专业电机芯片，内部集成 32-bit Cortex-M4 处理器

- 工作电压范围：2.5～5.5V
- 工作温度范围：-40℃～105℃
- 32-bit Cortex-M4 CPU 内核
  - 最高 80 MHz 工作频率
  - 单周期乘法和硬件除法
  - FPU
- 存储器
  - 256K Flash
  - 20 K SRAM
  - 40 K CCM
  - AMPU
- 时钟，复位和电源管理
  - 2.0 ～5.5 V 供电和 I/O 引脚
  - POR, PDR, PVD
  - 4-16 MHz 晶体振荡器
  - 内建出厂校准的 8MHz 的 RC 振荡器
  - 内建出厂校准的 40KHz 的 RC 振荡器
  - 产生 CPU 时钟的 PLL
- 低功耗
  - 睡眠，待机和停机模式
- 3×12 位 ADC, 1MSPS
  - 1×10 通道 , 1×7 通道, 1×10 通道
  - 通过可编程增益放大器放大输入信号
  - 转换范围: 0 ～AVCC
  - 温度传感器
- 2×DMA
  - 7 通道 DMA1, 8 通道 DMA3
  - 支持的外设： Timers, ADC, SPIs, I^2Cs ，USARTs
- 67 个快速 I/O 口
  - 53/67 I/Os，均可映像到 16 个外部中断
- 调试模式
  - 串行单线调试(SWD) 和 JTAG 接口
- 3×CMP
- 1×OPA
- 2×PGA
- 1×CRV
  - 双电平，32 个档位
  - 被用作比较器的参考电压源
- MEII 电机协同处理器
  - PID/CORDIC/SVPWM
- 9 个定时器
  - 3 个 16 位定时器，每个定时器有多达 4 个用于输入捕获/输出比较/PWM 或脉冲计数的通道和增量编码器输入
  - 3 个 16 位 6 通道带死区控制和紧急刹车，用于电机控制的 PWM 高级控制定时器。
  - 2 个看门狗定时器 (独立和窗口型)
  - 系统时间定时器：24 位自减型计数器
- 8 个通信接口
  - 2×I^2C 接口 (支持 SMBus/PMBus)
  - 3×USARTs
  - 2×SPIs
  - 1×CAN 接口(2.0B 主动)
- CRC 计算单元：96 位芯片唯一代码
- 采用绿色封装：
  LQFP64, LQFP80

版本历史

版本号	更改内容
V0.1	创建
V0.2	补全外设模块内容
V0.3	外设模块格式修正
V0.4	ADC 寄存器修正，封面中文化修正，新增封装图,CMP 寄存器更新，PGA 寄存器更正
V0.5	新增 ME 章节,新增 TIM20 及其描述,新增通用 TIMx 章节，修正页眉页脚
V0.6	运算放大器主要特征更新
V0.7	ADC 模块更新
V0.8	CMP 寄存器更新
V0.9	新增电源模块相应章节描述，新增备份寄存器 BKP 章节
V0.11	ADC 新增时序图章节
V0.12	中断部分新增寄存器内容
V0.13	增加高级定时器内部触发源列表

目录

1 概述 13
1.1 简介 13
1.2 框图 14
1.3 引脚排列 15
1.4 引脚定义 17
2 存储器模块 19
2.1 概述 19
2.2 存储器映射图 20
2.3 CCM 框图 21
2.4 特殊功能寄存器说明 22
2.4.1 闪存访问控制寄存器(FLASH_ACR) 22
2.4.2 FPEC 键寄存器(FLASH_KEYR) 23
2.4.3 闪存 OPTKEY 寄存器(FLASH_OPTKEYR) 24
2.4.4 闪存状态寄存器(FLASH_SR) 25
2.4.5 闪存控制寄存器(FLASH_CR) 26
2.4.6 闪存地址寄存器(FLASH_AR) 27
2.4.7 选择字节寄存器(FLASH_OBR) 28
2.4.8 写保护寄存器(FLASH_WRPR) 29
3 时钟单元 30
3.1 时钟框图 30
3.2 特殊功能寄存器说明 31
3.2.1 时钟控制寄存器(CR) 31
3.2.2 时钟配置寄存器(CFGR) 33
3.2.3 时钟中断寄存器(CIR) 36
3.2.4 APB2 外设复位寄存器 (APB2RSTR) 38
3.2.5 APB1 外设复位寄存器 (APB1RSTR) 40
3.2.6 AHB 外设时钟使能寄存器(AHBENR) 42
3.2.7 APB2 外设时钟使能寄存器(APB2ENR) 43
3.2.8 APB1 外设时钟使能寄存器(APB1ENR) 45
3.2.9 备份域控制寄存器 (BDCR) 47
3.2.10 控制/状态寄存器 (CSR) 48
3.2.11 AHB 外设复位寄存器（AHBRSTR） 49
4 电源单元 51
4.1 电源框图 51
4.2 电源管理器 51
4.2.1 上电复位(POR)和掉电复位(PDR) 51
4.2.2 可编程电压检测器（PVD） 52
4.3 低功耗模式 53
4.3.1 降低系统时钟 53
4.3.2 外部时钟的控制 53
4.3.3 睡眠模式 53
4.3.4 停止模式 54
4.3.5 待机模式 55
4.3.6 低功耗模式下的自动唤醒(AWU) 56
4.4 特殊功能寄存器说明 57
4.4.1 电源控制寄存器(PWR_CR) 57
4.4.2 电源控制/状态寄存器(PWR_CSR) 58
5 备份寄存器（BKP） 60
5.1 BKP 简介 60

5.2 BKP 特性 60
5.3 BKP 功能描述 60
5.3.1 侵入检测 60
5.3.2 RTC 校准 60
5.4 特殊功能寄存器说明 61
5.4.1 备份数据寄存器 x(BKP_DRx) (x = 1…10) 61
5.4.2 RTC 时钟校准寄存器(BKP_RTCCR) 62
5.4.3 备份控制寄存器(BKP_CR) 63
5.4.4 备份控制/状态寄存器(BKP_CSR) 64
6 GPIO 模块 65
6.1 概述 65
6.2 引脚排列 65
6.3 引脚定义 67
6.4 功能描述 69
6.4.1 GPIO 锁定机制 69
6.5 特殊功能寄存器说明 69
6.5.1 GPIO 复用功能低位寄存器 (GPIOx_AFRL)(x=A-E) 69
6.5.2 GPIO 复用功能高位寄存器 (GPIOx_AFRH) (x=A-E) 70
6.5.3 GPIO 端口上拉/下拉寄存器(GPIOx_PUPDR) (x=A-E) 71
6.5.4 GPIO 端口方向配置寄存器 (GPIOx_MODER)(x=A-E) 72
6.5.5 GPIO 端口输出类型配置寄存器 (GPIOx_OTYPER)(x=A-E) 73
6.5.6 GPIO 端口输出速度配置寄存器 (GPIOx_OSPEEDR)(x=A-E) 74
6.5.7 GPIO 端口输入数据寄存器 (GPIOx_IDR)(x=A-E) 75
6.5.8 GPIO 端口输出数据寄存器 (GPIOx_ODR)(x=A-E) 76
6.5.9 GPIO 端口位设置/清除寄存器 (GPIOx_BSRR)(x=A-E) 76
6.5.10 GPIO 端口配置锁定寄存器 (GPIOx_LCKR)(x=A-E) 77
7 中断模块 79
7.1 中断向量说明 79
7.2 EXTI 中断说明 81
7.3 特殊功能寄存器列表 81
7.4 特殊功能寄存器说明 81
7. 4. 1 中断屏蔽寄存器(EXTI_IMR) 81
7. 4. 2 事件屏蔽寄存器(EXTI_EMR) 81
7. 4. 3 上升沿触发选择寄存器(EXTI_RTSR) 82
7. 4. 4 下降沿触发选择寄存器(EXTI_FTSR) 82
7. 4. 5 软件中断事件寄存器(EXTI_SWIER) 83
7. 4. 6 挂起寄存器(EXTI_PR) 83
8 ADC 模块 84
8.1 概述 84
8.2 ADC 主要特征 84
8.2.1 ADC 模拟输入与通道号 84
8.3 ADC 功能描述 85
8.3.1 单个 ADC 框图 85
8.3.2 ADC 开关控制 87
8.3.3 ADC 时钟 87
8.3.4 单次转换模式 87
8.3.5 连续转换模式 87
8.3.6 扫描模式 88
8.3.7 外部触发转换 88
8.3.8 DMA 请求 89

8.3.9 双 ADC 模式 89
8.3.10 温度传感器 89
8.3.11 ADC 中断 89
8.3.12 时序图 90
8.4 双 ADC 模式 90
8.4.1 同步注入模式 92
8.4.2 同步规则模式 93
8.4.3 快速交叉模式 93
8.4.4 慢速交叉模式 94
8.4.5 交替触发模式 94
8.4.6 混合的规则/注入同步模式 95
8.4.7 混合的同步规则+交替触发模式 95
8.4.8 混合同步注入 + 交叉模式 96
8.5 特殊功能寄存器说明 98
8.5.1 ADC1/3 状态寄存器（ADC_SR） 98
8.5.2 ADC2 状态寄存器（ADC_SR） 99
8.5.3 ADC1/3 控制寄存器 1（ADC_CR1） 99
8.5.4 ADC2 控制寄存器 1（ADC_CR1） 102
8.5.5 ADC1/3 控制寄存器 2(ADC_CR2) 103
8.5.6 ADC2 控制寄存器 2（ADC_CR2） 106
8.5.7 ADC 采样时间寄存器 1(ADC_SMPR1) 108
8.5.8 ADC 注入通道数据偏移寄存器 x (ADC_JOFRx)(x=1..4) 109
8.5.9 ADC 看门狗高阀值寄存器(ADC_HTR) 110
8.5.10 ADC 看门狗低阀值寄存器(ADC_LRT) 110
8.5.11 ADC 规则序列寄存器 1(ADC_SQR1) 111
8.5.12 ADC 规则序列寄存器 2(ADC_SQR2) 112
8.5.13 ADC 规则序列寄存器 3(ADC_SQR3) 112
8.5.14 ADC 注入序列寄存器(ADC_JSQR) 113
8.5.15 ADC 注入数据寄存器 x (ADC_JDRx) (x= 1..4) 114
8.5.16 ADC 规则数据寄存器(ADC_DR) 115
8.5.17 ADC 校准寄存器(ADC_CAL) 115
9 SPI 串行外设接口 117
9.1 SPI 简介 117
9.2 SPI 特征 117
9.3 SPI 功能描述 117
9.3.1 概述 117
9.3.2 配置 SPI 为从模式 120
9.3.3 配置 SPI 为主模式 120
9.3.4 配置 SPI 为单工通信 121
9.3.5 数据发送与接收过程 121
9.3.6 CRC 计算 126
9.3.7 状态标志 127
9.3.8 关闭 SPI 127
9.3.9 错误标志 128
9.3.10 SPI 中断 128
9.4 特殊功能寄存器说明 129
9.4.1 SPI 控制寄存器 1(SPI_CR1) 129
9.4.2 SPI 控制寄存器 2(SPI_CR2) 131
9.4.3 SPI 状态寄存器(SPI_SR) 132
9.4.4 SPI 数据寄存器(SPI_DR) 133

9.4.5 SPI CRC 多项式寄存器(SPI_CRCPR) 134
9.4.6 SPI Rx CRC 寄存器(SPI_RXCRCR) 135
9.4.7 SPI Tx CRC 寄存器(SPI_TXCRCR) 135
10 通用 TIMx 定时器（TIM2 & TIM3 & TIM4） 137
10.1 TIM 概述 137
10.2 TIM2 和 TIM3 和 TIM4 主要特性 137
10.3 TIM 框图 137
10.4 TIM2 和 TIM3 和 TIM4 功能描述 138
10.4.1 时基单元 138
10.4.2 计数器模式 140
10.4.3 时钟选择 147
10.4.4 捕获/比较通道 149
10.4.5 输入捕获模式 151
10.4.6 PWM 输入模式 151
10.4.7 强置输出模式 152
10.4.8 输出比较模式 152
10.4.9 PWM 模式 153
10.4.10 单脉冲模式 155
10.4.11 在外部事件时清除 OCxREF 信号 156
10.4.12 编码器接口模式 157
10.4.13 定时器输入异或功能 159
10.4.14 TIMx 定时器和外部触发的同步 159
10.4.15 定时器同步 161
使用一个定时器作为另一个定时器的预分频器 161
使用一个定时器使能另一个定时器 162
使用一个定时器去启动另一个定时器 163
使用一个定时器作为另一个的预分频器 164
10.5 通用 TIMx 寄存器 165
10.5.1 控制寄存器 1(TIMx_CR1) 165
10.5.2 控制寄存器 2(TIMx_CR2) 167
10.5.3 从模式控制寄存器(TIMx_SMCR) 168
10.5.4 DMA/中断使能寄存器(TIMx_DIER) 170
10.5.5 状态寄存器(TIMx_SR) 172
10.5.6 事件产生寄存器(TIMx_EGR) 173
10.5.7 捕获/比较模式寄存器 1(TIMx_CCMR1) 174
10.5.8 捕获/比较模式寄存器 2(TIMx_CCMR2) 177
10.5.9 捕获/比较使能寄存器(TIMx_CCER) 179
10.5.10 计数器(TIMx_CNT) 181
10.5.11 预分频器(TIMx_PSC) 181
10.5.12 自动重装载寄存器(TIMx_ARR) 182
10.5.13 捕获/比较寄存器 1(TIMx_CCR1) 183
10.5.14 捕获/比较寄存器 2(TIMx_CCR2) 183
10.5.15 捕获/比较寄存器 3(TIMx_CCR3) 184
10.5.16 捕获/比较寄存器(TIMx_CCR4) 185
10.5.17 DMA 控制寄存器(TIMx_DCR) 186
10.5.18 连续模式的 DMA 地址(TIMx_DMAR) 187
11 高级定时器（TIM1 & TIM8 & TIM20） 188
11.1 TIM 概述 188
11.2 TIM1 和 TIM8 和 TIM20 主要特性 188
11.3 TIM 框图 188

11.4 TIM1 和 TIM8 和 TIM20 功能描述 189
11.4.1 时基单元 189
11.4.2 计数器模式 191
11.4.3 重复计数器 198
11.4.4 时钟选择 199
11.4.5 捕获/比较通道 201
11.4.6 输入捕获模式 203
11.4.7 PWM 输入模式 204
11.4.8 强置输出模式 205
11.4.9 输出比较模式 205
11.4.10 PWM 模式 206
11.4.11 非对称模式 208
11.4.12 互补输出和死区插入 212
11.4.13 使用刹车功能 213
11.4.14 在外部事件时清除 OCxREF 信号 215
11.4.15 产生六步 PWM 输出 216
11.4.16 单脉冲模式 217
11.4.17 编码器接口模式 218
11.4.18 定时器输入异或功能 220
11.4.19 与霍尔传感器的接口 220
11.4.20 TIMx 定时器和外部触发的同步 221
11.4.21 定时器同步 224
11.5 TIM1 和 TIM8 和 TIM20 寄存器 224
11.5.1 TIM1 和 TIM8 和 TIM20 控制寄存器 2(TIMx_CR2) 224
11.5.2 TIM1 和 TIM8 和 TIM20 DMA/中断使能寄存器(TIMx_DIER) 226
11.5.3 TIM1 和 TIM8 和 TIM20 状态寄存器(TIMx_SR) 227
11.5.4 TIM1 和 TIM8 和 TIM20 事件产生寄存器(TIMx_EGR) 229
11.5.5 TIM1 和 TIM8 和 TIM20 捕获/比较使能寄存器(TIMx_CCER) 230
11.5.6 TIM1 和 TIM8 和 TIM20 刹车和死区寄存器(TIMx_BDTR) 232
11.5.7 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 5 (TIMx_CCR5) 236
11.5.8 TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 (TIMx_CCMR3) 236
11.5.9 TIM1 和 TIM8 和 TIM20 控制寄存器 1(TIMx_CR1) 237
11.5.10 TIM1 和 TIM8 和 TIM20 从模式控制寄存器(TIMx_SMCR) 239
11.5.11 TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 1(TIMx_CCMR1) 241
11.5.12 TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 2(TIMx_CCMR2) 245
11.5.13 TIM1 和 TIM8 和 TIM20 计数器(TIMx_CNT) 247
11.5.14 TIM1 和 TIM8 和 TIM20 预分频器(TIMx_PSC) 248
11.5.15 TIM1 和 TIM8 和 TIM20 自动重装载寄存器(TIMx_ARR) 248
11.5.16 TIM1 和 TIM8 和 TIM20 重复计数寄存器(TIMx_RCR) 249
11.5.17 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1(TIMx_CCR1) 250
11.5.18 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 2(TIMx_CCR2) 251
11.5.19 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 3(TIMx_CCR3) 251
11.5.20 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器(TIMx_CCR4) 252
11.5.21 TIM1 和 TIM8 和 TIM20 DMA 控制寄存器(TIMx_DCR) 253
11.5.22 TIM1 和 TIM8 和 TIM20 连续模式的 DMA 地址(TIMx_DMAR) 254
11.5.23 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 6(TIMx_CCR6) 255
11.5.24 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1N (TIMx_CCR1N) 256
11.5.25 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 2N (TIMx_CCR2N) 257
11.5.26 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 3N (TIMx_CCR3N) 257
12 I2C 接口 259

12.1 I2C 简介 259
12.2 I2C 主要特点 259
12.3 I2C 功能描述 260
12.3.1 模式选择 260
12.3.2 I2C 从模式 262
12.3.3 I2C 主模式 264
12.3.4 错误条件 267
12.3.5 SDA/SCL 线控制 268
12.3.6 SMBus 269
12.4 DMA 请求 271
12.5 包错误校验(PEC) 272
12.6 I2C 中断请求 272
12.7 I2C 相关寄存器 274
12.7.1 控制寄存器 1(I2C_CR1) 274
12.7.2 控制寄存器 2(I2C_CR2) 276
12.7.3 自身地址寄存器 1(I2C_OAR1) 277
12.7.4 自身地址寄存器 2(I2C_OAR2) 278
12.7.5 数据寄存器(I2C_DR) 279
12.7.6 状态寄存器 1(I2C_SR1) 279
12.7.7 状态寄存器 2 (I2C_SR2) 283
12.7.8 时钟控制寄存器(I2C_CCR) 284
12.7.9 TRISE 寄存器(I2C_TRISE) 285
13 CAN 总线 287
13.1 bxCAN 简介 287
13.2 bxCAN 主要特点 287
13.3 bxCAN 总体描述 287
13.3.1 CAN 2.0B 主动内核 288
13.3.2 控制、状态和配置寄存器 288
13.3.3 发送邮箱 288
13.3.4 接收过滤器 288
13.3.5 bxCAN 工作模式 289
13.3.6 初始化模式 289
13.3.7 正常模式 289
13.3.8 睡眠模式(低功耗) 289
13.3.9 静默模式 291
13.3.10 环回模式 291
13.3.11 环回静默模式 293
13.4 bxCAN 功能描述 293
13.4.1 发送处理 293
13.4.2 时间触发通信模式 297
13.4.3 接收管理 297
13.4.4 标识符过滤 299
13.4.5 报文存储 304
13.4.6 出错管理 306
13.4.7 位时间特性 308
13.5 bxCAN 中断 310
13.6 CAN 控制寄存器说明 312
13.6.1 CAN 主控制寄存器 (CAN_MCR) 312
13.6.2 主状态寄存器 (CAN_MSR) 313
13.6.3 CAN 发送状态寄存器 (CAN_TSR) 314

13.6.4 CAN 接收 FIFO 0 寄存器（CAN_RF0R) 316
13.6.5 CAN 接收 FIFO 1 寄存器(CAN_RF1R) 317
13.6.6 CAN 中断使能寄存器（CAN_IER) 318
13.6.7 CAN 错误状态寄存器（CAN_ESR) 320
13.6.8 CAN 位时序寄存器（CAN_BTR) 321
13.7 CAN 邮箱寄存器说明 322
13.7.1 发送邮箱标识符寄存器（CAN_TIxR) (x=0..2) 322
13.7.2 发送邮箱数据长度和时间戳寄存器（CAN_TDTxR) (x=0..2) 323
13.7.3 发送邮箱低字节数据寄存器（CAN_TDLxR) (x=0..2) 324
13.7.4 发送邮箱高字节数据寄存器（CAN_TDHxR) (x=0..2) 325
13.7.5 接收 FIFO 邮箱标识符寄存器（CAN_RIxR) (x=0..1) 325
13.7.6 接收 FIFO 邮箱数据长度和时间戳寄存器（CAN_RDTxR) (x=0..1) 326
13.7.7 接收 FIFO 邮箱低字节数据寄存器（CAN_RDLxR) (x=0..1) 327
13.7.8 接收 FIFO 邮箱高字节数据寄存器（CAN_RDHxR) (x=0..1) 328
13.8 CAN 过滤器寄存器 329
13.8.1 CAN 过滤器主控寄存器（CAN_FMR) 329
13.8.2 CAN 过滤器模式寄存器（CAN_FM1R) 329
13.8.3 CAN 过滤器位宽寄存器（CAN_FS1R) 330
13.8.4 CAN 过滤器 FIFO 关联寄存器（CAN_FFA1R) 331
13.8.5 CAN 过滤器激活寄存器（CAN_FA1R) 331
13.8.6 CAN 过滤器组 i 的寄存器 x (CAN_FiRx) (i = 0..13； x=1..2) 332
14 通用同步异步收发器(USART) 334
14.1 USART 介绍 334
14.2 USART 主要特性 334
14.3 USART 功能概述 335
14.4 发送器 337
14.4.1 字符发送 337
14.4.2 可配置的停止位 338
14.4.3 单字节通信 339
14.4.4 断开符号 340
14.4.5 空闲符号 340
14.5 接收器 340
14.5.1 起始位侦测 341
14.5.2 字符接收 341
14.5.3 断开符号 342
14.5.4 空闲符号 342
14.5.5 溢出错误 342
14.5.6 噪音错误 342
14.5.7 帧错误 343
14.5.8 接收期间的可配置的停止位 344
14.6 USART 接收器容忍时钟的变化 344
14.7 多处理器通信 345
14.8 校验控制 346
14.9 LIN(局域互联网)模式 347
14.10 USART 同步模式 350
14.11 单线半双工通信 353
14.12 智能卡 353
14.13 USART 中断请求 355
14.14 USART 寄存器说明 356
14.14.1 状态寄存器(USART_SR) 356

14.14.2 数据寄存器(USART_DR) 358
14.14.3 波特比率寄存器(USART_BRR) 359
14.14.4 控制寄存器 1(USART_CR1) 360
14.14.5 控制寄存器 2(USART_CR2) 362
14.14.6 控制寄存器 3(USART_CR3) 363
14.14.7 保护时间和预分频寄存器(USART_GTPR) 364
15 CMP 比较器 367
15.1 概述 367
15.2 比较器框图 367
15.3 CRV 框图 367
15.4 比较器功能描述 368
15.4.1 比较器开关控制 368
15.4.2 比较器用法 368
15.4.3 CRV 相关概述 368
15.5 特殊功能寄存器说明 368
15.5.1 CMP 控制寄存器 1(CMPx_CR1) 368
15.5.2 CMP 控制寄存器 2(CMPx_CR2) 370
15.5.3 CMP 调节寄存器(CMPx_CAL) 372
15.5.4 CRV 控制寄存器（CRV_CR1） 372
15.5.5 CRV 通道 1 输出选择寄存器（CRV_CH1） 373
15.5.6 CRV 通道 2 输出选择寄存器（CRV_CH2） 374
15.5.7 CRV 通道 3 输出选择寄存器（CRV_CH3） 375
16 OPAMP 运算放大器 377
10.1 概述 377
10.2 OPAMP 框图 377
10.3 运算放大器主要特征 377
10.4 特殊功能寄存器说明 378
10.4.1 OPA 控制寄存器(OPA_CR) 378
10.4.2 OPA 控制寄存器 2(OPA_CR2) 379
10.4.3 OPA 控制寄存器 3(OPA_CR3) 380
17 AMPU 模块 382
17.1 AMPU 简介 382
17.2 AMPU 主要功能 382
17.3 AMPU 整体流程图 383
17.4 AMPU 认证功能 383
17.4.1 认证功能描述 383
17.4.2 认证流程 384
17.4.3 KEY 更新流程 384
17.4.4 KEY 认证上限功能 385
17.5 AMPU 数据区(Data) 385
17.5.1 数据区概述 385
17.5.2 Data 读取流程 386
17.5.3 Data Program 流程 387
17.6 四个独立可编程保护区域(Region0-3) 387
17.6.1 四个独立可编程保护区域概述 387
17.6.2 AMPU Region 功能使用推荐方案 388
17.6.3 Region Program 流程 388
17.6.4 Region 非 Program 流程 389
17.7 FLASH 配置 389
17.7.1 FLASH 配置概述 389

17.7.2 Program Fail(SR_PGF) 389
17.7.3 Autoload Fail(SR_ALF) 389
17.8 调试支持 390
17.8.1 调试支持概述 390
17.8.2 调试支持配置 390
17.9 AMPU 寄存器说明 390
17.9.1 AMPU 操作寄存器（AMPU_OP） 390
17.9.2 AMPU 状态寄存器（AMPU_SR） 391
17.9.3 AMPU 缓冲 ID 寄存器（AMPU_BUF_ID） 392
17.9.4 AMPU 缓冲数据寄存器（AMPU_BUF_Data） 392
17.9.5 AMPU 密钥数据寄存器（AMPU_KEY） 393
17.9.6 AMPU 控制寄存器（AMPU_CR） 394
17.9.7 AMPU 区域基地址低位寄存器（AMPU_ RBADRLx） 394
17.9.8 AMPU 区域基地址高位寄存器（AMPU_ RBADRHx） 395
17.9.9 AMPU 区域结束地址低位寄存器（AMPU_ READRLx） 396
17.9.10 AMPU 区域结束地址高位寄存器（AMPU_ READRHx） 396
17.9.11 AMPU 区域配置寄存器（AMPU_ RCSRx） 397
18 ME 电机专用协同处理器 399
18.1 概述 399
18.2 PID 闭环控制模块 400
18.2.1 概述 400
18.2.2 PID 使用方法 400
18.2.3 PID_CR（PID 控制寄存器） 401
18.2.4 PID_OUT（PID 输出数据寄存器） 401
18.2.5 PID_REF（PID 参考数据寄存器） 402
18.2.6 PID_FB（PID 反馈数据寄存器） 402
18.2.7 PID_ERR（PID 误差数据寄存器） 402
18.2.8 PID_INTG（PID 积分累积数据寄存器） 403
18.2.9 PID_KP(PID 比例放大增益/缩小寄存器) 403
18.2.10 PID_KI(PID 积分放大增益/缩小寄存器) 404
18.2.11 PID_KD(PID 微分放大增益/缩小寄存器) 404
18.2.12 PID_KIMULP（PID 积分末端放大增益寄存器） 405
18.2.13 PID_INTGLIM（PID 积分累积数据限制寄存器） 405
18.2.14 PID_OUTLIM（PID 输出限制寄存器） 406
18.2.15 PID_IER(PID 中断寄存器) 406
18.2.16 PID_IFR(PID 中断标志寄存器) 407
18.2.17 PID_SR(PID 状态寄存器) 407
18.3 角度变换（SIN COS Table） 408
18.3.1 使用方法 408
18.3.2 ME_Angle（ME 输入角度寄存器） 408
18.3.3 ME_Cos（ME 输出 Cos 数据寄存器） 408
18.3.4 ME_Sin（ME 输出 Sin 数据寄存器） 408
18.4 坐标转换 409
18.4.1 使用方法 409
注： 409
18.4.2 ME_Ia(自然坐标系 Ia 数据寄存器) 409
18.4.3 ME_Ib(自然坐标系 Ib 数据寄存器) 410
18.4.4 ME_Ialpha（静止坐标系 Ialpha 数据寄存器） 410
18.4.5 ME_Ibeta（静止坐标系 Ibeta 数据寄存器） 410
18.4.6 ME_Iq（旋转坐标系 Iq 数据寄存器） 411

18.4.7 ME_Id（旋转坐标系 Id 数据寄存器） 411
18.4.8 ME_Vq（旋转坐标系 Vq 数据寄存器） 411
18.4.9 ME_Vd（旋转坐标系 Vd 数据寄存器） 411
18.4.10 ME_Valpha（静止坐标系 Valpha 数据寄存器） 412
18.4.11 ME_Vbeta（静止坐标系 Vbeta 数据寄存器） 412
18.5 SVPWM 412
18.5.1 使用方法 412
18.5.2 ME_TPWM（PWM 通道占空比限制上限数据寄存器） 413
18.5.3 ME_LSMIN（PWM 通道占空比限制下限数据寄存器） 413
18.5.4 ME_PDCH1（PWM 通道 1 占空比数据寄存器） 413
18.5.5 ME_PDCH2（PWM 通道 2 占空比数据寄存器） 414
18.5.6 ME_PDCH3（PWM 通道 3 占空比数据寄存器） 414
18.5.7 ME_SVZone（PWM 扇区数据寄存器） 414
18.5.8 ME_Taon（Taon 数据寄存器） 414
18.5.9 ME_Tbon（Tbon 数据寄存器） 415
18.5.10 ME_Tcon（Tcon 数据寄存器） 415
18.5.11 ME_FOVM（PWM Full scale 数据寄存器） 415
18.6 ME 相关寄存器 416
18.6.1 ME_CR（ME 控制寄存器） 416
18.6.2 ME_IER（ME 中断使能寄存器） 418
18.6.3 ME_IFR（ME 中断标志寄存器） 419
18.6.4 ME_SR（ME 状态寄存器） 420
19 AC/DC（待后续测试） 421
19.1 Absolute maximum ratings 421
19.2 General operating conditions 422
19.3 Operating conditions PWR UP_DOW 422
19.4 Embedded reset and power 423
19.5 Embedded internal reference 423
19.6 Supply current 424
19.7 External clock 425
19.8 Internal clock source 426
19.9 PLL characteristics 427
19.10 Memory characteristics 427
19.11 EMC characteristics 428
19.12 electrical sensitivity 428
19.13 IO port chracteristics 429
19.14 NRST 431
19.15 TIM Timer 431
19.16 Communications interfaces 432
19.17 CAN 433
19.18 ADC 433
19.19 Temperature sensor 435
19.20 CMP 435
19.21 OPA/PGA 436
20 封装 439
LQFP80 439

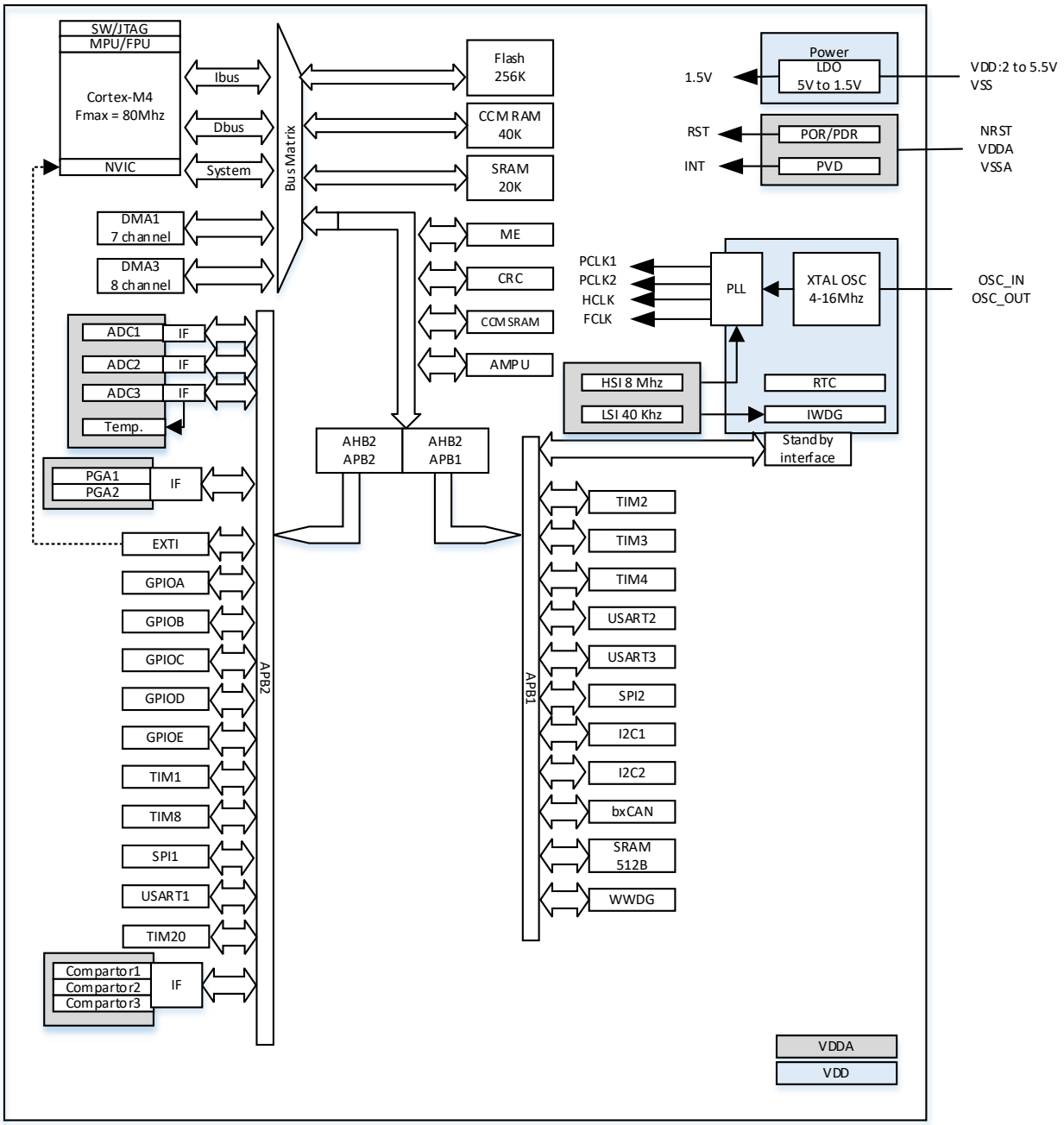
1 概述

1.1 简介

RX32S50 系列是高性能、低功耗、多功能马达专用 32 bit Cortex M4 MCU 芯片，工作温度范围 -40~105℃，工作电压 2.5-5.5V。芯片针对电机应用集成三个高级定时器、三个 12bit SAR ADC、可满足单电阻/双电阻采样的不同需求芯片整合了内部高频/低频 RC、可编程增益放大器、 比较器，在使用上能更节省布局空间，更节省成本。

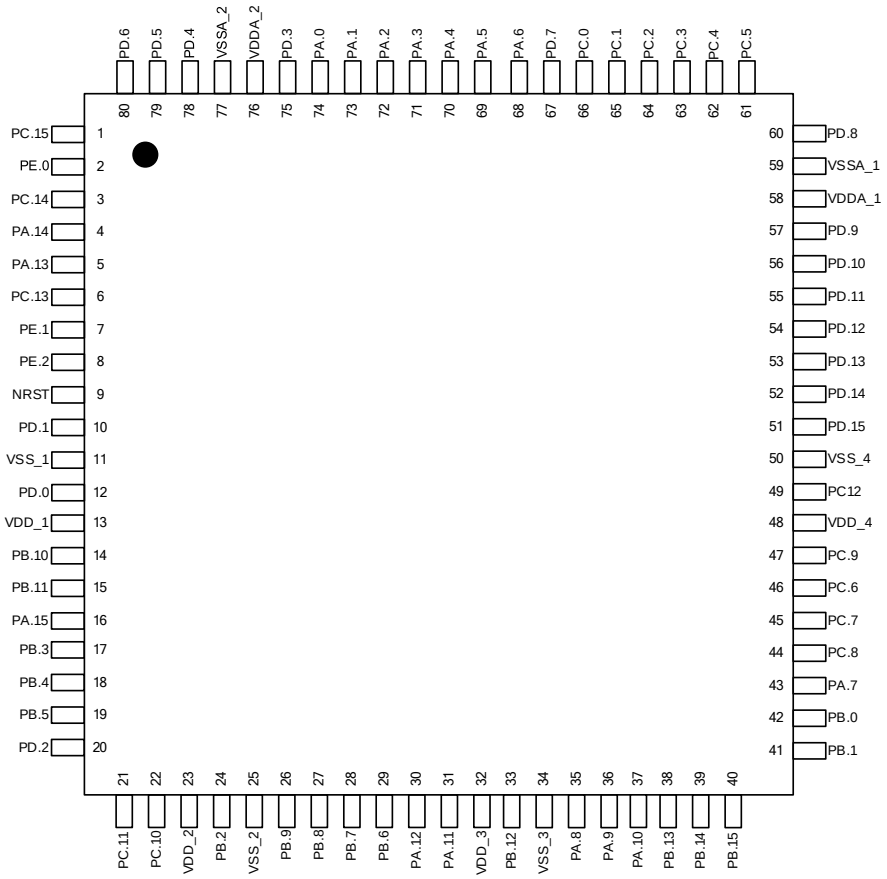
- 工作电压范围：2.5V～5.5V
- 工作温度范围：-40℃～105℃
- 采用 Cortex-M4 的处理器、256K Flash、20K SRAM、40K CCM
- 高速度：CPU 最高工作频率达到 80MHz
- 采用绿色封装：LQFP64；LQFP80；

1.2 框图

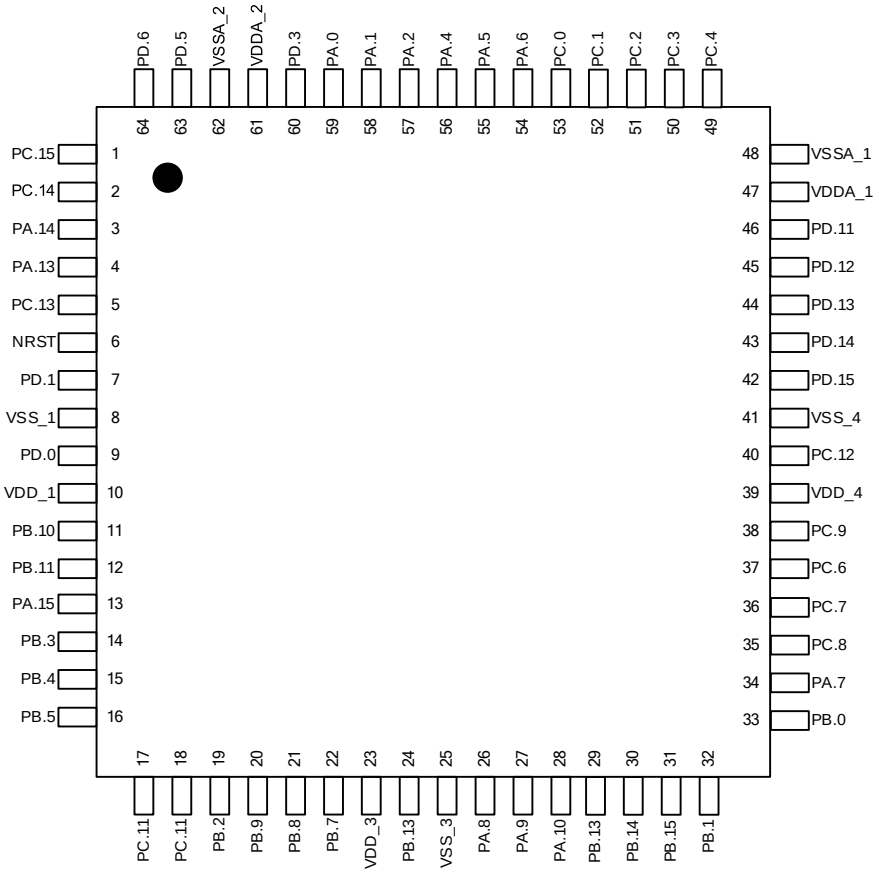

SW/JTAG
MPU/FPU
Cortex-M4
Fmax = 80Mhz
NVIC
Ibus
Dbus
System
Bus Matrix
Flash
256K
CCM RAM
40K
SRAM
20K
DMA1
7 channel
DMA3
8 channel
ME
CRC
CCM SRAM
AMPU
ADC1
ADC2
ADC3
Temp.
IF
PGA1
PGA2
EXTI
GPIOA
GPIOB
GPIOC
GPIOD
GPIOE
TIM1
TIM8
SPI1
USART1
TIM20
Comparator1
Comparator2
Comparator3
APB2
AHB2
APB2
AHB2
APB1
APB1
TIM2
TIM3
TIM4
USART2
USART3
SPI2
I2C1
I2C2
bxCAN
SRAM
512B
WWDG
Power
LDO
5V to 1.5V
1.5V
VDD:2 to 5.5V
VSS
RST
INT
POR/PDR
PVD
NRST
VDDA
VSSA
PCLK1
PCLK2
HCLK
FCLK
PLL
XTAL OSC
4-16Mhz
OSC_IN
OSC_OUT
HSI 8 Mhz
LSI 40 Khz
RTC
IWDG
Standby
interface
VDDA
VDD

1.3 引脚排列

LQFP80



LQFP64


PD.6 64
PD.5 63
VSSA_2 62
VDDA_2 61
PD.3 60
PA.0 59
PA.1 58
PA.2 57
PA.4 56
PA.5 55
PA.6 54
PC.0 53
PC.1 52
PC.2 51
PC.3 50
PC.4 49
PC.15 1
PC.14 2
PA.14 3
PA.13 4
PC.13 5
NRST 6
PD.1 7
VSS_1 8
PD.0 9
VDD_1 10
PB.10 11
PB.11 12
PA.15 13
PB.3 14
PB.4 15
PB.5 16
48 VSSA_1
47 VDDA_1
46 PD.11
45 PD.12
44 PD.13
43 PD.14
42 PD.15
41 VSS_4
40 PC.12
39 VDD_4
38 PC.9
37 PC.6
36 PC.7
35 PC.8
34 PA.7
33 PB.0
17 PC.11
18 PC.11
19 PB.2
20 PB.9
21 PB.8
22 PB.7
23 VDD_3
24 PB.13
25 VSS_3
26 PA.8
27 PA.9
28 PA.10
29 PB.13
30 PB.14
31 PB.15
32 PB.1

1.4 引脚定义

80 PIN	标识	引脚类型	Main Fuction (after reset)	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	Analog		
1	PC15	I/O	PC15									ADC1_IN6		
2	PE0	I/O	PE0			TIM20_CH4	USART2_CK					ADC1_IN7		
3	PC14	I/O	PC14/WKUP									ADC1_IN8		
4	PA14	I/O	JTCK/SWCLK	JTCK/SWCLK										
5	PA13	I/O	JTMS/SWDIO	JTMS/SWDIO										
6	PC13	I/O	PC13/TAMPER-RTC											
7	PE1	I/O	PE1			TIM20_CH5	USART2_TX							
8	PE2	I/O	PE2			TIM20_CH6	USART2_RX							
9	NRST	I												
10	PD1	I/O	OSC_OUT									OSC_OUT		
11	VSS_1	G												
12	PD0	I/O	OSC_IN									OSC_IN		
13	VDD_1	P												
14	PB10	I/O	PB10		TIM2_CH3		USART1_TX		I2C2_SCL					
15	PB11	I/O	PB11		TIM2_CH4		USART1_RX		I2C2_SDA					
16	PA15	I/O	JTDI	JTDI	TIM2_CH1_ETR			SPI1_NSS						
17	PB3	I/O	JTDO	JTDO/TRACESWO	TIM2_CH2		USART1_RX	SPI1_SCK						
18	PB4	I/O	JNTRST	JNTRST	TIM3_CH1			SPI1_MISO						
19	PB5	I/O	PB5		TIM3_CH2		USART1_TX	SPI1_MOSI	I2C1_SMBAI					
20	PD2	I/O	PD2		TIM3_ETR									
21	PC11	I/O	PC11		TIM3_CH4		USART3_RX							
22	PC10	I/O	PC10		TIM3_CH3		USART3_TX							
23	VDD_2	P												
24	PB2	I/O	PB2											
25	VSS_2	G												
26	PB9	I/O	PB9		TIM4_CH4				I2C1_SDA	CANTX				
27	PB8	I/O	PB8		TIM4_CH3				I2C1_SCL	CANRX				
28	PB7	I/O	PB7		TIM4_CH2		USART3_RX		I2C1_SDA					
29	PB6	I/O	PB6		TIM4_CH1		USART3_TX		I2C1_SCL					
30	PA12	I/O	PA12			TIM1_ETR	USART1_RTS			CANTX				
31	PA11	I/O	PA11				USART1_CTS			CANRX				
32	VDD_3	P												
33	PB12	I/O	PB12			TIM1_BKIN	USART1_CK	SPI2_NSS	I2C2_SMBAI	TIM8_ETR				
34	VSS_3	G												
35	PA8	I/O	PA8	MCO		TIM1_CH1								
36	PA9	I/O	PA9			TIM1_CH2								
37	PA10	I/O	PA10			TIM1_CH3								
38	PB13	I/O	PB13			TIM1_CH1N				TIM8_CH6				
39	PB14	I/O	PB14			TIM1_CH2N				TIM8_CH5				
40	PB15	I/O	PB15			TIM1_CH3N				TIM8_CH4				
41	PB1	I/O	PB1			TIM1_CH4				TIM8_CH3N				
42	PB0	I/O	PB0			TIM1_CH5				TIM8_CH2N				
43	PA7	I/O	PA7			TIM1_CH6				TIM8_CH1N				
44	PC8	I/O	PC8							TIM8_CH3				
45	PC7	I/O	PC7							TIM8_CH2				
46	PC6	I/O	PC6							TIM8_CH1				
47	PC9	I/O	PC9		TIM3_CH4	TIM8_BKIN				TIM8_CH4				
48	VDD_4	P												
49	PC12	I/O	PC12			TIM20_BKIN	USART3_CK							
50	VSS_4	G												
51	PD15	I/O	PD15		TIM4_CH4	TIM20_CH1		SPI2_NSS						
52	PD14	I/O	PD14		TIM4_CH3	TIM20_CH2	USART2_CK	SPI2_SCK						
53	PD13	I/O	PD13		TIM4_CH2	TIM20_CH3	USART2_TX	SPI2_MOSI						
54	PD12	I/O	PD12		TIM4_CH1	TIM20_CH1N	USART2_RX	SPI2_MISO						
55	PD11	I/O	PD11		TIM2_CH4	TIM20_CH2N	USART2_RTS							
56	PD10	I/O	PD10		TIM2_CH3	TIM20_CH3N	USART2_CTS							
57	PD9	I/O	PD9									ADC3_IN0		
58	VDDA_1	P												
59	VSSA_1	G												
60	PD8	I/O	PD8									ADC3_IN1		
61	PC5	I/O	PC5									ADC3_IN2		
62	PC4	I/O	PC4									ADC3_IN3		CMP3_N
63	PC3	I/O	PC3									ADC3_IN4		CMP3_P
64	PC2	I/O	PC2									ADC3_IN5	OPA_O	
65	PC1	I/O	PC1									ADC3_IN6	OPA_N	
66	PC0	I/O	PC0									ADC3_IN7	OPA_P	
67	PD7	I/O	PD7									ADC2_IN5		
68	PA6	I/O	PA6									ADC2_IN4		
69	PA5	I/O	PA5									ADC2_IN3	PGA2_N	CMP2_N
70	PA4	I/O	PA4									ADC2_IN2	PGA2_P	CMP2_P
71	PA3	I/O	PA3									ADC2_IN1		
72	PA2	I/O	PA2									ADC2_IN0		
73	PA1	I/O	PA1									ADC1_IN0	PGA1_N	CMP1_N
74	PA0	I/O	PA0									ADC1_IN1	PGA1_P	CMP1_P
75	PD3	I/O	PD3									ADC1_IN2		
76	VDDA_2	P												
77	VSSA_2	G												
78	PD4	I/O	PD4								CMP1_OUT	ADC1_IN3		
79	PD5	I/O	PD5								CMP2_OUT	ADC1_IN4		
80	PD6	I/O	PD6								CMP3_OUT	ADC1_IN5		

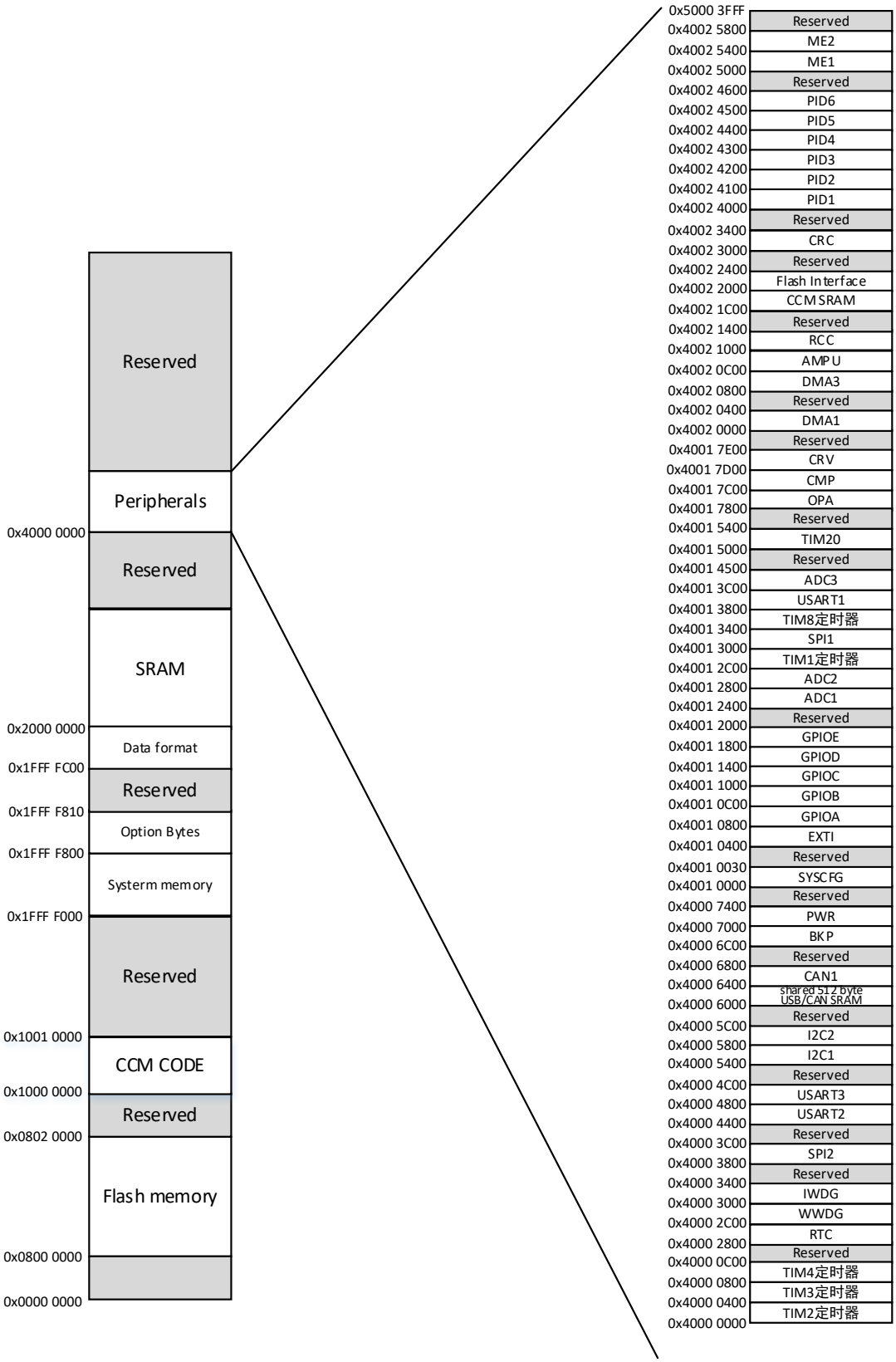
64 PIN	标识	引脚类型	Main Fuction (after reset)	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	Analog		
1	PC15	I/O	PC15									ADC1_IN6		
2	PC14	I/O	PC14/WKUP									ADC1_IN8		
3	PA14	I/O	JTCK/SWCLK	JTCK/SWCLK										
4	PA13	I/O	JTMS/SWDIO	JTMS/SWDIO										
5	PC13	I/O	PC13/TAMPER-RTC											
6	NRST	I												
7	PD1	I/O	OSC_OUT									OSC_OUT		
8	VSS_1	G												
9	PD0	I/O	OSC_IN									OSC_IN		
10	VDD_1	P												
11	PB10	I/O	PB10		TIM2_CH3		USART1_TX		I2C2_SCL					
12	PB11	I/O	PB11		TIM2_CH4		USART1_RX		I2C2_SDA					
13	PA15	I/O	JTDI	JTDI	TIM2_CH1_ETR			SPI1_NSS						
14	PB3	I/O	JTDO	JTDO/TRACESWO	TIM2_CH2		USART1_RX	SPI1_SCK						
15	PB4	I/O	JNTRST	JNTRST	TIM3_CH1			SPI1_MISO						
16	PB5	I/O	PB5		TIM3_CH2		USART1_TX	SPI1_MOSI	I2C1_SMBAI					
17	PC11	I/O	PC11		TIM3_CH4		USART3_RX							
18	PC10	I/O	PC10		TIM3_CH3		USART3_TX							
19	PB2	I/O	PB2											
20	PB9	I/O	PB9		TIM4_CH4				I2C1_SDA	CANTX				
21	PB8	I/O	PB8		TIM4_CH3				I2C1_SCL	CANRX				
22	PB7	I/O	PB7		TIM4_CH2		USART3_RX		I2C1_SDA					
23	VDD_3	P												
24	PB12	I/O	PB12			TIM1_BKIN	USART1_CK	SPI2_NSS	I2C2_SMBAI	TIM8_ETR				
25	VSS_3	G												
26	PA8	I/O	PA8	MCO		TIM1_CH1								
27	PA9	I/O	PA9			TIM1_CH2								
28	PA10	I/O	PA10			TIM1_CH3								
29	PB13	I/O	PB13			TIM1_CH1N				TIM8_CH6				
30	PB14	I/O	PB14			TIM1_CH2N				TIM8_CH5				
31	PB15	I/O	PB15			TIM1_CH3N				TIM8_CH4				
32	PB1	I/O	PB1			TIM1_CH4				TIM8_CH3N				
33	PB0	I/O	PB0			TIM1_CH5				TIM8_CH2N				
34	PA7	I/O	PA7			TIM1_CH6				TIM8_CH1N				
35	PC8	I/O	PC8							TIM8_CH3				
36	PC7	I/O	PC7							TIM8_CH2				
37	PC6	I/O	PC6							TIM8_CH1				
38	PC9	I/O	PC9		TIM3_CH4	TIM8_BKIN				TIM8_CH4				
39	VDD_4	P												
40	PC12	I/O	PC12			TIM20_BKIN	USART3_CK							
41	VSS_4	G												
42	PD15	I/O	PD15		TIM4_CH4	TIM20_CH1		SPI2_NSS						
43	PD14	I/O	PD14		TIM4_CH3	TIM20_CH2	USART2_CK	SPI2_SCK						
44	PD13	I/O	PD13		TIM4_CH2	TIM20_CH3	USART2_TX	SPI2_MOSI						
45	PD12	I/O	PD12		TIM4_CH1	TIM20_CH1N	USART2_RX	SPI2_MISO						
46	PD11	I/O	PD11		TIM2_CH4	TIM20_CH2N	USART2_RTS							
47	VDDA_1	P												
48	VSSA_1	G												
49	PC4	I/O	PC4									ADC3_IN3		CMP3_N
50	PC3	I/O	PC3									ADC3_IN4		CMP3_P
51	PC2	I/O	PC2									ADC3_IN5	OPA_O	
52	PC1	I/O	PC1									ADC3_IN6	OPA_N	
53	PC0	I/O	PC0									ADC3_IN7	OPA_P	
54	PA6	I/O	PA6									ADC2_IN4		
55	PA5	I/O	PA5									ADC2_IN3	PGA2_N	CMP2_N
56	PA4	I/O	PA4									ADC2_IN2	PGA2_P	CMP2_P
57	PA2	I/O	PA2									ADC2_IN0		
58	PA1	I/O	PA1									ADC1_IN0	PGA1_N	CMP1_N
59	PA0	I/O	PA0									ADC1_IN1	PGA1_P	CMP1_P
60	PD3	I/O	PD3									ADC1_IN2		
61	VDDA_2	P												
62	VSSA_2	G												
63	PD5	I/O	PD5								CMP2_OUT	ADC1_IN4		
64	PD6	I/O	PD6								CMP3_OUT	ADC1_IN5		

2 存储器模块

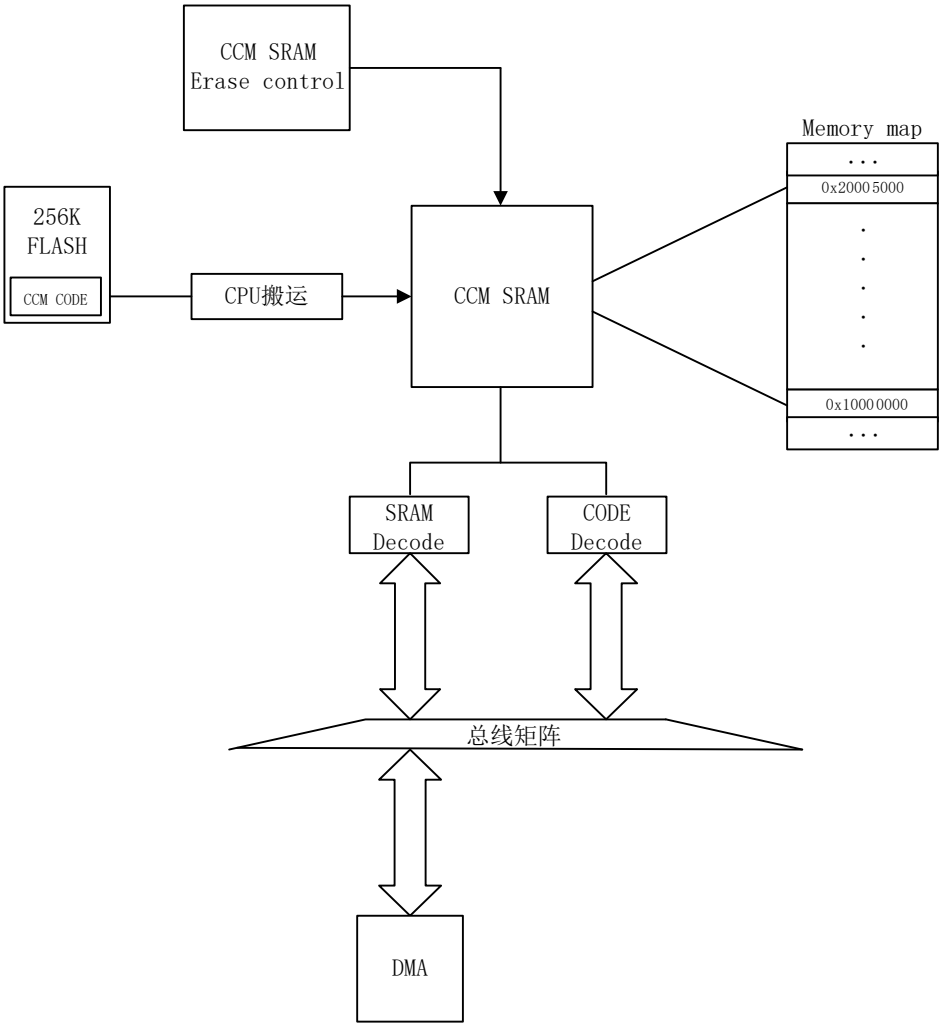
2.1 概述

RX32S50 系列内置可编程高可靠 256K Flash、20KSRAM 和 40KCCM。其中 Flash 可进行读、写、页擦
除和全擦除操作。程序放在 CCM SRAM 区域，能够提高代码运算效率，CCM SRAM 区域可实现 DMA 应用，搬运数据。Code 在 CCM SRAM 区域（0x10000000）可以通过软件擦除，掉电不能擦除。Data 在 CCM SRAM 区域（0x20005000）可以通过软件擦除或电擦除。注：0x10000000 与 0x20005000 地址相互映射。

2.2 存储器映射图


Reserved
Peripherals
0x4000 0000
Reserved
SRAM
0x2000 0000
Data format
0x1FFF FC00
Reserved
0x1FFF F810
Option Bytes
0x1FFF F800
Systerm memory
0x1FFF F000
Reserved
0x1001 0000
CCM CODE
0x1000 0000
Reserved
0x0802 0000
Flash memory
0x0800 0000
0x0000 0000
0x5000 3FFF
0x4002 5800 Reserved
0x4002 5400 ME2
0x4002 5000 ME1
0x4002 4600 Reserved
0x4002 4500 PID6
0x4002 4400 PID5
0x4002 4300 PID4
0x4002 4200 PID3
0x4002 4100 PID2
0x4002 4000 PID1
0x4002 3400 Reserved
0x4002 3000 CRC
0x4002 2400 Reserved
0x4002 2000 Flash Interface
0x4002 1C00 CCM SRAM
0x4002 1400 Reserved
0x4002 1000 RCC
0x4002 0C00 AMPU
0x4002 0800 DMA3
0x4002 0400 Reserved
0x4002 0000 DMA1
0x4001 7E00 Reserved
0x4001 7D00 CRV
0x4001 7C00 CMP
0x4001 7800 OPA
0x4001 5400 Reserved
0x4001 5000 TIM20
0x4001 4500 Reserved
0x4001 3C00 ADC3
0x4001 3800 USART1
0x4001 3400 TIM8定时器
0x4001 3000 SPI1
0x4001 2C00 TIM1定时器
0x4001 2800 ADC2
0x4001 2400 ADC1
0x4001 2000 Reserved
0x4001 1800 GPIOE
0x4001 1400 GPIOD
0x4001 1000 GPIOC
0x4001 0C00 GPIOB
0x4001 0800 GPIOA
0x4001 0400 EXTI
0x4001 0030 Reserved
0x4001 0000 SYSCFG
0x4000 7400 Reserved
0x4000 7000 PWR
0x4000 6C00 BKP
0x4000 6800 Reserved
0x4000 6400 CAN1
0x4000 6000 shared 512 byte USB/CAN SRAM
0x4000 5C00 Reserved
0x4000 5800 I2C2
0x4000 5400 I2C1
0x4000 4C00 Reserved
0x4000 4800 USART3
0x4000 4400 USART2
0x4000 3C00 Reserved
0x4000 3800 SPI2
0x4000 3400 Reserved
0x4000 3000 IWDG
0x4000 2C00 WWDG
0x4000 2800 RTC
0x4000 0C00 Reserved
0x4000 0800 TIM4定时器
0x4000 0400 TIM3定时器
0x4000 0000 TIM2定时器

2.3 CCM 框图



CCM 操作流程：

- 函数定义在 CCM 区域
- compiler 完成,CPU 在 FLASH 上分配地址空间给放在 CCM 区域的 code
- 在 startup 时,CPU 从 FLASH 往 CCM 区域搬运

2.4 特殊功能寄存器说明

2.4.1 闪存访问控制寄存器(FLASH_ACR)

闪存访问控制寄存器 (FLASH_ACR)			基地址： 0x4002 2000 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	PRFTBS X	PRFTBE	HLFCYA	LATENCY		
Reset:	0	0	1	1	0	0	0	0

位	功能描述
PRFTBS	**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。
PRFTBE	**预取缓冲区使能** 0：关闭预取缓冲区； 1：启用预取缓冲区。
HLFCYA	**闪存半周期访问使能** 0：禁止半周期访问； 1：启用半周期访问。
LATENCY[2:0]	**时间延迟** 这些位表示SYSCLK(系统时钟)周期与闪存访问时间的比例 000：零等待状态，当 0 < SYSCLK ≤ 24MHz 001：一个等待状态，当 24MHz < SYSCLK ≤ 48MHz 010：两个等待状态，当 48MHz < SYSCLK ≤ 72MHz

2.4.2 FPEC 键寄存器(FLASH_KEYR)

FPEC 键寄存器(FLASH_KEYR)			基地址： 0x4002 2000 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	FKEYR[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	FKEYR[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	FKEYR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FKEYR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FKEYR[31:0]	FPEC 键 这些位用于输入 FPEC 的解锁键。

注：所有这些位是只写的，读出时返回 0。

2.4.3 闪存 OPTKEY 寄存器(FLASH_OPTKEYR)

闪存 OPTKEY 寄存器 (FLASH_OPTKEYR)			基地址： 0x4002 2000 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	OPTKEYR[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	OPTKEYR[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OPTKEYR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OPTKEYR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OPTKEYR[31:0]	**选择字节键** 这些位用于输入选择字节的键以解除 OPTWRE。

注：所有这些位是只写的，读出时返回 0。

2.4.4 闪存状态寄存器(FLASH_SR)

闪存状态寄存器(FLASH_SR)			基地址： 0x4002 2000 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	EOP	WRPRTERR	X	PGERR	X	BSY
Reset:	0	0	0	0	0	0	0	0

位	功能描述
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。
WRPRTERR	**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。
PGERR	**编程错误** 试图对内容不是’0xFFFF’的地址编程时，硬件设置这位为’1’，写入’1’可以清除这位状态。 注：进行编程操作之前，必须先清除 FLASH_CR 寄存器的 STRT 位。
BSY	**操作进行位** 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’。

2.4.5 闪存控制寄存器(FLASH_CR)

闪存控制寄存器(FLASH_CR)			基地址： 0x4002 2000 偏移地址： 0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	CCMSRAM_RST	CCMSRAM_PE
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	EOPIE	X	ERRIE	OPTWRE	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	LOCK	STRT	OPTER	OPTPG	X	MER	PER	PG
Reset:	1	0	0	0	0	0	0	0

位	功能描述
CCMSRAM_RST	**CCM SRAM erase when system reset** 0: CCM SRAM erased when a system reset occurs 1: CCM SRAM is not erased when a system reset occurs This bit is loaded with values from Flash memory at OBL
CCMSRAM_PE	**CCM SRAM parity check enable** 0: CCM SRAM parity check enable 1: CCM SRAM parity check disenable This bit is loaded with values from Flash memory at OBL
EOPIE	**允许操作完成中断** 该位允许在FLASH_SR寄存器中的EOP位变为’1’时产生中断。 0：禁止产生中断； 1：允许产生中断。
ERRIE	**允许错误状态中断** 该位允许在发生FPEC错误时产生中断(当FLASH_SR寄存器中的PGERR/WRPRTERR置为’1’时)。 0：禁止产生中断； 1：允许产生中断。
OPTWRE	**允许写选择字节** 当该位为’1’时，允许对选择字节进行编程操作。当在FLASH_OPTKEYR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。
LOCK	**锁存位** 只能写’1’。 当该位为’1’时表示FPEC和FLASH_CR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。

STRT	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 BSY 变为’1’时清为’0’。
OPTER	**擦除选择字节**
OPTPG	**烧写选择字节** 对选择字节编程。
MER	**全擦除** 选择擦除所有用户页。
PER	**页擦除** 选择擦除页。
PG	**编程** 选择编程操作。

2.4.6 闪存地址寄存器(FLASH_AR)

闪存地址寄存器(FLASH_AR)			基地址： 0x4002 2000 偏移地址： 0x14					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	FAR[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	FAR[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	FAR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	FAR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FAR	**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_SR 中的 BSY 位为’1’时，不能写这个寄存器。

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

2.4.7 选择字节寄存器(FLASH_OBR)

选择字节寄存器 (FLASH_OBR)			基地址： 0x4002 2000 偏移地址： 0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	1	1	1
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	1	1	1	1	1	1	1	1
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	USER[9:8]	
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	USER[7:2]						RDPRT	OPTERR
Reset:	1	1	1	1	1	1	0	0

位	功能描述
USER[9:2]	**用户选择字节** 这里包含OBL加载的用户选择字节位 [9:5]：未用 位4：nRST_STDBY 位3：nRST_STOP 位 2：WDG_SW
RDPRT	**读保护** 当设置为1，表示闪存存储器的读保护有效。 *注：该位为只读。*
OPTERR	**选择字节错误** 当该位为’1’时表示选择字节和它的反码不匹配。 *注意：该位为只读。*

2.4.8 写保护寄存器(FLASH_WRPR)

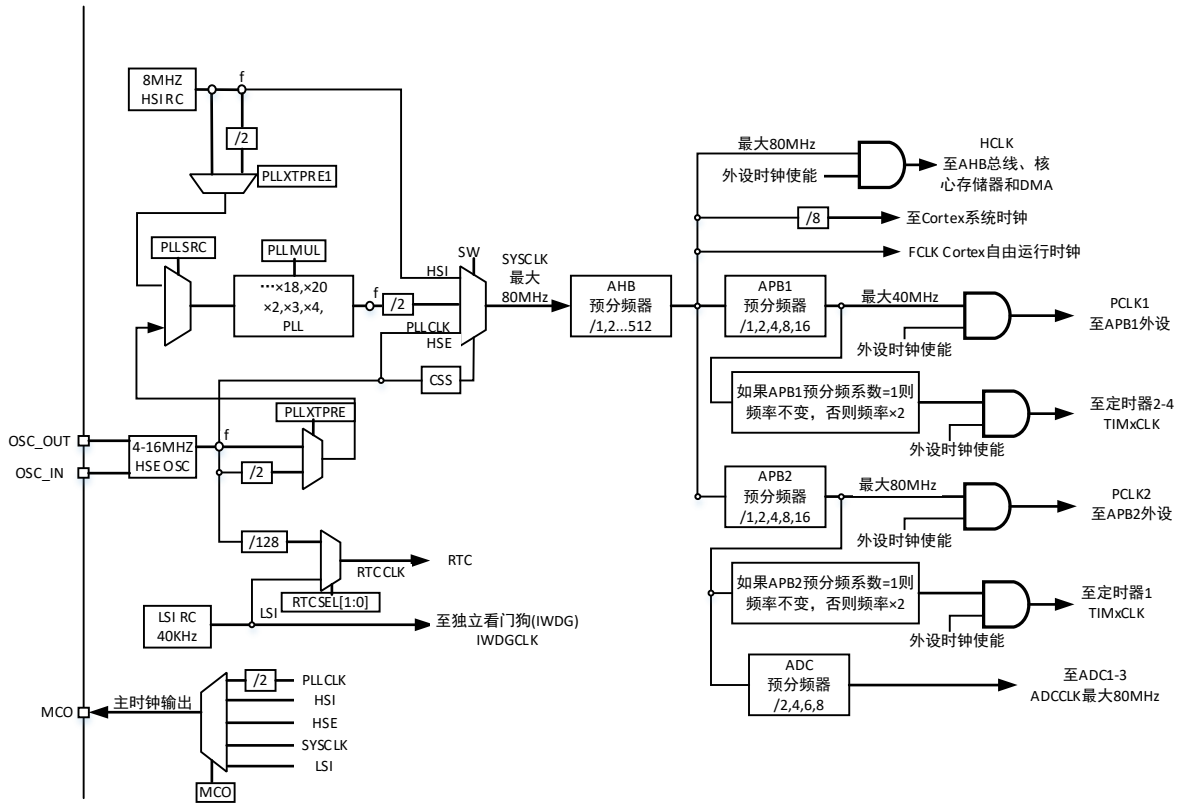
写保护寄存器(FLASH_WRPR)			基地址： 0x4002 2000 偏移地址： 0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	WRP[31:24]							
Reset:	1	1	1	1	1	1	1	1
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	WRP[23:16]							
Reset:	1	1	1	1	1	1	1	1
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	WRP[15:8]							
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	WRP[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述
WRP[31:0]	**写保护** 该寄存器包含由 OBL 加载的写保护选择字节。 0：写保护生效 1：写保护失效 注意：这些位为只读。

3 时钟单元

3.1 时钟框图

时钟符号说明：
HSI：高速内部 RC 时钟（8MHz）
HSE：高速外部 OSC 时钟（4MHz~16MHz）
LSI：低速内部 RC 时钟（40kHz）
PLL：锁相环倍频输出，其时钟输入源可选择为 HSI、HSE 或者 HSE/2。倍频可选择为 2~20 倍，输出频率最大不得超过 160MHz(PLL 输出后 2 分频得 SYSCLKmax=80MHz)。



3.2 特殊功能寄存器说明

3.2.1 时钟控制寄存器(CR)

时钟控制寄存器(CR)			基地址： 0x4002 1000 偏移地址： 0x00					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	HSE_CTR[31:30]		HSI_CTR[29:28]		PLL_CTR[27:26]		PLLRDY X	PLLON
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	HSEBIAS	CSSON	HSEBYP	HSERDY X	HSEON
Reset:	0	0	0	1	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read:	HSICAL[15:8]							
Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	HSITRIM[7:3]					X	HSIRDY X	HSION
Reset:	1	0	0	0	0	0	1	1

位	功能描述
HSE_CTR[31:30]	HSE时钟滤波控制位，CMUOPT=1时该寄存器有效 =00，0.0ns（Default） =01，1.5ns =10，3.0ns =11，4.5ns
HSI_CTR[29:28]	HRC时钟滤波控制位，CMUOPT=1时该寄存器有效 =00，0.0ns（Default） =01，1.5ns =10，3.0ns =11，4.5ns
PLL_CTR[27:26]	PLL时钟滤波控制位,CMUOPT=1时该寄存器有效 =00，0.0ns（Default） =01，1.5ns =10，1.5ns =11，1.5ns
PLLRDY	PLL时钟就绪标志位(PLLclockreadyflag) PLL锁定后由硬件置’1’。 0：PLL未锁定； 1：PLL锁定。

PLLON	PLL使能（PLL enable） 由软件置’1’或清零。 当进入待机和停止模式时，该位由硬件清零。当PLL时钟被用作或被选择将要作为系统时钟时，该位不能被清零。 0：PLL关闭； 1：PLL使能。
HSEBIAS	HSEBIAS控制位 1：HSE BIAS 電阻連接（default） 0：HSE BIAS 電阻斷開
CSSON	时钟安全系统使能（Clock security system enable） 由软件置’1’或清零以使能时钟监测器。 0：时钟监测器关闭； 1：如果外部4-16MHz振荡器就绪，时钟监测器开启。
HSEBYP	外部高速时钟旁路（External high-speed clock bypass） 在调试模式下由软件置’1’或清零来旁路外部晶体振荡器。只有在外部4-16MHz振荡器关闭的情 况下，才能写入该位。 0：外部4-16MHz振荡器没有旁路； 1：外部4-16MHz外部晶体振荡器被旁路。
HSERDY	外部高速时钟就绪标志（External high-speed clock ready flag） 由硬件置’1’来指示外部4-16MHz振荡器已经稳定。在HSEON位清零后，该位需要6个外部4- 25MHz振荡器周期清零。 0：外部4-16MHz振荡器没有就绪； 1：外部4-16MHz振荡器就绪。
HSEON	外部高速时钟使能（External high-speed clock enable） 由软件置’1’或清零。 当进入待机和停止模式时，该位由硬件清零，关闭4-16MHz外部振荡器。当外部4-16MHz振荡 器被用作或被选择将要作为系统时钟时，该位不能被清零。 0：HSE振荡器关闭； 1：HSE振荡器开启。
HSICAL[15:8]	内部高速时钟校准（Internal high-speed clock calibration） 在系统启动时，这些位被自动初始化
HSITRIM[7:3]	内部高速时钟调整（Internal high-speed clock trimming） 由软件写入来调整内部高速时钟，它们被叠加在HSICAL数值上。 这些位在HSICAL的基础上，让用户可以输入一个调整数值，根据电压和温度的变化调整 内部HSI RC振荡器的频率。 默认数值为16，可以把HSI调整到8MHz±1%；每步HSICAL的变化调整约40kHz。
HSIRDY	内部高速时钟就绪标志（Internal high-speed clock ready flag） 由硬件置’1’来指示内部8MHz振荡器已经稳定。在HSION位清零后，该位需要6个内部8MHz振 荡器周期清零。 0：内部8MHz振荡器没有就绪； 1：内部8MHz振荡器就绪。

HSION	**内部高速时钟使能（Internal high-speed clock enable）** 由软件置’1’或清零。 当从待机和停止模式返回或用作系统时钟的外部4-16MHz振荡器发生故障时，该位由硬件置’1’ 来启动内部8MHz的RC振荡器。当内部8MHz振荡器被直接或间接地用作或被选择将要作为系 统时钟时，该位不能被清零。 0：内部8MHz振荡器关闭； 1：内部8MHz振荡器开启。

注：访问：无等待，字，半字和字节访问

3.2.2 时钟配置寄存器(CFGR)

时钟配置寄存器（CFGR）			基地址：0x4002 1000 偏移地址：0x04					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	PLLXTPRE1	X	X	X	X	MCO[26:24]		
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	PLLMUL[21:18]				PLL XTPRE	PLLSRC
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	ADCPRE[15:14]		PPRE2[13:11]			PPRE1[10:8]		
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	HPRE[7:4]				SWS[3:2]		SW[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PLLXTPRE1	**HSI 分频器作为 PLL 输入（HSI divider for PLL entry）** **由软件置‘1’或清‘0’来分频 HSE 后作为 PLL 输入时钟。只能在关闭 PLL 时才能写入此为** 0：HSI 2 分频 1：HIS 不分频
MCO[26:24]	**微控制器时钟输出(Microcontroller clock output)** 由软件置’1’或清零。 000：没有时钟输出； 001：没有时钟输出； 010：LSI时钟输出； 011：没有时钟输出； 100：系统时钟(SYSCLK)输出； 101：内部RC振荡器时钟(HSI)输出；

	110：外部振荡器时钟(HSE)输出； 111：PLL时钟2分频后输出。 注意：- 该时钟输出在启动和切换MCO时钟源时可能会被截断。 - 在系统时钟作为输出至 MCO 引脚时，请保证输出时钟频率不超过 50MHz (I/O 口最高频率)。
PLLMUL[21:18]	**PLL倍频系数(PLL multiplication factor)** 由软件设置来确定PLL倍频系数。只有在PLL关闭的情况下才可被写入。 注意：倍频受PLLMUL[4:0]控制，PLL的输出频率不能超过72MHz 0000：PLL 2倍频输出　　1000：PLL 10倍频输出 0001：PLL 3倍频输出　　1001：PLL 11倍频输出 0010：PLL 4倍频输出　　1010：PLL 12倍频输出 0011：PLL 5倍频输出　　1011：PLL 13倍频输出 0100：PLL 6倍频输出　　1100：PLL 14倍频输出 0101：PLL 7倍频输出　　1101：PLL 15倍频输出 0110：PLL 8倍频输出　　1110：PLL 16倍频输出 0111：PLL 9倍频输出　　1111：PLL 16倍频输出
PLLXTPRE	**HSE分频器作为PLL输入(HSE divider for PLL entry)** 由软件置’1’或清’0’来分频HSE后作为PLL输入时钟。只能在关闭PLL时才能写入此位。 0：HSE不分频 1：HSE 2分频
PLLSRC	**PLL输入时钟源(PLL entry clock source)** 由软件置’1’或清’0’来选择PLL输入时钟源。只能在关闭PLL时才能写入此位。 0：HSI振荡器时钟经2分频后作为PLL输入时钟 1：HSE 时钟作为 PLL 输入时钟。
ADCPRE[15:14]	**ADC预分频(ADC prescaler)** 由软件置’1’或清’0’来确定ADC时钟频率 00：PCLK2 2分频后作为ADC时钟 01：PCLK2 4分频后作为ADC时钟 10：PCLK2 6分频后作为ADC时钟 11：PCLK2 8 分频后作为 ADC 时钟
PPRE2[13:11]	**高速APB预分频(APB2) (APB high-speed prescaler (APB2))** 由软件置’1’或清’0’来控制高速APB2时钟(PCLK2)的预分频系数。 0xx：HCLK不分频 100：HCLK 2分频 101：HCLK 4分频 110：HCLK 8分频 111：HCLK 16 分频

PPRE1[10:8]	**低速APB预分频(APB1) (APB low-speed prescaler (APB1))** 由软件置’1’或清’0’来控制低速APB1时钟(PCLK1)的预分频系数。 警告：软件必须保证APB1时钟频率不超过36MHz。 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
HPRE[7:4]	**AHB预分频(AHB Prescaler)** 由软件置’1’或清’0’来控制AHB时钟的预分频系数。 0xxx：SYSCLK不分频 1000：SYSCLK 2分频 1100：SYSCLK 64分频 1001：SYSCLK 4分频 1101：SYSCLK 128分频 1010：SYSCLK 8分频 1110：SYSCLK 256分频 1011：SYSCLK 16分频 1111：SYSCLK 512分频 注意：当 AHB 时钟的预分频系数大于 1 时，必须开启预取缓冲器。
SWS[3:2]	**系统时钟切换状态(System clock switch status)** 由硬件置’1’或清’0’来指示哪一个时钟源被作为系统时钟。 00：HSI作为系统时钟； 01：HSE作为系统时钟； 10：PLL输出作为系统时钟； 11：LSI 作为系统时钟。
SW[1:0]	**系统时钟切换(System clock switch)** 由软件置’1’或清’0’来选择系统时钟源。 在从停止或待机模式中返回时或直接或间接作为系统时钟的HSE出现故障时，由硬件强制选择 HSI作为系统时钟(如果时钟安全系统已经启动) 00：HSI作为系统时钟； 01：HSE作为系统时钟； 10：PLL输出作为系统时钟； 11：LSI 作为系统时钟。

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

3.2.3 时钟中断寄存器(CIR)

时钟中断寄存器(CIR)			基地址： 0x4002 1000 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	CSSC	X	X	PLLRDYC	HSERDYC	HSIRDYC	X	LSIRDYC
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	PLLRDYIE	HSERDYIE	HSIRDYIE	X	LSIRDYIE
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CSSF	X	X	PLLRDYF	HSERDYF	HSIRDYF	X	LSIRDYF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LSIRDYF	**LSI 就绪中断标志 (LSI ready interrupt flag)** 在内部低速时钟就绪且 LSIRDYIE 位被置’1’时，由硬件置’1’。 由软件通过置’1’ LSIRDYC 位来清除。 0：无内部 40kHz RC 振荡器产生的时钟就绪中断； 1：内部 40kHz RC 振荡器导致时钟就绪中断。
HSIRDYF	**HSI 就绪中断标志 (HSI ready interrupt flag)** 在内部高速时钟就绪且 HSIRDYIE 位被置’1’时，由硬件置’1’。 由软件通过置’1’ HSIRDYC 位来清除。 0：无内部 8MHz RC 振荡器产生的时钟就绪中断； 1：内部 8MHz RC 振荡器导致时钟就绪中断。
HSERDYF	**HSE 就绪中断标志 (HSE ready interrupt flag)** 在外部低速时钟就绪且 HSERDYIE 位被置’1’时，由硬件置’1’。 由软件通过置’1’ HSERDYC 位来清除。 0：无外部 4-16MHz 振荡器产生的时钟就绪中断； 1：外部 4-16MHz 振荡器导致时钟就绪中断。
PLLRDYF	**PLL 就绪中断标志 (PLL ready interrupt flag)** 在 PLL 就绪且 PLLRDYIE 位被置’1’时，由硬件置’1’。 由软件通过置’1’ PLLRDYC 位来清除。 0：无 PLL 上锁产生的时钟就绪中断； 1：PLL 上锁导致时钟就绪中断。
CSSF	**时钟安全系统中断标志 (Clock security system interrupt flag)** 在外部 4-16MHz 振荡器时钟出现故障时，由硬件置’1’。 由软件通过置’1’ CSSC 位来清除。 0：无 HSE 时钟失效产生的安全系统中断； 1：HSE 时钟失效导致了时钟安全系统中断。

LSIRDYIE	**LSI 就绪中断使能 (LSI ready interrupt enable)** 由软件置’1’或清’0’来使能或关闭内部 40kHz RC 振荡器就绪中断。 0：LSI 就绪中断关闭； 1：LSI 就绪中断使能。
HSIRDYIE	**HSI 就绪中断使能 (HSI ready interrupt enable)** 由软件置’1’或清’0’来使能或关闭内部 8MHz RC 振荡器就绪中断。 0：HSI 就绪中断关闭； 1：HSI 就绪中断使能。
HSERDYIE	**HSE 就绪中断使能 (HSE ready interrupt enable)** 由软件置’1’或清’0’来使能或关闭外部 4-16MHz 振荡器就绪中断。 0：HSE 就绪中断关闭； 1：HSE 就绪中断使能。
PLLRDYIE	**PLL 就绪中断使能 (PLL ready interrupt enable)** 由软件置’1’或清’0’来使能或关闭 PLL 就绪中断。 0：PLL 就绪中断关闭； 1：PLL 就绪中断使能。
LSIRDYC	**清除 LSI 就绪中断 (LSI ready interrupt clear)** 由软件置’1’来清除 LSI 就绪中断标志位 LSIRDYF。 0：无作用； 1：清除 LSI 就绪中断标志位 LSIRDYF。
HSIRDYC	**清除 HSI 就绪中断 (HSI ready interrupt clear)** 由软件置’1’来清除 HSI 就绪中断标志位 HSIRDYF。 0：无作用； 1：清除 HSI 就绪中断标志位 HSIRDYF。
HSERDYC	**清除 HSE 就绪中断 (HSE ready interrupt clear)** 由软件置’1’来清除 HSE 就绪中断标志位 HSERDYF。 0：无作用； 1：清除 HSE 就绪中断标志位 HSERDYF。
PLLRDYC	**清除 PLL 就绪中断 (PLL ready interrupt clear)** 由软件置’1’来清除 PLL 就绪中断标志位 PLLRDYF。 0：无作用； 1：清除 PLL 就绪中断标志位 PLLRDYF。
CSSC	**清除时钟安全系统中断 (Clock security system interrupt clear)** 由软件置’1’来清除 CSSF 安全系统中断标志位 CSSF。 0：无作用； 1：清除 CSSF 安全系统中断标志位。

3.2.4 APB2 外设复位寄存器 (APB2RSTR)

APB2 外设复位寄存器 (APB2RSTR)			基地址： 0x4002 1000 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	CMPRST	OPARST	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	TIM20RST	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ADC3RST	USART1RST	TIM8RST	SPI1RST	TIM1RST	ADC2RST	ADC1RST	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	IOPERST	IOPDRST	IOPCRST	IOPBRST	IOPARST	X	SYSCFGRST
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SYSCFGRST	SYSCFG 复位 (SYSCFG reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 SYSCFG。
IOPARST	IO 端口 A 复位 (IO port A reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 IO 端口 A。
IOPBRST	IO 端口 B 复位 (IO port B reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 IO 端口 B。
IOPCRST	IO 端口 C 复位 (IO port C reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 IO 端口 C。
IOPDRST	IO 端口 D 复位 (IO port D reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 IO 端口 D。
IOPERST	IO 端口 E 复位 (IO port E reset) 由软件置’1’或清’0’ 0：无作用； 1：复位 IO 端口 E。

ADC1RST	ADC1 接口复位（ADC 1 interface reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 ADC1 接口。
ADC2RST	ADC2 接口复位（ADC 2 interface reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 ADC2 接口。
TIM1RST	TIM1 定时器复位（TIM1 timer reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM1 定时器。
SPI1RST	SPI1RST：SPI1 复位（SPI 1 reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 SPI1。
TIM8RST	TIM8 定时器复位（TIM8 timer reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM8 定时器。
USART1RST	USART1 复位（USART1 reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 USART1。
ADC3RST	ADC3 接口复位（ADC3 interface reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 ADC3 接口。
TIM20RST	TIM20 复位（TIM20 reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM20。 注：同时控制 CMP,CRV
OPARST	OPA 复位（OPA reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 OPA。
CMPRST	CMP 复位（CMP reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 CMP。 注：同时控制 CMP,CRV

3.2.5 APB1 外设复位寄存器 (APB1RSTR)

APB1 外设复位寄存器 (APB1RSTR)			基地址： 0x4002 1000 偏移地址： 0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	PWRRST	BKPRST	X	CANRST	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	I2C2RST	I2C1RST	X	X	USART3RST	USART2RST	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	SPI2RST	X	X	WWDGRST	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	TIM4RST	TIM3RST	TIM2RST
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TIM2RST	**定时器 2 复位 (Timer 2 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM2 定时器。
TIM3RST	**定时器 3 复位 (Timer 3 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM3 定时器。
TIM4RST	**定时器 4 复位 (Timer 4 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 TIM4 定时器。。
WWDGRST	**窗口看门狗复位 (Window watchdog reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位窗口看门狗。
SPI2RST	**SPI2 复位 (SPI 2 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 SPI2。
USART2RST	**USART2 复位 (USART 2 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 USART2。

USART3RST	**USART3 复位 (USART 3 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 USART3。
I2C1RST	**I2C1 复位 (I2C 1 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 I2C 1。
I2C2RST	**I2C2 复位 (I2C 2 reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 I2C 2。
CANRST	**CAN 复位 (CAN reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位 CAN。
BKPRST	**备份接口复位 (Backup interface reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位备份接口。
PWRRST	**电源接口复位 (Power interface reset)** 由软件置’1’或清’0’ 0：无作用； 1：复位电源接口。

3.2.6 AHB 外设时钟使能寄存器(AHBENR)

AHB 外设时钟使能寄存器 (AHBENR)			基地址： 0x4002 1000 偏移地址： 0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	ME	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	CRCEN	X	FLITFEN	DMA3EN	SRAMEN	X	DMA1EN
Reset:	0	0	0	1	0	1	0	0

位	功能描述
DMA1EN	DMA1 时钟使能 (DMA1 clock enable) 由软件置’1’或清’0’。 0：DMA1 时钟关闭； 1：DMA1 时钟开启。
SRAMEN	SRAM 时钟使能 (SRAM interface clock enable) 由软件置’1’或清’0’来开启或关闭睡眠模式时 SRAM 时钟。 0：睡眠模式时 SRAM 时钟关闭； 1：睡眠模式时 SRAM 时钟开启。
DMA3EN	DMA3 时钟使能 (DMA3 interface clock enable) 由软件置’1’或清’0’。 0：DMA3 时钟关闭； 1：DMA3 时钟开启。
FLITFEN	闪存接口电路时钟使能 (FLITF clock enable) 由软件置’1’或清’0’来开启或关闭睡眠模式时闪存接口电路时钟。 0：睡眠模式时闪存接口电路时钟关闭； 1：睡眠模式时闪存接口电路时钟开启。
CRCEN	CRC 时钟使能 (CRC clock enable) 由软件置’1’或清’0’。 0：CRC 时钟关闭； 1：CRC 时钟开启。
ME	ME 时钟使能 (ME clock enable) 由软件置’1’或清’0’。 0：ME 时钟关闭； 1：ME 时钟开启。

3.2.7 APB2 外设时钟使能寄存器(APB2ENR)

APB2 外设时钟使能寄存器 (APB2ENR)			基地址：0x4002 1000 偏移地址：0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	CMPEN	PGAEN	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	TIM20EN	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ADC3EN	USART1 EN	TIM8EN	SPI1EN	TIM1EN	ADC2EN	ADC1EN	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	IOPDEN	IOPCEN	IOPBEN	IOPAEN	X	SYSCFGE N
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CMPEN	CMP 接口时钟使能 (CMP interface clock enable) 由软件置’1’或清’0’ 0：CMP接口时钟关闭； 1：CMP接口时钟开启。 注：同时控制CMP,CRV
PGAEN	PGA 接口时钟使能 (PGA interface clock enable) 由软件置’1’或清’0’ 0：PGA 接口时钟关闭； 1：PGA 接口时钟开启。
TIM20EN	TIM20 接口时钟使能 (TIM20 interface clock enable) 由软件置’1’或清’0’ 0：TIM20 接口时钟关闭； 1：TIM20 接口时钟开启。
ADC3EN	ADC3接口时钟使能(ADC 3 interface clock enable) 由软件置1或清0 0:ADC3接口时钟关闭 1:ADC3 接口时钟开启。
USART1EN	USART1时钟使能(USART1 clock enable) 由软件置1或清0 0：USART1时钟关闭 1：USART1 时钟开启。
TIM8EN	TIM8定时器时钟使能(TIM8 Timer clock enable) 由软件置1或清0 0：TIM8定时器时钟关闭

	1：TIM8 定时器时钟开启。
SPI1EN	**SPI1时钟使能(SPI 1 clock enable)** 由软件置1或清0 0：SPI1时钟关闭 1：SPI1 时钟开启。
TIM1EN	**TIM1定时器时钟使能(TIM1 Timer clock enable)** 由软件置1或清0 0：TIM1定时器时钟关闭 1：TIM1 定时器时钟开启。
ADC2EN	**ADC2接口时钟使能(ADC 2 interface clock enable)** 由软件置1或清0 0：ADC2接口时钟关闭 1：ADC2 接口时钟开启。
ADC1EN	**ADC1接口时钟使能(ADC 1 interface clock enable)** 由软件置1或清0 0：ADC1接口时钟关闭 1：ADC1 接口时钟开启。
IOPDEN	**IO端口D时钟使能(I/O port D clock enable)** 由软件置1或清0 0：IO端口D时钟关闭 1：IO 端口 D 时钟开启。
IOPCEN	**IO端口C时钟使能(I/O port C clock enable)** 由软件置1或清0 0：IO端口C时钟关闭 1：IO 端口 C 时钟开启。
IOPBEN	**IO端口B时钟使能(I/O port B clock enable)** 由软件置1或清0 0：IO端口B时钟关闭 1：IO 端口 B 时钟开启。
IOPAEN	**IO端口A时钟使能(I/O port A clock enable)** 由软件置1或清0 0：IO端口A时钟关闭 1：IO 端口 A 时钟开启。
SYSCFGEN	**SYSCFG时钟使能 (SYSCFG clock enable)** 由软件置’1’或清’0’ 0：SYSCFG时钟关闭; 1：SYSCFG 时钟开启。

注：访问：字，半字和字节访问通常无访问等待周期。但在 APB2 总线上的外设被访问时，将插入等待状态
直到 APB2 的外设访问结束。
当外设时钟没有启用时，软件不能读出外设寄存器的数值，返回的数值始终是 0x0

3.2.8 APB1 外设时钟使能寄存器(APB1ENR)

APB1 外设时钟使能寄存器 (APB1ENR)			基地址： 0x4002 1000 偏移地址： 0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	PWREN	BKPEN	X	CANEN	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	I2C2EN	I2C1EN	X	X	USART3EN	USART2EN	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	SPI2EN	X	X	WWDGEN	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	TIM4EN	TIM3EN	TIM2EN
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TIM2EN	**定时器 2 时钟使能 (Timer 2 clock enable)** 由软件置’1’或清’0’ 0：定时器 2 时钟关闭； 1：定时器 2 时钟开启。
TIM3EN	**定时器 3 时钟使能 (Timer 3 clock enable)** 由软件置’1’或清’0’ 0：定时器 3 时钟关闭； 1：定时器 3 时钟开启。
TIM4EN	**定时器 4 时钟使能 (Timer 4 clock enable)** 由软件置’1’或清’0’ 0：定时器 4 时钟关闭； 1：定时器 4 时钟开启。
WWDGEN	**窗口看门狗时钟使能 (Window watchdog clock enable)** 由软件置’1’或清’0’ 0：窗口看门狗时钟关闭； 1：窗口看门狗时钟开启。
SPI2EN	**SPI 2 时钟使能 (SPI 2 clock enable)** 由软件置’1’或清’0’ 0：SPI 2 时钟关闭； 1：SPI 2 时钟开启。
USART2EN	**USART2 时钟使能 (USART 2 clock enable)** 由软件置’1’或清’0’ 0：USART2 时钟关闭； 1：USART2 时钟开启。

USART3EN	**USART3 时钟使能（USART 3 clock enable）** 由软件置’1’或清’0’ 0：USART3 时钟关闭； 1：USART3 时钟开启。
I2C1EN	**I2C 1 时钟使能（I2C 1 clock enable）** 由软件置’1’或清’0’ 0：I2C 1 时钟关闭； 1：I2C 1 时钟开启。
I2C2EN	**I2C 2 时钟使能（I2C 2 clock enable）** 由软件置’1’或清’0’ 0：I2C 2 时钟关闭； 1：I2C 2 时钟开启。
CANEN	**CAN 时钟使能（CAN clock enable）** 由软件置’1’或清’0’ 0：CAN 时钟关闭； 1：CAN 时钟开启。
BKPEN	**备份接口时钟使能（Backup interface clock enable）** 由软件置’1’或清’0’ 0：备份接口时钟关闭； 1：备份接口时钟开启。
PWREN	**电源接口时钟使能（Power interface clock enable）** 由软件置’1’或清’0’ 0：电源接口时钟关闭； 1：电源接口时钟开启。

3.2.9 备份域控制寄存器 (BDCR)

备份域控制寄存器 (BDCR)			基地址：0x4002 1000 偏移地址：0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	BDRST
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	RTCEN	X	X	X	X	X	RTCSEL[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BDRST	**备份域软件复位**(Backup domain software reset) 由软件置’1’或清’0’ 0：复位未激活； 1：复位整个备份域。
RTCEN	**RTC时钟使能**(RTC clock enable) 由软件置’1’或清’0’ 0：RTC时钟关闭； 1：RTC 时钟开启。
RTCSEL[9:8]：	**RTC时钟源选择**(RTC clock source selection) 由软件设置来选择RTC时钟源。 一旦RTC时钟源被选定，直到下次后备域被复位，它不能在被改变。 可通过设置BDRST位来清除。 00：无时钟； 10：LSI振荡器作为RTC时钟； 11：HSE 振荡器在 128 分频后作为 RTC 时钟。

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

3.2.10 控制/状态寄存器 (CSR)

控制/状态寄存器 (CSR)			基地址： 0x4002 1000 偏移地址： 0x24					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	X	RMVF
Reset:	1	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	LSIRDY	LSION
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LSION	**内部低速振荡器使能 (Internal low-speed oscillator enable)** 由软件置’1 或清’0’。 0：内部 40kHz RC 振荡器关闭； 1：内部 40kHz RC 振荡器开启。
LSIRDY	**内部低速振荡器就绪 (Internal low-speed oscillator ready)** 由硬件置’1 或清’0’来指示内部 40kHz RC 振荡器是否就绪。在 LSION 清零后，3 个内部 40kHz RC 振荡器的周期后 LSIRDY 被清零。 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。
RMVF	**清除复位标志 (Remove reset flag)** 由软件置’1’来清除复位标志。 0：无作用； 1：清除复位标志。
PINRSTF	**NRST 引脚复位标志 (PIN reset flag)** 在 NRST 引脚复位发生时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无 NRST 引脚复位发生； 1：发生 NRST 引脚复位。
PORRSTF	**上电/掉电复位标志 (POR/PDR reset flag)** 在上电/掉电复位发生时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无上电/掉电复位发生； 1：发生上电/掉电复位。
SFTRSTF	**软件复位标志 (Software reset flag)** 在软件复位发生时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无软件复位发生； 1：发生软件复位。

IWDGRSTF	**独立看门狗复位标志（Independent watchdog reset flag）** 在独立看门狗复位发生在 VDD 区域时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无独立看门狗复位发生； 1：发生独立看门狗复位。
WWDGRSTF	**窗口看门狗复位标志（Window watchdog reset flag）** 在窗口看门狗复位发生时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无窗口看门狗复位发生； 1：发生窗口看门狗复位。
LPWRRSTF	**低功耗复位标志（Low-power reset flag）** 在低功耗管理复位发生时由硬件置’1’；由软件通过写 RMVF 位清除。 0：无低功耗管理复位发生; 1：发生低功耗管理复位.

3.2.11 AHB 外设复位寄存器（AHBRSTR）

AHB 外设复位寄存器（AHBRSTR）			基地址： 0x4002 1000 偏移地址： 0x28					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	X	X	X	X	MERST	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	X	CRCRST	X	X	DMA3RST	X	X	DMA1RST
Reset:	0	0	0	0	0	0	0	0

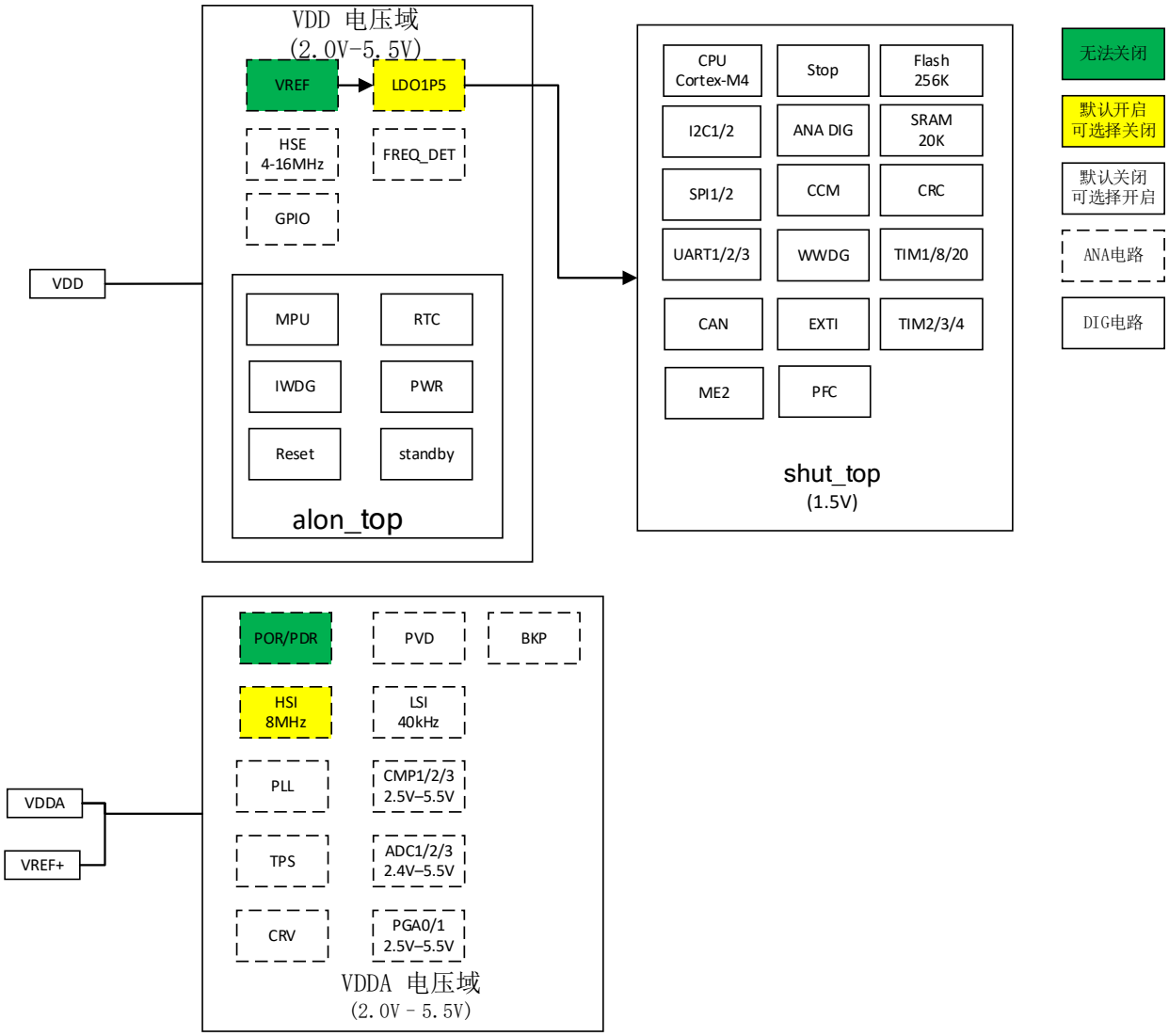
位	功能描述
DMA1RST	DMA1 复位（DMA1 reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 DMA1。
DMA3RST	DMA3 复位（DMA3 reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 DMA3。
CRCRST	CRC 复位（CRC reset） 由软件置’1’或清’0’

	0：无作用； 1：复位 CRC。
MERST	ME 接口复位（ME interface reset） 由软件置’1’或清’0’ 0：无作用； 1：复位 ME 接口。

4 电源单元

4.1 电源框图

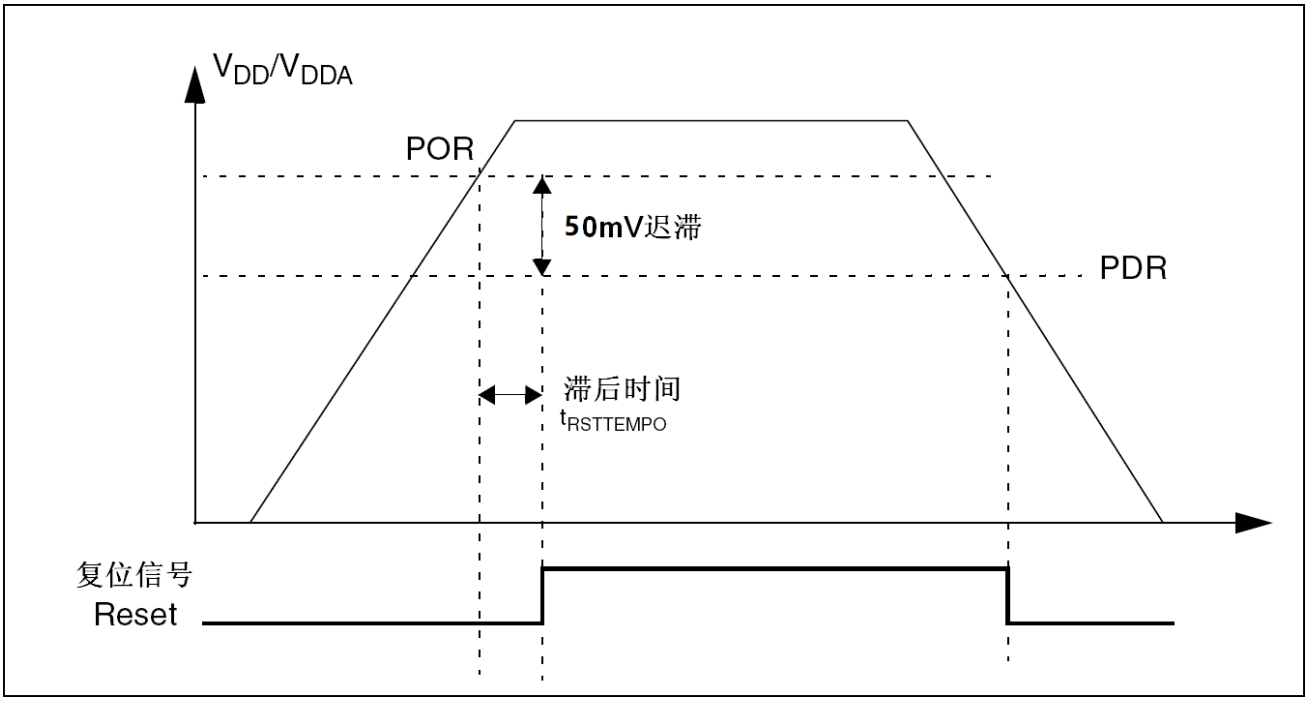
- VDD：2.5V ~ 5.5V



4.2 电源管理器

4.2.1 上电复位(POR)和掉电复位(PDR)

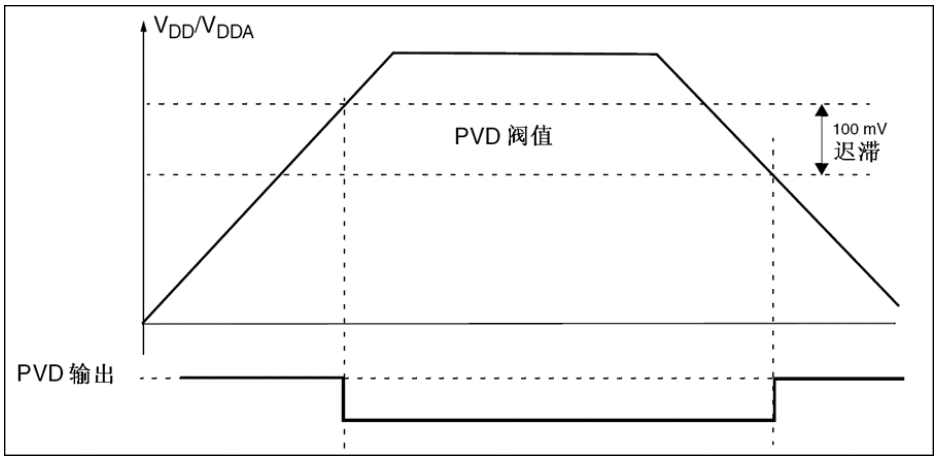
RX32F103 内部有一个完整的上电复位(POR)和掉电复位(PDR)电路，当供电电压达到 2V 时系统既能正常工作。当 VDD/VDDA 低于指定的限位电压 VPOR/VPDR 时，系统保持为复位状态，而无需外部复位电路。关于上电复位和掉电复位的细节请参考数据手册的电气特性部分。
图:上电复位和掉电复位的波形图



4.2.2 可编程电压检测器（PVD）

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

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



4.3 低功耗模式

在系统或电源复位以后，微控制器处于运行状态。当 CPU 不需继续运行时，可以利用多种低功耗模式来节省功耗，例如等待某个外部事件时。用户需要根据最低电源消耗、最快速启动时间和可用的唤醒源等条件，选定一个最佳的低功耗模式。
RX32F103 有三种低功耗模式：

- Sleep mode:睡眠模式(Cortex™-M3 内核停止，所有外设包括 Cortex-M3 核心的外设，如 NVIC、系统时钟(SysTick)等仍在运行)
- Stop mode:停止模式(所有的时钟都已停止)
- Standby mode:待机模式(1.8V 电源关闭)

此外，在运行模式下，可以通过以下方式中的一种降低功耗：

- 降低系统时钟
- 关闭 APB 和 AHB 总线上未被使用的外设时钟。

表：低功耗模式一览

模式	进入	唤醒	对1.8V区域时钟的影响	对V_{DD}区域时钟的影响	电压调节器
睡眠 (SLEEP-NOW或SLEEP-ON-EXIT)	WFI	任一中断	CPU 时钟关，对其他时钟和ADC时钟无影响	无	开
	WFE	唤醒事件			
停机	PDDS和LPDS位+SLEEPDEEP位+WFI或WFE	任一外部中断(在外部中断寄存器中设置)	关闭所有1.8V 区域的时钟	HSI 和HSE的振荡器关闭	开启或处于低功耗模式(依据电源控制寄存器(PWR_CR)的设定)
待机	PDDS位+SLEEPDEEP位+WFI或WFE	WKUP引脚的上升沿、RTC闹钟事件、NRST引脚上的外部复位、IWDG复位			关

4.3.1 降低系统时钟

在运行模式下，通过对预分频寄存器进行编程，可以降低任意一个系统时钟(SYSCLK 、HCLK、PCLK1、PCLK2)的速度。进入睡眠模式前，也可以利用预分频器来降低外设的时钟。详见时钟配置寄存器(CFGR)。

4.3.2 外部时钟的控制

在运行模式下，任何时候都可以通过停止为外设和内存提供时钟(HCLK 和 PCLKx)来减少功耗。为了在睡眠模式下更多地减少功耗，可在执行 WFI 或 WFE 指令前关闭所有外设的时钟。
通过设置 AHB 外设时钟使能寄存器 (AHBENR) 、 APB2 外设时钟使能寄存器(APB2ENR)和 APB1 外设时钟使能寄存器(APB1ENR)来开关各个外设模块的时钟。

4.3.3 睡眠模式

进入睡眠模式

通过执行 WFI 或 WFE 指令进入睡眠状态。根据 Cortex™-M3 系统控制寄存器中的 SLEEPONEXIT 位的值，有两种选项可用于选择睡眠模式进入机制：

● SLEEP-NOW：如果 SLEEPONEXIT 位被清除，当 WRI 或 WFE 被执行时，微控制器立即进入睡眠模式。

● SLEEP-ON-EXIT：如果 SLEEPONEXIT 位被置位，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

在睡眠模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。

退出睡眠模式

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

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

● 在外设控制寄存器中使能一个中断，而不是在 NVIC(嵌套向量中断控制器)中使能，并且在 Cortex-M3 系统控制寄存器中使能 SEVONPEND 位。当 MCU 从 WFE 中唤醒后，外设的中断挂起位和外设的 NVIC 中断通道挂起位(在 NVIC 中断清除挂起寄存器中)必须被清除。

● 配置一个外部或内部的 EXIT 线为事件模式。当 MCU 从 WFE 中唤醒后，因为与事件线对应的挂起位未被设置，不必清除外设的中断挂起位或外设的 NVIC 中断通道挂起位。

该模式唤醒所需的时间最短，因为没有时间损失在中断的进入或退出上。

表：SLEEP-NOW 模式

SLEEP-NOW模式	说明
进入	在以下条件下执行WFI(等待中断)或WFE(等待事件)指令： – SLEEPDEEP = 0 和 – SLEEPONEXIT = 0 参考Cortex-M3系统控制寄存器。
退出	如果执行WFI进入睡眠模式： 中断：参考中断向量表 如果执行WFE进入睡眠模式： 唤醒事件：参考唤醒事件管理
唤醒延时	无

表：SLEEP-ON-EXIT 模式

SLEEP-ON_EXIT模式	说明
进入	在以下条件下执行WFI指令： – SLEEPDEEP = 0和 – SLEEPONEXIT = 1 参考Cortex™-M3系统控制寄存器
退出	中断：参考中断向量表
唤醒延时	无

4.3.4 停止模式

停止模式是在 Cortex™-M3 的深睡眠模式基础上结合了外设的时钟控制机制，在停止模式下电压调节器可运行在正常或低功耗模式。此时在 1.8V 供电区域的的所有时钟都被停止，PLL、HSI 和 HSE RC 振荡器的功能被禁止，SRAM 和寄存器内容被保留下来。

在停止模式下，所有的 I/O 引脚都保持它们在运行模式时的状态。

进入停止模式

在停止模式下，通过设置电源控制寄存器(PWR_CR)的 LPDS 位使内部调节器进入低功耗模式， 能够降低更多的功耗。

如果正在进行闪存编程，直到对内存访问完成，系统才进入停止模式。

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

- 独立看门狗(IWDG)：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。
- 实时时钟(RTC)：通过备份域控制寄存器 (BDCR)的 RTCEN 位来设置。
- 内部 RC 振荡器(LSI RC)：通过控制/状态寄存器 (CSR)的 LSION 位来设置。
- 外部 32.768kHz 振荡器(LSE)：通过备份域控制寄存器 (BDCR)的 LSEON 位设置。

在停止模式下，如果在进入该模式前 ADC 和 DAC 没有被关闭，那么这些外设仍然消耗电流。通过设置寄存器 ADC_CR2 的 ADON 位和寄存器 DAC_CR 的 ENx 位为 0 可关闭这 2 个外设。

退出停止模式

当一个中断或唤醒事件导致退出停止模式时，HSI RC 振荡器被选为系统时钟。

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

表：停止模式

停止模式	说明
进入	在以下条件下执行WFI(等待中断)或WFE(等待事件)指令： – 设置Cortex-M3系统控制寄存器中的SLEEPDEEP位 – 清除电源控制寄存器(PWR_CR)中的PDDS位 – 通过设置PWR_CR中LPDS位选择电压调节器的模式 注：为了进入停止模式，所有的外部中断的请求位(挂起寄存器(EXTI_PR))和RTC的闹钟标志都必须被清除，否则停止模式的进入流程将会被跳过，程序继续运行。
退出	如果执行WFI进入停止模式： 设置任一外部中断线为中断模式(在NVIC中必须使能相应的外部中断向量)。参见中断向量表。 如果执行WFE进入停止模式： 设置任一外部中断线为事件模式。参见唤醒事件管理。
唤醒延时	HSI RC唤醒时间 + 电压调节器从低功耗唤醒的时间。

4.3.5 待机模式

待机模式可实现系统的最低功耗。该模式是在 Cortex-M3 深睡眠模式时关闭电压调节器。整个 1.8V 供电区域被断电。PLL、HSI 和 HSE 振荡器也被断电。SRAM 和寄存器内容丢失。只有备份的寄存器和待机电路维持供电。

进入待机模式

可以通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗(IWDG)：可通过写入看门狗的键寄存器或硬件选择来启动 IWDG。一旦启动了独立看门狗，除了系统复位，它不能再被停止。
- 实时时钟(RTC)：通过备用区域控制寄存器(BDCR)的 RTCEN 位来设置。
- 内部 RC 振荡器(LSI RC)：通过控制/状态寄存器(CSR)的 LSION 位来设置。
- 外部 32.768kHz 振荡器(LSE)：通过备用区域控制寄存器(BDCR)的 LSEON 位设置。

退出待机模式

当一个外部复位(NRST 引脚)、IWDG 复位、WKUP 引脚上的上升沿或 RTC 闹钟事件的上升沿发生时，微控制器从待机模式退出。从待机唤醒后，除了电源控制/状态寄存器(PWR_CSR)，所有寄存器被复位。
从待机模式唤醒后的代码执行等同于复位后的执行(采样启动模式引脚、读取复位向量等)。电源控制/状态寄存器(PWR_CSR)将会指示内核由待机状态退出。
关于如何退出待机模式，详见下表。
表：待机模式

待机模式	说明
进入	在以下条件下执行WFI(等待中断)或WFE(等待事件)指令： – 设置Cortex™-M3系统控制寄存器中的SLEEPDEEP位 – 设置电源控制寄存器(PWR_CR)中的PDDS位 – 清除电源控制/状态寄存器(PWR_CSR)中的WUF位
退出	WKUP引脚的上升沿、RTC闹钟事件的上升沿、NRST引脚上外部复位、IWDG复位。
唤醒延时	复位阶段时电压调节器的启动。

待机模式下的输入/输出端口状态
在待机模式下，所有的 I/O 引脚处于高阻态，除了以下的引脚：
- 复位引脚(始终有效)
- 当被设置为防侵入或校准输出时的 TAMPER 引脚
- 被使能的唤醒引脚

调试模式
默认情况下，如果在进行调试微处理器时，使微处理器进入停止或待机模式，将失去调试连接。这是因为 Cortex™-M3 的内核失去了时钟。
然而，通过设置 DBGMCU_CR 寄存器中的某些配置位，可以在使用低功耗模式下调试软件。

4.3.6 低功耗模式下的自动唤醒(AWU)

RTC 可以在不需要依赖外部中断的情况下唤醒低功耗模式下的微控制器(自动唤醒模式)。RTC 提供一个可编程的时间基数，用于周期性从停止或待机模式下唤醒。通过对备份区域控制寄存器(BDCR)的 RTCSEL[1:0]位的编程，三个 RTC 时钟源中的二个时钟源可以选作实现此功能。
- 低功耗 32.768kHz 外部晶振(LSE)

该时钟源提供了一个低功耗且精确的时间基准。(在典型情形下消耗小于 1μA)
- 低功耗内部 RC 振荡器(LSI RC)

使用该时钟源，节省了一个 32.768kHz 晶振的成本。但是 RC 振荡器将少许增加电源消耗。为了用 RTC 闹钟事件将系统从停止模式下唤醒，必须进行如下操作：
- 配置外部中断线 17 为上升沿触发。
- 配置 RTC 使其可产生 RTC 闹钟事件。

如果要从待机模式中唤醒，不必配置外部中断线 17。

4.4 特殊功能寄存器说明

4.4.1 电源控制寄存器(PWR_CR)

电源控制寄存器(PWR_CR)			基地址： 0x4000 7000 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	DBP
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DBP	**取消后备区域的写保护** 在复位后，RTC和后备寄存器处于被保护状态以防意外写入。设置这位允许写入这些寄存器。 0：禁止写入RTC和后备寄存器 1：允许写入RTC和后备寄存器 注：如果RTC的时钟是HSE/128，该位必须保持为'1'。
PLS[7:5]	**PVD 电平选择** 这些位用于选择电源电压监测器的电压阀值 000：2.2V 100：2.6V 001：2.3V 101：2.7V 010：2.4V 110：2.8V 011：2.5V 111：2.9V 注：详细说明参见数据手册中的电气特性部分。
PVDE	**电源电压监测器(PVD)使能** 0：禁止PVD 1：开启PVD
CSBF	**清除待机位** 始终读出为0 0：无功效 1：清除 SBF 待机位(写)

CWUF	**清除唤醒位** 始终读出为0 0：无功效 1：2 个系统时钟周期后清除 WUF 唤醒位(写)
PDDS	**掉电深睡眠** 与LPDS位协同操作 0：当CPU进入深睡眠时进入停机模式，调压器的状态由LPDS位控制。 1：CPU 进入深睡眠时进入待机模式。
LPDS	**深睡眠下的低功耗** PDDS=0时，与PDDS位协同操作 0：在停机模式下电压调压器开启 1：在停机模式下电压调压器出于低功耗模式

注：复位值：0x0000 0000(从待机模式唤醒时清除)

4.4.2 电源控制/状态寄存器(PWR_CSR)

电源控制/状态寄存器(PWR_CSR)			基地址： 0x4000 7000 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	EWUP
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	PVD0	SBF	WUF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
EWUP	**使能WKUP引脚** 0：WKUP引脚为通用I/O。WKUP引脚上的事件不能将CPU从待机模式唤醒 1：WKUP引脚用于将CPU从待机模式唤醒，WKUP引脚被强置为输入下拉的配置(WKUP引脚上的上升沿将系统从待机模式唤醒) 注：在系统复位时清除这一位。
PVD0	**PVD输出**

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

注：复位值：0x0000 0000(从待机模式唤醒时不被清除)与标准的 APB 读相比，读此寄存器需要额外的 APB 周期。

5 备份寄存器（BKP）

5.1 BKP 简介

备份寄存器是 42 个 16 位的寄存器，可用来存储 84 个字节的用户应用程序数据。他们处在备份域里，当 VDD 电源被切断，他们仍然由 VBAT 维持供电。当系统在待机模式下被唤醒，或系统复位或电源复位时，他们也不会被复位。
此外，BKP 控制寄存器用来管理侵入检测和 RTC 校准功能。
复位后，对备份寄存器和 RTC 的访问被禁止，并且备份域被保护以防止可能存在的意外的写操作。执行以下操作可以使能对备份寄存器和 RTC 的访问。

- 通过设置寄存器 APB1ENR 的 PWREN 和 BKPEN 位来打开电源和后备接口的时钟
- 电源控制寄存器(PWR_CR)的 DBP 位来使能对后备寄存器和 RTC 的访问。

5.2 BKP 特性

- 20 字节数据后备寄存器
- 用来管理防侵入检测并具有中断功能的状态/控制寄存器
- 用来存储 RTC 校验值的校验寄存器。

5.3 BKP 功能描述

5.3.1 侵入检测

当 TAMPER 引脚上的信号从 0 变成 1 或者从 1 变成 0(取决于备份控制寄存器 BKP_CR 的 TPAL 位)， 会产生一个侵入检测事件。侵入检测事件将所有数据备份寄存器内容清除。
然而为了避免丢失侵入事件，侵入检测信号是边沿检测的信号与侵入检测允许位的逻辑与，从而在侵入检测引脚被允许前发生的侵入事件也可以被检测到。

- 当 TPAL=0 时：如果在启动侵入检测 TAMPER 引脚前(通过设置 TPE 位)该引脚已经为高电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件(尽管在 TPE 位置’1’后并没有出现上升沿)。
- 当 TPAL=1 时：如果在启动侵入检测引脚 TAMPER 前(通过设置 TPE 位)该引脚已经为低电平，一旦启动侵入检测功能，则会产生一个额外的侵入事件(尽管在 TPE 位置’1’后并没有出现下降沿)。

设置 BKP_CSR 寄存器的 TPIE 位为’1’，当检测到侵入事件时就会产生一个中断。
在一个侵入事件被检测到并被清除后，侵入检测引脚 TAMPER 应该被禁止。然后，在再次写入备份数据寄存器前重新用 TPE 位启动侵入检测功能。这样，可以阻止软件在侵入检测引脚上仍然有侵入事件时对备份数据寄存器进行写操作。这相当于对侵入引脚 TAMPER 进行电平检测。
注：当 VDD 电源断开时，侵入检测功能仍然有效。为了避免不必要的复位数据备份寄存器，TAMPER 引脚应该在片外连接到正确的电平。

5.3.2 RTC 校准

为方便测量，RTC 时钟可以经 64 分频输出到侵入检测引脚 TAMPER 上。通过设置 RTC 校验寄存器(BKP_RTCCR)的 CCO 位来开启这一功能。
通过配置 CAL[6:0]位，此时钟可以最多减慢 121ppm。

5.4 特殊功能寄存器说明

5.4.1 备份数据寄存器 x(BKP_DRx) (x = 1…10)

备份数据寄存器 x(BKP_DRx) (x = 1…10)			基地址： 0x4000 6C00 偏移地址： 0x04～0x28,0x40～0xBC					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	D[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	D[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
D[15:0]	**备份数据** 这些位可以被用来写入用户数据。 注意：BKP_DRx寄存器不会被系统复位、电源复位、从待机模式唤醒所复位。 它们可以由备份域复位来复位或(如果侵入检测引脚 TAMPER 功能被开启时)由侵入引脚事件复位。

5.4.2 RTC 时钟校准寄存器(BKP_RTCCR)

RTC 时钟校准寄存器 (BKP_RTCCR)			基地址： 0x4000 6C00 偏移地址： 0x2C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	ASOS	ASOE
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCO	CAL[6:0]						
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ASOS	**闹钟或秒输出选择(Alarm or second output selection)** 当设置了ASOE位，ASOS位可用于选择在TAMPER引脚上输出的是RTC秒脉冲还是闹钟脉冲信号。 0：输出RTC闹钟脉冲 1：输出秒脉冲 注：该位只能被后备区的复位所清除
ASOE	**允许输出闹钟或秒脉冲(Alarm or second output enable)** 根据ASOS位的设置，该位允许RTC闹钟或秒脉冲输出到TAMPER引脚上。 输出脉冲的宽度为一个 RTC 时钟的周期。设置了 ASOE 位时不能开启 TAMPER 的功能。注：该位只能被后备区的复位所清除
CCO	**校准时钟输出(Calibration clock output)** 0：无影响 1：此位置1可以在侵入检测引脚输出经64分频后的RTC时钟。当CCO位置1时，必须关闭侵入检测功能以避免检测到无用的侵入信号。 注：当 VDD 供电断开时，该位被清除。
CAL[6:0]	**校准值(Calibration value)** 校准值表示在每2^{20}个时钟脉冲内将有多少个时钟脉冲被跳过。这可以用来对RTC进行校准，以$1000000/2^{20}$ppm的比例减慢时钟。 RTC 时钟可以被减慢 0~121ppm。

5.4.3 备份控制寄存器(BKP_CR)

备份控制寄存器(BKP_CR)			基地址： 0x4000 6C00 偏移地址： 0x30					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	TPAL	TPE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TPAL	**侵入检测TAMPER引脚有效电平(TAMPER pin active level)** 0：侵入检测TAMPER引脚上的高电平会清除所有数据备份寄存器(如果TPE位为1) 1：侵入检测 TAMPER 引脚上的低电平会清除所有数据备份寄存器(如果 TPE 位为 1)
TPE	**启动侵入检测TAMPER引脚(TAMPER pin enable)** 0：侵入检测TAMPER引脚作为通用IO口使用 1：开启侵入检测引脚作为侵入检测使用

5.4.4 备份控制/状态寄存器(BKP_CSR)

备份控制/状态寄存器 (BKP_CSR)			基地址： 0x4000 6C00 偏移地址： 0x34					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	TIF	TEF
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	TPIE	CTI	CTE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TIF	**侵入中断标志(Tamper interrupt flag)** 当检测到有侵入事件且TPIE位为1时，此位由硬件置1。通过向CTI位写1来清除此标志位(同时也清除了中断)。如果TPIE位被清除，则此位也会被清除。 0：无侵入中断 1：产生侵入中断 注意：仅当系统复位或由待机模式唤醒后才复位该位
TEF	**侵入事件标志(Tamper event flag)** 当检测到侵入事件时此位由硬件置1。通过向CTE位写1可清除此标志位 0：无侵入事件 1：检测到侵入事件 注：侵入事件会复位所有的 BKP_DRx 寄存器。只要 TEF 为 1，所有的 BKP_DRx 寄存器就一直保持复位状态。当此位被置 1 时，若对 BKP_DRx 进行写操作，写入的值不会被保存。
TPIE	**允许侵入TAMPER引脚中断(TAMPER pin interrupt enable)** 0：禁止侵入检测中断 1：允许侵入检测中断(BKP_CR寄存器的TPE位也必须被置1) 注 1：侵入中断无法将系统内核从低功耗模式唤醒。 注 2：仅当系统复位或由待机模式唤醒后才复位该位。
CTI	**清除侵入检测中断(Clear tamper interrupt)** 此位只能写入，读出值为0。 0：无效 1：清除侵入检测中断和 TIF 侵入检测中断标志
CTE	**清除侵入检测事件(Clear tamper event)** 此位只能写入，读出值为0。 0：无效 1：清除 TEF 侵入检测事件标志(并复位侵入检测器)。

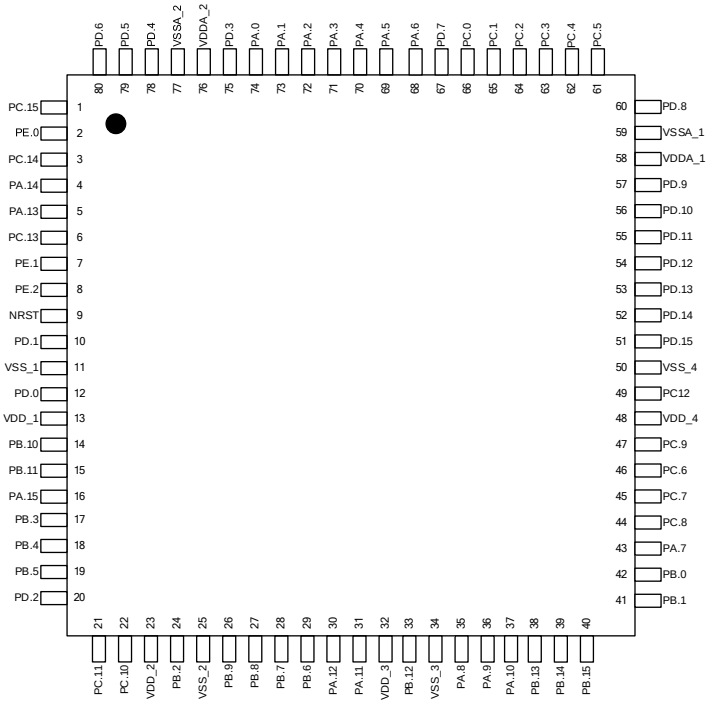
6 GPIO 模块

6.1 概述

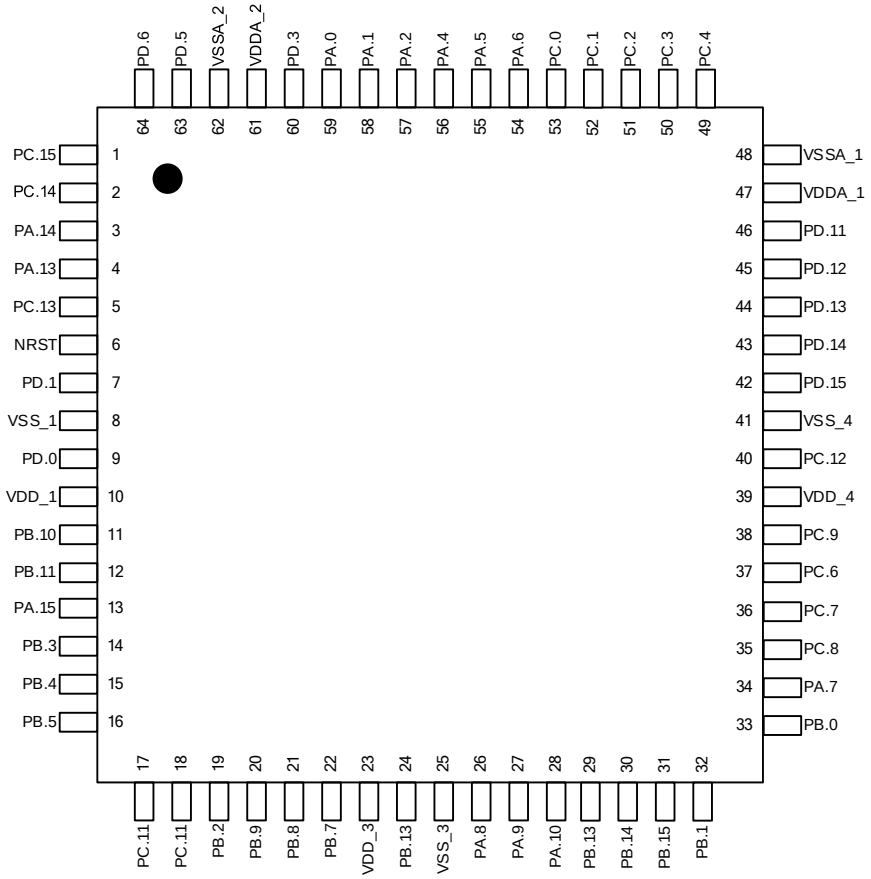
RX32S50 系列提供的 I/O 包括：PA.0~PA.15，PB.0~PB.15，PC.0~PC.15，PD.0~PD.15，PE.0~PE.2，支持 67 个双向 I/O 引脚。JTAG pin 复用其他功能后不能自动恢复 JTAG 功能，需手动配回 JTAG 功能。

6.2 引脚排列

LQFP80



LQFP64


PD.6 64
PD.5 63
VSSA_2 62
VDDA_2 61
PD.3 60
PA.0 59
PA.1 58
PA.2 57
PA.4 56
PA.5 55
PA.6 54
PC.0 53
PC.1 52
PC.2 51
PC.3 50
PC.4 49
PC.15 1
PC.14 2
PA.14 3
PA.13 4
PC.13 5
NRST 6
PD.1 7
VSS_1 8
PD.0 9
VDD_1 10
PB.10 11
PB.11 12
PA.15 13
PB.3 14
PB.4 15
PB.5 16
48 VSSA_1
47 VDDA_1
46 PD.11
45 PD.12
44 PD.13
43 PD.14
42 PD.15
41 VSS_4
40 PC.12
39 VDD_4
38 PC.9
37 PC.6
36 PC.7
35 PC.8
34 PA.7
33 PB.0
17 PC.11
18 PC.11
19 PB.2
20 PB.9
21 PB.8
22 PB.7
23 VDD_3
24 PB.13
25 VSS_3
26 PA.8
27 PA.9
28 PA.10
29 PB.13
30 PB.14
31 PB.15
32 PB.1

6.3 引脚定义

SO PIN	标识	引脚类型	Main Fuction (after reset)	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	Analog		
1	PC15	I/O	PC15									ADC1_IN6		
2	PE0	I/O	PE0			TIM20_CH4	USART2_CK					ADC1_IN7		
3	PC14	I/O	PC14/WKUP									ADC1_IN8		
4	PA14	I/O	JTCK/SWCLK	JTCK/SWCLK										
5	PA13	I/O	JTMS/SWDIO	JTMS/SWDIO										
6	PC13	I/O	PC13/TAMPER-RTC											
7	PE1	I/O	PE1			TIM20_CH5	USART2_TX							
8	PE2	I/O	PE2			TIM20_CH6	USART2_RX							
9	NRST	I												
10	PD1	I/O	OSC_OUT									OSC_OUT		
11	VSS_1	G												
12	PD0	I/O	OSC_IN									OSC_IN		
13	VDD_1	P												
14	PB10	I/O	PB10		TIM2_CH3		USART1_TX		I2C2_SCL					
15	PB11	I/O	PB11		TIM2_CH4		USART1_RX		I2C2_SDA					
16	PA15	I/O	JTDI	JTDI	TIM2_CH1_ETR			SPI1_NSS						
17	PB3	I/O	JTDO	JTDO/TRACESWO	TIM2_CH2		USART1_RX	SPI1_SCK						
18	PB4	I/O	JNTRST	JNTRST	TIM3_CH1			SPI1_MISO						
19	PB5	I/O	PB5		TIM3_CH2		USART1_TX	SPI1_MOSI	I2C1_SMBAI					
20	PD2	I/O	PD2		TIM3_ETR									
21	PC11	I/O	PC11		TIM3_CH4		USART3_RX							
22	PC10	I/O	PC10		TIM3_CH3		USART3_TX							
23	VDD_2	P												
24	PB2	I/O	PB2											
25	VSS_2	G												
26	PB9	I/O	PB9		TIM4_CH4				I2C1_SDA	CANTX				
27	PB8	I/O	PB8		TIM4_CH3				I2C1_SCL	CANRX				
28	PB7	I/O	PB7		TIM4_CH2		USART3_RX		I2C1_SDA					
29	PB6	I/O	PB6		TIM4_CH1		USART3_TX		I2C1_SCL					
30	PA12	I/O	PA12			TIM1_ETR	USART1_RTS			CANTX				
31	PA11	I/O	PA11				USART1_CTS			CANRX				
32	VDD_3	P												
33	PB12	I/O	PB12			TIM1_BKIN	USART1_CK	SPI2_NSS	I2C2_SMBAI	TIM8_ETR				
34	VSS_3	G												
35	PA8	I/O	PA8	MCO		TIM1_CH1								
36	PA9	I/O	PA9			TIM1_CH2								
37	PA10	I/O	PA10			TIM1_CH3								
38	PB13	I/O	PB13			TIM1_CH1N				TIM8_CH6				
39	PB14	I/O	PB14			TIM1_CH2N				TIM8_CH5				
40	PB15	I/O	PB15			TIM1_CH3N				TIM8_CH4				
41	PB1	I/O	PB1			TIM1_CH4				TIM8_CH3N				
42	PB0	I/O	PB0			TIM1_CH5				TIM8_CH2N				
43	PA7	I/O	PA7			TIM1_CH6				TIM8_CH1N				
44	PC8	I/O	PC8							TIM8_CH3				
45	PC7	I/O	PC7							TIM8_CH2				
46	PC6	I/O	PC6							TIM8_CH1				
47	PC9	I/O	PC9		TIM3_CH4	TIM8_BKIN				TIM8_CH4				
48	VDD_4	P												
49	PC12	I/O	PC12			TIM20_BKIN	USART3_CK							
50	VSS_4	G												
51	PD15	I/O	PD15		TIM4_CH4	TIM20_CH1		SPI2_NSS						
52	PD14	I/O	PD14		TIM4_CH3	TIM20_CH2	USART2_CK	SPI2_SCK						
53	PD13	I/O	PD13		TIM4_CH2	TIM20_CH3	USART2_TX	SPI2_MOSI						
54	PD12	I/O	PD12		TIM4_CH1	TIM20_CH1N	USART2_RX	SPI2_MISO						
55	PD11	I/O	PD11		TIM2_CH4	TIM20_CH2N	USART2_RTS							
56	PD10	I/O	PD10		TIM2_CH3	TIM20_CH3N	USART2_CTS							
57	PD9	I/O	PD9									ADC3_IN0		
58	VDDA_1	P												
59	VSSA_1	G												
60	PD8	I/O	PD8									ADC3_IN1		
61	PC5	I/O	PC5									ADC3_IN2		
62	PC4	I/O	PC4									ADC3_IN3		CMP3_N
63	PC3	I/O	PC3									ADC3_IN4		CMP3_P
64	PC2	I/O	PC2									ADC3_IN5	OPA_O	
65	PC1	I/O	PC1									ADC3_IN6	OPA_N	
66	PC0	I/O	PC0									ADC3_IN7	OPA_P	
67	PD7	I/O	PD7									ADC2_IN5		
68	PA6	I/O	PA6									ADC2_IN4		
69	PA5	I/O	PA5									ADC2_IN3	PGA2_N	CMP2_N
70	PA4	I/O	PA4									ADC2_IN2	PGA2_P	CMP2_P
71	PA3	I/O	PA3									ADC2_IN1		
72	PA2	I/O	PA2									ADC2_IN0		
73	PA1	I/O	PA1									ADC1_IN0	PGA1_N	CMP1_N
74	PA0	I/O	PA0									ADC1_IN1	PGA1_P	CMP1_P
75	PD3	I/O	PD3									ADC1_IN2		
76	VDDA_2	P												
77	VSSA_2	G												
78	PD4	I/O	PD4								CMP1_OUT	ADC1_IN3		
79	PD5	I/O	PD5								CMP2_OUT	ADC1_IN4		
80	PD6	I/O	PD6								CMP3_OUT	ADC1_IN5		

64 PIN	标识	引脚类型	Main Fuction (after reset)	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	Analog		
1	PC15	I/O	PC15									ADC1_IN6		
2	PC14	I/O	PC14/WKUP									ADC1_IN8		
3	PA14	I/O	JTCK/SWCLK	JTCK/SWCLK										
4	PA13	I/O	JTMS/SWDIO	JTMS/SWDIO										
5	PC13	I/O	PC13/TAMPER-RTC											
6	NRST	I												
7	PD1	I/O	OSC_OUT									OSC_OUT		
8	VSS_1	G												
9	PD0	I/O	OSC_IN									OSC_IN		
10	VDD_1	P												
11	PB10	I/O	PB10		TIM2_CH3		USART1_TX		I2C2_SCL					
12	PB11	I/O	PB11		TIM2_CH4		USART1_RX		I2C2_SDA					
13	PA15	I/O	JTDI	JTDI	TIM2_CH1_ETR			SPI1_NSS						
14	PB3	I/O	JTDO	JTDO/TRACESWO	TIM2_CH2		USART1_RX	SPI1_SCK						
15	PB4	I/O	JNTRST	JNTRST	TIM3_CH1			SPI1_MISO						
16	PB5	I/O	PB5		TIM3_CH2		USART1_TX	SPI1_MOSI	I2C1_SMBAI					
17	PC11	I/O	PC11		TIM3_CH4		USART3_RX							
18	PC10	I/O	PC10		TIM3_CH3		USART3_TX							
19	PB2	I/O	PB2											
20	PB9	I/O	PB9		TIM4_CH4				I2C1_SDA	CANTX				
21	PB8	I/O	PB8		TIM4_CH3				I2C1_SCL	CANRX				
22	PB7	I/O	PB7		TIM4_CH2		USART3_RX		I2C1_SDA					
23	VDD_3	P												
24	PB12	I/O	PB12			TIM1_BKIN	USART1_CK	SPI2_NSS	I2C2_SMBAI	TIM8_ETR				
25	VSS_3	G												
26	PA8	I/O	PA8	MCO		TIM1_CH1								
27	PA9	I/O	PA9			TIM1_CH2								
28	PA10	I/O	PA10			TIM1_CH3								
29	PB13	I/O	PB13			TIM1_CH1N				TIM8_CH6				
30	PB14	I/O	PB14			TIM1_CH2N				TIM8_CH5				
31	PB15	I/O	PB15			TIM1_CH3N				TIM8_CH4				
32	PB1	I/O	PB1			TIM1_CH4				TIM8_CH3N				
33	PB0	I/O	PB0			TIM1_CH5				TIM8_CH2N				
34	PA7	I/O	PA7			TIM1_CH6				TIM8_CH1N				
35	PC8	I/O	PC8							TIM8_CH3				
36	PC7	I/O	PC7							TIM8_CH2				
37	PC6	I/O	PC6							TIM8_CH1				
38	PC9	I/O	PC9		TIM3_CH4	TIM8_BKIN				TIM8_CH4				
39	VDD_4	P												
40	PC12	I/O	PC12			TIM20_BKIN	USART3_CK							
41	VSS_4	G												
42	PD15	I/O	PD15		TIM4_CH4	TIM20_CH1		SPI2_NSS						
43	PD14	I/O	PD14		TIM4_CH3	TIM20_CH2	USART2_CK	SPI2_SCK						
44	PD13	I/O	PD13		TIM4_CH2	TIM20_CH3	USART2_TX	SPI2_MOSI						
45	PD12	I/O	PD12		TIM4_CH1	TIM20_CH1N	USART2_RX	SPI2_MISO						
46	PD11	I/O	PD11		TIM2_CH4	TIM20_CH2N	USART2_RTS							
47	VDDA_1	P												
48	VSSA_1	G												
49	PC4	I/O	PC4									ADC3_IN3		CMP3_N
50	PC3	I/O	PC3									ADC3_IN4		CMP3_P
51	PC2	I/O	PC2									ADC3_IN5	OPA_O	
52	PC1	I/O	PC1									ADC3_IN6	OPA_N	
53	PC0	I/O	PC0									ADC3_IN7	OPA_P	
54	PA6	I/O	PA6									ADC2_IN4		
55	PA5	I/O	PA5									ADC2_IN3	PGA2_N	CMP2_N
56	PA4	I/O	PA4									ADC2_IN2	PGA2_P	CMP2_P
57	PA2	I/O	PA2									ADC2_IN0		
58	PA1	I/O	PA1									ADC1_IN0	PGA1_N	CMP1_N
59	PA0	I/O	PA0									ADC1_IN1	PGA1_P	CMP1_P
60	PD3	I/O	PD3									ADC1_IN2		
61	VDDA_2	P												
62	VSSA_2	G												
63	PD5	I/O	PD5								CMP2_OUT	ADC1_IN4		
64	PD6	I/O	PD6								CMP3_OUT	ADC1_IN5		

6.4 功能描述

6.4.1 GPIO 锁定机制

锁定机制允许冻结 IO 配置，当在一个端口位上执行了锁定（LOCK）程序，在下一次复位之前，将不能再更改端口位的配置。

6.5 特殊功能寄存器说明

6.5.1 GPIO 复用功能低位寄存器 (GPIOx_AFRL)(x=A-E)

GPIO 复用功能低位寄存器 (GPIOx_AFRL)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1400(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	AFR7[3:0]				AFR6[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	AFR5[3:0]				AFR4[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	AFR3[3:0]				AFR2[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	AFR1[3:0]				AFR0[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
AFRy[3:0]:	**端口x位y的复用功能选择(y = 0..7)** 这些位通过软件写入，用于配置复用功能I/O AFRy 选择: 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1xxx: Reserved

6.5.2 GPIO 复用功能高位寄存器 (GPIOx_AFRH) (x=A-E)

GPIO 复用功能高位寄存器 (GPIOx_AFRH)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x24					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	AFR15[3:0]				AFR14[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	AFR13[3:0]				AFR12[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	AFR11[3:0]				AFR10[3:0]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	AFR9[3:0]				AFR8[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
AFRy[3:0]:	**端口x位y的复用功能选择(y = 8..15)** 这些位通过软件写入，用于配置复用功能I/O AFRy 选择: 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1xxx: Reserved

6.5.3 GPIO 端口上拉/下拉寄存器(GPIOx_PUPDR) (x=A-E)

GPIO 端口上拉/下拉寄存器 (GPIOx_PUPDR) (x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PUPDRy[1:0]	端口x配置位(y = 0..15) 这些位通过软件写入，用于配置I/O上拉或下拉 PUPDRy selection: 00: 无上拉或下拉 01: 上拉 10: 下拉 11: 保留

注: Default:

0x6400 0000 for port A;

0x0000 0100 for port B;

0x0000 0000 for other ports.

6.5.4 GPIO 端口方向配置寄存器 (GPIOx_MODER)(x=A-E)

GPIO 端口配置寄存器 (GPIOx_MODER)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
MODERy[1:0]	端口 x 配置位(y = 0…15) (Port x configuration bits) 软件通过这些位配置相应的 I/O 端口,用于配置 I/O 方向模式 00：输入（复位状态） 01：通用输出模式 10：复用功能模式 11：模拟模式

注：Default:

- 0xA800 0000 for port A
- 0x0000 0280 for port B
- 0x0000 0000 for other ports

6.5.5 GPIO 端口输出类型配置寄存器 (GPIOx_OTYPER)(x=A-E)

GPIO 端口输出类型配置寄存器 (GPIOx_OTYPER)(x=A-E)			基地址：0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OTy	端口 x 配置位 (Port x configuration bits) (y = 0..15) 这些位通过软件写入，用于配置 I/O 端口的输出类型。 0：输出推挽（复位状态） 1：输出开漏

6.5.6 GPIO 端口输出速度配置寄存器 (GPIOx_OSPEEDR)(x=A-E)

GPIO 端口输出速度配置寄存器 (GPIOx_OSPEEDR)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OSPEEDRy[1:0]	端口 x 配置位 (Port x configuration bits) (y = 0..15) 这些位通过软件写入，用于配置 I/O 输出速度。 x0：低速 01：中速 11：高速

注：Default:

- 0x0C00 0000 for port A
- 0x0000 00C0 for port B
- 0x0000 0000 for other ports

6.5.7 GPIO 端口输入数据寄存器 (GPIOx_IDR)(x=A-E)

GPIO 端口输入数据寄存器 (GPIOx_IDR)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE)					
			偏移地址： 0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
IDRy	端口输入数据 (Port input data) (y = 0..15) 这些位为只读形式，只能在字模式下访问。它们包含相应 I/O 端口的输入值。

6.5.8 GPIO 端口输出数据寄存器 (GPIOx_ODR)(x=A-E)

GPIO 端口输出数据寄存器 (GPIOx_ODR)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ODRy	**端口输出数据** (Port output data) (y = 0..15) 这些位可通过软件读取和写入。 注意：对于原置位 /复位，通过写入 GPIOx_BSRR 寄存器，可分别对 ODR 位进行置位和复位 (x = A..E)。

6.5.9 GPIO 端口位设置/清除寄存器 (GPIOx_BSRR)(x=A-E)

GPIO 端口位设置/清除寄存器 (GPIOx_BSRR)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8

Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BRy	**端口 x 复位位 y (Port x reset bit y) (y = 0..15)** 这些位为只写形式，只能在字、半字或字节模式下访问。读取这些位可返回值 0x0000。 0：不会对相应的 ODRx 位执行任何操作 1：对相应的 ODRx 位进行复位 注意： 如果同时对 BSx 和 BRx 置位，则 BSx 的优先级更高。
BSy	**端口 x 置位位 y (Port x set bit y) (y= 0..15)** 这些位为只写形式，只能在字、半字或字节模式下访问。读取这些位可返回值 0x0000。 0：不会对相应的 ODRx 位执行任何操作 1：对相应的 ODRx 位进行置位

6.5.10 GPIO 端口配置锁定寄存器 (GPIOx_LCKR)(x=A-E)

GPIO 端口配置锁定寄存器 (GPIOx_LCKR)(x=A-E)			基地址： 0x4001 0800(GPIOA), 0x4001 0C00(GPIOB) 0x4001 1000(GPIOC) 0x4001 1800(GPIOD) 0x4001 1800(GPIOE) 偏移地址： 0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	LCKK
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LCKK	**锁定键 (Lock key)** 可随时读取此位。可使用锁定键写序列对其进行修改。 0：端口配置锁定键未激活。

	1：端口配置锁定键已激活。直到 MCU 复位时，才锁定 GPIOx_LCKR 寄存器。 锁定键写序列： WR LCKR[16] = ‘1’ + LCKR[15:0] WR LCKR[16] = ‘0’ + LCKR[15:0] WR LCKR[16] = ‘1’ + LCKR[15:0] RD LCKR RD LCKR[16] = ‘1’（此读操作为可选操作，但它可确认锁定已激活） 注意： 在锁定键写序列期间，不能更改 LCK[15:0] 的值。 锁定序列中的任何错误都将中止锁定操作。 在任一端口位上的第一个锁定序列之后，对 LCKK 位的任何读访问都将返回“1”，直到下一次 CPU 复位为止。
LCKy	**端口 x 锁定位 y (Port x lock bit y) (y= 0..15)** 这些位都是读/写位，但只能在 LCKK 位等于“0”时执行写操作。 0：端口配置未锁定 1：端口配置已锁定

注：锁定键写序列仅操作一次，应将需要 LOCK 的 IO 用或的写法，同时 LOCK

7 中断模块

7.1 中断向量说明

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 时钟安全系统(CSS)联接到NMI向量	0x0000_0008
	-1	固定	硬件失效(HardFault)	所有类型的失效	0x0000_000C
	0	可设置	存储管理(MemManage)	存储器管理	0x0000_0010
	1	可设置	总线错误(BusFault)	预取指失败，存储器访问失败	0x0000_0014
	2	可设置	错误应用(UsageFault)	未定义的指令或非法状态	0x0000_0018
	-	-	-	保留	0x0000_001C ~0x0000_002B
	3	可设置	SVCall	通过SWI指令的系统服务调用	0x0000_002C
	4	可设置	调试监控(DebugMonitor)	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	窗口定时器中断	0x0000_0040
1	8	可设置	PVD / CMP	EXTI的电源电压检测(PVD) / CMP中断	0x0000_0044
2	9	可设置	TAMPER	侵入检测中断	0x0000_0048
3	10	可设置	RTC	实时时钟(RTC)全局中断	0x0000_004C
4	11	可设置	FLASH	闪存全局中断	0x0000_0050
5	12	可设置	CMU	时钟控制(CMU)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通道1	DMA1通道1全局中断	0x0000_006C
12	19	可设置	DMA1通道2	DMA1通道2全局中断	0x0000_0070
13	20	可设置	DMA1通道3	DMA1通道3全局中断	0x0000_0074
14	21	可设置	DMA1通道4	DMA1通道4全局中断	0x0000_0078
15	22	可设置	DMA1通道5	DMA1通道5全局中断	0x0000_007C
16	23	可设置	DMA1通道6	DMA1通道6全局中断	0x0000_0080
17	24	可设置	DMA1通道7	DMA1通道7全局中断	0x0000_0084
18	25	可设置	ADC1_2	ADC1和ADC2的全局中断	0x0000_0088
19	26	可设置	CAN_TX	CAN发送中断	0x0000_008C

20	27	可设置	CAN_RX0	CAN接收0中断	0x0000_0090
21	28	可设置	CAN_RX1	CAN接收1中断	0x0000_0094
22	29	可设置	CAN_SCE	CAN SCE中断	0x0000_0098
23	30	可设置	EXTI9_5	EXTI线[9:5]中断	0x0000_009C
24	31	可设置	TIM1_BRK	TIM1刹车中断	0x0000_00A0
25	32	可设置	TIM1_UP	TIM1更新中断	0x0000_00A4
26	33	可设置	TIM1_TRG_COM	TIM1触发和通信中断	0x0000_00A8
27	34	可设置	TIM1_CC	TIM1捕获比较中断	0x0000_00AC
28	35	可设置	TIM2	TIM2全局中断	0x0000_00B0
29	36	可设置	TIM3	TIM3全局中断	0x0000_00B4
30	37	可设置	TIM4	TIM4全局中断	0x0000_00B8
31	38	可设置	I2C1_EV	I^2C1事件中断	0x0000_00BC
32	39	可设置	I2C1_ER	I^2C1错误中断	0x0000_00C0
33	40	可设置	I2C2_EV	I^2C2事件中断	0x0000_00C4
34	41	可设置	I2C2_ER	I^2C2错误中断	0x0000_00C8
35	42	可设置	SPI1	SPI1全局中断	0x0000_00CC
36	43	可设置	SPI2	SPI2全局中断	0x0000_00D0
37	44	可设置	USART1	USART1全局中断	0x0000_00D4
38	45	可设置	USART2	USART2全局中断	0x0000_00D8
39	46	可设置	USART3	USART3全局中断	0x0000_00DC
40	47	可设置	EXTI15_10	EXTI线[15:10]中断	0x0000_00E0
41	48	可设置	RTCAlarm	连到EXTI的RTC闹钟中断	0x0000_00E4
42	49	-	-	保留	0x0000_00E8
43	50	可设置	TIM8_BRK	TIM8刹车中断	0x0000_00EC
44	51	可设置	TIM8_UP	TIM8更新中断	0x0000_00F0
45	52	可设置	TIM8_TRG_COM	TIM8触发和通信中断	0x0000_00F4
46	53	可设置	TIM8_CC	TIM8捕获比较中断	0x0000_00F8
47	54	可设置	ADC3	ADC3全局中断	0x0000_00FC
-	-	-	保留	保留	0x0000_0100 -0x0000_0130
61	68	可设置	PID1_6	PID1-6全局中断	0x0000_0134
62	69	可设置	ME1_2	ME1-2全局中断	0x0000_0138
-	-	-	保留	保留	0x0000_013C
64	71	可设置	CMP1_2_3	CMP1_2_3全局中断	0x0000_0140
-	-	保留	保留	保留	0x0000_0144
66	73	可设置	DMA3通道1	DMA3通道1全局中断	0x0000_0148
67	74	可设置	DMA3通道2	DMA3通道2全局中断	0x0000_014C
68	75	可设置	DMA3通道3	DMA3通道3全局中断	0x0000_0150
69	76	可设置	DMA3通道4	DMA3通道4全局中断	0x0000_0154
70	77	可设置	DMA3通道5	DMA3通道5全局中断	0x0000_0158
71	78	可设置	DMA3通道6	DMA3通道6全局中断	0x0000_015C
72	79	可设置	DMA3通道7	DMA3通道7全局中断	0x0000_0160

73	80	可设置	DMA3通道8	DMA3通道8全局中断	0x0000_0164
77	84	可设置	TIM20_BRK	TIM20刹车中断	0x0000_0174
78	85	可设置	TIM20_UP	TIM20更新中断	0x0000_0178
79	86	可设置	TIM20_TRG_COM	TIM20触发和通信中断	0x0000_017C
80	87	可设置	TIM20_CC	TIM20捕获比较中断	0x0000_0180

7.2 EXTI 中断说明

EXTI 中断即为外部 INT 口管脚中断，RX32S50 系列共有 16 个 INT 口，即 INT0-INT15，每个 INT 口都可以配置为上升沿触发和下降沿触发。用户在使用 INT 功能前应先将对应 IO 口配置为 INT 复用功能。

7.3 特殊功能寄存器列表

EXTI_IMR	中断屏蔽寄存器
EXTI_EMR	事件屏蔽寄存器
EXTI_RTSR	上升沿触发选择寄存器
EXTI_FTSR	下降沿触发选择寄存器
EXTI_SWIER	软件中断事件寄存器
EXTI_PR	挂起寄存器

7.4 特殊功能寄存器说明

7.4.1 中断屏蔽寄存器(EXTI_IMR)

偏移地址：0x00

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												MR19	MR18	MR17	MR16
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位31:20	保留，必须始终保持为复位状态(0)。
位19:0	**MRx**: 线x上的中断屏蔽 (Interrupt Mask on line x) 0：屏蔽来自线x上的中断请求； 1：开放来自线x上的中断请求。 注：位19只适用于互联型产品，对于其它产品为保留位。

7.4.2 事件屏蔽寄存器(EXTI_EMR)

偏移地址：0x04

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												MR19	MR18	MR17	MR16
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR15	MR14	MR13	MR12	MR11	MR10	MR9	MR8	MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位31:20	保留，必须始终保持为复位状态(0)。
位19:0	**MRx**: 线x上的事件屏蔽 (Event Mask on line x) 0：屏蔽来自线x上的事件请求； 1：开放来自线x上的事件请求。 注：位19只适用于互联型产品，对于其它产品为保留位。

7.4.3 上升沿触发选择寄存器(EXTI_RTSR)

偏移地址：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												TR19	TR18	TR17	TR16
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位31:19	保留，必须始终保持为复位状态(0)。
位18:0	**TRx**: 线x上的上升沿触发事件配置位 (Rising trigger event configuration bit of line x) 0：禁止输入线x上的上升沿触发(中断和事件) 1：允许输入线x上的上升沿触发(中断和事件) 注：位19只适用于互联型产品，对于其它产品为保留位。

注意:外部唤醒线是边沿触发的，这些线上不能出现毛刺信号。

在写 EXTI_RTSR 寄存器时，在外部中断线上的上升沿信号不能被识别，挂起位也不会被置位。在同一中断线上，可以同时设置上升沿和下降沿触发。即任一边沿都可触发中断。

7.4.4 下降沿触发选择寄存器(EXTI_FTSR)

偏移地址：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												TR19	TR18	TR17	TR16
												rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

位31:19	保留，必须始终保持为复位状态(0)。
位18:0	**TRx**: 线x上的下降沿触发事件配置位 (Falling trigger event configuration bit of line x) 0：禁止输入线x上的下降沿触发(中断和事件) 1：允许输入线x上的下降沿触发(中断和事件) 注：位19只适用于互联型产品，对于其它产品为保留位。

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

7.4.5 软件中断事件寄存器(EXTI_SWIER)

偏移地址：0x10
复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												SWIER19	SWIER18	SWIER17	SWIER16
												rw	rw	rw	rw

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

位31:19	保留，必须始终保持为复位状态(0)。
位18:0	**SWIERx**: 线x上的软件中断 (Software interrupt on line x) 当该位为'0'时，写'1'将设置EXTI_PR中相应的挂起位。如果在EXTI_IMR和EXTI_EMR中允许产生该中断，则此时将产生一个中断。 注：通过清除EXTI_PR的对应位(写入'1')，可以清除该位为'0'。 注：位19只适用于互联型产品，对于其它产品为保留位。

7.4.6 挂起寄存器(EXTI_PR)

偏移地址：0x14
复位值：0xXXXX XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												PR19	PR18	PR17	PR16
												rc w1	rc w1	rc w1	rc w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR15	PR14	PR13	PR12	PR11	PR10	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0
rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1	rc w1

位31:19	保留，必须始终保持为复位状态(0)。
位18:0	**PRx**: 挂起位 (Pending bit) 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置'1'。在该位中写入'1'可以清除它，也可以通过改变边沿检测的极性清除。 注：位19只适用于互联型产品，对于其它产品为保留位。

8 ADC 模块

8.1 概述

12 位 ADC 是一种逐次逼近型模拟数字转换器。它有多达 27 个通道，可测量 23 个外部信号源。各通道的 A/D 转换可以单次、连续、扫描或间断模式执行。ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阀值。

ADC 的输入时钟不得超过 12MHz，它是由 APB 经分频产生。

8.2 ADC 主要特征

●12 位分辨率
●转换结束、注入转换结束和发生模拟看门狗事件时产生中断
●单次和连续转换模式
●从通道 0 到通道 n 的自动扫描模式
●自校准
●带内嵌数据一致性的数据对齐
●采样间隔可以按通道分别编程
●规则转换和注入转换均有外部触发选项
●间断模式
●双重模式(带 2 个 ADC 的器件)
●ADC 供电要求：2.5V 到 5.5V
●ADC 输入范围：VREF- ≤ VIN ≤ VREF+
●规则通道转换期间有 DMA 请求产生。

8.2.1 ADC 模拟输入与通道号

ADC1 模拟输入与通道号对应关系如下表所示

Channel No	通道对应 IO
0	PA1
1	PA0
2	PD3
3	PD4
4	PD5
5	PD6
6	PC15
7	PE0
8	PC14
9	PGA1

ADC2 模拟输入与通道号对应关系如下表所示

C	通道对应 IO
0	PA2
1	PA3
2	PA4
3	PA5
4	PA6

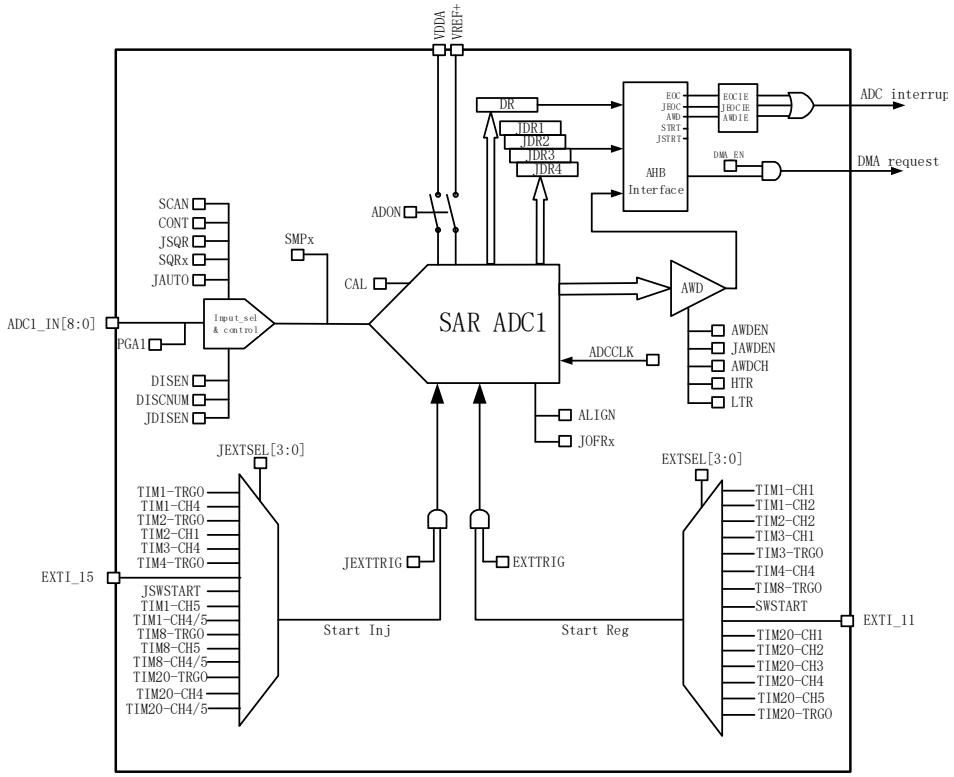
5	PD7
6	PGA2

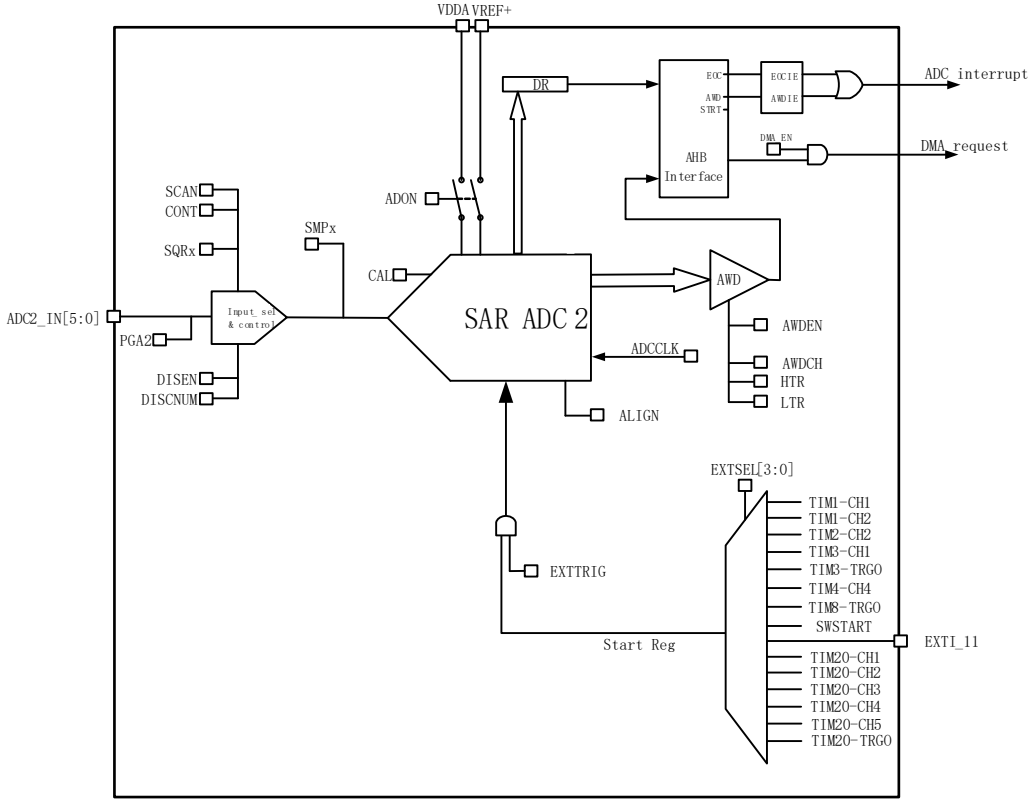
ADC3 模拟输入与通道号对应关系如下表所示

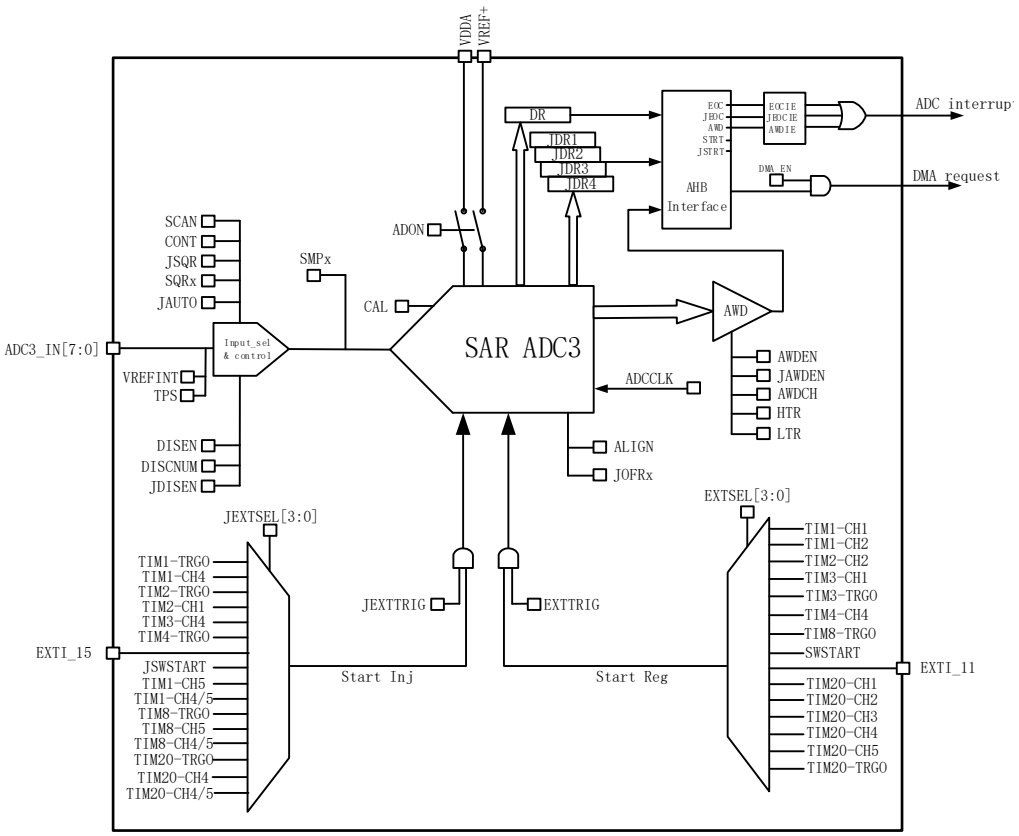
Channel No	通道对应 IO
0	PD9
1	PD8
2	PC5
3	PC4
4	PC3
5	PC2
6	PC1
7	PC0
8	VREFINT
9	TPS

8.3 ADC 功能描述

8.3.1 单个 ADC 框图




VDDA VREF+
DR
EOC
AWD
STRT
AHB
Interface
EOCIE
AWDIE
ADC_interrupt
DMA_EN
DMA_request
SCAN
CONT
SQRx
SMPx
ADON
CAL
ADC2_IN[5:0]
PGA2
Input_sel
& control
SAR ADC 2
AWD
AWDEN
ADCCLK
AWDCH
HTR
LTR
DISEN
DISCNUM
ALIGN
EXTSEL[3:0]
TIM1-CH1
TIM1-CH2
TIM2-CH2
TIM3-CH1
TIM3-TRGO
TIM4-CH4
TIM8-TRGO
SWSTART
EXTI_11
EXTTRIG
Start Reg
TIM20-CH1
TIM20-CH2
TIM20-CH3
TIM20-CH4
TIM20-CH5
TIM20-TRGO


VDDA
VREF+
DR
JDR1
JDR2
JDR3
JDR4
EOC
JEOC
AWD
STRT
JSTRT
AHB
Interface
EOCIE
JEOCIE
AWDIE
ADC_interrup
DMA_EN
DMA_request
SCAN
CONT
JSQR
SQRx
JAUTO
SMPx
ADON
CAL
ADC3_IN[7:0]
Input_sel
& control
SAR ADC3
AWD
AWDEN
JAWDEN
AWDCH
HTR
LTR
VREFINT
TPS
ADCCLK
DISEN
DISCNUM
JDISEN
ALIGN
JOFRx
EXTSEL[3:0]
JEXTSEL[3:0]
TIM1-TRGO
TIM1-CH4
TIM2-TRGO
TIM2-CH1
TIM3-CH4
TIM4-TRGO
JEXTTRIG
EXTTRIG
EXTI_15
JSWSTART
TIM1-CH5
TIM1-CH4/5
TIM8-TRGO
TIM8-CH5
TIM8-CH4/5
TIM20-TRGO
TIM20-CH4
TIM20-CH4/5
Start Inj
Start Reg
TIM1-CH1
TIM1-CH2
TIM2-CH2
TIM3-CH1
TIM3-TRGO
TIM4-CH4
TIM8-TRGO
SWSTART
EXTI_11
TIM20-CH1
TIM20-CH2
TIM20-CH3
TIM20-CH4
TIM20-CH5
TIM20-TRGO

ADC 引脚说明

名称	信号类型	注解
V_{REF+}	输入，模拟参考正极	ADC使用的高端/正极参考电压，2.5V ≤ $V_{REF+ \le VDDA}$
V_{DDA}(1)	输入，模拟电源	等效于V_{DD}的模拟电源且：2.5V ≤ V_{DDA} ≤ V_{DD}(5.5V)
V_{REF-}	输入，模拟参考负极	ADC使用的低端/负极参考电压，$V_{REF- = VSSA}$
V_{SSA}(1)	输入，模拟电源地	等效于V_{SS}的模拟电源地
ADCx_IN[15:0]	模拟输入信号	16个模拟输入通道

1．V_{DDA} 和 V_{SSA} 应该分别连接到 V_{DD} 和 V_{SS}。

8.3.2 ADC 开关控制

通过设置 ADC_CR2 寄存器的 ADON 位可给 ADC 上电。当第一次设置 ADON 位时，它将 ADC 从断电状态下唤醒。

ADC 上电延迟一段时间后(tSTAB)，再次设置 ADON 位时开始进行转换。

通过清除 ADON 位可以停止转换，并将 ADC 置于断电模式。

8.3.3 ADC 时钟

由时钟控制器提供的 ADCCLK 时钟和 PCLK2(APB2 时钟)同步。RCC 控制器为 ADC 时钟提供一个专用的可编程预分频器。

8.3.4 单次转换模式

单次转换模式下，ADC 只执行一次转换。该模式既可通过设置 ADC_CR2 寄存器的 ADON 位(只适用于规则通道)启动也可通过外部触发启动(适用于规则通道或注入通道)，这时 CONT 位为 0。

一旦选择通道的转换完成：

●如果一个规则通道被转换：

—转换数据被储存在 16 位 ADC_DR 寄存器中

—EOC(转换结束)标志被设置

—如果设置了 EOCIE，则产生中断。

●如果一个注入通道被转换：

—转换数据被储存在 16 位的 ADC_DRJ1 寄存器中

—JEOC(注入转换结束)标志被设置

—如果设置了 JEOCIE 位，则产生中断。然后 ADC 停止。

8.3.5 连续转换模式

在连续转换模式中，当前面 ADC 转换一结束马上就启动另一次转换。此模式可通过外部触发启动或通过设置 ADC_CR2 寄存器上的 ADON 位启动，此时 CONT 位是 1。

每个转换后：

●如果一个规则通道被转换：

—转换数据被储存在 16 位的 ADC_DR 寄存器中

—EOC(转换结束)标志被设置

—如果设置了 EOCIE，则产生中断。

●如果一个注入通道被转换：
—转换数据被储存在 16 位的 ADC_DRJ1 寄存器中
—JEOC(注入转换结束)标志被设置
—如果设置了 JEOCIE 位，则产生中断。

8.3.6 扫描模式

此模式用来扫描一组模拟通道。

扫描模式可通过设置 ADC_CR1 寄存器的 SCAN 位来选择。一旦这个位被设置，ADC 扫描所有被 ADC_SQRX 寄存器(对规则通道)或 ADC_JSQR(对注入通道)选中的所有通道。在每个组的每个通道上执行单次转换。在每个转换结束时，同一组的下一个通道被自动转换。如果设置了 CONT 位，转换不会在选择组的最后一个通道上停止，而是再次从选择组的第一个通道继续转换。

如果设置了 DMA 位，在每次 EOC 后，DMA 控制器把规则组通道的转换数据传输到 SRAM 中。而注入通道转换的数据总是存储在 ADC_JDRx 寄存器中。

8.3.7 外部触发转换

转换可以由外部事件触发(例如定时器捕获，EXTI 线)。如果设置了 EXTTRIG 控制位，则外部事件就能够触发转换。EXTSEL[2:0]和 JEXTSEL2:0]控制位允许应用程序选择 8 个可能的事件中的某一个，可以触发规则和注入组的采样。

注意：当外部触发信号被选为 ADC 规则或注入转换时，只有它的上升沿可以启动转换。

表 ADC1 和 ADC2 用于注入通道的外部触发

触发源	连接类型	JEXTSEL2[3:0]	TRGOPT
TIM1_TRGO事件	来自片上定时器的内部信号	0000	1
TIM1_CC4事件		0001	
TIM2_TRGO事件		0010	
TIM2_CC1事件		0011	
TIM3_CC4事件		0100	
TIM4_TRGO事件		0101	
EXTI线15	外部引脚	0110	
JSWSTART	软件控制位	0111	
TIM1_CC5事件	来自片上定时器的内部信号	1000	
TIM1_CC4/CC5事件		1001	
TIM8_TRGO事件		1010	
TIM8_ CC5事件		1011	
TIM8_ CC4/CC5事件		1100	

8.3.8 DMA 请求

因为规则通道转换的值储存在一个仅有的数据寄存器中，所以当转换多个规则通道时需要使用 DMA，这可以避免丢失已经存储在 ADC_DR 寄存器中的数据。

只有在规则通道的转换结束时才产生 DMA 请求，并将转换的数据从 ADC_DR 寄存器传输到用户指定的目的地址。

注：只有 ADC1 拥有 DMA 功能。由 ADC2 转化的数据可以通过双 ADC 模式，利用 ADC1 的 DMA 功能传输。

8.3.9 双 ADC 模式

在双 ADC 模式里，根据 ADC1_CR1 寄存器中 DUALMOD[2:0]位所选的模式，转换的启动可以是 ADC1 主和 ADC3 从的交替触发或同步触发。

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

共有 6 种可能的模式：

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

还有可以用下列方式组合使用上面的模式：

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

注意：在双 ADC 模式里，为了在主数据寄存器上读取从转换数据，必须使能 DMA 位，即使不使用 DMA 传输规则通道数据。

8.3.10 温度传感器

温度传感器可以用来测量器件周围的温度(TA)。

温度传感器在内部和 ADC1_IN16 输入通道相连接，此通道把传感器输出的电压转换成数字值。

当没有被使用时，传感器可以置于关电模式。

注意：必须设置 TSVREFE 位激活内部通道：ADC1_IN16(温度传感器)和 ADC1_IN17(VREFINT)的转换。

温度传感器输出电压随温度线性变化，由于生产过程的变化，温度变化曲线的偏移在不同芯片上会有不同(最多相差 45° C)。内部温度传感器更适合于检测温度的变化，而不是测量绝对的温度。如果需要测量精确的温度，应该使用一个外置的温度传感器。

使用传感器读温度：

1. 选择 ADC1_IN16 输入通道
2. 选择采样时间为 17.1 μs
3. 设置 ADC 控制寄存器 2(ADC_CR2)的 TSVREFE 位，以唤醒关电模式下的温度传感器
4. 通过设置 ADON 位启动 ADC 转换(或用外部触发)
5. 读 ADC 数据寄存器上的 VSENSE 数据结果
6. 利用下列公式得出温度

公式说明如下:

读温度为使用传感器：

利用下列公式得出温度

温度(° C) = {($CODE_B$ - $CODE_{ADC}$) / Avg_Slope} + 25

备注： VDD=3.3V: $CODE_B$ = 1813.6 Avg_Slope=4.412

VDD=5V: $CODE_B$ = 1175.7 Avg_Slope=2.849

$CODE_B$：Offset 数值

$CODE_{ADC}$:在不同温度下，ADC 转换的数值

Avg_Slope ：温度与$CODE_{ADC}$曲线的平均斜率(单位为 code/ ° C)。

8.3.11 ADC 中断

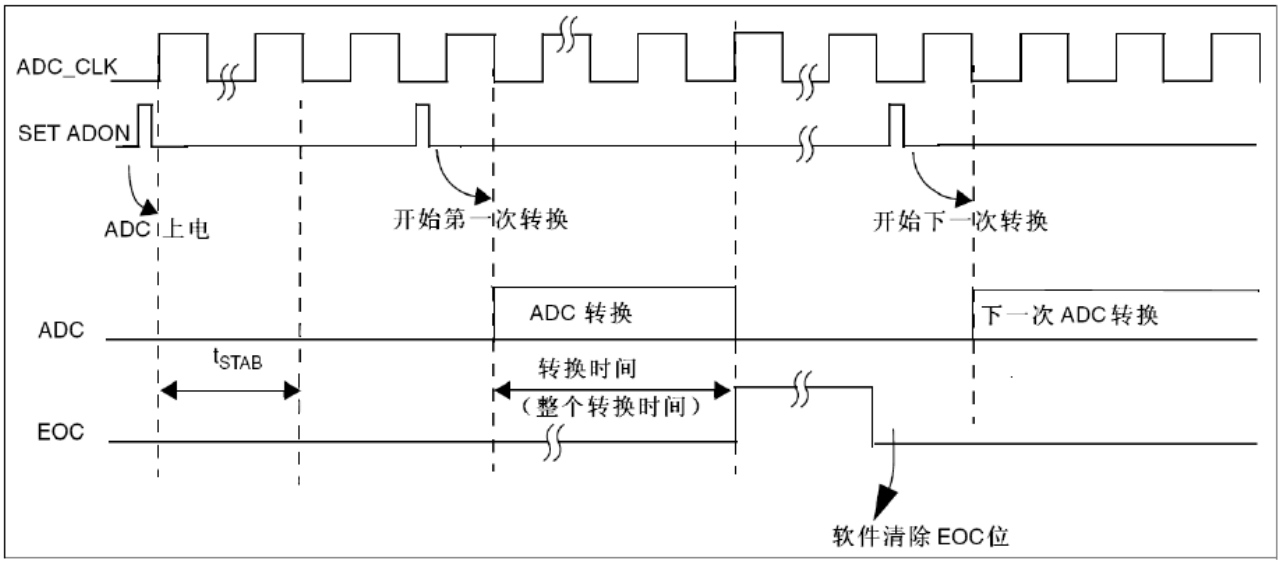
规则和注入组转换结束时能产生中断，当模拟看门狗状态位被设置时也能产生中断。它们都有独立的中断使能位。
注：ADC1 和 ADC2 的中断映射在同一个中断向量上，而 ADC3 的中断有自己的中断向量。
ADC_SR 寄存器中有 2 个其他标志，但是它们没有相关联的中断：
JSTRT(注入组通道转换的启动)
STRT(规则组通道转换的启动)

表 ADC 中断

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

8.3.12 时序图

如下图所示，ADC 在开始精确转换前需要一个稳定时间 tSTAB。在开始 ADC 转换和 14 个时钟周期后，EOC 标志被设置，16 位 ADC 数据寄存器包含转换的结果。
图：时序图

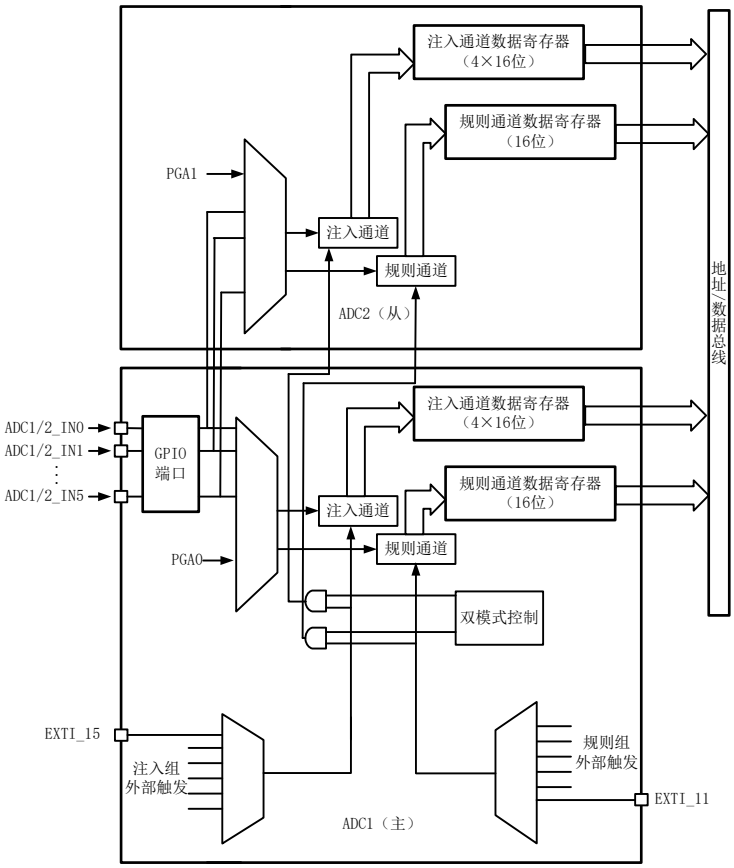


8.4 双 ADC 模式

在双 ADC 模式里，根据 ADC_CR1 寄存器中 DUALMOD[2:0]位所选的模式，转换的启动可以是 ADC1 主和 ADC3 从的交替触发或同步触发。
注意：在双 ADC 模式里，当转换配置成由外部事件触发时，用户必须将其设置成仅触发主 ADC，从 ADC 设置成软件触发，这样可以防止意外的触发从转换。但是，主和从 ADC 的外部触发必须同时被激活。
共有 5 种可能的模式：
—同步注入模式
—同步规则模式
—快速交叉模式
—慢速交叉模式
—交替触发模式
还有可以用下列方式组合使用上面的模式：
—同步注入模式 + 同步规则模式
—同步规则模式 + 交替触发模式

—同步注入模式 + 交叉模式
注意：在双 ADC 模式里，为了在主数据寄存器上读取从转换数据，必须使能 DMA 位，即使不使用 DMA 传输规则通道数据。

双 ADC 框图



8.4.1 同步注入模式

此模式转换一个注入通道组。外部触发来自 ADC1 的注入组多路开关(由 ADC1_CR2 寄存器的 JEXTSEL 选择)，它同时给 ADC3 提供同步触发。

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

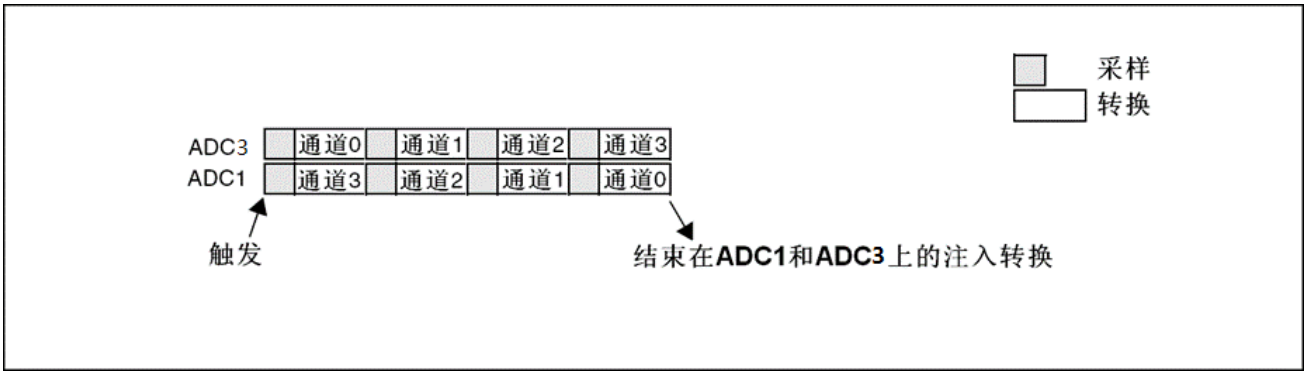
在 ADC1 或 ADC3 的转换结束时：

●转换的数据存储在每个 ADC 接口的 ADC_JDRx 寄存器中。

●当所有 ADC1/ADC3 注入通道都被转换时，产生 JEOC 中断(若任一 ADC 接口开放了中断)。

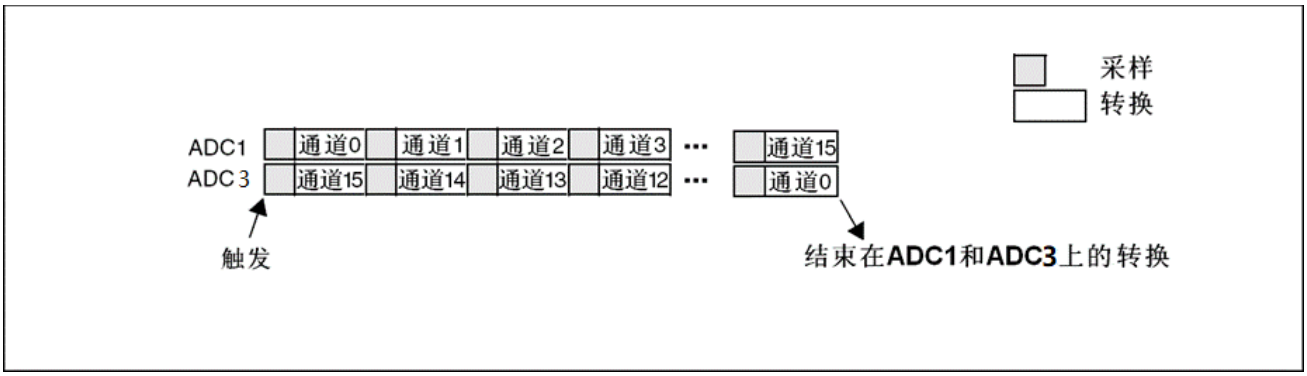
注：在同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

图：在 4 个通道上的同步注入模式



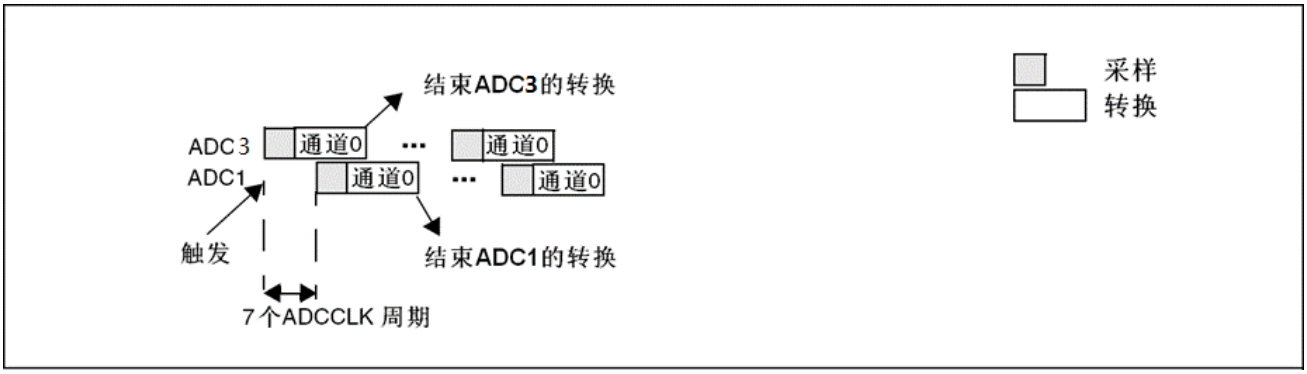
8.4.2 同步规则模式

此模式在规则通道组上执行。外部触发来自 ADC1 的规则组多路开关(由 ADC_CR2 寄存器的 EXTSEL 选择)，它同时给 ADC3 提供同步触发。
注意：不要在 2 个 ADC 上转换相同的通道((两个 ADC 在同一个通道上的采样时间不能重叠)。
在 ADC1 或 ADC3 的转换结束时：
●产生一个 32 位 DMA 传输请求(如果设置了 DMA 位)，32 位的 ADC_DR 寄存器内容传输到 SRAM 中，它上半个字包含 ADC2 的转换数据，低半个字包含 ADC1 的转换数据。
●当所有 ADC1/ADC3 规则通道都被转换完时，产生 EOC 中断(若任一 ADC 接口开放了中断)。
注：在同步规则模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。
图：在 16 个通道上的同步规则模式



8.4.3 快速交叉模式

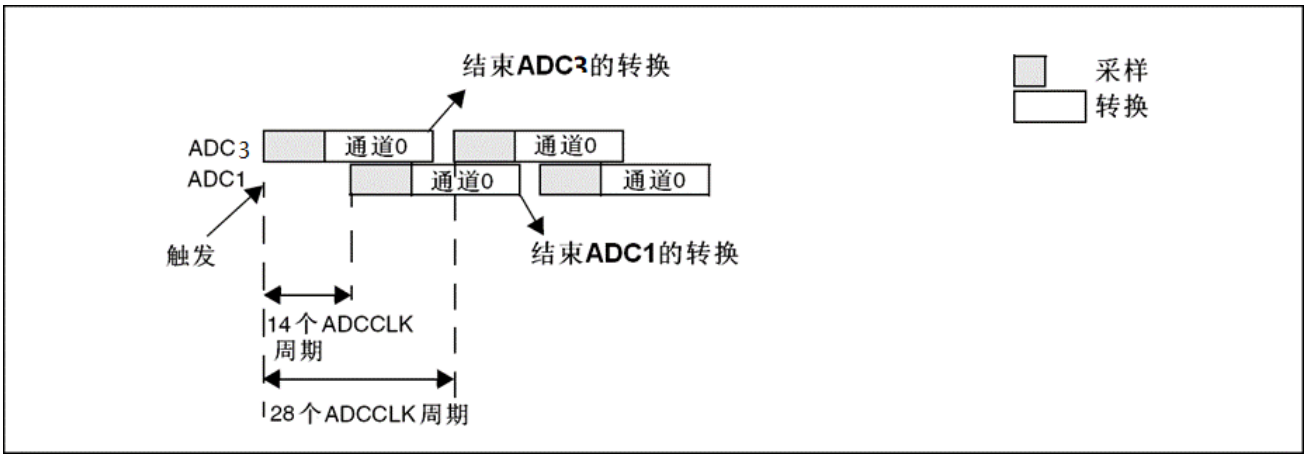
此模式只适用于规则通道组。外部触发来自 ADC1 的规则通道多路开关。外部触发产生后：
ADC3 立即启动并且 ADC1 在延迟 7 个 ADC 时钟周期后启动
如果同时设置了 ADC1 和 ADC3 的 CONT 位，所选的两个 ADC 规则通道将被连续地转换。
ADC1 产生一个 EOC 中断后(由 EOCIE 使能)，产生一个 32 位的 DMA 传输请求(如果设置了 DMA 位)，ADC1_DR 寄存器的 32 位数据被传输到 SRAM，ADC1_DR 的上半个字包含 ADC2 的转换数据，低半个字包含 ADC1 的转换数据。
注意：最大允许采样时间<7 个 ADCCLK 周期，避免 ADC1 和 ADC2 转换相同通道时发生两个采样周期的重叠。
图：在 1 个通道上连续转换模式下的快速交叉模式



8.4.4 慢速交叉模式

此模式只适用于规则通道组。外部触发来自 ADC1 的规则通道多路开关。外部触发产生后：
ADC3 立即启动并且 ADC1 在延迟 14 个 ADC 时钟周期后启动
在延迟第二次 14 个 ADC 周期后 ADC2 再次启动，如此循环。
注意：最大允许采样时间<14 个 ADCCLK 周期，以避免和下个转换重叠。

ADC1 产生一个 EOC 中断后(由 EOCIE 使能)，产生一个 32 位的 DMA 传输请求(如果设置了 DMA 位)，ADC1_DR 寄存器的 32 位数据被传输到 SRAM，ADC1_DR 的上半个字包含 ADC3 的转换数据，低半个字包含 ADC1 的转换数据。
在 28 个 ADC 时钟周期后自动启动新的 ADC3 转换。
在这个模式下不能设置 CONT 位，因为它将连续转换所选择的规则通道。
注意：应用程序必须确保当使用交叉模式时，不能有注入通道的外部触发产生。
图：在 1 个通道上的慢速交叉模式

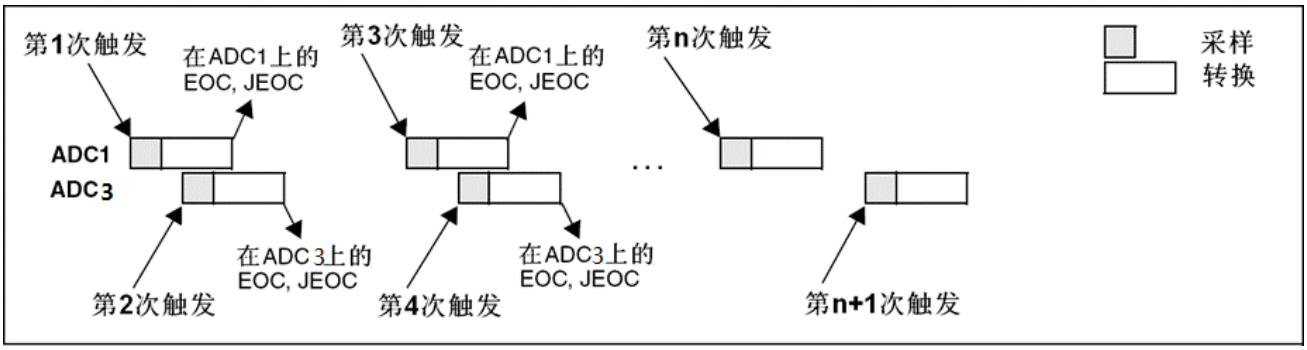


8.4.5 交替触发模式

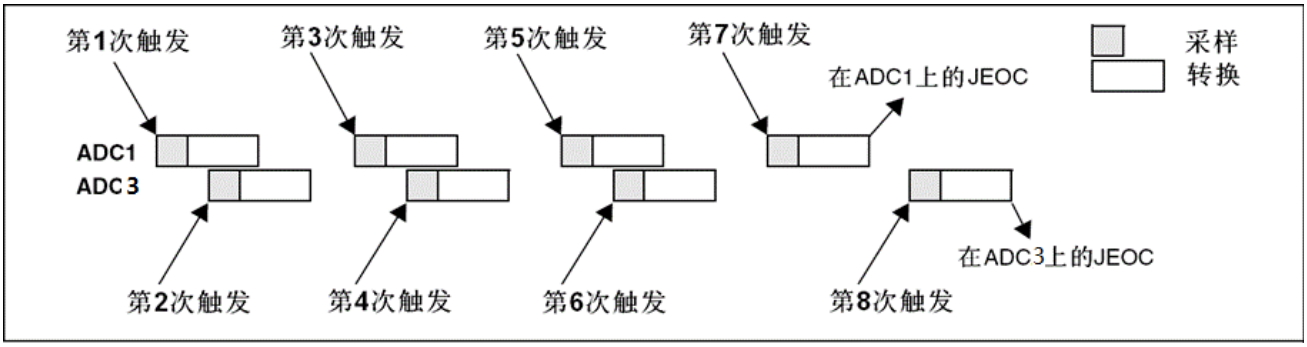
此模式只适用于注入通道组。外部触发源来自 ADC1 的注入通道多路开关。
●当第一个触发产生时，ADC1 上的所有注入组通道被转换。
●当第二个触发到达时，ADC3 上的所有注入组通道被转换。
●如此循环……
如果允许产生 JEOC 中断，在所有 ADC1 注入组通道转换后产生一个 JEOC 中断。如果允许产生 JEOC 中断，在所有 ADC3 注入组通道转换后产生一个 JEOC 中断。

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

图：交替触发：每个 ADC1 的注入通道组



如果 ADC1 和 ADC3 上同时使用了注入间断模式：
●当第一个触发产生时，ADC1 上的第一个注入通道被转换。
●当第二个触发到达时，ADC3 上的第一个注入通道被转换。
●如此循环……
如果允许产生 JEOC 中断，在所有 ADC1 注入组通道转换后产生一个 JEOC 中断。如果允许产生 JEOC 中断，在所有 ADC3 注入组通道转换后产生一个 JEOC 中断。
当所有注入组通道都转换完后，如果又有另一个外部触发，则重新开始交替触发过程。
图 ：交替触发：在间断模式下每个 ADC 上的 4 个注入通道



8.4.6 混合的规则/注入同步模式

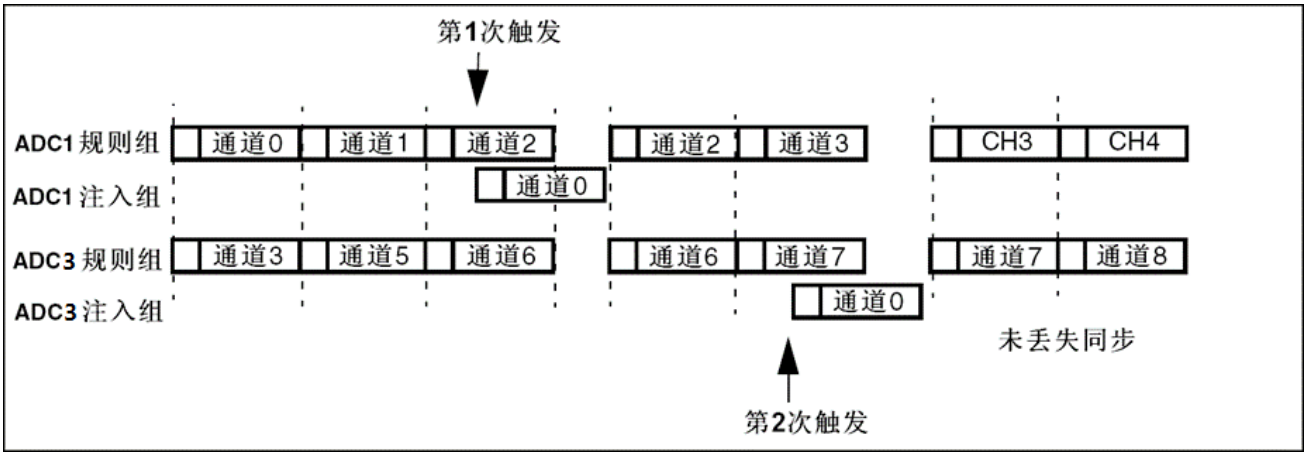
规则组同步转换可以被中断，以启动注入组的同步转换。
注：在混合的规则/注入同步模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比 2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

8.4.7 混合的同步规则+交替触发模式

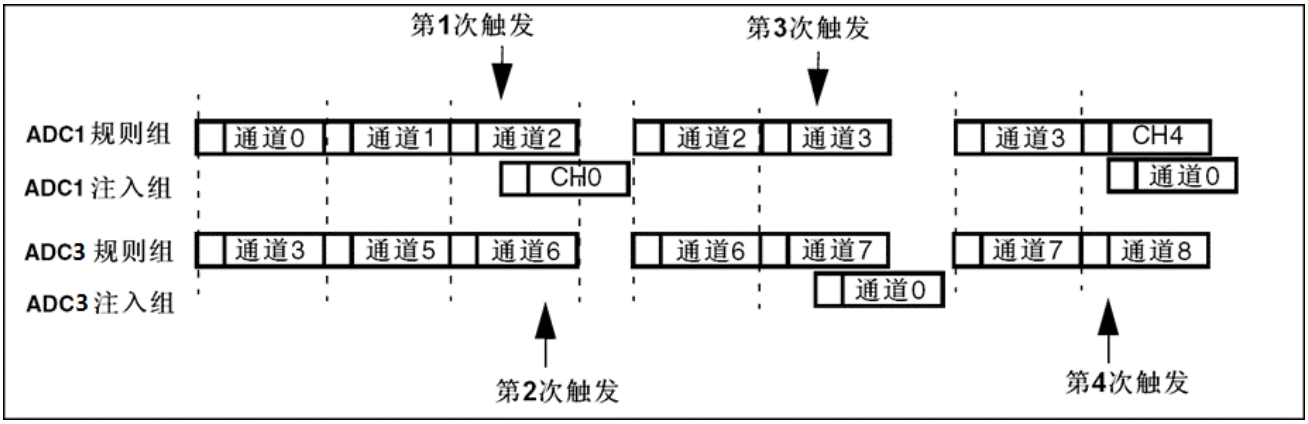
规则组同步转换可以被中断，以启动注入组交替触发转换。图 38 显示了一个规则同步转换被交替触发所中断。
注入交替转换在注入事件到达后立即启动。如果规则转换已经在运行，为了在注入转换后确保同步，所有的 ADC(主和从)的规则转换被停止，并在注入转换结束时同步恢复。
注：在混合的同步规则+交替触发模式中，必须转换具有相同时间长度的序列，或保证触发的间隔比

2 个序列中较长的序列长，否则当较长序列的转换还未完成时，具有较短序列的 ADC 转换可能会被重启。

图：交替+规则同步

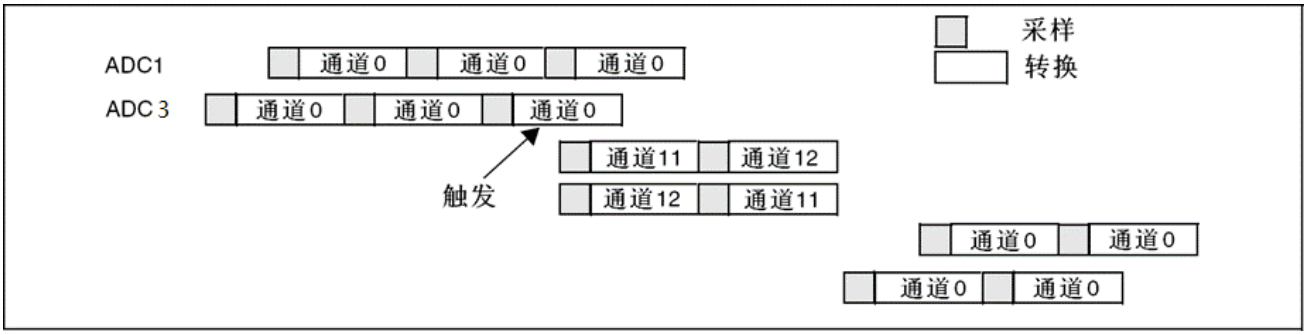


如果触发事件发生在一个中断了规则转换的注入转换期间，这个触发事件将被忽略。下图示出了这种情况的操作(第 2 个触发被忽略)。
图：触发事件发生在注入转换期间



8.4.8 混合同步注入 + 交叉模式

一个注入事件可以中断一个交叉转换。这种情况下，交叉转换被中断，注入转换被启动，在注入序列转换结束时，交叉转换被恢复。下图是这种情况的一个例子。
注：当 ADC 时钟预分频系数设置为 4 时，交叉模式恢复后不会均匀地分配采样时间，采样间隔是 8 个 ADC 时钟周期与 6 个 ADC 时钟周期轮替，而不是均匀的 7 个 ADC 时钟周期。
图：交叉的单通道转换被注入序列 CH11 和 CH12 中断


采样
转换
ADC1
通道0
通道0
通道0
ADC 3
通道0
通道0
通道0
通道11
通道12
通道12
通道11
触发
通道0
通道0
通道0
通道0

8.5 特殊功能寄存器说明

8.5.1 ADC1/3 状态寄存器（ADC_SR）

ADC1/3 状态寄存器（ADC_SR）			基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	STRT	JSTRT	JEOC	EOC	AWD
Reset:	0	0	0	0	0	0	0	0

位	功能描述
STRT	**规则通道开始位（Regular channel Start flag）** 该位由硬件在规则通道转换开始时设置，由软件清除。 0：规则通道转换未开始； 1：规则通道转换已开始。
JSTRT	**注入通道开始位（Injected channel Start flag）** 该位由硬件在注入通道组转换开始时设置，由软件清除。 0：注入通道组转换未开始； 1：注入通道组转换已开始。
JEOC	**注入通道转换结束位（Injected channel end of conversion）** 该位由硬件在所有注入通道组转换结束时设置，由软件清除 0：转换未完成； 1：转换完成。
EOC	**转换结束位（End of conversion）** 该位由硬件在(规则或注入)通道组转换结束时设置，由软件清除或由读取 ADC_DR 时清除 0：转换未完成； 1：转换完成。
AWD	**模拟看门狗标志位（Analog watchdog flag）** 该位由硬件在转换的电压值超出了 ADC_LTR 和 ADC_HTR 寄存器定义的范围时设置，由软件清除

	0：没有发生模拟看门狗事件； 1：发生模拟看门狗事件。

8.5.2 ADC2 状态寄存器（ADC_SR）

ADC2 状态寄存器（ADC_SR）			基地址： 0x4001 2800(ADC2) 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	STRT	X	X	EOC	AWD
Reset:	0	0	0	0	0	0	0	0

位	功能描述
STRT	**规则通道开始位（Regular channel Start flag）** 该位由硬件在规则通道转换开始时设置，由软件清除。 0：规则通道转换未开始； 1：规则通道转换已开始。
EOC	**转换结束位（End of conversion）** 该位由硬件在(规则或注入)通道组转换结束时设置，由软件清除或由读取 ADC_DR 时清除 0：转换未完成； 1：转换完成。
AWD	**模拟看门狗标志位（Analog watchdog flag）** 该位由硬件在转换的电压值超出了 ADC_LTR 和 ADC_HTR 寄存器定义的范围时设置，由软件清除 0：没有发生模拟看门狗事件； 1：发生模拟看门狗事件。

8.5.3 ADC1/3 控制寄存器 1（ADC_CR1）

ADC1/3 控制寄存器 1（ADC_CR1）	基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x04

	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	AWDEN	JAWDEN	X	X	DUALMOD[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DISCNUM[15:13]			JDISCEN	DISCEN	XJAUTO	AWDSGL	SCAN
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
AWDEN	**在规则通道上开启模拟看门狗** (Analog watchdog enable on regular channels) 该位由软件设置和清除。 0：在规则通道上禁用模拟看门狗； 1：在规则通道上使用模拟看门狗。
JAWDEN	**在注入通道上开启模拟看门狗** (Analog watchdog enable on injected channels) 该位由软件设置和清除。 0：在注入通道上禁用模拟看门狗； 1：在注入通道上使用模拟看门狗。
DUALMOD[19:16]	**双模式选择** (Dual mode selection) 软件使用这些位选择操作模式。 0000：独立模式 0001：混合的同步规则+注入同步模式 0010：混合的同步规则+交替触发模式 0011：混合同步注入+快速交叉模式 0100：混合同步注入+慢速交叉模式 0101：注入同步模式 0110：规则同步模式 0111：快速交叉模式 1000：慢速交叉模式 1001：交替触发模式 注： 在 ADC2 和 ADC3 中这些位为保留位 在双模式中，改变通道的配置会产生一个重新开始的条件，这将导致同步丢失。建议在进行任何配置改变前关闭双模式。
DISCNUM[15:13]	**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
JDISCEN	**在注入通道上的间断模式（Discontinuous mode on injected channels）** 该位由软件设置和清除，用于开启或关闭注入通道组上的间断模式 0：注入通道组上禁用间断模式； 1：注入通道组上使用间断模式。
DISCEN	**在规则通道上的间断模式（Discontinuous mode on regular channels）** 该位由软件设置和清除，用于开启或关闭规则通道组上的间断模式 0：规则通道组上禁用间断模式； 1：规则通道组上使用间断模式。
JAUTO	**自动的注入通道组转换（Automatic Injected Group conversion）** 该位由软件设置和清除，用于开启或关闭规则通道组转换结束后自动的注入通道组转换 0：关闭自动的注入通道组转换； 1：开启自动的注入通道组转换。
AWDSGL	**扫描模式中在一个单一的通道上使用看门狗（Enable the watchdog on a single channel in scan mode）** 该位由软件设置和清除，用于开启或关闭由 AWDCH[4:0]位指定的通道上的模拟看门狗功能 0：在所有的通道上使用模拟看门狗； 1：在单一通道上使用模拟看门狗。
SCAN	**扫描模式（Scan mode）** 该位由软件设置和清除，用于开启或关闭扫描模式。在扫描模式中，转换由 ADC_SQRx 或 ADC_JSQRx 寄存器选中的通道。 0：关闭扫描模式； 1：使用扫描模式。 注：如果分别设置了 EOCIE 或 JEOCIE 位，只在最后一个通道转换完毕后才会产生 EOC 或 JEOC 中断。
JEOCIE	**允许产生注入通道转换结束中断（Interrupt enable for injected channels）** 该位由软件设置和清除，用于禁止或允许所有注入通道转换结束后产生中断。 0：禁止 JEOC 中断； 1：允许 JEOC 中断。当硬件设置 JEOC 位时产生中断。
AWDIE	**允许产生模拟看门狗中断（Analog watchdog interrupt enable）** 该位由软件设置和清除，用于禁止或允许模拟看门狗产生中断。在扫描模式下，如果看门狗检测到超范围的数值时，只有在设置了该位时扫描才会中止。 0：禁止模拟看门狗中断； 1：允许模拟看门狗中断。
EOCIE	**允许产生 EOC 中断（Interrupt enable for EOC）** 该位由软件设置和清除，用于禁止或允许转换结束后产生中断。 0：禁止 EOC 中断； 1：允许 EOC 中断。当硬件设置 EOC 位时产生中断。
AWDCH[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，用于选择模拟看门狗保护的输入通道。 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 …… 00101：ADC 模拟输入通道 5 00110：ADC 模拟输入通道 6 00111：ADC 模拟输入通道 7 仅 ADC1/3 有效

	01000：ADC 模拟输入通道 8 仅 ADC1/3 有效 01001：ADC 模拟输入通道 9 仅 ADC1/3 有效 保留所有其他数值。 注：ADC1 的模拟输入通道 9 在芯片内部连到了 PGA1。 ADC2 的模拟输入通道 6 在芯片内部连到了 PGA2。 ADC3 的模拟输入通道 8 连接 VREFINT，通道 9 连接 TPS。

8.5.4 ADC2 控制寄存器 1（ADC_CR1）

ADC2 控制寄存器 1（ADC_CR1）			基地址： 0x4001 2800(ADC2) 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	AWDEN	X	X	X	X			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DISCNUM[15:13]			X	DISCEN	X	AWDSGL	SCAN
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	AWDIE	EOCIE	AWDCH[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
AWDEN	**在规则通道上开启模拟看门狗** (Analog watchdog enable on regular channels) 该位由软件设置和清除。 0：在规则通道上禁用模拟看门狗； 1：在规则通道上使用模拟看门狗。
DISCNUM[15:13]	**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
DISCEN	**在规则通道上的间断模式** (Discontinuous mode on regular channels) 该位由软件设置和清除，用于开启或关闭规则通道组上的间断模式 0：规则通道组上禁用间断模式； 1：规则通道组上使用间断模式。
AWDSGL	**扫描模式中在一个单一的通道上使用看门狗** (Enable the watchdog on a single channel in scan mode)

	该位由软件设置和清除，用于开启或关闭由 AWDCH[4:0]位指定的通道上的模拟看门狗功能 0：在所有的通道上使用模拟看门狗； 1：在单一通道上使用模拟看门狗。
SCAN	**扫描模式（Scan mode）** 该位由软件设置和清除，用于开启或关闭扫描模式。在扫描模式中，转换由 ADC_SQRx 或 ADC_JSQRx 寄存器选中的通道。 0：关闭扫描模式； 1：使用扫描模式。 注：如果分别设置了 EOCIE 或 JEOCIE 位，只在最后一个通道转换完毕后才会产生 EOC 或 JEOC 中断。
AWDIE	**允许产生模拟看门狗中断（Analog watchdog interrupt enable）** 该位由软件设置和清除，用于禁止或允许模拟看门狗产生中断。在扫描模式下，如果看门狗检测到超范围的数值时，只有在设置了该位时扫描才会中止。 0：禁止模拟看门狗中断； 1：允许模拟看门狗中断。
EOCIE	**允许产生 EOC 中断（Interrupt enable for EOC）** 该位由软件设置和清除，用于禁止或允许转换结束后产生中断。 0：禁止 EOC 中断； 1：允许 EOC 中断。当硬件设置 EOC 位时产生中断。
AWDCH[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，用于选择模拟看门狗保护的输入通道。 00000：ADC 模拟输入通道 0 00001：ADC 模拟输入通道 1 …… 00101：ADC 模拟输入通道 5 00110：ADC 模拟输入通道 6 00111：ADC 模拟输入通道 7 仅 ADC1/3 有效 01000：ADC 模拟输入通道 8 仅 ADC1/3 有效 01001：ADC 模拟输入通道 9 仅 ADC1/3 有效 保留所有其他数值。 注：ADC1 的模拟输入通道 9 在芯片内部连到了 PGA1。 ADC2 的模拟输入通道 6 在芯片内部连到了 PGA2。 ADC3 的模拟输入通道 8 连接 VREFINT，通道 9 连接 TPS。

8.5.5 ADC1/3 控制寄存器 2(ADC_CR2)

ADC1/3 控制寄存器 2（ADC_CR2）			基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X				X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	TSVREFE	SWSTART	JSWSTART	EXTTRIG	EXTSEL[19:16]			

Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	JEXTTRIG	JEXTSEL[14:11]				ALIGN	X	DMA
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	X	X	X	X	RSTCAL	CAL	CONT	ADON
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TSVREFE	**温度传感器和VREFINT使能(Temperature sensor and VREFINT enable)** 该位由软件设置和清除，用于开启或禁止温度传感器和VREFINT通道。在多于1个ADC的器件中，该位仅出现在ADC1中。 0：禁止温度传感器和VREFINT； 1：启用温度传感器和 VREFINT。
SWSTART	**开始转换规则通道(Start conversion of regular channels)** 由软件设置该位以启动转换，转换开始后硬件马上清除此位。如果在EXTSEL位中选择了SWSTART为触发事件，该位用于启动一组规则通道的转换。 0：复位状态； 1：开始转换规则通道。
JSWSTART	**开始转换注入通道(Start conversion of injected channels)** 由软件设置该位以启动转换，软件可清除此位或在转换开始后硬件马上清除此位。如果在JEXTSEL位中选择了JSWSTART为触发事件，该位用于启动一组注入通道的转换， 0：复位状态； 1：开始转换注入通道。
EXTTRIG	**规则通道的外部触发转换模式 (External trigger conversion mode for regular channels)** 该位由软件设置和清除，用于开启或禁止可以启动规则通道组转换的外部触发事件。 0：不用外部事件启动转换； 1：使用外部事件启动转换。
EXTSEL[19:16]	**选择启动规则通道组转换的外部事件 (External event select for regular group)** 这些位选择用于启动规则通道组转换的外部事件 ADC1/2/3的触发配置如下： 0000：定时器1的CC1事件 0001：定时器1的CC2事件 0010：定时器2的CC2事件 0011：定时器3的CC1事件 0100：定时器3的TRGO事件 0101：定时器4的CC4事件 0110：定时器8的TRGO事件 0111：SWSTART 1000：EXTI线11 1001：定时器20的CC1事件

	1010：定时器20的CC2事件 1011：定时器20的CC3事件 1100：定时器20的CC4事件 1101：定时器20的CC5事件 1110：定时器20的TRGO事件
JEXTTRIG	**注入通道的外部触发转换模式 (External trigger conversion mode for injected channels)** 该位由软件设置和清除，用于开启或禁止可以启动注入通道组转换的外部触发事件。 0：不用外部事件启动转换； 1：使用外部事件启动转换。
JEXTSEL[14:11]	**选择启动注入通道组转换的外部事件 (External event select for injected group)** 这些位选择用于启动注入通道组转换的外部事件。 ADC1/2/3的触发配置如下： 0000：定时器1的TRGO事件 0001：定时器1的CC4事件 0010：定时器2的TRGO事件 0011：定时器2的CC1事件 0100：定时器3的CC4事件 0101：定时器4的TRGO事件 0110：EXTI线15 0111：JSWSTART 1000：定时器1的CC5事件 1001：定时器1的CC4和CC5事件 1010：定时器8的TRGO事件 1011：定时器8的CC5事件 1100：定时器8的CC4和CC5事件 1101：定时器20的TRGO事件 1110：定时器20的CC4事件 1111：定时器20的CC4和CC5事件
ALIGN	**数据对齐(Data alignment)** 该位由软件设置和清除。 0：右对齐； 1：左对齐。
DMA	**直接存储器访问模式(Direct memory access mode)** 该位由软件设置和清除。 0：不使用DMA模式； 1：使用DMA模式。 注：ADC1/2/3 均能产生 DMA 请求。
RSTCAL	**复位校准(Reset calibration)** 该位由软件设置并由硬件清除。在校准寄存器被初始化后该位将被清除。 0：校准寄存器已初始化； 1：初始化校准寄存器。

	注：如果正在进行转换时设置 RSTCAL，清除校准寄存器需要额外的周期。
CAL	**A/D校准**(A/D Calibration) 该位由软件设置以开始校准，并在校准结束时由硬件清除。 0：校准完成； 1：开始校准。
CONT	**连续转换**(Continuous conversion) 该位由软件设置和清除。如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式； 1：连续转换模式。
ADON	**开/关 A/D 转换器**(A/D converter ON / OFF) 该位由软件设置和清除。当该位为’0’时，写入’1’将把 ADC 从断电模式下唤醒。 当该位为’1’时，写入’1’将启动转换。应用程序需注意，在转换器上电至转换开始有一个延迟 tSTAB。 0：关闭 ADC 转换/校准，并进入断电模式； 1：开启 ADC 并启动转换。 注：如果在这个寄存器中与 ADON 一起还有其他位被改变，则转换不被触发。这是为了防止触 发错误的转换。

8.5.6 ADC2 控制寄存器 2（ADC_CR2）

ADC2 控制寄存器 2（ADC_CR2）			基地址：0x4001 2800(ADC2) 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X				X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	TSVREFE	SWSTART	X	EXTTRIG	EXTSEL[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	ALIGN	X	DMA
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	RSTCAL	CAL	CONT	ADON
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TSVREFE	**温度传感器和VREFINT使能**(Temperature sensor and VREFINT enable)

	该位由软件设置和清除，用于开启或禁止温度传感器和VREFINT通道。在多于1个ADC的器件中，该位仅出现在ADC1中。 0：禁止温度传感器和VREFINT； 1：启用温度传感器和 VREFINT。
SWSTART	**开始转换规则通道**(Start conversion of regular channels) 由软件设置该位以启动转换，转换开始后硬件马上清除此位。如果在EXTSEL位中选择了SWSTART为触发事件，该位用于启动一组规则通道的转换。 0：复位状态； 1：开始转换规则通道。
EXTTRIG	**规则通道的外部触发转换模式** (External trigger conversion mode for regular channels) 该位由软件设置和清除，用于开启或禁止可以启动规则通道组转换的外部触发事件。 0：不用外部事件启动转换； 1：使用外部事件启动转换。
EXTSEL[19:16]	**选择启动规则通道组转换的外部事件** (External event select for regular group) 这些位选择用于启动规则通道组转换的外部事件 ADC1/2/3的触发配置如下： 0000：定时器1的CC1事件 0001：定时器1的CC2事件 0010：定时器2的CC2事件 0011：定时器3的CC1事件 0100：定时器3的TRGO事件 0101：定时器4的CC4事件 0110：定时器8的TRGO事件 0111：SWSTART 1000：EXTI线11 1001：定时器20的CC1事件 1010：定时器20的CC2事件 1011：定时器20的CC3事件 1100：定时器20的CC4事件 1101：定时器20的CC5事件 1110：定时器20的TRGO事件
ALIGN	**数据对齐**(Data alignment) 该位由软件设置和清除。 0：右对齐； 1：左对齐。
DMA	**直接存储器访问模式**(Direct memory access mode) 该位由软件设置和清除。 0：不使用DMA模式； 1：使用DMA模式。 注：ADC1/2/3 均能产生 DMA 请求。
RSTCAL	**复位校准**(Reset calibration) 该位由软件设置并由硬件清除。在校准寄存器被初始化后该位将被清除。

	0：校准寄存器已初始化； 1：初始化校准寄存器。 注：如果正在进行转换时设置 RSTCAL，清除校准寄存器需要额外的周期。
CAL	A/D校准(A/D Calibration) 该位由软件设置以开始校准，并在校准结束时由硬件清除。 0：校准完成； 1：开始校准。
CONT	连续转换(Continuous conversion) 该位由软件设置和清除。如果设置了此位，则转换将连续进行直到该位被清除。 0：单次转换模式； 1：连续转换模式。
ADON	开/关 A/D 转换器(A/D converter ON / OFF) 该位由软件设置和清除。当该位为’0’时，写入’1’将把 ADC 从断电模式下唤醒。 当该位为’1’时，写入’1’将启动转换。应用程序需注意，在转换器上电至转换开始有一个延迟 tSTAB。 0：关闭 ADC 转换/校准，并进入断电模式； 1：开启 ADC 并启动转换。 注：如果在这个寄存器中与 ADON 一起还有其他位被改变，则转换不被触发。这是为了防止触发错误的转换。

8.5.7 ADC 采样时间寄存器 1(ADC_SMPR1)

ADC 采样时间寄存器 1(ADC_SMPR1)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	SMP9[29:27]			SMP8[26:24]		
Reset:	0	0	0	0	0	0	0	0

	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	SMP7[23:21]			SMP6[20:18]			SMP5[17:16]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	SMP5[15]	SMP4[14:12]			SMP3[11:9]			SMP2[8]
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	SMP2[7:6]		SMP1[5:3]			SMP0[2:0]		
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SMPx[2:0]	**选择通道x的采样时间(Channel x Sample time selection)(0-9)** 这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。 000：1.5周期　　100：41.5周期 001：7.5周期　　101：55.5周期 010：13.5周期　　110：71.5周期 011：28.5周期　　111：239.5周期 注：ADC1支持SMPR0-9，ADC2仅支持SMPR0-6，ADC3支持SMPR0-9 ADC1的模拟输入通道9在芯片内部连到了PGA1。 ADC2的模拟输入通道6在芯片内部连到了PGA2。 ADC3 的模拟输入通道 8 连接 VREFINT，通道 9 连接 TPS。

8.5.8 ADC 注入通道数据偏移寄存器 x (ADC_JOFRx)(x=1..4)

ADC 注入通道数据偏移寄存器 x (ADC_JOFRx)(x=1..4)			基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x14 0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	JOFRx[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read:	JOFRx[7:0]							

Write:								
Reset:	0	0	0	0	0	0	0	0

位	功能描述
JOFRx[11:0] (x=1..4)	**注入通道 x 的数据偏移 (Data offset for injected channel x)** 当转换注入通道时，这些位定义了用于从原始转换数据中减去的数值。转换的结果可以在 ADC_JDRx 寄存器中读出。

8.5.9 ADC 看门狗高阀值寄存器(ADC_HTR)

ADC 看门狗高阀值寄存器 (ADC_HTR)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x24					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	HTR[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	HTR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
HTR[11:0]	**模拟看门狗高阀值 (Analog watchdog high threshold)** 这些位定义了模拟看门狗的阀值高限。

8.5.10 ADC 看门狗低阀值寄存器(ADC_LRT)

ADC 看门狗低阀值寄存器 (ADC_LRT)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x28					
	Bit31	30	29	28	27	26	25	Bit24
Read:	X	X	X	X	X	X	X	X

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	LTR[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	LTR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LTR[11:0]	**模拟看门狗低阀值 (Analog watchdog low threshold)** 这些位定义了模拟看门狗的阀值低限。

8.5.11 ADC 规则序列寄存器 1(ADC_SQR1)

ADC 规则序列寄存器 1(ADC_SQR1)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x2C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	L[22:20]			X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述

L[22:20]	**规则通道序列长度（Regular channel sequence length）** 这些位由软件定义在规则通道转换序列中的通道数目。 000：1 个转换 001：2 个转换 …… 111：8 个转换

8.5.12 ADC 规则序列寄存器 2(ADC_SQR2)

ADC 规则序列寄存器 2(ADC_SQR2)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x30					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	SQR2_SQ8[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	SQR2_SQ8[7:5]			SQR2_SQ7[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SQR2_SQ8[9:5]	**规则序列中的第 8 个转换（8th conversion in regular sequence）** 注：这些位由软件定义转换序列中的第 8 个转换通道的编号 ADC1(channel 0~9)，ADC2(channel 0~6)，ADC3（channel 0-9） 保留所有其他数值。
SQR2_SQ7[4:0]	**规则序列中的第 7 个转换（7th conversion in regular sequence）**

8.5.13 ADC 规则序列寄存器 3(ADC_SQR3)

ADC 规则序列寄存器 3(ADC_SQR3)	基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3)

			偏移地址： 0x34					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	SQ6[29:25]					SQ5[24]
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	SQ5[23:20]				SQ4[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	SQ4[15]	SQ3[14:10]					SQ2[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	SQ2[7:5]			SQ1[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SQ6[29:25]	规则序列中的第 6 个转换（6th conversion in regular sequence）
SQ5[24:20]	规则序列中的第 5 个转换（5th conversion in regular sequence）
SQ4[19:15]	规则序列中的第 4 个转换（4th conversion in regular sequence）
SQ3[14:10]	规则序列中的第 3 个转换（3rd conversion in regular sequence）
SQ2[9:5]	规则序列中的第 2 个转换（2nd conversion in regular sequence）
SQ1[4:0]	规则序列中的第 1 个转换（1st conversion in regular sequence）

8.5.14 ADC 注入序列寄存器(ADC_JSQR)

ADC 注入序列寄存器 (ADC_JSQR)			基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x38					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	JL[21:20]		JSQ4[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	JSQ4[15]	JSQ3[14:10]					JSQ2[9:8]	
Reset:	0	0	0	0	0	0	0	0

	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	JSQ2[7:5]			JSQ1[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
JL[21:20]	**注入通道序列长度 (Injected sequence length)** 这些位由软件定义在规则通道转换序列中的通道数目。 00：1 个转换 01：2 个转换 10：3 个转换 11：4 个转换
JSQ4[19:15]	**注入序列中的第 4 个转换 (4th conversion in injected sequence)** 这些位由软件定义转换序列中的第 4 个转换通道的编号 ADC1(channel0~9)，ADC2(channel0~6)，ADC3（channel0-9） 保留所有其他数值。 注：不同于规则转换序列，如果 JL[1:0]的长度小于 4，则转换的序列顺序是从(4-JL)开始。例如：ADC_JSQR[21:0] = 10 00011 00011 00111 00010，意味着扫描转换将按下列通道顺序转换：7、3、3，而不是 2、7、3。
JSQ3[14:10]	**注入序列中的第 3 个转换 (3rd conversion in regular sequence)**
JSQ2[9:5]	**注入序列中的第 2 个转换 (2nd conversion in regular sequence)**
JSQ1[4:0]	**注入序列中的第 1 个转换 (1st conversion in regular sequence)**

8.5.15 ADC 注入数据寄存器 x (ADC_JDRx) (x= 1..4)

ADC 注入数据寄存器 x (ADC_JDRx) (x= 1..4)			基地址： 0x4001 2400(ADC1) 0x4001 3C00(ADC3) 偏移地址： 0x3C 0x48					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	JDRx[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	JDRx[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
JDRx[15:0] (x= 1..4)	注入转换的数据 (Injected data) 这些位为只读，包含了注入通道的转换结果。数据是左对齐或右对齐，

8.5.16 ADC 规则数据寄存器(ADC_DR)

ADC 规则数据寄存器 (ADC_DR)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x4C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	ADC2DATA[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	ADC2DATA[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DATA[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DATA[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ADC2DATA[31:16]	ADC2 转换的数据 (ADC2 data) - 在 ADC1 中：双模式下，这些位包含了 ADC2 转换的规则通道数据。 - 在 ADC2 和 ADC3 中：不使用这些位。
DATA[15:0]	规则转换的数据 (Regular data) 这些位为只读，包含了规则通道的转换结果。数据是左对齐或右对齐

8.5.17 ADC 校准寄存器(ADC_CAL)

ADC 校准寄存器(ADC_CAL)			基地址： 0x4001 2400(ADC1) 0x4001 2800(ADC2) 0x4001 3C00(ADC3) 偏移地址： 0x54					
	Bit31	30	29	28	27	26	25	Bit24

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	X	CAL_DATA						
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CAL_DATA[6:0]	**ADC 自校准的值写入该寄存器** 可以手动写入值，在 ADC 转换中生效。

9 SPI 串行外设接口

9.1 SPI 简介

串行外设接口(SPI)允许芯片与外部设备以半/全双工、同步、串行方式通信。此接口可以被配置成主模式，并为外部从设备提供通信时钟(SCK)。接口还能以多主配置方式工作。

它可用于多种用途，包括使用一条双向数据线的双线单工同步传输，还可使用 CRC 校验的可靠通信。

注：由于 SPI1 的部分引脚与 JTAG 引脚共享(SPI1_NSS 与 JTDI ，SPI1_SCK 与 JTDO，SPI1_,MISO 与 JNTRST)，因此这些引脚不受 IO 控制器控制，他们(在每次复位后)被默认保留为 JTAG 用途。如果用户想把引脚配置给 SPI1，必须(在调试时)关闭 JTAG 并切换至 SWD 接口，或者(在标准应用时)同时关闭 JTAG 和 SWD 接口。

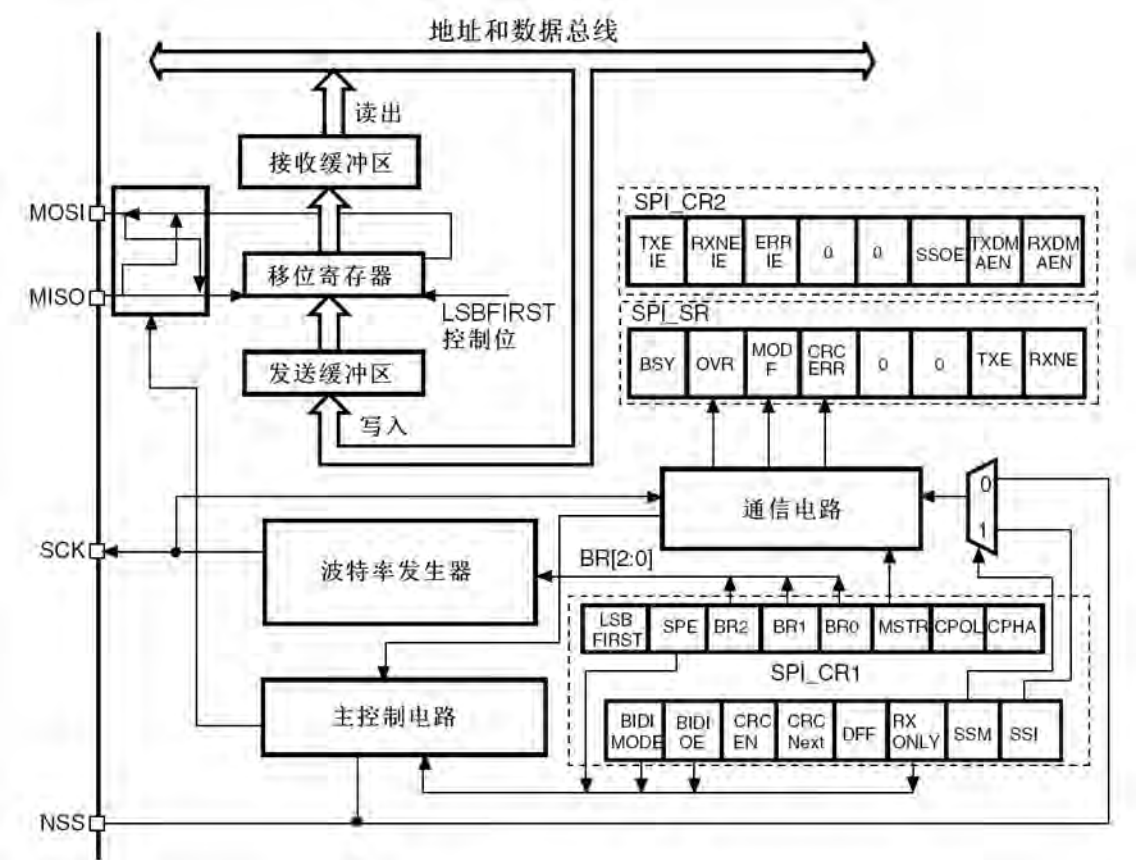
9.2 SPI 特征

●3 线全双工同步传输
●带或不带第三根双向数据线的双线单工同步传输
●8 或 16 位传输帧格式选择
●主或从操作
●支持多主模式
●8 个主模式波特率预分频系数(最大为 fPCLK/2)
●从模式频率 (最大为 fPCLK/2)
●主模式和从模式的快速通信
●主模式和从模式下均可以由软件或硬件进行 NSS 管理：主/从操作模式的动态改变
●可编程的时钟极性和相位
●可编程的数据顺序，MSB 在前或 LSB 在前
●可触发中断的专用发送和接收标志
●SPI 总线忙状态标志
●支持可靠通信的硬件 CRC
—在发送模式下，CRC 值可以被作为最后一个字节发送
—在全双工模式中对接收到的最后一个字节自动进行 CRC 校验
●可触发中断的主模式故障、过载以及 CRC 错误标志
●支持 DMA 功能的 1 字节发送和接收缓冲器：产生发送和接受请求

9.3 SPI 功能描述

9.3.1 概述

SPI 的方框图见下图。
图：SPI 框图



通常 SPI 通过 4 个引脚与外部器件相连：

●MISO：主设备输入/从设备输出引脚。该引脚在从模式下发送数据，在主模式下接收数据。

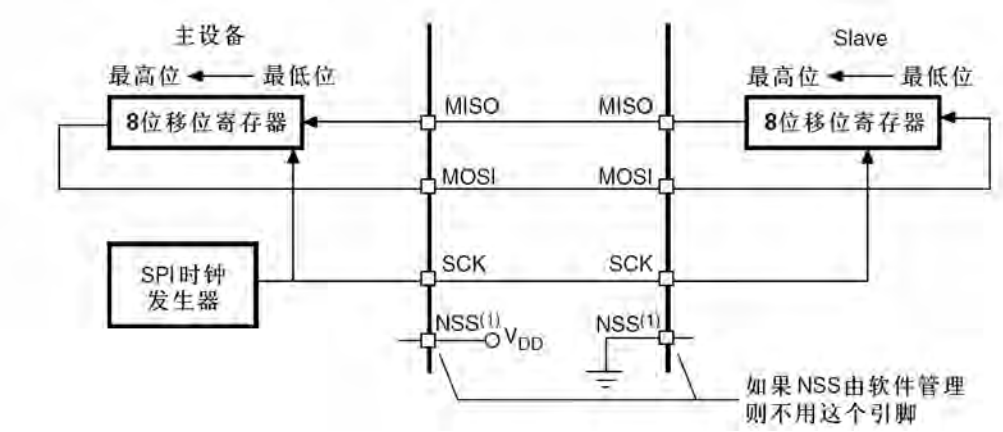
●MOSI：主设备输出/从设备输入引脚。该引脚在主模式下发送数据，在从模式下接收数据。

●SCK：串口时钟，作为主设备的输出，从设备的输入

●NSS：从设备选择。这是一个可选的引脚，用来选择主/从设备。它的功能是用来作为“片选引脚”，让主设备可以单独地与特定从设备通讯，避免数据线上的冲突。从设备的 NSS 引脚可以由主设备的一个标准 I/O 引脚来驱动。一旦被使能(SSOE 位)，NSS 引脚也可以作为输出引脚，并在 SPI 处于主模式时拉低；此时，所有的 SPI 设备，如果它们的 NSS 引脚连接到主设备的 NSS 引脚，则会检测到低电平，如果它们被设置为 NSS 硬件模式，就会自动进入从设备状态。当配置为主设备、NSS 配置为输入引脚(MSTR=1，SSOE=0)时，如果 NSS 被拉低，则这个 SPI 设备进入主模式失败状态：即 MSTR 位被自动清除，此设备进入从模式。

下图是一个单主和单从设备互连的例子。

图: 单主和单从应用



1.这里 NSS 引脚设置为输入

MOSI 脚相互连接，MISO 脚相互连接。这样，数据在主和从之间串行地传输(MSB 位在前)。

通信总是由主设备发起。主设备通过 MOSI 脚把数据发送给从设备，从设备通过 MISO 引脚回传数据。这意味全双工通信的数据输出和数据输入是用同一个时钟信号同步的；时钟信号由主设备通过 SCK 脚提供。

从选择(NSS)脚管理

有 2 种 NSS 模式：

●软件 NSS 模式：可以通过设置 SPI_CR1 寄存器的 SSM 位来使能这种模式(见图 211)。在这种模式下 NSS 引脚可以用作它用，而内部 NSS 信号电平可以通过写 SPI_CR1 的 SSI 位来驱动

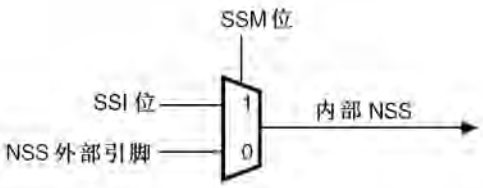
●硬件 NSS 模式，分两种情况：

—NSS 输出被使能：当工作为主 SPI，并且 NSS 输出已经通过 SPI_CR2 寄存器的 SSOE 位使能，这时 NSS 引脚被拉低，所有 NSS 引脚与这个主 SPI 的 NSS 引脚相连并配置为硬件 NSS 的 SPI 设备，将自动变成从 SPI 设备。

当一个 SPI 设备需要发送广播数据，它必须拉低 NSS 信号，以通知所有其它的设备它是主设备；如果它不能拉低 NSS，这意味着总线上有另外一个主设备在通信，这时将产生一个硬件失败错误(Hard Fault)。

—NSS 输出被关闭：允许操作于多主环境。

图：硬件/软件的从选择管理



时钟信号的相位和极性

SPI_CR 寄存器的 CPOL 和 CPHA 位，能够组合成四种可能的时序关系。CPOL(时钟极性)位控制在没有数据传输时时钟的空闲状态电平，此位对主模式和从模式下的设备都有效。如果 CPOL 被清’0’，SCK 引脚在空闲状态保持低电平；如果 CPOL 被置’1’，SCK 引脚在空闲状态保持高电平。

如果 CPHA(时钟相位)位被置’1’，SCK 时钟的第二个边沿(CPOL 位为 0 时就是下降沿，CPOL 位为’1’时就是上升沿)进行数据位的采样，数据在第二个时钟边沿被锁存。如果 CPHA 位被清’0’，SCK 时钟的第一边沿(CPOL 位为’0’时就是上升沿，CPOL 位为’1’时就是下降沿)进行数据位采样，数据在第一个时钟边沿被锁存。

CPOL 时钟极性和 CPHA 时钟相位的组合选择数据捕捉的时钟边沿。

图 212 显示了 SPI 传输的 4 种 CPHA 和 CPOL 位组合。此图可以解释为主设备和从设备的 SCK 脚、MISO 脚、MOSI 脚直接连接的主或从时序图。

注意： 1. 在改变 CPOL/CPHA 位之前，必须清除 SPE 位将 SPI 禁止。

2.主和从必须配置成相同的时序模式。

3. SCK 的空闲状态必须和 SPI_CR1 寄存器指定的极性一致(CPOL 为’1’时，空闲时应上拉 SCK 为高电平；CPOL 为’0’时，空闲时应下拉 SCK 为低电平)。
4.数据帧格式(8 位或 16 位)由 SPI_CR1 寄存器的 DFF 位选择，并且决定发送/接收的数据长度。

9.3.2 配置 SPI 为从模式

在从模式下，SCK 引脚用于接收从主设备来的串行时钟。SPI_CR1 寄存器中 BR[2:0]的设置不影响数据传输速率。
注：建议在主设备发送时钟之前使能 SPI 从设备，否则可能会发生意外的数据传输。在通信时钟的第一个边沿到来之前或正在进行的通信结束之前，从设备的数据寄存器必须就绪。在使能从设备和主设备之前，通信时钟的极性必须处于稳定的数值。
请按照以下步骤配置 SPI 为从模式：
配置步骤
1.设置 DFF 位以定义数据帧格式为 8 位或 16 位。
2.选择 CPOL 和 CPHA 位来定义数据传输和串行时钟之间的相位关系(见图 212)。为保证正确的数据传输，从设备和主设备的 CPOL 和 CPHA 位必须配置成相同的方式。
3.帧格式(SPI_CR1 寄存器中的 LSBFIRST 位定义的”MSB 在前”还是”LSB 在前”)必须与主设备相同。
4.硬件模式下(参考从选择(NSS)脚管理部分)，在完整的数据帧(8 位或 16 位)传输过程中，NSS 引脚必须为低电平。在 NSS 软件模式下，设置 SPI_CR1 寄存器中的 SSM 位并清除 SSI 位。
5.清除 MSTR 位、设置 SPE 位(SPI_CR1 寄存器)，使相应引脚工作于 SPI 模式下。在这个配置中 MOSI 引脚是数据输入，MISO 引脚是数据输出。
数据发送过程
在写操作中，数据字被并行地写入发送缓冲器。
当从设备收到时钟信号，并且在 MOSI 引脚上出现第一个数据位时，发送过程开始(译注：此时第一个位被发送出去)。余下的位(对于 8 位数据帧格式，还有 7 位；对于 16 位数据帧格式，还有
15 位)被装进移位寄存器。当发送缓冲器中的数据传输到移位寄存器时，SPI_SP 寄存器的 TXE
标志被设置，如果设置了 SPI_CR2 寄存器的 TXEIE 位，将会产生中断。
数据接收过程
对于接收器，当数据接收完成时：
●移位寄存器中的数据传送到接收缓冲器，SPI_SR 寄存器中的 RXNE 标志被设置。
●如果设置了 SPI_CR2 寄存器中的 RXNEIE 位，则产生中断。
在最后一个采样时钟边沿后，RXNE 位被置’1’，移位寄存器中接收到的数据字节被传送到接收缓冲器。
当读 SPI_DR 寄存器时，SPI 设备返回这个接收缓冲器的数值。
读 SPI_DR 寄存器时，RXNE 位被清除。

9.3.3 配置 SPI 为主模式

在主配置时，在 SCK 脚产生串行时钟。
配置步骤
1.通过 SPI_CR1 寄存器的 BR[2:0]位定义串行时钟波特率。
2.选择 CPOL 和 CPHA 位，定义数据传输和串行时钟间的相位关系(见图 212)。
3.设置 DFF 位来定义 8 位或 16 位数据帧格式。
4.配置 SPI_CR1 寄存器的 LSBFIRST 位定义帧格式。
5.如果需要 NSS 引脚工作在输入模式，硬件模式下，在整个数据帧传输期间应把 NSS 脚连接到高电平；在软件模式下，需设置 SPI_CR1 寄存器的 SSM 位和 SSI 位。如果 NSS 引脚工作在输出模式，则只需设置 SSOE 位。
6.必须设置 MSTR 位和 SPE 位(只当 NSS 脚被连到高电平，这些位才能保持置位)。在这个配置中，MOSI 引脚是数据输出，而 MISO 引脚是数据输入。

数据发送过程

当写入数据至发送缓冲器时，发送过程开始。

在发送第一个数据位时，数据字被并行地(通过内部总线)传入移位寄存器，而后串行地移出到 MOSI 脚上；MSB 在先还是 LSB 在先，取决于 SPI_CR1 寄存器中的 LSBFIRST 位的设置。数据从发送缓冲器传输到移位寄存器时 TXE 标志将被置位，如果设置了 SPI_CR1 寄存器中的 TXEIE 位，将产生中断。

数据接收过程

对于接收器来说，当数据传输完成时：

●传送移位寄存器里的数据到接收缓冲器，并且 RXNE 标志被置位。

●如果设置了 SPI_CR2 寄存器中的 RXNEIE 位，则产生中断。

在最后采样时钟沿，RXNE 位被设置，在移位寄存器中接收到的数据字被传送到接收缓冲器。读 SPI_DR 寄存器时，SPI 设备返回接收缓冲器中的数据。读 SPI_DR 寄存器将清除 RXNE 位。

一旦传输开始，如果下一个将发送的数据被放进了发送缓冲器，就可以维持一个连续的传输流。在试图写发送缓冲器之前，需确认 TXE 标志应该为’1’。

注：在 NSS 硬件模式下，从设备的 NSS 输入由 NSS 引脚控制或另一个由软件驱动的 GPIO 引脚控制。

9.3.4 配置 SPI 为单工通信

SPI 模块能够以两种配置工作于单工方式：

●1 条时钟线和 1 条双向数据线；

●1 条时钟线和 1 条数据线(只接收或只发送)；

1 条时钟线和 1 条双向数据线(BIDIMODE=1)

设置 SPI_CR1 寄存器中的 BIDIMODE 位而启用此模式。在这个模式下，SCK 引脚作为时钟，主设备使用 MOSI 引脚而从设备使用 MISO 引脚作为数据通信。传输的方向由 SPI_CR1 寄存器里的 BIDIOE 控制，当这个位是’1’的时候，数据线是输出，否则是输入。

1 条时钟和 1 条单向数据线(BIDIMODE=0)

在这个模式下，SPI 模块可以或者作为只发送，或者作为只接收。

●只发送模式类似于全双工模式(BIDIMODE=0，RXONLY=0)：数据在发送引脚(主模式时是 MOSI、从模式时是 MISO)上传输，而接收引脚(主模式时是 MISO、从模式时是 MOSI)可以作为通用的 I/O 使用。此时，软件不必理会接收缓冲器中的数据(如果读出数据寄存器，它不包含任何接收数据)。

●在只接收模式，可以通过设置 SPI_CR2 寄存器的 RXONLY 位而关闭 SPI 的输出功能；此时， 发送引脚(主模式时是 MOSI、从模式时是 MISO)被释放，可以作为其它功能使用。

配置并使能 SPI 模块为只接收模式的方式是：

●在主模式时，一旦使能 SPI，通信立即启动，当清除 SPE 位时立即停止当前的接收。在此模式下，不必读取 BSY 标志，在 SPI 通信期间这个标志始终为’1’。

●在从模式时，只要 NSS 被拉低(或在 NSS 软件模式时，SSI 位为’0’)同时 SCK 有时钟脉冲，SPI 就一直在接收。

9.3.5 数据发送与接收过程

接收与发送缓冲器

在接收时，接收到的数据被存放在一个内部的接收缓冲器中；在发送时，在被发送之前，数据将首先被存放在一个内部的发送缓冲器中。

对 SPI_DR 寄存器的读操作，将返回接收缓冲器的内容；写入 SPI_DR 寄存器的数据将被写入发送缓冲器中。

主模式下开始传输

●全双工模式(BIDIMODE=0 并且 RXONLY=0)

一当写入数据到 SPI_DR 寄存器(发送缓冲器)后，传输开始；

—在传送第一位数据的同时，数据被并行地从发送缓冲器传送到 8 位的移位寄存器中， 然后按顺序被串行地移位送到 MOSI 引脚上；
—与此同时，在 MISO 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI_DR 寄存器(接收缓冲器)中。
●单向的只接收模式(BIDIMODE=0 并且 RXONLY=1)
—SPE=1 时，传输开始；
—只有接收器被激活，在 MISO 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI_DR 寄存器(接收缓冲器)中。
●双向模式，发送时(BIDIMODE=1 并且 BIDIOE=1)
—当写入数据到 SPI_DR 寄存器(发送缓冲器)后，传输开始；
—在传送第一位数据的同时，数据被并行地从发送缓冲器传送到 8 位的移位寄存器中， 然后按顺序被串行地移位送到 MOSI 引脚上；
—不接收数据。
●双向模式，接收时(BIDIMODE=1 并且 BIDIOE=0)
—SPE=1 并且 BIDIOE=0 时，传输开始；
—在 MOSI 引脚上接收到的数据，按顺序被串行地移位进入 8 位的移位寄存器中，然后被并行地传送到 SPI_DR 寄存器(接收缓冲器)中。
—不激活发送器，没有数据被串行地送到 MOSI 引脚上。

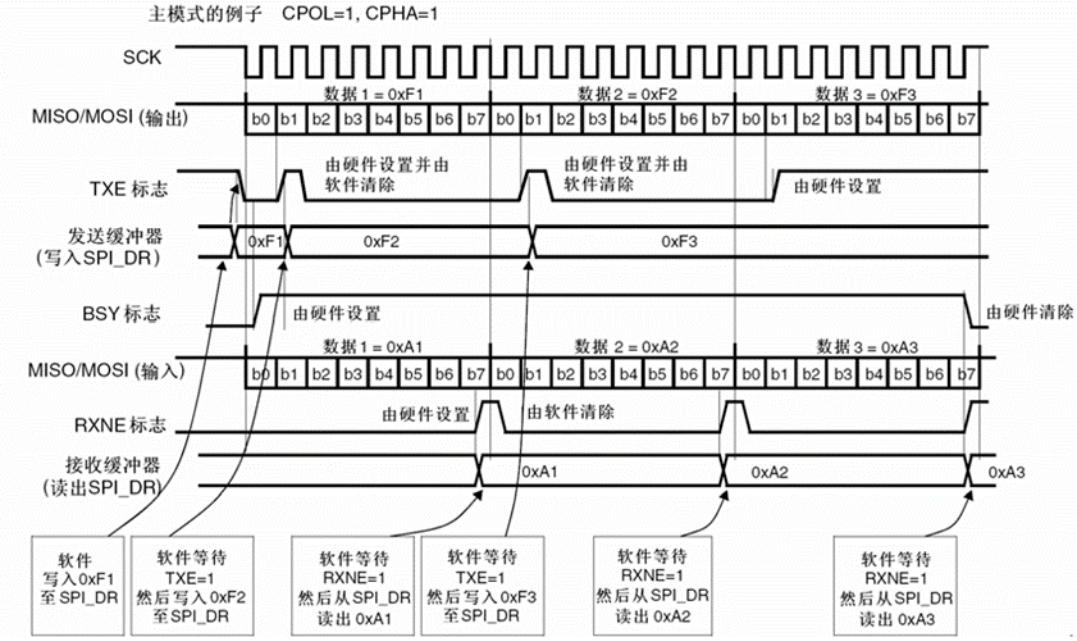
从模式下开始传输

●全双工模式(BIDIMODE=0 并且 RXONLY=0)
—当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始，随后的数据位依次移动进入移位寄存器；
—与此同时，在传输第一个数据位时，发送缓冲器中的数据被并行地传送到 8 位的移位寄存器，随后被串行地发送到 MISO 引脚上。软件必须保证在 SPI 主设备开始数据传输之前在发送寄存器中写入要发送的数据。
●单向的只接收模式(BIDIMODE=0 并且 RXONLY=1)
—当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始，随后数据位依次移动进入移位寄存器；
—不启动发送器，没有数据被串行地传送到 MISO 引脚上。
●双向模式，发送时(BIDIMODE=1 并且 BIDIOE=1)
—当从设备接收到时钟信号并且发送缓冲器中的第一个数据位被传送到 MISO 引脚上的时候，数据传输开始；
—在第一个数据位被传送到 MISO 引脚上的同时，发送缓冲器中要发送的数据被平行地传送到 8 位的移位寄存器中，随后被串行地发送到 MISO 引脚上。软件必须保证在 SPI 主设备开始数据传输之前在发送寄存器中写入要发送的数据；
—不接收数据。
●双向模式，接收时(BIDIMODE=1 并且 BIDIOE=0)
—当从设备接收到时钟信号并且第一个数据位出现在它的 MOSI 时，数据传输开始；
—从 MISO 引脚上接收到的数据被串行地传送到 8 位的移位寄存器中，然后被平行地传送到 SPI_DR 寄存器(接收缓冲器)；
—不启动发送器，没有数据被串行地传送到 MISO 引脚上。

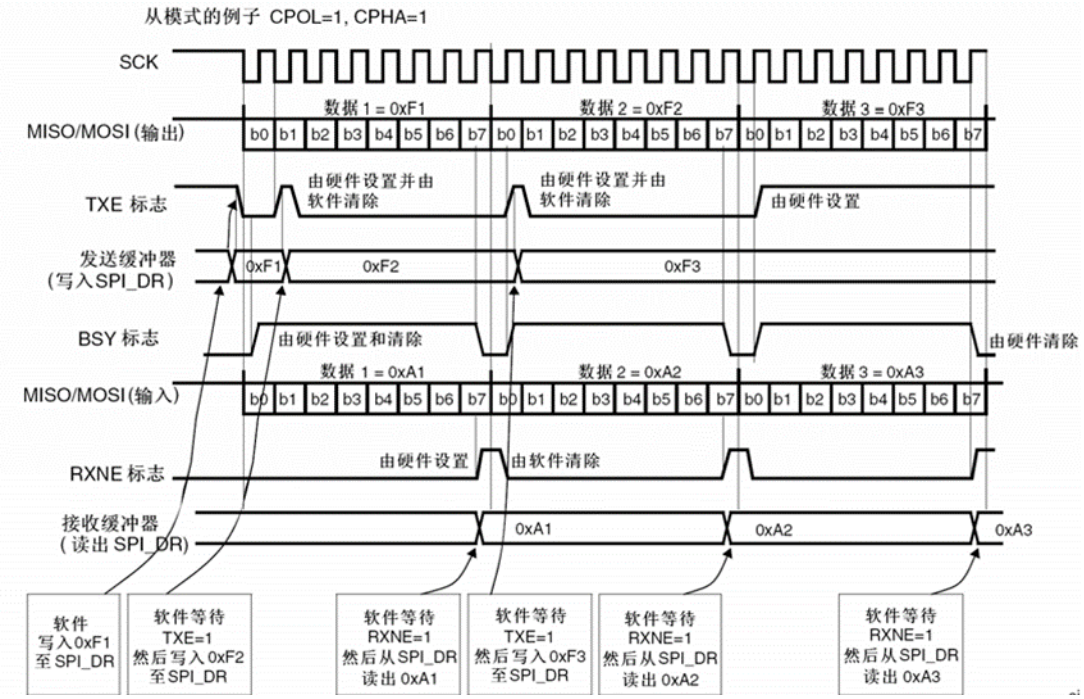
处理数据的发送与接收

当数据从发送缓冲器传送到移位寄存器时，设置 TXE 标志(发送缓冲器空)，它表示内部的发送缓冲器可以接收下一个数据；如果在 SPI_CR2 寄存器中设置了 TXEIE 位，则此时会产生一个中断；写入 SPI_DR 寄存器即可清除 TXE 位。
注：在写入发送缓冲器之前，软件必须确认 TXE 标志为’1’，否则新的数据会覆盖已经在发送缓冲器中的数据。

在采样时钟的最后一个边沿，当数据被从移位寄存器传送到接收缓冲器时，设置 RXNE 标志(接收缓冲器非空)；它表示数据已经就绪，可以从 SPI_DR 寄存器读出；如果在 SPI_CR2 寄存器中设置了 RXNEIE 位，则此时会产生一个中断；读出 SPI_DR 寄存器即可清除 RXNIE 标志位。
在一些配置中，传输最后一个数据时，可以使用 BSY 标志等待数据传输的结束。
主或从模式下(BIDIMODE=0 并且 RXONLY=0)全双工发送和接收过程模式
软件必须遵循下述过程，发送和接收数据：
1.设置 SPE 位为’1’，使能 SPI 模块；
2.在 SPI_DR 寄存器中写入第一个要发送的数据，这个操作会清除 TXE 标志；
3.等待 TXE=1，然后写入第二个要发送的数据。等待 RXNE=1，然后读出 SPI_DR 寄存器并获得第一个接收到的数据，读 SPI_DR 的同时清除了 RXNE 位。重复这些操作，发送后续的数据同时接收 n-1 个数据；
4.等待 RXNE=1，然后接收最后一个数据；
5.等待 TXE=1，在 BSY=0 之后关闭 SPI 模块。
也可以在响应 RXNE 或 TXE 标志的上升沿产生的中断的处理程序中实现这个过程。
图：主模式、全双工模式下(BIDIMODE=0 并且 RXONLY=0)连续传输时，TXE/RXNE/BSY 的变化示意图



图：从模式、全双工模式下(BIDIMODE=0 并且 RXONLY=0)连续传输时，TXE/RXNE/BSY 的变化示意图



只发送过程(BIDIMODE=0 并且 RXONLY=0)

在此模式下，传输过程可以简要说明如下，使用 BSY 位等待传输的结束：

1.设置 SPE 位为’1’，使能 SPI 模块；

2.在 SPI_DR 寄存器中写入第一个要发送的数据，这个操作会清除 TXE 标志；

3.等待 TXE=1，然后写入第二个要发送的数据。重复这个操作，发送后续的数据；

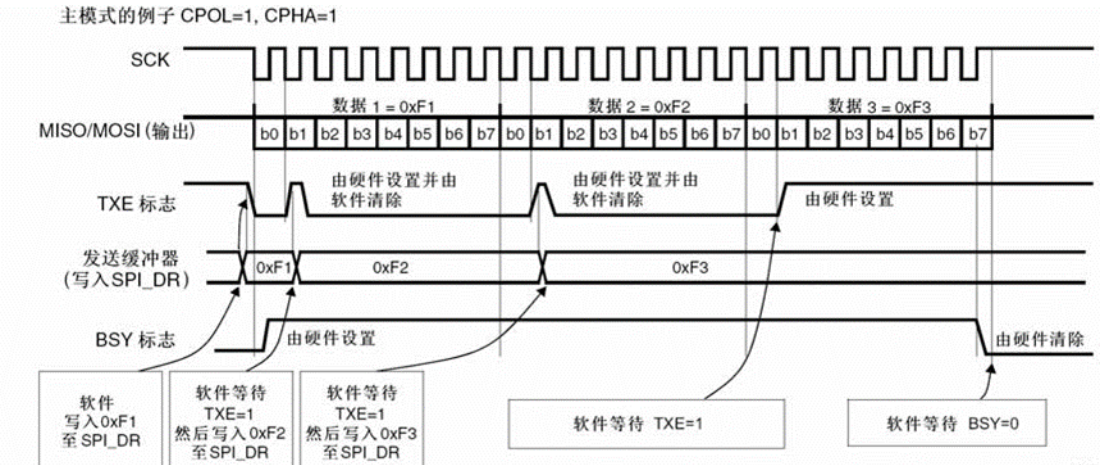
4.写入最后一个数据到 SPI_DR 寄存器之后，等待 TXE=1；然后等待 BSY=0，这表示最后一个数据的传输已经完成。

也可以在响应 TXE 标志的上升沿产生的中断的处理程序中实现这个过程。

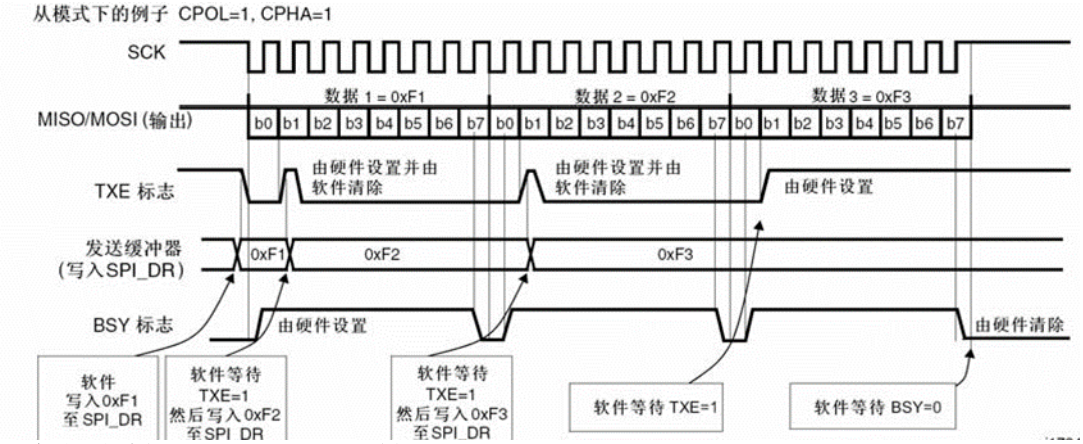
注： 1. 对于不连续的传输，在写入 SPI_DR 寄存器的操作与设置 BSY 位之间有 2 个 APB 时钟周期的延迟，因此在只发送模式下，写入最后一个数据后，最好先等待 TXE=1，然后再等待 BSY=0。

2. 只发送模式下，在传输 2 个数据之后，由于不会读出接收到的数据，SPI_SR 寄存器中的 OVR 位会变为’1’。(译注：软件不必理会这个 OVR 标志位)

图：主设备只发送模式(BIDIMODE=0 并且 RXONLY=0)下连续传输时，，TXE/BSY 变化示意图



图：从设备只发送模式(BIDIMODE=0 并且 RXONLY=0)下连续传输时，TXE/BSY 变化示意图



双向发送过程(BIDIMODE=1 并且 BIDIOE=1)

在此模式下，操作过程类似于只发送模式，不同的是：在使能 SPI 模块之前，需要在 SPI_CR2 寄存器中同时设置 BIDIMODE 和 BIDIOE 位为’1’。

单向只接收模式(BIDIMODE=0 并且 RXONLY=1)

在此模式下，传输过程可以简要说明如下：

1.在 SPI_CR2 寄存器中，设置 RXONLY=1；

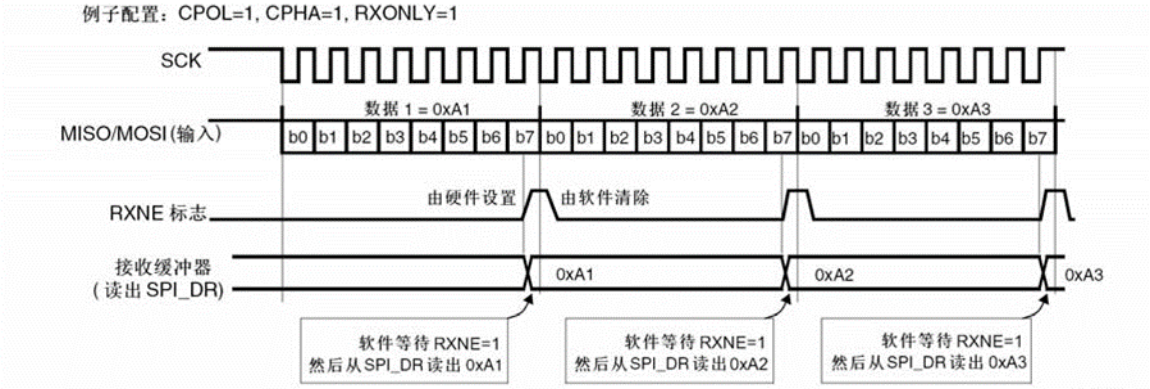
2.设置 SPE=1，使能 SPI 模块：

a)主模式下，立刻产生 SCK 时钟信号，在关闭 SPI(SPE=0)之前，不断地接收串行数据；

b)从模式下，当 SPI 主设备拉低 NSS 信号并产生 SCK 时钟时，接收串行数据。

3.等待 RXNE=1，然后读出 SPI_DR 寄存器以获得收到的数据(同时会清除 RXNE 位)。重复这个操作接收所有数据。也可以在响应 RXNE 标志的上升沿产生的中断的处理程序中实现这个过程。

图：只接收模式(BIDIMODE=0 并且 RXONLY=1)下连续传输时，RXNE 变化示意图



单向接收过程(BIDIMODE=1 并且 BIDIOE=0)

在此模式下，操作过程类似于只接收模式，不同的是：在使能 SPI 模块之前，需要在 SPI_CR2 寄存器中设置 BIDIMODE 为’1’并清除 BIDIOE 位为’0’。

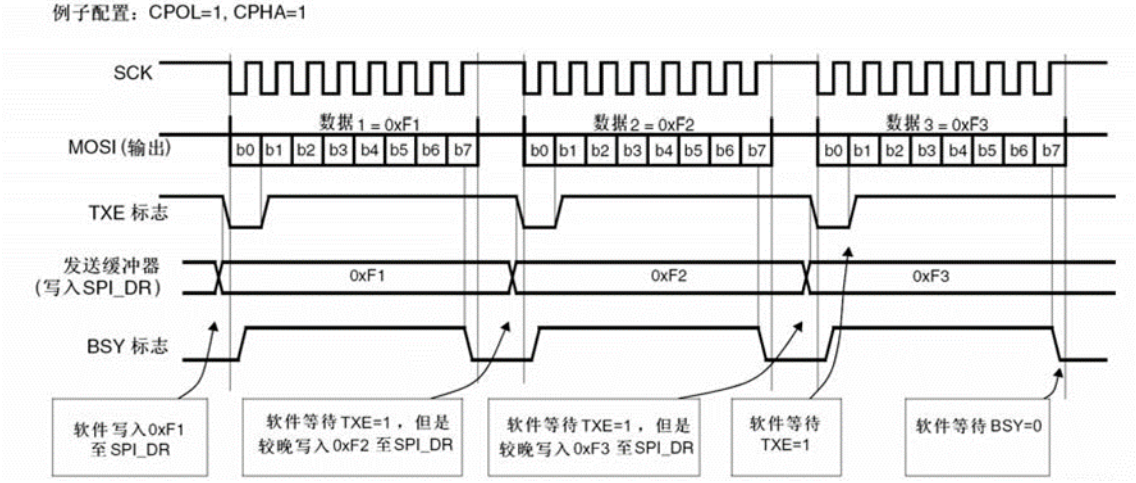
连续和非连续传输

当在主模式下发送数据时，如果软件足够快，能够在检测到每次 TXE 的上升沿(或 TXE 中断)，并立即在正在进行的传输结束之前写入 SPI_DR 寄存器，则能够实现连续的通信；此时，在每个数据项的传输之间的 SPI 时钟保持连续，同时 BSY 位不会被清除。

如果软件不够快，则会导致不连续的通信；这时，在每个数据传输之间会被清除(见下图)。在主模式的只接收模式下(RXONLY=1)，通信总是连续的，而且 BSY 标志始终为’1’。

在从模式下，通信的连续性由 SPI 主设备决定。不管怎样，即使通信是连续的，BSY 标志会在每个数据项之间至少有一个 SPI 时钟周期为低。

图：非连续传输发送(BIDIMODE=0 并且 RXONLY=0)时，TXE/BSY 变化示意图



9.3.6 CRC 计算

CRC 校验用于保证全双工通信的可靠性。数据发送和数据接收分别使用单独的 CRC 计算器。通过对每一个接收位进行可编程的多项式运算来计算 CRC。CRC 的计算是在由 SPI_CR1 寄存器中
CPHA 和 CPOL 位定义的采样时钟边沿进行的。
注意：该 SPI 接口提供了两种 CRC 计算方法，取决于所选的发送和/或接收的数据帧格式：8 位数据帧采用 CR8；16 位数据帧采用 CRC16。
CRC 计算是通过设置 SPI_CR1 寄存器中的 CRCEN 位启用的。设置 CRCEN 位时同时复位 CRC 寄存器(SPI_RXCRCR 和 SPI_TXCRCR)。当设置了 SPI_CR1 的 CRCNEXT 位，SPI_TXCRCR 的内
容将在当前字节发送之后发出。
在传输 SPI_TXCRCR 的内容时，如果在移位寄存器中收到的数值与 SPI_RXCRCR 的内容不匹配，则 SPI_SR 寄存器的 CRCERR 标志位被置 1。
如果在 TX 缓冲器中还有数据，CRC 的数值仅在数据字节传输结束后传送。在传输 CRC 期间，
CRC 计算器关闭，寄存器的数值保持不变。
SPI 通信可以通过以下步骤使用 CRC：
●设置 CPOL、CPHA、LSBFirst、BR、SSM、SSI 和 MSTR 的值；
●在 SPI_CRCPR 寄存器输入多项式；
●通过设置 SPI_CR1 寄存器 CRCEN 位使能 CRC 计算，该操作也会清除寄存器 SPI_RXCRCR 和 SPI_TXCRC；
●设置 SPI_CR1 寄存器的 SPE 位启动 SPI 功能；
●启动通信并且维持通信，直到只剩最后一个字节或者半字；
●在把最后一个字节或半字写进发送缓冲器时，设置 SPI_CR1 的 CRCNext 位，指示硬件在发送完成最后一个数据之后，发送 CRC 的数值。在发送 CRC 数值期间，停止 CRC 计算；
●当最后一个字节或半字被发送后，SPI 发送 CRC 数值，CRCNext 位被清除。同样，接收到的 CRC 与 SPI_RXCRCR 值进行比较，如果比较不相配，则设置 SPI_SR 上的 CRCERR 标志位，当设置了 SPI_CR2 寄存器的 ERRIE 时，则产生中断。
注意： 当 SPI 模块处于从设备模式时，请注意在时钟稳定之后再使能 CRC 计算，否则可能会得到错误的 CRC 计算结果。事实上，只要设置了 CRCEN 位，只要在 SCK 引脚上有输入时钟，不管 SPE 位的状态，都会进行 CRC 的计算。
当 SPI 时钟频率较高时，用户在发送 CRC 时必须小心。在 CRC 传输期间，使用 CPU 的时间应尽可能少；为了避免在接收最后的数据和 CRC 时出错，在发送 CRC 过程中应禁止函数调用。必须在发送/接收最后一个数据之前完成设置 CRCNEXT 位的操作。
当 SPI 时钟频率较高时，因为 CPU 的操作会影响 SPI 的带宽，建议采用 DMA 模式以避免 SPI 降低的速度。

当配置为从模式并且使用了 NSS 硬件模式，NSS 引脚应该在数据传输和 CRC 传输期间保持为低。
当配置 SPI 为从模式并且使用 CRC 的功能，即使 NSS 引脚为高时仍然会执行 CRC 的计算(译注： 当 NSS 信号为高时，如果 SCK 引脚上有时钟脉冲，则 CRC 计算会继续执行)。例如：当主设备交替地与多个从设备进行通信时，将会出现这种情况(译注：此时要想办法避免 CRC 的误操作)。
在不选中一个从设备(NSS 信号为高)转换到选中一个新的从设备(NSS 信号为低)的时候，为了保持主从设备端下次 CRC 计算结果的同步，应该清除主从两端的 CRC 数值。
按照下述步骤清除 CRC 数值：
1.关闭 SPI 模块(SPE=0);
2.清除 CRCEN 位为’0’；
3.设置 CRCEN 位为’1’；
4.使能 SPI 模块(SPE=1)。

9.3.7 状态标志

应用程序通过 3 个状态标志可以完全监控 SPI 总线的状态。
发送缓冲器空闲标志(TXE)
此标志为’1’时表明发送缓冲器为空，可以写下一个待发送的数据进入缓冲器中。当写入 SPI_DR 时，TXE 标志被清除。
接收缓冲器非空(RXNE)
此标志为’1’时表明在接收缓冲器中包含有效的接收数据。读 SPI 数据寄存器可以清除此标志。
忙(Busy)标志
BSY 标志由硬件设置与清除(写入此位无效果)，此标志表明 SPI 通信层的状态。
当它被设置为’1’ 时， 表明 SPI 正忙于通信，但有一个例外：在主模式的双向接收模式下(MSTR=1、BDM=1 并且 BDOE=0)，在接收期间 BSY 标志保持为低。
在软件要关闭 SPI 模块并进入停机模式(或关闭设备时钟)之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输，因此需要严格按照下述过程执行。
BSY 标志还可以用于在多主系统中避免写冲突。
除了主模式的双向接收模式(MSTR=1、BDM=1 并且 BDOE=0)，当传输开始时，BSY 标志被置’1’。
以下情况时此标志将被清除为’0’：
●当传输结束(主模式下，如果是连续通信的情况例外)；
●当关闭 SPI 模块；
●当产生主模式失效(MODF=1)。
如果通信不是连续的，则在每个数据项的传输之间，BSY 标志为低。当通信是连续时：
●主模式下：在整个传输过程中，BSY 标志保持为高；
●从模式下：在每个数据项的传输之间，BSY 标志在一个 SPI 时钟周期中为低。
注：不要使用 BSY 标志处理每一个数据项的发送和接收，最好使用 TXE 和 RXNE 标志。

9.3.8 关闭 SPI

当通讯结束，可以通过关闭 SPI 模块来终止通讯。清除 SPE 位即可关闭 SPI。
在某些配置下，如果再传输还未完成时，就关闭 SPI 模块并进入停机模式，则可能导致当前的传输被破坏，而且 BSY 标志也变得不可信。
为了避免发生这种情况，关闭 SPI 模块时，建议按照下述步骤操作：
在主或从模式下的全双工模式(BIDIMODE=0，RXONLY=0)
1.等待 RXNE=1 并接收最后一个数据；
2.等待 TXE=1；
3.等待 BSY=0；
4.关闭 SPI(SPE=0)，最后进入停机模式(或关闭该模块的时钟)。

在主或从模式下的单向只发送模式(BIDIMODE=0，RXONLY=0)或双向的发送模式(BIDIMODE=1，BIDIOE=1)

在 SPI_DR 寄存器中写入最后一个数据后：

1.等待 TXE=1；

2.等待 BSY=0；

3.关闭 SPI(SPE=0)，最后进入停机模式(或关闭该模块的时钟)。

在主或从模式下的单向只接收模式(MSTR=1，BIDIMODE=0，RXONLY=1)或双向的接收模式(MSTR=1，BIDIMODE=1，BIDIOE=0)

这种情况需要特别地处理，以保证 SPI 不会开始一次新的传输：

1.等待倒数第二个(第 n-1 个)RXNE=1；

2.在关闭 SPI(SPE=0)之前等待一个 SPI 时钟周期(使用软件延迟)；

3.在进入停机模式(或关闭该模块的时钟)之前等待最后一个 RXNE=1。

注：在主模式下的单向只发送模式(MSTR=1，BDM=1，BDOE=0)时，传输过程中 BSY 标志始终为低。

在从模式下的只接收模式(MSTR=0，BIDIMODE=0，RXONLY=1)或双向的接收模式(MSTR=0，BIDIMODE=1，BIDIOE=0)

1.可以在任何时候关闭 SPI(SPE=0)，SPI 会在当前的传输结束后被关闭；

2.如果希望进入停机模式，在进入停机模式(或关闭该模块的时钟)之前必须首先等待 BSY=0。

9.3.9 错误标志

主模式失效错误(MODF)

主模式失效仅发生在：NSS 引脚硬件模式管理下，主设备的 NSS 脚被拉低；或者在 NSS 引脚软件模式管理下，SSI 位被置为’0’时；MODF 位被自动置位。主模式失效对 SPI 设备有以下影响：

●MODF 位被置为’1’，如果设置了 ERRIE 位，则产生 SPI 中断；

●SPE 位被清为’0’。这将停止一切输出，并且关闭 SPI 接口；

●MSTR 位被清为’0’，因此强迫此设备进入从模式。下面的步骤用于清除 MODF 位：

1.当 MODF 位被置为’1’时，执行一次对 SPI_SR 寄存器的读或写操作；

2.然后写 SPI_CR1 寄存器。

在有多个 MCU 的系统中，为了避免出现多个从设备的冲突，必须先拉高该主设备的 NSS 脚，再对 MODF 位进行清零。在完成清零之后，SPE 和 MSTR 位可以恢复到它们的原始状态。

出于安全的考虑，当 MODF 位为’1’时，硬件不允许设置 SPE 和 MSTR 位。

通常配置下，从设备的 MODF 位不能被置为’1’。然而，在多主配置里，一个设备可以在设置了 MODF 位的情况下，处于从设备模式；此时，MODF 位表示可能出现了多主冲突。中断程序可以执行一个复位或返回到默认状态来从错误状态中恢复。

溢出错误

当主设备已经发送了数据字节，而从设备还没有清除前一个数据字节产生的 RXNE 时，即为溢出错误。当产生溢出错误时：

●OVR 位被置为’1’；当设置了 ERRIE 位时，则产生中断。

此时，接收器缓冲器的数据不是主设备发送的新数据，读 SPI_DR 寄存器返回的是之前未读的数据，所有随后传送的数据都被丢弃。

依次读出 SPI_DR 寄存器和 SPI_SR 寄存器可将 OVR 清除。

CRC 错误

当设置了 SPI_CR 寄存器上的 CRCEN 位时，CRC 错误标志用来核对接收数据的有效性。如果移位寄存器中接收到的值(发送方发送的 SPI_TXCRCR 数值)与接收方 SPI_RXCRCR 寄存器中的数值不匹配，则 SPI_SR 寄存器上的 CRCERR 标志被置位为’1’。

9.3.10 SPI 中断

SPI 中断请求表

中断事件	事件标志	使能控制位
发送缓冲器空标志	TXE	TXEIE
接收缓冲器非空标志	RXNE	RXNEIE
主模式失效事件	MODF	ERRIE
溢出错误	OVR	
CRC错误标志	CRCERR	

9.4 特殊功能寄存器说明

9.4.1 SPI 控制寄存器 1(SPI_CR1)

SPI 控制寄存器 1(SPI_CR1)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2)					
			偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	LSBFIRST	SPE	BR[5:3]			MSTR	CPOL	CPHA
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BIDIMODE	双向数据模式使能 (Bidirectional data mode enable) 0：选择“双线双向”模式； 1：选择“单线双向”模式。
BIDIOE	双向模式下的输出使能 (Output enable in bidirectional mode) 和BIDIMODE位一起决定在“单线双向”模式下数据的输出方向 0：输出禁止(只收模式)； 1：输出使能(只发模式)。 这个“单线”数据线在主设备端为 MOSI 引脚，在从设备端为 MISO 引脚。
CRCEN	硬件CRC校验使能 (Hardware CRC calculation enable)

	0：禁止CRC计算； 1：启动CRC计算。 注：只有在禁止SPI时(SPE=0)，才能写该位， 否则出错。该位只能在全双工模式下使用。
CRCNEXT	**下一个发送CRC (Transmit CRC next)** 0：下一个发送的值来自发送缓冲区。 1：下一个发送的值来自发送CRC寄存器。 注：在 SPI_DR 寄存器写入最后一个数据后应马上设置该位。
DFF	**数据帧格式 (Data frame format)** 0：使用8位数据帧格式进行发送/接收； 1：使用16位数据帧格式进行发送/接收。 注：只有当 SPI 禁止(SPE=0)时，才能 写该位，否则出错。
RXONLY	**只接收 (Receive only)** 该位和BIDIMODE位一起决定在“双线双向”模式下的传输方向。在多个从设备的配置中，在未被访问的从设备上该位被置1，使得只有被访问的从设备有输出，从而不会造成数据线上数据冲突。 0：全双工(发送和接收)； 1：禁止输出(只接收模式)。
SSM	**软件从设备管理 (Software slave management)** 当SSM被置位时，NSS引脚上的电平由SSI位的 值决定。0：禁止软件从设备管理； 1：启用软件从设备管理。
SSI	**内部从设备选择 (Internal slave select)** 该位只在SSM位为’1’时有意义。它决定了 NSS上的电平，在NSS引脚上的I/O操作无效。
LSBFIRST	**帧格式 (Frame format)** 0：先发送MSB； 1：先发送LSB。 注：当通信在进行时不能改变该位的值。
SPE	**SPI使能 (SPI enable)** 0：禁止SPI设备； 1：开启SPI设备。
BR[5:3]	**波特率控制 (Baud rate control)** 000： fPCLK/2001： fPCLK/4 010： fPCLK/8 011： fPCLK/16 100： fPCLK/32 101： fPCLK/64 110： fPCLK/128 111： fPCLK/256 当通信正在进行的时候，不能修改这些位。
MSTR	**主设备选择 (Master selection)** 0：配置为从设备； 1：配置为主设备。 注：当通信正在进行的时候，不能修改该 位。

CPOL	**时钟极性（Clock polarity）** 0： 空闲状态时，SCK保持低电平； 1： 空闲状态时，SCK保持高电平。 注：当通信正在进行的时候，不能修改该位
CPHA	**时钟相位（Clock phase）** 0： 数据采样从第一个时钟边沿开始； 1： 数据采样从第二个时钟边沿开始。 注：当通信正在进行的时候，不能修改该位。

9.4.2 SPI 控制寄存器 2(SPI_CR2)

SPI 控制寄存器 2(SPI_CR2)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2) 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TXEIE	RXEIE	ERRIE	X	X	SSOE	TXDMAEN	RXDMAEN
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TXEIE	**发送缓冲区空中断使能（Tx buffer empty interrupt enable）** 0：禁止TXE中断； 1：允许 TXE 中断，当 TXE 标志置位为’1’时产生中断请求。
RXNEIE	**接收缓冲区非空中断使能（RX buffer not empty interrupt enable）** 0：禁止RXNE中断； 1：允许 RXNE 中断，当 RXNE 标志置位时产生中断请求。
ERRIR	**错误中断使能（Error interrupt enable）** 当错误(CRCERR、OVR、MODF)产生时，该位控制是否产生中断 0：禁止错误中断； 1：允许错误中断。
SSOE	**SS输出使能（SS output enable）**

	0：禁止在主模式下SS输出，该设备可以工作在多主设备模式； 1：设备开启时，开启主模式下SS输出，该设备不能工作在多主设备模式。
TXDMAEN	**发送缓冲区DMA使能（Tx buffer DMA enable）** 当该位被设置时，TXE标志一旦被置位就发出DMA请求 0：禁止发送缓冲区DMA； 1：启动发送缓冲区DMA。
RXDMAEN	**接收缓冲区DMA使能（Rx buffer DMA enable）** 当该位被设置时，RXNE标志一旦被置位就发出DMA请求 0：禁止接收缓冲区DMA； 1：启动接收缓冲区DMA。

9.4.3 SPI 状态寄存器(SPI_SR)

SPI 状态寄存器(SPI_SR)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2)					
			偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	BSY	OVR	MODF	CRCERR	UDR	X	TXE	RXNE
Reset:	0	0	0	0	0	0	1	0

位	功能描述
BSY	**忙标志（Busy flag）** 0：SPI不忙； 1：SPI正忙于通信，或者发送缓冲非空。该位由硬件置位或者复位。
OVR	**溢出标志（Overrun flag）** 0：没有出现溢出错误； 1：出现溢出错误。

	该位由硬件置位，由软件序列复位。
MODF	**模式错误（Mode fault）** 0：没有出现模式错误； 1：出现模式错误。 该位由硬件置位，由软件序列复位。
CRCERR	**CRC错误标志（CRC error flag）** 0：收到的CRC值和SPI_RXCRCR寄存器中的值匹配； 1：收到的CRC值和SPI_RXCRCR寄存器中的值不匹配。该位由硬件置位，由软件写’0’而复位。
UDR	**下溢标志位（Underrun flag）** 0：未发生下溢； 1：发生下溢。 该标志位由硬件置’1’，由一个软件序列清’0’
TXE	**发送缓冲为空（Transmit buffer empty）** 0：发送缓冲非空； 1：发送缓冲为空。
RXNE	**接收缓冲非空（Receive buffer not empty）** 0：接收缓冲为空； 1：接收缓冲非空。

9.4.4 SPI 数据寄存器(SPI_DR)

SPI 数据寄存器(SPI_DR)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2)					
			偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read:	DR[7:0]							

Write:								
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DR[15:0]	**数据寄存器 (Data register)** 待发送或者已经收到的数据 数据寄存器对应两个缓冲区：一个用于写(发送缓冲)；另外一个用于读(接收缓冲)。写操作将数据写到发送缓冲区；读操作将返回接收缓冲区里的数据。 **对SPI模式的注释：**根据SPI_CR1的DFF位对数据帧格式的选择，数据的发送和接收可以是8位或者16位的。为保证正确的操作，需要在启用SPI之前就确定好数据帧格式。 对于8位的数据，缓冲器是8位的，发送和接收时只会用到SPI_DR[7:0]。在接收时，SPI_DR[15:8]被强制为0。 对于 16 位的数据，缓冲器是 16 位的，发送和接收时会用到整个数据寄存器，即SPI_DR[15:0]。

9.4.5 SPI CRC 多项式寄存器(SPI_CRCPR)

SPI CRC 多项式寄存器 (SPI_CRCPR)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2) 偏移地址： 0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CRCPOLY[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CRCPOLY[7:0]							
Reset:	0	0	0	0	0	1	1	1

位	功能描述
CRCPOLY[15:0]	**CRC多项式寄存器 (CRC polynomial register)** 该寄存器包含了CRC计算时用到的多项式。 其复位值为 0x0007，根据应用可以设置其他数值。

9.4.6 SPI Rx CRC 寄存器(SPI_RXCRCR)

SPI Rx CRC 寄存器 (SPI_RXCRCR)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2)					
			偏移地址： 0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	RxCRC[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	RxCRC[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
RxCRC[15:0]	**接收CRC寄存器** 在启用CRC计算时，RXCRC[15:0]中包含了依据收到的字节计算的CRC数值。当在SPI_CR1 的CRCEN位写入'1'时，该寄存器被复位。CRC计算使用SPI_CRCPR中的多项式。 当数据帧格式被设置为8位时，仅低8位参与计算，并且按照CRC8的方法进行；当数据帧格式为16位时，寄存器中的所有16位都参与计算，并且按照CRC16的标准。 注：当 BSY 标志为'1'时读该寄存器，将可能读到不正确的数值。

9.4.7 SPI Tx CRC 寄存器(SPI_TXCRCR)

SPI Tx CRC 寄存器 (SPI_TXCRCR)			基地址： 0x4001 3000(SPI1) 0x4000 3800(SPI2)					
			偏移地址： 0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X

Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	TxCRC[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	TxCRC[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TxCRC[15:0]	**发送CRC寄存器** 在启用CRC计算时，TXCRC[15:0]中包含了依据将要发送的字节计算的CRC数值。当在 SPI_CR1中的CRCEN位写入'1'时，该寄存器被复位。CRC计算使用 SPI_CRCPR中的多项式。 当数据帧格式被设置为8位时，仅低8位参与计算，并且按照CRC8的方法进行；当数据帧格式为16位时，寄存器中的所有16个位都参与计算，并且按照CRC16的标准。 注：当 BSY 标志为'1'时读该寄存器，将可能读到不正确的数值。

10 通用 TIMx 定时器（TIM2 & TIM3 & TIM4）

10.1 TIM 概述

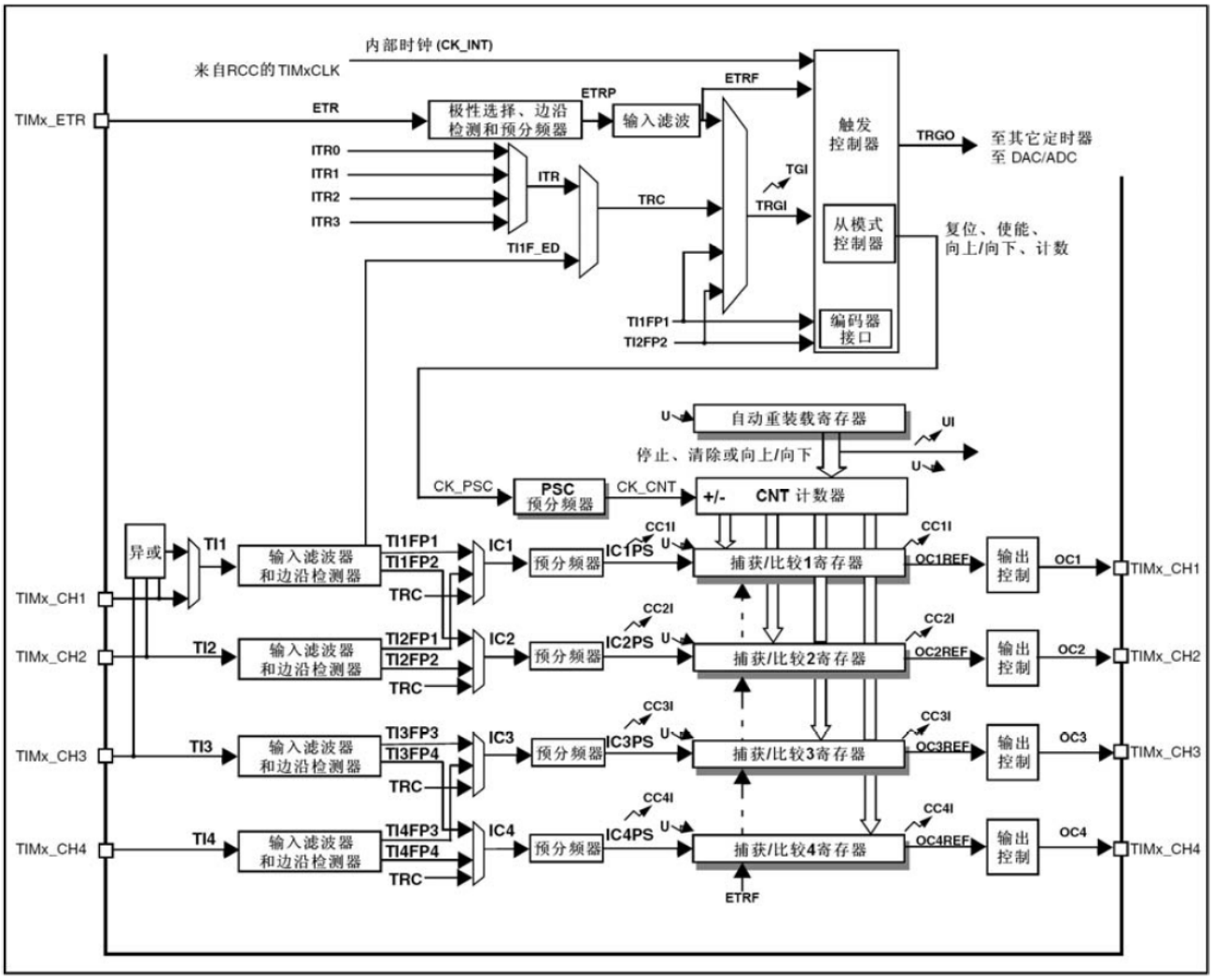
通用 TIMx 定时器(TIM2 和 TIM3 和 TIM4)由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、和 PWM)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

每个通用定时器(TIMx)是完全独立的，它们不共享任何资源。它们可以同步操作。

10.2 TIM2 和 TIM3 和 TIM4 主要特性

TIM2 和 TIM3 和 TIM4 定时器的功能包括：
●16 位向上、向下、向上/下自动装载计数器
●16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65535 之间的任意数值
●多达 4 个独立通道：
—输入捕获
—输出比较
—PWM 生成(边缘或中间对齐模式)
—单脉冲模式输出
●使用外部信号控制定时器和定时器互联的同步电路
●如下事件发生时产生中断/DMA：
—更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
—触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
—输入捕获
—输出比较
●支持针对定位的增量(正交)编码器和霍尔传感器电路
●触发输入作为外部时钟或者按周期的电流管理

10.3 TIM 框图



事件

中断和 DMA 输出

10.4 TIM2 和 TIM3 和 TIM4 功能描述

10.4.1 时基单元

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

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。时基单元包含：

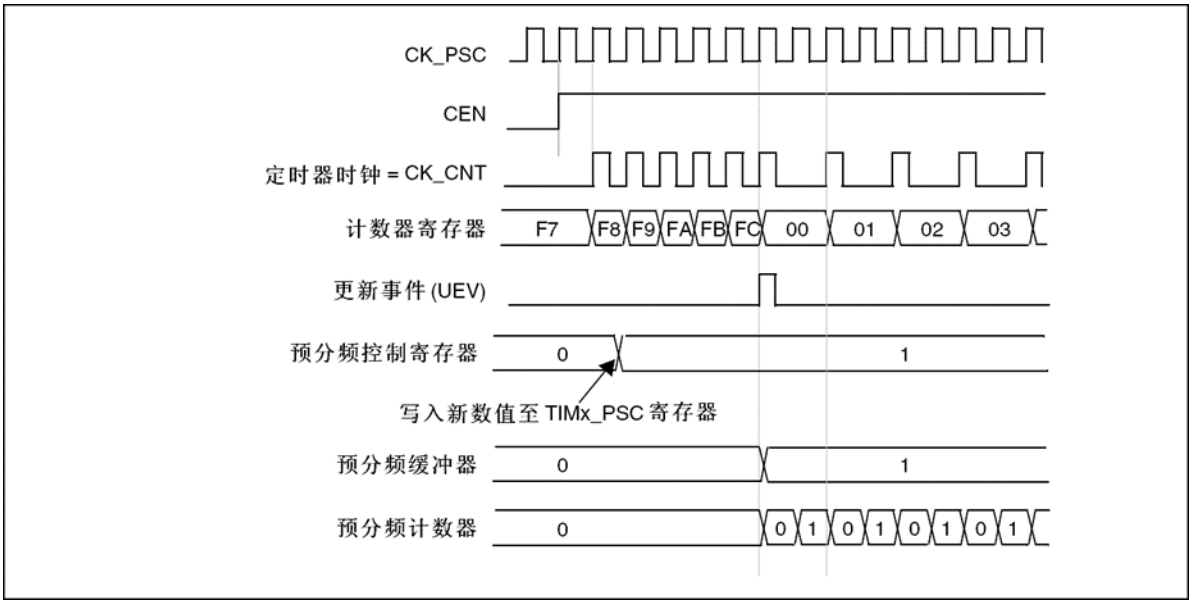
●计数器寄存器(TIMx_CNT)

●预分频器寄存器 (TIMx_PSC)

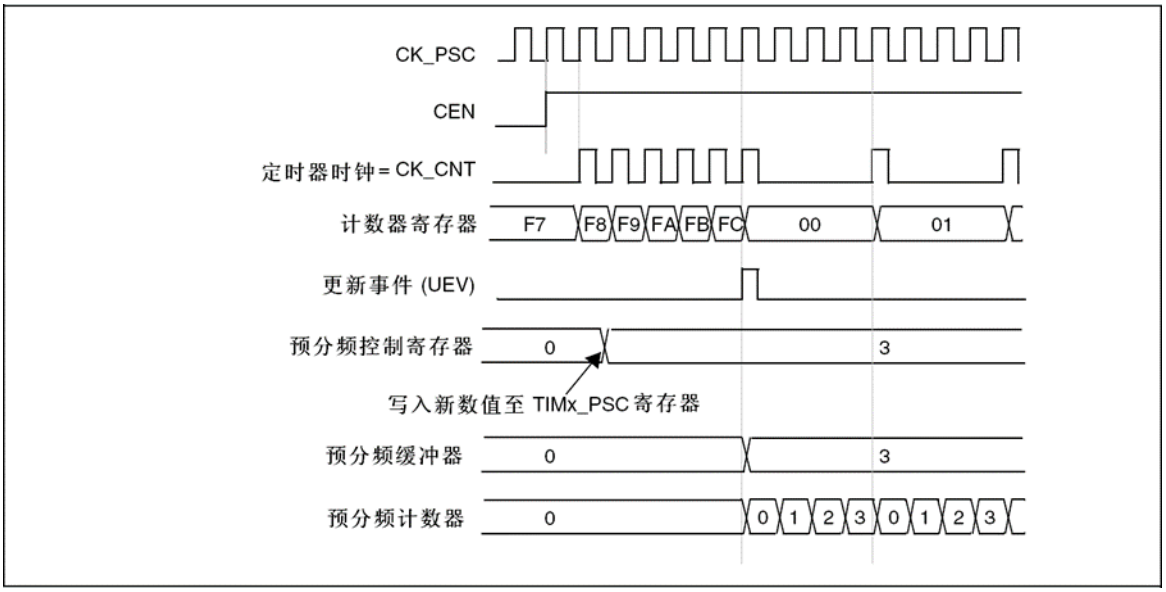
●自动装载寄存器（TIMx_ARR）
自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。(更多有关使能计数器的细节，请参见控制器的从模式描述)。
注：真正的计数器使能信号 CNT_EN 是在 CEN 的一个时钟周期后被设置。

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。
下图给出了在预分频器运行时，更改计数器参数的例子。
图：当预分频器的参数从 1 变到 2 时，计数器的时序图



图：当预分频器的参数从 1 变到 4 时，计数器的时序图



10.4.2 计数器模式

向上计数模式

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

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数(TIMx_RCR)时，产生更新事件(UEV)；否则每次计数器溢出时才产生更新事件。

在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

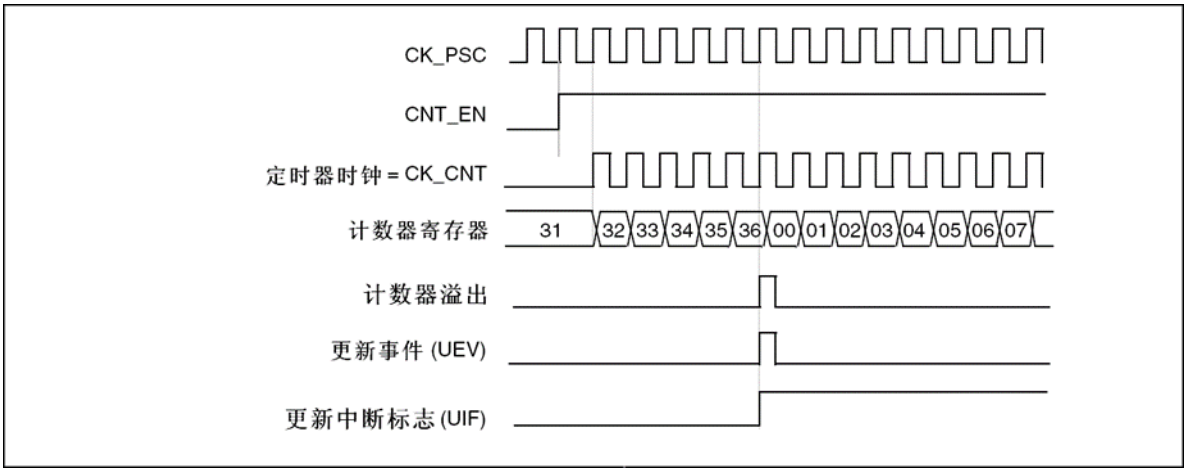
设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清’0’之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清’0’，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 URS 位)设置更新标志位(TIMx_SR 寄存器中的 UIF 位)。

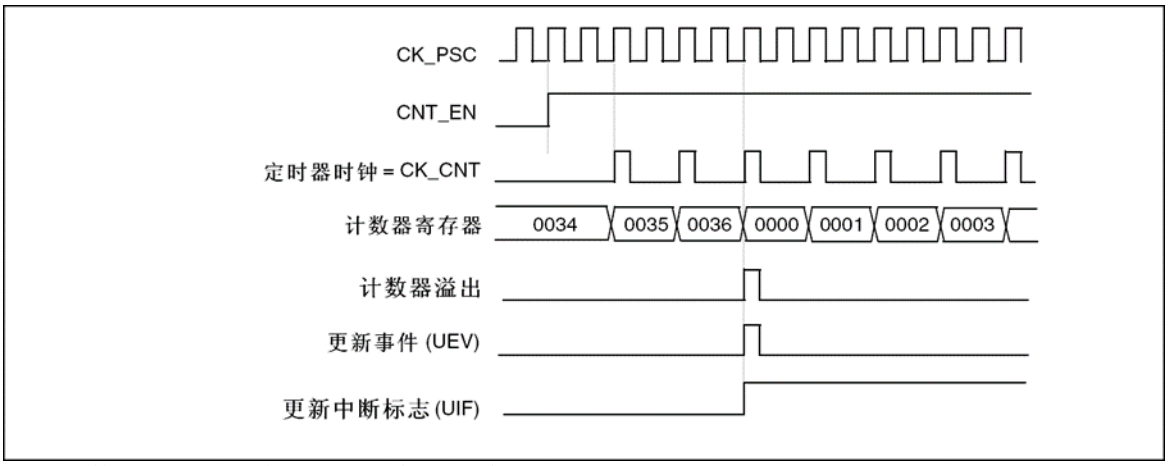
●自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。

●预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

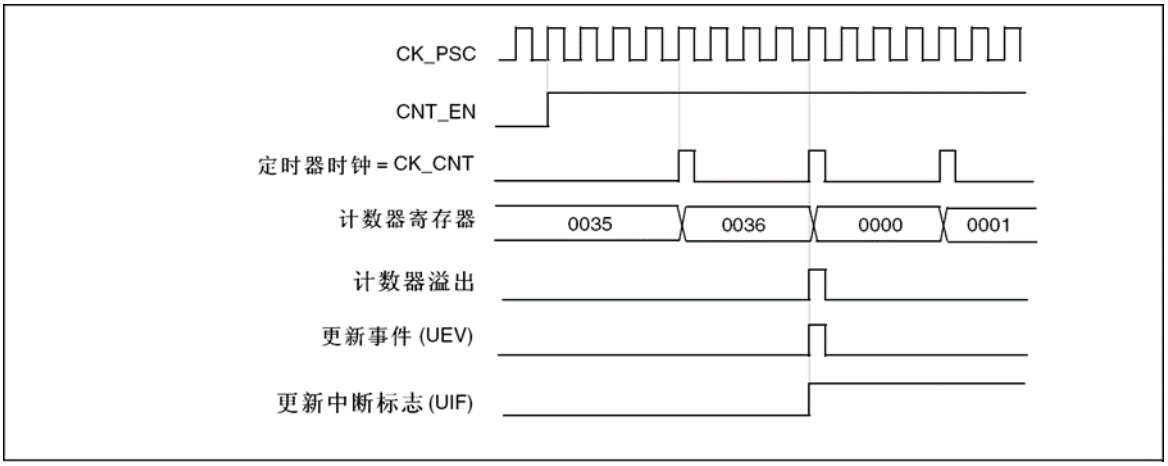
图：计数器时序图，内部时钟分频因子为 1



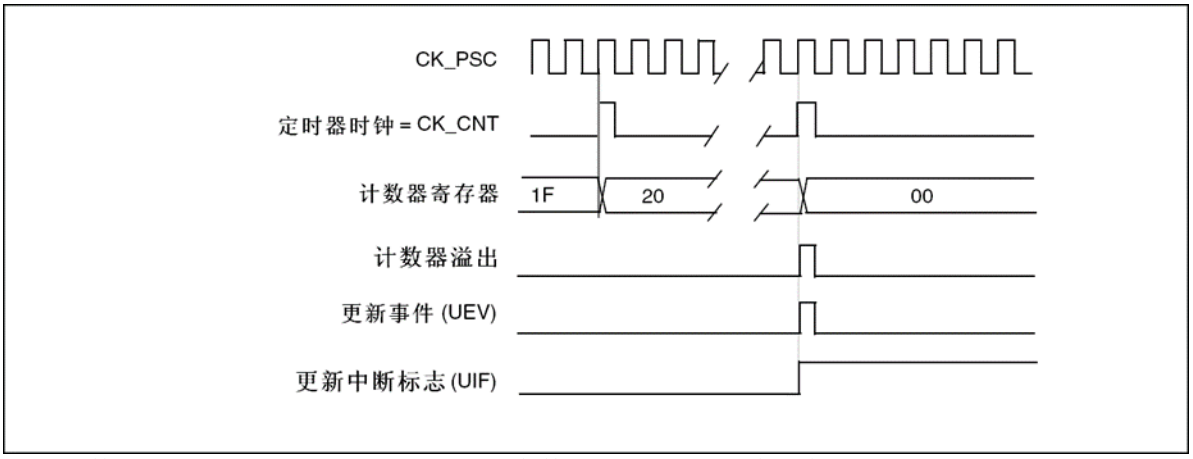
图：计数器时序图，内部时钟分频因子为 2



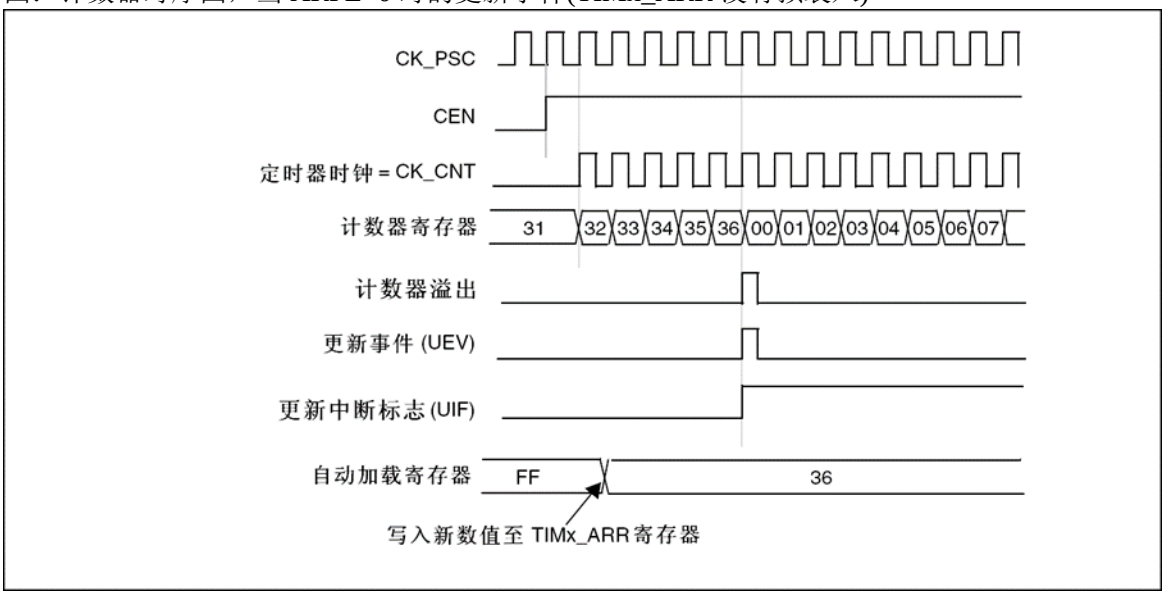
图：计数器时序图，内部时钟分频因子为 4



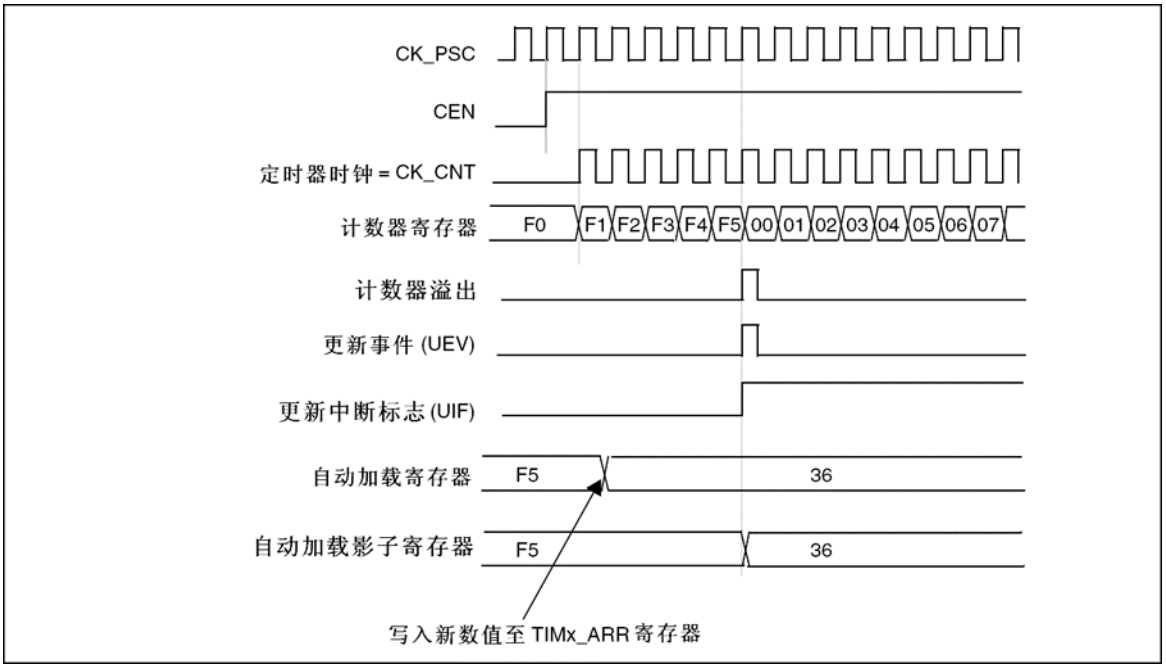
图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)



图：计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)



向下计数模式

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

在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位，也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始(但预分频系数不变)。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

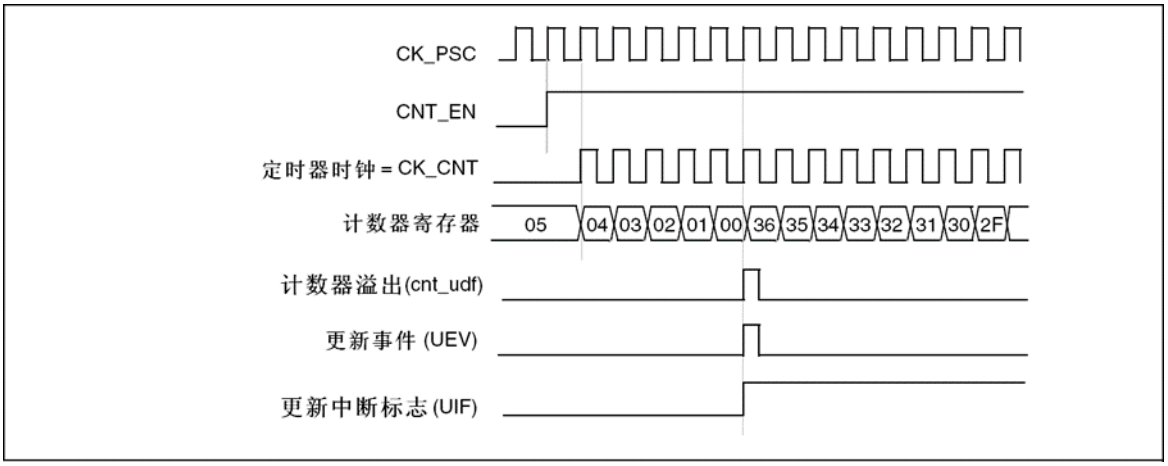
当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。

●预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。

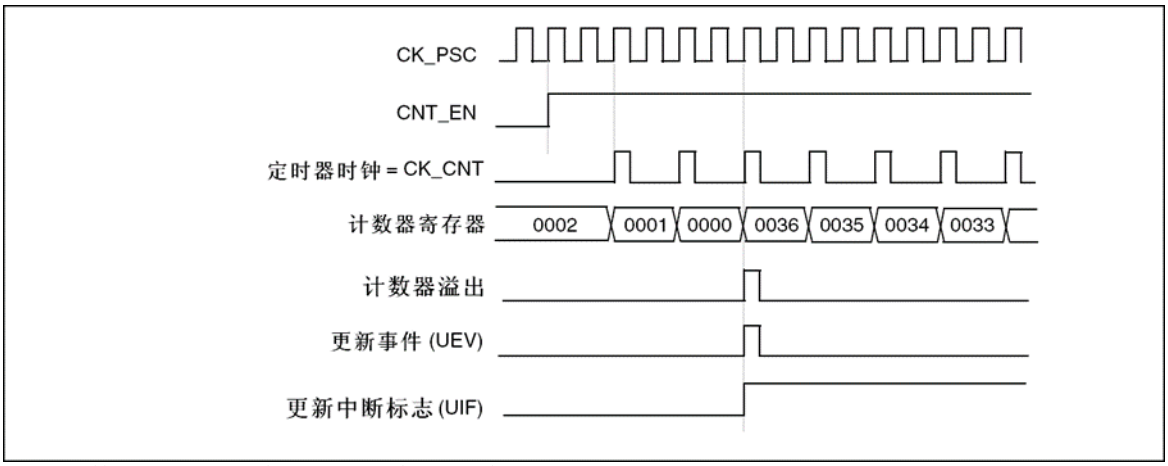
●当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

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

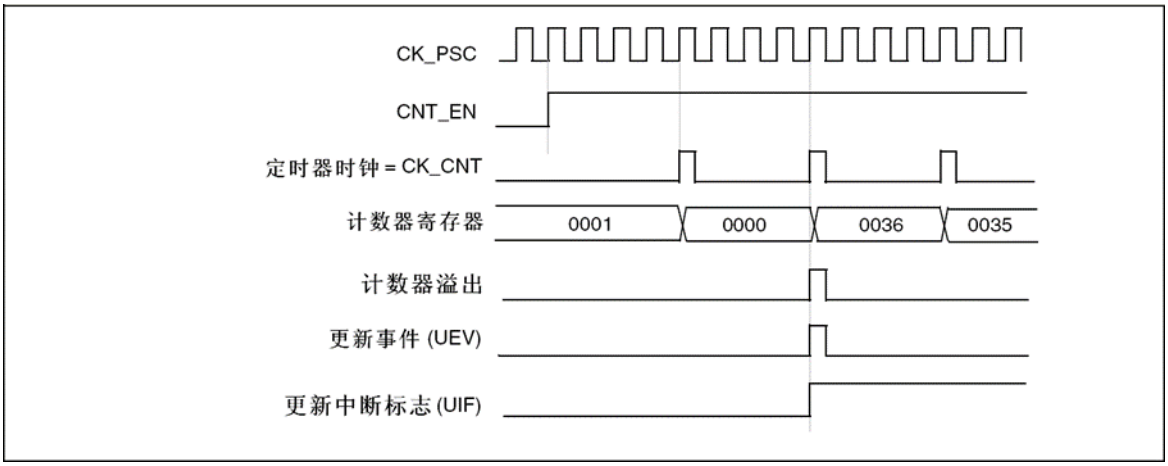
图：计数器时序图，内部时钟分频因子为 1



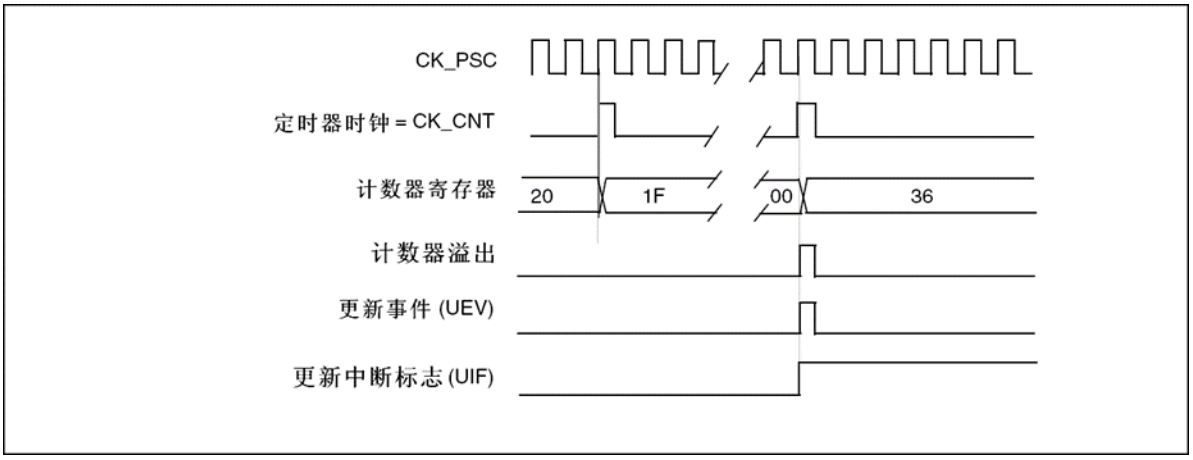
图：计数器时序图，内部时钟分频因子为 2



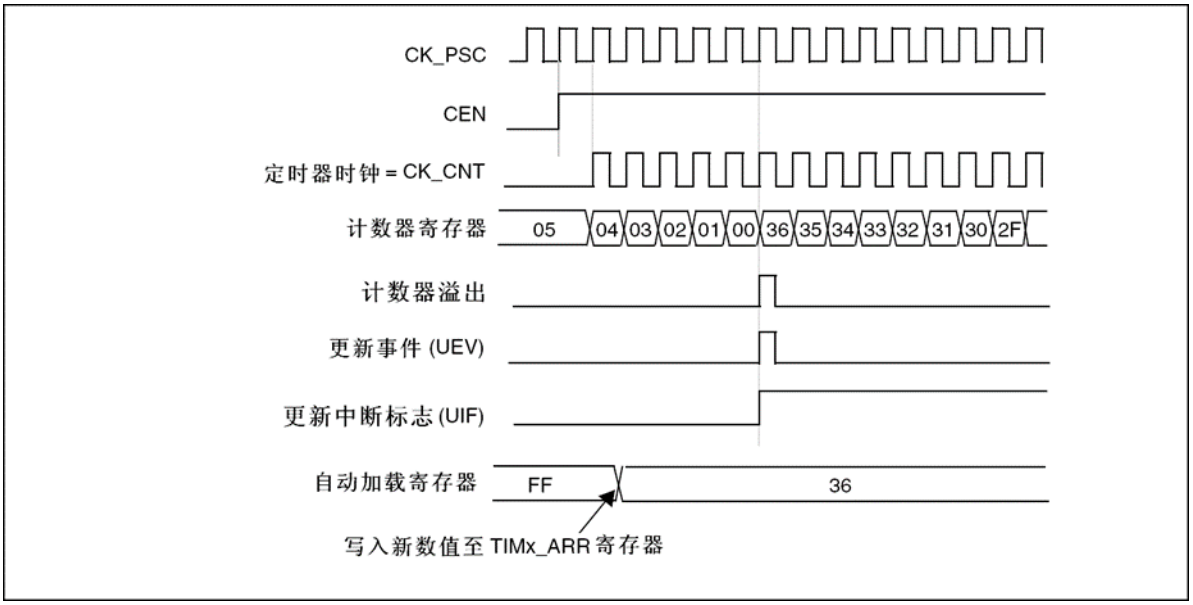
图：计数器时序图，内部时钟分频因子为 4



图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，当没有使用重复计数器时的更新事件



中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

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

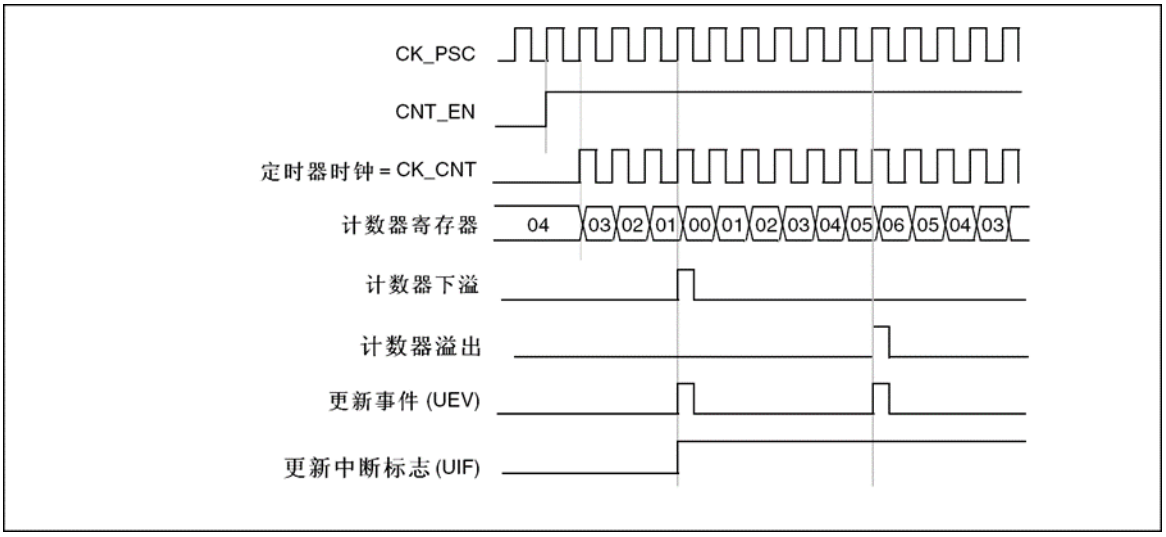
●预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。

●当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

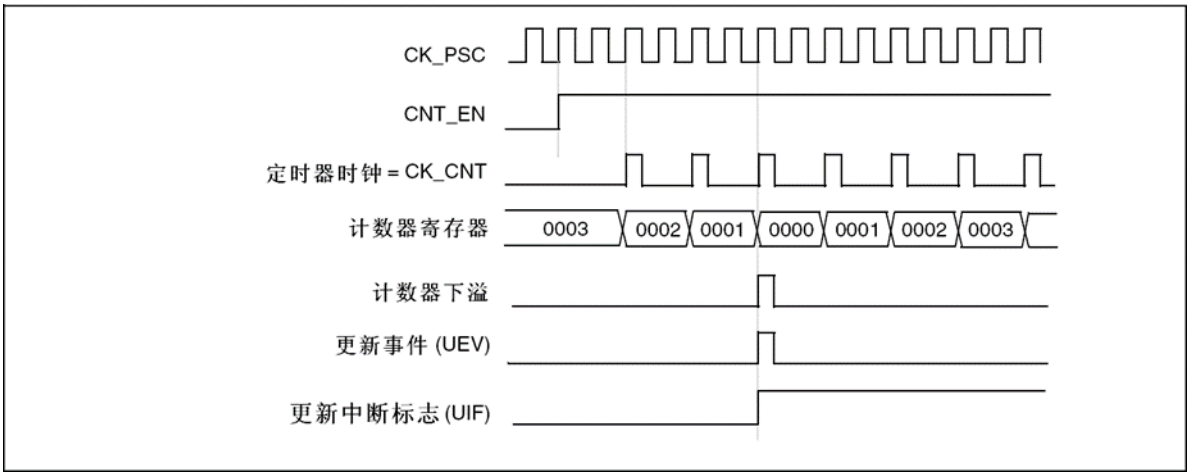
注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)。

以下是一些计数器在不同时钟频率下的操作的例子：

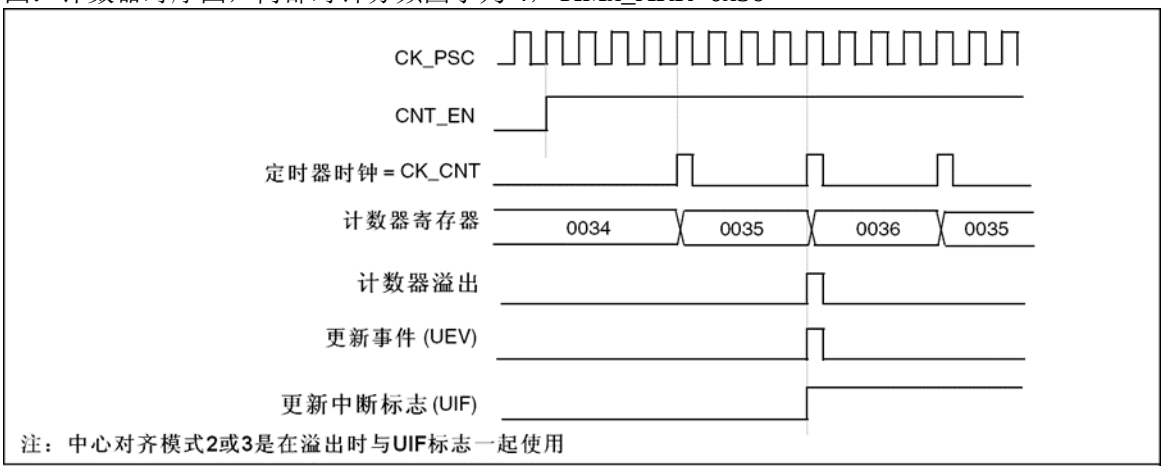
图：计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6



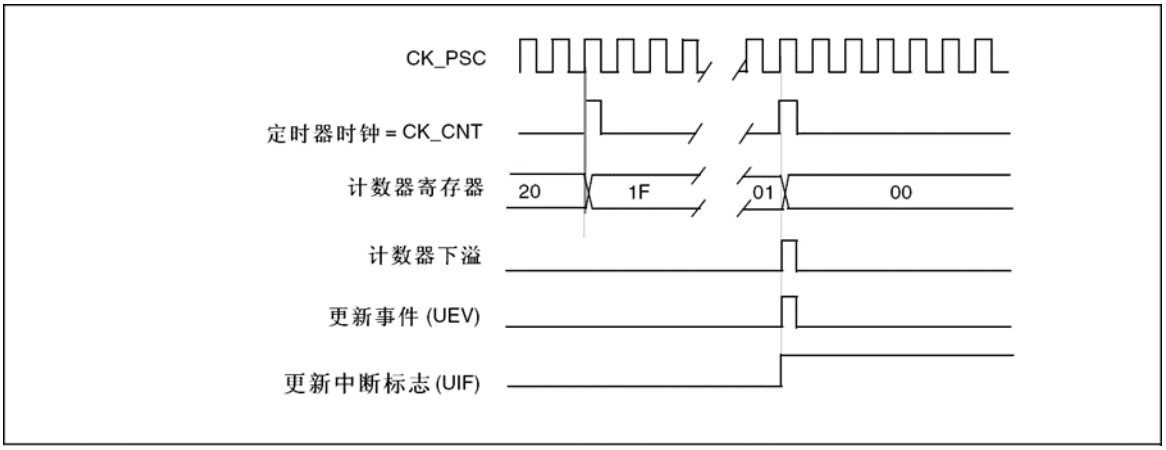
图：计数器时序图，内部时钟分频因子为 2



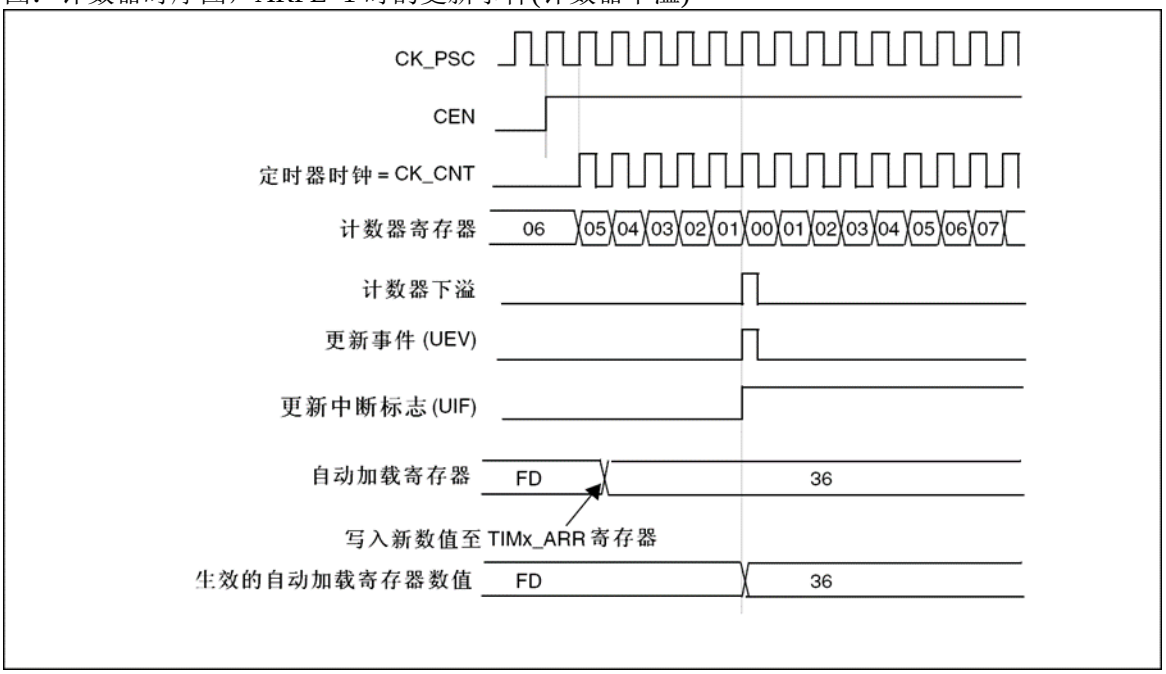
图：计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36



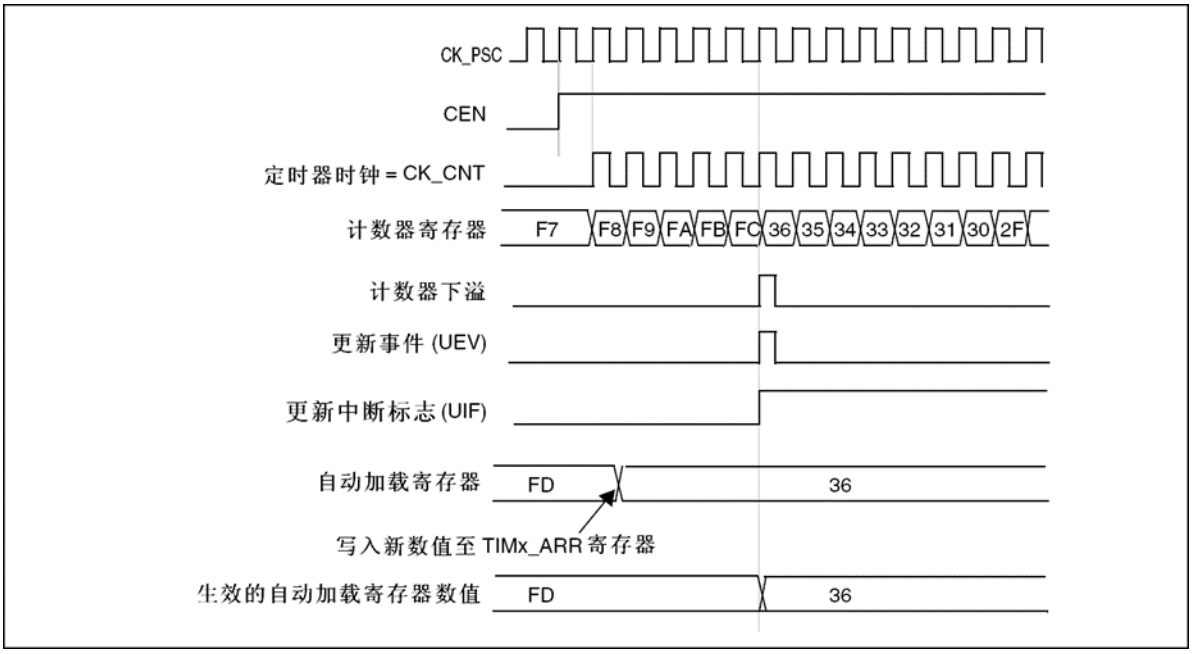
图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，ARPE=1 时的更新事件(计数器下溢)



图：计数器时序图，ARPE=1 时的更新事件(计数器溢出)



10.4.3 时钟选择

计数器时钟可由下列时钟源提供：

●内部时钟(CK_INT)

●外部时钟模式 1：外部输入引脚

●外部时钟模式 2：外部触发输入 ETR

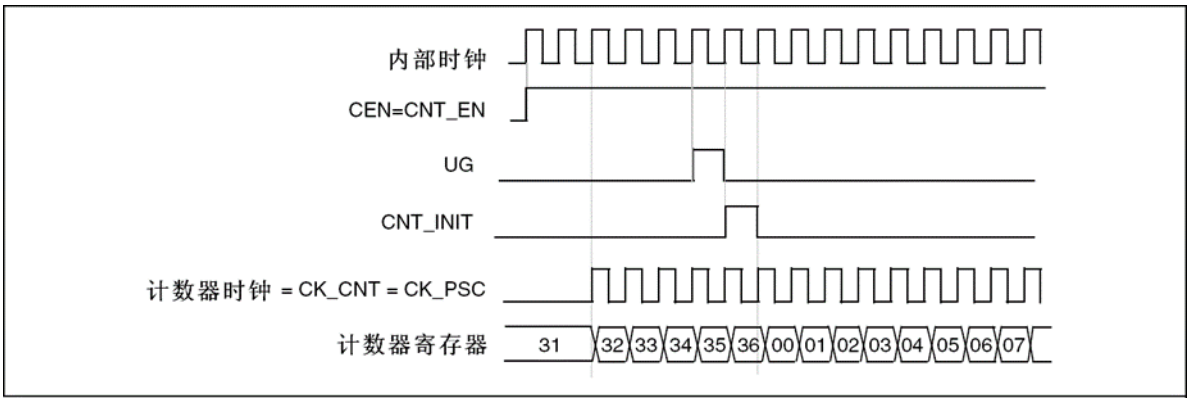
●内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。

内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=000)，则 CEN、DIR(TIMx_CR1 寄存器)和 UG 位(TIMx_EGR 寄存器)是事实上的控制位，并且只能被软件修改(UG 位仍被自动清除)。只要 CEN 位被写成’1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

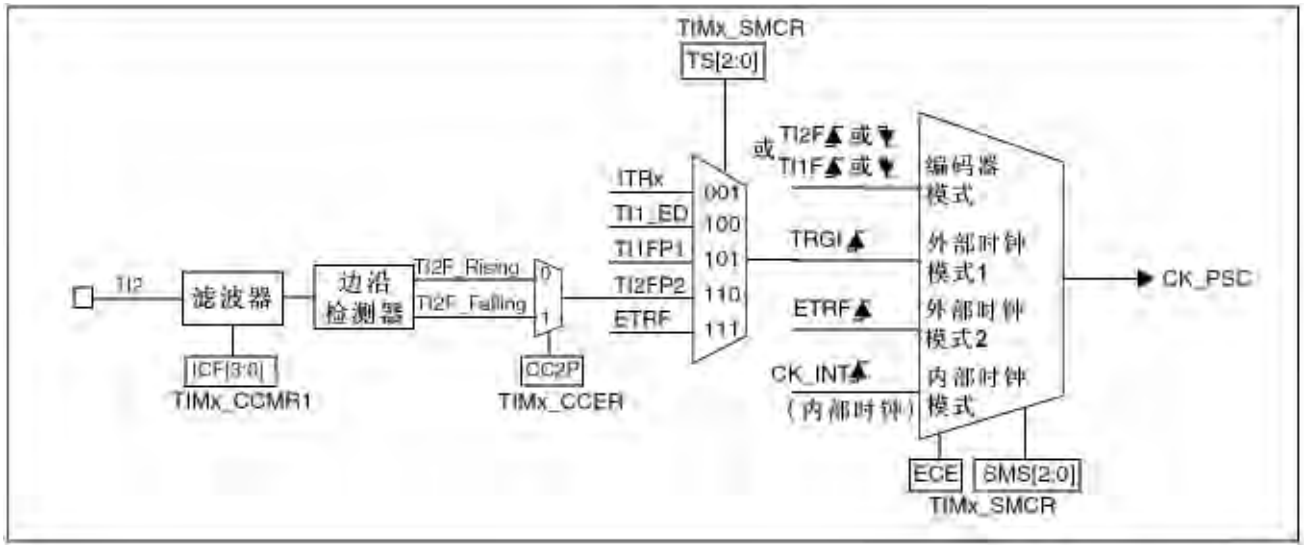
图：一般模式下的控制电路，内部时钟分频因子为 1



外部时钟源模式 1

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

图：TI2 外部时钟连接例子



例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

1、配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿

2、配置 TIMx_CCMR1 寄存器的 IC2F[3:0] ，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)

3、配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性

4、配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1

5、配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源

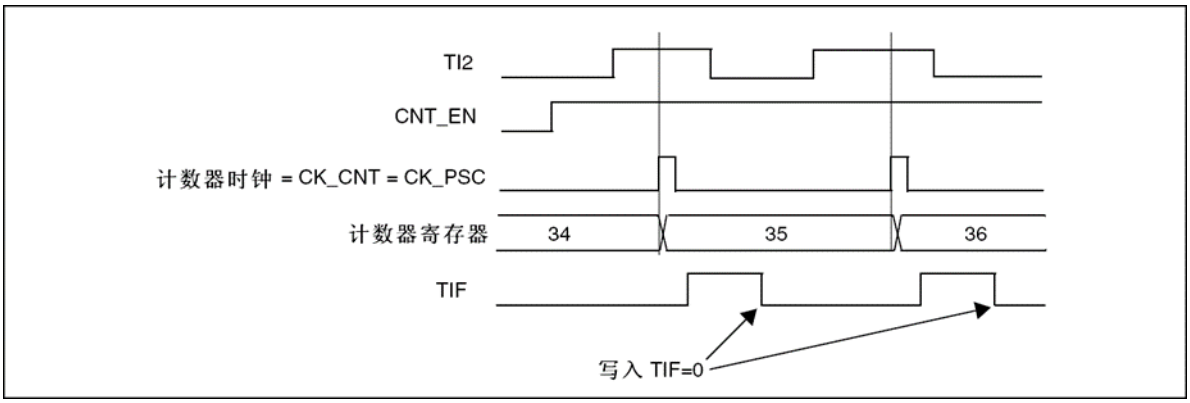
6、设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注：捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

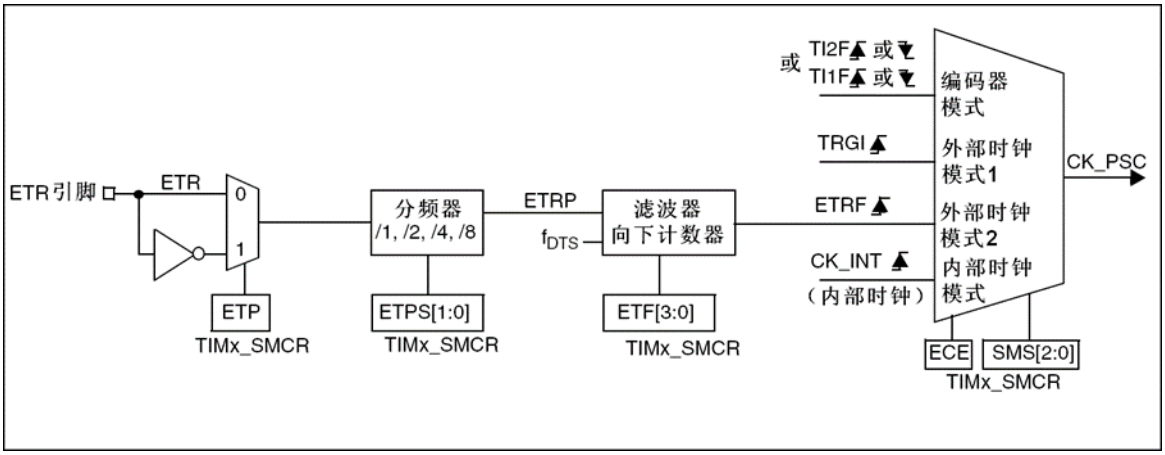
图：外部时钟模式 1 下的控制电路



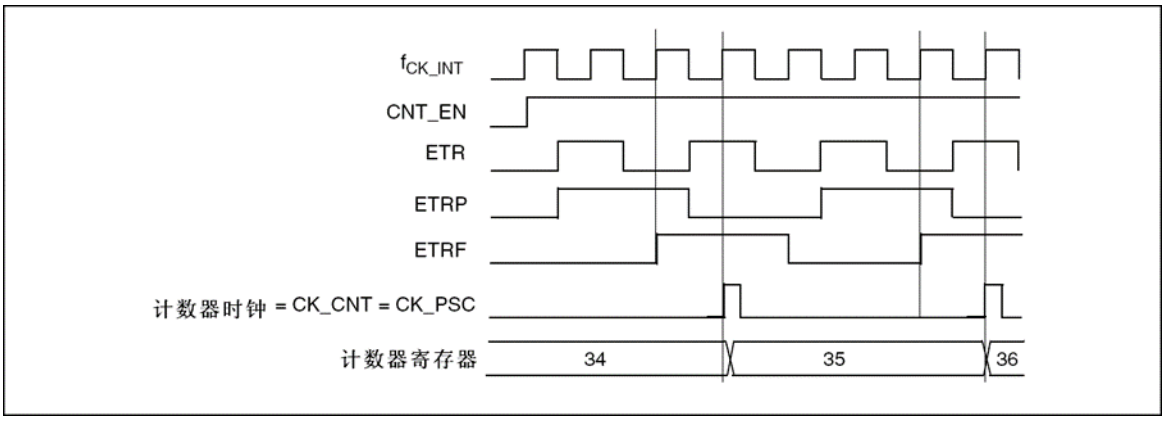
外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1 计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。下图是外部触发输入的框图

图：外部触发输入框图



例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：
1.本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000
2.设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0]=01
3.选择 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0
4.开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1
5.启动计数器，写 TIMx_CR1 寄存器中的 CEN=1
计数器在每 2 个 ETR 上升沿计数一次。
在 ETR 的上升沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。
图：外部时钟模式 2 下的控制电路



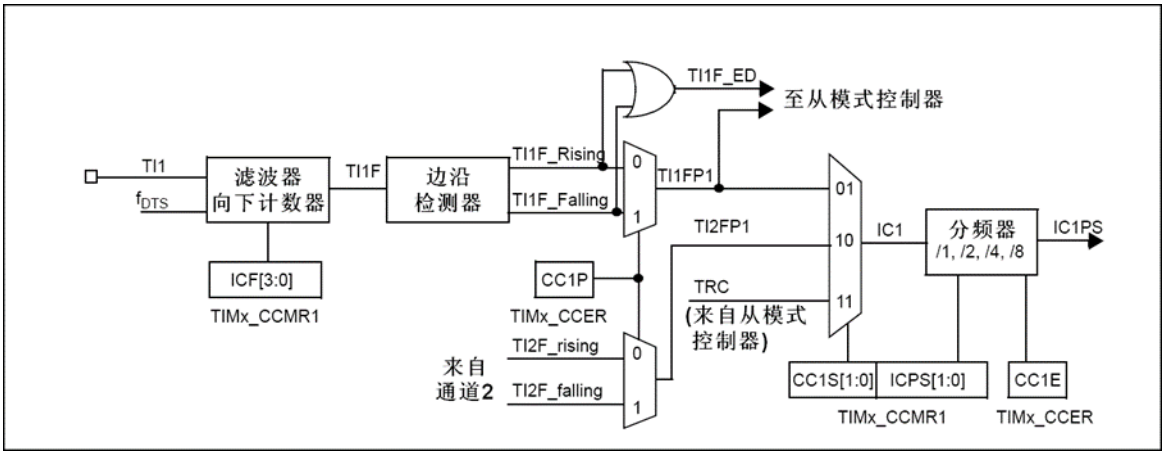
10.4.4 捕获/比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。下图是一个捕获/比较通道概览。

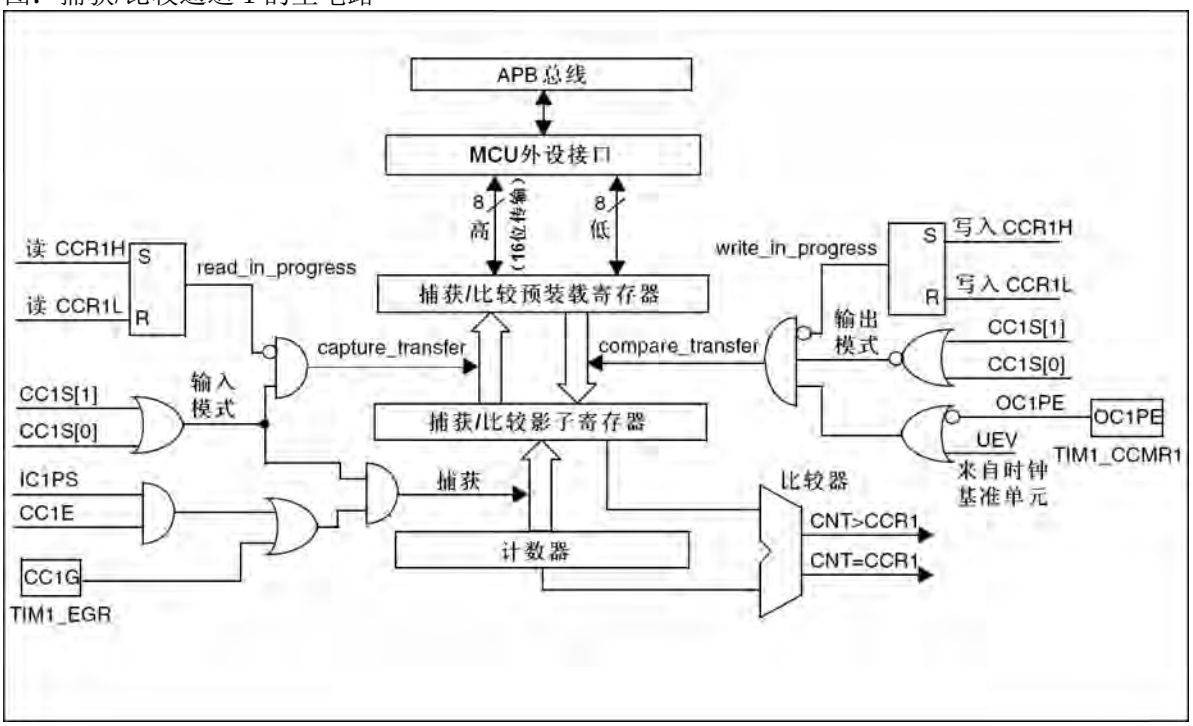
输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。

该信号通过预分频进入捕获寄存器(ICxPS)。

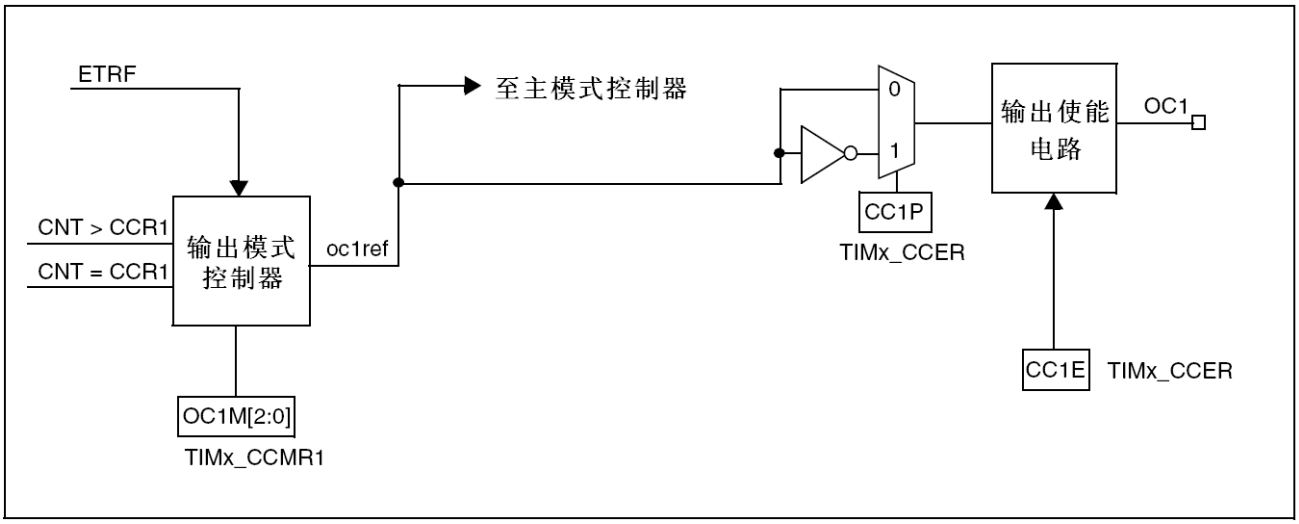
图：捕获/比较通道(如：通道 1 输入部分)



图：捕获/比较通道 1 的主电路



图：捕获/比较通道的输出部分(通道 1)



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

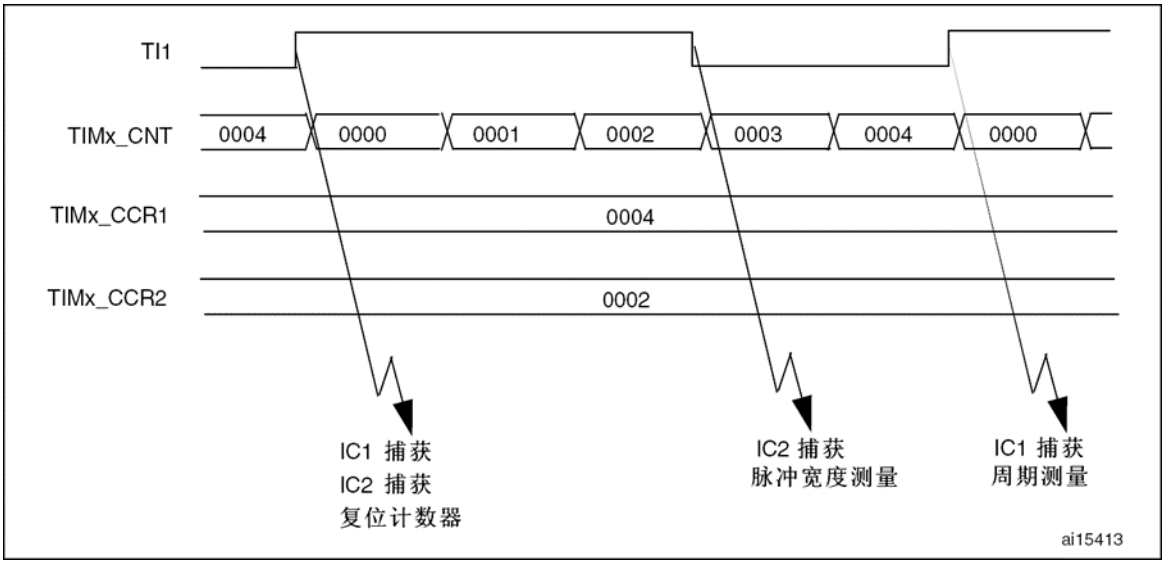
在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。
在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

10.4.5 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。
以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：
●选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
●根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动， 我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
●选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。
●配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
●设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
●如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。
当发生一个输入捕获时：
●产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
●CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 被置 1。
●如设置了 CC1IE 位，则会产生一个中断。
●如设置了 CC1DE 位，则还会产生一个 DMA 请求。
为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。
注：设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

10.4.6 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：
●两个 ICx 信号被映射至同一个 TIx 输入。
●这 2 个 ICx 信号为边沿有效，但是极性相反。
●其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。
例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)
●选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
●选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
●选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
●选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
●选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
●配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
●使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。
图：PWM 输入模式时序



因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIMx_CH1 /TIMx_CH2 信号。

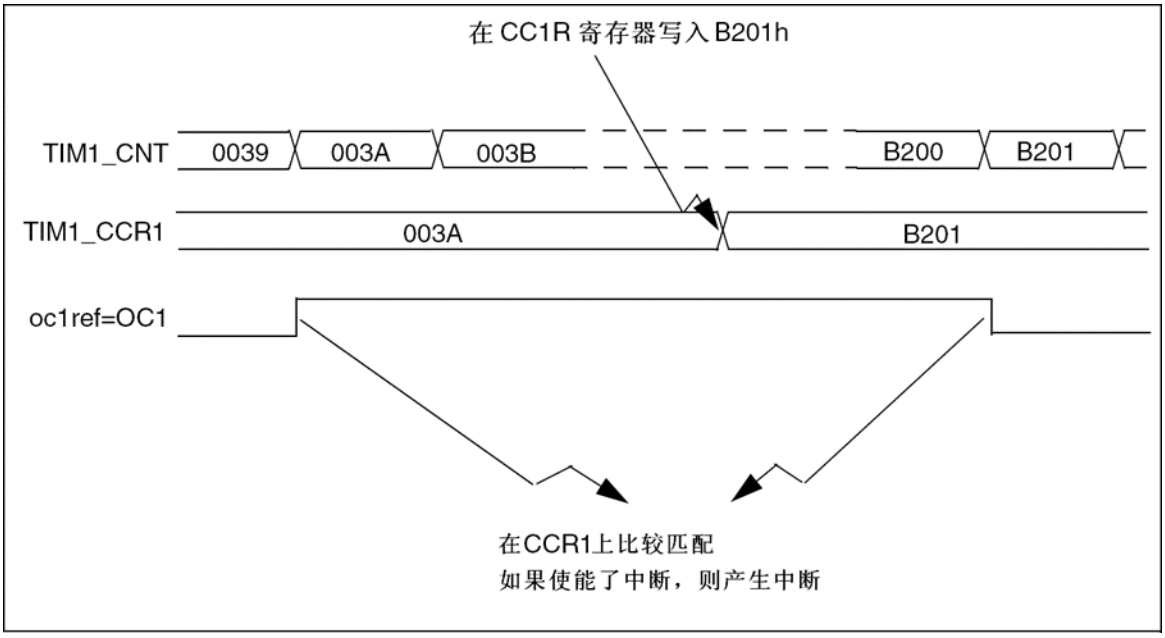
10.4.7 强置输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。
置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。
例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。
置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。
该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

10.4.8 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：
●将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
●设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
●若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
●若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。
TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。
同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。
输出比较模式的配置步骤：
1.选择计数器时钟(内部，外部，预分频器)。
2.将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3.如果要产生一个中断请求，设置 CCxIE 位。

4.选择输出模式，例如：
—要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
—置 OCxPE = 0 禁用预装载寄存器
—置 CCxP = 0 选择极性为高电平有效
—置 CCxE = 1 使能输出
5.设置 TIMx_CR1 寄存器的 CEN 位启动计数器
TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。
图：输出比较模式，翻转 OC1

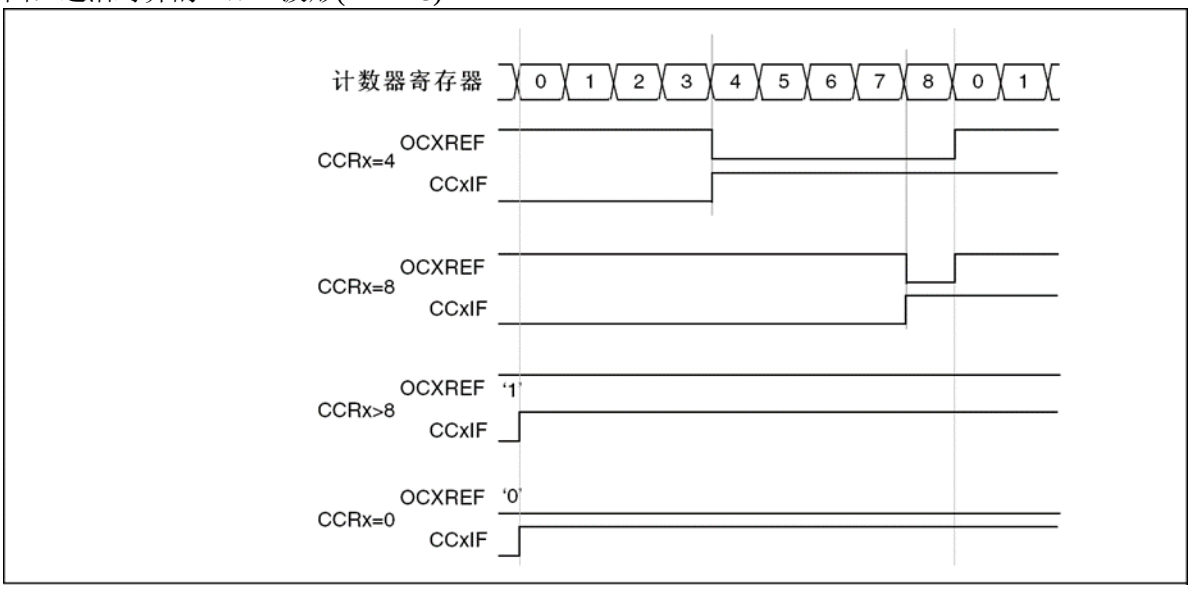


10.4.9 PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。
在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中) 使能自动重装载的预装载寄存器。
仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。
OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。
在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。
PWM 边沿对齐模式
●向上计数配置
当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

图：边沿对齐的 PWM 波形(ARR=8)



●向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。参看中央对齐模式章节。

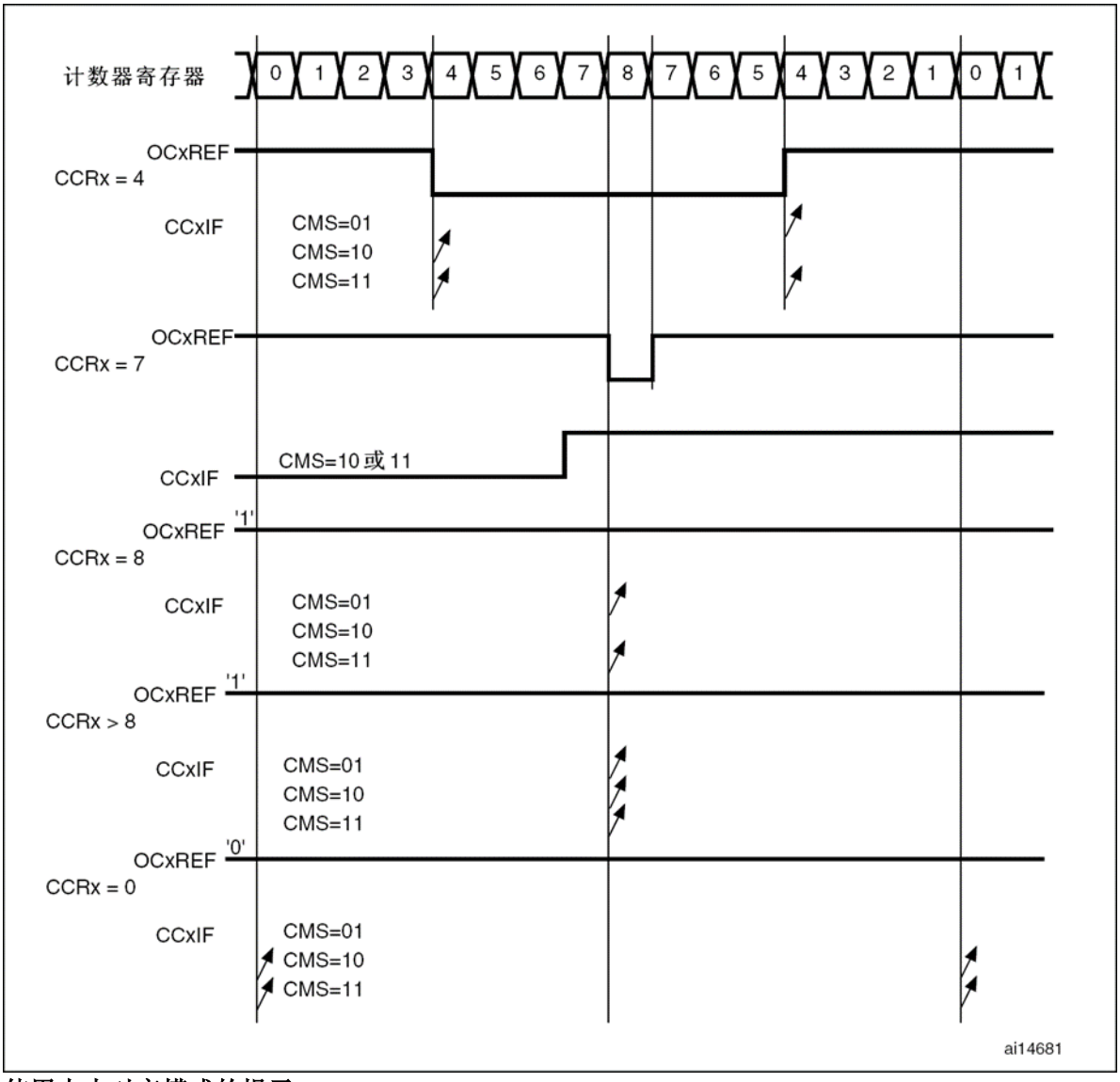
下图给出了一些中央对齐的 PWM 波形的例子

●TIMx_ARR=8

●PWM 模式 1

●TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图：中央对齐的 PWM 波形(APR=8)



使用中央对齐模式的提示：

●进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。

●不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：

—如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。

—如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

●使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

10.4.10单脉冲模式

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

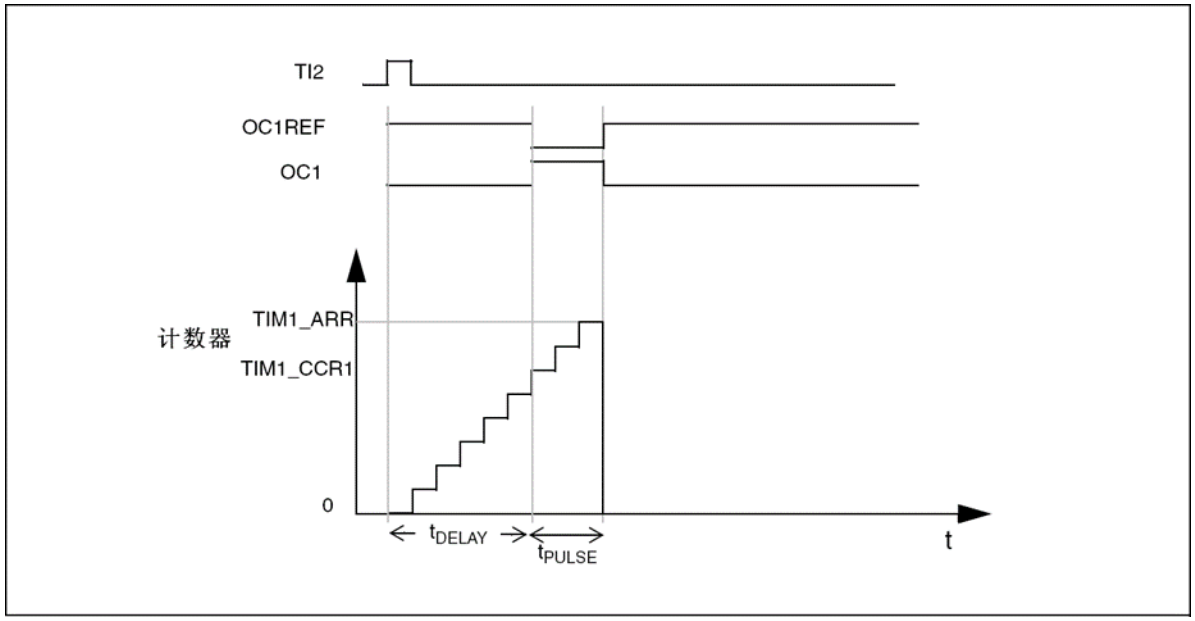
可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)， 必须如下配置：

●向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，

●向下计数方式：计数器 CNT > CCRx。

图：单脉冲模式的例子



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

假定 TI2FP2 作为触发 1:

●置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

●置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

●置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

●置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

●tDELAY 由 TIMx_CCR1 寄存器中的值定义。

●tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。

●假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；

首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

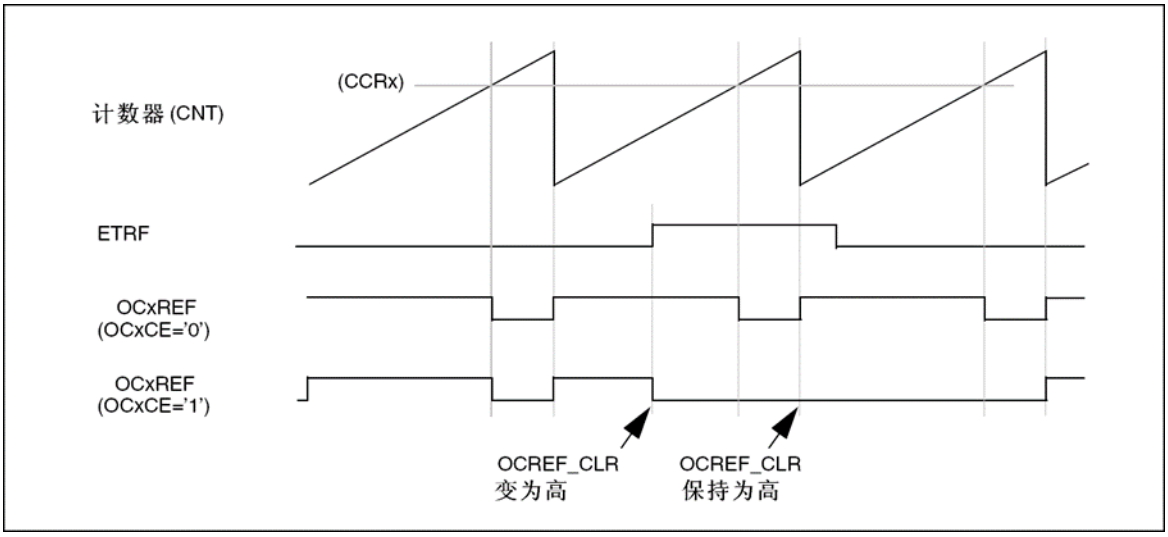
特殊情况：OCx 快速使能：

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM1 和 PWM2 模式时起作用。

10.4.11在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为’1’，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。
该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。
例如，OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：
1.外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。
2.必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。
3.外部触发极性(ETP)和外部触发滤波器(ETF)可以根据需要配置。
下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作。在这个例子中，定时器 TIMx 被置于 PWM 模式。
图：清除 TIMx 的 OCxREF



10.4.12编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI2 的边沿计数，则置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。
通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。
两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 73，假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1， TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。
编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。
在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合， 假设 TI1 和 TI2 不同时变换。
表：计数方向与编码器信号的关系

	相对信号的电平	TI1FP1信号	TI2FP2信号

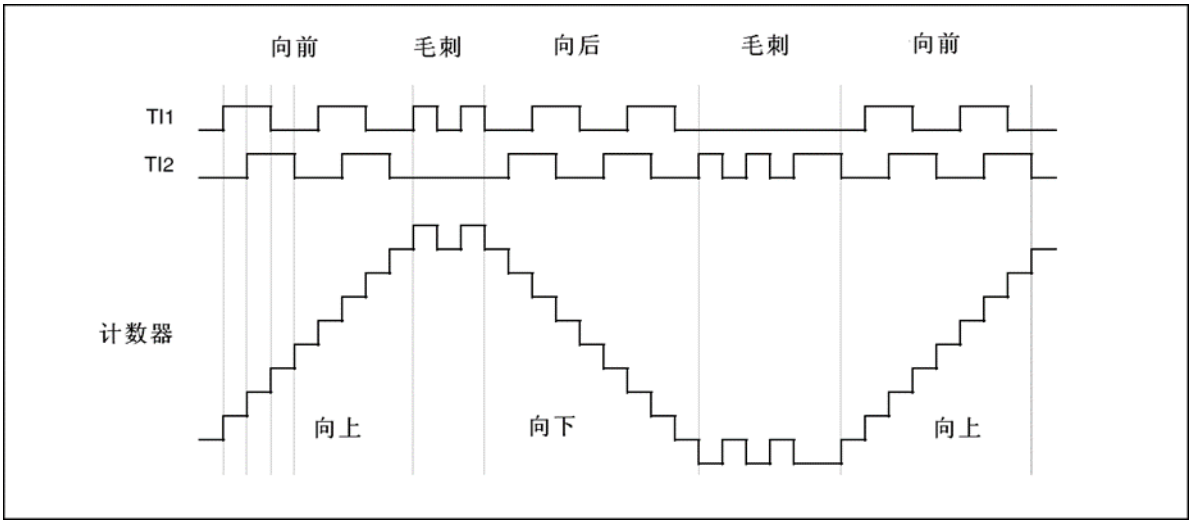
有效边沿	(TI1FP1对应TI2, TI2FP2对应TI1)	上升	下降	上升	下降
仅在TI1计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在TI2计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在TI1和TI2上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

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

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

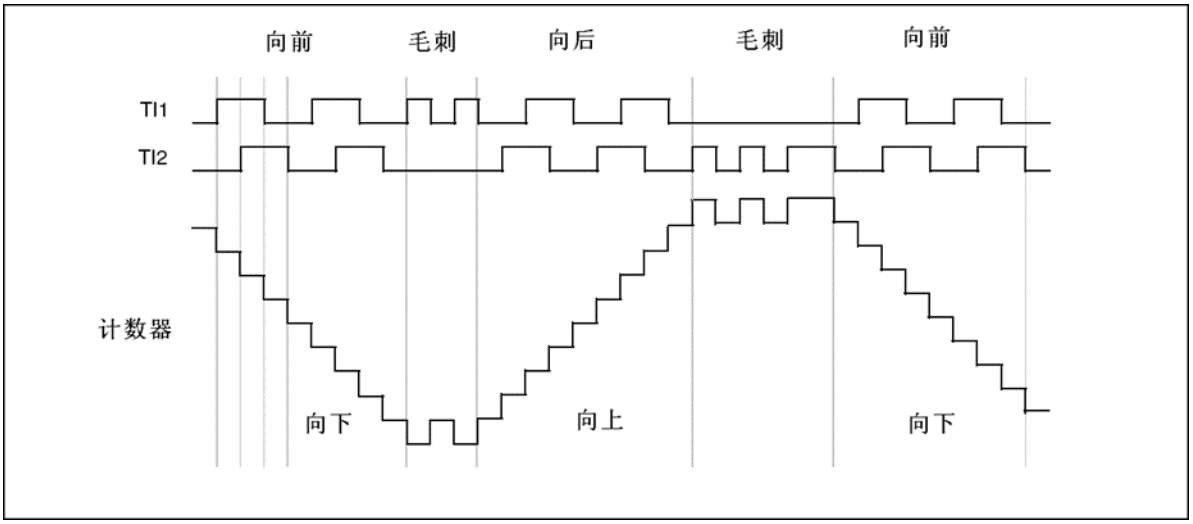
●CC1S=’01’ (TIMx_CCMR1 寄存器，IC1FP1 映射到 TI1)
●CC2S=’01’ (TIMx_CCMR2 寄存器，IC2FP2 映射到 TI2)
●CC1P=’0’ (TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
●CC2P=’0’ (TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
●SMS=’011’ (TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效).
●CEN=’1’ (TIMx_CR1 寄存器，计数器使能)

图：编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例(CC1P=’1’，其他配置与上例相同)

图：IC1FP1 反相的编码器接口模式实例



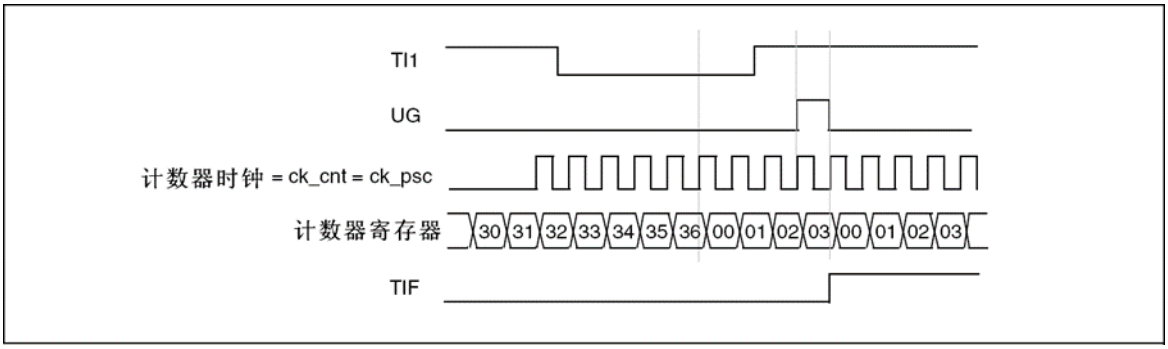
当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息(速度，加速度，减速度)。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，你可以把计数器的值锁存到第三个输入捕获寄存器(捕获信号必须是周期的并且可以由另一个定时器产生)；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

10.4.13定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。
异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下节给出了此特性用于连接霍尔传感器的例子。

10.4.14TIMx 定时器和外部触发的同步

TIMx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。
从模式：复位模式
在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR，TIMx_CCRx)都被更新了。
在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：
●配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。
●置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
●置 TIMx_CR1 寄存器中 CEN=1，启动计数器。
计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。
下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。
图：复位模式下的控制电路



从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

●配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

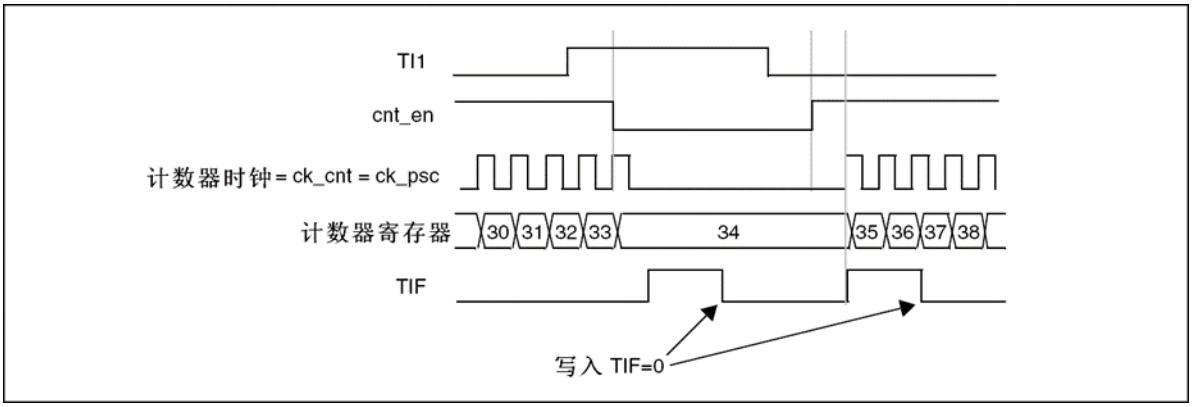
●置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

●置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

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

图：门控模式下的控制电路



从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

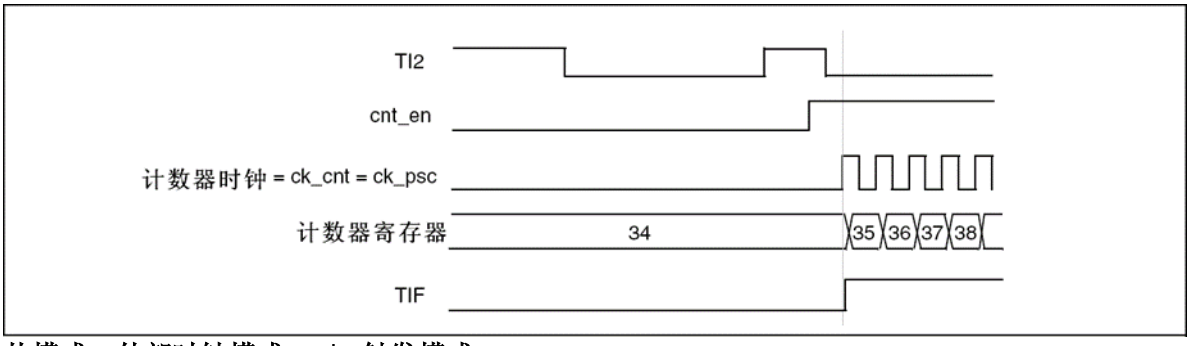
●配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。

●置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

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

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

图：触发器模式下的控制电路



从模式：外部时钟模式 2 ＋ 触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1.通过 TIMx_SMCR 寄存器配置外部触发输入电路：

—ETF=0000：没有滤波

—ETPS=00：不用预分频器

—ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。

2.按如下配置通道 1，检测 TI 的上升沿：

—IC1F=0000：没有滤波

—触发操作中不使用捕获预分频器，不需要配置

—置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源

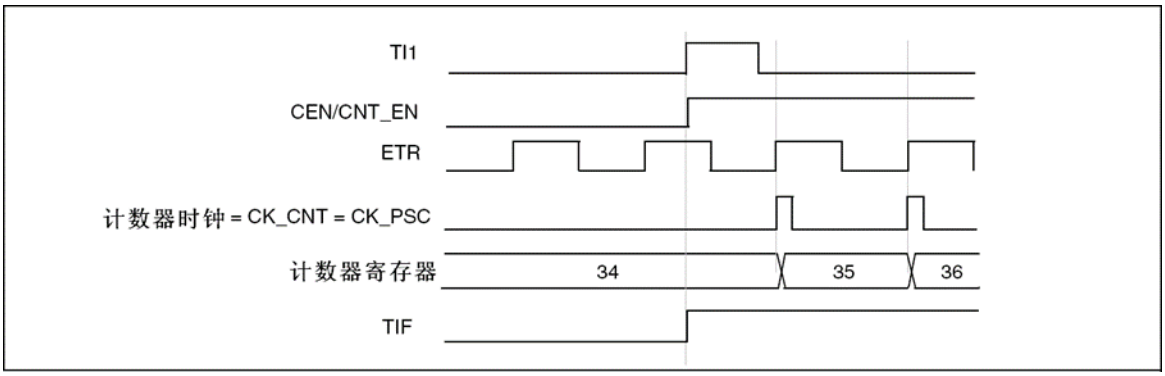
—置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)

3.置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

图：外部时钟模式 2＋触发模式下的控制电路



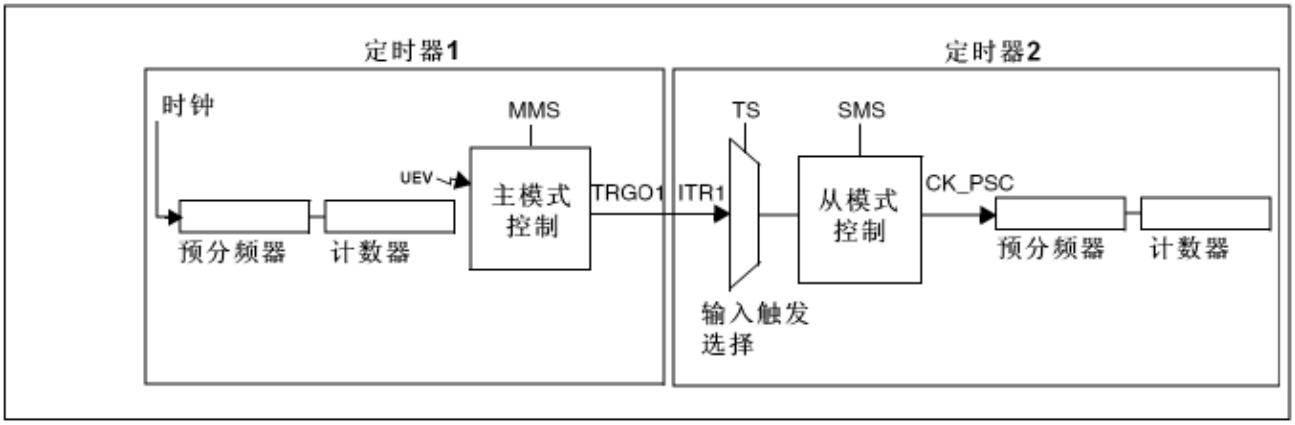
10.4.15定时器同步

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

下图显示了触发选择和主模式选择模块的概况。

使用一个定时器作为另一个定时器的预分频器

图：主/从定时器的例子



如：可以配置定时器 1 作为定时器 2 的预分频器。参考上图，进行下述操作：

● 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM1_CR2 寄存器的 MMS=’010’时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。

● 连接定时器 1 的 TRGO1 输出至定时器 2，设置 TIM2_SMCR 寄存器的 TS=’000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。

● 然后把从模式控制器置于外部时钟模式 1(TIM2_SMCR 寄存器的 SMS=111)；这样定时器 2 即可由定时器 1 周期性的上升沿(即定时器 1 的计数器溢出)信号驱动。

● 最后，必须设置相应(TIMx_CR1 寄存器)的 CEN 位分别启动两个定时器。

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

使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考上图的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/3)得到。

● 配置定时器 1 为主模式，送出它的输出比较参考信号(OC1REF)为触发输出(TIM1_CR2 寄存器的 MMS=100)

● 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)

● 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)

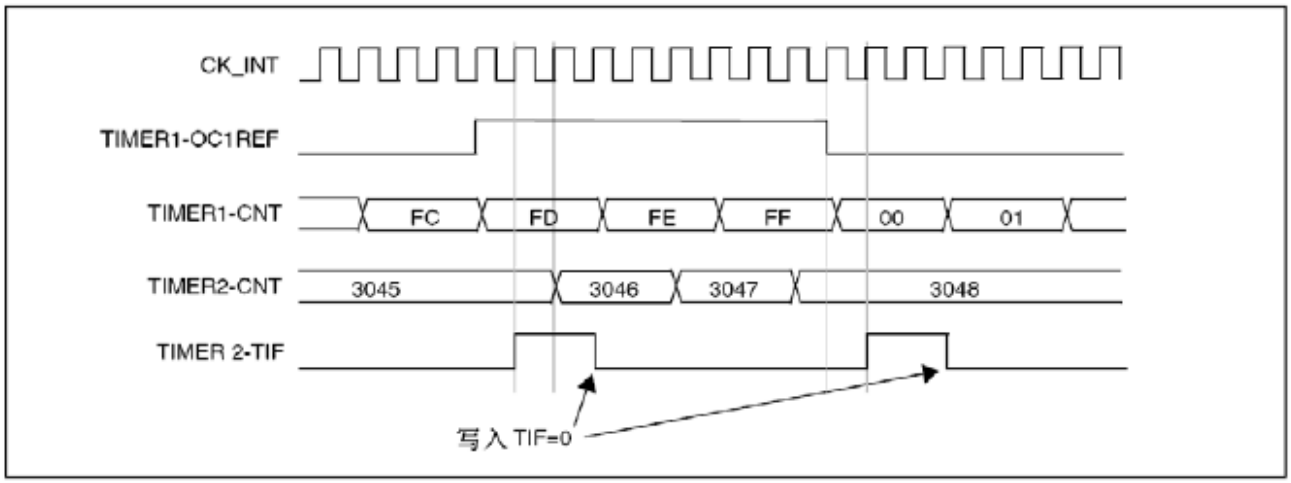
● 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)

● 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2

● 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

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

图：定时器 1 的 OC1REF 控制定时器 2

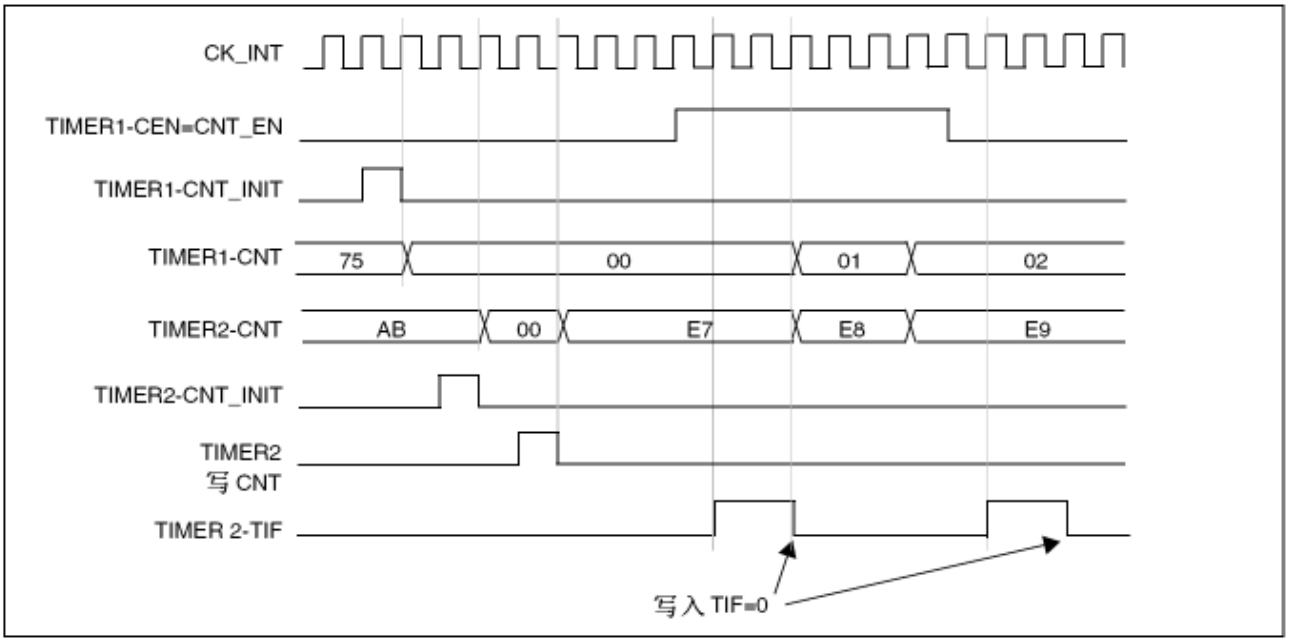


在上图的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 1 之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入

需要的任意数值。写 TIMx_EGR 寄存器的 UG 位即可复位定时器。 在下一个例子中，需要同步定时器 1 和定时器 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写’0’到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 2 随即停止。

- 配置定时器 1 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM1_CR2 寄存器的 MMS=100)。
- 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM1_EGR 寄存器的 UG=’1’，复位定时器 1。
- 置 TIM2_EGR 寄存器的 UG=’1’，复位定时器 2。
- 写’0xE7’至定时器 2 的计数器(TIM2_CNTL)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN=’1’以使能定时器 2。
- 置 TIM1_CR1 寄存器的 CEN=’1’以启动定时器 1。
- 置 TIM1_CR1 寄存器的 CEN=’0’以停止定时器 1。

图：通过使能定时器 1 可以控制定时器 2

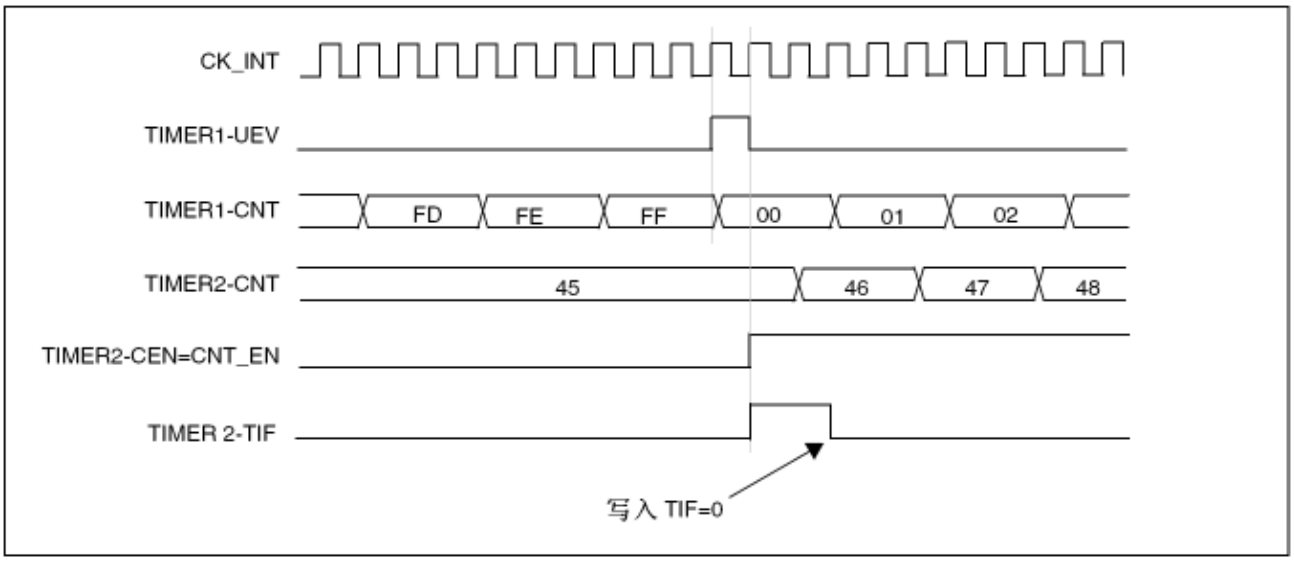


使用一个定时器去启动另一个定时器

在这个例子中，使用定时器 1 的更新事件使能定时器 2。参考图 138 的连接。一旦定时器 1 产生更新事件，定时器 2 即从它当前的数值(可以是非 0)按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CEN 位被自动地置’1’，同时计数器开始计数直到写’0’到 TIM2_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3(fCK_CNT=fCK_INT/3)。

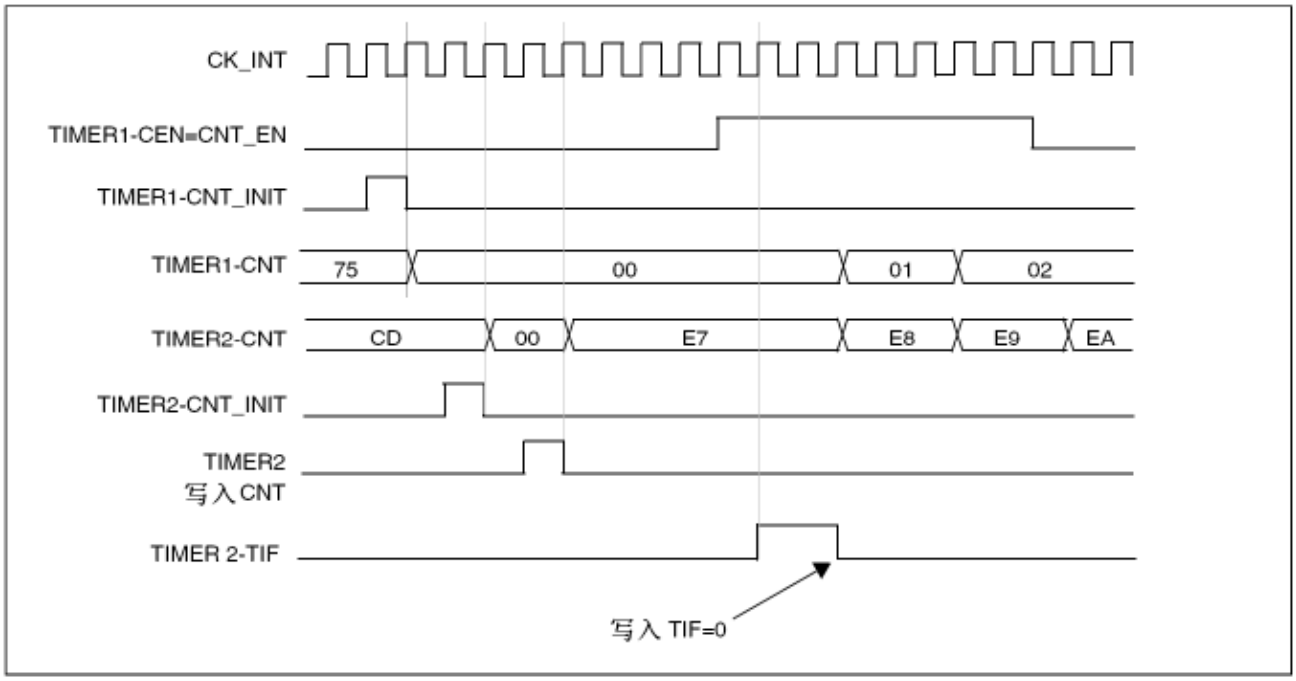
- 配置定时器 1 为主模式，送出它的更新事件(UEV)做为触发输出(TIM1_CR2 寄存器的 MMS=010)。
- 配置定时器 1 的周期(TIM1_ARR 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=110)
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

图：使用定时器 1 的更新触发定时器 2



在上一个例子中，可以在启动计数之前初始化两个计数器。显示在与 0 相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

图：利用定时器 1 的使能触发定时器 2



使用一个定时器作为另一个的预分频器

这个例子使用定时器 1 作为定时器 2 的预分频器。参考图 138 的连接，配置如下：

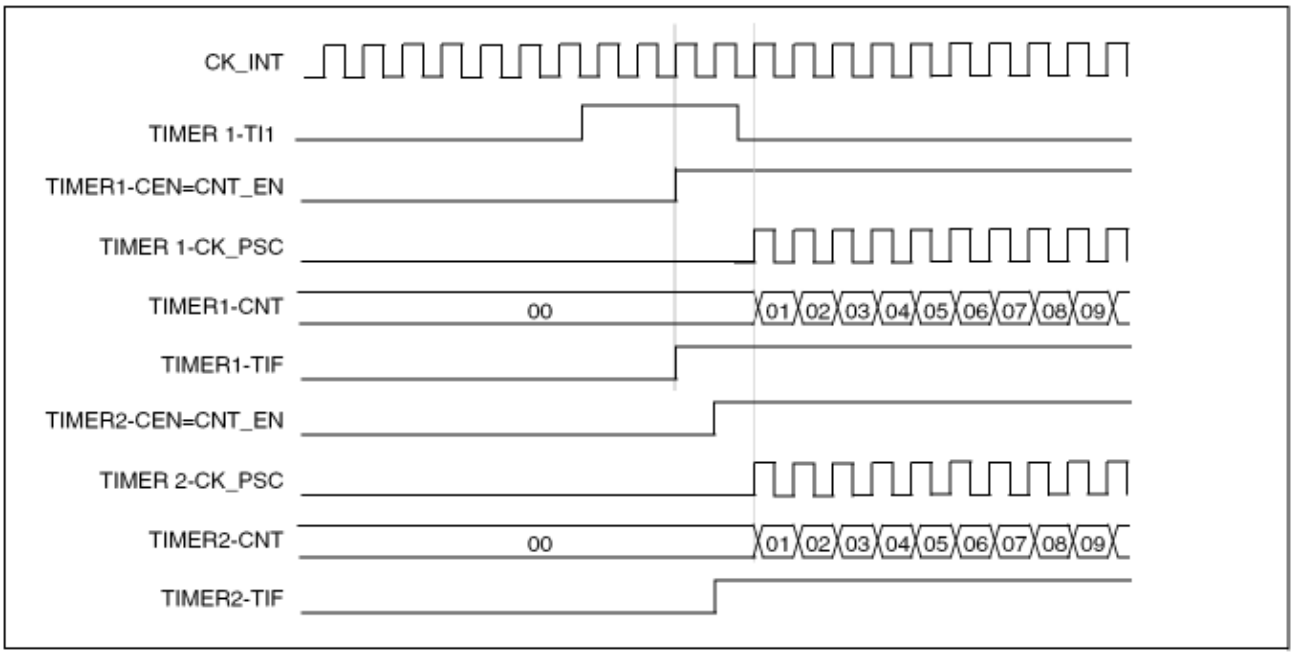
- 配置定时器 1 为主模式，送出它的更新事件 UEV 做为触发输出(TIM1_CR2 寄存器的 MMS=’010’)。然后每次计数器溢出时输出一个周期信号。
- 配置定时器 1 的周期(TIM1_ARR 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 使用外部时钟模式(TIM2_SMCR 寄存器的 SMS=111)
- 置 TIM1_CR2 寄存器的 CEN=1 以启动定时器 2。
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。 使用一个外部触发同步地启动 2 个定时器这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见上图。为保证计数器的对齐，定时器 1 必须配置为主/从模式(对应 TI1 为从，对应定时器 2 为主)：
- 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM1_CR2 寄存器的 MMS=’001’)。

- 配置定时器 1 为从模式，从 TI1 获得输入触发(TIM1_SMCR 寄存器的 TS=’100’)。
- 配置定时器 1 为触发模式(TIM1_SMCR 寄存器的 SMS=’110’)。
- 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM=’1’。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=’110’)。

当定时器 1 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被设置。

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

图：使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2



10.5 通用 TIMx 寄存器

10.5.1 控制寄存器 1(TIMx_CR1)

控制寄存器 1(TIMx_CR1)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4)					
			偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8

Read: Write:	X	X	X	X	X	X	CKD[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	ARPE	CMS[6:5]		DIR	OPM	URS	UDIS	CEN
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CKD[9:8]	**时钟分频因子（Clock division）** 这 2 位定义在定时器时钟(CK_INT)频率与数字滤波器(ETR,TIx)所用的采样时钟之间的分频比例。 00：tDTS = tCK_INT 01：tDTS = 2 x tCK_INT 10：tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
ARPE	**自动重装载预装载允许位（Auto-reload preload enable）** 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
CMS[6:5]	**选择中央对齐模式（Center-aligned mode selection）** 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
DIR	**计数方向（Direction）** 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
OPM	**单脉冲模式（One pulse mode）** 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
URS	**更新请求源（Update request source）** 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。

UDIS	**禁止更新（Update disable）** 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器) 1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
CEN	**使能计数器（Counter enable）** 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动 地通过硬件设置 CEN 位。

10.5.2 控制寄存器 2(TIMx_CR2)

控制寄存器 2(TIMx_CR2)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TI1S	MMS[6:4]			CCDS	X	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TI1S	**TI1 选择（TI1 selection）** 0：TIMx_CH1 引脚连到 TI1 输入；

	1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
MMS[6:4]	**主模式选择(Master mode selection)** 这3位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：**复位** – TIMx_EGR寄存器的UG位被用于作为触发输出(TRGO)。如果是触发输入产生的 复位(从模式控制器处于复位模式)，则TRGO上的信号相对实际的复位会有一个延迟。 001：**使能** – 计数器使能信号CNT_EN被用于作为触发输出(TRGO)。有时需要在同一时间启动 多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过CEN控制位和门控模式 下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO上会有一个延 迟，除非选择了主/从模式(见TIMx_SMCR寄存器中MSM位的描述)。 010：**更新** – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从 定时器的预分频器。 011：**比较脉冲** – 在发生一次捕获或一次比较成功时，当要设置CC1IF标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：**比较** – OC1REF信号被用于作为触发输出(TRGO)。 101：**比较** – OC2REF信号被用于作为触发输出(TRGO)。 110：**比较** – OC3REF信号被用于作为触发输出(TRGO)。 111：**比较** – OC4REF 信号被用于作为触发输出(TRGO)。
CCDS	**捕获/比较的 DMA 选择 (Capture/compare DMA selection)** 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。

10.5.3 从模式控制寄存器(TIMx_SMCR)

TIM1 和 TIM8 和 TIM20 从模式控制寄存器 (TIMx_SMCR)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x08					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	ETP	ECE	ETPS[13:12]		ETF[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read:	MSM	TS[6:4]			X	SMS[2:0]		

Write:								
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ETP	**外部触发极性 (External trigger polarity)** 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
ECE	**外部时钟使能位 (External clock enable)** 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。
ETPS[13:12]	**外部触发预分频 (External trigger prescaler)** 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使 用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
ETF[11:8]	**外部触发滤波 (External trigger filter)** 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个 事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 fDTS 采样 0001：采样频率 fSAMPLING=fCK_INT，N=2 0010：采样频率 fSAMPLING=fCK_INT，N=4 0011：采样频率 fSAMPLING=fCK_INT，N=8 0100：采样频率 fSAMPLING=fDTS/2，N=6 0101：采样频率 fSAMPLING=fDTS/2，N=8 0110：采样频率 fSAMPLING=fDTS/4，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1000：采样频率 fSAMPLING=fDTS/8，N=6 1001：采样频率 fSAMPLING=fDTS/8，N=8 1010：采样频率 fSAMPLING=fDTS/16，N=5 1011：采样频率 fSAMPLING=fDTS/16，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 1110：采样频率 fSAMPLING=fDTS/32，N=6 1111：采样频率 fSAMPLING=fDTS/32，N=8
MSM	**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。
TS[6:4]	**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 TIMx 内部触发连接。 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
SMS[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 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表：TIMx 内部触发连接

从定时器	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM2	TIM1	TIM8	TIM3	TIM4
TIM3	TIM1	TIM2	TIM5	TIM4
TIM4	TIM1	TIM2	TIM3	TIM8
TIM5	TIM5	TIM3	TIM4	TIM8

10.5.4 DMA/中断使能寄存器(TIMx_DIER)

DMA/中断使能寄存器 (TIMx_DIER)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X

Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	X	TDE	X	CC4DE	CC3DE	CC2DE	CC1DE	UDE
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	X	TIE	X	CC4IE	CC3IE	CC2IE	CC1IE	UIE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TDE	**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。
CC4DE	**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。
CC3DE	**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。
CC2DE	**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。
CC1DE	**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。
UDE	**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。
TIE	**触发中断使能 (Trigger interrupt enable)** 0：禁止触发中断； 1：使能触发中断。
CC4IE	**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。
CC3IE	**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。
CC2IE	**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。
CC1IE	**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。
UIE	**允许更新中断 (Update interrupt enable)** 0：禁止更新中断； 1：允许更新中断。

10.5.5 状态寄存器(TIMx_SR)

状态寄存器(TIMx_SR)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4)					
			偏移地址： 0x10					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	CC4OF	CC3OF	CC2OF	CC1OF	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	TIF	X	CC4IF	CC3IF	CC2IF	CC1IF	UIF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC4OF	捕获/比较4重复捕获标记(Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。
CC3OF	捕获/比较3重复捕获标记(Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。
CC2OF	捕获/比较2重复捕获标记(Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
CC1OF	捕获/比较1重复捕获标记(Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
TIF	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。
CC4IF	捕获/比较4中断标记(Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
CC3IF	捕获/比较3中断标记(Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
CC2IF	捕获/比较2中断标记(Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。
CC1IF	捕获/比较1中断标记(Capture/Compare 1 interrupt flag) 如果通道CC1配置为输出模式： 当计数器值与比较值匹配时该位由硬件置1，但在中心对称模式下除外(参考TIMx_CR1寄存器的CMS位)。它由软件清’0’。 0：无匹配发生；

	1：TIMx_CNT的值与TIMx_CCR1的值匹配。 **如果通道CC1配置为输入模式：** 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读TIMx_CCR1清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIF	**更新中断标记(Update interrupt flag)** 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若TIMx_CR1寄存器的URS=0、UDIS=0，当设置TIMx_EGR寄存器的UG=1时产生更新事件，通过软件对计数器CNT重新初始化时。 – 若TIMx_CR1寄存器的URS=0、UDIS=0，当计数器CNT被触发事件重新初始化时。

10.5.6 事件产生寄存器(TIMx_EGR)

事件产生寄存器(TIMx_EGR)			基地址：0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x14					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	TG	X	CC4G	CC3G	CC2G	CC1G	UG
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TG	**产生触发事件(Trigger generation)** 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
CC4G	**产生捕获/比较4事件(Capture/Compare 4 generation)** 参考 CC1G 描述。
CC3G	**产生捕获/比较3事件(Capture/Compare 3 generation)** 参考 CC1G 描述。
CC2G	**产生捕获/比较2事件(Capture/Compare 2 generation)** 参考 CC1G 描述。
CC1G	**产生捕获/比较 1 事件 (Capture/Compare 1 generation)** 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作；

	1：在通道 CC1 上产生一个捕获/比较事件： **若通道 CC1 配置为输出：** 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 **若通道 CC1 配置为输入：** 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
UG	**产生更新事件（Update generation）** 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清’0’（但是预分频系 数不变）。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

10.5.7 捕获/比较模式寄存器 1(TIMx_CCMR1)

捕获/比较模式寄存器 1(TIMx_CCMR1)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x18					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	OC2CE	OC2M[14:12]			OC2PE	OC2FE	CC2S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	OC1CE	OC1M[6:4]			OC1PE	OC1FE	CC1S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输出比较模式

位	功能描述
OC2CE	**输出比较 2 清 0 使能**
OC2M[14:12]	**输出比较 2 模式**
OC2PE	**输出比较 2 预装载使能**
OC2FE	**输出比较 2 快速使能**
CC2S[9:8]	**捕获/比较 2 选择。(Capture/Compare 2 selection)** 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出；

	01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
OC1CE	**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。
OC1M[6:4]	**输出比较 1 模式 (Output Compare 1 mode)** 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P CC1NP. 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001 ：匹配 时 设置通道 1 为 有效电平。 当计数器 TIMx_CNT 的值与 捕获 / 比较寄 存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010 ：匹配 时 设置通道 1 为 无效电平。 当计数器 TIMx_CNT 的值与 捕获 / 比较寄 存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1－ 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2－ 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输 出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式 切换到 PWM 模式时，OC1REF 电平才改变。
OC1PE	**输出比较 1 预装载使能 (Output Compare 1 preload enable)** 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数 值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的 预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输 出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
OC1FE	**输出比较 1 快速使能 (Output Compare 1 fast enable)** 该位用于加快 CC 输出对触发输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有 效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。

	1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与 比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
CC1S[1:0]	**捕获/比较 1 选择**(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

捕获/比较模式寄存器 1(TIMx_CCMR1)			**基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4)** **偏移地址： 0x18**					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	IC2F[15:12]				IC2PSC[11:10]		CC2S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	IC1F[7:4]				IC1PSC[3:2]		CC1S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输入捕获模式

位	功能描述
IC2F[15:12]	**输入捕获 2 滤波器**
IC2PSC[11:10]	**输入/捕获 2 预分频器**
CC2S[9:8]	**捕获/比较 2 选择** (Capture/Compare 2 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入

	被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
IC1F[7:4]	**输入捕获 1 滤波器** 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 0001：采样频率 fSAMPLING=fCK_INT，N=2 0010：采样频率 fSAMPLING=fCK_INT，N=4 0011：采样频率 fSAMPLING=fCK_INT，N=8 0100：采样频率 fSAMPLING=fDTS/2，N=6 0101：采样频率 fSAMPLING=fDTS/2，N=8 0110：采样频率 fSAMPLING=fDTS/4，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1000：采样频率 fSAMPLING=fDTS/8，N=6 1001：采样频率 SAMPLING=fDTS/8，N=8 1010：采样频率 SAMPLING=fDTS/16，N=5 1011：采样频率 SAMPLING=fDTS/16，N=6 1100：采样频率 SAMPLING=fDTS/16，N=8 1101：采样频率 SAMPLING=fDTS/32，N=5 1110：采样频率 SAMPLING=fDTS/32，N=6 1111：采样频率 SAMPLING=fDTS/32，N=8
IC1PSC[3:2]	**输入/捕获 1 预分频器** (Input capture 1 prescaler) 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
CC1S[1:0]	**捕获/比较 1 选择** (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

10.5.8 捕获/比较模式寄存器 2(TIMx_CCMR2)

捕获/比较模式寄存器 2(TIMx_CCMR2)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OC4CE	OC4M[14:12]			OC4PE	OC4FE	CC4S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OC3CE	OC3M[6:4]			OC3PE	OC3FE	CC3S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输出比较模式

位	功能描述
OC4CE	**输出比较 4 清 0 使能**
OC4M[14:12]	**输出比较 4 模式**
OC4PE	**输出比较 4 预装载使能**
OC4FE	**输出比较 4 快速使能**
CC4S[9:8]	**捕获/比较 4 选择 (Capture/Compare 4 selection)** 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
OC3CE	**输出比较 3 清 0 使能**
OC3M[6:4]	**输出比较 3 模式**
OC3PE	**输出比较 3 预装载使能**
OC3FE	**输出比较 3 快速使能**
CC3S[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器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

捕获/比较模式寄存器 2(TIMx_CCMR2)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X

Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	IC4F[15:12]				IC4PSC[11:10]		CC4S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	IC3F[7:4]				IC3PSC[3:2]		CC3S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输入捕获模式

位	功能描述
IC4F[15:12]	**输入捕获 4 滤波器**
IC4PSC[11:10]	**输入/捕获 4 预分频器**
CC4S[9:8]	**捕获/比较 4 选择 (Capture/Compare 4 selection)** 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
IC3F[7:4]	**输入捕获 3 滤波器**
IC3PSC[3:2]	**输入/捕获 3 预分频器**
CC3S[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器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

10.5.9 捕获/比较使能寄存器(TIMx_CCER)

捕获/比较使能寄存器 (TIMx_CCER)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read:	X	X	X	X	X	X	X	X

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	CC4P	CC4E	X	X	CC3P	CC3E
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	CC2P	CC2E	X	X	CC1P	CC1E
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC4P	**输入/捕获4输出极性(Capture/Compare 4 output polarity)** 参考 CC1P 的描述。
CC4E	**输入/捕获4输出使能(Capture/Compare 4 output enable)** 参考 CC1E 的描述。
CC3P	**输入/捕获3输出极性(Capture/Compare 3 output polarity)** 参考 CC1P 的描述。
CC3E	**输入/捕获3输出使能(Capture/Compare 3 output enable)** 参考 CC1E 的描述。
CC2P	**输入/捕获2输出极性(Capture/Compare 2 output polarity)** 参考 CC1P 的描述。
CC2E	**输入/捕获2输出使能(Capture/Compare 2 output enable)** 参考 CC1E 的描述。
CC1P	**输入/捕获1输出极性(Capture/Compare 1 output polarity)** **CC1通道配置为输出：** 0：OC1高电平有效； 1：OC1低电平有效。 **CC1通道配置为输入：** 该位选择是IC1还是IC1的反相信号作为触发或捕获信号。 0：不反相：捕获发生在IC1的上升沿；当用作外部触发器时，IC1不反相。 1：反相：捕获发生在IC1的下降沿；当用作外部触发器时，IC1反相。
CC1E	**输入/捕获 1 输出使能(Capture/Compare 1 output enable)** **CC1 通道配置为输出：** 0： 关闭－ OC1 禁止输出 1：开启－ OC1 信号输出到对应的输出引脚 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表:标准 OCx 通道的输出控制位

CCxE位	OCx输出状态
0	禁止输出(OCx=0，OCx_EN=0)
1	OCx = OCxREF + 极性，OCx_EN=1

注： 连接到标准 OCx 通道的外部 I/O 引脚状态，取决于 OCx 通道状态和 GPIO 以及 AFIO 寄存器。

10.5.10计数器(TIMx_CNT)

计数器(TIMx_CNT)			基地址：0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4)					
			偏移地址： 0x24					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CNT							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CNT							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CNT[15:0]	计数器的值

10.5.11预分频器(TIMx_PSC)

预分频器(TIMx_PSC)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4)					
			偏移地址： 0x28					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	PSC[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PSC[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PSC[15:0]	**预分频器的值（Prescaler value）** 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’。

10.5.12 自动重装载寄存器(TIMx_ARR)

自动重装载寄存器 (TIMx_ARR)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x2C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ARR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ARR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ARR	**自动重装载的值（Prescaler value）** ARR 包含了将要装载入实际的自动重装载寄存器的值。

	当自动重装载的值为空时，计数器不工作。

10.5.13捕获/比较寄存器 1(TIMx_CCR1)

捕获/比较寄存器 1(TIMx_CCR1)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x34					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR1[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR1[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR1[15:0]	**捕获/比较通道 1 的值 (Capture/Compare 1 value)** **若 CC1 通道配置为输出：** CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

10.5.14捕获/比较寄存器 2(TIMx_CCR2)

捕获/比较寄存器 2(TIMx_CCR2)	基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x38

	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR2[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR2[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR2[15:0]	捕获/比较通道 2 的值 (Capture/Compare 2 value) **若 CC2 通道配置为输出：** CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 **若 CC2 通道配置为输入：** CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

10.5.15捕获/比较寄存器 3(TIMx_CCR3)

捕获/比较寄存器 3(TIMx_CCR3)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x03C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR3[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR3[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR3[15:0]	**捕获/比较通道 3 的值 (Capture/Compare 3 value)** **若 CC3 通道配置为输出：** CCR3 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 **若 CC3 通道配置为输入：** CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。

10.5.16捕获/比较寄存器(TIMx_CCR4)

捕获/比较寄存器 (TIMx_CCR4)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x40					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR4[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR4[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述

CCR4[15:0]	**捕获/比较通道 4 的值 (Capture/Compare 4 value)** **若 CC4 通道配置为输出：** CCR4 包含了装入当前捕获/比较 4 寄存器的值(预装载值)。 如果在 TIMx_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 **若 CC4 通道配置为输入：** CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。

10.5.17DMA 控制寄存器(TIMx_DCR)

DMA 控制寄存器(TIMx_DCR)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x48					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	DBL[12:8]				
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	DBA[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DBL[12:8]	**DMA 连续传送长度 (DMA burst length)** 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)，即：定义传输的字节数目： 00000：1 个字节 00001：2 个字节 00010：3 个字节 …… …… 10001：18 个字节
DBA[4:0]	**DMA 基地址 (DMA base address)** 这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，

	DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， ……

10.5.18连续模式的 DMA 地址(TIMx_DMAR)

连续模式的 DMA 地址 (TIMx_DMAR)			基地址： 0x4000 0000(TIM2),0x4000 0400(TIM3) 0x4000 0800(TIM4) 偏移地址： 0x4C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DMAB[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DMAB[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DMAB[15:0]	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： TIMx_CR1 地址 + DBA + DMA 索引，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

11 高级定时器（TIM1 & TIM8 & TIM20）

11.1 TIM 概述

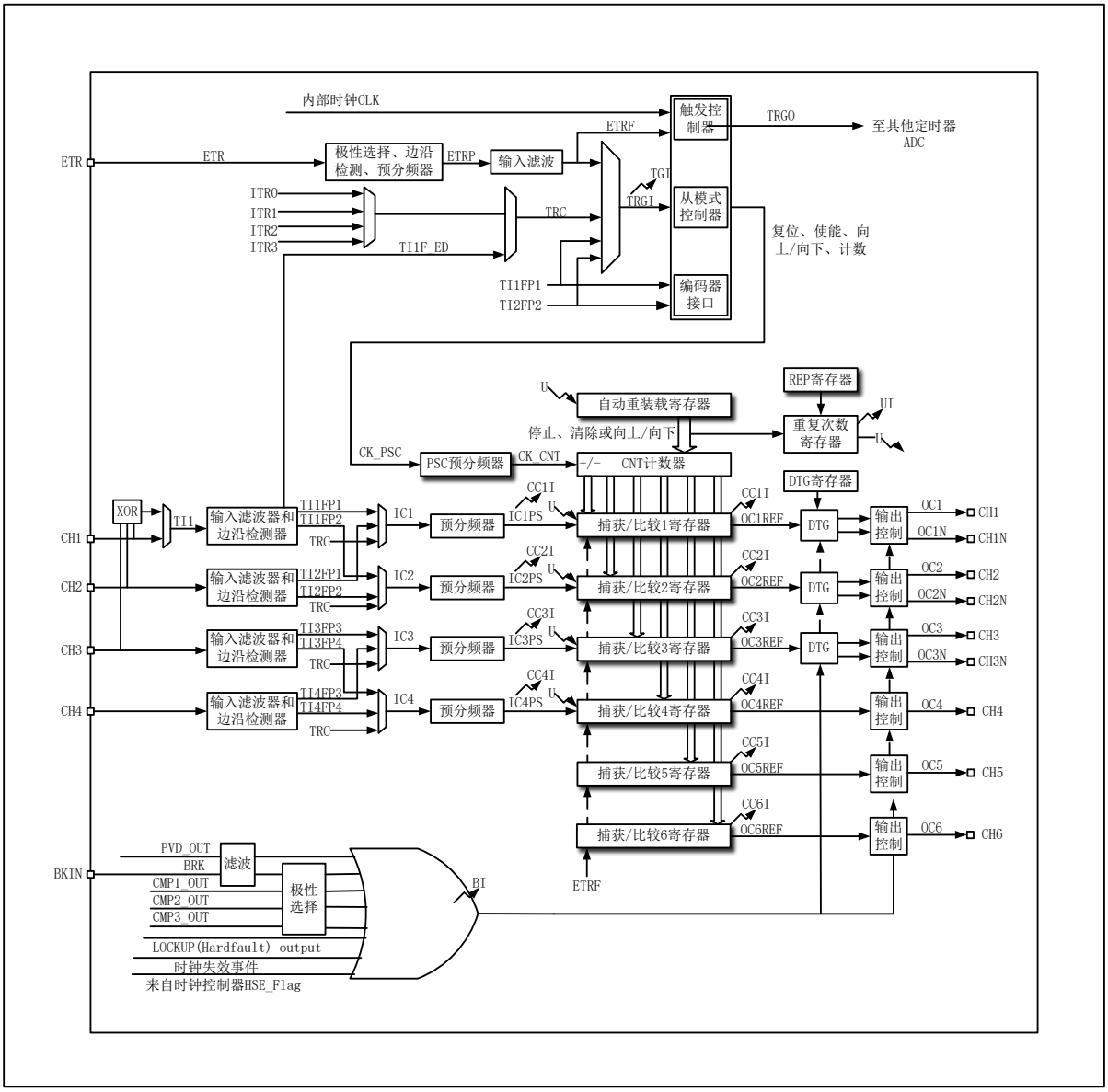
高级控制定时器(TIM1 和 TIM8 和 TIM20)由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

高级控制定时器(TIM1 和 TIM8 和 TIM20)和通用定时器(TIMx)是完全独立的，它们不共享任何资源。它们可以同步操作。

11.2 TIM1 和 TIM8 和 TIM20 主要特性

TIM1 和 TIM8 和 TIM20 定时器的功能包括：
●16 位向上、向下、向上/下自动装载计数器
●16 位可编程(可以实时修改)预分频器，计数器时钟频率的分频系数为 1～65535 之间的任意数值
●多达 4 个独立通道：
—输入捕获
—输出比较
—PWM 生成(边缘或中间对齐模式)
—单脉冲模式输出
●死区时间可编程的互补输出
●使用外部信号控制定时器和定时器互联的同步电路
●允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
●刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
●如下事件发生时产生中断/DMA：
—更新：计数器向上溢出/向下溢出，计数器初始化(通过软件或者内部/外部触发)
—触发事件(计数器启动、停止、初始化或者由内部/外部触发计数)
—输入捕获
—输出比较
—刹车信号输入
●支持针对定位的增量(正交)编码器和霍尔传感器电路
●触发输入作为外部时钟或者按周期的电流管理

11.3 TIM 框图



事件

中断和 DMA 输出

11.4 TIM1 和 TIM8 和 TIM20 功能描述

11.4.1 时基单元

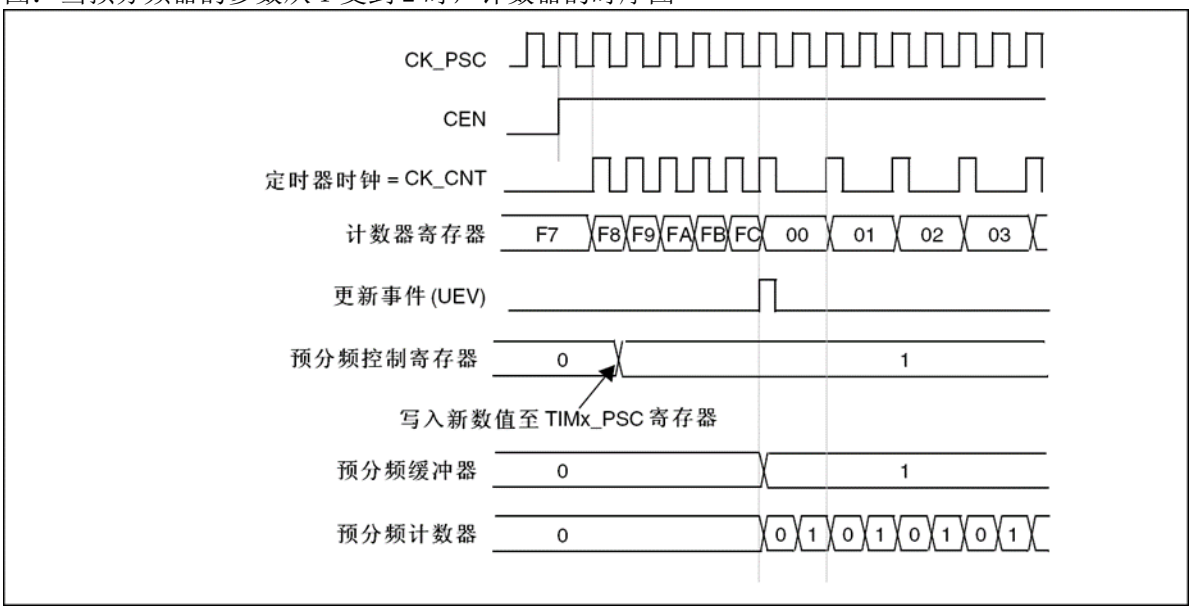
可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

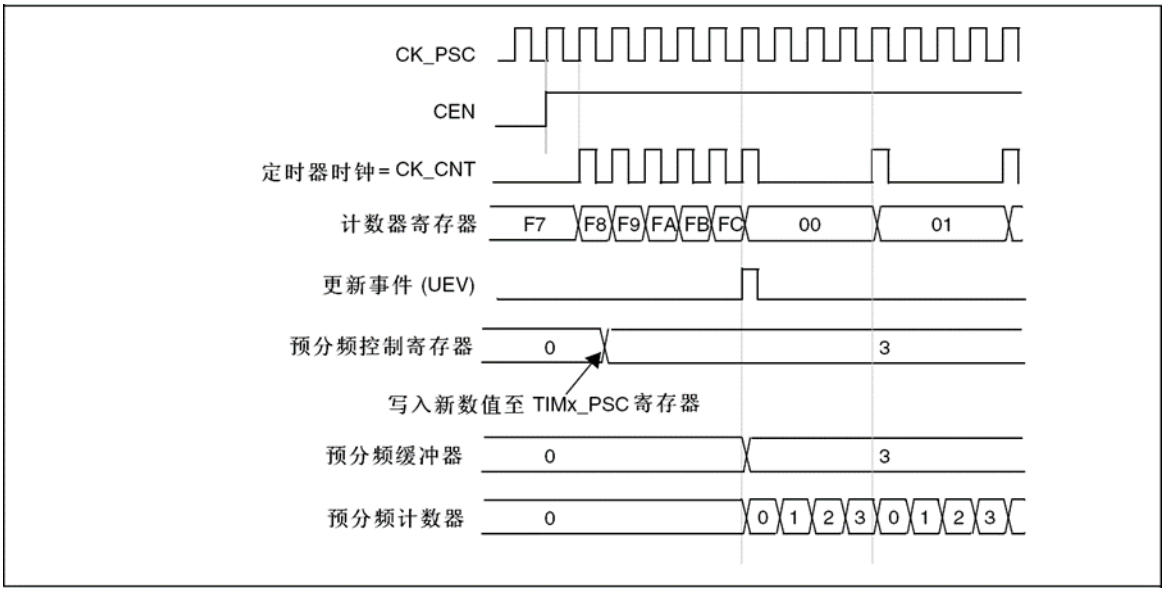
时基单元包含：
●计数器寄存器(TIMx_CNT)
●预分频器寄存器 (TIMx_PSC)
●自动装载寄存器 (TIMx_ARR)
●重复次数寄存器 (TIMx_RCR)
自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMx_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(向下计数时的下溢条件)并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMx_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。(更多有关使能计数器的细节，请参见控制器的从模式描述)。
注：在设置了 TIMx_CR 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 TIMx_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。
下图给出了在预分频器运行时，更改计数器参数的例子。
图：当预分频器的参数从 1 变到 2 时，计数器的时序图



图：当预分频器的参数从 1 变到 4 时，计数器的时序图



11.4.2 计数器模式

向上计数模式

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

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数(TIMx_RCR)时，产生更新事件(UEV)；否则每次计数器溢出时才产生更新事件。

在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

设置 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清’0’之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清’0’，同时预分频器的计数也被请 0(但预分频器的数值不变)。此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时(依据 URS 位)设置更新标志位(TIMx_SR 寄存器中的 UIF 位)。

●重复计数器被重新加载为 TIMx_RCR 寄存器的内容。

●自动装载影子寄存器被重新置入预装载寄存器的值(TIMx_ARR)。

●预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

图：计数器时序图，内部时钟分频因子为 1

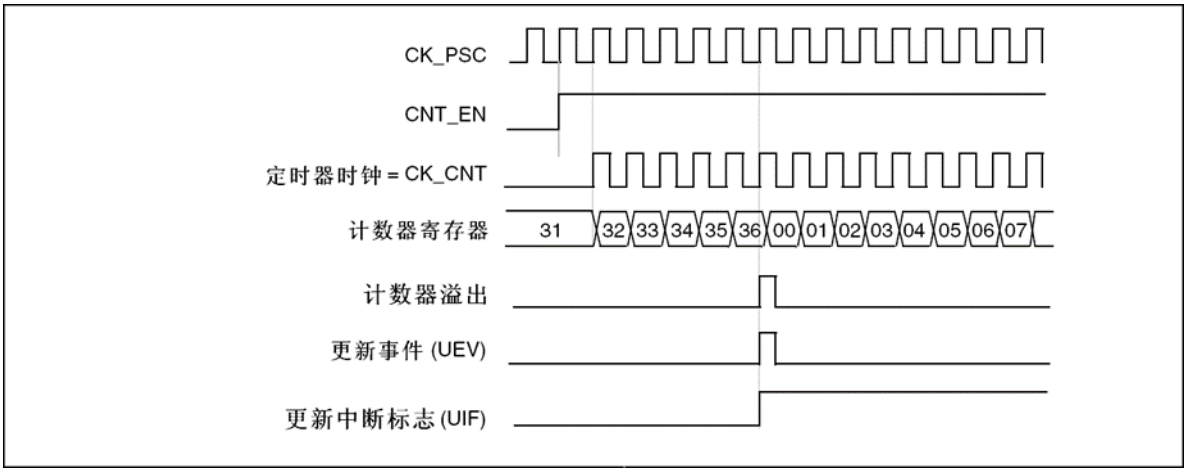
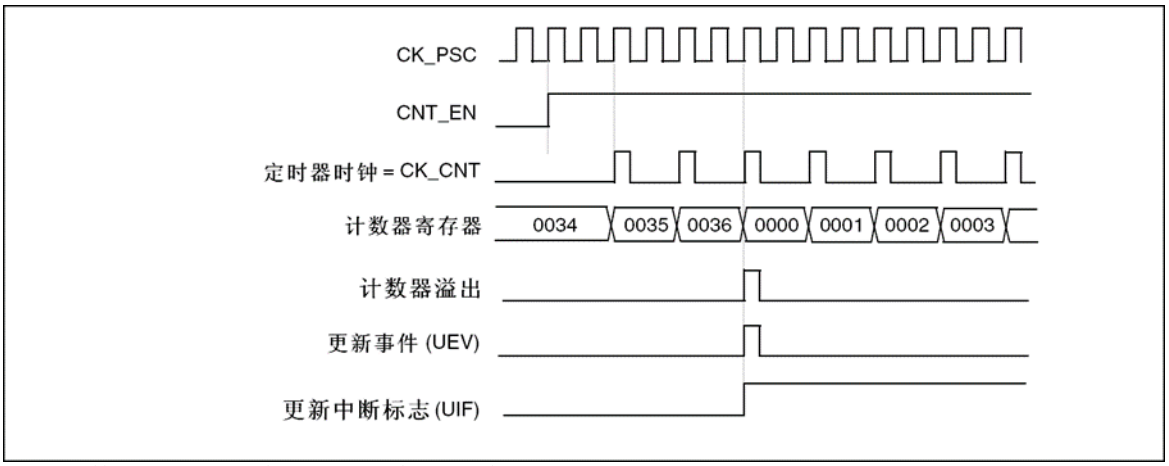
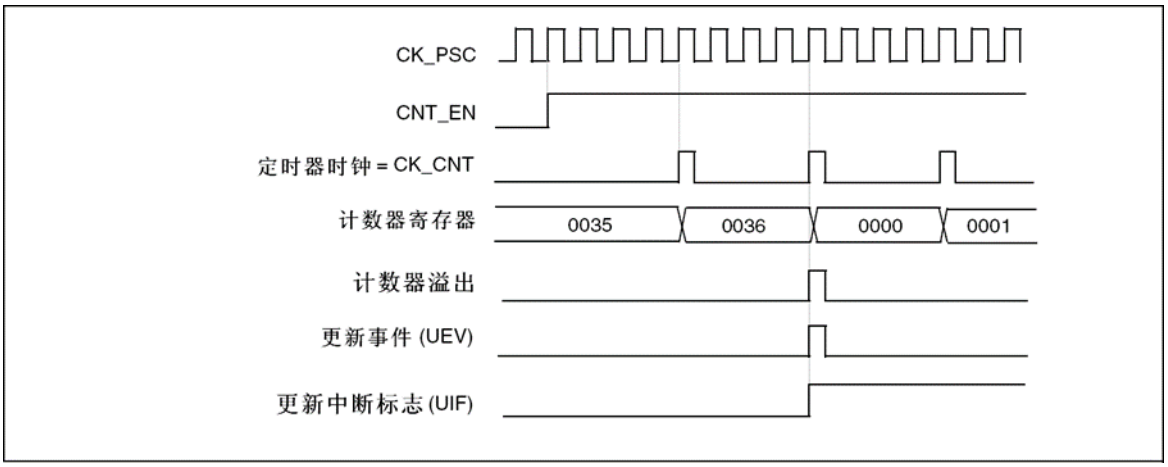


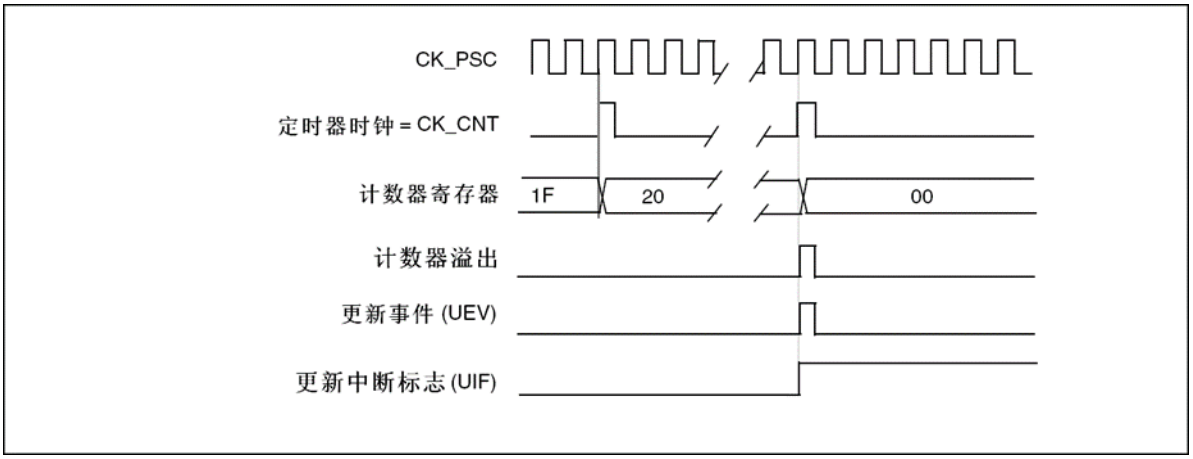
图: 计数器时序图，内部时钟分频因子为 2



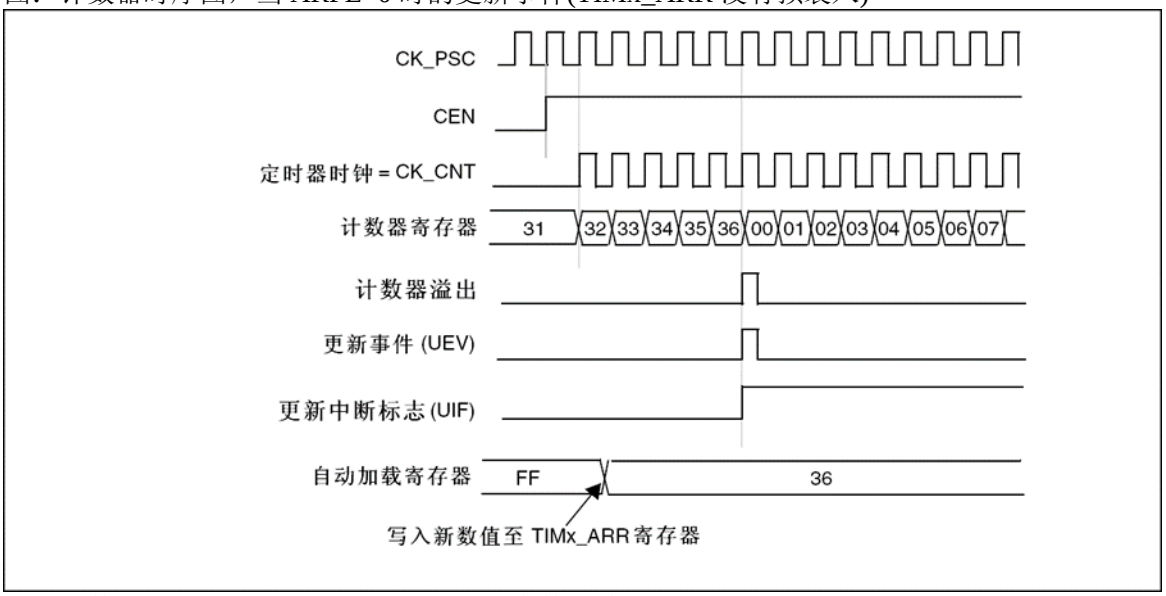
图：计数器时序图，内部时钟分频因子为 4



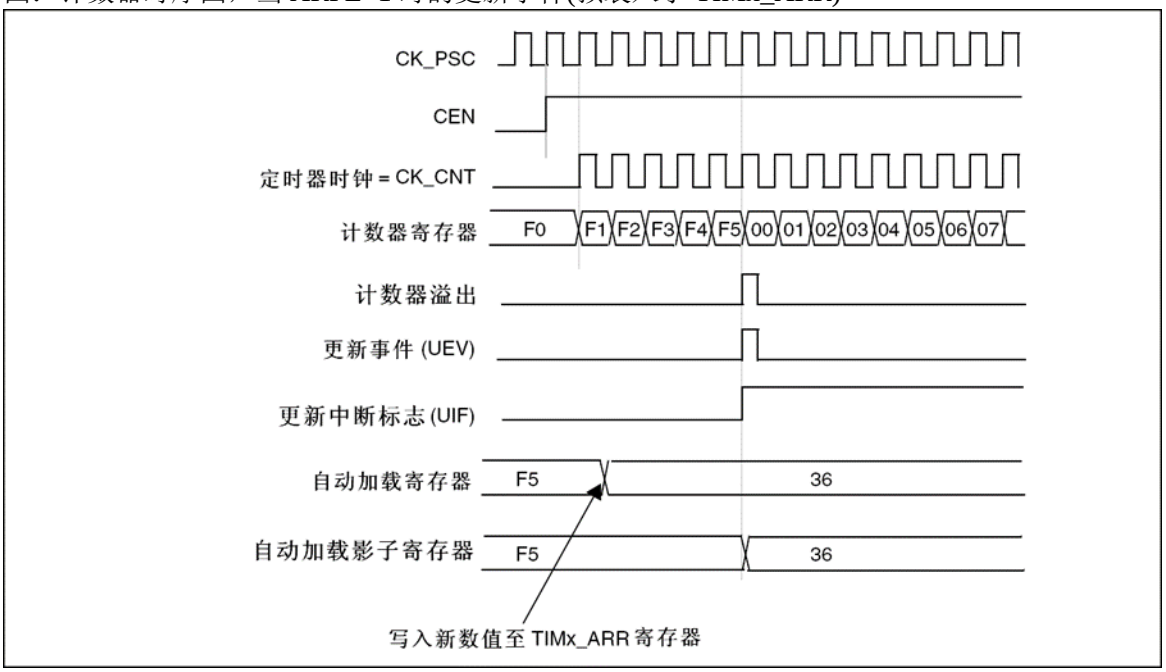
图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

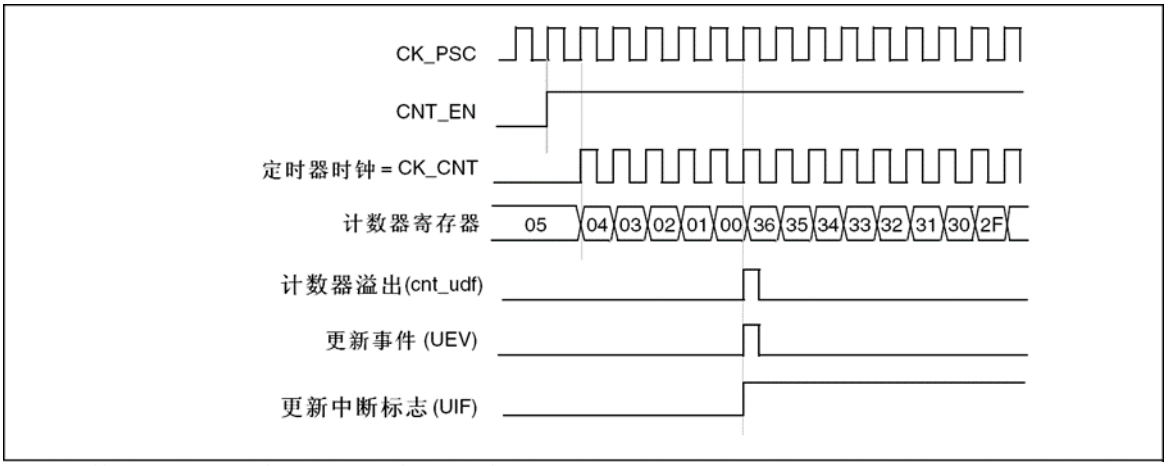


图：计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

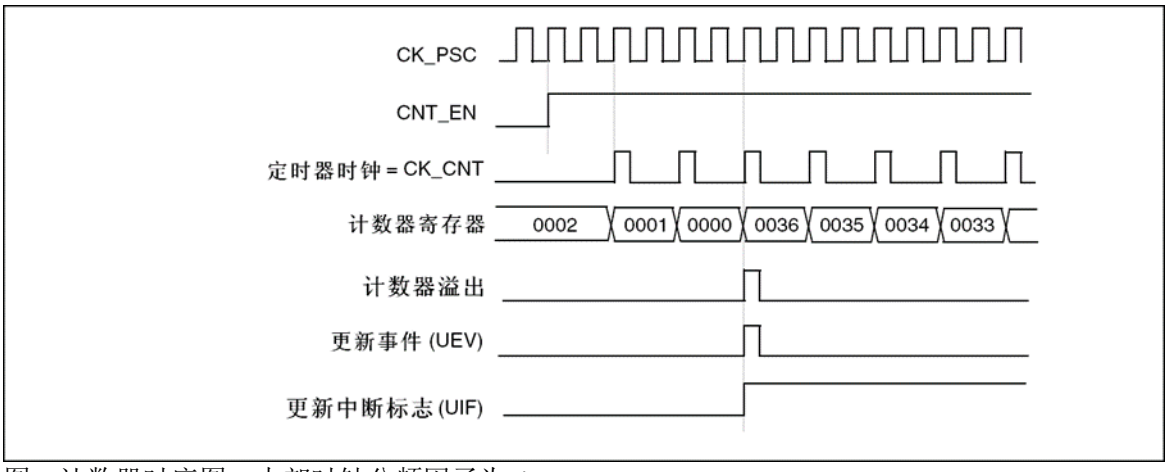


向下计数模式

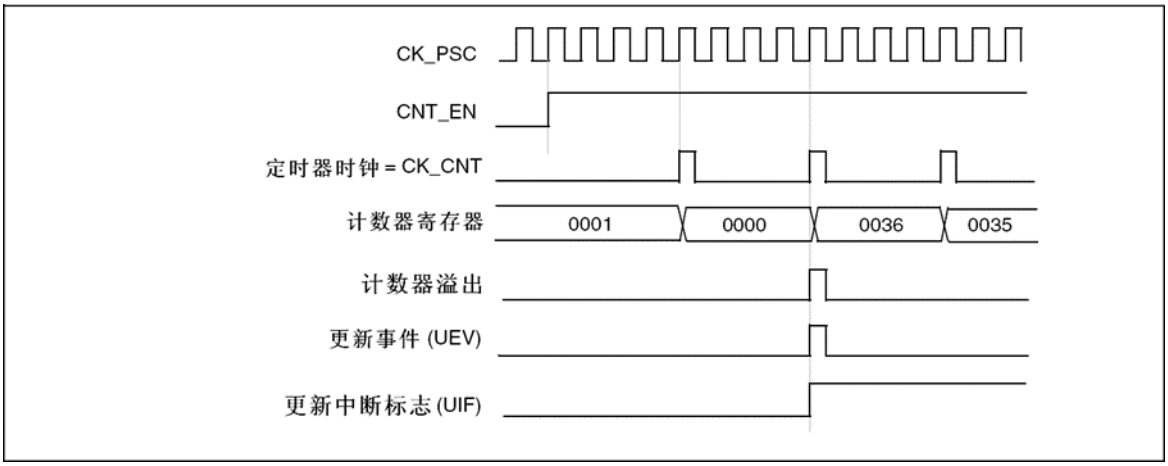
在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。
如果使用了重复计数器，当向下计数重复了重复计数寄存器(TIMx_RCR)中设定的次数后，将产生更新事件(UEV)，否则每次计数器下溢时才产生更新事件。
在 TIMx_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位，也同样可以产生一个更新事件。
设置 TIMx_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始(但预分频系数不变)。
此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。
当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位的设置)更新标志位(TIMx_SR 寄存器中的 UIF 位)也被设置。
●重复计数器被重置为 TIMx_RCR 寄存器中的内容
●预分频器的缓存器被加载为预装载的值(TIMx_PSC 寄存器的值)。
●当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。
以下是一些当 TIMx_ARR=0x36 时，计数器在不同时钟频率下的操作例子。
图：计数器时序图，内部时钟分频因子为 1



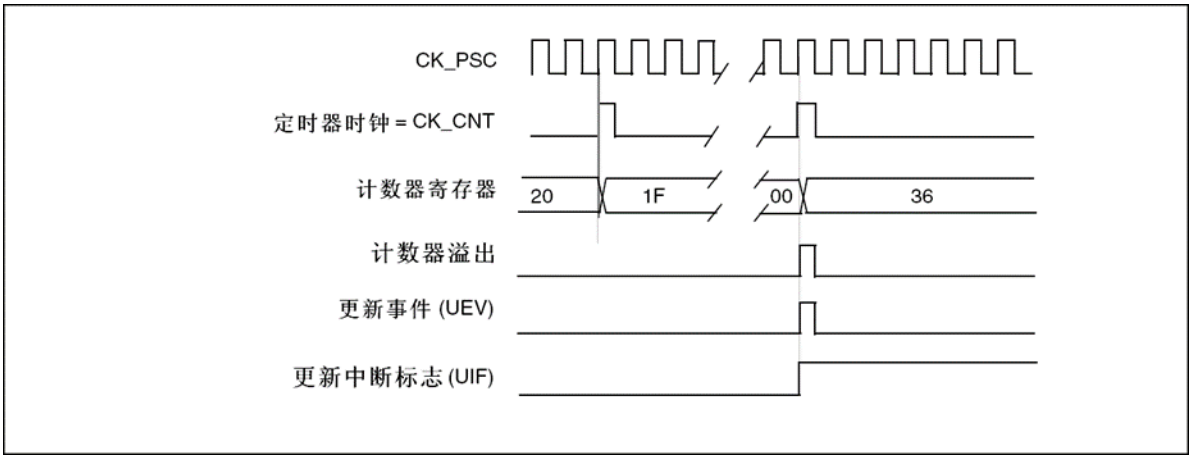
图：计数器时序图，内部时钟分频因子为 2



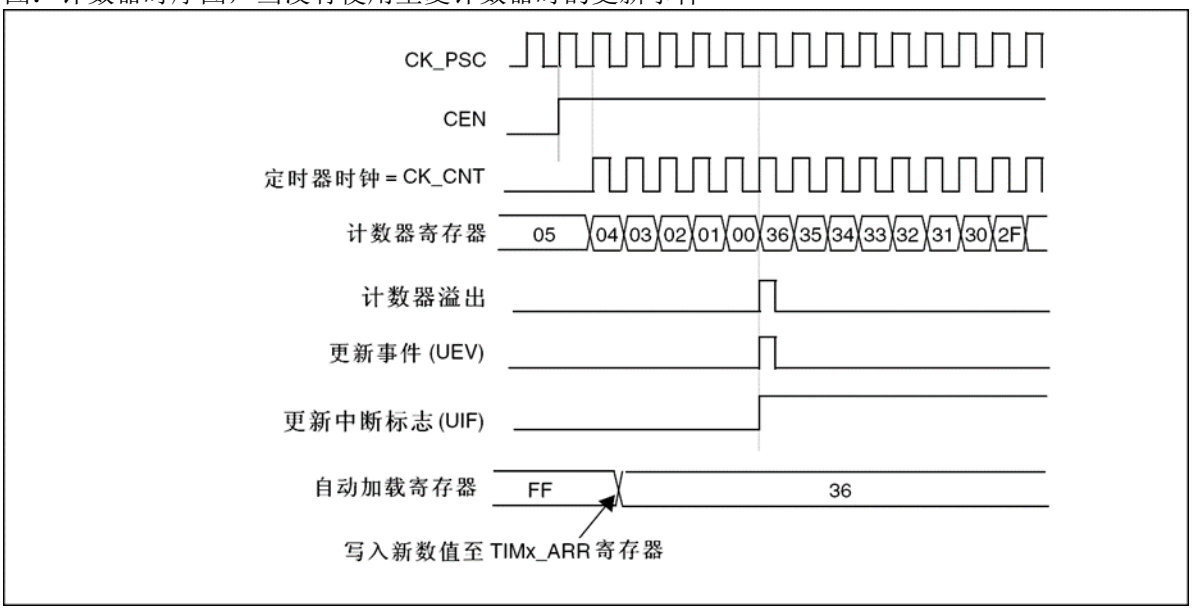
图：计数器时序图，内部时钟分频因子为 4



图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，当没有使用重复计数器时的更新事件



中央对齐模式(向上/向下计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(TIMx_ARR 寄存器)−1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

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

设置 TIMx_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMx_CR1 寄存器中的 URS 位(选择更新请求) ，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

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

●重复计数器被重置为 TIMx_RCR 寄存器中的内容

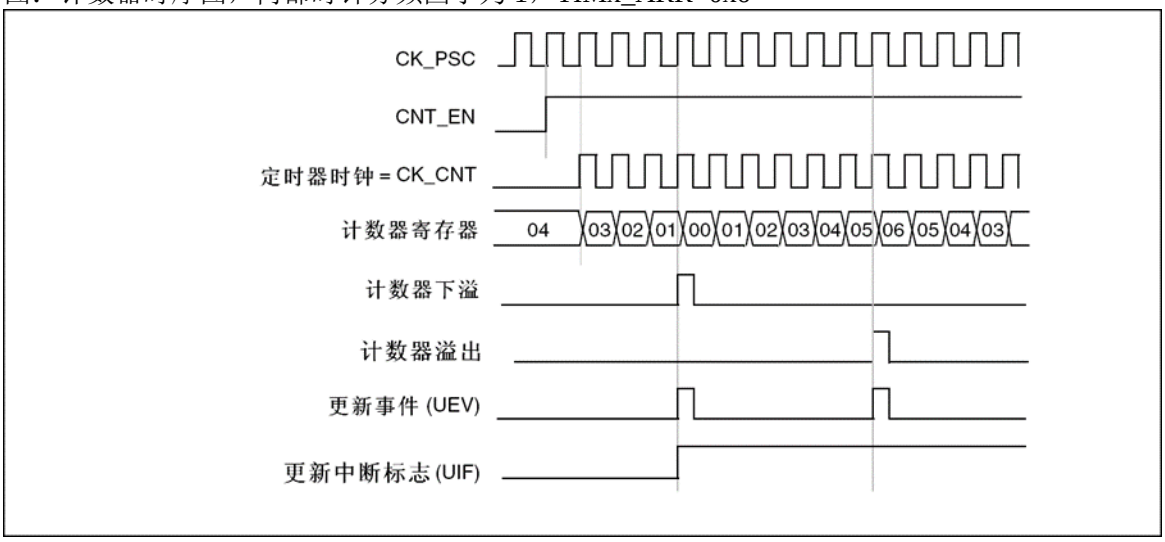
●预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值。

●当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

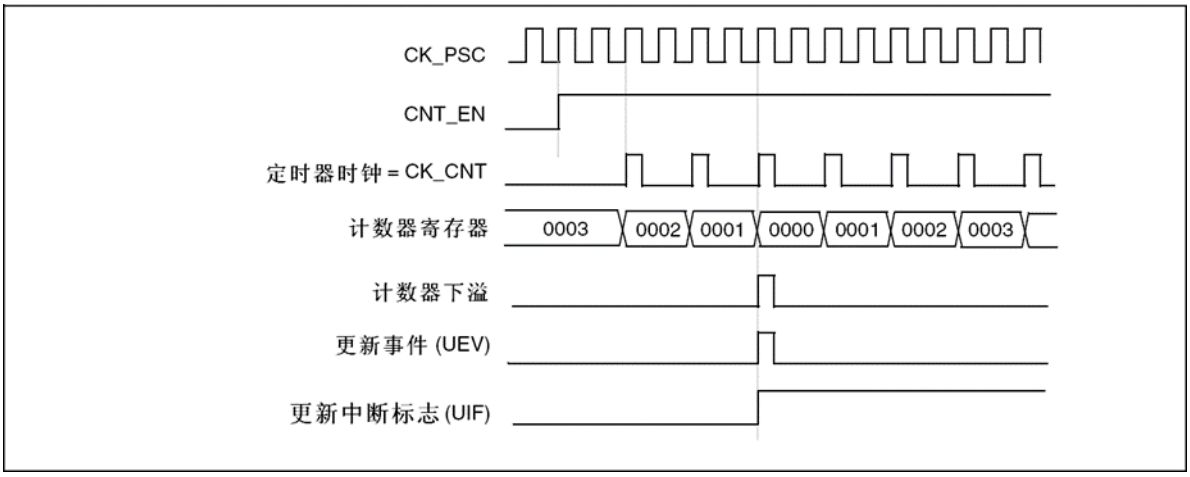
注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值(计数器被装载为新的值)。

以下是一些计数器在不同时钟频率下的操作的例子：

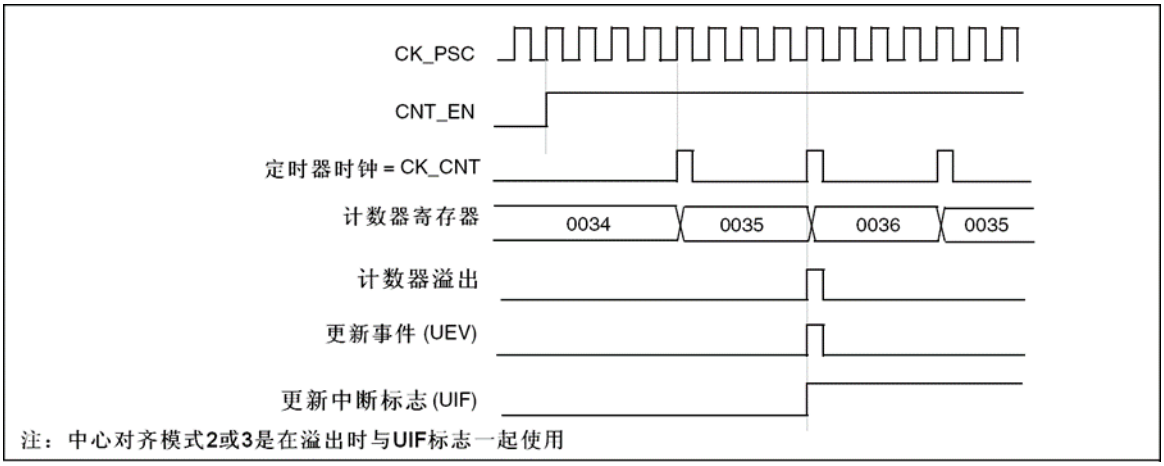
图：计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6



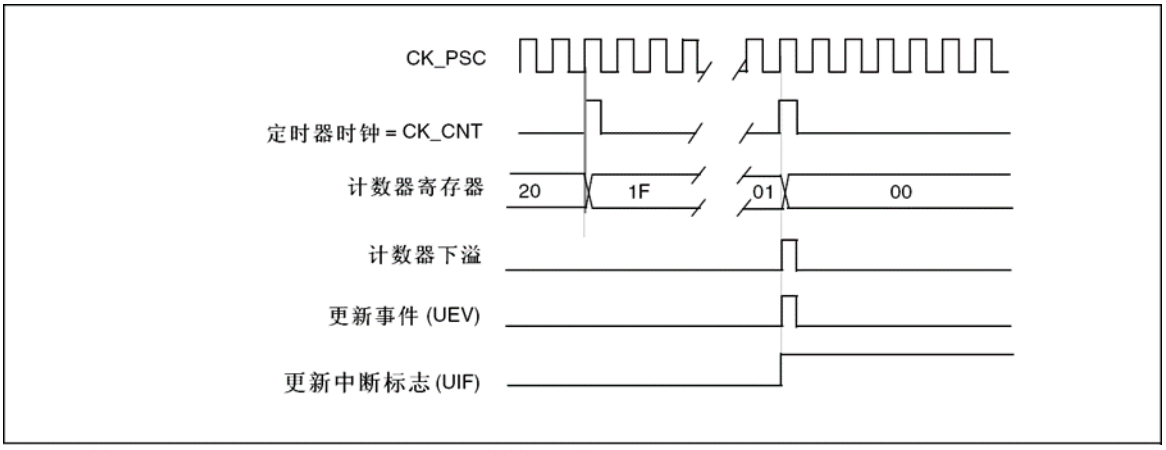
图：计数器时序图，内部时钟分频因子为 2



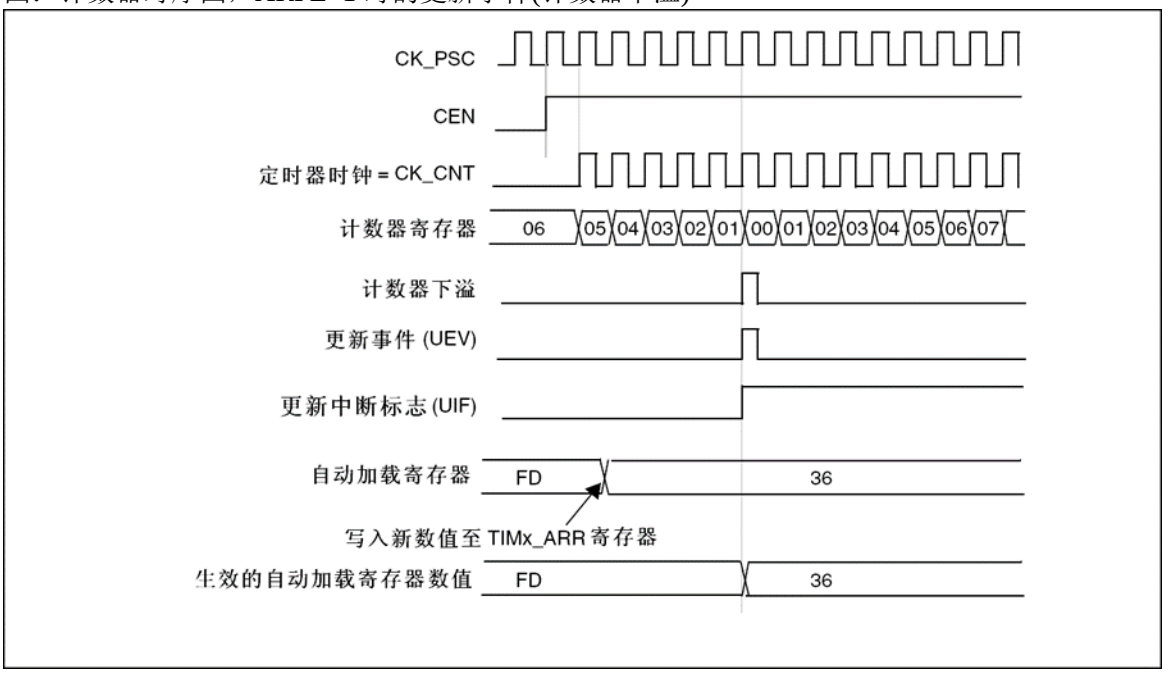
图：计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36



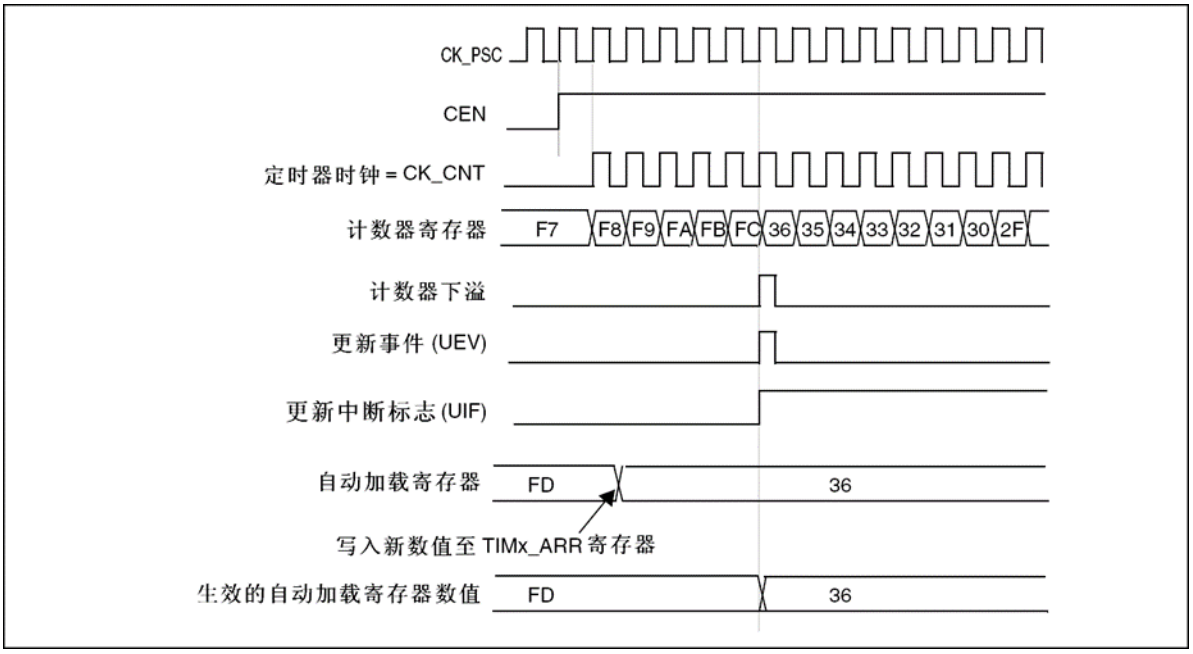
图：计数器时序图，内部时钟分频因子为 N



图：计数器时序图，ARPE=1 时的更新事件(计数器下溢)



图：计数器时序图，ARPE=1 时的更新事件(计数器溢出)



11.4.3 重复计数器

13.3.1 节“时基单元”解释了计数器上溢/下溢时更新事件(UEV)是如何产生的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器(TIMx_ARR 自动重载入寄存器， TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/ 比较寄存器 TIMx_CCRx)，N 是 TIMx_RCR 重复计数寄存器中的值。重复计数器在下述任一条件成立时递减：

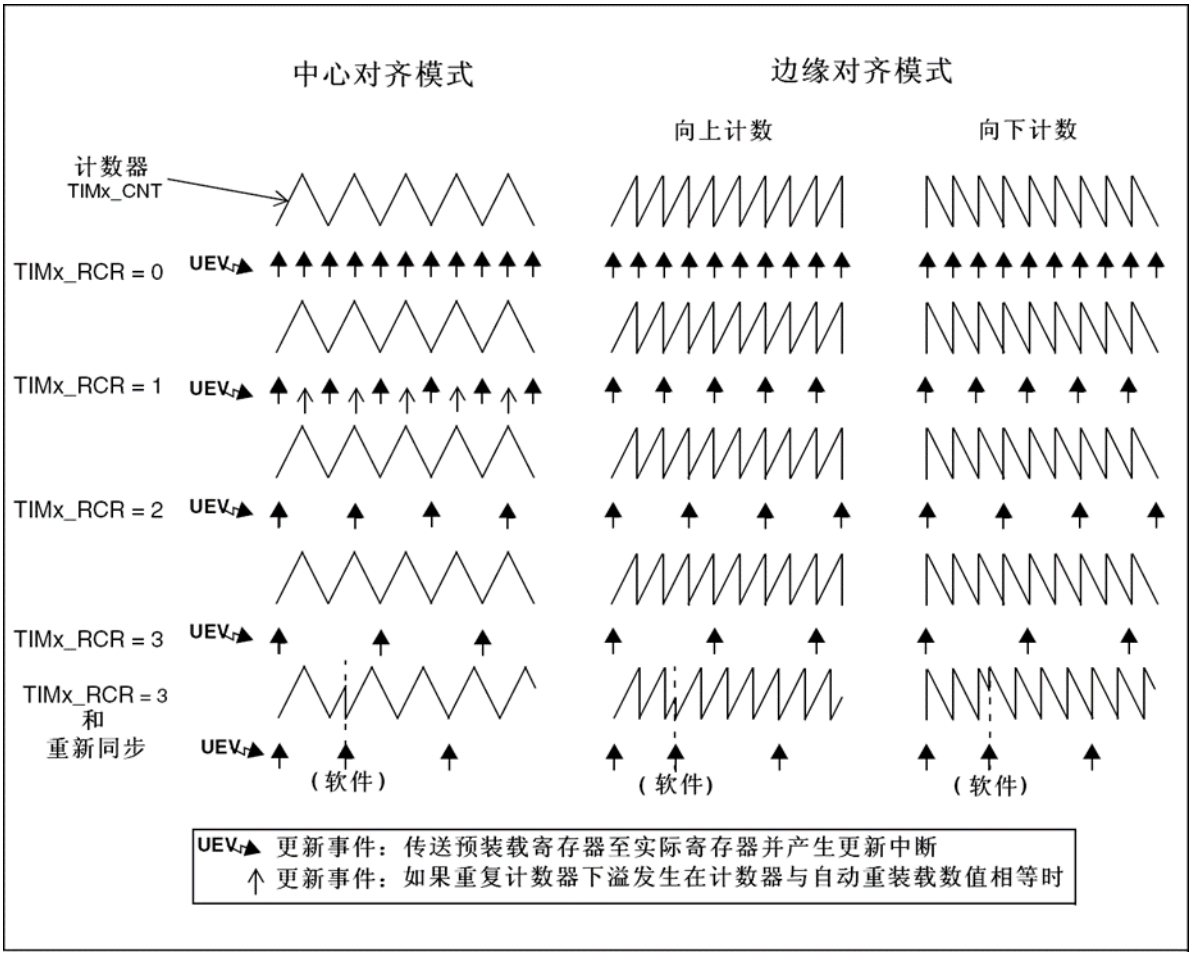
●向上计数模式下每次计数器溢出时，

●向下计数模式下每次计数器下溢时，

●中央对齐模式下每次上溢和每次下溢时。虽然这样限制了 PWM 的最大循环周期为 128，但它能够在每个 PWM 周期 2 次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 2xTck。

重复计数器是自动加载的，重复速率是由 TIMx_RCR 寄存器的值定义(参看图 70)。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

图：不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置



11.4.4 时钟选择

计数器时钟可由下列时钟源提供：

●内部时钟(CK_INT)

●外部时钟模式 1：外部输入引脚

●外部时钟模式 2：外部触发输入 ETR

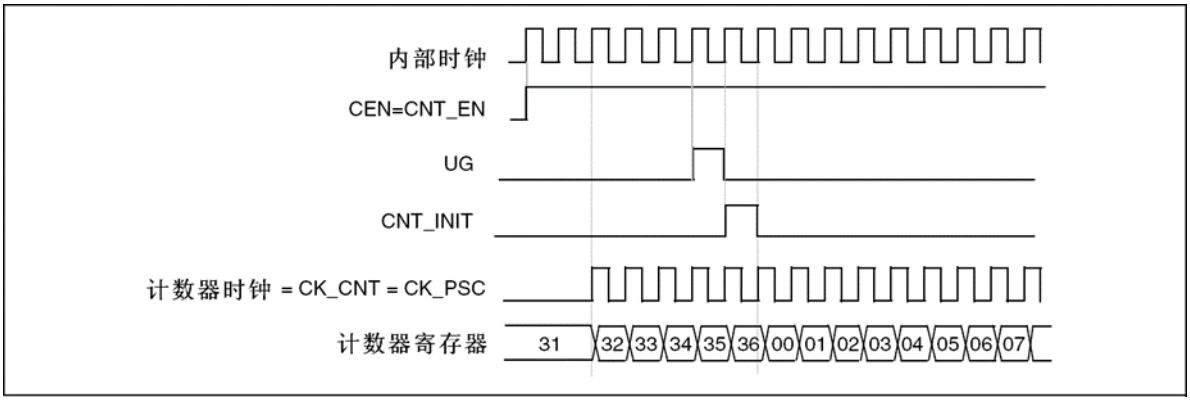
●内部触发输入(ITRx)：使用一个定时器作为另一个定时器的预分频器。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。详见下一章。

内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=000)，则 CEN、DIR(TIMx_CR1 寄存器)和 UG 位(TIMx_EGR 寄存器)是事实上的控制位，并且只能被软件修改(UG 位仍被自动清除)。只要 CEN 位被写成’1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

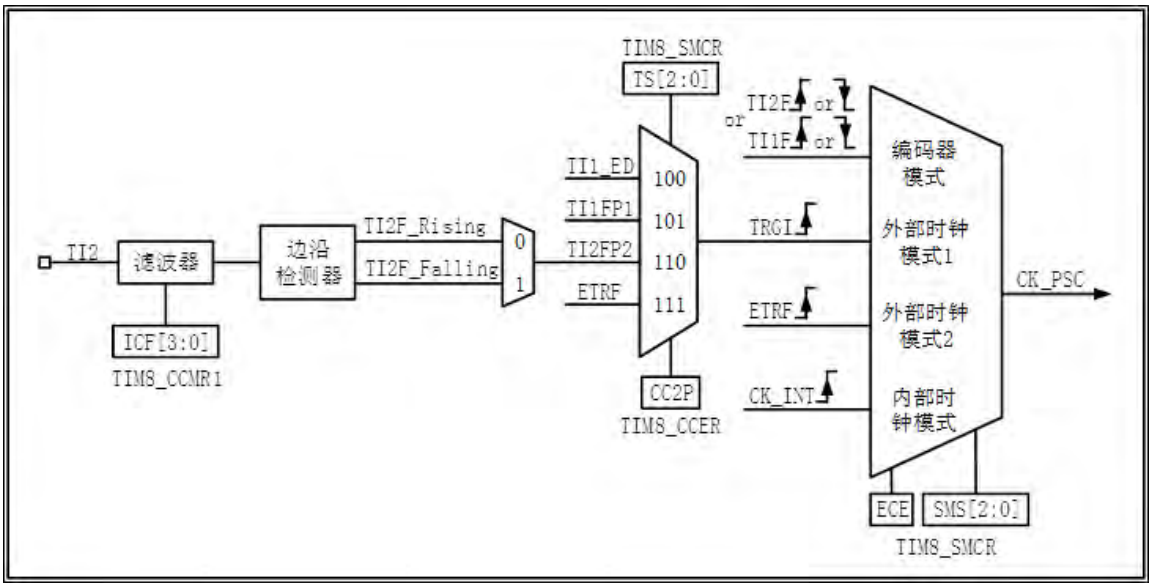
图：一般模式下的控制电路，内部时钟分频因子为 1



外部时钟源模式 1

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

图：TI2 外部时钟连接例子



例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤： 配置 TIM8_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿

配置 TIM8_CCMR1 寄存器的 IC2F[3:0] ，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)

配置 TIM8_CCER 寄存器的 CC2P=0，选定上升沿极性

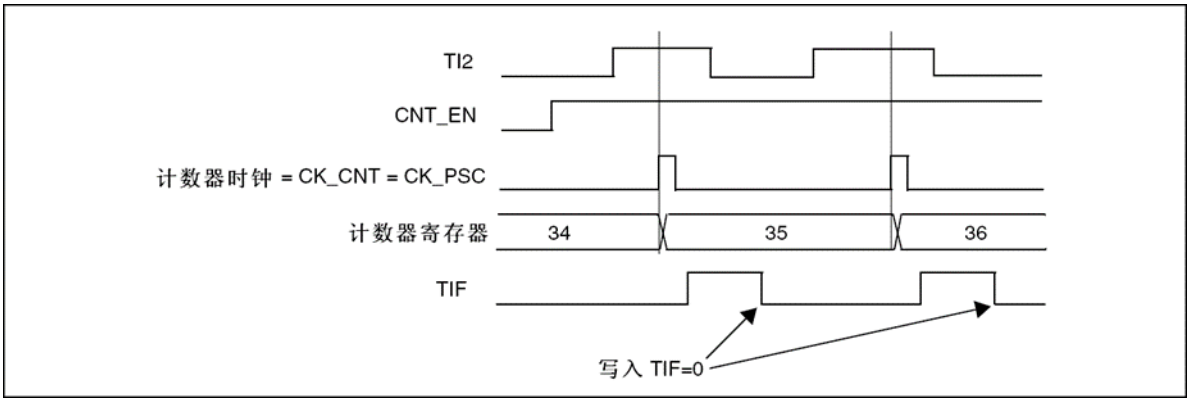
配置 TIM8_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1 配置 TIM8_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源 设置 TIM8_CR1 寄存器的 CEN=1，启动计数器

注：捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。

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

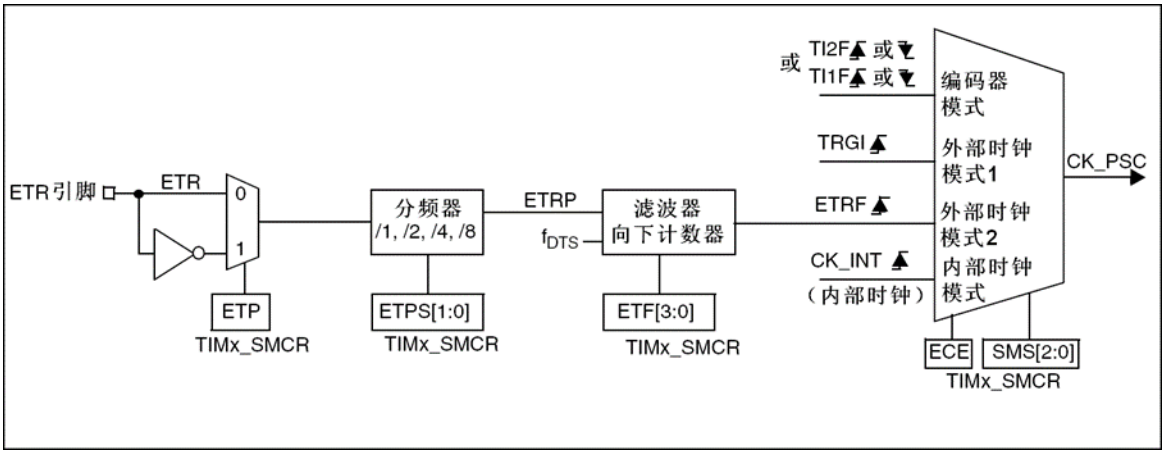
图：外部时钟模式 1 下的控制电路



外部时钟源模式 2

选定此模式的方法为：令 TIMx_SMCR 寄存器中的 ECE=1 计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。下图是外部触发输入的框图

图：外部触发输入框图



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

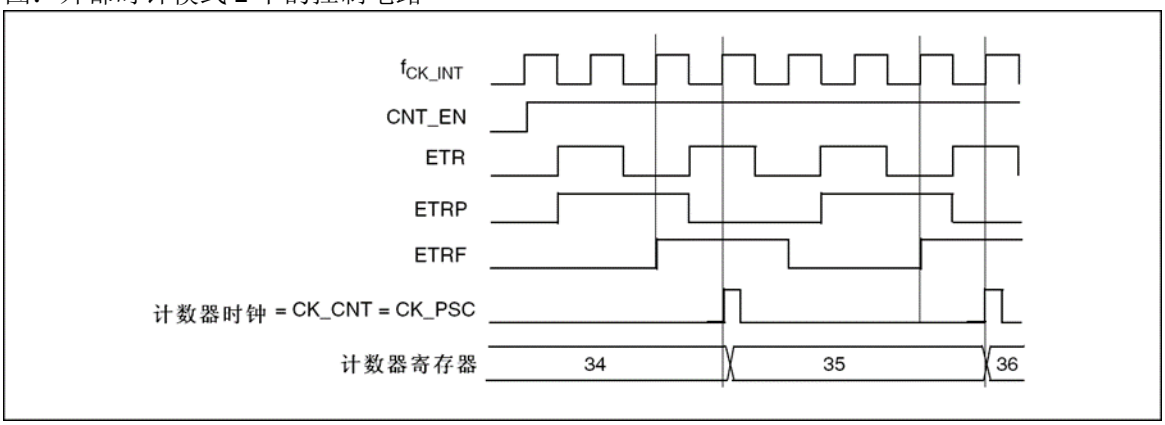
4. 本例中不需要滤波器，置 TIMx_SMCR 寄存器中的 ETF[3:0]=0000 设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS[1:0]=01

选择 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0 开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1

启动计数器，写 TIMx_CR1 寄存器中的 CEN=1 计数器在每 2 个 ETR 上升沿计数一次。

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

图：外部时钟模式 2 下的控制电路

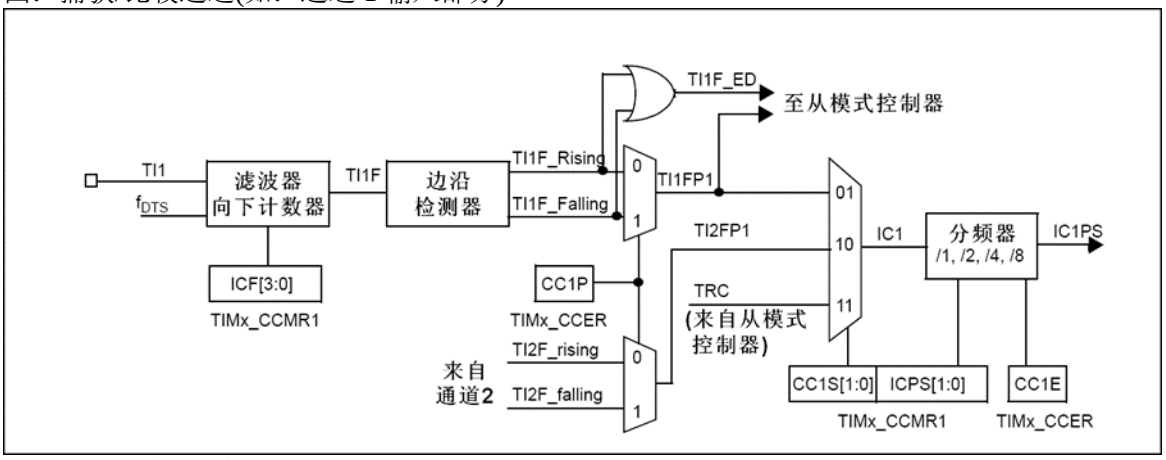


11.4.5 捕获/比较通道

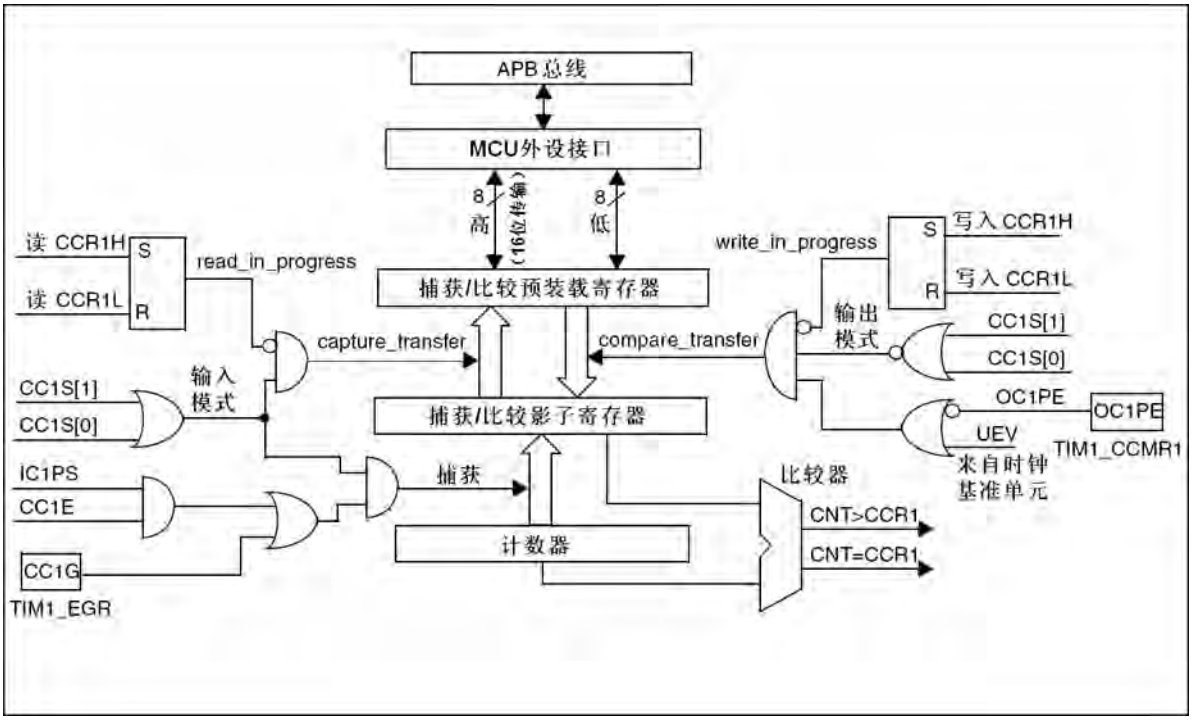
每一个捕获/比较通道都是围绕着一个捕获/比较寄存器(包含影子寄存器)，包括捕获的输入部分(数字滤波、多路复用和预分频器)，和输出部分(比较器和输出控制)。下图是一个捕获/比较通道概览。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号(TIxFPx)，它可以作为从模式控制器的输入触发或者作为捕获控制。该信号通过预分频进入捕获寄存器(ICxPS)。

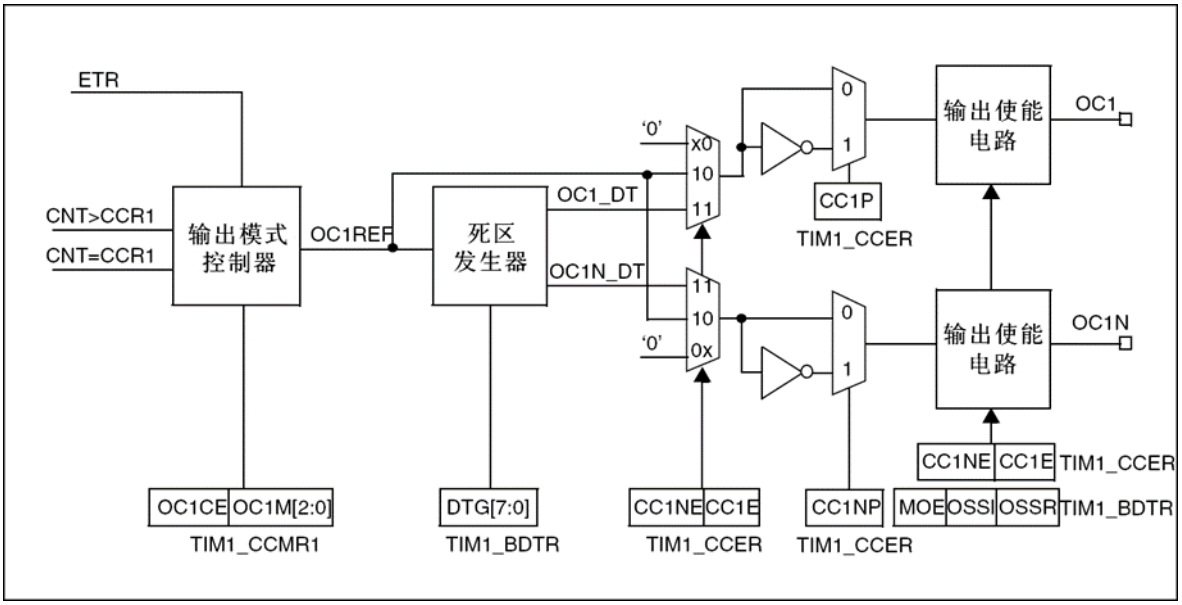
图：捕获/比较通道(如：通道 1 输入部分)



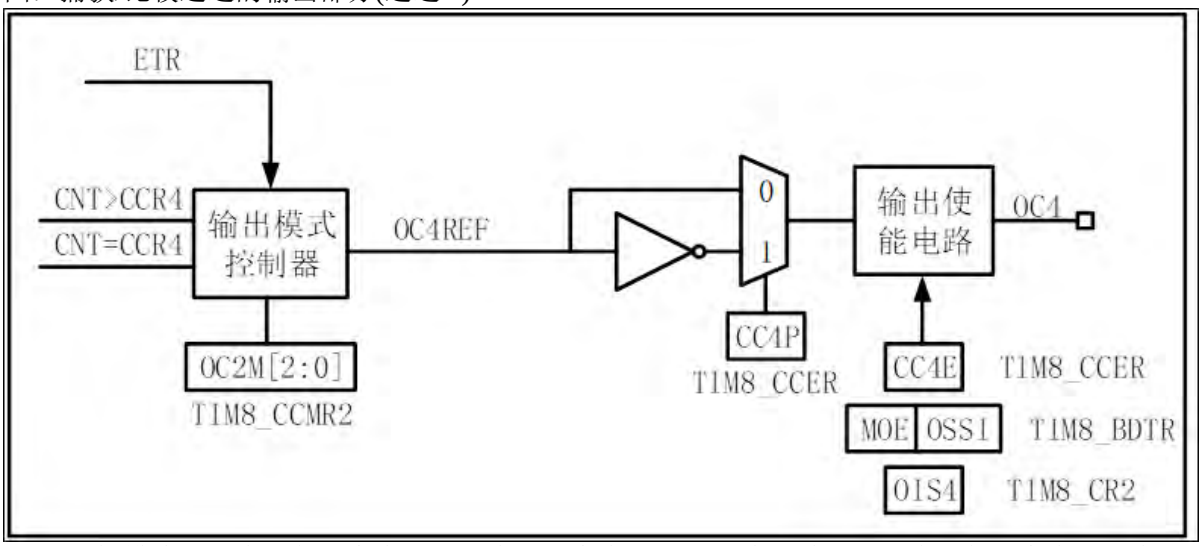
图：捕获/比较通道 1 的主电路



图：捕获/比较通道的输出部分(通道 1 至 3)



图：捕获/比较通道的输出部分(通道 4)



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。
在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

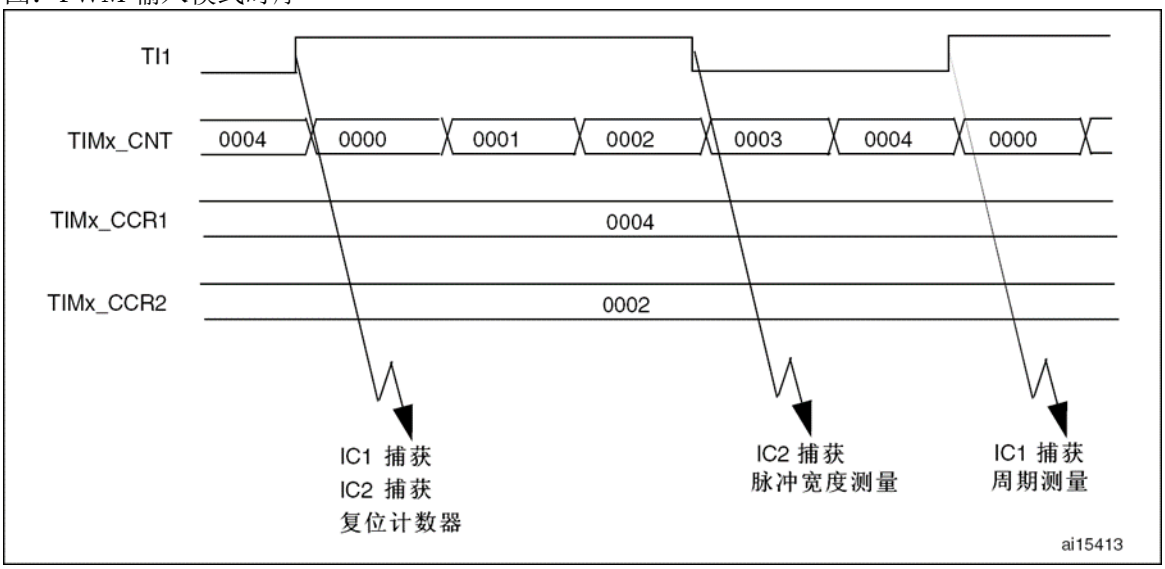
11.4.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(TIMx_CCRx)中。当发生捕获事件时，相应的 CCxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 DMA 请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF(TIMx_SR 寄存器)被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。
以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMx_CCR1 寄存器中，步骤如下：
●选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为’00’，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
●根据输入信号的特点，配置输入滤波器为所需的带宽(即输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位)。假设输入信号在最多 5 个内部时钟周期的时间内抖动， 我们须配置

滤波器的带宽长于 5 个时钟周期；因此我们可以(以 fDTS 频率)连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIMx_CCMR1 寄存器中写入 IC1F=0011。
●选择 TI1 通道的有效转换边沿，在 TIMx_CCER 寄存器中写入 CC1P=0(上升沿)。
●配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止(写 TIMx_CCMR1 寄存器的 IC1PS=00)。
●设置 TIMx_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
●如果需要，通过设置 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。当发生一个输入捕获时：
●产生有效的电平转换时，计数器的值被传送到 TIMx_CCR1 寄存器。
●CC1IF 标志被设置(中断标志)。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 被置 1。
●如设置了 CC1IE 位，则会产生一个中断。
●如设置了 CC1DE 位，则还会产生一个 DMA 请求。
为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。
注：设置 TIMx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断和/或 DMA 请求。

11.4.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：
●两个 ICx 信号被映射至同一个 TIx 输入。
●这 2 个 ICx 信号为边沿有效，但是极性相反。
●其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。
例如，你需要测量输入到 TI1 上的 PWM 信号的长度(TIMx_CCR1 寄存器)和占空比(TIMx_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)
●选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01(选中 TI1)。
●选择 TI1FP1 的有效极性(用来捕获数据到 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
●选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
●选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
●选择有效的触发输入信号：置 TIMx_SMCR 寄存器中的 TS=101(选择 TI1FP1)。
●配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
●使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。
图：PWM 输入模式时序



因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIMx_CH1

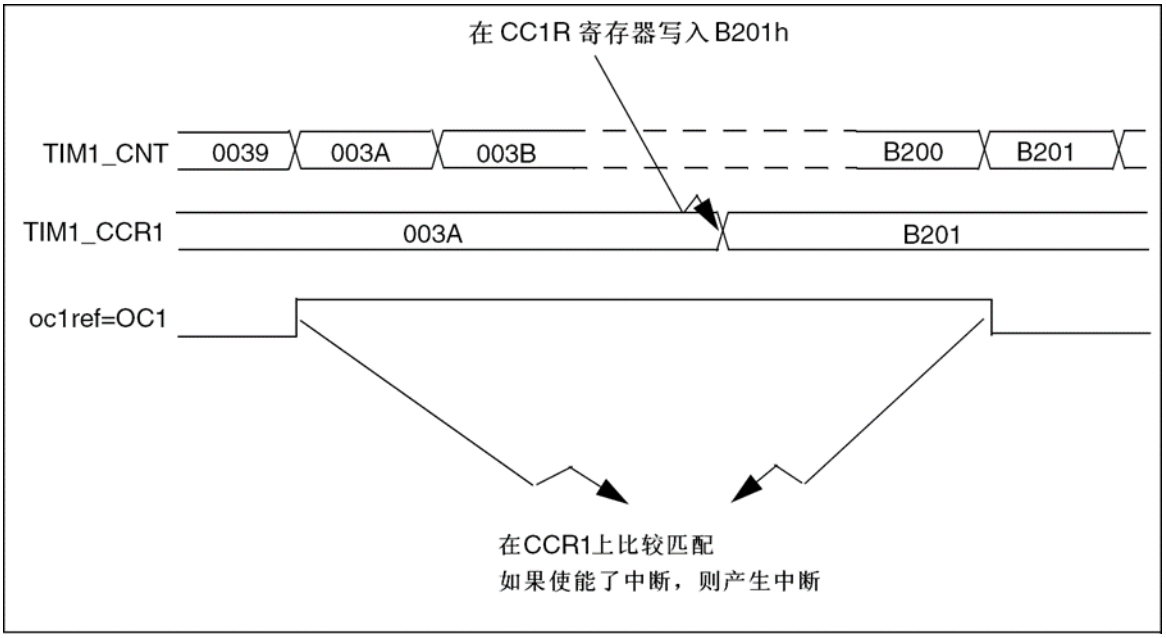
/TIMx_CH2 信号。

11.4.8 强置输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN) 能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。
置 TIMx_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。
例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。
置 TIMx_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。
该模式下，在 TIMx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

11.4.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：
●将输出比较模式(TIMx_CCMRx 寄存器中的 OCxM 位)和输出极性(TIMx_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
●设置中断状态寄存器中的标志位(TIMx_SR 寄存器中的 CCxIF 位)。
●若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CCxIE 位)，则产生一个中断。
●若设置了相应的使能位(TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。
TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否需要使用预装载寄存器。在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。
同步的精度可以达到计数器的一个计数周期。输出比较模式(在单脉冲模式下)也能用来输出一个单脉冲。
输出比较模式的配置步骤：
1.选择计数器时钟(内部，外部，预分频器)。
2.将相应的数据写入 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3.如果要产生一个中断请求，设置 CCxIE 位。
4.选择输出模式，例如：
—要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
—置 OCxPE = 0 禁用预装载寄存器
—置 CCxP = 0 选择极性为高电平有效
—置 CCxE = 1 使能输出
5.设置 TIMx_CR1 寄存器的 CEN 位启动计数器
TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OCxPE=’0’，否则 TIMx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新)。下图给出了一个例子。
图：输出比较模式，翻转 OC1



11.4.10PWM 模式

脉冲宽度调制模式可以产生一个由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比的信号。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入’110’(PWM 模式 1)或’111’(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_CR1 寄存器的 ARPE 位，(在向上计数或中心对称模式中) 使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过(TIMx_CCER 和 TIMx_BDTR 寄存器中)CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，(依据计数器的计数方向)以确定是否符合 TIMx_CCRx≤TIMx_CNT 或者 TIMx_CNT≤TIMx_CCRx。

根据 TIMx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

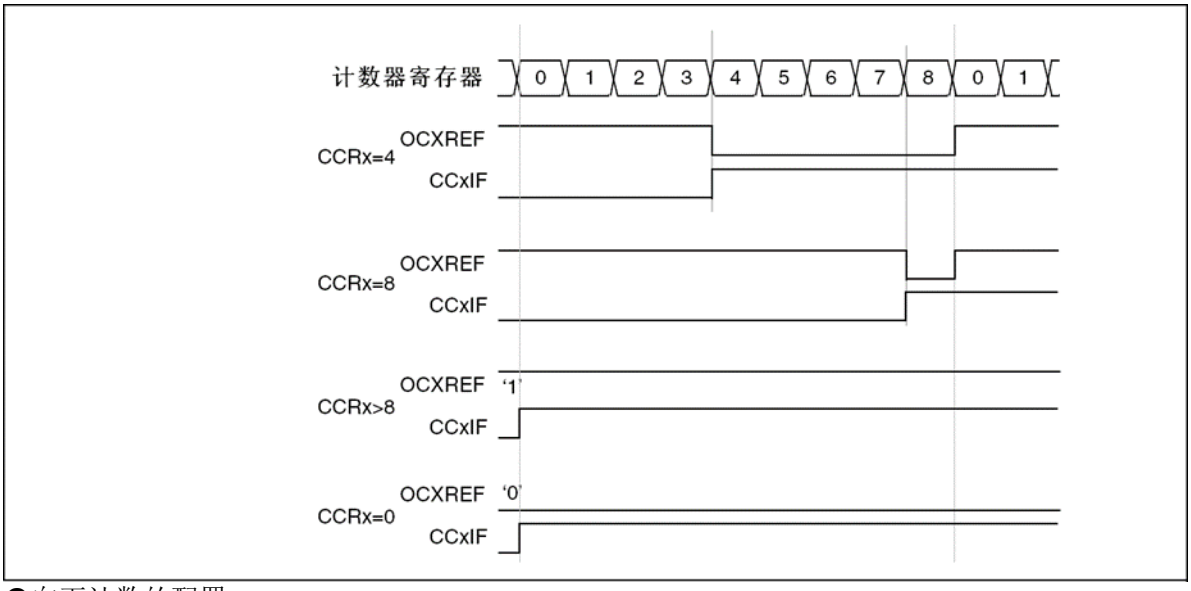
PWM 边沿对齐模式

●向上计数配置

当 TIMx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。

下面是一个 PWM 模式 1 的例子。当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为’1’。如果比较值为 0，则 OCxREF 保持为’0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

图：边沿对齐的 PWM 波形(ARR=8)



●向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_ARR 中的自动重装载值，则 OCxREF 保持为’1’。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为’00’时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIMx_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它。参看 13.3.2 节的中央对齐模式。

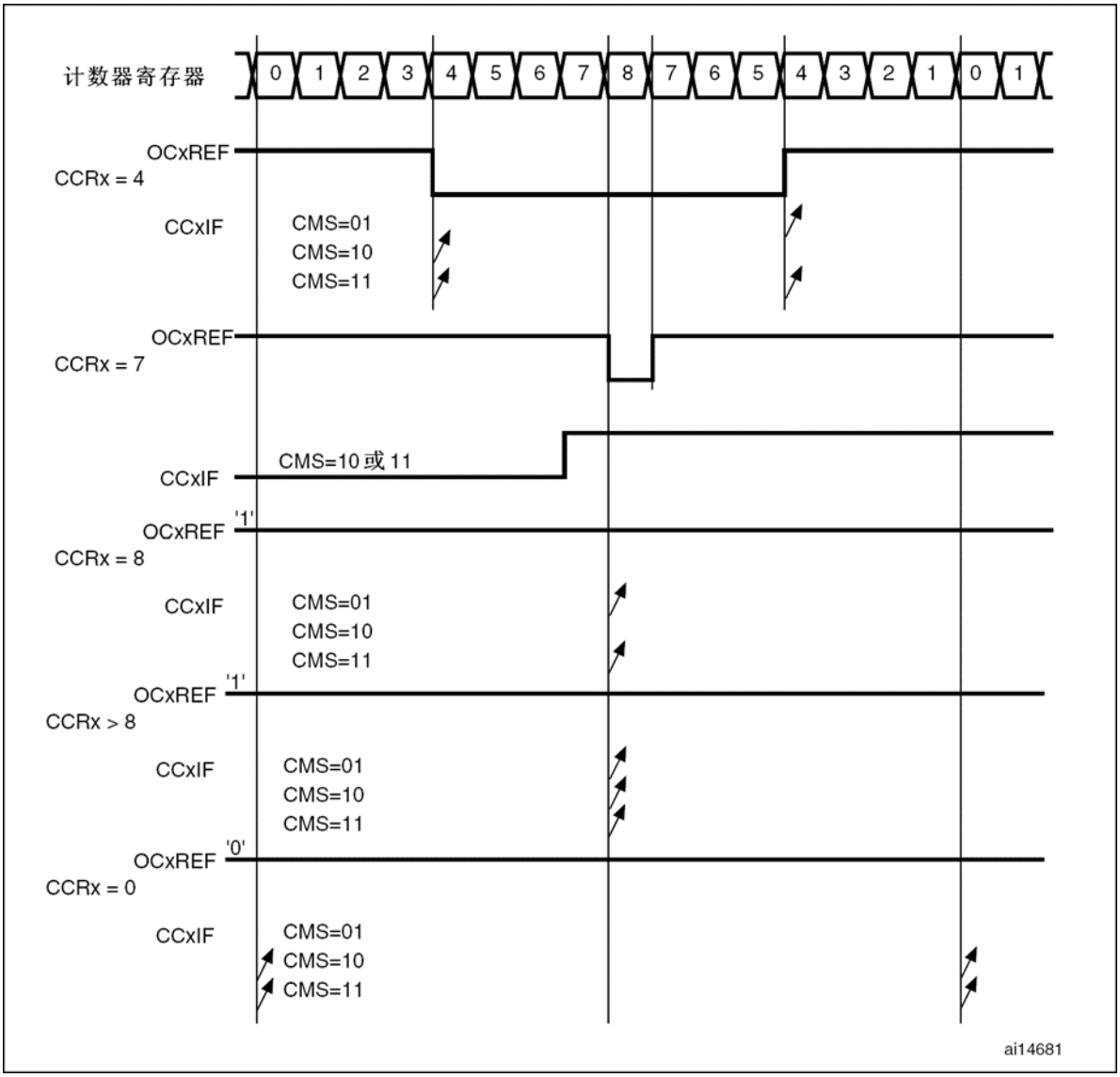
下图给出了一些中央对齐的 PWM 波形的例子

●TIMx_ARR=8

●PWM 模式 1

●TIMx_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

图：中央对齐的 PWM 波形(APR=8)



使用中央对齐模式的提示：

●进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。

●不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：

—如果写入计数器的值大于自动重加载的值(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。

—如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

●使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

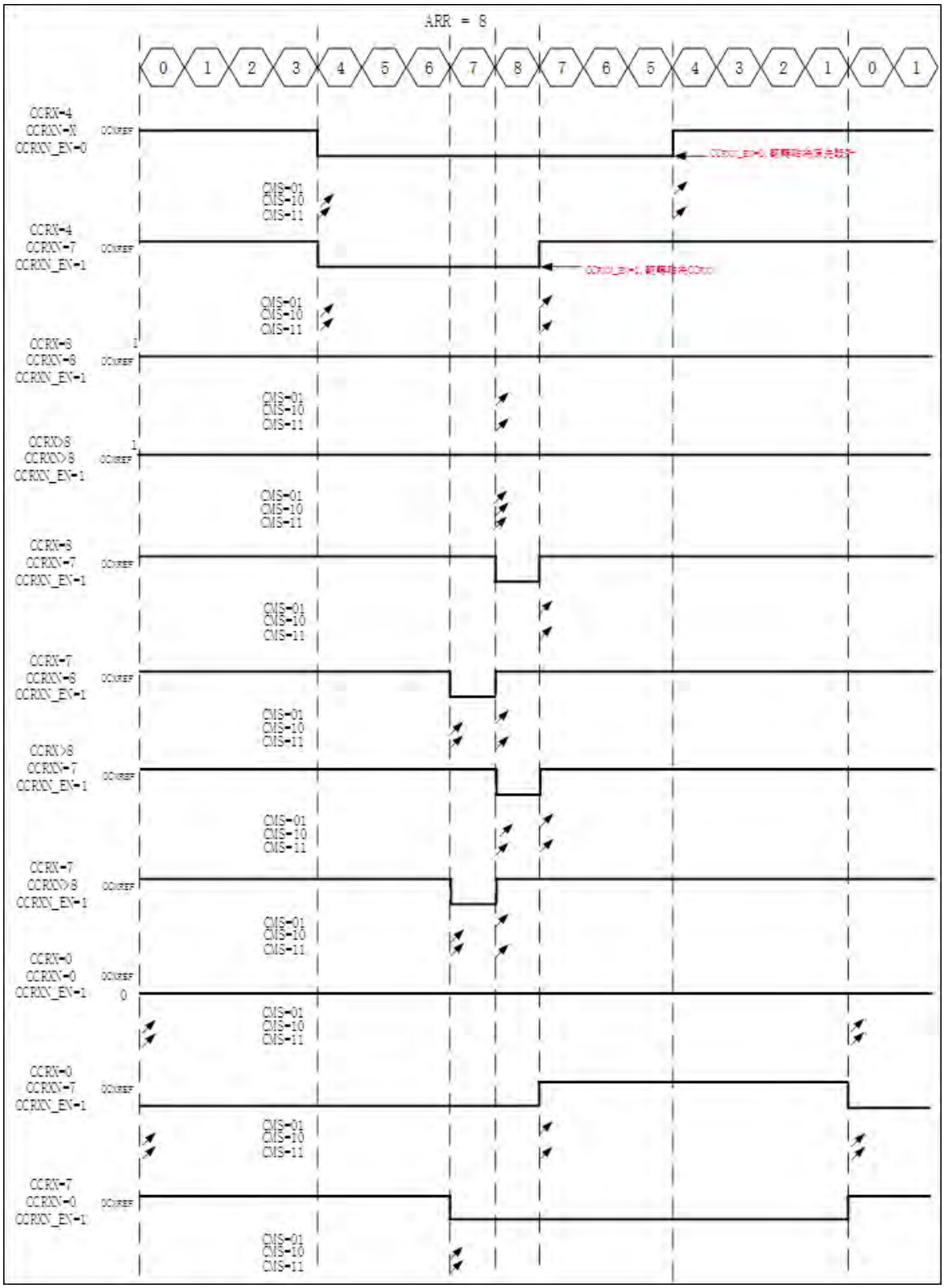
11.4.11非对称模式

中央计数模式的非对称模式

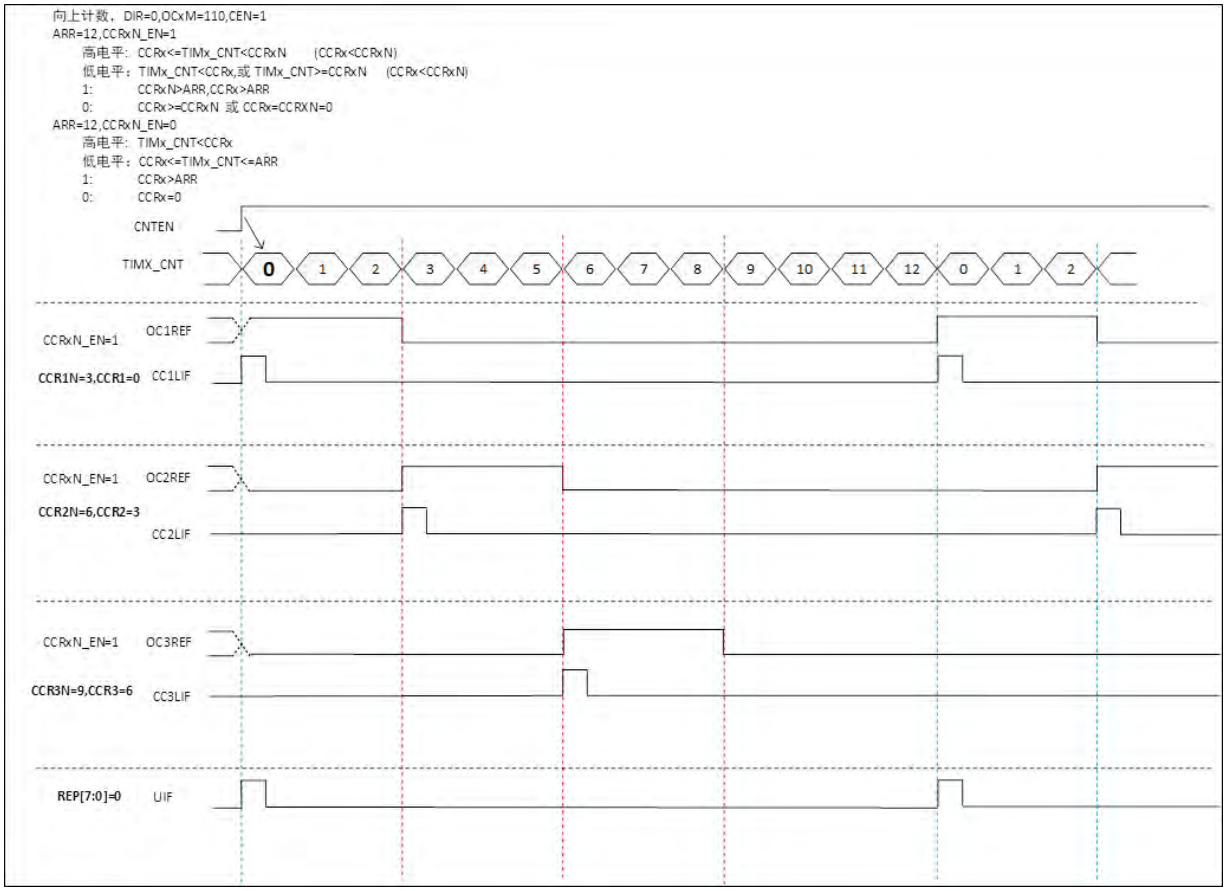
在 PWM 模式下，计数器选择中央对齐计数模式，由于中央对齐模式在一个周期内分别向上向下计数一次，所以周期就变成了普通模式的两倍，因此要想使输出波的频率不变，那必须改变自动重装载寄存器 ARR 的值，使其为普通模式的一半，则可以实现所需要的频率输出。

非对称 PWM 模式下寄存器配置如下：

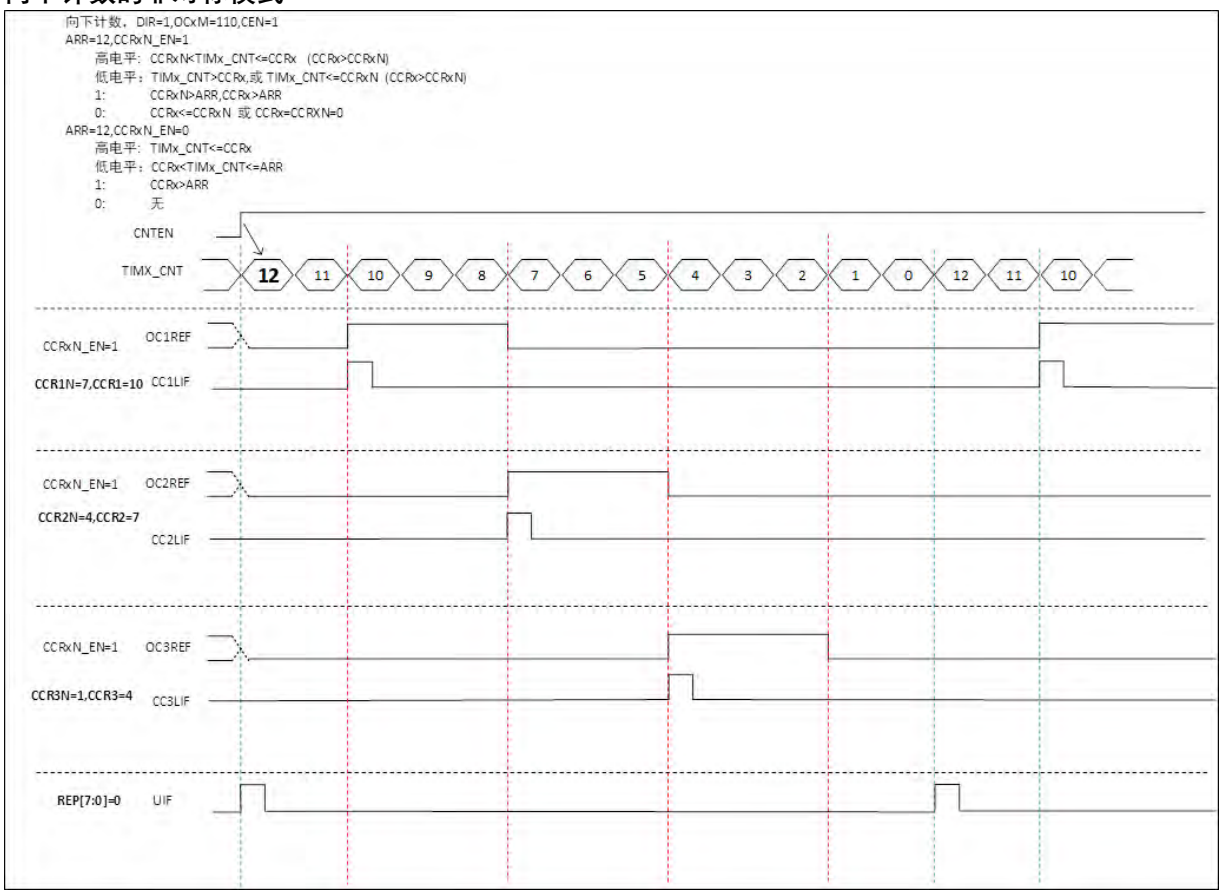
OCxREF 是由相应的 TIMx_CCRx 和 TIMx_CCRxN 控制
可以看出，只要在程序中动态修改 CCRx 和 CCRxN 的值，即可实现动态修改移相相位



向上计数的非对称模式

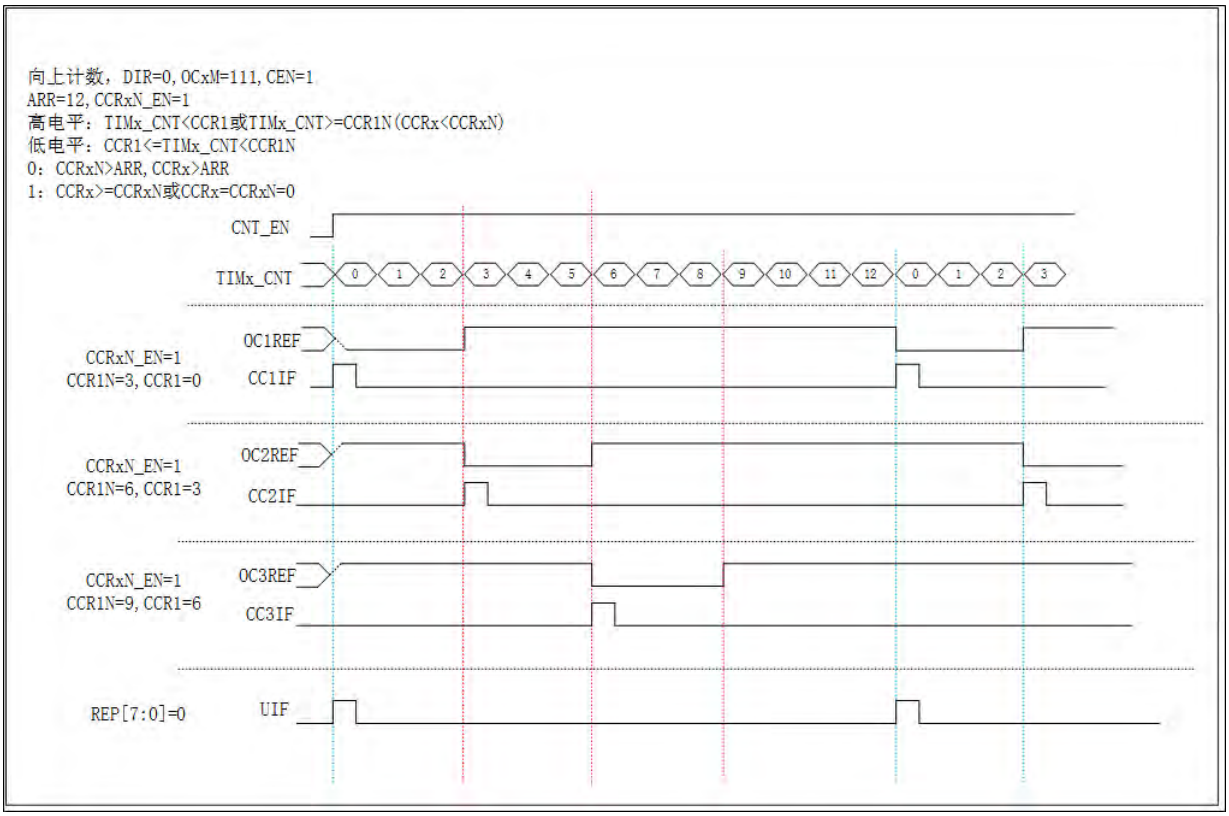

向上计数，DIR=0,OCxM=110,CEN=1
ARR=12,CCRxN_EN=1
高电平：CCRx<=TIMx_CNT<CCRxN (CCRx<CCRxN)
低电平：TIMx_CNT<CCRx,或 TIMx_CNT>=CCRxN (CCRx<CCRxN)
1: CCRxN>ARR,CCRx>ARR
0: CCRx>=CCRxN 或 CCRx=CCRXN=0
ARR=12,CCRxN_EN=0
高电平：TIMx_CNT<CCRx
低电平：CCRx<=TIMx_CNT<=ARR
1: CCRx>ARR
0: CCRx=0
CNTEN
TIMX_CNT
0 1 2 3 4 5 6 7 8 9 10 11 12 0 1 2
CCRxN_EN=1
OC1REF
CCR1N=3,CCR1=0
CC1LIF
CCRxN_EN=1
OC2REF
CCR2N=6,CCR2=3
CC2LIF
CCRxN_EN=1
OC3REF
CCR3N=9,CCR3=6
CC3LIF
REP[7:0]=0
UIF

向下计数的非对称模式

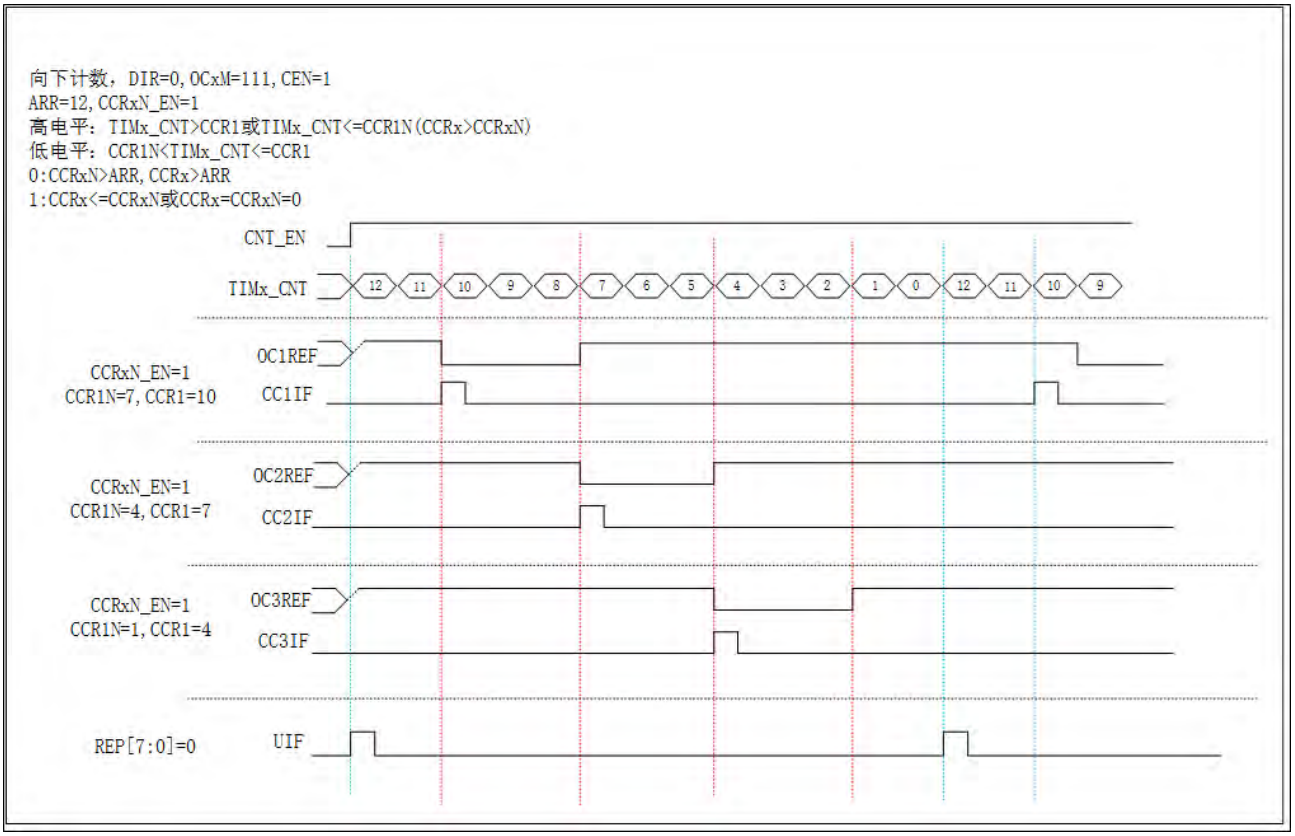

向下计数，DIR=1,OCxM=110,CEN=1
ARR=12,CCRxN_EN=1
高电平：CCRxN<TIMx_CNT<=CCRx (CCRx>CCRxN)
低电平：TIMx_CNT>CCRx,或 TIMx_CNT<=CCRxN (CCRx>CCRxN)
1: CCRxN>ARR,CCRx>ARR
0: CCRx<=CCRxN 或 CCRx=CCRXN=0
ARR=12,CCRxN_EN=0
高电平：TIMx_CNT<=CCRx
低电平：CCRx<TIMx_CNT<=ARR
1: CCRx>ARR
0: 无
CNTEN
TIMX_CNT
12 11 10 9 8 7 6 5 4 3 2 1 0 12 11 10
CCRxN_EN=1
OC1REF
CCR1N=7,CCR1=10
CC1LIF
CCRxN_EN=1
OC2REF
CCR2N=4,CCR2=7
CC2LIF
CCRxN_EN=1
OC3REF
CCR3N=1,CCR3=4
CC3LIF
REP[7:0]=0
UIF

	DIR=0,向上计数,OC1M=110	DIR=1,向下计数,OC1M=110
CCR1N_EN=1	CCR1<=TIMx_CNT<CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx<CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx<CCRxN);	CCR1N<TIMx_CNT<=CCR1,通道 1 为有效电平(OC1REF=1),(CCRx>CCRxN); TIMx_CNT>CCR1,TIMx_CNT<=CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx>CCRxN);
CCR1N_EN=0	TIMx_CNT<CCR1,通道 1 为有效电平(OC1REF=1); TIMx_CNT>=CCR1,通道 1 为无效电平(OC1REF=0);	TIMx_CNT<=CCR1,通道 1 为有效电平(OC1REF=1); TIMx_CNT>CCR1,通道 1 为无效电平(OC1REF=0);

向上计数



向下计数



	DIR=0,向上计数,OC1M=111	DIR=1,向下计数,OC1M=111
CCR1N_EN=1	CCR1<=TIMx_CNT<CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx<CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx<CCRxN);	CCR1N<TIMx_CNT<=CCR1,通道 1 为无效电平(OC1REF=0),(CCRx>CCRxN); TIMx_CNT>CCR1,TIMx_CNT<=CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx>CCRxN);
CCR1N_EN=0	TIMx_CNT<CCR1,通道 1 为无效电平(OC1REF=0); TIMx_CNT>=CCR1,通道 1 为有效电平(OC1REF=1);	TIMx_CNT<=CCR1,通道 1 为无效电平(OC1REF=0); TIMx_CNT>CCR1,通道 1 为有效电平(OC1REF=1);

11.4.12互补输出和死区插入

高级控制定时器(TIM1 和 TIM8)能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。
这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。
配置 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性(主输出 OCx 或互补输出 OCxN)。
互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见表 75 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。特别的是，在转换到 IDLE 状态时(MOE 下降到 0)死区被激活。

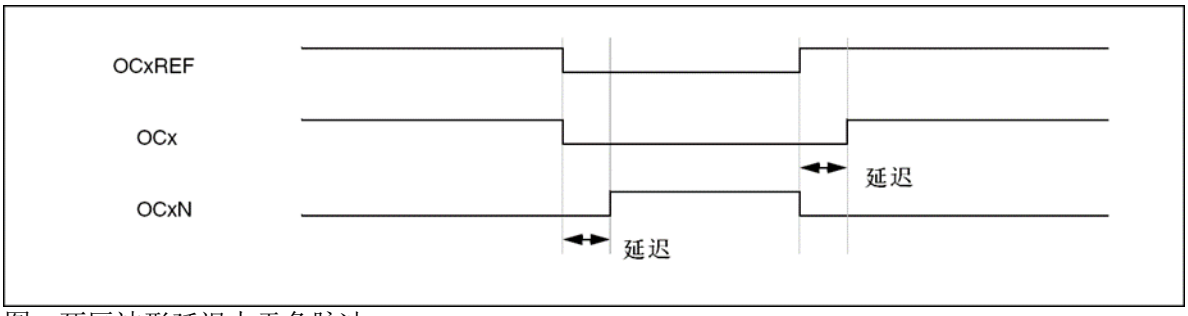
同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高有效：

●OCx 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。

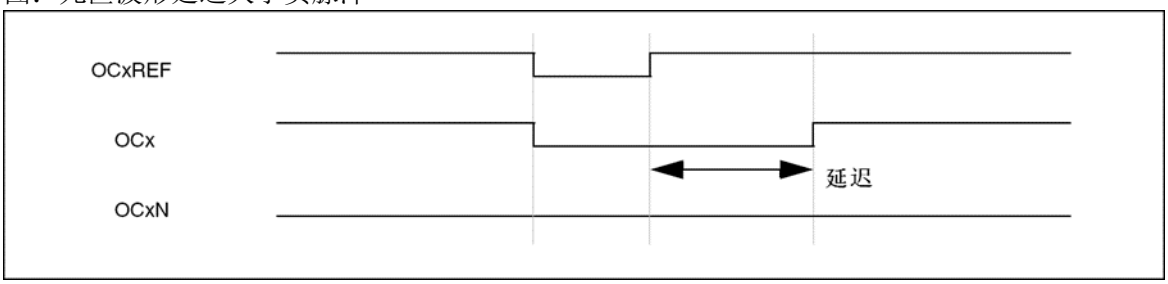
●OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。如果延迟大于当前有效的输出宽度(OCx 或者 OCxN)，则不会产生相应的脉冲。

下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。(假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1)

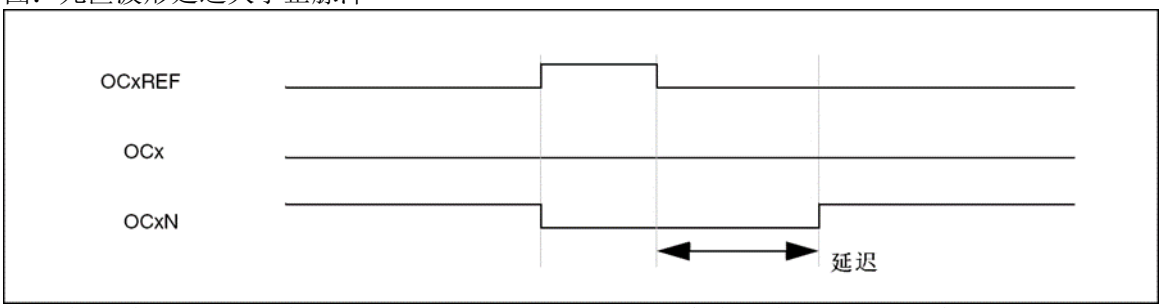
图：带死区插入的互补输出



图：死区波形延迟大于负脉冲



图：死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。

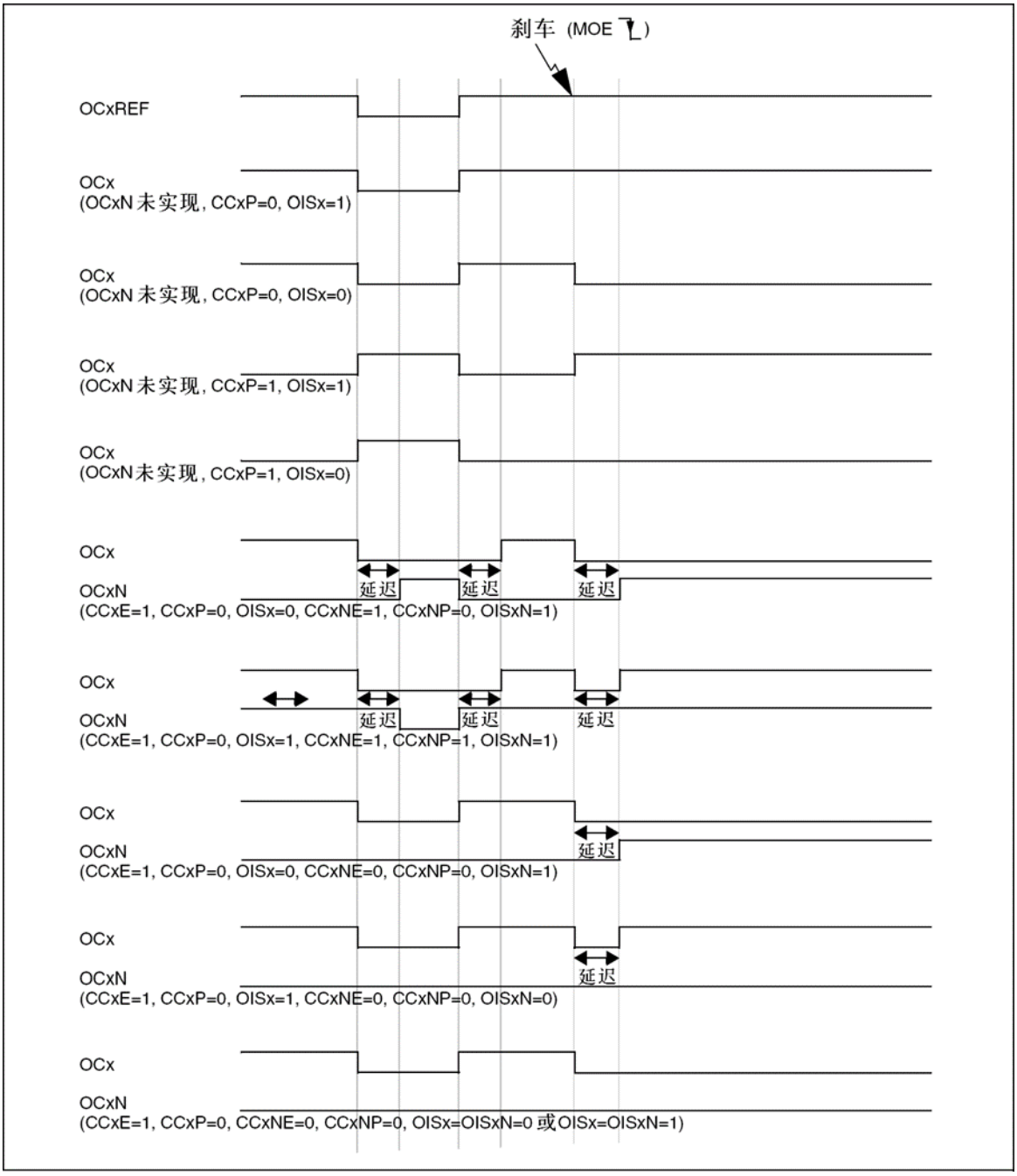
这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

注：当只使能 OCxN(CCxE=0, CCxNE=1)时，它不会反相，当 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。另一方面，当 OCx 和 OCxN 都被使能时(CCxE=CCxNE=1)， 当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

11.4.13使用刹车功能

当使用刹车功能时，依据相应的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位， TIMx_CR2 寄存器中的 OISx 和 OISxN 位)，输出使能信号和无效电平都会被修改。但无论何时， OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见表：带刹车功能的互补输出通道

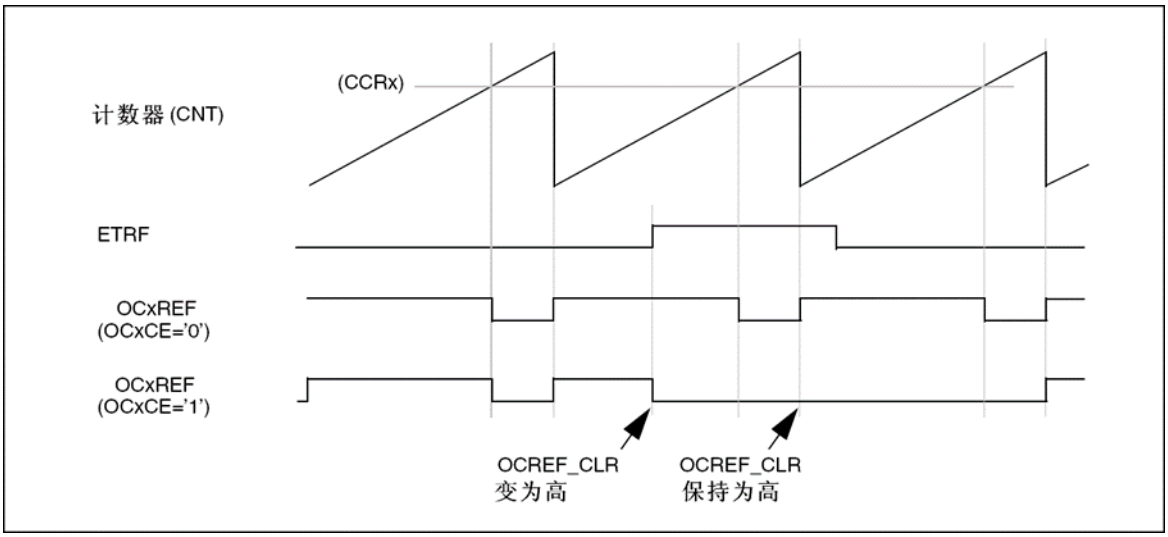
OCx 和 OCxN 的控制位。
刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生。
系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。
因为 MOE 下降沿可以是异步的，在实际信号(作用在输出端)和同步控制位(在 TIMx_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。
当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：
●MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
●一旦 MOE=0，每一个输出通道输出由 TIMx_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
●当使用互补输出时：
—输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
—如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
—如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
●如果设置了 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为’1’时，则产生一个中断。如果设置了 TIMx_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
●如果设置了 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位； 例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置’1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。
注：刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TIMx_BDTR 寄存器中的 BKE 位开启。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性)。用户可以通过 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 13.4.18 节 TIM1 和 TIM8 刹车和死区寄存器(TIMx_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。下图显示响应刹车的输出实例。
图：响应刹车的输出（先输出死区再刹车）



11.4.14在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为’1’，能够用 ETRF 输入端的高电平把 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。
该功能只能用于输出比较和 PWM 模式，而不能用于强置模式。
例如，OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：
1.外部触发预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。
2.必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。
3.外部触发极性(ETP)和外部触发滤波器(ETF)可以根据需要配置。
下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作。在这个例子中，定时器 TIMx 被置于 PWM 模式。

图：清除 TIMx 的 OCxREF



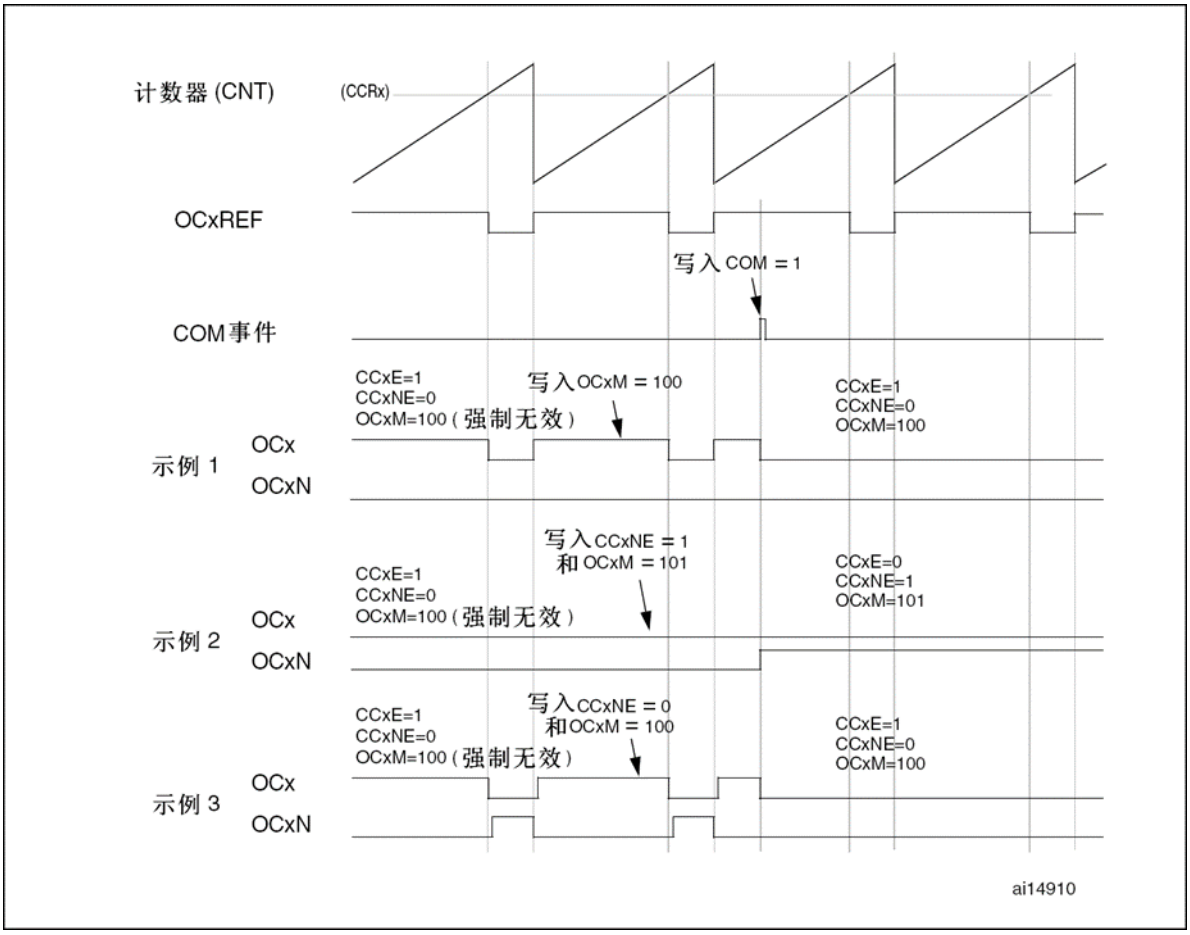
11.4.15产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。COM 可以通过设置 TIMx_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位(TIMx_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIMx_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

下图显示当发生 COM 事件时，三种不同配置下 OCx 和 OCxN 输出。

图：产生六步 PWM，使用 COM 的例子(OSSR=1)



11.4.16单脉冲模式

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

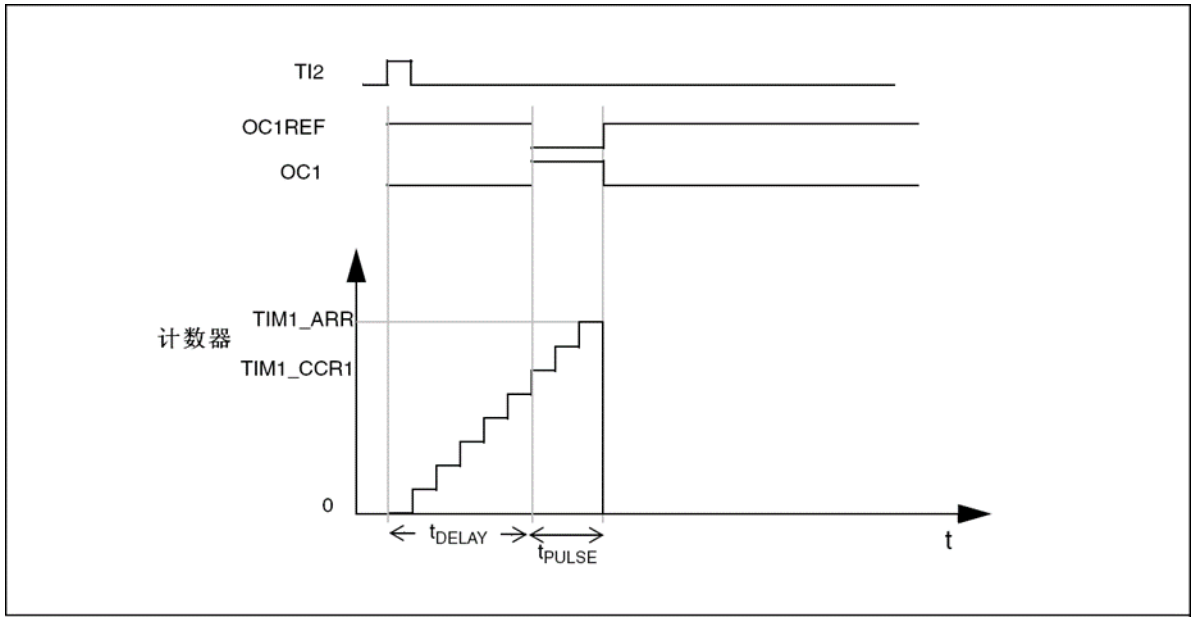
可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)， 必须如下配置：

●向上计数方式：计数器 CNT < CCRx ≤ ARR (特别地, 0 < CCRx)，

●向下计数方式：计数器 CNT > CCRx。

图：单脉冲模式的例子



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

假定 TI2FP2 作为触发 1:

●置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

●置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。

●置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。

●置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)

●tDELAY 由 TIMx_CCR1 寄存器中的值定义。

●tPULSE 由自动装载值和比较值之间的差值定义(TIMx_ARR - TIMx_CCR1)。

●假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；

首先要置 TIMx_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIMx_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

特殊情况：OCx 快速使能：

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果要以最小延时输出波形，可以设置 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM1 和 PWM2 模式时起作用。

11.4.17编码器接口模式

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

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 73，假定计数器已经启动(TIMx_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1， TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

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

表：计数方向与编码器信号的关系

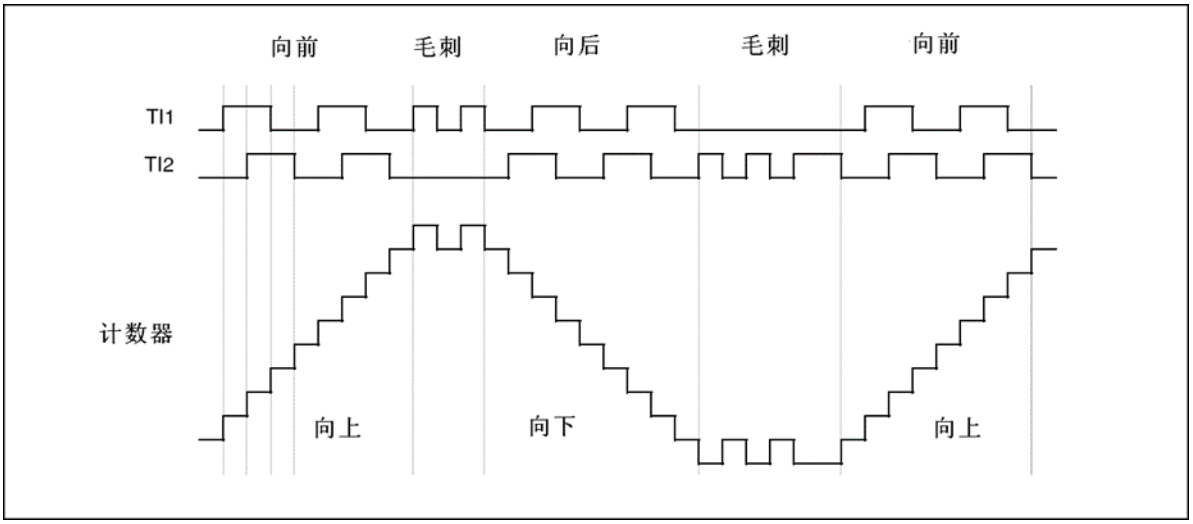
有效边沿	相对信号的电平 (TI1FP1对应TI2, TI2FP2对应TI1)	TI1FP1信号		TI2FP2信号	
		上升	下降	上升	下降
仅在TI1计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在TI2计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在TI1和TI2上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

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

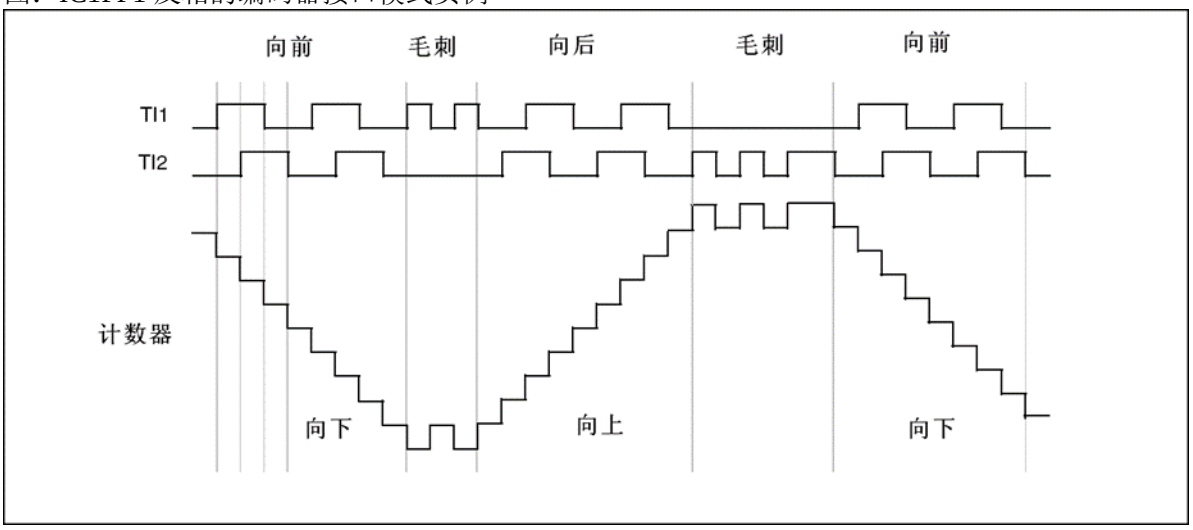
下图是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

●CC1S=’01’ (TIMx_CCMR1 寄存器，IC1FP1 映射到 TI1)
●CC2S=’01’ (TIMx_CCMR2 寄存器，IC2FP2 映射到 TI2)
●CC1P=’0’ (TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
●CC2P=’0’ (TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
●SMS=’011’ (TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效).
●CEN=’1’ (TIMx_CR1 寄存器，计数器使能)

图：编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例(CC1P=’1’，其他配置与上例相同)
图：IC1FP1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息(速度，加速度，减速度)。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，你可以把计数器的值锁存到第三个输入捕获寄存器(捕获信号必须是周期的并且可以由另一个定时器产生)；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

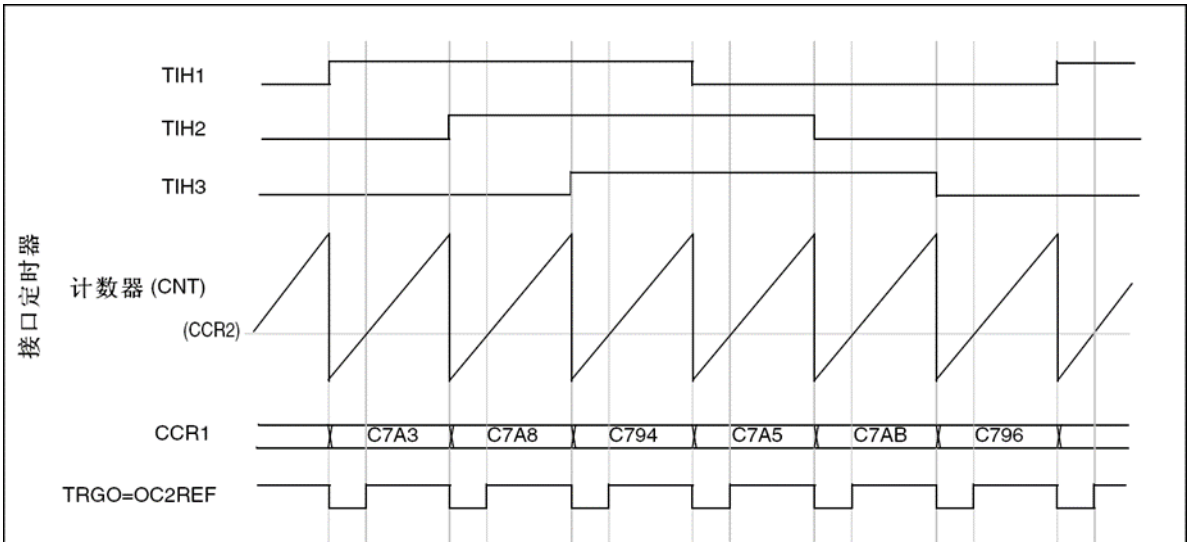
11.4.18定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。
异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下节给出了此特性用于连接霍尔传感器的例子。

11.4.19与霍尔传感器的接口

使用高级控制定时器(TIM1 或 TIM8)产生 PWM 信号驱动马达时，可以用另一个通用 TIMx(TIM2、

TIM3、TIM4 或 TIM5)定时器作为“接口定时器”来连接霍尔传感器，3 个定时器输入脚(CC1、CC2、CC3)通过一个异或门连接到 TI1 输入通道(通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，“接口定时器”捕获这个信号。
从模式控制器被配置于复位模式，从输入是 TI1F_ED。每当 3 个输入之一变化时，计数器从新从 0 开始计数。这样产生一个由霍尔输入端的任何变化而触发的时间基准。
“接口定时器”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC(见图 76)。捕获值反映了两个输入变化间的时间延迟，给出了马达速度的信息。
“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以(通过触发一个 COM 事件)用于改变高级定时器 TIM1 或 TIM8 各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。因此“接口定时器”通道必须编程为在一个指定的延时(输出比较或 PWM 模式)之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器 TIM1 或 TIM8。
举例：霍尔输入连接到 TIMx 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIMx 的 PWM 配置。
●置 TIMx_CR2 寄存器的 TI1S 位为’1’，配置三个定时器输入逻辑或到 TI1 输入，
●时基编程：置 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
●设置通道 1 为捕获模式(选中 TRC)：置 TIMx_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
●设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
●选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。
在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的(TIMx_CR2 寄存器中 CCPC=1)，同时触发输入控制 COM 事件(TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入下一步的 PWM 控制位(CCxE、OCxM)，这可以在处理 OC2REF 上升沿的中断子程序里实现。
下图显示了这个实例
图：霍尔传感器接口的实例



11.4.20TIMx 定时器和外部触发的同步

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

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器(TIMx_ARR，TIMx_CCRx)都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

●配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。

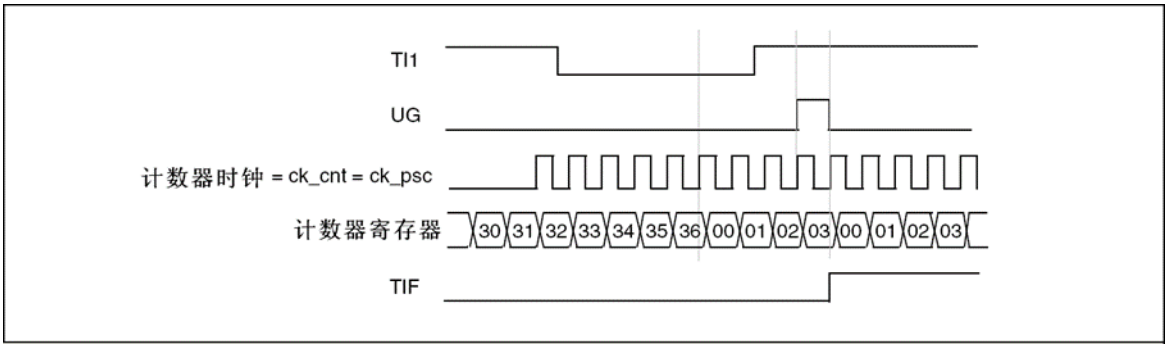
●置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

●置 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位和 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIMx_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

图：复位模式下的控制电路



从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

●配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。

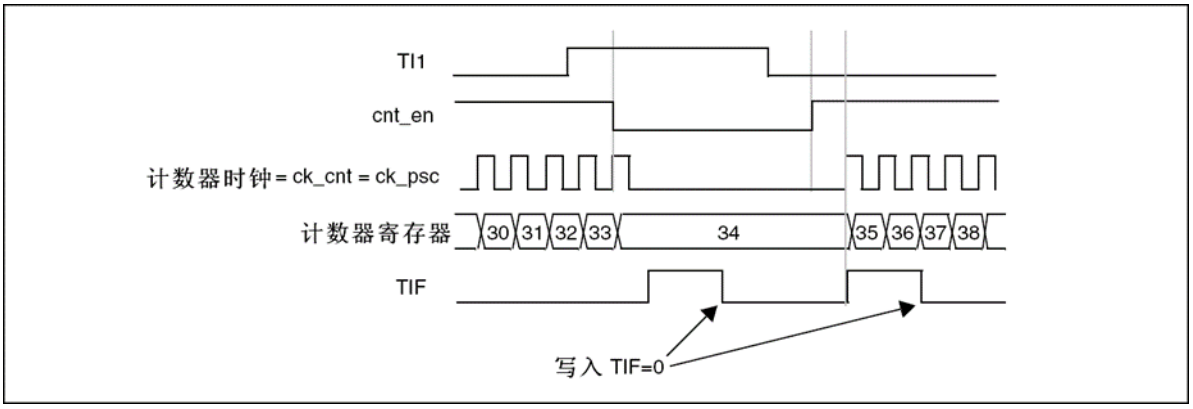
●置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

●置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

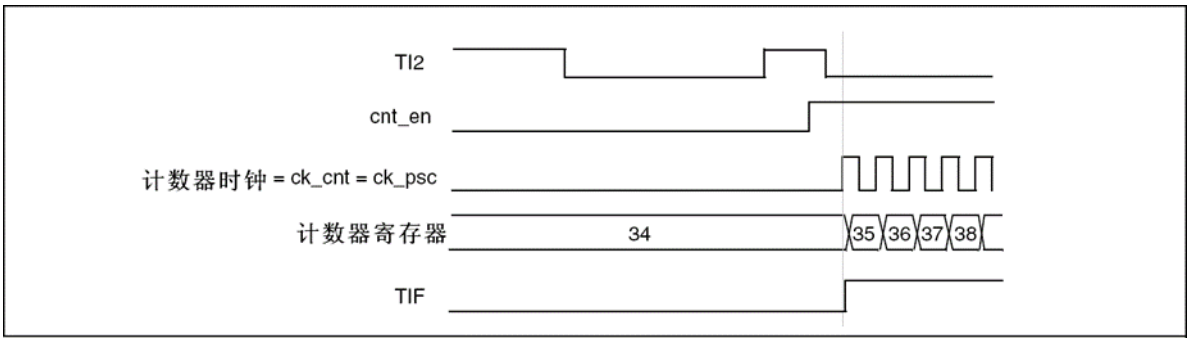
只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMx_SR 中的 TIF 标置。

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

图：门控模式下的控制电路

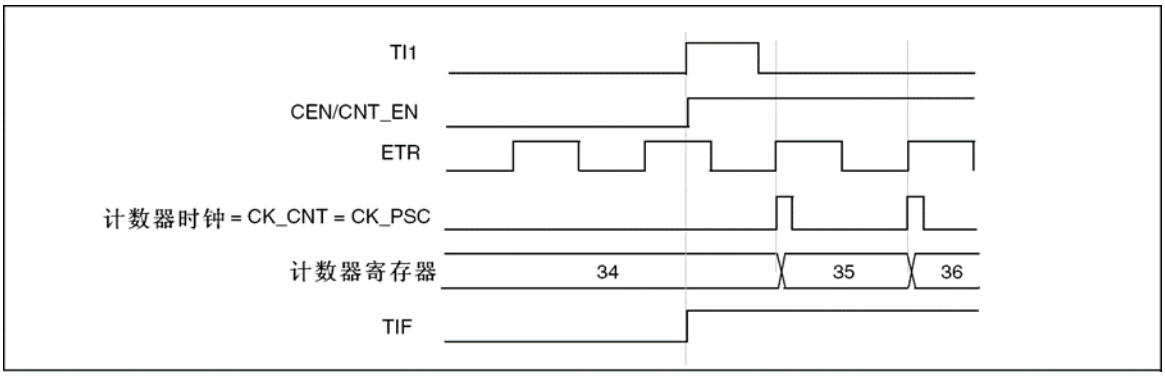


从模式：触发模式
输入端上选中的事件使能计数器。
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：
●配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性(只检测低电平)。
●置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。
当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。
TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。
图：触发器模式下的控制电路



从模式：外部时钟模式 2 ＋ 触发模式
外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。
在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：
1.通过 TIMx_SMCR 寄存器配置外部触发输入电路：
—ETF=0000：没有滤波
—ETPS=00：不用预分频器
—ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。
2.按如下配置通道 1，检测 TI 的上升沿：
—IC1F=0000：没有滤波
—触发操作中不使用捕获预分频器，不需要配置
—置 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
—置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)
3.置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。
ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

图：外部时钟模式 2＋触发模式下的控制电路



11.4.21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。

11.5 TIM1 和 TIM8 和 TIM20 寄存器

11.5.1 TIM1 和 TIM8 和 TIM20 控制寄存器 2(TIMx_CR2)

TIM1 和 TIM8 和 TIM20 控制寄存器 2(TIMx_CR2)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	OIS6
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OIS5	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TI1S	MMS[6:4]			CCDS	CCUS	X	CCPC
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OIS6	输出空闲状态 6(OC6 输出)。参见 OIS1 位。
OIS5	输出空闲状态 5(OC5 输出)。参见 OIS1 位。
OIS4	输出空闲状态 4(OC4 输出)。参见 OIS1 位。

OIS3N	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位。
OIS3	输出空闲状态 3(OC3 输出)。参见 OIS1 位。
OIS2N	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。
OIS2	输出空闲状态 2(OC2 输出)。参见 OIS1 位。
OIS1N	输出空闲状态1(OC1N输出) (Output Idle state 1) 0：当MOE=0时，死区后OC1N=0。 1：当MOE=0时，死区后OC1N=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
OIS1	输出空闲状态1(OC1输出) (Output Idle state 1) 0：当MOE=0时，如果实现了OC1N，则死区后OC1=0； 1：当MOE=0时，如果实现了OC1N，则死区后OC1=1。 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。
TI1S	TI1 选择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入。
MMS[6:4]	主模式选择(Master mode selection) 这3位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位 – TIMx_EGR寄存器的UG位被用于作为触发输出(TRGO)。如果是触发输入产生的 复位(从模式控制器处于复位模式)，则TRGO上的信号相对实际的复位会有一个延迟。 001：使能 – 计数器使能信号CNT_EN被用于作为触发输出(TRGO)。有时需要在同一时间启动 多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过CEN控制位和门控模式 下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO上会有一个延 迟，除非选择了主/从模式(见TIMx_SMCR寄存器中MSM位的描述)。 010：更新 – 更新事件被选为触发输入(TRGO)。例如，一个主定时器的时钟可以被用作一个从 定时器的预分频器。 011：比较脉冲 – 在发生一次捕获或一次比较成功时，当要设置CC1IF标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较 – OC1REF信号被用于作为触发输出(TRGO)。 101：比较 – OC2REF信号被用于作为触发输出(TRGO)。 110：比较 – OC3REF信号被用于作为触发输出(TRGO)。 111：比较 – OC4REF 信号被用于作为触发输出(TRGO)。
CCDS	捕获/比较的 DMA 选择 (Capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求； 1：当发生更新事件时，送出 CCx 的 DMA 请求。
CCUS	捕获/比较控制更新选择(Capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置COM位更新它们； 1：如果捕获/比较控制位是预装载的(CCPC=1)，可以通过设置COM位或TRGI上的一个上升沿 更新它们。 注：该位只对具有互补输出的通道起作用。
CCPC	捕获/比较预装载控制位 (Capture/compare preloaded control) 0：CCxE，CCxNE 和 OCxM 位不是预装载的；

	1：CCxE，CCxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新。 注：该位只对具有互补输出的通道起作用。

11.5.2 TIM1 和 TIM8 和 TIM20 DMA/中断使能寄存器(TIMx_DIER)

TIM1 和 TIM8 和 TIM20 DMA/中断使能寄存器 (TIMx_DIER)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	CC6DE	CC6IE	CC5DE
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CC5IE	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC6DE	允许捕获/比较5的DMA请求 (Capture/Compare 5 DMA request enable) 0：禁止捕获/比较5的DMA请求； 1：允许捕获/比较5的DMA请求。
CC6IE	允许捕获/比较6中断 (Capture/Compare 6 interrupt enable) 0：禁止捕获/比较6中断； 1：允许捕获/比较6中断。
CC5DE	允许捕获/比较5的DMA请求 (Capture/Compare 5 DMA request enable) 0：禁止捕获/比较5的DMA请求； 1：允许捕获/比较5的DMA请求。
CC5IE	允许捕获/比较5中断 (Capture/Compare 5 interrupt enable) 0：禁止捕获/比较5中断； 1：允许捕获/比较 5 中断。
TDE	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。
COMDE	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。

CC4DE	**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。
CC3DE	**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。
CC2DE	**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。
CC1DE	**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。
UDE	**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。
BIE	**允许刹车中断** (Break interrupt enable) 0：禁止刹车中断； 1：允许刹车中断。
TIE	**触发中断使能** (Trigger interrupt enable) 0：禁止触发中断； 1：使能触发中断。
COMIE	**允许COM中断**(COM interrupt enable) 0：禁止COM中断； 1：允许 COM 中断。
CC4IE	**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。
CC3IE	**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。
CC2IE	**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。
CC1IE	**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。
UIE	**允许更新中断** (Update interrupt enable) 0：禁止更新中断； 1：允许更新中断。

11.5.3 TIM1 和 TIM8 和 TIM20 状态寄存器(TIMx_SR)

TIM1 和 TIM8 和 TIM20 状态寄存器(TIMx_SR)	基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)
	偏移地址： 0x10

	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	CC6IF	CC5IF	CC4OF	CC3OF	CC2OF	CC1OF	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC6IF	**捕获/比较 6 中断标记** 参考 CC1IF 描述。
CC5IF	**捕获/比较 5 中断标记** 参考 CC1IF 描述。
CC4OF	**捕获/比较4重复捕获标记**(Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。
CC3OF	**捕获/比较3重复捕获标记**(Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。
CC2OF	**捕获/比较2重复捕获标记**(Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。
CC1OF	**捕获/比较1重复捕获标记**(Capture/Compare 1 overcapture flag) 仅当相应的通道被配置为输入捕获时，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为’1’。
BIF	**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。
TIF	**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。
COMIF	**COM中断标记**(COM interrupt flag) 一旦产生COM事件(当捕获/比较控制位：CCxE、CCxNE、OCxM已被更新)该位由硬件置’1’。它由软件清’0’。 0：无COM事件产生； 1：COM 中断等待响应。
CC4IF	**捕获/比较4中断标记**(Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
CC3IF	**捕获/比较3中断标记**(Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
CC2IF	**捕获/比较2中断标记**(Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。

CC1IF	**捕获/比较1中断标记(Capture/Compare 1 interrupt flag)** **如果通道CC1配置为输出模式：** 当计数器值与比较值匹配时该位由硬件置1，但在中心对称模式下除外(参考TIMx_CR1寄存器 的CMS位)。它由软件清’0’。 0：无匹配发生； 1：TIMx_CNT的值与TIMx_CCR1的值匹配。 当TIMx_CCR1的内容大于TIMx_APR的内容时，在向上或向上/下计数模式时计数器溢出，或 向下计数模式时的计数器下溢条件下，CC1IF位变高 **如果通道CC1配置为输入模式：** 当捕获事件发生时该位由硬件置’1’，它由软件清’0’或通过读TIMx_CCR1清’0’。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIF	**更新中断标记(Update interrupt flag)** 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： – 若TIMx_CR1寄存器的UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0时产生更新事件)。 – 若TIMx_CR1寄存器的URS=0、UDIS=0，当设置TIMx_EGR寄存器的UG=1时产生更新事件，通过软件对计数器CNT重新初始化时。 – 若TIMx_CR1寄存器的URS=0、UDIS=0，当计数器CNT被触发事件重新初始化时。

11.5.4 TIM1 和 TIM8 和 TIM20 事件产生寄存器(TIMx_EGR)

TIM1 和 TIM8 和 TIM20 事件产生寄存器(TIMx_EGR)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x14					
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	X	X	X	X	X	X	CC6G	CC5G
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC6G	**产生捕获/比较 6 事件 (Capture/Compare 5 generation)** 参考 CC1G 描述。
CC5G	**产生捕获/比较 5 事件 (Capture/Compare 5 generation)**

	参考 CC1G 描述。
BG	**产生刹车事件(Break generation)** 该位由软件置’1’，用于产生一个刹车事件，由硬件自动清’0’。 0：无动作； 1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断 和 DMA。
TG	**产生触发事件(Trigger generation)** 该位由软件置’1’，用于产生一个触发事件，由硬件自动清’0’。 0：无动作； 1：TIMx_SR 寄存器的 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
COMG	**捕获/比较事件，产生控制更新(Capture/Compare control update generation)** 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 **注：**该位只对拥有互补输出的通道有效。
CC4G	**产生捕获/比较4事件(Capture/Compare 4 generation)** 参考 CC1G 描述。
CC3G	**产生捕获/比较3事件(Capture/Compare 3 generation)** 参考 CC1G 描述。
CC2G	**产生捕获/比较2事件(Capture/Compare 2 generation)** 参考 CC1G 描述。
CC1G	**产生捕获/比较 1 事件 (Capture/Compare 1 generation)** 该位由软件置’1’，用于产生一个捕获/比较事件，由硬件自动清’0’。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
UG	**产生更新事件 (Update generation)** 该位由软件置’1’，由硬件自动清’0’。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清’0’(但是预分频系 数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清’0’；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。

11.5.5 TIM1 和 TIM8 和 TIM20 捕获/比较使能寄存器(TIMx_CCER)

TIM1 和 TIM8 和 TIM20 捕获/比较使能寄存器(TIMx_CCER)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read:	X	X	X	X	X	X	X	X

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	CC6P	CC6E
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CC5P	CC5E	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CC6P	输入/捕获 6 输出极性 参考 CC1P 的描述
CC6E	输入/捕获 6 输出使能 参考 CC1E 的描述
CC5P	输入/捕获 5 输出极性 参考 CC1P 的描述
CC5E	输入/捕获 5 输出使能 参考 CC1E 的描述
CC4P	输入/捕获4输出极性(Capture/Compare 4 output polarity) 参考 CC1P 的描述。
CC4E	输入/捕获4输出使能(Capture/Compare 4 output enable) 参考 CC1E 的描述。
CC3NP	输入/捕获3互补输出极性(Capture/Compare 3 complementary output polarity) 参考 CC1NP 的描述。
CC3NE	输入/捕获3互补输出使能(Capture/Compare 3 complementary output enable) 参考 CC1NE 的描述。
CC3P	输入/捕获3输出极性(Capture/Compare 3 output polarity) 参考 CC1P 的描述。
CC3E	输入/捕获3输出使能(Capture/Compare 3 output enable) 参考 CC1E 的描述。
CC2NP	输入/捕获2互补输出极性(Capture/Compare 2 complementary output polarity) 参考 CC1NP 的描述。
CC2NE	输入/捕获2互补输出使能(Capture/Compare 2 complementary output enable) 参考 CC1NE 的描述。
CC2P	输入/捕获2输出极性(Capture/Compare 2 output polarity) 参考 CC1P 的描述。
CC2E	输入/捕获2输出使能(Capture/Compare 2 output enable) 参考 CC1E 的描述。
CC1NP	输入/捕获1互补输出极性 (Capture/Compare 1 complementary output polarity)

	0：OC1N高电平有效； 1：OC1N低电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出) 则该位不能被修改。
CC1NE	**输入/捕获1互补输出使能**(Capture/Compare 1 complementary output enable) 0：关闭－ OC1N禁止输出，因此OC1N的电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1E位的值。 1：开启－ OC1N信号输出到对应的输出引脚，其输出电平依赖于MOE、OSSI、OSSR、OIS1、OIS1N和CC1E位的值。
CC1P	**输入/捕获1输出极性**(Capture/Compare 1 output polarity) **CC1通道配置为输出：** 0：OC1高电平有效； 1：OC1低电平有效。 **CC1通道配置为输入：** 该位选择是IC1还是IC1的反相信号作为触发或捕获信号。 0：不反相：捕获发生在IC1的上升沿；当用作外部触发器时，IC1不反相。 1：反相：捕获发生在IC1的下降沿；当用作外部触发器时，IC1反相。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。
CC1E	**输入/捕获 1 输出使能**(Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0： 关闭－ OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。 1： 开启－ OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、 OIS1、OIS1N 和 CC1NE 位的值。 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

11.5.6 TIM1 和 TIM8 和 TIM20 刹车和死区寄存器(TIMx_BDTR)

TIM1 和 TIM8 和 TIM20 刹车和死区寄存器(TIMx_BDTR)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x44					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	BKF[27:24]			
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	PVD_OE	Hardfault_OE	CMP3_OEP	CMP3_OE	CMP2_OEP	CMP2_OE	CMP1_OEP	CMP1_OE
Reset:	0	0	0	0	0	0	0	0

	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DTG[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BKF[27:24]	**输入的采样频率及数字滤波器长度。** 数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变 0000: No filter, BRK acts asynchronously 0001: fSAMPLING=fCK_INT, N=2 0010: fSAMPLING=fCK_INT, N=4 0011: fSAMPLING=fCK_INT, N=8 0100: fSAMPLING=fDTS/2, N=6 0101: fSAMPLING=fDTS/2, N=8 0110: fSAMPLING=fDTS/4, N=6 0111: fSAMPLING=fDTS/4, N=8 1000: fSAMPLING=fDTS/8, N=6 1001: fSAMPLING=fDTS/8, N=8 1010: fSAMPLING=fDTS/16, N=5 1011: fSAMPLING=fDTS/16, N=6 1100: fSAMPLING=fDTS/16, N=8 1101: fSAMPLING=fDTS/32, N=5 1110: fSAMPLING=fDTS/32, N=6 1111: fSAMPLING=fDTS/32, N=8
PVD_OE	**刹车功能 PVD_OUT 使能 (Break enable)** 0：禁止刹车输入 1：开启刹车输入 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
Hardfault_OE	**刹车功能 Hardfault_out 使能 (Break enable)** 0：禁止刹车输入 1：开启刹车输入 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
CMP3_OEP	**刹车输入 CMP3_out 极性 (Break polarity)** 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
CMP3_OE	**刹车功能 CMP3_out 使能 (Break enable)** 0：禁止刹车输入 1：开启刹车输入

	注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
CMP2_OEP	**刹车输入 CMP2_out 极性 (Break polarity)** 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
CMP2_OE	**刹车功能 CMP2_out 使能 (Break enable)** 0：禁止刹车输入 1：开启刹车输入 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
CMP1_OEP	**刹车输入CMP1_out极性 (Break polarity)** 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个APB时钟的延迟以后才能起作用。
CMP1_OE	**刹车功能 CMP1_out 使能 (Break enable)** 0：禁止刹车输入 1：开启刹车输入 注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
MOE	**主输出使能 (Main output enable)** 一旦刹车输入有效，该位被硬件异步清’0’。根据 AOE 位的设置值，该位可以由软件清’0’或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位(TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输 出。 有关 OC/OCN 使能的细节，参见 13.4.9 节，TIM8 和 TIM8 捕获/比较使能寄存器(TIMx_CCER)。
AOE	**自动输出使能 (Automatic output enable)** 0：MOE 只能被软件置’1’； 1：MOE 能被软件置’1’或在下一个更新事件被自动置’1’(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。
BKP	**刹车输入极性 (Break polarity)** 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
BKE	**刹车功能使能 (Break enable)** 0：禁止刹车输入(BRK 及 CCS 时钟失效事件)； 1：开启刹车输入(BRK 及 CCS 时钟失效事件)。

	注：当设置了 LOCK 级别 1 时(TIMx_BDTR 寄存器中的 LOCK 位)，该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
OSSR	**运行模式下“关闭状态”选择 (Off-state selection for Run mode)** 该位用于当MOE=1且通道为互补输出时。没有互补输出的定时器中不存在OSSR位。 参考OC/OCN使能的详细说明TIM8和TIM8捕获/比较使能寄存器(TIMx_CCER)。 0：当定时器不工作时，禁止OC/OCN输出(OC/OCN使能输出信号=0)； 1：当定时器不工作时，一旦CCxE=1或CCxNE=1，首先开启OC/OCN并输出无效电平，然后置OC/OCN使能输出信号=1。 注：一旦LOCK级别(TIMx_BDTR寄存器中的LOCK位)设为2，则该位不能被修改。
OSSI	**空闲模式下“关闭状态”选择 (Off-state selection for Idle mode)** 该位用于当 MOE=0 且通道设为输出时。 参考 OC/OCN 使能的详细说明(13.4.9 节，TIM8 和 TIM8 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。
LOCK	**锁定设置** 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、CMP1_OE、CMP1_OEP、CMP2_OE、CMP2_OEP、CMP3_OE、CMP3_OEP、Hardfault_OE、AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCNxP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OCxM/OCxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。
DTG[7:0]	**死区发生器** 设置这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间： DTG[7:5]=0xx => DT=DTG[7:0] × Tdtg，Tdtg = TDTS； DTG[7:5]=10x => DT=(64+DTG[5:0]) × Tdtg，Tdtg = 2 × TDTS； DTG[7:5]=110 => DT=(32+DTG[4:0]) × Tdtg，Tdtg = 8 × TDTS； DTG[7:5]=111 => DT=(32+DTG[4:0])× Tdtg，Tdtg = 16 × TDTS； 例：若 TDTS = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位

11.5.7 TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 5 (TIMx_CCR5)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器(TIMx_CCR5)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)					
			偏移地址： 0x54					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR5[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR5[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR5[15:0]	捕获/比较通道5的值 (Capture/Compare 5 value) 若CC5通道配置为输出： CCR5包含了装入当前捕获/比较5寄存器的值(预装载值)。 如果在TIMx_CCMR4寄存器(OC4PE位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较5寄存器中。 当前捕获/比较寄存器参与同计数器TIMx_CNT的比较，并在OC5端口上产生输出信号。 若CC5通道配置为输入： CCR5 包含了由上一次输入捕获 5 事件(IC5)传输的计数器值。

11.5.8 TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 (TIMx_CCMR3)

TIM1 和 TIM8 和 TIM20 CCMR3 (TIMx_CCMR3)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)					
			偏移地址： 0x58					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OC6CE	OC6M[14:12]			OC6PE	OC6FE	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OC5CE	OC5M[6:4]			OC5PE	OC5FE	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OC6CE	输出比较 6 清 0 使能
OC6M[14:12]	输出比较 6 模式
OC6PE	输出比较 6 预装载使能
OC6FE	输出比较 6 快速使能
OC5CE	输出比较 5 清 0 使能
OC5M[6:4]	输出比较 5 模式
OC5PE	输出比较 5 预装载使能
OC5FE	输出比较 5 快速使能

11.5.9 TIM1 和 TIM8 和 TIM20 控制寄存器 1(TIMx_CR1)

TIM1 和 TIM8 和 TIM20 控制寄存器 1(TIMx_CR1)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	CKD[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ARPE	CMS[6:5]		DIR	OPM	URS	UDIS	CEN
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CKD[9:8]	**时钟分频因子 (Clock division)** 这 2 位定义在定时器时钟(CK_INT)频率、死区时间和由死区发生器与数字滤波器(ETR,TIx)所用 的采样时钟之间的分频比例。 00：tDTS = tCK_INT 01：tDTS = 2 x tCK_INT 10：tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
ARPE	**自动重装载预装载允许位 (Auto-reload preload enable)** 0：TIMx_ARR 寄存器没有缓冲； 1：TIMx_ARR 寄存器被装入缓冲器。
CMS[6:5]	**选择中央对齐模式 (Center-aligned mode selection)** 00：边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，只在计数器向下计数时被设置。 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，只在计数器向上计数时被设置。 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道(TIMx_CCMRx 寄存器 中 CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被设置。 注：在计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
DIR	**计数方向 (Direction)** 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
OPM	**单脉冲模式 (One pulse mode)** 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除 CEN 位)时，计数器停止。
URS	**更新请求源 (Update request source)** 软件通过该位选择 UEV 事件的源 0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： - 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出/下溢才产生更新中断或 DMA 请求。
UDIS	**禁止更新 (Update disable)** 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新(UEV)事件由下述任一事件产生： - 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值。(译注：更新影子寄存器)

	1：禁止 UEV。不产生更新事件，影子寄存器(ARR、PSC、CCRx)保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。
CEN	**使能计数器（Counter enable）** 0：禁止计数器； 1：使能计数器。 注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动 地通过硬件设置 CEN 位。

11.5.10TIM1 和 TIM8 和 TIM20 从模式控制寄存器(TIMx_SMCR)

TIM1 和 TIM8 和 TIM20 从模式控制寄存器 (TIMx_SMCR)			基地址： 0x40012C00(TIM1),0x40013400(TIM8) 0x40015000(TIM20) 偏移地址： 0x08					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	ETP	ECE	ETPS[13:12]		ETF[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	MSM	TS[6:4]			X	SMS[2:0]		
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ETP	**外部触发极性（External trigger polarity）** 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0：ETR 不反相，高电平或上升沿有效； 1：ETR 被反相，低电平或下降沿有效。
ECE	**外部时钟使能位（External clock enable）** 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。
ETPS[13:12]	**外部触发预分频（External trigger prescaler）** 外部触发信号 ETRP 的频率必须最多是 TIMxCLK 频率的 1/4。当输入较快的外部时

	钟时，可以使 用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
ETF[11:8]	**外部触发滤波（External trigger filter）** 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个 事件计数器，它记录到 N 个事件后会产生一个输出的跳变。 0000：无滤波器，以 fDTS 采样 0001：采样频率 fSAMPLING=fCK_INT，N=2 0010：采样频率 fSAMPLING=fCK_INT，N=4 0011：采样频率 fSAMPLING=fCK_INT，N=8 0100：采样频率 fSAMPLING=fDTS/2，N=6 0101：采样频率 fSAMPLING=fDTS/2，N=8 0110：采样频率 fSAMPLING=fDTS/4，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1000：采样频率 fSAMPLING=fDTS/8，N=6 1001：采样频率 fSAMPLING=fDTS/8，N=8 1010：采样频率 fSAMPLING=fDTS/16，N=5 1011：采样频率 fSAMPLING=fDTS/16，N=6 1100：采样频率 fSAMPLING=fDTS/16，N=8 1101：采样频率 fSAMPLING=fDTS/32，N=5 1110：采样频率 fSAMPLING=fDTS/32，N=6 1111：采样频率 fSAMPLING=fDTS/32，N=8
MSM	**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。
TS[6:4]	**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 TIMx 内部触发连接。 注：这些位只能在未用到(如 SMS=000)时被改变，以避免在改变时产生错误的边沿检测。
SMS[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 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。 011：编码器模式 3 – 根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。

	100：复位模式 – 选中的触发输入(TRGI)的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。 111：外部时钟模式 1 – 选中的触发输入(TRGI)的上升沿驱动计数器。 注：如果 TI1F_EN 被选为触发输入(TS=100)时，不要使用门控模式。这是因为，TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。

表：TIMx 内部触发连接

从定时器	ITR0(TS = 000)	ITR1(TS = 001)	ITR2(TS = 010)	ITR3(TS = 011)
TIM1	保留	TIM2	TIM3	TIM4
TIM8	TIM1	TIM2	TIM4	保留
TIM20	TIM1	TIM2	TIM3	TIM8

11.5.11TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 1(TIMx_CCMR1)

TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 1(TIMx_CCMR1)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	OC2CE	OC2M[14:12]			OC2PE	OC2FE	CC2S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OC1CE	OC1M[6:4]			OC1PE	OC1FE	CC1S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输出比较模式

位	功能描述
OC2CE	输出比较 2 清 0 使能
OC2M[14:12]	输出比较 2 模式
OC2PE	输出比较 2 预装载使能
OC2FE	输出比较 2 快速使能

CC2S[9:8]	**捕获/比较 2 选择。(Capture/Compare 2 selection)** 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
OC1CE	**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。
OC1M[6:4]	**输出比较 1 模式 (Output Compare 1 mode)** 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P CC1NP. 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001 ：匹配 时 设置通道 1 为 有效电平。 当计数器 TIMx_CNT 的值与 捕获 / 比较寄 存器 1(TIMx_CCR1)相同时，强制 OC1REF 为高。 010 ：匹配 时 设置通道 1 为 无效电平。 当计数器 TIMx_CNT 的值与 捕获 / 比较寄 存器 1(TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1— 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2— 在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输 出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式 切换到 PWM 模式时，OC1REF 电平才改变。
OC1PE	**输出比较 1 预装载使能 (Output Compare 1 preload enable)** 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数 值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的 预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输 出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。
OC1FE	**输出比较 1 快速使能 (Output Compare 1 fast enable)** 该位用于加快 CC 输出对触发输入事件的响应。

	0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入有一个有 效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与 比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
CC1S[1:0]	**捕获/比较 1 选择**(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

TIM1 和 TIM8 捕获/比较模式寄存器 1(TIMx_CCMR1)			**基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)**					
			偏移地址： 0x18					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	IC2F[15:12]				IC2PSC[11:10]		CC2S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	IC1F[7:4]				IC1PSC[3:2]		CC1S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输入捕获模式

位	功能描述
IC2F[15:12]	**输入捕获 2 滤波器**
IC2PSC[11:10]	**输入/捕获 2 预分频器**
CC2S[9:8]	**捕获/比较 2 选择** (Capture/Compare 2 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上；

	10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。
IC1F[7:4]	**输入捕获 1 滤波器** 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样 0001：采样频率 fSAMPLING=fCK_INT，N=2 0010：采样频率 fSAMPLING=fCK_INT，N=4 0011：采样频率 fSAMPLING=fCK_INT，N=8 0100：采样频率 fSAMPLING=fDTS/2，N=6 0101：采样频率 fSAMPLING=fDTS/2，N=8 0110：采样频率 fSAMPLING=fDTS/4，N=6 0111：采样频率 fSAMPLING=fDTS/4，N=8 1000：采样频率 fSAMPLING=fDTS/8，N=6 1001：采样频率 SAMPLING=fDTS/8，N=8 1010：采样频率 SAMPLING=fDTS/16，N=5 1011：采样频率 SAMPLING=fDTS/16，N=6 1100：采样频率 SAMPLING=fDTS/16，N=8 1101：采样频率 SAMPLING=fDTS/32，N=5 1110：采样频率 SAMPLING=fDTS/32，N=6 1111：采样频率 SAMPLING=fDTS/32，N=8
IC1PSC[3:2]	**输入/捕获 1 预分频器 (Input capture 1 prescaler)** 这 2 位定义了 CC1 输入(IC1)的预分频系数。 一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
CC1S[1:0]	**捕获/比较 1 选择 (Capture/Compare 1 Selection)** 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。

关于寄存器的相关配置如下说明：

寄存器配置	CMS=0,DIR=0,向上计数,OC1M=111	CMS=0,DIR=1,向下计数,OC1M=111
CCR1N_EN=1	CCR1<=TIMx_CNT<CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx<CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx<CCRxN);	CCR1N<TIMx_CNT<=CCR1,通道 1 为无效电平(OC1REF=0),(CCRx>CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx>CCRxN);

CCR1N_EN=0	TIMx_CNT<CCR1,通道 1 为无效电平(OC1REF=0); TIMx_CNT>=CCR1,通道 1 为有效电平(OC1REF=1);	TIMx_CNT<=CCR1,通道 1 为无效电平(OC1REF=0); TIMx_CNT>CCR1,通道 1 为有效电平(OC1REF=1);
寄存器配置	**CMS=0,DIR=0,向上计数,OC1M=110**	**CMS=0,DIR=1,向下计数,OC1M=110**
CCR1N_EN=1	CCR1<=TIMx_CNT<CCR1N,通道 1 为有效电平(OC1REF=1),(CCRx<CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx<CCRxN);	CCR1N<TIMx_CNT<=CCR1,通道 1 为有效电平(OC1REF=1),(CCRx>CCRxN); TIMx_CNT<CCR1,TIMx_CNT>=CCR1N,通道 1 为无效电平(OC1REF=0),(CCRx>CCRxN);
CCR1N_EN=0	TIMx_CNT<CCR1,通道 1 为有效电平(OC1REF=1); TIMx_CNT>=CCR1,通道 1 为无效电平(OC1REF=0);	TIMx_CNT<=CCR1,通道 1 为有效电平(OC1REF=1); TIMx_CNT>CCR1,通道 1 为无效电平(OC1REF=0);

11.5.12TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 2(TIMx_CCMR2)

TIM1 和 TIM8 和 TIM20 捕获/比较模式寄存器 2(TIMx_CCMR2)			**基地址： 0x40012C00(TIM1),0x40013400(TIM8) 0x40015000(TIM20)** **偏移地址： 0x1C**					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	OC4CE	OC4M[14:12]			OC4PE	OC4FE	CC4S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	OC3CE	OC3M[6:4]			OC3PE	OC3FE	CC3S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输出比较模式

位	功能描述
OC4CE	输出比较 4 清 0 使能
OC4M[14:12]	输出比较 4 模式
OC4PE	输出比较 4 预装载使能

OC4FE	**输出比较 4 快速使能**
CC4S[9:8]	**捕获/比较 4 选择 (Capture/Compare 4 selection)** 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
OC3CE	**输出比较 3 清 0 使能**
OC3M[6:4]	**输出比较 3 模式**
OC3PE	**输出比较 3 预装载使能**
OC3FE	**输出比较 3 快速使能**
CC3S[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器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

TIM1 和 TIM8 捕获/比较模式寄存器 2(TIMx_CCMR2)			**基地址： 0x40012C00(TIM1),0x40013400(TIM8) 0x40015000(TIM20)** **偏移地址： 0x1C**					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	IC4F[15:12]				IC4PSC[11:10]		CC4S[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	IC3F[7:4]				IC3PSC[3:2]		CC3S[1:0]	
Reset:	0	0	0	0	0	0	0	0

输入捕获模式

位	功能描述
IC4F[15:12]	**输入捕获 4 滤波器**

IC4PSC[11:10]	**输入/捕获 4 预分频器**
CC4S[9:8]	**捕获/比较 4 选择 (Capture/Compare 4 selection)** 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。
IC3F[7:4]	**输入捕获 3 滤波器**
IC3PSC[3:2]	**输入/捕获 3 预分频器**
CC3S[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器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。

11.5.13TIM1 和 TIM8 和 TIM20 计数器(TIMx_CNT)

TIM1 和 TIM8 和 TIM20 计数器(TIMx_CNT)			基地址：0x40012C00(TIM1),0x40013400(TIM8) 0x40015000(TIM20) 偏移地址： 0x24					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	CNT							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	CNT							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CNT[15:0]	**计数器的值**

11.5.14TIM1 和 TIM8 和 TIM20 预分频器(TIMx_PSC)

TIM1 和 TIM8 和 TIM20 预分频器(TIMx_PSC)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)					
			偏移地址： 0x28					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	PSC[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PSC[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PSC[15:0]	**预分频器的值 (Prescaler value)** 计数器的时钟频率(CK_CNT)等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清’0’或被工作在复位模式的从控制器清’0’。

11.5.15TIM1 和 TIM8 和 TIM20 自动重装载寄存器(TIMx_ARR)

TIM1 和 TIM8 和 TIM20 自动重装载寄存器(TIMx_ARR)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)					
			偏移地址： 0x2C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read:	X	X	X	X	X	X	X	X

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ARR[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ARR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ARR	**自动重装载的值** (Prescaler value) ARR 包含了将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。

11.5.16TIM1 和 TIM8 和 TIM20 重复计数寄存器(TIMx_RCR)

TIM1 和 TIM8 和 TIM20 重复计数寄存器(TIMx_RCR)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x30					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	REP[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
REP[7:0]	**重复计数器的值** (Repetition counter value) 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从

	预装载寄存器 传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开 始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄 存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： — 在边沿对齐模式下，PWM 周期的数目； — 在中心对称模式下，PWM 半周期的数目；

11.5.17TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1(TIMx_CCR1)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1(TIMx_CCR1)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x34					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR1[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR1[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR1[15:0]	**捕获/比较通道 1 的值 (Capture/Compare 1 value)** 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄 存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信

	号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

11.5.18TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 2(TIMx_CCR2)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 2(TIMx_CCR2)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x38					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR2[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR2[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR2[15:0]	捕获/比较通道 2 的值 (Capture/Compare 2 value) 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄 存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 2 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC2 端口上产生输出信号。 若 CC2 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。

11.5.19TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 3(TIMx_CCR3)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 3(TIMx_CCR3)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x03C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR3[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR3[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR3[15:0]	**捕获/比较通道 3 的值 (Capture/Compare 3 value)** 若 CC3 通道配置为输出： CCR3 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄 存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 3 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC3 端口上产生输出信号。 若 CC3 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件(IC3)传输的计数器值。

11.5.20TIM1 和 TIM8 和 TIM20 捕获/比较寄存器(TIMx_CCR4)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器(TIMx_CCR4)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x40					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR4[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR4[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR4[15:0]	**捕获/比较通道 4 的值 (Capture/Compare 4 value)** 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值(预装载值)。 如果在 TIMx_CCMR4 寄存器(OC4PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 4 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC4 端口上产生输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件(IC4)传输的计数器值。

11.5.21TIM1 和 TIM8 和 TIM20 DMA 控制寄存器(TIMx_DCR)

TIM1 和 TIM8 和 TIM20 DMA 控制寄存器(TIMx_DCR)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x48					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	DBL[12:8]				
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0

Read: Write:	X	X	INDEX_RESET	DBA[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DBL[12:8]	DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的传送长度(当对 TIMx_DMAR 寄存器进行读或写时，定时器则 进行一次连续传送)，即：定义传输的次数，传输可以是半字(双字节)或字节： 00000：1 次传输 00001：2 次传输 00010：3 次传输 .... 10001：18 次传输 例：我们考虑这样的传输：DBL=7，DBA=TIM2_CR1 - 如果 DBL=7，DBA=TIM2_CR1 表示待传输数据的地址，那么传输的地址由下式给出： (TIMx_CR1 的地址) + DBA + (DMA 索引)，其中 DMA 索引 = DBL 其中(TIMx_CR1 的地址) + DBA 再加上 7，给出了将要写入或者读出数据的地址，这样数据的传 输将发生在从地址(TIMx_CR1 的地址) + DBA 开始的 7 个寄存器。 根据 DMA 数据长度的设置，可能发生以下情况： - 如果设置数据为半字(16 位)，那么数据就会传输给全部 7 个寄存器。 - 如果设置数据为字节，数据仍然会传输给全部 7 个寄存器：第一个寄存器包含第一个 MSB 字 节，第二个寄存器包含第一个 LSB 字节，以此类推。因此对于定时器，用户必须指定由 DMA 传 输的数据宽度。
INDEX_RESET	DMA 请求事件清零 DMAR index 控制位 0:DMA 请求事件不会清零 DMAR index 1:DMA 请求事件会清零 DMAR index 该位需要相应的 DMA 请求允许位使能才有意义，否则任何情况下 DMA 请求事件都不会清零 DMAR index
DBA[4:0]	DMA 基地址 (DMA base address) 这些位定义了 DMA 在连续模式下的基地址(当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义 为从 TIMx_CR1 寄存器所在地址开始的偏移量： 00000：TIMx_CR1， 00001：TIMx_CR2， 00010：TIMx_SMCR， .....

11.5.22TIM1 和 TIM8 和 TIM20 连续模式的 DMA 地址(TIMx_DMAR)

	基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20)

TIM1 和 TIM8 和 TIM20 连续模式的 DMA 地址(TIMx_DMAR)			偏移地址： 0x4C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DMAB[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DMAB[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DMAB[15:0]	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的存取操作： TIMx_CR1 地址 + DBA + DMA 索引，其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1)所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量，它取决于 TIMx_DCR 寄存器中定义的 DBL。

11.5.23TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 6(TIMx_CCR6)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器(TIMx_CCR6)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x0					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read:	CCR6[15:8]							

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR6[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR6[15:0]	**捕获/比较通道 6 的值 (Capture/Compare 5 value)** 若 CC6 通道配置为输出： CCR6 包含了装入当前捕获/比较 5 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC6PE 位)中未选择预装载特性，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 6 寄存器中。 当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC6 端口上产生输出信号。

11.5.24TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1N (TIMx_CCR1N)

TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 1N (TIMx_CCR1N)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x64					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	CCR1N_EN	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR1N[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR1N[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR1N_EN	**CCR1N 寄存器使能** 若 CC1 通道配置为输出： CCR1N_EN=1: CNT 向下数比较的寄存器为 CCR1N CCR1N_EN=0: CNT 向下数比较的寄存器为 CCR1 (默认状态)

CCR1N[15:0]	**捕获/比较通道 1N 的值 (Capture/Compare 1N value)** 若 CC1 通道配置为输出： CCR1N 包含了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 TIMx_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 CCR1N_EN 必須為 1,寄存器才有效果

11.5.25TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 2N (TIMx_CCR2N)

TIM1 和 TIM8 捕获/比较寄存器 2N (TIMx_CCR2N)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x68					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	CCR2N_EN	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR2N[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR2N[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR2N_EN	**CCR2N 寄存器使能** 若 CC2 通道配置为输出： CCR2N_EN=1: CNT 向下数比較的寄存器為 CCR2N CCR2N_EN=0: CNT 向下数比較的寄存器為 CCR2 (默认状态)
CCR2N[15:0]	**捕获/比较通道 2N 的值 (Capture/Compare 2N value)** 若 CC2 通道配置为输出： CCR2N 包含了装入当前捕获/比较 2 寄存器的值(预装载值)。 如果在 TIMx_CCMR2 寄存器(OC2PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 CCR2N_EN 必須為 1,寄存器才有效果

11.5.26TIM1 和 TIM8 和 TIM20 捕获/比较寄存器 3N (TIMx_CCR3N)

TIM1 和 TIM8 捕获/比较寄存器 3N (TIMx_CCR3N)			基地址： 0x4001 2C00(TIM1),0x4001 3400(TIM8) 0x4001 5000(TIM20) 偏移地址： 0x6C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	CCR3N_EN	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	CCR3N[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CCR3N[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CCR3N_EN	**CCR3N 寄存器使能** 若 CC3 通道配置为输出： CCR3N_EN=1: CNT 向下数比较的寄存器為 CCR3N CCR3N_EN=0: CNT 向下数比较的寄存器為 CCR3 (默认状态)
CCR3N[15:0]	**捕获/比较通道 3N 的值 (Capture/Compare 3N value)** 若 CC3 通道配置为输出： CCR3N 包含了装入当前捕获/比较 3 寄存器的值(预装载值)。 如果在 TIMx_CCMR3 寄存器(OC3PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。 CCR3N_EN 必須為 1,寄存器才有效果

12 I2C 接口

12.1 I2C 简介

I²C(芯片间)总线接口连接微控制器和串行 I²C 总线。它提供多主机功能，控制所有 I²C 总线特定的时序、协议、仲裁和定时。支持标准和快速两种模式，同时与 SMBus 2.0 兼容。

I²C 模块有多种用途，包括 CRC 码的生成和校验、SMBus(系统管理总线—System Management Bus)和 PMBus(电源管理总线—Power Management Bus)。

根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

12.2 I2C 主要特点

- 并行总线/I²C总线协议转换器
- 多主机功能：该模块既可做主设备也可做从设备
- I²C主设备功能
  - 产生时钟
  - 产生起始和停止信号
- I²C从设备功能
  - 可编程的I²C地址检测
  - 可响应2个从地址的双地址能力
  - 停止位检测
- 产生和检测7位/10位地址和广播呼叫
- 支持不同的通讯速度
  - 标准速度(高达100 kHz)
  - 快速(高达400 kHz)
- 状态标志：
  - 发送器/接收器模式标志
  - 字节发送结束标志
  - I²C总线忙标志
- 错误标志
  - 主模式时的仲裁丢失
  - 地址/数据传输后的应答(ACK)错误
  - 检测到错位的起始或停止条件
  - 禁止拉长时钟功能时的上溢或下溢
- 2个中断向量
  - 1个中断用于地址/数据通讯成功
  - 1个中断用于错误
- 可选的拉长时钟功能
- 具单字节缓冲器的DMA

- 可配置的PEC(信息包错误检测)的产生或校验：
  - 发送模式中PEC值可以作为最后一个字节传输
  - 用于最后一个接收字节的PEC错误校验
- 兼容SMBus 2.0
  - 25 ms时钟低超时延时
  - 10 ms主设备累积时钟低扩展时间
  - 25 ms从设备累积时钟低扩展时间
  - 带ACK控制的硬件PEC产生/校验
  - 支持地址分辨协议(ARP)
- 兼容SMBus

12.3 I2C 功能描述

I^2C 模块接收和发送数据，并将数据从串行转换成并行，或并行转换成串行。可以开启或禁止中断。接口通过数据引脚(SDA)和时钟引脚(SCL)连接到 I^2C 总线。允许连接到标准(高达 100kHz)或快速(高达 400kHz)的 I^2C 总线。

12.3.1 模式选择

接口可以下述 4 种模式中的一种运行：

- 从发送器模式
- 从接收器模式
- 主发送器模式
- 主接收器模式

该模块默认地工作于从模式。接口在生成起始条件后自动地从从模式切换到主模式；当仲裁丢失或产生停止信号时，则从主模式切换到从模式。允许多主机功能。

通信流

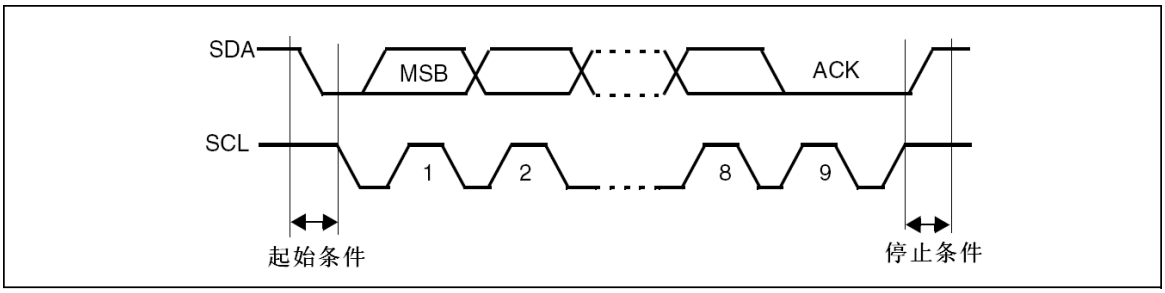
主模式时，I^2C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。起始条件和停止条件都是在主模式下由软件控制产生。

从模式时，I^2C 接口能识别它自己的地址(7 位或 10 位)和广播呼叫地址。软件能够控制开启或禁止广播呼叫地址的识别。

数据和地址按 8 位/字节进行传输，高位在前。跟在起始条件后的 1 或 2 个字节是地址(7 位模式为 1 个字节，10 位模式为 2 个字节)。地址只在主模式发送。

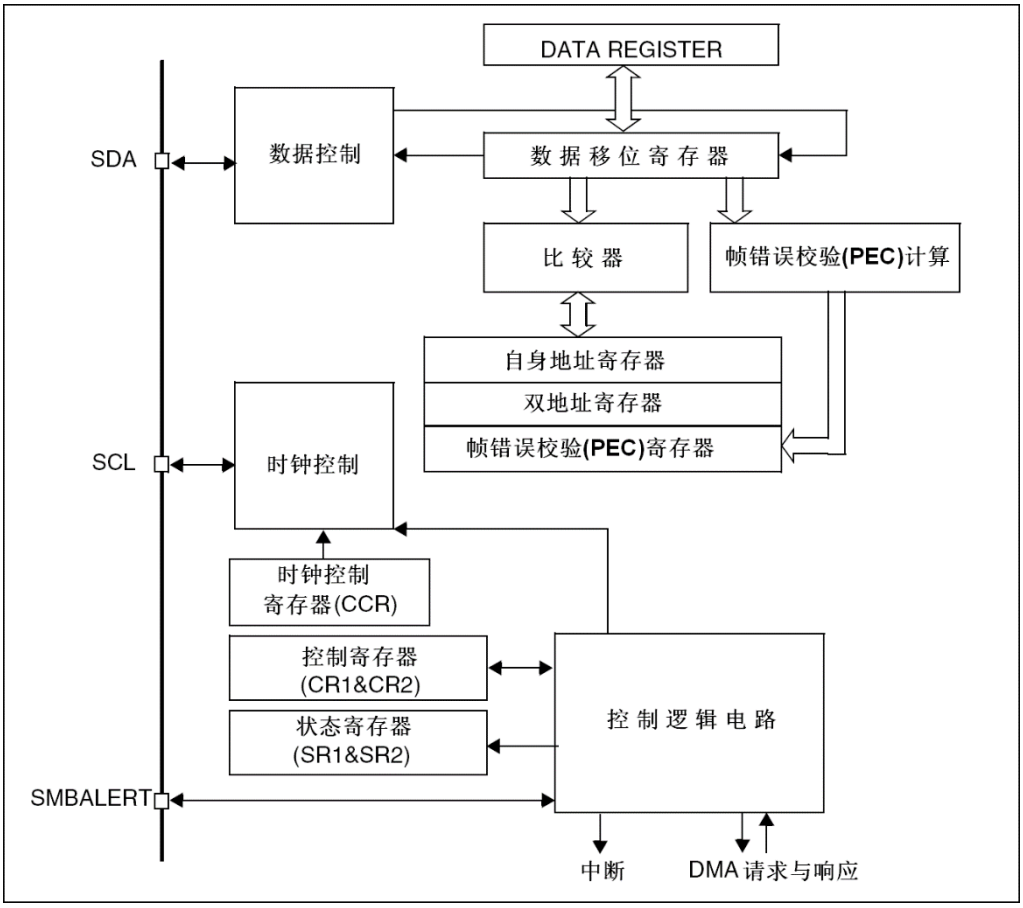
在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位(ACK)给发送器。参考下图

图 I^2C 总线协议



软件可以开启或禁止应答(ACK)，并可以设置 I²C 接口的地址(7 位、10 位地址或广播呼叫地址)。

I²C 接口的功能框图示于下图



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。

12.3.2 I2C 从模式

默认情况下，I²C 接口总是工作在从模式。从从模式切换到主模式，需要产生一个起始条件。

为了产生正确的时序，必须在 I2C_CR2 寄存器中设定该模块的输入时钟。输入时钟的频率必须至少是：

- 标准模式下为：2MHz
- 快速模式下为：4MHz

一旦检测到起始条件，在 SDA 线上接收到的地址被送到移位寄存器。然后与芯片自己的地址 OAR1 和 OAR2(当 ENDUAL=1)或者广播呼叫地址(如果 ENGC=1)相比较。

注：　在 10 位地址模式时，比较包括头段序列(11110xx0)，其中的 xx 是地址的两个最高有效位。

头段或地址不匹配：I²C 接口将其忽略并等待另一个起始条件。

头段匹配(仅 10 位模式)：如果 ACK 位被置'1'，I²C 接口产生一个应答脉冲并等待 8 位从地址。**地址匹配**：I²C 接口产生以下时序：

- 如果ACK被置'1'，则产生一个应答脉冲
- 硬件设置ADDR位；如果设置了ITEVFEN位，则产生一个中断
- 如果ENDUAL=1，软件必须读DUALF位，以确认响应了哪个从地址。

在 10 位模式，接收到地址序列后，从设备总是处于接收器模式。在收到与地址匹配的头序列并且最低位为'1'(即 11110xx1)后，当接收到重复的起始条件时，将进入发送器模式。

在从模式下 TRA 位指示当前是处于接收器模式还是发送器模式。

从发送器

在接收到地址和清除 ADDR 位后，从发送器将字节从 DR 寄存器经由内部移位寄存器发送到 SDA 线上。

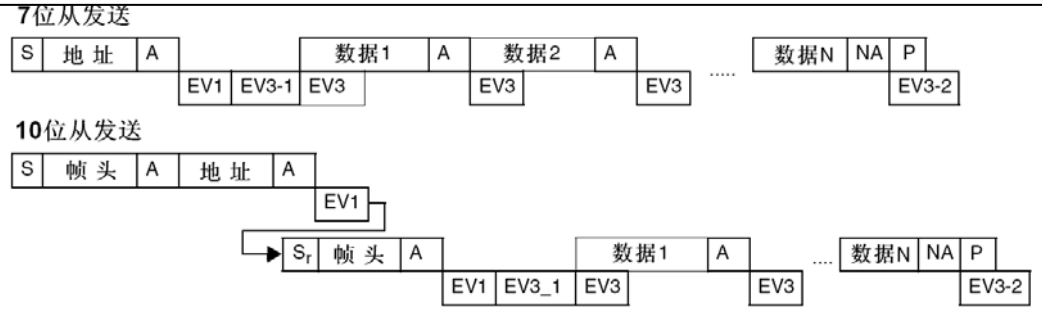
从设备保持 SCL 为低电平，直到 ADDR 位被清除并且待发送数据已写入 DR 寄存器。(见下图中的 EV1 和 EV3)。

当收到应答脉冲时:

TxE位被硬件置位，如果设置了ITEVFEN和ITBUFEN位，则产生一个中断。

如果 TxE 位被置位，但在下一个数据发送结束之前没有新数据写入到 I2C_DR 寄存器，则 BTF 位被置位，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。

图：从发送器的传送序列图



说明：S=Start(起始条件)，S_r=重复的起始条件，P=Stop(停止条件)，A=响应，NA=非响应，EVx=事件(ITEVFEN=1 时产生中断)

EV1：ADDR=1，读 SR1 然后读 SR2 将清除该事件。
EV3-1：TxE=1，移位寄存器空，数据寄存器空，写 DR。
EV3：TxE=1，移位寄存器非空，数据寄存器空，写 DR 将清除该事件。EV3-2：AF=1，在 SR1 寄存器的 AF 位写'0'可清除 AF 位。

注：1 – EV1 和 EV3_1 事件拉长 SCL 低的时间，直到对应的软件序列结束。2 – EV3 的软件序列必须在当前字节传输结束之前完成。

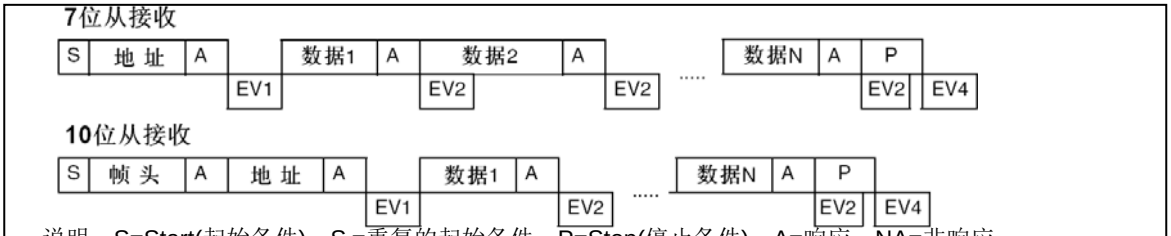
从接收器

在接收到地址并清除 ADDR 后，从接收器将通过内部移位寄存器从 SDA 线接收到的字节存进 DR 寄存器。I²C 接口在接收到每个字节后都执行下列操作:

- 如果设置了ACK位，则产生一个应答脉冲
- 硬件设置RxNE=1。如果设置了ITEVFEN和ITBUFEN位，则产生一个中断。

如果 RxNE 被置位，并且在接收新的数据结束之前 DR 寄存器未被读出，BTF 位被置位，在清除 BTF 之前 I²C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。(见下图)。

图：从接收器的传送序列图



说明：S=Start(起始条件)，S_r=重复的起始条件，P=Stop(停止条件)，A=响应，NA=非响应，EVx=事件(ITEVFEN=1 时产生中断)

EV1：ADDR=1，读 SR1 然后读 SR2 将清除该事件。
EV2：RxNE=1，读 DR 将清除该事件。
EV4：STOPF=1，读 SR1 然后写 CR1 寄存器将清除该事件。

注：1 – EV1 事件拉长 SCL 低的时间，直到对应的软件序列结束。2 – EV2 的软件序列必须在当前字节传输结束之前完成。

关闭从通信

在传输完最后一个数据字节后，主设备产生一个停止条件， I²C 接口检测到这一条件时：

- 设置STOPF=1，如果设置了ITEVFEN位，则产生

一个中断。 然后I²C接口等待读SR1寄存器，再写CR1寄存器。(见上图的EV4)。

12.3.3 I2C 主模式

在主模式时，I²C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。当通过 START 位在总线上产生了起始条件，设备就进入了主模式。

以下是主模式所要求的操作顺序：

- 在I2C_CR2寄存器中设定该模块的输入时钟以产生正确的时序
- 配置时钟控制寄存器
- 配置上升时间寄存器
- 编程I2C_CR1寄存器启动外设
- 置I2C_CR1寄存器中的START位为1，产生起始条件

I²C 模块的输入时钟频率必须至少是:

- 标准模式下为：2MHz
- 快速模式下为：4MHz

起始条件

当 BUSY=0 时，设置 START=1，I²C 接口将产生一个开始条件并切换至主模式(M/SL 位置位)。

注：在主模式下，设置 START 位将在当前字节传输完后由硬件产生一个重开始条件。

一旦发出开始条件:

- SB位被硬件置位，如果设置了ITEVFEN位，则会产生一个中断。

然后主设备等待读 SR1 寄存器，紧跟着将从地址写入 DR 寄存器(见图 245 和图 246 的 EV5)。

从地址的发送

从地址通过内部移位寄存器被送到 SDA 线上。

- 在10位地址模式时，发送一个头段序列产生以下事件：
  - ADD10位被硬件置位，如果设置了ITEVFEN位，则产生一个中断。
    然后主设备等待读 SR1 寄存器，再将第二个地址字节写入 DR 寄存器。
    ADDR 位被硬件置位，如果设置了 ITEVFEN 位，则产生一个中断
    随后主设备等待一次读SR1寄存器，跟着读SR2寄存器。
- 在7位地址模式时，只需送出一个地址字节。一旦该地址字节被送出，
  - ADDR位被硬件置位，如果设置了ITEVFEN位，则产生一个中断。
    随后主设备等待一次读 SR1 寄存器，跟着读 SR2 寄存器。根

据送出从地址的最低位，主设备决定进入发送器模式还是进入接收器模式。

- 在7位地址模式时，
  - 要进入发送器模式，主设备发送从地址时置最低位为'0'。
  - 要进入接收器模式，主设备发送从地址时置最低位为'1'。
- 在10位地址模式时
  - 要进入发送器模式，主设备先送头字节(11110xx0)，然后送最低位为'0'的从地址。
  - (这里xx代表10位地址中的最高2位。)
  - 要进入接收器模式，主设备先送头字节(11110xx0)，然后送最低位为'1'的从地址。然后再重新发送一个开始条件，后面跟着头字节(11110xx1)(这里xx代表10位地址中的最高2位。)

TRA 位指示主设备是在接收器模式还是发送器模式。

主发送器

在发送了地址和清除了 ADDR 位后，主设备通过内部移位寄存器将字节从 DR 寄存器发送到 SDA 线上。

主设备等待，直到 TxE 被清除。当收到应答脉冲时:

- TxE位被硬件置位，如果设置了INEVFEN和ITBUFEN位，则产生一个中断。

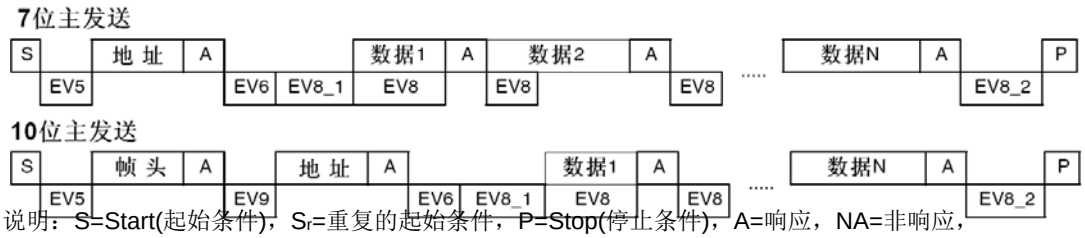
如果 TxE 被置位并且在上一次数据发送结束之前没有写新的数据字节到 DR 寄存器，则 BTF 被硬件置位，在清除 BTF 之前 I^2C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再写入 I2C_DR 寄存器将清除 BTF 位。

关闭通信

在 DR 寄存器中写入最后一个字节后，通过设置 STOP 位产生一个停止条件(见图 245 的 EV8_2)， 然后 I^2C 接口将自动回到从模式(M/S 位清除)。

注：当 TxE 或 BTF 位置位时，停止条件应安排在出现 EV8_2 事件时。

图：主发送器传送序列图



说明：S=Start(起始条件)，S_r=重复的起始条件，P=Stop(停止条件)，A=响应，NA=非响应，EVx=事件(ITEVFEN=1 时产生中断)。

EV5：SB=1，读 SR1 然后将地址写入 DR 寄存器将清除该事件。
EV6：ADDR=1，读 SR1 然后读 SR2 将清除该事件。
EV8_1：TxE=1，移位寄存器空，数据寄存器空，写 DR 寄存器。
EV8：TxE=1，移位寄存器非空，数据寄存器空，写入 DR 寄存器将清除该事件。
EV8_2：TxE=1，BTF=1，请求设置停止位。TxE 和 BTF 位由硬件在产生停止条件时清除。
EV9：ADDR10=1，读 SR1 然后写入 DR 寄存器将清除该事件。

注：1 – 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。2 – EV8 的软件序列必须在当前字节传输结束之前完成。

主接收器

在发送地址和清除 ADDR 之后，I^2C 接口进入主接收器模式。在此模式下，I^2C 接口从 SDA 线接收数据字节，并通过内部移位寄存器送至 DR 寄存器。在每个字节后，I^2C 接口依次执行以下操作：

- 如果ACK位被置位，发出一个应答脉冲。
- 硬件设置RxNE=1，如果设置了INEVFEN和ITBUFEN位，则会产生一个中断(见图246的EV7)。

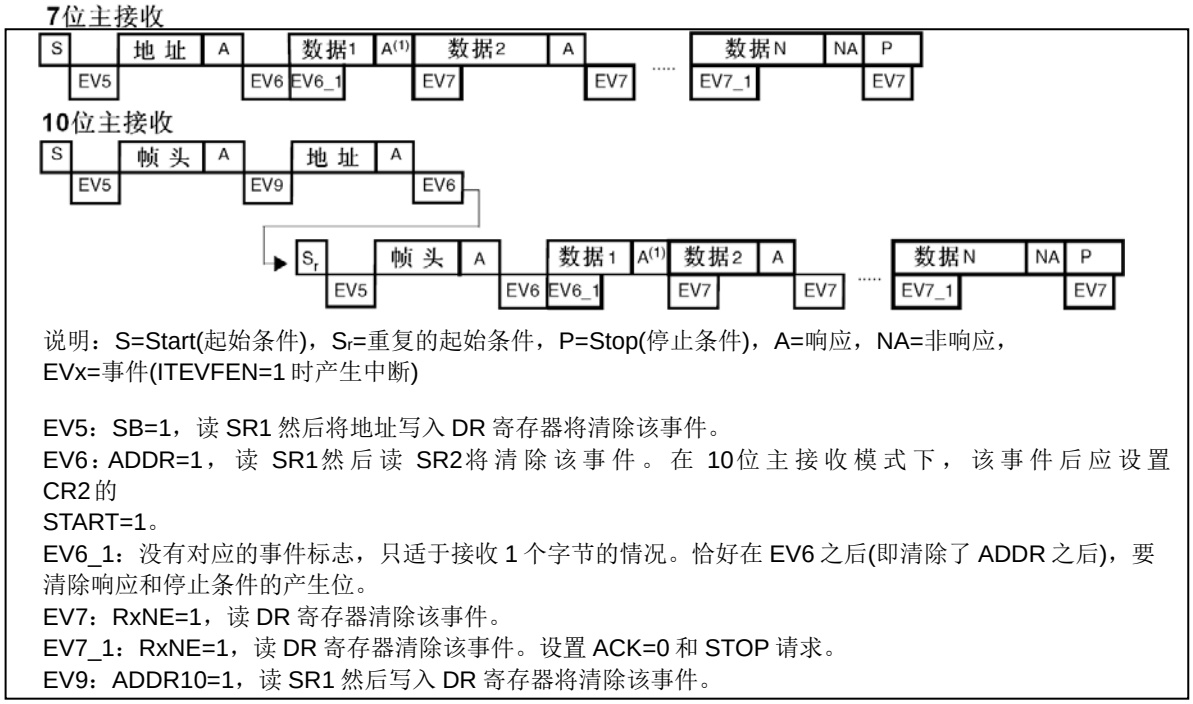
如果 RxNE 位被置位，并且在接收新数据结束前，DR 寄存器中的数据没有被读走，硬件将设置BTF=1，在清除 BTF 之前 I^2C 接口将保持 SCL 为低电平；读出 I2C_SR1 之后再读出 I2C_DR 寄存器将清除 BTF 位。

关闭通信

主设备在从从设备接收到最后一个字节后发送一个 NACK。接收到 NACK 后，从设备释放对 SCL 和 SDA 线的控制；主设备就可以发送一个停止/重起始条件。

- 为了在收到最后一个字节后产生一个NACK脉冲，在读倒数第二个数据字节之后(在倒数第二个RxNE事件之后)必须清除ACK位。
- 为了产生一个停止/重起始条件，软件必须在读倒数第二个数据字节之后(在倒数第二个RxNE 事件之后)设置 STOP/START 位。
- 只接收一个字节时，刚好在EV6之后(EV6_1时，清除ADDR之后)要关闭应答和停止条件的产生位。

在产生了停止条件后，I²C 接口自动回到从模式
(M/SL 位被清除)。图：主接收器传送序列图



1. 如果收到一个单独的字节，则是 NA。
2. EV5、EV6 和 EV9 事件拉长 SCL 低电平，直到对应的软件序列结束。
3. EV7 的软件序列必须在当前字节传输结束前完成。
4. EV6_1 或 EV7_1 的软件序列 必须在当前传输字节的 ACK 脉冲之前完成。

12.3.4 错误条件

以下条件可能造成通讯失败。

总线错误(BERR)

在一个地址或数据字节传输期间，当 I²C 接口检测到一个外部的停止或起始条件则产生总线错误。此时：

- BERR位被置位为'1'；如果设置了ITERREN位，则产生一个中断；
- 在从模式情况下，数据被丢弃，硬件释放总线：
  - 如果是错误的开始条件，从设备认为是一个重启动，并等待地址或停止条件。
  - 如果是错误的停止条件，从设备按正常的停止条件操作，同时硬件释放总线。
- 在主模式情况下，硬件不释放总线，同时不影响当前的传输状态。此时由软件决定是否要中止当前的传输。

应答错误(AF)

当接口检测到一个无应答位时，产生应答错误。此时：

- AF位被置位，如果设置了ITERREN位，则产生一个中断；
- 当发送器接收到一个NACK时，必须复位通讯：
  - 如果是处于从模式，硬件释放总线。
  - 如果是处于主模式，软件必须生成一个停止条件。

仲裁丢失(ARLO)

当 I²C 接口检测到仲裁丢失时产生仲裁丢失错误，此时：

- ARLO位被硬件置位，如果设置了ITERREN位，则产生一个中断；
- I²C接口自动回到从模式(M/SL位被清除)。当I²C接口丢失了仲裁，则它无法在同一个传输中响应它的从地址，但它可以在赢得总线的主设备发送重起始条件之后响应；
- 硬件释放总线。

过载/欠载错误(OVR)

在从模式下，如果禁止时钟延长， I²C 接口正在接收数据时，当它已经接收到一个字节(RxNE=1)，但在 DR 寄存器中前一个字节数据还没有被读出，则发生过载错误。此时：

- 最后接收的数据被丢弃；
- 在过载错误时，软件应清除RxNE位，发送器应该重新发送最后一次发送的字节。

在从模式下，如果禁止时钟延长，I²C 接口正在发送数据时，在下一个字节的时钟到达之前，新的数据还未写入 DR 寄存器(TxE=1)，则发生欠载错误。此时：

- 在DR寄存器中的前一个字节将被重复发出；
- 用户应该确定在发生欠载错时，接收端应丢弃重复接收到的数据。发送端应按I²C总线标准在规定的时间更新DR寄存器。

在发送第一个字节时，必须在清除 ADDR 之后并且第一个 SCL 上升沿之前写入 DR 寄存器；如果不能做到这点，则接收方应该丢弃第一个数据。

12.3.5 SDA/SCL 线控制

- 如果允许时钟延长：
  - 发送器模式：如果TxE=1且BTF=1：I²C接口在传输前保持时钟线为低，以等待软件读取SR1，然后把数据写进数据寄存器(缓冲器和移位寄存器都是空的)。
  - 接收器模式：如果RxNE=1且BTF=1：I²C接口在接收到数据字节后保持时钟线为低，以等待软件读SR1，然后读数据寄存器DR(缓冲器和移位寄存器都是满的)。
- 如果在从模式中禁止时钟延长：
  - 如果RxNE=1，在接收到下个字节前DR还没有被读出，则发生过载错。接收到的最后一个字节丢失。
  - 如果TxE=1，在必须发送下个字节之前却没有新数据写进DR，则发生欠载错。相同的字节将被重复发出。
  - 不控制重复写冲突。

12.3.6 SMBus

介绍

系统管理总线(SMBus)是一个双线接口。通过它，各设备之间以及设备与系统的其他部分之间可以互相通信。它基于 I^2C 操作原理。SMBus 为系统和电源管理相关的任务提供一条控制总线。一个系统利用 SMBus 可以和多个设备互传信息，而不需使用独立的控制线路。

系统管理总线(SMBus)标准涉及三类设备。从设备：接收或响应命令的设备。主设备：用来发送命令、产生时钟和终止发送的设备。主机：一种专用的主设备，它提供与系统 CPU 的主接口。主机必须具有主-从机功能并且必须支持 SMBus 提醒协议。一个系统里只允许有一个主机。

SMBus 和 I^2C 之间的相似点

- 2条线的总线协议(1个时钟，1个数据) + 可选的SMBus提醒线；
- 主-从通信，主设备提供时钟；
- 多主机功能
- SMBus数据格式类似于I^2C的7位地址格式(见图241)；

SMBus和I^2C之间的不同点

下表列出了 SMBus 和

I^2C 的不同点。表

SMBus	I^2C
最大传输速度100kHz	最大传输速度400kHz
最小传输速度10kHz	无最小传输速度
35ms时钟低超时	无时钟超时
固定的逻辑电平	逻辑电平由VDD决定
不同的地址类型(保留的、动态的等)	7位、10位和广播呼叫从地址类型
不同的总线协议(快速命令、处理呼叫等)	无总线协议

SMBus 与 I^2C 的比较

SMBus 应用用途

利用系统管理总线，设备可提供制造商信息，告诉系统它的型号/部件号，保存暂停事件的状态，报告不同类型的错误，接收控制参数，和返回它的状态。SMBus 为系统和电源管理相关的任务提供控制总线。

设备标识

在系统管理总线上，任何一个作为从模式的设备都有一个唯一的地址，叫做从地址。保留的从地址表请参考 2.0 版的 SMBus 规范

总线协议

SMBus 技术规范支持 9 个总线协议。有关这些协议的详细资料和 SMBus 地址类型，请参考 2.0 版的 SMBus 规范。这些协议由用户的软件来执行。

地址解析协议(ARP)

通过给每个从设备动态地分配一个新的唯一地址，可以解决 SMBus 的从地址冲突。地址解析协议(ARP)具有以下的特性：

- 使用标准SMBus物理层仲裁机制分配地址；
- 当设备维持供电期间，分配的地址仍保持不变，也允许设备在断电后保留其地址。
- 在地址分配后，没有额外的SMBus的打包开销(也就是说访问分配地址的设备与访问固定地址的设备所用时间是一样的)；
- 任何一个SMBus主设备可以遍历总线。

唯一的设备标识符(UDID)

为了分配地址，需要一种区分每个设备的机制，每个设备必须拥有一个唯一的设备标识符。

关于在 ARP 上 128 位的 UDID 的详细信息，参考 2.0 版的 SMBus 规范。

SMBus 提醒模式

SMBus 提醒是一个带中断线的可选信号，用于那些希望扩展它们的控制能力而牺牲一个引脚的设备。SMBALERT 和 SCL、SDA 信号一样，是一种线与信号。SMBALERT 通常和 SMBus 广播呼叫地址一起使用。与 SMBus 有关的消息为 2 字节。

一个只具有从功能的设备，可以通过设置 I2C_CR1 寄存器上的 ALERT 位，使用 SMBALERT 给主机发信号表示它希望进行通信。主机处理该中断并通过提醒响应地址 ARA(*Alert Response Address*，地址值为 0001100x)访问所有 SMBALERT 设备。只有那些将 SMBALERT 拉低的设备能应答 ARA。此状态是由 I2C_SR1 寄存器中的 SMBALERT 状态标记来标识的。主机执行一个修改过的接收字节操作。由从发送设备提供的 7 位设备地址被放在字节的 7 个最高位上，第八个位可以是'0'或'1'。

如果多个设备把 SMBALERT 拉低，最高优先级设备(最小的地址)将在地址传输期间通过标准仲裁赢得通信权。在确认从地址后，此设备不得再拉低它的 SMBALERT，如果当信息传输完成后，主机仍看到 SMBALERT 低，就知道需要再次读 ARA。

没有实现 SMBALERT 信号的主机可以定期访问 ARA。

有关 SMBus 提醒模式的更多详细资料，请参考 2.0 版的 SMBus 规范。

超时错误

在定时规范上 I^2C 和 SMBus 之间有很多差别。

SMBus 定义了一个时钟低超时，35ms 的超时。SMBus 规定 TLOW:SEXT 为从设备的累积时钟低扩展时间。SMBus 规定 TLOW:MEXT 为主设备的累积时钟低扩展时间。更多超时细节请参考 2.0 版的 SMBus 规范。

I2C_SR1 中的状态标志 Timeout 或 Tlow 错误表明了这个特性的状态。

如何使用 SMBus 模式的接口

为了从 I^2C 模式切换到 SMBus 模式，应该执行下列步骤：

- 设置I2C_CR1寄存器中的SMBus位；
- 按应用要求配置I2C_CR1寄存器中的SMBTYPE和ENARP位。

I2C 从模式。

软件程序必须处理多种 SMBus 协议。

- 如果ENARP=1且SMBTYPE=0，使用SMB设备默认地址。
- 如果ENARP=1且SMBTYPE=1，使用SMB主设备头字段。
- 如果SMBALERT=1，使用SMB提醒响应地址。

12.4 DMA 请求

DMA 请求(当被使能时)仅用于数据传输。发送时数据寄存器变空或接收时数据寄存器变满，则产生 DMA 请求。DMA 请求必须在当前字节传输结束之前被响应。当为相应 DMA 通道设置的数据传输量已经完成时，DMA 控制器发送传输结束信号 ETO 到 I^2C 接口，并且在中断允许时产生一个传输完成中断：

- 主发送器：在EOT中断服务程序中，需禁止DMA请求，然后在等到BTF事件后设置停止条件。
- 主接收器：当要接收的数据数目大于或等于2时，DMA控制器发送一个硬件信号EOT_1，它对应DMA传输(字节数－1)。如果在I2C_CR2寄存器中设置了LAST位，硬件在发送完
  EOT_1 后的下一个字节，将自动发送 NACK。在中断允许的情况下，用户可以在 DMA 传输完成的中断服务程序中产生一个停止条件

利用 DMA 发送

通过设置 I2C_CR2 寄存器中的 DMAEN 位可以激活 DMA 模式。只要 TxE 位被置位，

数据将由 DMA 从预置的存储区装载进 I2C_DR 寄存器。为 I^2C 分配一个 DMA 通道，

须执行以下步骤(x 是通道号)：

1. 在DMA_CPARx寄存器中设置I2C_DR寄存器地址。数据将在每个TxE事件后从存储器传送至这个地址。
2. 在DMA_CMARx寄存器中设置存储器地址。数据在每个TxE事件后从这个存储区传送至I2C_DR。
3. 在DMA_CNDTRx寄存器中设置所需的传输字节数。在每个TxE事件后，此值将被递减。
4. 利用DMA_CCRx寄存器中的PL[0:1]位配置通道优先级。
5. 设置DMA_CCRx寄存器中的DIR位，并根据应用要求可以配置在整个传输完成一半或全部完成时发出中断请求。
6. 通过设置DMA_CCTx寄存器上的EN位激活通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I^2C 接口发送一个传输结束的

EOT/ EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：　如果使用 DMA 进行发送时，不要设置 I2C_CR2 寄存器的 ITBUFEN 位。

利用 DMA 接收

通过设置 I2C_CR2 寄存器中的 DMAEN 位可以激活 DMA 接收模式。每次接收到数据字节时，将由 DMA 把 I2C_DR 寄存器的数据传送到设置的存储区(参考 DMA 说明)。设置 DMA 通道进行 I^2C 接收，须执行以下步骤(x 是通道号)：

1. 在DMA_CPARx寄存器中设置I2C_DR寄存器的地址。数据将在每次RxNE事件后从此地址传送到存储区。
2. 在DMA_CMARx寄存器中设置存储区地址。数据将在每次RxNE事件后从I2C_DR寄存器传送到此存储区。
3. 在DMA_CNDTRx寄存器中设置所需的传输字节数。在每个RxNE事件后，此值将被递减。
4. 用DMA_CCRx寄存器中的PL[0:1]配置通道优先级。

5. 清除DMA_CCRx寄存器中的DIR位，根据应用要求可以设置在数据传输完成一半或全部完成时发出中断请求。
6. 设置DMA_CCRx寄存器中的EN位激活该通道。

当 DMA 控制器中设置的数据传输数目已经完成时，DMA 控制器给 I^2C 接口发送一个传输结束的

EOT/ EOT_1 信号。在中断允许的情况下，将产生一个 DMA 中断。

注：　如果使用 DMA 进行接收时，不要设置 I2C_CR2 寄存器的 ITBUFEN 位。

12.5 包错误校验(PEC)

包错误校验(PEC)计算器是用于提高通信的可靠性，这个计算器使用下述 CRC-8 多项式对每一位串行数据进行计算：

$$C(x) = x^8 + x^2 + x + 1$$

- PEC计算由I2C_CR1寄存器的ENPEC位激活。PEC使用CRC-8算法对所有信息字节进行计算，包括地址和读/写位在内。
  - 在发送时：在最后一个TxE事件时设置I2C_CR1寄存器的PEC传输位，PEC将在最后一个字节后被发送。
  - 在接收时：在最后一个RxNE事件之后设置I2C_CR1寄存器的PEC位，如果下个接收到的字节不等于内部计算的PEC，接收器发送一个NACK。如果是主接收器，不管校对的结果如何， PEC后都将发送NACK。PEC位必须在接收当前字节的ACK脉冲之前设置。
- 在I2C_SR1寄存器中可获得PECERR错误标记/中断。
- 如果DMA和PEC计算器都被激活：

– 在发送时：当I^2C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。
– 在接收时：当I^2C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，并且将检查它。在接收到PEC后产生一个DMA请求。

●为了允许中间PEC传输，在I2C_CR2寄存器中有一个控制位(LAST位)用于判别是否真是最后

一个DMA传输。如果确实是最后一个主接收器的DMA请求，在接收到最后一个字节后自动发送NACK。

●仲裁丢失时PEC计算失效。

12.6 I2C 中断请求

下表列出了所有

的 I^2C 中断请求

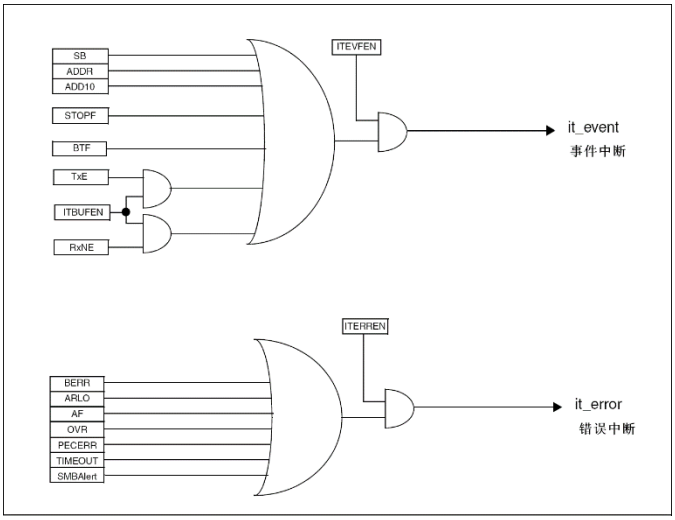
表：

中断事件	事件标志	开启控制位
起始位已发送(主)	SB	ITEVFEN
地址已发送(主) 或 地址匹配(从)	ADDR	
10位头段已发送(主)	ADD10	
已收到停止(从)	STOPF	
数据字节传输完成	BTF	
接收缓冲区非空	RxNE	ITEVFEN 和 ITBUFEN
发送缓冲区空	TxE	
总线错误	BERR	ITERREN
仲裁丢失(主)	ARLO	
响应失败	AF	
过载/欠载	OVR	
PEC错误	PECERR	
超时/Tlow错误	TIMEOUT	
SMBus提醒	SMBALERT	

注： 1. SB、ADDR、ADD10、STOPF、BTF、RxNE 和 TxE 通过逻辑或汇到同一个中断通道中。

2. BERR、ARLO、AF、OVR、PECERR、TIMEOUT 和 SMBALERT 通过逻辑汇到同一个中断通道

I2C 通道中断映射图



12.7 I2C 相关寄存器

12.7.1 控制寄存器 1(I2C_CR1)

控制寄存器 1(I2C_CR1)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2) 偏移地址：0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	SWRST	X	ALERT	PEC	POS	ACK	STOP	START
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	X	SMBUS	PE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SWRST	**软件复位 (Software reset)** 当被置位时，I²C处于复位状态。在复位该位前确信I²C的引脚被释放，总线是空的。 0：I²C模块不处于复位状态； 1：I²C模块处于复位状态。 注：该位可以用于 BUSY 位为’1’，在总线上又没有检测到停止条件时。
ALERT	**SMBus提醒 (SMBus alert)** 软件可以设置或清除该位；当PE=0时，由硬件清除。 0：释放SMBAlert引脚使其变高。提醒响应地址头紧跟在NACK信号后面； 1：驱动SMBAlert引脚使其变低。提醒响应地址头紧跟在ACK信号后面。
PEC	**数据包出错检测 (Packet error checking)** 软件可以设置或清除该位；当传送PEC后，或起始或停止条件时，或当PE=0时硬件将其清除。 0：无PEC传输； 1：PEC传输(在发送或接收模式)。注：仲裁丢失时，PEC的计算失效。
POS	**应答/PEC位置(用于数据接收) (Acknowledge/PEC Position (for data reception))** 软件可以设置或清除该位，或当PE=0时，由硬件清除。 0：ACK位控制当前移位寄存器内正在接收的字节的(N)ACK。 PEC位表明当前移位寄存器内的字节是PEC； 1：ACK位控制在移位寄存器里接收的下一个字节的(N)ACK。PEC位表明在移位寄存器里接收的下一个字节是PEC。

	注：POS位只能用在2字节的接收配置中，必须在接收数据之前配置。为了NACK第2个字节，必须在清除ADDR为之后清除ACK位。 为了检测第2个字节的PEC，必须在配置了POS位之后，拉伸ADDR事件时设置PEC位。
ACK	**应答使能（Acknowledge enable）** 软件可以设置或清除该位，或当PE=0时，由硬件清除。 0：无应答返回； 1：在接收到一个字节后返回一个应答(匹配的地址或数据)。
STOP	**停止条件产生（Stop generation）** 软件可以设置或清除该位；或当检测到停止条件时，由硬件清除；当检测到超时错误时，硬件将其置位。 在主模式下： 0：无停止条件产生； 1：在当前字节传输或在当前起始条件发出后产生停止条件。在从模式下： 0：无停止条件产生； 1：在当前字节传输或释放SCL和SDA线。 注：当设置了STOP、START 或PEC 位，在硬件清除这个位之前，软件不要执行任何对I2C_CR1的写操作；否则有可能会第2次设置STOP、START或PEC位。
START	**起始条件产生（Start generation）** 软件可以设置或清除该位，或当起始条件发出后或PE=0时，由硬件清除。 在主模式下： 0: 无起始条件产生； 1：重复产生起始条件。 在从模式下： 0：无起始条件产生； 1：当总线空闲时，产生起始条件。
NOSTRETCH	**禁止时钟延长(从模式)（Clock stretching disable (Slave mode)）** 该位用于当ADDR或BTF标志被置位，在从模式下禁止时钟延长，直到它被软件复位。 0：允许时钟延长； 1：禁止时钟延长。
ENGC	**广播呼叫使能（General call enable）** 0：禁止广播呼叫。以非应答响应地址00h； 1：允许广播呼叫. 以应答响应地址00h。
ENPEC	**PEC使能（PEC enable）** 0：禁止PEC计算； 1：开启PEC计算。
ENARP	**ARP使能（ARP enable）** 0：禁止ARP； 1：使能ARP。 如果SMBTYPE=0，使用SMBus设备的默认地址。如果SMBTYPE=1，使用SMBus的主地址。
SMBTYPE	**SMBus类型（SMBus type）** 0：SMBus设备； 1：SMBus主机。
SMBUS	**SMBus模式（SMBus mode）** 0：I2C模式； 1：SMBus模式。
PE	**I2C模块使能（Peripheral enable）** 0：禁用I2C模块； 1：启用I2C模块：根据SMBus位的设置，相应的I/O口需配置为复用功能。

	注：如果清除该位时通讯正在进行，在当前通讯结束后，I2C模块被禁用并返回空闲状态。由于在通讯结束后发生PE＝0，所有的位被清除。 在主模式下，通讯结束之前，绝不能清除该位。

12.7.2 控制寄存器 2(I2C_CR2)

控制寄存器 2(I2C_CR2)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2) 偏移地址：0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	FREQ[5:0]					
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LAST	DMA最后一次传输 (DMA last transfer) 0：下一次DMA的EOT不是最后的传输； 1：下一次DMA的EOT是最后的传输。 注：该位在主接收模式使用， 使得在最后一次接收数据时可以产生一个 NACK。
DMAEN	DMA请求使能 (DMA requests enable) 0：禁止DMA请求； 1：当 TxE=1 或 RxNE =1 时，允许 DMA 请求。
ITBUFEN	缓冲器中断使能 (Buffer interrupt enable) 0：当TxE=1或RxNE=1时，不产生任何中断； 1：当 TxE=1 或 RxNE=1 时，产生事件中断(不管 DMAEN 是何种状态)。
ITEVTEN	事件中断使能 (Event interrupt enable) 0：禁止事件中断； 1：允许事件中断。 在下列条件下，将产生该中断： – SB = 1 (主模式)； – ADDR = 1 (主/从模式)； – ADD10= 1 (主模式)；

	– STOPF = 1 (从模式)； – BTF = 1，但是没有TxE或RxNE事件； – 如果ITBUFEN = 1，TxE事件为1； – 如果ITBUFEN = 1，RxNE事件为1。
ITERREN	**出错中断使能 (Error interrupt enable)** 0：禁止出错中断； 1：允许出错中断。 在下列条件下，将产生该中断： – BERR = 1； – ARLO = 1； – AF = 1； – OVR = 1； – PECERR = 1； – TIMEOUT = 1； – SMBAlert = 1。
FREQ[5:0]	**I2C模块时钟频率 (Peripheral clock frequency)** 必须设置正确的输入时钟频率以产生正确的时序，允许的范围在2～36MHz之间： 000000：禁用 000001：禁用 000010：2MHz ... 100100：36MHz 大于100100：禁用。

12.7.3 自身地址寄存器 1(I2C_OAR1)

自身地址寄存器 1(I2C_OAR1)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2) 偏移地址：0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	ADDMODE	X	X	X	X	X	ADD	
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ADD							ADD0

Reset:	0	0	0	0	0	0	0	0

位	功能描述
ADDMODE	**寻址模式(从模式) (Addressing mode (slave mode))** 0：7位从地址(不响应10位地址)； 1：10 位从地址(不响应 7 位地址)。
ADD[9:8]	**接口地址 (Interface address)** 7位地址模式时不用关心。 10 位地址模式时为地址的 9~8 位。
ADD[7:1]	**接口地址 (Interface address)** 地址的 7~1 位。
ADD0	**接口地址 (Interface address)** 7位地址模式时不用关心。 10位地址模式时为地址第0位。

注：位 14 必须由软件强制为“1”

12.7.4 自身地址寄存器 2(I2C_OAR2)

自身地址寄存器 2(I2C_OAR2)			**基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2)**					
			偏移地址：0x0C					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	ADD2[7:1]							ENDUAL
Reset:	0	0	0	0	0	0	0	0

位	功能描述
ADD2[7:1]	**接口地址 (Interface address)** 在双地址模式下地址的 7~1 位。
ENDUAL	**双地址模式使能位 (Dual addressing mode enable)** 0：在7位地址模式下，只有OAR1被识别； 1：在 7 位地址模式下，OAR1 和 OAR2 都被识别。

12.7.5 数据寄存器(I2C_DR)

数据寄存器(I2C_DR)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2)					
			偏移地址：0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DR[7:0]	**8位数据寄存器 (8-bit data register)** 用于存放接收到的数据或放置用于发送到总线的数据 发送器模式：当写一个字节至DR寄存器时，自动启动数据传输。一旦传输开始(TxE=1)，如果能及时把下一个需传输的数据写入DR寄存器，I^2C模块将保持连续的数据流。 接收器模式：接收到的字节被拷贝到DR寄存器(RxNE=1)。在接收到下一个字节(RxNE=1)之前读出数据寄存器，即可实现连续的数据传送。 注：在从模式下，地址不会被拷贝进数据寄存器DR； 注：硬件不管理写冲突(如果TxE=0，仍能写入数据寄存器)； 注：如果在处理 ACK 脉冲时发生 ARLO 事件，接收到的字节不会被拷贝到数据寄存器里，因此不能读到它。

12.7.6 状态寄存器 1(I2C_SR1)

状态寄存器 1(I2C_SR1)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2)					
			偏移地址：0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read:	X	X	X	X	X	X	X	X

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	SMBALERT	TIMEOUT	X	PECERR	OVR	AF	ARL0	BERR
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	TxE	RxNE	X	STOPF	ADD10	BTF	ADDR	SB
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SMBALERT	**SMBus提醒 (SMBus alert)** 在SMBus主机模式下： 0：无SMBus提醒； 1：在引脚上产生SMBAlert提醒事件。 在SMBus从机模式下： 0：没有SMBAlert响应地址头序列； 1：收到SMBAlert响应地址头序列至SMBAlert变低。 - 该位由软件写’0’清除，或在 PE=0 时由硬件清除。
TIMEOUT	**超时或Tlow错误 (Timeout or Tlow error)** 0：无超时错误； 1：SCL 处于低已达到 25ms(超时)；或者主机低电平累积时钟扩展时间超过10ms(Tlow:mext)；或从设备低电平累积时钟扩展时间超过25ms(Tlow:sext)。 – 当在从模式下设置该位：从设备复位通讯，硬件释放总线。 – 当在主模式下设置该位：硬件发出停止条件。 该位由软件写’0’清除，或在 PE=0 时由硬件清除。
PECERR	**在接收时发生PEC错误 (PEC Error in reception)** 0：无PEC错误：接收到PEC后接收器返回ACK(如果ACK=1)； 1：有PEC错误：接收到PEC后接收器返回NACK(不管ACK是什么值)。 - 该位由软件写’0’清除，或在 PE=0 时由硬件清 除。
OVR	**过载/欠载 (Overrun/Underrun)** 0：无过载/欠载； 1：出现过载/欠载。 – 当NOSTRETCH=1时，在从模式下该位被硬件置位，同时： – 在接收模式中当收到一个新的字节时(包括ACK应答脉冲)，数据寄存器里的内容还未被读出，则新接收的字节将丢失。 – 在发送模式中当要发送一个新的字节时，却没有新的数据写入数据寄存器，同样的字节将被发送两次。 – 该位由软件写’0’清除，或在PE=0时由硬件清除。。 注：如果数据寄存器的写操作发生时间非常接近 SCL 的上升沿，发送的数据是不确定的，并发生保持时间错误。

AF	**应答失败 (Acknowledge failure)** 0：没有应答失败； 1：应答失败。 – 当没有返回应答时，硬件将置该位为’1’。 该位由软件写’0’清除，或在 PE=0 时由硬件清除。
ARLO	**仲裁丢失(主模式) (Arbitration lost (master mode))** 0：没有检测到仲裁丢失； 1：检测到仲裁丢失。 当接口失去对总线的控制给另一个主机时，硬件将将置该位为’1’。 – 该位由软件写’0’清除，或在PE=0时由硬件清除。 在ARLO事件之后，I2C接口自动切换回从模式(M/SL=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)。
BERR	**总线出错 (Bus error)** 0：无起始或停止条件出错； 1：起始或停止条件出错。 – 当接口检测到错误的起始或停止条件，硬件将该位置’1’。 – 该位由软件写’0’清除，或在PE=0时由硬件清除。
TxE	**数据寄存器为空(发送时) (Data register empty (transmitters))** 0：数据寄存器非空； 1：数据寄存器空。 – 在发送数据时，数据寄存器为空时该位被置’1’，在发送地址阶段不设置该位。 – 软件写数据到DR寄存器可清除该位；或在发生一个起始或停止条件后，或当PE=0时由硬件自动清除。 如果收到一个NACK，或下一个要发送的字节是PEC(PEC=1)，该位不被置位。 注：在写入第1个要发送的数据后，或设置了BTF时写入数据，都不能清除TxE位，这是因为数据寄存器仍然为空。
RxNE	**数据寄存器非空(接收时) (Data register not empty (receivers))** 0：数据寄存器为空； 1：数据寄存器非空。 – 在接收时，当数据寄存器不为空，该位被置’1’。在接收地址阶段，该位不被置位。 – 软件对数据寄存器的读写操作清除该位，或当PE=0时由硬件清除。 在发生ARLO事件时，RxNE不被置位。 注：当设置了BTF时，读取数据不能清除RxNE位，因为数据寄存器仍然为满。

STOPF	**停止条件检测位(从模式) (Stop detection (slave mode))** 0：没有检测到停止条件； 1：检测到停止条件。 – 在一个应答之后(如果ACK=1)，当从设备在总线上检测到停止条件时，硬件将该位置’1’。 – 软件读取SR1寄存器后，对CR1寄存器的写操作将清除该位，或当PE=0时，硬件清除该位。 注：在收到NACK后，STOPF位不被置位。
ADD10	**10位头序列已发送(主模式) (10-bit header sent (Master mode))** 0：没有ADD10事件发生； 1：主设备已经将第一个地址字节发送出去。 – 在10位地址模式下，当主设备已经将第一个字节发送出去时，硬件将该位置’1’。 – 软件读取SR1寄存器后，对CR1寄存器的写操作将清除该位，或当PE=0时，硬件清除该位。 注：收到一个NACK后，ADD10位不被置位。
BTF	**字节发送结束 (Byte transfer finished)** 0：字节发送未完成； 1：字节发送结束。 当NOSTRETCH=0时，在下列情况下硬件将该位置’1’： – 在接收时，当收到一个新字节(包括ACK脉冲)且数据寄存器还未被读取(RxNE=1)。 – 在发送时，当一个新数据将被发送且数据寄存器还未被写入新的数据(TxE=1)。 – 在软件读取SR1寄存器后，对数据寄存器的读或写操作将清除该位；或在传输中发送一个起始或停止条件后，或当PE=0时，由硬件清除该位。 注：在收到一个NACK后，BTF位不会被置位。 如果下一个要传输的字节是PEC(I2C_SR2寄存器中TRA为’1’，同时I2C_CR1寄存器中PEC 为’1’)，BTF位不会被置位。
ADDR	**地址已被发送(主模式)/地址匹配(从模式) (Address sent (master mode)/matched (slave mode))** 在软件读取SR1寄存器后，对SR2寄存器的读操作将清除该位，或当PE=0时，由硬件清除该位。 地址匹配(从模式) 0：地址不匹配或没有收到地址； 1：收到的地址匹配。 - 当收到的从地址与OAR寄存器中的内容相匹配、或发生广播呼叫、或SMBus设备默认地址或SMBus主机识别出SMBus提醒时，硬件就将该位置’1’(当对应的设置被使能时)。

	地址已被发送(主模式) 0：地址发送没有结束； 1：地址发送结束。 – 10位地址模式时，当收到地址的第二个字节的ACK后该位被置’1’。 – 7位地址模式时，当收到地址的ACK后该位被置’1’。注：在收到NACK后，ADDR位不会被置位。
SB	**起始位(主模式) (Start bit (Master mode))** 0：未发送起始条件； 1：起始条件已发送。 – 当发送出起始条件时该位被置’1’。 – 软件读取SR1寄存器后，写数据寄存器的操作将清除该位，或当PE=0时，硬件清除该位。

12.7.7 状态寄存器 2 (I2C_SR2)

状态寄存器 2 (I2C_SR2)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2) 偏移地址：0x18					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	PEC[7:0]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	DUALF	SMBHOST	SMBDEFAUL	GENCALL	X	TRA	BUSY	MSL
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PEC[15:8]	**数据包出错检测 (Packet error checking register)** 当 ENPEC=1 时，PEC[7:0]存放内部的 PEC 的值。
DUALF	**双标志(从模式) (Dual flag (Slave mode))** 0：接收到的地址与OAR1内的内容相匹配； 1：接收到的地址与OAR2内的内容相匹配。 - 在产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件将该位清除。

SMBHOST	**SMBus主机头系列(从模式) (SMBus host header (Slave mode))** 0：未收到SMBus主机的地址； 1：当SMBTYPE=1且ENARP=1时，收到SMBus主机地址。 – 在产生一个停止条件或一个重复的起始条件时，或 PE=0 时，硬件将该位清除。
SMBDEFAULT	**SMBus 设备默认地址(从模式) (SMBus device default address (Slave mode))** 0：未收到SMBus设备的默认地址； 1：当ENARP=1时，收到SMBus设备的默认地址。 – 在产生一个停止条件或一个重复的起始条件时，或PE=0时，硬件将该位清除。
GENCALL	**广播呼叫地址(从模式) (General call address (Slave mode))** 0：未收到广播呼叫地址； 1：当ENGC=1时，收到广播呼叫的地址。 – 在产生一个停止条件或一个重复的起始条件时，或PE=0时，硬件将该位清除。
TRA	**发送/接收 (Transmitter/receiver)** 0：接收到数据； 1：数据已发送； 在整个地址传输阶段的结尾，该位根据地址字节的R/W位来设定。 在检测到停止条件(STOPF=1)、重复的起始条件或总线仲裁丢失(ARLO=1)后，或当PE=0 时，硬件将其清除。
BUSY	**总线忙 (Bus busy)** 0：在总线上无数据通讯； 1：在总线上正在进行数据通讯。 – 在检测到SDA或SCl为低电平时，硬件将该位置’1’； – 当检测到一个停止条件时，硬件将该位清除。 该位指示当前正在进行的总线通讯，当接口被禁用(PE=0)时该信息仍然被更新。
MSL	**主从模式 (Master/slave)** 0：从模式； 1：主模式。 – 当接口处于主模式(SB=1)时，硬件将该位置位； – 当总线上检测到一个停止条件、仲裁丢失(ARLO=1时)、或当PE=0时，硬件清除该位。

12.7.8 时钟控制寄存器(I2C_CCR)

时钟控制寄存器(I2C_CCR)			基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2) 偏移地址：0x1C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8

Read: Write:	F/S	DUTY	X	X	CCR[11:8]			
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	CCR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
F/S	**I^2C主模式选项 (I^2C master mode selection)** 0：标准模式的I^2C； 1：快速模式的 I^2C。
DUTY	**快速模式时的占空比 (Fast mode duty cycle)** 0：快速模式下：T_{low}/T_{high} = 2； 1: 快速模式下：T_{low}/T_{high} = 16/9(见 CCR)。
CCR[11:0]	**快速/ 标准模式下的时钟控制分频系数(主模式)** **(Clock control register in Fast/Standard mode (Master mode))** 该分频系数用于设置主模式下的SCL时钟。在I2C标准模式或SMBus模式下： Thigh = CCR ×TPCLK1 Tlow = CCR ×TPCLK1 在I2C快速模式下： 如果DUTY = 0： Thigh = CCR ×TPCLK1 Tlow = 2 × CCR × TPCLK1 如果DUTY = 1： (速度达到400kHz) Thigh = 9 × CCR ×TPCLK1 Tlow = 16 × CCR ×TPCLK1 例如：在标准模式下，产生100kHz的SCL的频率： 如果FREQR = 08，TPCLK1 = 125ns，则CCR必须写入0x28(40×125ns = 5000 ns)。注：1. 允许设定的最小值为0x04，在快速DUTY模式下允许的最小值为0x01； 2. Thigh=tr(SCL)+tw(SCLH)，详见数据手册中对这些参数的定义； 3. Tlow=tf(SCL)+tw(SCLL)，详见数据手册中对这些参数的定义； 4. 这些延时没有过滤器； 5. 只有在关闭I2C时(PE = 0)才能设置CCR寄存器； 6. fCK应当是10MHz的整数倍，这样可以正确产生400kHz的快速时钟。

注：1. 要求 FPCLK1 应当是 10 MHz 的整数倍，这样可以正确地产生 400KHz 的快速时钟。
2. CCR 寄存器只有在关闭 I2C 时(PE=0)才能设置

12.7.9 TRISE 寄存器(I2C_TRISE)

TRISE 寄存器(I2C_TRISE)	基地址：0x4000 5400(I2C1) 0x4000 5800(I2C2)

			偏移地址：0x20					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	TRISE[5:0]					
Reset:	0	0	0	0	0	0	1	0

位	功能描述
TRISE[5:0]	**在快速/ 标准模式下的最大上升时间(主模式) (Maximum rise time in Fast/Standard mode (Master mode))** 这些位必须设置为I^2C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。如果在I2C_CR2寄存器中FREQ[5:0]中的值等于0x08且T_{PCLK1}=125ns，故TRISE[5:0]中必须写入09h(1000ns/125 ns = 8+1)。 滤波器的值也可以加到TRISE[5:0]内。 如果结果不是一个整数，则将整数部分写入 TRISE[5:0]以确保 t_{HIGH} 参数。注：只有当 I^2C 被禁用(PE=0)时，才能设置 TRISE[5:0]。

13 CAN 总线

13.1 bxCAN 简介

bxCAN 是基本扩展 CAN(Basic Extended CAN)的缩写，它支持 CAN 协议 2.0A 和 2.0B。它的设计目标是，以最小的 CPU 负荷来高效处理大量收到的报文。它也支持报文发送的优先级要求(优先级特性可软件配置)。

对于安全紧要的应用，bxCAN 提供所有支持时间触发通信模式所需的硬件功能。

13.2 bxCAN 主要特点

- 支持CAN协议2.0A和2.0B主动模式
- 波特率最高可达1兆位/秒
- 支持时间触发通信功能

发送

- 3个发送邮箱
- 发送报文的优先级特性可软件配置
- 记录发送SOF时刻的时间戳

接收

- 3级深度的2个接收FIFO
- 可变的过滤器组：
  - CAN1有28个过滤器组
- 标识符列表
- FIFO溢出处理方式可配置
- 记录接收SOF时刻的时间戳

时间触发通信模式

- 禁止自动重传模式
- 16位自由运行定时器
- 可在最后2个数据字节发送时间戳

管理

- 中断可屏蔽
- 邮箱占用单独1块地址空间，便于提高软件效率

13.3 bxCAN 总体描述

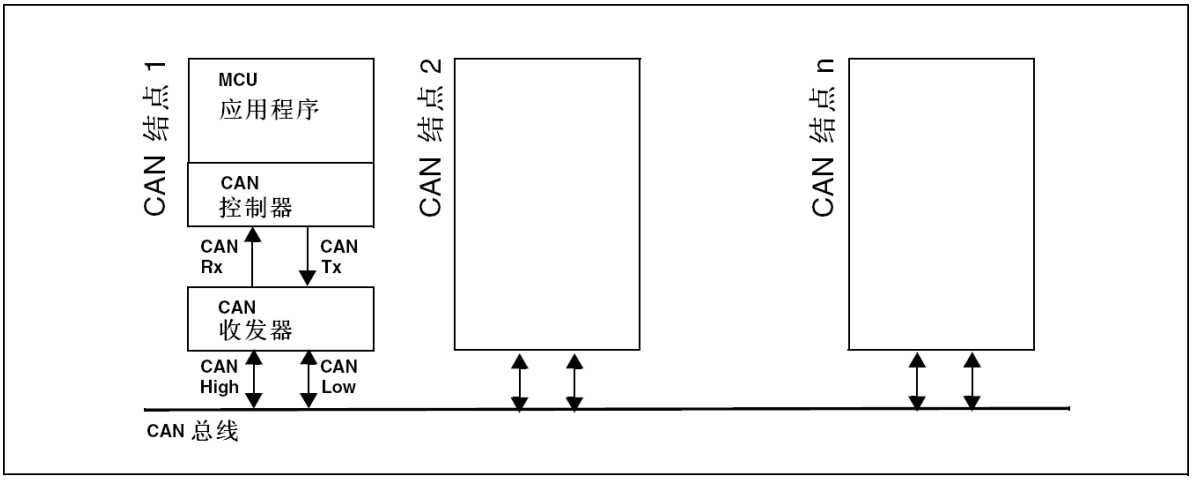
在当今的 CAN 应用中，CAN 网络的节点在不断增加，并且多个 CAN 常常通过网关连接起来，因此整个 CAN 网中的报文数量(每个节点都需要处理)急剧增加。除了应用层报文外，网络管理和诊断报文也被引入。

- 需要一个增强的过滤机制来处理各种类型的报文

此外，应用层任务需要更多 CPU 时间，因此报文接收所需的实时响应程度需要减轻。

- 接收FIFO的方案允许，CPU花很长时间处理应用层任务而不会丢失报文。

构筑在底层 CAN 驱动程序上的高层协议软件，要求跟 CAN 控制器之间
有高效的接口图:CAN 网拓扑结构



13.3.1 CAN 2.0B 主动内核

bxCAN 模块可以完全自动地接收和发送 CAN 报文；且完全支持标准标识符(11 位)和扩展标识符(29 位)。

13.3.2 控制、状态和配置寄存器

应用程序通过这些寄存器，可以：

- 配置CAN参数，如波特率
- 请求发送报文
- 处理报文接收
- 管理中断
- 获取诊断信息

13.3.3 发送邮箱

共有 3 个发送邮箱供软件来发送报文。发送调度器根据优先级决定哪个邮箱的报文先被发送。

13.3.4 接收过滤器

bxCAN 提供 28 个位宽可变/可配置的标识符过滤器组，软件通过对它们编程，从而
在引脚收到的报文中选择它需要的报文，而把其它报文丢弃掉

13.3.5 bxCAN 工作模式

bxCAN 有 3 个主要的工作模式：**初始化**、**正常和睡眠模式**。在硬件复位后，bxCAN 工作在睡眠模式以节省电能，同时 CANTX 引脚的内部上拉电阻被激活。软件通过对 CAN_MCR 寄存器的 INRQ 或 SLEEP 位置'1'，可以请求 bxCAN 进入**初始化**或**睡眠模式**。一旦进入了**初始化**或**睡眠模式**，bxCAN 就对 CAN_MSR 寄存器的 INAK 或 SLAK 位置'1'来进行确认，同时内部上拉电阻被禁用。当 INAK 和 SLAK 位都为'0'时，bxCAN 就处于**正常模式**。在进入**正常模式**前 bxCAN 必须跟 CAN 总线取得**同步**；为取得同步，bxCAN 要等待 CAN 总线达到空闲状态，即在 CANRX 引脚上监测到 11 个连续的隐性位。

13.3.6 初始化模式

软件初始化应该在硬件处于初始化模式时进行。设置 CAN_MCR 寄存器的 INRQ 位为'1'，请求 bxCAN 进入初始化模式，然后等待硬件对 CAN_MSR 寄存器的 INAK 位置'1'来进行确认。

清除 CAN_MCR 寄存器的 INRQ 位为'0'，请求 bxCAN 退出初始化模式，当硬件对 CAN_MSR 寄存器的 INAK 位清'0'就确认了初始化模式的退出。当 bxCAN 处于初始化模式时，禁止报文的接收和发送，并且 CANTX 引脚输出隐性位(高电平)。初始化模式的进入，不会改变配置寄存器。软件对 bxCAN 的初始化，至少包括位时间特性(CAN_BTR)和控制(CAN_MCR)这 2 个寄存器。 在对 bxCAN 的过滤器组(模式、位宽、FIFO 关联、激活和过滤器值)进行初始化前，软件要对 CAN_FMR 寄存器的 FINIT 位设置'1'。对过滤器的初始化可以在非初始化模式下进行。

注：　当 FINIT=1 时，报文的接收被禁止。
可以先对过滤器激活位清'0'(在 CAN_FA1R 中)，然后修改相应过滤器的值。
如果过滤器组没有使用，那么就应该让它处于非激活状态(保持其 FACT 位为清'0'状态)。

13.3.7 正常模式

在初始化完成后，软件应该让硬件进入正常模式，以便正常接收和发送报文。软件可以通过对 CAN_MCR 寄存器的 INRQ 位清'0'，来请求从初始化模式进入正常模式，然后要等待硬件对 CAN_MSR 寄存器的 INAK 位置'1'的确认。在跟 CAN 总线取得同步，即在 CANRX 引脚上监测到

11 个连续的隐性位(等效于总线空闲)后，bxCAN 才能正常接收和发送报文。

不需要在初始化模式下进行过滤器初值的设置，但必须在它处在非激活状态下完成(相应的 FACT 位为 0)。而过滤器的位宽和模式的设置，则必须在初始化模式中进入正常模式前完成。

13.3.8 睡眠模式(低功耗)

bxCAN 可工作在低功耗的睡眠模式。软件通过对 CAN_MCR 寄存器的 SLEEP 位置'1'，来请求进入这一模式。在该模式下，bxCAN 的时钟停止了，但软件仍然可以访问邮箱寄存器。

当 bxCAN 处于睡眠模式，软件必须对 CAN_MCR 寄存器的 INRQ 位置'1'并且同时对 SLEEP 位清'0'，才能进入初始化模式。

有 2 种方式可以唤醒(退出睡眠模式)bxCAN：通过软件对 SLEEP 位清'1'，或硬件检测到 CAN 总线的活动。

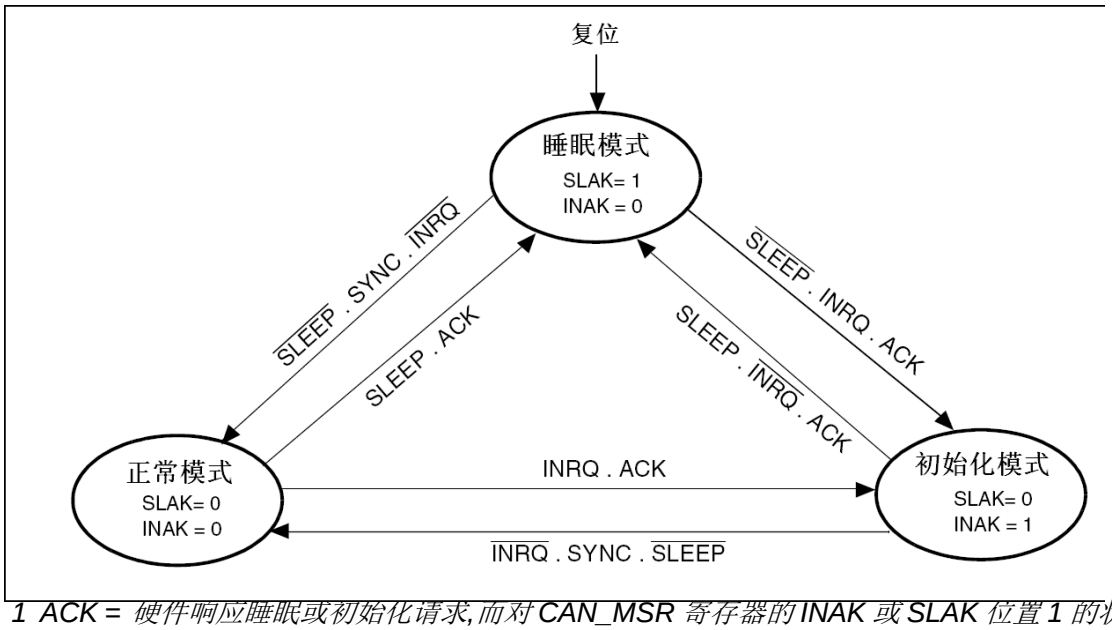
如果 CAN_MCR 寄存器的 AWUM 位为'1'，一旦检测到 CAN 总线的活动，硬件就自动对 SLEEP 位清'0'来唤醒 bxCAN。如果 CAN_MCR 寄存器的 AWUM 位为'0'，软件必须在唤醒中断里对 SLEEP 位清'0'才能退出睡眠状态。

注： *如果唤醒中断被允许(CAN_IER 寄存器的 WKUIE 位为'1')，那么一旦检测到 CAN 总线活动就会产生唤醒中断，而不管硬件是否会自动唤醒 bxCAN。*

在对 SLEEP 位清'0'后，睡眠模式的退出必须与 CAN 总线同步，请参考图 196：bxCAN 工作模式

。当硬件对 SLAK 位清'0'时，就确认了睡眠模式的退出。

图：bxCAN 工作模式



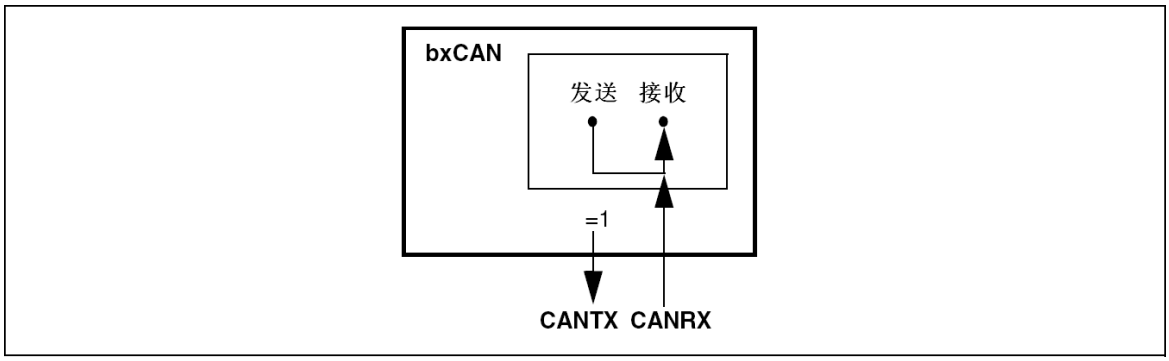
注：　1 ACK = 硬件响应睡眠或初始化请求,而对 CAN_MSR 寄存器的 INAK 或 SLAK 位置 1 的状态
2 SYNC = bxCAN 等待 CAN 总线变为空闲的状态，即在 CANRX 引脚上检测到连续的 11 个隐性位

13.3.9 静默模式

通过对 CAN_BTR 寄存器的 SILM 位置'1'，来选择静默模式。

在静默模式下，bxCAN 可以正常地接收数据帧和远程帧，但只能发出隐性位，而不能真正发送报文。如果 bxCAN 需要发出显性位(确认位、过载标志、主动错误标志)，那么这样的显性位在内部被接回来从而可以被 CAN 内核检测到，同时 CAN 总线不会受到影响而仍然维持在隐性位状态。因此，静默模式通常用于分析 CAN 总线的活动，而不会对总线造成影响－显性位(确认位、错误帧)不会真正发送到总线上。

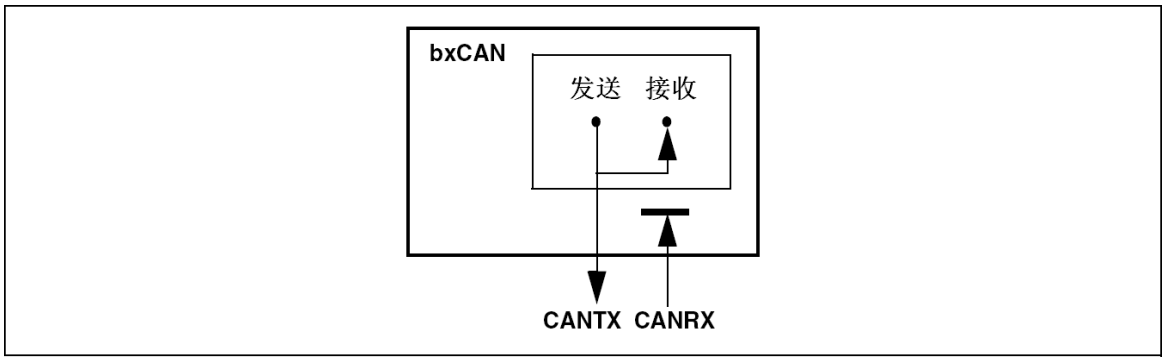
图：bxCAN 工作在静默模式



13.3.10环回模式

通过对 CAN_BTR 寄存器的 LBKM 位置'1'，来选择环回模式。在环回模式下，bxCAN 把发送的报文当作接收的报文并保存(如果可以通过接收过滤)在接收邮箱里。

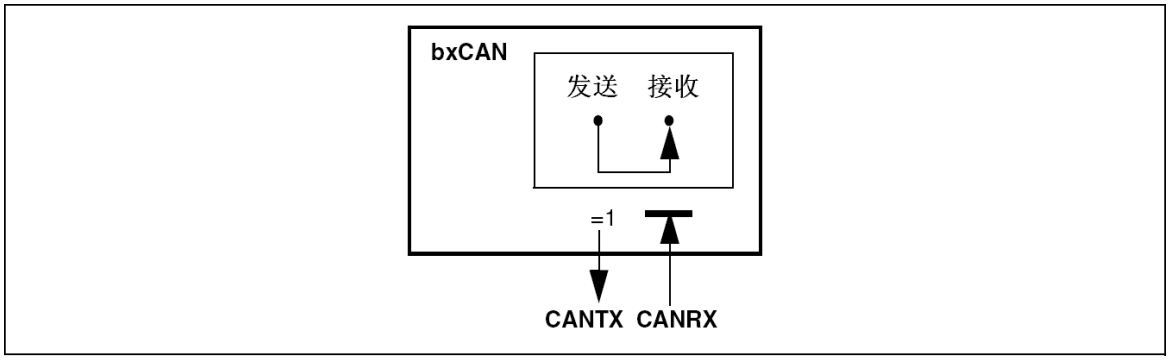
图：bxCAN 工作在环回模式



环回模式可用于自测试。为了避免外部的影响，在环回模式下 CAN 内核忽略确认错误(在数据/远程帧的确认位时刻，不检测是否有显性位)。在环回模式下，bxCAN 在内部把 Tx 输出回馈到 Rx 输入上，而完全忽略 CANRX 引脚的实际状态。发送的报文可以在 CANTX 引脚上检测到。

13.3.11环回静默模式

图：bxCAN 工作在环回静默模式



通过对 CAN_BTR 寄存器的 LBKM 和 SILM 位同时置'1'，可以选择环回静默模式。该模式可用于

"热自测试"，即可以像环回模式那样测试 bxCAN，但却不会影响 CANTX 和 CANRX 所连接的整个 CAN 系统。在环回静默模式下，CANRX 引脚与 CAN 总线断开，同时 CANTX 引脚被驱动到隐性位状态。

13.4 bxCAN 功能描述

13.4.1 发送处理

发送报文的流程为：应用程序选择 1 个**空置**的发送邮箱；设置标识符，数据长度和待发送数据； 然后对 CAN_TIxR 寄存器的 TXRQ 位置'1'，来请求发送。TXRQ 位置'1'后，邮箱就不再是空邮箱；而一旦邮箱不再为**空置**，软件对邮箱寄存器就不再有写的权限。TXRQ 位置 1 后，邮箱马上进入**挂号**状态，并等待成为最高优先级的邮箱，参见*发送优先级*。一旦邮箱成为最高优先级的邮箱，其状态就变为**预定**发送状态。一旦 CAN 总线进入空闲状态，预定发送邮箱中的报文就马上被发送(进入**发送**状态)。

一旦邮箱中的报文被成功发送后，它马上变为空置邮箱；硬件相应地对 CAN_TSR 寄存器的 RQCP 和 TXOK 位置 1，来表明一次成功发送。

如果发送失败，由于仲裁引起的就对 CAN_TSR 寄存器的 ALST 位置'1'，由于发送错误引起的就对 TERR 位置'1'。

发送优先级 由标识符决定

当有超过 1 个发送邮箱在挂号时，发送顺序由邮箱中报文的标识符决定。根据 CAN 协议，标识符数值最低的报文具有最高的优先级。如果标识符的值相等，那么邮箱号小的报文先被发送。

由发送请求次序决定

通过对 CAN_MCR 寄存器的 TXFP 位置'1'，可以把发送邮箱配置为发送 FIFO。在该模式下，发送的优先级由发送请求次序决定。该模式对分段发送很有用。

中止

通过对 CAN_TSR 寄存器的 ABRQ 位置'1'，可以中止发送请求。邮箱如果处于**挂号**或**预定**状态， 发送请求马上就被中止了。如果邮箱处于**发送**状态，那么中止请求可能导致 2 种结果。如果邮箱中的报文被成功发送，那么邮箱变为**空置**邮箱，并且 CAN_TSR 寄存器的 TXOK 位被硬件置'1'。如果邮箱中的报文发送失败了，那么邮箱变为**预定**状态，然后发送请求被中止，邮箱变为**空置**邮箱且 TXOK 位被硬件清'0'。因此如果邮箱处于**发送**状态，那么在发送操作结束后，邮箱都会变为**空置**邮箱。

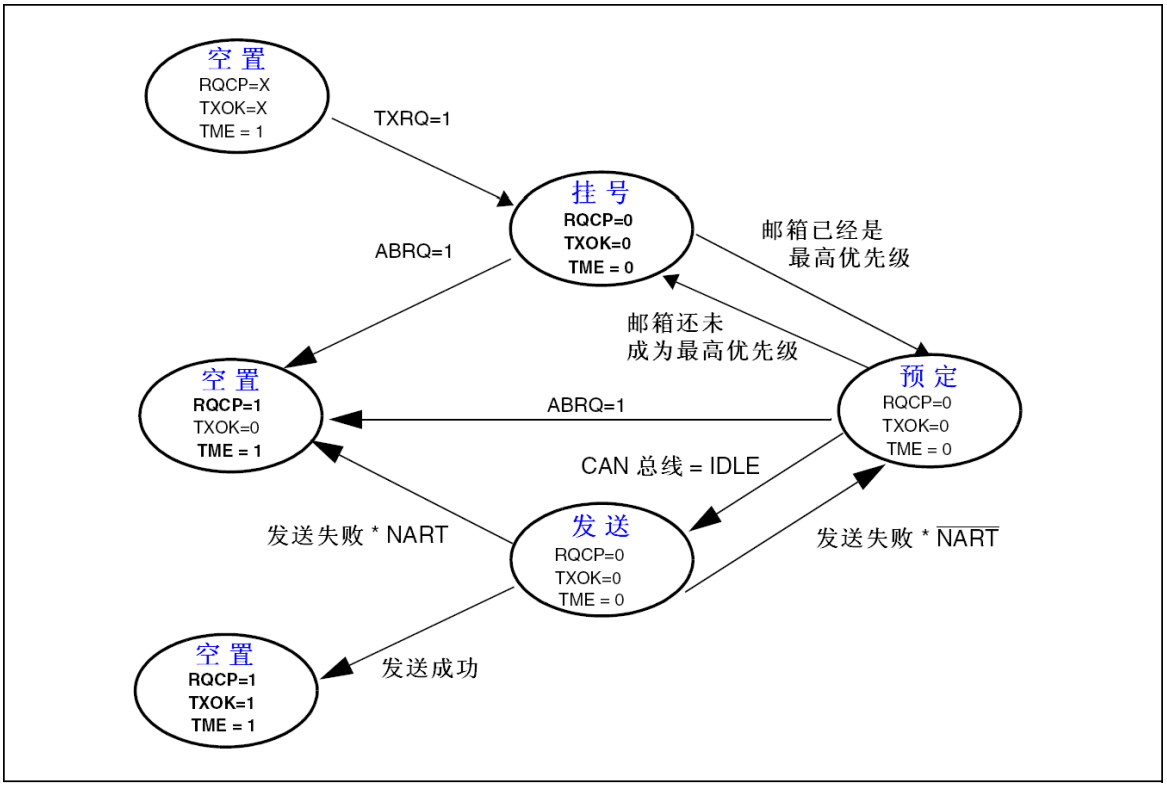
禁止自动重传模式

该模式主要用于满足 CAN 标准中，时间触发通信选项的需求。通过对 CAN_MCR 寄存器的 NART 位置'1'，来让硬件工作在该模式。

在该模式下，发送操作只会执行一次。如果发送操作失败了，不管是由于仲裁丢失或出错，硬件都不会再自动发送该报文。

在一次发送操作结束后，硬件认为发送请求已经完成，从而对 CAN_TSR 寄存器的 RQCP 位置'1'，同时发送的结果反映在 TXOK、ALST 和 TERR 位上。

图：发送邮箱状态


空 置
RQCP=X
TXOK=X
TME = 1
TXRQ=1
挂 号
RQCP=0
TXOK=0
TME = 0
ABRQ=1
邮箱已经是
最高优先级
邮箱还未
成为最高优先级
空 置
RQCP=1
TXOK=0
TME = 1
ABRQ=1
预 定
RQCP=0
TXOK=0
TME = 0
CAN 总线 = IDLE
发送失败 * NART
发 送
RQCP=0
TXOK=0
TME = 0
发送失败 * $\overline{\text{NART}}$
空 置
RQCP=1
TXOK=1
TME = 1
发送成功

13.4.2 时间触发通信模式

在该模式下，CAN 硬件的内部定时器被激活，并且被用于产生(发送与接收邮箱的)时间戳，分别存储在 CAN_RDTxR/CAN_TDTxR 寄存器中。内部定时器在每个 CAN 位时间(见 22.7.7 节)累加。内部定时器在接收和发送的帧起始位的采样点位置被采样，并生成时间戳。

13.4.3 接收管理

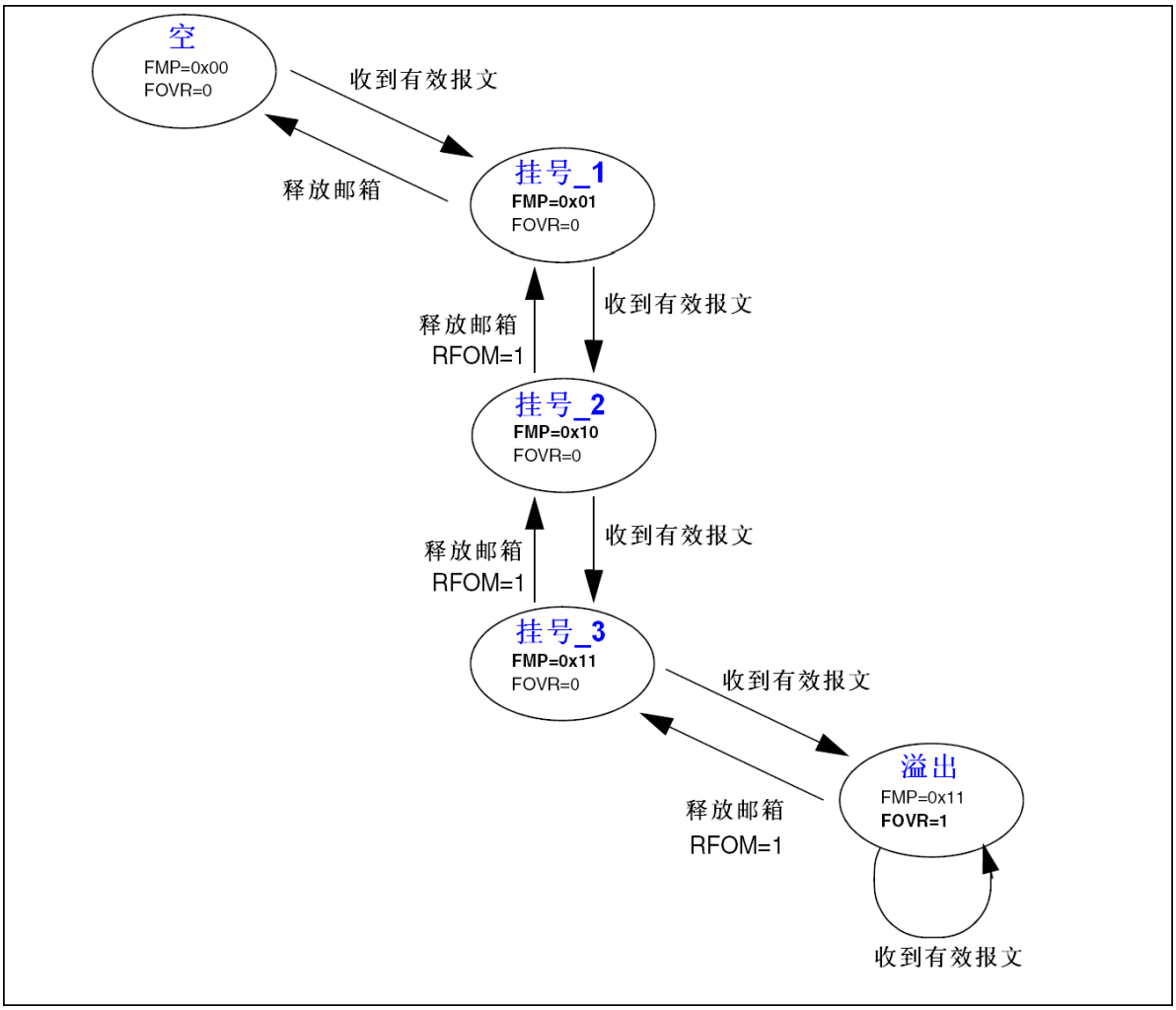
接收到的报文，被存储在 3 级邮箱深度的 FIFO 中。FIFO 完全由硬件来管理，从而节省了 CPU 的处理负荷，简化了软件并保证了数据的一致性。应用程序只能通过读取 FIFO 输出邮箱，来读取

FIFO 中最先收到的报文。

有效报文

根据 CAN 协议，当报文被正确接收(直到 EOF 域的最后一位都没有错误)，且通过了标识符过滤， 那么该报文被认为是有效报文。请参考 22.7.4 节：标识符过滤。

图：接收 FIFO 状态



FIFO 管理

FIFO 从空状态开始，在接收到第一个有效的报文后，FIFO 状态变为挂号_1(pending_1)，硬件相应地把 CAN_RFR 寄存器的 FMP[1:0]设置为'01'(二进制 01b)。软件可以读取 FIFO 输出邮箱来读出邮箱中的报文，然后通过对 CAN_RFR 寄存器的 RFOM 位设置'1'来释放邮箱，这样 FIFO 又变为空状态了。如果在释放邮箱的同时，又收到了一个有效的报文，那么 FIFO 仍然保留在挂号

_1 状态，软件可以读取 FIFO 输出邮箱来读出新收到的报文。

如果应用程序不释放邮箱，在接收到下一个有效的报文后，FIFO 状态变为挂号_2(pending_2)， 硬件相应地把 FMP[1:0]设置为'10'(二进制 10b)。重复上面的过程，第三个有效的报文把 FIFO 变

为**挂号_3** 状态(FMP[1:0]＝11b)。此时，软件必须对 RFOM 位设置 1 来释放邮箱，以便 FIFO 可以有空间来存放下一个有效的报文；否则，下一个有效的报文到来时就会导致一个报文的丢失。

溢出

当 FIFO 处于**挂号_3** 状态(即 FIFO 的 3 个邮箱都是满的)，下一个有效的报文就会导致**溢出**，并且一个报文会丢失。此时，硬件对 CAN_RFR 寄存器的 FOVR 位进行置'1'来表明溢出情况。至于哪个报文会被丢弃，取决于对 FIFO 的设置：

- 如果禁用了FIFO锁定功能(CAN_MCR寄存器的RFLM位被清'0')，那么FIFO中最后收到的报文就被新报文所覆盖。这样，最新收到的报文不会被丢弃掉。
- 如果启用了FIFO锁定功能(CAN_MCR寄存器的RFLM位被置'1')，那么新收到的报文就被丢弃，软件可以读到FIFO中最早收到的3个报文。

接收相关的中断

一旦往 FIFO 存入一个报文，硬件就会更新 FMP[1:0]位，并且如果 CAN_IER 寄存器的 FMPIE 位为'1'，那么就会产生一个中断请求。

当 FIFO 变满时(即第 3 个报文被存入)，CAN_RFR 寄存器的 FULL 位就被置'1'，并且如果

CAN_IER 寄存器的 FFIE 位为'1'，那么就会产生一个满中断请求。

在溢出的情况下，FOVR 位被置'1'，并且如果 CAN_IER 寄存器的 FOVIE 位为'1'，那么就会产生一个溢出中断请求。

13.4.4 标识符过滤

在 CAN 协议里，报文的标识符不代表节点的地址，而是跟报文的内容相关的。因此，发送者乙广播的形式把报文发送给所有的接收者。节点在接收报文时－根据标识符的值－决定软件是否需要该报文；如果需要，就拷贝到 SRAM 里；如果不需要，报文就被丢弃且无需软件的干预。

为满足这一需求，在互联型产品中，bxCAN 控制器为应用程序提供了 28 个位宽可变的、可配置的过滤器组(27~0)；在其它产品中，bxCAN 控制器为应用程序提供了 14 个位宽可变的、可配置的过滤器组(13~0)，以便只接收那些软件需要的报文。硬件过滤的做法节省了 CPU 开销，否则就必须由软件过滤从而占用一定的 CPU 开销。每个过滤器组 x 由 2 个 32 位寄存器，CAN_FxR0 和

CAN_FxR1 组成。

可变的位宽

每个过滤器组的位宽都可以独立配置，以满足应用程序的不同需求。根据位宽的不同，每个过滤器组可提供：

- 1个32位过滤器，包括：STDID[10:0]、EXTID[17:0]、IDE和RTR位
- 2个16位过滤器，包括：STDID[10:0]、IDE、RTR和

EXTID[17:15]位可参见图202。

此外过滤器可配置为，屏蔽位模式和标识符列表模式。

屏蔽位模式

在屏蔽位模式下，标识符寄存器和屏蔽寄存器一起，指定报文标识符的任何一位，应该按照“必须匹配”或“不用关心”处理。

标识符列表模式

在标识符列表模式下，屏蔽寄存器也被当作标识符寄存器用。因此，不是采用一个标识符加一个屏蔽位的方式，而是使用 2 个标识符寄存器。接收报文标识符的每一位都必须跟过滤器标识符相同。

过滤器组位宽和模式的设置

过滤器组可以通过相应的 CAN_FMR 寄存器配置。在配置一个过滤器组前，必须通过清除

CAN_FAR 寄存器的 FACT 位，把它设置为禁用状态。通过设置 CAN_FS1R 的相应 FSCx 位，可

以配置一个过滤器组的位宽，请参见图 202。通过 CAN_FMR 的 FBMx 位，可以配置对应的屏蔽/ 标识符寄存器的**标识符列表**模式或**屏蔽位**模式。

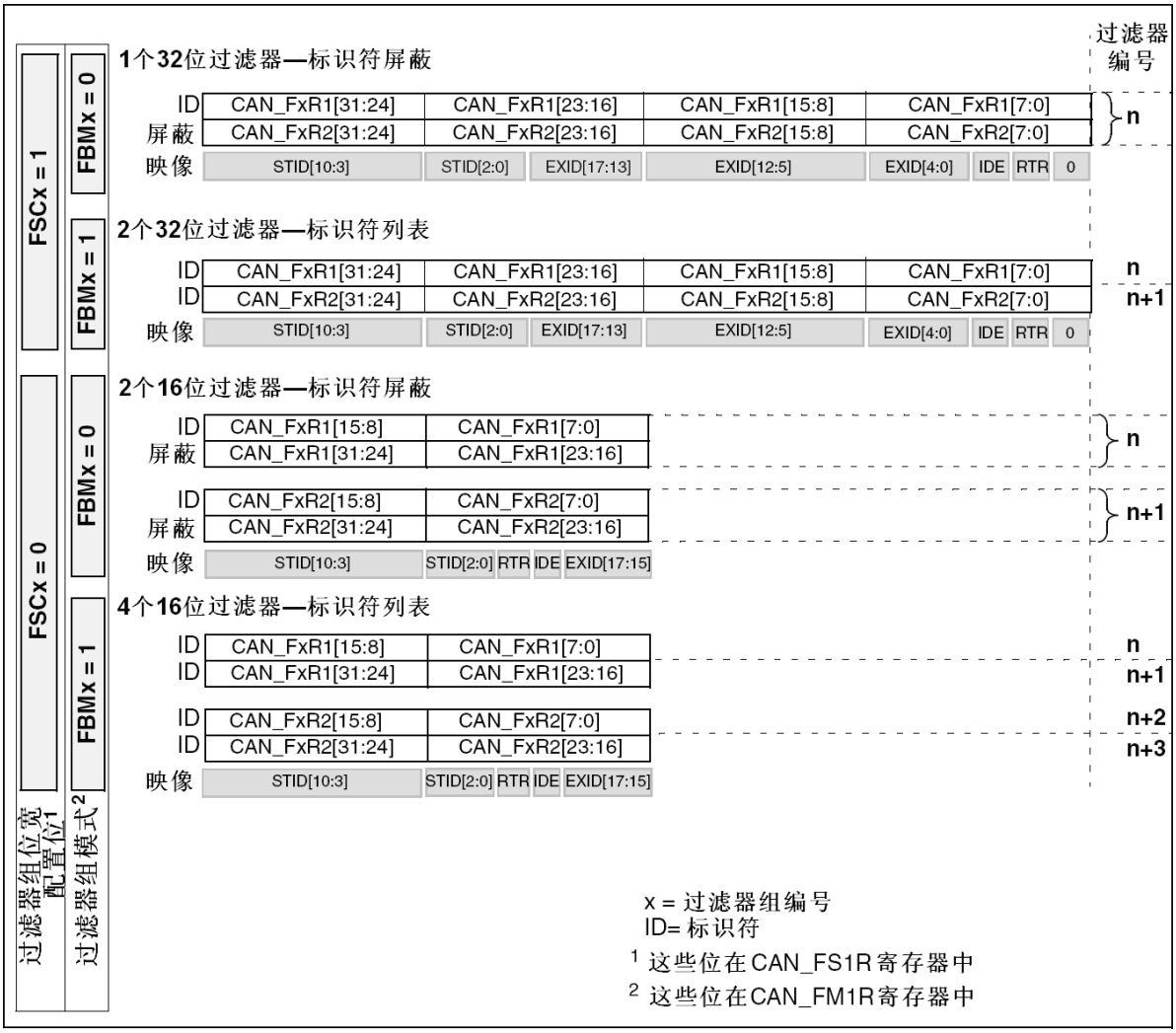
为了过滤出一组标识符，应该设置过滤器组工作在屏蔽位模式。

为了过滤出一个标识符，应该设置过滤器组工作在标识符列表模式。应用程序不用的过滤器组，应该保持在禁用状态。

过滤器组中的每个过滤器，都被编号为(叫做过滤器号)从 0 开始，到某个最大数值一取决于过滤器组的模式和位宽的设置。

关于过滤器配置，参见下图。

图：过滤器组位宽设置－寄存器组织



过滤器匹配序号

一旦收到的报文被存入 FIFO，就可被应用程序访问。通常情况下，报文中的数据被拷贝到 SRAM 中；为了把数据拷贝到合适的位置，应用程序需要根据报文的标识符来辨别不同的数据。bxCAN 提供了过滤器匹配序号，以简化这一辨别过程。

根据过滤器优先级规则，过滤器匹配序号和报文一起，被存入邮箱中。因此每个收到的报文， 都有与它相关联的过滤器匹配序号。

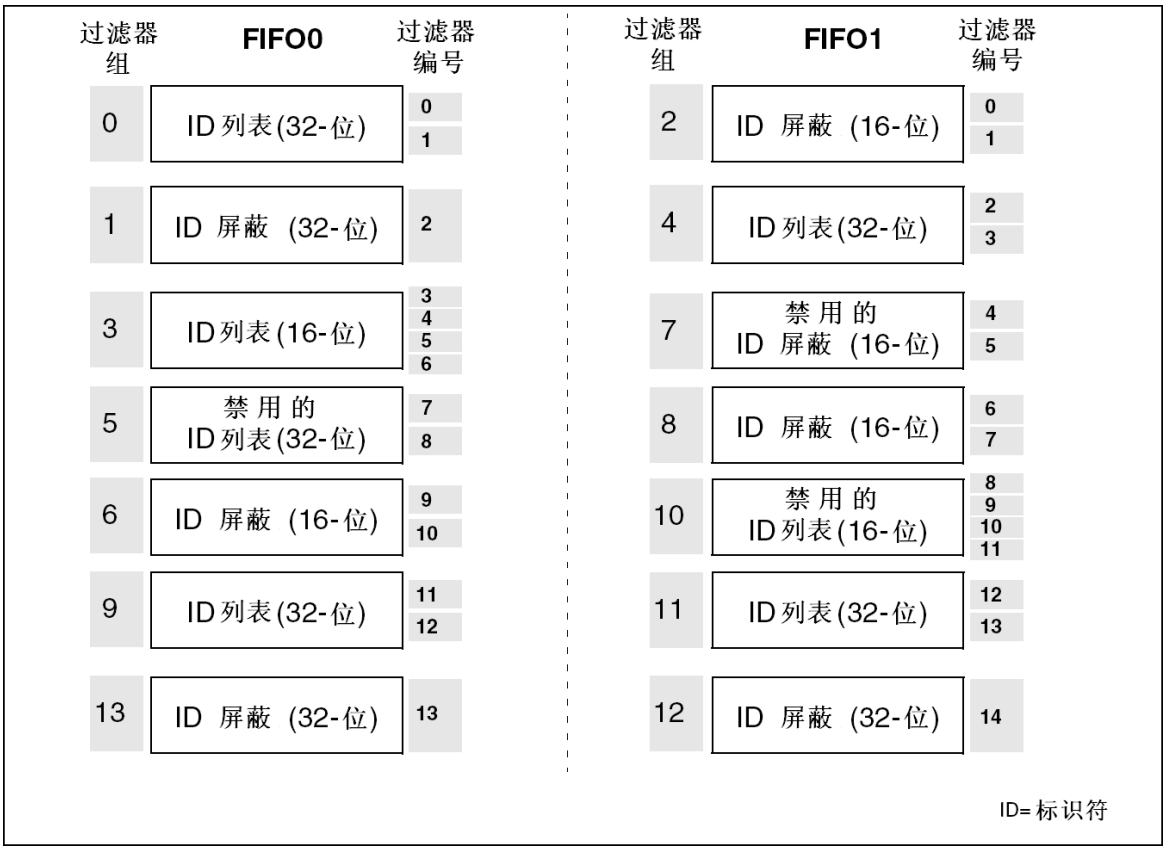
过滤器匹配序号可以通过下面两种方式来使用：

- 把过滤器匹配序号跟一系列所期望的值进行比较
- 把过滤器匹配序号当作一个索引来访问目标地址

对于标识符列表模式下的过滤器(非屏蔽方式的过滤器)，软件不需要直接跟标识符进行比较。对于屏蔽位模式下的过滤器，软件只须对需要的那些屏蔽位(必须匹配的位)进行比较即可。

在给过滤器编号时，并不考虑过滤器组是否为激活状态。另外，每个 FIFO 各自对其关联的过滤器进行编号。请参考下图的例子。

图：过滤器编号的例子

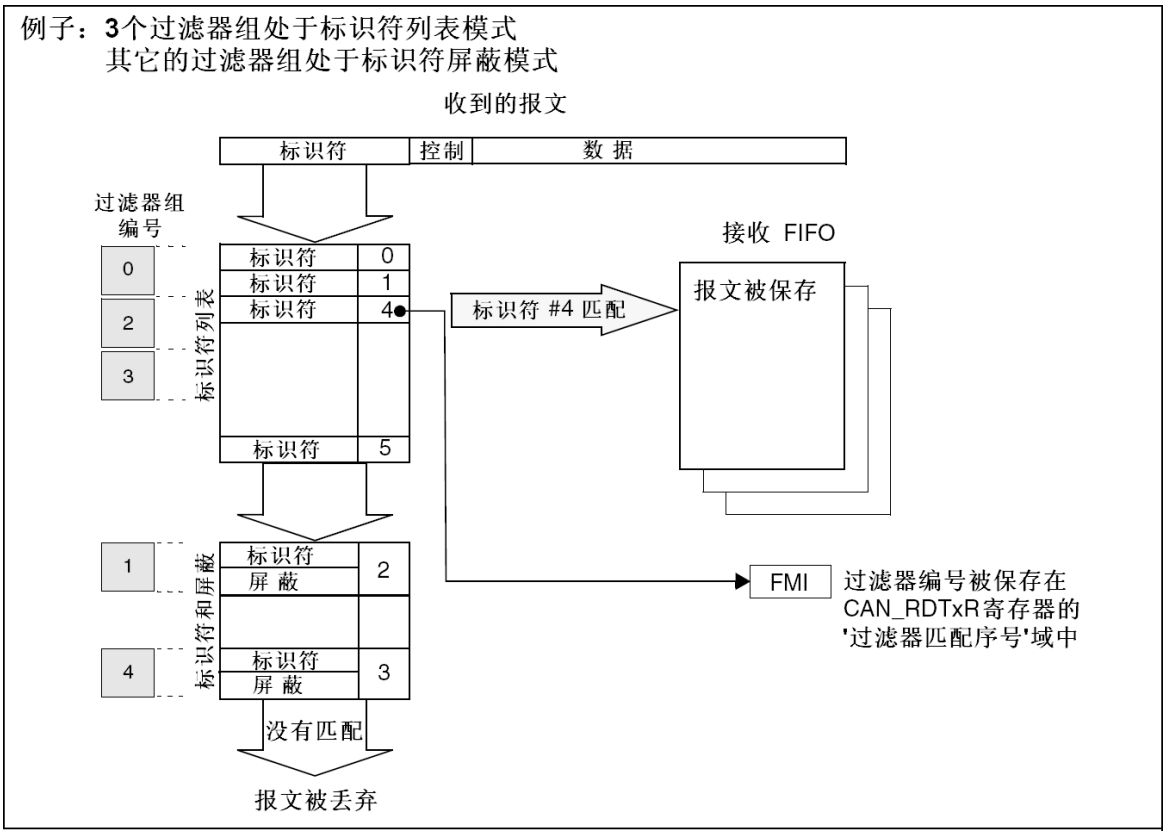


过滤器优先级规则

根据过滤器的不同配置，有可能一个报文标识符能通过多个过滤器的过滤；在这种情况下，存放在接收邮箱中的过滤器匹配序号，根据下列优先级规则来确定：

- 位宽为32位的过滤器，优先级高于位宽为16位的过滤器
- 对于位宽相同的过滤器，标识符列表模式的优先级高于屏蔽位模式
- 位宽和模式都相同的过滤器，优先级由过滤器号决定，过滤器号小的优先级高

图：过滤器机制的例子



上面的例子说明了 bxCAN 的过滤器规则：在接收一个报文时，其标识符首先与配置在标识符列表模式下的过滤器相比较；如果匹配上，报文就被存放到相关联的 FIFO 中，并且所匹配的过滤器的序号被存入过滤器匹配序号中。如同例子中所显示，报文标识符跟#4 标识符匹配，因此报文内容和 FMI4 被存入 FIFO。

如果没有匹配，报文标识符接着与配置在屏蔽位模式下的过滤器进行比较。

如果报文标识符没有跟过滤器中的任何标识符相匹配，那么硬件就丢弃该报文，且不会对软件有任何打扰。

13.4.5 报文存储

邮箱是软件和硬件之间传递报文的接口。邮箱包含了所有跟报文有关的信息：标识符、数据、控制、状态和时间戳信息。

发送邮箱

软件需要在一个空的发送邮箱中，把待发送报文的各种信息设置好(然后再发出发送的请求)。发送的状态可通过查询 CAN_TSR 寄存器获知。

表：发送邮箱寄存器列表

相对发送邮箱基地址的偏移量	寄存器名
0	CAN_TIxR
4	CAN_TDTxR

8	CAN_TDLxR
12	CAN_TDHxR

接收邮箱(FIFO)

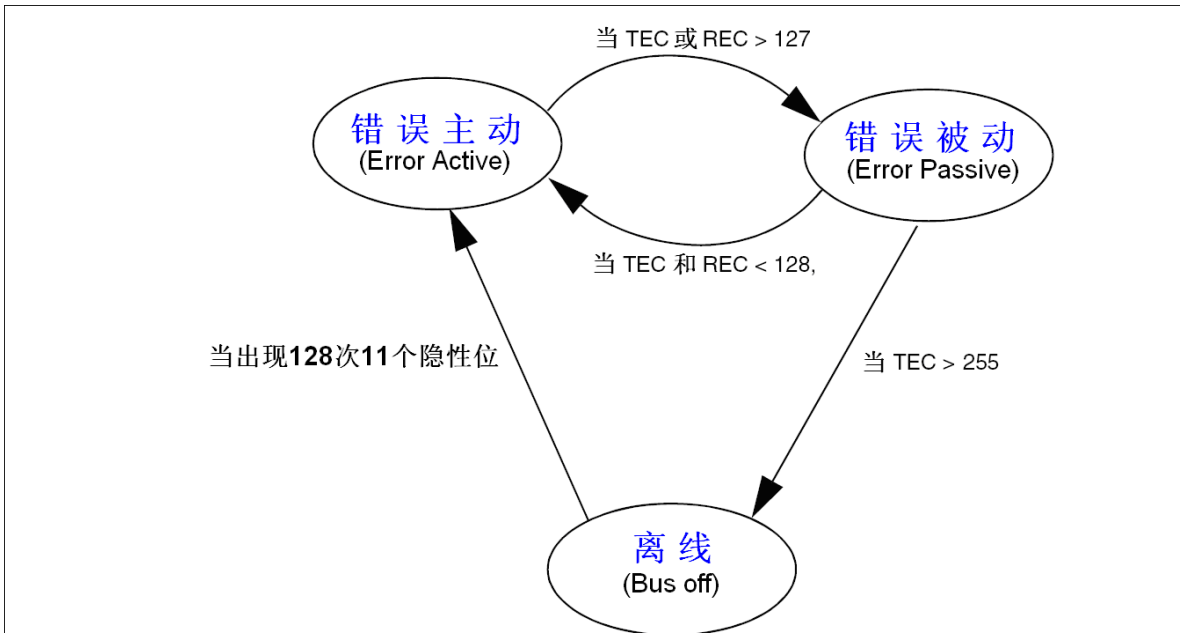
在接收到一个报文后，软件就可以访问接收 FIFO 的输出邮箱来读取它。一旦软件处理了报文(如把它读出来)，软件就应该对 CAN_RFxR 寄存器的 RFOM 位进行置'1'，来释放该报文，以便为后

面收到的报文留出存储空间。过滤器匹配序号存放在 CAN_RDTxR 寄存器的 FMI 域中。16 位的时间戳存放在 CAN_RDTxR 寄存器的 TIME[15:0]域中。

表：接收邮箱寄存器列表

相对接收邮箱基地址的偏移量	寄存器名
0	CAN_RIxR
4	CAN_RDTxR
8	CAN_RDLxR
12	CAN_RDHxR

图：CAN 错误状态图



13.4.6 出错管理

CAN 协议描述的出错管理，完全由硬件通过发送错误计数器(CAN_ESR 寄存器里的 TEC 域)，和接收错误计数器(CAN_ESR 寄存器里的 REC 域)来实现，其值根据错误的情况而增加或减少。关于 TEC 和 REC 管理的详细信息，请参考 CAN 标准。

软件可以读出它们的值来判断 CAN 网络的稳定性。

此外，CAN_ESR 寄存器提供了当前错误状态的详细信息。通过设置 CAN_IER 寄存器(比如 ERRIE 位)，当检测到出错时软件可以灵活地控制中断的产生。

离线恢复

当 TEC 大于 255 时，bxCAN 就进入离线状态，同时 CAN_ESR 寄存器的 BOFF 位被置'1'。在离线状态下，bxCAN 无法接收和发送报文。

根据 CAN_MCR 寄存器中 ABOM 位的设置，bxCAN 可以自动或在软件的请求下，从离线状态恢复(变为错误主动状态)。在这两种情况下，bxCAN 都必须等待一个 CAN 标准所描述的恢复过程(CAN RX 引脚上检测到 128 次 11 个连续的隐性位)。

如果 ABOM 位为'1'，bxCAN 进入离线状态后，就自动开启恢复过程。

如果 ABOM 位为'0'，软件必须先请求 bxCAN 进入然后再退出初始化模式，随后恢复过程才被开启。

注： *在初始化模式下，bxCAN 不会监视 CAN RX 引脚的状态，这样就不能完成恢复过程。**为了完成恢复过程，bxCAN 必须工作在正常模式**。*

13.4.7 位时间特性

位时间特性逻辑通过采样来监视串行的 CAN 总线，并且通过与帧起始位的边沿进行同步，及通过与后面的边沿进行重新同步，来调整其采样点。

它的操作可以简单解释为，如下所述把名义上的每位时间分为 3 段：

- **同步段(SYNC_SEG)**：通常期望位的变化发生在该时间段内。其值固定为1个时间单元(1 x t_{CAN})。
- **时间段1(BS1)**：定义采样点的位置。它包含CAN标准里的PROP_SEG和PHASE_SEG1。其值可以编程为1到16个时间单元，但也可以被自动延长，以补偿因为网络中不同节点的频率差异所造成的相位的正向漂移。
- **时间段2(BS2)**：定义发送点的位置。它代表CAN标准里的PHASE_SEG2。其值可以编程为 1 到 8 个时间单元，但也可以被自动缩短以补偿相位的负向漂移。

重新同步跳跃宽度(SJW)定义了，在每位中可以延长或缩短多少个时间单元的上限。其值可以编程为 1 到 4 个时间单元。

有效跳变被定义为，当 bxCAN 自己没有发送隐性位时，从显性位到隐性位的第 1 次转变。

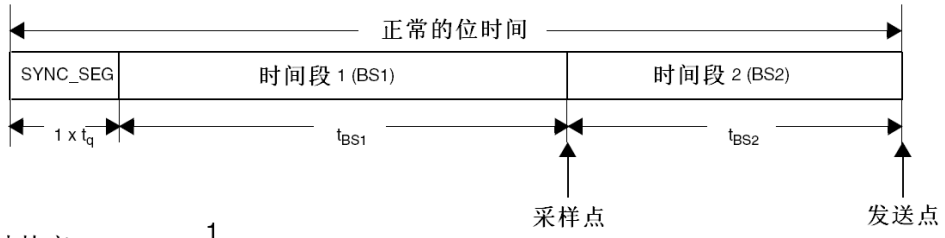
如果在时间段 1(BS1)而不是在同步段(SYNC_SEG)检测到有效跳变，那么 BS1 的时间就被延长最多 SJW 那么长，从而采样点被延迟了。

相反如果在时间段 2(BS2)而不是在 SYNC_SEG 检测到有效跳变，那么 BS2 的时间就被缩短最多

SJW 那么长，从而采样点被提前了。

为了避免软件的编程错误，对位时间特性寄存器(CAN_BTR)的设置，只能在 bxCAN 处于初始化状态下进行。

图：位时序



$$波特率 = \frac{1}{正常的位时间}$$

正常的位时间 = $1 \times t_q + t_{BS1} + t_{BS2}$

其中：

$t_{BS1} = t_q \times (TS1[3:0] + 1)$,

$t_{BS2} = t_q \times (TS2[2:0] + 1)$,

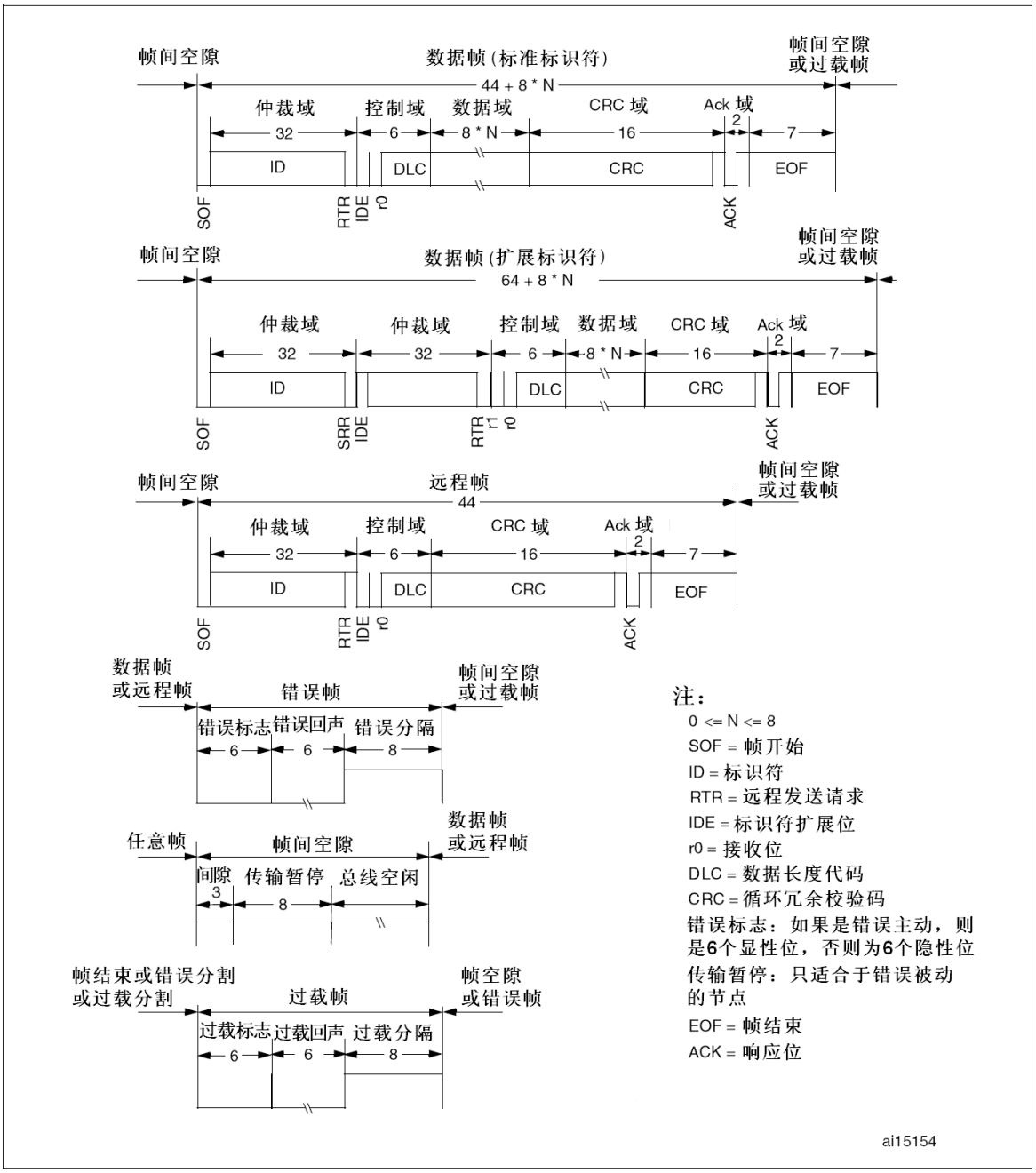
$t_q = (BRP[9:0] + 1) \times t_{PCLK}$

这里 t_q 表示1个时间单元

t_{PCLK} = APB 时钟的时间周期

BRP[9:0], TS1[3:0] 和 TS2[2:0] 在 CAN_BTR 寄存器中定义

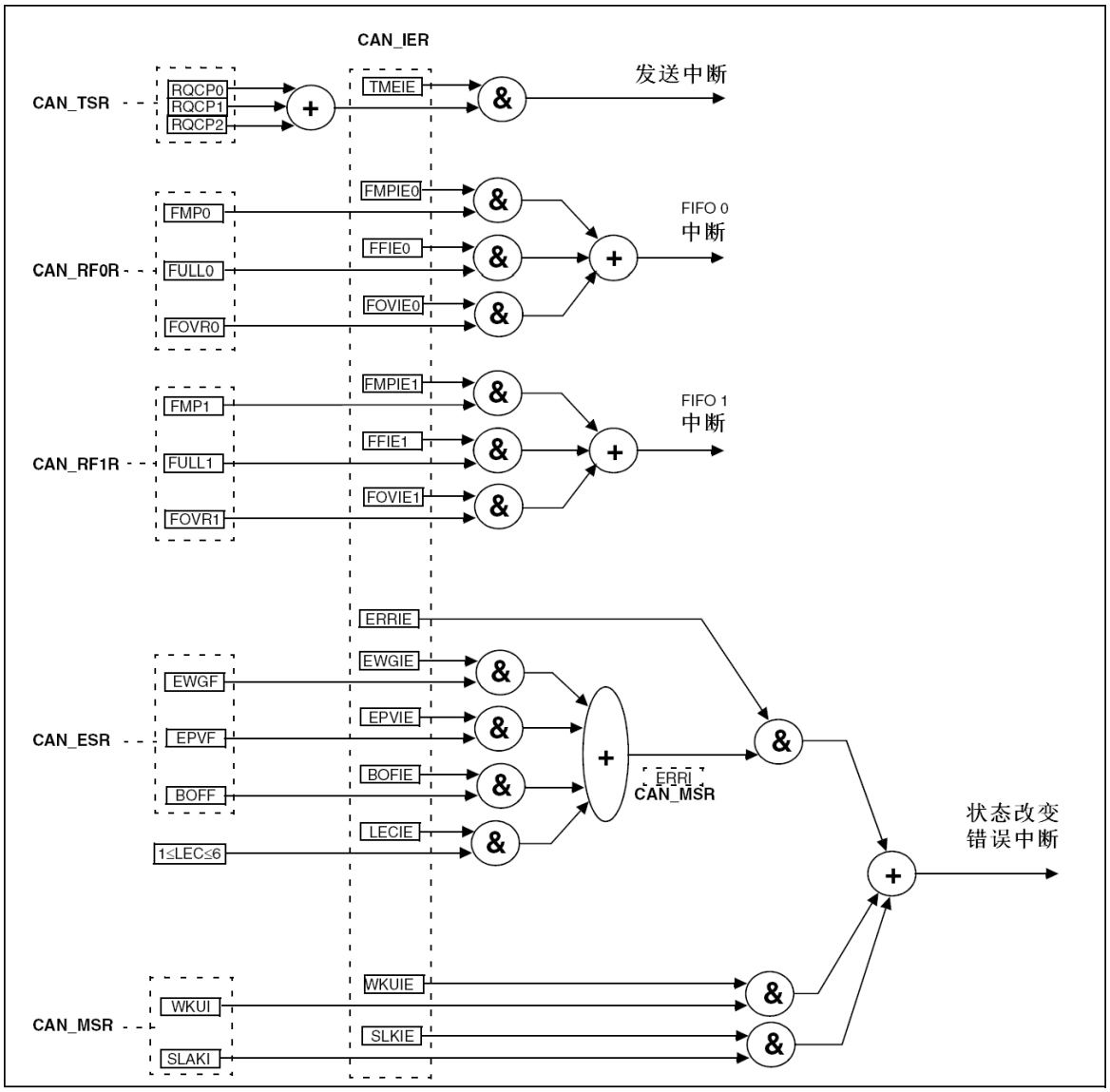
图：各种 CAN 帧


帧间空隙
数据帧(标准标识符)
帧间空隙
或过载帧
44 + 8 * N
仲裁域
控制域
数据域
CRC 域
Ack 域
32
6
8 * N
16
2
7
ID
DLC
CRC
EOF
SOF
RTR
IDE
r0
ACK
帧间空隙
数据帧(扩展标识符)
帧间空隙
或过载帧
64 + 8 * N
仲裁域
仲裁域
控制域
数据域
CRC 域
Ack 域
32
32
6
8 * N
16
2
7
ID
DLC
CRC
EOF
SOF
SRR
IDE
RTR
r1
r0
ACK
帧间空隙
远程帧
帧间空隙
或过载帧
44
仲裁域
控制域
CRC 域
Ack 域
32
6
16
2
7
ID
DLC
CRC
EOF
SOF
RTR
IDE
r0
ACK
数据帧
或远程帧
错误帧
帧间空隙
或过载帧
错误标志
错误回声
错误分隔
6
6
8
任意帧
帧间空隙
数据帧
或远程帧
间隙
传输暂停
总线空闲
3
8
帧结束或错误分割
或过载分割
过载帧
帧空隙
或错误帧
过载标志
过载回声
过载分隔
6
6
8
注：
0 <= N <= 8
SOF = 帧开始
ID = 标识符
RTR = 远程发送请求
IDE = 标识符扩展位
r0 = 接收位
DLC = 数据长度代码
CRC = 循环冗余校验码
错误标志：如果是错误主动，则是6个显性位，否则为6个隐性位
传输暂停：只适合于错误被动的节点
EOF = 帧结束
ACK = 响应位
ai15154

13.5 bxCAN 中断

bxCAN 占用 4 个专用的中断向量。通过设置 CAN 中断允许寄存器(CAN_IER)，每个中断源都可以单独允许和禁用。

图：事件标志和中断产生



- 发送中断可由下列事件产生：
  - 发送邮箱0变为空，CAN_TSR寄存器的RQCP0位被置'1'。
  - 发送邮箱1变为空，CAN_TSR寄存器的RQCP1位被置'1'。
  - 发送邮箱2变为空，CAN_TSR寄存器的RQCP2位被置'1'。
- FIFO0中断可由下列事件产生：
  - FIFO0接收到一个新报文，CAN_RF0R寄存器的FMP0位不再是'00'。
  - FIFO0变为满的情况，CAN_RF0R寄存器的FULL0位被置'1'。
  - FIFO0发生溢出的情况，CAN_RF0R寄存器的FOVR0位被置'1'。
- FIFO1中断可由下列事件产生：

- FIFO1接收到一个新报文，CAN_RF1R寄存器的FMP1位不再是'00'。
- FIFO1变为满的情况，CAN_RF1R寄存器的FULL1位被置'1'。
- FIFO1发生溢出的情况，CAN_RF1R寄存器的FOVR1位被置'1'。

● **错误和状态变化中断**可由下列事件产生：

— 出错情况，关于出错情况的详细信息请参考CAN错误状态寄存器(CAN_ESR)。
— 唤醒情况，在CAN接收引脚上监视到帧起始位(SOF)。
— CAN进入睡眠模式。

13.6 CAN 控制寄存器说明

对某些寄存器的错误访问会导致一个 CAN 节点对整个 CAN 网络的暂时性干扰。因此，软件只能在 CAN 处于初始化模式时修改 CAN_BTR 寄存器。
虽然错误数据的发送对 CAN 网的网络层不会带来问题，但却会对应用程序造成严重影响。因此，软件只能在发送邮箱为空的状态改变它，请参见图 200。
过滤器的数值只能在关闭对应过滤器组的状态下，或设置 FINIT 位为’1’后才能修改。此外，只有在设置整个过滤器为初始化模式下(即 FINIT=1)，才能修改过滤器的设置，即修改 CAN_FMxR，CAN_FSxR 和 CAN_FFAR 寄存器。

13.6.1 CAN 主控制寄存器 (CAN_MCR)

CAN 主控制寄存器 (CAN_MCR)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	DBF
Reset:	0	0	0	0	0	0	0	1
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	RESET	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TTCM	ABOM	AWUM	NART	RFLM	TXFP	SLEEP	INRQ
Reset:	0	0	0	0	0	0	1	0

位	功能描述
DBF	**调试冻结 (Debug freeze)** 0：在调试时，CAN 照常工作 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。
RESET	**bxCAN 软件复位 (bxCAN software master reset)** 0：本外设正常工作； 1：对 bxCAN 进行强行复位，复位后 bxCAN 进入睡眠模式(FMP 位和 CAN_MCR 寄存器被初始化为其复位值)。此后硬件自动对该位清’0’。
TTCM	**时间触发通信模式 (Time triggered communication mode)** 0：禁止时间触发通信模式；

	1：允许时间触发通信模式。
ABOM	**自动离线(Bus-Off)管理 (Automatic bus-off management)** 该位决定 CAN 硬件在什么条件下可以退出离线状态。 0：离线状态的退出过程是，软件对 CAN_MCR 寄存器的 INRQ 位进行置’1’随后清’0’后，一旦硬件检测到 128 次 11 位连续的隐性位，则退出离线状态； 1：一旦硬件检测到 128 次 11 位连续的隐性位，则自动退出离线状态。
AWUM	**自动唤醒模式 (Automatic wakeup mode)** 该位决定CAN处在睡眠模式时由硬件还是软件唤醒 0：睡眠模式通过清除CAN_MCR寄存器的SLEEP位，由软件唤醒； 1：睡眠模式通过检测 CAN 报文，由硬件自动唤醒。唤醒的同时，硬件自动对 CAN_MSR 寄存器的 SLEEP 和 SLAK 位清’0’ 。
NART	**禁止报文自动重传 (No automatic retransmission)** 0：按照CAN标准，CAN硬件在发送报文失败时会一直自动重传直到发送成功； 1：CAN 报文只被发送 1 次，不管发送的结果如何(成功、出错或仲裁丢失)。
RFLM	**接收FIFO锁定模式(Receive FIFO locked mode)** 0：在接收溢出时FIFO未被锁定，当接收FIFO的报文未被读出，下一个收到的报文会覆盖原有的报文； 1：在接收溢出时 FIFO 被锁定，当接收 FIFO 的报文未被读出，下一个收到的报文会被丢弃。
TXFP	**发送FIFO优先级 (Transmit FIFO priority)** 当有多个报文同时在等待发送时，该位决定这些报文的发送顺序 0：优先级由报文的标识符来决定； 1：优先级由发送请求的顺序来决定。
SLEEP	**睡眠模式请求 (Sleep mode request)** 软件对该位置’1’可以请求CAN进入睡眠模式，一旦当前的CAN活动(发送或接收报文)结束，CAN就进入睡眠。 软件对该位清’0’使CAN退出睡眠模式。 当设置了 AWUM 位且在 CAN Rx 信号中检测出 SOF 位时，硬件对该位清’0’。在复位后该位被置’1’，即 CAN 在复位后处于睡眠模式。
INRQ	**初始化请求 (Initialization request)** 软件对该位清’0’可使CAN从初始化模式进入正常工作模式：当CAN在接收引脚检测到连续的 11 个隐性位后， CAN 就达到同步，并为接收和发送数据作好准备了。为此，硬件相应地对 CAN_MSR寄存器的INAK位清’0’。 软件对该位置 1 可使 CAN 从正常工作模式进入初始化模式：一旦当前的 CAN 活动(发送或接收)结束，CAN 就进入初始化模式。相应地，硬件对 CAN_MSR 寄存器的 INAK 位置’1’。

13.6.2 主状态寄存器 (CAN_MSR)

主状态寄存器 (CAN_MSR)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	RX	SAMP	RXM	TXM
Reset:	0	0	0	0	1	1	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	X	X	X	SLAKI	WKUI	ERRI	SLAK	INAK
Reset:	0	0	0	0	0	0	1	0

位	功能描述
RX	**CAN 接收电平 (CAN Rx signal)** 该位反映 CAN 接收引脚(CAN_RX)的实际电平。
SAMP	**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)。
RXM	**接收模式 (Receive mode)** 该位为’1’表示 CAN 当前为接收器。
TXM	**发送模式 (Transmit mode)** 该位为’1’表示 CAN 当前为发送器。
SLAKI	**睡眠确认中断 (Sleep acknowledge interrupt)** 当SLKIE=1，一旦CAN进入睡眠模式硬件就对该位置’1，紧接着相应的中断被触发。当设置该位为’1’时，如果设置了CAN_IER寄存器中的SLKIE位，将产生一个状态改变中断。 软件可对该位清’0’，当SLAK位被清’0’时硬件也对该位清’0’。 注：当 SLKIE=0，不应该查询该位，而应该查询 SLAK 位来获知睡眠状态。
WKUI：	**唤醒中断挂号 (Wakeup interrupt)** 当CAN处于睡眠状态，一旦检测到帧起始位(SOF)，硬件就置该位为’1’；并且如果CAN_IER寄存器的WKUIE位为’1’，则产生一个状态改变中断。 该位由软件清’0’。
ERRI	**出错中断挂号 (Error interrupt)** 当检测到错误时，CAN_ESR寄存器的某位被置’1’，如果CAN_IER寄存器的相应中断使能位也被置’1’时，则硬件对该位置’1’；如果CAN_IER寄存器的ERRIE位为’1’，则产生状态改变中断。 该位由软件清’0’。
SLAK	**睡眠模式确认** 该位由硬件置’1’，指示软件CAN模块正处于睡眠模式。 该位是对软件请求进入睡眠模式的确认(对CAN_MCR寄存器的SLEEP位置’1’)。 当CAN退出睡眠模式时硬件对该位清’0’ (需要跟CAN总线同步)。 这里跟CAN总线同步是指，硬件需要在CAN的RX引脚上检测到连续的11位隐性位。 注：通过软件或硬件对CAN_MCR的SLEEP位清’0’，将启动退出睡眠模式的过程。有关清除 SLEEP 位的详细信息，参见 CAN_MCR 寄存器的 AWUM 位的描述。
INAK	**初始化确认** 该位由硬件置’1’，指示软件CAN模块正处于初始化模式。 该位是对软件请求进入初始化模式的确认(对CAN_MCR寄存器的INRQ位置’1’)。 当 CAN 退出初始化模式时硬件对该位清’0’ (需要跟 CAN 总线同步)。这里跟 CAN 总线同步是指， 硬件需要在 CAN 的 RX 引脚上检测到连续的 11 位隐性位。

13.6.3 CAN 发送状态寄存器 (CAN_TSR)

	基地址：0x4000 6400(CAN1)

CAN 发送状态寄存器 (CAN_TSR)			偏移地址：0x08					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	LOW2	LOW1	LOW0	TME2	TME1	TME0	CODE[25:24]	
Reset:	0	0	0	1	1	1	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	ABRQ2	X	X	X	TERR2	ALST2	TX0K2	RQCP2
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	ABRQ1	X	X	X	TERR1	ALST1	TX0K1	RQCP1
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	ABRQ0	X	X	X	TERR0	ALST0	TX0K0	RQCP0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LOW2	**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’。
LOW1	**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’。
LOW0	**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 LOW[31:29]被清’0’。
TME2	**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’。
TME1	**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’。
TME0	**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’。
CODE[25:24]	**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。
ABRQ2	**邮箱 2 中止发送（Abort request for mailbox 2）** 软件对该位置’1’，可以中止邮箱 2 的发送请求，当邮箱 2 的发送报文被清除时硬件对该位清’0’。如果邮箱 2 中没有等待发送的报文，则对该位置’1’没有任何效果。
TERR2	**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’。
ALST2	**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’。
TXOK2	**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：上次发送尝试失败； 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’。
RQCP2	**邮箱 2 请求完成 (Request completed mailbox 2)** 当上次对邮箱 2 的请求(发送或中止)完成后，硬件对该位置’1’。 软件对该位写’1’可以对其清’0’；当硬件接收到发送请求时也对该位清’0’(CAN_TI2R 寄存器的 TXRQ 位被置’1’)。 该位被清’0’时，邮箱 2 的其它发送状态位(TXOK2, ALST2 和 TERR2)也被清’0’。
ABRQ1	**邮箱 1 中止发送 (Abort request for mailbox 1)** 软件对该位置’1’，可以中止邮箱 1 的发送请求，当邮箱 1 的发送报文被清除时硬件对该位清’0’。如果邮箱 1 中没有等待发送的报文，则对该位置’1’没有任何效果。
TERR1	**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’。
ALST1	**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’。
TXOK1	**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：上次发送尝试失败； 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’。
RQCP1	**邮箱 1 请求完成 (Request completed mailbox 1)** 当上次对邮箱 1 的请求(发送或中止)完成后，硬件对该位置’1’。软件对该位写’1’可以对其清’0’；当硬件接收到发送请求时也对该位清’0’(CAN_TI1R 寄存器的 TXRQ 位被置’1’)。该位被清’0’时，邮箱 1 的其它发送状态位(TXOK1, ALST1 和 TERR1)也被清’0’。
ABRQ0	**邮箱 0 中止发送 (Abort request for mailbox 0)** 软件对该位置’1’可以中止邮箱 0 的发送请求，当邮箱 0 的发送报文被清除时硬件对该位清’0’。如果邮箱 0 中没有等待发送的报文，则对该位置 1 没有任何效果。
TERR0	**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’。
ALST0	**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’。
TXOK0	**邮箱 0 发送成功 (Transmission OK of mailbox 0)** 每次在邮箱 0 进行发送尝试后，硬件对该位进行更新： 0：上次发送尝试失败； 1：上次发送尝试成功。 当邮箱 0 的发送请求被成功完成后，硬件对该位置’1’。
RQCP1	**邮箱 0 请求完成 (Request completed mailbox 0)** 当上次对邮箱 0 的请求(发送或中止)完成后，硬件对该位置’1’。软件对该位写’1’可以对其清’0’；当硬件接收到发送请求时也对该位清’0’(CAN_TI0R 寄存器的 TXRQ 位被置’1’)。该位被清’0’时，邮箱 0 的其它发送状态位(TXOK0, ALST0 和 TERR0)也被清’0’。

13.6.4 CAN 接收 FIFO 0 寄存器 (CAN_RF0R)

CAN 接收 FIFO 0 寄存器 (CAN_RF0R)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	PFOM0	FOVR0	FULL0	X	FMP0[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
RFOM0	释放接收FIFO 0输出邮箱 (Release FIFO 0 output mailbox) 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’。
FOVR0	FIFO 0溢出 (FIFO 0 overrun) 当 FIFO 0 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’。
FULL0	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’。
FMP0[1:0]	FIFO 0 报文数目 (FIFO 0 message pending) FIFO 0报文数目这2位反映了当前接收FIFO 0中存放的报文数 目。每当1个新的报文被存入接收FIFO 0，硬件就对FMP0加1。 每当软件对 RFOM0 位写’1’来释放输出邮箱，FMP0 就被减 1，直到其为 0。

13.6.5 CAN 接收 FIFO 1 寄存器(CAN_RF1R)

CAN 接收 FIFO 1 寄存器 (CAN_RF1R)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X

Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	PFOM1	FOVR1	FULL1	X	FMP1[1:0]	
Reset:	0	0	0	0	0	0	0	0

位	功能描述
RFOM1	**释放接收FIFO 1输出邮箱 (Release FIFO 1 output mailbox)** 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'。
FOVR1	**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'。
FULL1	**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'。
FMP1[1:0]	**FIFO 1报文数目 (FIFO 1 message pending)** FIFO 1报文数目这2位反映了当前接收FIFO 1中存放的报文数 目。每当1个新的报文被存入接收FIFO 1，硬件就对FMP1加1。 每当软件对 RFOM1 位写 1 来释放输出邮箱，FMP1 就被减 1，直到其为 0。

13.6.6 CAN 中断使能寄存器 (CAN_IER)

CAN 中断使能寄存器 (CAN_IER)			基地址：0x4000 6400(CAN1) 偏移地址：0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	SCLKIE	WKUIE
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8

Read: Write:	ERRIE	X	X	X	LECIE	BOFIE	EPVIE	EWGIE
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	X	FOVIE1	FFIE1	FMPIE1	FOVIE0	FFIE0	FMPIE0	TMEIE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SLKIE	**睡眠中断使能 (Sleep interrupt enable)** 0：当 SLAKI 位被置'1'时，不产生中断； 1：当 SLAKI 位被置'1'时，产生中断。
WKUIE	**唤醒中断使能 (Wakeup interrupt enable)** 0：当 WKUI 位被置'1'时，不产生中断； 1：当 WKUI 位被置'1'时，产生中断。
ERRIE	**错误中断使能 (Error interrupt enable)** 0：当 CAN_ESR 寄存器有错误挂号时，不产生中断； 1：当 CAN_ESR 寄存器有错误挂号时，产生中断。
LECIE	**上次错误号中断使能 (Last error code interrupt enable)** 0：当检测到错误，硬件设置 LEC[2:0]时，不设置 ERRI 位； 1：当检测到错误，硬件设置 LEC[2:0]时，设置 ERRI 位为'1'。
BOFIE	**离线中断使能 (Bus-off interrupt enable)** 0：当 BOFF 位被置'1'时，不设置 ERRI 位； 1：当 BOFF 位被置'1'时，设置 ERRI 位为'1'。
EPVIE	**错误被动中断使能 (Error Passive Interrupt Enable)** 0：当 EPVF 位被置'1'时，不设置 ERRI 位； 1：当 EPVF 位被置'1'时，设置 ERRI 位为'1'。
EWGIE	**错误警告中断使能 (Error warning interrupt enable)** 0：当 EWGF 位被置'1'时，不设置 ERRI 位； 1：当 EWGF 位被置'1'时，设置 ERRI 位为'1'。
FOVIE1	**FIFO 1 溢出中断使能 (FIFO overrun interrupt enable)** 0：当 FIFO 1 的 FOVR 位被置'1'时，不产生中断； 1：当 FIFO 1 的 FOVR 位被置'1'时，产生中断。
FFIE1	**FIFO 1 满中断使能 (FIFO full interrupt enable)** 0：当 FIFO 1 的 FULL 位被置'1'时，不产生中断； 1：当 FIFO 1 的 FULL 位被置'1'时，产生中断。
FMPIE1	**FIFO 1 消息挂号中断使能 (FIFO message pending interrupt enable)** 0：当 FIFO 1 的 FMP[1:0]位为非 0 时，不产生中断； 1：当 FIFO 1 的 FMP[1:0]位为非 0 时，产生中断。
FOVIE0	**FIFO 0 溢出中断使能 (FIFO overrun interrupt enable)** 0：当 FIFO 0 的 FOVR 位被置'1'时，不产生中断； 1：当 FIFO 0 的 FOVR 位被置'1'时，产生中断。
FFIE0	**FIFO 0 满中断使能 (FIFO full interrupt enable)** 0：当 FIFO 0 的 FULL 位被置'1'时，不产生中断； 1：当 FIFO 0 的 FULL 位被置'1'时，产生中断。
FMPIE0	**FIFO 0 消息挂号中断使能 (FIFO message pending interrupt enable)** 0：当 FIFO 0 的 FMP[1:0]位为非 0 时，不产生中断； 1：当 FIFO 0 的 FMP[1:0]位为非 0 时，产生中断。
TMEIE	**发送邮箱空中断使能 (Transmit mailbox empty interrupt enable)** 0：当 RQCPx 位被置'1'时，不产生中断； 1：当 RQCPx 位被置'1'时，产生中断。

13.6.7 CAN 错误状态寄存器 (CAN_ESR)

CAN 错误状态寄存器 (CAN_ESR)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	REC[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	TEC[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	LEC[6:4]			X	BOFF	EPVF	WEGF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
REC[31:24]	**接收错误计数器 (Receive error counter)** 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。
TEC[23:16]	**9位发送错误计数器的低8位 (Least significant byte of the 9-bit transmit error counter)** 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。
LEC[6:4]	**上次错误代码 (Last error code)** 在检测到CAN总线上发生错误时，硬件根据出错情况设置。当报文被正确发送或接收后，硬件清除其值为'0'。 硬件没有使用错误代码7，软件可以设置该值，从而可以检测代码的更新。 000: 没有错误； 001: 位填充错； 010: 格式(Form)错； 011: 确认(ACK)错； 100: 隐性位错； 101: 显性位错； 110: CRC错； 111: 由软件设置。
BOFF	**离线标志 (Bus-off flag)** 当进入离线状态时，硬件对该位置'1'。当发送错误计数器TEC溢出，即大于255时，CAN进入离线状态。
EPVF	**错误被动标志 (Error passive flag)**

	当出错次数达到错误被动的阈值时，硬件对该位置'1'。 (接收错误计数器或发送错误计数器的值>127)。
EWGF	**错误警告标志 (Error warning flag)** 当出错次数达到警告的阈值时，硬件对该位置'1'。 (接收错误计数器或发送错误计数器的值≥96)。

13.6.8 CAN 位时序寄存器 (CAN_BTR)

CAN 位时序寄存器 (CAN_BTR)			基地址：0x4000 6400(CAN1) 偏移地址：0x1C					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	SILM	LBKM	X	X	X	X	SJW[1:0]	
Reset:	0	0	0	0	0	0	0	1
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	TS2[22:20]			TS1[19:16]			
Reset:	0	0	1	0	0	0	1	1
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	X	X	X	X	X	X	BRP[9:8]	
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	BRP[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SILM:	**静默模式(用于调试) (Silent mode (debug))** 0: 正常状态； 1: 静默模式。
LBKM	**环回模式(用于调试) (Loop back mode (debug))** 0: 禁止环回模式； 1: 允许环回模式。
SJW[25:24]	**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。 $t_{RJW} = t_{CAN} \times (SJW[1:0] + 1)$。
TS2[22:20]	**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 $t_{BS2} = t_{CAN} \times (TS2[2:0] + 1)$。
TS1[19:16]	**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 tBS1 = tCAN x (TS1[3:0] + 1)

	关于位时间特性的详细信息，请参考22.7.7节位时间特性。
BRP[9:0]	**波特率分频器 (Baud rate prescaler)** 该位域定义了时间单元(tq)的时间长度 tq = (BRP[9:0]+1) x tPCLK

注：当 CAN 处于初始化模式时，该寄存器只能由软件访问。

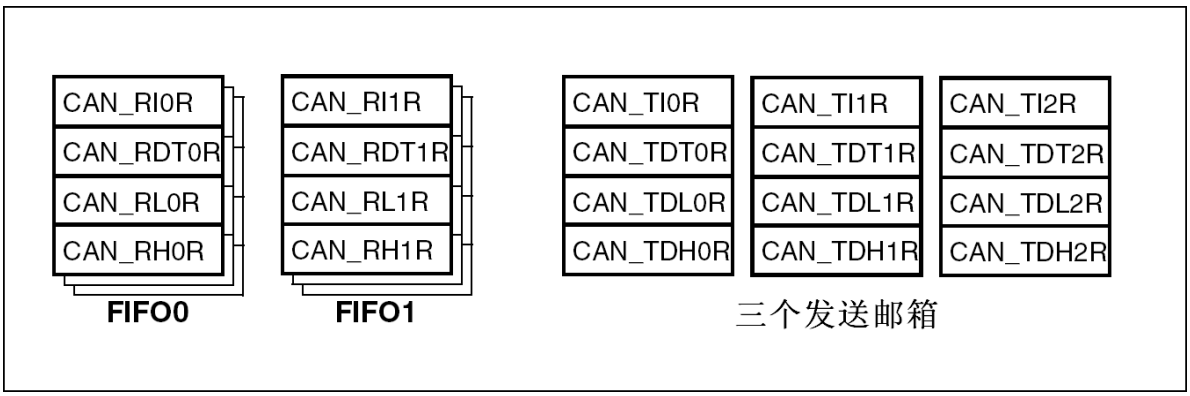
13.7 CAN 邮箱寄存器说明

本节描述发送和接收邮箱寄存器。关于寄存器映像的详细信息，请参考 22.7.5 节报文存储。除了下述例外，发送和接收邮箱几乎一样：

- CAN_RDTxR 寄存器的 FMI 域；
- 接收邮箱是只读的；
- 发送邮箱只有在它为空时才是可写的，CAN_TSR 寄存器的相应 TME 位为’1’，表示发送邮箱为空。

共有 3 个发送邮箱和 2 个接收邮箱。每个接收邮箱为 3 级深度的 FIFO，并且只能访问 FIFO 中最先收到的报文。

每个邮箱包含 4 个寄存器



13.7.1 发送邮箱标识符寄存器 (CAN_TIxR) (x=0..2)

发送邮箱标识符寄存器 (CAN_TIxR) (x=0..2)			基地址：0x4000 6400(CAN1) 偏移地址：0x18 0x19 0x1A					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	STID/EXID[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	STID/EXID[23:21]			EXID[20:16]				
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8

Read: Write:	EXID[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	EXID[7:3]					IDE	RTR	TXRQ
Reset:	0	0	0	0	0	0	0	0

位	功能描述
STID/EXID [31:21]	**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。
EXID[20:3]	**扩展标识符 (Extended identifier)** 扩展身份标识的低字节。
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。
RTR	**远程发送请求 (Remote transmission request)** 0：数据帧； 1：远程帧。
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，硬件对其清'0'。

注：复位值: 0xXXXX XXXX，X=未定义位(除了第 0 位，复位时 TXRQ=0)
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

13.7.2 发送邮箱数据长度和时间戳寄存器 (CAN_TDTxR) (x=0..2)

发送邮箱数据长度和时间戳寄存器 (CAN_TDTxR) (x=0..2)			基地址：0x4000 6400(CAN1) 偏移地址：0x184 0x194 0x1A4					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	TIME[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	TIME[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	TGT
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	DLC[3:0]			

Reset:	0	0	0	0	0	0	0	0

位	功能描述
TIME[31:16]	**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。
TGT	**发送时间戳 (Transmit global time)** 只有在CAN处于时间触发通信模式，即CAN_MCR寄存器的TTCM位为'1'时，该位才有效。 0：不发送时间戳TIME[15:0]； 1：发送时间戳TIME[15:0]。在长度为8的报文中，时间戳TIME[15:0]是最后2个发送的字节：TIME[7:0]作为第 7 个字节，TIME[15:8]为第 8 个字节，它们替换了写入 CAN_TDHxR[31:16]的数据(DATA6[7:0]和 DATA7[7:0])。为了把时间戳的 2 个字节发送出去，DLC 必须编程为 8。
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。

注：当邮箱不在空置状态时，该寄存器的所有位为写保护。

13.7.3 发送邮箱低字节数据寄存器 (CAN_TDLxR) (x=0..2)

发送邮箱低字节数据寄存器 (CAN_TDLxR) (x=0..2)			基地址：0x4000 6400(CAN1) 偏移地址：0x188 0x198 0x1A8					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	DATA3[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	DATA2[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	DATA1[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	DATA0[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DATA3[31:24]	**数据字节3 (Data byte 3)** 报文的数据字节 3。
DATA2[23:16]	**数据字节2 (Data byte 2)** 报文的数据字节 2。
DATA1[15:8]	**数据字节1 (Data byte 1)**

	报文的数据字节1。
DATA0[7:0]	**数据字节0 (Data byte 0)** 报文的数据字节0。 报文包含0到8个字节数据，且从字节0开始。

注：当邮箱不在空置状态时，该寄存器的所有位为写保护。

13.7.4 发送邮箱高字节数据寄存器 (CAN_TDHxR) (x=0..2)

发送邮箱高字节数据寄存器 (CAN_TDHxR) (x=0..2)			基地址：0x4000 6400(CAN1) 偏移地址：0x18C 0x19C 0x1AC					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	DATA7[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	DATA6[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DATA5[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DATA4[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DATA7[31:24]	**数据字节7 (Data byte 7)** 报文的数据字节7 **注:** 如果 CAN_MCR 寄存器的 TTCM 位为'1'，且该邮箱的 TGT 位也为'1'，那么 DATA7 和 DATA6 将被 TIME 时间戳代替。
DATA6[23:16]	**数据字节6 (Data byte 6)** 报文的数据字节 6。
DATA5[15:8]	**数据字节5 (Data byte 5)** 报文的数据字节 5。
DATA4[7:0]	**数据字节4 (Data byte 4)** 报文的数据字节 4。

注：当邮箱不在空置状态时，该寄存器的所有位为写保护。

13.7.5 接收 FIFO 邮箱标识符寄存器 (CAN_RIxR) (x=0..1)

	基地址：0x4000 6400(CAN1)

接收 FIFO 邮箱标识符寄存器 (CAN_RIxR) (x=0..1)			偏移地址：0x1B0 0x1C0					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	STID/EXID[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	STID/EXID[23:21]			EXID[20:16]				
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	EXID[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	EXID[7:3]					IDE	RTR	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
STID/EXID[31:21]	**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。
EXID[20:3]	**扩展标识符 (Extended identifier)** 扩展标识符的低字节。
IDE	**标识符选择 (Identifier extension)** 该位决定接收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。
RTR	**远程发送请求 (Remote transmission request)** 0：数据帧； 1：远程帧。

注：所有接收邮箱寄存器都是只读的。

13.7.6 接收 FIFO 邮箱数据长度和时间戳寄存器 (CAN_RDTxR) (x=0..1)

接收 FIFO 邮箱数据长度和时间戳寄存器 (CAN_RDTxR) (x=0..1)			基地址：0x4000 6400(CAN1) 偏移地址：0x1B4 0x1C4					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	TIME[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read:	TIME[23:16]							

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	FMI[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	DLC[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
TIME[31:16]	**报文时间戳 (Message time stamp)** 该域包含了，在接收该报文 SOF 的时刻，16 位定时器的值。
FMI[15:8]	**过滤器匹配序号 (Filter match index)** 这里是存在邮箱中的信息传送的过滤器序号。关于标识符过滤的细节，请参考 22.7.4 中有关过滤器匹配序号。
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。

注：所有接收邮箱寄存器都是只读的。

13.7.7 接收 FIFO 邮箱低字节数据寄存器 (CAN_RDLxR) (x=0..1)

接收 FIFO 邮箱低字节数据寄存器 (CAN_RDLxR) (x=0..1)			基地址：0x4000 6400(CAN1) 偏移地址：0x1B8 0x1C8					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	DATA3[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	DATA2[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DATA1[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DATA0[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DATA3[31:24]	**数据字节3 (Data byte 3)** 报文的数据字节 3。
DATA2[23:16]	**数据字节2 (Data byte 2)** 报文的数据字节 2。
DATA1[15:8]	**数据字节1 (Data byte 1)** 报文的数据字节 1。
DATA0[7:0]	**数据字节0 (Data byte 0)** 报文的数据字节0。 报文包含 0 到 8 个字节数据，且从字节 0 开始。

注：所有接收邮箱寄存器都是只读的。

13.7.8 接收 FIFO 邮箱高字节数据寄存器 (CAN_RDHxR) (x=0..1)

接收 FIFO 邮箱高字节数据寄存器 (CAN_RDHxR) (x=0..1)			基地址：0x4000 6400(CAN1) 偏移地址：0x1BC 0x1CC					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	DATA7[31:24]							
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	DATA6[23:16]							
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DATA5[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DATA4[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DATA7[31:24]	**数据字节7 (Data byte 7)** 报文的数据字节 7
DATA6[23:16]	**数据字节6 (Data byte 6)** 报文的数据字节 6。
DATA5[15:8]	**数据字节5 (Data byte 5)** 报文的数据字节 5。
DATA4[7:0]	**数据字节4 (Data byte 4)** 报文的数据字节 4。

注：所有接收邮箱寄存器都是只读的。

13.8 CAN 过滤器寄存器

13.8.1 CAN 过滤器主控寄存器 (CAN_FMR)

CAN 过滤器主控寄存器 (CAN_FMR)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x200					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	1	0	1	0	1	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	1	1	1	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	1	1	1	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	FINIT
Reset:	0	0	0	0	0	0	0	1

位	功能描述
FINIT	**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。

注:该寄存器的非保留位完全由软件控制,剩余保留位强制为复位值。

13.8.2 CAN 过滤器模式寄存器 (CAN_FM1R)

CAN 过滤器模式寄存器 (CAN_FM1R)			基地址：0x4000 6400(CAN1)					
			偏移地址：0x204					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X

Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	FBM13	FBM12	FBM11	FBM10	FBM9	FBM8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FBM7	FBM6	FBM5	FBM4	FBM3	FBM2	FBM1	FBM0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FBM[13:0]	**FBMx：** 过滤器模式 (Filter mode) 过滤器组x的工作模式。 0: 过滤器组x的2个32位寄存器工作在标识符屏蔽位模式； 1: 过滤器组x的2个32位寄存器工作在标识符列表模式。

注：只有在设置 CAN_FMR(FINIT=1)，使过滤器处于初始化模式下，才能对该寄存器写入。

13.8.3 CAN 过滤器位宽寄存器 (CAN_FS1R)

CAN 过滤器位宽寄存器 (CAN_FS1R)			基地址：0x4000 6400(CAN1) 偏移地址：0x20C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	FSC13	FSC12	FSC11	FSC10	FSC9	FSC8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FSC7	FSC6	FSC5	FSC4	FSC3	FSC2	FSC1	FSC0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FSC [13:0]	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。

注：只有在设置 CAN_FMR(FINIT=1)，使过滤器处于初始化模式下，才能对该寄存器写入。

13.8.4 CAN 过滤器 FIFO 关联寄存器 (CAN_FFA1R)

CAN 过滤器 FIFO 关联寄存器 (CAN_FFA1R)			基地址：0x4000 6400(CAN1) 偏移地址：0x214					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	FFA13	FFA12	FFA11	FFA10	FFA9	FFA8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FFA7	FFA6	FFA5	FFA4	FFA3	FFA2	FFA1	FFA0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FFA[13:0]	**过滤器位宽设置 (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 FIF01

注：只有在设置 CAN_FMR(FINIT=1)，使过滤器处于初始化模式下，才能对该寄存器写入。

13.8.5 CAN 过滤器激活寄存器 (CAN_FA1R)

CAN 过滤器激活寄存器 (CAN_FA1R)			基地址：0x4000 6400(CAN1) 偏移地址：0x21C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	FACT13	FACT12	FACT11	FACT10	FACT9	FACT8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FACT7	FACT6	FACT5	FACT4	FACT3	FACT2	FACT1	FACT0
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FACT[13:0]	过滤器激活 (Filter active) 软件对某位设置'1'来激活相应的过滤器。只有对FACTx位清'0'，或对CAN_FMR寄存器的FINIT位设置'1'后，才能修改相应的过滤器寄存器 x(CAN_FxR[0:1])。 0：过滤器被禁用； 1：过滤器被激活。

13.8.6 CAN 过滤器组 i 的寄存器 x (CAN_FiRx) (i＝0..13；x=1..2)

CAN 过滤器组 i 的寄存器 x (CAN_FiRx) (i＝0..13；x=1..2)			基地址：0x4000 6400(CAN1) 偏移地址：0x240h…0x31C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	FB13	FB12	FB11	FB10	FB9	FB8
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0
Reset:	0	0	0	0	0	0	0	0

位	功能描述

FB[13:0]	**过滤器位 (Filter bits)** **标识符模式** 寄存器的每位对应于所期望的标识符的相应位的电平。 0: 期望相应位为显性位； 1: 期望相应位为隐性位。 **屏蔽位模式** 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。 0: 不关心，该位不用于比较； 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。

注:

1.共有 14 组过滤器：i=0..13。每组过滤器由 2 个 32 位的寄存器，CAN_FiR[2:1]组成。

2.只有在 CAN_FAxR 寄存器相应的 FACTx 位清'0'，或 CAN_FMR 寄存器的 FINIT 位为'1'时，才能修改相应的过滤器寄存器。

3. 屏蔽位模式下的屏蔽/标识符寄存器，跟标识符列表模式下的寄存器位定义相同。

14 通用同步异步收发器(USART)

14.1 USART 介绍

通用同步异步收发器(USART)提供了一种灵活的方法与使用工业标准 NRZ 异步串行数据格式的外部设备之间进行全双工数据交换。USART 利用分数波特率发生器提供宽范围的波特率选择。

它支持同步单向通信和半双工单线通信，也支持 LIN(局部互连网)，智能卡协议和 IrDA(红外数据组织)SIR ENDEC 规范，以及调制解调器(CTS/RTS)操作。它还允许多处理器通信。

使用多缓冲器配置的 DMA 方式，可以实现高速数据通信。

14.2 USART 主要特性

- 全双工的，异步通信
- NRZ标准格式
- 分数波特率发生器系统
  - 发送和接收共用的可编程波特率，最高达4.5Mbits/s
- 可编程数据字长度(8位或9位)
- 可配置的停止位-支持1或2个停止位
- LIN主发送同步断开符的能力以及LIN从检测断开符的能力
  - 当USART硬件配置成LIN时，生成13位断开符；检测10/11位断开符
- 发送方为同步传输提供时钟
- IRDA SIR 编码器解码器
  - 在正常模式下支持3/16位的持续时间
- 智能卡模拟功能
  - 智能卡接口支持ISO7816-3标准里定义的异步智能卡协议
  - 智能卡用到的0.5和1.5个停止位
- 单线半双工通信
- 可配置的使用DMA的多缓冲器通信
  - 在SRAM里利用集中式DMA缓冲接收/发送字节
- 单独的发送器和接收器使能位
- 检测标志
  - 接收缓冲器满
  - 发送缓冲器空
  - 传输结束标志
- 校验控制
  - 发送校验位
  - 对接收数据进行校验
- 四个错误检测标志

- 溢出错误
- 噪音错误
- 帧错误
- 校验错误

● 10个带标志的中断源

- CTS改变
- LIN断开符检测
- 发送数据寄存器空
- 发送完成
- 接收数据寄存器满
- 检测到总线为空闲
- 溢出错误
- 帧错误
- 噪音错误
- 校验错误

● 多处理器通信 -- 如果地址不匹配，则进入静默模式

● 从静默模式中唤醒(通过空闲总线检测或地址标志检测)

● 两种唤醒接收器的方式：地址位(MSB，第9位)，总线空闲

14.3 USART 功能概述

接口通过三个引脚与其他设备连接在一起。任何 USART 双向通信至少需要两个脚：接收数据输入(RX)和发送数据输出(TX)。

RX：接收数据串行输。通过过采样技术来区别数据和噪音，从而恢复数据。

TX：发送数据输出。当发送器被禁止时，输出引脚恢复到它的 I/O 端口配置。当发送器被激活， 并且不发送数据时，TX 引脚处于高电平。在单线和智能卡模式里，此 I/O 口被同时用于数据的发送和接收。

● 总线在发送或接收前应处于空闲状态

● 一个起始位

● 一个数据字(8或9位)，最低有效位在前

● 0.5，1.5，2个的停止位，由此表明数据帧的结束

● 使用分数波特率发生器 —— 12位整数和4位小数的表示方法。

● 一个状态寄存器(USART_SR)

● 数据寄存器(USART_DR)

● 一个波特率寄存器(USART_BRR)，12位的整数和4位小数

● 一个智能卡模式下的保护时间寄存器(USART_GTPR)

在同步模式中需要下列引脚：

● **CK**：发送器时钟输出。此引脚输出用于同步传输的 时钟， (在Start位和Stop位上没有时钟脉冲，软件可选地，可以在最后一个数据位送出一个时钟脉冲)。数据可以在RX上同步被接收。这可以用来控制带有移位寄存器的外部设备(例如LCD驱动器)。时钟相位和极性都是软件可编程的。在智能卡模式里，CK可以为智能卡提供时钟。

在 IrDA 模式里需要下列引脚：

- **IrDA_RDI:** IrDA模式下的数据输入。
- **IrDA_TDO:** IrDA模式下

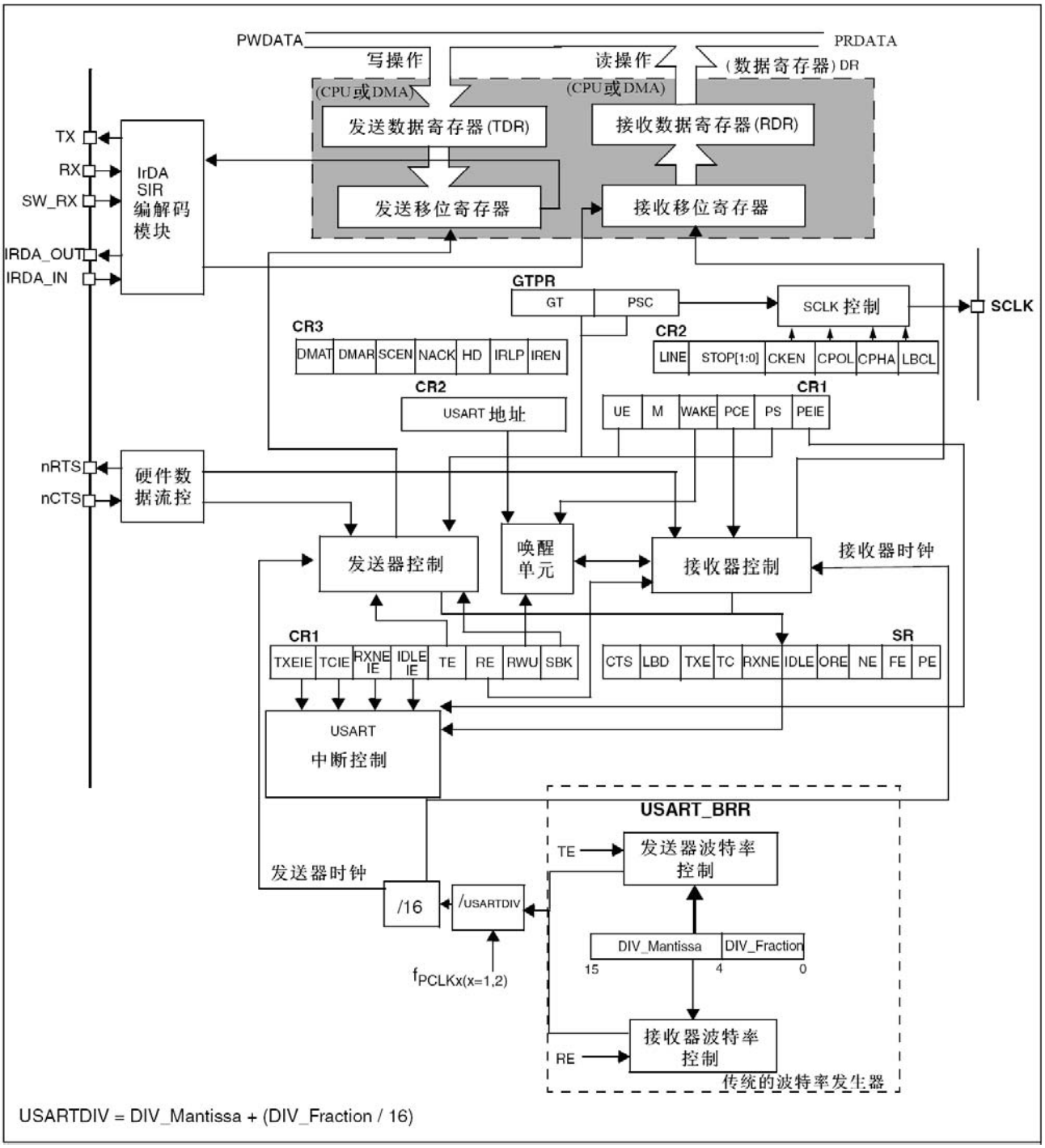
的数据输出。下列引脚在硬

件流控模式中需要：

- **nCTS:** 清除发送，若是高电平，在当前数据传输结束时阻断下一次的数据发送。
- **nRTS:** 发送请求，若是低电平，表明USART准

备好接收数图 :

USART框图



14.4 发送器

发送器根据 M 位的状态发送 8 位或 9 位的数据字。当发送使能位(TE)被设置时，发送移位寄存器中的数据在 TX 脚上输出，相应的时钟脉冲在 CK 脚上输出。

14.4.1 字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式里，USART_DR 寄存器包含了一个内部总线和发送移位寄存器之间的缓冲器(见图 248)。

每个字符之前都有一个低电平的起始位；之后跟着的停止位，其数目可配置。

USART 支持多种停止位的配置：0.5、1、1.5 和 2 个停止位。

注：1．在数据传输期间不能复位 TE 位，否则将破坏 TX 脚上的数据，因为波特率计数器停止计数。

正在传输的当前数据将丢失。

2． TE 位被激活后将发送一个空闲帧。

14.4.2 可配置的停止位

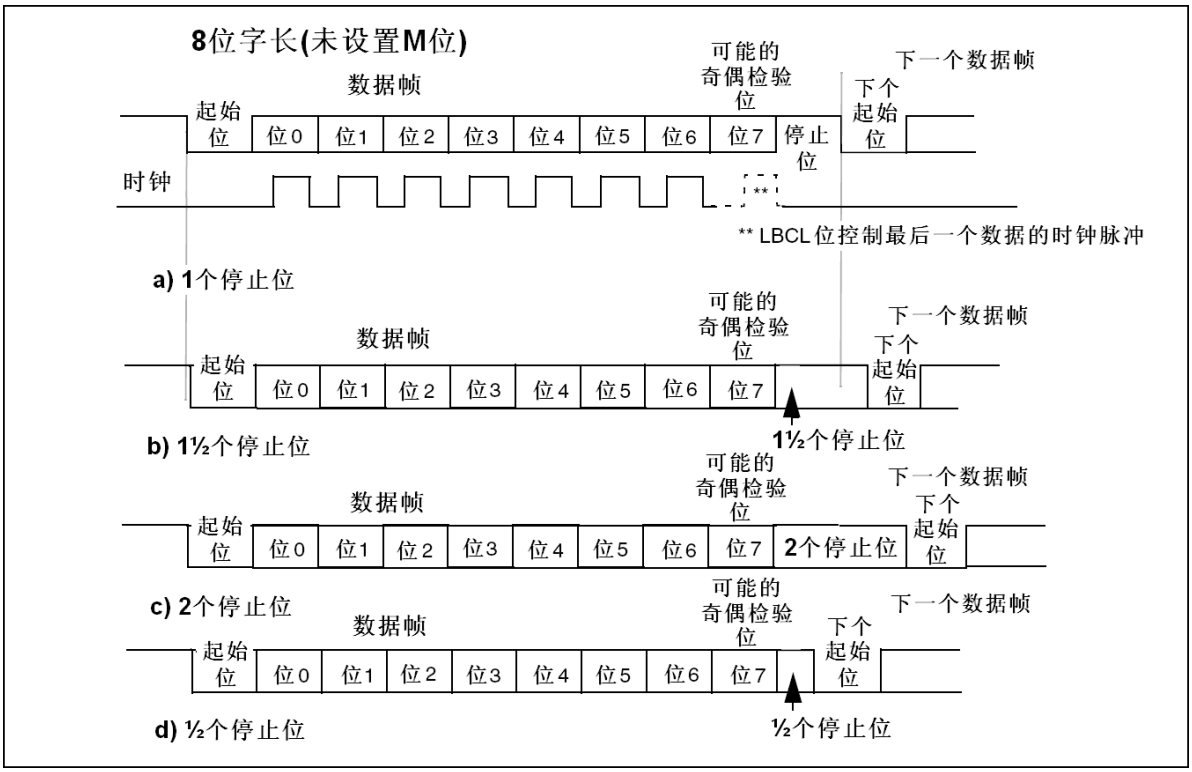
随每个字符发送的停止位的位数可以通过控制寄存器 2 的位 13、12 进行编程。

1. 1个停止位：停止位位数的默认值。
2. 2个停止位：可用于常规USART模式、单线模式以及调制解调器模式。
3. 0.5个停止位：在智能卡模式下接收数据时使用。
4. 1.5个停止位：在智能卡模式下发送和接收

数据时使用。空闲帧包括了停止位。

断开帧是 10 位低电平，后跟停止位(当 m=0 时)；或者 11 位低电平，后跟停止位(m=1 时)。不可能传输更长的断开帧(长度大于 10 或者 11 位)。

图：配置停止位



配置步骤：

1. 通过在USART_CR1寄存器上置位UE位来激活USART
2. 编程USART_CR1的M位来定义字长。
3. 在USART_CR2中编程停止位的位数。
4. 如果采用多缓冲器通信，配置USART_CR3中的DMA使能位(DMAT)。按多缓冲器通信中的描述配置DMA寄存器。
5. 利用USART_BRR寄存器选择要求的波特率。
6. 设置USART_CR1中的TE位，发送一个空闲帧作为第一次数据发送。
7. 把要发送的数据写进USART_DR寄存器(此动作清除TXE位)。在只有一个缓冲器的情况下，对每个待发送的数据重复步骤7。
8. 在USART_DR寄存器中写入最后一个数据字后，要等待TC=1，它表示最后一个数据帧的传输结束。当需要关闭USART或需要进入停机模式之前，需要确认传输结束，避免破坏最后一次传输。

14.4.3单字节通信

清零 TXE 位总是通过对数据寄存器的写操作来完成的。TXE 位由硬件来设置，它表明：

- 数据已经从TDR移送到移位寄存器，数据发送已经开始
- TDR寄存器被清空
- 下一个数据可以被写进USART_DR寄存器而不会覆

盖先前的数据如果TXEIE位被设置，此标志将产生一个

中断。

如果此时 USART 正在发送数据，对 USART_DR 寄存器的写操作把数据存进 TDR 寄存器，并在当前传输结束时把该数据复制进移位寄存器。

如果此时 USART 没有在发送数据，处于空闲状态，对 USART_DR 寄存器的写操作直接把数据放进移位寄存器，数据传输开始，TXE 位立即被置起。

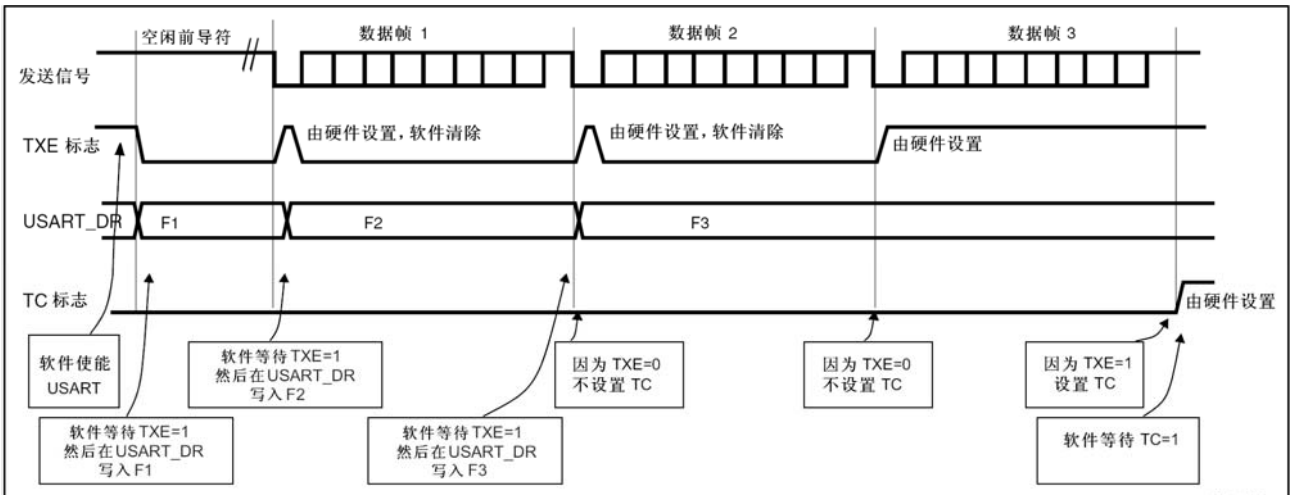
当一帧发送完成时(停止位发送后)并且设置了 TXE 位，TC 位被置起，如果 USART_CR1 寄存器中的 TCIE 位被置起时，则会产生中断。在 USART_DR 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式(详见下图)之前，必须先等待 TC=1。

使用下列软件过程清除 TC 位：

1. 读一次 USART_SR 寄存器；
2. 写一次 USART_DR 寄存器。

注：TC 位也可以通过软件对它写’0’来清除。此清零方式只推荐在多缓冲器通信模式下使用。

图：发送时 TC/TXE 的变化情况



14.4.4断开符号

设置 SBK 可发送一个断开符号。断开帧长度取决 M 位(见图 249)。如果设置 SBK=1，在完成当前数据发送后，将在 TX 线上发送一个断开符号。断开字符发送完成时(在断开符号的停止位时)SBK 被硬件复位。USART 在最后一个断开帧的结束处插入一逻辑’1’，以保证能识别下一帧的起始位。

注意：如果在开始发送断开帧之前，软件又复位了 SBK 位，断开符号将不被发送。如果要发送两个连续的断开帧，SBK 位应该在前一个断开符号的停止位之后置起。

14.4.5空闲符号

置位 TE 将使得 USART 在第一个数据帧前发送一空闲帧。

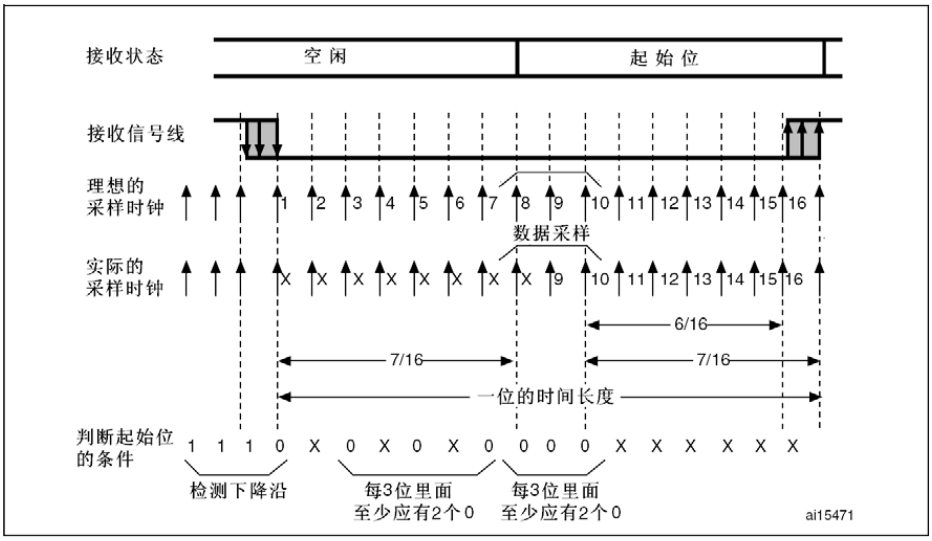
14.5 接收器

USART 可以根据 USART_CR1 的 M 位接收 8 位或 9 位的数据字。

14.5.1起始位侦测

在 USART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。该序列为：1 1 1 0 X 0 X 0 X 0 0 0

图：起始位侦测



注：如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态(不设置标志位)等待下降沿。

如果 3 个采样点都为'0'(在第 3、5、7 位的第一次采样，和在第 8、9、10 的第二次采样都为'0')， 则确认收到起始位，这时设置 RXNE 标志位，如果 RXNEIE=1，则产生中断。

如果两次 3 个采样点上仅有 2 个是'0'(第 3、5、7 位的采样点和第 8、9、10 位的采样点)，那么起始位仍然是有效的，但是会设置 NE 噪声标志位。如果不能满足这个条件，则中止起始位的侦测过程，接收器会回到空闲状态(不设置标志位)。

如果有一次 3 个采样点上仅有 2 个是'0'(第 3、5、7 位的采样点或第 8、9、10 位的采样点)，那么起始位仍然是有效的，但是会设置 NE 噪声标志位。

14.5.2字符接收

在 USART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里，USART_DR 寄存器包含的缓冲器位于内部总线和接收移位寄存器之间。

配置步骤：

1. 将USART_CR1寄存器的UE置1来激活USART。
2. 编程USART_CR1的M位定义字长
3. 在USART_CR2中编写停止位的个数
4. 如果需多缓冲器通信，选择USART_CR3中的DMA使能位(DMAR)。按多缓冲器通信所要求的配置DMA寄存器。
5. 利用波特率寄存器USART_BRR选择希望的波特率。
6. 设置USART_CR1的RE位。激活接收器，使它开始

寻找起始位。当一字符被接收到时，

- RXNE位被置位。它表明移位寄存器的内容被转移到RDR。换句话说，数据已经被接收并且可以被读出(包括与之有关的错误标志)。
- 如果RXNEIE位被设置，产生中断。
- 在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起，
- 在多缓冲器通信时，RXNE在每个字节接收后被置起，并由DMA对数据寄存器的读操作而清零。
- 在单缓冲器模式里，由软件读USART_DR寄存器完成对RXNE位清除。RXNE标志也可以通过对它写0来清除。RXNE位必须在下一字符接收结束前被清零，以避免溢出错误。

  注：在接收数据时，RE位不应该被复位。如果RE位在接收时被清零，当前字节的接收被丢失。

14.5.3断开符号

当接收到一个断开帧时，USART 像处理帧错误一样处理它。

14.5.4空闲符号

当一空闲帧被检测到时，其处理步骤和接收到普通数据帧一样，但如果 IDLEIE 位被设置将产生一个中断。

14.5.5溢出错误

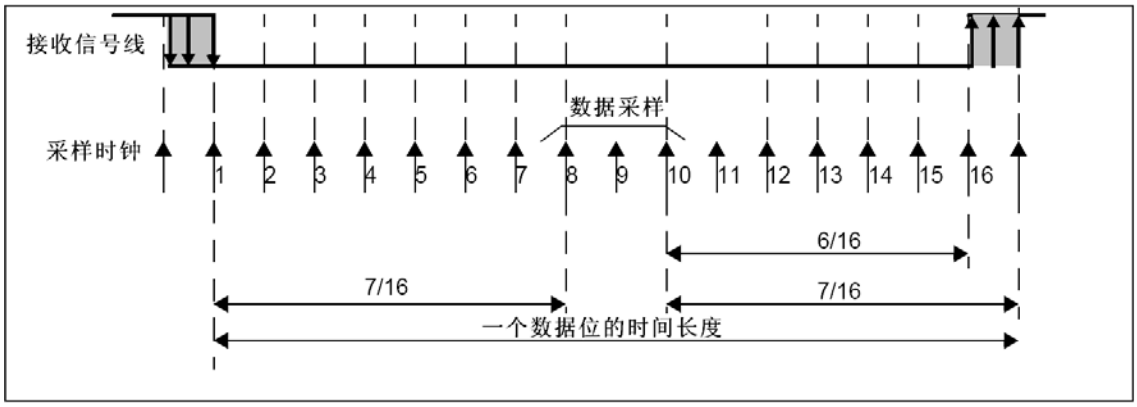
如果 RXNE 还没有被复位，又接收到一个字符，则发生溢出错误。数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。RXNE 标记是接收到每个字节后被置位的。如果下一个数据已被收到或先前 DMA 请求还没被服务时，RXNE 标志仍是置起的，溢出错误产生。

当溢出错误产生时：

- ORE位被置位。
- RDR内容将不会丢失。读USART_DR寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。随后接收到的数据都将丢失。
- 如果RXNEIE位被设置或EIE和DMAR位都被设置，中断产生。
- 顺序执行对USART_SR和USART_DR寄存器的读操作，可复位ORE位
- 注意： 当ORE位置位时，表明至少有1个数据已经丢失。有两种可能性：
- 如果RXNE=1，上一个有效数据还在接收寄存器RDR上，可以被读出。
- 如果RXNE=0，这意味着上一个有效数据已经被读走，RDR已经没有东西可读。当上一个有效数据在RDR中被读取的同时又接收到新的(也就是丢失的)数据时，此种情况可能发生。在读序列期间(在USART_SR寄存器读访问和USART_DR读访问之间)接收到新的数据，此种情况也可能发生。

14.5.6噪音错误

使用过采样技术(同步模式除外)，通过区别有效输入数据和噪音来进行数据恢复。图：检测噪声的数据采样



表：检测噪声的数据采样

采样值	NE状态	接收的位	数据有效性
000	0	0	有效
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效
111	0	1	有效

当在接收帧中检测到噪音时：

- 在RXNE位的上升沿设置NE标志。
- 无效数据从移位寄存器传送到USART_DR寄存器。
- 在单个字节通信情况下，没有中断产生。然而，因为NE标志位和RXNE标志位是同时被设置，RXNE将产生中断。在多缓冲器通信情况下，如果已经设置了USART_CR3寄存器中

EIE 位，将产生一个中断。

先读出 USART_SR，再读出 USART_DR 寄存器，将清除 NE 标志位

14.5.7 帧错误

当以下情况发生时检测到帧错误：

由于没有同步上或大量噪音的原因，停止位没有在预期的时间上接和收识别出来。当帧错误被检测到时：

- FE位被硬件置起
- 无效数据从移位寄存器传送到USART_DR寄存器。
- 在单字节通信时，没有中断产生。然而，这个位和RXNE位同时置起，后者将产生中断。在多缓冲器通信情况下，如果USART_CR3寄存器中EIE位被置位的话，将产生中断。

顺序执行对 USART_SR 和 USART_DR 寄存器的读操作，可复位 FE 位。

14.5.8接收期间的可配置的停止位

被接收的停止位的个数可以通过控制寄存器 2 的控制位来配置，在正常模式时，可以是 1 或 2 个， 在智能卡模式里可能是 0.5 或 1.5 个。

1. 0.5个停止位(智能卡模式中的接收)：不对0.5个停止位进行采样。因此，如果选择0.5个停止位则不能检测帧错误和断开帧。
2. 1个停止位：对1个停止位的采样在第8，第9和第10采样点上进行。
3. 1.5个停止位(智能卡模式)：当以智能卡模式发送时，器件必须检查数据是否被正确的发送出去。所以接收器功能块必须被激活(USART_CR1寄存器中的RE =1)，并且在停止位的发送期间采样数据线上的信号。如果出现校验错误，智能卡会在发送方采样NACK信号时，即总线上停止位对应的时间内时，拉低数据线，以此表示出现了帧错误。FE在1.5 个停止位结束时和 RXNE 一起被置起。对 1.5 个停止位的采样是在第 16，第 17 和第 18 采样点进行的。1.5 个的停止位可以被分成 2 部分：一个是 0.5 个时钟周期，期间不做任何事情 。随后是 1 个时钟周期的停止位，在这段时间的中点处采样。详见第 25.3.11 节：智能卡。
4. 2个停止位：对2个停止位的采样是在第一停止位的第8，第9和第10个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被设置。第二个停止位不再检查帧错误。在第一个停止位结束时RXNE标志将被设置。

14.6 USART 接收器容忍时钟的变化

只有当整体的时钟系统地变化小于 USART 异步接收器能够容忍的范围，USART 异步接收器才能正常地工作。影响这些变化的因素有：

- DTRA：由于发送器误差而产生的变化(包括发送器端振荡器的变化)
- DQUANT：接收器端波特率取整所产生的误差
- DREC：接收器端振荡器的变化
- DTCL：由于传输线路产生的变化(通常是由于收发器在由低变高的转换时序，与由高变低转换时序之间的不一致性所造成)。

需要满足：DTRA + DQUANT + DREC + DTCL < USART 接收器的容忍度

对于正常接收数据，USART 接收器的容忍度等于最大能容忍的变化，它依赖于下述选择：

- 由USART_CR1寄存器的M位定义的10或11位字符长度
- 是否使用分数波特率产生

表：当 DIV_Fraction=0 时，USART 接收器的容忍度

M位	认为NF是错误	不认为NF是错误
0	3.75%	4.375%
1	3.41%	3.97%

表：当 DIV_Fraction!=0 时，USART 接收器的容忍度

M位	认为NF是错误	不认为NF是错误
0	3.33%	3.88%
1	3.03%	3.53%

注：在特殊的情况下，即当收到的帧包含一些在 M=0 时，正好是 10 位(M=1 时是 11 位)的空闲帧，上面 2 个表格中的数据可能会有少许出入。

14.7 多处理器通信

通过 USART 可以实现多处理器通信(将几个 USART 连在一个网络里)。例如某个 USART 设备可以是主，它的 TX 输出和其他 USART 从设备的 RX 输入相连接；USART 从设备各自的 TX 输出逻辑地与在一起，并且和主设备的 RX 输入相连接。

在多处理器配置中，我们通常希望只有被寻址的接收者才被激活，来接收随后的数据，这样就可以减少由未被寻址的接收器的参与带来的多余的 USART 服务开销。

未被寻址的设备可启用其静默功能置于静默模式。在静默模式里：

- 任何接收状态位都不会被设置。
- 所有接收中断被禁止。
- USART_CR1寄存器中的RWU位被置1。RWU可以被硬件自动控制或在某个条件下由软件写入。

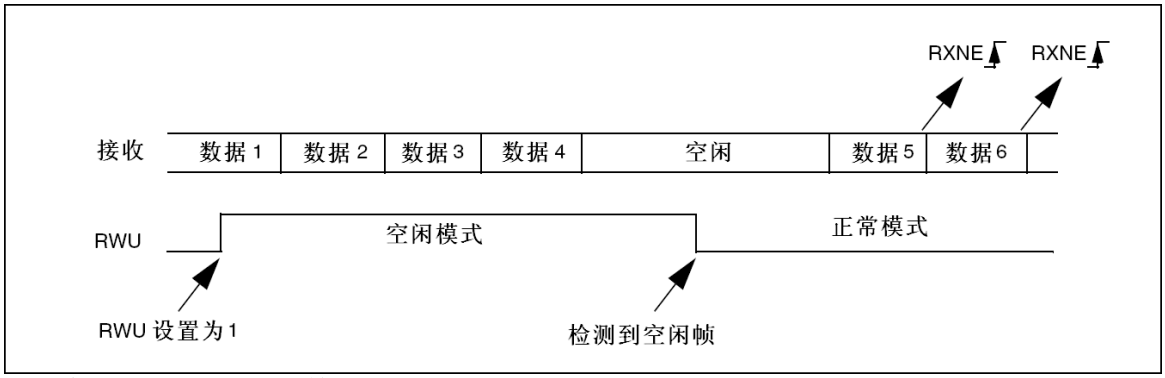
根据 USART_CR1 寄存器中的 WAKE 位状态，USART 可以用二种方法进入或退出静默模式。

- 如果WAKE位被复位：进行空闲总线检测。
- 如果WAKE位被设置：进行地址标记检测。

空闲总线检测(WAKE=0)

当 RWU 位被写 1 时，USART 进入静默模式。当检测到一空闲帧时，它被唤醒。然后 RWU 被硬件清零，但是 USART_SR 寄存器中的 IDLE 位并不置起。RWU 还可以被软件写 0。下图给出利用空闲总线检测来唤醒和进入静默模式的一个例子

图：利用空闲总线检测的静默模式



地址标记(address mark)检测(WAKE=1)

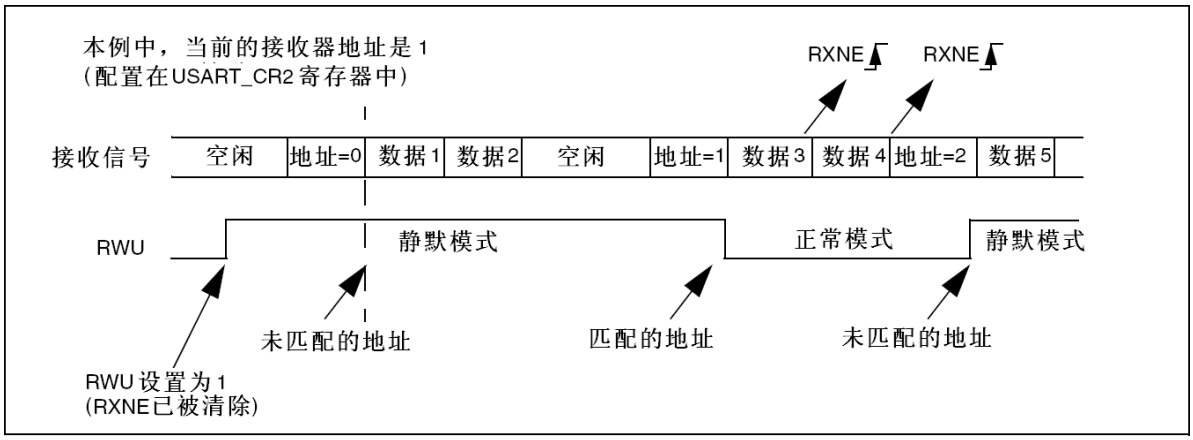
在这个模式里，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。在一个地址字节中， 目标接收器的地址被放在 4 个 LSB 中。这个 4 位地址被接收器同它自己地址做比较，接收器的地址被编程在 USART_CR2 寄存器的 ADD。

如果接收到的字节与它的编程地址不匹配时，USART 进入静默模式。此时，硬件设置 RWU 位。接收该字节既不会设置 RXNE 标志也不会产生中断或发出 DMA 请求，因为 USART 已经在静默模式。

当接收到的字节与接收器内编程地址匹配时，USART 退出静默模式。然后 RWU 位被清零，随后的字节被正常接收。收到这个匹配的地址字节时将设置 RXNE 位，因为 RWU 位已被清零。

当接收缓冲器不包含数据时(USART_SR 的 RXNE=0)，RWU 位可以被写 0 或 1。否则，该次写操作被忽略。下图给出利用地址标记检测来唤醒和进入静默模式的例子。

图：利用地址标记检测的静默模式



14.8 校验控制

设置 USART_CR1 寄存器上的 PCE 位，可以使能奇偶控制(发送时生成一个奇偶位，接收时进行奇偶校验)。根据 M 位定义的帧长度，可能的 USART 帧格式列在下表中。

表：帧格式

M位	PCE位	USART帧
0	0	| 起始位 | 8位数据 | 停止位 |
0	1	| 起始位 | 7位数据 | 奇偶检验位 | 停止位 |
1	0	| 起始位 | 9位数据 | 停止位 |
1	1	| 起始位 | 8位数据 | 奇偶检验位 | 停止位 |

偶校验：校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中'1'的个数为偶数。

例如：数据=00110101，有 4 个'1'，如果选择偶校验(在 USART_CR1 中的 PS＝0)，校验位将是'0'。

奇校验：此校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中'1'的个数为奇数。

例如：数据=00110101，有 4 个'1'，如果选择奇校验(在 USART_CR1 中的 PS＝1)，校验位将是'1'。

传输模式：如果 USART_CR1 的 PCE 位被置位，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去(如果选择偶校验偶数个'1'，如果选择奇校验奇数个'1')。如果奇偶校验失败， USART_SR 寄存器中的 PE 标志被置'1'，并且如果 USART_CR1 寄存器的 PEIE 在被预先设置的话，中断产生。

14.9 LIN(局域互联网)模式

LIN 模式是通过设置 USART_CR2 寄存器的 LINEN 位选择。在 LIN 模式下，下列位必须保持为 0：

- USART_CR2寄存器的CLKEN位
- USART_CR3寄存器的STOP[1:0]，SCEN，HDSEL和IREN

LIN发送

LIN 主发送，但和正常 USART 发送有以下区别：

- 清零M位以配置8位字长
- 置位LINEN位以进入LIN模式。这时，置位SBK将发送13位'0'作为断开符号。然后发一位'1'，以允许对下一个开始位的检测。

LIN接收

当 LIN 模式被使能时，断开符号检测电路被激活。该检测完全独立于 USART 接收器。断开符号只要一出现就能检测到，不管是在总线空闲时还是在发送某数据帧其间，数据帧还未完成，又插入了断开符号的发送。

当接收器被激活时(USART_CR1 的 RE=1)，电路监测 RX 上的起始信号。监测起始位的方法同检测断开符号或数据是一样的。当起始位被检测到后，电路对每个接下来的位，在每个位的第 8， 9 ，10 个过采样时钟点上进行采样。如果 10 个(当 USART_CR2 的 LBDL = 0) 或 11 个(当 USART_CR2 的 LBDL = 1)连续位都是'0'，并且又跟着一个定界符，USART_SR 的 LBD 标志被设置。如果 LBDIE 位=1，中断产生。在确认断开符号前，要检查定界符，因为它意味 RX 线已经回到高电平。

如果在第 10 或 11 个采样点之前采样到了'1'，检测电路取消当前检测并重新寻找起始位。如果 LIN 模式被禁止，接收器继续如正常 USART 那样工作，不需要考虑检测断开符号。

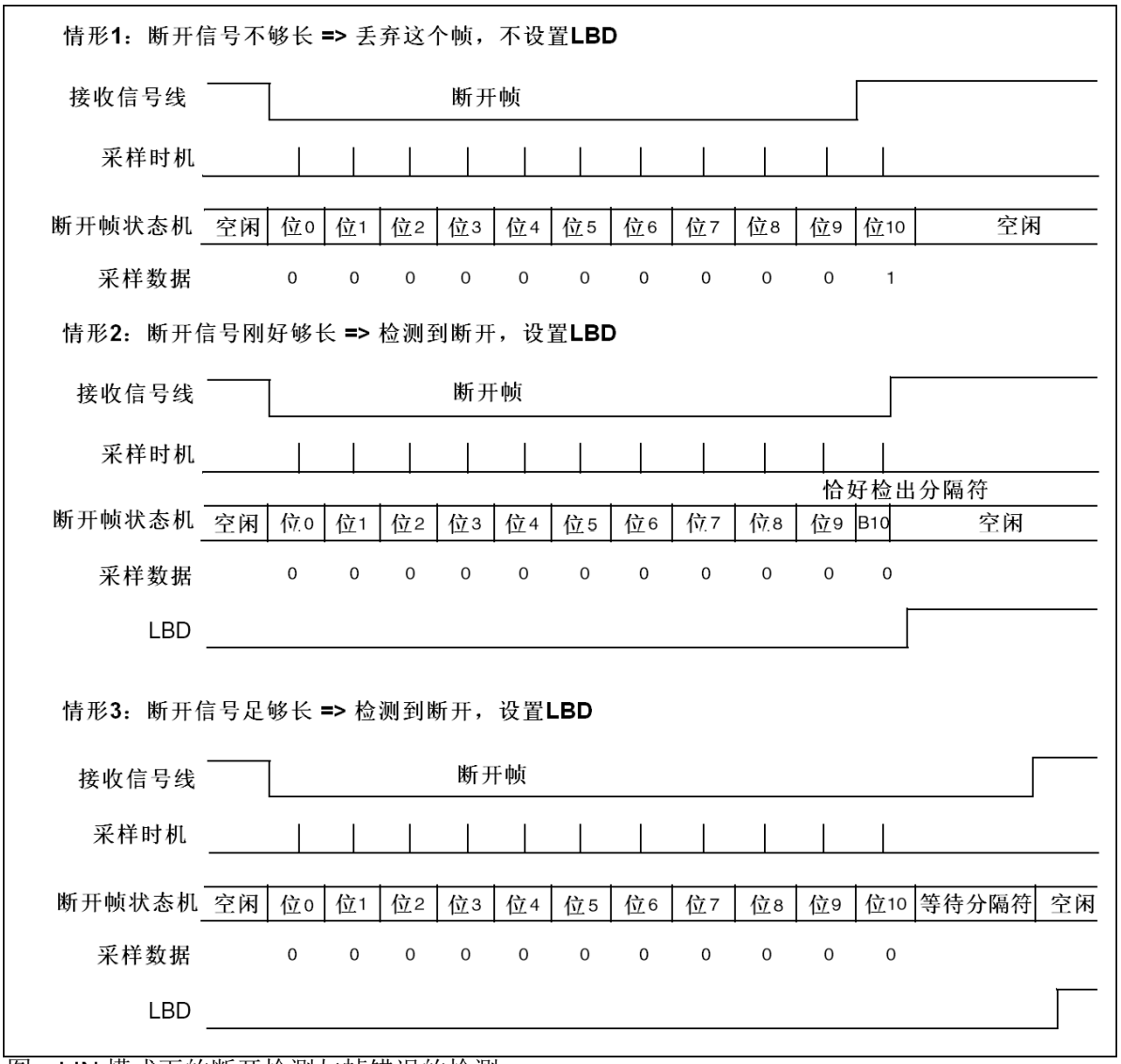
如果 LIN 模式没有被激活(LINEN=0)，接收器仍然正常工作于 USART 模式，不会进行断开检测。

如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误(也就是停止位检测到'0'，这种情况出现在断开帧)，接收器就停止，直到断开符号检测电路接收到一个'1'(这种情况发生于断开符号没有完整的发出来)，或一个定界符(这种情况发生于已经检测到一个完整的断开符号)。

下图说明了断开符号检测器状态机的行为和断开符号标

志的关系以及给出了一个断开帧的例子。

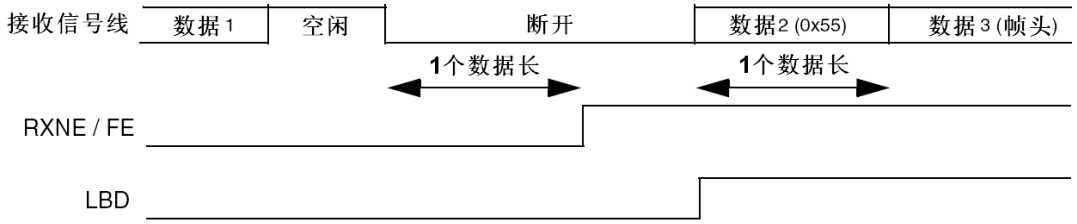
图：LIN 模式下的断开检测(11 位断开长度– 设置了 LBDL 位)



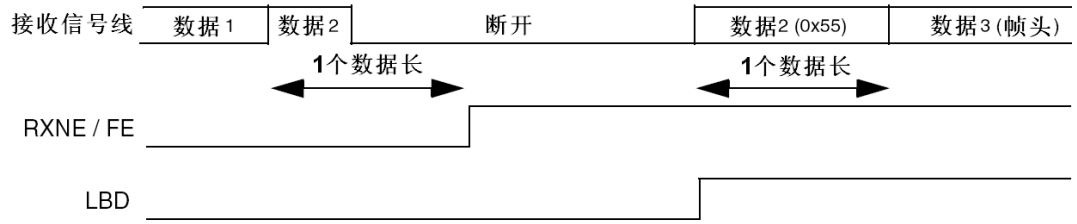
图：LIN 模式下的断开检测与帧错误的检测

下面的例子中，假定**LBDL=1**（断开帧长度为**11**位），**M=0**（**8**位数据）

情形**1**：断开发生在空闲之后



情形**2**：断开发生在正在接收数据时



14.10 USART 同步模式

通过在 USART_CR2 寄存器上写 CLKEN 位选择同步模式在同步模式里，下列位必须保持清零状态：

- USART_CR2寄存器中的LINEN位
- USART_CR3寄存器中的SCEN,HDSEL和IREN位

USART 允许用户以主模式方式控制双向同步串行通信。CK 脚是 USART 发送器时钟的输出。在起始位和停止位期间，CK 脚上没有时钟脉冲。根据 USART_CR2 寄存器中 LBCL 位的状态，决定在最后一个有效数据位期间产生或不产生时钟脉冲。USART_CR2 寄存器的 CPOL 位允许用户选择时钟极性，USART_CR2 寄存器上的 CPHA 位允许用户选择外部时钟的相位

在总线空闲期间，实际数据到来之前以及发送断开符号的时候，外部 CK 时钟不被激活。

同步模式时，USART 发送器和异步模式里工作一模一样。但是因为 CK 是与 TX 同步的(根据 CPOL 和 CPHA)，所以 TX 上的数据是随 CK 同步发出的。

同步模式的 USART 接收器工作方式与异步模式不同。如果 RE=1，数据在 CK 上采样(根据 CPOL 和 CPHA 决定在上升沿还是下降沿)，不需要任何的过采样。但必须考虑建立时间和持续时间(取决于波特率，1/16 位时间)。

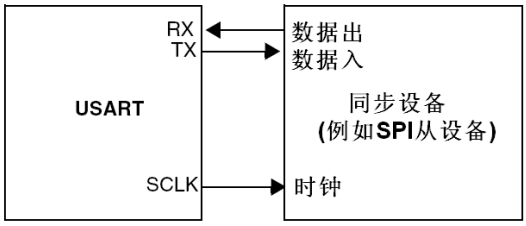
注意：1．CK 脚同 TX 脚一起联合工作。因而，只有在使能了发送器(TE＝1)，并且发送数据时(写入数据至 USART_DR 寄存器)才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。

2．LBCL,CPOL 和 CPHA 位的正确配置，应该在发送器和接收器都被禁止时；当使能了发送器或接收器时，这些位不能被改变

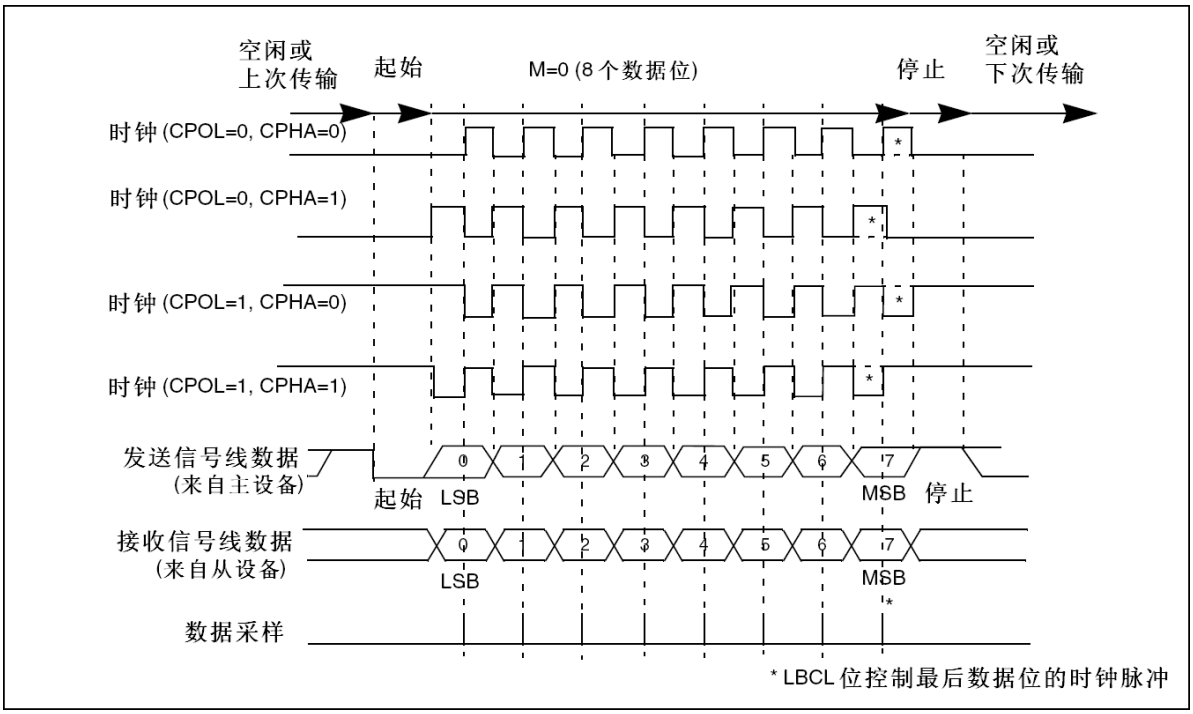
3．建议在同一条指令中设置 TE 和 RE，以减少接收器的建立时间和保持时间。

4． USART 只支持主模式：它不能用来自其他设备的输入时钟接收或发送数据(CK 永远是输出)。

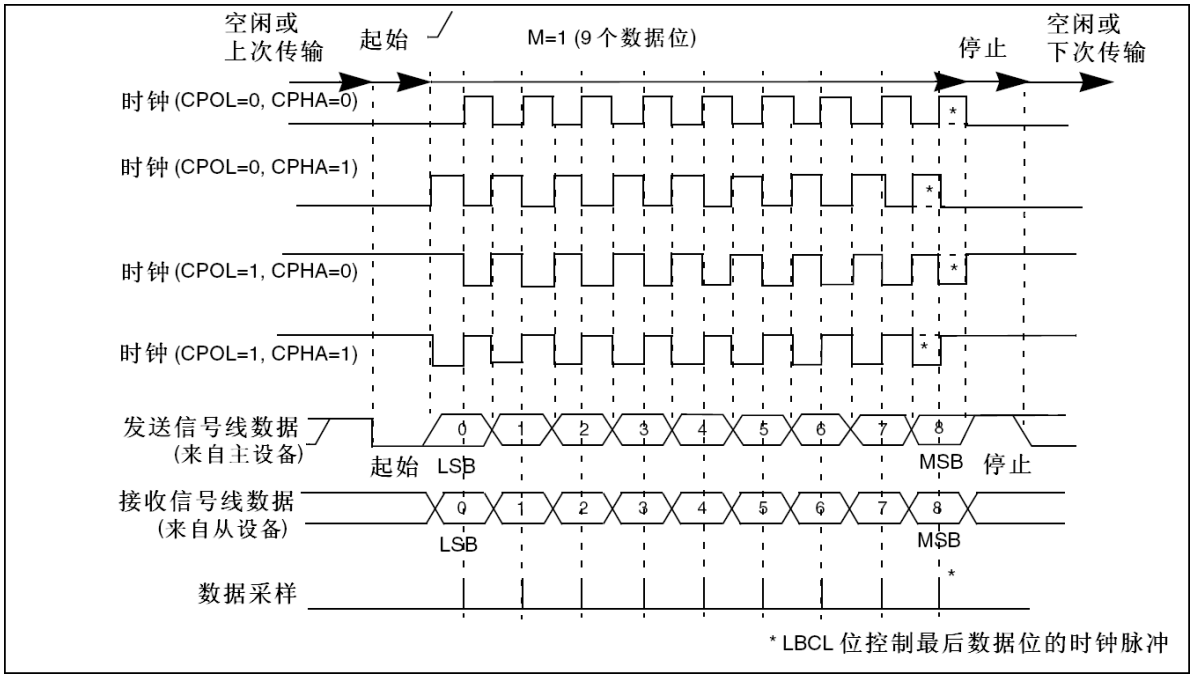
图：USART 同步传输的例子



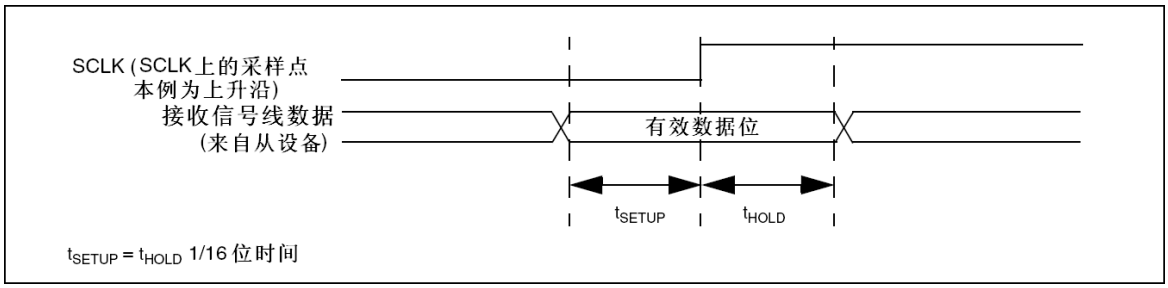
图：USART 数据时钟时序示例(M=0)



图：USART 数据时钟时序示例(M=1)



图：RX 数据采样/保持时间



注：　在智能卡模式下 CK 的功能不同，有关细节请参考智能卡模式部分。

14.11 单线半双工通信

单线半双方模式通过设置 USART_CR3 寄存器的 HDSEL 位选择。在这个模式里，下面的位必须保持清零状态：

- USART_CR2寄存器的LINEN和CLKEN位
- USART_CR3寄存器的SCEN和IREN位

USART 可以配置成遵循单线半双工协议。在单线半双工模式下，TX 和 RX 引脚在芯片内部互连。使用控制位"HALF DUPLEX SEL"(USART_CR3 中的 HDSEL 位)选择半双工和全双工通信。

当 HDSEL 为'1'时

- RX不再被使用
- 当没有数据传输时，TX总是被释放。因此，它在空闲状态的或接收状态时表现为一个标准 I/O 口。这就意味该 I/O 在不被 USART 驱动时，必须配置成悬空输入(或开漏的输出高)。

除此以外，通信与正常 USART 模式类似。由软件来管理线上的冲突(例如通过使用一个中央仲裁器)。特别的是，发送从不会被硬件所阻碍。当 TE 位被设置时，只要数据一写到数据寄存器上， 发送就继续。

14.12 智能卡

设置 USART_CR3 寄存器的 SCEN 位选择智能卡模式。在智能卡模式下，下列位必须保持清零：

- USART_CR2寄存器的LINEN位
- USART_CR3寄存器的HDSEL 位和IREN位

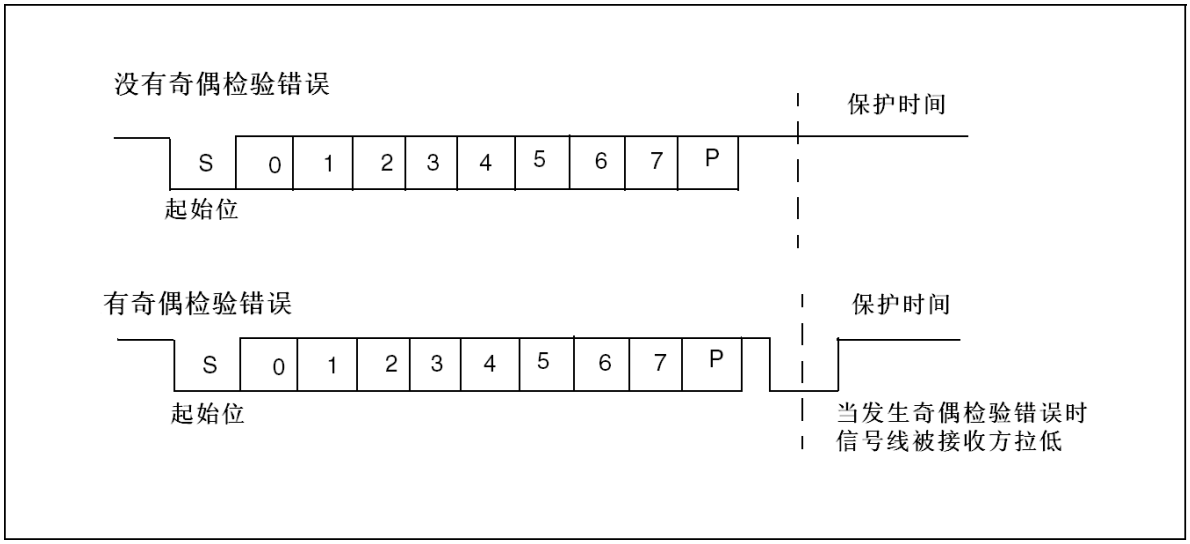
此外，CLKEN 位可以被设置，以提供时钟给智能卡。

该接口符合 ISO7816-3 标准，支持智能卡异步协议。USART 应该被设置为：

- 8位数据位加校验位： 此时USART_CR1寄存器中M=1、PCE=1
- 发送和接收时为1.5个停止位：即USART_CR2寄存器的STOP=11

注： 也可以在接收时选择0.5个停止位，但为了避免在2种配置间转换，建议在发送和接收时使用1.5 个停止位。

下图给出的例子说明了数据线上，在有校验错误和没校验错误两种情况下的信号。图: ISO7816-3 异步协议



当与智能卡相连接时，USART 的 TX 驱动一根智能卡也驱动的双向线。为了做到这点，SW_RX 必须和 TX 连接到相同的 I/O 口。在发送开始位和数据字节期间，发送器的输出使能位 TX_EN 被置起，在发送停止位期间被释放(弱上拉)，因此在发现校验错误的情况下接收器可以将数据线拉低。如果 TX_EN 不被使用，在停止位期间 TX 被拉到高电平：这样的话，只要 TX 配置成开漏，接收器也可以驱动这根线。

智能卡是一个单线半双工通信协议

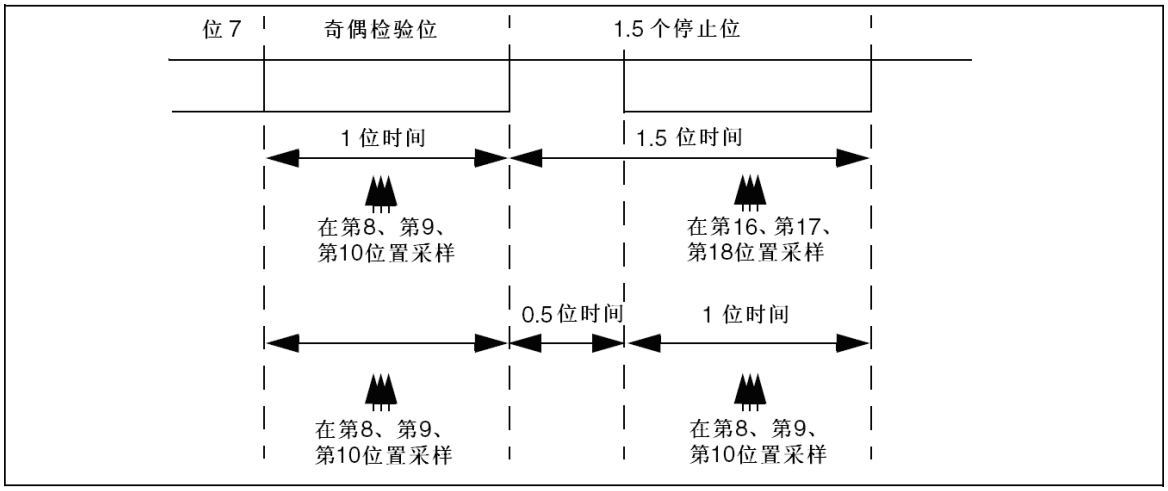
- 从发送移位寄存器把数据发送出去，要被延时最小1/2波特时钟。在正常操作时，一个满的发送移位寄存器将在下一个波特时钟沿开始向外移出数据。在智能卡模式里，此发送被延迟1/2波特时钟。
- 如果在接收一个设置为0.5或1.5个停止位的数据帧期间，检测到一奇偶校验错误，在完成接收该帧后(即停止位结束时)，发送线被拉低一个波特时钟周期。这是告诉智能卡发送到
  USART 的数据没有被正确地接收到。此 NACK 信号(拉低发送线一个波特时钟周期)在发送端将产生一个帧错误(发送端被配置成 1.5 个停止位)。应用程序可以根据协议处理重新发送数据。如果设置了 NACK 控制位，发生校验错误时接收器会给出一个 NACK 信号；否则就不会发送 NACK。
- TC标志的置起可以通过编程保护时间寄存器得以延时。在正常操作时，当发送移位寄存器变空并且没有新的发送请求出现时，TC被置起。在智能卡模式里，空的发送移位寄存器将触发保护时间计数器开始向上计数，直到保护时间寄存器中的值。TC在这段时间被强制拉低。当保护时间计数器达到保护时间寄存器中的值时，TC被置高。
- TC标志的撤销不受智能卡模式的影响。
- 如果发送器检测到一个帧错误(收到接收器的NACK信号)，发送器的接收功能模块不会把 NACK 当作起始位检测。根据 ISO 协议，接收到的 NACK 的持续时间可以是 1 或 2 波特时钟周期。
- 在接收器这边，如果一个校验错误被检测到，并且NACK被发送，接收器不会把NACK检测成起始位。

注意： 1. 断开符号在智能卡模式里没有意义。一个带帧错误的00h数据将被当成数据而不是断开符号。

2. 当来回切换 TE 位时，没有 IDLE 帧被发送。ISO 协议没有定义 IDLE 帧。

下图详述了 USART 是如何采样 NACK 信号的。在这个例子里，USART 正在发送数据，并且被配置成 1.5 个停止位。为了检查数据的完整性和 NACK 信号，USART 的接收功能块被激活。

图：使用 1.5 停止位检测奇偶检验错



USART 可以通过 CK 输出为智能卡提供时钟。在智能卡模式里，CK 不和通信直接关联，而是先通过一个 5 位预分频器简单地用内部的外设输入时钟来驱动智能卡的时钟。分频率在预分频寄存器 USART_GTPR 中配置。CK 频率可以从 $f_{CK}/2$ 到 $f_{CK}/62$，这里的 f_{CK} 是外设输入时钟。

14.13 USART 中断请求

表：USART 中断请求

中断事件	事件标志	使能位
发送数据寄存器空	TXE	TXEIE
CTS标志	CTS	CTSIE
发送完成	TC	TCIE
接收数据就绪可读	TXNE	TXNEIE
检测到数据溢出	ORE	
检测到空闲线路	IDLE	IDLEIE
奇偶检验错	PE	PEIE
断开标志	LBD	LBDIE
噪声标志，多缓冲通信中的溢出错误和帧错误	NE或ORT或FE	EIE(1)

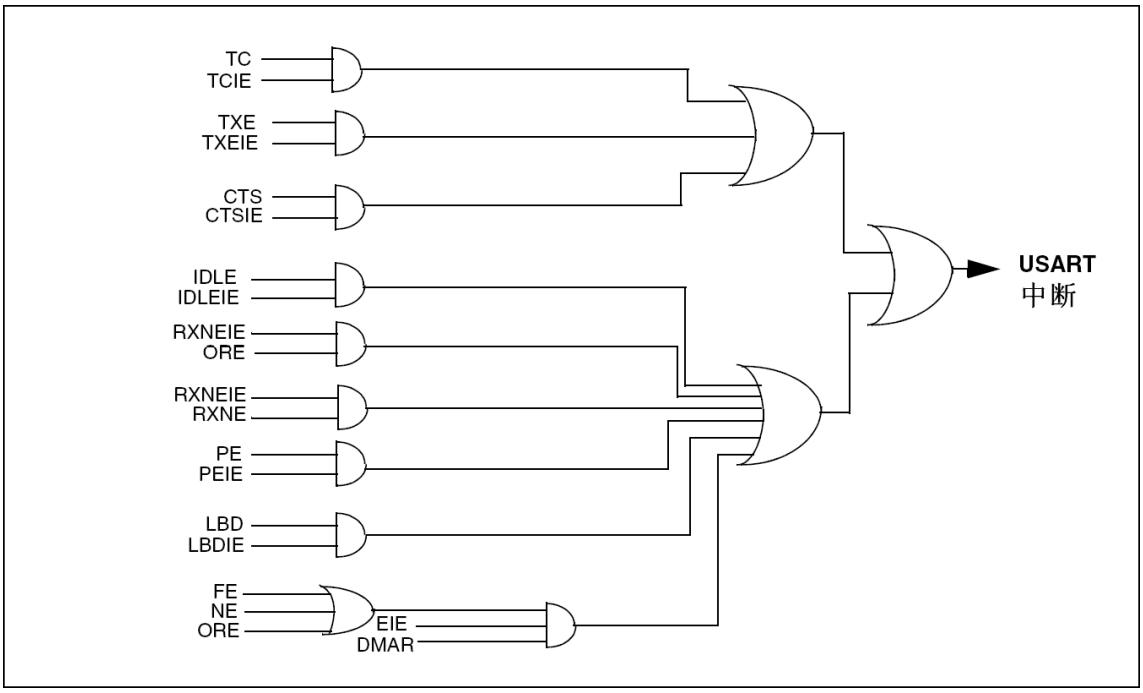
1．仅当使用 DMA 接收数据时，才使用这个标志位。

USART 的各种中断事件被连接到同一个中断向量(见下图)，有以下各种中断事件：

- 发送期间：发送完成、清除发送、发送数据寄存器空。
- 接收期间：空闲总线检测、溢出错误、接收数据寄存器非空、校验错误、LIN断开符号检测、噪音标志(仅在多缓冲器通信)和帧错误(仅在多缓冲器通信)。

如果设置了对应的使能控制位，这些事件就可以产生各自的中断。

图：USART 中断映像图



14.14 USART 寄存器说明

14.14.1状态寄存器(USART_SR)

状态寄存器(USART_SR)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0xC0					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	CTS	LBD
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TXE	TC	RXNE	TDLE	ORE	NE	FE	PE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CTS	**CTS 标志（CTS flag）** 如果设置了 CTSE 位，当 nCTS 输入变化状态时，该位被硬件置高。由软件将其清零。如果 USART_CR3 中的 CTSIE 为’1’，则产生中断。 0：nCTS 状态线上没有变化； 1：nCTS 状态线上发生变化。 注：UART4 和 UART5 上不存在这一位。
LBD	**LIN 断开检测标志（LIN break detection flag）** 当探测到 LIN 断开时，该位由硬件置’1’，由软件清’0’（向该位写 0）。如果 USART_CR3 中的 LBDIE = 1，则产生中断。 0：没有检测到 LIN 断开； 1：检测到 LIN 断开。 注意：若 LBDIE=1，当 LBD 为’1’时要产生中断。
TXE	**发送数据寄存器空（Transmit data register empty）** 当 TDR 寄存器中的数据被硬件转移到移位寄存器的时候，该位被硬件置位。如果 USART_CR1 寄存器中的 TXEIE 为 1，则产生中断。对 USART_DR 的写操作，将该位清零。 0：数据还没有被转移到移位寄存器； 1：数据已经被转移到移位寄存器。 注意：单缓冲器传输中使用该位。
TC	**发送完成（Transmission complete）** 当包含有数据的一帧发送完成后，并且 TXE=1 时，由硬件将该位置’1’。如果 USART_CR1 中的 TCIE 为’1’，则产生中断。由软件序列清除该位(先读 USART_SR，然后写入 USART_DR)。TC 位也可以通过写入’0’来清除，只有在多缓存通讯中才推荐这种清除程序。 0：发送还未完成； 1：发送完成。
RXNE	**读数据寄存器非空（Read data register not empty）** 当 RDR 移位寄存器中的数据被转移到 USART_DR 寄存器中，该位被硬件置位。如果 USART_CR1 寄存器中的 RXNEIE 为 1，则产生中断。对 USART_DR 的读操作可以将该位清零。RXNE 位也可以通过写入 0 来清除，只有在多缓存通讯中才推荐这种清除程序。 0：数据没有收到； 1：收到数据，可以读出。
IDLE	**监测到总线空闲（IDLE line detected）** 当检测到总线空闲时，该位被硬件置位。如果 USART_CR1 中的 IDLEIE 为’1’，则产生中断。由软件序列清除该位(先读 USART_SR，然后读 USART_DR)。 0：没有检测到空闲总线； 1：检测到空闲总线。 注意：IDLE 位不会再次被置高直到 RXNE 位被置起(即又检测到一次空闲总线)
ORE	**过载错误（Overrun error）** 当 RXNE 仍然是’1’的时候，当前被接收在移位寄存器中的数据，需要传送至 RDR 寄存器时，硬件将该位置位。如果 USART_CR1 中的 RXNEIE 为’1’的话，则产生中断。由软件序列将其清零(先读 USART_SR，然后读 USART_CR)。 0：没有过载错误； 1：检测到过载错误。 注意：该位被置位时，RDR 寄存器中的值不会丢失，但是移位寄存器中的数据会被覆盖。如果设置了 EIE 位，在多缓冲器通信模式下，ORE 标志置位会产生中断的。

NE	**噪声错误标志（Noise error flag）** 在接收到的帧检测到噪音时，由硬件对该位置位。由软件序列对其清玲(先读 USART_SR，再读 USART_DR)。 0：没有检测到噪声； 1：检测到噪声。 注意：该位不会产生中断，因为它和 RXNE 一起出现，硬件会在设置 RXNE 标志时产生中断。在多缓冲区通信模式下，如果设置了 EIE 位，则设置 NE 标志时会产生中断。
FE	**帧错误（Framing error）** 当检测到同步错位，过多的噪声或者检测到断开符，该位被硬件置位。由软件序列将其清零(先读 USART_SR，再读 USART_DR)。 0：没有检测到帧错误； 1：检测到帧错误或者 break 符。 注意：该位不会产生中断，因为它和 RXNE 一起出现，硬件会在设置 RXNE 标志时产生中断。如果当前传输的数据既产生了帧错误，又产生了过载错误，硬件还是会继续该数据的传输，并且只设置 ORE 标志位。 在多缓冲区通信模式下，如果设置了 EIE 位，则设置 FE 标志时会产生中断。
PE	**校验错误（Parity error）** 在接收模式下，如果出现奇偶校验错误，硬件对该位置位。由软件序列对其清零(依次读 USART_SR 和 USART_DR)。在清除 PE 位前，软件必须等待 RXNE 标志位被置’1’。如果 USART_CR1 中的 PEIE 为’1’，则产生中断。 0：没有奇偶校验错误； 1：奇偶校验错误。

14.14.2数据寄存器(USART_DR)

数据寄存器(USART_DR)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	DR
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0

Read: Write:	DR[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DR[8:0]	**数据值 (Data value)** 包含了发送或接收的数据。由于它是由两个寄存器组成的，一个给发送用(TDR)，一个给接收用(RDR)，该寄存器兼具读和写的功能。TDR 寄存器提供了内部总线和输出移位寄存器之间的并行接口。RDR 寄存器提供了输入移位寄存器和内部总线之间的并行接口。 当使能校验位(USART_CR1 中 PCE 位被置位)进行发送时，写到 MSB 的值(根据数据的长度不同，MSB 是第 7 位或者第 8 位)会被后来的校验位该取代。 当使能校验位进行接收时，读到的 MSB 位是接收到的校验位。

14.14.3波特比率寄存器(USART_BRR)

波特比率寄存器(USART_BRR)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	DIV_Mantissa[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DIV_Mantissa[7:4]				DIV_Fraction[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
DIV_Mantissa[15:4]	**USARTDIV的整数部分** 这 12 位定义了 USART 分频器除法因子(USARTDIV)的整数部分。
DIV_Fraction[3:0]	**USARTDIV的小数部分** 这 4 位定义了 USART 分频器除法因子(USARTDIV)的小数部分。

注：　如果 TE 或 RE 被分别禁止，波特计数器停止计数

14.14.4控制寄存器 1(USART_CR1)

控制寄存器 1(USART_CR1)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	UE	M	WAKE	PCE	PS	PEIE
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	TXEIE	TCIE	RXNEIE	TDLEIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

位	功能描述
UE	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分频器和输出被禁止； 1：USART 模块使能。
M	字长 (Word length) 该位定义了数据字的长度，由软件对其设置和清零 0：一个起始位，8个数据位，n个停止位； 1：一个起始位，9个数据位，n个停止位。 注意：在数据传输过程中(发送或者接收时)，不能修改这个位。
WAKE	唤醒的方法 (Wakeup method) 这位决定了把USART唤醒的方法，由软件对该位设置和清零。 0：被空闲总线唤醒； 1：被地址标记唤醒。
PCE	检验控制使能 (Parity control enable) 用该位选择是否进行硬件校验控制(对于发送来说就是校验位的产生；对于接收来说就是校验位的检测)。当使能了该位，在发送数据的最高位(如果M=1，最高位就是第9位；如果M=0，最高位就是第8位)插入校验位；对接收到的数据检查其校验位。软件对它置’1’或清’0’。一旦设置了该位，当前字节传输完成后，校验控制才生效。 0：禁止校验控制； 1：使能校验控制。

PS	校验选择（Parity selection） 当校验控制使能后，该位用来选择是采用偶校验还是奇校验。软件对它置’1’或清’0’。当前字节传输完成后，该选择生效。 0：偶校验； 1：奇校验。
PEIE	PE中断使能（PE interrupt enable）该位由软件设置或清除。 0：禁止产生中断； 1：当USART_SR中的PE为’1’时，产生USART中断。
TXEIE	发送缓冲区空中断使能（TXE interrupt enable） 该位由软件设置或清除。 0：禁止产生中断； 1：当USART_SR中的TXE为’1’时，产生USART中断。
TCIE	发送完成中断使能（Transmission complete interrupt enable） 该位由软件设置或清除。 0：禁止产生中断； 1：当USART_SR中的TC为’1’时，产生USART中断。
RXNEIE	接收缓冲区非空中断使能（RXNE interrupt enable） 该位由软件设置或清除。 0：禁止产生中断； 1：当USART_SR中的ORE或者RXNE为’1’时，产生USART中断。
IDLEIE	IDLE中断使能（IDLE interrupt enable） 该位由软件设置或清除。 0：禁止产生中断； 1：当USART_SR中的IDLE为’1’时，产生USART中断。
TE	发送使能（Transmitter enable） 该位使能发送器。该位由软件设置或清除。 0：禁止发送； 1：使能发送。 注意：1. 在数据传输过程中，除了在智能卡模式下，如果TE位上有个0脉冲(即设置为’0’之后再设置为’1’)，会在当前数据字传输完成后，发送一个“前导符”(空闲总线)。 2. 当TE被设置后，在真正发送开始之前，有一个比特时间的延迟。
RE	接收使能（Receiver enable）该位由软件设置或清除。 0：禁止接收； 1：使能接收，并开始搜寻RX引脚上的起始位。
RWU	接收唤醒（Receiver wakeup） 该位用来决定是否把USART置于静默模式。该位由软件设置或清除。当唤醒序列到来时，硬件也会将其清零。 0：接收器处于正常工作模式； 1：接收器处于静默模式。 注意：1. 在把USART置于静默模式(设置RWU位)之前，USART要已经先接收了一个数据字节。否则在静默模式下，不能被空闲总线检测唤醒。 2. 当配置成地址标记检测唤醒(WAKE位=1)，在RXNE位被置位时，不能用软件修改RWU位。
SBK	发送断开帧（Send break） 使用该位来发送断开字符。该位可以由软件设置或清除。操作过程应该是软件设置位它，然后在断开帧的停止位时，由硬件将该位复位。 0：没有发送断开字符； 1：将要发送断开字符。

14.14.5控制寄存器 2(USART_CR2)

控制寄存器 2(USART_CR2)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x10					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	LBDIE	LBDL	X	ADD[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
LINEN	**LIN模式使能 (LIN mode enable)** 该位由软件设置或清除。 0：禁止LIN模式； 1：使能LIN模式。 在LIN模式下，可以用USART_CR1寄存器中的SBK位发送LIN同步断开符(低13位)，以及检测LIN 同步断开符。
STOP[13:12]	**停止位 (STOP bits)** 这2位用来设置停止位的位数 00：1个停止位； 01：0.5个停止位； 10：2个停止位； 11：1.5个停止位;
CLKEN	**时钟使能 (Clock enable)** 该位用来使能CK引脚 0：禁止CK引脚； 1：使能CK引脚。
CPOL	**时钟极性 (Clock polarity)** 在同步模式下，可以用该位选择SLCK引脚上时钟输出的极性。和CPHA位一起配合来产生需要的时钟/数据的采样关系 0：总线空闲时CK引脚上保持低电平； 1：总线空闲时CK引脚上保持高电平。

CPHA	**时钟相位 (Clock phase)** 在同步模式下，可以用该位选择SLCK引脚上时钟输出的相位。和CPOL位一起配合来产生需要的时钟/数据的采样关系。 0：在时钟的第一个边沿进行数据捕获； 1：在时钟的第二个边沿进行数据捕获。
LBCL	**最后一位时钟脉冲 (Last bit clock pulse)** 在同步模式下，使用该位来控制是否在 CK 引脚上输出最后发送的那个数据字节(MSB)对应的时钟脉冲 0：最后一位数据的时钟脉冲不从 CK 输出； 1：最后一位数据的时钟脉冲会从 CK 输出。 注意：1．最后一个数据位就是第 8 或者第 9 个发送的位(根据 USART_CR1 寄存器中的 M 位所定义的 8 或者 9 位数据帧格式)。
LBDIE	**LIN 断开符检测中断使能 (LIN break detection interrupt enable)** 断开符中断屏蔽(使用断开分隔符来检测断开符) 0：禁止中断； 1：只要 USART_SR 寄存器中的 LBD 为’1’就产生中断。
LBDL	**LIN 断开符检测长度 (LIN break detection length)** 该位用来选择是 11 位还是 10 位的断开符检测 0：10 位的断开符检测； 1：11 位的断开符检测。
ADD[3:0]	**本设备的 USART 节点地址该位域给出本设备 USART 节点的地址。** 这是在多处理器通信下的静默模式中使用的，使用地址标记来唤醒某个 USART 设备。

注：在使能发送后不能改写这三个位（CPOL、CPHA、LBCL）

14.14.6控制寄存器 3(USART_CR3)

控制寄存器 3(USART_CR3)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x14					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	CTSIE	CTSE	RTSE
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CTSIE	**CTS中断使能（CTS interrupt enable）** 0：禁止中断； 1：USART_SR寄存器中的CTS为’1’时产生中断。
CTSE	**CTS使能（CTS enable）** 0：禁止CTS硬件流控制； 1：CTS模式使能，只有nCTS输入信号有效(拉成低电平)时才能发送数据。如果在数据传输的过程中，nCTS信号变成无效，那么发完这个数据后，传输就停止下来。如果当nCTS为无效时，往数据寄存器里写数据，则要等到nCTS有效时才会发送这个数据。
RTSE	**RTS使能（RTS enable）** 0：禁止RTS硬件流控制； 1：RTS 中断使能，只有接收缓冲区内有空余的空间时才请求下一个数据。当前数据发送完成后，发送操作就需要暂停下来。如果可以接收数据了，将 nRTS 输出置为有效(拉至低电平)。
DMAT	**DMA使能发送（DMA enable transmitter）** 该位由软件设置或清除。 0：禁止发送时的DMA模式。 1：使能发送时的DMA模式；
DMAR	**DMA使能接收（DMA enable receiver）** 该位由软件设置或清除。 0：禁止接收时的DMA模式。 1：使能接收时的DMA模式；
SCEN	**智能卡模式使能（Smartcard mode enable）** 该位用来使能智能卡模式 0：禁止智能卡模式； 1：使能智能卡模式。
NACK	**智能卡NACK使能（Smartcard NACK enable）** 0：校验错误出现时，不发送NACK； 1：校验错误出现时，发送NACK。
HDSEL	**半双工选择（Half-duplex selection）选择单线半双工模式** 0：不选择半双工模式； 1：选择半双工模式。
IRLP	**红外低功耗（IrDA low-power）** 该位用来选择普通模式还是低功耗红外模式 0：通常模式； 1：低功耗模式。
IREN	**红外模式使能（IrDA mode enable）该位由软件设置或清除。** 0：不使能红外模式； 1：使能红外模式。
EIE	**错误中断使能（Error interrupt enable）** 在多缓冲区通信模式下，当有帧错误、过载或者噪声错误时(USART_SR中的FE=1，或者ORE=1，或者NE=1)产生中断。 0：禁止中断； 1：只要USART_CR3中的DMAR=1，并且USART_SR中的FE=1，或者ORE=1，或者NE=1， 则产生中断

14.14.7保护时间和预分频寄存器(USART_GTPR)

保护时间和预分频寄存器 (USART_GTPR)			基地址：0x40001 3800(USART1) 0x4000 4400(USART2) 0x4000 4800(USART3) 偏移地址：0x18					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	GT[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PSC[7:0]							
Reset:	0	0	0	0	0	0	0	0

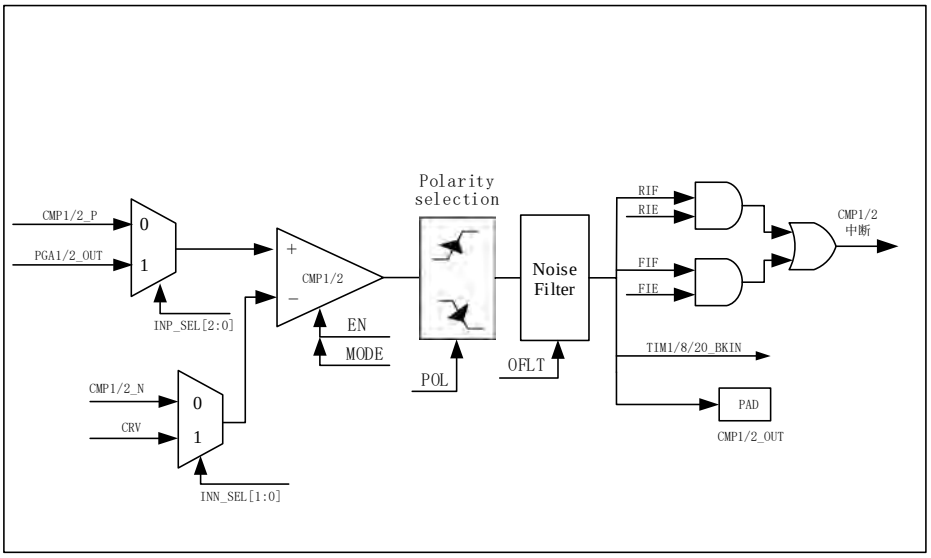
位	功能描述
GT[15:8]	**保护时间值 (Guard time value)** 该位域规定了以波特时钟为单位的保护时间。在智能卡模式下，需要这个功能。当保护时间过去后，才会设置发送完成标志。 注：UART4 和 UART5 上不存在这一位。
PSC[7:0]	**预分频器值 (Prescaler value)** **- 在红外(IrDA)低功耗模式下：** PSC[7:0]=红外低功耗波特率 对系统时钟分频以获得低功耗模式下的频 率： 源时钟被寄存器中的值(仅有8位有效) 分频 00000000：保留 – 不要写入该值； 00000001：对源时钟1分频； 00000010：对源时钟2分频； …… **- 在红外(IrDA)的正常模式下：**PSC只能设置为00000001 **- 在智能卡模式下：** PSC[4:0]：预分频值 对系统时钟进行分频，给智能卡提供时钟。 寄存器中给出的值(低5位有效)乘以2后，作为对源时钟的分频因子 00000：保留 – 不要写入该值； 00001：对源时钟进行2分频； 00010：对源时钟进行4分频； 00011：对源时钟进行6分频； …… 注意：1．位[7:5]在智能卡模式下没有意义。

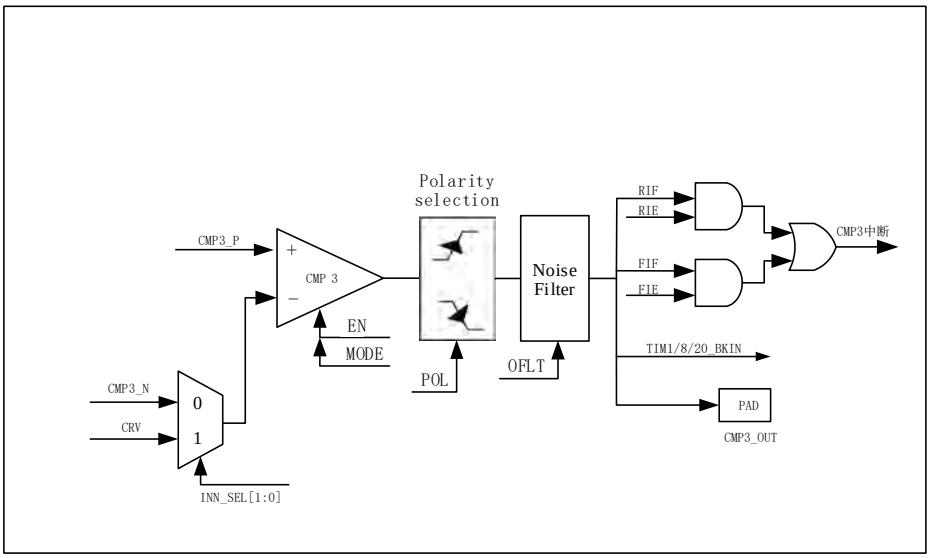
15 CMP 比较器

15.1 概述

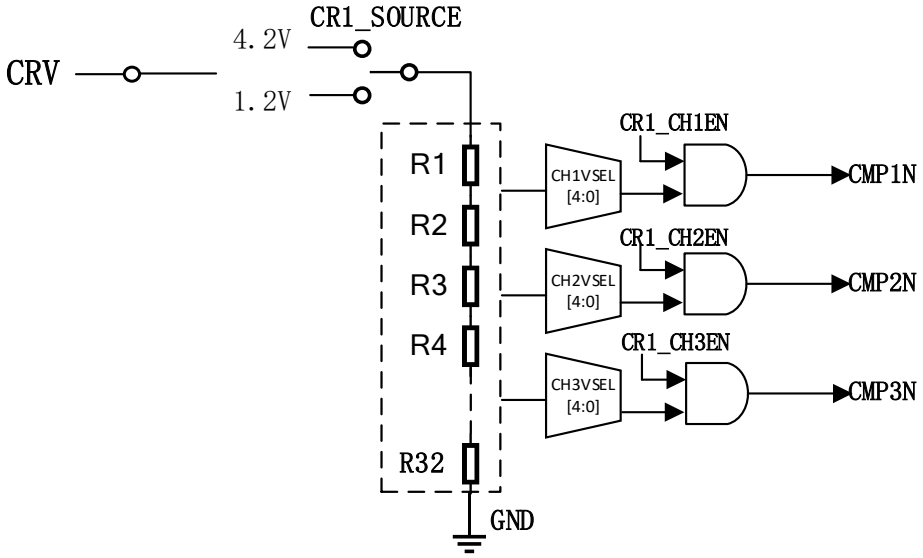
芯片内嵌 3 个通用比较器 CMP1、CMP2、CMP3 可独立使用,也可与 TIM 和 OPA 结合使用。

15.2 比较器框图





15.3 CRV 框图



15.4 比较器功能描述

15.4.1 比较器开关控制

通过设置CMPx_CR 寄存器的EN 位可给CMP 上电。设置EN 位时，它将CMP 从断电状态唤醒，清除EN 位可停止比较器工作。

15.4.2 比较器用法

比较器CMP1/2 比较所选择的INP 和INM 端口上的信号，具体流程如下：

1. 配置 CMPx_CR2 寄存器的 INP_SEL 位和 INM_SEL 位，选择所要比较的信号；
2. 配置 CMPx_CR1 寄存器的 EN 位，比较器开始上电工作；
3. 比较的结果存放于 CMPx_CR1 寄存器的 OUT 位。

另外，当CMP1 和CMP2 的INM_SEL 选择CRV 时，需要配置CMP_CR2 寄存器的INM_SEL 位.

15.4.3 CRV 相关概述

CRV 时钟使能和复位与 CMP 时钟使能复位相同。

15.5 特殊功能寄存器说明

15.5.1 CMP 控制寄存器 1(CMPx_CR1)

CMP 控制寄存器 1(CMPx_CR1)			基地址： 0x4001 7C00(CMP1)，0x4001 7C10(CMP2)，0x4001 7C20(CMP3) 偏移地址： 00H					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	FHYST[19:18]		RHYST[17:16]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	FIF	RIF	FIE	RIE	X	OFLT[2:0]		
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read:	LOCK	OUT	X	X	POL	MODE[1:0]		EN
Write:		X						
Reset:	0	0	0	0	0	0	0	0

位	功能描述
FHYST[19:18]	**下降迟滞电压** 00: 0mV 01: 5mV 10: 10mV 11: 20mV
RHYST[17:16]	**上升迟滞电压** 00: 0mV 01: 5mV 10: 10mV 11: 20mV
FIF	**下降沿中断事件标志位** 1:下降沿中断事件发生 0:无下降沿中断事件
RIF	**上升沿中断事件标志位** 1:上升沿中断事件发生 0:无上升沿中断事件
FIE	**下降沿中断使能标志位** 1:下降沿中断使能 0:下降沿中断禁止
RIE	**上升沿中断使能标志位** 1:上升沿中断使能 0:上升沿中断禁止
OFLT[10:8]	**比较器的输出滤波**,连续的 PCLK 时钟比较输出不变则认为有效,否则保持不变 000b:1 个时钟周期,无滤波 001b:4 个时钟周期 010b:16 个时钟周期 011b:32 个时钟周期 100b:64 个时钟周期 101b:128 个时钟周期 110b:256 个时钟周期

	111b:512 个时钟周期
LOCK	**CR2 只读控制位** 1:CR2 只能 read 0:CR2能read/write
OUT	**比较器的输出,**read only 1:高输出 0:低输出 注意: CPU 读取 OUT 值时，OFLT 值必须设为 001b 以上
POL	**输出极性** 1: 反相输出 0: 同相输出
MODE[2:1]	**CMP采样速率控制位** 00b:极低功率 01b:低功率 10b:中等速率 11b:高速率
EN	**比较器使能控制位** 1:比较器使能 0:比较器禁止 注：因内部结构规划，在使能CMP3时，需要先对CMP2进行使能。

15.5.2 CMP 控制寄存器 2(CMPx_CR2)

CMP 控制寄存器 2(CMPx_CR2)			基地址： 0x4001 7C00(CMP1)，0x4001 7C10(CMP2)，0x4001 7C20(CMP3) 偏移地址： 04H					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	X	X	BLANKING [17:16]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	BLANKING[15]	X	X	X	X	INP_SEL[10:8]		
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	INN_SEL[7:6]		X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BLANKING[17:15]	**电流尖峰遮蔽功能信号源选择** 000: 不遮蔽 001: TIM1 OC4 输出作为遮蔽信号源 010: TIM2 OC3 输出作为遮蔽信号源

	011: TIM3 OC3 输出作为遮蔽信号源 100: TIM4 OC3 输出作为遮蔽信号源 101: TIM20 OC4 输出作为遮蔽信号源 其他配置: 保留
INP_SEL[10:8]	**P 端输入源选择** 000:CMP1_P,CMP2_P,CMP3_P 001:PGA1_out(CMP1),PGA2_out(CMP2) 010:保留 011:保留 100:保留
INM_SEL[7:6]	**N 端输入源选择** 00: CMP1_N, CMP2_N, CMP3_N 01: CRV

15.5.3 CMP 调节寄存器(CMPx_CAL)

CMP 调节寄存器(CMPx_CAL)			基地址： 0x4001 7C00(CMP1)，0x4001 7C10(CMP2)，0x4001 7C20(CMP3) 偏移地址： 08H					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	CAL_NADJ[3:0]			
Reset:	0	0	0	0	0	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	CAL_NEN	CAL_PEN	CAL_PADJ[3:0]			
Reset:	0	0	0	0	1	0	0	0

位	功能描述
CAL_NADJ[11:8]:	CMP NMOS offset 调节(default:0111)
CAL_NEN	CMP NMOS 校正使能位 1:CMP NMOS 校正使能 0:CMP NMOS 校正禁止
CAL_PEN	CMP PMOS 校正使能位 1:CMP PMOS 校正使能 0:CMP PMOS 校正禁止
CAL_PADJ[3:0]:	CMP PMOS offset 调节(default:1000)

15.5.4 CRV 控制寄存器（CRV_CR1）

CRV 控制寄存器（CRV_CR1）			基地址： 0x4001 7D00 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	CR1_EN	X	X	X	CR1_SOURCE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CR1_SOURCE	CRV 来源 (Compartor Reference Voltage source) 0：1.2V； 1：4.2V。
CR1_EN	比较器参考电压源使能 0:禁止 1:使能

15.5.5 CRV 通道 1 输出选择寄存器（CRV_CH1）

CRV 通道 1 输出选择寄存器（CRV_CH1）			基地址： 0x4001 7D00 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	CH1VSEL[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CH1VSEL[4:0]	通道 1 输出选择(CRV CH1 output select)： 00000b: 1/32 SOURCE　　10000b: 17/32 SOURCE 00001b: 2/32 SOURCE　　10001b: 18/32 SOURCE 00010b: 3/32 SOURCE　　10010b: 19/32 SOURCE 00011b: 4/32 SOURCE　　10011b: 20/32 SOURCE

	00100b: 5/32 SOURCE 10100b: 21/32 SOURCE 00101b: 6/32 SOURCE 10101b: 22/32 SOURCE 00110b: 7/32 SOURCE 10110b: 23/32 SOURCE 00111b: 8/32 SOURCE 10111b: 24/32 SOURCE 01000b: 9/32 SOURCE 11000b: 25/32 SOURCE 01001b: 10/32 SOURCE 11001b: 26/32 SOURCE 01010b: 11/32 SOURCE 11010b: 27/32 SOURCE 01011b: 12/32 SOURCE 11011b: 28/32 SOURCE 01100b: 13/32 SOURCE 11100b: 29/32 SOURCE 01101b: 14/32 SOURCE 11101b: 30/32 SOURCE 01110b: 15/32 SOURCE 11110b: 31/32 SOURCE 01111b: 16/32 SOURCE 11111b: 32/32 SOURCE

15.5.6 CRV 通道 2 输出选择寄存器（CRV_CH2）

CRV 通道 2 输出选择寄存器（CRV_CH2）			基地址： 0x4001 7D00 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	CH2VSEL[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CH2VSEL[4:0]	通道 2 输出选择(CRV CH2 output select): 00000b: 1/32 SOURCE 10000b: 17/32 SOURCE 00001b: 2/32 SOURCE 10001b: 18/32 SOURCE 00010b: 3/32 SOURCE 10010b: 19/32 SOURCE 00011b: 4/32 SOURCE 10011b: 20/32 SOURCE 00100b: 5/32 SOURCE 10100b: 21/32 SOURCE 00101b: 6/32 SOURCE 10101b: 22/32 SOURCE 00110b: 7/32 SOURCE 10110b: 23/32 SOURCE 00111b: 8/32 SOURCE 10111b: 24/32 SOURCE 01000b: 9/32 SOURCE 11000b: 25/32 SOURCE

	01001b: 10/32 SOURCE 11001b: 26/32 SOURCE 01010b: 11/32 SOURCE 11010b: 27/32 SOURCE 01011b: 12/32 SOURCE 11011b: 28/32 SOURCE 01100b: 13/32 SOURCE 11100b: 29/32 SOURCE 01101b: 14/32 SOURCE 11101b: 30/32 SOURCE 01110b: 15/32 SOURCE 11110b: 31/32 SOURCE 01111b: 16/32 SOURCE 11111b: 32/32 SOURCE

15.5.7 CRV 通道 3 输出选择寄存器（CRV_CH3）

CRV 通道 3 输出选择寄存器（CRV_CH3）			基地址： 0x4001 7D00 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	CH3VSEL[4:0]				
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CH3VSEL[4:0]	通道 3 输出选择(CRV CH3 output select): 00000b: 1/32 SOURCE 10000b: 17/32 SOURCE 00001b: 2/32 SOURCE 10001b: 18/32 SOURCE 00010b: 3/32 SOURCE 10010b: 19/32 SOURCE 00011b: 4/32 SOURCE 10011b: 20/32 SOURCE 00100b: 5/32 SOURCE 10100b: 21/32 SOURCE 00101b: 6/32 SOURCE 10101b: 22/32 SOURCE 00110b: 7/32 SOURCE 10110b: 23/32 SOURCE 00111b: 8/32 SOURCE 10111b: 24/32 SOURCE 01000b: 9/32 SOURCE 11000b: 25/32 SOURCE 01001b: 10/32 SOURCE 11001b: 26/32 SOURCE 01010b: 11/32 SOURCE 11010b: 27/32 SOURCE 01011b: 12/32 SOURCE 11011b: 28/32 SOURCE 01100b: 13/32 SOURCE 11100b: 29/32 SOURCE

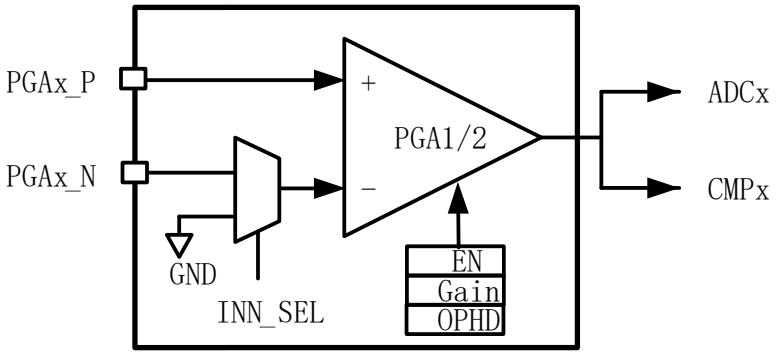
	01101b: 14/32 SOURCE	11101b: 30/32 SOURCE
	01110b: 15/32 SOURCE	11110b: 31/32 SOURCE
	01111b: 16/32 SOURCE	11111b: 32/32 SOURCE

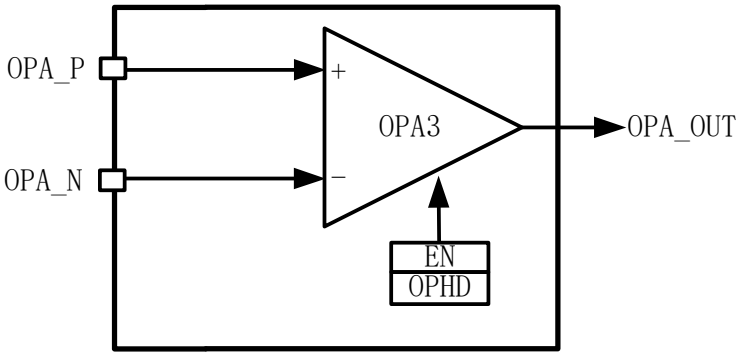
16 OPAMP 运算放大器

10.1 概述

芯片内嵌三个运算放大器，每个运算放大器的输入和 OPA_3 的输出都连接到 I/O ，每个运算放大器的输出通过共享I/O 可以与ADC ，CMP 比较器相连。

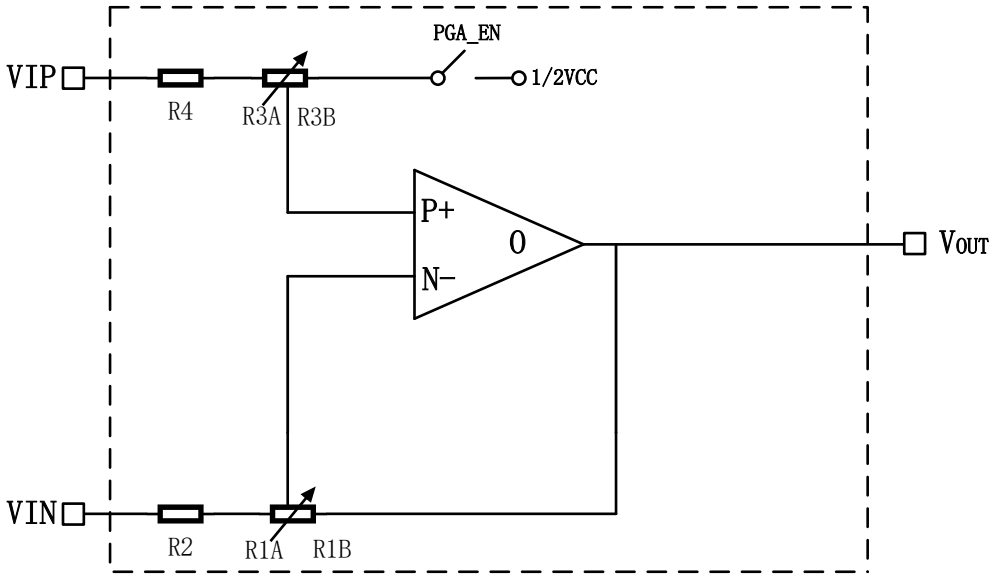
10.2 OPAMP 框图





10.3 运算放大器主要特征

PGA内部原理图



注：R2=R4，R3A=R1A，R3B=R1B

- PGA 输入阻抗为：R4+R3A 或 R2+R1A，详细数值参见 PGA 电气特性
- PGA 增益计算公式为：

$$GAIN = \frac{R3B}{R3A + R4} = \frac{R1B}{R1A + R2}$$

- PGA 输出计算公式为：

$$Vout = (VIP - VIN) \times GAIN + 1/2VCC$$

- 轨对轨输入/输出
- 输出连接到 I/O 或 ADC 上

10.4 特殊功能寄存器说明

10.4.1 OPA 控制寄存器(OPA_CR)

OPA 控制寄存器(OPA_CR)			基地址： 0x4001 7800 偏移地址： 00H					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8

Read: Write:	X	PGA2_INN_SEL	PGA1_INN_SEL	OPA3_EN	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: Write:	PGA2_Gain[7:5]			PGA2_EN	PGA1_Gain[3:1]			PGA1_EN
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PGA2_INN_SEL	**PGA2 负端选择位** 1：接 PGA2_N 0：内部接地
PGA1_INN_SEL	**PGA1 负端选择位** 1：接 PGA1_N 0：内部接地
OPA3_EN	**OPA3 使能位** 1:OPA3 使能 0:OPA3禁止
PGA2_Gain[7:5]	**PGA2 放大倍率选择** 000b:2X 001b:4X 010b:6X 011b:8X 100b:10.8X 101b:13.18X 110b:16X 111b:18X
PGA2_EN	**PGA2使能位** 1:PGA2使能 0:PGA2禁止
PGA1_Gain[3:1]	**PGA1 放大倍率选择** 000b:2X 001b:4X 010b:6X 011b:8X 100b:10.8X 101b:13.18X 110b:16X 111b:18X
PGA1_EN	**PGA1使能位** 1:PGA1使能 0:PGA1禁止

10.4.2 OPA 控制寄存器 2(OPA_CR2)

OPA 控制寄存器 2(OPA_CR2)			基地址： 0x4001 7800 偏移地址： 04H					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	OPHD3	X	OPHD2	OPHD1
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OPHD3	**OPA3高驱动功能使能位** 1：高驱动功能使能 0：高驱动功能禁止
OPHD2	**PGA2高驱动功能使能位** 1：高驱动功能使能 0：高驱动功能禁止
OPHD1	**PGA1高驱动功能使能位** 1：高驱动功能使能 0：高驱动功能禁止

10.4.3 OPA 控制寄存器 3(OPA_CR3)

OPA 控制寄存器 3(OPA_CR3)			基地址： 0x4001 7800 偏移地址： 08H					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	PGA2_Test[6:5]		PGA1_Test[4:3]		PGA2_CAL	PGA1_CAL	OPA_CAL
Reset:	0	0	0	0	0	0	0	0

位	功能描述

PGA2_Test[6:5]	**PGA2测试模式** 00: 关闭测试模式 01: 开启
PGA1_Test[4:3]	**PGA1测试模式** 00: 关闭测试模式 01: 开启
PGA2_CAL	**PGA2校正使能** 0:关闭 1:校正使能
PGA1_CAL	**PGA1校正使能** 0:关闭 1:校正使能
OPA_CAL	**OPA校正使能** 0:关闭 1:校正使能

17 AMPU 模块

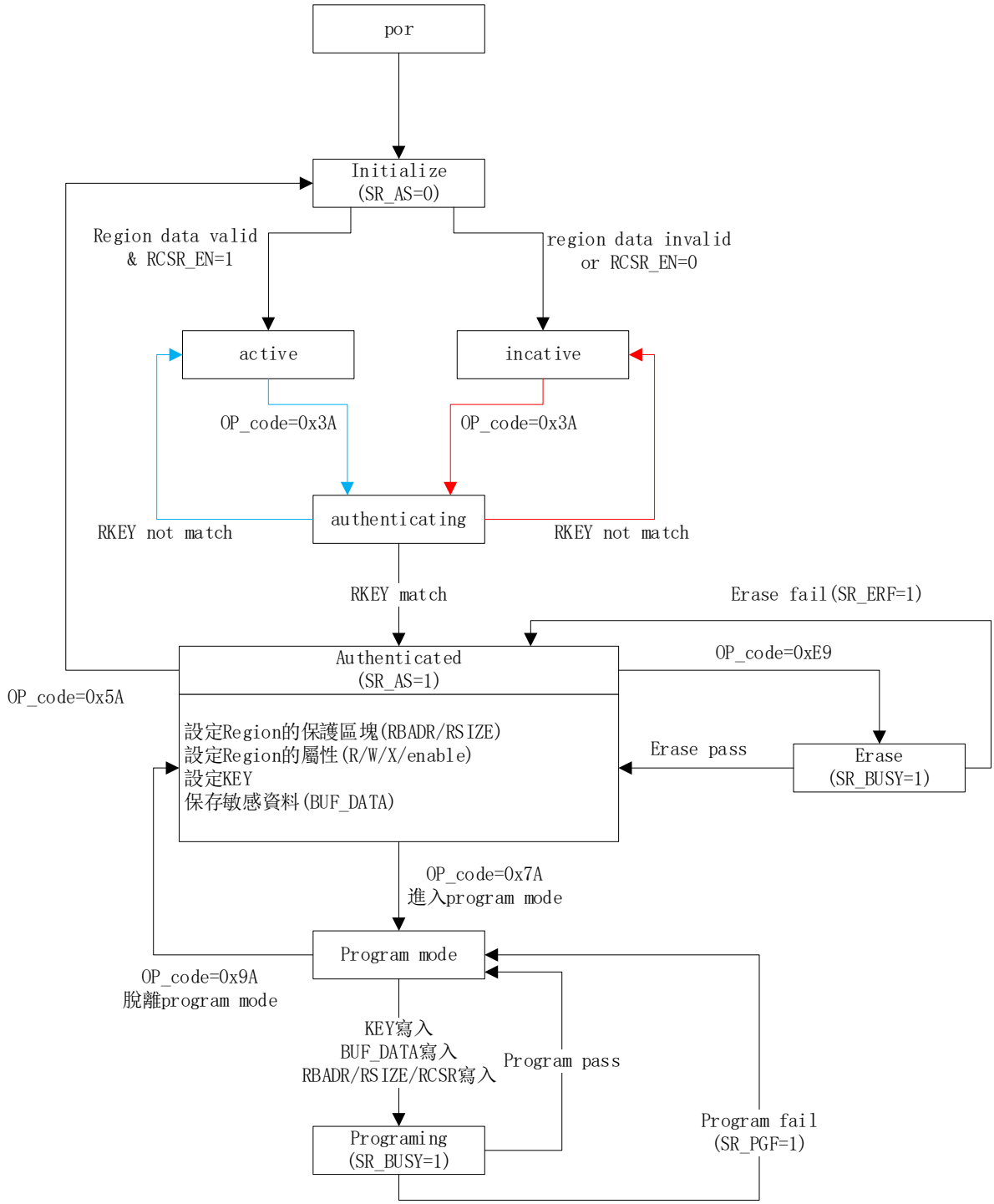
17.1 AMPU 简介

认证型存储单元保护模块(AMPU)是一种提供芯片内部存储单元保护机制的模块，主要包含一个 222 × 32 bit 的模块内数据区和四个独立可编程保护区域，访问或修改 AMPU 的设定和保护区域都需要先认证并通过。

17.2 AMPU 主要功能

- 访问或修改 AMPU 的设定和保护区域都需要先认证并通过
  - AMPU 的认证方式是写入一个 128 bit 的认证密钥(KEY)
  - 可配置认证次数上限
- 一个 222 × 32 bit 的内部数据区(Data)
- 四个独立可编程保护区域(Region0-3)，可分别设定保护区域的首地址和尾地址
- 四个独立可编程保护区域(Region0-3) ，可分别设定各自区域的读/写/执行状态
- 四个独立可编程保护区域(Region0-3)如果有重叠，重叠部分的区域编号越小优先级越高
- AMPU 的设定掉电后可保存，上电后自加载
  - AMPU 的 CR、KEY、Region0-3、Data 均有对应的 Flash 配置区
  - 可以通过 Erase 和 Program 指令来修改 Flash 配置区

17.3 AMPU 整体流程图



17.4 AMPU 认证功能

17.4.1 认证功能描述

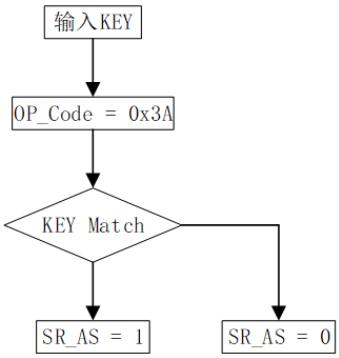
访问或修改 AMPU 的设定和保护区域都需要先认证并通过，认证的方式是通过写入一个 128 bit 的认证密钥(KEY)，该密钥(KEY)只能通过全擦 AMPU 的方式进行更新。

KEY 默认为: 0xFFFFFFFF_FFFFFFFF_FFFFFFFF。

在未认证通过的情况下(SR_AS = 0)，AMPU 除 SR 寄存器外，所有寄存器均不可操作，且读取到的都是 0x5A5AA5A5。

17.4.2 认证流程

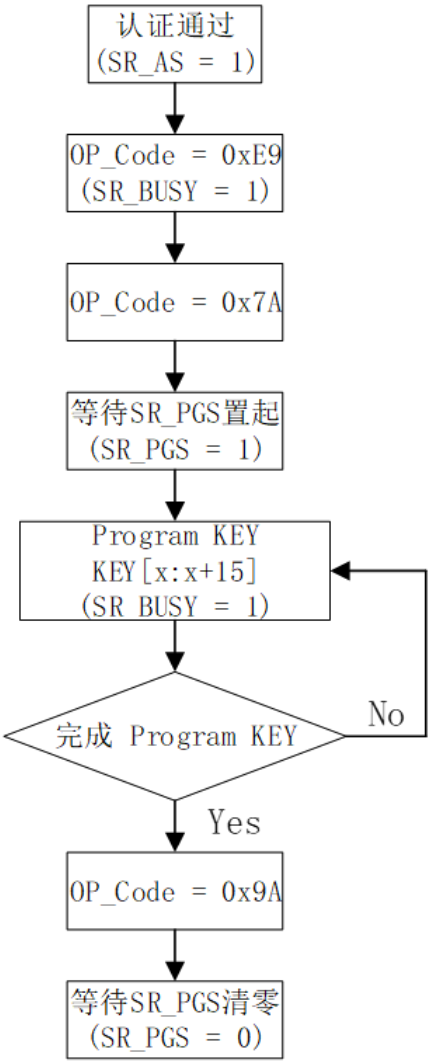
1. 写入 AMPU 有效的 KEY(对应 AMPU 寄存器中的 KEY[0:127])
2. OP_Code 写入 0x3A(认证启动指令)
3. 等待 SR_AS 标志位置起(认证通过)



17.4.3 KEY 更新流程

1. 认证通过
2. OP_Code 写入 0xE9(擦除启动指令，Erase)
3. 等待 SR_BUSY 标志位清零(擦除开始后，SR_BUSY 由硬件置 1，直到擦除完成，SR_BUSY 自动清零)
4. OP_Code 写入 0x7A(进入 Program 指令)
5. 等待 SR_PGS 置起(成功进入 Program 状态)
6. 写入新 KEY
7. 等待 SR_BUSY 标志位清零(写入新 KEY 后，SR_BUSY 由硬件置 1，直到写入完成，SR_BUSY 自动清零，128 bit 的 KEY[0:127]被拆分成 8 个寄存器，每写入一个寄存器都必须等待 SR_BUSY 清零，否则可能会出错)
8. OP_Code 写入 0x9A(退出 Program 指令)
9. 等待 SR_PGS 清零(成功退出 Program 状态)

注：Program 新 KEY 后，新 KEY 即刻生效，但不会强制退出验证通过状态，即 SR_AS 会保持置起状态。退出验证通过状态，再进行认证时需要输入新的 KEY 才会通过验证。



17.4.4 KEY 认证上限功能

- KEY 的认证次数可以配置上限，当认证次数达到上限后就会触发 Authentication Limitation Fail(SR_ALF)，AMPU 将会被锁住，即使输入正确的 KEY 也不能将 AMPU 解锁，只能执行 0x59(Remove 指令)。Remove 指令会将 Flash、CCM、SRAM、AMPU 都清除至默认状态。
- 也可以配置 CR_ALFE 功能位，在产生 Authentication Limitation Fail(SR_ALF)后直接全擦芯片，将 Flash、CCM、SRAM、AMPU 都清除至默认状态。
- 触发 Authentication Limitation Fail(SR_ALF)后不会影响芯片其他功能的使用。

17.5 AMPU 数据区(Data)

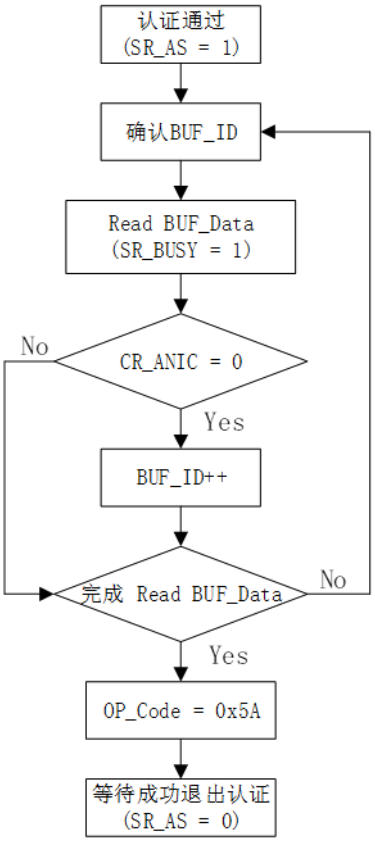
17.5.1 数据区概述

AMPU 模块内部自带一个 222 × 32 bit 的数据区，用于存储一些关键并需要加密存储的数据，该数据区在 Flash 配置区内，因此掉电可以保存。
在未验证通过的状态下(SR_AS = 0)，读取 BUF_Data 只会得到 0x5A5AA5A5，也不可写入。

17.5.2 Data 读取流程

1. 认证通过
2. 确认 BUF_ID 的值(反应当前读取的 Data 的相对地址)
3. 读取 BUF_Data
4. 读完成后，OP_Code = 0x5A
5. 等待 SR_AS = 0，确认退出认证通过状态

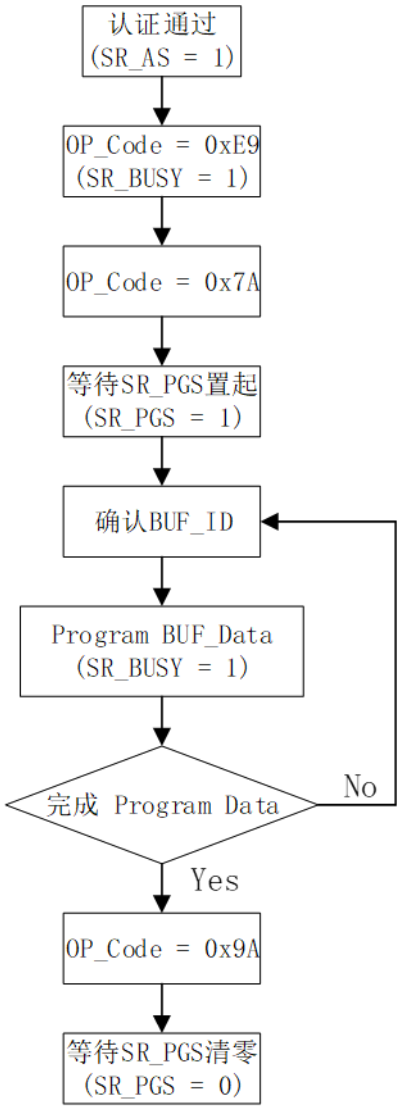
注：CR_AINC 用于控制读/写 BUF_Data 过程中，BUF_ID 是否自增长。也可以通过写入 BUF_ID 来选择读/写的 BUF_Data



17.5.3 Data Program 流程

1. 认证通过
2. OP_Code 写入 0xE9(擦除启动指令，Erase)
3. 等待 SR_BUSY 标志位清零(擦除开始后，SR_BUSY 由硬件置 1，直到擦除完成，SR_BUSY 自动清零)
4. OP_Code 写入 0x7A(进入 Program 指令)
5. 等待 SR_PGS 置起(成功进入 Program 状态)
6. 确认 BUF_ID 的值(反应当前写入的 Data 的相对地址)
7. 写入 BUF_Data
8. 等待 SR_BUSY 标志位清零(写入新 BUF_Data 后，SR_BUSY 由硬件置 1，直到写入完成，SR_BUSY 自动清零)
9. 不同 BUF_ID 的 BUF_Data 的写入只需要重复步骤 6 至 8 即可
10. OP_Code 写入 0x9A(退出 Program 指令)
11. 等待 SR_PGS 清零(成功退出 Program 状态)

注：CR_AINC 用于控制读/写 BUF_Data 过程中，BUF_ID 是否自增长。也可以通过写入 BUF_ID 来选择读/写的 BUF_Data



17.6 四个独立可编程保护区域(Region0-3)

17.6.1 四个独立可编程保护区域概述

AMPU 可以通过寄存器来设定保护区域的首地址和尾地址，以及对应区域的读/写/执行状态，主要控制对象为 FLASH/SRAM/CCM。相应的寄存器配置均有 Flash 配置区进行“注册”，掉电可保存，上电自加载。
如果需要 Region 的配置掉电可保存，则需要在配置 Region 之前进入 Program 状态，否则 Region 的配置在上电复位后将加载 Flash 配置区的状态。

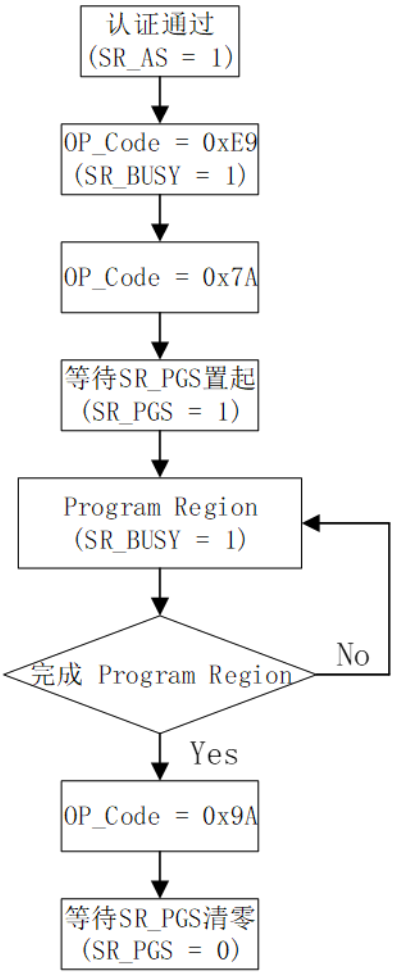
- 写：可设定保护区域的写保护，开启写保护后，该区域将无法写入，但是 FLASH 的 Erase 功能会将 FLASH 擦除至 0xFF
- 读：可设定保护区域的读保护，开启读保护后，读取该区域的值将会得到 0xFF
- 执行：可设定保护区域是否可以执行函数，如果强制在不可执行的保护区域执行函数将会触发 HardFault

17.6.2 AMPU Region 功能使用推荐方案

1. Program Region 的配置，将 AMPU 进行“注册”，使得指定区域内的读/写/执行状态在掉电后可以保存，上电后自加载，起到保护数据和代码的作用。
2. 上电后，不执行 Program 命令，验证通过后，仅暂时性修改 Region 的读/写/执行状态，即“登录”，然后退出验证通过状态，方便代码执行，同时最大限度保护 AMPU，防止 AMPU 被修改。芯片掉电再上电后，Region 区通过自加载处于保护状态。

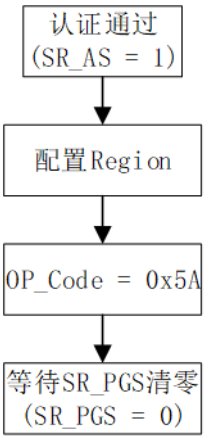
17.6.3 Region Program 流程

1. 认证通过
2. OP_Code 写入 0xE9(擦除启动指令，Erase)
3. 等待 SR_BUSY 标志位清零(擦除开始后，SR_BUSY 由硬件置 1，直到擦除完成，SR_BUSY 自动清零)
4. OP_Code 写入 0x7A(进入 Program 指令)
5. 等待 SR_PGS 置起(成功进入 Program 状态)
6. 配置一个 Region 寄存器
7. 等待 SR_BUSY 标志位清零(对 Region 寄存器执行写入操作后，SR_BUSY 由硬件置 1，直到写入完成，SR_BUSY 自动清零)
8. 要完整配置 Region 功能需要在写入每一个寄存器时重复步骤 6 和 7
9. OP_Code 写入 0x9A(退出 Program 指令)
10. 等待 SR_PGS 清零(成功退出 Program 状态)



17.6.4 Region 非 Program 流程

1. 认证通过
2. 配置 Region 寄存器
3. OP_Code 写入 0x5A(退出验证状态指令)
4. 等待 SR_AS 清零(成功退出 Program 状态)



17.7 FLASH 配置

17.7.1 FLASH 配置概述

芯片为 AMPU 的 CR、KEY、Region、BUF_Data 寄存器提供了一块 Flash 用于“注册”，从而实现掉电存储和上电自加载。

- 在 Program 状态下(SR_PGS = 1)，对 CR、KEY、Region、BUF_Data 的配置将会写入到对应的 Flash 区域(即，将 AMPU 的配置进行“注册”)，因此写入操作要遵守 Flash 的写入规范。
- 非 Program 状态下(SR_PGS = 0)，对 CR、KEY、Region、BUF_Data 的配置不会写入到对应的 Flash 区域(即，“登录”使用 AMPU)，因此可以对寄存器按位写入。

17.7.2 Program Fail(SR_PGF)

- 填入新值前必须保证 Flash 区域的值为 0xFF，否则会产生 Program Fail(SR_PGF)。
- Program Fail(SR_PGF)只表征当前 program 操作的结果，因此如果上一步的 Program 导致 Program Fail，而当前步骤中的 Program 是正确的，则 SR_PGF 会被清零。
- 执行 OP_Code = 0xE9(Erase 指令可以擦除 AMPU 的所有 FLASH 配置)。

17.7.3 Autoload Fail(SR_ALF)

- CR、KEY、Region 的 Flash 配置具有互补自检功能(即，填入 16bit 寄存器配置的同时，会有高 16bit 的互补值)，因此在产生 Program Fail 后复位芯片，会产生 Autoload Fail(SR_ACF)。
- 如果芯片复位后，产生了 Autoload Fail(ACF)，则说明此时 AMPU 的 Flash 配置不正常，因此 AMPU 会将整块芯片都上锁(即，整块芯片处于不可读不可写，且只有前 16 Byte 可执行)，以防 AMPU 被故意破坏，导致数据泄露。只有执行 0x59(Remove 指令)后，芯片才可以恢复正常工作。

17.8 调试支持

17.8.1 调试支持概述

AMPU 可以配置在 Debug 模式下，验证能否通过，以及配置是否起效。

17.8.2 调试支持配置

- DBGX = 0 时，验证不通过，几乎不能配置 AMPU，复位后仍有效，退出 DBG 模式后 AMPU 正常，可以防止 AMPU 在 Debug 模式下被破坏。
- DBGX = 1 时，验证可以通过，可以配置 AMPU，方便调试。

17.9 AMPU 寄存器说明

17.9.1 AMPU 操作寄存器（AMPU_OP）

AMPU 操作寄存器（AMPU_OP）			基地址： 0x4002 0C00 偏移地址： 0x00					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	OP_Code							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
OP_Code	0x59: Remove（Erase CCM, FALSH 256K, and AMPU to blank） 使用Remove时需要对FLASH的AR清零。 0xE9: 擦除启动(SR_AS=1有效) 0x9A: program停止(SR_AS=1有效) 0x7A: program启动(SR_AS=1有效)

	0x5A: 脱离认证(SR_AS=1有效) 0x3A: 认证启动,KEY必须先写入, 认证通过SR_AS=1

17.9.2 AMPU 状态寄存器（AMPU_SR）

AMPU 状态寄存器（AMPU_SR）			基地址： 0x4002 0C00 偏移地址： 0x04					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	BLANK_R3	BLANK_R2	BLANK_R1	BLANK_R0	X	KBLANK	CRBLANK	BLANK
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	ACF	ALF	ERF	PGF	X	PGS	BUSY	AS
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BLANK_R3	1: Region 3 is Blank 0: Region 3 is not Blank
BLANK_R2	1: Region 2 is Blank 0: Region 2 is not Blank
BLANK_R1	1: Region 1 is Blank 0: Region 1 is not Blank
BLANK_R0	1: Region 0 is Blank 0: Region 0 is not Blank
KBLANK	1: AMPU's Key is Blank 0: AMPU's Key is not Blank
CRBLANK	1: CR is Blank 0: CR is not Blank
BLANK	1: All Blank 0: not All Blank
ACF	1: autoload check fail 0: autoload check pass
ALF	1: auth limitation fail 0: not auth limitation fail
ERF	1: erase fail 0: erase pass

PGF	1: program fail 0: program pass
PGS	1: program status (OP_Code=0x7A) 0: not program status (OP_Code=0x9A)
BUSY	1: AMPU operation busy 0: AMPU operation done
AS	Authenticated Status 1: 已认证状态 0: 非认证状态

17.9.3 AMPU 缓冲 ID 寄存器（AMPU_BUF_ID）

AMPU 缓冲寄存器（AMPU_BUF）			基地址： 0x4002 0C00 偏移地址： 0x08					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	BUF_ID							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
BUF_ID	buffer index (0-221) 注:SR_AS=1操作有效

17.9.4 AMPU 缓冲数据寄存器（AMPU_BUF_Data）

AMPU 缓冲数据寄存器（AMPU_BUF_Data）			基地址： 0x4002 0C00 偏移地址： 0x0C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	BUF_Data[31:24]							

Reset:	1	1	1	1	1	1	1	1
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	BUF_Data[23:16]							
Reset:	1	1	1	1	1	1	1	1
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	BUF_Data[15:8]							
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	BUF_Data[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述
BUF_Data	buffer data 当读取或写入一笔资料，buffer index+1 注:SR_AS=1操作有效，RCSR_AS=0都回0xFFFFFFFF

17.9.5 AMPU 密钥数据寄存器（AMPU_KEY）

AMPU 密钥数据寄存器（AMPU_KEY）			基地址： 0x4002 0C00 偏移地址： 0x10,0x14,0x18,0x1C,0x20,0x24,0x28,0x2C					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	KEY[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	KEY[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
KEY[15:0]	KEY default 写入0xFFFF，不可写入0x0000，会导致ALF。

17.9.6 AMPU 控制寄存器（AMPU_CR）

AMPU 控制寄存器（AMPU_CR）			基地址： 0x4002 0C00 偏移地址： 0x30					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	FAUTH	AINC	DBGEN	ALFE	X	ALIM[2:0]		
Reset:	1	1	0	0	0	0	1	0

位	功能描述
FAUTH	AMPU設定權限 1: AMPU access by NonPrivilege or Privlege 0: AMPU access by Privilege
AINC	0: BUF_ID部会自动加1 1: BUF_ID经过read/write会自动加1
DBGEN	1: Debug not disable auth and read region 0: Debug disable auth.and read region
ALFE	SR_AFL為1時,是否擦除保護區域 1: 全擦芯片(Erase CCM, FALSH 256K, and AMPU to blank) 0: 不做任何行为
ALIM	Authenticated limitation count 认证失败次数> ALIM, SR_ALF置1 设定次数为ALIM+1

17.9.7 AMPU 区域基地址低位寄存器（AMPU_ RBADRLx）

AMPU 区域基地址低位寄存器（AMPU_ RBADRLx）			基地址： 0x4002 0C00 偏移地址： 0x40+（x*0x20）(x=0-3)					
	Bit31	30	29	28	27	26	25	Bit24

Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	RBADRLx[15:8]							
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	RBADRLx[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述
RBADRLx[15:0] (x:0-3)	region base address Low 16 bits note:SR_AS=1操作有效

17.9.8 AMPU 区域基地址高位寄存器（AMPU_ RBADRHx）

AMPU 区域基地址高位寄存器（AMPU_ RBADRHx）			基地址： 0x4002 0C00 偏移地址： 0x44+（x*0x20）(x=0-3)					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	RBADRHx[15:8]							
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	RBADRHx[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述

RBADRHx[15:0] (x:0-3)	region base address High 16 bits note:SR_AS=1操作有效

17.9.9 AMPU 区域结束地址低位寄存器（AMPU_ READRLx）

AMPU 区域结束地址低位寄存器（AMPU_ READRLx）			基地址： 0x4002 0C00 偏移地址： 0x48+（x*0x20）(x=0-3)					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	READRLx[15:8]							
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	READRLx[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述
READRLx[15:0] (x:0-3)	region end address Low 16 bits note:SR_AS=1操作有效

17.9.10AMPU 区域结束地址高位寄存器（AMPU_ READRHx）

AMPU 区域结束地址高位寄存器（AMPU_ READRHx）			基地址： 0x4002 0C00 偏移地址： 0x4c+（x*0x20）(x=0-3)					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read:	READRHx[15:8]							

Write:								
Reset:	1	1	1	1	1	1	1	1
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	READRHx[7:0]							
Reset:	1	1	1	1	1	1	1	1

位	功能描述
READRHx[15:0] (x:0-3)	region end address High 16 bits note:SR_AS=1操作有效

17.9.11AMPU 区域配置寄存器（AMPU_ RCSRx）

AMPU 区域配置寄存器（AMPU_ RCSRx）			基地址： 0x4002 0C00 偏移地址： 0x50+（x*0x20）(x=0-3)					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	RCSRx_PV	RCSRx_NPV	RCSRx_W	RCSRx_R	RCSRx_X	RCSRx_DIS
Reset:	0	0	1	1	1	1	1	1

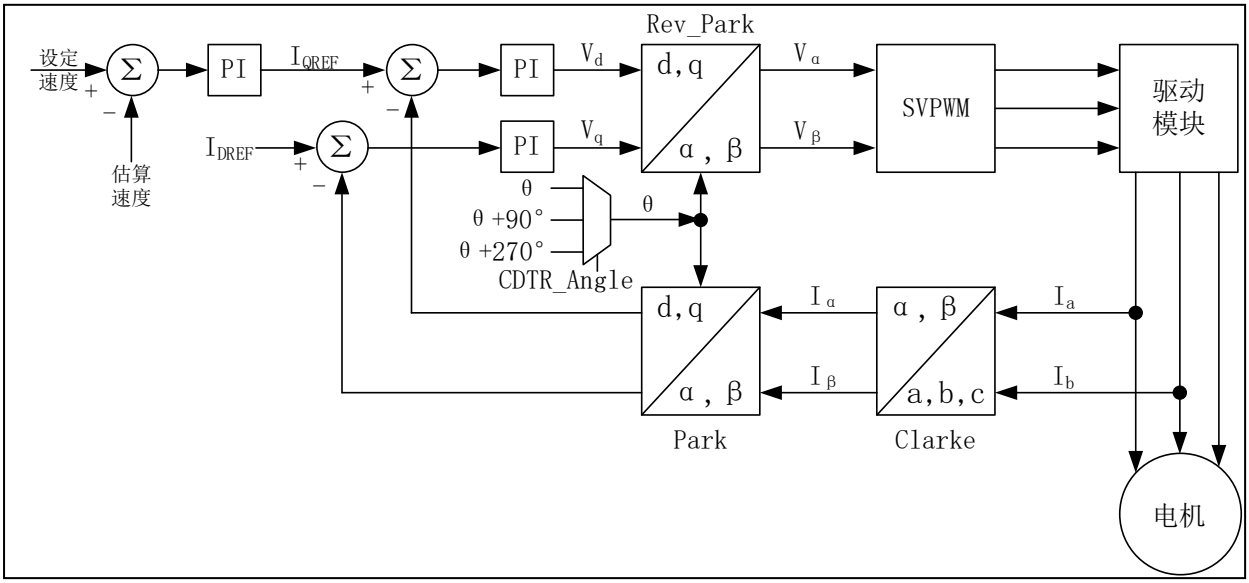
位	功能描述
RCSRx_PV (x:0-3)	Region在Privilege mode有效位 1: 有效 0: 无效
RCSRx_NPV (x:0-3)	Region在NonPrivilege mode有效位 1: 有效 0: 无效
RCSRx_W (x:0-3)	1: region 可写入 0: region 不可写入 note:SR_AS=1操作有效

RCSRx_R (x:0-3)	1: region 可读取 0: region 不可读取 note:SR_AS=1操作有效
RCSRx_X (x:0-3)	1: region 可执行程序 0: region 不可执行程序 note:SR_AS=1操作有效
RCSRx_DIS (x:0-3)	1: region disable 0: region enable note:SR_AS=1操作有效

18 ME 电机专用协同处理器

18.1 概述

RX32S50 提供一电机专用模块(Motor Engine),简称 ME.
ME 里面包含 PID 硬件运算器,坐标转换(CDTR)以及 SVPWM 的硬件运算器.
坐标转换(CDTR)包含“Clarke 变换”、“Park 变换”、“反 Park 变换”
PID 模块提供“速度”和“电流”的硬件加速



18.2 PID 闭环控制模块

18.2.1 概述

PID 主要是由三个计算单元组成,比例单元(P),积分单元(I)和微分单元(D),主要公式如下:

$$OUT = K_p \times e(t) + K_i \times \int_0^t e(T)dT + K_d \frac{de(t)}{dt}$$

$$REF - FB = Error_{[n]}$$

比例单元

$$P_{OUT} = Error_{[n]} \times KPG \div 2^{\wedge KPD[3:0]}$$

积分单元

$$I_{MIDOUT[n]} = Error_{[n]} \times KIG \div 2^{\wedge KID[3:0]}$$

$$I_{OUT} = (I_{MIDOUT[1]} + I_{MIDOUT[2]} \cdots\cdots + I_{MIDOUT[n]}) \div KIMULP$$

注：$-INTGLIM \le (I_{MIDOUT[1]} + I_{MIDOUT[2]} \cdots\cdots + I_{MIDOUT[n]}) \le INTGLIM$

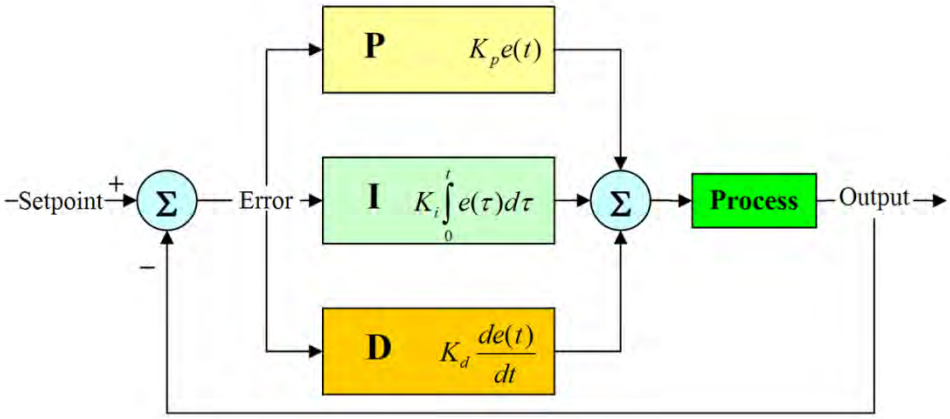
微分单元

$$D_{OUT} = (Error_{[n]} - Error_{[n-1]}) \times KDG \div 2^{\wedge KDD[3:0]}$$

$$OUT = P_{OUT} + I_{OUT} + D_{OUT}$$

注：$-OUTLIM \le OUT \le OUTLIM$

透过调整 Kp,Ki,Kd 的增益来调整其特性.



RX32S50 系列提供 3 组 PID 硬件加速的配置,透过灵活的参数配置调整计算的方式.

18.2.2 PID 使用方法

1. 设置 PID_CR_TRGSEL 选择 FB 的输入方式
2. 设置 PID_CR_STRSEL 选择 PID 的启动方式(CR_STRSEL=0)
3. 设置 CR_PL 选择设定 PIDx 的优先级
4. 配置相应的 KPG,KIG,KDG,KIMULP,DIV_KPD,DIV_KID,DIV_KDD,INTGLIM,OUTLIM
5. 将参考值写入 REF
6. 将反馈值写入 FB
7. 设置 PID_CR_STR 启动 PIDx

18.2.3 PID_CR（PID 控制寄存器）

PID_CR(PID 控制寄存器)			基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 00H					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	PL		TRGSEL		TRGEN	STRSEL	CLR	STR
Reset:	0	0	0	0	0	0	0	0

位	功能描述
PL	**优先级设定** 000：优先级 1(最高) 001：优先级 2 010：优先级 3 011：优先级 4 注：如果两个 PID 的优先级相同，以 PID 的排列为优先级，比如 PID1 优先级高于 PID2 优先级
TRGSEL	**Feedback 值选择** 00：当 ParkBUSY=1->0,ME1_Iq 输入到 FB 01：当 ParkBUSY=1->0,ME1_Id 输入到 FB 10： 当 ParkBUSY=1->0,ME2_Iq 输入到 FB 11：当 ParkBUSY=1->0,ME2_Id 输入到 FB
TRGEN	0: 外部触发关闭 1: 外部触发使能
STRSEL	**启动 PID 方式选择** 0:PID_CR_STR 启动 PID 1:当 FB 更新后直接启动 PID
CLR	1:清除 INTG & ERR 为 0 0:不清除 INTG & ERR
STR	1:启动 PID 运算,下一个 clock 硬件清除为 0 0:未启动 PID 运算 注：如果在 PIDx 正在运行另一个 PID 启动,需等到 PIDx 计算结束才会运行另一个 PID。 如果两个 PIDx 同时启动，将会先运行高优先级的，然后运行低优先级

18.2.4 PID_OUT（PID 输出数据寄存器）

PID_OUT（PID 输出数据寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 04H

	Bit15…Bit0
Read:	OUT[15:0]
Write:	X
Reset:	0

位	功能描述
OUT[15:0]	输出 PID 运算的结果（有符号数）

18.2.5 PID_REF（PID 参考数据寄存器）

PID_REF（PID 参考数据寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 08H
	Bit15…Bit0
Read: Write:	REF[15:0]
Reset:	0

位	功能描述
REF[15:0]	输入的 Reference 值 ERR = REF - FB.

18.2.6 PID_FB（PID 反馈数据寄存器）

PID_FB（PID 反馈数据寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 0CH
	Bit15…Bit0
Read: Write:	FB[15:0]
Reset:	0

位	功能描述
FB[15:0]	输入的 Feedback 值. ERR = REF - FB.

18.2.7 PID_ERR（PID 误差数据寄存器）

PID_ERR(PID 误差数据寄存器)	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 10H

	Bit16…Bit0
Read: Write:	ERR[16:0]
Reset:	0

位	功能描述
ERR[16:0]	PID 运算后保留的 Error 值 ERR = REF - FB.

18.2.8 PID_INTG（PID 积分累积数据寄存器）

PID_INTG（PID 积分累积寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 14H
	Bit31…Bit0
Read: Write:	INTG[31:0]
Reset:	0

位	功能描述
INTG[31:0]	PID 运算过程中，积分环节误差累加运算后保留 Integral 的值.

18.2.9 PID_KP(PID 比例放大增益/缩小寄存器)

PID_KP(PID 比例放大增益/缩小寄存器)	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 18H							
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	X	X	X	X	KPD[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	KPG[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read:	KPG[7:0]							

Write:								
Reset:	0	0	0	0	0	0	0	0

位	功能描述
KPD[19:16]	KPG 增益放大后除 2 的幂数，默认为 0 $P_{OUT} = Error \times KP \div 2^{\wedge KPD[3:0]}$
KPG[15:0]	KP Gain 值 KP 放大倍率

注：公式中 Pout 仅表示中间数据便于理解，无实际意义

18.2.10PID_KI(PID 积分放大增益/缩小寄存器)

PID_KI(PID 积分放大增益/缩小寄存器)			基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 1CH					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	KID[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	KIG[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	KIG[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
KID[19:16]	KPG 增益放大后除 2 的幂数，默认为 0 $I_{MIDOUT} = Error_{(n-1)} \times KI \div 2^{\wedge KID[3:0]}$
KIG[15:0]	KI Gain 值 KI 放大倍率

注：公式中 I_{MIDOUT} 仅表示中间数据便于理解，无实际意义

18.2.11PID_KD(PID 微分放大增益/缩小寄存器)

PID_KD(PID 微分放大增益/缩小寄存器)	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6)

			偏移地址： 20H					
	Bit31	30	29	28	27	26	25	Bit24
Read: Write:	X	X	X	X	X	X	X	X
Reset:	0	0	0	0	0	0	0	0
	Bit23	22	21	20	19	18	17	Bit16
Read: Write:	X	X	X	X	KDD[19:16]			
Reset:	0	0	0	0	0	0	0	0
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	KDG[15:8]							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	KDG[7:0]							
Reset:	0	0	0	0	0	0	0	0

位	功能描述
KDD[19:16]	KDG 增益放大后除 2 的幂数，默认为 0 $D_{OUT} = [Error_n - Error_{(n-1)}] \times \mathrm{KD} \div 2^{\wedge KDD[3:0]}$
KDG[15:0]	KD Gain 值 KI 放大倍率

注：公式中 D_{OUT} 仅表示中间数据便于理解，无实际意义

18.2.12PID_KIMULP（PID 积分末端放大增益寄存器）

PID_KIMULP（PID 积分末端放大增益寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 24H
	Bit9…Bit0
Read: Write:	KIMULP[9:0]
Reset:	0

位	功能描述
KIMULP[15:0]	KI 放大倍率 KI_OUT=PID_INTG÷KIMULP

18.2.13PID_INTGLIM（PID 积分累积数据限制寄存器）

PID_INTGLIM（PID 积分中间数据限制寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 28H
	Bit30…Bit0
Read: Write:	INTGLIM[30:0]
Reset:	0

位	功能描述
INTGLIM[30:0]	限制 Integral 的上限为 INTGLIM,下限为-INTGLIM

18.2.14PID_OUTLIM（PID 输出限制寄存器）

PID_OUTLIM（PID 输出限制寄存器）	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 2CH
	Bit14…Bit0
Read: Write:	OUTLIM[14:0]
Reset:	0

位	功能描述
OUTLIM [14:0]	限制 OUT 的上限为 OUTLIM,下限为-OUTLIM（无符号数）

18.2.15PID_IER(PID 中断寄存器)

PID_IER(PID 中断寄存器)	基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 30H							
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	IER_DMA	IER_IE
Reset:	0	0	0	0	0	0	0	0

位	功能描述
IER_DMA	PIDx 计算完成触发 DMA 请求 1:使能 0:禁止

IER_IE	PIDx 中断使能 1:使能 0:禁止

18.2.16PID_IFR(PID 中断标志寄存器)

PID_IFR(PID 中断标志寄存器)			基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 34H					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	IFR_IF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
IFR_IF	PIDx 转换完成中断标志 1:发生 PIDx 中断事件,写入 0 清除 0:无 PIDx 中断事件

18.2.17PID_SR(PID 状态寄存器)

PID_SR(PID 状态寄存器)			基地址： 0x40024000(PID1)，0x40024100(PID2)，0x40024200(PID3)，0x40024300(PID4)，0x40024400(PID5)，0x40024500(PID6) 偏移地址： 38H					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	X	X	X	SR_BUSY
Reset:	0	0	0	0	0	0	0	0

位	功能描述
SR_BUSY	1: PIDx 运算中 0: PIDx 闲置 注：PIDx_BUSY 为 1 时,(KPG,KIG,KDG,KIMULP,DIV,INTGLIM,OUTLIM)无法写入

18.3 角度变换（SIN COS Table）

18.3.1 使用方法

1. 写入 Angle 启动运算
2. 读出 Cos, Sin, IPDSin 值

18.3.2 ME_Angle（ME 输入角度寄存器）

ME_Angle（ME 输入角度寄存器）	基地址： 0x4002 5000(ME1)，0x4002 5400(ME2) 偏移地址： 0CH
	Bit31…Bit0
Read: Write:	Angle[31:0]
Reset:	0

位	功能描述
Angle[15:0]	输入角度值,范围为 0-1023（360 度分为 1024 份） 当进行坐标转换 Park 和 Rev Park 前,要先写入此寄存器

18.3.3 ME_Cos（ME 输出 Cos 数据寄存器）

ME_Cos（ME 输出 Cos 数据寄存器）	基地址： 0x4002 5000(ME1)，0x4002 5400(ME2) 偏移地址： 30H
	Bit15…Bit0
Read: Write:	Cos[15:0]
Reset:	0

位	功能描述
Cos[15:0]	根据 Angle 值,输出对应的 Cos table 值

18.3.4 ME_Sin（ME 输出 Sin 数据寄存器）

ME_Sin（ME 输出 Sin 数据寄存器）	基地址： 0x4002 5000(ME1)，0x4002 5400(ME2) 偏移地址： 34H
	Bit15…Bit0
Read: Write:	Sin[15:0]
Reset:	0

位	功能描述
Sin[15:0]	根据 Angle 值,输出对应的 Sin table 值

18.4 坐标转换

18.4.1 使用方法

1. 写入 Ia,Ib,Ic,Cos,Sin,Vq,Vd
2. ME_CR_MESTR[2:0]写入 1 启动坐标转换运算
3. 运算是否完成,可以透过 ME_SR_CDTR 来判断
4. 转换完的数值为 Ialpha, Ibeta, Iq, Id, Valpha, Vbeta

注：

1. ME_CR_MESTR[2:0]的配置

ME_CR_MESTR[2:0]	项目
0b000	不启动运算
0b001	Clarke
0b010	Park
0b011	Clarke, Park
0b100	Rev Park
0b101	Clarke, Rev Park
0b110	Park, Rev Park
0b111	Clarke, Park, Rev Park

2. 当同时启动 Clarke,Park 的转换,必须先算完 Clarke 后才能算 Park
3. 启动运算，CDTR_BUSY 由硬件置 1；运算完后，CDTR_BUSY 由硬件置 0；CDTR_BUSY 为 1 时，（Ia,Ib,Cos,Sin,Vq,Vd）无法写入值。

18.4.2 ME_Ia(自然坐标系 Ia 数据寄存器)

ME_Ia(自然坐标系 Ia 数据寄存器)	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 04H
	Bit15…Bit0
Read: Write:	Ia[15:0]
Reset:	0

位	功能描述
Ia[15:0]	Ia 值.

	Clarke 运算的输入

18.4.3 ME_Ib(自然坐标系 Ib 数据寄存器)

ME_Ib(自然坐标系 Ib 数据寄存器)	基地址: 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址: 08H
	Bit15…Bit0
Read: Write:	Ib[15:0]
Reset:	0

位	功能描述
Ib[15:0]	Ib 值. Clarke 运算的输入

18.4.4 ME_Ialpha（静止坐标系 Ialpha 数据寄存器）

ME_Ialpha（静止坐标系 Ialpha 数据寄存器）	基地址: 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址: 10H
	Bit15…Bit0
Read: Write:	Ialpha[15:0]
Reset:	0

位	功能描述
Ialpha[15:0]	Ialpha 值. Clarke 运算的输出 Park 运算的输入

18.4.5 ME_Ibeta（静止坐标系 Ibeta 数据寄存器）

ME_Ibeta（静止坐标系 Ibeta 数据寄存器）	基地址: 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址: 14H
	Bit15…Bit0
Read: Write:	Ibeta[15:0]
Reset:	0

位	功能描述
Ialpha[15:0]	Ibeta 值. Clarke 运算的输出 Park 运算的输入

18.4.6 ME_Iq（旋转坐标系 Iq 数据寄存器）

ME_Iq（旋转坐标系 Iq 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 18H
	Bit15…Bit0
Read: Write:	Iq[15:0]
Reset:	0

位	功能描述
Iq[15:0]	Iq 值. Park 运算的输出

18.4.7 ME_Id（旋转坐标系 Id 数据寄存器）

ME_Id（旋转坐标系 Id 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 1CH
	Bit15…Bit0
Read: Write:	Id[15:0]
Reset:	0

位	功能描述
Id[15:0]	Id 值. Park 运算的输出

18.4.8 ME_Vq（旋转坐标系 Vq 数据寄存器）

ME_Vq（旋转坐标系 Vq 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 20H
	Bit15…Bit0
Read: Write:	Vq[15:0]
Reset:	0

位	功能描述
Vq[15:0]	Vq 值. Rev Park 运算的输入

18.4.9 ME_Vd（旋转坐标系 Vd 数据寄存器）

ME_Vd（旋转坐标系 Vd 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 24H

	Bit15…Bit0
Read:	Vd[15:0]
Write:	
Reset:	0

位	功能描述
Vd[15:0]	**Vd 值.** Rev Park 运算的输入

18.4.10ME_Valpha（静止坐标系 Valpha 数据寄存器）

ME_Valpha（静止坐标系 Valpha 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 28H
	Bit15…Bit0
Read:	Valpha[15:0]
Write:	
Reset:	0

位	功能描述
Valpha[15:0]	**Valpha 值.** Rev Park 运算的输出 SVPWM 运算的输入

18.4.11ME_Vbeta（静止坐标系 Vbeta 数据寄存器）

ME_Vbeta（静止坐标系 Vbeta 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 2CH
	Bit15…Bit0
Read:	Vbeta[15:0]
Write:	
Reset:	0

位	功能描述
Vbeta[15:0]	**Vbeta 值.** Rev Park 运算的输出, SVPWM 运算的输入

18.5 SVPWM

18.5.1 使用方法

1. Clarke_STRSEL,Park_STRSEL,RevPark_STRSEL,SVPWM_STRSEL 选择各个模块的启动方式软件启动
2. ME_CR_PL 设置优先级

3. 设置 RevPark 的输入讯号
4. 设置 Clark,Park 运算的坐标系公式
5. 设置 Angle 是否需要加 90 或 270 度
6. 输入 Ia,Ib,Vq,Vd,Angle
7. 设置 ME_CR_MESTR 启动运算
8. 运算是否完成可透过相关标志位判断
9. 读出 PDCH1,PDCH2,PDCH3,SVZone_N,SVZone_Sector

18.5.2 ME_TPWM（PWM 通道占空比限制上限数据寄存器）

ME_TPWM（PWM 通道占空比限制上限数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 58H
	Bit15…Bit0
Read: Write:	TPWM[15:0]
Reset:	0

位	功能描述
TPWM[15:0]	限制 PDCHx 的最大值（x=1,2,3）

18.5.3 ME_LSMIN（PWM 通道占空比限制下限数据寄存器）

ME_LSMIN（PWM 通道占空比限制下限数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 5CH
	Bit15…Bit0
Read: Write:	LSMIN[15:0]
Reset:	0

位	功能描述
LSMIN[15:0]	限制 PDCHx 的最小值（x=1,2,3）

18.5.4 ME_PDCH1（PWM 通道 1 占空比数据寄存器）

ME_PDCH1（PWM 通道 1 占空比数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 3CH
	Bit31…Bit0
Read: Write:	PDCH1[31:0]
Reset:	0

位	功能描述
PDCH1[31:0]	PwmDutyCH1 值,SVPWM 运算的输出

18.5.5 ME_PDCH2（PWM 通道 2 占空比数据寄存器）

ME_PDCH2（PWM 通道 2 占空比数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 40H
	Bit31…Bit0
Read: Write:	PDCH2[31:0]
Reset:	0

位	功能描述
PDCH2[31:0]	PwmDutyCH2 值,SVPWM 运算的输出

18.5.6 ME_PDCH3（PWM 通道 3 占空比数据寄存器）

ME_PDCH3（PWM 通道 3 占空比数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 44H
	Bit31…Bit0
Read: Write:	PDCH3[31:0]
Reset:	0

位	功能描述
PDCH3[31:0]	PwmDutyCH3 值,SVPWM 运算的输出

18.5.7 ME_SVZone（PWM 扇区数据寄存器）

ME_SVZone（PWM 扇区数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 38H
	Bit15…Bit8
Read: Write:	SVZone_Sector[15:8]
Reset:	0
	Bit7…Bit0
Read: Write:	SVZone_N[7:0]
Reset:	0

位	功能描述
SVZone_ Sector[15:8]	SVPWMZone_Sector 值,SVPWM 运算的输出
SVZone_N[7:0]	SVPWMZone_N 值,SVPWM 运算的输出

18.5.8 ME_Taon（Taon 数据寄存器）

ME_Taon（Taon 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2)

	偏移地址： 48H
	Bit31…Bit0
Read: Write:	Taon[31:0]
Reset:	0

位	功能描述
Taon[15:0]	输出 Taon

18.5.9 ME_Tbon（Tbon 数据寄存器）

ME_Tbon（Tbon 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 4CH
	Bit31…Bit0
Read: Write:	Tbon[31:0]
Reset:	0

位	功能描述
Tbon[15:0]	输出 Tbon

18.5.10ME_Tcon（Tcon 数据寄存器）

ME_Tcon（Tcon 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 50H
	Bit31…Bit0
Read: Write:	Taon[31:0]
Reset:	0

位	功能描述
Tcon[15:0]	输出 Tcon

18.5.11ME_FOVM（PWM Full scale 数据寄存器）

ME_FOVM（PWM Full scale 数据寄存器）	基地址： 0x4002 5000(ME1) 0x4002 5400(ME2) 偏移地址： 54H
	Bit15…Bit0
Read: Write:	FOVM[15:0]
Reset:	0

位	功能描述

FOVM[15:0]	输入 PWM Full scale 值

18.6 ME 相关寄存器

18.6.1 ME_CR（ME 控制寄存器）

ME_CR（ME 控制寄存器）			基地址：0x40025000(ME1)，0x40025400(ME2) 偏移地址： 00H					
	Bit31	**30**	**29**	**28**	**27**	**26**	**25**	**Bit24**
Read: **Write:**	LOCK		Park_IqIbetaSymbol	Park_IqIalphaSymbol	Park_IdIbetaSymbol	Park_IdIalphaSymbol	Park_IdIqAngle	CDTR_Angle[24:23]
Reset:	0	0	0	0	0	0	0	0
	Bit23	**22**	**21**	**20**	**19**	**18**	**17**	**Bit16**
Read: **Write:**	CDTR_Angle[24:23]	Clarke_SEL	SV_FIVE	RevPark_VdSEL[20:18]			RevPark_VqSEL[17:15]	
Reset:	0	0	0	0	0	0	0	0
	Bit15	**14**	**13**	**12**	**11**	**10**	**9**	**Bit8**
Read: **Write:**	RevPark_VqSEL[17:15]	X	X	PL	SVPWM_STRSEL	RevPark_STRSEL	Park_STRSEL	Clarke_STRSEL
Reset:	0	0	0	0	0	0	0	0
	Bit7	**6**	**5**	**4**	**3**	**2**	**1**	**Bit0**
Read: **Write:**	X				MESTR[3:0]			
Reset:	0	0	0	0	0	0	0	0

位	功能描述
MESTR[3:0]	**坐标转换信道选择** BIT0:写入 1,坐标转换进行 Clarke 运算 BIT1:写入 1,坐标转换进行 Park 运算 BIT2:写入 1,坐标转换进行 Rev Park 运算 BIT3:写入 1,坐标转换进行 SVPWM 运算 下一个 clock 清除为 0
Clarke_STRSEL	**启动 Clarke 方式选择** 0:CR_MESTR 启动 Clarke 1:当 Ib 有更新后直接启动 Clarke 注：当 STRSEL=1，需先写入 Ia 的值，然后写入 Ib 的值启动 Clark 运算

Park_STRSEL	**启动 Park 方式选择** 0: CR_MESTR 启动 Park 1:当 Ibeta 有更新后直接启动 Park 注：当 STRSEL=1，需先写入 Ialpha 的值，然后写入 Ibeta 的值启动 Park 运算
RevPark_STRSEL	**启动 RevPark 方式选择** 0: CR_MESTR 启动 RevPark 1:当 Vd 有更新后直接启动 RevPark 注：当 STRSEL=1，需先写入 Vq 的值，然后写入 Vd 的值启动 RevPark 运算
SVPWM_STRSEL	**启动 SVPWM 方式选择** 0: CR_MESTR 启动 SVPWM 1:当 Vbeta 有更新后直接启动 SVPWM 注：当 STRSEL=1，需先写入 Valpha 的值，然后写入 Vbeta 的值启动 SVPWM 运算
PL	**优先级设定** 0:优先级高 1：优先级低
RevPark_VqSEL[17:15]	SVPWM 输入 Vq 选择 000：软件输入 Vq 001：当 PID1_BUSY=1->0,PID1_OUT 输入 010：当 PID2_BUSY=1->0,PID2_OUT 输入 011：当 PID3_BUSY=1->0,PID3_OUT 输入 100：当 PID4_BUSY=1->0,PID4_OUT 输入 101：当 PID5_BUSY=1->0,PID5_OUT 输入 110：当 PID6_BUSY=1->0,PID6_OUT 输入 111：保留
RevPark_VdSEL[20:18]	SVPWM 输入 Vd 选择 000：软件输入 Vd 001：当 PID1_BUSY=1->0,PID1_OUT 输入 010：当 PID2_BUSY=1->0,PID2_OUT 输入 011：当 PID3_BUSY=1->0,PID3_OUT 输入 100：当 PID4_BUSY=1->0,PID4_OUT 输入 101：当 PID5_BUSY=1->0,PID5_OUT 输入 110：当 PID6_BUSY=1->0,PID6_OUT 输入 111：保留
SV_FIVE	0: 计算 Taon 右移 1 位 1: 计算 Taon 不右移

Clarke_SEL	0:Iα=Ia,Iβ=(Ib-Ic)*18918/32768 1:Iα=Ia,Iβ=(Ic-Ib)*18918/32768
CDTR_Angle[24:23]	00:θ=θ 01:θ=θ 10:θ=θ+90° 11:θ=θ+270°
Park_IdIqAngle	0: Id:Ialpha*Sin,Ibeta*Cos Iq:Ialpha*Cos,Ibeta*Sin 1: Id:Ialpha*Cos,Ibeta*Sin Iq:Ialpha*Sin,Ibeta*Cos
Park_IdIalphaSymbol	0:Ialpha*(+1) 1:Ialpha*(-1)
Park_IdIbetaSymbol	0:Ibeta*(+1) 1:Ibeta*(-1)
Park_IqIalphaSymbol	0:Ialpha*(+1) 1:Ialpha*(-1)
Park_IqIbetaSymbol	0:Ibeta*(+1) 1:Ibeta*(-1)
LOCK	00:锁定关闭，寄存器无写保护； 01:锁定级别 1:不能写入 SV_FIVE 10:锁定级别 2:不能写入锁定级别 1 中的各位,也不能写入 Clarke_SEL,CDTR_Angle,Park_IdAngle,Park_IdIalphaSymbol,Park_IdIbetaSymbol,Park_IqAngle,Park_IqIalphaSymbol,Park_IqIbetaSymbol 11:锁定级别 3:不能写入锁定级别 2 中的各位,也不能写入 Clarke_STRSEL,Park_STRSEL,RevPark_STRSEL,SVPWM_STRSEL,RevPark_VqSEL,RevPark_VdSEL 注：在系统复位后，只能写一次 LOCK 位，一旦写入 CR 寄存器，则其内容冻结直至复位。

18.6.2 ME_IER（ME 中断使能寄存器）

ME_IER（ME 中断使能寄存器）			基地址： 0x40025000(ME1)，0x40025400(ME2) 偏移地址： 60H					
	Bit15	14	13	12	11	10	9	Bit8
Read:	X							

Write:								
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	CDTR_SVPWM_DMA	CDTR_RevPark_DMA	CDTR_Park_DMA	CDTR_Clarke_DMA	X	X	SVE	CDTR
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CDTR	**坐标转换中断使能** 1:使能 0:禁止
SVE	**SVPWM 中断使能** 1:使能 0:禁止
CDTR_Clarke_DMA	**Clarke 计算完成触发 DMA 请求** 1:使能 0:禁止
CDTR_Park_DMA	**Park 计算完成触发 DMA 请求** 1:使能 0:禁止
CDTR_RevPark_DMA	**RePark 计算完成触发 DMA 请求** 1:使能 0:禁止
CDTR_SVPWM_DMA	**SVPWM 计算完成触发 DMA 请求** 1:使能 0:禁止

18.6.3 ME_IFR（ME 中断标志寄存器）

ME_IFR（ME 中断标志寄存器）			基地址： 0x40025000(ME1)，0x40025400(ME2) 偏移地址： 64H					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	SVF	CDTRF_ReParkF	CDTRF_ParkF	CDTRF_ClarkF
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CDTRF_ClarkF	**CDTRF_Clark 中断旗标** 1:发生 CDTRF_Clark 中断事件,写入 0 清除 0:无 CDTRF_Clark 中断事件

CDTRF_ParkF	**CDTRF_Park 中断旗标** 1:发生 CDTRF_Park 中断事件,写入 0 清除 0:无 CDTRF_Park 中断事件
CDTRF_ReParkF	**CDTRF_RePark 中断旗标** 1:发生 CDTRF_RePark 中断事件,写入 0 清除 0:无 CDTRF_RePark 中断事件
SVF	**SVPWM 中断旗标** 1:发生 SVPWM 中断事件,写入 0 清除 0:无 SVPWM 中断事件

18.6.4 ME_SR（ME 状态寄存器）

ME_SR（ME 状态寄存器）			基地址： 0x40025000(ME1)，0x40025400(ME2) 偏移地址： 68H					
	Bit15	14	13	12	11	10	9	Bit8
Read: Write:	X							
Reset:	0	0	0	0	0	0	0	0
	Bit7	6	5	4	3	2	1	Bit0
Read: Write:	X	X	X	X	SVBUSY	CDTR_RevParkBUSY	CDTR_ParkBUSY	CDTR_ClarkeBUSY
Reset:	0	0	0	0	0	0	0	0

位	功能描述
CDTR_ClarkeBUSY	**CDTR_Clark 状态位** 1:CDTR_Clark 运算中 0:无 CDTR_Clark 闲置 注：ClarkeBUSY 位置起，Ia,Ib 值无法写入
CDTR_ParkBUSY	**CDTR_Park 状态位** 1:CDTR_Park 运算中 0:CDTR_Park 闲置 注：ParkBUSY 位置起，Angel,Ialpha,Ibeta 值无法写入
CDTR_RevParkBUSY	**CDTR_RePark 状态位** 1:CDTR_RePark 运算中 0:CDTR_RePark 闲置 注：RevParkBUSY 位置起，Angel,Vq,Vd 值无法写入
SVBUSY	**SVPWM 状态位** 1:SVPWM 运算中 0:SVPWM 闲置 注：SVBUSY 位置起，FOVM,TPWM,LSMIN,Valpha,Vbeta 值无法写入

19 AC/DC（待后续测试）

19.1 Absolute maximum ratings

Voltage characteristics				
Symbol	**Ratings**	**Min**	**Max**	**Unit**
VDD - VSS	External main supply voltage (including VDDA and VDD)(1)			**V**
VIN(2)	Input voltage on five volt tolerant pin			
	Input voltage on any other pin			
$\lvert\triangle V_{DDx}\rvert$	Variations between different VDD power pins			**mV**
$\lvert V_{SSX} - V_{SS}\rvert$	Variations between all the different ground pins			
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)			**KV**

1. All main power (VDD, VDDA) and ground (VSS, VSSA) pins must always be connected to the external power supply, in the permitted range.
2. VIN maximum must always be respected. Refer to Current characteristics for the maximum allowed injected current values.

Current characteristics				
Symbol	**Ratings**	**Condition**	**Max.**	**Unit**
I_{VDD}	Total current into VDD/VDDA power lines (source)(1)	VCC=3.3V		**mA**
		VCC=5V		
I_{VSS}	Total current out of VSS ground lines (sink)(1)	VCC=3.3V		
		VCC=5V		
I_{IO}	Output current sunk by any I/O and control pin	VCC=3.3V		
		VCC=5V		
	Output current source by any I/Os and control pin	VCC=3.3V		
		VCC=5V		
$I_{INJ(PIN)}$(2)	Injected current on five volt tolerant pins(3)			
	Injected current on any other pin(4)			
$SI_{INJ(PIN)}$	Total injected current (sum of all I/O and control pins)(5)			

1. All main power (VDD, VDDA) and ground (VSS, VSSA) pins must always be connected to the external power supply, in the permitted range.
2. Negative injection disturbs the analog performance of the device. See note 2. on page 76.
3. Positive injection is not possible on these I/Os. A negative injection is induced by VIN<VSS. IINJ(PIN) must never be exceeded. Refer to Voltage characteristics for the maximum allowed input voltagevalues.
4. A positive injection is induced by VIN>VDD while a negative injection is induced by VIN<VSS. IINJ(PIN) must never be exceeded. Refer to Voltage characteristics for the maximum allowed input voltagevalues.

5. When several inputs are submitted to a current injection, the maximum ΣIINJ(PIN) is the absolute sum of the positive and negative injected currents (instantaneous values).

Thermal characteristics			
Symbol	**Ratings**	**Value**	**Unit**
TSTG	Storage temperature range		**°C**
TJ	Maximum junction temperature		**°C**

19.2 General operating conditions

Symbol	Parameter	Conditions	Min	Max	Unit
f_{HCLK}	Internal AHB clock frequency	-			MHz
f_{PCLK1}	Internal APB1 clock frequency	-			
f_{PCLK2}	Internal APB2 clock frequency	-			
V_{DD}	Standard operating voltage	-			V
$V_{DDA(1)}$	Analog operating voltage (ADC or OPA or PGA or CMP not used)	Must be the same potential as VDD (2)			
	Analog operating voltage (ADC or OPA or PGA or CMP used)				
V_{BAT}	Backup operating voltage	-			
V_{IN}	I/O input voltage	Standard IO			V
		FT IO(3) VDD=3.3V			
		BOOT0			
P_D	Power dissipation at TA = 105 °C (3)	LQFP80			mW
		LQFP64			

General operating conditions (continued)

Symbol	Parameter	Conditions	Min	Max	Unit
T_A	Ambient temperature	Maximum power dissipation			°C
T_J	Junction temperature range				

1.当使用 ADC 时，参见 12.18ADC 章节。
2.建议使用相同电源为 VDD 和 VDDA 供电，在上电和正常操作期间，VDD 和 VDDA 之间最多允许有 300mv 的差别。
3.如果 T_A 较低，只要 T_J 不超过 $T_J$max(参见第 1 节)，则允许更高的 PD 值。

19.3 Operating conditions PWR UP_DOW

Symbol	Parameter	Conditions	Min	Max	Unit
t_{VDD}	V_{DD} rise time rate	VCC=3.3V			µs/V
	V_{DD} fall time rate				
	V_{DD} rise time rate	VCC=5V			

	V_{DD} fall time rate				

19.4 Embedded reset and power

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{PVD}	Programmable voltage detector level selection	PLS[2:0]=000 (rising edge)				V
		PLS[2:0]=000 (falling edge)				
		PLS[2:0]=001 (rising edge)				
		PLS[2:0]=001 (falling edge)				
		PLS[2:0]=010 (rising edge)				
		PLS[2:0]=010 (falling edge)				
		PLS[2:0]=011 (rising edge)				
		PLS[2:0]=011 (falling edge)				
		PLS[2:0]=100 (rising edge)				
		PLS[2:0]=100 (falling edge)				
		PLS[2:0]=101 (rising edge)				
		PLS[2:0]=101 (falling edge)				
		PLS[2:0]=110 (rising edge)				
		PLS[2:0]=110 (falling edge)				
		PLS[2:0]=111 (rising edge)				
		PLS[2:0]=111 (falling edge)				
$V_{PVDhyst}$(2)	PVD hysteresis	-				mV
$V_{POR/PDR}$	Power on/power down reset threshold	Falling edge				V
		Rising edge				
$V_{PDRhyst}$(2)	PDR hysteresis	-				mV
$T_{RSTTEMPO}$(2)	Reset temporization	VCC=3.3V				ms
		VCC=5V				

1. The product behavior is guaranteed by design down to the minimum VPOR/PDR value.
2. Guaranteed by design.

19.5 Embedded internal reference

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{REFINT}	Internal reference voltage	–40 °C < TA < +105 °C				**V**
		–40 °C < TA < +85 °C				
TS_vrefint(1)	ADC sampling time when reading the internal reference voltage					**us**
V_{RERINT}(2)	Internal reference voltage spread over the temperature range	VDD=3.3V				**mV**
		VDD=5V				
T_{coeff}(2)	Temperature coefficient	-				**ppm/°C**

1.Shortest sampling time can be determined in the application by multiple interations.
2.Guaranteed by design.

19.6 Supply current

Maximum current consumption in Run mode, code with data processing running from Flash

Symbol	Parameter	Conditions	fHCLK	Max(1)			Unit
				TA = 25 °C	TA = 105 °C	TA = 125 °C	
I_{DD}	Supply current in Run mode	External clock(2), all peripherals enabled，VCC=3.3V	84 MHz				mA
		External clock(2), all peripherals disabled，VCC=3.3V	84 MHz				
		External clock(2), all peripherals enabled，VCC=5V	84 MHz				
		External clock(2), all peripherals disabled，VCC=5V	84 MHz				

1. Guaranteed based on test during characterization.
2. External clock is 8 MHz and PLL is on when fHCLK > 8 MHz.

Maximum current consumption in Sleep mode, code running from Flash or RAM

Symbol	Parameter	Conditions	fHCLK	Max(1)			Unit
				TA = 25°C	TA = 105 °C	TA=125 ℃	
I_{DD}	Supply current in Sleep mode	External clock(2), all peripherals enabled，VCC=3.3V	8 MHz				uA
		External clock(2), all peripherals disabled，VCC=3.3V					
		External clock(2), all peripherals enabled，VCC=5V	8 MHz				
		External clock(2), all peripherals disabled，VCC=5V					

1. Based on characterization, tested in production at VDD max, fHCLK max with peripherals enabled.
2. External clock is 8 MHz and PLL is on when fHCLK > 8 MHz.

Typical and maximum current consumptions in Stop and Standby modes

Symbol	Parameter	Conditions	Typ(1)		MAX			Unit
			VCC = 3.3V	VCC = 5.0V	TA = 常温	TA =105 °C	TA = 125 °C	
I_{DD}	Supply current in Stop mode	Regulator in Run mode, low-speed and high-speed internal RC oscillators and high-speed oscillator OFF (no independent watchdog)						µA
		Regulator in Low-power mode, lowspeed and high-speed internal RC oscillators and high-speed oscillator OFF (no independent watchdog)						
	Supply current in Standby mode	Low-speed internal RC oscillator and independent watchdog OFF, lowspeed oscillator and RTC OFF(VCC=3.3V)						
		Low-speed internal RC oscillator and independent watchdog OFF, lowspeed						

		oscillator and RTC OFF(VCC=5V)						

1. Typical values are measured at TA = 25 °C.
2. Guaranteed based on test during characterization.

Peripheral current consumption

Peripherals		µA/MHz
APB1 (up to 36 MHz)	DMA1	
	TIM2	
	TIM3	
	TIM4	
	SPI2	
	USART2	
	USART3	
	I2C1	
	I2C2	
	CAN1	
	WWDG	
	PWR	
	BKP	

Peripheral current consumption (continued)

Peripherals		µA/MHz
APB2 (up to 72 MHz)	APB2-Bridge	
	GPIOA	
	GPIOB	
	GPIOC	
	GPIOD	
	SPI1	
	USART1	
	TIM1	
	TIM8	
	OPA	
	CMP	
	ADC1(1)	
	ADC2(1)	

1. Specific conditions for measuring ADC current consumption: f HCLK = 56 MHz, f APB1 = f HCLK /2, f APB2 =f HCLK , f ADCCLK = f APB2/4 , When ADON bit in the ADCx_CR2 register is set to 1, a current consumption ofanalog part equal to 0.65 mA must be added for each ADC.

19.7 External clock

High-speed external user clock characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
fHSE_ext	User external clock source frequency(1)	VCC=3.3V				MHz
VHSEH	OSC_IN input pin high level voltage					

VHSEL	OSC_IN input pin low level voltage					V
tw(HSE) tw(HSE)	OSC_IN high or low time(1)					ns
tr(HSE) tf(HSE)	OSC_IN rise or fall time(1)					
Cin(HSE)	OSC_IN input capacitance(1)	-				pF
DuCy(HSE)	Duty cycle	-				%
IL	OSC_IN Input leakage current	VDD=3.3V				µA
		VDD=5V				µA

1. Guaranteed by design.

HSE 4-16 MHz oscillator characteristics(1)(2)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
fOSC_IN	Oscillator frequency	-				MHz
RF	Feedback resistor	-				KΩ
C	Recommended load capacitance versus equivalent serial resistance of the crystal (RS)(3)	RS = 40 Ù				pF
i2	HSE driving current	8MHz，VDD = 3.3 V				mA
		8MHz，VDD = 5V				
gm	Oscillator transconductance	Startup				mA/V
tSU(HSE)(4)	startup time	VDD is stabilized				ms

1. Resonator characteristics given by the crystal/ceramic resonator manufacturer.
2. Guaranteed based on test during characterization.
3. The relatively low value of the RF resistor offers a good protection against issues resulting from use in a humid environment, due to the induced leakage and the bias condition change. However, it is recommended to take this point into account if the MCU is used in tough humidity conditions.
4. tSU(HSE) is the startup time measured from the moment it is enabled (by software) to a stabilized 8 MHz oscillation is reached. This value is measured for a standard crystal resonator and it can vary significantly with the crystal manufacturer

19.8 Internal clock source

High-speed internal (HSI) RC oscillator

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
f_{HSI}	Frequency	-					MHz
$DuCy_{(HSI)}$	Duty cycle	-					
ACC_{HSI}	Accuracy of the HSI oscillator	User-trimmed with the CR register(2)					
		Factory-calibrated(4)(5)	TA = –40 to 125 °C				%
$t_{su(HSI)}$(4)	HSI oscillator startup time	-					µs
$I_{DD(HIS)}$(4)	HSI oscillator power consumption	-					µA

LSI oscillator characteristics(1)

Symbol	Parameter	Min	Typ	Max	Unit

fLSI(2)	Frequency				kHz
tsu(LSI)	LSI oscillator startup time				µs
IDD(LSI)(3)	LSI oscillator power consumption				µA

Low-power mode wakeup timings

Symbol	Parameter	Voltage	min	max	Unit
tWUSLEEP	Wakeup from Sleep mode	3.3V			µs
		5V			
tWUST (1)	Wakeup from Stop mode (regulator in run mode)	3.3V			
		5V			
tWUSTDBY	Wakeup from Standby mode	3.3V			
		5V			

1. VDD = 3.3 V, TA = –40 to 105 °C unless otherwise specified.
2. Guaranteed by design.
3. Guaranteed based on test during characterization.
4. The actual frequency of HSI oscillator may be impacted by a reflow, but does not drift out of the specified range.

19.9 PLL characteristics

Symbol	Parameter	Value			
		Min(1)	Typ	Max(1)	Unit
fPLL_IN	PLL input clock(2)				MHz
	PLL input clock duty cycle				%
fPLL_OUT	PLL multiplier output clock				MHz
tLOCK	PLL lock time				µs
Jitter	Cycle-to-cycle jitter				ps

1. Guaranteed based on test during characterization.
2. Take care of using the appropriate multiplier factors so as to have PLL input clock values compatible with the range defined by f PLL_OUT

19.10 Memory characteristics

Symbol	Parameter	Conditions	Min(1)	Typ	Max(1)	Unit
tprog	Byte programming time	TA = –40 to +105 °C				µs
tERASE	Sector erase time	TA = –40 to +105 °C				ms
	Chip erase time	TA = –40 to +105 °C				

1. Guaranteed by design.

Flash memory characteristics (continued)

Symbol	Parameter	Conditions	Min(1)	Typ	Max(1)	Unit
IDD	Supply current	Read mode 40 MHz , VDD = 1.35 ~ 1.65V				mA
		Write / Erase modes VDD = 1.35 ~ 1.65V				
		Standby Current				µA

		Deep Standby Current				

1. Guaranteed by design.

Flash memory endurance and data retention

Symbol	Parameter	Conditions	Value			Unit
			Min(1)	Typ	Max	
NEND	Endurance	TA = –40 to +105 °C				kcycles
tRET	Data retention	TA = 25 °C				Years
		TA = 85 °C				
		TA = 125 °C				

1. Guaranteed based on test during characterization.

19.11 EMC characteristics

EMS characteristics

Symbol	Parameter	Conditions	Level/ Class
VFESD	Voltage limits to be applied on any I/O pin to induce a functional disturbance	VDD = 5 V, TA = +25 °C, fHCLK = 72 MHz conforms to IEC 61000-4-2	IEC3
VEFTB	Fast transient voltage burst limits to be applied through 100 pF on VDD and VSS pins to induce a functional disturbance	VDD = 5V, TA = +25 °C, fHCLK = 72 MHz conforms to IEC 61000-4-4	IEC3

EMI characteristics

Symbol	Parameter	Conditions	Monitored frequency band	Max vs. [fHSI/fHCLK]	Unit
				72 MHz	
SEMI	Peak level	VDD = 3.3 V, TA = 26 °C,55%RH,LQFP64 package compliant with IEC 61967-2	0.1 to 30 MHz		dBµV
			30 to 300 MHz		
			300 MHz to 1GHz		
			SAE EMI Level		-
		VDD = 5 V, TA = 26 °C,55%RH,LQFP64 package compliant with IEC 61967-2	0.1 to 30 MHz		dBµV
			30 to 300 MHz		
			300 MHz to 1GHz		
			SAE EMI Level		-

19.12 electrical sensitivity

Symbol	Ratings	Conditions	Class	Maximum value(1)	Unit
VESD(HBM)	Electrostatic discharge voltage (human body model)	TA = +25 °C conforming to JESD22-A114	Class 2		V
VESD(CDM)	Electrostatic discharge voltage (charge device model)	TA = +25 °C conforming to ANSI/ESD STM5.3.1	Class C5		

1. Guaranteed based on test during characterization

Electrical sensitivities

Symbol	Parameter	Conditions	Class	Maximum value(1)	Unit
LU	Static latch-up class	TA = +105 °C conforming to JESD78A	Level A		mA

19.13 IO port chracteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
VIL	Low level input voltage	All I/Os except BOOT0，DESIGN				V
		All I/Os except BOOT0，VDD=3.3V				
		All I/Os except BOOT0，VDD=5V				
VIH	High level input voltage	All I/Os except BOOT0，DESIGN				
		All I/Os except BOOT0，VDD=3.3V				
		All I/Os except BOOT0，VDD=5V				
Vhys	All IO Schmitt trigger voltage hysteresis(4)	-				mV
Ilkg	Input leakage current(6)	VIN=VDD				µA
		VIN = 0				
RPU	Weak pull-up equivalent resistor(7)	VDD=3.3V				kΩ
		VDD=5V				
RPD	Weak pull-down equivalent resistor(7)	VDD=3.3V				
		VDD=5V				
CIO	I/O pin capacitance	-				pF

1. Data based on design simulation.
2. Tested in production.
3. FT = Five-volt tolerant. In order to sustain a voltage higher than V DD +0.3 the internal pull-up/pull-down resistors must be disabled.
4. Hysteresis voltage between Schmitt trigger switching levels. Guaranteed based on test during characterization.
5. With a minimum of 100 mV.
6. Leakage could be higher than max. if negative current is injected on adjacent pins.

7. Pull-up and pull-down resistors are designed with a true resistance in series with a switchable PMOS/NMOS. This PMOS/NMOS contribution to the series resistance is minimum (~10% order) .

Output voltage characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
VOL(1)	Output low level voltage for an I/O pin when 1 pins are sunk at same time	VDD=3.3V，IIO = -8 mA			V
		VDD=5V，IIO = -8 mA			
VOH(3)	Output high level voltage for an I/O pin when1 pins are sourced at same time	VDD=3.3V，Iio = +8 mA			
		VDD=5V，Iio = +8 mA			
VOL(1)(4)	Output low level voltage for an I/O pin when 1 pins are sunk at same time	VDD=3.3V I_{io} = -20 Ma			
		VDD=5V，IIO = -20 mA			
VOH(3)(4)	Output high level voltage for an I/O pin when 1 pins are sourced at same time	VDD=3.3V，IIO = +20 mA			
		VDD=5V，IIO = +20 mA			

I/O AC characteristics(1)

MODEx[1:0] bit value(1)	Symbol	Parameter	Conditions	Min	Max	Unit
10	fmax(IO)out	Maximum frequency(2)	CL = 50pF, VDD = 3.3V			MHz
			CL = 50pF, VDD =5V			
	tf(IO)out	Output high to low level fall time	CL = 50 pF, VDD = 3.3V			ns
			CL = 50 pF, VDD =5V			
	tr(IO)out	Output low to high level rise time	CL = 50 pF, VDD = 3.3V			
			CL = 50 pF, VDD = 5V			
01	fmax(IO)out	Maximum frequency(2)	CL = 50 pF, VDD = 3.3V			MHz
			CL = 50 pF, VDD = 5V			
	tf(IO)out	Output high to low level fall time	CL = 50 pF, VDD = 3.3V			ns
			CL = 50 pF, VDD = 5V			
	tr(IO)out	Output low to high level rise time	CL = 50 pF, VDD = 3.3V			
			CL = 50 pF, VDD = 5V			
11	Fmax(IO)out	Maximum frequency(2)	CL =30 pF, VDD = 3.3V			MHz
			CL =30 pF, VDD =5V			
			CL = 50 pF, VDD = 3.3 V			
			CL = 50 pF, VDD =5V			
	tf(IO)out	Output high to low level fall time	CL =30 pF, VDD = 3.3V			ns

			CL =30 pF, VDD =5V			
			CL = 50 pF, VDD = 3.3 V			
			CL = 50 pF, VDD =5V			
	tr(IO)out	Output low to high level rise time	CL =30 pF, VDD = 3.3V			ns
			CL =30 pF, VDD =5V			
			CL = 50 pF, VDD = 3.3 V			
			CL = 50 pF, VDD =5V			
-	tEXTIpw	Pulse width of external signals detected by the EXTI controller	-			ns

19.14 NRST

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
VIL(NRST)(1)	NRST Input low level voltage	VDD=5V				V
		VDD=3.3V				
VIH(NRST) (1)	NRST Input high level voltage	VDD=5V				
		VDD=3.3V				
Vhys(NRST)	NRST Schmitt trigger voltage hysteresis	VDD=5V				V
		VDD=3.3V				
RPU	Weak pull-up equivalent resistor(2)					kΩ
VF(NRST)(1)	NRST Input filtered pulse	-				ns
VNF(NRST)	NRST Input not filtered pulse	-				ns

19.15 TIM Timer

Symbol	Parameter	Conditions	Min	Max	Unit
tres(TIM)	Timer resolution time	-			tTIMxCLK
		fTIMxCLK = 72 MHz			ns
fEXT	Timer external clock frequency on CH1 to CH4	-			MHz
		fTIMxCLK = 72 MHz			MHz
ResTIM	Timer resolution	-			bit
tCOUNTER	16-bit counter clock period when internal clock is selected	-			tTIMxCLK
		fTIMxCLK = 72			μs

		MHz			
tMAX_COUNT	Maximum possible count	-			tTIMxCLK
		fTIMxCLK = 72 MHz			s

19.16 Communications interfaces

I2C characteristics

Symbol	Parameter	Standard mode I2C(1)		Fast mode I2C(1)		Unit
		Min	Max	Min	Max	
tw(SCLL)	SCL clock low time					
tw(SCLH)	SCL clock high time					µs
tsu(SDA)	SDA setup time					
th(SDA)	SDA data hold time					
tr(SDA) tr(SCL)	SDA and SCL rise time					
tf(SDA) tf(SCL)	SDA and SCL fall time					ns
th(STA)	Start condition hold time					
tsu(STA)	Repeated Start condition setup time					µs
tsu(STO)	Stop condition setup time					ms
tw(STO:STA)	Stop to Start condition time (bus free)					ms
Cb	Capacitive load for each bus line					pF
tSP	Pulse width of spikes that are suppressed by the analog filter					ns

1.Design by Guarantee

SCL frequency (fPCLK1= 36 MHz.,VDD_I2C = 3.3 V、5V)(1)(2)

fSCL (kHz)	I2C_CCR value
	RP = 4.7 kΩ

1. RP = External pull-up resistance, fSCL = I2C speed,
2. For speeds around 200 kHz, the tolerance on the achieved speed is of ±5%. For other speed ranges, the tolerance on the achieved speed ±2%. These variations depend on the accuracy of the external components used to design the application.

SPI characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
fSCK	SPI clock frequency	Master mode			MHz
1/tc(SCK)		Slave mode			
tr(SCK) tf(SCK)	SPI clock rise and fall time	Capacitive load: C = 30 pF			ns

Min SCK H/L	Mini SCK High / Low width	Master / Slave mode			ns
tsu(NSS)(1)	NSS setup time	Slave mode			ns
th(NSS)(1)	NSS hold time	Slave mode			
tw(SCKH)(1)	SCK high and low time	Master mode, fPCLK = 36 MHz, presc = 4			
tw(SCKL)(1)					
tsu(MI)(1)	Data input setup time	Master mode			
tsu(SI)(1)		Slave mode			
th(MI)(1)	Data input hold time	Master mode			
th(SI)(1)		Slave mode			
ta(SO)(1)	Data output access time	Slave mode, fPCLK = 20 MHz			
tdis(SO)(1)	Data output disable time	Slave mode			
tv(SO) (1)	Data output valid time	Slave mode (after enable edge)			
tv(MO)(1)	Data output valid time	Master mode (after enable edge)			
th(SO)(1)	Data output hold time	Slave mode (after enable edge)			
th(MO)(1)		Master mode (after enable edge)			

1.Design by Guarantee

USART Maximum Running Baud Rate

Interface	Maximum Baud Rate
USART1	
USART2 / USART3	

19.17 CAN

Refer to **I/O current injection characteristics** for more details on the input/output alternate function characteristics (CAN_TX and CAN_RX)

Symbol	Parameter	Min	Max	Unit	Condition
fB	Bit timing			MHz	PCLK=8M,t(BS1)=4tq,t(BS2)=3tq, tq=tPCLK

19.18 ADC

ADC Chracteristics

	Parameter	Conditions	Min	Typ	Max	Unit
VDDA	Power supply	-				V
VREF+	Positive reference voltage	-				V
IVREF	Current on the VREF input pin	-				µA
fADC	ADC clock frequency	-				MHz
fs(2)	Sampling rate	-				MHz
fTRIG(2)	External trigger	fADC = 12MHz				kHz

	frequency					1/fADC
VAIN (3)	Conversion voltage range					V
RAIN(2)	External input impedance	See Equation 1 and Table 47 for details				kΩ
RADC	Sampling switch resistance	-				kΩ
CADC(2)	Internal sample and hold capacitor	-				pF
t CAL(2)	Calibration time	fADC = 12 MHz				µs
		-				1/fADC
tlat(2)	Injection trigger conversion latency	fADC = 12 MHz				µs
		-				1/fADC
tlatr (2)	Regular trigger conversion latency	fADC = 12 MHz				µs
		-				1/fADC
t s(2)	Sampling time	fADC = 12 MHz				µs
		-				1/fADC
tSTAB(2)	Power-up time	-				µs
tCONV(2)	Total conversion time (including sampling time)	fADC = 12 MHz				µs
		-				1/fADC

1. Guaranteed based on test during characterization.
2. Guaranteed by design.

RAIN max for fADC = 12 MHz(1)

Ts (cycles)	tS (µs)	RAIN max (kΩ)
1.5		
7.5		
13.5		
28.5		
41.5		
55.5		
71.5		
239.5		

ADC accuracy - limited test conditions (1) (2)

Symbol	Parameter	Test conditions	Typ	Max(3)	Unit
ET	Total unadjusted error	fPCLK2 = 72 MHz,			LSB
EO	Offset error	fADC = 12 MHz, RAIN < 10 kΩ,			
		VDDA = 3 V to 5 V			
EG	Gain error	TA = 25 °C			
ED	Differential linearity error	Measurements made after			
EL	Integral linearity error	ADC calibration			

19.19 Temperature sensor

VCC=3.3V

Symbol	Parameter	Min	Typ	Max	Unit
TL	VSENSE linearity with temperature				°C
Avg_Slope	Average slope				mV/°C
V25	Voltage at 25 °C				V
Tstart	Startup time				µs
TS_temp	ADC sampling time when reading the temperature				µs

VCC=5V

Symbol	Parameter	Min	Typ	Max	Unit
TL	VSENSE linearity with temperature				°C
Avg_Slope	Average slope				mV/°C
V25	Voltage at 25 °C				V
Tstart	Startup time				µs
TS_temp	ADC sampling time when reading the temperature				µs

19.20 CMP

Symbol	Parameter	CONDITIONS	MIN	TYP	MAX	UNITS
V_{DD5}						V
I_{CC}	Operating current	VDD=3.3V,0				uA
		VDD=3.3V,1				
		VDD=3.3V,10				
		VDD=3.3V,11				
I_{CC}	Operating current	VDD=5V,0				uA
		VDD=5V,1				
		VDD=5V,10				
		VDD=5V,11				
V_{TH}	阈值电压	V_{TH}[3:0]				V
V_{OS}	Input offset voltage					mV
响应时间	VCM=VDD/2;	I_{CC}=100u, V(INP)-V(INN)=100mV				ns
		I_{CC}=100u, V(INP)-V(INN)=-100mV				
		I_{CC}=60u,				

		V(INP)-V(INN)=100mV				
		I_{CC}=60u,				
		V(INP)-V(INN)=-100mV				
		I_{CC}=38u,				
		V(INP)-V(INN)=100mV				
		I_{CC}=38u,				
		V(INP)-V(INN)=-100mV				
		I_{CC}=26u,				
		V(INP)-V(INN)=100mV				
		I_{CC}=26u,				
		V(INP)-V(INN)=-100mV				
$V_{HY(rise)}$	Signal low to high hysteresis	0				mV
		1				
		10				
		11				
$V_{HY(fall)}$	Signal high to low hysteresis	0				mV
		1				
		10				
		11				

19.21 OPA/PGA

OPA

Symbol	Parameter	CONDITIONS	MIN	TYP	MAX	UNITS
V_{DD5}	电源电压					V
ICC	Operating current	UINT GAIN，VCC=3.3V				uA
		UINT GAIN，VCC=5V				uA
CMIR	共模輸入电压					V
$V_{OS(IN)}$	输入失调电压	VCC=3.3V				mV
		VCC=5V				mV
AV	开环增益（条件）	C_{LOAD}=25pF				dB

GBW	单位增益带宽	CLOAD=25pF/ RLOAD=4K				MHz
PM	相位裕度	C_{LOAD}=25pF/ R_{LOAD}=4K				degree
SR	压摆率	C_{LOAD}=25pF				V/usec
T_{WAKEUP}	唤醒时间 0.1%精度（Unity gain）	VCC=3.3V,C_{LOAD}=25pF Current source wakeup				us
		VCC=3.3V,R_{LOAD}=4K Current source wakeup				
		VCC=3.3V,C_{LOAD}=25pF OPA wake up Current source ready				
		VCC=3.3V,R_{LOAD}=4K OPA wake up Current source ready				
		VCC=5V,C_{LOAD}=25pF Current source wakeup				
		VCC=5V,R_{LOAD}=4K Current source wakeup				
		VCC=5V,C_{LOAD}=25pF OPA wake up Current source ready				
		VCC=5V,R_{LOAD}=4K OPA wake up Current source ready				
R_{LOAD}	电阻性负载					K-ohm
C_{LOAD}	电容性负载					pF
$V_{OUT(SAT)}$	高饱和输出电压	R_{LOAD}=4K,输入 V_{DD5}				V
	低饱和输出电压	R_{LOAD}=4K,输入 0V				
$V_{N(referred\text{-}to\text{-}input)}$		@1KHz,R_{LOAD}=4K				nV/sqrt(Hz)
		@10KHz R_{LOAD}=4K				

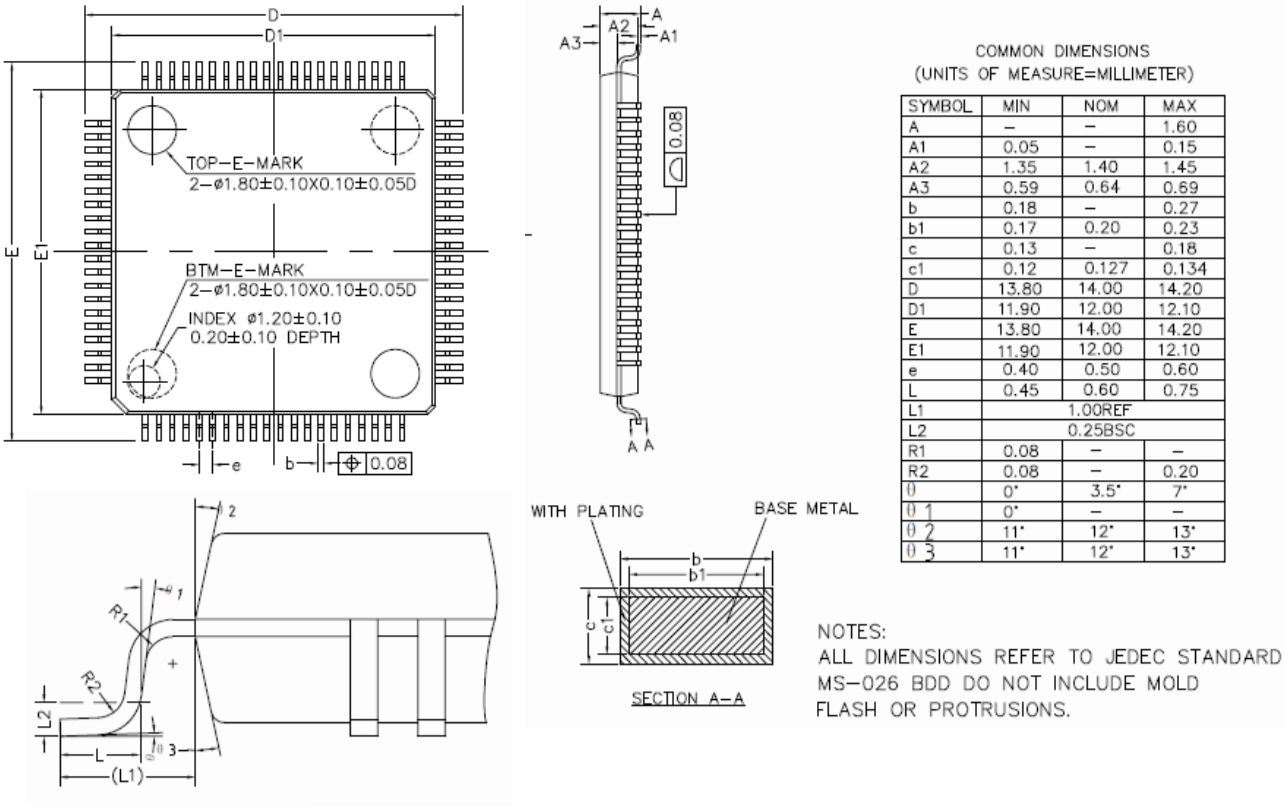
PGA

Symbol	Parameter	CONDITIONS	MIN	TYP	MAX	UNITS
V_{DD5}	电源电压					V
ICC	Operating current	Gain=32，VCC=3.3V				mA
		Gain=32，VCC=5V				

CMIR	Common mode input range	-				V
V_{OLR}	Output range					V
R_{INDIF}	Differential Input impedance					K-ohm
T_{ST}	Settling time	1% of final value (Cload=10pF)				ns
Av	Amplifier gain					V/V
PGA gain error	PGA gain error					%

20 封装

LQFP80



COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

SYMBOL	MIN	NOM	MAX
A	–	–	1.60
A1	0.05	–	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	–	0.27
b1	0.17	0.20	0.23
c	0.13	–	0.18
c1	0.12	0.127	0.134
D	13.80	14.00	14.20
D1	11.90	12.00	12.10
E	13.80	14.00	14.20
E1	11.90	12.00	12.10
e	0.40	0.50	0.60
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	–	–
R2	0.08	–	0.20
θ	0°	3.5°	7°
θ1	0°	–	–
θ2	11°	12°	13°
θ3	11°	12°	13°

NOTES:
ALL DIMENSIONS REFER TO JEDEC STANDARD MS-026 BDD DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.