list-style-type: none"> <li>■ CC3 通道只能配置为输出:</li> </ul> <p>CCDDAT3 包含要与计数器 TIMx_CNT 比较的值（仅当 TIMx_CTRL1.DIR = 1 且处于非对称模式时），在 OC3 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC3PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p>
15:0	CCDAT3[15:0]	<p>捕获/比较通道3的值 (Capture/Compare 3 value)</p> <ul style="list-style-type: none"> <li>■ CC3 通道配置为输出:</li> </ul> <p>CCDAT3 包含要与计数器 TIMx_CNT 比较的值（当 TIMx_CTRL1.DIR = 1 且处于非对称模式时除外），在 OC3 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC3PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC3 通道配置为输入:</li> </ul> <p>CCDAT3 包含由最后一个输入捕获 3 事件 (IC3) 传输的计数器值。</p> <p>当配置为输入模式时，寄存器 CCDAT3 和 CCDDAT3 只能读取。</p> <p>当配置为输出模式时，寄存器 CCDAT3 和 CCDDAT3 是可读写的。</p>

### 9.4.18 捕获/比较寄存器 4 (TIMx\_CC DAT4)

偏移地址: 0x40

复位值: 0x0000 0000

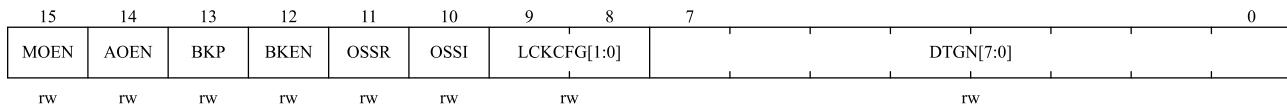


位域	名称	描述
31:16	CCDDAT4[15:0]	<p>捕获/比较通道4向下计数值(Capture/Compare 4 down-counting value), 专用于中央对齐非对称模式。仅适用于 TIM1。</p> <ul style="list-style-type: none"> <li>■ CC4 通道只能配置为输出:</li> </ul> <p>CCDDAT4 包含要与计数器 TIMx_CNT 比较的值（仅当 TIMx_CTRL1.DIR = 1 且处于非对称模式时），在 OC4 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC4PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p>
15:0	CCDAT4[15:0]	<p>捕获/比较通道4的值 (Capture/Compare 4 value)</p> <ul style="list-style-type: none"> <li>■ CC4 通道配置为输出:</li> </ul> <p>CCDAT4 包含要与计数器 TIMx_CNT 比较的值（当 TIMx_CTRL1.DIR = 1 且处于非对称模式时除外），在 OC4 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC4PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC4 通道配置为输入:</li> </ul> <p>CCDAT4 包含由最后一个输入捕获 4 事件 (IC4) 传输的计数器值。</p> <p>当配置为输入模式时，寄存器 CCDAT4 和 CCDDAT4 只能读取。</p> <p>当配置为输出模式时，寄存器 CCDAT4 和 CCDDAT4 是可读写的。</p>

#### 9.4.19 刹车和死区寄存器 (TIMx\_BKDT)

偏移地址: 0x44

复位值: 0x0000



**注释:** 根据锁定设置, AOEN、BKP、BKEN、OSSR 和 DTGN[7:0]位均可被写保护, 有必要在第一次写入 TIMx\_BKDT 寄存器时对它们进行配置。

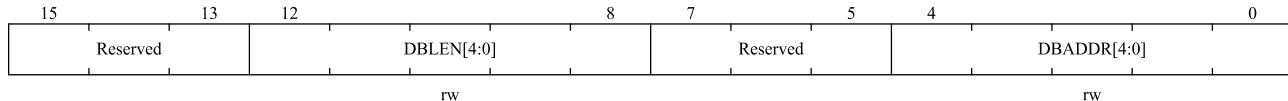
位域	名称	描述
15	MOEN	<p>主输出使能 (Main output enable)</p> <p>该位可由软件或硬件根据 TIMx_BKDT.AOEN 位设置，一旦刹车输入有效，该位由硬件异步清零。它仅对配置为输出的通道有效。</p> <p>0: OC 和 OCN 输出被禁用或强制进入空闲状态。</p> <p>1: 如果设置了 TIMx_CCEN.CCxEN 或 TIMx_CCEN.CCxNEN 位，则使能 OC 和 OCN 输出。有关更多详细信息，请参见章节9.4.10。</p>
14	AOEN	<p>自动输出使能 (Automatic output enable)</p> <p>0: 只有软件可以设置TIMx_BKDT.MOEN;</p> <p>1: 软件设置TIMx_BKDT.MOEN; 或者如果刹车输入未激活，则在下一次更新事件发生时，硬件自动设置 TIMx_BKDT.MOEN。</p>
13	BKP	<p>刹车输入极性 (Break polarity)</p> <p>0: 刹车输入低电平有效;</p> <p>1: 刹车输入高电平有效。</p> <p>注: 任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。</p>
12	BKEN	<p>刹车功能使能 (Break enable)</p> <p>0: 禁止刹车输入 (BRK及CCS时钟失效事件)；</p> <p>1: 开启刹车输入 (BRK及CCS时钟失效事件)。</p> <p>注: 任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。</p>
11	OSSR	<p>当 TIMx_BKDT.MOEN=1 且通道为互补输出时使用该位。</p> <p>没有互补输出的定时器中不存在 OSSR 位。</p> <p>0: 当定时器不工作时，禁用OCx/OCxN输出 (OCx/OCxN使能输出信号 = 0)</p> <p>1: 当定时器不工作时，一旦CCxEN = 1或CCxNEN = 1，首先开启OCx/OCxN并输出无效电平，然后置OCx/OCxN使能输出信号 =1</p> <p>有关更多详细信息，请参见第 9.4.10 节，捕获/比较启用寄存器 (TIMx_CCEN)。</p>
10	OSSI	<p>空闲模式下“关闭状态”选择 (Off-state selection for Idle mode)</p> <p>当 TIMx_BKDT.MOEN=0 且通道配置为输出时使用该位。</p> <p>0: 当定时器不工作时，禁用OCx/OCxN输出 (OCx/OCxN使能输出信号 = 0)</p> <p>1: 当定时器不工作时，一旦CCxEN = 1或CCxNEN = 1，首先开启OCx/OCxN并输出空闲电平，然后置OCx/OCxN使能输出信号 =1</p>

位域	名称	描述
		有关更多详细信息, 请参见第 9.4.10 节, 捕获/比较启用寄存器 (TIMx_CCEN)。
9:8	LCKCFG[1:0]	<p>锁定设置 (Lock configuration) 该位为防止软件错误而提供写保护。</p> <p>这些位提供针对软件错误的写保护。</p> <p>00:</p> <ul style="list-style-type: none"> <li>- 没有写保护。</li> </ul> <p>01:</p> <ul style="list-style-type: none"> <li>- 锁定级别 1</li> </ul> <p>TIMx_BKDT.DTGN、TIMx_BKDT.BKEN、TIMx_BKDT.BKP、TIMx_BKDT.AOEN、 TIMx_CTRL2.OIx、TIMx_CTRL2.OIxN 位启用写保护。</p> <p>10:</p> <ul style="list-style-type: none"> <li>- 锁定 2 级</li> </ul> <p>除了 LOCK Level 1 模式下的寄存器写保护外, TIMx_CCEN.CCxP 和 TIMx_CCEN.CCxNP (如果相应通道配置为输出模式), TIMx_BKDT.OSSR 和 TIMx_BKDT.OSSI 位也使能写保护。</p> <p>11:</p> <ul style="list-style-type: none"> <li>- 锁定 3 级</li> </ul> <p>除了 LOCK Level 2 中的寄存器写保护外, TIMx_CCMODx.OCxMD 和 TIMx_CCMODx.OCxPEN 位 (如果相应通道配置为输出模式) 也启用写保护。</p> <p>注意: 系统复位后, LCKCFG 位只能写一次。一旦写入 TIMx_BKDT 寄存器, LCKCFG 将受到保护, 直到下一次复位。</p>
7:0	DTGN[7:0]	<p>死区发生器设置 (Dead-time generator setup)</p> <p>这些位定义插入的互补输出之间的死区持续时间。 DTGN值与死区时间的关系如下:</p> <p>DTGN[7:5]=0xx =&gt; DT=DTGN[7:0] × T<sub>dtgn</sub>, T<sub>dtgn</sub> = T<sub>DTS</sub>;</p> <p>DTGN[7:5]=10x =&gt; DT=(64+DTGN[5:0]) × T<sub>dtgn</sub>, T<sub>dtgn</sub> = 2 × T<sub>DTS</sub>;</p> <p>DTGN[7:5]=110 =&gt; DT=(32+DTGN[4:0]) × T<sub>dtgn</sub>, T<sub>dtgn</sub> = 8 × T<sub>DTS</sub>;</p> <p>DTGN[7:5]=111 =&gt; DT=(32+DTGN[4:0]) × T<sub>dtgn</sub>, T<sub>dtgn</sub> = 16 × T<sub>DTS</sub>;</p>

## 9.4.20 DMA 控制寄存器 (TIMx\_DCTRL)

偏移地址: 0x48

复位值: 0x0000

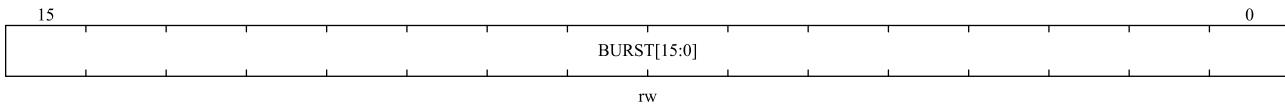


位域	名称	描述
15:13	Reserved	保留, 必须保持复位值。
12:8	DBLEN[4:0]	<p>DMA连续传送长度 (DMA burst length)</p> <p>该位字段定义 DMA 将访问 (写入/读取) TIMx_DADDR 寄存器的次数。</p> <p>00000: 1次传输 00001: 2次传输 00010: 3次传输 ... 10001: 18次传输</p>
7:5	Reserved	保留, 必须保持复位值。
4:0	DBADDR[4:0]	<p>DMA地址 (DMA base address)</p> <p>该位字段定义 DMA 访问 TIMx_DADDR 寄存器的第一个地址。</p> <p>当第一次通过 TIMx_DADDR 完成访问时, 该位域指定您刚刚访问的地址。然后第二次访问TIMx_DADDR, 会访问到“DMA Base Address + 4”的地址</p> <p>00000: TIMx_CTRL1, 00001: TIMx_CTRL2, 00010: TIMx_SMCTRL, ..... 10001: TIMx_BKDT, 10010: TIMx_DCTRL</p>

## 9.4.21 连续模式的 DMA 地址 (TIMx\_DADDR)

偏移地址: 0x4C

复位值: 0x0000

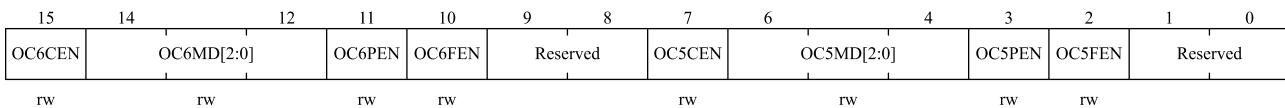


位域	名称	描述
15:0	BURST[15:0]	<p>DMA 访问缓冲区。</p> <p>当对该寄存器分配读或写操作时，将访问位于地址范围（DMA base address + DMA burst length × 4）的寄存器。</p> <p>DMA base address = The address of TIM_CTRL1 + TIMx_DCTRL.DBADDR * 4;</p> <p>DMA burst len = TIMx_DCTRL.DBLEN + 1.</p> <p>例子：</p> <p>如果 TIMx_DCTRL.DBLEN = 0x3（4 次传输），TIMx_DCTRL.DBADDR = 0xD (TIMx_CCDAT1)，DMA 数据长度 = 半字，DMA 存储器地址 = SRAM 中的缓冲区地址，DMA 外设地址 = TIMx_DADDR 地址。</p> <p>当事件发生时，TIMx 将向 DMA 发送请求，并传输 4 次数据。</p> <p>第一次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT1 寄存器；</p> <p>第二次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT2 寄存器；</p> <p>.....</p> <p>第四次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT4 寄存器；</p>

#### 9.4.22 捕获/比较寄存器 (TIMx\_CCMOD3)

偏移地址: 0x54

复位值: 0x0000



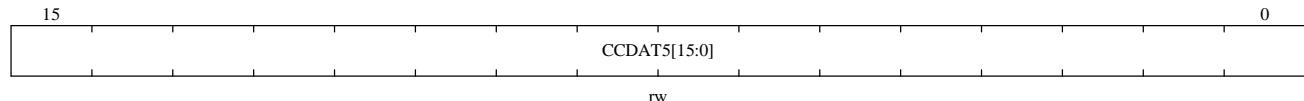
位域	名称	描述
15	OC6CEN	输出比较6清0使能 (Output compare 6 clear enable)
14:12	OC6MD[2:0]	输出比较6模式 (Output compare 6 mode)
11	OC6PEN	输出比较6预装载使能 (Output compare 6 preload enable)
10	OC6FEN	输出比较6快速使能 (Output compare 6 fast enable)

位域	名称	描述
9:8	Reserved	保留，必须保持复位值。
7	OC5CEN	输出比较5清0使能 (Output compare 3 clear enable)
6:4	OC5MD[2:0]	输出比较5模式 (Output compare 3 mode)
3	OC5PEN	输出比较5预装载使能 (Output compare 5 preload enable)
2	OC5FEN	输出比较5快速使能 (Output compare 5 fast enable)
1:0	Reserved	保留，必须保持复位值。

#### 9.4.23 捕获/比较寄存器 5 (TIMx\_CC DAT5)

偏移地址: 0x58

复位值: 0x0000

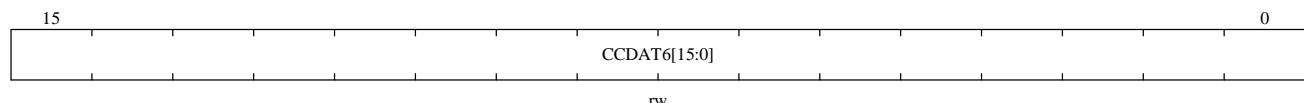


位域	名称	描述
15:0	CC DAT5[15:0]	<p>捕获/比较通道5的值 (Capture/Compare 5 value)</p> <p>■ CC5 通道只能配置为输出:</p> <p>CC DAT5 包含要与计数器 TIMx_CNT 比较的值，在 OC5 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD3.OC5PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <p>CC5 用于比较器消隐。</p>

#### 9.4.24 捕获/比较寄存器 6 (TIMx\_CC DAT6)

偏移地址: 0x5C

复位值: 0x0000

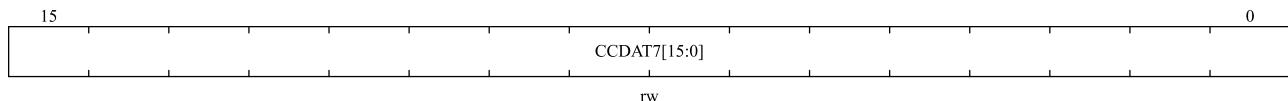


位域	名称	描述
15:0	CCDAT6[15:0]	<p>捕获/比较通道6的值 (Capture/Compare 6 value)</p> <ul style="list-style-type: none"> <li>■ CC6 通道只能配置为输出:</li> </ul> <p>CCDAT6 包含要与计数器 TIMx_CNT 比较的值，在 OC6 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD3.OC6PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p>

#### 9.4.25 捕获/比较寄存器 7(TIMx\_CC DAT7)

Offset address: 0x60

Reset value: 0x0000

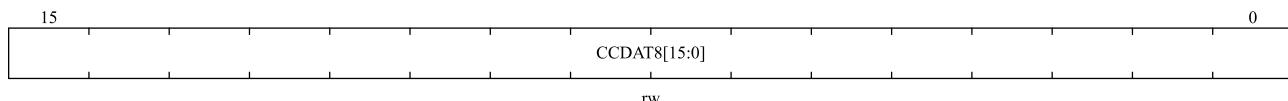


位域	名称	描述
15:0	CCDAT7[15:0]	<p>捕获/比较通道7的值 (Capture/Compare 7 value)</p> <ul style="list-style-type: none"> <li>■ CC7 通道只能配置为输出:</li> </ul> <p>CCDAT7 包含要与计数器 TIMx_CNT 比较的值，在 OC7 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD3.OC7PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <p>注: 此功能仅适用于TIM1</p>

#### 9.4.26 捕获/比较寄存器 8 (TIMx\_CC DAT8)

Offset address: 0x64

Reset value: 0x0000

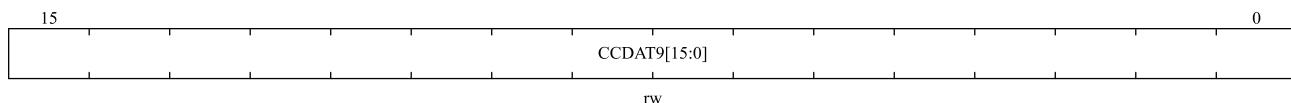


位域	名称	描述
15:0	CCDAT8[15:0]	<p>捕获/比较通道8的值 (Capture/Compare 8 value)</p> <p>■ CC8 通道只能配置为输出:</p> <p>CCDAT8 包含要与计数器 TIMx_CNT 比较的值，在 OC8 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC8PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <p>注: 此功能仅适用于TIM1</p>

#### 9.4.27 捕获/比较寄存器 9 (TIMx\_CC DAT9)

Offset address: 0x68

Reset value: 0x0000

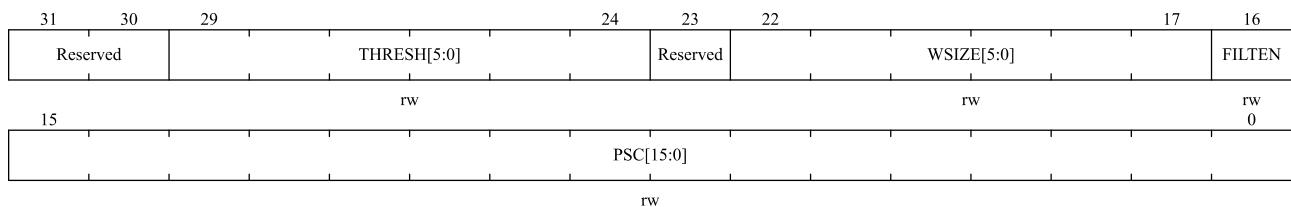


位域	名称	描述
15:0	CCDAT9[15:0]	<p>捕获/比较通道9的值 (Capture/Compare 9 value)</p> <p>■ CC9 通道只能配置为输出:</p> <p>CCDAT9 包含要与计数器 TIMx_CNT 比较的值，在 OC9 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC9PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <p>注: 此功能仅适用于TIM1</p>

#### 9.4.28 刹车滤波 (TIMx\_BRKFILT)

Offset address: 0x6C

Reset value: 0x0000 0000



位域	名称	描述
31:30	Reserved	保留，必须保持复位值。
29:24	THRESH[5:0]	<p>采样逻辑电平有效的阈值数(Threshold)，最大 63：</p> <p>有效逻辑电平的阈值。在采样窗口内，如果逻辑高的数量大于或等于阈值，则下一个逻辑电平将为逻辑高。同样的规则适用于逻辑低。如果窗口内 1 和 0 的数量都小于阈值，则过滤器输出保持不变。阈值应设置为大于或等于 Window 值的一半。</p> <p>推荐阈值范围为：</p> <p><b>最小值：</b>比最大毛刺大小的上限（预分频时钟周期）多 1 个预分频时钟周期，并且需要大于窗口大小的一半。</p> <p>例如，如果毛刺大小为 3.2*（预分频时钟周期），则阈值应为 <math>\lceil 3.2 \rceil = 4 + 1 = 5</math></p> <p><b>最大值：</b>有效信号最小尺寸的底值（在预分频时钟周期内），需要小于窗口尺寸。</p> <p>例如，如果最小信号大小为 3.2*（预分频时钟周期），则阈值应为下限 <math>(3.2) = 3</math>。</p>
23	Reserved	保留，必须保持复位值。
22:17	WSIZE[5:0]	<p>逻辑电平检查的窗口大小值（Window size），最大 63：</p> <p>窗口大小决定了在获得下一个逻辑级别时将考虑多少采样值。内置 FIFO 为 64 位，最大索引为 63，只能将窗口大小设置为 63。</p>
16	FILTEN	<p>滤波器使能（Filter enable）：</p> <p>0: 滤波器禁能。</p> <p>1: 滤波器使能。</p>
15:0	PSC[15:0]	<p>用于配置滤波器采样时钟的预分频寄存器值（Prescaler）：</p> <p>对于此过滤器，它支持 65535 比例（16 位）。</p> <p>时钟预分频器将系统时钟缩放到采样时钟。采样时钟决定两个采样点之间的距离。只有采样点的值才会考虑有效的逻辑电平计算。</p>

## 10 通用定时器（TIM2、TIM3、TIM4 和 TIM5）

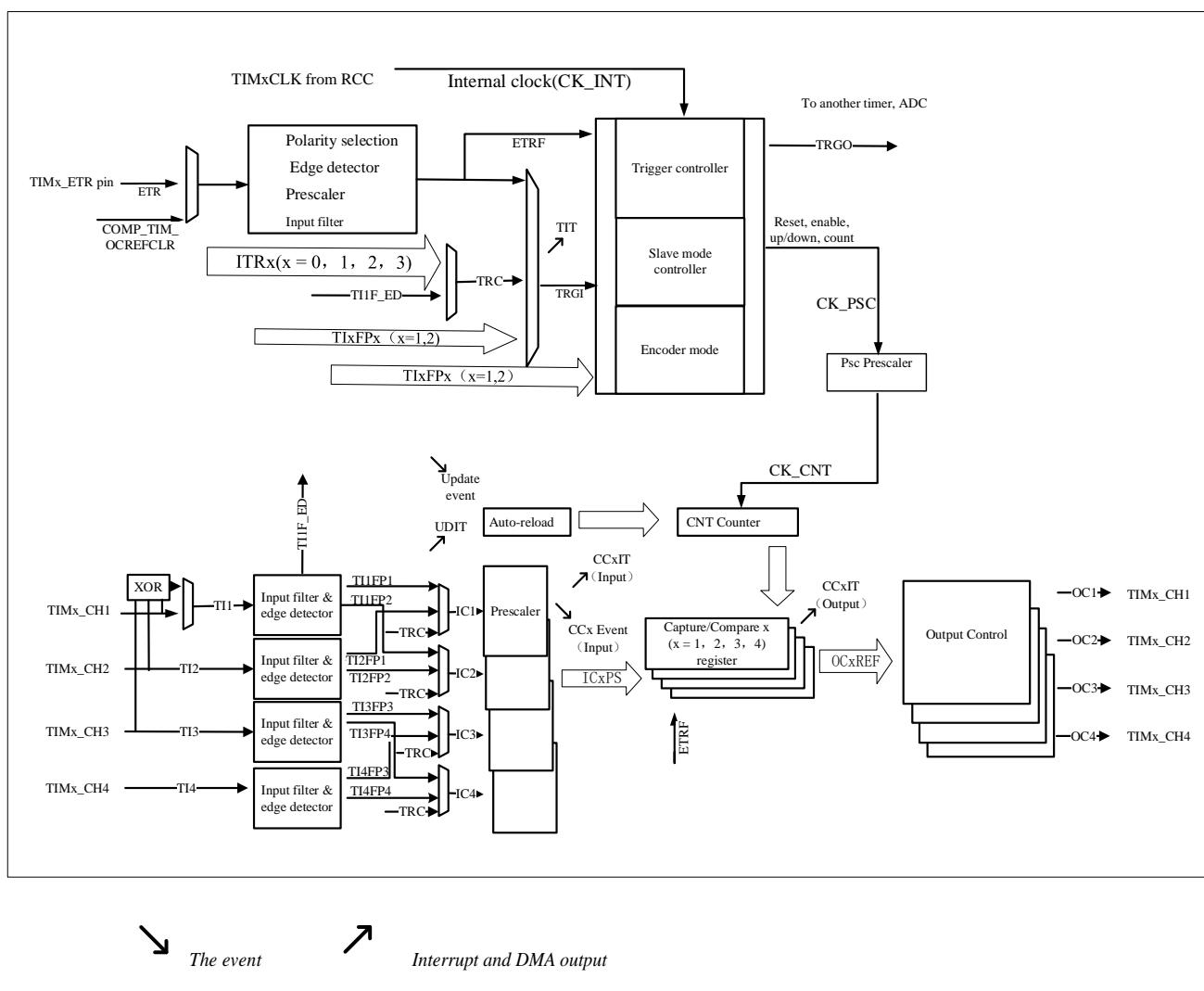
### 10.1 TIM2、TIM3、TIM4 和 TIM5 简介

通用定时器（TIM2、TIM3、TIM4 和 TIM5）主要用于以下场合：对输入信号进行计数、测量输入信号的脉冲宽度和产生输出波形等。

### 10.2 TIM2、TIM3、TIM4 和 TIM5 主要特性

- 16 位自动装载计数器。（可实现向上计数、向下计数、向上/下计数）
- 16 位可编程预分频器。（分频系数可配置为 1 到 65536 之间的任意值）
- TIM2、TIM3、TIM4 和 TIM5 最多支持 4 个通道
- 通道工作模式：PWM 输出、输出比较、单脉冲模式输出、输入捕获
- 如下事件发生时产生中断/DMA：
  - ◆ 更新事件
  - ◆ 触发事件
  - ◆ 输入捕获
  - ◆ 输出比较
- 可通过外部信号控制定时器
- 多个定时器从内部连接在一起，以实现定时器同步或链接
- 增量（正交）编码器接口：用于追踪运行轨迹和解析旋转方位
- 霍尔传感器接口：用于三相电机控制

图 10-1 TIMx ( $x=2,3,4$  and  $5$ ) 框图



For  $\text{TIM}_x$  ( $x = 2, 3, 4$ ), the ETR input is from IOM only, for  $\text{TIM}_5$ , the ETR input is not support.

For  $\text{TIM}_x$  ( $x = 2, 3, 4, 5$ ), The capture channel 1 input can come from IOM or comparator output

For  $\text{TIM}_2$ , capture channel 2 comes from IOM or LSE, for  $\text{TIM}_3$ ,  $\text{TIM}_4$  and  $\text{TIM}_5$ , capture channel 2 comes from IOM only

For  $\text{TIM}_2$ , capture channel 3 comes from IOM or LSI, for  $\text{TIM}_3$ ,  $\text{TIM}_4$  and  $\text{TIM}_5$ , capture channel 3 comes from IOM only

For  $\text{TIM}_2$ , capture channel 4 comes from IOM or HSE/I28, for  $\text{TIM}_3$ ,  $\text{TIM}_4$  and  $\text{TIM}_5$ , capture channel 4 comes from IOM only

For each GPTIM, there is a sliding filter in front of  $\text{TIM}_x\text{-CH}1$ ,  $\text{TIM}_x\text{-CH}2$ ,  $\text{TIM}_x\text{-CH}3$  and  $\text{TIM}_x\text{-CH}4$ . Sliding filter is only for IOM input, not for LSE/LSI/HSE input.

## 10.3 TIM2、TIM3、TIM4、TIM5 功能描述

### 10.3.1 时基单元

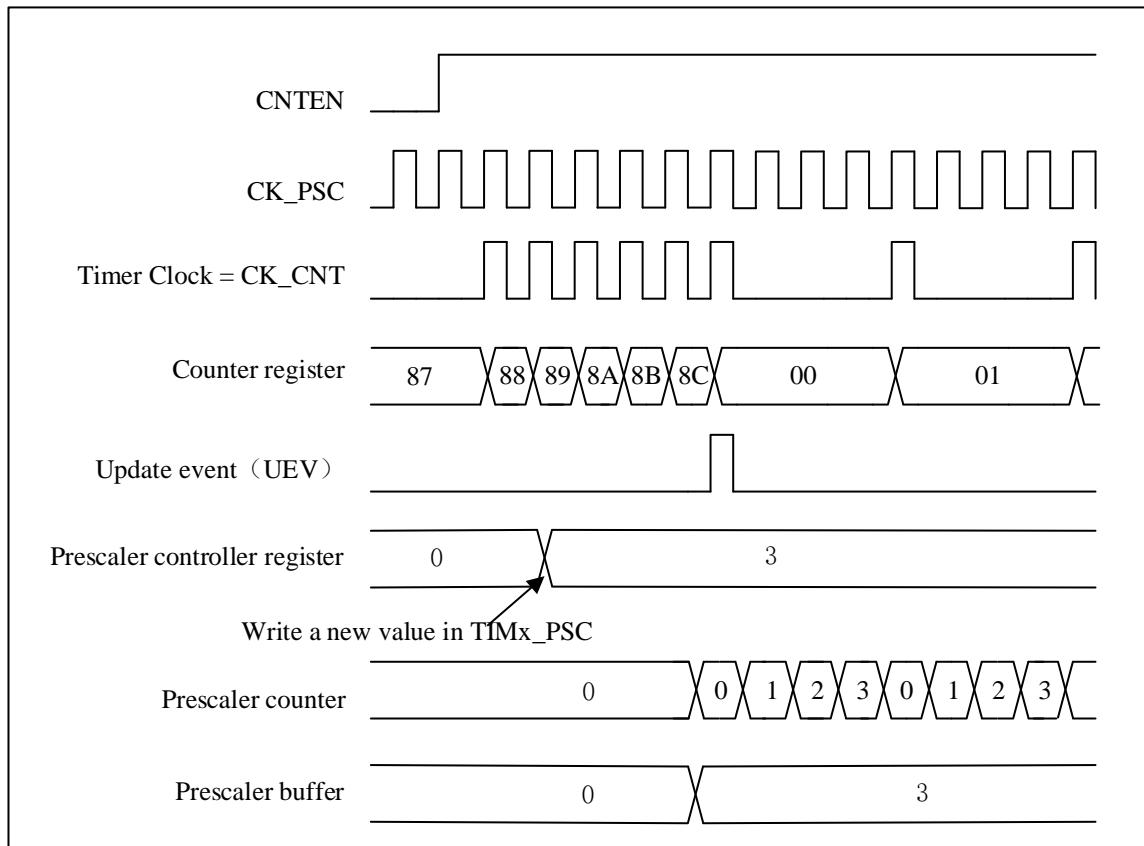
通用器的时基单元主要包括：预分频器、计数器和自动重装载寄存器。当时基单元工作时，软件可以随时读取和写入相应的寄存器（ $\text{TIM}_x\text{-PSC}$ 、 $\text{TIM}_x\text{-CNT}$  和  $\text{TIM}_x\text{-AR}$ ）。

根据自动重装载预装载使能位 (TIMx\_CTRL1.ARREN) 的设置, 预装载寄存器的值会立即或在每次更新事件 UEV 时传输到影子寄存器。TIMx\_CTRL1.UPDIS=0 时, 当计数器达到上溢/下溢条件或软件设置 TIMx\_EVTGEN.UDGN, 将生成更新事件。计数器 CK\_CNT 仅在 TIMx\_CTRL1.CNTEN 位被设置时有效。计数器在 TIMx\_CTRL.CNTEN 位被设置后一个时钟周期之后开始计数。

### 10.3.1.1 预分频器描述

TIMx\_PSC 寄存器由一个 16 位计数器组成, 可用于计数器时钟频率按 1 和 65536 之间的任意分频。因为这个控制器带有缓冲器, 可以在运行时动态改变。新的预分频器值只有在下次更新事件中才会被采用。

图 10-2 当预分频的参数从 1 到 4, 计数器的时序图



### 10.3.2 计数器模式

#### 10.3.2.1 向上计数模式

使用向上计数模式, 计数器将从 0 计数到寄存器 TIMx\_AR 的值, 然后重置为 0。并产生一个计数器溢出事件。

如果设置 TIMx\_CTRL1.UPRS 位(选择更新请求)和 TIMx\_EVTGEN.UDGN 位, 将产生一个更新事件(UEV)。但是 TIMx\_STS.UDITF 不会被硬件置起, 因此不会产生更新中断或 DMA 更新请求。这是为了避免清除计数器时产生更新中断。

取决于 TIMx\_CTRL1.UPRS 的配置, 当发生更新事件时, 所有寄存器都会更新, TIMx\_STS.UDITF 被设置

- 当  $\text{TIMx\_CTRL1.ARPE} = 1$ , 预装载寄存器( $\text{TIMx\_AR}$ )的值被更新到自动装载影子寄存器
- 预加载值 ( $\text{TIMx\_PSC}$ ) 被重新加载到预分频器影子寄存器中

为了避免在向预装载寄存器中写入新值时更新影子寄存器, 可以通过设置  $\text{TIMx\_CTRL1.UPDIS}=1$  来禁止更新事件。

当产生一个更新事件时, 计数器仍将被清除, 预分频器计数器也将被设置为 0 (但预分频器值将保持不变)。

下图给出一些示例, 展示了向上计数模式计数器在不同分频因子下的动作。

图 10-3 当内部时钟分频因子 =  $2/N$  时, 向上计数的时序图

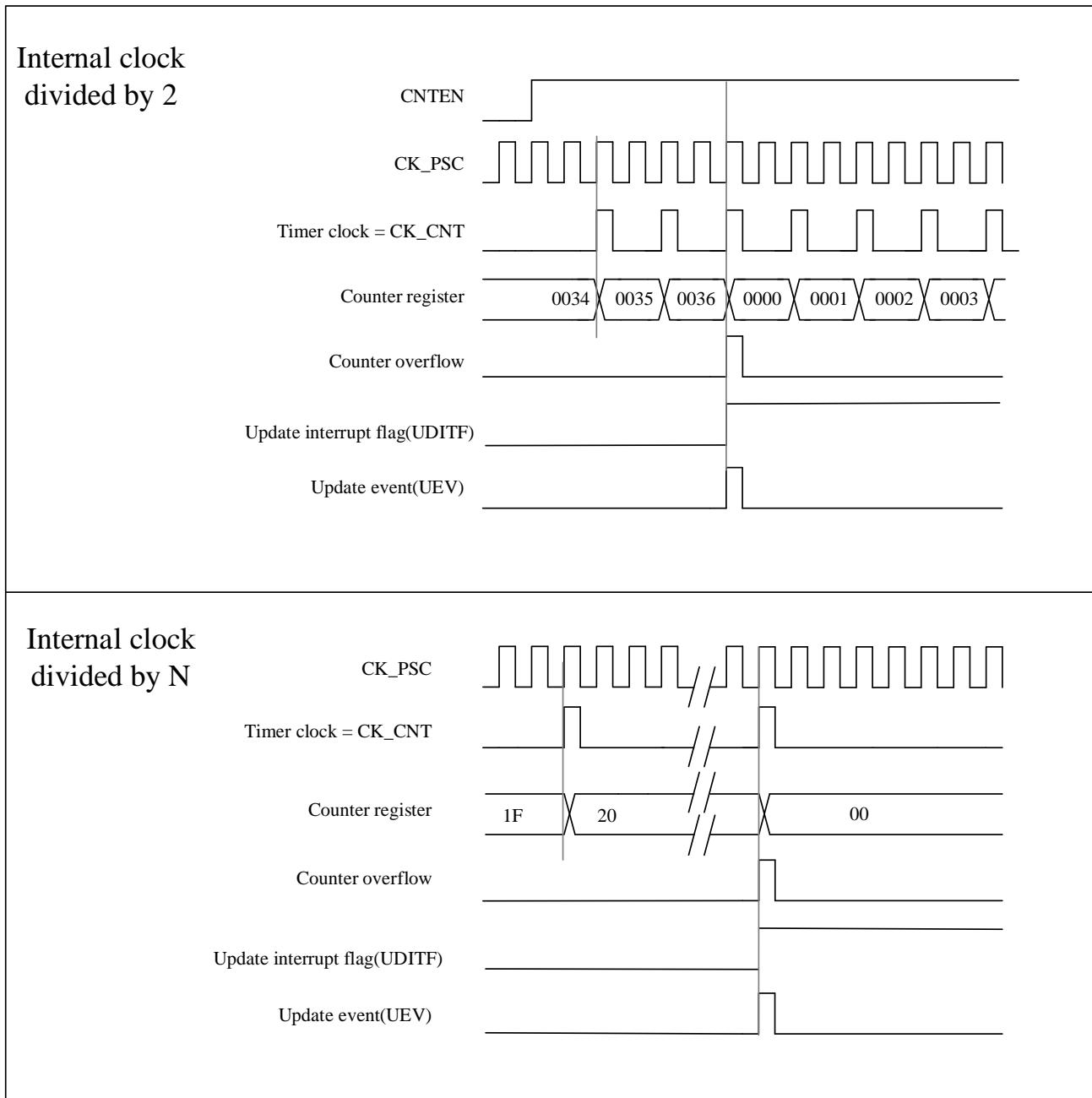
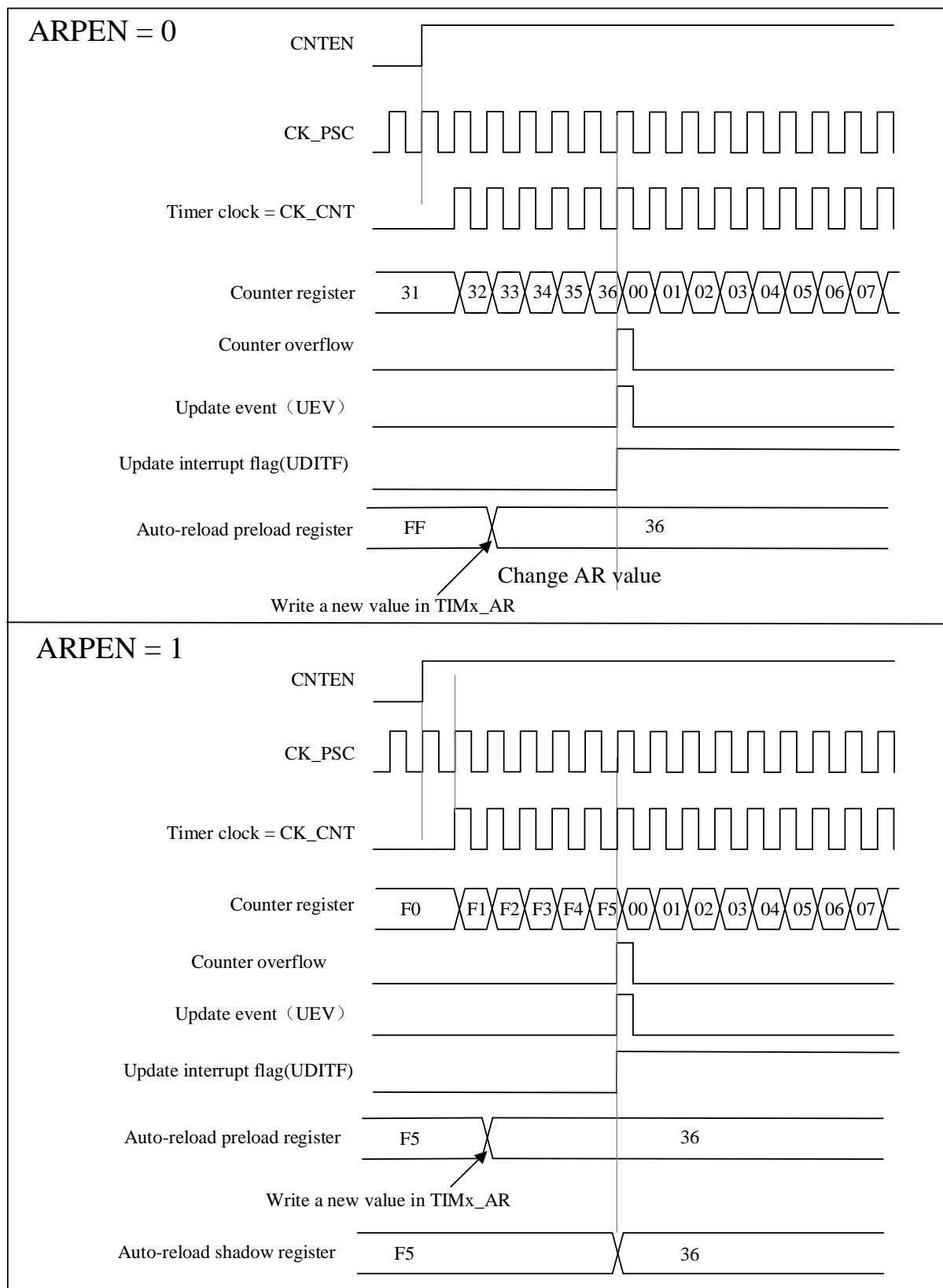


图 10-4 当 ARPEN=0/1 产生更新事件时，向上计数的时序图



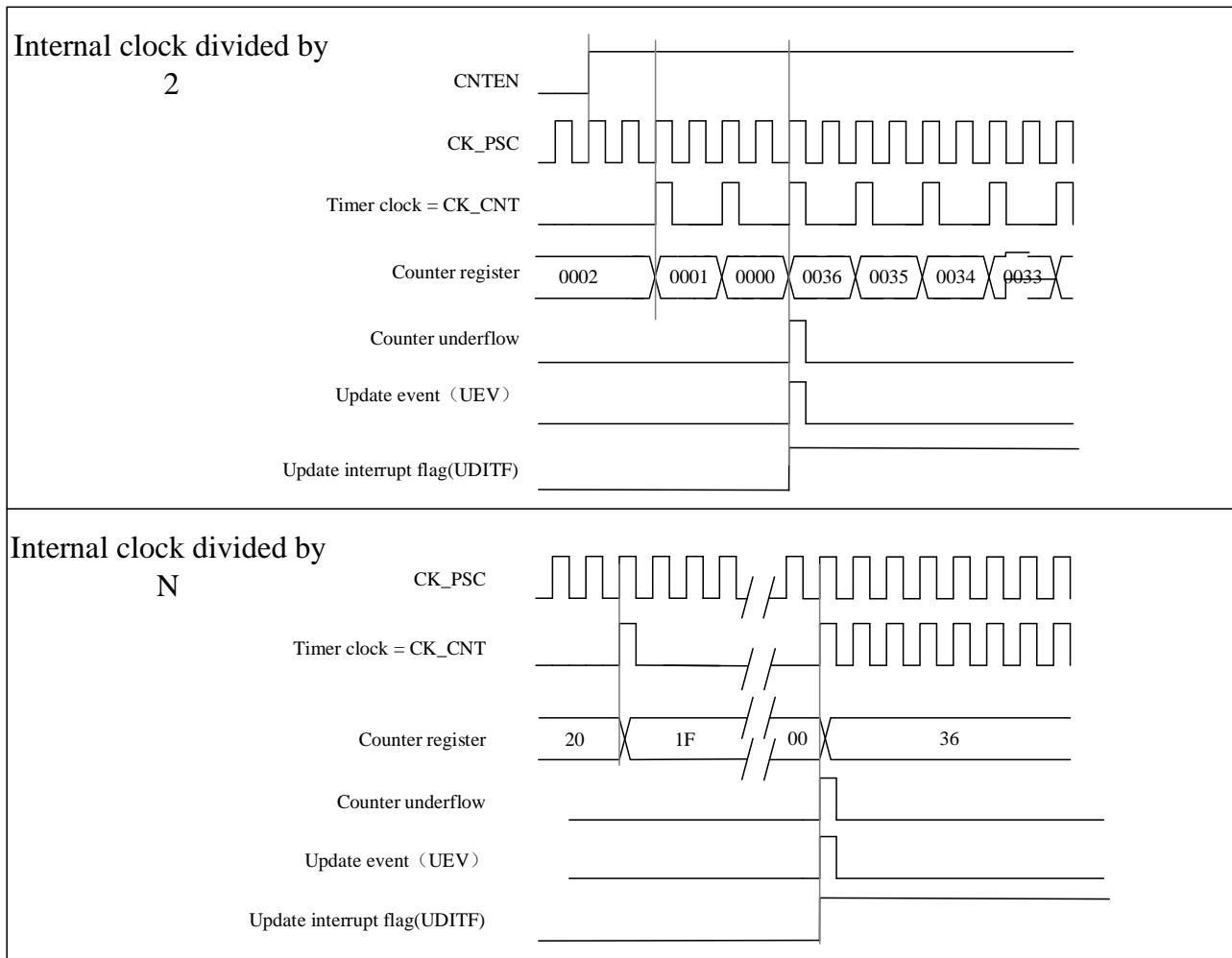
### 10.3.2.2 向下计数模式

向下计数模式，计数器将从寄存器 TIMx\_AR 的值减至 0，然后从自动重装载值重新开始，并产生计数器向下溢出事件。

向下计数模式和向上计数模式配置更新事件和更新寄存器的过程相同，请查阅 10.3.2.1 章节。

下图给出一些示例，展示了向下计数模式计数器在不同分频因子下的动作。

图 10-5 内部时钟分频因子 = 2/N 时，向下计数时序图



### 10.3.2.3 中央对齐模式

#### 10.3.2.3.1 中央对齐对称模式

在中央对齐对称模式下，计数器从 0 增加到值 (TIMx\_AR) -1，产生计数器溢出事件。然后，它从自动重装载值 (TIMx\_AR) 向下计数到 1，并生成一个计数器向下溢出事件。然后计数器重置为 0 并再次开始计数。

在这种模式下，TIMx\_CTRL1.DIR 方向位无效，由硬件更新和指定当前计数方向。当 TIMx\_CTRL1.CAMSEL 位不等于“00”时，中央对齐模式有效。

每次计数上溢和计数下溢时都会生成更新事件。或者，也可以通过设置 TIMx\_EVTGEN.UDGN 位（通过软件或使用从模式控制器）来生成更新事件。在这种情况下，计数器从 0 重新开始计数，预分频器的计数器也

从 0 重新开始计数。

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新载入之前被更新。

图 10-6 内部时钟分频因子 = 2/N，中央对齐时序图

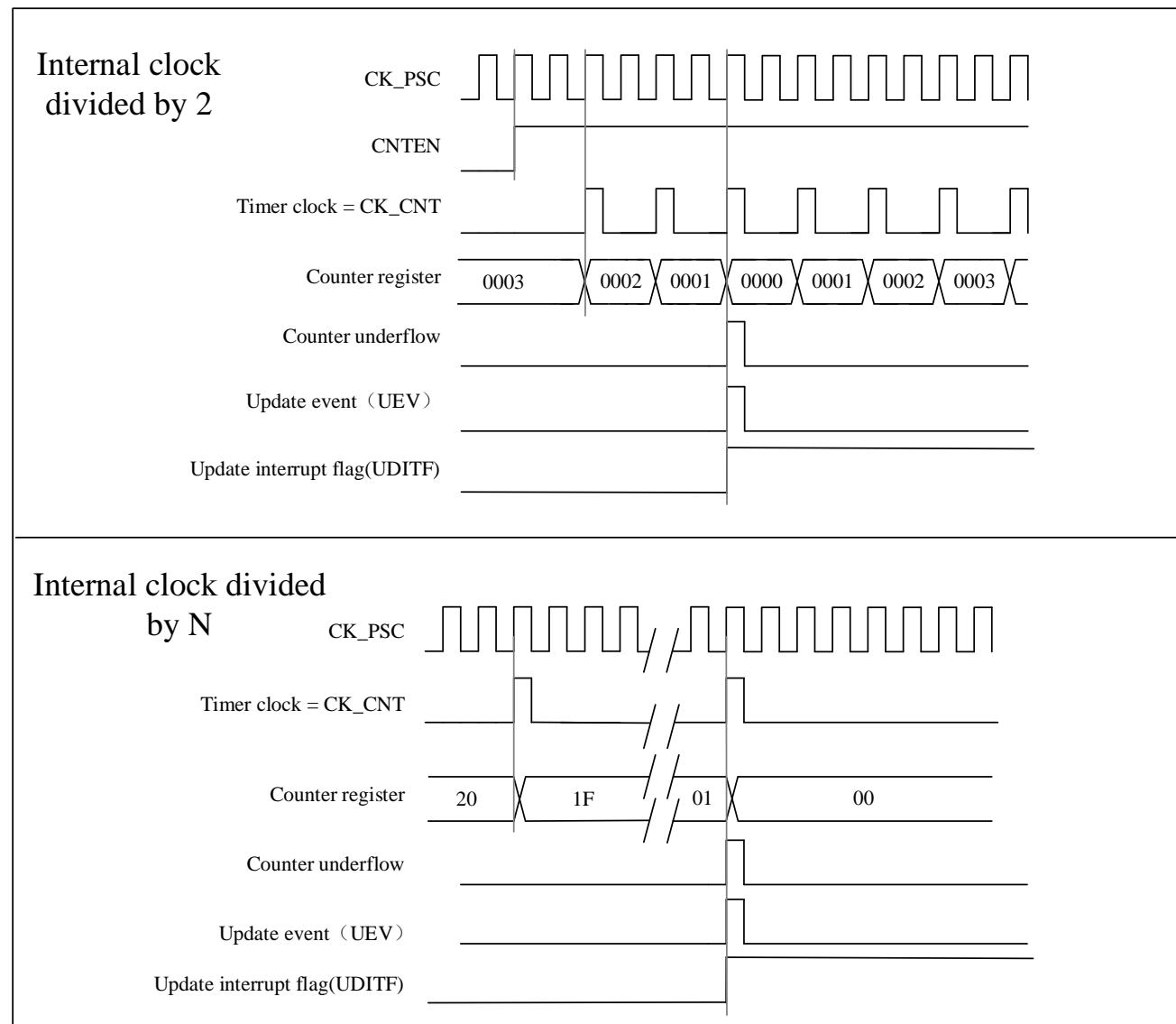
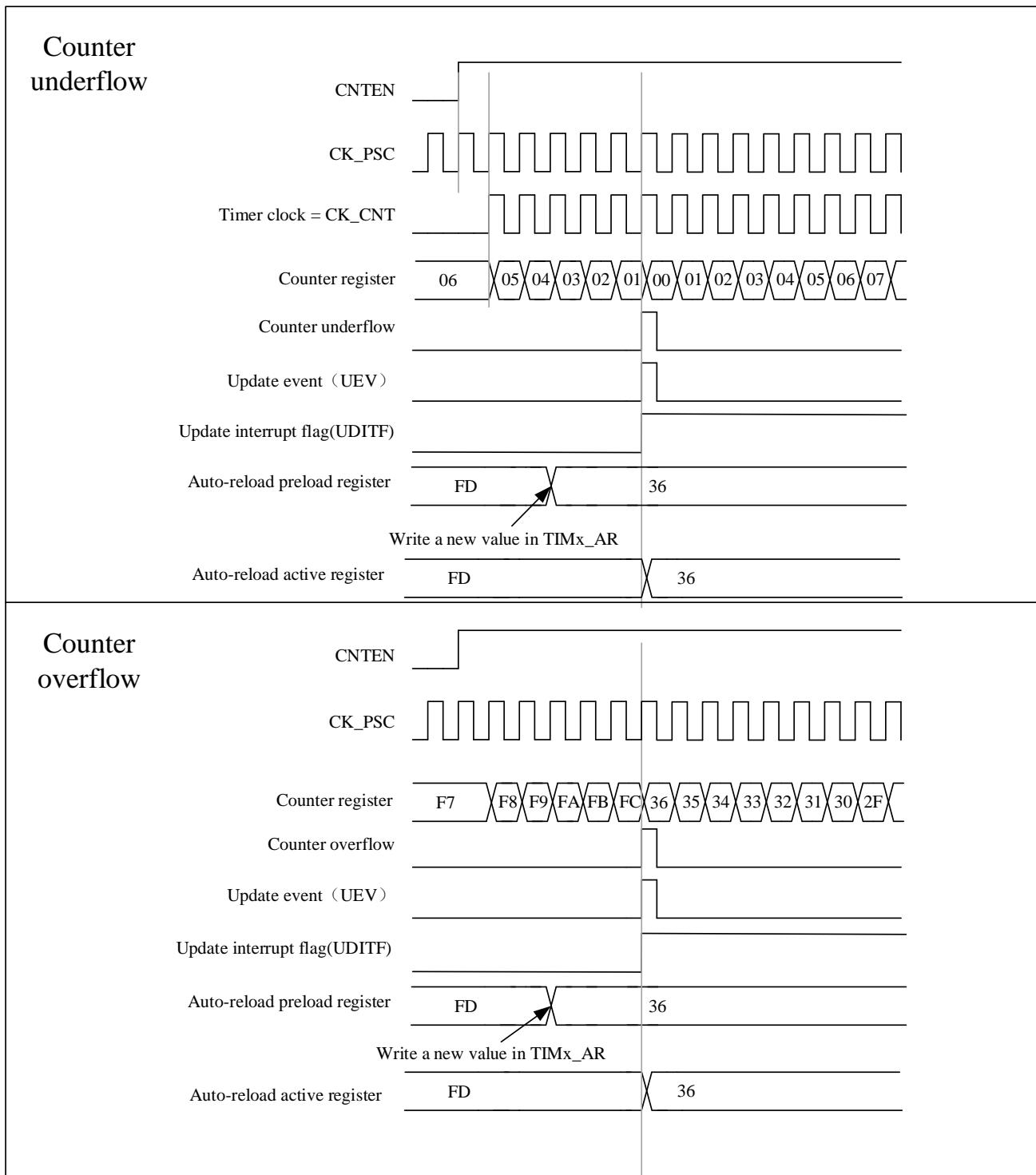


图 10-7 包含计数器上溢和下溢的中央对齐时序图(ARPEN=1)



### 10.3.3 时钟选择

■ 通用定时器的内部时钟: CK\_INT:

■ 两种外部时钟模式:

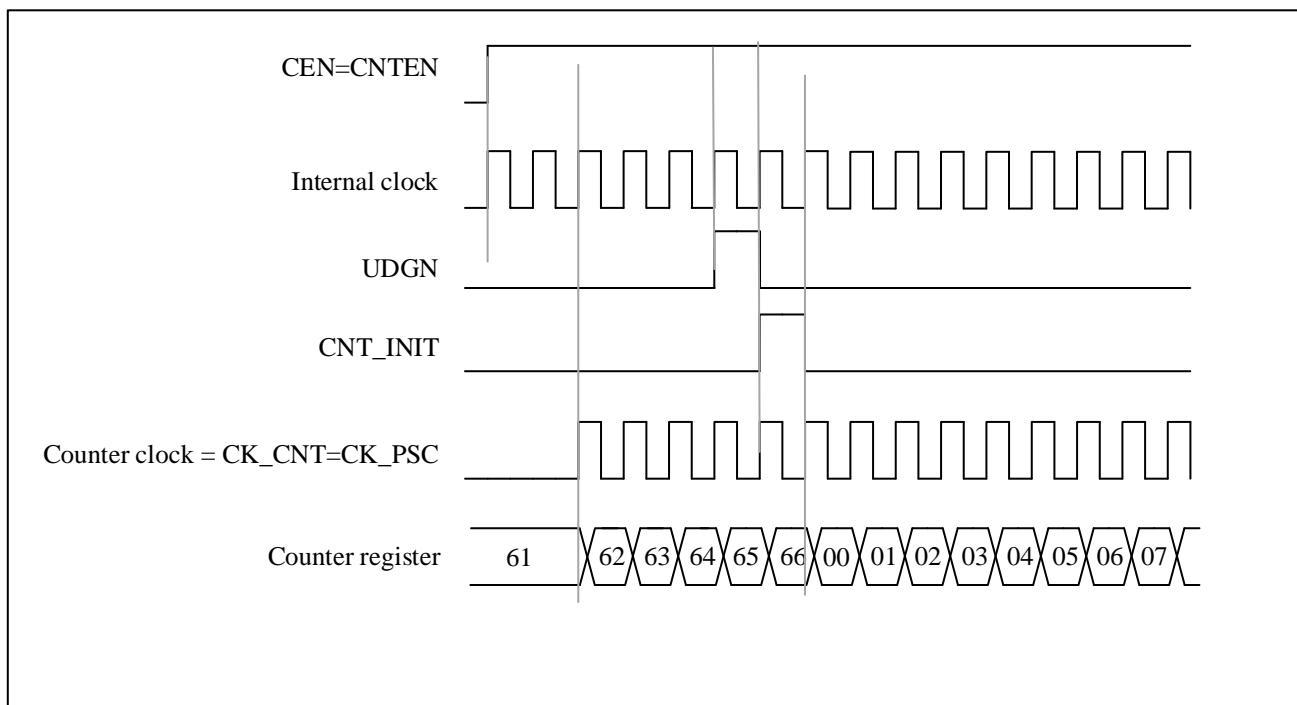
- 外部输入引脚

- 外部触发输入 ETR
- 内部触发输入 (ITRx): 一个定时器用作另一个定时器的预分频器

### 10.3.3.1 内部时钟源(CK\_INT)

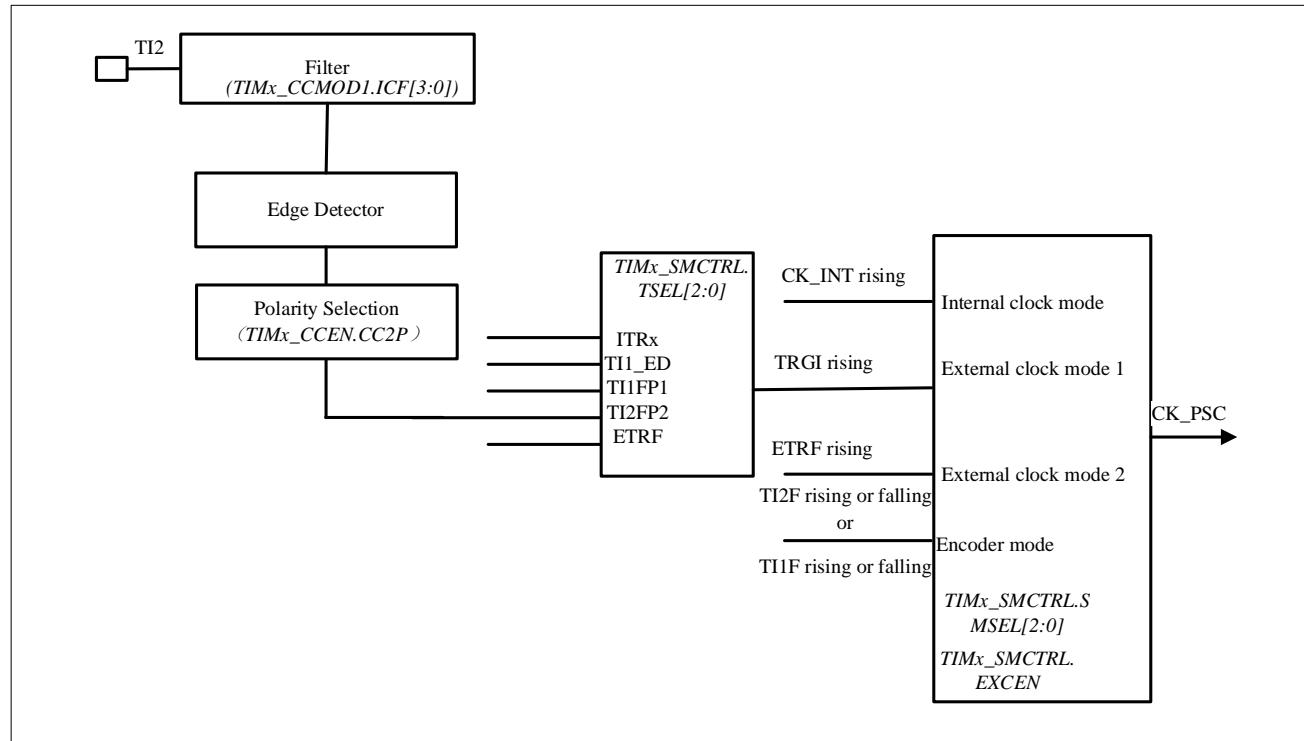
当 `TIMx_SMCTRL.SMSEL` 等于“000”时，从模式控制器被禁用。这三个控制位 (`TIMx_CTRL1.CNTEN`、`TIMx_CTRL1.DIR`、`TIMx_EVTGEN.UDGN`) 只能由软件改变 (`TIMx_EVTGEN.UDGN` 除外，它保持自动清零)。前提是 `TIMx_CTRL1.CNTEN` 位被软写为'1'，预分频器的时钟源由内部时钟 `CK_INT` 提供。

图 10-8 正常模式下的控制电路，内部时钟除以 1



### 10.3.3.2 外部时钟源模式 1

图 10-9 TI2 外部时钟连接示例



通过配置 `TIMx_SMCTRL.SMSEL=111` 选择该模式。计数器可以配置为在所选输入的时钟上升沿或下降沿进行计数。

例如，配置向上计数模式在 `TI2` 输入的时钟上升沿计数，配置步骤如下：

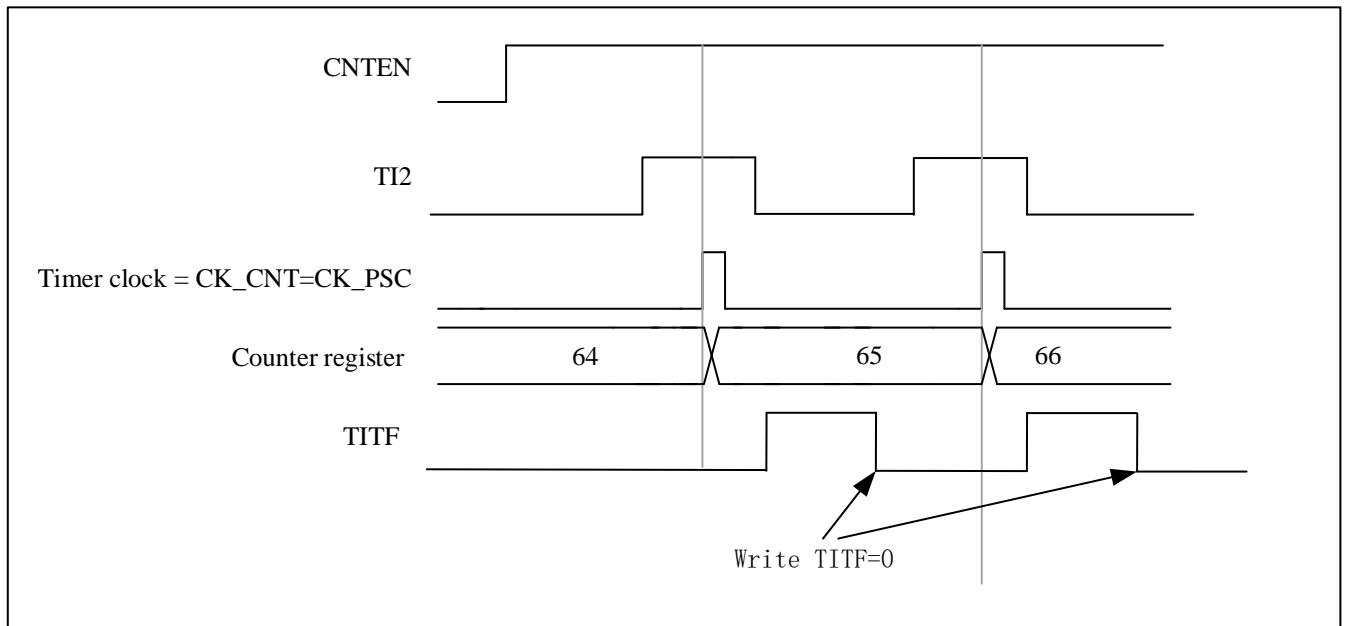
- 配置 `TIMx_CCMOD1.CC2SEL` 等于‘01’，CC2 通道配置为输入，IC2 映射到 `TI2`
- 配置 `TIMx_CCEN.CC2P` 等于‘0’，选择时钟上升沿极性
- 通过配置 `TIMx_CCMOD1.IC2F[3:0]` 选择输入滤波器带宽（如果不需滤波器，保持 IC2F 位为‘0000’）
- 配置 `TIMx_SMCTRL.SMSEL` 等于‘111’，选择定时器外部时钟模式 1
- 配置 `TIMx_SMCTRL.TSEL` 等于‘110’，选择 `TI2` 作为触发输入源
- 配置 `TIMx_CTRL1.CNTEN` 等于 ‘1’ 以启动计数器

**注意：捕获预分频器不用于触发，所以不需要配置**

当定时器时钟的上升沿出现在 `TI2=1` 时，计数器计数一次并且 `TIMx_STS .TITF` 标志被拉高。

`TI2` 的上升沿与计数器实际时钟之间的延迟取决于 `TI2` 输入端的再同步电路。

图 10-10 外部时钟模式 1 的控制电路

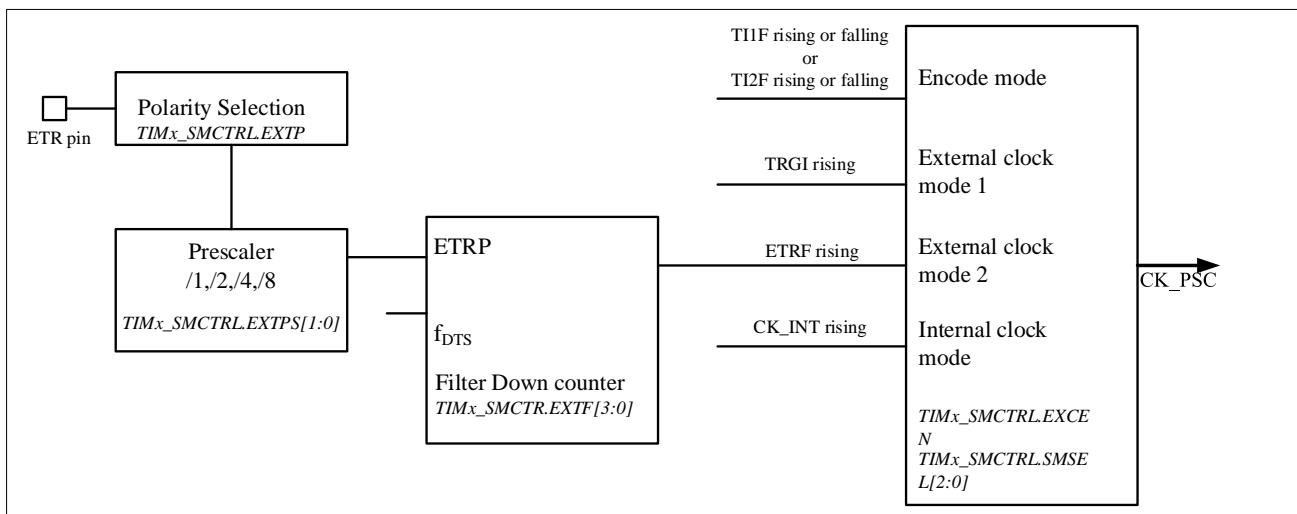


### 10.3.3.3 外部时钟源模式 2

此模式由 `TIMx_SMCTRL.EXCEN` 选择等于 1。计数器可以在外部触发输入 ETR 的每个上升沿或下降沿计数。

下图为外部时钟源模式 2 的外部触发输入模块示意图。

图 10-11 外部触发输入框图



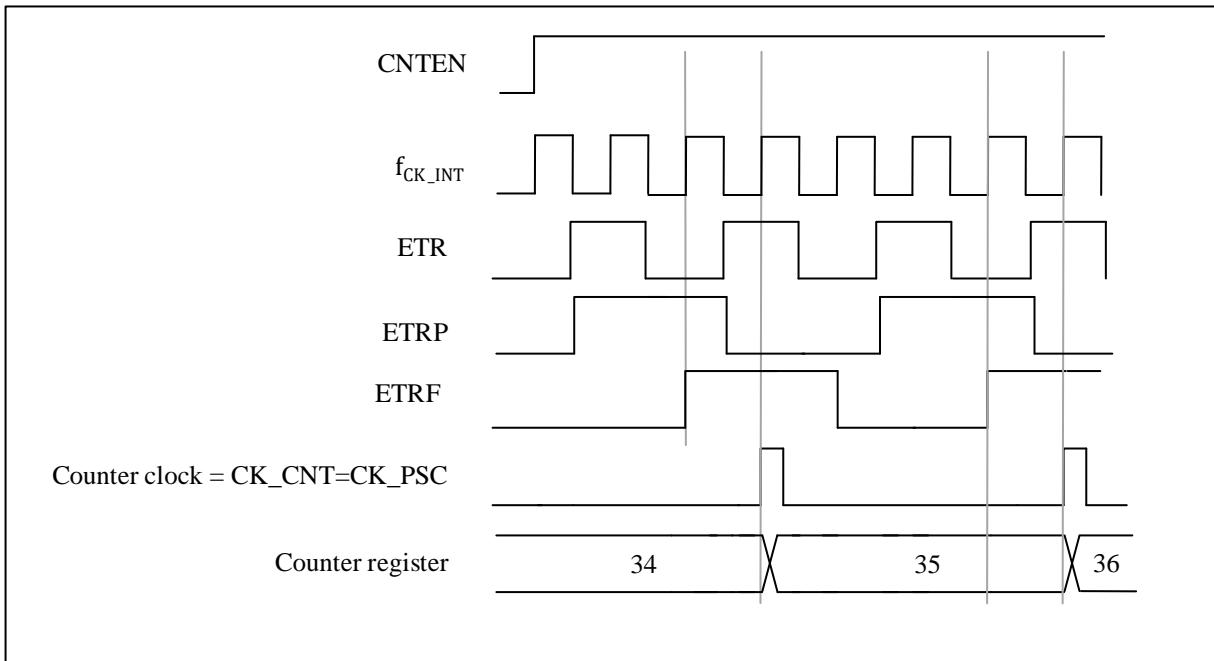
例如，使用以下配置步骤使向上计数器在 ETR 上每 2 个上升沿计数一次。

- 由于在这种情况下不需要过滤器，因此使 `TIMx_SMCTRL.EXTF[3:0]` 等于‘0000’
- 通过使 `TIMx_SMCTRL.EXTPS[1:0]` 等于 ‘01’ 来配置预分频器
- 通过设置 `TIMx_SMCTRL.EXTP` 等于‘0’来选择 ETR 引脚的极性，ETR 的上升沿有效

- 外部时钟模式 2 通过设置 `TIMx_SMCTRL.EXCEN` 等于‘1’来选择
- 通过设置 `TIMx_CTRL1.CNTEN` 等于“1”启动计数器。

计数器每 2 个 ETR 上升沿计数一次。ETR 的上升沿与计数器的实际时钟之间的延迟是由于 ETRP 信号上的再同步电路造成的。

图 10-12 外部时钟模式 2 的控制电路

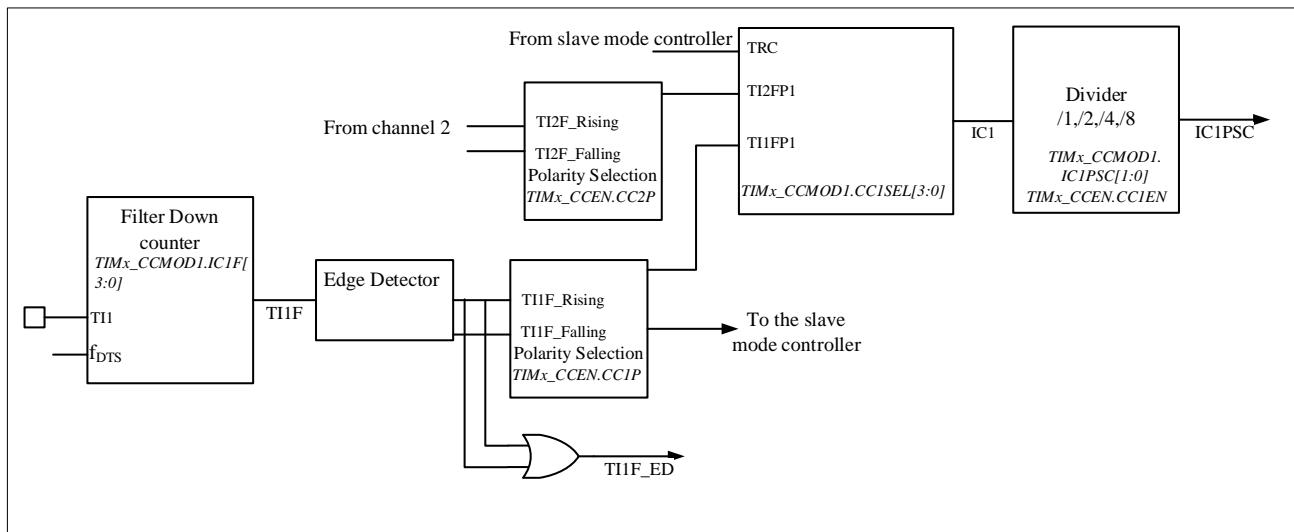


### 10.3.4 捕获/比较通道

捕获/比较通道包括捕获/比较寄存器和影子寄存器。输入部分由数字滤波器、多路复用器和预分频器组成。输出部分包括比较器和输出控制。

输入信号 `TIx` 被采样和滤波以产生信号 `TIxF`。然后由极性选择功能的边沿检测器生成信号 (`TIxF_rising` 或 `TIxF_falling`)，其极性由 `TIMx_CCEN.CCxP` 位选择。该信号可用作从模式控制器的触发输入。同时，信号 `ICx` 经过分频后送入捕获寄存器。下图显示了捕获/比较通道的框图。

图 10-13 捕获/比较通道（例如：通道 1 输入级）



输出部分生成一个中间波形 OCxRef (高电平有效) 作为参考。极性作用在链的末端。

图 10-14 捕获/比较通道 1 主电路

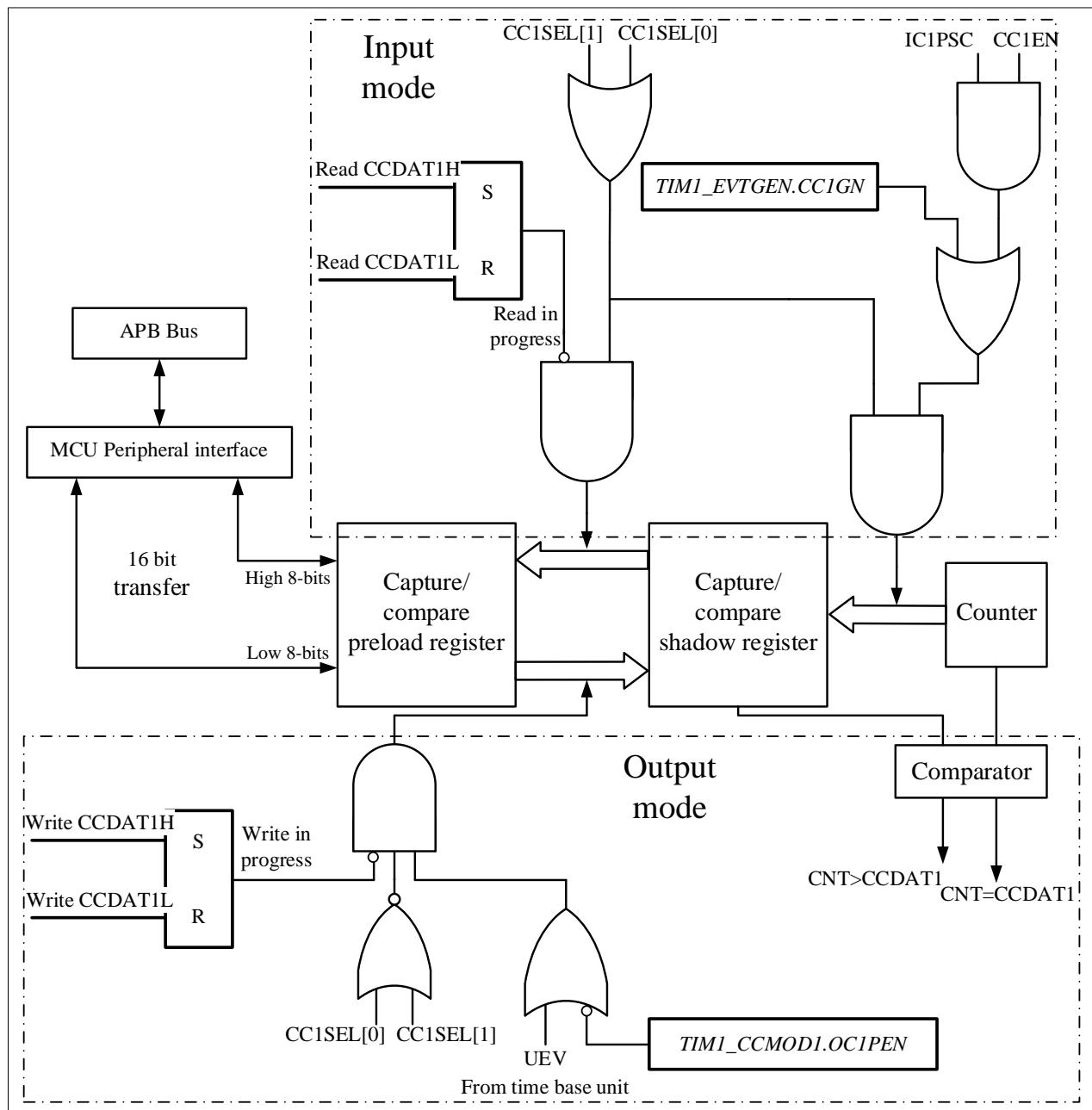
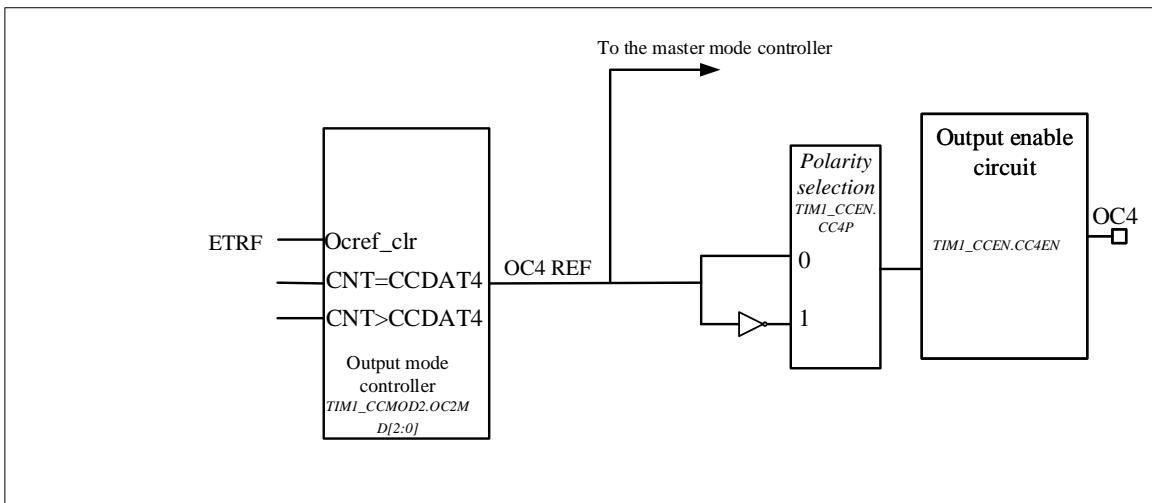


图 10-15 通道 x 的输出部分 (x=1/2/3/4, 以通道 4 为例子)



在捕获/比较时，读取和写入始终访问预加载的寄存器。两个具体工作流程如下：

在捕获模式下，捕获实际上是在影子寄存器中完成的，然后将影子寄存器中的值复制到预加载寄存器中。

在比较模式下，与捕获模式相反，预加载寄存器的值被复制到影子寄存器中，并与计数器进行比较。

### 10.3.5 输入捕获模式

在捕获模式下， $\text{TIM}_x\_\text{CCDAT}_x$  寄存器用于在检测到  $\text{IC}_x$  信号后锁存计数器值。

有一个捕获中断标志  $\text{TIM}_x\_\text{STS.CC}x\text{ITF}$ ，如果相应的中断使能被拉高，它可以发出中断或 DMA 请求。

$\text{TIM}_x\_\text{STS.CC}x\text{ITF}$  位在发生捕获事件时由硬件设置，并由软件或读取  $\text{TIM}_x\_\text{CCDAT}_x$  寄存器清零。

当  $\text{TIM}_x\_\text{CCDAT}_x$  寄存器中的计数器值被捕获并且  $\text{TIM}_x\_\text{STS.CC}1\text{ITF}$  已被拉高时，重复捕获标志  $\text{TIM}_x\_\text{STS.CC}x\text{OCF}$  设置为 1。与前者不同， $\text{TIM}_x\_\text{STS.CC}x\text{OCF}$  通过向其写入 0 来清除。

为实现  $\text{TI}1$  输入的上升沿将计数器值捕获到  $\text{TIM}_x\_\text{CCDAT}1$  寄存器中，配置流程如下：

- 选择有效输入：

将  $\text{TIM}_x\_\text{CCMOD}1.\text{CC}1\text{SEL}$  配置为“01”。此时输入为 CC1 通道， $\text{IC}1$  映射到  $\text{TI}1$ 。

- 编程所需的输入滤波器持续时间：

通过配置  $\text{TIM}_x\_\text{CCMOD}x.\text{IC}xF$  位来定义  $\text{TI}1$  输入的采样频率和数字滤波器的长度。示例：如果输入信号抖动多达 5 个内部时钟周期，我们必须选择比这 5 个时钟周期更长的滤波器持续时间。当检测到具有新电平的 8 个连续样本（以  $f_{\text{DTS}}$  频率采样）时，我们可以验证  $\text{TI}1$  上的转换。然后配置  $\text{TIM}_x\_\text{CCMOD}1.\text{IC}1F$  到“0011”

- 通过配置  $\text{TIM}_x\_\text{CCEN .CC}1\text{P}=0$ ，选择上升沿作为  $\text{TI}1$  通道的有效跳变极性

- 配置输入预分频器。在本例中，配置  $\text{TIM}_x\_\text{CCMOD}1.\text{IC}1\text{PSC}='00'$  以禁用预分频器，因为我们想要捕获每个有效转换

- 通过配置  $\text{TIM}_x\_\text{CCEN .CC}1\text{EN}='1'$  启用捕获。

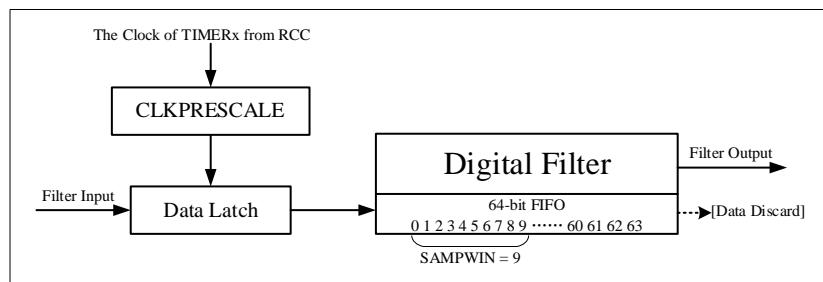
如果要使能 DMA 请求，可以配置  $\text{TIM}_x\_\text{DINTEN.CC}1\text{DEN}=1$ 。如果要使能相关中断请求，可以配置

TIMx\_DINTEN.CC1IEN = 1。

### 10.3.5.1 通道输入滤波

寄存器 TIMx\_CxFILT(x = 1, 2, 3, 4) 描述如下：

图 10-16 滑动滤波



- 数字滤波器通过 RCC 的 TIMx 时钟采样通道输入信号，在 64 位 FIFO 中累积采样。仅在 TIMx\_CxFILT.WSIZE [5:0] 中定义的窗口大小内采样数据，最大大小为 64。
- 过滤器输出采样窗口内的多数值，该值由 TIMx\_CxFILT.THRESH [5:0] 中的阈值定义，最大阈值为 63。此值应等于或大于窗口大小的一半。如果采样窗口内的逻辑 1 和逻辑 0 计数均不大于阈值，则数字滤波器保持先前的输出值。
- TIMx\_CxFILT.PSC 寄存器决定相应数字滤波器的采样率。过滤器 FIFO 在每个采样时钟从输入中捕获一个采样值。
- 如果数字滤波器关闭，滤波器输入将像电线一样绕过输出。

### 10.3.6 PWM 输入模式

PWM 输入模式和普通输入捕获模式有一些区别，包括：

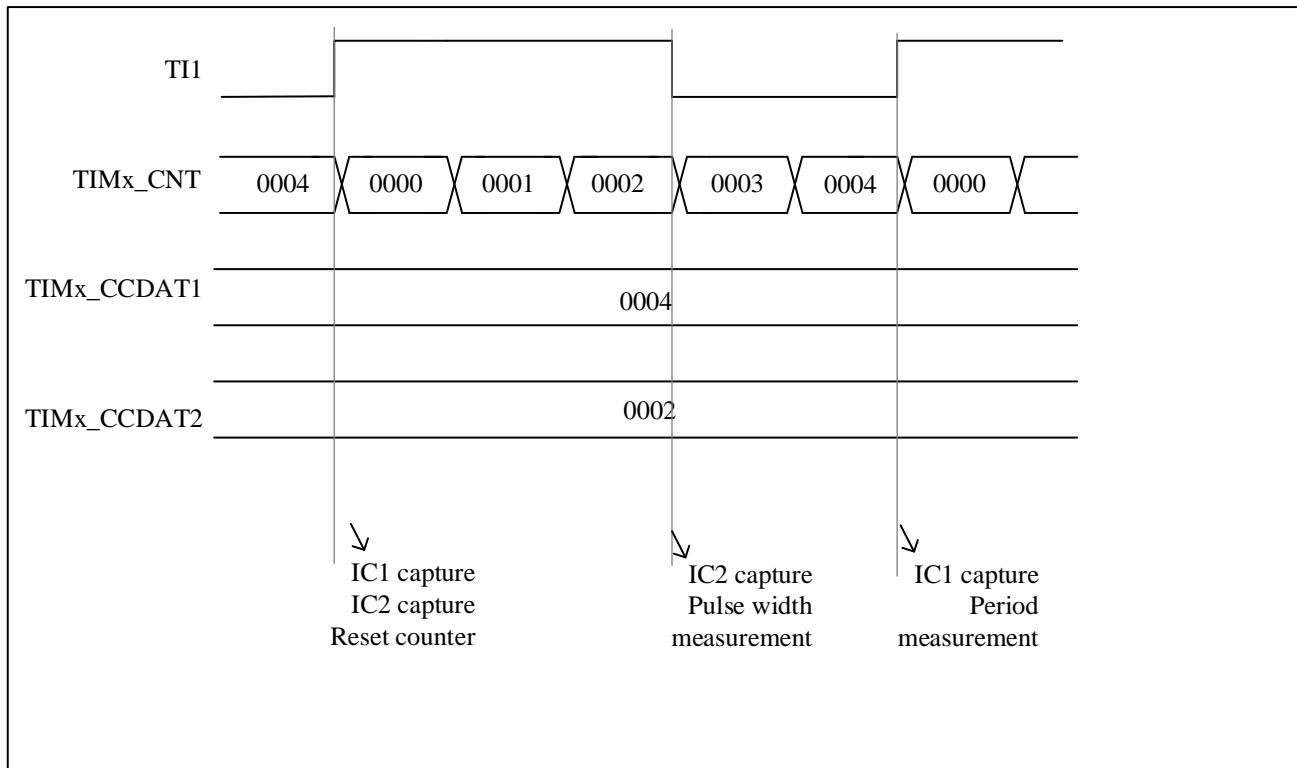
- 两个 IC<sub>x</sub> 信号映射到同一个 TI<sub>x</sub> 输入
- 两个 IC<sub>x</sub> 信号在极性相反的边沿有效
- 选择两个 TI<sub>x</sub>FP 信号之一作为触发输入
- 从机模式控制器配置为复位模式

例如，下面的配置流程可以用来知道 TI1 上 PWM 信号的周期和占空比（这取决于 CK\_INT 的频率和预分频器的值）。

- 配置 TIMx\_CCMOD1.CC1SEL 等于 ‘01’ 以选择 TI1 作为 TIMx\_CCDAT1 的有效输入
- 配置 TIMx\_CCEN.CC1P 等于 ‘0’ 选择滤波定时器输入 1(TI1FP1) 的有效极性，在上升沿有效
- 配置 TIMx\_CCMOD1.CC2SEL 等于 ‘10’ 选择 TI1 作为 TIMx\_CCDAT2 的有效输入
- 配置 TIMx\_CCEN.CC2P 等于 1 选择滤波定时器输入 2(TI1FP2) 的有效极性，下降沿有效
- 配置 TIMx\_SMCTRL.TSEL=101 选择 Filtered timer input 1 (TI1FP1) 作为有效触发输入
- 配置 TIMx\_SMCTRL.SMSEL=100 配置从模式控制器为复位模式

- 配置 `TIMx_CCEN.CC1EN=1` 和 `TIMx_CCEN.CC2EN=1` 以启用捕获

图 10-17 PWM 输入模式时序



由于只有滤波器定时器输入 1(TI1FP1) 和滤波器定时器输入 2(TI2FP2) 连接到从模式控制器，因此 PWM 输入模式只能与 `TIMx_CH1/TIMx_CH2` 信号一起使用。

### 10.3.7 强制输出模式

在输出模式 (`TIMx_CCMODx.CCxSEL=00`) 下，软件可以直接将输出比较信号强制为有效或无效电平。

用户可以设置 `TIMx_CCMODx.OCxMD=101` 强制输出比较信号为有效电平。 `OCxREF` 将被强制为高电平，`OCx` 得到与 `CCxP` 极性位相反的值。另一方面，用户可以设置 `TIMx_CCMODx.OCxMD=100` 强制输出比较信号为无效电平，即 `OCxREF` 被强制为低电平。

在此模式下，`TIMx_CCDATx` 影子寄存器和计数器的值仍然相互比较。

输出比较寄存器 `TIMx_CCDATx` 和计数器 `TIMx_CNT` 之间的比较对 `OCxREF` 没有影响。并且仍然可以设置标志。因此，仍然可以发送中断和 DMA 请求。

### 10.3.8 输出比较模式

用户可以使用此模式来控制输出波形，或指示一段时间已过。

当捕获/比较寄存器和计数器的值相同时，输出比较函数的操作如下：

- `TIMx_CCMODx.OCxMD` 为输出比较模式，`TIMx_CCEN.CCxP` 为输出极性。当比较匹配时，如果设置 `TIMx_CCMODx.OCxMD=000`，则输出管脚将保持其电平；如果设置 `TIMx_CCMODx.OCxMD=001`，则设置输出管脚有效；如果设置 `TIMx_CCMODx.OCxMD=010`，则输出管脚将为设置为无效；如果设

置 `TIMx_CCMODx.OCxMD=011`, 则输出引脚将设置为翻转。

- 设置 `TIMx_STS.CCxITF`
- 如果用户设置了 `TIMx_DINTEN.CCxIEN`, 将产生相应的中断
- 如果用户设置 `TIMx_DINTEN.CCxDEN` 并设置 `TIMx_CTRL2.CCDSEL` 选择 DMA 请求, 将发送 DMA 请求

用户可以设置 `TIMx_CCMODx.OCxPEN` 来选择是否使用捕获/比较预加载寄存器 (`TIMx_CCDATx`) 来选择捕获/比较影子寄存器。

时间分辨率是计数器的一个计数周期。

在单脉冲模式下, 输出比较模式也可用于输出单脉冲。

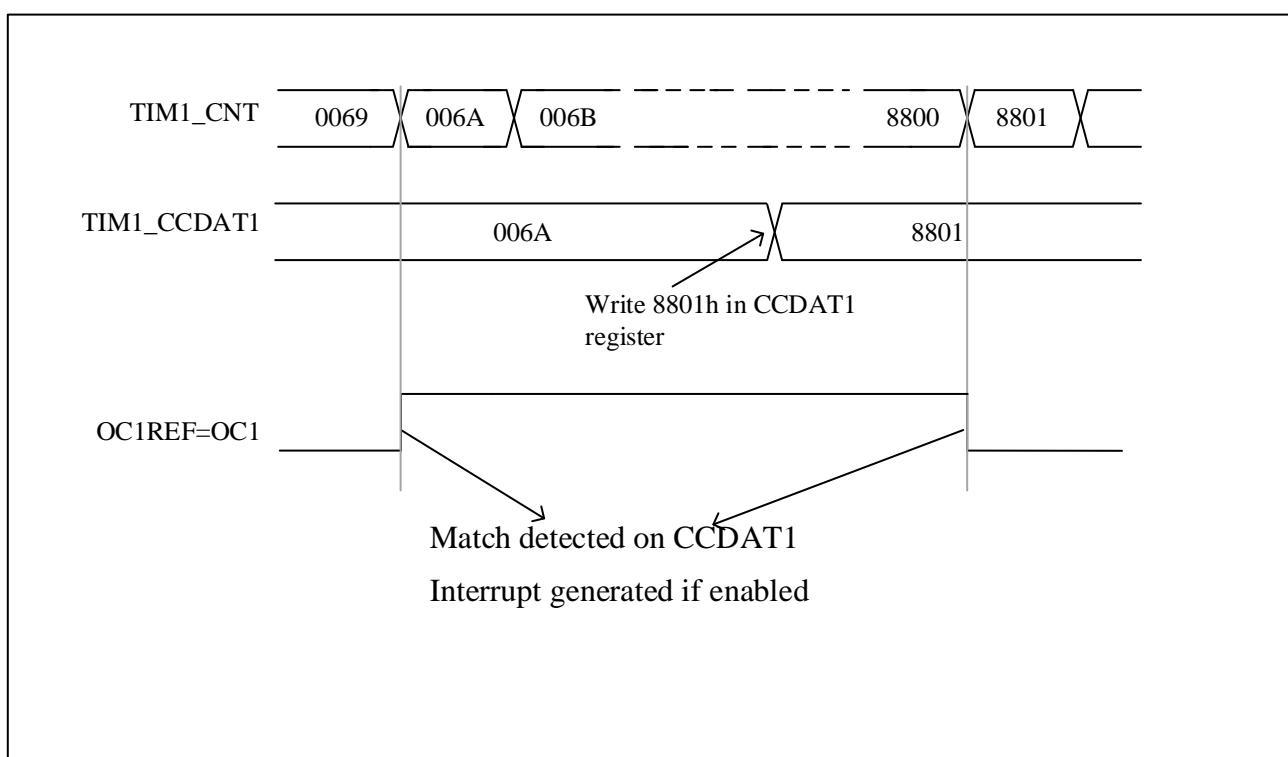
以下是输出比较模式的配置步骤:

- 首先, 用户应该选择计数器时钟
- 其次, 用所需数据设置 `TIMx_AR` 和 `TIMx_CCDATx`
- 如果用户需要产生中断, 设置 `TIMx_DINTEN.CCxIEN`
- 然后通过设置 `TIMx_CCEN.CCxP`、`TIMx_CCMODx.OCxMD`、`TIMx_CCEN.CCxEN` 等选择输出模式
- 最后, 设置 `TIMx_CTRL1.CNTEN` 启用计数器

用户可以随时通过设置 `TIMx_CCDATx` 来更新输出波形, 只要不启用预加载寄存器。否则, `TIMx_CCDATx` 影子寄存器将在下一次更新事件中更新。

例如:

图 10-18 输出比较模式, 开启 OC1



## 10.3.9 PWM 模式

用户可以使用 PWM 模式产生一个信号，其占空比由 TIMx\_CCDATx 寄存器的值决定，其频率由 TIMx\_AR 寄存器的值决定。并且取决于 TIMx\_CTRL1.CAMSEL 的值，TIM 可以在边沿对齐模式或中央对齐模式下产生 PWM 信号。

用户可以通过设置 TIMx\_CCMODx. OCxMD=110 或设置 TIMx\_CCMODx.OCxMD=111 来设置 PWM 模式 1 或 PWM 模式 2。要使能预加载寄存器，用户必须设置相应的 TIMx\_CCMODx.OCxPEN。然后设置 TIMx\_CTRL1.AR PEN 自动重加载预加载寄存器。

用户可以通过设置 TIMx\_CCEN.CCxCp 来设置 OCx 的极性。

当 TIM 处于 PWM 模式时，TIMx\_CNT 和 TIMx\_CCDATx 的值总是相互比较。

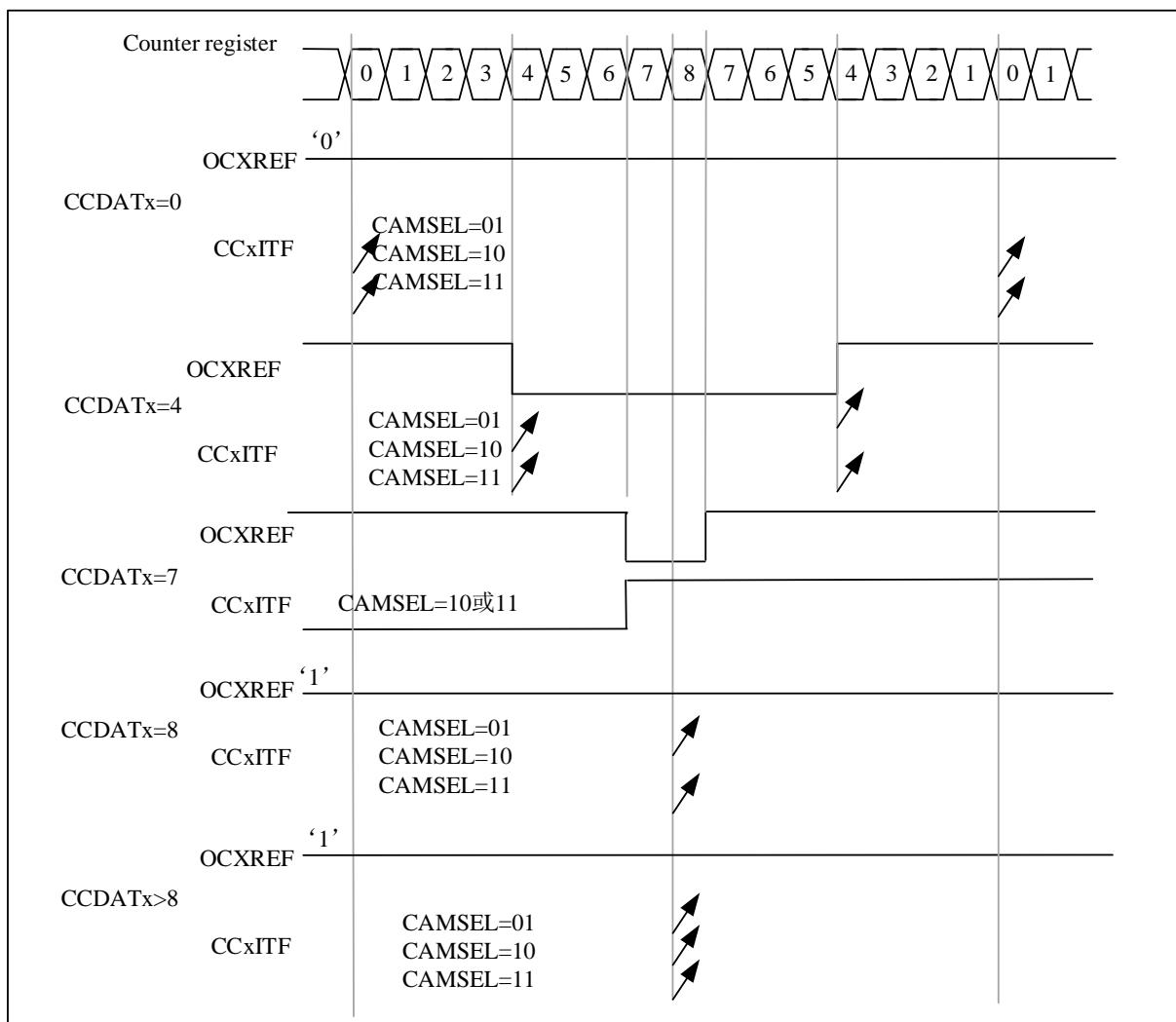
只有当更新事件发生时，预加载寄存器才会转移到影子寄存器。因此，用户必须在计数器开始计数之前通过设置 TIMx\_EVTGEN.UDGN 来复位所有寄存器。

### 10.3.9.1 PWM 中央对齐模式

如果用户设置 TIMx\_CTRL1.CAMSEL 等于 01、10 或 11，PWM 中央对齐模式将被激活。比较标志的设置取决于 TIMx\_CTRL1.CAMSEL 的值。设置比较标志的情况有 3 种，仅当计数器向上计数时，仅当计数器向下计数时，或当计数器向上计数和向下计数时。用户不应通过软件修改 TIMx\_CTRL1.DIR，它是由硬件更新的。

中央对齐 PWM 波形示例如下，波形设置为：TIMx\_AR=8，PWM 模式 1，当计数器向下计数对应 TIMx\_CTRL1.CAMSEL=01 时设置比较标志。

图 10-19 中央对齐的 PWM 波形 (AR=8)



使用中央对齐模式时用户应注意的事项如下：

- 计数器向上或向下计数取决于 TIMx\_CTRL1.DIR 的值。注意不要同时更改 DIR 和 CAMSEL 位
- 用户在中央对齐模式下不要写计数器，否则会导致意想不到的结果。例如：
  - ◆ 如果写入计数器的值为 0 或者是 TIMx\_AR 的值，则方向会被更新，但不会产生更新事件
  - ◆ 如果写入计数器的值大于自动重载的值，则方向不会更新
- 为了安全起见，建议用户在启动计数器之前设置 TIMx\_EVTGEN.UDGN 以通过软件生成更新，并且在计数器运行时不要写入计数器

### 10.3.9.2 PWM 边沿对齐模式

边沿对齐模式有两种配置，向上计数和向下计数。

- 向上计数

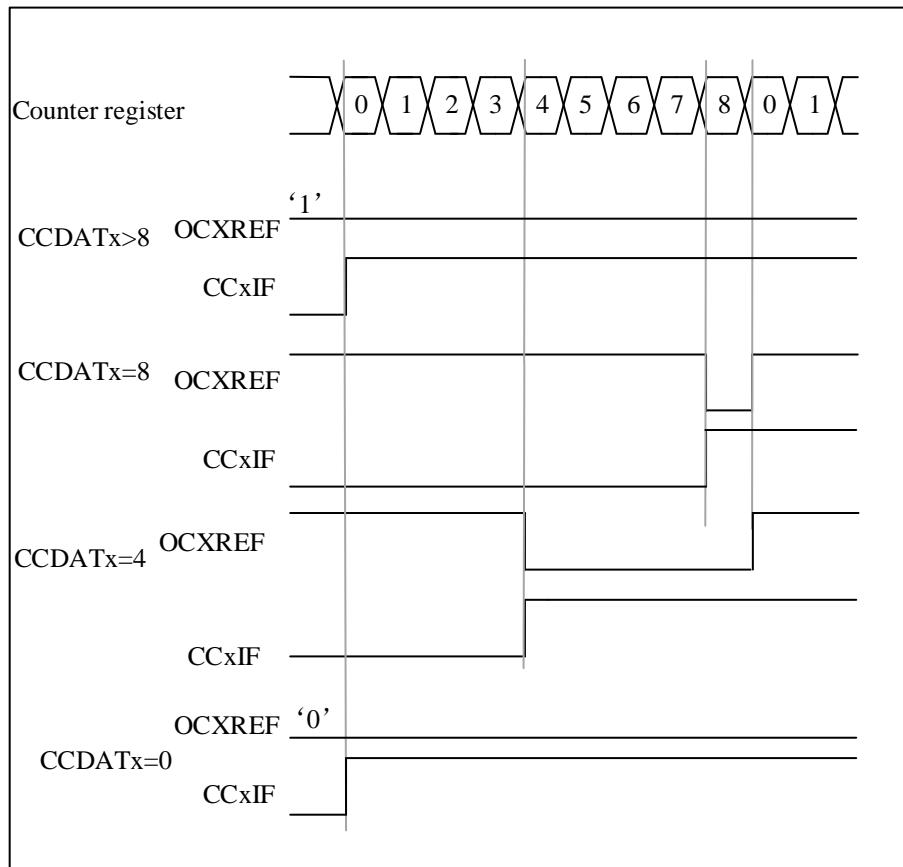
用户可以设置 TIMx\_CTRL1.DIR=0 使计数器向上计数。

PWM 模式 1 的示例：

当  $\text{TIMx\_CNT} < \text{TIMx\_CCDATx}$  时， $\text{OCxREF}$  为高电平，否则为低电平。如果  $\text{TIMx\_CCDATx}$  中的比较值大于自动重载值，则  $\text{OCxREF}$  将保持为 1。相反，如果比较值为 0，则  $\text{OCxREF}$  将保持为 0。

当  $\text{TIMx\_AR}=8$  时，PWM 波形如下：

图 10-20 边沿对齐 PWM 波形 (AR=8)



### ● 向下计数

用户可以设置  $\text{TIMx\_CTRL1.DIR}=1$  使计数器向下计数。

PWM 模式 1 的示例：

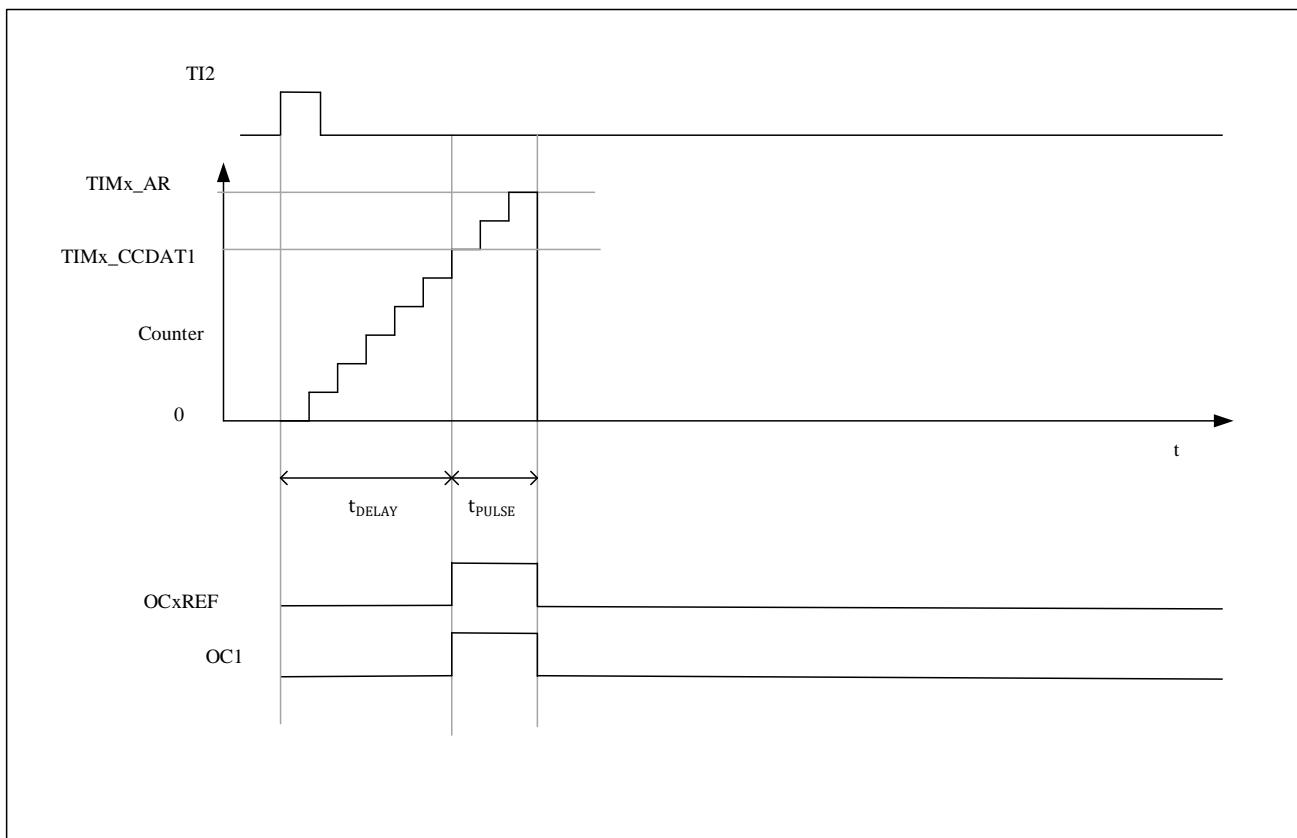
当  $\text{TIMx\_CNT} > \text{TIMx\_CCDATx}$  时， $\text{OCxREF}$  为低电平，否则为高电平。如果  $\text{TIMx\_CCDATx}$  中的比较值大于自动重载值，则  $\text{OCxREF}$  将保持为 1。

注：若第  $n$  个 PWM 周期  $\text{CCDATx}$  影子寄存器  $>= \text{AR}$  值，第  $(n+1)$  个 PWM 周期  $\text{CCDATx}$  的影子寄存器值是 0。第  $(n+1)$  个 PWM 周期的计数器为 0 的时刻，虽然计数器 =  $\text{CCDATx}$  影子寄存器的值 = 0， $\text{OCxREF} = '0'$ ，但不会产生比较事件。

### 10.3.10 单脉冲模式

在单脉冲模式(ONEPM)中，接收到触发信号，经过可控延迟  $t_{\text{DELAY}}$  后产生脉宽可控的脉冲  $t_{\text{PULSE}}$ 。输出模式需要配置为输出比较模式或 PWM 模式。选择单脉冲模式后，计数器会在更新事件 UEV 产生后停止计数。

图 10-21 单脉冲模式示例



以下是单脉冲模式的示例：

从 TI2 输入检测到上升沿触发，延迟  $t_{DELAY}$  后在 OC1 上产生宽度为  $t_{PULSE}$  的脉冲。

1. 计数器配置：向上计数，计数器  $TIMx\_CNT < TIMx\_CCDAT1 \leq TIMx\_AR$ ；
2. TI2FP2 映射到 TI2,  $TIMx\_CCMOD1.CC2SEL = '01'$ ; TI2FP2 配置为上升沿检测,  $TIMx\_CCEN.CC2P = '0'$ ;
3. TI2FP2 充当从模式控制器的触发器 (TRGI) 并启动计数器,  $TIMx\_SMCTRL.TSEL = '110'$ ,  $TIMx\_SMCTRL.SMSEL = '110'$  (触发模式);
4.  $TIMx\_CCDAT1$  写入要延迟的计数值 ( $t_{DELAY}$ ),  $TIMx\_AR - TIMx\_CCDAT1$  为脉宽  $t_{PULSE}$  的计数值;
5. 配置  $TIMx\_CTRL1.ONEPM=1$  使能单脉冲模式, 配置  $TIMx\_CCMOD1.OC1MD = '111'$  选择 PWM2 模式;
6. 等待 TI2 有外部触发事件, OC1 输出一个单脉冲波形;

#### 10.3.10.1 特殊情况：OCx 快速使能：

在单脉冲模式下，通过  $TIx$  输入检测到一个边沿，并触发计数器开始计数到比较值，然后输出一个脉冲。这些操作限制了可以达到的最小延迟  $t_{DELAY}$ 。

您可以设置  $TIMx\_CCMODx.OCxFEN=1$  开启 OCx 快速使能，在触发上升沿后，OCxREF 信号将被强制转换为与比较匹配立即发生的电平相同的电平，而不管比较结果如何。OCxFEN 快速使能仅在通道模式配置为 PWM1 和 PWM2 模式时生效。

### 10.3.11 在外部事件上清除 OCxREF 信号

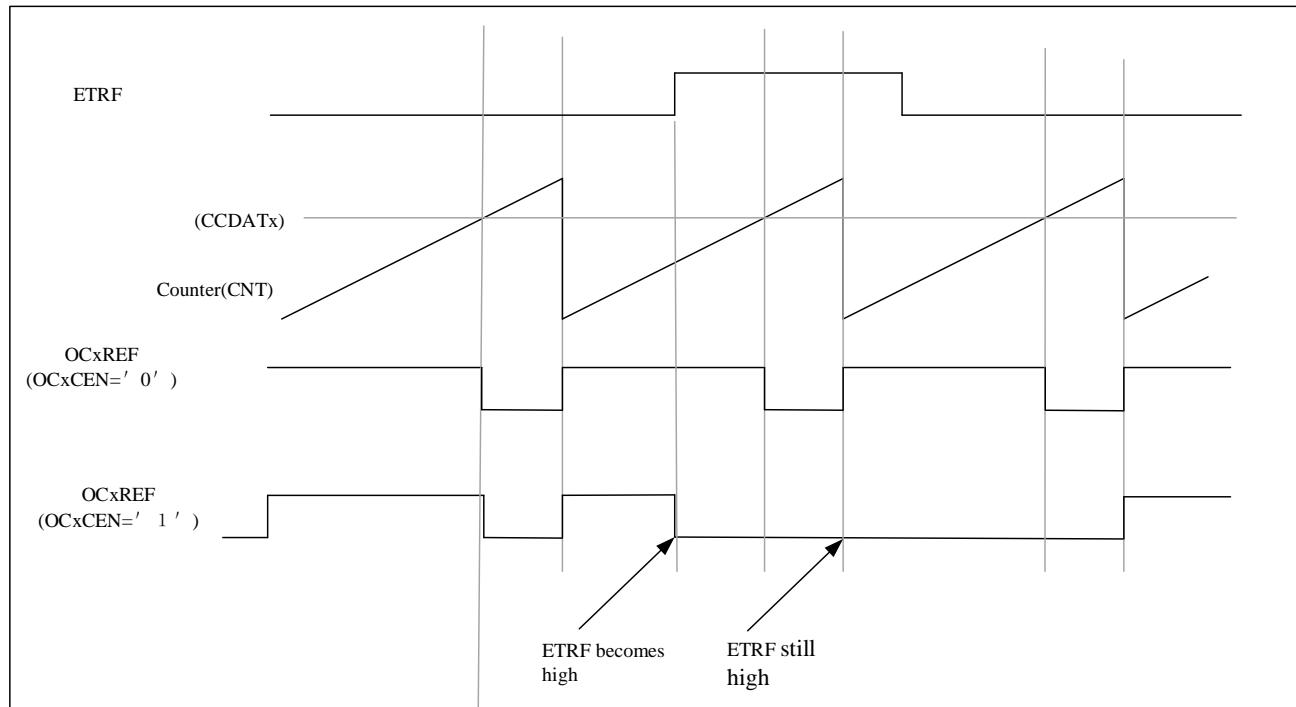
如果用户设置 `TIMx_CCMODx.OCxCEN=1`, `ETRF` 输入的高电平可用于驱动 `OCxREF` 信号为低电平, `OCxREF` 信号将保持低电平, 直到下一次 UEV 发生。只有输出比较和 PWM 模式可以使用该功能。在强制模式下不能使用。

例如: 为了控制电流, 用户可以将 `ETR` 信号连接到比较器的输出端, `ETR` 的操作如下:

- 设置 `TIMx_SMCTRL.EXTPS=00` 禁用外部触发预分频器。
- 设置 `TIMx_SMCTRL.EXCEN=0` 禁用外部时钟模式 2。
- 设置 `TIMx_SMCTRL.EXTP` 和 `TIMx_SMCTRL.EXTF`, 根据需要配置外触发极性和外触发滤波器。

例: 当 `ETRF` 输入变高时, `OCxREF` 信号对于不同的 `OCxCEN` 值的行为。在这种情况下, 定时器设置为 PWM 模式。

图 10-22 清除 TIMx 的 `OCxREF`



### 10.3.12 调试模式

当微控制器处于调试模式 (Cortex-M4 内核停止) 时, 根据 DBG 模块中的 `DBG_TIMx_STOP` 配置, TIMx 计数器可以继续正常工作或停止。详见 23.4.3 章节。

### 10.3.13 TIMx 定时器和外部触发的同步

TIMx 定时器可以通过从模式 (复位、触发和门控) 中的触发器进行同步。

### 10.3.13.1 从模式：复位模式

在复位模式下，触发事件可以复位计数器和预分频器。更新预加载寄存器 TIMx\_AR、TIMx\_CCDATx，并产生更新事件 UEV (TIMx\_CTRL1.UPRS=0)。

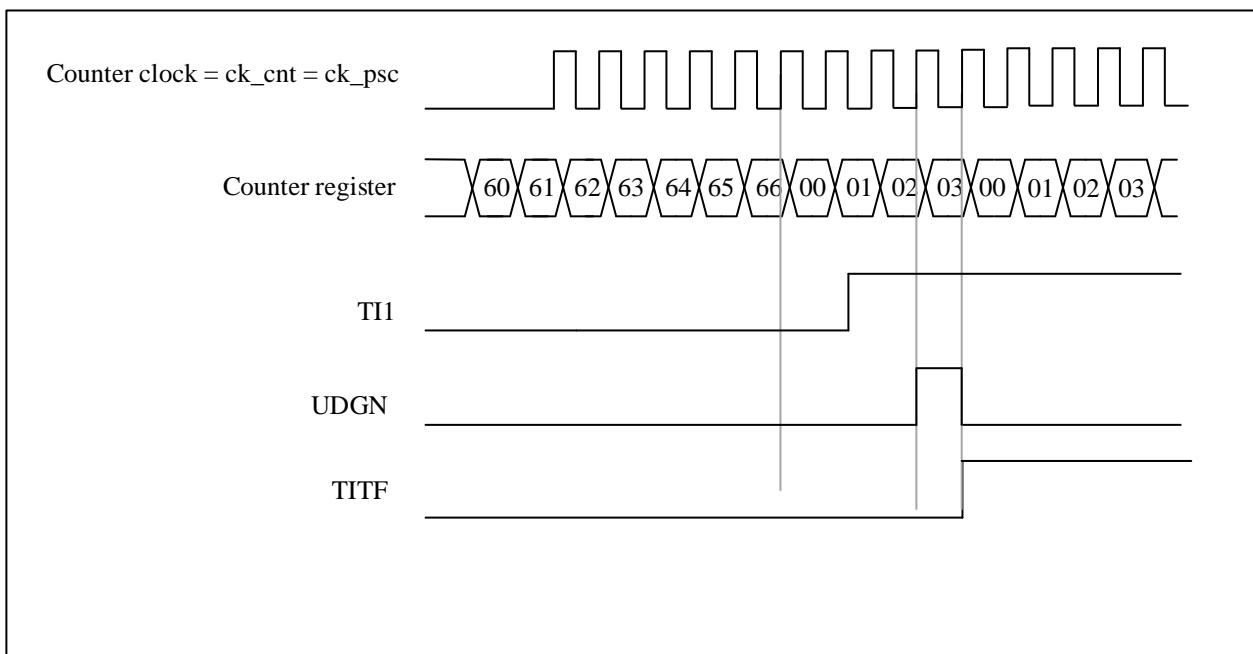
以下是复位模式的示例：

1. 通道 1 配置为输入检测 TI1 的上升沿 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=0);
2. 从模式选择为复位模式 (TIMx\_SMCTRL.SMSEL=100)，触发输入选择为 TI1 (TIMx\_SMCTRL.TSEL=101);
3. 启动定时器 (TIMx\_CTRL1.CNTEN = 1)

启动定时器后，当 TI1 检测到上升沿时，计数器复位并重新开始计数，并设置触发标志 (TIMx\_STS.TITF=1);

TI1 的上升沿与实际计数器复位之间的延迟是由于 T11 输入的重新同步电路引起的。

图 10-23 复位模式下的控制电路



### 10.3.13.2 从模式：触发模式

在触发模式下，输入端口的触发事件（上升沿/下降沿）可以触发计数器开始计数。

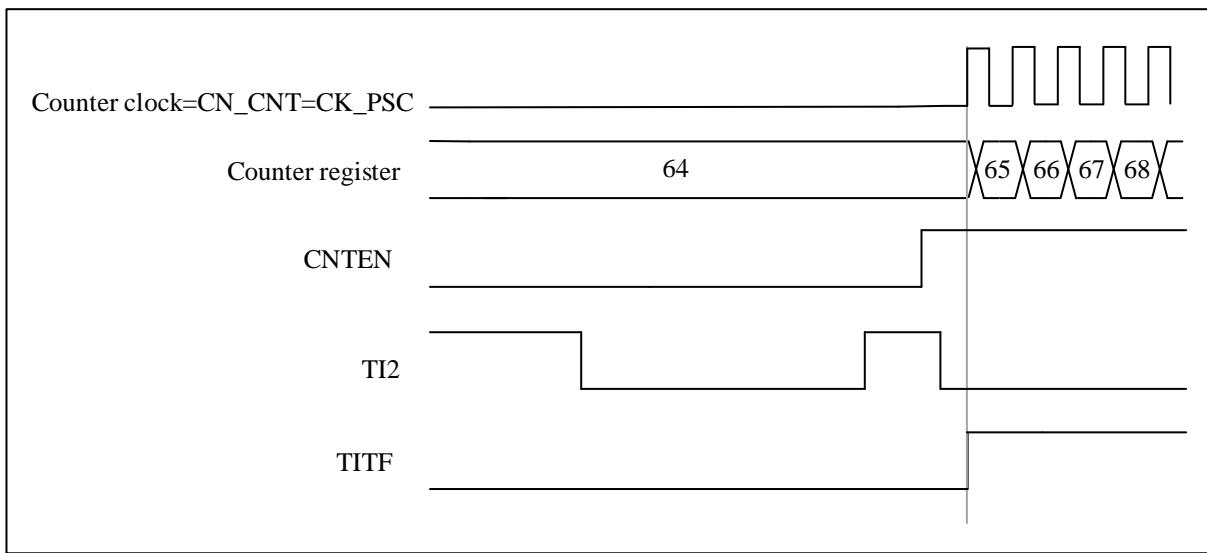
以下是触发模式的示例：

1. 通道 2 配置为输入，检测 TI2 的上升沿 (TIMx\_CCMOD1.CC2SEL=01, TIMx\_CCEN.CC2P=0);
2. 选择从模式为触发模式 (TIMx\_SMCTRL.SMSEL=110)，触发输入选择 TI2 (TIMx\_SMCTRL.TSEL=110);

当 TI2 检测到上升沿时，计数器开始计数，触发标志置位 (TIMx\_STS.TITF=1);

TI2 的上升沿与实际计数器启动之间的延迟是由于 T12 输入的重新同步电路引起的。

图 10-24 触发器模式下的控制电路



### 10.3.13.3 从模式：门控模式

在门控模式下，输入端口的电平极性可以控制计数器是否计数。

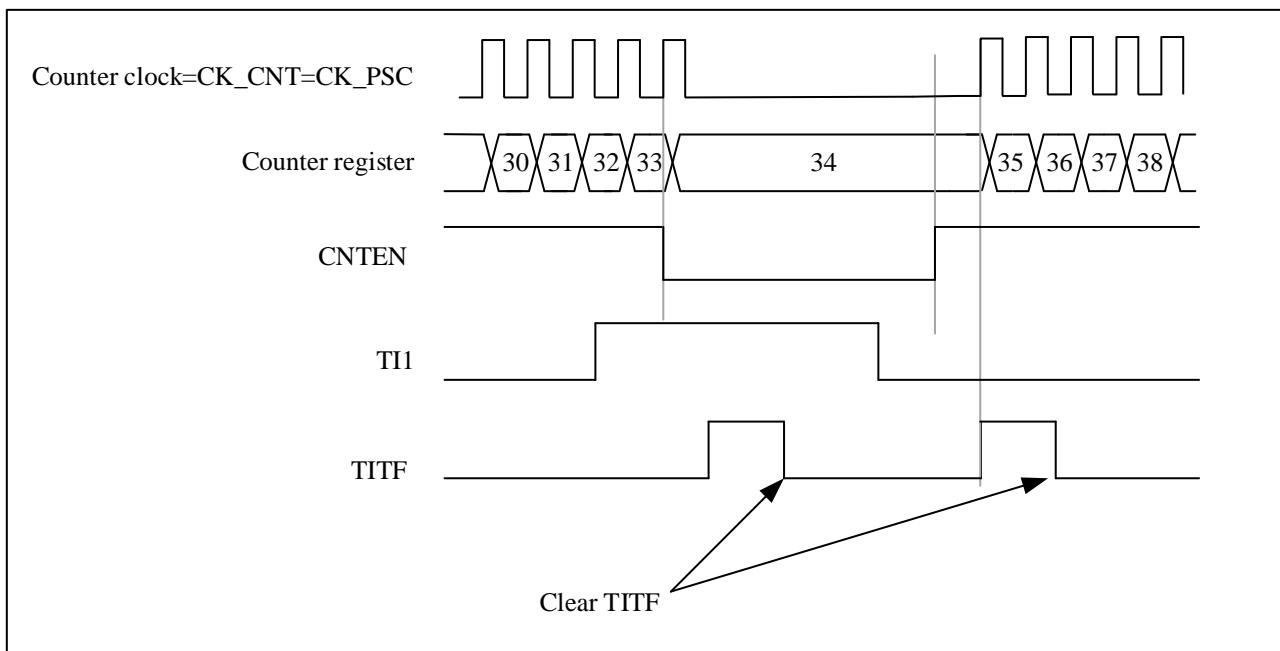
以下是门控模式的示例：

1. 通道 1 配置为 TI1 上的输入检测低电平有效 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=1);
2. 选择从模式为门控模式 (TIMx\_SMCTRL.SMSEL=101)，选择 TI1 作为 TRGI (TIMx\_SMCTRL.TSEL=101);
3. 启动定时器 (TIMx\_CTRL1.CNTEN = 1)

当 TI1 检测到电平由低变高时，计数器停止计数，当 TI1 检测到电平由高变低时，计数器开始计数，开始或停止计数时触发标志置位 (TIMx\_STS.TITF=1);

TI1 的上升沿与实际计数器停止之间的延迟是由于 T11 输入的重新同步电路引起的。

图 10-25 门控模式下的控制电路



#### 10.3.13.4 从模式：触发模式 +外部时钟模式 2

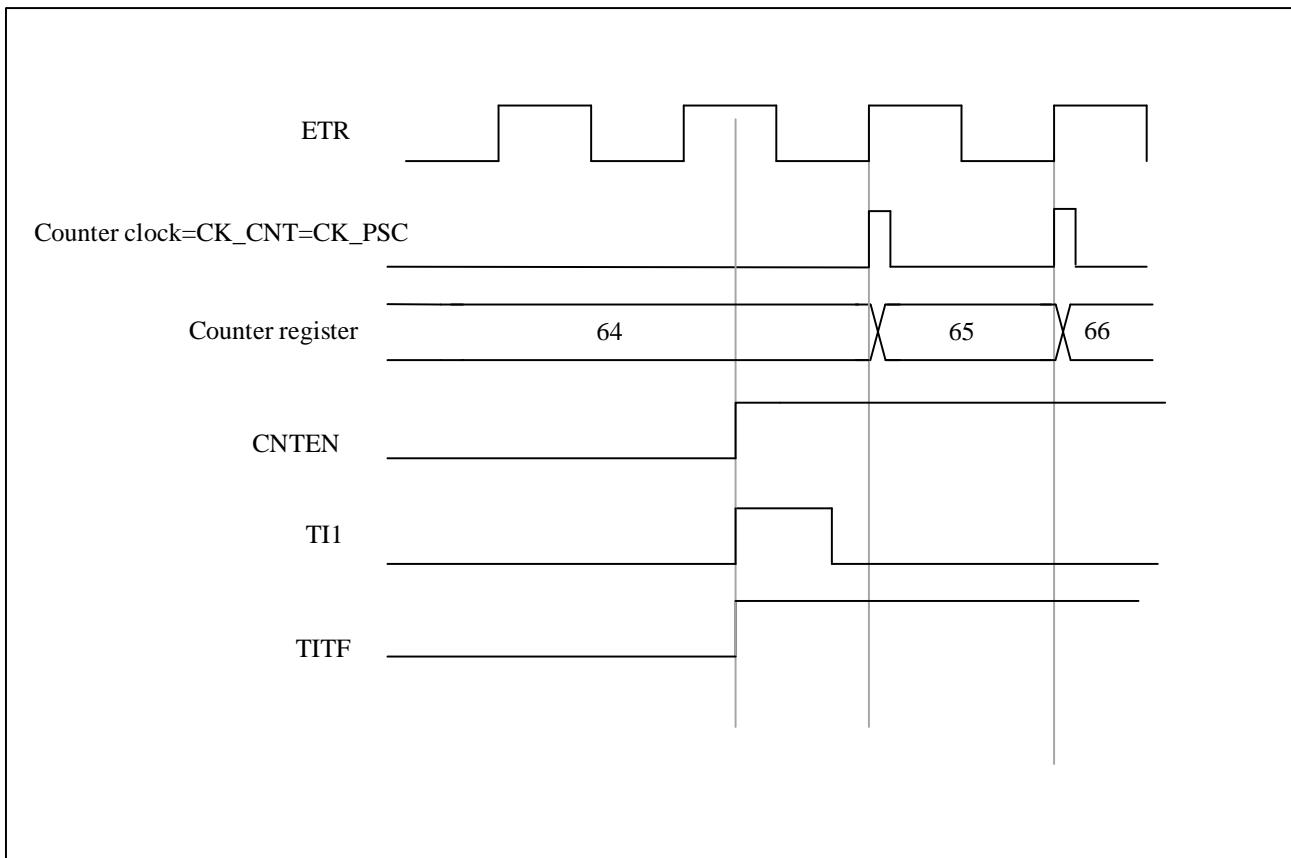
在复位模式、触发模式和门控模式下，计数器时钟可选择为外部时钟模式 2，ETR 信号作为外部时钟源输入。这时候触发选择需要选择非 ETRF (TIMx\_SMCTRL.TSEL=111)。

这是一个例子：

1. 通道 1 配置为输入检测 TI1 的上升沿 (TIMx\_CCMOD1.CC1SEL=01, TIMx\_CCEN.CC1P=0);
2. 使能外部时钟模式 2 (TIMx\_SMCTRL.EXCEN=1), 外部触发极性选择上升沿 (TIMx\_SMCTRL.EXTP=0), 触发模式作为从模式 (TIMx\_SMCTRL.SMSEL=110), TRGI 选择 TI1 ( TIMx\_SMCTRL.TSEL=101);

当 TI1 检测到上升沿时，计数器在 ETR 的上升沿开始计数，并设置触发标志 (TIMx\_STS.TITF=1);

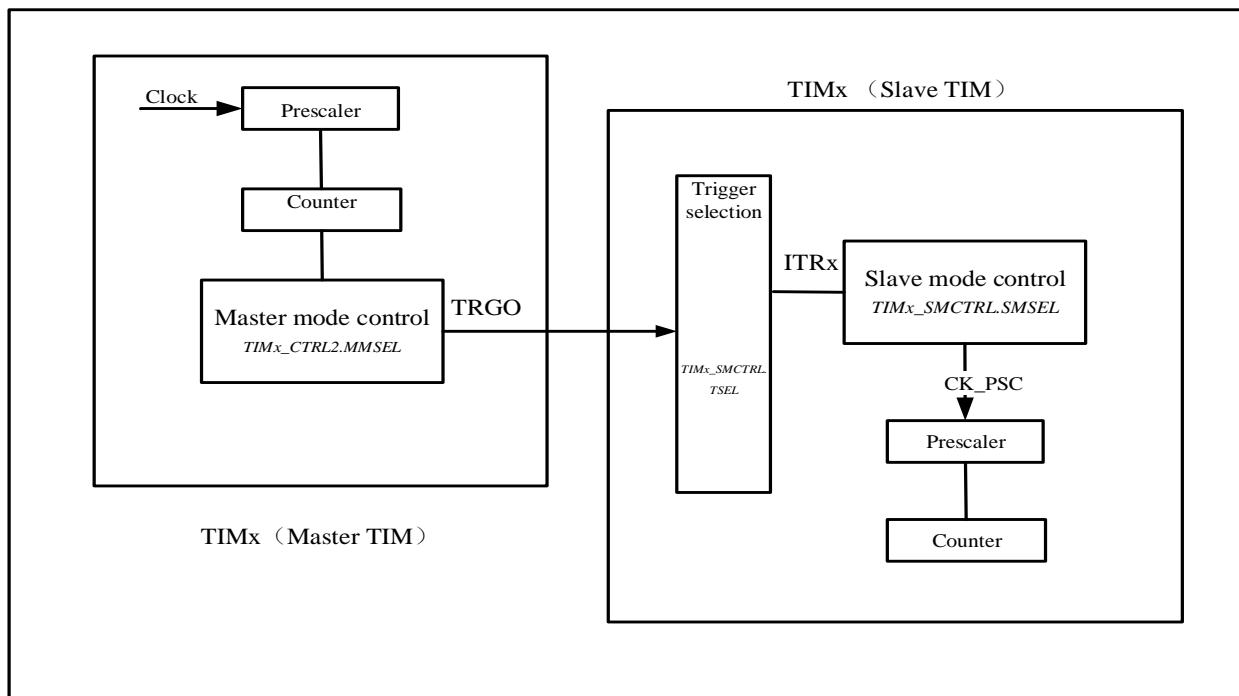
图 10-26 外部时钟模式 2+触发模式下的控制电路



### 10.3.14 定时器同步

所有 TIMx 定时器都在内部相互连接。该实现允许任何主定时器提供触发以复位、启动、停止或为其他从定时器提供时钟。主时钟用于内部计数器，可以预分频。下图为定时器互连框图。同步功能不支持连接的动态变化。用户应在启用主定时器的触发器或时钟之前配置并启用从定时器。

图 10-27 主/从定时器的例子



#### 10.3.14.1 主定时器作为另一个定时器的预分频器

定时器 1 作为定时器 2 的预分频器。TIM1 是主，TIM2 是从。

用户需要为此配置执行以下步骤。

- 设置 `TIM1_CTRL2.MMSEL='010'` 以使用 TIM1 的更新事件作为触发输出。
- 配置 `TIM2_SMCTRL.TSEL='000'`，将 TIM1 的 TRGO 连接到 TIM2。
- 配置 `TIM2_SMCTRL.SMSEL = '111'`，从模式控制器将配置为外部时钟模式 1。
- 通过设置 `TIM2_CTRL1.CNTEN = "1"`, 启动 TIM2。
- 通过设置 `TIM1_CTRL1.CNTEN = "1"`, 启动 TIM1。

*注：如果用户通过配置 `MMSEL = '1xx'` 选择 `OCx` 作为 TIM1 的触发输出，则 `OCx` 上升沿将用于驱动 timer2。*

#### 10.3.14.2 主定时器使能另一个定时器

在本例中，TIM2 通过 TIM1 的输出比较使能。TIM1 的 OC1REF 输出为高电平后，TIM2 计数器将开始计数。两个计数器的时钟均基于 `CK_INT`，通过预分频器除以 3 ( $f_{CK\_CNT} = f_{CK\_INT}/3$ )。

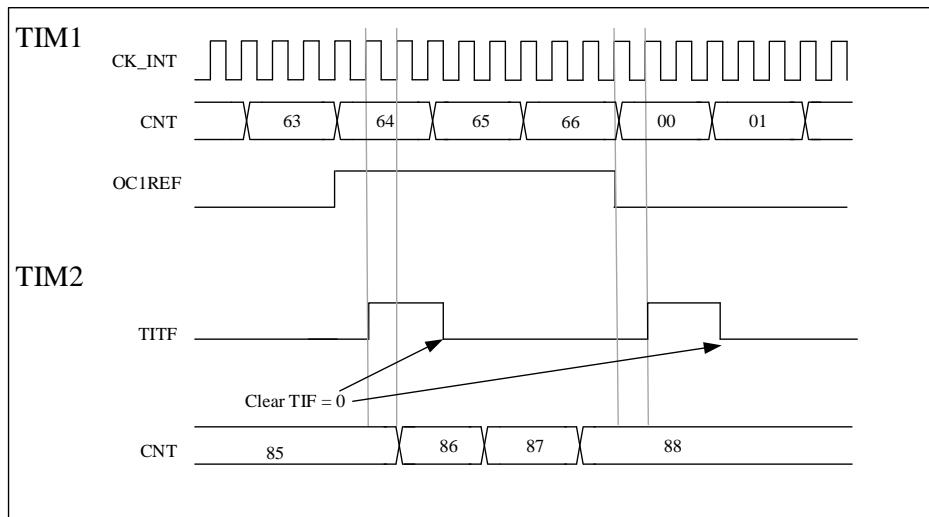
配置步骤如下所示。

- 设置 `TIM1_CTRL2.MMSEL='100'` 以使用 TIM1 的 OC1REF 作为触发输出。
- 配置 `TIM1_CCMOD1` 寄存器来配置 OC1REF 输出波形。
- 设置 `TIM2_SMCTRL.TSEL = '000'` 将 TIM1 触发输出连接到 TIM2。
- 设置 `TIM2_SMCTRL.SMSEL = '101'` 将 TIM2 设置为门控模式。
- 设置 `TIM2_CTRL1.CNTEN = '1'` 来启动 TIM2。

- 设置 `TIM1_CTRL1.CNTEN= '1'` 以启动 TIM1。

注: *TIM2* 时钟与 *TIM1* 时钟不同步, 该模式仅影响 *TIM2* 计数器使能信号。

图 10-28 定时器 2 由定时器 1 的 OC1REF 门控



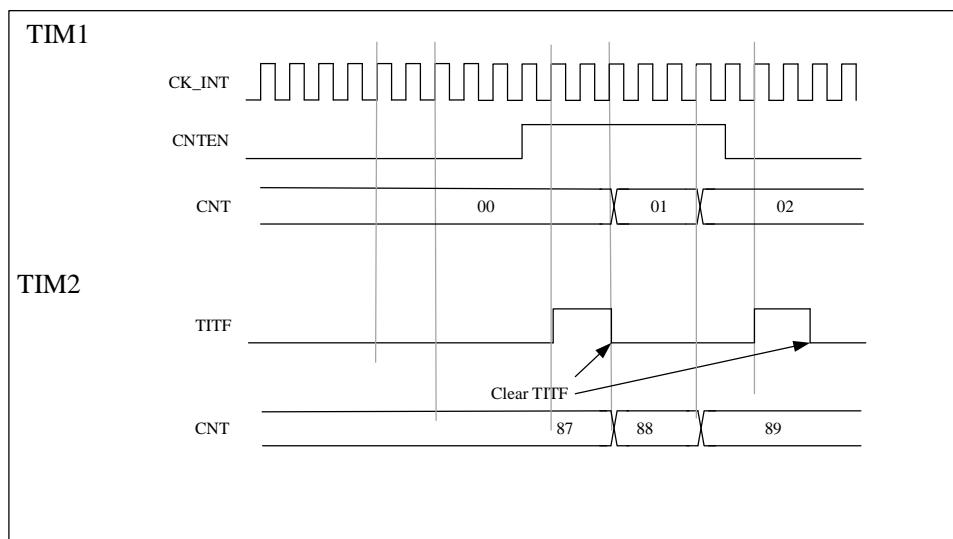
在下一个示例中, 用 TIM1 的使能信号门控 TIM2, 设置 `TIM1_CTRL1.CNTEN= '0'` 以停止 TIM1。

仅当 TIM1 使能时, TIM2 才基于分频的内部时钟计数。两个计数器的时钟均基于 `CK_INT`, 通过预分频器除以 3 ( $f_{CK\_CNT} = f_{CK\_INT}/3$ )。

配置步骤如下所示

- 设置 `TIM1_CTRL2.MMSEL='001'` 使用 TIM1 的使能信号作为触发输出
- 设置 `TIM2_SMCTRL.TSEL = '000'` 配置 TIM2 从 TIM1 获取触发输入
- 设置 `TIM2_SMCTRL.SMSEL = '101'` 将 TIM2 配置为门控模式。
- 设置 `TIM2_CTRL1.CNTEN= '1'` 来启动 TIM2。
- 设置 `TIM1_CTRL1.CNTEN= '1'` 以启动 TIM1。
- 设置 `TIM1_CTRL1.CNTEN= '0'` 以停止 TIM1。

图 10-29 定时器 2 由定时器 1 的使能门控



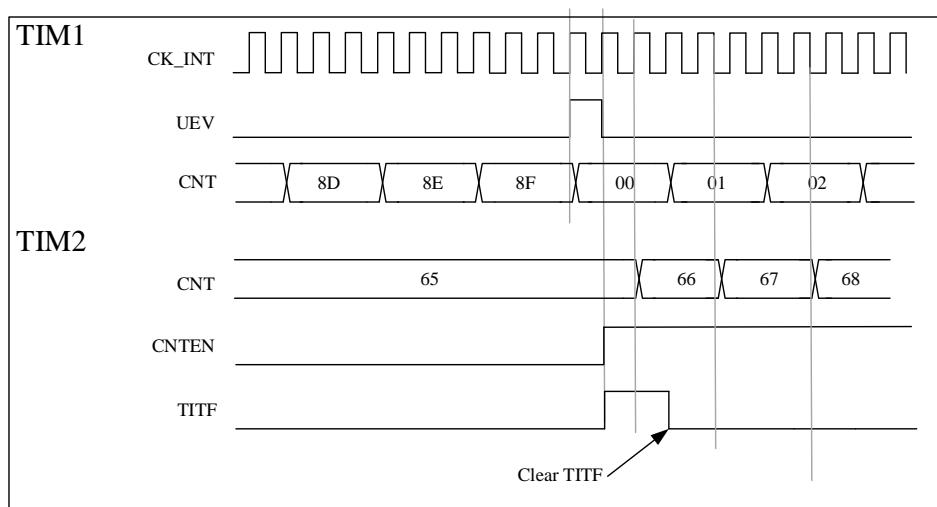
#### 10.3.14.3 主定时器启动另一个定时器

在这个例子中，我们可以使用更新事件作为触发源。TIM1 是主，TIM2 是从。

配置步骤如下图所示：

- 设置 TIM1\_CTRL2.MMSEL='010' 使用 TIM1 的更新事件作为触发输出
- 配置 TIM1\_AR 寄存器设置输出周期。
- 设置 TIM2\_SMCTRL.TSEL='000' 将 TIM1 触发输出连接到 TIM2。
- 设置 TIM2\_SMCTRL.SMSEL='110' 将 TIM2 设置为触发模式。
- 设置 TIM1\_CTRL1.CNTEN=1 启动 TIM1。

图 10-30 使用定时器 1 的更新触发定时器 2



#### 10.3.14.4 使用一个外部触发同步地启动 2 个定时器

在本例中，TIM1 的 TI1 输入上升时使能 TIM1，使能 TIM1 时使能 TIM2。为确保计数器对齐，TIM1 必须配

置为主/从模式。对于 TIM1，TIM1 是从；对于 TIM2，TIM1 是主。

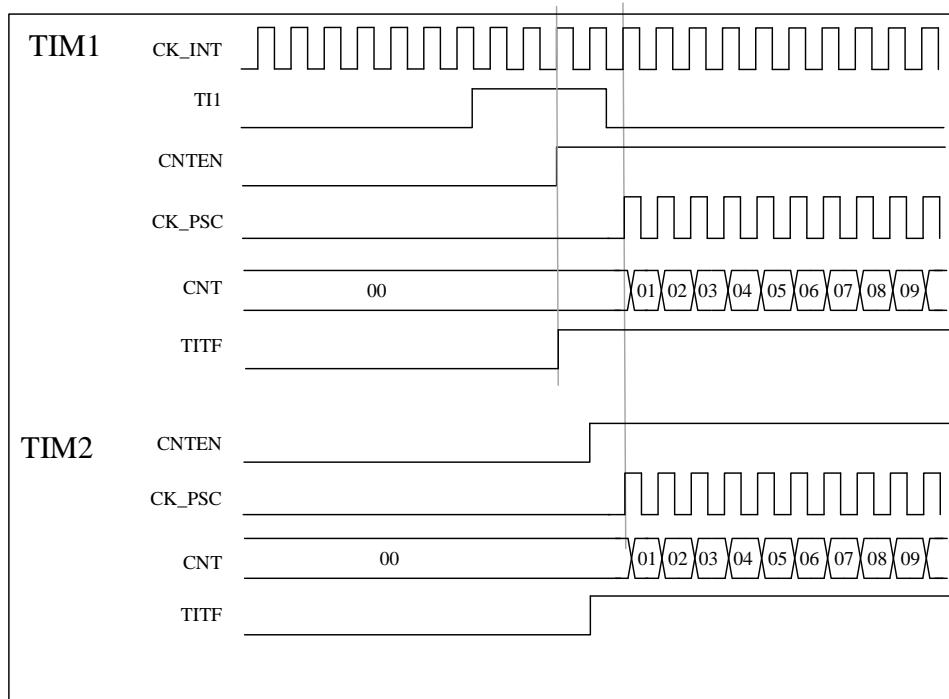
配置步骤如下图所示：

- 设置 TIM1.MMSEL = ‘001’ 使用使能信号作为触发输出
- 设置 TIM1\_SMCTRL.TSEL = ‘100’ 将 TIM1 配置为从模式并接收 TI1 的触发输入。
- 设置 TIM1\_SMCTRL.SMSEL = ‘110’ 将 TIM1 配置为触发模式。
- 设置 TIM1\_SMCTRL.MSMD = ‘1’ 将 TIM1 配置为主/从模式。
- 设置 TIM2\_SMCTRL.TSEL = ‘000’ 将 TIM1 触发输出连接到 TIM2。
- 设置 TIM2\_SMCTRL.SMSEL = ‘110’ 将 TIM2 配置为触发模式。

当 TI1 上升沿到来时，两个定时器开始根据内部时钟同步计数，两个 TITF 标志同时置位。

注：下图显示了在主/从模式下 CNTEN 和 TIM1 的 CK\_PSC 之间的延迟。

图 10-31 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2



### 10.3.15 编码器接口模式

编码器使用两个输入 TI1 和 TI2 作为接口，计数器对 TI1FP1 或 TI2FP2 上的每个边沿变化进行计数。计数方向由硬件 TIMx\_CTRL1.DIR 自动控制。编码器计数模式共有三种：

1. 计数器只在 TI1 的边沿计数，TIMx\_SMCTRL.SMSEL = ‘001’；
2. 计数器只在 TI2 的边沿计数，TIMx\_SMCTRL.SMSEL = ‘010’；
3. 计数器同时在 TI1 和 TI2 的边沿计数，TIMx\_SMCTRL.SMSEL = ‘011’；

编码器接口相当于使用带方向选择的外部时钟，计数器只在 0 和自动重载值 (TIMx\_AR.AR [15:0]) 之间连续计数。因此，需要提前配置自动重载寄存器 TIMx\_AR。

注意：编码器模式和外部时钟模式 2 不兼容，不能同时选择。

计数方向与编码器信号的关系如下表：

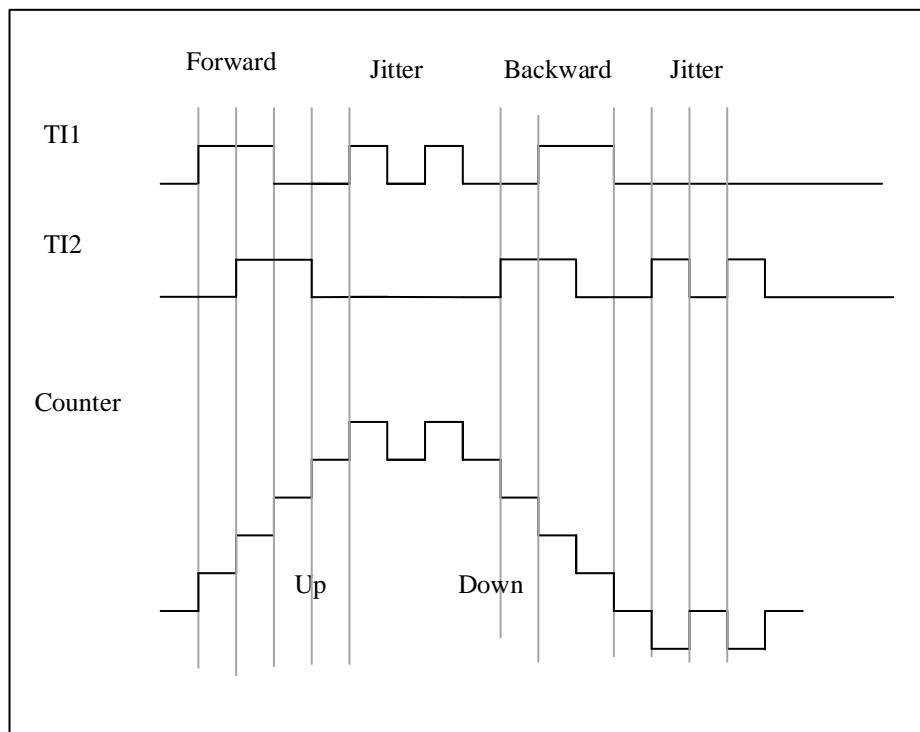
表 10-1 计数方向与编码器信号的关系

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

以下是选择了双边沿触发以抑制输入抖动的编码器示例：

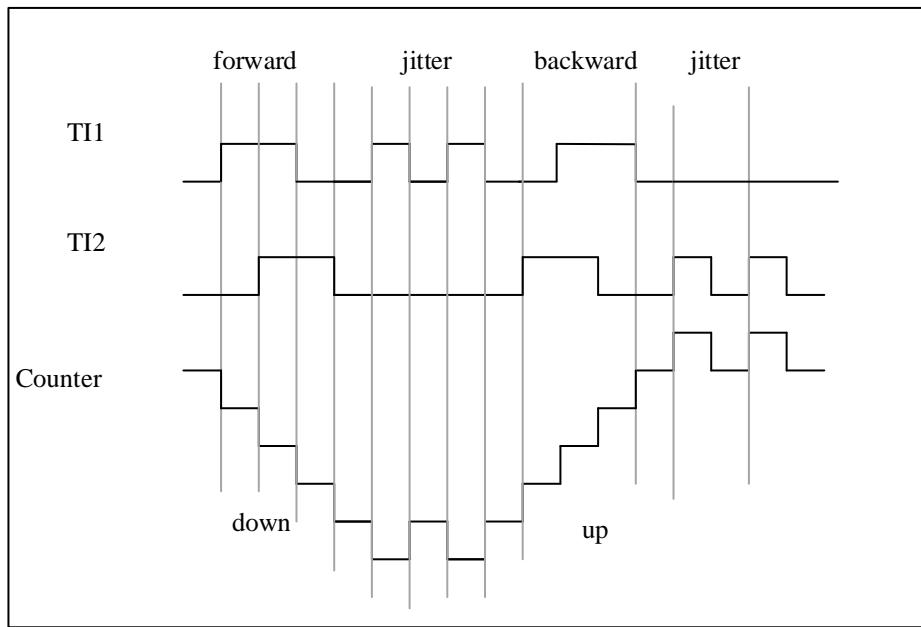
1. IC1FP1 映射到 TI1 (TIMx\_CCMOD1.CC1SEL=‘01’), IC1FP1 不反相 (TIMx\_CCEN.CC1P=‘0’);
2. IC1FP2 映射到 TI2 (TIMx\_CCMOD2.CC2SEL=‘01’), IC2FP2 不反相 (TIMx\_CCEN.CC2P=‘0’);
3. 输入在上升沿和下降沿均有效 (TIMx\_SMCTRL.SMSEL = ‘011’);
4. 启用计数器 TIMx\_CTRL1.CNTEN=‘1’;

图 10-32 编码器模式下的计数器操作实例



下图为 IC1FP1 极性反转时的计数器行为示例（CC1P='1'，其他配置同上）

图 10-33 IC1FP1 反相的编码器接口模式实例



### 10.3.16 与霍尔传感器的接口

请查阅9.3.20

## 10.4 TIMx 寄存器描述 (x=2, 3, 4 and 5)

关于在寄存器描述里面所用到的缩写，详见第 1.1 节。

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

### 10.4.1 寄存器总览

表 10-2 TIM2、TIM3、TIM4 and TIM5 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
000 h	TIMx_CTL1																	
	Reset Value																	
004 h	TIMx_CTL2																	
	Reset Value																	
008 h	TIMx_SM_CTRL																	
	Reset Value																	
00Ch	TIMx_DIN_TEN																	
	Reset Value																	
010 h	TIMx_STS																	
	Reset Value																	
014 h	TIMx_EV_TGEN																	
	Reset Value																	
018 h	TIMx_CC_MOD1_Output_comparison mode																	
	Reset Value																	



#### 10.4.2 控制寄存器 1 (TIMx\_CTRL1)

偏移地址: 0x00

复位值: 0x0000

位域	名称	描述
31:16	Reserved	保留，必须保持复位值。
15	CLRSEL	OCxREF clear selection 0: 选择外部OCxREF clear (来自 ETR) 1: 选择OCxREF clear (来自比较器) <i>注: TIM5 不适用.</i>
14	C4SEL	Channel 4 Selection 0: 选择外部CH4信号 (来自 IOM)

位域	名称	描述
		1: 选择内部CH4信号 (来自HSE/128) 注: 仅TIM2适用. 对于 TIM3, TIM4 and TIM5, 只接到 IOM.
13	C3SEL	Channel 3 Selection 0: 选择外部CH3 信号(来自 IOM) 1: 选择内部CH3 信号 (来自 LSI) 注: 仅TIM2适用. 对于 TIM3, TIM4 and TIM5, 只接到 IOM.
12	C2SEL	Channel 2 Selection 0: 选择外部 CH2 (来自 IOM) signal 1: 选择内部 CH2 (来自 LSE) signal 注: 仅TIM2适用. 对于 TIM3, TIM4 and TIM5, 只接到 IOM..
11	C1SEL	Channel 1 selection 0: 选择外部 CH1 signal 来自 IOM 1: 选择内部 CH1 signal 来自 COMP
10	Reserved	保留, 必须保持复位值。
9:8	CLKD[1:0]	时钟分频因子 (Clock division) CLKD[1:0] 表示 CK_INT (定时器时钟) 和 DTS (用于死区时间发生器和数字滤波器 (ETR、TIx) 的时钟) 之间的分频比。 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	ARPEN	自动重装载预装载允许位 (Auto-reload preload enable) 0: TIMx_AR 寄存器的影子寄存器禁用 1: TIMx_AR 寄存器的影子寄存器使能

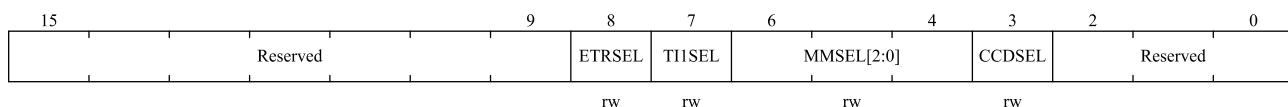
位域	名称	描述
6:5	CAMSEL[1:0]	<p>选择中央对齐模式 (Center-aligned mode selection)</p> <p>00: 边缘对齐模式。 TIMx_CTRL1.DIR 指定向 上计数或向下计数。</p> <p>01: 中央对齐模式1。 计数器在中央对齐模式下计数，向下计数时输出比较中断标志位设置为 1。</p> <p>10: 中央对齐模式2。 计数器在中央对齐模式下计数，向上计数时输出比较中断标志位设置为1。</p> <p>11: 中央对齐模式3。 计数器在中央对齐模式下计数，向上计数或向下计数时输出比较中断标志位设置为 1。</p> <p><b>注意：</b>当计数器仍然启用时 (<math>TIMx\_CTRL1.CNTEN = 1</math>)，不允许从边缘对齐模式切换到中央对齐模式。</p>
4	DIR	<p>方向 (Direction)</p> <p>0: 计数器向上计数；</p> <p>1: 计数器向下计数。</p> <p><b>注：</b>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。</p>
3	ONEPM	<p>单脉冲模式 (One pulse mode)</p> <p>0: 禁用单脉冲模式，发生更新事件时不影响计数器计数。</p> <p>1: 使能单脉冲模式，下次更新事件发生时计数器停止计数</p>
2	UPRS	<p>更新请求源 (Update request source)</p> <p>该位用于通过软件选择 UEV 事件源。</p> <p>0: 如果更新中断或 DMA 请求使能，以下任何事件都会产生更新中断或 DMA 请求：</p> <ul style="list-style-type: none"> <li>- 计数器上溢/下溢</li> <li>- TIMx_EVTGEN.UDGN 位被设置</li> <li>- 从模式控制器的更新生成</li> </ul> <p>1: 如果更新中断或 DMA 请求使能，只有计数器上溢/下溢会产生更新中断或 DMA 请求。</p>

位域	名称	描述
1	UPDIS	<p>更新禁用 (Update disable)</p> <p>该位用于启用/禁用软件生成的更新事件 (UEV) 事件。</p> <p>0: 启用。 如果满足以下条件之一，将生成 UEV:</p> <ul style="list-style-type: none"> <li>- 计数器上溢/下溢</li> <li>- TIMx_EVTGEN.UDGN 位被设置</li> <li>- 从模式控制器的更新生成</li> </ul> <p>影子寄存器将使用预加载值进行更新。</p> <p>1: UEV 禁用。 不生成更新事件，影子寄存器 (AR、PSC 和 CC DATx) 保持它们的值。 如果 TIMx_EVTGEN.UDGN 位置位或从模式控制器发出硬件复位，则重新初始化计数器和预分频器。</p>
0	CNTEN	<p>使能计数器 (Counter enable)</p> <p>0: 禁止计数器；</p> <p>1: 使能计数器。</p> <p>注: 在软件设置了CNTEN位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置CNTEN位。</p>

### 10.4.3 控制寄存器 2 (TIMx\_CTRL2)

偏移地址: 0x04

复位值: 0x0000



位域	名称	描述
15: 9	Reserved	保留，必须保持复位值。
8	ETRSEL	<p>ETR选择(ETR Selection)</p> <p>0:选择外部 ETR 信号(来自IOM);</p> <p>1: Reserved</p> <p>注: 对于TIM5 不适用。也就是说, TIM5 的ETR 不能接到外部输入。</p>

位域	名称	描述
7	TI1SEL	<p>TI1选择 (TI1 selection)</p> <p>0: TIMx_CH1引脚连到TI1输入；</p> <p>1: TIMx_CH1、TIMx_CH2和TIMx_CH3引脚经异或后连到TI1输入。</p>
6:4	MMSEL[2:0]	<p>主模式选择</p> <p>这 3 位 (TIMx_CTRL2.MMSEL [2:0]) 用于选择在主模式下发送到从定时器的同步信息 (TRGO)。可能的组合如下：</p> <p>000: 复位 - 当 TIMx_EVTGEN.UDGN 置位或从模式控制器产生复位时，将出现 TRGO 脉冲。在后一种情况下，TRGO 上的信号与实际复位相比有所延迟。</p> <p>001: 使能 - TIMx_CTRL1.CNTEN 位用作触发输出 (TRGO)。有时需要同时启动多个定时器或者在一段时间内开启从定时器。</p> <p>当 TIMx_CTRL1.CNTEN 位置位或门控模式下的触发输入为高电平时，计数器使能信号置位。</p> <p>当计数器使能信号由触发输入控制时，TRGO 上有一个延迟，除非选择了主/从模式（参见 TIMx_SMCTRL.MSMD 位的说明）。</p> <p>010: 更新 - 选择更新事件作为触发输出 (TRGO)。例如，主定时器时钟可用作从定时器预分频器。</p> <p>011: 比较脉冲 - 当 TIMx_STS.CC1ITF 被设置时（即使它已经是高电平），即捕获或比较成功时，触发输出发送一个正脉冲 (TRGO)。</p> <p>100: 比较 - OC1REF 信号用作触发输出 (TRGO)。</p> <p>101: 比较 - OC2REF 信号用作触发输出 (TRGO)。</p> <p>110: 比较 - OC3REF 信号用作触发输出 (TRGO)。</p> <p>111: 比较 - OC4REF 信号用作触发输出 (TRGO)。</p>
3	CCDSEL	<p>捕获/比较的DMA选择 (Capture/compare DMA selection)</p> <p>0: 当发生CCx事件时，送出CCx的DMA请求；</p> <p>1: 当发生更新事件时，送出CCx的DMA请求。</p>
2:0	Reserved	保留，必须保持复位值。

#### 10.4.4 从模式控制寄存器 (TIMx\_SMCTRL)

偏移地址: 0x08

复位值: 0x0000

15	14	13	12	11	8	7	6	4	3	2	0
EXTP	EXCEN	EXTPS[1:0]		EXTF[3:0]	MSMD	TSEL[2:0]	Reserved	SMSEL[2:0]			

rw

rw

rw

rw

rw

rw

rw

位域	名称	描述
15	EXTP	<p>外部触发极性 (External trigger polarity)</p> <p>该位选择是用ETR还是ETR的反相来作为触发操作</p> <p>0: ETR高电平或上升沿有效;</p> <p>1: ETR低电平或下降沿有效。</p>
14	EXCEN	<p>外部时钟使能位 (External clock enable) 该位启用外部时钟模式2。启用后，计数器由ETRF信号上的任意有效边沿驱动。</p> <p>0: 禁止外部时钟模式2;</p> <p>1: 使能外部时钟模式2。</p> <p>注 1: 当同时使能外部时钟模式 1 和外部时钟模式 2 时，外部时钟的输入为 ETRF。</p> <p>注2: 以下从机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, TRGI 无法连接到 ETRF (<math>\text{TIMx\_SMCTRL.TSEL} \neq '111'</math>)。</p> <p>注 3: 设置 <math>\text{TIMx\_SMCTRL.EXCEN}</math> 位与选择外部时钟模式 1 并将 TRGI 连接到 ETRF (<math>\text{TIMx\_SMCTRL.SMSEL} = 111</math> 和 <math>\text{TIMx\_SMCTRL.TSEL} = 111</math>) 的效果相同</p>
13:12	EXTPS[1:0]	<p>外部触发预分频 (External trigger prescaler)</p> <p>外部触发信号 ETRP 的频率必须最多为 TIMxCLK 频率的 1/4。当输入更快的外部时钟时，可以使用预分频器来降低 ETRP 的频率。</p> <p>00: 关闭预分频;</p> <p>01: ETRP频率除以2;</p> <p>10: ETRP频率除以4;</p> <p>11: ETRP频率除以8。</p>
11:8	EXTF[3:0]	<p>外部触发滤波 (External trigger filter)</p> <p>这些位用于定义 ETRP 信号的采样频率和 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，在记录连续 N 个事件后生成验证输出。</p> <p>0000: 无滤波器，以fDTS采样 1000: 采样频率fSAMPLING=fDTS/8, N=6</p> <p>0001: 采样频率fSAMPLING=fCK_INT, N=2 1001: 采样频率fSAMPLING=fDTS/8, N=8</p> <p>0010: 采样频率fSAMPLING=fCK_INT, N=4 1010: 采样频率fSAMPLING=fDTS/16, N=5</p>

位域	名称	描述
		0011: 采样频率fSAMPLING=fCK_INT, N=8    1011: 采样频率fSAMPLING=fDTS/16, N=6 0100: 采样频率fSAMPLING=fDTS/2, N=6    1100: 采样频率fSAMPLING=fDTS/16, N=8 0101: 采样频率fSAMPLING=fDTS/2, N=8    1101: 采样频率fSAMPLING=fDTS/32, N=5 0110: 采样频率fSAMPLING=fDTS/4, N=6    1110: 采样频率fSAMPLING=fDTS/32, N=6 0111: 采样频率fSAMPLING=fDTS/4, N=8    1111: 采样频率fSAMPLING=fDTS/32, N=8
7	MSMD	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	TSEL[2:0]	触发选择 (Trigger selection) 这3位选择用于同步计数器的触发输入。 000: 内部触发0 (ITR0)  100: TI1的边沿检测器 (TI1F_ED) 001: 内部触发1 (ITR1)  101: 滤波后的定时器输入1 (TI1FP1) 010: 内部触发2 (ITR2)  110: 滤波后的定时器输入2 (TI2FP2) 011: 内部触发3 (ITR3)  111: 外部触发输入 (ETRF) 更多有关ITRx的细节, 参见表10-3。 注: 这些位只能在未用到 (如SMSEL=000) 时被改变, 以避免在改变时产生错误的边沿检测。
3	Reserved	保留, 必须保持复位值。
2:0	SMSEL[2:0]	从模式选择 (Slave mode selection) 当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关 (见输入控制寄存器和控制寄存器的说明) 000: 关闭从模式 – 如果CNTEN=1, 则预分频器直接由内部时钟驱动。 001: 编码器模式1 – 根据TI2FP2的电平, 计数器在TI1FP1的边沿向上/下计数。 010: 编码器模式2 – 根据TI1FP1的电平, 计数器在TI2FP2的边沿向上/下计数。

位域	名称	描述
		<p>011: 编码器模式3 – 根据另一个信号的输入电平, 计数器在TI1FP1和TI2FP2的边沿向上/下计数。</p> <p>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, 并更新影子寄存器。</p> <p>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。</p> <p>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。</p> <p>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。</p> <p>注: 如果TI1F_ED被选为触发输入 (TSEL=100) 时, 不要使用门控模式。这是因为, TI1F_ED在每次TI1F变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。</p>

表 10-3 TIMx 内部触发连接

Slave timer	ITR0 (TSEL = 000)	ITR1 (TSEL = 001)	ITR2 (TSEL = 010)	ITR3 (TSEL = 011)
<b>TIM2</b>	TIM1	TIM8	TIM3	TIM4
<b>TIM3</b>	TIM1	TIM2	TIM5	TIM4
<b>TIM4</b>	TIM1	TIM2	TIM3	TIM8
<b>TIM5</b>	TIM2	TIM3	TIM4	TIM8

#### 10.4.5 DMA/中断使能寄存器 (TIMx\_DINTEN)

偏移地址: 0x0C

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TDEN	Reserved	CC4DEN	CC3DEN	CC2DEN	CC1DEN	UDEN	Reserved	TIEN	Reserved	CC4IEN	CC3IEN	CC2IEN	CC1IEN	UIEN

rw                    rw

位域	名称	描述
15	Reserved	保留, 必须保持复位值。
14	TDEN	允许触发DMA请求 (Trigger DMA request enable) 0: 禁止触发DMA请求; 1: 允许触发DMA请求。

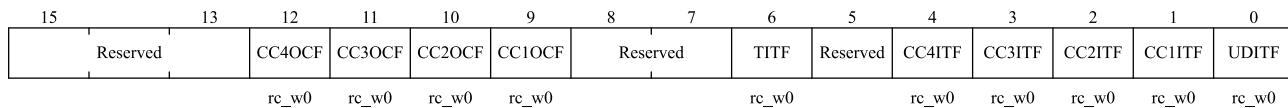
位域	名称	描述
13	Reserved	保留，必须保持复位值。
12	CC4DEN	允许捕获/比较4的DMA请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较4的DMA请求; 1: 允许捕获/比较4的DMA请求。
11	CC3DEN	允许捕获/比较3的DMA请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较3的DMA请求; 1: 允许捕获/比较3的DMA请求。
10	CC2DEN	允许捕获/比较2的DMA请求 (Capture/Compare 2 DMA request enable) 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	CC1DEN	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	UDEN	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，必须保持复位值。
6	TIEN	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，必须保持复位值。
4	CC4IEN	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	CC3IEN	允许捕获/比较3中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较3中断; 1: 允许捕获/比较3中断。

位域	名称	描述
2	CC2IEN	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	CC1IEN	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	UIEN	允许更新中断 (Update interrupt enable) 0: 禁止更新中断; 1: 允许更新中断。

#### 10.4.6 状态寄存器 (TIMx\_STS)

偏移地址: 0x10

复位值: 0x0000



位域	名称	描述
15:13	Reserved	保留, 必须保持复位值。
12	CC4OCF	捕获/比较4重复捕获标记 (Capture/Compare 4 overcapture flag) 参见CC1OCF描述。
11	CC3OCF	捕获/比较3重复捕获标记 (Capture/Compare 3 overcapture flag) 参见CC1OCF描述。
10	CC2OCF	捕获/比较2重复捕获标记 (Capture/Compare 2 overcapture flag) 参见CC1OCF描述。
9	CC1OCF	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_CCDAT1寄存器时, CC1ITF的状态已经为‘1’。

位域	名称	描述
8:7	Reserved	保留，必须保持复位值。
6	TITF	<p>触发器中断标记 (Trigger interrupt flag)</p> <p>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’。</p> <p>0：无触发器事件产生；</p> <p>1：触发中断等待响应。</p>
5	Reserved	保留，必须保持复位值。
4	CC4ITF	<p>捕获/比较4中断标记 (Capture/Compare 4 interrupt flag)</p> <p>参考CC1ITF描述。</p>
3	CC3ITF	<p>捕获/比较3中断标记 (Capture/Compare 3 interrupt flag)</p> <p>参考CC1ITF描述。</p>
2	CC2ITF	<p>捕获/比较2中断标记 (Capture/Compare 2 interrupt flag)</p> <p>参考CC1ITF描述。</p>
1	CC1ITF	<p>捕获/比较1中断标记 (Capture/Compare 1 interrupt flag)</p> <p><b>如果通道CC1配置为输出模式：</b></p> <p>除中央对齐模式外，当计数器值与比较值相同时，该位由硬件设置（参见 TIMx_CTRL1.CAMSEL 位描述）。该位由软件清零。</p> <p>0：未发生匹配。</p> <p>1：TIMx_CNT 的值与 TIMx_CCDAT1 的值相同。</p> <p>当 TIMx_CCDAT1 的值大于 TIMx_AR 的值时，如果计数器溢出（在向上计数和向上/向下计数模式下）和向下计数模式下溢，则 TIMx_STS.CC1ITF 位将变为高电平。</p> <p><b>如果通道CC1配置为输入模式：</b></p> <p>当捕捉事件发生时，该位由硬件设置。该位由软件或读取 TIMx_CCDAT1 清零。</p> <p>0：未发生输入捕捉。</p> <p>1：发生输入捕捉。计数器值已在 TIMx_CCDAT1 中捕获。在 IC1 上检测到与所选极性相同的边沿。</p>
0	UDITF	<p>更新中断标志 (Update interrupt flag)</p> <p>当在以下条件下发生更新事件时，该位由硬件设置：</p> <ul style="list-style-type: none"> <li>- 当 TIMx_CTRL1.UPDIS = 0 时，计数器上/下溢。</li> </ul>

位域	名称	描述
		<ul style="list-style-type: none"> <li>- 当 TIMx_CTRL1.UPRS = 0 时, TIMx_CTRL1.UPDIS = 0, 并通过软件设置 TIMx_EVTGEN.UDGN 位以重新初始化 CNT。</li> <li>- 当 TIMx_CTRL1.UPRS = 0 时, TIMx_CTRL1.UPDIS = 0, 并且计数器 CNT 由触发事件重新初始化。 (参见 TIMx_SMCTRL 寄存器说明)</li> </ul> <p>该位由软件清零。</p> <p>0: 未发生更新事件 1: 发生更新中断</p>

#### 10.4.7 事件产生寄存器 (TIMx\_EVTGEN)

偏移地址:0x14

复位值:0x0000

15	Reserved	7	TGN	5	Reserved	4	CC4GN	3	CC3GN	2	CC2GN	1	CC1GN	0	UDGN
		w		w		w		w		w		w		w	

位域	名称	描述
15:7	Reserved	保留, 必须保持复位值。
6	TGN	<p>产生触发事件 (Trigger generation)</p> <p>当由软件置位时, 该位可以产生一个触发事件。而此时TIMx_STS.TITF = 1, 如果相应的中断和DMA被使能, 就会产生相应的中断和DMA。该位由硬件自动清零。</p> <p>0: 无动作 1: 产生触发事件</p>
5	Reserved	保留, 必须保持复位值。
4	CC4GN	<p>产生捕获/比较4事件 (Capture/Compare 4 generation)</p> <p>参考CC1GN描述。</p>
3	CC3GN	<p>产生捕获/比较3事件 (Capture/Compare 3 generation)</p> <p>参考CC1GN描述。</p>
2	CC2GN	<p>产生捕获/比较2事件 (Capture/Compare 2 generation)</p> <p>参考CC1GN描述。</p>

位域	名称	描述
1	CC1GN	<p>产生捕获/比较1事件 (Capture/Compare 1 generation)</p> <p>当由软件设置时，该位可以产生一个捕获/比较事件。该位由硬件自动清零。</p> <p><b>CC1对应通道为输出模式时：</b></p> <p>TIMx_STS.CC1ITF 标志将被拉高，如果相应的中断和 DMA 被使能，就会产生相应的中断和 DMA。</p> <p><b>CC1对应通道为输入模式时：</b></p> <p>TIMx_CCDAT1 将捕获当前计数器值，并将 TIMx_STS.CC1ITF 标志拉高，如果相应的中断和 DMA 被使能，则会产生相应的中断和 DMA。如果 TIMx_STS.CC1ITF 已经拉高，则拉高 TIMx_STS.CC1OCF。</p> <p>0: 无动作 1: 生成 CC1 捕获/比较事件</p>
0	UDGN	<p>产生更新事件 (Update generation) 该位由软件置'1'，由硬件自动清'0'。</p> <p>当由软件设置时，该位可以生成更新事件。而此时计数器会重新初始化，预分频计数器会被清零，计数器在中央对齐或向上计数模式下会被清零，但在向下计数模式下取 TIMx_AR 寄存器的值。该位由硬件自动清零。</p> <p>0: 无动作 1: 生成更新事件</p>

#### 10.4.8 捕获/比较模式寄存器 1 (TIMx\_CCMOD1)

偏移地址: 0x18

复位值: 0x0000

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

输出比较模式：

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC2CEN		OC2MD[2:0]		OC2PEN	OC2FEN	CC2SEL[1:0]		OC1CEN		OC1MD[2:0]		OC1PEN	OC1FEN
rw		rw		rw	rw	rw		rw		rw		rw	rw

位域	名称	描述
15	OC2CEN	输出比较2清0使能 (Output Compare 2 clear enable)
14:12	OC2MD[2:0]	输出比较2模式 (Output Compare 2 mode)

位域	名称	描述
11	OC2PEN	输出比较2预装载使能 (Output Compare 2 preload enable)
10	OC2FEN	输出比较2快速使能 (Output Compare 2 fast enable)
9:8	CC2SEL[1:0]	<p>捕获/比较2选择。 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向 (输入/输出)，及输入脚的选择：</p> <p>00: CC2通道被配置为输出；</p> <p>01: CC2通道被配置为输入， IC2映射在TI2上；</p> <p>10: CC2通道被配置为输入， IC2映射在TI1上；</p> <p>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</p> <p>注: CC2SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC2EN=0) 才是可写的。</p>
7	OC1CEN	<p>输出比较1清'0'使能 (Output Compare 1 clear enable)</p> <p>0: OC1REF 不受ETRF输入的影响；</p> <p>1: 一旦检测到ETRF输入高电平，清除OC1REF=0。</p>
6:4	OC1MD[2:0]	<p>输出比较1模式 (Output Compare 1 mode)</p> <p>这些位用于管理输出参考信号 OC1REF，它决定了 OC1 和 OC1N 的值，在高电平有效，而 OC1 和 OC1N 的有效电平取决于 TIMx_CCEN.CC1P 和 TIMx_CCEN.CC1NP 位。</p> <p>000: 冻结。 TIMx_CCDAT1 寄存器和计数器 TIMx_CNT 之间的比较对 OC1REF 信号没有影响。</p> <p>001: 将通道 1 设置为匹配时的有效电平。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被强制为高电平。</p> <p>010: 将通道 1 设置为匹配时的无效电平。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被强制为低电平。</p> <p>011: 翻转。当 TIMx_CCDAT1 = TIMx_CNT 时，OC1REF 信号将被翻转。</p> <p>100: 强制无效电平。 OC1REF 信号被强制为低电平。</p> <p>101: 强制有效电平。 OC1REF 信号被强制为高电平。</p> <p>110: PWM 模式 1 - 在向上计数模式下，如果 TIMx_CNT &lt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为高电平，否则为低电平。在向下计数模式下，如果 TIMx_CNT &gt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为低电平，否则为高电平。</p> <p>111: PWM 模式 2 - 在向上计数模式下，如果 TIMx_CNT &lt; TIMx_CCDAT1，则通道 1 的 OC1REF 信号为低电平，否则为高电平。在向下计数模式下，如果 TIMx_CNT &gt;</p>

位域	名称	描述
		<p>TIMx_CC DAT1，则通道 1 的 OC1REF 信号为高电平，否则为低电平。</p> <p>注 1：在 PWM 模式 1 或 PWM 模式 2 中，OC1REF 电平仅在比较结果改变或输出比较模式从冻结模式切换到 PWM 模式时才会改变。</p>
3	OC1PEN	<p>输出比较 1 预加载使能（Output Compare 1 preload enable）</p> <p>0：禁用 TIMx_CC DAT1 寄存器的预加载功能。支持随时对 TIMx_CC DAT1 寄存器进行写操作，写入的值立即生效。</p> <p>1：使能 TIMx_CC DAT1 寄存器的预加载功能。仅对预加载寄存器进行读写操作。当更新事件发生时，TIMx_CC DAT1 的值被加载到影子寄存器中。</p> <p>注 1：只有当 TIMx_CTRL1.ONEPM = 1（在单脉冲模式下）时，才能使用 PWM 模式而不验证预加载寄存器，否则无法预测其他行为。</p>
2	OC1FEN	<p>输出比较1 快速使能（Output Compare 1 fast enable）</p> <p>该位用于加快CC输出对触发输入事件的响应。</p> <p>0：根据计数器与CCDAT1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。</p> <p>1：输入到触发器的有效沿的作用就像发生了一次比较匹配。因此，OC1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。</p> <p>OCxFEN只在通道被配置成PWM1或PWM2模式时起作用。</p>
1:0	CC1SEL[1:0]	<p>捕获/比较1 选择。（Capture/Compare 1 selection）</p> <p>这2位定义通道的方向（输入/输出），及输入脚的选择：</p> <p>00：CC1通道被配置为输出；</p> <p>01：CC1通道被配置为输入，IC1映射在TI1上；</p> <p>10：CC1通道被配置为输入，IC1映射在TI2上；</p> <p>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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。</p> <p>注：CC1SEL仅在通道关闭时（TIMx_CCEN寄存器的CC1EN=0）才是可写的。</p>

输入捕获模式：

15 IC2F[3:0]	12 IC2PSC[1:0]	11 CC2SEL[1:0]	10 rw	9 rw	8 rw	7 rw	IC1F[3:0]	4 rw	3 rw	2 rw	1 rw	0 rw
-----------------	-------------------	-------------------	----------	---------	---------	---------	-----------	---------	---------	---------	---------	---------

位域	名称	描述																
15:12	IC2F[3:0]	输入捕获2滤波器 (Input capture 2 filter)																
11:10	IC2PSC[1:0]	输入/捕获2预分频器 (Input capture 2 prescaler)																
9:8	CC2SEL[1:0]	<p>捕获/比较2选择 (Capture/Compare 2 selection)</p> <p>这2位定义通道的方向（输入/输出），及输入脚的选择：</p> <p>00: CC2通道被配置为输出；</p> <p>01: CC2通道被配置为输入， IC2映射在TI2上；</p> <p>10: CC2通道被配置为输入， IC2映射在TI1上；</p> <p>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输入被选中时（由TIMx_SMCTRL寄存器的TSEL位选择）。</p> <p><b>注：CC2SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC2EN=0) 才是可写的。</b></p>																
7:4	IC1F[3:0]	<p>输入捕获1滤波器 (Input capture 1 filter)</p> <p>这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到N个事件后会产生一个输出的跳变：</p> <table> <tbody> <tr> <td>0000: 无滤波器，以<math>f_{DTS}</math>采样</td> <td>1000: 采样频率<math>f_{SAMPLING}=f_{DTS}/8</math>, N=6</td> </tr> <tr> <td>0001: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=2</td> <td>1001: 采样频率<math>f_{SAMPLING}=f_{DTS}/8</math>, N=8</td> </tr> <tr> <td>0010: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=4</td> <td>1010: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=5</td> </tr> <tr> <td>0011: 采样频率<math>f_{SAMPLING}=f_{CK\_INT}</math>, N=8</td> <td>1011: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=6</td> </tr> <tr> <td>0100: 采样频率<math>f_{SAMPLING}=f_{DTS}/2</math>, N=6</td> <td>1100: 采样频率<math>f_{SAMPLING}=f_{DTS}/16</math>, N=8</td> </tr> <tr> <td>0101: 采样频率<math>f_{SAMPLING}=f_{DTS}/2</math>, N=8</td> <td>1101: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=5</td> </tr> <tr> <td>0110: 采样频率<math>f_{SAMPLING}=f_{DTS}/4</math>, N=6</td> <td>1110: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=6</td> </tr> <tr> <td>0111: 采样频率<math>f_{SAMPLING}=f_{DTS}/4</math>, N=8</td> <td>1111: 采样频率<math>f_{SAMPLING}=f_{DTS}/32</math>, N=8</td> </tr> </tbody> </table>	0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=6	0001: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=8	0010: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=5	0011: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=6	0100: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=8	0101: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=5	0110: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=6	0111: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=8
0000: 无滤波器，以 $f_{DTS}$ 采样	1000: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=6																	
0001: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=2	1001: 采样频率 $f_{SAMPLING}=f_{DTS}/8$ , N=8																	
0010: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=4	1010: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=5																	
0011: 采样频率 $f_{SAMPLING}=f_{CK\_INT}$ , N=8	1011: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=6																	
0100: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=6	1100: 采样频率 $f_{SAMPLING}=f_{DTS}/16$ , N=8																	
0101: 采样频率 $f_{SAMPLING}=f_{DTS}/2$ , N=8	1101: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=5																	
0110: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=6	1110: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=6																	
0111: 采样频率 $f_{SAMPLING}=f_{DTS}/4$ , N=8	1111: 采样频率 $f_{SAMPLING}=f_{DTS}/32$ , N=8																	
3:2	IC1PSC[1:0]	<p>输入/捕获1预分频器 (Input capture 1 prescaler)</p> <p>这2位定义了CC1输入 (IC1) 的预分频系数。</p> <p>一旦TIMx_CCEN.CC1EN=0，则预分频器复位。</p> <p>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；</p> <p>01: 每2个事件触发一次捕获；</p> <p>10: 每4个事件触发一次捕获；</p> <p>11: 每8个事件触发一次捕获。</p>																

位域	名称	描述
1:0	CC1SEL[1:0]	<p>捕获/比较1选择 (Capture/Compare 1 Selection)</p> <p>这2位定义通道的方向 (输入/输出)，及输入脚的选择：</p> <ul style="list-style-type: none"> <li>00: CC1通道被配置为输出；</li> <li>01: CC1通道被配置为输入，IC1映射在TI1上；</li> <li>10: CC1通道被配置为输入，IC1映射在TI2上；</li> <li>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</li> </ul> <p>注：CC1SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC1EN=0) 才是可写的。</p>

#### 10.4.9 捕获/比较模式寄存器 2 (TIMx\_CCMOD2)

偏移地址: 0x1C

复位值: 0x0000

参看以上 CCMOD1 寄存器的描述

输出比较模式：

15	14	12	11	10	9	8	7	6	4	3	2	1	0
OC4CEN	OC4MD[2:0]	OC4PEN	OC4FEN	CC4SEL[1:0]	OC3CEN	OC3MD[2:0]	OC3PEN	OC3FEN	CC3SEL[1:0]				

rw                    rw

位域	名称	描述
15	OC4CEN	输出比较4清0使能 (Output compare 4 clear enable)
14:12	OC4MD[2:0]	输出比较4模式 (Output compare 4 mode)
11	OC4PEN	输出比较4预装载使能 (Output compare 4 preload enable)
10	OC4FEN	输出比较4快速使能 (Output compare 4 fast enable)
9:8	CC4SEL[1:0]	<p>捕获/比较4选择 (Capture/Compare 4 selection)</p> <p>该2位定义通道的方向 (输入/输出)，及输入脚的选择：</p> <ul style="list-style-type: none"> <li>00: CC4通道被配置为输出；</li> <li>01: CC4通道被配置为输入，IC4映射在TI4上；</li> <li>10: CC4通道被配置为输入，IC4映射在TI3上；</li> <li>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</li> </ul>

位域	名称	描述
		注: CC4SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC4EN=0) 才是可写的。
7	OC3CEN	输出比较3清0使能 (Output compare 3 clear enable)
6:4	OC3MD[2:0]	输出比较3模式 (Output compare 3 mode)
3	OC3PEN	输出比较3预装载使能 (Output compare 3 preload enable)
2	OC3FEN	输出比较3快速使能 (Output compare 3 fast enable)
1:0	CC3SEL[1:0]	<p>捕获/比较3选择 (Capture/Compare 3 selection)</p> <p>这2位定义通道的方向 (输入/输出), 及输入脚的选择:</p> <p>00: CC3通道被配置为输出;</p> <p>01: CC3通道被配置为输入, IC3映射在TI3上;</p> <p>10: CC3通道被配置为输入, IC3映射在TI4上;</p> <p>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</p> <p>注: CC3SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC3EN=0) 才是可写的。</p>

输入捕获模式:

15 IC4F[3:0]	12 rw	11 IC4PSC[1:0]	10 rw	9 CC4SEL[1:0]	8 rw	7 IC3F[3:0]	4 rw	3 IC3PSC[1:0]	2 rw	1 CC3SEL[1:0]	0 rw
-----------------	----------	-------------------	----------	------------------	---------	----------------	---------	------------------	---------	------------------	---------

位域	名称	描述
15:12	IC4F[3:0]	输入捕获4滤波器 (Input capture 4 filter)
11:10	IC4PSC[1:0]	输入/捕获4预分频器 (Input capture 4 prescaler)
9:8	CC4SEL[1:0]	<p>捕获/比较4选择 (Capture/Compare 4 selection)</p> <p>这2位定义通道的方向(输入/输出), 及输入脚的选择:</p> <p>00: CC4通道被配置为输出;</p> <p>01: CC4通道被配置为输入, IC4映射在TI4上;</p> <p>10: CC4通道被配置为输入, IC4映射在TI3上;</p> <p>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输入被选中时 (由TIMx_SMCTRL寄存器的TSEL位选择)。</p> <p>注: CC4SEL仅在通道关闭时 (TIMx_CCEN寄存器的CC4EN=0) 才是可写的。</p>

位域	名称	描述
7:4	IC3F[3:0]	输入捕获3滤波器 (Input capture 3 filter)
3:2	IC3PSC[1:0]	输入/捕获3预分频器 (Input capture 3 prescaler)
1:0	CC3SEL[1:0]	<p>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输入被选中时            (由TIMx_SMCTRL寄存器的TSEL位选择)。</p> <p>注: CC3SEL仅在通道关闭时(TIMx_CCEN寄存器的CC3EN=0)才是可写的。</p>

#### 10.4.10 捕获/比较使能寄存器 (TIMx\_CCEN)

偏移地址: 0x20

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CC4P	CC4EN		Reserved	CC3P	CC3EN		Reserved	CC2P	CC2EN		Reserved	CC1P	CC1EN	

位域	名称	描述
15:14	Reserved	保留, 必须保持复位值。.
13	CC4P	捕获/比较4输出极性 (Capture/Compare 4 output polarity) 参考TIMx_CCEN.CC1P的描述。
12	CC4EN	捕获/比较4输出使能 (Capture/Compare 4 output enable) 参考TIMx_CCEN.CC1EN 的描述。
11:10	Reserved	保留, 必须保持复位值。
9	CC3P	捕获/比较3输出极性 (Capture/Compare 3 output polarity) 参考TIMx_CCEN.CC1P的描述。
8	CC3E	捕获/比较3输出使能 (Capture/Compare 3 output enable) 参考TIMx_CCEN.CC1E 的描述。

位域	名称	描述
7:6	Reserved	保留，必须保持复位值。
5	CC2P	捕获/比较2输出极性 (Capture/Compare 2 output polarity) 参考TIMx_CCEN.CC1P的描述。
4	CC2EN	捕获/比较2输出使能 (Capture/Compare 2 output enable) 参考TIMx_CCEN.CC1EN的描述。
3:2	Reserved	保留，必须保持复位值。
1	CC1P	捕获/比较1输出极性 (Capture/Compare 1 output polarity) <b>CC1对应通道为输出模式时：</b> 0: OC1 高电平有效 1: OC1 低电平有效 <b>CC1对应通道为输入模式时：</b> 此时，该位用于选择是使用IC1还是IC1的反相信号作为触发信号或捕捉信号。 0: 非反相：当 IC1 产生上升沿时发生捕获动作。当用作外部触发时，IC1 是非反相的。 1: 反相：当 IC1 产生下降沿时发生捕获动作。当用作外部触发时，IC1 被反相。
0	CC1EN	捕获/比较1输出使能 (Capture/Compare 1 output enable) <b>CC1通道配置为输出：</b> 0: 关闭 – OC1禁止输出。 1: 开启 – OC1信号输出到对应的输出引脚。 <b>CC1通道配置为输入：</b> 该位决定了计数器的值是否能捕获入TIMx_CCDAT1寄存器。 0: 捕获禁止； 1: 捕获使能。

表 10-4 标准 OCx 的输出控制位

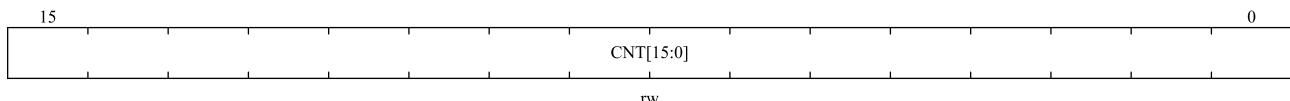
CCxEN	OCx output status
0	Disable output (OCx=0)
1	OCx = OCxREF + polarity

注：连接到标准  $OCx$  通道的外部 I/O 引脚的状态取决于  $OCx$  通道状态以及 GPIO 和 AFIO 寄存器。

#### 10.4.11 计数器 (TIMx\_CNT)

偏移地址: 0x24

复位值: 0x0000

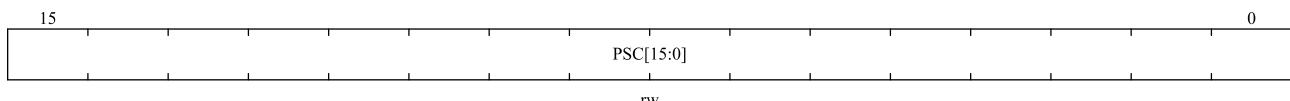


位域	名称	描述
15:0	CNT[15:0]	计数器的值 (Counter value)

#### 10.4.12 预分频器 (TIMx\_PSC)

偏移地址: 0x28

复位值: 0x0000

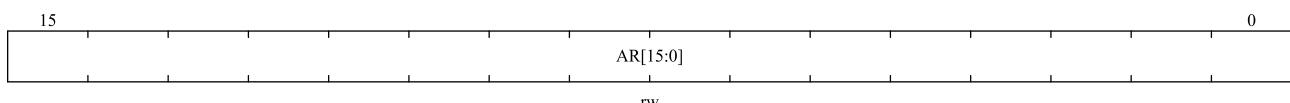


位域	名称	描述
15:0	PSC[15:0]	预分频器的值 (Prescaler value) 计数器时钟 $f_{CK\_CNT} = f_{CK\_PSC} / (PSC [15:0] + 1)$ 。 每次发生更新事件时，PSC 值都会加载到预分频器的影子寄存器中。

#### 10.4.13 自动重装载寄存器 (TIMx\_AR)

偏移地址: 0x2C

复位值: 0xFFFF

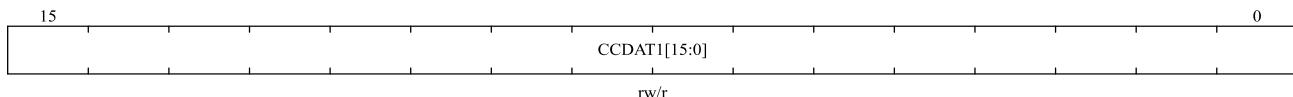


位域	名称	描述
15:0	AR[15:0]	自动重装载的值 (Auto-reload value) AR包含了将要装载入实际的自动重装载寄存器的值。详细参考10.3.1节：有关AR的更新和动作。当自动重装载的值为空时，计数器不工作。

#### 10.4.14 捕获/比较寄存器 1 (TIMx\_CC DAT1)

偏移地址: 0x34

复位值: 0x0000

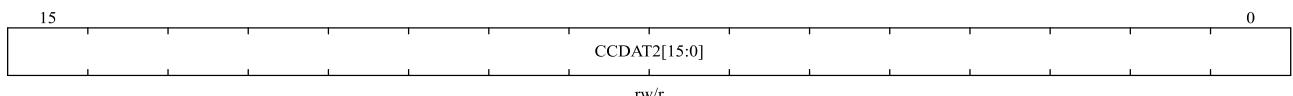


位域	名称	描述
15:0	CC DAT1[15:0]	<p>捕获/比较通道1的值 (Capture/Compare 1 value)</p> <ul style="list-style-type: none"> <li>■ CC1 通道配置为输出:</li> </ul> <p>CC DAT1 包含要与计数器 TIMx_CNT 比较的值，在 OC1 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC1PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC1 通道配置为输入:</li> </ul> <p>CC DAT1 包含由最后一个输入捕获 1 事件 (IC1) 传输的计数器值。</p> <p>当配置为输入模式时，寄存器 CC DAT1 只能读取。</p> <p>当配置为输出模式时，寄存器 CC DAT1 是可读写的。</p>

#### 10.4.15 捕获/比较寄存器 2 (TIMx\_CC DAT2)

偏移地址: 0x38

复位值: 0x0000



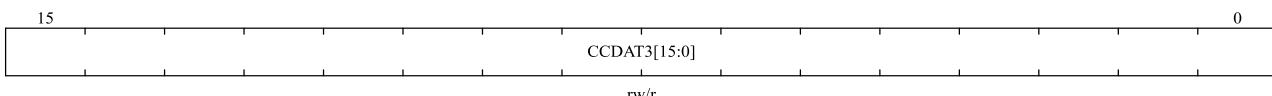
位域	名称	描述
15:0	CC DAT2[15:0]	<p>捕获/比较通道2的值 (Capture/Compare 2 value)</p> <ul style="list-style-type: none"> <li>■ CC2 通道配置为输出:</li> </ul> <p>CC DAT2 包含要与计数器 TIMx_CNT 比较的值，在 OC2 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD1.OC2PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC2 通道配置为输入:</li> </ul> <p>CC DAT2 包含由最后一个输入捕获 2 事件 (IC2) 传输的计数器值。</p>

位域	名称	描述
		当配置为输入模式时，寄存器 CCDAT2只能读取。 当配置为输出模式时，寄存器 CCDAT2是可读写的。

#### 10.4.16 捕获/比较寄存器 3 (TIMx\_CC DAT3)

偏移地址: 0x3C

复位值: 0x0000

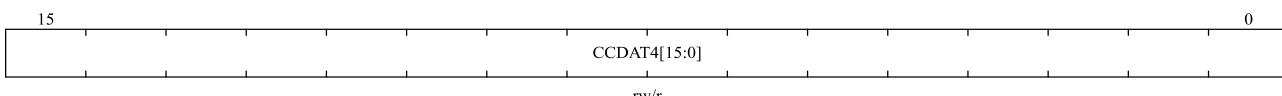


位域	名称	描述
15:0	CCDAT3[15:0]	<p>捕获/比较通道3的值 (Capture/Compare 3 value)</p> <ul style="list-style-type: none"> <li>■ CC3 通道配置为输出: CCDAT3 包含要与计数器 TIMx_CNT 比较的值，在 OC3 输出上发出信号。如果未在 TIMx_CCMOD2.OC3PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</li> <li>■ CC3 通道配置为输入: CCDAT3 包含由最后一个输入捕获 3 事件 (IC3) 传输的计数器值。 当配置为输入模式时，寄存器 CCDAT3只能读取。 当配置为输出模式时，寄存器 CCDAT3是可读写的。</li> </ul>

#### 10.4.17 捕获/比较寄存器 4 (TIMx\_CC DAT4)

偏移地址: 0x40

复位值: 0x0000



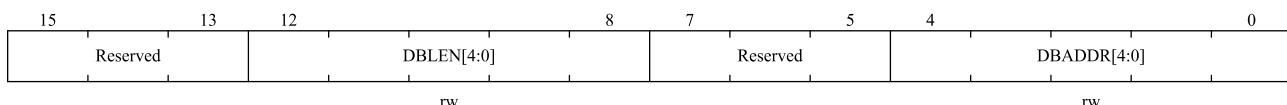
位域	名称	描述
15:0	CCDAT4[15:0]	<p>捕获/比较通道4的值 (Capture/Compare 4 value)</p> <ul style="list-style-type: none"> <li>■ CC4 通道配置为输出:</li> </ul>

位域	名称	描述
		<p>CCDAT4 包含要与计数器 TIMx_CNT 比较的值，在 OC4 输出上发出信号。</p> <p>如果未在 TIMx_CCMOD2.OC4PEN 位中选择预加载功能，则写入的值会立即传输到有效寄存器。否则，仅当更新事件发生时，此预加载值才会传输到活动寄存器。</p> <ul style="list-style-type: none"> <li>■ CC4 通道配置为输入：</li> </ul> <p>CCDAT4 包含由最后一个输入捕获 4 事件 (IC4) 传输的计数器值。</p> <p>当配置为输入模式时，寄存器 CCDAT4 只能读取。</p> <p>当配置为输出模式时，寄存器 CCDAT4 是可读写的。</p>

#### 10.4.18 DMA 控制寄存器 (TIMx\_DCTRL)

偏移地址: 0x48

复位值: 0x0000



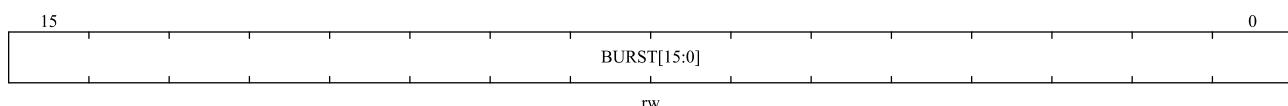
位域	名称	描述
15:13	Reserved	保留，必须保持复位值。
12:8	DBLEN[4:0]	<p>DMA连续传送长度 (DMA burst length)</p> <p>该位字段定义 DMA 将访问 (写入/读取) TIMx_DADDR 寄存器的次数。</p> <p>00000: 1次传输 00001: 2次传输 00010: 3次传输 ... 10001: 18次传输</p>
7:5	Reserved	保留，必须保持复位值。
4:0	DBADDR[4:0]	<p>DMA基地址 (DMA base address)</p> <p>该位字段定义 DMA 访问 TIMx_DADDR 寄存器的第一个地址。</p> <p>当第一次通过 TIMx_DADDR 完成访问时，该位域指定您刚刚访问的地址。然后第二次访问 TIMx_DADDR，会访问到“DMA Base Address + 4”的地址</p> <p>00000: TIMx_CTRL1, 00001: TIMx_CTRL2,</p>

位域	名称	描述
		00010: TIMx_SMCTRL, ..... 01011: TIMx_AR, 01100: Reserved, 01101: TIMx_CCDAT1, ..... 10000: TIMx_CCDAT4, 10001: Reserved, 10010: TIMx_DCTRL

#### 10.4.19 连续模式的 DMA 地址 (TIMx\_DADDR)

偏移地址: 0x4C

复位值: 0x0000

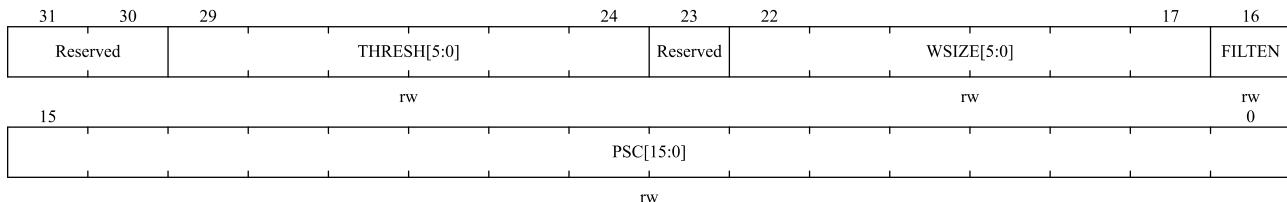


位域	名称	描述
15:0	BURST[15:0]	<p>DMA 访问缓冲区。</p> <p>当对该寄存器分配读或写操作时，将访问位于地址范围 (DMA base address + DMA burst length × 4) 的寄存器。</p> <p>DMA base address = The address of TIMx_CTRL1 + TIMx_DCTRL.DBADDR * 4;</p> <p>DMA burst len = TIMx_DCTRL.DBLEN + 1.</p> <p>例子：</p> <p>如果 TIMx_DCTRL.DBLEN = 0x3 (4 次传输)，TIMx_DCTRL.DBADDR = 0xD (TIMx_CCDAT1)，DMA 数据长度 = 半字，DMA 存储器地址 = SRAM 中的缓冲区地址，DMA 外设地址 = TIMx_DADDR 地址。</p> <p>当事件发生时，TIMx 将向 DMA 发送请求，并传输 4 次数据。</p> <p>第一次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT1 寄存器；</p> <p>第二次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT2 寄存器；</p> <p>.....</p> <p>第四次，对 TIMx_DADDR 寄存器的 DMA 访问将映射到访问 TIMx_CCDAT4 寄存器；</p>

## 10.4.20 通道1滤波寄存器(TIMx\_C1FILT)

Offset address: 0x70

Reset value: 0x0000



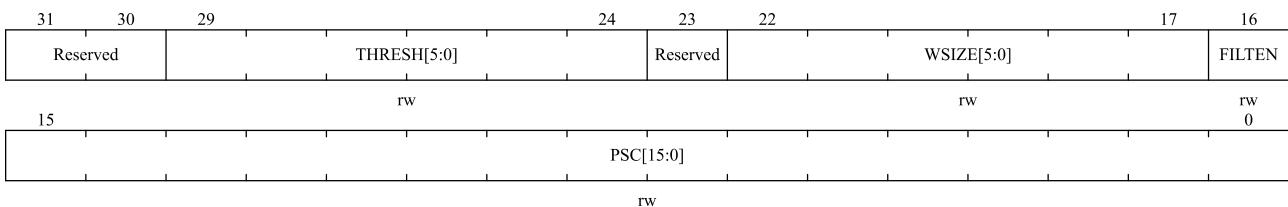
位域	名称	描述
31:30	Reserved	保留，必须保持复位值。
29:24	THRESH[5:0]	<p>采样逻辑电平有效的阈值数(Threshold)，最大 63：</p> <p>有效逻辑电平的阈值。在采样窗口内，如果逻辑高的数量大于或等于阈值，则下一个逻辑电平将为逻辑高。同样的规则适用于逻辑低。如果窗口内 1 和 0 的数量都小于阈值，则过滤器输出保持不变。阈值应设置为大于或等于 Window 值的一半。</p> <p>推荐阈值范围为：</p> <p><b>最小值：</b>比最大毛刺大小的上限（预分频时钟周期）多 1 个预分频时钟周期，并且需要大于窗口大小的一半。</p> <p>例如，如果毛刺大小为 3.2*（预分频时钟周期），则阈值应为 <math>\lceil 3.2 \rceil = 4 + 1 = 5</math></p> <p><b>最大值：</b>有效信号最小尺寸的底值（在预分频时钟周期内），需要小于窗口尺寸。</p> <p>例如，如果最小信号大小为 3.2*（预分频时钟周期），则阈值应为下限 <math>(3.2) = 3</math>。</p>
23	Reserved	保留，必须保持复位值。
22:17	WSIZE[5:0]	<p>逻辑电平检查的窗口大小值（Window size），最大 63：</p> <p>窗口大小决定了在获得下一个逻辑级别时将考虑多少采样值。内置 FIFO 为 64 位，最大索引为 63，只能将窗口大小设置为 63。</p>
16	FILTEN	<p>滤波器使能（Filter enable）：</p> <p>0：滤波器禁能。</p> <p>1：滤波器使能。</p>
15:0	PSC[15:0]	<p>用于配置滤波器采样时钟的预分频寄存器值（Prescaler）：</p> <p>对于此过滤器，它支持 65535 比例（16 位）。</p> <p>时钟预分频器将系统时钟缩放到采样时钟。采样时钟决定两个采样点之间的距离。只</p>

位域	名称	描述
		有采样点的值才会考虑有效的逻辑电平计算。

### 10.4.21 通道2滤波寄存器(TIMx\_C2FILT)

Offset address: 0x74

Reset value: 0x0000



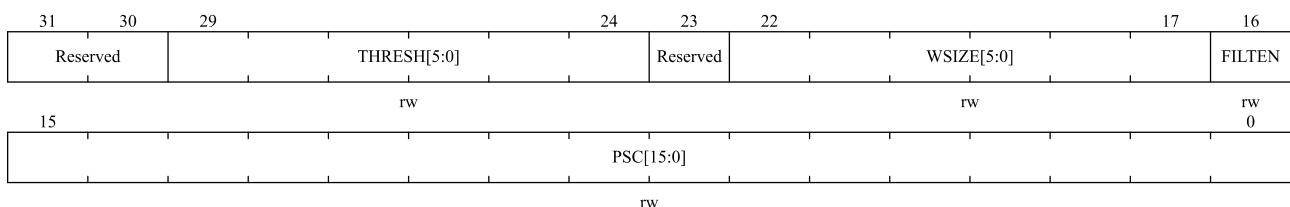
位域	名称	描述
31:30	Reserved	保留，必须保持复位值。
29:24	THRESH[5:0]	<p>采样逻辑电平有效的阈值数(Threshold)，最大 63：</p> <p>有效逻辑电平的阈值。在采样窗口内，如果逻辑高的数量大于或等于阈值，则下一个逻辑电平将为逻辑高。同样的规则适用于逻辑低。如果窗口内 1 和 0 的数量都小于阈值，则过滤器输出保持不变。阈值应设置为大于或等于 Window 值的一半。</p> <p>推荐阈值范围为：</p> <p><b>最小值：</b>比最大毛刺大小的上限（预分频时钟周期）多 1 个预分频时钟周期，并且需要大于窗口大小的一半。</p> <p>例如，如果毛刺大小为 3.2*（预分频时钟周期），则阈值应为 <math>\lceil 3.2 \rceil = 4 + 1 = 5</math></p> <p><b>最大值：</b>有效信号最小尺寸的底值（在预分频时钟周期内），需要小于窗口尺寸。</p> <p>例如，如果最小信号大小为 3.2*（预分频时钟周期），则阈值应为下限 <math>(3.2) = 3</math>。</p>
23	Reserved	保留，必须保持复位值。
22:17	WSIZE[5:0]	<p>逻辑电平检查的窗口大小值（Window size），最大 63：</p> <p>窗口大小决定了在获得下一个逻辑级别时将考虑多少采样值。内置 FIFO 为 64 位，最大索引为 63，只能将窗口大小设置为 63。</p>
16	FILTEN	<p>滤波器使能（Filter enable）：</p> <p>0：滤波器禁能。</p> <p>1：滤波器使能。</p>

位域	名称	描述
15:0	PSC[15:0]	<p>用于配置滤波器采样时钟的预分频寄存器值（Prescaler）：</p> <p>对于此过滤器，它支持 65535 比例（16 位）。</p> <p>时钟预分频器将系统时钟缩放到采样时钟。 采样时钟决定两个采样点之间的距离。 只有采样点的值才会考虑有效的逻辑电平计算。</p>

#### 10.4.22 通道3 滤波寄存器(TIMx\_C3FILT)

Offset address: 0x78

Reset value: 0x0000



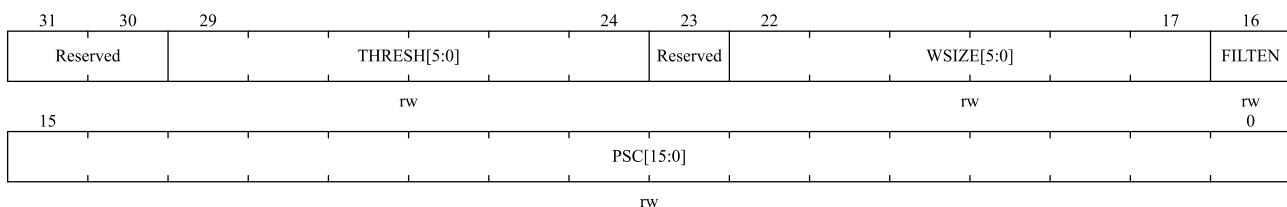
位域	名称	描述
31:30	Reserved	保留，必须保持复位值。
29:24	THRESH[5:0]	<p>采样逻辑电平有效的阈值数(Threshold)，最大 63：</p> <p>有效逻辑电平的阈值。 在采样窗口内，如果逻辑高的数量大于或等于阈值，则下一个逻辑电平将为逻辑高。 同样的规则适用于逻辑低。 如果窗口内 1 和 0 的数量都小于阈值，则过滤器输出保持不变。 阈值应设置为大于或等于 Window 值的一半。</p> <p>推荐阈值范围为：</p> <p><b>最小值：</b>比最大毛刺大小的上限（预分频时钟周期）多 1 个预分频时钟周期，并且需要大于窗口大小的一半。</p> <p>例如，如果毛刺大小为 3.2*（预分频时钟周期），则阈值应为 <math>\lceil 3.2 \rceil = 4 + 1 = 5</math></p> <p><b>最大值：</b>有效信号最小尺寸的底值（在预分频时钟周期内），需要小于窗口尺寸。</p> <p>例如，如果最小信号大小为 3.2*（预分频时钟周期），则阈值应为下限 <math>(3.2) = 3</math>。</p>
23	Reserved	保留，必须保持复位值。
22:17	WSIZE[5:0]	逻辑电平检查的窗口大小值（Window size），最大 63：

位域	名称	描述
16	FILTEN	滤波器使能 (Filter enable)： 0: 滤波器禁能。 1: 滤波器使能。
15:0	PSC[15:0]	用于配置滤波器采样时钟的预分频寄存器值 (Prescaler)： 对于此过滤器，它支持 65535 比例 (16 位)。 时钟预分频器将系统时钟缩放到采样时钟。采样时钟决定两个采样点之间的距离。只有采样点的值才会考虑有效的逻辑电平计算。

### 10.4.23 通道4滤波寄存器(TIMx\_C4FILT)

Offset address: 0x7C

Reset value: 0x0000



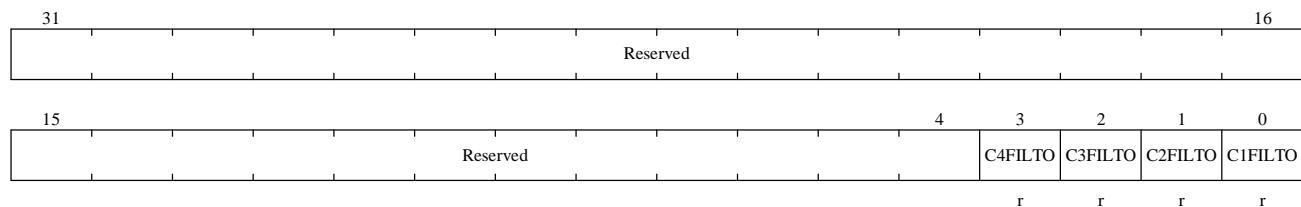
位域	名称	描述
31:30	Reserved	保留，必须保持复位值。
29:24	THRESH[5:0]	采样逻辑电平有效的阈值数(Threshold)，最大 63：  有效逻辑电平的阈值。在采样窗口内，如果逻辑高的数量大于或等于阈值，则下一个逻辑电平将为逻辑高。同样的规则适用于逻辑低。如果窗口内 1 和 0 的数量都小于阈值，则过滤器输出保持不变。阈值应设置为大于或等于 Window 值的一半。  推荐阈值范围为：  <b>最小值：</b> 比最大毛刺大小的上限（预分频时钟周期）多 1 个预分频时钟周期，并且需要大于窗口大小的一半。  例如，如果毛刺大小为 3.2*（预分频时钟周期），则阈值应为 $\lceil 3.2 \rceil = 4 + 1 = 5$  <b>最大值：</b> 有效信号最小尺寸的底值（在预分频时钟周期内），需要小于窗口尺寸。  例如，如果最小信号大小为 3.2*（预分频时钟周期），则阈值应为下限 $(3.2) = 3$ 。
23	Reserved	保留，必须保持复位值。

位域	名称	描述
22:17	WSIZE[5:0]	逻辑电平检查的窗口大小值 (Window size) , 最大 63: 窗口大小决定了在获得下一个逻辑级别时将考虑多少采样值。内置 FIFO 为 64 位, 最大索引为 63, 只能将窗口大小设置为 63。
16	FILTEN	滤波器使能 (Filter enable) : 0: 滤波器禁能。 1: 滤波器使能。
15:0	PSC[15:0]	用于配置滤波器采样时钟的预分频寄存器值 (Prescaler) : 对于此过滤器, 它支持 65535 比例 (16 位)。 时钟预分频器将系统时钟缩放到采样时钟。采样时钟决定两个采样点之间的距离。只有采样点的值才会考虑有效的逻辑电平计算。

#### 10.4.24 通道滤波输出寄存器 (TIMx\_FILTO)

Offset address: 0x80

Reset value: 0x0000



位域	名称	描述
31:4	Reserved	保留, 必须保持复位值。
3	C4_FILTO	通道4滤波输出。只读。
2	C3_FILTO	通道3滤波输出。只读。
1	C2_FILTO	通道2滤波输出。只读。
0	C1_FILTO	通道1滤波输出。只读。

## 11 基本定时器 (TIM6)

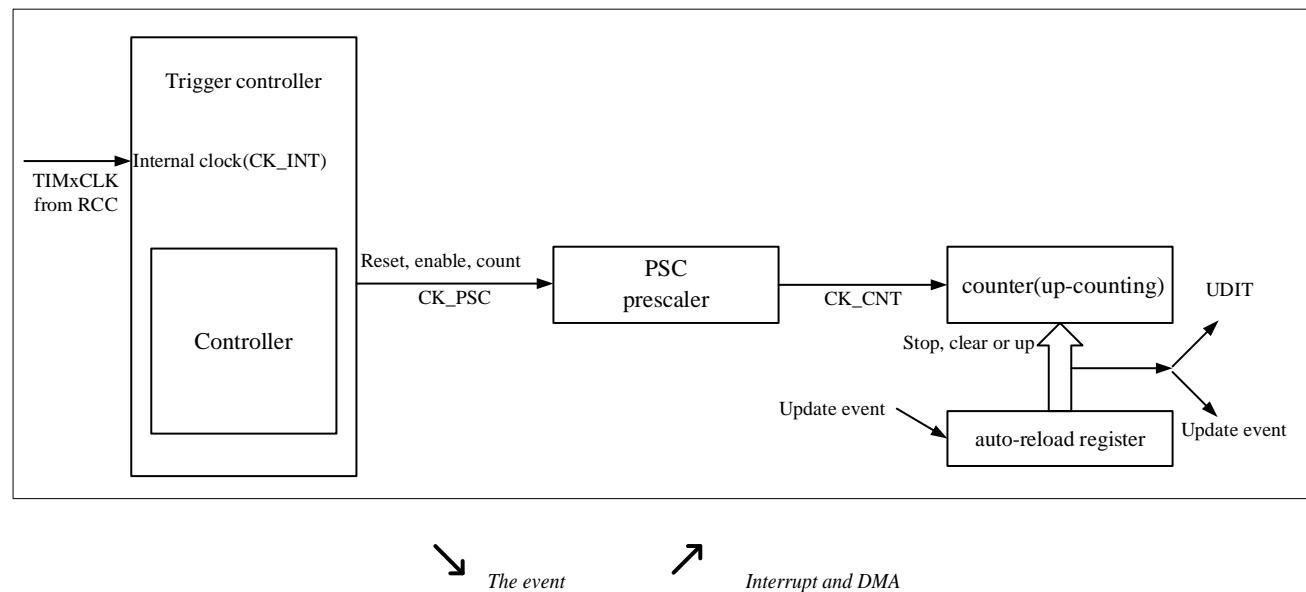
### 11.1 基本定时器简介

基本定时器包含一个 16 位计数器。

### 11.2 基本定时器主要特性

- 16 位自动重载向上计数计数器。
- 16 位可编程预分频器。(分频系数可配置为 1 到 65536 之间的任意值)
- 产生中断/DMA 的事件如下：
  - ◆ 更新事件

图 11-1 TIMx 的框图 ( $x = 6$ )



### 11.3 基础定时器描述

#### 11.3.1 时基单元

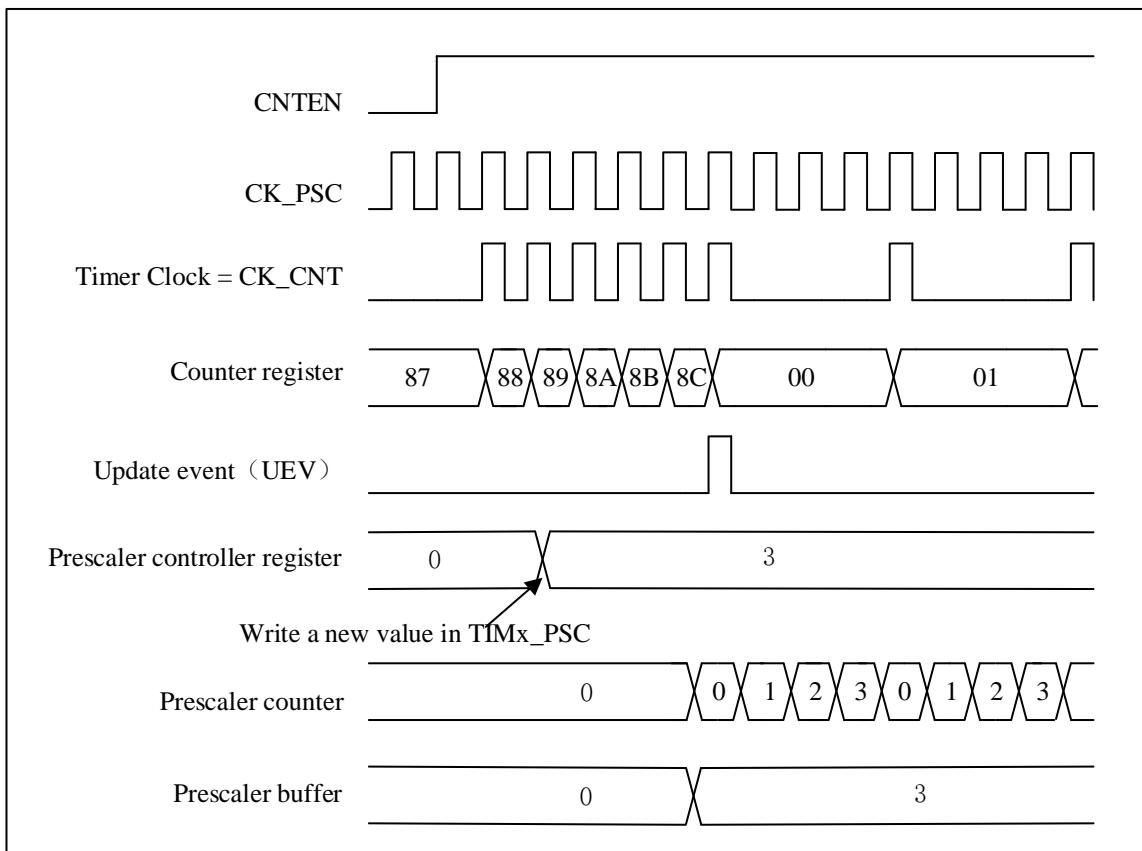
时基单元主要包括：预分频器、计数器、自动重载和重复计数器。当时基单元工作时，软件可以随时读写相应的寄存器 (TIMx\_PSC、TIMx\_CNT 和 TIMx\_AR)。

根据自动重载预加载使能位(TIMx\_CTRL1.ARREN)的设置，预加载寄存器的值会立即或在每次更新事件UEV 时传输到影子寄存器。当计数器达到溢出条件且 TIMx\_CTRL1.UPDIS=0 时，将生成更新事件。仅当 TIMx\_CTRL1.CNTEN 位置位时，计数器 CK\_CNT 才有效。计数器在 TIMx\_CTRL1.CNTEN 位置位后一个时钟周期开始计数。

### 11.3.1.1 预分频器描述

TIMx\_PSC 寄存器包含一个 16 位计数器，可用于将计数器时钟频率除以 1 到 65536 之间的任何因子。它可以在缓冲时动态更改。仅在下一次更新事件时才考虑预分频器值。

图 11-2 预分频器分频从 1 到 4 的计数器时序图



### 11.3.2 计数模式

#### 11.3.2.1 向上计数模式

在向上计数模式下，计数器会从 0 计数到寄存器 TIMx\_AR 的值，然后复位为 0。并产生计数器溢出事件。

如果设置了 TIMx\_CTRL1.UPRS 位(选择更新请求)和 TIMx\_EVTGEN.UDGN 位，则会生成更新事件(UEV)，并且不会由硬件设置 TIMx\_STS.UDITF。因此，不会产生更新中断或更新 DMA 请求。此设置用于您想要清除计数器但不想产生更新中断的场景。

如果 TIMx\_CTRL1.UPRS =0，当更新事件发生时，所有寄存器都被更新并且 TIMx\_STS.UDITF 被设置：

- 当 TIMx\_CTRL1.AR PEN =1 时，使用预加载值(TIMx\_AR)更新自动重载影子寄存器。
- 预分频器影子寄存器重新加载预加载值(TIMx\_PSC)。

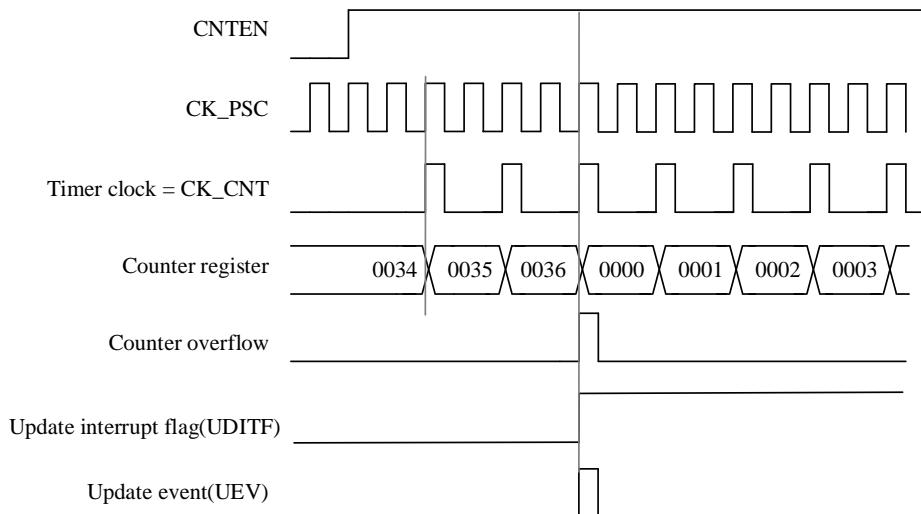
为避免在将新值写入预加载寄存器时更新影子寄存器，您可以通过设置 TIMx\_CTRL1.UPDIS=1 来禁用更新。

当更新事件发生时，计数器仍将被清零，预分频器计数器也将设置为 0 (但预分频器值将保持不变)。

下图显示了向上计数模式下不同除法因子的计数器行为和更新标志的一些示例。

图 11-3 向上计数时序图，内部时钟分频因子 = 2/N

Internal clock divided by 2



Internal clock divided by N

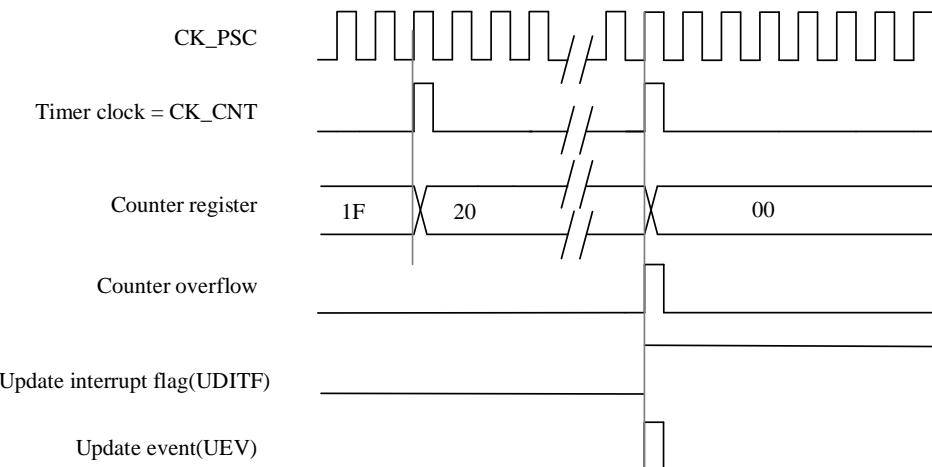
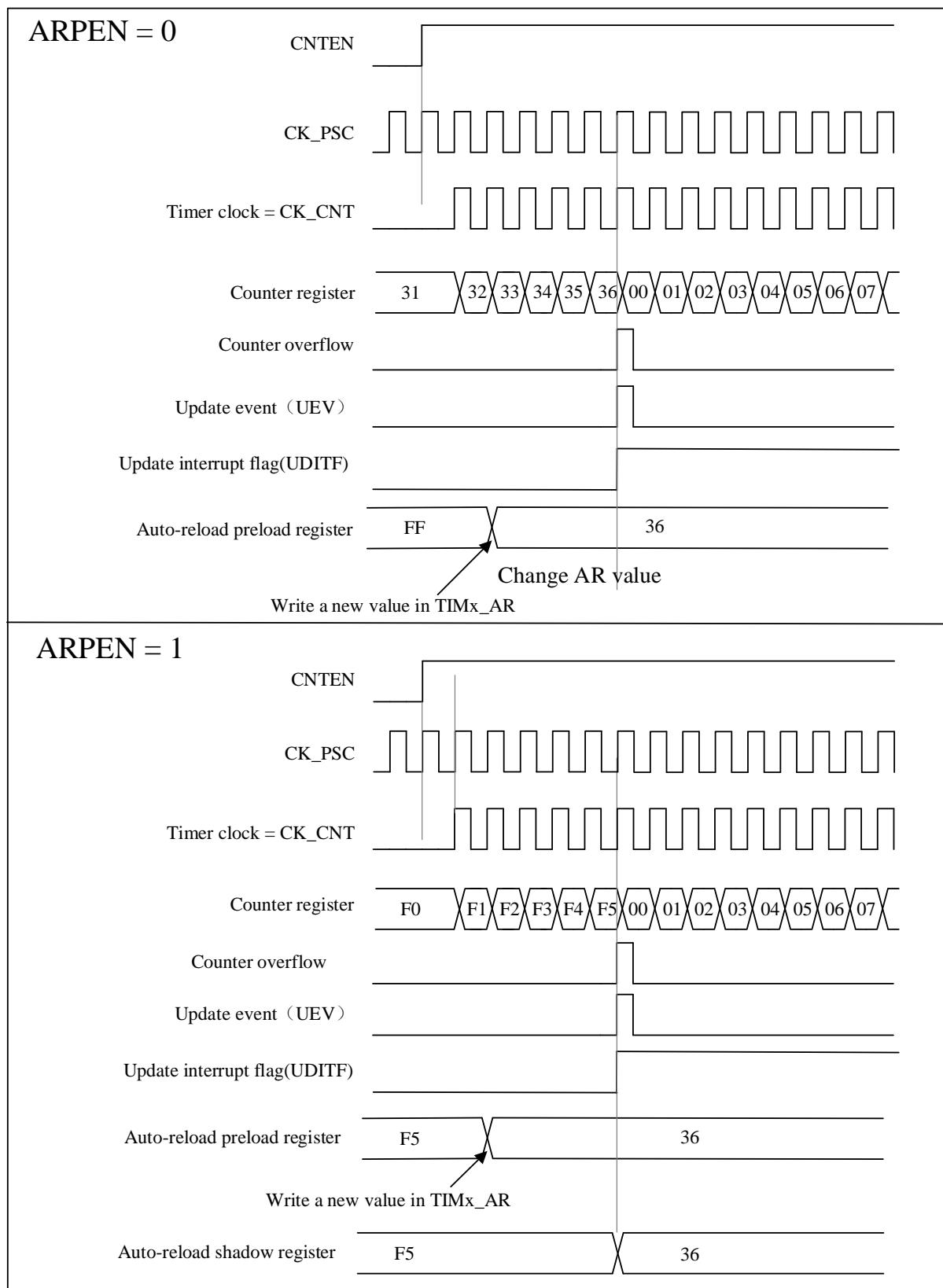


图 11-4 ARPEN=0/1 时向上计数、更新事件的时序图



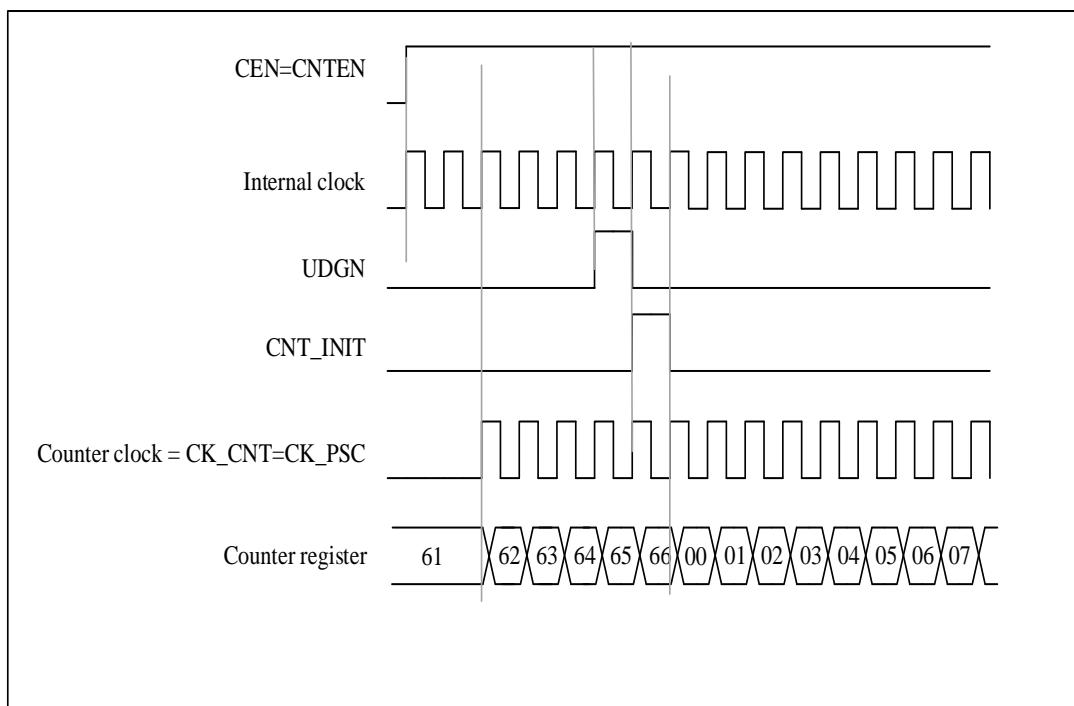
### 11.3.3 时钟选择

- 定时器内部时钟: CK\_INT

#### 11.3.3.1 内部时钟源 (CK\_INT)

前提是 TIMx\_CTRL1.CNTEN 位由软件写为'1', 预分频器的时钟源由内部时钟 CK\_INT 提供。

图 11-5 正常模式下的控制电路, 内部时钟分频系数为 1



### 11.3.4 调试模式

当微控制器处于调试模式 (Cortex-M4 内核停止) 时, 根据 DBG.TIMx\_STOP 位配置, TIMx 计数器可以继续正常工作或停止。有关详细信息, 请参阅 23.4.3 节。

## 11.4 TIMx 寄存器描述(x=6)

有关寄存器中使用的缩写, 请参阅第 1.1 节

这些外设寄存器可以作为半字 (16 位) 或一个字 (32 位) 操作。

## 11.4.1 寄存器总览

表 11-1 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h	TIMx_CTRL1																																
	Reset Value																																
004h																																	
008h																																	
00Ch	TIMx_DINTEN																																
	Reset Value																																
010h	TIMx_STS																																
	Reset Value																																
014h	TIMx_EVTGEN																																
	Reset Value																																
018h																																	
01Ch																																	
020h																																	
024h	TIMx_CNT																																
	Reset Value																																
028h	TIMx_PSC																																
	Reset Value																																
02Ch	TIMx_AR																																
	Reset Value																																

## 11.4.2 控制寄存器 1 (TIMx\_CTRL1)

地址偏移: 0x00

复位值: 0x0000

15	Reserved	8	7	6	4	3	2	1	0
			ARPEN	Reserved		ONEPM	UPRS	UPDIS	CNTEN

位域	名称	描述
15:8	Reserved	保留，必须保持复位值

位域	名称	描述
7	ARPEN	自动重装载预装载允许位（Auto-reload preload enable） 0: TIMx_AR 寄存器的影子寄存器禁用 1: TIMx_AR 寄存器的影子寄存器使能
6:4	Reserved	保留，必须保持复位值
3	ONEPM	单脉冲模式（One pulse mode） 0: 禁用单脉冲模式，发生更新事件时不影响计数器计数。 1: 使能单脉冲模式，计数器在下一次更新事件发生时停止计数（清 TIMx_CTRL1.CNTEN 位）。
2	UPRS	更新请求源（Update request source） 该位用于通过软件选择 UEV 事件源。 0: 如果更新中断或 DMA 请求使能，以下任何事件都会产生更新中断或 DMA 请求： – 计数器溢出 – TIMx_EVTGEN.UDGN 位被设置 1: 如果更新中断或 DMA 请求使能，只有计数器溢出会产生更新中断或 DMA 请求
1	UPDIS	禁止更新（Update disable） 该位用于启用/禁用软件生成的更新事件（UEV）事件。 0: 启用UEV。如果满足以下条件之一，将生成UEV： – 计数器溢出 – TIMx_EVTGEN.UDGN 位被设置 影子寄存器将使用预加载值进行更新。 1: UEV禁用。不生成更新事件，影子寄存器（AR、PSC）保持其值。如果设置了 TIMx_EVTGEN.UDGN位，则重新初始化计数器和预分频器。
0	CNTEN	使能计数器（Counter enable） 0: 禁止计数器； 1: 使能计数器。

#### 11.4.3 DMA/中断使能寄存器 (TIMx\_DINTEN)

地址偏移: 0x0C

复位值: 0x0000

Bitmask diagram for the UDEN field:

- Bits 15, 8, 7, and 1 are reserved.
- Bits 9 through 1 are labeled "UIEN".
- The width of the field is 8 bits.
- Labels "rw" are positioned below the bit 9 and bit 1 columns.

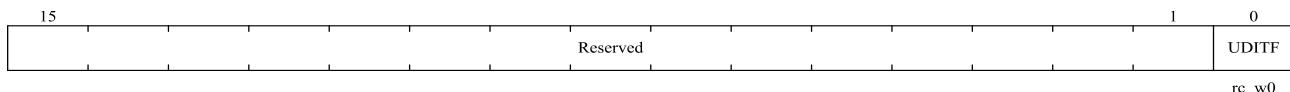
位域	名称	描述
15:9	Reserved	保留, 必须保持复位值
8	UDEN	更新DMA请求使能 (Update DMA request enable) 0: 禁止更新DMA请求 1: 使能更新DMA请求
7:1	Reserved	保留, 必须保持复位值

位域	名称	描述
0	UIEN	更新中断使能 (Update interrupt enable) 0: 禁止更新中断 1: 使能更新中断

#### 11.4.4 状态寄存器 (TIMx\_STS)

地址偏移: 0x10

复位值: 0x0000

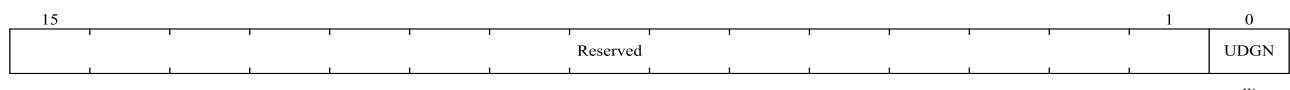


位域	名称	描述
15:1	Reserved	保留, 必须保持复位值
0	UDITF	更新中断标志 (Update interrupt flag) 当在以下条件下发生更新事件时, 该位由硬件设置: – 当 TIMx_CTRL1.UPDIS = 0 且计数器值溢出时。 – 当 TIMx_CTRL1.UPRS = 0 时, TIMx_CTRL1.UPDIS = 0, 并通过软件设置 TIMx_EVTGEN.UDGN 位以重新初始化 CNT。 该位由软件清零。 0: 未发生更新事件 1: 发生更新中断

#### 11.4.5 事件产生寄存器 (TIMx\_EVTGEN)

地址偏移: 0x14

复位值: 0 x0000

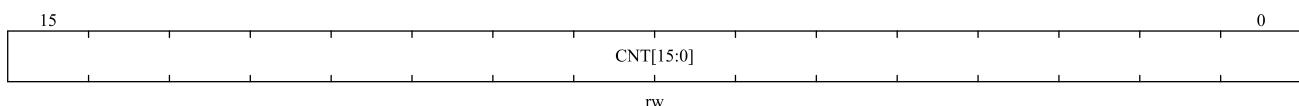


位域	名称	描述
15: 1	Reserved	保留, 必须保持复位值
0	UDGN	产生更新事件 (Update generation) 软件可以设置该位来更新配置寄存器的值, 硬件会自动清除它。 0: 无效果。 1: 定时器计数器将重新启动, 所有影子寄存器将被更新。它也将重新启动预分频器计数器。

#### 11.4.6 计数器 (TIMx\_CNT)

地址偏移: 0x24

复位值: 0x0000

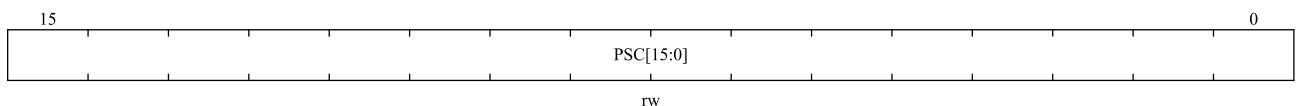


位域	名称	描述
15:0	CNT[15:0]	计数器数值 (Counter value)

#### 11.4.7 预分频器 (TIMx\_PSC)

地址偏移: 0x28

复位值: 0x0000

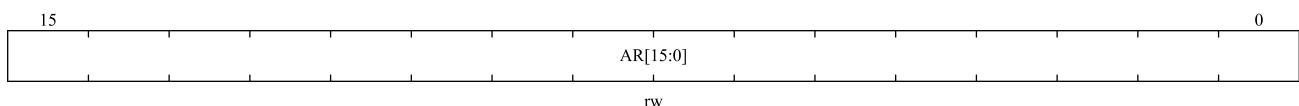


位域	名称	描述
15:0	PSC[15:0]	预分频器数值 (Prescaler value) PSC寄存器值将在更新事件时更新到预分频器寄存器。计数器时钟频率是输入时钟分频 PSC+1。

#### 11.4.8 自动重装载寄存器 (TIMx\_AR)

地址偏移: 0x2C

复位值: 0xFFFF



位域	名称	描述
15:0	AR[15:0]	自动重装载数值 (Auto-reload value) 这些位定义将加载到实际自动重载寄存器中的值。 有关详细信息, 请参阅 11.3.1。 当TIMx_AR.AR [15:0]值为空时, 计数器不工作。

## 12 低功耗定时器 (LPTIM)

### 12.1 简介

LPTIM 是一个具有多个时钟源的 16 位定时器，它可以在除 Standby 模式之外的所有功耗模式下保持运行。LPTIM 可以在没有内部时钟源的情况下运行，可以用作“脉冲计数器”。此外，LPTIM 可以将系统从低功耗模式唤醒，以极低的功耗实现“超时功能”。

### 12.2 主要特性

- 16 位向上计数器
- 3bit 预分频，8 种分频因子（1、2、4、8、16、32、64、128）
- 多个时钟源

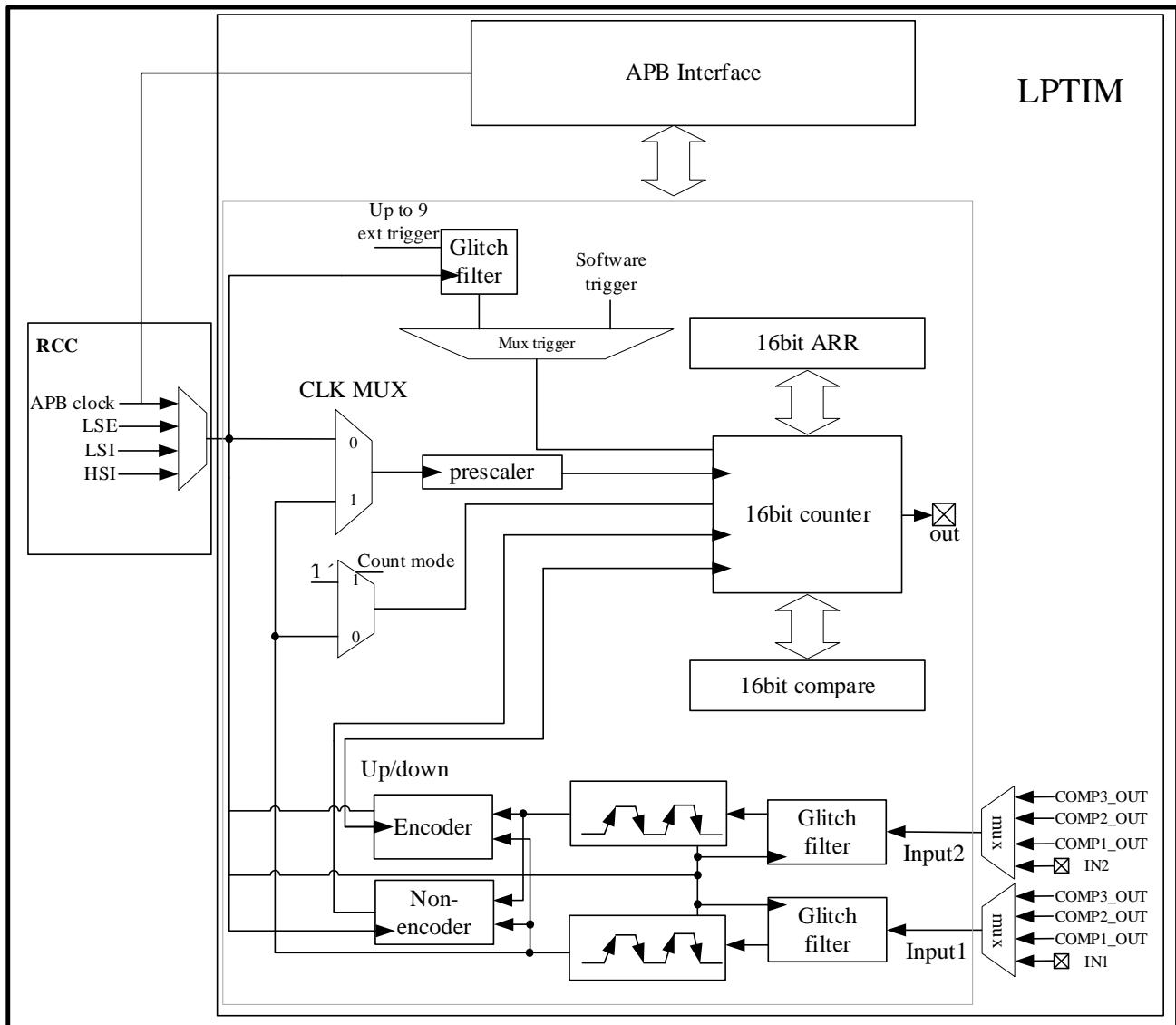
内部时钟源：LSE, LSI, HSI 或者 APB1 时钟

外部时钟源：通过 LPTIM Input1 输入的外部时钟源（工作时无 LP 振荡器运行，用于脉冲计数器应用）

- 16 bit 自动装载寄存器 (LPTIM\_ARR)
- 16 bit 比较寄存器 (LPTIM\_COMP)
- 连续或单触发计数模式
- 可编程软件或硬件输入触发
- 用于过滤毛刺的可编程数字滤波器
- 可配置输出（方波，PWM）
- 可配置 IO 极性
- 编码器模式
- 脉冲计数模式，支持单脉冲计数、双脉冲计数（正交和非正交）

## 12.3 功能框图

图 12-1 LPTIM 主框图



## 12.4 功能描述

### 12.4.1 LPTIM 复位和时钟

LPTIM 可以使用内部时钟源或外部时钟源。内部时钟源可通过 RCC\_RDCTRL.LPTIMSEL[2:0]位进行配置。外部时钟源可从比较器 1、2、3 或 GPIO 中选择。对于外部时钟源，LPTIM 有两种配置：

- LPTIM 使用外部时钟和内部时钟
- LPTIM 仅使用来自比较器或 Input1 的外部时钟。此配置适用于低功耗应用。

LPTIM\_CFG.CLKSEL 和 LPTIM\_CFG.CNTMEN 位用于时钟源配置。有效时钟沿通过 LPTIM\_CFG.CLKP-

OL[1:0]位进行配置。

LPTIM 仅使用外部时钟源时。它只能选择一个有效时钟沿。LPTIM 只有在使用内部时钟源或同时使用外部和内部时钟源时才能选择两个有效时钟沿。

**注意：**当外部时钟的两个边沿都有效时，*LPTIM* 需要使用内部时钟对外部时钟进行过采样。内部时钟频率应至少比外部时钟频率高 4 倍。

## 12.4.2 分频系数

LPTIM 计数器前面有一个可配置的 2 次幂预分频器。预分频比由 LPTIM\_CFG.CLKPRE[2:0]控制。下表列出了所有可能的分频因子：

表 12-1 预分频因子

控制位	对应的分频因子
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

## 12.4.3 毛刺滤波器

LPTIM 具有用于输入的毛刺滤波器，以消除毛刺并防止意外计数或触发。毛刺滤波器需要一个内部时钟源才能运行。并且时钟源应该在毛刺滤波器启用之前提供。这是保证滤波器正常运行所必需的。

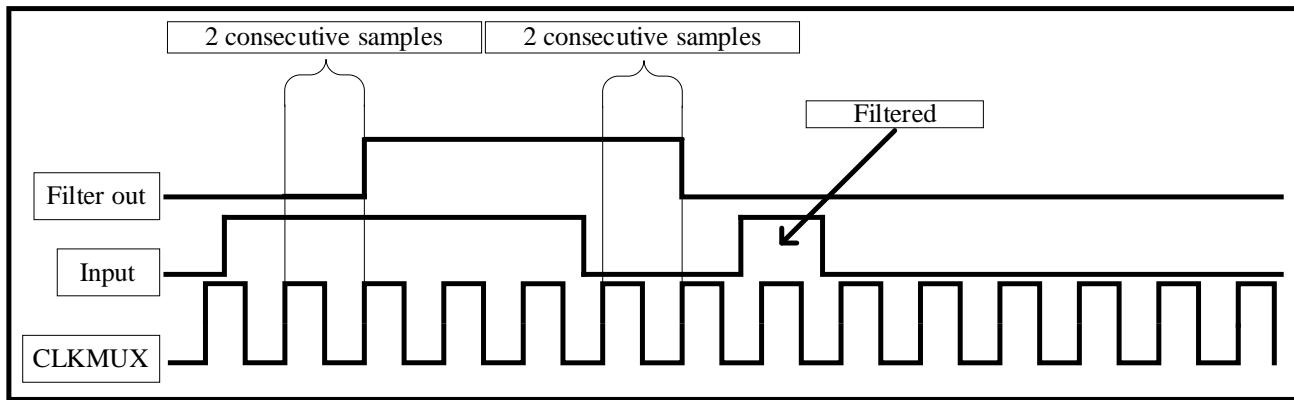
毛刺滤波器有两个主要用途：

- 外部输入：通过 LPTIM\_CFG.CLKFLT[1:0]位配置滤波器灵敏度。
- 内部触发器输入：通过 LPTIM\_CFG.RIGFLT[1:0]位配置滤波器灵敏度。

**注意：**这两组滤波器只对对应的输入有效。

滤波器灵敏度作用于在 LPTIM 输入之一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跳变。图 12-2 展示了当检测到 2 个连续样本时的毛刺滤波器行为。

图 12-2 毛刺滤波器时序图



注意：如果不使用内部时钟，则需要通过清除 `LPTIM_CFG.CLKFLT[1:0]` 和 `LPTIM_CFG.TRIGFLT[1:0]` 位来关闭毛刺滤波器。如果不使用毛刺滤波器，用户可以使用 GPIO 或比较器中的数字滤波器或外部模拟滤波器来消除毛刺。

#### 12.4.4 开启定时器

`LPTIM_CTRL.LPTIMEN` 位用于启用或禁用 LPTIM 内核逻辑。设置 `LPTIM_CTRL.LPTIMEN` 位后，需要延迟两个计数器时钟才能打开 LPTIM。

`LPTIM_CFG` 和 `LPTIM_INTEN` 寄存器的值只能在 LPTIM 关闭时修改。

#### 12.4.5 多路触发器

LPTIM 计数器可以由软件触发启动，也可以由 9 个触发输入之一上的有效边沿触发。触发源通过 `LPTIM_CFG.TRGEN[1:0]` 位进行配置。如果 `LPTIM_CFG.TRGEN[1:0] = '00'`，可以通过设置 `LPTIM_CTRL.TSTCM` 或 `LPTIM_CTRL.SNGMST` 位来触发 LPTIM 启动计数器。`LPTIM_CFG.TRGEN[1:0]` 的其他值用于配置触发的有效边沿。一旦检测到有效边沿，内部计数器将启动。

`LPTIM_CFG.TRGSEL[3:0]` 仅在 `LPTIM_CFG.TRGEN[1:0] ≠ '00'` 时用于选择 9 个触发输入之一。

如果 LPTIM 使用外部触发，将被视为异步触发。对于异步触发，LPTIM 需要两个计数器时钟周期延迟来进行同步。

如果超时功能被禁用，以及 LPTIM 已经启动，新的触发事件将被忽略。

注意：如果 LPTIM 未启用，任何对 `LPTIM_CTRL.SNGMST` 或 `LPTIM_CTRL.TSTCM` 的写入都将被丢弃。

表 12-2 `LPTIM_CFG.TRGSEL[3:0]` 对应的 9 个触发输入

控制位	对应的触发输入
0000	PB6 或 PA10
0001	RTC alarm A
0010	RTC alarm B

0011	RTC_TAMP1
0100	RTC_TAMP2
0101	RTC_TAMP3
0110	COMP1_OUT
0111	COMP2_OUT
1000	COMP3_OUT

## 12.4.6 工作模式

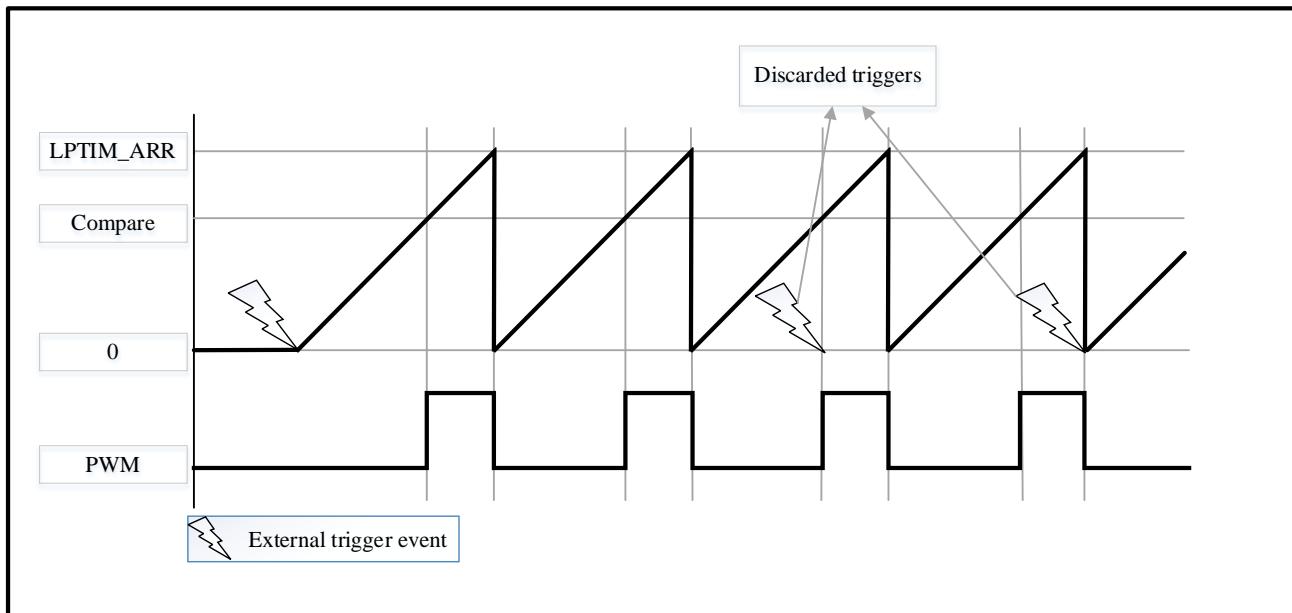
LPTIM 有两种工作模式：

- 连续模式：触发事件将启动 LPTIM 并继续运行，直到用户关闭 LPTIM 时停止。
- 单触发模式：触发事件将启动 LPTIM，并在计数器值达到 LPTIM\_ARR.ARRVAL[15:0]时停止。

### 连续模式：

启用连续模式必须设置 LPTIM\_CTRL.TSTCM 位为 1。如果 LPTIM 使用外部触发，则在设置 LPTIM\_CTRL.TSTCM 位后外部触发事件到达时，内部计数器将启动。连续模式启动后，硬件将丢弃任何后续的外部触发事件。如果使用软件触发，设置 LPTIM\_CTRL.TSTCM 位将启动内部计数器并进入连续模式。任何后续的外部触发事件都将被丢弃，如图 12-3 所示。

图 12-3 LPTIM 输出波形，连续计数模式配置



LPTIM\_CTRL.SNGMST 和 LPTIM\_CTRL.TSTCM 位只能在 LPTIM 开启后设置（LPTIM\_CTRL.LPTIMEN = ‘1’）。

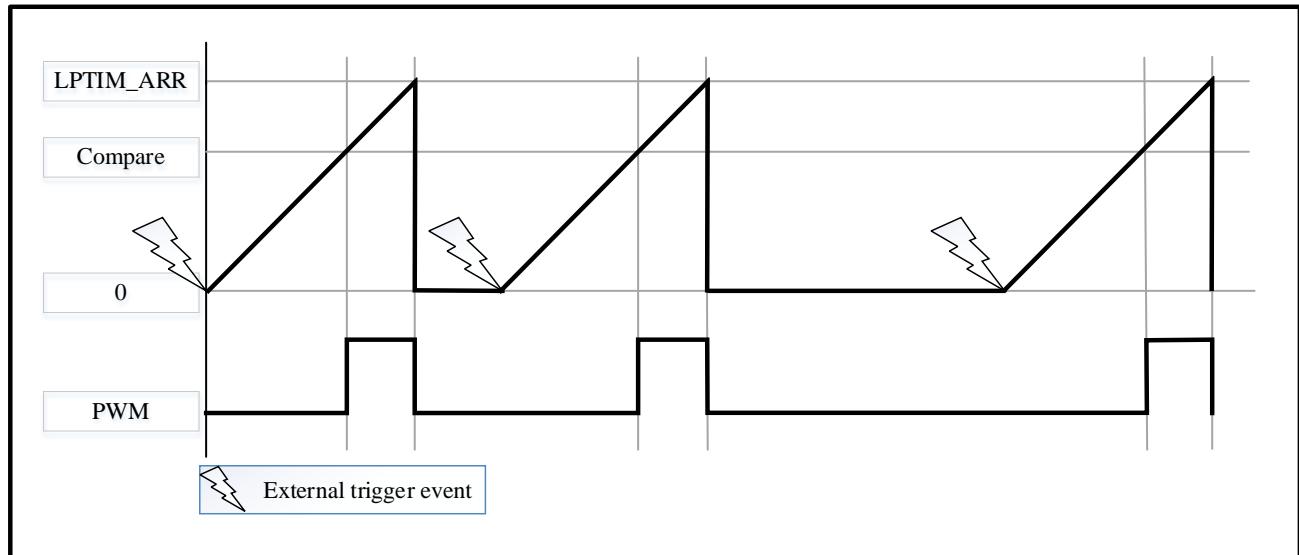
LPTIM 可以从单触发模式切换到连续模式。如果之前选择了连续计数模式，则设置 LPTIM\_CTRL.SNGMST 位会将 LPTIM 切换到单触发模式。如果定时器使能，计数器一旦达到 LPTIM\_ARR 寄存器值就会停止。如果之前选择单触发模式，将 LPTIM\_CTRL.TSTCM 位设置为 1 会将 LPTIM 切换到连续计数模式。如果定时器使能，一旦达到 LPTIM\_ARR 寄存器值，计数器将重新启动。

#### 单触发模式：

启用单触发模式必须设置 LPTIM\_CTRL.SNGMST 位为 1。一个新的触发事件将重新启动 LPTIM。硬件将在内部计数器启动后和计数器值等于 LPTIM\_ARR.ARRVAL[15:0]值之前放弃所有触发事件。

如果选择了外部触发，则在设置 LPTIM\_CTRL.SNGMST 位之后到达的每个外部触发事件，以及在定时器寄存器停止（包含零值）之后，定时器将重新启动一个新的计数周期，如图 12-4 所示。

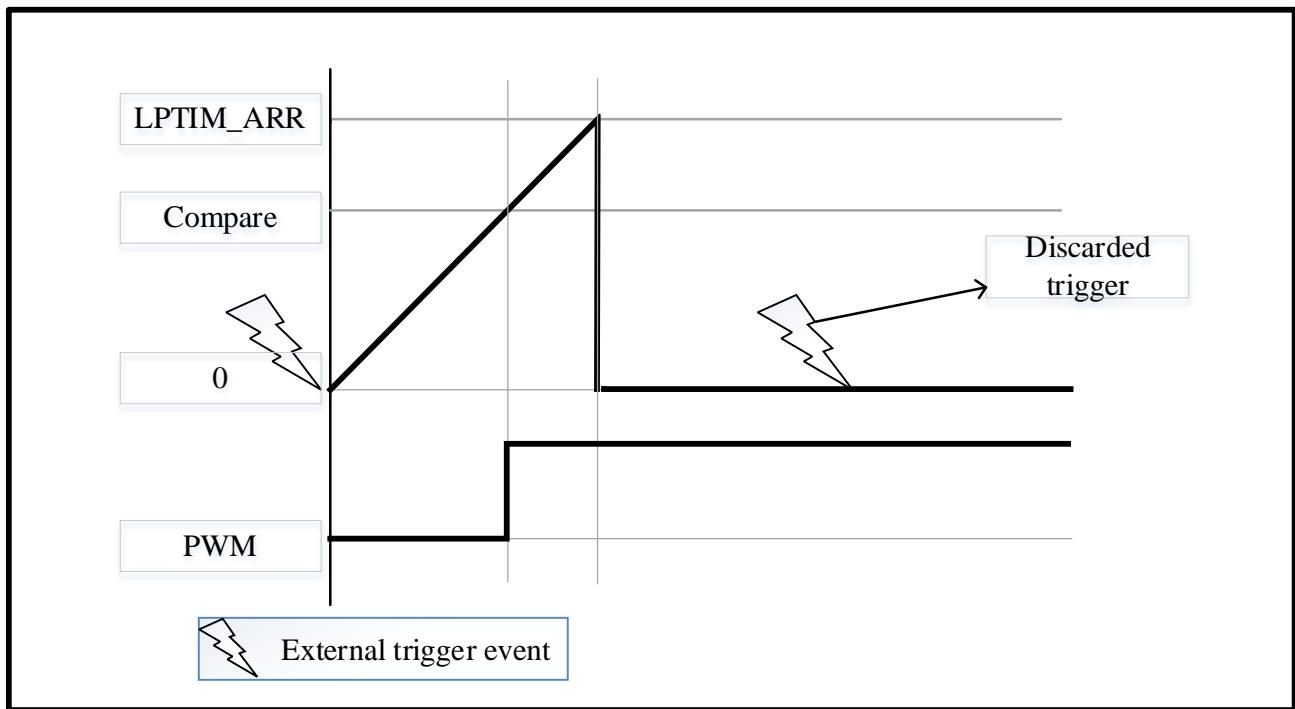
图 12-4 LPTIM 输出波形，单触发计数模式配置



#### 单触发模式：

当置 LPTIM\_CFG.WAVE 位为 1 后使用单触发模式。在单触发模式下，计数器在第一个触发事件发生时启动一次，任何后续触发事件将被硬件丢弃，如图 12-5 所示。

图 12-5 LPTIM 输出波形，一次模式



在软件启动( $(\text{LPTIM\_CFG.TRGEN}[1:0] = 00)$ )的情况下， $\text{LPTIM\_CTRL.SNGMST}$ 位置 1 将启动定时器进行单触发计数。

#### 12.4.7 波形发生器

LPTIM 自动加载寄存器 (LPTIM\_ARR) 和比较寄存器 (LPTIM\_COMP) 用于生成 LPTIM 输出波形。

LPTIM 支持的波形如下所示：

- **PWM 模式：**当发生补偿匹配事件时 LPTIM 输出。(即 LPTIM\_CNT 寄存器值与 LPTIM\_COMP 寄存器值匹配。) 当发生 arr 匹配时，LPTIM 输出被复位。(即 LPTIM\_CNT 寄存器值与 LPTIM\_ARR 寄存器值匹配。)
- **单脉冲模式：**被触发的第一个脉冲与 PWM 波形相同，发生 arr 匹配时永久复位输出。
- **一次模式：**输出波形类似于单脉冲模式，只是输出保持在最后一个信号电平(取决于输出配置的极性)。

上述波形配置要求 LPTIM\_ARR 寄存器值必须配置为大于 LPTIM\_COMP 寄存器值。

LPTIM 输出波形可以通过 LPTIM\_CFG.WAVE 位配置如下：

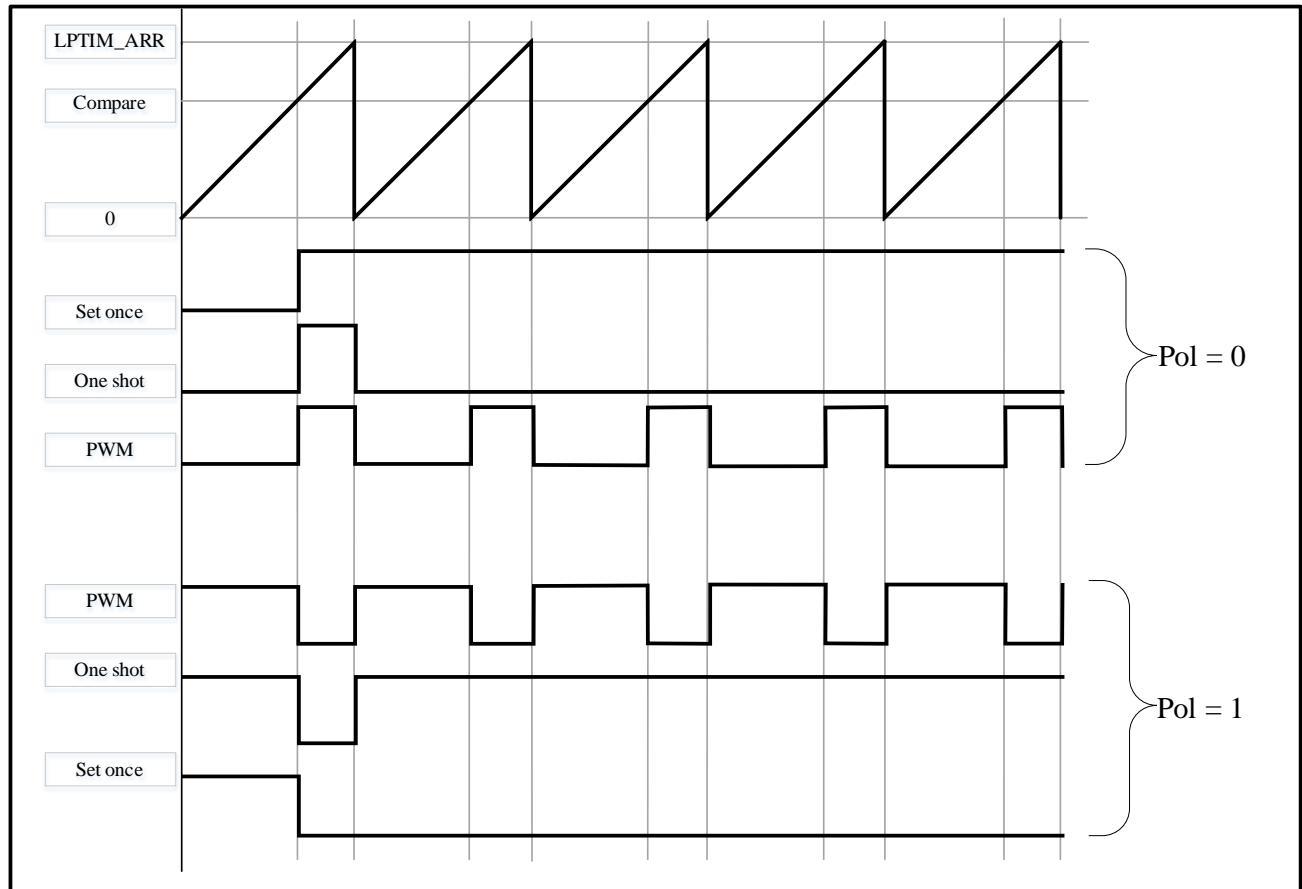
- 清除 LPTIM\_CFG.WAVE 位将强制 LPTIM 根据 LPTIM\_CTRL.TSTCM 或 LPTIM\_CTRL.SNGMST 位的值来生成 PWM 波形或者单脉冲波形。
- LPTIM\_CTRL.WAVE 置 1 将强制 LPTIM 生成一次模式波形。

LPTIM\_CFG.WAVEPOL 位控制 LPTIM 输出的极性。即使定时器被关闭，用户配置极性后，输出空闲稳定电平将立即改变。

可以生成频率高达 LPTIM 时钟频率除以 2 的信号。只有当 LPTIM 计数器计数外部时钟有效时，才能实现时钟频率除以 2。(例如 LPTIM\_CFG.CLKSEL = 0, LPTIM\_CFG.CLKPOL[1:0] = 10, LPTIM\_COMP.CMPVAL[15:0] = 'd1(50%占空比)/'d2, LPTIM\_ARR.ARRVAL[15:0] = 'd3。d1、d2、d3 分别表示十进制 1、2、3)

图 12-6 展示了 LPTIM 输出上生成的三种可能波形，另外也表明可以用 LPTIM\_CFG.WAVEPOL 位来设置极性变化。

图 12-6 波形发生器



## 12.4.8 寄存器更新

LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器在软件写操作后立即更新。如果 LPTIM 已经启动，LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器在计数器溢出时更新。

LPTIM APB 接口和 LPTIM 内核逻辑使用的是不同的时钟，因此在 APB 写操作后，需要经过一定的延迟，写入值才能用于计数器以及比较器。在此延迟时间内，必须避免对这些寄存器进行任何额外的写操作。

LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器的更新方式由 LPTIM\_CFG.RELOAD 决定：

- LPTIM\_CFG.RELOAD = 1：如果 LPTIM 已经启动了，LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器在计数器溢出时更新。在计数器溢出时，延迟时间 = 2~3 个 APB 时钟周期
- LPTIM\_CFG.RELOAD = 0：LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器在软件写访问后更新。延迟

时间 = 2~3 个 APB 时钟周期 + 2~3 个 LPTIM 内部分频时钟周期。

LPTIM\_INTSTS.ARRUPD 标志位和 LPTIM\_INTSTS.CMPUPD 标志位分别指示何时完成对 LPTIM\_ARR 寄存器和 LPTIM\_COMP 寄存器的写操作。

在对 LPTIM\_ARR 寄存器或 LPTIM\_COMP 寄存器进行写操作之后，任何 LPTIM\_INTSTS.ARRUPD 标志位和 LPTIM\_INTSTS.CMPUPD 标志位置 1 之前的后续写操作都将导致不可预知的结果。所以只能在前一次写操作完成后才能对同一寄存器进行新的写操作。

## 12.4.9 计数器模式

内部计数器可以对来自 LPTIM Input1 或内部时钟周期的外部触发事件进行计数。这可以通过 LPTIM\_CFG.CLKSEL 和 LPTIM\_CFG.CNTMEN 位进行配置。

如果 LPTIM 正在计数外部触发，用户可以配置 LPTIM\_CFG.CLKPOL[1:0]位来选择上升沿、下降沿或上下沿为有效沿。可以选择以下计数模式，具体取决于 LPTIM\_CFG.CLKSEL 和 LPTIM\_CFG.CNTMEN：

- LPTIM\_CFG.CLKSEL = 0: LPTIM 使用内部时钟源来提供时钟。
  - LPTIM\_CFG.CNTMEN = 0, LPTIM 配置为由内部时钟源提供时钟，LPTIM 计数器配置为在每个内部时钟脉冲后更新。
  - LPTIM\_CFG.CNTMEN = 1, 使用提供给 LPTIM 的内部时钟对 LPTIM 外部 Input1 进行采样。为了不错过任何事件，外部 Input1 信号的变化频率不得超过提供给 LPTIM 的内部时钟频率。此外，不得对提供给 LPTIM 的内部时钟进行预分频(LPTIM\_CFG.CLKPRE[2:0] = 000)。
- LPTIM\_CFG.CLKSEL = 1: LPTIM 使用外部时钟源来提供时钟。
  - LPTIM\_CFG.CNTMEN 位的值是不相关的。在这个配置中，LPTIM 不需要内部时钟源(除非启用了毛刺滤波器)。在 LPTIM 外部 Input1 上注入的信号用作 LPTIM 的系统时钟。这种配置适用于未启用嵌入式振荡器的操作模式。
  - 对于这种配置，LPTIM 计数器可以在 Input1 时钟信号的上升沿或下降沿进行更新，但不能同时在上升沿和下降沿更新。
  - 由于在 LPTIM 外部 Input1 上注入的信号也用于对 LPTIM 内核逻辑进行计时，所以在计数器递增之前存在一些初始延迟(启用 LPTIM 之后)。更准确地说，LPTIM 外部 Input1 上的前 2 到 5 个触发沿(在启用 LPTIM 之后)将丢失。

## 12.4.10 编码器模式

编码器模式可以处理来自正交编码器的信号，用于检测旋转元件的角位置。编码器模式允许计数器对 0 和 LPTIM\_ARR.ARRVAL[15:0]值内的事件进行计数 (0 到 LPTIM\_ARR.ARRVAL[15:0]或 LPTIM\_ARR.ARRVAL[15:0]到 0)。在这种情况下，用户必须在启用计数器之前配置 LPTIM\_ARR.ARRVAL[15:0]。Input1 和 Input2 为计数器生成一个时钟信号。计数方向取决于这两个输入信号之间的相位。

编码器模式仅在 LPTIM 由内部时钟源提供时钟时可用。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟频率的 4 分频。这是强制性的，以保证 LPTIM 的正常运行。

计数方向的变化由 LPTIM\_INTSTS 寄存器中的两个 Down 和 Up 标志更新。此外，可以通过设置 LPTIM\_INTEN.DOWNIE 和 LPTIM\_INTEN.UPIE 位为两个方向改变事件生成中断。

用户可以通过设置 LPTIM\_CFG.ENC 位来启用编码器模式。并且 LPTIM 需要首先配置为连续模式。

当编码器模式激活时，LPTIM 计数器会根据增量编码器的速度和方向自动修改计数值。因此，它的内容总是代表编码器的位置。由向上和向下标志指示的计数方向对应于编码器转子的旋转方向。

使用 LPTIM\_CFG.CLKPOL[1:0]位配置为不同的触发沿，可能会有不同的计数场景。下表总结了可能的组合，假设 Input1 和 Input2 不同时切换。

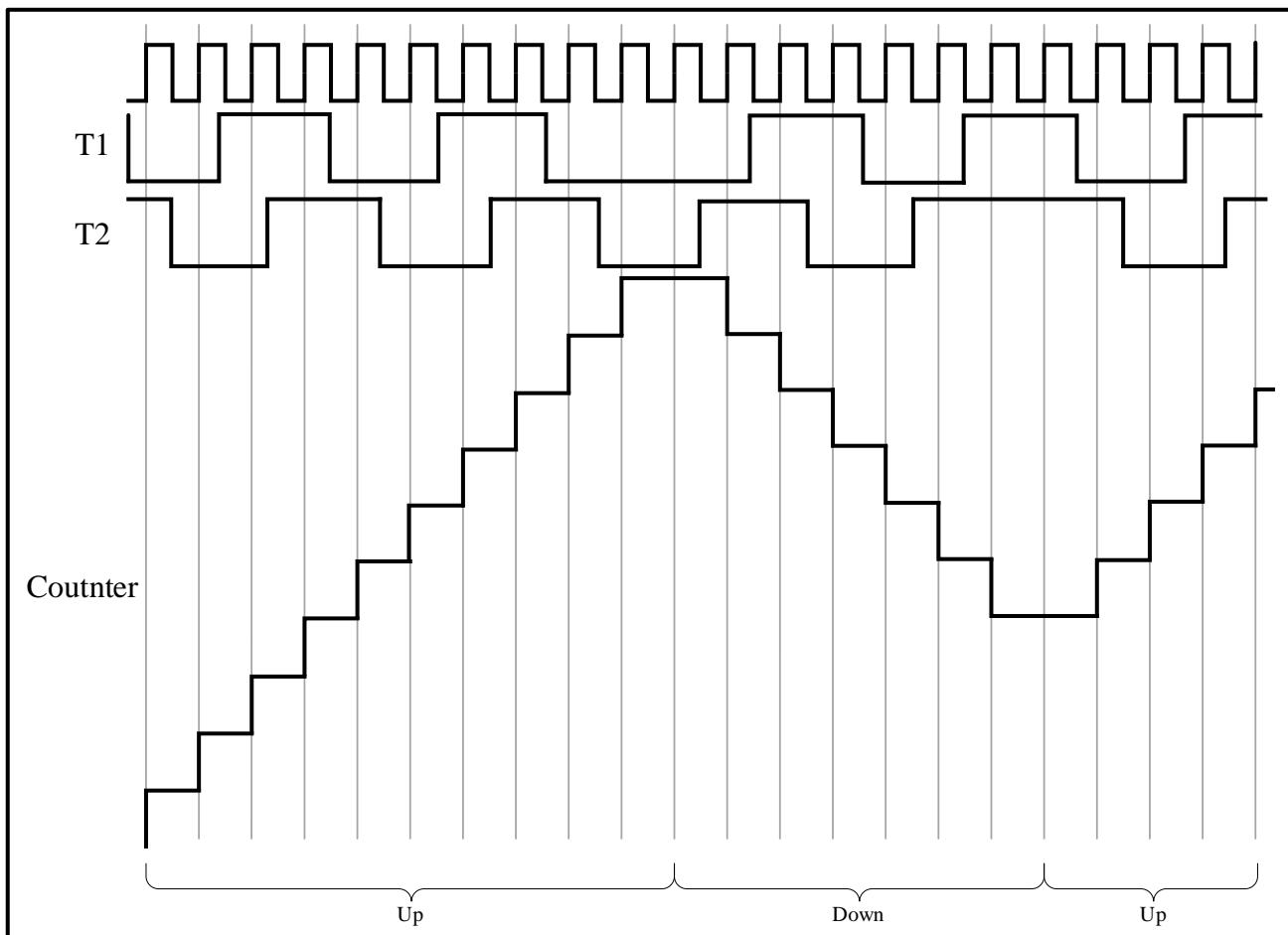
表 12-3 编码器计数的场景

触发沿	信号相反(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	High	Down	未计数	Up	未计数
	Low	Up	未计数	Down	未计数
下降沿	High	未计数	Up	未计数	Down
	Low	未计数	Down	未计数	Up
上下沿	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

下图显示了配置了上下沿触发的编码器模式的计数序列。

**注意：**在此模式下，LPTIM 必须由内部时钟源提供时钟，因此 LPTIM\_CFG.CLKSEL 位必须保持其复位值等于“0”。此外，预分频器分频比必须等于其复位值 1 (LPTIM\_CFG.CLKPREF[2:0]位必须为“000”）。

图 12-7 编码器模式计数序列



#### 12.4.11 非正交编码器模式

此模式允许处理来自非正交编码器的信号，用于检测来自外部接口的后续正脉冲。非正交编码器接口模式仅用作具有方向选择的外部时钟。这意味着计数器只是在 0 和编程到 LPTIM\_ARR 寄存器中的自动重载值之间连续计数（0 到 LPTIM\_ARR.ARRVAL[15:0] 或 LPTIM\_ARR.ARRVAL[15:0] 到 0，具体取决于方向）。因此，您必须在开始之前配置 LPTIM\_ARR。从两个外部输入信号 Input1 和 Input2 生成时钟信号来为 LPTIM 计数器计时。这两个信号之间的顺序决定了计数方向。

非编码器模式仅在 LPTIM 由内部时钟源提供时钟时可用。Input1 和 Input2 输入上的信号频率不得超过 LPTIM 内部时钟频率的 4 分频。这是强制性的，以保证 LPTIM 正常运行。

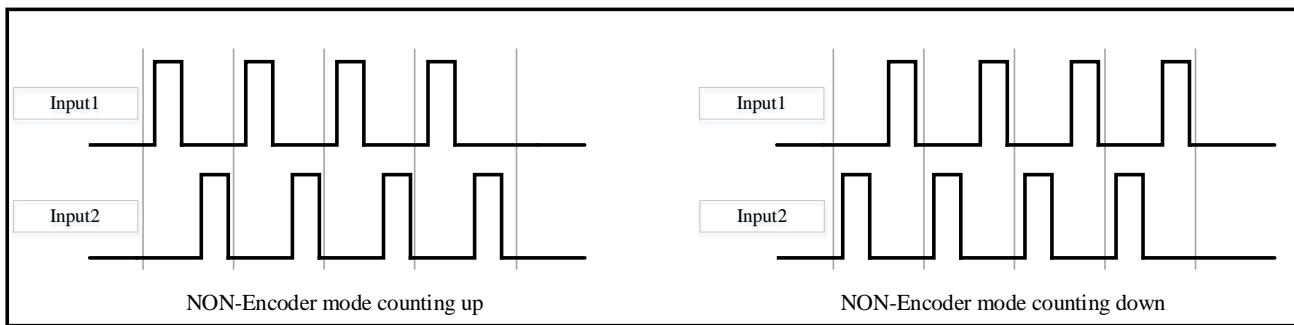
LPTIM\_INTSTS 寄存器中的两个向下和向上标志指示方向变化。此外，可以通过设置 LPTIM\_INTEN.DOWNIE 和 LPTIM\_INTEN.UPIE 位为两个方向改变事件生成中断。

要激活非编码器模式，LPTIM\_CFG.NENC 位必须设置为“1”。LPTIM 必须首先配置为连续模式。

当非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。由向上和向下标志指示的计数方向对应于编码器转子的旋转方向。

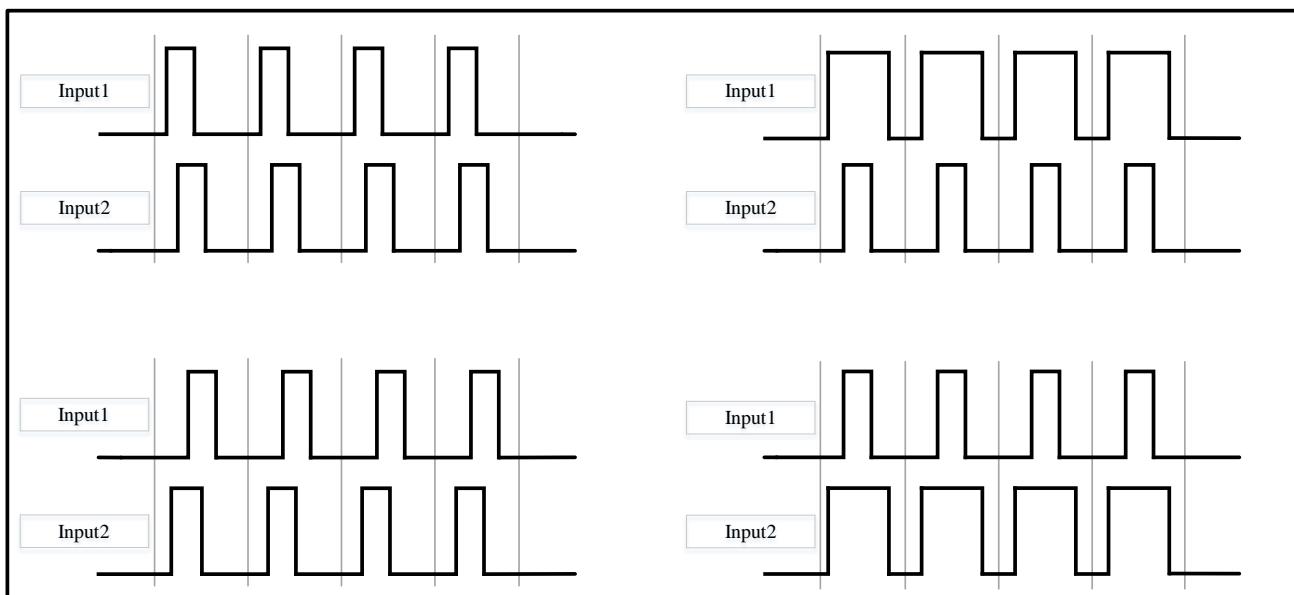
如图 12-8，只要 Input1 和 Input2 不都是高电平，解码器模块都可以正确工作。

图 12-8 非正交编码器正常工作 Input1、Input2 波形



如果 Input1 和 Input2 的波形如图 12-9 所示，则解码模块无法正常工作。计数器将忽略这些波形并保持先前的值。

图 12-9 非正交编码器非正常工作 Input1、Input2 波形



### 12.4.12 超时功能

当 LPTIM\_CFG.TIMOUTEN 位使能时，LPTIM 计数器将由一个选定触发输入的有效边沿复位。

当使用超时功能时，LPTIM 计数器将被选定的触发输入事件复位并重新启动。如果在配置的时间内没有触发，就会发生比较匹配事件。等待时间通过超时值配置。

### 12.4.13 LPTIM 中断

通过 LPTIM\_INTEN 寄存器启用以下事件，则会生成中断/唤醒事件：

- 比较匹配
- 自动重载匹配(无论方向是否是编码器模式)
- 外部触发事件

- 自动重载寄存器更新成功
- 比较寄存器更新成功
- 方向改变(编码器模式), 可编程(上/下/两边)

**注意:** 如果 LPTIM\_INTEN 寄存器(中断使能寄存器)中的任何位在 LPTIM\_INTSTS 寄存器(状态寄存器)中相应的标志被置 1 后才被设置, 将不能产生相应的中断。

表 12-4 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_COMP)的内容匹配时, 将触发中断标志 LPTIM_INSTS.CMPM
自动从重匹配	当计数器寄存器(LPTIM_CNT)的内容与自动重新加载寄存器(LPTIM_ARR)的内容匹配时, 将触发中断标志 LPTIM_INSTS.ARRM
外部触发事件	当检测到外部触发事件时, 将触发中断标志 LPTIM_INSTS.EXTRIG
重装寄存器更新成功	当 LPTIM_ARR 寄存器执行完成写操作时, 将触发中断标志 LPTIM_INSTS.CMPUPD
比较寄存器更新成功	当 LPTIM_COMP 寄存器执行完成写操作时, 将触发中断标志 LPTIM_INSTS.ARRUPD
方向改变	用于编码器模式。两个中断标志被嵌入到信号中 方向切换: -LPTIM_INSTS.Up 标志信号向上计数方向改变 -LPTIM_INSTS.Down 标志向下计数方向改变

## 12.5 LPTIM 寄存器

### 12.5.1 LPTIM 寄存器总览

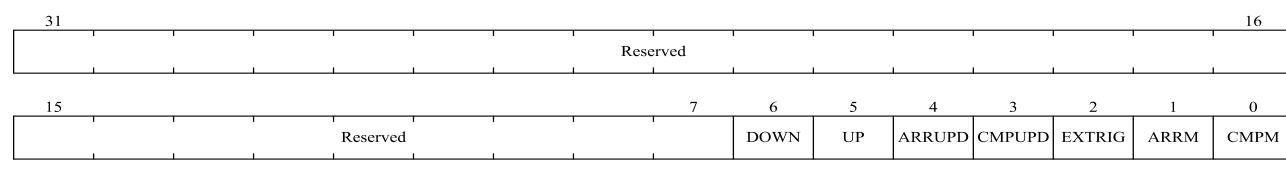
表 12-5 LPTIM 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
000h	LPTIM_INTSTS																																		
		Reset Value																																	
004h	LPTIM_INTCLR																																		
		Reset Value																																	
008h	LPTIM_INTEN																																		
		Reset Value																																	
00Ch	LPTIM_CFG																																		
		Reset Value																																	
010h	LPTIM_CTRL																																		
		Reset Value																																	
014h	LPTIM_COMP																																		
		Reset Value																																	
018h	LPTIM_ARR																																		
		Reset Value																																	
01Ch	LPTIM_CNT																																		
		Reset Value																																	
020h	LPTIM_OPT																																		
		Reset Value																																	

### 12.5.2 LPTIM 中断状态寄存器(LPTIM\_INTSTS)

偏移地址: 0x00

复位值: 0x0000 0000



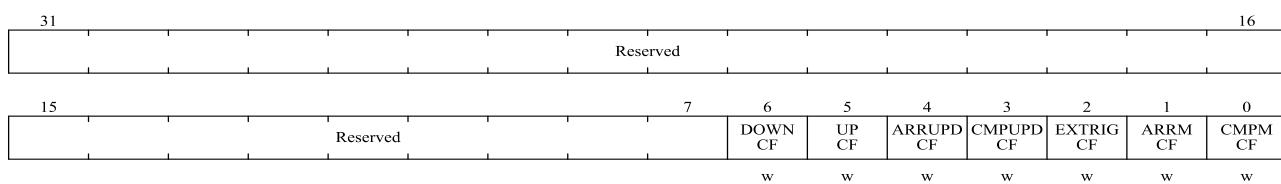
位域	名称	描述
31:7	Reserved	保留, 必须保持复位值。
6	DOWN	计数器方向从递增变为递减。

位域	名称	描述
		在编码器模式, 由硬件置 1, 以通知应用程序, 计数器方向由递增变为递减
5	UP	计数器方向从递减变为递增。 在编码器模式, 由硬件置 1, 以通知应用程序, 计数器方向由递减变为递增
4	ARRUPD	自动重装寄存器更新成功。 由硬件置 1, 以通知应用程序 APB 总线对 LPTIM_ARR 寄存器的写操作已经成功完成。 详细内容请查阅 12.4.8。
3	CMPUPD	比较器寄存器更新完成。 由硬件置 1, 以通知应用程序 APB 总线对 LPTIM_COMP 寄存器的写操作已经成功完成。 详细内容请查阅 12.4.8。
2	EXTRIG	外部触发事件 由硬件置 1, 以通知应用程序所选外部触发器已经输入有效触发边沿。如果因为计数器已经忽略触发器, 则不将此标志位置 1。
1	ARRM	自动重装匹配。 由硬件置 1, 以通知应用程序 LPTIM_CNT 寄存器的值已经达到 LPTIM_ARR 寄存器的值。
0	CMPM	比较器匹配。 由硬件置 1, 以通知应用程序 LPTIM_CNT 寄存器的值已经达到 LPTIM_COMP 寄存器的值。

### 12.5.3 LPTIM 中断清除寄存器(LPTIM\_INTCLR)

偏移地址: 0x04

复位值: 0x0000 0000



位域	名称	描述
31: 7	Reserved	保留, 必须保持复位值。
6	DOWNCF	计数器方向从递增变为递减标志清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 DOWN 标志
5	UPCF	计数器方向从递减变为递增标志清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 UP 标志
4	ARRUPDCF	自动重装寄存器更新成功清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 ARRUPD 标志
3	CMPUPDCF	比较器寄存器更新成功清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 CMPUPD 标志

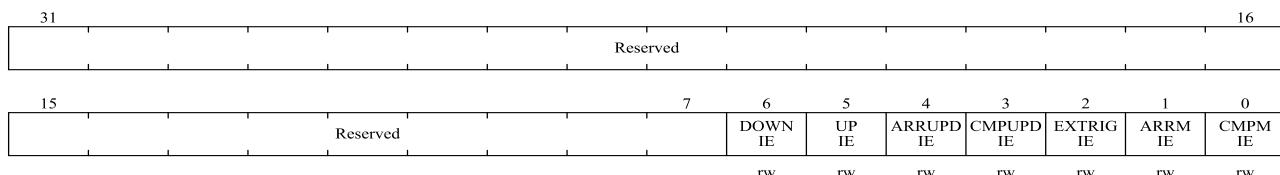
位域	名称	描述
2	EXTRIGCF	外部触发沿事件清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 EXTRIG 标志
1	ARRMCF	自动重装匹配清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 ARRM 标记
0	CMPMCF	比较器匹配清除位。 将 1 写入这个位清除 LPTIM_INTSTS 寄存器中的 CMPM 标志

## 12.5.4 LPTIM 中断使能寄存器(LPTIM\_INTEN)

偏移地址: 0x08

复位值: 0x0000 0000

注意: LPTIM\_INTEN 寄存器只能在 LPTIM 不工作的时候修改(LPTIM\_CTRL.LPTIMEN = '0')



位域	名称	描述
31:7	Reserved	保留, 必须保持复位值。
6	DOWNIE	计数器方向从递增变为递减中断使能位。 0: 计数器方向变化, 向下计数中断禁止 1: 计数器方向变化, 向下计数中断使能
5	UPIE	计数器方向从递减变为递增中断使能位。 0: 计数器方向变化, 向上计数中断禁止 1: 计数器方向变化, 向上计数中断使能
4	ARRUPDIE	自动重装寄存器更新成功中断使能位。 0: 自动重装寄存器更新成功中断禁止 1: 自动重装寄存器更新成功中断使能
3	CMPUPDIE	比较器寄存器更新成功中断使能位。 0: 比较器寄存器更新成功中断禁止 1: 比较器寄存器更新成功中断使能
2	EXTRIGIE	外部触发事件中断使能位。 0: 外部触发事件中断禁止 1: 外部触发事件中断使能
1	ARRMIE	自动重装匹配中断使能位。 0: 自动重装匹配中断禁止 1: 自动重装匹配中断使能
0	CMPMIE	比较器匹配中断使能位。 0: 比较器匹配中断禁止 1: 比较器匹配中断使能

## 12.5.5 LPTIM 配置寄存器(LPTIM\_CFG)

偏移地址: 0x0C

复位值: 0x0000 0000

注意: LPTIM\_CFG 寄存器只能在 LPTIM 不工作的时候修改(LPTIM\_CTRL.LPTIMEN = '0')

31	Reserved				NENC	ENC	CNTMEN	RELOAD	WAVE POL	WAVE	TIMOUT EN	TRGEN[1:0]	TRG SEL[3]
15	13	12	11		rw 9	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1
TRGSEL[2:0]	Reserved	CLKPRE[2:0]		Reserved	TRIGFLT[1:0]	Reserved	CLKFLT[1:0]	CLKPOL[1:0]	CLKSEL				
	rw			rw		rw		rw		rw		rw	rw

位域	名称	描述
31:26	Reserved	保留, 必须保持复位值。
25	NENC	非正交编码器模式使能位。 0: 非正交编码器禁止 1: 非正交编码器使能
24	ENC	编码器模式使能位。 0: 编码器禁止 1: 编码器使能
23	CNTMEN	计数器模式使能位。 该位选择 LPTIM 使用哪个时钟源来对计数器进行计时： 0: 计数器根据每个内部时钟脉冲递增 1: 计数器根据 LPTIM 外部 Input1 上的每个有效时钟脉冲递增
22	RELOAD	寄存器更新模式。 该位控制 LPTIM_ARR 和 LPTIM_COMP 寄存器更新方式。 0: 寄存器在每个 APB 总线写访问之后更新 1: 寄存器在当前的 LPTIM 期间结束时更新
21	WAVEPOL	波形极性位。 该为控制输出极性。 0: LPTIM 输出反映 LPTIM_ARR 和 LPTIM_COMP 寄存器之间的比较结果 1: LPTIM 输出反映了 LPTIM_ARR 和 LPTIM_COMP 寄存器之间的比较结果的反相。
20	WAVE	波形形状位。 该位控制输出波形形状。 0: 禁用单触发模式, PWM/单脉冲波形(取决于 LPTIM_CTRL.TSTCM 或 LPTIM_CTRL.SNGMST 位) 1: 激活单触发模式
19	TIMOUTEN	超时使能位。 该位开启超时功能。 0: 计时器已经启动时到达的触发器事件将被忽略 1: 当计时器已经启动时到达的触发事件将复位并重新启动计数器
18:17	TRGEN[1:0]	触发极性使能位。

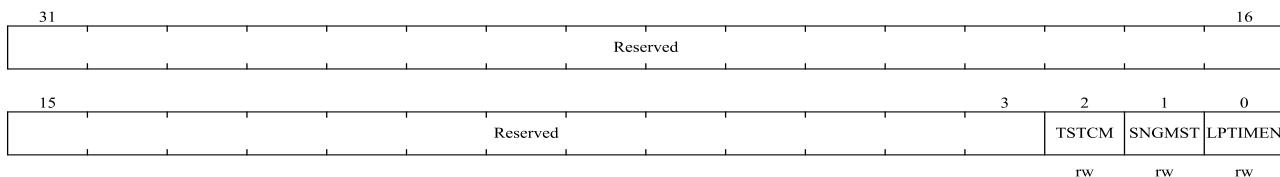
位域	名称	描述
		<p>该位控制 LPTIM 计数器是否由外部触发器启动。如果选择外部触发器选项，触发器触发沿可以有三种配置：</p> <p>00: 软件触发（计数器开始的时候由软件启动）      01: 下降沿触发      10: 上升沿触发      11: 上下沿触发</p>
16:13	TRGSEL[3:0]	<p>触发选择位</p> <p>从以下 9 个触发源中选择其中一个作为 LPTIM 的触发事件：</p> <p>0000: PB6 或 PA10      0001: RTC alarm A      0010: RTC alarm B      0011: RTC_TAMP1      0100: RTC_TAMP2      0101: RTC_TAMP3      0110: COMP1_OUT      0111: COMP2_OUT      1000: COMP3_OUT      其他值保留</p>
12	Reserved	保留，必须保持复位值。
11:9	CLKPRE[2:0]	<p>时钟分频因子</p> <p>000: /1      001: /2      010: /4      011: /8      100: /16      101: /32      110: /64      111: /128</p>
8	Reserved	保留，必须保持复位值。
7:6	TRIGFLT[1:0]	<p>数字滤波器灵敏度配置位。</p> <p>该位设置当内部触发器上发生电平变化时，在将其视为有效的电平转换之前应该检测的连续相等信号的数量。</p> <p>00: 任何电平都可以触发      01: 触发器的触发电平变化必须在至少 2 个时钟周期内保持稳定，才能被视为有效的触发      10: 触发器的触发电平变化必须在至少 4 个时钟周期内保持稳定，才能被视为有效的触发      11: 触发器的触发电平变化必须在至少 8 个时钟周期内保持稳定，才能被视为有效的触发  <b>注意：必须提供内部时钟源才能使用此功能。</b></p>
5	Reserved	保留，必须保持复位值
4:3	CLKFLT[1:0]	外部时钟信号数字滤波器灵敏度配置位。

位域	名称	描述
		<p>该位设置当外部时钟信号发生电平变化时，在将其视为有效电平转换之前应检测的连续相等信号的数量。</p> <p>00: 任何外部时钟电平变化都是有效信号</p> <p>01: 外部时钟电平变化必须在至少 2 个时钟周期内保持稳定，才能被视为有效的信号</p> <p>10: 外部时钟电平变化必须在至少 4 个时钟周期内保持稳定，才能被视为有效的信号</p> <p>11: 外部时钟电平变化必须在至少 8 个时钟周期内保持稳定，才能被视为有效的信号</p> <p><b>注意：必须提供内部时钟源才能使用此功能。</b></p>
2:1	CLKPOL[1:0]	<p>时钟极性位。</p> <p>当 LPTIM 由外部时钟源提供计时，CLKP[1:0]配置计数器使用的触发沿：</p> <p>00: 上升沿用来计数</p> <p>01: 下降沿用来计数</p> <p>10: 上下沿用来计数。</p> <p>11: 上下沿都不用于计数</p> <p><b>注意：当外部时钟信号上升沿和下降沿都是有效出发沿时，LPTIM 还必须由一个内部时钟源进行计时，其频率至少等于外部时钟频率的四倍。</b></p> <p>当 LPTIM 配置为编码器模式(LPTIM_CFG.ENC 位置 1):</p> <p>00: 启用编码器上升沿计数模式</p> <p>01: 启用编码器下降沿计数模式</p> <p>10: 启用编码器双边沿计数模式</p>
0	CLKSEL	<p>时钟源选择位。</p> <p>该位配置 LPTIM 将选择使用的时钟源。</p> <p>00: LPTIM 由内部时钟源(APB 时钟或任何嵌入式振荡器)计时</p> <p>01: LPTIM 由外部时钟源通过 LPTIM 外部 Input1 进行计时</p>

### 12.5.6 LPTIM 控制寄存器(LPTIM\_CTRL)

偏移地址： 0x10

复位值: 0x0000 0000



位域	名称	描述
31:3	Reserved	保留，必须保持复位值。
2	TSTCM	定时器以连续模式启动位。 这个位由软件置 1，由硬件清除。 在软件启动的情况下(LPTIM_CFG.TRGEN[1:0] = 00)，设置这个位以连续模式启

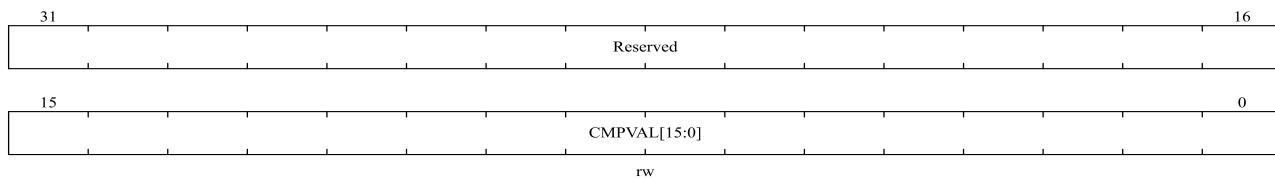
位域	名称	描述
		动 LPTIM。如果软件启动被禁用(LPTIM_CFG.TRGEN[1:0] ≠ 00)，则一旦检测到外部触发器，设置此位将以连续模式启动计时器。如果在进行单脉冲模式计数时设置了这个位，那么计时器将不会在 LPTIM_ARR 和 LPTIM_CNT 寄存器的下一次匹配时停止，而 LPTIM 计数器将继续在连续模式下计数。这个位只能在启用 LPTIM 时设置。由硬件自动复位。
1	SNGMST	定时器以单脉冲模式启动位。 这个位由软件设置，由硬件清除。 如果软件启动(LPTIM_CFG.TRGEN[1:0] = 00)，设置此位将以单脉冲模式启动 LPTIM。如果软件启动被禁用(LPTIM_CFG.TRGEN[1:0] = 00)，则一旦检测到外部触发器，就设置这个位以单脉冲模式启动 LPTIM。如果这个位是在 LPTIM 处于连续计数模式时设置的，那么 LPTIM 将在 LPTIM_ARR 和 LPTIM_CNT 寄存器之间的后续匹配处停止。这个位只能在启用 LPTIM 时设置。由硬件自动复位。
0	LPTIMEN	LPTIM 使能位。 该位由软件置 1 和清除。 0: LPTIM 禁止 1: LPTIM 开启

### 12.5.7 LPTIM 比较寄存器(LPTIM\_COMP)

偏移地址: 0x14

复位值: 0x0000 0000

注意: LPTIM\_COMP 寄存器只能在启用 LPTIM 时修改 (LPTIM\_CTRL.LPTIMEN = 1)



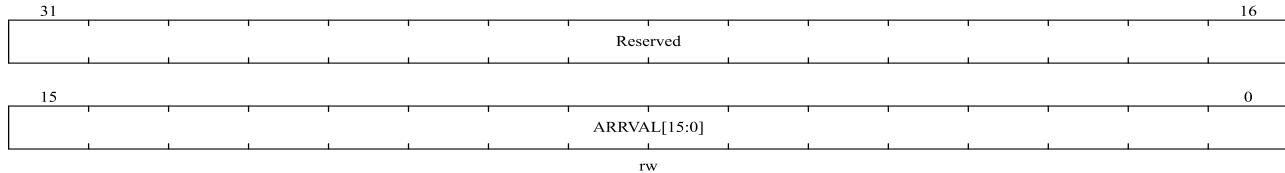
位域	名称	描述
31:16	Reserved	保留，必须保持复位值。
15:0	CMPVAL[15:0]	比较值 CMPVAL[15:0]是 LPTIM 使用的比较值

### 12.5.8 LPTIM 自动重载寄存器(LPTIM\_ARR)

偏移地址: 0x18

复位值: 0x0000 0001

注意: LPTIM\_ARR 寄存器只能在启用 LPTIM 时修改 (LPTIM\_CTRL.LPTIMEN = 1)

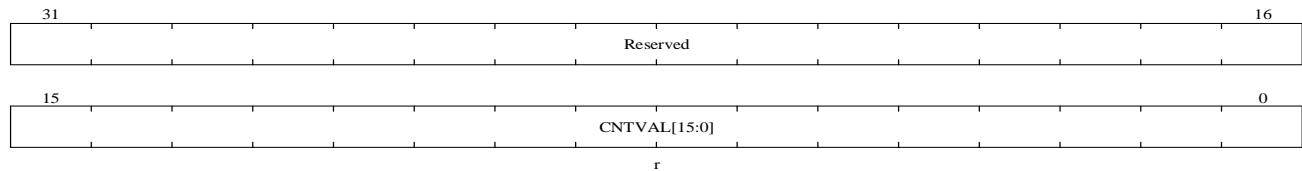


位域	名称	描述
31:16	Reserved	保留, 必须保持复位值。
15:0	ARRVAL[15:0]	自动重载值 ARRVAL[15:0]是 LPTIM 的自动重载值, 该值必须严格大于 LPTIM_COMP.CMPVAL[15:0]值。

### 12.5.9 LPTIM 计数寄存器(LPTIM\_CNT)

偏移地址: 0x1C

复位值: 0x0000 0000

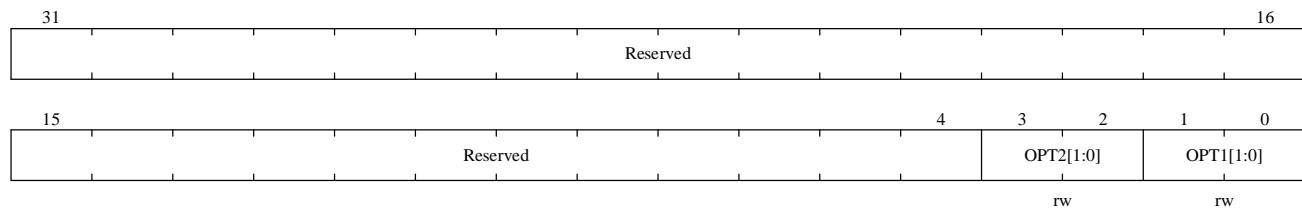


位域	名称	描述
31:16	Reserved	保留, 必须保持复位值。
15:0	CNTVAL[15:0]	计数器数值。 当 LPTIM 使用异步时钟运行时, 读取 LPTIM_CNT 寄存器可能会返回不可靠的值。因此, 在这种情况下, 有必要执行两个连续的读访问, 并验证两个返回的值是否相同。如果相同, 则读访问是可靠的。

### 12.5.10 LPTIM 选项寄存器(LPTIM\_OPT)

偏移地址: 0x20

复位值: 0x0000 0000



位域	名称	描述
31:4	Reserved	保留，必须保持复位值。
3:2	OPT2[1:0]	LPTIM Input2 连接选择位 00: LPTIM Input 2 连接到 I/O 01: LPTIM Input 2 连接到 COMP1_OUT 10: LPTIM Input 2 连接到 COMP2_OUT 11: LPTIM Input 2 连接到 COMP3_OUT
1:0	OPT1[1:0]	LPTIM Input1 连接选择位 00: LPTIM Input 1 连接到 I/O 01: LPTIM Input 1 连接到 COMP1_OUT 10: LPTIM Input 1 连接到 COMP2_OUT 11: LPTIM Input 1 连接到 COMP3_OUT

## 13 独立看门狗 (IWDG)

### 13.1 简介

N32G430 内置独立看门狗 (IWDG) 和窗口看门狗 (WWDG) 定时器，解决软件错误导致的问题。看门狗定时器使用非常灵活，提高了系统的安全性和定时控制的准确性。

独立看门狗 (IWDG) 由运行在 40KHz 的低速内部时钟 (LSI 时钟) 驱动，在死循环事件或 MCU 卡死发生时，它仍然可以运行。这可以提供更高的安全级别、定时精度和看门狗的灵活性。它可以通过重置来解决由于软件故障引起的系统故障。IWDG 最适合需要看门狗在主应用程序之外作为完全独立进程运行但时序精度限制较低的应用程序。

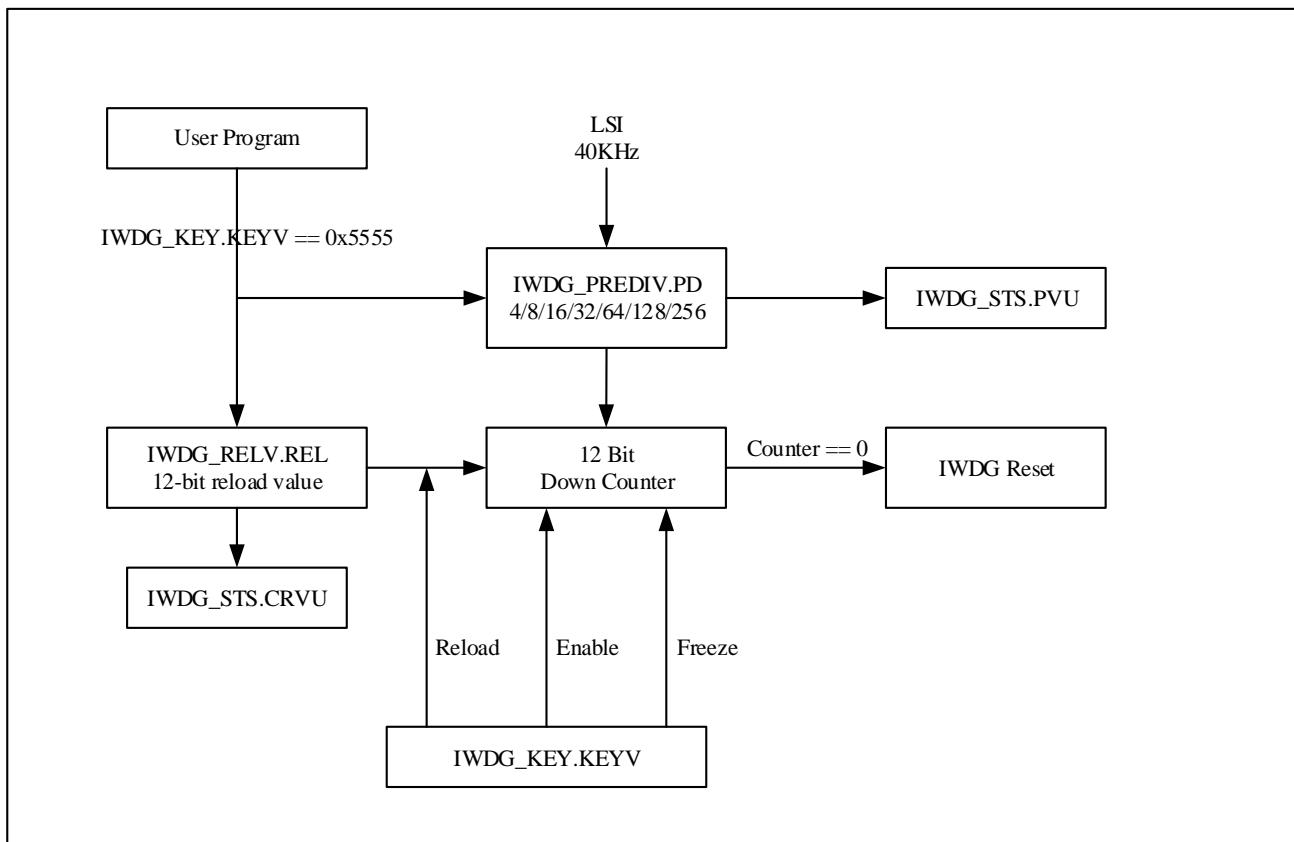
IWDG 可以通过 APB1 总线配置，时钟频率为 32MHz。但是，IWDG 主要工作在 LSI 时钟和 VDD 电源电压的备用电源域 (BKP)。

### 13.2 主要特征

- 独立的 12 位递减计数器
- RC 振荡器提供独立的时钟源，也可以工作在 SLEEP/STOP0/STOP2/STANDBY 模式
- 可以匹配复位和低功耗唤醒
- 当递减计数器达到 0x000 时，系统复位（如果激活了看门狗）

## 13.3 功能描述

图 13-1 独立看门功能框图



注意：看门狗功能在 VDD 供电区，在 SLEEP/STOP0/STOP2/STANDBY 模式下仍能正常工作。

要启用 IWDG，我们需要将 0xCCCC 写入 IWDG\_KEY.KEYV[15:0]位。计数器从复位值(0xFFFF)开始递减计数。当计数器计数到 0x000 时，它会向 MCU 产生一个复位信号 (IWDG\_RESET)。除此之外，只要在复位前将 0xAAAA (重装载请求)写入 IWDG\_KEY.KEYV[15:0]位，计数器值就会设置为 IWDG\_RELV.REL[11:0]位中的重装载值并防止看门狗复位整个设备。

如果通过选项字节使能“硬件看门狗定时器”功能，则看门狗将在系统上电后自动开始运行并产生系统复位，除非软件在计数器到达‘0’之前重新加载计数器。

### 13.3.1 寄存器访问保护

IWDG\_PREDIV 和 IWDG\_RELV 寄存器是写保护的。要修改这两个寄存器的值，用户需要将 0x5555 写入 IWDG\_KEY.KEYV[15:0]位。写入其他值会再次启用写保护。IWDG\_STS.PVU 表示预分频值更新是否正在进行，IWDG\_STS.CRVU 表示 IWDG 是否正在更新重装载值。当预分频值和/或重装载值正在更新时，硬件设置 IWDG\_STS.PVU 位和/或 IWDG\_STS.CRVU 位。预分频值和/或重装载值更新完成后，硬件清除 IWDG\_STS.PVU 位和/或 IWDG\_STS.CRVU 位。

重新加载操作（把 0xAAAA 写入 IWDG\_KEY.KEYV[15:0]）也将导致寄存器再次变为写保护。

### 13.3.2 调试模式

在调试模式下（Cortex-M4 内核停止），IWDG 计数器将继续正常工作或停止，具体取决于调试模块中的 `DBG_CTRL.IWDG_STOP` 位。如果该位设置为“1”，则计数器停止。该位为“0”时，计数器正常工作。详见 23.3.2 调试模块章节。

### 13.3.3 IWDG 冻结

一旦 IWDG 使能（无论是硬件还是软件），除非生成系统复位或通过将 0x4567 写入 `IWDG_KEY.KEYV[15:0]` 位来配置运行时冻结，否则 IWDG 不会停止计数。用户还可以在某些工作模式下配置 IWDG 冻结。IWDG 在 sleep、stop0、stop2 和 standby 模式下提供冻结选项。当 IWDG 开启时，它将强制 LSI 时钟开启。

## 13.4 用户界面

IWDG 模块用户界面包含 4 个寄存器：密钥寄存器（`IWDG_KEY`）、预分频寄存器（`IWDG_PREDIV`）、重载寄存器（`IWDG_RELV`）和状态寄存器（`IWDG_STS`）。对应的定义、宽度、地址、读写访问和复位信息如下表所示。

### 13.4.1 操作流程

当 IWDG 从软件（将 0xAAAA 写入 `IWDG_KEY.KEYV[15:0]`）或硬件（清零 `WDG_SW` 位）复位启用时。它从 0xFFFF 开始递减计数。向下计数间隙由预分频 LSI 时钟确定。重新加载计数器后，新一轮递减计数器的值将从 `IWDG_RELV.REL[11:0]` 中的值开始，而不是 0xFFFF。

程序正常运行时，软件需要在计数器到达 0 前喂狗，开始新一轮的递减计数。当计数器达到 0 时，表示程序故障。IWDG 在这种情况下产生复位信号。

如果用户想要配置 IWDG 预分频和重装载值寄存器，需要先将 0x5555 写入 `IWDG_KEY.KEYV[15:0]`。然后确认 `IWDG_STS.CRVU` 位和 `IWDG_STS.PVU` 位。`IWDG_STS.CRVU` 位指示重装载值更新正在进行，`IWDG_STS.PVU` 表示预分频值更新正在进行。只有当这两位为 0 时，用户才能更新相应的值。当更新正在进行时，硬件将相应位设置为 1。此时，读取 `IWDG_PREDIV.PD[2:0]` 或 `IWDG_RELV.REL[11:0]` 无效，因为数据需要同步到 LSI 时钟域。从 `IWDG_PREDIV.PD[2:0]` 或 `IWDG_RELV.REL[11:0]` 读取的值将在硬件清除 `IWDG_STS.PVU` 位或 `IWDG_STS.CRVU` 位后才有效。

如果应用程序使用多个重装载值或预分频值，则必须等到 `IWDG_STS.CRVU` 位复位后才能更改重装载值，`IWDG_STS.PVU` 位复位后才能更改预分频值。但是，在更新预分频值和重装载值后，或只更新预分频值后，或只更新重装载值后，无需等到 `IWDG_STS.CRVU` 位或 `IWDG_STS.PVU` 位复位，可能继续执行代码（即使在进低功耗模式的情况下，写入操作也会被考虑并完成）。

预分频寄存器和重装载寄存器控制产生复位的时间，如表 13-1。

表 13-1 IWDG 计数最大和最小复位时间

预分频因子	<code>PD[2:0]</code>	最小时长（ms） <code>RL[11:0]=0</code>	最大时长（ms） <code>RL[11:0]=0xFFFF</code>
/4	000	0.1	409.6

/8	001	0.2	819.2
/16	010	0.4	1638.4
/32	011	0.8	3276.8
/64	100	1.6	6553.6
/128	101	3.2	13107.2
/256	11x	6.4	26214.4

### 13.4.2 IWDG 配置流程

软件配置流程：

- 将 0x5555 写入 IWDG\_KEY.KEYV[15:0]位以启用对 IWDG\_PREDIV 和 IWDG\_RELV 寄存器的写访问；
- 检查 IWDG\_STS.PVU 位或 IWDG\_STS.CRVU 位，如果为 0，则继续下一步；
- 配置 IWDG\_PREDIV.PD[2:0]位以选择预分频值；
- 配置 IWDG\_RELV.REL[11:0]位重装载值；
- 将 0xAAAA 写入 IWDG\_KEY.KEYV[15:0]位，用重装载值更新计数器；
- 通过软件或硬件将 0xCCCC 写入 IWDG\_KEY.KEYV[15:0]位来启用看门狗。

如果用户想改变预分频值和重装载值，重复步骤 1~5。如果没有，只需按照第 5 步喂狗。

## 13.5 IWDG 寄存器

### 13.5.1 IWDG 寄存器总览

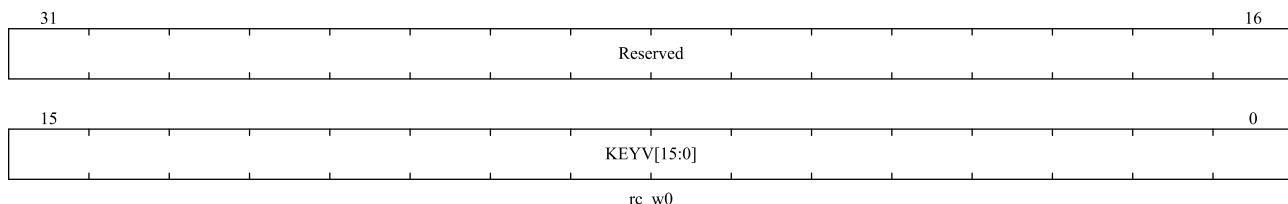
表 13-2 IWDG 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KEY	Reserved																								KEYV[15:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0													
0x04	IWDG_PREDIV	Reserved																									PD[2:0]						
	Reset value	0	0	0	0	0	0																										
0x08	IWDG_RELV	Reserved																								REL[11:0]							
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1													
0x0C	IWDG_STS	Reserved																									CRVU PVU						
	Reset value	0	0	0	0	0	0																										

### 13.5.2 IWDG 密钥寄存器 (IWDG\_KEY)

偏移地址: 0x00

复位值: 0x00000000

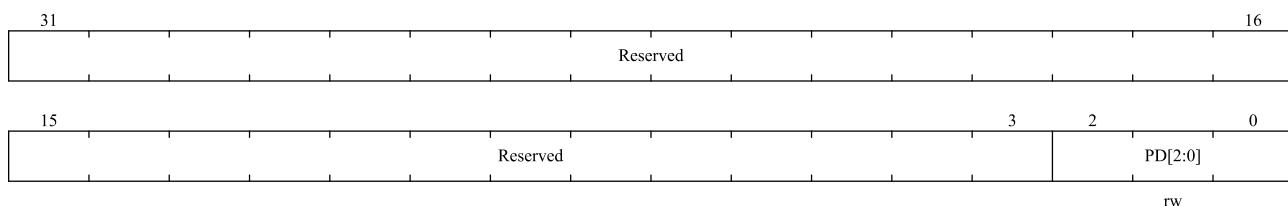


位域	名称	描述
31:16	保留	保留, 必须保持复位值。
15:0	KEYV[15:0]	密钥寄存器: 只有特定的值才能发挥特定的作用 0xCCCC: 启动看门狗计数器, 如果硬件看门狗使能则无效, (如果选择了硬件看门狗, 则不受该命令字限制) 0xAAAA: 用 IWDG_RELV 寄存器中的 REL 值重新加载计数器以防止复位 0x5555: 禁用 IWDG_PREDIV 和 IWDG_RELV 寄存器的写保护 0x4567: IWDG 的运行时间冻结, 运行时停止 IWDG 0x89AB: 从运行时冻结中恢复

### 13.5.3 IWDG 预分频寄存器 (IWDG\_PREDIV)

偏移地址: 0x04

复位值: 0x00000000



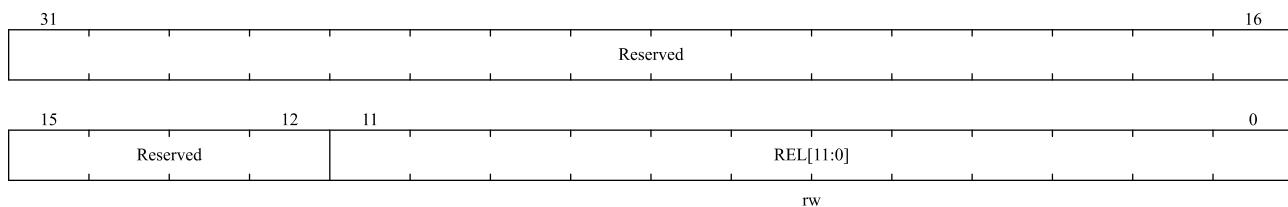
位域	名称	描述
31:3	保留	保留, 必须保持复位值。
2:0	PD[2:0]	预分频因子 预分频值: 当 IWDG_KEY.KEYV[15:0]不是 0x5555 时具有写访问保护。IWDG_STS.PVU 位必须为 0, 否则 PD[2:0]值无法更改。分频系数如下: 000: 预分频因子=4 001: 预分频因子=8 010: 预分频因子=16 011: 预分频因子=32 100: 预分频因子=64 101: 预分频因子=128

位域	名称	描述
		其他: 预分频因子=256 注意: 读取该寄存器将返回来自 VDD 电压域的预分频值。如果正在进行写操作, 则回读值可能无效。因此, 读取值仅在 IWDG_STS.PVU 位为 0 时有效。

### 13.5.4 IWDG 重装载寄存器 (IWDG\_RELV)

偏移地址: 0x08

复位值: 0x00000FFF

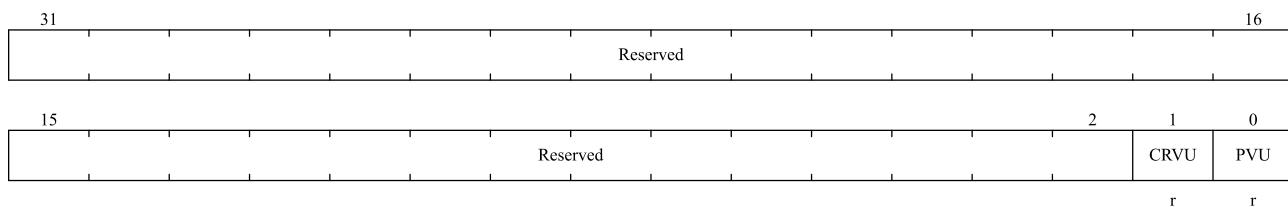


位域	名称	描述
31:12	保留	保留, 必须保持复位值。
11:0	REL[11:0]	看门狗计数器重装载值。 带写保护。定义看门狗计数器的重装载值, 每次将 0xFFFF 写入 IWDG_KEY.KEYV[15:0] 位时将其加载到计数器。然后计数器从该值开始倒计时。看门狗超时周期可以根据这个重装载值和时钟预分频值计算, 参考表 13-2。 该寄存器只能在 IWDG_STS.CRVU 位为 0 时修改。 注意: 读取该寄存器将返回来自 VDD 电压域的重装载值。如果正在进行写操作, 则回读值可能无效。因此, 读取值仅在 IWDG_STS.CRVU 位为 0 时有效。

### 13.5.5 IWDG 状态寄存器 (IWDG\_STS)

偏移地址: 0x0C

复位值: 0x00000000



位域	名称	描述
31:2	保留	保留, 必须保持复位值。
1	CRVU	看门狗重装载值更新 重装载值更新: 该位表示正在更新重装载值。硬件置位, 硬件清零。软件只能在 IWDG_KEY.KEYV[15:0] 位的值为 0xFFFF 且该位为 0 时尝试更改 IWDG_RELV.REL[11:0] 的值。

位域	名称	描述
0	PVU	看门狗预分频值更新 预分频值更新：该位表示正在更新预分频值。硬件置位，硬件清零。软件只能在 IWDG_KEY.KEYV[15:0]位的值为 0x5555 且该位为 0 时尝试更改 IWDG_PREDIV.PD[2:0] 的值。

## 14 窗口看门狗 (WWDG)

### 14.1 简介

窗口看门狗 (WWDG) 的时钟是 APB1 32MHz 时钟频率除以 4096 得到的，通过时间窗口的配置来检测程序运行是否异常。因此，WWDG 适用于精确定时，常用于监控因外部干扰或无法预见的逻辑条件导致应用程序偏离其正常操作顺序的软件故障。当 WWDG 递减计数器在达到窗口寄存器值之前或 WWDG\_CTRL.T6 位变为 0 之后刷新时，系统复位发生。

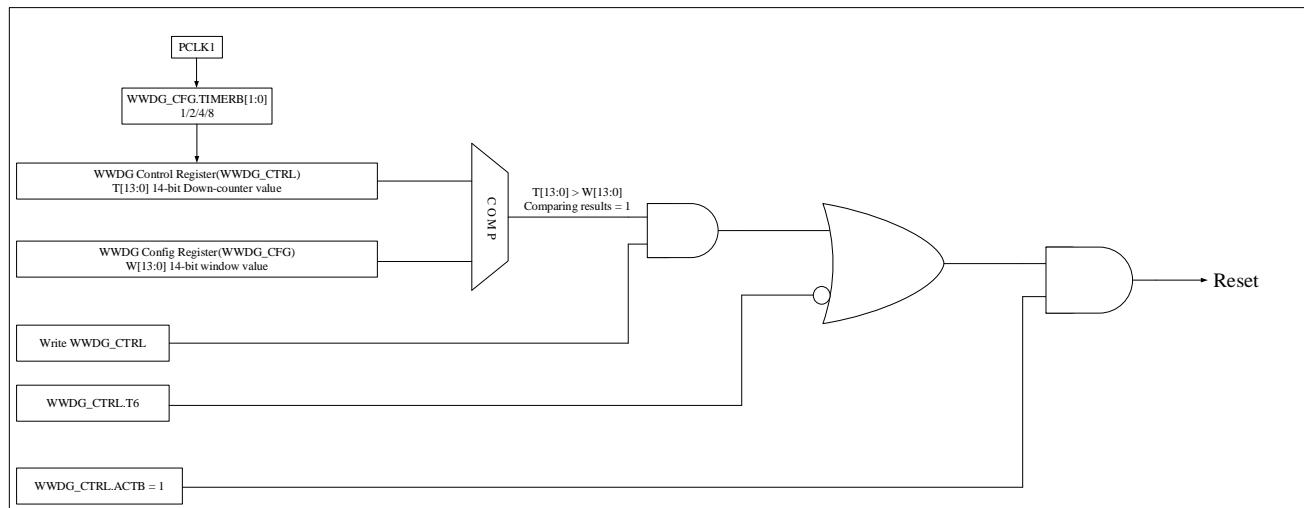
### 14.2 主要特征

- 14 位独立递减计数器可编程
- WWDG 启用后，在以下情况下会发生复位
  - ◆ 递减计数器的值小于 0x40
  - ◆ 当递减后的计数器值大于窗口寄存器的值时，重新加载
- 提前唤醒中断：如果看门狗启动并且中断使能，当计数值达到 0x40 时会产生唤醒中断 (WWDG\_CFG.EWINT)

### 14.3 功能描述

如果看门狗被激活 (WWDG\_CTRL.ACTB)，当 14 位 (WWDG\_CTRL.T[13:0]) 递减计数器到达 0x3F (WWDG\_CTRL.T6 位清零)，或者软件重新加载计数器时计数器值大于窗口寄存器的值，将产生系统复位。为了避免系统复位，软件在正常运行时必须定期刷新窗口中的计数器值。

图 14-1 窗口看门狗功能框图



设置 WWDG\_CTRL.ACTB 位以启用看门狗，此后，WWDG 将保持打开状态，直到发生复位。14 位递减计数器独立运行，无论 WWDG 是否使能，计数器都会持续递减计数。因此，需要将高 8 位 MSB(WWDG\_CTRL.T[13:6]) 中的某一位设置为 1，以防止在启用后立即复位。由时钟 APB1 和

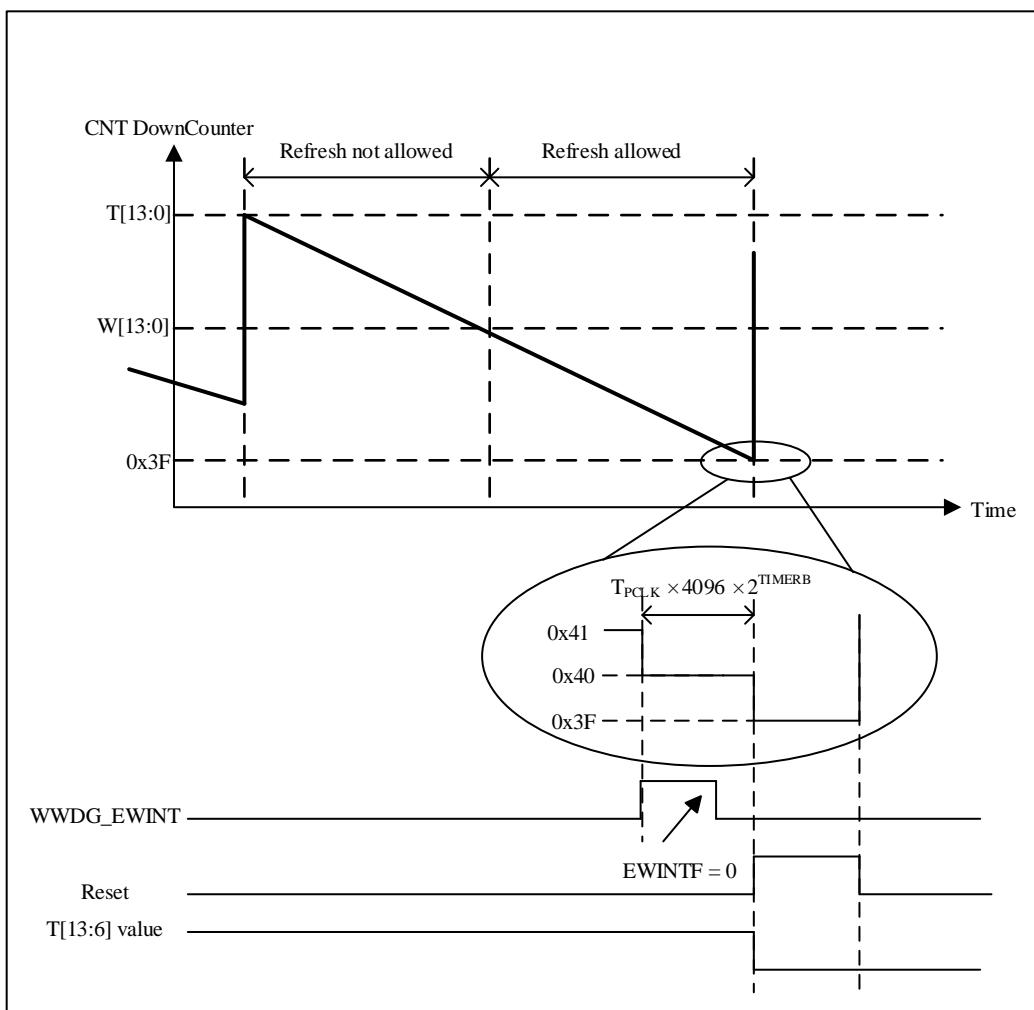
WWDG\_CFG.TIMERB[1:0]位设置的预分频器值决定了计数器的递减速速度。WWDG\_CFG.W[13:0]位设置窗口的上限。

当递减计数器在达到窗口寄存器值之前或 WWDG\_CTRL.T6 位变为 0 之后刷新，将产生系统复位。图 14-2 描述了窗口寄存器的工作过程。

设置 WWDG\_CFG.EWINT 位以启用提前唤醒中断。当递减计数器到达 0x40 时，将产生中断。您可以分析软件故障的原因或将重要数据保存在相应的中断服务程序（ISR）中，并重新加载计数器以防止 WWDG 复位。将“0”写入 WWDG\_STS.EWINTF 位以清除中断。

## 14.4 刷新看门狗和中断产生的时许

图 14-2 WWDG 的刷新窗口和中断时序



看门狗刷新窗口在 WWDG\_CFG.W[13:0] 值（最大值 0xFFFF）和 0x3F 之间，在此窗口外刷新将向 MCU 生成复位请求。计数器使用分频后的 APB1 时钟从 0xFFFF 向下计数到 0x3F，最大计数时间和最小计数时间如表 14-1 所示（假设 APB1 时钟为 32MHz），计算公式为：

$$T_{WWDG} = T_{PCLK1} \times 4096 \times 2^{TIMERB} \times (T[13:0] - 0X3F + 1)$$

其中：

T<sub>WWDG</sub>:WWDG 超时

T<sub>PCLK1</sub>:APB1 时钟间隔，单位为：ms

PCLK1=32MHz 时的最小-最大超时时长

表 14-1 WWDG 的最大和最小计数时间

TIMERB	最大超时 (ms)	最小超时 (ms)
0	2089	0.128
1	4178	0.256
2	8356	0.512
3	16712	1.024

## 14.5 调试模式

在调试模式下（Cortex-M4 内核停止），WWDG 计数器将继续正常工作或停止，具体取决于调试模块中的 DBG\_CTRL.WWDG\_STOP 位。如果该位设置为“1”，则计数器停止。该位为“0”时，计数器正常工作。详见 23.3.2 调试模块章节。

## 14.6 用户界面

### 14.6.1 WWDG 配置流程

1. 配置 RCC\_APB1PCLKEN.WWDGEN[11]位使能 WWDG 模块的时钟
2. 软件设置 WWDG\_CFG.TIMERB[15:14]位来配置 WWDG 的预分频因子
3. 软件配置 WWDG\_CTRL.T[13:0]位，设置计数器的起始值。需要将高 8MSB 位(WWDG\_CTRL.T[13:6])中的某一位设置为 1，以防止在启用后立即复位
4. 配置 WWDG\_CFG.W[13:0]位配置上边界窗口值
5. 设置 WWDG\_CTRL.ACTB[14]位使能 WWDG
6. 软件操作 WWDG\_STS.EWINTF[0]位清除唤醒中断标志
7. 配置 WWDG\_CFG.EWINT[16]位使能提前唤醒中断

## 14.7 WWDG 寄存器

### 14.7.1 WWDG 寄存器总览

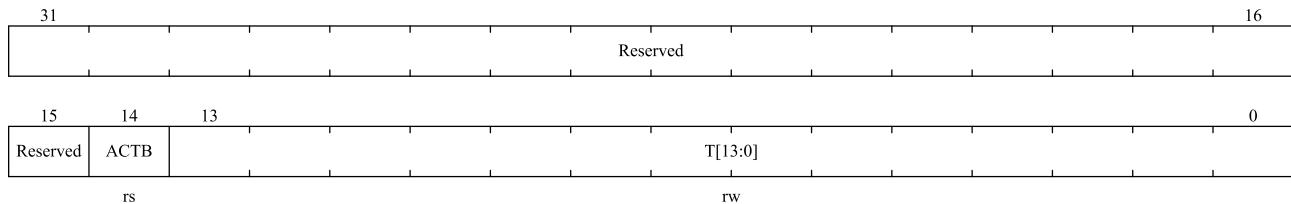
表 14-2 WWDG 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0x00	WWDG_CTRL																																		
	Reset value																		0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	
0x04	WWDG_CFG																																		
	Reset value																		0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0x08	WWDG_STS																																		
	Reset value																																		

### 14.7.2 WWDG 控制寄存器 (WWDG\_CTRL)

偏移地址: 0x00

复位值: 0x00003FFF

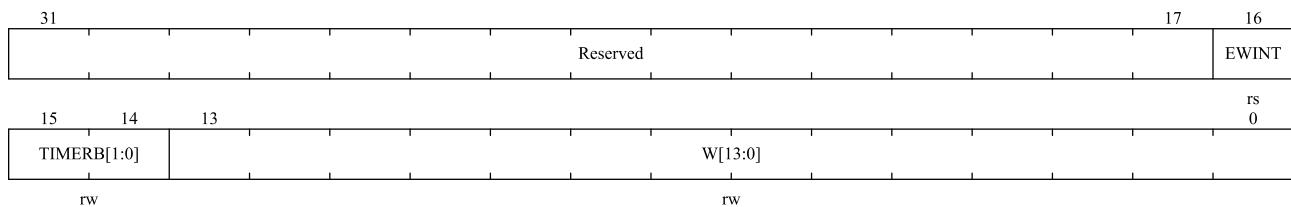


位域	名称	描述
31:15	保留	保留，必须保持复位值。
14	ACTB	激活位 当 ACTB=1 时，看门狗可以产生复位。该位由软件置位，仅在复位后由硬件清零。当 ACTB=1 时，看门狗可以产生复位。 0: 禁用看门狗 1: 启用看门狗
13:0	T[13:0]	这些位包含看门狗计数器的值。它每( $4096 \times 2^{\text{TIMERB}}$ )个 PCLK1 周期递减。当它从 0x40 翻转到 0x3F (T6 清零) 时，会产生一个复位。

### 14.7.3 WWDG 配置寄存器 (WWDG\_CFG)

偏移地址: 0x04

复位值: 0x00003FFF

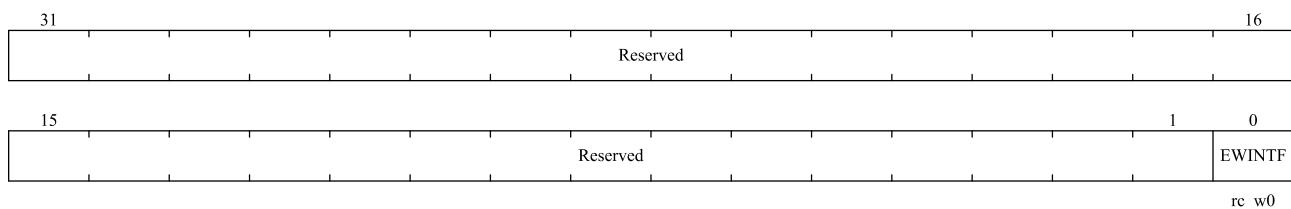


位域	名称	描述
31:17	保留	保留, 必须保持复位值。
16	EWINT	提前唤醒中断 设置后, 只要计数器达到值 0x40, 就会发生中断。此中断仅在复位后由硬件清除。
15:14	TIMERB[1:0]	时基 预分频器的时基可以修改如下: 00: CK 计数器时钟 (PCLK1 除以 4096) 除以 1 01: CK 计数器时钟 (PCLK1 除以 4096) 除以 2 10: CK 计数器时钟 (PCLK1 除以 4096) 除以 4 11: CK 计数器时钟 (PCLK1 除以 4096) 除以 8
13:0	W[13:0]	14 位窗口值 这些位包含要与递减计数器比较的窗口值。

### 14.7.4 WWDG 状态寄存器 (WWDG\_STS)

偏移地址: 0x08

复位值: 0x0000



位域	名称	描述
31:1	保留	保留, 必须保持复位值。
0	EWINTF	提前唤醒中断标志 当计数器达到值 0x40 时, 该位由硬件设置。它必须由软件通过写入“0”来清零。写入“1”无效。如果中断未使能, 该位也会置位。

## 15 模数转换 (ADC)

### 15.1 简述

12 位 ADC 是使用逐次逼近的高速模数转换器。它有多个通道，19 个通道，可测量 16 个外部和 3 个内部信号源。每个通道的 A/D 转换有四种执行模式：单次、连续、扫描或间断。ADC 转换值存储（左对齐/右对齐）在 16 位数据寄存器中。可以通过模拟看门狗检测输入电压是否在用户定义的高/低阈值内，并且 ADC 的输入时钟的最大频率为 80MHz。

### 15.2 主要特征

- 支持 1 个 ADC，支持单端和差分输入，最多可测量 16 个外部和 3 个内部源。
- 支持 12 位、10 位、8 位、6 位分辨率，12bit 分辨率下最高采样率为 4.7MSPS。
- ADC 时钟源分为工作时钟源、采样时钟源和计时时钟源。
  - ◆ 仅可配置 AHB\_CLK 作为工作时钟源。
  - ◆ 可配置 PLL 作为采样时钟源，最高可到 80MHz，支持分频 1,2,4,6,8,10,12,16,32,64,128,256。
  - ◆ 可配置 AHB\_CLK 作为采样时钟源，最高可到 80MHz，支持分频 1,2,4,6,8,10,12,16,32。
  - ◆ 计时时钟用于内部计时功能，频率必须配置成 1MHz。
- 支持定时器触发 ADC 采样。
- 各通道的采样时间间隔可编程
- 支持扫描模式
- 支持单次转换
- 支持连续转换
- 支持间断模式
- 支持自校准
- 支持 DMA
- 中断生成
  - ◆ 转换结束
  - ◆ 注入转换结束
  - ◆ 模拟看门狗事件
- 带内嵌数据一致性的数据对齐
- 可以外部触发注入转换和规则转换
- ADC 的工作电压在 2.4V 到 3.6V 之间
- ADC 支持转换的电压在  $V_{REF-}$  和  $V_{REF+}$  之间

## 15.3 功能描述

图 15-1 ADC 框图

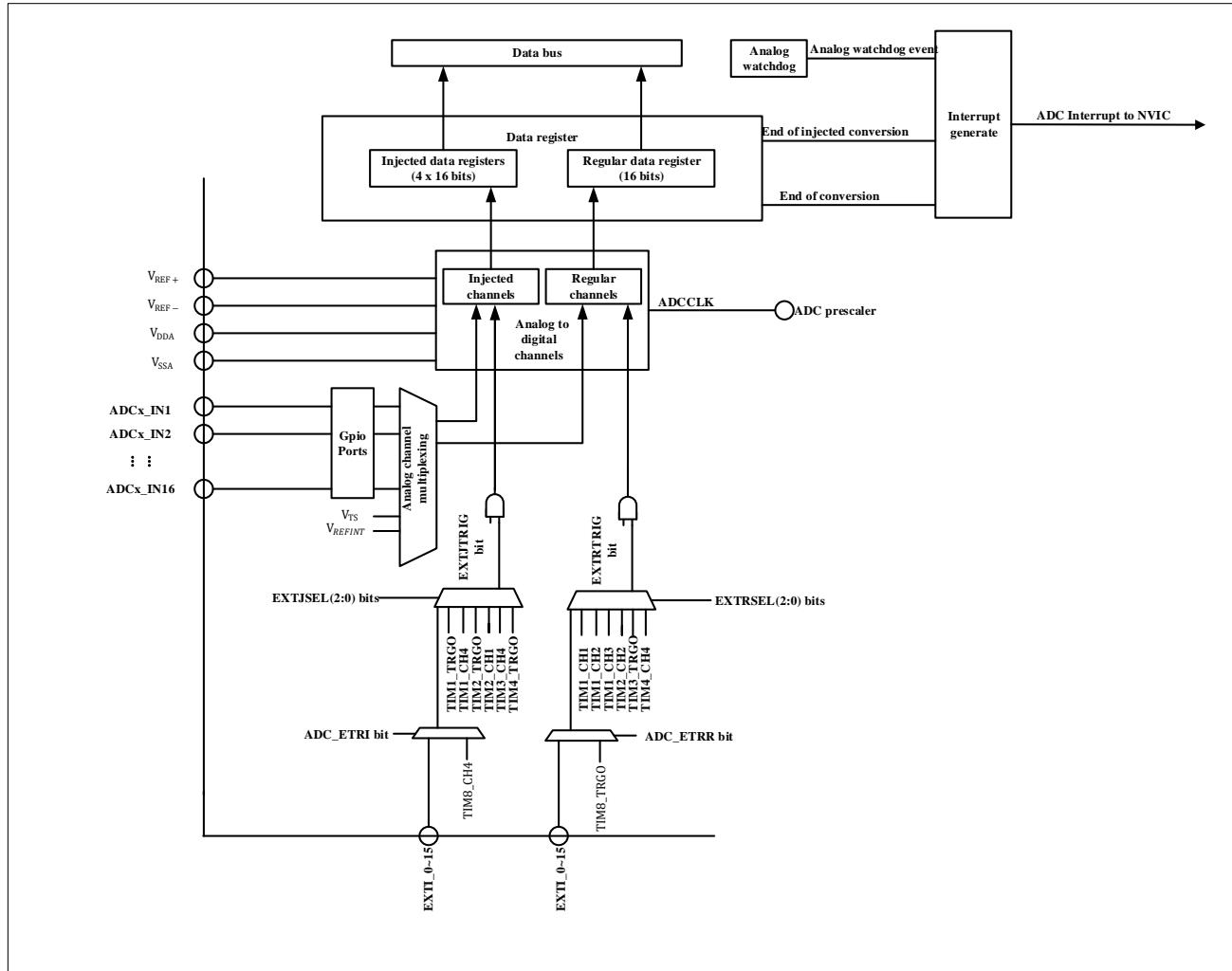


表 15-1 ADC 引脚说明

名称	信号类型	说明
V <sub>REF+</sub>	输入, 模拟参考正极	ADC使用的高端/正极参考电压, $2.4V \leq V_{REF+} \leq V_{DDA}$
V <sub>DDA</sub>	输入, 模拟电源	等效于V <sub>DD</sub> 的模拟电源且: $2.4V \leq V_{DDA} \leq V_{DD}(3.6V)$
V <sub>REF-</sub>	输入, 模拟参考负极	ADC使用的低端/负极参考电压, $V_{REF-} = V_{SSA}$
V <sub>SSA</sub>	输入, 模拟电源地	等效于V <sub>SS</sub> 的模拟电源地
ADCx_IN[16:1]	模拟输入信号	16个模拟外部输入通道

1. V<sub>DDA</sub> 和 V<sub>SSA</sub> 应该分别连接到 V<sub>DD</sub> 和 V<sub>SS</sub>。
2. 如果有 V<sub>REF-</sub> 引脚 (取决于封装), 必须和 V<sub>SSA</sub> 相连接。
3. 如果没有 V<sub>REF+</sub> 引脚 (取决于封装), 尽量保证 V<sub>DDA</sub> 和 V<sub>DD</sub> 的电压值一致, 否则会影响 ADC 精度。

### 15.3.1 ADC 时钟

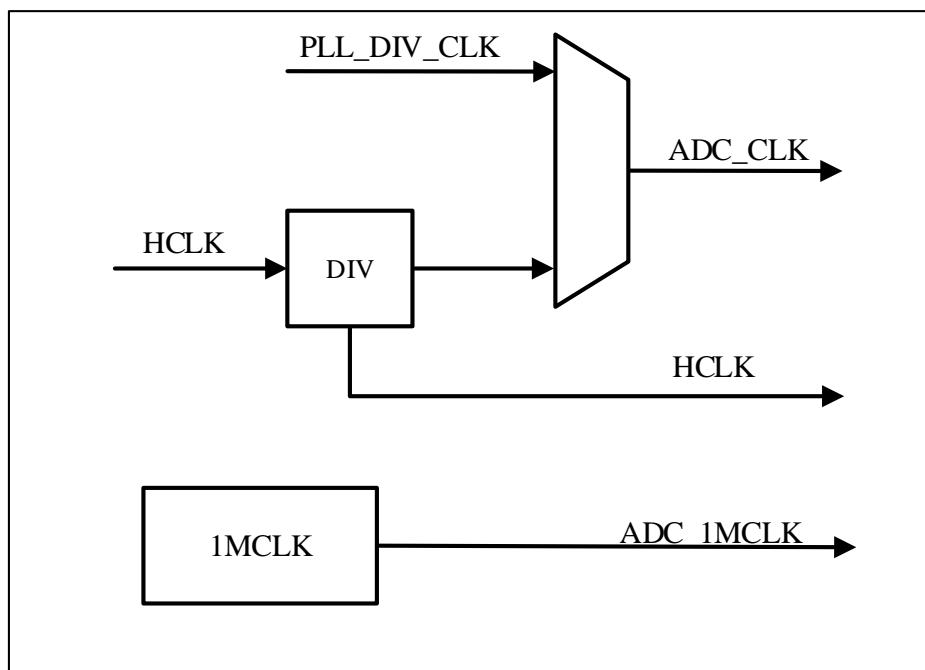
ADC 需要三个时钟，ADC\_CLK, HCLK, ADC\_1MCLK。

- HCLK 用于寄存器的访问时钟。
- ADC\_CLK 为 ADC 的工作时钟，ADC\_CLK 有两个源（HCLK 的分频或 PLL 的分频），HCLK 分频与系统是同步时钟，PLL 的分频与系统是异步时钟，用同步时钟的好处是在触发 ADC 响应触发时，没有不确定性，用 PLL 的分频时钟的好处是可以独立处理 ADC 的工作时钟，不会影响到挂在 HCLK 的其他模块。
- ADC\_1MCLK 用于内部计时功能，在 RCC 中配置，频率大小必须配置成 1MHz。

注：

1. 配置 PLL 作为时钟源时，最高可到 80M，支持分频 1,2,4,6,8,10,12,16,32,64,128,256。
2. 配置 AHB\_CLK 分频作为工作时钟最高可到 80M，支持分频 1,2,4,6,8,10,12,16,32。
3. 切换 ADC IM 时钟源时，需要保证 HSI 时钟开启。

图 15-2 ADC 时钟



### 15.3.2 ADC 开关控制

只有在上电过程完成后，您才能进行下一步。您可以通过轮询 ADC\_CTRL3.RDY 来检查上电是否完成。

您可以设置 ADC\_CTRL2.ON 来打开 ADC。第一次设置 ADC\_CTRL2.ON 时，它将 ADC 从断电状态唤醒。在 ADC 的上电延迟 (tSTAB) 之后，当 ADC\_CTRL2.ON 再次置位时，转换开始。

可以通过清除 ADC\_CTRL2.ON 将 ADC 置于断电模式来停止转换。在这种模式下，ADC 几乎不消耗功率

(仅几  $\mu$ A)。可以通过轮询 ADC\_CTRL3.PDRDY 来检查掉电情况。

在 ADC Disable 的时候默认都是 PowerDown 模式，这个模式下只要不断电，不需要重新校正，校正值会在 ADC 自动保持。为了进一步的降低功耗，ADC 有设计一个深睡眠模式。会在 ADC Disable 进入深睡眠模式，ADC 内部的校正值会丢失，需要重新校正。深睡眠模式可以省大概  $0.2\mu$ A 的功耗。

### 15.3.3 通道选择

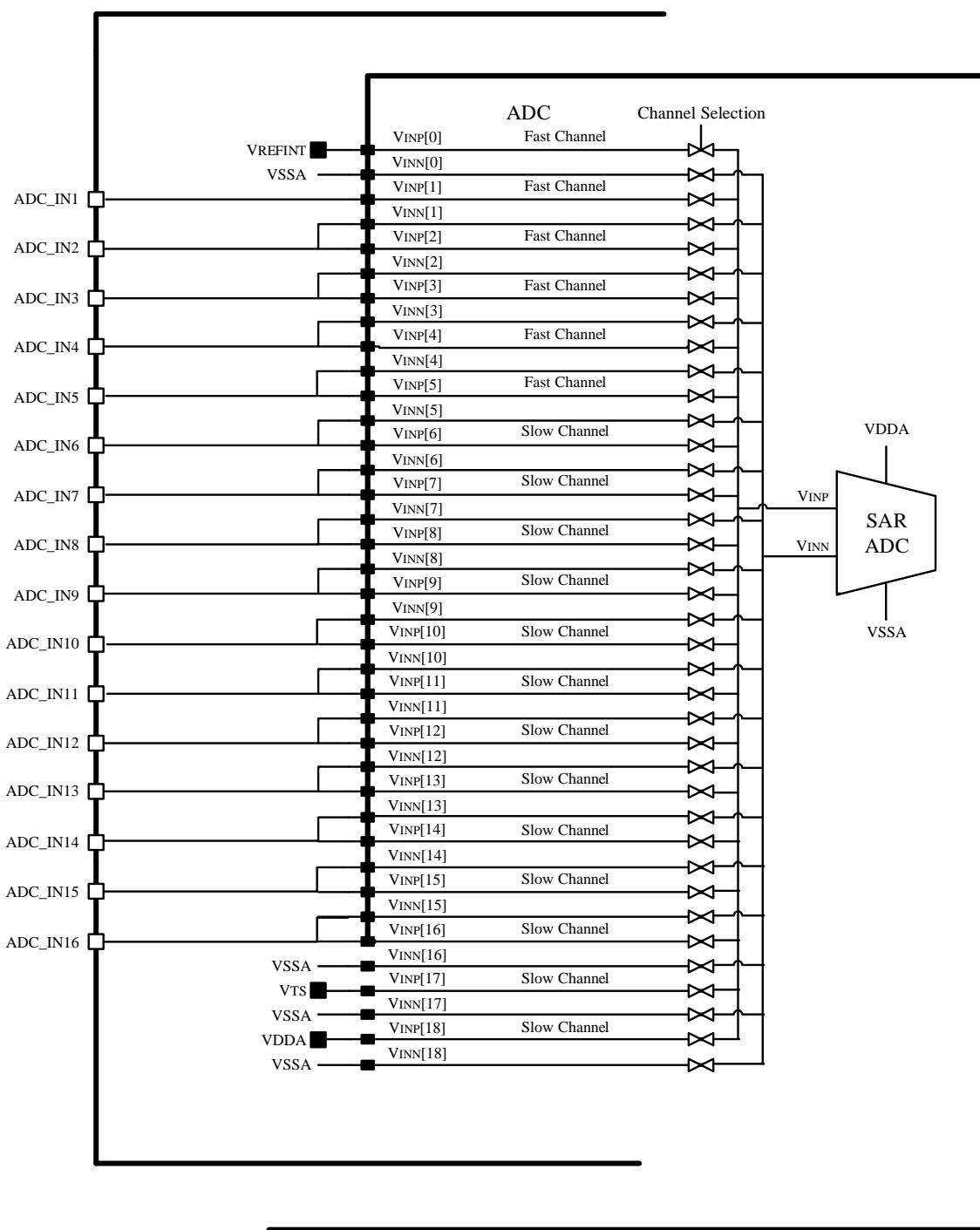
每个通道可以配置为规则序列和注入序列。

注入序列由多次转换组成，最多为 4 次。ADC\_JSEQ 寄存器指定注入通道和注入通道的转换顺序。ADC\_JSEQ.JLEN[21:20] 位指定注入序列长度。

规则序列由多次转换组成，最多 16 次。ADC\_RSEQx 寄存器指定规则通道和规则通道的转换顺序。ADC\_RSEQ1.LEN[23:20] 位指定规则通道序列长度。

注意：在转换期间，禁止更改 ADC\_RSEQx 或 ADC\_JSEQ 寄存器；ADC\_RSEQx 或 ADC\_JSEQ 寄存器只能在 ADC 空闲时更改。

图 15-3 ADC 通道引脚连接



#### 15.3.4 内部通道

- 温度传感器连接到通道 ADC\_IN17
- V<sub>REFINT</sub> 连接到 ADC\_IN0
- VDDA 连接到通道 ADC\_IN18

ADC 内部通道可以按规则或者注入通道的方式进行转换。

### 15.3.5 单次转换模式

ADC 可以通过配置 ADC\_CTRL2.CTU 为 0 进入单次转换模式。在该模式下，外部触发（适用于规则或注入通道）或设置 ADC\_CTRL2.ON=1（仅适用于规则通道）可以启动 ADC 启动转换，ADC 只进行一次转换。

转换开始后，当一个注入通道转换完成时，注入通道转换结束标志（ADC\_STS.JENDC）将被设置为 1。如果注入通道转换结束中断（ADC\_CTRL1.JENDCIEN）位被设置为 1，一个中断将生成，转换后的数据将存储在 ADC\_JDATx 寄存器中。

转换开始后，当一个规则通道转换完成时，规则通道转换结束标志（ADC\_STS.ENDC）将被置 1。如果规则通道转换结束中断（ADC\_CTRL1.ENDCIEN）位被置 1，则一个中断将生成，转换后的数据将存储在 ADC\_DAT 寄存器中。

单次转换后，ADC 停止。

### 15.3.6 连续转换模式

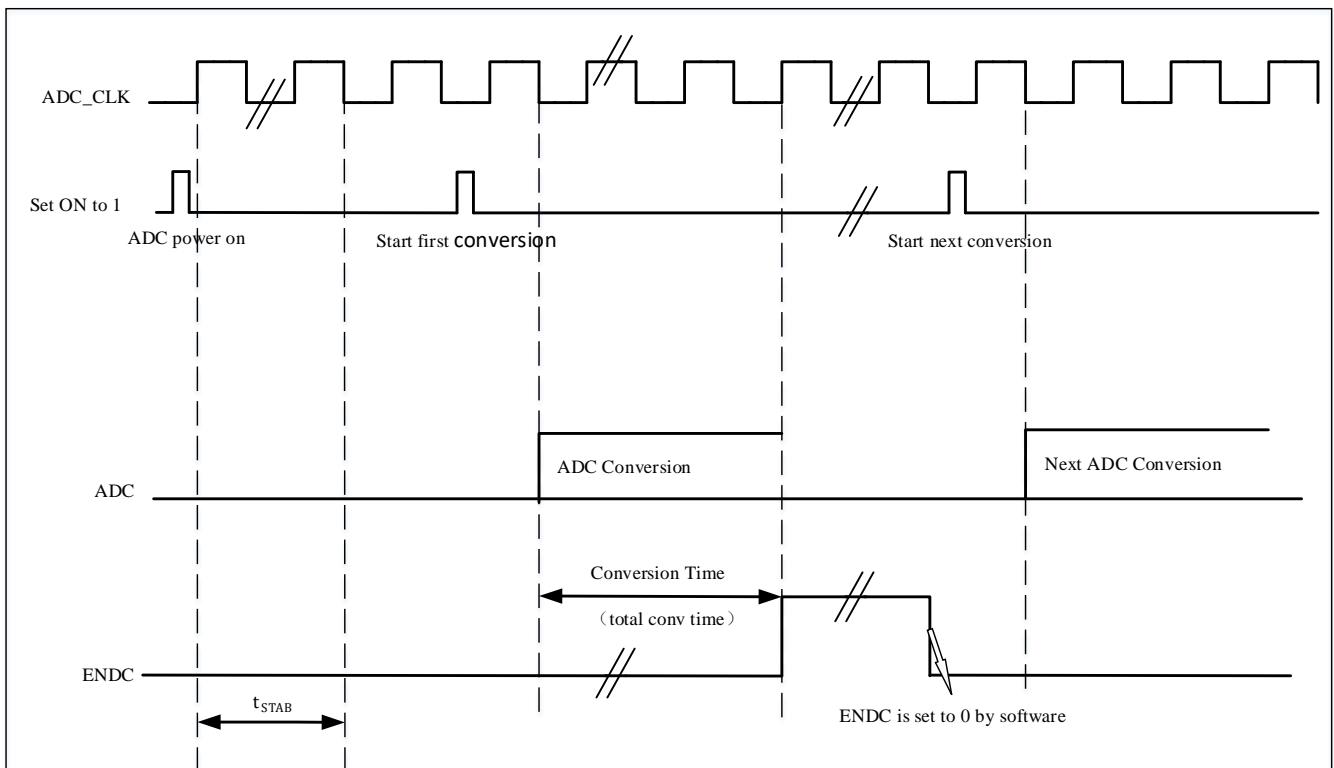
ADC 可以通过配置 ADC\_CTRL2.CTU 为 1 进入连续转换模式。在该模式下，外部触发或设置 ADC\_CTRL2.ON 为 1 可以启动 ADC 开始转换，ADC 会持续转换选择的通道。连续模式仅对规则通道有效，对注入通道无效。

转换开始后，当规则通道转换完成时，规则通道转换结束标志位（ADC\_STS.ENDC）将设置为 1。如果规则通道转换结束中断位（ADC\_CTRL1.ENDCIEN）设置为 1，将产生一个中断。转换后的数据将存储在 ADC\_DAT 寄存器中。

### 15.3.7 时序图

ADC\_CTRL2.ON 首次设置为 1 时，ADC 上电。ADC 上电后，ADC 需要一定的时间来保证其稳定性。ADC 稳定后，再次对 ADC\_CTRL2.ON 写 1，ADC 开始转换，转换结束标志位将在转换完成后设置为 1。

图 15-4 时序图



### 15.3.8 模拟看门狗

可以通过设置 ADC\_CTRL1.AWDGERCH 为 1 在规则通道上打开模拟看门狗，也可以通过将 ADC\_CTRL1.AWDGEJCH 设置为 1 在注入通道上启用模拟看门狗。可以通过配置 ADC\_WDGHIGH.HTH 设置模拟看门狗的高阈值，模拟看门狗的低阈值可以通过 ADC\_WDGLOW.LTH 来设置。模拟看门狗的阈值与数据对齐的方式无关，因为 ADC 的转换值与阈值的比较是在对齐之前完成。当 ADC 转换的值高于模拟看门狗的高阈值或低于模拟看门狗的低阈值时，如果 ADC\_CTRL1.AWDGIEN 已配置，则模拟看门狗标志 (ADC\_STS.AWDG) 将被置为 1，此时会产生中断。通过配置 ADC\_CTRL1.AWDGSGLEN 和 ADC\_CTRL1.AWDGCH[4:0]，可以控制模拟看门狗作用于一个或多个通道。

表 15-2 模拟看门狗通道选择

模拟看门狗警戒的通道	ADC_CTRL1寄存器控制位		
	AWDGSGLEN位	AWDGERCH位	AWDGEJCH位
无	任意值	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有注入和规则通道	0	1	1
单一的 <sup>(1)</sup> 注入通道	1	0	1
单一的 <sup>(1)</sup> 规则通道	1	1	0
单一的 <sup>(1)</sup> 注入或规则通道	1	1	1

(1)由 ADC\_CTRL1.AWDGCH[4:0]选择

### 15.3.9 扫描模式

通过配置 ADC\_CTRL1.SCAMD 为 1 可以开启扫描转换模式，通过配置四个寄存器 ADC\_RSEQ1、ADC\_RSEQ2、ADC\_RSEQ3、ADC\_JSEQ 可以选择转换通道序列，ADC 会对所有选择的规则或注入通道进行扫描转换。转换开始后，通道将一个一个转换。如果此时 ADC\_CTRL2.CTU 为 1，则在所有选中的规则通道的转换完成后，将从转换序列的第一个通道重新开始转换。注入通道不支持连续转换。DMA 功能可以通过设置 ADC\_CTRL2.ENDMA 为 1 来开启，DMA 会在规则通道转换完成后将数据传输到 SRAM 中。

### 15.3.10 注入通道管理

#### 15.3.10.1 自动注入

如果设置了 ADC\_CTRL1.AUTOJC 位，则 ADC\_JSEQx 选择的注入通道将在 ADC\_RSEQx 选中的规则通道转换完成后自动转换。单个触发器可以转换多达 16+4 个通道。设置 ADC\_CTRL2.CTU 转换序列将被连续转换。

开启该功能时，需要关闭注入通道的外部触发。

此功能不能与间断模式同时使用。

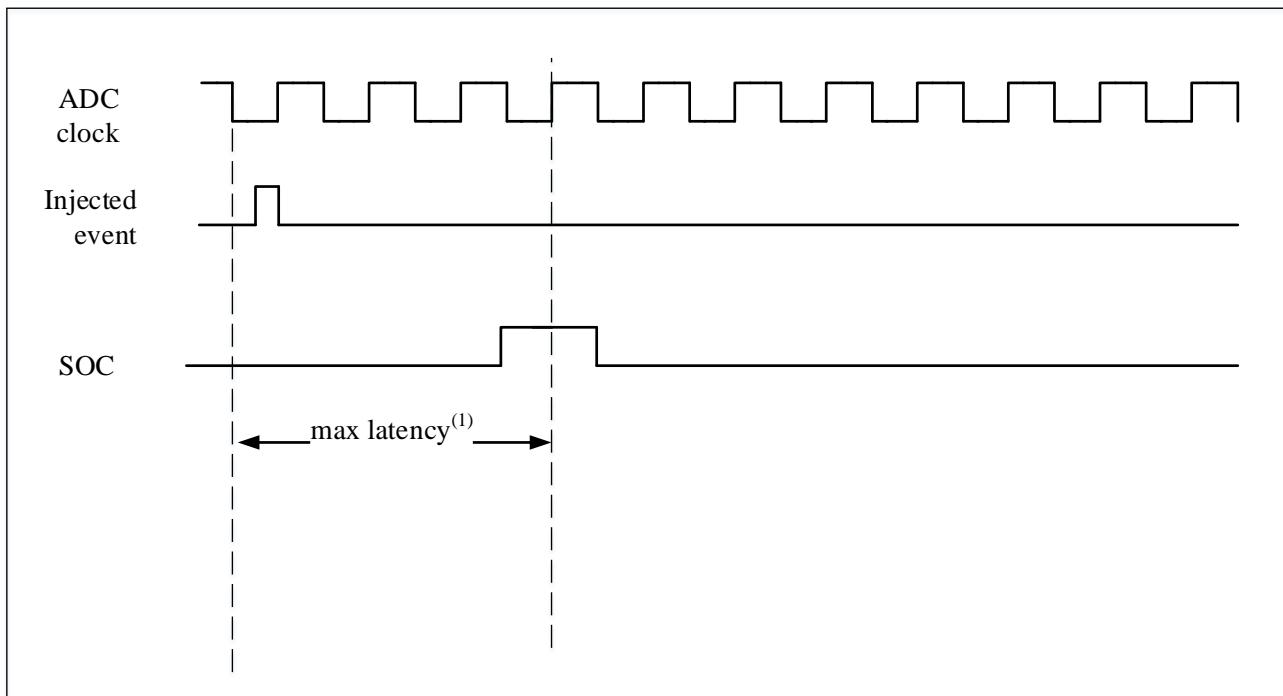
当 ADC 时钟预分频因子为 2 时，当转换序列从规则变为注入或注入变为规则时，会有 2 个 ADC 时钟间隔的延迟。当 ADC 时钟预分频因子为 4 到 8 时，当转换序列从规则变为注入或注入变为规则时，会有 1 个 ADC 时钟间隔的延迟。

#### 15.3.10.2 触发注入

将 ADC\_CTRL1.AUTOJC 设置为 0 并将 ADC\_CTRL1.SCAMD 设置为 1 以开启触发注入功能。在此功能中，通过设置 ADC\_CTRL2.ON 或通过外部触发连续转换的规则通道。规则通道转换时，如果产生外部注入触发，则暂停当前转换，注入序列通道开始转换。当注入序列通道转换完成后，将恢复中断的规则序列通道转换。如果在注入转换过程中产生了规则事件，则规则序列通道将在注入序列通道转换完成后开始转换。

使用此功能时，注入通道触发的时间间隔需要大于注入序列完成转换所需的时间。

图 15-5 注入转换延时



(1) 最大延迟值请参考数据手册中的电气特性部分。

### 15.3.11 间断模式

#### 15.3.11.1 规则通道

配置 ADC\_CTRL1.DREGCH 为 1，开启规则通道的间断模式，通过配置 ADC\_RSEQ1、ADC\_RSEQ2、ADC\_RSEQ3 获取规则转换序列，配置 ADC\_CTRL1.DCTU[2:0]控制每次触发后转换 n 个通道。

当触发信号产生时，对规则序列的 n 个通道进行转换然后停止，直到下一个触发信号产生，从上一次转换停止的地方开始继续转换 n 个通道，直到规则序列的所有通道被转换（如果最后一个触发发生并且转换序列中的剩余通道小于 n，则只转换剩余未转换的通道并停止转换），并且转换结束标志位也将被设置为 1。当转换序列中所有通道的转换完成后，下一个触发信号出现时，再次从规则序列的第一个通道开始转换。

#### 15.3.11.2 注入通道

配置 ADC\_CTRL1.DJCH 为 1，开启注入通道的间断模式，通过配置 ADC\_JSEQ 获取注入转换序列。

当触发信号产生时，它将转换注入转换序列的 1 个通道，然后停止。直到下一个触发信号产生，它会从上一次转换停止的通道处继续转换 1 个通道，直到注入序列的所有通道都被转换，并且转换结束标志位也将设置为 1。当转换序列中所有通道的转换完成时，下一个触发信号出现时，再次从注入序列的第一个通道开始转换。

同时间只能设置规则转换和注入转换其中一种为间断模式，不能同时设置自动注入功模式和间断模式。

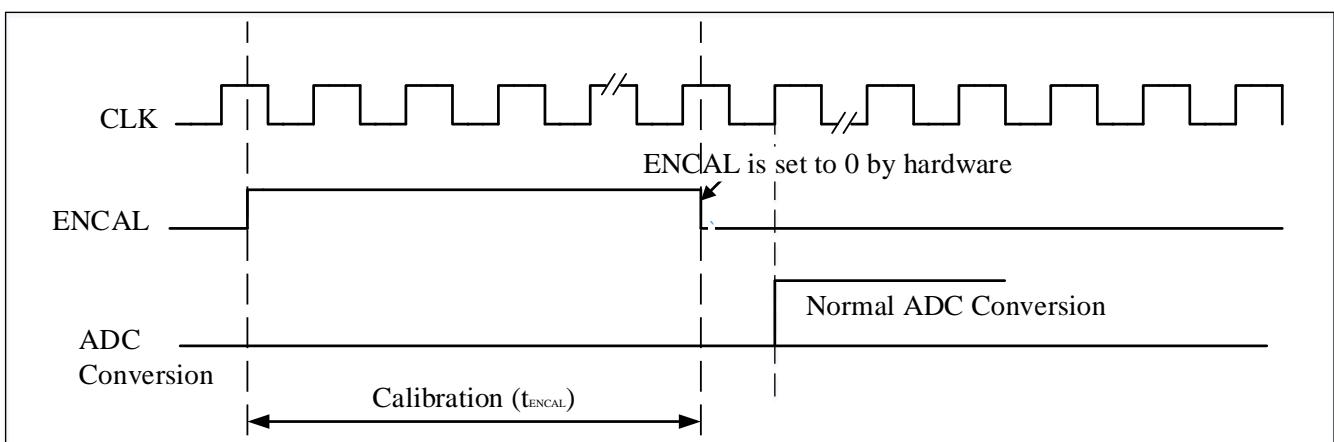
## 15.4 校准

为了减少误差，ADC 具有内置的自校准机制。在 A/D 转换之前，计算每个电容器的误差修正值。该值用于消除转换期间内部电容器组引起的误差。将 ADC\_CTRL2.ENCAL 位设置为 1 以启动自校准。在校准过程中，ADC\_CTRL2.ENCAL 位保持为 1。校准后，ADC\_CTRL2.ENCAL 位由硬件复位，然后可以开始 A/D 转换。

注意：

1. 建议每次上电后进行校准。如果 ADC 已经转换并处于连续转换模式，则无法完成校准操作。
2. 默认为单端校正，需要进行双端自动校正，必须设置 ADC\_CTRL3.CALDIF 为 1。然后将 ADC\_CTRL2.ENCAL 位写入 1，等待校正完成（校准完成后 ADC\_CTRL2.ENCAL 位自动清 0）

图 15-6 校准时序图



## 15.5 数据对齐

转换后的数据有两种对齐方式：左对齐和右对齐。对齐可以通过 ADC\_CTRL2.ALIG 位设置。ADC\_CTRL2.ALIG=0 为右对齐，如表 15-3 所示，ADC\_CTRL2.ALIG=1 为左对齐，如表 15-4 所示。

对于注入序列，SYM 位是扩展的符号值，寄存器中存储的数据是转换结果减去 ADC\_JOFFSETx 寄存器中用户定义的偏移量，所以结果可以是负值；对于规则序列，不需要减去偏移值。

表 15-3 数据右对齐

注入

(12bit)

SYM	SYM	SYM	SYM	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
-----	-----	-----	-----	-----	-----	----	----	----	----	----	----	----	----	----	----

(10bit)

SYM	SYM	SYM	SYM	SYM	SYM	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
-----	-----	-----	-----	-----	-----	----	----	----	----	----	----	----	----	----	----

(8bit)

SYM	D7	D6	D5	D4	D3	D2	D1	D0							
-----	-----	-----	-----	-----	-----	-----	-----	----	----	----	----	----	----	----	----

(6bit)

SYM	D5	D4	D3	D2	D1	D0								
-----	-----	-----	-----	-----	-----	-----	-----	-----	----	----	----	----	----	----

规则

(12bit)

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
---	---	---	---	-----	-----	----	----	----	----	----	----	----	----	----	----

(10bit)

0	0	0	0	0	0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----

(8bit)

0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0
---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----

(6bit)

0	0	0	0	0	0	0	0	0	D5	D4	D3	D2	D1	D0
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----

表 15-4 数据左对齐

注入

(12bit)

SYM	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0
-----	-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---

(10bit)

SYM	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0
-----	----	----	----	----	----	----	----	----	----	----	---	---	---	---	---

(8bit)

SYM	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0
-----	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---

(6bit)

SYM	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0	0	0
-----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---

规则

(12bit)

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
-----	-----	----	----	----	----	----	----	----	----	----	----	---	---	---	---

(10bit)

D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0
----	----	----	----	----	----	----	----	----	----	---	---	---	---	---	---

(8bit)

D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0	0
----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---

(6bit)

D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0	0	0	0	0
----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---	---

## 15.6 通道采样时间

ADC 使用若干个 ADC\_CLK 周期对输入电压采样，采样周期数目可以通过 ADC\_SAMPT1.SAMPx[2:0]和 ADC\_SAMPT2.SAMPx[2:0]位更改。每个通道可以分别用不同的时间采样。总转换时间如下计算：

$$T_{CONV} = \text{采样时间} + x \text{ 个周期}$$

注意：x 在分辨率分别为 6/8/10/12 bit 时对应的值为 6.5/8.5/10.5/12.5。

例如：

当 ADCCLK=80MHz，分辨率为 12bit，采样时间为 4.5 周期

$$T_{CONV} = 4.5 + 12.5 = 17 \text{ 周期} = 0.2125\mu\text{s}$$

## 15.7 外部触发转换

对于规则序列，软件将 ADC\_CTRL2.EXTRTRIG 位设置为 1，则规则通道可以使用外部事件的上升沿触发启动转换，然后软件通过配置 ADC\_CTRL2.EXTRSEL[2:0]来选择规则序列的外部触发源。外部触发源选择如下表所示。如果选择 EXTI\_line0~15 或 TIM8\_TRGO 作为外部触发源，可以设置 AFIO\_RMP\_CFG.ADC\_ETRR 和 AFIO\_RMP\_CFG EXTI\_ETRR[3:0]位来实现；如果选择 SWSTRRCH 作为外部触发源，则可以通过将 ADC\_CTRL2.SWSTRRCH 设置为 1 来启动规则通道转换。

表 15-5 ADC 用于规则通道的外部触发

EXTRSEL[2:0]	触发源	类型
000	TIM1_CC1 事件	来自片上定时器的内部信号
001	TIM1_CC2 事件	
010	TIM1_CC3 事件	
011	TIM2_CC2 事件	
100	TIM3_TRGO 事件	
101	TIM4_CC4 事件	
110	EXTI line 0~15/TIM8_TRGO 事件	外部引脚/来自片上定时器的内部信号
111	SWSTRRCH	软件控制位

对于注入序列，软件将 ADC\_CTRL2.EXTJTRIG 位设置为 1，则注入通道可以使用外部事件的上升沿触发启动转换，软件将通过配置 ADC\_CTRL2.EXTJSEL[2:0]选择注入序列的外部触发源。外部触发源选择如下表所示。如果选择 EXTI\_line0~15 或 TIM8\_CC4 作为外部触发源，可以设置 AFIO\_RMP\_CFG.ADC\_ETRI 和 AFIO\_RMP\_CFG EXTI\_ETRI[3:0]位来实现；如果选择 SWSTRJCH 作为外部触发源，则可以通过将 ADC\_CTRL2.SWSTRJCH 设置为 1 来启动注入通道转换。

表 15-6 ADC 用于注入通道的外部触发

EXTJSEL[2:0]	触发源	类型
000	TIM1_TRGO 事件	来自片上定时器的内部信号

EXTJSEL[2:0]	触发源	类型
001	TIM1_CC4 事件	
010	TIM2_TRGO 事件	
011	TIM2_CC1 事件	
100	TIM3_CC4 事件	
101	TIM4_TRGO 事件	
110	EXTI line 0~15/TIM8_CC4 事件	外部引脚/来自片上定时器的内部信号
111	SWSTRJCH	软件控制位

注意：注入触发可以打断规则转换

## 15.8 DMA 请求

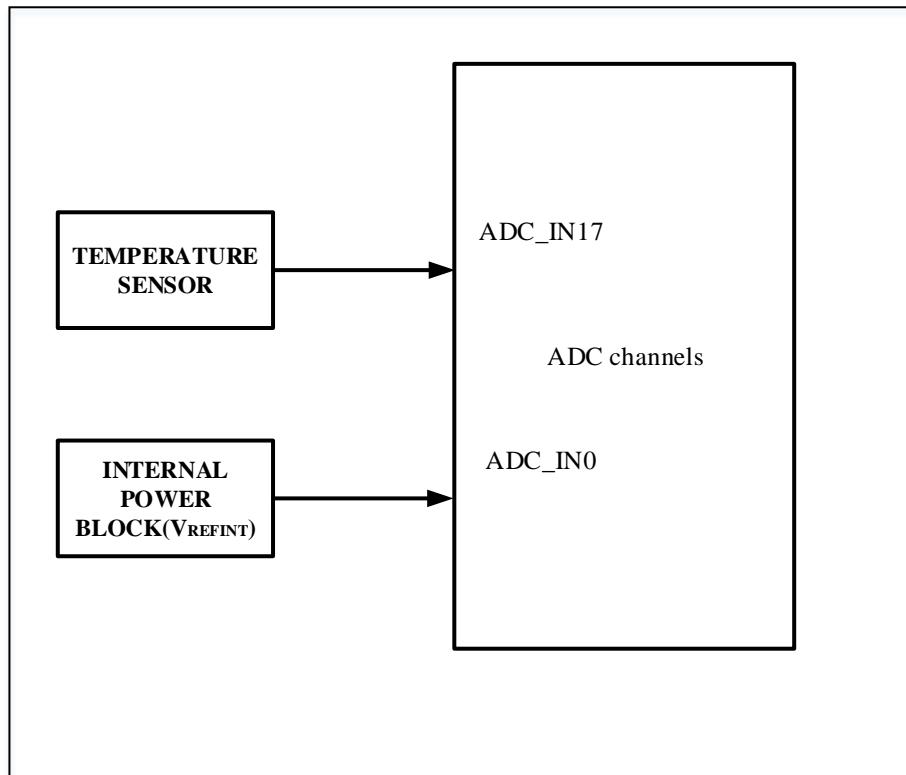
为避免多个规则通道转换时数据过多，导致 ADC\_DAT 寄存器中保存的规则通道转换结果丢失，可以将 ADC\_CTRL2.ENDMA 位设置为 1，以此使用 DMA。当 ADC 规则通道转换结束时，会产生一个 DMA 请求。DMA 收到请求后，会将转换后的数据从 ADC\_DAT 寄存器传送到用户指定的目标地址。

## 15.9 温度传感器

设置 ADC\_CTRL2.TEMPEN 位为 1，使能温度传感器和 VREFINT，设备工作时使用温度传感器检测环境温度。温度传感器采样的输出电压通过 ADC\_IN17 通道转换为数字值。温度传感器工作时，理想的采样时间为 17.1us；当温度传感器不工作时，ADC\_CTRL2.TEMPEN 位可通过软件清零以降低功耗。图 15-7 是温度传感器的框图。

温度传感器的输出电压随温度线性变化。不同的芯片由于生产工艺的不同，在温度曲线上会有不同的偏移量。通过测试，发现最大偏移为 3 ℃。这一特性使得内部温度传感器更适合检测温度变化。不适合测量绝对温度。当需要精确的温度测量时，应使用外部温度传感器。

图 15-7 温度传感器和 V<sub>REFINT</sub> 通道框图



### 15.9.1 测量温度值

1. 配置通道 (ADC\_IN17) 和通道的采样时间为 17.1us
2. 将 ADC\_CTRL2.TEMPEN 位设置为 1 以启用温度传感器和 VREFINT
3. 设置 ADC\_CTRL2.ON 位为 1 以启动 ADC 转换 (或通过外部触发)
4. 读取 ADC 数据寄存器中的温度数据，通过以下公式计算温度值：

$$\text{温度 } (\text{ }^\circ\text{C}) = \{(V_{\text{Temperature}} - V_{\text{SENSE}}) / \text{Avg\_Slope}\} + \text{Temperature\_T}_{\text{offset}}$$

其中：

$V_{\text{Temperature}}$  = Temperature 对应的  $V_{\text{SENSE}}$

$T_{\text{offset}} = 1.25 \text{ }^\circ\text{C}$

Temperature 为校准温度

Avg\_Slope = 温度和  $V_{\text{SENSE}}$  曲线的平均斜率 ( $\text{mV/ }^\circ\text{C}$  或  $\mu\text{V/ }^\circ\text{C}$ )

参考数据手册的电气特性章节中 Avg\_Slope 的实际值。

注意：在传感器从断电模式唤醒到正确输出  $V_{\text{SENSE}}$  之前，有一个建立时间；ADC 上电后还有一个建立时间，所以为了缩短延迟，ADC\_CTRL2.TEMPEN 和 ADC\_CTRL2.ON 位应该同时置位。

## 15.10 中断

ADC 中断可以来自规则或注入序列转换的结束、输入电压不在模拟看门狗设置的范围、规则或注入通道转换的任何结束。这些中断具有独立的中断使能位。

ADC\_STS 寄存器中有 2 个状态标志：注入序列通道转换启动(JSTR)和规则序列通道转换启动(STR)。但是 ADC 中没有与这两个标志相关的中断。

表 15-7 ADC 中断

中断事件	事件标志	使能控制位
规则或注入序列转换结束	ENDC	ENDCIEN
注入序列转换结束	JENDC	JENDCIEN
超出模拟看门狗阈值	AWDG	AWDGIEN
任何通道转换结束	ENDCA	ENDCAIEN
任何注入通道转换结束	JENDCA	JENDCAIEN

## 15.11 ADC 寄存器

### 15.11.1 ADC 寄存器总览

表 15-8 ADC 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
000h	ADC_STS																																		
	Reset Value																																		
004h	ADC_CTRL1																																		
	Reset Value																																		
008h	ADC_CTRL2																																		
	Reset Value																																		
00Ch	ADC_SAMPT1																																		
	Reset Value																																		
010h	ADC_SAMPT2																																		
	Reset Value																																		
014h	ADC_JOFFSET1																																		
	Reset Value																																		
018h	ADC_JOFFSET2																																		
	Reset Value																																		
01Ch	ADC_JOFFSET3																																		
	Reset Value																																		
020h	ADC_JOFFSET4																																		
	Reset Value																																		
024h	ADC_WDGHIGH																																		
	Reset Value																																		

028h	ADC_WDGLOW	Reserved								LTH[11:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
02Ch	ADC_RSEQ1	Reserved				LEN[3:0]		SEQ16[4:0]			SEQ15[4:0]			SEQ14[4:0]			SEQ13[4:0]			
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	
030h	ADC_RSEQ2	Reserved	SEQ12[4:0]			SEQ11[4:0]			SEQ10[4:0]			SEQ9[4:0]			SEQ8[4:0]			SEQ7[4:0]		
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
034h	ADC_RSEQ3	Reserved	SEQ6[4:0]			SEQ5[4:0]			SEQ4[4:0]			SEQ3[4:0]			SEQ2[4:0]			SEQ1[4:0]		
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
038h	ADC_JSEQ	Reserved				JLEN[1:0]	JSEQ4[4:0]			JSEQ3[4:0]			JSEQ2[4:0]			JSEQ1[4:0]			Reserved	
	Reset Value						0	0	0	0	0	0	0	0	0	0	0	0	0	
03Ch	ADC_JDAT1	Reserved								JDAT1[15:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
040h	ADC_JDAT2	Reserved								JDAT2[15:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
044h	ADC_JDAT3	Reserved								JDAT3[15:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
048h	ADC_JDAT4	Reserved								JDAT4[15:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
04Ch	ADC_DAT	Reserved								DAT[15:0]										
	Reset Value									0	0	0	0	0	0	0	0	0	0	
050h	ADC_DIFSEL	Reserved				DIFSEL[18:0]														
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reserved
054h	ADC_CALFACT	Reserved	CALFACTD[6:0]				Reserved												CALFACTS[6:0]	
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
058h	ADC_CTRL3	Reserved								VBTMEN	DPWMOD	JENDCAEN	ENDCAEN	BPCAL	PDRDY	RDY	CKMOD	CALALD	CALDF	
	Reset Value									0	0	0	0	0	0	0	0	0	RES1[1:0]	
05Ch	ADC_SAMPT3	Reserved													SAMPSEL	SAMPT[2:0]			0	
	Reset Value														0	0	0	0	0	

## 15.11.2 ADC 状态寄存器(ADC\_STS)

地址偏移: 0x00

复位值: 0x0000 0000

31	Reserved															16
15	Reserved							JENDCA	ENDCA	STR	JSTR	JENDC	ENDC	AWDG		
	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0									

位域	名称	描述
31:7	Reserved	保留，必需保持复位值。
6	JENDCA	任意注入通道转换结束位 (Any Injected channel End of conversion flag) 在任意注入通道转换结束时被硬件设置为1，由软件清除。 0: 转换没有完成； 1: 转换完成。
5	ENDCA	任意通道转换结束位 (Any End of conversion flag)

位域	名称	描述
		任意规则或注入通道转换结束时被硬件设置为1，由软件清除。 0: 转换没有完成； 1: 转换完成。
4	STR	规则通道开始位 (Regular channel Start flag) 规则通道转换开始时该位被硬件设置为1，由软件清除。 0: 规则通道转换未开始； 1: 规则通道转换已开始。
3	JSTR	注入通道开始位 (Injected channel Start flag) 注入通道序列转换开始时被硬件设置为1，由软件清除。 0: 注入通道序列转换未开始； 1: 注入通道序列转换已开始。
2	JENDC	注入通道转换结束位 (Injected channel end of conversion) 所有注入通道序列转换结束时被硬件设置为1，由软件清除 0: 转换未完成； 1: 转换完成。
1	ENDC	转换结束位 (End of conversion) 规则或注入通道序列转换结束时被硬件设置为1，由软件清除 0: 转换未完成； 1: 转换完成。
0	AWDG	模拟看门狗标志位 (Analog watchdog flag) 转换的电压值超出了ADC_LTR和ADC_HTR寄存器定义的范围时被硬件设置为1，由软件清除 0: 没有发生模拟看门狗事件； 1: 发生模拟看门狗事件。

### 15.11.3 ADC 控制寄存器 1(ADC\_CTRL1)

地址偏移: 0x04

复位值: 0x0000 0000

31	Reserved								24	23	22	21	20	19	16
									AWDG_ERCH	AWDG_EJCH	Reserved	Reserved	DUSEL[3:0]		
15	DCTU[2:0]	DJCH	DREGCH	AUTOJC	AWDG_SGLEN	SCANMDD	JEENDC_IEN	AWD_GIEN	ENDCIEN		AWDGCH[4:0]			rW	0
	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW					

位域	名称	描述
31:24	Reserved	保留，必需保持复位值。
23	AWDGERCH	在规则通道上开启模拟看门狗 (Analog watchdog enable on regular channels) 该位由软件设置和清除。 0: 在规则通道上关闭模拟看门狗； 1: 在规则通道上开启模拟看门狗。
22	AWDGEJCH	在注入通道上开启模拟看门狗 (Analog watchdog enable on injected channels)

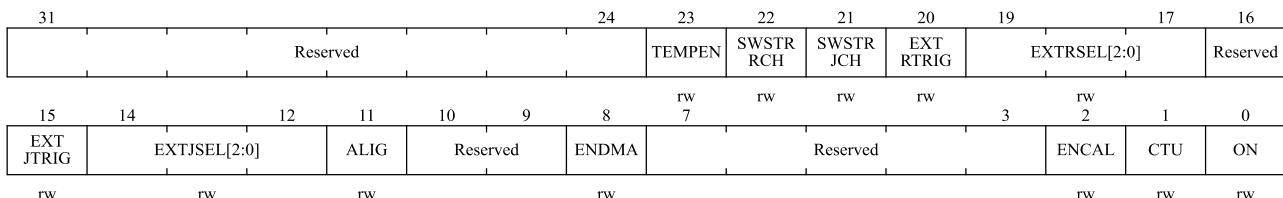
位域	名称	描述
		该位由软件设置和清除。 0: 在注入通道上关闭模拟看门狗; 1: 在注入通道上开启模拟看门狗。
21:20	Reserved	保留, 必需保持复位值。
19:16	DUSEL[3:0]	双模式选择 软件使用这些位来选择操作模式。本芯片不可用, 需保持在独立模式。 0000: 独立模式 0001-1111: 保留
15:13	DCTU[2:0]	间断模式通道计数 (Discontinuous mode channel count) 软件通过这些位定义在间断模式下, 收到外部触发后转换规则通道的数目 000: 1个通道 001: 2个通道 ..... 111: 8个通道
12	DJCH	在注入通道上的间断模式 (Discontinuous mode on injected channels) 该位由软件设置和清除, 用于开启或关闭注入通道序列上的间断模式 0: 注入通道序列上禁用间断模式; 1: 注入通道序列上使用间断模式。
11	DREGCH	在规则通道上的间断模式 (Discontinuous mode on regular channels) 该位由软件设置和清除, 用于开启或关闭规则通道序列上的间断模式 0: 规则通道序列上禁用间断模式; 1: 规则通道序列上使用间断模式。
10	AUTOJC	自动的注入通道序列转换 (Automatic Injected Group conversion) 该位由软件设置和清除, 用于规则通道序列转换结束后注入通道序列转换的开启或关闭 0: 关闭自动的注入通道序列转换; 1: 开启自动的注入通道序列转换。
9	AWDGSGLEN	扫描模式中在一个单一的通道上使用看门狗 (Enable the watchdog on a single channel in scan mode) 该位由软件设置和清除, 用于AWDGCH[4:0]位指定的通道上的模拟看门狗功能开启或所有通道上模拟看门狗功能开启 0: 在所有的通道上使用模拟看门狗; 1: 在单一通道上使用模拟看门狗。
8	SCANMD	扫描模式 (Scan mode) 该位由软件设置和清除以启用或禁用扫描模式。在扫描模式下, 转换由ADC_RSEQx或ADC_JSEQ 寄存器选定的通道。 0: 禁用扫描模式。 1: 启用扫描模式。 注意: 如果单独设置 ADC_CTRL1.ENDCIEN 或 ADC_CTRL1.JENDCIEN 位, ADC_STS.ENDC 或 ADC_STS.JENDC 中断仅在最后一个通道转换后发生。
7	JENDCIEN	注入通道的中断使能 (Interrupt enable for injected channels) 该位由软件设置和清除, 以在所有注入通道转换完成后禁止或允许中断。 0: 禁止JENDC中断。

位域	名称	描述
		1: 使能JENDC中断。硬件设置ADC_STS.JENDC位时发生中断。
6	AWDGIEN	模拟看门狗中断使能 (Analog watchdog interrupt enable) 该位由软件设置和清除以禁止或允许模拟看门狗产生中断。在扫描模式下，如果看门狗检测到超出范围的值，则仅当该位置位时才会中止扫描。 0: 禁止模拟看门狗中断。 1: 使能模拟看门狗中断。
5	ENDCIEN	ENDC的中断使能 (Interrupt enable for ENDC) 该位由软件设置和清除，以禁止或允许在规则或注入转换序列转换结束后发生中断。 0: 禁止ENDC中断。 1: 使能ENDC中断。硬件设置ADC_STS.ENDC位时发生中断。
4:0	AWDGCH[4:0]	模拟看门狗通道选择位 (Analog watchdog channel select bits) 这些位由软件设置和清除以选择模拟看门狗保护的输入通道。 00000: ADC模拟输入通道0 00001: ADC模拟输入通道1 ... 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 保留所有其他值。

#### 15.11.4 ADC 控制寄存器 2(ADC\_CTRL2)

偏移地址: 0x08

复位值: 0x0000 0000



位域	名称	描述
31:24	Reserved	保留，必需保持复位值。
23	TEMPEN	温度传感器和V <sub>REFINT</sub> 开启 (Temperature sensor and V <sub>REFINT</sub> enable) 该位由软件设置和清除以开启或关闭温度传感器和V <sub>REFINT</sub> 通道。 0: 关闭温度传感器和V <sub>REFINT</sub> 。 1: 开启温度传感器和V <sub>REFINT</sub> 。
22	SWSTRRCH	开始转换规则通道 (Start conversion of regular channels) 该位由软件设置以启动转换，并在转换开始后由硬件清零。如果在

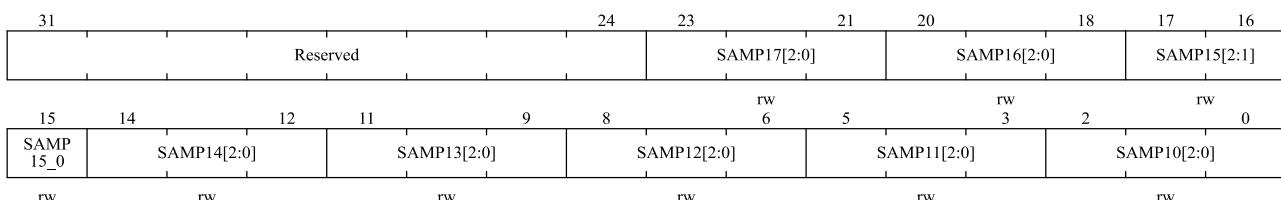
位域	名称	描述
		ADC_CTRL2.EXTRSEL[2:0] 位中选择 SWSTRRCH 作为触发事件，该位用于启动一组规则通道的转换 0: 复位状态。 1: 开始转换规则通道。
21	SWSTRJCH	开始转换注入通道 (Start conversion of injected channels) 该位由软件设置以启动转换，并且可以在转换开始后立即由软件或硬件清零。如果在 ADC_CTRL2.EXTJSEL[2:0]位中选择 SWSTRJCH作为触发事件，该位用于启动一组注入通道的转换 0: 复位状态。 1: 开始转换注入通道。
20	EXTRTRIG	规则通道的外触发转换模式 (External trigger conversion mode for regular channels) 该位由软件设置和清零，以启用或禁用可以启动规则序列转换的外部触发事件。 0: 不使用外部事件启动转换。 1: 使用外部事件启动转换。
19:17	EXTRSEL[2:0]	选择启动规则通道序列转换的外部事件 (External event select for regular group) 这些位选择外部事件以启动规则序列转换 ADC的触发配置如下 000: 定时器1的CC1事件 100: 定时器3的TRGO事件 001: 定时器1的CC2事件 101: 定时器4的CC4事件 010: 定时器1的CC3事件 110: EXTI_line0~15/TIM8_TRGO事件 011: 定时器2的CC2事件 111: SWSTRRCH
16	Reserved	保留，必需保持复位值。
15	EXTJTRIG	注入通道的外部触发转换模式 (External trigger conversion mode for injected channels) 该位由软件设置和清零，以启用或禁用可以启动注入序列转换的外部触发事件。 0: 不使用外部事件启动转换。 1: 使用外部事件启动转换。
14:12	EXTJSEL[2:0]	选择启动注入通道序列转换的外部事件 (External event select for injected group) 这些位选择用于触发注入序列转换的外部事件。 ADC的触发配置如下 000: 定时器1的TRGO事件 100: 定时器3的CC4事件 001: 定时器1的CC4事件 101: 定时器4的TRGO事件 010: 定时器2的TRGO事件 110: EXTI_line0~15/TIM8_CC4事件 011: 定时器2的CC1事件 111: SWSTRJCH
11	ALIG	数据对齐 (Data alignment) 该位由软件设置和清除。请参阅表15-3和表15-4 0: 右对齐。 1: 左对齐。
10:9	Reserved	保留，必需保持复位值。
8	ENDMA	直接存储器访问模式 (Direct memory access mode) 该位由软件设置和清除。有关详细信息，请参见DMA控制器章节。 0: 不使用DMA模式。 1: 使用DMA模式。

位域	名称	描述
7:3	Reserved	保留, 必需保持复位值。
2	ENCAL	A/D 校准 (A/D Calibration) 该位由软件设置以开始校准, 并在校准结束时由硬件清零。 0: 校准完成; 1: 开始校准。
1	CTU	连续转换 (Continuous conversion) 该位由软件设置和清除。如果该位置位, 则转换将继续, 直到该位被清除。 0: 单次转换模式。 1: 连续转换模式。
0	ON	A/D 转换器开/关 (A/D converter ON/OFF) 该位由软件设置和清除。当该位为“0”时, 写入“1”会将ADC从断电模式中唤醒。 当该位为“1”时, 写入“1”开始转换。应注意, 转换器上电与转换开始之间存在延迟tSTAB, 请参见图15-4 0: 关闭ADC转换/校准并进入掉电模式。 1: 启动ADC并开始转换。 <b>注意:</b> 如果该寄存器中的其他位随着ON发生变化, 则不会触发转换。这是为了防止触发错误的转换。

### 15.11.5 ADC 采样时间寄存器 1(ADC\_SAMPT1)

偏移地址: 0x0C

复位值: 0x0000 0000



位域	名称	描述
31:24	Reserved	保留, 必需保持复位值。
23:0	SAMPx[2:0]	通道x采样时间选择 (Channel x Sample time selection) 这些位用于独立选择每个通道的采样时间。通道选择位在采样期间必须保持不变。 ADC_SAMPT3.SAMPSEL = 0, 采样时间设置如下: 000: 1.5周期 100: 41.5周期 001: 7.5周期 101: 55.5周期 010: 13.5周期 110: 71.5周期 011: 28.5周期 111: 239.5周期 ADC_SAMPT3.SAMPSEL = 1, 采样时间设置如下: 000: 1.5周期 100: 19.5周期 001: 2.5周期 101: 61.5周期 010: 4.5周期 110: 181.5周期 011: 7.5周期 111: 601.5周期

### 15.11.6 ADC 采样时间寄存器 2(ADC\_SAMPT2)

地址偏移: 0x10

复位值: 0x0000 0000

31	30	29	27	26	24	23	21	20	18	17	16
Reserved		SAMP9[2:0]		SAMP8[2:0]		SAMP7[2:0]		SAMP6[2:0]		SAMP5[2:1]	
15	14	rw	12	11	rw	9	8	rw	6	5	rw
SAMP5_0		SAMP4[2:0]		SAMP3[2:0]		SAMP2[2:0]		SAMP1[2:0]		SAMP0[2:0]	
		rw									

位域	名称	描述																
31:30	Reserved	保留, 必需保持复位值。																
29:0	SAMPx[2:0]	<p>通道x采样时间选择 (Channel x Sample time selection)          这些位用于独立选择每个通道的采样时间。通道选择位在采样期间必须保持不变。          ADC_SAMPT3.SAMPSEL = 0, 采样时间设置如下:</p> <table> <tr><td>000: 1.5周期</td><td>100: 41.5周期</td></tr> <tr><td>001: 7.5周期</td><td>101: 55.5周期</td></tr> <tr><td>010: 13.5周期</td><td>110: 71.5周期</td></tr> <tr><td>011: 28.5周期</td><td>111: 239.5周期</td></tr> </table> <p>ADC_SAMPT3.SAMPSEL = 1, 采样时间设置如下:</p> <table> <tr><td>000: 1.5周期</td><td>100: 19.5周期</td></tr> <tr><td>001: 2.5周期</td><td>101: 61.5周期</td></tr> <tr><td>010: 4.5周期</td><td>110: 181.5周期</td></tr> <tr><td>011: 7.5周期</td><td>111: 601.5周期</td></tr> </table>	000: 1.5周期	100: 41.5周期	001: 7.5周期	101: 55.5周期	010: 13.5周期	110: 71.5周期	011: 28.5周期	111: 239.5周期	000: 1.5周期	100: 19.5周期	001: 2.5周期	101: 61.5周期	010: 4.5周期	110: 181.5周期	011: 7.5周期	111: 601.5周期
000: 1.5周期	100: 41.5周期																	
001: 7.5周期	101: 55.5周期																	
010: 13.5周期	110: 71.5周期																	
011: 28.5周期	111: 239.5周期																	
000: 1.5周期	100: 19.5周期																	
001: 2.5周期	101: 61.5周期																	
010: 4.5周期	110: 181.5周期																	
011: 7.5周期	111: 601.5周期																	

### 15.11.7 ADC 注入通道数据偏移寄存器 x(ADC\_JOFFSETx) (x=1...4)

地址偏移: 0x14-0x20

复位值: 0x0000 0000

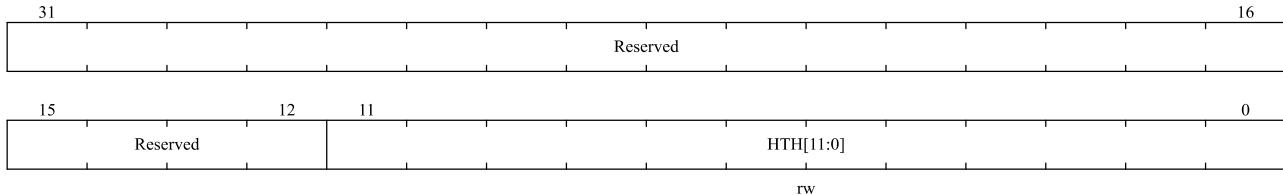
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

位域	名称	描述
31:12	Reserved	保留, 必需保持复位值。
11:0	OFFSETJCHx[11:0]	<p>注入通道x的数据偏移 (Data offset for injected channel x)          这些位定义在将转换注入通道时用于从原始转换数据中减去的值。转换结果可以在ADC_JDATx寄存器中读取。</p>

### 15.11.8 ADC 看门狗高阈值寄存器(ADC\_WDGHIGH)

地址偏移: 0x24

复位值: 0x0000 0FFF

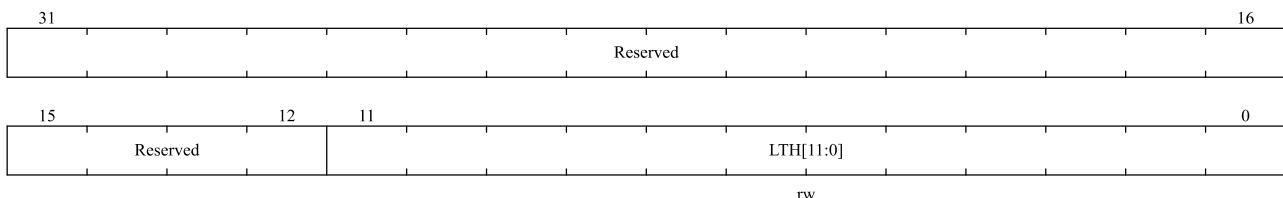


位域	名称	描述
31:12	Reserved	保留, 必需保持复位值。
11:0	HTH[11:0]	模拟看门狗高阈值 (Analog watchdog high threshold) 这些位定义模拟看门狗的高阈值。

### 15.11.9 ADC 看门狗低阈值寄存器(ADC\_WDGLOW)

地址偏移: 0x28

复位值: 0x0000 0000

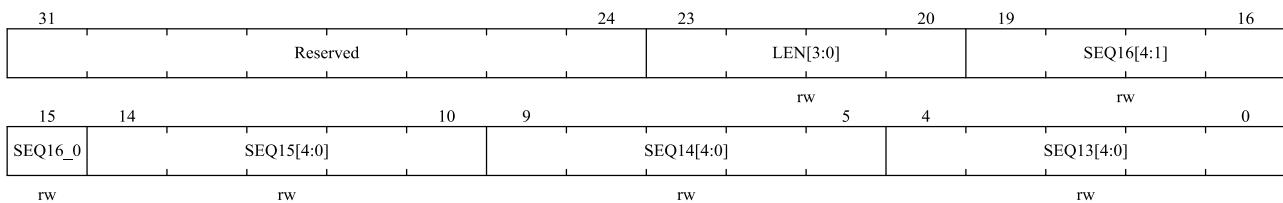


位域	名称	描述
31:12	Reserved	保留, 必需保持复位值。
11:0	LTH[11:0]	模拟看门狗低阈值 (Analog watchdog low threshold) 这些位定义模拟看门狗的低阈值。

### 15.11.10 ADC 规则序列寄存器 1(ADC\_RSEQ1)

地址偏移: 0x2C

复位值: 0x0000 0000

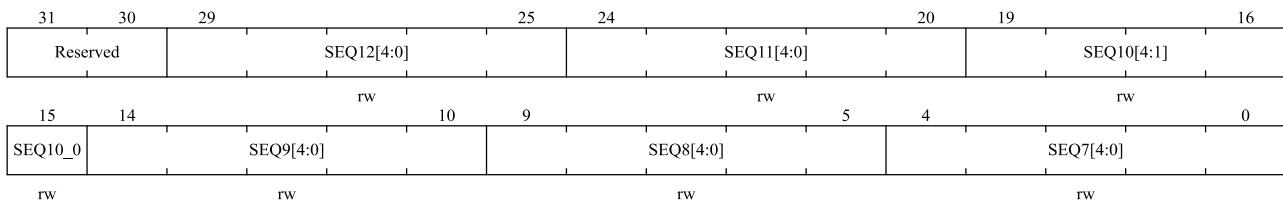


位域	名称	描述
31:24	Reserved	保留，必需保持复位值。
23:20	LEN[3:0]	规则通道序列长度 (Regular channel sequence length) 这些位由软件定义规则序列通道转换中的通道数。 0000: 1个转换 0001: 2个转换 ... 1111: 16个转换
19:15	SEQ16[4:0]	规则序列中的第16个转换 (16th conversion in regular sequence) 这些位由软件定义转换序列中第16个转换通道的编号 (0到18)。
14:10	SEQ15[4:0]	规则序列中的第15个转换 (15th conversion in regular sequence)
9:5	SEQ14[4:0]	规则序列中的第14个转换 (14th conversion in regular sequence)
4:0	SEQ13[4:0]	规则序列中的第13个转换 (13th conversion in regular sequence)

### 15.11.11 ADC 规则序列寄存器 2(ADC\_RSEQ2)

地址偏移: 0x30

复位值: 0x0000 0000

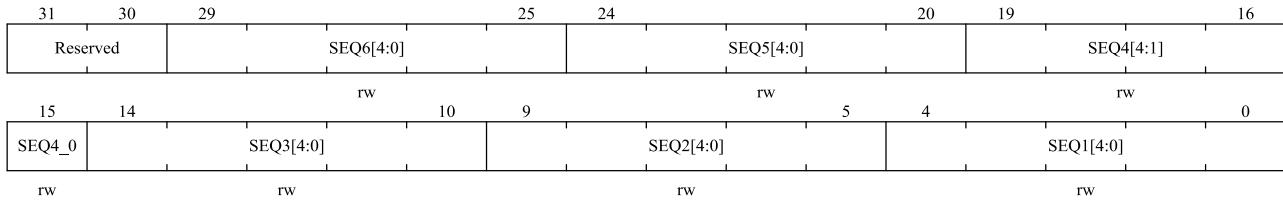


位域	名称	描述
31:30	Reserved	保留，必需保持复位值。
29:25	SEQ12[4:0]	规则序列中的第12个转换 (12th conversion in regular sequence) 这些位由软件定义转换序列中第12个转换通道的编号 (0到18)。
24:20	SEQ11[4:0]	规则序列中的第11个转换 (11th conversion in regular sequence)
19:15	SEQ10[4:0]	规则序列中的第10个转换 (10th conversion in regular sequence)
14:10	SEQ9[4:0]	规则序列中的第9个转换 (9th conversion in regular sequence)
9:5	SEQ8[4:0]	规则序列中的第8个转换 (8th conversion in regular sequence)
4:0	SEQ7[4:0]	规则序列中的第7个转换 (7th conversion in regular sequence)

### 15.11.12 ADC 规则序列寄存器 3(ADC\_RSEQ3)

地址偏移: 0x34

复位值: 0x0000 0000

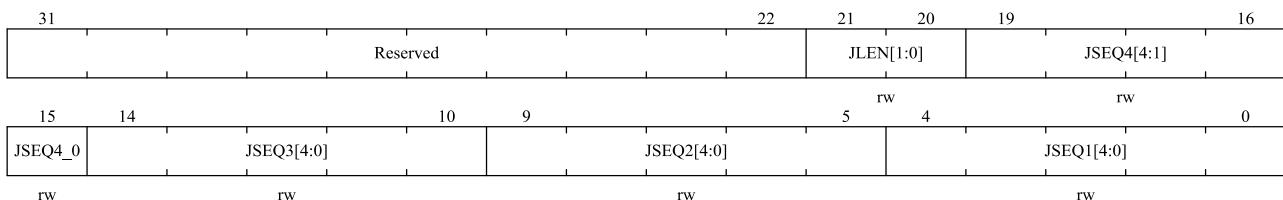


位域	名称	描述
31:30	Reserved	保留，必需保持复位值。
29:25	SEQ6[4:0]	规则序列中的第6个转换 (6th conversion in regular sequence) 这些位由软件定义转换序列中第6个转换通道的编号 (0到18)。
24:20	SEQ5[4:0]	规则序列中的第5个转换 (5th conversion in regular sequence)
19:15	SEQ4[4:0]	规则序列中的第4个转换 (4th conversion in regular sequence)
14:10	SEQ3[4:0]	规则序列中的第3个转换 (3rd conversion in regular sequence)
9:5	SEQ2[4:0]	规则序列中的第2个转换 (2nd conversion in regular sequence)
4:0	SEQ1[4:0]	规则序列中的第1个转换 (1st conversion in regular sequence)

### 15.11.13 ADC 注入序列寄存器(ADC\_JSEQ)

地址偏移: 0x38

复位值: 0x0000 0000

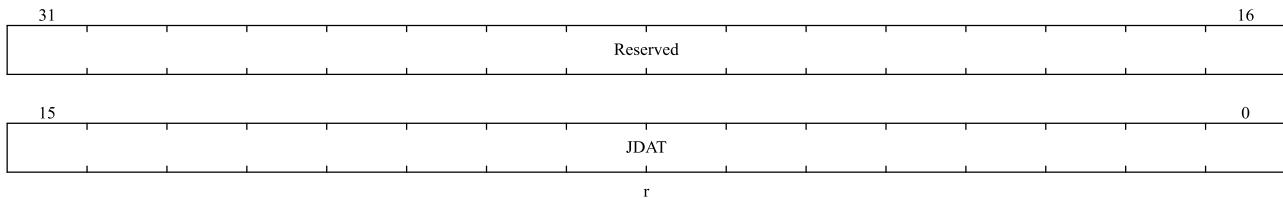


位域	名称	描述
31:22	Reserved	保留，必需保持复位值。
21:20	JLEN[1:0]	注入通道序列长度 (Injected sequence length) 这些位由软件定义为注入通道转换序列中的通道数。 00: 1个转换 01: 2个转换 10: 3个转换 11: 4个转换
19:15	JSEQ4[4:0]	注入序列中的第4个转换 (4th conversion in injected sequence) 这些位由软件定义为转换序列中第4个转换通道的编号 (0 到 17)。 注意：与规则转换序列不同，如果ADC_JSEQ.JLEN[1:0]的长度小于4，则转换序列从(4-JLEN)开始。例如，ADC_JSEQ[21:0] = 10 00011 00011 00111 00010 表示扫描转换将按照以下通道顺序进行转换：7、3、3 而不是 2、7、3。
14:10	JSEQ3[4:0]	注入序列中的第3个转换 (3rd conversion in injected sequence)
9:5	JSEQ2[4:0]	注入序列中的第2个转换 (2nd conversion in injected sequence)
4:0	JSEQ1[4:0]	注入序列中的第1个转换 (1st conversion in injected sequence)

### 15.11.14 ADC 注入数据寄存器 x(ADC\_JDATx) (x= 1...4)

地址偏移: 0x3C - 0x48

复位值: 0x0000 0000

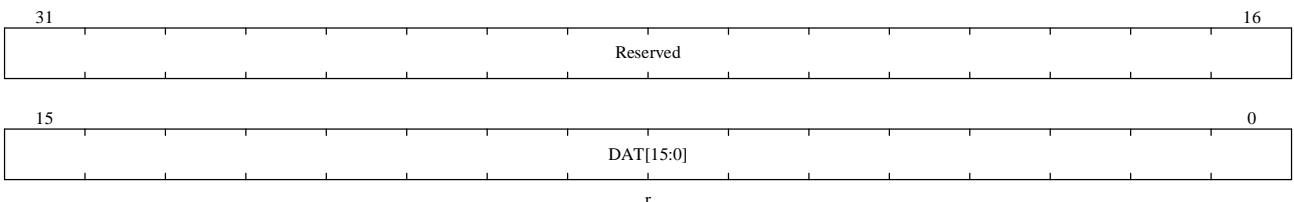


位域	名称	描述
31:16	Reserved	保留, 必需保持复位值。
15:0	JDAT[15:0]	注入转换的数据 (Injected data) 这些位是只读的, 包含注入通道的转换结果。数据左对齐或右对齐。

### 15.11.15 ADC 规则数据寄存器(ADC\_DAT)

地址偏移: 0x4C

复位值: 0x0000 0000

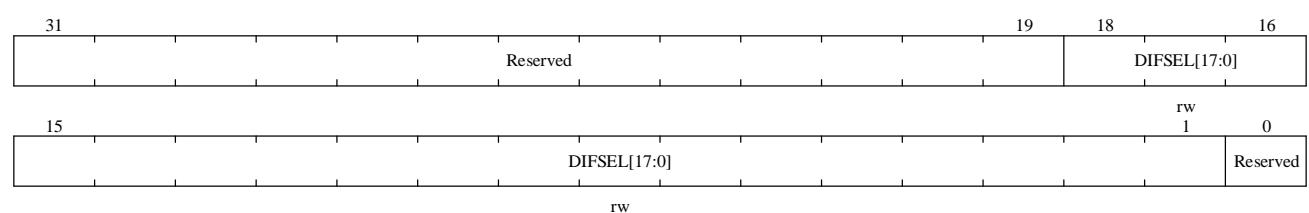


位域	名称	描述
31:16	Reserved	保留, 必需保持复位值。
15:0	DAT[15:0]	规则转换的数据 (Regular data) 这些位是只读的, 包含规则通道的转换结果。数据左对齐或右对齐, 如表15-3和表15-4所示。

### 15.11.16 ADC 差分模式选择寄存器(ADC\_DIFSEL)

地址偏移: 0x50

复位值: 0x0000 0000

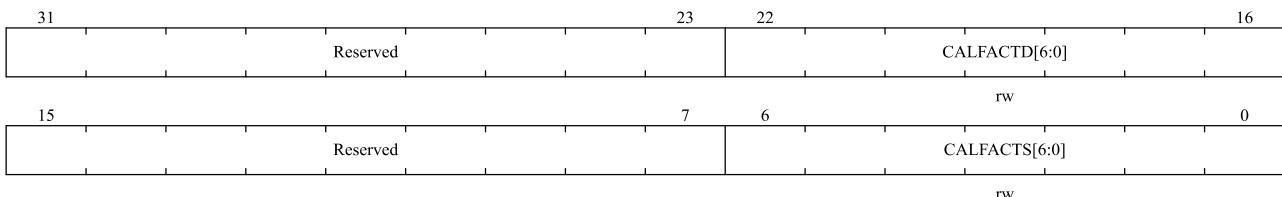


位域	名称	描述
31:19	Reserved	保留，必需保持复位值。
18:1	DIFSEL[17:0]	差分模式通道18-1选择 (Differential mode for channels 18 to 1) DIFSEL[i] = 0: ADC通道输入i+1配置为单端模式； DIFSEL[i] = 1: ADC 通道输入i+1配置为差分模式
0	Reserved	保留，必需保持复位值。

### 15.11.17 ADC 校正因子(ADC\_CALFACT)

地址偏移: 0x54

复位值: 0x0000 0000



位域	名称	描述
31:23	Reserved	保留，必需保持复位值。
22:16	CALFACTD[6:0]	差分模式校正因子 (Calibration Factors in differential mode) 该位可由硬件或软件写入 差分输入校准完成后，硬件会根据校准系数进行更新。 软件可以使用新的校准因子写入这些位。如果新的校准系数与模拟ADC中存储的当前系数不同，则将在启动新的差分校准时应用该系数。 注：软件只允许在ADC_CTRL2.ON = 1, ADC_STS.STR = 0, ADC_STS.JSTR = 0时写入 (ADC 没有处理转换或开始转换)
15:7	Reserved	保留，必需保持复位值。
6:0	CALFACTS[6:0]	单端模式校正因子 (Calibration Factors in Single-Ended mode) 该位可由硬件或软件写入 单端输入校准完成后，硬件会根据校准系数进行更新。 软件可以使用新的校准因子写入这些位。如果新的校准系数与模拟ADC中存储的当前系数不同，则将在启动新的单端校准时应用该系数。 注：软件只允许在ADC_CTRL2.ON = 1, ADC_STS.STR = 0, ADC_STS.JSTR = 0时写入 (ADC 没有处理转换或开始转换)

### 15.11.18 ADC 控制寄存器 3(ADC\_CTRL3)

地址偏移: 0x58

复位值: 0x0000 0043

31																									16
Reserved																									
15		12	11	10	9	8	7	6	5	4	3	2	1	0											
	Reserved		VABT MEN	DPWMOD	JENDC AIEN	ENDC AIEN	BPCAL	PDRDY	RDY	CKMOD	CALALD	CALDIF		RES[1:0]											
			rw	rw	rw	rw	rw	r	r	rw	rw	rw		rw											

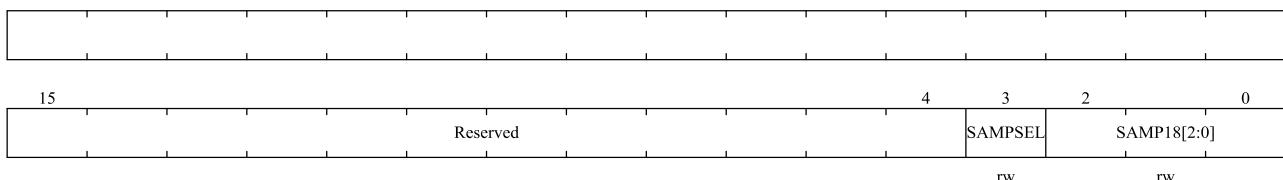
位域	名称	描述
31:12	Reserved	保留，必需保持复位值。
11	VBATMEN	Vbat监测使能 (Vbat monitor enable) 0: 禁止 1: 使能
10	DPWMOD	深度低功耗模式 (Deep Power Mode) 0: ADC 禁用时, ADC 进入低功耗模式 1: ADC禁用时, ADC进入深度低功耗模式
9	JENDCAIEN	任何注入通道中断使能 (Interrupt enable for any injected channels) 该位由软件设置和清除以启用/禁用注入通道转换结束中断 0: ADC_STS.JENDCA 禁用中断 1: ADC_STS.JENDCA 中断使能
8	ENDCAIEN	任何规则通道中断使能 (Interrupt enable for any regular channels) 该位由软件设置和清除以启用/禁用规则通道转换以结束中断 0: ADC_STS.ENDCA 禁用中断 1: ADC_STS.ENDCA 启用中断
7	BPCAL	旁路校准 (Bypass calibration) 0: 禁用 1: 使能
6	PDRDY	ADC掉电准备 (Power down ready) 0: 没有准备好 1: 准备好
5	RDY	ADC准备 (Ready) 0: 没有准备好 1: 准备好
4	CKMOD	时钟模式 (Clock Mode) 0: 同步时钟选择AHB 1: 异步时钟选择PLL
3	CALALD	校准自动载入 (calibration auto load) 0: 禁止 1: 使能
2	CALDIF	差分模式校准 (Differential mode for calibration) 该位由软件设置和清除, 以配置校准的单端或差分输入模式 0: 写入ENCAL位将在单端输入模式下启动校准 1: 写入ENCAL位将在差分输入模式下启动校准
1:0	RES[1:0]	数据分辨率 (Data resolution) 该位由软件设置和清除, ,以选择转换的分辨率

位域	名称	描述
		00: 6位 01: 8位 10: 10位 11: 12位

### 15.11.19 ADC 采样时间寄存器 3(ADC\_SAMPT3)

地址偏移: 0x5C

复位值: 0x0000 0000



位域	名称	描述
31:4	Reserved	保留, 必需保持复位值。
3	SAMPSEL	采样时间选择 (Sample time selection) SAMPSEL位为0时, 采样时间配置SAMPx[2:0]的取值如下 000: 1.5周期      100: 41.5周期 001: 7.5周期      101: 55.5周期 010: 13.5周期      110: 71.5周期 011: 28.5周期      111: 239.5周期 SAMPSEL位为1时, 采样时间配置SAMPx[2:0]的取值如下 000: 1.5周期      100: 19.5周期 001: 2.5周期      101: 61.5周期 010: 4.5周期      110: 181.5周期 011: 7.5周期      111: 601.5周期
2:0	SAMP18[2:0]	通道采样时间 (Channel Sample time) 通道采样时间定义和ADC_SAMPT2一致

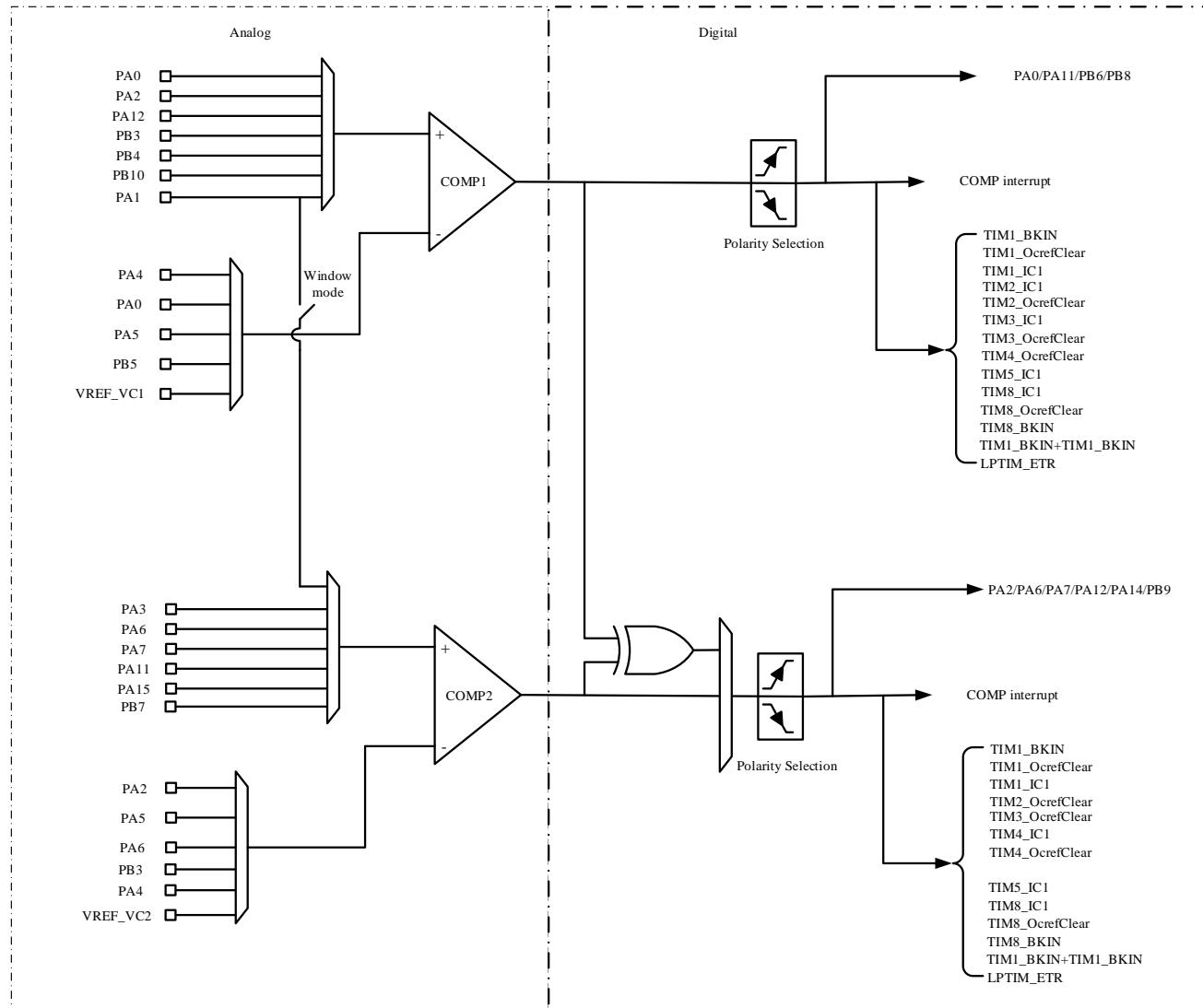
## 16 比较器 (COMP)

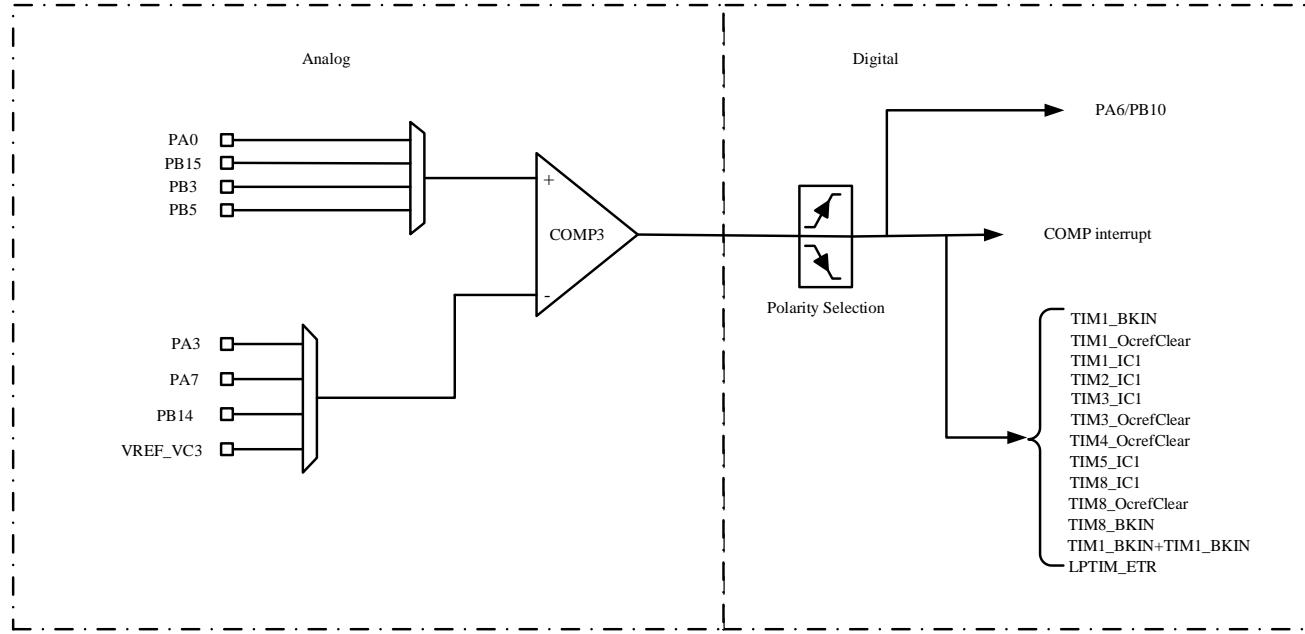
COMP 模块用于比较两个输入模拟电压的大小，并根据比较结果输出高/低电平。当“INP”输入端电压高于“INM”输入端电压时，比较器输出为高电平，当“INP”输入端电压低于“INM”输入端电压时，比较器输出为低电平。

### 16.1 COMP 系统连接框图

COMP 模块最多支持 3 个独立比较器，统一挂接在 APB1 总线上。

图 16-1 比较器系统连接图





## 16.2 COMP 特性

- 最多 3 个独立的比较器
- 内置三个 64 级可编程的比较电压参考源 VREF\_VC1, VREF\_VC2, VREF\_VC3
- 支持滤波时钟, 滤波复位
- 输出极性可配置高、低
- 迟滞配置可配置无、低、中、高
- 比较结果可输出到 I/O 端口或触发定时器, 用于捕获事件、OCREF\_CLR 事件、刹车事件、产生中断
- 输入通道可复选 I/O 端口、DAC 的通道输出、VREF\_VC1、VREF\_VC2 和 VREF\_VC3
- 可配只读或读写, 在锁定的情况下需要复位才能解锁
- 支持消隐 (Blanking), 可配置产生 Blanking 的消隐源
- COMP1/COMP2 可以组成窗口比较器
- 可通过产生中断的方式将系统从 Sleep、Stop0 模式唤醒
- 可配置滤波窗口大小
- 可配置滤波阈值大小
- 可配置用于滤波的采样频率

## 16.3 COMP 配置流程

完整的配置项包括如下所示, 某些项目如果采用系统默认配置, 跳过相应的配置项。

1. 可配置的迟滞等级 COMPx\_CTRL.HYST[1:0]。
2. 配置输出极性 COMPx\_CTRL.POL。
3. 配置输入选择，比较器正极 COMPx\_CTRL.INPSEL[3:0]，负极 COMPx\_CTRL.INMSEL [2:0]。
4. 配置输出选择 COMPx\_CTRL.OUTSEL[3:0]。
5. 配置消隐源 COMPx\_CTRL.BLKING[2:0]。
6. 配置比较器窗口模式 COMP\_WINMODE.CMP12MD。
7. 配置滤波器采样窗口 COMPx\_FILC.SAMPW[4:0]。
8. 配置阈值 COMPx\_FILC.THRESH[4:0]（阈值应当大于 COMPx\_FILC.SAMPW[4:0]/2）。
9. 配置滤波器采样频率（对于计时器应用，采样频率应当大于 5MHz）。
10. 打开滤波器使能 COMPx\_FILC.FILEN。
11. 打开比较器使能 COMPx\_CTRL.EN。

**注:**对于以上步骤，需先打开滤波器使能，再打开比较器使能，比较器使能需要在滤波（若启用）配置、使能完成后启用，此外在比较器控制寄存器锁定 LOCK 的情况下，只有通过复位才能取消锁定。

## 16.4 COMP 工作模式

### 16.4.1 窗口模式

- 比较器 1 和比较器 2 共享 PA1 形成窗口比较器。使用窗口模式可以测量连接到 PA1 端口的外部电压是否在比较器 1 和比较器 2 的负端电压范围之间。

### 16.4.2 独立比较器

3 个比较器可独立配置，完成比较器功能。比较器的输出可以输出到 IO 端口，每一个比较器都有不同的重映射端口，通过配置 COMPx\_CTRL.OUTSEL[3:0]可以选择比较器的输出，连接到相应的端口。

比较器输出，支持触发事件，比如可以配置成定时器 1，定时器 8 的刹车功能。

**注：**具体配置参考比较器互联关系

## 16.5 比较器互联关系

比较器输出端口的互联，可以参考 AFIO 重映射章节，描述了比较器 OUT 重映射的值，具体配置如下表所示。

INPSEL	COMP1	COMP2	COMP3
000	PA0	PA1	PA0
001	PA2	PA3	PB1
010	PA12	PA6	PB11
011	PB3	PA7	PB15
100	PB4	PA11	PB3

101	PB10	PA15	PB5
110	PA1	PB7	FLOATING
111	FLOATING	FLOATING	FLOATING

比较器 INP 引脚有如下配置

INMSEL	COMP1	COMP2	COMP3
000	VREF_VC1	VERF_VC2	VREF_VC3
001	PA4	PA2	PA3
010	PA0	PA5	PA7
011	PA5	PA6	PB2
100	PB5	PB3	PB14
101	FLOATING	PA4	FLOATING
110	FLOATING	FLOATING	FLOATING
111	FLOATING	FLOATING	FLOATING

比较器输出 TRIG 的信号有如下互联关系

TRIG	COMP1	COMP2	COMP3
0000	NC	NC	NC
0001	TIM1_BKIN	TIM1_BKIN	TIM1_BKIN
0010	TIM1_OCrefclear	TIM1_OCrefclear	TIM1_OCrefclear
0011	TIM1_IC1	TIM1_IC1	TIM1_IC1
0100	TIM2_IC1	TIM2_OCrefclear	TIM2_IC1
0101	TIM2_OCrefclear	TIM3_OCrefclear	NC
0110	TIM3_IC1	TIM4_IC1	TIM3_IC1
0111	TIM3_OCrefclear	TIM4_OCrefclear	TIM3_OCrefclear
1000	TIM4_OCrefclear	TIM5_IC1	TIM4_OCrefclear
1001	TIM5_IC1	TIM8_IC1	TIM5_IC1
1010	TIM8_IC1	TIM8_OCrefclear	NC
1011	TIM8_OCrefclear	NC	TIM8_IC1
1100	NC	NC	TIM8_OCrefclear
1101	TIM8_BKIN	TIM8_BKIN	TIM8_BKIN
1110	TIM1_BKIN+TIM8_BKIN	TIM1_BKIN + TIM8_BKIN	TIM1_BKIN + TIM8_BKIN
1111	LPTIM_ETR	LPTIM_ETR	LPTIM_ETR

## 16.6 中断

COMP 支持中断响应，COMP1, COMP2, COMP3 共享 1 个中断入口，中断产生有如下 2 种情况。

- COMPx\_CTRL.POL 极性不反转，中断使能，当 INPSEL > INMSEL 时，COMPx\_CTRL.OUT 由硬件置为 1 时即产生比较器中断。
- COMPx\_CTRL.POL 极性反转，中断使能，当 INPSEL < INMSEL 时，COMPx\_CTRL.OUT 由硬件置为 1 时即产生比较器中断。

注意：使用 COMP 中断时，需要配置对应的 EXTI 中断，可参考 6.2.4 章节

## 16.7 COMP 寄存器

### 16.7.1 COMP 寄存器总览

表 16-1 COMP 寄存器总览

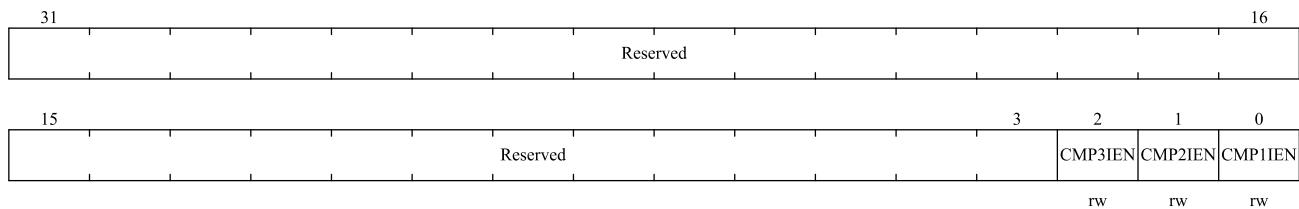
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
000h	COMP_INTEN	Reserved																									CMP3HEN 0	CMP2HEN 0	CMP1HEN 0																				
	Reset Value																																																
008h	COMP_WINMODE	Reserved																									CMP3LK 0	CMP2LK 0	CMP1LK 0																				
	Reset Value																																																
00Ch	COMP_LOCK	Reserved																									CMP3LN 0	CMP2LN 0	CMP1LN 0																				
	Reset Value																																																
010h	COMP1_CTRL	Reserved	FILTVAL 0	OUT	BLKING[2:0]				HYST[1:0] 0	OUTSEL[3:0]				POL 0	INPSEL[2:0]				INMSEL[2:0] 0	EN 0	FILEN 0	CMP3LK 0	CMP2LK 0	CMP1LK 0																									
	Reset Value																																																
014h	COMP1_FILC	Reserved												SAMPW [4:0]				THRESH[4:0]				FILEN 0	SAMPW 0	THRESH 0																									
	Reset Value																																																
018h	COMP1_FILP	Reserved						CLKPSC[15:0]																	FILEN 0	CLKPSC 0	INMSEL 0																						
	Reset Value																																																
01Ch		Reserved																																															
020h	COMP2_CTRL	Reserved	FILTVAL 0	OUT	BLKING[2:0]				HYST[1:0] 0	OUTSEL[3:0]				POL 0	INPSEL[2:0]				INMSEL[2:0] 0	EN 0	FILEN 0	CMP3LN 0	CMP2LN 0	CMP1LN 0																									
	Reset Value																																																
024h	COMP2_FILC	Reserved												SAMPW [4:0]				THRESH[4:0]				FILEN 0	SAMPW 0	THRESH 0																									
	Reset Value																																																
028h	COMP2_FILP	Reserved						CLKPSC[15:0]																	FILEN 0	CLKPSC 0	INMSEL 0																						
	Reset Value																																																
02Ch		Reserved																																															
030h	COMP3_CTRL	Reserved	FILTVAL 0	OUT	BLKING[2:0]				HYST[1:0] 0	OUTSEL[3:0]				POL 0	INPSEL[2:0]				INMSEL[2:0] 0	EN 0	FILEN 0	CMP2XO 0	SAMPW 0	THRESH 0																									
	Reset Value																																																
034h	COMP3_FILC	Reserved												SAMPW [4:0]				THRESH[4:0]				FILEN 0	SAMPW 0	THRESH 0																									
	Reset Value																																																
038h	COMP3_FILP	Reserved						CLKPSC[15:0]																	FILEN 0	CLKPSC 0	INMSEL 0																						
	Reset Value																																																

03Ch	Reserved															
040h	COMP_INVREF	Reserved	VV3TRM				VV3EN	VV2TRM				VV2EN	VV1TRM			VV1EN
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0
044h	COMP_TEST		Reserved												IBEN	VEN
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	0	0
040h	COMP_INVREF		Reserved												CMP3IS	CMP2IS
	Reset Value		0	0	0	0	0	0	0	0	0	0	0	0	CMP1IS	0

## 16.7.2 COMP 中断使能寄存器 (COMP\_INTEN)

偏移地址: 0x00

复位值: 0x0000 0000



位域	名称	描述
31:3	Reserved	保留，必需保持复位值。
2	CMP3IEN	该位控制COMP3的中断启用 0: 禁用 1: 使用
1	CMP2IEN	该位控制COMP2的中断启用 0: 禁用 1: 使用
0	CMP1IEN	该位控制COMP1的中断启用 0: 禁用 1: 使用

## 16.7.3 COMP 窗口比较寄存器 (COMP\_WINMODE)

偏移地址: 0x08

复位值: 0x0000 0000

31	Reserved	16
15	Reserved	1 0
rw		

位域	名称	描述
31:1	Reserved	保留，必需保持复位值。
0	CMP12MD	该位选择窗口模式：比较器的两个正端输入共享PA1输入 1: 比较器1和2不在窗口模式 1: 比较器1和2用于窗口模式

#### 16.7.4 COMP 锁寄存器(COMP\_LOCK)

偏移地址: 0x0C

复位值: 0x0000 0000

31	Reserved	16
15	Reserved	3 2 1 0
rw rw rw		

位域	名称	描述
31:3	Reserved	保留，必需保持复位值。
2	CMP3LK	仅写一次，是由软件控制的，只能通过系统重置来清除 设置此位可以将COMP3_CTRL设置为只读 0: COMP3_CTRL是可读写的 1: COMP3_CTRL是只读的
2	CMP2LK	仅写一次，是由软件控制的，只能通过系统重置来清除 设置此位可以将COMP2_CTRL设置为只读 0: COMP2_CTRL是可读写的 1: COMP2_CTRL是只读的
2	CMP1LK	仅写一次，是由软件控制的，只能通过系统重置来清除 设置此位可以将COMP1_CTRL设置为只读 0: COMP1_CTRL是可读写的 1: COMP1_CTRL是只读的

### 16.7.5 COMP1 控制寄存器 (COMP1\_CTRL)

偏移地址: 0x10

复位值: 0x0000 0000

31	Reserved								22	21	20	19	18	16
									FILTVAL	Reserved	OUT	BLKING[2:0]		
15	14	13	12	9	8	7		r 5	r 4	r 3	rw 1			0
HYST[1:0]	POL		OUTSEL[3:0]	Reserved		INPSEL[2:0]	Reserved		INMSEL[2:0]	EN				
rw	rw		rw			rw		rw	rw	rw				rw

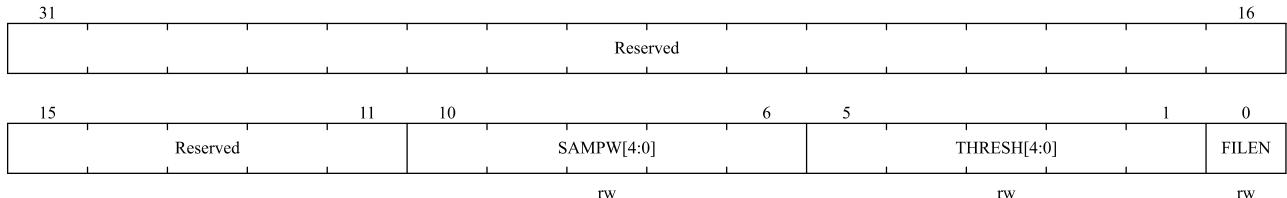
位域	名称	描述
31:22	Reserved	保留，必需保持复位值。
21	FILTVAL	该只读位反应比较器1数字滤波后的输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
20	Reserved	保留，必需保持复位值。
19	OUT	这个只读位是比较器1输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
18: 16	BLKING[2:0]	这些位选择哪个定时器输出控制比较器1输出消隐。. 000: 不消隐 001: 选择Tim1 OC5作为消隐源 010: 选择Tim8 OC5作为消隐源 其他配置:保留
15:14	HYST[1:0]	这些位选择比较器1的迟滞等级。 00: 保留; 01: 低迟滞; 10: 中等迟滞; 11: 高迟滞。
13	POL	该位用于反转比较器1的输出 0: 输出未反转 1: 输出反转
12:9	OUTSEL[3:0]	这些位选择必须与比较器1输出连接的定时器输入 0000: 保留 0001: TIM1_BKIN 0010: TIM1_OCrefclear 0011: TIM1_IC1 0100: TIM2_IC1 0101: TIM2_OCrefclear 0110: TIM3_IC1 0111: TIM3_OCrefclear 1000: TIM4_OCrefclear

位域	名称	描述
		1001: TIM5_IC1 1010: TIM8_IC1 1011: TIM8_OCrefclear 1100: Reserved 1101: TIM8_BKIN 1110: TIM1_BKIN+TIM8_BKIN 1111: LPTIM_ETR  注意: 根据产品的不同, 当定时器不可用时, 会保留相应的组合。
8	Reserved	保留, 必需保持复位值。
7:5	INPSEL[2:0]	比较器1的正端输入选择。 000: PA0 001: PA2 010: PA12 011: PB3 100: PB4 101: PB10 110: PA1 111: 比较器1的输入应该浮空
4	Reserved	保留, 必需保持复位值。
3:1	INMSEL[2:0]	比较器1的负端输入选择。 000: VREF_VC1 001: PA4 010: PA0 011: PA5 100: PB5 101: 浮空 110: 浮空 111: 浮空
0	EN	该位打开/关闭COMP1  0: 比较器已禁用 1: 比较器已启用

### 16.7.6 COMP1 滤波控制寄存器 (COMP1\_FILC)

偏移地址: 0x14

复位值: 0x0000 0000

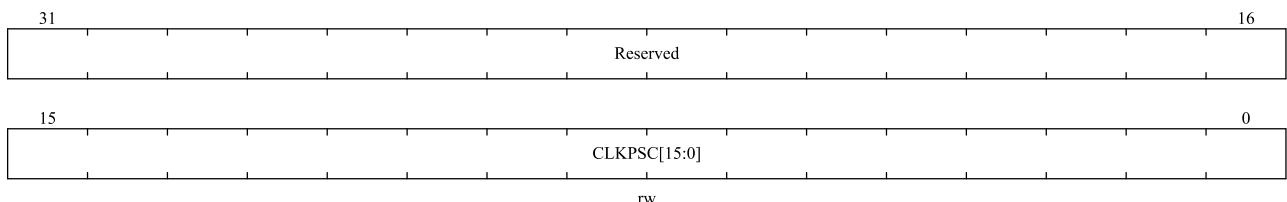


位域	名称	描述
31:11	Reserved	保留，必需保持复位值。
10:6	SAMPW[4:0]	低通滤波器采样窗口大小，采样窗口 = SAMPW + 1。
5:1	THRESH[4:0]	低通滤波器门限置，样本窗口中至少出现相反状态的采样阈值，才能改变输出状态，此值要求大于 SAMPW / 2。
0	FILEN	滤波器使能位 0: 关闭 1: 使能

### 16.7.7 COMP1 滤波器分频寄存器 (COMP1\_FILP)

偏移地址: 0x18

复位值: 0x0000 0000



位域	名称	描述
31:16	Reserved	保留，必需保持复位值。
15:0	CLKPSC[15:0]	低通滤波采样时钟预分频，系统时钟分频数 = CLKPSC + 1。 0: 每1个时钟 1: 每2个时钟 2: 每3个时钟 ...

### 16.7.8 COMP2 控制寄存器 (COMP2\_CTRL)

偏移地址: 0x20

复位值: 0x0000 0000

31	Reserved							22	21	20	19	18	16
								FILTVAL	PWRMODE	OUT	BLKING[2:0]		
15	14	13	12	9	8	7	r 5	rw 4	r 3	rw 1	rw 0		
HYST[1:0]	POL		OUTSEL[3:0]		Reserved	INPSEL[2:0]	Reserved		INMSEL[2:0]	EN			
rw	rw		rw			rw			rw		rw		

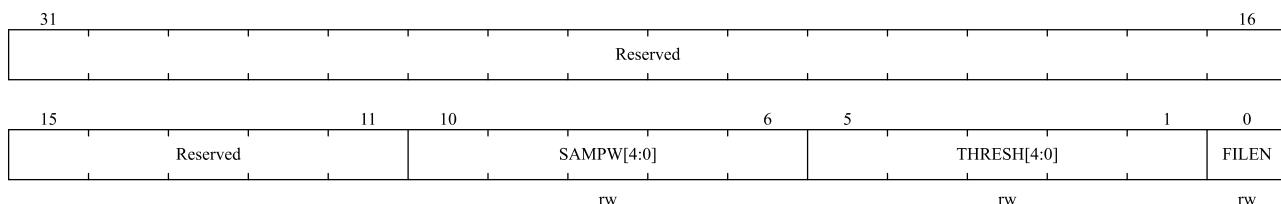
位域	名称	描述
31:22	Reserved	保留，必需保持复位值。
21	FILTVAL	该只读位反应比较器2数字滤波后的输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
20	Reserved	保留，必需保持复位值。
19	OUT	这个只读位是比较器2输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
18: 16	BLKING[2:0]	这些位选择哪个定时器输出控制比较器2输出消隐。. 000: 不消隐 001: 选择Tim1 OC5作为消隐源 010: 选择Tim8 OC5作为消隐源 其他配置:保留
15:14	HYST[1:0]	这些位选择比较器2的迟滞等级。 00: 保留; 01: 低迟滞; 10: 中等迟滞; 11: 高迟滞。
13	POL	该位用于反转比较器2的输出 0: 输出未反转 1: 输出反转
12:9	OUTSEL[3:0]	这些位选择必须与比较器2输出连接的定时器输入 0000: 保留 0001: TIM1_BKIN 0010: TIM1_OCrefclear 0011: TIM1_IC1 0100: TIM2_OCrefclear 0101: TIM3_OCrefclear 0110: TIM4_IC1 0111: TIM4_OCrefclear 1000: TIM5_IC1 1001: TIM8_IC1 1010: TIM8_OCrefclear 1011: 保留 1100: 保留 1101: TIM8_BKIN

位域	名称	描述
		1110: TIM1_BKIN + TIM8_BKIN 1111: LPTIM_ETR 注意: 根据产品的不同, 当定时器不可用时, 会保留相应的组合。
8	Reserved	保留, 必需保持复位值。
7:5	INPSEL[2:0]	比较器2的正端输入选择。 000: 窗口模式连接 (PA1) 001: PA3 010: PA6 011: PA7 100: PA11 101: PA15 110: PB7 111: 浮空
4	Reserved	保留, 必需保持复位值。
3:1	INMSEL[2:0]	比较器2的负端输入选择。 000: VREF_VC2 001: PA2 010: PA5 011: PA6 100: PB3 101: PA4 110: 浮空 111: 浮空
0	EN	该位打开/关闭COMP2 0: 比较器已禁用 1: 比较器已启用

### 16.7.9 COMP2 滤波控制寄存器 (COMP2\_FILC)

偏移地址: 0x24

复位值: 0x0000 0000



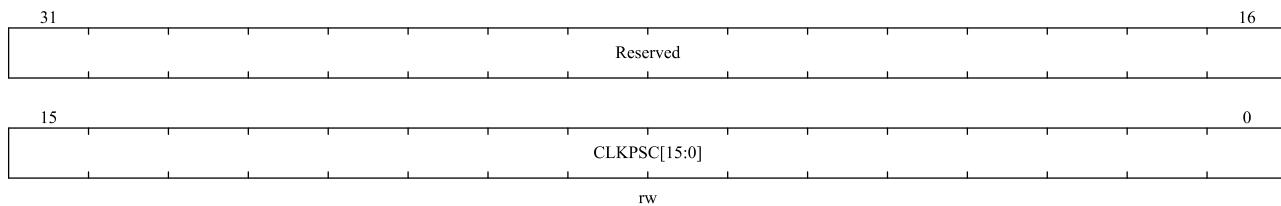
位域	名称	描述
31:11	Reserved	保留, 必需保持复位值。

位域	名称	描述
10:6	SAMPW[4:0]	低通滤波器采样窗口大小，采样窗口 = SAMPW + 1。
5:1	THRESH[4:0]	低通滤波器门限置，样本窗口中至少出现相反状态的采样阈值，才能改变输出状态，此值要求大于 SAMPW / 2。
0	FILEN	滤波器使能位 0: 关闭 1: 使能

### 16.7.10 COMP2 滤波器分频寄存器 (COMP2\_FILP)

偏移地址: 0x28

复位值: 0x0000 0000

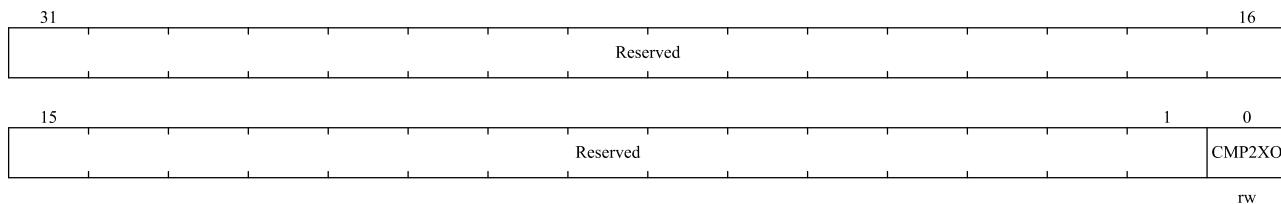


位域	名称	描述
31:16	Reserved	保留，必需保持复位值。
15:0	CLKPSC[15:0]	低通滤波采样时钟预分频，系统时钟分频数 = CLKPSC + 1。 0: 每1个时钟 1: 每2个时钟 2: 每3个时钟 ...

### 16.7.11 COMP2 输出选择寄存器 (COMP2\_OSEL)

偏移地址: 0x2C

复位值: 0x0000 0000



位域	名称	描述
31:1	Reserved	保留，必需保持复位值。

位域	名称	描述
0	CMP2XO	比较器2比较结果和比较器1输出异或后输出 0: COMP2 输出 1: COMP1和COMP2结果异或输出

### 16.7.12 COMP3 控制寄存器 (COMP3\_CTRL)

偏移地址: 0x30

复位值: 0x0000 0000

31	Reserved								22	21	20	19	18	16
									FILTVAL	Reserved	OUT	BLKING[2:0]		
15	14	13	12	9	8	7		r 5	4	r 3		rw 1	rw 0	
HYST[1:0]	POL	OUTSEL[3:0]			Reserved	INPSEL[2:0]		Reserved	INMSEL[2:0]		EN			
rw	rw	rw				rw			rw			rw	rw	

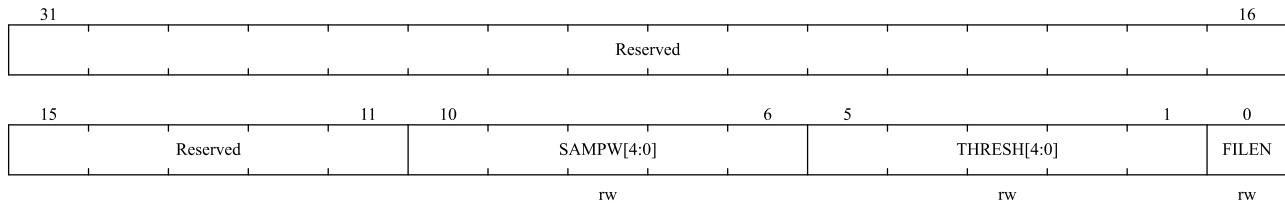
位域	名称	描述
31:22	Reserved	保留, 必需保持复位值。
21	FILTVAL	该只读位反应比较器3数字滤波后的输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
20	Reserved	保留, 必需保持复位值。
19	OUT	这个只读位是比较器3输出状态。 0: 输出低(正端输入低于负端输入). 1: 输出高(正端输入高于负端输入).
18: 16	BLKING[2:0]	这些位选择哪个定时器输出控制比较器3输出消隐。. 000: 不消隐 001: 选择Tim1 OC5作为消隐源 010: 选择Tim8 OC5作为消隐源 其他配置:保留
15:14	HYST[1:0]	这些位选择比较器3的迟滞等级。 00: 保留; 01: 低迟滞; 10: 中等迟滞; 11: 高迟滞。
13	POL	该位用于反转比较器3的输出 0: 输出未反转 1: 输出反转
12:9	OUTSEL[3:0]	这些位选择必须与比较器3输出连接的定时器输入 0000: 保留. 0001: TIM1_BKIN 0010: TIM1_OCrefclear

位域	名称	描述
		0011: TIM1_IC1 0100: TIM2_IC1 0101: Reserved 0110: TIM3_IC1 0111: TIM3_OCrefclear 1000: TIM4_OCrefclear 1001: TIM5_IC1 1010: Reserved. 1011: TIM8_IC1 1100: TIM8_OCrefclear 1101: TIM8_BKIN 1110: TIM1_BKIN + TIM8_BKIN 1111: LPTIM_ETR  注意：根据产品的不同，当定时器不可用时，会保留相应的组合。
8	Reserved	保留，必需保持复位值。
7:5	INPSEL[2:0]	比较器3的正端输入选择。 000: PA0 001: PB1 010: PB11 011: PB15 100: PB3 101: PB5 110: 浮空 111: 浮空
4	Reserved	保留，必需保持复位值。
3:1	INMSEL[2:0]	比较器3的负端输入选择。 000: VREF_VC3 001: PA3 010: PA7 011: PB2 100: PB14 101: 浮空 110: 浮空 111: 浮空
0	EN	该位打开/关闭COMP3  0: 比较器已禁用 1: 比较器已启用

### 16.7.13 COMP3 滤波控制寄存器 (COMP3\_FILC)

偏移地址: 0x34

复位值: 0x0000 0000

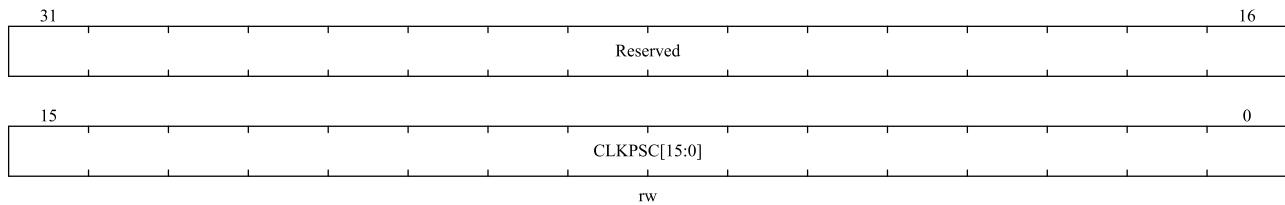


位域	名称	描述
31:11	Reserved	保留, 必需保持复位值。
10:6	SAMPW[4:0]	低通滤波器采样窗口大小, 采样窗口 = SAMPW + 1。
5:1	THRESH[4:0]	低通滤波器门限置, 样本窗口中至少出现相反状态的采样阈值, 才能改变输出状态, 此值要求大于 SAMPW / 2。
0	FILEN	滤波器使能位 0: 关闭 1: 使能

### 16.7.14 COMP3 滤波器分频寄存器 (COMP3\_FILP)

偏移地址: 0x38

复位值: 0x0000 0000

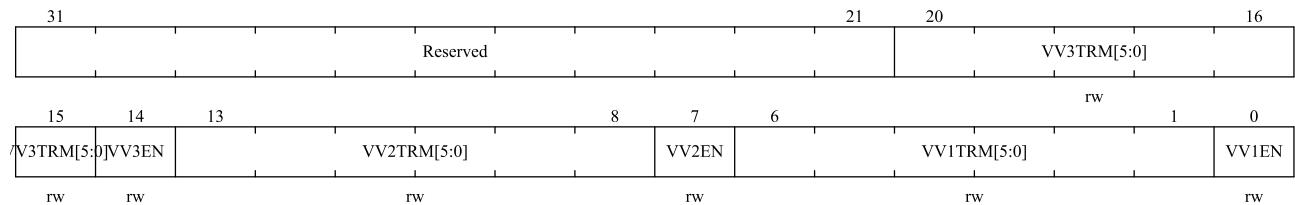


位域	名称	描述
31:16	Reserved	保留, 必需保持复位值。
15:0	CLKPSC[15:0]	低通滤波采样时钟预分频, 系统时钟分频数 = CLKPSC + 1。 0: 每1个时钟 1: 每2个时钟 2: 每3个时钟 ...

### 16.7.15 COMP 参考电压寄存器(COMP\_VREFSCL)

偏移地址: 0x40

复位值: 0x0000 0000

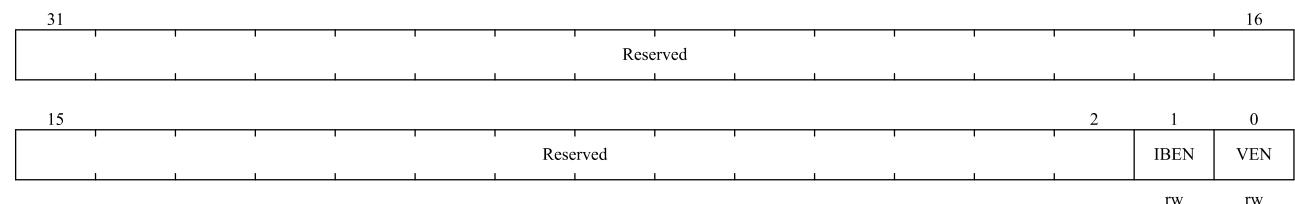


位域	名称	描述
31:21	Reserved	保留, 必需保持复位值。
20:15	VV3TRM [5:0]	DAC3 输出电压选择
14	VV3EN	DAC3 电压标定器: 0: 禁用 1: 启用
13:8	VV2TRM [5:0]	DAC2输出电压选择
7	VV2EN	DAC2电压标定器: 0: 禁用 1: 启用
6:1	VV1TRM [5:0]	DAC1输出电压选择
0	VV1EN	DAC1电压标定器: 0: 禁用 1: 启用

### 16.7.16 COMP 测试寄存器 (COMP\_TEST)

偏移地址: 0x44

复位值: 0x0000 0000



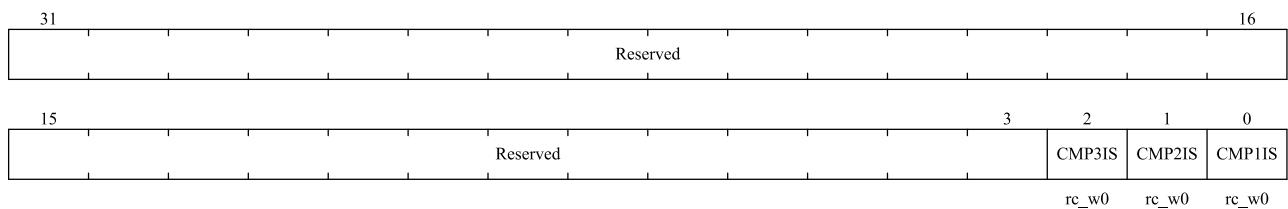
位域	名称	描述
31:2	Reserved	保留, 必需保持复位值。
1	IBEN	比较器偏置电流测试软件启用位, 一旦启用, 偏置电流可以在comp_ib_test上测量到 0: 禁用

位域	名称	描述
		1: 启用 0: 禁用
0	VEN	比较器VREF测试软件使能: 0: 禁用 1: 启用

### 16.7.17 COMP 中断状态寄存器 (COMP\_INTSTS)

偏移地址: 0x48

复位值: 0x0000 0000



位域	名称	描述
31:3	Reserved	保留, 必需保持复位值。
2	CMP3IS	COMP3中断状态位, 写0清除
1	CMP2IS	COMP2中断状态位, 写0清除
0	CMP1IS	COMP1中断状态位, 写0清除

## 17 I<sup>2</sup>C 接口

### 17.1 简介

I<sup>2</sup>C(inter-integrated circuit)总线是一种广泛应用的总线结构，它只有两根双向线，即数据总线 SDA 和时钟总线 SCL，通过这两根线，所有与 I<sup>2</sup>C 总线兼容的设备都可以通过 I<sup>2</sup>C 总线彼此直接通信。

I<sup>2</sup>C 接口连接微控制器和串行 I<sup>2</sup>C 总线，可用于 MCU 和外部 I<sup>2</sup>C 设备的通讯。I<sup>2</sup>C 接口模块实现了 I<sup>2</sup>C 协议的标速模式和快速模式，具备 CRC 计算和校验功能、支持 SMBus(系统管理总线)和 PMBus(电源管理总线)，此外它提供多主机功能，控制所有 I<sup>2</sup>C 总线特定的时序、协议、仲裁和定时。I<sup>2</sup>C 接口模块也支持 DMA 模式，可有效减轻 CPU 的负担。

### 17.2 特性

- 同一接口既可实现主机功能又可实现从机功能
- 是并行总线到 I<sup>2</sup>C 总线协议的转换器
- 支持 7 位和 10 位的地址模式和广播寻址
- 作为 I<sup>2</sup>C 主设备可以产生时钟、起始信号和停止信号
- 作为 I<sup>2</sup>C 从设备具有可编程的 I<sup>2</sup>C 地址检测、停止位检测的功能
- 支持标速(最高 100 kHz)和快速(400 kHz、1MHz)模式
- 支持中断向量，字节成功传输中断和错误事件中断
- 可选的时钟延展功能
- 支持 DMA 模式
- 可选择的 PEC (报文错误检测) 生成和校验
- 可编程的模拟和数字滤波
- 兼容 SMBus 2.0 和 PMBus

注：不是所有产品中都包含上述所有特性，请参考相关的数据手册，确认该产品支持的 I<sup>2</sup>C 功能。

### 17.3 功能描述

I<sup>2</sup>C 接口通过数据引脚 (SDA) 和时钟引脚 (SCL) 连接到 I<sup>2</sup>C 总线与外部设备进行通信，可以连接到标准 (高达 100kHz) 或快速 (400kHz、1MHz) 的 I<sup>2</sup>C 总线。I<sup>2</sup>C 模块接收时将数据从串行转换成并行，发送时将数据从并行转换成串行。支持中断模式，用户可以根据需要开启或禁止中断。

#### 17.3.1 SDA/SCL 控制

I<sup>2</sup>C 模块有两条接口线：串行数据线 (SDA) 和串行时钟线 (SCL)。连接到总线上的设备通过这两根线互相传递信息。SDA 和 SCL 都是双向线，通过一个电流源或者上拉电阻接到电源正极。当总线空闲时，两条线

都是高电平。连接到总线的设备输出极必须带开漏或者开集，以提供线与功能。I<sup>2</sup>C 总线上的数据在标准模式下可以达到 100 kbit/s，在快速模式下可以达到 1000kbit/s。由于 I<sup>2</sup>C 总线上可能会连接不同工艺的设备，逻辑‘0’和逻辑‘1’的电平并不是固定的，取决于 VDD 的实际电平。

如果允许时钟延长，即 SCL 线拉低，就可以避免在接收时发生过载错误以及在发送时发生欠载错误。

比如，在发送模式时，如果发送数据寄存器为空且字节发送结束位置起（I2C\_STS1.TXDATE = 1, I2C\_STS1.BSF=1），I<sup>2</sup>C 接口在传输前保持时钟线为低，以等待软件读取 STS1 后把数据写进数据寄存器（缓冲器和移位寄存器都是空的）；在接收模式时，如果数据寄存器非空且字节发送结束位置起（I2C\_STS1.RXDATNE = 1, I2C\_STS1.BSF = 1），I<sup>2</sup>C 接口在接收到数据字节后保持时钟线为低，以等待软件读 STS1，然后读数据寄存器 DAT（缓冲器和移位寄存器都是满的）。

如果从模式中禁止时钟延长，在接收模式时，如果接收数据寄存器非空（I2C\_STS1.RXDATNE = 1），在接收到下个字节前数据还没有被读出，则发生过载错误，最后一字节也将被丢弃。在发送模式时，如果发送数据寄存器空（I2C\_STS1.TXDATE = 1），在必须发送下个字节之前还没有新数据写进数据寄存器，则发生欠载错误。相同的字节将被重复发出。这种情况下不控制重复写冲突。

### 17.3.2 软件通信流程

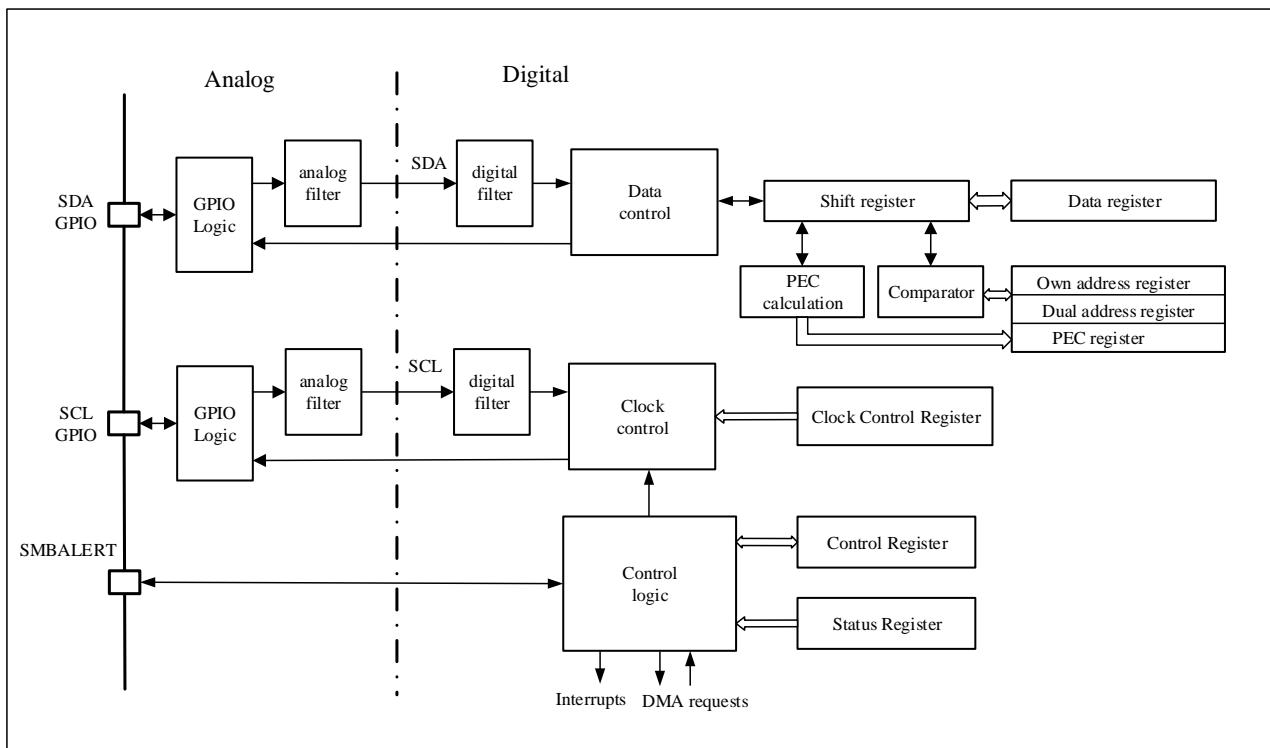
I<sup>2</sup>C 设备的数据传输分为主机和从机。主机是指负责初始化总线上数据的传输并产生时钟信号的设备，此时任何被寻址的设备都是从机。不管 I<sup>2</sup>C 设备是主机还是从机，都可以发送或接收数据。I<sup>2</sup>C 接口支持 4 种运行模式：

- 从机发送器模式
- 从机接收器模式
- 主机发送器模式
- 主机接收器模式

系统复位后，I<sup>2</sup>C 默认工作在从机模式下。通过软件配置 I<sup>2</sup>C 接口在总线上发送一个起始位，随后接口自动地从从模式切换到主模式；当仲裁丢失或产生停止信号时，则从主模式切换到从模式。

I<sup>2</sup>C 接口的功能框图如下：

图 17-1 I<sup>2</sup>C 功能框图

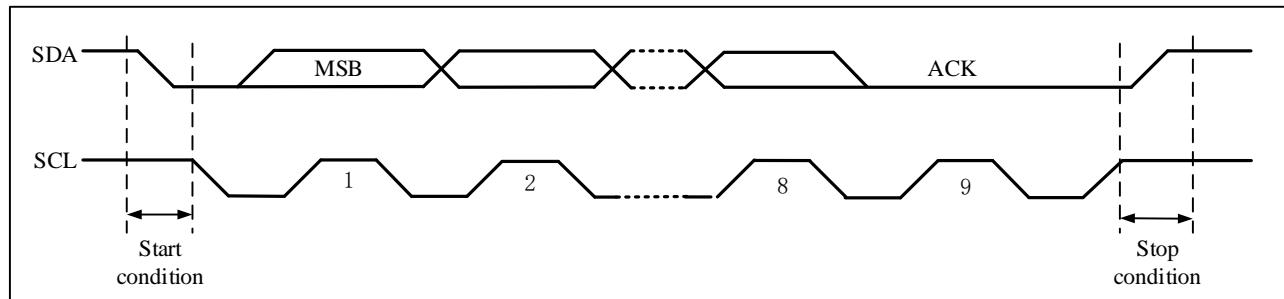


注：在 SMBus 模式下，SMBALERT 是可选信号。如果禁止了 SMBus，则不能使用该信号。

### 17.3.2.1 开始和停止条件

所有的数据传输总是以起始位开始并以停止位结束。起始条件和停止条件都是在主模式下由软件控制产生。起始位（START）是指：在 SCL 为高时，SDA 线上出现一个从高到低的电平转换。停止位（STOP）是指在 SCL 为高时，SDA 线上出现一个从低到高的电平转换。如下图所示：

图 17-2 I<sup>2</sup>C 总线协议



### 17.3.2.2 时钟同步与仲裁

I<sup>2</sup>C 接口支持多主机仲裁，即两个主机可以同时在空闲总线上开始传送数据，因此就必须通过一些机制来决定哪个主机获取总线的控制权，这一般是通过时钟同步和仲裁来完成的。

I<sup>2</sup>C 电路具有两个关键特点：

- SDA 和 SCL 为漏极开路结构，通过外部的上拉电阻实现了信号的“线与”逻辑；
- SDA 和 SCL 引脚在输出信号的同时还将引脚上的电平进行检测，检测是否与刚才输出一致。这为“时

钟同步”和“总线仲裁”提供硬件基础。

I<sup>2</sup>C 设备对总线的操作是通过“把线路接地”来输出逻辑 0。基于 I<sup>2</sup>C 总线的特点，如果一设备发送逻辑 0，其他发送逻辑 1，那么线路看到的只有逻辑 0，所以线路上不可能出现电平冲突的现象。

总线的物理接法允许主设备往总线写数据的同时读取数据。这样两主设备争总线的时候发送逻辑 0 的并不知道竞争的发生，只有发送逻辑 1 的才会发现冲突（当写一个逻辑 1，却读到了 0）从而退出竞争。

## 时钟同步

SCL 线的高到低切换会使器件开始数它们的低电平周期，而且一旦器件的时钟变低电平，它会使 SCL 线保持这种状态直到到达时钟的高电平。但是，如果另一个时钟仍处于低电平周期，这个时钟的低到高切换不会改变 SCL 线的状态，因此，SCL 线被有最长低电平周期的器件保持低电平，此时，低电平周期短的器件会进入高电平的等待状态。

当所有有关的器件数完了它们的低电平周期后，时钟线被释放并且变成高电平，之后器件时钟和 SCL 线的状态没有差别，而且所有器件会开始数它们的高电平周期，首先完成高电平周期的器件会再次将 SCL 线拉低。

这样，产生的同步 SCL 时钟的低电平周期由低电平时钟周期最长的器件决定，而高电平周期由高电平时钟周期最短的器件决定。

## 仲裁

仲裁和同步一样，都是为了解决多主机情况下的总线控制冲突。仲裁的过程与从机无关。当两个主机在总线空闲的时候都产生了一个有效的起始位，这种情况就需要决定由哪个主机来完成数据传输，这就是仲裁的过程。

各个主控制器没有对总线实施控制的优先级别，都是由仲裁决定的。总线控制随即而定且逐位进行，他们遵循“低电平优先”的原则，即谁先发送低电平谁就会掌握对总线的控制权。在每一位的仲裁期间，当 SCL 为高时，每个主机都检查自己的 SDA 电平是否和自己发送的相同。理论上讲，如果两个主机所传输的内容完全相同，那么他们能够成功传输而不出现错误。如果一个主机发送高电平但检测到 SDA 电平为低，则认为自己仲裁失败并关闭自己的 SDA 输出驱动，而另一个主机则继续完成自己的传输。

### 17.3.2.3 I<sup>2</sup>C 数据通信流程

每个 I<sup>2</sup>C 设备都通过唯一的地址进行识别，根据设备功能，他们既可以是发送器也可作为接收器。

I<sup>2</sup>C 主机负责产生起始位和结束位来开始和结束一次传输，并且负责产生 SCL 时钟。

I<sup>2</sup>C 模块支持 7 位和 10 位的地址，用户可以通过软件配置 I<sup>2</sup>C 从机的地址。I<sup>2</sup>C 从机检测到 I<sup>2</sup>C 总线上的起始位之后，就开始从总线上接收地址，并把接收到的地址和自身的地址进行比较，一旦两个地址相同，I<sup>2</sup>C 从机将发送一个确认应答(ACK)，并响应总线的后续命令：发送或接受所要求的数据。此外，如果软件开启了广播呼叫，则 I<sup>2</sup>C 从机始终对一个广播地址(0x00)发送确认应答。

数据和地址按 8 位字节进行传输，高位在前。跟在起始条件后的 1 或 2 个字节是地址（7 位模式为 1 个字节，10 位模式为 2 个字节）。地址只在主模式发送。在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位（ACK）给发送器。如图 17-2 I<sup>2</sup>C 总线协议所示。

软件可以开启或禁能应答（ACK），并可以设置 I<sup>2</sup>C 接口的地址（7 位、10 位地址或广播呼叫地址）。

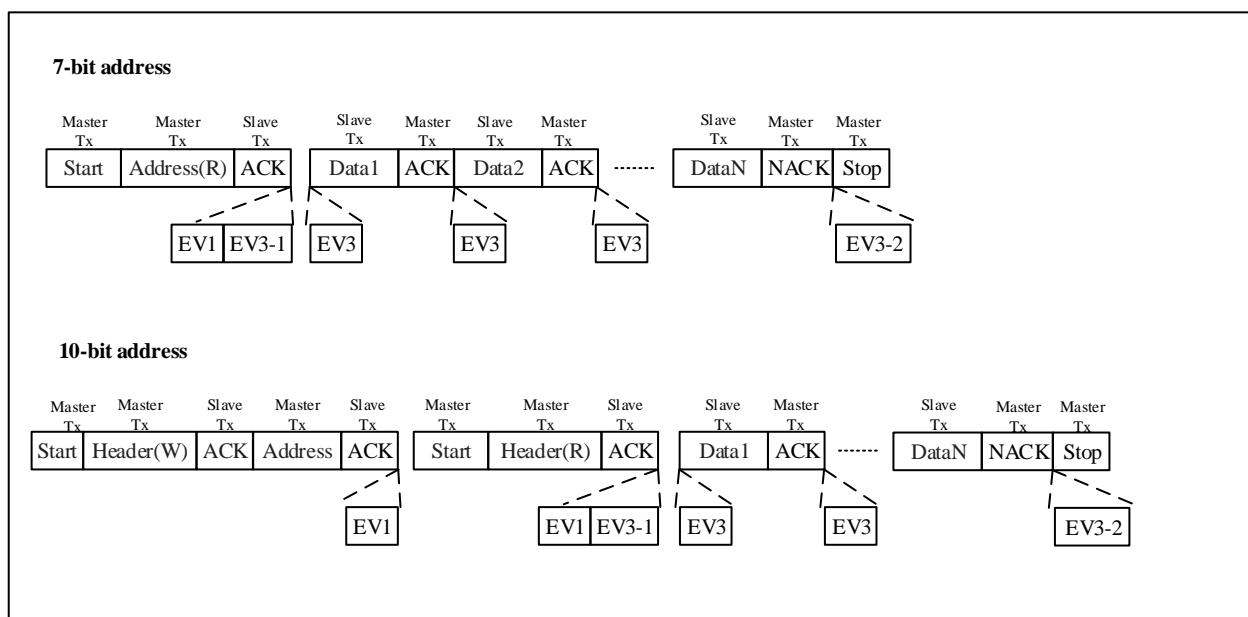
### 17.3.2.4 I<sup>2</sup>C 从机发送模式

在从机模式下发送接收标记位（I2C\_STS2.TRF）指示当前是处于接收器模式还是发送器模式。当在发送器

模式下要发送数据到 I<sup>2</sup>C 总线，软件应该按照下面的步骤操作：

1. 首先，使能 I<sup>2</sup>C 外设时钟，配置 I<sup>2</sup>C\_CTRL1 中时钟相关寄存器，确保输出正确的 I<sup>2</sup>C 时序。当这两步都完成以后，I<sup>2</sup>C 运行在从机模式下，等待接收起始位和地址。
2. I<sup>2</sup>C 从机先收到一个起始位，随后收到匹配的 7 位或 10 地址，I<sup>2</sup>C 硬件将 I<sup>2</sup>C\_STS1.ADDRF 位（接收到了地址并且和自身的地址匹配）置 1，软件应该定期查询此位或者中断监视此位，发现置位后，软件读 I<sup>2</sup>C\_STS1 寄存器然后读 I<sup>2</sup>C\_STS2 寄存器来清除 I<sup>2</sup>C\_STS1.ADDRF 位。如果地址是 10 位格式，I<sup>2</sup>C 主机应该接着再产生一个 START 并发送一个地址头到 I<sup>2</sup>C 总线。从机在检测到 START 和紧接着的地地址头之后会继续将 I<sup>2</sup>C\_STS1.ADDRF 位置 1。软件继续通过读 I<sup>2</sup>C\_STS1 寄存器和接着读 I<sup>2</sup>C\_STS2 寄存器来第二次清除 I<sup>2</sup>C\_STS1.ADDRF 位。
3. I<sup>2</sup>C 进入数据发送状态，现在移位寄存器和数据寄存器 I<sup>2</sup>C\_DAT 都是空的，所以硬件将 I<sup>2</sup>C\_STS1.TXDATE (发送数据空)位置 1。此时软件可以写入第一个字节数据到 I<sup>2</sup>C\_DAT 寄存器，但是，因为写入 I<sup>2</sup>C\_DAT 寄存器的字节被立即移入内部移位寄存器了，所以 I<sup>2</sup>C\_STS1.TXDATE 位并没有被清 0。当移位寄存器非空的时候，I<sup>2</sup>C 开始发送数据到 I<sup>2</sup>C 总线。
4. 第一个字节的发送期间，软件写第二个字节到 I<sup>2</sup>C\_DAT，此时 I<sup>2</sup>C\_DAT 寄存器和移位寄存器都不是空。I<sup>2</sup>C\_STS1.TXDATE 位被清 0。
5. 第一个字节发送完成之后，I<sup>2</sup>C\_STS1.TXDATE 再次被置起，软件写第三个字节到 I<sup>2</sup>C\_DAT，同时 I<sup>2</sup>C\_STS1.TXDATE 位被清 0。在此之后，只要依然有数据待等待被发送且 I<sup>2</sup>C\_STS1.TXDATE 被置 1，软件都可以写入一个字节到 I<sup>2</sup>C\_DAT 寄存器。
6. 倒数第二个字节发送期间，软件写最后一个数据到 I<sup>2</sup>C\_DAT 寄存器来清除 I<sup>2</sup>C\_STS1.TXDATE 标志位，之后就再不用关心 I<sup>2</sup>C\_STS1.TXDATE 的状态。I<sup>2</sup>C\_STS1.TXDATE 位会在倒数第二个字节发送完成后置起，直到检测到 STOP 结束位时被清 0。
7. 根据 I<sup>2</sup>C 协议，I<sup>2</sup>C 主机不会对接收到的最后一个字节发送应答，所以在最后一个字节发送结束后，I<sup>2</sup>C 从机的 ACKFAIL 位（应答出错）会置起以通知软件发送结束。软件写 0 到 I<sup>2</sup>C\_STS1.ACKFAIL 位可以清除此位。

图 17-3 从发送器传送序列



说明:

1. EV1: I2C\_STS1.ADDRF = 1, 读 STS1 然后读 STS2 将清除该事件。
2. EV3-1: I2C\_STS1.TXDATE=1, 移位寄存器空, 数据寄存器空, 写 DAT。
3. EV3: I2C\_STS1.TXDATE=1, 移位寄存器非空, 数据寄存器空, 写 DAT 将清除该事件。
4. EV3-2: I2C\_STS1.ACKFAIL=1, 在 STS1 的 ACKFAIL 位写“0”可清除该事件。

注:

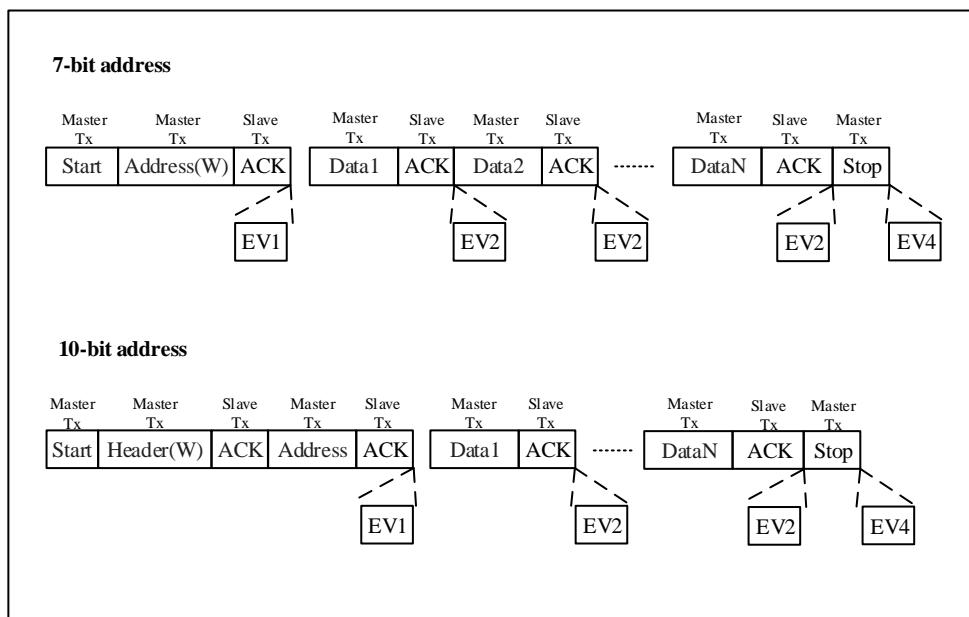
- a) EV1 和 EV3-1 事件拉长 SCL 低的时间, 直到对应的软件序列结束。
- b) EV3 的软件序列必须在当前字节传输结束之前完成。

### 17.3.2.5 I<sup>2</sup>C 从机接收模式

在从机模式下接收数据时, 软件应该按如下步骤操作:

1. 首先, 使能 I<sup>2</sup>C 外设时钟, 配置 I2C\_CTRL1 中时钟相关寄存器, 确保输出正确的 I<sup>2</sup>C 时序。当这两步都完成以后, I<sup>2</sup>C 运行在从机模式下, 等待接收起始位和地址。
2. 在接收到 START 起始条件和匹配的 7 位或 10 地址之后, I<sup>2</sup>C 硬件将 I2C\_STS1.ADDRF 位 (接收到地址并且和自身的地址匹配) 置 1, 此位应该通过软件轮询或者中断来检测, 发现置起后, 软件通过先读 I2C\_STS1 寄存器然后再读 I2C\_STS2 寄存器来清除 I2C\_STS1.ADDRF 位。一旦 I2C\_STS1.ADDRF 位被清 0, I<sup>2</sup>C 从机就开始接收来自 I<sup>2</sup>C 总线的数据。
3. 当接收到第一个字节后, I2C\_STS1.RXDATNE 位 (接收数据非空) 被硬件置 1, 如果设置了 I2C\_CTRL2.EVTINTEN 和 I2C\_CTRL2.BUFINTEN 位, 则产生一个中断。软件应该通过轮询或者中断来检测该位, 一旦发现置起后, 软件可以读取 I2C\_DAT 寄存器的第一个字节, 此时 I2C\_STS1.RXDATNE 位被清 0。注意, 如果设置了 I2C\_CTRL1.ACKEN 位, 则在接收到一个字节后, 从机应该产生一个应答脉冲。
4. 任何时候, 只要 I2C\_STS1.RXDATNE 位被置 1, 软件均可以从 I2C\_DAT 寄存器读取一个字节。当接收到最后一个字节后, I2C\_STS1.RXDATNE 被置 1, 软件读取最后一个字节。
5. 当从机检测到 I<sup>2</sup>C 总线上的停止位 (STOP) 后, 将 I2C\_STS1.STOPF 位置 1, 如果设置了 I2C\_CTRL2.EVTINTEN 位, 则产生一个中断。软件通过先读 I2C\_STS1 寄存器再写 I2C\_CTRL1 寄存器来清除 I2C\_STS1.STOPF 位 ((见下图的 EV4))。

图 17-4 从机接收器传送序列



说明：

1. EV1: I2C\_STS1.ADDRF = 1, 读 STS1 然后读 STS2 将清除该事件。.
2. EV2: I2C\_STS1.RXDATNE = 1, 读 DAT 将清除该事件。
3. EV4: I2C\_STS1.STOPF=1, 读 STS1 然后写 CTRL1 寄存器将清除该事件。

注: a) EV1 事件拉长 SCL 低的时间, 直到对应的软件序列结束。

b) EV2 的软件序列必须在当前字节传输结束之前完成。

### 17.3.2.6 I<sup>2</sup>C 主机发送模式

在主模式时, I<sup>2</sup>C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。当通过 START 位在总线上产生了起始条件, 设备就进入了主模式。

在主机模式下发送数据到 I<sup>2</sup>C 总线时, 软件应该按如下步骤操作:

1. 首先, 使能 I<sup>2</sup>C 外设时钟, 配置 I2C\_CTRL1 中时钟相关寄存器, 确保输出正确的 I<sup>2</sup>C 时序。当这两步都完成以后, I<sup>2</sup>C 默认运行在从机模式下, 等待接收起始位和地址。
2. 当 BUSY=0 时, 设置 I2C\_CTRL1 寄存器中的 STARTGEN 位为 1, I<sup>2</sup>C 接口将产生一个开始条件并切换至主模式 (I2C\_STS2.MSMODE 位置位)。
3. 一旦发出开始条件, I<sup>2</sup>C 硬件将 I2C\_STS1.STARTBF 位 (起始位标志) 置 1 然后进入主机模式, 如果设置了 I2C\_CTRL2.EVTINTEN 位, 则会产生一个中断。接着软件读 I2C\_STS1 寄存器然后写一个 7 位地址位或带有地址头的 10 位地址位到 I2C\_DAT 寄存器来清除 I2C\_STS1.STARTBF 位。I2C\_STS1.STARTBF 位被清 0 后 I<sup>2</sup>C 就开始发送地址或者地址头到 I<sup>2</sup>C 总线。

在 10 位地址模式时, 发送一个头序列会产生以下事件:

- I2C\_STS1.ADDR10F 位被硬件置位, 如果设置了 I2C\_CTRL2.EVTINTEN 位, 则产生一个中断。然后主设备读 STS1 寄存器, 再将第二个地址字节写入 DAT 寄存器。

- I2C\_STS1.ADDRF 位被硬件置位，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断。随后主设备读 STS1 寄存器，跟着读 STS2 寄存器

注：在发送器模式，主设备先发送头字节 (1110xx0)，然后发送从地址的低 8 位。(这里 xx 代表 10 位地址中的最高 2 位)。

在 7 位地址模式时，只需送出一个地址字节，一旦该地址字节被送出：

- I2C\_STS1.ADDRF 位被硬件置位，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断。随后主设备等待一次读 STS1 寄存器，跟着读 STS2 寄存器。

注：在发送器模式，主设备发送从地址时置最低位为‘0’。

注：在 7 位地址模式时，不要将从机地址配置成 0xF0，防止 I2C\_STS1.ADDR10F 位被硬件置位。

4. 7 位或 10 位的地址位发送完成后，I<sup>2</sup>C 硬件将 I2C\_STS1.ADDRF 位（地址已被发送）置 1，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断，软件通过读 I2C\_STS1 寄存器然后读 I2C\_STS2 寄存器来清除 I2C\_STS1.ADDRF

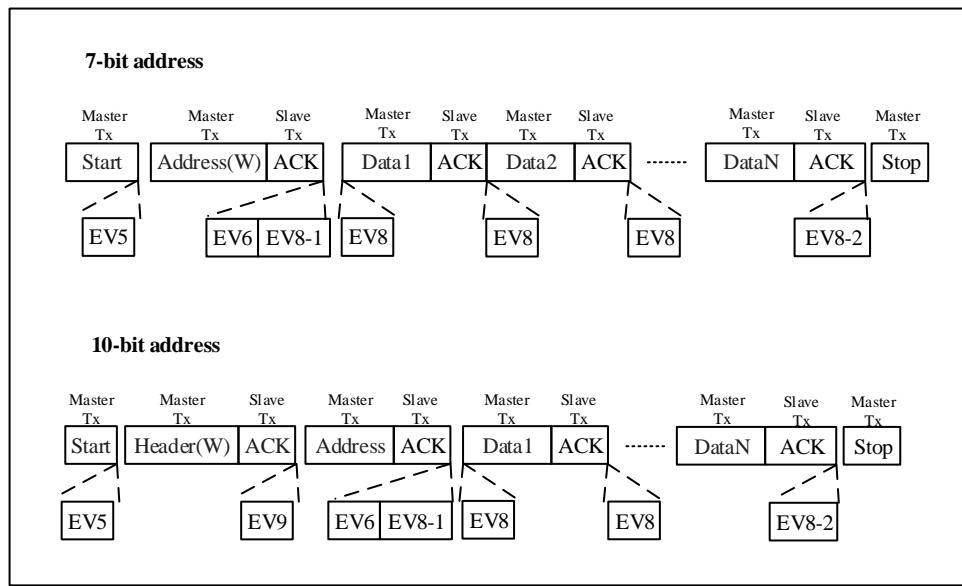
5. I<sup>2</sup>C 进入数据发送状态，因为移位寄存器和数据寄存器（I2C\_DAT）都是空的，所以硬件将 I2C\_STS1.TXDATE 位（发送数据空）置 1，接着软件写第一个字节数据到 I2C\_DAT 寄存器，但是，因为写入 I2C\_DAT 寄存器的字节被立即移入内部移位寄存器，所以 I2C\_STS1.TXDATE 位此时不会被清零。一旦移位寄存器非空，I<sup>2</sup>C 就开始发送数据到总线。

6. 在第一个字节的发送过程中，软件写第二个字节到 I2C\_DAT，此时 I2C\_STS1.TXDATE 被清零。任何时候，只要还有数据等待被发送，且 I2C\_STS1.TXDATE 位被置 1，软件都可以向 I2C\_DAT 寄存器写入一个字节。

7. 在倒数第二个字节发送过程中，软件写入最后一个字节数据到 I2C\_DAT 来清除 I2C\_STS1.TXDATE 标志位，此后就不用关心 I2C\_STS1.TXDATE 位的状态。I2C\_STS1.TXDATE 位会在倒数第二个字节发送完成后被置起，直到发送停止位（STOP）时被清零。

8. 最后一个字节发送结束后，因为移位寄存器和 I2C\_DAT 寄存器此时都为空，I<sup>2</sup>C 主机将 I2C\_STS1.BSF 位（字节发送结束）置位，在清除 I2C\_STS1.BSF 位之前 I<sup>2</sup>C 接口将保持 SCL 为低电平；读出 I2C\_STS1 之后再写入 I2C\_DAT 寄存器将清除 I2C\_STS1.BSF 位。软件此时设置 I2C\_CTRL1.STOPGEN 位产生一个停止条件（见图 22-5 的 EV8\_2），然后 I<sup>2</sup>C 接口将自动回到从模式（I2C\_STS2.MSMODE 位清除）。

图 17-5 主发送器传送序列



说明：

1. EV5: I2C\_STS1.STARTBF = 1, 读 STS1 然后将地址写入 DAT 寄存器将清除该事件。
2. EV6: I2C\_STS1.ADDRF = 1, 读 STS1 然后读 STS2 将清除该事件。
3. EV8\_1: I2C\_STS1.TXDATE = 1, 移位寄存器空, 数据寄存器空, 写 DAT 寄存器。
4. EV8: I2C\_STS1.TXDATE = 1, 移位寄存器非空, 数据寄存器空, 写 DAT 寄存器将清除该事件。
5. EV8\_2: I2C\_STS1.TXDATE = 1, I2C\_STS1.BSF = 1, 请求设置停止位。这两个事件由硬件在产生停止条件时清除。
6. EV9: I2C\_STS1.ADDR10F = 1, 读 STS1 然后写入 DAT 寄存器将清除该事件。

注: a) EV5、EV6、EV9、EV8\_1 和 EV8\_2 事件拉长 SCL 低的时间, 直到对应的软件序列结束。

b) EV8 的软件序列必须在当前字节传输结束之前完成。

c) 当 TXDATE 或 BSF 位置位时, 停止条件应安排在出现 EV8\_2 事件时。

### 17.3.2.7 I<sup>2</sup>C 主机接收模式

在主机模式下从 I<sup>2</sup>C 总线接收数据软件应该按如下步骤操作：

1. 首先, 使能 I<sup>2</sup>C 外设时钟, 配置 I2C\_CTRL1 中时钟相关寄存器, 确保输出正确的 I<sup>2</sup>C 时序。使能和配置以后, I<sup>2</sup>C 默认运行在从机模式下, 等待接收起始位和地址。
2. 当 BUSY=0 时, 设置 I2C\_CTRL1.STARTGEN 位为 1, I<sup>2</sup>C 接口将产生一个开始条件并切换至主模式 (I2C\_STS2.MSMODE 位置位),
3. 一旦发出开始条件, I<sup>2</sup>C 硬件将 I2C\_STS1.STARTBF 位 (起始位标志) 置 1 然后进入主机模式, 如果设置了 I2C\_CTRL2.EVTINTEN 位, 则会产生一个中断。接着软件读 I2C\_STS1 寄存器然后写一个 7 位地址位或带有地址头的 10 位地址位到 I2C\_DAT 寄存器来清除 I2C\_STS1.STARTBF 位。I2C\_STS1.STARTBF 位被清 0 后 I<sup>2</sup>C 就开始发送地址或者地址头到 I<sup>2</sup>C 总线。

在 10 位地址模式时，发送一个头序列会产生以下事件：

- I2C\_STS1.ADDR10F 位被硬件置位，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断。然后主设备读 STS1 寄存器，再将第二个地址字节写入 DAT 寄存器。
- I2C\_STS1.ADDRF 位被硬件置位，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断。随后主设备读 STS1 寄存器，跟着读 STS2 寄存器。

注：在接收器模式，主设备先发送头字节（11110xx0），然后发送从地址的低 8 位，然后再重新发送一个开始条件，后面跟着头字节（11110xx1）（这里 xx 代表 10 位地址中的最高 2 位）。

在 7 位地址模式时，只需送出一个地址字节，一旦该地址字节被送出：

- I2C\_STS1.ADDRF 位被硬件置位，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断。随后主设备等待一次读 STS1 寄存器，跟着读 STS2 寄存器。

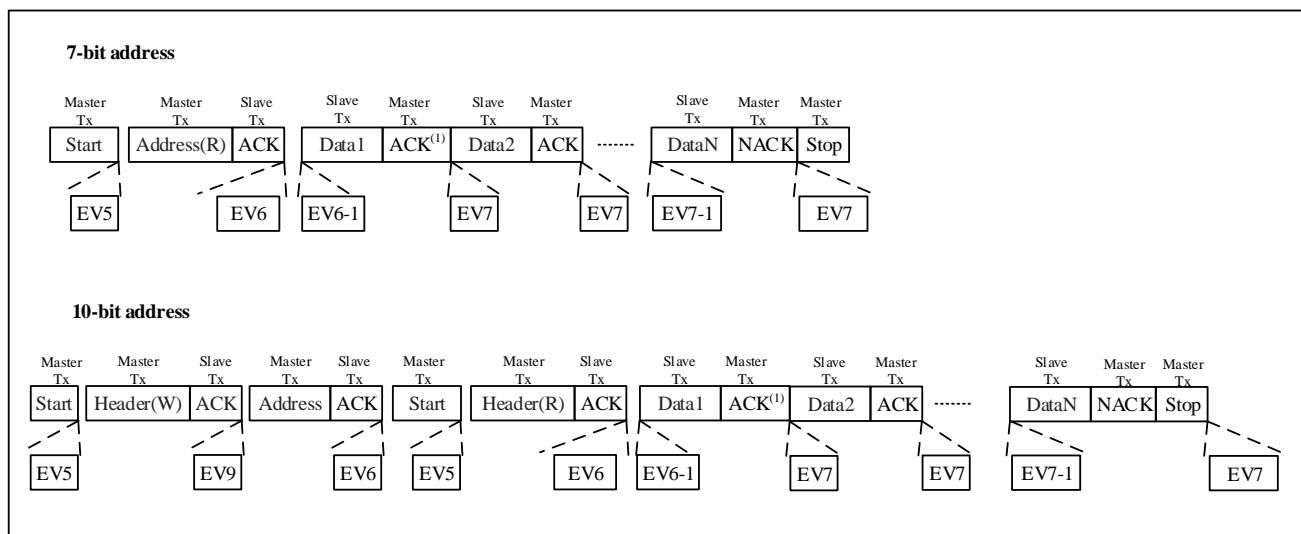
注：在接收器模式，主设备发送从地址时置最低位为‘1’。

注：在 7 位地址模式时，不要将从机地址配置成 0xF0，防止 I2C\_STS1.ADDR10F 位被硬件置位。

4. 7 位或 10 位的地址位发送完成后，I<sup>2</sup>C 硬件将 I2C\_STS1.ADDRF 位（地址已被发送）置 1，如果设置了 I2C\_CTRL2.EVTINTEN 位，则产生一个中断，软件通过读 I2C\_STS1 寄存器然后读 I2C\_STS2 寄存器来清除 I2C\_STS1.ADDRF，在发送地址和清除 I2C\_STS1.ADDRF 之后，如果地址是 10 位格式，软件应该再次将 STARTGEN 位置 1 来重新产生一个 START。在 START 产生后，I2C\_STS1.STARTBF 位会被置 1。软件应该通过先读 I2C\_STS1 然后写地址头到 I2C\_DAT 来清除 I2C\_STS1.STARTBF 位，然后地址头被发到 I<sup>2</sup>C 总线，I2C\_STS1.ADDRF 再次被置 1。软件应该再次通过先读 I2C\_STS1 然后读 I2C\_STS2 来清除 I2C\_STS1.ADDRF 位。
5. 在发送完地址和清除 I2C\_STS1.ADDRF 之后，PC 接口进入主机接收器模式。在此模式下，I<sup>2</sup>C 接口从 SDA 线接收数据字节，并通过内部移位寄存器送至 DAT 寄存器。一旦接收到第一个字节，硬件会将 I2C\_STS1.RXDATNE 位（接收数据非空标志位）置 1，如果 ACKEN 位被置位，发出一个应答脉冲。此时软件可以从 I2C\_DAT 寄存器读取第一个字节，之后 I2C\_STS1.RXDATNE 位被清 0。此后，只要 I2C\_STS1.RXDATNE 被置 1，软件就可以从 I2C\_DAT 寄存器读取一个字节。
6. 主设备在从从设备接收到最后一个字节后发送一个 NACK。接收到 NACK 后，从设备释放对 SCL 和 SDA 线的控制；主设备就可以发送一个停止/重新起始条件。为了在收到最后一个字节后产生一个 NACK 脉冲，接收完倒数第二个字节(N-1)数据之后，软件应该立即清除 ACKEN 位。为了产生一个停止/重新起始条件，软件必须在读倒数第二个数据字节之后将 I2C\_CTRL1.STOPGEN 位或者 I2C\_CTRL1.STARTGEN 置 1，这一过程需要在最后一个字节接收完毕之前完成，以确保 NACK 发送给最后一个字节。
7. 最后一个字节接收完毕后，I2C\_STS1.RXDATNE 位被置 1，软件可以读取最后一个字节。由于 I2C\_CTRL1.ACKEN 已经在前一步骤中被清 0，I<sup>2</sup>C 不再为最后一个字节发送 ACK，并在最后一个字节发送完毕后产生一个 STOP 停止位。

注意：以上步骤要求字节数目  $N > 1$ ，如果  $N = 1$ ，步骤 6 应该在步骤 4 之后就执行，且需要在字节接收完成之前完成。

图 17-6 主接收器传送序列图



说明:

1. EV5: I2C\_STS1.STARTBF=1, 读 STS1 然后将地址写入 DAT 寄存器清除该事件。
2. EV6: I2C\_STS1.ADDRF=1, 读 STS1 然后读 STS2 将清除该事件。在 10 位主接收模式下，该事件后应设置 CTRL1 的 STARTGEN=1。
3. EV6\_1: 没有对应的事件标志，至适于接收 1 个字节的情况。恰好在 EV6 之后（即清除了 ADDRF 之后），要清除响应和停止条件的产生位。
4. EV7: I2C\_STS1.RXDATNE=1, 读 DAT 寄存器消除该事件。
5. EV7\_1: I2C\_STS1.RXDATNE =1, 读 DAT 寄存器清除该事件。设置 I2C\_CTRL1.ACKEN=0 和 I2C\_CTRL1.STOPGEN=1。
6. EV9: I2C\_STS1.ADDR10F=1, 读 STS1 然后写入 DAT 寄存器将清除该事件。

注:

- a) 如果收到一个单独的字节，则是NA。
- b) EV5、EV6和EV9事件拉长SCL低电平，直到对应的软件序列结束。
- c) EV7的软件序列必须在当前字节传输结束前完成。
- d) EV6\_1或EV7\_1的软件序列必须在当前传输字节的ACK脉冲之前完成。

### 17.3.3 错误条件

I<sup>C</sup> 的错误主要有总线错误、应答错误、仲裁丢失、过载/欠载错误。这些错误都可能造成通讯的失败。

#### 17.3.3.1 应答错误 (ACKFAIL)

当接口检测到应答位与期望不符时，将产生应答错误。此时 I2C\_STS1.ACKFAIL 被置位，如果设置了 I2C\_CTRL2.ERRINTEN 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：如果处于从模式，硬件会释放总线。如果是处于主模式，软件必须生产一个停止条件。

### 17.3.3.2 总线错误 (BUSERR)

在一个地址或数据字节传输期间,当 I<sup>2</sup>C 接口检测到一个外部的停止或起始条件则产生总线错误, I2C\_STS1.BUSERR 被置位。如果设置了 I2C\_CTRL2.ERRINTEN 位为“1”, 则产生一个中断。

在主模式情况下,硬件不释放总线,同时不影响当前的传输状态。此时由软件决定是否要中止当前的传输。

在从模式情况下,数据被丢弃,硬件释放总线。如果检测到错误的开始条件,从设备认为是一个重启动,并等待地址或停止条件。如果检测到错误的停止条件,从设备按正常的停止条件操作,同时硬件释放总线。

### 17.3.3.3 仲裁丢失 (ARLOST)

当 I<sup>2</sup>C 接口检测到仲裁丢失时产生仲裁丢失错误, I2C\_STS1.ARLOST 位被置位。如果设置了 I2C\_CTRL2.ERRINTEN 位为“1”, 则产生一个中断。

I<sup>2</sup>C 接口自动回到从模式 (I2C\_STS2.MSMODE 位被清除)。当 I<sup>2</sup>C 接口丢失了仲裁,则它无法在同一个传输中响应它的从地址,但它可以在赢得总线的主设备重新发送起始条件之后响应。否则硬件释放总线。

### 17.3.3.4 过载/欠载错误 (OVERRUN)

在从机接收模式下,如果禁止时钟延长,容易发生过载/欠载错误。

当它已经接收到一个字节 (I2C\_STS1.RXDATNE=1),但在 DAT 寄存器中前一个字节数据还没有被读出,则发生过载错误,数据寄存器最后接收的数据被丢弃;同时软件应清除 I2C\_STS1.RXDATNE 位,发送器重新发送最后一次发送的字节。

在从机发送模式下,如果禁止时钟延长,在当前字节已经发送完成,而 DAT 仍然为空(I2C\_STS1.TXDATE=1),则发生欠载错误。此时,在 DAT 寄存器中的前一个字节将被重复发出;用户应该确定在发生欠载错时,接收端应丢弃重复接收到的数据。发送端应按 I<sup>2</sup>C 总线标准在规定的时间更新 DAT 寄存器。

在发送第一个字节时,必须在清除 I2C\_STS1.ADDRF 之后并且第一个 SCL 上升沿之前写入 DAT 寄存器;如果不能做到这点,则接收方应该丢弃第一个数据。

## 17.3.4 DMA 应用

在传输时,当数据寄存器为空或满时,能够生成 DMA 请求。DMA 能够写数据到 I<sup>2</sup>C 数据寄存器,或从 I<sup>2</sup>C 数据寄存器读出数据以减少 CPU 的开销。

DMA 请求必须在当前字节传输结束之前被响应。当相应 DMA 通道设置的数据传输已经完成时, DMA 控制器发送传输结束信号 EOT 到 I<sup>2</sup>C 接口,并且在中断使能时产生一个中断。

在主机发送模式,在 EOT 中断服务程序中,需禁止 DMA 请求,然后在等到 I2C\_STS1.BSF 事件后设置停止条件。

在主机接收模式,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,它对应 DMA 传输(字节数-1)。如果在寄存器中设置了 I2C\_CTRL2.DMALAST 位,硬件在发送完 EOT\_1 后的下一个字节,将自动发送 NACK。在中断允许的情况下,用户可以在 DMA 传输完成的中断服务程序中产生一个停止条件。

### 17.3.4.1 DMA 发送

DMA 模式通过 I2C\_CTRL2.DMAEN 位使能。只要 I2C\_STS1.TXDATE 位被置位,数据将由 DMA 从预置的存储区装载进 I2C\_DAT 寄存器。设置 DMA 通道进行 I<sup>2</sup>C 发送,须执行以下步骤 (x 是通道号):

1. 在 DMA\_PADDRx 寄存器中设置 I2C\_DAT 寄存器地址。数据将在每个 I2C\_STS1.TXDATE 事件后从存储器传送至这个地址。
2. 在 DMA\_MADDRx 寄存器中设置存储器地址。数据在每个 I2C\_STS1.TXDATE 事件后从这个存储区传送至 I2C\_DAT。
3. 在 DMA\_TXNUMx 寄存器中设置所需传输的字节数。在每个 I2C\_STS1.TXDATE 事件后，此值递减，直到 0。
4. 利用 DMA\_CHCFGx 寄存器中的 PRIOLVL[1:0]位配置通道优先级。
5. 设置 DMA\_CHCFGx 寄存器中的 DIR 位，并根据应用要求可以配置在整个传输完成一半或全部完成时发出中断请求。
6. 通过设置 DMA\_CHCFGx 寄存器上的 CHEN 位激活通道。
7. 当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I2C 接口发送一个传输结束的 EOT/EOT\_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

*注：如果使用 DMA 进行发送时，不要设置 I2C\_CTRL2 寄存器的 BUFINTEN 位。*

#### 17.3.4.2 DMA 接收

DMA 模式通过 I2C\_CTRL2 寄存器中的 DMAEN 位使能。每次接收到数据字节时，将由 DMA 把 I2C\_DAT 寄存器的数据传送到设置的存储区。设置 DMA 通道进行 I2C 接收，须执行以下步骤（x 是通道号）

1. 在 DMA\_PADDRx 寄存器中设置 I2C\_DAT 寄存器的地址。数据将在每次 I2C\_STS1.RXDATNE 事件后从此地址传送到存储区。
2. 在 DMA\_MADDRx 寄存器中设置存储区地址。数据将在每次 I2C\_STS1.RXDATNE 事件后从 I2C\_DAT 寄存器传送到此存储区。
3. 在 DMA\_TXNUMx 寄存器中设置所需的传输字节数。在每个 I2C\_STS1.RXDATNE 事件后，此值递减，直到 0。
4. 用 DMA\_CHCFGx 寄存器中的 PRIOLVL[1:0]配置通道优先级。
5. 清除 DMA\_CHCFGx 寄存器中的 DIR 位，根据应用要求可以设置在数据传输完成一半或全部完成时发出中断请求。
6. 设置 DMA\_CHCFGx 寄存器中的 CHEN 位激活该通道。
7. 当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I2C 接口发送一个传输结束的 EOT/EOT\_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

*注：如果使用 DMA 进行接收时，不要设置 I2C\_CTRL2 寄存器的 BUFINTEN 位。*

#### 17.3.5 包错误校验 (PEC)

将 I2C\_CTRL1 寄存器的 PECEN 位置 1 就可以使能 PEC 功能，PEC 使用 CRC-8 算法对包括地址和读/写位在内的所有信息字节进行计算，从而提高通信的可靠性。包错误校验 (PEC) 计算器使用的 CRC-8 多项式为  $C(x) = x^8 + x^2 + x + 1$ 。

在发送模式时，软件可以在最后一个 I2C\_STS1.TXDATE 事件时设置 I2C\_CTRL1 寄存器的 PEC 传输位，PEC 将在最后一个字节后被发送。在接收模式时，软件在最后一个 I2C\_STS1.RXDATNE 事件之后设置

I<sup>2</sup>C\_CTRL1.PEC 位，然后接收 PEC 字节，并将接收到的 PEC 字节与内部计算的 PEC 值进行比较。如果不等于内部计算的 PEC，接收器发送一个 NACK。如果是主机接收器模式，不管校对的结果如何，PEC 后都将发送 NACK。需要注意，I<sup>2</sup>C\_CTRL1.PEC 位必须在接收当前字节的 ACK 脉冲之前设置。

如果 DMA 和 PEC 计算器都被激活，I<sup>2</sup>C 将自动发送或者检查 PEC 值。

在发送模式时，当 I<sup>2</sup>C 接口从 DMA 控制器处接收到 EOT 信号时，它在最后一个字节后自动发送 PEC。在接收模式时，当 I<sup>2</sup>C 接口从 DMA 处接收到一个 EOT\_1 信号时，它将自动把下一个字节作为 PEC，并且和内部计算的 PEC 进行比较。在接收到 PEC 后产生一个 DMA 请求。

为了允许中间 PEC 传输，I<sup>2</sup>C\_CTRL2.DMALAST 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。

当仲裁丢失的时候，PEC 计算失效。

### 17.3.6 噪声滤波

I<sup>2</sup>C 接口标准要求能够过滤在 SCL/SDA 线上的 50ns 的毛刺，因此设计了模拟滤波器和数字滤波器。模拟滤波器默认时开启的，也可以通过设置 I<sup>2</sup>C\_TMRSE.SCLAFENN/SDAAFENN 位禁用此功能。模拟滤波器通过配置 I<sup>2</sup>C\_TMRSE.SCLAFE/SDAAFW 去设置过滤毛刺的宽度 5ns,15ns,25ns,35ns。数字滤波器通过设置 I<sup>2</sup>C\_TMRSE.SCLDFW/SDADFW 为非 0 值去使能。其最大滤波宽度为 SCLDEW[3:0]/SDADFW[3:0]\*T<sub>PCLK</sub>。使能数字滤波器将增加 SDA 线的保持时间，增加值为(SDADFW[3:0]+1)\*T<sub>PCLK</sub>。

表 17-1 SDADFW/ SCLDFW 推荐配置

APB1 时钟	SDADFW/ SCLDFW 最大值	
	标准模式	快速模式
2MHz<=APB1<=5MHz	0	0
5MHz<APB1<=10MHz	10	0
10MHz<APB1<=20MHz	15	2
20MHz<APB1<=30MHz	15	2
30MHz<APB1<=32MHz	15	3

### 17.3.7 SMBus

#### 17.3.7.1 介绍

系统管理总线（System Management Bus，简写为 SMBus 或 SMB）是一种结构简单的单端双线制总线。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。SMBus 是 I<sup>2</sup>C 的一种衍生总线形式，为系统和电源管理相关的任务提供一条控制总线。SMBus 基于 I<sup>2</sup>C 通信标准，是一个与系统管理和电源管理相关的控制总线。想要了解更多信息，请参考 SMBus 规范 V2.0(<http://smbus.org/specs/>)。

SMBus 有三类设备标准：

- 主设备：发送命令、产生时钟和终止发送设备；
- 从设备：接收或响应命令设备；

- **主机：**一个系统仅有一个主机，它提供与系统 CPU 的主接口。主机具有主-从机功能并必须支持 SMBus 提醒协议。

SMBus 与 I2C 的相似点：

- 总线协议都是两条线（一个时钟线 SCL 和一个数据线 SDA），再加一条可选的 SMBus 提醒线；
- 数据格式相似，SMBus 数据格式类似于 I2C 的 7 位地址格式（见图 17-2）；
- 都是主-从通信模式，且主设备提供时钟；
- 都支持多主机功能；

SMBus 和 I2C 的不同点：

表 17-2 SMBus 与 I<sup>2</sup>C 的比较

SMBus	I <sup>2</sup> C
最大传输速度 100KHz	最大传输速度 1MHz
最小传输速度 10KHz	无最小传输速度
35ms 时钟低超时	无时钟超时
固定逻辑电平	逻辑电平由 VDD 决定
不同的地址类型（保留的，动态的等）	7 位、10 位和广播呼叫从地址类型
不同的总线协议(快递命令，处理呼叫等)	无总线协议

### 17.3.7.2 SMBus 用途

SMBus 利用系统管理总线，可实现轻量级的通信需求。一般来说，SMBus 最常见于计算机主板，主要用于电源传输 ON/OFF 指令的通信，为系统和电源管理相关的任务提供控制总线。

### 17.3.7.3 设备标识

在 SMBus 中，任意一个设备作为从设备时都有一个地址，叫从地址。

为了给每一个设备分配地址，必须有一个唯一的设备标识（UDID）分配给设备。

### 17.3.7.4 总线协议

SMBus 技术规范包括 9 个总线协议。想要了解关于 SMBus 的详细信息和地址类型请参考 SMBus 技术规范 V2.0(<http://smbus.org/specs/>)。用户可以决定使用哪些协议。

**注：SMBus 不支持 Quick command 协议**

SMBus 上每个报文交互都遵从 SMBus 协议中预定义的格式。SMBus 是 I2C 规范中数据传输格式的子集。只要 I2C 设备可通过 SMBus 协议之一进行访问，便视为兼容 SMBus 规范。

### 17.3.7.5 地址解析协议 (ARP)

通过 SMBus 协议动态分配新的唯一地址给每个从设备来解决地址冲突，这是地址解析协议（ARP）。

任何一个 SMBus 主设备都可以遍历总线。

使用 SMBus 物理层仲裁机制分配地址。当设备维持供电期间，分配的地址保持不变，协议也允许再断电后保留其地址。

地址分配之后，没有额外的 SMBus 打包开销（访问分配地址的设备与访问固定地址的设备所用的时间是一样的）。

### 17.3.7.6 超时错误

SMBus 有一种超时特性：假如某个通信耗时太久，便会自动复位设备。这就是为什么 SMBus 有最小传输速率的要求——为了防止超时后长时间锁死总线。I<sup>2</sup>C 在本质上可以视为一个“直流”总线，也就是说当主机正在访问从机的时候，假如从机正在执行一些子程序无法及时响应，从机可以拉住主机的时钟。这样便可以提醒主机：从机正忙，但并不想放弃当前的通信。从机的当前任务结束之后，将可以继续 I<sup>2</sup>C 会话。I<sup>2</sup>C 总线协议中并没有限制这个延时的上限，但在 SMBus 系统中，这个时间被限定为 35ms。按照 SMBus 协议的假定，如果某个会话耗时太久，就意味着总线出了问题，此时所有设备都应当复位以消除这种（问题）状态。这样就不允许从设备将时钟拉低太长时间。I2C\_STS1.TIMOUT 错误表明了这个特性的状态。

### 17.3.7.7 SMBus 提醒模式

SMBus 提供了一个可选的中断信号 SMBALERT（与 SCL 和 SDA 一样，是一种有线与信号），设备使用该信号来扩展其控制能力，但需要牺牲一个引脚。SMBALERT 通常和 SMBus 广播呼叫地址结合使用。关于 SMBus 的消息有 2 个字节。

仅具有从机功能的设备可以设置 I2C\_CTRL1.SMBALERT 位以指示它要与主机通信。主机处理该中断并通过提醒响应地址 ARA (Alert Response Address, 地址值为 0001100x) 访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应答 ARA。此状态是由 I2C\_STS1.SMBALERT 来标识的。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’。

当多个设备的 SMBALERT 为低时，在地址传输过程中，最高优先级（地址越小优先级越高）可以通过标准仲裁赢得总线通信。如果确认从机地址，则设备的 SMBALERT 不再为低。如果报文传输完毕，设备的 SMBALERT 仍为低，表示主机将再次读取 ARA。没有采用 SMBALERT 信号时主机可以定期访问 ARA。

### 17.3.7.8 SMBus 通信流程

SMBus 的通讯流程和标准 I<sup>2</sup>C 的流程相似。如果要使用 SMBus 模式，还需要在程序中配置 SMBus 特定寄存器、并响应 SMBus 特定标志位、实现在 SMBus 手册中介绍的上层协议。

- 首先，设置 I2C\_CTRL1.SMBMODE 位；并按照应用要求配置 I2C\_CTRL1.SMBTYPE 和 I2C\_CTRL1.AR PEN 位。如果 I2C\_CTRL1.AR PEN=1 且 I2C\_CTRL1.SMBTYPE=0，使用 SMB 设备默认地址；如果 I2C\_CTRL1.AR PEN=1 且 I2C\_CTRL1.SMBTYPE=1，使用 SMB 主设备头字段。
- 为了支持 ARP 协议 (I2C\_CTRL1.AR PEN=1)，在 SMBus 主机模式下 (I2C\_CTRL1.SMBTYPE=1)，软件需要响应标志位 I2C\_STS2.SMBHADDR (在 SMBus 从机模式下，响应 I2C\_STS2.SMBDADDR 标志位)，并实现 ARP 协议中的功能。
- 为了支持 SMBus 警告模式，软件应该响应 I2C\_STS1.SMBALERT 标志位，并实现相应的功能。

## 17.4 调试模式

当微控制器进入调试模式 (Cortex-M4 核心处于停止状态) 时，根据 DBG 模块中的 DBG\_CTRL.I2Cx\_SMBUS\_TIMEOUT 配置位，SMBUS 超时控制或者继续正常工作或者可以停止。详见 23.3.2 节。

## 17.5 中断请求

下表列出了所有的 I<sup>2</sup>C 中断请求：

表 17-3 I<sup>2</sup>C 中断请求

中断功能	中断事件	事件标志	设置控制位
I2C 事件中断	起始位已发送 (主)	STARTBF	EVTINTEN
	地址已发送 (主) 或地址匹配 (从)	ADDRF	
	10 位头段地址已发送 (主)	ADDR10F	
	收到停止位 (从)	STOPF	
	数据字节传输完成	BSF	
	接收缓冲区非空	RXDATNE	
	发送缓冲区为空	TXDATE	
I2C 错误中断	总线错误	BUSERR	ERRINTEN
	仲裁丢失 (主)	ARLOST	
	应答失败	ACKFAIL	
	过载/欠载	OVERRUN	
	PEC 错误	PECERR	
	超时/Tlow 错误	TIMOUT	
	SMBus 提醒	SMBALERT	

注: 1. STARTBF, ADDR, ADDR10F, STOPF, BSF, RXDATNE 和 TXDATE 通过逻辑或汇到同一个中断通道中。

2. BUSERR、ARLOST、ACKFAIL、OVERRUN、PECERR、TIMOUT 和 SMBALERT 通过逻辑或汇到同一个中断通道中。

## 17.6 I<sup>2</sup>C 寄存器描述

可以用半字（16位）或字（32位）的方式操作这些外设寄存器。

### 17.6.1 I<sup>2</sup>C 寄存器总览

表 17-4 I<sup>2</sup>C 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
000h	I2C_CTRL1	Reserved														0	SWRESET	Reserved														0	0	0	0	0	0		
	Reset Value															0	SMBALERT	Reserved														0	0	0	0	0	0		
004h	I2C_CTRL2	Reserved														0	PFC	Reserved														0	0	0	0	0	0		
	Reset Value															0	DMAEN	Reserved														0	0	0	0	0	0		
008h	I2C_OADDR1	Reserved														0	ADDRMODE	Reserved														0	0	0	0	0	0		
	Reset Value															0	DMALAST	Reserved														0	0	0	0	0	0		
00Ch	I2C_OADDR2	Reserved														0	BLFINTEN	Reserved														0	0	0	0	0	0		
	Reset Value															0	EVINTEN	Reserved														0	0	0	0	0	0		
010h	I2C_DAT	Reserved														0	STORGEN	Reserved														0	0	0	0	0	0		
	Reset Value															0	STARTGEN	Reserved														0	0	0	0	0	0		
014h	I2C_STS1	Reserved														0	ADDR[9:8]	Reserved														0	0	0	0	0	0		
	Reset Value															0	DATA[7:0]	Reserved														0	0	0	0	0	0		
018h	I2C_STS2	Reserved														0	TXDATE	Reserved														0	0	0	0	0	0		
	Reset Value															0	PECERR	Reserved														0	0	0	0	0	0		
01Ch	I2C_CLKCTRL	Reserved														0	OVERRUN	Reserved														0	0	0	0	0	0		
	Reset Value															0	ACKFAIL	Reserved														0	0	0	0	0	0		
020h	I2C_TMRISE	SCLAFFENN	SCLAFW[1:0]				SDAFAENN				SDAFAFW[1:0]				SCLDFW[3:0]				SDADFW[3:0]				Reserved				TMRISE[5:0]				0	0	0	0	0	0			
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### 17.6.2 I<sup>2</sup>C 控制寄存器 1 (I2C\_CTRL1)

地址偏移: 0x00

复位值: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW RESET	Reserved	SMB ALERT	PEC	ACK POS	ACKEN	STOP GEN	START GEN	NO EXTEND	GCEN	PECEN	ARPEN	SMB TYPE	Reserved	SMB MODE	EN

位域	名称	描述
15	SWRESET	<p>软件复位 (Software reset)</p> <p>在复位该位前确认 I<sup>2</sup>C 的引脚被释放，总线是空的。</p> <p>0: I<sup>2</sup>C 模块不处于复位状态； 1: I<sup>2</sup>C 模块处于复位状态。</p> <p>注：该位可以用于I<sup>2</sup>C_STS2.BUSY 位为‘1’，在总线上又没有检测到停止条件时。</p>
14	Reserved	保留，必须保持复位值
13	SMBALERT	<p>SMBus提醒 (SMBus alert)</p> <p>软件可以设置或清除该位；当I<sup>2</sup>C_CTRL1.EN=0时，由硬件清除。</p> <p>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。</p>
12	PEC	<p>数据包出错检测 (Packet error checking)</p> <p>软件可以设置或清除该位；当传送完PEC后或检测到起始或停止条件时或当I<sup>2</sup>C_CTRL1.EN=0时硬件将其清除。</p> <p>0: 无 PEC 1: PEC传输</p> <p>注：仲裁丢失时，PEC 的计算失效。</p>
11	ACKPOS	<p>应答/PEC位置 (Acknowledge/PEC Position (for data reception))</p> <p>软件可以设置或清除该位，或当I<sup>2</sup>C_CTRL1.EN=0时，由硬件清除。</p> <p>0: I<sup>2</sup>C_CTRL1.ACKEN 位决定是否向目前正在接收的字节发送ACK；I<sup>2</sup>C_CTRL1.PEC 位表示当前移位寄存器中的字节为 PEC。 1: I<sup>2</sup>C_CTRL1.ACKEN 位决定是否向下一个接收到的字节发送ACK；I<sup>2</sup>C_CTRL1.PEC 位指示移位寄存器中接收到的下一个字节是 PEC。</p> <p>注：ACKPOS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，I<sup>2</sup>C_CTRL1.ACKEN 位必须在 I<sup>2</sup>C_STS1.ADDRF 位清零后清零。 为了检测第 2 个字节的 PEC，必须在配置 ACKPOS 位之后，扩展 ADDR 事件时设置 I<sup>2</sup>C_CTRL1.PEC 位。</p>
10	ACKEN	<p>应答使能 (Acknowledge enable)</p> <p>软件可以设置或清除该位，或当I<sup>2</sup>C_CTRL1.EN=0时，由硬件清除。</p> <p>0: 无应答返回； 1: 在接收到一个字节后返回一个应答 (匹配的地址或数据)。</p>
9	STOPGEN	<p>停止条件产生 (Stop generation)</p> <p>软件可以设置或清除该位；或当检测到停止条件时，由硬件清除；当检测到SMBus超时错误时，硬件将其置位。</p> <p>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。</p> <p>在从模式下： 0: 无停止条件产生； 1: 在当前字节传输或释放SCL和SDA线。</p> <p>注：当设置了STOPGEN、STARTGEN 或 PEC 位，在硬件清除这个位之前，软件不要执行任何对I<sup>2</sup>C_CTRL1 的写操作；否则有可能会第 2 次设置STOPGEN、STARTGEN 或 PEC 位。</p>
8	STARTGEN	起始条件产生 (Start generation)

位域	名称	描述
		软件可以设置或清除该位，当起始条件发出后或I2C_CTRL1.EN=0时，由硬件清除。 0: 无起始条件产生； 1: 产生起始条件。
7	NOEXTEND	禁止时钟延长（从模式）（Clock stretching disable （Slave mode）） 该位决定在从机模式下数据未就绪（I2C_STS1.ADDRF 或 I2C_STS1.BSF 标志置位）时是否拉低 SCL。通过软件复位清零。 0: 允许时钟延长； 1: 禁止时钟延长。
6	GCEN	广播呼叫使能（General call enable） 0: 禁止广播呼叫。不应答(NACK)地址 00h; 1: 允许广播呼叫. 以应答(ACK)地址 00h。
5	PECEN	PEC使能（PEC enable） 0: 禁止PEC模式； 1: 开启 PEC 模式。
4	ARPEN	ARP使能（ARP enable） 0: 禁止ARP； 1: 使能ARP。 如果I2C_CTRL1.SMBTYPE=0, 使用SMBus设备的默认地址。 如果 I2C_CTRL1.SMBTYPE=1, 使用 SMBus 的主地址。
3	SMBTYPE	SMBus类型（SMBus type） 0: SMBus设备； 1: SMBus 主机。
2	Reserved	保留，必须保持复位值
1	SMBMODE	SMBus模式（SMBus mode） 0: I2C模式； 1: SMBus 模式。
0	EN	I <sup>2</sup> C模块使能（Peripheral enable） 0: 禁用I <sup>2</sup> C模块； 1: 启用I <sup>2</sup> C模块。 注：如果清除该位时通讯正在进行，在当前通讯结束后，I <sup>2</sup> C模块被禁用并返回空闲状态。 由于在通讯结束后发生EN=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。

### 17.6.3 I<sup>2</sup>C 控制寄存器 2 (I2C\_CTRL2)

地址偏移: 0x04

复位值: 0x0000

15	13	12	11	10	9	8	7	6	5	0
Reserved	DMA LAST	DMA EN	BUFINTEEN	EVTINTEN	ERRINTEN	Reserved			CLKFREQ[5:0]	rw

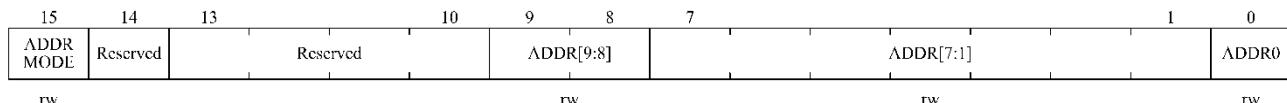
位域	名称	描述
15:13	Reserved	保留, 必须保持复位值
12	DMALAST	DMA最后一次传输 (DMA last transfer) 0: 下一次DMA的EOT不是最后的传输; 1: 下一次DMA的EOT是最后的传输。 <i>注: 该位在主接收模式使用, 使得在最后一次接收数据时可以产生一个NACK。</i>
11	DMAEN	DMA请求使能 (DMA requests enable) 0: 禁止DMA请求; 1: 使能 DMA 请求。
10	BUFININTEN	缓冲器中断使能 (Buffer interrupt enable) 0: 当 I2C_STS1.TXDATE=1 或 I2C_STS1.RXDATNE=1 时, 不产生任何中断; 1: 当 I2C_STS1.TXDATE=1 或 I2C_STS1.RXDATNE=1 时 (I2C_CTRL2.EVTINTEN=1), 产生事件中断 (不管 DMAEN 是何种状态)。
9	EVTINTEN	事件中断使能 (Event interrupt enable) 0: 禁止事件中断 1: 允许事件中断 在下列条件下, 将产生该中断: I2C_STS1.STARTBF= 1 (主模式) I2C_STS1.ADDR F= 1 (主/从模式) I2C_STS1.ADD10F= 1 (主模式) I2C_STS1.STOPF = 1 (从模式) I2C_STS1.BSF = 1, 但是没有I2C_STS1.TXDATE或I2C_STS1.RXDATNE事件 如果I2C_STS1.BUFININTEN= 1, I2C_STS1.TXDATE事件为1 如果 I2C_STS1.BUFININTEN= 1, I2C_STS1.RXDATNE 事件为 1
8	ERRINTEN	出错中断使能 (Error interrupt enable) 0: 禁止出错中断; 1: 允许出错中断。 在下列条件下, 将产生该中断: I2C_STS1.BUSERR= 1 I2C_STS1.ARLOST= 1 I2C_STS1.ACKFAIL= 1 I2C_STS1.OVERRUN= 1 I2C_STS1.PECERR= 1 I2C_STS1.TIMOUT= 1 I2C_STS1.SMBALERT= 1
7:6	Reserved	保留, 必须保持复位值
5:0	CLKFREQ[5:0]	I <sup>2</sup> C模块时钟频率 (Peripheral clock frequency) CLK FREQ[5:0] 应该为APB1 时钟频率以产生正确的时序: 000000: 禁用 000001: 禁用 000010: 2MHz 000011: 3MHz ...

位域	名称	描述
		100000: 32MHz 100001~111111: : 禁用。

#### 17.6.4 I<sup>2</sup>C 自身地址寄存器 1 (I2C\_OADDR1)

地址偏移: 0x08

复位值: 0x0000

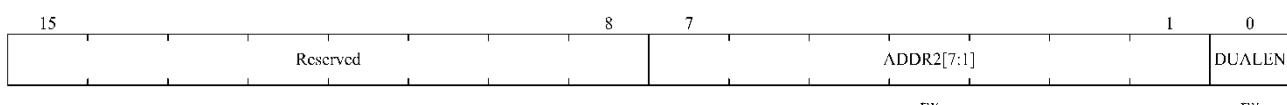


位域	名称	描述
15	ADDRMODE	寻址模式 (从模式) (Addressing mode (slave mode)) 0: 7位从地址 (不响应10位地址) ; 1: 10位从地址 (不响应7位地址)。
14	Reserved	必须始终由软件保持为‘1’。
13:10	Reserved	保留, 必须保持复位值
9:8	ADDR[9:8]	接口地址 (Interface address) 地址的9~8位。 注: 7位地址模式时不用关心
7:1	ADDR[7:1]	接口地址 (Interface address) 地址的7~1位。
0	ADDR0	接口地址 (Interface address) 地址第0位。 注: 7位地址模式时不用关心

#### 17.6.5 I<sup>2</sup>C 自身地址寄存器 2 (I2C\_OADDR2)

地址偏移: 0x0C

复位值: 0x0000



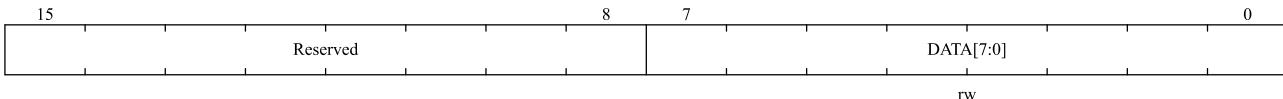
位域	名称	描述
15:8	Reserved	保留, 必须保持复位值
7:1	ADDR2[7:1]	接口地址 (Interface address) 在双地址模式下地址的7~1位。
0	DUALEN	双地址模式使能位 (Dual addressing mode enable) 0: 不使能双地址模式, 只有OADDR1被识别; 1: 使能双地址模式, OADDR1和OADDR2都被识别。

		注：仅7位地址模式有效
--	--	-------------

## 17.6.6 I<sup>2</sup>C 数据寄存器 (I2C\_DAT)

地址偏移: 0x10

复位值: 0x0000

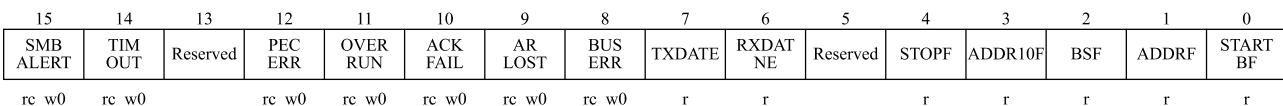


位域	名称	描述
15:8	Reserved	保留，必须保持复位值
7:0	DATA[7:0]	8位数据寄存器 (8-bit data register) 发送或接收数据 注意：从模式下。地址不会被拷贝到数据寄存器 注意：如果 I2C_STS1.TXDATE =0, 数据仍然会被写入数据寄存器 注意：如果在处理ACK时，发生了ARLOST事件，接收到的字节不会被拷贝到数据寄存器，因此不能读到它。

## 17.6.7 I<sup>2</sup>C 状态寄存器 1 (I2C\_STS1)

地址偏移: 0x14

复位值: 0x0000



位域	名称	描述
15	SMBALERT	SMBus提醒 (SMBus alert) 该位由软件写‘0’清除，或在I2C_CTRL1.EN =0时由硬件清除。 0：无SMBus提醒（主模式）或没有SMBAlert响应地址头序列（从模式）； 1：在引脚上产生 SMBAlert 提醒事件（主模式）或收到 SMBAlert 响应地址（从模式）；
14	TIMOUT	超时或Tlow错误 (Timeout or Tlow error) 该位由软件写‘0’清除，或在I2C_CTRL1.EN=0=0时由硬件清除。 0：无超时错误； 1：超时错误发生； 错误情形： <ul style="list-style-type: none"> <li>■ SCL 处于低已达到 25ms (Timeout);</li> <li>■ 主机的低电平累积时钟扩展时间超过 10ms (Tlow:mext);</li> <li>■ 从设备的低电平累积时钟扩展时间超过 25ms (Tlow:sext);</li> </ul> 从模式超时：从设备复位通讯，硬件释放总线。

位域	名称	描述
		主模式超时：硬件发出停止条件。
13	Reserved	保留，必须保持复位值
12	PECERR	<p>在接收时发生PEC错误（PEC Error in reception）      该位由软件写‘0’清除，或在I2C_CTRL1.EN =0时由硬件清除。      0: 无 PEC 错误: 接收到 PEC 后接收器返回 ACK (如果 I2C_CTRL1.ACKEN=1)      1: PEC 错误: 接收到 PEC 后接收器返回 NACK (不管 I2C_CTRL1.ACKEN 是否置位)</p>
11	OVERRUN	<p>过载/欠载 (Overrun/Underrun)      该位由软件写‘0’清除，或在I2C_CTRL1.EN=0时由硬件清除。      0: 无过载/欠载;      1: 出现过载/欠载。      当I2C_CTRL1.NOEXTEND=1时，在从模式下该位被硬件置位。同时：      在接收模式中当接收到一个新的字节时，如果数据寄存器里的内容还未被读出，则发生过载错误，新接收的字节将会丢失。      在发送模式中要发送一个新的字节时，却没有新的数据写入数寄存器，将发生欠载错误，同样的数据将会被发送两次。</p>
10	ACKFAIL	<p>应答失败 (Acknowledge failure)      该位由软件写‘0’清除，或在I2C_CTRL1.EN=0时由硬件清除。      0: 没有应答失败;      1: 应答失败。</p>
9	ARLOST	<p>仲裁丢失 (仲裁模式)      该位由软件写‘0’清除，或在I2C_CTRL1.EN=0时由硬件清除。      0: 没有仲裁丢失;      1: 仲裁丢失。      当接口失去对总线的控制时，硬件将置该位为‘1’。在ARLOST事件之后，I<sup>2</sup>C接口自动切换回从模式(I2C_STS2.MSMODE=0)。      注：在 SMBUS 模式下，在从模式下对数据的仲裁仅仅发生在数据阶段，或应答传输区间（不包括地址的应答）。</p>
8	BUSERR	<p>总线错误 (Bus error)      该位由软件写‘0’清除，或在I2C_CTRL1.EN=0时由硬件清除。      0: 无起始或停止条件出错;      1: 起始或停止条件出错。</p>
7	TXDATE	<p>数据寄存器为空 (发送时)      软件写数据到DAT寄存器可清除该位；或在发生一个起始或停止条件后，或当I2C_CTRL1.EN=0时由硬件自动清除。      0: 数据寄存器非空;      1: 数据寄存器空。      在发送数据时，数据寄存器为空时该位被置‘1’，在发送地址阶段不设置该位。      如果收到一个NACK，或下一个要发送的字节是PEC (I2C_CTRL1.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BSF时写入数据，都不能清除TXDATE位，这是因为数据寄存器仍然为空。</p>
6	RXDATNE	数据寄存器非空 (接收时)

位域	名称	描述
		<p>软件对数据寄存器的读写操作清除该位，或当I2C_CTRL1.EN=0时由硬件清除。</p> <p>0: 数据寄存器为空； 1: 数据寄存器非空。</p> <p>在接收时，当数据寄存器不为空，该位被置‘1’。在接收地址阶段，该位不被置位。</p> <p>在发生ARLOST事件时，RXDATNE不被置位。</p> <p><i>注：当设置了BSF时，读取数据不能清除RXDATNE位，因为数据寄存器仍然为满。</i></p>
5	Reserved	保留，必须保持复位值
4	STOPF	<p>停止条件检测位（从模式）</p> <p>软件读取STS1寄存器后，对I2C_CTRL1寄存器的写操作将清除该位，或当I2C_CTRL1.EN=0时，硬件清除该位。</p> <p>0: 没有检测到停止条件； 1: 检测到停止条件。</p> <p>在一个应答之后，当从设备在总线上检测到停止条件时，硬件将该位置‘1’。</p> <p><i>注：在收到NACK后，I2C_STS1.STOPF位不被置位。</i></p>
3	ADDR10F	<p>10位头序列已发送（主模式）</p> <p>软件读取STS1寄存器后，对CTRL1寄存器的写操作将清除该位，或当I2C_CTRL1.EN=0时，硬件清除该位。</p> <p>0: 没有ADDR10F事件发生； 1: 主设备已经将第一个地址字节发送出去。</p> <p>在10位地址模式下，当主设备已经将第一个地址字节发送出去时，硬件将该位置‘1’。</p> <p><i>注：收到一个NACK后，I2C_STS1.ADDR10F位不被置位。</i></p>
2	BSF	<p>字节传输结束（Byte transfer finished）</p> <p>在软件读取STS1寄存器后，对数据寄存器的读或写操作将清除该位；或在传输中发送一个起始或停止条件后，或当I2C_CTRL1.EN =0时，由硬件清除该位。</p> <p>0: 字节传输未完成； 1: 字节传输结束。</p> <p>当I2C_CTRL1.NOEXTEND =0时，在下列情况下硬件将该位置‘1’：</p> <p>在接收时，当收到一个新字节（包括ACK脉冲）且数据寄存器还未被读取（I2C_STS1.RXDATNE =1）。</p> <p>在发送时，当一个新数据将被发送且数据寄存器还未被写入新的数据（I2C_STS1.TXDATE =1）。</p> <p><i>注：在收到一个NACK后，BSF位不会被置位。</i></p> <p>如果下一个要传输的字节是PEC（I2C_STS2.TRF为‘1’，同时I2C_CTRL1.PEC为‘1’），BSF位不会被置位。</p>
1	ADDRF	<p>地址已被发送（主模式）/地址匹配（从模式）</p> <p>在软件读取STS1寄存器后，对STS2寄存器的读操作将清除该位，或当I2C_CTRL1.EN=0时，由硬件清除该位。</p> <p>0: 地址不匹配或没有收到地址（从模式），地址发送未完成（主模式）； 1: 收到的地址匹配（从模式），地址发送完成（主模式）。</p> <p>在主模式下：</p> <p>7位地址模式时，当收到地址的ACK后该位被置‘1’，10位地址模式时，当收到地址的第二个字节的ACK后该位被置‘1’。</p>

位域	名称	描述
		<p>在从模式下：</p> <p>当收到的从地址与OADDR寄存器中的内容相匹配、或广播呼叫或SMBus设备默认地址或SMBus主机或SMBus提醒被识别时，硬件就将该位置‘1’（当对应的设置被使能时）。</p> <p>注：在收到NACK后，I2C_STS1.ADDRF位不会被置位。</p>
0	STARTBF	<p>起始位（主模式）</p> <p>软件读取I2C_STS1寄存器后，写数据寄存器的操作将清除该位，或当I2C_CTRL1.EN=0时，硬件清除该位。</p> <p>0：未发送起始条件； 1：起始条件已发送。</p> <p>当发出起始条件时该位被置‘1’。</p>

## 17.6.8 I<sup>2</sup>C 状态寄存器 2 (I2C\_STS2)

地址偏移：0x18

复位值：0x0000

15	PECVAL[7:0]	8	7	6	5	4	3	2	1	0
	r		r	r	r	r	Reserved	TRF	r	r

位域	名称	描述
15:8	PECVAL[7:0]	<p>数据包出错检测 (Packet error checking register)</p> <p>当 I2C_CTRL1.PECEN=1 时，PECVAL[7:0]存放内部的 PEC 值。</p>
7	DUALFLAG	<p>双标志 (从模式)</p> <p>在产生一个停止条件或一个重复的起始条件时，或I2C_CTRL1.EN =0时，硬件将该位清除。</p> <p>0：接收到的地址与OADDR1内的内容相匹配； 1：接收到的地址与 OADDR2 内的内容相匹配。</p>
6	SMBHADDR	<p>SMBus主机头地址 (从模式)</p> <p>在产生一个停止条件或一个重复的起始条件时，或I2C_CTRL1.EN =0时，硬件将该位清除。</p> <p>0：未收到SMBus主机的地址； 1：当 I2C_CTRL1.SMBTYPE=1 且 I2C_CTRL1.AR PEN=1 时，收到 SMBus 主机地址。</p>
5	SMBDADDR	<p>SMBus 设备默认地址 (从模式)</p> <p>在产生一个停止条件或一个重复的起始条件时，或I2C_CTRL1.EN=0时，硬件将该位清除。</p> <p>0：未收到SMBus设备的默认地址； 1：当 I2C_CTRL1.AR PEN=1 时，收到 SMBus 设备的默认地址。</p>
4	GCALLADDR	<p>广播呼叫地址 (从模式)</p> <p>在产生一个停止条件或一个重复的起始条件时，或I2C_CTRL1.EN=0时，硬件将该位清除。</p> <p>0：未收到广播呼叫地址； 1：当 I2C_CTRL1.GCEN =1 时，收到广播呼叫的地址。</p>

位域	名称	描述
3	Reserved	保留，必须保持复位值
2	TRF	发送/接收（Transmitter/receiver） 在检测到停止条件（I2C_STS1.STOPF =1）、重复的起始条件或总线仲裁丢失（I2C_STS1.ARLOST =1）后，或当I2C_CTRL1.EN=0时，硬件将其清除。 0：数据接收模式； 1：数据发送模式； 在整个地址传输阶段的结尾，该位根据地址字节的 R/W 位来设定。
1	BUSY	总线忙（Bus busy） 当检测到一个停止条件时，硬件将该位清除。 0：在总线上无数据通讯； 1：在总线上正在进行数据通讯。 在检测到SDA或SCL为低电平时，硬件将该位置‘1’； 注：该位指示当前正在进行的总线通讯，当接口被禁用（I2C_CTRL1.EN =0）时该信息仍然被更新。
0	MSMODE	主从模式（Master/slave） 当总线上检测到一个停止条件、仲裁丢失（I2C_STS1.ARLOST =1）时、或当I2C_CTRL1.EN =0时，硬件清除该位。 0：从模式； 1：主模式。 当接口处于主模式（I2C_CTRL1.STARTBF=1）时，硬件将该位置位；

### 17.6.9 时钟控制寄存器(I2C\_CLKCTRL)

地址偏移: 0x1C

复位值: 0x0000

注: 1. 要求  $F_{PCLK1}$  应当是  $10\text{MHz}$  的整数倍, 这样可以正确地产生  $400\text{kHz}$  的快速时钟。

2. CLKCTRL 寄存器只有在关闭 I<sup>2</sup>C 时 (*I2C\_CTRL1.EN*=0) 才能设置。

位域	名称	描述
15	FSMODE	I2C 主模式选择 0: I2C 标准模式(占空比默认 1/1); 1: I2C 快速模式(占空比可配置).
14	DUTY	快速模式占空比 (Duty cycle in fast mode) 0: $T_{low}/T_{high} = 2$ ; 1: $T_{low}/T_{high} = 16/9$
13:12	Reserved	保留, 必须保持复位值
11:0	CLKCTRL[11:0]	快速/ 标准模式下的时钟控制分频系数 (主模式)

位域	名称	描述
		<p>该分频系数用于设置主模式下的SCL时钟。</p> <ul style="list-style-type: none"> <li>■ 如果 duty cycle = Tlow/Thigh = 1/1:  <math>CLKCTRL = f_{PCLK1}(\text{Hz})/100000/2</math>  <math>T_{low} = CLKCTRL \times T_{PCLK1}</math>  <math>T_{high} = CLKCTRL \times T_{PCLK1}</math> </li> <li>■ 如果 duty cycle = Tlow/Thigh = 2/1:  <math>CLKCTRL = f_{PCLK1}(\text{Hz})/100000/3</math>  <math>T_{low} = 2 \times CLKCTRL \times T_{PCLK1}</math>  <math>T_{high} = CLKCTRL \times T_{PCLK1}</math> </li> <li>■ 如果 duty cycle = Tlow/Thigh = 16/9:  <math>CLKCTRL = f_{PCLK1}(\text{Hz})/100000/25</math>  <math>T_{low} = 16 \times CLKCTRL \times T_{PCLK1}</math>  <math>T_{high} = 9 \times CLKCTRL \times T_{PCLK1}</math> </li> </ul> <p>例如, 如果 <math>f_{PCLK1}(\text{Hz}) = 8\text{MHz}</math>, duty cycle = 1/1, <math>CLKCTRL = 8000000/100000/2 = 0x28</math>。  注意: 1. 允许设定的最小值为 0x04, 在快速 DUTY 模式下允许的最小值为 0x01。  2. <math>T_{high} = T_{f(SCL)} + T_{w(SCLH)}</math>, 详见数据手册中对这些参数的定义。  3. <math>T_{low} = T_{f(SCL)} + T_{w(SCLL)}</math>, 详见数据手册中对这些参数的定义。</p>

### 17.6.10 上升时间寄存器 (I2C\_TMRISE)

地址偏移: 0x20

复位值: 0x0002

注意: I2C\_TMRISE 寄存器仅在主模式下有效, 当 I2C 禁用 (I2C\_CTRL1.EN=0) 时才能配置。

31	30	29	28	27	26	25	22	21	18	17	16
SCLAF ENN	SCLAFW[1:0]	SDAAF ENN	SDAAFW[1:0]		SCLDFW[3:0]		SDADFW[3:0]		Reserved		
rw 15	rw	rw	rw	rw	rw	rw	rw 6	rw 5	rw 0		
							Reserved		TMRISE[5:0]		

位域	名称	描述
31	SCLAFENN	SCL 模拟滤波器使能 0: 使能 1: 禁用
30:29	SCLAFW[1:0]	SCL 模拟滤波器宽度选择 00: 5ns 01: 15ns 10: 25ns 11: 35ns  注意: 模拟滤波器的变化范围很大, 这取决于不同 PVT 设置

位域	名称	描述
28	SDAAFENN	SDA 模拟滤波器 0: 使能 1: 禁用
27:26	SDAAFW[1:0]	SDA 模拟滤波器宽度选择: 00: 5ns 01: 15ns 10: 25ns 11: 35ns 注意: 模拟滤波器的变化范围很大, 这取决于不同 PVT 设置
25:22	SCLDFW[3:0]	SCL 数字滤波器宽度选择: 0000: 禁用数字滤波器 其他值: pclk 周期数
21:18	SDADFW[3:0]	SDA 数字滤波器宽度选择: 0000: 禁用数字滤波器 其他值: pclk 周期数
17:6	Reserved	保留, 必须保持复位值
5:0	TMRISE[5:0]	在快速/ 标准模式下的最大上升时间 (主模式) 这些位必须设置为I <sup>2</sup> C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_CTRL2.CLKFREQ[5:0]中的值等于0x08且TpCLK1=125ns, 故TMRISE[5:0]中必须写入09h (1000ns/125 ns + 1)。 如果结果不是一个整数, 则将整数部分写入 TMRISE[5:0]以确保 tHIGH 参数。 数字滤波器的值需要被加到 TMRISE[5:0]中。

## 18 通用同步异步收发器 (USART)

### 18.1 简介

通用同步异步收发器 (USART) 是一种全双工串行数据交换接口，支持同步或异步通信。可灵活配置，以便于与多种外部设备进行全双工数据交换。

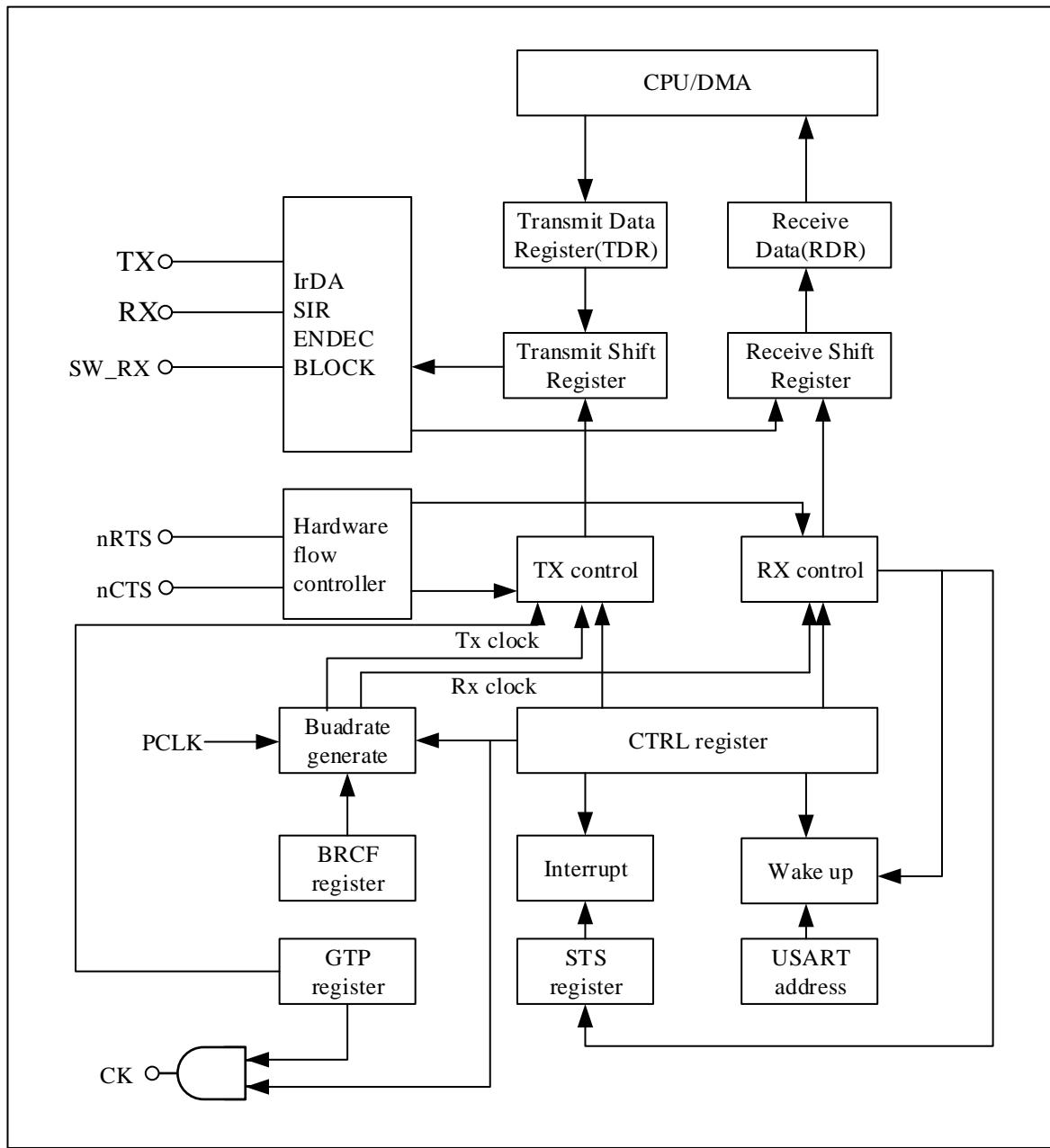
USART 接口发送与接收波特率可配置，也支持通过 DMA 进行连续通信。USART 还支持多处理器通信、LIN 模式、同步模式、单线半双工通信、智能卡异步协议、IrDA SIR ENDEC 功能、以及硬件流控制功能。

### 18.2 主要特性

- 支持全双工通信
- 支持单线半双工通信
- 波特率可配置，最高波特率可达 4Mbit/s
- 支持 8bit 或 9bit 数据帧
- 支持 1bit 或 2bit 停止位
- 支持硬件生成校验位及校验位检查
- 支持硬件流控: RTS、CTS
- 支持 DMA 收发
- 支持多处理器通信：如果地址不匹配，则进入静默模式，可通过空闲总线检测或地址标识唤醒
- 支持同步模式，允许用户在主模式下控制双向同步串行通信
- 支持智能卡异步协议，符合 ISO7816-3 标准
- 支持串行红外协议（IrDA SIR）编码与解码，提供正常与低功耗两种运行模式
- 支持 LIN 模式
- 支持多钟错误检测：数据溢出错误、帧错误、噪声错误、检验错误
- 支持多个中断请求：发送数据寄存器为空、CTS 标志、发送完成、数据已接收、数据溢出、总线空闲、检验错误、LIN 模式断开帧检测、以及多缓冲区通信中的噪声标志/溢出错误/帧错误

## 18.3 功能框图

图 18-1 USART 框图



## 18.4 功能描述

如图 18-1 所示, USART 的双向通信都需要使用 RX 和 TX 引脚与外部器件连接。其中 TX 为数据发送引脚 (输出), 当发送功能使能但没有发送数据时, TX 引脚输出高电平, 当发送功能被禁用时, TX 引脚为普通 IO 端口, 状态由应 IO 配置决定。RX 为数据接收引脚 (输入), 接收数据时采用了过采样技术。

当设备作为发送端时, 通过 TX 引脚发送数据, 作为接收端时则通过 RX 引脚接收数据。当没有数据收

发时，总线处于空闲状态。通用数据帧格式为：1个起始位+8或9位数据（最低有效位在前）+1个检验位（可选）+0.5,1,1.5或2个停止位。USART 使用分数波特率发生器来配置发送与接收波特率。

从功能框图上可以看出，使用硬件流控功能时，需要 nRTS 输出引脚和 nCTS 输入引脚。当 USART 作为接收端时，如果已准备好接收数据，nRTS 输出低电平。当 USART 作为发送端时，nCTS 有效（低电平）才发送下一个数据，nCTS 无效（高电平）则不发送数据。

当使用同步模式时需要用到 CK 引脚，用于同步传输时钟输出，时钟极性与相位可软件配置。但在发送起始位与停止位时，CK 引脚不输出同步时钟。在智能卡模式下也需要 CK 引脚提供时钟。

### 18.4.1 USART 帧格式

起始位：1位，低电平有效

数据位：可通过 USART\_CTRL1.WL 配置为 8 或 9 位，最低有效位在前。

停止位：高电平有效。

空闲帧：全部由‘1’组成的一个完整的数据帧，包括起始位。

断开帧：全部由‘0’组成的一个完整的数据帧，包括停止位。在断开帧结束后，发送端再插入1或2个停止位来应答起始位。

图 18-2 字长=8 设置

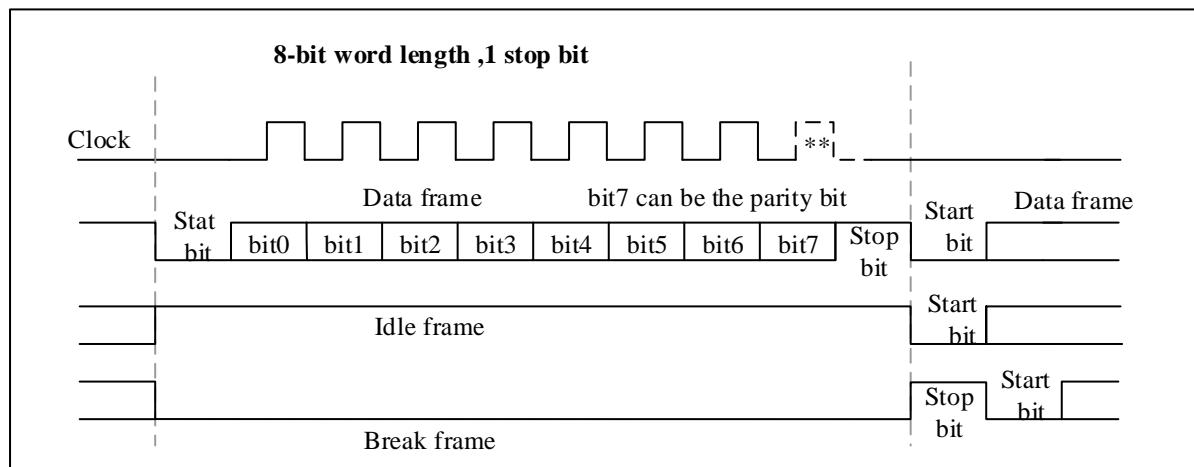
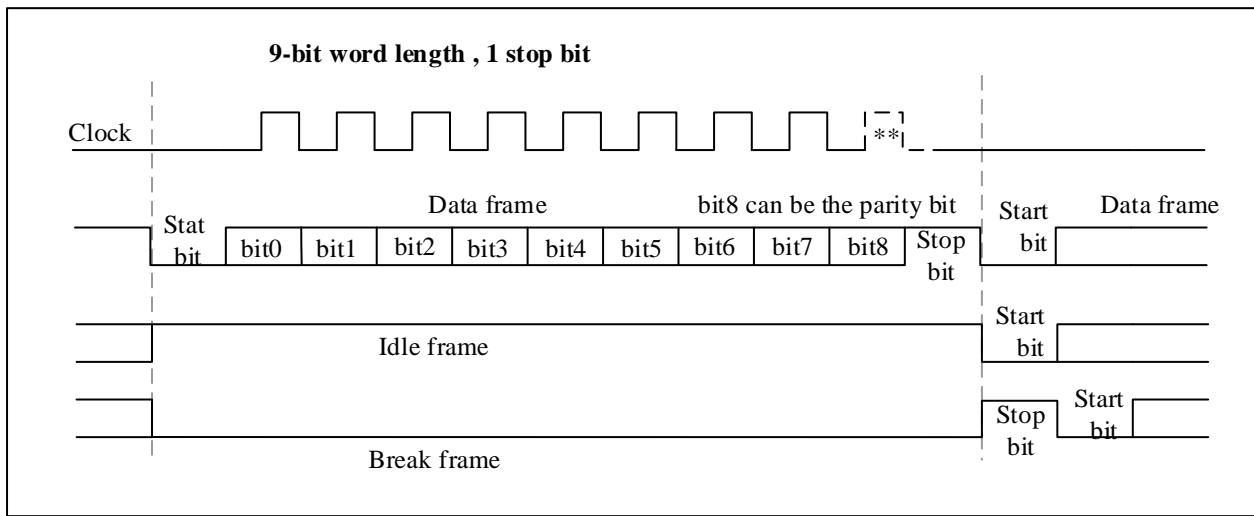


图 18-3 字长=9 设置



## 18.4.2 发送器

当发送功能使能后，进入移位寄存器中的数据通过 TX 引脚输出。.

### 18.4.2.1 空闲帧

USART\_CTRL1.TXEN 置 1 后，USART 会在发送数据之前发送一个空闲帧。

### 18.4.2.2 字符发送

在空闲帧结束后，字符可正常发送。在每个字符发送前，先发送一个起始位（低电平）。发送器根据数据长度配置发送 8 位或 9 位数据，其中最低有效位先发送。如果在数据传输时 USART\_CTRL1.TXEN 被清零，将导致波特率计数器停止计数，从而破坏正在传输的数据。

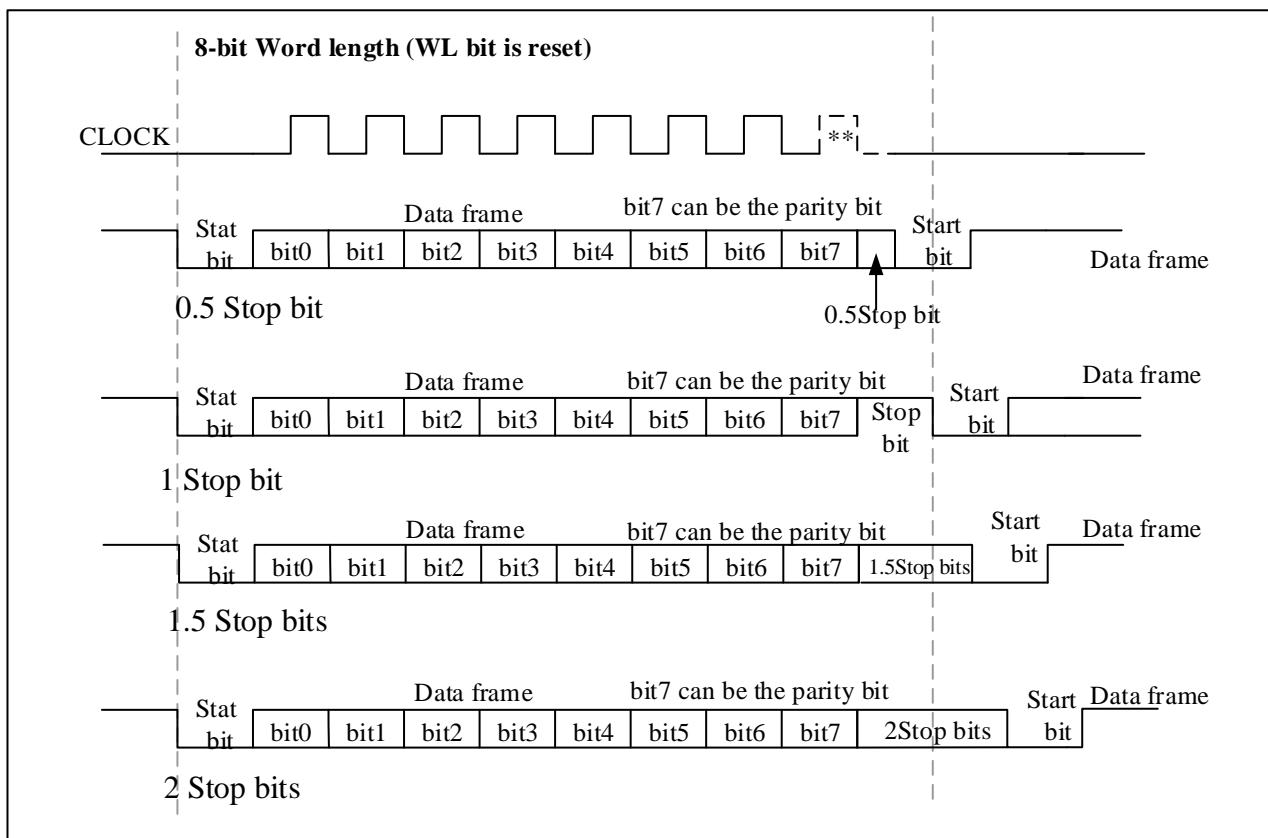
### 18.4.2.3 停止位

字符发送完成后，发送器自动发送停止位。停止位位数可通过 USART\_CTRL2.STPB[1:0]配置。

表 18-1 停止位配置

USART_CTRL2.STPB[1:0]	停止位长度 (位)	功能描述
00	1	默认
01	0.5	用于智能卡模式下数据接收
10	2	用于常规 USART 模式、单线模式以及调制解调器模式。
11	1.5	用于智能卡模式下数据发送和接收

图 18-4 停止位配置



#### 18.4.2.4 断开帧

可通过置位 USART\_CTRL1.SDBRK 来发送 1 个断开帧。当数据长度为 8 位时，断开帧由 10 位低电平组成，当数据长度为 9 位时，断开帧由 11 位低电平组成。断开帧结束后将插入一位停止位（高电平）。

断开帧发送完成后，USART\_CTRL1.SDBRK 被硬件清零，同时自动发送停止位。因此，如果要连续发送断开帧，必须在前一个断开帧与停止位发送完成后再次置位 USART\_CTRL1.SDBRK。

如果在断开帧开始发送前软件清零 USART\_CTRL1.SDBRK，当前断开帧不会发送。.

#### 18.4.2.5 发送流程

1. 置位 USART\_CTRL1.UEN 来使能 USART；
2. 配置波特率、数据长度、校验位、停止位长度、以及根据需要配置相关 DMA；
3. 使能发送功能 (USART\_CTRL1.TXEN)；
4. 通过软件或 DMA 将要发送的数据依次写入数据寄存器 USART\_DAT，当数据写入数据寄存器时将清零 USART\_STS.TXDE；

5. 当所有数据已写入到数据寄存器 USART\_DAT 后，等待发送完成标志位 USART\_STS.TXC 置 1，数据发送完成。

#### 18.4.2.6 单字节通信

对数据寄存器 USART\_DAT 的写操作将清零标志位 USART\_STS.TXDE。

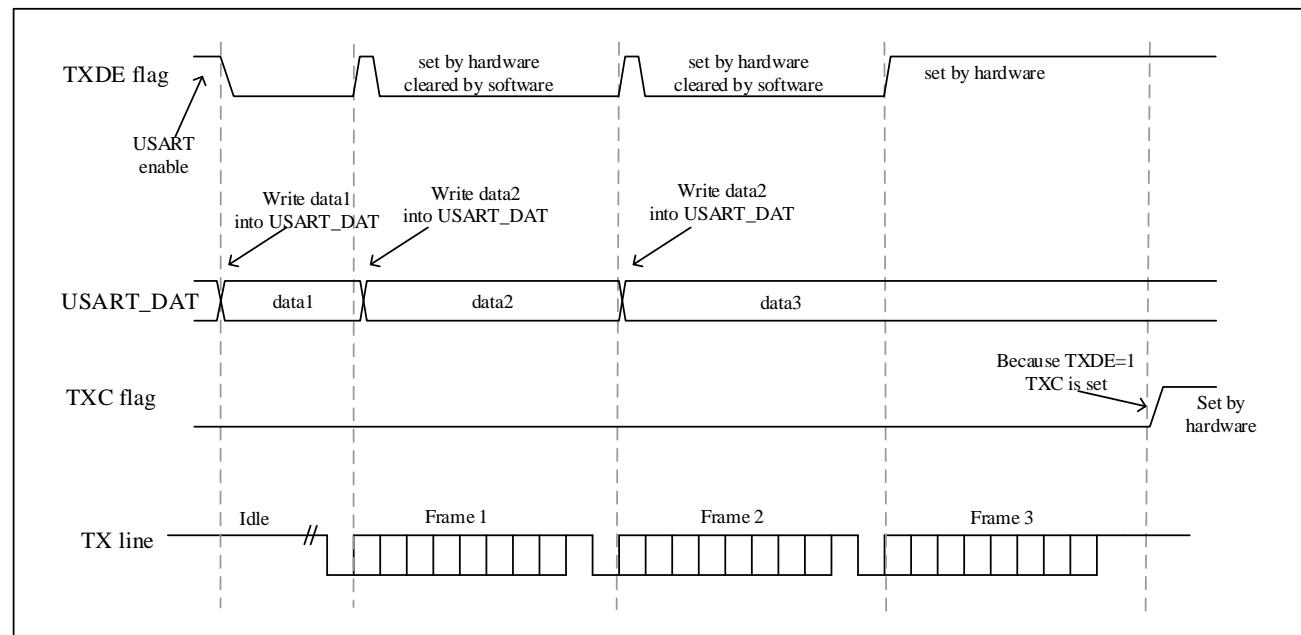
当数据已从发送数据寄存器送到移位寄存器时，USART\_STS.TXDE 位由硬件置 1，表示数据开始发送。如果 USART\_CTRL1.TXDEIEN 已置 1，将产生一个中断。此时，可将下一个数据写入数据寄存器 USART\_DAT。

对数据寄存器 USART\_DAT 进行写操作时：

- 如果移位寄存器空闲，数据将直接送到移位寄存器，同时 USART\_STS.TXDE 硬件置 1
- 如果移位寄存器正在发送数据，数据保存在数据寄存器，待上一个数据发送完成后，再送到移位寄存器

当一帧数据发送完成后并且 USART\_STS.TXDE 置 1，USART\_STS.TXC 被硬件置 1。如果 USART\_CTRL1.TXCIEN 已置 1，将产生一个中断。USART\_STS.TXC 通过以下软件操作清零：先读一次 USART\_STS 寄存器，再写一次 USART\_DAT 寄存器。

图 18-5 发送时 TXC/TXDE 的变化情况



#### 18.4.3 接收器

##### 18.4.3.1 起始位检测

在 USART 中，如果识别到一个特殊的采样序列 1 1 1 0 X 0 X 0 X 0 0 0 0，就认为检测到一个起始位。

在第 3、5、7 位的采样，以及在第 8、9、10 位的采样都为'0'（也即 6 个'0'），则确认收到起始位，并将 USART\_STS.RXDNE 置 1，但不会置位 NEF 噪声标志。如果 USART\_CTRL1.RXDNEIEN 已置 1，则产生一个中断。

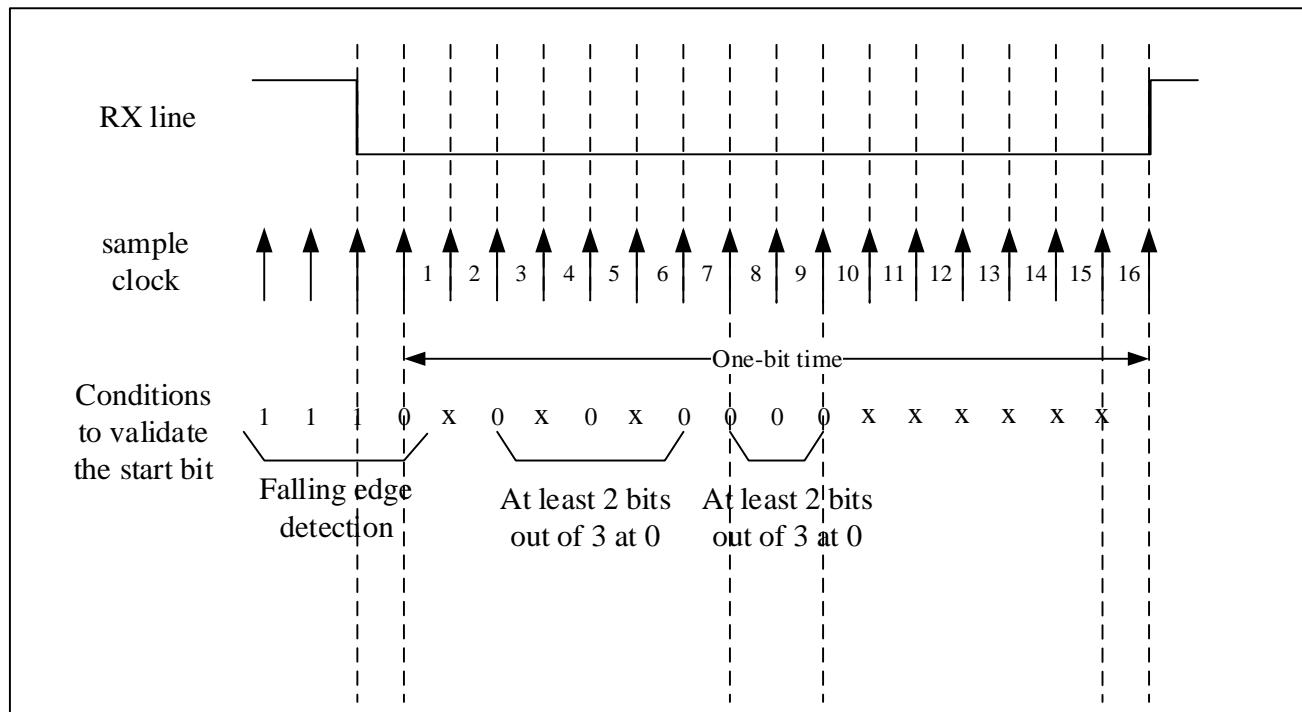
第 3、5、7 位的采样有两个'0'，与此同时，第 8、9、10 位的采样有两个'0'点，也确认收到起始位，但是会置位 NEF 噪声标志位。

第 3、5、7 位的采样有三个'0'，与此同时，第 8、9、10 位的采样有两个'0'点，也确认收到起始位，并置位 NEF 噪声标志位。

第 3、5、7 位的采样有两个'0'，与此同时，第 8、9、10 位的采样有三个'0'点，也确认收到起始位，并置位 NEF 噪声标志位。

如果在第 3、5、7、8、9、10 位的采样值满足不了上面四种要求，USART 接收器认为没有接受到正确的起始位，将退出起始位侦测并回到空闲状态等待下降沿。

图 18-6 起始位检测



#### 18.4.3.2 停止位

停止位长度可通过 USART\_CTRL2.STPB[1:0]配置。常规模式下，可配置为 1 位或 2 位。智能卡模式下，可配置为 0.5 位或 1.5 位。

1. 0.5 位停止位（智能卡模式中的数据接收）：不对停止位进行采样。因此，此时不能检测帧错误和断开帧。
2. 1 个停止位：默认情况下通过三个点对 1 个停止位的采样，选择第 8, 第 9 和第 10 采样位上进行。
3. 1.5 个停止位（智能卡模式）：智能卡模式下发送数据时，器件必须检查数据是否被正确的发送出去。所以接收器功能块必须被激活（USART\_CTRL2.RXEN=1），并且在停止位的发送期间采样数据线上的信号。如果出现校验错误智能卡会在发送方采样 NACK 信号时拉低数据线，表示出现了帧错误。USART\_STS.FEF 与 USART\_STS.RXDNE 在停止位结束后被置 1。对 1.5 个停止位的采样是在第 16, 第 17 和第 18 采样点进行的，可分成两个部分：0.5 个数据位周期，接收器不做任何处理；然后是 1 个数据位周期，接收器对其进行采样。参照图 18.4.14 智能卡模式。
4. 2 个停止位：对 2 个停止位的采样是在第一停止位的第 8, 第 9 和第 10 个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被置起。在第一个停止位结束时 USART\_STS.RXDNE 标志将被设置。第二个停止位将不会检测帧错误。

#### 18.4.3.3 接收流程

1. 将 USART\_CTRL1.UEN 置 1 来使能 USART；
2. 配置波特率、数据长度、校验位、停止位长度、以及根据需要配置相关 DMA；
3. 使能接收器 (USART\_CTRL1.RXEN)，开始起始位检测；
4. 接收 8 位或 9 位数据，通过 RX 引脚送往接收移位寄存器，最低有效位在前；
5. 当数据由接收移位寄存器送到 RDR 寄存器，USART\_STS.RXDNE 被置 1，表示数据可以被读出。如果 USART\_CTRL2.RXDNEIEN 已置 1，将产生一个中断；
6. 当接收过程中检测到帧错误、噪音或溢出错误，这样错误标志将被置 1。如果在数据传输过程中 USART\_CTRL1.RXEN 被清零，当前接收数据丢失；
7. USART\_STS.RXDNE 通过对 USART\_DAT 寄存器进行读操作清零：
  - 在多缓冲器通信模式，USART\_STS.RXDNE 通过 DMA 对数据寄存器的读操作清零。
  - 在单缓冲器通信模式，USART\_STS.RXDNE 通过软件对数据寄存器的读操作清零。

#### 18.4.3.4 空闲帧检测

当一空闲帧被检测到时，USART\_STS.IDLEF 置 1。此时如果 USART\_CTRL1.IDLEIEN 已置 1，将产生一个中断。USART\_STS.IDLEF 可通过以下软件操作清零：先读 USART\_STS 寄存器，再读 USART\_DAT 寄存器。

### 18.4.3.5 断开帧检测

当一断开帧被检测到时，帧错误标志 USART\_STS.FEF 被硬件置 1，可通过以下软件操作清零：先读 USART\_STS 寄存器，再读 USART\_DAT 寄存器。

### 18.4.3.6 帧错误

如果在预期的时间内没有接收和识别到停止位，产生一个帧错误，标志位 USART\_STS.FEF 置 1，同时无效数据将从移位寄存器送到 USART\_DAT 寄存器。在单字节通信时，没有帧错误中断产生，因为此时 USART\_STS.RXDNE 位同时置 1，后者将产生中断。在多缓冲器通信情况下，如果 USART\_CTRL3.ERRIEN 已置 1，将产生一个中断。

### 18.4.3.7 溢出错误

如果 USART\_STS.RXDNE 已被置 1，而接收移位寄存器又有数据需要送入数据寄存器，则发生溢出错误，同时标志位 USART\_STS.OREF 硬件置 1。此时数据寄存器中的数据不会丢失，但移位寄存器中的数据将被覆盖。USART\_STS.OREF 可通过以下软件操作清零：先读 USART\_STS 寄存器，再读 USART\_DAT 寄存器。

当产生溢出错误时，因 USART\_STS.RXDNE 已置 1，将产生一个接收中断。多缓冲器通信模式下，如果 USART\_CTRL3.ERRIEN 已置 1，将产生一个错误中断。

### 18.4.3.8 噪声错误

当接收器检测到噪声错误时，USART\_STS.NEF 被置 1，可通过以下软件操作清零：先读 USART\_STS 寄存器，再读 USART\_DAT 寄存器。在单字节通信模式下会产生错误中断，因为此时 USART\_STS.RXDNE 也被置 1 并产生接收中断。在多缓冲器通信模式，如果 USART\_CTRL3.ERRIEN 已置 1，将产生一个错误中断。

表 18-2 噪声检测的数据采样

采样值	NE 状态	接收的位	数据有效性
000	0	0	有效
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效

111	0	1	有效
-----	---	---	----

## 18.4.4 分数波特率计算

波特率通过 USART\_BRCF 寄存器配置，分频系数由整数部分和小数部分组成，同时适用于发送器与接收器。在写入 USART\_BRCF 之后，波特率计数器会被波特率寄存器的新值替换。因此，不要在通信过程中改变波特率寄存器的数值。

$$\text{TX / RX 波特率} = f_{CK} / (16 * \text{USARTDIV})$$

其中  $f_{CK}$  为 USART 外设时钟：

- PCLK1 用于 USART2，最高 32MHz
- PCLK2 用于 USART1, UART3, UART4, 最高 64 MHz.

USARTDIV 为无符号分频系数

### 18.4.4.1 分频系数 USARTDIV 与 USART\_BRCF 寄存器配置

示例 1:

如果 USARTDIV = 27.75，则：

$$\text{DIV\_Decimal} = 16 * 0.75 = 12 = 0x0C$$

$$\text{DIV\_Integer} = 27 = 0x1B$$

$$\text{因此 USART\_BRCF} = 0x1BC$$

示例 2:

如果 USARTDIV = 20.98，则：

$$\text{DIV\_Decimal} = 16 * 0.98 = 15.68$$

取最接近的整数 DIV\_Decimal = 16 = 0x10，超出可配置范围，因此需要向整数位进位

$$\text{从而 DIV\_Integer} = 20 + 1 = 21 = 0x15$$

$$\text{DIV\_Decimal} = 0x0$$

$$\text{因此 USART\_BRCF} = 0x150$$

示例 3:

如果 USART\_BRCF = 0x19B：

$$\text{DIV\_Integer} = 0x19 = 25$$

$$\text{DIV\_Decimal} = 0x0B = 11$$

$$\text{USARTDIV} = 25 + 11 / 16 = 25.6875$$

表 18-3 设置波特率时的误差计算

波特率		$f_{PCLK}=32MHz$			$f_{PCLK}=64MHz$		
序号	Kbps	实际	寄存器设置值	误差%	实际	寄存器设置值	误差%
1	2.4	2.400	833.333	0%	2.4	1666.667	0%
2	9.6	9.600	208.333	0.02%	9.6	416.667	0%
3	19.2	19.2	104.167	0.02%	19.2	208.333	0.02%
4	57.6	57.6	34.722	0.05%	57.6	69.444	0.05%
5	115.2	115.384	17.361	0.16%	115.2	34.722	0.05%
6	230.4	230.769	8.681	0.16%	230.769	17.361	0%
7	460.8	461.538	3.6875	0.69%	461.538	8.681	0%
8	921.6	923.076	2.170	1%	923.076	4.340	0.8%
9	1687.5	1687.7	1.185	0%	2250	2.370	0%
10	3375	不可能	不可能	不可能	3375	1.185	0%

注意: CPU 的时钟频率越低, 则某一特定波特率的误差也越低。

#### 18.4.5 USART 接收器容忍时钟的变化

应用中可能会出现发送误差(包括发射端时钟的变化)、接收端波特率误差及时钟变化、传输线变化(通常由数据上升沿和下降沿时序不一致引起)。这些因素都会影响整个时钟系统的变化。只有当上述四个变化之和小于 USART 接收机的容差时, USART 异步接收机才能正常工作。

正常接收数据时, USART 接收器的容忍度为最大能容忍的变化, 取决于数据位长度的选择, 以及是否使用分数波特率分频系数。

表 18-4 当 DIV\_Decimal =0 时, USART 接收器的容忍度

WL 位	认为 NF 是错误	不认为 NF 是错误
0	3.75%	4.375%
1	3.41%	3.97%

表 18-5 当 DIV\_Decimal !=0 时, USART 接收器的容忍度

WL 位	认为 NF 是错误	不认为 NF 是错误
0	3.33%	3.88%
1	3.03%	3.53%

## 18.4.6 校验控制

通过设置 USART\_CTRL1.PCEN 来使能校验功能。

使能后, 在发送数据时自动生成并发送校验位, 接收数据时对校验位进行检查。

表 18-6 帧格式

WL 位	PCEN 位	USART 帧
0	0	起始位   8 位数据   停止位
0	1	起始位   7 位数据   奇偶校验位   停止位
1	0	起始位   9 位数据   停止位
1	1	起始位   8 位数据   奇偶校验位   停止位

### 偶校验

USART\_CTRL1.PSEL 设置为 0, 使能偶校验

偶校验表示一帧数据（包括校验位）中'1'的个数为偶数。例如：数据=11000101, 有 4 个'1', 则发送端偶校验位为'0'（总共 4 个'1'）。接收端对数据中'1'个数进行确认：如果是偶数，校验通过；如果是奇数，表示产生了校验错误，USART\_STS.PEF 标志位置 1，此时如果 USART\_CTRL1.PEIVEN 已置 1，产生一个中断。

### 奇校验

USART\_CTRL1.PSEL 设置为 1, 使能奇校验

奇校验表示一帧数据（包括校验位）中'1'的个数为奇数。例如：数据=11000101, 有 4 个'1', 则发送端奇校验位为'1'（总共 5 个'1'）。接收端对数据中'1'个数进行确认：如果是奇数，校验通过；如果是偶数，表示产生了校验错误，USART\_STS.PEF 标志位置 1，此时如果 USART\_CTRL1.PEIVEN 已置 1，产生一个中断。

## 18.4.7 DMA 通信

USART 支持 DMA 通信，此时采用多缓冲模式可达到较高的通信效率。

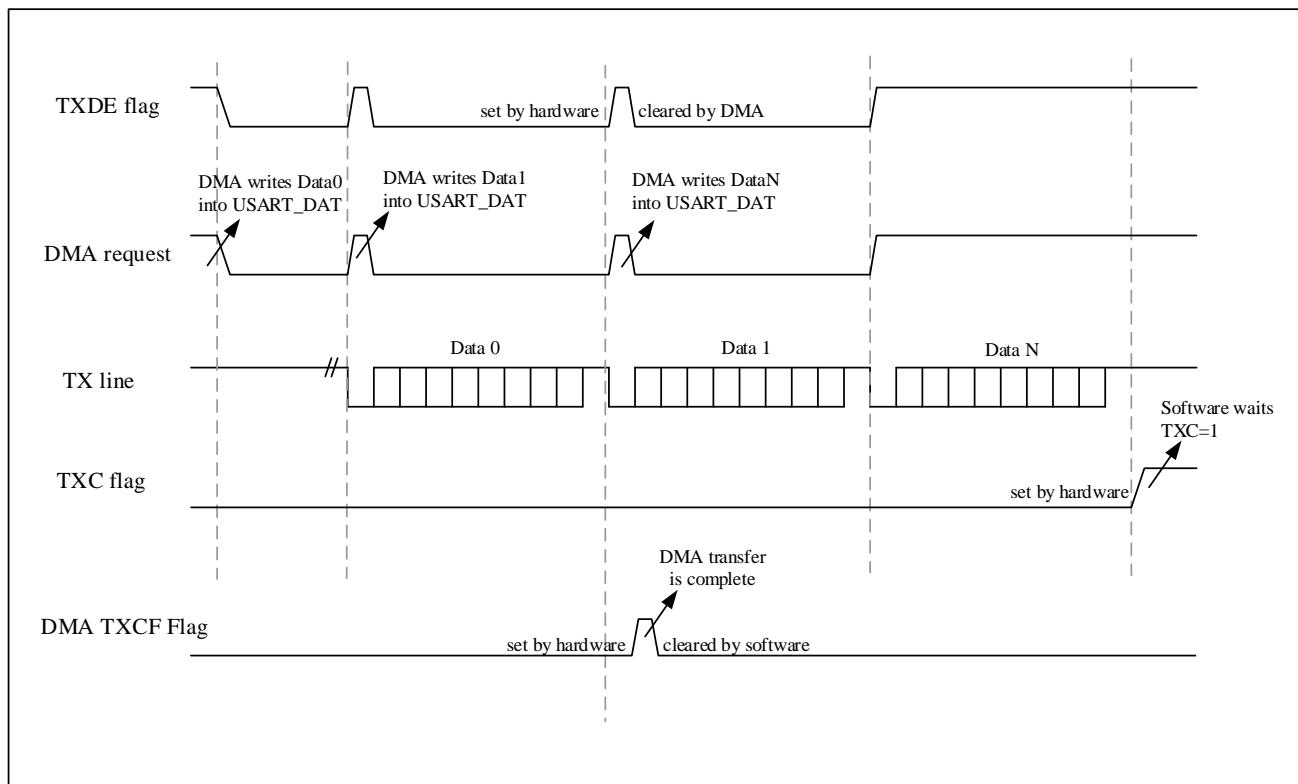
### 18.4.7.1 DMA 发送

发送器通过将 USART\_CTRL3.DMATXEN 置 1 来使能 DMA 发送。当发送移位寄存器为空时 (USART\_STS.TXDE=1),, DMA 将数据由 SRAM 送到数据寄存器 USART\_DAT。

使用 DMA 发送功能时，按照以下流程对 DMA 进行配置：

1. 设置 DMA 传输的源地址，DMA 传输时从此地址读取要发送的数据
2. 设置 DMA 传输的目地地址为 USART\_DAT 寄存器地址
3. 设置要传输的总的字节数.
4. 设置 DMA 通道优先级、循环模式、地址增加模式、传输数据宽度、中断（传输完成一半还是全部完成时）
5. 激活当前 DMA 通道
6. 传输完成后，标志位 DMA\_INTSTS.TXCFx 被置 1

图 18-7 DMA 发送



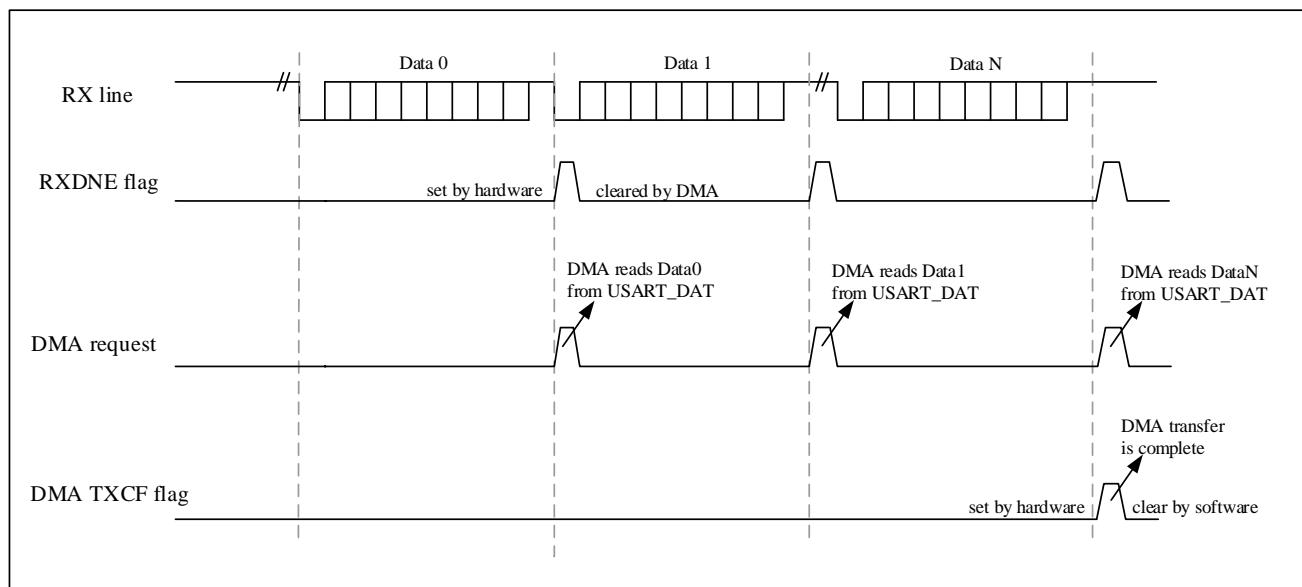
### 18.4.7.2 DMA 接收

接收器通过将 USART\_CTRL3.DMATXEN 置 1 来使能 DMA 接收。当收到 1 字节数据时 (USART\_STS.RXDNE=1), DMA 将数据从数据寄存器 USART\_DAT 读出数据，送到 SRAM。

使用 DMA 接收功能时，按照以下流程对 DMA 进行配置：

1. 设置 DMA 传输的源地址为 USART\_DAT 寄存器地址，DMA 传输时从此地址读取要发送的数据
2. 设置 DMA 传输的目地地址，DMA 传输时将数据送到此地址。
3. 设置要传输的总的字节数。
4. 设置 DMA 通道优先级、循环模式、地址增加模式、传输数据宽度、中断（传输完成一半还是全部完成时）
5. 激活当前 DMA 通道
6. 传输完成后，标志位 DMA\_INTSTS.TXCFx 被置 1

图 18-8 DMA 接收

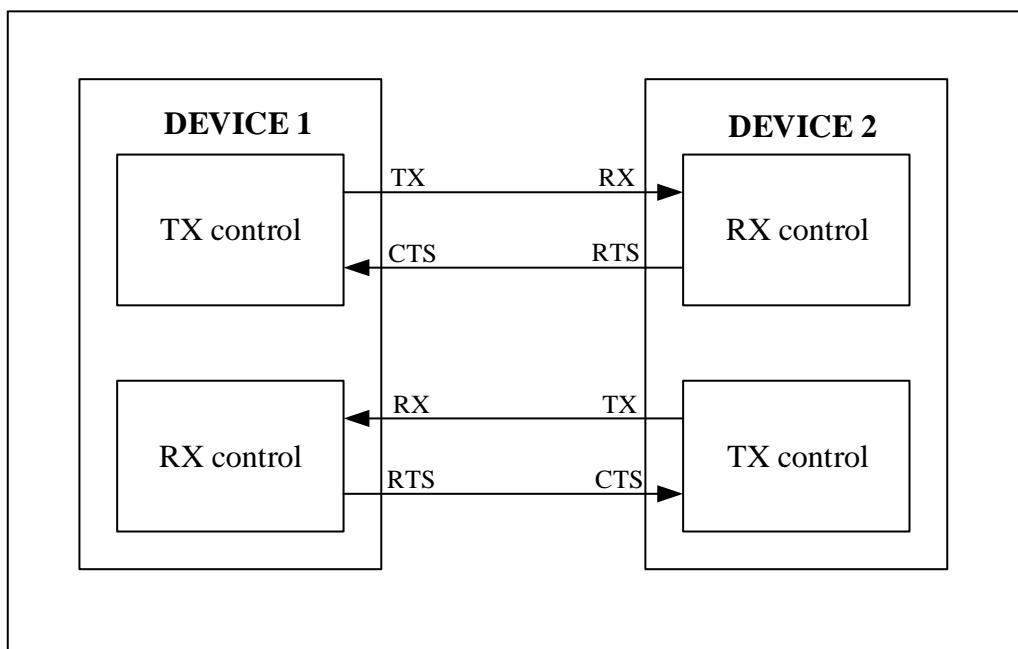


在多缓冲器通信模式，当检测到帧错误、溢出错误、噪声错误时，相应标志位置 1。如果此时 USART\_CTRL3.ERRIEN 已置 1，产生一个错误中断。

#### 18.4.8 硬件流控

USART 支持硬件流控，用于协调发送端与接收端时序，避免数据丢失。连接方式见下图。

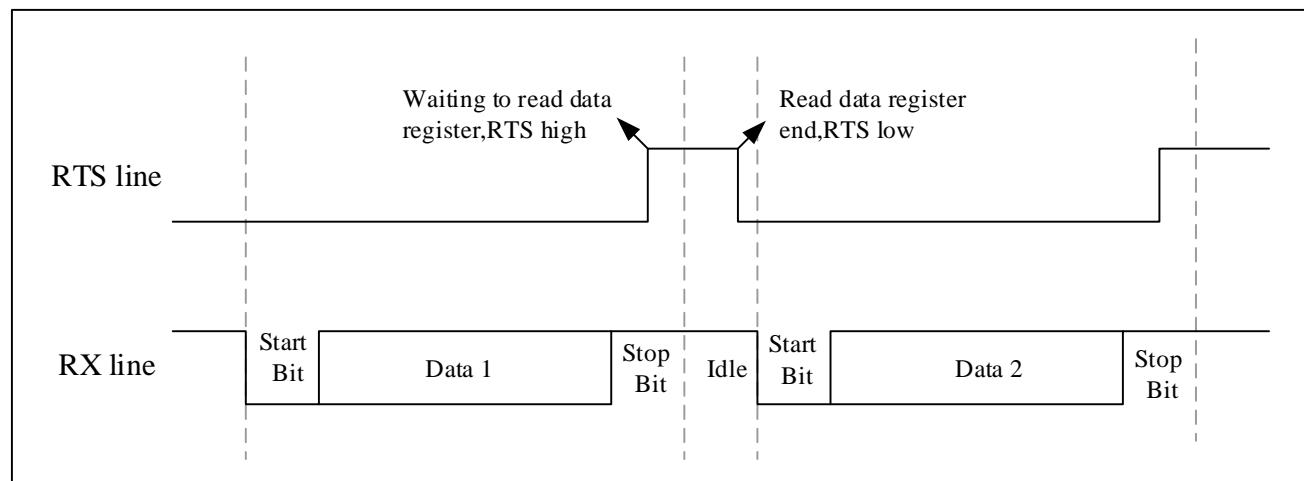
图 18-9 两个 USART 间的硬件流控制



#### 18.4.8.1 RTS 流控制

将 USART\_CTRL3.RTSEN 置 1，RTS 流控制使能。通过 nRTS 引脚输出低电平表示当前 USART 的接收器已准备好接收数据。当接收器收到数据后，nRTS 输出高电平，提示发送端暂停发送下一帧数据。如果接收器准备后接收下一帧数据，再次通过 nRTS 输出低电平。

图 18-10 RTS 流控制

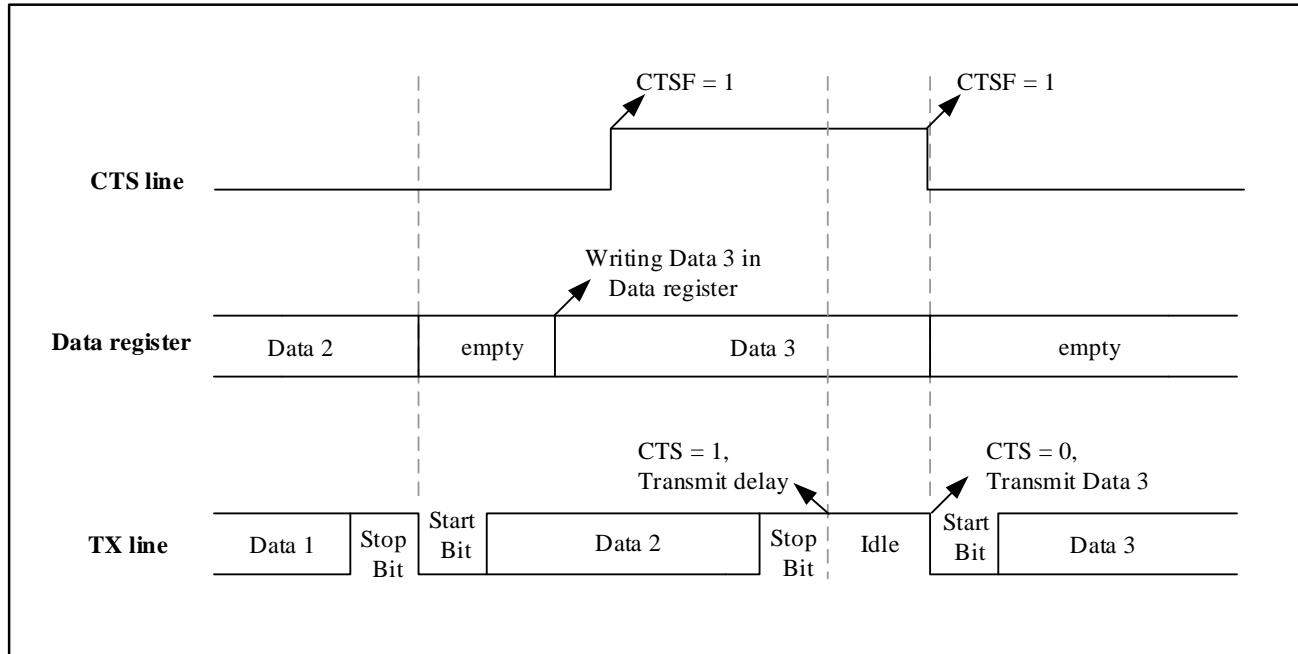


#### 18.4.8.2 CTS 流控制

将 USART\_CTRL3.CTSEN 置 1，CTS 流控制使能。nCTS 为输入引脚，用于判断是否能发送数据给其他设备。nCTS 检测到低电平时，表示可以发送数据给其他设备。如果在数据传输时 nCTS 被拉高（失效），在当前数据传输完成后停止发送。如果在 nCTS 无效时写数据到数据寄存器，数据保持，直到 nCTS 有效后开始

发送。当 nCTS 输入信号状态发生变化时, USART\_STS.CTSF 置 1。此时如果 USART\_CTRL3.CTSIEN 已置 1, 将产生一个中断。

图 18-11 CTS 流控制



## 18.4.9 多处理器通信

USART 支持多处理器通信: 多个设备同时连接到 USART 进行通信, 因此必须判定哪一个设备作为主设备, 其他设备自动做为从设备。主机的 TX 引脚直接连接到其他从设备的 RX 引脚, 所有从设备的 TX 引脚通过逻辑与的方式合并, 再连接到主设备的 RX 引脚。

在多处理器通信模式下, 从设备处于静默模式, 主设备在需要时通过指定方式唤醒某一个从设备, 从而从设备可以和主设备进行正常通信。

USART 可通过空闲总线检测或地址标识检测的方式从静默模式唤醒。

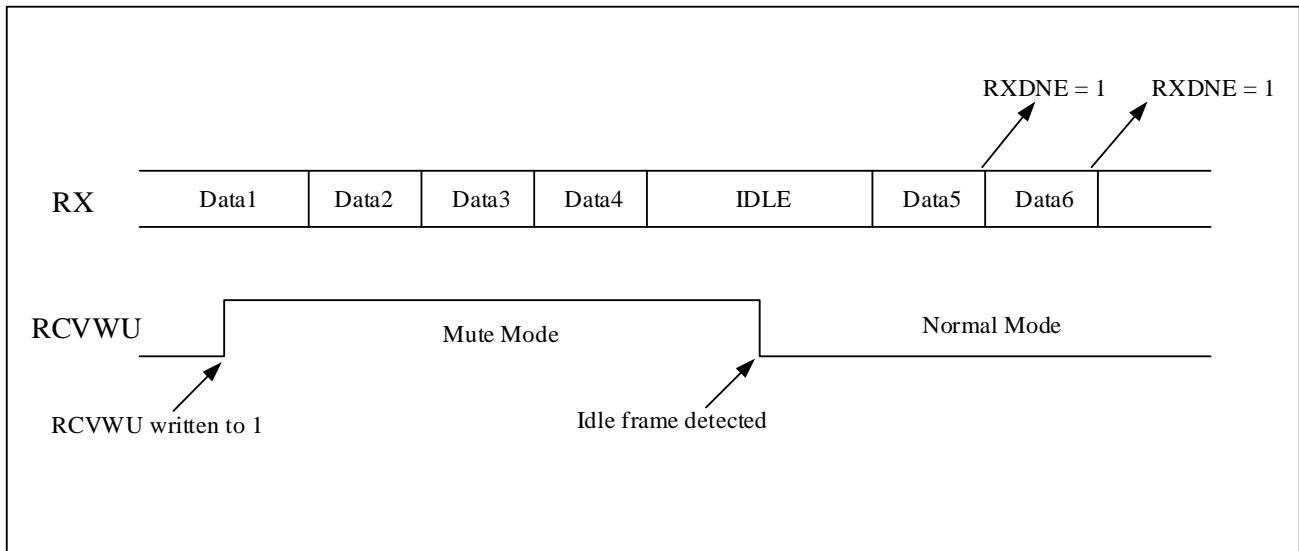
### 18.4.9.1 空闲总线检测

空闲总线检测流程如下:

1. 清零 USART\_CTRL1.WUM 位, USART 启用空闲总线检测功能。
2. 当 USART\_CTRL1.RCVWU 已置 1(可通过硬件自动控制或由在特定条件下由软件配置), USART 进入静默模式, 此时接收状态标志位不会置位, 同时接收中断被禁用。
3. 如图 18-12 所示, 当检测到空闲帧时, USART 被唤醒, 同时 USART\_CTRL1.RCVWU 被硬件清零, 此时

USART\_STS.IDLEF 标志位不会被置 1。

图 18-12 静默模式下的空闲总线检测



#### 18.4.9.2 地址标识检测

当 USART\_CTRL1.WUM 置 1 时, USART 启用地址标识检测功能。标识地址通过 USART\_CTRL2.ADDR[3:0] 来配置。如果接收的数据最高有效位 (MSB) 为 1, 当前数据为地址, 低 4 位有效; 如果 MSB =0, 则当前数据为普通数据。

此模式下, USART 可通过以下方式进入静默模式:

- 当接收器没有数据处理时, 可通过软件将 USART\_CTRL1.RCVWU 置 1, 使 USART 进入静默模式。

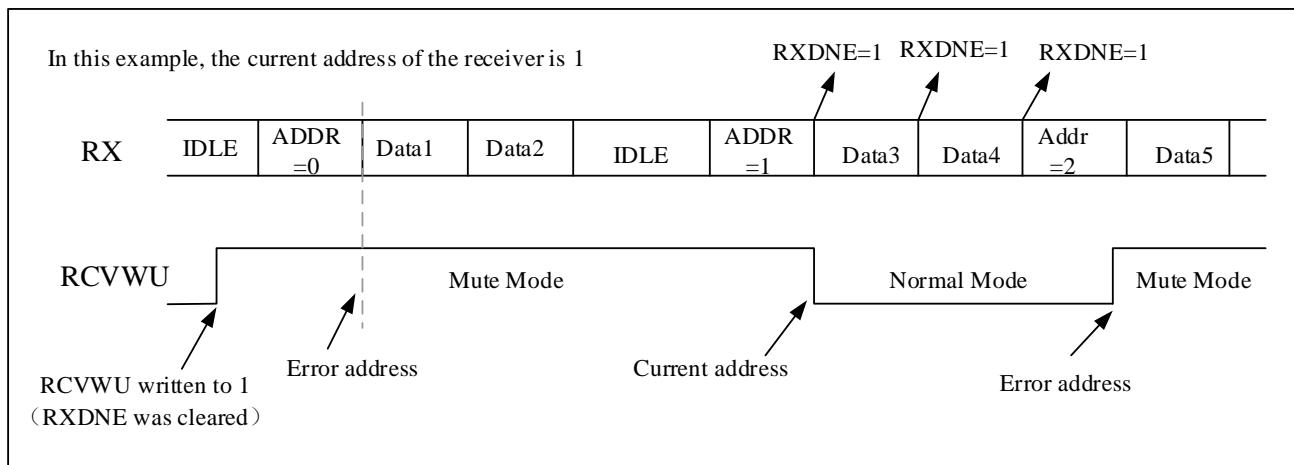
**注意:** 当接收数据寄存器为空时, ( $USART\_SR.RXDNE=0$ ), USART\_CTRL1.RCVWU 位可通过软件写 0 或写 1。否则, 对 USART\_CTRL1.RCVWU 的写操作被忽略。

- 当接收器收到的地址与预设的地址标识不匹配时, USART\_CTRL1.RCVWU 由硬件置 1。

静默模式下, 所有接收状态标志位不会置位, 同时所有接收中断被禁用。

当接收器收到的地址与预设的地址标识相同时, USART 从静默模式唤醒, USART\_CTRL1.RCVWU 被硬件清零, 同时 USART\_STS.RXDNE 位置 1, 此时可进行正常的数据传输。

图 18-13 静默模式下的地址标识检测



## 18.4.10 同步模式

USART 支持同步串行通信模式。同步模式下，USART 只能做为主设备，无法通过外部输入的时钟进行数据收发。通过将 USART\_CTRL2.CLKEN 位置 1 来启用同步模式。

**注意：**要启用同步模式，必须将以下控制位全部清零：USART\_CTRL2.LINMEN，USART\_CTRL3.SCMEN，USART\_CTRL3.HDMEN，USART\_CTRL3.IRDAMEN。

### 18.4.10.1 同步时钟

CK 引脚为同步时钟输出引脚。在总线空闲时、实际数据发送之前、以及发送断开标识时，同步时钟不输出。

同步时钟相位与极性可软件配置，且只能在发送器与接收器都被禁用时配置。

当时钟极性配置为 0 (USART\_CTRL2.CLKPOL=0) 时，空闲时 CK 引脚输出低电平；当时钟极性配置为 1 (USART\_CTRL2.CLKPOL=1) 时，空闲时 CK 引脚输出高电平。

当相位配置为 0 (USART\_CTRL2.CLKPHA=0) 时，数据在第一个时钟沿采样；当相位配置为 1 (USART\_CTRL2.CLKPHA=1) 时，数据在第二个时钟沿采样。

在发送起始位或停止位时，CK 引脚保持空闲状态，无时钟不输出。

未发送数据时无法接收同步数据。因为时钟仅在发送器被激活且数据写入 USART\_DAT 寄存器时可用。

USART\_CTRL2.LBCLK 位控制数据字节的最高有效位 (MSB) 是否有时钟脉冲。此位只能在发送器与接收器都被禁用时配置。当 USART\_CTRL2.LBCLK = 1 时，则最后一位数据的时钟脉冲将从 CK 输出；当 USART\_CTRL2.LBCLK = 0 时，则最后一位数据的时钟脉冲不从 CK 输出。

### 18.4.10.2 同步发送

同步模式下的数据发送与异步模式相同，数据从 TX 引脚输出，同时从 CK 引脚输出对应的时钟脉冲。

### 18.4.10.3 同步接收

同步模式下的数据接收与异步模式不同。数据在 CK 引脚输出的有效时钟沿采样，而不使用过采样。但必须考虑数据建立时间与保持时间 (1/16 数据位周期，具体时间依赖于波特率)。

图 18-14 USART 同步传输示例

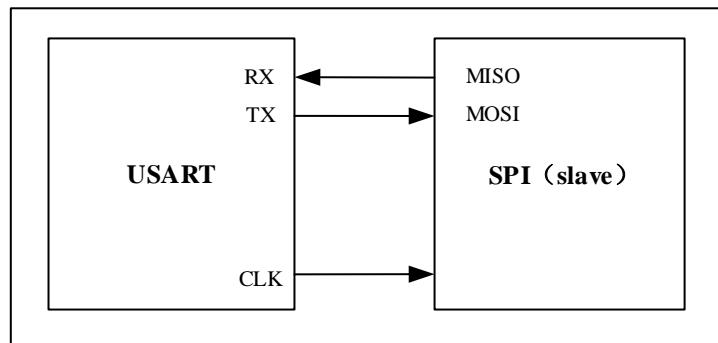


图 18-15 USART 数据时钟时序示例 (WL=0)

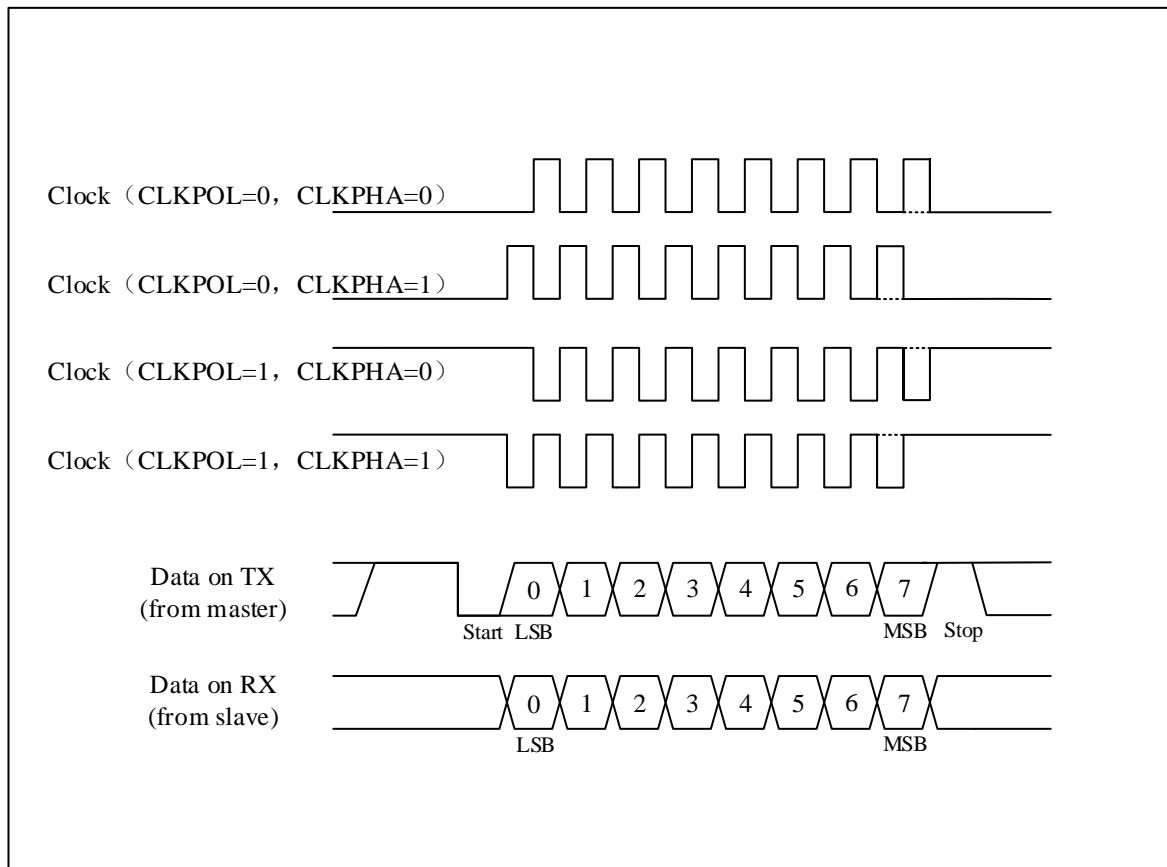


图 18-16 USART 数据时钟时序示例 (WL=1)

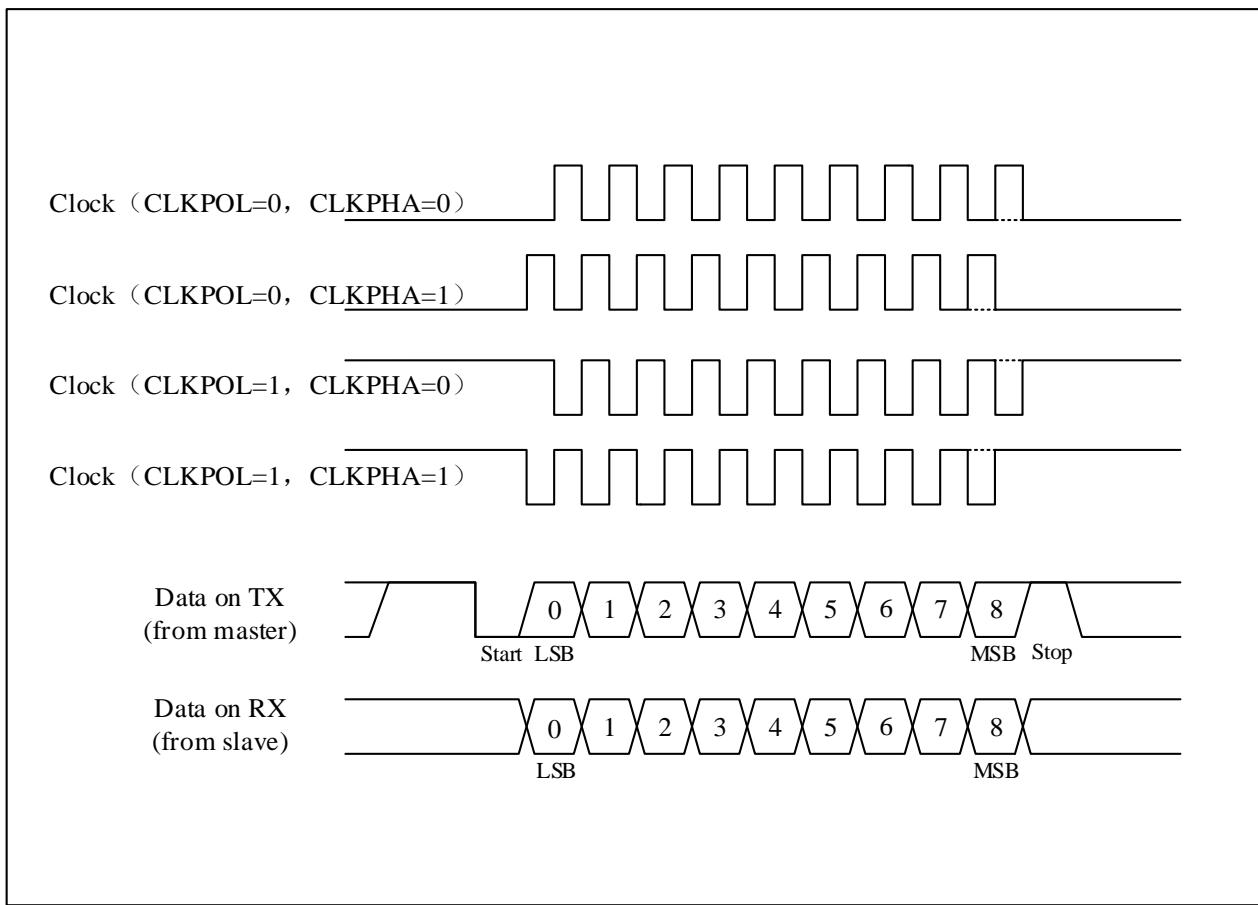
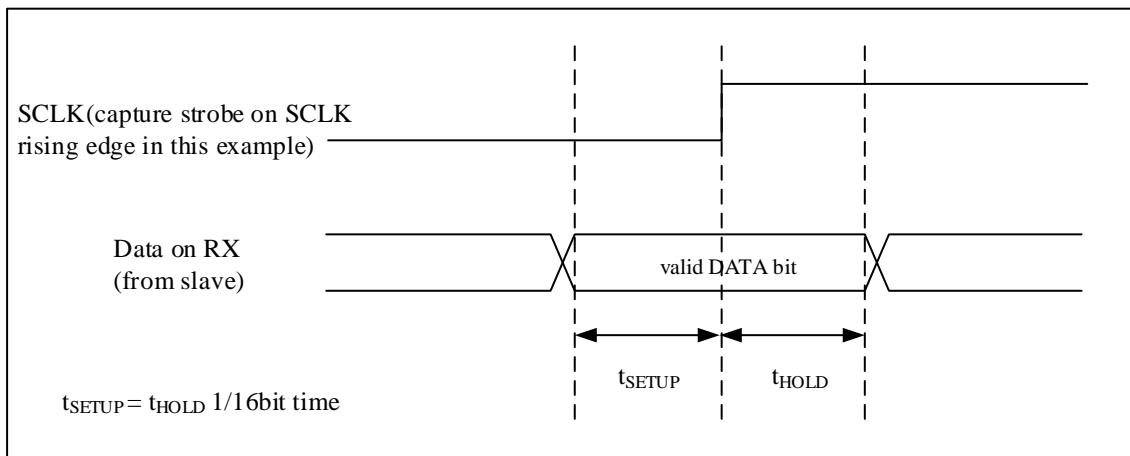


图 18-17 RX 数据采样/保持时间



注意：在智能卡模式下，CK 引脚功能与同步模式不同，有关细节请参考智能卡模式部分。

#### 18.4.11 单线半双工模式

USART 支持单线半双工通信模式，允许数据双方向收发，但同一时间只能单向接收数据或发送数据，数据

通信的冲突由软件控制。

通过设置 USART\_CTRL3.HDMEN 位来选择单线半双工模式，此时以下控制位必须全部清零：

USART\_CTRL2.CLKEN、USART\_CTRL2.LINMEN、USART\_CTRL3.SCMEN、USART\_CTRL3.IRDAMEN。

启用单线半双工通信模式后，TX 引脚与 RX 引脚在芯片内部相连，外部 RX 引脚不再使用。当没有数据发送时，TX 引脚被释放。因此，TX 引脚未被 USART 使用时，必须配置为浮空输入或开漏输出高电平。

### 18.4.12 串行 IrDA 红外编解码模式

USART 支持 IrDA (Infrared Data Association ) SIR ENDEC 规范。

通过设置 USART\_CTRL3.IRDAMEN 位来选择是否使用 IrDA 模式。当启用 IrDA 模式时，以下配置位必须全部清零： USART\_CTRL2.CLKEN、USART\_CTRL2.STPB[1:0]、USART\_CTRL2.LINMEN、USART\_CTRL3.HDMEN、USART\_CTRL3.SCMEN。

通过设置 USART\_CTRL3.IRDALP 位，可选择 IrDA 的正常工作模式或低功耗模式。

当使用 IrDA 模式时，停止位长度必须配置为 1 位。

#### 18.4.12.1 IrDA 正常模式

当 USART\_CTRL3.IRDALP=0，IrDA 工作在正常模式。

IrDA 是一个半双工通信接口，因此在发送和接收之间最小要有 10ms 的延时。数据采用反相归零(RZI)调制，即采用红外光源脉冲表示逻辑 0。脉冲宽度规定为一个位周期的 3/16，如图 18-19 所示。最大波特率为 115200bps。

USART 将数据送到 SIR 编码器进行调制后输出。调制后的数据流输出给外部红外发送器进行发送。接收时，先通过外部红外接收器接收数据并解调后，发送到 SIR 解码器，解码后再将数据送给 USART。

发送编码器与解码器输入极性相反。空闲时，编码器输出为低电平，而解码器输入为高电平。编码器输出高脉冲表示逻辑 0，输出低电平作为逻辑 1。解码器输入则与之相反。

当 USART 正在发送数据给 IrDA 编码器时，解码器将忽略数据线上的所有数据。当 USART 正在从解码器接收数据时，发送到编码器的数据也被忽略，不进行编码操作。

脉冲宽度可软件配置。IrDA 规范要求脉冲宽度大于 1.41us。如果脉冲宽度小于 2 个周期，数据被过滤而丢失。PSCV 是在 USART\_GTP 寄存器配置的预分频值。

### 18.4.12.2 IrDA 低功耗模式

当 USART\_CTRL3.IRDALP=1，IrDA 工作在低功耗模式。

在低功耗模式下发送数据时，脉冲宽度为 3 倍 PSCV 周期。经 PSCV 分频后的时钟频率最小值为 1.42MHz，典型值为 1.8432MHz，(1.42 MHz < 时钟频率 < 2.12 MHz)。

接收数据时，有效低电平信号宽度必须大于 2 个 PSCV 周期。

图 18-18 IrDA SIR ENDEC-框图

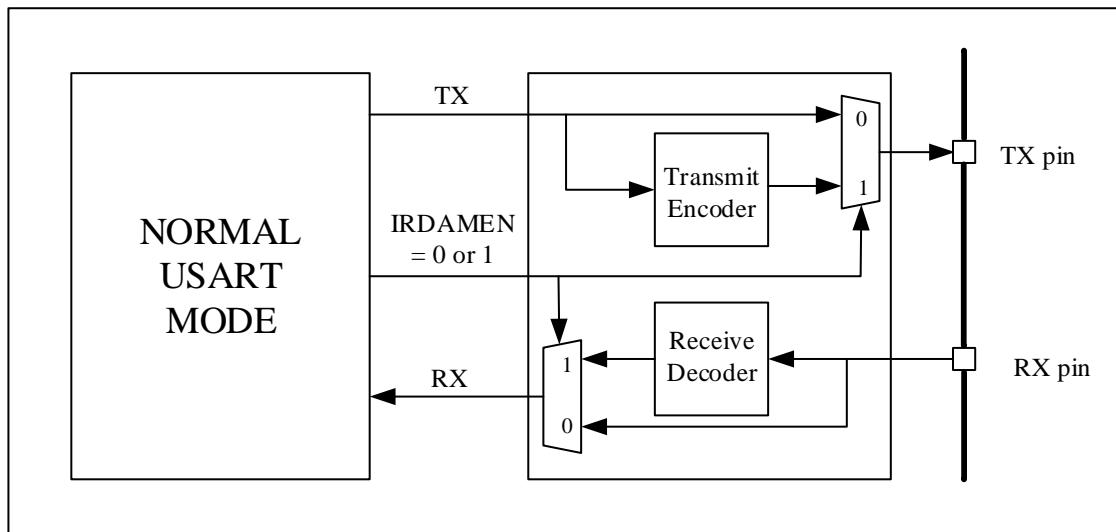
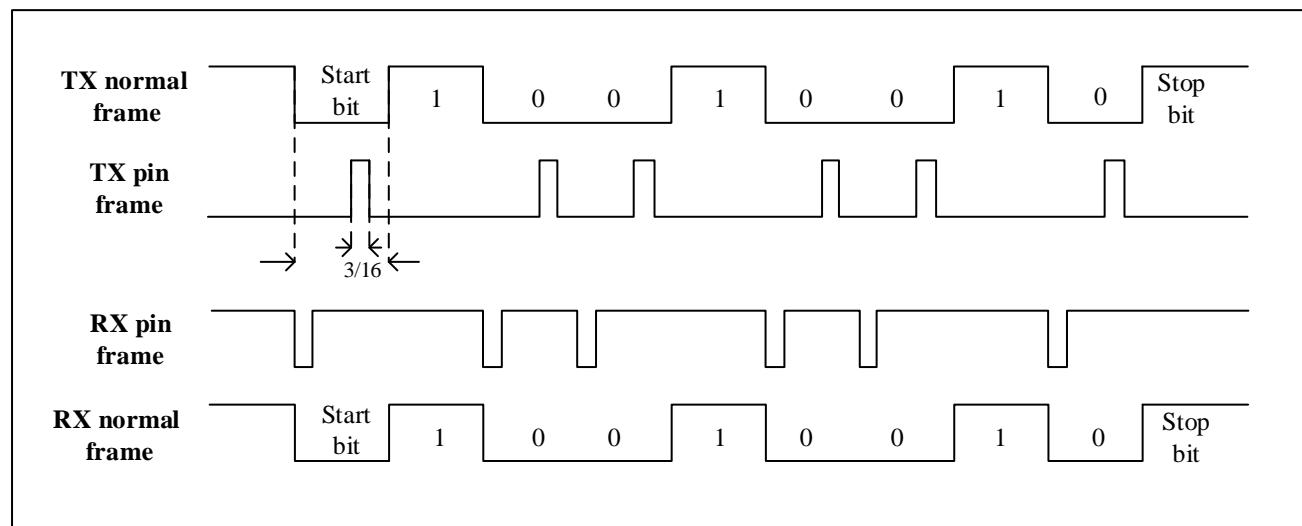


图 18-19 IrDA 数据调制 (3/16)-正常模式



### 18.4.13 LIN 模式

USART 支持 LIN (Local Interconnection Network)模式，支持作为主机时发送同步断开帧，也支持作为从机检测断开帧。通过设置 USART\_CTRL2.LINMEN 位来使能 LIN 模式。.

注意，当使用 LIN 模式时，以下配置位必须全部被清零：`USART_CTRL2.STPB[1:0]`、`USART_CTRL2.CLKEN`、`USART_CTRL3.SCMEN`、`USART_CTRL3.HDMEN`、`USART_CTRL3.IRDAMEN`。

#### 18.4.13.1 LIN 发送

在 LIN 模式下发送数据时，数据长度只能配置为 8 位。将 `USART_CTRL1.SDBRK` 置 1 将发送一个 13 位“0”断开帧，并插入一个停止位。

#### 18.4.13.2 LIN 接收

当总线空闲或数据传输过程中均可检测断开帧。断开帧检测机制独立于 USART 接收器。

通过配置 `USART_CTRL2.LINBDL` 位，断开帧检测有效低电平位可选择 10 位或 11 位。

当接收器检测到一个起始位，采样电路在每个位的第 8, 9, 10 个过采样时钟点进行过采样。如果 10 个或 11 个位都是‘0’，并且又跟着一个定界符，表示检测到一个断开帧，`USART_STS.LINBDF` 位置 1。在确认为断开帧前，必须检测定界符，意味着 RX 线已经回归空闲状态(高电平)。此时如果 `USART_CTRL2.LINBDIEN` 已置 1，将产生一个中断。

如果在 10 个或 11 个位前收到了‘1’，当前断开帧检测被取消，并重新寻找起始位。

图 18-20 LIN 模式下的断开检测（11 位断开帧长度-设置了 LINBDF 位）

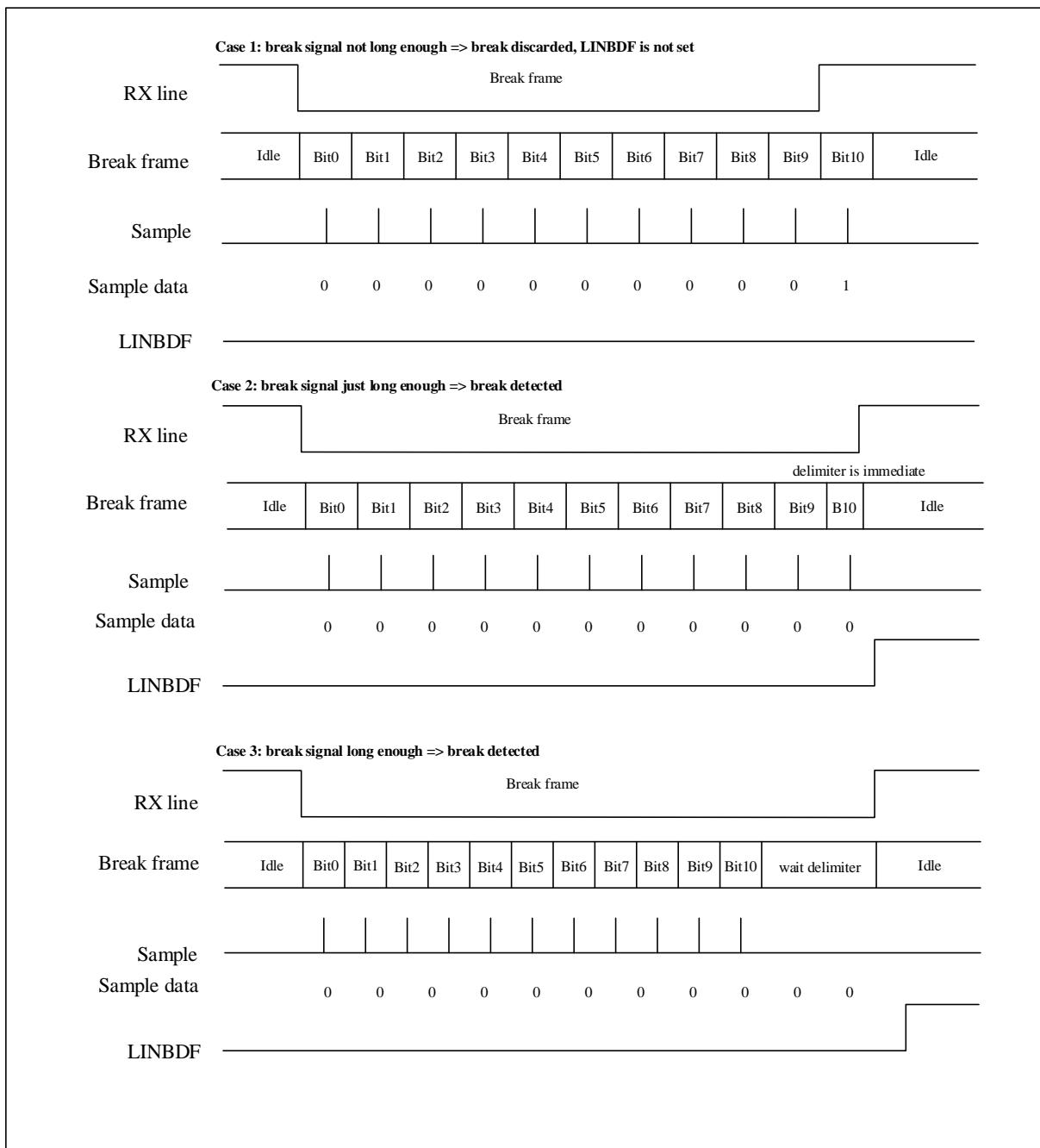
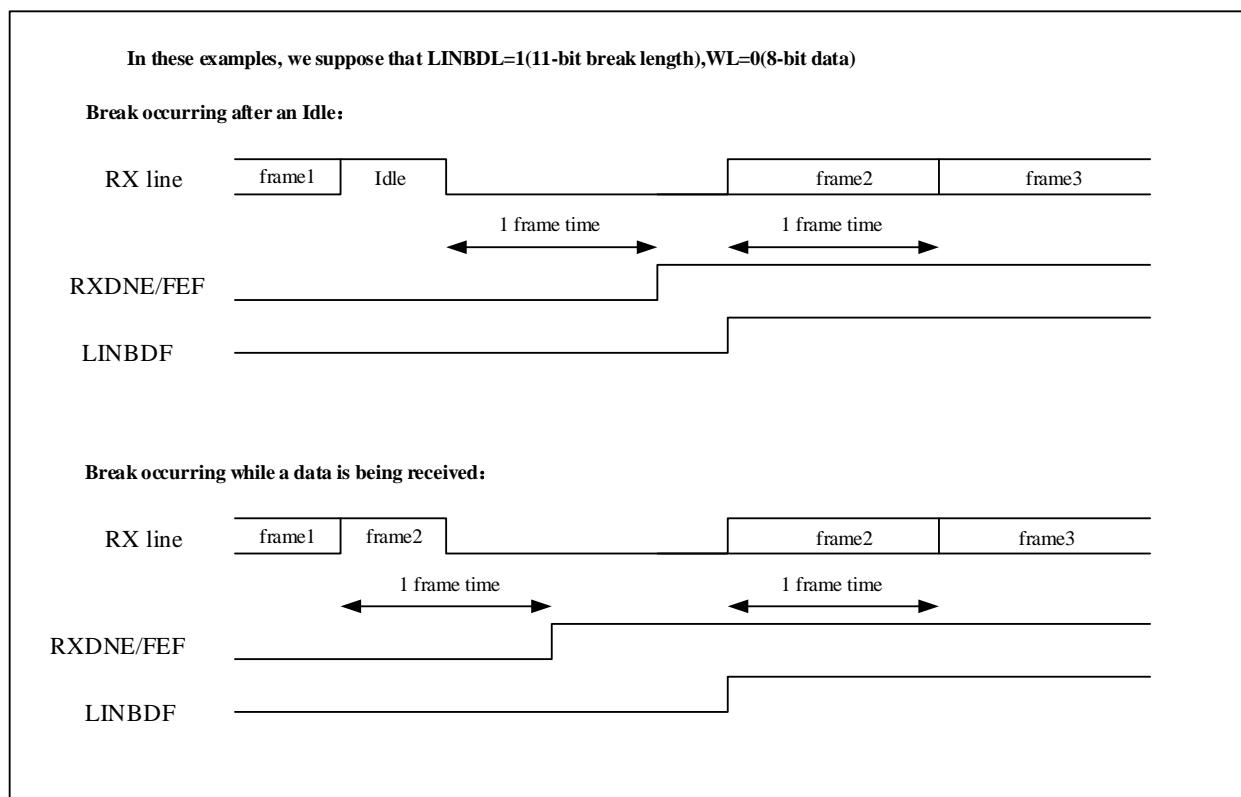


图 18-21 LIN 模式下的断开检测与帧错误的检测



#### 18.4.14 智能卡模式 (ISO7816)

USART 支持智能卡规范，支持 ISO7816-3 标准中定义的智能卡协议。

通过配置 USART\_CTRL3.SCMEN 位来选择是否启用智能卡模式。使用智能卡模式时，以下配置位必须全部清零：USART\_CTRL2.LINMEN、USART\_CTRL3.HDMEN、USART\_CTRL3.IRDAMEN。

智能卡模式中，USART 通过 CK 引脚提供时钟，时钟频率通过预分频寄存器配置，配置范围为  $f_{CK}/2$  至  $f_{CK}/62$  其中  $f_{CK}$  为当前 USART 输入外设时钟。

智能卡模式下，接收数据时可采用 0.5 位或 1.5 位停止位，但发送数据时停止位长度只能配置为 1.5 位。因此建议在发送和接收时均使用 1.5 个停止位，以避免在 2 种停止位长度配置间频繁切换。

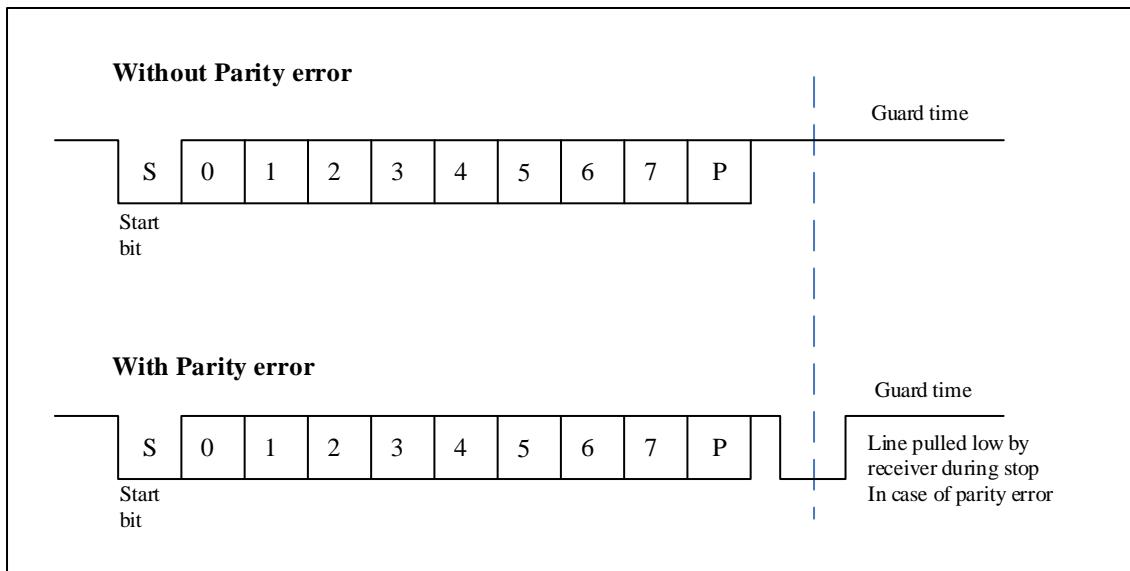
智能卡模式下，数据长度必须配置为 8 位，且校验位需要配置。

当接收器检测到一个校验错误时，在停止位后将数据发送线拉低一个波特时钟作为 NACK 信号 (USART\_CTRL3.SCNACK 位置 1 时)，同时在发送端产生一个帧错误 (发送端停止位为 1.5 位)。

当发送器接收到来自接收器的 NACK 信号 (帧错误) 时，它不会将 NACK 作为起始位 (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)。

下图为有无校验错误时的两种时序示例。

图 18-22 ISO7816-3 异步协议



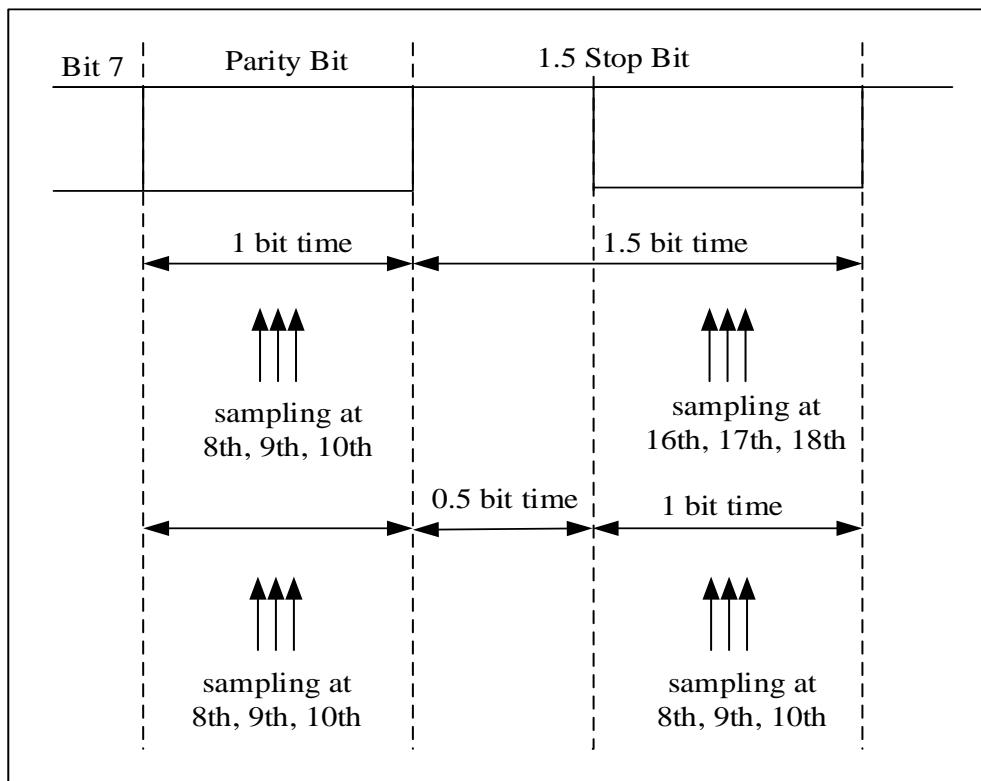
在智能卡模式下，不支持断开帧。如果接收到断开帧，视为一个带帧错误的 00h 数据帧处理。.

在普通模式下，数据在下一个波特时钟内从发送移位寄存器移出，而在智能卡模式下，数据发送比起普通模式至少延迟 1/2 个位波特时钟。

普通模式下，数据发送完成并且 USART\_STS.TXDE = 1 时，USART\_STS.TXC 位被置 1。在智能卡模式下，数据发送完成且保护时间达到预设值（USART\_GTP.GTV[7:0]）时，USART\_STS.TXC 位才被置 1，且 USART\_STS.TXC 标志的清零不受智能卡模式影响。

下图为 USART 采样 NACK 信号示意图。

图 18-23 使用 1.5 停止位检测奇偶检验错误



## 18.5 中断请求

USART 的各种中断事件是逻辑或的关系。如果某个事件对应的中断使能位已置 1，将产生一个相应的中断。但同一个时间只产生一个中断请求。

表 18-7 USART 中断请求

中断函数	中断事件	事件标志	使能
USART 全局中断	发送数据寄存器空	TXDE	TXDEIEN
	CTS 标志	CTSF	CTSIEN
	发送完成	TXC	TXCIEN
	接收数据就绪可读	RXDNE	RXDNEIEN
	检测到数据溢出	ORERR	
	检测到空闲线路	IDLEF	IDLEIEN
	奇偶检验错	PEF	PEIEN
	断开标志	LINBDF	LINBDIEN

	噪声标志, 多缓冲通信中的溢出错误和帧错误 <sup>(1)</sup>	NEF/OREF/FEF	ERRIEN <sup>(1)</sup>
--	--------------------------------------	--------------	-----------------------

(1) 仅当使用 DMA 接收数据(USART\_CTRL3.DMARXEN=1)时, 才使用这个标志位。

## 18.6 模式配置

表 18-8 USART 模式设置<sup>(1)</sup>

通信模式	USART1	USART2	UART3	UART4
异步模式	Y	Y	Y	Y
硬件流控模式	Y	Y	N	N
DMA 通讯模式	Y	Y	Y	Y
多处理器	Y	Y	Y	Y
同步模式	Y	Y	N	N
智能卡模式	Y	Y	N	N
单线半双工模式	Y	Y	Y	Y
IrDA 红外模式	Y	Y	Y	Y
LIN	Y	Y	Y	Y

(1) Y = 支持该模式, N = 不支持该模式

## 18.7 USART 寄存器

### 18.7.1 USART 寄存器总览

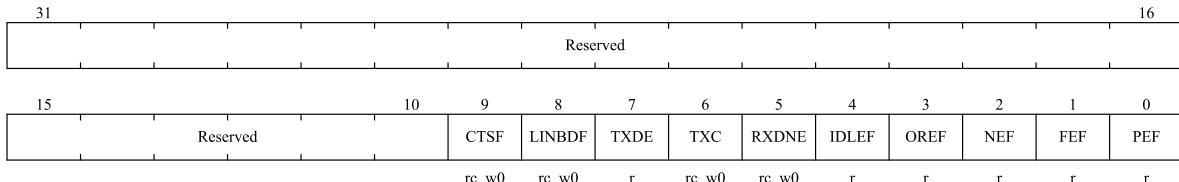
表 18-9 USART 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
000h	USART_STS																																				
	Reset Value																																				
004h	USART_DAT																																				
	Reset Value																																				
008h	USART_BRCF																																				
	Reset Value																																				
00Ch	USART_CTRL1																																				
	Reset Value																																				
010h	USART_CTRL2																																				
	Reset Value																																				
014h	USART_CTRL3																																				
	Reset Value																																				
018h	USART_GTP																																				
	Reset Value																																				

## 18.7.2 USART 状态寄存器 (USART\_STS)

偏移地址: 0x00

复位值: 0x0000 00C0



位域	名称	描述
31:10	Reserved	保留, 必需保持复位值。
9	CTSF	<p>CTS 标志 (CTS flag)。 如果设置了 USART_CTRL3.CTSEN 位, 当 nCTS 输入变化时, 该位由硬件置位。如果设置了 USART_CTRL3.CTSIEN 位, 将产生中断。 该位由软件清 0。</p> <p>0: nCTS 状态线没有变化。 1: nCTS 状态线发生变化。 注意: 该位对 UART3/4 无效。</p>
8	LINBDF	<p>LIN 断开检测标志 (LIN break detection flag)。 如果设置了 USART_CTRL2.LINMEN 位, 当检测到 LIN 断开, 该位由硬件置位。如果 USART_CTRL2.LINBDIEN 被置位时, 将产生中断。 该位由软件清 0。</p> <p>0: 没有检测到 LIN 断开字符。 1: 检测到 LIN 断开字符。</p>
7	TXDE	<p>发送数据缓冲区空 (Transmit data register empty)。 上电复位或待发送数据已发送至移位寄存器后, 该位置 1。 USART_CTRL1.TXDEIEN 被置位将产生中断。 该位在软件将待发送数据写入 USART_DAT 时被清 0。</p> <p>0: 发送数据缓冲区不为空。 1: 发送数据缓冲区空。</p>
6	TXC	<p>发送完成 (Transmission complete)。 上电复位后, 该位被置 1。如果 USART_STS.TXDE 置位, 在当前数据发送完成时该位置 1。 USART_CTRL1.TXCIEN 被置位将产生中断。 该位由软件清 0。</p> <p>0: 发送没有完成。 1: 发送完成。</p>
5	RXDNE	<p>读数据缓冲区非空 (Read data register not empty)。 当读数据缓冲区接收到来自移位寄存器的数据时, 该位置 1。当寄存器</p>

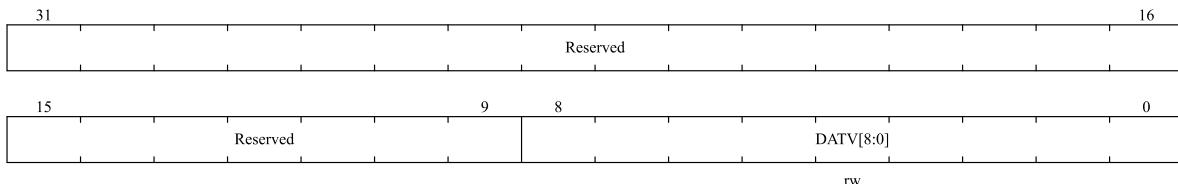
		<p>USART_CTRL1.RXDNEIEN 位被置位，将会有中断产生。 软件可以通过对该位写 0 或读 USART_DAT 寄存器来将该位清 0。 0: 读数据缓冲区为空。 1: 读数据缓冲区不为空。</p>
4	IDLEF	<p>空闲线检测标志 (IDLE line detected)。 在一个帧时间内，在 RX 引脚检测到空闲状态，该位置 1。当寄存器 USART_CTRL1.IDLEIEN 位被置位，将会有中断产生。 软件先读 USART_STS，再读 USART_DAT 可清除该位。 0: 未检测到空闲帧。 1: 检测到空闲帧。 <i>注意：IDLEF 位不会再次被置高直到 RXDNE 位被置起（即又检测到一次空闲总线）。</i></p>
3	OREF	<p>溢出错误 (Overrun error)。 在 RXDNE 置位的情况下，如果 USART_DAT 寄存器接收到来自移位寄存器的数据，该位置 1。当寄存器 USART_CTRL3.ERRIEN 位被置位，将会有中断产生。 软件先读 USART_STS，再读 USART_DAT 可清除该位。 0: 没有检测到溢出错误。 1: 检测到溢出错误。</p>
2	NEF	<p>噪声错误标志 (Noise error flag)。 在接收到的帧检测到噪音时，由硬件对该位置位。由软件序列对其清零 (先读 USART_STS，再读 USART_DAT)。 0: 没检测到噪声错误。 1: 检测到噪声错误。 <i>注意：该位不会产生中断，因为它和 USART_STS.RXDNE 一起出现，硬件会在设置 USART_STS.RXDNE 标志时产生中断。在多缓冲区通信模式下，如果设置了 USART_CTRL3.ERRIEN 位，则设置 NEF 标志时会产生中断。</i></p>
1	FEF	<p>帧错误 (Framing error)。 当检测到同步错位，过多的噪音或者检测到断开符，该位被硬件置位。 由软件序列将其清零 (先读 USART_STS，再读 USART_DAT)。 0: 未检测到帧错误。 1: 检测到帧错误或者断开帧 (break frame)。 <i>注意：该位不会产生中断，因为它和 USART_STS.RXDNE 一起出现，硬件会在设置 USART_STS.RXDNE 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 OREF 标志位。</i> <i>在多缓冲区通信模式下，如果设置了 USART_CTRL3.ERRIEN 位，则设置 FEF 标志时会产生中断。</i></p>
0	PEF	<p>校验错误 (Parity error)。 当接收到的数据帧校验位与预期校验值不同时，该位置位。 软件先读 USART_STS，再读 USART_DAT 可清除该位。 0: 没检测到校验错误。</p>

		1: 检测到校验错误。
--	--	-------------

### 18.7.3 USART 数据寄存器(USART\_DAT)

偏移地址: 0x04

复位值: 未定义 (不确定值)



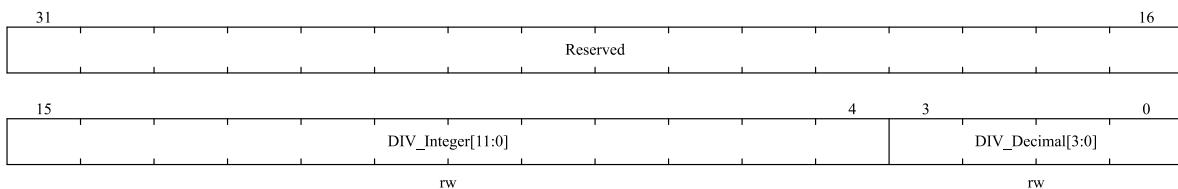
位域	名称	描述
31:9	Reserved	保留, 必需保持复位值。
8:0	DATV[8:0]	数据值 (Data value) 包含了发送或接收的数据; 软件可以通过写这些位来改变发送数据, 或读这些位的值来获取接收数据。 如果使能了奇偶校验, 当发送数据被写入寄存器, 数据的最高位 (第 7 位或第 8 位取决于 USART_CTRL1.WL 位) 将被校验位取代。

### 18.7.4 USART 波特率配置寄存器 (USART\_BRCF)

偏移地址: 0x08

复位值: 0x0000 0000

注意: USART\_CTRL1.UEN=1 时, 不能写该寄存器; 如果 USART\_CTRL1.TXNE 或 USART\_CTRL1.RXNE 被分别禁止, 波特计数器停止计数。



位域	名称	描述
31:16	Reserved	保留, 必需保持复位值。
15:4	DIV_Integer [11:0]	波特率分频器的整数部分。
3:0	DIV_Decimal[3:0]	波特率分频器的小数部分。

## 18.7.5 USART 控制寄存器 1 (USART\_CTRL1)

偏移地址: 0x0C

复位值: 0x0000 0000

31	Reserved															16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved	UEN	WL	WUM	PCEN	PSEL	PEIEN	TXDE IEN	TXC IEN	RXDNE IEN	IDLE IEN	TXEN	RXEN	RCVWU	SDBRK		
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

位域	名称	描述
31:14	Reserved	保留, 必需保持复位值。
13	UEN	USART 使能 (USART enable)。 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, 以减少功耗。该位由软件设置和清零。 0: USART 禁用。 1: USART 使能。
12	WL	字长 (Word length)。 0: 8 数据位。 1: 9 数据位。 注意: 在数据传输过程中 (发送或者接收时), 不能修改这个位。
11	WUM	从静默模式唤醒方法 (Wake up mode)。 0: 空闲帧唤醒。 1: 地址标识唤醒。
10	PCEN	校验控制使能 (Parity control enable)。 0: 校验控制禁用。 1: 校验控制被使能。
9	PSEL	校验模式 (Parity selection)。 0: 偶校验。 1: 奇校验。
8	PEIEN	校验错误中断使能 (PE interrupt enable)。 如果该位置 1, USART_STS.PEF 被置位时产生中断。 0: 校验错误中断禁用。 1: 校验错误中断使能。
7	TXDEIEN	发送缓冲区空中断使能 (TXDE interrupt enable)。 如果该位置 1, USART_STS.TXDE 被置位时产生中断。 0: 发送缓冲区空中断禁止。 1: 发送缓冲区空中断使能。
6	TXCIEN	发送完成中断使能 (Transmission complete interrupt enable)。 如果该位置 1, USART_STS.TXC 被置位时产生中断。

位域	名称	描述
		0: 发送完成中断禁用。 1: 发送完成中断使能。
5	RXDNEIEN	读数据缓冲区非空中断和过载错误中断使能 (RXDNE interrupt enable)。 如果该位置 1, USART_STS.RXDNE 或 USART_STS.OREF 被置位时产生中断。 0: 读数据缓冲区非空中断和过载错误中断禁用。 1: 读数据缓冲区非空中断和过载错误中断使能。
4	IDLEIEN	IDLE 线检测中断使能 (IDLE interrupt enable)。 如果该位置 1, USART_STS.IDLEF 被置位时产生中断。 0: IDLE 线检测中断禁用。 1: IDLE 线检测中断禁用使能。
3	TXEN	发送器使能 (Transmitter enable)。 0: 发送器禁用。 1: 发送器使能。
2	RXEN	接收器使能 (Receiver enable)。 0: 接收器禁用。 1: 接收器使能。
1	RCVWU	接收器从静默模式中唤醒 (Receiver wakeup) 软件可以通过将该位置 1 使得 USART 进入静默模式, 将该位清 0 唤醒 USART。 空闲帧唤醒模式下 (USART_CTRL1.WUM=0), 当检测到空闲帧时, 该位由硬件清 0。地址标识唤醒模式下 (USART_CTRL1.WUM=1), 当接收到一个地址匹配帧时, 该位由硬件清 0; 或接收到一个地址非匹配帧时, 由硬件置 1。 0: 接收器处于普通工作模式。 1: 接收器处于静默模式。
0	SDBRK	发送断开帧 (Send break)。 软件通过将该位置 1 发送断开帧。 断开帧传输结束由硬件清 0 该位。 0: 没有发送断开帧。 1: 发送断开帧。

## 18.7.6 USART 控制寄存器 2 (USART\_CTRL2)

偏移地址: 0x10

复位值: 0x0000 0000

Reserved															16
15	14	13	12	11	10	9	8	7	6	5	4	3	0		
Reserved	LINMEN	STPB[1:0]	CLKEN	CLKPOL	CLKPHA	LBCLK	Reserved	LINBDIEN	LINBDL	Reserved					ADDR[3:0]
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw					rw

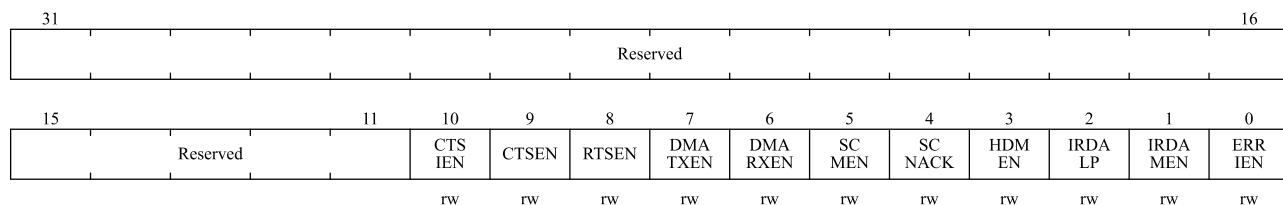
位域	名称	描述
31:15	Reserved	保留，必需保持复位值。
14	LINMEN	LIN 模式使能 (LIN mode enable) 0: LIN 模式禁用 1: LIN 模式使能
13:12	STPB[1:0]	停止位长 (STOP bits)。 00: 1 停止位。 01: 0.5 停止位。 10: 2 停止位。 11: 1.5 停止位。 <b>注意：对于 UART3/4，只有1位停止位和2位停止位是有效的。</b>
11	CLKEN	时钟使能 (Clock enable) 0: CK 引脚禁用 1: CK 引脚使能 <b>注意：该位对 UART3/4 无效。</b>
10	CLKPOL	时钟极性 (Clock polarity)。 该位用来设定在同步模式下 CK 引脚的极性。 0: CK 引脚不对外发送时保持为低电平。 1: CK 引脚不对外发送时保持为高电平。 <b>注意：该位对 UART3/4 无效。</b>
9	CLKPHA	时钟相位 (Clock phase)。 该位用来设定在同步模式下 CK 引脚的相位。 0: 在首个时钟边沿采样第一个数据。 1: 在第二个时钟边沿采样第一个数据。 <b>注意：该位对 UART3/4 无效。</b>
8	LBCLK	最后一位时钟脉冲 (Last bit clock pulse)。 该位用来设定在同步模式下是否在 CK 引脚上输出最后发送的那个数据字节 (MSB) 对应的时钟脉冲。 0: 最后一位数据的时钟脉冲不从 CK 输出。 1: 最后一位数据的时钟脉冲会从 CK 输出。 <b>注意：该位对 UART3/4 无效。</b>
7	Reserved	保留，必需保持复位值。
6	LINBDIEN	LIN 断开帧检测中断使能 (LIN break detection interrupt enable)。 如果该位置 1，当 USART_STS.LBDF 被置位时将产生中断。 0: 断开信号检测中断禁用 1: 断开信号检测中断使能
5	LINBDL	LIN 断开帧检测长度 (LIN break detection length)。

位域	名称	描述
		<p>该位用来设定在断开帧长度。</p> <p>0: 10 位 1: 11 位</p> <p><b>注意:</b> LINBDL 可用于 LIN 模式及其他模式下的断开帧的检测长度控制, 且检测长度和 LIN 模式相同。</p>
4	Reserved	保留, 必需保持复位值。
3:0	ADDR[3:0]	<p>USART 地址。</p> <p>在多处理器通信下的静默模式中使用的, 使用地址标识来唤醒某个 USART 设备。</p> <p>地址标识唤醒模式下 (USART_CTRL1.WUM=1), 如果接收到的数据帧低四位与 ADDR[3:0]值不相等, USART 就会进入静默模式; 如果接收到的数据帧低四位与 ADDR[3:0]值相等, USART 就会被唤醒。</p>

### 18.7.7 USART 控制寄存器 3 (USART\_CTRL3)

偏移地址: 0x14

复位值: 0x0000 0000



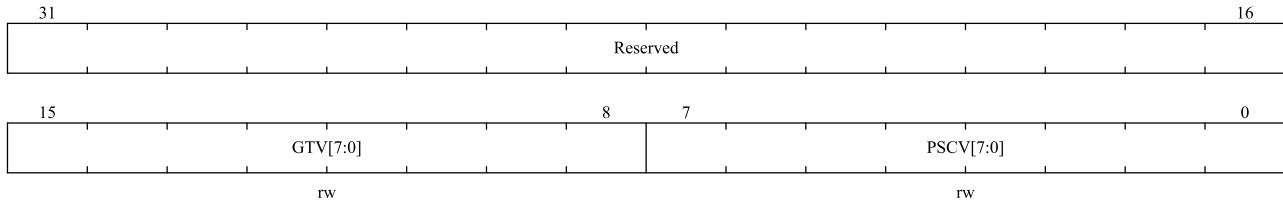
位域	名称	描述
31:11	Reserved	保留, 必需保持复位值。
10	CTSIEN	<p>CTS 中断使能 (CTS interrupt enable)。</p> <p>如果该位置 1, 当 USART_STS.CTSF 被置位时将产生中断。</p> <p>0: CTS 中断禁用。 1: CTS 中断使能。</p> <p><b>注意:</b> 该位对 UART3/4 无效。</p>
9	CTSEN	<p>CTS 使能 (CTS enable)。</p> <p>该位用于使能 CTS 硬件流控制功能。</p> <p>0: CTS 硬件流控制禁用。 1: CTS 硬件流控制使能。</p> <p><b>注意:</b> 该位对 UART3/4 无效。</p>
8	RTSEN	<p>RTS 使能 (RTS enable)。</p> <p>该位用于使能 RTS 硬件流控制功能。</p> <p>0: RTS 硬件流控制禁用。 1: RTS 硬件流控制使能。</p>

位域	名称	描述
		注意：该位对UART3/4 无效。
7	DMATXEN	DMA 发送使能 (DMA transmitter enable)。 0: DMA 发送模式禁用。 1: DMA 发送模式使能。
6	DMARXEN	DMA 接收使能 (DMA receiver enable)。 0: DMA 接收模式禁用。 1: DMA 接收模式使能。
5	SCMEN	智能卡模式使能 (Smart card mode enable)。 该位用于使能智能卡模式。 0: 智能卡模式禁用。 1: 智能卡模式使能。 注意：该位对UART3/4 无效。
4	SCNACK	在智能卡模式 NACK 使能 (Smart card NACK enable)。 该位用于智能卡模式在奇偶校验错误发生时使能发送 NACK。 0: 当出现校验错误时不发送 NACK。 1: 当出现校验错误时发送 NACK。 注意：该位对UART3/4 无效。
3	HDMEN	半双工模式使能 (Half-duplex mode enable)。 该位用于使能半双工模式。 0: 半双工模式禁用。 1: 半双工模式使能。
2	IRDALP	IrDA 低功耗模式 (IrDA low-power)。 该位用于为 IrDA 模式选择低功耗模式。 0: 正常模式。 1: 低功耗模式。
1	IRDAMEN	IrDA 模式使能 (IrDA mode enable)。 0: IrDA 禁用。 1: IrDA 使能。
0	ERRIEN	错误中断使能 (Error interrupt enable)。 当 DMA 接收模式 (USART_CTRL3.DMARXEN=1) 使能时，如果该位被置 1，USART_STS.FEF、USART_STS.OREF、USART_STS.NEF 被置位将产生中 断。 0: 错误中断禁用。 1: 错误中断使能。

## 18.7.8 USART 保护时间和预分频寄存器(USART\_GTP)

偏移地址： 0x18

复位值： 0x0000 0000



位域	名称	描述
31:16	Reserved	保留，必需保持复位值。
15:8	GTV[7:0]	<p>智能卡模式下的保护时间值（Guard time value）。</p> <p>该位域规定了以波特时钟为单位的保护时间。在智能卡模式下，需要这个功能。USART_STS.TXC 标志置位时间延时 GTV[7:0]个波特时钟周期。</p> <p><b>注意：该位对 UART3/4 无效。</b></p>
7:0	PSCV[7:0]	<p>预分频器值（Prescaler value）。</p> <p>在 IrDA 低功耗模式下，这些位用来设定将外设时钟（PCLK1/PCLK2）分频产生低功耗频率的分频系数。</p> <p>00000000: 保留 – 不要写入该值      00000001: 对源时钟 1 分频      ...      11111111: 对源时钟 255 分频</p> <p>在 IrDA 正常模式下，PSCV 只能设置成 00000001。</p> <p>在智能卡模式下，PSCV[4:0]用于设定外设时钟（APB1/APB2）生成智能卡时钟的分频系数。实际的分频系数为 PSCV[4:0]设定值的两倍。</p> <p>00000: 保留 – 不要写入该值      00001: 对源时钟 2 分频      00010: 对源时钟 4 分频      ...      11111: 对源时钟 62 分频</p> <p>在智能卡模式下，PSCV[7:5]保留。</p> <p><b>注意：该位对 UART3/4 无效。</b></p>

## 19 串行外设接口/内置音频总线 (SPI/I<sup>2</sup>S)

### 19.1 SPI/I<sup>2</sup>S 介绍

本模块是 SPI/I<sup>2</sup>S， 默认工作在 SPI 模式， 通过寄存器设置， 可以选择工作在 I<sup>2</sup>S 模式。

SPI 可以工作在主模式或从模式， 支持全双工和单工高速通讯模式， 并且具有硬件 CRC 计算能力且可配置多主模式。

I<sup>2</sup>S 可以工作在单工的主模式或从模式， 支持 4 种音频标准：飞利浦 I<sup>2</sup>S 标准、MSB 对齐标准、LSB 对齐标准和 PCM 标准。

这两种都是同步串行接口通讯协议。

### 19.2 SPI 和 I<sup>2</sup>S 主要特征

#### 19.2.1 SPI 特征

- 全双工和单工同步模式
- 支持主模式、从模式和多主模式
- 支持 8bit 或 16bit 数据帧格式
- 数据位顺序可编程
- 硬件或软件片选管理
- 时钟极性和时钟相位可配置
- 发送和接收支持硬件 CRC 计算及校验
- 支持 DMA 传输功能

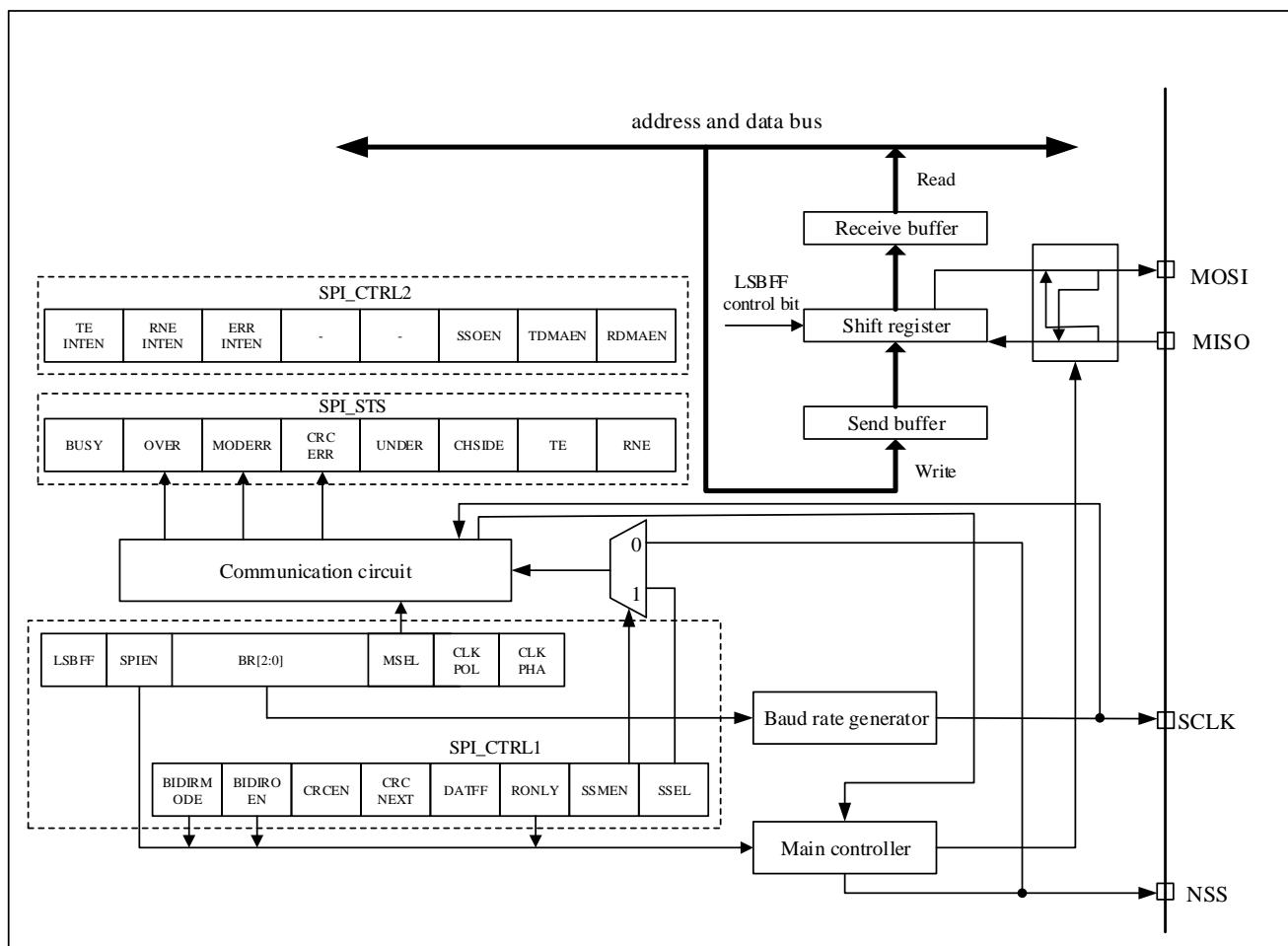
#### 19.2.2 I<sup>2</sup>S 特征

- 单工同步模式
- 支持主模式和从模式操作
- 4 种音频标准可以支持：飞利浦 I<sup>2</sup>S 标准、MSB 对齐标准、LSB 对齐标准和 PCM 标准
- 音频采样频率可配置，范围从 8KHz 到 96KHz
- 稳态时钟极性可配置
- 数据方向 MSB
- 支持 DMA 传输功能

## 19.3 SPI 功能描述

### 19.3.1 通用描述

图 19-1 SPI 框图



为了连接外部设备，SPI 接口有 4 个引脚与外设器件连接，具体如下：

- **SCLK:** 串行时钟引脚，该信号从主设备 SCLK 引脚输出，由从设备 SCLK 引脚输入
- **MISO:** 主输入/从输出引脚，数据从主设备的 MISO 引脚输入，由从设备的 MISO 引脚输出
- **MOSI:** 主输出/从输入引脚，数据从主设备的 MOSI 引脚输出，由从设备的 MOSI 引脚输入
- **NSS:** 片选引脚，有两种 NSS 引脚类型，外部引脚和内部引脚。如果内部引脚检测到高电平，SPI 工作在主模式，相反，SPI 工作在从模式。用户可以使用主设备的一个标准 I/O 引脚控制从设备的 NSS 引脚

#### 软件 NSS 模式

当 SPI\_CTRL1.SSMEN=1(图 19-2)，软件从设备管理被使能。

软件 NSS 模式时，不需要使用 NSS 引脚。在这种模式下，通过写 SPI\_CTRL1.SSEL 位（主模式 SSEL=1，从模式 SSEL=0），驱动内部 NSS 信号电平

## 硬件 NSS 模式

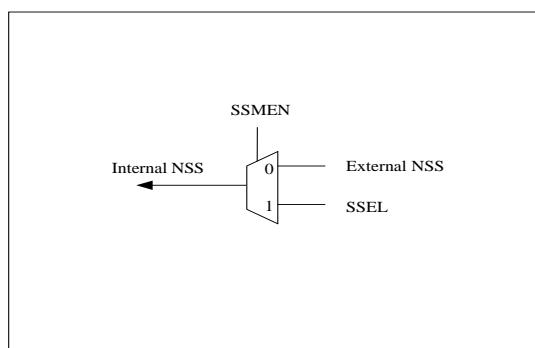
当 SPI\_CTRL1.SSMEN=0(图 19-2 硬件/软件的从选择管理), 软件从设备管理被禁能。

**输入模式:** 主设备的 NSS 输出被禁止 (SPI\_CTRL1.MSEL = 1, SPI\_CTRL2.SSOEN = 0), 允许操作在多主模式下。在整个数据帧传输期间主机应该连接 NSS 到高电平, 从机应该连接 NSS 到低电平。

**输出模式:** 主设备的 NSS 输出被使能 (SPI\_CTRL1.MSEL = 1, SPI\_CTRL2.SSOEN = 1), 主设备必须驱动 NSS 到低电平, 所有与主设备连接并且设置为硬件 NSS 模式的设备将会检测到低电平, 并自动进入从模式。当主设备的 NSS 没有被驱动到低电平, 设备进入从模式, 并产生主模式失效错误。

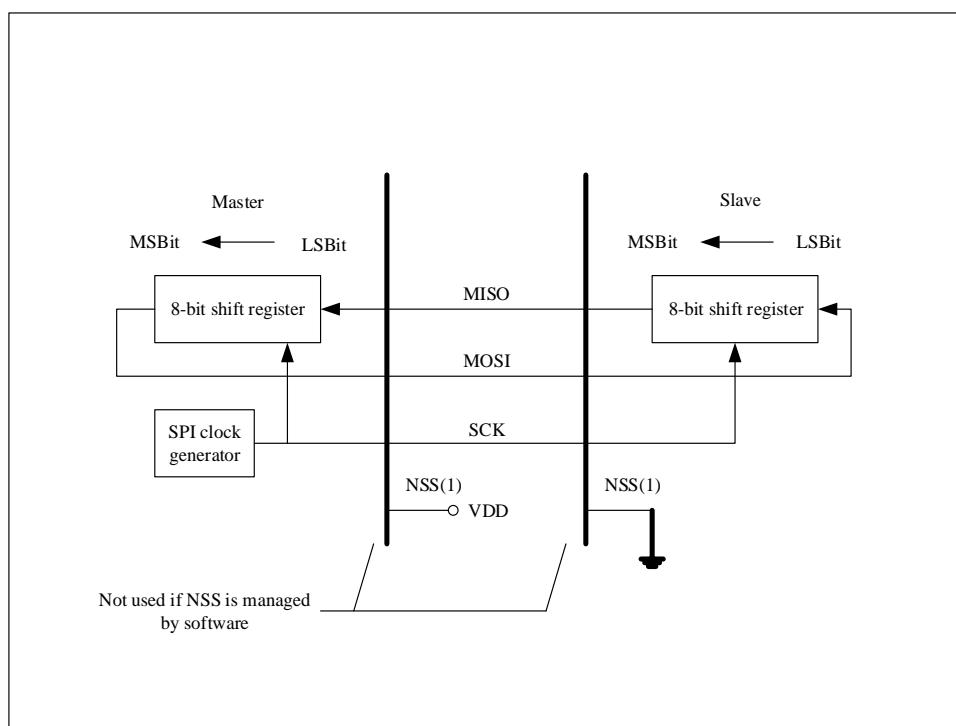
**注意:** 软件模式和硬件模式的选择取决于在通讯协议里是否需要对 NSS 控制, 如果不需要, 可以选择软件模式, 并且释放 GPIO 引脚作其他使用

图 19-2 硬件/软件的从选择管理



下图是单个主设备和单个从设备互联的例子。

图 19-3 单主和单从应用



**注意:** NSS 引脚被设置为输入。

SPI 是一个环形总线结构。主设备的 MOSI 引脚连接到从设备的 MOSI 引脚，并且主设备的 MISO 引脚连接到从设备的 MISO 引脚，以便数据可以在设备之间传输。主设备和从设备之间的连续数据传输，通过 MOSI 引脚发送数据到从设备，然而从设备的最高位被发送到主设备的最低位通过 MISO，当数据的第二位被发送，最低位的数据会左移一位并将新数据存入最低位。

### SPI 时序模式

通过设置 SPI\_CTRL1.CLKPOL 位和 SPI\_CTRL1.CLKPHA 位，用户可以选择数据捕获的时钟沿。

当 CLKPOL = 0, CLKPHA = 0，空闲时 SCLK 引脚将保持低电平，数据将在第一个时钟沿被采样，即上升沿。

当 CLKPOL = 0, CLKPHA = 1，空闲时 SCLK 引脚将保持低电平，数据将在第二个时钟沿被采样，即下降沿。

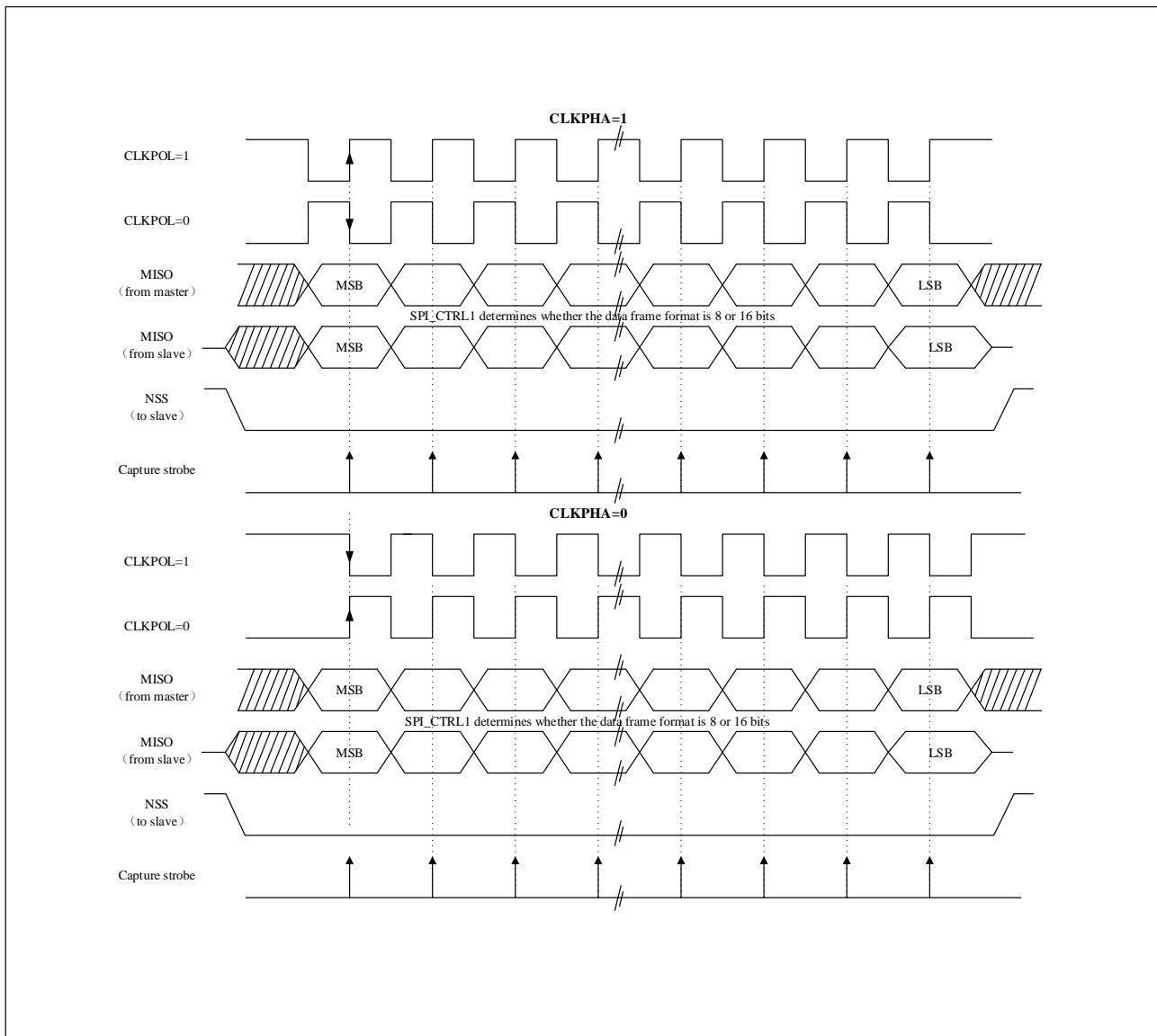
当 CLKPOL = 1, CLKPHA = 0，空闲时 SCLK 引脚将保持高电平，数据将在第一个时钟沿被采样，即下降沿。

当 CLKPOL = 1, CLKPHA = 1，空闲时 SCLK 引脚将保持高电平，数据将在第二个时钟沿被采样，即上升沿。

不管选择哪种时序模式，主设备和从设备的时序模式配置必须相同。

图 19-4 数据时钟时序图是当 SPI\_CTRL1.LSBFF=0 时，SPI 传输的 4 种 CLKPHA 和 CLKPOL 位组合时序。

图 19-4 数据时钟时序图



### 数据格式

通过设置 SPI\_CTRL1.LSBFF 位，用户可以选择数据的位顺序，当 LSBFF=0，SPI 将先发送数据的高位 (MSB)，当 LSBFF=1，SPI 将先发送数据的低位 (LSB)。

通过设置 SPI\_CTRL1.DATFF 位，用户可以选择数据帧格式。

### 19.3.2 SPI 工作模式

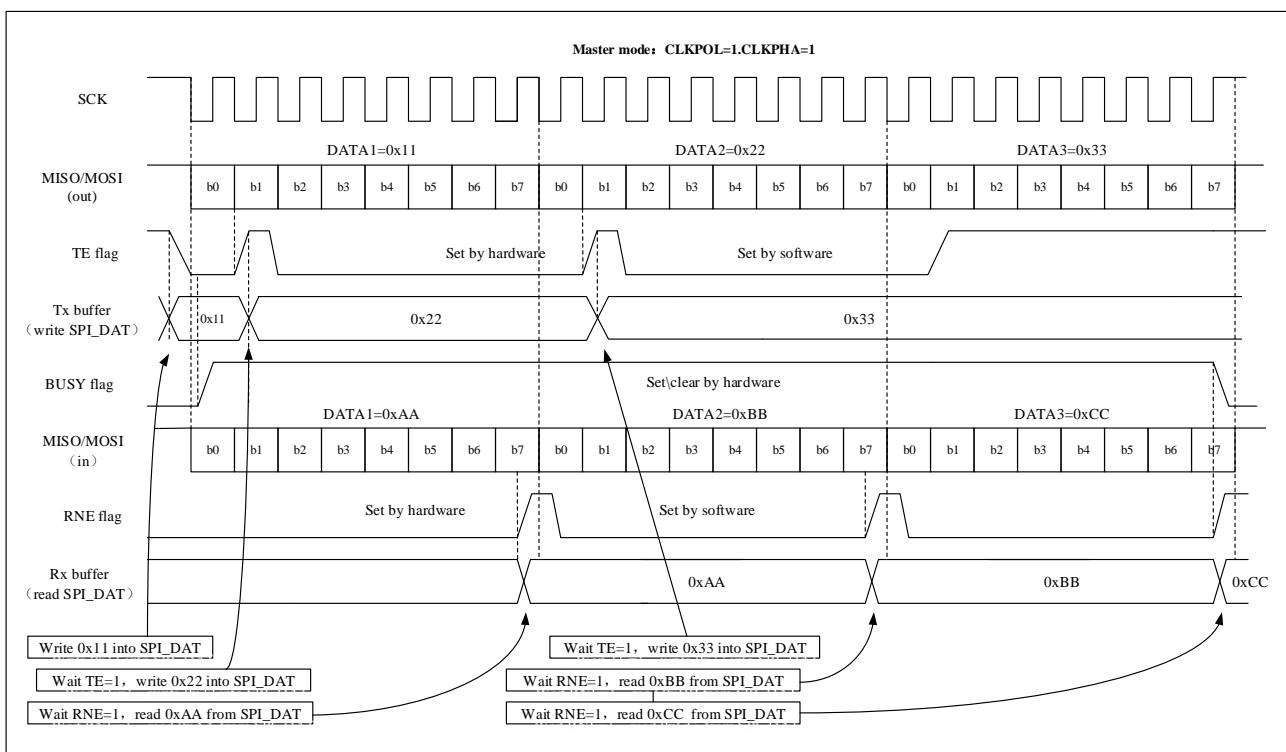
#### ■ 主机全双工模式 (MSEL=1, BIDIRMODE=0, RONLY=0)

第一个数据被写到SPI\_DAT寄存器后，将会开始传输，数据第一个位被发送时，数据字节并行从数据寄存器装载进入移位寄存器，然后数据位按照SPI\_CTRL1.LSBFF位的配置，数据位按照MSB或LSB顺序被串行移位进入MOSI引脚。与此同时，在MISO引脚上接收到的数据，按照同样顺序被串行地移位进入移位寄存器，然后并行装载入SPI\_DAT寄存器。

1. 设置SPI\_CTRL1.SPIEN位为1，使能SPI模块；
2. 写待发送的第一个数据到SPI\_DAT（这个写操作会清除SPI\_STS.TE标志位）；
3. 等待SPI\_STS.TE标志位置1后，再写入第二个待发送的数据到SPI\_DAT寄存器，等待SPI\_STS.RNE标志位置1后，读取SPI\_DAT寄存器获得第一个接收的数据，读取SPI\_DAT寄存器，SPI\_STS.RNE标志位会清0。重复上述操作，发送后续的数据，同时接收第n-1个数据；
4. 等待SPI\_STS.RNE置1后，读取最后一个数据；
5. 等待SPI\_STS.TE标志位置1，等待SPI\_STS.BUSY标志位清除后再关闭SPI模块。

数据的发送和接收处理可以在 SPI\_STS.RNE 标志位或 SPI\_STS.TE 标志位的上升沿产生的中断处理程序中实现。

图 19-5 主机全双工模式下连续传输时，SPI\_STS.TE/RNE/BUSY 的变化示意图



## ■ 主机双线单向仅发送模式 (MSEL=1, BIDIRMODE=0, RONLY=0)

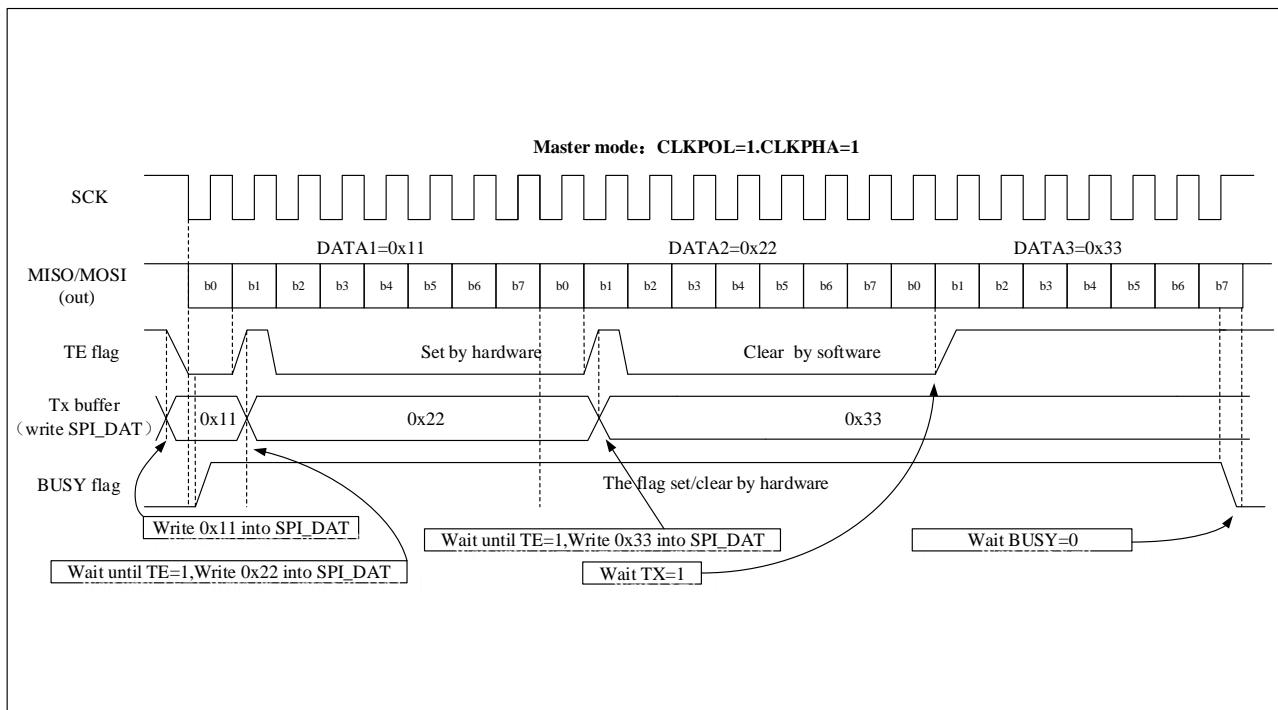
双线单向仅发送模式和全双工模式相似，但是在双线单向仅发送模式，接收的数据将不会被读取，因此 SPI\_STS.OVER 标志位将会置位，软件应该忽略这个位。软件操作流程如下(图 19-6 主机单向只发送模式下连续传输时，SPI\_STS.TE/BUSY 变化示意图)：

1. 设置SPI\_CTRL1.SPIEN位为1，使能SPI模块；
2. 写待发送的第一个数据到 SPI\_DAT 寄存器（该操作会清除 SPI\_STS.TE 标志位）；
3. 等待 SPI\_STS.TE 标志位置 1，写待发送的第二个数据到 SPI\_DAT 寄存器，重复这个操作发送后续的数据；
4. 写最后一个数据到 SPI\_DAT 寄存器，等待 SPI\_STS.TE 标志位置 1，然后等待 SPI\_STS.BUSY 位清除，

完成所有数据的发送。

数据发送可以在 SPI\_STS.TE 标志位上升沿产生的中断处理程序里实现。

图 19-6 主机单向只发送模式下连续传输时，SPI\_STS.TE/BUSY 变化示意图



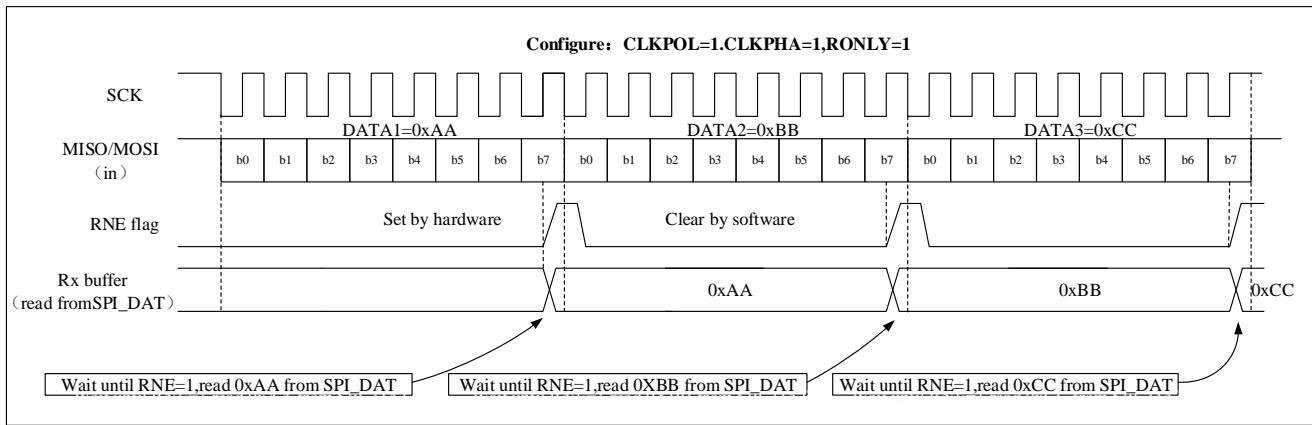
#### ■ 主机双线单向仅接收模式 (MSEL=1, BIDIRMODE=0, RONLY=1)

当 SPI\_CTRL1.SPIEN=1，开始接收过程。来自 MISO 引脚的数据位依次连续移位进入 8bit 移位寄存器，然后并行传送数据到 SPI\_DAT 寄存器。软件操作流程如下（见图 19-7 只接收模式 (BIDIRMODE=0 且 RONLY=1) 下连续传输时，RNE 变化示意图）：

1. 设置 SPI\_CTRL1.RONLY=1，使能仅接收模式；
2. 主机模式下，设置 SPIEN 位为 1，使能 SPI 模块，SCLK 信号会立即产生，在 SPI 关闭前 (SPIEN=0)，数据连续被接收。从机模式下，当主设备驱动 NSS 信号低电平并且产生 SCLK，数据持续被接收；
3. 等待 SPI\_STS.RNE 位置 1，读取 SPI\_DAT 寄存器获得接收的数据，当读取 SPI\_DAT 寄存器，SPI\_STS.RNE 位将会清除。重复这个操作接收所有数据。

数据处理可以在 SPI\_STS.RNE 标志位产生的中断处理程序里实现。

图 19-7 只接收模式 (BIDIRMODE=0 且 RONLY=1) 下连续传输时, RNE 变化示意图



#### ■ 主机单线双向发送模式 (MSEL = 1, BIDIRMODE = 1, BIDIROEN = 1, RONLY = 0)

数据写进SPI\_DAT寄存器后，传输过程开始。这个模式不接收数据。发送第一个数据位的同时，被发送的数据并行装载进8位移位寄存器，然后根据LSBFF位的配置，SPI按照MSB或LSB顺序将数据位串行移位到MOSI引脚。

主机单线双向发送的软件操作流程和仅发送模式的流程相同。

#### ■ 主机单线双向接收模式 (MSEL = 1, BIDIRMODE = 1, BIDIROEN = 0, RONLY = 0)

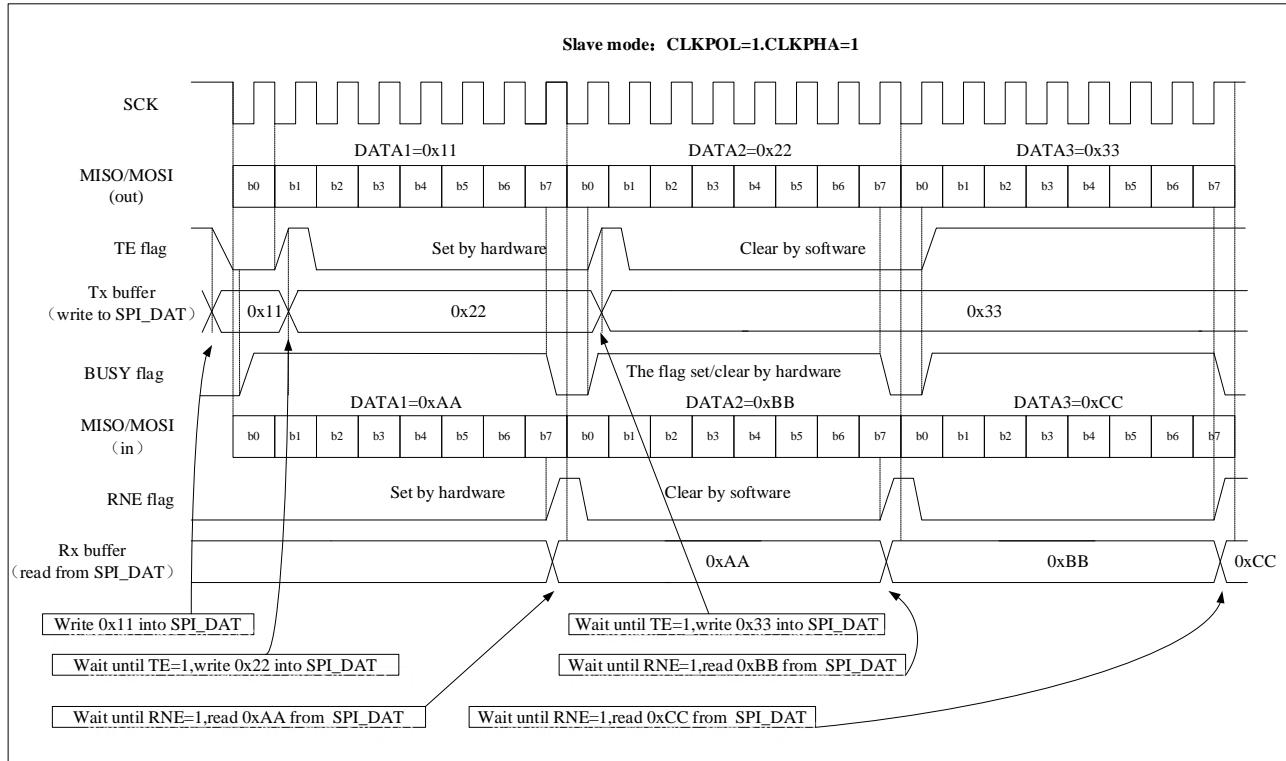
该模式下，当SPI使能 (SPI\_CTRL1.SPIEN=1)，接收过程开始。该模式下，没有数据输出，接收到的数据位顺序且连续移位进入8位移位寄存器，并行的传输进SPI\_DAT寄存器（接收缓存）。

主机单线双向接收模式的软件操作流程和仅接收模式一样。

#### ■ 从机全双工模式 (MSEL=0, BIDIRMODE=0 and RONLY=0)

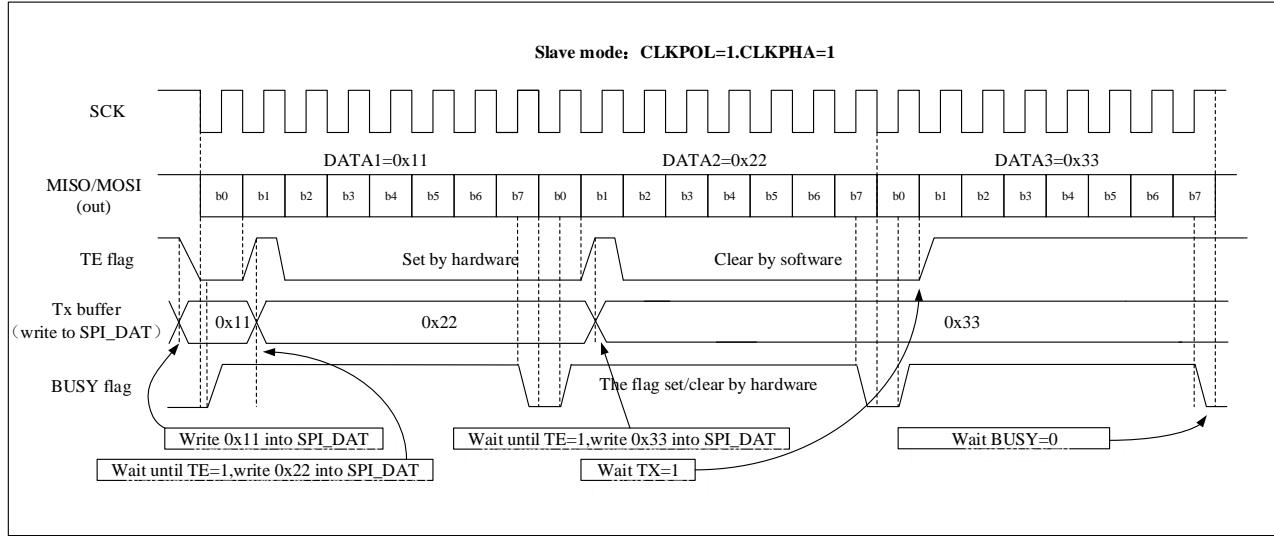
当从设备接收到第一个时钟沿，数据传输过程开始。主设备开始数据传输之前，软件必须确保待发送的数据写入 SPI\_DAT 寄存器。

图 19-8 从机全双工模式下连续传输时, SPI\_STS.TE/RNE/BUSY 的变化示意图



### ■ 从机双线单向仅发送模式 (MSEL=0, BIDIRMODE=0 and RONLY=0)

图 19-9 从机单向只发送模式下连续传输时, SPI\_STS.TE/BUSY 变化示意图



### ■ 从机双线单向仅接收模式 (MSEL=0, BIDIRMODE=0 and RONLY=1)

当从设备接收到时钟信号和来自MOSI引脚的第一个数据位, 数据接收过程开始。接收到的数据位顺序且连续地串行移位到8位移位寄存器, 然后并行地装载到SPI\_DAT寄存器(接收缓存)。

### ■ 从机单线双向发送模式 (MSEL=0, BIDIRMODE=1 and BIDIROEN=1)

当从设备接收到第一个时钟沿，数据发送过程开始。该模式没有数据接收，SPI主机开始数据传输前，软件必须确保待发送数据已经被写进SPI\_DAT寄存器。

### ■ 从机单线双向接收模式（MSEL=0,BIDIRMODE=1 and BIDIROEN=0）

当从设备接收到第一个时钟沿和来自MOSI引脚的数据位时，数据接收开始。该模式没有数据输出，接收到的数据位顺序且连续地串行移位到8位移位寄存器，然后并行地装载到SPI\_DAT寄存器（接收缓存）。

注意：从机的软件操作流程参考主机的。

## SPI 初始化流程

1. 通过设置 SPI\_CTRL1.BR[2:0]位配置数据传输的波特率；
2. 选择时钟极性（SPI\_CTRL1.CLKPOL）和时钟相位（SPI\_CTRL1.CLKPHA），定义数据传输和时钟的相位关系；
3. 设置 SPI\_CTRL1.DATFF 位定义帧格式为 8bit 还是 16bit；
4. 配置 SPI\_CTRL1.LSBFF 定义数据位发送的顺序是 LSB 还是 MSB；
5. 配置 NSS 模式；
6. 配置 SPI\_CTRL1.MSEL、SPI\_CTRL1.BIDIRMODE、SPI\_CTRL1.BIDIROEN 和 SPI\_CTRL1.RONLY 位；
7. 设置 SPI\_CTRL1.SPIEN 位使能 SPI 模块。

## SPI 协议基本的发送和接收处理

当 SPI 发送 1 个数据帧，首先，数据帧从数据缓存装载进移位寄存器，然后装载的数据被发送。当来自发送缓存的数据传输进移位寄存器，发送缓存为空，SPI\_STS.TE 标志位置 1，然后下一个数据可装载进入发送缓存。如果 SPI\_CTRL2.TEINTEN 位置 1，中断将会产生。写 SPI\_DAT 寄存器可以对 SPI\_STS.TE 标志位清 0。

采样时钟的最后一个边沿，当数据从移位寄存器传输进接收缓存，SPI\_STS.RNE 标志位置 1，数据准备就绪，可以从 SPI\_DAT 寄存器读取。如果 SPI\_CTRL2.RNEINTEN 标志位置 1，中断将会产生。读 SPI\_DAT 寄存器可以对 SPI\_STS.RNE 标志位清 0。

主模式下，当数据写进发送缓存，发送过程开始。当前数据帧发送完成前，如果下个数据写进 SPI\_DAT 寄存器，连续发送可以实现。

从机模式下，NSS 引脚为低，当第一个时钟沿到来，发送过程开始。为了避免意外的数据传输，数据发送前（主机发送时钟前，建议先使能 SPI 模块），软件必须写数据到发送缓存。

在有些配置里，当发送最后数据时，SPI\_STS.BUSY 标志位可以用于等待数据发送结束。

## 连续和非连续传输

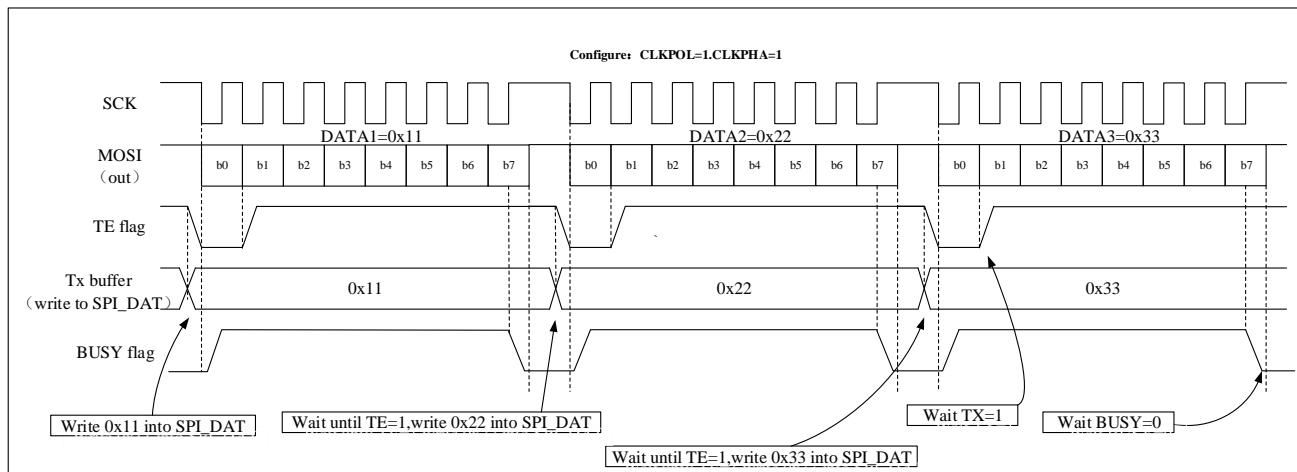
当主模式下发送数据，如果软件足够快，检测到每个 TE 上升沿（或 TE 中断），且正进行的传输结束前，立即将数据写入 SPI\_DAT 寄存器，此时，每个数据项之间的 SPI 时钟保持连续，SPI\_STS.BUSY 标志位将不会被清除，连续通讯可以实现。

如果软件不够快，将导致不连续的通讯，此时，每个数据传输之间 SPI\_STS.BUSY 标志位会被清除（见图 19-10 BIDIRMODE=0, RONLY=0 非连续传输发送时，SPI\_STS.TE/BUSY 变化示意图）。

主机仅接收模式下 (SPI\_CTRL1.RONLY=1), 通讯总是连续的, 并且 BUSY 标志位总是为高。

从模式下, 通讯的连续性由主设备决定, 任何情况下, 即使通讯是连续的, 每个数据项之间, BUSY 标志位将至少有 1 个 SPI 时钟周期为低 (见图 19-9 从机单向只发送模式下连续传输时, SPI\_STS.TE/BUSY 变化示意图)

图 19-10 BIDIRMODE=0, RONLY=0 非连续传输发送时, SPI\_STS.TE/BUSY 变化示意图



### 19.3.3 状态标志

SPI\_STS 寄存器有 3 个标志位监控 SPI 的状态。

#### 发送缓存空标志位 (TE)

当发送缓存空, TE 标志位置 1, 意味着可以将新数据写进 SPI\_DAT 寄存器。当发送缓存非空, 该标志位将硬件清 0。

#### 接收缓存非空标志位 (RNE)

当接收缓存非空, RNE 标志位置 1, 因此用户知道接收缓存有数据。读取 SPI\_DAT 寄存器后, 该标志位将硬件清 0。

#### 忙标志位 (BUSY)

当传输开始, BUSY 标志位置 1, 传输结束后 BUSY 标志位硬件清 0。

仅当设备在主机单线双向接收模式, 当通讯进行中, BUSY 标志位将会设置为 0。

下面情况, BUSY 标志位将会清 0:

- 传输结束 (主机模式下连续通讯除外)
- 关闭 SPI 模块 (SPI\_CTRL1.SPIEN=0)
- 主机模式错误发生 (SPI\_STS.MODERR=1)

当通讯是不连续的: 每个数据项传输之间, BUSY 标志位清 0。

当通讯是连续的: 在主机模式, 整个传输过程, BUSY 标志位保持为高。在从机模式, 每个数据项传输之间 BUSY 标志位会有 1 个 SPI 时钟周期为低。因此不要使用 BUSY 标志位处理每个数据项的发送和接收。

### 19.3.4 关闭 SPI

为了关闭 SPI 模块，不同的操作模式需要采用不同的操作步骤：

#### 在主机或从机全双工模式（BIDIRMODE=0, RONLY=0）

1. 等待 SPI\_STS.RNE 标志位置 1，并且接收到最后一个字节；
2. 等待 SPI\_STS.TE 标志位置 1；
3. 等待 SPI\_STS.BUSY 标志位清 0；
4. 关闭 SPI 模块（SPI\_CTRL1.SPIEN=0）。

#### 在主机或从机单向发送模式（BIDIRMODE=0, RONLY=0）或双向只发送模式（BIDIMODE=1, BIDIROEN=1）

1. 向 SPI\_DAT 寄存器写完最后一个字节后，等待 SPI\_STS.TE 标志位置 1；
2. 等待 SPI\_STS.BUSY 标志位清 0；
3. 关闭 SPI 模块（SPI\_CTRL1.SPIEN=0）。

#### 在主机单向只接收模式（MSEL=1, BIDIRMODE=0, RONLY=1）或主机双向只接收模式（MSEL=1, BIDIRMODE=1, BIDIROEN=0）

1. 等待倒数第二个 SPI\_STS.RNE 置 1；
2. 关闭 SPI 模块前（SPI\_CTRL1.SPIEN=0），等待 1 个 SPI 时钟周期（使用软件延时）；
3. 进入关机模式前（或关闭 SPI 模块时钟），等待最后一个 SPI\_STS.RNE 置 1。

#### 从机单向只接收模式（MSEL=0, BIDIRMODE=0, RONLY=1）或双向接收模式（MSEL=0, BIDIRMODE=1, BIDIROEN=0）

1. 可以在任意时间关闭 SPI 模块（SPI\_CTRL1.SPIEN = 0），并且当前传输结束后，SPI 模块将被关闭；
2. 如果想进入关机模式，进入关机模式之前（或关闭 SPI 模块时钟），必须等待 SPI\_STS.BUSY 标志位为 0。

### 19.3.5 使用 DMA 进行 SPI 通讯

用户可以选择 DMA 进行 SPI 数据传输，应用程序可以得到释放，系统效率可以大大提升。

当发送缓存 DMA 使能（SPI\_CTRL2.TDMAEN 位置 1），每次 SPI\_STS.TE 标志位置 1，会产生 DMA 请求，DMA 自动将数据写入 SPI\_DAT 寄存器，这将会清除 TE 标志位。当接收缓存 DMA 使能（SPI\_CTRL2.RDMAEN 位置 1），每次 SPI\_STS.RNE 标志位置 1，会产生 DMA 请求，DMA 自动读取 SPI\_DAT 寄存器，这将会清除 SPI\_STS.RNE 标志位。

当 SPI 仅用于数据发送，仅需要使能 SPI 的发送 DMA 通道。这时，接收数据没有读取，SPI\_STS.OVER 标志位置 1，软件不需要处理这个标志。

当 SPI 仅用于数据接收，仅 SPI 的接收 DMA 通道需要使能。

在发送模式，DMA 已经发送完所有待发送的数据后（DMA\_INTSTS.TXCF 标志位变为 1），SPI\_STS.BUSY 标志位可以被监视确认 SPI 通讯结束，这样可以避免当 SPI 关闭或进入停机模式，破坏最后数据的发送。因

此，软件需要等待 SPI\_STS.TE 标志位置 1，并且等待 SPI\_STS.BUSY 标志位为 0。

图 19-11 使用 DMA 发送

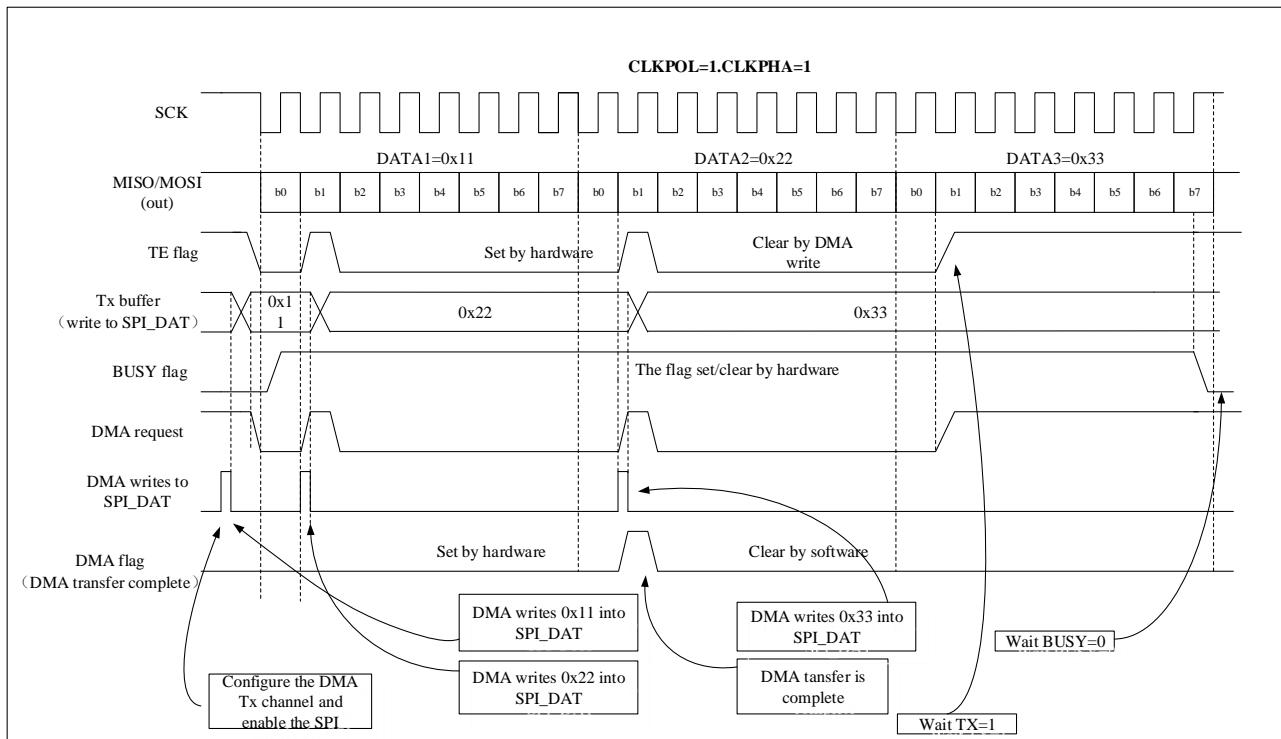
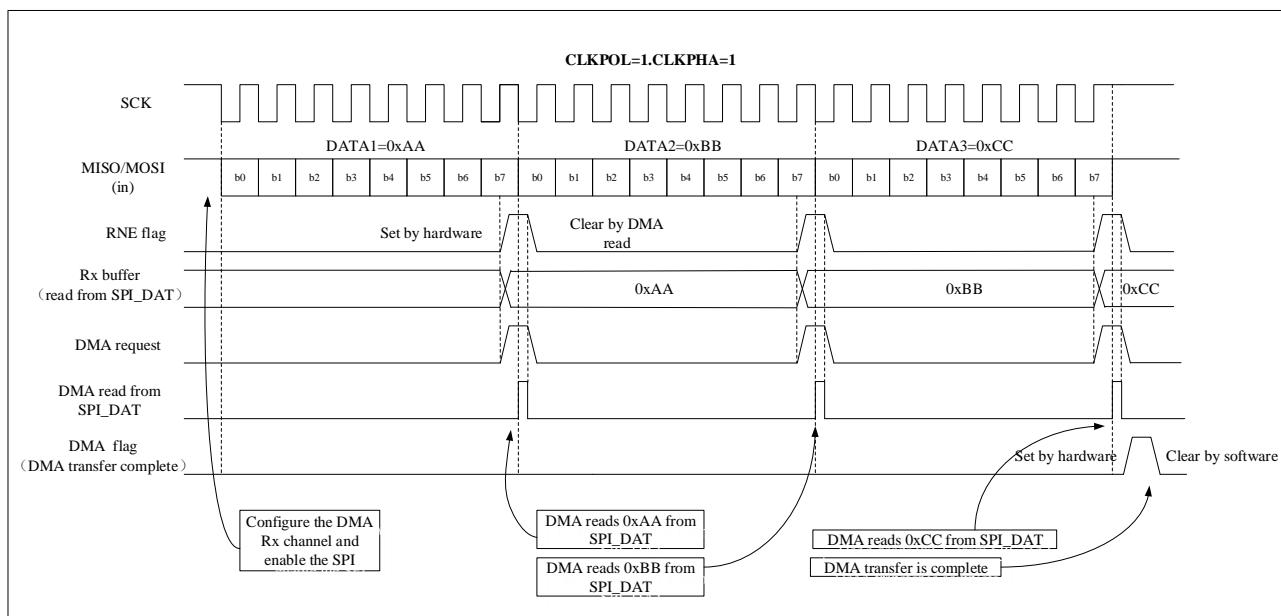


图 19-12 使用 DMA 接收



### 19.3.6 CRC 计算

SPI 包含两个独立的 CRC 计算器，用于数据发送和数据接收，以确保数据传输的正确性。根据发送和接收数据帧格式，CRC 采用不同的计算方法，8 位数据帧格式采用 CRC8，16 位数据帧格式采用 CRC16。SPI

CRC 计算使用的多项式由 SPI\_CRCPOLY 寄存器设置，用户设置 SPI\_CTRL1.CRCEN 位使能 CRC 计算。

在发送模式，最后的数据写进发送缓存后，设置 SPI\_CTRL1.CRCNEXT 位为 1，这指示发送完数据后硬件将开始发送 CRC 值 (SPI\_CRCTDAT 值)。发送 CRC 时，CRC 计算将停止。

在接收模式，倒数第二个数据帧接收到后，设置 SPI\_CTRL1.CRCNEXT 位为 1。接收到的 CRC 和 SPI\_CRCRDAT 值进行比较，如果他们不同，SPI\_STS.CRCERR 位置 1，当 SPI\_CTRL2.ERRINTEN 位置 1，中断产生。

为了保持主设备-从设备下次的 CRC 计算结果的同步，用户应清除主设备-从设备的 CRC 值。SPI\_CTRL1.CRCEN 位置 1 会复位 SPI\_CRCRDAT 寄存器和 SPI\_CRCTDAT 寄存器。按顺序采用下面步骤：SPI\_CTRL1.SPIEN=0; SPI\_CTRL1.CRCEN=0; SPI\_CTRL1.CRCEN=1; SPI\_CTRL1.SPIEN=1。

最重要的是，当 SPI 配置为从模式且 CRC 使能，只要 SCLK 引脚有时钟脉冲，即使 NSS 引脚为高，CRC 计算将仍然被执行。这种情况常见于当主设备和多个从设备交替通讯时，因此这需要避免 CRC 误操作。

当 SPI 硬件 CRC 检查使能 (SPI\_CTRL1.CRCEN=1) 且 DMA 使能，通讯结束时硬件自动完成 CRC 字节发送和接收。

### 19.3.7 错误标志位

#### 主模式失效位 (MODERR)

以下两种情况将会导致主机失效错误：

- NSS 引脚硬件管理模式，主设备 NSS 引脚被驱动低电平；
- NSS 引脚软件管理模式，SSEL 位被设置为 0。

当主模式失效错误发生，SPI\_STS.MODERR 标志位置 1。如果用户允许相应的中断，则产生中断。SPI\_CTRL1.SPIEN 位和 SPI\_CTRL1.MSEL 将写保护，且硬件清除。SPI 关闭且强制进入从模式。

软件执行 SPI\_STS 寄存器读或写操作，然后写 SPI\_CTRL1 寄存器可以清除 SPI\_STS.MODERR 位（在多主配置下，主机的 NSS 引脚必须先拉高）。

通常，从机的 SPI\_STS.MODERR 位不能设置为 1。然而，在多主配置下，从设备的 SPI\_STS.MODERR 位可能置位。这种情况下，SPI\_STS.MODERR 位指示存在多主冲突。中断程序可以执行复位或返回默认状态从错误状态恢复。

#### 上溢错误 (OVER)

当 SPI\_STS.RNE 位置 1，但是仍然有数据发送进入接收缓存，上溢错误将发生，此时，上溢标志 SPI\_STS.OVER 置 1。如果用户使能相应的中断，则产生中断。所有接收到的数据丢失，且 SPI\_DAT 寄存器仅保留之前未读的数据。

依次读 SPI\_DAT 寄存器和 SPI\_STS 寄存器可以清除 SPI\_STS.OVER 位。

#### CRC 错误 (CRCERR)

CRC 错误标志用于检查接收数据的有效性。当接收到的 CRC 值和 SPI\_CRCRDAT 值不匹配，CRC 错误发生。

### 19.3.8 SPI 中断

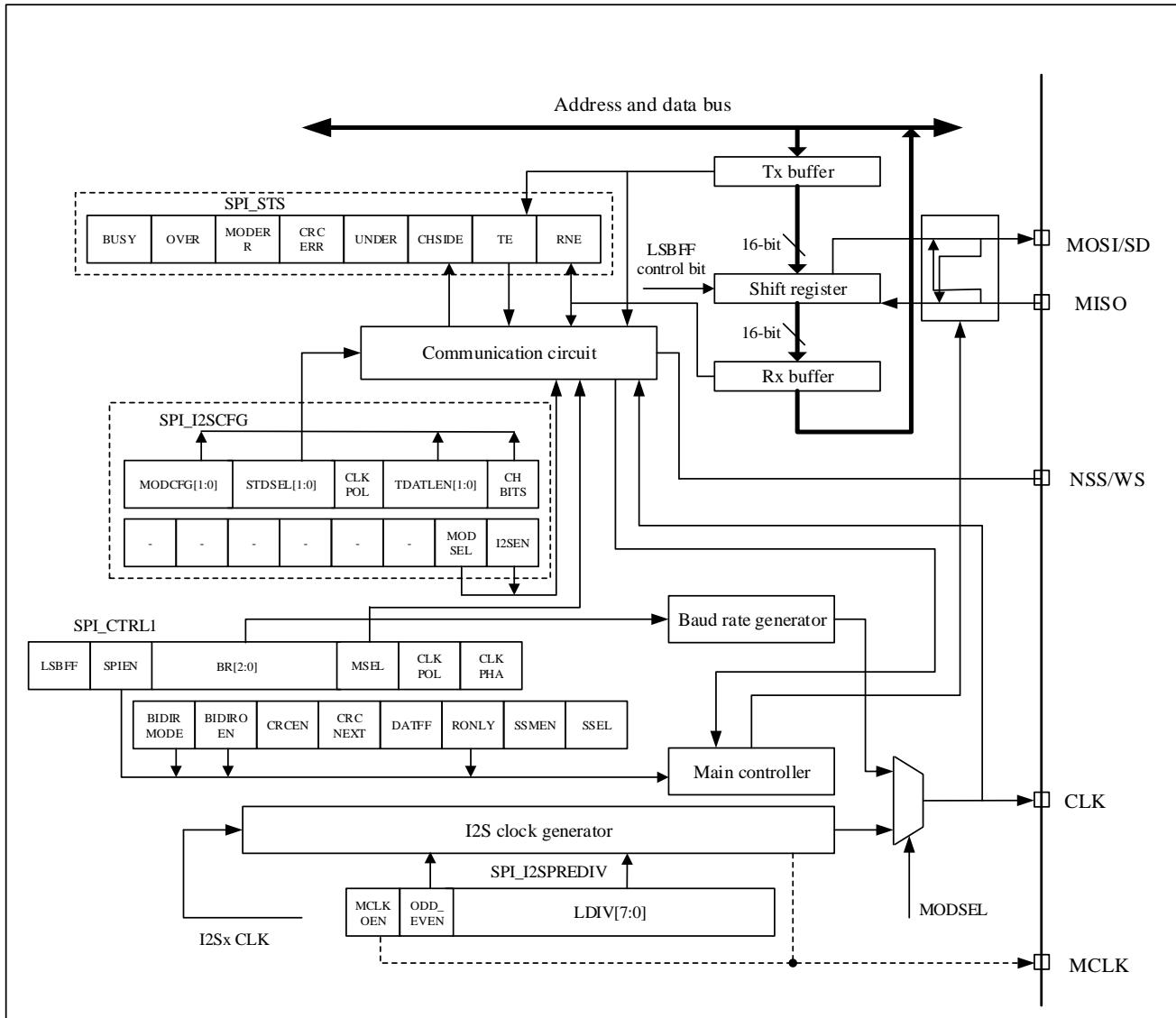
表 19-1 SPI 中断请求

中断事件	事件标志位	使能控制位
发送缓存空标志	TE	TEINTEN
接收缓存非空标志	RNE	RNEINTEN
主模式失效事件	MODERR	ERRINTEN
上溢错误	OVER	
CRC 错误标志	CRCERR	

## 19.4 I<sup>2</sup>S 功能描述

I<sup>2</sup>S 的框图如下图显示：

图 19-13 I<sup>2</sup>S 框图



I<sup>2</sup>S 接口使用和 SPI 接口相同的引脚、标志和中断。SPI\_I2SCFG.MODSEL 位置 1 选择 I<sup>2</sup>S 音频接口。

I<sup>2</sup>S 总共有 4 个引脚，其中 3 个引脚与 SPI 共享：

- CLK：串行时钟（和 SCLK 引脚共享），每次 1 位音频数据发送，CLK 产生一个脉冲。
- SD：串行数据（和 MOSI 引脚共享），用于数据发送和接收；
- WS：通道选择（和 NSS 引脚共享），主模式下用作数据控制信号输出，从模式下用作输入；
- MCLK：主机时钟（独立映射，可选），输出  $256 \times F_s$  时钟信号，保证系统间更好的同步。

注意： $F_s$  是音频信号的采样频率

在主模式下，I<sup>2</sup>S 使用自己的时钟发生器产生时钟信号用于通讯，这个时钟发生器也是主时钟输出的时钟源（SPI\_I2SPREDIV.MCLKOEN 位置'1'，主时钟输出使能）。

### 19.4.1 支持的音频协议

通过设置 SPI\_I2SCFG.STDSEL[1:0]位，可以选择 4 种音频标准

- I<sup>2</sup>S 飞利浦标准
- MSB 对齐标准
- LSB 对齐标准
- PCM 标准

左声道和右声道的音频数据通常是时分复用的，左声道总是先于右声道发送数据。通过检查 SPI\_STS.CHSIDE 位，用户可以区分接收到的数据属于哪个声道。然而，PCM 音频标准里，SPI\_STS.CHSIDE 位是没有意义的。

通过设置 SPI\_I2SCFG.TDATLEN 位，用户可以设置待传输的数据长度，通过设置 SPI\_I2SCFG.CHBITS 位，设置通道的数据位宽。下面有 4 种数据格式发送数据：

- 16 位数据打包成 16 位的数据帧
- 16 位数据打包成 32 位的数据帧（前面的 16 位是有意义的数据，后面的 16 位数据被硬件设置为 0）
- 24 位数据打包成 32 位数据帧（前面的 24 位数据是有意义的数据，后面 8 位数据被硬件设置为 0）
- 32 位的数据打包成 32 位数据帧

I<sup>2</sup>S 使用和 SPI 相同的 SPI\_DAT 寄存器发送和接收 16 位宽的数据。如果 I<sup>2</sup>S 需要发送或接收 24 位或 32 位宽数据，CPU 需读或写 SPI\_DAT 寄存器 2 次。另一方面，当 I<sup>2</sup>S 发送或接收 16 位宽数据，CPU 仅需读或写 SPI\_DAT 寄存器一次。

不管采用哪个数据格式和通讯标准，I<sup>2</sup>S 总是先发送数据高位（MSB）。

#### I<sup>2</sup>S 飞利浦标准

采用 I<sup>2</sup>S 飞利浦标准，发送数据的设备在时钟下降沿改变数据，接收数据的设备在时钟上升沿采样数据。WS 信号在第一个数据位（MSB）发送前一个时钟应有效，时钟信号下降沿将变化。

图 19-14 I<sup>2</sup>S 飞利浦协议波形 (16/32 位全精度, CLKPOL = 0)

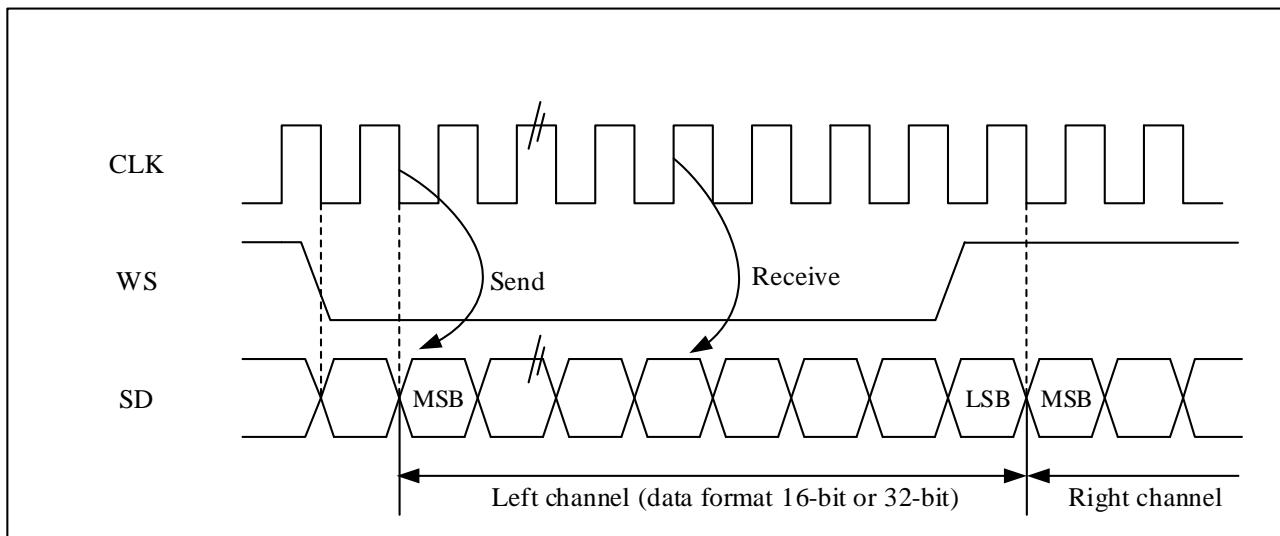
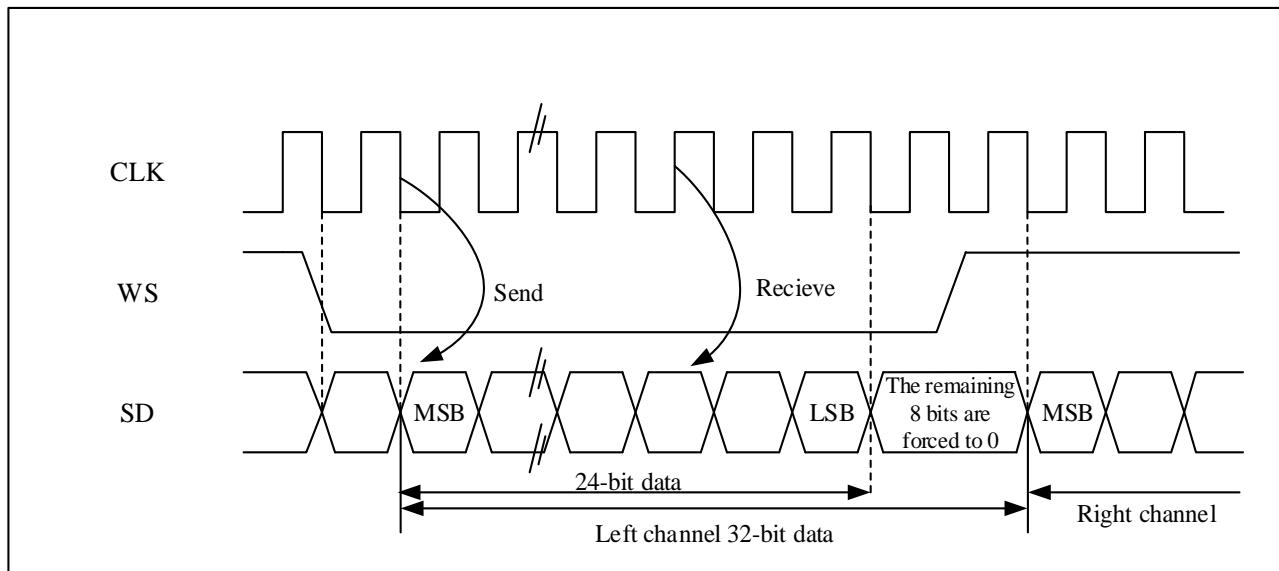
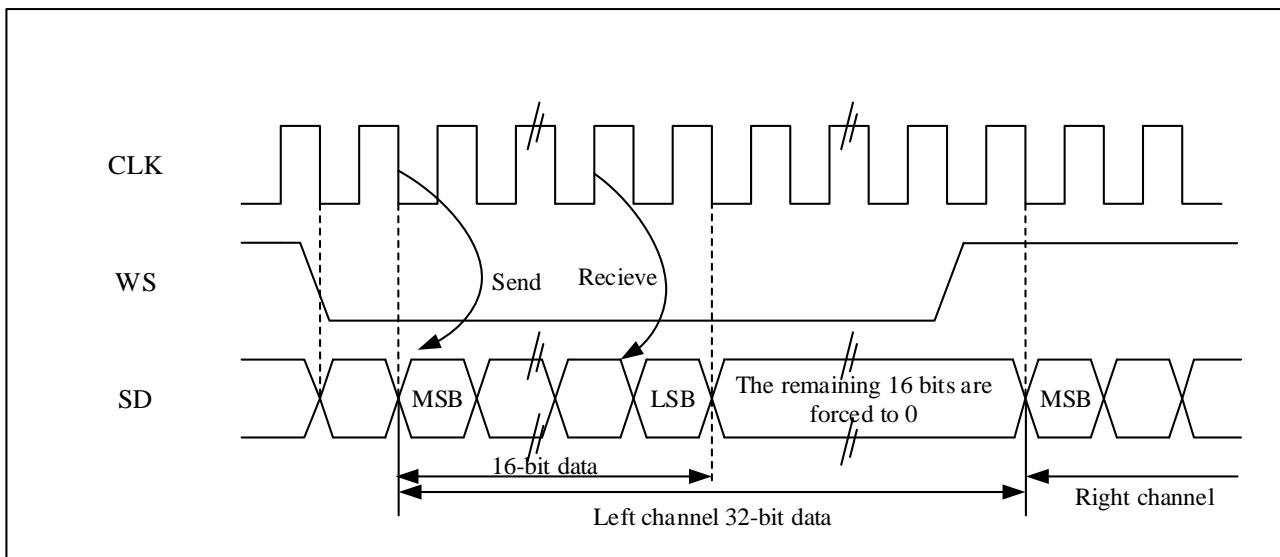


图 19-15 I<sup>2</sup>S 飞利浦协议标准波形 (24 位帧, CLKPOL = 0)



如果 24 位数据需要打包成 32 位数据帧格式，每帧数据传输时，CPU 需要读或写 SPI\_DAT 寄存器 2 次。例如，如果用户发送 24 位数据 0x95AA66，CPU 将首先写 0x95AA 进 SPI\_DAT 寄存器，然后再写 0x66XX 进 SPI\_DAT 寄存器（仅高 8 位数据有效，低 8 位数据是无意义的，可以是任何值）；如果用户接收 24 位数据 0x95AA66，CPU 将首先读 SPI\_DAT 寄存器得到 0x95AA，然后再读 SPI\_DAT 寄存器得到 0x6600（仅高 8 位数据有效，低 8 位数据总是 0）。

图 19-16 I<sup>2</sup>S 飞利浦协议标准波形（16 位扩展至 32 位包帧，CLKPOL=0）



如果 16 位数据需要打包进 32 位数据帧格式，每帧数据传输时，CPU 仅需要读或写 SPI\_DAT 寄存器一次。用于扩展到 32 位的低 16 位数据总是设置为 0x0000。例如，如果用户发送或接收 16 位的数据 0x89C1（扩展到 32 位数据是 0x89C10000）。数据发送过程中，高 16 位半字（0x89C1）需要写进 SPI\_DAT 寄存器；直到 SPI\_STS.TE 位置 1，用户可以写入新的数据。如果用户使能相应的中断，则中断产生。发送由硬件执行，即使最后 16 位（0x0000）没有发送，硬件将设置 SPI\_STS.TE 位为 1，且产生相应的中断。接收数据过程，每次设备收到高 16 位半字（0x89C1）后，SPI\_STS.RNE 标志位将置 1。如果用户使能相应的中断，则中断产生。这样，在 2 次读和写之间 CPU 有更多时间，且可以防止上溢或下溢的情况发生。

### MSB 对齐标准

在 MSB 对齐标准里，发送数据的设备将在时钟下降沿改变数据，接收数据的设备在时钟上升沿采样数据。WS 信号和第一个数据位（MSB）同时产生。

这个标准里，数据发送和接收处理和 I<sup>2</sup>S 飞利浦标准一样。

图 19-17 MSB 对齐 16 位或 32 位全精度，CLKPOL=0

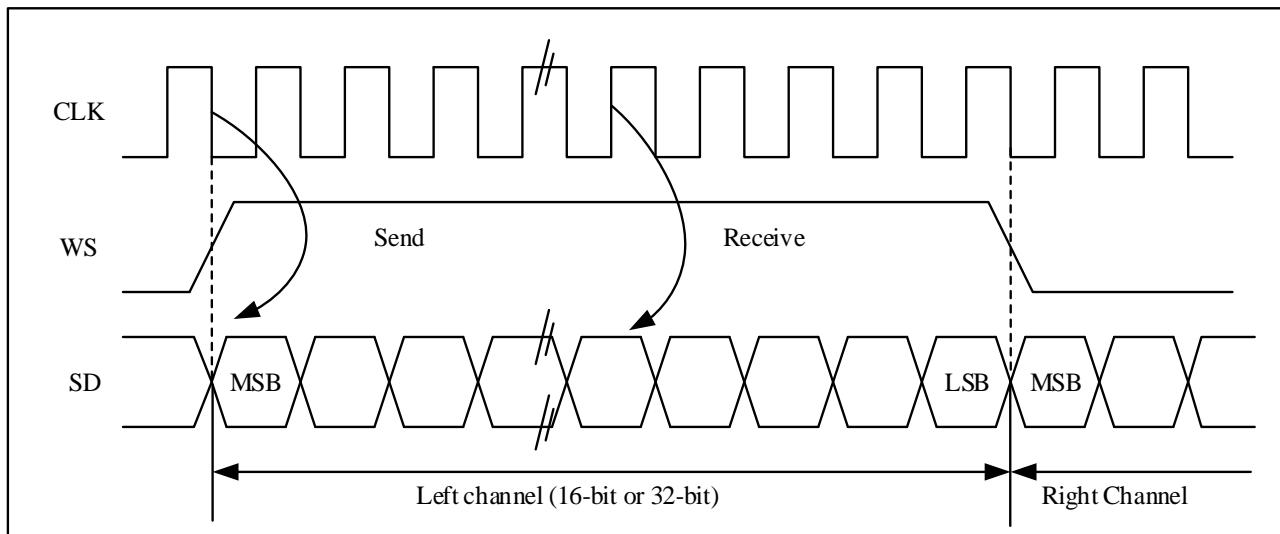


图 19-18 MSB 对齐 24 位数据, CLKPOL = 0

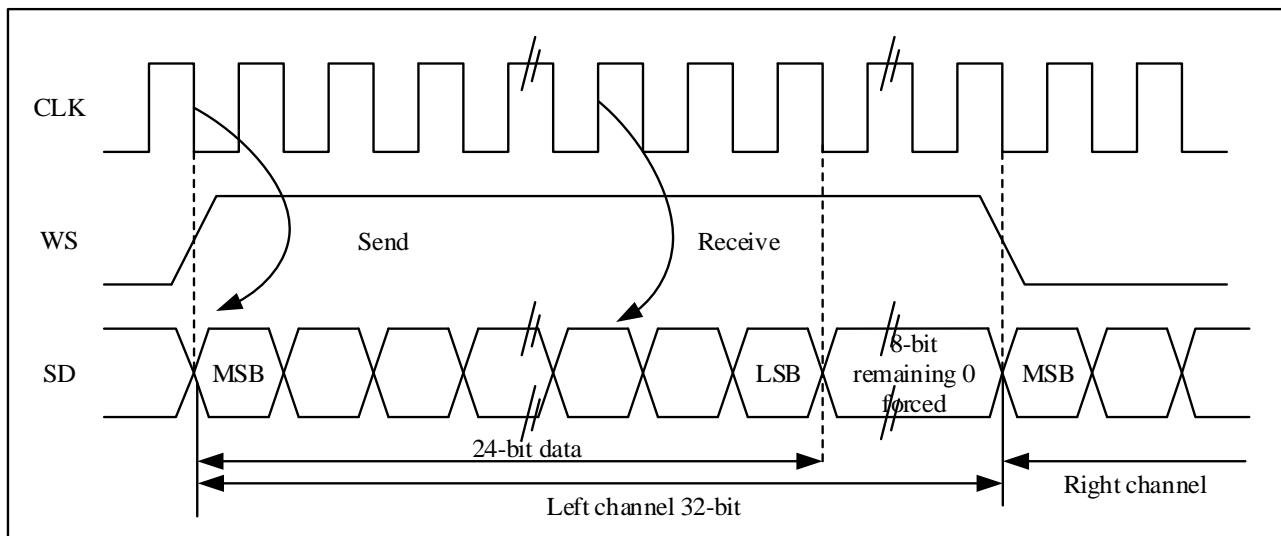
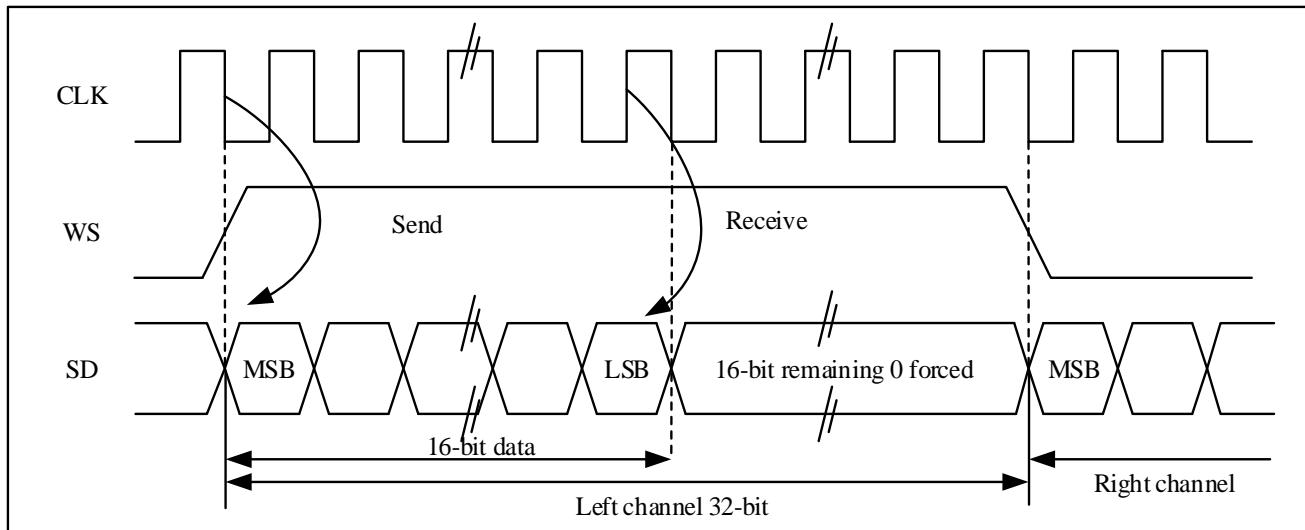


图 19-19 MSB 对齐 16 位数据扩展到 32 位包帧, CLKPOL = 0



### LSB 对齐标准

16 位或 32 位全精度帧格式, LSB 对齐标准和 MSB 对齐标准是一样的。

图 19-20 LSB 对齐 16 位或 32 位全精度, CLKPOL = 0

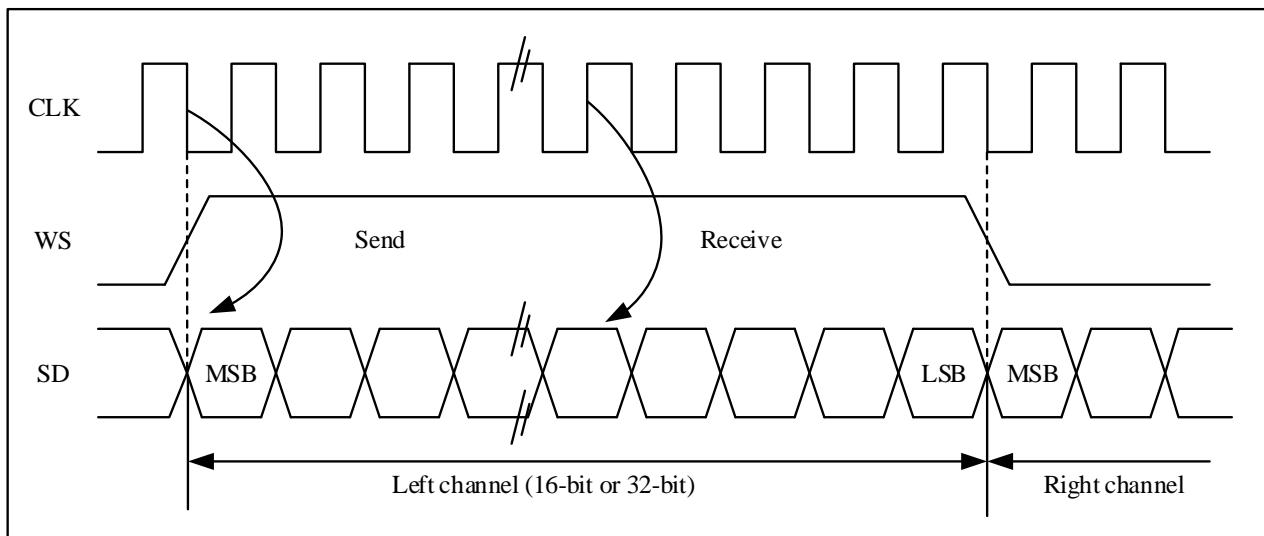
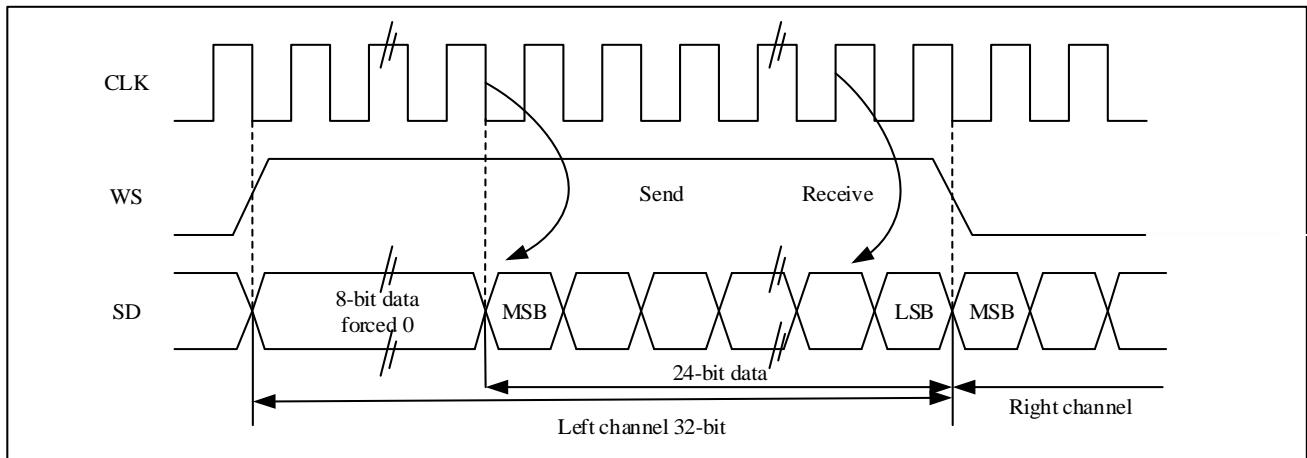
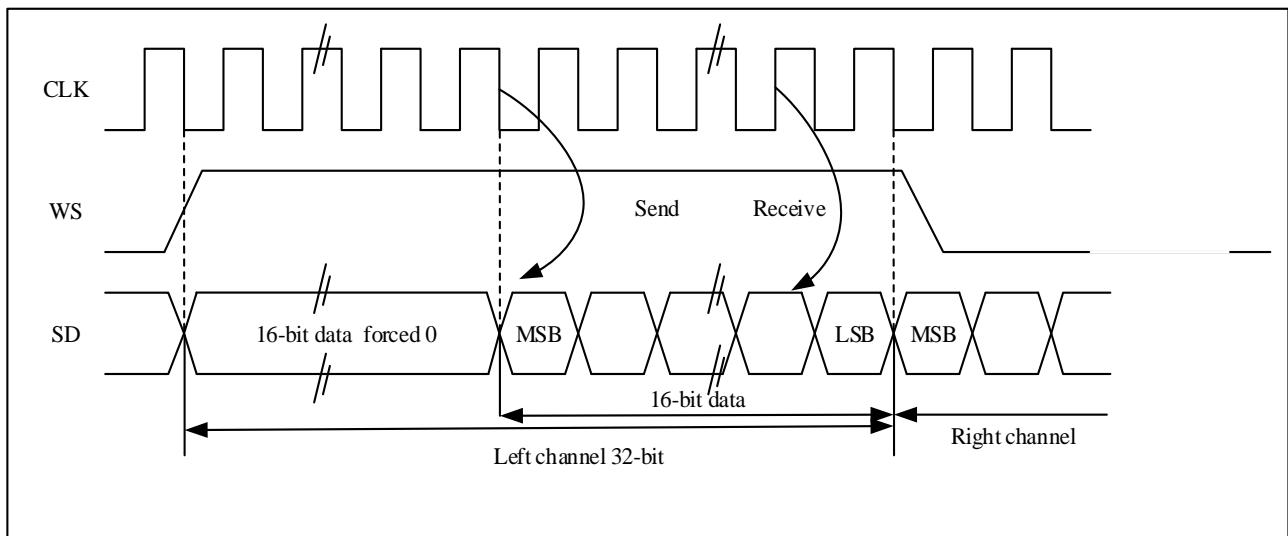


图 19-21 LSB 对齐 24 位数据, CLKPOL = 0



如果 24 位数据需要打包成 32 位数据帧格式, 每帧数据传输时, CPU 需要读或写 SPI\_DAT 寄存器 2 次。例如, 用户发送 24 位数据 0x95AA66, CPU 将先写 0xXX95 (仅低 8 位数据有效, 高 8 位数据没有意义, 可以是任何值) 进 SPI\_DAT 寄存器, 然后再写 0xAA66 进 SPI\_DAT 寄存器。如果用户接收 24 位数据 0x95AA66, CPU 将先读 SPI\_DAT 寄存器得到 0x0095 (仅低 8 位数据有效, 高 8 位总是为 0), 然后再读 SPI\_DAT 寄存器得到 0xAA66。

图 19-22 LSB 对齐 16 位数据扩展到 32 位包帧，CLKPOL = 0



如果 16 位数据需要打包进 32 位数据帧格式，每帧数据传输时，CPU 仅需要读或写 SPI\_DAT 寄存器一次。扩展到 32 位数据的高 16 位被硬件设置为 0x0000，如果用户发送或接收 16 位数据 0x89C1（扩展到 32 位数是 0x000089C1）。发送过程中，高 16 位半字（0x0000）需要先写到 SPI\_DAT 寄存器；一旦有效数据开始发送，下一个 TE 事件将产生。接收数据过程中，一旦设备接收到有效数据，RNE 事件将发生。这样，在 2 次读和写之间 CPU 将有更多时间，可以防止上溢或下溢的情况发生。

### PCM 标准

在 PCM 标准里，有短帧和长帧两种帧结构。用户可以设置 SPI\_I2SCFG.PCMFSYNC 位选择帧结构。WS 信号指示帧同步信息。

用于同步长帧的 WS 信号是 13 位有效的；用于同步短帧的 WS 信号长度是 1 位。

数据接收和发送的处理标准和 I<sup>2</sup>S 飞利浦标准是一样的。

图 19-23 PCM 标准波形 (16 位)

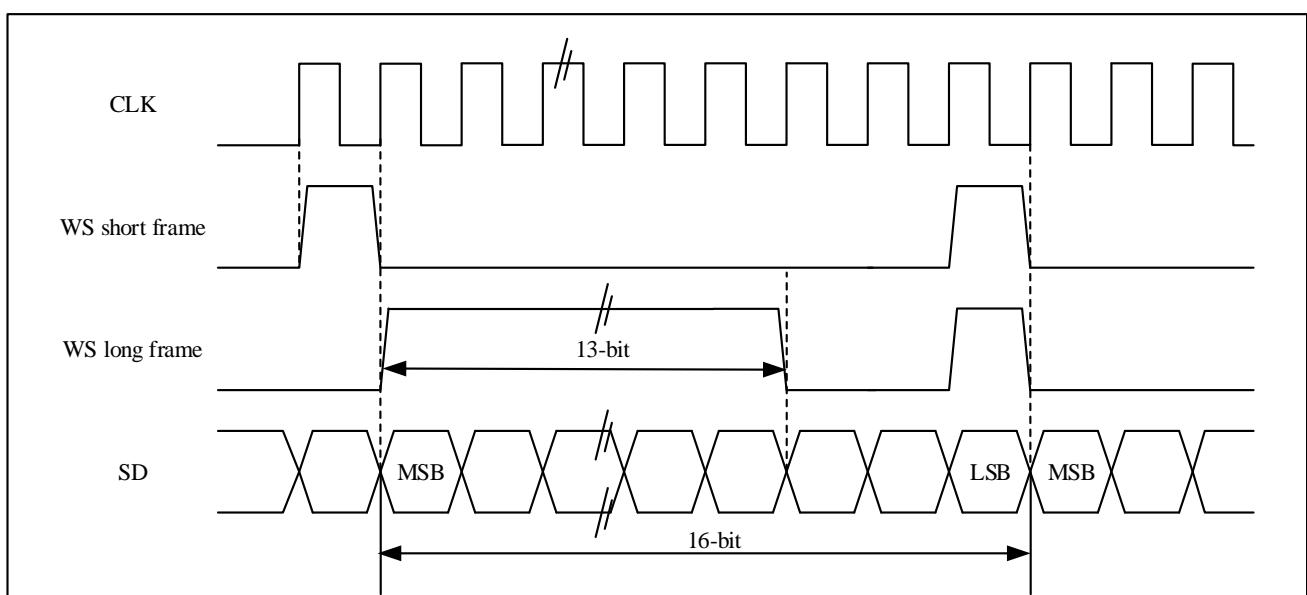
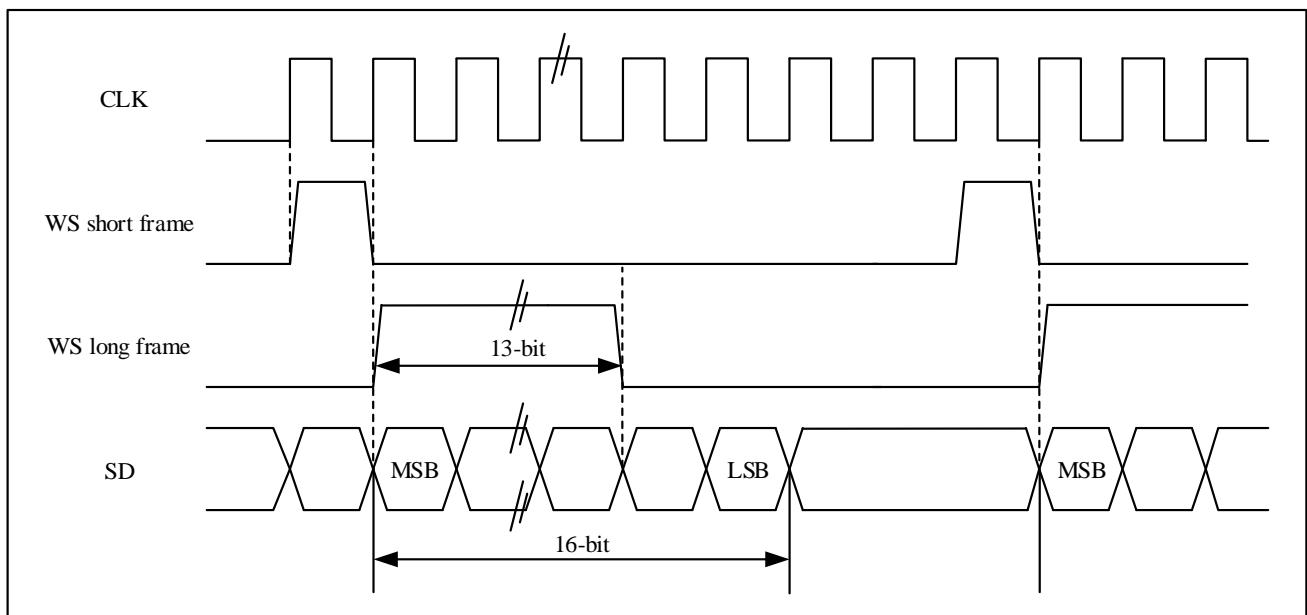


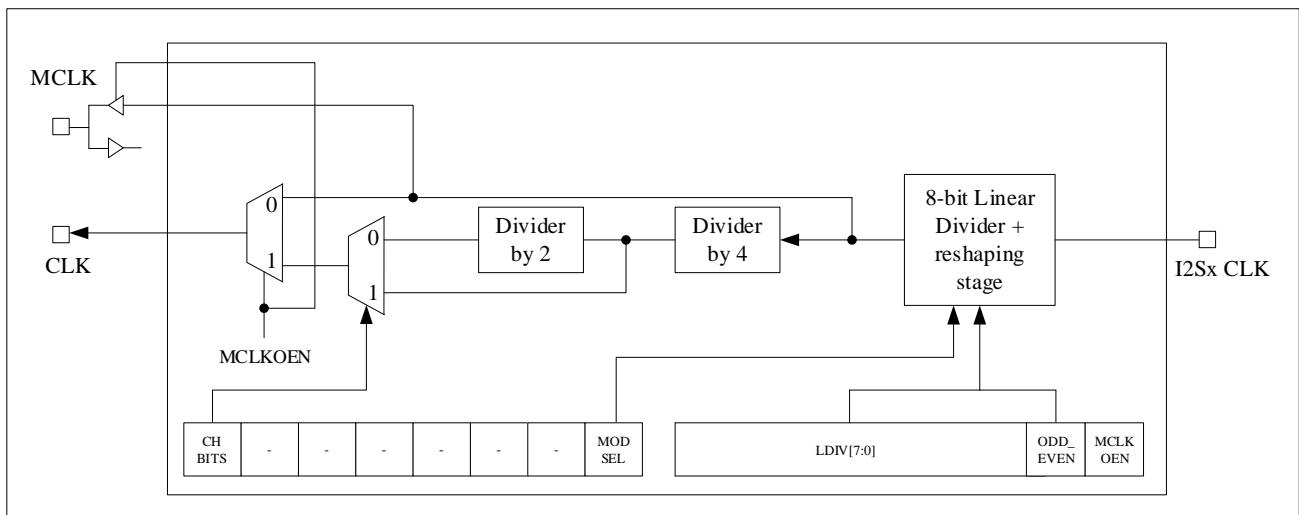
图 19-24 PCM 标准波形（16 位扩展到 32 位包帧）



## 19.4.2 时钟发生器

对于主设备，为了获得需要的音频频率，需要正确地对线性分频器进行设置

图 19-25 I<sup>2</sup>S 时钟发生器结构



注：I<sup>2</sup>SxCLK 的时钟源是驱动 AHB 时钟的 HSI、HSE 或 PLL 系统时钟。

I<sup>2</sup>S 的比特率决定了在 I<sup>2</sup>S 数据线上的数据流和 I<sup>2</sup>S 的时钟信号频率。

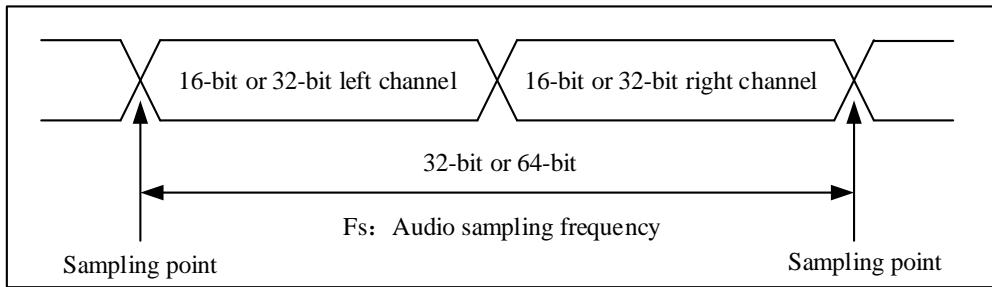
I<sup>2</sup>S 比特率 = 每个声道的比特数 × 声道数目 × 音频采样频率

对于一个具有左右声道和 16 位音频的信号，I<sup>2</sup>S 比特率计算如下：

$$I^2S \text{ 比特率} = 16 \times 2 \times F_s$$

如果包长为 32 位，则有：I<sup>2</sup>S 比特率 = 32 × 2 × F<sub>s</sub>

图 19-26 音频采样频率定义



通过设置 SPI\_I2SPREDIV.ODD\_EVEN 位和 SPI\_I2SPREDIV.LDIV 位，可以设置音频的采样信号频率。音频的采样频率可以是 96kHz、48kHz、44.1kHz、32kHz、22.05kHz、16kHz、11.025kHz 或者 8kHz（或任何此范围内的数值）。参照以下公式设置线性分频器：

$$\text{MCLKOEN}=1, \text{CHBITS}=0 \text{ 时}, F_s = I^2SxCLK / [(16 \times 2) \times ((2 \times LDIV) + ODD\_EVEN) \times 8]$$

$$\text{MCLKOEN}=1, \text{CHBITS}=1 \text{ 时}, F_s = I^2SxCLK / [(32 \times 2) \times ((2 \times LDIV) + ODD\_EVEN) \times 4]$$

$$\text{MCLKOEN}=0, \text{CHBITS}=0 \text{ 时}, F_s = I^2SxCLK / [(16 \times 2) \times ((2 \times LDIV) + ODD\_EVEN)]$$

$$\text{MCLKOEN}=0, \text{CHBITS}=1 \text{ 时}, F_s = I^2SxCLK / [(32 \times 2) \times ((2 \times LDIV) + ODD\_EVEN)]$$

可参照下表的时钟配置得到精确的音频频率。

表 19-2 使用标准的 8MHz HSE 时钟得到精确的音频频率

SYSCLK (MHz)	I <sup>2</sup> S_LDIV		I <sup>2</sup> S_ODD_EVEN		MCLK	期望值 Fs(Hz)	实际的 Fs(Hz)		误差	
	16 bits	32 bits	16 bits	32 bits			16 bits	32 bits	16 bits	32 bits
72	11	6	1	0	without	96000	97826.09	93750	1.90%	2.34%
72	23	11	1	1	without	48000	47872.34	48913.04	0.27%	1.90%
72	25	13	1	0	without	44100	44117.65	43269.23	0.04%	1.88%
72	35	17	0	1	without	32000	32142.86	32142.86	0.44%	0.44%
72	51	25	0	1	without	22050	22058.82	22058.82	0.04%	0.04%
72	70	35	1	0	without	16000	15675.75	16071.43	0.27%	0.45%
72	102	51	0	0	without	11025	11029.41	11029.41	0.04%	0.04%
72	140	70	1	1	without	8000	8007.11	7978.72	0.09%	0.27%
72	2	2	0	0	yes	96000	70312.15	70312.15	26.76%	26.76%
72	3	3	0	0	yes	48000	46875	46875	2.34%	2.34%
72	3	3	0	0	yes	44100	46875	46875	6.29%	6.29%
72	4	4	1	1	yes	32000	31250	31250	2.34%	2.34%
72	6	6	1	1	yes	22050	21634.61	21634.61	1.88%	1.88%
72	9	9	0	0	yes	16000	15625	15625	2.34%	2.34%
72	13	13	0	0	yes	11025	10817.3	10817.3	1.88%	1.88%
72	17	17	1	1	yes	8000	8035.71	8035.71	0.45%	0.45%

### 19.4.3 I<sup>2</sup>S 发送和接收流程

#### I<sup>2</sup>S 初始化流程

1. 用户可以设置 SPI\_I2SPREDIV.LDIV[7:0] 和 SPI\_I2SPREDIV.ODD\_EVEN 位配置相关的预分频和串行时钟波特率；
2. 如果用户需要主设备提供主时钟 MCLK 给外部的 DAC/ADC 音频设备，设置 SPI\_I2SPREDIV.MCLKOEN 位为 1。（根据不同的时钟输出，计算 LDIV 和 ODD\_EVEN，见 19.4.2 章节）
3. 用户可以设置 SPI\_I2SCFG 寄存器的 CLKPOL 位定义空闲时通讯时钟极性；用户可以设置 SPI\_I2SCFG.MODSEL 位为 1 配置设备为 I<sup>2</sup>S 模式，且设置 SPI\_I2SCFG.MODCFG[1:0] 选择 I<sup>2</sup>S 的主从模式和传输方向（发送或接收）；设置 SPI\_I2SCFG.STDSEL[1:0] 选择相应的 I<sup>2</sup>S 标准（PCM 标准下，设置 PCMF SYNC 位选择同步模式）；设置 SPI\_I2SCFG.TDATLEN[1:0] 选择数据位数，通过 SPI\_I2SCFG.CHBITS 位选择每个通道的数据位数。
4. 当用户需要使能中断或 DMA，配置操作和 SPI 一样。
5. 最后，设置 SPI\_I2SCFG.I2SEN 位为 1 开始 I<sup>2</sup>S 通讯。

## 发送流程

### 主模式

当 I<sup>2</sup>S 工作在主模式下，CLK 引脚输出串行时钟，WS 引脚产生声道选择信号，设置 SPI\_I2SPREDIV.MCLKOEN 位选择是否输出主时钟（MCLK）。

当数据写进发送缓存，发送过程开始。当前声道的数据并行从发送缓存传输到移位寄存器，SPI\_STS.TE 标志位置 1。此时，另外一个声道的数据应该被写进 SPI\_DAT 寄存器。通过 SPI\_STS.CHSIDE 标志位，可以检查当前数据相应的声道。SPI\_STS.CHSIDE 值当 SPI\_STS.TE 标志位置 1 时更新。完整的数据帧包括左声道和右声道，并且设备不可以仅传输部分数据帧。当 SPI\_STS.TE 标志位置 1，如果 SPI\_CTRL2.TEINTEN 位置 1，则产生中断。

写入数据的操作取决于选择的 I<sup>2</sup>S 标准。详见 19.4.1 章节

当用户要关闭 I<sup>2</sup>S 功能时，等待 SPI\_STS.TE 标志位置 1 且 SPI\_STS.BUSY 标志位为 0，然后清除 SPI\_I2SCFG.I2SEN 位为 0。

### 从模式

从模式的发送过程和主模式相似。不同处如下：

当 I<sup>2</sup>S 工作在从模式，不需要配置时钟，并且 CLK 引脚和 WS 引脚和主设备的相应引脚连接。当外部的主设备发送时钟信号，并且当 WS 信号要求数据传输时，发送过程开始。仅当从设备使能且数据已经写入 I<sup>2</sup>S 数据寄存器，外部的主设备才可以开始通讯。

当代表下个数据传输的第一个时钟沿到达前，新数据还没有写进 SPI\_DAT 寄存器，下溢产生，且 SPI\_STS.UNDER 标志位置 1。如果 SPI\_CTRL2.ERRINTEN 位置 1，中断产生，指示错误已经发生。

SPI\_STS.CHSIDE 标志指示当前被发送的数据对应哪个声道，和主模式发送过程相比，在从模式，SPI\_STS.CHSIDE 取决于外部主 I<sup>2</sup>S 设备的 WS 信号（WS 信号为 1 意味着左声道）

## 接收流程

### 主模式

音频数据总是以 16 位包格式接收，根据配置的数据和声道长度，接收到的音频数据需要一次或两次传输到接收缓存。

当数据从移位寄存器传输到接收缓存，SPI\_STS.RNE 标志位置 1，数据准备就绪可以从 SPI\_DAT 寄存器读取，如果 SPI\_CTRL2 寄存器的 RNEINTEN 位置 1，中断产生。读 SPI\_DAT 寄存器清除 RNE 标志位。如果之前的接收的数据没有读取，新的数据再次接收进来，上溢发生，OVER 标志位置 1，如果 SPI\_CTRL2 寄存器的 ERRINTEN 位置 1，中断产生，指示错误已经发生。

通过 SPI\_STS.CHSIDE 位，当前已经传输的数据对应的声道可以得到确认，当 SPI\_STS.RNE 标志位置 1，SPI\_STS.CHSIDE 值更新。

读数据的操作取决于选择的 I<sup>2</sup>S 标准，详见 19.4.1 章节。

当用户关闭 I<sup>2</sup>S 功能，不同的音频标准、数据长度、声道长度采用不同的步骤：

- 如果数据长度是 16 位，声道长度是 32 位 (SPI\_I2SCFG.TDATLEN = 00, SPI\_I2SCFG.CHBITS = 1)，LSB 对齐标准 (SPI\_I2SCFG.STDSEL=10)。
  1. 等待倒数第二个 SPI\_STS.RNE 标志位置 1；
  2. 软件延时，等待 17 个 I<sup>2</sup>S 时钟；
  3. 关闭 I<sup>2</sup>S (SPI\_I2SCFG.I2SEN=0)。
- 如果数据长度是 16 位，声道长度是 32 位 (SPI\_I2SCFG.TDATLEN = 00, SPI\_I2SCFG.CHBITS = 1)，MSB 对齐标准 (SPI\_I2SCFG.STDSEL=01)，I<sup>2</sup>S 飞利浦标准 (SPI\_I2SCFG.STDSEL=00) 或 PCM 标准 (SPI\_I2SCFG.STDSEL=11)
  1. 等待最后一个 SPI\_STS.RNE 标志位置 1；
  2. 软件延时，等待 1 个 I<sup>2</sup>S 时钟；
  3. 关闭 I<sup>2</sup>S (SPI\_I2SCFG.I2SEN=0)。
- 其他的 SPI\_I2SCFG.TDATLEN 和 SPI\_I2SCFG.CHBITS 组合和 SPI\_I2SCFG.STDSEL 选择的任意音频模式
  1. 等待倒数第二个 SPI\_STS.RNE 标志位置 1；
  2. 软件延时，等待 1 个 I<sup>2</sup>S 时钟；
  3. 关闭 I<sup>2</sup>S (SPI\_I2SCFG.I2SEN=0)。

#### 从模式

从模式的接收过程和主模式的相似，不同处如下：

SPI\_STS.CHSIDE 标志指示当前发送数据对应的哪个声道。和主模式接收流程相比，在从模式下，SPI\_STS.CHSIDE 取决于外部主设备的 WS 信号。关闭 I<sup>2</sup>S 功能时，当 SPI\_STS.RNE 标志位为 1 时清除 SPI\_I2SCFG.I2SEN 位为 0。

### 19.4.4 状态标识

在 SPI\_STS 寄存器有下面 4 个标志位，用于监视 I<sup>2</sup>S 总线状态

#### 发送缓存空标志位 (TE)

当发送缓存空，该标志位置 1，指示新数据可以写进 SPI\_DAT 寄存器。当发送缓存非空，该标志位清 0。

#### 接收缓存非空标志 (RNE)

当接收缓存非空，该标志位置 1，指示有效数据已经接收到接收缓存。当读取 SPI\_DAT 寄存器，该标志位清 0。

### 忙标志 (BUSY)

当传输开始，BUSY 标志位置 1，当传输结束后，BUSY 标志位硬件清 0（软件操作是无效的）。

主设备模式 (SPI\_I2SCFG.MODCFG=11) 下，在接收期间 BUSY 标志位被置 0。

当 I<sup>2</sup>S 模块关闭或传输完成，该标志位清 0。

在从机连续通讯模式，每个数据项传输之间，BUSY 标志位变低 1 个 I<sup>2</sup>S 时钟。因此，不要使用 BUSY 标志位处理每一个数据项的发送和接收。

### 声道标志 (CHSIDE)

CHSIDE 位用来指示当前收发的数据所在的通道，在 PCM 标准下，这个标志位没有意义。

在发送模式下，该标志位当 TE 标志位置 1 时更新；在接收模式下，该标志位当 RNE 标志位置 1 时更新。在收发过程中，如果发生上溢 (OVER) 或下溢 (UNDER) 错误，该标志位无意义，需要将 I<sup>2</sup>S 关闭再打开。

## 19.4.5 错误标志位

SPI\_STS 寄存器有 2 个错误标志位。

### 上溢标志位 (OVER)

当 RNE 标志位置 1，但是仍有数据发送到接收缓存，上溢错误将会发生，这时，OVER 标志置 1。如果用户使能相应的中断，中断将会产生。从这以后所有收到的数据将会丢失，SPI\_DAT 寄存器仅保留之前未读的数据。

依次读 SPI\_DAT 寄存器和 SPI\_STS 寄存器，可以清除 OVER 标志位。

### 下溢标志位 (UNDER)

在从发送模式下，当发送数据的第一个时钟沿到达，如果发送缓存仍然是空的，UNDER 标志位置 1。如果用户使能相应的中断，中断将会产生。

读 SPI\_STS 寄存器清除 UNDER 位。

## 19.4.6 I<sup>2</sup>S 中断

所有 I<sup>2</sup>S 中断如下表所列。

表 19-3 I<sup>2</sup>S 中断请求

中断事件	事件标志位	使能控制位
发送缓存空标志	TE	TEINTEN
接收缓存非空标志	RNE	RNEINTEN
下溢标志	UNDER	ERRINTEN
上溢标志	OVER	

## 19.4.7 DMA 功能

工作在 I<sup>2</sup>S 模式，不需要数据传输保护功能，因此不需要支持 CRC，其他的 DMA 的功能与 SPI 模式是一样的。

## 19.5 SPI 和 I<sup>2</sup>S 寄存器描述

### 19.5.1 SPI 寄存器总览

表 19-4 SPI 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	BIDIRMODE	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
000h	SPI_CTRL1	Reserved															BIDIRMODE	15	BIDIROEN	14	CRCEN	13	CRCNEXT	12	DATFF	11	RONLY	10	SSMEN	9	SSEL	8	LSBFF	7	SPLEN	6	BR[2:0]		
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
004h	SPI_CTRL2	Reserved															TEINTEN	0	0	RNEINTEN	0	ERRINTEN	0	0	0	0	0	0	0	0	0	0	0						
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
008h	SPI_STS	Reserved															BUSY	0	OVER	0	MODERR	0	CRCCR	0	ERRINTEN	0	0	0	0	0	0	0	0	0					
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
00Ch	SPI_DAT	Reserved															DAT[15:0]													CRCPOLY[15:0]									
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
010h	SPI_CRCPOLY	Reserved															CRCRDAT[15:0]													CRCTDAT[15:0]									
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
014h	SPI_CRCRDAT	Reserved															CRCTDAT[15:0]													CRCRDAT[15:0]									
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
018h	SPI_CRCTDAT	Reserved															CRCPOLY[15:0]													DAT[15:0]									
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
01Ch	SPI_I2SCFG	Reserved															MODSEL	1	I2SEN	0	PCMFSYNC	1	Reserved	0	STDSEL	1	CLKPOL	0	TDATLEN	1	CHBITS	0	0						
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						
020h	SPI_I2SPREDIV	Reserved															MCLKOEN	1	ODD_EVEN	0	LDIV[7:0]	LDIV[7:0]																	
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						

### 19.5.2 SPI 控制寄存器 1 (SPI\_CTRL1) (不能用于 I<sup>2</sup>S 模式)

地址偏移: 0x00

复位值: 0x0000

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

位域	名称	描述
15	BIDIRMODE	<p>双向数据模式使能 0: 选择“双线单向”模式。 1: 选择“单线双向”模式。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
14	BIDIROEN	<p>双向模式下输出使能 0: 输出禁止 (仅接收模式)。 1: 输出使能 (仅发送模式)。 在主机模式下, “单线”数据线是 MOSI 引脚, 在从机模式下, “单线”数据线是 MISO 引脚。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
13	CRCEN	<p>硬件 CRC 计算使能 0: CRC 计算禁能。 1: CRC 计算使能。 该位只能用于全双工模式。 <b>注意:</b> 该位仅在 SPI 禁能时可写 (SPI_CTRL1.SPIEN=0), 否则会产生错误。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
12	CRCNEXT	<p>下个发送 CRC 0: 下个发送的数据来自发送缓存。 1: 下个发送的数据来自 CRC 寄存器。 <b>注意:</b> 最后数据写进 SPI_DAT 寄存器后, 该位应该立即写入。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
11	DATFF	<p>数据帧格式 0: 8 位数据帧格式用于发送/接收。 1: 16 位数据帧格式用于发送/接收 <b>注意:</b> 该位仅 SPI 禁能时可以写入 (SPI_CTRL1.SPIEN=0), 否则会产生错误。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
10	RONLY	<p>仅接收模式 该位和 BIDIRMODE 位一起决定双线单向模式的传输方向。在多个从设备的应用场景, 该位仅被访问的从设备置 1, 仅被访问的从设备可以输出, 从而避免数据线冲突。 0: 全双工 (发送和接收模式)。 1: 输出禁能 (仅接收模式)。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
9	SSMEN	<p>软件从设备管理 当 SSMEN 位置 1, NSS 引脚电平由 SSEL 位的值决定。 0: 软件从设备管理被禁止。 1: 使能软件从设备管理。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
8	SSEL	<p>内部从设备选择 该位仅在 SSMEN 位置 1 时有意义。它决定了 NSS 电平, 且 NSS 引脚的 I/O 操作无效。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>

位域	名称	描述
7	LSBFF	<p>帧格式 0: 先发送 MSB。 1: 先发送 LSB。</p> <p><b>注意:</b> 通讯过程中该位不能被改变。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
6	SPIEN	<p>SPI 使能 0: 禁能 SPI 模块。 1: 使能 SPI 模块。</p> <p><b>注意:</b> 不能用在 I<sup>2</sup>S 模式。 <b>注意:</b> 当关闭 SPI 模块, 请遵循 19.3.4 的流程操作。</p>
5:3	BR[2:0]	<p>波特率控制 000: fPCLK/2 001: fPCLK/4 010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256</p> <p><b>注意:</b> 当通讯过程中, 这些位不能被修改。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
2	MSEL	<p>主设备选择 0: 配置为从设备。 1: 配置为主设备。</p> <p><b>注意:</b> 通讯过程中该位不能被修改。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
1	CLKPOL	<p>时钟极性 0: 空闲状态下, SCLK 保持为低电平。 1: 空闲状态下, SCLK 保持为高电平。</p> <p><b>注意:</b> 通讯过程中该位不能被修改。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>
0	CLKPHA	<p>时钟相位 0: 第一个时钟沿采样数据 1: 第二个时钟沿采样数据。</p> <p><b>注意:</b> 通讯过程中该位不能被修改。 <b>注意:</b> 不能用在 I<sup>2</sup>S 模式。</p>

### 19.5.3 SPI 控制寄存器 2 (SPI\_CTRL2)

地址偏移: 0x04

复位值: 0x0000

15	Reserved	8	7	6	5	4	3	2	1	0
			TE INTEN	RNE INTEN	ERR INTEN		Reserved	SSOEN	TDMAEN	RDMAEN

rw      rw      rw      rw      rw      rw

位域	名称	描述
15:8	Reserved	保留, 必须保持复位值
7	TEINTEN	发送缓存空中断使能 0: 禁能 TE 中断 1: 允许 TE 中断, 当 SPI_STS.TE 标志位置 1 时产生中断请求。
6	RNEINTEN	接收缓存非空中断使能 0: 禁能 RNE 中断; 1: 允许 RNE 中断, 当 SPI_STS.RNE 标志位置 1 时产生中断请求。
5	ERRINTEN	错误中断使能 当错误 (CRCERR, OVER, MODERR) 发生时, 该位控制是否产生中断 0: 禁能错误中断; 1: 允许错误中断。
4:3	Reserved	保留, 必须保持复位值。
2	SSOEN	NSS 输出使能 0: 在主模式下禁止输出 NSS, 该设备可以工作在多主设备模式; 1: 设备开启时, 打开主模式下 NSS 输出, 该设备不能工作在多主设备模式。 注意: 不能用于 PS 模式。
1	TDMAEN	发送缓冲区 DMA 使能 当该位被设置时, TE 标志位一旦被置位就发出 DMA 请求 0: 禁止发送缓冲区 DMA; 1: 使能发送缓冲区 DMA。
0	RDMAEN	接收缓冲区 DMA 使能 当该位被设置时, RNE 标志位一旦被置位就发出 DMA 请求 0: 禁止接收缓冲区 DMA; 1: 使能接收缓冲区 DMA。

#### 19.5.4 SPI 状态寄存器(SPI\_STS)

地址偏移: 0x08

复位值: 0x0002

15	Reserved	8	7	6	5	4	3	2	1	0
			BUSY	OVER	MODERR	CRCERR	UNDER	CHSIDE	TE	RNE

r      r      r      rc\_w0      r      r      r      r

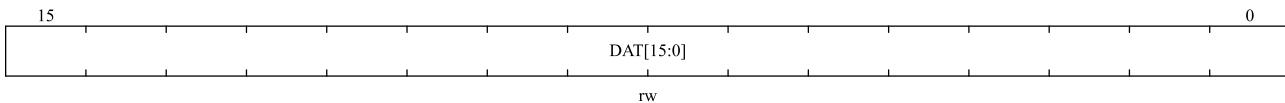
位域	名称	描述
15:8	Reserved	保留, 必须保持复位值。

位域	名称	描述
7	BUSY	忙标志 0: SPI 不忙; 1: SPI 忙于通讯或发送缓存非空。 该位硬件清 0 或置 1 <b>注意:</b> 使用该标志需要特别注意, 详见 19.3.3 章节和 19.3.4 章节。
6	OVER	上溢标志 0: 没有上溢错误; 1: 发生上溢错误。 该位硬件置位, 根据软件操作序列清除, 更详细的信息详见 19.3.7 章节。
5	MODERR	模式错误 0: 没有模式错误; 1: 模式错误发生。 该位硬件置位, 根据软件操作序列清除, 更详细的信息详见 19.3.7 章节。 <b>注意:</b> 不能用于 I <sup>2</sup> S 模式。
4	CRCERR	CRC 错误标志 0: 接收到的 CRC 值和 SPI_CRCRDAT 寄存器值匹配; 1: 接收到的 CRC 值和 SPI_CRCRDAT 寄存器值不匹配。 该位硬件置位, 软件写 0 清除。 <b>注意:</b> 不能用于 I <sup>2</sup> S 模式。
3	UNDER	下溢标志 0: 没有下溢产生; 1: 有下溢产生。 该位硬件置 1, 通过软序列清 0, 详见 19.4.5 章节。 <b>注意:</b> 不能用于 SPI 模式。
2	CHSIDE	声道 0: 需要发送或接收左声道; 1: 需要发送或接收右声道。 <b>注意:</b> 不能用于 SPI 模式。PCM 模式是没有意义的。
1	TE	发送缓存空 0: 发送缓存非空; 1: 发送缓存空。
0	RNE	接收缓存非空 0: 接收缓存空; 1: 接收缓存非空。

## 19.5.5 SPI 数据寄存器(SPI\_DAT)

地址偏移: 0x0C

复位值: 0x0000

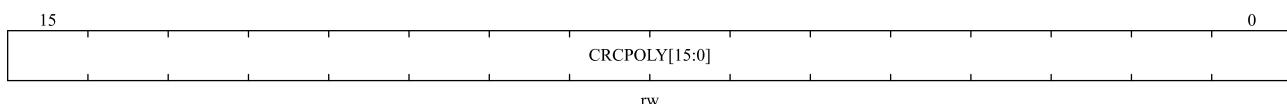


位域	名称	描述
15:0	DAT[15:0]	<p>数据寄存器 待发送或者已经收到的数据 数据寄存器对应两个缓冲区：一个用于写（发送缓冲）；另外一个用于读（接收缓冲）。写操作将数据写到发送缓冲区；读操作将返回接收缓冲区里的数据。</p> <p>对 SPI 模式的注释：根据 SPI_CTRL1 寄存器的 DATFF 位对数据帧格式的选择，数据的发送和接收可以是 8 位或者 16 位的。为保证正确的操作，需要在启用 SPI 之前就确定好数据帧格式。</p> <p>对于 8 位的数据，缓冲器是 8 位的，发送和接收时只会用到 SPI_DAT[7:0]。在接收时，SPI_DAT[15:8]被强制为 0。</p> <p>对于 16 位的数据，缓冲器是 16 位的，发送和接收时会用到整个数据寄存器，即 SPI_DAT[15:0]。</p>

### 19.5.6 SPI CRC 多项式寄存器 (SPI\_CRCPOLY) (不能用于 I<sup>2</sup>S 模式)

地址偏移：0x10

复位值：0x0007

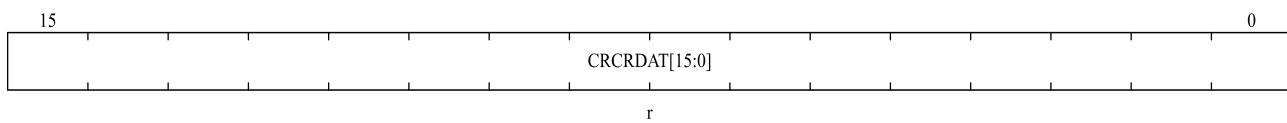


位域	名称	描述
15:0	CRCPOLY [15:0]	<p>CRC 多项式寄存器 这个寄存器包含了用于 CRC 计算的多项式 复位值是 0x0007，根据应用可以设置其他值 注意：不能用于 I<sup>2</sup>S 模式。</p>

### 19.5.7 SPI 接收 CRC 寄存器 (SPI\_CRCRDAT) (不能用于 I<sup>2</sup>S 模式)

地址偏移：0x14

复位值：0x0000



位域	名称	描述
15:0	CRCRDAT	<p>接收 CRC 寄存器 当 CRC 计算使能时，CRCRDAT[15:0]中包含了后续收到的字节计算的 CRC 数值。当向 SPI_CTRL1 寄存器的 CRCEN 位写入‘1’时，该寄存器被复位。CRC 计算使用 SPI_CRCPOLY 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</p>

注意：当BUSY标志为'1'时读该寄存器，将可能读到不正确的数值。

注意：不能用于在  $I^2S$  模式。

#### 19.5.8 SPI 发送 CRC 寄存器 (SPI\_CRCTDAT)

地址偏移: 0x18

复位值: 0x0000

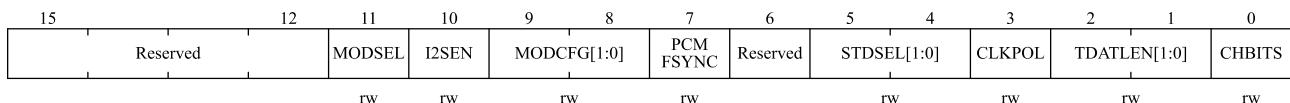


位域	名称	描述
15:0	CRCTDAT	<p>发送 CRC 寄存器</p> <p>当 CRC 计算使能时，CRCTDAT[15:0]中包含了后续发送的字节计算的 CRC 数值。当向 SPI_CTRL1 寄存器中的 CRCEN 位写入'1'时，该寄存器被复位。CRC 计算使用 SPI_CRCPOLY 中的多项式。</p> <p>当数据帧格式被设置为 8 位时，仅低 8 位参与计算，并且按照 CRC8 的方法进行；当数据帧格式为 16 位时，寄存器中的所有 16 个位都参与计算，并且按照 CRC16 的标准。</p> <p>注意：当 BUSY 标志为'1'时读该寄存器，将可能读到不正确的数值。</p> <p>注意：不能用于在 PS 模式。</p>

### 19.5.9 SPI\_I<sup>2</sup>S 配置寄存器 (SPI\_I2SCFG)

地址偏移: 0x1c

复位值: 0x0000



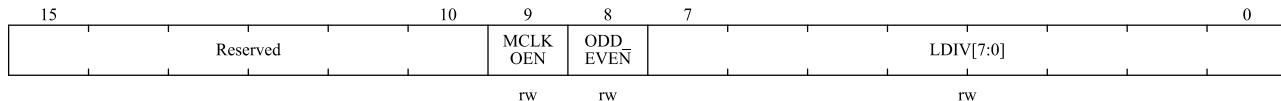
位域	名称	描述
15:12	Reserved	保留，必须保持复位值。
11	MODSEL	I <sup>2</sup> S 模式选择 0: 选择 SPI 模式 1: 选择 I <sup>2</sup> S 模式  注意: 该位仅当 SPI 或 I <sup>2</sup> S 关闭时可设置。
10	I <sup>2</sup> SEN	I <sup>2</sup> S 使能 0: 关闭 I <sup>2</sup> S; 1: 使能 I <sup>2</sup> S。  注意: 不能用于 SPI 模式。
9:8	MODCFG	I <sup>2</sup> S 模式设置

位域	名称	描述
		00: 从设备发送; 01: 从设备接收; 10: 主设备发送; 11: 主设备接收。 <b>注意:</b> 该位仅当 I <sup>2</sup> S 关闭时可设置。 <b>注意:</b> 不能用于 SPI 模式。
7	PCMFSYNC	PCM 帧同步 0: 短帧同步; 1: 长帧同步。 <b>注意:</b> 该位仅当 SPI_I2SCFG.STDSEL = 11 (PCM 标准使用) 有意义。 <b>注意:</b> 不能用于 SPI 模式。
6	Reserved	保留, 必须保持复位值。
5:4	STDSEL	I <sup>2</sup> S 标准选择 00: I <sup>2</sup> S 飞利浦标准; 01: 高字节对齐标准 (左对齐); 10: 低字节对齐标准 (右对齐); 11: PCM 标准。 <b>详见 19.4.1 章节</b> <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置 <b>注意:</b> 不能用于 SPI 模式。
3	CLKPOL	静止时钟极性 0: I <sup>2</sup> S 时钟静止态为低电平; 1: I <sup>2</sup> S 时钟静止态为高电平。 <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置 <b>注意:</b> 不能用于 SPI 模式。
2:1	TDATLEN	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。 <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置 <b>注意:</b> 不能用于 SPI 模式。
0	CHBITS	声道长度 (每个音频声道的数据位数) 0: 16 位宽; 1: 32 位宽。 仅当 TDATLEN = 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。 <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置 <b>注意:</b> 不能用于 SPI 模式。

## 19.5.10 SPI\_I<sup>2</sup>S 预分频寄存器 (SPI\_I2SPREDIV)

地址偏移: 0x20

复位值: 0x0002



位域	名称	描述
15:10	Reserved	保留, 必须保持复位值。
9	MCLKOEN	主时钟输出使能 0: 关闭主设备主时钟输出; 1: 主时钟输出使能。 <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置, 该位仅用于 I <sup>2</sup> S 主模式, 不能用于 SPI 模式。
8	ODD_EVEN	奇系数预分频 0: 实际频率分频系数 = LDIV × 2; 1: 实际频率分频系数 = (LDIV × 2)+1。 <b>详见 19.4.2 章节</b> <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置, 该位仅用于 I <sup>2</sup> S 主模式, 不能用于 SPI 模式。
7:0	LDIV	I <sup>2</sup> S 线性预分频 禁止设置 LDIV [7:0] = 0 或 LDIV [7:0] = 1 <b>详见 19.4.2 章节</b> <b>注意:</b> 为正确操作, 该位仅在 I <sup>2</sup> S 关闭时设置, 该位仅用于 I <sup>2</sup> S 主模式, 不能用于 SPI 模式。

## 20 实时时钟(RTC)

### 20.1 简介

- 实时时钟（RTC）是一个独立的 BCD 定时器/计数器
- 软件支持夏时令补偿
- 可编程周期性自动唤醒定时器
- 两个 32 位寄存器包含时、分、秒、年、月、日（几号）、星期（星期几）
- 独立的 32 位寄存器包含亚秒
- 两个编程闹钟
- 两个 32 位寄存器包含编程闹钟时、分、秒、年、月、日（几号）、星期（星期几）
- 两个独立的 32 位寄存器包含编程闹钟亚秒
- 数字精密校准功能
- 参考时钟检测：一个更加精确的外部时钟源（50 或 60Hz）能够用于改进日历精度
- 三个可配置滤波和内部上拉的入侵检测事件
- 时间戳功能
- 20 个备份寄存器，可在低功耗模式下保持数据
- 多个中断/事件唤醒源，包括闹钟 A、闹钟 B、唤醒定时器、时间戳、入侵
- RCC 寄存器使能 RTC 模块且电压保持在工作范围内，RTC 在任何模式下都不会停止（包括 RUN 模式、SLEEP 模式、STOP0 模式、STOP2 模式和 STANDBY 模式）
- RTC 提供多种唤醒源可以使 MCU 从所有的低功耗模式下唤醒（SLEEP 模式，STOP0 模式，STOP2 模式和 STANDBY 模式）

## 20.1.1 主要特性

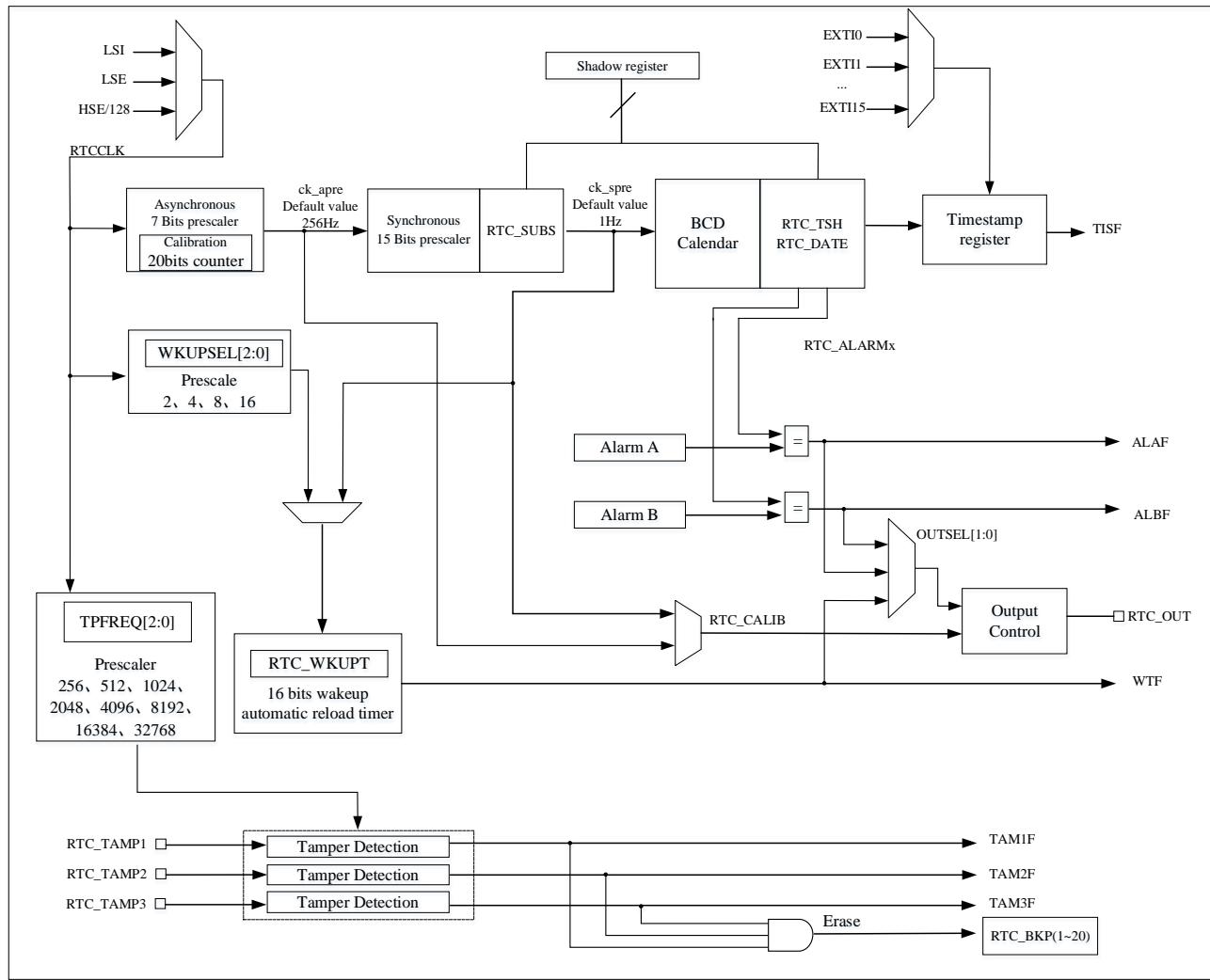
表 20-1 RTC 功能支持

主要功能	描述
时钟	RTC 时钟源可以选择 LSI、LSE 或 HSE/128
Reset	<p>APB 接口被系统复位， RTC 模块通过 APB 同步的一些寄存器会被复位 下面寄存器当系统复位时会被清除</p> <ul style="list-style-type: none"> <li>● RTC_SUBS</li> <li>● RTC_TSH</li> <li>● RTC_DATA</li> <li>● RTC_INITSTS(一些 bits)</li> </ul> <p>RTC 内核可以通过备份域复位而复位 复位 RTC，以及在低功耗模式下保留一些寄存器的内容，包括：</p> <ul style="list-style-type: none"> <li>● RTC_CTRL</li> <li>● RTC_PRE</li> <li>● RTC_CALIB</li> <li>● RTC_SCTRL</li> <li>● RTC_TSSS, RTC_TST 和 RTC_TSD</li> <li>● RTC_TMPCFG</li> <li>● RTC_WKUPT</li> <li>● RTC_ALRMASS/RTC_ALRMA</li> <li>● RTC_ALRMBSS/RTC_ALRMB</li> <li>● RTC_OPT</li> <li>● RTC_BKP(1~20)</li> </ul>
Calendar	日历包含亚秒、秒、分、时（12 小时或 24 小时制）、星期、日、月、年，这些数据都存在 APB 模块的影子寄存器中。
Wakeup Timer	输出寄存器 RTC_OUT 可以配置为发送唤醒事件到 GPIO，同时我们可以选择中断/事件来唤醒 CPU 的 SLEEP、STOP0、STOP2 和 STANDBY 模式。
Alarm	RTC_OUT 配置输出到 GPIO，也可以唤醒 CPU 或触发 PWR 在匹配发生时从 SLEEP、STOP0、STOP2 和 STANDBY 模式中唤醒
Tamper	3 个入侵检测逻辑是系统唤醒的一个来源，如果入侵事件发生在其中一个输入线。当启用时，入侵事件也会导致备份寄存器的删除。它也是对 LP 定时器进行硬件触发的一个来源
Timestamp	GPIO 事件可触发保存时间戳功能。它是从低功耗模式唤醒的一个来源。 另外，入侵事件可以是时间戳事件的来源。
Interrupts/events	闹钟 A/闹钟 B 中断/事件 唤醒中断/事件 时间戳中断/事件 入侵中断/事件
Backup registers	20 个备份寄存器

## 20.2 RTC 功能描述

### 20.2.1 RTC 框图

图 20-1 RTC 功能框图



RTC 包括以下功能模块：

- Alarm A 和 Alarm B 事件/中断
- 时间戳事件/中断
- 入侵事件/中断
- 20 个 32 位备份寄存器
- RTC 输出功能：
  - ◆ 256 Hz 或者 1Hz 时钟输出(当 LSE 频率是 32.768 kHz)
  - ◆ 闹钟输出 (极性可配置), 闹钟 A 和闹钟 B 可选

- ◆ 自动唤醒输出（极性可配置）
- RTC 输入功能：
  - ◆ 时间戳事件检测
  - ◆ 50 或者 60Hz 参考时钟输入
  - ◆ 入侵事件检测
- 通过配置输出寄存器控制 PC13：
  - ◆ 设置 RTC\_OPT.TYPE 位配置 PC13 开漏/推挽输出

## 20.2.2 RTC 控制的 GPIO

时间戳输入来自 IOM（映射到 PC13）或者 EXTI 模块，如果是 EXTI 模块，具体请参考时间戳触发源选择 (EXTI\_TS\_SEL)。

RTC\_OUT (闹钟、唤醒事件或者校准输出 (256Hz 或者 1Hz)) 映射到 PC13，不管 PC13 GPIO 是什么配置，PC13 的引脚配置由 RTC 控制为输出。

PC13 引脚被用作 TAMPER1 入侵检测引脚，PA0 引脚被用作 TAMPER2 入侵检测引脚，

PA8 引脚被用作 TAMPER3 入侵检测引脚。

PA10 或者 PB15 能够被用作 RTC\_REFCLKIN 参考时钟输入引脚。

## 20.2.3 RTC 寄存器写保护

PWR\_CTRL.DBKP 位(见电源控制寄存器 (PWR\_CTRL)) 默认被清除，所以 PWR\_CTRL.DBKP 必须置 1 去使能 RTC 寄存器写功能。一旦备份域复位，所有的 RTC 写保护寄存器都会写保护，所有的 RTC 写保护寄存器需要按如下步骤去解锁写保护：

- 将 0xCA 写入 RTC\_WRP 寄存器
- 将 0x53 写入 RTC\_WRP 寄存器

在解锁这些寄存器后，不能重新使能写保护除非 RTC 被软件复位或者重新上电。解锁机制只检查 RTC\_WRP 寄存器的写操作。在解锁过程中、解锁前、解锁后，对其他寄存器的写操作不会影响解锁结果。

## 20.2.4 RTC 时钟和预分频

RTC 时钟源：

- LSE 时钟
- LSI 时钟
- HSE/128 时钟

为了降低功耗，将预分频器分为异步预分频器和同步预分频器。如果同时使用两个预分频器，建议异步预分频器的值尽可能大。

- 7 位异步预分频器由 RTC\_PRE.DIVA[6:0] 位控制

- 15 位同步预分频器由 RTC\_PRE.DIVS[14:0] 位控制

$f_{ck\_apre}$  和  $f_{ck\_spre}$  公式如下：

$$f_{ck\_apre} = \frac{f_{RTCCLK}}{RTC\_PRE.DIVS[14:0]+1}$$

$$f_{ck\_spre} = \frac{f_{RTCCLK}}{(RTC\_PRE.DIVS[14:0]+1)*(RTC\_PRE.DIVA[6:0]+1)}$$

$ck\_apre$  时钟用于对 RTC\_SUBS 亚秒递减计数器提供时钟。当到达 0 时，用 RTC\_PRE.DIVS[14:0]的值重新加载 RTC\_SUBS。

## 20.2.5 RTC 日历

这里有三个影子寄存器，分别是 RTC\_DATE, RTC\_TSH 和 RTC\_SUBS。RTC 时间和日期寄存器可以通过影子寄存器访问。也可以直接访问，以避免等待同步时间。这三个影子寄存器如下：

- RTC\_DATE：设置和读取日期
- RTC\_TSH：设置和读取时间
- RTC\_SUBS：读取亚秒

每隔两个 RTCCLK 周期之后，将当前的日历值复制到影子寄存器中，并将 RTC\_INITSTS.RSYF 位置为 1。此过程在低功耗(停止和待机)模式下不执行。当退出这些模式时，影子寄存器在 2 个 RTCCLK 周期后更新值。

默认情况下，当用户尝试访问日历寄存器时，它将访问影子寄存器的内容。用户可以通过设置 RTC\_CTRL.BYPS 位直接访问日历寄存器。

当 RTC\_CTRL.BYPS=0，日历从影子寄存器获取值，当读 RTC\_SUBS、RTC\_TSH 或 RTC\_DATE 寄存器时，有必要确保 APB1 时钟的频率( $f_{APB1}$ )至少 7 倍于 RTC 时钟频率( $f_{RTCCLK}$ )，而且不允许出现 APB1 时钟频率低于 RTC 时钟频率的情况。系统复位将复位影子寄存器。

## 20.2.6 日历初始化和配置

预分频值和日历值可通过以下步骤进行初始化：

- 通过设置 RTC\_INITSTS.INITM 位为 1 进入初始模式，然后等待 RTC\_INITSTS.INITF 位被置 1
- 设置 RTC\_PRE.DIVS[14:0] 和 RTC\_PRE.DIVA[6:0] 位
- 写入初始日历值，包括时间和日期到影子寄存器 (RTC\_TSH 和 RTC\_DATE)，通过 RTC\_CTRL.HFMT 位配置时间格式 (12 小时或 24 小时制)
- 通过清除 RTC\_INITSTS.INITM 位退出初始化模式

日历计数器的值将在 4 个 RTCCLK 时钟周期后自动从影子寄存器加载，然后重新启动日历计数器。

注意：RTC 进入初始化模式前，需保证 RTC\_SUBS.SS[15:0]的值不小于 2，需要读一下 RTC\_DATE 寄存器。

## 20.2.7 日历读取

### 1. 当 RTC\_CTRL.BYPS=0 时读取日历

如果 RTC\_CTRL.BYPS=0，则从影子寄存器读取日历值。为了正确读取 RTC 日历寄存器(RTC\_SUBS, RTC\_TSH 和 RTC\_DATE)，APB1 时钟频率必须设置为大于 RTC 时钟频率的 7 倍。在任何情况下，APB1 时钟频率都不能小于 RTC 时钟频率。

如果 APB1 时钟频率不大于或不等于 RTC 时钟频率的 7 倍，请参考下面的步骤读取日历值：

- 读取 RTC\_SUBS、RTC\_TSH 和 RTC\_DATE 值两次
- 比较两次读到的数据，如果相等，则认为读到的数据是正确的，如果不相等，需要读第三次数据
- 第三次读到的数据可以认为是正确的

影子寄存器(RTC\_SUBS, RTC\_TSH 和 RTC\_DATE)每两个 RTCCLK 周期更新一次。如果用户希望在短时间内(小于两个 RTCCLK 周期)读取日历值，则第一次读取后必须软件清除 RTC\_INITSTS.RSYF 位。

在一些情况下，在读取日历之前需要等待 RTC\_INITSTS.RSYF 位被置 1。

- 从低功耗模式(待机模式)唤醒后，清除 RTC\_INITSTS.RSYF 位，然后等待 RTC\_INITSTS.RSYF 位重新置 1。
- 系统复位。
- 日历完成初始化。
- 日历完成同步。

### 2. 当 RTC\_CTRL.BYPS=1 时读取日历

如果 RTC\_CTRL.BYPS=1，直接从日历计数器中读取日历值。这种配置的优点是，从低功耗模式唤醒后读取日历值没有延迟，缺点是 RTC\_SUBS、RTC\_TSH 和 RTC\_DATE 的这些数据可能不是同一时刻的。

为了保证读取的日历值的正确性，需要分别读取 RTC\_SUBS、RTC\_TSH 和 RTC\_DATE 两次，然后对两次读取的数据进行比较，如果两者相等，则认为读取的数据是正确的。

注意：读 RTC\_SUBS 和 RTC\_TSH 后，需要最后读一下 RTC\_DATE 寄存器。

## 20.2.8 校准时钟输出

当 RTC\_CTRL.COEN 位置 1，PC13 引脚将输出校准时钟。如果 RTC\_CTRL.CALOSEL=0 和 RTC\_PRE.DIVA[6:0] = 0x7F，RTC\_CALIB 频率结果为  $f_{RTCCLK} / RTC\_PRE.DIVA[6:0]$ 。当 RTCCLK 频率为 32.768 kHz 时，校准输出 256Hz。由于下降沿有轻微的抖动，建议使用上升沿。

当 RTC\_CTRL.CALOSEL=1，“RTC\_PRE.DIVS[14:0]+1”是 256 的非零整数倍，RTC\_CALIB 频率由公式  $f_{RTCCLK} / (256 * (DIVA+1))$  给出。当 RTCCLK 频率为 32.768 kHz 和 RTC\_PRE.DIVA[6:0] = 0x7F 时，校准输出 1Hz。

注意：当选择 RTC\_CALIB 或 RTC\_ALARM 输出时，RTC\_OUT 引脚(PC13)被自动配置为输出。

## 20.2.9 可编程闹钟

RTC 有 2 个可编程闹钟：闹钟 A 和闹钟 B。

通过 RTC\_CTRL.ALxEN 位可以使能或关闭 RTC 闹钟。如果 Alarm 值与日历值相匹配，则 RTC\_INITSTS.ALxF 标志被置 1。如果 RTC\_CTRL.ALxIEN 使能，可以选择任意日历字段来触发闹钟中断。

闹钟输出：当 RTC\_CTRL.OUTSEL[1:0]配置后，闹钟 A 或闹钟 B 可以映射到 RTC\_ALxRM 输出，可以通过 RTC\_CTR.OPOL 位配置输出极性。

注意：当秒字段被选择(RTC\_ALARMx.MASK1 位复位)，RTC\_PRE.DIVS[14:0]必须大于 3，以保证正确操作。

## 20.2.10 闹钟配置

闹钟 A 和闹钟 B 配置步骤如下：

- 通过清除 RTC\_CTRL.ALAEN/RTC\_CTRL.ALBEN 位失能闹钟 A/闹钟 B
- 配置闹钟 x 寄存器 (RTC\_ALRMxSS/RTC\_ALARMx)
- 通过设置 RTC\_CTRL.ALAIEN/RTC\_CTRL.ALBIEN 位为 1 使能闹钟 A/闹钟 B 中断（这一步根据需要添加）
- 通过设置 RTC\_CTRL.ALAEN/ RTC\_CTRL.ALBEN 位为 1 使能闹钟 A/闹钟 B

## 20.2.11 闹钟输出

当 RTC\_CTRL.OUTSEL[1:0] !=0, RTC\_ALARM 输出功能开启。根据 RTC\_CTR.OUTSEL[1:0]的值选择闹钟 A 输出、闹钟 B 输出或者唤醒输出。

RTC\_CTRL.OPOL 位控制闹钟 A、闹钟 B 或唤醒输出的极性。

RTC\_OPT.TYPE 位控制 RTC\_ALARM 引脚开漏或者推挽输出。

选择 RTC\_CALIB 或 RTC\_ALARM 输出时，RTC\_OUT 引脚 (PC13) 会自动配置为输出。

## 20.2.12 周期性自动唤醒

16 位可编程自动加载计数器可以在达到 0 时产生周期性唤醒标志。它也可以将唤醒定时器的范围扩展到 17 位。通过设置 RTC\_CTRL.WTEN 可以启用周期性自动唤醒功能。

可以选择两种唤醒输入时钟源：

- 2、4、8 或 16 分频的 RTC 时钟 (RTCCLK)。

假设 RTCCLK 来自 LSE (32.768KHz)，在分辨率到 61us 的情况下，可以配置唤醒中断周期为 122us ~ 32s。

- 内部时钟 ck\_spre。

假设 ck\_spre 频率为 1Hz，可用唤醒时间范围为 2s ~ 36h，分辨率为 1 秒

- ◆ 当 RTC\_CTRL.WKUPSEL [2:0] = 10x，周期范围为 2s 到 18h

- ◆ 当 RTC\_CTRL.WKUPSEL[2:0] = 11x, 周期范围为 18h 到 36h.

当 RTC\_CTRL.WTEN 位设置为 1 之后, 向下计数器正在运行, 当它达到 0 时, RTC\_INITSTS.WTF 位会被置 1, 通过设置 RTC\_CTRL.WTIEN 位为 1, 当周期性唤醒中断被启用触发时, 设备可以退出低功耗模式。

周期性唤醒输出: 当 RTC\_CTRL.OUTSEL[1:0]选择周期性唤醒后可以映射到 RTC\_ALxRM 输出, 自动将 RTC\_OUT 引脚(PC13)配置为输出, 输出极性可由 RTC\_CTR.OPOL 位配置。

### 20.2.13 唤醒定时器配置

唤醒计时器自动重新加载值配置如下:

- 通过清除 RTC\_CTRL.WTEN 关闭唤醒定时器, 然后等待 RTC\_INITSTS.WTWF 标志位被置 1
- 通过设置 RTC\_CTRL.WKUPSEL[2:0]选择唤醒定时器时钟
- 通过设置 RTC\_WKUPT.WKUPT[15:0]配置唤醒自动重加载值
- 通过设置 RTC\_CTRL.WTIEN 位使能唤醒中断 (此步可根据需要选择)
- 通过设置 RTC\_CTRL.WTEN 位开启唤醒定时器

### 20.2.14 时间戳功能

时间戳可以通过将 RTC\_CTRL.TSEN 位设置为 1 来启用。当在 RTC\_TS 引脚上检测到时间戳事件时, 该事件的日历值将存储在时间戳寄存器 (RTC\_TSSS、RTC\_TST、RTC\_TSD) 中, 并且 RTC\_INITSTS.TISF 位被设置为 1。如果 RTC\_CTRL.TSIEN 设置为 1, 则时间戳事件可以产生中断。如果在 RTC\_INITSTS.TISF 已经设置为 1 时检测到新的时间戳事件, 则硬件将 RTC\_INITSTS.TISOVF 标志设置为 1, 并且时间戳寄存器 (RTC\_TST 和 RTC\_TSD) 将继续保存前一个事件的值, 这意味着当 RTC\_INITSTS.TISF=1 时, 时间戳寄存器 (RTC\_TST 和 RTC\_TSD) 数据不会改变。

在同步过程引起的时间戳事件再次发生后, RTC\_INITSTS.TISF 在 2 个 RTC\_CLK 周期内设置为 1。RTC\_INITSTS.TISOVF 的生成没有延迟。这意味着如果两个时间戳事件非常接近, 这可能导致 RTC\_INITSTS.TISOVF 为“1”而 RTC\_INITSTS.TISF 为“0”。因此, 在检测到 RTC\_INITSTS.TISF 为“1”后, 再检测 RTC\_INITSTS.TISOVF 位。当 RTC\_TMPCFG.TPTS 位设置为 1 时, 入侵事件可以触发时间戳事件。

如果启用时间戳事件, 时间戳将在时间戳寄存器中捕获读取的日历。当入侵事件和时间戳事件都启用时, 入侵事件也会导致时间戳捕获。时间戳事件可以在 EXTI 选择的 16 个 GPIO 端口中的任何一个上生成。通过设置相应的 EXTI\_TS\_SEL.TSSEL[3:0] 位来选择任一端口的 GPIO 引脚。

### 20.2.15 入侵检测

共有三个入侵检测引脚, RTC\_TAMP1 引脚为 PC13, RTC\_TAMP2 引脚为 PA0, RTC\_TAMP3 引脚为 PA8。RTC\_TAMPx 引脚可用作入侵事件检测功能输入引脚。有两种检测模式, 边缘检测模式和可配置滤波功能的电平检测模式。

当检测到 RTC\_TAMPx 事件时, 如果 RTC\_TMPCFG.TPxNOE=0, 则 RTC\_BKP(1~20)寄存器将被擦除。

#### 入侵检测初始化

共有三个入侵检测引脚, 每个引脚都可以独立配置。用户需要在设置 RTC\_TMPCFG.TPxEN 位之前配置入侵检测。当入侵检测使能后检测到入侵事件时, 如果 RTC\_TMPCFG.TPxINTEN 位置 1, 则入侵事件可以产

生中断并且 RTC\_INITSTS.TAMxF 位将被置 1。

当 RTC\_INITSTS.TAMxF 位为 1 时，无法检测到同一引脚上的新入侵事件。

### 入侵事件的时间戳

当 RTC\_TMPCFG.TPTS 位设置为 1 时，任何入侵事件都可以触发时间戳事件，并且 RTC\_INITSTS.TISF 位和 RTC\_INITSTS.TISOVF 位将被设置为正常的时间戳事件。

### 入侵输入的边缘检测

当 RTC\_TMPCFG.TPFLT[1:0] 位设置为 0 时，入侵检测设置为边沿检测，上升沿或下降沿由 RTC\_TMPCFG.TPxTRG 位控制。当检测到相应的边沿时，RTC\_TAMPx 引脚将产生一个入侵检测事件。

由于 RTC\_BKP(1~20) 可以在检测到入侵事件时复位，因此需要确保不会同时发生入侵事件检测和写入 RTC\_BKP(1~20)。建议在写入 RTC\_BKP(1~20) 后启动入侵检测功能。

### RTC\_TAMPx 输入的滤波电平检测

当 RTC\_TMPCFG.TPFLT[1:0]位设置为 1/2/3 时，入侵检测设置为电平检测。RTC\_TMPCFG.TPFLT[1:0] 的值决定了采样次数。

每次采样前可通过入侵引脚的内部上拉电阻进行预充电，预充电时间由 RTC\_TMPCFG.TPPRCH[1:0]位控制。当 RTC\_TMPCFG.TPPUDIS 设置为 1 时，预充电将被禁用。

使用 RTC\_TMPCFG.TPFREQ[2:0] 确定电平检测的采样频率，可以优化入侵检测延迟和上拉功耗之间的最佳平衡。

## 20.2.16 夏时令功能配置

夏令时功能可通过 RTC\_CTRL.SU1H、RTC\_CTRL.AD1H 和 RTC\_CTRL.BAKP 位控制。设置 RTC\_CTRL.SU1H 位为 1 时日历会减一小时，设置 RTC\_CTRL.AD1H 为 1 时会增加一小时。RTC\_CTRL.BAKP 位可用于记住或不记住此调整。

## 20.2.17 RTC 亚秒寄存器位移操作

当日历的值与外部精密时钟相比有亚秒级的偏差时，可以使用移位功能来提高日历的精度。

日历可以使用 RTC\_SCTRL.AD1S 和 RTC\_SCTRL.SUBF[14:0]位来控制最大延迟或提前 1s。调整分辨率为 1/(RTC\_PRE.DIVS[14:0]+1)，表示 RTC\_PRE.DIVS[14:0]的值越大，分辨率越高。为了使同步预分频器输出保持在 1Hz，RTC\_PRE.DIVS[14:0]越高意味着 RTC\_PRE.DIVA[6:0]越低，则功耗越大。

注意：在开始移位操作之前，用户必须检查 RTC\_SUBS.SS[15]位是否为 0。

每当写入 RTC\_SCTRL 寄存器时，硬件都会设置 RTC\_INITSTS.SHOPF 标志，表明平移操作处于挂起状态。一旦平移操作完成，该位由硬件清零。

## 20.2.18 RTC 数字时钟精密校准

数字精密校准是通过调整校准周期内的 RTC 时钟脉冲数来实现的。数字精度校准分辨率为 0.954 PPM，范围从 -487.1 PPM 到 +488.5 PPM。

当输入频率为 32768 Hz 时，校准周期可配置为  $2^{20}/2^{19}/2^{18}$  个 RTCCLK 周期或 32/16/8 秒。精密校准寄存器

(RTC\_CALIB) 表示将在指定周期内减少 RTC\_CALIB.CM[8:0] 个 RTCCLK 时钟周期。

RTC\_CALIB.CM[8:0] 的值表示在指定周期内要减少的 RTCCLK 脉冲数。RTC\_CALIB.CP 可用于增加 488.5PPM，每 2<sup>11</sup> 个 RTCCLK 周期将插入一个 RTCCLK 脉冲。

当 RTC\_CALIB.CM[8:0] 和 RTC\_CALIB.CP 组合使用时，增加的周期范围为 -511 到 +512 个 RTCCLK 周期，校准范围为 -487.1ppm 到 +488.5ppm，分辨率约为 0.954ppm。

有效校准频率 ( $f_{CAL}$ ) 可使用以下公式计算：

$$f_{CAL} = f_{RTCCLK} * \left(1 + \frac{RTC\_CALIB.CP * 512 - RTC\_CALIB.CM[8:0]}{2^n + RTC\_CALIB.CM[8:0] - RTC\_CALIB.CP * 512}\right)$$

注意：n=20/19/18

### 当 RTC\_PRE.DIVA[6:0]<3 时校准

当异步预分频器值 (RTC\_PRE.DIVA[6:0]) 小于 3 时，不能将 RTC\_CALIB.CP 设置为 1，如果 RTC\_CALIB.CP 值已设置为 1，则将被忽略。

假设 RTCCLK 频率为 32768Hz，当 RTC\_PRE.DIVA[6:0]<3 时，RTC\_PRE.DIVS[14:0] 的值应该减小：

- 当 RTC\_PRE.DIVA[6:0]=2, RTC\_PRE.DIVS[14:0]=8189.
- 当 RTC\_PRE.DIVA[6:0]=1, RTC\_PRE.DIVS[14:0]=16379.
- 当 RTC\_PRE.DIVA[6:0]=0, RTC\_PRE.DIVS[14:0]=32759.

有效校准频率 ( $f_{CAL}$ ) 可使用以下公式计算：

$$f_{CAL} = f_{RTCCLK} * \left(1 + \frac{256 - RTC\_CALIB.CM[8:0]}{2^n + RTC\_CALIB.CM[8:0] - 256}\right)$$

注意：n=20/19/18

### 验证 RTC 校准

RTC 输出 1Hz 波形，用于测量和验证 RTC 精度。

在有限测量周期内测量 RTC 频率时，最多可能出现 2 个 RTCCLK 周期测量误差。如果测量周期与校准周期相同，则可以消除误差。

- 校准周期为 32 秒（默认）

使用精确的 32 秒周期测量 1Hz 校准输出可以确保测量误差在 0.447ppm 以内（32 秒内为 0.5 个 RTCCLK 周期）。

- 校准周期为 16 秒。

使用精确的 16 秒周期测量 1Hz 校准输出可以确保测量误差在 0.954ppm 以内（16 秒内为 0.5 个 RTCCLK 周期）。

- 校准周期为 8 秒。

使用精确的 8 秒周期测量 1Hz 校准输出可以确保测量误差在 1.907ppm 以内（8 秒内为 0.5 个 RTCCLK 周期）。

### 动态重新校准

当 RTC\_INITSTS.INITF=0 时，RTC\_CALIB 寄存器可以通过以下步骤更新：

- 等待 RTC\_INITSTS.RECPF=0
- 一个新值被写入 RTC\_CALIB，然后 RTC\_INITSTS.RECPF 位自动置 1
- 新的校准设置将在数据写入 RTC\_CALIB 后的 3 个 ck\_apre 周期内生效

## 20.2.19 RTC 低功耗模式

RTC 在低功耗模式下的工作状态。

低功耗模式	RTC 工作状态	退出低功耗模式
SLEEP	正常工作	RTC 中断
STOP0	RTC 时钟源为 LSE 或 LSI 时正常工作	闹钟 A、闹钟 B、定期唤醒、入侵事件和时间戳事件
STOP2	RTC 时钟源为 LSE 或 LSI 时正常工作	闹钟 A、闹钟 B、定期唤醒、入侵事件和时间戳事件
STANDBY	RTC 时钟源为 LSE 或 LSI 时正常工作	闹钟 A、闹钟 B、定期唤醒、入侵事件和时间戳事件

注意：当使用 RTC 事件将 MCU 从 STANDBY 模式唤醒时，需要将 PWR\_CTRLSTS.WKUPRTCEN 位使能。

## 20.3 RTC 寄存器

### 20.3.1 RTC 寄存器总览

表 20-2 RTC 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0											
000h	RTC_TSH	Reserved				APM		HOU[1:0]				HOU[3:0]				Reserved	MIT[2:0]		MIU[3:0]				Reserved	SCT[2:0]		SCU[3:0]																		
						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0														
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
004h	RTC_DATE	Reserved				YRT[3:0]				YRU[3:0]				WDU[2:0]				Reserved	MOT		MOU[3:0]				Reserved	DAT[1:0]		DAU[3:0]																
						0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	1													
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1												
008h	RTC_CTRL	Reserved				COEN	OUTSEL[1:0]		OPOL		CALOSEL		BAKP		SU1H		AD1H		TSEN		WTEN		ALBEN		ALAEN		HFMT		BYP		REFCLKEN		TEDGE		WKUPSEL[2:0]									
						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
00Ch	RTC_INITSTS	Reserved				RECPL				TAM3F		TAM2F		TAM1F		TISOVF		TISF		WTF		WTEN		ALBF		ALAF		INITF		RSYF		INITSF		SHOPF		WTWF		ALBWF						
						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
	Reset Value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0												
010h	RTC_PRE	Reserved				DIVA[6:0]								DIVS[14:0]																REFCLKEN		TEDGE		WKUPSEL[2:0]										
						1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1											
	Reset Value					1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1											
014h	RTC_WKUPT	Reserved				WKUPT[15:0]																	WKUPT[15:0]																REFCLKEN		TEDGE		WKUPSEL[2:0]	
						1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1							
	Reset Value					1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1							

### 20.3.2 RTC 日历时间寄存器（RTC\_TSH）

偏移地址: 0x00

复位值: 0x0000 0000

31	Reserved						23	22	21	20	19	16
							APM	HOT[1:0]			HOU[3:0]	
15	14	12	11	8	7	rw	6	rw	4	3	rw	0
Reserved	MIT[2:0]		MIU[3:0]		Reserved	SCT[2:0]				SCU[3:0]		
	rw		rw			rw		rw		rw		

位域	名称	描述
31:23	Reserved	保留，必须保持复位值。
22	APM	AM/PM 格式。 0: AM 格式或者 24 小时格式 1: PM 格式
21:20	HOT[1:0]	小时的十位(BCD 格式)。
19:16	HOU[3:0]	小时的个位(BCD 格式)。
15	Reserved	保留，必需保持复位值。
14:12	MIT [2: 0]	分钟的十位(BCD 格式)。
11:8	MIU[3:0]	分钟的个位(BCD 格式)。
7	Reserved	保留，必需保持复位值。
6:4	SCT[2:0]	秒的十位(BCD 格式)。
3:0	SCU[3:0]	秒的个位(BCD 格式)。

### 20.3.3 RTC 日历日期寄存器 (RTC\_DATE)

偏移地址: 0x04

复位值: 0x0000 2101

31	Reserved						24	23	YRT[3:0]			19	16
15	13	12	11	8	7	6	r	5	4	3	r	0	
WDU[2:0]	MOT		MOU[3:0]		Reserved	SCT[1:0]		DAT[1:0]		DAU[3:0]			
r	r		r			r		r		r			

位域	名称	描述
31:24	Reserved	保留，必须保持复位值。
23:20	YRT[3:0]	年的十位(BCD 格式)。
19:16	YRU[3:0]	年的个位(BCD 格式)。
15:13	WDU[2:0]	星期几 000: 禁止 001: 星期一 ... 111: 星期天
12	MOT	月的十位(BCD 格式)。
11:8	MOU[3:0]	月的个位(BCD 格式)。

位域	名称	描述
7:6	Reserved	保留，必须保持复位值。
5:4	DAT[1:0]	日期的十位(BCD 格式)。
3:0	DAU[3:0]	日期的个位(BCD 格式)。

#### 20.3.4 RTC 控制寄存器(RTC\_CTRL)

偏移地址： 0x08

复位值: 0x0000 0000

位域	名称	描述
31:24	Reserved	保留，必须保持复位值。
23	COEN	校准输出使能。 0: 校准输出禁止 1: 校准输出开启
22:21	OUTSEL[1:0]	输出选择位。 该位用来选择要连接到 RTC_ALARM 输出的标志。 00: 输出禁止 01: 阵列 A 输出开启 10: 阵列 B 输出开启 11: 唤醒输出开启
20	OPOL	输出极性位。 此位用于配置 RTC_ALARM 输出的极性。 0: 当 ALAF/ALBF/WTF 标志位置 1(取决于 OUTSEL [1:0]), 该引脚输出高电平 1: 当 ALAF/ALBF/WTF 标志位置 1(取决于 OUTSEL [1:0]), 该引脚输出低电平
19	CALOSEL	校准输出选择位。 当 COEN=1 时, 该位选择在 RTC_CALIB 上输出哪个信号。 在 RTCCLK 为 32.768 kHz 且预分频为默认值(RTC_PRE.DIVA[6:0]=127 和 RTC_PRE.DIVS[14:0]=255) 的条件下, 这些频率有效。 0: 校准输出 256Hz(默认预分频设置) 1: 校准输出 1Hz(默认预分频设置)
18	BAKP	这个位可以由用户写入, 以记住是否执行了夏令时的更改。
17	SU1H	减去 1 小时位(冬季时间更改)。 当设置此位时, 如果当前小时不是 0, 则从日历时间中减去 1 小时。这个位总是

位域	名称	描述
		被读取为 0。当当前小时为 0 时，设置此位无效。 0: 无使用 1: 用当前时间减去 1 小时。这可以用于冬季改变户外初始化模式
16	AD1H	加 1 小时位(夏季时间更改)。 设置此位后，将 1 小时添加到日历时间中。这个位总是被读为 0。 0: 无使用 1: 用当前时间加上 1 小时。这可以用于夏季改变户外初始化模式
15	TSIEN	时间戳中断使能位。 0: 时间戳中断禁止 1: 时间戳中断开启
14	WTIEN	唤醒定时器中断使能位。 0: 唤醒定时器中断禁止 1: 唤醒定时器中断开启
13	ALBIEN	闹钟 B 中断使能位。 0: 闹钟 B 中断禁止 1: 闹钟 B 中断开启
12	ALAIEN	闹钟 A 中断使能位。 0: 闹钟 A 中断禁止 1: 闹钟 A 中断开启
11	TSEN	时间戳使能位。 0: 时间戳禁止 1: 时间戳开启
10	WTEN	唤醒定时器使能位。 0: 唤醒定时器禁止 1: 唤醒定时器开启
9	ALBEN	闹钟 B 使能位。 0: 闹钟 B 禁止 1: 闹钟 B 开启
8	ALAEN	闹钟 A 使能位。 0: 闹钟 A 禁止 1: 闹钟 A 开启
7	Reserved	保留，必须保持复位值。
6	HFMT	小时格式位。 0: 24 小时格式 1: AM/PM 格式
5	BYPS	旁路影子寄存器位。 0: 日历值(从 RTC_SUBS、RTC_TSH 和 RTC_DATE 读取时)取自影子寄存器，影子寄存器每两个 RTCCLK 周期更新一次。 1: 日历值(从 RTC_SUBS、RTC_TSH 和 RTC_DATE 读取时)直接从日历计数器中获取。 <i>注意：如果 APBI 时钟的频率小于 RTCCLK 的 7 倍，则 BYPS 必须设置为 1</i>
4	REFCLKEN	RTC_REFIN 参考时钟检测位(50 或 60hz)。

位域	名称	描述
		0: RTC_REFIN 检测禁止 1: RTC_REFIN 检测开启 注意: DIVS 必须为 0x00FF
3	TEDGE	时间戳事件触发沿配置位。 0: RTC_TS 输入上升沿生成一个时间戳事件 1: RTC_TS 输入下降沿生成一个时间戳事件 注意: 更改 TEDGE 时必须重置 RTC_CTRL.TSEN, 以避免 RTC_INITSTS.TISF 外置 1。
2:0	WKUPSEL[2:0]	唤醒时钟选择位。 000: 选择 RTC/16 时钟 001: 选择 RTC/8 时钟 010: 选择 RTC/4 时钟 011: 选择 RTC/2 时钟 10x: 选择 ck_spre(通常 1Hz)时钟 11x: 选择 ck_spre(通常 1Hz)时钟并且唤醒定时器计数器值配置成 $2^{16}$

### 20.3.5 RTC 初始状态寄存器 (RTC\_INITSTS)

偏移地址: 0x0C

复位值: 0x0000 0007

31	Reserved															17	16
																RECPF	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	r	0	
TAM3F	TAM2F	TAM1F	TISOVF	TISF	WTF	ALBF	ALAF	INITM	INITF	RSYF	INITSF	SHOPF	WTWF	ALBWF	ALAWF		
rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	r	rc_w0	r	r	r	r	r	r	

位域	名称	描述
31:17	Reserved	保留, 必须保持复位值。
16	RECPF	重新校准挂起标志位。 当软件写入 RTC_CALIB 寄存器时, RECPF 状态标志自动设置为 1, 表示 RTC_CALIB 寄存器被阻塞。当考虑到新的校准设置时, 这个位将恢复为 0。
15	TAM3F	RTC_TAMP3 检测标志位。 当在 RTC_TAMP3 输入口上检测到侵入事件时, 硬件将设置此标志。通过软件写 0 清除
14	TAM2F	RTC_TAMP2 检测标志位。 当在 RTC_TAMP2 输入口上检测到侵入事件时, 硬件将设置此标志。通过软件写 0 清除
13	TAM1F	RTC_TAMP1 检测标志位。 当在 RTC_TAMP1 输入口上检测到侵入事件时, 硬件将设置此标志。通过软件写 0 清除
12	TISOVF	时间戳溢出标志位。

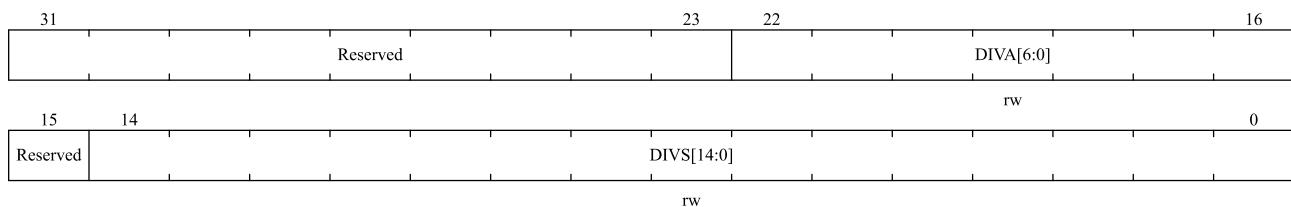
位域	名称	描述
		当时间戳事件发生的同时 TISF 位已经被置 1 时，硬件将此标志置 1。建议在清除 TISF 位之后再检查并清除 TISOVF 位。否则，如果时间戳事件恰好在清除 TISF 位之前刚刚发生，则溢出事件可能会被漏掉。
11	TISF	时间戳标志位。 当发生时间戳事件时，硬件将设置此标志。此标志通过写入 0 被软件清除。
10	WTF	唤醒定时器标志位。 当唤醒自动重载计数器达到 0，硬件将设置此标志。此标志通过写入 0 被软件清除。在 WTF 再次设置为 1 之前，此标志必须由软件至少在 1.5 RTCCLK 周期内清除。
9	ALBF	闹钟 B 标志位。 当时间/日期寄存器(RTC_TSH 和 RTC_DATE)与闹钟 B 寄存器(RTC_ALARMB)匹配时，硬件将设置此标志。此标志通过写入 0 被软件清除。
8	ALAF	闹钟 A 标志位。 当时间/日期寄存器(RTC_TSH 和 RTC_DATE)与闹钟 A 寄存器(RTC_ALARMA)匹配时，硬件将设置此标志。此标志通过写入 0 被软件清除。
7	INITM	进入初始化模式 0：自由运行模式 1：进入初始化模式，设置日历时间值、日期值、预分频值。
6	INITF	初始标志位。 当这个位设置为 1 时，RTC 处于初始化状态，可以更新时间、日期和预分频寄存器。 0：日历寄存器更新禁止 1：日历寄存器更新允许
5	RSYF	寄存器同步标志位。 当日历值被复制到影子寄存器中时，该标志由硬件设置为“1”。当处于初始化模式、移位操作挂起 (SHOPF=1) 或处于旁路影子寄存器模式 (RTC_CTRL.BYPS=1) 时，该位由硬件清零，该位也可以通过软件清零。 在初始化模式下，该位通过软件或硬件清除。 0：日历影子寄存器尚未同步 1：日历影子寄存器同步
4	INITSF	初始状态标志位。 当日历年字段不等于 0 (备份域复位状态)时，由硬件设置此位。 0：日历没有被初始化 1：日历已经被初始化
3	SHOPF	平移操作挂起标志位。 当向 RTC_SCTRL 寄存器写入一个平移操作时，硬件立即设置此标志。当执行相应的平移操作时，硬件将清除它。写入 SHOPF 位不起作用。 0：没有位移操作挂起 1：有位移操作挂起
2	WTWF	唤醒定时器写标志位。 0：唤醒时间配置更新不允许 1：唤醒时间配置更新允许

位域	名称	描述
1	ALBWF	闹钟 B 写标志。 当 RTC_CTRL.ALBEN 位设置为 0，同时闹钟 B 值可更改时，硬件将该位置 1。它在初始化模式下被硬件清除。 0：闹钟 B 更新不允许 1：闹钟 B 更新允许
0	ALAWF	闹钟 A 写标志。 当 RTC_CTRL.ALAREN 位设置为 0，同时闹钟 A 可更改时，硬件将该位置 1。它在初始化模式下被硬件清除。 0：闹钟 A 更新不允许 1：闹钟 A 更新允许

### 20.3.6 RTC 预分频寄存器(RTC\_PRE)

偏移地址: 0x10

复位值: 0x007F 00FF

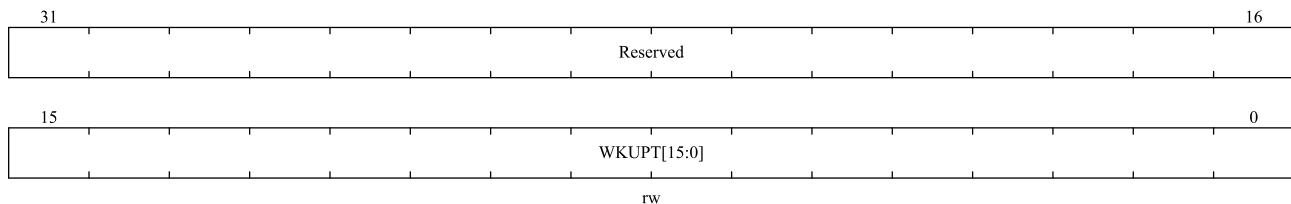


位域	名称	描述
31:23	Reserved	保留，必须保持复位值。
22:16	DIVA[6:0]	异步分频参数位。 $f_{ck\_apre} = \text{RTCCLK}/(\text{DIVA}[6:0]+1)$
15	Reserved	保留，必须保持复位值。
14:0	DIVS[14:0]	同步分频位。 $f_{ck\_spre} = f_{ck\_apre}/(\text{DIVS}[14:0]+1)$

### 20.3.7 RTC 唤醒定时器寄存器(RTC\_WKUPT)

偏移地址: 0x14

复位值: 0x0000 FFFF



位域	名称	描述
31:16	Reserved	保留，必须保持复位值。
15:0	WKUPT[15:0]	<p>唤醒自动重载值位 当 RTC_CTRL.WTEN=1 时，每 (WKUPT[15:0] + 1) 个 ck_wut 周期设置 RTC_INITSTS.WTF 标志。当 RTC_CTRL.WKUPSEL[2]=1 时，唤醒定时器变为 17 位。该位不能配成 0。</p> <p>注意： 这个寄存器的变化（如第二次设置或以后的设置）需要在唤醒中断中进行更改，否则更改后的设置不会立即生效，而是在下次唤醒后生效；特别是当 RTC_CTRL.WKUPSEL[2:0] 设置为 010 时，修改后的设置不会立即生效，而是在下一个周期唤醒后生效。</p>

### 20.3.8 RTC 闹钟 A 寄存器(RTC\_ALARMA)

偏移地址：0x1C

复位值：0x0000 0000

31	30	29	28	27	24	23	22	21	20	19	16
MASK4	WKDSEL	DTT[1:0]		DTU[3:0]		MASK3	APM	HOT[1:0]		HOU[3:0]	
rw 15	rw 14	rw 12	rw 11	rw 8	rw 7	rw 6	rw 4	rw 3	rw	rw 0	
MASK2	MIT[2:0]		MIU[3:0]		MASK1	SET[2:0]		SEU[3:0]			
rw	rw		rw		rw	rw		rw		rw	

位域	名称	描述
31	MASK4	闹钟日期掩码位。 0: 日期/日匹配 1: 日期/日不匹配
30	WKDSEL	星期几选择位。 0: DTU[3:0]代表日期的个位 1: DTU[3:0]代表星期几。DTT[1:0]为无关位
29:28	DTT[1:0]	日期的十位(BCD 格式)。
27:24	DTU[3:0]	日期的个位(BCD 格式)
23	MASK3	闹钟小时掩码位。 0: 小时匹配 1: 小时不匹配
22	APM	AM/PM 符号位。 0: AM 或 24 小时制 1: PM
21:20	HOT[1:0]	小时的十位(BCD 格式)。
19:16	HOU[3:0]	小时的个位(BCD 格式)。
15	MASK2	闹钟分钟掩码位。 0: 分钟匹配 1: 分钟不匹配

位域	名称	描述
14:12	MIT[2:0]	分钟的十位(BCD 格式)。
11:8	MIU[3:0]	分钟的个位(BCD 格式)。
7	MASK1	闹钟秒掩码位。 0: 秒匹配 1: 秒不匹配
6:4	SET[2:0]	秒的十位(BCD 格式)。
3:0	SEU[3:0]	秒的个位(BCD 格式)。

### 20.3.9 RTC 闹钟 B 寄存器 (RTC\_ALARM\_B)

偏移地址: 0x20

复位值: 0x0000 0000

31	30	29	28	27	24	23	22	21	20	19	16
MASK4	WKDSEL	DTT[1:0]		DTU[3:0]	MASK3	APM	HOT[1:0]		HOU[3:0]		
rw 15	rw 14	rw 12	rw 11	rw 8	rw 7	rw 6	rw 4	rw 3	rw 0		
MASK2	MIT[2:0]		MIU[3:0]	MASK1	SET[2:0]		SEU[3:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

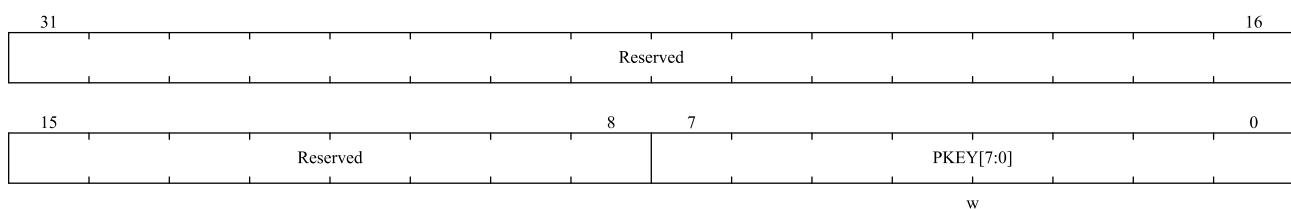
位域	名称	描述
31	MASK4	闹钟日期掩码位。 0: 日期/日匹配 1: 日期/日不匹配
30	WKDSEL	星期几选择位。 0: DTU[3:0]代表日期的个位 1: DTU[3:0]代表星期几。DTT[1:0]为无关位
29:28	DTT[1:0]	日期的十位(BCD 格式)。
27:24	DTU[3:0]	日期的个位(BCD 格式)
23	MASK3	闹钟小时掩码位。 0: 小时匹配 1: 小时不匹配
22	APM	AM/PM 符号位。 0: AM 或 24 小时制 1: PM
21:20	HOT[1:0]	小时的十位(BCD 格式)。
19:16	HOU[3:0]	小时的个位(BCD 格式)。
15	MASK2	闹钟分钟掩码位。 0: 分钟匹配 1: 分钟不匹配
14:12	MIT[2:0]	分钟的十位(BCD 格式)。
11:8	MIU[3:0]	分钟的个位(BCD 格式)。

位域	名称	描述
7	MASK1	闹钟秒掩码位。 0: 秒匹配 1: 秒不匹配
6:4	SET[2:0]	秒的十位(BCD 格式)。
3:0	SEU[3:0]	秒的个位(BCD 格式)。

### 20.3.10 RTC 写保护寄存器(RTC\_WRP)

偏移地址: 0x24

复位值: 0x0000 0000

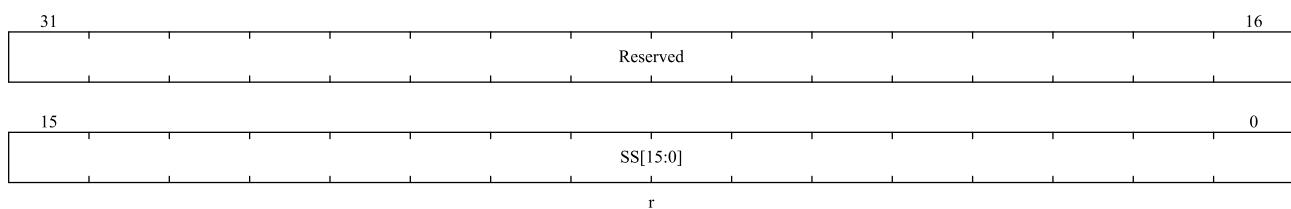


位域	名称	描述
31:8	Reserved	保留, 必须保持复位值。
7:0	PKEY[7:0]	写保护秘钥 读取该字节总是返回 0x00。 有关如何解锁 RTC 寄存器写保护的详细信息, 请参阅 RTC 写保护寄存器章节。

### 20.3.11 RTC 亚秒寄存器(RTC\_SUBS)

偏移地址: 0x28

复位值: 0x0000 0000



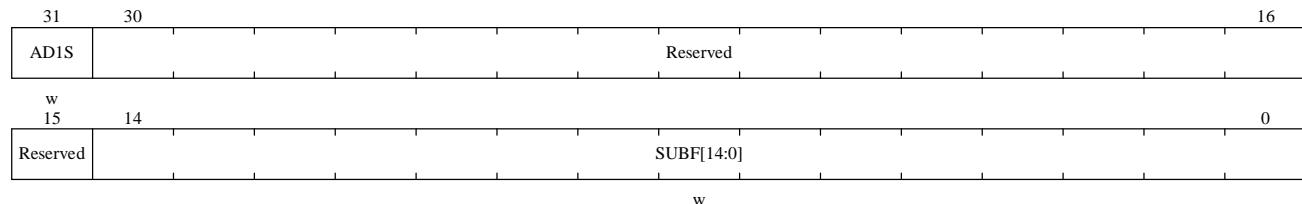
位域	名称	描述
31:16	Reserved	保留, 必须保持复位值。
15:0	SS[15:0]	亚秒值。 该值是同步预分频器计数器值。 此亚秒值由以下公式计算: $\text{亚秒值} = (\text{RTC\_PRE.DIVS}[14:0]-\text{SS})/(\text{RTC\_PRE.DIVS}[14:0]+1)$ 注意: SS[15:0]只有在移位操作完成后才能大于 RTC_PRE.DIVS[14:0]。 在这种情况下, 正确的时间/日期比 RTC_TSH/RTC_DATE 指示的时间/日期慢一

位域	名称	描述
		秒。

### 20.3.12 RTC 平移控制寄存器(RTC\_SCTRL)

偏移地址: 0x2C

复位值: 0x0000 0000

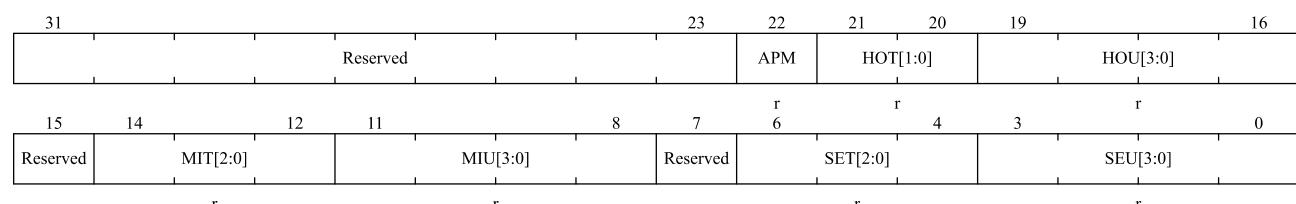


位域	名称	描述
31	AD1S	加一秒 0: 不加一秒。 1: 时钟/日历增加一秒 该位只能写入且读取为零。当 RTC_INITSTS.SHOPF=1 时, 写入该位没有影响。
30:15	Reserved	保留, 必须保持复位值。
14:0	SUBF[14:0]	减去亚秒值位 这些位只能写入且读取为零。当 RTC_INITSTS.SHOPF=1 时, 写入该位没有影响。写入 SUBF[14:0]的值被添加到同步预分频计数器, 时钟将延迟: 延迟 (秒) = (SUBF[14:0]+1) / (DIVS[14:0] + 1) AD1S 位可以与 SUBF[14:0] 位一起使用: 提前 (秒) = (1 - ((SUBF[14:0]+1) / (DIVS[14:0] + 1))). 注意: RTC_INITSTS.RSYF 位将在写入 SUBF[14:0] 时被清除。当 RTC_INITSTS.RSYF=1 时, 影子寄存器已更新为平移后的时间。

### 20.3.13 RTC 时间戳时间寄存器 (RTC\_TST)

偏移地址: 0x30

复位值: 0x0000 0000



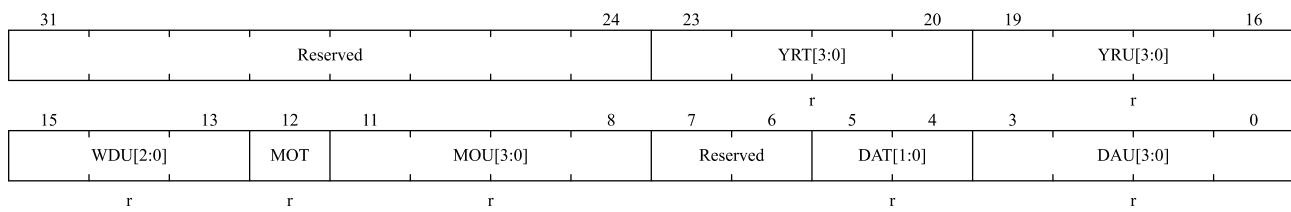
位域	名称	描述
31:23	Reserved	保留, 必须保持复位值。
22	APM	AM/PM 符号位。

位域	名称	描述
		0: AM 或 24 小时制 1: PM
21:20	HOT[1:0]	小时的十位(BCD 格式)。
19:16	HOU[3:0]	小时的个位(BCD 格式)。
15	Reserved	保留, 必须保持复位值。
14:12	MIT[2:0]	分钟的十位(BCD 格式)。
11:8	MIU[3:0]	分钟的个位(BCD 格式)。
7	Reserved	保留, 必须保持复位值。
6:4	SET[2:0]	秒的十位(BCD 格式)。
3:0	SEU[3:0]	秒的个位(BCD 格式)。

### 20.3.14 RTC 时间戳日期寄存器 (RTC\_TSD)

偏移地址: 0x34

复位值: 0x0000 0000

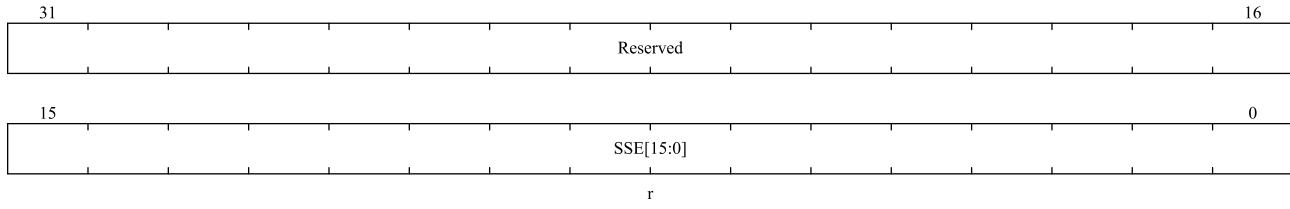


位域	名称	描述
31:24	Reserved	保留, 必须保持复位值。
23:20	YRT[3:0]	年的十位(BCD 格式)。
19:16	YRU[3:0]	年的个位(BCD 格式)。
15:13	WDU[2:0]	星期几 000: 禁止 001: 星期一 ... 111: 星期天
12	MOT	月份的十位(BCD 格式)。
11:8	MOU[3:0]	月份的个位(BCD 格式)。
7:6	Reserved	保留, 必须保持复位值。
5:4	DAT[1:0]	日期的十位(BCD 格式)。
3:0	DAU[3:0]	日期的个位(BCD 格式)。

### 20.3.15 RTC 时间戳亚秒寄存器(RTC\_TSST)

偏移地址: 0x38

复位值: 0x0000 0000

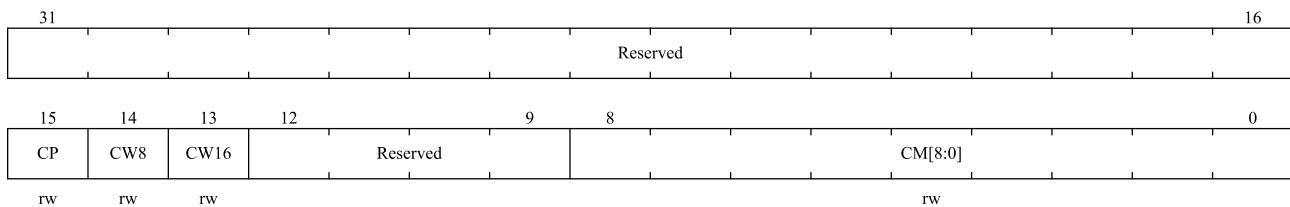


位域	名称	描述
31:16	Reserved	保留，必须保持复位值。
15:0	SSE[15:0]	亚秒值 SSE[15:0] 是同步预分频计数器中的值。 亚秒值由以下公式提供： $\text{亚秒值} = (\text{RTC\_PRE.DIVS}[14:0] - \text{SSE}[15:0]) / (\text{RTC\_PRE.DIVS}[14:0] + 1)$ 注意：SSE[15:0]只能在移位操作后大于 RTC_PRE.DIVS[14:0]。在这种情况下，正确的时间/日期比 RTC_TSH/RTC_DATE 指示的时间少一秒。

### 20.3.16 RTC 校准寄存器(RTC\_CALIB)

偏移地址: 0x3C

复位值: 0x0000 0000



位域	名称	描述
31:16	Reserved	保留，必须保持复位值。
15	CP	将 RTC 频率提高 488.5 ppm。 此功能与 CM[8:0]一起使用。当 RTCCLK 频率为 32768 Hz 时，在 32 秒窗口期间添加的 RTCCLK 脉冲数为 $((512 * CP) - CM[8:0])$ 。 0: 不增加 RTCCLK 脉冲 1: 每 $2^{11}$ 个脉冲有效插入一个 RTCCLK 脉冲
14	CW8	使用 8 秒校准周期位。 0: 不使用 1: 选择 8 秒校准周期 注意：当 CW8 = 1 时，CM[1:0]将始终保持为‘00’
13	CW16	使用 16 秒校准周期位。 0: 不使用 1: 选择 16 秒校准周期，如果 CW8 = 1，则不能将该位置 1 注意：当 CW16 = 1 时，CM[0]将始终保持为‘0’
12:9	Reserved	保留，必须保持复位值。
8:0	CM[8:0]	负校准位。 $2^{20}$ 个 RTCCLK 脉冲中的屏蔽脉冲数。这有效地降低了分辨率为 0.9537 ppm 的

位域	名称	描述
		日历频率。

### 20.3.17 RTC 入侵配置寄存器 (RTC\_TMPCFG)

偏移地址: 0x40

复位值: 0x0000 0000

31	Reserved	TP3MF	TP3NOE	TP3INTEN	TP2MF	TP2NOE	TP2INTEN	TP1MF	TP1NOE	TP1INTEN	16
15	14	13	12	11	10	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3
TPPUDIS	TPPRCH[1:0]	TPFLT[1:0]	TPFREQ[2:0]	TPTS	TP3TRG	TP3EN	TP2TRG	TP2EN	TPINTEN	TP1TRG	TP1EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位域	名称	描述
31:25	Reserved	保留, 必须保持复位值。
24	TP3MF	入侵 3 掩码标志。 0: 不屏蔽入侵 3 事件。 1: 屏蔽入侵 3 事件。 注意: 当 TP3MF 置位时, 不得使能 Tamper 3 中断。
23	TP3NOE	入侵 3 不擦除位。 0: 入侵 3 事件擦除备份寄存器 1: 入侵 3 事件不擦除备份寄存器
22	TP3INTEN	入侵 3 中断使能位。 0: TPINTEN = 0 时禁止入侵 3 中断 1: 使能入侵 3 中断
21	TP2MF	入侵 2 掩码标志。 0: 不屏蔽入侵 2 事件。 1: 屏蔽入侵 2 事件。 注意: 当 TP2MF 置位时, 不得使能 Tamper 2 中断。
20	TP2NOE	入侵 2 不擦除位。 0: 入侵 2 事件擦除备份寄存器 1: 入侵 2 事件不擦除备份寄存器
19	TP2INTEN	入侵 2 中断使能位。 0: TPINTEN = 0 时禁止入侵 2 中断 1: 使能入侵 2 中断
18	TP1MF	入侵 1 掩码标志。 0: 不屏蔽入侵 1 事件。 1: 屏蔽入侵 1 事件。 注意: 当 TP1MF 置位时, 不得使能 Tamper 1 中断。
17	TP1NOE	入侵 1 不擦除位。 0: 入侵 1 事件擦除备份寄存器 1: 入侵 1 事件不擦除备份寄存器

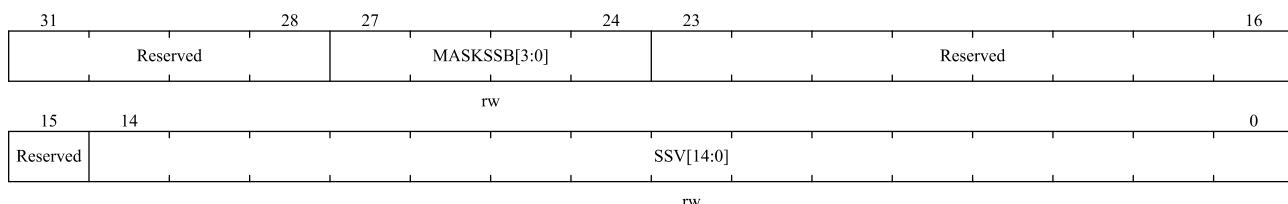
位域	名称	描述
16	TP1INTEN	入侵 1 中断使能位。 0: TPINTEN = 0 时禁止入侵 1 中断 1: 使能入侵 1 中断
15	TPPUDIS	RTC_TAMPx 上拉禁用位。 0: 每次采样前启用预充电 RTC_TAMPx 引脚。 1: 禁用预充电 RTC_TAMPx 引脚
14:13	TPPRCH[1:0]	RTC_TAMPx 预充电持续时间。 这些位确定每次采样前的预充电时间。 0x0: 1 个 RTCCLK 周期 0x1: 2 个 RTCCLK 周期 0x2: 4 个 RTCCLK 周期 0x3: 8 个 RTCCLK 周期
12:11	TPFLT[1:0]	RTC_TAMPx 过滤器计数。 这些位决定在有效电平时的连续采样次数。 0x0: 在有效电平上 1 次采样后触发入侵事件 0x1: 在有效电平上连续 2 次采样后触发入侵事件 0x2: 在有效电平上连续 4 次采样后触发入侵事件 0x3: 在有效电平上连续 8 次采样后触发入侵事件
10:8	TPFREQ[2:0]	入侵采样频率。 该位决定对每个 RTC_TAMPx 输入进行采样时的频率。 0x0: 每 32768 个 RTCCLK 采样一次 (当 RTCCLK = 32.768 KHz 时为 1 Hz) 0x1: 每 16384 个 RTCCLK 采样一次 0x2: 每 8192 个 RTCCLK 采样一次 0x3: 每 4096 个 RTCCLK 采样一次 0x4: 每 2048 个 RTCCLK 采样一次 0x5: 每 1024 个 RTCCLK 采样一次 0x6: 每 512 个 RTCCLK 采样一次 0x7: 每 256 个 RTCCLK 采样一次
7	TPTS	发生入侵检测事件时激活时间戳位。 0: 发生入侵检测事件时不保存时间戳 1: 发生入侵检测事件时保存时间戳 即便 RTC_CTRL.TSEN=0, TPTS 仍有效。
6	TP3TRG	入侵 3 事件触发模式。 如果 TPFLT[1:0] != 00, 入侵检测处于电平模式： 0: 低电平触发入侵检测事件。 1: 高电平触发入侵检测事件。 如果 TPFLT[1:0] = 00, 入侵检测处于边沿模式： 0: 上升沿触发入侵检测事件。 1: 下降沿触发入侵检测事件
5	TP3EN	RTC_TAMP3 检测使能位。 0: 禁止 RTC_TAMP3 输入检测 1: 开启 RTC_TAMP3 输入检测

位域	名称	描述
4	TP2TRG	入侵 2 事件触发模式。 如果 TPFLT[1:0] != 00, 入侵检测处于电平模式: 0: 低电平触发入侵检测事件。 1: 高电平触发入侵检测事件。 如果 TPFLT[1:0] = 00, 入侵检测处于边沿模式: 0: 上升沿触发入侵检测事件。 1: 下降沿触发入侵检测事件
3	TP2EN	RTC_TAMP2 检测使能位。 0: 禁止 RTC_TAMP2 输入检测 1: 开启 RTC_TAMP2 输入检测
2	TPINTEN	入侵事件中断使能。 0: 禁止入侵中断 1: 使能入侵中断 注: 该位使能所有入侵引脚事件的中断, 与 TPxINTEN 电平无关。如果该位清零, 每个入侵事件中断可以通过设置 TPxINTEN 单独启用。
1	TP1TRG	入侵 1 事件触发模式。 如果 TPFLT[1:0] != 00, 入侵检测处于电平模式: 0: 低电平触发入侵检测事件。 1: 高电平触发入侵检测事件。 如果 TPFLT[1:0] = 00, 入侵检测处于边沿模式: 0: 上升沿触发入侵检测事件。 1: 下降沿触发入侵检测事件
0	TP1EN	RTC_TAMP1 检测使能位。 0: 禁止 RTC_TAMP1 输入检测 1: 开启 RTC_TAMP1 输入检测

### 20.3.18 RTC 闹钟 A 亚秒寄存器(RTC\_ALRMASS)

偏移地址: 0x44

复位值: 0x0000 0000



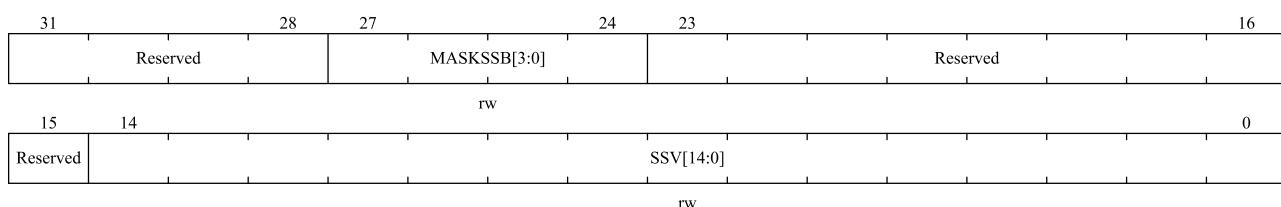
位域	名称	描述
31:28	Reserved	保留, 必须保持复位值。
27:24	MASKSSB[3:0]	屏蔽此位开始的最高有效位。 0x0: 闹钟的亚秒不比较。当秒单位增加时设置闹钟 (假设其余字段匹配)。 0x1: 只比较 SS[0], 不比较其他位。

位域	名称	描述
		0x2: 仅比较 SS[1:0], 不比较其他位。 0x3: 仅比较 SS[2:0], 不比较其他位。 ... 0xC: 仅比较 SS[11:0], 不比较其他位。 0xD: 仅比较 SS[12:0], 不比较其他位。 0xE: 仅比较 SS[13:0], 不比较其他位。 0xF: 比较 SS[14:0] 从不比较同步计数器 RTC_SUBS.SS[15]位。
23:15	Reserved	保留, 必须保持复位值。
14:0	SSV[14:0]	亚秒值。 该值与同步预分频计数器 RTC_SUBS.SS[14:0]进行比较, 比较的位数由 MASKSSB[3:0]控制。

### 20.3.19 RTC 闹钟 B 亚秒寄存器 (RTC\_ALRMBSS)

偏移地址: 0x48

复位值: 0x0000 0000

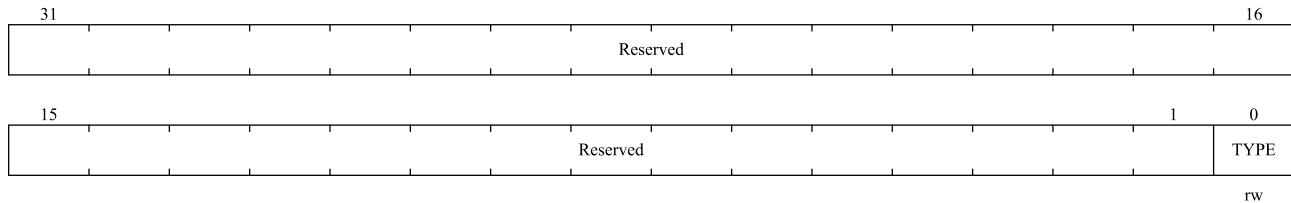


位域	名称	描述
31:28	Reserved	保留, 必须保持复位值。
27:24	MASKSSB[3:0]	屏蔽此位开始的最高有效位。 0x0: 闹钟的亚秒不比较。当秒单位增加时设置闹钟 (假设其余字段匹配)。 0x1: 只比较 SS[0], 不比较其他位。 0x2: 仅比较 SS[1:0], 不比较其他位。 0x3: 仅比较 SS[2:0], 不比较其他位。 ... 0xC: 仅比较 SS[11:0], 不比较其他位。 0xD: 仅比较 SS[12:0], 不比较其他位。 0xE: 仅比较 SS[13:0], 不比较其他位。 0xF: 比较 SS[14:0]。 从不比较同步计数器 RTC_SUBS.SS[15]位。
23:15	Reserved	保留, 必须保持复位值。
14:0	SSV[14:0]	亚秒值 该值与同步预分频计数器 RTC_SUBS.SS[14:0]进行比较, 比较的位数由 MASKSSB[3:0] 控制。

### 20.3.20 RTC 选项寄存器 (RTC\_OPT)

偏移地址: 0x4C

复位值: 0x0000 0000

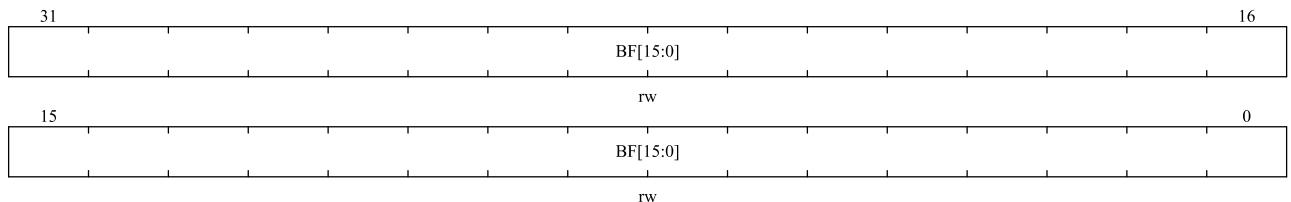


位域	名称	描述
31:1	Reserved	保留, 必须保持复位值。
0	TYPE	PC13 上的 RTC_ALARM 输出类型位。 0: 开漏输出 1: 推挽输出

### 20.3.21 RTC 备份寄存器 (RTC\_BKP(1~20))

偏移地址: 0x50 ~ 0x9C

复位值: 0x0000 0000



位域	名称	描述
31:0	BF[31:0]	备份数据 这些寄存器可以通过软件进行读写。 这些寄存器在 MR 关闭时由 BKR 供电, 因此当系统复位时, 这些寄存器不会复位, 并且在器件工作在低功耗模式时寄存器的内容仍然有效。 如果 RTC_TMPCFG.TPxNOE=0, 当检测到入侵事件时这些寄存器被复位。

## 21 蜂鸣器

### 21.1 介绍

蜂鸣器模块支持互补输出并且可以产生周期性信号来驱动外部无源蜂鸣器，用于产生提示音或警报声。

### 21.2 功能描述

作为独立模块，蜂鸣器挂载在 APB2 总线上，最高工作频率为 64MHz，两个输出之一通常是关闭的。如果启用互补输出，则两路输出同时开启并输出互补。

#### 21.2.1 主要特点

在正常操作模式下，蜂鸣器可以从三个主时钟源中选择：

1. Beeper 支持 (64MHz–1MHz) APB 时钟、40KHz LSI 时钟和 32.768KHz LSE 时钟的时钟范围。
2. 蜂鸣器输出可调节音调，具体取决于寄存器配置 (32 Hz 至 64MHz)。

表 21-1 蜂鸣器支持的最大和最小频率以及相应的配置。

	频率	时钟源	预分频系数	蜂鸣器分频	旁路使能
最大	64 MHz	APB2 (64MHz)	0	/	1
最小	32 Hz	LSE (32.768K)	/	1024	0

3. 鸣器也可以通过设置 BEEPER\_CTRL.BYPASSEN 位将时钟信号旁路到输出端口。

蜂鸣器 2 输出是蜂鸣器 1 输出的互补输出，设置寄存器位将启用该输出端口。对于两个蜂鸣器，它只需要配置一次。

#### 21.2.2 设定蜂鸣器频率预分频因子

蜂鸣器设计支持 1 至 64 MHz 之间的 APB2 时钟工作频率，为了防止失真，BEEPER\_CTRL.PSC 位需要写入正确的数值。

设计预分频比的目的用于将 APB2 时钟按比例缩小成低至 1 MHz 脉冲信号，用于生成 64 MHz ~ 976.5625 Hz 的信号。但是，为了达到准确的 8 KHz 或 4 KHz，我们可以将 APB2 时钟设置为 64 MHz，分频比为 16，以获得 4 MHz 信号输出。然后通过将 BEEPER\_CTRL.BEEP DIV 设置为 250 我们可以获得 8 KHz 信号，设置 BEEPER\_CTRL.BEPPDIV = 500，我们可以获得 4 KHz 信号。

#### 21.2.3 旁路时钟

通过将 BEEPER\_CTRL.BYPASSEN 位置 1，可以将第一阶段的输出时钟可以被旁路到其他端口，这就意味着 PCLK/LSI/LSE 可以直接传递到其他端口。然而，对于 PCLK 在预分频后，占空比将不会是 50%。蜂鸣器支持的最大和最小频率如表 21-1 所示。

设置这些寄存器值后，蜂鸣器经过两级分频后可以输出相应的频率，使得蜂鸣器可以输出高达 64 MHz 和

低至 32 Hz 的频率。

如果用户想要蜂鸣器 2 进行互补输出，只需要将 BEEPER\_CTRL.INVEN[2]位置 1 即可。

## 21.2.4 输出占空比

分频器有两级，第一级支持奇分频。此阶段的输出的占空比波形是没有 50% 的，这种状况下仅适用于 PCLK。对于 LSE、LSI 时钟源，它们可以输出 50% 的占空比。这意味着在旁路使能的情况下，使用 LSE 或 LSI 时钟源可以输出 50% 占空比的波形。在没有旁路的情况下，蜂鸣器（蜂鸣器 1 或 2）也可以输出占空比为 50% 的波形。

## 21.3 蜂鸣器寄存器

### 21.3.1 BEEPER 寄存器总览

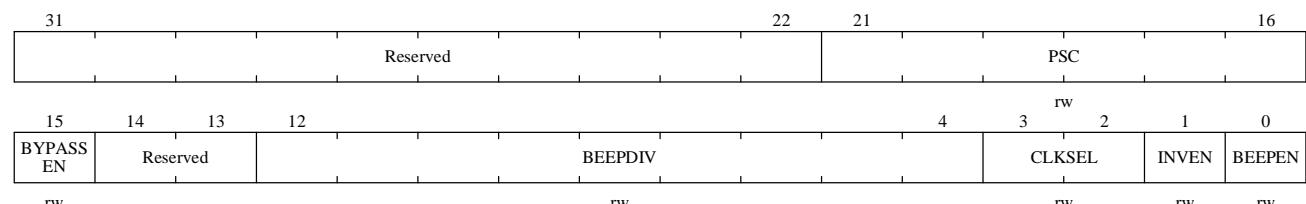
表 21-2 BEEPER 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
000h	BEEPER_CTRL																																	
	Reset Value																																	

### 21.3.2 蜂鸣器控制寄存器(BEEPER\_CTRL)

偏移地址: 0x00

默认值: 0x0000 0000



位域	名称	描述
31:22	Reserved	保留，必须保持复位值。
21:16	PSC	APB 时钟预分频因数。 00_0000: 无分频 00_0001: 分频系数为 2 ..... ..... 11_1111: 分频系数为 64
15	BYPASSEN	旁通功能使能 在第二阶段选择时钟信号旁路 1: 旁通使能 0: 旁通失能

位域	名称	描述
14:13	Reserved	保留，必须保持复位值。
12:4	BEEPDIV	蜂鸣器分频系数 0_0000_0000: 输出频率值除以 2 ((BEEPDIV+1)*2) 0_0000_0001: 输出频率值除以 4 0_0000_0010: 输出频率值除以 6 ..... ..... 1_1111_1111: 输出频率值除以 1024
3:2	CLKSEL	时钟源选择 0x : 使用预分频 APB 时钟作为下一阶段时钟 10 : 使用 LSI 时钟作为下一级时钟源 11 : 使用 LSE 时钟作为下一级时钟源
1	INVEN	蜂鸣器互补输出使能 蜂鸣器 1 和蜂鸣器 2 是互补的输出信号。 0: 只有一个输出，另一个输出关闭。 1: 两个输出都打开，并且输出是互补的
0	BEEPEN	蜂鸣器使能 1: 蜂鸣器使能 0: 蜂鸣器失能

## 22 控制器局域网(CAN)

### 22.1 CAN 简介

作为 CAN 网络接口，基本扩展 CAN 支持 CAN 协议 2.0A 和 2.0B。它可以高效地处理大量接收到的消息，大大降低 CPU 资源的消耗。报文发送的优先级特性可以通过软件进行配置，CAN 的硬件功能可以支持时间触发的通信方式，适用于一些对安全性要求较高的应用。

### 22.2 CAN 的主要特性

- 波特率最高支持 1Mbit/s
- 支持 CAN 协议 2.0A/B.
- 支持时间触发通讯功能
- 支持独立的中断控制。
- CAN Core 管理 CAN 和 512 字节 SRAM 存储器之间的通信（参见图 22-3）

#### 时间触发通信模式

- 16 位自由运行定时器
- 禁止自动重传模式。
- 可配置最后 2 个数据字节为发送时间戳。

#### 发送

- 有三个发送邮箱。
- 记录发送 SOF 时间的时间戳功能
- 软件可以配置发送消息的优先级特性。

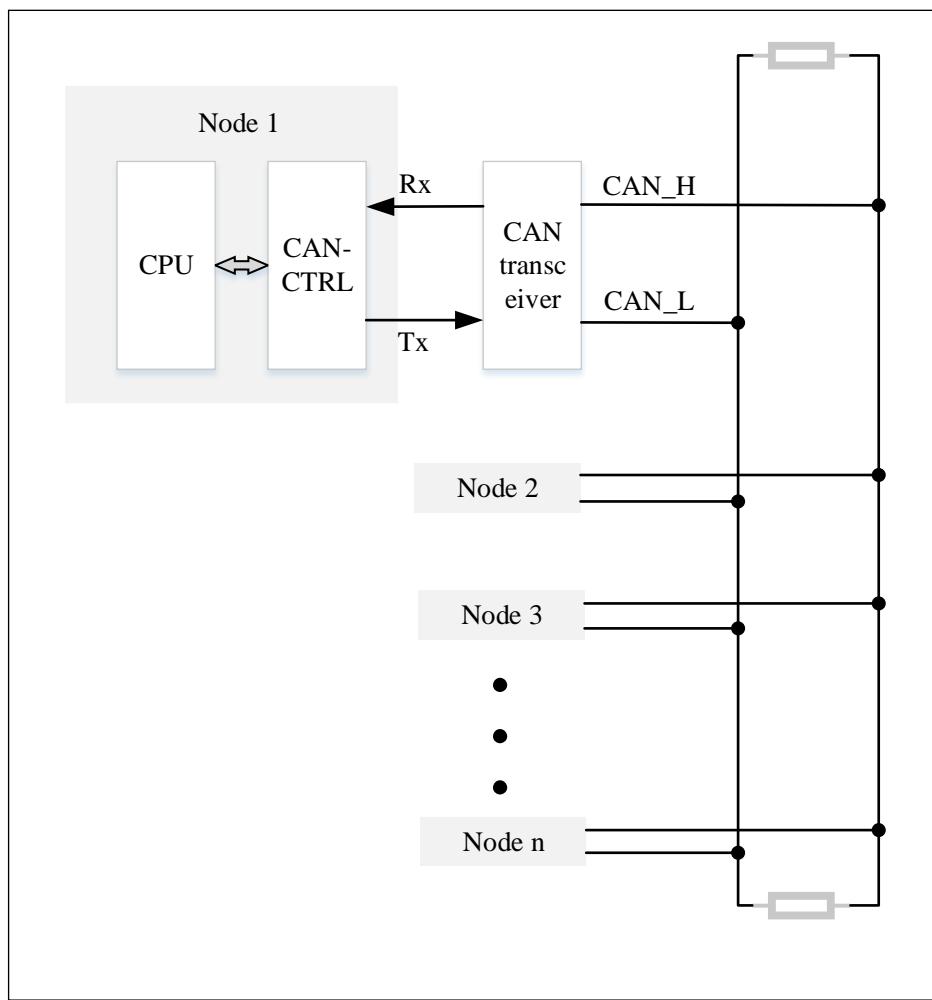
#### 接收

- 过滤器组支持标识符列表模式
- 有两个接收 FIFO，每个深度为 3 级
- 共有 14 个过滤器组
- 可配置的 FIFO 溢出处理方法
- 记录接收 SOF 时间的时间戳功能

### 22.3 CAN 整体介绍

随着 CAN 的广泛应用，CAN 网络的节点数量迅速增长，多个 CAN 节点通过 CAN 网络连接。随着 CAN 节点数量的增加，CAN 网络中的报文也急剧增加，会占用大量的 CPU 资源。在这个 CAN 控制器中，增加了接收 FIFO 和过滤机制作为 CPU 消息处理的硬件支持，降低了 CAN 消息的实时响应要求。

图 22-1 CAN 网络拓扑



### 22.3.1 CAN 模块

CAN 模块可自动收发 CAN 报文，支持标准标识符（11 位）和扩展标识符（29 位）。

### 22.3.2 CAN 工作模式

初始化、正常和睡眠模式是 CAN 的三种主要工作模式。CANTX 引脚的内部上拉电阻在硬件复位后被激活，CAN 工作在睡眠模式以降低功耗。

软件可以设置 CAN\_MCTRL.INIRQ 和 CAN\_MCTRL.SLPRQ 位来配置 CAN 进入初始化或睡眠模式。软件读 CAN\_MSTS.INIAK 或 CAN\_MSTS.SLPAK 位的值来确认是否进入初始化或睡眠模式，此时 CANTX 引脚的内部上拉电阻被禁用。

当 CAN\_MSTS.INIAK 和 CAN\_MSTS.SLPAK 位都为“0”时，CAN 处于正常模式，它必须与 CAN 总线同步才能进入正常模式。当在 CANRX 引脚上监测到 11 个连续的隐性位时，CAN 总线处于空闲状态，同步完成。

#### 22.3.2.1 正常模式

初始化完成后，软件配置 CAN 进入正常模式。清除 CAN\_MCTRL.INIRQ 位并等待硬件清除 CAN\_MSTS.INIRQ 位以确认 CAN 进入正常模式。只有与 CAN 总线同步完成后，CAN 才能正常收发报文。

过滤器组的位宽和模式的设置必须在过滤器处于初始化模式（CAN\_FMC.FINITM 位为 1）时完成。过滤器初始值的设置必须在未激活时完成（对应的 CAN\_FA1.FAC 位为 0）。

### 22.3.2.2 初始化模式

软件只有在 CAN 处于初始化模式时才能进行初始化配置。设置 CAN\_MCTRL.INIRQ 位并清除 CAN\_MCTRL.SLPRQ 位，并等待硬件设置 CAN\_MSTS.INIAK 位以确认 CAN 进入初始化模式。进入初始化模式时，寄存器配置不受影响。CAN 处于初始化模式时，报文收发被禁止，CANTX 引脚输出一个隐性位（高电平）。要退出初始化模式，清除 CAN\_MCTRL.INIRQ 位，并等待硬件清除 CAN\_MSTS.INIAK 位以确认 CAN 退出初始化模式。

通过软件对 CAN 进行初始化配置，至少需要配置位时间特性寄存器（CAN\_BTIM）和控制寄存器（CAN\_MCTRL）。软件需要设置 CAN\_FMC.FINITM 位来初始化配置 CAN 的过滤器组（模式、位宽、FIFO 关联、激活和过滤器值）。配置 CAN 的过滤器组不一定需要处于初始化模式。

需要注意的是当 CAN\_FMC.FINITM=1 时，禁止接收报文。如果要修改对应过滤器的值，需要先清除过滤器激活位（在 CAN\_FA1 中）。有必要保持未使用的过滤器组处于非活动状态（将其 CAN\_FA1.FAC 位保持为“0”）。

### 22.3.2.3 睡眠模式（低功耗）

要进入睡眠模式，需要设置 CAN\_MCTRL.SLPRQ 位，并等待硬件设置 CAN\_MSTS.SLPAK 位以确认 CAN 进入睡眠模式。在未使用时 CAN 可以配置为睡眠模式以降低功耗。在睡眠模式下，CAN 的时钟停止工作，但软件仍然可以访问发送/接收邮箱寄存器。当 CAN 处于睡眠模式时，CAN\_MCTRL.INIRQ 位必须置 1，并且 CAN\_MCTRL.SLPRQ 位必须同时清零才能进入初始化模式。

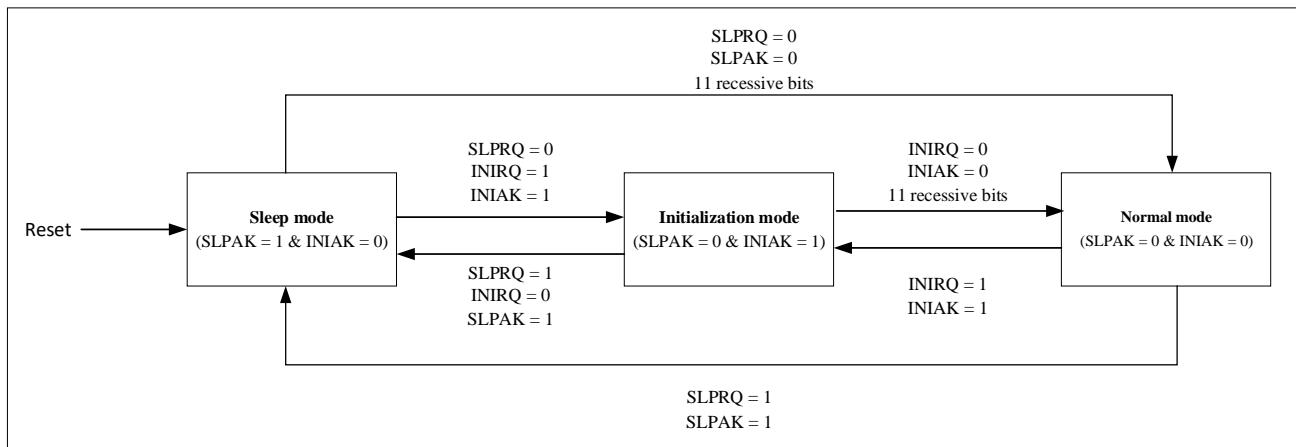
唤醒 CAN 有两种方式（CAN 退出睡眠模式）：

- 当 CAN\_MCTRL.AWKUM 位被置位（使能硬件自动唤醒）时，一旦检测到 CAN 总线的活动，硬件将自动清除 CAN\_MSTS.SLPRQ 位以唤醒 CAN。
- 当 CAN\_MCTRL.AWKUM 位清零（使能软件唤醒），并且唤醒中断发生时，软件必须清零 CAN\_MCTRL.SLPRQ 位才能退出睡眠状态。

如果唤醒中断（设置 CAN\_INTE.WKUITE 位）使能，一旦检测到 CAN 总线活动，将产生唤醒中断，无论硬件是否使能自动唤醒 CAN。

在进入正常模式之前，CAN 必须与 CAN 总线同步等到 CAN\_MSTS.SLPAK 位清零以确认已退出睡眠模式。请参考图 22-2。

图 22-2 CAN 工作模式



### 22.3.3 发送邮箱

应用程序可以通过三个发送邮箱发送消息，发送三个邮箱消息的顺序是由发送调度器根据消息的优先级决定的，优先级可以由消息的标识符决定，也可以由发送请求的顺序决定。

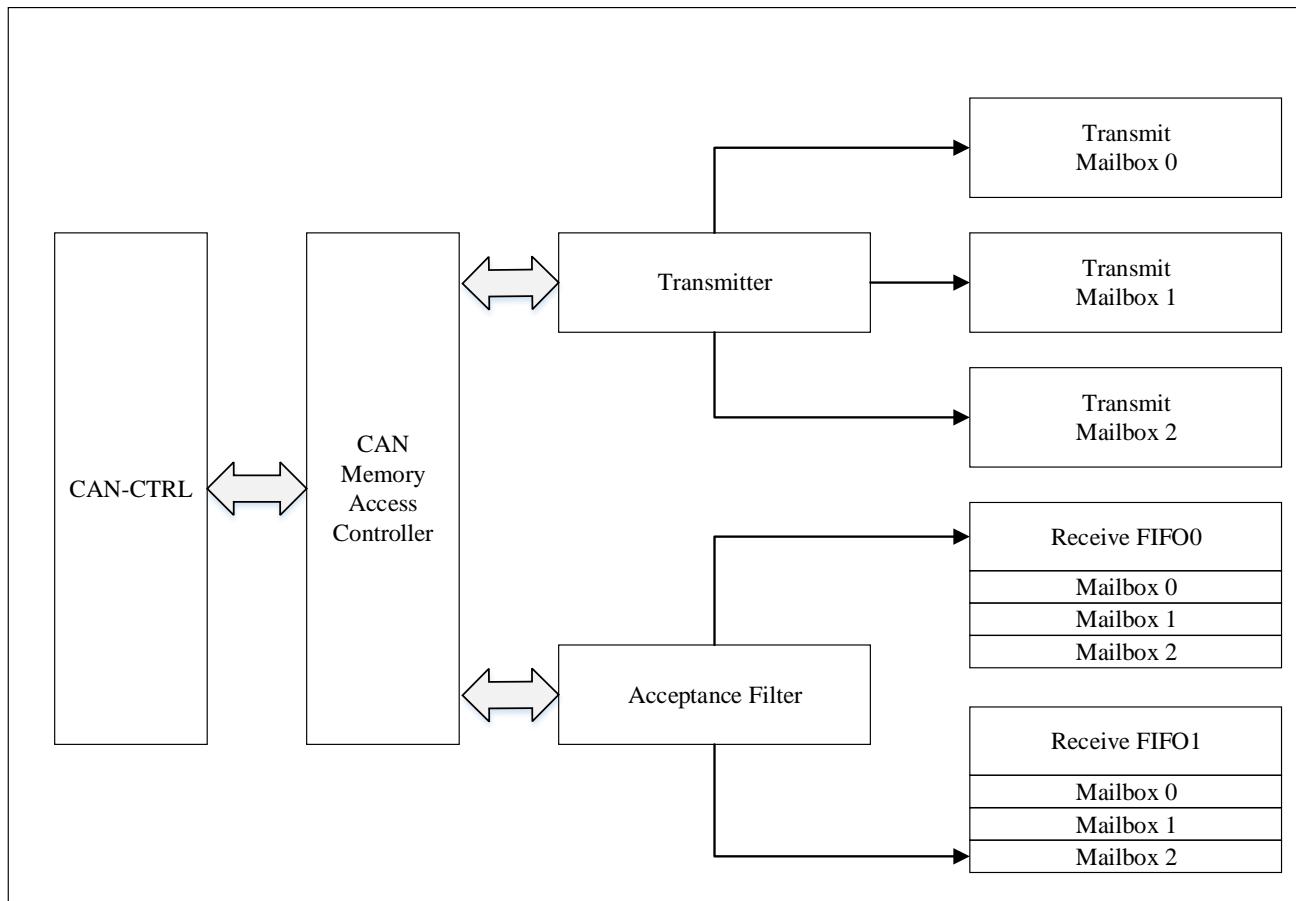
### 22.3.4 接收过滤器

CAN 有 14 个可配置的标识符过滤器组。应用配置标识符过滤器组后，接收邮箱会自动接收需要的邮件，丢弃其他邮件。

### 22.3.5 接收 FIFO

CAN 有两个接收 FIFO，每个 FIFO 可以存储三个完整的报文。无需应用程序管理，由硬件管理。

图 22-3 单 CAN 框图



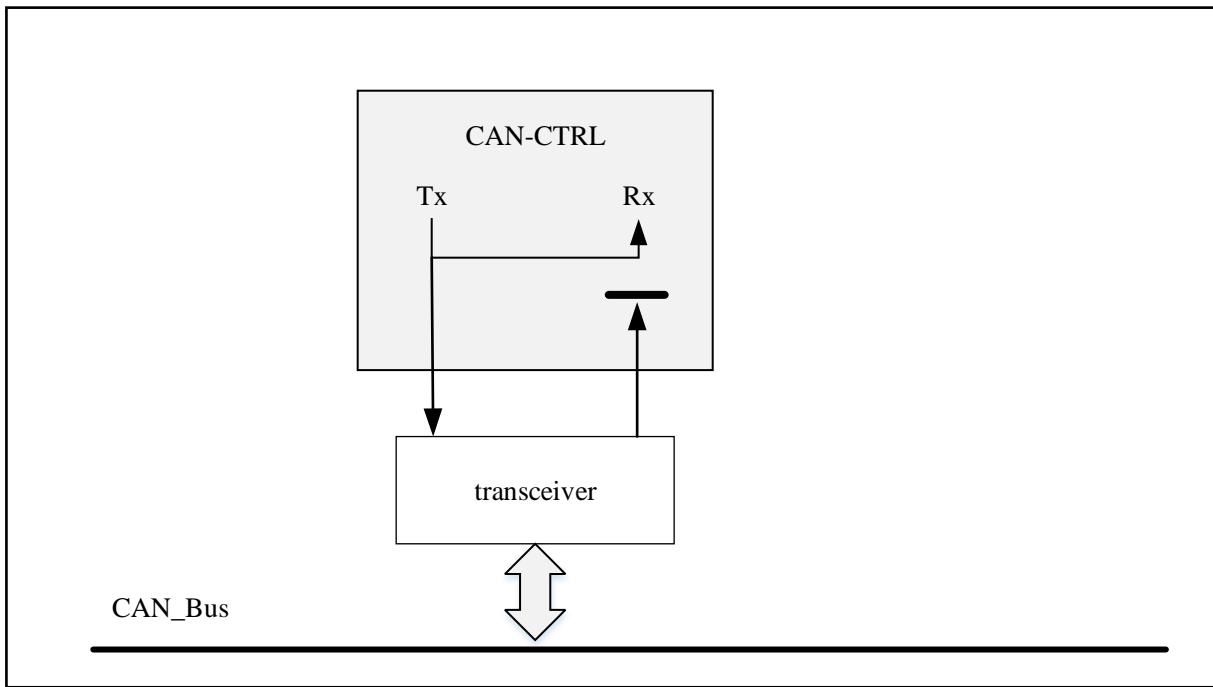
### 22.3.6 CAN 测试模式

在初始化模式下，必须通过组合 CAN\_BTIM.SLM 位和 CAN\_BTIM.LBM 位来选择测试模式。选择测试模式后，软件需要清除 CAN\_MCTRL.INIRQ 位退出初始化模式，进入测试模式。

#### 22.3.6.1 回环模式

回环模式可用于自检。在回环模式下，CAN 将发送的消息作为接收消息保存在接收邮箱中（如果可以通过接收过滤）。在回环模式下，CAN 在内部将 Tx 输出反馈到 Rx 输入，完全忽略 CANRX 引脚的实际状态。可以在 CANTX 引脚上检测到发送的消息。为了避免外部影响，CAN 内核忽略了确认错误（在数据/远程帧确认位的时刻，不检测是否有显性位）。要进入回环模式，应清除 CAN\_BTIM.SLM 位并设置 CAN\_BTIM.LBM 位。

图 22-4 回环模式

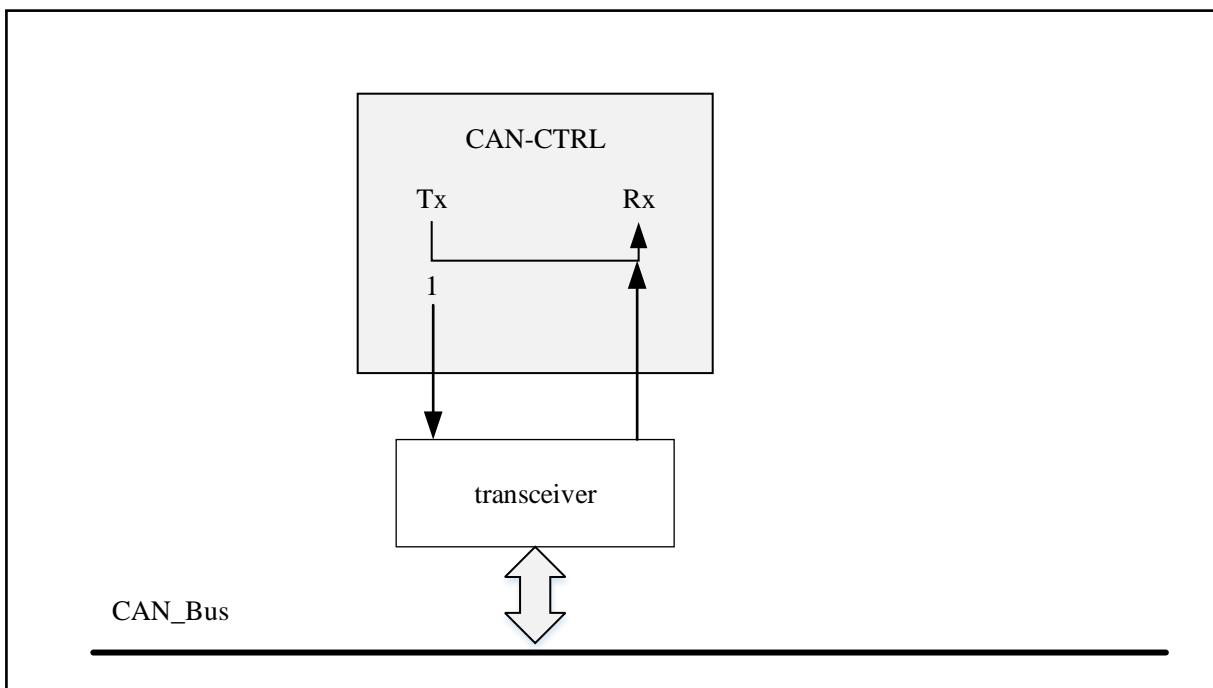


### 22.3.6.2 静默模式

在静默模式下，CAN 可以正常接收数据帧和远程帧，但只能发送隐性位，不能真正发送消息。如果 CAN 需要发送过载标志、主动错误标志或 ACK 位（这些是显性位），这些显性位在内部连接回来以便被 CAN 内核检测到。同时，CAN 总线不会受到影响，仍然保持在隐性位状态。因此，静默模式通常用于分析 CAN 总线的活动，而不影响总线，因为显性位实际上并未发送到总线。

要进入静默模式，应设置 CAN\_BTIM.SLM 位并清除 CAN\_BTIM.LBM 位。

图 22-5 静默模式

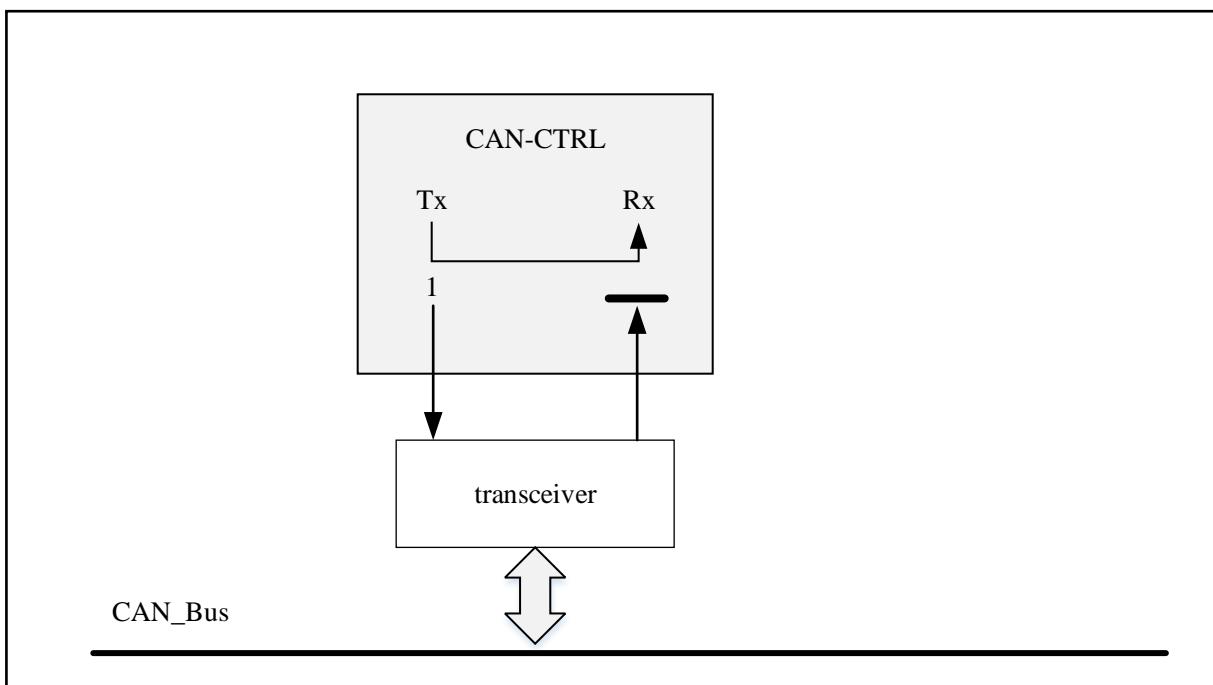


### 22.3.6.3 回环静默模式

在回环静默模式下，CANRX 引脚与 CAN 总线断开，而 CANTX 引脚被驱动为隐性位状态。它可用于“热自检”，就像可以在回环模式下测试 CAN 一样，但不影响 CANTX 和 CANRX 连接的整个 CAN 系统。

要进入回环静默模式，应设置 CAN\_BTIM.SLM 位和 CAN\_BTIM.LBM 位。

图 22-6 回环静默模式



## 22.3.7 CAN 调试模式

CAN 可以根据以下配置位的状态继续正常工作或停止工作:

- 调试支持(DBG)模块中 CAN 的 DBG\_CTRL.CAN\_STOP 位。参见第 23.3.2 节：外设调试支持。
- CAN\_MCTRL.DBGF 位见 22.7.3.1 节:CAN\_MCTRL.

当微控制器处于调试模式时，Cortex-M4F 内核处于挂起状态.

## 22.4 CAN 功能描述

### 22.4.1 发送处理

发送消息的流程如下:

- 应用程序选择一个空的发送邮箱;
- 将标识符、数据长度和要发送的数据写入发送邮箱寄存器;
- 设置 CAN\_TMIx.TXRQ 位请求发送 (设置 CAN\_TMIx.TXRQ 后邮箱不再是空邮箱，软件无权写入邮箱寄存器);
- 邮箱进入 pending 状态，等待成为最高优先级，见 22.4.1.1 发送优先级;
- 邮箱成为最高优先级邮箱后，转为就绪状态;
- CAN 总线一进入空闲状态就发送就绪邮箱中的消息，然后进入发送状态。
- 邮箱中的消息发送成功后变为空邮箱;
- 硬件设置 CAN\_TSTS 寄存器中对应邮箱的 RQCPM 和 TXOKM 位，表示发送成功。

需要注意的是，如果发送失败，CAN\_TSTS.ALSTM 位被设置表示失败是由仲裁引起的，或者 CAN\_TSTS.TERRM 位被设置表示是传输错误引起的（具体错误请查看错误状态寄存器 CAN\_ESTS 的 LEC[2:0]错误代码位）.

#### 22.4.1.1 发送优先级

由发送请求的顺序决定:

设置 CAN\_MCTRL.TXFP 位，发送邮箱可以配置为发送 FIFO。此时，发送的优先级由发送请求的顺序决定。这种模式对于分段传输很有用。

由标识符决定:

根据 CAN 协议，具有最低标识符的消息具有最高优先级。如果标识符的值相等，则先发送邮箱号小的消息。当注册多个发送邮箱时，发送顺序由邮箱中邮件的标识符决定。

#### 22.4.1.2 取消发送

设置 CAN\_TSTS.ABRQM 位可以中止发送请求。如果邮箱就绪或挂起，发送请求将立即中止。如果邮箱处于发送状态，请求中止可能会导致两种结果：

- 如果邮箱中的消息发送失败，邮箱变为就绪状态，则发送请求中止，邮箱变为空邮箱并清除

CAN\_TSTS.TXOKM 位；

- 如果邮箱中的消息发送成功，邮箱变为空邮箱，CAN\_TSTS.TXOKM 位将由硬件设置为 1。

因此，当发送邮箱处于发送状态时，无论发送结果如何，发送操作完成后邮箱都会变成空邮箱。

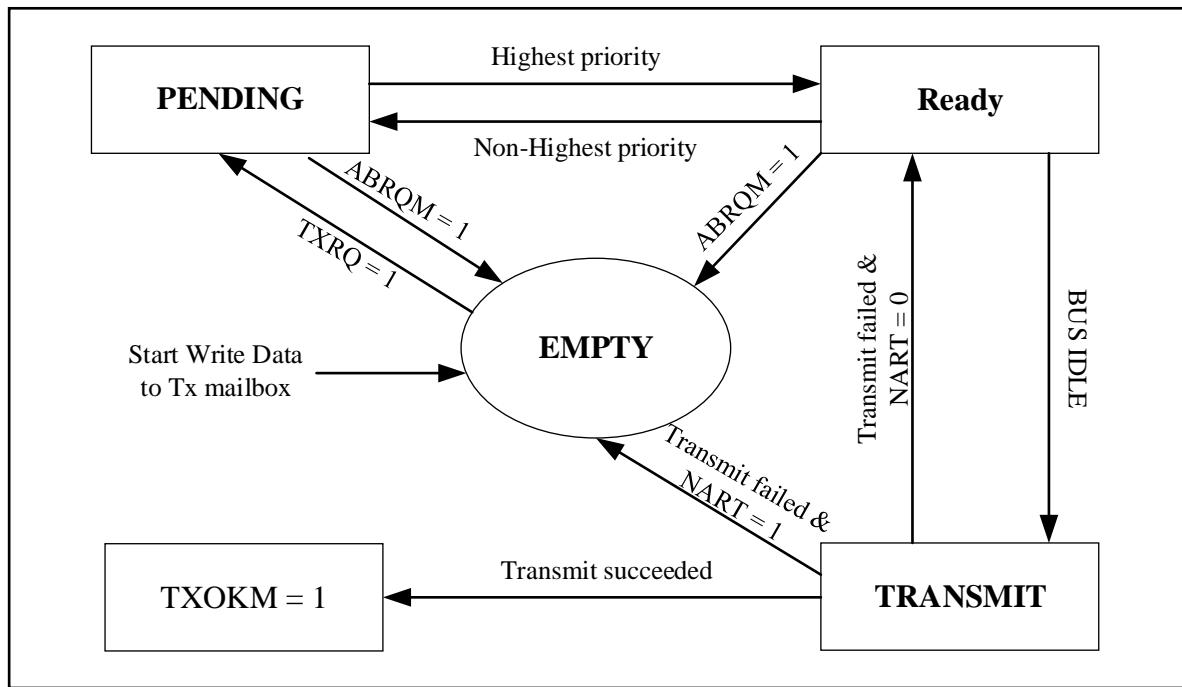
## 22.4.2 时间触发通信模式

CAN 的内部定时器在时间触发通信模式下被激活。它在每个 CAN 位时间递增（参见第 22.4.7 节）。CAN 在收发帧起始位的采样点位置对内部定时器的值进行采样，采样值即为收发邮箱的时间戳。内部定时器生成的时间戳将分别存储在 CAN\_RMDTx/CAN\_TMDTx 寄存器中。

## 22.4.3 非自动重传模式

要启用非自动重传模式，应设置 CAN\_MCTRL.NART 位。该模式对应 CAN 标准中时间触发通信选项的功能。在非自动重传模式下，发送操作只会执行一次。如果发送操作失败，无论是仲裁丢失还是错误，硬件都不会自动再次发送报文。发送操作结束时，硬件判定发送请求已经完成，硬件设置 CAN\_TSTS.RQCPM 位。同时，传输结果可以查询 CAN\_TSTS.TXOKM、CAN\_TSTS.ALSTM 和 CAN\_TSTS.TERRM 位。

图 22-7 发送邮箱状态



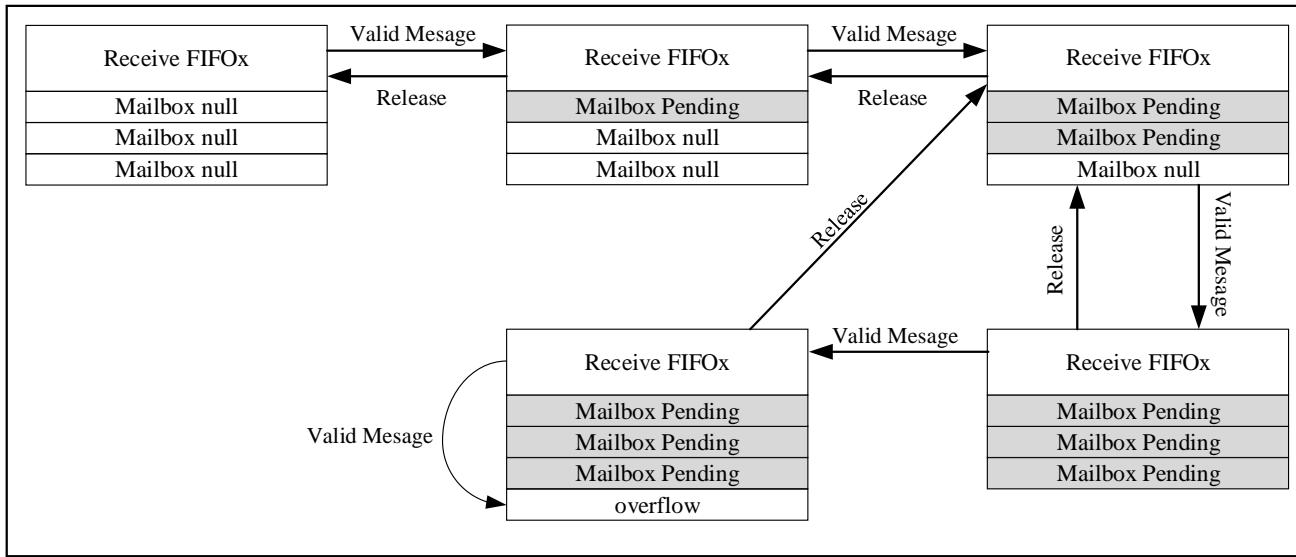
## 22.4.4 接收管理

具有 3 级深度的 FIFO 用于存储接收到的消息。当应用程序读取 FIFO 输出邮箱时，它会读取 FIFO 中第一个接收到的消息。FIFO 完全由硬件管理，可以简化应用程序，保证数据的一致性，减少 CPU 的处理时间。

### 22.4.4.1 有效报文

根据 CAN 协议，当消息被正确接收（直到 EOF 字段的最后一位没有错误发送）并通过标识符过滤时，则该消息被标记为有效消息。请参考 22.4.5 章节：标识符过滤。

图 22-8 接收邮箱状态



#### 22.4.4.2 FIFO 接收

一个 FIFO 包含三级深度的邮箱，初始状态为空。接收到第一个有效消息后，其中一个邮箱将被挂起，硬件会将 CAN\_RFF.FFMP[1:0]位设置为 1，表示收到有效消息。在释放邮箱之前再次收到有效消息。这时候会同时挂起两个邮箱，硬件会将 CAN\_RFF.FFMP[1:0]位设置为 2，表示有两个有效的报文处于挂起状态。如上所述，第三条有效消息将暂停所有三个邮箱并将 CAN\_RFF.FFMP[1:0]位设置为 3。

当 FIFO 的三级邮箱全部挂起时，再次接收到有效报文会导致邮箱溢出丢失报文，硬件会将 CAN\_RFF.FFOVR 位设置为 1，表示事件发生。丢失消息的规则取决于 FIFO 的配置。如果禁用 FIFO 锁定功能（清除 CAN\_MCTRL.RFLM 位），则 FIFO 中接收的最后一条消息将被新消息覆盖。这样最新接收到的报文不会被丢弃；如果启用 FIFO 锁定功能（设置 CAN\_MCTRL.RFLM 位），那么新接收到的报文将被丢弃，软件可以读取 FIFO 中的前三个报文。

#### 22.4.4.3 FIFO 释放

存储在 FIFO 中的消息将通过相应的接收邮箱读取。软件读取邮箱报文并通过设置 CAN\_RFR.RFOM 位为 1 释放邮箱，CAN\_RFF.FFMP[1:0]位减 1 直到为 0。

#### 22.4.4.4 接收相关中断

当消息存储在接收 FIFO 中时，硬件将更新 CAN\_RFF.FFMP[1:0]位。如果当前使能了 FIFO 报文挂起中断（CAN\_INTE.FMPITE 位置位），则将产生挂起中断请求。

当存储第三条消息时，FIFO 变满，然后 CAN\_RFF.FFULL 位将被置位，如果当前使能 FIFO 满中断（CAN\_INTE.FFITE 位被置位），将产生一个 FIFO 满中断请求。

当 FIFO 溢出时，FFOVR 位将被设置。如果当前使能 FIFO 溢出中断（CAN\_INTE.FOVITE 位置位），将产生 FIFO 溢出中断请求。

#### 22.4.5 标识符过滤

在 CAN 网络中，当 CAN 处于发送器状态时，它通过向总线发送消息的方式向各个节点广播消息；当 CAN 处于接收者状态时，节点收到报文后根据报文的标识来判断是否需要该报文。如果报文有效，CAN 内核将

报文复制到 SRAM (CAN 自身的 SRAM); 如果不需要, 该消息将被丢弃。此过程不需要软件干预。与软件过滤相比, 硬件过滤降低了 CPU 使用率。CAN 控制器为应用程序提供了 14 个位宽可变的可配置过滤器组 (0~13) 来满足这一需求, 这些过滤器组用于接收仅软件所需的那些消息。每个过滤器组 x 包含两个 32 位寄存器, 即 CAN\_FxR0 和 CAN\_FxR1。

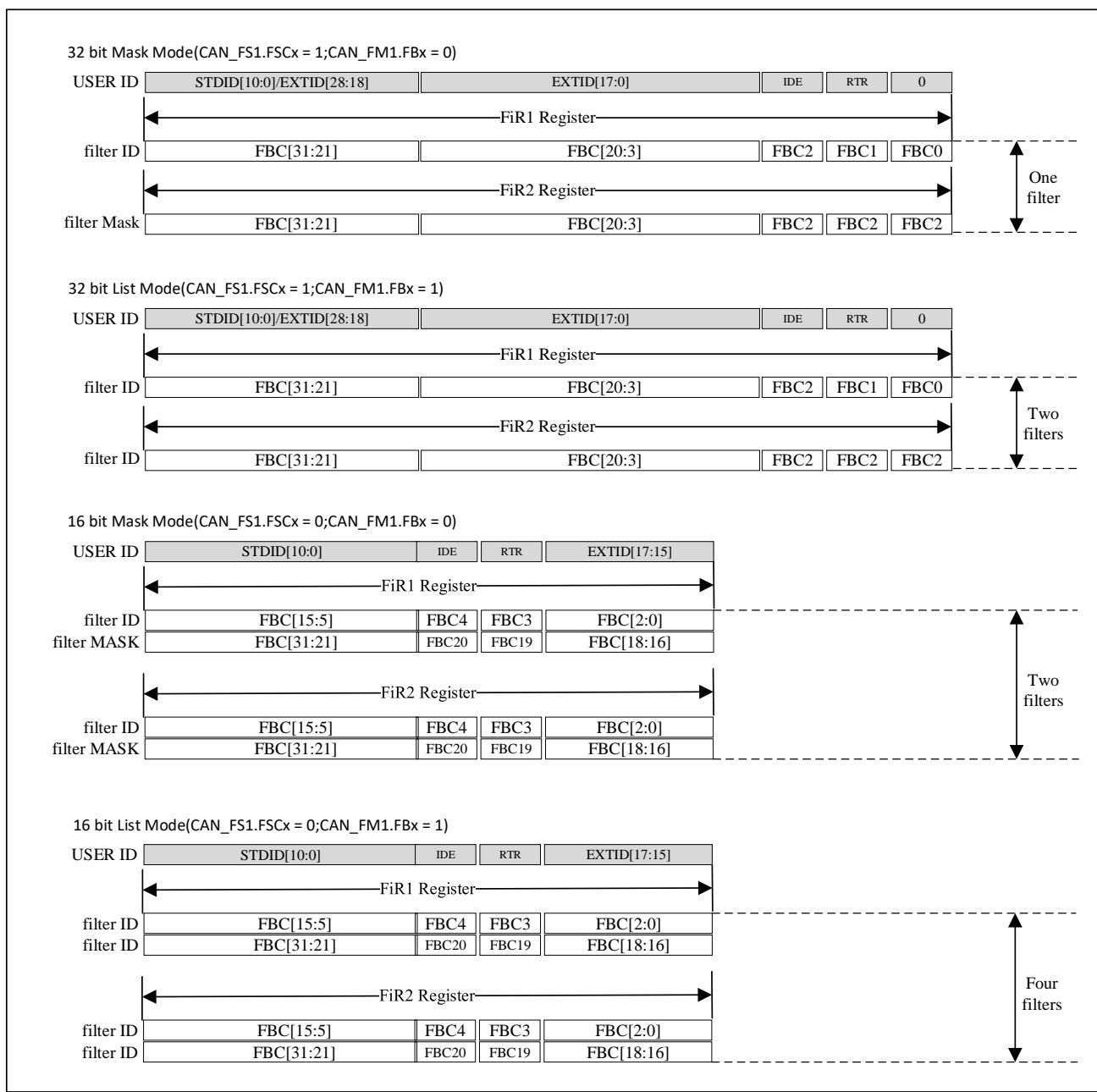
#### 22.4.5.1 过滤器位宽和模式的设置

根据过滤器组的模式和位宽设置, 对过滤器组中的每个过滤器进行编号 (过滤器编号, 从 0 到一个确定的最大值)。过滤器配置见下图。过滤器组可以通过对应的 CAN\_FMC 寄存器进行配置。应用程序未使用的过滤器组应保持禁用状态。在配置过滤器组之前, 必须通过清除 CAN\_FA1.FAC 位将其设置为禁用状态。

通过设置相应的 CAN\_FS1.FSCx 位, 您可以配置过滤器组的位宽, 参见图 22-9。

通过 CAN\_FM1.FBx 位, 可以将对应的掩码/标识符寄存器配置为标识符列表或标识符掩码模式。过滤器组设置为工作在掩码模式, 可过滤出一组标识符。过滤器组设置为在标识符列表模式下工作, 可过滤掉一个标识符。

图 22-9 过滤器位宽设置-寄存器组织



### 22.4.5.2 可变位宽

每个过滤器组的位宽可以独立配置。每个过滤器组可配置为一个 32 位过滤器：包括 STDID[10:0]、EXTID[17:0]、IDE 和 RTRQ 位；或两个 16 位过滤器，包括 STDID[10:0]、IDE、RTRQ 和 EXTID[17:15]位，见图 22-9。此外，过滤器可以配置为两种不同的模式，即掩码模式和标识符列表模式。

#### 掩码模式

过滤器 ID 用于存储标识符格式，过滤器掩码用于指示哪些位必须检查，哪些位可以忽略。

#### 标识符列表模式

过滤器 ID 用于存储标识符格式。此时没有掩码进行比较，可以使用掩码位多存储一个过滤器 ID。但此时消息的标识需要与过滤器 ID 格式完全一致，否则无法通过过滤器。

### 22.4.5.3 过滤匹配序列号

CAN 内核收到有效消息后，会将消息 ID 与过滤器一一匹配，直到有一个过滤器通过或所有过滤器都失败。如果此消息未能通过任何启用的过滤器，则它将被丢弃。否则当 CAN 内核找到 ID 可以通过的第一个过滤器时，它会将过滤器索引与 CAN 报文打包并根据过滤器设置存储在 SRAM 中的接收 FIFO 中（CAN\_FFA1 决定存储在哪个 FIFO 中）。用户可以在 CAN\_RMDTx 寄存器的 FMI[7:0]位中找到滤波器索引。此过滤器匹配索引可以帮助识别它在此接收 FIFO 中的消息类型。

过滤器匹配序列号有两种使用方式。第一个是将过滤器匹配序列号与一系列预期值进行比较。另一种是使用过滤器匹配的序列号作为索引来访问目标地址。在对过滤器进行编号时，不考虑过滤器组是否处于活动状态。此外，每个 FIFO 为其关联的过滤器编号。请参考以下示例。

对于掩码模式的过滤器，软件只需要比较需要的掩码位（必须匹配的位）。对于标识符列表模式的过滤器（非筛选过滤器），软件无需直接与标识符进行比较。

表 22-1 过滤器编号示例

指向 FIFOx	过滤器组	过滤器模式	FIFO0 过滤器编号
FIFO	0	32 位掩码模式	0
	2	16 位掩码模式	1/2
	5	32 位列表模式	3/4
	7	16 位列表模式	5/6/7/8
	9	32 位列表模式	9/10
	11	16 位列表模式	11/12/13/14
	13	32 位掩码模式	15
指向 FIFOx	过滤器组	过滤器模式	FIFO1 过滤器编号
FIFO1	1	32 位列表模式	0/1
	3	16 位列表模式	2/3/4/5
	4	32 位掩码模式	6
	6	16 位掩码模式	7/8
	8	32 位掩码模式	9
	10	16 位掩码模式	10/11
	12	32 位列表模式	12/13

#### 22.4.5.4 过滤器优先规则

根据过滤器的不同配置，一个消息标识符可以被多个过滤器过滤是可能的；在这种情况下，首先根据位宽确定接收邮箱中存储的过滤器匹配序列号，32位宽的过滤器比16位宽的过滤器具有更高的优先级。对于具有相同位宽的过滤器，标识符列表模式优先于掩码模式。如果过滤器具有相同的位宽和模式，则优先级由过滤器组编号确定，编号较小的过滤器组优先级较高。在过滤器组内，过滤器编号越小，优先级越高。

图 22-10 过滤机制示例

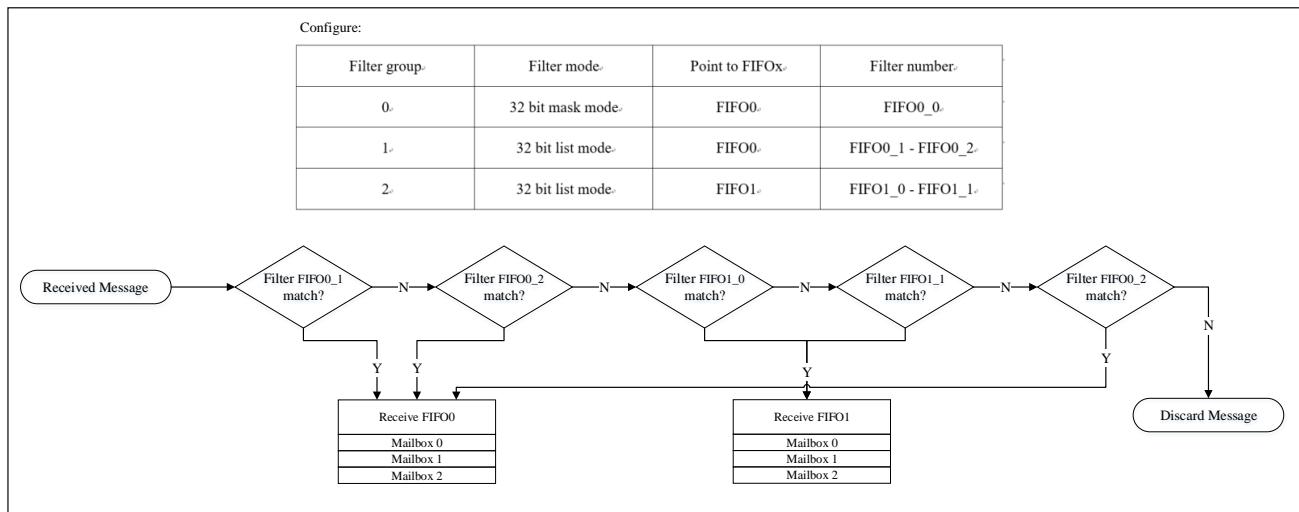


图 22-10 说明了 CAN 的过滤规则。收到消息时，首先将其标识符与标识符列表模式中配置的过滤器进行比较。如果匹配，则将消息存储在关联的 FIFO 中，并将匹配过滤器的序列号存储在过滤器匹配序列号中。如果不匹配，则将消息标识符与掩码模式中配置的过滤器进行比较。如果消息标识符与过滤器中的任何标识符不匹配，硬件将自动丢弃该消息而无需软件干预。

#### 22.4.6 消息存储

邮箱包含与消息相关的所有信息：标识符、数据、控制、状态和时间戳信息。邮箱是软件和硬件之间传递消息的接口。

##### 22.4.6.1 发送邮箱

在启用发送请求之前，应通过软件将消息写入一个空的发送邮箱。您可以通过 CAN\_TSTS 寄存器查询发送状态。

表 22-2 发送邮箱寄存器列表

相对发送邮箱的基地址偏移	寄存器名称
0	CAN_TMIx
4	CAN_TMDTx
8	CAN_TMDLx
12	CAN_TMDHx

## 22.4.6.2 接收邮箱(FIFO)

CAN\_RMDTx.FMI[7:0]字段可以存储过滤器匹配序列号, CAN\_RMDTx.MTIM[15:0]字段可以存储 16 位时间戳。软件可以访问接收 FIFO 的输出邮箱读取接收到的报文.一旦软件对报文进行了处理, 比如读出, 软件应该设置 CAN\_RFFx.RFFOM 位来释放相应的接收 FIFO, 为以后的报文预留存储空间.

表 22-3 接收邮箱寄存器列表

相应接收邮箱的基地址偏移	寄存器名称
0	CAN_RMIx
4	CAN_RMDTx
8	CAN_RMDLx
12	CAN_RMDHx

## 22.4.7 位时序

位时间特性逻辑通过采样监控串行 CAN 总线, 并通过与帧起始位的边沿同步并与下一个边沿重新同步来调整其采样点。为避免软件编程错误, 设置位时间特性寄存器 (CAN\_BTIM) 只能在 CAN 初始化时进行。其操作可以简单理解为将每个位时间分为三段: 同步段 (SYNC\_SEG)、时间段 1 (BS1) 和时间段 2 (BS2)。通常, 位变化将发生在 SYNC\_SEG 中。其值固定为 1 个时间单位( $1 \times t_{CAN}$ )。

BS1 定义了采样点的位置。它包括 CAN 标准中的 PROP\_SEG 和 PHASE\_SEG1。其值可编成 1~16 个时间单位, 但为了补偿网络中不同节点频率差异引起的相位正向漂移, 也可自动扩展。

在 BS2 中, 它定义了发送点的位置。它代表 CAN 标准中的 PHASE\_SEG2。它的值可以编程为 1 到 8 个时间单位, 但也可以自动缩短以补偿相位的负向漂移。

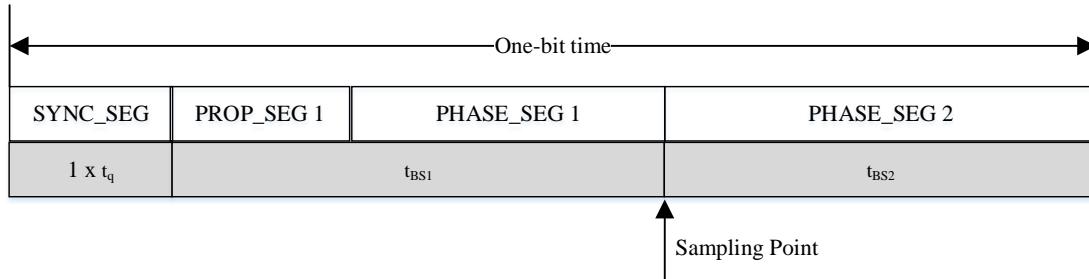
如果在 BS1 中检测到有效转换但在 SYNC\_SEG 中没有检测到, 则 BS1 的时间最多延长 RSJW 以延迟采样点。反之, 如果在 BS2 中检测到有效转换但在 SYNC\_SEG 中没有检测到, 则 BS2 的时间最多缩短 RSJW 以提前采样点。

在上面的描述中, RSJW (重新同步跳转宽度) 定义了每个比特可以延长或缩短多少时间单元的上限。它的值可以编程为 1 到 4 个时间单位。有效转换定义为 CAN 本身不发送隐性位时从显性位到隐性位的第一次转换。

注:

1. CAN 位的时间特性和重同步机制详见 ISO11898 标准。
2. 为了提高 CAN 位时间精度, 不推荐使用 HSI 作为时钟源

图 22-11 位时序



Notes :

$$t_{PCLK} = \text{time period of the APB1 clock},$$

$$t_q = (\text{CAN_BTIM.BRTP}[9:0] + 1) \times t_{PCLK},$$

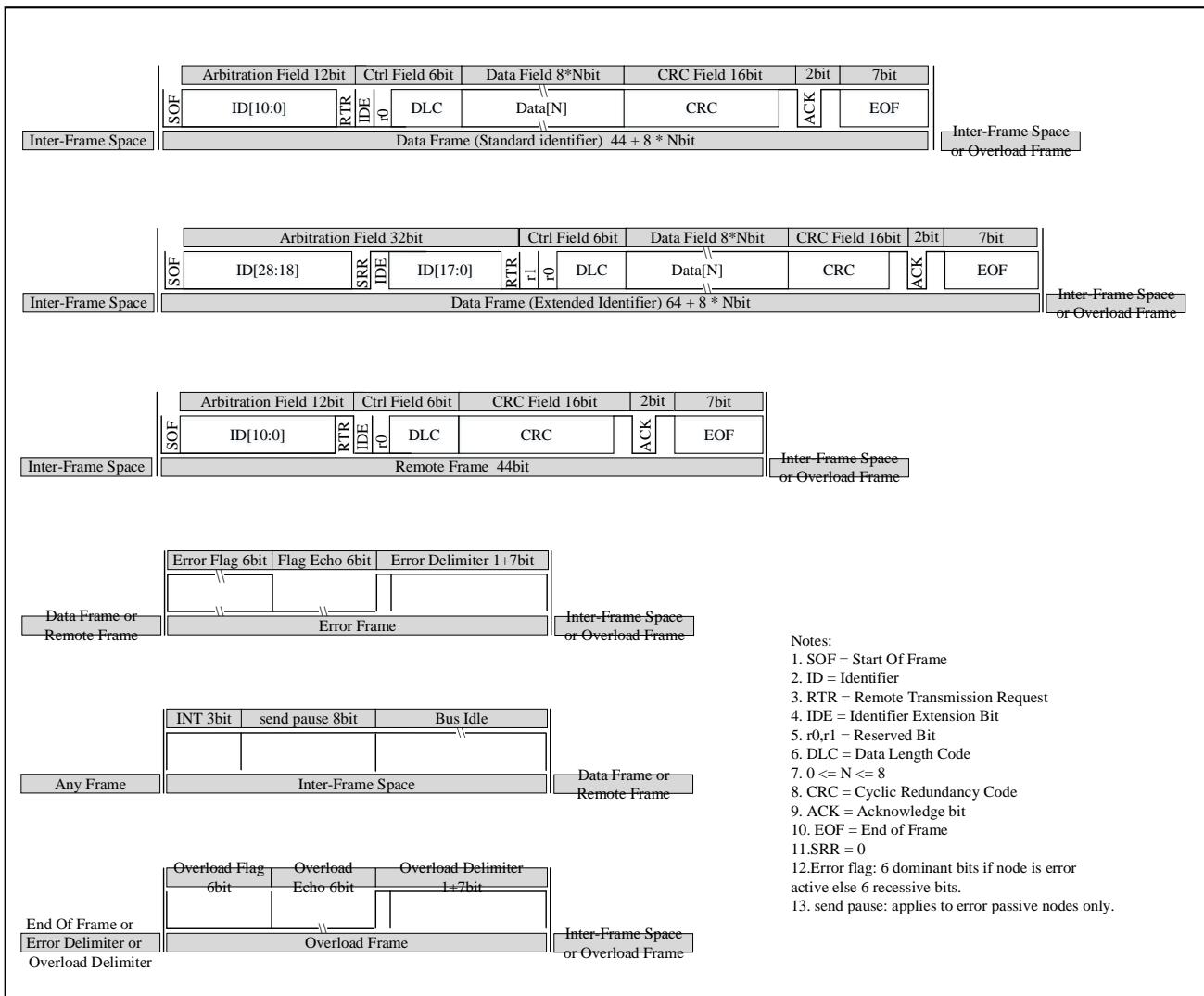
$$t_{BS1} = t_q \times (\text{CAN_BTIM.TBS1}[3:0] + 1),$$

$$t_{BS2} = t_q \times (\text{CAN_BTIM.TBS2}[2:0] + 1),$$

$$\text{One-bit time} = 1 \times t_q + t_{BS1} + t_{BS2}$$

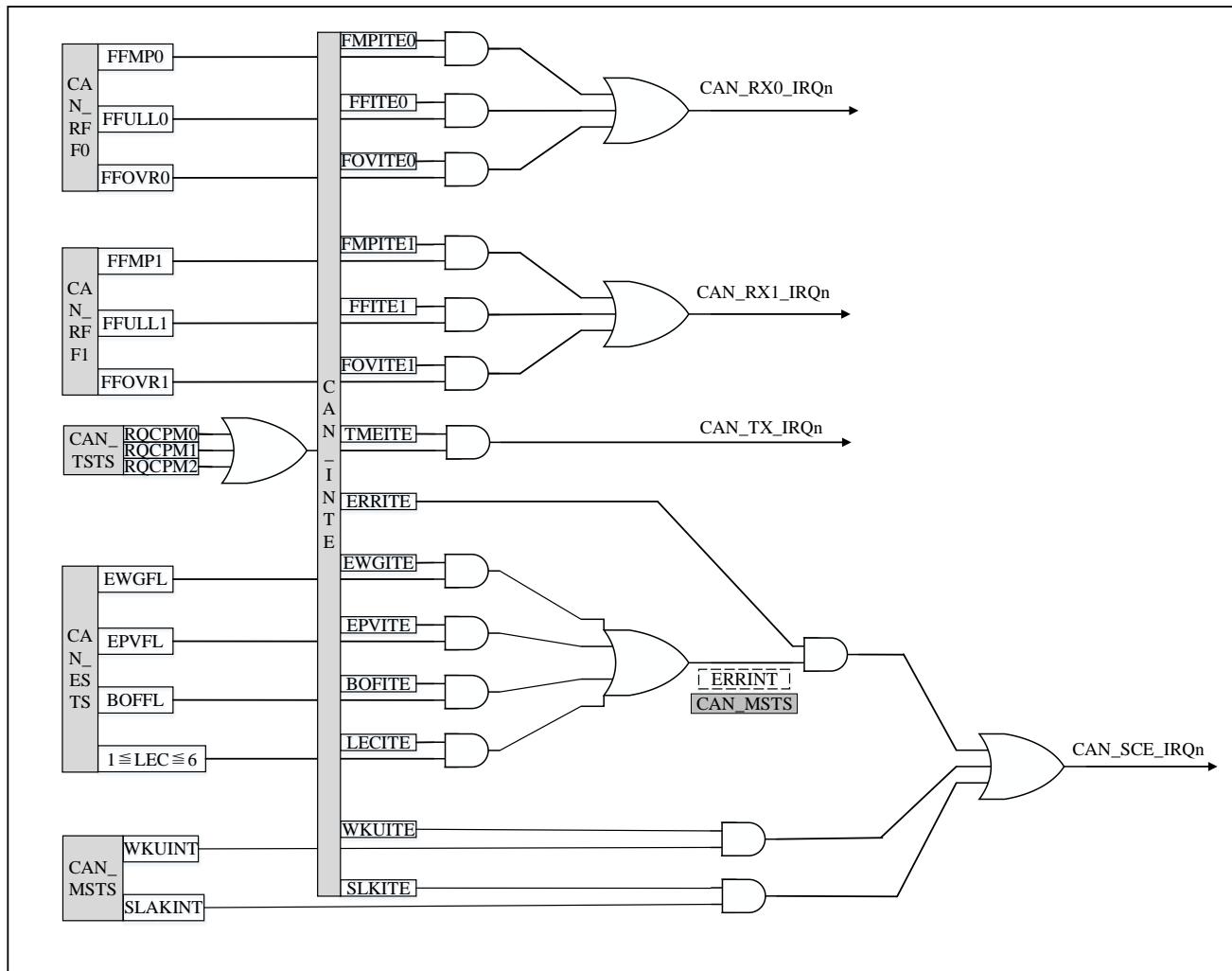
$$\text{BaudRate} = \frac{1}{\text{One-bit time}}$$

图 22-12 各类帧格式



## 22.5 CAN 中断

图 22-13 事件标志和中断产生



CAN 有四个中断向量。通过设置 CAN 中断启用寄存器 (CAN\_INTE)，您可以单独启用或禁用每个中断源。以下是可以产生每个中断的事件。

- FIFO0 中断(CAN\_RX0\_IRQn):

FIFO0 收到一条新消息，并且 CAN\_RFF0.FFMP0 位不是'00';

当 FIFO0 变满时，并且 CAN\_RFF0.FFULL0 位被置位；

当 FIFO0 溢出，并且 CAN\_RFF0.FFOVR0 位被置位。

- FIFO1 中断(CAN\_RX1\_IRQn):

FIFO1 接收到一条新消息，并且 CAN\_RFF1.FFMP1 位不是“00”。

当 FIFO1 变满时，并且 CAN\_RFF1.FFULL1 位被置位。

当 FIFO1 溢出，并且 CAN\_RFF1.FFOVR1 位被置位。

- 发送中断(CAN\_TX\_IRQn):

发送邮箱 x 变空，对应的 CAN\_TSTS.RQCPM $x$  位被置位( $x=1/2/3$ )。

#### ■ 错误和状态转变中断(CAN\_SCE\_IRQn):

CAN 进入睡眠模式；

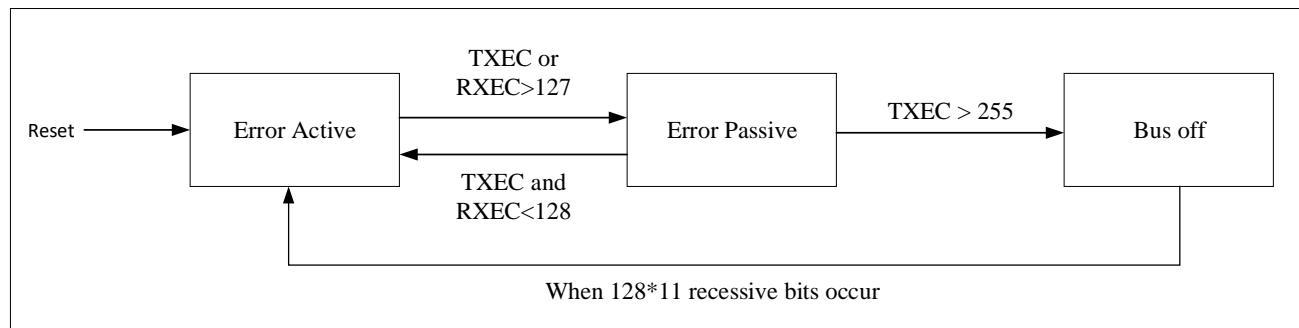
唤醒条件，在 CAN 接收引脚上监控帧开始位(SOF)。

错误情况，请参考 CAN 错误状态寄存器(CAN\_ESTS)了解错误详情。

### 22.5.1 错误管理

如 CAN 协议所述，错误管理完全由硬件通过发送错误计数器（CAN\_ESTS.TXEC 字段）和接收错误计数器（CAN\_ESTS.RXEC 字段）来实现。计数器值会根据错误情况增加或减少。如果您想了解有关 CAN\_ESTS.TXEC 和 CAN\_ESTS.RXEC 管理的更多详细信息，请参阅 CAN 标准。

图 22-14 CAN 错误状态框图



软件可以读取发送/接收错误计数器的值来判断 CAN 网络的稳定性，读取 CAN\_ESTS.LEC[2:0]位可以得到当前错误状态的详细信息。更重要的是，通过设置 CAN\_INTE 寄存器（如 CAN\_INTE.LECITE 位），软件可以灵活控制检测到错误时中断的产生。

### 22.5.2 总线关闭恢复

当 TXEC 大于 255 时，CAN\_ESTS.BOFFL 位置位，表示 CAN 总线关闭。此时，CAN 无法接收和发送消息。

在正常模式下，根据 CAN\_MCTRL.ABOM 位，CAN 可以自动或根据软件的要求，从总线关闭状态恢复，并切换到主动错误状态。如果设置了 CAN\_MCTRL.ABOM 位，恢复过程将在进入总线关闭状态后自动启动。否则，软件必须请求 CAN 进入然后退出初始化模式后，才会开始恢复过程。在这两种情况下，CAN 都必须等待 CAN 标准中描述的恢复过程，即在 CANRX 引脚上检测到 128\*11 个连续的隐性位。

在初始化模式下，CAN 不会监控 CANRX 引脚的状态，因此无法完成恢复过程。

## 22.6 CAN 配置流程

本章将介绍 CAN 的常用配置过程，其他详细信息如各模式的功能和寄存器位将在本手册的其他部分进行介绍。CAN 配置流程可以分为多个阶段。只要满足先前的要求（例如，过滤器值），就可以随时更改某些配置。

#### ■ 准备阶段：

1. 配置 RCC 使能 CAN 时钟
2. 配置 RCC 使能 AFIO 和 GPIO 时钟

3. 写入 GPIO 寄存器以将 CAN TX 和 CAN RX 信号映射到所需的 GPIO 引脚。

■ 基础配置阶段:

1. 复位后 CAN 设备以睡眠模式启动。
2. 通过清除 CAN\_MCTRL.SLPRQ 位退出睡眠模式。
3. 通过设置 CAN\_MCTRL.INIRQ 位进入初始化模式。
4. 等待 CAN\_MSTS.INIAK 位变为 1 (进入初始化模式)。
5. 通过将值写入 CAN\_BTIM.BSJW、CAN\_BTIM.TBS2、CAN\_BTIM.TBS1 和 CAN\_BTIM.BRTP 位来配置 CAN 的位时序。CAN 总线的波特率由以下公式定义:

$$\text{波特率} = \frac{1}{(1 + (TBS1 + 1) + (TBS2 + 1)) * (BRTP * t_{pclk})}$$

6. 通过写入寄存器中的 CAN\_BTIM.SLM (静默) 或 CAN\_BTIM.LBM 来配置 CAN 的工作模式选项。
7. 通过 CAN\_MCTRL 配置 CAN 行为 (TTCM、ABOM、AWKUM、NART、RFLM、TXFP)。该寄存器中的大部分配置都可以即时更改，但建议不要这样做。否则在几个周期内，CAN 行为将变得不可预测。
8. 通过清除 CAN\_MCTRL.INIRQ 位退出初始化模式。
9. 等待 CAN\_MSTS.INIAK 位变为 0 (退出初始化模式)。
10. 用户可以使用过滤器来过滤他们想要接收的消息。要配置滤波器，用户需要将“1”写入 CAN\_FMC.FINITM 位以请求滤波器进入初始化模式。当滤波器处于初始化模式时，CAN 停止接收。
11. 为每个过滤器配置工作模式 (CAN\_FM1)、过滤器比例 (CAN\_FS1) 和过滤器分配 (CAN\_FFA1)。用户还可以在此期间更改过滤器值(CAN\_FiRx)。完成过滤器配置后，清除 CAN\_FMC.FINITM 位以退出过滤器初始化。

■ 发送:

1. 使能必要的发送相关中断 CAN\_INTE.TMEITE 位。
2. 检查 CAN\_TSTS 中每个邮箱的状态位。如果任何一个 TMEMx(x=0~2)为‘1’的邮箱，用户可以将等待发送的消息写入对应的邮箱地址。CAN\_TMIx.TXRQ(x=0~2)位必须在该邮箱被编程后写入“1”。
3. 一段时间后或等待发送中断后，返回检查 CAN\_TSTS 中的发送状态。重复步骤 2~3 进行新消息传输。

■ 接收:

1. 当相应的过滤器被禁用时，用户也可以更改过滤器值 (CAN\_FiRx)。要禁用某个过滤器，用户需要将“0”写入 CAN\_FA1 寄存器中的相应位。
2. 在 CAN\_INTE 寄存器中配置接收相关的中断。
3. 一旦 CAN 接收到报文并将其存储在接收 FIFO 中，用户需要按时读取相应的 FIFO 并通过向寄存器 CAN\_RFFx(x=0,1)中的 RFFOMx 写入“1”释放接收邮箱。

## 22.7 CAN 寄存器

这些外设寄存器必须以字的方式 (32 位) 操作。

### 22.7.1 寄存器描述

#### 22.7.1.1 寄存器访问保护

当 CAN 节点正常工作时，不正确的访问/修改某些配置寄存器可能会导致节点出现硬件错误，暂时干扰整个 CAN 网络。因此，只有在 CAN 内核处于初始化模式时才允许修改 CAN\_BTIM 寄存器。

只有当发送邮箱状态位 CAN\_TSTS.TMEM=1 时用户才能修改发送邮箱的数据.

### 22.7.1.2 控制和状态寄存器

通过配置这些寄存器，用户可以：配置 CAN 参数，例如工作模式和波特率；开始消息发送；处理消息接收；中断设置；读取诊断信息。

### 22.7.1.3 邮箱寄存器描述

发送邮箱和接收邮箱基本相同，只是接收邮箱是只读的，包含 CAN\_RMDTx.FMI 字段。发送邮箱为空时可写。

注意：设置了对应的 CAN\_TSTS.TMEM 位，表示发送邮箱为空。

有 3 个发送邮箱和 2 个接收 FIFO。每个接收 FIFO 有 3 个邮箱，只能访问 FIFO 中第一个接收到的报文。

每个邮箱包含 4 个寄存器：

FIFO0 包含 CAN\_RMI0、CAN\_RMDT0、CAN\_RM\_DL0、CAN\_RMDH0；

FIFO1 包含 CAN\_RMI1、CAN\_RMDT1、CAN\_RMDL1、CAN\_RMDH1；

发送邮箱（x）包含 CAN\_TMIx、CAN\_TMDTx、CAN\_TMDLx、CAN\_TMDHx；x=0,1,2。

#### 22.7.1.4 过滤寄存器描述

只有在关闭相应的过滤器组或设置了 CAN\_FMC.FINITM 位时才能修改过滤器的值。另外，只有当整个滤波器设置为初始化模式（即 CAN\_FMC.FINITM=1）时，才能修改滤波器的设置，即可以修改 CAN\_FM1、CAN\_FS1 和 CAN\_FFA1 寄存器。

### 22.7.2 CAN 寄存器总览

表 22-4 CAN 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
	Reset Value																																								
010h	CAN_RFF1																																								
	Reset Value																																								
014h	CAN_INTE																																								
	Reset Value																																								
018h	CAN_ESTS																																								
	Reset Value	0	SLM	0	LBM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
01Ch	CAN_BTIM																																								
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
020h - 17Fh																																									
180h	CAN_TMI0																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0							
184h	CAN_TMDT0																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
188h	CAN_TMDL0																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
18Ch	CAN_TMDH0																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
190h	CAN_TMI1																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0							
194h	CAN_TMDT1																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
198h	CAN_TMDL1																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
19Ch	CAN_TMDH1																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x							
1A0h	CAN_TMI2																																								
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	0							
1A4h	CAN_TMDT2																																								



Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
210h																																		
214h	CAN_FFA1																																	
	Reset Value																																	
218h																																		
21Ch	CAN_FA1																																	
	Reset Value																																	
220h																																		
224h - 23Fh																																		
240h	CAN_F0B1																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
244h	CAN_F0B2																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
248h	CAN_F1B1																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
24Ch	CAN_F1B2																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
.	.																																	
2A8h	CAN_F13B1																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
2ACh	CAN_F13B2																																	
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		

## 22.7.3 CAN 控制和状态寄存器

寄存器描述中使用的缩写，请参考 1.1 部分。

### 22.7.3.1 CAN 主控制寄存器(CAN\_MCTRL)

偏移地址:0x00

复位值:0x0001 0002

31	Reserved												17	16
15	14	Reserved	8	7	6	5	4	3	2	1	rw	0		
MRST			TTCM	ABOM	AWKUM	NART	RFLM	TXFP	SLPRQ	INIRQ	rs			
			rw	rw	rw	rw	rw	rw	rw	rw				

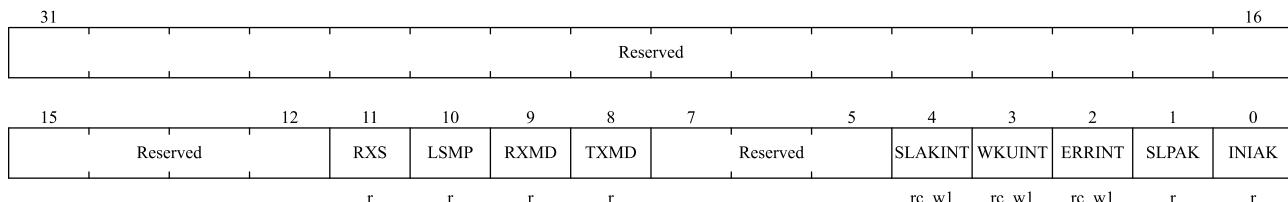
位域	名称	描述
31:17	Reserved	保留，必须保持复位值
16	DBGF	<p>调试冻结 (Debug freeze)</p> <p>0: 在调试时，CAN 照常工作</p> <p>1: 在调试时，冻结 CAN 的接收/发送。仍然可以正常地读写和控制接收 FIFO。。</p> <p>注：CAN 冻结时必须设置 DBG_CTRL.CAN_STOP 位，请参考 22.3.7：调试模式。</p>
15	MRST	<p>CAN 软件复位 (bxCAN software master reset)</p> <p>0: 该外设正常工作；</p> <p>1: 强制复位 CAN，之后 CAN 进入睡眠模式，CAN_RFFx.FFMP 位和 CAN_MCTRL 寄存器被初始化为其复位值。之后，硬件自动清除该位。</p>
14:8	Reserved	保留，必须保持复位值
7	TTCM	<p>时间触发通信模式 (Time triggered communication mode)</p> <p>0: 禁用时间触发通信模式；</p> <p>1: 使能时间触发通信模式。</p> <p>注：关于时间触发通信方式的更多信息，请参考22.4.2：时间触发通信方模式。</p>
6	ABOM	<p>自动离线 (Bus-Off) 管理 (Automatic bus-off management)</p> <p>该位确定 CAN 硬件可以退出总线关闭状态的条件。</p> <p>0: 退出总线关闭状态的过程是软件设置 CAN_MCTRL.INIRQ 位然后清零后，一旦硬件检测到 128 个连续的 11 位隐性位，就退出总线关闭状态；</p> <p>1: 一旦硬件检测到 128 个连续的 11 位隐性位，将自动退出总线关闭状态。</p> <p>注：有关总线关闭状态的更多信息，请参阅 22.5.1：错误管理。</p>
5	AWKUM	<p>自动唤醒模式 (Automatic wakeup mode)</p> <p>该位决定 CAN 在睡眠模式下是被硬件唤醒还是软件唤醒。</p> <p>0: 软件通过清除 CAN_MCTRL.SLPRQ 位唤醒睡眠模式；</p> <p>1: 睡眠模式由硬件通过检测 CAN 报文自动唤醒。</p> <p>在唤醒的同时，硬件自动清除 CAN_MSTS.SLPRQ 和 CAN_MSTS.SLPAK</p>

位域	名称	描述
		位。
4	NART	禁止报文自动重传 (No automatic retransmission) 0: 根据 CAN 标准, 当 CAN 硬件发送报文失败时, 会自动重传, 直到发送成功; 1: CAN 报文只发送一次, 与发送结果无关 (成功、错误或仲裁失败)。
3	RFLM	接收 FIFO 锁定模式 (Receive FIFO locked mode) 0: 接收溢出时 FIFO 不锁定, 当接收 FIFO 的报文没有被读出时, 下一个接收到的报文会覆盖上一条报文; 1: 接收溢出时锁定 FIFO。当接收 FIFO 的报文没有被读出时, 下一个接收到的报文将被丢弃。
2	TXFP	发送 FIFO 优先级 (Transmit FIFO priority) 当有多个消息同时等待发送时, 该位决定这些消息的发送顺序。 0: 优先级由消息的标识符决定; 1: 优先级由发送请求的顺序决定。
1	SLPRQ	睡眠模式请求 (Sleep mode request) 软件可以通过设置该位请求 CAN 进入睡眠模式, 一旦当前 CAN 活动 (发送或接收报文) 结束, CAN 将进入睡眠模式。 软件清零使 CAN 退出睡眠模式。 当 CAN_MCTRL.AWKUM 位置位并且在 CAN Rx 信号中检测到 SOF 位时, 硬件会清除该位。 该位在复位后置位, 即 CAN 在复位后处于睡眠模式。
0	INIRQ	初始化请求 (Initialization request) 软件清除该位可以使 CAN 退出初始化模式: 当 CAN 离开初始化模式进入正常模式时, 需要在接收引脚上检测到 11 个连续的隐性位, CAN 将同步并准备好接收和发送数据, 此时硬件相应地清除 CAN_MSTS.INIAK 位。 通过软件设置该位使 CAN 从正常操作模式进入初始化模式: 一旦当前 CAN 活动 (发送或接收) 结束, 此时硬件设置 CAN_MSTS.INIAK 位, CAN 进入初始化模式。

### 22.7.3.2 CAN 主状态寄存器 (CAN\_MSTS)

偏移地址: 0x04

复位值: 0x0000c02



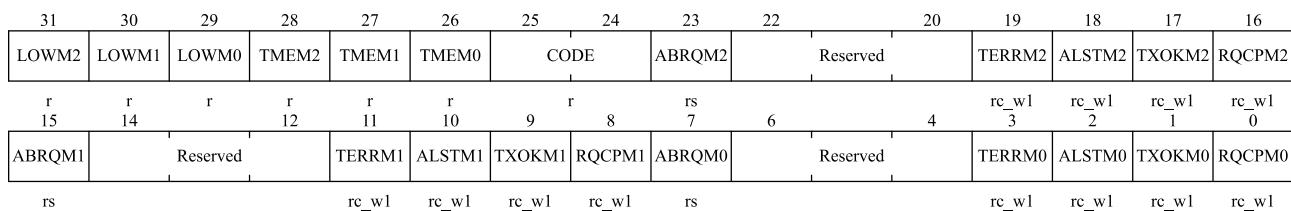
位域	名称	描述
31:12	Reserved	保留, 必须保持复位值
11	RXS	CAN 接收电平 (CAN Rx signal)

位域	名称	描述
		该位反映 CAN 接收引脚 (CAN_RX) 的实际电平。
10	LSMP	上次采样值 (Last sample point) CAN 接收引脚的上次采样值 (对应于当前接收位的值)。
9	RXMD	接收模式 (Receive mode) 该位为'1'表示 CAN 当前为接收器。
8	TXMD	发送模式 (Transmit mode) 该位为'1'表示 CAN 当前为发送器。
7:5	Reserved	保留, 必须保持复位值
4	SLAKINT	睡眠确认中断 (Sleep acknowledge interrupt) 当 CAN_INTE.SLKITE=1 时, 一旦 CAN 进入睡眠模式, 硬件会设置该位, 然后触发相应的中断。当该位置位时, 如果 CAN_INTE.SLKITE 位置位, 将产生状态改变中断。 软件可以清零该位, 当 CAN_MSTS.SLPAK 位清零时, 硬件也会清零该位。 注: 当 CAN_INTE.SLKITE=0 时, 不应查询该位, 但应查询 CAN_MSTS.SLPAK 位以了解睡眠状态。
3	WKUINT	唤醒中断 (Wakeup interrupt) 当 CAN 处于睡眠状态时, 一旦检测到帧起始位 (SOF), 硬件将设置该位; 如果设置了 CAN_INTE.WKUIITE 位, 则会生成状态改变中断。 该位由软件清零。
2	ERRINT	错误中断 (Error interrupt) 当检测到错误时, 会设置 CAN_ESTS 寄存器的某个位, 如果 CAN_INTE 寄存器的相应中断使能位也被设置, 则硬件会设置该位; 如果设置了 CAN_INTE.ERRITE 位, 则会生成状态更改中断。该位由软件清零。
1	SLPAK	睡眠模式确认 (Sleep acknowledge) 该位由硬件置位, 表示软件 CAN 模块处于睡眠模式。该位用于确认软件请求进入睡眠模式 (设置 CAN_MCTRL.SLPRQ 位)。 当 CAN 退出睡眠模式 (CAN 离开睡眠模式并进入正常模式, 需要与 CAN 总线同步) 时, 硬件清零该位。这里与 CAN 总线同步意味着硬件需要检测 CAN 的 RX 引脚上的 11 个连续的隐性位。 注: 通过软件或硬件清除 CAN_MCTRL.SLPRQ 位将启动退出睡眠模式的过程。有关清除 CAN_MCTRL.SLPRQ 位的详细信息, 请参见 CAN_MCTRL.AWKUM 位的说明。
0	INIAK	初始化确认 (Initialization acknowledge) 该位由硬件置位, 表示软件 CAN 模块处于初始化模式。该位是软件请求进入初始化模式的确认 (CAN_MCTRL.INIRQ 位被设置)。 当 CAN 退出初始化模式 (CAN 离开初始化模式并进入正常模式, 需要与 CAN 总线同步) 时, 硬件清零该位。这里与 CAN 总线同步意味着硬件需要检测 CAN 的 RX 引脚上的 11 个连续的隐性位。

### 22.7.3.3 CAN 发送状态寄存器 (CAN\_TSTS)

偏移地址: 0x08

复位值: 0x1C00 0000



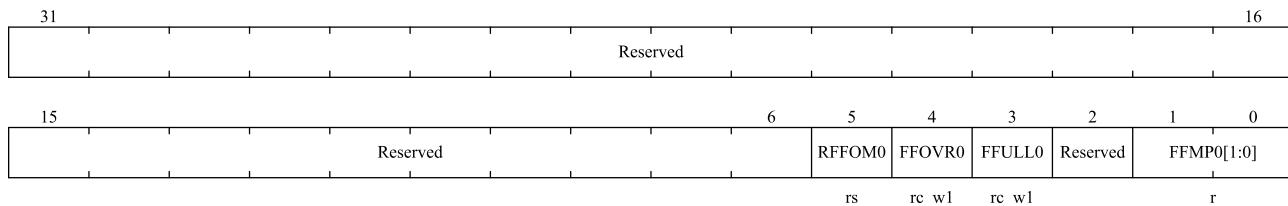
位域	名称	描述
31	LOWM2	邮箱 2 最低优先级标志 (Lowest priority flag for mailbox 2) 当多个邮箱在等待发送报文, 且邮箱 2 的优先级最低时, 硬件对该位置'1'。
30	LOWM1	邮箱 1 最低优先级标志 (Lowest priority flag for mailbox 1) 当多个邮箱在等待发送报文, 且邮箱 1 的优先级最低时, 硬件对该位置'1'。
29	LOWM0	邮箱 0 最低优先级标志 (Lowest priority flag for mailbox 0) 当多个邮箱在等待发送报文, 且邮箱 0 的优先级最低时, 硬件对该位置'1'。 注: 如果只有 1 个邮箱在等待, 则 CAN_TSTS.LOWM[2:0] 被清'0'。
28	TMEM2	发送邮箱 2 空 (Transmit mailbox 2 empty) 当邮箱 2 中没有等待发送的报文时, 硬件对该位置'1'。
27	TMEM1	发送邮箱 1 空 (Transmit mailbox 1 empty) 当邮箱 1 中没有等待发送的报文时, 硬件对该位置'1'。
26	TMEM0	发送邮箱 0 空 (Transmit mailbox 0 empty) 当邮箱 0 中没有等待发送的报文时, 硬件对该位置'1'。
25:24	CODE[1:0]	邮箱号 (Mailbox code) 当有至少 1 个发送邮箱为空时, 这 2 位表示下一个空的发送邮箱号。 当所有的发送邮箱都为空时, 这 2 位表示优先级最低的那个发送邮箱号。
23	ABRQM2	邮箱 2 中止发送 (Abort request for mailbox 2) 设置该位, 软件可以停止邮箱 2 的发送请求, 当邮箱 2 的发送消息空闲时, 硬件清零该位。如果邮箱 2 中没有等待发送的消息, 则设置该位无效。
22:20	Reserved	保留, 必须保持复位值
19	TERRM2	邮箱 2 发送失败 (Transmission error of mailbox 2) 当邮箱 2 因为出错而导致发送失败时, 对该位置'1'。
18	ALSTM2	邮箱 2 仲裁丢失 (Arbitration lost for mailbox 2) 当邮箱 2 因为仲裁丢失而导致发送失败时, 对该位置'1'。
17	TXOKM2	邮箱 2 发送成功 (Transmission OK of mailbox 2) 每次在邮箱 2 进行发送尝试后, 硬件对该位进行更新: 0: 上一次发送尝试尚未成功; 1: 上次发送尝试成功。 当邮箱 2 的发送请求被成功完成后, 硬件对该位置'1'。请参见图 22-7。
16	RQCPM2	邮箱 2 请求完成 (Request completed mailbox 2) 当邮箱 2 的上次请求 (发送或中止) 完成时, 硬件将设置该位。

位域	名称	描述
		软件向该位写'1'可以清零；当硬件接收到发送请求时，也可以清零该位（CAN_TMI2.TXRQ 位被置位）。 当该位清零时，邮箱 2 的其他发送状态位（CAN_TSTS.TXOKM2、CAN_TSTS.ALSTM2 和 CAN_TSTS.TERRM2 位）也被清零。
15	ABRQM1	邮箱 1 中止发送（Abort request for mailbox 1） 设置该位，软件可以停止邮箱 1 的发送请求，当邮箱 1 的发送报文空闲时，硬件清零该位。如果邮箱 1 中没有等待发送的消息，则设置该位无效。
14:12	Reserved	保留，必须保持复位值
11	TERRM1	邮箱 1 发送失败（Transmission error of mailbox 1） 当邮箱 1 因为出错而导致发送失败时，对该位置'1'。
10	ALSTM1	邮箱 1 仲裁丢失（Arbitration lost for mailbox 1） 当邮箱 1 因为仲裁丢失而导致发送失败时，对该位置'1'。
9	TXOKM1	邮箱 1 发送成功（Transmission OK of mailbox 1） 每次在邮箱 1 进行发送尝试后，硬件对该位进行更新： 0：上次发送尝试尚未成功； 1：上次发送尝试成功。 当邮箱 1 的发送请求被成功完成后，硬件对该位置'1'。请参见图 22-7。
8	RQCPM1	邮箱 1 请求完成（Request completed mailbox 1） 当邮箱 1 的上次请求（发送或中止）完成时，硬件设置该位。 软件向该位写'1'可以清零；当硬件接收到发送请求时，也可以清除该位（CAN_TMI1.TXRQ 位被置位）。 当该位清零时，邮箱 1 的其他发送状态位（CAN_TSTS.TXOKM1、CAN_TSTS.ALSTM1 和 CAN_TSTS.TERRM1 位）也被清零。
7	ABRQM0	邮箱 0 中止发送（Abort request for mailbox 0） 软件可以通过设置该位来停止邮箱 0 的发送请求，当邮箱 0 的发送消息空闲时硬件清零该位。如果邮箱 0 中没有等待发送的消息，则设置该位无效。
6:4	Reserved	保留，必须保持复位值
3	TERRM0	邮箱 0 发送失败（Transmission error of mailbox 0） 当邮箱 0 因为出错而导致发送失败时，对该位置'1'。
2	ALSTM0	邮箱 0 仲裁丢失（Arbitration lost for mailbox 0） 当邮箱 0 因为仲裁丢失而导致发送失败时，对该位置'1'。
1	TXOKM0	邮箱 0 发送成功（Transmission OK of mailbox 0） 每次尝试发送邮箱 0 后，硬件都会更新此位： 0：上次发送尝试尚未成功； 1：上次发送尝试成功。 当邮箱 0 的发送请求成功完成时，硬件设置该位。 参见图 22-7。
0	RQCPM0	邮箱 0 请求完成（Request completed mailbox 0） 当邮箱 0 的上次请求（发送或中止）完成时，硬件设置该位。 软件向该位写'1'可以清零；当硬件接收到发送请求时，也可以清除该位（CAN_TMI0.TXRQ 位被置位）。 当该位清零时，邮箱 0 的其他发送状态位（CAN_TSTS.TXOKM0、CAN_TSTS.ALSTM0 和 CAN_TSTS.TERRM0 位）也被清零。

### 22.7.3.4 CAN 接收 FIFO0 寄存器 (CAN\_RFF0)

偏移地址: 0x0c

复位值: 0x0000 0000

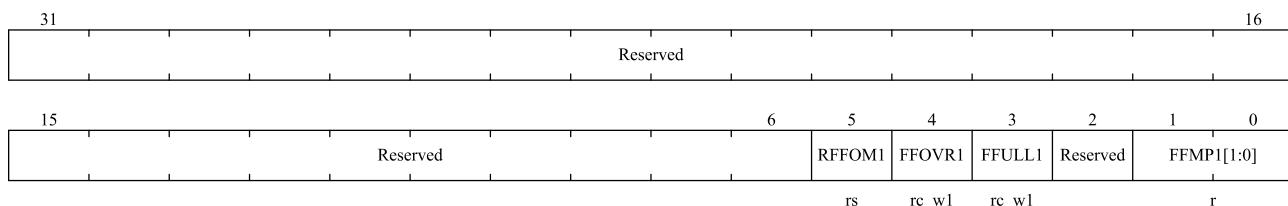


位域	名称	描述
31:6	Reserved	保留, 必须保持复位值
5	RFFOM0	释放接收 FIFO 0 输出邮箱 (Release FIFO 0 output mailbox) 软件通过设置该位来释放接收 FIFO 的输出邮箱。如果接收 FIFO 为空, 则设置该位没有影响, 即只有在 FIFO 中有报文时, 设置该位才有意义。如果 FIFO 中有两条以上的消息, 由于 FIFO 的特性, 软件需要释放输出邮箱才能访问第二条消息。 当输出邮箱被释放时, 硬件清除该位。
4	FFOVR0	FIFO 0 溢出 (FIFO 0 overrun) 当 FIFO0 已满, 又收到新的报文且报文符合过滤条件, 硬件对该位置'1'。该位由软件清'0'。
3	FFULL0	FIFO 0 满 (FIFO 0 full) 当 FIFO 0 中有 3 个报文时, 硬件对该位置'1'。该位由软件清'0'。
2	Reserved	保留, 必须保持复位值
1:0	FFMP0[1:0]	FIFO 0 报文数目 (FIFO 0 message pending) FIFO 0 报文数目这两位反映了当前存储在接收 FIFO0 中的报文数量。每在接收 FIFO0 中存储一条新报文, 硬件将 CAN_RFF0.RFFOM0 加 1。 每次软件向 CAN_RFF0.RFFOM0 位写入“1”以释放输出邮箱时, CAN_RFF0.FFMP0 减 1, 直到为 0。

### 22.7.3.5 CAN 接收 FIFO 1 寄存器 (CAN\_RFF1)

偏移地址: 0x10

复位值: 0x0000 0000



位域	名称	描述
31:6	Reserved	保留, 必须保持复位值
5	RFFOM1	释放接收 FIFO 1 输出邮箱 (Release FIFO 1 output mailbox)

位域	名称	描述
		软件通过对该位置'1'来释放接收 FIFO 的输出邮箱。如果接收 FIFO 为空，那么对该位置'1'没有任何效果，即只有当 FIFO 中有报文时对该位置'1'才有意义。如果 FIFO 中有 2 个以上的报文，由于 FIFO 的特点，软件需要释放输出邮箱才能访问第 2 个报文。 当输出邮箱被释放时，硬件对该位清'0'。
4	FFOVR1	FIFO 1 溢出 (FIFO 1 overrun) 当 FIFO 1 已满，又收到新的报文且报文符合过滤条件，硬件对该位置'1'。该位由软件清'0'。
3	FFULL1	FIFO 1 满 (FIFO 1 full) 当 FIFO 1 中有 3 个报文时，硬件对该位置'1'。该位由软件清'0'。
2	Reserved	保留，必须保持复位值
1:0	FFMP1[1:0]	FIFO 1 报文数目 (FIFO 1 message pending) FIFO 1 中的报文数 这两位反映了当前接收 FIFO 1 中存储的报文数量。接收 FIFO 1 中每存储一条新报文，硬件将 CAN_RFF1.RFFOM1 加 1。 每次软件通过向 CAN_RFF1.RFFOM1 位写入“1”来释放输出邮箱时，CAN_RFF1.RFFOM1 减 1，直到为 0。

### 22.7.3.6 CAN 中断使能寄存器(CAN\_INTE)

偏移地址: 0x14

复位值: 0x0000 0000

31	Reserved														18	17	16
15	14	12	11	10	9	8	7	6	5	4	3	2	rw	1	rw	0	
ERRITE	Reserved	LECITE	BOFITE	EPVITE	EWGITE	Reserved	FOVITE1	FFITE1	FMPITE1	FOVITE0	FFITE0	FMPITE0	TMEITE	rw	rw	rw	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

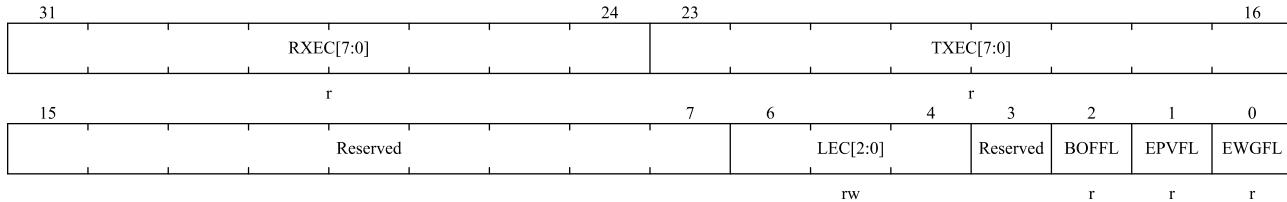
位域	名称	描述
31:18	Reserved	保留，必须保持复位值
17	SLKITE	睡眠中断使能 (Sleep interrupt enable) 0: 当 CAN_MSTS.SLAKINT 位置位时，不产生中断； 1: 当 CAN_MSTS.SLAKINT 位置位时，产生中断。
16	WKUITE	唤醒中断使能 (Wakeup interrupt enable) 0: 当 CAN_MSTS.WKUINT 位置位时，不产生中断； 1: 当 CAN_MSTS.WKUINT 位置位时，产生中断。
15	ERRITE	错误中断使能 (Error interrupt enable) 0: 当 CAN_ESTS 寄存器有错误发生时，不产生中断； 1: 当 CAN_ESTS 寄存器有错误发生时，产生中断。
14:12	Reserved	保留，必须保持复位值
11	LECITE	上次错误号中断使能 (Last error code interrupt enable) 0: 检测到错误时，当硬件置位 CAN_ESTS.LEC[2:0]时，CAN_MSTS.ERRINT 位不置位；

位域	名称	描述
		1: 检测到错误时, 当硬件设置 CAN_ESTS.LEC[2:0]时, 设置 CAN_MSTS.ERRINT 位。
10	BOFITE	离线中断使能 (Bus-off interrupt enable) 0: 当设置 CAN_ESTS.BOFFL 位时, 不设置 CAN_MSTS.ERRINT 位; 1: 设置 CAN_ESTS.BOFFL 位时, 设置 CAN_MSTS.ERRINT 位。
9	EPVITE	错误被动中断使能 (Error Passive Interrupt Enable) 0: 当设置 CAN_ESTS.EPVFL 位时, 不设置 CAN_MSTS.ERRINT 位; 1: 当设置 CAN_ESTS.EPVFL 位时, 设置 CAN_MSTS.ERRINT 位。
8	EWGITE	错误警告中断使能 (Error warning interrupt enable) 0: 当设置 CAN_ESTS.EWGFL 位时, 不设置 CAN_MSTS.ERRINT 位; 1: 当设置 CAN_ESTS.EWGFL 位时, 设置 CAN_MSTS.ERRINT 位。
7	Reserved	保留, 必须保持复位值
6	FOVITE1	FIFO 1 溢出中断使能 (FIFO overrun interrupt enable) 0: 当 CAN_RFF1.FOR 位置位时, 不产生中断; 1: 当 CAN_RFF1.FO 位置位时, 产生中断。
5	FFITE1	FIFO 1 满中断使能 (FIFO full interrupt enable) 0: 当 CAN_RFF1.FULL 位置位时, 不产生中断; 1: 当 CAN_RFF1.FULL 位置位时, 产生中断。
4	FMPITE1	FIFO 1 消息挂号中断使能 (FIFO message pending interrupt enable) 0: 当 CAN_RFF1.FFMP[1:0]位为非 0 时, 不产生中断; 1: 当 CAN_RFF1.FFMP[1:0]位不为 0 时, 产生中断。
3	FOVITE0	FIFO 0 溢出中断使能 (FIFO overrun interrupt enable) 0: 当 CAN_RFF0.FFOVR 位置位时, 不产生中断; 1: 当 CAN_RFF0.FFOVR 位置位时, 产生中断。
2	FFITE0	FIFO 0 满中断使能 (FIFO full interrupt enable) 0: 当 CAN_RFF0.FFULL 位置位时, 不产生中断; 1: 当 CAN_RFF0.FFULL 位置位时, 产生中断。
1	FMPITE0	FIFO 0 消息挂号中断使能 (FIFO message pending interrupt enable) 0: 当 CAN_RFF0.FFMP[1:0]位为非 0 时, 不产生中断; 1: 当 CAN_RFF0.FFMP[1:0]位不为 0 时, 产生中断。
0	TMEITE	发送邮箱空中断使能 (Transmit mailbox empty interrupt enable) 0: 当 CAN_TSTS.RPMx 位置位时, 不产生中断; 1: 当 CAN_TSTS.RPMx 位置位时, 产生中断。 注: 请参考 22.5 节 CAN 中断。

### 22.7.3.7 CAN 错误状态寄存器(CAN\_ESTS)

偏移地址: 0x18

复位值: 0x0000 0000



位域	名称	描述
31:24	RXEC[7:0]	<p>接收错误计数器 (Receive error counter)</p> <p>这个计数器按照 CAN 协议的故障界定机制的接收部分实现。按照 CAN 的标准，当接收出错时，根据出错的条件，该计数器加 1 或加 8；而在每次接收成功后，该计数器减 1，或当该计数器的值大于 127 时，设置它的值为 120。当该计数器的值超过 127 时，CAN 进入错误被动状态。</p>
23:16	TXEC[7:0]	<p>9 位发送错误计数器的低 8 位 (Least significant byte of the 9-bit transmit error counter)</p> <p>与上面相似，这个计数器按照 CAN 协议的故障界定机制的发送部分实现。</p>
15:7	Reserved	保留，必须保持复位值
6:4	LEC[2:0]	<p>上次错误代码 (Last error code)</p> <p>当 CAN 总线上检测到错误时，根据错误情况设置硬件。当正确发送或接收消息时，硬件将其值清除为“0”。</p> <p>硬件不使用错误代码 7，软件可以设置此值，以便检测到代码更新。</p> <ul style="list-style-type: none"> <li>000: 没有错误；</li> <li>001: 位填充错误；</li> <li>010: 错误的格式 (Form)；</li> <li>011: 确认 (ACK) 错误；</li> <li>100: 隐性错位 (CAN 传输隐性但在总线上检测显性)；</li> <li>101: 显性错位 (CAN 传输显性但在总线上检测到隐性)；</li> <li>110: CRC 错误；</li> <li>111: 软件设置。</li> </ul>
3	Reserved	保留，必须保持复位值
2	BOFFL	<p>离线标志 (Bus-off flag)</p> <p>当总线关闭时，硬件设置该位。当传输错误计数器 CAN_TSTS.TXEC 溢出，即大于 255 时，CAN 总线关闭。请参阅 22.5.1 部分。</p>
1	EPVFL	<p>错误被动标志 (Error passive flag)</p> <p>当出错次数达到错误被动的阈值时，硬件对该位置‘1’。</p> <p>(接收错误计数器或发送错误计数器的值&gt;127)。</p>
0	EWGFL	<p>错误警告标志 (Error warning flag)</p> <p>当出错次数达到警告的阈值时，硬件对该位置‘1’。</p> <p>(接收错误计数器或发送错误计数器的值≥96)。</p>

### 22.7.3.8 CAN 位时序寄存器 (CAN\_BTIM)

偏移地址: 0x1C

复位值: 0x0023 0000

注：当 CAN 处于初始化模式时，该寄存器只能由软件访问。

31	30	29	Reserved	26	25	24	23	22	20	19	16
SLM	LBM		Reserved		RSJW[1:0]	Reserved		TBS2[2:0]		TBS1[3:0]	
rw 15	rw 15			rw 10	rw 9			rw 0		rw 0	
			Reserved					BRTP[9:0]			
									rw		

位域	名称	描述
31	SLM	静默模式（用于调试）(Silent mode (debug)) 0: 正常状态； 1: 静默模式。
30	LBM	回环模式（用于调试）(Loop back mode (debug)) 0: 禁止回环模式； 1: 允许回环模式。
29:26	Reserved	保留，必须保持复位值
25:24	RSJW[1:0]	重新同步跳跃宽度（Resynchronization jump width） 为了重新同步，该位域定义了 CAN 硬件在每位中可以延长或缩短多少个时间单元的上限。 $t_{RJW} = t_{CAN} \times (RSJW[1:0] + 1)$ .
23	Reserved	保留，必须保持复位值
22:20	TBS2[2:0]	时间段 2 (Time segment 2) 该位域定义了时间段 2 占用了多少个时间单元 $t_{BS2} = t_{CAN} \times (TBS2[2:0] + 1)$ .
19:16	TBS1[3:0]	时间段 1 (Time segment 1) 该位域定义了时间段 1 占用了多少个时间单元 $t_{BS1} = t_{CAN} \times (TBS1[3:0] + 1)$ 有关位时间特性的详细信息，请参阅第 22.4.7 节：位时序。
15:10	Reserved	保留，必须保持复位值
9:0	BRTP[9:0]	波特率分频器 (Baud rate prescaler) 该位域定义了时间单元 ( $t_q$ ) 的时间长度 $t_q = (BRTP[9:0] + 1) \times t_{PCLK}$

## 22.7.4 CAN 邮箱寄存器

本节介绍发送和接收邮箱寄存器。有关寄存器映射的更多信息，请参阅 22.4.6 部分：消息存储。

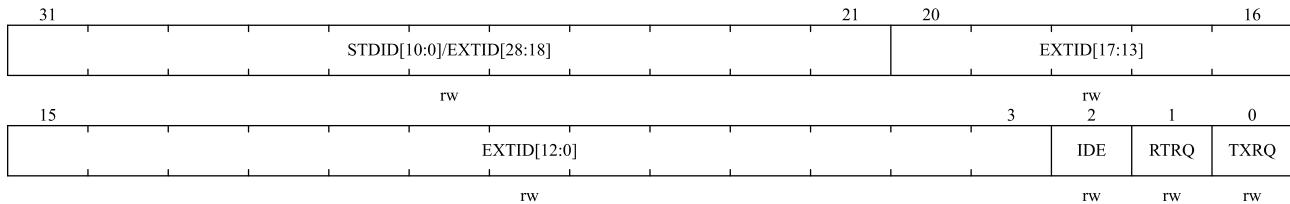
### 22.7.4.1 发送邮箱标识寄存器(CAN\_TMIx)(x=0..2)

偏移地址: 0x180, 0x190, 0x1A0

复位值: 0xXXXX XXXX, X=未定义位（除了第 0 位，复位时 TXRQ=0）

注: 1 当其所属的邮箱处在等待发送的状态时，该寄存器是写保护的；

2 该寄存器实现了发送请求控制功能（第 0 位）—复位值为 0。



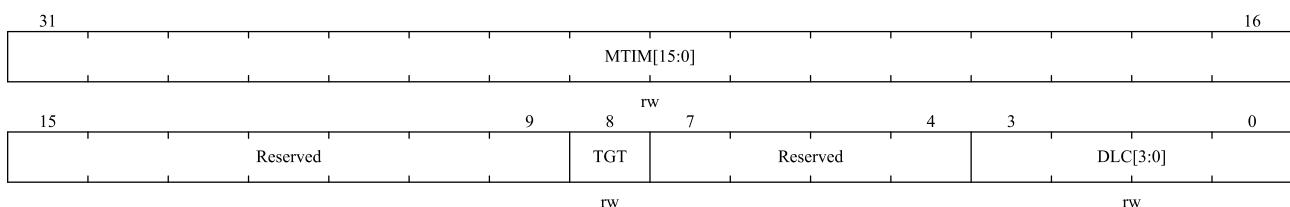
位域	名称	描述
31:21	STDID[10:0]/EXTID[28:18]	标准标识符或扩展标识符 (Standard identifier or extended identifier) 依据 IDE 位的内容，这些位或是标准标识符，或是扩展身份标识的高字节。
20:3	EXTID[17:0]	扩展标识符 (Extended identifier) 扩展身份标识的低字节。
2	IDE	标识符选择 (Identifier extension) 该位决定发送邮箱中报文使用的标识符类型 0: 使用标准标识符； 1: 使用扩展标识符。
1	RTRQ	远程发送请求 (Remote transmission request) 0: 数据帧； 1: 远程帧。
0	TXRQ	发送数据请求 (Transmit mailbox request) 由软件对其置'1'，来请求发送邮箱的数据。当数据发送完成，邮箱为空时，硬件对其进行清'0'。

#### 22.7.4.2 发送邮箱数据长度和时间戳寄存器 (CAN\_TMDTx)(x=0..2)

当邮箱不在空置状态时，该寄存器的所有位为写保护。

偏移地址: 0x184, 0x194, 0x1A4

复位值: 未定义



位域	名称	描述
31:16	MTIM[15:0]	报文时间戳 (Message time stamp) 该域包含了，在发送该报文 SOF 的时刻，16 位定时器的值。
15:9	Reserved	保留，必须保持复位值
8	TGT	发送时间戳 (Transmit global time) 该位仅在 CAN 处于时间触发通信模式时有效，即 CAN_MCTRL.TTCM 位置位。 0: 不发送时间戳 CAN_TMDTx.MTIM[15:0];

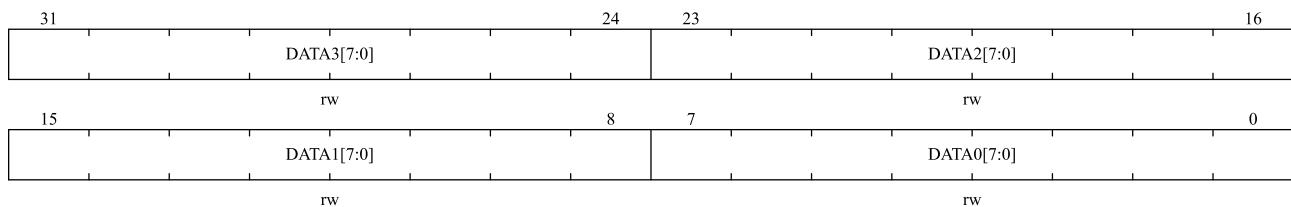
位域	名称	描述
		1: 发送时间戳 CAN_TMDTx.MTIM[15:0]。在长度为 8 的消息中，时间戳 CAN_TMDTx.MTIM[15:0]是发送的最后两个字节：CAN_TMDTx.MTIM[7:0] 是第七个字节，CAN_TMDTx.MTIM[15:8]是第八个字节。它们替换了写入 CAN_TMDHx[31:16]的数据(CAN_TMDLx.DATA6[7:0]和 CAN_TMDLx.DATA7[7:0])。为了发送时间戳的 2 个字节，DLC 必须编程为 8。
7:4	Reserved	保留，必须保持复位值
3:0	DLC[3:0]	发送数据长度 (Data length code) 该域指定了数据报文的数据长度或者远程帧请求的数据长度。1 个报文包含 0 到 8 个字节数据，而这由 DLC 决定。

#### 22.7.4.3 发送邮箱低字节数据寄存器(CAN\_TMDLx) (x=0..2)

当邮箱不在空置状态时，该寄存器的所有位为写保护。

偏移地址: 0x188, 0x198, 0x1A8

复位值: 未定义



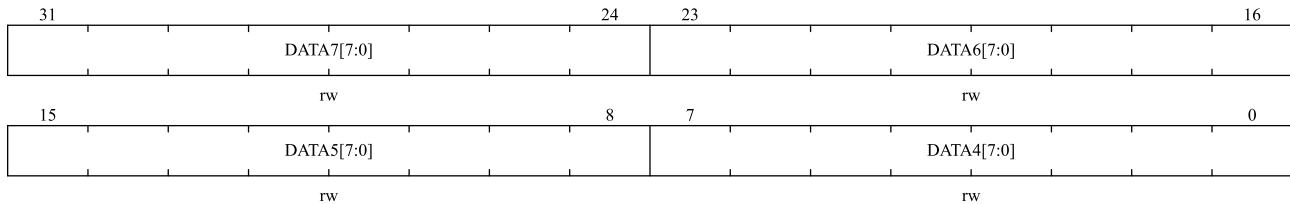
位域	名称	描述
31:24	DATA3[7:0]	数据字节 3 (Data byte 3) 报文的数据字节 3。
23:16	DATA2[7:0]	数据字节 2 (Data byte 2) 报文的数据字节 2。
15:8	DATA1[7:0]	数据字节 1 (Data byte 1) 报文的数据字节 1。
7:0	DATA0[7:0]	数据字节 0 (Data byte 0) 报文的数据字节 0。 注: 报文包含 0 到 8 个字节数据，且从字节 0 开始。

#### 22.7.4.4 发送邮箱高字节数据寄存器(CAN\_TMDHx) (x=0..2)

当邮箱不在空置状态时，该寄存器的所有位为写保护。

偏移地址: 0x18c, 0x19c, 0x1ac

复位值: 未定义



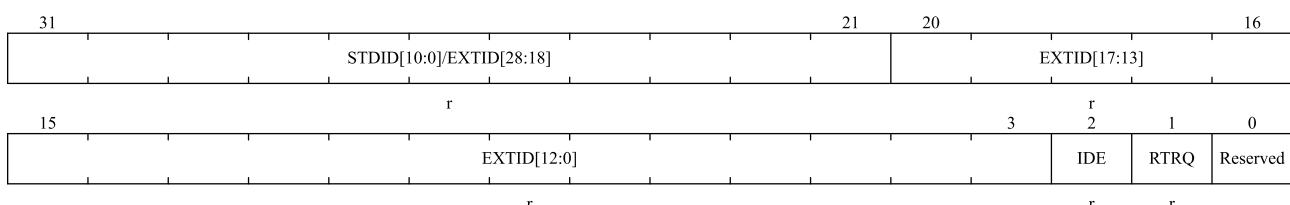
位域	名称	描述
31:24	DATA7[7:0]	数据字节 7 (Data byte 7) 报文的数据字节 7。 注意：如果 CAN_MCTRL.TTCM 位为“1”且此邮箱的 CAN_TMDTx.TGT 位也为“1”，则 DATA7 和 DATA6 将被时间戳替换。
23:16	DATA6[7:0]	数据字节 6 (Data byte 6) 报文的数据字节 6。
15:8	DATA5[7:0]	数据字节 5 (Data byte 5) 报文的数据字节 5。
7:0	DATA4[7:0]	数据字节 4 (Data byte 4) 报文的数据字节 4。

#### 22.7.4.5 接收 FIFO 邮箱标识符寄存器(CAN\_RMIx) (x=0..1)

偏移地址: 0x1B0, 0x1C0

复位值: 未定义

注: 所有接收邮箱寄存器都是只读的。



位域	名称	描述
31:21	STDID[10:0]/EXTID[28:18]	标准标识符或扩展标识符 (Standard identifier or extended identifier) 根据 CAN_RMIx.IDE 位的内容，这些位要么是标准标识符，要么是扩展标识符的高字节。
20:3	EXTID[17:0]	扩展标识符 (Extended identifier) 扩展身份标识的低字节。
2	IDE	标识符选择 (Identifier extension) 该位决定发送邮箱中报文使用的标识符类型 0: 使用标准标识符； 1: 使用扩展标识符。
1	RTRQ	远程发送请求 (Remote transmission request) 0: 数据帧； 1: 远程帧。

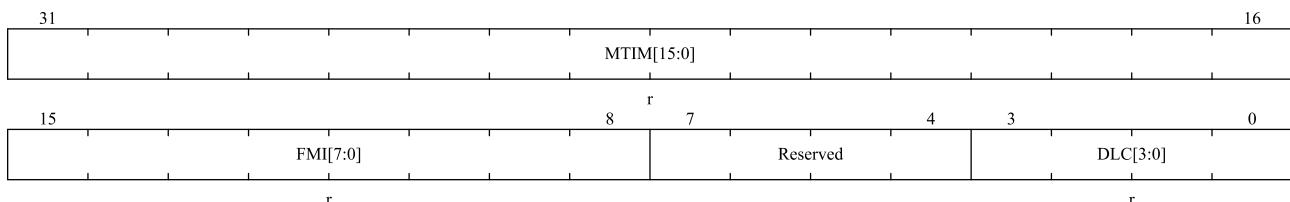
位域	名称	描述
0	Reserved	保留，必需保持复位值。

#### 22.7.4.6 接收 FIFO 邮箱数据长度和时间戳寄存器(CAN\_RMDTx)( x=0..1)

偏移地址: 0x1B4, 0x1C4

复位值: 未定义

注: 所有接收邮箱寄存器都是只读的。



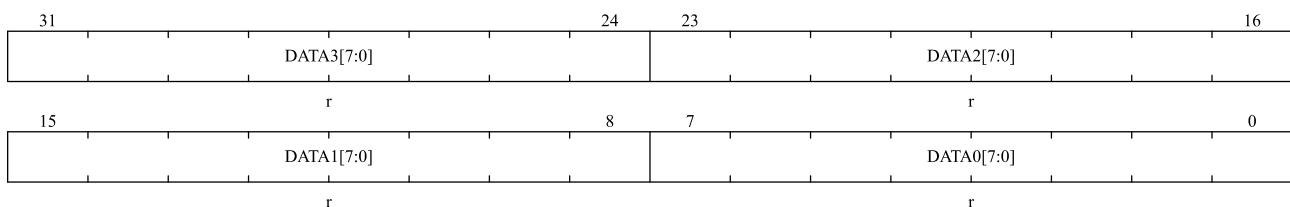
位域	名称	描述
31:16	MTIM[15:0]	报文时间戳 (Message time stamp) 该域包含了，在发送该报文 SOF 的时刻，16 位定时器的值。
15:8	FMI[7:0]	过滤器匹配序号 (Filter match index) 这是存储在邮箱中的信息传输的过滤器序列号。有关标识符过滤的详细信息，请参阅22.4.5 部分：标识符过滤。
7:4	Reserved	保留，必须保持复位值
3:0	DLC[3:0]	接收数据长度 (Data length code) 该域表明接收数据帧的数据长度 (0~8)。对于远程帧请求，数据长度 DLC 恒为 0。

#### 22.7.4.7 接收 FIFO 邮箱低字节数据寄存器(CAN\_RMDLx)( x=0..1)

偏移地址: 0x1B8, 0x1C8

复位值: 未定义

注: 所有接收邮箱寄存器都是只读的。



位域	名称	描述
31:24	DATA3[7:0]	数据字节 3 (Data byte 3) 报文的数据字节 3。
23:16	DATA2[7:0]	数据字节 2 (Data byte 2) 报文的数据字节 2。
15:8	DATA1[7:0]	数据字节 1 (Data byte 1)

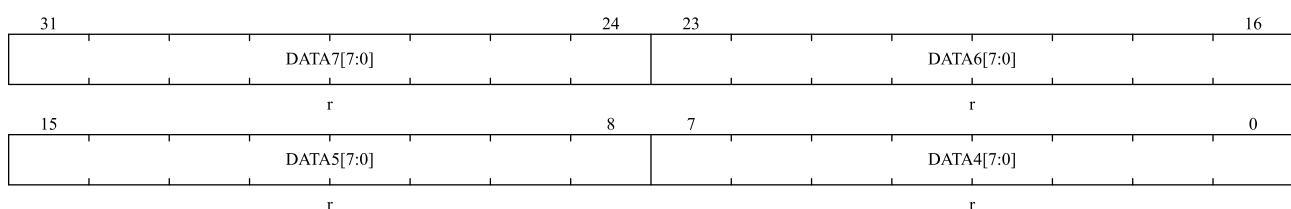
位域	名称	描述
		报文的数据字节 1。
7:0	DATA0[7:0]	数据字节 0 (Data byte 0) 报文的数据字节 0。 注：报文包含 0 到 8 个字节数据，且从字节 0 开始。

#### 22.7.4.8 接收 FIFO 邮箱低高字节数据寄存器(CAN\_RMDHx) (x=0..1)

偏移地址: 0x1BC, 0x1CC

复位值: 未定义

注：所有接收邮箱寄存器都是只读的。



位域	名称	描述
31:24	DATA7[7:0]	数据字节 7 (Data byte 7) 报文的数据字节 7。
23:16	DATA6[7:0]	数据字节 6 (Data byte 6) 报文的数据字节 6。
15:8	DATA5[7:0]	数据字节 5 (Data byte 5) 报文的数据字节 5。
7:0	DATA4[7:0]	数据字节 4 (Data byte 4) 报文的数据字节 4。

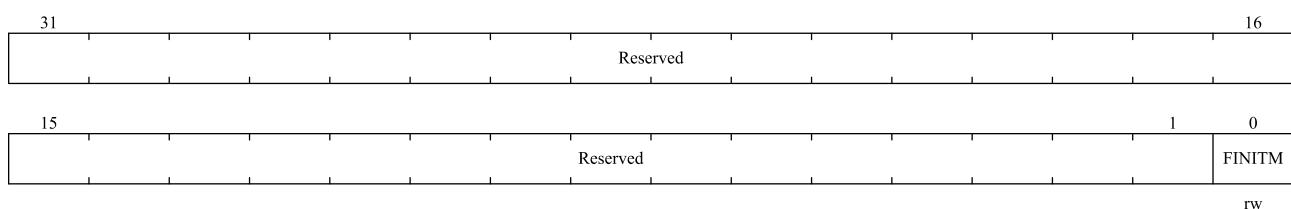
#### 22.7.5 CAN 过滤器寄存器

##### 22.7.5.1 CAN 过滤器主控制寄存器(CAN\_FMC)

偏移地址: 0x200

复位值: 0x2A1C 0E01

注：该寄存器的非保留位完全由软件控制。



位域	名称	描述
31:1	Reserved	保留，必须保持复位值

位域	名称	描述
0	FINITM	过滤器初始化模式 (Filter init mode) 针对所有过滤器组的初始化模式设置。 0: 过滤器组工作在正常模式; 1: 过滤器组工作在初始化模式。

### 22.7.5.2 CAN 过滤器模式寄存器(CAN\_FM1)

偏移地址: 0x204

复位值: 0x0000 0000

注: 当设置 CAN\_FMC.FINITM 位将滤波器置于初始化模式时, 才能写入该寄存器。

31															16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	FB13	FB12	FB11	FB10	FB9	FB8	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0	

rw      rw

位域	名称	描述
31:28	Reserved	保留, 必须保持复位值
13:0	Fbx	过滤器模式 (Filter mode) 过滤器组 x 的工作模式。 0: CAN_FiRx 的两个 32 位寄存器工作在标识符屏蔽模式; 1: CAN_FiRx 的两个 32 位寄存器工作在标识符列表模式。

### 22.7.5.3 CAN 过滤器位宽寄存器(CAN\_FS1)

偏移地址: 0x20C

复位值: 0x0000 0000

注: 当设置 CAN\_FMC.FINITM 位将滤波器置于初始化模式时, 才能写入该寄存器。

31															16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	FSC13	FSC12	FSC11	FSC10	FSC9	FSC8	FSC7	FSC6	FSC5	FSC4	FSC3	FSC2	FSC1	FSC0	

rw      rw

位域	名称	描述
31:28	Reserved	保留, 必须保持复位值
13:0	FSCx	过滤器位宽设置 (Filter scale configuration) 过滤器组 x (13~0) 的位宽。 0: 过滤器位宽为 2 个 16 位; 1: 过滤器位宽为单个 32 位。

## 22.7.5.4 CAN 过滤器 FIFO 关联寄存器 (CAN\_FFA1)

偏移地址: 0x214

复位值: 0x0000 0000

注: 当设置 CAN\_FMC.FINITM 位将滤波器置于初始化模式时, 才能写入该寄存器。

																17	16
																Reserved	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved	FAF13	FAF12	FAF11	FAF10	FAF9	FAF8	FAF7	FAF6	FAF5	FAF4	FAF3	FAF2	FAF1	FAF0			
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			

位域	名称	描述
31:28	Reserved	保留, 必须保持复位值
13:0	FAFx	过滤器 x 的过滤器 FIFO 分配 (Filter FIFO assignment for filter x) 报文在通过了某过滤器的过滤后, 将被存放到其关联的 FIFO 中。 0: 过滤器被关联到 FIFO0; 1: 过滤器被关联到 FIFO1。

## 22.7.5.5 CAN 过滤器激活寄存器(CAN\_FA1)

偏移地址: 0x21C

复位值: 0x0000 0000

																31	16
																Reserved	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved	FAC13	FAC12	FAC11	FAC10	FAC9	FAC8	FAC7	FAC6	FAC5	FAC4	FAC3	FAC2	FAC1	FAC0			
	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			

位域	名称	描述
31:28	Reserved	保留, 必须保持复位值
13:0	FACx	过滤器激活 (Filter active) 软件为某位设置“1”以激活相应的过滤器。只有在清除 CAN_FA1.FACx 位或设置 CAN_FMC.FINITM 位后, 才能修改相应的滤波器寄存器 i(CAN_FiR[2:1])。 0: 过滤器被禁用; 1: 过滤器被激活。

## 22.7.5.6 CAN 过滤器 i 寄存器 x (CAN\_FiRx) (i=0..13;x=1..2)

偏移地址: 0x240h, 0x31C

复位值: 未定义

注: 14 组滤波器:  $i = 0 \dots 13$ 。每组滤波器由两个 32 位寄存器 CAN\_FiR[2:1] 组成。

只有当相应的 CAN\_FA1.FACx 位被清零或 CAN\_FMC.FINIT 位被置位时，才能修改相应的过滤寄存器。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FBC31	FBC30	FBC29	FBC28	FBC27	FBC26	FBC25	FBC24	FBC23	FBC22	FBC21	FBC20	FBC19	FBC18	FBC17	FBC16
rw 15	rw 14	rw 13	rw 12	rw 11	rw 10	rw 9	rw 8	rw 7	rw 6	rw 5	rw 4	rw 3	rw 2	rw 1	rw 0
FBC15	FBC14	FBC13	FBC12	FBC11	FBC10	FBC9	FBC8	FBC7	FBC6	FBC5	FBC4	FBC3	FBC2	FBC1	FBC0

位域	名称	描述
31:0	FBC[31:0]	<p>过滤器位 (Filter bits)</p> <p>标识符模式</p> <p>寄存器的每位对应于所期望的标识符的相应位的电平。</p> <p>0: 期望相应位为显性位;</p> <p>1: 期望相应位为隐性位。</p> <p>屏蔽位模式</p> <p>寄存器的每位指示是否对应的标识符寄存器位一定要与期望的标识符的相应位一致。</p> <p>0: 不关心, 该位不用于比较;</p> <p>1: 必须匹配, 到来的标识符位必须与滤波器对应的标识符寄存器位相一致。</p>

**注:** 根据滤波器位宽和模式的不同设置, 滤波器组中的两个寄存器的功能是不同的。关于过滤器的映射、功能描述和屏蔽寄存器的关联, 请参见 22.4.5 章节: 标识符过滤。

**掩码模式**中的掩码/标识符寄存器与**标识符列表模式**中的寄存器位定义相同。

滤波器组寄存器地址见表 22-4。

## 23 调试支持 (DBG)

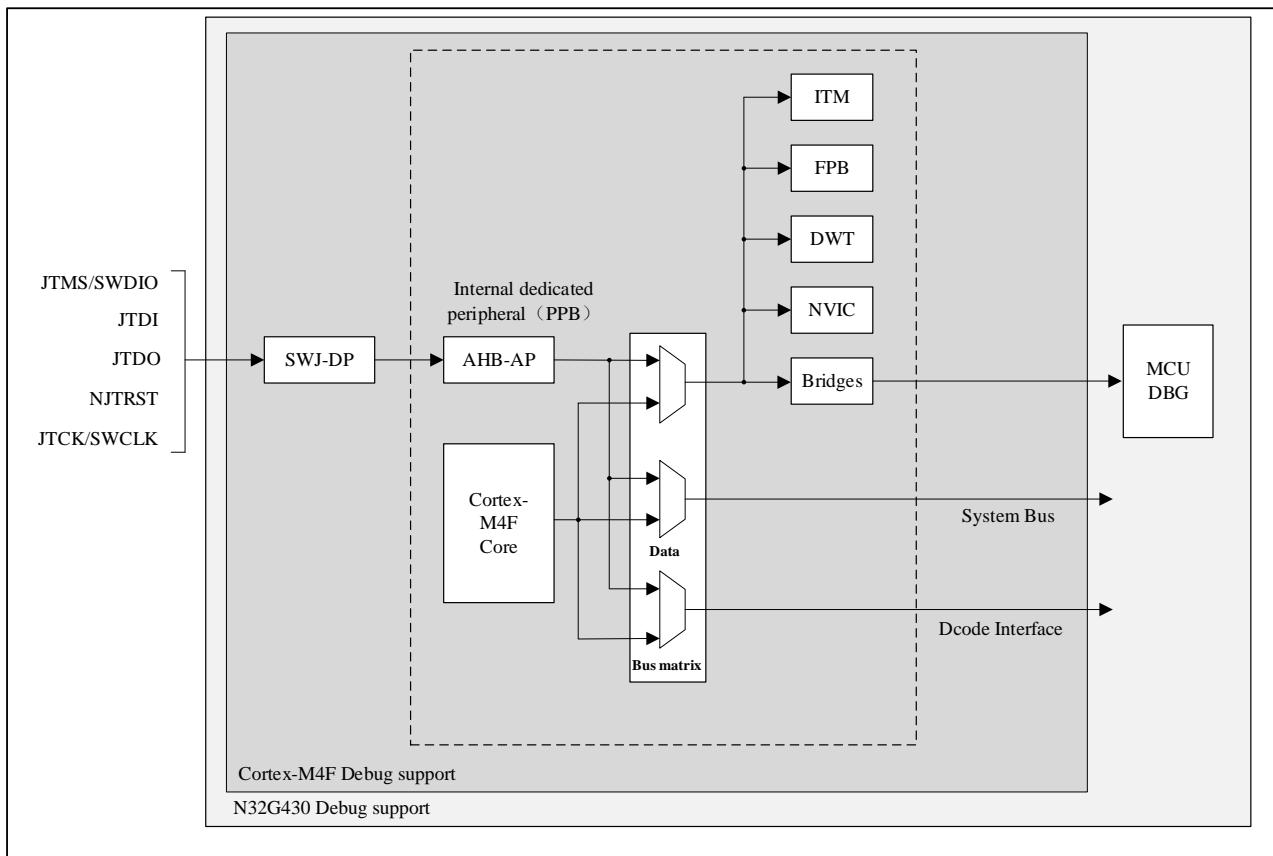
### 23.1 概述

N32G430 使用集成硬件调试模块的 Cortex®-M4 内核。支持指令断点（指令取值时停止）和数据断点（数据访问时停止）。当内核停止时，用户可以查看内核的内部状态和系统的外部状态。用户查询操作完成后，可以恢复内核和外设，继续执行相应的程序。N32G430 内核的硬件调试模块在连接到调试器时即可使用（在未被禁用的情况下）。

N32G430 支持以下调试接口：

- 串行接口
- JTAG 调试接口

图 23-1 N32G430 级和 Cortex®-M4 级调试框图



ARM Cortex™-M4F 内核硬件调试模块可以提供以下调试功能：

- SWJ-DP：串行/JTAG 调试端口
- AHB-AP：AHB 访问接口
- ITM：执行跟踪单元
- FPB：闪存指令断点

- DWT: 数据触发

参考文档:

- Cortex™-M4 技术参考手册 (TRM)
- ARM 调试接口 V5 结构规范
- ARM CoreSight 开发工具集 (r1p0 版) 技术参考手册

调试系统支持低功耗模式调试和部分外设调试。支持调试的外设包括: CAN、I2C 接口和 TIMER、WWDG、IWDG 模块。用户在进行低功耗调试或外设调试时, 需要将调试控制寄存器 (DBG\_CTRL) 的相应位置 1。

## 23.2 JTAG/SW 功能

调试工具可以通过上述的 SW 调试接口或 JTAG 调试接口调用调试功能。

### 23.2.1 切换 JTAG/SW 接口

芯片默认使用 JTAG 调试接口。如果需要切换调试接口, 可以通过以下操作进行 SW 接口和 JTAG 接口的相互切换:

JTAG 调试接口切换到 SW 调试接口:

1. 发送 50 个以上 TCK 周期的 TMS=1 信号;
2. 发送 16-bit TMS=111001110011110(0xE79E LSB) 信号;
3. 发送 50 个以上 TCK 周期的 TMS=1 信号。

SW 调试接口切换到 JTAG 调试接口:

1. 发送 50 个以上 TCK 周期的 TMS=1 信号;
2. 发送 16-bit TMS=1110011100111100(0xE73C LSB) 信号;
3. 发送 50 个以上 TCK 周期的 TMS=1 信号。

### 23.2.2 引脚分配

JTAG 调试接口包括五个引脚: JTCK (JTAG 时钟引脚)、JTMS (JTAG 模式选择引脚)、JTDI (JTAG 数据输入引脚)、JTDO (JTAG 数据输出引脚) 和 NJTRST (JTAG 数据复位引脚, 低电平复位引脚)。

SWD (串行调试) 接口包括两个引脚: SWCLK (时钟引脚) 和 SWDIO (数据输入输出引脚), 提供两个引脚的接口: 数据输入输出引脚 (SWDIO) 和时钟引脚 (SWCLK)。

JTAG 调试接口和 SW 调试接口的管脚分配见下表 (SWDIO 与 JTMS 复用, SWCLK 与 JTCK 复用):

表 23-1 调试端口引脚

调试端口	引脚分配
JTMS/SWDIO	PA13
JTCK/SWCLK	PA14
JTDI	PA15

JTDO	PB3
NJTRST	PB4

- 当 JTAG 调试接口和 SW 调试接口都使能时，复位后默认使用 5 线 JTAG 调试接口。
- 使用 JTAG 接口时，用户可以不使用 NJTRST 管脚。此时可将 NJTRST 管脚（PB4，内部硬件上拉）可以用作通用 GPIO。
- 使用 SW 接口时，JTDI(PA15)、JTDO(PB3)、NJTRST(PB4)三个引脚可作为通用 GPIO 使用。
- 不使用调试功能时，以上五个引脚可作为通用 GPIO 使用。

## 23.3 MCU 调试功能

### 23.3.1 低功耗模式支持

N32G430 可以提供多种低功耗模式（详见参考 3.电源控制章节）。在进行调试时，要保证内核的 FCLK 和 HCLK 是开启状态，并为内核调试提供必要的时钟。用户可以根据特定的操作（保证低功耗模式下 FCLK 或 HCLK 的输出），在低功耗模式下进行 MCU 调试。

如果用户想要在低功耗模式下调试 MCU，首先需要调试器配置与低功耗模式相关的寄存器：

- SLEEP 模式：

调试器需要配置 DBG\_CTRL 寄存器的 SLEEP 位（配置 HCLK 输出同 FCLK 相同频率）。

- STOP 模式：

调试器需要配置 DBG\_CTRL 寄存器的 STOP 位，启动内部 RC 振荡器，为 HCLK 和 FCLK 提供时钟。

### 23.3.2 外设调试支持

当 DBG\_CTRL 寄存器中外设控制对应位置 1 时，当内核停止后相应的外设进入调试状态：

- Timer 外设：定时器的计数器停止并进入调试状态；
- I2C 外设：I2C 的 SMBUS 保持状态并进入调试状态；
- WWDG/IWDG 外设：WWDG/IWDG 计数器停止并进入调试状态；
- CAN 外设：CAN 接口接收寄存器停止并进入调试状态。

## 23.4 寄存器

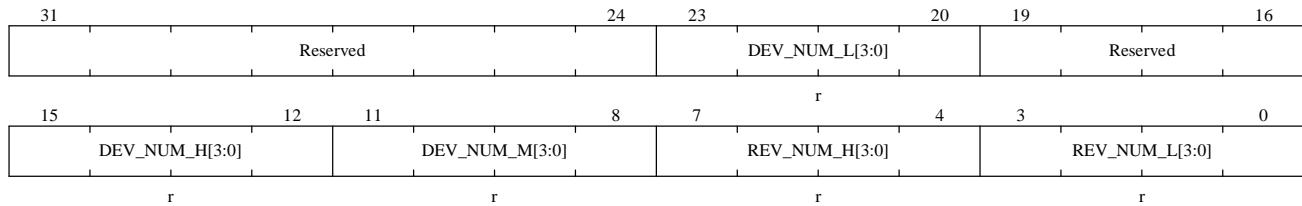
### 23.4.1 DBG 寄存器总览

下面列出了 DBG 寄存器映射和复位值。这些外设寄存器必须作为字（32 位）操作。该寄存器的地址为 0xE0042000。

表 23-2 DBG 寄存器总览

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	DBG_ID	Reserved				DEV_NUM_L[3:0]				DEV_NUM_H[3:0]				Reserved				DEV_NUM_M[3:0]				REV_NUM_H[3:0]				REV_NUM_L[3:0]				Reserved				
Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
000h	DBG_CTRL	Reserved				DEV_NUM_L[3:0]				DEV_NUM_H[3:0]				Reserved				DEV_NUM_M[3:0]				REV_NUM_H[3:0]				REV_NUM_L[3:0]				Reserved				
004h	Reset Value																																	

### 23.4.2 ID 寄存器 (DBG\_ID)



地址: 0x04

只支持 32 位访问，固定值不能修改

位域	名称	描述
31:24	Reserved	保留，必须保持复位值。
23:20	DEV_NUM_L[3:0]	设备型号的低 4 位。 设备型号由高、中、低共 12 位组成，代表芯片的型号。
19:16	Reserved	保留，必须保持复位值。
15:12	DEV_NUM_H[3:0]	设备型号的高 4 位。 参考 DEV_NUM_L[3:0]的说明。
11:8	DEV_NUM_M[3:0]	设备型号的中 4 位。 参考 DEV_NUM_L[3:0]的说明。
7:4	REV_NUM_H[3:0]	芯片版本号高 4 位。
3:0	REV_NUM_L[3:0]	芯片版本号低 4 位。

### 23.4.3 调试控制寄存器 (DBG\_CTRL)

地址: 0x04

POR 默认值: 0x0000 0000 (系统复位不会重置)

31	Reserved										20	19	18	17	16
											TIM6_STOP	TIM5_STOP	TIM8_STOP	I2C2SM_BUS_TIMEOUT	
15 I2C1SM_BUS_TIMEOUT	14 CAN_STOP	13 TIM4_STOP	12 TIM3_STOP	11 TIM2_STOP	10 TIM1_STOP	9 WWDG_STOP	8 IWDG_STOP	7		Reserved	rw 3	rw 2	rw 1	rw 0	
rw	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	

位域	名称	描述
31:20	Reserved	保留，必须保持复位值。
19:17	TIMx_STOP	当内核停止时停止定时器计数器 (x=6,5,8)。 由软件置 1 或清零。 0: 内核停止时，时钟仍提供给相关定时器的计数器，定时器输出正常工作； 1: 当内核停止时，关闭相关定时器的计数器的时钟，同时关闭定时器的输出。
16:15	I2CxSMBUS_TIMEOUT	当内核停止时停止 SMBUS 超时模式 (x=2,1)。 由软件置 1 或清零。 0: 与正常模式操作相同； 1: 暂停 SMBUS 的超时控制。
14	CAN_STOP	当内核进入调试状态时，CAN 停止运行。 由软件置 1 或清零。 0: CAN 仍在正常运行； 1: CAN 的接收寄存器不继续接收数据
13:10	TIMx_STOP	当内核进入调试状态时，计数器停止工作 (x=4,3,2,1)。 由软件置 1 或清零。 0: 所选定时器的计数器仍正常工作； 1: 所选定时器的计数器停止工作。
9	WWDG_STOP	当内核进入调试状态时，调试窗口看门狗停止工作。 由软件置 1 或清零。 0: 窗口看门狗计数器仍正常工作； 1: 窗口看门狗计数器停止工作。
8	IWDG_STOP	当内核进入调试状态时，看门狗停止工作。 由软件置 1 或清零。 0: 门狗计数器仍然正常工作； 1: 看门狗计数器停止工作。
7:3	Reserved	保留，必须保持复位值。
2	STDBY	调试待机模式。 由软件置 1 或清零。 0: (FCLK 关、HCLK 关) 整个数字电路部分都断电。从软件的角度来看，退出 STANDBY 模式与复位相同（除了一些状态位指示微控制器刚从 STANDBY 状态退出）。 1: (FCLK 开、HCLK 开) 数字电路部分不掉电，FCLK 和 HCLK 时钟由内部 RLD 振荡器提供时钟。
1	STOP	调试停止模式。 由软件置 1 或清零。

位域	名称	描述
		<p>0: (FCLK 关、HCLK 关)在停止模式下，时钟控制器禁用所有时钟（包括 HCLK 和 FCLK）。退出 STOP 模式时，时钟的配置与复位后的配置相同（微控制器由 8MHz 内部 RC 振荡器（HSI）提供时钟）。因此，软件必须重新配置时钟控制系统来启动 PLL、晶振等。</p> <p>1:( FCLK 开、HCLK 开) 在停止模式下，FCLK 和 HCLK 时钟由内部 RC 振荡器提供。退出停止模式时，软件必须重新配置时钟系统以启动 PLL、晶振等（与配置该位为 0 时的操作相同）。</p>
0	SLEEP	<p>调试睡眠模式。 由软件置 1 或清零。</p> <p>0:(FCLK 开, HCLK 关) 在睡眠模式下，FCLK 由先前已配置好的系统时钟提供，而 HCLK 则关闭。由于睡眠模式不会重置已配置好的时钟系统，因此在退出睡眠模式时，无需重新配置时钟系统。</p> <p>1:(FCLK ON, HCLK ON) 在睡眠模式下，FCLK 和 HCLK 时钟均由先前配置好的系统时钟提供。</p>

## 24 唯一设备序列号 (UID)

### 24.1 简介

MCU 系列产品内置两个不同长度的唯一设备序列号，分别为 96 位的 UID(Unique device ID)和 128 位的 UCID(Unique Customer ID)，这两个设备序列号存放在闪存存储器的系统配置块中，它们所包含的信息在出厂时编写，并保证对任意一个 MCU 微控制器在任何情况下都是唯一的，用户应用程序或外部设备可以通过 CPU 或 JTAG/SWD 接口读取，不可被修改。

UID 为 96 位，通常用做为序列号或作为密码，在编写闪存时，将此唯一标识与软件加解密算法相结合，进一步提高代码在闪存存储器内的安全性。

UCID 为 128 位，遵守国民技术芯片序列号定义，它包含芯片生产及版本相关信息。

### 24.2 UID 寄存器

起始地址： 0x1FFF\_F7F0 长度 96 位。

### 24.3 UCID 寄存器

起始地址： 0x1FFF\_F7C0 长度 128 位。

## 25 版本历史

日期	版本	修改记录
2022.5.10	V1.0	初始版本
2022.8.22	V1.1	<ul style="list-style-type: none"> <li>1. L2 级别读保护: 添加 SRAM 启动被禁止, 选项字节写/页擦除被禁止</li> <li>2. 修改 9.2 章节, ADTIM 的主要特性</li> <li>3. 修改 10.2 章节, GPTIM 的主要特性</li> <li>4. 修改 13.3 章节, IWDG 框图和 IWDG 工作低功耗模式 SLEEP/STOP0/STOP2/STANDBY</li> <li>5. 修改 14.4 章节 APB 为 APB1 和 14.3 章节倒数计数器描述为递减计数器</li> <li>6. 修改 13.4.2 IWDG 初始化流程描述和 IWDG 章节重载值为重装在值/预分频器值为预分频值</li> <li>7. 修改 18.4.12.2IrDA 低功耗模式使能配置</li> <li>8. 修改 CAN_TSTS.LOWM0 位描述</li> <li>9. 主存储区 FLASH 名字由 Flash 统一为 Main Flash</li> <li>10. 表 2-6 更新, 删除系统配置区描述</li> <li>11. 更新 FLASH 部分 L2 等级描述</li> <li>12. 修改 11.3.1 章节描述</li> <li>13. 15.3.1 章节, 修改 ADC_1MCLK 描述</li> <li>14. 15.3.11.2 章节, 修改注入通道的间断模式描述</li> <li>15. 15.6 章节, 修改 ADC 转换时间计算。</li> <li>16. 表 18-4, 修改 fraction 单词为 decimal</li> <li>17. 15.9 章节, 修改温度计算公式</li> <li>18. 修改 DBG 章节 JNTRST 为 NJTRST</li> <li>19. 修改 RTC_CALIB 描述 CM8, CM16 为 CW8, CW16</li> <li>20. 修改 RTC 周期性可用唤醒时间范围为 2s ~ 36h</li> <li>21. 3.2.4 章节增加进入 STANDBY 模式时, PA13 和 PA14 引脚配置说明</li> <li>22. 修改 CRC32 计算时间</li> </ul>
2023.3.14	V1.2.0	<ul style="list-style-type: none"> <li>1. 修改表 2-3 Page31 地址</li> <li>2. 修改 7.4.6 章节 DMA 配置顺序</li> <li>3. 新增 5.2.8 章节 GPIO 使用注意事项</li> </ul>

		<ol style="list-style-type: none"> <li>4. 修改 DBG_CTRL 寄存器图</li> <li>5. 修改表 12-2</li> <li>6. 修改 LPTIM_CFG. TRGSEL[3:0]和 LPTIM_CFG. CLKPOL[1:0]描述</li> <li>7. 修改 ADC_JSEQ.JLEN[1:0]描述</li> <li>8. 修改 13.3.3 章节 IWDG 冻结描述</li> <li>9. 修改 COMPx_CTRL.OUTTRG[3:0]为 COMPx_CTRL.OUTSEL[3:0]</li> <li>10. 修改表 5-17 格式</li> <li>11. 修改表 2-6 L2 级别读保护修改权限，不允许修改读保护等级</li> <li>12. 3.2.4 章节添加 SRAM 保持可选</li> <li>13. 修改表 5-35，删除 TIM9 描述</li> <li>14. 删除 COMP3 的 TIM5_OCrefclear 信号</li> <li>15. 20.2.6 章节增加 RTC 进入初始化模式注意事项</li> <li>16. 修改 RTC_SCTRL. SUBF[14:0]寄存器描述</li> <li>17. 修改 COMP_LP 相关描述</li> <li>18. FLASH_AC. LATENCY 寄存器描述新增 D 版本芯片访问 Flash 时延说明</li> <li>19. 修改 AFIO_FILT_CFG. IOFLITCFG[4:0]寄存器描述</li> <li>20. 增加表 25-1 SDADFW/ SCLDFW 推荐配置</li> <li>21. 修改 I2C_TMRISE.TMRISE[5:0]寄存器描述</li> <li>22. 修改图 19-25 I2S 时钟发生器结构</li> <li>23. 修改 6.2.4 EXTI 线 18 连接到 RTC 时间戳事件、RTC 入侵事件和 LSECSS 事件</li> <li>24. 修改 5.2.5.4 章节，增加 5.2.5.5 章节，将 HSE 引脚和 LSE 引脚分开描述，当 LSE 配置成外部时钟输入模式时，OSC32_OUT 引脚不可作为普通 IO 使用</li> <li>25. 删除 COMPx_CTRL.PWRMODE 寄存器位描述</li> <li>26. 修改图 4-2，修改 ADC_CLK 为 ADC_SYNC_CLK</li> <li>27. 修改 3.2.4.2 章节退出 STANDBY 模式方式</li> <li>28. 修改 17.3.2.6 和 17.3.2.7 章节,增加从机地址配置注意</li> <li>29. 修改 20.2.19 章节,增加 RTC 唤醒 STANDBY 注意事项</li> </ol>
2023.5.30	V1.2.1	<ol style="list-style-type: none"> <li>1. 修改 20.2.6 和 20.2.7 章节，增加日历初始化和日历读取注意事项</li> <li>2. 修改 PWR_CTRL2. IWDGRSTEN 寄存器描述笔误</li> <li>3. 修改 16.2 章节内置三个 64 级可编程的比较电压参考源</li> <li>4. 修改 16.6 章节，增加注意事项</li> </ol>

		<p>5. 修改 4.3.1 章节，增加 PLL 作为系统时钟源时注意事项</p> <p>6. 修改 COMPx CTRL. HYST[1:0]寄存器描述</p>
--	--	---

## 26 声明

国民技术股份有限公司（下称“国民技术”）对此文档拥有专属产权。依据中华人民共和国的法律、条约以及世界其他法域相适用的管辖，此文档及其中描述的国民技术产品（下称“产品”）为公司所有。

国民技术在此并未授予专利权、著作权、商标权或其他任何知识产权许可。所提到或引用的第三方名称或品牌（如有）仅用作区别之目的。

国民技术保留随时变更、订正、增强、修改和改良此文档的权利，恕不另行通知。请使用人在下单购买前联系国民技术获取此文档的最新版本。

国民技术竭力提供准确可信的资讯，但即便如此，并不推定国民技术对此文档准确性和可靠性承担责任。

使用此文档信息以及生成产品时，使用者应当进行合理的设计、编程并测试其功能性和安全性，国民技术不对任何因使用此文档或本产品而产生的任何直接、间接、意外、特殊、惩罚性或衍生性损害结果承担责任。

国民技术对于产品在系统或设备中的应用效果没有任何故意或保证，如有任何应用在其发生操作不当或故障情况下，有可能致使人员伤亡、人身伤害或严重财产损失，则此类应用被视为“不安全使用”。

不安全使用包括但不限于：外科手术设备、原子能控制仪器、飞机或宇宙飞船仪器、所有类型的安全装置以及其他旨在支持或维持生命的应用。

所有不安全使用的风险应由使用人承担，同时使用人应使国民技术免于因为这类不安全使用而导致被诉、支付费用、发生损害或承担责任时的赔偿。

对于此文档和产品的任何明示、默示之保证，包括但不限于适销性、特定用途适用性和不侵权的保证责任，国民技术可在法律允许范围内进行免责。

未经明确许可，任何人不得以任何理由对此文档的全部或部分进行使用、复制、修改、抄录和传播。